University of Windsor
Scholarship at UWindsor

Electronic Theses and Dissertations

1991

High performance VLSI circuit techniques.

Sami. Bizzan
University of Windsor

Follow this and additional works at: http://scholar.uwindsor.ca/etd

Recommended Citation
Bizzan, Sami., "High performance VLSI circuit techniques." (1991). *Electronic Theses and Dissertations.* Paper 2296.

This online database contains the full-text of PhD dissertations and Masters' theses of University of Windsor students from 1954 forward. These documents are made available for personal study and research purposes only, in accordance with the Canadian Copyright Act and the Creative Commons license—CC BY-NC-ND (Attribution, Non-Commercial, No Derivative Works). Under this license, works must always be attributed to the copyright holder (original author), cannot be used for any commercial purposes, and may not be altered. Any other use would require the permission of the copyright holder. Students may inquire about withdrawing their dissertation and/or thesis from this database. For additional inquiries, please contact the repository administrator via email (scholarship@uwindsor.ca) or by telephone at 519-253-3000ext. 3208.

National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

NL-339 (r. 88/04) c

Canada

HIGH PERFORMANCE VLSI CIRCUIT TECHNIQUES

by

Sami Bizzan

A Thesis

Submitted to the Faculty of Graduate Studies through the

Department of Electrical Engineering in Partial Fulfillment

of the Requirements for the Degree of

Master of Applied Science at the

University of Windsor

Windsor, Ontario, Canada

December, 1991

National Library of Canada
Bibliothèque nationale du Canada

Canadian Theses Service
Service des thèses canadiennes

Ottawa, Canada
K1A 0N4

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-72802-7

Canadä

Sami Bizzan 1991

© All Rights Reserved

ABSTRACT

In this thesis a novel analytical approach to sizing NFET chains is presented. The technique allows a delay-area curve to be easily obtained, allowing trade-off decisions by the circuit designer. If the NFET chain is limited by a maximum gate width, the problem of optimum sizing is transformed to a single-variable optimization procedure, thus saving a substantial amount of CPU time. A unique channel resistance approximation method for use in the transistor sizing problem is presented which yields improved results over prior published methods. This thesis also develops a simple model to enable sizing of the precharge PFET; acceptable results are obtained. Finally, some problems inherent with dynamic logic structures are discussed and some possible remedies are suggested.

ACKNOWLEDGEMENTS

Thank you GOD for giving me the chance to be alive

I would like to express my sincere thanks and appreciation to **Dr. G. A. Jullien** and **Dr. W. C. Miller** for their tremendous support and guidance throughout the progress of this thesis. I would also like to thank **Bruce Erickson** and the VLSI research group members for their valuable help when it was needed. I would like to thank my wife for her encouragement and patience during the hard times of this research. Last but not the least, I would like to express my appreciation to my father, mother, brothers, and sisters for giving me the moral support needed to complete this work.

TABLE OF CONTENTS

ABSTRACT iv

ACKNOWLEDGEMENTS v

TABLE OF CONTENTS vi

LIST OF FIGURES x

LIST OF TABLES xiv

Chapter 1: Introduction 1

1.1 Introduction 1

1.2 Research Objectives 3

1.3 Thesis Organization 4

Chapter 2: CMOS Logic Gates 6

2.1 Introduction 6

2.2 Logic Gate Delay Characterization 6

2.2.1 FET Delay 6

2.2.2 Inverter Delay 8

2.2.3 Switching Node Delay 9

2.3 Static Versus Dynamic Gates 10

2.3.1 CMOS Static Gate Structure 10

2.3.2 CMOS Dynamic Gate Structure 11

2.3.3 Comparison between Static and Dynamic Gates 13

2.4 Worst Case Methodology 15

2.5 RC Chain and Elmore's Delay Formula 15

2.6 Monotonicity of the Delay with Area 21

2.7 Accuracy of Elmore's Delay Formula 22

Chapter 3: Transistor Sizing Approaches for NFET Chains 25

3.1 Introduction 25

3.2 Discharge Delay of an NFET Chain 27

3.2.1 Precharge Cycle 28

3.2.2 Evaluation Cycle 29

3.3 RC Model 30

3.4 Parasitic Capacitance Approximation 33

3.5 Channel Resistance Approximation 36

3.5.1 Channel Resistance Approximation for Linear Region NFETs 37

3.5.2 Channel Resistance Approximation for Saturation Region NFET 38

3.6 A Survey of Optimization Approach to NFET Sizing 39

3.6.1 Optimization I 40

3.6.2 Optimization II 41

3.7 Analytical Approach to NFET Sizing 42

3.7.1 Analytical Approach 43

3.7.2 Single Variable Optimization 44

3.8 Results and Discussions 45

3.9 Summary 50

Chapter 4: Practical Aspects of Circuit Sizing 52

4.1 Introduction 52

4.2 Precharge PFET Sizing 53

4.2.1 Precharge Delay Model 53

4.2.2 Precharge Channel Resistance Approximation 57

4.2.3 PFET Sizing Algorithm 58

4.2.4 Precharge PFET Sizing Results and Discussions 59

4.3 Charge Sharing 63

4.3.1 Charge Sharing Problem 63

4.3.2 Transistor Sizing and Charge Sharing 64

4.3.3 Remedies to the Charge Sharing Problem 67

4.4 Summary 70

Chapter 5: Conclusions and Further Research 71

5.1 Introduction 71

5.2 Conclusions 71

5.3 Further Research 73

REFERENCES 74

Appendix A: 3μ-DLM Technology SPICE Parameters and RC Model Computed Parameters 77

A.1 SPICE Transistor Parameters 78

A.2 RC Model Computed Parameters 79

Appendix B: NFET Chain Sizing Source Codes 81

B.1 OPTIMIZATION I Source Code 82

B.2 OPTIMIZATION II Source Code 92

B.3 SINGLE-VARIABLE OPTIMIZATION Source Code 101

B.4 ANALYTICAL APPROACH Source Code 111

Appendix C: Precharge PFET Sizing Source Codes 120

C.1 Precharge Delay Approximation Source Code 121

C.2 Precharge PFET Sizing Source Code 131

Appendix D: A Report on the Design, Layout, and Testing of 4-bit Parallel Full Adder 141

D.1 Introduction 142

D.2 Circuit Design 142

D.2.1 Latches 142

D.2.2 Switching Graph Method 143

D.3 Full Adder Layout 150

D.3.1 Initial Full Adder Cell Layout 150

D.3.2 Full Adder Cell Manual Transistor Sizing 152

D.3.3 Full Adder Cell with P-type Latch 155

D.3.4 N-P Latch and P-N Latch Layouts 158

D.4 4-bit Parallel Full Adder Layout 163

D.5 Proposed Testing Scheme 166

D.6 Chip Testing Results 166

D.7 Conclusions 168

References 169

VITA AUCTORIS 170

LIST OF FIGURES

2.1 NFET and its Model for TFET Calculation 7

2.2 FET Delay Versus Width 8

2.3 CMOS Static Inverter 8

2.4 Output waveforms of the Static Inverter 9

2.5 Logic Block 9

2.6 Output Waveforms of Switching Node 9

2.7 Static Gate Structure 11

2.8 Dynamic Gate Structure 12

2.9 RC chain circuits 17

2.10 7-RC-Link Discharge Waveforms 23

2.11 Discharge Delay Versus No. of RC Links 24

3.1 Typical CMOS Dynamic Gate Chain 27

3.2 Dynamic Gate Timing Diagram 28

3.3 Long Precharge State 29

3.4 Worst Case Discharge State 30

3.5 (a) Dynamic Gate Chain (b) RC Model 31

3.6 CMOS Transistor Capacitance Model 34

3.7 Standard Transistor Layout 35

3.8 Test Circuit 38

3.9 Delay VS Width for the Test Circuit 39

3.10 Delay VS Area 47

3.11 Discharge Delay VS Chain Height 48

3.12 Total Chain Area VS Chain Height 49

3.13 RC model delay VS SPICE delay (for sized NFET chains) 50

4.1 Precharge Delay Model 53

4.2 (a) Dynamic Gate Chain (b) RC Model (c) Precharge Delay Model 56

4.3 Precharge Test Circuit 57

4.4 Precharge Delay Versus Chain Height 61

4.5 Precharge Delay Versus Width 62

4.6 N-logic Block Dynamic Logic Structure 64

4.7 6 input NAND Dynamic Structure 65

4.8 Worst Case Precharge and Evaluation of the Circuits in Table 4.3 67

4.9 Worst Case Charge Sharing Condition of the Circuits in Table 4.3 67

4.10 Worst Case Precharge and Evaluation of the Circuit (C) with and without Precharge Distribution 69

4.11 Worst Case Charge Sharing Condition of the Circuit (C) with and without Precharge Distribution 70

D.1 Single Phase Dynamic Latches(a) P-logic Latch (b) N-logic Latch 143

D.2 Sum Binary Switching Tree 145

D.3 Reduced Sum Binary Tree 146

D.4 Resulting Switching Graphs 147

D.5 Two Possible Realizations of the Sum 147

D.6 Carry Realization 148

D.7 Circuit Schematic for One Bit Full Adder 149

D.8 One Bit Full Adder Cell Layout 151

D.9 Final One Bit Full Adder Cell Layout 153

D.10 SPICE simulation for the Final One Bit Full Adder Cell Layout 154

D.11 One Bit Full Adder Cell Layout with P-logic Latch for the Carry 156

D.12 SPICE simulation of the One Bit Full Adder Cell Layout with P-logic Latch 157

D.13 N-P Latch Layout 159

D.14 N-P Latch SPICE Simulation 160

D.15 P-N Latch Layout.. 161

D.16 P-N Latch SPICE Simulation.. 162

D.17 Circuit Schematic for 4-bit Parallel Full Adder.. 164

D.18 4-bit Parallel Full Adder Layout.. 165

D.19 Test Vector Timing Diagram.. 166

D.20 Input/Output Scope Plot of the Full Adder Cell.. 167

LIST OF TABLES

2.1 Comparison between Dynamic and Static gates.. 14

3.1 Glossary Table... 45

4.1 Precharge Test Circuit Time Constants 58

4.2 Precharge PFET sizing example results... 63

4.3 Sample Circuit Sizes and the Predicted Charge Sharing Problem Severity............ 66

D.1 Sum and Carry Truth Table... 144

D.2 Logical Test Results of the Full Adder Cell... 168

CHAPTER 1

Introduction

1.1 Introduction

Integrated circuit technology has made tremendous impact on our society in the last few years, in areas ranging from consumer products to business management to manufacturing control. The economic and strategic rewards gained by the use of very large scale integration, *VLSI*, have motivated the relentless search for high performance and perfection. Data processing speed, silicon area, power consumption, reliability, and testability of the VLSI end product, commonly referred to as a *chip*, have been the focus of many research activities in recent years. One of the most important aspects of chip performance is the rate at which data is being processed. Some research activities have targeted the fabrication technology itself to allow the designer to produce fast operating circuit structures as in the development of BiCMOS (bipolar and MOSFET technologies on the same chip) technology. Others have targeted the circuit design techniques within the fabrication technology and they have discovered numerous methods such as parallel structures, pipelined systems, dynamic logic blocks, and single phase clocking schemes.

At the transistor level, some scholars have been dealing with the problem of transistor size selection and transistor layout shape for increased speed and/or decreased area of the circuit under consideration. To reduce parasitic capacitances, and thus increasing circuit performance, many layout techniques are practiced, such as node merging and doughnut shaped transistor layout [8]. Transistor sizing techniques have played a major role in circuit design and layout to meet the never ending demands on circuit performance. Recently, it has been shown that 10% of the delay and 30% of the gate area can be decreased simultaneously by transistor sizing techniques [15] and also as much as 60%-90% increase in clocking speed alone has been reported [22], pushing the limits of the given fabrication technology.

A method to estimate circuit delay is essential in the transistor sizing problem and, certainly, SPICE [11] is the standard approach commonly used thus far. Indeed, AT&T bell laboratories have developed a transistor sizing program, ADVICE, which is used intensively and utilizes SPICE for the circuit delay approximation [17]. SPICE is a general purpose circuit simulation program that not only approximates the circuit delay but also solves for all node voltages and implements iterative procedures for the numerical integrations needed for the transient analysis. SPICE, therefore, is not suitable for the transistor sizing problem due to the large amount of CPU time it consumes each time a circuit delay is required to be evaluated. Time efficient methods for circuit delay approximations have been developed recently such as the RC tree [13] and macromodeling [9]. The RC tree is well suited for the transistor sizing problem since it deals with individual transistors rather than blocks of them, as is the case with macromodeling. There are some known problems associated with the RC tree method such as inaccurate handling of pass transistors, large delay error when considering short transistor chains, difficulties in modeling some of the logic structures, and difficulties in approximating its elements. CRYSTAL [12], TILOS [6], and TMODS [18] are some of the delay simulators or timing

verification programs that has been based on the RC tree method and used intensively in the VLSI design process.

Recent trends in the transistor sizing problem is the use of some type of heuristic optimization algorithms along with the RC tree method to alleviate some of the problems sighted earlier and this was done in SLOP [22] (transistor sizing program based on TMODS). This approach is proven to have a great success in circuit speed optimization for low transistor count circuits. The iterative nature with this approach and most optimization algorithms limit their use in large problems and this enhances the need of some type of analytical formulation for the transistor sizing problem.

A literature search has uncovered the fact that most, if not all, analytical approaches to transistor sizing problem are based on semiconductor device non-linear equations and parameters that make them applicable to only a few simple circuits such as inverters and buffers. In this thesis, a unique analytical formulation is introduced which is based on the RC model and the circuit topology, rather than the transistor equations; this approach alleviates the problems encountered with large transistor count circuits. The dream of obtaining global optimum transistor sizing for chips is closer than ever with this approach. Other aspects of this approach is the possibility of integrating the following;

1. Layout techniques such as node merging, different transistor layout shapes, floor plans .... etc.

2. Physical limitations on circuit behavior such as charge sharing, clock skew, and coupling capacitances.

3. Automated logic realization techniques such as binary switching trees being investigated by the VLSI research group at the University of Windsor.

1.2 Research Objectives

The main purpose of this work is to study and investigate available transistor sizing techniques and possibly devise and formulate more efficient methods. Ultimately, this research is targeted to integrate with other circuit design and layout techniques being developed by the VLSI research group at the University of Windsor.

1.3 Thesis Organization

Chapter 2 covers the pre-requisite material needed to understand and follow subsequent discussions. In particular, a concise review of the fundamental ideas relating to a circuit delay definition, logic gate types and operating mechanisms, worst case methodology, and RC chain delay approximation is presented. This chapter also provides a background for the definitions and notations used throughout the thesis.

The main theme and contributions of this research is contained in chapter 3. The RC model and its application to CMOS dynamic gates is given in this chapter and detailed explanations of MOSFET parasitic capacitances and channel resistances approximations are described. Transistor sizing techniques using optimization algorithms and the RC model are discussed for comparison purposes. In this chapter, a novel analytical approach based on the RC model is introduced and several sizing techniques for series transistors are described. This is followed by detailed discussions of the results of these various sizing techniques which give some insight into their advantages and disadvantages.

Chapter 4 presents a precharge delay model based on the RC model and develops an algorithm to size the Precharge PFET. Chapter 4 also contains discussions and suggests solutions to the inherent problem of charge sharing or node coupling of a typical dynamic structure. Chapter 5 contains thesis conclusions and introduces suggestions for further research activities around the topics investigated in this thesis.

Appendix A contains Northern Telecom Canada 3μ-DLM fabrication technology parameters that are used in SPICE simulations and in the transistor sizing techniques and also lists some computed parameters for the RC model implementations. FORTRAN source codes, that employ various transistor sizing techniques for a single NFET dynamic chain, discussed in chapter 3, are given in Appendix B and these codes run on Sun Microsystems SPARC station 2. Precharge delay approximation and precharge PFET sizing FORTRAN source codes are given in Appendix C. Finally, Appendix D contains a report on the design, layout, and testing of a four bit parallel full adder test circuit that has been optimized manually for high clocking speeds. This report is added to this thesis because it was the motivation of this research and to show the difficulties encountered if manual transistor sizing is sought even for such a small circuit. Other aspects of the report is the implementation of single phase clocking scheme with dynamic latches and the logic function realization using binary switching trees (currently being investigated by the VLSI research group at the University of Windsor).

CHAPTER 2

CMOS Logic Gates

2.1 Introduction

Various topics deemed essential in subsequent discussions are reviewed in this chapter. Some simplifying assumptions and definitions about circuit operating mechanisms are given here to facilitate further development of the transistor sizing techniques.

2.2 Logic Gate Delay Characterization

The basic definition of delay which is used widely in logic gate design is the time lapse between the input occurrence and the output occurrence. This definition can be interpreted in many ways according to the type of circuit under consideration and, in the following sub-sections, we detail some more specific commonly used definitions.

2.2.1 FET Delay

A measure of the performance of a FET as a switching device is determined by the time it takes to discharge its own parasitic capacitance [16]. In the *switched on* mode, the FET channel resistance is approximated by,

$$\frac{VDD}{I_{Dmax}\,(V_G{=}VDD, V_D{=}VDD)}$$

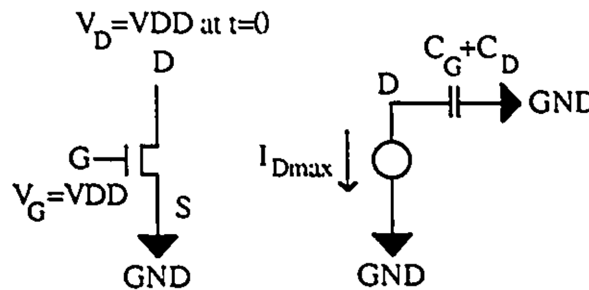

Figure 2.1 NFET and its Model for T_{FET} Calculation

Figure 2.1 shows an NFET and its conceptual model for calculating T_{FET}. The parasitic capacitance of the FET depends on the way the FET is used, but typically is given by:

$$C = C_G + C_D$$

where C_G is the gate capacitance and C_D is the drain diffused island capacitance. The time constant of the FET is then defined by:

$$T_{FET} = \frac{VDD}{I_{Dmax}}\,(C_G + C_D) \qquad (2.1)$$

Figure 2.2 shows FET delay, T_{FET}, versus FET width, W, with constant channel length. This delay definition is useful in developing initial ideas about the technology at hand; in other words, it usually answers the question whether transistor sizing enhances circuit performance or not. This delay definition, however, is not useful in the transistor sizing problem since, in a circuit structure, the circuit delay could be more sensitive to the FET parasitic capacitance or the channel resistance than the product of both; this will be demonstrated in chapter 3.

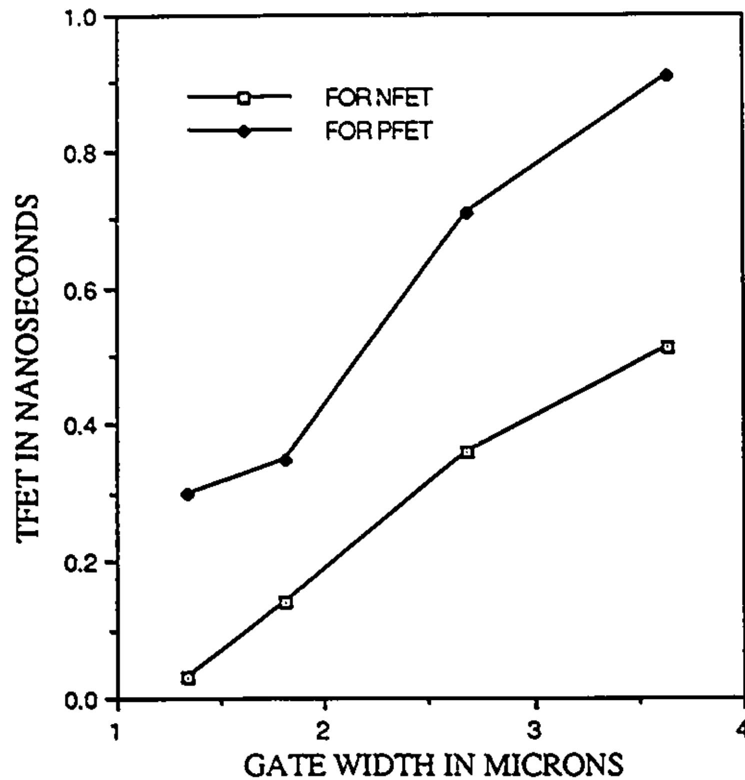

Figure 2.2 FET Delay Versus Width

2.2.2 Inverter Delay

Inverters are considered to be the basic logic units in the family of functional logic blocks and their delay often used as a delay unit for other more complex logic structures. Figure 2.3 shows a typical CMOS static inverter and Figure 2.4 shows input/output waveforms.

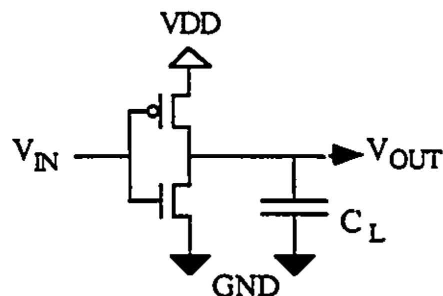

Figure 2.3 CMOS Static Inverter

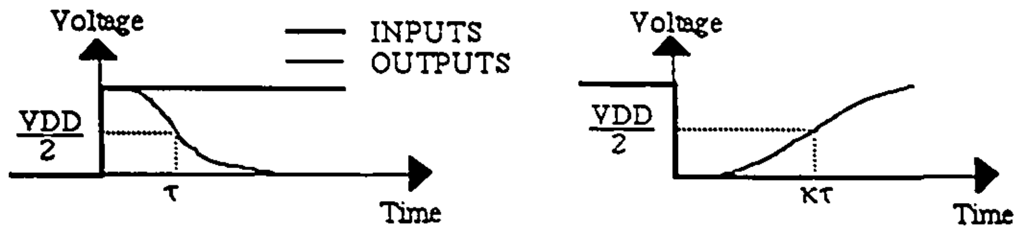

Figure 2.4 Output waveforms of the Static Inverter

The inverter pulldown delay is τ and its pullup is $k\tau$ where k is a multiplicative factor to account for the difference in delays. In a pipelined system, it is desirable to match the inverter delays; the size of the PFET is selected such that its pullup is equal to the inverter pulldown. τ is specified as the time the inverter output voltage takes to cross some reference voltage such as $\frac{VDD}{2}$ [10].

2.2.3 Switching Node Delay

This delay is used intensively in this thesis and is defined as the time the output node voltage takes to drop to 36% of its initial value or to rise to 64% of its final value; it is measured from the application of the input vector.

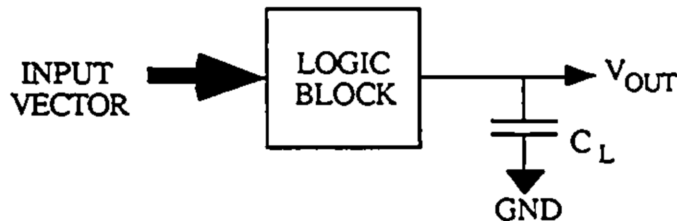

Figure 2.5 Logic Block

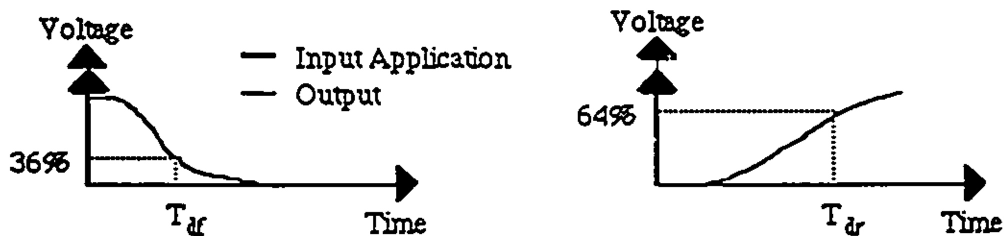

Figure 2.6 Output Waveforms of Switching Node

In complex logic structures, the circuit topology is not symmetrical to all bits of the input vector and, therefore, different input vectors will result in different delays. Usually the worst case condition that produces the longest delay is considered for system timing purposes. It is worth mentioning that this delay definition originates from single time constant linear circuits and is useful when considering linear models with dominant time constants for the logic block; this will be shown in chapter 3.

2.3 Static Versus Dynamic Gates

The CMOS static gate logic family has been the most popular for VLSI fabrication over the last decade. One of its major advantages is the very small static power consumption (when the logic gate output is not changing state). A disadvantage is the silicon area required to implement the two complementary FET blocks, that are used in the CMOS static structure. In regularly clocked systems of gates, there is never the requirement that gates hold their input/output logic levels indefinitely. In such systems, it is possible to replace the p-channel FET block by a single transistor and form a gate that functions dynamically (the p-FET is clocked rather than used with static logic levels). The advantages are reduced area and reduced self load capacitance of the logic gate. In general CMOS dynamic gates offer good area efficiency with high speed operation and moderate power consumption.

We will compare typical structures of static and dynamic CMOS gates in the following sub-sections.

2.3.1 CMOS Static Gate Structure

Figure 2.7 shows a typical static gate structure; it consists of two logic block types; n-logic which is made of n-channel MOSFETs and p-logic which is made of p-channel MOSFETs. The output node lies between these two logic blocks and VDD is connected to the top of the

p-logic block while GND is connected to the bottom of the n-logic block. Note that the same input vector is applied to both blocks.

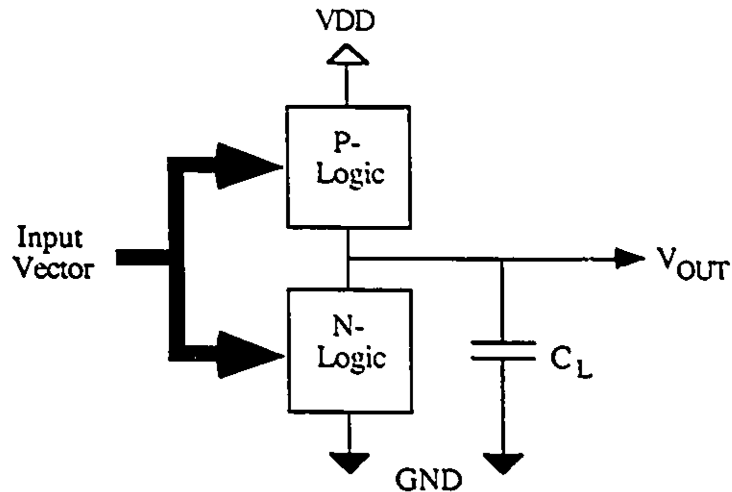

Figure 2.7 Static Gate Structure

It is always the case, for static gates to operate properly, that the p-logic block complements the n-logic block and the n-logic block complements the p-logic block. If an input vector results in a connection between VDD and the output node through the p-logic block then the n-logic block must isolate the output node from GND and the reverse is true. This mechanism ensures that no path exists between VDD and GND (if the output node is not changing), and thus static power is limited to leakage currents only; these are very small in normal operating environments. Dynamic power consumption is due to the migration of charges from VDD to the output node to GND and is proportional to the switching rate of the static gate and to the node capacitances between VDD and GND.

2.3.2 CMOS Dynamic Gate Structure

Dynamic gates differ in their structures depending on the type and technique of mapping the logic functions. Figure 2.8 shows two CMOS dynamic gate types; n-logic and p-logic. We restrict ourselves to the n-logic since the p-logic works in a similar manner.

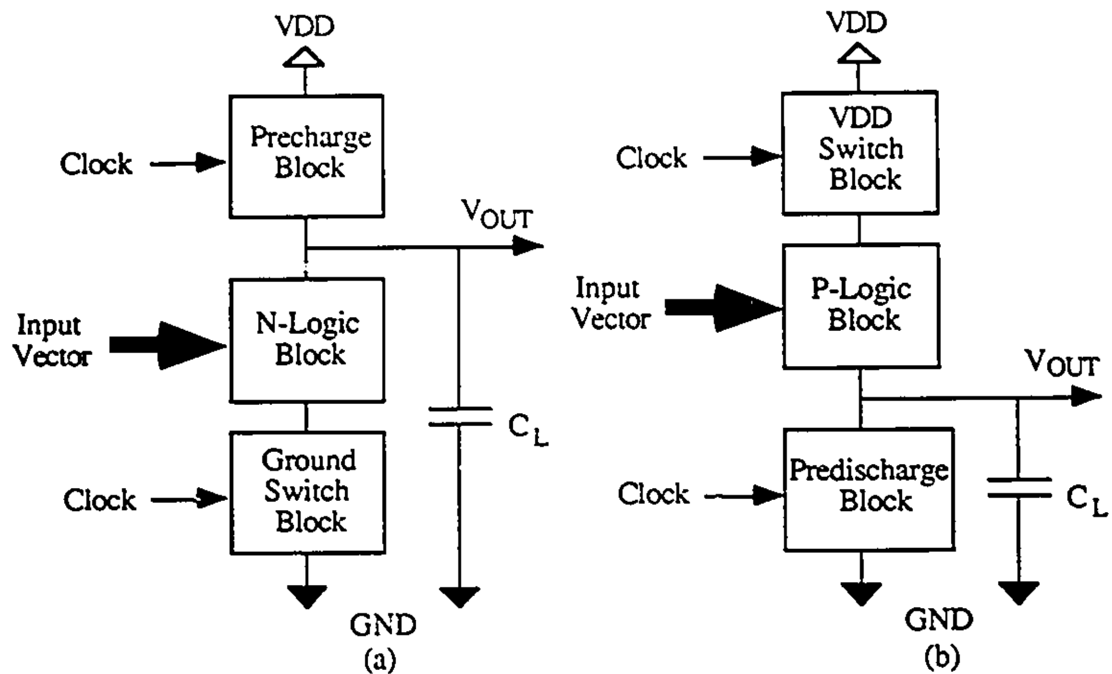

Figure 2.8 Dynamic Gate Structure

The dynamic gate uses a clocked load with a ground switch to prevent the n-logic block from producing static current drain when the load is being clocked. The dynamic gate structure is shown in Figure 2.8(a). The precharge block, which is usually made of a single PFET, connects the evaluation node to VDD when the clock is low while the ground switch, a single NFET, isolates the circuit from GND. This phase of the clock is called the precharge cycle and results in a charged high evaluation node. After the input settles, the clock makes a low to high transition and the precharge block isolates the evaluation node from VDD while the ground switch provides a path to GND for the n-logic block. This phase of the clock is called the evaluation cycle. During this part of the cycle, the n-logic block, which is made of interconnections of NFETs, either discharges or keeps high the evaluation node according to the input vector. The power consumption of a dynamic gate is roughly proportional to the clock frequency, especially when the evaluation node has to stay low or has to make transitions, and to internal node capacitances. It is very beneficial

to a circuit designer to have an expression that estimates or provides an upper limit of the power consumption of a dynamic gate. We can write,

$$P_{dynamic\ gate} = f_{clock} \cdot C_{internal} \cdot VDD^2 \qquad (2.2)$$

where f_{clock} is the clocking frequency of the dynamic gate and $C_{internal}$ is the summation of all internal node capacitances involved in the precharge and evaluation cycles. Equation (2.2) gives the worst case power consumption since (1) the Cinternal does not necessarily discharge each period of the clock and (2) most of the internal node capacitances are not likely to be fully charged in the precharge cycle. Approximation of CMOS switching circuit node capacitances will be discussed in chapter 3 and more detailed explanations of specific dynamic gate families, such as domino, NORA, and zipper, can be found in [16].

2.3.3 Comparison between Static and Dynamic Gates

Table 2.1 summarizes the advantages of each gate type over the other. The first column lists the VLSI circuit aspect and the second column states which gate provides best performance in this category. The third column contains comments rationalizing the choice made in the second column. It is important to note that the order in which the circuit aspects are listed do not reflect their importance in the circuit design and, in fact, the type of application in which the circuit is being used will dictate the order of priority of each circuit aspect.

Circuit Aspect	Gate Type	Comments
Data Processing Speed	Dynamic	Dynamic gates are faster than their counterpart static gates because of the following (for n-logic); (1) Pullup is done during the precharge cycle rather than the evaluation cycle. (2) Reduced output node capacitance and reduced overlap current.
Power Consumption	Static	In general, static gates consume less power than dynamic gates since the latter has to be precharged each cycle.
Layout Area	Dynamic	Dynamic gates occupy less area since they only require one logic block.
Transistor Count	Dynamic	The same explanation as above.
Noise Immunity	Static	The output node of static gates is solidly connected to either VDD or GND while in dynamic gates the output node is left floating. This makes the dynamic gate more susceptible to circuit noise such as voltage surges with coupling capacitances and/or leakage currents.
Input Vector Loading	Dynamic	The input vector is driving one logic block in dynamic gates and two logic blocks in static gates.
Gate Clocking	Static	Static gates do not need internal clocking which makes them easier to design and layout. Dynamic gates need internal clocking to control the precharge/discharge and the evaluation cycles.

Table 2.1 Comparison between Dynamic and Static gates

2.4 Worst Case Methodology

It is common practice in the VLSI design process to adopt a worst case methodology in order to ensure that the designed system will not fail under normal processing and operating conditions. In this methodology, system processing parameters, input vector, output loading, and operating environment are set to such worst case conditions; the design is then modified to operate under these conditions. It is important to note that some of the factors affecting the system performance might be dependent on each other and their worst case conditions should be considered collectively. Care must be taken in applying the worst case methodology so as not to be too pessimistic; this can result in an expensive or redundant design.

In this thesis, the worst case methodology is adopted and applied in approximating the discharge and precharge delays of typical dynamic logic gates. This helps in determining the clocking speed of the logic gate at which logic computation failure will not occur.

2.5 RC Chain and Elmore's Delay Formula

In this section, we discuss a useful method of approximating delays of an RC chain in closed form [16]. A detailed proof for the formula is obtained by considering the two-stage cascaded RC chain first and then generalized to the N-stage RC chain which was first derived in [5].

With reference to Figure 2.9(a), the problem can be stated as follows. At t=0, capacitors C_0 and C_1 are both charged to VDD, and the circuit begins to discharge. The objective is to find a simple closed-form formula that includes R_0, R_1, C_0, and C_1 that gives the time

needed for node 1 voltage $V_1(t)$ to drop from VDD to $\frac{VDD}{e}$, where e is the base of the natural logarithm. The result is given in equation. (2.3).

$$T_D = R_0(C_0+C_1)+R_1C_1=R_0C_0+(R_0+R_1)C_1 \tag{2.3}$$

and the proof follows. The circuit equations satisfied by node 0 voltage $V_0(t)$ and node 1 voltage $V_1(t)$ are

$$C_0\frac{dV_0}{dt}=\frac{V_1-V_0}{R_1}-\frac{V_0}{R_0} \tag{2.4}$$

$$C_1\frac{dV_1}{dt}=-\frac{V_1-V_0}{R_1}$$

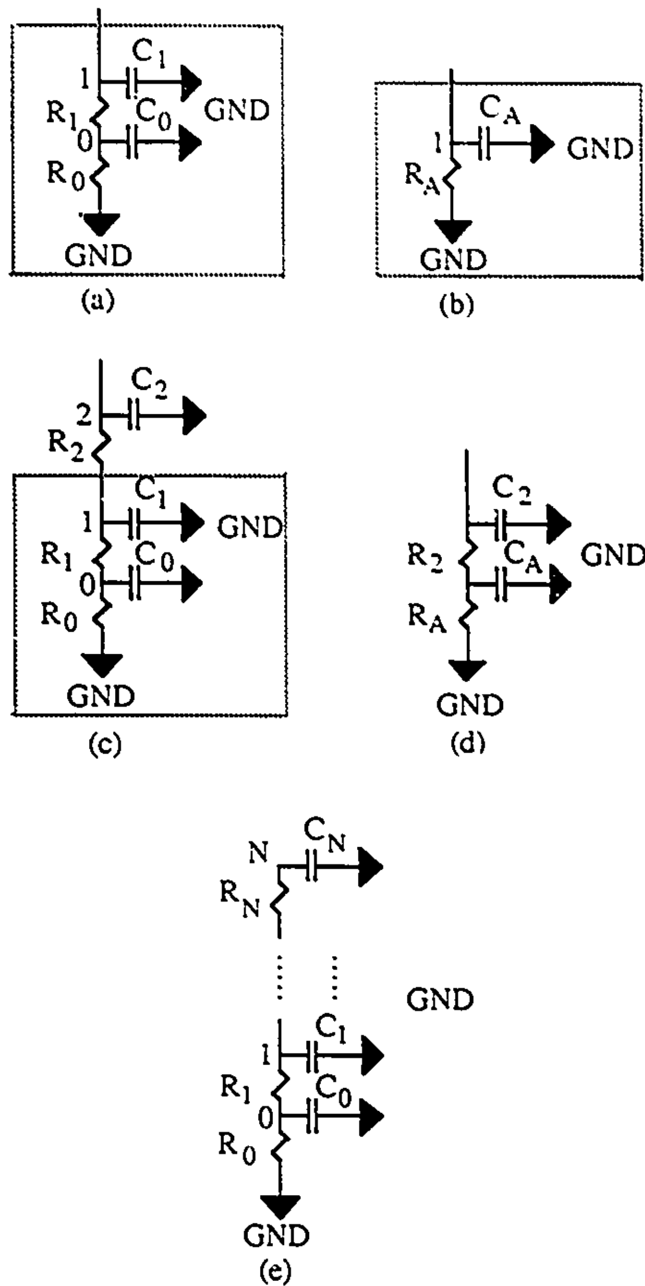

Figure 2.9 RC chain circuits

These simultaneous equations can be solved by the standard method of circuit theory: We replace the operator d/dt by s and we seek the equation satisfied by s.

We have

$$s^2 - \frac{C_0+C_1+(R_1/R_0)C_1}{R_1C_0C_1}s + \frac{1}{R_0R_1C_0C_1} = 0 \tag{2.5}$$

If α and β are the roots, they satisfy

$$\alpha + \beta = \frac{C_0+C_1+(R_1/R_0)C_1}{R_1C_0C_1} \qquad \alpha\beta = \frac{1}{R_0R_1C_0C_1} \tag{2.6}$$

Node voltages $V_0(t)$ and $V_1(t)$ are written as

$$V_0(t) = V_{00}e^{-\alpha t} + V_{01}e^{-\beta t} \tag{2.7}$$

$$V_1(t) = V_{10}e^{-\alpha t} + V_{11}e^{-\beta t}$$

and they satisfy the following four initial conditions at t=0.

$$V_0(0) = V_1(0) = VDD$$

$$\left[\frac{dV_1(t)}{dt}\right]_{t=0} = 0 \qquad \left[\frac{dV_0(t)}{dt}\right]_{t=0} = -\frac{VDD}{C_0R_0}$$

We obtain

$$V_0(t) = VDD\frac{(C_0R_0\beta-1)e^{-\alpha t} - (C_0R_0\alpha-1)e^{-\beta t}}{C_0R_0(\beta-\alpha)} \tag{2.8}$$

$$V_1(t) = VDD\frac{\beta e^{-\alpha t} - \alpha e^{-\beta t}}{\beta-\alpha}$$

When $R_0 >> R_1$, equation (2.5) has one solution that is small: The s^2 term in equation (2.5) is neglected and we obtain

$$s \approx \frac{1}{R_0(C_0+C_1)} \qquad (=\alpha)$$

and another solution that is large: The term in Equation (2.5) that does not contain s is neglected and we obtain

$$s \approx \frac{C_0+C_1}{R_1C_0C_1} \qquad (=\beta)$$

We define α and β in this way, so that $\beta >> \alpha$. When t is positive, the terms in equation (2.8) that involve $e^{-\beta t}$ may be approximated by zero. Then

$$V_1(t) = VDD \frac{\beta}{\beta-\alpha} e^{-\alpha t} = VDD\, e^{-\alpha t+\log \frac{\beta}{\beta-\alpha}}$$

and therefore by requiring

$$-\alpha T_D + \log \frac{\beta}{\beta-\alpha} = -1$$

we get

$$T_D = \frac{\alpha+\beta}{\alpha\beta} \qquad (2.9)$$

when $\beta >> \alpha$. Using equation (2.6), we obtain equation (2.3).

When $R_1 >> R_0$, equation (2.5) has the solution

$$s \approx \frac{1}{R_1 C_1} \qquad (=\alpha)$$

and another solution,

$$s \approx \frac{1}{R_0 C_0} \qquad (=\beta)$$

and again $\beta >> \alpha$. Then following the same manipulation we arrive at equation (2.3). When neither $R_0 >> R_1$ nor $R_1 >> R_0$ is satisfied, the simplification does not hold, but the formula is still quite accurate.

The error when $R_0=R_1=R$ and $C_0=C_1=C$ is estimated as follows. Since

$$\alpha = \frac{3-\sqrt{5}}{2}\left[\frac{1}{RC}\right] = \frac{0.3819}{RC} \quad \text{and} \quad \beta = \frac{3+\sqrt{5}}{2}\left[\frac{1}{RC}\right] = \frac{2.6180}{RC}$$

the delay time is determined from

$$\frac{1}{e}=\frac{1}{2\sqrt{5}}\left\{(3+\sqrt{5})\, e^{-\frac{3-\sqrt{5}}{2}\left(\frac{t}{RC}\right)}-(3-\sqrt{5})\, e^{-\frac{3+\sqrt{5}}{2}\left(\frac{t}{RC}\right)}\right\}$$

as $t = 3.03(RC)$. Equation (2.3) gives $t = 3(RC)$. Therefore, even in this case the agreement is good.

The analysis shows that the two-stage RC chain shown in Figure 2.9(a) can be approximated by the single-stage RC chain shown in Figure 2.9(b), consisting of only R_A and C_A. We showed that

$$R_AC_A = T_D = (R_0+R_1)C_1+R_0C_0$$

The two circuits can be made quite similar in their characteristics by further requiring that the low-frequency impedance looking into the single port is the same. This requirement means that the two circuits, one of them is a simple approximation of the other , must drive the same resistive impedance at DC. This results in

$$R_A = R_0+R_1$$

When the three-stage circuit shown in Figure 2.9(c) is considered, the first two stages can be substituted for by Figure 2.9(b). Then the circuit of Figure 2.9(b) has a delay

$$\begin{aligned} T_D &= (R_A+R_2)C_2+R_AC_A \\ &= (R_0+R_1+R_2)C_2+(R_0+R_1)C_1+R_0C_0 \end{aligned}$$

As a generalization, the discharge delay of the last node of the cascaded N-stage RC chain circuit is given by

$$T_D=\sum_{i=0}^{N} R_i \sum_{j=i}^{N} C_j = \sum_{i=0}^{N} C_i \sum_{j=0}^{i} R_j \qquad (2.10)$$

2.6 Monotonicity of the Delay with Area

In the previous section, we have shown a closed form formulation approximating the discharge delay of an RC chain. In this section, we will use this formulation along with a set of simplifying assumptions to prove that the discharge delay decreases monotonically with transistor chain area if the transistor size distribution is selected properly.

Refering to Figure 2.9(e), each RC link is assumed to be related to a transistor geometry by the following;

$$R_i = \frac{K_r}{W_i} \tag{2.11}$$

and

$$C_i = K_c W_i \tag{2.12}$$

where W_i is the gate width of the ith transistor (assuming fixed channel length) and K_r and K_c are transistor constants. More detailed discussions about modelling of transistor chains, capacitance approximation, and resistance approximation will be presented in chapter 3. Now, we re-write equation (2.10) in the form of delay terms as follows;

$$T_D = \sum_{i=0}^{N} T_{Di} \tag{2.13}$$

where

$$T_{Di} = R_i \sum_{j=i}^{N} C_j \tag{2.14}$$

Substituting equations (2.11) and (2.12) in equation (2.14), we obtain;

$$T_{Di} = K_r K_c \sum_{j=i}^{N} \frac{W_j}{W_i} \tag{2.14}$$

Now, let us consider the case where all the transistor widths in the chain are fixed to W_{FIXED} except for the bottom one which corresponds to i=0, we obtain

$$T_{D0} = K_r K_c \sum_{j=i}^{N} \frac{W_{FIXED}}{W_0} \tag{2.15}$$

From equation (2.15), It is clear that T_{D0} decreases monotonically with W_0 and hence, by equation (2.13), the discharge delay of the chain also decreases monotonically with W_0. Note that all delay terms other than T_{D0} in equation (2.13) are independent of W_0.

Here we have shown that it is possible to decrease the delay by increasing the transistor chain area and in chapter 3 we will present several techniques to minimize the discharging delay of NFET chains.

2.7 Accuracy of Elmore's Delay Formula

Before leaving this chapter, we will discuss the accuracy of Elmore's delay formula. Figure 2.9 shows discharging node waveforms of a seven link RC chain. The waveforms are obtained by SPICE simulation and by applying Elmore's single time constant. It is clear that the waveforms are not well matched and this is expected because Elmore's formula assumes single time constant and ignores many minor poles that modify the discharging node waveform. Figure 2.11 shows a plot of the discharge delay (when the output node drops to 36% of its original value) versus the number of links in the RC chain. With its computational simplicity, Elmore's formula predicts the discharge delay quite accurately and, hence, for those problems that do not require waveform shapes, the RC model and Elmore's formula are quite acceptable. The simple algebraic expressions produced by

Elmore's approximations will be indispensable in obtaining the analytic results of Chapter 3.

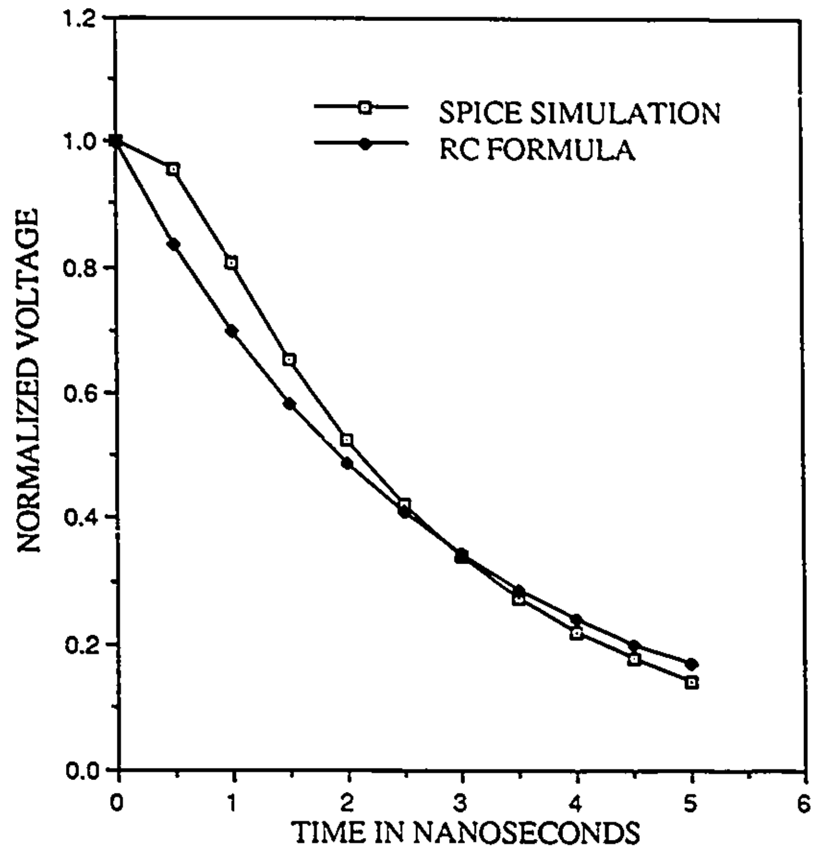

Figure 2.10 7-RC-Link Discharge Waveforms

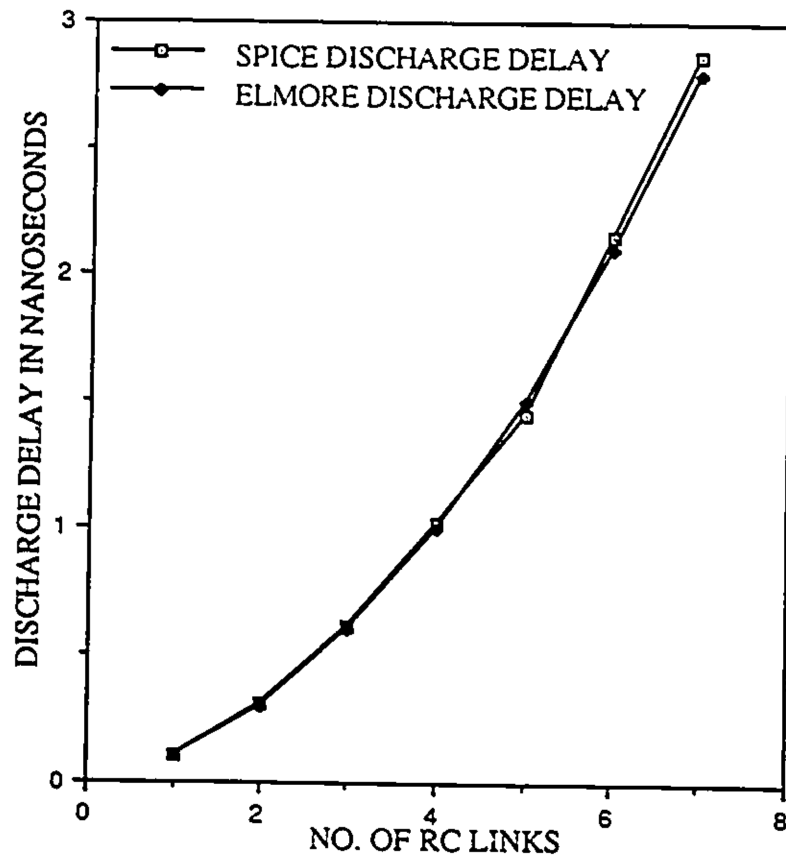

Figure 2.11 Discharge Delay Versus No. of RC Links

CHAPTER 3

Transistor Sizing Approaches for NFET Chains

3.1 Introduction

Switching circuits can be analyzed using standard circuit simulators such as SPICE [11] and ASTAP [20]. But such circuit simulators require a great amount of CPU time and memory storage. To reduce the complexities of the analysis and the models, a number of new techniques have been developed, for example, using tables instead of transistor equations [3], using macromodels [9], ...etc.

One of the successful techniques that has emerged in recent years is the RC tree model [13],[7] based on Elmore's delay formula [5]. The RC model is the heart of many switch level simulators such as TMODS [18] and CRYSTAL [12].

Recently, RC model based delay simulators have been used in circuit speed optimization; for example SLOP [22]. Circuit sizing techniques are powerful since it has been shown that 10% of the delay and 30% of the gate area can be decreased simultaneously [15]. From an extensive literature search it appears that all sizing techniques, so far published, rely on

some type of iterative optimization procedure rather than an analytical formulation to the sizing problem. It is clear that an analytical approach to circuit sizing will both save substantial amounts of CPU time and, more importantly, provide an algebraic foundation on which to base automated design procedures.

A rule of thumb that has been evolved in the design of VLSI circuits is to limit the number of transistors connected serially in a logic block (logic height) in order to maintain a minimum performance level; however, as the feature size becomes smaller in the sub-micron region, this rule can be relaxed [14] and higher logic chains are possible. This encourages more extensive use of NAND/NOR complex gates and makes the transistor sizing more important and effective. In such circumstances, the use of analytical techniques in transistor sizing will eliminate the very large computational time that will be required for iterative techniques.

In this chapter we introduce a new RC analytical model and several sizing techniques for series transistor chains. Section 3.2 explains the necessary mechanisms of a typical dynamic NFET chain structure that are needed to generate an adequate delay model. The RC model is discussed in section 3.3 and its parasitic capacitance approximation is dealt with in section 3.4. In section 3.5, a unique approach is introduced to approximate the channel resistance. SPICE level 2 model equations are used to approximate the resistance of those NFETs that mainly discharge in the linear region while a test circuit is proposed to approximate the resistance of the top NFET that mainly discharges in the saturation region. This approach not only assures that an optimum top NFET width, relative to the other NFET widths, is achieved but also reduces the discharge delay error for the scaled NFET chains to a $\pm$ 5% window. Section 3.6 discusses existing optimization techniques, for NFET sizing, that we use for comparison purposes in section 3.8. In section 3.7 we introduce a novel analytical approach to NFET sizing. Delay-area curves for optimum NFET chain sizes are obtained easily, allowing suitable trade-off decisions by the circuit

designer. A further novel technique is presented in this chapter which transforms any N-variable optimization problem for NFET chain sizing to a single-variable optimization if the constraint of limiting the transistor widths is desired. Section 3.8 presents the results and discussions and section 3.9 contains the final conclusions.

3.2 Discharge Delay of an NFET Chain

Consider the dynamic gate structure shown in Figure 3.1. The NFET chain consists of N+1 serially connected transistors from the evaluation node to the ground. The bottom NFET is the ground switch while the top PFET is the precharge transistor. C_L models the load capacitance connected to the evaluation node. IN_1, IN_2,..., IN_N is the input vector and PHI is the gate clock connected to both the ground switch and the precharge PFET. This dynamic structure is general to most dynamic gates with only slight modifications.

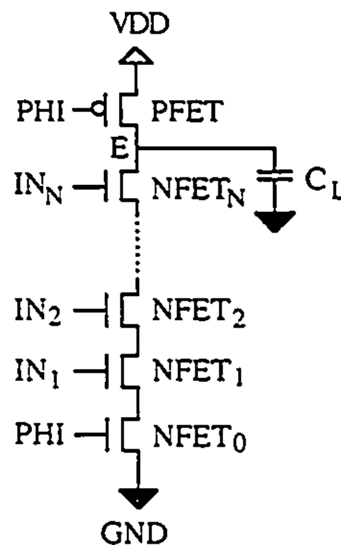

Figure 3.1 Typical CMOS Dynamic Gate Chain

The problem becomes too complex if the dependence of the discharge delay on the rate of change of the input voltage is included in the analysis. Therefore, we assume that the gate

input voltages switch instantly. The clock has two distinct cycles in which the gate operates; precharge cycle and evaluation cycle as shown in Figure 3.2.

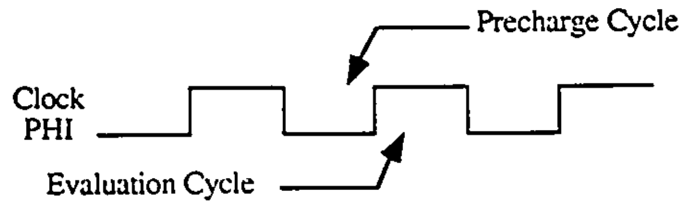

Figure 3.2 Dynamic Gate Timing Diagram

3.2.1 Precharge Cycle

In this cycle, the clock level is low and the ground switch $NFET_0$ is off isolating the circuit from ground. The precharge PFET is on and the evaluation node is pulled up to VDD. Now, let us consider the worst case precharge condition where all inputs to the NFET chain are high (we assume a voltage of VDD in our analysis). If the precharge cycle is kept on for a relatively long period of time, the top transistor, $NFET_N$, in the chain will enter the cutoff region leaving $V_S = V_D\text{-}V_{THN} = VDD\text{-}V_{THN}$. Where V_{THN} is the threshold voltage of $NFET_N$ including the back-bias effect. Transistors $NFET_1$ to $NFET_{N-1}$ will continue to conduct until V_{DS} =0 and all internal nodes of the chain will be charged up to $VDD\text{-}V_{THN}$. The state of the dynamic gate after this long precharge period is shown in Figure 3.3.

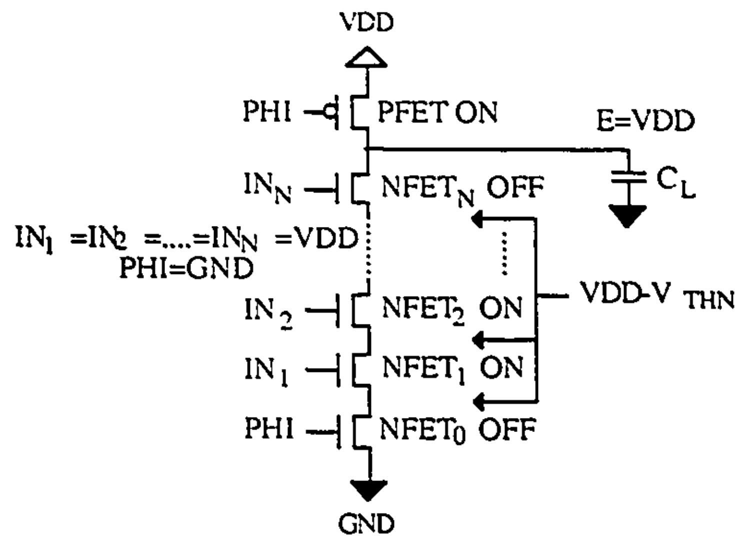

Figure 3.3 Long Precharge State

3.2.2 Evaluation Cycle

After the input vector stabilizes, the clock level switches high and the precharge PFET is turned off. The ground switch $NFET_0$ turns on and the evaluation node discharges low or remains high depending on the state of the input vector. The worst case evaluation condition is where the evaluation node has to discharge low and the input vector is high. In this case, the top transistor, $NFET_N$, pulls out of the cutoff region to the saturation region and then to the linear region. All other transistors in the chain stay in the linear region throughout the discharge period including the ground switch. Figure 3.4 shows the state of the dynamic gate after worst case evaluation condition.

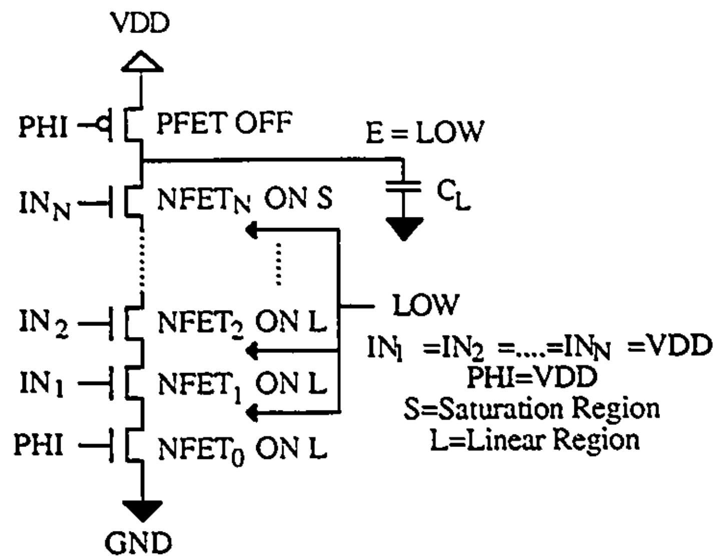

Figure 3.4 Worst Case Discharge State

It is worth mentioning that it is the discharge delay which limits the clocking speed of the gate and any improvement to the discharge delay greatly enhances the logic gate performance. There are two dominant factors, working against each other, that directly affect the discharge delay. The first is the evaluation node capacitance which consists of C_L and the PFET drain capacitance. The second factor is the size of the NFET chain. As the size of an NFET transistor in the chain increases, the current driving capability also increases which tends to decrease the delay; however, the parasitic capacitances associated with the NFET increase and this tends to increase the delay. In the next section, we deal with the problem of predicting the discharge delay for given sizes of the NFET chain [15], using these dominant factors.

3.3 RC Model

In order to analyze the delay characteristics of an NFET chain, the chain must be represented by a simple and manageable circuit model that contains the essential physical mechanisms. In this work we use the established RC model presented in [13] and [7] since

it predicts the discharge delay in terms of circuit parameters not device parameters [21]. This former method is useful in developing algorithms for gate-level delay simulators, but the present objective is to approximate the NFET chain pulldown delay.

Each transistor in the NFET chain is replaced by a series resistance, R_i, and a parallel parasitic capacitance, C_i. The loading effect of the precharge PFET is added to C_L. Figure 3.5 shows the process of constructing the RC model.

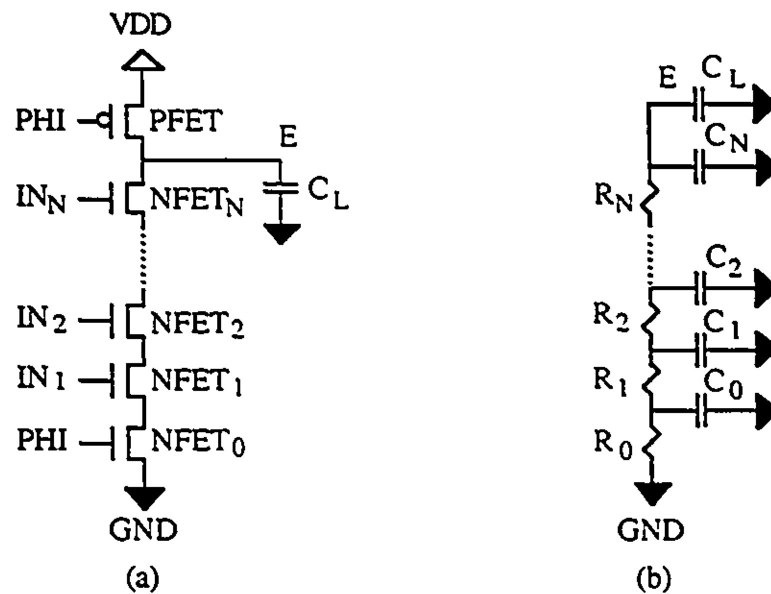

Figure 3.5 (a) Dynamic Gate Chain (b) RC Model

C_i consists of the following contributions;

1. Drain-diffused island capacitance of $NFET_{i-1}$.
2. Source-diffused island capacitance of $NFET_i$.
3. 1/2 the gate-to-channel capacitance of $NFET_{i-1}$.
4. Gate-to-drain overlap capacitance of $NFET_{i-1}$.
5. 1/2 the channel-to-substrate capacitance of $NFET_{i-1}$.
6. 1/2 the gate-to-channel capacitance of $NFET_i$.

7. Gate-to-drain overlap capacitance of $NFET_i$.

8. 1/2 the channel-to-substrate capacitance of $NFET_i$.

Normally, the above parasitic capacitances are lumped to node N_i [16] and thus over estimate the actual delay of the circuit [12]. R_i models the average channel resistance during discharge and depends on the voltage of the nodes N_{i+1}, N_i, and IN_i. We assume that the inputs switch instantly and V_{IN_i}=VDD; therefore, the current, I_i, that flows through $NFET_i$, and R_i are ideally related by equation (3.1)

$$R_i = \frac{1}{T_D}\int_0^{T_D} \frac{VN_{i+1} - VN_i}{I_i}\,dt \tag{3.1}$$

Now, for a given technology, it is possible to calculate C_i and R_i as will be shown shortly. The discharge delay where the evaluation node drops to 0.36 of its original value (VDD) is approximately given by Elmore's delay formula [5]

$$T_D = \sum_{i=0}^{N} R_i \sum_{j=i}^{N} C_j = \sum_{i=0}^{N} C_i \sum_{j=0}^{i} R_j \tag{3.2}$$

In order to take account of the C_L load, from Figure 3.5(b), we re-write equation (3.2) as:

$$T_D = \sum_{i=0}^{N} R_i \left[\sum_{j=i}^{N} C_j + C_L\right] = \sum_{i=0}^{N} C_i \sum_{j=0}^{i} R_j + C_L \sum_{j=0}^{N} R_j \tag{3.3}$$

It is worth noting that equation (3.3) implicitly assume that all internal node voltages of the RC chain are initially equal to the evaluation node voltage. This assumption, however, is not true for the NFET discharge chain and we expect some extra error due to this. Another

source of error is in the approximation of R_i where the channel resistance is not linear and in some cases is difficult to linearize, as in the case of the saturation region.

With appropriate linear definitions for the non-linear channel resistances and parasitic capacitances, the approximate RC model produces satisfactory results for delay approximation [7] and, more importantly, gives excellent results in sizing NFET chains [15]. This will also be demonstrated later in this chapter.

3.4 Parasitic Capacitance Approximation

The capacitances involved in the RC model are relatively easy to calculate. At this point it is necessary to simplify matters and ignore non-linear capacitances or suppress the non-linearity by assuming fixed voltages. The errors due to these assumptions can be taken care of by scaling the resulting delay in accordance with SPICE simulations. Figure 3.6 shows the capacitances considered in the RC model representation of an NFET (likewise for a PFET).

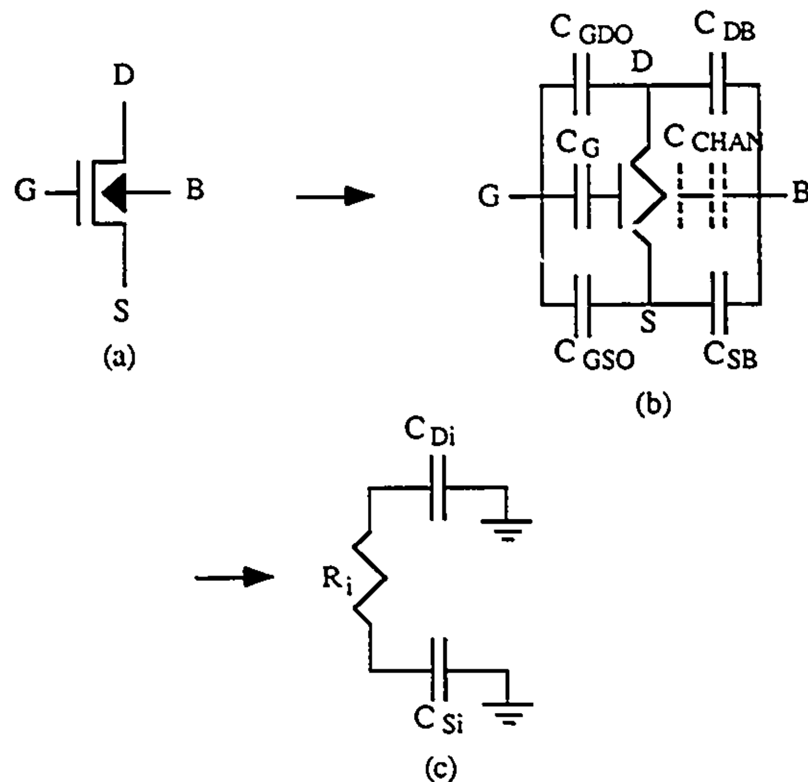

Figure 3.6 CMOS Transistor Capacitance Model

There are two things worth mentioning concerning Figure 3.6(b). The first is that the channel capacitance, C_{CHAN}, which is due to the channel potential with respect to the bulk voltage, has little affect on the total delay of the model, and therefore is not considered. The second thing is that the gate-to-channel capacitance, C_G, is distributed over the channel resistance and, for simplicity, we assume that it is lumped between C_{Di} and C_{Si}. Furthermore, in the RC model, a step input is assumed and, hence, the gate voltage is constant and equal to VDD. Therefore all capacitances connected to the gate voltage, V_G, will be replaced by a ground terminal (virtual ground) without affecting the behavior of the capacitance.

Before we explain the various capacitances in Figure 3.6(b,c), the transistor layout has to be defined. The following standard layout technique is assumed, throughout the thesis, using technology rules from our target 3μ-DLM process [4].

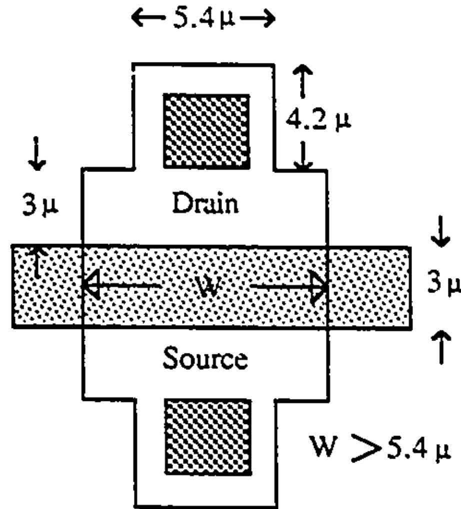

Figure 3.7 Standard Transistor Layout

The drain and source island area and perimeter are easily found and given by,

$$AD = AS = W \times 3 \times 10^{-6} + 22.68 \times 10^{-12}$$

$$PD = PS = W + 14.4 \times 10^{-6}$$

Note that other layout techniques, such as node merging, and other technology rules can be easily incorporated. Now we consider the capacitances shown in Figure 3.6(b) and their evaluations according to SPICE models [2],[19].

1. Gate-to-channel capacitance

$$C_G = COX \times \text{gate area} + CEDGE \times \text{gate perimeter}$$

2. Gate-to-drain overlap capacitance

$$C_{GDO} = CGDO \times W$$

3. Drain-to-substrate capacitance

$$C_{BD} = CJ \frac{AD}{\left[1 - \frac{V_{BD}}{PB}\right]^{MJ}} + CJSW \frac{PD}{\left[1 - \frac{V_{BD}}{PB}\right]^{MJSW}}$$

Assuming $V_{BD}=0$ to remove the non-linearity of the capacitance, we get

$$C_{BD} = CJ \times AD + CJSW \times PD$$

4. Gate-to-source overlap capacitance

$$C_{GSO} = CGSO \times W$$

5. Source-to-substrate capacitance as in (3.3)

$$C_{BS} = CJ \times AS + CJSW \times PS$$

Capacitances 2,3, and half of 1 are lumped to C_{Di} in Figure 3.6(c). Likewise capacitances 4,5, and half of 1 are lumped to C_{Si}.

3.5 Channel Resistance Approximation

We are modeling the non-linear behavior of the MOSFET channel with a linear resistor. There are two distinct approaches, to approximate the channel resistance, sighted in the literature. The first approach is to simulate a test circuit using SPICE and then extract the average discharge resistance from the data obtained [24]. The second approach is to use SPICE model equations to calculate the average channel resistance [15]. Here we present a unique method that combines both approaches which results in a smaller area-delay product for the NFET chain.

As explained in section 3.2, the transistors in the discharge chain are either in the linear or saturation region. The top transistor in the chain mainly operates in the saturation region while the others mainly operate in the linear region.

3.5.1 Channel Resistance Approximation for Linear Region NFETs

We model transistors in the linear region with a minimum channel resistance using the SPICE MOSFET level-2 model, as shown in equation (3.4);

$$I_{DS} = \beta[(V_{GS}-V_{BIN}-\frac{\eta V_{DS}}{2})VDS - \frac{2}{3}\gamma S\,[(2\phi_F+V_{DS}-V_{BS})^{\frac{3}{2}} - (2\phi_F-V_{BS})^{\frac{3}{2}}]] \quad (3.4)$$

where,

$$\beta = \frac{KP.W}{L-2X_{jl}}$$

then,

$$R_{DS}(V_{DS}) = \frac{\delta V_{DS}}{\delta I_{DS}} = \frac{1}{\beta\left[V_{GS}-V_{BIN}-\eta V_{DS}-\gamma_S(2\phi_F+V_{DS}-V_{BS})^{\frac{1}{2}}\right]} \quad (3.5)$$

The minimum channel resistance is found for $V_{DS}=0$, and this value can be used for the RC model [15]. Hence:

$$R_{DS}(\text{minimum}) = \frac{1}{\beta\left[V_{GS}-V_{BIN}-\gamma_S(2\phi_F-V_{BS})^{\frac{1}{2}}\right]} \quad (3.6)$$

At this point, it is convenient to express the resistance with W as the single independent variable:

$$RDS = \frac{PURL}{W} \quad (3.6)$$

where:

$$PURL = \frac{L-2X_{jl}}{KP\left[V_{GS}-V_{BIN}-\gamma_S(2\phi_F-V_{BS})^{\frac{1}{2}}\right]} \qquad (3.7)$$

3.5.2 Channel Resistance Approximation for Saturation Region NFET

For the transistor working mainly in the saturation region (the top MOSFET connected to the evaluation node), we use SPICE simulations, along with the RC model and an optimization routine, in order to define its per unit channel resistance. Consider the test circuit shown in Figure 3.8.

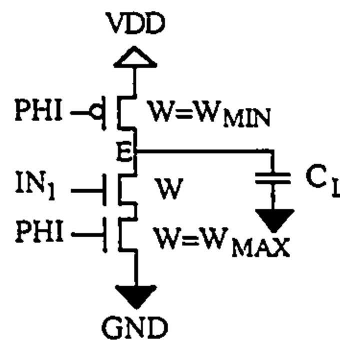

Figure 3.8 Test Circuit

W_{MAX} and W_{MIN} are the maximum and the minimum transistor widths that most likely will be used in the transistor sizing problem. Node IN_1 is kept at VDD and C_L is replaced by a minimum size static inverter to more appropriately simulate the output loading. The following algorithm is used to obtain the per unit resistance, PURS, of the top NFET:

(1) Using SPICE simulations of the test circuit, we define the clock signal, PHI, as switching from low to high at the start of the simulation; the discharge delay of node E is recorded for a range of values of W.

(2) Plot the discharge delay versus W, and select an optimum width from the plot; an example is shown in Figure 3.9.

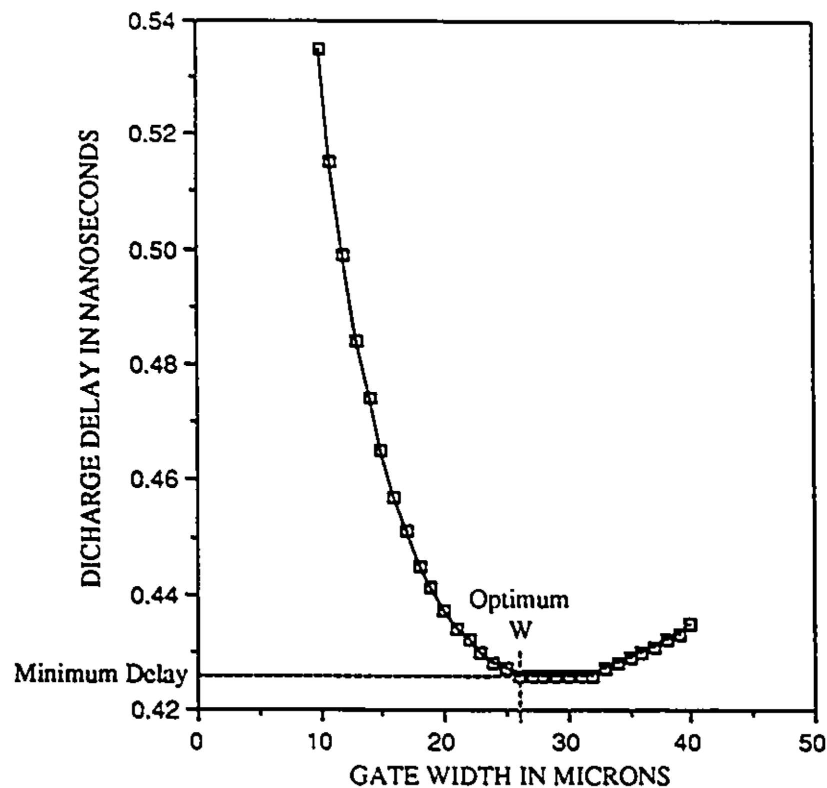

Figure 3.9 Delay VS Width for the Test Circuit

(3) Assign an initial value to PURS (e.g. the value of PURL), and use the optimization routine of section 3.6 and the RC model to find the optimum W for the test circuit given in Figure 3.8.

(4) Adjust PURS until the optimum W obtained in step (3.3) matches that from step (2).

For this particular example and using our target 3μ-DML technology, it is found that

$$PURS = 1.7 \times PURL \tag{3.8}$$

The fact that two different values of resistance are obtained validates our improved modeling procedure.

3.6 A Survey of Optimization Approach to NFET Sizing

In this section we will discuss optimization techniques normally used to size NFET discharge chains. We will use the results of this survey in a comparison study with our new approaches. These results are presented in Section 3.7.

The RC model is used to define the cost function, discharge delay, which is to be minimized. The delay is given by equation (3.3). Usually NFET sizes are only given by the width of the channel; the length of the channel is kept constant at the minimum feature size offered by the given technology. The resistance and capacitance approximations are calculated according to sections 3.4 and 3.5.

Two optimization approaches are shown here that are commonly used in the transistor sizing problem; they are classified according to the type of constraint applied.

3.6.1 Optimization I

A maximum and a minimum transistor widths constraint, given by equation (3.9), is applied,

$$W_{MIN} < W_i < W_{MAX} \tag{3.9}$$

W_{MIN} is usually set to the minimum feature size of the given technology and W_{MAX} is chosen according to layout constraints. The width of the lowest NFET in the chain can be eliminated from the size vector and set to Wmax. This is true because the discharge delay is more sensitive to the channel resistance of the lowest NFET than its parasitic capacitance contributions, and so we choose the minimum resistance possible by setting the width to Wmax. It is also possible to further reduce the computational time by choosing an appropriate starting vector. Initial optimization results show that the minimum delay occurs when the NFET sizes gradually decrease from the grounded end to the evaluation node, and this suggests using the starting vector of equation (3.10):

$$W_i = W_{MAX} + \frac{W_{MIN} - W_{MAX}}{N\ i} \tag{3.10}$$

Equation (3.10) produces a linear width tapering from the bottom to the top transistor. The following is a brief description of the algorithm used in this approach;

1. Record the delay for W_i, W_i+STEP, W_i-STEP.
2. Replace W_i by W_i±STEP if, Delay (W_i±STEP) $<$ Delay (W_i)
3. Apply the constraint, $W_{MIN} < Wi < W_{MAX}$
4. Repeat steps 1-3 for all transistor widths.
5. Stop when no transistor width changes.

3.6.2 Optimization II

In this approach, an area constraint is used rather than a transistor width constraint. The algorithm given below is similar to several sighted in the literature e.g. SLOP [22] and TILOS [6];

1. Start with minimum transistor widths.
2. Calculate the delay sensitivity of all transistors

   Sensitivity_i = Delay (W_i) - Delay (W_i+STEP)
3. Replace Wi by Wi+STEP only for the transistor with the largest sensitivity.
4. Repeat 2 to 3 until user specified area limit is achieved.

A major drawback of the above algorithm is the requirement of an initial width vector of minimum transistor sizes, and thus the algorithm tends to require many iterations, even if only moderate size logic gates are considered.

3.7 Analytical Approach to NFET Sizing

It has been shown, in the previous section, that optimization techniques can be used along with the RC model to size a discharging NFET chain. As the number of transistors to be sized increases, optimization techniques tend to be extremely slow and alternate analytical approaches become attractive. In this section, a novel analytical approach is introduced which not only speeds the computation time, but also provides the design flexibility afforded by an algebraic formulation.

We start by re-writing equation (3.3) as follows:

$$T_D = \sum_{i=0}^{N} T_{Di} \tag{3.11}$$

where,

$$T_{Di} = R_i \left[\sum_{j=i}^{N} C_j + C_L \right] \tag{3.12}$$

From extensive simulations we find that we can make the assumption that a close to optimum transistor size distribution is achieved when all of the delay terms are identical:

$$T_{D0} = T_{D1} = T_{D2} = \ldots\ldots = T_{DN} \tag{3.13}$$

The results in section 3.8 will demonstrate the validity of this assumption.

We will discuss two approaches to the sizing problem, based on the assumption in (3.13). The first approach produces an area-delay curve that the designer can use to select the best trade-off point; this is a purely analytical technique. The second approach allows the designer to select a maximum limit to the width of the largest (bottom) transistor in the

chain; the technique uses a single variable optimization procedure to arrive at the optimum sizing profile.

3.7.1 Analytical Approach

Let the number of transistors in the chain be N+1 and the discharge delay be T_D. From equations. (3.12) and (3.13), we obtain:

$$T_{Di} = \frac{T_D}{N+1} = R_i \left[\sum_{j=i}^{N} C_j + C_L \right] \tag{3.14}$$

Consider the top NFET (i=N):

$$\frac{T_D}{N+1} = R_N (C_N + C_L) \tag{3.15}$$

Now we can define a single resistor value for this NFET (in saturation) similar to the definition for resistance for NFETs in the linear region (see equation (3.6)). Let:

$$R_N = \frac{PURS}{W_N} \tag{3.16}$$

Any node capacitance in the equivalent circuit for the chain is dependent on the two transistors connected to that node. In the case of the top node, however, the capacitance is only dependent on the width of the top transistor (we assume the pre-charge transistor is a known fixed size). We can therefore solve the problem, analytically, by solving for widths, starting from the top node. We know, from section 3.4, that the top node capacitance has the form:

$$C_N = K_1 + K_2 W_N \tag{3.17}$$

Note that C_L is included in K_1. We can now solve for W_N:

$$\frac{T_D}{N+1} = \frac{PURS}{W_N} (K_1 + K_2 W_N) \tag{3.18}$$

and:

$$W_N = \frac{PURS\ K_1}{\frac{T_D}{N+1} - PURS\ K_2} \tag{3.19}$$

Now consider any other transistor in the chain. From equation (3.6):

$$R_i = \frac{PURL}{W_i} \tag{3.20}$$

and from section 3.4:

$$C_i = K_3 + K_4\ W_i + K_5\ W_{i+1} \tag{3.21}$$

substituting (3.20) and (3.21) in equation (3.14), we get:

$$W_i = \frac{PURL(K_3 + K_5\ W_{i+1} + C_{i+1} + ... + C_N)}{\frac{T_D}{N+1} - PURL\ K_4} \tag{3.22}$$

Note that i ranges from N-1 to 0 therefore, when we solve for W_i, all transistor widths above the ith node in the chain are known, and hence the right hand side of equation (3.22) is readily computed. Sections 3.4 and 3.5 discuss the methods used to generate parameters PURS, PURL, K_1, K_2, K_3, K_4, and K_5.

3.7.2 Single Variable Optimization

In this optimization technique, we will use W_N as the single variable, along with a constraint W_{MAX} on the maximum width of any transistor in the chain. The cost function to be minimized is the delay. We know that if W_N is given then all the transistor sizes in the chain can be found by using equation (3.22). If, in this calculation, W_i for i=j exceeds the constraint, we simply replace W_i by W_{MAX} $\forall\ i \leq j$ (i.e. a constant width from node j to the bottom of the chain). This method always transforms an N+1 variable optimization method to one of only one variable. Our experience is that this approach yields results very

close to those of a conventional N+1 variable optimization procedure. We will demonstrate this in the next section.

3.8 Results and Discussions

Our test circuit is that shown in Figure 3.1 with C_L replaced by a minimum size static inverter. The results presented here are for our target 3μ-DML technology offered by Northern Telecom, Canada. A glossary of the terms used in the plots is defined in table 3.8.1.

W_P	Precharge transistor width
W_{MAX}	Maximum transistor width allowed, in this case 30μ
W_{MIN}	Minimum transistor width allowed, in this case 5.4μ
FIXED WIDTHS	All transistors widths in the chain are set to W_{MAX}
OPTIMIZATION I	Transistor widths obtained by a standard optimization algorithm with maximum width limitation constraint.
OPTIMIZATION II	Transistor widths obtained by a standard optimization algorithm with area limit constraint.
ANALYTICAL APPROACH	Analytical approach described in section 3.7.1
S-V OPTIMIZATION	Top transistor width is selected by the optimizer and the rest of the chain is scaled analytically (see section 3.7.2)

Table 3.1 Glossary Table

A delay-area curve for a seven level height NFET chain is given in Figure 3.10. In this example our analytical approach yields a smaller delay than the standard Optimization I algorithm, and compares well with the Optimization II angle particularly in the small area range. The Optimization approach II took as many as 2000 iterations, per point, in the large area range (STEP=0.6μ) whereas the analytical approach produces results in one calculation per point.

Discharge delay versus chain height for various methods is shown in Figure 3.11. There are two sizes of W_P used to simulate different loading effects on the evaluation node for each method. It is clear that that Optimization I and the S-V Optimization method result in very similar discharge delay curves for both values of W_P compared to the Fixed Widths method. It is also observed that smaller delays are obtained by chain sizing for (1) higher logic heights and (2) for low capacitive loading on the evaluation node. Note that these delays are obtained by SPICE level-2 simulations with the same transistor sizes as produced by the respective methods.

Figure 3.12 shows chain area versus chain height for the same transistor sizes used in Figure 3.11. It is apparent that a substantial chain area reduction can be obtained by circuit sizing techniques (up to 30% savings for 7 level height chain). The Optimization I and the S-V Optimization methods show very close results for $W_P=W_{MIN}$ a larger difference for $W_P=W_{MAX}$, but still within very acceptable limits.

Finally, Figure 3.13 shows the accuracy of the delays, for sized chains, computed by the RC model. The deviations between plots indicate that the transistor sizing is within ± 5% of the optimum that would have been obtained by using SPICE as a delay calculator.

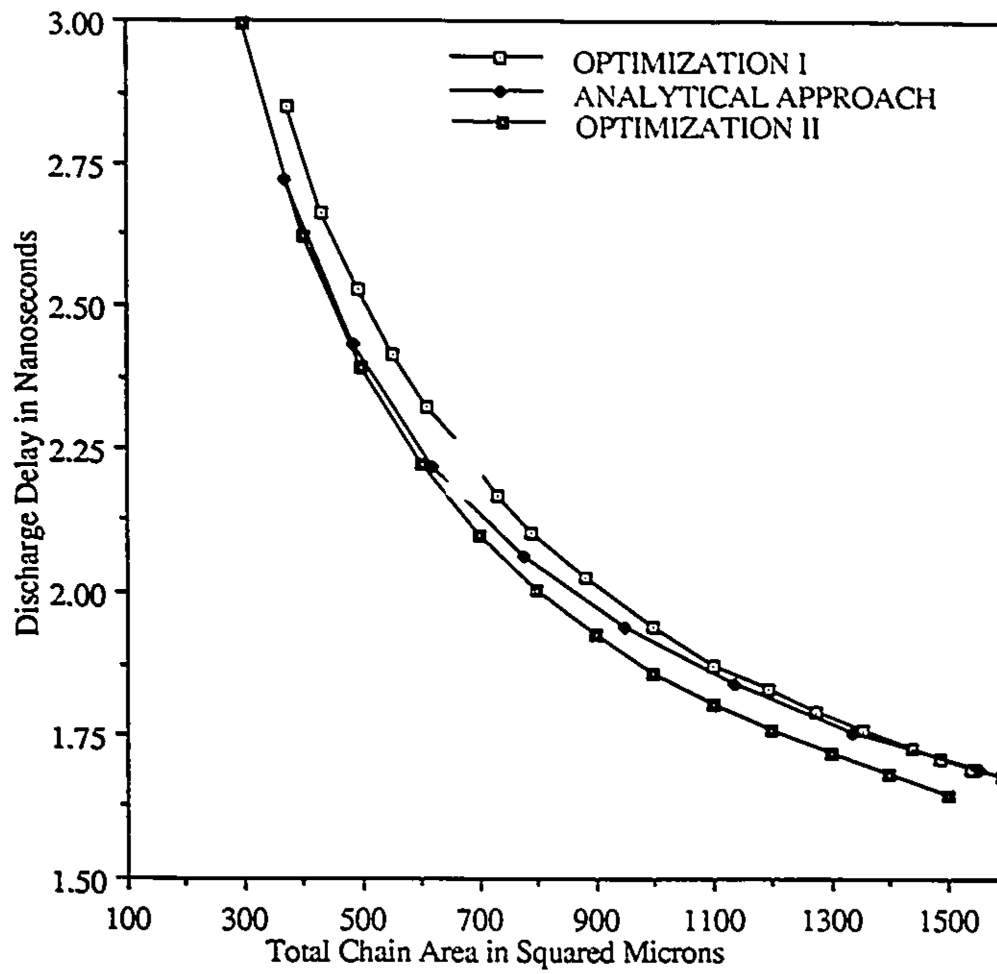

Figure 3.10 Delay VS Area

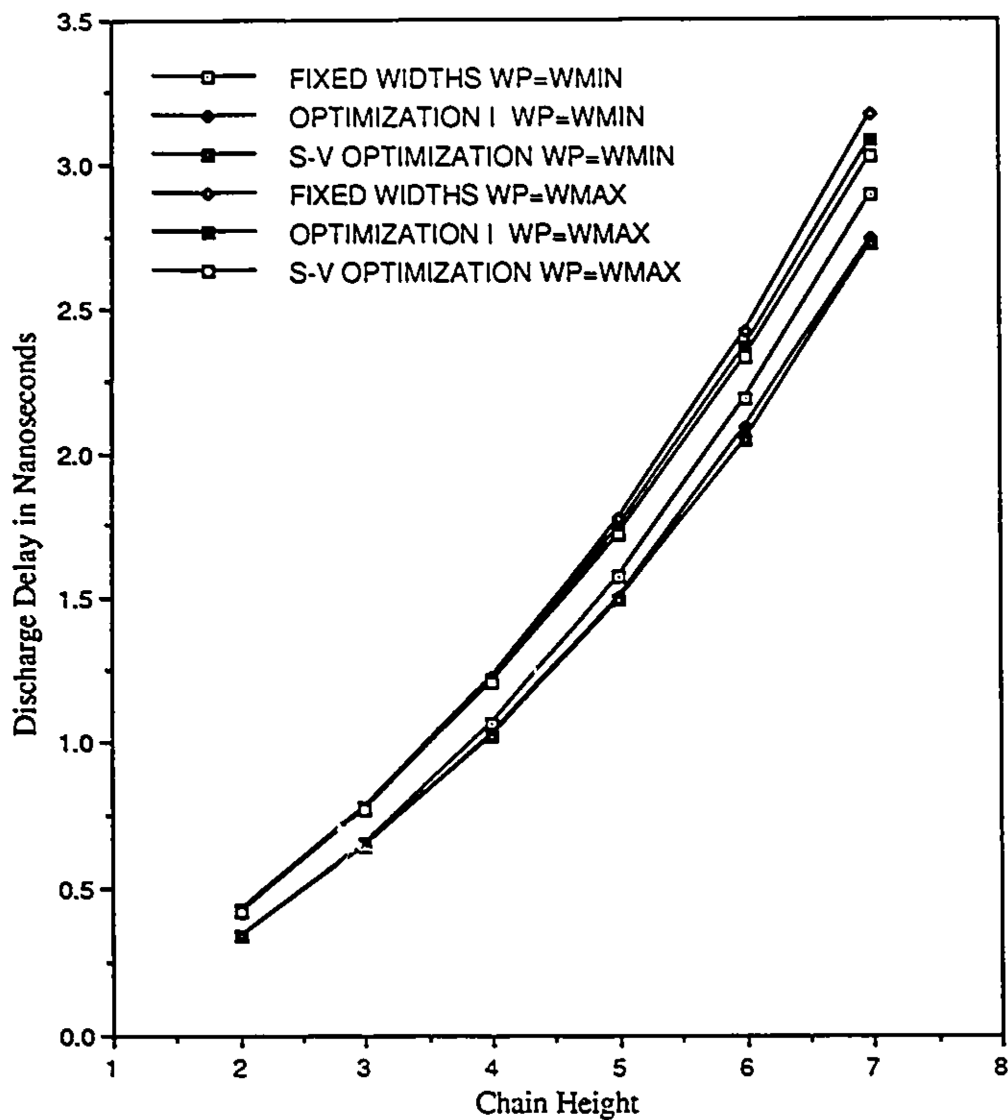

Figure 3.11 Discharge Delay VS Chain Height

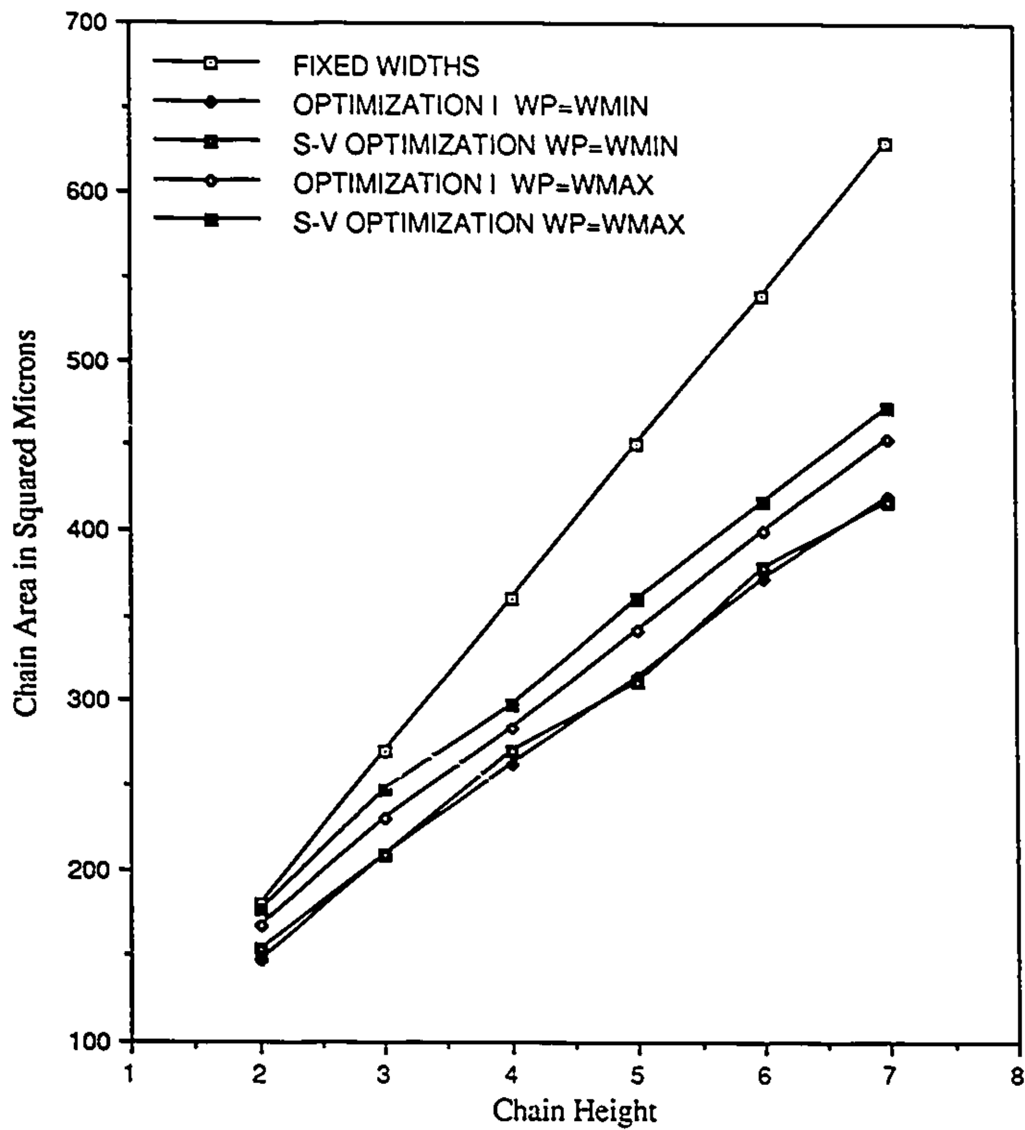

Figure 3.12 Total Chain Area VS Chain Height

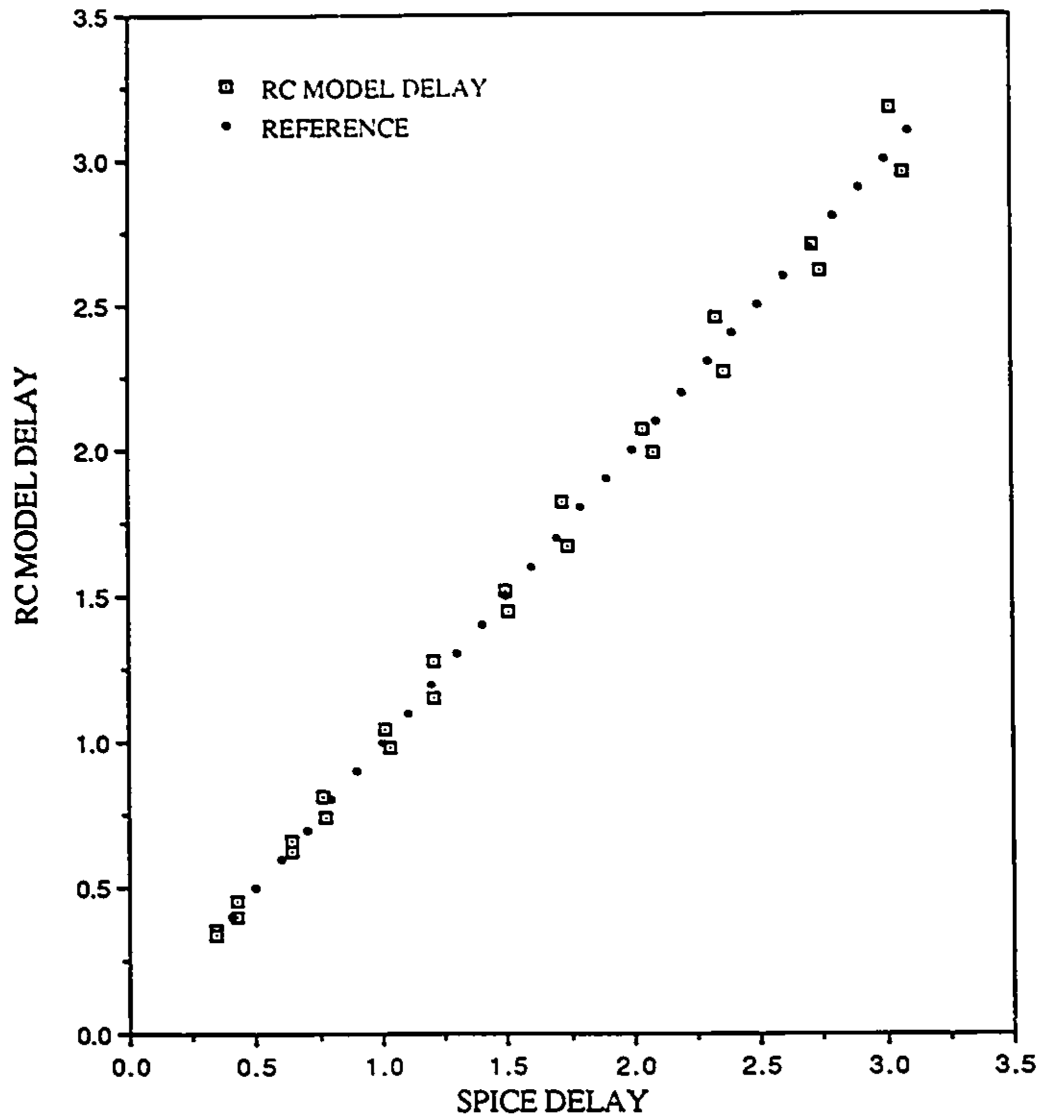

Figure 3.13 RC model delay VS SPICE delay (for sized NFET chains)

3.9 Summary

This chapter has used the RC model to produce a purely analytical approach to the sizing of NFET chains. This analytical method is up to 2000 times faster in computation time than competing iterative optimization techniques. The ability to express the sizing problem

algebraically will undoubtedly have ramifications in the construction of complex NFET block design tools.

This chapter has also presented a single variable optimization approach for NFET chain sizing that implements a maximum width constraint to produce a sub-optimal delay-area curve. We have demonstrated that the results of this approach compare favorably to those obtained from existing iterative N-variable optimization strategies using the same constraint.

CHAPTER 4

Practical Aspects of Circuit Sizing

4.1 Introduction

In this chapter, practical aspects of circuit sizing techniques are considered, such as precharge sizing and the charge sharing, or node coupling, problem. We start this chapter by introducing a simple RC precharge delay model that parallels the developments made earlier for the discharge delay in chapter 3 . The main difference between the two models is that the precharge model has two dominant time constants; this produces a transcendental equation that can be solved by standard numerical techniques, such as the Newton-Raphson method. Also, in this chapter, a major problem associated with dynamic logic structures, due to the floating nature of the evaluation node, is discussed and some possible solutions are suggested. These solutions not only eliminate the charge sharing problem but also increase the stability of the evaluation node against leakage currents and surges and, hence, increases the frequency range at which the dynamic structure can operate.

4.2 Precharge PFET Sizing

This section concentrates on the precharge cycle of the dynamic structure. The underlying assumption throughout this section is that all transistor sizes of the circuit are known and only the precharge PFET is to be found based on the desired precharge delay, T_P. Usually T_P is selected to be equal to the discharge delay, T_D, of the logic gate so that the clock is symmetrical. Capacitance approximation of NFETs and PFETs and channel resistance approximations of the NFETs are those discussed in detail in chapter 3 and only the modelling of the precharge PFET channel resistance will be presented here.

4.2.1 Precharge Delay Model

Figure 4.1 shows a two time constant model used in approximating the precharge delay. R_P is the precharge PFET channel resistance and its approximation will be described in the next sub-section. C_E is the evaluation node capacitance which includes contributions from the loading PFET and NFET (assuming minimum feature size static inverter loading) and contributions from the precharge PFET and the NFETs connected to the evaluation node. R_B and C_B model the n-logic block beneath the evaluation node.

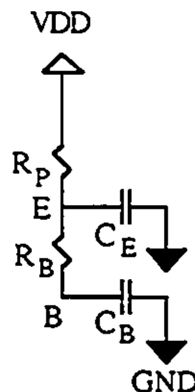

Figure 4.1 Precharge Delay Model

Now, our task is to find an explicit equation that predicts the precharge delay, T_P, as a function of the model parameters. To this end we use the circuit model of Figure 4.1 in conjunction with standard techniques from circuit analysis theory.

Using Laplace transforms, we apply Krichoff's Current Law, KCL, at node V_B, to obtain

$$\frac{V_E - V_B}{R_B} - s\, C_B\, V_B = 0 \tag{4.1}$$

Then

$$V_B = \frac{V_E}{1 + s\, C_B\, R_B} \tag{4.2}$$

Applying KCL at node V_E, we obtain

$$\frac{\frac{VDD}{s} - V_E}{R_P} - s\, C_E\, V_E - \frac{V_E - V_B}{R_B} = 0 \tag{4.3}$$

By substituting equation (4.2) in (4.3) and solving for V_E, we get

$$\frac{V_E(s)}{VDD} = \frac{1}{C_E\, C_B\, R_P\, R_B} \cdot \frac{1 + s\, C_B\, R_B}{s\left[s^2 + \frac{C_B R_B + C_E R_P + C_B R_P}{C_E\, C_B\, R_P\, R_B}\, s + \frac{1}{C_E\, C_B\, R_P\, R_B}\right]} \tag{4.4}$$

Let,

$$A = \frac{1}{C_E\, C_B\, R_P\, R_B} \qquad A_4 = C_B\, R_B$$

$$A_5 = \frac{C_B R_B + C_E R_P + C_B R_P}{C_E\, C_B\, R_P\, R_B} \tag{4.5}$$

Now equation (4.4) becomes,

$$\frac{V_E(s)}{VDD} = A\, \frac{1 + s\, A_4}{s\left[s^2 + A_5\, s + A\right]}$$

$$= A\, \frac{1 + s\, A_4}{s\,(s - S_1)\,(s - S_2)}$$

$$= A \left[\frac{A_1}{(s - S_1)} + \frac{A_2}{(s - S_2)} + \frac{A_3}{s} \right] \quad (4.6)$$

where the poles are given by the quadratic formula,

$$S_{1,2} = -\frac{A_5}{2} \pm \frac{\sqrt{A_5^2 - 4\,A}}{2} \quad (4.7)$$

and constants A_1, A_2, and A_3 are given by using the partial fraction decomposition method as follows,

$$A_1 = \frac{1 + A_4\,S_1}{S_1\,(S_1 - S_2)}$$

$$A_2 = \frac{1 + A_4\,S_2}{S_2\,(S_2 - S_1)}$$

$$A_3 = \frac{1}{(-S_1)\,(-S_2)} \quad (4.8)$$

In order to express V_E in the time domain, we take the inverse Laplace transform of equation (4.6) to obtain equation (4.9).

$$\frac{V_E(t)}{VDD} = A \left[A_1\,e^{S_1 t} + A_2\,e^{S_2 t} + A_3 \right] \quad (4.9)$$

As explained in chapter 2, the precharge delay used in this thesis is defined by the time the evaluation node voltage takes to reach 64% of its final value from the start of the precharge cycle. Hence, equation (4.9) can be written in terms of T_P as follows,

$$0.64 = A \left[A_1\,e^{S_1 T_P} + A_2\,e^{S_2 T_P} + A_3 \right] \quad (4.10)$$

Equation (4.10) is a transcendental equation in T_P and hence, T_P can only be solved for by numerical techniques such as the bisection method, Newton-Raphson method, or secant method.

For the precharge model to be of value to us, its elements should be readily calculable or available. The first step we take, in this quest, is to determine the worst case precharge

condition of the dynamic gate. In a complex dynamic logic gate, the n-logic block contains many paths from the evaluation node to ground and the worst case precharge condition will depend on the circuit topology of the logic structure. To demonstrate the circuit model, we consider a single NFET chain for the n-logic block. Then, C_B is defined as the summation of all node capacitances along the NFET chain excluding the evaluation node capacitance and R_B is defined as the summation of all NFET channel resistances along the chain excluding the ground switch NFET. It is clear from the above discussion that all inputs to the worst case path are assumed to be high or at VDD during the precharge cycle. Figure 4.2 shows the process of modelling a typical dynamic gate for precharge delay approximation.

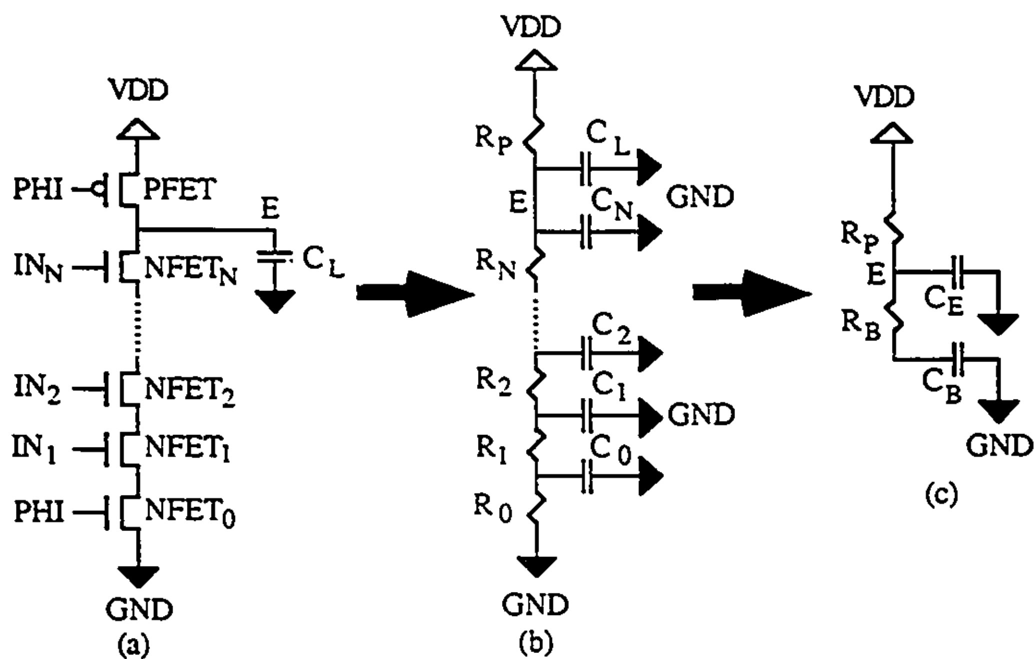

Figure 4.2 (a) Dynamic Gate Chain (b) RC Model (c) Precharge Delay Model

From Figure 4.2, we can write,

$$C_E = C_N + C_L \tag{4.11}$$

$$C_B = \sum_{i=0}^{N-1} C_i \qquad \text{and} \qquad R_B = \sum_{i=1}^{N} R_i \tag{4.12}$$

The method of calculating node capacitances and channel resistances for NFET chains, including evaluation node capacitance, is described in detail in chapter 3 and no further discussion is required. The approximation of R_P is described in the next sub-section.

4.2.2 Precharge Channel Resistance Approximation

SPICE simulations for a simple test circuit are carried out to approximate the precharge per unit channel resistance. Consider the test circuit shown in Figure 4.3.

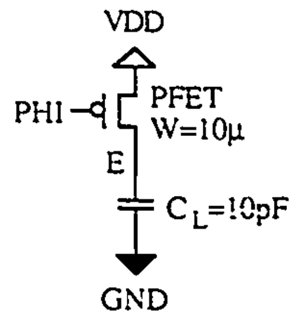

Figure 4.3 Precharge Test Circuit

PFET channel width, W_P, is set to a medium transistor width (e.g 10μ) where the channel length is set to the minimum offered by the targeted technology (e.g 3μ). C_L is chosen very large so that the parasitic capacitance contributions of the PFET are negligible; now the time constant of the test circuit can be written as:

$$T = R_P C_L \tag{4.13}$$

Let,

$$R_P = \frac{PURP}{W_P} \tag{4.14}$$

then,

$$PURP = \frac{T\,W_P}{C_L} \tag{4.15}$$

During SPICE simulations, initially the PHI and E nodes are set to 0 volts. At t=0 PHI switches to high or VDD. The time constant of the circuit is measured as the time taken for the voltage at node E to reach 64% of VDD. The following table gives two example measurements.

Test Circuit Parameters	Test Circuit Time Constant [nanoseconds]
WP=10μ and C_L=1pF	8.86
WP=10μ and C_L=10pF	88.2

Table 4.1 Precharge Test Circuit Time Constants

Table 4.1 shows that when C_L increases by 10 times then T also approximately increases by the same multiplier; this indicates that C_L is the dominant capacitance in the circuit. Now, using the second row of table 4.1 and equation 4.15, PURP is easily calculated as:

$$PURP = 8.82 \times 10^{-2} \tag{4.16}$$

4.2.3 PFET Sizing Algorithm

We are now in a position take on the problem of precharge PFET sizing, and the following algorithm has been developed to determine the PFET size.

1. Set W_P to an initial value.

2. Evaluate R_P, C_B, A, A_1, A_2, A_3, S_1, and S_2.

3. Evaluate,

$$f(W_P) = A\left[A_1 e^{S_1 T_P} + A_2 e^{S_2 T_P} + A_3\right] - 0.64 \tag{4.17}$$

4. Use $f(W_P)$ to modify W_P and go to step 2 until,

$$|W_P(new) - W_P(old)| < \varepsilon \tag{4.18}$$

Some observations can be made about the above algorithm. Many precharge model parameters have to be initialized or calculated before running the algorithm; parameters such as R_B, C_B, and PURP and other model parameters that depend on W_P such as C_E and R_P will be calculated during each iteration of the algorithm. T_P is usually desired to be equal to the discharge delay of the dynamic gate so that the gate clock will be symmetrical and easier to generate. ε is a measure of the resolution of the resulting W_P and is best set to the resolution of the gate width of the fabrication technology (e.g 0.6μ in our 3μ-DML target technology). Note that, if for a minimum PFET width (given by the technology), the resulting T_P is less than the desired T_P then $f(W_{Pminimum}) > 0$ and hence W_P is set to $W_{Pminimum}$ and no further calculations are required. The method by which W_P is modified in step 4 is left open and many techniques can be used such as Newton-Raphson or the bisection method.

4.2.4 Precharge PFET Sizing Results and Discussions

In the previous sub-sections, we have presented techniques to both approximate the precharge delay and to size the precharge PFET if the desired precharge delay is specified and the sizes of the n-logic block are known. In this sub-section, various results will be presented and discussed. In Appendix C, two FORTRAN codes are given; one is to calculate T_P for an N-1 input dynamic NAND structure and the other is to size the precharge PFET for the same gate.

Figure 4.4 shows a plot of the precharge delay versus the number of levels of a typical dynamic NAND gate. One curve is obtained by using the precharge delay model described above and the other is obtained by SPICE level 2 simulations; all transistor gate sizes are fixed to 5.4μ including the loading NFET and PFET on the evaluation node. Note that it was necessary to scale the precharge delay obtained by the model in order to reduce the overall error between the model and SPICE and this scaling needs to be done only once for a given technology. The discrepancy between the model and SPICE results is due largely to the lumping of the NFET chain capacitances and resistances and also to the nonlinearity of the channel resistances of the FETs. Figure 4.5 shows a plot of the precharge delay of a typical 6 input or 7 level dynamic NAND gate for various sizes of the circuit structure; one curve is obtained by using the precharge delay model and the other by SPICE level 2 simulations. Transistor widths of the NFET chain and the precharge PFET are kept equal and varied from 5.4μ to 30μ. The precharge delay model shows acceptable results compared to SPICE simulations and seems well suited for precharge PFET sizing.

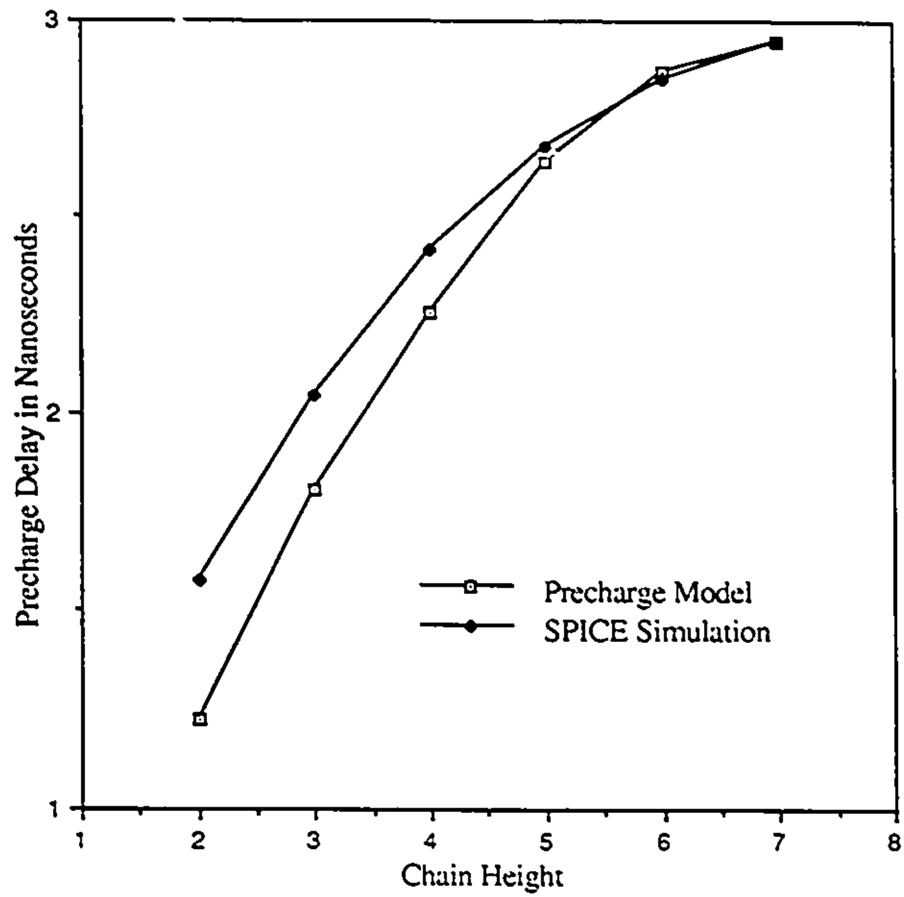

Figure 4.4 Precharge Delay Versus Chain Height

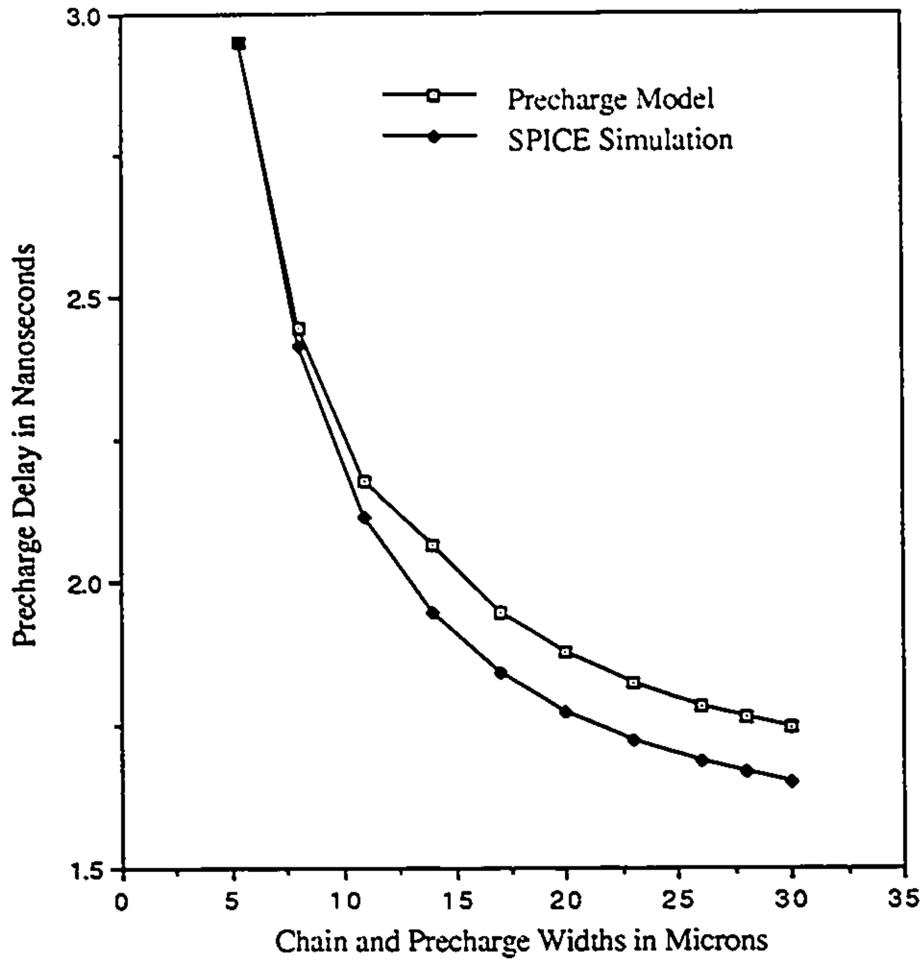

Figure 4.5 Precharge Delay Versus Width

Table 4.2 shows sample precharge PFET sizes obtained by using the precharge model and the algorithm presented in sub-section 4.2.3. For simplicity, the bisection method is used to approximate the size of the PFET that satisfies the resulting transcendental equation.

Circuit Description	Precharge PFET Width
7 level NFET chain, W_{CHAIN}=5.4μ, T_P=3nS	5.4μ
7 level NFET chain, W_{CHAIN}=30μ, T_P=3nS	19.8μ

7 level NFET chain, W_{CHAIN}=5.4μ, Tp=2nS	6.6μ
7 level NFET chain, W_{CHAIN}=30μ, Tp=2nS	26.4u

Table 4.2 Precharge PFET sizing example results

4.3 Charge Sharing

Dynamic logic structures exhibit problems due to the floating nature of the evaluation node during the evaluation cycle. This section discusses the problem of charge sharing, including the effect of transistor sizing, and we suggest some possible remedies.

4.3.1 Charge Sharing Problem

With reference to Figure 4.6, the worst case charge sharing scenario is stated as follows; suppose that the input vector is set such that in the precharge cycle only the evaluation node is charged up to VDD and all internal nodes of the n-logic block are initially discharged to GND. At the end of the precharge cycle, the input vector switches its state such that only those NFETs which are connected to the ground switch are turned off. This will result in a node coupling between all internal nodes of the n-logic block and the evaluation node (except the ground switch node) during the evaluation cycle and, hence, the evaluation node has to share its charge with these internal nodes. In most cases, this effect produces a faulty low evaluation node potential, by dropping below the acceptable logic '1' noise margin threshold for the logic circuit connected to the evaluation node.

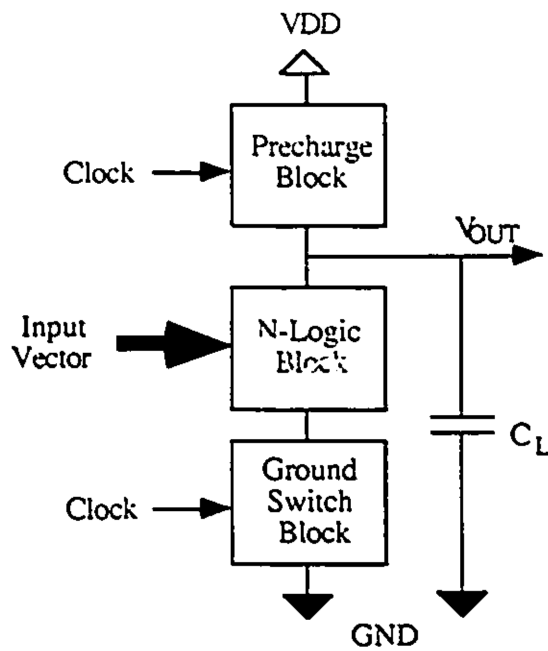

Figure 4.6 N-logic Block Dynamic Logic Structure

4.3.2 Transistor Sizing and Charge Sharing

In this sub-section, we will show the effect of transistor sizing on the charge sharing problem of a dynamic gate structure. A specific dynamic structure is chosen as an illustrative example and, here again, we consider the 6 input NAND dynamic structure shown in Figure 4.7

Figure 4.7 6 input NAND Dynamic Structure

Let us consider the worst case charge sharing condition as explained in the previous sub-section. We can estimate the charge accumulated on the evaluation node as:

$$Q_E = C_E V_{Einitial} \tag{4.19}$$

where $V_{Einitial}$ is the evaluation node voltage at the beginning of the evaluation cycle. Now let:

$$C_{internal} = \sum_{i=1}^{N-1} C_i \tag{4.20}$$

then the evaluation node voltage at the end of the evaluation cycle, assuming a very long evaluation duration, is given by,

$$V_{Efinal} = \frac{Q_E}{C_E + C_{internal}} = V_{Einitial} \frac{C_E}{C_E + C_{internal}} \tag{4.21}$$

We will now give specific example solutions by defining the sizes of the transistors of Figure 4.5 and calculating C_E and $C_{internal}$ by the methods presented earlier in chapter 3. Table 4.3 summarizes the results.

Transistor Sizes [μ]	C_E [FF]	$C_{internal}$ [FF]	Ratio $(\frac{V_{Efinal}}{V_{Einitial}})$
(A) All transistor widths = 5.4	67.65	391.9	0.1472
(B) All transistor widths = 30 except W_{LP} = 5.4 and W_{LN} = 5.4	182.4	1308	0.1224
(C) W_0 = 67.2, W_1 = 48.6, W_2 = 34.8, W_3 = 24.0, W_4 = 16.8, W_5 = 10.8, W_6 = 7.8, W_P = 16.8, W_{LP} = 5.4, and W_{LN} = 5.4	92.86	1262	0.0685

Table 4.3 Sample Circuit Sizes and the Predicted Charge Sharing Problem Severity

Note that the circuit sizes in (C) are obtained using the techniques described in chapter 3 and in section 4.2; the total silicon area occupied by (C) and (B) are practically the same. The last column of Table 4.3 shows that the voltage of the evaluation node drops to about 15% in (A), 12% in (B), and 7% in (C) of its original value during faulty low evaluation under worst case charge sharing condition. This clearly indicates that transistor sizing will not create charge sharing problems other than those that exist in the original circuit design. The results also show the inherent nature of the charge sharing problem, even with minimum size transistors. To avoid this problem, we can impose either restrictions on the input operating mechanism or modify the structure itself. In the next sub-section, we will suggest some solutions based on this rationale. SPICE level 2 simulations, given in Figure 4.8 and 4.9, show both normal and faulty operations of the circuits in (A), (B), and (C) and which support the analytical calculations made in Table 4.3.

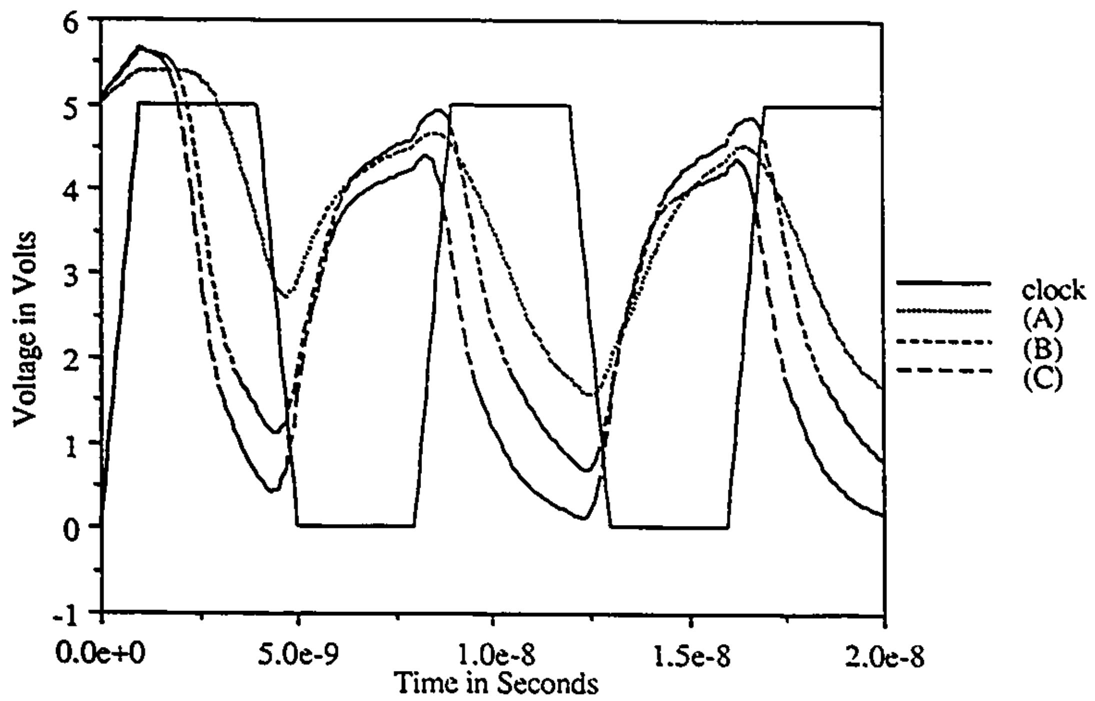

Figure 4.8 Worst Case Precharge and Evaluation of the Circuits in Table 4.3

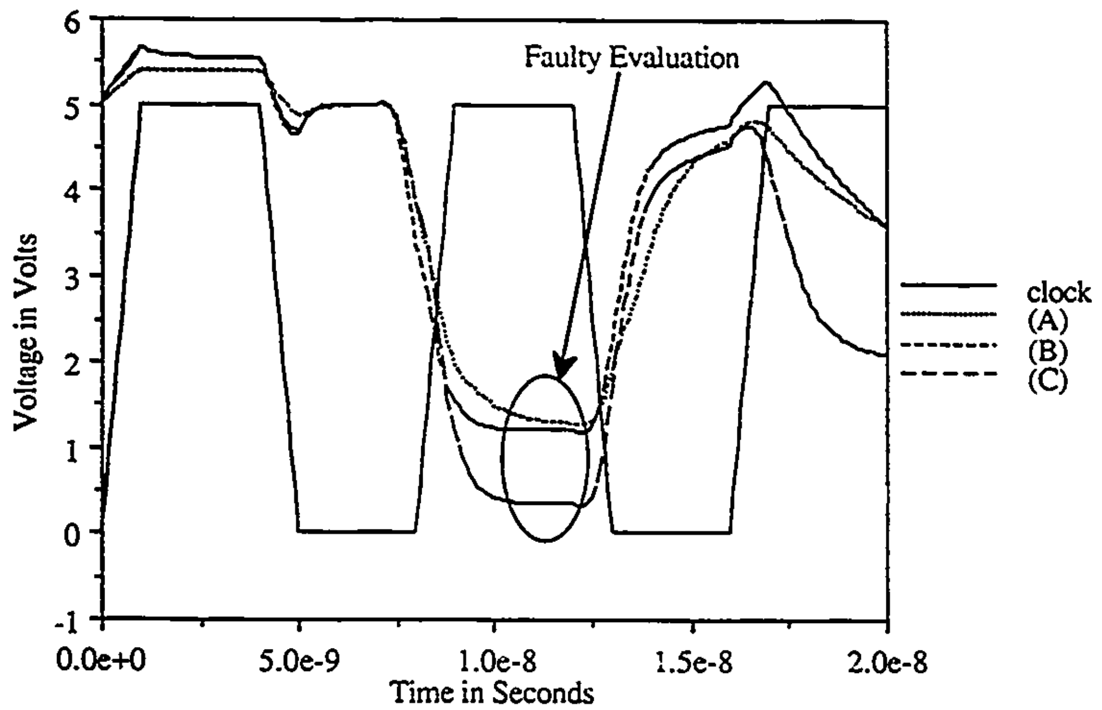

Figure 4.9 Worst Case Charge Sharing Condition of the Circuits in Table 4.3

4.3.3 Remedies to the Charge Sharing Problem

We have seen that charge sharing problem is inherent within the dynamic structure due to the floating nature of the evaluation node in the evaluation cycle. This problem can be alleviated by considering techniques such as the following:

1. *Precharge Distribution* : It is possible to distribute precharge over all internal nodes that provide large contributions to the charge sharing problem. This can be done by connecting precharge PFET drains (minimum size PFETs are sufficient in most cases) to all the internal nodes; this also has the effect of eliminating the worst case precharge condition and, hence, a minimum size precharge PFET, connected to the evaluation node, is sufficient.

2. *Input Vector ORing* : Another way of removing the charge sharing problem is to logically OR all input vector bits with the clock such that at the precharge cycle, all inputs are high and all internal nodes of the n-logic block are charged high. This distributes charge to the internal nodes during pre-charge rather than during the worst case charge sharing evaluation cycle.

3. *Latch Selection* : If the dynamic logic structure is to be pipelined, we can select an appropriate latching circuit to ensure that the input vector always settles during the precharge cycle. This case is realized by using the single phase clocking scheme presented in [23] and [1].

Note that the suggestion made in 1 changes the dynamic structure and that in 2 restricts the input mechanism whereas the suggestion in 3 restricts the mechanism of the data transfer among latching points or pipelined blocks. If the number of internal nodes of the logic block are small, then suggestion 1 is appropriate; if the number of input vector bits are small, then suggestion 2 is more appropriate. For a completely pipelined system, suggestion 3 is the choice. A combination of the above can be considered for more efficient designs and care must be exercised to ensure that the system is timed properly.

Figure 4.10 and 4.11 shows SPICE simulations of the circuit in Figure 4.5 with the sizes given in Table 4.3(C) and the implementation of the precharge distribution technique. The added gate area due to the precharge distribution is marginal at about 6 %.

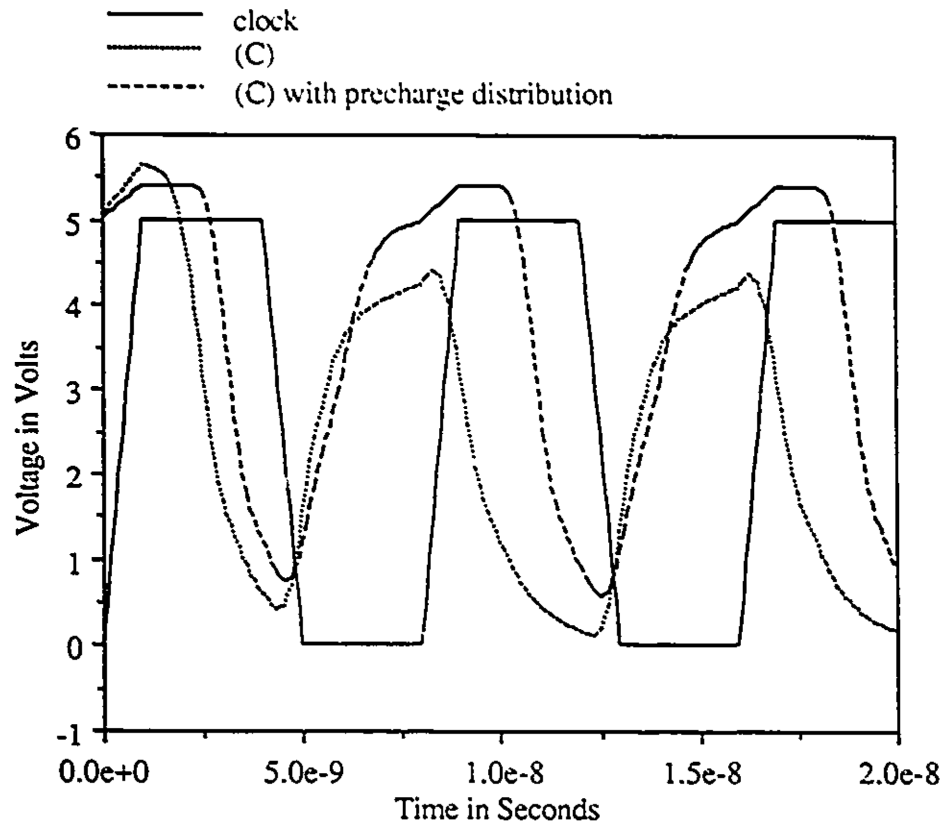

Figure 4.10 Worst Case Precharge and Evaluation of the Circuit (C) with and without Precharge Distribution

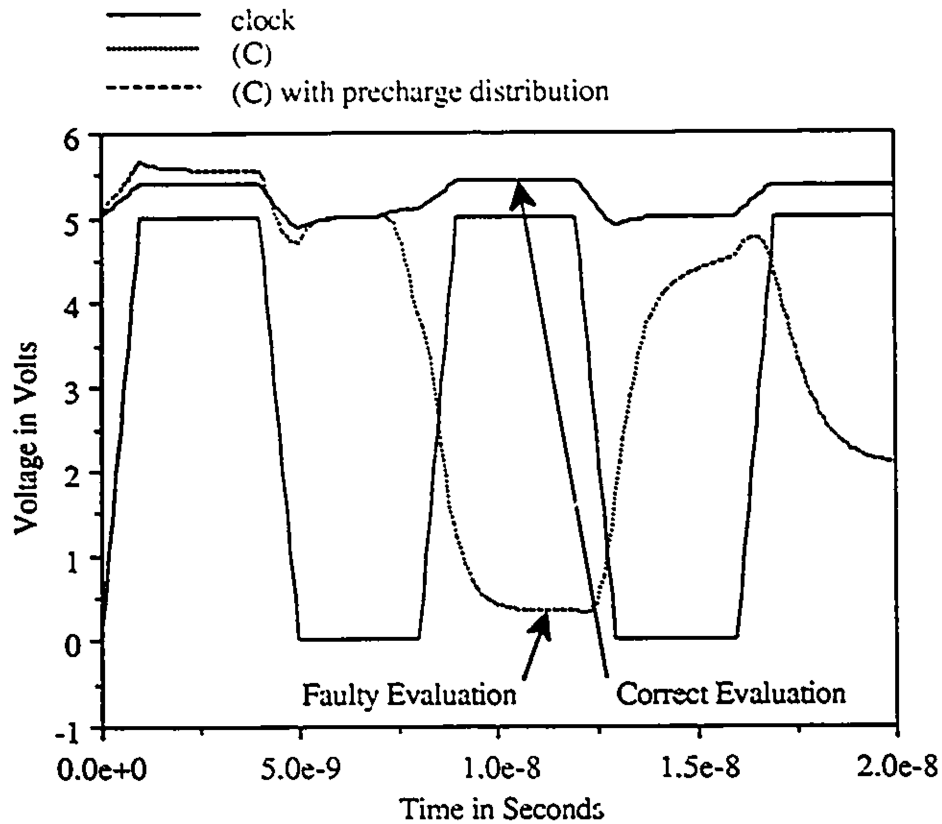

Figure 4.11 Worst Case Charge Sharing Condition of the Circuit (C) with and without Precharge Distribution

4.4 Summary

This chapter has presented a simple precharge model based on the RC model and has successfully demonstrated a technique to size the precharge PFET. Various suggestions are discussed to overcome the inherent charge sharing problem or node coupling of a dynamic structure that not only removes the problem but also stabilizes the evaluation node and, hence, increasing the noise immunity and widening the operating frequency range of the structure.

CHAPTER 5

Conclusions and Further Research

5.1 Introduction

In this chapter, conclusions are drawn from both the research and the literature review to stress the important points in this work and also to pave the way for further research investigations.

5.2 Conclusions

The major conclusions are summarized as follows:

1. The linear RC model, along with the approximate Elmore's delay formula, offers simplicity while providing an acceptable level of accuracy in modeling CMOS digital dynamic circuits for discharge delay approximation.

2. A precharge delay model, based on the RC model, has been successfully implemented to approximate the precharge delay. This model preserves two time constants: one is associated with the precharge PFET and the other is to accommodate the worst case n-logic block condition during the precharge

cycle. This model also resulted in a solvable transcendental equation, and standard numerical techniques such as Newton-Raphson method, have been proposed.

3. Parasitic capacitance approximations of a CMOS transistor have been explained in detail with the analysis based on the transistor layout geometry and SPICE level 2 extraction parameters.

4. Several techniques are presented to approximate the channel resistance of a MOSFET depending on the operating region of the transistor. Channel resistance of the NFETs that mainly operate in the linear region are modelled by taking the minimum resistance of the SPICE level 2 MOSFET equation while the channel resistance of the top NFET of the chain, that mainly operates in the saturation region, is approximated by SPICE level 2 simulations. This represents a new approach to delay approximation, and produces much closer results to those obtained by simulation.

5. Several traditional transistor NFET sizing techniques, based on the RC model and existing optimization algorithms, are implemented for comparison purposes.

6. A novel analytical approach to sizing NFET chains, based on the RC model, is introduced which not only saves CPU time, by avoiding the use of iterative optimization techniques, but also offers an analytical formulation that can be integrated into automated circuit synthesis and layout tools.

7. A second approach to sizing NFET chains is presented which transforms an N-variable optimization problem into a single-variable problem , providing a maximum NFET width constraint is desired rather than an area constraint.

8. Precharge PFET sizing is successfully demonstrated.It utilizes the precharge RC delay model and can be integrated easily with the NFET chain sizing techniques.

9. The charge sharing, or node coupling, problem of a dynamic logic structure, due to the floating nature of the evaluation node, is discussed and several suggestions are given to overcome this problem.

5.3 Further Research

The following is a partial list which suggests future research activities that can be based on this thesis.

1. The development of a more general analytical formulation that could include pass transistors and more complex logic structures such as multiple evaluation nodes.

2. Unifying both facets of the transistor sizing problem by considering the precharge PFET sizing with the NFET sizing in a single analytical approach.

3. Integrating the techniques developed in this thesis with automated logic circuit synthesis and layout tools that are targeted to produce high performance VLSI chips.

4. An in-depth study of the problems associated with dynamic gates such as charge sharing and noise immunity and also a complete analysis of the power consumption of such logic gates.

REFERENCES

[1] M. Afghahi and C. Svensson, "A Unified Single-Phase Clocking Scheme for VLSI Systems", IEEE Journal of Sloid-State Circuits, Vol. 25, No. 1, February, 225-232, 1990.

[2] P. Antognetti and G. Massobrio, "Semiconductor Device Modeling with SPICE", McGraw-Hill Book Company, 1988.

[3] B. R. Chawla, H. K. Gummel and P. Kozak, "MOTIS-An MOS Timing Simulator", IEEE Trans. of circuits and systems, CAS-22, Dec., 901-910, 1975.

[4] C. M. Corporation, "Giude to the Integrated Circuit Implementation Services of the Canadian Microelectronics Corporation", 1980.

[5] W. C. Elmore, "The transit response of damped linear networks with particular regard to wideband amplifiers", J. Appl. Phys., 19, No. 1, Jan., 55-63, 1948.

[6] J. P. Fishburn and A. E. Dunlop, "TILOS: Aposynomial programing approach to transistor sizing", Proc. of the Int. Conf. Computer Aiaded Design, 326-328, 1985.

[7] T. Lin and C. A. Mead, "Signal delay in general RC networks", IEEE Trans. computer-aided design, CAD-3, Oct., 331-349, 1984.

[8] C. Longway and R. Siferd, "A Doughnut Layout Style for Improved Switching Speed with CMOS VLSI Gates", Vol. 24, No. 1, February, 194-198, 1989.

[9] M. D. Matson and L. A. Glasser, "Macromodelling and optimization of digital MOS VLSI circuits", IEEE Trans. on computer aided design, CAD-5, No. 4, Oct., 659-678, 1986.

[10] Carver Mead and Lynn Conway, "Introduction to VLSI Systems", Addison-Wesley, 1980.

[11] L. W. Nagel, "SPICE2: A Computer program to simulate semiconductor circuits", May, 1975.

[12] J. K. Ousterhout, "A Switch-Level Timing Verifier for Digital MOS VLSI", IEEE tran. on computer-aided design, CAD-4, No. 3, July, 336-349, 1985.

[13] J. Rubinstein, P. Penfield and M. A. Horowitz, "Signal Delay in RC Networks", IEEE Trans. computer-aided design, CAD-2, July, 202-211, 1983.

[14] T. Sakurai and A. R. Newton, "Delay Analysis of Series-Connected MOSFET Circuits", IEEE Journal of Solid-State Circuits, 26, NO. 2, FEBRUARY, 122-131, 1991.

[15] M. Shoji, "FET Scaling in Domino CMOS Gates", IEEE Journal of solid-state circuits, SC-20, No. 5, Oct., 1067-71, 1985.

[16] M. Shoji, "CMOS DIGITAL CIRCUIT TECHNOLOGY", Prentice Hall Inc., 1988.

[17] J. M. Shyu, A. Sangiovanni-Vincentelli, J. P. Fishburn and A. E. Dunlop, "Optimization-Based Transistor Sizing", Vol. 23, No. 2, April, pp. 400-409, 1988.

[18] R. Sundblad and C. Svensson, "Fully dynamic switch level simulation of CMOS circuits", IEEE Trans. on computer-aided design, CAD-6, No. 2, March, 282-89, 1987.

[19] A. Vladimirescu and S. Liu, "The Simulation of MOS Integrated Circuits using SPICE2", 1980.

[20] W. T. Weeks, A. J. Jiminez, G. W. Mahoney, D. Mehta, H. Qassamzadeh and T. R. Scott, "Algorithms for ASTAP-A network analysis program", IEEE Trans. of circuit theory, vol 20, No. 6, Nov., 628-634, 1973.

[21] C. Y. Wu, J. S. Hwang, C. Chang and C. C. Chang, "An efficient timing model for CMOS cominatorial logic gates", IEEE Trans. on Computer-Aided Design, CAD-4, October, 636-650, 1985.

[22] J. Yuan and C. Svensson, "CMOS Circuit Speed Optimization Based on Switch Level Simulation", Proc. of the IEEE International Symposium on Circuits and Systems., 3, 2109-2112, 1988.

[23] J. Yuan and C. Svensson, "High-Speed CMOS Circuit Technique", IEEE Journal of Solid-State Circuits, Vol. 24, No. 1, February, 62-69, 1989.

[24] L. M. Brocco, S. P. Mccormick and J. Allen, "Macromodeling CMOS Circuits for Timing Simulation," IEEE Tran. on Computer-aided Design, vol. 7. No. 12, December, 1237-1249, 1988.

Appendix A

3μ-DLM Technology SPICE Parameters and RC Model Computed Parameters

A.1 SPICE Transistor Parameters

Parameter	NMOS	PMOS	Units
VTO	0.7	-0.8	V
KP	40E-6	12E-6	(A/V^2)
GAMMA	1.1	0.6	$(V^{0.5})$
PHI	0.6	0.6	V
LAMBDA	0.01	0.03	1/V
RD	40	100	ohms
RS	40	100	ohms
CBD			F
CBS			F
IS			A
PB	0.7	0.6	V
CGSO	3.0E-10	2.5E-10	F/m
CGDO	3.0E-10	2.5E-10	F/m
CGBO	5.0E-10	5.0E-10	F/m
RSH	25	80	ohms/sq.
CJ	4.4E-4	1.5E-4	(F/m^2)
MJ	0.5	0.6	-
CJSW	4.0E-10	4.0E-10	F/m
MJSW	0.3	0.6	-
JS	1.0E-5	1.0E-5	(A/m^2)
TOX	5.0E-8	5.0E-8	m

NSUB	1.7E16	5.0E15	($1/cm^2$)
NSS	0	0	($1/cm^2$)
NFS	0	0	($1/cm^2$)
TPG	1	1	-
XJ	6.0E-7	5.0E-7	m
LD	3.5E-7	2.5E-7	m
UO	775	250	($cm^2/V\ s$)
VMAX	1.0E5	0.7E5	m/s
Gate (COX)	6.9E-4	6.9E-4	($pF/\mu m^2$)
Gate (COXEdge)	0.5E-4	0.5E-4	($pF/\mu m$)

A.2 RC Model Computed Parameters

Parameter	Computed Value	Units
PURL	1.403E-2	ohm·m
PURS	2.385E-2	ohm·m
PURP	8.820E-2	ohm·m
K_1 (W_P=5.4μ)	5.088E-14	F
K_1 (W_P=16.8μ)	6.864E-14	F
K_1 (W_P=30μ)	8.920E-14	F
K_2	3.105E-9	F/m
K_3	3.178E-14	F
K_4	3.105E-9	F/m
K_5	3.105E-9	F/m
K_6 (W_N=5.4μ)	5.924E-14	F

K_7	1.558E-9	F/m

Appendix B

NFET Chain Sizing Source Codes

B.1 OPTIMIZATION I Source Code

```
C**********************************************
C                  OPTIMIZATION I              *
C      This program sizes NFET chains according to a    *
C user specified maximum transistor width constraint    *
C and uses direct search technique for the optimization *
C procedure.                                            *
C**********************************************

       INTEGER N,I,LIMIT,ITER
       IMPLICIT REAL (A-Z)
       REAL WI(10),W(10)
       COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

C...   OPEN OUTPUT FILE

       OPEN(UNIT=2,FILE='optimizationI.o',STATUS='NEW')
       WRITE(2,*) '******** OPTIMIZATION I RESULTS ********'
       WRITE(2,*) '******** OPTIMIZATION I RESULTS ********'
       WRITE(2,*)

C...   INTIALIZE VARIABLES

       LIMIT=500
       WMIN=5.4e-6
       WMAX=30.0e-6
       L=3.0e-6
       STEP=0.6e-6
       WP=10.0e-6
       WLP=WMIN
       WLN=WMIN
       VBN=0.0
       VBP=5.0
       N=7

C...   CALCULATE THE PER UNIT RESISTANCES

       PURL=NFET_MIN_RESISTANCE(1.0,5.0,0.0)
       PURS=PURL*1.7

C...   CALCULATE INTIAL NFET SIZES BASED ON LINEAR TAPPERING

       CC=(WMAX-WMIN)/(N-1)
       DO I=1,N
       WI(I)=WMAX+CC-CC*I
       ENDDO
```

```
C...  CONVERT INITIAL SIZE VECTOR TO STEPS OF MICRONS

      DO I=1,N
      WI(I)=STEP*AINT(WI(I)/STEP+0.5)
      IF(WI(I).LT.WMIN) WI(I)=WMIN
      IF(WI(I).GT.WMAX) WI(I)=WMAX
      ENDDO

C...  DOCUMENT PROBLEM SET UP AND PARAMETERS

      WRITE(2,*)' PURL  =',PURL
      WRITE(2,*)' PURS  =',PURS
      WRITE(2,*)' LIMIT        =',LIMIT
      WRITE(2,*)' WMIN         =',WMIN
      WRITE(2,*)' WMAX         =',WMAX
      WRITE(2,*)' N      =',N
      WRITE(2,*)' L      =',L
      WRITE(2,*)' STEP  =',STEP
      WRITE(2,*)' WP    =',WP
      WRITE(2,*)' WLP   =',WLP
      WRITE(2,*)' WLN   =',WLN

C...  DOCUMENT INITIAL SIZE VECTOR

      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR  LEVEL=',I,'    INITIAL  TRANSISTOR
SIZE=',WI(I)
      ENDDO

      CALL OPTMIZEI(N,W,WI,TD,LIMIT,ITER)

C...  DOCUMENT SOLUTION

      WRITE(2,*)
      WRITE(2,*)'  ITERATIONS PERFORMED        =',ITER
      WRITE(2,*)
      WRITE(2,*)
      WRITE(2,*)'  MINIMUM DISCHARGE DELAY  =',TD
      WRITE(2,*)

C...  PRINT OPTMIZED VECTOR AND TOTAL GATE AREA

      WTOT=0.0
      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR LEVEL=',I,'  TRANSISTOR SIZE=',W(I)
      WTOT=WTOT+W(I)
      ENDDO
      WRITE(2,*)
      WRITE(2,*)'  TOTAL AREA OF THE CHAIN      =',WTOT*L
      WRITE(2,*)

      STOP
      END
```

```
C**********************************************
C              OPTIMIZATION ALGORITHM        *
C**********************************************

      SUBROUTINE OPTMIZEI(N,W,WI,TD,LIMIT,ITER)

      IMPLICIT REAL (A-Z)
      INTEGER N,I,j,LIMIT,ITER
      REAL WI(N),W(N)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

      ITER=0
 10   ITER=ITER+1
      IF(ITER.GT.LIMIT) RETURN

      DO I=1,N
      W(I)=WI(I)
      ENDDO

      TD=DELAY(N,W)
      DO j=1,N
            Wj=W(j)
            Wjp=Wj+STEP*.5
            W(j)=Wjp
            TDp=DELAY(N,W)
            Wjn=Wj-STEP*.5
            W(j)=Wjn
            TDn=DELAY(N,W)
            IF(TDp.LT.TD.AND.Wjp.LT.WMAX) THEN
            WI(j)=Wjp
            ELSEIF(TDn.LT.TD.AND.Wjn.GT.WMIN) THEN
            WI(j)=Wjn
            ELSE
            WI(j)=Wj
            ENDIF
            W(j)=Wj
      ENDDO

C...  TEST FOR CONVERGENCE

      DO I=1,N
      IF(WI(I).NE.W(I)) GOTO 10
      ENDDO

C...  CONVERT FINAL SIZE VECTOR TO STEPS OF MICRONS
C     AND RECALCULATE THE DELAY

      DO I=1,N
      W(I)=STEP*AINT(W(I)/STEP+0.5)
      IF(W(I).LT.WMIN) W(I)=WMIN
      IF(W(I).GT.WMAX) W(I)=WMAX
      ENDDO
      TD=DELAY(N,W)
```

```
      RETURN
      END

C*******************************************
C              NFET DELAY                  *
C       This routine approximates the discharge   *
C delay of an N-level NFET chain by using the RC  *
C model along with Elmore's delay formula.        *
C*******************************************

      FUNCTION DELAY(N,W)

      IMPLICIT REAL (A-Z)
      INTEGER N,I,J
      REAL W(N),R(10),C(10)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

C...  Initialize variables

      DO J=1,N
      R(J) =0.0
      C(J) =0.0
      ENDDO


C...  Assign node capacitances and channel resistances of
C     all transistors to the RC model except the top one.

      DO I=1,N-1
      C(I)=NSCAP(W(I+1),0.0,VBN)+.5*NGCAP(W(I+1))+NGSOCAP(W(I+1))
   +  +NSCAP(W(I),0.0,VBN)+.5*NGCAP(W(I))+NGDOCAP(W(I))
      R(I)=PURL/W(I)
      ENDDO

C...  Assign node capacitance and channel resistance of the
C     top transistor to the RC model.

      C(N)=NSCAP(W(N),0.0,VBN)+.5*NGCAP(W(N))+NGDOCAP(W(N))
   +
      +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)+NGCAP(WLN)+P
GCAP(WLP)
      R(N)=PURS/W(N)

C...  Now, calculate the delay using Elmore's formula.

      TD=0.0
      DO I=1,N
      CABOVE=0.0
      DO J=I,N
      CABOVE=CABOVE+C(J)
      ENDDO
      TD=TD+R(I)*CABOVE
```

```
      ENDDO

C...  Return the computed delay

      DELAY=TD*1.175
      RETURN
      END

C*****************************************
C              SUBROUTINE NSCAP          *
C                                        *
C      This subroutine evaluates NMOSFET source*
C capacitance using standard layout geometry and  *
C the given fabrication technology  parameters.   *
C*****************************************

      FUNCTION NSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)

      VBS=VB-VS

            CGSO=3.0E-10
            CJSW=4.0E-10
            CJ=4.4E-4
            MJ=0.5
            MJSW=0.3
            PB=0.7
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VBS.LE.FC*PB) THEN
      NSCAP=CJ*AS/(1.0-VBS/PB)**MJ+CJSW*PS/(1.0-VBS/PB)**MJSW
      ELSE
      NSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VBS/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VBS/PB*MJSW)
      ENDIF

      RETURN
      END

C*****************************************
C              SUBROUTINE PSCAP          *
C                                        *
C      This subroutine evaluates PMOSFET source  *
C capacitance using standard layout geometry and  *
C the given fabrication technology  parameters.   *
C*****************************************

      FUNCTION PSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)
```

```
      VSB=VS-VB

            CGSO=2.5E-10
            CJSW=4.0E-10
            CJ=1.5E-4
            MJ=0.6
            MJSW=0.6
            PB=0.6
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VSB.LE.FC*PB) THEN
      PSCAP=CJ*AS/(1.0-VSB/PB)**MJ+CJSW*PS/(1.0-VSB/PB)**MJSW
      ELSE
      PSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VSB/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VSB/PB*MJSW)
      ENDIF

      RETURN
      END


C******************************************
C             NGCAP SUBROUTINE             *
C                                          *
C      This subroutine evaluates NMOSFET gate   *
C capacitance using standard layout geometry and  *
C the given fabrication technology  parameters.    *
C******************************************

      FUNCTION NGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      NGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END
C******************************************
C             PGCAP SUBROUTINE             *
C                                          *
C      This subroutine evaluates PMOSFET gate   *
C capacitance using standard layout geometry and  *
C the given fabrication technology  parameters.    *
C******************************************

      FUNCTION PGCAP(W)
```

```
      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

            COX=6.9E-4
            COXE=0.5E-10

      PGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END
C*****************************************
C              PGSOCAP SUBROUTINE        *
C                                        *
C      This subroutine evaluates PMOSFET gate-  *
C source overlap capacitance using standard     *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C*****************************************

      FUNCTION PGSOCAP(W)

      IMPLICIT REAL (A-Z)

            CGSO=2.5E-10

      PGSOCAP=CGSO*W

      RETURN
      END
C*****************************************
C              PGDOCAP SUBROUTINE        *
C                                        *
C      This subroutine evaluates PMOSFET gate-  *
C drain overlap capacitance using standard      *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C*****************************************

      FUNCTION PGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=2.5E-10

      PGDOCAP=CGDO*W

      RETURN
      END
```

```
C*****************************************
C              NGSOCAP SUBROUTINE         *
C                                         *
C       This subroutine evaluates PMOSFET gate-  *
C source overlap capacitance using standard       *
C layout geometry and the give fabrication        *
C technology parameters.                          *
C*****************************************

      FUNCTION NGSOCAP(W)

      IMPLICIT REAL (A-Z)

            CGSO=3.0E-10

      NGSOCAP=CGSO*W

      RETURN
      END

C*****************************************
C              NGDOCAP SUBROUTINE         *
C                                         *
C       This subroutine evaluates NMOSFET gate-  *
C drain overlap capacitance using standard        *
C layout geometry and the give fabrication        *
C technology parameters.                          *
C*****************************************

      FUNCTION NGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=3.0E-10

      NGDOCAP=CGDO*W

      RETURN
      END

C*****************************************
C              GEOM SUBROUTINE            *
C                                         *
C       This subroutine calculates NFET/PFET     *
C drain/source area and perimeter based on a      *
C standard layout technique and the given         *
C fabrication technology rules.                   *
C*****************************************

      SUBROUTINE GEOM(W,AS,PS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
```

```
      PS=W+14.4E-6
      AS=W*L+22.68E-12

      RETURN
      END

C*****************************************
C     NFET_MIN_RESISTANCE SUBROUTINE*
C                                       *
C     This subroutine approximates the per    *
C unit channel resistance for those NFETs that  *
C mainly discharge in the linear region.      *
C*****************************************

      FUNCTION NFET_MIN_RESISTANCE(W,VG,VS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      VGS=VG-VS
      VBS=VBN-VS

            EPSSI=11.7*8.85E-12
            Q=1.6022E-19
            UO=775E-4
            VTO=0.7
            VMAX=1.0E5
            PHI=0.6
            COX=6.9E-4
            NSUB=1.7E22
            XJL=3.5E-7
            GAMMA=1.1
            XJ=6.0E-7
            ETA=1.0


C...  CALCULATE INTERMEDIATE PARAMETERS

      US=UO
      PHIF=PHI/2.0
      VFB=VTO-PHI-2.0*SQRT(Q*EPSSI*NSUB*PHIF)/COX
      LEFF=L-2.0*XJL
      BETA=W/LEFF*US*COX
      VBI=VFB+PHI
      VBIN=VBI
      XD=SQRT(2.0*EPSSI/Q/NSUB)
      WD=XD*SQRT(PHI-VBS+VDS)
      WS=XD*SQRT(PHI-VBS)
      ALFAD=0.5*XJ/LEFF*(SQRT(1.0+2*WD/XJ)-1.0)
      ALFAS=0.5*XJ/LEFF*(SQRT(1.0+2*WS/XJ)-1.0)
      GAMMAS=GAMMA*(1.0-ALFAS-ALFAD)
```

```
      NFET_MIN_RESISTANCE=1.316/BETA/(VGS-VBIN-GAMMAS*(2*PHI-
VBS)**0.5)

      RETURN
      END

C***************END OF OPTIMIZATION I PROGRAM***************
```
C***************END OF OPTIMIZATION I PROGRAM***************

B.2 OPTIMIZATION II Source Code

```
C***********************************************
C                    OPTIMIZATION II           *
C                                              *
C       This FORTRAN code sizes NFET chains using        *
C maximum gate area constraint and implements maximum  *
C sensitivity approach for the optimization procedure.  *
C***********************************************

      IMPLICIT REAL (A-Z)
      INTEGER N,I,LIMIT,ITER
      REAL WI(10),W(10)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

C...  OPEN OUTPUT FILE

      OPEN(UNIT=2,FILE='optimizationII.o',STATUS='NEW')
      WRITE(2,*) '******** OPTIMIZATION II RESULTS ********'
      WRITE(2,*) '******** OPTIMIZATION II RESULTS ********'
      WRITE(2,*)

C...  INTIALIZE VARIABLES AND SET UP PROBLEM

      LIMIT=1000
      L=3.0e-6
      WMIN=5.4E-6
      STEP=0.6E-6
      WP=10.0e-6
      WLP=WMIN
      WLN=WMIN
      VBN=0.0
      VBP=5.0
      N=7
      MAXAREA=210e-6*L

C...  CALCULATE THE PER UNIT RESISTANCES

      PURL=NFET_MIN_RESISTANCE(1.0,5.0,0.0)
      PURS=PURL*1.7

C...  SET INTIAL FET SIZES TO MINIMUM WIDTH

      DO I=1,N
      WI(I)=WMIN
      ENDDO

C...  DOCUMENT PROBLEM SET UP AND PARAMETERS
```

```
      WRITE(2,*)' PURL =',PURL
      WRITE(2,*)' PURS =',PURS
      WRITE(2,*)' LIMIT        =',LIMIT
      WRITE(2,*)' WMIN         =',WMIN
      WRITE(2,*)' MAXAREA      =',MAXAREA
      WRITE(2,*)' N      =',N
      WRITE(2,*)' L      =',L
      WRITE(2,*)' STEP  =',STEP
      WRITE(2,*)' WP     =',WP
      WRITE(2,*)' WLP   =',WLP
      WRITE(2,*)' WLN   =',WLN

C...  DOCUMENT INITIAL SIZE VECTOR

      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR   LEVEL=',I,'     INITIAL   TRANSISTOR
SIZE=',WI(I)
      ENDDO

      CALL OPTMIZEII(N,W,WI,TD,MAXAREA,AREA,LIMIT,ITER)


C...  DOCUMENT SOLUTION IN A FILE

      WRITE(2,*)
      WRITE(2,*) 'LIMIT=',LIMIT,'     ITERATIONS PERFORMED=',ITER
      WRITE(2,*)
      WRITE(2,*)
      WRITE(2,*) 'MINIMUM DISCHARGE DELAY=',TD
      WRITE(2,*)

C...print optimized vector and the final gate area

      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR LEVEL=',I,'  TRANSISTOR SIZE=',W(I)
      ENDDO
      WRITE(2,*)
      WRITE(2,*)'MAXIMUM GATE AREA SPECIFIED=',MAXAREA
      WRITE(2,*)'RESULTED GATE AREA        =',AREA
      WRITE(2,*)

      STOP
      END


C***********************************************
C              OPTIMIZATION ALGORITHM              *
C***********************************************

      SUBROUTINE OPTMIZEII(N,W,WI,TD,MAXAREA,AREA,LIMIT,ITER)

      IMPLICIT REAL (A-Z)
      REAL WI(N),W(N)
```

```
      INTEGER N,I,j,LIMIT,ITER,INDEX
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

      AREA=0.0
      DO I=1,N
      W(I)=WI(I)
      AREA=AREA+W(I)*L
      ENDDO

      ITER=0
10    ITER=ITER+1
      IF(ITER.GT.LIMIT) RETURN
      TD=DELAY(N,W)
      MAXSENS=0.0

      DO j=1,N
            Wj=W(j)
            W(j)=Wj+STEP
            TDp=DELAY(N,W)
            SENS=TD-TDp
            IF(SENS.GT.MAXSENS) THEN
            MAXSENS=SENS
            INDEX=j
            ENDIF
            W(j)=Wj
      ENDDO

c...  INCREASE ONLY THE FET WITH MAXIMUM SENSITIVITY
C     AND CHECK AREA LIMIT

      W(INDEX)=W(INDEX)+STEP
      AREA=AREA+STEP*L
      IF(AREA.LT.MAXAREA) GOTO 10

C...  CONVERT FINAL SIZE VECTOR TO STEPS OF MICRONS
C     AND RECALCULATE THE DELAY AND AREA

      AREA=0.0
      DO I=1,N
      W(I)=STEP*AINT(W(I)/STEP+0.5)
      IF(W(I).LT.WMIN) W(I)=WMIN
      AREA=AREA+W(I)
      ENDDO
      AREA=AREA*L
      TD=DELAY(N,W)

      RETURN
      END

C*****************************************
C              NFET DELAY                *
C       This routine approximates the discharge    *
C delay of an N-level NFET chain by using the RC   *
C model along with Elmore's delay formula.         *
```

Appendix B

```
C******************************************

      FUNCTION DELAY(N,W)

      IMPLICIT REAL (A-Z)
      INTEGER N,I,J
      REAL W(N),R(10),C(10)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

C...  Initialize variables

      DO J=1,N
      R(J) =0.0
      C(J) =0.0
      ENDDO


C...  Assign node capacitances and channel resistances of
C     all transistors to the RC model except the top one.

      DO I=1,N-1
      C(I)=NSCAP(W(I+1),0.0,VBN)+.5*NGCAP(W(I+1))+NGSOCAP(W(I+1))
    + +NSCAP(W(I),0.0,VBN)+.5*NGCAP(W(I))+NGDOCAP(W(I))
      R(I)=PURL/W(I)
      ENDDO

C...  Assign node capacitance and channel resistance of the
C     top transistor to the RC model.

      C(N)=NSCAP(W(N),0.0,VBN)+.5*NGCAP(W(N))+NGDOCAP(W(N))
    +
      +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+FGDOCAP(WP)+NGCAP(WLN)+P
GCAP(WLP)
      R(N)=PURS/W(N)

C...  Now, calculate the delay using Elmore's formula.

      TD=0.0
      DO I=1,N
      CABOVE=0.0
      DO J=I,N
      CABOVE=CABOVE+C(J)
      ENDDO
      TD=TD+R(I)*CABOVE
      ENDDO

C...  Return the computed delay

      DELAY=TD*1.175
      RETURN
      END


C******************************************
```

```
C              SUBROUTINE NSCAP              *
C                                            *
C       This subroutine evaluates NMOSFET source*
C capacitance using standard layout geometry and   *
C the given fabrication technology  parameters.    *
C*****************************************

      FUNCTION NSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)

      VBS=VB-VS

            CGSO=3.0E-10
            CJSW=4.0E-10
            CJ=4.4E-4
            MJ=0.5
            MJSW=0.3
            PB=0.7
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VBS.LE.FC*PB) THEN
      NSCAP=CJ*AS/(1.0-VBS/PB)**MJ+CJSW*PS/(1.0-VBS/PB)**MJSW
      ELSE
      NSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VBS/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VBS/PB*MJSW)
      ENDIF

      RETURN
      END

C*****************************************
C              SUBROUTINE PSCAP              *
C                                            *
C       This subroutine evaluates PMOSFET source  *
C capacitance using standard layout geometry and   *
C the given fabrication technology  parameters.    *
C*****************************************

      FUNCTION PSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)
      VSB=VS-VB

            CGSO=2.5E-10
            CJSW=4.0E-10
            CJ=1.5E-4
            MJ=0.6
            MJSW=0.6
            PB=0.6
            FC=0.5
```

```
      CALL GEOM(W,AS,PS)

      IF(VSB.LE.FC*PB) THEN
      PSCAP=CJ*AS/(1.0-VSB/PB)**MJ+CJSW*PS/(1.0-VSB/PB)**MJSW
      ELSE
      PSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VSB/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VSB/PB*MJSW)
      ENDIF

      RETURN
      END


C*******************************************
C              NGCAP SUBROUTINE            *
C                                          *
C       This subroutine evaluates NMOSFET gate  *
C capacitance using standard layout geometry and  *
C the given fabrication technology  parameters.   *
C*******************************************

      FUNCTION NGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON  PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      NGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C*******************************************
C              PGCAP SUBROUTINE            *
C                                          *
C       This subroutine evaluates PMOSFET gate  *
C capacitance using standard layout geometry and  *
C the given fabrication technology  parameters.   *
C*******************************************

      FUNCTION PGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON  PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      PGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE
```

```
      RETURN
      END

C******************************************
C             PGSOCAP SUBROUTINE          *
C                                         *
C     This subroutine evaluates PMOSFET gate-  *
C source overlap capacitance using standard    *
C layout geometry and the give fabrication     *
C technology parameters.                       *
C******************************************

      FUNCTION PGSOCAP(W)

      IMPLICIT REAL (A-Z)


            CGSO=2.5E-10

      PGSOCAP=CGSO*W

      RETURN
      END

C******************************************
C             PGDOCAP SUBROUTINE          *
C                                         *
C     This subroutine evaluates PMOSFET gate-  *
C drain overlap capacitance using standard     *
C layout geometry and the give fabrication     *
C technology parameters.                       *
C******************************************

      FUNCTION PGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=2.5E-10

      PGDOCAP=CGDO*W

      RETURN
      END

C******************************************
C             NGSOCAP SUBROUTINE          *
C                                         *
C     This subroutine evaluates PMOSFET gate-  *
C source overlap capacitance using standard    *
C layout geometry and the give fabrication     *
C technology parameters.                       *
C******************************************

      FUNCTION NGSOCAP(W)
```

```
      IMPLICIT REAL (A-Z)

            CGSO=3.0E-10

      NGSOCAP=CGSO*W

      RETURN
      END
C*****************************************
C              NGDOCAP SUBROUTINE         *
C                                         *
C       This subroutine evaluates NMOSFET gate-  *
C drain overlap capacitance using standard *
C layout geometry and the give fabrication *
C technology parameters.                   *
C*****************************************

      FUNCTION NGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=3.0E-10

      NGDOCAP=CGDO*W

      RETURN
      END
C*****************************************
C              GEOM SUBROUTINE            *
C                                         *
C       This subroutine calculates NFET/PFET  *
C drain/source area and perimeter based on a *
C standard layout technique and the given  *
C fabrication technology rules.            *
C*****************************************

      SUBROUTINE GEOM(W,AS,PS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      PS=W+14.4E-6
      AS=W*L+22.68E-12

      RETURN
      END
C*****************************************
C       NFET_MIN_RESISTANCE SUBROUTINE*
C                                         *
```

```
C       This subroutine approximates the per          *
C unit channel resistance for those NFETs that        *
C mainly discharge in the linear region.              *
C*****************************************

        FUNCTION NFET_MIN_RESISTANCE(W,VG,VS)

        IMPLICIT REAL (A-Z)
        COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


        VGS=VG-VS
        VBS=VBN-VS

                EPSSI=11.7*8.85E-12
                Q=1.6022E-19
                UO=775E-4
                VTO=0.7
                VMAX=1.0E5
                PHI=0.6
                COX=6.9E-4
                NSUB=1.7E22
                XJL=3.5E-7
                GAMMA=1.1
                XJ=6.0E-7
                ETA=1.0


C...    CALCULATE INTERMEDIATE PARAMETERS

        US=UO
        PHIF=PHI/2.0
        VFB=VTO-PHI-2.0*SQRT(Q*EPSSI*NSUB*PHIF)/COX
        LEFF=L-2.0*XJL
        BETA=W/LEFF*US*COX
        VBI=VFB+PHI
        VBIN=VBI
        XD=SQRT(2.0*EPSSI/Q/NSUB)
        WD=XD*SQRT(PHI-VBS+VDS)
        WS=XD*SQRT(PHI-VBS)
        ALFAD=0.5*XJ/LEFF*(SQRT(1.0+2*WD/XJ)-1.0)
        ALFAS=0.5*XJ/LEFF*(SQRT(1.0+2*WS/XJ)-1.0)
        GAMMAS=GAMMA*(1.0-ALFAS-ALFAD)

        NFET_MIN_RESISTANCE=1.316/BETA/(VGS-VBIN-GAMMAS*(2*PHI-
VBS)**0.5)

        RETURN
        END

C**************END OF OPTIMIZATION II PROGRAM***************
C**************END OF OPTIMIZATION II PROGRAM***************
```

B.3 SINGLE-VARIABLE OPTIMIZATION Source Code

```
C*******************************************
C       SINGLE VARIABLE OPTIMIZATION       *
C                                          *
C       This code sizes an NFET chain with a single *
C variable optimization. The rest of the chain widths *
C are solved for analytically during the optimization *
C process.                                 *
C*******************************************

        IMPLICIT REAL (A-Z)
        INTEGER N,I,LIMIT,ITER
        REAL WI,W(10)
        COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
     +  ,K1,K2,K3,K4,K5

C...    OPEN OUTPUT FILE

        OPEN(UNIT=2,NAME='s_v_optimization.o',STATUS='NEW')
        WRITE(2,*) '******** SINGLE VARIABLE OPTIMIZATION RESULTS
********'
        WRITE(2,*) '******** SINGLE VARIABLE OPTIMIZATION RESULTS
********'
        WRITE(2,*)

C...    INTIALIZE VARIABLES AND SET UP PROBLEM

        LIMIT=100
        L=3.0E-6
        WMIN=5.4E-6
        WMAX=30.0E-6
        STEP=0.6E-6
        WP=WMAX
        WLP=WMIN
        WLN=WMIN
        VBN=0.0
        VBP=5.0
        N=7
        WI=WMIN

C...    CALCULATE THE PER UNIT RESISTANCES

        PURL=NFET_MIN_RESISTANCE(1.0,5.0,0.0)
        PURS=PURL*1.7
```

```
      CALL S_V_OPTMIZE(N,WI,W,TD,LIMIT,ITER)

C...  DOCUMENT PROBLEM SET UP AND PARAMETERS

      WRITE(2,*)' PURL =',PURL
      WRITE(2,*)' PURS =',PURS
      WRITE(2,*)' LIMIT     =',LIMIT
      WRITE(2,*)' WMIN      =',WMIN
      WRITE(2,*)' WMAX      =',WMAX
      WRITE(2,*)' N    =',N
      WRITE(2,*)' L    =',L
      WRITE(2,*)' STEP =',STEP
      WRITE(2,*)' WP   =',WP
      WRITE(2,*)' WLP  =',WLP
      WRITE(2,*)' WLN  =',WLN
      WRITE(2,*)' K1   =',K1
      WRITE(2,*)' K2   =',K2
      WRITE(2,*)' K3   =',K3
      WRITE(2,*)' K4   =',K4
      WRITE(2,*)' K5   =',K5

C...  DOCUMENT SOLUTION

      WRITE(2,*)
      WRITE(2,*)' ITERATIONS PERFORMED        =',ITER
      WRITE(2,*)
      WRITE(2,*)
      WRITE(2,*)' MINIMUM DISCHARGE DELAY   =',TD
      WRITE(2,*)

C...  PRINT OPTMIZED VECTOR AND TOTAL GATE AREA

      WTOT=0.0
      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR LEVEL=',I,'  TRANSISTOR SIZE=',W(I)
      WTOT=WTOT+W(I)
      ENDDO
      WRITE(2,*)
      WRITE(2,*)' TOTAL AREA OF THE CHAIN      =',WTOT*L
      WRITE(2,*)

      STOP
      END

C************************************************
C              OPTIMIZATION ALGORITHM          *
C************************************************

      SUBROUTINE S_V_OPTMIZE(N,WI,W,TD,LIMIT,ITER)

      IMPLICIT REAL (A-Z)
      INTEGER N,I,LIMIT,ITER
      REAL WI,W(N)
```

```
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

     W(N)=WI
      ITER=0
 10   ITER=ITER+1
      IF(ITER.GT.LIMIT) RETURN
      TD=SIZE_DELAY(N,W,ITER)
      Wj=W(N)
      Wjp=Wj+STEP*.5
      W(N)=Wjp
      TDp=SIZE_DELAY(N,W,ITER)
      Wjn=Wj-STEP*.5
      W(N)=Wjn
      TDn=SIZE_DELAY(N,W,ITER)
      IF(TDp.LT.TD.AND.Wjp.LT.WMAX) THEN
      W(N)=Wjp
      ELSEIF(TDn.LT.TD.AND.Wjn.GT.WMIN) THEN
      W(N)=Wjn
      ELSE
      W(N)=Wj
      ENDIF

C...  TEST FOR CONVERGENCE

      IF(W(N).NE.Wj) GOTO 10

C...  CONVERT FINAL SIZE VECTOR TO STEPS OF MICRONS
C     AND RECALCULATE THE DELAY

      DO I=1,N
      W(I)=STEP*AINT(W(I)/STEP+0.5)
      IF(W(I).LT.WMIN) W(I)=WMIN
      IF(W(I).GT.WMAX) W(I)=WMAX
      ENDDO
      TD=SIZE_DELAY(N,W,0)

      RETURN
      END

C*****************************************
C            SIZE_DELAY SUBROUTINE        *
C     This routine computes the NFET chain *
C widths using the analytical formula except for *
C the top NFET which is selected by the optimiz- *
c ation routine. This routine also returns the *
C discharge delay of the chain using the RC *
C model and Elmore's formula              *
C*****************************************

      FUNCTION SIZE_DELAY(N,W,FLAG)

      IMPLICIT REAL (A-Z)
      INTEGER N,I,FLAG
      REAL W(N),R(10),C(10)
```

```
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
   +  ,K1,K2,K3,K4,K5

C...  INTIALIZE VARIABLES

      DO I=1,N
      R(I) =0.0
      C(I) =0.0
      ENDDO

C...  CALCULATE CAPACITANCE CONSTANTS K1, K2, K3, K4, AND K5
C     ONLY ONCE

      IF(FLAG.EQ.1) THEN
      WN=0.0
      K1=NSCAP(WN,0.0,VBN)+.5*NGCAP(WN)+NGDOCAP(WN)
   +  +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
   +  +NGCAP(WLN)+PGCAP(WLP)
      WN=WMIN
      CN_WN=NSCAP(WN,0.0,VBN)+.5*NGCAP(WN)+NGDOCAP(WN)
   +  +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
   +  +NGCAP(WLN)+PGCAP(WLP)
      K2=(CN_WN-K1)/WN

      W1=0.0
      W2=0.0
      K3=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
   +  +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
      W1=WMIN
      W2=0.0
      C_W1=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
   +  +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
      K4=(C_W1-K3)/W1
      W1=0.0
      W2=WMIN
      C_W2=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
   +  +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
      K5=(C_W2-K3)/W2

C...  RECALCULATE THE DELAY ONLY FOR THE FINAL SIZE VECTOR

      ELSEIF(FLAG.EQ.0) THEN
            C(N)=K1+K2*W(N)
            R(N)=PURS/W(N)
            DO I=N-1,1,-1
            C(I)=K3+K4*W(I)+K5*W(I+1)
            R(I)=PURL/W(I)
            ENDDO
            GOTO 20
      ENDIF

C...  SIZE THE REST OF THE CHAIN ANALYTICALY

      C(N)=K1+K2*W(N)
```

```
      R(N)=PURS/W(N)
      CABOVE=C(N)
      TDTERM=R(N)*C(N)
      DO I=N-1,1,-1
      W(I)=PURL*(K3+K5*W(I+1)+CABOVE)/(TDTERM-PURL*K4)
      IF(W(I).GT.WMAX) W(I)=WMAX
      IF(W(I).LT.WMIN) W(I)=WMIN
      C(I)=K3+K4*W(I)+K5*W(I+1)
      R(I)=PURL/W(I)
      CABOVE=CABOVE+C(I)
      ENDDO

C...  NOW, CALCULATE THE DELAY USING ELMORE'S DELAY FORMULA

 20   TD=0.0
      DO I=1,N
      CABOVE=0.0
      DO J=I,N
      CABOVE=CABOVE+C(J)
      ENDDO
      TD=TD+R(I)*CABOVE
      ENDDO

C...  RETURN THE SIZES AND THE DELAY

      SIZE_DELAY=TD*1.175

      RETURN
      END

C******************************************
C              SUBROUTINE NSCAP           *
C                                         *
C     This subroutine evaluates NMOSFET source*
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C******************************************

      FUNCTION NSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)

      VBS=VB-VS

              CGSO=3.0E-10
              CJSW=4.0E-10
              CJ=4.4E-4
              MJ=0.5
              MJSW=0.3
              PB=0.7
              FC=0.5

      CALL GEOM(W,AS,PS)
```

```
      IF(VBS.LE.FC*PB) THEN
      NSCAP=CJ*AS/(1.0-VBS/PB)**MJ+CJSW*PS/(1.0-VBS/PB)**MJSW
      ELSE
      NSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VBS/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VBS/PB*MJSW)
      ENDIF

      RETURN
      END

C******************************************
C             SUBROUTINE PSCAP               *
C                                            *
C     This subroutine evaluates PMOSFET source *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C******************************************

      FUNCTION PSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)
      VSB=VS-VB

              CGSO=2.5E-10
              CJSW=4.0E-10
              CJ=1.5E-4
              MJ=0.6
              MJSW=0.6
              PB=0.6
              FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VSB.LE.FC*PB) THEN
      PSCAP=CJ*AS/(1.0-VSB/PB)**MJ+CJSW*PS/(1.0-VSB/PB)**MJSW
      ELSE
      PSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VSB/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VSB/PB*MJSW)
      ENDIF

      RETURN
      END


C******************************************
C             NGCAP SUBROUTINE               *
C                                            *
C     This subroutine evaluates NMOSFET gate   *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C******************************************

      FUNCTION NGCAP(W)
```

```
      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      NGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C*******************************************
C              PGCAP SUBROUTINE            *
C                                          *
C       This subroutine evaluates PMOSFET gate   *
C capacitance using standard layout geometry and   *
C the given fabrication technology parameters.   *
C*******************************************

      FUNCTION PGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      PGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C*******************************************
C              PGSOCAP SUBROUTINE          *
C                                          *
C       This subroutine evaluates PMOSFET gate-   *
C source overlap capacitance using standard   *
C layout geometry and the give fabrication   *
C technology parameters.   *
C*******************************************

      FUNCTION PGSOCAP(W)

      IMPLICIT REAL (A-Z)


            CGSO=2.5E-10

      PGSOCAP=CGSO*W

      RETURN
      END
```

```
C*****************************************
C            PGDOCAP SUBROUTINE           *
C                                         *
C     This subroutine evaluates PMOSFET gate-  *
C drain overlap capacitance using standard    *
C layout geometry and the give fabrication    *
C technology parameters.                      *
C*****************************************

      FUNCTION PGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=2.5E-10

      PGDOCAP=CGDO*W

      RETURN
      END

C*****************************************
C            NGSOCAP SUBROUTINE           *
C                                         *
C     This subroutine evaluates PMOSFET gate-  *
C source overlap capacitance using standard   *
C layout geometry and the give fabrication    *
C technology parameters.                      *
C*****************************************

      FUNCTION NGSOCAP(W)

      IMPLICIT REAL (A-Z)

            CGSO=3.0E-10

      NGSOCAP=CGSO*W

      RETURN
      END

C*****************************************
C            NGDOCAP SUBROUTINE           *
C                                         *
C     This subroutine evaluates NMOSFET gate-  *
C drain overlap capacitance using standard    *
C layout geometry and the give fabrication    *
C technology parameters.                      *
C*****************************************

      FUNCTION NGDOCAP(W)

      IMPLICIT REAL (A-Z)
```

```
          CGDO=3.0E-10

      NGDOCAP=CGDO*W

      RETURN
      END

C*****************************************
C          GEOM SUBROUTINE               *
C                                        *
C     This subroutine calculates NFET/PFET *
C drain/source area and perimeter based on a *
C standard layout technique and the given *
C fabrication technology rules.          *
C*****************************************

      SUBROUTINE GEOM(W,AS,PS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      PS=W+14.4E-6
      AS=W*L+22.68E-12

      RETURN
      END

C*****************************************
C     NFET_MIN_RESISTANCE SUBROUTINE*
C                                        *
C     This subroutine approximates the per *
C unit channel resistance for those NFETs that *
C mainly discharge in the linear region. *
C*****************************************

      FUNCTION NFET_MIN_RESISTANCE(W,VG,VS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      VGS=VG-VS
      VBS=VBN-VS

          EPSSI=11.7*8.85E-12
          Q=1.6022E-19
          UO=775E-4
          VTO=0.7
          VMAX=1.0E5
          PHI=0.6
          COX=6.9E-4
          NSUB=1.7E22
          XJL=3.5E-7
```

```
          GAMMA=1.1
          XJ=6.0E-7
          ETA=1.0

C...  CALCULATE INTERMEDIATE PARAMETERS

      US=UO
      PHIF=PHI/2.0
      VFB=VTO-PHI-2.0*SQRT(Q*EPSSI*NSUB*PHIF)/COX
      LEFF=L-2.0*XJL
      BETA=W/LEFF*US*COX
      VBI=VFB+PHI
      VBIN=VBI
      XD=SQRT(2.0*EPSSI/Q/NSUB)
      WD=XD*SQRT(PHI-VBS+VDS)
      WS=XD*SQRT(PHI-VBS)
      ALFAD=0.5*XJ/LEFF*(SQRT(1.0+2*WD/XJ)-1.0)
      ALFAS=0.5*XJ/LEFF*(SQRT(1.0+2*WS/XJ)-1.0)
      GAMMAS=GAMMA*(1.0-ALFAS-ALFAD)

      NFET_MIN_RESISTANCE=1.316/BETA/(VGS-VBIN-GAMMAS*(2*PHI-
VBS)**0.5)

      RETURN
      END

C*********END OF SINGLE VARIABLE OPTIMIZATION PROGRAM************
C*********END OF SINGLE VARIABLE OPTIMIZATION PROGRAM**************
```

B.4 ANALYTICAL APPROACH Source Code

```
C*****************************************
C               ANALYTICAL APPROACH          *
C                                            *
C       This code sizes an NFET chain with the   *
C analytical formula. No iterative techniques    *
C involved.                                      *
C*****************************************

      IMPLICIT REAL (A-Z)
      INTEGER N,I,LIMIT,ITER
      REAL WI,W(10)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
     + ,K1,K2,K3,K4,K5

C...  OPEN OUTPUT FILE

      OPEN(UNIT=2,NAME='analytical.o',STATUS='NEW')
      WRITE(2,*)'******** ANALYTICAL APPROACH RESULTS ********'
      WRITE(2,*)'******** ANALYTICAL APPROACH RESULTS ********'
      WRITE(2,*)

C...  INTIALIZE VARIABLES AND SET UP PROBLEM

      LIMIT=100
      L=3.0E-6
      WMIN=5.4E-6
      DESIRED_DELAY=1.692e-9
      STEP=0.6E-6
      WP=WMAX
      WLP=WMIN
      WLN=WMIN
      VBN=0.0
      VBP=5.0
      N=7

C...  CALCULATE THE PER UNIT RESISTANCES

      PURL=NFET_MIN_RESISTANCE(1.0,5.0,0.0)
      PURS=PURL*1.7
      TD=DESIRED_DELAY/1.175

      CALL ANALYTICAL(N,W,TD)

C...  DOCUMENT PROBLEM SET UP AND PARAMETERS

      WRITE(2,*)'  PURL         =',PURL
```

```
      WRITE(2,*)' PURS          =',PURS
      WRITE(2,*)' DESIRED_DELAY    =',DESIRED_DELAY
      WRITE(2,*)' WMIN             =',WMIN
      WRITE(2,*)' N          =',N
      WRITE(2,*)' L          =',L
      WRITE(2,*)' STEP       =',STEP
      WRITE(2,*)' WP         =',WP
      WRITE(2,*)' WLP        =',WLP
      WRITE(2,*)' WLN        =',WLN
      WRITE(2,*)' K1         =',K1
      WRITE(2,*)' K2         =',K2
      WRITE(2,*)' K3         =',K3
      WRITE(2,*)' K4         =',K4
      WRITE(2,*)' K5         =',K5

C...  DOCUMENT SOLUTION

      WRITE(2,*)
      WRITE(2,*)
      WRITE(2,*)' RESULTED DISCHARGE DELAY=',TD
      WRITE(2,*)

C...  PRINT OPTMIZED VECTOR AND TOTAL GATE AREA

      WTOT=0.0
      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR LEVEL=',I,' TRANSISTOR SIZE=',W(I)
      WTOT=WTOT+W(I)
      ENDDO
      WRITE(2,*)
      WRITE(2,*)' TOTAL AREA OF THE CHAIN      =',WTOT*L
      WRITE(2,*)

      STOP
      END


C*******************************************
C     ANALYTICAL SUBROUTINE                *
C     This routine computes the NFET chain *
C widths using the analytical formula.     *
C*******************************************

      SUBROUTINE ANALYTICAL(N,W,TD)

      IMPLICIT REAL (A-Z)
      INTEGER N,I
      REAL W(N),R(10),C(10)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
   +  ,K1,K2,K3,K4,K5

C...  INTIALIZE VARIABLES

      DO I=1,N
```

```
      R(I) =0.0
      C(I) =0.0
      ENDDO

C...  CALCULATE CAPACITANCE CONSTANTS K1, K2, K3, K4, AND K5
C     ONLY ONCE

      WN=0.0
      K1=NSCAP(WN,0.0,VBN)+.5*NGCAP(WN)+NGDOCAP(WN)
  +   +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
  +   +NGCAP(WLN)+PGCAP(WLP)
      WN=WMIN
      CN_WN=NSCAP(WN,0.0,VBN)+.5*NGCAP(WN)+NGDOCAP(WN)
  +   +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
  +   +NGCAP(WLN)+PGCAP(WLP)
      K2=(CN_WN-K1)/WN

      W1=0.0
      W2=0.0
      K3=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
  +   +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
      W1=WMIN
      W2=0.0
      C_W1=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
  +   +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
      K4=(C_W1-K3)/W1
      W1=0.0
      W2=WMIN
      C_W2=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
  +   +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
      K5=(C_W2-K3)/W2


C...  SIZE THE NFET CHAIN ANALYTICALY

      TDTERM=TD/N
      W(N)=K1/((TDTERM/PURS)-K2)
      IF(W(N).LT.WMIN) W(N)=WMIN
      C(N)=K1+K2*W(N)
      CABOVE=C(N)
      DO I=N-1,1,-1
      W(I)=PURL*(K3+K5*W(I+1)+CABOVE)/(TDTERM-PURL*K4)
      IF(W(I).LT.WMIN) W(I)=WMIN
      C(I)=K3+K4*W(I)+K5*W(I+1)
      CABOVE=CABOVE+C(I)
      ENDDO

C...  CONVERT THE COMPUTED SIZE VECTOR TO STEPS OF MICRONS

      W(N)=STEP*AINT(W(N)/STEP+0.5)
      C(N)=K1+K2*W(N)
      R(N)=PURS/W(N)
      DO I=N-1,1,-1
      W(I)=STEP*AINT(W(I)/STEP+0.5)
```

```
      C(I)=K3+K4*W(I)+K5*W(I+1)
      R(I)=PURL/W(I)
      ENDDO

C...  NOW, CALCULATE THE DELAY USING ELMORE'S DELAY FORMULA

      TD=0.0
      DO I=1,N
      CABOVE=0.0
      DO J=I,N
      CABOVE=CABOVE+C(J)
      ENDDO
      TD=TD+R(I)*CABOVE
      ENDDO

C...  RETURN THE SIZES AND THE DELAY

      TD=TD*1.175

      RETURN
      END

C******************************************
C             SUBROUTINE NSCAP            *
C                                         *
C     This subroutine evaluates NMOSFET source*
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C******************************************

      FUNCTION NSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)

      VBS=VB-VS

            CGSO=3.0E-10
            CJSW=4.0E-10
            CJ=4.4E-4
            MJ=0.5
            MJSW=0.3
            PB=0.7
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VBS.LE.FC*PB) THEN
      NSCAP=CJ*AS/(1.0-VBS/PB)**MJ+CJSW*PS/(1.0-VBS/PB)**MJSW
      ELSE
      NSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VBS/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VBS/PB*MJSW)
      ENDIF

      RETURN
```

```
      END

C*******************************************
C              SUBROUTINE PSCAP            *
C                                          *
C      This subroutine evaluates PMOSFET source *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C*******************************************

      FUNCTION PSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)
      VSB=VS-VB

            CGSO=2.5E-10
            CJSW=4.0E-10
            CJ=1.5E-4
            MJ=0.6
            MJSW=0.6
            PB=0.6
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VSB.LE.FC*PB) THEN
      PSCAP=CJ*AS/(1.0-VSB/PB)**MJ+CJSW*PS/(1.0-VSB/PB)**MJSW
      ELSE
      PSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VSB/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VSB/PB*MJSW)
      ENDIF

      RETURN
      END


C*******************************************
C              NGCAP SUBROUTINE            *
C                                          *
C      This subroutine evaluates NMOSFET gate *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C*******************************************

      FUNCTION NGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      NGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE
```

```
      RETURN
      END

C****************************************
C              PGCAP SUBROUTINE          *
C                                        *
C      This subroutine evaluates PMOSFET gate   *
C capacitance using standard layout geometry and   *
C the given fabrication technology  parameters.    *
C****************************************

      FUNCTION PGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON  PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      PGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C****************************************
C              PGSOCAP SUBROUTINE        *
C                                        *
C      This subroutine evaluates PMOSFET gate-   *
C source overlap capacitance using standard       *
C layout geometry and the give fabrication        *
C technology parameters.                          *
C****************************************

      FUNCTION PGSOCAP(W)

      IMPLICIT REAL (A-Z)


            CGSO=2.5E-10

      PGSOCAP=CGSO*W

      RETURN
      END

C****************************************
C              PGDOCAP SUBROUTINE        *
C                                        *
C      This subroutine evaluates PMOSFET gate-   *
C drain overlap capacitance using standard        *
C layout geometry and the give fabrication        *
C technology parameters.                          *
```

```
C*****************************************

        FUNCTION PGDOCAP(W)

        IMPLICIT REAL (A-Z)

                CGDO=2.5E-10

        PGDOCAP=CGDO*W

        RETURN
        END

C*****************************************
C               NGSOCAP SUBROUTINE              *
C                                               *
C       This subroutine evaluates PMOSFET gate- *
C source overlap capacitance using standard     *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C*****************************************

        FUNCTION NGSOCAP(W)

        IMPLICIT REAL (A-Z)

                CGSO=3.0E-10

        NGSOCAP=CGSO*W

        RETURN
        END

C*****************************************
C               NGDOCAP SUBROUTINE              *
C                                               *
C       This subroutine evaluates NMOSFET gate- *
C drain overlap capacitance using standard      *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C*****************************************

        FUNCTION NGDOCAP(W)

        IMPLICIT REAL (A-Z)

                CGDO=3.0E-10

        NGDOCAP=CGDO*W

        RETURN
        END

C*****************************************
```

```
C              GEOM SUBROUTINE                *
C                                             *
C       This subroutine calculates NFET/PFET  *
C drain/source area and perimeter based on a  *
C standard layout technique and the given     *
C fabrication technology rules.               *
C*********************************************

      SUBROUTINE GEOM(W,AS,PS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      PS=W+14.4E-6
      AS=W*L+22.68E-12

      RETURN
      END

C*********************************************
C       NFET_MIN_RESISTANCE SUBROUTINE*
C                                             *
C       This subroutine approximates the per  *
C unit channel resistance for those NFETs that *
C mainly discharge in the linear region.      *
C*********************************************

      FUNCTION NFET_MIN_RESISTANCE(W,VG,VS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      VGS=VG-VS
      VBS=VBN-VS

            EPSSI=11.7*8.85E-12
            Q=1.6022E-19
            UO=775E-4
            VTO=0.7
            VMAX=1.0E5
            PHI=0.6
            COX=6.9E-4
            NSUB=1.7E22
            XJL=3.5E-7
            GAMMA=1.1
            XJ=6.0E-7
            ETA=1.0

C...  CALCULATE INTERMEDIATE PARAMETERS

      US=UO
```

```
      PHIF=PHI/2.0
      VFB=VTO-PHI-2.0*SQRT(Q*EPSSI*NSUB*PHIF)/COX
      LEFF=L-2.0*XJL
      BETA=W/LEFF*US*COX
      VBI=VFB+PHI
      VBIN=VBI
      XD=SQRT(2.0*EPSSI/Q/NSUB)
      WD=XD*SQRT(PHI-VBS+VDS)
      WS=XD*SQRT(PHI-VBS)
      ALFAD=0.5*XJ/LEFF*(SQRT(1.0+2*WD/XJ)-1.0)
      ALFAS=0.5*XJ/LEFF*(SQRT(1.0+2*WS/XJ)-1.0)
      GAMMAS=GAMMA*(1.0-ALFAS-ALFAD)

      NFET_MIN_RESISTANCE=1.316/BETA/(VGS-VBIN-GAMMAS*(2*PHI-
VBS)**0.5)

      RETURN
      END

C***************END OF ANALYTICAL APPROACH PROGRAM***************
C***************END OF ANALYTICAL APPROACH PROGRAM***************
```

Appendix C

Precharge PFET Sizing Source Codes

C.1 Precharge Delay Approximation Source Code

```
C*************************************************
C              PRECHARGE DELAY                    *
C                                                 *
C       This code approximates an NFET chain      *
C precharge delay.                                *
C*************************************************

        IMPLICIT REAL (A-Z)
        INTEGER N,I
        REAL W(10)
        COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
     +  ,K1,K2,K3,K4,K5,PURP
        COMMON /fun/S1,S2,A1,A2,A3,A

C...    OPEN OUTPUT FILE

        OPEN(UNIT=2,NAME='precharge_delay.o',STATUS='NEW')
        WRITE(2,*)'********  ********'
        WRITE(2,*)'********  ********'
        WRITE(2,*)

C...    INTIALIZE VARIABLES AND SET UP PROBLEM

        L=3.0E-6
        WMIN=5.4E-6
        STEP=0.6E-6
        WLP=WMIN
        WLN=WMIN
        VBN=0.0
        VBP=5.0
        N=7

C...    SET PRECHARGE PFET WIDTH

        WP=WMIN

C...    SET NFET CHAIN WIDTHS

        DO I=1,N
        W(I)=WMIN
        ENDDO

C...    CALCULATE THE PER UNIT RESISTANCES

        PURL=NFET_MIN_RESISTANCE(1.0,5.0,0.0)
        PURS=PURL*1.7
```

```
      PURP=8.82e-2

      CALL precharge_delay(N,W,TP)

C...  DOCUMENT PROBLEM SET UP AND PARAMETERS

      WRITE(2,*)' PURL        =',PURL
      WRITE(2,*)' PURS        =',PURS
      WRITE(2,*)' PURP        =',PURP
      WRITE(2,*)' WMIN              =',WMIN
      WRITE(2,*)' N           =',N
      WRITE(2,*)' L           =',L
      WRITE(2,*)' STEP        =',STEP
      WRITE(2,*)' WP          =',WP
      WRITE(2,*)' WLP         =',WLP
      WRITE(2,*)' WLN         =',WLN
      WRITE(2,*)' K1          =',K1
      WRITE(2,*)' K2          =',K2
      WRITE(2,*)' K3          =',K3
      WRITE(2,*)' K4          =',K4
      WRITE(2,*)' K5          =',K5

C...  DOCUMENT SOLUTION

      WRITE(2,*)
      WRITE(2,*)
      WRITE(2,*)' RESULTED PRECHARGE DELAY=',TP
      WRITE(2,*)

C...  PRINT NFET CHAIN WIDTHS

      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR LEVEL=',I,' TRANSISTOR SIZE=',W(I)
      ENDDO

      STOP
      END


C***********************************************
C            PRECHARGE_DELAY SUBROUTINE        *
C     This subroutine approximates the precha- *
C rge delay using the precharge RC model and the *
C bisection method.                            *
C***********************************************

      SUBROUTINE precharge_delay(N,W,TP)

      IMPLICIT REAL (A-Z)
      INTEGER N,I
      REAL W(N),R(10),C(10)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L
   +  ,K1,K2,K3,K4,K5,PURP
      COMMON /fun/S1,S2,A1,A2,A3,A
```

```
C...   INTIALIZE VARIABLES

       DO I=1,N
       R(I) =0.0
       C(I) =0.0
       ENDDO

C...   CALCULATE CAPACITANCE CONSTANTS K1, K2, K3, K4, AND K5
C      ONLY ONCE

       WN=0.0
       K1=NSCAP(WN,0.0,VBN)+.5*NGCAP(WN)+NGDOCAP(WN)
    +  +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
    +  +NGCAP(WLN)+PGCAP(WLP)
       WN=WMIN
       CN_WN=NSCAP(WN,0.0,VBN)+.5*NGCAP(WN)+NGDOCAP(WN)
    +  +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
    +  +NGCAP(WLN)+PGCAP(WLP)
       K2=(CN_WN-K1)/WN

       W1=0.0
       W2=0.0
       K3=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
    +  +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
       W1=WMIN
       W2=0.0
       C_W1=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
    +  +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
       K4=(C_W1-K3)/W1
       W1=0.0
       W2=WMIN
       C_W2=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
    +  +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
       K5=(C_W2-K3)/W2


C...   CALCULATE RP,RE,CE, and CB

C      W(N)=STEP*AINT(W(N)/STEP+0.5)
       RP=PURP/WP
       RE=PURS/W(N)
       CE=K1+K2*W(N)
       CB=0.0
       DO I=N-1,2,-1
       CB=CB+K3+K4*W(I)+K5*W(I+1)
       RE=RE+PURL/W(I)
       ENDDO
       CB=CB+K3+K4*W(1)+K5*W(1+1)

C...   NOW, CALCULATE THE PRECHARGE DELAY USING 2 TIME CONSTANT
MODEL
c...   CALCULATE INTERMEDIATE VARIABLES
```

```
      A=1.0/(CE*CB*RP*RE)
      A4=CB*RE
      A5=(CB*RE+CE*RP+CB*RP)/(CE*CB*RP*RE)
      A6=1.0/(CE*CB*RP*RE)

c...  check

      arg=A5*A5-4.0*A6
      if(arg.lt.0.0) then
      write(2,*)'***error: S1 and S2 are imaginary***'
      write(2,*)'***error: S1 and S2 are imaginary***'
      stop
      endif

      S1=(-A5+sqrt(arg))/2.0
      S2=(-A5-sqrt(arg))/2.0
      A1=(1.0+A4*S1)/S1/(S1-S2)
      A2=(1.0+A4*S2)/S2/(S2-S1)
      A3=1.0/(-S1)/(-S2)

      call bisection(1.0e-15,10.0,TP)

      TP=TP*0.5296

      RETURN
      END

      subroutine bisection(t1,t2,t)

      IMPLICIT REAL (A-Z)

      f1=eval(t1)
      f2=eval(t2)
      if(f1.lt.0.0.and.f2.lt.0.0) goto 15
      if(f1.gt.0.0.and.f2.gt.0.0) goto 15
      if(f1.lt.0.0) then
      tn=t1
      tp=t2
      else
      tn=t2
      tp=t1
      endif

 10   t=(tn+tp)/2.0
      f=eval(t)
      if(f.lt.0.0) tn=t
      if(f.gt.0.0) tp=t
      if(abs((tn-tp)/tp).lt.1.0e-2) RETURN
      goto 10

 15   write(2,*)'ERROR: bad interival'
      STOP
      END
```

```
      function eval(t)

      IMPLICIT REAL (A-Z)
      COMMON /fun/S1,S2,A1,A2,A3,A

      eval=A1*exp(S1*t)+A2*exp(S2*t)+A3-(0.64/A)

      RETURN
      END


C***********************************************
C              SUBROUTINE NSCAP                *
C                                              *
C      This subroutine evaluates NMOSFET source *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.  *
C***********************************************

      FUNCTION NSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)

      VBS=VB-VS

            CGSO=3.0E-10
            CJSW=4.0E-10
            CJ=4.4E-4
            MJ=0.5
            MJSW=0.3
            PB=0.7
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VBS.LE.FC*PB) THEN
      NSCAP=CJ*AS/(1.0-VBS/PB)**MJ+CJSW*PS/(1.0-VBS/PB)**MJSW
      ELSE
      NSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VBS/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VBS/PB*MJSW)
      ENDIF

      RETURN
      END
C***********************************************
C              SUBROUTINE PSCAP                *
C                                              *
C      This subroutine evaluates PMOSFET source *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.  *
C***********************************************
```

```
      FUNCTION PSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)
      VSB=VS-VB

            CGSO=2.5E-10
            CJSW=4.0E-10
            CJ=1.5E-4
            MJ=0.6
            MJSW=0.6
            PB=0.6
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VSB.LE.FC*PB) THEN
      PSCAP=CJ*AS/(1.0-VSB/PB)**MJ+CJSW*PS/(1.0-VSB/PB)**MJSW
      ELSE
      PSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VSB/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VSB/PB*MJSW)
      ENDIF

      RETURN
      END

C**********************************************
C             NGCAP SUBROUTINE               *
C                                            *
C     This subroutine evaluates NMOSFET gate *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
C**********************************************

      FUNCTION NGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


            COX=6.9E-4
            COXE=0.5E-10

      NGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C**********************************************
C             PGCAP SUBROUTINE               *
C                                            *
C     This subroutine evaluates PMOSFET gate *
C capacitance using standard layout geometry and *
C the given fabrication technology parameters.   *
```

```
C***********************************************

      FUNCTION PGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


              COX=6.9E-4
              COXE=0.5E-10

      PGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END
C***********************************************
C              PGSOCAP SUBROUTINE              *
C                                              *
C       This subroutine evaluates PMOSFET gate- *
C source overlap capacitance using standard    *
C layout geometry and the give fabrication     *
C technology parameters.                       *
C***********************************************

      FUNCTION PGSOCAP(W)

      IMPLICIT REAL (A-Z)


              CGSO=2.5E-10

      PGSOCAP=CGSO*W

      RETURN
      END
C***********************************************
C              PGDOCAP SUBROUTINE              *
C                                              *
C       This subroutine evaluates PMOSFET gate- *
C drain overlap capacitance using standard     *
C layout geometry and the give fabrication     *
C technology parameters.                       *
C***********************************************

      FUNCTION PGDOCAP(W)

      IMPLICIT REAL (A-Z)

              CGDO=2.5E-10

      PGDOCAP=CGDO*W
```

```
      RETURN
      END

C***********************************************
C               NGSOCAP SUBROUTINE             *
C                                               *
C       This subroutine evaluates PMOSFET gate- *
C source overlap capacitance using standard     *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C***********************************************

      FUNCTION NGSOCAP(W)

      IMPLICIT REAL (A-Z)

            CGSO=3.0E-10

      NGSOCAP=CGSO*W

      RETURN
      END

C***********************************************
C               NGDOCAP SUBROUTINE             *
C                                               *
C       This subroutine evaluates NMOSFET gate- *
C drain overlap capacitance using standard      *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C***********************************************

      FUNCTION NGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=3.0E-10

      NGDOCAP=CGDO*W

      RETURN
      END

C***********************************************
C               GEOM SUBROUTINE                *
C                                               *
C       This subroutine calculates NFET/PFET    *
C drain/source area and perimeter based on a    *
C standard layout technique and the given       *
C fabrication technology rules.                 *
C***********************************************

      SUBROUTINE GEOM(W,AS,PS)
```

```
      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L

      PS=W+14.4E-6
      AS=W*L+22.68E-12

      RETURN
      END

C**************************************************
C     NFET_MIN_RESISTANCE SUBROUTINE             *
C                                                 *
C     This subroutine approximates the per       *
C unit channel resistance for those NFETs that   *
C mainly discharge in the linear region.         *
C**************************************************

      FUNCTION NFET_MIN_RESISTANCE(W,VG,VS)

      IMPLICIT REAL (A-Z)
      COMMON PURL,PURS,WMAX,WMIN,STEP,WP,WLP,WLN,VBN,VBP,L


      VGS=VG-VS
      VBS=VBN-VS

            EPSSI=11.7*8.85E-12
            Q=1.6022E-19
            UO=775E-4
            VTO=0.7
            VMAX=1.0E5
            PHI=0.6
            COX=6.9E-4
            NSUB=1.7E22
            XJL=3.5E-7
            GAMMA=1.1
            XJ=6.0E-7
            ETA=1.0


C...  CALCULATE INTERMEDIATE PARAMETERS

      US=UO
      PHIF=PHI/2.0
      VFB=VTO-PHI-2.0*SQRT(Q*EPSSI*NSUB*PHIF)/COX
      LEFF=L-2.0*XJL
      BETA=W/LEFF*US*COX
      VBI=VFB+PHI
      VBIN=VBI
      XD=SQRT(2.0*EPSSI/Q/NSUB)
      WD=XD*SQRT(PHI-VBS+VDS)
      WS=XD*SQRT(PHI-VBS)
      ALFAD=0.5*XJ/LEFF*(SQRT(1.0+2*WD/XJ)-1.0)
      ALFAS=0.5*XJ/LEFF*(SQRT(1.0+2*WS/XJ)-1.0)
```

```
      GAMMAS=GAMMA*(1.0-ALFAS-ALFAD)

      NFET_MIN_RESISTANCE=1.316/BETA/(VGS-VBIN-GAMMAS*(2*PHI-
VBS)**0.5)

      RETURN
      END

C******END OF PRECHARGE DELAY APPROXIMATION  PROGRAM******
```
C******END OF PRECHARGE DELAY APPROXIMATION PROGRAM******

C.2 Precharge PFET Sizing Source Code

```
C*************************************************
C              PRECHARGE SIZING                  *
C                                                *
C       This code sizes precharge PFET according *
C to two time constant delay model for the       *
C dynamic logic structure.                       *
C*************************************************

      IMPLICIT REAL (A-Z)
      INTEGER N,I
      REAL W(10)
      COMMON  /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
      COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP
      COMMON /TECHNOLOGY/L,STEP
      COMMON /FUN/RB,CB

C...  OPEN OUTPUT FILE

      OPEN(UNIT=2,NAME='precharge_sizing.o',STATUS='NEW')
      WRITE(2,*)'******** PRECHARGE PFET SIZING ********'
      WRITE(2,*)'******** PRECHARGE PFET SIZING ********'
      WRITE(2,*)

C...  INTIALIZE VARIABLES AND SET UP PROBLEM

      L=3.0E-6
      WMIN=5.4E-6
      DESIRED_DELAY=3E-9
      STEP=0.6E-6
      WLP=WMIN
      WLN=WMIN
      VBN=0.0
      VBP=5.0
      N=7

C...  SET NFET CHAIN WIDTHS

      W(7)=5.4E-6
      W(6)=5.4E-6
      W(5)=5.4E-6
      W(4)=5.4E-6
      W(3)=5.4E-6
      W(2)=5.4E-6
      W(1)=5.4E-6

C...  CALCULATE THE PER UNIT RESISTANCES
```

```
      PURL=NFET_MIN_RESISTANCE(1.0,5.0,0.0)
      PURS=PURL*1.7
      PURP=8.82e-2
      TP=DESIRED_DELAY/0.5296

      CALL precharge_sizing(N,W,TP,WP)

C...  CONVERT PRECHARGE WIDTH TO STEPS OF MICRONS

      WP=STEP*AINT(WP/STEP+0.5)

C...  DOCUMENT PROBLEM SET UP AND PARAMETERS

      WRITE(2,*)' PURL          =',PURL
      WRITE(2,*)' PURS          =',PURS
      WRITE(2,*)' PURP          =',PURP
      WRITE(2,*)' DESIRED_DELAY    =',DESIRED_DELAY
      WRITE(2,*)' WMIN                 =',WMIN
      WRITE(2,*)' N             =',N
      WRITE(2,*)' L             =',L
      WRITE(2,*)' STEP          =',STEP
      WRITE(2,*)' WLP           =',WLP
      WRITE(2,*)' WLN           =',WLN
      WRITE(2,*)' K1            =',K1
      WRITE(2,*)' K2            =',K2
      WRITE(2,*)' K3            =',K3
      WRITE(2,*)' K4            =',K4
      WRITE(2,*)' K5            =',K5
      WRITE(2,*)' K6            =',K6
      WRITE(2,*)' K7            =',K7

C...  DOCUMENT SOLUTION

      WRITE(2,*)
      WRITE(2,*)
      WRITE(2,*)'  RESULTED PRECHARGE PFET WIDTH=',WP
      WRITE(2,*)

C...  PRINT NFET CHAIN WIDTHS

      WRITE(2,*)'NFET CHAIN WIDTHS'
      DO I=N,1,-1
      WRITE(2,*)'TRANSISTOR LEVEL=',I,'  TRANSISTOR SIZE=',W(I)
      ENDDO

      STOP
      END


C*************************************************
C              PRECHARGE SIZING ROUTINE          *
C      This routine sizes the precharge PFET by  *
C using two time constant model and root finder  *
```

```
C algorithm such as bisection method.                   *
C***********************************************

        SUBROUTINE precharge_sizing(N,W,TP,WP)

        IMPLICIT REAL (A-Z)
        INTEGER N,I
        REAL W(N),R(10),C(10)
        COMMON /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
        COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP
        COMMON /TECHNOLOGY/L,STEP
        COMMON /FUN/RB,CB

C...    INTIALIZE VARIABLES

        DO I=1,N
        R(I) =0.0
        C(I) =0.0
        ENDDO

C...    CALCULATE CAPACITANCE CONSTANTS K6, K7, K3, K4, AND K5
C       ONLY ONCE

        WP=0.0
        K6=NSCAP(W(N),0.0,VBN)+.5*NGCAP(W(N))+NGDOCAP(W(N))
   +    +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
   +    +NGCAP(WLN)+PGCAP(WLP)
        WP=WMIN
        CN_WP=NSCAP(W(N),0.0,VBN)+.5*NGCAP(W(N))+NGDOCAP(W(N))
   +    +PSCAP(WP,0.0,VBP)+.5*PGCAP(WP)+PGDOCAP(WP)
   +    +NGCAP(WLN)+PGCAP(WLP)
        K7=(CN_WP-K6)/WP

        W1=0.0
        W2=0.0
        K3=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
   +    +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
        W1=WMIN
        W2=0.0
        C_W1=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
   +    +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
        K4=(C_W1-K3)/W1
        W1=0.0
        W2=WMIN
        C_W2=NSCAP(W2,0.0,NVB)+.5*NGCAP(W2)+NGSOCAP(W2)
   +    +NSCAP(W1,0.0,NVB)+.5*NGCAP(W1)+NGDOCAP(W1)
        K5=(C_W2-K3)/W2

C...    CALCULATE RB and CB

        RB=PURS/W(N)
        CB=0.0
        DO I=N-1,2,-1
        CB=CB+K3+K4*W(I)+K5*W(I+1)
```

```
      RB=RB+PURL/W(I)
      ENDDO
      CB=CB+K3+K4*W(I)+K5*W(I+1)

      call bisection_size(WMIN,500E-6,TP,WP)

      RETURN
      END

      subroutine bisection_size(wp1,wp2,TP,WP)

      IMPLICIT REAL (A-Z)
      COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP

C...  CHECK IF THE MINIMUM PRECHARGE PFET WIDTH IS SUFFICIENT

      IF(eval(TP,WMIN).GT.0.0) then
      WP=WMIN
      RETURN
      ENDIF

      f1=eval(TP,wp1)
      f2=eval(TP,wp2)
      if(f1.lt.0.0.and.f2.lt.0.0) goto 15
      if(f1.gt.0.0.and.f2.gt.0.0) goto 15
      if(f1.lt.0.0) then
      wpn=wp1
      wpp=wp2
      else
      wpn=wp2
      wpp=wp1
      endif

 10   wp=(wpn+wpp)/2.0
      f=eval(TP,wp)
      if(f.lt.0.0) wpn=wp
      if(f.gt.0.0) wpp=wp
      if(abs((wpn-wpp)/wpp).lt.1.0e-2) RETURN
      goto 10

 15   write(2,*)'ERROR: bad interival'
      STOP
      END

      function eval(TP,WP)

      IMPLICIT REAL (A-Z)
      COMMON /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
      COMMON /FUN/RB,CB

C...  NOW, CALCULATE THE PRECHARGE DELAY USING 2 TIME CONSTANT
MODEL

      CE=K6+K7*WP
```

```
      RP=PURP/WP

c...  CALCULATE INTERMEDIATE VARIABLES

      A=1.0/(CE*CB*RP*RB)
      A4=CB*RB
      A5=(CB*RB+CE*RP+CB*RP)/(CE*CB*RP*RB)

c...  check

      arg=A5*A5-4.0*A
      if(arg.lt.0.0) then
      write(2,*)'***error: S1 and S2 are imaginary***'
      write(2,*)'***error: S1 and S2 are imaginary***'
      stop
      endif

      S1=(-A5+sqrt(arg))/2.0
      S2=(-A5-sqrt(arg))/2.0
      A1=(1.0+A4*S1)/S1/(S1-S2)
      A2=(1.0+A4*S2)/S2/(S2-S1)
      A3=1.0/(-S1)/(-S2)


      eval=A1*exp(S1*TP)+A2*exp(S2*TP)+A3-(0.64/A)

      RETURN
      END


C*************************************************
C              SUBROUTINE NSCAP                  *
C                                                *
C       This subroutine evaluates NMOSFET source *
C capacitance using standard layout geometry and *
C the given fabrication technology  parameters.  *
C*************************************************

      FUNCTION NSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)

      VBS=VB-VS

            CGSO=3.0E-10
            CJSW=4.0E-10
            CJ=4.4E-4
            MJ=0.5
            MJSW=0.3
            PB=0.7
            FC=0.5

      CALL GEOM(W,AS,PS)
```

```
      IF(VBS.LE.FC*PB) THEN
      NSCAP=CJ*AS/(1.0-VBS/PB)**MJ+CJSW*PS/(1.0-VBS/PB)**MJSW
      ELSE
      NSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VBS/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VBS/PB*MJSW)
      ENDIF

      RETURN
      END

C***********************************************
C            SUBROUTINE PSCAP                  *
C                                              *
C     This subroutine evaluates PMOSFET source *
C capacitance using standard layout geometry and *
C the given fabrication technology  parameters. *
C***********************************************

      FUNCTION PSCAP(W,VS,VB)

      IMPLICIT REAL (A-Z)
      VSB=VS-VB

            CGSO=2.5E-10
            CJSW=4.0E-10
            CJ=1.5E-4
            MJ=0.6
            MJSW=0.6
            PB=0.6
            FC=0.5

      CALL GEOM(W,AS,PS)

      IF(VSB.LE.FC*PB) THEN
      PSCAP=CJ*AS/(1.0-VSB/PB)**MJ+CJSW*PS/(1.0-VSB/PB)**MJSW
      ELSE
      PSCAP=CJ*AS/(1.0-FC)**(1.0+MJ)*(1.0-FC*(1.0+MJ)+VSB/PB*MJ)
   +  +CJSW*PS/(1.0-FC)**(1.0+MJSW)*(1.0-FC*(1.0+MJSW)+VSB/PB*MJSW)
      ENDIF

      RETURN
      END


C***********************************************
C            NGCAP SUBROUTINE                  *
C                                              *
C     This subroutine evaluates NMOSFET gate   *
C capacitance using standard layout geometry and *
C the given fabrication technology  parameters. *
C***********************************************

      FUNCTION NGCAP(W)
```

```
      IMPLICIT REAL (A-Z)
      COMMON  /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
      COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP
      COMMON /TECHNOLOGY/L,STEP
      COMMON /FUN/RB,CB


            COX=6.9E-4
            COXE=0.5E-10

      NGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C************************************************
C             PGCAP SUBROUTINE                  *
C                                               *
C       This subroutine evaluates PMOSFET gate  *
C capacitance using standard layout geometry and *
C the given fabrication technology  parameters.  *
C************************************************

      FUNCTION PGCAP(W)

      IMPLICIT REAL (A-Z)
      COMMON  /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
      COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP
      COMMON /TECHNOLOGY/L,STEP
      COMMON /FUN/RB,CB

            COX=6.9E-4
            COXE=0.5E-10

      PGCAP=L*W*COX+(W+3.0E-6)*2.0*COXE

      RETURN
      END

C************************************************
C             PGSOCAP SUBROUTINE                *
C                                               *
C       This subroutine evaluates PMOSFET gate- *
C source overlap capacitance using standard     *
C layout geometry and the give fabrication      *
C technology parameters.                        *
C************************************************

      FUNCTION PGSOCAP(W)

      IMPLICIT REAL (A-Z)


            CGSO=2.5E-10
```

```
      PGSOCAP=CGSO*W

      RETURN
      END

C***********************************************
C            PGDOCAP SUBROUTINE               *
C                                             *
C       This subroutine evaluates PMOSFET gate-  *
C drain overlap capacitance using standard    *
C layout geometry and the give fabrication    *
C technology parameters.                      *
C***********************************************

      FUNCTION PGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=2.5E-10

      PGDOCAP=CGDO*W

      RETURN
      END

C***********************************************
C            NGSOCAP SUBROUTINE               *
C                                             *
C       This subroutine evaluates PMOSFET gate-  *
C source overlap capacitance using standard   *
C layout geometry and the give fabrication    *
C technology parameters.                      *
C***********************************************

      FUNCTION NGSOCAP(W)

      IMPLICIT REAL (A-Z)

            CGSO=3.0E-10

      NGSOCAP=CGSO*W

      RETURN
      END

C***********************************************
C            NGDOCAP SUBROUTINE               *
C                                             *
C       This subroutine evaluates NMOSFET gate-  *
C drain overlap capacitance using standard    *
C layout geometry and the give fabrication    *
C technology parameters.                      *
C***********************************************
```

```
      FUNCTION NGDOCAP(W)

      IMPLICIT REAL (A-Z)

            CGDO=3.0E-10

      NGDOCAP=CGDO*W

      RETURN
      END

C***********************************************
C           GEOM SUBROUTINE                    *
C                                              *
C     This subroutine calculates NFET/PFET     *
C drain/source area and perimeter based on a   *
C standard layout technique and the given      *
C fabrication technology rules.                *
C***********************************************

      SUBROUTINE GEOM(W,AS,PS)

      IMPLICIT REAL (A-Z)
      COMMON /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
      COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP
      COMMON /TECHNOLOGY/L,STEP
      COMMON /FUN/RB,CB


      PS=W+14.4E-6
      AS=W*L+22.68E-12

      RETURN
      END

C***********************************************
C     NFET_MIN_RESISTANCE SUBROUTINE           *
C                                              *
C     This subroutine approximates the per     *
C unit channel resistance for those NFETs that *
C mainly discharge in the linear region.       *
C***********************************************

      FUNCTION NFET_MIN_RESISTANCE(W,VG,VS)

      IMPLICIT REAL (A-Z)
      COMMON /C_R/PURL,PURS,PURP,K1,K2,K3,K4,K5,K6,K7
      COMMON /SETUP/WMAX,WMIN,WLP,WLN,VBN,VBP
      COMMON /TECHNOLOGY/L,STEP
      COMMON /FUN/RB,CB


      VGS=VG-VS
```

```
      VBS=VBN-VS

            EPSSI=11.7*8.85E-12
            Q=1.6022E-19
            UO=775E-4
            VTO=0.7
            VMAX=1.0E5
            PHI=0.6
            COX=6.9E-4
            NSUB=1.7E22
            XJL=3.5E-7
            GAMMA=1.1
            XJ=6.0E-7
            ETA=1.0

C...  CALCULATE INTERMEDIATE PARAMETERS

      US=UO
      PHIF=PHI/2.0
      VFB=VTO-PHI-2.0*SQRT(Q*EPSSI*NSUB*PHIF)/COX
      LEFF=L-2.0*XJL
      BETA=W/LEFF*US*COX
      VBI=VFB+PHI
      VBIN=VBI
      XD=SQRT(2.0*EPSSI/Q/NSUB)
      WD=XD*SQRT(PHI-VBS+VDS)
      WS=XD*SQRT(PHI-VBS)
      ALFAD=0.5*XJ/LEFF*(SQRT(1.0+2*WD/XJ)-1.0)
      ALFAS=0.5*XJ/LEFF*(SQRT(1.0+2*WS/XJ)-1.0)
      GAMMAS=GAMMA*(1.0-ALFAS-ALFAD)

      NFET_MIN_RESISTANCE=1.316/BETA/(VGS-VBIN-GAMMAS*(2*PHI-
VBS)**0.5)

      RETURN
      END

C******** END OF PRECHARGE PFET SIZING PROGRAM ***********
```

C******** END OF PRECHARGE PFET SIZING PROGRAM ***********

Appendix D

A Report on the Design, Layout, and Testing of 4-bit Parallel Full Adder

D.1 Introduction

Data processing speed is a key issue in today's computer system developments and, in fact, our present era is named the information age because of the incredible amount of data that can be processed and transferred around the world in an extremely short period of time.

This report attempts to show that very high speed VLSI chips can be obtained using 3µ-DLM fabrication technology. A simple and effective method of realizing logic functions directly from their truth table called *Switching Graph Method* is implemented to realize full adder test circuit. This full adder is incorporated with single phase dynamic latches that can be clocked at extremely high speeds.

A 4-bit parallel full adder is constructed from the basic full adder cell with a fully testable pipelined structure in order to demonstrate the possibility of implementing two dimensional systolic array structure with single phase clocking scheme operating at high throughput rates.

D.2 Circuit Design

The design process of the 4-bit parallel full adder is explained in detail in the following sub-sections;

D.2.1 Latches

True single phase dynamic latches presented in [D1] are adopted in the design of the 4-bit parallel full adder. These latches are capable of operating at very high speeds with reduced

charge sharing problems and a reduced number of transistors compared to other single phase dynamic latches. The latches are fully verified with SPICE simulations and given in Figure D.1.

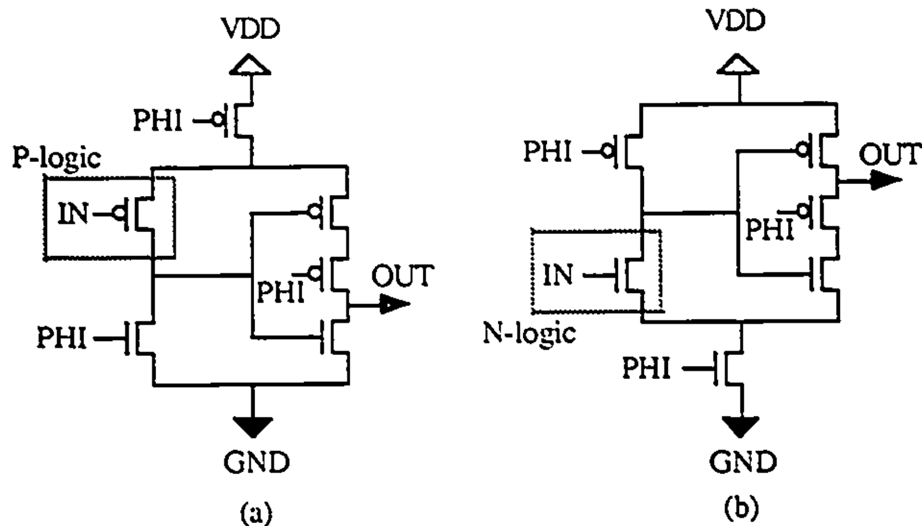

Figure D.1 Single Phase Dynamic Latches: (a) P-logic Latch (b) N-logic Latch

D.2.2 Switching Graph Method

Here we discuss a very simple and useful method of realizing a logic function directly from its truth table. This method is new and being investigated by the VLSI group at the University of Windsor and it is summarized in the following steps;

1. Generate a truth table of the logic function.
2. Construct a binary switching tree for each logic function.
3. Reduce the tree.
4. Equate the graph branches into transistors and the graph nodes into circuit nodes.
5. Incorporate the resulting circuits with latches and generate input vector complements as necessary.

Note that in step 2 we called the result a binary switching tree and in step 4 we called it a switching graph because the name tree is no longer valid after reduction since we might end up with nodes that have multiple branches from both directions. Let us now apply the above steps to a full adder logic function:

1. Generate a truth table of the full adder logic function

A	B	C	Sum	Carry
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Table D.1 Sum and Carry Truth Table

2. Construction a binary tree for the sum

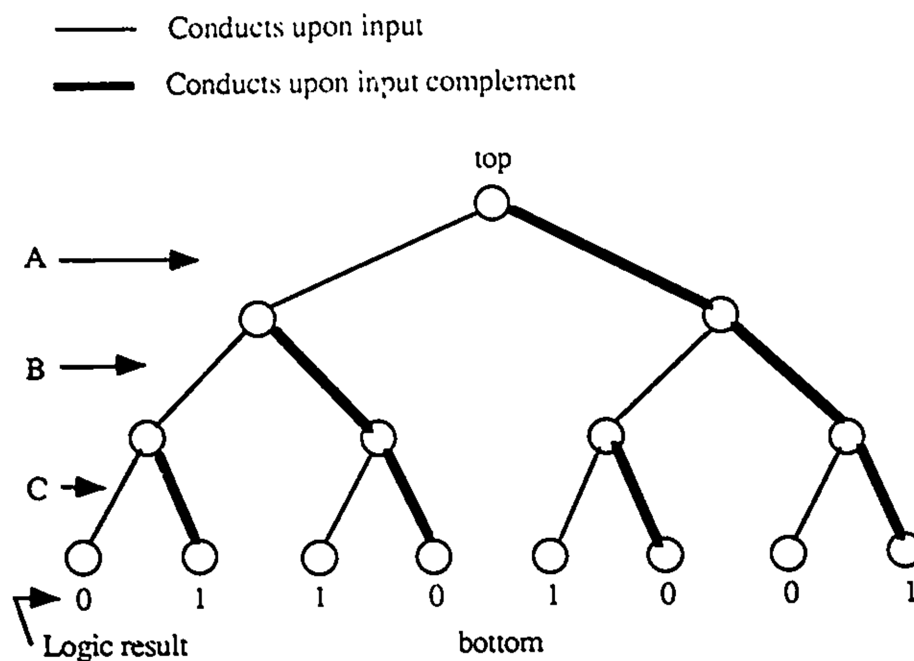

Figure D.2 Sum Binary Switching Tree

3. Reduce The Resulting Binary Tree

Now, if we wish to incorporate the logic function in an n-logic block latch then we need only one connecting path from the top to the bottom of the tree for each output logic 1 of the logic function. We discard all other branches that yield a 0 logic output. This results in the following sum reduced tree structure:

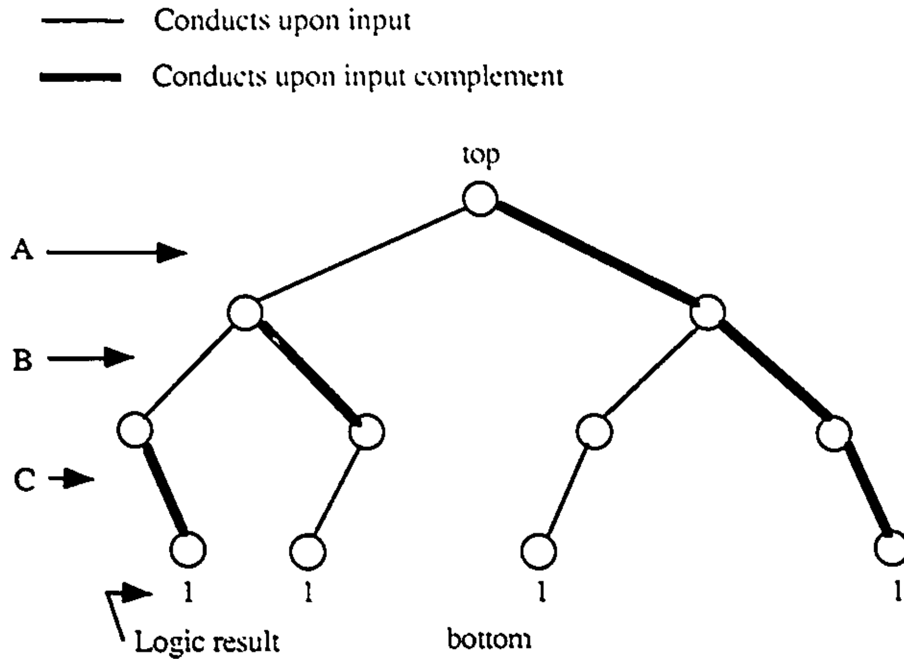

Figure D.3 Reduced Sum Binary Tree

It is possible to further reduce the tree by allowing cross communication or merging among the nodes by appropriate branches. This step is done by inspection at this stage, however, more detailed tree reduction rules are being developed by the VLSI research group at the University of Windsor and the interested reader may refer to the papers and internal reports of this research group. The final result may not be unique and different structures could be found as follows;

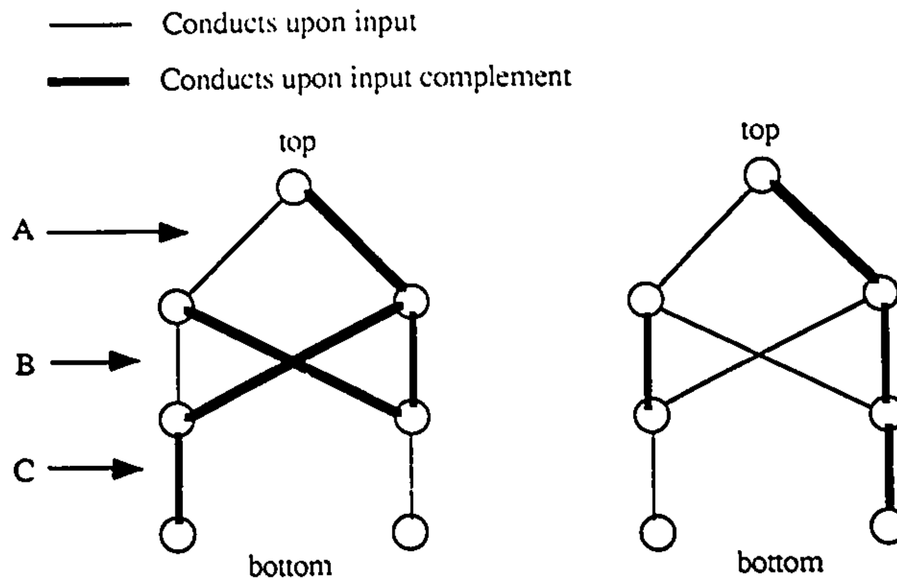

Figure D.4 Resulting Switching Graphs

4. Equating the switching graph into a circuit diagram

This step is straightforward. Since we chose n-logic block, the branches are equated to n-channel transistors and if the branch represents logic 1, then the input logic is fed directly to the gate of that transistor. If the branch represents logic 0, then the input logic complement is fed to the gate of that transistor. The two possible implementations of the logic sum are shown below;

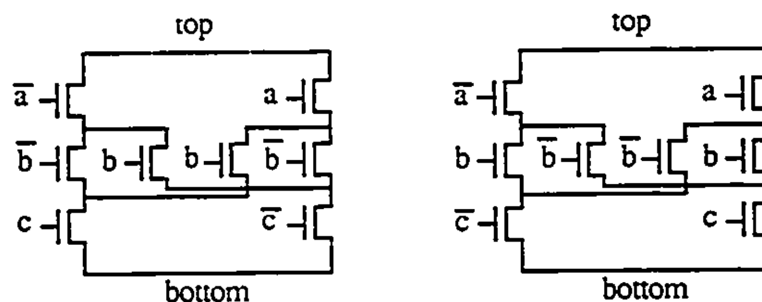

Figure D.5 Two Possible Realizations of the Sum

Similarly, the carry logic results in the following circuit;

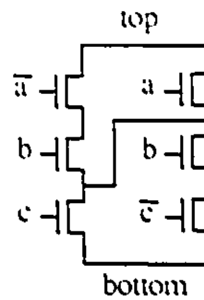

Figure D.6 Carry Realization

5. Complete Full Adder Dynamic Logic structure (n-logic block)

Figure D.7 shows the complete circuit structure for the sum and the carry along with the n-logic latch and input inversion stage. Note that clocked inverters are implemented where static inverters are sufficient (this was a very bad design decision that will be explained latter).

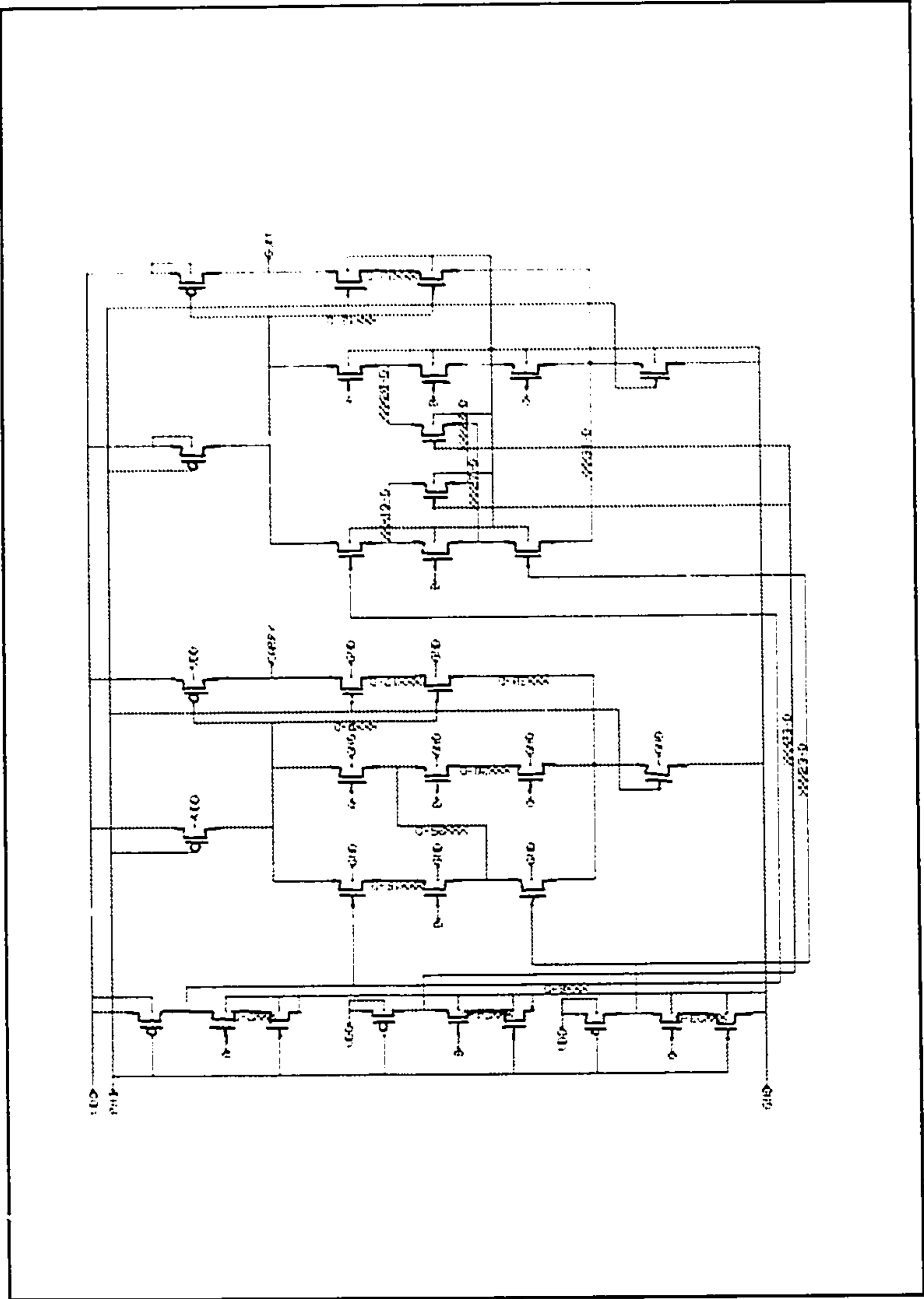

Figure D.7 Circuit Schematic for One Bit Full Adder

D.3 Full Adder Layout

The above method resulted in separate circuits for the sum and carry logic functions. The following sub-sections describe the steps taken to layout the one bit full adder cell and also show the transistor size optimization that has been carried out manually through the layout and SPICE simulations. All the layouts are done on the DAISY TERMINALS and the testing procedures are done by MASKAP II with the TEKTRONIX terminal. SPICE parameters are extracted directly from the layout and then the circuit simulated using HSPICE MOSFET level 2 simulations on the SUN workstation.

D.3.1 Initial Full Adder Cell Layout

Figure D.8 shows the initial one bit full adder cell layout. Note that the transistor sizes are chosen arbitrarily and the use of clocked inverters to generate input complements for the switching graph logic structure. The layout is generated with full custom approach (no automation is involved) and layout rules are followed to keep the silicon area as small as possible.

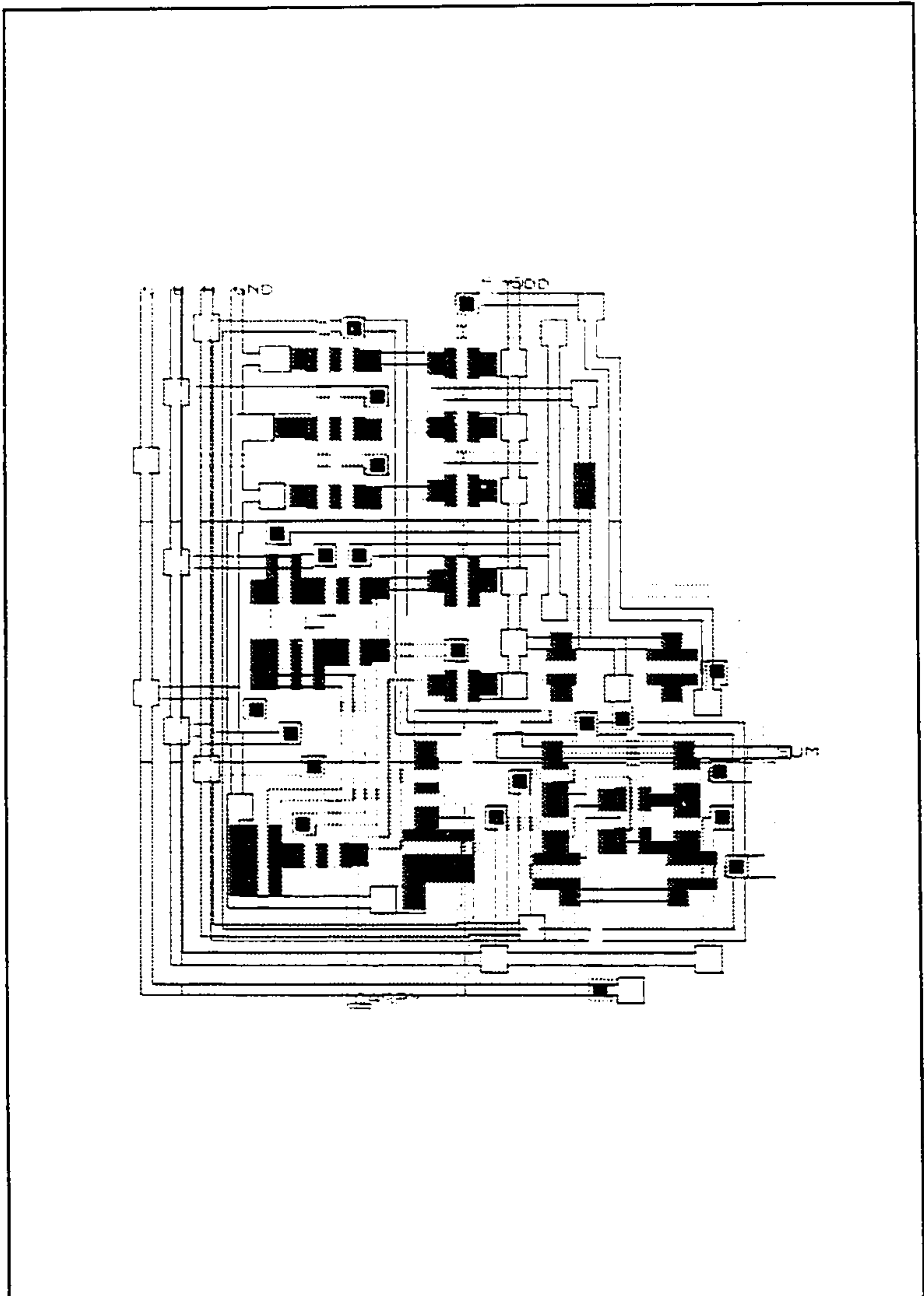

Figure D.8 One Bit Full Adder Cell Layout

D.3.2 Full Adder Cell Manual Transistor Sizing

SPICE parameters are extracted from the layout obtained in the previous sub-section and SPICE level 2 simulations are carried out to determine the clocking speed of the cell. At this point, a driving capability of 250FF is specified for the sum and carry outputs. This load specification should be adequate for driving subsequent cells. A worst case input vector setting is simulated along with the 250FF load capacitors on the output nodes and then the circuit nodes are examined for speed and the transistors connected to the slowest switching node are increased in size in the layout. SPICE parameters are extracted again from the modified layout and the simulation is repeated. This process is carried out several times until a clocking period of 6nS is achieved and the final cell layout is given in Figure D.9 and its SPICE simulation is given in Figure D.10.

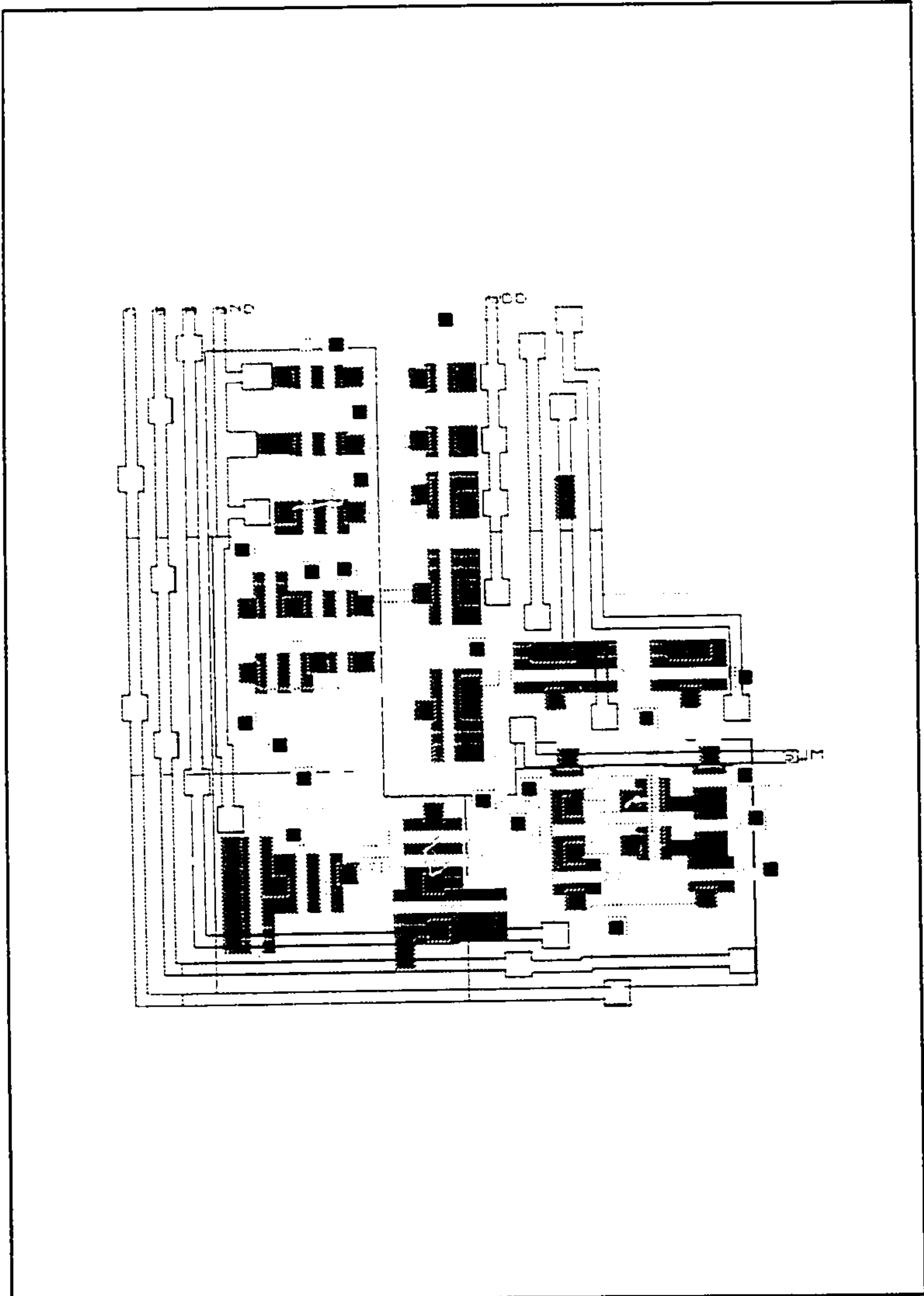

Figure D.9 Final One Bit Full Adder Cell Layout

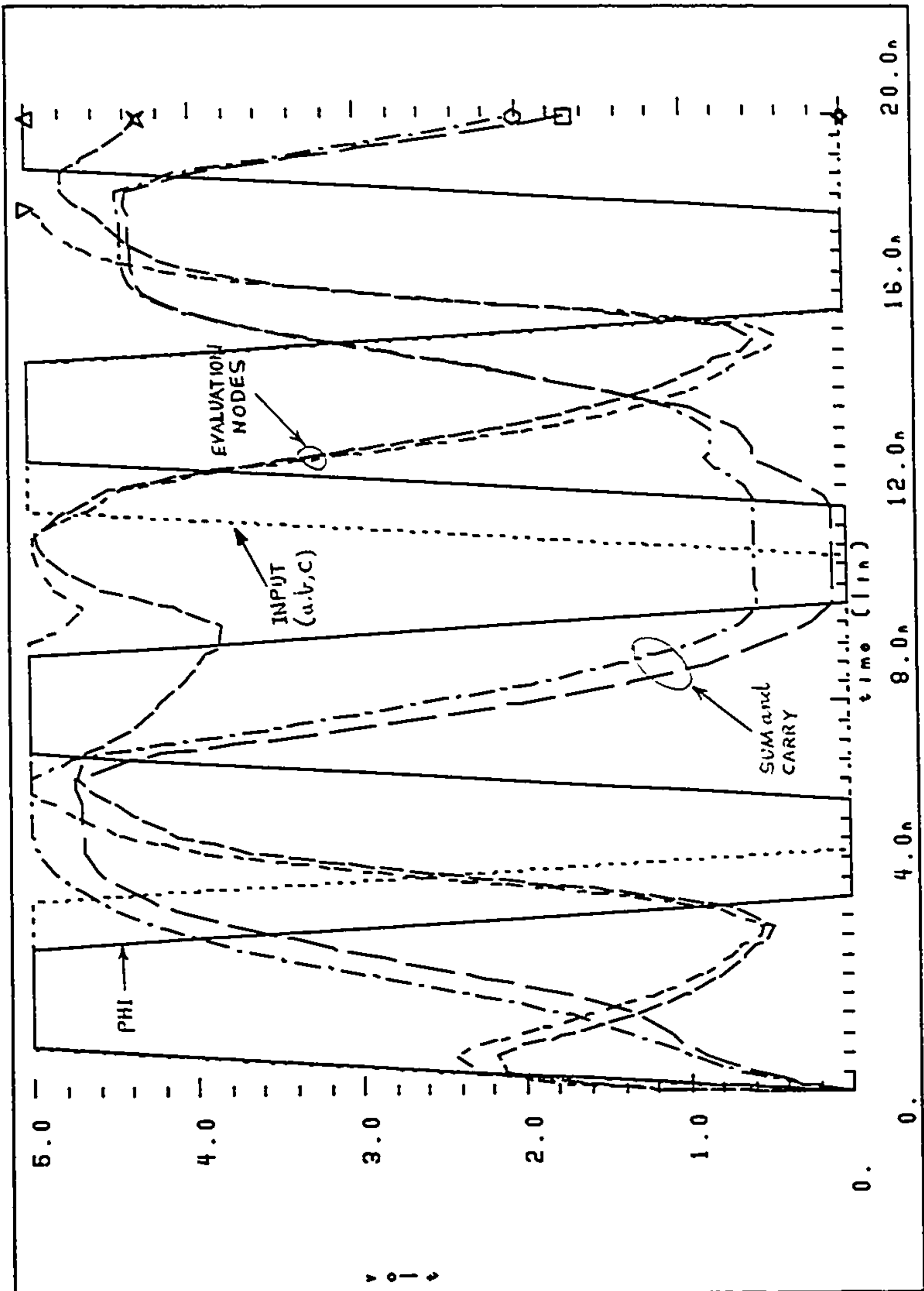

Figure D.10 SPICE simulation for the Final One Bit Full Adder Cell Layout

D.3.3 Full Adder Cell with P-type Latch

The full adder discussed so far is an n-logic block . To remove transparency between logic blocks, p-logic block should be used in between every two communicating n-logic blocks. P-type latch is used for the carry and its layout is incorporated with the full adder cell in order to save silicon area. This p-type latch should be able to drive a 150FF load which is the load of the input carry of the next full adder cell. Two cycles of layout were conducted to achieve the required driving capability in a clock period of 6nS. The final layout and its SPICE simulation is given in Figure D.11 and in Figure D.12 respectively.

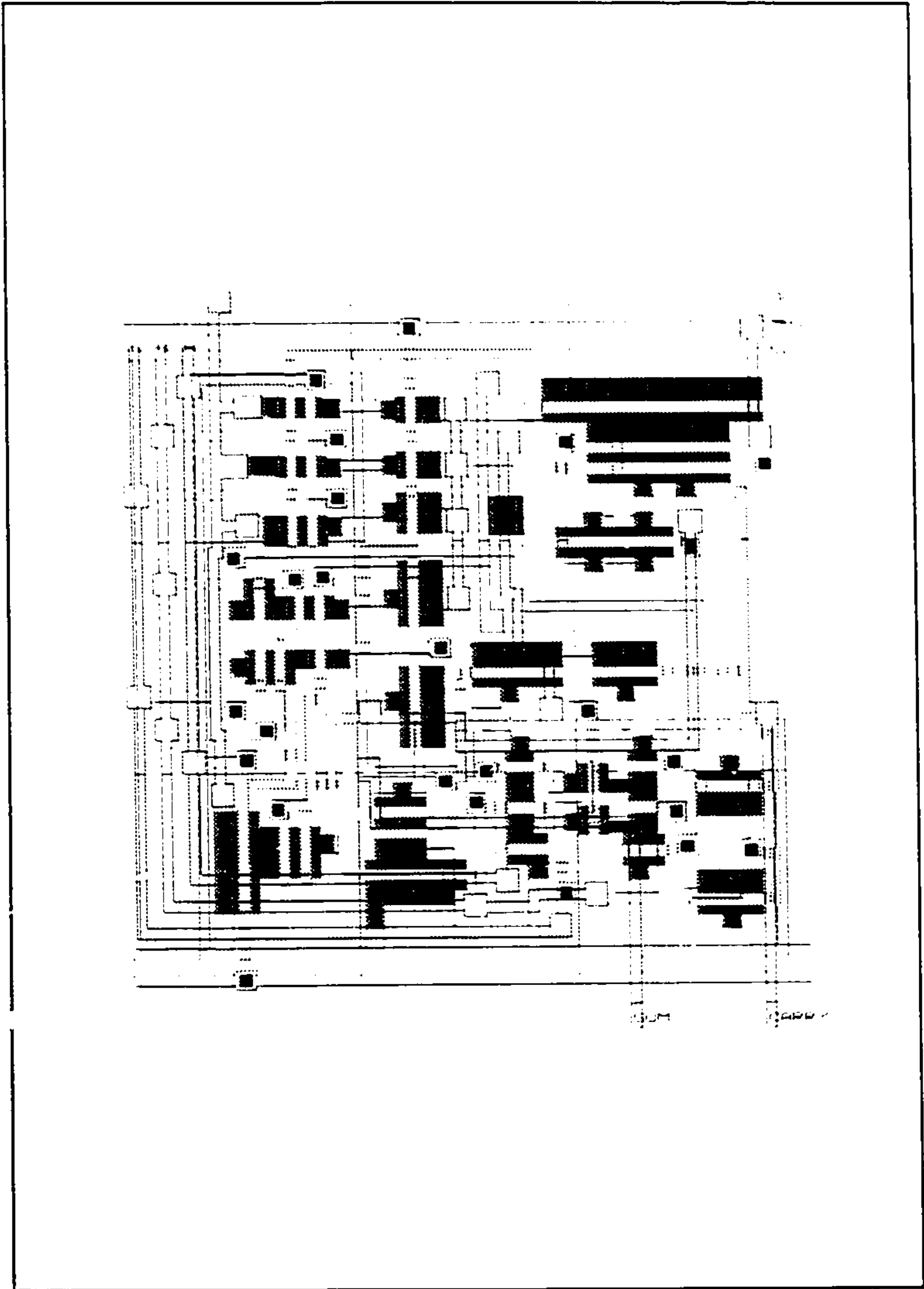

Figure D.11 One Bit Full Adder Cell Layout with P-logic Latch for the Carry

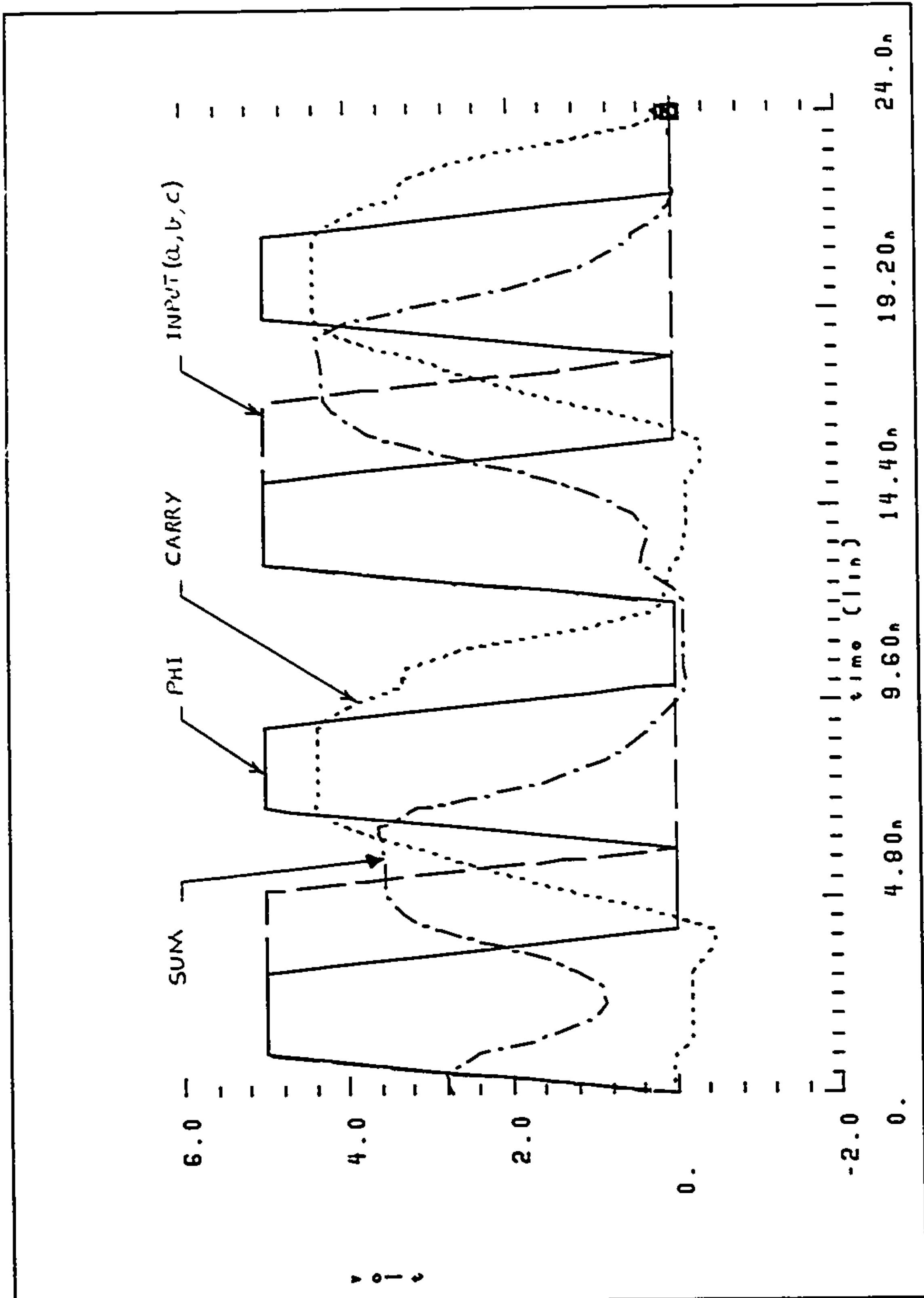

Figure D.12 SPICE simulation of the One Bit Full Adder Cell Layout with P-logic Latch

D.3.4 N-P Latch and P-N Latch Layouts

N-P latch stages (n-logic latch followed by p-logic latch) are needed to delay the input bits until the carry is ready from the previous full adder cell. These N-P lathes should have a driving capability of at least 100FF load which is the load of A and B inputs of the full adder cell. P-N latch stages are needed to delay the sum of the lesser significant bit until the next sum is ready. The P-N latch should be able to drive another latch (30FF load) and output pads or probes and a 100FF driving capability is chosen for this latch also. The final layouts of the latches and their SPICE simulations are given in Figures D.13-16.

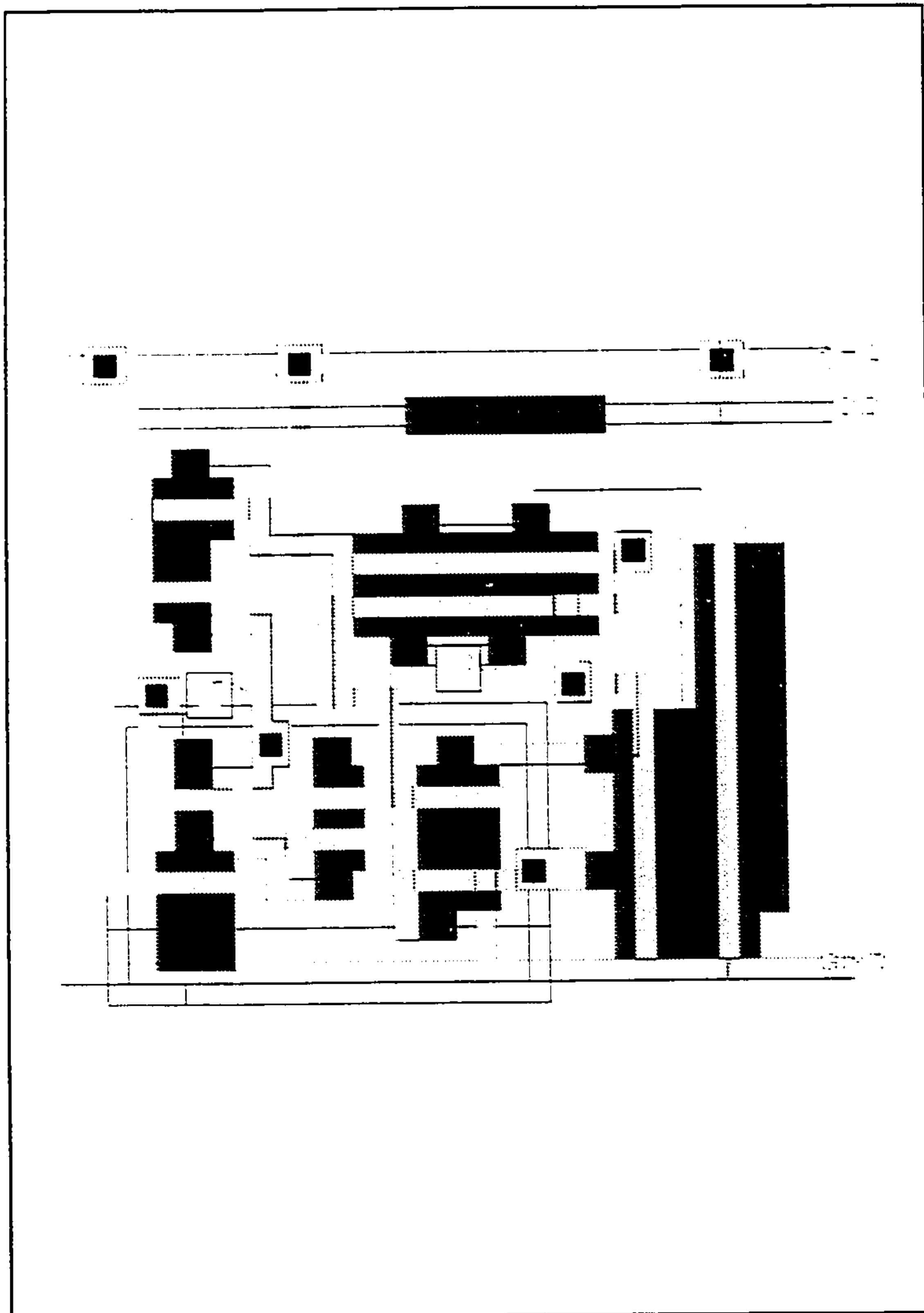

Figure D.13 N-P Latch Layout

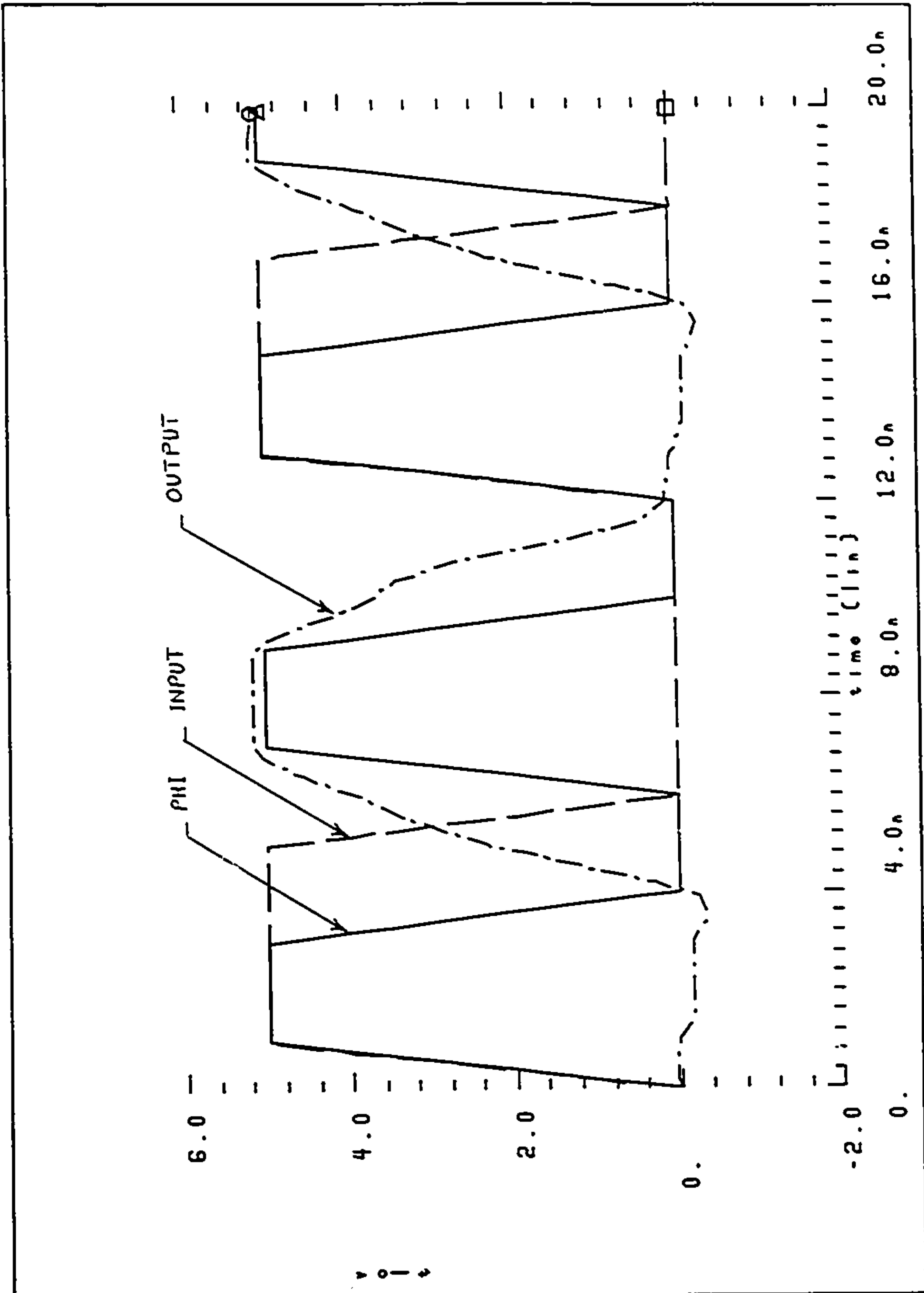

Figure D.14 N-P Latch SPICE Simulation

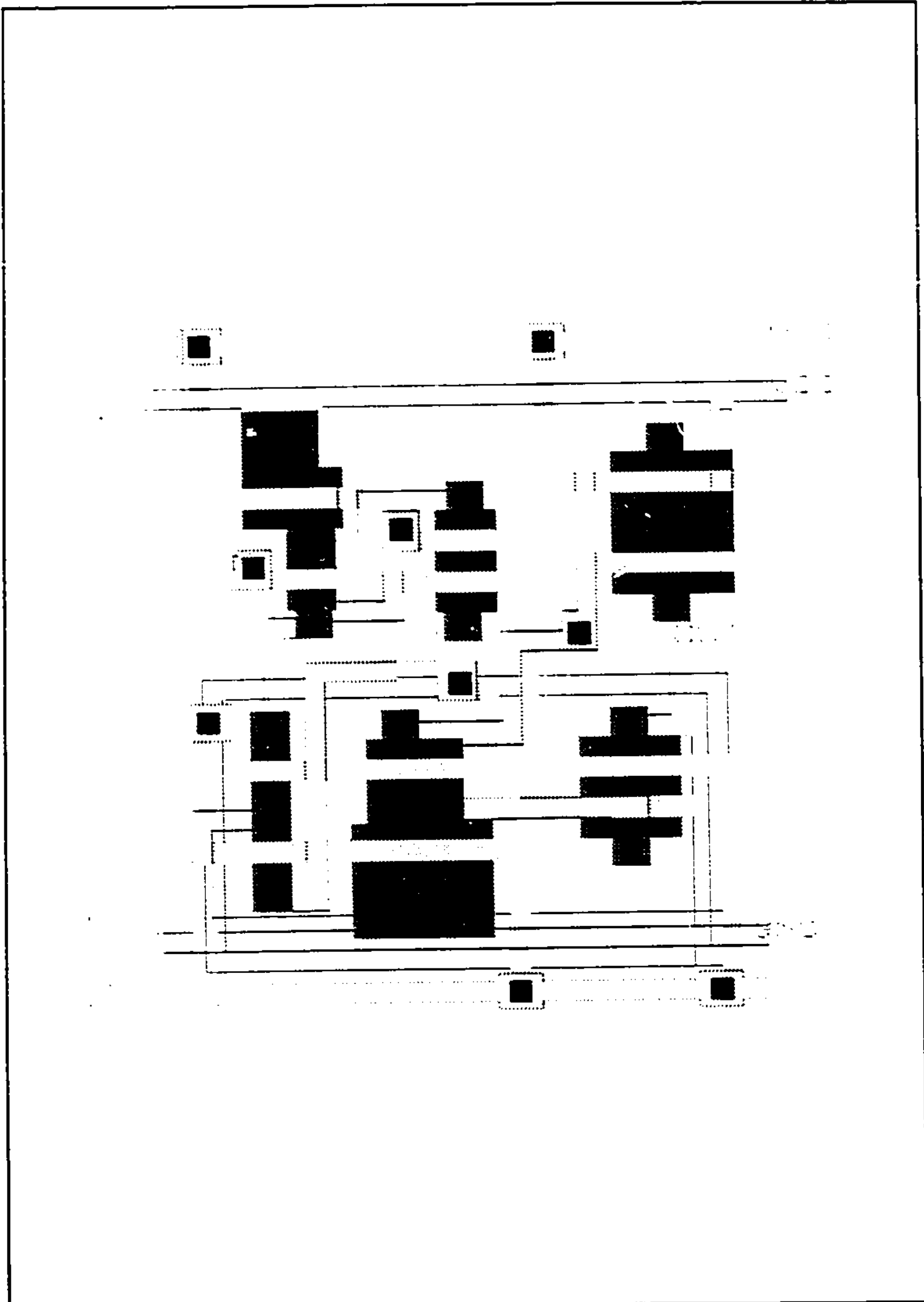

Figure D.15 P-N Latch Layout

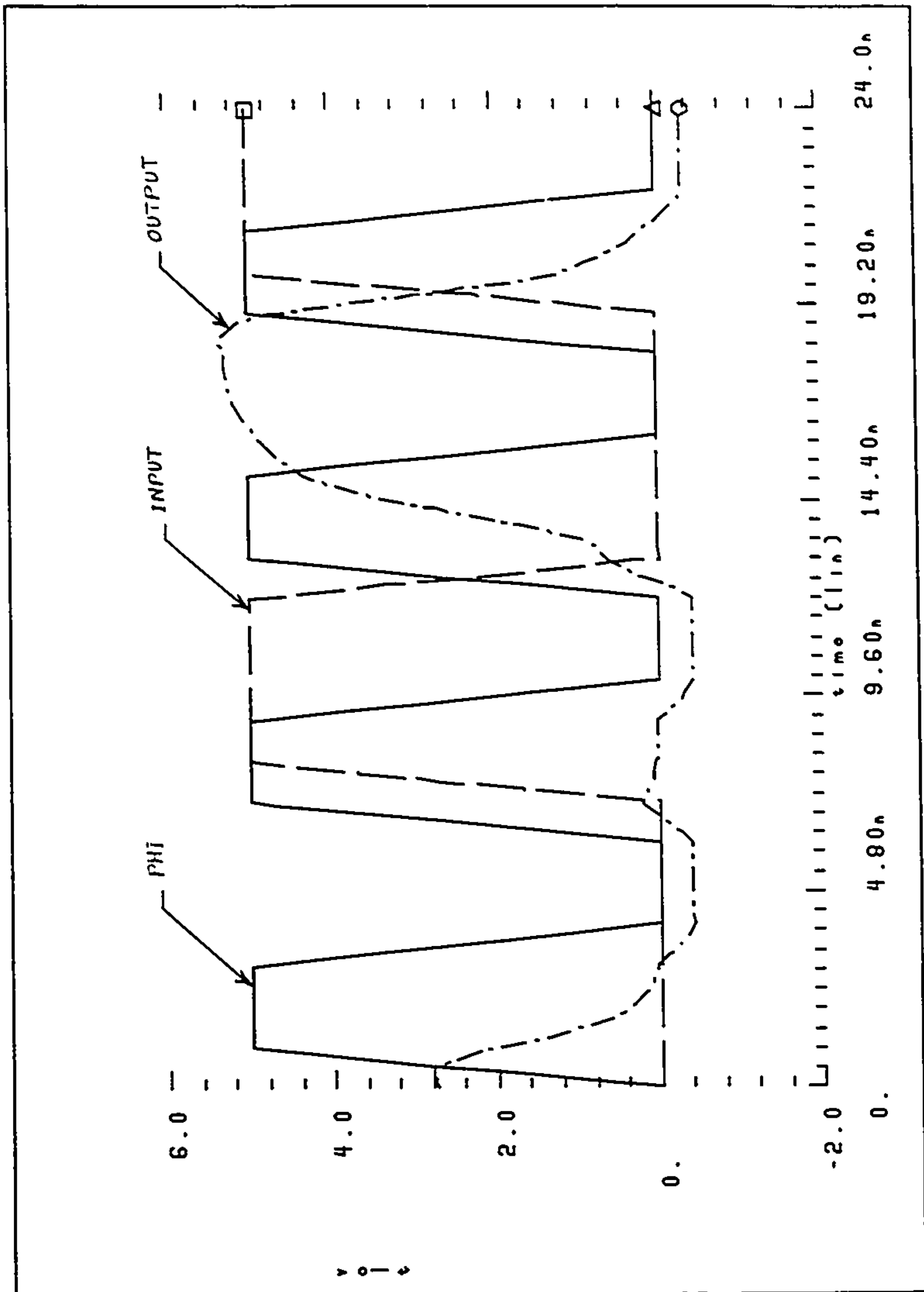

Figure D.16 P-N Latch SPICE Simulation

D.4 4-bit Parallel Full Adder Layout

The construction of 4-bit parallel full adder is straightforward with the use of the cells developed in section D.3. Figure D.17 shows the circuit schematics of the 4-bit parallel full adder and Figure D.18 shows its layout. Note that the input data and the resulting bit sums propagate from top to bottom while the carry propagates diagonally through the full adder cells. This will introduce clock skew problems and their severity is yet to be seen after a complete chip testing.

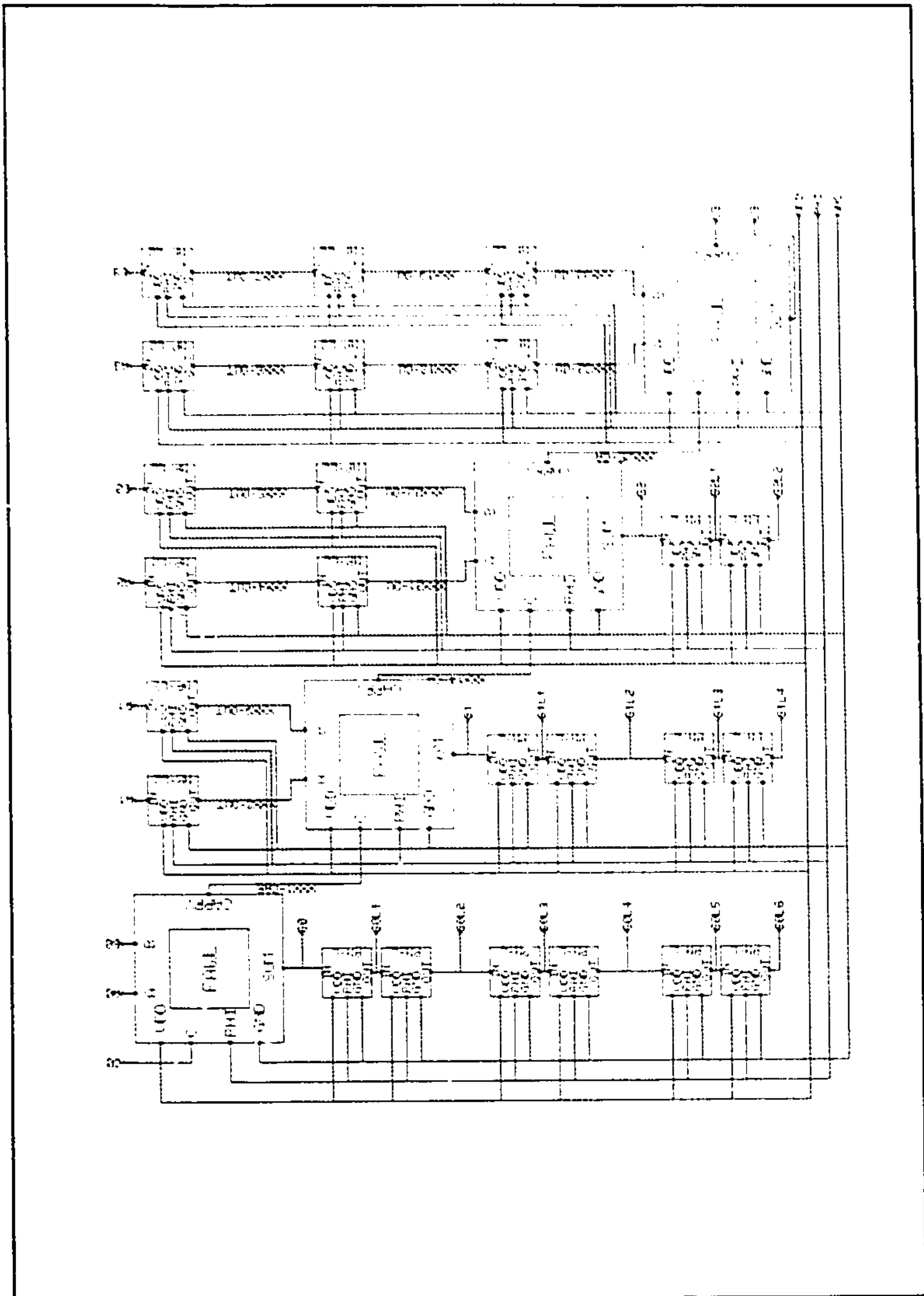

Figure D.17 Circuit Schematic for 4-bit Parallel Full Adder

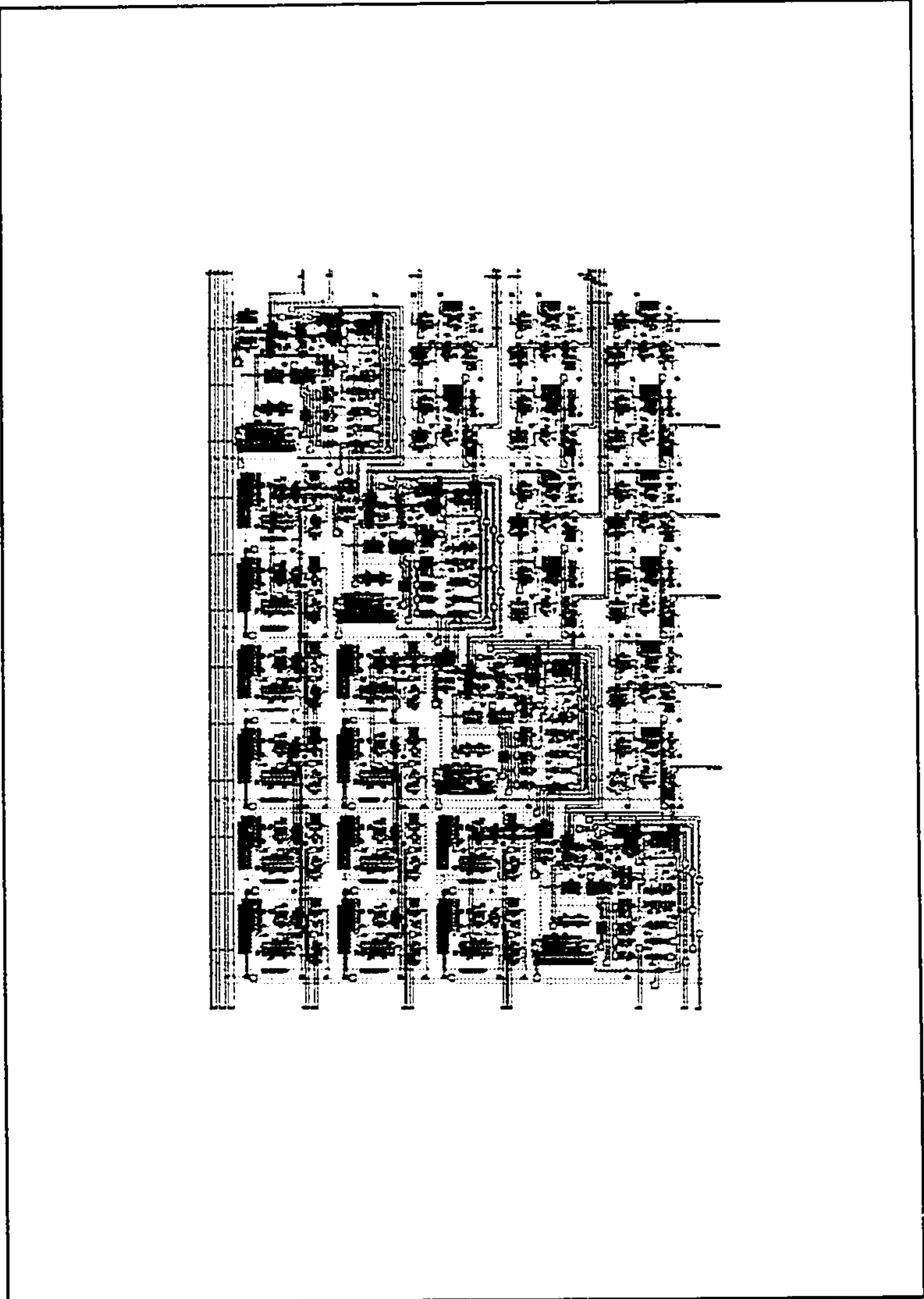

Figure D.18 4-bit Parallel Full Adder Layout

D.5 Proposed Testing Scheme

Input pads are generally capable of inputting data at much higher rates than output pads and here we propose a testing scheme that utilizes this fact. The timing diagram depicted i Figure D.19 allows the input data to flow at such high speed and the output result to flow at the desired low speed without degrading the test integrity. Note that the output reading periods should be long enough for the output pad to respond properly to the output signal of the dynamic logic structure and should be short enough for the output node not to lose its evaluated state due to leakage currents.

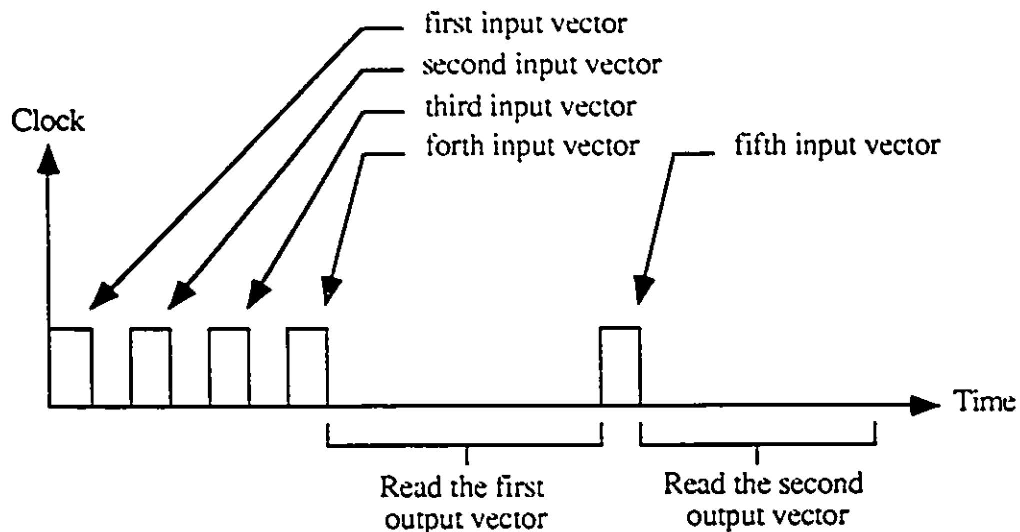

Figure D.19 Test Vector Timing Diagram

D.6 Chip Testing Results

After fabrication, the 4-bit parallel full adder chip is received and the testing process is commenced using the ASIX-2 tester with IBM compatible computer interface and TEKTRONICS digital scope. The testing process is divided into three stages: the first stage is to test the functionality of the full adder cells, the second stage is to test the whole chip

for functionality and performance, and the third stage is to study the two dimensional clock and data movement at high speeds. The first stage testing is carried out for the most significant bit full adder cell since the both the sum and the carry are available through output pads and Figure D.20 shows a sample input/output scope plots of the actual testing results.

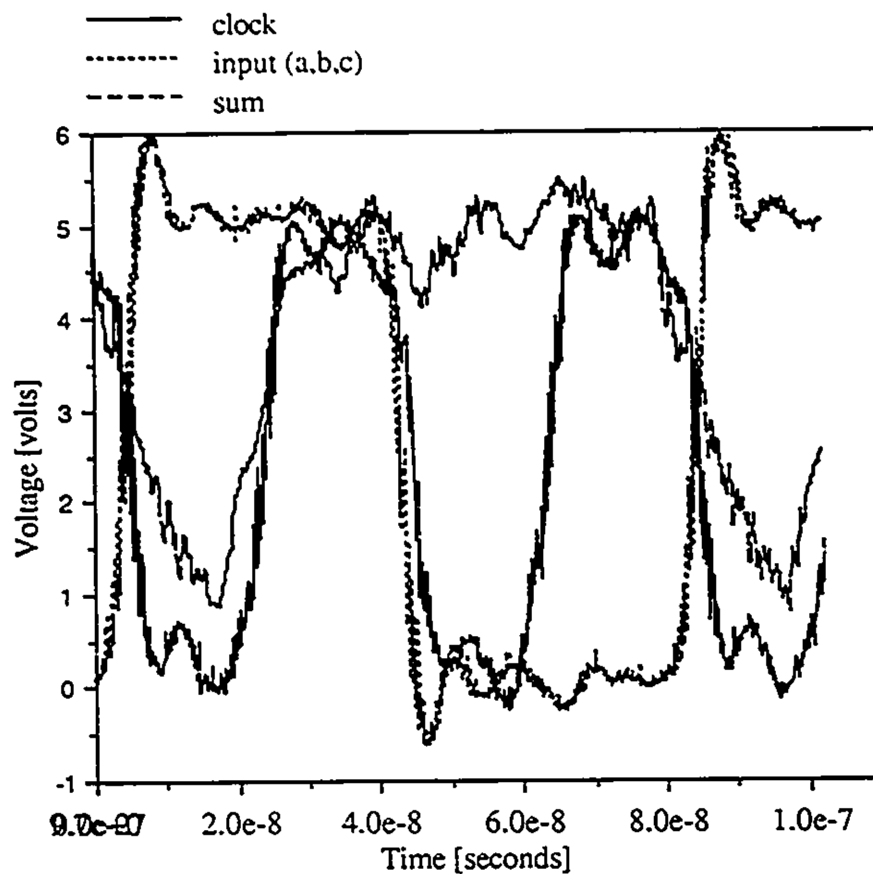

Figure D.20 Input/Output Scope Plot of the Full Adder Cell

Table D.2 shows the tested logical outputs of the full adder cell and its inputs. Note that all other inputs to the chip are set to zero and hence the carry input to the cell is also zero. Out of these four tested outputs one is wrong (third row) where the carry switched high instead of staying low and this occurs because the evaluation node voltage drops low due to charge sharing rather than correct logic evaluation. In section 4.3 of this thesis, the charge sharing

problem is investigated thoroughly and in view of the discussions made earlier, the charge sharing problem of the full adder cell caused by the dynamic inverters that used to obtain input vector complements. These inverters made the input vector settle during the beginning of the evaluation cycle rather than during the precharge cycle, therefore defeating the useful property of the single phase dynamic latches used in the chip design. To rectify the problem, simply replace the dynamic inverters by static inverters at the input stage of the full adder cell and this will reduce the area and complexity of the cells by reducing the number of transistors at the input stage (static inverters implemented by two transistors while dynamic inverters use at least three transistors). Further tests are terminated at this stage but the chip could be useful in testing two dimensional clock and data movements at high speeds.

A	B	C	Sum	Carry
1	1	0	0	1
0	1	0	1	1 (wrong)
1	0	0	1	0
0	0	0	0	0

Table D.2 Logical Test Results of the Full Adder Cell

D.7 Conclusions

It has been demonstrated that logic function implementation using switching graph method is simple and straight forward. High speed throughput rates are possible using dynamic single phase clock latches shown in this report. A full adder unit cell has been designed using the switching graph method and implemented using the single phase clock dynamic

latches. A full adder cell is successfully laid out and its SPICE simulations show encouraging results for the 6nS clocking period and 250FF load driving capability.

In order to demonstrate a two dimensional array using the single phase clocking scheme, a 4-bit parallel adder is constructed using the three cells obtained previously; N-P latch, P-N latch, and the full adder cells. SPICE simulations shows encouraging results for this scheme with the targeted clock period of 6nS. The total silicon area occupied by the 4-bit parallel full adder with the latches is about 900μ-1300μ (excluding pads) which contains 444 transistors.

The fabricated chip was tested and the testing results show a faulty full adder cell output due to the charge sharing problem discussed in section 4.3 of this thesis. To rectify the problem, simply replace the dynamic inverters by static inverters at the input stage of the full adder cell and this will reduce the area and complexity of the cells by reducing the number of transistors at the input stage (static inverters implemented by two transistors while dynamic inverters use at least three transistors).

References

[D1] J. Yuan and C. Svensson, "High-Speed CMOS Circuit Technique", IEEE Journal of Solid-State Circuits, Vol. 24, No. 1, February, 62-69, 1989.

VITA AUCTORIS

Sami Bizzan was born in February 25, 1965 in Tripoli, Libya. He completed his high school education at El-Nusur High School in Tripoli, Libya and came to Canada in 1983. In 1989, he graduated from University of Windsor with a Bachelor of Applied Science in Electrical Engineering and also completed the Masters of Applied Science requirements in 1991 at the University of Windsor. At this time, he is happily married with a lovely sun called Mohammed. Currently, he is doing his Ph.D with Dr. G. A. Jullien and his interests are high performance DSP structures, RNS arithmetics, and pipelining schemes.