Multilevel Interconnect Architectures for Gigascale Integration (GSI)

A Thesis
Presented to
The Academic Faculty

by

Raguraman Venkatesan

In Partial Fulfillment
of the Requirements for the Degree of
Doctor of Philosophy in Electrical and Computer Engineering

School of Electrical and Computer Engineering
Georgia Institute of Technology
February 2003

Copyright © 2003 by Raguraman Venkatesan

Multilevel Interconnect Architectures for Gigascale Integration (GSI)

Approved:

Prof. James D Meindl, Committee Chair

Prof. Jeffrey A Davis, Co-Advisor

Prof. Donald S Wills

Prof. Gary May

Prof. William Hunt

Prof. Paul Kohl
(School of Chemical Engineering)

Date Approved 2/3/03

DEDICATION

To

My parents

MR. S.VENKATESAN

&

MRS. SARASWATHY VENKATESAN

and my sister and brother-in-law

DEEPA & NARAYAN.

ACKNOWLEDGEMENTS

Wow! What a terrific train ride this has been! The engine is happy to have brought the coaches, this thesis, to its destination. My parents built the engine, and my sister painted it. Without them, there would be no engine, no train and no train-ride. The Meindl Metro Company drew the maps, laid the tracks and maintained the signals. It always knew where the train was supposed to go. Driver Davis drove the train remarkably through dark tunnels, up steep mountains and past lush meadows, stopping at the right stations and speeding over narrow bridges. The mostly pedagogic, sometimes frivolous, and usually philosophical, discussions with the guys in the GSI sweatshop - Jim, Azad, Chen, Tony, Raghunath, Hiren, Muhannad, Kaveh, Keith, Chirag, Azeez - was the coal that kept the engine running, and the oil that kept the wheels turning. Jennifer took excellent care of the train, just as a "GSI-Mom-away-from-Mom" would. And, my roommates, friends and other random people were the cheerful group who showed up at every station, cheering the train - and giving the occasional kick-in-the-back - making the journey memorable. I am just the engine, and I am glad I have reached the destination. What a terrific ride this has been! Thank you, everyone.

TABLE OF CONTENTS

DEDICATION iii

ACKNOWLEDGEMENTS iv

LIST OF TABLES xi

LIST OF FIGURES xiv

SUMMARY xx

1 INTRODUCTION AND BACKGROUND 1

1.1 Introduction 1

1.2 Multilevel interconnect networks 3

1.3 Inductance models for high speed interconnect systems 8

1.4 Inductance effects on multilevel architecture 10

1.5 Software tool 11

1.6 Summary 11

2 *N-TIER* MULTILEVEL INTERCONNECT DESIGN METHODOLOGY 12

2.1 Introduction 12

2.2 Optimum *n-tier* design methodology 13

2.2.1 Wiring demand models 16

2.2.2 Time delay models 17

2.2.3 *n-tier* design algorithm 19

2.3 Performance enhancement using optimized *n-tier* architectures 22

2.3.1 2-tier design 23

2.3.2 Conventional multilevel network designs 24

2.3.3 Optimized *n-tier* design 25

2.3.4 Minimum macrocell area 27

2.3.5 Maximum clock frequency 28

2.3.6 Minimum number of metal levels 28

2.4 Optimum aspect ratio 32

2.4.1 Aspect ratio design plane 32

2.4.2 Aspect ratio optimization 38

2.4.2.1 Maximum clock frequency optimization 38

2.4.2.2 Minimum area optimization 39

2.4.2.3 Minimum power optimization 40

2.4.2.4 Minimum number of levels optimization 41

2.5 Discussion of assumptions 42

2.6 Summary 44

3 OPTIMAL REPEATER INSERTION IN MULTILEVEL INTERCONNECT SYSTEM DESIGN 45

3.1 Introduction 45

3.2 Repeater insertion models and methodology 46

3.2.1 Time delay models for repeater insertion 47

3.2.2 Gate area models 50

3.2.3 Power dissipation models 50

3.2.4 Repeater insertion methodology 51

3.3 Performance enhancement using repeaters for *n-tier* architectures . . 54

3.3.1 Minimum macrocell area 54

3.3.2 Maximum frequency optimization 58

3.3.3 Minimum number of metal levels 60

3.3.4 Minimum power dissipation 62

3.4 Prospects of repeater insertion for GSI 66

3.5 Summary 69

4 PHYSICAL MODELS FOR TRANSIENT RESPONSE OF DISTRIBUTED *RLC* INTERCONNECTS 71

4.1 Introduction 71

4.2 Single line transient model with step input response 75

4.2.1 Derivation of an inverse Laplace transform identity 76

4.2.2 First reflection 86

4.2.3 Higher order reflections 91

4.2.4 Comprehensive model 98

4.3 Single line transient model - finite rise time input 102

4.3.1 First reflection 103

4.3.2 Higher order reflections 108

4.3.3 Comprehensive model 112

4.4 Crosstalk induced in coupled lines 117

4.5 Summary 118

5 COMPACT MODELS FOR TIME DELAY, CROSSTALK AND REPEATER INSERTION 120

5.1 Introduction 120
5.2 Unified time delay model - step input 121
5.2.1 Open Circuit Termination 123
5.2.2 Lossless line with a load capacitance 124
5.2.3 Lossy *rlc* line with load capacitance 125
5.2.4 Unified Time Delay Model 127
5.3 Unified time delay model - ramp input 128
5.3.1 Lossless line 131
5.3.2 Lossy *rlc* line 132
5.3.3 Unified delay model 132
5.4 Optimum repeater insertion 134
5.4.1 Unified time delay model for repeater insertion 134
5.4.2 Optimum repeater insertion 135
5.5 Crosstalk in two coupled lines 138
5.6 Delay-Crosstalk design space 144
5.7 Global interconnect width optimization 147
5.8 Summary 147

6 EFFECTS OF INDUCTANCE ON *N-TIER* MULTILEVEL ARCHITECTURES 148

6.1 Introduction 148
6.2 Time delay and repeater insertion 148
6.3 Crosstalk 151
6.4 *n-Tier* design case study comparisons 153
6.4.1 *rc* models, without repeaters 154
6.4.2 *rc* models, with repeaters 156

6.4.3 *rlc* models for delay, crosstalk and repeater insertion 159
6.5 Non-ideal return paths 165
6.6 Optimum aspect ratio using *rlc* models 171
6.6.1 Aspect ratio design space 173
6.6.2 "Low power" vs "high performance" aspect ratios 176
6.6.3 Optimum aspect ratio 178
6.7 Summary 180

7 *MINDS* - MULTILEVEL INTERCONNECT NETWORK DESIGN SIMULATOR 182

7.1 Introduction 182
7.2 Assumptions 183
7.3 Capabilities 184
7.3.1 Variable pitch usage model 184
7.3.2 Variable wiring efficiency 185
7.3.3 Layer assignment 185
7.4 Sensitivity of *MINDS* to system parameters 186
7.4.1 Rent's constants 186
7.4.2 Wiring efficiency factor 187
7.5 Rapid extraction of Rent's coefficients 188
7.6 Validation of area estimates with *MINDS* 190
7.6.1 Megacell area estimation 190
7.6.2 Total chip size estimation 192
7.7 Summary 194

8 CONCLUSIONS AND FUTURE WORK 195

8.1 Integrated system level inter- and intra- macrocell interconnect design 195

8.2 Wiring efficiency models 198

8.3 3-D and X architecture 198

8.4 Distributed *rlc* interconnect modeling 200

8.5 Conclusion of dissertation 200

APPENDIX A — CODE FOR *N-TIER* METHODOLOGY **203**

A.1 Notes 203

A.2 Main code 205

A.3 Sample input file 252

A.4 Output file 253

APPENDIX B — CODE FOR TRANSIENT RESPONSE OF DISTRIBUTED *RLC* LINES **255**

B.1 Notes 255

B.2 Main code 255

B.3 Sample input file 290

B.4 Output file 291

REFERENCES **292**

LIST OF PUBLICATIONS **299**

VITA **301**

LIST OF TABLES

1.1 MOSFET and interconnect latency for $1.0\mu m$, 100nm, and 35nm technology generation 3

2.1 Summary of various parameters used in the ASIC macrocell case study based on the ITRS. 23

2.2 Comparison of conventional designs with new optimized *n-tier* designs. 26

2.3 Interconnect parameters for the various design points 30

2.4 Summary of optimum aspect ratios for different system optimizations. 42

3.1 Summary of various parameters used for repeater insertion in the ASIC macrocell case study based on the ITRS 100nm technology generation (see Table 2.1 also). 54

3.2 Interconnect parameters for the various design points 55

5.1 Using repeaters to decrease crosstalk (and increase wiring density) without penalizing time delay. 146

6.1 L_{RLC} for different clock frequencies. 151

6.2 ITRS 2001 projections for 45nm technology generation. 154

6.3 Three different *n-tier* designs for the 45 nm generation, without using repeaters. 155

6.4 n-tier wire dimensions for 8 level design of Table 6.3. 156

6.5 Five different *n-tier* designs using *rc* models with optimum repeaters for the 45nm technology node generation. 156

6.6 Logic gate count and density for the five designs of Table 6.5. 157

6.7 Multilevel interconnect geometries for peak crosstalk=20% of V_{dd}. Crosstalk is calculated using the compact models in Chapter 5. 160

6.8 Multilevel interconnect geometries for peak crosstalk=10% of V_{dd}. Crosstalk is calculated using the compact models in Chapter 5. 162

6.9 Multilevel interconnect geometries for *worst case* peak crosstalk=20% of V_{dd}. Crosstalk is calculated using equations (6.15) and (6.16). . . . 163

6.10 Multilevel interconnect geometries for *worst case* peak crosstalk=10% of V_{dd}. Crosstalk is calculated using equations (6.15) and (6.16). . . . 164

6.11 Multilevel interconnect geometries for worst peak crosstalk=10% of V_{dd}, for ideal return paths assumption. 169

6.12 Multilevel interconnect geometries for worst peak crosstalk=10% of V_{dd}, for non-ideal return paths. 170

6.13 Multilevel interconnect design specifications for peak crosstalk=10% of V_{dd}. 175

6.14 Multilevel interconnect design specifications for peak crosstalk=15% of V_{dd}. 176

6.15 Multilevel interconnect design specifications for peak crosstalk=10% of V_{dd}, f_c=9.511GHz : "Tall" design. 177

6.16 Multilevel interconnect design specifications for peak crosstalk=10% of V_{dd}, f_c=9.511GHz : "Flat" design. 178

6.17 Optimum relative interconnect cross-sectional dimensions for each tier. The 130nm *n-tier* case study with f_c=1.6GHz and peak crosstalk of $0.1V_{dd}$, is compared to Intel's 130nm technology specifications. The pitch is expressed in units of minimum feature size F. 179

6.18 Optimum relative interconnect cross-sectional dimensions for each tier. The 45*nm* *n-tier* case study with f_c=11.511GHz and peak crosstalk of $0.1V_{dd}$, is compared to Intel's 130nm technology specifications. The pitch is expressed in units of minimum feature size F. 180

6.19 Multilevel interconnect design specifications using optimum aspect ratios for each tier. 181

7.1 Effect of variations in p on die size estimates of *MINDS* for a hypothetical microprocessor. 187

7.2 Effect of variations in k on die size estimates of *MINDS* for a hypothetical microprocessor. 188

7.3 *MINDS* predicted area and actual data area (logic) for Processors B and C. 193

7.4 *MINDS* predicted area and actual data area (logic) for Processors $E-H$.193

LIST OF FIGURES

1.1 Moore's Law - number of transistors on a chip doubles every two years [5]. 2

1.2 Reverse scaled multilevel interconnect architecture, from [6]. 4

1.3 Interconnect density distribution. The vertical axis represents the density of interconnects of length L in units of number of interconnects per gate pitch and the horizontal axis represents interconnect length in gate pitches. 7

2.1 Three dimensional view of reverse scaled multiple metal layers. 14

2.2 Cross-sectional view of an *n-tier* multilevel interconnect architecture. 15

2.3 Cross-sectional view of a multilevel interconnect architecture, where W, S and T are the wire width, spacing and thickness, H is the height of the inter-level dielectric, and $p = W + S$ is the wire pitch, respectively. 15

2.4 3-conductor 2-ground plane approximation: c_m is the line-to-line or mutual capacitance and c_g is the line-to-ground or ground capacitance. 18

2.5 Flowchart for *n-tier* design methodology. 21

2.6 Number of levels vs. macrocell area for 2-tier design 24

2.7 Number of levels versus macrocell area for the *n-tier* design of a 68M transistor macrocell. 27

2.8 Cross section of multilevel interconnect architecture for (a) conventional doubling of width scaling, and optimal *n-tier* designs for (b) minimum area, (c) maximum frequency, and (d) minimum number of metal levels. 31

2.9 Forbidden region for number of levels constraint. 35

2.10 Forbidden region for crosstalk constraint. 35

2.11 Unshaded area is the allowable region that satisfies the number of levels constraint and the crosstalk constraint. 36

2.12 The feasible region shrinks for $H/W = 1.0$. 37

2.13 The feasible region disappears for $H/W = 1.5$. 38

2.14 Optimum aspect ratio for maximum frequency optimization. The feasible region is shaded in light grey. 39

2.15 Optimum aspect ratio for minimum area optimization. Feasible region for A_m=1.62cm^2 is shaded in light grey, and that for A_m=1cm^2 is shaded in dark grey. The system clock frequency is kept constant at 2GHz. 40

2.16 Optimum aspect ratio for minimum power optimization. 42

3.1 (a) Interconnect with single-driver (b) Interconnect with k repeaters, size of each of which is h times that of a minimum-sized repeater. . . 47

3.2 Time delay vs. number of repeaters, normalized to optimum time delay and optimum number of repeaters, respectively. Using half the optimum number of repeaters increases time delay by only 10%. . . . 48

3.3 Top-down repeater insertion methodology. 52

3.4 Flowchart for *n-tier* with repeaters design methodology. 53

3.5 Minimum area optimization using repeaters for $n \leq 8$ levels and f_c=2GHz. 56

3.6 Minimum area optimization using repeaters for $n \leq 8$ levels and f_c=2GHz, with the additional constraint of power-limited area. 57

3.7 Maximum clock frequency optimization using repeaters for $A_m = 1.62cm^2$ and $n \leq 8$ levels. 59

3.8 Maximum clock frequency optimization using repeaters for $A_m = 1.62cm^2$ and $n \leq 8$ levels, when n_{cp}=5, instead of 10. 60

3.9 Minimum number of levels optimization using repeaters for $A_m = 1.62cm^2$ and f_c=2GHz. 61

3.10 Minimum power optimization using repeaters for $n \leq 8$ levels and f_c=2GHz. 63

3.11 Cross-section of n-tier architecture for (a) no repeaters, and for optimal repeater insertion for (b) minimum area, (c) maximum clock frequency and (d) minimum number of metal levels. 65

3.12 Projection of the number of metal levels for highly-connected logic macrocells from 1999 to 2014. 67

3.13 Projection of the effects of repeaters on macrocell design characteristics. 68

3.14 Tripartite functional allocation of transistors in a future GSI chip among logic, memory and communication transistors. 69

4.1 State-of-the-art global interconnect structures - (a) one and, (b) two signal interconnects (S) flanked by power/ground lines (G) and sandwiched between ground planes. 72

4.2 Distributed *rlc* interconnect of length L with a resistive driver (R_{tr}), load capacitance (C_L) and driven by a step input $V_{dd}u_o(t)$. The line resistance, capacitance and inductance per unit length are r, c and l respectively. 73

4.3 HSPICE simulations of the output transient response of a 1cm long distributed *rlc* interconnect. Neglecting the output capacitance (C_p) results in a 5% error, while neglecting both C_p and the input capacitance (C_L) results in a 14% error, for the 50% time delay. 74

4.4 Output transient response of a 1cm long distributed *rlc* interconnect to a step input waveform- complete model vs. HSPICE simulations for $C_L = 0$ and $C_L = 500fF$. 101

4.5 Output transient response of a 1cm long distributed *rlc* interconnect to a ramp input waveform - complete model vs. HSPICE simulations for $C_L = 0$, and input rise times, T_{rise}, of $20ps$ and $100ps$. 115

4.6 Output transient response of a 1cm long distributed *rlc* interconnect to a ramp input waveform- complete model vs. HSPICE simulations for $C_L = 500fF$, and input rise times, T_{rise}, of $20ps$ and $100ps$. 116

4.7 Transients in 1 cm long two coupled distributed *rlc* lines - complete model vs. HSPICE simulations for active and quiescent lines. 118

5.1 Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized load capacitance ($\frac{C_L}{cL}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$), for a lossless line ($\frac{rL}{Z_o} = 0$). 125

5.2 Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$) and normalized load capacitance ($\frac{C_L}{cL}$). 126

5.3 Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$) and normalized load capacitance ($\frac{C_L}{cL}$). 129

5.4 Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$), normalized load capacitance ($\frac{C_L}{cL}$) and normalized rise time ($\frac{T_{rise}}{t_f}$), for a ramp input. . 133

5.5 Normalized delay ($\frac{t_d}{t_f}$) vs. number of repeaters for a 3cm long line. The new model has an error of about 2% when compared to HSPICE simulations, which is much better than models in [13] and [30]. 136

5.6 Optimum number of repeaters vs. normalized line resistance ($\frac{rL}{Z_o}$). . . 139

5.7 Optimum size of repeaters vs. normalized line resistance ($\frac{rL}{Z_o}$). 140

5.8 (a) Typical rc charging circuit (b) Equivalent circuit representation used for approximate crosstalk model. 141

5.9 Normalized peak crosstalk vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$) and normalized load capacitance ($\frac{C_L}{cL}$). 143

5.10 Design plane - r vs number of repeaters, showing envelope around the region of operation. 145

5.11 Crosstalk and delay vs. number of repeaters, for fixed $r = 70\Omega/cm$. Inserting 8 repeaters (B) decreases crosstalk by 48% and increases time delay by 6%, compared to inserting 2 repeaters (A). 146

6.1 3-coupled lines (SSS) configuration, with ideal return paths. Orthogonal layers are approximated as ground planes. 166

6.2 2 signal lines sandwiched between 2 power/ground lines (GSSG configuration), with non-ideal return paths. 167

6.3 Quiescent line transients for 3-coupled (ideal return paths) and 2-coupled (non-ideal return paths) signal lines. 170

6.4 Aspect ratio design space for a 8.78M logic gate macrocell with maximum 8 metal levels. 174

7.1 Sensitivity of *MINDS* to Rent's exponent p. 187

7.2 Sensitivity of *MINDS* to Rent's coefficient k. 188

7.3 Sensitivity of *MINDS* to wiring efficiency factor e_w. 189

7.4 Normalized *MINDS* output area vs. actual data area for 16 different logic blocks belonging to Processor D. 191

8.1 Integrated system level interconnect design for inter and intra macro-cell wiring. 197

SUMMARY

The main objective of this thesis is to establish methodologies for the optimal design of the multilevel interconnect network in a GSI chip. A stochastic wiring distribution is used to develop an optimal *n-tier* reverse-scaling methodology for the design of each metal layer. An optimal repeater insertion methodology is developed that minimizes the number of metal layers, clock period, area or power dissipation. In order to design the multilevel interconnect network for future high speed GSI microprocessors, analytical models are developed for the transient response of distributed *rlc* interconnects. These models are used to develop compact models for time delay, crosstalk and repeater insertion in distributed *rc* and *rlc* interconnects. The new models are used to investigate the impact of inductance on a multilevel interconnect architecture and to optimize the aspect ratios of the physical dimensions of different metal levels. The models and methodologies have been incorporated into a software tool called *MINDS*, which stands for Multilevel Interconnect Network Design Simulator.

CHAPTER 1

Introduction and Background

1.1 Introduction

Rudolph Diesel's invention of the internal combustion engine in 1893 and Bardeen, Brattain and Shockley's invention of the solid-state transistor in 1948 can be considered to be the two pillars of twentieth century engineering. However, the rapid evolution of microelectronics technology over the last five decades and the extent of its penetration into everyday life is unparalleled in human history. Ever since the invention of the wheel by Neolithic Man, no other singular technology has had such a profound impact on mankind in so short a duration. The skeptics, some of whom were the stalwarts of this fledgling industry, lived to see their predictions woefully off the mark even in their own lifetimes. IBM chairman T.J. Watson predicted in the 1940's that "there is a world market for, maybe, five computers." In the 1960's, K. Olson, chairman and founder of DEC, maintained that "there is no reason why anyone would want a computer in their home." For the year 2000, the worldwide market of general purpose and embedded microprocessors alone was estimated to be around 5 billion chips [1].

The oft-quoted Moore's Law states that the number of transistors on a chip will double every two years [2], [3], [4], [5] (Figure 1.1). This empirical observation is

arguably both the carrot and the stick when considering forces that drive the semiconductor industry. Recent collaboration among a consortia of industrial, academic and government organizations has resulted in the development of the biennial International Technology Roadmap for Semiconductors (ITRS), which outlines the perceived characteristics, requirements and limitations of semiconductor technology over the next fifteen years [6]. The roadmap projects billion-transistor chips by the end of the current decade with transistor gate lengths as small as 45nm. Intel recently announced that it can design and build a billion-transistor microprocessor with four Itanium 2 cores sharing a large cache memory, using 65nm process technology by 2007 [7]. This is just one example of the industry frequently exceeding the roadmap

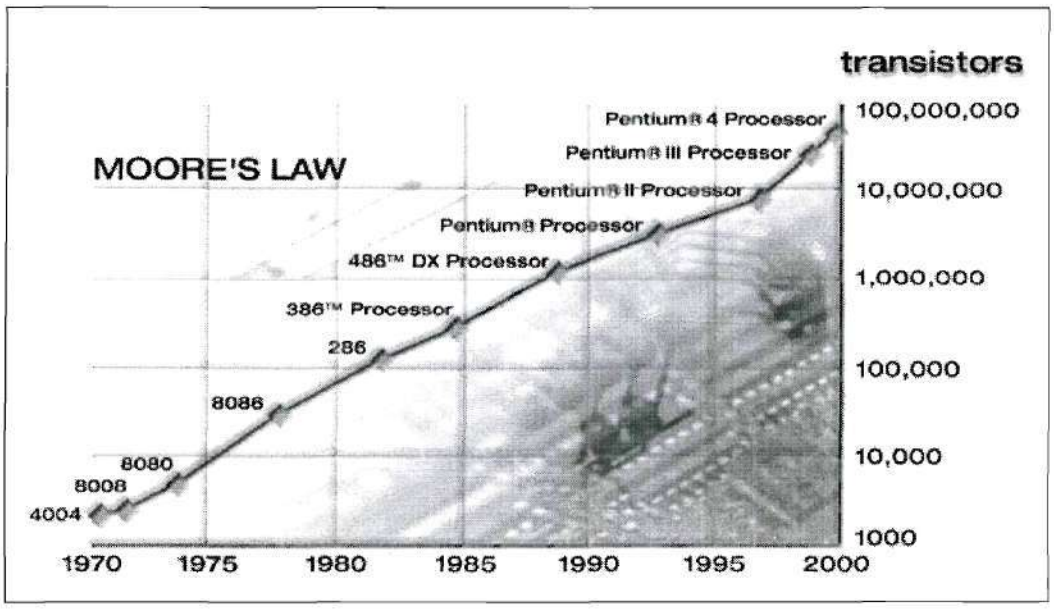


Figure 1.1: Moore's Law - number of transistors on a chip doubles every two years [5].

projections, necessitating significant upward revisions of the ITRS every two years (consequently causing considerable pain to graduate students).

The transistor and interconnnect limits for such Giga Scale Integration (GSI) systems have been elegantly elucidated in a hierarchical framework whose levels are described as fundamental, material, device, circuit and system [8], [9]. System limits are the most numerous and difficult to model in the hierarchy. They depend on all other limits and are the most restrictive ones [10]. Because the transistor limits are reduced with each technology generation, they are being increasingly overshadowed by the interconnect limits which are increasing with each succesive technology generation. As minimum lithographic feature sizes scale from 1μm to 35nm [11], the *rc* delay of a benchmark 1mm interconnect devolves from being 20 times faster to 100 times slower compared to the intrinsic switching delay of a minimum geometry MOSFET, as seen in Table 1.1 [12]. Therefore, the main objectives of this thesis are to research system level interconnect design and to propose novel interconnect design methodologies that optimize the power and performance of future GSI systems.

Table 1.1: MOSFET and interconnect latency for 1.0μm, 100nm, and 35nm technology generation

Technology Generation	MOSFET Switching Delay($t_d = CV/I$)	"rc" Response Time (L_{int}=1mm)	Time-of-flight (L_{int}=1mm)
1.0μm (Al, SiO_2)	$\approx 20ps$	$\approx 1ps$	$\approx 6.6ps$
100nm ($Cu, \kappa = 2.0$)	$\approx 5ps$	$\approx 30ps$	$\approx 4.6ps$
35nm ($Cu, \kappa = 2.0$)	$\approx 2.5ps$	$\approx 250ps$	$\approx 4.6ps$

1.2 Multilevel interconnect networks

As CMOS semiconductor technology approaches gigascale integration (GSI), the increasingly restrictive limits posed by interconnects on processor performance make it

imperative to optimize the wiring network of future technology generations [8], [9]. Multiple metal layers are required to accommodate all wiring resources on a chip [13], [14]. Since the maximum number of metal layers on a chip is determined by technology constraints, the chip area is increasingly becoming wire-limited i.e. the area is determined by the interconnect system, not by the transistors. This is a significant shift from earlier generations where the size of transistors determined the chip area.

The ITRS has projected the use of up to 10 levels of wiring for the 45nm technology node in the year 2010 [6]. Reverse scaled multilevel wiring networks are extensively used in current VLSI systems to mitigate the impact of wiring on chip size and system performance [15]. A typical multilevel wiring network is shown in Figure 1.2. Interconnects on adjacent layers are orthogonal to each other so that routing congestion is minimized. Vias run through the inter level dielectric (ILD) insulation

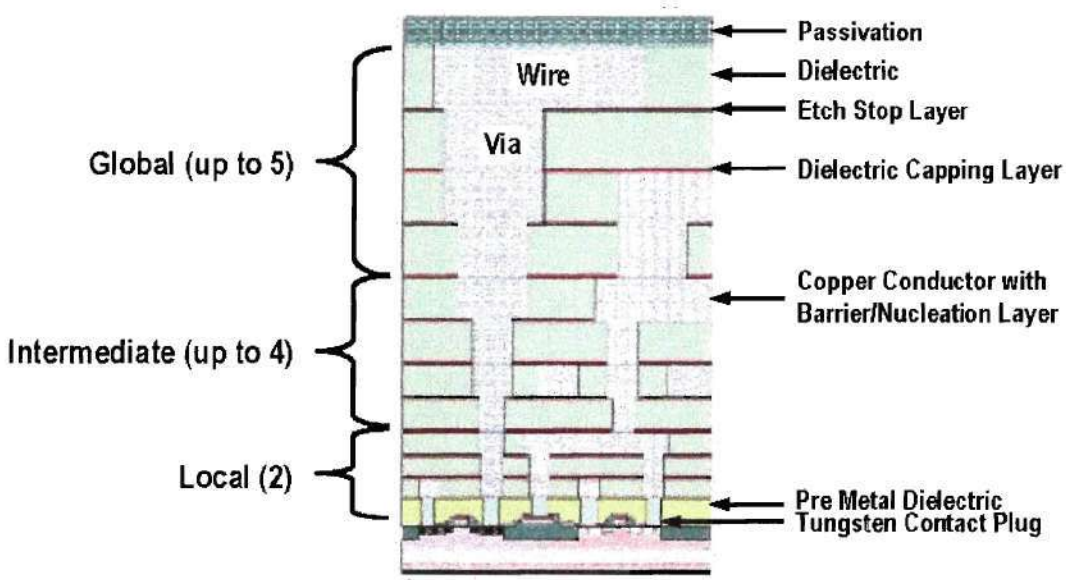


Figure 1.2: Reverse scaled multilevel interconnect architecture, from [6].

between metal layers to connect the substrate transistors with interconnects on the upper layers. Some vias also connect interconnects on different layers. The lower layers accommodate shorter wires and have smaller wire cross-sections. The wire widths increase monotonically from one layer to the next so as to control the delay of longer wires. Such a scaling arrangement permits data transfer at the same clock frequency independent of the length of the interconnect.

Layer assignment and wire sizing problems have become increasingly important in recent years. The main problem facing VLSI designers is determining what dimensions to use for the interconnects on each metal level. This problem is complicated by the dependence of system performance, yield and cost on the design of a multilevel interconnect structure. A very aggressive interconnect network design might decrease yield and increase cost to undesirable levels. On the other hand, a submarginal interconnect network design that inadequately utilizes system resources would significantly decrease a company's profit margin.

Sai-Halasz suggested a rudimentary thumb rule for designing the multilevel interconnect structure : double the wire widths on every successive pair of metal layers [16]. Such heuristic scaling rules have the advantage of being easy to implement, but the disadvantage of not being able to extract maximum performance for a set of specified technology constraints. More efficient scaling techniques are needed to meet the performance targets projected for the future. Kahng and Stroobandt describe a layer-assignment algorithm in [17], where wires are assigned to layers that are "best fit" depending on time delay constraints. The algorithm minimizes the total number of wiring layers for a fixed die area for predetermined upper and lower bounds on wire pitches of each layer. It also considers the impact of vias on available wiring area, using models from [18]. However, it is stated in [17] that certain delay constraints might result in "an area-inefficient solution." Also, since only optimal repeater width

and count is used throughout the algorithm, their algorithm "does not yet guarantee that [the solution] lies within the constraints of the repeater area" [17].

Chapter 2 describes a methodology to design optimal reverse-scaled multilevel interconnect networks using a stochastic wiring distribution [19], [20] that estimates the interconnect lengths *a priori*. The wiring distribution is based on Rent's rule which relates the number of I/O terminals T of a system of N gates as

$$T = kN^p \tag{1.1}$$

where k is Rent's coefficient and p is Rent's exponent. This empirical relationship is used to derive a closed-form expression for the complete signal wiring distribution in [19] given by :

Region 1: $1 \leq L < \sqrt{N}$

$$i(L) = \Gamma \frac{\alpha k}{2} \left(\frac{L^3}{3} - 2\sqrt{N}L^2 + 2NL \right) L^{2p-4} \tag{1.2}$$

Region 2: $\sqrt{N} \leq L \leq 2\sqrt{N}$

$$i(L) = \Gamma \frac{\alpha k}{6} \left(2\sqrt{N} - L \right)^3 L^{2p-4} \tag{1.3}$$

where

$$\Gamma = \frac{2N\left(1 - N^{p-1}\right)}{\left(-N^p \frac{1+2p-2^{2p-1}}{p(2p-1)(p-1)(2p-3)} - \frac{1}{6p} + \frac{2\sqrt{N}}{2p-1} - \frac{N}{p-1}\right)}, \tag{1.4}$$

$i(L)$ is the interconnect density function, L is the interconnect length in units of average gate pitch, $\alpha = \frac{\text{fanout}}{\text{fanout}+1}$ and Γ is a normalizing factor. One gate pitch is equal to the distance between neighbouring logic gates in a logic macrocell that is modelled as a homogenous array of random logic gates, and is equal to $\sqrt{\frac{A_m}{N}}$, where A_m is the macrocell area. The wiring distribution expressions in 1.2-1.4 are verified against

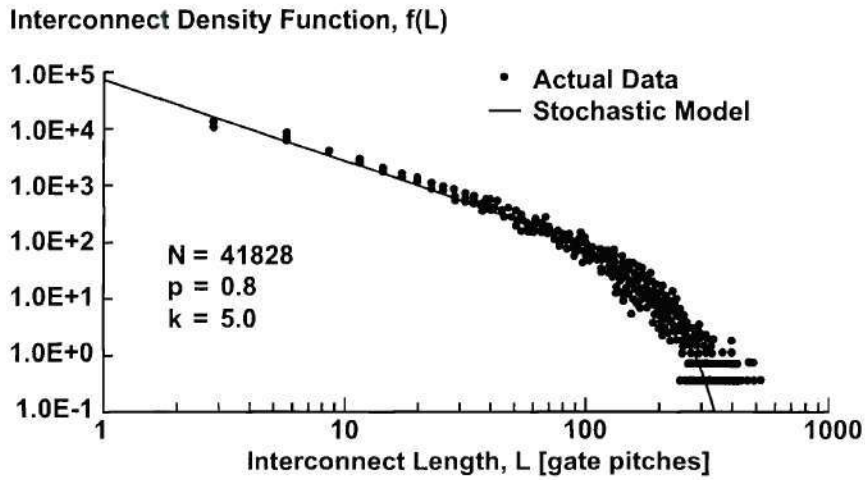


Figure 1.3: Interconnect density distribution. The vertical axis represents the density of interconnects of length L in units of number of interconnects per gate pitch and the horizontal axis represents interconnect length in gate pitches.

data from a commercial microprocessor in Figure 1.3 [19] and show good agreement compared to previous models.

The new *n-tier* multilevel interconnect design methodology described in Chapter 2 uses the stochastic distribution in 1.2-1.3 to determine the wire pitches for each metal layer in an optimized wiring network that minimizes area, cycle time, or number of levels. Unlike the algorithms in [17], this new algorithm is self-consistent and produces solutions that obey all the user-defined constraints on maximum permissible time delay for each metal layer. This methodology is well suited for technology prediction and for determining interconnect process parameters for future technology generations.

The wire pitch information for the *n-tier* design in Chapter 2 is derived using rc models for interconnect time delay. The rc delay, τ, of an interconnect of length L with no repeaters is given by [13], [21]

$$\tau = 1.1rcL^2 \tag{1.5}$$

where r and c are the interconnect resistance and capacitance per unit length. However, when optimal number of repeaters are inserted in this interconnect, the delay is given by [13]

$$\tau = 2.5\sqrt{rcR_oC_o}\ L \tag{1.6}$$

where R_o and C_o are the output resistance and input capacitance of a minimum-size repeater. Since repeater insertion improves the dependence of wire delay on wire length from a quadratic (1.5) to a linear relationship (1.6), repeaters can be an effective tool to reduce the stringent interconnect limits on future GSI systems. A novel repeater insertion methodology is developed in Chapter 3 that optimally inserts repeaters in a multilevel wiring network to decrease chip size, cycle time, number of metal levels or power dissipation, within the constraints of available repeater area.

1.3 Inductance models for high speed interconnect systems

As clock frequencies move into the tens of giga-hertz [6], inductive effects start influencing system performance. Traditional resistance-capacitance models (rc) are inadequate to model high speed interconnects in future GSI systems. Considerable effort has been expended to simulate and characterize the transmission line behaviour of high speed interconnects, for example, using waveform relaxation analysis [22], [23] and Asymptotic Waveform Evaluation (AWE) techniques [24]. Model order reduction has been used to derive and implement SPICE like models for interconnect circuit simulation [25], [26]. An algorithm requiring numerical computation of convolution integrals using analytical models for the impulse response is described in [27]. More

recently, research has focused on developing approximate models for the transient response of distributed *rlc* lines. A physical model for transient response of distributed *rlc* interconnects with open-circuit termination is rigorously derived in [28] and [29]. However, in order to accurately analyze interconnects that drive capacitive loads, such as high-speed clock lines or repeater-inserted signal lines, there is a need for models that account for the load capacitance. Ismail and Friedman describe curve-fitted empirical models in [30] and use them to extract expressions for time-delay of distributed *rlc* lines. These expressions lack physical insight because they are based on curve fitting. Cao *et al.* [31] have developed 2-pole second order waveform models that have less accuracy than a complete physical model. Banerjee and Mehrotra [32], [33], [34] have used second and fourth order Padé approximations for the transfer function, the solutions of which require numerical iterations to calculate line characteristics such as time delay and crosstalk.

In Chapter 4, the modeling effort in [28] and [29] is extended to describe transient expressions for distributed *rlc* lines with a load capacitance. A convergent series of modified Bessel functions is used to derive the time domain equivalent expression of the Laplace domain transfer function. The models are also extended to derive expressions for the crosstalk induced in two- and three-coupled distributed *rlc* lines. These analytical models are well suited for implementation in CAD algorithms for circuit simulation. They are portable and can be easily incorporated into system level simulation tools.

This new complete physical model is then used to develop compact expressions for time delay, crosstalk and repeater insertion in distributed *rlc* lines in Chapter 5. The use of repeaters to reduce crosstalk without increasing time delay is highlighted by formulating a new design plane that trades repeater area for smaller wiring cross

sections. The accuracy of the models developed in this chapter is better than the previous models when compared with HSPICE simulations. Compact models for delay, crosstalk and repeater insertion enable rapid, on-the-fly calculations that facilitate quick design evaluations. They are useful for system designers who need not depend on time consuming, full-wave simulators for delay and crosstalk estimations.

1.4 Inductance effects on multilevel architecture

New *rlc* models are used in Chapter 6 to analyze the effects of inductance on multilevel GSI interconnect architectures. The inclusion of inductance in the *n-tier* architecture forces the re-design of some of the metal layers with an increased spacing between the metal lines or the inclusion of co-planar shielding lines so that target crosstalk constraints are satisfied. The presence of repeaters tends to mitigate crosstalk due to the capacitive loading provided by the repeaters. Quantitatively, the effect of including inductance is observed by the slight increase in the number of metal layers. Specific attention is focussed on determining an optimum aspect ratio for the interconnects for high-speed and/or low-power designs which would maintain tolerable inductive crosstalk while ensuring high performance or low power. The assumption that an ideal return path is in close vicinity to the signal interconnects is changed to investigate the impact of non-ideal ground return paths on the interconnect system dimensions. Insertion of co-planar power/ground shielding lines reduces the crosstalk, but decreases wiring density, resulting in an increase in wire area.

1.5 Software tool

All these models have been integrated into a single software tool that can design an optimal *n-tier* multilevel interconnect network for any future GSI system. This software tool is called the Multilevel Interconnect Network Design Simulator (*MINDS*). Chapter 7 describes the main features and capabilities of *MINDS*. Some of the applications of *MINDS* are highlighted, and the results are verified with data from commercial products. The main program files in C language are provided in the appendix. It is envisaged that *MINDS* would be a useful and timely addition to the suite of device, circuit and system level simulation tools developed at Georgia Tech, such as Generic System Simulator (GENESYS) [35], Global Interconnect Distribution Estimator (GLIDE) [36], [37] and Rapid Interconnect Delay Evaluator (RIDE) [38], [39].

1.6 Summary

Finally, Chapter 8 draws conclusions and presents a summary of possible future work in this area. Some of the future work could include deriving a new wiring distribution for the recently announced 'X-architecture', and developing a multilevel network methodology which effectively utilizes diagonal wiring also to maximize performance and minimize wire length.

CHAPTER 2

n-tier Multilevel Interconnect Design Methodology

2.1 Introduction

Due to continuing reduction in the minimum on-chip feature size with technology scaling, interconnects are increasingly overshadowing transistors as the limiting factors in determining system performance and characteristics [9], [12]. Reverse-scaled multilevel wiring networks are extensively used in current VLSI systems to mitigate the impact of wiring on chip size and system performance. The most recent microprocessors have as many as six metal layers [40], while the ITRS has projected the use of up to 10 levels of wiring for the 45nm technology node in the year 2010 [6]. An optimal design of the reverse scaled metal layers is necessary to achieve higher performance at lower cost and lower power dissipation.

The optimal *n-tier* design architecture methodology is developed in this chapter. Section 2.2 provides a description of the models used for the design, and an explanation of the *n-tier* algorithm. Then, a case study is selected from the ITRS projections for the 100nm technology node [11]. The performance improvement achieved by using the *n-tier* architecture compared to a conventional design is quantified for this

case study in Section 2.3. A new aspect ratio design plane is defined in Section 2.4, which clearly elucidates the effects of varying the relative interconnect geometries on system performance. Optimal values for the different cross-sectional geometries are discussed. Finally, Section 2.5 discusses the assumptions used in the *n-tier* design methodology, and Section 2.6 provides a summary of this chapter.

2.2 Optimum *n-tier* design methodology

The key objective of the *n-tier* design architecture is to scale the cross-sectional dimensions of the interconnects on each metal level in a logic macrocell so that the available routing area is utilized completely, and the wire timing constraints are satisfied. This design philosophy automatically ensures that the number of metal levels required for a design is minimized for a specified target clock frequency and macrocell area.

Some of the key assumptions necessary for the *n-tier* methodology are as follows :

1. The interconnects in the system obey the stochastic wiring distribution (Rentian Distribution) described in (1.1)-(1.4) [19]. This distribution is applicable to a homogeneous square array of random logic gates. The common terminology for such a system is Manhattan routing, which implies that the gates are laid out in regular grid-like arrays, and the interconnects run in orthogonal x-y directions. The Rentian Distribution describes the interconnect distribution for signal interconnects only, and is not applicable for clock, power/ground and memory interconnects. Therefore, the *n-tier* design methodology is applicable only to intra-macrocell wiring in logic macrocells, which may constitute an ASIC System-on-a-chip (SOC). Moreover, all logic gates are assumed to be of the same type and placed in a homogeneous square array. For the case studies

used in the rest of this thesis, the logic macrocell is assumed to consist of 3-input NAND gates with a fan-out of 3.

2. It is assumed that the routing exclusively alternates in the x-y orthogonal directions for every metal layer. A three-dimensional view of the metal layers in a multilevel interconnect system is shown in Figure 2.1. The wire cross-sectional

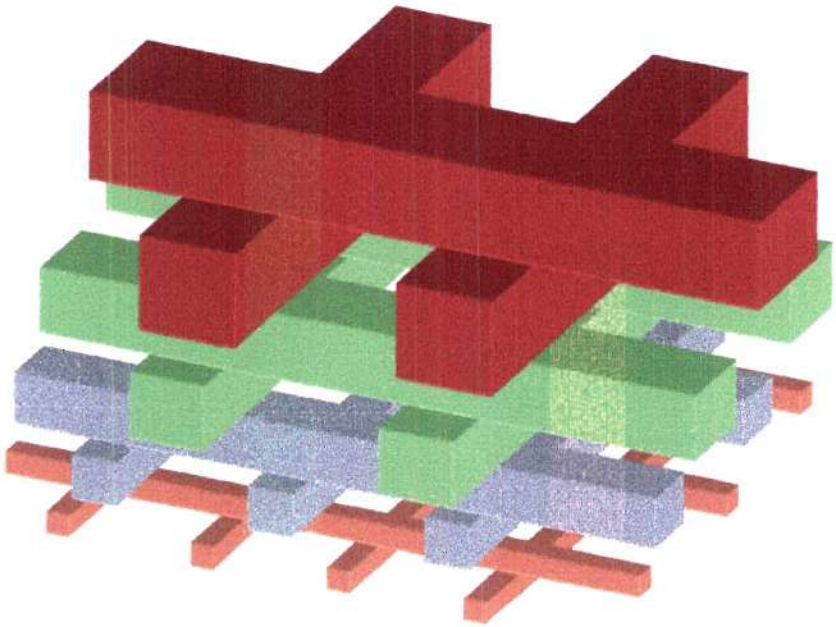


Figure 2.1: Three dimensional view of reverse scaled multiple metal layers.

dimensions are equal for every orthogonal pair of metal layers, which is called a *pair*. A collection of *pairs* of metal levels with the same cross-sectional dimensions is called a *tier*. The multilevel interconnect architecture consists of several such *tiers* (hence the name *n-tier* design). For example, Figure 2.2 shows the cross-section of a multilevel interconnect system that comprises of three tiers, where Tier 1 has two pairs of metal levels and Tiers 2 and 3 have one pair of metal levels each. Short wires are routed on the *tier* with the smallest pitch and successively longer wires are routed on tiers with progressively larger pitches.

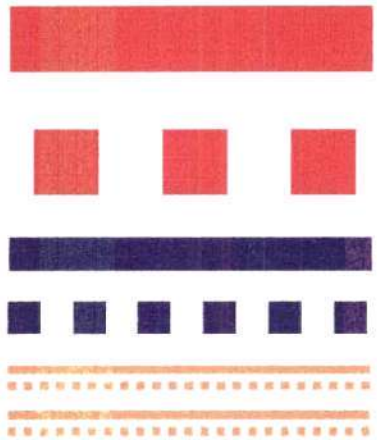


Figure 2.2: Cross-sectional view of an *n-tier* multilevel interconnect architecture.

3. The interconnect cross-sectional parameters are shown in Figure 2.3, where W, S and T are the wire metal width, spacing and thickness, respectively, and H is the height of the inter-level dielectric. Traditionally, the aspect ratio is defined as the ratio of the metal thickness to metal width (=T/W). However, in this thesis, the term aspect ratio will be used generically for all the interconnect cross-sectional ratios namely, T/W, S/W and H/W. For the *n-tier* multilevel interconnect design methodology, it is assumed that the aspect ratios are constant for all interconnects of a particular tier.

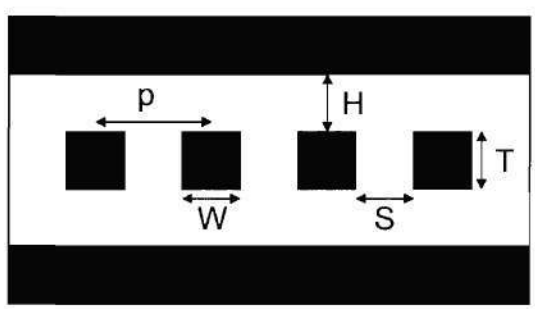


Figure 2.3: Cross-sectional view of a multilevel interconnect architecture, where W, S and T are the wire width, spacing and thickness, H is the height of the inter-level dielectric, and $p = W + S$ is the wire pitch, respectively.

4. It is assumed that the maximum interconnect time delay for each tier is specified as a fraction β of the clock period. The tier pitch is chosen such that the time delay of the longest interconnect on that tier equals the specified maximum delay constraint. For example, the maximum permissible time delay for the case study discussed later is assumed to be a fraction $\beta = 0.25$ of the clock period for the lowest tier and $\beta = 0.9$ of the clock period for all other tiers.

5. The wiring efficiency factor, e_w, is the fraction of the total area of any metal layer that is used for available routing [13],[16]. It is also assumed that the wiring efficiency factor is specified for each tier. The wiring efficiency is dependent on the router efficiency, power, ground and clock line blockage and via blockage, as will be discussed in Section 2.5. For the example case study, e_w is assumed to be 40% for all the levels.

2.2.1 Wiring demand models

The Rentian Distribution [19] is used to estimate the number and length of interconnects based on system parameters such as Rent's constants, macrocell area, clock frequency and number of gates. The total length of interconnects on any tier is estimated by using a supply-demand equation. The range of interconnect lengths on the tth tier is calculated by equating the area available for wiring, A_{av}, to the area that is required for wiring, A_{req}:

$$A_{av} = A_{req}. \tag{2.1}$$

The area available for routing is the total area of all the layers in the tth tier times the wiring efficiency factor. The area required for routing is the total length of

interconnects times the wire pitch i.e.

$$e_w n_t A_m = \chi p_t \sqrt{\frac{A_m}{N_g}} \int_{L_{t-1}}^{L_t} li(l)dl \tag{2.2}$$

where n_t is the number of metal levels in the tth tier, A_m is the macrocell area, χ converts point-to-point interconnect length to wiring net length [19], p_t, L_{t-1} and L_t are the wire pitch, shortest and longest interconnect length on the tth tier in physical units and gate pitches, respectively, l is the interconnect length in gate pitches, N_g is the number of logic gates in the system and $i(l)$ is the interconnect density function defined in (1.1)-(1.4) [19]. The only unknowns in (2.2) are p_t, L_{t-1} and L_t. Since the *n-tier* design starts by designing the lowest tier first, tier $t-1$ would have been designed before tier t. Hence, L_{t-1} is also a known quantity. (For the first tier $t = 1$, L_0 is by definition equal to 1 gate pitch.) The pitch, p_t, is calculated using rc models for interconnect time delay.

2.2.2 Time delay models

From (1.5) the rc delay for the longest interconnect on tier t is approximated as:

$$\tau = 1.1rc\frac{A_m}{N_g}L_t^2. \tag{2.3}$$

For cross-sectional geometry shown in Figure 2.3, the interconnect line resistance per unit length can be written as

$$r = \frac{\rho}{WT} \tag{2.4}$$

where ρ is the resistivity of the metal (for bulk Al, ρ=2.86×10$^{-6}\Omega$-cm; for bulk Cu, ρ=1.67×10$^{-6}\Omega$-cm). The orthogonal metal levels above and below any level are assumed to behave as approximate ground planes. Therefore, the capacitance per unit length is calculated using a 3-conductor 2-ground plane model as in Figure 2.4.

Using the models from [41], the mutual (c_m) and ground (c_g) capacitances can be written as

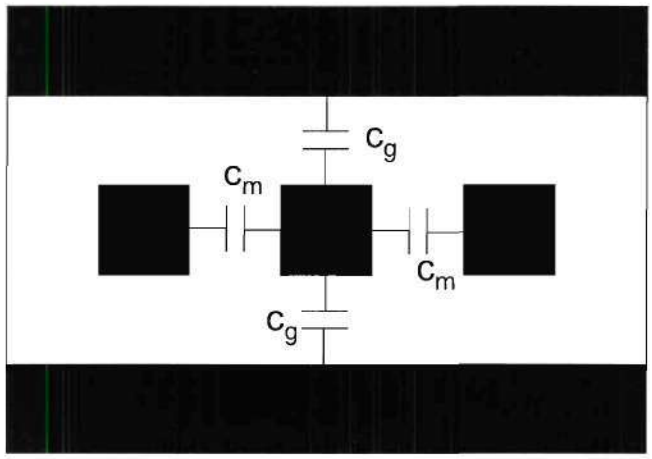


Figure 2.4: 3-conductor 2-ground plane approximation: c_m is the line-to-line or mutual capacitance and c_g is the line-to-ground or ground capacitance.

$$\begin{aligned}\frac{c_m}{\varepsilon_r\varepsilon_o} &= \frac{T}{S}\left(1 - 1.897e^{\frac{-H}{0.31S}+\frac{-T}{2.474S}} + 1.302e^{\frac{-H}{0.082S}} - 0.1292e^{\frac{-T}{1.326S}}\right) \\ &\quad +1.722\left(1 - 0.6548e^{\frac{-W}{0.3477H}}\right)e^{\frac{-S}{0.651H}} \qquad (2.5)\end{aligned}$$

$$\frac{c_g}{\varepsilon_r\varepsilon_o} = \frac{W}{H} + 1.086\left(1 + 0.685e^{\frac{-T}{1.343S}} - 0.9964e^{\frac{-S}{1.421H}}\right)\left(\frac{S}{S+2H}\right)^{0.0476}\left(\frac{T}{H}\right)^{0.337} \qquad (2.6)$$

Since each conductor can undergo any one of three transitions (switch to high polarity, switch to low polarity or remain unchanged), there are 27 possible combinations in which the three lines can change polarity. The worst case delay, when both the neighbouring lines are switching polarity in a direction opposite to that of the center line, occurs with a probability of 2/27. The less pessimistic and more probable scenario, when the polarity of the two neighbours remains unchanged while the center line switches, has a probability of occurence of 8/27. For this second case, the capacitance per unit length is given by

$$c_{int} = 2c_m + 2c_g. \qquad (2.7)$$

For simplicity, it is assumed that all the interconnect aspect ratios are equal to 1 i.e. $W = T = S = H = p_t/2$. Susbstituting in (2.4) and (2.7), the resistance and capacitance per unit length can be written as

$$r = \frac{4\rho}{p_t^2} \tag{2.8}$$

and

$$c = 6.2\epsilon_o\epsilon_r. \tag{2.9}$$

Hence, the time delay in (2.3) can be written as

$$\tau = 4\frac{1.1\rho\varepsilon_r\varepsilon_o 6.2}{p_t^2}\frac{A_m}{N_g}L_t^2 \tag{2.10}$$

The maximum permissible delay of the longest interconnect on each tier is assumed to be equal to a fraction β of the clock period ($=1/f_c$) i.e.

$$\tau = \frac{\beta}{f_c} \tag{2.11}$$

The wiring pitch of the local tier is equal to twice the minimum feature size ($p_t = 2F$); for all non-local tiers (i.e. $p_t > 2F$), the pitch is obtained by equating the time delay of the longest interconnect to the maximum permissible delay. Therefore equating (2.10) and (2.11), the pitch of the tth tier can be written as

$$p_t = 2\sqrt{\frac{1.1\rho\varepsilon_r\varepsilon_o 6.2 f_c}{\beta}}\sqrt{\frac{A_m}{N_g}}L_t. \tag{2.12}$$

2.2.3 *n-tier* design algorithm

For a given value of A_m and f_c, the n-tier design methodology starts with the lowest tier and moves upwards, filling each tier with interconnects. To maximize wire density, this algorithm first attempts to use minimum width interconnects on the local tier. Solving (2.2), using $p_t = 2F$, gives the longest interconnect length on the

local tier. For non-local tiers, (2.2) and (2.12) are solved simultaneously, increasing the pitch of each tier so that the longest interconnect on that tier satisfies the timing constraint (delay $\tau = \beta/f_c$). Thus wire density is maximized and results in a minimum number of metal levels for the specified area and clock frequency. When the twice-die-edge-long interconnect has been included on a tier, the algorithm stops, and the resulting number of metal levels is the minimum number of levels required to route all interconnects in the macrocell. This procedure is repeated for different values of the macrocell area and clock frequency, to determine various optimizations described in the following sections. A flowchart showing the complete *n-tier* design methodology is shown in Figure 2.5.

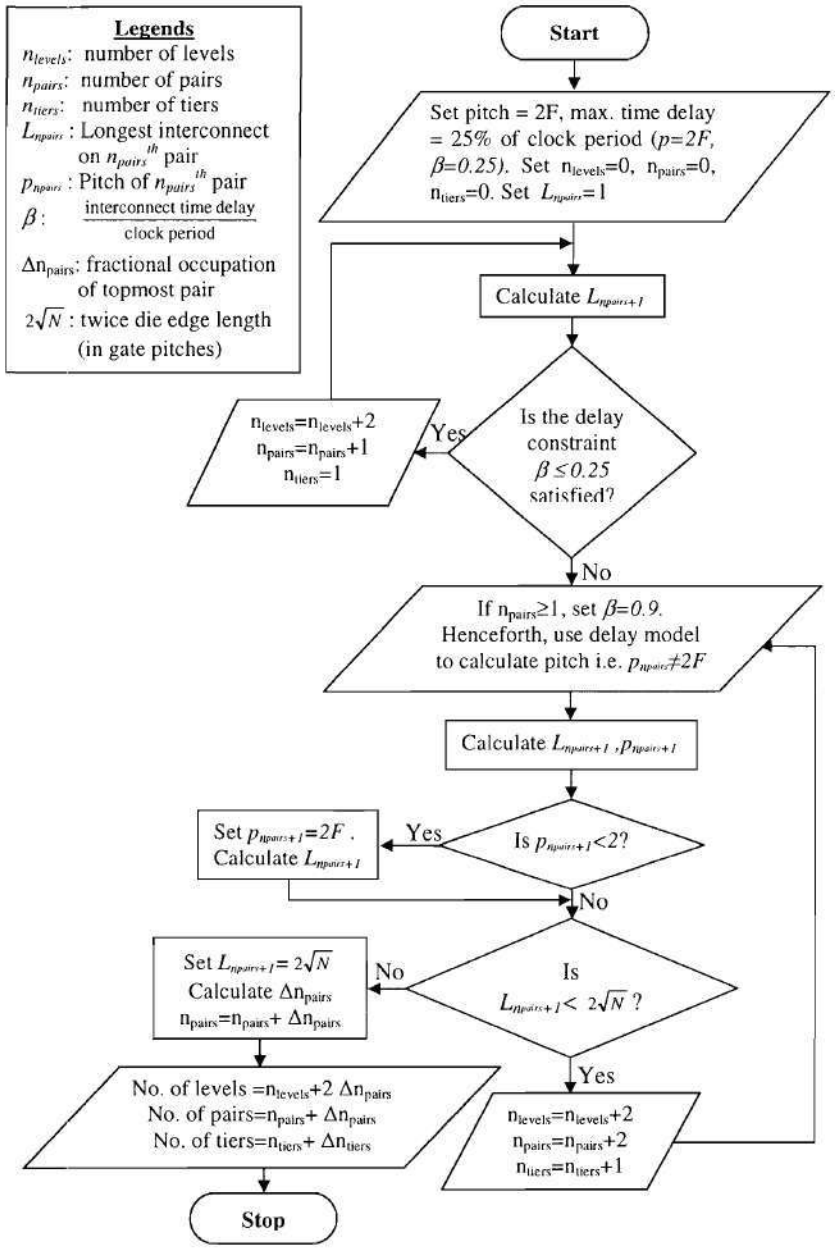


Figure 2.5: Flowchart for *n-tier* design methodology.

2.3 Performance enhancement using optimized *n-tier* architectures

The performance enhancement achieved by an optimized *n-tier* architecture is demonstrated for an ASIC macrocell case study using 100nm technology projections from the International Technology Roadmap for Semiconductors (ITRS) [11]. In this case study, the macrocell has 11.3M logic gates (assuming the use of 3-input 6-transistor static CMOS NAND gates, this corresponds to approximately 68M transistors), a low permittivity dielectric ($\epsilon_r = 2$) and copper interconnects. For the system to meet the ITRS projections of a 2GHz clock frequency and a maximum of 8 metal levels for the 100 nm technology generation, the maximum size of the macrocell is 11.3M logic gates. The following values were assumed for the other parameters in (2.2)-(2.12) : Rent's exponent $p = 0.6$, Rent's coefficient $k = 4$ [42], $\alpha = 0.75$ ($\alpha =$ fanout/(fanout+1); for 3-input NAND gates, fanout=3), $\chi = 0.667$ [19] and $e_w = 0.4$ [16]. Shorter interconnects (first tier) are more likely to be constituents of critical paths and hence are assigned a smaller β (=0.25), so that the critical path gates can have a larger (remaining) time delay. A larger β (=0.9) is assigned to longer interconnects because they would most likely be used for cross-chip communication only (requiring more delay). A summary of the values assumed for the various parameters is provided in Table 2.1.

Three optimizations are defined in this section that minimize the number of metal levels, the macrocell area, or the cycle time. To demonstrate the advantages of an *n-tier* architecture, a comparison is made with conventional designs where the wiring pitch is heuristically scaled from one tier to the next, according to [16].

Table 2.1: Summary of various parameters used in the ASIC macrocell case study based on the ITRS.

Parameter	Symbol	Value
ITRS Technology generation		100nm
Minimum feature size	F	100nm
Number of metal layers	n	8
Clock frequency	f_c	2GHz
ILD dielectric constant	ϵ_r	2.0
Interconnect resisitivity	ρ (Cu)	$1.673\times10^{-6}\Omega$-cm
Number of logic gates	N_g	11.3Million
Type of logic gates		3-input NAND
Average fanout	f.o.	3
Rent's exponent	p	0.6
Rent's coefficient	k	4
Wiring efficiency	e_w	40%
Local wire delay	β_{local}	25% of clock period
Non-local wire delay	β_{global}	90% of clock period

2.3.1 2-tier design

2-tier designs comprised of a thin local tier and a thicker global tier. If possible, the first tier is designed with pitch equal to twice the minimum feature size. Otherwise, its pitch is calculated by setting its maximum time delay (i.e. time delay of the longest interconnect on that tier) equal to 25% of the clock period. The global pitch is calculated by making the time delay of a die-edge-length interconnect equal to 90% of the clock period. The plot of number of metal levels vs. macrocell area for various frequencies for the 2-tier design of the case study is shown in Figure 2.6. Whenever the longest interconnect on the first (local) tier cannot meet the timing constraint, some of the interconnects on that (local) pair are moved to the second (global) tier with a greater pitch, which consequently increases the number of levels. The points of abrupt upward transitions in Figure 2.6 illustrate this point to the extreme and correspond to the loss of a complete local pair.

The curves in Figure 2.6 saturate when all pairs are non-local (i.e. $p_n > 2F$). As

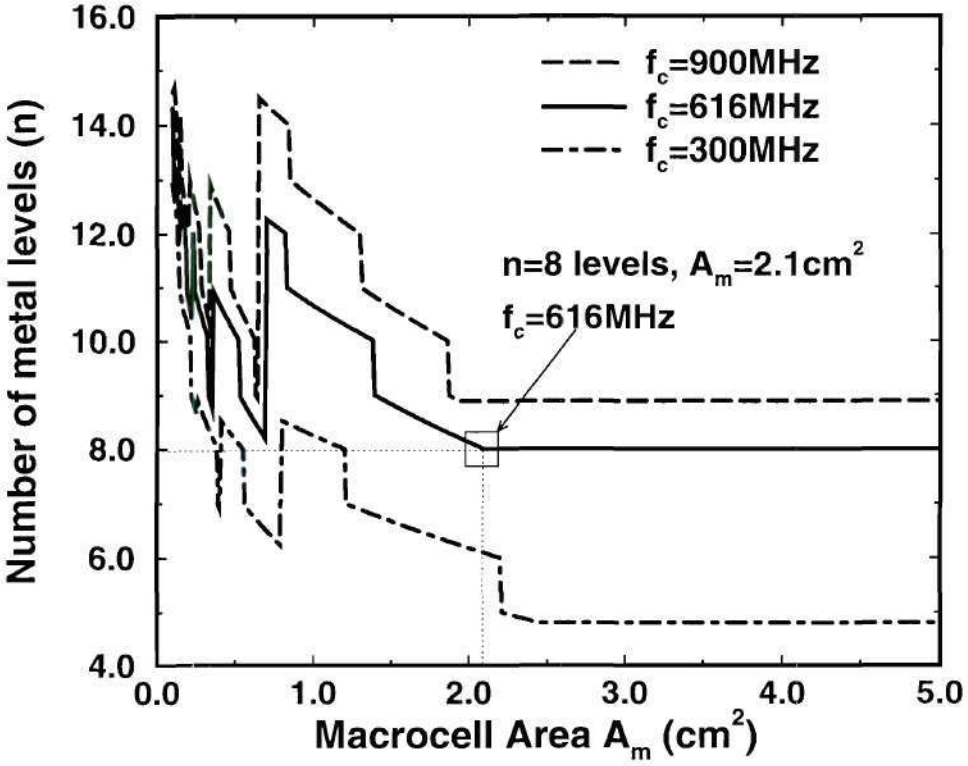


Figure 2.6: Number of levels vs. macrocell area for 2-tier design

seen by combining (2.2) and (2.12), for a non-local tier, the maximum interconnect length does not depend on the macro-cell area A_m. Therefore, when all pairs become a part of the global tier, the number of levels does not change with increase in macrocell area because the interconnect pitch and length also scale up in the same ratio. As seen from Figure 2.6, the maximum operational frequency for the 2-tier design for 8 metal levels is 616 MHz. This corresponds to a macrocell area of 2.10cm^2.

2.3.2 Conventional multilevel network designs

In conventional multilevel interconnect architectures, the metal layers are usually designed such that the wiring pitch of each layer is a multiple of the width of the thinnest metal level. The first column of Table 2.2 shows several such conventional

designs, with the first entry being a 2-tier design having 4 local levels with pitch = 2F and 4 global levels with pitch = 8F. The operational frequency is determined by setting the maximum time delay among all the interconnects in the system equal to 90% of the clock period (25% for the interconnects on the local tier). In 3-tier designs, there is an intermediate semi-global tier. The next two entries in Table 2.2 are examples of 3-tier architectures: one with 4 local, 2 semi-global and 2 global levels, and the other with 2 local, 4 semi-global and 2 global levels. In both cases, the semi-global pitch is equal to 4F while the local and global pitches remain unchanged. Sai-Halasz [16] recommends a multilevel network where the pitch is doubled for every successive pair of metal levels. Such a 4-tier network with wiring pitches equal to 2F, 4F, 8F and 16F for each pair of metal levels is also shown in Table 2.2. The process described in [43] also uses a similar wire scaling pattern, and therefore, this design would henceforth be called the conventional baseline design. Using this design, the macrocell in the case study has an area of 1.45cm^2 (baseline area) with 8 metal levels and can operate at 1.56 GHz (baseline clock frequency). Such conventional designs are not optimal designs because the time delay of the longest interconnect on some tiers may be less than the maximum permissible delay (i.e. $\beta < 0.9$). This creates opportunity for decreasing the metal pitch or macrocell area, which could be used to reduce the number of metal levels or increase the operational clock frequency. This is the main motivation for the optimized *n-tier* design methodology.

2.3.3 Optimized *n-tier* design

The *n-tier* multilevel interconnect architecture is designed such that the wire pitch of each tier is dependent on the time-delay of the longest interconnect routed in that tier. This ensures that the time delay constraints are exactly satisfied and the wiring

Table 2.2: Comparison of conventional designs with new optimized *n-tier* designs.

	Conventional Designs	Optimized *n-tier* design		
		Min. area optimization	Max. frequency optimization	Min. # levels optimization
	2-Tier (2F,2F,8F,8F)			
$A_m =$	0.82cm^2	0.37cm^2	0.82cm^2	0.82cm^2
$f_c =$	217MHz	217MHz	1.47GHz	217MHz
$n =$	8 levels	8 levels	8 levels	$4.98 \approx 6$ levels
	3-Tier (2F,2F,4F,8F)			
$A_m =$	0.68cm^2	0.47cm^2	0.68cm^2	0.68cm^2
$f_c =$	672MHz	672MHz	1.16GHz	672MHz
$n =$	8 levels	8 levels	8 levels	$6.57 \approx 8$ levels
	3-Tier (2F,4F,4F,8F)			
$A_m =$	1.01cm^2	0.48cm^2	1.01cm^2	1.01cm^2
$f_c =$	710MHz	710MHz	1.6GHz	710MHz
$n =$	8 levels	8 levels	8 levels	$6.24 \approx 8$ levels
	4-Tier (2F,4F,8F,16F)			
$A_m =$	1.45cm^2	0.98cm^2	1.45cm^2	1.45cm^2
$f_c =$	1.56GHz	1.56GHz	1.86GHz	1.56GHz
$n =$	8 levels	8 levels	8 levels	$6.77 \approx 8$ levels

density is maximized. Figure 2.7 plots the number of metal levels versus macrocell area of the *n-tier* design for clock frequencies of 1, 1.56, 1.86, 2 and 3.99 GHz (the optimized *n-tier* designs have clock frequencies of 1.56, 1.86 and 2 GHz). The curves in Figure 2.7 saturate when the minimum feature size width and spacing can no longer be used for any tier (i.e. $p_{n_{tier}} > 2F$ for all tiers). This is because, when (2.12) is substituted in (2.2), the longest interconnect length $L_{n_{tier}}$ of the lowest tier also becomes independent of macrocell area, A_m. Therefore, the number of metal levels remains constant because the interconnect pitch and length increase at the same rate as the macrocell area is increased. The improvement in system design using the *n-tier* methodology is quantified in the following three optimizations and the results are summarized in the last three columns of Table 2.2.

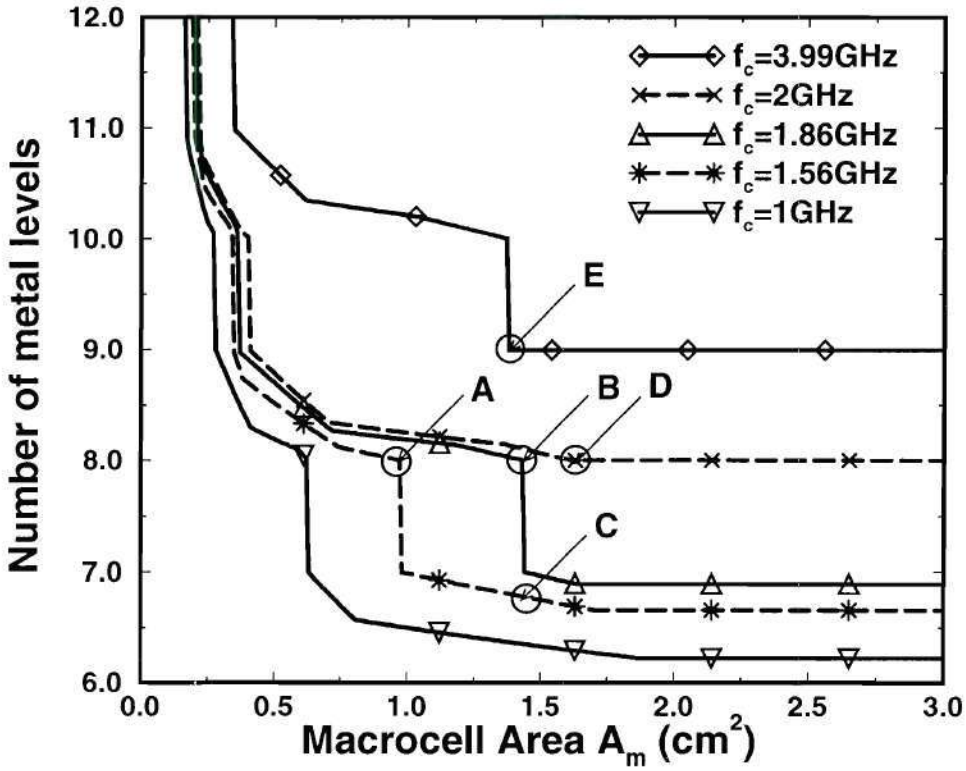


Figure 2.7: Number of levels versus macrocell area for the *n-tier* design of a 68M transistor macrocell.

2.3.4 Minimum macrocell area

For the same number of metal levels and clock frequency, the optimized *n-tier* architecture reduces the macrocell area by decreasing the wiring pitch resulting in greater packing density for the interconnects. The minimum macrocell area optimization is shown by point A on the curve in Figure 2.7 which plots the number of metal levels against macrocell area for the *n-tier* design, at the baseline clock frequency of 1.56GHz. For a maximum of n=8 metal levels, the conventional baseline design requires a macrocell area of 1.45cm^2 whereas the *n-tier* design requires a macrocell area of only 0.98cm^2, which is a 32% reduction in cell size.

2.3.5 Maximum clock frequency

For the same number of metal levels and macrocell area, the optimized *n-tier* architecture improves clock frequency by increasing the wiring pitch, so that the area wasted by the conventional baseline design is utilized. The maximum clock frequency optimization using the *n-tier* design is shown by point B on the curve in Figure 2.7, which plots the number of levels versus macrocell area for the *n-tier* design at a clock frequency of 1.86GHz. This corresponds to a 16% reduction in the cycle time over the baseline clock frequency of 1.56GHz for the conventional baseline design. This is the maximum clock frequency achievable using the *n-tier* design for n=8 metal levels and A_m=1.45cm^2. To achieve higher clock frequencies, the curves will shift upward and to the right, as seen in Figure 2.7, necessitating a greater number of metal levels or larger area. For example, the maximum clock frequency that can be achieved using the *n-tier* design with 8 metal layers (and no constraint on the area) is 2GHz, as shown by point D in Figure 2.7. This design point (with A_m = 1.62cm^2, f_c=2GHz and n=8 metal layers) is called the *n-tier baseline design* point. The performance improvement achieved by the addition of just one metal layer is demonstrated by design point E in Figure 2.7, which shows that the wire limited clock frequency doubles to almost 4GHz. ("Liquid routing" permits an odd number of metal levels because there is no preferred direction for the metal lines on any particular layer [44]. This concept is discussed in more detail in Section 8.3).

2.3.6 Minimum number of metal levels

The optimized *n-tier* architecture reduces the number of metal levels, for the same area and clock frequency, by decreasing the wiring pitch resulting in greater packing density for the interconnects. The reduction in the number of metal levels using the

n-tier design is shown by point C on the curve in Figure 2.7 which plots the number of levels versus macrocell area for the *n-tier* design at the baseline clock frequency of 1.56 GHz. Since the levels are grouped in x-y orthogonal pairs, the number of metal levels should be rounded off to the next higher even integer. For the baseline macrocell area of A_m=1.45cm^2, the *n-tier* design requires 6.77 metal levels, which when rounded off to 8 metal levels, is the same as for the conventional baseline design. Thus, for this example, although there is no actual reduction in the number of metal levels, 62% of the wiring tracks on the topmost tier have been freed and can be used to accommodate additional power, ground and clock wiring resources. However, if CAD algorithms incorporate "liquid" routing [44], then the optimized design would only need 7 metal layers, thus eliminating an entire metal level.

Detailed comparison between the conventional baseline design and the optimized *n-tier* designs is shown in Table 2.3. The cross sections of the multilevel interconnect architecture for the conventional design and the optimized *n-tier* designs is shown in Figure 2.8.

Table 2.3: Interconnect parameters for the various design points

Tier # (n)	number of levels	L_{n-1} (Gate pitches)	L_{n-1} (cms)	L_n (Gate pitches)	L_n (cms)	p_n (μm)
Conventional baseline architecture : A_m=1.45cm^2, f_c=1.56GHz and n=8 levels.						
Tier 4	2	1884	0.67	6723	2.41	1.6
Tier 3	2	818	0.29	1884	0.67	0.8
Tier 2	2	174	0.06	818	0.29	0.4
Tier 1	2	1	0.0004	174	0.06	0.2
(*n-tier*) Minimum area : A_m=0.98cm^2, f_c=1.56GHz and n=8 levels.						
Tier 4	2	1760	0.52	6723	1.98	1.17
Tier 3	2	735	0.22	1760	0.52	0.61
Tier 2	2	93	0.03	735	0.22	0.26
Tier 1	2	1	0.0003	93	0.03	0.2
(*n-tier*) Maximum frequency : A_m=1.45cm^2, f_c=1.86GHz and n=8 levels.						
Tier 4	2	1847	0.66	6723	2.41	1.56
Tier 3	2	846	0.30	1847	0.66	0.85
Tier 2	2	174	0.06	846	0.30	0.39
Tier 1	2	1	0.0004	174	0.06	0.2
(*n-tier*) Minimum # of levels : A_m=1.45cm^2, f_c=1.56GHz and n=6.77 $\approx$ 8 levels.						
Tier 4	0.77$\approx$2	2000	0.72	6723	2.41	1.42
Tier 3	2	893	0.32	2000	0.72	0.85
Tier 2	2	174	0.06	893	0.32	0.38
Tier 1	2	1	0.0004	174	0.06	0.2

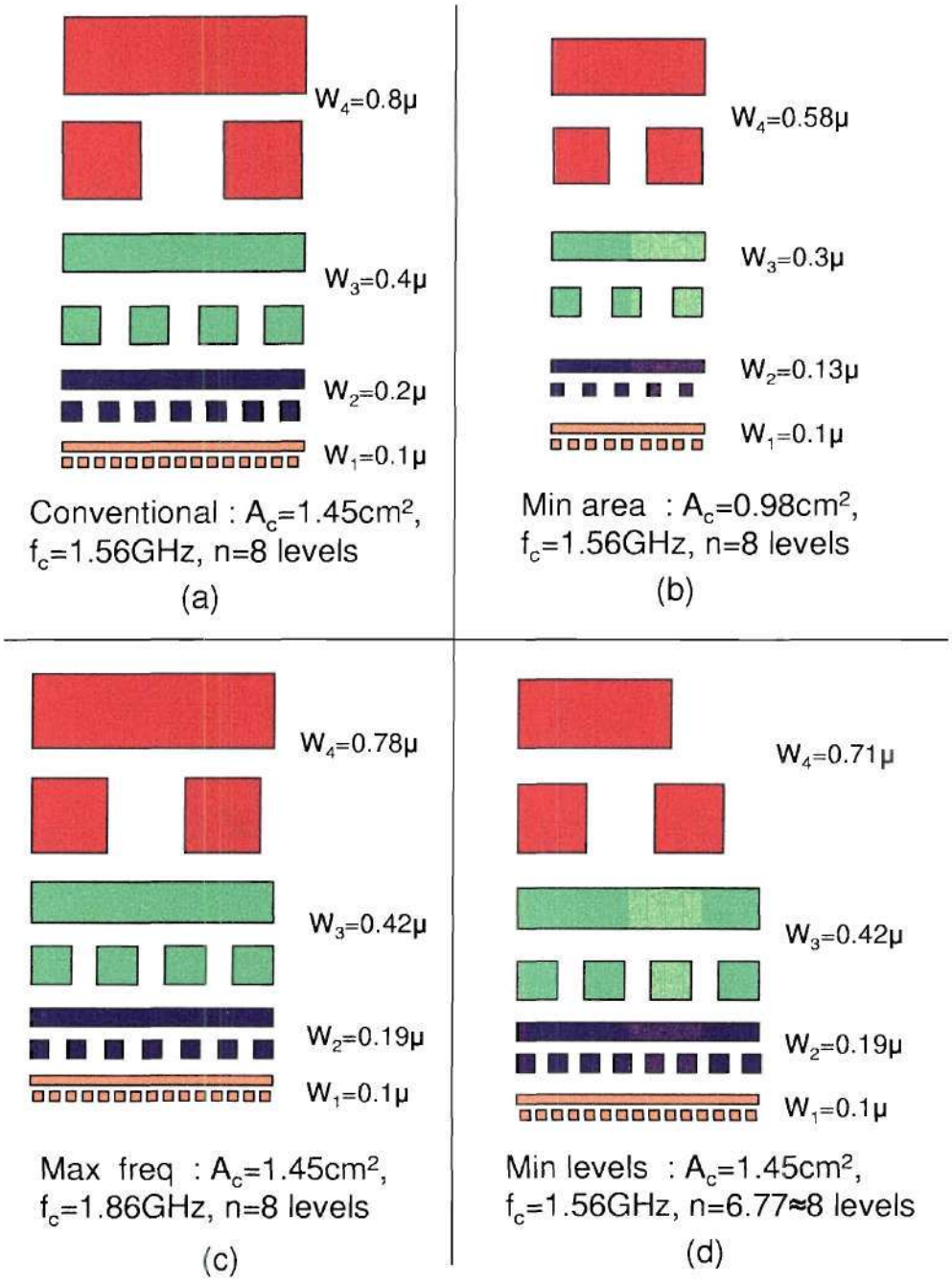


Figure 2.8: Cross section of multilevel interconnect architecture for (a) conventional doubling of width scaling, and optimal *n-tier* designs for (b) minimum area, (c) maximum frequency, and (d) minimum number of metal levels.

Thus, the *n-tier* design methodology can be used to determine the interconnect pitches for different tiers so that optimal performance is extracted from the system for the available wiring resources. Additional improvements in performance can be achieved through the insertion of repeaters. Chapter 3 describes and quantifies the impact of an optimum repeater insertion methodology for the *n-tier* design process.

2.4 Optimum aspect ratio

In the previous sections, the interconnect dimensions are assumed to have a unity aspect ratio (i.e. $W = S = T = H$). However, in order to increase wiring density without increasing line resistance, current commercial microprocessor designs employ interconnects whose thickness is greater than their width. Since such a design increases the mutual coupling between adjacent parallel interconnects, the wires need to be spaced further apart to keep the crosstalk within limits. The height of the inter-level dielectric (ILD) determines the capacitance between metal layers. A larger ILD height would decrease the *rc* delay by decreasing capacitance to orthogonal layers, but it would increase the capacitive crosstalk by increasing the contribution of the mutual capacitance to the total wiring capacitance. In order to balance crosstalk constraints with high wire density demands, an optimal value for the various interconnect aspect ratios needs to be calculated that takes into account the fabrication constraints and reliability problems, which limit the range of feasible aspect ratios.

2.4.1 Aspect ratio design plane

Consider the *n-tier baseline design* for the ASIC macrocell case study in Section 2.3.5, with 11.3M logic gates, area=1.62cm^2, f_c=2GHz and 8 metal levels. These are the optimal system characteristics when all the cross-sectional dimensions are

assumed equal. The objective of this section is to investigate different combinations of aspect ratios (S/W, T/W and H/W) and select the optimum set of aspect ratios that maximize system clock frequency, or minimize macrocell area, number of levels or power dissipation. It is assumed that S/W, T/W and H/W are the same for *all* the multilevel interconnect tiers. This assumption is relaxed later in Section 6.6.3 of Chapter 6, when the aspect ratios are optimized for each tier.

In order to avoid drawing three dimensional graphs, a fixed value of the ILD height/wire width ratio (H/W) is assumed; then curves are drawn with metal thickness/wire width (T/W) on the y-axis and metal spacing/wire width (S/W) on the x-axis. Any combination of the three aspect ratios (S/W, T/W and H/W) can be used for the system design if it is technologically achievable. However, the feasible aspect ratios are only those which satisfy the following two system level constraints, which somewhat oppose each other :

1. Number of levels constraint : The number of levels required to interconnect the system for any set of values for the aspect ratios is estimated using the *n-tier* methodology. The aspect ratios should be such that all the interconnects of the system can be accomodated within the specified number of metal layers. For the ITRS case study, the maximum permissible number of metal layers is 8. This limits the width and the spacing between the interconnects and is directly related to the time delay constraints imposed on the interconnects of each layer. Hence, this constraint could also be called the delay constraint.

2. Crosstalk constraint : The capacitive coupling between adjacent interconnects induces undesirable crosstalk noise, causing signal reliability problems. The worst case *rc* crosstalk is given by [21] :

$$\text{Peak Crosstalk} = \frac{1}{2}\frac{c_m}{c_m + c_g} \tag{2.13}$$

where c_m and c_g are the mutual and ground capacitances per unit length, defined in (2.5) and (2.6), respectively. As a first approximation, the length dependence of the rc crosstalk, described in [45], is ignored. The capacitive coupling between adjacent interconnects is decreased by increasing the spacing between the interconnects. The crosstalk is also reduced by increasing the ground capacitance (by decreasing the height of the ILD); however, this is at the expense of increased rc wire delay. The usable aspect ratios are those for which the worst case rc crosstalk is no more than the specified maximum. For the ASIC case study, the maximum permissible crosstalk is assumed to be equal to 20% of the supply voltage.

Any set of values for the three ratios would be a feasible solution if and only if both these constraints are not violated.

The curve in Figure 2.9 is a locus of points for which the number of metal layers of the system is equal to 8. The region to the right of the curve either employs larger spacing for the same thickness, or thinner wires for the same spacing, thereby leading to an increase in the number of metal layers. Therefore this shaded area is a forbidden region because it results in a number of metal layers which is greater than 8. The area to the left of the curve is the allowable region because it results in less than 8 metal layers for the system.

The curve in Figure 2.10 is a locus of points for which the rc crosstalk is equal to 20%. The region to the left of the curve either has smaller spacing for the same wire thickness or thicker wires for the same spacing, which implies larger mutual capacitance and crosstalk beyond $0.2V_{dd}$. This shaded area, therefore, is also a forbidden region. The area to the right of the curve is the allowable region because it results in a crosstalk less than $0.2V_{dd}$.

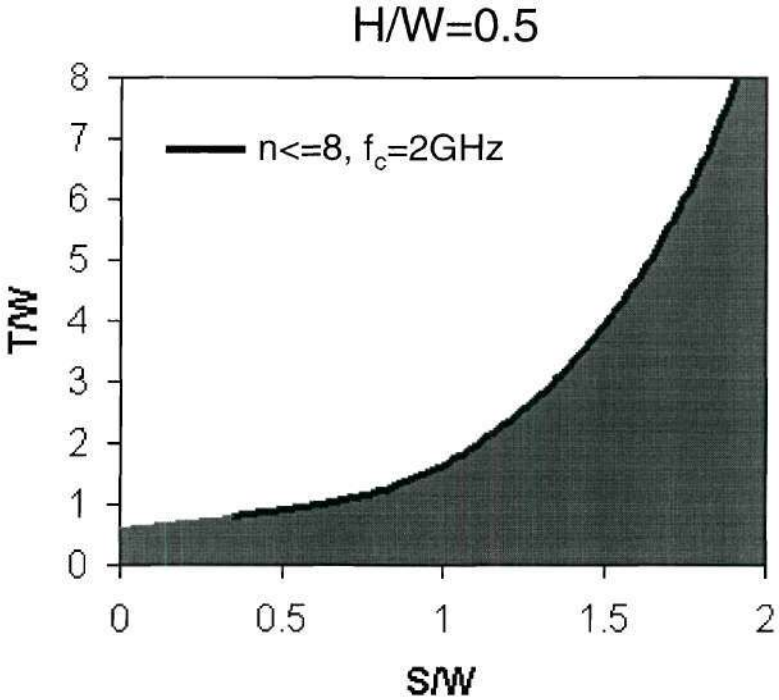


Figure 2.9: Forbidden region for number of levels constraint.

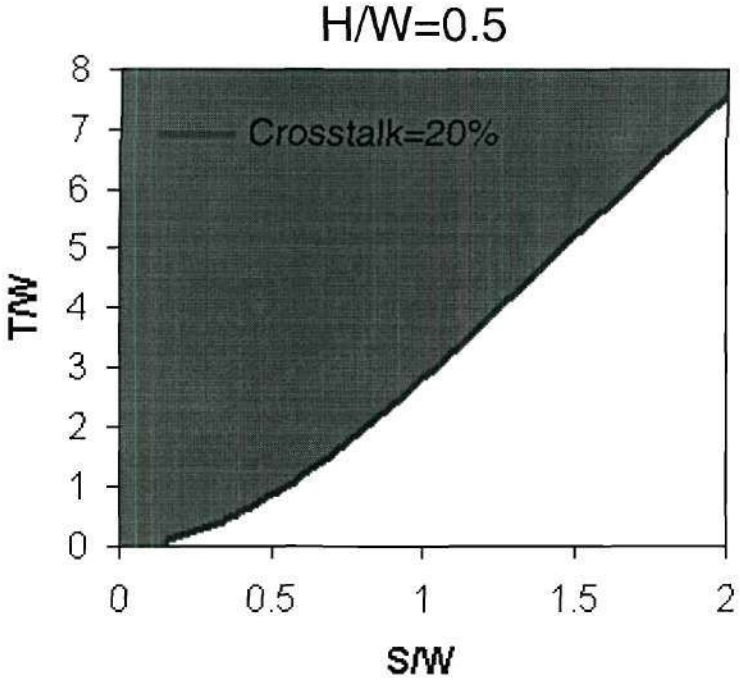


Figure 2.10: Forbidden region for crosstalk constraint.

The curves in Figures 2.9 and 2.10 are combined in Figure 2.11. The shaded area of Figure 2.11 is the forbidden region : the aspect ratios in that region cannot be used because they violate either the number of levels constraint or the crosstalk constraint. The unshaded area is the allowable region of aspect ratios which satisfies both the constraints of $n \leq 8$ metal layers and crosstalk $\leq 0.2V_{dd}$.

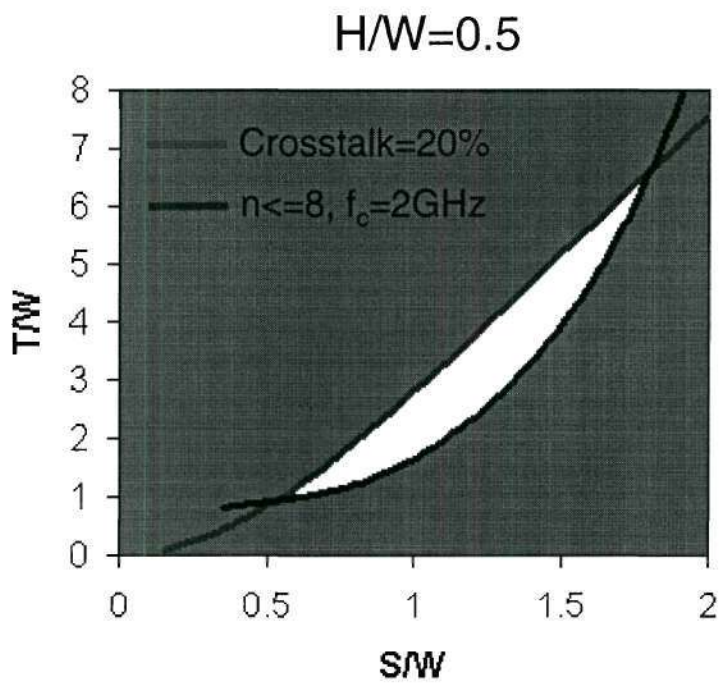


Figure 2.11: Unshaded area is the allowable region that satisfies the number of levels constraint and the crosstalk constraint.

Figure 2.12 shows the forbidden and allowable regions for $H/W = 1.0$. The allowable region has shrunk to a very narrow sliver of optimization opportunity. This is because the *n-tier* design case study is already optimized for a unity aspect ratio. Moreover, the crosstalk is equal to $0.2V_{dd}$ when all the aspect ratios are equal to unity. Therefore, there is not much room to improve on this design when the oxide height aspect ratio is equal to 1.0. Figure 2.13 shows that when H/W is increased further to

1.5, the feasible region disappears altogether. This means that no combination of the interconnect aspect ratios is adequate to satisfy both the number of levels constraint and the crosstalk constraint.

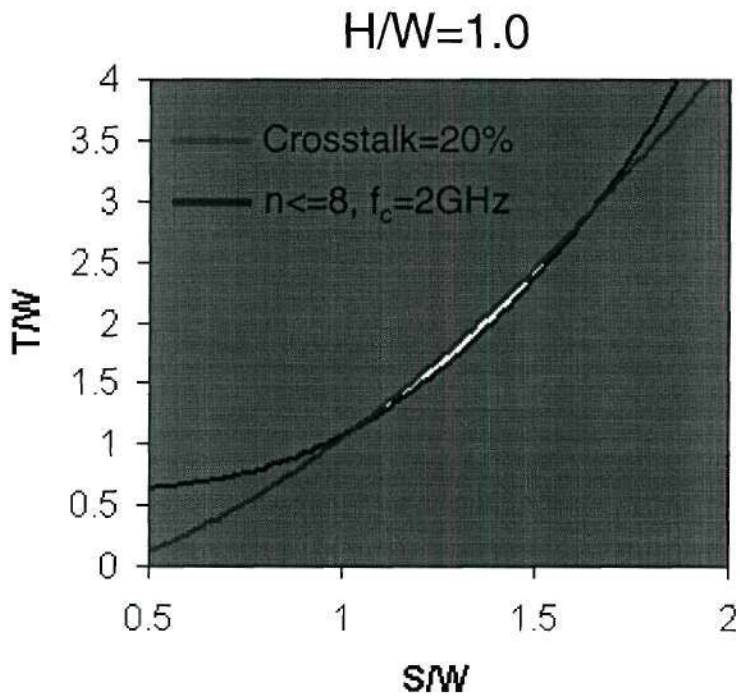


Figure 2.12: The feasible region shrinks for $H/W = 1.0$.

When the ILD height is increased, the ground capacitance decreases, which increases the effect of the mutual capacitive coupling. Therefore to maintain constant crosstalk, the spacing has to increase for the same wire thickness. This implies that the constant crosstalk locus will shift to the right. At the same time, the rc product decreases, because the capacitance is decreasing. This means that the number of metal layers will remain constant even if spacing is increased, for the same metal thickness. Therefore the locus of constant number of levels also shifts to the right. The shift in the constant crosstalk curve is greater than the shift in the constant levels curve, and therefore the allowable region progressively shrinks till it disappears

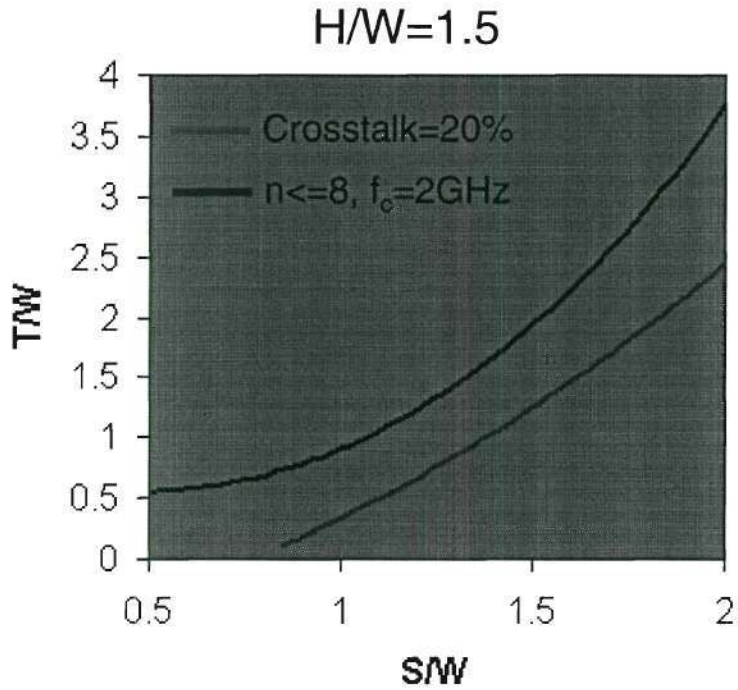


Figure 2.13: The feasible region disappears for $H/W = 1.5$.

completely.

The aim now is to investigate what the optimum aspect ratio should be in order to optimize the system characteristics of clock frequency, number of metal levels, area and power dissipation. Therefore, an ILD height/wire width ratio of $H/W = 0.5$ is used for the designs in the following section.

2.4.2 Aspect ratio optimization

2.4.2.1 Maximum clock frequency optimization

When the clock frequency is increased, the constant crosstalk curve remains unchanged because crosstalk depends only on the ratio of wire geometries. However, the constant number of metal layers curve will shift to the left, as in Figure 2.14. This is because in order to meet the higher clock frequency, the S/W ratio has to be

decreased for the same T/W ratio. The maximum permissible clock frequency is that frequency for which the constant levels locus is tangential to the allowable region (or, the constant crosstalk locus). For the case study under consideration, the maximum clock frequency is 2.8GHz, an improvement of 40% over the original 2GHz. For this design, T/W=2.7, S/W=1.0 and H/W=0.5.

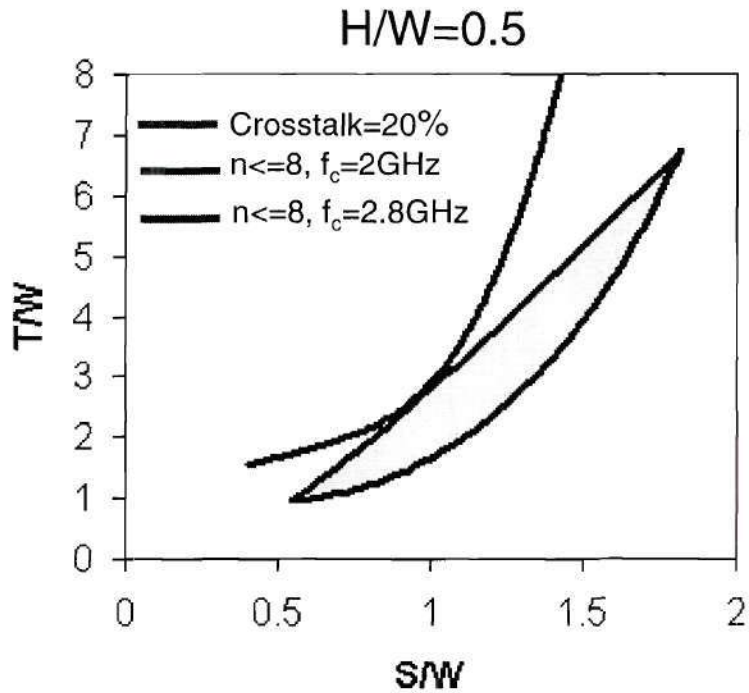


Figure 2.14: Optimum aspect ratio for maximum frequency optimization. The feasible region is shaded in light grey.

2.4.2.2 Minimum area optimization

When the area is decreased, the crosstalk curve again remains unchanged. However, to meet the number of levels constraint, spacing between the interconnects has to be decreased for the same wire thickness. Therefore, the constant number of levels locus again shifts to the left. Figure 2.15 shows the allowable region for an area of 1cm^2 (shaded in dark grey), which is smaller than, and is contained in, the allowable region

for the original area of 1.62cm^2 (shaded in light grey). The minimum area is the one for which the number of levels curve is tangential to the crosstalk locus; in this case the minimum area is 0.75cm^2, which is a reduction of over 53%. The corresponding aspect ratios are $T/W = 2.2$, $S/W = 0.8$ and $H/W = 0.5$.

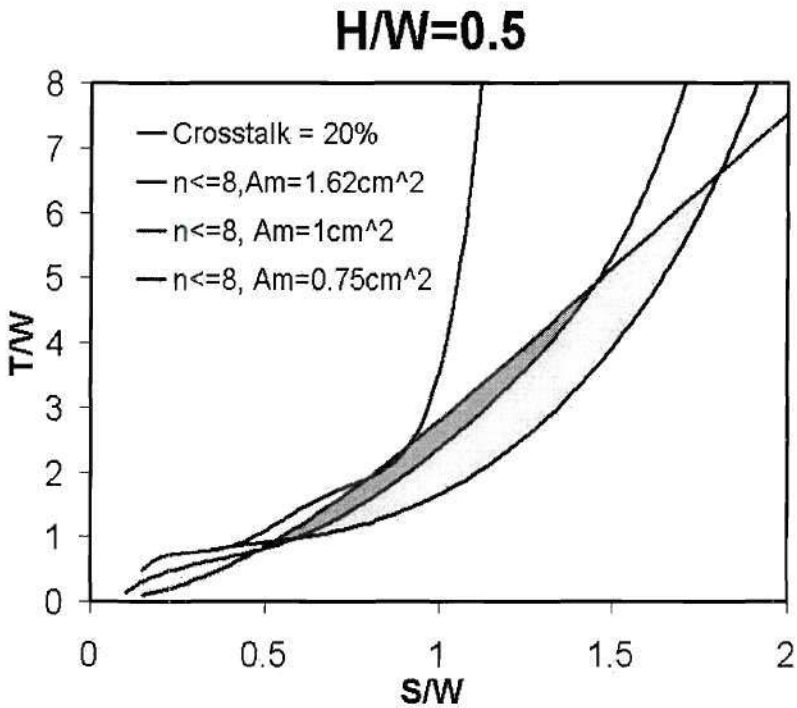


Figure 2.15: Optimum aspect ratio for minimum area optimization. Feasible region for A_m=1.62cm^2 is shaded in light grey, and that for A_m=1cm^2 is shaded in dark grey. The system clock frequency is kept constant at 2GHz.

2.4.2.3 Minimum power optimization

The interconnect power is proportional to the capacitance per unit length. Therefore, if capacitance per unit length is minimized, power dissipation will be reduced. In Figure 2.16, the constant capacitance contours are superimposed on the feasible region space. Each contour represents a constant value for the capacitance per unit length, normalized to the capacitance per unit length of an interconnect with unity aspect

ratio (UAR) i.e. W=S=T=H, where

$$C_{norm} = \frac{c_{int}}{c_{int,UAR}}. \tag{2.14}$$

There are three contours plotted for C_{norm}=2.0, 1.5 and 1.0. The contour for C_{norm} = 1.5 is tangential to the feasible region; the point of tangency determines the aspect ratio which minimizes power dissipation, and is within the feasible region. This point corresponds to $T/W = 1.2$, $S/W = 0.75$ and $H/W = 0.5$. The interconnect power dissipation at this design point is 50% greater than that of an all-unity-aspect-ratio design (because of the smaller oxide height). However, drawing the constant capacitance contours enables the designer to determine which aspect ratio results in lowest power dissipation among all possible design points within the feasible region. Therefore, it will still be considered to be the minimum power optimization. Note that the aspect ratios for the minimum power design point are different from that of the minimum area design point. Another important conclusion is the difference in aspect ratios for the "high performance" design (maximum frequency) and "low power" design (minimum power). The high performance design requires thicker wires, whereas the low power design requires thinner wires. This point will be further elaborated in Section 6.6.2.

2.4.2.4 Minimum number of levels optimization

All the points within the allowable region have between 7 and 8 metal layers. Therefore, other than reducing the area occupied by the interconnects on the topmost tier, one does not get much advantage in optimizing the number of metal levels, by choosing a different aspect ratio within the allowable region. This is due to the optimization inherent in the *n-tier* methodology which utilizes all the available area on each tier. Therefore, the minimum number of levels optimization is not explicitly

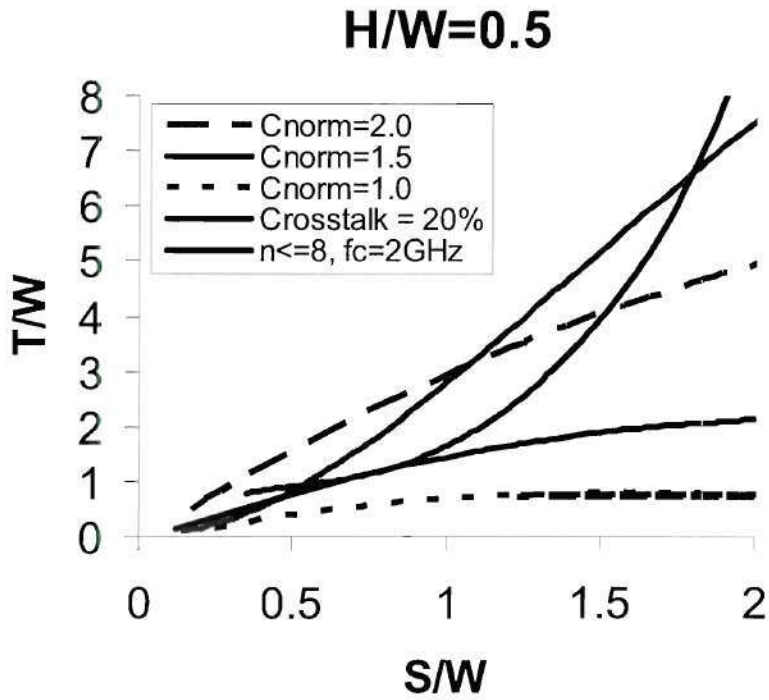


Figure 2.16: Optimum aspect ratio for minimum power optimization.

quantified here. Table 2.4 summarizes the optimum aspect ratios for the different optimizations.

Table 2.4: Summary of optimum aspect ratios for different system optimizations.

System optimization	Performance compared to W=H=T=S design	T/W	S/W	H/W
Maximum clock frequency	40% increase in clock frequency	2.7	1.0	0.5
Minimum area	53% decrease in area	2.2	0.8	0.5
Minimum power	50% higher power dissipation	1.2	0.75	0.5

2.5 Discussion of assumptions

One of the key assumptions used in the *n-tier* methodology is that the wiring efficiency factor is constant for all metal layers. However the wiring efficiency factor can be

expressed as a product of three factors :

$$e_w = e_{rout} \cdot e_{p/c} \cdot e_{via} \tag{2.15}$$

where e_{rout} is the router efficiency, $e_{p/c}$ is the power/ground/clock efficiency (= 1 - power/ground/clock blockage) and e_{via} is the via efficiency (= 1 - via blockage). The router efficiency is assumed to be a constant for all metal layers. Since power/ground/clock lines are primarily routed on global layers, the blockage caused because of these lines is greater on the upper layers and lesser on the lower layers. Chen et al. [18] have described a new via blockage model that improves Sai-Halasz's empirical model in [16], according to [46]. Using an *n-tier* case study from [47], similar to the ones described in Section 2.3, it is shown in [18] that the via blockage is significant for the lowest two layers and decreases rapidly for the upper metal layers. Because via blockage dominates at lower levels and power/ground/clock blockage dominates at higher levels, assuming a constant wiring efficiency factor for all levels is a reasonable first order approximation. An enhancement to the *n-tier* methodology would entail the use of experimentally observed values for power/ground/clock blockage and the use of models in [18] to calculate the wiring efficiency factor for each metal level.

Another assumption is that the maximum permissible time delay is 90% of the clock period for interconnects in all the non-local tiers. However, for chips with large areas and high clock frequencies, this constraint might be impossible to achieve for full chip wires. For example, the ITRS projects that the chip size of a high performance microprocessor would be 310mm^2 for the 45nm technology generation in 2010 [6]. The delay of a die edge long global interconnect, with length equal to 1.76cms and all cross-sectional dimensions equal to 1 micron, is 0.63ns. However, the clock frequency projected by the ITRS for the 45nm technology generation is 11.5GHz,

which corresponds to a clock period of 0.087ns, *almost an order of magnitude smaller than the delay of the global interconnect.* One of the solutions would be to pipeline the interconnects by inserting flip flops as has been indicated in [48]. This would make the global interconnects much thinner but communication latency would be increased.

2.6 Summary

A stochastic interconnect distribution has been used to develop a new *n-tier* multilevel interconnect optimization technique. The key utility of this methodology is to find a set of wiring pitches for each metal level so that performance targets are achieved without wasting system resources. This methodology allows direct computation of the optimum solution without exhaustive iterative techniques saving considerable re-design time.

Optimal reverse scaling enables optimizations of macrocell area, clock frequency and number of metal levels. For a case study from the 100nm technology generation, the *n-tier* design methodology is demonstrated to minimize macrocell area, clock cycle time or number of wiring tracks on the top-most tier by 32%, 16% and 62%, respectively, compared to a conventional multilevel interconnect design. An exploration of the aspect ratio design space demonstrates that choosing an optimal aspect ratio for the interconnect cross-sectional dimensions reduces macrocell area and clock cycle time by 53% and 40% respectively, over the unity aspect ratio design. The improvement in these system characteristics when repeaters are used extensively is quantified in Chapter 3.

CHAPTER 3

Optimal Repeater Insertion in Multilevel Interconnect System Design

3.1 Introduction

Repeaters improve the dependency of time delay on interconnect length from a quadratic to a linear relationship [13], [14]. Repeater insertion provides a viable option to relieve the demanding interconnect limitations placed on future GSI systems. Repeaters are increasingly being used by chip designers to improve microprocessor performance [48]. The potential of repeaters to improve the *n-tier* architecture design is quantitatively demonstrated in the following sections.

An optimal top-down repeater insertion methodology for the *n-tier* multilevel interconnect architecture is developed in this chapter. Section 3.2 describes the models used for the design, and explains the repeater insertion algorithm. The performance improvement achieved by inserting repeaters in the *n-tier* multilevel interconnect architecture is quantified by comparing with a design without repeaters in Section 3.3. The increasingly important role that will be played by repeaters for future GSI systems is investigated in Section 3.4, and a summary of the chapter is provided in Section 3.5.

3.2 Repeater insertion models and methodology

The advantages of repeater insertion can be harnessed in various ways. By using repeaters, an interconnect can be made narrower and still operate at the same clock frequency. This option decreases wire area which could result in either a reduction in macrocell area or a reduction in the number of metal layers. On the other hand, if interconnect cross-sectional dimensions are kept constant, then the operational frequency can be increased by using repeaters. The interconnect dimensions for the multilevel interconnect architecture are calculated using the repeater delay models in Section 3.2.1.

The number of repeaters that can be inserted in a multilevel wire network depends on available silicon area for repeaters. Assuming that the chip area is wire-limited, there is un-utilized silicon area that is available for repeater insertion. Only a fraction (e_{rep}) of the free area is assumed available for repeater insertion to account for practical routing and placement constraints, and additional silicon area needed for on-chip decoupling capacitors. Therefore, the total area used for repeater insertion can be expressed as :

$$A_{rep} = e_{rep} A_{free} = e_{rep} \left(A_m - A_{logic_gates} \right) . \tag{3.1}$$

where A_{rep} is the area occupied by repeaters, e_{rep} is the repeater insertion efficiency, A_{free} is the free area and A_{logic_gates} is the area occupied by logic gates. The area occupied by logic gates and repeaters is estimated using the gate area models described in Section 3.2.2.

An important concern with repeater insertion is the increase in power dissipation due to the introduction of a large number of active devices. The power dissipation of interconnects, logic gates and repeaters are estimated using the power dissipation models of Section 3.2.3.

3.2.1 Time delay models for repeater insertion

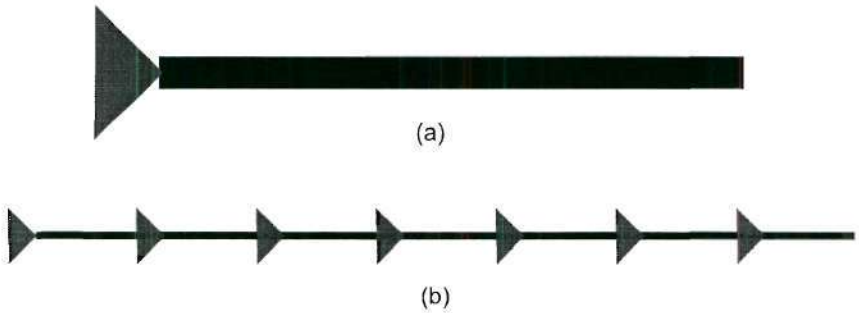


Figure 3.1: (a) Interconnect with single-driver (b) Interconnect with k repeaters, size of each of which is h times that of a minimum-sized repeater.

For a single driver interconnect circuit, such as the one in Figure 3.1(a), the 50% rc time delay is given by [21]

$$\tau = 0.4rcL^2 + 0.7\left(R_{tr}cL + R_{tr}C_L + rC_LL\right) \tag{3.2}$$

However, when k repeaters are inserted in the interconnect, and the size of each repeater is h times that of a minimum-sized repeater (Figure 3.1(b)), then the overall 50% delay of the interconnect system is given by

$$\tau = k\left[0.7\frac{R_o}{h}\left(\frac{cL}{k} + hC_o\right) + \frac{rL}{k}\left(0.4\frac{cL}{k} + 0.7hC_o\right)\right]. \tag{3.3}$$

Bakoglu [13],[14] derived an expression for the optimal time delay of an interconnect when the number and size of equi-spaced repeaters is "optimal", which minimizes the cumulative delay of repeaters and interconnect segments. This optimal time delay is given by

$$\tau = 2.46\sqrt{rcR_oC_o}\ L \tag{3.4}$$

where R_o and C_o are the output resistance and input capacitance of a minimum size inverter, respectively. The corresponding optimum number of repeaters, k_{opt}, and optimum size of repeaters, h_{opt}, are

$$k_{opt} = \sqrt{\frac{0.4rcL^2}{0.7R_oC_o}} \tag{3.5}$$

and

$$h_{opt} = \sqrt{\frac{R_o c}{r C_o}}. \tag{3.6}$$

However, the time delay of an interconnect decreases very slowly beyond a certain point with increasing number of repeaters, as seen in Figure 3.2. Therefore, it is

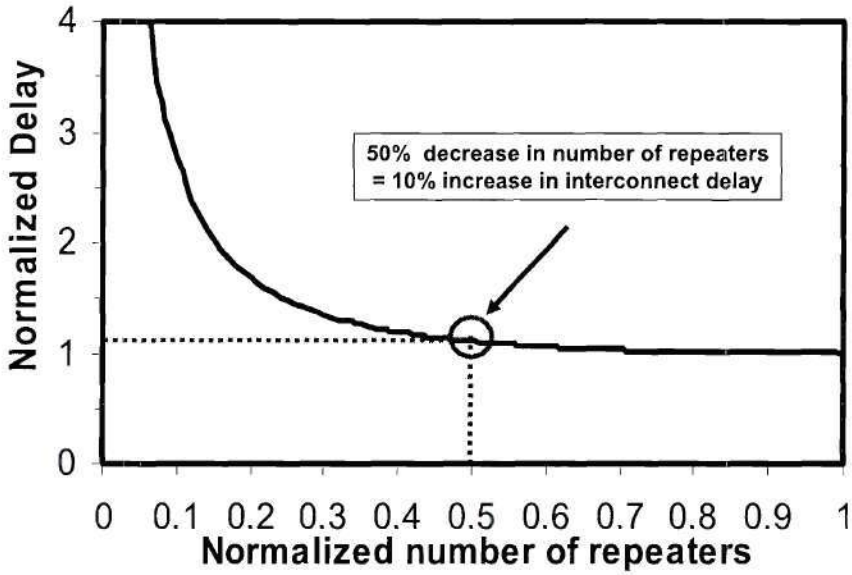


Figure 3.2: Time delay vs. number of repeaters, normalized to optimum time delay and optimum number of repeaters, respectively. Using half the optimum number of repeaters increases time delay by only 10%.

possible to achieve a "sub-optimal" design by reducing the number of repeaters significantly with only slight increases in delay. Therefore, let the number of repeaters be a factor ζ times the "optimal" number of repeaters, $0 < \zeta \leq 1$, while the size of repeaters is equal to the optimal size, h_{opt}. For this sub-optimal design, the time delay can be expressed as

$$\tau_\zeta = \zeta k_{opt} \left[0.7 \frac{R_o}{h_{opt}} \left(\frac{cL}{\zeta k_{opt}} + h_{opt} C_o \right) + \frac{rL}{\zeta k_{opt}} \left(0.4 \frac{cL}{\zeta k_{opt}} + 0.7 h_{opt} C_o \right) \right]. \tag{3.7}$$

Substituting (3.5) and (3.6) into (3.7) and simplifying gives the time delay for the

sub-optimal design as

$$\tau_\zeta = \left(1.4 + 0.53\zeta + \frac{0.53}{\zeta}\right)\sqrt{rcR_oC_o}\ L. \tag{3.8}$$

When the full quota of optimum number of repeaters are inserted, then $\zeta = 1$; this reduces (3.8) to (3.4), as is to be expected. Also, the smallest possible value of ζ is such that the number of repeaters is zero, or the number of repeaters plus driver equals 1. i.e.

$$\zeta k_{opt} = 1, \tag{3.9}$$

or

$$\zeta = \frac{1}{k_{opt}}. \tag{3.10}$$

Therefore, ζ is always greater than 0, precluding the possibility of singularities in (3.7) and (3.8).

This sub-optimal design in (3.8) utilizes a smaller number of repeaters resulting in a larger delay. The trade-off between the performance and the number of repeaters is shown in Figure 3.2, which plots the ratio of the sub-optimal delay in (3.8) to the optimal delay in (3.4) versus ζ. *A 50% reduction in the number of repeaters from the optimal number imposes a performance penalty of only 10%.* The considerable savings in silicon area, wiring complexity and power dissipation encourages this worthwhile trade-off. Hence, the maximum number of repeaters that will be inserted in any interconnect in this work is equal to half the Bakoglu optimum number.

For simplicity, it is again assumed that all the interconnect aspect ratios in any tier of the multilevel architecture are equal to unity. Substituting (2.8) and (2.9) in (3.8), the 50% time delay (τ_ζ) of the longest interconnect of tier t, with sub-optimum repeaters, can be written as

$$\tau_\zeta = \frac{\beta}{f_c} = \left(1.4 + 0.53\zeta + \frac{0.53}{\zeta}\right)\frac{2}{p_t}\sqrt{6.2\rho\varepsilon_r\varepsilon_o R_o C_o}\sqrt{\frac{A_m}{N_g}}L_t. \tag{3.11}$$

Therefore, the wire pitch of the tth tier can be written as

$$p_t = \left(1.4 + 0.53\zeta + \frac{0.53}{\zeta}\right) \frac{2}{\beta/f_c} \sqrt{6.2\rho\varepsilon_r\varepsilon_o R_o C_o} \sqrt{\frac{A_m}{N_g}} L_t. \tag{3.12}$$

3.2.2 Gate area models

The area of a logic gate or a repeater, A_g, is calculated as [35]

$$A_g = k_I \left(1 + \frac{4\sqrt{G_{ar}}\,(f_i - 1)}{\sqrt{k_I}}\right) \left(1 + \frac{(1+\beta_g)\,(w_k - 1)}{\sqrt{k_I G_{ar}}}\right) F^2 \tag{3.13}$$

where $k_I \approx 102$ is the area of a minimum sized inverter with respect to F^2, $G_{ar} \approx \frac{17}{6}$ is the gate aspect ratio, $f_i \approx 3$ is the number of inputs, $\beta_g \approx \frac{2}{f_i}$ is the ratio of pFET to nFET width and w_k is the nFET width to feature size ratio. The pFET width is constrained to satisfy equal worst case rise and fall times, and w_k is calculated by equating the critical path delay to the cycle time ($= 1/f_c$) [49], [50],

$$\frac{1}{f_c} = \frac{n_{cp} T_{PDn} f_{ineff}}{b}, \tag{3.14}$$

where T_{PDn} is the nFET propagation delay including the transition time effect that is derived from the physical alpha-power law model [49], [50], f_{ineff} is the effective fan-in factor for series connected MOSFETs [51], n_{cp} is number of gates in a critical path and b is the clock skew factor (=0.9). Using (3.8), (3.1) and (3.13), the total logic gate area is estimated, which defines the limit on the maximum amount of area available for repeaters.

3.2.3 Power dissipation models

The total power dissipation (P_{total}) in a macrocell is defined as

$$P_{total} = P_{logic} + P_{int} + P_{rep} \tag{3.15}$$

where P_{logic}, P_{int} and P_{rep} are the power dissipation in the logic gates, interconnects and repeaters, respectively. The power dissipation per gate (i.e. logic gate or repeater), P_g, is defined as

$$P_g = \frac{a}{2} w_k C_{go} V_{dd}^2 f_c \tag{3.16}$$

where a is the activity factor (=0.1), V_{dd} is the supply voltage and C_{go} is the gate overlap, junction and fan-out capacitance for a minimum sized gate. Note that this is the average switching power (during a clock period) and that static or leakage power is neglected. The impact of leakage power is discussed later in Section 3.3.4. The product of the number of gates and power dissipation per gate gives the total logic or repeater power dissipation. The dynamic power dissipated in the interconnects is calculated by

$$P_{int} = \frac{a}{2} C_w V_{dd}^2 f_c \tag{3.17}$$

where C_w is the total wiring capacitance given by

$$C_w = c\left(L_{total}\sqrt{\frac{A_m}{N_g}}\right) \tag{3.18}$$

where c is the distributed wiring capacitance per unit length, L_{total} is the total length of interconnects in gate pitches and $\sqrt{\frac{A_m}{N_g}}$ is the average gate pitch.

3.2.4 Repeater insertion methodology

For a given macrocell area, A_m, and clock frequency, f_c, the *n-tier* methodology described in Section 2.2 is first used to design the multilevel interconnect architecture. The models described in (3.8)-(3.18) are then used to determine the maximum number and size of repeaters that can be inserted in the multilevel architecture. The repeater insertion efficiency, e_{rep}, is assumed to be equal to 60%. Since longer global interconnects benefit the most by using repeaters, a top-down repeater insertion methodology is adopted, as shown in Figure 3.3. Repeaters are first inserted in

the uppermost tier. Repeater insertion then continues downward to the lower tiers depending on the amount of free silicon area available. This algorithm tries to insert 50% of the optimal number of repeaters (ζ=0.5) in all the wires on a given tier; if the

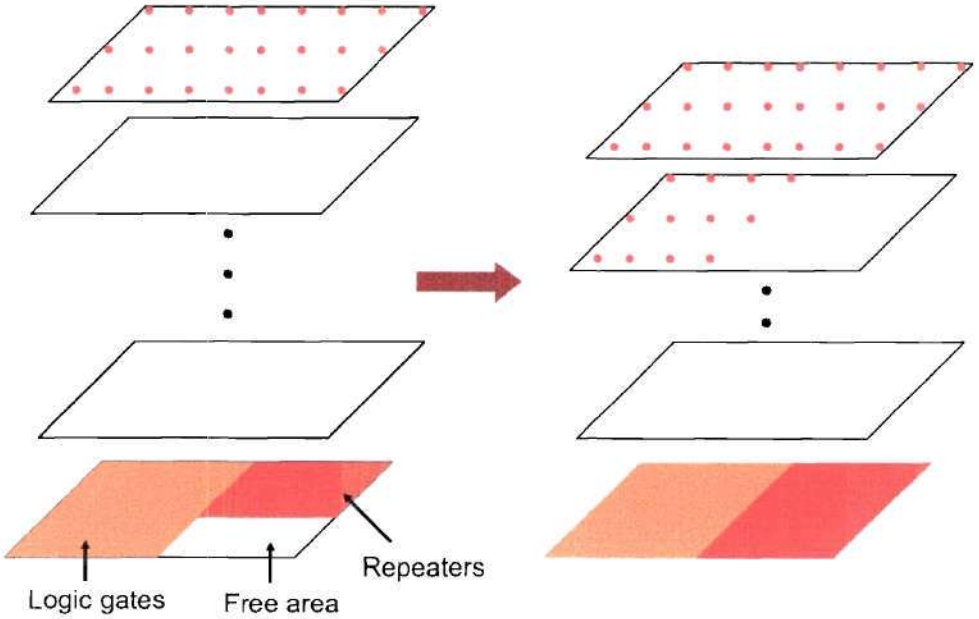


Figure 3.3: Top-down repeater insertion methodology.

area required for repeaters is not available, ζ is systematically decreased until either the repeater area constraint is satisfied or the number of repeaters reaches zero. If the interconnect width decreases to minimum width, then repeater insertion is discontinued. The resulting architecture is the new multilevel interconnect architecture with repeaters. The flowchart for the complete repeater insertion methodology for the *n-tier* architecture is shown in Figure 3.4.

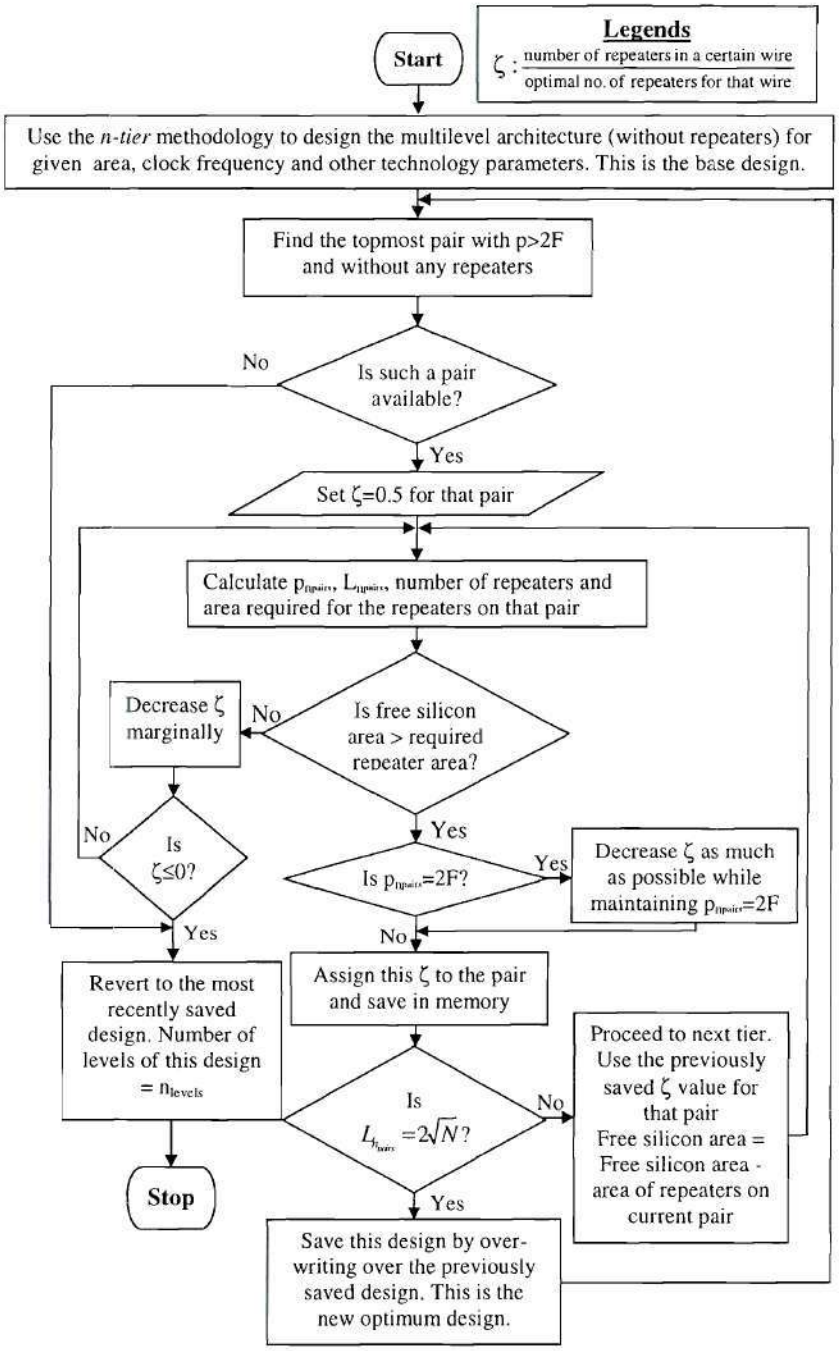


Figure 3.4: Flowchart for *n-tier* with repeaters design methodology.

3.3 Performance enhancement using repeaters for *n-tier* architectures

The maximum clock frequency achievable by the optimized *n-tier* architecture without repeaters for the macrocell case study (n=8 metal levels, A_m=1.62cm^2) described in Section 2.3 is 2GHz, as shown by point D (*n-tier baseline design*) in Figure 2.7. In this section, repeaters are inserted into the *n-tier* design to further improve the system design. The parameters used for the repeater insertion methodology, in addition to those in Table 2.1, are summarized in Table 3.1. Repeaters are used to minimize the area, cycle time, number of metal levels or power dissipation of the macrocell case study. These optimizations are compared against the *n-tier* baseline design (without repeaters) and the results are summarized in Table 3.2.

Table 3.1: Summary of various parameters used for repeater insertion in the ASIC macrocell case study based on the ITRS 100nm technology generation (see Table 2.1 also).

Parameter	Symbol	Value
Supply voltage	V_{dd}	1.0V
Threshold voltage	V_T	0.125V
Gate oxide thickness	t_{ox}	1.8nm
Number of gates in critical path	n_{cp}	10
Activity factor	a	0.1
Area of minimum sized inverter	[F^2]	102
Inverter aspect ratio	G_{ar}	17.0/6.0
Clock skew factor	b	0.9
Repeater efficiency	e_{rep}	60%

3.3.1 Minimum macrocell area

The area of a macrocell with 8 metal levels and f_c=2GHz is minimized using repeaters in this section. The wire-limited area is the minimum area required to route

Table 3.2: Interconnect parameters for the various design points

Tier # (n)	No. of levels	L_n gate pitches	L_n (cms)	p_n (μm)	number of repeaters	ζ	NFET $\frac{W}{L}$ ratios	Dynamic Power
Without repeaters : *n-tier* baseline design								
A_m=1.62cm^2, f_c=2GHz and $n_{n_{levels}}$=8 metal levels.								
Tier 4	2	6723	2.55	1.71	0	0		P_{total}=43W
Tier 3	2	1871	0.71	0.95	0	0	$\frac{W}{L}_{logic}$=17	P_{logic}=71%
Tier 2	2	885	0.34	0.45	0	0		P_{int}=29%
Tier 1	2	208	0.08	0.20	0	0		P_{rep}=0%
With repeaters : Minimum area and power								
A_m=0.48cm^2, f_c=2GHz and $n_{n_{levels}}$=6.97≈8 metal levels.								
Tier 4	0.97≈2	6723	1.39	0.73	49840	0.216		P_{total}=24W
Tier 3	2	1003	0.21	0.28	0	0	$\frac{W}{L}_{logic}$=9	P_{logic}=69%
Tier 2	2	283	0.06	0.20	0	0	$\frac{W}{L}_{rep}$=114	P_{int}=29%
Tier 1	2	34	0.007	0.20	0	0		P_{rep}=2%
With repeaters : Maximum frequency								
A_m=1.62cm^2, f_c=2.44GHz and $n_{n_{levels}}$=6.49≈8 metal levels.								
Tier 4	0.49≈2	6723	2.55	1.36	34855	0.308		P_{total}=150W
Tier 3	2	1684	0.64	0.94	0	0	$\frac{W}{L}_{logic}$=63	P_{logic}=89%
Tier 2	2	810	0.31	0.45	0	0	$\frac{W}{L}_{rep}$=211	P_{int}=10%
Tier 1	2	193	0.07	0.21	0	0		P_{rep}=1%
With repeaters : Minimum no. of levels								
A_m=1.62cm^2, f_c=2GHz and $n_{n_{levels}}$=4.99≈6 metal levels.								
-	-	-	-	-	-	-		P_{total}=44.4W
Tier 3	0.99≈2	6723	2.57	1.21	75390	0.26	$\frac{W}{L}_{logic}$=17	P_{logic}=69%
Tier 2	2	885	0.34	0.45	0	0	$\frac{W}{L}_{rep}$=188	P_{int}=28%
Tier 1	2	208	0.08	0.20	0	0		P_{rep}=3%

all the interconnects within a maximum of 8 metal levels. The transistor-limited area is the minimum area required for accommodating the logic gates and repeaters. The macrocell area is equal to the larger of these two areas i.e. it is either wire limited or transistor limited, depending on whether the wire-limited or transistor-limited area is greater. Before any repeaters are inserted, the macrocell area is wire-limited, as illustrated in Figure 3.5. By inserting repeaters, the interconnects become narrower which decreases the required wire area and subsequently reduces wiring capacitance, C_w (3.18). As C_w decreases, the logic gates can be made smaller thereby decreasing

the transistor-limited area. As number of repeaters increases, the wire-limited macrocell area decreases until it equals the transistor-limited area. If more repeaters are inserted, then the macrocell area becomes transistor limited, and starts increasing (due to increases in the number of repeaters and the size of logic gates). Therefore, for a given number of metal levels, the macrocell area is minimized when the wire- and transistor-limited areas become equal. As seen from Figure 3.5, optimal repeater insertion decreases the macrocell area from 1.62cm^2 to 0.39cm^2, which is an almost 76% reduction in area.

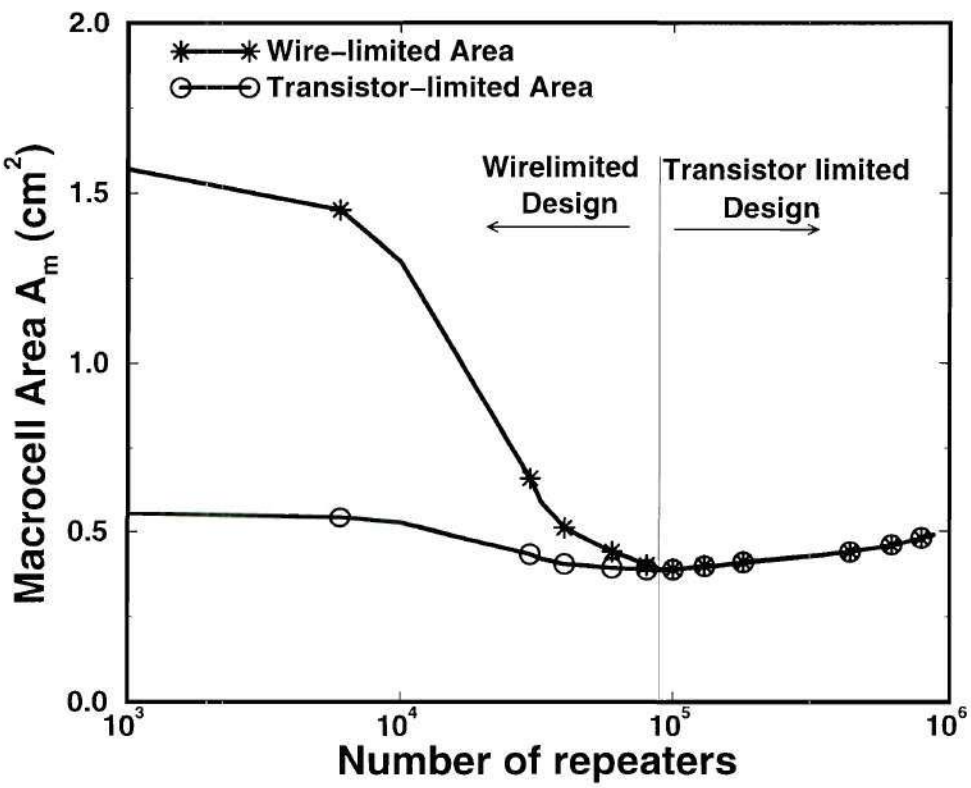


Figure 3.5: Minimum area optimization using repeaters for $n \leq 8$ levels and f_c=2GHz.

Lately, power dissipation has become an important concern for microprocessors. Repeaters are active devices, and hence insertion of repeaters is traditionally considered to be achieving superior performance at the expense of higher power dissipation.

Therefore, the repeater insertion methodology should ensure that the power budget is not surpassed. The introduction of the additional power constraint on macrocell area minimization is shown in Figure 3.6. It is assumed that the maximum heat removal capacity for an air-cooled system is 50 W/cm^2. The power limited area is defined as the minimum macrocell area required to keep the power dissipation density within the specified upper bound. This is calculated by dividing the total power dissipation of the macrocell by 50 W/cm^2. The macrocell area is the maximum of the wire limted, transistor limited and power limited areas.

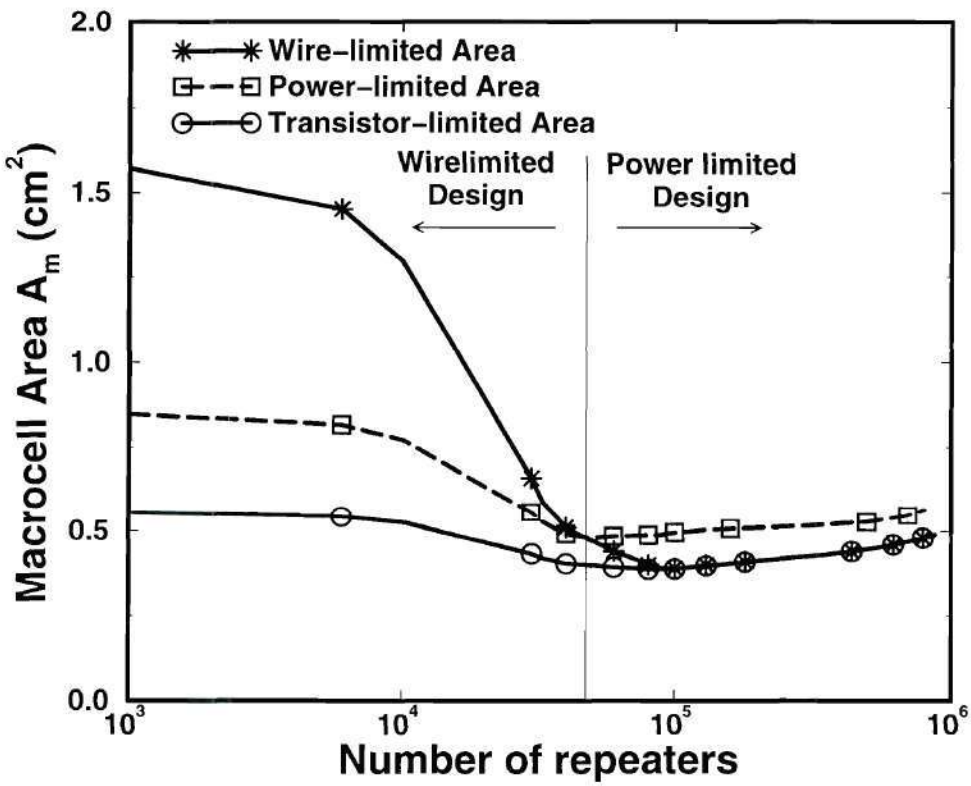


Figure 3.6: Minimum area optimization using repeaters for $n \leq 8$ levels and f_c=2GHz, with the additional constraint of power-limited area.

Increasing the number of power-dissipating repeaters increases the power-limited area. As the number of repeaters increases, the wire-limited macrocell area decreases

until it equals the power-limited area, as shown in Figure 3.6. If more repeaters are inserted, then the macrocell area becomes power-limited and begins to increase. Thus, for this example, the macrocell area is minimized when the wire and power-limited areas become equal. Since this occurs *before* the transistor-limited minimum of Figure 3.5, the new minimum power-limited macrocell area is slightly larger than the earlier transistor-limited minimum area. As seen from Figure 3.6, optimal repeater insertion decreases the macrocell area from 1.62cm^2 to 0.48cm^2, almost a 70% reduction in the cell size. If more advanced heat removal mechanisms are used, such as liquid or two phase cooling, then the power limited area curve would shift lower. Then the wire-limited area may decrease till it equals the transistor-limited area. From here the macrocell area would start to increase due to an increase in the number of repeaters and the size of logic gates required to achieve the desired clock frequency for an increased wiring capacitance.

3.3.2 Maximum frequency optimization

Figure 3.7 shows a plot of clock frequency versus number of repeaters to maximize the clock frequency through repeater insertion for a macrocell with A_m=1.62cm^2 and 8 metal levels.

The wire-limited clock frequency is the maximum frequency for which all the transistors can be wired within 8 metal levels for the specified macrocell area. The wire-limited clock frequency increases with an increase in the number of repeaters because repeaters decrease wire delay for a constant pitch. The transistor-limited clock frequency is the maximum frequency at which the logic critical path transistors can operate if they are enlarged to occupy all remaining macrocell area after accounting for repeaters. Transistor-limited clock frequency decreases with an increase in the

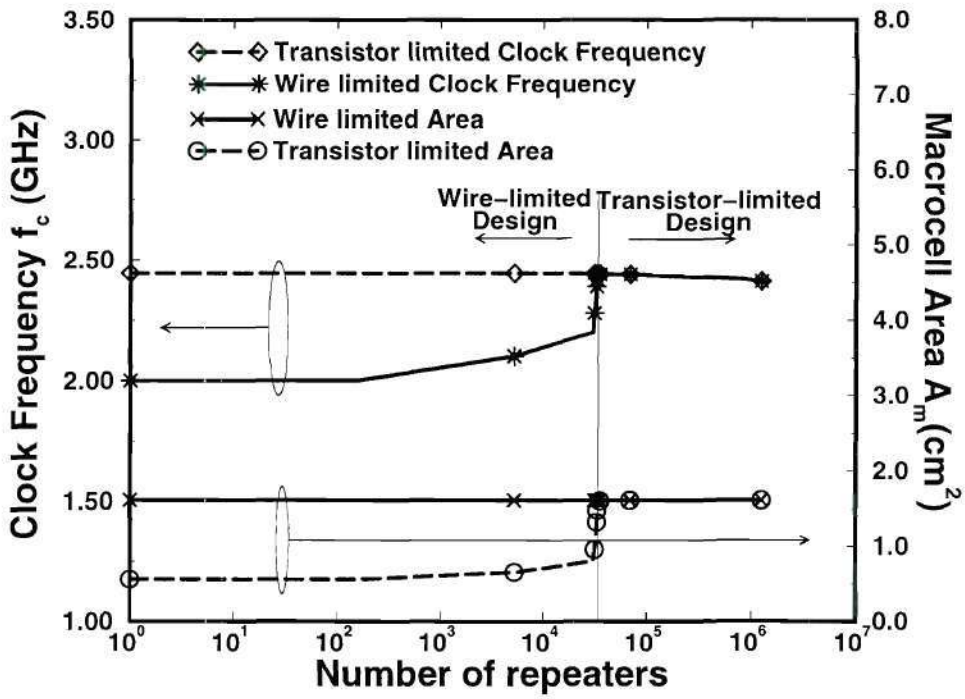


Figure 3.7: Maximum clock frequency optimization using repeaters for $A_m = 1.62cm^2$ and $n \leq 8$ levels.

number of repeaters due to a decrease in the area available for logic gates. Initially, when the macrocell area is wire-limited, the clock frequency is also wire-limited. As the number of repeaters increases, the wire and transistor limited-frequencies eventually converge when the macrocell area becomes transistor-limited. More repeaters can be added only by shrinking the logic transistors which reduces the transistor-limited frequency. Hence, the clock frequency peaks at the point where the wire and transistor limited frequency curves intersect. Therefore, repeater insertion increases the maximum clock frequency from 2GHz to 2.44GHz, which is a 22% improvement.

For aggressive designs with a smaller number of gates per critical path, the clock frequency optimization yields even better results. If the number of gates in the critical path is decreased from 10 to 5, the size of the logic gates decreases, because the gates are sized so that the critical path delay equals the clock period. The reduction in total

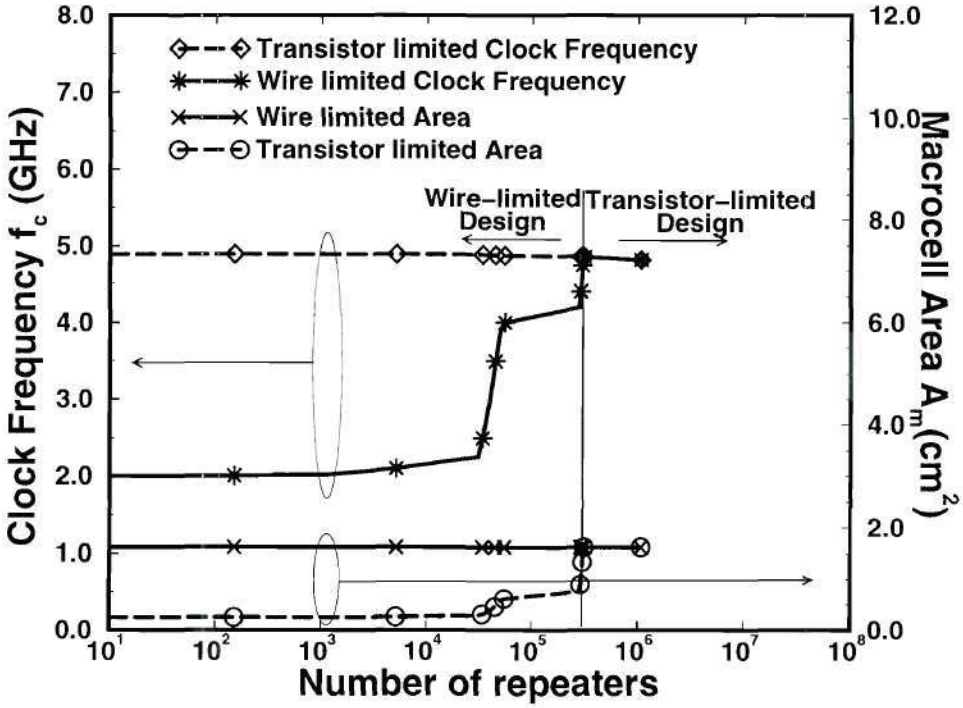


Figure 3.8: Maximum clock frequency optimization using repeaters for $A_m = 1.62cm^2$ and $n \leq 8$ levels, when n_{cp}=5, instead of 10.

logic gate area has two desirable consequences. Firstly, there is more area available for repeaters, and secondly, the transistor-limited clock frequency increases. The maximum clock frequency optimization for the case study with $n_{cp} = 5$ is shown in Figure 3.8. The transistor-limited clock frequency has almost doubled to nearly 5GHz. Using almost an order of magnitude greater number of repeaters, the maximum clock frequency that can be achieved for a macrocell with area A_m=1.62cm^2 and 8 metal levels has increased from 2GHz to 4.85GHz, which is an improvement of 142%.

3.3.3 Minimum number of metal levels

The number of metal levels required for a macrocell with A_m=1.62cm^2 and f_c=2GHz is minimized using repeaters. Figure 3.9 plots the number of metal levels vs. the number of repeaters for this design. Increasing the number of repeaters decreases

the number of metal levels and increases the transistor-limited area. Once the wire-

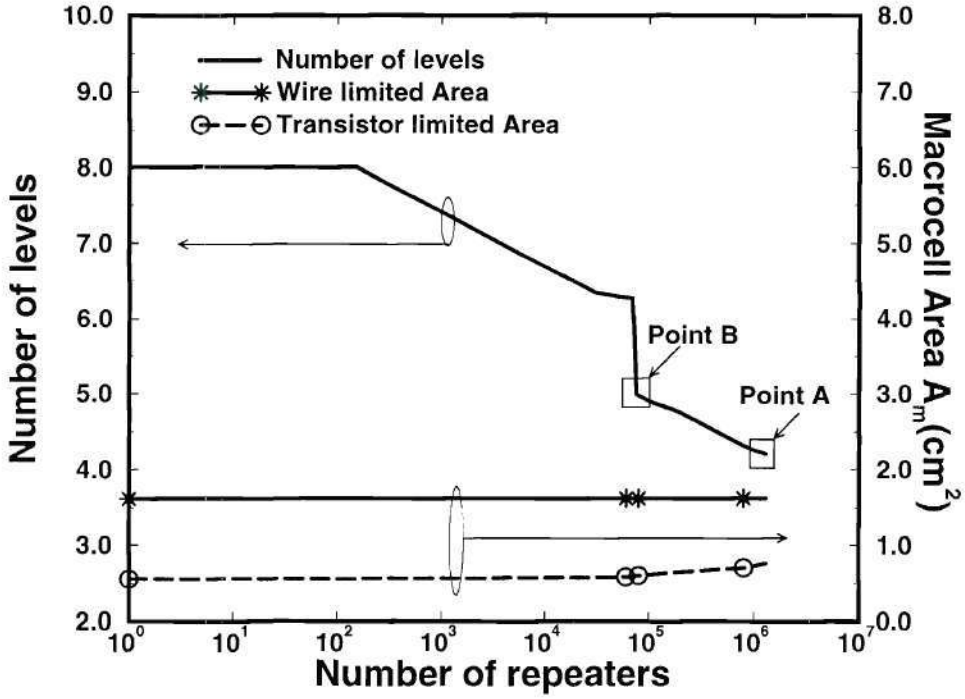


Figure 3.9: Minimum number of levels optimization using repeaters for $A_m = 1.62cm^2$ and f_c=2GHz.

limited and transistor-limited areas become equal, continued repeater insertion will increase macrocell area. However, in Figure 3.9, the minimum number of levels optimization occurs before the convergence of wire and transistor-limited areas when all tiers become saturated with repeaters (point A), such that they either have a quasi-optimum number of repeaters (ζ=0.5) or have reached minimum wire pitch. Since the levels are grouped in pairs, the number of levels should be rounded to the next higher even integral value. Therefore, in this example, optimal repeater insertion decreases the number of levels from 8 to 6, which is a reduction of 2 metal levels. From a designer's perspective, it would be prudent to choose point B, in Figure 3.9, as the design point since it has the same number of levels (six levels) but requires a far fewer

number of repeaters than point A. The specifications for point B are tabulated in Table 3.2.

3.3.4 Minimum power dissipation

The total power dissipation of the macrocell with 8 metal levels and operating at f_c=2GHz can be significantly reduced using repeaters. Figure 3.10 plots the power dissipation versus the number of repeaters for this design. Comparing Figure 3.6 and Figure 3.10, as the macrocell area decreases with an increase in the number of repeaters, the power dissipation in logic gates and interconnects also decreases. Reducing macrocell area decreases the average interconnect length and hence reduces the average wiring capacitance. Therefore, the size of the logic gates can be reduced, which decreases the gate capacitance in (3.16), reducing the total logic gate power. In Figure 3.10, the repeater power dissipation increases monotonically because the effect of decreasing repeater size is overshadowed by the increase in the number of repeaters. From (3.17), the total interconnect power is also decreased because of a reduction in the macrocell area. Because the repeater power is small compared to the logic gate and interconnect power, the total power dissipation decreases and reaches a minimum when the macrocell area is minimized. Beyond this point, the increase in the macrocell area and number of repeaters causes the power dissipation in the logic gates, interconnects, and repeaters to increase and results in an increase in the total power dissipation. Moreover the design becomes power-limited because the power dissipation density is at the maximum permissible limit of 50 W/cm^2. From Figure 3.10, repeater insertion decreases the total power dissipation from 43W to 24W, a reduction of 44%, which corresponds to the minimum area design point of A_m=0.48cm^2.

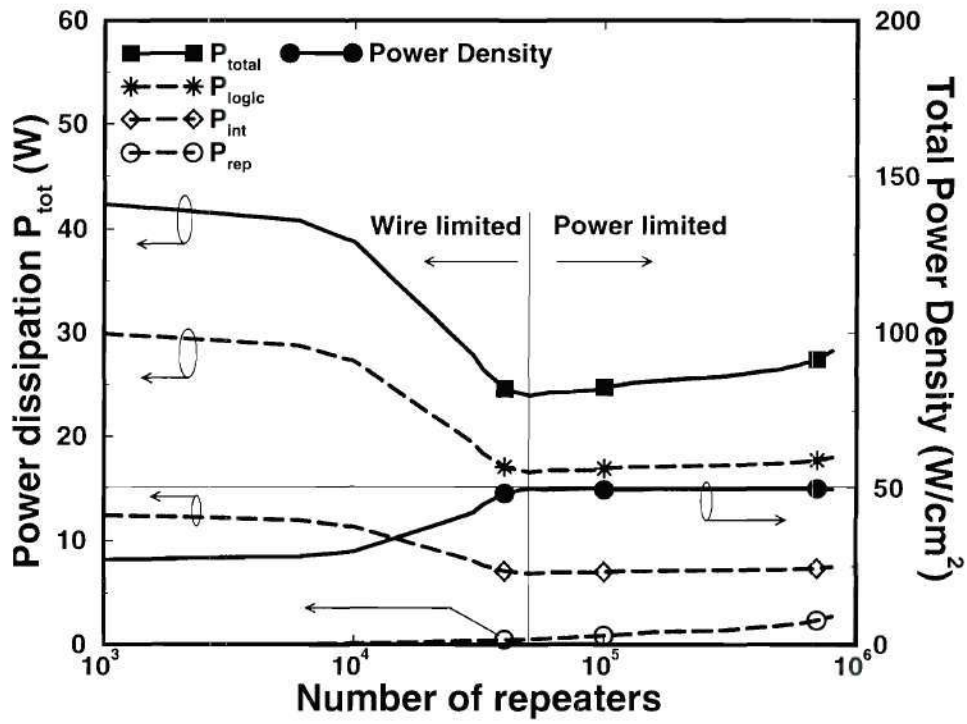


Figure 3.10: Minimum power optimization using repeaters for $n \leq 8$ levels and f_c=2GHz.

For simplicity, (3.16) does not include the leakage power; however, it can be demonstrated that the minimum area design point also corresponds to the minimum total leakage power for this example. The leakage power for a CMOS chip is given by

$$P_{off} = W_{tot} V_{dd} I_{off} \tag{3.19}$$

where W_{tot} is the cumulative macrocell transistor width and I_{off} is the off-current per device width [52]. From Table 3.2, the W/L ratio of the logic gates decreases 46% by inserting repeaters. Although the W/L ratio of the repeaters is comparatively larger, the number of repeaters is two orders of magnitude smaller than the number of logic gates. Therefore, when the macrocell area is minimized, the W/L ratio for logic gates is at a minimum and this minimizes the total leakage power. Hence, the inclusion of leakage power should not change the minimum area and power design point.

The wiring cross-section of the multilevel interconnect architecture for the *n-tier* design without repeaters, and the optimal repeater-inserted designs is shown in Figure 3.11. The area occupied by repeaters in all the above optimized designs is less than 2% of the macrocell area. In [48], the authors report that the total area consumed by repeaters is about 6% of the total die area. This indicates that the results of the methodology demonstrated in this paper are well within acceptable industry limits.

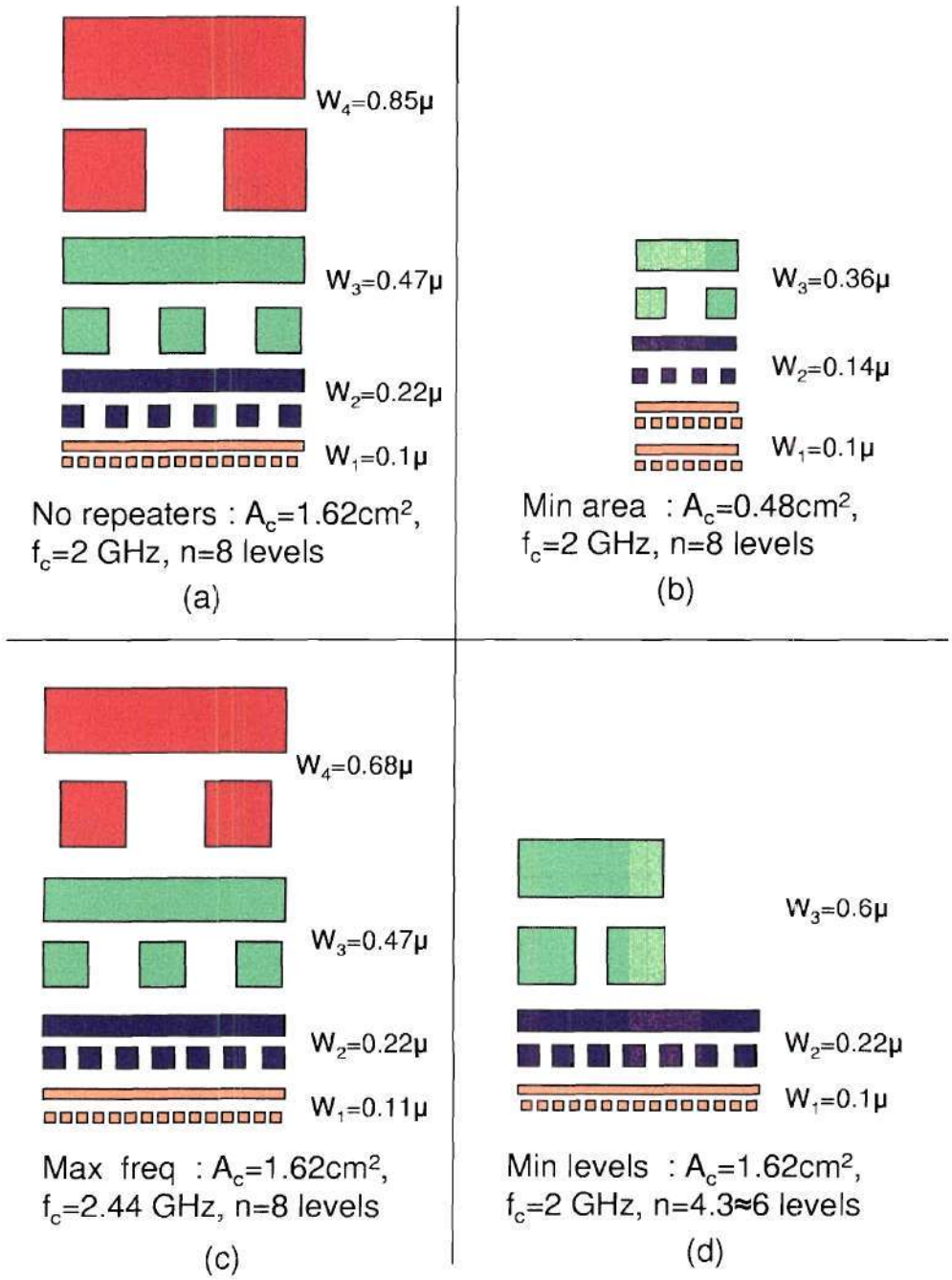


Figure 3.11: Cross-section of n-tier architecture for (a) no repeaters, and for optimal repeater insertion for (b) minimum area, (c) maximum clock frequency and (d) minimum number of metal levels.

3.4 Prospects of repeater insertion for GSI

Conventional reverse-scaled multilevel interconnect architecture enhances wire performance at the expense of lower wire density. On-chip wiring will limit die size, clock frequency and minimum number of metal layers for future designs, thereby forcing the use of *n-tier* multilevel interconnect architectures that utilize repeaters extensively to increase signal propagation speed and/or wire density. Until recently, repeater insertion was an ad-hoc activity, performed on an as-needed basis to alleviate delay violations in critical paths. However, the latest research in this area advocates providing dedicated repeater blocks on-chip and using repeaters more extensively for the signal interconnects.

To highlight the increasing importance of repeater insertion to keep Moore's Law on track, an ASIC design is investigated here based on ITRS projections for 15 years [11]. Starting with the assumption that there are one million highly-connected logic gates in a logic macrocell for the 180nm generation of 1999, the number of metal levels is projected over the next 15 years by doubling the number of logic gates in a macrocell every two years. The macrocell areas, clock frequencies and other technology parameters for each design is based on the ITRS projections. As seen in Figure 3.12, without the use of repeaters (scenario 1), the number of required metal levels approaches unrealistic values beyond 2005. In fact, the number of projected levels in 2014 is almost an order of magnitude larger than the number of levels prescribed by the ITRS.

The interconnect area required for the Moore's Law scaling beyond 2005 for the first scenario exceeds the available area provided by the ITRS. To avoid this situation, one possible solution is to limit the size of each macrocell so that wiring demand is reduced. Therefore, as an alternative to the historical approach of doubling number

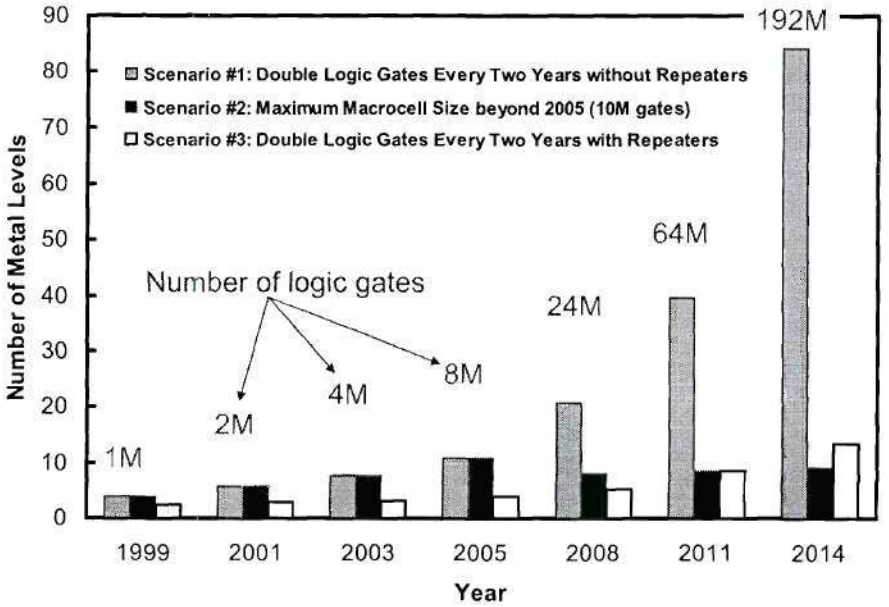


Figure 3.12: Projection of the number of metal levels for highly-connected logic macrocells from 1999 to 2014.

of logic gates, Figure 3.12 also shows that restricting the maximum number of gates per macrocell to around 10 Million (scenario 2), keeps the number of metal levels per macrocell to a controllable number through 2014. This example demonstrates that the highly restrictive interconnect limits are powerful enough to disrupt Moore's Law.

In order to get Moore's Law back on track, another possible solution is to use interconnect repeaters to increase wire density. Figure 3.12 shows the results of repeater insertion in an *n-tier* multilevel interconnect architecture (scenario 3) for megacells whose number of logic gates doubles every two years. These results suggest that significant repeater insertion throughout a logic megacell could keep the number of metal levels within reasonable limits through 2011. Extensive repeater insertion, however, would drastically alter current chip design characteristics. For example, as seen in Figure 3.13, the number of transistors required for repeater insertion beyond 2005 is about 4-13% of the total logic transistors; the amount of area required for

repeaters is roughly 40% of the total macrocell area; and the power dissipated by repeaters is about 20-30% of the total power dissipation.

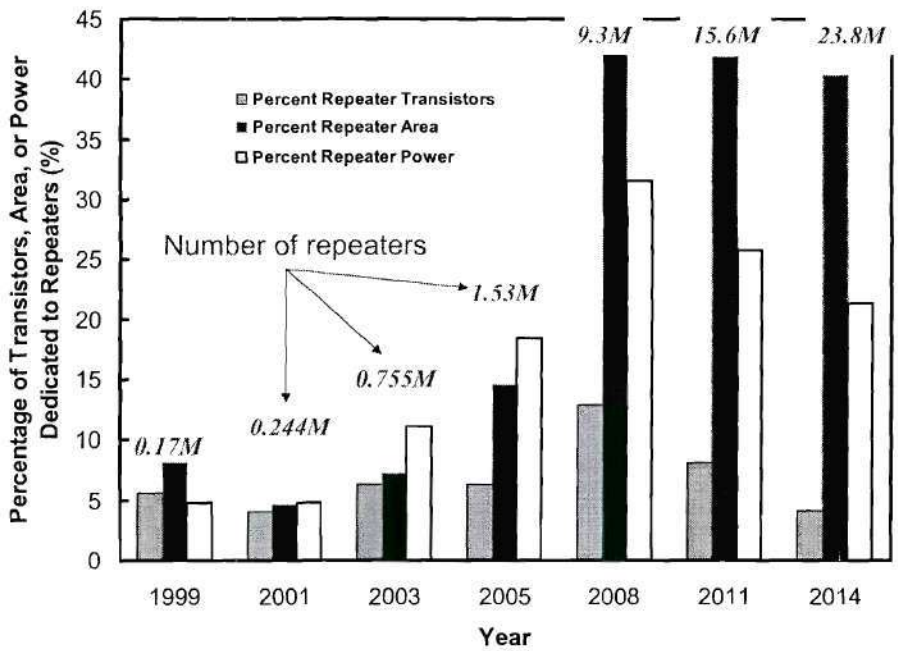


Figure 3.13: Projection of the effects of repeaters on macrocell design characteristics.

This discussion suggests that repeaters could have an increasingly important role to play to keep Moore's Law alive for the next decade. When on-chip inductance effects start to dominate, then repeaters will have an even more significant function to decrease inductive crosstalk, as will be described in Chapter 5. The presence of three levels of cache in recent microprocessors is already commandeering a significant chunk of transistors for on-chip memory. It may well be possible that the increasing use of repeaters would force a tripartite allocation of the transistors in a future GSI chip for logic, memory and communication (repeaters), as shown in Figure 3.14.

Therefore, there is an urgent need to include repeater insertion at an early stage of the design cycle to ensure that they are optimally utilized in chip design. Recent research on interconnect-centric design is one possible way to implement this

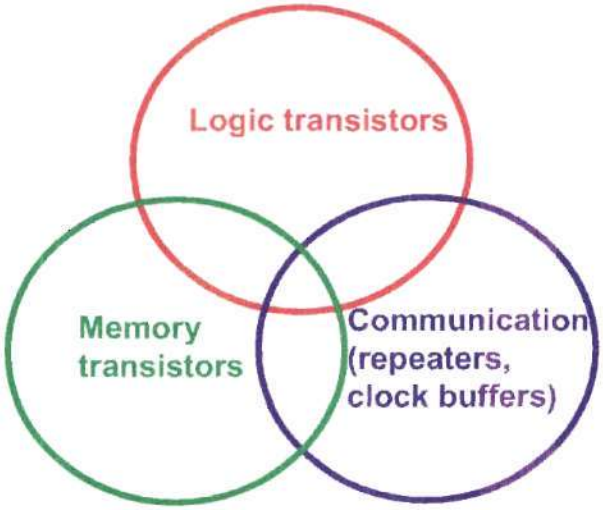


Figure 3.14: Tripartite functional allocation of transistors in a future GSI chip among logic, memory and communication transistors.

objective.

3.5 Summary

The *n-tier* methodology described in Chapter 2 is enhanced by developing a top-down repeater insertion methodology that uses the free silicon area for inserting repeaters in the interconnects. Starting with the upper-most tier, this methodology inserts repeaters until all free silcon area is consumed. It has been demonstrated that the area and power dissipation of a macrocell can be minimized *simultaneously* by using repeaters. Optimizations of clock frequency and number of metal layers have also been quantified using a case study. These results illustrate the main advantages of extensively utilizing repeaters in an *n-tier* multilevel interconnect architecture to alleviate the restrictive wiring demands of future GSI systems. The *n-tier* design methodologies enable an *a priori* (i.e. before physical design and layout) optimization

of the multilevel interconnect architecture. Detailed system level optimization, instead of conventional heuristics is necessary to extract maximum system performance for future designs.

In order to design the multilevel interconnect network for multi-GHz frequency designs, *rc* models are inadequate. New distributed *rlc* models are developed in Chapters 4 and 5 which will be used to evaluate the effects of inductance on the *n-tier* methodology in Chapter 6. The *n-tier* design is validated against commercial data in Chapter 7. The complete C code for the *n-tier* design methodology with repeater insertion is provided in Appendix A.

CHAPTER 4

Physical models for transient response of distributed *rlc* interconnects

4.1 Introduction

As clock frequencies increase into the tens of gigahertz, on-chip inductive effects will increasingly influence propagation delay and signal reliability. Because magnetic fields are not localized, spurious voltage waveforms can be induced on an interconnect not only by co-planar lines but also by interconnects in parallel metal layers above and below the metal layer on which the interconnect is located. Co-planar power and ground lines mitigate the inductive effects by shielding the interconnect from other signal lines. This chapter focuses on models that describe transient waveforms on the state-of-the-art global interconnect structures that have one or two signal lines sandwiched between co-planar ground lines as in Figure 4.1 [53], [54].

Full-wave field solvers are used by industrial CAD groups to evaluate the impact of inductance on propagation delay and signal reliability. However, such simulation tools use computationally intensive algorithms leading to long design iteration times. Hence, there is a need for accurate analytical models that can calculate in a fraction of the time taken by complex field solvers. Being able to rapidly pinpoint interconnect

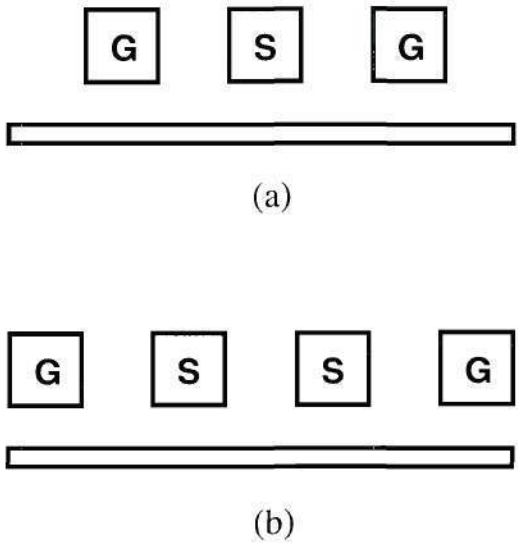


Figure 4.1: State-of-the-art global interconnect structures - (a) one and, (b) two signal interconnects (S) flanked by power/ground lines (G) and sandwiched between ground planes.

noise and signal reliability problems caused by inductance in a GSI multilevel interconnect network is extremely important for future multi-gigahertz systems. Direction of current research has focused on approximate models for the transient response of distributed *rlc* lines as has been discussed in Section 1.3. In this chapter a new complete physical model for the transient response of a high-speed global interconnect is rigorously derived. This work improves an earlier model by including a CMOS capacitive load termination to a distributed *rlc* line, which more accurately models on-chip and off-chip high-speed global wires that drive non-trivial capacitive loads. In addition to key physical insight, the new transient expressions derived here provide quick and accurate estimation of interconnect time delay and crosstalk, which is necessary for rapid design space exploration for global wiring networks in future GSI systems.

In Section 4.2, the work in [28] is significantly extended by rigorously solving the Laplace domain transfer function of a capacitively loaded distributed *rlc* line with a step input (Figure 4.2). It has been claimed that the effects of the output parasitic capacitance of the driver (C_p) is non-negligible [34]. HSPICE simulations in Figure 4.3 for an interconnect with a large driver show that neglecting C_p causes only a 5% error, whereas neglecting both C_p and the driver input capacitance (C_L) causes a 14% error in the calculation of the 50% rise time. In general, the percentage error caused by neglecting both C_p and C_L is about three times that caused by neglecting C_p alone, for different driver sizes. One of the key advantages of the increasingly prevalent SOI technology is the reduced output parasitic capacitance [55], which implies that for future technology generations, the ratio of the output parasitic capacitance to the input gate capacitance will be smaller. Therefore, the output capacitance of the driver is neglected in this analysis.

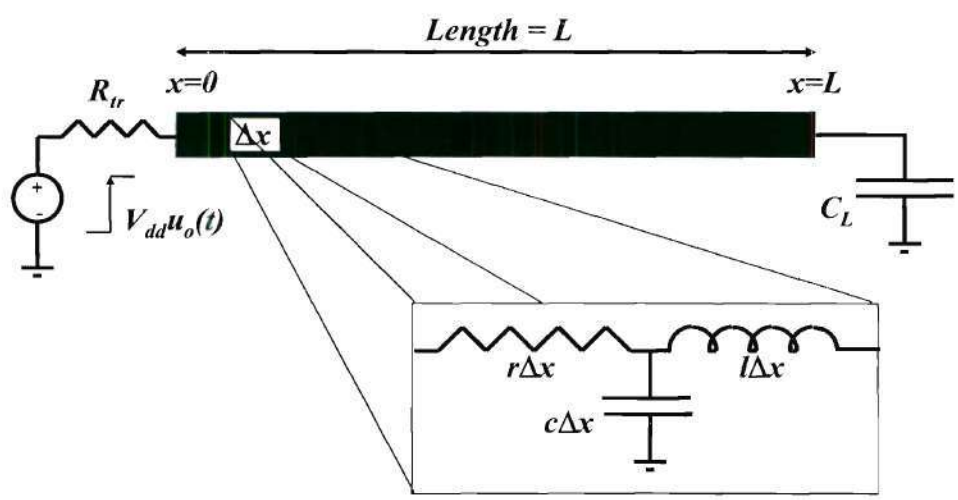


Figure 4.2: Distributed *rlc* interconnect of length L with a resistive driver (R_{tr}), load capacitance (C_L) and driven by a step input $V_{dd}u_o(t)$. The line resistance, capacitance and inductance per unit length are r, c and l respectively.

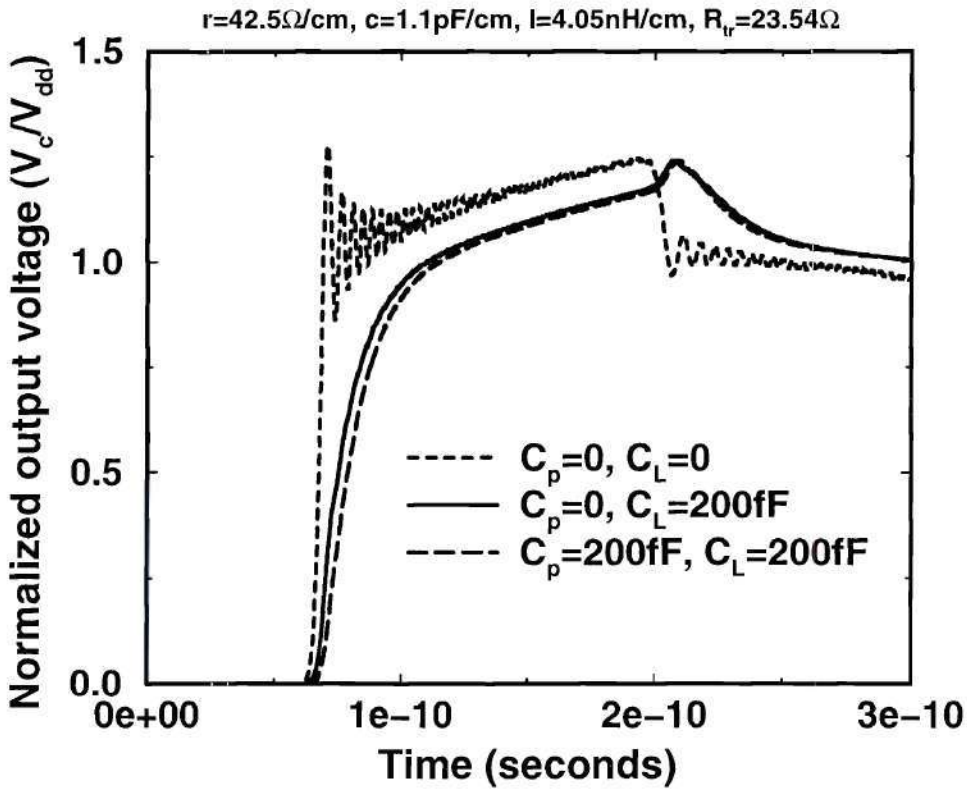


Figure 4.3: HSPICE simulations of the output transient response of a 1cm long distributed *rlc* interconnect. Neglecting the output capacitance (C_p) results in a 5% error, while neglecting both C_p and the input capacitance (C_L) results in a 14% error, for the 50% time delay.

A convergent series of modified Bessel functions is used to obtain a highly accurate and explicit solution for the transient response. This transient model is extended for a ramp input, with a finite rise time, in Section 4.3. The step-input transient response is used to derive models for crosstalk in two-coupled lines in Section 4.4. All analytical models are verified against HSPICE simulations and are demonstrated to have a high degree of accuracy. These solutions are used to develop unified models for time delay, repeater insertion and crosstalk in *rc* and *rlc* interconnects in Chapter 5.

4.2 Single line transient model with step input response

The voltage at the end of the line of length L, $V_{fin}(L,s)$, is described in the Laplace domain as [28]

$$V_{fin}(x=L,s) = (1+\Gamma_L)\sum_{n=0}^{g}\left[\Gamma_L^n\Gamma_S^n V_{inf}\left((2n+1)\,L,s\right)\right], \tag{4.1}$$

where $g = \left\langle 0.5\left(\frac{t}{L\sqrt{lc}}+1.0\right)\right\rangle - 1.0$ is the number of reflections, $\langle\Re\rangle$ being the notation for the decimal truncation of any real number $\Re$ (e.g. $\langle 2.7\rangle = 2$). $V_{inf}(x,s)$ is the Laplace domain voltage at x for a distributed rlc line of infinite length given by

$$V_{inf}(x,s) = \frac{V_{dd}}{s}\frac{Z_o\sqrt{(s+r/l)/s}}{Z_o\sqrt{(s+r/l)/s}+R_{tr}}e^{-x\sqrt{lc}\sqrt{s(s+r/l)}}, \tag{4.2}$$

and Γ_L and Γ_S are the reflection coefficients at the load and source ends respectively, defined as :

$$\Gamma_L = \frac{Z_L - Z(s)}{Z_L + Z(s)} = \frac{1 - sC_L Z_o\sqrt{1+\frac{r}{sl}}}{1 + sC_L Z_o\sqrt{1+\frac{r}{sl}}} \tag{4.3}$$

and

$$\Gamma_S = \frac{Z_S - Z(s)}{Z_S + Z(s)} = \frac{R_{tr} - Z_o\sqrt{1+\frac{r}{sl}}}{R_{tr} + Z_o\sqrt{1+\frac{r}{sl}}}, \tag{4.4}$$

where $Z_o = \sqrt{l/c}$ is the characteristic impedance of the line, $Z(s) = Z_o\sqrt{1+\frac{r}{sl}}$, $Z_L = \frac{1}{sC_L}$ and $Z_S = R_{tr}$. Equation (4.1) can be rewritten as

$$V_{fin}(x=L,s) = V_1(s) + V_2(s), \tag{4.5}$$

where $V_1(s)$ is the first reflection ($n=0$) term given by

$$V_1(s) = (1+\Gamma_L)\,V_{inf}\left(L,s\right), \tag{4.6}$$

and $V_2(s)$ is the second and higher order reflections ($n \geq 1$) term given by

$$V_2(s) = \sum_{n=1}^{g}\left[\left(\Gamma_L^n\Gamma_S^n + \Gamma_L^{n+1}\Gamma_S^n\right)V_{inf}\left((2n+1)\,L,s\right)\right]. \tag{4.7}$$

The time domain equivalent expressions of $V_1(s)$ and $V_2(s)$ are derived in the following sections. First, an important inverse Laplace transform identity is derived in Section 4.2.1. This identity is used to derive expressions for $V_1(t)$ and $V_2(t)$ in Sections 4.2.2 and 4.2.3 respectively. The conditions for the convergence of the series summations and resulting comprehensive time domain model are presented in Section 4.2.4. The comprehensive time domain model is validated against HSPICE simulations for various line parameters, demonstrating the accuracy of the model.

4.2.1 Derivation of an inverse Laplace transform identity

The following inverse Laplace transform identities are given in [56]:

$$L^{-1}\left[\frac{\left(s-\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}}\right]=I_n\left(\sigma t\right) \tag{4.8}$$

and

$$L^{-1}\left[\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-n}\frac{e^{-b\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\right]=\left(\frac{t-b}{t+b}\right)^{\frac{n}{2}}I_n\left(\sigma\sqrt{t^2-b^2}\right)u_o\left(t-b\right),\quad n\geq 0, \tag{4.9}$$

where $I_n(-)$ is the modified Bessel function of integral order n. In order to derive the time domain response of the distributed *rlc* interconnect, it is necessary to derive the following inverse Laplace transform identity which is a slight variation of (4.9) :

$$L^{-1}\left[\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{n}\frac{e^{-b\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\right]=?,\quad \text{for } n>0. \tag{4.10}$$

Bessel functions have the following property :

$$I_n\left(x\right)=i^{-n}J_n\left(ix\right), \tag{4.11}$$

where $J_n(-)$ is the Bessel function of integral order n. Using (4.11) and the series expansion for the Bessel function, one can write

$$I_n\left(\sigma t\right)=i^{-n}J_n\left(i\sigma t\right)=\sum_{j=0}^{\infty}\left[\frac{\sigma^{2j+n}t^{2j+n}}{2^{2j+n}j!\left(n+j\right)!}\right]. \tag{4.12}$$

The Laplace transform of (4.12) is

$$L\left[I_n\left(\sigma t\right)\right] = \frac{1}{\sigma} I_n\left(\frac{s}{\sigma}\right) = \sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n} j!\,(n+j)!}\frac{\sigma^{2j+n}}{s^{2j+n+1}}\right]. \tag{4.13}$$

From (4.8) and (4.13),

$$\frac{\left(s-\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} = \sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n} j!\,(j+n)!}\frac{\sigma^{2j+n}}{s^{2j+n+1}}\right]. \tag{4.14}$$

However,

$$\frac{\left(s-\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} = \frac{\left(s-\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} \times \frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\left(s+\sqrt{s^2-\sigma^2}\right)^n}. \tag{4.15}$$

Therefore,

$$\frac{\left(s-\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} = \frac{\left(s+\sqrt{s^2-\sigma^2}\right)^{-n}}{\sigma^{-n}\sqrt{s^2-\sigma^2}} = \sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n} j!\,(j+n)!}\frac{\sigma^{2j+n}}{s^{2j+n+1}}\right]. \tag{4.16}$$

Using simple algebra, one can write

$$\frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} = \frac{s^n\left(1+\sqrt{1-\frac{\sigma^2}{s^2}}\right)^n}{s\sigma^n\sqrt{1-\frac{\sigma^2}{s^2}}} = \frac{s^{n-1}\left(1+\sqrt{1-\frac{\sigma^2}{s^2}}\right)^n}{\sigma^n\sqrt{1-\frac{\sigma^2}{s^2}}}. \tag{4.17}$$

Let

$$z = \frac{\sigma^2}{s^2}. \tag{4.18}$$

Therefore, (4.17) can be written as

$$\frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} = \frac{\sigma^{-1}}{z^{\frac{n-1}{2}}}\frac{\left(1+\sqrt{1-z}\right)^n}{\sqrt{1-z}}. \tag{4.19}$$

Let

$$W = \frac{\left(1+\sqrt{1-z}\right)^n}{\sqrt{1-z}}. \tag{4.20}$$

Substituting this in (4.19) gives,

$$\frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} = \frac{\sigma^{-1}}{z^{\frac{n-1}{2}}}W. \tag{4.21}$$

Applying the Binomial theorem, one gets

$$W = \sum_{r=0}^{n} \left[\begin{pmatrix} n \\ r \end{pmatrix} \left(\sqrt{1-z}\right)^{r-1} \right]. \tag{4.22}$$

The coefficient in (4.22) can be written as

$$\begin{pmatrix} n \\ r \end{pmatrix} = \frac{\Gamma(n+1)}{\Gamma(r+1)\Gamma(n-r+1)}, \tag{4.23}$$

where the Gamma function is defined for real x as

$$\Gamma(x) = (x-1)(x-2)(x-3).... \tag{4.24}$$

For integral values of x, the Gamma function reduces to

$$\Gamma(x) = (x-1)! \tag{4.25}$$

and hence the coefficient in (4.22) becomes

$$\begin{pmatrix} n \\ r \end{pmatrix} = \frac{n!}{r!(n-r)!}. \tag{4.26}$$

The series summation is separated into odd and even parts: in (4.22), when r is even, let $r = 2m$; when r is odd, let $r = 2l+1$. Therefore, one can break (4.22) into the even and odd parts (w.r.t to r) thus :

$$W = \sum_{m=0}^{\langle \frac{n}{2} \rangle} \left[\begin{pmatrix} n \\ 2m \end{pmatrix} (1-z)^{\frac{2m-1}{2}} \right] + \sum_{l=0}^{\langle \frac{n}{2} \rangle} \left[\begin{pmatrix} n \\ 2l+1 \end{pmatrix} (1-z)^{l} \right], \tag{4.27}$$

where the notation $\langle \Re \rangle$ denotes the greatest integer less than or equal to $\Re$. Applying the binomial theorem again, one gets

$$\begin{aligned} W &= \sum_{m=0}^{\langle \frac{n}{2} \rangle} \left[\begin{pmatrix} n \\ 2m \end{pmatrix} \sum_{k=0}^{\infty} \left[\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix} (-1)^k z^k \right] \right] \\ &+ \sum_{l=0}^{\langle \frac{n}{2} \rangle} \left[\begin{pmatrix} n \\ 2l+1 \end{pmatrix} \sum_{k=0}^{l} \left[\begin{pmatrix} l \\ k \end{pmatrix} (-1)^k z^k \right] \right] \end{aligned} \tag{4.28}$$

or,

$$\begin{aligned} W &= \sum_{k=0}^{\infty}\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{k}(-1)^k z^k\right] \\ &+ \sum_{k=0}^{l}\sum_{l=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2l+1}\binom{l}{k}(-1)^k z^k\right]. \end{aligned} \tag{4.29}$$

The following two identities can be proved by mathematical induction :

$$\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{k}\right]-\sum_{l=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2l+1}\binom{l}{k}\right]=0,\ \text{for } 0\le k\le\left\langle\frac{n}{2}\right\rangle, \tag{4.30}$$

and

$$\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{k}\right]=0,\ \text{for } \left\langle\frac{n}{2}\right\rangle+1\le k\le n-1. \tag{4.31}$$

Using identities (4.30) and (4.31) in (4.29), one gets,

$$\begin{aligned} W &= 2\sum_{k=0}^{\langle\frac{n}{2}\rangle}\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{k}(-1)^k z^k\right] \\ &+ \sum_{k=n}^{\infty}\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{k}(-1)^k z^k\right]. \end{aligned} \tag{4.32}$$

Let $j=k-n$. Therefore, $k=j+n$. Substituting this in the second term of (4.32) one gets,

$$\begin{aligned} W &= 2\sum_{k=0}^{\langle\frac{n}{2}\rangle}\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{k}(-1)^k z^k\right] \\ &+ \sum_{j=0}^{\infty}\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{j+n}(-1)^{j+n} z^{j+n}\right]. \end{aligned} \tag{4.33}$$

The following identity can also be proved by mathematical induction :

$$\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\binom{n}{2m}\binom{\frac{2m-1}{2}}{j+n}\right]=\frac{(-1)^{j+n}(2j+n)!}{2^{2j+n}j!\,(j+n)!},\ \text{for } 0\le j\le\infty. \tag{4.34}$$

Using identity (4.34), one can rewrite (4.33) as :

$$W = 2\sum_{k=0}^{\langle \frac{n}{2} \rangle}\sum_{m=0}^{\langle \frac{n}{2} \rangle}\left[\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k z^k\right] + \sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n}j!\,(j+n)!}z^{j+n}\right]. \quad (4.35)$$

Using (4.35) and the definition of W in (4.20), one can rewrite (4.19) as

$$\frac{\sigma^{-1}\left(1+\sqrt{1-z}\right)^n}{z^{\frac{n-1}{2}}\sqrt{1-z}} = \frac{\sigma^{-1}}{z^{\frac{n-1}{2}}}\left(\begin{array}{l} 2\sum_{k=0}^{\langle \frac{n}{2} \rangle}\sum_{m=0}^{\langle \frac{n}{2} \rangle}\left[\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k z^k\right] \\ +\sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n}j!(j+n)!}z^{j+n}\right] \end{array}\right) \quad (4.36)$$

or,

$$\begin{aligned} \frac{\sigma^{-1}\left(1+\sqrt{1-z}\right)^n}{z^{\frac{n-1}{2}}\sqrt{1-z}} &= 2\sum_{k=0}^{\langle \frac{n}{2} \rangle}\sum_{m=0}^{\langle \frac{n}{2} \rangle}\left[\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k \sigma^{-1} z^{k+\frac{-n+1}{2}}\right] \\ &+ \sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n}j!\,(j+n)!}\sigma^{-1}z^{j+\frac{n+1}{2}}\right]. \end{aligned} \quad (4.37)$$

Using (4.18) and (4.19), one can rewrite (4.37) as:

$$\begin{aligned} \frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} &= 2\sum_{k=0}^{\langle \frac{n}{2} \rangle}\sum_{m=0}^{\langle \frac{n}{2} \rangle}\left[\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k \frac{\sigma^{2k-n}}{s^{2k-n+1}}\right] \\ &+ \sum_{j=0}^{\infty}\left[\frac{(2j+n)!}{2^{2j+n}j!\,(j+n)!}\frac{\sigma^{2j+n}}{s^{2j+n+1}}\right]. \end{aligned} \quad (4.38)$$

Substituting (4.16) into (4.38), one can write,

$$\begin{aligned} \frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}} &= 2\sum_{k=0}^{\langle \frac{n}{2} \rangle}\sum_{m=0}^{\langle \frac{n}{2} \rangle}\left[\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k \sigma^{2k-n}s^{n-2k-1}\right] \\ &+ \left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-n}\frac{1}{\sqrt{s^2-\sigma^2}}. \end{aligned} \quad (4.39)$$

Multiplying each term by $e^{-b\sqrt{s^2-\sigma^2}}$, one gets,

$$\frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}}e^{-b\sqrt{s^2-\sigma^2}} = 2\sum_{k=0}^{\langle \frac{n}{2} \rangle}\sum_{m=0}^{\langle \frac{n}{2} \rangle}\left[\begin{array}{l} \begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k \\ \times\sigma^{2k-n}s^{n-2k-1}e^{-b\sqrt{s^2-\sigma^2}} \end{array}\right]$$

$$+ \left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-n} \frac{e^{-b\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}. \tag{4.40}$$

The inverse Laplace transform of the second term on the right side of (4.40) is given by (4.9). To find the inverse Laplace transform of the first term on the right side of (4.40), one needs an expression for the inverse Laplace transform of

$$s^x e^{-b\sqrt{s^2-\sigma^2}}, \quad x \geq -1, \; x \in I. \tag{4.41}$$

It can be observed that the exponents of s in the first term of (4.40) will always be greater than or equal to -1. In fact, the exponent of s is -1 only when n is even and $k = \left\langle \frac{n}{2} \right\rangle$. However, for all even values of n,

$$\sum_{x=0}^{\left\langle \frac{n}{2} \right\rangle} \left[\begin{pmatrix} n \\ 2x \end{pmatrix} \begin{pmatrix} \frac{2x-1}{2} \\ \left\langle \frac{n}{2} \right\rangle \end{pmatrix} \right] = 0 \; . \tag{4.42}$$

Therefore, the coefficient of terms containing s^{-1} is always zero. This implies, that only non-negative powers of s are left in the first term of (4.40). Therefore, an expression is needed for the inverse Laplace transform of (4.41) for $x \geq 0$ only.

To do that, start with identity 3.2.48 on page 251 of [56] which is :

$$L^{-1}\left[e^{-bs} - e^{-b\sqrt{s^2+\sigma^2}}\right] = \frac{\sigma b J_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}} u_o\left(t-b\right), \tag{4.43}$$

where $J_n(t)$ is the Bessel function of order n. Using the linearity property of Laplace transforms, one gets

$$L^{-1}\left[e^{-bs} - e^{-b\sqrt{s^2+\sigma^2}}\right] = L^{-1}\left[e^{-bs}\right] - L^{-1}\left[e^{-b\sqrt{s^2+\sigma^2}}\right] \tag{4.44}$$

Therefore,

$$L^{-1}\left[e^{-b\sqrt{s^2+\sigma^2}}\right] = -\frac{\sigma b J_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}} u_o\left(t-b\right) + L^{-1}\left[e^{-bs}\right] \tag{4.45}$$

or,

$$L^{-1}\left[e^{-b\sqrt{s^2+\sigma^2}}\right] = -\frac{\sigma b J_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)+\delta\left(t-b\right). \tag{4.46}$$

Replacing σ by $i\sigma$, where i is the complex operator $i=\sqrt{-1}$, (4.46) can be written as

$$L^{-1}\left[e^{-b\sqrt{s^2-\sigma^2}}\right] = -\frac{i\sigma b J_1\left[i\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)+\delta\left(t-b\right) \tag{4.47}$$

Using (4.11), (4.47) can be written as

$$L^{-1}\left[e^{-b\sqrt{s^2-\sigma^2}}\right] = -\frac{i^2\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)+\delta\left(t-b\right) \tag{4.48}$$

or,

$$L^{-1}\left[e^{-b\sqrt{s^2-\sigma^2}}\right] = \frac{\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)+\delta\left(t-b\right). \tag{4.49}$$

Laplace transforms have the following property :

$$\text{If } \; L^{-1}\left[F\left(s\right)\right]=f\left(t\right), \;\; \text{then } \; L^{-1}\left[sF\left(s\right)-f\left(0\right)\right]=f'\left(t\right). \tag{4.50}$$

In (4.49), let

$$F\left(s\right)=e^{-b\sqrt{s^2-\sigma^2}}, \;\; \text{and} \;\; f\left(t\right)=\frac{\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)+\delta\left(t-b\right). \tag{4.51}$$

Since $f\left(0\right)=0$, using (4.50) and (4.51) in (4.49), one gets

$$L^{-1}\left[se^{-b\sqrt{s^2-\sigma^2}}-0\right]=\frac{d}{dt}\left[\frac{\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)+\delta\left(t-b\right)\right]. \tag{4.52}$$

Now, by applying property (4.11) again, one gets

$$\frac{d}{dt}\left[\frac{\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)\right]=\frac{d}{dt}\left[\frac{\sigma b}{i}\frac{J_1\left[i\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)\right], \tag{4.53}$$

or

$$\begin{aligned}&\frac{d}{dt}\left[\frac{\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)\right]\\ &=\frac{d}{dt}\left[i\sigma\sqrt{t^2-b^2}J_1\left[i\sigma\sqrt{t^2-b^2}\right]\frac{b}{i^2\left(t^2-b^2\right)}u_o\left(t-b\right)\right].\end{aligned} \tag{4.54}$$

Following are two useful properties of Bessel functions :

$$\frac{d}{dt}\left[t^n J_n(t)\right] = t^n J_{n-1}(t)\,, \tag{4.55}$$

and

$$J_{n+1}(t) = \frac{2n}{t} J_n(t) - J_{n-1}(t)\,. \tag{4.56}$$

Using (4.55), the right hand side of (4.54) can be written as

$$\begin{aligned}
&\frac{d}{dt}\left[i\sigma\sqrt{t^2-b^2}J_1\left[i\sigma\sqrt{t^2-b^2}\right]\frac{b}{i^2\left(t^2-b^2\right)}u_o(t-b)\right] \\
&= i\sigma\sqrt{t^2-b^2}J_0\left[i\sigma\sqrt{t^2-b^2}\right]\frac{i\sigma t}{\sqrt{t^2-b^2}}\frac{b}{i^2\left(t^2-b^2\right)}u_o(t-b) \\
&+ i\sigma\sqrt{t^2-b^2}J_1\left[i\sigma\sqrt{t^2-b^2}\right]\frac{b}{i^2}\frac{(-1)\,2t}{\left(t^2-b^2\right)^2}u_o(t-b) + \delta(t-b)
\end{aligned} \tag{4.57}$$

i.e.

$$\begin{aligned}
&= \sigma^2 b\frac{t}{\left(t^2-b^2\right)}J_0\left[i\sigma\sqrt{t^2-b^2}\right]u_o(t-b) \\
&+ i\sigma b\frac{2t\sqrt{t^2-b^2}}{\left(t^2-b^2\right)^2}J_1\left[i\sigma\sqrt{t^2-b^2}\right]u_o(t-b) + \delta(t-b)
\end{aligned} \tag{4.58}$$

which can be written as

$$\begin{aligned}
&= \sigma^2 b\frac{t}{t^2-b^2}\left[J_0\left[i\sigma\sqrt{t^2-b^2}\right] + \frac{2i}{\sigma\sqrt{t^2-b^2}}J_1\left[i\sigma\sqrt{t^2-b^2}\right]\right]u_o(t-b) + \delta(t-b) \\
&= \sigma^2 b\frac{t}{t^2-b^2}\left[J_0\left[i\sigma\sqrt{t^2-b^2}\right] - \frac{2}{\sigma i\sqrt{t^2-b^2}}J_1\left[i\sigma\sqrt{t^2-b^2}\right]\right]u_o(t-b) + \delta(t-b) \\
&= \sigma^2 b\frac{-t}{t^2-b^2}\left[\frac{2}{i\sigma\sqrt{t^2-b^2}}J_1\left[i\sigma\sqrt{t^2-b^2}\right] - J_0\left[i\sigma\sqrt{t^2-b^2}\right]\right]u_o(t-b) + \delta(t-b)\,.
\end{aligned} \tag{4.59}$$

Finally, using (4.56), one gets,

$$= -\sigma^2 b\frac{t}{t^2-b^2}J_2\left[i\sigma\sqrt{t^2-b^2}\right]u_o(t-b) + \delta(t-b)\,. \tag{4.60}$$

Using (4.11) on (4.60), (4.54) can be written as

$$\frac{d}{dt}\left[\frac{\sigma b I_1\left[\sigma\sqrt{t^2-b^2}\right]}{\sqrt{t^2-b^2}}u_o\left(t-b\right)\right]=\sigma^2 b\frac{t}{t^2-b^2}I_2\left[\sigma\sqrt{t^2-b^2}\right]u_o\left(t-b\right)+\delta\left(t-b\right). \tag{4.61}$$

Therefore, (4.52) can be written as

$$L^{-1}\left[se^{-b\sqrt{s^2-\sigma^2}}\right]=\sigma^2 b\frac{t}{t^2-b^2}I_2\left[\sigma\sqrt{t^2-b^2}\right]u_o\left(t-b\right)+\delta\left(t-b\right)+\delta'\left(t-b\right). \tag{4.62}$$

Just as (4.62) was derived from (4.49), similarly the inverse Laplace transform of (4.41) for larger values of m can be calculated by repeatedly using (4.50), (4.11), (4.55) and (4.56) :

$$L^{-1}\left[s^2e^{-b\sqrt{s^2-\sigma^2}}\right]=\left[\frac{\sigma^3 bt^2}{\left(\sqrt{t^2-b^2}\right)^3}I_3\left[-\right]+\frac{\sigma^2 b}{\left(\sqrt{t^2-b^2}\right)^2}I_2\left[-\right]\right]u_o\left(t-b\right)+\delta\left(t-b\right)+\ldots, \tag{4.63}$$

where $I_n\left[-\right]=I_n\left[\sigma\sqrt{t^2-b^2}\right]$, and the ellipsis after $\delta\left(t-b\right)$ indicates higher order derivatives of the delta function. Similarly,

$$L^{-1}\left[s^3e^{-b\sqrt{s^2-\sigma^2}}\right]=\left[\frac{\sigma^4 bt^3}{\left(\sqrt{t^2-b^2}\right)^4}I_4\left[-\right]+\frac{3\sigma^3 bt}{\left(\sqrt{t^2-b^2}\right)^3}I_3\left[-\right]\right]u_o\left(t-b\right)+\delta\left(t-b\right)+\ldots \tag{4.64}$$

$$L^{-1}\left[s^4e^{-b\sqrt{s^2-\sigma^2}}\right]=\left[\begin{array}{l}\frac{\sigma^5 bt^4}{\left(\sqrt{t^2-b^2}\right)^5}I_5\left[-\right]+\frac{6\sigma^4 bt^2}{\left(\sqrt{t^2-b^2}\right)^4}I_4\left[-\right]\\+\frac{3\sigma^3 b}{\left(\sqrt{t^2-b^2}\right)^3}I_3\left[-\right]\end{array}\right]u_o\left(t-b\right)+\delta\left(t-b\right)+\ldots \tag{4.65}$$

$$L^{-1}\left[s^5e^{-b\sqrt{s^2-\sigma^2}}\right]=\left[\begin{array}{l}\frac{\sigma^6 bt^5}{\left(\sqrt{t^2-b^2}\right)^6}I_6\left[-\right]+\frac{10\sigma^5 bt^3}{\left(\sqrt{t^2-b^2}\right)^5}I_5\left[-\right]\\+\frac{15\sigma^4 bt}{\left(\sqrt{t^2-b^2}\right)^3}I_4\left[-\right]\end{array}\right]u_o\left(t-b\right)+\delta\left(t-b\right)+\ldots \tag{4.66}$$

$$L^{-1}\left[s^6e^{-b\sqrt{s^2-\sigma^2}}\right]=\left[\begin{array}{l}\frac{\sigma^7 bt^6}{\left(\sqrt{t^2-b^2}\right)^7}I_7\left[-\right]+\frac{15\sigma^6 bt^4}{\left(\sqrt{t^2-b^2}\right)^6}I_6\left[-\right]\\+\frac{45\sigma^5 bt^2}{\left(\sqrt{t^2-b^2}\right)^5}I_5\left[-\right]+\frac{15\sigma^4 b}{\left(\sqrt{t^2-b^2}\right)^4}I_4\left[-\right]\end{array}\right]u_o\left(t-b\right)+\delta\left(t-b\right)+\ldots\ . \tag{4.67}$$

By analyzing equations (4.63)-(4.67), a pattern can be identified for the coefficients of the terms in the Laplace transform relations. Therefore, the general form of the inverse Laplace transform of (4.41) can be written as :

$$\begin{aligned} L^{-1}\left[s^x e^{-b\sqrt{s^2-\sigma^2}}\right] &= \sum_{y=1}^{\left\langle \frac{x}{2}\right\rangle+1}\left[C_{x,y}\frac{\sigma^{x-y+2}bt^{x-2y+2}}{\left(\sqrt{t^2-b^2}\right)^{x-y+2}}I_{x-y+2}\left(\sigma\sqrt{t^2-b^2}\right)\right]u_o\left(t-b\right) \\ &+ \delta\left(t-b\right)+...+\delta^x\left(t-b\right), \quad x\geq 0, \end{aligned} \tag{4.68}$$

where the coefficient $C_{x,y}$ is defined recursively as :

$$\begin{aligned} &C_{x,y}=0 && \text{if } y=0,\ \forall\, x \\ &C_{x,y}=1 && \text{if } x=0,\ y=1 \\ &C_{x,y}=0 && \text{if } x=0,\ y>1 \\ &C_{x,y}=\left(x-2y+3\right)C_{x-1,y-1}+C_{x-1,y} && \forall \text{ other } x \text{ and } y \end{aligned} \tag{4.69}$$

Using (4.9) and (4.68), the inverse Laplace transform of (4.40) can be written as :

$$\begin{aligned} &L^{-1}\left[\frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}}e^{-b\sqrt{s^2-\sigma^2}}\right]= \\ &2\sum_{k=0}^{\left\langle\frac{n}{2}\right\rangle}\sum_{m=0}^{\left\langle\frac{n}{2}\right\rangle}\left[\begin{array}{l}\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k\sigma^{2k-n} \\ \times\sum_{y=1}^{\left\langle\frac{n-2k-1}{2}\right\rangle+1}\left[\begin{array}{l} C_{n-2k-1,y}\frac{\sigma^{n-2k-y+1}bt^{n-2k-2y+1}}{\left(\sqrt{t^2-b^2}\right)^{n-2k-y+1}} \\ \times I_{n-2k-y+1}\left(\sigma\sqrt{t^2-b^2}\right)\end{array}\right]\end{array}\right]u_o\left(t-b\right) \\ &+2\sum_{k=0}^{\left\langle\frac{n}{2}\right\rangle}\sum_{m=0}^{\left\langle\frac{n}{2}\right\rangle}\left[\begin{array}{l}\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k\sigma^{2k-n} \\ \times\left(\delta\left(t-b\right)+\delta^1\left(t-b\right)+...+\delta^m\left(t-b\right)\right)\end{array}\right] \\ &+\left(\frac{t-b}{t+b}\right)^{\frac{n}{2}}I_n\left(\sigma\sqrt{t^2-b^2}\right)u_o\left(t-b\right). \end{aligned} \tag{4.70}$$

Assuming that for time $t\neq b$, the delta function and all its derivatives vanish, one

gets,

$$L^{-1}\left[\frac{\left(s+\sqrt{s^2-\sigma^2}\right)^n}{\sigma^n\sqrt{s^2-\sigma^2}}e^{-b\sqrt{s^2-\sigma^2}}\right]=$$

$$2\sum_{k=0}^{\langle\frac{n}{2}\rangle}\sum_{m=0}^{\langle\frac{n}{2}\rangle}\left[\begin{array}{l}\begin{pmatrix} n \\ 2m \end{pmatrix}\begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix}(-1)^k\sigma^{2k-n} \\ \times\sum_{y=1}^{\langle\frac{n-2k-1}{2}\rangle+1}\left[\begin{array}{l} C_{n-2k-1,y}\frac{\sigma^{n-2k-y+1}bt^{n-2k-2y+1}}{\left(\sqrt{t^2-b^2}\right)^{n-2k-y+1}} \\ \times I_{n-2k-y+1}\left(\sigma\sqrt{t^2-b^2}\right)\end{array}\right]\end{array}\right]u_o(t-b)$$

$$+\left(\frac{t-b}{t+b}\right)^{\frac{n}{2}}I_n\left(\sigma\sqrt{t^2-b^2}\right)u_o(t-b),\quad \text{for } n>0 \tag{4.71}$$

where the coefficients $C_{x,y}$ are recursively defined in (4.69).

4.2.2 First reflection

Now, consider the Laplace domain output response of a distributed *rlc* interconnect, given by (4.1). Applying the transformation [57],

$$s\rightarrow s-\sigma \tag{4.72}$$

and the substitution,

$$s=\frac{\sigma}{2}\left(a+\frac{1}{a}\right) \tag{4.73}$$

where

$$\sigma=\frac{r}{2l}\quad\text{and}\quad a=\frac{1}{\sigma}\left(s+\sqrt{s^2-\sigma^2}\right), \tag{4.74}$$

the reflection coefficients in (4.3) and (4.4) can be rewritten as

$$\Gamma_L^{'}=\frac{1-\frac{\sigma}{2a}\left(a^2-1\right)C_LZ_o}{1+\frac{\sigma}{2a}\left(a^2-1\right)C_LZ_o}, \tag{4.75}$$

and

$$\Gamma_S^{'}=\frac{R_{tr}-Z_o\left(\frac{a+1}{a-1}\right)}{R_{tr}+Z_o\left(\frac{a+1}{a-1}\right)}. \tag{4.76}$$

Using the series expansions :

$$\frac{1}{x+R} = \sum_{i=0}^{\infty} \left[(-R)^i x^{-i-1} \right], \quad \text{if } \left| \frac{x}{R} \right| > 1 \tag{4.77}$$

and

$$\frac{1}{x+R} = \sum_{i=0}^{\infty} \left[R^{-i-1} (-x)^i \right], \quad \text{if } \left| \frac{x}{R} \right| < 1, \tag{4.78}$$

(4.2) can be rewritten as

$$V'_{inf}(x,s) = V_{dd} \frac{Z_o}{Z_o + R_{tr}} \frac{e^{-x\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}} \left\{ 1 + \frac{1}{1-G_s} \sum_{u=1}^{\infty} \left[a^{-u} \left(4 - (1+G_s)^2 G_s^{u-1} \right) \right] \right\} \tag{4.79}$$

where

$$G_s = \frac{R_{tr} - Z_o}{R_{tr} + Z_o}. \tag{4.80}$$

Using (4.75), (4.76) and (4.79), (4.5) can be rewritten as

$$V'_{fin}(x = L, s) = V'_1(s) + V'_2(s), \tag{4.81}$$

with

$$V'_1(s) = \left(1 + \Gamma'_L \right) V'_{inf}(L, s) \tag{4.82}$$

and

$$V'_2(s) = \sum_{n=1}^{g} \left[\left(\Gamma'^n_L \Gamma'^n_S + \Gamma'^{n+1}_L \Gamma'^n_S \right) V'_{inf}\left((2n+1) L, s \right) \right], \tag{4.83}$$

where $V'_1(s)$ is the first reflection term ($n = 0$) and $V'_2(s)$ contains higher order reflections ($n \geq 1$).

The time domain expression for the first reflection term (4.82) is derived here. The time domain expression for the higher order reflection terms (4.83) is derived in Section 4.2.3.

Using (4.75) and the theory of partial fractions, the coefficient of $V'_{inf}(L, s)$ in (4.82) can be written as

$$1 + \Gamma'_L = \frac{P}{a+M} + \frac{Q}{a+N} \tag{4.84}$$

where

$$P = \frac{B\left(B+\sqrt{B^2+4}\right)}{\sqrt{B^2+4}}, Q = \frac{-B\left(B-\sqrt{B^2+4}\right)}{\sqrt{B^2+4}}$$
$$M = \frac{B+\sqrt{B^2+4}}{2}, N = \frac{B-\sqrt{B^2+4}}{2}, B = \frac{2}{\sigma C_L Z_o}. \tag{4.85}$$

Since $B > 0, \Rightarrow M > 1, \& |N| < 1$.

Initially, (4.77) is used to expand (4.84), assuming $\left|\frac{a}{M}\right| > 1$ and $\left|\frac{a}{N}\right| > 1$; when the series expansion no longer converges i.e. $\left|\frac{a}{M}\right| < 1$, then (4.78) will be used for the first term of (4.84). The two different expansions are differentiated using the subscripts A and B for $V_1(t)$. The convergence criteria which will determine the usability of the two different expansions will be provided in Section 4.2.4.

Using (4.77), (4.84) can be written as

$$1 + \Gamma_L^{'} = P\sum_{i=0}^{\infty}\left[(-M)^i a^{-i-1}\right] + Q\sum_{i=0}^{\infty}\left[(-N)^i a^{-i-1}\right]. \tag{4.86}$$

Using (4.74), (4.79) and (4.86), (4.82) can be written as

$$\begin{aligned} V_{1A}^{'}(s) &= V_{dd}\frac{Z_o}{Z_o + R_{tr}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}} \\ &\times \left\{ \begin{array}{l} \left(\begin{array}{l} P\sum_{i=0}^{\infty}\left[(-M)^i\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-i-1}\right] \\ +Q\sum_{i=0}^{\infty}\left[(-N)^i\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-i-1}\right] \end{array} \right) \\ \times\left(1+\frac{1}{1-G_s}\sum_{u=1}^{\infty}\left[\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-u}\left(4-(1+G_s)^2 G_s^{u-1}\right)\right]\right) \end{array} \right\}. \end{aligned} \tag{4.87}$$

Using the inverse Laplace transform identity of (4.9) in (4.87), the time domain expression for $V_{1A}^{'}(t)$ can be written as

$$V_{1A}^{'}(t) = V_{dd}\frac{Z_o}{Z_o + R_{tr}}$$

$$\times \left\{ \begin{array}{l} P\sum_{i=0}^{\infty}\left[(-M)^{i}\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+1}{2}} I_{i+1}\left(\sigma\sqrt{t^{2}-L^{2}lc}\right)\right] \\ +Q\sum_{i=0}^{\infty}\left[(-N)^{i}\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+1}{2}} I_{i+1}\left(\sigma\sqrt{t^{2}-L^{2}lc}\right)\right] \\ +\frac{P}{1-G_s}\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}(-M)^{i}\left(4-(1+G_s)^{2}G_s^{u-1}\right) \\ \times\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}} I_{i+u+1}\left(\sigma\sqrt{t^{2}-L^{2}lc}\right)\end{array}\right] \\ +\frac{Q}{1-G_s}\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}(-N)^{i}\left(4-(1+G_s)^{2}G_s^{u-1}\right) \\ \times\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}} I_{i+u+1}\left(\sigma\sqrt{t^{2}-L^{2}lc}\right)\end{array}\right] \end{array} \right\}$$

$$\times \quad u_o\left(t-L\sqrt{lc}\right). \tag{4.88}$$

When (4.88) no longer converges, the first and second terms in (4.84) should be expanded using (4.78) and (4.77) respectively as

$$1+\Gamma_L' = P\sum_{i=0}^{\infty}\left[M^{-i-1}(-a)^{i}\right] + Q\sum_{i=0}^{\infty}\left[(-N)^{i}a^{-i-1}\right]. \tag{4.89}$$

Using (4.74), (4.79) and (4.89), (4.82) can be written as

$$\begin{aligned} V_{1B}'(s) &= V_{dd}\frac{Z_o}{Z_o+R_{tr}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}} \\ &\times \left\{ \begin{array}{l} \left(\begin{array}{l} P\sum_{i=0}^{\infty}\left[M^{-i-1}(-1)^{i}\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{i}\right] \\ +Q\sum_{i=0}^{\infty}\left[(-N)^{i}\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-i-1}\right] \end{array} \right) \\ \times\left(1+\frac{1}{1-G_s}\sum_{u=1}^{\infty}\left[\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-u}\left(4-(1+G_s)^{2}G_s^{u-1}\right)\right]\right) \end{array} \right\}. \end{aligned} \tag{4.90}$$

Using the inverse Laplace transform identities of (4.9) and (4.71), the time domain

expression for $V_{1B}^{'}(t)$ can be written as

$$
\begin{aligned}
V_{1B}^{'}(t) &= V_{dd}\tfrac{Z_o}{Z_o+R_{tr}} \\
&\times \left\{ \begin{array}{l}
P\sum\limits_{i=0}^{\infty}\left[\begin{array}{l} M^{-i-1}(-1)^i \\ \times\left(\begin{array}{l} 2\sum\limits_{k=0}^{\langle\frac{i}{2}\rangle}\sum\limits_{m=0}^{\langle\frac{i}{2}\rangle}\sum\limits_{y=1}^{\langle\frac{i-2k-1}{2}\rangle+1} \\ \left[\begin{array}{l}\binom{i}{2m}\binom{\frac{2m-1}{2}}{k}(-1)^k C_{i-2k-1,y}\sigma^{-y+1}L\sqrt{lc} \\ \times\frac{t^{i-2k-2y+1}}{\left(\sqrt{t^2-L^2lc}\right)^{i-2k-y+1}}I_{i-2k-y+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right] \\ +\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i}{2}}I_i\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right)\end{array}\right] \\
+Q\sum\limits_{i=0}^{\infty}\left[(-N)^i\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+1}{2}}I_{i+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\right] \\
+\frac{P}{1-G_s}\sum\limits_{i=0}^{\infty}\sum\limits_{u=1}^{\infty} \\
\left[\begin{array}{l} M^{-i-1}\left(4-(1+G_s)^2G_s^{u-1}\right)(-1)^i \\ \times\left(\begin{array}{l} 2\beta(i-u)\sum\limits_{k=0}^{\langle\frac{i-u}{2}\rangle}\sum\limits_{m=0}^{\langle\frac{i-u}{2}\rangle}\sum\limits_{y=1}^{\langle\frac{i-u-2k-1}{2}\rangle+1} \\ \left[\begin{array}{l}\binom{i-u}{2m}\binom{\frac{2m-1}{2}}{k}(-1)^k C_{i-u-2k-1,y}\sigma^{-y+1} \\ \times L\sqrt{lc}\frac{t^{i-u-2k-2y+1}}{\left(\sqrt{t^2-L^2lc}\right)^{i-u-2k-y+1}}I_{i-u-2k-y+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right] \\ +\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\left|\frac{i-u}{2}\right|}I_{|i-u|}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right)\end{array}\right] \\
+\frac{Q}{1-G_s}\sum\limits_{i=0}^{\infty}\sum\limits_{u=1}^{\infty}\left[(-N)^i D\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\right]
\end{array}\right\}, \\
&\times u_o\left(t-L\sqrt{lc}\right)
\end{aligned}
\tag{4.91}
$$

where

$$D = 4-(1+G_s)^2G_s^{u-1} \tag{4.92}$$

and

$$\beta(x) = \begin{cases} 0, \text{ if } x<0 \\ 1, \text{ if } x\geq 0 \end{cases}, \tag{4.93}$$

and the coefficients $C_{x,y}$ are recursively defined in (4.69). Accounting for the transformation used in (4.72), the time domain transient response for the first reflection can be written as

$$V_1(t) = e^{\frac{-rt}{2l}} \left(V_{1A}^{'}(t) \text{ or } V_{1B}^{'}(t) \right). \tag{4.94}$$

4.2.3 Higher order reflections

The time domain equivalent expressions for the higher order reflection terms in (4.83) is derived here. Using simple algebra, (4.75) can be rewritten as

$$\begin{aligned}
\Gamma_L^{'} &= \frac{2a - a^2\sigma C_L Z_o + \sigma C_L Z_o}{2a + a^2\sigma C_L Z_o - \sigma C_L Z_o} \\
&= \frac{\left(1 + \frac{\sigma C_L Z_o}{2a - a^2\sigma C_L Z_o}\right) \times (2a - a^2\sigma C_L Z_o)}{\left(1 - \frac{\sigma C_L Z_o}{2a + a^2\sigma C_L Z_o}\right) \times (2a + a^2\sigma C_L Z_o)} \\
&= \frac{\left(1 + \frac{\sigma C_L Z_o}{2a - a^2\sigma C_L Z_o}\right)}{\left(1 - \frac{\sigma C_L Z_o}{2a + a^2\sigma C_L Z_o}\right)} \times \frac{\left(\frac{2}{a\sigma C_L Z_o} - 1\right)}{\left(\frac{2}{a\sigma C_L Z_o} + 1\right)}.
\end{aligned} \tag{4.95}$$

Substituting for B from (4.85)

$$\Gamma_L^{'} = \frac{\left(1 + \frac{1}{aB - a^2}\right)}{\left(1 - \frac{1}{aB + a^2}\right)} \times \frac{(-1)\left(1 - \frac{B}{a}\right)}{\left(1 + \frac{B}{a}\right)}. \tag{4.96}$$

Similarly, (4.76) can be written as

$$\begin{aligned}
\Gamma_S^{'} &= \frac{R_{tr} - Z_o\left(\frac{a+1}{a-1}\right)}{R_{tr} + Z_o\left(\frac{a+1}{a-1}\right)} \\
&= \frac{aR_{tr} - R_{tr} - aZ_o - Z_o}{aR_{tr} - R_{tr} + aZ_o + Z_o} \\
&= \frac{a\frac{R_{tr} - Z_o}{R_{tr} + Z_o} - 1}{a - \frac{R_{tr} - Z_o}{R_{tr} + Z_o}}.
\end{aligned} \tag{4.97}$$

Subsituting for G_s from (4.80),

$$\Gamma_S^{'} = \frac{aG_s - 1}{a - G_s}, \tag{4.98}$$

or,

$$\Gamma_S' = G_s \left(\frac{1 - \frac{1}{aG_s}}{1 - \frac{G_s}{a}} \right). \tag{4.99}$$

The Binomial expansions given by

$$(x+y)^n = \sum_{r=0}^{n} \left[\begin{pmatrix} n \\ r \end{pmatrix} x^{n-r} y^r \right], \quad n \in I, \tag{4.100}$$

and the following series expansion,

$$\left(\frac{1+x}{1-y} \right)^n = \sum_{i=0}^{n} \sum_{j=0}^{\infty} \left[\frac{n\,(n-1+j)!}{i!j!\,(n-i)!} x^i y^j \right], \quad \text{for } n > 0,\ n \in I,\ |y| < 1 \tag{4.101}$$

will be useful for expanding the series in (4.83). In order to ensure convergence of the series by satisfying the conditions of (4.101), the series of terms in (4.96) and (4.99) has to be represented in two different ways. The first representation results in an expansion with all negative powers of a (subscript A) and the second representation has both positive and negative powers of a (subscript B). The convergence criteria which determines the usability of each representation is derived in Section 4.2.4.

The nth power of the load-end reflection coefficient in (4.96) can be written as

$$\left(\Gamma_L'\right)^n = \left(\frac{1 + \frac{1}{aB - a^2}}{1 - \frac{1}{aB + a^2}} \right)^n (-1)^n \left(\frac{1 - \frac{B}{a}}{1 + \frac{B}{a}} \right)^n. \tag{4.102}$$

The first term on the right side of (4.102) can be written using (4.101) as

$$\left(\frac{1 + \frac{1}{aB - a^2}}{1 - \frac{1}{aB + a^2}} \right)^n = \sum_{i=0}^{n} \sum_{j=0}^{\infty} \left[\frac{n\,(n-1+j)!}{i!j!\,(n-i)!} \left(\frac{1}{aB - a^2} \right)^i \left(\frac{1}{aB + a^2} \right)^j \right], \tag{4.103}$$

or

$$\begin{aligned} \left(\frac{1 + \frac{1}{aB - a^2}}{1 - \frac{1}{aB + a^2}} \right)^n &= \sum_{i=0}^{n} \sum_{j=0}^{\infty} \left[\frac{n\,(n-1+j)!}{i!j!\,(n-i)!} \left(\frac{1}{a^2 \left(\frac{B}{a} - 1 \right)} \right)^i \left(\frac{1}{a^2 \left(1 + \frac{B}{a} \right)} \right)^j \right] \\ &= \sum_{i=0}^{n} \sum_{j=0}^{\infty} \left[\frac{n\,(n-1+j)!}{i!j!\,(n-i)!} a^{-2i-2j} \frac{(-1)^i}{\left(1 - \frac{B}{a}\right)^i} \frac{1}{\left(1 + \frac{B}{a}\right)^j} \right]. \end{aligned} \tag{4.104}$$

Using the Binomial expansion in (4.77), one gets

$$\left(\frac{1+\frac{1}{aB-a^2}}{1-\frac{1}{aB+a^2}}\right)^n = \sum_{i=0}^{n}\sum_{j=0}^{\infty}\left[\begin{array}{l}\frac{n(n-1+j)!}{i!j!(n-i)!}a^{-2i-2j}\left(-1\right)^i\sum_{d=0}^{\infty}\left[\begin{pmatrix}-i\\ d\end{pmatrix}\left(-\frac{B}{a}\right)^d\right]\\ \times\sum_{e=0}^{\infty}\left[\begin{pmatrix}-j\\ e\end{pmatrix}\left(\frac{B}{a}\right)^e\right]\end{array}\right]$$

$$= \sum_{i=0}^{n}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\left[\begin{array}{l}\frac{n(n-1+j)!}{i!j!(n-i)!}\left(-1\right)^{i+d}\begin{pmatrix}-i\\ d\end{pmatrix}\begin{pmatrix}-j\\ e\end{pmatrix}\\ \times B^{d+e}a^{-2i-2j-d-e}\end{array}\right]. \quad (4.105)$$

The second term on the right side of (4.102) can be written using (4.101) as

$$\left(-1\right)^n\left(\frac{1-\frac{B}{a}}{1+\frac{B}{a}}\right)^n = \left(-1\right)^n\sum_{h=0}^{n}\sum_{p=0}^{\infty}\left[\frac{n\left(n-1+p\right)!}{h!p!\left(n-h\right)!}\left(-\frac{B}{a}\right)^h\left(-\frac{B}{a}\right)^p\right]$$

$$= \left(-1\right)^n\sum_{h=0}^{n}\sum_{p=0}^{\infty}\left[\frac{n\left(n-1+p\right)!}{h!p!\left(n-h\right)!}\left(-1\right)^{h+p}B^{h+p}a^{-h-p}\right]. \quad (4.106)$$

Combining (4.105) and (4.106), (4.102) can be written as

$$\left(\Gamma_L'\right)^n = \sum_{i=0}^{n}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\left[\frac{n\left(n-1+j\right)!}{i!j!\left(n-i\right)!}\left(-1\right)^{i+d}\begin{pmatrix}-i\\ d\end{pmatrix}\begin{pmatrix}-j\\ e\end{pmatrix}B^{d+e}a^{-2i-2j-d-e}\right]$$

$$\times\ \left(-1\right)^n\sum_{h=0}^{n}\sum_{p=0}^{\infty}\left[\frac{n\left(n-1+p\right)!}{h!p!\left(n-h\right)!}\left(-1\right)^{h+p}B^{h+p}a^{-h-p}\right]. \quad (4.107)$$

Finally,

$$\left(\Gamma_L'\right)^n = \sum_{i=0}^{n}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\sum_{h=0}^{n}\sum_{p=0}^{\infty}\left[\begin{array}{l}\frac{n(n-1+j)!}{i!j!(n-i)!}\frac{n(n-1+p)!}{h!p!(n-h)!}\left(-1\right)^{i+d+n+h+p}\\ \times\begin{pmatrix}-i\\ d\end{pmatrix}\begin{pmatrix}-j\\ e\end{pmatrix}B^{d+e+h+p}\\ \times a^{-2i-2j-d-e-h-p}\end{array}\right]. \quad (4.108)$$

The nth power of the source-end reflection coefficient (4.99) can be written as

$$\left(\Gamma_S'\right)^n = G_s^n\left(\frac{1-\frac{1}{aG_s}}{1-\frac{G_s}{a}}\right)^n, \quad (4.109)$$

which can be expanded using (4.101) as

$$\left(\Gamma_S^{'}\right)^n = G_s^n \sum_{q=0}^{n} \sum_{w=0}^{\infty} \left[\frac{n\,(n-1+w)!}{q!w!\,(n-q)!} (-1)^q\, G_s^{w-q} a^{-q-w} \right]. \tag{4.110}$$

Combining (4.108) and (4.110) gives

$$\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = \sum_{i=0}^{n}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\sum_{h=0}^{n}\sum_{p=0}^{\infty}\sum_{q=0}^{n}\sum_{w=0}^{\infty} \left[\begin{array}{l} \frac{n(n+j-1)!}{i!j!(n-i)!}\frac{n(n+p-1)!}{h!p!(n-h)!}\frac{n(n-1+w)!}{q!w!(n-q)!} \\ \times (-1)^{i+d+n+h+p+q} \\ \times \begin{pmatrix} -i \\ d \end{pmatrix} \begin{pmatrix} -j \\ e \end{pmatrix} B^{d+e+h+p} \\ \times G_s^{n+w-q} a^{-2i-2j-d-e-h-p-q-w} \end{array} \right]. \tag{4.111}$$

For the sake of compactness, (4.111) is written as

$$\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = S_C^{'} a^{-2i-2j-d-e-h-p-q-w}. \tag{4.112}$$

Similarly,

$$\left(\Gamma_L^{'}\right)^{n+1}\left(\Gamma_S^{'}\right)^n = \sum_{i=0}^{n+1}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\sum_{h=0}^{n}\sum_{p=0}^{\infty}\sum_{q=0}^{n}\sum_{w=0}^{\infty} \left[\begin{array}{l} \frac{(n+1)(n+j)!}{i!j!(n-i+1)!}\frac{(n+1)(n+p)!}{h!p!(n-h+1)!}\frac{n(n-1+w)!}{q!w!(n-q)!} \\ \times (-1)^{i+d+n+h+p+q+1} \\ \times \begin{pmatrix} -i \\ d \end{pmatrix} \begin{pmatrix} -j \\ e \end{pmatrix} B^{d+e+h+p} \\ \times G_s^{n+w-q} a^{-2i-2j-d-e-h-p-q-w} \end{array} \right], \tag{4.113}$$

which is shortened to

$$\left(\Gamma_L^{'}\right)^{n+1}\left(\Gamma_S^{'}\right)^n = S_C^{''} a^{-2i-2j-d-e-h-p-q-w}. \tag{4.114}$$

Substituting (4.79), (4.112) and (4.114) into (4.83), and using the inverse Laplace transform identity in (4.9), the time domain equivalent of $V_2^{'}(s)$ is

$$V_{2A}^{'}(t) \;=\; V_{dd}\frac{Z_o}{Z_o + R_{tr}}$$

$$\times \sum_{n=1}^{g} \left(\begin{array}{l} \left(S_C^{'} + S_C^{''}\right) \left(\frac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\frac{\theta}{2}} I_\theta \left(\sigma\sqrt{t^2 - (2n+1)^2 L^2 lc}\right) \\ + \frac{1}{1-G_s} \left(S_C^{'} + S_C^{''}\right) \sum_{u=1}^{\infty} \left[\begin{array}{l} D \left(\frac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\frac{\phi}{2}} \\ \times I_\phi \left(\sigma\sqrt{t^2 - (2n+1)^2 L^2 lc}\right) \end{array} \right] \end{array} \right)$$

$$\times \quad u_o \left(t - (2n+1)L\sqrt{lc}\right) \tag{4.115}$$

where $\theta = 2i + 2j + d + e + h + p + q + w$ and $\phi = 2i + 2j + d + e + h + p + q + w + u$, and D is defined in (4.92).

When (4.115) no longer converges, (4.75) and (4.99) can be combined to write (4.111) as

$$\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = \left(\frac{1 - \frac{\sigma}{2}\left(a - \frac{1}{a}\right) C_L Z_o}{1 + \frac{\sigma}{2}\left(a - \frac{1}{a}\right) C_L Z_o}\right)^n G_s^n \left(\frac{1 - \frac{1}{aG_s}}{1 - \frac{G_s}{a}}\right)^n, \tag{4.116}$$

which can be expanded using (4.101) as

$$\begin{aligned} \left(\Gamma_L^{'}\Gamma_S^{'}\right)^n &= \sum_{i=0}^{n}\sum_{j=0}^{\infty} \left[\frac{n\,(n-1+j)!}{i!j!\,(n-i)!} (-1)^{i+j} \left(\frac{\sigma}{2} C_L Z_o\right)^{i+j} \left(a - \frac{1}{a}\right)^{i+j}\right] \\ &\times \quad G_s^n \sum_{p=0}^{n}\sum_{q=0}^{\infty} \left[\frac{n\,(n-1+q)!}{p!q!\,(n-p)!} (-1)^p (G_s)^{q-p} \left(\frac{1}{a}\right)^{p+q}\right]. \end{aligned} \tag{4.117}$$

Using the Binomial expansion of (4.100), one gets

$$\begin{aligned} \left(\Gamma_L^{'}\Gamma_S^{'}\right)^n &= \sum_{i=0}^{n}\sum_{j=0}^{\infty} \left[\begin{array}{l} \frac{n(n-1+j)!}{i!j!(n-i)!} (-1)^{i+j} \left(\frac{\sigma}{2} C_L Z_o\right)^{i+j} \\ \times \sum_{h=0}^{i+j} \left[\begin{pmatrix} i+j \\ h \end{pmatrix} (-1)^h a^{i+j-2h} \right] \end{array} \right] \\ &\times \quad G_s^n \sum_{p=0}^{n}\sum_{q=0}^{\infty} \left[\frac{n\,(n-1+q)!}{p!q!\,(n-p)!} (-1)^p G_s^{q-p} a^{-p-q}\right], \end{aligned} \tag{4.118}$$

which can be rewritten as

$$\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = \sum_{i=0}^{n}\sum_{j=0}^{\infty}\sum_{h=0}^{i+j}\sum_{p=0}^{n}\sum_{q=0}^{\infty} \left[\begin{array}{l} G_s^{n+q-p} \frac{n(n-1+j)!}{i!j!(n-i)!} \frac{n(n-1+q)!}{p!q!(n-p)!} \\ \times (-1)^{i+j+p+h} B^{-i-j} \begin{pmatrix} i+j \\ h \end{pmatrix} a^{i+j-2h-p-q} \end{array} \right]. \tag{4.119}$$

For the sake of compactness (4.119) is written as

$$\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = S_D^{'} a^{i+j-2h-p-q}. \tag{4.120}$$

Similarly,

$$\Gamma_L^{'}\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = \sum_{i=0}^{n+1}\sum_{j=0}^{\infty}\sum_{h=0}^{i+j}\sum_{p=0}^{n}\sum_{q=0}^{\infty}\left[\begin{array}{l} G_s^{n+q-p}\frac{(n+1)(n+j)!}{i!j!(n+1-i)!}\frac{n(n-1+q)!}{p!q!(n-p)!} \\ \times(-1)^{i+j+p+h}B^{-i-j}\begin{pmatrix} i+j \\ h \end{pmatrix} a^{i+j-2h-p-q} \end{array}\right], \tag{4.121}$$

which is shortened to

$$\Gamma_L^{'}\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n = S_D^{''} a^{i+j-2h-p-q}. \tag{4.122}$$

Substituting (4.120) and (4.122) into (4.83) and using the inverse Laplace transform

identities (4.9) and (4.71) gives $V_{2B}^{'}(t)$, the inverse Laplace transform of (4.83) :

$$
\begin{aligned}
V_{2B}^{'}(t) = {} & V_{dd}\tfrac{Z_o}{Z_o+R_{tr}} \\
& \Bigg[\left(S_D^{'}+S_D^{''}\right) \\
& \times \Bigg(2\beta(\varsigma) \sum_{k=0}^{\langle \frac{\varsigma}{2} \rangle} \sum_{m=0}^{\langle \frac{\varsigma}{2} \rangle} \sum_{y=1}^{\langle \frac{\varsigma-2k-1}{2} \rangle+1} \\
& \left[\begin{pmatrix} \varsigma \\ 2m \end{pmatrix} \begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix} (-1)^k C_{\varsigma-2k-1,y}\sigma^{-y+1}(2n+1)L\sqrt{lc} \right. \\
& \left. \times \frac{t^{\varsigma-2k-2y+1}}{\left(\sqrt{t^2-(2n+1)^2L^2lc}\right)^{\varsigma-2k-y+1}} I_{\varsigma-2k-y+1}\left(\sigma\sqrt{t^2-(2n+1)^2L^2lc}\right) \right] \\
& + \left(\tfrac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\left|\frac{\varsigma}{2}\right|} I_{|\varsigma|}\left(\sigma\sqrt{t^2-(2n+1)^2L^2lc}\right) \Bigg) \\
& \times \sum_{n=1}^{g} \Bigg. + \tfrac{1}{1-G_s}\left(S_D^{'}+S_D^{''}\right) \\
& \times \sum_{u=1}^{\infty} \Bigg[2D\beta(\xi) \sum_{k=0}^{\langle \frac{\xi}{2} \rangle} \sum_{m=0}^{\langle \frac{\xi}{2} \rangle} \sum_{y=1}^{\langle \frac{\xi-2k-1}{2} \rangle+1} \\
& \left[\begin{pmatrix} \xi \\ 2m \end{pmatrix} \begin{pmatrix} \frac{2m-1}{2} \\ k \end{pmatrix} (-1)^k C_{\xi-2k-1,y}\sigma^{-y+1}(2n+1)L\sqrt{lc} \right. \\
& \left. \times \frac{t^{\xi-2k-2y+1}}{\left(\sqrt{t^2-(2n+1)^2L^2lc}\right)^{\xi-2k-y+1}} I_{\xi-2k-y+1}\left(\sigma\sqrt{t^2-(2n+1)^2L^2lc}\right) \right] \\
& + D\left(\tfrac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\left|\frac{\xi}{2}\right|} I_{|\xi|}\left(\sigma\sqrt{t^2-(2n+1)^2L^2lc}\right) \Bigg] \Bigg] \\
& \times u_o\left(t-(2n+1)L\sqrt{lc}\right)
\end{aligned}
\tag{4.123}
$$

where $\varsigma = i+j-2h-p-q$ and $\xi = i+j-2h-p-q-u$, and D is defined in (4.92).

Accounting for the transformation in (4.72), the time domain transient response for the higher order reflections is

$$
V_2(t) = e^{\frac{-rt}{2l}}\left(V_{2A}^{'}(t) \text{ or } V_{2B}^{'}(t)\right). \tag{4.124}
$$

4.2.4 Comprehensive model

The complete time domain transient response is the inverse Laplace transform of (4.5), which can be written as

$$V_{fin}(L,t) = V_1(t) \ + \ V_2(t). \tag{4.125}$$

From (4.94) and (4.124), this comprehensive model consists of three separate models which are as follows :

$$\begin{aligned}
&\text{Model 1}: \ V_{fin}(L,t) = e^{\frac{-rt}{2l}}\left(V_{1A}^{'}(t) + V_{2A}^{'}(t)\right) && (4.88) \text{ and } (4.115) \\
&\text{Model 2}: \ V_{fin}(L,t) = e^{\frac{-rt}{2l}}\left(V_{1B}^{'}(t) + V_{2A}^{'}(t)\right) && (4.91) \text{ and } (4.115) \\
&\text{Model 3}: \ V_{fin}(L,t) = e^{\frac{-rt}{2l}}\left(V_{1B}^{'}(t) + V_{2B}^{'}(t)\right) && (4.91) \text{ and } (4.123)
\end{aligned} \tag{4.126}$$

The choice of model depends on conditions which are developed based on simple convergence rules.

The Ratio Test for the convergence of an infinite series [58] states that a series converges if

$$\left|\frac{T_{n+1}}{T_n}\right| < 1, \ \forall \ n \geq N_0, \ N_0 \in R \tag{4.127}$$

where T_n is the nth term, and N_0 is some real number. For Model 1, the convergence problem is mainly due to the first term which is

$$P\sum_{i=0}^{\infty}\left[(-M)^i\left(\frac{t-t_f}{t+t_f}\right)^{\frac{i+1}{2}} I_{i+1}\left(\sigma\sqrt{t^2-t_f^2}\right)\right]. \tag{4.128}$$

The modified Bessel function can be approximated for small arguments $x \ll n$ as

$$I_n(x) \simeq \frac{1}{n!}\left(\frac{x}{2}\right)^n. \tag{4.129}$$

Assuming that for some $i = n$, $n \geq N_o$,

$$\sigma\sqrt{t^2-t_f^2} \ll n+1, \tag{4.130}$$

(4.128) can be approximated as

$$\approx P\left(-M\right)^{n}\sqrt{\frac{t-t_f}{t+t_f}}^{n+1}\left(\frac{\sigma\sqrt{t^2-t_f^2}}{2}\right)^{n+1}\frac{1}{(n+1)!}, \tag{4.131}$$

or

$$\approx P\left(-M\right)^{n}\left(t-t_f\right)^{n+1}\left(\frac{\sigma}{2}\right)^{n+1}\frac{1}{(n+1)!}. \tag{4.132}$$

Therefore,

$$\left|\frac{T_{n+1}}{T_n}\right| = \left|-\frac{1}{n+2}M\frac{\sigma}{2}\left(t-t_f\right)\right|. \tag{4.133}$$

Experimental observations indicate that the series in (4.128) converges if

$$\left|-\frac{1}{n+2}\frac{M\sigma}{2}\left(t-t_f\right)\right| < 1, \quad \forall \ n \geq N_o, \ N_o = 30. \tag{4.134}$$

Or,

$$t - t_f < \left|-\left(n+2\right)\frac{2}{M\sigma}\right| \quad \forall \ n \geq 30, \tag{4.135}$$

i.e.

$$t < \frac{64}{M\sigma} + t_f. \tag{4.136}$$

This first boundary point is denoted t_1. Therefore, Model 1 converges if

$$t \leq t_1, \text{ where } t_1 = \frac{64}{M\sigma} + t_f. \tag{4.137}$$

For Model 2, the convergence problem is caused by the summation of the index p in the second term,

$$\sum_{i=0}^{n+1}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\sum_{h=0}^{n}\sum_{p=0}^{\infty}\sum_{q=0}^{n}$$
$$\times\sum_{w=0}^{\infty}\left[\begin{array}{l}\frac{(n+1)(n+j)!}{i!j!(n-i+1)!}\frac{(n+1)(n+p)!}{h!p!(n-h+1)!}\frac{n(n-1+w)!}{q!w!(n-q)!}\\ \times\left(-1\right)^{i+d+n+h+p+q+1}\begin{pmatrix}-i\\ d\end{pmatrix}\begin{pmatrix}-j\\ e\end{pmatrix}B^{d+e+h+p}G_s^{n+w-q}\\ \times\left(\sqrt{\frac{t-(2n+1)t_f}{t+(2n+1)t_f}}\right)^{2i+2j+d+e+h+p+q+w}\\ \times I_{2i+2j+d+e+h+p+q+w}\left(t^2-\left(2n+1\right)^2t_f^2\right)\end{array}\right]. \tag{4.138}$$

Setting $n = 1$, $i = 0$, $j = 0$, $d = 0$, $e = 0$, $h = 0$, $q = 0$ and $w = 0$, gives

$$\sum_{p=0}^{\infty}\left[\frac{(1+p)!}{p!}(-1)^{p+2}B^{p}G_{s}\left(\sqrt{\frac{t-3t_{f}}{t+3t_{f}}}\right)^{p}I_{p}\left(\sigma\sqrt{t^{2}-(3)^{2}t_{f}^{2}}\right)\right], \tag{4.139}$$

which can be approximated using (4.129) as

$$\approx\sum_{p=0}^{\infty}\left[(1+p)(-1)^{p}B^{p}G_{s}(t-3t_{f})^{p}\frac{1}{p!}\left(\frac{\sigma}{2}\right)^{p}\right]. \tag{4.140}$$

For some $p = n$, the ratio of the $n+1^{th}$ term to the nth term is

$$\frac{T_{n+1}}{T_{n}}=\frac{n+2}{n+1}(-1)B(t-3t_{f})\left(\frac{\sigma}{2}\right)\frac{1}{n+1}. \tag{4.141}$$

Therefore, $\left|\frac{T_{n+1}}{T_{n}}\right| < 1$ implies

$$\left|\frac{n+2}{n+1}(-1)B(t-3t_{f})\left(\frac{\sigma}{2}\right)\frac{1}{n+1}\right| < 1, \tag{4.142}$$

or,

$$t < \frac{(n+1)^{2}}{n+2}\frac{2}{B\sigma}+3t_{f}. \tag{4.143}$$

Experimental observations indicate that the series in (4.139) converges if (4.143) is true for all $n \geq N_o$, $N_o = 9$. This second boundary point is denoted by t_2. Therefore Model 2 converges if

$$t \leq t_{2},\ \text{where } t_{2}=\frac{100}{11}\frac{2}{\mathrm{B}\sigma}+3t_{f}. \tag{4.144}$$

Based on (4.137) and (4.144), the comprehensive model can be tabulated as

If $t_1 \leq t_2$, use	**Model 1** for $t \leq t_1$	**Model 2** for $t_1 \leq t \leq t_2$	**Model 3** for $t_2 < t$
If $t_1 \geq t_2$, use	**Model 1** for $t \leq t_2$	–	**Model 3** for $t_2 < t$

The comprehensive time domain model can be written as

$$V_{fin}(L, t_{f} < t < 3t_{f}) = V_{1}(t), \tag{4.145}$$

and

$$V_{fin}(L, t > 3t_f) = V_1(t) + V_2(t). \tag{4.146}$$

The comprehensive model is compared to HSPICE simulations for two values of the load capacitance in Figure 4.4. The error between HSPICE simulations and the new model is negligible (less than 1%) for a wide range of line parameters for $t \leq 5t_f$.

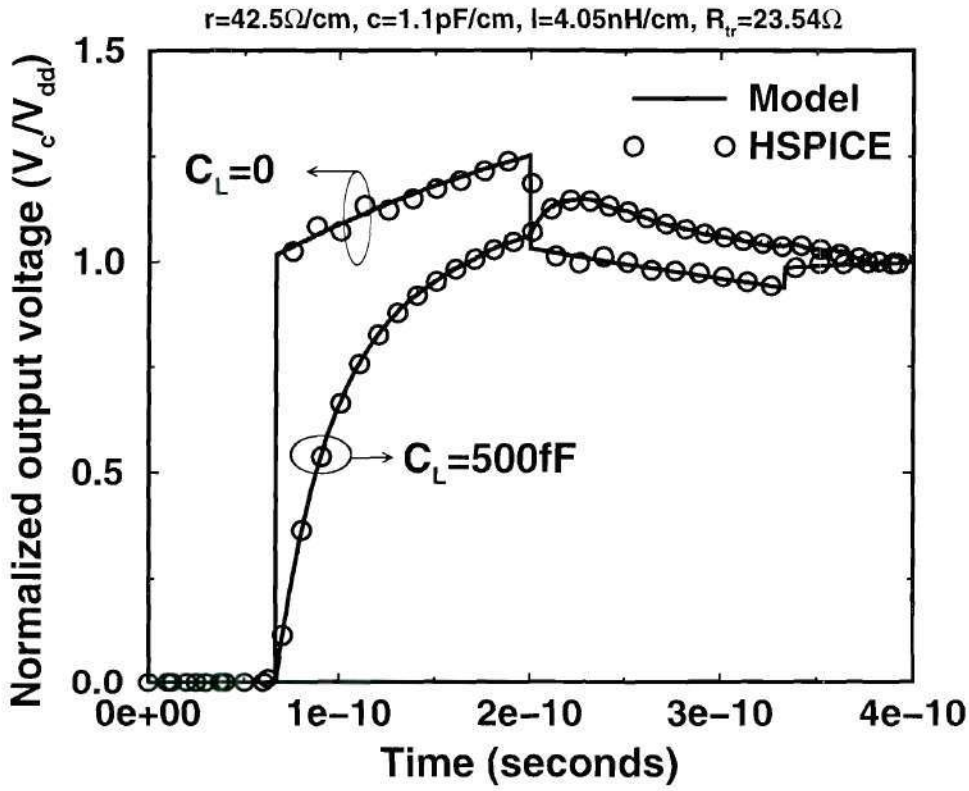


Figure 4.4: Output transient response of a 1cm long distributed *rlc* interconnect to a step input waveform- complete model vs. HSPICE simulations for $C_L = 0$ and $C_L = 500fF$.

4.3 Single line transient model - finite rise time input

The transient response of an open-circuit single line to a ramp input with a finite rise time T_{rise} is derived in [10]. The complete transient response of a distributed *rlc* interconnect with a capacitive load termination to a ramp input is derived in this section.

The transfer function of a finite distributed *rlc* interconnect is given by

$$V_{fin}\left(L,s\right)=\left(1+\Gamma_L\right)\sum_{n=0}^{g}\left[\Gamma_S^n\Gamma_L^n V_{inf}\left(\left(2n+1\right)L,s\right)\right], \tag{4.147}$$

where $V_{inf}(x,s)$ is the transient response of an infinitely long line, given by

$$V_{inf}\left(x,s\right)=V_{in}\left(s\right)\frac{Z_o\sqrt{\frac{s+r/l}{s}}}{R_{tr}+Z_o\sqrt{\frac{s+r/l}{s}}}e^{-x\sqrt{lc}\sqrt{s(s+r/l)}}. \tag{4.148}$$

where $V_{in}(s)$ is the Laplace transform of the input signal $V_{int}(t)$. A ramp input with a finite rise time T_{rise} can be described as

$$V_{in}\left(t\right)=\frac{V_{dd}}{T_{rise}}\left(tu_o\left(t\right)-\left(t-T_{rise}\right)u_o\left(t-T_{rise}\right)\right). \tag{4.149}$$

The Laplace transform of the ramp input is given by

$$V_{in}\left(s\right)=\frac{V_{dd}}{s^2T_{rise}}\left(1-e^{-sT_{rise}}\right). \tag{4.150}$$

In the following sections, the time domain expression for the first reflection will be derived in Section 4.3.1. The time domain expression for the higher order reflection terms is derived in Section 4.3.2. Finally, the comprehensive time domain model is derived in Section 4.3.3.

4.3.1 First reflection

Consider the term corresponding to the first reflection ($n = 0$) in (4.147)

$$V_1(L,s) = (1+\Gamma_L)\, V_{inf}(L,s)\,. \tag{4.151}$$

Using (4.150), the first reflection term can be expressed in the Laplace domain as

$$V_1(L,s) = \frac{V_{dd}}{s^2 T_{rise}}\left(1-e^{-sT_{rise}}\right)(1+\Gamma_L)\frac{Z_o\sqrt{\frac{s+r/l}{s}}}{R_{tr}+Z_o\sqrt{\frac{s+r/l}{s}}}e^{-L\sqrt{lc}\sqrt{s(s+r/l)}} \tag{4.152}$$

Using the following transformation,

$$s \to s-\sigma \tag{4.153}$$

(4.152) can be written as

$$V_1'(L,s) = \frac{V_{dd}}{T_{rise}}e^{-L\sqrt{lc}\sqrt{(s-\sigma)(s+\sigma)}}\left(1-e^{-(s-\sigma)T_{rise}}\right)\frac{1+\Gamma_L'}{(s-\sigma)^2}\frac{Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}{R_{tr}+Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}, \tag{4.154}$$

or

$$\begin{aligned} V_1'(L,s) &= \frac{V_{dd}}{T_{rise}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\left(1-e^{-(s-\sigma)T_{rise}}\right)\left(1+\Gamma_L'\right) \\ &\times \left[\frac{1}{s-\sigma}\sqrt{\frac{s+\sigma}{s-\sigma}}\frac{Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}{R_{tr}+Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}\right] \end{aligned} \tag{4.155}$$

First, a simpler problem shall be solved. Let

$$P_1'(L,s) = \frac{V_{dd}}{T_{rise}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\left(1+\Gamma_L'\right)\left[\frac{1}{s-\sigma}\sqrt{\frac{s+\sigma}{s-\sigma}}\frac{Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}{R_{tr}+Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}\right]. \tag{4.156}$$

Using the substitution

$$s = \frac{\sigma}{2}\left(a+\frac{1}{a}\right), \tag{4.157}$$

the term in parenthesis in (4.156) can be written as

$$\frac{1}{s-\sigma}\sqrt{\frac{s+\sigma}{s-\sigma}}\frac{Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}}{R_{tr}+Z_o\sqrt{\frac{s+\sigma}{s-\sigma}}} = \frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}a\frac{(a+1)^2}{(a-1)^3}\frac{1}{a+G_s}. \tag{4.158}$$

Using partial fractions, (4.158) can be rewritten as

$$= \frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\left(\begin{array}{l}\frac{4}{(1-G_s)(a-1)^3}+4\frac{1-2G_s}{(1-G_s)^2(a-1)^2}\\ +\frac{5G_s^2-2G_s+1}{(1-G_s)^3(a-1)}-G_s\frac{(1+G_s)^2}{(1-G_s)^3(a-G_s)}\end{array}\right). \tag{4.159}$$

Using series summation identities, (4.159) can be expressed as

$$= \frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\sum_{u=1}^{\infty}\left[a^{-u}\left(\begin{array}{l}2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}\\ +\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\end{array}\right)\right]. \tag{4.160}$$

As in (4.86), partial fractions can be used to write

$$1+\Gamma_L^{'} = P\sum_{i=0}^{\infty}\left[(-M)^i a^{-i-1}\right]+Q\sum_{i=0}^{\infty}\left[(-N)^i a^{-i-1}\right], \tag{4.161}$$

where P, Q, M and N are defined in (4.85). The subscript A is used in the following equation to differentiate it from a later equation, where a slightly different series expansion is used to represent the partial fractions, in order to obey convergence criteria. Therefore, (4.156) can be written as

$$\begin{aligned} P_{1A}^{'}(L,s) &= \frac{V_{dd}}{T_{rise}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\left(P\sum_{i=0}^{\infty}\left[(-M)^i a^{-i-1}\right]+Q\sum_{i=0}^{\infty}\left[(-N)^i a^{-i-1}\right]\right)\\ &\times \frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\sum_{u=1}^{\infty}\left[a^{-u}\left(\begin{array}{l}2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}\\ +\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\end{array}\right)\right]. \end{aligned} \tag{4.162}$$

Substituting for a from (4.157), (4.162) can be rewritten as

$$\begin{aligned} P_{1A}^{'}(L,s) &= \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\\ &\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(P\left(-M\right)^i+Q\left(-N\right)^i\right)\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-i-u-1}\end{array}\right]. \end{aligned} \tag{4.163}$$

Using the inverse Laplace transform identity of (4.9), the time domain equivalent of (4.163) can be written as

$$P_{1A}^{'}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$

$$\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(P\left(-M\right)^i+Q\left(-N\right)^i\right)\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}\\ \times I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right]$$

$$\times \quad u_o\left(t-L\sqrt{lc}\right) \tag{4.164}$$

Now, Laplace transforms have the following property :

$$L^{-1}\left[F\left(s\right)e^{-sT_{rise}}\right]=f\left(t-T_{rise}\right). \tag{4.165}$$

From (4.164) and (4.165), the time domain equivalent of (4.155) can be written as

$$\begin{aligned}V_{1A}^{'}\left(L,t\right) &= \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\\ &\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(P\left(-M\right)^i+Q\left(-N\right)^i\right)\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right]\\ &\times \quad u_o\left(t-L\sqrt{lc}\right)\\ &- \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{\sigma T_{rise}}\\ &\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(P\left(-M\right)^i+Q\left(-N\right)^i\right)\left(\frac{t-T_{rise}-L\sqrt{lc}}{t-T_{rise}+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}\\ \times I_{i+u+1}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-L^2lc}\right)\end{array}\right]\\ &\times \quad u_o\left(t-T_{rise}-L\sqrt{lc}\right).\end{aligned} \tag{4.166}$$

Finally, accounting for the transformation in (4.153), the time domain expression for the first reflection can be written as

$$V_{1A}\left(L,t\right)=e^{-\sigma t}V_{1A}^{'}\left(L,t\right) \tag{4.167}$$

When (4.167) does not converge, then a different series expansion can be used to write

$$1+\Gamma_L^{'}=P\sum_{i=0}^{\infty}\left[M^{-i-1}\left(-a\right)^i\right]+Q\sum_{i=0}^{\infty}\left[\left(-N\right)^i a^{-i-1}\right] \tag{4.168}$$

Therefore, (4.156) can be written as

$$P'_{1B}(L,s) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\frac{e^{-L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}$$
$$\times\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2(u-1)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(PM^{-i-1}(-1)^i\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{i-u}+Q(-N)^i\left(\frac{s+\sqrt{s^2-\sigma^2}}{\sigma}\right)^{-i-u-1}\right)\end{array}\right]. \tag{4.169}$$

Using the inverse Laplace transforms of (4.9) and (4.71), the time domain equivalent of $P'_{1B}(L,s)$ can be written as

$$P'_{1B}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$
$$\times\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2(u-1)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(\begin{array}{l}PM^{-i-1}(-1)^i\\ \left(\begin{array}{l}2\beta(i-u)\sum_{k=0}^{\langle\frac{i-u}{2}\rangle}\sum_{m=0}^{\langle\frac{i-u}{2}\rangle}\sum_{y=1}^{\langle\frac{i-u-2k-1}{2}\rangle+1}\\ \left[\begin{array}{l}\begin{pmatrix} i-u \\ 2m\end{pmatrix}\begin{pmatrix}\frac{2m-1}{2}\\ k\end{pmatrix}(-1)^k C_{i-u-2k-1,y}\\ \times\frac{\sigma^{-y+1}L\sqrt{lc}t^{i-u-2k-2y+1}}{\left(\sqrt{t^2-L^2lc}\right)^{i-u-2k-y+1}}I_{i-u-2k-y+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right]\\ +\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{|i-u|}{2}}I_{|i-u|}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right)\\ +Q(-N)^i\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right)\end{array}\right]$$
$$\times u_o\left(t-L\sqrt{lc}\right). \tag{4.170}$$

Using the Laplace transform property of (4.165), $V'_{1B}(L,t)$ can be written as

$$V'_{1B}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$

$$
\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}
\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right) \\
\times\left(\begin{array}{l}
PM^{-i-1}\left(-1\right)^i \\
\quad\times\left(\begin{array}{l}
2\beta\left(i-u\right)\sum\limits_{k=0}^{\left\langle\frac{i-u}{2}\right\rangle}\sum\limits_{m=0}^{\left\langle\frac{i-u}{2}\right\rangle}\sum\limits_{y=1}^{\left\langle\frac{i-u-2k-1}{2}\right\rangle+1} \\
\left[\begin{array}{l}
\left(\begin{array}{c} i-u \\ 2m \end{array}\right)\left(\begin{array}{c} \frac{2m-1}{2} \\ k \end{array}\right)\left(-1\right)^k C_{i-u-2k-1,y} \\
\times\frac{\sigma^{-y+1}L\sqrt{lc}t^{i-u-2k-2y+1}}{\left(\sqrt{t^2-L^2lc}\right)^{i-u-2k-y+1}}I_{i-u-2k-y+1}\left(\sigma\sqrt{t^2-L^2lc}\right)
\end{array}\right] \\
+\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{|i-u|}{2}}I_{|i-u|}\left(\sigma\sqrt{t^2-L^2lc}\right)
\end{array}\right) \\
+Q\left(-N\right)^i\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)
\end{array}\right)
\end{array}\right]
$$

$$
\times u_o\left(t-L\sqrt{lc}\right)
$$

$$
-\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{\sigma T_{rise}}
$$

$$
\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}
\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right) \\
\times\left(\begin{array}{l}
PM^{-i-1}\left(-1\right)^i \\
\quad\times\left(\begin{array}{l}
2\beta\left(i-u\right)\sum\limits_{k=0}^{\left\langle\frac{i-u}{2}\right\rangle}\sum\limits_{m=0}^{\left\langle\frac{i-u}{2}\right\rangle}\sum\limits_{y=1}^{\left\langle\frac{i-u-2k-1}{2}\right\rangle+1} \\
\left[\begin{array}{l}
\left(\begin{array}{c} i-u \\ 2m \end{array}\right)\left(\begin{array}{c} \frac{2m-1}{2} \\ k \end{array}\right)\left(-1\right)^k C_{i-u-2k-1,y} \\
\times\frac{\sigma^{-y+1}L\sqrt{lc}(t-T_{rise})^{i-u-2k-2y+1}}{\left(\sqrt{(t-T_{rise})^2-L^2lc}\right)^{i-u-2k-y+1}} \\
\times I_{i-u-2k-y+1}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-L^2lc}\right)
\end{array}\right] \\
+\left(\frac{t-T_{rise}-L\sqrt{lc}}{t-T_{rise}+L\sqrt{lc}}\right)^{\frac{|i-u|}{2}}I_{|i-u|}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-L^2lc}\right)
\end{array}\right) \\
+Q\left(-N\right)^i\left(\frac{t-T_{rise}-L\sqrt{lc}}{t-T_{rise}+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}I_{i+u+1}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-L^2lc}\right)
\end{array}\right)
\end{array}\right]
$$

$$
\times u_o\left(t-T_{rise}-L\sqrt{lc}\right). \tag{4.171}
$$

Finally, the time domain expression for the first reflection is given by

$$
V_{1B}\left(L,t\right)=e^{-\sigma t}V_{1B}^{'}\left(L,t\right) \tag{4.172}
$$

4.3.2 Higher order reflections

The higher order reflection terms ($n \geq 0$) in (4.147) is given by :

$$V_2(L,s) = \sum_{n=1}^{g} \left[\left(\Gamma_L^n \Gamma_S^n + \Gamma_L^{n+1} \Gamma_S^n \right) V_{inf}\left((2n+1) L, s \right) \right]. \tag{4.173}$$

Using the Laplace domain expression for a ramp input with a finite rise time T_{rise}, the higher order reflection terms can be described as

$$\begin{aligned} V_2(L,s) &= \frac{V_{dd}}{s^2 T_{rise}} \left(1 - e^{-sT_{rise}}\right) \frac{Z_o \sqrt{\frac{s+r/l}{s}}}{R_{tr} + Z_o \sqrt{\frac{s+r/l}{s}}} \\ &\times \sum_{n=1}^{g} \left[\left(\Gamma_L^n \Gamma_S^n + \Gamma_L^{n+1} \Gamma_S^n \right) e^{-(2n+1)L\sqrt{lc}\sqrt{s(s+r/l)}} \right]. \end{aligned} \tag{4.174}$$

Using the transformation in 4.153) and the substitution in (4.157), we get

$$\begin{aligned} V_2'(L,s) &= \frac{V_{dd}}{T_{rise}} \left(1 - e^{-(s-\sigma)T_{rise}}\right) \frac{Z_o \sqrt{\frac{s+\sigma}{s-\sigma}}}{R_{tr} + Z_o \sqrt{\frac{s+\sigma}{s-\sigma}}} \\ &\times \sum_{n=1}^{g} \left[\frac{\left(\Gamma_L'^n \Gamma_S'^n + \Gamma_L'^{n+1} \Gamma_S'^n \right)}{(s-\sigma)^2} e^{-(2n+1)L\sqrt{lc}\sqrt{(s-\sigma)(s+\sigma)}} \right]. \end{aligned} \tag{4.175}$$

$$P_2'(L,s) = \frac{V_{dd}}{T_{rise}} \left[\frac{1}{s-\sigma} \sqrt{\frac{s+\sigma}{s-\sigma}} \frac{Z_o \sqrt{\frac{s+\sigma}{s-\sigma}}}{R_{tr} + Z_o \sqrt{\frac{s+\sigma}{s-\sigma}}} \right] \tag{4.176}$$

$$\times \sum_{n=1}^{g} \left[\left(\Gamma_L'^n \Gamma_S'^n + \Gamma_L'^{n+1} \Gamma_S'^n \right) \frac{e^{-(2n+1)L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}} \right]. \tag{4.177}$$

As proved earlier in (4.112), the nth powers of the reflection coefficients can be written as

$$\left(\Gamma_L' \Gamma_S' \right)^n = S_C' a^{-2i-2j-d-e-h-p-q-w} \tag{4.178}$$

Similarly, as proved earlier in (4.114),

$$\left(\Gamma_L' \right)^{n+1} \left(\Gamma_S' \right)^n = S_C'' a^{-2i-2j-d-e-h-p-q-w}. \tag{4.179}$$

Using these series summations, we can write

$$P'_{2A}(L,s) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}\sum_{u=1}^{\infty}\left[a^{-u}\left(\begin{array}{l}2(u-1)\frac{-G_s(u+2)+u}{(1-G_s)^2}\\ +\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\end{array}\right)\right]$$
$$\times\sum_{n=1}^{g}\left[\begin{array}{l}\left(S'_C a^{-2i-2j-d-e-h-p-q-w}+S''_C a^{-2i-2j-d-e-h-p-q-w}\right)\\ \times\frac{e^{-(2n+1)L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\end{array}\right], \quad (4.180)$$

which can be simplified to

$$P'_{2A}(L,s) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$
$$\times\sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S'_C+S''_C\right)\left(2(u-1)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times a^{-2i-2j-d-e-h-p-q-w-u}\frac{e^{-(2n+1)L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\end{array}\right]. \quad (4.181)$$

Using the inverse Laplace transform identity of (4.9), the time domain equiavalent of $P'_{2A}(L,s)$ can be written as

$$P'_{2A}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$
$$\times\sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S'_C+S''_C\right)\left(\begin{array}{l}2(u-1)\frac{-G_s(u+2)+u}{(1-G_s)^2}\\ +\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\end{array}\right)\\ \times\left(\frac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\frac{\gamma}{2}} I_\gamma\left(\sigma\sqrt{t^2-(2n+1)^2L^2lc}\right)\end{array}\right] \quad (4.182)$$
$$\times u_o\left(t-(2n+1)L\sqrt{lc}\right)$$

where

$$\gamma = -2i-2j-d-e-h-p-q-w-u. \quad (4.183)$$

Using the Laplace transform property of (4.165), the time domain equivalent of (4.175) can be written as

$$V'_{2A}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$
$$\times\sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S'_C+S''_C\right)\left(\begin{array}{l}2(u-1)\frac{-G_s(u+2)+u}{(1-G_s)^2}\\ +\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\end{array}\right)\\ \times\left(\frac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\frac{\gamma}{2}} I_\gamma\left(\sigma\sqrt{t^2-(2n+1)^2L^2lc}\right)\end{array}\right]$$

$$\times u_o\left(t-(2n+1)L\sqrt{lc}\right)$$
$$-\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{\sigma T_{rise}}$$
$$\times\sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S_C^{'}+S_C^{''}\right)\left(\begin{array}{l}2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}\\+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\end{array}\right)\\\times\left(\frac{t-T_{rise}-(2n+1)L\sqrt{lc}}{t-T_{rise}+(2n+1)L\sqrt{lc}}\right)^{\frac{\gamma}{2}}I_\gamma\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]$$
$$\times u_o\left(t-T_{rise}-(2n+1)L\sqrt{lc}\right). \tag{4.184}$$

Finally, the time domain expression for the higher order reflection terms is

$$V_{1B}\left(L,t\right)=e^{-\sigma t}V_{1B}^{'}\left(L,t\right). \tag{4.185}$$

When (4.185) no longer converges, then a different series expansion can be used to write

$$\left(\Gamma_L^{'}\Gamma_S^{'}\right)^n=S_D^{'}a^{i+j-2h-p-q} \tag{4.186}$$

and

$$\left(\Gamma_L^{'}\right)^{n+1}\left(\Gamma_S^{'}\right)^n=S_D^{''}a^{i+j-2h-p-q}, \tag{4.187}$$

as proved earlier in (4.120) and (4.121). Therefore (4.177) can be written as

$$P_{2B}^{'}\left(L,s\right)=\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$
$$\times\sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S_D^{'}+S_D^{''}\right)\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\\times a^{i+j-2h-p-q-u}\frac{e^{-(2n+1)L\sqrt{lc}\sqrt{s^2-\sigma^2}}}{\sqrt{s^2-\sigma^2}}\end{array}\right]. \tag{4.188}$$

Using the inverse Laplace transform identities of (4.9) and (4.71), we get

$$P_{2B}^{'}\left(L,t\right)=\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$

$$\times \sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S_D^{'}+S_D^{''}\right)\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times 2\beta\left(\lambda\right)\sum\limits_{k=0}^{\langle\frac{\lambda}{2}\rangle}\sum\limits_{m=0}^{\langle\frac{\lambda}{2}\rangle}\sum\limits_{y=1}^{\langle\frac{\lambda-2k-1}{2}\rangle+1}\left[\begin{array}{l}\begin{pmatrix}\lambda\\ 2m\end{pmatrix}\begin{pmatrix}\frac{2m-1}{2}\\ k\end{pmatrix}\left(-1\right)^k C_{\lambda-2k-1,y}\\ \times\frac{\sigma^{-y+1}(2n+1)L\sqrt{lc}t^{\lambda-2k-2y+1}}{\left(\sqrt{t^2-(2n+1)^2L^2lc}\right)^{\lambda-2k-y+1}}\\ \times I_{\lambda-2k-y+1}\left(\sigma\sqrt{t^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]\\ +\left(\frac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\frac{|\lambda|}{2}}I_{|\lambda|}\left(\sigma\sqrt{t^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]$$

$$\times u_o\left(t-\left(2n+1\right)L\sqrt{lc}\right). \tag{4.189}$$

where

$$\lambda = i+j-2h-p-q-u. \tag{4.190}$$

Using the Laplace transform property of (4.165), we get

$$V_{2B}^{'}\left(L,t\right)=\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}$$

$$\times \sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S_D^{'}+S_D^{''}\right)\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times 2\beta\left(\lambda\right)\sum\limits_{k=0}^{\langle\frac{\lambda}{2}\rangle}\sum\limits_{m=0}^{\langle\frac{\lambda}{2}\rangle}\sum\limits_{y=1}^{\langle\frac{\lambda-2k-1}{2}\rangle+1}\left[\begin{array}{l}\begin{pmatrix}\lambda\\ 2m\end{pmatrix}\begin{pmatrix}\frac{2m-1}{2}\\ k\end{pmatrix}\left(-1\right)^k C_{\lambda-2k-1,y}\\ \times\frac{\sigma^{-y+1}(2n+1)L\sqrt{lc}t^{\lambda-2k-2y+1}}{\left(\sqrt{t^2-(2n+1)^2L^2lc}\right)^{\lambda-2k-y+1}}\\ \times I_{\lambda-2k-y+1}\left(\sigma\sqrt{t^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]\\ +\left(\frac{t-(2n+1)L\sqrt{lc}}{t+(2n+1)L\sqrt{lc}}\right)^{\frac{|\lambda|}{2}}I_{|\lambda|}\left(\sigma\sqrt{t^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]$$

$$\times u_o\left(t-\left(2n+1\right)L\sqrt{lc}\right)$$

$$-\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{\sigma T_{rise}}$$

$$\times \sum_{n=1}^{g}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(S_D^{'}+S_D^{''}\right)\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{\left(1-G_s\right)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{\left(1-G_s\right)^3}\right)\\ \times 2\beta\left(\lambda\right)\sum\limits_{k=0}^{\langle\frac{\lambda}{2}\rangle}\sum\limits_{m=0}^{\langle\frac{\lambda}{2}\rangle}\sum\limits_{y=1}^{\langle\frac{\lambda-2k-1}{2}\rangle+1}\\ \left[\begin{array}{l}\begin{pmatrix}\lambda\\ 2m\end{pmatrix}\begin{pmatrix}\frac{2m-1}{2}\\ k\end{pmatrix}\left(-1\right)^k C_{\lambda-2k-1,y}\\ \times\frac{\sigma^{-y+1}(2n+1)L\sqrt{lc}(t-T_{rise})^{\lambda-2k-2y+1}}{\left(\sqrt{(t-T_{rise})^2-(2n+1)^2L^2lc}\right)^{\lambda-2k-y+1}}\\ \times I_{\lambda-2k-y+1}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]\\ +\left(\frac{t-T_{rise}-(2n+1)L\sqrt{lc}}{t-T_{rise}+(2n+1)L\sqrt{lc}}\right)^{\frac{|\lambda|}{2}}I_{|\lambda|}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-\left(2n+1\right)^2L^2lc}\right)\end{array}\right]$$

$$\times u_o\left(t-T_{rise}-\left(2n+1\right)L\sqrt{lc}\right). \tag{4.191}$$

Finally, we get

$$V_{2B}\left(L,t\right)=e^{-\sigma t}V_{2B}^{'}\left(L,t\right). \tag{4.192}$$

4.3.3 Comprehensive model

The convergence criteria which determine the ranges in which the different models are valid, and therefore define the comprehensive model, is developed in this section.

For Model 1, the convergence problem occurs in the first term, for the series summation of the index i :

$$\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{\left(1-G_s\right)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{\left(1-G_s\right)^3}\right)\\ \times P\left(-M\right)^i\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right]. \tag{4.193}$$

Let $u=1$. Therefore,

$$2\left(u-1\right)\frac{-G_s\left(u+2\right)+u}{\left(1-G_s\right)^2}+\frac{5G_s^2-2G_s+1-G_s^u\left(1+G_s\right)^2}{\left(1-G_s\right)^3}=1. \tag{4.194}$$

Hence, (4.193) can be written as

$$\sum_{i=0}^{\infty}\left[P\left(-M\right)^i\left(\frac{t-t_f}{t+t_f}\right)^{\frac{i+2}{2}}I_{i+2}\left(\sigma\sqrt{t^2-t_f^2}\right)\right]. \tag{4.195}$$

Using the asymptotic approximation of a modified Bessel function in (4.129), we get

$$\approx P\left(-M\right)^{n}\sqrt{\frac{t-t_f}{t+t_f}}^{n+2}\left(\frac{\sigma\sqrt{t^2-t_f^2}}{2}\right)^{n+2}\frac{1}{(n+2)!}, \tag{4.196}$$

or,

$$\approx P\left(-M\right)^{n}\left(t-t_f\right)^{n+2}\left(\frac{\sigma}{2}\right)^{n+2}\frac{1}{(n+2)!}. \tag{4.197}$$

For some $i=n$, the ratio of the $n+1^{th}$ term to the nth term can therefore be written as

$$\left|\frac{T_{n+1}}{T_n}\right| = \left|-\frac{1}{n+3}M\frac{\sigma}{2}\left(t-t_f\right)\right|. \tag{4.198}$$

Experimental observations indicate that the series converges if

$$t-t_f < \left|-\left(n+3\right)\frac{2}{M\sigma}\right| \quad \forall \;\; n\geq 1. \tag{4.199}$$

This first boundary point is denoted by t_1. Therefore the series converges for

$$t\leq t_1, \quad \text{where} \;\; t_1 = \frac{8}{M\sigma}+t_f. \tag{4.200}$$

For Model 3, the convergence problem arises due to the first term which is

$$\sum_{i=0}^{n+1}\sum_{j=0}^{\infty}\sum_{d=0}^{\infty}\sum_{e=0}^{\infty}\sum_{h=0}^{n}\sum_{p=0}^{\infty}\sum_{q=0}^{n}\sum_{w=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\frac{(n+1)(n+j)!}{i!j!(n-i+1)!}\frac{(n+1)(n+p)!}{h!p!(n-h+1)!}\frac{n(n-1+w)!}{q!w!(n-q)!}\\ \times\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{\left(1-G_s\right)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{\left(1-G_s\right)^3}\right)\\ \times\left(-1\right)^{i+d+n+h+p+q+1}\begin{pmatrix}-i\\ d\end{pmatrix}\begin{pmatrix}-j\\ e\end{pmatrix}B^{d+e+h+p}\\ \times G_s^{n+w-q}\left(\sqrt{\frac{t-(2n+1)t_f}{t+(2n+1)t_f}}\right)^{2i+2j+d+e+h+p+q+w+u}\\ \times I_{2i+2j+d+e+h+p+q+w+u}\left(t^2-\left(2n+1\right)^2t_f^2\right)\end{array}\right] \tag{4.201}$$

Setting $n=1$, $i=0$, $j=0$, $d=0$, $e=0$, $h=0$, $q=0$, $w=0$ and $u=1$, this reduces to

$$\sum_{p=0}^{\infty}\left[\frac{(1+p)!}{p!}\left(-1\right)^{p+2}B^pG_s\left(\sqrt{\frac{t-3t_f}{t+3t_f}}\right)^{p+1}I_{p+1}\left(\sigma\sqrt{t^2-\left(3\right)^2t_f^2}\right)\right]. \tag{4.202}$$

Using the asymptotic approximation of (4.129), we get

$$\approx \sum_{p=0}^{\infty} \left[(1+p)\,(-1)^p\, B^p G_s\,(t-3t_f)^{p+1}\, \frac{1}{(p+1)!} \left(\frac{\sigma}{2}\right)^{p+1} \right]. \tag{4.203}$$

For some $p = n$, the ratio of the $n+1^{th}$ term to the nth term can be written as

$$\frac{T_{n+1}}{T_n} = \frac{n+2}{n+1}\,(-1)\,B\,(t-3t_f)\left(\frac{\sigma}{2}\right)\frac{1}{n+2}. \tag{4.204}$$

For convergence, the following condition should be satisfied for all $n \geq N_o$, for some N_o :

$$\left| \frac{1}{n+1}\,(-1)\,B\,(t-3t_f)\left(\frac{\sigma}{2}\right) \right| < 1. \tag{4.205}$$

Experimental observations indicate that$N_o = 1$. This second boundary point is denoted as t_2. Therefore, the series converges if

$$t < t_2, \quad \text{where} \quad t_2 = \frac{4}{B\sigma} + 3t_f. \tag{4.206}$$

The comprehensive model for the transient response of a distributed *rlc* interconnect with a ramp input can be tabulated as

If $t_1 \leq t_2$, use	**Model 1** for $t \leq t_1$	**Model 2** for $t_1 \leq t \leq t_2$	**Model 3** for $t_2 < t$
If $t_1 \geq t_2$, use	**Model 1** for $t \leq t_2$	–	**Model 3** for $t_2 < t$

These models for the ramp input transient response of capacitively-loaded interconnects are now verified against HSPICE simulations. Figures 4.5 and 4.6 plot the transient response for a ramp input with rise time T_{rise} equal to $20ps$ and $100ps$ for $C_L = 0$ and $C_L = 500fF$, respectively. The error between HSPICE simulations and the new model is negligible (less than 1%) for $t \leq 5t_f$. It is clear from Figures 4.5 and 4.6 that a finite rise time input increases the 50% time delay and decreases the peak overshoot of the output voltage. These models are necessary to determine the transient response of high speed *rlc* interconnects with non-trivial driver signal rise time.

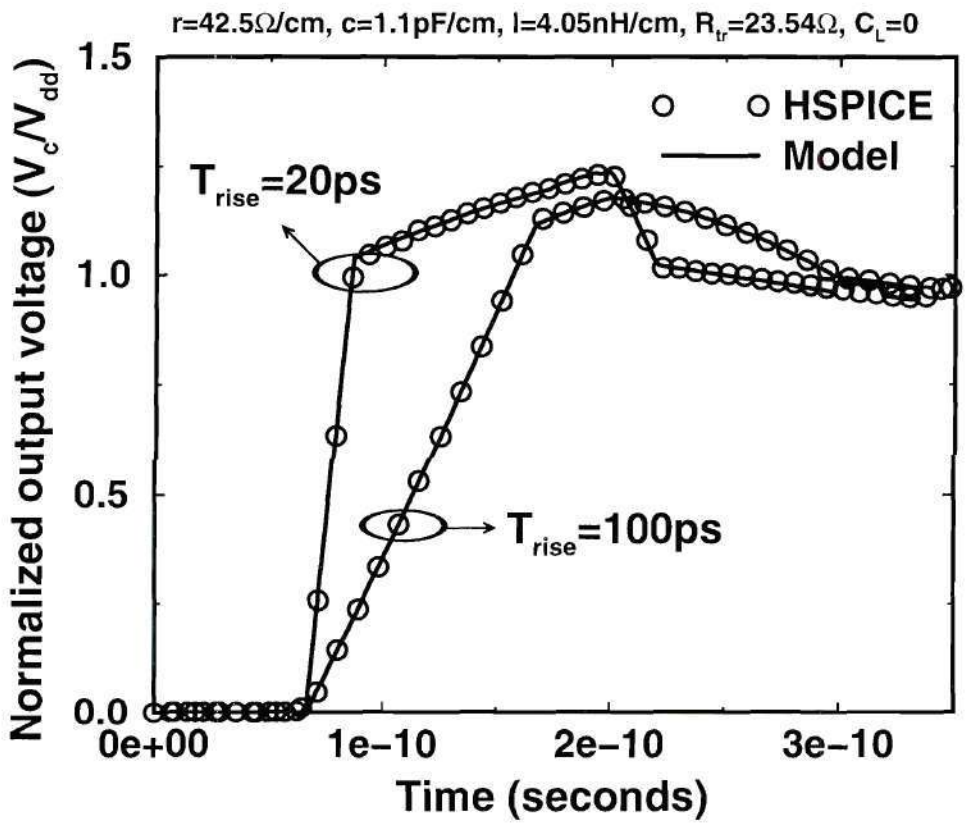


Figure 4.5: Output transient response of a 1cm long distributed *rlc* interconnect to a ramp input waveform - complete model vs. HSPICE simulations for $C_L = 0$, and input rise times, T_{rise}, of $20ps$ and $100ps$.

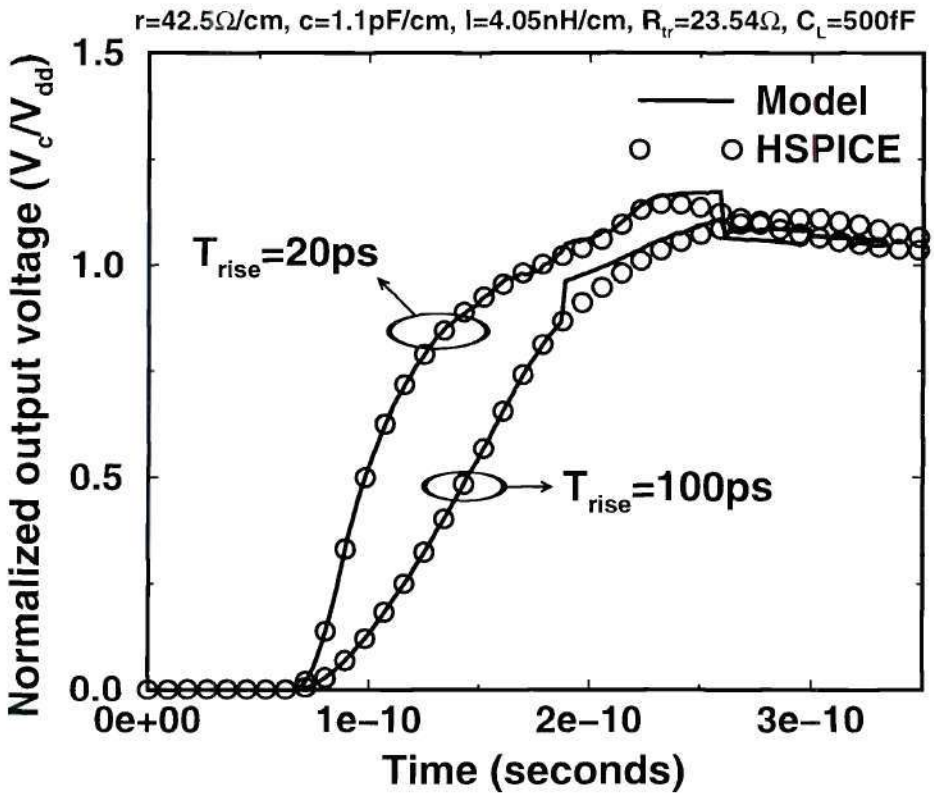


Figure 4.6: Output transient response of a 1cm long distributed *rlc* interconnect to a ramp input waveform- complete model vs. HSPICE simulations for $C_L = 500fF$, and input rise times, T_{rise}, of $20ps$ and $100ps$.

4.4 Crosstalk induced in coupled lines

The crosstalk induced in two-coupled lines which are flanked by shielding power/ground lines, as in Figure 4.1b, is modeled in this section. Using an analysis similar to that in [29], the decoupled PDE's for the common (plus) and differential (minus) modes of voltage wave propagation in two-coupled lines can be written as

$$\frac{\partial^2}{\partial x^2} V_+ (x,t) = (l_s + l_m)\, c_g \frac{\partial^2}{\partial t^2} V_+ (x,t) + r c_g \frac{\partial}{\partial t} V_+ (x,t) \tag{4.207}$$

and

$$\frac{\partial^2}{\partial x^2} V_- (x,t) = (l_s - l_m)\,(c_g + 2c_m)\, \frac{\partial^2}{\partial t^2} V_- (x,t) + r\,(c_g + 2c_m)\, \frac{\partial}{\partial t} V_- (x,t)\,, \tag{4.208}$$

where $V_+(x,t) = V_Q(x,t) + V_A(x,t)$ and $V_-(x,t) = V_A(x,t) - V_Q(x,t)$, $V_A(x,t)$ and $V_Q(x,t)$ being the active and quiescent line transient voltages. Using the initial and boundary conditions, the solutions for the common and differential modes can be written as

$$V_+(x,t) = V_{fin}\,(r, l_s + l_m, c_g, x, t) \tag{4.209}$$

$$V_-(x,t) = V_{fin}\,(r, l_s - l_m, c_g + 2c_m, x, t)\,, \tag{4.210}$$

where $V_{fin}(r,l,c,x,t)$ is the transient response of a single distributed *rlc* line given by (4.145)-(4.146). The transient solutions for the active and quiescent line are given by

$$V_A(x,t) = \frac{V_+(x,t) + V_-(x,t)}{2} \tag{4.211}$$

$$V_Q(x,t) = \frac{V_+(x,t) - V_-(x,t)}{2} \tag{4.212}$$

Equation (4.212) gives the crosstalk induced in a quiescent line, and it is compared to HSPICE simulations in Figure 4.7. Again, the error between the crosstalk model and HSPICE simulations is negligible (less than 1%) for a wide range of input parameters for $t \leq 5t_f$.

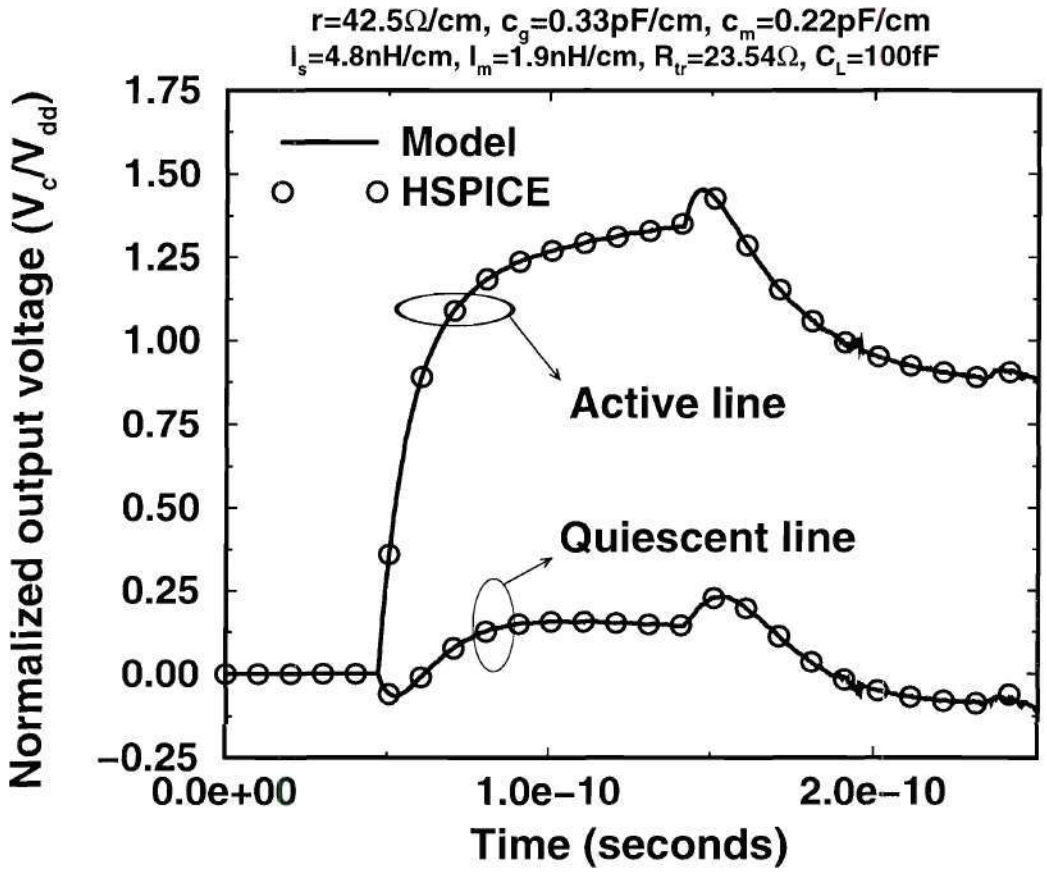


Figure 4.7: Transients in 1 cm long two coupled distributed *rlc* lines - complete model vs. HSPICE simulations for active and quiescent lines.

4.5 Summary

A highly accurate and explicit solution has been rigorously derived for the transient response of single and two-coupled distributed *rlc* interconnects with a capacitive load termination. These solutions are validated against HSPICE simulations, and will be used to derive a unified model for rise time, repeater insertion and crosstalk in Chapter 5. These new solutions have potential applications in CAD for rapid timing analysis, global wire sizing and repeater insertion, signal integrity estimation, and reliability modeling (e.g. voltage overshoot and high current density concerns). The

C code for the transient response of a single distributed *rlc* line is given in Appendix B.

CHAPTER 5

Compact models for time delay, crosstalk and repeater insertion

5.1 Introduction

A new physical model for the transient output voltage of a distributed *rlc* interconnect with a step input and a capacitive load is described in Chapter 4, which extends earlier work in [28], [29]. The inclusion of a load capacitance and wire inductance allows application of this model to evaluate effects of repeaters in high speed global wires. The crosstalk models described in this work focus on modeling state-of-the-art global interconnect structures such as those described in [53] and [54], where one or two signal interconnects are sandwiched between shielding ground lines (Figure 4.1).

In this chapter, the new model for a distributed *rlc* interconnect is used to derive novel compact expressions for: (1) time delay, (2) peak crosstalk for coupled lines, (3) optimum number and size of repeaters, and (4) time delay for repeater-inserted distributed *rc* and *rlc* lines. These new models are used to define a design space that illustrates the trade-off between number of repeaters and wire cross section for specified delay and crosstalk constraints.

5.2 Unified time delay model - step input

The complete analytical models for the transient response of rlc lines derived in Chapter 4 can be expressed using just the following four basis variables :

$$
\begin{aligned}
R_{ratio} &= \frac{rL}{Z_o} = \frac{\text{line resistance}}{\text{characteristic impedance}} \\
C_{ratio} &= \frac{C_L}{cL} = \frac{\text{load capacitance}}{\text{line capacitance}} \\
T_{ratio} &= \frac{t}{t_f} = \frac{\text{time}}{\text{time of flight}} \\
R_T &= \frac{R_{tr}}{Z_o} = \frac{\text{driver output resistance}}{\text{characteristic impedance}}
\end{aligned} \tag{5.1}
$$

These normalized, dimensionless ratios (called the "ratio notation") provide physical insight into the interplay of various line parameters. Interconnects with quite different line parameters, such as length, size of driver and load, and resistance, capacitance and inductance per unit length, may still have similar values for the basis variables. Hence, their transient characteristics will also be similar, when represented using the normalized ratios. These dimensionless ratios will be used to describe interconnect properties such as time delay and crosstalk.

The voltage at the end of a finite line with a capacitive termination for time $t \leq 3t_f$ is given by (4.88), and is re-written here for convenience :

$$
\begin{aligned}
V_{fin}(L, t_f < t < 3t_f) &= V_{dd} \frac{Z_o}{Z_o + R_{tr}} e^{\frac{-rt}{2l}} \\
&\times \left\{ \begin{array}{l}
P \sum_{i=0}^{\infty} \left[(-M)^i \left(\frac{t - L\sqrt{lc}}{t + L\sqrt{lc}} \right)^{\frac{i+1}{2}} I_{i+1} \left(\sigma \sqrt{t^2 - L^2 lc} \right) \right] \\
+Q \sum_{i=0}^{\infty} \left[(-N)^i \left(\frac{t - L\sqrt{lc}}{t + L\sqrt{lc}} \right)^{\frac{i+1}{2}} I_{i+1} \left(\sigma \sqrt{t^2 - L^2 lc} \right) \right] \\
+\frac{P}{1 - G_s} \sum_{i=0}^{\infty} \sum_{u=1}^{\infty} \left[\begin{array}{l} (-M)^i \left(4 - (1 + G_s)^2 G_s^{u-1} \right) \\ \times \left(\frac{t - L\sqrt{lc}}{t + L\sqrt{lc}} \right)^{\frac{i+u+1}{2}} I_{i+u+1} \left(\sigma \sqrt{t^2 - L^2 lc} \right) \end{array} \right] \\
+\frac{Q}{1 - G_s} \sum_{i=0}^{\infty} \sum_{u=1}^{\infty} \left[\begin{array}{l} (-N)^i \left(4 - (1 + G_s)^2 G_s^{u-1} \right) \\ \times \left(\frac{t - L\sqrt{lc}}{t + L\sqrt{lc}} \right)^{\frac{i+u+1}{2}} I_{i+u+1} \left(\sigma \sqrt{t^2 - L^2 lc} \right) \end{array} \right]
\end{array} \right\}
\end{aligned}
$$

$$\times u_o\left(t - L\sqrt{lc}\right) \tag{5.2}$$

where

$$P = \frac{B\left(B + \sqrt{B^2+4}\right)}{\sqrt{B^2+4}}, Q = \frac{-B\left(B - \sqrt{B^2+4}\right)}{\sqrt{B^2+4}}$$
$$M = \frac{B + \sqrt{B^2+4}}{2}, N = \frac{B - \sqrt{B^2+4}}{2}, B = \frac{2}{\sigma C_L Z_o} \tag{5.3}$$

Let $V_{fin}(L, t_d) = V_{dd}/2$, i.e. $t = t_d$ is the 50% time delay. Since t_d is close to $t_f, t_d - t_f << t_f$. For small arguments $x << n$, the modified Bessel function can be approximated as [59]

$$I_n(x) \approx \frac{1}{n!}\left(\frac{x}{2}\right)^n \quad n > 0 \tag{5.4}$$

Using (5.4) and exponential series expansion, the first term in the parenthesis in (5.2) can be approximated as

$$P\sum_{i=0}^{\infty}\left[(-M)^i\left(\frac{t - L\sqrt{lc}}{t + L\sqrt{lc}}\right)^{\frac{i+1}{2}} I_{i+1}\left(\frac{r}{2l}\sqrt{t^2 - L^2lc}\right)\right]$$
$$\approx \frac{-P}{M}\left[e^{-M\frac{r}{4l}\left(t - L\sqrt{lc}\right)} - 1\right] \tag{5.5}$$

Similarly, the remaining terms in parenthesis in (5.2) can be approximated as

$$Q\sum_{i=0}^{\infty}\left[(-N)^i\left(\frac{t - L\sqrt{lc}}{t + L\sqrt{lc}}\right)^{\frac{i+1}{2}} I_{i+1}\left(\sigma\sqrt{t^2 - L^2lc}\right)\right]$$
$$\approx \frac{-Q}{N}\left[e^{-N\frac{r}{4l}\left(t - L\sqrt{lc}\right)} - 1\right] \tag{5.6}$$

$$\frac{P}{1-G_s}\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}(-M)^i\left(4 - (1+G_s)^2 G_s^{u-1}\right)\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}} \\ \times I_{i+u+1}\left(\sigma\sqrt{t^2 - L^2lc}\right)\end{array}\right]$$
$$\approx \frac{P(3+G_s)}{M^2}\left[e^{-M\frac{r}{4l}(t-t_f)} - 1 + M\frac{r}{4l}(t - t_f)\right]u_o\left(t - L\sqrt{lc}\right) \tag{5.7}$$

$$\frac{Q}{1-G_s}\sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}(-N)^i\left(4 - (1+G_s)^2 G_s^{u-1}\right)\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}} \\ \times I_{i+u+1}\left(\sigma\sqrt{t^2 - L^2lc}\right)\end{array}\right]$$

$$\approx \frac{Q}{(1-G_s)}\left[\begin{array}{l} 4\left(e^{\frac{r}{4l}(t-t_f)}-1-\frac{r}{4l}(t-t_f)\right) \\ -\left(\frac{1+G_s}{G_s}\right)^2\left(\begin{array}{l} e^{G_s\frac{r}{4l}(t-t_f)}-1 \\ -G_s\frac{r}{4l}(t-t_f)\end{array}\right)\end{array}\right]u_o\left(t-L\sqrt{lc}\right) \tag{5.8}$$

Since $P >> 1$, $M >> 1$, $Q << 1$ and $N << 1$, the approximations in (5.6), (5.7) and (5.8) can be neglected in comparison to (5.5). Also, $\frac{P}{M} \approx 2$. Setting $t = t_d$, $t_f = L\sqrt{lc}$ and $M = \frac{4l}{rC_L Z_o}$, we can rewrite (5.2) as,

$$V_{fin}(L,t_d) = \frac{V_{dd}}{2} = V_{dd}\frac{2Z_o}{Z_o+R_{tr}}e^{\frac{-rt_d}{2l}}\left[1-e^{-\frac{1}{C_L Z_o}(t_d-t_f)}\right]u_o(t_d-t_f) \tag{5.9}$$

Due to the presence of a load capacitance, the voltage at the end of the line cannot change instantaneously. Therefore, at $t = t_f$, $V_{fin}(L,t_f) = 0$. The 50% time delay is obtained by solving the following equation for t_d :

$$\frac{2Z_o}{Z_o+R_{tr}}\left[1-e^{-\frac{1}{C_L Z_o}(t_d-t_f)}\right]e^{\frac{-rt_d}{2l}}u_o(t_d-t_f) = \frac{1}{2}. \tag{5.10}$$

Two special case of (5.10) are first investigated - an open circuit termination (i.e. C_L=0) and a lossless line (r=0). These special cases are then used to derive a compact expression for the 50% time delay in a lossy *rlc* line with a capacitive load termination.

5.2.1 Open Circuit Termination

The open circuit termination time delay, $t_{d,oc}$, can be calculated by setting $C_L = 0$ in (5.10) :

$$\frac{2Z_o}{Z_o+R_{tr}}e^{\frac{-rt_{d,oc}}{2l}}u_o(t_{d,oc}-t_f) \geq \frac{1}{2} \tag{5.11}$$

The equality sign in (5.10) has changed to an inequality, because the absence of a load capacitance does not prevent the voltage from rising instantaneously. Therefore, at $t = t_{d,oc}$, $V_{fin}(L,t_{d,oc}) \geq V_{dd}/2$.

For $t_{d,oc} = t_f$, (5.11) can be simplified to

$$\frac{rL}{Z_o} \leq 2 \ln \left(\frac{4Z_o}{Z_o + R_{tr}} \right). \tag{5.12}$$

Since the left side of (5.12) is always non-negative, the following additional constraint is automatically imposed :

$$R_{tr} \leq 3Z_o. \tag{5.13}$$

Equations (5.12) and (5.13) are the same conditions obtained in [28] for time of flight operation of a distributed rlc interconnect with an open circuit termination.

5.2.2 Lossless line with a load capacitance

The time delay of a lossless line, $t_{d,lc}$, can be calculated by setting $r = 0$ in (5.10) :

$$\frac{2Z_o}{Z_o + R_{tr}} \left[1 - e^{-\frac{1}{C_L Z_o}(t_{d,lc} - t_f)} \right] u_o \left(t_{d,lc} - t_f \right) = \frac{1}{2}, \tag{5.14}$$

which can be simplified to

$$t_{d,lc} = t_f + C_L Z_o \ln \left(\frac{4Z_o}{3Z_o - R_{tr}} \right). \tag{5.15}$$

Using a linear best-fit curve, (5.15) can be approximated as

$$t_{d,lc} \approx t_f + C_L \left(0.45 R_{tr} + 0.25 Z_o \right), \tag{5.16}$$

which can also be written using the "ratio notation" of (5.1) as

$$\frac{t_{d,lc}}{t_f} \approx 1 + \frac{C_L}{cL} \left(0.45 \frac{R_{tr}}{Z_o} + 0.25 \right), \tag{5.17}$$

or

$$t_{d,lc,ratio} = \frac{t_{d,lc}}{t_f} \approx 1 + C_{ratio} \left(0.45 R_T + 0.25 \right). \tag{5.18}$$

Equations (5.17) and (5.18) give the time delay of a lossless line ("lc" line) with a load capacitance, and they are compared to HSPICE simulations in Figure 5.1.

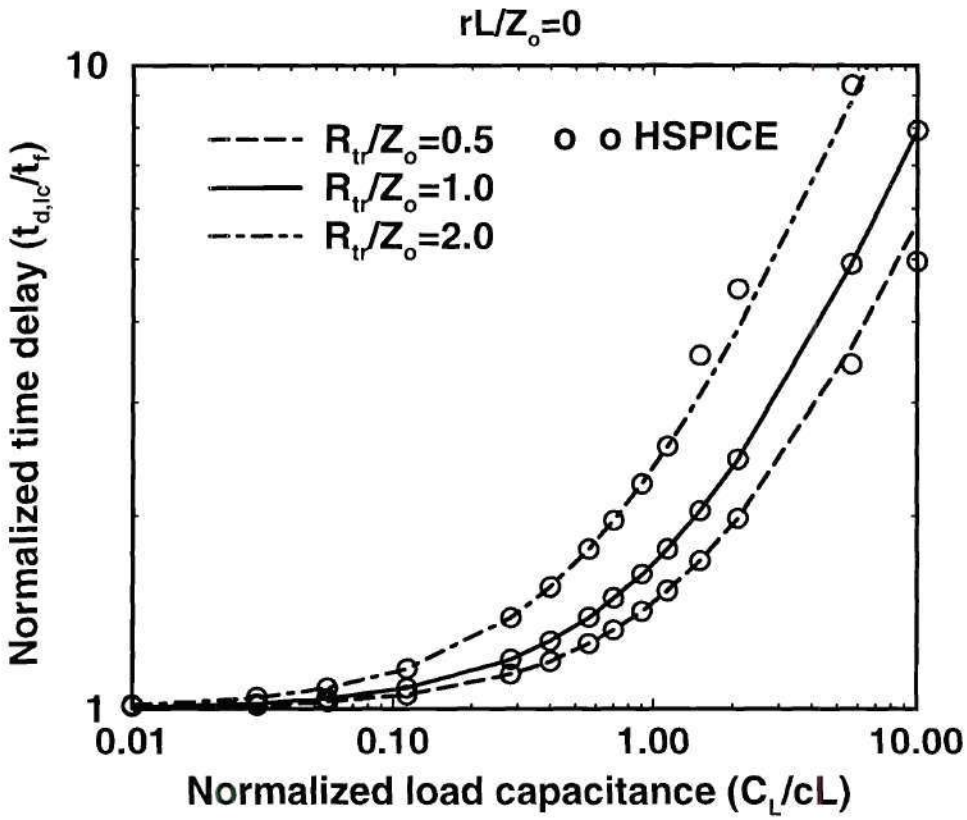


Figure 5.1: Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized load capacitance ($\frac{C_L}{cL}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$), for a lossless line ($\frac{rL}{Z_o} = 0$).

5.2.3 Lossy rlc line with load capacitance

The normalized time delay, computed using the complete transient expressions in Chapter 4, is plotted against the normalized line resistance in Figure 5.2, showing an almost linear relationship between the two quantities. Therefore, the time delay expression in (5.17) can be written as

$$\frac{t_{d,rlc}}{t_f} = 1 + \frac{C_L}{cL}\left(x\frac{rL}{Z_o} + 0.45\frac{R_{tr}}{Z_o} + 0.25\right), \tag{5.19}$$

or

$$t_{d,rlc} = t_f + C_L\left(x\ rL + 0.45R_{tr} + 0.25Z_o\right), \tag{5.20}$$

where $x\frac{C_L}{cL}$ is the magnitude of the slope of the curves in Figure 5.2.

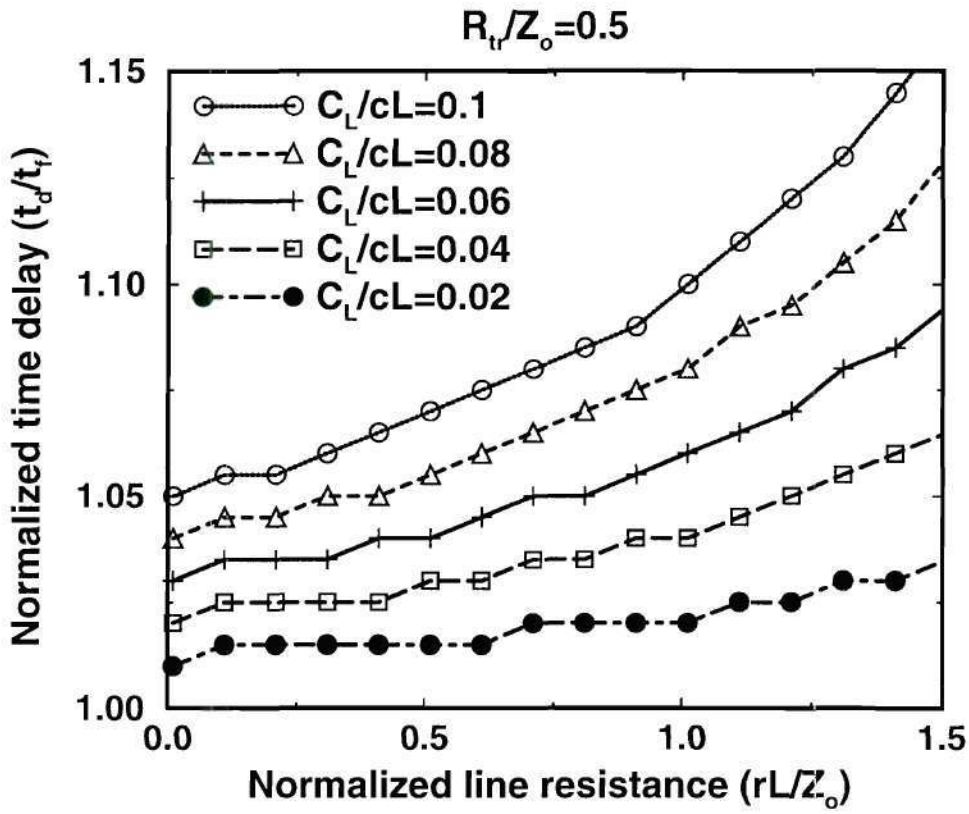


Figure 5.2: Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$) and normalized load capacitance ($\frac{C_L}{cL}$).

To estimate the value of x, consider the rc delay model of [21] given by

$$t_{d,rc} = 0.377rcL^2 + 0.693\left(R_{tr}cL + rLC_L + R_{tr}C_L\right). \tag{5.21}$$

For a highly resistive line, the distributed rlc line model should give the same time delay as the rc line model i.e. the models in (5.20) and (5.21) should converge to the same result. Assuming

$$rL >> Z_o,\ \ C_L >> cL,\ \text{ and } rL >> R_{tr} \tag{5.22}$$

(5.20) will reduce to

$$t_{d,rlc} \approx x\ rLC_L, \tag{5.23}$$

and, (5.21) will reduce to

$$t_{d,rc} \approx 0.693\ rLC_L. \tag{5.24}$$

Comparing (5.23) and (5.24), it can be estimated that $x = 0.693$. Substituting this value in (5.20) and simplifying gives the approximate model for time delay of a distributed *rlc* line with load capacitance as

$$t_{d,rlc} = t_f + 0.693C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right) \tag{5.25}$$

5.2.4 Unified Time Delay Model

Inorder to get a unified time delay model for *rc* and *rlc* lines, (5.21) can be modified as

$$t_{d,rc} = 0.377rcL^2 + 0.693R_{tr}cL + 0.693C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right). \tag{5.26}$$

It was implicitly assumed in [21] that $R_{tr} = Z_o$; the modification in (5.26) has relaxed this assumption. The actual time delay for a line is equal to the *rc* delay if resistance dominates, or is equal to the *rlc* delay if inductance dominates. Since every interconnect has a distributed inductance associated with it, the terms *rc* and *rlc* are only suggestive of whether the line is dominantly resistive or inductive. Hence the more general case is the time delay model which will reduce to the *rc* delay if resistance dominates, or to the *rlc* delay if inductance dominates. The time delay of an interconnect is the greater of the *rc* and *rlc* model delay i.e.

$$t_d = \max\left(t_{d,rlc},\ t_{d,rc}\right). \tag{5.27}$$

Therefore, by comparing (5.25) and (5.26), the unified expression for time delay can be written as

$$t_d = \max\left(t_f,\ 0.377rcL^2 + 0.693R_{tr}cL\right) + 0.693C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right). \tag{5.28}$$

It can be interpreted from (5.28) that the rise time consists of two parts : (a) time for the signal to reach the load end of the line, and (b) time to charge up the load capacitance to $V_{dd}/2$. The time for signal propagation through the interconnect is dictated by time-of-flight for *rlc* lines and by time to charge up the line capacitance in *rc* lines. Therefore, in the "max" function, if time of flight t_f dominates, then the line behaviour is inductive, otherwise it is a resistive line.

Using the "ratio notation" of (5.1), (5.28) can be rewritten as

$$\frac{t_d}{t_f} = \max\left(1,\ 0.377\frac{rL}{Z_o} + 0.693\frac{R_{tr}}{Z_o}\right) + 0.693\frac{C_L}{cL}\left(\frac{rL}{Z_o} + 0.65\frac{R_{tr}}{Z_o} + 0.36\right), \quad (5.29)$$

or

$$t_{d,ratio} = \max\left(1,\ 0.377R_{ratio} + 0.693R_T\right) + 0.693C_{ratio}\left(R_{ratio} + 0.65R_T + 0.36\right). \quad (5.30)$$

The normalized unified time delay model in (5.29) is plotted against the normalized line resistance for two different sets of values for $\frac{C_L}{cL}$ and $\frac{R_{tr}}{Z_o}$ in Figure 5.3. It is easily deduced that the unified time delay model in (5.28) has the best matching to HSPICE simulations compared to time delay models in [21] and [30] : for $0.1 \leq \frac{rL}{Z_o} \leq 5$, $0 \leq \frac{C_L}{cL} \leq 0.1$ and $0.25 \leq \frac{R_{tr}}{Z_o} \leq 1$, the error between the unified models of equations (5.28)-(5.29) and HSPICE simulations is about 2%.

5.3 Unified time delay model - ramp input

The voltage at the end of a finite line with a capacitive termination for a ramp input, for time $t \leq 3t_f$ is given by (4.166), and is re-written here for convenience :

$$V_{1A}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o + R_{tr}}e^{-\sigma t}$$

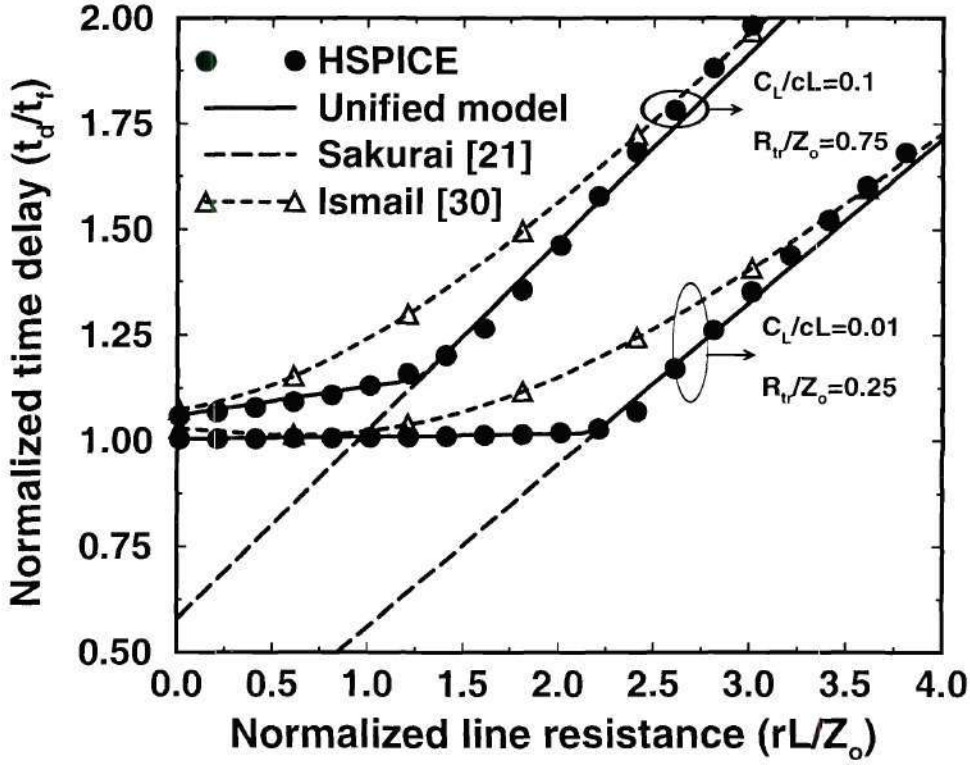


Figure 5.3: Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$) and normalized load capacitance ($\frac{C_L}{cL}$).

$$
\begin{aligned}
&\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(P\left(-M\right)^i+Q\left(-N\right)^i\right)\left(\frac{t-L\sqrt{lc}}{t+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}\\ \times I_{i+u+1}\left(\sigma\sqrt{t^2-L^2lc}\right)\end{array}\right]u_o\left(t-L\sqrt{lc}\right)\\
&-\frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}e^{\sigma T_{rise}}\\
&\times \sum_{i=0}^{\infty}\sum_{u=1}^{\infty}\left[\begin{array}{l}\left(2\left(u-1\right)\frac{-G_s(u+2)+u}{(1-G_s)^2}+\frac{5G_s^2-2G_s+1-G_s^u(1+G_s)^2}{(1-G_s)^3}\right)\\ \times\left(P\left(-M\right)^i+Q\left(-N\right)^i\right)\left(\frac{t-T_{rise}-L\sqrt{lc}}{t-T_{rise}+L\sqrt{lc}}\right)^{\frac{i+u+1}{2}}\\ \times I_{i+u+1}\left(\sigma\sqrt{\left(t-T_{rise}\right)^2-L^2lc}\right)\end{array}\right]\\
&\times u_o\left(t-T_{rise}-L\sqrt{lc}\right) \qquad (5.31)
\end{aligned}
$$

Let the 50% time delay be t_d. Assume that $t_d \leq 3t_f$ and $t_d > t_f + T_{rise}$. (This also implies that $T_{rise} \leq 2t_f$). Therefore, setting $u = 1$, the transient expression can be

approximated as

$$\begin{aligned} V_{1A}(L,t) &= \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t} \\ &\times \left(\begin{array}{l} \sum_{i=0}^{\infty}\left[P(-M)^i\left(\frac{t-t_f}{t+t_f}\right)^{\frac{i+2}{2}} I_{i+2}\left(\sigma\sqrt{t^2-t_f^2}\right)\right] \\ -e^{\sigma T_{rise}}\sum_{i=0}^{\infty}\left[\begin{array}{l} P(-M)^i\left(\frac{t-T_{rise}-t_f}{t-T_{rise}+t_f}\right)^{\frac{i+2}{2}} \\ \times I_{i+2}\left(\sigma\sqrt{(t-T_{rise})^2-t_f^2}\right)\end{array}\right] \end{array} \right) \end{aligned} \tag{5.32}$$

which can be written as

$$\begin{aligned} V_{1A}(L,t) &= \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}\frac{P}{M^2} \\ &\times \left(\begin{array}{l} \sum_{i=0}^{\infty}\left[(-M)^{i+2}\left(\frac{t-t_f}{t+t_f}\right)^{\frac{i+2}{2}} I_{i+2}\left(\sigma\sqrt{t^2-t_f^2}\right)\right] \\ -e^{\sigma T_{rise}}\sum_{i=0}^{\infty}\left[\begin{array}{l} (-M)^{i+2}\left(\frac{t-T_{rise}-t_f}{t-T_{rise}+t_f}\right)^{\frac{i+2}{2}} \\ \times I_{i+2}\left(\sigma\sqrt{(t-T_{rise})^2-t_f^2}\right)\end{array}\right] \end{array} \right) \end{aligned} \tag{5.33}$$

which is approximately equal to

$$\begin{aligned} V_{1A}(L,t) &= \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}\frac{P}{M^2} \\ &\times \left(\begin{array}{l} \sum_{i=0}^{\infty}\left[(-M)^{i+2}(t-t_f)^{i+2}\left(\frac{\sigma}{2}\right)^{i+2}\frac{1}{(i+2)!}\right] \\ -e^{\sigma T_{rise}}\sum_{i=0}^{\infty}\left[(-M)^{i+2}(t-T_{rise}-t_f)^{i+2}\left(\frac{\sigma}{2}\right)^{i+2}\frac{1}{(i+2)!}\right] \end{array} \right) \end{aligned} \tag{5.34}$$

Using the Taylor series expansion of an exponential function, we get

$$\begin{aligned} V_{1A}(L,t) &= \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}\frac{P}{M^2} \\ &\times \left(\begin{array}{l} \left(\exp\left(\frac{-M\sigma}{2}(t-t_f)\right)-1+\frac{M\sigma}{2}(t-t_f)\right) \\ -e^{\sigma T_{rise}}\left(\begin{array}{l} \exp\left(\frac{-M\sigma}{2}(t-T_{rise}-t_f)\right)-1 \\ +\frac{M\sigma}{2}(t-T_{rise}-t_f)\end{array}\right) \end{array} \right) \end{aligned} \tag{5.35}$$

Using $M \simeq B$ and $\frac{P}{M} \simeq 2$, and approximating the exponential function using the first three terms, we get

$$V_{1A}(L,t) = \frac{V_{dd}}{T_{rise}}\frac{2}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}\frac{2}{B}$$

$$\times \quad \left(\left(\frac{1}{2C_L^2 Z_o^2}\left(t-t_f\right)^2\right) - e^{\sigma T_{rise}}\left(\frac{1}{2C_L^2 Z_o^2}\left(t-T_{rise}-t_f\right)^2\right)\right) \quad (5.36)$$

or,

$$V_{1A}\left(L,t\right) = \frac{V_{dd}}{T_{rise}}\frac{1}{\sigma}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}\left(\frac{\sigma}{C_L Z_o}\left(t-t_f\right)^2 - e^{\sigma T_{rise}}\frac{\sigma}{C_L Z_o}\left(t-T_{rise}-t_f\right)^2\right) \quad (5.37)$$

or,

$$V_{1A}\left(L,t\right) = \frac{V_{dd}}{T_{rise}}\frac{1}{C_L Z_o}\frac{Z_o}{Z_o+R_{tr}}e^{-\sigma t}\left(\left(t-t_f\right)^2 - e^{\sigma T_{rise}}\left(t-T_{rise}-t_f\right)^2\right) \quad (5.38)$$

5.3.1 Lossless line

For a lossless line ("lc" line), $r = 0$. This implies that $\varsigma = 0$. Therefore the 50% rise time equation can be written as

$$\frac{V_{dd}}{T_{rise}}\frac{1}{C_L}\frac{1}{Z_o+R_{tr}}\left(\left(t_{d,lc}-t_f\right)^2 - \left(t_{d,lc}-T_{rise}-t_f\right)^2\right) = \frac{V_{dd}}{2} \quad (5.39)$$

which can be simplified to give

$$\frac{1}{C_L}\frac{1}{Z_o+R_{tr}}\left(2\left(t_{d,lc}-t_f\right)-T_{rise}\right) = \frac{1}{2} \quad (5.40)$$

or,

$$t_{d,lc} = t_f + \frac{T_{rise}}{2} + \frac{\left(Z_o+R_{tr}\right)C_L}{4} \quad (5.41)$$

which can be written using the ratio notation of (5.1) as

$$\frac{t_{d,lc}}{t_f} = 1 + \frac{T_{rise}}{2t_f} + \frac{\left(1+\frac{R_{tr}}{Z_o}\right)\frac{C_L}{cL}}{4}, \quad (5.42)$$

or

$$t_{d,lc,ratio} = 1 + \frac{T_{rise,ratio}}{2} + \frac{\left(1+R_T\right)C_{ratio}}{4}. \quad (5.43)$$

5.3.2 Lossy *rlc* line

The time delay of a lossy *rlc* line is expected to be of the form

$$\frac{t_d}{t_f} = 1 + \left(x_1 + x_2\frac{rL}{Z_o} + x_3\frac{R_{tr}}{Z_o} + x_4\frac{C_L}{cL}\right)\frac{T_{rise}}{t_f} + \left(y_1 + y_2\frac{rL}{Z_o} + y_3\frac{R_{tr}}{Z_o}\right)\frac{C_L}{cL}. \quad (5.44)$$

Using heuristical arguments, it can be shown that the coefficients are given as follows :

$$\frac{t_d}{t_f} = 1+\left(0.25 + 0.15\frac{rL}{Z_o} + 0.25\frac{R_{tr}}{Z_o}\right)\frac{T_{rise}}{t_f}+\left(0.25 + 0.693\frac{rL}{Z_o} + 0.45\frac{R_{tr}}{Z_o}\right)\frac{C_L}{cL}, \quad (5.45)$$

which can be written as

$$t_d = t_f+\left(0.25 + 0.15\frac{rL}{Z_o} + 0.25\frac{R_{tr}}{Z_o}\right)T_{rise}+0.693C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right). \quad (5.46)$$

5.3.3 Unified delay model

Using the unified delay model for the step response derived earlier, the expression for the unified delay model for *rc* and *rlc* lines can be derived using heuristic arguments as

$$\begin{aligned} t_d &= \max\left(t_f + 0.15\frac{rL}{Z_o}T_{rise}, 0.377rcL^2 + 0.693R_{tr}cL\right) + 0.25\left(1 + \frac{R_{tr}}{Z_o}\right)T_{rise} \\ &+ 0.693C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right). \end{aligned} \quad (5.47)$$

When $T_{rise} = 0$ (for a step input), (5.47) reduces to (5.28), as expected. Using the ratio notation, (5.47) can be written as

$$\begin{aligned} \frac{t_d}{t_f} &= \max\left(1 + 0.15\frac{rL}{Z_o}\frac{T_{rise}}{t_f},\ 0.377\frac{rL}{Z_o} + 0.693\frac{R_{tr}}{Z_o}\right) + 0.25\left(1 + \frac{R_{tr}}{Z_o}\right)\frac{T_{rise}}{t_f} \\ &+ 0.693\frac{C_L}{cL}\left(\frac{rL}{Z_o} + 0.65\frac{R_{tr}}{Z_o} + 0.36\right), \end{aligned} \quad (5.48)$$

or

$$\begin{aligned} t_{d,ratio} &= \max\left(1 + 0.15R_{ratio}T_{rise,ratio},\ 0.377R_{ratio} + 0.693R_T\right) \\ &+ 0.25\left(1 + R_T\right)T_{rise,ratio} + 0.693C_{ratio}\left(R_{ratio} + 0.65R_T + 0.36\right). \end{aligned} \quad (5.49)$$

The normalized unified time delay model for a ramp input (5.48) is plotted against normalized line resistance $\frac{rL}{Z_o}$ for different sets of line parameters in Figure 5.4. The model in (5.47) has a 2% error when compared with HSPICE simulations for $0.1 \leq \frac{rL}{Z_o} \leq 5$, $0 \leq \frac{C_L}{cL} \leq 0.1$ and $0.25 \leq \frac{R_{tr}}{Z_o} \leq 1$.

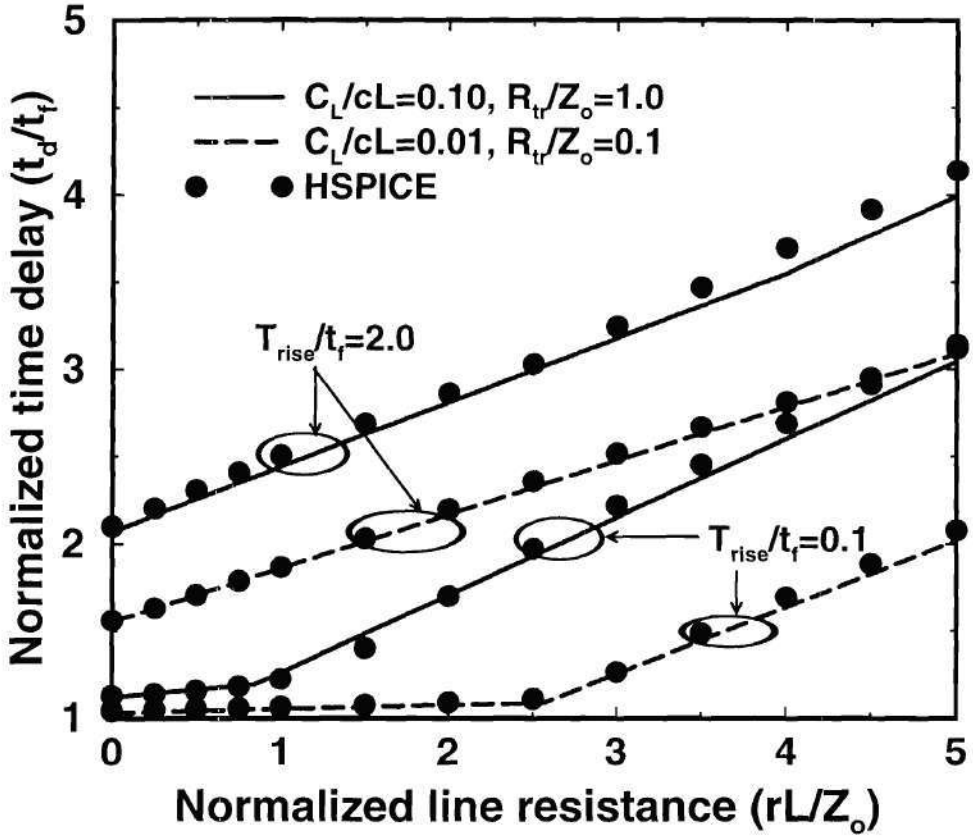


Figure 5.4: Normalized time delay ($\frac{t_d}{t_f}$) vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$), normalized load capacitance ($\frac{C_L}{cL}$) and normalized rise time ($\frac{T_{rise}}{t_f}$), for a ramp input.

5.4 Optimum repeater insertion

Based on an analysis similar to [13] expressions for repeater insertion in distributed *rlc* lines are derived here.

5.4.1 Unified time delay model for repeater insertion

Assume that k repeaters are inserted in a line of length L, and width of each repeater is h times minimum feature size. Also, let the output resistance and input capacitance of a minimum-sized repeater be R_o and C_o respectively. Using (5.25), the time delay (rlc model) of the line is given by

$$t_{d,rlc_rep} = k\left[\frac{t_f}{k} + 0.693hC_o\left(\frac{rL}{k} + 0.65\frac{R_o}{h} + 0.36Z_o\right)\right]. \tag{5.50}$$

Simplifying,

$$t_{d,rlc_rep} = t_f + 0.693C_o\left(hrL + 0.65kR_o + 0.36hkZ_o\right). \tag{5.51}$$

Similarly, the rc time delay model for a repeater-inserted line can be written using (5.26) as

$$t_{d,rc_rep} = \left(0.377\frac{rcL^2}{k} + 0.693\frac{R_ocL}{h}\right) + 0.693C_o\left(hrL + 0.65kR_o + 0.36hkZ_o\right). \tag{5.52}$$

Comparing (5.51) and (5.52), the unified time delay expression for a line with repeaters can be written as :

$$t_{d,rep} = \max\left(t_{d,rlc_rep},\ t_{d,rc_rep}\right) \tag{5.53}$$

or

$$t_{d,rep} = \max\left(t_f,\ 0.377\frac{rcL^2}{k} + 0.693\frac{R_ocL}{h}\right) + 0.693C_o\left(hrL + 0.65kR_o + 0.36hkZ_o\right). \tag{5.54}$$

The "max" function in (5.54) can be used to distinguish between the *rc* and the *rlc* regions. Therefore, if the time of flight for an interconnect segment is greater than the *rc* charging time for its distributed line capacitance, i.e.

$$0.377\frac{rcL^2}{k} + 0.693\frac{R_o cL}{h} \leq t_f, \tag{5.55}$$

or

$$0.377\frac{rL}{kZ_o} + 0.693\frac{R_o}{hZ_o} \leq 1, \tag{5.56}$$

the line behaviour is inductive, otherwise it is resistive, as is borne out by the "max" function in (5.54).

The unified time delay model in (5.54) is compared to the delay models in [13] and [30] in Figure 5.5, in which the normalized time delay is plotted against number of repeaters for a 3cm long interconnect. The delay calculated by HSPICE simulations is also shown in Figure 5.5. The "knee" in the curves of Figure 5.5 is the transition from the *rc* to the *rlc* region, which is determined by the inequality in (5.56). In the region where *rlc* models are to be used, Bakoglu's models [13] severely under estimate the actual delay, due to violation of time of flight constraint. Ismail and Friedman's model [30] for time delay has a good fit to HSPICE simulations only at the extremities, since it is a curve fitted model. The best fit to the HSPICE simulations is given by the new unified model for repeater insertion in *rc* and *rlc* lines, which have a maximum error of about 2%. It is evident from Figure 5.5 that the delay is minimized at the boundary between the *rc* and *rlc* regions. This is the "optimal" repeater insertion design, and it is investigated in the next section.

5.4.2 Optimum repeater insertion

It can be interpreted from (5.54) that for a constant size of repeaters, it is not desirable to insert more repeaters than necessary for time of flight operation of the

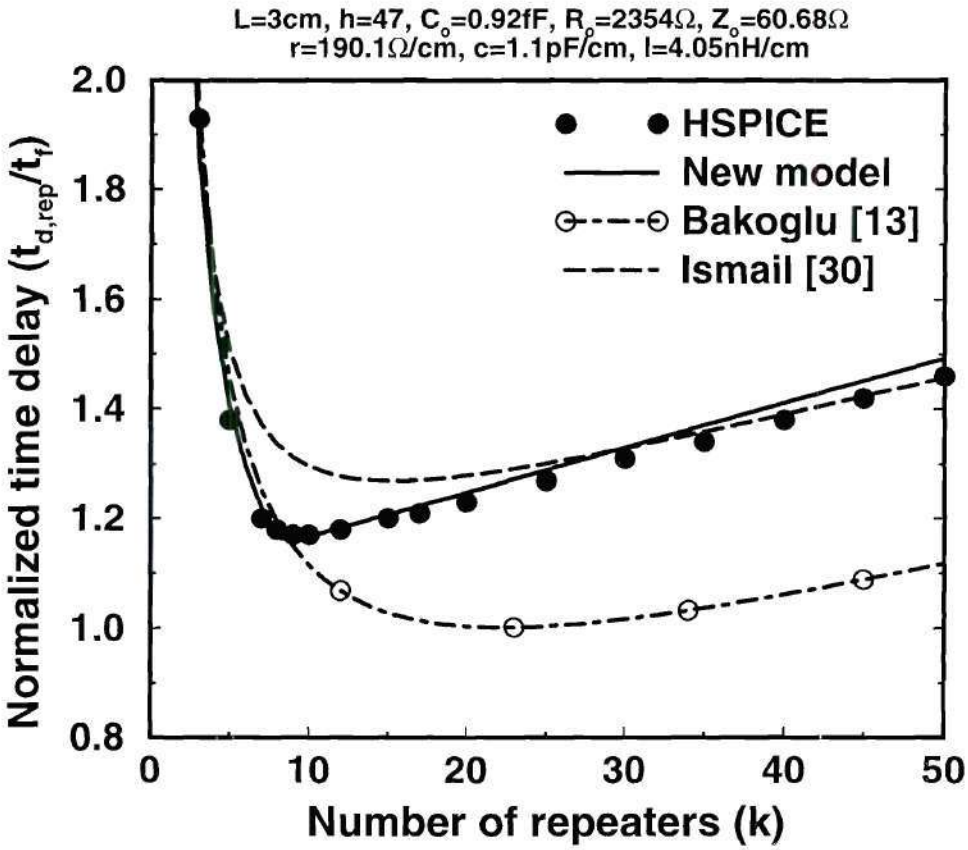


Figure 5.5: Normalized delay ($\frac{t_d}{t_f}$) vs. number of repeaters for a 3cm long line. The new model has an error of about 2% when compared to HSPICE simulations, which is much better than models in [13] and [30].

interconnect segments between each pair of repeaters. This is because adding more repeaters increases total time delay by increasing repeater delay without decreasing interconnect delay. Similarly, it is not desirable to increase size of repeaters, for a constant repeater count, once time of flight operation is achieved for the interconnect segments. Therefore, optimal repeater insertion would entail inserting just sufficient number (and size) of repeaters to operate at the boundary of *rc* and *rlc* region. This implies that the boundary condition for *rc* and *rlc* regions (5.56) should be satisfied in the equality. i.e.

$$0.377\frac{rL}{kZ_o} + 0.693\frac{R_o}{hZ_o} = 1. \tag{5.57}$$

Solving (5.57) for h, we get

$$h = \frac{0.693k\frac{R_o}{Z_o}}{k - 0.377\frac{rL}{Z_o}}. \tag{5.58}$$

Substituting (5.58) in (5.54) and setting the derivative of time delay with respect to k equal to 0, gives the optimum number of repeaters, $k_{opt,rlc}$. Substituting this value in (5.58), gives the optimum size of repeaters $h_{opt,rlc}$:

$$k_{opt,rlc} = 0.9513\frac{rL}{Z_o} \text{ and } h_{opt,rlc} = 1.1479\frac{R_o}{Z_o}, \tag{5.59}$$

or, using the ratio notation of (5.1),

$$k_{opt,rlc} = 0.9513R_{ratio} \text{ and } h_{opt,rlc} = 1.1479R_{o,T}. \tag{5.60}$$

Substituting (5.59) into (5.54), we will get the optimum time delay as

$$t_{d_opt,rlc} = t_f + 1.496\frac{rLR_oC_o}{Z_o}. \tag{5.61}$$

Using the ratio notation of (5.1), the optimal time delay in (5.61) can be written as

$$\frac{t_{d_opt,rlc}}{t_f} = 1.0 + 1.496\frac{rL}{Z_o} \cdot \frac{C_o}{cL} \cdot \frac{R_o}{Z_o}, \tag{5.62}$$

or

$$t_{d_opt,rlc,ratio} = 1.0 + 1.496R_{ratio} \cdot C_{o,ratio} \cdot R_{o,T}. \tag{5.63}$$

The optimum number and size of repeaters, derived using rc models in [13], is :

$$k_{opt,rc} = \sqrt{\frac{0.377rcL^2}{0.693R_oC_o}} \text{ and } h_{opt,rc} = \sqrt{\frac{R_oc}{C_or}}, \tag{5.64}$$

and the corresponding optimum time delay is

$$t_{d_opt,rc} = 2.5\sqrt{R_oC_orcL^2}. \tag{5.65}$$

If the repeaters are designed using the optimum rc repeater models in (5.64) and (5.65), there is a possibility of violating the time of flight constraint - in which case

the optimum time delay in (5.65) will under-estimate the actual delay of the line. The boundary condition for validity of the repeater models is obtained by substituting (5.64) in (5.55), giving the following condition for optimum repeater insertion :

Condition I : "If $1.332\sqrt{\frac{rL}{Z_o}\frac{C_o}{cL}\frac{R_o}{Z_o}} \leq 1$, i.e. $1.332\sqrt{R_{ratio}C_{o,ratio}R_{o,T}} \leq 1$, use optimum *rlc* repeater insertion equations in (5.59)-(5.63); otherwise, use optimum *rc* repeater insertion equations in (5.64)-(5.65)."

Condition I implies that the number and size of repeaters that will minimize total delay of the interconnect and its repeaters is either the optimum *rc* design or the optimum *rlc* design, depending on the line characteristics such as length and resistance per unit length. For example, the optimal *rlc* design may result in a greater delay compared to the optimum *rc* design for a long and highly resistive line, because of the delay of the large number of repeaters required to achieve time of flight operation for the interconnect segments. The optimum number of repeaters and size of repeaters is plotted against normalized line resistance ($\frac{rL}{Z_o}$) in Figures 5.6 and 5.7, respectively. The new model advocates a smaller optimum number and size of repeaters, which leads to considerable reduction in on-chip repeater area, repeater power dissipation and via blockage. On the other hand, using the Bakoglu (*rc*) optimum size and number for the repeaters, might result in a greater time delay, compared to using the *rlc* optimum size and number of repeaters.

5.5 Crosstalk in two coupled lines

A novel combination of the distributed *rlc* line analysis for the transient voltage induced in the quiescent line and a lumped element circuit representation for the charge up of the load capacitance is used to derive a compact crosstalk model.

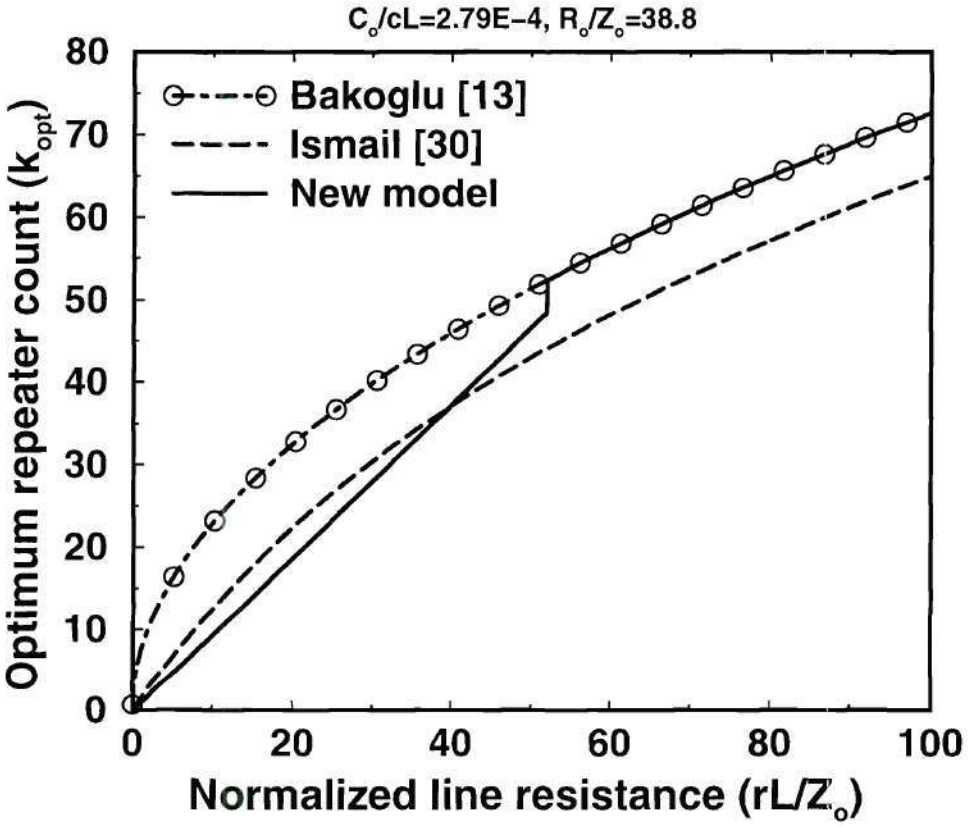


Figure 5.6: Optimum number of repeaters vs. normalized line resistance ($\frac{rL}{Z_o}$).

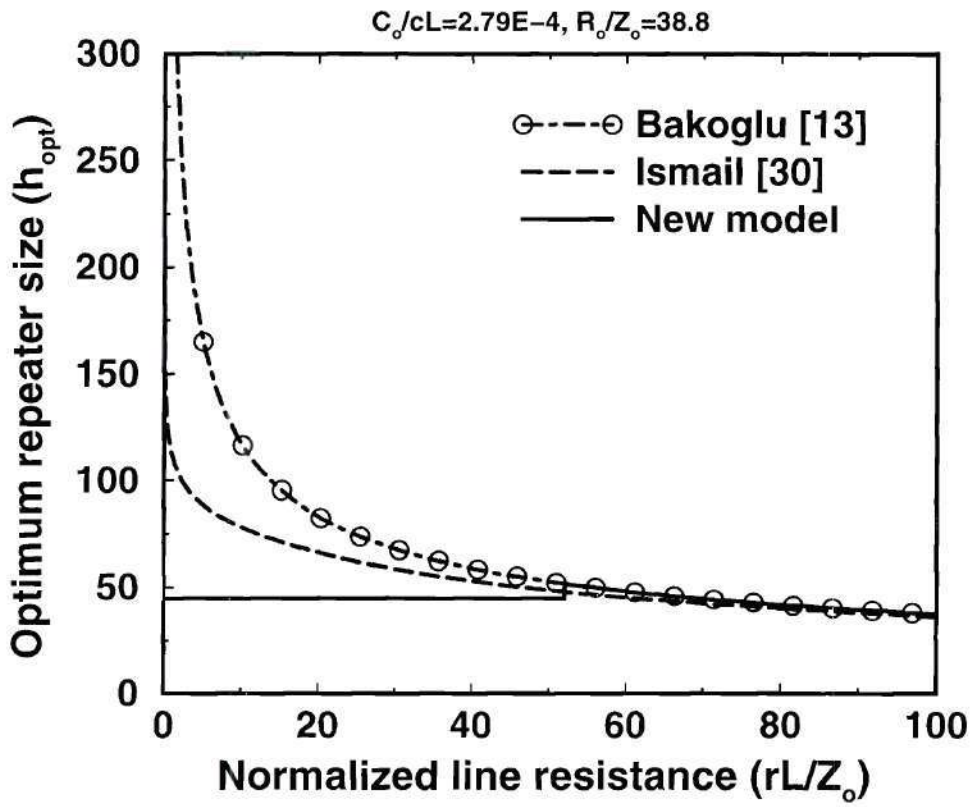


Figure 5.7: Optimum size of repeaters vs. normalized line resistance ($\frac{rL}{Z_o}$).

For convenience, the *rlc* time delay model in (5.25) is rewritten here as :

$$t_d = t_f + 0.693C_L\left(r_{int}L + 0.65R_{tr} + 0.36Z_o\right) \tag{5.66}$$

For a simple *rc* charging circuit (Figure 5.8a), the output voltage (V_o) and 50% rise time ($t_{d,50\%}$) are given by

$$V_o(t) = V_{dd}\left(1 - e^{-\frac{t}{RC}}\right) \text{ and } t_{d,50\%} = 0.693RC \tag{5.67}$$

Comparing (5.66) and (5.67), the lumped element equivalent circuit model of the quiescent *rlc* line can be represented by Figure 5.8b. Therefore, the transients induced in the quiescent line are being approximated as an *rc* charging of the load capacitance. The effective impedance in the current path is

$$R = rL + 0.65R_{tr} + 0.36Z_o \tag{5.68}$$

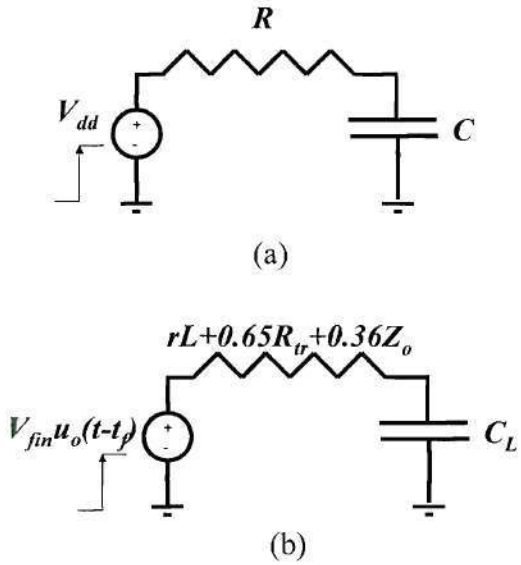


Figure 5.8: (a) Typical *rc* charging circuit (b) Equivalent circuit representation used for approximate crosstalk model.

and the magnitude of the voltage source (V_{fin}) is equal to the voltage induced at a point on the quiescent distributed *rlc* line which is just before the load capacitance. This voltage can be modelled as $V_{inf}(x = L, t = t_f)$, the voltage induced at $x = L$ in an infinitely long quiescent line at $t = t_f$ [29]. V_{fin} is the difference between the common and differential mode solutions to the decoupled partial differential equation system, and it can be written as :

$$V_{fin} = V_{inf}(x = L, t = t_f) = V_{dd}\left(\frac{e^{-\frac{rL}{2Z_o^+}}}{1+\frac{R_{tr}}{Z_o^+}} - \frac{e^{-\frac{rL}{2Z_o^-}}}{1+\frac{R_{tr}}{Z_o^-}}\right) u_o\left(t - t_f\right). \tag{5.69}$$

Therefore, the voltage across the load capacitor of the quiescent line can be written as

$$V_Q(t) = V_{fin}\left(1 - e^{-\frac{t-t_f}{C_L(rL+0.65R_{tr}+0.36Z_o)}}\right) u_o\left(t - t_f\right) \tag{5.70}$$

Assume that the peak crosstalk occurs at $t = t_{peak}$, when the derivative of (5.70) with respect to t is equal to slope S. It was observed experimentally that the error is

minimal for $|S| = |0.5/t_f|$. Solving

$$\left|\frac{\delta V_Q(t)}{\delta t}\right| = \left|\frac{0.5V}{t_f}\right| \tag{5.71}$$

for t_{peak} gives :

$$t_{peak} = t_f + C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right)\ln\left(\frac{V_{fin}}{\frac{0.5}{t_f}C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right)}\right). \tag{5.72}$$

Substituting (5.72) in (5.70) gives the peak crosstalk $V_{p,rlc}$:

$$V_{p,rlc} = V_{fin} - \frac{0.5}{t_f}C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right). \tag{5.73}$$

For distributed rc lines with a load capacitance, Sakurai provides the following expression for peak crosstalk [21] :

$$V_{p,rc} = -K_1 V_{dd}\left(\frac{1}{1+2\eta}\right)^{1/2\eta}\left(\frac{\eta}{1+2\eta}\right) \tag{5.74}$$

where

$$K_1 = -1.01\frac{R_T + C_T + 1}{R_T + C_T + \pi/4}, \tag{5.75}$$

and

$$R_T = \frac{R_{tr}}{rL}, C_T = \frac{C_L}{cL} \text{ and } \eta = \frac{c_c}{c}, \tag{5.76}$$

where c_c is the coupling or mutual capacitance per unit length. (Note: the negative sign in (5.74) is missing in the model in [21]).

However, this expression is not very accurate for rc lines with large load capacitance. In order to get a more accurate expression, we will use the same method used to get the compact crosstalk expression for rlc lines, but base it on Sakurai's models for rc lines.

Based on this analysis, the new rc expression for crosstalk is

$$\begin{aligned}\frac{V_{p,rc}}{V_{dd}} &= 0.58\frac{R_T + C_T + 1}{R_T + C_T + \pi/4}\left[\begin{array}{l}\exp\left(-\frac{1.04}{R_TC_T + R_T + C_T + (2/\pi)^2}\frac{1}{1+2\eta}\right)\\ -\exp\left(-\frac{1.04}{R_TC_T + R_T + C_T + (2/\pi)^2}\right)\end{array}\right]\\ &- \frac{0.05}{t_f}C_L\left(rL + 0.65R_{tr} + 0.36Z_o\right)\end{aligned} \tag{5.77}$$

where

$$\sigma_1 = \frac{1.04}{R_T C_T + R_T + C_T + (2/\pi)^2} \tag{5.78}$$

The *rlc* (5.73) and *rc* (5.77) crosstalk models are unified by the following condition :

Condition II : If $L \leq L_{\max}$, $V_{peak} = V_{p,rlc}$, else $V_{peak} = \max(V_{p,rlc}, V_{p,rc})$ where $L_{\max}$ is the solution of $\partial V_{p,rlc}/\partial L = 0$.

The normalized peak crosstalk is plotted against the normalized line resistance for different values of $\frac{R_{tr}}{Z_o}$ and $\frac{C_L}{cL}$ in Figure 5.9, which compares Condition II with HSPICE simulations of a distributed *rlc* line, as well as with a lumped element model similar to [31]; the error between Condition II and HSPICE is less than 10% for practical values of $0.1 \leq \frac{rL}{Z_o} \leq 5$, $0 \leq \frac{C_L}{cL} \leq 0.1$ and $0.25 \leq \frac{R_{tr}}{Z_o} \leq 1$.

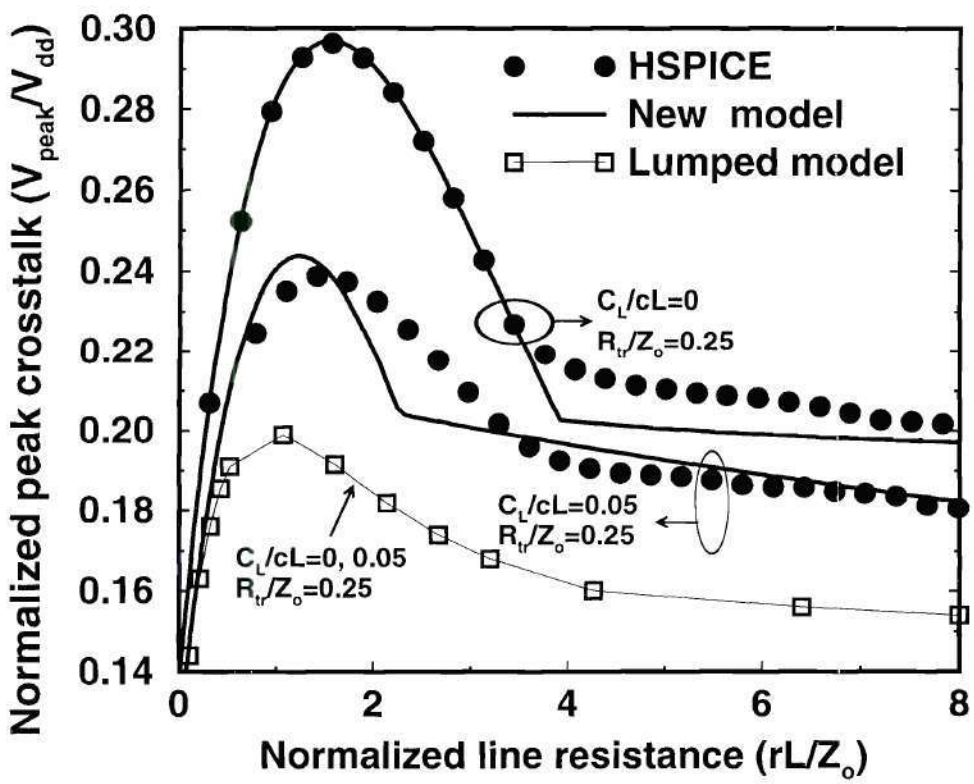


Figure 5.9: Normalized peak crosstalk vs. normalized line resistance ($\frac{rL}{Z_o}$) for different values of normalized driver resistance ($\frac{R_{tr}}{Z_o}$) and normalized load capacitance ($\frac{C_L}{cL}$).

5.6 Delay-Crosstalk design space

Inserting repeaters in global interconnects can cause: 1) an increase in C_{ratio} for each interconnect segment, by adding load capacitance and decreasing the line capacitance (because the segment length is shorter than total wire length), and 2) reduce propagation delay. By reducing wire cross-sectional area (increasing wire resistance), the inductive effects can be mitigated resulting in a decrease in inductive crosstalk. However, the higher line resistance causes propagation delay to increase. An optimal combination of repeater insertion and reduced wire cross-sectional area would decrease inductive crosstalk and wiring area while maintaining constant delay. In fact, for a target delay and peak crosstalk, there is a range of permissible wire resistances (cross sectional areas) for each number of repeaters (Figure 5.10). The upper limit of each range, determined by the delay constraint, maximizes wiring density and minimizes crosstalk while the lower limit, determined by the crosstalk constraint, minimizes propagation delay. Thus the specified design constraints on maximum permissible time delay and crosstalk delineate a feasible region in the "per unit length line resistance - repeater count" design plane in Figure 5.10.

The other design decision that can be aided by Figure 5.10 is determining how many repeaters to use to balance crosstalk and time delay, if the interconnect geometries are fixed. Figure 5.11 shows the trade-off between crosstalk and time delay vs. number of repeaters for a fixed r (or wire cross sectional area). For a 3cm long line, inserting 2 repeaters would decrease time delay but increase crosstalk (point A). This illustrates that unless interconnect repeater circuits are carefully designed, inserting repeaters to reduce delay could increase crosstalk. However, when 10 repeaters are inserted (point B), then the peak crosstalk decreases by 48% with only a 6% delay

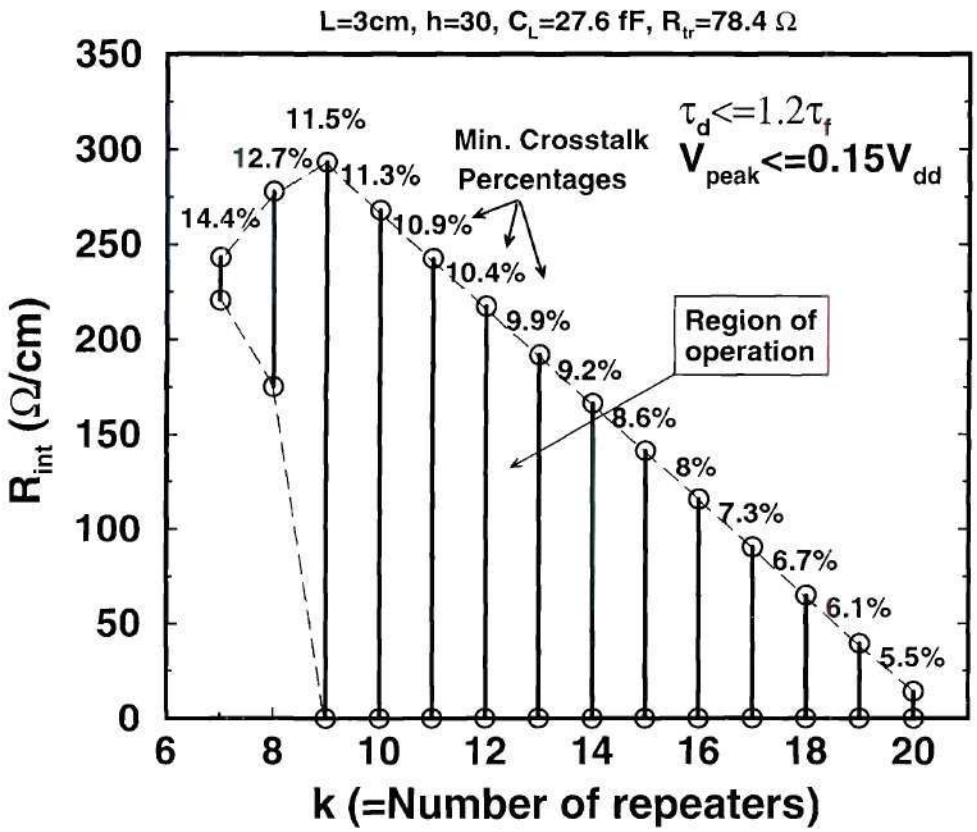


Figure 5.10: Design plane - r vs number of repeaters, showing envelope around the region of operation.

penalty, compared to point A. Table 5.1, illustrates a carefully planned repeater circuit design that inserts 8 repeaters in a 3cm long global interconnect and reduces peak crosstalk by 51% and cross sectional wiring area by 83% without any additional delay penalty, when compared to a line without repeaters.

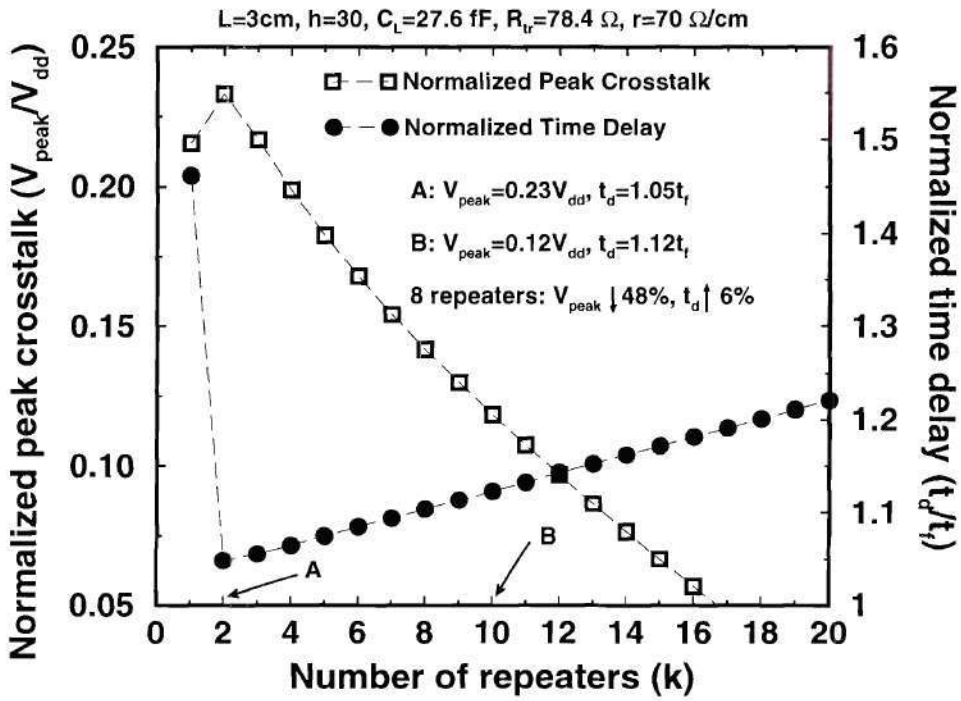


Figure 5.11: Crosstalk and delay vs. number of repeaters, for fixed $r = 70\Omega/cm$. Inserting 8 repeaters (B) decreases crosstalk by 48% and increases time delay by 6%, compared to inserting 2 repeaters (A).

Table 5.1: Using repeaters to decrease crosstalk (and increase wiring density) without penalizing time delay.

# of rep. + driver	r Ω/cm	Time Delay	Peak Cross Talk
1	48.9	$1.2t_f$	23.4%
9	293.5	$1.2t_f$	11.5%

5.7 Global interconnect width optimization

The compact expressions for unified time delay in repeater-inserted *rc* and *rlc* interconnects derived in Section 5.4 have also been used to optimize the design of global interconnects [60], [61]. An optimal on-chip wire width is calculated for global interconnects which offers the best trade-off between "chip bisectional bandwidth" and latency. At the optimal design point, the bisectional bandwidth-reciprocal latency product is maximized, so that the global interconnects can transfer as many bits as possible with low latency. The value of the optimal wire width is shown to be independent of the interconnect length, and is determined only by the resistivity of the metal, the intrinsic delay of repeaters, and the wire geometry. The full derivation of this optimal wire width for global interconnects is provided in [60] and [61].

5.8 Summary

Highly accurate models for transient response of distributed *rlc* lines with a capacitive load, derived in Chapter 4, have been used to derive new, compact expressions for time delay, crosstalk and repeater insertion. For practical ranges of the various line parameters, the maximum error between the time delay and crosstalk models, and HSPICE simulations is shown to be 2% and 10%, respectively. In addition, a novel design plane is described which can be used to determine the number of repeaters to be inserted and the wire geometry to be used to achieve specified delay and crosstalk constraints. It is shown that careful repeater insertion, with the aid of these new physical models, can reduce crosstalk by 51% and wiring cross sectional area by 83% without increasing delay compared to a single driver interconnect circuit.

These compact models for time delay, crosstalk and repeater insertion will be used in Chapter 6 to investigate the impact of inductance on the *n-tier* design architecture.

CHAPTER 6

Effects of inductance on *n-tier* multilevel architectures

6.1 Introduction

The new models for time delay, crosstalk and repeater insertion in high speed interconnects derived in Chapters 4 and 5 are used to analyze the effects of inductance on the multilevel interconnect architecture described in Chapters 2 and 3. Section 6.2 adapts the time delay and repeater insertion models to the *n-tier* architecture, while Section 6.3 does so for the crosstalk expressions. A new case study from the $45nm$ generation is discussed in Section 6.4. The effects of non-ideal return paths on the multilevel architecture are investigated in Section 6.5. The optimum aspect ratio problem is revisited using the new *rlc* interconnect models in Section 6.6.

6.2 Time delay and repeater insertion

The optimum time delay using Bakoglu's *rc* models for repeater insertion is given by (3.4) which is re-written here for convenience :

$$t_{d_opt,RC} = 2.5L\sqrt{rcR_oC_o} \tag{6.1}$$

If sub-optimum number of repeaters are used, then the time delay is given by

$$t_{d_\zeta,RC} = \left(1.4 + 0.53\zeta + \frac{0.53}{\zeta}\right) L\sqrt{rcR_oC_o}, \tag{6.2}$$

where ζ is the fraction of optimum number of repeaters for that particular interconnect.

The optimum time delay using the new *rlc* repeater models derived in Chapter 5 is given by

$$t_{d_opt,RLC} = L\sqrt{lc} + 1.553\frac{rLR_oC_o}{Z_o}. \tag{6.3}$$

The *n-tier* methodology assumes that the time delay of the longest interconnect on any tier is specified as a fraction β of the clock period τ_{clk}. Therefore, (6.2) can be rewritten as

$$\left(1.4 + 0.53\zeta + \frac{0.53}{\zeta}\right) L\sqrt{rcR_oC_o} = \beta\tau_{clk} \tag{6.4}$$

and (6.3) can be rewritten as

$$L\left(\sqrt{lc} + 1.553\frac{rR_oC_o}{Z_o}\right) = \beta\tau_{clk}. \tag{6.5}$$

For designing an interconnect of a particular length (say, the longest interconnect on a tier), with a target delay and number of repeaters in a system belonging to a certain technology generation, all the quantities in (6.4) or (6.5) are known except for r and c, which depend on the wire cross-sectional geometry. Therefore, the utility of (6.4) and (6.5) lies in the ability to determine the wiring cross-section for a given interconnect length, depending on whether it needs to operate in the *rc* or the *rlc* region.

It was proved earlier in Chapter 5 that the time delay of a repeater-inserted interconnect is minimum when each interconnect segment operates at time-of-flight. The condition for using the new optimum *rlc* models for repeater insertion is given

by

$$1.332\sqrt{\frac{rL}{Z_o}\cdot\frac{C_o}{cL}\cdot\frac{R_o}{Z_o}} \leq 1. \tag{6.6}$$

Setting

$$Z_o = \sqrt{\frac{l}{c}} \text{ and } \frac{1}{\sqrt{lc}} = \frac{c_o}{\sqrt{\varepsilon_r}}, \tag{6.7}$$

(6.6) can be simplified to

$$1.332\sqrt{rcR_oC_o}\frac{c_o}{\sqrt{\varepsilon_r}} \leq 1. \tag{6.8}$$

Equation (6.8) imposes the condition for time of flight operation of an interconnect, based only on the wire cross sectional parameters and independent of the wire length. Using (6.4) in (6.8), the condition for time of flight operation for an interconnect of length L, can be rewritten as

$$\frac{1.332}{\left(1.4+0.53\zeta+\frac{0.53}{\zeta}\right)}\frac{\beta\tau_{clk}}{L}\frac{c_o}{\sqrt{\varepsilon_r}} \leq 1, \tag{6.9}$$

or,

$$L \geq \frac{1.332}{\left(1.4+0.53\zeta+\frac{0.53}{\zeta}\right)}\beta\tau_{clk}\frac{c_o}{\sqrt{\varepsilon_r}}. \tag{6.10}$$

Equation (6.10) is the condition that determines whether to use optimum *rc* or *rlc* models for repeater insertion for an interconnect of a given length, in order to satisfy a target time delay constraint. *This is an important condition because it is independent of the wire width, thickness, spacing or height of interlevel dielectric.* Therefore, it is useful to determine which of the metal layers in a multi-level interconnect framework should be designed using the *rc* and *rlc* models. The length $\beta\tau_{clk}\frac{c_o}{\sqrt{\varepsilon_r}}$ can be interpreted as the distance traveled by light in a time which is equal to the maximum permissible delay, and it is a fundamental limit imposed by the speed of light in the dielectric medium.

Let us define the right side of (6.10) as L_{RLC} where

$$L_{RLC} = \frac{1.332}{\left(1.4+0.53\zeta+\frac{0.53}{\zeta}\right)}\beta\tau_{clk}\frac{c_o}{\sqrt{\varepsilon_r}}. \tag{6.11}$$

The new *rlc* models for optimum repeater insertion should be used for all interconnects which are longer than L_{RLC}, irrespective of the width; for all interconnects smaller than L_{RLC}, *rc* models for optimum repeater insertion should be used. Table 6.1 provides L_{RLC} values calculated for different clock frequencies (with $\beta = 0.9$). From Table 6.1, it is evident that *as clock frequencies increase, optimum rlc repeater models will need to be used for shorter and shorter interconnects.*

Table 6.1: L_{RLC} for different clock frequencies.

Clock frequency	L_{RLC}	
	$\zeta = 1.0$	$\zeta = 0.5$
2GHz	3.75cms	3.39cms
6GHz	1.25cms	1.13cms
10GHz	0.75cms	0.68cms
14GHz	0.54cms	0.49cms

In the *n-tier* multilevel interconnect design methodology, the condition stated in (6.10) needs to be applied only for the longest interconnect on each layer. If the longest interconnect on each layer needs optimum *rlc* models, then all the interconnects on that tier will be designed using the optimum *rlc* models. Otherwise, all the interconnects on that tier will be designed using *rc* models.

6.3 Crosstalk

The crosstalk in *rc* and *rlc* lines is given by the following condition :

$$\text{If } L \leq L_{\max},\ V_{peak} = V_{p,RLC},\ \text{else } V_{peak} = \max(V_{p,RLC}, V_{p,RC}), \tag{6.12}$$

where $L_{\max}$ is the solution of

$$\partial V_{p,RLC}/\partial L = 0, \tag{6.13}$$

and $V_{p,RLC}$ and $V_{p,RC}$ are given by (5.73) and (5.77) respectively.

The *n-tier* design methodology accepts a target maximum crosstalk constraint and the interconnects on each tier are designed so that crosstalk is within the permissible limits. If any particular tier does not have any repeaters, then the gate capacitance of the average logic gate is assumed to be the capacitive load at the end of each interconnect on that tier. If repeaters are inserted, then it is assumed that the crosstalk voltage does not propagate through the repeaters (since it is typically less than 20% of V_{dd}). Therefore, only the interconnect segment length between any two adjacent repeaters is considered for crosstalk estimations. Moreover, the capacitive load for each interconnect segment is equal to the input capacitance of the repeater at the end of that segment.

Let the shortest and longest interconnect lengths on any tier be denoted by L_1 and L_2 respectively (i.e. $L_1 < L_2$). And, let L_{ct} be the length of interconnect on that tier which has the maximum crosstalk. The aim now is to determine L_{ct} and re-design the tier so that the crosstalk on an interconnect of length L_{ct} is less than the maximum permissible crosstalk.

First consider the case where no repeaters are inserted in the tier. Then L_{ct} is determined by

$$\begin{aligned} &\text{if} && L_2 < L_{max} && L_{ct} = L_2, \\ &\text{else if} && L_1 < L_{max} < L_2 && L_{ct} = L_{max}, \\ &\text{else} && && L_{ct} = L_1. \end{aligned} \tag{6.14}$$

If repeaters are inserted in some or all the interconnects of that tier, then L_1 and L_2 are set equal to the interconnect segment length between repeaters, and L_{ct} is evaluated using (6.14).

If the crosstalk of L_{ct} is greater than the maximum permissible crosstalk, then the spacing between the interconnects on that tier is increased marginally, while keeping

the other aspect ratios constant. Increasing the spacing reduces the area available for interconnects and therefore the length of the longest interconnect that can be accomodated on that tier decreases. This necessitates a re-computation of the total length of interconnects that can be accomodated on that tier, for the increased spacing between interconnects. The tier is again checked for possible violation of the crosstalk constraint, and the spacing is again increased if required. This process is repeated till all the interconnects on that tier satisfy the crosstalk constraint.

The time delay, crosstalk and repeater insertion models for *rlc* interconnects have been incorporated into the *n-tier* design methodology as described above. In order to highlight the effects of inductance on the multilevel interconnect design, a case study is chosen from the 45nm technology generation. The *rc* design of the multilevel interconnect network of this case study is compared to the *rlc* design in the following section.

6.4 *n-Tier* design case study comparisons

The latest ITRS 2001 edition projects a 45 nm technology node for the year 2010 [6]. The motivation for us to choose a case study that is so far out on the roadmap is to observe considerable inductive effects in the interconnect design process. The main ITRS projections for the 45 nm generation are summarized in Table 6.2.

Some of the questions that designers would ask are : Is it necessary to use all the 10 metal levels for intra-macrocell wiring? If so, where will the inter-macrocell (global) wires be accommodated? Probably the topmost 2 or 4 metal levels should be allocated for the global inter-macrocell wires, while the remaining 8 or 6 metal levels are used for the shorter intra-macrocell wires. What would be the optimum strategy that minimizes chip resources? Which layers will experience significant inductive

Table 6.2: ITRS 2001 projections for 45nm technology generation.

Parameter	Symbol	Value
ITRS year		2010
ITRS technology generation		45nm
Minimum feature size	F	45 nm
On-chip clock frequency (Local and global)	f_c	11.511GHz
Number of metal levels	n	10
Chip size	A_{chip}	310mm^2
ILD dielectric constant	ε_r	2.0
Oxide Thickness	t_{ox}	1.2nm
Supply voltage	V_{dd}	0.6V
Threshold voltage	V_T	0.075V
Number of gates in critical path (assumption)	n_{cp}	5

effects and therefore require additional shielding wires? To answer these questions, different permutations of interconnect metal levels utilization is analyzed in Section 6.4.1 using *rc* models, without repeaters. A representative design is chosen for further optimization using repeaters in Section 6.4.2. Finally, the effects of inductance on this design is investigated in Section 6.4.3.

6.4.1 *rc* models, without repeaters

Let us consider three cases, where 6, 8 and 10 metal layers are used for the intra-macrocell wiring, respectively. First the maximum number of gates per macrocell is estimated using the *n-tier* methodology described in Chapter 2. Then, the minimum area of each macrocell is estimated using the *n-tier* design. These calculations are performed using *rc* delay models, and without using repeaters. Moreover, a unity aspect ratio is assumed for all the metal layers. The results are summarized in Table 6.3. From Table 6.3, it can be inferred that the main difference between the three designs is the number of gates per macrocell, and the number of macrocells per chip; the total number of gates (or transistors) per chip remains approximately the same. A designer would have to decide if he should use a large number of small sized macrocells

Table 6.3: Three different *n-tier* designs for the 45 nm generation, without using repeaters.

		6 level design	8 level design	10 level design
Number of logic gates per macrocell	(N_m)	1.180M	2.66M	4.71M
Clock frequency	(f_c)	11.511GHz	11.511GHz	11.511GHz
Number of metal levels used	(n)	6	8	10
Area of macrocell	(A_m)	0.023cm^2	0.051cm^2	0.09cm^2
Area of logic gates	$(A_{logic\ gates})$	0.016cm^2	0.0395cm^2	0.0742cm^2
Number of repeaters		0	0	0
Total # of logic transistors per macrocell	(=6*Nm)	7.086M	16.02M	28.32M
Total # of macrocells in the chip	$(= 3.1\text{cm}^2/A_m)$	134	60	34
Logic gate density per unit area	$(= N_m/A_m)$	51.34M/cm^2	52.35M/cm^2	52.44M/cm^2
Total # of logic gates in the chip		159.17M	162.29M	162.57M
Total # of transistors in the chip		955.07M	973.76M	975.47M

(and thus have 4 metal layers for routing global inter-macrocell interconnects), or whether he should opt for a smaller number of larger macrocells, and have fewer (2 or 0) levels left for the global interconnects.

According to current literature [36], [37], [62], 2 levels are sufficient for global interconnects. Therefore it seems prudent to choose the second design, where 8 metal levels are used for the intra-macrocell wiring and 2 levels for full chip inter-macrocell wiring. For this case, $2.66M$ logic gates (or approximately 16 million logic transistors) are available in each macrocell. And, 60 such macrocells can be accommodated in a chip with area 3.1cm^2, the assumption being that all the transistors on the chip are logic transistors. The *n-tier* design of each macrocell, without repeaters, is shown in Table 6.4.

However, [62] also shows that the design plane gets smaller and smaller as the technology scales down. This might imply that 2 levels would not be sufficient to accommodate the global wiring levels, especially if additional shielding lines have to

Table 6.4: n-tier wire dimensions for 8 level design of Table 6.3.

	L_{n-1} (cms)	L_n(cms)	Pitch
Tier 4	0.1262	0.4517	730nm
Tier 3	0.0603	0.1262	410nm
Tier 2	0.0148	0.0603	190nm
Tier 1	0.00014	0.0148	90nm

be added to combat the effects of inductance [63]. Hence, any design which provides a larger number of layers for global interconnects would be more welcome.

6.4.2 *rc* models, with repeaters

When repeaters are used, then more logic transistors can be packed into a smaller area, for the same number of metal levels or clock frequency. Depending on the number of repeaters that can be used, the size of each macrocell (both in terms of number of logic gates, and area) varies. Few examples of an *n-tier* design with optimum *rc* repeaters for the 2010 generation macrocell case study with 11.511GHz clock frequency are shown in Table 6.5.

Table 6.5: Five different *n-tier* designs using *rc* models with optimum repeaters for the 45nm technology node generation.

Design #	Number of logic gates	Area	Number of Metal levels	Number of repeaters	Total Power Dissipation per macrocell
A	2.66M	0.051 cm^2	4.57	239,607	9.80 W
B	4.82M	0.077 cm^2	6.87	89,326	16.78 W
C	6.5M	0.135 cm^2	6.45	643,776	28.11 W
D	8.78M	0.191 cm^2	6.96	806,826	40.31 W
E	12.8M	0.310 cm^2	8.40	1,392,600	65.58 W

The logic gate density (number of logic gates per cm^2), total number of macrocells that can be accommodated in a 310mm^2 chip, and total number of logic gates and logic transistors in a 310mm^2 chip, for each of the 5 designs in Table 6.5, are tabulated in Table 6.6. The total power dissipation of each of the five designs of Table 6.6 is much

greater than the ITRS projected value of 288W for full chip power dissipation, for the 45nm technology generation [6]. This observation highlights the need to aggresively develop novel low power circuit design techniques and more efficient heat removal techniques in order to meet the performance targets of the future.

Table 6.6: Logic gate count and density for the five designs of Table 6.5.

Design #	Logic gate density	Number of macrocells	Total # of logic gates	Total # of logic transistors	Total power dissipation
A	52.2M/cm^2	60	159.6M	957.6M	588 W
B	62.6M/cm^2	40	192.8M	1156.8M	671 W
C	48.1M/cm^2	22	143M	858M	618 W
D	46.0M/cm^2	16	140M	840M	645 W
E	41.3M/cm^2	10	128M	768M	655.8 W

Each of these designs has its advantages and disadvantages, as detailed below :

- Design A has the smallest area per macrocell and therefore, 60 macrocells of type A can be accommodated in one chip. The utility of repeaters becomes evident when Design A in Table 6.5 is compared with the 8 metal layer design of Table 6.3 : for the same number of logic gates and macrocell area, using repeaters extensively has decreased the number of metal levels from 8 to less than 5, a reduction of over 37%. Design A provides 5 levels for global inter-macrocell wiring. (The large number of macrocells per chip would require more global wiring area, compared to other designs). It also has the lowest full-chip power dissipation, as well as a high logic gate density. But it uses a relatively large number of repeaters, about 9% of the total number of logic gates.

- Design B has the highest logic gate density and uses the smallest number of repeaters, about 1.2% of number of logic gates. It has the largest number of logic transistors in the full chip (1.1Billion), but it also has the highest full chip power dissipation. Moreover it uses almost 7 levels, leaving 3 metal levels for

global wires that need to connect 40 macrocells.

- Design C has one of the smaller gate densities. It also has one of the highest repeater counts (almost 10% of number of logic gates), and therefore has a high power dissipation. However it provides more than 3 metal levels for global wiring.

- Design D has a large number of logic gates per macrocell, although it still has a relatively good logic gate density. The number of repeaters is also small (about 8% of total number of logic gates). Like Design B, it allows 3 metal levels for global wires, and the overall power dissipation is not very high.

- Design E has the largest number of logic gates per macrocell. It has the smallest logic gate density, and it has the highest number of repeaters (greater than 10% of the number of logic gates). It has a relatively large full chip power dissipation and has less than two levels available for global wires.

The macrocell areas for Designs D and E are large enough that inductance would become important in determining number of repeaters and wire dimensions. Since inductance reduces number of repeaters, these two designs might require a lesser number of repeaters when designed using inductance models. The significantly lesser inductance effects observed in the smaller macrocells might be a convincing argument in favor of using several, smaller macrocells as opposed to using few, large macrocells.

While Design A is small enough to be used in a standard cell library design, D and E are very large macrocells. In fact, E has about $60M$ logic transistors, which is as big as the latest IA-64 Itanium microprocessors. Hence using 10-12 type E macrocells in one chip, would be similar to putting a multi-chip module onto a system-on-package. The design complexity involved in designing large, stand-alone microprocessors might

justify the costs of putting several type E macrocells onto a single chip, giving rise to something like a chip multiple processor (CMP), as opposed to building the large microprocessor by using several smaller building blocks such as Type A macrocells.

On the other hand, if a complete re-design of the microprocessor is carried out, and the designers are able to compartmentalize the design into several small macrocells, then type A might be a viable option.

Another possible permutation would be to use a collection of different types of macrocells i.e. maybe have a couple of type D and E mega-macrocells, together with several of type A and B mini-macrocells. That would really satisfy both the camps - the one which advocates large, macrocells employing the brute force of several million transistors, and the other, which supports decentralizing the logic design into several smaller macrocells, with significant amount of inter-macrocell communication.

For the investigation into the effects of inductance on the *n-tier* design, design type D is chosen as the case study, because it has a large number of gates per macrocell, has a relatively good logic density, and is large enough to demonstrate inductive effects. After determining that inductive effects are indeed significant for this design, the *n-tier* structure will be re-designed so that 1) wire spacing will be increased to reduce mutual capacitance and decrease crosstalk, 2) additional shielding lines will be added on those layers where inductance is important, and 3) the number of repeaters used will be determined using *rlc* models. The effects of inductance on the *n-tier* design will be quantified using this case study.

6.4.3 *rlc* models for delay, crosstalk and repeater insertion

An example case study is chosen from the $45nm$ technology generation of the ITRS 2001 roadmap in order to investigate the impact of inductance on *n-tier* multilevel

interconnect architectures. For this investigation it is assumed that all the aspect ratios are equal to unity. If the target peak crosstalk constraint is not met, then the spacing between the interconnects will be increased, to reduce mutual inductive coupling.

The baseline case study ($N = 8.78M$ logic gates, f_c=11.511GHz, A_c=0.191cm^2, no repeaters, no inductance, and peak crosstalk equal to 20% of supply voltage) has 12.51 metal levels and the spacing to width ratio (S/W)=1 for all metal layers. When repeaters are inserted using rc delay models (i.e. no inductance), the number of levels decreases considerably to 6.96. For a peak crosstalk of 20%, $S/W = 1$ for all metal layers. When inductance is included in the delay calculation using repeaters, the number of levels increases slightly to 7.16. Since the peak crosstalk is 20%, $S/W = 1$ for all metal layers. The interconnect geometries for these three designs is shown in Table 6.7.

Table 6.7: Multilevel interconnect geometries for peak crosstalk=20% of V_{dd}. Crosstalk is calculated using the compact models in Chapter 5.

	No repeaters, rc models		With repeaters, rc models		With repeaters, rlc models	
Number of metal levels	12.51≈14		6.96≈8		7.16≈8	
Number of repeaters	0		0.81M		0.67M	
	Width (μm)	S/W	Width (μm)	S/W	Width (μm)	S/W
Tier 7	0.703	1	-	-	-	-
Tier 6	0.629	1	-	-	-	-
Tier 5	0.437	1	-	-	-	-
Tier 4	0.302	1	0.632	1	0.520	1
Tier 3	0.192	1	0.159	1	0.152	1
Tier 2	0.099	1	0.072	1	0.075	1
Tier 1	0.048	1	0.048	1	0.048	1

The following observations can be deduced from Table 6.7 :

- By inserting repeaters, the wire width can be decreased on the lower metal tiers, resulting in an increased wiring density. This decreases the overall number of

metal layers.

- In the third design (using *rlc* repeaters), the inductance effects are noticeable only on Tier 4. Since the *rlc* models do not violate time of flight constraint, the width of Tier 4 is less than the width of the corresponding Tier 4 of the *rc* repeaters design.

- The slightly larger width on Tier 2 of the *rlc* repeaters design pushes some of the interconnects to the next tier, which eventually causes a slight increase in the number of metal layers for the *rlc* repeaters design.

When the maximum permissible crosstalk is lowered to 10% of the supply voltage, then an increased spacing is necessary on most of the metal layers to meet the target crosstalk constraint. This results in an increased number of metal layers. Without using repeaters, and neglecting inductance, the number of metal layers is 14.09. Using *rc* repeater models, the number of levels decreases to 7.05 and the increased spacing is necessary only on the topmost tier (because the repeaters act as capacitive loads decreasing the peak crosstalk of each interconnect segment). When inductance is included in the repeater analysis, then the number of levels increases to 8.003. This is mainly due to the larger spacing required on the top two tiers. The S/W ratios for each tier in each of these three cases are shown in Table 6.8, from which the following notes can be observed :

- The capacitive loading of repeaters reduces the crosstalk. Hence, the spacing need not be increased in Tiers 2 and 3 when repeaters are used, in contrast to the first case without any repeaters. Also the width of the interconnects is smaller when repeaters are used. These two factors reduce the number of metal layers.

Table 6.8: Multilevel interconnect geometries for peak crosstalk=10% of V_{dd}. Crosstalk is calculated using the compact models in Chapter 5.

	No repeaters, *rc* models		With repeaters, *rc* models		With repeaters, *rlc* models	
Number of metal levels	14.09≈16		7.05≈8		8.0003≈10	
Number of repeaters	0		0.82M		0.78M	
	Width (μm)	S/W	Width (μm)	S/W	Width (μm)	S/W
Tier 8	0.664	1.53	-	-	-	-
Tier 7	0.678	1.28	-	-	-	-
Tier 6	0.506	1.3	-	-	-	-
Tier 5	0.362	1.43	-	-	0.498	1.35
Tier 4	0.256	1.49	0.601	1.34	0.480	1.35
Tier 3	0.169	1.45	0.159	1	0.158	1
Tier 2	0.094	1.18	0.072	1	0.073	1
Tier 1	0.048	1	0.048	1	0.048	1

- When spacing is increased to reduce crosstalk, the line capacitance decreases. This implies that there is some room to increase the line resistance while maintaining constant *rc* time delay. Therefore, the interconnect width in such cases is seen to decrease marginally, although the pitch (=width+spacing) might still be greater than the unity aspect ratio case. This observation can lead to situations such as that exemplified by the design of Tier 8 (without repeaters design), which has a smaller width than Tier 7, due to the much larger S/W ratio. The pitch of Tier 8, however, is larger than that of Tier 7.

The designs in Tables 6.7 and 6.8 were executed using compact models for crosstalk developed in Chapter 5. Now consider the case where every tier is designed for the *worst case* peak crosstalk. Using *rc* models the worst case peak crosstalk is given by

$$\frac{V_{worstcase,rc}}{V_{dd}} = 0.5\frac{c_m}{c_m + c_{gnd}}. \tag{6.15}$$

Using *rlc* models, the worst case peak crosstalk is given by

$$\frac{V_{worstcase,rlc}}{V_{dd}} = \frac{\pi}{4}\frac{c_m}{c_m + c_{gnd}}. \tag{6.16}$$

For a peak crosstalk of 20% of V_{dd}, without using repeaters, and neglecting inductance, the number of levels is 12.52, a marginal increase over the earlier case (Table 6.7), due to the fact that the S/W ratio has increased to 1.01 for all the tiers. When rc repeaters are used, the number of levels is 6.965, due to an increased $S/W = 1.01$ for all the tiers. When inductance is included in the repeater insertion models, the number of levels is 8.37, with $S/W = 1.45$ for all the tiers. The interconnect geometries for these cases is shown in Table 6.9.

Table 6.9: Multilevel interconnect geometries for *worst case* peak crosstalk=20% of V_{dd}. Crosstalk is calculated using equations (6.15) and (6.16).

	No repeaters, rc models		With repeaters, rc models		With repeaters, rlc models	
Number of metal levels	12.52≈14		6.97≈8		8.37≈10	
Number of repeaters	0		0.81M		0.99M	
	Width (μm)	S/W	Width (μm)	S/W	Width (μm)	S/W
Tier 7	0.702	1.01	-	-	-	-
Tier 6	0.624	1.01	-	-	-	-
Tier 5	0.434	1.01	-	-	0.494	1.45
Tier 4	0.301	1.01	0.631	1.01	0.229	1.45
Tier 3	0.192	1.01	0.158	1.01	0.119	1.45
Tier 2	0.099	1.01	0.072	1.01	0.056	1.45
Tier 1	0.048	1.01	0.048	1.01	0.045	1.45

Some important observations from Table 6.9 are :

- It is evident that when inductance is included in the analysis, an increased spacing is required to meet the worst case crosstalk constraints compared to using rc models, with and without repeaters.

- The pitch of each tier is equal to $(1 + S/W) * Width$. Hence although the width of the interconnects on Tier 1 in the rlc repeaters design is smaller than that of interconnects on Tier 1 of the rc repeaters design, the pitch is larger due to the higher S/W ratio of the former.

- In the third case (which includes inductance), *rlc* models are used for repeaters insertion only on Tier 5. However, the S/W ratio needs to be increased for all tiers starting from Tier 1 to meet the worst case crosstalk constraint. This shows that the effects of inductance on crosstalk become significant much earlier compared to the effects of inductance on time delay, as has been postulated in [64].

When the peak crosstalk decreases to 10% of V_{dd}, the number of levels increases to 14.90 when no repeaters are used and inductance is neglected; and, $S/W = 1.68$ for all the tiers. When *rc* repeaters are used, then the number of levels is 8.65, and all the tiers still have $S/W = 1.68$. When inductance is included in the repeater calculations, then the number of levels is 10.27 and the spacing is increased on all tiers to $S/W = 2.11$. The interconnect geometries for these designs is shown in Table 6.10.

Table 6.10: Multilevel interconnect geometries for *worst case* peak crosstalk=10% of V_{dd}. Crosstalk is calculated using equations (6.15) and (6.16).

	No repeaters, *rc* models		With repeaters, *rc* models		With repeaters, *rlc* models	
Number of metal levels	14.90≈16		8.65≈10		10.27≈12	
Number of repeaters	0		1.12M		1.15M	
	Width (μm)	S/W	Width (μm)	S/W	Width (μm)	S/W
Tier 8	0.659	1.68	-	-	-	-
Tier 7	0.541	1.68	-	-	-	-
Tier 6	0.403	1.68	-	-	0.481	2.11
Tier 5	0.298	1.68	0.639	1.68	0.259	2.11
Tier 4	0.210	1.68	0.197	1.68	0.148	2.11
Tier 3	0.133	1.68	0.106	1.68	0.079	2.11
Tier 2	0.065	1.68	0.050	1.68	0.045	2.11
Tier 1	0.045	1.68	0.045	1.68	0.045	2.11

The following observations can be made from Table 6.10 :

- When inductance is included in the design of the *n-tier* multilevel architecture,

the number of levels increases due to increased spacing between the interconnects to meet the target crosstalk constraints. This is especially true if the maximum permissible crosstalk is very restrictive e.g. 10% of supply voltage. Also the number of repeaters and size of repeaters is different when inductance is included.

- Designing all the tiers so that the worst case peak crosstalk is less than the specified value can cause significant increase in number of metal layers required.

6.5 Non-ideal return paths

In this section, the effects of non-ideal current return paths on the multilevel interconnect architecture geometry is investigated. So far, it was assumed that an ideal current return path was present in the close vicinity of the signal interconnect. The orthogonal interconnects on the metal layers above and below the plane of interest were approximated as ground planes. Therefore, the crosstalk induced in a quiescent line by active neighbours was calculated using a 3-coupled line, 2 ground plane model. This configuration is referred to as the SSS configuration (S=Signal line) and it is depicted in Figure 6.1.

The ground (c_g) and mutual (c_m) capacitances are calculated using analytical models in [41]. The self and mutual inductances for three coupled lines is evaluated using the following matrix relation :

$$\begin{bmatrix} l_{11} & l_{12} & l_{13} \\ l_{12} & l_{22} & l_{23} \\ l_{13} & l_{23} & l_{33} \end{bmatrix} \begin{bmatrix} 2c_g + c_m & -c_m & 0 \\ -c_m & 2c_g + c_m & -c_m \\ 0 & -c_m & 2c_g + c_m \end{bmatrix} = \frac{\varepsilon_r}{c_o^2} [I] . \tag{6.17}$$

The worst case crosstalk on the inner quiescent line occurs when all the three conductors are initially low, and both the outer conductors switch high. The transient

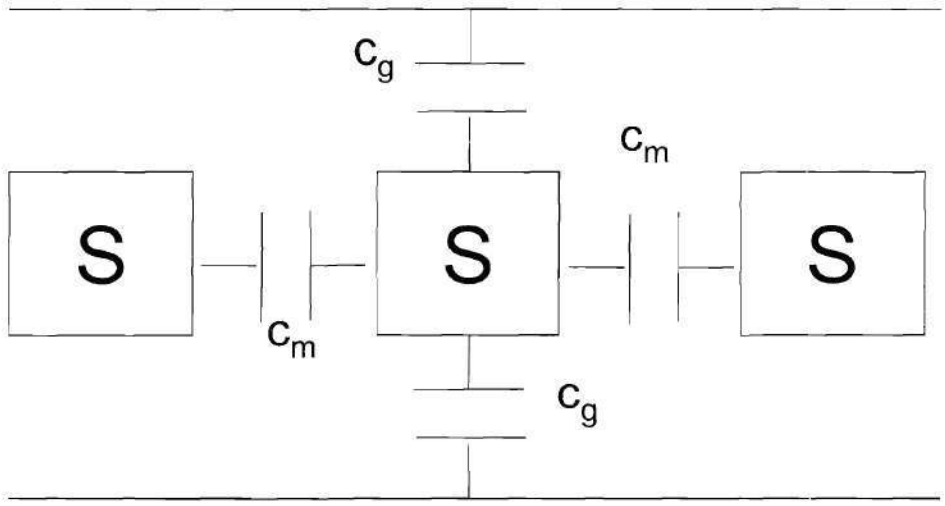


Figure 6.1: 3-coupled lines (SSS) configuration, with ideal return paths. Orthogonal layers are approximated as ground planes.

expression for the worst case crosstalk on the inner conductor is given by

$$V_Q(l,t) = \frac{2}{3}\begin{pmatrix} V_{fin}\left(L,t,l=\frac{1}{2c_g v^2}, c=2c_g\right) \\ -V_{fin}\left(L,t,l=\frac{1}{(2c_g+3c_m)v^2}, c=2c_g+3c_m\right) \end{pmatrix}. \tag{6.18}$$

In real on-chip conditions, an ideal return path is not always present nearby. Therefore, the capacitance and inductance matrices no longer obey the simple relation $[L][C] = \frac{1}{v^2}[I]$. The flux linkage between adjacent signal lines increases because the current return path is not nearby. Therefore, the crosstalk induced in a quiescent line by an adjacent active neighbour will increase. In order to reduce the crosstalk, additional power and ground lines are inserted between the signal lines to shield them from one another [54], [53], [63]. However, the insertion of additional shielding lines will decrease the area available for signal interconnects and hence reduce wiring density. The number of signal lines between any pair of shielding lines is determined by two opposing constraints - the maximum permissible crosstalk and the minimum permissible wiring density. The best design for minimum crosstalk is to alternate signal and power/ground lines i.e. sandwich every signal line between two shielding

lines. A better compromise between wiring density and signal integrity is achieved by inserting two signal lines between every pair of shielding lines. For the purpose of this discussion, this arrangement will be referred to as the GSSG configuration (G=Ground line, S=Signal line), which is depicted in Figure 6.2.

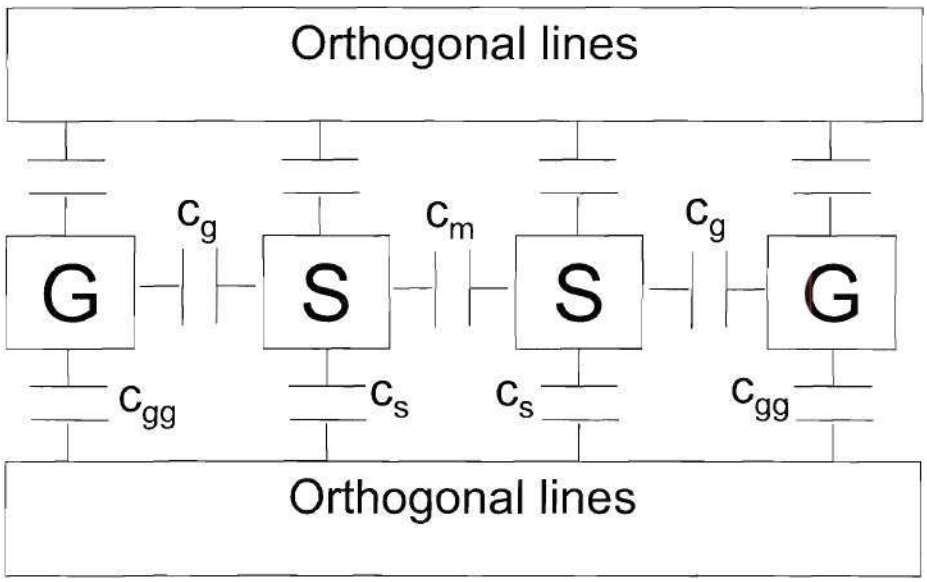


Figure 6.2: 2 signal lines sandwiched between 2 power/ground lines (GSSG configuration), with non-ideal return paths.

The crosstalk induced in the quiescent line by the active neighbour is calculated using 2-coupled line crosstalk models. The line-to-line and line-to-ground capacitances are calculated using Chern's models [41]. The inductance values are calculated using the following matrix relation from [63] :

$$\begin{bmatrix} l_s & l_m \\ l_m & l_s \end{bmatrix} \begin{bmatrix} 2c_g + c_m & -c_m \\ -c_m & 2c_g + c_m \end{bmatrix} = \frac{\varepsilon_r}{c_o^2}\,[I]\,, \tag{6.19}$$

where $c_g = c_m$, because the cross sectional dimensions are the same for both signal and ground lines. The crosstalk induced on one of the signal lines by the adjacent

active neighbour is given by

$$V_Q(l,t) = \frac{1}{2}\begin{pmatrix} V_{fin}(L,t,l=l_s+l_m,c=2c_g) \\ -V_{fin}(L,t,l=l_s-l_m,c=2c_g+2c_m) \end{pmatrix}. \tag{6.20}$$

For both the cases, it is assumed that the interconnect metal thickness T, dielectric height H and interconnect width W are all equal. i.e. $T_{wire} = \frac{T}{W} = 1$, and $H_{ILD} = \frac{H}{W} = 1$. However, the spacing between the adjacent interconnects, $S_{wire} = \frac{S}{W}$, is varied in order to meet the crosstalk constraints. (for the case study, the maximum permissible crosstalk is assumed to 10% of the supply voltage).

The insertion of additional shielding lines reduces the area available for signal interconnects. The wiring efficiency is a product of three factors, and it can be expressed as

$$e_w = (1 - b_{via}) \times \left(1 - b_{p/g/c}\right) \times e_{rout}, \tag{6.21}$$

where e_w is the wiring efficiency, b_{via} is the via blockage, $b_{p/g/c}$ is the blockage due to power/ground/clock lines and e_{rout} is the router efficiency. Since the shielding lines are usually inserted only in the top metal layers, it can be safely assumed that the via blockage on these layers is negligible. i.e. bvia=0. Initially it was assumed that the wiring efficiency is 40%. Assume that the router efficiency is 50%. Therefore, the blockage due to power/ground/clock lines is given by $b_{p/g/c} = 1 - \frac{e_w}{e_{rout}} = 0.2$ or 20%. When two signal lines are inserted between each pair of power/ground lines, the power/ground/clock blockage increases to 33%, or $b_{p/g/c} = 0.33$. Assuming that the same router is used ($e_{rout} = 0.5$), the new wiring efficiency is determined to be $e_w = \left(1 - b_{p/g/c}\right) \times e_{rout} = 0.33$ or 33%. Therefore, the insertion of shielding lines decreases the wiring efficiency on that tier to 33%.

The multilevel interconnect design of the 45nm generation case study for a maximum permissible crosstalk of 10%, assuming ideal return paths, is shown in Table

6.11. The number of metal layers is 8.0003, which corresponds to marginally greater than 4 tiers. The S/W ratio has been increased to 1.35 on Tiers 4 and 5 to satisfy the peak crosstalk constraint. This is evident from Figure 6.3 which shows the crosstalk induced on the quiescent line vs. time. For $S/W = 1.35$, the peak crosstalk is less than 10% of V_{dd}. Since there are no additional shielding lines, the wiring efficiency is equal to 40% for all the metal layers. When non-ideal return paths are considered,

Table 6.11: Multilevel interconnect geometries for worst peak crosstalk=10% of V_{dd}, for ideal return paths assumption.

A_m=0.191cm^2, f_c=11.511GHz, n=8.0003 levels						
Tier #	e_w	L_n (gate pitches)	Width (microns)	pitch (microns)	S/W	$n_{repeaters}$
Tier 5	0.4	5926.21	0.498	1.171	1.35	16
Tier 4	0.4	5134.71	0.480	1.129	1.35	25700
Tier 3	0.4	1614.65	0.158	0.315	1.0	168600
Tier 2	0.4	668.9	0.073	0.145	1.0	585600
Tier 1	0.4	106.58	0.048	0.959	1.0	0

additional power/ground shielding lines are inserted after every pair of signal lines. The top two tiers of the multilevel architecture is re-designed, as shown in Table 6.12. The insertion of additional shielding lines has decreased the wiring efficiency of the top two tiers to 33%. However, the spacing between the interconnects need not be as large as in the previous design, because the co-planar ground lines keep the crosstalk within permissible levels. Therefore the S/W ratio is 1 for Tier 4 and just 0.75 for Tier 5. This is evident from Figure 6.3 which shows the crosstalk induced on the quiescent line, for Tiers 4 and 5, which is less than 10% of V_{dd} for S/W equal to 1 and 0.75 respectively. The interesting point to note here is that as the S/W ratio decreases, the peak crosstalk also decreases! This is because as S/W reduces, the co-planar ground lines also get closer to the signal lines (in addition to the two signal lines coming nearer). Consequently, more flux lines terminate on the ground lines and this outweighs the effect of increase in proximity of the signal lines. Although

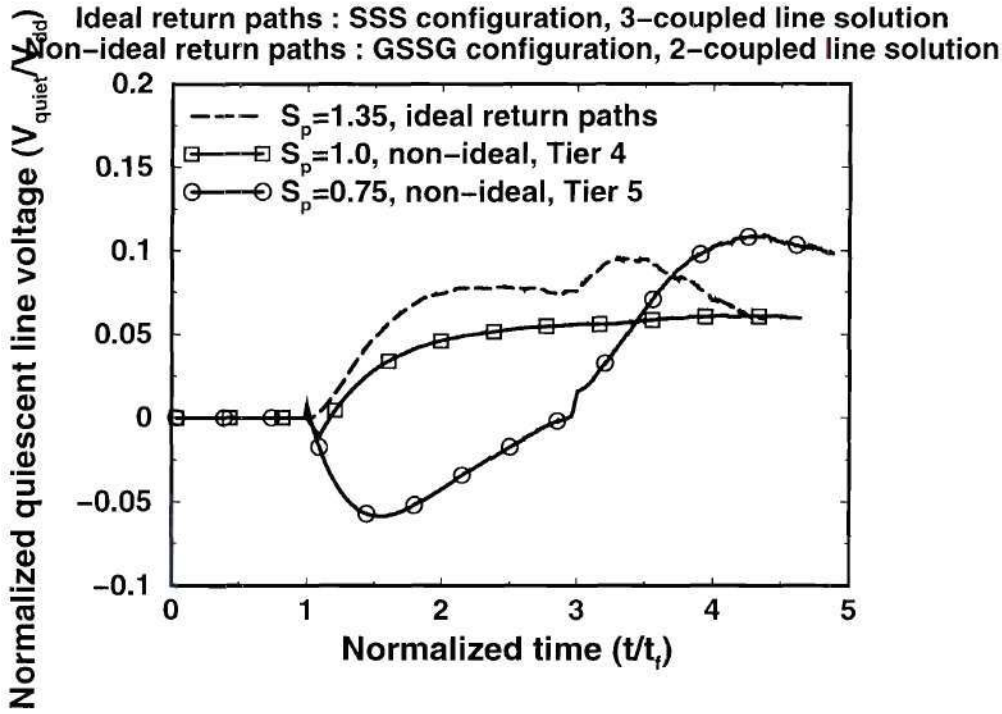


Figure 6.3: Quiescent line transients for 3-coupled (ideal return paths) and 2-coupled (non-ideal return paths) signal lines.

a smaller spacing would increase the wiring density, the absolute value of the wire widths has to increase so that the delay is maintained unchanged. Hence, very small spacing between the interconnects will necessitate much wider wires, which decreases wiring density.

Table 6.12: Multilevel interconnect geometries for worst peak crosstalk=10% of V_{dd}, for non-ideal return paths.

A_m=0.191cm^2, f_c=11.511GHz, n=8.117 levels						
Tier #	e_w	L_n (gate pitches)	Width (microns)	pitch (microns)	S/W	$n_{repeaters}$
Tier 5	0.33	5926.21	0.549	0.960	0.75	8600
Tier 4	0.33	2815.68	0.327	0.654	1.0	29800
Tier 3	0.4	1614.65	0.158	0.315	1.0	168600
Tier 2	0.4	668.9	0.073	0.145	1.0	585600
Tier 1	0.4	106.58	0.048	0.096	1.0	0

Summarizing, in this section non-ideal return paths have been included in the design of the *n-tier* multilevel interconnect architecture. The increased effect of inductance due to inclusion of non-ideal return paths is countered by inserting additional co-planar shielding lines. This reduces the area available for signal interconnects. However, the reduced spacing between the interconnects compensates for the reduced wiring density to some extent. The inclusion of non-ideal return paths in the multilevel interconnect architecture results in a slight increase in the number of metal layers. For the ideal return path case, the number of levels could be approximated to 8. By including non-ideal return paths, the required number of levels is rounded up to 10. However, only a small fraction of the area on the uppermost tier is used for the intra-macrocell wiring. The remaining area can be used for inter-macrocell global wiring.

The relative widths of the two signal lines and the surrounding two power/ground lines can be optimized further to minimize delay while meeting the crosstalk constraint, as has been described in detail in [63]. Such a refinement would help to further optimize the interconnect design of the global interconnects.

6.6 Optimum aspect ratio using *rlc* models

The primary target in designing the multilevel interconnect stack is to achieve higher wiring density for larger clock frequencies. Higher clock frequencies require the use of *rlc* models to account for the inductive effects which appear in high speed circuits. VLSI designers are increasingly using larger aspect ratio interconnects where the interconnect height is greater than the width [43], [65]. This reduces the line resistance and permits the utilization of narrower wires, which increases wiring density.

However, a larger wire thickness would imply greater capacitive coupling between adjacent interconnects, which would aggravate the crosstalk problem. The magnitude of the crosstalk can be decreased by using a thinner inter-level dielectric, which would increase the ground capacitance to the orthogonal metal layers above and below the plane of interest. It can also be diminished by increasing the spacing between the wires; however, both these solutions will lower the wiring density. Therefore, it is important to determine an optimum aspect ratio for the interconnects which

1. maximizes wiring density, thereby reducing the overall number of metal layers,
2. satisfies target crosstalk constraints for all the metal layers,
3. and is technologically feasible.

The optimum aspect ratio problem becomes compounded when repeaters are used in the design. The repeater insertion methodology in Chapter 3 used a logic gate area calculator which was based on the assumption that every logic gate drives an average length interconnect, and hence the logic gate size is determined by the line capacitance of the interconnect. When higher aspect ratio (i.e. thicker) interconnects are used, the line capacitance increases, which forces the sizes of the logic gates to increase. This in turn would use up more of the free area on the silicon for logic gates, leaving space for inserting fewer repeaters. Therefore, using higher aspect ratio interconnects is like a double-edged sword : while it can increase the wiring density by decreasing interconnect width, it can also prevent further reduction in the wire widths by limiting the number of repeaters that can be inserted.

6.6.1 Aspect ratio design space

The purpose of this discussion is to investigate the design space of all possible aspect ratios using repeaters, and then determine the interconnect geometry which is most optimal. Therefore in order to maintain consistency in the comparison between different designs, it will be assumed that the logic gates are always sized assuming a unity aspect ratio for the interconnects. This would always provide the same area for repeater insertion, irrespective of the interconnect aspect ratio. This is a valid assumption to make because the chip designers would always try to minimize the capacitive loads for logic gates. Of course, the size of repeaters will still be determined by the aspect ratio of the interconnect in which they are placed.

The design space for an optimal aspect ratio is now investigated using the case study from Section 6.4.3. This macrocell has $8.78M$ logic gates, 8 metal layers, and operates at f_c=11.511GHz. It is assumed that the wire height to wire width ratio (T/W) and the dielectric height to wire width ratio (H/W) are the same for all the metal layers (this assumption will be relaxed later, in Section 6.6.3). The wire spacing to wire width ratio (S/W) for each metal layer is adjusted so that the peak crosstalk of all the interconnects of that tier is no larger than the maximum specified crosstalk.

The aspect ratio design plane is plotted in Figure 6.4. It shows the (T/W) ratio on the y-axis and the (H/W) ratio on the x-axis. The envelopes are contour plots for designs with a maximum of 8 metal levels and peak crosstalk no greater than 10% and 15% of V_{dd}, respectively. All the points enclosed by the envelope belong to the "feasible" region i.e. they satisfy both the number of metal levels and crosstalk constraints.

The feasible region for peak crosstalk equal to $0.1V_{dd}$ is a subset of that for peak crosstalk equal to $0.15V_{dd}$. This demonstrates that a more stringent restriction on

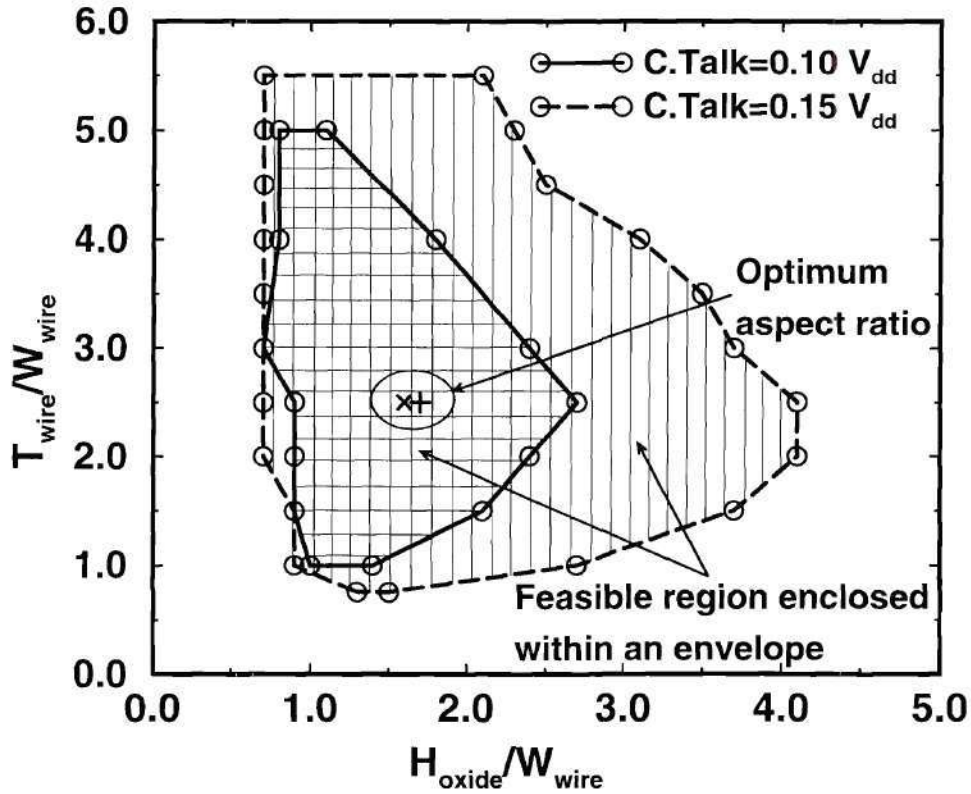


Figure 6.4: Aspect ratio design space for a 8.78M logic gate macrocell with maximum 8 metal levels.

the crosstalk decreases the design space available for designers. Moreover, a detailed analysis of the design points in the two feasible regions reveals that the number of metal layers is slightly greater when the peak crosstalk is more restrictive. This is the penalty that has to be paid for a achieving a design with a lower crosstalk.

The boundaries of the feasible region can be interpreted to be the constraints placed by the number of metal levels. On the left boundary, for a specific T/W ratio, decreasing the H/W ratio will increase the ground capacitance. This would decrease the crosstalk but the rc product of the wires will increase, and therefore the overall number of metal layers will increase beyond the specified maximum of 8. On

the right boundary, for a specific T/W ratio, increasing the H/W ratio will decrease the ground capacitance. This would increase the crosstalk, and therefore the spacing between the wires needs to be increased to keep the crosstalk within bounds. Due to increased spacing, the wiring density will decrease, leading to an increase in the number of metal layers. Therefore, although all the points within the feasible region have the same crosstalk, they have different number of metal layers; and the number of metal layers is minimum at some point in the interior of the feasible region.

The design points which have the lowest number of metal layers are marked with a "×" and a "+" in Figure 6.4 for peak crosstalk equal to 10% and 15% of V_{dd}, respectively. These two points represent the optimum aspect ratio that minimize the number of metal layers (by maximizing wiring density) while satisfying the crosstalk constraints. The T/W ratio is equal to 2.5 for both the optimum design points whereas the H/W ratio varies between 1.6 and 1.7. The number of metal layers is only fractionally greater for $T/W = 2.0$. The complete design specifications of the multilevel architecture for peak crosstalk equal to $0.1V_{dd}$ and $0.15V_{dd}$ are shown in Tables 6.13 and 6.14, respectively.

Table 6.13: Multilevel interconnect design specifications for peak crosstalk=10% of V_{dd}.

A_m=0.191cm^2, f_c=11.511GHz, n = 6.79 levels, n_{rep} = 734716, Peak CTalk = $0.1V_{dd}$, $Power_{total}$ = 43.04W							
Tier	Longest length (gate pitches)	Width (microns)	Pitch (microns)	$\frac{S}{W}$	$\frac{T}{W}$	$\frac{H}{W}$	Number of repeaters
4	5926.2	0.3053	1.157	2.79	2.5	1.6	36121
3	1847.08	0.1096	0.3496	2.19	2.5	1.6	132890
2	800.15	0.0532	0.1452	1.73	2.5	1.6	565705
1	130.42	0.0450	0.09	1.0	2.5	1.6	0

As explained earlier, although the wires are narrower for the design in Table 6.13, the wire pitch is greater due to a larger spacing compared to the design in Table 6.14, and this is the reason for the marginally greater number of metal layers in the former.

Table 6.14: Multilevel interconnect design specifications for peak crosstalk=15% of V_{dd}.

A_m=0.191cm^2, f_c=11.511GHz, n = 6.60 levels, n_{rep} = 674277, Peak CTalk = 0.15V_{dd}, $Power_{total}$=42.96W							
Tier	Longest length (gate pitches)	Width (microns)	Pitch (microns)	$\frac{S}{W}$	$\frac{T}{W}$	$\frac{H}{W}$	Number of repeaters
4	5926.2	0.3123	1.0119	2.24	2.5	1.7	30566
3	1975.98	0.1209	0.3385	1.8	2.5	1.7	128314
2	817.53	0.0569	0.1434	1.52	2.5	1.7	515397
1	130.42	0.0450	0.09	1.0	2.5	1.7	0

6.6.2 "Low power" vs "high performance" aspect ratios

Currently the personal computer industry focuses on two separate, but inter-related, market segments : 1) the high performance computing sector which includes desktop computers, workstations and servers, and 2) the low power sector which includes mobile, low power devices such as laptop computers, and hand held devices such as Personal Digital Assistants (PDAs) and cellular phones. Interconnect design can be optimized to suit either of the two design requirements, as will be shown in this section.

Low power devices usually operate on a lower clock frequency, compared to their high performance counterparts. Decreasing the clock frequency, by itself, has a significant effect on reducing the overall power dissipation. Moreover, a smaller clock frequency also permits the utilization of narrower wires, which reduces the overall number of metal layers too. Consider the design case study described in Table 6.13. If the clock frequency is decreased by 2GHz, the total power dissipation decreases by 48%. The number of metal layers, however, decreases marginally. The multilevel design characteristics for this "tall" design (where the wire thickness is greater than its width) are shown in Table 6.15.

However, a "flat" wiring scheme, wherein the interconnect thickness is less than

Table 6.15: Multilevel interconnect design specifications for peak crosstalk=10% of V_{dd}, f_c=9.511GHz : "Tall" design.

A_m=0.191cm^2, f_c=9.511GHz, n = 6.30 levels, n_{rep} = 938278, Peak CTalk = $0.1V_{dd}$, $Power_{total}$ = 22.36W							
Tier	Longest length (gate pitches)	Width (microns)	Pitch (microns)	$\frac{S_{wire}}{W_{wire}}$	$\frac{T_{wire}}{W_{wire}}$	$\frac{H_{oxide}}{W_{wire}}$	Number of repeaters
4	5926.2	0.2817	1.0423	2.7	2.5	1.6	10150
3	2348.84	0.1142	0.3769	2.3	2.5	1.6	117315
2	982.56	0.0522	0.1305	1.5	2.5	1.6	810814
1	130.42	0.0450	0.09	1.0	2.5	1.6	0

its width, and the oxide height is greater than the wire width, has the potential of reducing the power dissipation even further. This can be understood by a simple calculation. The power dissipation in logic gates, interconnects and repeaters is proporational to the capacitance per unit lenght of the line. If the thickness of the metal and the height of the oxide are scaled by a factor Ω, such that

$$T = \frac{W}{\Omega} \quad \text{and} \quad H = \Omega W, \tag{6.22}$$

then the line capacitance c also scales by the same factor Ω :

$$c_{new} = \frac{c_{old}}{\Omega}. \tag{6.23}$$

The rc delay and peak crosstalk are, however, proportional to the product TH. Hence, they will remain approximately constant. This analysis shows that if the wire thickness is decreased, and the dielectric height is increased, by a factor $\Omega \geq 1$, then the power dissipation will decrease by approximately the same factor [10], [66].

Therefore, when a "flat" wiring scheme is applied to the "tall" design in Table 6.15, the power dissipation is further reduced by 21%, compared to the "tall" design. The number of metal layers increases marginally, but it is now equal to the number of metal layers of the high performance design. The multilevel design characteristics of this "flat" design are shown in Table 6.16.

Table 6.16: Multilevel interconnect design specifications for peak crosstalk=10% of V_{dd}, f_c=9.511GHz : "Flat" design.

A_m=0.191cm^2, f_c=9.511GHz, n = 6.80 levels, n_{rep} = 1112000, Peak CTalk = 0.1V_{dd}, $Power_{total}$ = 17.77W							
Tier	Longest length (gate pitches)	Width (microns)	Pitch (microns)	$\frac{S_{wire}}{W_{wire}}$	$\frac{T_{wire}}{W_{wire}}$	$\frac{H_{oxide}}{W_{wire}}$	Number of repeaters
4	5926.2	0.4197	1.3850	2.3	0.75	2.1	20309
3	1841.3	0.1360	0.3672	1.7	0.75	2.1	153682
2	827.1	0.0669	0.1338	1.0	0.75	2.1	938124
1	114.3	0.0469	0.0938	1.0	0.75	2.1	0

6.6.3 Optimum aspect ratio

Finally, the *n-tier* design software is used to quickly determine the feasible region design space for every tier of the multi-level interconnect architecture. The motivation is to optimize the interconnect width, thickness and spacing and oxide height for each tier, so that wiring density is maximized and crosstalk constraints are satisfied. The range of values for T/W and H/W is narrowed using technological constraints. Current microprocessor technology and the ITRS roadmap indicate that the maximum values for T/W and H/W is between 2-2.5; higher values for these two ratios decreases the yield. Starting with the first tier, all possible combinations of the aspect ratios are evaluated, and the set of values which results in the highest wiring density for that tier is chosen. Sometimes, multiple such combinations may exist, in which case the designers are free to choose any set of values depending on overall design specifications and technology criteria.

The results of optimizing the aspect ratios for every tier of a 7M gate, 1.6GHz, 2.0cm^2 macrocell case study from the 130nm technology node are compared to a 6 metal layer, 130nm commercial process technology [65] in Table 6.17. The process specifications in [65] are a revised version of the 130nm process technology in [43]. There is a striking similarity between the aspect ratios for the model predicted optimal

design and the data from the commercial process technology.

For the 45nm case study discussed earlier, one possible set of values for the aspect ratios of each tier that minimizes total number of metal layers, and satisfies a maximum crosstalk constraint of $0.1V_{dd}$ is shown in Table 6.18. The interconnect aspect ratios of the 130nm process technology in [65] are shown alongside, for comparison. Once again, there is a lot of similarity between the aspect ratios for the 6 metal layer 130nm generation commercial technology and the 7 metal layer *n-tier* 45nm generation case study.

Table 6.17: Optimum relative interconnect cross-sectional dimensions for each tier. The 130nm *n-tier* case study with f_c=1.6GHz and peak crosstalk of $0.1V_{dd}$, is compared to Intel's 130nm technology specifications. The pitch is expressed in units of minimum feature size F.

	***n-tier* 130nm technology generation case study**				**Intel's 130nm technology presented at IEDM 2001**			
Metal layer	Pitch	S/W	T/W	H/W	Pitch	S/W*	T/W	H/W
6	12.35F	1.20	1.96	0.6	8.8F	1.0	2.1	N/A
5	12.35F	1.20	1.96	0.6	8.18F	1.01	1.7	N/A
4	5.25F	1.25	1.76	1.0	5.5F	1.01	1.6	N/A
3	5.25F	1.25	1.76	1.0	3.27F	1.0	1.7	N/A
2	2F	1.0	1.38	1.6	3.27F	1.0	1.7	N/A
1	2F	1.0	1.38	1.6	2.25F	0.78	1.7	N/A

* Calculated from published data

Using the values for the wiring cross-sectional aspect ratios shown in Table 6.18 for the 45nm generation case study results in the *n-tier* design shown in Table 6.19. The number of metal levels is 6.55, which is lesser than that of the design in Table 6.13. This is due to the fact that a larger number of interconnects is accomodated on each tier compared to the previous design. The length of the longest interconnect on each tier in Table 6.19 is greater than the corresponding length in Table 6.13, demonstrating that the wiring density on each tier has indeed increased due to the use of optimal geometry scaling for each tier.

Table 6.18: Optimum relative interconnect cross-sectional dimensions for each tier. The 45*nm* *n-tier* case study with f_c=11.511GHz and peak crosstalk of $0.1V_{dd}$, is compared to Intel's 130nm technology specifications. The pitch is expressed in units of minimum feature size F.

	n-tier **45nm technology generation case study**				**Intel's 130nm technology presented at IEDM 2001**			
Metal layer	Pitch	S/W	T/W	H/W	Pitch	S/W*	T/W	H/W
7	23.4F	1.80	2.04	1.0	-	-	-	-
6	8F	1.40	1.88	1.0	8.8F	1.0	2.1	N/A
5	8F	1.40	1.88	1.0	8.18F	1.01	1.7	N/A
4	3.1F	1.05	1.78	1.2	5.5F	1.01	1.6	N/A
3	3.1F	1.05	1.78	1.2	3.27F	1.0	1.7	N/A
2	2F	1.0	1.32	1.5	3.27F	1.0	1.7	N/A
1	2F	1.0	1.32	1.5	2.25F	0.78	1.7	N/A

* Calculated from published data

Another interesting observation is the difference in metal thickness and oxide height ratios for the local tiers and the global tiers. The performance (clock frequency) is determined by the longest global interconnects. Therefore, using the discussion in Section 6.6.2, the optimum aspect ratio of the global tiers can be judged as obeying a "high performance" design pattern - relatively, large T/W and small H/W ratios. On the other hand, there is a large number of short interconnects, which contribute significantly to the total interconnect power dissipation. Therefore, the optimum aspect ratio of the local tiers follows a "low power" design pattern - relatively, smaller T/W and larger H/W ratios. The aspect ratios of the intermediate tiers lie between thense two extremes, striking a balance between high performance and low power designs.

6.7 Summary

The effects of inductance on multilevel interconnect architecture has been investigated in this chapter using the *n-tier* models and methodologies from Chapters 2

Table 6.19: Multilevel interconnect design specifications using optimum aspect ratios for each tier.

A_m=0.191cm^2, f_c=11.511GHz, n = 6.55 levels, Peak CTalk = $0.1V_{dd}$.						
Tier	Longest length (gate pitches)	Width (microns)	Pitch (microns)	$\frac{S_{wire}}{W_{wire}}$	$\frac{T_{wire}}{W_{wire}}$	$\frac{H_{oxide}}{W_{wire}}$
4	5926.2	0.3757	1.052	1.80	2.04	1.0
3	1994.1	0.150	0.3598	1.40	1.88	1.0
2	860.86	0.068	0.1394	1.05	1.78	1.2
1	130.42	0.0450	0.09	1.0	1.32	1.5

and 3 together with the inductance models in Chapters 4 and 5. In order to decrease inductive crosstalk, the spacing between signal lines needs to be increased, which decreases wiring density. Inductance effects are most severe on the global tiers, where co-planar return paths need to be provided to act as shielding lines to decrease crosstalk. An optimum aspect ratio for the multilevel interconnect system has also been established, which justifies the aspect ratios for the different metal layers used in commercial products. In Chapter 7, the *n-tier* design methodology software tool will be described in detail. It will be used for different system level interconnect prediction applications, and compared to data from commericial products.

CHAPTER 7

MINDS - Multilevel Interconnect Network Design Simulator

7.1 Introduction

The models, methodologies and algorithms described in Chapters 2-6 have been incorporated into a software tool written in the C programming language, which is called the *Multilevel Interconnect Network Design Simulator*, or *MINDS*. The key assumptions that are necessary for this system level interconnect design tool are enumerated in Section 7.2. This section also provides a listing of the various parameters used, and the values assumed for those parameters for the simulations of Chapters 2 and 3. The additional functionalites of *MINDS* which relax some of these assumptions are listed separately in Section 7.3. Issues of sensitivity of the simulator results to system parameters are addressed in Section 7.4. A unique application of *MINDS* in rapidly estimating the Rent's constants of a system is outlined in Section 7.5. Finally, the results of *MINDS* are validated using data from a commercial family of microprocessors in Section 7.6

7.2 Assumptions

The list of assumptions for the *n-tier* methodology were enumerated in Section 2.2. In addition, here is a comprehensive list of assumptions made in *MINDS*.

1. The macrocell consists of a homogenous array of identical gates. For the examples discussed in this disseration, all the gates are assumed to be 2-input NAND gates, with a fanout of 3.

2. Values of Rent's coefficient k and Rent's exponent p are estimated using historical data. A value of $k = 4$ and $p = 0.6$ has been used in this work, unless otherwise stated.

3. Wiring efficiency (e_w) is estimated based on via blockage, power/ground/clock blockage and router efficiency. Wiring efficiency is 40% for all the metal layers in the system, unless otherwise stated.

4. The repeater efficiency (e_{rep}), which is the fraction of free susbstrate area used for repeaters, is 60%.

5. The ratio of maximum permissible time delay to clock period, is β_{local} for the local tier, and β_{global} for all other tiers. For all the designs discussed here, $\beta_{local} = 0.25$ and $\beta_{global} = 0.9$.

6. The gates are sized so that the time delay of a critical path equals the clock period [50].

7. The aspect ratio of the interconnects on all the metal layers is equal to unity (i.e. $W = S = T = H$), unless otherwise stated.

7.3 Capabilities

In order to account for "real-life" design characteristics, some additional capabilities have been incorporated into *MINDS* so as to make the design predictions more realistic and accurate.

7.3.1 Variable pitch usage model

To strike a balance between the conflicting demands posed by a high wiring density requirement (necessitating the use of thin wires) and designing critical paths with minimum time delay (requiring low-resistance "fatter" wires), designers tend to use multiple width wires on the same metal level [67]. A majority of the wires have the thinner width and only the critical path wires have the thicker width. The pitch usage factor associated with each pitch is defined as the fraction of the metal layer area occupied by wires with that particular pitch. Let p_1 and p_2 be the two different pitches used on a metal layer, with the pitch usage factor being α_{p1} and α_{p2}, respectively. Rewriting (2.2),

$$\text{Total wire length } = \chi\sqrt{\frac{A_m}{N}}\int_{L_{n-1}}^{L_n} li(l)dl = \frac{e_{w,n}n_tA_m}{p_n} \tag{7.1}$$

Assume there are q different pitches p_1, p_2, ... p_q with pitch usage p_1, p_2, ... p_q, such that

$$\sum_{i=1}^{q}\alpha_{pi} = 1 \tag{7.2}$$

Then (7.1) can be rewritten as

$$\text{Total wire length } = \frac{\alpha_{p1}e_{w,n}n_tA_m}{p_1} + \frac{\alpha_{p2}e_{w,n}n_tA_m}{p_2} + ... + \frac{\alpha_{pq}e_{w,n}n_tA_m}{p_q} \tag{7.3}$$

Equating (7.2) and (7.3) one gets,

$$\frac{1}{p_n} = \frac{\alpha_{p1}}{p_1} + \frac{\alpha_{p2}}{p_2} + ... + \frac{\alpha_{pq}}{p_q} \tag{7.4}$$

For $q = 2$ [67],

$$p_n = \frac{p_1 p_2}{\alpha_{p1} p_2 + \alpha_{p2} p_1} \tag{7.5}$$

7.3.2 Variable wiring efficiency

Due to the disproportionate usage of power and clock lines on each of the metal levels, the wiring efficiency for signal wires could vary from one level to the next. *MINDS* accepts wiring efficiency as an input from the user for each metal layer and uses it to compute the length of the longest interconnect on each layer. The overall wiring efficiency of each layer is defined as

$$e_w = e_{rout} \left(1 - b_{pow/clk}\right) (1 - b_{via}) \tag{7.6}$$

where e_{rout} is the efficiency of the routing tool, $b_{pow/clk}$ is the blockage due to power/ground or clock lines and b_{via} is the via blockage. The blockage due to power/ground or clock lines is also a user-defined input. The via blockage is estimated based on the models described in [5]. Knowing e_w, $b_{pow/clk}$ and b_{via}, the router efficiency e_{rout} can be easily calculated from (7.6).

7.3.3 Layer assignment

MINDS calculates the longest interconnect length on each metal layer using the stochastic wiring distribution in [19]. Thus it can assign layers to certain benchmark interconnect lengths which are used by the designers in extraction tools to estimate the circuit performance. Knowing the layers on which these interconnect lengths will be located will help to design wire pitches appropriately. For example, a design requirement might be that the 90th percentile length (i.e. the interconnect length which is longer than 90% of all the interconnects in the system) should have a maximum

time delay of 90% of the clock period. By assigning the 90th percentile interconnect to a particular metal layer, *MINDS* helps the designers to determine the wire pitch of that layer. This capability gives *MINDS* the ability to efficiently design the multilevel interconnect network to minimize the wire-limited chip size. The layer assignment algorithm in *MINDS* is modular and can be modified to suit user specifications.

7.4 Sensitivity of *MINDS* to system parameters

The sensitivity of *MINDS* to system parameters is analyzed in this section. The main system parameters considered here are the Rent's constants p and k and the wiring efficiency factor e_w.

7.4.1 Rent's constants

Two key inputs required by *MINDS* are the Rent's exponent p and Rent's coefficient k. It is important to estimate these values accurately because the results of *MINDS* are very sensitive to p and k as seen from Figures 7.1 and 7.2. The die sizes on the y-axis are the estimates of *MINDS* for a hypothetical microprocessor with some pre-assigned values of wiring pitches for each metal level. Figure 7.1 shows the sensitivity of *MINDS* to the value of Rent's exponent p by plotting the die sizes for a constant value of $k = 4$; the sensitivity of *MINDS* to k is shown by Figure 7.2 which plots the die sizes for a constant value of $p = 0.55$. The effect of fluctuations in the values of p and k by 1%, 3% and 5% on the estimated die size is tabulated in Tables 7.1 and 7.2. This analysis suggests that accurate estimation of Rent's parameters within approximately $\pm 5\%$ is essential to accurately design the multilevel interconnect architecture and estimate optimal die size using *MINDS*.

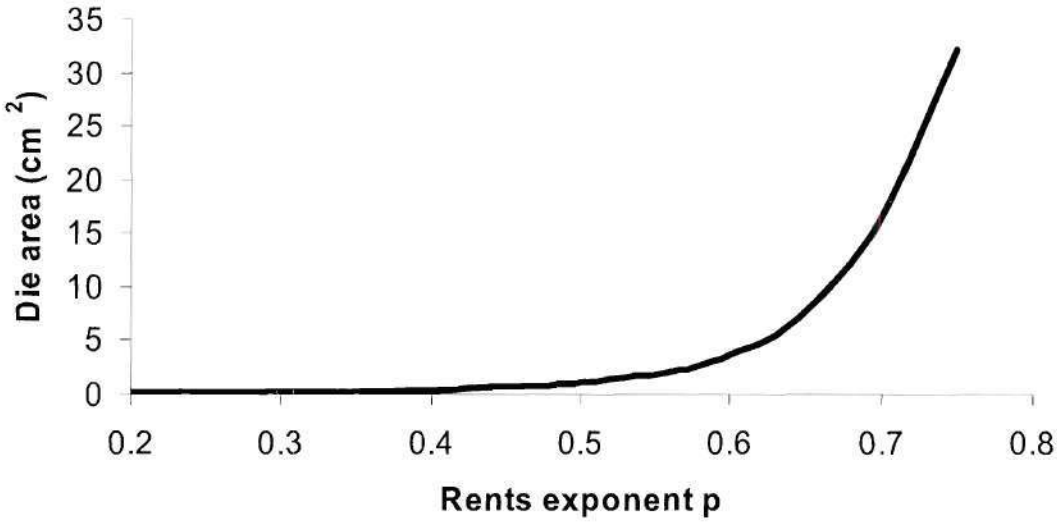


Figure 7.1: Sensitivity of *MINDS* to Rent's exponent p.

Table 7.1: Effect of variations in p on die size estimates of *MINDS* for a hypothetical microprocessor.

p value ($k = 4$)	Die Area (cm^2)	% change in area with ±1% change in p	% change in area with ±3% change in p	% change in area with ±5% change in p
$p = 0.55$	1.91	+26% -7%	+23% -18%	+41% -28%
$p = 0.65$	7.34	+11% -9%	+33% -19%	+62% -33%
$p = 0.75$	32.3	+11% -1%	+56% -27%	+68% -42%
$p = 0.85$	139	+13% -6%	+31% -15%	+48% -49%

7.4.2 Wiring efficiency factor

The wiring efficiency factor, e_w in (7.6), determines the area available for wiring and thus directly influences the optimal chip size. To evaluate the sensitivity of *MINDS* to the wiring efficiency factor, it is assumed that e_w is constant for all the metal layers. For the same hypothetical microprocessor with pre-assigned wiring pitches as before, the die area predicted by *MINDS* is plotted against different values of the wiring efficiency factor in Figure 7.3. This analysis highlights the importance of estimating the wiring efficiency factor accurately. Within a microprocessor, it varies from one

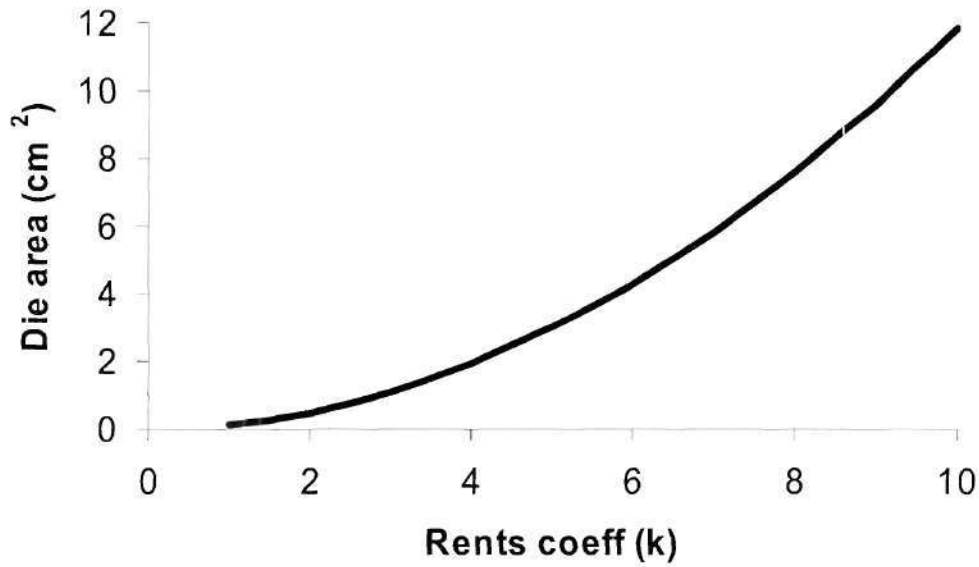


Figure 7.2: Sensitivity of *MINDS* to Rent's coefficient k.

Table 7.2: Effect of variations in k on die size estimates of *MINDS* for a hypothetical microprocessor.

k value ($p = 0.55$)	Die Area (cm^2)	% change in area with $\pm 1\%$ change in k	% change in area with $\pm 3\%$ change in k	% change in area with $\pm 5\%$ change in k
$k = 3$	1.07	+3% -2%	+7% -6%	+10% -9%
$k = 4$	1.91	+2% -2%	+6% -6%	+10% -10%
$k = 5$	2.98	+2% -2%	+6% -6%	+10% -10%
$k = 10$	11.89	+2% -2%	+6% -6%	+10% -10%

metal layer to the next due to the differences in the area occupied by power and clock lines, bonding pads and vias on different layers. Fortunately, analysis of data from processors belonging to different families shows little variation in this factor; hence it can be assumed to be fairly constant for the same metal layer across generations.

7.5 Rapid extraction of Rent's coefficients

The experiments discussed in the previous section use Rent's parameters that are extracted from earlier generations of the same family of commercial microprocessors

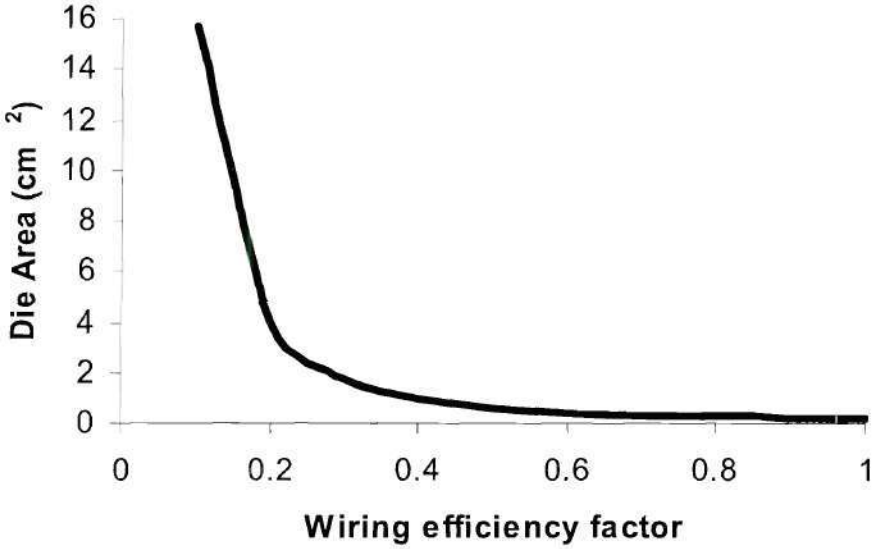


Figure 7.3: Sensitivity of *MINDS* to wiring efficiency factor e_w.

using *MINDS* to obtain the best fit to the data area. The values of Rent's constants obtained depend on the specified bounds for p and k (in this case, the specified bounds were $0.5 \leq p \leq 0.7$ and $4 \leq k \leq 7$). One advantage of this unique Rent's constants extraction method is the ease with which these parameters were obtained. These "rapidly extracted" parameters are useful as long as radical changes in system design are not implemented from one generation to the next. These Rent's parameters are useful for estimating the area of the next generation microprocessors.

Many papers have discussed the extraction methods of Rent's rule. Landman and Russo [68] investigated the earliest extraction methods in which they used special circuit partitioning algorithms to ascertain and extract the parameters associated with Rent's rule. Hagen *et al.* looked at similar methods for describing the Rent's exponent for individual netlists before and after they had been placed into a real VLSI system [69]. Zarkesh-Ha developed a program to compute the Rent's parameters for ASICs by partitioning the design and extracting the number of pins versus number of

gates in the partitioned block [37]. The relationship between the Rent's parameters obtained using the rapid extraction method and those estimated using traditional partitioning approaches is yet to be determined. Theoretically, they should be approximately equal; however, practically they may be different if the rapidly extracted Rent's parameters have a strong dependence on other system level parameters. If the architectures change extensively in a new design, then a more traditional extraction of Rent's parameters from the netlist may be needed.

7.6 Validation of area estimates with *MINDS*

The validity of the model predictions is verified by comparing with data from four generations of a leading microprocessor family.

7.6.1 Megacell area estimation

MINDS was used to estimate the area of logic functional blocks or megacells of a microprocessor called Processor D. The hypothesis of this work is that a single Rent's exponent and Rent's coefficient would be sufficient to obtain approximations for a variety of logic blocks. For this experiment, the Rent's constants were estimated for 16 randomly chosen logic blocks by fitting the stochastic wiring distribution to the interconnect length distribution data for each block. The arithmetic average of the p values and the geometric average of the k values were assumed to be the average values of the Rent's constants for all the logic blocks.

The number of logic gates for each megacell is approximated by assuming that the average logic gate can be described as a 3-input 6-transistor static NAND gate. Hence, number of gates is equal to number of schematic transistors (as opposed to layout transistors) divided by 6. The wiring pitches and variable pitch usage for each

of the 6 metal levels were extracted from the data and input by the user. The wiring efficiency and power/clock blockage factors were also calculated from the data and averaged out for all the logic blocks for each layer, and these were given as inputs to *MINDS*. In addition, intra-megacell communication was assumed to be on average within the first four metal levels.

The megacell areas predicted by *MINDS* are compared to the actual area for each of the 16 logic blocks in Figure 7.4. The megacell areas have been normalized to the size of the largest megacell. There is reasonable agreement between the *MINDS*'s predicted megacell area and actual data, with an average error of about 15%.

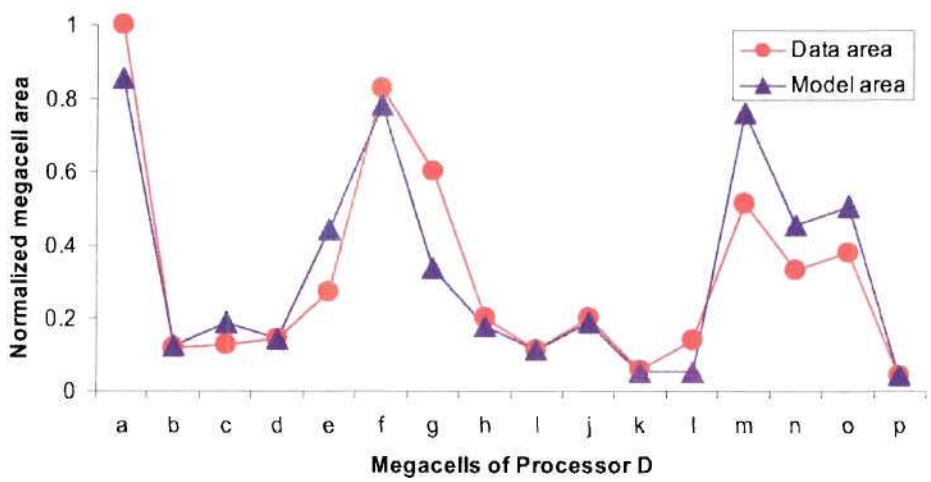


Figure 7.4: Normalized *MINDS* output area vs. actual data area for 16 different logic blocks belonging to Processor D.

Megacells $m - p$ belong to a class of logic blocks which are almost entirely built of library cells. The high level of optimization in library cells implies that the actual data area would be smaller than the model prediction. Megacells $i - l$ have a very high degree of manual design component in them. Therefore, the lower level of optimization in their design means that actual data area would be greater than the model predicted

area. Megacells $a - h$ have a little bit of both - manual design component and library cells. Hence the error in model prediction and actual data would be double sided (i.e. both plus and minus) for these megacells, as is illustrated in Figure 7.4.

7.6.2 Total chip size estimation

In addition to *MINDS* being used to predict the size of small logic blocks within a design, it can also be used to estimate the cumulative logic area on a chip. With this ability, *MINDS* can be used to predict total die size of future microprocessors when the user has estimated key system parameters e.g. Rent's constants. An experiment to validate the predictive capability of *MINDS* is demonstrated in this section.

The data available to us for Processors A, B and C, belonging to a family of commercial microprocessors, comprised of number of logic transistors, total number of transistors, logic die area, number of metal levels and pitch of each metal layer. First, the values of Rent's constants were chosen so that the actual die area of Processor A is exactly equal to the area predicted by *MINDS*. These values of Rent's constants were then used to predict the die areas of Processors B and C. Each of these three microprocessors belongs to a different technology generation, with 4, 5 and 6 metal layers, respectively. The number of transistors increases progressively in these three microprocessors. The only similarity is that they all have the same micro-architecture which permits us to assume that the Rent's constants are the same for all the three generations.

By using $p = 0.51$ and $k = 4$, *MINDS* predicts the area of Processor A to be 141mm^2, which is approximately equal to actual logic die area of 143mm^2. Using these values of Rent's constants and appropriate pitches for the metal layers, *MINDS* was used to predict the die areas for later generation processors B and C. The model

predicted areas and actual data area are tabulated in Table 7.3. It can be observed that there is less than 1% and 10% difference for Processors B and C respectively, between the model predicted area and actual logic die area.

Table 7.3: *MINDS* predicted area and actual data area (logic) for Processors B and C.

	MINDS predicted Area (mm^2)	Data Area (logic) (mm^2)
Processor B	126	127
Processor C	90	102

This experiment was extended further by using data from other microprocessors described in the literature - specifically, Processors E [70], F [71], G [72] and H [73]. The number of logic transistors in each of these processors was estimated by assuming that the ratio of logic transistors to total number of transistors is the same as in Processors $A - C$. Using the same values of $p = 0.51$ and $k = 4$, *MINDS* was used to predict the logic die areas for Processors $E - H$. From the die micrographs shown in Figure 1 of [71] and Figure 7 of [74], the actual logic die area is estimated to be about 70% of the total die area. It is assumed that older processors with smaller caches have 80% logic area and newer processors with larger caches have only 60% of the die area available for logic circuitry. These assumptions were used to estimate the logic die areas for processors $E - H$ from the actual die area given in [70]-[73]. From Table 7.4 the error in the *MINDS* predictions is less than 4% in each case.

Table 7.4: *MINDS* predicted area and actual data area (logic) for Processors $E - H$.

	MINDS predicted Area (mm^2)	Data Area (logic) (mm^2)
Processor E	263	255
Processor F	116	117
Processor G	144	140
Processor H	82	79

7.7 Summary

Using the stochastic wiring distribution and the *n-tier* design methodology, a new software tool called *MINDS* has been created to help design the multilevel interconnect network and predict the optimal die sizes. *MINDS* has the capability to account for variable pitch usage and metal-layer-dependent wiring efficiency factors. Comparison of model predicted areas with actual logic block data for a commercial microprocessor shows an average error of 15%. Similarly, *MINDS* is used to extract high-level system parameters from earlier generations of the same microprocessor family, and is then used to predict the chip sizes for later generations. The error in the model predictions compared to data is less than 10%. Possible applications of *MINDS* are to enable advanced floor planning methodologies and to aid in the design of the multilevel interconnect networks for optimal GSI designs.

CHAPTER 8

Conclusions and Future Work

This chapter identifies potential avenues of research in the area of system level interconnect modeling. Some of the possible focus areas would be integrating global interconnect modeling with the *n-tier* multilevel design methodology, accurate modeling of the wiring resources consumed by power, ground and clock lines, applying the *n-tier* design to diagonal wiring and 3-D architectures and enhancing the distributed *rlc* interconnect models by including real drivers. Finally, the conclusions and summary of this dissertation are presented.

8.1 Integrated system level inter- and intra- macrocell interconnect design

This dissertation has proposed an optimal reverse scaling methodology for intra-macrocell interconnect design that minimizes area, cycle time, number of metal levels or power dissipation. It is based on a stochastic wiring distribution for a homogenous array of random logic gates. Other research work has focussed on modeling global interconnects that connect different macrocells, using a heterogenous Rent's rule [36], [37].

The heterogeneous Rent's rule [75] states that if there n different macrocells on

a chip, each with N_i gates and with Rent's constants of p_i and k_i, then the overall system also obeys Rent's rule thus :

$$T_{eq} = k_{eq} N_{eq}^{p_{eq}} \tag{8.1}$$

where

$$k_{eq} = \sqrt[N_{eq}]{\prod_{i=1}^{n} k_i^{N_i}}, \tag{8.2}$$

$$p_{eq} = \frac{\sum_{i=1}^{n} p_i N_i}{N_{eq}} \tag{8.3}$$

and

$$N_{eq} = \sum_{i=1}^{n} N_i. \tag{8.4}$$

If it is assumed that all the macrocells have the same Rent's constants, i.e.

$$p_1 = p_2 = p_3 ... = p_n = p \tag{8.5}$$

and

$$k_1 = k_2 = k_3 ... = k_n = k, \tag{8.6}$$

then the heterogenous Rent's rule reduces to the homogenous Rent's rule :

$$p_{eq} = p, k_{eq} = k, N_{eq} = N, T_{eq} = T \tag{8.7}$$

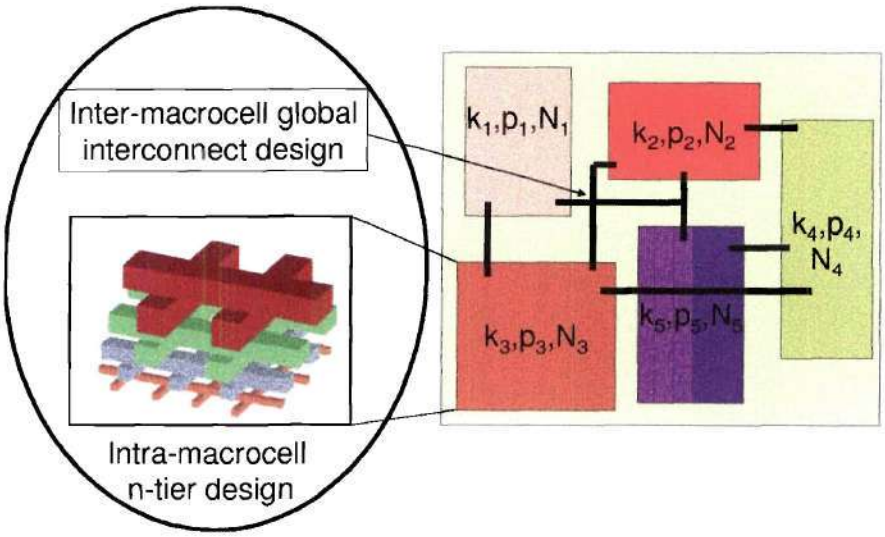
and

$$T = kN^p. \tag{8.8}$$

The routing, netlist and placement information is used to derive the stochastic global net-length distribution for signal interconnects, which is combined with clock and power line models to obtain an integrated global interconnect architecture in [37].

As shown in Figure 8.1, combining the *n-tier* design methodology for intra-macrocell signal interconnects with the models for global inter-macrocell interconnects will result in an integrated interconnect model for a GSI System-on-a-chip. Applications

of such an integrated architecture include : 1) optimum partitioning of the chip into logic blocks of different sizes to minimize overall wiring requirement, while maximizing system performance 2) sharing of metal layers for intra and inter macrocell wiring, reducing total number of metal layers, and 3) using dedicated buffer blocks to augment system performance on other macrocells. Moreover, recent work on global interconnect width optimization demonstrates that an optimum width of global interconnects maximizes the bandwidth-reciprocal latency of these interconnects [60]. Incorporating this optimum wire width design into the global inter-macrocell wiring design will enable the system to achieve higher throughput without compromising on time delay.



Integrated inter and intra-macrocell system level interconnect design

Figure 8.1: Integrated system level interconnect design for inter and intra macrocell wiring.

8.2 Wiring efficiency models

The work in this dissertation assumed a pre-determined wiring efficiency for each metal layer. The sensitivity of the *n-tier* methodology to the wiring efficiency was demonstrated in Section 7.4.2. Accurate estimation of the wiring efficiency would enhance the predictive capability of *MINDS*. A model for power/ground and clock lines within the logic macrocell, which in turn would depend on the percentage of power/ground and clock lines routed in the global layers, would enable realistic computation of the blockage caused due to these non-signal interconnects on the lower layers. *MINDS* estimates the via blockage caused by intra-macrocell interconnects using models from [18]. The blockage caused by the vias reaching up to the global inter-macrocell wires should be included in the analysis, as it would restrict the area available for wiring on all the lower layers. Finally, the router efficiency can be estimated by comparing the results of a router optimized physical layout with manual layout, and also by using the same router on different designs. Incorporating these models into the *n-tier* design methodology will narrow the gap between the simulation results and actual designs.

8.3 3-D and X architecture

Three dimensional integration has been proposed as a futuristic alternative to alleviating the wiring demand problem of future GSI systems. Multiple strata constitute the 3-D chip, with each stratum comprising of an active layer of transistors and several layers of interconnects. A new stochastic wiring distribution has been derived for 3-D interconnects in [76]. The *n-tier* design methodology has been used to optimize the 3-D system performance by minimizing area, cycle time and number of metal layers/strata. One potential use of the additional layer of transistors is to use it for

repeaters. Since repeaters are usually required on the upper metal layers, fabricating repeaters on the stratum above the global tier will eliminate the via blockage due to the repeaters. The *n-tier*-3D architecture should therefore be enhanced by investigating the effects of repeaters on system performance.

Recently, use of diagonal interconnects, along with the conventional orthogonal (Manhattan - architecture) interconnects, has been shown to improve chip performance by 10%, reduce power dissipation by 20% and reduce die cost by 30% [44]. This is due to a wire length reduction of 20% and via reduction of 30%. The results reported in [44] dedicate two metal layers solely to diagonal wires. There are different ways to include diagonal interconnects such as alternating between diagonal and orthogonal metal layers, interspersing diagonal and orthogonal layers etc. Using diagonal interconnects will change the interconnect distribution since there will be a larger number of shorter interconnects and a smaller number of longer interconnects. Derivation of a new stochastic wiring distribution for diagonal interconnections has enabled development of a *n-tier* interconnect methodology that optimizes the interconnect design by optimally using both orthogonal and diagonal interconnects [77]. It has been used to design a multi-tier interconnect methodology assuming that every layer has both orthogonal and diagonal interconnects ("liquid" routing). This is different from the basic premise of the *n-tier architecture* which assumes a preferred direction for all the interconnects on any metal layer. Further enhancements of the new multi-tier architecture would include repeater insertion and inclusion of inductance models for the interconnects.

8.4 Distributed *rlc* interconnect modeling

The models for distributed *rlc* interconnects in this dissertation assumed that the driver transistor operates in the linear region and could therefore be modelled as a resistor. For circuits that switch at high frequency, it is important to consider the non-linear behavior of the transistor. The parasitic capacitance at the output nodes of the repeaters can also not be ignored for small geometry devices. Including real drivers in the interconnect modeling will enable accurate estimation of the effects of repeaters on signal propagation. It will also improve the suitability of these new analytical models as a substitute of numerical circuit simulators.

8.5 Conclusion of dissertation

The overall goal of this research was to optimally design the multilevel interconnect network in future GSI systems. To accomplish this research, three tasks were identified : 1) to formulate an optimum methodology for design of the reverse-scaled interconnect architecture, 2) to develop models for time delay, crosstalk and repeater insertion, when inductance is included in the interconnect models, and 3) to quantify the impact of inductance on the multilevel interconnect architecture.

There are seven new and significant contributions of this research. These contributions are as follows : 1) a new *n-tier* multilevel interconnect design methodology has been formulated (Chapter 2), 2) a new methodology for optimal repeater insertion in the *n-tier* multilevel interconnect architecture has been presented (Chapter 3), 3) analytical expressions have been derived for the transient response of distributed *rlc* interconnects, for a step and a ramp input, and they have been extended for two coupled lines (Chapter 4), 4) compact models have been derived for unified time delay, unified crosstalk and repeater insertion in distributed *rc* and *rlc* lines (Chapter

5), 5) the impact of inductance on the multilevel interconnect architecture has been quantified, by including effects of non-ideal return paths and additional shielding lines (Chapter 6), 6) a new aspect ratio design plane has been devised to optimize the multilevel interconnect aspect ratios (Chapters 2 and 6), and 7) a software tool has been developed to design the *n-tier* multilevel interconnect network (Chapter 7), which can be easily incorporated into other system design simulators such as GENESYS.

In addition there are eight new and interesting specific results from this dissertation. They are : 1) the *n-tier* methodology was used to decrease macrocell area, cycle time and wiring area on topmost tier by 32%, 16% and 62%, respectively, compared to a conventional design with doubling of wire pitches for each pair of metal layers, 2) the repeater insertion methodology was used to reduce macrocell area, cycle time, number of metal layers and power dissipation by 70%, 22%, 25% and 44%, respectively, compared to the *n-tier* design without repeaters, 3) macrocell area and power dissipation are minimized simultaneously, by optimal repeater insertion, compared to a design without repeaters, 4) compact models for time delay, crosstalk and repeater insertion are derived that are applicable to both *rc* and *rlc* regions of interconnect operation, and are therefore, "unified" models, 5) these models provide physical insight into signal propagation in the shallow *rlc* region, enabling optimization of global interconnect widths for maximum throughput [61], 6) careful insertion of repeaters reduces crosstalk by 51% and wiring cross-sectional area by 83%, without increasing time delay, 6) optimal values for the metal thickness-wire width aspect ratio (T/W) and oxide height to wire width aspect ratio (H/W) for each tier of a 6-layer 130nm technology generation case study are very similar to those of a commercial 130nm process technology, 8) inclusion of inductance increases area required for wiring due to increased spacing to satisfy crosstalk constraints, and introduction of additional shielding lines that decreases wiring efficiency.

In addition, from this research there are a variety of significant extensions that could be investigated including 1) an integrated system level inter and intra macrocell interconnect design 2) development of more accurate wiring efficiency models, 3) three dimensional architecture and diagonal interconnect designs and 4) including real drivers for accurate interconnect transient response models.

APPENDIX A

CODE FOR *N-TIER* METHODOLOGY

A.1 Notes

The C code for the *n-tier* design methodology is provided in Section A.2. A sample input file is given in Section A.3 and the corresponding output is given in Section A.4. The main inputs which need to be specified are the minimum feature size, number of logic gates, macrocell area, clock frequency, and values of the interconnect aspect ratios (S/W, T/W and H/W). Also, the total number of repeaters for the design can be constraied to a specified maximum value. The maximum permissible crosstalk can also be specified. Setting (resetting) two flags includes (precludes) inductive effects and worst case crosstalk models into the design.

Comments have been provided in the main C program to explain the function of different variables. A brief description of some of the subroutines in the main code is as follows :

- main() : It reads the input file, calls the sub-routines for the *n-tier* design and the *n-tier* with repeaters design, prints the output and exits.

- main_algo() : It implements the *n-tier* algorithm, without repeaters. The results

are written into an array, which is available to the other sub-routines.

- main_repeaters() : It takes the results of the main_algo() sub-routine, and inserts repeaters in the design. The output is written to a new array.
- Repeater_redesign() : It designs a tier by adding repeaters to it, using *rc* models.
- RLC_Repeater_redesign() : It designs a tier by adding repeaters to it, using *rlc* models.
- L2_calculation() : It solves a non-linear equation to calculate the longest interconnect on any tier.
- check_num_repeaters_OK() : It checks if area is available for the required number of repeaters for any design.
- Cm_calculation() : It calculates mutual capacitance using Chern's models.
- Cgnd_calculation() : It calculates ground capacitance using Chern's models.
- C_total() : It calculates the total capacitance.
- Cw_calculation() : It calculates the average wiring capacitance.
- pitch() : It calculates wire pitch, without repeaters.
- pitch_repeaters() : It calculates wire pitch, with *rc* repeaters.
- pitch_RLCrepeaters() : It calculates wire pitch, with *rlc* repeaters.
- Crosstalk_Redesign() : It recalculates the spacing between wires, and therefore the longest interconnect length of each tier, in order to satisfy specified crosstalk constraints.

- Crosstalk_OK() : It checks if specified crosstalk constraints are satisfied.
- RLC_Crosstalk() : It calculates crosstalk using *rlc* models.
- RC_Crosstalk() : It calculates crosstalk using *rc* models.
- RLC_worstcaseCrosstalk() : It calculates worstcase crosstalk using *rlc* models.
- RC_worstcaseCrosstalk() : It calculates worstcase crosstalk using *rc* models.
- Ro() : It calculates the output resistance of a minimum feature size inverter.
- Co() : It calculates the input capacitance of a minimum feature size inverter.
- L2_equation1() and L2_equation2() : Supply demand equation, using the Davis Distribution (Stochastic wiring distribution).
- L2_equationrepeaters1() and L2_equationrepeaters2() : Supply demand equation, with repeaters, using the Davis Distribution.
- demand_function() : Interconnect demand function, using the Davis Distribution.
- integral_region1() and integral_region2() : Equations for the Davis Distribution.
- GateArea() : Calculates area of a CMOS gate - author Keith Bowman.
- zbrent() : Solves a non-linear equation in one variable, from Numerical recipes in C.

A.2 Main code

```
/***PROGRAM FOR DESIGNING THE N-TIER MULTILEVEL INTERCONNECT NETWORK. THE
PROGRAM USES REPEATERS, AND INDUCTANCE MODELS FOR DELAY AND CROSSTALK, IF
REQUIRED.***/
```

```
#include <stdio.h>
#include<stdlib.h>
#include <math.h>
#include <string.h>
#include <time.h>
#define ITMAX 5000
#define EPS 3.0e-8
#define SIGN(a,b) ((b) >= 0.0 ? fabs(a) : -fabs(a))

#define kai     0.667    /* kai is point to point to net length
                            correction factor. kai = 4/(fanout + 3) */
#define alpha   0.75     /* alpha = fanout/(fanout+1) */
#define k       4        /* Rents coefficient k */
#define rents   0.6      /* Rents exponent p */
#define eff     0.4      /* Wiring efficiency is 40% */
#define rep_eff 0.6      /* Repeater usage efficiency is 60% */
#define rho     1673E-9  /* resistivity of metal, here Copper [Ohm-cm]*/
#define er      2.0      /* ILD permittivity. For low-k, er=2.0 */
#define eo      8850E-17 /* Permittivity of vaccuum */
#define co      3E10     /* Speed of light in vaccuum [cm/s]*/
#define max_no_levels 30 /* Maximum number of levels among all the
                            designs. Sets array size. Use a large
                            enough number */
#define tox     1.8e-7   /* Oxide thickness [nm] */
/*Should be 1.2e-7 for 45nm generation Ragu 05/20/2002*/

#define Vdd     1.0      /* Supply voltage [V] */
/*Should be 0.6V for 45nm generation Ragu 05/20/2002*/

#define Vtn     (Vdd/8.0) /* Long Channel Threshold voltage [V] */

#define ncp     10  /* Number of gates in Critical Path */
/*Should be 5 for 45nm generation Ragu 05/20/2002*/

float F;                          /* Minimum Feature Size*/
float N, N1,N2, delta_N;          /* Number of gates */
float fc, fc1, fc2, delta_fc;     /* Clock frequency */
float Ac, Ac1, Ac2, delta_Ac;     /* Macrocell area */
float Constrained_No_of_Repeaters;/* Total permissible # of repeaters */
float Cw;                         /* Wiring Capacitance */
float Gamma;                      /* Normalizing factor in Davis Dist. */
float activity=0.1;               /* activity factor is 10% */
float beta;                       /* Delay as a fraction of clock period*/
float beta_local=0.25;            /* beta of local tier */
float beta_global=0.9;            /* beta of global tier */
float L1=1.0;                     /* Shortest interconnect = 1gate pitch*/
float L2;                         /* Longest interconnect of a tier */
float Lavg;                       /* Average interconnect length*/
float L1temptemp;                 /* Temporary variable */
/*float zeta;*/                   /* Defined as a local variable zeta=0
```

```
                                    for no repeaters, 1 for opt rep, and
                                    <1 for subopt rep*/
float zetatemp;                  /* Temporary variable */
float zetatemp1;                 /* Temporary variable */
float zeta_inpitchrepeaters;     /* global variable used by
                                    pitch_repeaters function*/
float deltazeta=0.001;           /* Incremental value for zeta */
float norepeaters=0;             /* global flags used in L2_calculation*/
float withrepeaters=1;           /* global flags used in L2_calculation*/
float withRLCrepeaters=2;        /* global flags used in L2_calculation*/
float L2original[max_no_levels][13]; /*[Ln,pn,zeta, no of repeaters,power
                                    of repeaters,not pairs,wk of repeaters,
                                    W,S,T,H, flag_useRLCrepeaters,
                                    flag_useRLC]*/
float L2new[max_no_levels][13];  /* Contains the with  repeaters
                                    (suboptimal too) design**/
float L2temp[max_no_levels][13];
float rep_levels_n;
float rep_levels_n_temp;
float rep_levels_n_temp3;
float onlyopt_rep_levels_n;
float subopt_rep_levels_n;
float nlevels;
float size_of_repeaters;
float size_of_repeaters_this_design;
float size_of_repeaters_temp;
float WIDTH;                      /* Width of wires on any given tier */
float Spwire, Spwire1, Spwire2, delta_Spwire; /* Spwire=Wire Spacing/Width */
float Tmetal, Tmetal1, Tmetal2, delta_Tmetal; /* Tmetal=Metal thickness/Width*/
float Hoxide, Hoxide1, Hoxide2, delta_Hoxide; /* Hoxide=Oxide height/Width */
double Zo, Zoplus, Zominus;      /* Characteristic impedance */

double wk;                       /* width of nfet in keith's equations*/
double wk_logic;                 /* wk of logic gates, used for crosstalk
                                    calculations*/
int flag=0;  /* flag is set to 1 when pitch > 2F */
int flag2=0; /*flag2 = 0 means pitch is calculated using Width=F, i.e. minimum
 feature size width interconnects are used; flag2 = 1 means pitch is calculated
 using rc or rlc time delay models */
int flag2_is_0=1; /*depends on value of flag2. Used for writing to arrays*/
int flag4;   /* Used to set beta of lowest tier ALWAYS =  beta_local.
 flag4 =0 means first PAIR is being designed. It is set when the first
 two metal layers have been designed */
int flag5;   /*Indicates that beta of first PAIR has been forced to remain
 equal to beta_local, using flag4. Used to reset beta to beta_global for
 second PAIR of metal layers and higher tiers*/
int flag_stop_repeater_redesign;
int flag_stop_rep_redesign_maxreached;
int flag_no_repeaters_inserted;
int flag_start_optrep_design=0;
int index1; /* row number for arrays. index1 = tier number*/
```

```
int notpairs=0;
int flag_repeater_notpairs=0;
int flag_repeater_pitchis2F=0;
int flag_first_round;
int flag_useRLC=0;
int flag_useRLCrepeaters=0;
float globalflag_includeinductance;
float globalflag_worstcaseCrosstalk;
double Na;
double size_of_logic_gates;
float n_repeaters;
float n_repeaters_this_design;
float n_repeaters_temp;
float  power_n_repeaters;
float power_n_repeaters_this_design;
float power_n_repeaters_temp;
float power_logicgates;
float power_interconnects;
float Max_Crosstalk;

double fcmax;                    /* maximum frequency possible with given
                                    number of critical path gates*/

float main_algo(void);
float L_critical_calculation(float);
float L2_calculation(float, int);
float L2_equation1(float);
float L2_equation2(float);
float L2_equationrepeaters1(float);
float L2_equationrepeaters2(float);
float integral_region1(float, float);
float integral_region2(float, float);
float pitch(float);
float I1_calculation(float);
float m_calculation(float);
float actual_beta(float, float);
void main_repeaters(void);
float Repeater_redesign(float, float);
float RLC_Repeater_redesign(float);
float L2_repeater_calculation(float);
float pitch_repeaters(float);
float pitch_RLCrepeaters(float);
float Crosstalk_Redesign(float, float, float);
int Crosstalk_OK(float, float, float);
double RLC_Crosstalk(float,float,float);
double RC_Crosstalk(float,float,float);
double RC_worstcaseCrosstalk(void);
double RLC_worstcaseCrosstalk(void);
float lmax_calculation(float);
void Ctalk_global_parameter_calculation(void);
int check_num_repeaters_OK(float, float, float, float, int);
```

```
float mrep_calculation(float);
float Ro(void);
float Co(void);
float Cgnd_calculation(void);
float Cm_calculation(void);
float C_total(void);
float Cw_calculation(double);
float demand_function(float L1temp, float L2temp);
float powerdissipation(float , float);

void  keith(void);
double GateArea( double fin, double fout, float );
void Max_CP_freq( double fin, double fout );
float Ido_satdraincurrent(void);
double Cov( double fin, double betaA );
double Cjbottom( double fin, double betaA );
double Cjside_wall( double fin, double betaA );
double Cfout( double fout, double betaA );
double Cdevice( double fin, double fout, double betaA );
double Vtlong( void );
double NaEqn( double Vtntarget );

float zbrent(float (*func)(float), float x1, float x2, float tol);


main()
{
 /* THIS PROGRAM READS THE INPUT FILE, CALLS THE ROUTINES FOR N-TIER
    DESIGN AND N-TIER WITH REPEATERS DESIGN, PRINTS THE OUTPUT & EXITS */
 int i;
 float main_algo(void);
 float n;
 int temp;
 char junk[100];
 FILE *fp,*fp0,*fp1, *fp2;
 fp=fopen("out.dat","w");
 fp0=fopen("input.dat","r");
 fp1=fopen("out_powerdiss_constrepeaters.dat","w");
 fp2=fopen("results4plot.dat","w");
 fscanf(fp0, "%s %f", junk, &F);
 fscanf(fp0, "%s %f", junk, &N1);
 fscanf(fp0, "%s %f", junk, &N2);
 fscanf(fp0, "%s %f", junk, &delta_N);
 fscanf(fp0, "%s %f", junk, &Ac1);
 fscanf(fp0, "%s %f", junk, &Ac2);
 fscanf(fp0, "%s %f", junk, &delta_Ac);
 fscanf(fp0, "%s %f", junk, &fc1);
 fscanf(fp0, "%s %f", junk, &fc2);
 fscanf(fp0, "%s %f", junk, &delta_fc);
 fscanf(fp0, "%s %f", junk, &nlevels);
 /*Not used in this program - Ragu 05/24/2002*/
```

```
fscanf(fp0, "%s %f", junk, &Spwire1);
fscanf(fp0, "%s %f", junk, &Spwire2);
fscanf(fp0, "%s %f", junk, &delta_Spwire);
fscanf(fp0, "%s %f", junk, &Tmetal1);
fscanf(fp0, "%s %f", junk, &Tmetal2);
fscanf(fp0, "%s %f", junk, &delta_Tmetal);
fscanf(fp0, "%s %f", junk, &Hoxide1);
fscanf(fp0, "%s %f", junk, &Hoxide2);
fscanf(fp0, "%s %f", junk, &delta_Hoxide);
fscanf(fp0, "%s %f", junk, &Constrained_No_of_Repeaters);
fscanf(fp0, "%s %f", junk, &Max_Crosstalk);
fscanf(fp0, "%s %f", junk, &globalflag_includeinductance);
fscanf(fp0, "%s %f", junk, &globalflag_worstcaseCrosstalk);

Spwire=Spwire1;
Tmetal=Tmetal1;
Hoxide=Hoxide1;

printf("*****************************************************************\n")
printf("##Ro=%g, Co=%g, Ctotal=%g\n", Ro(),Co(),C_total());

for(fc=fc1; fc <fc2; fc=fc+delta_fc)      /*Loop for clock frequency*/
for(N=N1; N<=N2; N=N+delta_N)             /*Loop for number of gates*/
{
 Gamma=2*N*(1-pow(N,rents-1))/(-pow(N,rents)*((1+(2*rents)-
 pow(2,(2*rents-1)))/(rents*(2*rents-1)*(rents-1)*(2*rents-3)))-
 (1/(6*rents))+(2*sqrt(N)/((2*rents)-1))-N/(rents-1));
 printf("Gamma=%f\n", Gamma);
 for(Ac=Ac1; Ac<=Ac2; Ac=Ac+delta_Ac)     /*Loop for macrocell area*/
 {
  n = main_algo();                        /*main_algo does n-tier design*/
  printf("For N=%f, fc=%f and Ac=%f,no.of levels=%f\n\n",N, fc,Ac,n);
  temp=fprintf(fp, "%g, %g  %f  %f\n",N,fc,Ac,n);
  main_repeaters();                       /*main_repeaters inserts repeaters*/
  Max_CP_freq(2.0,3.0);
  printf("\nFor N=%f, fc=%f and Ac=%f,no.of levels=%f\n",N, fc,Ac,n);
  if(flag_no_repeaters_inserted==1) rep_levels_n=n;
  printf("the new total number of levels with constrained # of repeaters is
  %f\n",rep_levels_n);
  printf("wire limited area = %f, transistor limited area =%f\n",Ac,size_of_
  logic_gates+(size_of_repeaters_this_design/rep_eff));
  printf("logic  area = %f, rep area =%f\n",size_of_logic_gates,
  size_of_repeaters_this_design);
  printf("No of repeaters is %g,  Power_logic= %f, Power_rep = %f, Power_int =
  %f, Power_total= %f \n",  n_repeaters_this_design,power_logicgates, power_n_
  repeaters_this_design,power_interconnects,power_n_repeaters_this_design+powe
  r_logicgates+power_interconnects);
  temp=fprintf(fp2, "%g  %f  %g %f\n", fc, Ac, N, rep_levels_n);
  temp=fprintf(fp1, "%f %f  %f  %f %f %f %f  %f %g  %f  %f  %f  %f %f %f\n",
  fc/1E9,fcmax/1E9,Ac,size_of_logic_gates,size_of_repeaters_this_design,size_
  of_logic_gates+(size_of_repeaters_this_design/rep_eff),(size_of_logic_gates
```

```
   +(size_of_repeaters_this_design/rep_eff))/Ac,rep_levels_n,n_repeaters_this_
   design,power_logicgates,power_n_repeaters_this_design,power_interconnects,
   power_n_repeaters_this_design+power_logicgates+power_interconnects,(power_n_
   repeaters_this_design+power_logicgates+power_interconnects)/Ac,(power_n_
   repeaters_this_design+power_logicgates+power_interconnects)/50);
  }
 }
 printf("Program executed successfully\n");
 /*temp=fclose(*fp);*/
}


float main_algo(void)
{
 /* THIS IS THE MAIN ALGORITHM FOR N-TIER DESIGN, WITHOUT REPEATERS */
 float  L_critical,n=0,m=0;
 float junk;
 extern float L1; /*L1 is smallest interconnect length on any tier*/
 float I1;
 extern int flag;
 int flag1=0;     /*flag1 is set when topmost tier is reached i.e. L2 is
  greater than 2sqrt(N), and so top tier is incompletely filled*/
 float pitchtemp; /*local variable used to print values to screen*/

 L1=1;            /*for first tier, L1=1 gate pitch*/
 beta=beta_local; /*i.e. for first tier, beta equals beta_local*/
 flag2=0;         /*i.e. width=F, for first tier*/
 flag2_is_0=1;    /*i.e. flag2=0*/
 flag4=0;         /*i.e. first PAIR is being designed*/
 flag5=0;         /*i.e. not forced beta =beta_local for first PAIR, yet*/
 flag=0;          /*i.e. minimum width interconnects are being used*/
 index1=1;        /*i.e. Entries for first row of arrays; tier number = 1*/
 L2original[0][0]=1.0;  /*L1 of lowest tier is 1 gate pitch*/

 /* What follows is not the average interconnect length. To get the average
 interconnect lenght, Lavg should by multiplied by 4/(fo+3) */
 Lavg=((1/rents)-sqrt(N)/(rents-0.5)-1/(6*sqrt(N)*(rents+0.5)) +
 pow(N,rents)*((-rents-1+pow(4,(rents-0.5)))/(2*(rents+0.5)*(rents-0.5)*
 rents*(rents-1))))/(pow(N,(rents-0.5))*((-2*rents)-1+pow(2,(2*rents)-1))
 /(rents*((2*rents)-1)*(rents-1)*((2*rents)-3))-1/(6*rents*sqrt(N))+1/(rents
 -0.5)-sqrt(N)/(rents-1));
 /*************************************************************************/
 size_of_logic_gates=N*GateArea(2.0,3.0, 1.0);
 /*equals total area of all logic gates in the macrocell*/
 wk_logic=wk; /*This is used in Crosstalk calculations for size of logic gate,
 when no repeaters are used*/
 printf("AA Gate Area=%f, size_of_logic_gates = %f\n",GateArea(2.0,3.0,1.0)/
 pow(F,2),size_of_logic_gates);
 keith();
 printf("wk=%f, Size of logic gates is %g, size of chip is %f\n", wk, size_of
 _logic_gates, Ac);
```

```
if(size_of_logic_gates > Ac)
printf("WARNING!!! Area is not wire limited\n");
power_logicgates=N*powerdissipation(2.0,3.0); /*fanin of 2, fanout of 3*/
power_interconnects=activity*0.5*Cw_calculation(3)*N*pow(Vdd,2)*fc; /*fanin
of 2, fanout of 3*/
printf("Interconnects power: Cw=%g, Lavg=%g, power_interconnects=%g\n",Cw_
calculation(3),Lavg,power_interconnects);
printf("Power-D of logic gates is %g, of interconnects is %g\n",power_logic
gates, power_interconnects);
L_critical=zbrent(&L_critical_calculation, 0, 2.1*sqrt(N), 0.001);
printf("L_critical=%f\n",L_critical);
// if(globalflag_includeinductance==1) flag_useRLC=1; else flag_useRLC=0;

while (flag1==0) /*i.e. topmost tier is not reached*/
{
 Spwire=Spwire1;
 notpairs=0;     /*Both levels of a "Pair" have same aspect ratios*/
 flag_useRLC=0;  /*Try designing with RC models*/
 L2=L2_calculation(L1,norepeaters);
  /*Calculate L2, for the tier*/
 L2=Crosstalk_Redesign(L1,L2,0.0);
  /*Adjust spacing to meet crosstalk. Recalculate L2*/
 if (L2 <= sqrt(N)) pitchtemp=pitch(L2); else pitchtemp=pitch(sqrt(N));
 if(globalflag_includeinductance==1.0)
 {
  if ((pitchtemp/(1.0+Spwire)) > sqrt(1.1*rho*C_total()*co*L2*sqrt(Ac/N)/
  (Tmetal*sqrt(er))))
  {
   flag_useRLC=1; /*Use RLC models for delay*/
   L2=L2_calculation(L1,norepeaters);
   L2=Crosstalk_Redesign(L1,L2,0.0);
   if (L2 <= sqrt(N)) pitchtemp=pitch(L2); else pitchtemp=pitch(sqrt(N));
  }
 }
 printf("#beta=%f,  L1=%f, L2=%f, L2cms=%f,pitch=%g,actual_beta=%f,n=%f\n",
 beta,L1,L2,L2*sqrt(Ac/N),pitchtemp,actual_beta(L2, pitchtemp), n);
 if ((L2 > L_critical) && (flag==0))
  /*i.e. still on first tier, but width needs to be larger than F*/
 {
  if (flag4==0) /*i.e. if first PAIR is being designed*/
  {
   beta=beta_local; /*then beta forced to remain = beta_local*/
   flag5=1;         /*Set flag5 to indicate this has been done*/
  }
  else beta=beta_global;
  flag2=1;      /*width is now calculated using delay models, not = F*/
  flag2_is_0=0; /*flag2 is not 0*/
  flag_useRLC=0;/*Use RC models*/
  L2=L2_calculation(L1, norepeaters);
  printf("##beta=%f,  L1=%f, L2=%f,L2cms=%f,pitch=%g,actual_beta=%f,n=%f\n",
  beta,L1,L2,L2*sqrt(Ac/N),pitchtemp,actual_beta(L2, pitchtemp), n);
```

```
 L2=Crosstalk_Redesign(L1,L2,0.0);
 if (L2 <= sqrt(N)) pitchtemp=pitch(L2); else pitchtemp=pitch(sqrt(N));
 printf("###beta=%f,  L1=%f,L2=%f,L2cms=%f,pitch=%g,actual_beta=%f,n=%f\n",
 beta,L1,L2,L2*sqrt(Ac/N),pitchtemp,actual_beta(L2, pitchtemp), n);
 if(globalflag_includeinductance==1.0)
 {
  if ((pitchtemp/(1.0+Spwire)) > sqrt(1.1*rho*C_total()*co*L2*sqrt(Ac/N)/(
  Tmetal*sqrt(er))))
  {
   flag_useRLC=1;
   L2=L2_calculation(L1,norepeaters);
   L2=Crosstalk_Redesign(L1,L2,0.0);
   if (L2 <= sqrt(N)) pitchtemp=pitch(L2); else pitchtemp=pitch(sqrt(N));
  }
 }
 printf("beta=%f,  L1=%f, L2=%f,  L2cms=%f, pitch=%f, actual_beta=%f,n=%f\n",
 beta,L1,L2,L2*sqrt(Ac/N),pitchtemp,actual_beta(L2, pitchtemp), n);
 if( pitch(L2) < ((1.0+Spwire)*F))
 /* if width is less than F (due to pitch calculation), then it is corrected*/
 {
  flag2=0;        /*Force Width = F*/
  flag2_is_0=1;  /*flag2 is 0*/
  L2=L2_calculation(L1,norepeaters);
  L2=Crosstalk_Redesign(L1,L2,0.0);
  if (L2 <= sqrt(N)) pitchtemp=pitch(L2); else pitchtemp=pitch(sqrt(N));
  if(globalflag_includeinductance==1.0)
  {
   if ((pitchtemp/(1.0+Spwire)) > sqrt(1.1*rho*C_total()*co*L2*sqrt(Ac/N)/(
   Tmetal*sqrt(er))))
   {
    flag_useRLC=1;
    L2=L2_calculation(L1,norepeaters);
    L2=Crosstalk_Redesign(L1,L2,0.0);
    if (L2 <= sqrt(N)) pitchtemp=pitch(L2); else pitchtemp=pitch(sqrt(N));
   }
  }
  printf("beta=%f,  L1=%f, L2=%f,L2cms=%f,  pitch=%f, actual_beta=%f,n=%f\n",
  beta,L1,L2,L2*sqrt(Ac/N),pitchtemp,actual_beta(L2, pitchtemp), n);
  flag2=1; /*Width is calculated using delay models*/
           /*flag2_is_0 is reset later*/
 }
 else    flag=1; /*i.e. width is greater than F*/
}
if (fabs(L2-2*sqrt(N)) > 1 && (L2 < 2*sqrt(N)) )
 /*if L2 is less than 2sqrt(N), i.e. topmost tier is not yet reached*/
{
 if (flag4==0) flag4=1; /*if this is the first PAIR, then set flag4*/
 n=n+2.0;  /*increment number of levels by 2*/
 L1=L2;    /*L1 of next tier = L2 of previous tier*/
 /*The design of tier # index1 is written into the array L2original[]*/
 L2original[index1][0]=L2;
```

```
  L2original[index1][1]=pitchtemp;
  L2original[index1][2]=0; /*i.e. no repeaters in this tier*/
  L2original[index1][5]=0;/*not pairs is 0*/
  L2original[index1][7]=    1.0*L2original[index1][1]/(1.0+Spwire);
  WIDTH=L2original[index1][7];
  L2original[index1][8]= Spwire*L2original[index1][1]/(1.0+Spwire);
  L2original[index1][9]= Tmetal*L2original[index1][1]/(1.0+Spwire);
  L2original[index1][10]=Hoxide*L2original[index1][1]/(1.0+Spwire);
  L2original[index1][11]=0;/*flag_useRLCrepeaters=0*/
  L2original[index1][12]=flag_useRLC; /*Indicates if n-tier design w/o
  repeaters is ToF limited or not*/
  if(flag2_is_0==1) /*i.e. if minimum width wires had been used */
  {
   L2original[index1][1]= (1.0+Spwire)*F;
   L2original[index1][7]= 1.0*L2original[index1][1]/(1.0+Spwire);
   WIDTH=L2original[index1][7];
   L2original[index1][8]= Spwire*L2original[index1][1]/(1.0+Spwire);
   L2original[index1][9]= Tmetal*L2original[index1][1]/(1.0+Spwire);
   L2original[index1][10]=Hoxide*L2original[index1][1]/(1.0+Spwire);
   L2original[index1][11]=0;/*flag_useRLCrepeaters=0*/
   L2original[index1][12]=flag_useRLC; /*Indicates if n-tier design w/o
   repeaters is ToF limited or not*/
   flag2_is_0=0; /*reset to reflect value of flag2*/
  }
  index1=index1+1.0; /*increment tier number, which is also array index*/
  if (flag5==1) beta=beta_global;
   /*i.e. if flag5 was set, then force beta to beta_global for next pair.
     If flag5 was not set, next pair will also belong to the first tier. */
 }
 else flag1=1; /* topmost tier is reached. so exit while loop*/
}

/*Following commands are evaluated only when topmost tier is reached*/
I1=I1_calculation(L1);
m=m_calculation(I1);
if(m > (2.0))
printf("\nError in calculating global tier\n\n");
printf("m=%f, global pitch=%f\n",m,pitch(sqrt(N)));
n=n+m; /*ERROR HERE : SHOULD BE n=n+2*m. I LEAVE IT UNCORRECTED BECAUSE
 ALL PLOTS/NUMBERS/TABLES IN THESIS WERE CREATED WITH n=n+m*/
L2original[index1][0]=2*sqrt(N);
junk=Crosstalk_Redesign(L1,sqrt(N),0.0);
L2original[index1][1]= pitch(sqrt(N));
L2original[index1][2]= 0; /*no repeaters here*/
L2original[index1][7]= 1.0*L2original[index1][1]/(1.0+Spwire);
WIDTH=L2original[index1][7];
L2original[index1][8]= Spwire*L2original[index1][1]/(1.0+Spwire);
L2original[index1][9]= Tmetal*L2original[index1][1]/(1.0+Spwire);
L2original[index1][10]=Hoxide*L2original[index1][1]/(1.0+Spwire);
L2original[index1][11]=0;/*flag_useRLCrepeaters=0*/
L2original[index1][12]=flag_useRLC; /*Indicates if n-tier design w/o
```

```
 repeaters is ToF limited or not*/

 /*The n-tier design,without repeaters is now in the array L2original[]*/

 printf("the number of levels is %f\n",n);
 return n;
}


void main_repeaters()
{
 /* THIS IS THE MAIN ALGORITHM FOR N-TIER DESIGN, WITH REPEATERS */
 int i,j,temp;
 int index=0, index2=0;
 int flag_current_suboptdesign_OK=0;
 int flag_stop_suboptrep_redesign=0;
 float  rep_levels_n_temp2;
 float k_opt;
 flag_start_optrep_design=1;
 flag_stop_repeater_redesign=0;
 flag_no_repeaters_inserted=1;

 /*The original array for n-tier design without repeaters, L2original[]
   is copied to a new array L2new[] and a temporary array L2temp[]*/
 for(i=0; i < max_no_levels; i++)
 {
  for(j=0;j<=12;j++)
  {
   L2new[i][j]=L2original[i][j];
   L2temp[i][j]=L2original[i][j];
  }
 }

 /********Suboptimal code starts here****************/
 /*********************************************************************/
 printf("\n***N-Tier Design Before Repeater Insertion***\n");
 for(i=0; i < max_no_levels; i++)
 {
  for(j=0;j<=12;j++)
  {
   L2temp[i][j]=L2new[i][j];
  }
 }
 for(i=0;i<=index1;i++)
 {
  printf("L2[%d][0]=%f,  pitch=%g, UseRLC=%d, zeta=%g, #rep=%g,powrep=%g",i,
  L2temp[i][0],
  L2temp[i][1], L2temp[i][12], L2temp[i][2], L2temp[i][3],L2temp[i][4]);
  printf("W=%g, Spwire=%g, Tmetal=%g, Hoxide=%g",L2temp[i][7],L2temp[i][8]/
  L2temp[i][7],
  L2temp[i][9]/L2temp[i][7], L2temp[i][10]/L2temp[i][7]);
```

```
 printf("\n");
}
printf("\n");

n_repeaters_this_design=0;
/**Total number of repeaters of valid design is initialized to 0*/
power_n_repeaters_this_design=0;
/***Total power of repeaters of valid design is initialized to 0 here*/
size_of_repeaters_this_design=0;

while(flag_stop_repeater_redesign==0)
 /*i.e. while repeater design is not stopped*/
{
 for(index2=index1;index2>=1;index2--)
  /*start from the topmost tier, and proceed downwards*/
 {
  if( (L2temp[index2][7]-(1.0*F) > 0.1*F) && (L2temp[index2-1][0] <
    (2*sqrt(N) + 1)) )
   /*Proceed only if width is greater than F, AND, previous (lower)
     tier is NOT the topmost tier*/
  {
   flag_current_suboptdesign_OK=0;
    /*Initialize flag indicating that current tier needs repeaters*/
   if(index2 > 1) beta=beta_global;
   else beta=beta_local;
    /*sets beta values depending on which tier is being
      redesigned with repeaters*/
   L2temp[index2][2]=0.5; /*Zeta is initialized to 0.5*/
   zetatemp=0.5; /*Default value of zeta is 0.5*/
   k_opt=4.0; /*Number of repeaters initialized to some value, greater than 2*/

   while((flag_current_suboptdesign_OK==0)&&(flag_stop_suboptrep_redesign==0))
    /*i.e. current design needs repeaters and repeater redesign is'nt stopped*/
   {
    n_repeaters=0;       /*Number of repeaters is reset to 0 here*/
    power_n_repeaters=0; /*Power of repeaters is reset to 0 here*/
    size_of_repeaters=0; /*Size of repeaters is reset to 0 here*/
    if(zetatemp<=(1/k_opt))
     /*Smallest value of zeta is such that zeta*k_opt=1. If zeta is decreased
       to less than this value, then stop the repeater design and break */
    {
     flag_stop_suboptrep_redesign=1;
     break;
    }
    else /*proceed with repeater redesign on the current tier*/
    {
     flag_stop_rep_redesign_maxreached=0;
      /*the previous (lower) tier was not the topmost tier*/
     temp=index2;
      /*Set temp equal to # of current tier which is being redesigned*/
     while(flag_stop_rep_redesign_maxreached==0)
```

```
 /*while topmost tier has not yet been redesigned*/
{
 /*read width, S/W,T/W,H/W of current tier from array*/
 WIDTH=L2temp[temp][7];
 Spwire=Spwire1;
 Tmetal=L2temp[temp][9]/L2temp[temp][7];
 Hoxide=L2temp[temp][10]/L2temp[temp][7];
 rep_levels_n_temp=2*(temp-1);
  /*equal to number of levels below current tier*/
 if(L2temp[temp-1][5]==1) rep_levels_n_temp=2*(temp-1) - 1;
  /*accounting for not pairs*/
 zeta_inpitchrepeaters=L2temp[temp][2];
  /*read zeta value from array*/
 flag_useRLCrepeaters=0;
  /*use RC repeater models*/
 L2temp[temp][11]=0;  /*flag_useRLCrepeaters=0*/
  /*flag_useRLCrepeaters=0 means use rc models for repeaters. It is a
  global variable and will be set to 1 if needed*/
 if(globalflag_includeinductance==0)
 {
  L2temp[temp][0]=Repeater_redesign(L2temp[temp-1][0],L2temp[temp][2]);
   /*Recalculate L2 for current tier, with repeaters*/
  L2temp[temp][0]=Crosstalk_Redesign(L2temp[temp-1][0],L2temp[temp][0],
  L2temp[temp][2]);
   /*Adjust spacing and recalculate L2, to satisfy crosstalk*/
   /*Then write all values to array*/
  L2temp[temp][1]=pitch_repeaters(L2temp[temp][0]);
  L2temp[temp][7]=    1.0*L2temp[temp][1]/(1.0+Spwire);
  WIDTH=L2temp[temp][7];
  L2temp[temp][8]= Spwire*L2temp[temp][1]/(1.0+Spwire);
  L2temp[temp][9]= Tmetal*L2temp[temp][1]/(1.0+Spwire);
  L2temp[temp][10]=Hoxide*L2temp[temp][1]/(1.0+Spwire);
 }
 else
 {
  /*Use rlc models, if needed, for time delay*/
  flag_useRLCrepeaters=0;
  L2temp[temp][11]=0;  /*flag_useRLCrepeaters=0*/
  /*Start with rc models. will use RLC if needed*/
  /*Recalculate L2, pitch etc. and write to array*/
  L2temp[temp][0]=Repeater_redesign(L2temp[temp-1][0],L2temp[temp][2]);
  L2temp[temp][0]=Crosstalk_Redesign(L2temp[temp-1][0],L2temp[temp][0],
  L2temp[temp][2]);
  L2temp[temp][1]=pitch_repeaters(L2temp[temp][0]);
  L2temp[temp][7]=    1.0*L2temp[temp][1]/(1.0+Spwire);
  WIDTH=L2temp[temp][7];
  L2temp[temp][8]= Spwire*L2temp[temp][1]/(1.0+Spwire);
  L2temp[temp][9]= Tmetal*L2temp[temp][1]/(1.0+Spwire);
  L2temp[temp][10]=Hoxide*L2temp[temp][1]/(1.0+Spwire);

  if(((L2temp[temp][0]*sqrt(Ac/N))>(1.332/(1.4+0.53*
```

```
  zeta_inpitchrepeaters+0.53/
 zeta_inpitchrepeaters))*beta*co/(fc*sqrt(er))) )
  /*if TOF is violated, then use rlc models*/
 {
  /*use RLC repeater models*/
  /*if((globalflag_includeinductance==1.0) &&(L2temp[temp][7]*
  L2temp[temp][9]) >= (pow(1.332*co,2.0)*rho*C_total()*Ro()*Co()/er) )*/
  {
   flag_useRLCrepeaters=1;
   L2temp[temp][11]=1.0;
   L2temp[temp][0]=RLC_Repeater_redesign(L2temp[temp-1][0]);
   L2temp[temp][0]=Crosstalk_Redesign(L2temp[temp-1][0],L2temp[temp][0],
   L2temp[temp][2]);
   L2temp[temp][1]=pitch_RLCrepeaters(L2temp[temp][0]);
   L2temp[temp][7]=    1.0*L2temp[temp][1]/(1.0+Spwire);
   WIDTH=L2temp[temp][7];
   L2temp[temp][8]= Spwire*L2temp[temp][1]/(1.0+Spwire);
   L2temp[temp][9]= Tmetal*L2temp[temp][1]/(1.0+Spwire);
   L2temp[temp][10]=Hoxide*L2temp[temp][1]/(1.0+Spwire);
  }
 }
}
k_opt = sqrt(0.4*rho*(C_total())/(0.7*L2temp[temp][1]*Tmetal*
L2temp[temp][1]*Ro()*Co()));
/*NOTE : k_opt is half the Bakoglu optimum number of repeaters. It is
not used in any important calculation, so I have not corrected it -
Ragu 05/26/2002*/

if(check_num_repeaters_OK(L2temp[temp-1][0],L2temp[temp][0],
L2temp[temp][7],L2temp[temp][2],
temp)==0)
/*if area is not available for that many repeaters */
{
 flag_current_suboptdesign_OK=0;
 /*then current design is not OK. so break (and then redo
 the calculation with a smaller zeta)*/
 /*printf("Ran out of repeater space\n");*/
 break;
}
else
{
 /*current design is ok. so add number of layer in current tier to
   previous total, and move to next (upper) tier*/
 rep_levels_n_temp2=rep_levels_n_temp+rep_levels_n_temp3;
 flag_current_suboptdesign_OK=1;
 if (fabs(L2temp[temp][0]-2*sqrt(N)) < 1.0)
 rep_levels_n_temp2=rep_levels_n_temp+rep_levels_n_temp3;
}

if((fabs(L2temp[temp][0]-2*sqrt(N)) < 1.0)  ||  (L2temp[temp][0] >
2*sqrt(N)) )
```

```
      /*if current tier was the topmost tier, then dont proceed any further*/
     {
      flag_stop_rep_redesign_maxreached=1;
       /*current tier is topmost tier. so do not proceed further*/
      //break;
     }
     else
     {
      /*proceed to next (upper) tier*/
      temp++;
     }
    }
   }
   if(flag_current_suboptdesign_OK==0)
   {
    /*if current design was not OK, then redo with a smaller zeta*/
    zetatemp=L2temp[index2][2]-deltazeta;
    L2temp[index2][2]=zetatemp;
   }
   else
   {
    L2temp[index2][2]=zetatemp;
    flag_no_repeaters_inserted=0;
    n_repeaters_this_design=n_repeaters;
    power_n_repeaters_this_design=power_n_repeaters;
    size_of_repeaters_this_design=size_of_repeaters;
   }
  }
  if(flag_current_suboptdesign_OK==1)
  {
   flag_stop_repeater_redesign=0;
   rep_levels_n=rep_levels_n_temp2;
   for(i=0;i < max_no_levels; i++)
   {
    for(j=0;j<=12;j++)
    {
     L2new[i][j]=L2temp[i][j];
    }
   }
   rep_levels_n=rep_levels_n_temp2;
   subopt_rep_levels_n=rep_levels_n;
  }
  else
  {
   flag_stop_repeater_redesign=1;
   break;
  }
 }
 else
 {
  printf("Tier %d iteration aborted because it is already minimum
```

```
    feature size\n",index2);
   }
   if(flag_stop_repeater_redesign==1) break;
   if (index2==0) break;
  }
  if(index2==0)
  {
   flag_stop_repeater_redesign=1;
  }
 }
 /*************************************************************************/
 /********Suboptimal code ends here****************/
 printf("\n***N-Tier Design After Repeater Insertion***\n");
 for(i=0;i<=index1;i++)
 {
  printf("L2[%d]=%f,  pitch=%g, W=%g, Spwire=%g, Tmetal=%g,
  Hoxide=%g",i,L2new[i][0],L2new[i][1],
  L2new[i][7],L2new[i][8]/L2new[i][7], L2new[i][9]/L2new[i][7],
  L2new[i][10]/L2new[i][7]);
  printf(" nrep=%g, RLCrep=%f\n",L2new[i][3], L2new[i][11]);
 }
}


float powerdissipation(float fanin, float fanout)
{
 /* Gives the power dissipation of a gate*/
 float Cload, Pout;
 Cload=wk*Cdevice(fanin, fanout, 2/fanin);
 Pout=activity*0.5*Cload*pow(Vdd,2)*fc;
 return Pout;
}


float Repeater_redesign(float temp, float zeta)
{
 /* Calculates L2 with repeaters*/
 float L2temp,I1rep, mrep;
 flag_repeater_pitchis2F=0;
 flag_repeater_notpairs=0;
 rep_levels_n_temp3=0;
 L1temptemp=temp;
 zetatemp1=zeta;
 zeta_inpitchrepeaters=zeta;
 L2temp=L2_calculation(L1temptemp, withrepeaters);
 if(pitch_repeaters(L2temp) < ((1.0+Spwire)*F))
 {
  flag_repeater_pitchis2F=1;
  L2temp=L2_calculation(L1temptemp, withrepeaters);
 }
 if( (fabs(L2temp-2*sqrt(N)) > 1.0) && (L2temp < 2*sqrt(N)))
```

```
 /*If L2 is less than 2*sqrt(N),increment no of levels by 2*/
 {
  rep_levels_n_temp3 = 2.0 ;
 }
 else
 /*else set L2 = sqrt(N) and calculate fraction of level occupied*/
 {
  L2temp=2*sqrt(N);
  if(pitch_repeaters(L2temp) < (1.0*Spwire)*F) flag_repeater_pitchis2F=1;
  I1rep=I1_calculation(L1temptemp);
  mrep=mrep_calculation(I1rep);
  if(mrep > 2.0)
  printf("Error in calculating global tier with repeaters\n");
  rep_levels_n_temp3=rep_levels_n_temp3+mrep;
 }
 return L2temp;
}


float RLC_Repeater_redesign(float temp)
{
 /* Calculates L2 with repeaters using RLC models for optimum repeater
 insertion*/
 float L2temp,I1rep, mrep;
 flag_repeater_pitchis2F=0;
 flag_repeater_notpairs=0;
 rep_levels_n_temp3=0;
 L1temptemp=temp;
 L2temp=L2_calculation(L1temptemp, withRLCrepeaters);
 /*if(pitch_repeaters(L2temp) < ((1.0+Spwire)*F))
 {
  flag_repeater_pitchis2F=1;
  L2temp=L2_calculation(L1temptemp, withrepeaters);
 }*/
 if( (fabs(L2temp-2*sqrt(N)) > 1.0) && (L2temp < 2*sqrt(N)))
 /*If L2 is less than 2*sqrt(N), increment no of levels by 2*/
 {
  rep_levels_n_temp3 = 2.0 ;
 }
  else
 /*else set L2 = sqrt(N) and calculate fraction of level occupied*/
 {
  L2temp=2*sqrt(N);
  if(pitch_repeaters(L2temp) < (1.0*Spwire)*F) flag_repeater_pitchis2F=1;
  I1rep=I1_calculation(L1temptemp);
  mrep=mrep_calculation(I1rep);
  if(mrep > 2.0)
  printf("Error in calculating global tier with repeaters\n");
  rep_levels_n_temp3=rep_levels_n_temp3+mrep;
 }
 return L2temp;
```

```
}


float L2_calculation(float raguL1, int flag_withrepeaters)
{
 /*CALCULATES L2 USING L1*/
 float remainder;
 L1=raguL1;
 if (flag_withrepeaters==0)
 {
  L2=zbrent(&L2_equation1,L1,2.1*sqrt(N),0.001);
  remainder=L2_equation1(L2);
  if ((fabs(L2 - 2*sqrt(N))>1) && (L2< 2*sqrt(N)) && (fabs(remainder)>0.01))
  {
   L2=zbrent(&L2_equation1,L2,2.1*sqrt(N),0.001);
   remainder=L2_equation1(L2);
  }
  if ((L1 < sqrt(N)) && (L2 > sqrt(N)) )
  {
   L2=zbrent(&L2_equation2,sqrt(N),2.1*sqrt(N),0.001);
   remainder=L2_equation2(L2);
   if ((fabs(L2 - 2*sqrt(N)) > 1)  && (L2 <  2*sqrt(N)) &&
   (fabs(remainder) > 0.01))
   {
    L2=zbrent(&L2_equation2,L2,2.1*sqrt(N),0.001);
    remainder=L2_equation2(L2);
   }
  }
  if (L2 > 2*sqrt(N))
  L2=2*sqrt(N);
 }
 else  if (flag_withrepeaters==1)
 {
  zeta_inpitchrepeaters=zetatemp1;
  L2=zbrent(&L2_equationrepeaters1,L1,2.1*sqrt(N),0.001);
  remainder=L2_equationrepeaters1(L2);
  if ((fabs(L2 - 2*sqrt(N)) > 1)  && (L2 <  2*sqrt(N)) &&
  (fabs(remainder) > 0.01))
  {
   L2=zbrent(&L2_equationrepeaters1,L2,2.1*sqrt(N),0.001);
   remainder=L2_equationrepeaters1(L2);
  }
  if ((L1 < sqrt(N)) && (L2 > sqrt(N)) )
  {
   L2=zbrent(&L2_equationrepeaters2,sqrt(N),2.1*sqrt(N),0.001);
   remainder=L2_equationrepeaters2(L2);
   if ((fabs(L2 - 2*sqrt(N)) > 1)  && (L2 <  2*sqrt(N)) &&
   (fabs(remainder) > 0.01))
   {
    L2=zbrent(&L2_equationrepeaters2,L2,2.1*sqrt(N),0.001);
    remainder=L2_equationrepeaters2(L2);
```

```
    }
   }
   if (L2 > 2*sqrt(N))  L2=2*sqrt(N);
  }
  else /*if flag_withrepeaters=2, i.e. rlc repeaters are used*/
  {
   L2=zbrent(&L2_equationrepeaters1,L1,2.1*sqrt(N),0.001);
   remainder=L2_equationrepeaters1(L2);
   if ((fabs(L2 - 2*sqrt(N)) > 1)  && (L2 <  2*sqrt(N)) &&
   (fabs(remainder) > 0.01))
   {
    L2=zbrent(&L2_equationrepeaters1,L2,2.1*sqrt(N),0.001);
    remainder=L2_equationrepeaters1(L2);
   }
   if ((L1 < sqrt(N)) && (L2 > sqrt(N)) )
   {
    L2=zbrent(&L2_equationrepeaters2,sqrt(N),2.1*sqrt(N),0.001);
    remainder=L2_equationrepeaters2(L2);
    if ((fabs(L2 - 2*sqrt(N)) > 1)  && (L2 <  2*sqrt(N)) &&
    (fabs(remainder) > 0.01))
    {
     L2=zbrent(&L2_equationrepeaters2,L2,2.1*sqrt(N),0.001);
     remainder=L2_equationrepeaters2(L2);
    }
   }
   if (L2 > 2*sqrt(N)) L2=2*sqrt(N);
  }
  return L2;
}


int check_num_repeaters_OK(float L1temp, float L2tempp, float width,
    float zeta, int temp)
{
 /** Checks if number of repeaters is ok : returns 1 if number of
 repeaters is less than constrained number of repeaters AND area of
 repeaters plus logic gates is less than free area, else returns 0**/
 /**The metal spacing, metal thickness and oxide height are global
 parameters. Therefore, they need not be passed as arguments to this
 function - Ragu 05/26/2002*/
 float RC_kopt, RC_hopt, RLC_kopt, RLC_hopt;
 float Lrep_min,bakoglu_scaling_factor;
 n_repeaters_temp=0;
 power_n_repeaters_temp=0;
 size_of_repeaters_temp=0;
 if (flag_useRLCrepeaters==0)
 {
  Lrep_min=sqrt((0.7*Ro()*Co()*N*width*Tmetal*width)/(0.4*rho*C_total()*Ac));
  /*Lrep_min is the minimum length of interconnect of a specified width,
  that needs repeaters. It is calculated by setting Bakoglu's expression
  for optimum repeaters equal to 1 - Ragu 05/26/2002 */
```

```
 RC_hopt = sqrt(Ro()*C_total()*width*Tmetal*width/(rho*Co()));
 bakoglu_scaling_factor = RC_hopt;
 /*bakoglu_scaling_factor is equal to h_opt in Bakoglu's expression for
 repeater size - Ragu 05/26/2002*/
 RC_kopt=zeta*sqrt((0.4*rho*C_total())/(0.7*Ro()*Co()*width*Tmetal*width));
 /*RC_kopt is the optimum number of RC repeaters for a specified width
 of a unit length of interconnect - Ragu 05/27/2002*/
 if(Lrep_min <= L1temp)
  n_repeaters_temp=RC_kopt*sqrt(Ac/N)*(demand_function(L1temp,L2tempp));
 else
  n_repeaters_temp=RC_kopt*sqrt(Ac/N)*(demand_function(Lrep_min,L2tempp));
 n_repeaters=n_repeaters+n_repeaters_temp;
 /*n_repeaters is total number of (optimal RC) repeaters required for
 the range of interconnect lengths from L1 to L2*/
 size_of_repeaters_temp= n_repeaters_temp*GateArea(1.0,1.0,
 bakoglu_scaling_factor);
 power_n_repeaters_temp=n_repeaters_temp*powerdissipation(1.0,1.0);
 L2temp[temp][3]=n_repeaters_temp;
 L2temp[temp][4]=power_n_repeaters_temp;
 L2temp[temp][6]=bakoglu_scaling_factor;
 power_n_repeaters=power_n_repeaters+power_n_repeaters_temp;
 size_of_repeaters=size_of_repeaters+size_of_repeaters_temp;
}
else
{
 Lrep_min=(sqrt(er)*width*Tmetal*width)/(0.9513*rho*co*C_total());
 /*Lrep_min is the minimum length of interconnect of a specified width,
 that needs repeaters. It is calculated by setting new RLC expression for
 optimum repeaters equal to 1 - Ragu 05/27/2002 */
 RLC_kopt=(0.9513*rho*co*C_total()/(sqrt(er)*width*Tmetal*width));
 /*RLC_kopt is the optimum number of RLC repeaters for a specified width
 of a unit length of interconnect - Ragu 05/27/2002*/
 if(Lrep_min <= L1temp)
  n_repeaters_temp=RLC_kopt*sqrt(Ac/N)*(demand_function(L1temp,L2tempp));
 else
  n_repeaters_temp=RLC_kopt*sqrt(Ac/N)*(demand_function(Lrep_min,L2tempp));
 /*n_repeaters is total number of (optimal RLC) repeaters required for
 the range of interconnect lengths from L1 to L2*/
 RLC_hopt=1.1479*Ro()*co*C_total()/sqrt(er);
 n_repeaters=n_repeaters+n_repeaters_temp;
 size_of_repeaters_temp=n_repeaters_temp*GateArea(1.0,1.0,RLC_hopt);
 power_n_repeaters_temp=n_repeaters_temp*powerdissipation(1.0,1.0);
 L2temp[temp][3]=n_repeaters_temp;
 L2temp[temp][4]=power_n_repeaters_temp;
 L2temp[temp][6]=RLC_hopt;
 power_n_repeaters=power_n_repeaters+power_n_repeaters_temp;
 size_of_repeaters=size_of_repeaters+size_of_repeaters_temp;
}
if( (((size_of_repeaters/rep_eff)+size_of_logic_gates) > Ac)  ||
(n_repeaters > Constrained_No_of_Repeaters))
{
```

```
  /* if(n_repeaters>Constrained_No_of_Repeaters) printf("Rejecting
  because number of repeaters constraint is violated\n"); */
  /* if(  size_of_repeaters/rep_eff + size_of_logic_gates > Ac)
  printf("Rejecting because area constraint is violated\n");*/
  return 0;
 }
 else
 {
  //printf("Area and number of repeaters constraints are met\n");
  return 1;
 }
}


float Cm_calculation(void)
{
 /* CALCULATES MUTUAL CAPACITANCE USING CHERN'S MODELS */
 float Cm;
 Cm=(Tmetal*eo*er/Spwire)*(1.0-1.897*exp((-Hoxide/(0.31*Spwire))+(-Tmetal/(
    2.474*Spwire)))+1.302*exp(-Hoxide/(0.082*Spwire))-0.1292*exp(-Tmetal/
    (1.326*Spwire)))+1.722*eo*er*exp(-Spwire/(0.651*Hoxide))*(1-(0.6548*
    exp(-1.0/(0.3477*Hoxide))));
 return Cm;
}


float Cgnd_calculation(void)
{
 /* CALCULATES GROUND CAPACITANCE USING CHERN'S MODELS */
 float Cgnd;
 Cgnd = (1.0*eo*er/Hoxide) +1.086*eo*er*(1+0.685*exp(-Tmetal/(1.343*Spwire))
        -0.9964*exp(-Spwire/(1.421*Hoxide)))*pow(Spwire/(Spwire+2*Hoxide),
        0.0476)*pow(Tmetal/Hoxide,0.337);
 return Cgnd;
}


float C_total(void)
{
 /* CALCULATES TOTAL CAPACITANCE FOR 3-LINE 2-GROUND PLANE, NON-WORST CASE*/
 float C_tot;
 C_tot = 2*Cgnd_calculation() + 2*Cm_calculation();
 return C_tot;
}


float Cw_calculation(double fo)
{
 /** Calculates wiring capacitance. Lavg times kai is the actual average
 interconnect length*/
 float Cw;
```

```
 Cw=(4/(fo+3))*Lavg*fo*sqrt(Ac/N)*C_total();
 return Cw;
}


float L_critical_calculation(float x)
{
 /* Returns the critical value of L2 beyond which minimum width cannot be
 used*/
 float pitch(float x);
 float y;
 flag2=1;
 flag2_is_0=0;
 y=pitch(x)-((1.0+Spwire)*F);
 flag2=0;
 flag2_is_0=1;
 return y;
}


float  actual_beta(float x, float pp)
{
  /*This sub routine is incorrect. It is not important, so I did not
  bother to correct it - Ragu 05/26/2002 */
  float y;
  y=fc*4*pow(x,2.0)*(Ac/N)*(1.1*rho*er*eo*6.08)/(pow(pp,2.0)) ;
  /*y=fc*4*pow(x,2)*(Ac/N)*(1.1*rho*er*6.08*eo);*/
  /*y=eo/(pow(pp,2));*/
  /*printf("fc=%f, x=%f, Ac=%f, N=%f, y=%f\n", fc,x,Ac,N,y);*/
  return y;
}


float pitch(float x)
{
 /* Calculates pitch without using repeaters*/
 float y;
 extern int flag2;
 if (flag_useRLC==0)
 {
  /* calculates pitch without using repeaters, using RC models*/
  if(flag2==1)
  {
    y=((1.0+Spwire)/sqrt(Tmetal))*x*(sqrt(Ac/N))*(sqrt((1.1*rho*C_total())/
      (beta/fc)));
  }
  else
  {
   y=(1.0+Spwire)*F;
  }
 }
```

```
 else
 {
  /* calculates pitch without using repeaters, using RLC models*/
  if(flag2==1)
  {
   y=((1.0+Spwire)/sqrt(Tmetal))*sqrt(1.1*rho*C_total()*co*x*sqrt(Ac/N)/sqrt
     (er));
   /* RLC time delay, approximated to Time of Flight */
  }
  else
  {
   y=(1.0+Spwire)*F;
  }
 }
 return y;
}


float pitch_repeaters(float x)
{
 /* Calculates pitch with repeaters */
 float y,zeta;
 zeta=zeta_inpitchrepeaters;
 if(flag_repeater_pitchis2F == 1)
 y=(1.0+Spwire)*F;
 else
 y=((1.0+Spwire)/sqrt(Tmetal))*x*(sqrt(Ac/N))*(1.4+(0.53*zeta)+(0.53/zeta))
   *sqrt(rho*C_total()*Ro()*Co())/(beta/fc);
 return y;
}


float pitch_RLCrepeaters(float x)
{
 /* Calculates pitch with repeaters using RLC models for optimum
    repeaters*/
 float y;
 if(flag_repeater_pitchis2F == 1)
  y=(1.0+Spwire)*F;
 else
  y=(1.0+Spwire)*sqrt((1.496*Ro()*Co()*rho*x*(sqrt(Ac/N))*co*C_total())/(Tmetal
    *sqrt(er)*((beta/fc)-(x*(sqrt(Ac/N))*sqrt(er)/co))));
 return y;
}


float Crosstalk_Redesign(float Length1, float Length2, float zeta)
{
 /* CALLS APPROPRIATE FUNCTIONS TO REDESIGN L2 TO MEET CROSSTALK
    CONSTRAINTS. RETURNS NEW VALUE OF L2*/
 float pitchtemp;
```

```
if(globalflag_includeinductance==0)
{
 if(flag_start_optrep_design==0)
 {
  if (Length2 <= sqrt(N)) pitchtemp=pitch(Length2);
  else pitchtemp=pitch(sqrt(N));
 }
 else pitchtemp=pitch_repeaters(Length2);
}
else
{
 if(flag_start_optrep_design==0)
 {
  if (Length2 <= sqrt(N)) pitchtemp=pitch(Length2);
  else pitchtemp=pitch(sqrt(N));
 }
 else
 {
  if(flag_useRLCrepeaters==0)
  pitchtemp=pitch_repeaters(Length2);
  else pitchtemp=pitch_RLCrepeaters(Length2);
 }
}
WIDTH=pitchtemp/(1.0+Spwire);
while(Crosstalk_OK(Length1, Length2, zeta) == 0)
{
 Spwire=Spwire+delta_Spwire;
 if(globalflag_includeinductance==0)
 {
  if(flag_start_optrep_design==0)
  {
   Length2=L2_calculation(Length1, norepeaters);
   if (Length2 <= sqrt(N)) pitchtemp=pitch(Length2);
   else pitchtemp=pitch(sqrt(N));
  }
  else
  {
   Length2=Repeater_redesign(Length1,zeta);
   pitchtemp=pitch_repeaters(Length2);
  }
 }
 else
 {
  if(flag_start_optrep_design==0)
  {
   Length2=L2_calculation(Length1, norepeaters);
   if (Length2 <= sqrt(N)) pitchtemp=pitch(Length2);
   else pitchtemp=pitch(sqrt(N));
  }
  else
  {
```

```
    if(flag_useRLCrepeaters==0)
    {
     Length2=Repeater_redesign(Length1,zeta);
     pitchtemp=pitch_repeaters(Length2);
    }
    else
    {
     Length2=RLC_Repeater_redesign(Length1);
     pitchtemp=pitch_RLCrepeaters(Length2);
    }
   }
  }
  WIDTH=pitchtemp/(1.0+Spwire);
 }
 return Length2;
}


int Crosstalk_OK(float Length1, float Length2, float zeta)
{
 /* CHECKS IF CROSSTALK IS SATISFIED FOR GIVEN VALUES OF L1 AND L2.
    RETURNS 1 [TRUE] OR 0 [FALSE] */
 float lmax, Length, width, Crosstalk;
 float k_count, h_size;
 int dummy;
 width=WIDTH;
 lmax=zbrent(&lmax_calculation,0.001,10.0, 0.001);
 Length1=Length1*sqrt(Ac/N);
 Length2=Length2*sqrt(Ac/N);
 if(globalflag_worstcaseCrosstalk==1)
 {
  if(globalflag_includeinductance==1)
   Crosstalk=RLC_worstcaseCrosstalk();
  else
   Crosstalk=RC_worstcaseCrosstalk();
 }
 else
 {
   /*following loop is for no repeaters (RC and RLC), and RC repeaters
   cases -Ragu 09/02/02*/
  if(globalflag_includeinductance==0)
  {
   if(flag_start_optrep_design==0)
   {
    k_count=1.0;
    h_size=wk_logic;
    Length=Length2;
    //if(flag_useRLC==0)
    //if(globalflag_includeinductance==0)
    ///*No repeaters, RC case*/
    Length=Length2;
```

```
  Crosstalk=RC_Crosstalk(Length,k_count,h_size);
 }
 else
 {
  ///*With repeaters, RC case*/
  k_count=zeta*sqrt((0.4*rho*C_total())/(0.7*Ro()*Co()*width*Tmetal*width));
  //k_count is RC number of repeaters for unit length of interconnect
  h_size = sqrt(Ro()*C_total()*width*Tmetal*width/(rho*Co()));
  Length2=1/k_count;
  if(Length1 >= zeta/k_count) Length1=1/k_count;
  Length=Length2;
  Crosstalk=RC_Crosstalk(Length,k_count,h_size);
  ///*following is a redudant condition, used as a "safety net" */
  if(Crosstalk>(0.5*Cm_calculation()/(Cm_calculation()+Cgnd_calculation()));
  Crosstalk=0.5*Cm_calculation()/(Cm_calculation()+Cgnd_calculation());
 }
}
else
{
 if(flag_start_optrep_design==0)
 {
  ///*No repeaters, RLC case*/
  k_count=1.0;
  h_size=wk_logic;
  if(Length2 <= lmax)
  Length=Length2;
  else if(Length1<= lmax)
  Length=lmax;
  else Length=Length1;
  Crosstalk=RLC_Crosstalk(Length,k_count,h_size);
 }
 else
 {
  if(flag_useRLCrepeaters==0)
  {
   ///*With repeaters, RC case*/
   k_count=zeta*sqrt((0.4*rho*C_total())/(0.7*Ro()*Co()*width*Tmetal*width));
   //k_count is RC number of repeaters for unit length of interconnect
   h_size = sqrt(Ro()*C_total()*width*Tmetal*width/(rho*Co()));
   Length2=1/k_count;
   if(Length1 >= zeta/k_count) Length1=1/k_count;
   if(Length2 <= lmax)
    Length=Length2;
   else if(Length1<= lmax)
    Length=lmax;
   else Length=Length1;
   Crosstalk=RLC_Crosstalk(Length,k_count,h_size);
  }
  else
  {
   ///*following loop is for RLC repeaters - Ragu 09/02/02*/
```

```
      k_count=(0.9513*rho*co*C_total()/(sqrt(er)*width*Tmetal*width));
      ///*k_count is RLC number of repeaters for unit length of interconnect*/
      h_size=1.1479*Ro()*co*C_total()/sqrt(er);
      if(Length2 >= (1/k_count) )
      {
       Length2=1/k_count;
       Length1=1/k_count;
      }
      if(Length2 <= lmax)
       Length=Length2;
      else if(Length1<= lmax)
       Length=lmax;
      else Length=Length1;
      Crosstalk=RLC_Crosstalk(Length,k_count,h_size);
     }
    }
   }
  }
  if(Crosstalk <= Max_Crosstalk)
   return 1;
  else
  return 0;
}


double RLC_Crosstalk(float Length, float k_count, float h_size)
{
 /* CALCULATES CROSSTALK IN RLC INTERCONNECTS */
 double Rtr, Cload, Rint;
 double Rratio, Rratioplus, Rratiominus;
 double RT, RTplus, RTminus;
 double Cratio;
 double V_peak, V_peakRC, V_peakRLC;
 float lmax, L;
 // L=Length/k_count;
 /*Not required because length is already equal to one interconnect
 segment*/
 L=Length;
 Rtr=Ro()/h_size;
 Cload=h_size*Co();
 Rint=rho/(WIDTH*Tmetal*WIDTH);
 Rratioplus=Rint*L/Zoplus;
 Rratiominus=Rint*L/Zominus;
 Rratio=Rint*L/Zo;
 RTplus=Rtr/Zoplus;
 RTminus=Rtr/Zominus;
 RT=Rtr/Zo;
 Cratio=Cload/(C_total()*L);
 lmax=zbrent(&lmax_calculation,0.001,10.0, 0.001);
 V_peakRC=-0.58*((1+(RT/Rratio)+Cratio)/((3.1416/4)+(RT/Rratio)+Cratio))*(exp
  ((-1.04*1.0)/((RT*Cratio/Rratio)+Cratio+(RT/Rratio)+0.40528))-exp((-1.04*
```

```
  1.0)/(((RT*Cratio/Rratio)+Cratio+(RT/Rratio)+0.40528)*(1+2*(Cm_calculation()/
  Cgnd_calculation()))))-(0.05*(Cratio)*((Rratio)+(0.65*RT)+(0.36)));
 V_peakRLC=exp(-Rratioplus/(2))/(1+(RTplus))-exp(-Rratiominus/(2))/(1+(RTminus
  )) -(0.5*Cratio*(Rratio+0.65*RT+0.36));
 if ((V_peakRLC <= V_peakRC)  && (lmax < L))
 {
  V_peak=V_peakRC;
 }
 else
 {
  V_peak=V_peakRLC;
 }
 if(V_peak < 0)
  // V_peak=RC_worstcaseCrosstalk();
  V_peak=0;
 return V_peak;
}


double RC_Crosstalk(float Length, float k_count, float h_size)
{
 /* CALCULATES CROSSTALK IN RC INTERCONNECTS */
 double Rtr, Cload, Rint, K1;
 double Rratio, Rratioplus, Rratiominus;
 double RT, RTplus, RTminus;
 double Cratio;
 double V_peak, V_peakRC, V_peakRLC;
 float lmax, L;
 // L=Length/k_count;
 /*Not required because length is already equal to one interconnect segment*/
 L=Length;
 Rtr=Ro()/h_size;
 Cload=h_size*Co();
 Rint=rho/(WIDTH*Tmetal*WIDTH);
 Rratioplus=Rint*L/Zoplus;
 Rratiominus=Rint*L/Zominus;
 Rratio=Rint*L/Zo;
 RTplus=Rtr/Zoplus;
 RTminus=Rtr/Zominus;
 RT=Rtr/Zo;
 Cratio=Cload/(C_total()*L);
 V_peakRC=-0.58*((1+(RT/Rratio)+Cratio)/((3.1416/4)+(RT/Rratio)+Cratio))*(exp
  ((-1.04*1.0)/((RT*Cratio/Rratio)+Cratio+(RT/Rratio)+0.40528))-exp((-1.04*
  1.0)/(((RT*Cratio/Rratio)+Cratio+(RT/Rratio)+0.40528)*(1+2*(Cm_calculation()
  /Cgnd_calculation()))))) -(0.05*(Cratio)*((Rratio)+(0.65*RT)+(0.36)));
 V_peak=V_peakRC;
 return V_peak;
}


double RC_worstcaseCrosstalk(void)
```

```
{
 /* CALCULATES WORST CASE RC CROSSTALK */
 double temp;
 temp=0.5*Cm_calculation()/(Cm_calculation()+Cgnd_calculation());
 return temp;
}


double RLC_worstcaseCrosstalk(void)
{
 /* CALCULATES WORST CASE RLC CROSSTALK */
 double temp;
 temp=(3.1415/4)*Cm_calculation()/(Cm_calculation()+Cgnd_calculation());
 return temp;
}


float lmax_calculation(float x)
{
 /* CALCULATES LMAX FOR USE IN COMPUTATION OF RLC CROSSTALK */
 float y;
 double Rtr, Cload, S;
 double Rint, cratio, RT, RTplus, RTminus;
 double rratioplus, rratiominus, rratio;
 Ctalk_global_parameter_calculation(); /*This calculates Zo, Zoplus, and
 Zominus*/
 Rint=rho/(WIDTH*Tmetal*WIDTH); /*This is per unit length*/
 Cload=Co();
 cratio=Cload/(C_total()*1.0); /*This is per unit length*/
 rratioplus=Rint*x/Zoplus;
 rratiominus=Rint*x/Zominus;
 rratio=Rint*x/Zo;
 cratio=cratio/x;
 Rtr=Ro();
 RTplus=Rtr/Zoplus;
 RTminus=Rtr/Zominus;
 RT=Rtr/Zo;
 /*if Cload is constant and Cratio varies with L use the following formula
 --- most general case*/
 /* y=(1/(1+RTplus))*(-rratioplus/2)*exp(-rratioplus/2) + (1/(1+RTminus))*
      (rratiominus/2)*exp(-rratiominus/2) + S*cratio*(0.65*RT + 0.36);*/
 /*if Cratio is constant for changing L, ie. Cload changes with L, then
 use the following formula -- for our VLSI plots  Ragu 10/26/01*/
 y=(1/(1+RTplus))*(-rratioplus/2)*exp(-rratioplus/2) + (1/(1+RTminus))*
   (rratiominus/2)*exp(-rratiominus/2) - 0.5*(cratio*x)*(rratio);
 return y;
}


void Ctalk_global_parameter_calculation()
{
```

```
 /*This calculates Zo, Zoplus, and Zominus*/
 double ls, lm, Cintplus, Cintminus, Lintplus, Lintminus;
 ls=(er/(co*co))*(Cgnd_calculation()+Cm_calculation())/(Cgnd_calculation()
    *(Cgnd_calculation()+(2*Cm_calculation())));
 lm=(er/(co*co))*Cm_calculation()/(Cgnd_calculation()*(Cgnd_calculation()
    +(2*Cm_calculation())));
 Cintplus=Cgnd_calculation();
 Cintminus=Cgnd_calculation()+(2*Cm_calculation());
 Lintplus=ls+lm;
 Lintminus=ls-lm;
 Zoplus=sqrt(Lintplus/Cintplus);
 Zominus=sqrt(Lintminus/Cintminus);
 Zo=sqrt(ls/(Cm_calculation()+Cgnd_calculation()));
}


float Ro(void)
{
 /* Calculates minimum feature size device resistance*/
 float mun,Cox,Ro;
 mun=600.0;
 Cox=3.9*eo/tox;
 // Ro=1/(mun*Cox*(Vdd-Vtn));
 Ro=Vdd/(2.3*Ido_satdraincurrent());
 // printf("Ro=%g\n", Ro);
 return Ro;
}


float Co(void)
{
 /* Calculates minimum feature size device capacitance*/
 float beta_fanin_one,Cox,Co;
 beta_fanin_one=2;
 /***beta_fanin_one=mun/mup  This fixes the ratio of mobilities of minimum
 feature size nfet and pfet in an inverter to 2*/
 Cox=3.9*eo/tox;
 Co=pow(F,2)*Cox*(1 + beta_fanin_one);
 // printf("Co=%g\n", Co);
 return Co;
}


float L2_equation1(float x)
{
 /*SUPPLY-DEMAND EQUATION FOR L2 CALCULATION, WITHOUT REPEATERS, FOR
   BOTH L1 AND L2 IN SAME REGION OF DAVIS DISTRIBUTION */
 float y;
 extern float L1;
 float pitch(float x);
 if (L1 < sqrt(N))
```

```
 y=kai*pitch(x)*sqrt(Ac/N)*integral_region1(L1,x) - (2*eff*Ac);
 else
 y=kai*pitch(sqrt(N))*sqrt(Ac/N)*integral_region2(L1,x) - (2*eff*Ac);
 return y;
}


float L2_equation2(float x)
{
 /*SUPPLY-DEMAND EQUATION FOR L2 CALCULATION, WITHOUT REPEATERS, FOR
   L1 AND L2 IN DIFFERENT REGIONS OF DAVIS DISTRIBUTION */
 float y;
 extern float L1;
 float pitch(float x);
 y=kai*pitch(sqrt(N))*sqrt(Ac/N)*( integral_region1(L1,sqrt(N)) +
   integral_region2(sqrt(N),x))  -(2*eff*Ac);
 return y;
}


float L2_equationrepeaters1(float x)
{
 /*SUPPLY-DEMAND EQUATION FOR L2 CALCULATION, WITH REPEATERS, FOR
   BOTH L1 AND L2 IN SAME REGION OF DAVIS DISTRIBUTION */
 float y;
 extern float L1;
 float pitch_repeaters(float x);
 if(flag_useRLCrepeaters == 0)
 {
  if (L1 < sqrt(N))
   y=kai*pitch_repeaters(x)*sqrt(Ac/N)*integral_region1(L1,x)-(2*eff*Ac);
  else
   y=kai*pitch_repeaters(sqrt(N))*sqrt(Ac/N)*integral_region2(L1,x)-(2*eff*Ac);
 }
 else
 {
  if (L1 < sqrt(N))
   y=kai*pitch_RLCrepeaters(x)*sqrt(Ac/N)*integral_region1(L1,x)-(2*eff*Ac);
  else
   y=kai*pitch_RLCrepeaters(sqrt(N))*sqrt(Ac/N)*integral_region2(L1,x)
     -(2*eff*Ac);
 }
 return y;
}


float L2_equationrepeaters2(float x)
{
 /*SUPPLY-DEMAND EQUATION FOR L2 CALCULATION, WITHOUT REPEATERS, FOR
   L1 AND L2 IN DIFFERENT REGIONS OF DAVIS DISTRIBUTION */
 float y;
```

```
 extern float L1;
 float pitch_repeaters(float x);
 if(flag_useRLCrepeaters==0)
  y=kai*pitch_repeaters(sqrt(N))*sqrt(Ac/N)*(integral_region1(L1,sqrt(N))
    + integral_region2(sqrt(N),x))  -(2*eff*Ac);
 else
  y=kai*pitch_RLCrepeaters(sqrt(N))*sqrt(Ac/N)*(integral_region1(L1,sqrt(N))
   + integral_region2(sqrt(N),x))  -(2*eff*Ac);
 return y;
}


float demand_function(float L1temp, float L2temp)
{
 /* INTERCONNECT DEMAND FUNCTION FROM DAVIS DISTRIBUTION */
 float y;
 if (L2temp < sqrt(N))
  y=integral_region1(L1temp, L2temp);
 else if ((L1temp < sqrt(N)) && (L2temp>sqrt(N)))
  y=integral_region1(L1temp, sqrt(N)) + integral_region2(sqrt(N),L2temp);
 else if ((L1temp > sqrt(N)) && (L2temp>sqrt(N)))
  y=integral_region2(L1temp, L2temp);
 return y;
}


float integral_region1(float Llower, float x)
{
 /* INTEGRAL OF REGION 1 OF DAVIS DISTRIBUTION */
 float y;
 y=((alpha*k)/2)*Gamma*((1/((6*rents)+3))*(pow(x,(1+(2*rents)))-pow(Llower,
   (1+(2*rents))))-(sqrt(N)/rents)*(pow(x,(2*rents))-pow(Llower,(2*rents)))
   +(2*N/((2*rents)-1))*(pow(x,((2*rents)-1))-pow(Llower,((2*rents)-1))));
 return y;
}


float integral_region2(float Llower, float Lupper)
{
 /* INTEGRAL OF REGION 2 OF DAVIS DISTRIBUTION */
 float y;
 y=((alpha*k)/6)*Gamma*((1/((2*rents)-2))*(8*pow(N,1.5))*(pow(Lupper,(-2+
  (2*rents)))-pow(Llower,(-2+(2*rents))))-(12*N/(2*rents-1))*(pow(Lupper,(2*
  rents-1))-pow(Llower,(2*rents-1)))+(6*sqrt(N)/(2*rents))*(pow(Lupper,(2*
  rents))-pow(Llower,(2*rents)))-(1/(2*rents+1))*(pow(Lupper,2*rents+1)-pow(
  Llower,2*rents+1)));
 return y;
}

float m_calculation(float I1)
{
```

```
 /*Returns the occupation fraction of the topmost tier, for the no
 repeaters n-tier case*/
 float m;
 m=(1/(2*eff*Ac))*kai*(pitch(sqrt(N)))*(sqrt(Ac/N))*(I1);
 return m;
}


float mrep_calculation(float I1rep)
{
 /*Returns the occupation fraction of the topmost tier, for  with
 repeaters n-tier case*/
 float m;
 zeta_inpitchrepeaters=zetatemp;
 m=(1/(2*eff*Ac))*kai*(pitch_repeaters(sqrt(N)))*(sqrt(Ac/N))*(I1rep);
 return m;
}


float I1_calculation(float L1temp)
{
 float y;
 if (L1temp < sqrt(N) )
  y=integral_region1(L1temp,sqrt(N)) + integral_region2(sqrt(N),2*sqrt(N));
 else
  y=integral_region2(L1temp, 2*sqrt(N));
 return y;
}


/*******************Keith's Code begins*************************/
/****The following code was written by Keith Bowman******/

void keith()
{
  printf("Gate Area per feature size = %.5f\n", ( GateArea( 2.0, 3.0,1.0 )
  / pow( F, 2.0 ) ) );
}


/*****************************************************************************/
/* Function to calculate the area of a CMOS gate in [cm^2] by determining the
   NFET and PFET widths required to satisfy a specified clock frequency for a
   number of critical path logic gates.  Variable fin represents the number of
   inputs into the gate and fout represents the average number of output
   gates. */

double GateArea( double fin, double fout, float bakoglu_scaling_factor ){

//Define these variables
//Assign values for all of these parameters:
```

```
//F=feature size
//fc=clock frequency
//Cw=wiring capacitance
//tox=oxide thickness
//Vdd=supply voltage
//Vtn=threshold voltage
//eo
//
//Global variable:
//Na=doping concentration

  /* Local variable */
  double esi, eox, ni, q, Vsub, eta, phi, vnorm, vsatn;
  double muTn, muNn, mu0n;
  double Cd0, Cox, L;
  double mueffn, mueffna, thetan, Ec0n, Ecan;
  double Vd0n, Vdan, alphan;
  double  b, fineff;
  double kI, Gar, betaA;
  double Eqn;

  /* Permittivity of silicon [F/cm]. */
  esi = 11.8 * eo;
  /* Permittivity of oxide [F/cm]. */
  eox = 3.9 * eo;
  /* Intrinsic carrier concentration. */
  ni = 1.5e10;
  /* Electronic charge [C]. */
  q = 1.60218e-19;
  /* Substrate bias is equal to zero. */
  Vsub = 0.0;

  /* Area of a minimum sized inverter [F^2]. */
  kI = 102.0;
  /* Inverter aspect ratio. */
  Gar = ( 17.0 / 6.0 );
  /* Clock skew factor. */
  b = 0.9;

  /*Cw WIRING CAPACITANCE IS CALCULATED HERE - MODIFIED BY RAGU */
  Cw=Cw_calculation(fout);
  /* Check the fin value. */
  if ( fin == 1.0 )
    /* Effective fan-in for an inverter. */
    fineff = 1.0;
  else
    /* Effective fan-in due to the delay of two or more series connected
       NFETs. */
    fineff = 1.5 * ( fin / 2.0 );

  /* PFET to NFET width ratio assumes the NFET mobility is twice the PFET
```

```
   mobility. */
betaA = 2.0 / fin;

/* Effective channel length */
L = 0.8 * F;
/* Oxide capacitance. */
Cox = eox / tox;

/* Empirical velocity parameter used for calculating theta. */
vnorm = 2.2e9;
/* NFET saturation velocity used for the critical field calculation. */
vsatn = 1.434e9 * pow( 300.0, -0.87 );

/* Doping concentration calculated by the long channel threshold voltage. */
Na = NaEqn( Vtn );
/* Mobility for negligible doping concentrations which is a function of
   temperature. */
muTn = 1360.0 * pow( ( 300.0 ) / 300.0, -2.42 );
/* Mobility for significant doping concentration which is not a function of
   temperature. */
muNn = ( 5.1e18 + 92.0 * pow( Na, 0.91 ) ) / ( 3.75e15 + pow( Na, 0.91 ) );
/* Mobility for the NFET calculated as a function of doping concentration and
   temperature. */
mu0n = muTn * muNn / ( muTn + muNn );

/* Difference in potential from the intrinsic fermi level and the bulk fermi
   level. */
phi = ( 0.0259 ) * log( Na / ni );
/* Depletion depth capacitance as a function of doping concentration. */
Cd0 = sqrt( q * esi * Na / ( 2.0 * ( 2.0 * phi - Vsub ) ) );
/* Eta as a function of doping concentration. */
eta = 1.0 + Cd0 / Cox;

/* Value used for high field mobility effects in the NFET. */
thetan = mu0n / ( 2.0 * tox * vnorm );

/* NFET effective mobility. */
mueffn = mu0n / ( 1.0 + thetan * ( Vdd - Vtn ) );
/* NFET effective mobility. */
mueffna = mu0n / ( 1.0 + thetan * ( ( Vdd - Vtn ) / 2.0 ) );


/* Critical field for the NFET with Vgs=Vdd. */
Ec0n = ( vsatn / mueffn );
/* Critical field for the NFET with Vgs=(Vdd+Vtn)/2. */
Ecan = ( vsatn / mueffna );

/* Saturation voltage with Vgs = Vdd. */
Vd0n = L * Ec0n * ( sqrt( 1.0 + 2.0 * ( Vdd - Vtn ) / ( L * Ec0n * eta ) )
- 1.0 );
/* Saturation voltage with Vgs=(Vdd+Vtn)/2 */
```

```
  Vdan = L * Ecan * ( sqrt( 1.0 + ( Vdd - Vtn ) / ( L * Ecan * eta ) ) - 1.0 );

  /* Parameter used in the drain current model. */
  alphan = ( 1.0 / log( 2.0 ) ) * log( 2.0 * ( VdOn * ( Vdd - Vtn -
  ( eta / 2.0 ) * VdOn ) ) / ( Vdan * ( Vdd - Vtn - eta * Vdan ) ) );

   /* NFET effective mobility including both vertical and lateral high
      field effects for Ec. */
     mueffn = muOn / ( ( 1.0 + thetan * ( Vdd - Vtn ) ) * ( 1.0 + VdOn /
              (EcOn * L  )  )  );



  /* Check the fin value. */
  if ( fin == 1.0 )
    /* NFET width for an inverter per unit F.(Put in BAKOGLU EQUATION HERE!) */
    { wk = bakoglu_scaling_factor;
    }
  else
    /* NFET width required to satisfy the performance constraint. The factor
      (L/F) is due to the effective channel length such that the width is per
      unit F. */
    wk = ( L / F ) * Cw / ( ( ( 8 * mueffn * Cox * VdOn * ( Vdd - Vtn - ( eta
         / 2.0 ) * VdOn ) ) / ( fc * Vdd * (ncp * fineff / b ) * ( 9 - (
         10 / ( alphan + 1 ) ) * ( ( Vdd - Vtn ) / Vdd ) ) ) ) - Cdevice(
         fin, fout, betaA ) );
// printf("WKKKK=%g\n",wk);

  /* Area of CMOS gate. */
  Eqn = ( kI * ( 1.0 + ( 4.0 * ( fin - 1.0 ) * sqrt( Gar / kI ) ) ) * ( 1.0 +
        ( ( 1.0 + betaA ) * ( wk - 1.0 ) ) / ( sqrt( kI * Gar ) ) ) ) *
        pow( F, 2.0 );

  /* Return the Normal Gaussian Probability for x */
  return(Eqn);
}


void Max_CP_freq( double fin, double fout){

/*  Gives the maximum clock frequency possible for the given number of
    critical path gates */

//Define these variables
//Assign values for all of these parameters:
//F=feature size
//fc=clock frequency
//Cw=wiring capacitance
//tox=oxide thickness
//Vdd=supply voltage
//Vtn=threshold voltage
```

```
//eo
//
//Global variable:
//Na=doping concentration

  /* Local variable */
  double esi, eox, ni, q, Vsub, eta, phi, vnorm, vsatn;
  double muTn, muNn, mu0n;
  double Cd0, Cox, L;
  double mueffn, mueffna, thetan, Ec0n, Ecan;
  double Vd0n, Vdan, alphan;
  double b, fineff;
  double kI, Gar, betaA;
  double Eqn;
  double wkmax;

  /* Permittivity of silicon [F/cm]. */
  esi = 11.8 * eo;
  /* Permittivity of oxide [F/cm]. */
  eox = 3.9 * eo;
  /* Intrinsic carrier concentration. */
  ni = 1.5e10;
  /* Electronic charge [C]. */
  q = 1.60218e-19;
  /* Substrate bias is equal to zero. */
  Vsub = 0.0;

  /* Area of a minimum sized inverter [F^2]. */
  kI = 102.0;
  /* Inverter aspect ratio. */
  Gar = ( 17.0 / 6.0 );
  /* Clock skew factor. */
  b = 0.9;

  /*Cw WIRING CAPACITANCE IS CALCULATED HERE - MODIFIED BY RAGU */
  Cw=Cw_calculation(fout);
  /* Check the fin value. */
  if ( fin == 1.0 )
    /* Effective fan-in for an inverter. */
    fineff = 1.0;
  else
     /* Effective fan-in due to the delay of two or more series connected
    NFETs. */
    fineff = 1.5 * ( fin / 2.0 );

  /* PFET to NFET width ratio assumes the NFET mobility is twice the PFET
     mobility. */
  betaA = 2.0 / fin;

  /* Effective channel length */
  L = 0.8 * F;
```

```
/* Oxide capacitance. */
Cox = eox / tox;

/* Empirical velocity parameter used for calculating theta. */
vnorm = 2.2e9;
/* NFET saturation velocity used for the critical field calculation. */
vsatn = 1.434e9 * pow( 300.0, -0.87 );

/* Doping concentration calculated by the long channel threshold voltage. */
Na = NaEqn( Vtn );

/* Mobility for negligible doping concentrations which is a function of
   temperature. */
muTn = 1360.0 * pow( ( 300.0 ) / 300.0, -2.42 );
/* Mobility for significant doping concentration which is not a function of
   temperature. */
muNn = ( 5.1e18 + 92.0 * pow( Na, 0.91 ) ) / ( 3.75e15 + pow( Na, 0.91 ) );
/* Mobility for the NFET calculated as a function of doping concentration and
   temperature. */
mu0n = muTn * muNn / ( muTn + muNn );

/* Difference in potential from the intrinsic fermi level and the bulk fermi
   level. */
phi = ( 0.0259 ) * log( Na / ni );
/* Depletion depth capacitance as a function of doping concentration. */
Cd0 = sqrt( q * esi * Na / ( 2.0 * ( 2.0 * phi - Vsub ) ) );
/* Eta as a function of doping concentration. */
eta = 1.0 + Cd0 / Cox;

/* Value used for high field mobility effects in the NFET. */
thetan = mu0n / ( 2.0 * tox * vnorm );

/* NFET effective mobility. */
mueffn = mu0n / ( 1.0 + thetan * ( Vdd - Vtn ) );
/* NFET effective mobility. */
mueffna = mu0n / ( 1.0 + thetan * ( ( Vdd - Vtn ) / 2.0 ) );

/* Critical field for the NFET with Vgs=Vdd. */
Ec0n = ( vsatn / mueffn );
/* Critical field for the NFET with Vgs=(Vdd+Vtn)/2. */
Ecan = ( vsatn / mueffna );

/* Saturation voltage with Vgs = Vdd. */
Vd0n = L * Ec0n * ( sqrt( 1.0 + 2.0 * ( Vdd - Vtn ) / ( L * Ec0n * eta
       ) ) - 1.0 );
/* Saturation voltage with Vgs=(Vdd+Vtn)/2 */
Vdan = L * Ecan * ( sqrt( 1.0 + ( Vdd - Vtn ) / ( L * Ecan * eta ) ) - 1.0 );

/* Parameter used in the drain current model. */
alphan = ( 1.0 / log( 2.0 ) ) * log( 2.0 * ( Vd0n * ( Vdd - Vtn - ( eta
         / 2.0 ) * Vd0n ) ) / ( Vdan * ( Vdd - Vtn - eta * Vdan ) ) );
```

```
   /* NFET effective mobility including both vertical and lateral high
      field effects for Ec. */
    mueffn = mu0n / ( ( 1.0 + thetan * ( Vdd - Vtn ) ) * ( 1.0 + Vd0n /
             (Ec0n * L  )  )  );

   Eqn = ( Ac - (size_of_repeaters_this_design/rep_eff) ) / N;
   printf("in max_cp_freq Arep=%f, Alogic=%g\n",size_of_repeaters_this_design,
   N*Eqn);
   wkmax =  (Eqn /  ( kI * ( 1.0 + ( 4.0 * ( fin - 1.0 ) * sqrt( Gar / kI
            ) ) ) * pow(F,2) ) - 1.0 ) / ( ( 1.0 + betaA ) / (sqrt( kI *
            Gar ) ) )  + 1.0 ;
   fcmax =  ( ( ( 8 * mueffn * Cox * Vd0n * ( Vdd - Vtn - ( eta / 2.0 ) *
            Vd0n ) ) / (Vdd*(ncp * fineff / b ) * ( 9 - ( 10 / (alphan + 1
            ) ) * ( ( Vdd - Vtn ) / Vdd ) ) ) ) )  /  (  ( ( 1/ wkmax)*
            (L/F) *Cw)  + Cdevice( fin, fout, betaA ) ) ;

  /* Return the Normal Gaussian Probability for x */

    printf("Maximum frequency for given number of critical path gates is
    %g\n", fcmax);

}


float Ido_satdraincurrent(){

/*Gives the saturated drain current for a minimum feature size nfet. It
is used to calculate the capacitance of a minimum  feature size inverter */

  /* Local variable */
  double esi, eox, ni, q, Vsub, eta, phi, vnorm, vsatn;
  double muTn, muNn, mu0n;
  double Cd0, Cox, L;
  double mueffn, mueffna, thetan, Ec0n, Ecan;
  double Vd0n, Vdan, alphan;
  double b, fineff=1.0, fin=1.0, fout=1.0;
  double kI, Gar, betaA;
  double Eqn;
  double fcmax;
  float Ido_draincurrent;

  /* Permittivity of silicon [F/cm]. */
  esi = 11.8 * eo;
  /* Permittivity of oxide [F/cm]. */
  eox = 3.9 * eo;
  /* Intrinsic carrier concentration. */
  ni = 1.5e10;
  /* Electronic charge [C]. */
  q = 1.60218e-19;
  /* Substrate bias is equal to zero. */
```

```
Vsub = 0.0;

/* Area of a minimum sized inverter [F^2]. */
kI = 102.0;
/* Inverter aspect ratio. */
Gar = ( 17.0 / 6.0 );
/* Clock skew factor. */
b = 0.9;

/*Cw WIRING CAPACITANCE IS CALCULATED HERE - MODIFIED BY RAGU */
Cw=Cw_calculation(fout);

/* PFET to NFET width ratio assumes the NFET mobility is twice the PFET
mobility. */
betaA = 2.0;

/* Effective channel length */
L = 0.8 * F;
/* Oxide capacitance. */
Cox = eox / tox;

/* Empirical velocity parameter used for calculating theta. */
vnorm = 2.2e9;
/* NFET saturation velocity used for the critical field calculation. */
vsatn = 1.434e9 * pow( 300.0, -0.87 );

/* Doping concentration calculated by the long channel threshold voltage. */
Na = NaEqn( Vtn );

/* Mobility for negligible doping concentrations which is a function of
   temperature. */
muTn = 1360.0 * pow( ( 300.0 ) / 300.0, -2.42 );
/* Mobility for significant doping concentration which is not a function of
   temperature. */
muNn = ( 5.1e18 + 92.0 * pow( Na, 0.91 ) ) / ( 3.75e15 + pow( Na, 0.91 ) );
/* Mobility for the NFET calculated as a function of doping concentration and
   temperature. */
muOn = muTn * muNn / ( muTn + muNn );

/* Difference in potential from the intrinsic fermi level and the bulk fermi
   level. */
phi = ( 0.0259 ) * log( Na / ni );
/* Depletion depth capacitance as a function of doping concentration. */
Cd0 = sqrt( q * esi * Na / ( 2.0 * ( 2.0 * phi - Vsub ) ) );
/* Eta as a function of doping concentration. */
eta = 1.0 + Cd0 / Cox;

/* Value used for high field mobility effects in the NFET. */
thetan = muOn / ( 2.0 * tox * vnorm );

/* NFET effective mobility. */
```

```
  mueffn = mu0n / ( 1.0 + thetan * ( Vdd - Vtn ) );
  /* NFET effective mobility. */
  mueffna = mu0n / ( 1.0 + thetan * ( ( Vdd - Vtn ) / 2.0 ) );

  /* Critical field for the NFET with Vgs=Vdd. */
  Ec0n = ( vsatn / mueffn );
  /* Critical field for the NFET with Vgs=(Vdd+Vtn)/2. */
  Ecan = ( vsatn / mueffna );

  /* Saturation voltage with Vgs = Vdd. */
  Vd0n = L * Ec0n * ( sqrt( 1.0 + 2.0 * ( Vdd - Vtn ) / ( L * Ec0n * eta
         ) ) - 1.0 );
  /* Saturation voltage with Vgs=(Vdd+Vtn)/2 */
  Vdan = L * Ecan * ( sqrt( 1.0 + ( Vdd - Vtn ) / ( L * Ecan * eta ) ) - 1.0 );

  /* Parameter used in the drain current model. */
  alphan = ( 1.0 / log( 2.0 ) ) * log( 2.0 * ( Vd0n * ( Vdd - Vtn - ( eta
           / 2.0 ) * Vd0n ) ) / ( Vdan * ( Vdd - Vtn - eta * Vdan ) ) );

   /* NFET effective mobility including both vertical and lateral high
      field effects for Ec. */
    mueffn = mu0n / ( ( 1.0 + thetan * ( Vdd - Vtn ) ) * ( 1.0 + Vd0n /
             (Ec0n * L  )  )  );

    Ido_draincurrent =    mueffn * Cox * Vd0n * ( Vdd - Vtn - ( eta / 2.0
     ) * Vd0n )  ;

  /* Return the Normal Gaussian Probability for x */

    return Ido_draincurrent;
}


/* Capacitance Equations: */
/*****************************************************************************/
/*  Function to calculate the overlap capacitance for a CMOS NAND
    gate.  Variable fin represents the number of inputs and betaA is the
    PFET to NFET width ratio.*/

double Cov( double fin, double betaA ){

  /* Local variables */
  double eox, Cox, Cg0, Eqn;

  /* Permittivity of oxide [F/cm]. */
  eox = 3.9 * eo;
  /* Oxide capacitance [F/cm^2]. */
  Cox = eox / tox;

  /* Input capacitance [F]. */
  Cg0 = pow( F, 2.0 ) * Cox;
```

```
  /* Overlap capacitance [F]. */
  Eqn = ( 1.0e-1 ) * Cg0 * ( 1 + fin * betaA );

  /* Return the overlap capacitance */
  return(Eqn);
}


/*****************************************************************************/
/* Function to calculate the bottom junction capacitance for a CMOS NAND
   gate.  Variable fin represents the number of inputs and betaA is the PFET to
   NFET width ratio. */

double Cjbottom( double fin, double betaA ){

  /* Local variables */
  double q, esi, Vsub, Nv, Nb, mjb, Lds, Vbib, Cjb0, Cjb, Eqn;

  /* Permittivity of silicon [F/cm]. */
  esi = 11.8 * eo;
  /* Electronic charge [C] */
  q = 1.60218e-19;
  /* Substrate bias is equal to zero. */
  Vsub = 0.0;
  /* Effective density of states in the valence band. */
  Nv = 1.04e19;

  /* Bottom junction doping concentration. */
  Nb = Na;

  /* Bottom junction gradient coefficient. */
  mjb = 0.5;

  /* Length of the drain and source is assumed equal to 4 feature lengths, 4F.
     */
  Lds = 4.0;

  /* Built-in-voltage for the drain-body bottom junction. */
  Vbib = 1.12 - ( 0.0259 ) * log( Nv / Nb );

  /* The p-n bottom junction depletion capacitance with zero bias on the drain.
     */
  Cjb0 = sqrt( ( q * esi * Nb ) / ( 2.0 * ( Vbib - Vsub ) ) );

  /* Bottom junction capacitance [F/cm2]. */
  Cjb = Cjb0 * ( ( Vbib - Vsub ) / ( mjb * Vdd ) ) * ( pow( 1.0 + Vdd / (
        Vbib - Vsub ), mjb ) - 1.0 );

  /* Bottom junction capacitance [F]. */
  Eqn = ( Cjb * Lds * pow( F, 2.0 ) ) * ( 1.0 + fin * betaA );
```

```
  /* Return the bottom junction capacitance */
  return(Eqn);
}


/*****************************************************************************/
/* Function to calculate the sidewall junction capacitance for a CMOS NAND
   gate.  Variable fin represents the number of inputs and betaA is the PFET to
   NFET width ratio. */

double Cjside_wall( double fin, double betaA ){

  /* Local variables */
  double q, esi, Vsub, Nv, Nc, yj, mjsw, Lds, Vbisw, Cjsw0, Cjsw, Eqn;

  /* Permittivity of silicon [F/cm]. */
  esi = 11.8 * eo;
  /* Electronic charge [C] */
  q = 1.60218e-19;
  /* Substrate bias is equal to zero. */
  Vsub = 0.0;
  /* Effective density of states in the valence band. */
  Nv = 1.04e19;

  /* Channel junciton doping concentration. */
  Nc = Na;

  /* Junction Depth. */
  yj = F / 5.0;

  /* Channel junction gradient coeffiicient. */
  mjsw = 0.5;

  /* Length of the drain and source is assumed equal to 4 feature lengths, 4F.
     */
  Lds = 4.0;

  /* Built-in-voltage for the drain-body side wall junction. */
  Vbisw = 1.12 - ( 0.0259 ) * log( Nv / Nc );

  /* The p-n side wall junction depletion capacitance with zero bias on the
     drain. */
  Cjsw0 = sqrt( ( q * esi * Nc ) / ( 2.0 * ( Vbisw - Vsub ) ) );

  /* Side wall junction capacitance [F / cm2]. */
  Cjsw = Cjsw0 * ( ( Vbisw - Vsub ) / ( mjsw * Vdd ) ) * ( pow( 1.0 + Vdd
         / ( Vbisw - Vsub ), mjsw ) - 1.0 );

  /* Side wall junction capacitance [F]. */
  Eqn = ( Cjsw * yj ) * 2.0 * F * ( ( 1.0 + Lds ) + fin * ( betaA + Lds ) );
```

```
  /* Return the side wall junction capacitance */
  return(Eqn);
}


/*****************************************************************************/
/* Function to calculate the gate capacitance of fanout gates.  Variable fout
   represents the average number of outputs and betaA is the PFET to NFET width
   ratio. */

double Cfout( double fout, double betaA ){

  /* Local variables */
  double eox, Cox, Cg0, Eqn;

  /* Permittivity of oxide [F/cm]. */
  eox = 3.9 * eo;
  /* Oxide capacitance [F/cm^2]. */
  Cox = eox / tox;

  /* Input capacitance [F]. */
  Cg0 = pow( F, 2.0 ) * Cox;

  /* Gate capacitance of fanout gates. */
  Eqn = ( 1.0 + betaA ) * fout * Cg0;

  /* Return the gate capacitance of fanout gates */
  return(Eqn);
}


/*****************************************************************************/
/* Function to calculate the device capacitance for a CMOS NAND gate.  Variable
   fin represents the number of inputs and betaA is the PFET to NFET width
   ratio. */

double Cdevice( double fin, double fout, double betaA ){

  /* Local variable */
  double Eqn;

  /* Device capacitance [F]. */
  Eqn = Cov( fin, betaA ) + Cjbottom( fin, betaA ) + Cjside_wall( fin,
        betaA ) + Cfout( fout, betaA );

  /* Return the device capacitance */
  return(Eqn);
}
```

```
/*************************************************************************/
/* Function to numerically calculate the bulk doping concentration for a target
   threshold voltage.  Variable Vtntarget represents the target threshold
   voltage. */

double NaEqn( double Vtntarget ){

  /* Local variables */
  double VtTolerance, NaA, NaB, VtDiff, VtB, VtC, Eqn;

  /* Tolerance set at 0.1% of 1 [mV]. */
  VtTolerance = 1e-6;

  /* Assign NaA with the smallest possible Na. */
  NaA = 1e14;
  /* Assign NaB with the largest possible Na. */
  NaB = 5e19;

  /* Initialize VtDiff. */
  VtDiff = 1.0;

  /* While statement to perform the Bisection method. */
  while ( VtTolerance < VtDiff ){

    /* Assign Na equal to NaB. */
    Na = NaB;

    /* Calculate Vt with Vt=VtB. */
    VtB = Vtlong();

    /* Assign Na by choosing the middle value between NaA and NaB. */
    Na = ( NaA + NaB ) / 2.0;

    /* Calculate Vt with the assigned Na. */
    VtC = Vtlong();

    /* Check the Na effect on Vtlong. */
    if ( ( ( Vtntarget - VtB ) * ( Vtntarget - VtC ) ) <= 0.0 )
    /* Na is at the low end. */
    NaA = Na;

    else

    /* Na is at the high end. */
    NaB = Na;

    /* Check the convergence of the Bisection method. */
    VtDiff = fabs( VtC - Vtntarget );

  /* End of while statement which concludes the Bisection method. */
  }
```

```
  /* Assign the bulk doping concentration. */
  Eqn = Na;

  /* Return the bulk doping concentration */
  return(Eqn);
}


/*****************************************************************************/
/* Function to calculate the long channel threshold voltage. */

double Vtlong( void ){

  /* Local variables */
  double esi, eox, ni, q, Qi, Vsub, Cox, Vg0, phi, Vfb, Eqn;

  /* Permittivity of silicon [F/cm]. */
  esi = 11.8 * eo;
  /* Permittivity of oxide [F/cm]. */
  eox = 3.9 * eo;
   /* Intrinsic carrier concentration. */
  ni = 1.5e10;
  /* Electronic charge [C] */
  q = 1.60218e-19;
  /* Oxide interface charge */
  Qi = 1.0e10;
  /* Substrate bias is equal to zero. */
  Vsub = 0.0;

  /* Oxide capacitance [F/cm^2]. */
  Cox = eox / tox;

  /* Band gap voltage at 300*K.  Divide by (q) and the units become [V]. */
  Vg0 = 1.12;

  /* Difference in potential from the intrinsic fermi level and the bulk
     fermi level. */
  phi = ( 0.0259 ) * log( Na / ni );

  /* Flatband Voltage equation */
  Vfb = -( Vg0 / 2.0 ) - phi - ( q * Qi / Cox );

  /* Threshold voltage equation. */
  Eqn = Vfb + 2.0 * phi + ( sqrt( 2.0 * q * Na * esi * ( 2.0 * phi - Vsub
        ) ) ) / Cox;

  /* Return Vtlong */
  return(Eqn);
}
```

```
/* EOF */
/*******************Keith's Code ends*************************/


/***The following code is from Numerical Recipes in C. It solves a
non-linear equation in one variable ***/

float zbrent(float (*func)(float), float x1, float x2, float tol)
{
 int iter;
 float a=x1, b=x2, c=x2, d, e, min1, min2;
 float fa=(*func)(a), fb=(*func)(b), fc, p, q, r, s, tol1, xm;
 if ((fa > 0.0 && fb > 0.0) || (fa < 0.0 && fb < 0.0))
 {
  /*printf("fa=%f, fb=%f\n", fa,fb);
  printf("root must be bracketed in zbrent\n");*/
 }
 fc=fb;
 for (iter=1; iter<=ITMAX; iter++)
 {
  if ((fb > 0.0 && fc > 0.0) || (fb < 0.0 && fc < 0.0))
  {
   c=a;
   fc=fa;
   e=d=b-a;
  }
  if (fabs(fc) < fabs(fb))
  {
   a=b;
   b=c;
   c=a;
   fa=fb;
   fb=fc;
   fc=fa;
  }
  tol1=2.0*EPS*fabs(b)+0.5*tol;
  xm=0.5*(c-b);
  if (fabs(xm)<=tol1 || fb == 0.0) return b;
  if (fabs(e) >= tol1 && fabs(fa) > fabs(fb))
  {
   s=fb/fa;
   if (a == c)
   {
    p=2.0*xm*s;
    q=1.0-s;
   }
   else
   {
    q=fa/fc;
    r=fb/fc;
    p=s*(2.0*xm*q*(q-r)-(b-a)*(r-1.0));
```

```
    q=(q-1.0)*(r-1.0)*(s-1.0);
   }
   if (p > 0.0) q = -q;
   p=fabs(p);
   min1=3.0*xm*q-fabs(tol1*q);
   min2=fabs(e*q);
   if (2.0*p < (min1 < min2 ? min1 : min2))
   {
    e=d;
    d=p/q;
   }
   else
   {
    d=xm;
    e=d;
   }
  }
  else
  {
   d=xm;
   e=d;
  }
  a=b;
  fa=fb;
  if (fabs(d) > tol1)
   b+=d;
  else
   b+=SIGN(tol1,xm);
  fb=(*func)(b);
 }
 printf("Maximum number of iterations exceeded in zbrent/n");
 return 0.0;
}
```

A.3 Sample input file

Min_Feature_Size	[cms]	100E-7
Starting_N	[# of gates]	11.3E6
Ending_N	[# of gates]	13.0E6
Incremental_N	[# of gates]	10E6
Starting_Ac	[cm^2]	0.98
Ending_Ac	[cm^2]	5.0
Incremental_Ac	[cm^2]	10.0
Starting_fc	[Hz]	1.56E9
Ending_fc	[Hz]	2.0E9
Incremental_fc	[Hz]	10.0E9
Levels_nlevels	[#]	100

Spacingratio_Spwire1	[S/W]	1.0
Spacingratio_Spwire2	[S/W]	2.0
Delta_Spwire	[S/W]	10.0
Thicknessratio_Tmetal1	[T/W]	1.0
Thicknessratio_Tmetal2	[T/W]	2.0
Delta_Tmetal	[T/W]	10.0
Heightratio_Hoxide1	[H/W]	1.0
Heightratio_Hoxide2	[H/W]	2.0
Delta_Hoxide	[H/W]	10.0
Constrained_No_Repeaters	[#]	1E10
Max_Crosstalk	[Vc/Vdd]	0.3
globalflag_includeinductance	[T=1.0/F=0.0]	0.0
globalflag_worstcaseCrosstalk	*[T=1.0/F=0.0]*	*0.0*

A.4 Output file

```
*****************************************************************************
Gamma=0.803740
AA Gate Area=276.837486, size_of_logic_gates = 0.312826
Gate Area per feature size = 276.83749
wk=6.341874, Size of logic gates is 0.312826, size of chip is 0.980000
Interconnects power: Cw=8.69032e-15, Lavg=13.4289,
power_interconnects=7.65965
Power-D of logic gates is 8.65302, of interconnects is 7.65965
L_critical=302.304260
beta=0.900000,  L1=93.101028, L2=734.309875,  L2cms=0.216249,
pitch=0.000026, actual_beta=0.881513,n=2.000000
m=0.998990, global pitch=0.000117
the number of levels is 6.998990
For N=11300000.000000, fc=1560000000.000000 and Ac=0.980000,
no.of levels=6.998990


***N-Tier Design Before Repeater Insertion***
L2[0][0]=1.000000,  pitch=0, UseRLC=0, zeta=0, #rep=0,powrep=0
W=0, Spwire=NaN, Tmetal=NaN, Hoxide=NaN
L2[1][0]=93.101028,  pitch=2e-05, UseRLC=0, zeta=0, #rep=0,powrep=0
W=1e-05, Spwire=1, Tmetal=1, Hoxide=1
L2[2][0]=734.309875,  pitch=2.56044e-05, UseRLC=0, zeta=0, #rep=0,powrep=0
W=1.28022e-05, Spwire=1, Tmetal=1, Hoxide=1
L2[3][0]=1758.728882,  pitch=6.13244e-05, UseRLC=0, zeta=0, #rep=0,powrep=0
W=3.06622e-05, Spwire=1, Tmetal=1, Hoxide=1
L2[4][0]=6723.094727,  pitch=0.000117212, UseRLC=0, zeta=0, #rep=0,powrep=0
W=5.86062e-05, Spwire=1, Tmetal=1, Hoxide=1

Tier 1 iteration aborted because it is already minimum feature size
```

```
***N-Tier Design After Repeater Insertion***
L2[0]=1.000000,  pitch=0, W=0, Spwire=NaN, Tmetal=NaN, Hoxide=NaN
nrep=0, RLCrep=0.000000
L2[1]=93.101028,  pitch=2e-05, W=1e-05, Spwire=1, Tmetal=1, Hoxide=1
nrep=0, RLCrep=0.000000
L2[2]=1288.107666,  pitch=2e-05, W=1e-05, Spwire=1, Tmetal=1, Hoxide=1
nrep=1.36508e+06, RLCrep=0.000000
L2[3]=6723.094727,  pitch=5.59451e-05, W=2.79726e-05, Spwire=1, Tmetal=1,
Hoxide=1 nrep=135524, RLCrep=0.000000
L2[4]=6723.094727,  pitch=5.59451e-05, W=2.79726e-05, Spwire=1, Tmetal=1,
Hoxide=1 nrep=83184.5, RLCrep=0.000000
in max_cp_freq Arep=0.110183, Alogic=0.796362
Maximum frequency for given number of critical path gates is 2.2934e+09

For N=11300000.000000, fc=1560000000.000000 and Ac=0.980000,
no.of levels=6.998990
the new total number of levels with constrained # of repeaters is 4.388412
wire limited area = 0.980000, transistor limited area =0.496464
logic  area = 0.312826, rep area =0.110183
No of repeaters is 1.5006e+06,  Power_logic= 8.653018, Power_rep = 3.985439,
Power_int = 7.659645, Power_total= 20.298101
Program executed successfully
```

APPENDIX B

CODE FOR TRANSIENT RESPONSE OF DISTRIBUTED *RLC* LINES

B.1 Notes

The C code that implements the transient response of a single, distributed *rlc* interconnect, with a capacitive load termination is provided in Section B.2. A sample input file is given in Section B.3 and the corresponding output file is given in B.4. The main inputs that need to be specified are the resistance, capacitance and inductance per unit length of interconnect, driver output resistance, input capacitance of load, length of interconnect, and start, end and step times for the simulation.

B.2 Main code

```
/***PROGRAM THAT CALCULATES THE TRANSIENT RESPONSE OF A SINGLE DISTRIBUTED
RLC INTERCONNECT, WITH A CAPACITIVE LOAD TERMINATION ***/


#include <math.h>
#include <stdio.h>
#include <time.h>
#include <string.h>

#define ACC 40.0
```

```
#define BIGNO 1.0e10
#define BIGNI 1.0e-10

float  Rint, Cint, Lint, Rtr, Cload, Length;
float  starttimeT, endtimeT, deltatimeT;
double Zo, Gs, sigma, timeT, tf, B, P, Q, M, N;
double Rratio, Cratio, Tratio;

double bessi0(double);
double bessi1(double);
double bessi(int, double);
double besselfunction(int, double );
double Model1(double);
double Model2(double);
double Model3(double);
double factorial(double);
double Choose(double, double);
double Choose1(double, double);
double Choose2(double, double);
double factor_term1(double, double, double, double, double, double, double);
double factor_term3(double, double, double, double, double, double, double);
double factor_term1_model3(double, double, double, double, double);
double factor_term3_model3(double, double, double, double, double);
double inverselaplace(int, double, double, double);
int Coefficient(int, int);
void initialize_Coefficient(void);
int Call_Coefficient(int, int);

int Coeff[10,10];


main()
{
 int n;
 double timeT, Vc, V_peak_overshoot;
 double t1, t2;
 char junk[100];
 FILE *fp0, *fp1;
 fp0=fopen("input_singlelinetransient.dat","r");
 fp1=fopen("output_singlelinetransient.dat","w");
 printf("\n\n ************************************************************
 ************************\n\n");
 printf("Program to calculate step -input transient response of single,
 distributed rlc interconnect\n");
 printf("               Code for paper in proceedings DAC 2002, pp.763-766
 \n");
 printf("                        Author : Raguraman Venkatesan\n");
 printf("                         Gigascale Integration Group\n");
 printf("                    Georgia Institute of Technology, Atlanta\n");
 printf("                                 July 2002\n");
 printf("\n ************************************************************
```

```
**********************\n\n");

initialize_Coefficient();
fscanf(fp0, "%s %g", junk, &Rint);
fscanf(fp0, "%s %g", junk, &Cint);
fscanf(fp0, "%s %g", junk, &Lint);
fscanf(fp0, "%s %g", junk, &Rtr);
fscanf(fp0, "%s %g", junk, &Cload);
fscanf(fp0, "%s %g", junk, &Length);
fscanf(fp0, "%s %g", junk, &starttimeT);
fscanf(fp0, "%s %g", junk, &endtimeT);
fscanf(fp0, "%s %g", junk, &deltatimeT);

Zo=sqrt(Lint/Cint);
tf=length*sqrt(Lint*Cint);
Gs=(Rtr-Zo)/(Rtr+Zo);
Rratio=Rint*length/Zo;
Cratio=Cload/(Cint*length);
B=4/(Rratio*Cratio);
P=B*(B+sqrt(B*B + 4))/sqrt(B*B + 4);
Q=-B*(B-sqrt(B*B + 4))/sqrt(B*B + 4);
M=(B+sqrt(B*B+4.0))/2.0;
N=(B-sqrt(B*B+4.0))/2.0;
sigma=Rint/(2*Lint);
printf("* Zo=%g, sigma=%g, B=%g, Gs=%g, tf=%g\n", Zo,sigma,B,Gs, tf);
M=(B+sqrt(B*B+4.0))/2.0;
t1=(128*Lint/(M*Rint)) + tf;
t2=((100/11)*Cload*Zo) + 3*tf;
V_peak_overshoot=0;
for (timeT=starttimeT; timeT<=endtimeT; timeT=timeT+deltatimeT)
{
 Tratio=timeT/tf;
 printf("t1=%g, t2=%g, t=%g\n", t1, t2, timeT);
 if (t1 < t2)
 {
  if (timeT <= t1) Vc=Model1(timeT);
  else if (timeT <= t2) Vc=Model2(timeT);
  else Vc=Model3(timeT);
 }
 else
 {
  if (timeT <= t2) Vc=Model1(timeT);
  else Vc=Model3(timeT);
 }
 if (Vc > V_peak_overshoot) V_peak_overshoot=Vc;
 //if (Vc != -100) fprintf(fp1,"%g   %g\n", Tratio, Vc);
 if (Vc != -100)
 {
  fprintf(fp1,"%g   %g\n", timeT, Vc);
 }
 /*Vc = -100 if timeT is multiple of time of flight. This
```

```
  is to avoid singularities */
 }
 printf("\nProgram concluded. See output_singlelinetransient.dat for
 output data points\n");
 printf("\n\n ************************************************************
 **************************\n\n");
}


double bessi0(double x)
{
 double ax, ans;
 double y;
 if (( ax=fabs(x)) < 3.75)
 {
 y=x/3.75;
 y*=y;
 ans=1.0+y*(3.5156229+y*(3.0899424+y*(1.2067492+y*(0.2659732+y
    *(0.360768e-1+y*0.45813e-2)))));
 }
 else
 {
 y=3.75/ax;
 ans=(exp(ax)/sqrt(ax))*(0.39894228+y*(0.1328592e-1+y*
     (0.225319e-2+y*(-0.157565e-2+y*(0.916281e-2+y*(-0.2057706
     e-1+y*(0.2635537e-1+y*(-0.1647633e-1+y*0.392377e-2))))))));
 }
 return ans;
}


double bessi1(double x)
{
 double ax, ans;
 double y;
 if (( ax=fabs(x)) < 3.75)
 {
  y=x/3.75;
  y*=y;
  ans=ax*(0.5+y*(0.87890594+y*(0.51498869+y*(0.15084934+y
      *(0.2658733e-1+y*(0.301532e-2+y*0.32411e-3))))));
 }
 else
 {
 y=3.75/ax;
 ans=0.2282967e-1+y*(-0.2895312e-1+y*(0.1787654e-1-y*
     0.420059e-2));
 ans=0.39894228+y*(-0.3988024e-1+y*(-0.362018e-2+y*
     (0.163801e-2+y*(-0.1031555e-1+y*ans))));
 ans *= (exp(ax)/sqrt(ax));
 }
```

```
 return x < 0.0 ? -ans : ans;
}


double bessi(int n, double x)
{
 double bessi0(double x);
 int j;
 double bi, bim, bip, tox, ans;
 if (n < 2) printf("Index n less than 2 in bessi");
 if (x == 0.0) return 0.0;
 else
 {
  tox=2.0/fabs(x);
  bip=ans=0.0;
  bi=1.0;
  for (j=2*(n+(int) sqrt(ACC*n)); j>0;j--)
  {
   bim=bip+j*tox*bi;
   bip=bi;
   bi=bim;
   if (fabs(bi) > BIGNO)
   {
    ans *= BIGNI;
    bi *= BIGNI;
    bip *= BIGNI;
   }
   if (j==n) ans=bip;
  }
  ans *= bessi0(x)/bi;
  return x < 0.0 && (n & 1) ? -ans : ans;
 }
}


double Model1(double timeT)
{
 int g,i,j,k,l,m,n,p,q,r,u,order;
 double tolerance = 1E-2;
 double tolerance_k = 0.1;
 double timeTeff;
 double Vc=0,In,sum_old=0,sum1=0,sum2=0,sum2a=0,sum2b=0,sum2c=0,
  sum2d=0,term1=0,term2=0,term3=0,term4=0,term3a=0,term4a=0;
 double sumi=0, sumiold=0, sumj=0, sumjold=0, sumk=0, sumkold=0,
  suml=0, sumlold=0, summ=0, summold=0, sumn=0, sumnold=0, sump=0,
  sumpold=0, sumq=0, sumqold=0, sumr=0, sumrold=0, sumu=0,
  sumuold=0;
 double temp1, temp2,temp3,temp4;
 double Vinf;
 double a,c;
 double test;
```

```
double Vd=1.0;
double tp1, tp2, tp3, tp4, tp5, tp6, tp7;
int flag_tolerance_reached_for_i=0, flag_tolerance_reached_for_j=0,
flag_tolerance_reached_for_k=0, flag_tolerance_reached_for_l=0,
flag_tolerance_reached_for_p=0,flag_tolerance_reached_for_r=0,
flag_tolerance_reached_for_u=0;
//printf("Starting program\n");
timeTeff=timeT-tf;
//printf(" ********** MODEL 1 : time=%g, tf=%g\n", timeT, tf);
g=0.5*((timeT/tf) + 1.0)  - 1;
//printf("Zo=%g, sigma=%g, B=%g, Gs=%g, g=%d\n", Zo,sigma,B,Gs,g);
/* first we calculate the n=0 term ie. the infinite line
voltage*/
if( (Tratio - (int)Tratio) == 0) return -100;
else
{
 if(Tratio < 1.0)  sum1=0.0;
 else
 {
  /* the following is for getting the n=0 term ie. Vinf(L,s) */
  /*First we find term1*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term1=pow(-M,(double)i)*inverselaplace(-i-1,Tratio,1.0,Rratio);
   sumiold=sumi;
   sumi=sumi+term1;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term1=P*sumi;
  //printf("Term1 = %g\n",term1);

  /*Next we find term2*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term2=pow(-N,(double)(i))*inverselaplace(-i-1,Tratio,1.0,Rratio);
   sumiold=sumi;
   sumi=sumi+term2;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term2=Q*sumi;
  //printf("Term2 = %g\n",term2);

  /*Next we find term3*/
  j=0;sumj=0;sumjold=0;flag_tolerance_reached_for_j=0;
  while (flag_tolerance_reached_for_j==0)
```

```
 {
  j++;
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term3=pow(-M,(double)i)*(4-(pow(1+Gs,2.0)*pow(Gs,(double)(j-1))))*
         inverselaplace(-i-j-1,Tratio,1.0,Rratio);
   sumiold=sumi;
   sumi=sumi+term3;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  sumjold=sumj;
  sumj=sumj+sumi;
  if(fabs(sumj-sumjold) <= fabs(tolerance*sumj))
  flag_tolerance_reached_for_j=1;
 }
 term3=(P/(1-Gs))*sumj;
 //printf("Term3 = %g\n",term3);

 /*Next we find term4*/
 j=0;sumj=0;sumjold=0;flag_tolerance_reached_for_j=0;
 while (flag_tolerance_reached_for_j==0)
 {
  j++;
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term4=pow(-N,(double)(i))*(4-(pow(1+Gs,2.0)*pow(Gs,(double)(j-1))))
         *inverselaplace(-i-j-1,Tratio,1.0,Rratio);
   sumiold=sumi;
   sumi=sumi+term4;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  sumjold=sumj;
  sumj=sumj+sumi;
  if(fabs(sumj-sumjold) <= fabs(tolerance*sumj))
  flag_tolerance_reached_for_j=1;
 }
 term4=(Q/(1-Gs))*sumj;
 //printf("Term4 = %g\n",term4);
 /*****************************************************************/
 sum1=term1+term2+term3+term4;
 //printf("sum1 = %g\n",sum1);
}
sum1=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*sum1;
//printf("Voltage1=%g\n", sum1);
```

```
/*******************************************************************/
/* now we do the summation from n=1 to n=g where g is the number of
reflections*/
sumi=0,sumj=0,sumk=0,suml=0,summ=0,sumn=0,sump=0,sumq=0,sumr=0,sumu=0;
/* first we find sum2a=term1+term2*/
for(n=1; n<=g; n++)
{
 if(Tratio < (2*n+1))  sum2=0;
 else
 {
  sumi=0;
  for(i=0; i <=n; i ++)
  {
   summ=0;
   for(m=0; m <=n; m ++)
   {
    sumq=0;
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
     while(flag_tolerance_reached_for_r==0)
     {
      r=r+1;
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
       p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
       while(flag_tolerance_reached_for_p==0)
       {
        p=p+1;
        l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
        while(flag_tolerance_reached_for_l==0)
        {
         l=l+1;
         k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
         while(flag_tolerance_reached_for_k==0)
         {
          k=k+1;
          term1=pow(Gs,(double)(n+r-q))*(factor_term1((double)i,(double)j,
          (double)m,(double)n,(double)p,(double)q,(double)r))*pow(-1,(double)
          (n+i+m+p+k+q))*Choose((double)(-i),(double)k)*Choose((double)(-j),
          (double)l)*pow((4/(Rratio*Cratio)),(double)(k+l+m+p))*inverselaplace
          ((-2*i)-(2*j)-k-l-m-p-q-r,Tratio,(double)(2*n+1),Rratio);
          temp1=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term1);
          sumkold=sumk;
          sumk=sumk+temp1;
          sum2=sum2+temp1;
          if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
          flag_tolerance_reached_for_k=1;
         }
```

```
          sumlold=suml;
          suml=sum2;
          if( fabs(sumlold-suml) <= fabs(tolerance*suml))
          flag_tolerance_reached_for_l=1;
         }
         sumpold=sump;
         sump=sum2;
         if( fabs(sumpold-sump) <= fabs(tolerance*sump))
         flag_tolerance_reached_for_p=1;
        }
        sumjold=sumj;
        sumj=sum2;
        if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
        flag_tolerance_reached_for_j=1;
       }
       sumrold=sumr;
       sumr=sum2;
       if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
       flag_tolerance_reached_for_r=1;
      }
     }
    }
   }
  }
 }
sum2a=sum2;
//printf(" * sum2a=%g\n\n", sum2);
sum2=0;
/*************sum2b***************/
for(n=1; n<=g; n++)
{
 if(Tratio < (2*n+1))  sum2=0;
 else
 {
  sumi=0;
  for(i=0; i <=n; i ++)
  {
   summ=0;
   for(m=0; m <=n; m ++)
   {
    sumq=0;
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
     while(flag_tolerance_reached_for_r==0)
     {
      r=r+1;
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
```

```
 p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
 while(flag_tolerance_reached_for_p==0)
 {
  p=p+1;
  l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
  while(flag_tolerance_reached_for_l==0)
  {
   l=l+1;
   k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
   while(flag_tolerance_reached_for_k==0)
   {
    k=k+1;
    u=0;sumu=0;sumuold=0;flag_tolerance_reached_for_u=0;
    while (flag_tolerance_reached_for_u==0)
    {
     u=u+1;
     term2=(1/(1-Gs))*pow(Gs,(double)(n+r-q))*(factor_term1((double)i,
     (double)j,(double)m,(double)n,(double)p,(double)q,(double)r))*
     pow(-1,(double)(n+i+m+p+k+q))*Choose((double)(-i),(double)k)
     *Choose((double)(-j),(double)l)*pow((4/(Rratio*Cratio)),(double)
     (k+l+m+p))*(4-(pow(1+Gs,2.0)*pow(Gs,(double)(u-1))))*
     inverselaplace((-2*i)-(2*j)-k-l-m-p-q-r-u,Tratio,2*n+1,Rratio);
     temp2=(Vd*(Zo/(Zo+Rtr))*exp(-Rratio*Tratio/2.0))*(term2);
     sumuold=sumu;
     sumu=sumu+temp2;
     sum2=sum2+temp2;
     if (fabs(sumu-sumuold) <=fabs(tolerance*sumu))
     flag_tolerance_reached_for_u=1;
    }
    sumkold=sumk;
    sumk=sum2;
    if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
    flag_tolerance_reached_for_k=1;
   }
   sumlold=suml;
   suml=sum2;
   if( fabs(sumlold-suml) <= fabs(tolerance*suml))
   flag_tolerance_reached_for_l=1;
  }
  sumpold=sump;
  sump=sum2;
  if( fabs(sumpold-sump) <= fabs(tolerance*sump))
  flag_tolerance_reached_for_p=1;
 }
 sumjold=sumj;
 sumj=sum2;
 if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
 flag_tolerance_reached_for_j=1;
}
sumrold=sumr;
sumr=sum2;
```

```
      if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
      flag_tolerance_reached_for_r=1;
     }
    }
   }
  }
 }
}
sum2b=sum2;
//printf(" * sum2b=%g\n\n", sum2);
sum2=0;

/***************************************/
/* next we find sum2b=term3+term4*/
/*******************sum2c*********************/
for(n=1; n<=g; n++)
{
 if(Tratio <= (2*n+1))  sum2=0;
 else
 {
  for(i=0; i <=n+1; i ++)
  {
   for(m=0; m <=n+1; m ++)
   {
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
     while(flag_tolerance_reached_for_r==0)
     {
      r=r+1;
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
       p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
       while(flag_tolerance_reached_for_p==0)
       {
        p=p+1;
        l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
        while(flag_tolerance_reached_for_l==0)
        {
         l=l+1;
         k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
         while(flag_tolerance_reached_for_k==0)
         {
          k=k+1;
          term3=pow(Gs,(double)(n+r-q))*(factor_term3((double)i,(double)j,
          (double)m,(double)n,(double)p,(double)q,(double)r))*pow(-1,
          (double)(n+i+m+p+k+q+1))*Choose((double)(-i),(double)k)*Choose
          ((double)(-j),(double)l)*pow((4/(Rratio*Cratio)),(double)(k+l+m
          +p))*inverselaplace((-2*i)-(2*j)-k-l-m-p-q-r,Tratio,(double)
```

```
          (2*n+1),Rratio);
          temp3=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term3);
          sumkold=sumk;
          sumk=sumk+temp3;
          sum2=sum2+temp3;
          if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
          flag_tolerance_reached_for_k=1;
         }
         sumlold=suml;
         suml=sum2;
         if( fabs(sumlold-suml) <= fabs(tolerance*suml))
         flag_tolerance_reached_for_l=1;
        }
        sumpold=sump;
        sump=sum2;
        if( fabs(sumpold-sump) <= fabs(tolerance*sump))
        flag_tolerance_reached_for_p=1;
       }
       sumjold=sumj;
       sumj=sum2;
       if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
       flag_tolerance_reached_for_j=1;
      }
      sumrold=sumr;
      sumr=sum2;
      if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
      flag_tolerance_reached_for_r=1;
     }
    }
   }
  }
 }
}
sum2c=sum2;
//printf(" * sum2c=%g\n\n", sum2);
sum2=0;
/*************************************/
/*#######################################################*/
sumi=0,sumj=0,sumk=0,suml=0,summ=0,sumn=0,sump=0,sumq=0,sumr=0,sumu=0;
for(n=1; n<=g; n++)
{
 if(Tratio <= (2*n+1))  sum2=0;
 else
 {
  for(i=0; i <=n+1; i ++)
  {
   for(m=0; m <=n+1; m ++)
   {
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
```

```
while(flag_tolerance_reached_for_r==0)
{
 r=r+1;
 j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
 while(flag_tolerance_reached_for_j==0)
 {
  j=j+1;
  p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
  while(flag_tolerance_reached_for_p==0)
  {
   p=p+1;
   l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
   while(flag_tolerance_reached_for_l==0)
   {
    l=l+1;
    k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
    while(flag_tolerance_reached_for_k==0)
    {
     k=k+1;
     u=0;sumu=0;sumuold=0;flag_tolerance_reached_for_u=0;
     while (flag_tolerance_reached_for_u==0)
     {
      u=u+1;
      term4=(1/(1-Gs))*pow(Gs,(double)(n+r-q))*(factor_term3((double)i,
      (double)j,(double)m,(double)n,(double)p,(double)q,(double)r))*pow
      (-1,(double)(n+i+m+p+k+q+1))*Choose((double)(-i),(double)k)*
      Choose((double)(-j),(double)l)*pow((4/(Rratio*Cratio)),(double)
      (k+l+m+p))*(4-(pow(1+Gs,2.0)*pow(Gs,(double)(u-1))))*inverselaplace
      ((-2*i)-(2*j)-k-l-m-p-q-r-u,Tratio,(double)(2*n+1),Rratio);
      temp4=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term4);
      sumuold=sumu;
      sumu=sumu+temp4;
      sum2=sum2+temp4;
      if (fabs(sumu-sumuold) <=fabs(tolerance*sumu))
      flag_tolerance_reached_for_u=1;
     }
     sumkold=sumk;
     sumk=sum2;
     if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
     flag_tolerance_reached_for_k=1;
    }
    sumlold=suml;
    suml=sum2;
    if( fabs(sumlold-suml) <= fabs(tolerance*suml))
    flag_tolerance_reached_for_l=1;
   }
   sumpold=sump;
   sump=sum2;
   if( fabs(sumpold-sump) <= fabs(tolerance*sump))
   flag_tolerance_reached_for_p=1;
  }
```

```
          sumjold=sumj;
          sumj=sum2;
          if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
          flag_tolerance_reached_for_j=1;
         }
         sumrold=sumr;
         sumr=sum2;
         if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
         flag_tolerance_reached_for_r=1;
        }
       }
      }
     }
    }
   }
  }
  sum2d=sum2;
  //printf(" * sum2d=%g\n\n", sum2);
  sum2=sum2a+sum2b+sum2c+sum2d;
  /*#####################################################*/
  Vinf=sum1;
  Vc=(sum1+sum2);
  //printf("sum1 = %g, sum2 = %g\n", sum1, sum2);
  //printf("Voltage at the output is = %g\n\n", Vc);
  return Vc;
 }
}


/***************************Model 2 begins here************************/
double Model2(double timeT)
{
 int g,i,j,k,l,m,n,p,q,r,u,order;
 double tolerance = 1E-2;
 double tolerance_k = 0.1;
 double timeTeff;
 double Vc=0,In,sum_old=0,sum1=0,sum2=0,sum2a=0,sum2b=0,sum2c=0,sum2d=0,
 term1=0,term2=0,term3=0,term4=0,term3a=0,term4a=0;
 double sumi=0, sumiold=0, sumj=0, sumjold=0, sumk=0, sumkold=0, suml=0,
 sumlold=0, summ=0, summold=0, sumn=0, sumnold=0,sump=0, sumpold=0, sumq=0,
 sumqold=0, sumr=0, sumrold=0, sumu=0, sumuold=0;
 double temp1, temp2,temp3,temp4;
 double tp1,tp2,tp3,tp4,tp5;/*to be used for debugging only*/
 double Vinf;
 double a,c;
 double test;
 double Vd=1.0;
 int flag_tolerance_reached_for_i=0, flag_tolerance_reached_for_j=0,
 flag_tolerance_reached_for_k=0,flag_tolerance_reached_for_l=0,
 flag_tolerance_reached_for_p=0,flag_tolerance_reached_for_r=0,
 flag_tolerance_reached_for_u=0;
 Zo=sqrt(Lint/Cint);
```

```
timeTeff=timeT-tf;
//printf("*********** MODEL 2 : time=%g, tf=%g\n", timeT, tf);
g=0.5*((timeT/tf) + 1.0)  - 1;
//printf("Zo=%g, tf=%g, sigma=%g, B=%g, Gs=%g\n",Zo, tf, sigma, B, Gs);
/* first we calculate the n=0 term ie. the infinite line voltage*/
if( (timeT/tf - (int)(timeT/tf)) == 0) return -100;
else
{
 if(timeT < tf)  sum1=0.0;
 else
 {
  //printf("In this loop\n");
  /* the following is for getting the n=0 term ie. Vinf(L,s) */
  /*********************************************************************/
  /*First we find term1*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term1=pow(-1,(double)i)*pow(M,(double)(-i-1))*inverselaplace(i,Tratio,1.0,
   Rratio);
   sumiold=sumi;
   sumi=sumi+term1;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term1=P*sumi;
  //printf("Term1 = %g\n",term1);
  /*Next we find term2*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term2=pow(-N,(double)(i))*inverselaplace(-i-1,Tratio,1.0,Rratio);
   sumiold=sumi;
   sumi=sumi+term2;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term2=Q*sumi;
  //printf("Term2 = %g\n",term2);
  /*Next we find term3*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   j=0;sumj=0;sumjold=0;flag_tolerance_reached_for_j=0;
   while (flag_tolerance_reached_for_j==0)
   {
    j++;
    term3=pow(-1,(double)i)*pow(M,(double)(-i-1))*(4-pow(1+Gs,2.0)*pow(Gs,
```

```
    (double)(j-1)))*inverselaplace(i-j,Tratio,1.0,Rratio);
    sumjold=sumj;
    sumj=sumj+term3;
    if(fabs(sumj-sumjold) <= fabs(tolerance*sumj))
    flag_tolerance_reached_for_j=1;
   }
   sumiold=sumi;
   sumi=sumi+sumj;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term3=(P/(1-Gs))*sumi;
  //printf("Term3 = %g\n",term3);
  /*Next we find term4*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   j=0;sumj=0;sumjold=0;flag_tolerance_reached_for_j=0;
   while (flag_tolerance_reached_for_j==0)
   {
    j++;
    term4=pow(-N,(double)(i))*(4-pow(1+Gs,2.0)*pow(Gs,(double)(j-1)))*
    inverselaplace(-i-j-1,Tratio,1.0,Rratio);
    sumjold=sumj;
    sumj=sumj+term4;
    if(fabs(sumj-sumjold) <= fabs(tolerance*sumj))
    flag_tolerance_reached_for_j=1;
   }
   sumiold=sumi;
   sumi=sumi+sumj;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term4=(Q/(1-Gs))*sumi;
  //printf("Term4 = %g\n",term4);
  /*******************************************************************/
  sum1=term1+term2+term3+term4;
  //printf("sum1 = %g\n",sum1);
 }
 sum1=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*sum1;
 //printf("Voltage1=%g\n\n", sum1);
 /*******************************************************************/
 /* now we do the summation from n=1 to n=g where g is the number of
 reflections*/
 sumi=0,sumj=0,sumk=0,suml=0,summ=0,sumn=0,sump=0,sumq=0,sumr=0,sumu=0;
 /* first we find sum2a=term1+term2*/
 for(n=1; n<=g; n++)
 {
  if(timeT < ((2*n+1)*tf))  sum2=0;
  else
```

```
{
 sumi=0;
 for(i=0; i <=n; i ++)
 {
  summ=0;
  for(m=0; m <=n; m ++)
  {
   sumq=0;
   for(q=0; q <=n; q ++)
   {
    r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
    while(flag_tolerance_reached_for_r==0)
    {
     r=r+1;
     j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
     while(flag_tolerance_reached_for_j==0)
     {
      j=j+1;
      p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
      while(flag_tolerance_reached_for_p==0)
      {
       p=p+1;
       l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
       while(flag_tolerance_reached_for_l==0)
       {
        l=l+1;
        k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
        while(flag_tolerance_reached_for_k==0)
        {
         k=k+1;
         term1=pow(Gs,(double)(n+r-q))*(factor_term1((double)i,(double)j,
         (double)m,(double)n,(double)p,(double)q,(double)r))*pow(-1,
         (double)(n+i+m+p+k+q))*Choose((double)(-i),(double)k)*Choose
         ((double)(-j),(double)l)*pow((4/(Rratio*Cratio)),(double)(k+l
         +m+p))*inverselaplace((-2*i)-(2*j)-k-l-m-p-q-r,Tratio,(double)
         (2*n+1),Rratio);
         temp1=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term1);
         sumkold=sumk;
         sumk=sumk+temp1;
         sum2=sum2+temp1;
         if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
         flag_tolerance_reached_for_k=1;
        }
        sumlold=suml;
        suml=sum2;
        if( fabs(sumlold-suml) <= fabs(tolerance*suml))
        flag_tolerance_reached_for_l=1;
       }
       sumpold=sump;
       sump=sum2;
       if( fabs(sumpold-sump) <= fabs(tolerance*sump))
```

```
        flag_tolerance_reached_for_p=1;
       }
       sumjold=sumj;
       sumj=sum2;
       if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
       flag_tolerance_reached_for_j=1;
      }
      sumrold=sumr;
      sumr=sum2;
      if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
      flag_tolerance_reached_for_r=1;
     }
    }
   }
  }
 }
}
sum2a=sum2;
//printf(" * sum2a=%g\n\n", sum2);
sum2=0;
/*************sum2b***************/
for(n=1; n<=g; n++)
{
 if(timeT < ((2*n+1)*tf))  sum2=0;
 else
 {
  sumi=0;
  for(i=0; i <=n; i ++)
  {
   summ=0;
   for(m=0; m <=n; m ++)
   {
    sumq=0;
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
     while(flag_tolerance_reached_for_r==0)
     {
      r=r+1;
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
       p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
       while(flag_tolerance_reached_for_p==0)
       {
        p=p+1;
        l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
        while(flag_tolerance_reached_for_l==0)
        {
         l=l+1;
```

```
          k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
          while(flag_tolerance_reached_for_k==0)
          {
           k=k+1;
           u=0;sumu=0;sumuold=0;flag_tolerance_reached_for_u=0;
           while (flag_tolerance_reached_for_u==0)
           {
            u=u+1;
            term2=(1/(1-Gs))*pow(Gs,(double)(n+r-q))*(factor_term1((double)i,
            (double)j,(double)m,(double)n,(double)p,(double)q,(double)r))*
            pow(-1,(double)(n+i+m+p+k+q))*Choose((double)(-i),(double)k)*
            Choose((double)(-j),(double)l)*pow((4/(Rratio*Cratio)),(double)
            (k+l+m+p))*(4-(pow(1+Gs,2.0)*pow(Gs,(double)(u-1))))*
            inverselaplace((-2*i)-(2*j)-k-l-m-p-q-r-u,Tratio,2*n+1,Rratio);
            temp2=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term2);
            sumuold=sumu;
            sumu=sumu+temp2;
            sum2=sum2+temp2;
            if(fabs(sumu-sumuold) <=fabs(tolerance*sumu))
            flag_tolerance_reached_for_u=1;
           }
           sumkold=sumk;
           sumk=sum2;
           if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
           flag_tolerance_reached_for_k=1;
          }
          sumlold=suml;
          suml=sum2;
          if( fabs(sumlold-suml) <= fabs(tolerance*suml))
          flag_tolerance_reached_for_l=1;
         }
         sumpold=sump;
         sump=sum2;
         if( fabs(sumpold-sump) <= fabs(tolerance*sump))
         flag_tolerance_reached_for_p=1;
        }
        sumjold=sumj;
        sumj=sum2;
        if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
        flag_tolerance_reached_for_j=1;
       }
       sumrold=sumr;
       sumr=sum2;
       if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
       flag_tolerance_reached_for_r=1;
      }
     }
    }
   }
  }
 }
```

```
sum2b=sum2;
//printf(" * sum2b=%g\n\n", sum2);
sum2=0;
/*************************************/
/* next we find sum2b=term3+term4*/
/*****************sum2c*******************/
for(n=1; n<=g; n++)
{
 if(timeT <= ((2*n+1)*tf))  sum2=0;
 else
 {
  for(i=0; i <=n+1; i ++)
  {
   for(m=0; m <=n+1; m ++)
   {
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
     while(flag_tolerance_reached_for_r==0)
     {
      r=r+1;
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
       p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
       while(flag_tolerance_reached_for_p==0)
       {
        p=p+1;
        l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
        while(flag_tolerance_reached_for_l==0)
        {
         l=l+1;
         k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
         while(flag_tolerance_reached_for_k==0)
         {
          k=k+1;
          term3=pow(Gs,(double)(n+r-q))*(factor_term3((double)i,(double)j,
          (double)m,(double)n,(double)p,(double)q,(double)r))*pow(-1,(double)
          (n+i+m+p+k+q+1))*Choose((double)(-i),(double)k)*Choose((double)(-j
          ),(double)l)*pow((4/(Rratio*Cratio)),(double)(k+l+m+p))*
          inverselaplace((-2*i)-(2*j)-k-l-m-p-q-r,Tratio,(double)
          (2*n+1),Rratio);
          temp3=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term3);
          sumkold=sumk;
          sumk=sumk+temp3;
          sum2=sum2+temp3;
          if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
          flag_tolerance_reached_for_k=1;
         }
         sumlold
```

```
          suml;
          suml=sum2;
          if( fabs(sumlold-suml) <= fabs(tolerance*suml))
          flag_tolerance_reached_for_l=1;
         }
         sumpold=sump;
         sump=sum2;
         if( fabs(sumpold-sump) <= fabs(tolerance*sump))
         flag_tolerance_reached_for_p=1;
        }
        sumjold=sumj;
        sumj=sum2;
        if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
        flag_tolerance_reached_for_j=1;
       }
       sumrold=sumr;
       sumr=sum2;
       if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
       flag_tolerance_reached_for_r=1;
      }
     }
    }
   }
  }
 }
sum2c=sum2;
//printf(" * sum2c=%g\n\n", sum2);
sum2=0;
/*************************************/
/*#######################################################*/
sumi=0,sumj=0,sumk=0,suml=0,summ=0,sumn=0,sump=0,sumq=0,sumr=0,sumu=0;
for(n=1; n<=g; n++)
{
 if(timeT <= ((2*n+1)*tf))  sum2=0;
 else
 {
  for(i=0; i <=n+1; i ++)
  {
   for(m=0; m <=n+1; m ++)
   {
    for(q=0; q <=n; q ++)
    {
     r=-1;sumr=0;sumrold=0;flag_tolerance_reached_for_r=0;
     while(flag_tolerance_reached_for_r==0)
     {
      r=r+1;
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
       p=-1;sump=0;sumpold=0;flag_tolerance_reached_for_p=0;
```

```
 while(flag_tolerance_reached_for_p==0)
 {
  p=p+1;
  l=-1;suml=0;sumlold=0;flag_tolerance_reached_for_l=0;
  while(flag_tolerance_reached_for_l==0)
  {
   l=l+1;
   k=-1;sumk=0; sumkold=0;flag_tolerance_reached_for_k=0;
   while(flag_tolerance_reached_for_k==0)
   {
    k=k+1;
    u=0;sumu=0;sumuold=0;flag_tolerance_reached_for_u=0;
    while (flag_tolerance_reached_for_u==0)
    {
     u=u+1;
     term4=(1/(1-Gs))*pow(Gs,(double)(n+r-q))*(factor_term3((double)i,
     (double)j,(double)m,(double)n,(double)p,(double)q,(double)r))*pow
     (-1,(double)(n+i+m+p+k+q+1))*Choose((double)(-i),(double)k)*
     Choose((double)(-j),(double)l)*pow((4/(Rratio*Cratio)),(double)
     (k+l+m+p))*(4-(pow(1+Gs,2.0)*pow(Gs,(double)(u-1))))*
     inverselaplace((-2*i)-(2*j)-k-l-m-p-q-r-u,Tratio,(double)
     (2*n+1),Rratio);
     temp4=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term4);
     sumuold=sumu;
     sumu=sumu+temp4;
     sum2=sum2+temp4;
     if(fabs(sumu-sumuold) <=fabs(tolerance*sumu))
     flag_tolerance_reached_for_u=1;
    }
    sumkold=sumk;
    sumk=sum2;
    if( fabs(sumkold-sumk) <= fabs(tolerance*sumk))
    flag_tolerance_reached_for_k=1;
   }
   sumlold=suml;
   suml=sum2;
   if( fabs(sumlold-suml) <= fabs(tolerance*suml))
   flag_tolerance_reached_for_l=1;
  }
  sumpold=sump;
  sump=sum2;
  if( fabs(sumpold-sump) <= fabs(tolerance*sump))
  flag_tolerance_reached_for_p=1;
 }
 sumjold=sumj;
 sumj=sum2;
 if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
 flag_tolerance_reached_for_j=1;
}
sumrold=sumr;
sumr=sum2;
```

```
        if( fabs(sumrold-sumr) <= fabs(tolerance*sumr))
        flag_tolerance_reached_for_r=1;
       }
      }
     }
    }
   }
  }
  sum2d=sum2;
  //printf(" * sum2d=%g\n\n", sum2);
  sum2=sum2a+sum2b+sum2c+sum2d;
  /*#######################################################*/
  /*****************************************************************/
  //printf("sum2c+sum2d=%g\n", sum2c+sum2d);
  //printf("sum2=%g\n", sum2a+sum2b+sum2c+sum2d);
  Vinf=sum1;
  Vc=(sum1+sum2a+sum2b+sum2c+sum2d);
  //printf("Voltage at the output is = %g\n\n", Vc);
  return Vc;
 }
}
/*******************Model 2 ends here ****************************************/


/********************Model 3 begins here ******************************/

double Model3(double timeT)
{
 int g, n,i,j,m,p,q,k,order;
 double tolerance = 1E-2;
 double tolerance_k = 0.1;
 double timeTeff;
 double Vc=0,In,sum_old=0,sum1=0,sum2=0,sum2a=0,sum2b=0, sum2c=0,sum2d=0,
 term1=0,term2=0,term3=0,term4=0,term3a=0,term4a=0;
 double sumi=0, sumiold=0, sumk=0, sumkold=0, sumq=0, sumqold=0, sumj=0,
 sumjold=0;
 double temp1, temp2,temp3,temp4;
 double tp1, tp2,tp3,tp4;/*to be used for debugging only*/
 double Vinf;
 double a,c;
 double test;
 double Vd=1.0;
 int flag_tolerance_reached_for_i=0, flag_tolerance_reached_for_j = 0,
 flag_tolerance_reached_for_k=0,flag_tolerance_reached_for_q=0;
 timeTeff=timeT-tf;
 //printf("********* MODEL 3 : time=%g, tf=%g\n", timeT, tf);
 g=0.5*((timeT/tf) + 1.0)  - 1;
 //printf("Zo=%g, tf=%g, sigma=%g, B=%g, Gs=%g\n",Zo, tf, sigma, B, Gs);
 /* first we calculate the n=0 term ie. the infinite line voltage*/
 if( (timeT/tf - (int)(timeT/tf)) == 0) return -100;
 else
```

```
{
 if(timeT < tf)  sum1=0.0;
 else
 {
  //printf("In this loop\n");
  /* the following is for getting the n=0 term ie. Vinf(L,s) */
  //printf("P=%g, Q=%g, M=%g, N=%g\n",P,Q,M,N);
  /*****************************************************************/
  /*First we find term1*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term1=pow(-1,(double)i)*pow(M,(double)(-i-1))*inverselaplace(i,Tratio,1.0,
   Rratio);
   sumiold=sumi;
   sumi=sumi+term1;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term1=P*sumi;
  //printf("Term1 = %g\n",term1);
  /*Next we find term2*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   term2=pow(-N,(double)(i))*inverselaplace(-i-1,Tratio,1.0,Rratio);
   sumiold=sumi;
   sumi=sumi+term2;
   if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
   flag_tolerance_reached_for_i=1;
  }
  term2=Q*sumi;
  //printf("Term2 = %g\n",term2);
  /*Next we find term3*/
  i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
  while (flag_tolerance_reached_for_i==0)
  {
   i++;
   j=0;sumj=0;sumjold=0;flag_tolerance_reached_for_j=0;
   while (flag_tolerance_reached_for_j==0)
   {
     j++;
    term3=pow(-1,(double)i)*pow(M,(double)(-i-1))*(4-pow(1+Gs,2.0)*pow(Gs,
    (double)(j-1)))*inverselaplace(i-j,Tratio,1.0,Rratio);
    sumjold=sumj;
    sumj=sumj+term3;
    if(fabs(sumj-sumjold) <= fabs(tolerance*sumj))
    flag_tolerance_reached_for_j=1;
   }
```

```
  sumiold=sumi;
  sumi=sumi+sumj;
  if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
  flag_tolerance_reached_for_i=1;
 }
 term3=(P/(1-Gs))*sumi;
 //printf("Term3 = %g\n",term3);

 /*Next we find term4*/
 i=-1;sumi=0;sumiold=0;flag_tolerance_reached_for_i=0;
 while (flag_tolerance_reached_for_i==0)
 {
  i++;
  j=0;sumj=0;sumjold=0;flag_tolerance_reached_for_j=0;
  while (flag_tolerance_reached_for_j==0)
  {
   j++;
   term4=pow(-N,(double)(i))*(4-pow(1+Gs,2.0)*pow(Gs,(double)(j-1)))*
   inverselaplace(-i-j-1,Tratio,1.0,Rratio);
   sumjold=sumj;
   sumj=sumj+term4;
   if(fabs(sumj-sumjold) <= fabs(tolerance*sumj))
   flag_tolerance_reached_for_j=1;
  }
  sumiold=sumi;
  sumi=sumi+sumj;
  if (fabs(sumi-sumiold) <= fabs(tolerance*sumi))
  flag_tolerance_reached_for_i=1;
 }
 term4=(Q/(1-Gs))*sumi;
 //printf("Term4 = %g\n",term4);
 /*****************************************************************/
 sum1=term1+term2+term3+term4;
 //printf("sum1 = %g\n",sum1);
}
sum1=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*sum1;
//printf("Voltage1=%g\n\n", sum1);
/*******************************************************************/
/* now we do the summation from n=1 to n=g where g is the number of
reflections*/
/* first we find sum2a=term1+term2*/
sum2=0;
for(n=1; n<=g; n++)
{
 if(timeT < ((2*n+1)*tf))  sum2=0;
 else
 {
  for(i=0; i <=n; i ++)
  {
   for(p=0; p<=n; p++)
   {
```

```
      j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
      while(flag_tolerance_reached_for_j==0)
      {
       j=j+1;
       for(m=0; m<=(i+j); m++)
       {
        q=-1;sumq=0;sumqold=0;flag_tolerance_reached_for_q=0;
        while(flag_tolerance_reached_for_q==0)
        {
         q=q+1;
         term1=pow(Gs,(double)(n+q-p))*factor_term1_model3(i,j,n,p,q)*pow(-1,
         (double)(i+j+p+m))*pow(((Rratio*Cratio)/4),(double)(i+j))*Choose(
         (double)(i+j),(double)m)*inverselaplace(i+j-2*m-p-q,Tratio,2*n+1,
         Rratio);
         temp1=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term1);
         sum2=sum2+temp1;
         sumqold=sumq;
         sumq=sum2;
         if( fabs(sumqold-sumq) <= fabs(tolerance*sumq))
         flag_tolerance_reached_for_q=1;
        }
       }
       sumjold=sumj;
       sumj=sum2;
       if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
       flag_tolerance_reached_for_j=1;
      }
     }
    }
   }
  }
sum2a=sum2;
sum2=0;
//printf("sum2a = %g\n",sum2a);
/*********************************************************************/
for(n=1; n<=g; n++)
{
 if(timeT < ((2*n+1)*tf))  sum2=0;
 else
 {
  for(i=0; i <=n; i ++)
  {
   for(p=0; p<=n; p++)
   {
    j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
    while(flag_tolerance_reached_for_j==0)
    {
     j=j+1;
     for(m=0; m<=(i+j); m++)
     {
      q=-1;sumq=0;sumqold=0;flag_tolerance_reached_for_q=0;
```

```
        while(flag_tolerance_reached_for_q==0)
        {
         q=q+1;
         flag_tolerance_reached_for_k=0;
         k=0;sumk=0;sumkold=0;
         while (flag_tolerance_reached_for_k==0)
         {
          k=k+1;
          term2=(1/(1-Gs))*pow(Gs,(double)(n+q-p))*factor_term1_model3
          (i,j,n,p,q)*pow(-1,(double)(i+j+p+m))*pow((Rratio*Cratio/4),
          (double)(i+j))*Choose((double)(i+j),(double)m)*(4-(pow(1+Gs,2.0)
          *pow(Gs,(double)(k-1))))*inverselaplace(i+j-2*m-p-q-k,Tratio,2*n+1,
          Rratio);
          sumkold=sumk;
          temp2=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term2);
          sumk=sumk+temp2;
          sum2=sum2+temp2;
          if (fabs(sumk-sumkold) <= fabs(tolerance_k*sumk))
          flag_tolerance_reached_for_k=1;
         }
         sumqold=sumq;
         sumq=sum2;
         if( fabs(sumqold-sumq) <= fabs(tolerance*sumq))
         flag_tolerance_reached_for_q=1;
        }
       }
       sumjold=sumj;
       sumj=sum2;
       if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
       flag_tolerance_reached_for_j=1;
      }
     }
    }
   }
  }
sum2b=sum2;
sum2=0;
//printf("sum2b = %g\n",sum2b);
/*********************************************************************/
//printf("sum2a+sum2b=%g\n", sum2a+sum2b);
sum2=0;
/*************************************/
/* next we find sum2b=term3+term4*/
for(n=1; n<=g; n++)
{
 if(timeT < ((2*n+1)*tf))  sum2=0;
 else
 {
  for(i=0; i <=n+1; i ++)
  {
   for(p=0; p<=n; p++)
```

```
    {
     j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
     while(flag_tolerance_reached_for_j==0)
     {
      j=j+1;
      for(m=0; m<=(i+j); m++)
      {
       q=-1;sumq=0;sumqold=0;flag_tolerance_reached_for_q=0;
       while(flag_tolerance_reached_for_q==0)
       {
        q=q+1;
        term3=pow(Gs,(double)(n+q-p))*factor_term3_model3(i,j,n,p,q)*pow(-1,
        (double)(i+j+p+m))*pow((Rratio*Cratio/4),(double)(i+j))*Choose(
        (double)(i+j),(double)m)*inverselaplace(i+j-2*m-p-q,Tratio,
        2*n+1,Rratio);
        temp3=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term3);
        sum2=sum2+temp3;
        sumqold=sumq;
        sumq=sum2;
        if( fabs(sumqold-sumq) <= fabs(tolerance*sumq))
        flag_tolerance_reached_for_q=1;
       }
      }
      sumjold=sumj;
      sumj=sum2;
      if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
      flag_tolerance_reached_for_j=1;
     }
    }
   }
  }
 }
sum2c=sum2;
//printf("sum2c = %g\n",sum2c);
sum2=0;
/*****************************************************************/
for(n=1; n<=g; n++)
{
 if(timeT < ((2*n+1)*tf))  sum2=0;
 else
 {
  for(i=0; i <=n+1; i ++)
  {
   for(p=0; p<=n; p++)
   {
    j=-1;sumj=0; sumjold=0;flag_tolerance_reached_for_j=0;
    while(flag_tolerance_reached_for_j==0)
    {
     j=j+1;
     for(m=0; m<=(i+j); m++)
     {
```

```
        q=-1;sumq=0;sumqold=0;flag_tolerance_reached_for_q=0;
        while(flag_tolerance_reached_for_q==0)
        {
         q=q+1;
         flag_tolerance_reached_for_k=0;
         k=0;sumk=0;sumkold=0;
         while (flag_tolerance_reached_for_k==0)
         {
          k=k+1;
          term4=(1/(1-Gs))*pow(Gs,(double)(n+q-p))*factor_term3_model3
          (i,j,n,p,q)*pow(-1,(double)(i+j+p+m))*pow((Rratio*Cratio/4),
          (double)(i+j))*Choose((double)(i+j),(double)m)*(4-(pow(1+Gs,2.0)
          *pow(Gs,(double)(k-1))))*inverselaplace(i+j-2*m-p-q-k,Tratio,
          2*n+1,Rratio);
          sumkold=sumk;
          temp4=(Vd*((1-Gs)/2.0)*exp(-Rratio*Tratio/2.0))*(term4);
          sumk=sumk+temp4;
          sum2=sum2+temp4;
          if (fabs(sumk-sumkold) <= fabs(tolerance_k*sumk))
          flag_tolerance_reached_for_k=1;
         }
         sumqold=sumq;
         sumq=sum2;
         if( fabs(sumqold-sumq) <= fabs(tolerance*sumq))
         flag_tolerance_reached_for_q=1;
        }
       }
       sumjold=sumj;
       sumj=sum2;
       if( fabs(sumjold-sumj) <= fabs(tolerance*sumj))
       flag_tolerance_reached_for_j=1;
      }
     }
    }
   }
  }
  sum2d=sum2;
  //printf("sum2d = %g\n",sum2d);
  /*************************************************************/
  //printf("sum2c+sum2d=%g\n", sum2c+sum2d);
  //printf("sum2=%g\n", sum2a+sum2b+sum2c+sum2d);
  Vinf=sum1;
  Vc=(sum1+sum2a+sum2b+sum2c+sum2d);
  //printf("Voltage at the output is = %g\n\n", Vc);
  return Vc;
 }
}
/*********************Model 3 ends here ************************/


double factor_term1(double i, double j, double m, double n, double p,
```

```
                    double q, double r)
{
 /*returns a messy factor for term1 and term2*/
 double temp1, temp2, temp3, temp4, temp5, temp6, temp_total;
 int GETOUT;
 GETOUT = 0;
 temp1 = n-1.0+j;
 temp2 = n-1.0+p;
 temp3 = n-1.0+r;
 temp4 = n-i;
 temp5 = n-m;
 temp6 = n-q;
 temp_total=n*n*n;
 while( !GETOUT)
 {
  if (temp1 <= 0.0) temp1 = 1.0;
  if (temp2 <= 0.0) temp2 = 1.0;
  if (temp3 <= 0.0) temp3 = 1.0;
  if (temp4 <= 0.0) temp4 = 1.0;
  if (temp5 <= 0.0) temp5 = 1.0;
  if (temp6 <= 0.0) temp6 = 1.0;
  if (i <= 0.0) i = 1.0;
  if (j <= 0.0) j = 1.0;
  if (m <= 0.0) m = 1.0;
  if (p <= 0.0) p = 1.0;
  if (q <= 0.0) q = 1.0;
  if (r <= 0.0) r = 1.0;
  if (temp1*temp2*temp3*temp4*temp5*temp6*i*j*m*p*q*r == 1.0) GETOUT = 1;
  temp_total *= temp1*temp2*temp3/(i*j*m*p*q*r*temp4*temp5*temp6);
  temp1 -=1.0;
  temp2 -=1.0;
  temp3 -=1.0;
  temp4 -=1.0;
  temp5 -=1.0;
  temp6 -=1.0;
  i-=1.0;
  j-=1.0;
  m-=1.0;
  p-=1.0;
  q-=1.0;
  r-=1.0;
 }//while end
 return( temp_total );
}


double factor_term3(double i, double j, double m, double n, double p,
                    double q, double r)
{
 /*returns a messy factor for term3 and term4*/
 double temp1, temp2, temp3, temp4, temp5, temp6, temp_total;
```

```
 int GETOUT;
 GETOUT = 0;
 temp1 = n+j;
 temp2 = n+p;
 temp3 = n-1.0+r;
 temp4 = n+1.0-i;
 temp5 = n+1.0-m;
 temp6 = n-q;
 temp_total=(n+1.0)*(n+1.0)*n;
 while( !GETOUT)
 {
  if (temp1 <= 0.0) temp1 = 1.0;
  if (temp2 <= 0.0) temp2 = 1.0;
  if (temp3 <= 0.0) temp3 = 1.0;
  if (temp4 <= 0.0) temp4 = 1.0;
  if (temp5 <= 0.0) temp5 = 1.0;
  if (temp6 <= 0.0) temp6 = 1.0;
  if (i <= 0.0) i = 1.0;
  if (j <= 0.0) j = 1.0;
  if (m <= 0.0) m = 1.0;
  if (p <= 0.0) p = 1.0;
  if (q <= 0.0) q = 1.0;
  if (r <= 0.0) r = 1.0;
  if (temp1*temp2*temp3*temp4*temp5*temp6*i*j*m*p*q*r == 1.0) GETOUT = 1;
  temp_total *= temp1*temp2*temp3/(i*j*m*p*q*r*temp4*temp5*temp6);
  temp1 -=1.0;
  temp2 -=1.0;
  temp3 -=1.0;
  temp4 -=1.0;
  temp5 -=1.0;
  temp6 -=1.0;
  i-=1.0;
  j-=1.0;
  m-=1.0;
  p-=1.0;
  q-=1.0;
  r-=1.0;
 }//while end
 return( temp_total );
}


double factor_term1_model3(double i, double j, double n, double p, double q)
{
 /*returns a messy factor for term1 and term2*/
 double temp1, temp2, temp3, temp4, temp5, temp6, temp_total;
 int GETOUT;
 GETOUT = 0;
 temp1 = n-1.0+j;
 temp2 = n-1.0+q;
 temp4 = n-i;
```

```
 temp6 = n-p;
 temp_total=n*n;
 while( !GETOUT)
 {
  if (temp1 <= 0.0) temp1 = 1.0;
  if (temp2 <= 0.0) temp2 = 1.0;
  if (temp4 <= 0.0) temp4 = 1.0;
  if (temp6 <= 0.0) temp6 = 1.0;
  if (i <= 0.0) i = 1.0;
  if (j <= 0.0) j = 1.0;
  if (p <= 0.0) p = 1.0;
  if (q <= 0.0) q = 1.0;
  if (temp1*temp2*temp4*temp6*i*j*p*q == 1.0) GETOUT = 1;
  temp_total *= temp1*temp2/(i*j*p*q*temp4*temp6);
  temp1 -=1.0;
  temp2 -=1.0;
  temp4 -=1.0;
  temp6 -=1.0;
  i-=1.0;
  j-=1.0;
  p-=1.0;
  q-=1.0;
 }//while end
 return( temp_total );
}


double factor_term3_model3(double i, double j, double n, double p, double q)
{
 /*returns a messy factor for term3 and term4*/
 double temp1, temp2, temp3, temp4, temp5, temp6, temp_total;
 int GETOUT;
 GETOUT = 0;
 temp1 = n+j;
 temp3 = n-1.0+q;
 temp4 = n+1.0-i;
 temp6 = n-p;
 temp_total=(n+1.0)*n;
 while( !GETOUT)
 {
  if (temp1 <= 0.0) temp1 = 1.0;
  if (temp3 <= 0.0) temp3 = 1.0;
  if (temp4 <= 0.0) temp4 = 1.0;
  if (temp6 <= 0.0) temp6 = 1.0;
  if (i <= 0.0) i = 1.0;
  if (j <= 0.0) j = 1.0;
  if (p <= 0.0) p = 1.0;
  if (q <= 0.0) q = 1.0;
  if (temp1*temp3*temp4*temp6*i*j*p*q == 1.0) GETOUT = 1;
  temp_total *= temp1*temp3/(i*j*p*q*temp4*temp6);
  temp1 -=1.0;
```

```
  temp3 -=1.0;
  temp4 -=1.0;
  temp6 -=1.0;
  i-=1.0;
  j-=1.0;
  p-=1.0;
  q-=1.0;
 }//while end
 return( temp_total );
}


double inverselaplace(int n, double tratio, double b, double rratio)
{
 double inverse,m,k,temp;
 int y;
 if (n < 0)
 {
  n=-1*n;
  inverse=pow((tratio-b)/(tratio+b),(double)(n/2.0))*besselfunction(n,
  (rratio/2.0)*sqrt((tratio*tratio)-(b*b)));
 }
 else if (n > 0)
 {
  temp=0;
  for (k=0; k<=(int)(n/2); k++)
  {
   for (m=0; m<=(int)(n/2); m++)
   {
    for (y=1; y<= (int)((n-(2*k)-1)/2)+1; y++)
    {
     temp=temp+Choose((double)n,2*m)*Choose((2*m-1)/2,k)*pow(-1,k)*
     Call_Coefficient((int)(n-2*k-1),y)*pow(rratio/2.0,-y+1)*b*pow
     (tratio,n-2*k-2*y+1)*besselfunction(n-2*k-y+1,(rratio/2.0)*
     sqrt(tratio*tratio-b*b))/pow(sqrt(tratio*tratio-b*b),n-2*k-y+1);
    }
   }
  }
  temp=2*temp+pow((tratio-b)/(tratio+b),(double)(n)/2)*besselfunction
  (n,(rratio/2.0)*sqrt(tratio*tratio-b*b));
  inverse=temp;
 }
 else if (n==0)
 {
  inverse=besselfunction(0,(rratio/2.0)*sqrt((tratio*tratio)-(b*b)));
 }
 return inverse;
}


double besselfunction(int order, double x)
```

```
{
 double In;
 if (order < 0) order=-order;
 if( order == 0)
 In=bessi0(x);
 else if( order == 1)
 In=bessi1(x);
 else
 In=bessi(order,x);
 return In;
}


int absolute(int number)
{
 if (number >=0) number=number;
 else number=-number;
 return number;
}


double factorial(double x)
{
 double i;
 double temp=1.0;
 if (x < 0) temp=0;
 else if (x == 0) temp=1.0;
 else
 {
  for (i=1.0; i<=x; i++)
  {
   temp=temp*i;
  }
 }
 return temp;
}


double Choose(double x, double y)
{
 int GETOUT;
 double result=1.0;
 double temp, temp1, temp_total;
 if( (y - (int)y) != 0) result=0.0;
 else if( (x - (int)x) == 0)
 {
  if (y < 0 || (x > 0 && x < y)) result=0.0;
  else if (x < 0) result=pow(-1,y)*Choose1(-x+y-1,y);
  else if (x==0 && y!=0) result=0.0;
  else if (x==0 && y==0) result=1.0;
  else result=Choose1(x,y);
```

```
 }
 else
 {
  if (y < 0) result=0.0;
  else if (x < 0) result=pow(-1,y)*Choose2(-x+y-1,y);
  else if (x==0 && y!=0) result=0.0;
  else if (x==0 && y==0) result=1.0;
  else result=Choose2(x,y);
 }
 return result;
}


double Choose1(double x, double y)
{
 /*for integer x*/
 int GETOUT;
 double result=1.0;
 double temp1, temp_total;
 temp_total=1.0;
 GETOUT = 0;
 temp1 = x-y;
 while( !GETOUT)
 {
  if (temp1 <= 0.0) temp1 = 1.0;
  if (x <= 0.0) x = 1.0;
  if (y <= 0.0) y = 1.0;
  if (temp1*x*y == 1.0) GETOUT = 1;
  temp_total *= x/(y*temp1);
  temp1 -= 1.0;
  x-=1.0;
  y-=1.0;
 }//while end
 return temp_total;
}


double Choose2(double x, double y)
{
 /*for fractional x*/
 int GETOUT;
 double result=1.0;
 double temp, temp1;
 temp=x;   temp1=y;
 while(temp >= (x-y+1.0) )
 {
  if (temp1 <= 0.0) temp1 = 1.0;
  result=result*temp/temp1;
  temp=temp-1.0;
  temp1=temp1-1.0;
 }
```

```
 return result;
}


void initialize_Coefficient(void)
{
 int i,j;
 for (i=0; i<=20; i++)
 {
  for (j=0; j<=20; j++)
  {
   Coeff[i,j]=Coefficient(i,j);
  }
 }
 //printf("Done Initializing\n");
}


int Call_Coefficient(int x, int y)
{
 int coeff;
 if ( x>=0 && x<=20 && y>=0 && y<=20)
 coeff=Coeff[x,y];
 else coeff=Coefficient(x,y);
 return coeff;
}


int Coefficient(int x, int y)
{
 int Coeff=0;
 if ((x<=-1) || (y<=-1)) return 0;
 else if (y==0) return 0;
 else if (y==1) return 1;
 else if (x==0) return 0;
 else if (y > ((int)(x/2)+1)) return 0;
 else
 {
  Coeff=((x+3-(2*y))*Coefficient(x-1,y-1))+Coefficient(x-1,y);
  return Coeff;
 }
}
```

B.3 Sample input file

```
Rint            [Ohms-cm]       42.5
Cint            [F/cm]          1.0987e-12
```

```
Lint            [H/cm]          4.04508E-9
Rtr             [Ohms]          23.54
Cload           [F]             5e-18
length          [cms]           1.0
starttime       [secs]          1E-12
endtime         [secs]          4E-10
deltatime       [secs]          5E-11
```

B.4 Output file

```
*************************************************************************
Program to calculate step -input transient response of single, distributed
rlc interconnect
                Code for paper in proceedings DAC 2002, pp.763-766
                        Author : Raguraman Venkatesan
                         Gigascale Integration Group
                    Georgia Institute of Technology, Atlanta
                                 July 2002

*************************************************************************
Program concluded. See output_singlelinetransient.dat for output data
points
*************************************************************************


1e-12       0
5.1e-11     0
1.01e-10    1.09152
1.51e-10    1.1817
2.01e-10    1.02935
2.51e-10    0.994209
3.01e-10    0.960205
3.51e-10    0.991409
```

REFERENCES

[1] T.R.Halfhill, "Embedded microprocessors," *Computer World*, Aug. 2000.

[2] G.E.Moore, "Cramming more components onto integrated circuits," *Electronics*, vol. 38, Apr. 1965.

[3] G.E.Moore, "Progress in digital integrated electronics," in *Proceedings of IEDM*, 1975.

[4] S.Borkar, "Obeying moore's law beyond 0.18 micron," in *Proceedings of ASIC/SOC Conference*, pp. 26–31, 2000.

[5] http://www.intel.com/research/silicon/mooreslaw.htm

[6] Semiconductor Research Association, San Jose, CA, *International Technology Roadmap for Semiconductors*, 2001.

[7] J.Crawford, "Keynote speech," in *Microprocessor Forum*, (San Jose, CA), Oct 2002.

[8] J.D.Meindl, "Low power microelectronics : Retrospect and prospect," *Proc. IEEE*, vol. 83, pp. 619–635, Apr. 1995.

[9] J.A.Davis, R.Venkatesan, A.Kaloyeros, M.Bylansky, S.J.Souri, K.Banerjee, K.C.Saraswat, A.Rahman, A.Reif, and J.D.Meindl, "Interconnect limits on gigascale integration (gsi) in the 21st century," *Proc. IEEE*, vol. 89, pp. 305–324, Mar. 2001.

[10] J.A.Davis, *A hierarchy of interconnect limits for gigascale integration.* PhD thesis, Georgia Institute of Technology, July 1999.

[11] Semiconductor Research Association, San Jose, CA, *International Technology Roadmap for Semiconductors*, 1999.

[12] J.D.Meindl, R.Venkatesan, J.A.Davis, J.Joyner, A.Naeemi, P.Zarkesh-Ha, M.Bakir, T.Mule, P.A.Kohl, and K.P.Martin, "Interconnecting device opportunities for gigascale integration (gsi)," in *Proceedings of IEDM*, (Washington DC), pp. 525–528, 2001.

[13] H.B.Bakoglu, *Circuits, interconnections and packaging for VLSI*. Reading, MA: Addison-Wesley, 1990.

[14] H.B.Bakoglu and J.D.Meindl, "Optimal interconnection circuits for vlsi," *IEEE Trans. Elect. Devices*, vol. 32, May 1985.

[15] J.A.Davis and J.D.Meindl, *Interconnect design issues and opportunities for Giga Scale Integration (GSI)*. Kluwer Academic Publishers, 2002.

[16] G.A.Sai-Halasz, "Performance trends in high-end processors," *Proc. IEEE*, vol. 83, pp. 18–34, Jan. 1995.

[17] A.B.Kahng and D.Stroobandt, "Wiring layer assignments with consistent stage delays," in *Proceedings of the International Workshop on System Level Interconnect Prediction*, (San Diego, CA), pp. 115–122, 2000.

[18] Q.Chen, J.A.Davis, P.Zarkesh-Ha, and J.D.Meindl, "A compact physical via-blockage model," *IEEE Trans. VLSI Systems*, vol. 8, pp. 689–692, Dec. 2000.

[19] J.A.Davis, V.K.De, and J.D.Meindl, "A stochastic wire-length distribution for gigascale integration (gsi) - part i : Derivation and validation," *IEEE Trans. Electron Devices*, vol. 45, pp. 580–589, Mar. 1998.

[20] J.A.Davis, V.K.De, and J.D.Meindl, "A stochastic wire-length distribution for gigascale integration (gsi) - part ii : Applications to clock frequency, power dissipation and chip size estimation," *IEEE Trans. Electron Devices*, vol. 45, pp. 590–597, Mar. 1998.

[21] T.Sakurai, "Closed form expressions for interconnection delay, coupling and crosstalk in vlsi's," *IEEE Trans. Electron Devices*, vol. 40, pp. 118–124, Jan. 1993.

[22] F.Y.Chang, "Transient simulation of nonuniform coupled lossy transmission lines characterized with frequency-dependent parameters - part i : Waveform relaxation analysis," *IEEE Trans. Circuits and Systems - 1 : Fundamental Theory and Applications*, vol. 39, pp. 585–603, Aug. 1992.

[23] F.Y.Chang, "Transient simulation of nonuniform coupled lossy transmission lines characterized with frequency-dependent parameters - part ii : Discrete-time analysis," *IEEE Trans. Circuits and Systems - 1 : Fundamental Theory and Applications*, vol. 39, pp. 907–927, Nov. 1992.

[24] J.E.Bracken, V.Raghavan, and R.A.Rohrer, "Interconnect simulation with asymptotic waveform evaluation (awe)," *IEEE Trans. Circuits and Systems - 1 : Fundamental Theory and Applications*, vol. 39, pp. 869–878, Nov. 1992.

[25] L.M.Silveira, I.M.Elfadel, J.K.White, M.Chilukuri, and K.S.Kundert, "Efficient frequency-domain modeling and circuit simulation of transmisson lines," *IEEE Trans. Components, Packaging, and Manufacturing Technology - Part B*, vol. 17, pp. 505–513, Nov. 1994.

[26] D.B.Kuznetsov and J.E.Schutt-Aine, "Optimal transient simulation of transmission lines," *IEEE Trans. Circuits and Systems - 1 : Fundamental Theory and Applications*, vol. 43, pp. 110–121, Feb. 1996.

[27] J.S.Roychowdhury, A.R.Newton, and D.O.Pederson, "Algorithms for the transient simulation of lossy interconnect," *IEEE Trans. CAD of Integrated Circuits and Systems*, vol. 13, pp. 96–104, Jan. 1994.

[28] J.A.Davis and J.D.Meindl, "Compact distributed *rlc* interconnect models - part i : Single line transient, time delay and overshoot expressions," *IEEE Trans. Electron Devices*, vol. 47, pp. 2068–2077, Nov. 2000.

[29] J.A.Davis and J.D.Meindl, "Compact distributed *rlc* interconnect models - part ii : Coupled line transient expressions and peak crosstalk in multilevel interconnect networks," *IEEE Trans. Electron Devices*, vol. 47, pp. 2078–2087, Nov. 2000.

[30] Y.I.Ismail and E.G.Friedman, "Effects of inductance on the propagation delay and repeater insertion in vlsi circuits," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 8, pp. 195–206, Apr. 2000.

[31] Y.Cao, X.Huang, D.Sylvester, N.Chang, and C.Hu, "A new analytical delay and noise model for on-chip rlc interconnect," in *Proceedings of IEDM*, (San Francisco, CA), pp. 823–826, 2000.

[32] K.Banerjee and A.Mehrotra, "Analysis of on-chip inductance effects using a novel performance optimization methodology for distributed rlc interconnnects," in *Proceedings of Design Automation Conf.*, (Las Vegas), pp. 798–803, 2001.

[33] K.Banerjee and A.Mehrotra, "Accurate analysis of on-chip inductance effects and implications for optimal repeater insertion and technology scaling," in *Proceedings of IEEE Sym. on VLSI Circuits*, (Kyoto, Japan), pp. 195–198, 2001.

[34] K.Banerjee and A.Mehrotra, "Analysis of on-chip inductance effects for distributed rlc interconnects," *IEEE Trans. CAD of Integrated Circuits and Systems*, vol. 21, pp. 904–915, Aug. 2002.

[35] J.C.Eble, *A generic system simulator with novel on-chip cache and throughput models for gigascale integration.* PhD thesis, Georgia Institute of Technology, Nov. 1998.

[36] P. Zarkesh-Ha, J. A. Davis, and J. D. Meindl, "Prediction of net-length distribution for global interconnects in a heterogeneous system-on-a-chip," *IEEE Trans. VLSI Systems*, pp. 649–659, Dec. 2000.

[37] P.Zarkesh-Ha, *Global interconnect modeling for a gigascale system-on-a-chip (GSoC).* PhD thesis, Georgia Institute of Technology, Feb 2001.

[38] A.Bhavnagarwala, A.Kapoor, and J.D.Meindl, "Impact of interconnect parameter variation on wire-tree delay," in *Proceedings of Fifth Intl. Workshop on Statistical Metrology*, (Honolulu, HI), pp. 80–83, 2000.

[39] A.Bhavnagarwala, *Voltage scaling constraints for static CMOS logic and memory circuits.* PhD thesis, Georgia Institute of Technology, May 2001.

[40] G.Hinton *et al.*, "A 0.18-μm cmos ia-32 processor with a 4-ghz integer execution unit," *IEEE J. Solid-State Circuits*, vol. 36, pp. 1617–1627, Nov. 2001.

[41] J.H.Chern, J.Huang, L.Alredge, P.C.Li, and P.Yang, "Multilevel metal capacitance models for cad design synthesis systems," *IEEE Electron Device Letters*, vol. 13, pp. –, Jan. 1992.

[42] W.E.Donath, "Wire length distribution for placement of computer logic," *IBM J. Res. Dev.*, vol. 2, pp. 152–155, May 1981.

[43] S.Tyagi *et al.*, "A 130 nm generation logic technology featuring 70 nm transistors, dual vt transistors and 6 layers of cu interconnects," in *Proceedings of IEDM*, (San Francisco, CA), 2000.

[44] S.L.Teig, "The x-architecture : not your father's diagonal wiring," in *Proceedings of the Workshop on System Level Interconnect Prediction*, (San Diego, CA), pp. 33–37, 2002.

[45] J.A.Davis and J.D.Meindl, "Length, scaling and material dependence of crosstalk between distributed rc interconnects," in *Proceedings of International Interconnect Technology Conference*, (San Francisco, CA), pp. 227–229, 1999.

[46] A.B.Kahng, S.Mantik, and D.Stroobandt, "Requirements for models of achievable routing," in *Proceedings of ISPD*, (San Diego, CA), pp. 4–11, 2000.

[47] R.Venkatesan, J.A.Davis, and J.D.Meindl, "Performance enhancement through optimal n-tier multilevel interconnect architectures," in *Proceedings of ASIC*, (Washington DC), pp. 19–23, 1999.

[48] R.McInerney *et al.*, "Methodology for repeater insertion management in the rtl, layout, floorplan and full chip timing databases of the itanium microprocessor," in *Proceedings of ISPD*, (San Diego, CA), pp. 99–104, 2000.

[49] K.A.Bowman, B.L.Austin, X.Tang, J.C.Eble, and J.D.Meindl, "A physical alpha-power law mosfet model," *IEEE J. Solid-State Circuits*, vol. 34, pp. 410–414, Oct. 1999.

[50] K.A.Bowman, *A circuit level perspective of opportunities and limitations for gigascale integration.* PhD thesis, Georgia Institute of Technology, April 2001.

[51] T.Sakurai and A.R.Newton, "Delay analysis for series-connected mosfet circuits," *IEEE J. Solid State Circuits*, vol. 26, pp. 122–131, Feb. 1991.

[52] Y.Taur *et al.*, "Cmos scaling into the nanometer regime," *Proc. IEEE*, vol. 85, pp. 486–504, Apr. 1997.

[53] F.E.Anderson, J.S.Wills, and E.Z.Berta, "The core clock system on the next generation itanium^{TM} microprocessor," in *Proceedings of ISSCC*, (San Francisco, CA), pp. 146–147, 2002.

[54] A.Kowalczyk *et al.*, "First generation majc dual microprocessor," in *Proceedings of ISSCC*, (San Francisco, CA), pp. 236–237, 2001.

[55] K.Bernstein and N.J.Rohrer, *SOI circuit design concepts.* Kluwer academic publishers, 1999.

[56] G.E.Roberts and H.Kaufman, *Table of Laplace Transforms.* W.B.Saunders Company, 1966.

[57] O.Heaviside, *Electromagnetic Theory.* New York: Chelsea Publishing Company, 1893 (Reprint - 1971).

[58] E. Kreyszig, *Advanced Engineering Mathematics.* John Wiley and Sons, 1997.

[59] W.H.Press, S.A.Teukolsky, W.T.Vetterling, and B.P.Flannery, *Numerical Recipes in C.* Cambridge University Press, 1992.

[60] A.Naeemi, R.Venkatesan, and J.D.Meindl, "System-on-a-chip global interconnect optimization," in *Proceedings of ASIC/SOC Conference*, (Rochester, NY), pp. 399–403, 2002.

[61] A.Naeemi, R.Venkatesan, and J.D.Meindl, "Optimal global interconnecting devices for gigascale integration," *to be published in IEEE Trans. Electron Devices*, 2003.

[62] P.Zarkesh-Ha and J.D.Meindl, "An integrated architecture for global interconnects in a gigascale system-on-a-chip (gsoc)," in *Proceedings of Symposium on VLSI Technology*, (Honolulu, HI), pp. 194–195, 2000.

[63] A.Naeemi, J.A.Davis, and J.D.Meindl, "Analytical models for coupled distributed rlc lines with ideal and non-ideal return paths," in *Proceedings of IEDM*, (Washington DC), pp. –, 2001.

[64] A.Deutsch *et al.*, "On-chip wiring design challenges for gigahertz operation," *Proc. IEEE*, vol. 89, pp. 529–555, Apr. 2001.

[65] S.Thompson *et al.*, "An enhanced 130 nm generation logic technology featuring 60 nm transistors, optimized for high performance and low power at 0.7-1.4v," in *Proceedings of IEDM*, (San Francisco, CA), pp. 13.3.1–13.3.4, 2001.

[66] R.Venkatesan, J.A.Davis, K.A.Bowman, and J.D.Meindl, "Minimum power and area n-tier multilevel interconnect architectures using optimal repeater insertion," in *Proceedings of ISLPED*, (Rapallo/Portofino, Italy), pp. 167–172, 2000.

[67] K.Raol in *Private communication.*

[68] B.S.Landman and R.L.Russo, "On a pin versus block relationship for partition of logic paths," *IEEE Trans. on Computing*, vol. C-20, pp. 1469–1479, Dec. 1971.

[69] L.Hagen, A.B.Kahng, F.J.Kurdahi, and C.Ramachandran, "On the intrinsic rent parameter and spectra-based partitioning methodologies," *IEEE Trans. CAD of Integrated Circuits and Systems*, vol. 13, pp. 27–37, Jan. 1994.

[70] A.Saini, "Design of the intel pentium processor," in *Proceedings of ICCD*, pp. 258–261, 1993.

[71] R.F.Crick *et al.*, "A 150mhz 0.6μm bicmos super scalar microprocessor," *IEEE Journal of Solid State Circuits*, vol. 29, pp. 1455–1463, Dec. 1994.

[72] M.R.Choudhry and J.S.Miller, "A 300mhz cmos microprocessor with multimedia technology," in *Proceedings of ISSCC*, pp. 170–171, 1997.

[73] J.Schutz and R.Wallace, "A450mhz ia32 p6 family microprocessor," in *Proceedings of ISSCC*, pp. 236–237, 1998.

[74] D.A.Draper *et al.*, "An x86 microprocessor with multimedia extensions," in *Proceedings of ISSCC*, pp. 172–173, 1997.

[75] P.Zarkesh-Ha *et al.*, "On a pin versus gate relationship for heterogeneous systems : heterogenous rent's rule," in *Proceedings of CICC*, pp. 93–96, 1998.

[76] J.W.Joyner, R.Venkatesan, P.Zarkesh-Ha, J.A.Davis, and J.D.Meindl, "Impact of three dimensional architectures on homogeneous digital circuits," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 9, pp. 922–928, Dec. 2001.

[77] J.W.Joyner in *Private communication.*

LIST OF PUBLICATIONS

- Conference Publications

1. **R.Venkatesan**, J.A.Davis and J.D.Meindl, "Performance enhancement through optimal n-tier multilevel interconnect architectures," *Proc. IEEE ASIC/SOC Conf.*, Wash. DC, Sept. 15-18 1999, pp. 19-23.
2. J.A. Davis, **R.Venkatesan**, K.A. Bowman and J.D. Meindl, "Gigascale integration (GSI) interconnect limits and n-tier multilevel interconnect architectural solutions," *Workshop on System Level Interconnect Prediction*, San Diego CA, April 8-9 2000, pp. 147-148.
3. **R.Venkatesan**, J.A.Davis, K.A.Bowman and J.D.Meindl, "Optimal repeater insertion for n-tier multilevel interconnect architectures," *Proc. IITC*, San Francisco, June 5-7 2000, pp. 132-134.
4. **R.Venkatesan**, J.A.Davis, K.A.Bowman and J.D.Meindl, "Minimum power and area n-tier multilevel interconnect architectures using optimal repeater insertion," *Proc. ISLPED*, Rapallo/Portofino, Italy, Jul 26-27 2000, pp. 167-172.
5. J.D.Meindl, **R.Venkatesan**, J.A.Davis, J.Joyner, A.Naeemi, P.Zarkesh-Ha, M. Bakir, T.Mule, P.A.Kohl and K.P.Martin, "Interconnecting device opportunities for gigascale integration (GSI)," *International Electron Devices Meeting (IEDM)*, Washington D.C., Dec 2001, pp. 525-528.
6. **R.Venkatesan**, J.A.Davis and J.D.Meindl, "A complete physical model for distributed RLC interconnects - transient voltage, time delay and crosstalk," *Proc. Design Automation Conference*, New Orleans, June 2002, pp. 763-766.
7. **R.Venkatesan**, J.A.Davis and J.D.Meindl, "Time delay, crosstalk and repeater insertion models for high," *Proc. ASIC/SOC Conference*, Rochester, NY, Sept. 25-28, 2002, pp. 404-408.
8. A.Naeemi, **R.Venkatesan** and J.D.Meindl, "System-on-a-chip global interconnect optimization," *Proc. ASIC/SoC Conference*, Rochester, NY, Sept. 25-28, 2002, pp. 399-403.

9. J.W.Joyner, **R.Venkatesan**, J.A.Davis and J.D.Meindl, "A system limit on liquid routing diagonal interconnect architecture," submitted to IITC, San Francisco CA, June, 2003.
10. **R.Venkatesan**, J.A.Davis and J.D.Meindl, "Optimal aspect ratio for GSI multilevel interconnect architectures," to be submitted to ASIC/SOC Conference, Portland OR, Sep, 2003.

- Journal Publications

1. J.A.Davis, **R.Venkatesan**, A.Kaloyeros, M.Bylansky, S.J.Souri, K.Banerjee, K.C.Saraswat, A.Rahman, R.Reif and J.D.Meindl, "Gigascale integration (GSI) interconnect limits in the 21st century," *Proceedings of the IEEE*, Vol. 89, No. 3, March 2001, pp. 305-324.
2. **R.Venkatesan**, J.A.Davis, K.A.Bowman and J.D.Meindl, "Optimal n-tier multilevel interconnect architectures for gigascale integration," *IEEE Transactions on VLSI Systems*, Vol. 9, No. 6, Dec 2001, pp. 899-912.
3. J.W.Joyner, **R.Venkatesan**, P.Zarkesh-Ha, J.A.Davis and J.D. Meindl, "Impact of three dimensional architectures on homogenous digital circuits," *IEEE Transactions on VLSI Systems*, Vol. 9, No. 6, Dec 2001, pp. 922-928.
4. **R.Venkatesan**, J.A.Davis and J.D.Meindl, "A complete physical model for distributed rlc interconnects with capacitive load Part 1 : Single line transients and coupled line crosstalk," to be published in *IEEE Trans. Electron Devices.*
5. **R.Venkatesan**, J.A.Davis and J.D.Meindl, "A complete physical model for distributed rlc interconnects with capacitive load Part 2 : Unified models for time delay, crosstalk and repeater insertion," to be published in *IEEE Trans. Electron Devices.*
6. A.Naeemi, **R.Venkatesan** and J.D.Meindl, "Optimal global interconnecting devices for Giga Scale Integration", to be published in *IEEE Trans. Electron Devices.*

- Contribution to a book chapter

1. J.A.Davis and J.D.Meindl (Editors), *Interconnect design issues and opportunities for Giga Scale Integration*, Kluwer Academic Publishers, in press.

VITA

Raguraman Venkatesan was born on May 5th 1976, in Sindri, Bihar, India. He received his B.Tech. degree in Electrical Engineering from the Indian Institute of Technology, Bombay, in June 1998. He had the honour of joining Prof. James Meindl's Giga Scale Integration (GSI) research group at Georgia Institute of Technology, Atlanta, as a Doctoral Candidate and Graduate Research Assistant in September 1998. Since Fall 1999, he has also been a member of Prof. Jeffrey Davis's Advanced Interconnect Modeling and Design (AIMD) research group at Georgia Tech. He received his M.S. degree in Electrical and Computer Engineering from Georgia Tech in May 2000. In the summer of 2000, he worked as a graduate technical intern with the Advanced Design - Logic Technology Development Group of Intel Corporation, Hillsboro, Oregon. He was awarded the Intel PhD Fellowship in 2001. His research interests are multilevel interconnect architectures, system level interconnect design and inductance modeling. He has authored or co-authored 16 journal and international conference publications and one book chapter over the past 4 years. He has also served as a reviewer for IEEE Transactions on VLSI Systems, IEEE Transactions on Electron Devices and the International Interconnect Technology Conference. After completing his PhD, he shall be joining the Advanced Design - Logic Technology Development Group, of Intel Corporation, Hillsboro, Oregon, in March 2003.