UNIVERSITY OF SOUTHAMPTON

Modelling Statistical Variability Within Circuits Using Nano-CMOS Technologies

by

Michael Merrett

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

in the Faculty of Physical and Applied Science
School of Electronics and Computer Science

June 2012
Systems have been designed and synthesized using CMOS technology for many years, with improvements in the fabrication process allowing designs to be scaled onto smaller areas with relative ease. The introduction of nano-scale CMOS technologies has ended this time of simple scaling, as variations within the silicon now dramatically affect circuit performance and manufacturing yield. These random physical variations cannot be removed from the manufacturing process, requiring that their effects are modelled, predicted and accommodated within the design process.

This thesis presents an investigation into the challenges of including these effects within the design process, with a review of the recent research conducted in incorporating variability within timing analysis tools. The conclusion from the literature review is that an accurate, efficient and transparent method of predicting the impact of statistical process variations on the performance of a circuit has not yet been created and adopted by the IC design industry.

The investigation begins with the modelling of transistor based statistical process variations at the standard cell level, where it is determined that simple statistical models do not accurately reflect the extremes in performance, and can provide overly pessimistic predictions. The techniques of Monte Carlo Cell Characterisation (MCCC) and Monte Carlo Static Timing Analysis (MCSTA) are introduced as more suitable approaches, which accurately reflect the performance of circuits as modelled by Monte Carlo SPICE simulations, with far less pessimism than the traditional method of Corner Analysis or even modern Statistical Static Timing Analysis.

The final section of this thesis focuses on practical implementations of MCSTA, where the sample sizes required to accurately predict circuit behaviour (to within 1% of SPICE) can be reduced to as few as ten, using simple statistical sampling techniques.
Declaration Of Authorship

I, Michael Anthony Merrett, declare that the thesis entitled Modelling Statistical Variability Within Circuits Using Nano-CMOS Technologies and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University;
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- Where I have consulted the published work of others, this is always clearly attributed;
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- Parts of this work have been published as listed in Appendix A;

Signed:

Date: 12/06/2012
# Contents

<table>
<thead>
<tr>
<th>Acknowledgements</th>
<th>xix</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1 Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Motivation For Research</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Aims</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Thesis Outline</td>
<td>3</td>
</tr>
<tr>
<td>1.4 Contributions</td>
<td>4</td>
</tr>
<tr>
<td>1.5 Declaration</td>
<td>4</td>
</tr>
<tr>
<td><strong>2 Background - The Significance of Process Variation</strong></td>
<td>5</td>
</tr>
<tr>
<td>2.1 Silicon Complexity</td>
<td>5</td>
</tr>
<tr>
<td>2.1.1 Transistor Scaling</td>
<td>5</td>
</tr>
<tr>
<td>2.1.2 Power Consumption</td>
<td>7</td>
</tr>
<tr>
<td>2.1.2.1 Dynamic Power</td>
<td>7</td>
</tr>
<tr>
<td>2.1.2.2 Static Power</td>
<td>8</td>
</tr>
<tr>
<td>2.1.3 Variability</td>
<td>10</td>
</tr>
<tr>
<td>2.1.3.1 Local Statistical Process Variations</td>
<td>11</td>
</tr>
<tr>
<td>2.2 System Complexity</td>
<td>11</td>
</tr>
<tr>
<td>2.2.1 Electronic Design Automation</td>
<td>12</td>
</tr>
<tr>
<td>2.2.2 Closing the Productivity Gap</td>
<td>14</td>
</tr>
<tr>
<td>2.2.2.1 Abstraction</td>
<td>14</td>
</tr>
<tr>
<td>2.2.2.2 Synthesis</td>
<td>15</td>
</tr>
<tr>
<td>2.2.2.3 Reuse</td>
<td>15</td>
</tr>
<tr>
<td>2.2.2.4 Verification</td>
<td>15</td>
</tr>
<tr>
<td>2.3 Literature Review - Shortfalls in Current Circuit Analysis Methods</td>
<td>16</td>
</tr>
<tr>
<td>2.3.1 Timing Analysis</td>
<td>17</td>
</tr>
<tr>
<td>2.3.1.1 Static Timing Analysis</td>
<td>17</td>
</tr>
<tr>
<td>2.3.1.2 Timing Slack</td>
<td>18</td>
</tr>
<tr>
<td>2.3.1.3 Delay Modelling</td>
<td>19</td>
</tr>
<tr>
<td>2.3.1.4 Corner Analysis</td>
<td>19</td>
</tr>
<tr>
<td>2.3.1.5 Statistical Static Timing Analysis</td>
<td>21</td>
</tr>
<tr>
<td>2.3.1.6 Delay Correlations and False Paths</td>
<td>22</td>
</tr>
<tr>
<td>2.3.1.7 Inter-Die and Intra-Die Variations</td>
<td>23</td>
</tr>
<tr>
<td>2.3.1.8 Incremental Analysis</td>
<td>26</td>
</tr>
<tr>
<td>2.3.1.9 Spatial Correlation</td>
<td>26</td>
</tr>
<tr>
<td>2.3.1.10 Uncertainties and Sensitivity to Variation</td>
<td>28</td>
</tr>
<tr>
<td>2.3.1.11 Non-normal Distributions of Process Variations</td>
<td>29</td>
</tr>
</tbody>
</table>
2.3.1.12 Monte Carlo Simulations .................................. 29
2.4 Summary .................................................................. 30

3 Monte Carlo Cell Characterisation ................................. 31
3.1 Atomistic Transistor Models ................................. 31
3.2 SPICE Compact Models .......................................... 33
3.3 RandomSpice ......................................................... 33
3.4 Cell Characterisation ............................................... 35
  3.4.1 Measurements .................................................. 35
    3.4.1.1 Delays ..................................................... 35
    3.4.1.2 Power ..................................................... 37
  3.4.2 Statistical Analysis of an Inverter at 130nm ............. 37
  3.4.3 Statistical Analysis of a D-Type Flip-Flop at 35nm .... 40
    3.4.3.1 Testbench ................................................. 40
    3.4.3.2 Measured Characteristics ......................... 40
    3.4.3.3 Results .................................................. 42
  3.4.4 Standardisation using Liberty NCX ..................... 45
    3.4.4.1 Library Templates .................................. 48
    3.4.4.2 Cell Templates ....................................... 49
    3.4.4.3 Cell SPICE Netlists ............................... 50
    3.4.4.4 Transistor Models ................................. 50
    3.4.4.5 HSpice Simulation Files ......................... 51
    3.4.4.6 Cell Libraries ....................................... 51
3.5 Variation Cell Characterisation ............................... 53
  3.5.1 Standardised Statistical Analysis of an Inverter at 35nm 54
  3.5.2 Goodness of Fit Testing .................................. 62
    3.5.2.1 Kolmogorov-Smirnov ......................... 62
    3.5.2.2 Two-Sample Kolmogorov-Smirnov .......... 62
    3.5.2.3 Lilliefors ............................................. 62
    3.5.2.4 Chi-Square Goodness of Fit .................. 63
3.6 Summary .................................................................. 64

4 Monte Carlo Circuit Analysis ..................................... 65
4.1 Static Timing Analysis ........................................... 65
  4.1.1 Standardisation using PrimeTime ....................... 67
  4.1.2 Example of STA ........................................... 68
  4.1.3 Monte Carlo Static Timing Analysis .................... 71
  4.1.4 Repeatability ............................................... 74
  4.1.5 Trial Results ............................................... 75
    4.1.5.1 Critical Path Analysis ......................... 76
    4.1.5.2 Circuit Performance Predictions ............. 78
  4.2 Power Analysis .................................................. 80
    4.2.1 Example of Power Analysis ......................... 81
    4.2.2 Monte Carlo Power Analysis ....................... 81
    4.2.3 Trial Results ........................................... 82
4.3 Summary .................................................................. 83
5 Critical Assessment

5.1 Comparison with Corner Analysis at 130nm

5.1.1 Test Circuit

5.1.2 Method

5.1.2.1 Monte Carlo SPICE Simulations

5.1.2.2 Corner Analysis

5.1.2.3 Monte Carlo Static Timing Analysis

5.1.3 Further sigma VTs

5.1.4 Results and Analysis

5.1.4.1 Circuit Level Statistical Variation

5.1.4.2 Variation in Critical Paths

5.1.4.3 Power, Performance and Yield

5.2 Comparison with Statistical Static Timing Analysis at 35nm

5.2.1 Test Circuits

5.2.1.1 4-bit Fast Adder

5.2.1.2 16-bit Multiplier

5.2.2 Method

5.2.2.1 Monte Carlo SPICE Simulations

5.2.2.2 Statistical Static Timing Analysis

5.2.3 Monte Carlo Static Timing Analysis

5.2.4 Results

5.3 Summary

6 Implementation - Practical MCSTA

6.1 Metric of Variability

6.1.1 Description of the proposed metric

6.1.1.1 Early estimation of path delay variability

6.1.2 Evaluating the Design Metrics

6.1.2.1 Simulation Framework

6.1.2.2 Analysis of Results

6.1.3 Summary

6.2 Statistical Sampling

6.2.1 Comparing Standard Cell Variation Instances

6.2.2 Method

6.2.3 Results

6.2.3.1 35nm 1bit Adder

6.2.3.2 35nm 4bit Fast Adder

6.2.4 Sample Size Reduction

6.2.5 Summary

7 Conclusions And Future Work

7.1 Conclusions and Contributions

7.2 Future Work

A Publications

Bibliography
List of Figures

2.1 A Time-line of Transistors per Processor and Manufacturing Process Widths 6
2.2 Schematic of a CMOS Inverter .................................................. 8
2.3 Leakage Current Mechanisms in Nanometre Transistors ................. 9
2.4 Continuous and Atomistic Transistor Models .............................. 12
2.5 A Simplified System Design Flow ............................................ 14
2.6 Illustration of Setup and Hold times ................................... 18
2.7 Illustration of Setup and Hold Timing Slacks .............................. 19
2.8 Corner Analysis ................................................................. 20
2.9 SSTA Delay Element .......................................................... 21
2.10 Converging Circuit Paths ...................................................... 23
2.11 Fanin and Fanout Logic Cones .............................................. 27

3.1 Random Discrete Dopants in a 35nm MOSFET ...................... 32
3.2 Work flow of RandomSpice statistical simulation engine ............ 34
3.3 Transition and Propagation Delays ....................................... 36
3.4 Setup and Hold Times ......................................................... 36
3.5 Power Measurements .......................................................... 37
3.6 SPICE Variation Test Circuit for a CMOS Inverter .................. 38
3.7 Distribution of Propagation Delays through a 130nm CMOS Inverter .... 39
3.8 Dtype Flip-Flop Characterisation Testbench ......................... 41
3.9 Dtype Flip-Flop Characterisation Timing Measurements ............ 41
3.10 Probability Density Function of Clock-nQ Delays for a Dtype Flip-Flop . 42
3.11 The Effects of increasing Supply Voltage on the Output Delay of a ...
Dtype Flip-Flop ................................................................. 43
3.12 The Effects of increasing Supply Voltage on the Output Transition Time
of a 35nm Dtype Flip-Flop .................................................. 44
3.13 The Effects of increasing Supply Voltage on the Leakage Energy of a ...
Dtype Flip-Flop ................................................................. 46
3.14 The Effects of increasing Supply Voltage on the Switching Energy of a ...
35nm Dtype Flip-Flop ............................................................. 47
3.15 Liberty NCX Characterisation Flow ...................................... 48
3.16 Example of a library template file. ........................................ 49
3.17 Example of a inverter cell template file. ................................ 50
3.18 Inverter sub-circuit definition ........................................... 50
3.19 Transistor model definitions .............................................. 51
3.20 Extract from an Inverter Cell Library Entry. .......................... 52
3.21 Variation Characterisation Flow .......................................... 53
3.22 Distribution of Propagation Delays through a 35nm CMOS Inverter, for a Falling Output Transition ................................. 55
3.23 Normal and Inverse Gaussian Probability plots of Propagation Delays through a 35nm CMOS Inverter, for a Falling Output Transition ...... 56
3.24 Propagation Delays versus Input Transition Time .............................................. 56
3.25 Propagation Delays versus Input Transition Time versus Load Capacitance 57
3.26 Output Transition Times versus Input Transition Time versus Load Capacitance ................................................................. 58
3.27 Switching Energy versus Input Transition Time versus Load Capacitance 58
3.28 Inverter fall delay normal plots ................................................................. 60
3.29 Inverter fall delay inverse normal plots ....................................................... 61

4.1 Delay Examples ................................................................. 66
4.2 Static Timing Analysis .............................................................. 67
4.3 Example of a Verilog gate level netlist ................................................. 68
4.4 Static Timing Analysis .............................................................. 69
4.5 Example of design constraints ........................................................... 70
4.6 Example of a timing report ............................................................... 70
4.7 Example of randomised Verilog gate level netlists ....................................... 72
4.8 Example of timing reports for the same path through randomised circuit instances .............................................................................. 73
4.9 Monte Carlo Static Timing Analysis ...................................................... 74
4.10 Comparison of HSPICE and PrimeTime STA Timings for a Critical Path 77
4.11 Histogram of Critical Path Delays ......................................................... 78
4.12 Circuit Yield Predictions ................................................................. 79
4.13 Example of PrimeTime PX Power Report ............................................. 81
4.14 Histogram of Leakage Power .............................................................. 83
4.15 Histogram of Dynamic Power ............................................................ 83

5.1 Gate level schematic of a one bit full adder .............................................. 86
5.2 Input waveforms for Monte Carlo SPICE simulations of 130nm Adder circuit ................................................................. 87
5.3 Reduction of error in the mean and standard deviation of delay distributions with increases in sample sizes ........................................ 90
5.4 Reduction of error in the mean and standard deviation of power distributions with increases in sample sizes ................................. 91
5.5 Q-Q plots of the propagation delay through different sample sizes of inverters. ........................................................................ 92
5.6 Q-Q plots of the power consumption of different sample sizes of inverters. 92
5.7 A comparison between SPICE and Corner Analysis performed at 3σ. The lines created by Corner Analysis at 3σ represent where 99.7% of the circuits are predicted to perform for each level of variation. The distribution created by SPICE represents where 100% of the circuits were actually found to perform at the highest level of variability. .......................... 93
5.8 A scatter plot of the average power per input transition against the maximum path delay through the adder. These data were generated from 10,000 MCSTA and SPICE runs for each level of injected variation. The test sequence did not include simultaneous input switching. The shape and location of the distributions match significantly better than the predictions made by Corner Analysis.

5.9 Histograms of the average power per input transition illustrate the close match between the shape and locations of the MCSTA and SPICE Power distributions. The frequency is the number of samples recorded for each power measurement.

5.10 Histograms of the maximum path delay through the adder illustrate the close match between the shape and locations of the MCSTA and SPICE Delay distributions. The frequency is the number of samples recorded for each path delay or power measurement.

5.11 A scatter plot of the average power per input transition against the maximum path delay through the adder. These data were generated from 10,000 MCSTA and SPICE runs for each level of injected variation. The test sequence did not include simultaneous input switching. The shape and location of the distributions match significantly better than the predictions made by Corner Analysis.

5.12 The number of paths identified as having the longest delay through the adder circuit, for a given STA run, increased with variability.

5.13 Power, Performance and Yield plots.

5.14 Transistor model definition which allows Liberty NCX to perform ‘Transistor Mismatch’ cell characterisation.

5.15 Example of a cell netlist that instantiates ‘Transistor Mismatch’ transistor models.

5.16 Modifications to variation netlist made by Liberty NCX for statistical characterisation.

6.1 Transistor diagram for an N-Input NOR Gate.

6.2 Transistor diagram for an N-Input NAND Gate.

6.3 Test Circuits.

6.4 Testbench used for SPICE simulations.

6.5 A plot of simulated path delay variability ($\sigma_{\mu_{\text{path}}}$) against predicted path delay variability (Equation 6.5) for test path Figure 6.3(a). $k$ has been initially set to 1.

6.6 A plot of simulated path delay variability ($\sigma_{\mu_{\text{path}}}$) against predicted path delay variability (Equation 6.5) for the test paths in Figure 3.1. $\sqrt{k}$ is set to the calculated value of 0.0567.
6.7 A plot of simulated path delay variability ($\sigma_{\text{path}}$) against predicted path delay variability (Equation 6.5) for the test paths in Figures 3.1. The values of $n_{\text{stack}}$ have been changed to 1.2 for cells with two transistors in series. 

6.8 A plot of simulated path delay variability ($\sigma_{\text{path}}$) against predicted path delay variability (Equation 6.5) for the test path in Figure 6.3(b), where the number of inputs to the NAND gates are varied. 

6.9 Plot comparing the distribution of propagation delays for the rising transition of a 35nm Inverter (blue) with a fitted Gaussian distribution (red). 

6.10 Cumulative PDFs of delays through the critical path of a 35nm 1bit Adder circuit, using Gaussian models for each individual cell delay. 

6.11 Monte Carlo Static Timing Analysis with Statistical Sampling. 

6.12 The cumulation of the Gaussian models of delays through the critical path of a 35nm 1bit Adder circuit are annotated with the estimated delays of two randomised netlists. The estimated probabilities of obtaining randomised netlists faster than netlist 1 and netlist 2 are 0.57 and 0.98 respectively. Netlist 2 is estimated to be within the top two percent of slowest circuits and may be of some interest to the designer. 

6.13 The Gaussian model of the distribution of delays through the critical path of the 1bit Adder circuit, with the points at which each of the 50,000 randomised netlists appear on the distribution. The distribution is well represented by the 50,000 samples. 

6.14 CDF of the Gaussian model of delays through the critical path of the 1bit Adder circuit, with the points at which each of the 50,000 randomised netlists appear on the distribution. 648 of the samples are estimated to be within the slowest 1% of circuit performance. 

6.15 Histograms of estimated and measured critical path delays for 50,000 randomised samples of the 35nm 1bit Adder circuit. The measured delays are on average 8% slower than the predicted delays. 

6.16 A scatter plot of the estimated delay against measured delay for each randomised Adder netlists. The correlation between the estimated delay and the measured delay indicates that the Gaussian model may be of use for predicting extremes of circuit behaviour before STA. 

6.17 A scatter plot of estimated probabilities against measured probabilities indicates that there is a weak correlation for predictions within the center of the distribution, while there are strong correlations at the extremes of the distribution. 

6.18 The aim of generating a probability estimate with a Gaussian model is to predict where the measured delay appears within the measured distribution. This plot provides a comparison of the cumulative distribution of measured critical path delays through the 1bit Adder against the probability that was estimated for each randomised netlist. 

6.19 Histograms of delays for randomised netlists that were selected for their estimated probabilities. 

6.20 A histogram of delays for randomised netlists that were selected with an estimated probability of over 3-sigma, 0.99865. A plot of the tail of the distribution of delays obtained from 50,000 MCSTA samples is also included.
6.21 CDF of the Gaussian model of delays through the critical path of the Fast Adder circuit, with the points at which each of the 50,000 randomised netlists appear on the distribution. 595 of the samples are estimated to be within the slowest 1% of circuit performance. 

6.22 Histograms of estimated and measured critical path delays for 50,000 randomised samples of the 35nm Fast Adder circuit. The measured delays are on average 26% slower than the predicted delays.

6.23 A scatter plot of estimated probabilities against measured probabilities for the Fast Adder indicates that there is a weak correlation for predictions within the center of the distribution, while there are stronger correlations at the extremes of the distribution.

6.24 Histograms of delays for randomised netlists of the Fast Adder that were selected for their estimated probabilities.
List of Tables

5.1 Power measurements from SPICE, MCSTA and Corner Analysis at 99.7% Yield. Errors are absolute percentage errors with respect to the SPICE predictions ......................................................... 94
5.2 Delay measurements from SPICE, MCSTA and Corner Analysis at 99.7% Yield. Errors are absolute percentage errors with respect to the SPICE predictions ......................................................... 95
5.3 Error between MCSTA and SPICE when generating distributions of delay and power consumption ................................................................. 96
5.4 A comparison of the time taken to perform SPICE and MCSTA simulations 97
5.5 Paths Identified As Having The Greatest Delay ....................................... 98
5.6 Test Circuit Sizes ................................................................................. 101
5.7 Comparison of Path Delays using SPICE and STA, without variability ..... 105
5.8 Comparison of Power Consumption using SPICE and Power Analysis, without variability ......................................................... 106
5.9 Path Delay Statistics ........................................................................... 107
5.10 Power Consumption Statistics .............................................................. 107
5.11 Time required for circuit analysis .......................................................... 110
5.12 Changes in the accuracy of predicted delay distributions as the MCSTA sample size is reduced for analysis of the 35nm Multiplier circuit. Predictions of \(3\sigma\) delays remain to within 2% of SPICE even at a sample size of 10. ........................................................................................................... 111
6.1 Example of Rising Cell Delay through three variation instances of a standard inverter cell ................................................................. 126
6.2 Translation of the variation instances into look up tables of mean and standard deviation for the Gaussian models of the standard inverter cell . 126
6.3 The mean and standard deviation of the delay through a critical path can be obtained from the mean and standard deviations of the delays of the cells within the path. The means \((\mu)\) and standard deviations \((\sigma)\) of cell delays are calculated from the multiple variation instances of each cell within a variation cell library. ........................................................................... 130
6.4 Estimations of delays through the critical paths of two randomised instances of a 35nm 1-bit Adder Circuit ......................................................... 131
6.5 Probability of obtaining a faster critical path delay than the estimations of randomised netlists. ................................................................. 132
6.6 Comparison of the estimated critical path delays with the measured path delays. ................................................................. 135
6.7 Comparison between critical path delays from samples selected by their estimated probability thresholds and critical path delays from the CDF of the 50,000 Sample 1bit Adder MCSTA run. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 140

6.8 The number of randomised netlists generated to find 50 netlists with a probability estimate within the target threshold, and a comparison of the time taken to perform STA on the 50 selected samples rather than all of the generated samples . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 141

6.9 Comparison between critical path delays from the first 50 samples to be generated within a range of estimated probability thresholds and critical path delays from the CDF of the 50,000 Sample MCSTA run. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 142

6.10 Comparison of the Gaussian estimations of critical path delays with the 50,000 sampled and measured path delays, for the 35nm Fast Adder Circuit. 143

6.11 Comparison between critical path delays from samples selected by their estimated probability thresholds and critical path delays from the CDF of the 50,000 Sample Fast Adder MCSTA run. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 145

6.12 Comparison between critical path delays from the first 50 samples to be generated within a range of estimated probability thresholds and the critical path delays from the CDF of the 50,000 Sample Fast Adder MCSTA run. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 145

6.13 A plot of the percentage error in the average delay obtained by a range of sample sizes for a series of different probability thresholds. The percentage errors are with respect to the delay obtained from a distribution of 50,000 samples. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 146
Acknowledgements

I would like to thank my supervisor, Professor Mark Zwolinski, for his support, guidance and endless patience throughout this Ph.D., his wisdom and friendship have been invaluable to me.

My thanks also go to all of the members of the nanoCMOS project, without whom this research would not have been possible, especially the insights of Yangang Wang, Campbell Millar, Scott Roy, Dave Reid, Plamen Asenov, Zhenyu Liu, Steve Furber and Asen Asenov.

Finally my thanks go to my friends and family, especially my wife Josephine and my daughter Evelyn, for their unwavering love and their blind faith in me. I would like to dedicate this work to them, to our unborn child and to any future children we may have.
Chapter 1

Introduction

This report presents an investigation into the challenges of including statistical process variations within the task of designing digital integrated circuits, with a review of the research conducted in incorporating variability within timing and power analysis tools. This investigation has led to the development of a Monte Carlo Static Timing Analysis method, the method, results and critical analysis of which are presented in this report.

The following chapter forms a brief introduction to the document, providing an overview of the research in general as well as contributions made to the research by the author of the thesis, and a declaration that the work is original.

1.1 Motivation For Research

The International Technology Roadmap for Semiconductors (ITRS) is a report sponsored by the five leading chip manufacturing regions in the world (Europe, Japan, Korea, Taiwan and the USA). The objectives for the roadmap are to provide a summary of the current problems facing the industry, and an industry-wide consensus on the required paths of research and development to overcome these problems [1]. A brief overview of some of these problems is provided here, as they demonstrate the significance of this project.

The largest threat to the continuation of the industry, as identified within the roadmap, is the cost of designing new systems. The expense of creating the masks for the production of a modern integrated circuit (IC) is usually in the region of millions of dollars, while the costs of the design process is regularly reaching tens of millions of dollars. Shortfalls within a project can force the repetition of the manufacturing process, multiplying the mask generation costs. This cost of re-spinning a design puts pressure on engineers to fully verify their system before passing it on to production, which is countered by strong time-to-market demands, generated by short product life cycles. Changes in technology
have allowed for major developments within the semiconductor industry, including the scaling of transistor sizes down to tens of nanometres. It has been noticed, however, that the efficiency of the tools and methodologies used within the design industry is not keeping pace with this scaling, and that the complexity of new systems is increasing exponentially.

The accuracy of these design tools and methods has also been affected by the scaling of transistor sizes. The number and placement of individual dopant atoms within the silicon produce differences in the performance of individual transistors, and these affects are increasing in significance. The random physical differences can not be removed from the manufacturing process and so must be efficiently modelled and incorporated into the design process.

The nanoCMOS project [2], funded by the Engineering and Physical Sciences Research Council (EPSRC) [3], has been created as a collaborative effort between academic and industrial partners to meet the design challenges caused by random variations within nano-CMOS electronics.

1.2 Aims

The goal of the nanoCMOS project was to provide accurate and efficient predictions of the affects of manufacturing process variations on the power, performance and yield of designs. The objectives and contents of this thesis are strongly aligned with the goals of the nanoCMOS project, which is arranged as follows.

The first objective is to model the impact of statistical manufacturing process variations on standard cells, by providing a transparent and repeatable method of cell characterisation that makes use of commercially available tools and fits within the existing design flow.

The second objective is to use the characterised cells to predict the impact of statistical process variations within circuits, providing design engineers with a clear method of generating the critical information that will allow a design to be signed off for manufacturing.

The third objective is to demonstrate the accuracy of the proposed methods, illustrating the benefits over existing industry standards and alternative statistical modelling methods, and demonstrating ways in which power, performance and yield trade-offs can be made.

The final objective is to illustrate that the proposed methods not only allow for a final design signoff against process variations, but also allow for practical, rapid predictions to be made throughout the design flow, allowing early design decisions to be made that
can increase the tolerance of a design to process variations and reduce the possibility of costly design re-work.

1.3 Thesis Outline

Chapter 2 of this thesis begins with a discussion on the complexities of designing digital circuits using modern process technologies. The added challenges that are posed by manufacturing process variations are introduced, and a literature review of the methods used to predict the performance of modern digital circuits is provided, with a summary of why Statistical Static Timing Analysis (SSTA) has not yet been widely adopted by the industry and why a modified Monte Carlo approach could provide a more reliable and transparent approach.

Chapter 3 builds upon the concept of using Monte Carlo simulations by introducing Monte Carlo Cell Characterisation (MCCC), where the process of cell characterisation is described, and a method of incorporating statistical process variation is demonstrated. The statistical behaviour of a range of standard cells is analysed and characterised using the industry standard Liberty format, completing the first objective of this thesis with the generation of Variation Cell Libraries.

Chapter 4 focuses on the second objective of the thesis by demonstrating a novel method of propagating the performance distributions within Variation Cell Libraries through to gate level netlists. This proposed method is referred to as Monte Carlo Static Timing Analysis. The chapter introduces some of the standard industry practices of timing and power analysis for digital circuits, describes how the practices can be altered to incorporate variation data, and provides a comparison against Monte Carlo SPICE circuit simulations.

Chapter 5 demonstrates the accuracy of Monte Carlo Static Timing Analysis (MCSTA), comparing the proposed method with alternative methods such as Corner Analysis and Statistical Static Timing Analysis. This chapter focuses on the third objective of the thesis, with predictions of the impact of statistical process variations on the power, performance and yield of digital circuits.

Chapter 6 describes an investigation into a proposed metric [4] for predicting delay variations through logic paths, and also provides a description of how simple statistical sampling methods can be used to enhance the process of MCSTA. The proposed statistical sampling method allows a designer to rapidly target an area of interest within the performance distribution of a circuit and perform MCSTA on only the relevant samples, which dramatically reduces the analysis time required, provides a practical alternative to SSTA, and meets the final objective of this thesis.
Chapter 7 contains a summary of the findings described within the thesis and provides suggestions for future research that continues this work.

1.4 Contributions

This document provides a summary of the affects of variability on the design and manufacture of CMOS circuits, reviewing the ongoing research in the development of statistical analysis tools that attempt to predict these affects. Work on the modelling of these affects, in the form of a Monte Carlo Static Timing Analysis method.

1.5 Declaration

This thesis describes the research undertaken by the author while working within a collaborative research environment, and documents the original work of the author unless stated otherwise.
Chapter 2

Background - The Significance of Process Variation

The increase in the size and complexity of designs requires large improvements in design abstraction methods, while the significance of process variations now requires an increase in design refinement. Major changes must be made within the electronics design industry in order to counter what the ITRS roadmap claims is the ‘superexponentially increasing complexity of the design process’, caused by the combination of system design and device manufacturing challenges. This chapter is split into two sections, the first of which provides an introduction to complications created by the physical scaling of transistors, the second section provides an overview of the current system design complexities and how these are further compounded by those of the first section.

2.1 Silicon Complexity

2.1.1 Transistor Scaling

The semiconductor industry has been heavily focused upon following Moore’s Law since the term was coined during the 1980s. This principle was based on Gordon Moore’s observations in 1965: that the number of transistors per square inch would most likely double each year over a 10 year period [6]. Moore’s original observation was based on a four year trend which he extended to form a 10 year prediction, and in 1975 this prediction was found to have been surprisingly accurate. After these ten years however, a major recession and new manufacturing challenges within the industry caused Moore to modify his observations from that point onwards, slowing his prediction of growth to the doubling of components per chip every 2 years [7]. More recently Moore’s law is being misquoted as being the doubling of chip performance every 18 months, and while the specifics of this misnomered law and how long it may last are being discussed heavily
within the literature, the key point seems to be that the majority of companies feel that they must meet this target to remain competitive within the industry [8] [9].

This drive to comply with Moore’s law, with the aim of increasing productivity and profitability, has resulted in forty years of scaling down the sizes of devices and has pushed the process widths of CMOS technology to the nanometre level. Figure 2.1 displays the increase in the number of transistors per processor manufactured by Intel from 2,300 in 1971 to 1.9Billion in 2008, this has been made possible by their reduction in process widths from 10um to 45nm over the same period, allowing more transistors to be fabricated within the same area. The period of scaling up until the late 1990’s is often referred to as the period of ‘happy scaling’, the reasons for this are described well within [10] which notes that breakthroughs in technology and the manufacturing processes (driven by the improving power of personal computing) allowed for a combination of the following six factors:

1. **Reduction in Transistor Size:** This lowered the cost per transistor, as more transistors could be fabricated on a single silicon wafer.

2. **Retention of the Transistor Structure:** The transistor remained similar during the scaling process, simply becoming smaller, allowing much of the same circuit designs to be reused.

3. **Increased Clock Frequency:** The smaller devices could operate at higher switching speeds, improving the clock frequency without having to redesign the basic processor architecture.

4. **Retention of Digital Operations:** Design methods and tools could remain the same, as the abstraction of electrical signals into ‘1’s and ‘0’s could continue.
5. **Low Power**: The efficiency of the circuits improved as the total energy per function of the circuit decreased.

6. **High Yield**: The percentage of correctly functioning manufactured chips was primarily determined by the quality of the fabrication process, which continued to improve.

The challenge for the semiconductor industry is now that this combination of factors can no longer be so easily maintained. Factors such as leakage currents and process variations are having an increasing impact on the power, performance and yield of circuits. These are discussed in the remainder of this section.

### 2.1.2 Power Consumption

The reduction of chip power consumption is currently one of the largest technical problems within the semiconductor industry, having previously being regarded as secondary to factors such as speed and silicon area. The principle advantage of CMOS circuits over competing transistor technologies has been low power consumption, but increasing transistor counts and clock speeds have caused a large surge in dynamic power, while static power contributions have also increased due to the rising dominance of leakage currents. The total power dissipated by CMOS circuits is a combination of the Static (idle) and Dynamic (switching) power dissipations: equation 2.1.

\[
P_{\text{total}} = P_{\text{static}} + P_{\text{dynamic}} \tag{2.1}
\]

#### 2.1.2.1 Dynamic Power

The main component of dynamic power consumption within CMOS circuits is the power used to charge and discharge the load capacitance of any switching transistors within the circuit. For modelling purposes this is often simplified to equation 2.2, where \(A\) represents the proportion of transistors switching per clock cycle; \(C\) is the total capacitive load of the gate outputs; \(V\) is the supply voltage and \(f\) is the clock frequency.

\[
P_{\text{dynamic}} = ACV^2 f \tag{2.2}
\]

Smaller contributions to dynamic power loss are the momentary ‘short circuits’ between the supply voltage and ground rails while both pull up and pull down networks are enabled for a brief period during the transition of the gate output. This effect decreases with increases in load capacitance and decreases in input transition times [11].

The large number of transistors in modern circuits, combined with high clock speeds, has threatened to produce excessive power and heat management requirements. This is
being addressed by efforts from both manufacturing and design sectors of the industry, as a combination of the scaling down of the supply voltages within circuits and the use of low power circuit design techniques have helped to keep this in check. These circuit techniques include the use of a combination of parallel and pipelined circuit implementations, instead of increasing the clock frequency, to improve system performance [12].

![Figure 2.2: Schematic of a CMOS Inverter](image)

### 2.1.2.2 Static Power

The complimentary use of pMOS and nMOS transistors as pull-up and pull-down networks within CMOS gates was engineered such that when the inputs reach a steady logic state (1 or 0), only one set of transistors (either the pull up or pull down network) is enabled. The CMOS inverter shown in Figure 2.2 is provided as an example; if the input A is ‘0’ then the pMOS transistor is ‘on’ and the nMOS transistor is ‘off’ pulling the output Y up to the supply voltage Vdd. Conversely if the input A is ‘1’ then the nMOS transistor is ‘on’ and the pMOS transistor is ‘off’ pulling the output Y down to ground. Ideally, an ‘off’ transistor would not draw any current, and for many years the actual amount of current drawn was negligible. Unfortunately the effects of scaling down the sizes of transistors has meant that the behaviour of real transistors has become less than ideal, with the increasing prevalence of leakage effects such as sub-threshold conduction and oxide tunnelling. Details of these leakage mechanisms and their impacts on CMOS circuits are given by [13], from which a brief summary and Figure 2.3 is presented here:

1. **Sub-threshold Current**: In real transistors the current flow does not stop instantly when the gate voltage drops below the threshold voltage, the current instead drops exponentially as the input voltage decreases. This sub-threshold conduction has
increased exponentially with both decreases in threshold voltages that have come with the scaling down of transistor sizes, and also with increases in operating temperature which can lead to complications with self heating power consumption.

2. Oxide Tunnelling: Silicon oxide is used as the gate insulator in transistors due to its good insulating properties, but these properties decrease significantly as the oxide thickness is reduced. Electrons are able to tunnel through thin layers of oxide, with the probability of this increasing exponentially with decreases in oxide thickness.

3. Reverse bias diode leakage: The p-n junctions within the transistors form diodes which are kept in reverse bias. Reverse biased diodes conduct small amounts of current, but these amounts are negligible compared to the other leakage effects.

4. Hot Carrier Injection: Electrons can gain enough energy from the electric field near the silicon and oxide interface to cross from the substrate into the oxide layer.

5. Gate-Induced Drain Leakage: High electric field effects can cause minority carriers to be emitted in the drain region under the gate, these are then drawn into the substrate creating a current flow.

6. Punch-through: The drain-substrate and source-substrate depletion regions can extend into the transistor channel in small devices, this extension grows with increases in the reverse bias across the junctions. Reductions in channel length combined with this effect can cause the two depletion regions to overlap, which is referred to as punch-through.

![Leakage Current Mechanisms in Nanometre Transistors](image)

Figure 2.3: Leakage Current Mechanisms in Nanometre Transistors [13]
The effects of these mechanisms are increasing with the continuation of device scaling, and have a significant impact on the efficiency of circuits. This has forced IC designers to become more aware of the impact that design decisions have on power consumption. Such design decisions can range from: simple architectural concepts, such as encoding address lines with Gray code to reduce the number of simultaneously switching signals; to transistor level alterations that use low threshold voltage devices for critical high speed sections, and high threshold voltage devices for non critical blocks [14][15]. The impact of such design decisions can not always be predicted, as the information required to accurately calculate the power consumption of an IC is usually only obtainable after a significant proportion of the design has been finalised. This problem is further compounded by the impact of device variability, which reduces the accuracy of these calculations.

2.1.3 Variability

The term variability is used to describe the factors which can alter the performances of fabricated circuits which were intended to be identical. There are three main categories of variations: Environmental, Temporal and Process.

**Environmental:** Environmental factors are external to the IC, such as fluctuations in ambient temperature or supply voltage. Designers may have to characterize their device for extreme temperatures, or ensure that performance does not alter significantly over short term variations in the supply voltage.

**Temporal:** These describe how a device may change over time, possibly resulting in a degradation in performance or efficiency. These factors are important for the estimation of device lifetimes.

**Process:** Variations within the manufacturing process can lead to fluctuations in features such as: oxide and metal thickness; device dimensions; and dopant concentrations. These features control the performance characteristics of the device, and can alter from wafer to wafer, between chips on a single wafer, and even across an individual chip. The effects of system-wide variations, both global and local, can be successfully predicted using current software, but the reduction in transistor sizes has led to an increase in the effect of local random or statistical variations. The unpredictability of these variations from transistor to transistor within a chip is now having a major effect on the semiconductor industry.
2.1.3.1 Local Statistical Process Variations

Transistors are now so small that they contain features with dimensions of tens of nanometres. At this scale it is no longer suitable to assume that the p-type and n-type areas of silicon that form the transistors are regions of continuous charge, nor is it suitable to assume that the boundaries and interfaces between these regions are perfectly smooth. The positioning of individual atoms becomes more significant as the total number of atoms within the device decreases, the effects of which are described within [16], from which a summary is given here. Figure 2.4(a) shows the conventional continuous method of modelling a transistor, and does not take into account the atomistic, but significant, differences that occur between transistors. The transistor sketch shown in figure 2.4(b) is a predicted model of a 20nm transistor. At this size there are approximately only 50 silicon atoms across the length of the channel, and random variations in both the number and the discrete positions of dopant atoms (RDD) within the active region will create substantial variations in the characteristics of the device. The granularity of the materials in use also produces unavoidable line edge roughness (LER), each individual transistor having significantly varied geometries and oxide thickness.

The random physical differences between transistors within a chip are not yet modelled effectively by timing analysis tools, leading to significant discrepancies between the predicted performance of a synthesised netlist and the actual performance of the device in silicon.

2.2 System Complexity

The ITRS roadmap [1] has noted the development of a ‘design productivity gap’ that has emerged between the manufacturing and design sectors of the industry. Improvements in design methods and tools have not kept pace with those in manufacturing, as the increase in the number of transistors designed per day is not keeping pace with the increase in transistors available per day. The roadmap uses the term ‘system complexity’ to refer to this situation; exponentially increasing transistor counts enabled by process scaling, demands for lower cost and increased functionality. There is also an enormous pressure to reduce the time it takes for a design to be released in the marketplace.

This section of the report documents a brief overview of Electronic Design Automation (EDA) tools and processes, the principles used to improve the efficiency of the processes, and the incorporation of device variability into these tools.
2.2.1 Electronic Design Automation

Integrated circuits were originally designed and simulated by hand, using rolls of graph paper and timing diagrams. This process was sufficient for circuits consisting of tens of gates, but the success of the IC and the growth of the industry meant that this number
rapidly grew to beyond the scale of a few pieces of paper[17]. The article, [18], provides an overview of the development of EDA tools, noting that their evolution centres on increasing the levels of abstraction used to describe complex systems. This has most notably included the generation of physical design tools and standard cell libraries in the early 1980s; the ability to synthesise Register Transfer Logic (RTL) descriptions in the early 1990s; and the widespread use of Intellectual Property (IP) blocks in the early 2000s.

A simplified design flow is shown in Figure 2.5, and the design methods used within this flow are described below:

1. **Specification** This is the reference against which the performance of the design is judged. The ability to convert handwritten project specifications into forms used by verification or synthesis tools is one of the main aims of the EDA industry.

2. **Behavioural Description** At this level algorithms are used to describe the system function or behaviour. The description can be passed through simulators to check the behaviour, and formal verification tools can be used to prove the correctness of the algorithms. Sections of the Hardware Description Languages (HDLs) used at this level can be converted to RTL using synthesis tools.

3. **Register Transfer Level Description** The system is described as a number of registers and their corresponding transfer functions. These descriptions are more detailed and take longer to simulate, equivalence checking tools are therefore used to verify that the function matches that of the level above. This description can then be passed to synthesis tools.

4. **Synthesis** A gate level description of the system is generated by synthesis tools, which map the RTL description onto a chosen library of cells. The function, size and performance of these cells reflect the characteristics of the chosen fabrication process. An estimation of the timing and power usage of the circuit can then be generated from the information within the cell libraries.

5. **Place and Route** The gates and their interconnecting wires can be placed and routed automatically, and their positions and spacings checked against the design rules of the chosen fabrication process. The lengths of the wires are combined with the cell library information to generate timing and power calculations. If these are acceptable then the design can be passed on for fabrication. The resistive and capacitive effects of the interconnecting wires are referred to as parasitics.
2.2.2 Closing the Productivity Gap

The efficiency of the design flow must be improved in order to increase design productivity and resolve the issues of system complexity. The work on this area can be divided into four key areas: abstraction, synthesis, reuse and verification.

2.2.2.1 Abstraction

Reducing the information contained within a system description allows for faster simulations and verification times [19]. The development and widespread use of System level descriptions (a level above behavioural) will allow for the description of systems in terms of information transactions, removing details such as clock cycles, bus widths and communication protocols. These principles have been within the literature for a
significant amount of time [20], and have been incorporated into popular system level
description languages such as SystemC and System Verilog.

2.2.2.2 Synthesis

The automated synthesis of RTL descriptions produced a large boost to productivity in
the 1990s. The ability to generate hardware directly from behavioural or system level
logic would remove large sections of the design process, and further improve productivity
[18].

2.2.2.3 Reuse

The simplest way to avoid design work is to reuse existing designs. This is the concept
behind the surge in use of reusable IP blocks over the last decade. Creating IP for reuse
requires well designed code that allows the block to be both portable and parameterisable
[21].

2.2.2.4 Verification

It is a fundamental requirement that a design is verified before it is manufactured, in
order to avoid heavy costs, but the task of verification has become more time consuming
than creating the design itself. The verification process can be separated into three key
areas:

1. Functional Verification Simulations used to be the sole method of testing designs,
but simulating every possible input sequence for a design can take more processing
time than is available for the entire project, even at the highest modelling levels
[22]. It is therefore standard practice to use a limited number of input patterns in
a trade-off between design confidence and computation time. The choice of these
input patterns can be simplified by the use of code coverage metrics, which show
how many lines of system code have been left untested by a simulation. This can
be combined with the use of Assertion statements, an exception handling method
that can halt simulations and highlight errors within the design. The develop-
ment of these tools is simplifying the generation of comprehensive test-benches,
and removes the task of manually examining simulation waveforms [23]. Formal
verification methods such as model checking, allow an exhaustive exploration of all
possible behaviours of a system, removing the need for the selection of appropriate
test vectors, but rely instead upon the generation of correct models [22].
2. **Physical Verification** This involves checking parameters such as the thickness and separation of wires within the placed and routed circuit layout. Design rule checking (DRC) tools ensure that the design follows the rules of the chosen fabrication process. These rules are created by the manufactures, taking systematic process variations into account: sufficient spacing must be left between wires, this ensures that systematic changes in metal widths will not cause the separate wires to merge. The use of standard cell libraries issued by the chosen manufacturer, or reused by designers with their own fabrication facilities, dramatically reduces the complexity of this task. Layout versus schematic (LVS) tools are formal analysis tools that ensure the layout correctly represents the circuit that it was synthesised from [12].

3. **Timing Verification** Timing analysis tools are used to predict the performance of the fabricated circuit, using information stored within cell libraries and the circuit layout. The accuracy of these tools have traditionally been sufficient as a final sign off for design to be manufactured. This accuracy is now being severely affected by the increase of statistical variability within the manufacturing process, which is reducing both the performance and yield of fabricated devices [24].

Accurate timing and power analysis is required to indicate that a design is ready to be fabricated. Inaccuracies between the performance of a fabricated design and the circuit simulations can lead to the expensive process of redesign, and re-fabrication [25]. The ability to overcome the challenges of increasing variability depends upon the ability of designers and manufacturing process developers to work together [26]. Timing analysis is a key stage of the design flow in which statistical process variations must be successfully modelled, and so the following section of this thesis discusses the development of modern timing analysis tools.

### 2.3 Literature Review - Shortfalls in Current Circuit Analysis Methods

A significant amount of effort has been expended on improving the accuracy of circuit analysis, which is justified by the large expense incurred when mistakes are made during the design process which are only discovered once a circuit has been manufactured. The analysis of the performance of a circuit is not simply used to verify that the circuit will operate correctly, but can also be used within the marketing of the circuit and establishing the costs of an entire project. The industry can not (in general) afford to find out how well a circuit will perform after manufacturing and then attempt to market the product, instead the performance and power consumption of the circuit must be known so that the prices of the products and the number to manufacture can be established. In some cases circuit analysis may be used to establish whether there is any point in creating the design on silicon at all.
The pressures of getting a product to market within a short time scale, increases in circuit complexity and the significance of manufacturing process variations have driven the vast amounts of research into modern circuit analysis tools. This chapter provides a review of some of the key steps within this research, and provides some explanations as to why circuit analysis is still a hot topic within research.

### 2.3.1 Timing Analysis

The main aim of timing analysis is to assess whether a circuit will operate correctly at a certain operating frequency, or range of operating frequencies. This typically does not test whether the logical function, or behaviour of the circuit is correct (as this is usually asserted beforehand), but instead checks whether it is physically possible for the circuit to operate under a set of timing constraints. Timing constraints include the frequency at which synchronous systems are clocked, and the expected delays or latencies of asynchronous inputs. An example of this is where combinational logic exists between an input to a circuit and the input to a sequential element (such as a register), Figure 2.6(a), where the delay through the combinational logic must be short enough to allow the signal to remain in a stable state for a period before and after the next active clock edge, Figure 2.6(b). If the combinational logic delay is too long then the setup time of the sequential element may be violated, and the circuit will not be able to function at the required clock speed, whereas if the combinational logic delay is too short then the hold time may be violated, and the circuit will not function, even if the clock speed is reduced. Logic paths that are most likely to fail either setup or hold timing constraints are referred to as the critical paths of the design, because if these paths are corrected then the performance of the entire circuit can be improved.

Timing analysis plays a fundamental part in the assessment of the performance of a circuit, and is used during the signing off of a design, where it is allowed to proceed for manufacturing. The accuracy of the analysis is critical for high performance and complex designs, as the failure of a single logic path may render a whole circuit completely useless. The literature contains a large amount of work on the improvement of timing analysis, each presenting differing algorithms and statistical methods for achieving the required levels of accuracy when modelling variation. A brief summary of some of the developments within this process is given here.

#### 2.3.1.1 Static Timing Analysis

The word static in this case refers to the fact that this analysis is performed without applying test signals to the circuit. The first Static Timing Analysis (STA) tools were developed in the late 1970s to replace timing simulations, and at first these consisted of algorithms that searched for the longest and shortest paths through the combinational
logic. Early attempts were computationally expensive and tended to fail with (then) complex circuits. Verification programs such as the SCALD timing verifier, described in [27], were introduced to allow designers to verify that the setup, hold and pulse width requirements for circuits were satisfied. These developments simplified the timing analysis of circuits, but each timing tool required the designer to learn and master a different HDL [28][29].

2.3.1.2 Timing Slack

The Timing Analysis program, described within [30] and [31], propagated delays through a circuit to provide a worst case delay. It also introduced the concept of timing Slack, a measure of the severity of a delay. This concept has allowed designers to focus on sections of a design with negative slack, that have not met the timing constraints Figure 2.7(a), or to relax sections of the design that had been over engineered, with positive slack Figure 2.7(b). This form of feedback, which can be used to direct the attention of designers and automated design tools, has now become a standard requirement for these forms of verification tools [32].
2.3.1.3 Delay Modelling

STA allows each delay element within a circuit (gates and wires) to be replaced with a single delay value. This delay value is interpolated from a look up table within a cell library in which the delay elements (cells) have been characterised for a fixed set of process corners. Timing analysis tools, such as those developed by Synopsys, model the delay and output transition time for the element to be obtained for a given input transition time and output load capacitance.

2.3.1.4 Corner Analysis

Traditionally multiple cell libraries are generated during the cell characterisation process, e.g., Fast (best case), Typical, and Slow (worst case). The devices are characterized for
chosen combinations of various process and environmental parameters. These combinations are typically set at the extremes or corners of each parameter, hence this process is referred to as Corner Analysis. The use of the Slow libraries is a method of testing the performance of the circuit at what is hoped to be the worst case, with the aim of ensuring that every single manufactured circuit will not violate the Setup timing checks, while the use of Fast libraries ensures that the absolute fastest circuit will not violate the Hold timing checks. Figure 2.8 displays a simple example in which two process parameters are chosen. Three libraries have been generated from the expected best, worst and nominal Supply Voltage and Operating Temperature values. The performance of the circuit under test is analysed at each of these three corners by performing STA with each of the three generated libraries. Although it is known that the performance of the cells within the libraries does not depend linearly upon the chosen parameters, it is assumed that the performance of the device is guaranteed at any point within the box created by these corners [12]. Corner Analysis is still the most widely used method of signing off a design for manufacturing in industry, but the number of parameters required to generate the libraries is increasing as the scaling of transistors continues, as factors such as changes in transistor length; threshold voltage and oxide thickness must be included. The use of best and worst case libraries from large numbers of process parameters reduces the likelihood of the corner cases ever actually being manufactured, which means that excessively large amounts of time and money can be spent on designing systems to meet unrealistic criteria.

![Figure 2.8: Three libraries generated for corner analysis](image-url)
2.3.1.5 Statistical Static Timing Analysis

While the probability of each system wide process parameter being at either their best or worst at the same time can be very small, the probability of the localised process parameters having the same value for every cell instance within a circuit is minute. STA does not currently allow the selection of minimum delays for some gates and maximum delays of others. Corner analysis is therefore suitable for testing systematic process variations, but not variations within a single chip. Random variations within the manufacturing process may cause some gates to operate slower while neighbouring gates operate faster. This is clearly explained within [33], where it is highlighted that STA can be both overly pessimistic and optimistic at the same time, an expensive waste of design time. [33] provides a clear description of the theory of the statistical delay calculations, from which an early form of Statistical Static Timing Analysis (SSTA) is created. The basic principles discussed within this paper (in 1997) are present in much of the current literature, and so a summary of the paper is given below.

Figure 2.9: SSTA Delay Element, adapted from [33]

Within [33] Figure 2.9 is used to describe the basic requirements of SSTA, where $T_1$ and $T_2$ are the arrival times of signals at the inputs of a 2 input NAND, $t$ is the propagation delay of the gate, and $T_{out}$ is the total delay. Traditional STA determines the total delay by evaluating $T_{out} = \max(T_1, T_2) + t$, which requires two basic operations, finding the maximum of $T_1$ and $T_2$ and adding $t$. These basic operations become more complicated when applied to statistical distributions instead of single values. The assumption is made within [33] that gate delays are normally distributed, which simplifies the adding operation as if $A$ and $B$ are normally distributed with means $\mu_A$ and $\mu_B$ and standard deviations $\sigma_A$ and $\sigma_B$ then $C = A + B$ gives a normal distribution $C$ with mean $\mu_C = \mu_A + \mu_B$ and standard deviation $\sigma_C = \sqrt{\sigma_A^2 + \sigma_B^2}$.

[33] continues by pointing out that the maximum calculation becomes more complicated. If $C = \max(A, B)$, then for any value $x$ it can be written:

$$P(C \leq x) = P((A \leq x) \cap (B \leq x))$$

The assumption is then made that $A$ and $B$ are statistically independent, and so the equation is re-written as equation 2.4, where the probability of $C$ being less than $x$ is
the multiple of the probabilities of $A$ and $B$ being less than $x$:

$$P(C \leq x) = P((A \leq x) \cdot (B \leq x))$$ (2.4)

The probability density functions (PDFs) of the distributions are then represented using the notation $f_A$ and the cumulative distribution functions (CDFs) are represented by $F_A$. This can be written $P(A \leq x) = F_A(x) = \int_{-\infty}^{x} f_A(x) \, dx$, which when taking the derivatives for the left and right of equation 2.4 gives:

$$f_C(x) = F_A(x) \cdot f_B(x) + F_B(x) \cdot f_A(x)$$ (2.5)

This means that to obtain the PDF of $C$ the CDFs and PDFs of both $A$ and $B$ must be known, and it must be possible to express these functions in a closed form to allow them to be multiplied. It is then highlighted within [33] that $f_C(x)$ is not a normal distribution even if $f_A(x)$ and $f_B(x)$ are, and that $F_A$ and $F_B$ cannot be expressed in closed form for normal distributions. It is instead suggested, with experiments and results, that $f_C$ can be approximated to a normal distribution.

The weaknesses of this early proposal for SSTA include the assumptions of normally distributed gate delays and approximations to normal distributions for propagated delays, which were based upon theorised gate delays that were not obtained from real manufacturing data. However the most significant weakness, as noted within [33], is the assumption that each of the gate delays are independent, and that correlations between delays are not modelled. The removal of these assumptions is non-trivial and remains a key focus within the literature.

2.3.1.6 Delay Correlations and False Paths

It is noted within [34] that circuits with re-converging paths have significant correlations between delays, and that timing analysis methods that ignore them may be inaccurate. Figure 2.10 is an example of a circuit containing converging paths, where the grey box with output C is simply a delay element: If the delays of signals x and y depend heavily on the delay of signal c, then the correlation of delays x and y must be taken into account when calculating the maximum delay at z. The inclusion of these correlations improves the accuracy of timing analysis, but has a significant increase on the computation time.

A related paper [35] also describes inaccuracies caused by the inclusion of false paths within the timing analysis. A false path is described as a path from a primary input to a primary output which cannot be activated by any input vector. This can include input vectors that will never be used within the lifetime of the circuit, or paths that will never depend on the delay of an input. Techniques to assist in the removal of these paths were incorporated within the previous algorithm for including correlations between delays.
The ability to set certain paths within a design as being false paths, which should be excluded from the timing analysis, has become a vital tool in timing guided optimisation tools. If a synthesis tool or a place and route tool is trying to change sections of a design in order to improve performance, then it may be forced to degrade other less critical areas. If a false path is not highlighted to the tool by a designer then significant amounts of unnecessary effort and optimisation may be performed on areas of the design that appear critical to the tool, but are not required to operate at maximum frequency, often at the expense of performance and area within the rest of the design. An example of such paths may be areas of a design that have been included for testing purposes, such as scan chains and the logic that switches between testing mode and normal operation. If the normal operation of the circuit is performed at much higher speeds than the testing procedures, then the results of the design automation tools may be improved if the testing logic is marked as false paths within the design.

### 2.3.1.7 Inter-Die and Intra-Die Variations

Many papers within the literature refer to different classes of process variations and use the terminology inter-die to refer to variations between chips on a wafer and intra-die to refer to variations within a single chip. The word die comes from the process of dicing a wafer into the separate circuits that have been fabricated on the surface of the wafer. An early reference to these classifications of process variations is made within [36], where a distinction is made between two forms of SSTA, Full-Chip analysis and Path-Based analysis. Full-chip analysis models the delay of a whole circuit as a random variable, and attempts to compute the probability distribution of the entire circuit, which becomes complicated when taking re-converging paths into account, and leads to very high run times. The proposed path based approach within [36] first performs traditional static timing analysis on a circuit, and records the worst \( n \) critical paths, each of these paths are then individually statistically analysed to produce distributions of each path delay.
Care must be taken to ensure that \( n \) is sufficiently large enough to allow all paths that could become critical to be examined. [36] continues with a discussion on how both inter-die and intra-die process variations can be included within their path based approach, the principles of which are found in many modern approaches and so a summary is given here.

A model for process parameters is proposed within [36], with an example given for a device \( i \) where the device length \( L_{\text{total},i} \) is the sum of inter-die device length \( L_{\text{inter}} \) and intra-die device length variation, \( \Delta L_{\text{intra},i} \):

\[
L_{\text{total},i} = L_{\text{inter}} + \Delta L_{\text{intra},i}
\]  

(2.6)

where \( L_{\text{inter}} \) and \( \Delta L_{\text{intra},i} \) are random variables with normal distributions.

All devices on a die (or chip) share the same value of the inter-die parameter (e.g. \( L_{\text{inter}} \)), which is set to the mean value for the chosen die. A separate independent random variable is then assigned to each device \( i \) for the intra-die parameter (e.g. \( L_{\text{intra},i} \)), but all of the random variables for the parameter have identical probability distributions. The total variation \( L_{\text{total}} \) and inter-die variation \( L_{\text{inter}} \) have a mean that is equal to the nominal value of the device length, but the intra-die variations have a mean value of zero. The assumption is made that all three random variables have a normal distribution, which is stated to be a common assumption for physical quantities. It is noted within the paper that the gate delays do not themselves have a normal distribution, because gate delay is a non-linear function of the device length.

An analysis of device length measurements from test die on manufactured wafers is presented in [36]. Where the intra die standard deviation for each type of structure on each die was computed, and the standard deviation of \( \Delta L_{\text{intra},i} \) was set to be equal to their average. \( \Delta L_{\text{intra},i} \) represents the device length deviation from the chip mean, which is why the variable has a mean of zero. Given the distributions of \( L_{\text{total}} \) and \( \Delta L_{\text{intra},i} \) the standard deviation of \( L_{\text{inter}} \) was calculated from the following equation:

\[
\sigma^2 L_{\text{total}} = \sigma^2 L_{\text{inter}} + \sigma^2 L_{\text{intra}}
\]  

(2.7)

The standard deviation of the total variation is equal to the sum of the standard deviations of the inter-die and intra-die variations, which assumes that the inter-die and intra-die variations can be modelled with two independent random variables.

Having established a method of modelling the process variation of a device parameter, the objective was to obtain the distribution of the delay through the path \( D_p \) from the total variation of the device parameters within the individual gates in the path.

\[
D_p = \sum_i (D_i(L_{\text{inter}} + \Delta L_{\text{intra},i}))
\]  

(2.8)
where $D_i$ is the delay of the $i^{th}$ gate within the path, as a function of its device length and the sum is taken over all gates in a path. The path delay $D_p$ is a random variable for which the distribution is hard to calculate, because $D_i$ is a non-linear function of device length, and the distribution cannot be accurately expressed in a closed form. This is the same fundamental issue that was discussed within Section 2.3.1.5, as not being able to represent distributions in closed forms makes the calculation of the maximum of or addition of multiple distributions non-trivial, and these are necessary operations to perform in the propagation of the maximum gate delays through a path. It is noted within the paper that Monte Carlo SPICE simulations of the entire path could be performed to compute the distribution of $D_p$, but that this would lead to unacceptable run times. Instead the following assumption is made within [36] for simplicity:

$$D_i(L_{inter} + \Delta L_{intra,i}) = D_i(L_{inter}) + \Delta D_i(\Delta L_{intra,i})$$ (2.9)

where $\Delta D_i(L_{intra,i})$ is the change in the gate delay due to small changes in the device length. The gate delay of the sum of the inter-die and intra-die lengths is now approximated by the sum of the delays of the inter-die and intra-die variations. This means that $\Delta D_i$ is assumed to be independent of $L_{inter}$, which it is claimed is valid if $\Delta L_{intra,i}$ is small when compared with $L_{inter}$. These assumptions allowed the calculation of $D_i(L_{inter})$ and $\Delta D_i(\Delta L_{intra,i})$ to performed independently and then combined to produce the total path delay distribution $D_p$:

$$D_p = \sum_i D_i(L_{inter}) + \sum_i \Delta D_i(\Delta L_{intra,i})$$ (2.10)

$D_i(L_{inter})$ is determined within [36] by simply enumerating the distribution of $L_{inter}$, as each of the gate delays due to inter-die variations are modelled by a single random variable. The distribution of $L_{inter}$ is replaced by 20 device lengths between the worst case and best case process corner values, requiring that each path is simulated 20 times to determine $D_{p,inter}$.

The enumeration process would require far too many simulations for the determination of $D_{p,intra}$, as this portion of the path delay is a function of multiple independent random variables. Instead a second simplifying assumption is made, that $\Delta D_i(\Delta L_{intra,i})$ can be approximated linearly so that the change in $D_i$ due to intra-die process variations is equal to the sensitivity of $D_i$ to the process variable multiplied by the change in the process variable:

$$\Delta D_i(\Delta L_{intra,i}) = \frac{\delta D_i}{\delta L_{intra,i}} \times \Delta L_{intra,i}$$ (2.11)

This simplification allows for the sensitivities of gates to process variations to be pre-computed, and used in a very simple calculation to produce changes in path delay. This concept of pre-characterising the sensitivity of gates to process variations before timing analysis is performed remains a key feature within the literature and is used within some
of the industry leading timing analysis tools. The sensitivities of gate delay to process variations are combined with sensitivities to the slope of the input transition to the gate and the output load capacitance of the gate.

2.3.1.8 Incremental Analysis

Various papers have been published by IBM with the aim of improving the accuracy and efficiency of their Einstat tool, noting the importance of incorporating device variability [24]. A paper introducing statistical methods to the tool received the best-paper award at the 2004 Design Automation Conference, [37]. The development of an additional tool, named EinsStat, is claimed to be the first incremental statistical timer. Any alterations made to a circuit, after timing analysis has been performed, had previously required the re-analysis of the entire circuit. Incremental analysis provides a significant improvement, as only the fanin/fanout cone of the altered section of the circuit needs to be reconsidered. Incremental analysis is a useful tool for designers when the results of timing analysis are used to make improvements within a logic path, and the results of these improvements need to be updated without re-analysing the entire circuit again.

An example of the fanin and fanout of a single register (Reg 3) is illustrated in Figure 2.11 where the fanin and fanout logic cones have been highlighted. If timing analysis is performed on the example circuit and it is determined that the performance of register 3 is detrimental to the maximum operational frequency of the circuit, then register 3 may be replaced by a different type of register cell (for instance a cell model with larger transistors and greater drive strengths, or lower threshold voltages) and the performance of the circuit re-analysed. Incremental analysis means that only the highlighted fanin and fanout cones of logic would need to be reassessed, as other sections of the design have remained untouched. This becomes especially useful during the later stages of the design process when the circuit is fully placed and routed, as timing analysis at this stage includes the parasitic resistive and capacitative effects of the wires between logic gates. The re-extraction of these effects and the re-timing of the entire design would be a significant cause of delay to the design process if they were repeated for every individual gate optimisation.

2.3.1.9 Spatial Correlation

Further methods of SSTA which specifically model intra-die variability are discussed within [38],[39] and [40]. Gate delays and process parameters (such as transistor gate lengths and metal widths) are modelled as normally distributed random variables. The spacial correlation of these variables is also considered, as it is assumed that devices close to one another will have similar characteristics compared to those further away.
The published results claim that ignoring these correlations produces overly optimistic performance predictions.

The approach within [38] uses a PERT-like traversal to create a statistical timing graph of a circuit, but each node is weighted by a random variable instead of a deterministic delay. The Program Evaluation and Review Technique (PERT) is an algorithm that is used to find the longest path in a graph, which in STA involves traversing a graph of circuit nodes and gates using the max and addition operators described in Section 2.3.1.5.

A very important point is made within [38] that the longest path in a design may change, depending on the impact of the random process variations. So finding the delay of a few critical paths is not enough, and correlations between paths due to structural or spatial correlations must be included. The aim of the approach is to create PDFs of the maximum of all path delays, which is essential for finding the probability of failure of the circuit.

Parameter variations within [38] are modelled as location-dependent normally distributed random variables, where the die is split into a number of grids. It is assumed that there are perfect correlations between process parameters within devices of the same grid, high correlation between nearby grids, and low or zero correlations in distant grids. Devices within the same grid are therefore assumed to have identical parameter variations. It is further assumed that correlation only exists between the same parameters, i.e. transistor lengths in nearby grids, but not between transistor lengths and widths within the same grid. It is claimed that the inclusion of these correlations produce a significant improvement to the accuracy of SSTA, but the process still makes approximations.
Chapter 2 Background - The Significance of Process Variation

to normal distributions of process parameters, and assumptions that the maximum of multiple gate delay distributions can also be approximated to a normal distribution.

2.3.1.10 Uncertainties and Sensitivity to Variation

Discussions of combining the measurements of timing Slack with timing Sensitivity are discussed within [41] and [42]. These approaches aim to replace certain unknown parameters (during the design stage) with possible ranges of values (rather than statistical distributions) allowing nominal slack calculations to be compared with other possible values. This would allow designers to see what variables could affect the timing of critical paths, or even which parameters cause a path to become more critical. The ranges used to generate sensitivities could be replaced with statistical distributions as an when they are known, allowing higher level design decisions to be made early on within the design process and later verified as the manufacturing process details are specified.

Variations in gate and wire delays have been considered in various papers, but it is noted within papers such as [43] [44] and [45], that factors including power supply and transition time variability had not yet be taken into account. Variations in the power supply and signal transition (switching) times effect delays and power consumption (as noted in section 2.1.2.1). The accuracy of the timing analysis is claimed to have improved dramatically when these extra factors are taken into account, but these are three separate approaches which cannot easily be combined, meaning that there are many separate SSTA algorithms that are unable to address issues that are overcome by commonly used STA methods, which remains a significant block to their widespread use within the industry [46].

Work on uncertainty within SSTA, [47] and [48], is expanded upon within [49]. One of the main arguments used within this paper is that SSTA is unable to handle certain variations, as not all variations are statistical in nature. An example is given of the fact that supply voltage and temperature variations are not necessarily statistical, but are instead uncertain, and that applying any kind of statistical distribution to an uncertain parameter can lead to incorrect conclusions. The majority of the statistical methods used in previous papers rely on parameters having a distribution, or rely on the Central Limit Theorem. The Central Limit Theorem effectively states that the sum of a large number of independent random variables will approach a Gaussian distribution. It is claimed within [49] that the assumption that gate delays and process parameters are independent random variables quickly becomes problematic and can cause significant inaccuracies, instead suggesting the use of uncertain parameters. [49] claims to be able to produce a form of all process corner timing analysis, by propagating sensitivities to variation throughout the circuit, but this is based upon the assumption that cell delays and slews have a linear dependence upon all process parameters. The accuracy of the assumption of a linear relationship between process parameters and delays had already
been dismissed within the literature, where speed binning results (sampled manufactured circuits are found to function over a range of maximum frequencies) indicate significant nonlinearities [50].

2.3.1.11 Non-normal Distributions of Process Variations

A number of efforts have been made to produce SSTA tools that consider non-normal distributions when dealing with delay variations, [51] [52]. It is claimed within [53], however, that the numerical methods used within these papers are too slow to become practical alternatives to corner analysis. An analytical method is discussed which is claimed to be much more efficient. The problem with this approach seems to be that the use of non-normal, and non-linear parameters still increases the complexity exponentially, and so the method is forced to rely upon the re-approximation of the majority of parameters back into normal distributions again.

A key issue of the majority of the the papers reviewed from the literature is that assumptions and approximations are made on the nature of gate delays and distributions of process parameters within a circuit, and where these assumptions are not made then the mathematical methods used are complex, time consuming, and non-transparent to designers [54]. The main result of these issues is that many designers within industry are still more willing to perform an over-pessimistic worst-case corner analysis of their design, and then spend time and energy making the design fit tough constraints, than risk underestimating the potential problems within a design by using SSTA and then finding problems within the design after manufacture, even after nearly two decades of research into statistical timing analysis. This is described within [55] where it is noted that significant variation-related developments are required to continue to enable the scaling of manufacturing processes, and that a significant barrier to the adoption of SSTA is the lack of understanding of the impact on the design methodology.

2.3.1.12 Monte Carlo Simulations

Each of the SSTA approaches mentioned so far have used Monte Carlo simulations as a form of golden reference against which the accuracy of their delay calculations is based. Monte Carlo consists of running a simulation several times, each time using different randomly generated values for a set of process parameters. The use of this method in place of SSTA is discussed within [56], where it is noted that STA is one of the most efficient computer aided design (CAD) algorithms, with a computational complexity equal to the number of delay nodes. The inclusion of probability distributions, correlations and random variables in SSTA algorithms leads to a dramatic increase of complexity. Alterations to the Monte Carlo method such as parallelisation are suggested, and a later
publication [57] discusses statistical methods of reducing the required number of simulations. The results from using this altered, Stratification and Hybrid Quasi Monte Carlo, approach are claimed to be faster than ‘traditional SSTA’ methods, but it is not made clear which method of SSTA is used within the comparison, or what impact the process has on current STA based design methodologies.

It is stated within [58] that Monte Carlo based statistical circuit analysis can be a practical alternative to SSTA if it is chosen and implemented intelligently, and a key focus on Monte Carlo based simulations is the reduction of the number of samples (and therefore computation time) required to produce accurate results. While [58] and [59] both describe intelligent methods of reducing this required sample size, such as independent component analysis where random vectors of correlated non-Gaussian components can be transformed to vectors of ‘nearly’ independent random variables, the lack of understanding of the impact on the design methodology and how designers can use the statistical information still remains a significant barrier to any statistical analysis becoming adopted by the industry [55].

2.4 Summary

Timing analysis is one of the last functions performed before a design is fabricated. Inaccuracies between the performance of a fabricated design and the timing simulations can lead to the expensive process of redesign, and re-fabrication. Timing analysis is therefore a key stage of the design flow in which statistical process variations must be successfully modelled. A successful solution to the problem of modelling variability within timing analysis must address the key concerns raised within the literature, accuracy, efficiency and transparency to the designer. The following chapters of this thesis focus on the impact of statistical process variations on timing analysis, and the work taken to account for these variations using existing commercial tools.
Chapter 3

Monte Carlo Cell Characterisation

It can be seen from reading the literature, that a suitable complete solution to the growing problem of device variability has not yet been found. The nanoCMOS project aims to create strong links between device, circuit, and system designers in an effort to alter design methodologies and accommodate variability. The following chapter provides details of an investigation into how device variations can be modelled at the standard cell level.

3.1 Atomistic Transistor Models

The potential impact of statistical variations within the structure of a transistor were mentioned within Section 2.1.3.1 of this document. Members of the nano-CMOS project have carried out work to produce a 3D atomistic simulator for characterising the statistical process variations within nano-scale transistors, [16][60][61], and on simulating the effects of line edge roughness (LER) [62][63]. This simulator was designed with the intention of discovering the impact of statistical process variations, and establishing where discrepancies between sign-off analysis tools and fabricated devices may originate.

An example of the results produced from the 3D simulations is given in Figure 3.1, where atomistic models have been generated and simulated to provide a distribution of current-voltage curves for a 35nm transistor that would traditionally have been represented by a single continuous charge model. Figure 3.1(a) shows a 3D rendered image of a single instance of a 35nm MOSFET, where factors such as the number and location of dopant atoms have been randomised. Figure 3.1(b) shows the distribution of Drain Current ($I_D$) versus Gate Voltage ($V_G$) curves generated from 200 randomised 3D MOSFET models, together with the I-V curves generated by a smooth, continuously doped transistor and
the average of the 200 randomised models. A logarithmic scale has been used to represent $I_D$, as the values of $I_D$ for $V_G = 0V$ are spread over two orders of magnitude, indicating a significant range of leakage current. The variation in $I_D$ decreases as $V_G$ increases, but these curves represent a large spread of voltages at which current will flow through the device. This 'switch on' point is referred to as the Threshold Voltage ($V_T$) and is a critical factor in the performance of transistors, as a range in $V_T$ implies a range in the time it would take for the transistors to switch on or off during an input transition.

![Simulation of random discrete dopants in a 35 nm MOSFET](image)

(a) Randomised discrete dopants in a single 35nm MOSFET

![Logarithmic plot of Drain Current versus Gate Voltage for 200 randomised 35nm MOSFETS](image)

(b) Logarithmic plot of Drain Current versus Gate Voltage for 200 randomised 35nm MOSFETS

**Figure 3.1:** Simulation of random discrete dopants in a 35 nm MOSFET device. Taken from [64]

The above research performed by the nano-CMOS project has concluded that RDD, Poly Silicon Granularity and LER are the three main sources of fluctuations in $V_T$. The
distributions of the I-V curves were confirmed to closely resemble the data observed by the manufacturer of the 35nm process, showing that the 3D atomistic simulator could accurately predict the impact of statistical process variations on the performance of a single transistor. The importance of this research is that it has highlighted that statistical process variations do have a significant impact on the performance of transistors, and 3D models of sub 35nm devices have been used to predict that variations in transistor performance will increase as the dimensions of devices continue to decrease.

The next natural step for this research, and this thesis, was to investigate the impact of statistical process variations on multiple transistors, to determine whether these performance variations would become compounded or simply averaged out. It was recognised that full atomistic simulations of networks of transistors would very quickly become impractical to perform, and so a method of abstraction was required.

### 3.2 SPICE Compact Models

A compact model is a set of mathematical equations that describe the behaviour of semiconductor devices, and can be used within circuit simulations. The Berkeley Short-channel IGFET Model (BSIM)[65] is an industry standard set of compact models that allow the behaviour of transistors to be accurately simulated at the SPICE level, and represents a level of abstraction above physical simulations.

Members of the nano-CMOS project performed an extraction method on the I-V curves generated by the 3D atomistic simulator, and found that a set of key BSIM parameters could be used to replicate the behaviour of the atomistic models. The values of these parameters were stored as a library of statistical compact models. This library then represented the distributions in performance of NMOS and PMOS transistors generated by the 35nm manufacturing process. These SPICE level transistor models could be used to assess the performance of multiple devices within small circuits, much more effectively than when using the atomistic simulator. Utilising the statistical compact model library then required the development of a tool to aid in the automation of the process.

### 3.3 RandomSpice

RandomSpice is a statistical circuit simulation tool that has been developed during the lifetime of the nano-CMOS project [66]. The tool enables large-scale Monte Carlo SPICE simulations of statistical and process variability, with the ability to perform data harvesting and statistical analysis of the simulation results. RandomSpice was designed to support the statistical compact models generated from the 3D atomistic
simulator, and to take advantage of parallel processor systems such as cluster computing technologies.

The flow diagram in Figure 3.2 illustrates how RandomSpice can be used. A template SPICE netlist, representing the circuit, input stimuli and output measurements for which the user wishes to model variability, is inputted to the RandomSpice engine. Transistor instances within the template are marked for variability simulations by the use of a keyword in the model instance line. RandomSpice then generates multiple randomised copies of the template circuit, where each marked transistor instance within the template is replaced by a model, with appropriate device dimensions, that has been randomly selected from the Compact Model Library. These multiple netlists are passed to a specified SPICE simulation tool, where simulations can be performed in series or spread over a parallel computing platform, the results from which are stored in a database. Statistical analysis can then be performed on the database of SPICE measurements, allowing the user to view information on the distributions of power consumption, circuit performance and the percentage of circuits that have met specific design criteria.

**Figure 3.2:** Work flow of RandomSpice statistical simulation engine, from [66] with permission.
Chapter 3 Monte Carlo Cell Characterisation

The features illustrated within Figure 3.2 represent the commercial version of the RandomSpice tool that has been developed since the conclusion of the nano-CMOS project. The work presented within this thesis took place during the development of RandomSpice, and represents a testing and evaluation stage in the creation of the tool. Features such as the statistical enhancement engine, database generation and statistical analysis were not yet fully implemented.

RandomSpice provides a method of abstracting the statistical information within 3D atomistic transistor models to the circuit level, and allows for the analysis of the impact of statistical variations on networks of transistors. The simplest CMOS logic gate, an inverter, provided an excellent example of a commonly used transistor network to test with RandomSpice.

3.4 Cell Characterisation

Multiple simulations can be used to verify the behaviour of a circuit over a range of operating conditions, but this becomes prohibitive for circuits of any significance. The possible combinations of input patterns, input transition times, delays between inputs, output loads, operating temperatures, and process corners make it impractical for a designer to check all of them. An alternative approach is to verify the behaviour and performance of sets of small circuits, typically logic gates, from which larger circuits can be made. The sets of small commonly used sub-circuits are referred to as cells, and the process of establishing their performance is referred to as cell characterisation.

3.4.1 Measurements

The performance of cells is generally divided into two forms; cell delays and cell power consumption. Brief descriptions of the most commonly used forms of measurement and the terms used to describe them are given here.

3.4.1.1 Delays

Figure 3.3 provides an illustration of what is referred to as transition times and propagation delays. A transition time is the length of time taken for an input or output to switch from one voltage to another. The voltages from which these measurements are taken should be specified with the measurement to avoid confusion, and are usually referred to as percentages of the supply voltage. For example, some rising transition times may be recorded from 10% to 90% of VDD, while others may be from 30% to 70% of VDD. A propagation delay is the time taken for an output to respond to a switching input, again the points from which these measurements are recorded should be specified with
the measurement. The most commonly used values are from 50% of the input voltage to 50% of the output voltage, but this should not always be assumed to be true.

Figure 3.3: Illustration of transition and propagation delays

Sequential cells such as registers will also have measurements to record the minimum setup and hold times required for the cell to function. The setup time is the minimum time that the input to a sequential cell must be stable before the active edge of the clock signal, and the hold time is the minimum time that the input must remain stable after the active edge of the clock signal. This is illustrated in Figure 3.4.

Figure 3.4: Illustration of setup and hold times
3.4.1.2 Power

The leakage current of the cell can be measured at a point within the simulation where the cell is at a steady state, where no inputs or outputs are in the process of switching. This leakage current may be different depending on which stable state the cell is in at the point of measurement. The switching energy of a cell can be measured by integrating the supply current multiplied by the supply voltage over the period of time from when the input began a transition to the end of the output response. The points at which signals are regarded to have started and finished a transition should also be recorded. Another form of dynamic energy is referred to as the internal switching energy, where an input transition does not result in the switching of an output.

![Figure 3.5: Illustration of Power Measurements. Switching energy is calculated by integrating the supply current multiplied by the supply voltage over the cell transition time.](image)

3.4.2 Statistical Analysis of an Inverter at 130nm

The 3D atomistic simulations described earlier within Chapter 3.1 are based upon foundry information from a process technology with transistors with a minimum length of 35nm, but the corresponding commercially sensitive information, such as a standard cell library or SPICE models for the standard cells, could not be made available to the nanoCMOS project. This meant that accurate variation models could be used at the transistor level, but commercial standard cells were not yet available for circuit analysis with a real manufacturing process. An alternative 130nm commercial technology node was found, where the standard cell library and SPICE models of standard cells were available for use by the nanoCMOS project. An early trial of the use of RandomSpice in cell characterisation was performed using a combination of the levels of transistor
performance variation found at 35nm and the commercial library that was available at 130nm. This allowed the performance of 130nm standard cells to be assessed as if they were subjected to the levels of statistical process variations found at 35nm.

An experiment was performed to test the impact of statistical process variations on a CMOS inverter gate. The SPICE netlist is provided in Figure 3.6, in which two transistors from a 130nm process are configured to create a CMOS inverter with input $A$ and output $Y$. The ratio of the width of the PMOS transistor to the NMOS transistor was 1.55, and the supply voltage to the gate was set at 1.2V. A piecewise linear (PWL) command was used to create a single rising transition on input $A$ and a measurement command was used to record the propagation of the input waveform through to output $Y$. The propagation delay was measured from the time that the input waveform reached 50% of the supply voltage to the point that the output waveform reached 50% of the supply voltage. The keywords $ATOMN$ and $ATOMP$ were used within the transistor instances to indicate to RandomSpice that these transistors were to be replaced by variation models within the Compact Model Library.

![Figure 3.6: SPICE Variation Test Circuit for a CMOS Inverter.](image)

The inverter netlist was passed into RandomSpice and 2000 randomised instances of the netlist were produced. Each randomised netlist was simulated using Hspice, and an individual propagation delay measurement was produced from each simulation. These measurements were collected and used to produce the histogram shown in Figure 3.7(a), in which the mean delay was recorded as 18.18ps and the range of delays (from minimum to maximum) was 3.85ps. The range of possible delays is over 20% of the mean delay, a wide spectrum in the performance of a single simple gate, and the distribution appears to be skewed, with a larger right hand side tail than left.

A normal probability plot can be used to compare a normal distribution against recorded data. If the normal probability plot is linear then this suggests that the recorded data
follows a normal distribution, whereas if the plot is non linear then it is likely that the recorded data follows an alternate distribution. Figure 3.7(b) contains a normal probability plot for the propagation delay distribution of the 2000 inverter instances (blue) with a linear reference (red), and the extremes of the distribution are distinctly non linear. The use of RandomSpice within these preliminary experiments indicated that statistical process variations in networks of multiple transistors may not be represented by simple Gaussian distributions.

This set of simulations provided a direction to the investigation of modelling statistical
process variations in circuits, as it is often assumed that the combination of the multiple sources of variation and multiple transistors would combine to produce normal distributions of circuit performance. The behaviour of the inverter under variation seemed to suggest that the propagation delay through the gate would not follow this assumption, but the simulation circuit used was very simplistic and did not represent a realistic environment under which the gate would operate. Factors such as the time taken for the input to the inverter to complete a transition; the amount of capacitance that the output would have to drive and the internal resistive and capacitive parasitics of the gate had been ignored. It was therefore important to investigate the impact of statistical process variations under more realistic conditions, and on larger networks of transistors.

3.4.3 Statistical Analysis of a D-Type Flip-Flop at 35nm

A second trial was performed using a D-type flip-flop [67], with transistor sizes that were scaled from the 130nm commercial technology to the same 35nm technology node as was used to generate the atomistic transistor model cards for RandomSpice. The purpose of this trial was to establish the impact of statistical process variations at the correct process node, although scaled transistor sizes were used for the standard cell, which may not reflect how a commercial 35nm D-type would actually be designed. The aim of this experiment was to record the effects of random process variability on the performance of a commonly used cell, and how this variation is affected by changes in the supply voltage to the cell.

3.4.3.1 Testbench

When characterising the performance of a cell it is important for the transition times of input signals to match the expected internal signal transition times, as input skew has an effect on delays and power consumption. It is also necessary to apply an appropriate load to the output signals to ensure that the measured characteristics are realistic. For these reasons inverters scaled to the same 35nm technology were used to buffer each input and output signal.

Voltage sources, set at 0V, were placed between the supply and device-under-test, as this provides a simple method of recording current flow within SPICE. This testbench is represented in figure 3.8.

3.4.3.2 Measured Characteristics

Full characterisation of a cell requires recording power and timing measurements for every possible output transition, over a range of: supply voltages; operating temperatures;
Figure 3.8: Dtype Flip-Flop Characterisation Testbench, showing I/O Buffers and 0V voltage sources

Figure 3.9: Dtype Flip-Flop Characterisation Timing Measurements
Chapter 3 Monte Carlo Cell Characterisation

Figure 3.10: Probability Density Function of Clock-nQ delays for 2000 Dtype models with a supply voltage of 1V. A normal distribution is provided for comparison.

input skews and output loads. The aim of this experiment was to simply investigate how random process variations could be represented at higher levels of the design flow, and it was decided that only a small section of the D-type characteristics were required when performing a proof of concept. For this reason the single transition of nQ from ‘1’ to ‘0’ was recorded over a range of supply voltages, with pre-set and clear signals set as inactive. Figure 3.9 shows the timing measurements that were obtained: the setup time between Data and the Clock edge; the delay between the clock edge and the nQ transition and the time taken for nQ to fall.

3.4.3.3 Results

The probability density function (PDF) of the output delay is shown in Figure 3.10, in which it can be seen that the delays are not normally distributed. These results represent a spread of delays for a fixed set of process corners, which would traditionally be represented by a single value within a delay model. The range of the delay values obtained highlights the importance of modelling these factors within cells. The output delay PDF was obtained by simulating 2000 Dtype models, each containing different combinations of the transistor model cards, at a supply voltage of 1V. The shortest delay was obtained by the simulation of a Nominal Dtype model (with no stochastic variations), the average delay of the distribution was approximately 10ps longer.
Chapter 3 Monte Carlo Cell Characterisation

(a) Changes in Mean and Standard Deviation of Output Delay over a Range of Supply Voltages

(b) Changes in Distributions of Output Delay over a Range of Supply Voltages

Figure 3.11: The Effects of increasing Supply Voltage on the Output Delay of a 35nm Dtype Flip-Flop

A key factor in the performance of transistors is the level of the supply voltage, with higher supply voltages increasing the operating speed. It was unclear what impact an increase in power supply would have on the variation in transistor delays, and so the Dtype simulations were repeated over a range of supply voltage levels, from 1V to 2V. It was found that the mean and standard deviations of the output delays of the Dtype decreased as the supply voltage increased, and the same was also found to be true for the output transition times. This indicated that an increase in supply voltage not only
changes delay times, but also reduces the possible spread of variation, and therefore increases timing reliability. The downward trends of the mean and standard deviation of output delay and for an increase in supply voltage is shown in Figure 3.11(a), and the impact on the shape of the distribution of delays is shown in Figure 3.11(b). The same trends are illustrated in Figure 3.12 for output transition times.
Increasing the supply voltage to improve operating frequency has an obvious cost, as increasing the supply voltage increases the power consumption of the cell. The results of these simulations have found that both the mean and the standard deviation in static power consumption increase with raised supply voltages, Figure 3.13(a), implying that not only is more power consumed, but that the power consumption becomes less predictable. The spread of the distribution of leakage energy for a supply voltage of 2V was found to be 7 times wider than than the leakage energy for a supply voltage of 1V, Figure 3.13(b). These results suggest that there is not simply a trade off between the length of delays and amount of power consumption, but also in the predictability of delays and power consumption.

The mean switching power of the cell was also found to increase with increases in supply voltage, as expected, but the standard deviation was found to decrease from the range of 1V to 1.4V and increase from 1.4V to 2V, Figure 3.14. In this case the switching power is a measure of the current flow through the cell while the output is in transition from high to low, and so depends on the output transition time. The initial reduction in standard deviation of the switching power seems to suggest that the decrease in variation of the switching time outweighs the increase in variation of the switching current, until the supply voltage reaches 1.4V. If the supply voltage can be altered to improve the reliability of power consumption and circuit performance (depending on the goals of the designer), then this behaviour in the distribution of switching energy may be useful as a guide to design automation tools. Although it would first have to be ascertained if this behaviour was still found to be true for a range of standard cells, under more accurate testing conditions. The changes in power consumption and delay distributions may be found to be insignificant over a realistic range of supply voltages for a real circuit, as doubling the supply voltage is not feasible without causing significant damage in most situations.

The process of generating testbenches and measurements for every possible input-output transition for a range of standard cells was beyond the scope of this project, especially when tools already exist for automating the characterisation of cells, and storing the results in a standardised format. The next logical step for this research was to establish whether and how statistical process variations could be incorporated within existing, industry standard design flows.

### 3.4.4 Standardisation using Liberty NCX

The industry standard method of assessing the performance of a circuit is to perform a form of STA or SSTA. These processes require the use of a library of characterised commonly used cells. It was decided that a widely used, industrial standard cell characterisation tool should be used when investigating the impact of statistical process
(a) Changes in Mean and Standard Deviation of Leakage Energy over a Range of Supply Voltages

(b) Changes in Distributions of Leakage Energy over a Range of Supply Voltages

**Figure 3.13:** The Effects of increasing Supply Voltage on the Leakage Energy of a 35nm Dtype Flip-Flop

variations on cells, as this would help to ensure that simulations occurred within realistic operating environments, and that the results were presented in an industry standard format.

Liberty NCX is a tool that automates the generation of cell libraries [68]. The user provides SPICE netlists for each cell to be characterised, SPICE models for any transistors used and template files that specify the parameters to be used at both the library and
Chapter 3 Monte Carlo Cell Characterisation

(a) Changes in Mean and Standard Deviation of Switching Energy over a Range of Supply Voltages

(b) Changes in Distributions of Switching Energy over a Range of Supply Voltages

Figure 3.14: The Effects of increasing Supply Voltage on the Switching Energy of a 35nm Dtype Flip-Flop

cell levels. Liberty NCX uses the collection of input data to generate a set of SPICE netlists that simulate the behaviour of the cells over a range of operating conditions. The results of these simulations are collected and translated into a cell library. It is possible to use an existing cell library as an input to Liberty NCX, from which library and cell templates can be generated. Figure 3.15 provides an illustration of this work flow from [68], from which more details are provided below.
3.4.4.1 Library Templates

This file specifies library characterisation parameters such as units, threshold limits, and which cells to include. The ranges over which the characterisation process will be performed can also be specified, allowing the user to specify what input transitions and output load capacitances will be applied to each cell. The threshold limits are used to specify the points at which transition times and propagation delays are to be measured. Figure 3.16 provides an example of some of these settings where the propagation delays are to be measured from 50% to 50% (of input to output voltages) and the transition times are to be measured from 30% to 70% (of the signal voltages). The example also includes 4 input transition times and 4 output capacitances to be used during the characterisation simulations, this will result in simulations of the cell being performed for each of the 16 possible combinations of input transition and output load.

A variation cell library that models statistical process variations could be generated without making major changes to a library template. In fact a template can be generated from an existing cell library for which the user wishes to model variability, this could then be used to model variation by simply adding the names of the variation cell netlists (e.g. INV_1 INV_2 INV_3 etc.) to the do list.
*Library_Template.opt
*Unit definitions
time_unit : 1ns ;
voltage_unit : 1V ;
leakage_power_unit : 1pW ;
capacitive_load_unit : 1.0000000 pf ;

*Threshold Limits
slew_lower_threshold_pct_fall : 30.0000000 ;
slew_upper_threshold_pct_fall : 70.0000000 ;
slew_lower_threshold_pct_rise : 30.0000000 ;
slew_upper_threshold_pct_rise : 70.0000000 ;
input_threshold_pct_fall : 50.0000000 ;
input_threshold_pct_rise : 50.0000000 ;
output_threshold_pct_fall : 50.0000000 ;
output_threshold_pct_rise : 50.0000000 ;

*Characterisation Ranges
ncx_input_transition_time_index : 0.02 0.04 0.08 0.16 ;
ncx_total_output_net_capacitance_index : 0.0006 0.003 0.007 0.015 ;

*List of Cells to Characterise
do {
   INV
}

Figure 3.16: Example of a library template file.

3.4.4.2 Cell Templates

The cell template specifies parameters such as the cell area and which input-output transitions to characterise. The example template in Figure 3.17 includes descriptions of the two pins of an inverter: Pin A is simply described as an input, while Pin Y is configured as an output pin with a maximum load capacitance, and output transitions for given input transitions on Pin A. The maximum load capacitance values are used by a synthesis tool to determine if a cell is of a suitable strength for driving the fanout network. There must be a template file for each cell to be characterised.

The generation of a variation cell library would require that each randomised instance of a cell is characterised under the same conditions. Every randomised cell would therefore require the same cell template, as the cell structure, logic function and sensitivity list are identical. A variation cell library would simply require the original cell template to be copied into, or symbolically linked to, files with names that match the variation instance names (e.g. INV_1.opt INV_2.opt INV_3.opt etc.)
3.4.4.3 Cell SPICE Netlists

These netlists contain transistor level descriptions of a cell. The description should be contained within a sub-circuit and the file name and cell name should match. The netlists will be referenced by \textit{.include} lines within the automatically generated simulation files, and so must be valid SPICE files. An example of an inverter sub-circuit is given in Figure 3.18.

The generation of variation instances of SPICE netlists using RandomSpice has already been described, and the same process can be applied to simple sub-circuit definitions. The only work required is that RandomSpice will use the same \texttt{.SUBCKT} definition for each randomised netlist, but will automatically add a suffix to the netlist file (e.g. INV\_1.spc INV\_2.spc INV\_3.spc etc.). The \texttt{.SUBCKT} lines must be altered so that the definition matches the file name (e.g. \texttt{.SUBCKT INV\_1, .SUBCKT INV\_2 etc.}).

\begin{verbatim}
*INV.spc
*INVERTER SUBCIRCUIT DEFINITION
.SUBCKT INV Y A VDD VSS
   MM1 Y A VSS VSS NMOS_MODEL L=0.12U W=0.58U
   MM2 Y A VDD VDD PMOS_MODEL L=0.12U W=0.9U
.ENDS
\end{verbatim}

\textbf{Figure 3.18:} Inverter sub-circuit definition

3.4.4.4 Transistor Models

Transistor (and other component) models that are common to multiple cells are included within these files. The parameters specified within the model files will reflect the
manufacturing process that has been selected for design fabrication. Different transistor model files may be used to represent different process corners and operating conditions. Very basic examples of the layout of transistor models are given in Figure 3.19.

The transistor models generated by RandomSpice can be included within each of the randomised cell netlists, allowing the use of a completely blank transistor model file to generate a variation cell library.

```
.MOST NMOS.MODEL NMOS
+ LEVEL = 49
+ TNOM = 2.5000E+01
.MOST PMOS.MODEL PMOS
+ LEVEL = 49
+ TNOM = 2.5000E+01
```

**Figure 3.19:** Transistor model definitions.

### 3.4.4.5 HSpice Simulation Files

The information within the library and cell templates is used by Liberty NCX to generate test-vectors for the cells under characterisation. The range and function of these simulation files will depend on the settings specified by the user, such as whether to perform both or one of power analysis and timing analysis. The number of simulation files per cell will depend on the function of the cell and the number of input to output combinations that have been specified for characterisation.

The generation of these files is automated and so no steps would be required to include statistical process variations into the procedure, but the use of Liberty NCX could be avoided if the SPICE simulation files for a single instance of a cell is obtained. The simulation files for the original cell could be used to stimulate the variation instances simply by changing which cell sub-circuit netlist is included within the simulations, and by renaming the instance of the cell under test to match the appropriate variation sub-circuit name.

### 3.4.4.6 Cell Libraries

Liberty is an Open Source standard for modelling cell libraries, and is widely supported throughout the semiconductor industry [69].

A cell library contains the results of the cell SPICE simulations, combined with the template options that have been used to generate the library. This can include the shapes of the waveforms used as input stimuli, user specified SPICE simulation commands, and the ranges of input slew and output loads used. The results of the characterisation simulations are contained within a separate section for each cell, an extract of which
is provided in Figure 3.20. Measurements recorded for combinational cells can include: rise and fall propagation delays for each output; rise and fall transition times for each output; rise and fall power consumption for each output; rise and fall power consumption for each input; capacitance of each input and the leakage power for each steady logic state. Sequential cells will also include minimum setup and hold times for relevant pins.

```plaintext
cell ("INV") {
  cell_leakage_power : 2.459391e+04;

  pin (A) {
    direction : "input";
    capacitance : 0.000485;
  }

  pin (Y) {
    direction : "output";
    function : "(!A)";

    internal_power () {
      related_pin : "A";
      rise_power {
        index_1("0.002, 0.004, 0.02, 0.06");
        index_2("0.00025, 0.0005, 0.0025, 0.005");
        values("0.0004768, 0.0004895, 0.0005045, 0.0005086", \
          "0.0004711, 0.0004795, 0.0005053, 0.0004974", \
          "0.0006158, 0.0006017, 0.0005397, 0.0005325", \
          "0.0011337, 0.0010868, 0.0009108, 0.0008077");
      }
    }

    timing () {
      related_pin : "A";
      cell_rise {
        index_1("0.002, 0.004, 0.02, 0.06");
        index_2("0.00025, 0.0005, 0.0025, 0.005");
        values("0.0033413, 0.0043633, 0.0120529, 0.0216118", \
          "0.0038020, 0.0047901, 0.0124993, 0.0219122", \
          "0.0061225, 0.0076602, 0.0165556, 0.0258178", \
          "0.0095542, 0.0117188, 0.0242205, 0.0361106");
      }
    }
  }
}
```

Figure 3.20: Extract from an Inverter Cell Library Entry.

The example contains two look up tables, the first for power consumption of the output performing a rise transition (\texttt{rise\_power}), and the second for the propagation delay of the output rising (\texttt{cell\_rise}). These tables contain two indexes with four values each,
and a table of 16 values for the measurements recorded at each index point. Index_1 in this example is input transition time, which refers to the rows in the table of values, and index_2 is load capacitance which refers to the columns. The power look up table within this example does not include the power required to charge the output load, hence why the table is contained within an `internal_power` section. Full details on the structure of Liberty format cell libraries can be found within the Liberty format documentation [70].

The characterisation of variation cell instances would simply result in an additional entry for each instance within the library. The cell INV would be replaced by multiple entries (INV_1, INV_2, INV_3 etc.), where each entry would have the same structure and indexes, but different measurements within the values of the look up tables.

### 3.5 Variation Cell Characterisation

![Variation Characterisation Flow](image)

**Figure 3.21: Variation Characterisation Flow**

An examination of an industrial standard cell characterisation tool and an industry standard cell library format has revealed that statistical process variations could be incorporated into the design flow with relative ease. The templates from an ordinary cell library can be duplicated with minor alterations, and the original cell sub-circuit
can be passed through RandomSpice. The cell characterisation process would then be performed on each individual variation cell instance, resulting in a separate entry into the cell library for each model of statistical process variation. The Liberty NCX work flow in Figure 3.21 has been updated with the modifications necessary to model statistical process variations, the entire process can be controlled using simple scripting languages.

The value used to seed the random number generator within RandomSpice is included as a comment within each netlist generated by the tool. This seed can be used to force RandomSpice to generate exactly the same netlist at a later date. These seed values are copied and added as comments within generated cell libraries during the Monte Carlo Static Timing Analysis (MCSTA) process, allowing the multiple cell netlists to be discarded after characterisation. This can be an important step for the characterisation of multiple cells in a parallel processing environment where large amounts of disk space may be consumed rapidly.

### 3.5.1 Standardised Statistical Analysis of an Inverter at 35nm

The method of incorporating statistical process variations into the Liberty NCX flow was tested using a 35nm inverter netlist, as the inverter is the simplest CMOS logic gate to simulate, and 35nm transistor variation model cards were the first made available for use with RandomSpice. The SPICE netlist for the 35nm inverter was passed through RandomSpice, generating 10,000 randomised instances. These instances were then characterised using the variation characterisation flow illustrated in Figure 3.21. This produced a variation cell library with 10,000 entries for different models of the inverter, providing distributions of delays and power consumption for each point in the measurement look up tables.

The distribution of delays for a single point within the look up table of the falling propagation delays is shown in Figure 3.22. This is comparing the delay of the 10,000 inverters at a fixed input transition and output load, for which the mean delay was found to be 4.02 ps and the range of delays was 2.08 ps. The range of possible delays is almost 52% of the mean delay, a much wider spectrum in the performance of the gate than at 130nm. This fits with the suggestion that statistical process variations have a much greater impact on smaller fabrication technologies than larger fabrication technologies. The characterisation process is performed using an industry standard tool, under realistic operating conditions, and so it is feasible to perform detailed investigations of the distributions of delays and power consumption under statistical process variations. The distribution of delays within Figure 3.22 appears to be slightly skewed, as was found during the initial testing of the inverter at 130nm in Section 3.4.2. For the purposes of comparison normal and Inverse Gaussian distributions are included within the plot, both of which appear to closely resemble the recorded data. If the distributions of cell delays
could be accurately represented by normal distributions, then many of the assumptions made within the literature on SSTA would be found to hold true.

Figure 3.22: Distribution of Propagation Delays through a 35nm CMOS Inverter, for a Falling Output Transition

One method of verifying whether a distribution is normally distributed is to generate a normal probability plot, where the probability axis is scaled in such a way as to produce a straight-lined CDF for normally distributed data. A normal probability plot is provided in Figure 3.23, where the recorded data is distinctly non-linear when compared to the Normal Fit line. The Inverse Gaussian fit follows the shape of the Recorded Data much closer, but is still not fully representative of the tails. The fact that the centre of the delay distribution matches a normal distribution means that the mean delay may be modelled relatively accurately, but it is the extreme values of the distributions that are critical for setup and hold timing analysis.

If the shape of this recorded distribution is common to all of the measurements for the inverter then this suggests that statistical circuit analyses that rely on Gaussian distributions will have some accuracy in the estimation of mean delays, but will suffer in predictions of maximum and minimum delay values. The propagation delay plots in Figure 3.22 are for a single input transition time and single output load and so only represents a single point of operation within the characterised space. An extra dimension is added within Figure 3.24 which illustrates the change in the propagation delay distribution as the input transition time is changed. The range of propagation delays, or the variation in the performance of the cell, increases as the input transition time increases. While the data within Figure 3.24 may appear to show that some of
Figure 3.23: Normal and Inverse Gaussian Probability plots of Propagation Delays through a 35nm CMOS Inverter, for a Falling Output Transition

Figure 3.24: Propagation Delays versus Input Transition Time
the characterised cells operate faster with long input transition times than with short input transition times, this is not the case and is simply an artefact of the measurement methods used during cell characterisation. The propagation delay of the cell is often defined as the time between the input and output reaching 50% of the supply voltage. A reduction in the propagation delay of a cell for an increase in input transition time simply shows that the halfway point of the output transition becomes closer to the halfway point of the input transition. Adding half of the transition time to the propagation delay would reveal the true degradation in performance during STA. The *Uniform* plot (in red) refers to the propagation delay of an inverter cell where the transistor models contains no variability, which represents the traditional model that would be used to predict the performance of the cell. The distribution of delays is further complicated when the range of possible output loads is included, as shown in Figure 3.25, where the maximum, mean and minimum planes of the distribution are shown. These plots confirm that there is a non-linear change in the distributions of propagation delay with changes in load capacitance and input transition time.

Examining three dimensional distributions is complicated for even this simple cell, and so it is much easier to perform analysis at each individual point within the lookup table. Figure 3.28 presents a series of normal probability plots for the falling propagation delays through the inverter, where the normal plots are presented in a grid format.
Figure 3.26: Output Transition Times versus Input Transition Time versus Load Capacitance

Figure 3.27: Switching Energy versus Input Transition Time versus Load Capacitance

that matches the falling propagation delay lookup table. The values used for input transition time increase from top to bottom, and the output load capacitance increases from left to right. The measured distributions of delays are in black, while the fitted normal distributions are in red. The scales of propagation delay and probability have been removed, as they become hard to read in this format and are not important when comparing the shapes of the two distributions. It appears from this data that increasing
the load capacitance changes the distribution of propagation delays to be less normally distributed, while increasing the input transition time causes the delay distributions to become more normally distributed. This is reflected within Figure 3.29, where fitted inverse Gaussian distributions (blue) are shown to match the measured data (black) for low input transition times and high load capacitances, but not low load capacitances and long input transition times. This pattern of changing distributions was also found to be true for rising propagation delays through the inverter.
Figure 3.28: Inverter fall delay normal plots
Figure 3.29: Inverter fall delay inverse normal plots
3.5.2 Goodness of Fit Testing

While graphically assessing the fit of distributions is a quick and simple method, it is not precise, especially as the axis of a plot and the number of sample bins in a histogram can be changed to make the data appear to be either more or less linear. A superior method is to perform Goodness of Fit tests, of which there are many different types for different situations. A brief summary of some commonly used tests are given here.

3.5.2.1 Kolmogorov-Smirnov

While the Kolmogorov-Smirnov is a popular and commonly used goodness of fit test, it focuses on the centre of distributions rather than tails. The proposed test distribution must also be specified, parameters such as the mean and standard deviation cannot be derived from the data. This therefore means that the Kolomogorov-Smirnov test is unsuitable in this situation, as the expected distribution of cell performance is unknown, and the tails of the distribution are critical for accurate circuit analysis.

3.5.2.2 Two-Sample Kolmogorov-Smirnov

This version of the Kolmogorov-Smirnov test allows for a comparison between two distributions, identifying whether they both originate from the same distribution. The test is not able to identify which distribution the test distributions originate from, simply whether the two test distributions come from the same source. While this is more suitable than the original Kolmogorov-Smirnov test it still focuses on the centres of the distributions to be compared.

3.5.2.3 Lilliefors

The Lilliefors goodness-of-fit test is used to check if a distribution comes from the normal family, for which the parameters must be estimated. The Lilliefors uses the null hypothesis that the sample being tested comes from a distribution in the normal family, against the alternative hypothesis that the sample does not come from a normal distribution. This test is suitable when a the null distribution is unknown and the parameters need to be estimated.

The Lilliefors test was used for the distributions of propagation delays and output transition times for each input transition and output load for 10,000 variation instances of each of six standard cells: NAND, NOR, AND, OR, inverter and buffer. This gives a total of 40 measurements, each having an individual look up table of 16 different combinations of input transition and output load.
Of the 640 generated timing distributions, 623 Lilliefors tests rejected the null hypothesis with a significance level of 5%, the remaining 16 were spread over a range of cells, measurements and operating conditions. The fact that 17 tests did not reject the null hypothesis does not mean that those 17 particular distributions were normal, simply that the hypothesis could not be rejected at this level of significance.

This test provides a more scientifically accepted method of checking the shapes of the measured distributions, and indicates that it is inaccurate to assume that the behaviour of cells can be completely predicted by normal distributions. It should therefore be more accurate to perform a timing analysis of a circuit by sampling from a library of cell delays, rather than using statistical models, which would require a form of Monte Carlo Static Timing Analysis.

### 3.5.2.4 Chi-Square Goodness of Fit

The Chi-square test uses the null hypothesis that the recorded data are random samples from a normal distribution, with a mean and variance that is estimated from the recorded data. The alternative hypothesis is that the data are not normally distributed with the estimated mean and variance. The test is performed by grouping the data into bins and comparing the binned data with expected counts that would be found if the distribution was normal. The chi-square test statistic is then calculated on each bin using Equation 3.1, where $O$ are the observed counts, $E$ are the expected counts and $N$ is the number of bins. This test statistic is compared to a critical value, which is obtained from the chi-square distribution with $N - P_{Est} - 1$ degrees of freedom and the chosen significance level $\alpha$, where $P_{Est}$ is the number of estimated parameters (in this case two, the mean and variance of the distribution). If the statistic is greater than the critical value then the null hypothesis is rejected.

$$\chi^2 = \sum_{i=1}^{N} \frac{(O_i - E_i)^2}{E_i}$$

(3.1)

Of the 640 different timing distributions that were generated for the set of standard cells, 633 Chi-square tests rejected the null hypothesis with a significance level of 5%, with the remaining 7 distributions spread over a range of cells, measurements and operating conditions. The fact that 7 tests did not reject the null hypothesis does not mean that those 7 particular distributions were normal, simply that the hypothesis could not be rejected at this level of significance.

The Chi-square goodness of fit test can also be used to compare sampled data with other expected distributions, as long as values can be generated for the expected bin counts.
A series of tests were performed on each of the distributions of delays and transition times. The Matlab distribution fitting tool was used to fit parameters for a series of distributions against the measured data. The fitted distributions were used to establish expected values against which the measured data were compared using the Chi-square goodness of fit test. Over 95% of each of the measured data sets were rejected at a 5% significance level for normal, log-normal, inverse Gaussian, generalised extreme value and gamma distributions.

3.6 Summary

The growing problem of device variability has not yet been solved, and it is important that the random statistical process variations within manufactured transistors can be predicted and designed for. The ability to model the effects of statistical process variations within atomistic models and the use of RandomSpice has been introduced, and demonstrated using Dtype Flip-Flops.

The standard format for cell characterisation is the Liberty format, and Variation Cell Characterisation has been introduced as a method that can be used to model statistical process variations within standard cells, using the industry standard format and industry standard characterisation tools.

Some statistical modelling of the characterised standard cells was attempted, and it was found that there was no simple way to statistically model the range of power and delays within the cells. It was instead suggested that sampling would provide the most accurate method of propagating timing and power variations to the circuit level. The following chapter discusses the steps required to perform circuit level statistical analysis.
Chapter 4

Monte Carlo Circuit Analysis

Research literature on the subject of statistical circuit analysis continually highlights the potential for pessimism when performing Corner Analysis on circuits. This pessimism can lead to the over design of circuits, where the design time is increased in the attempt to meet timing requirements, and the circuit size and power consumption can be dramatically increased unnecessarily. It is important for statistical process variations to be modelled accurately, removing this pessimism and allowing designers to achieve efficient design solutions as quickly as possible.

Statistical process variations can have a significant impact of the performance of transistors in modern manufacturing processes, and these distributions of performance have been captured at the cell level using Monte Carlo Cell Characterisation. The distributions of timing and power consumption within several standard cells have been examined and compared with well known distributions, but the significance of the shape of the cell distributions will not be known until the performance of whole circuits can be tested.

This section provides greater details on the traditional methods of analysing the timing and power consumption of designs, and how statistical process variations can be modelled within these methods.

4.1 Static Timing Analysis

Static Timing Analysis is a method of modelling the performance of a circuit, without having to perform time based simulations or create test vectors. Elements within a circuit that can cause delays between an input signal transition and an output signal transition, such as cells or wires, are replaced by a single delay value which is propagated along the signal path. The propagation of these delays requires the ability to add delays that are in series, and to compare the delays through converging paths. Finding the
greatest delay from an input to node N3 in Figure 4.1 provides a very basic example of this concept using the steps described below:

1. Use the transition time at Input A and the load capacitance of node N1 to find delay $A \rightarrow N1$.
2. Use the transition time at Input B and the load capacitance of node N1 to find delay $B \rightarrow N1$.
3. Set delay U1 equal to the greater of $A \rightarrow N1$ and $B \rightarrow N1$.
4. Use the transition time at Input C and the load capacitance of node N2 to find delay $C \rightarrow N2$.
5. Use the transition time of the output of U1 and the load capacitance of N3 to find delay $N1 \rightarrow N3$.
6. Use the transition time of the output of U2 and the load capacitance of N3 to find delay $N2 \rightarrow N3$.
7. Set delay U3 equal to the greater of $U1 + (N1 \rightarrow N3)$ and $U2 + (N2 \rightarrow N3)$.

The process of establishing which path represents the longest or shortest delays between inputs, registers or outputs relies on the ability to compare delays (to determine maximum or minimum) and to add delays. This process is relatively simple when each cell is represented by a single delay value, but becomes non-trivial for the majority of probability distributions. The previous chapter discussed a method of characterising cells under realistic conditions and the necessity of producing cell libraries in a standardised format, this section continues this principle by using a commercially available STA tool that follows these standards.
4.1.1 Standardisation using PrimeTime

PrimeTime is an industrial standard timing analysis tool, capable of performing STA, SSTA, power analysis and model extraction [71]. The user provides a gate level netlist of a design to be analysed, a cell library from which to determine timing and power information and a set of constraints within which the design is required to perform. PrimeTime uses the specified cell library to establish the delays through the netlist and to verify that these delays meet the minimum setup and hold times of internal registers. These delays can be used to generate timing reports, where any paths that are too long for the current design constraints can be identified to the designer. This process is performed after a design has been synthesised to a gate level description, where each element in a design has a reference within the cell library, rather than abstract behavioural models. Any significant faults within the timing of the design can then be identified and used to target the synthesis tool to spend extra effort on this section of the design. If the timing analysis is acceptable then the results can be used to guide a Place & Route tool which will create a description for the layout of the chip, including the lengths and thicknesses of the interconnecting wires between gates. Details of this layout, including the parasitic effects of interconnect capacitances and resistances, can then be fed back into PrimeTime to verify that the timing constraints are still met. This process of analysis, combined with equivalence checking and logic simulation form the
process of signing off a design for fabrication. Design constraints include factors such as the clock frequency and the expected transition times at the circuit inputs. Timing models can be extracted or imported to represent interconnecting intellectual property blocks. Figure 4.2 provides an illustration of this synthesis and timing analysis flow.

4.1.2 Example of STA

Circuits are synthesised from a behavioural description or register transfer level description to a structural hardware description language (HDL) format using tools such as Synopsys Design Compiler [72] and a chosen cell library. This gate level form of the design contains a list of circuit elements, the connections between them and which cell models within the library represent them. A trivial example of a gate level Verilog netlist is given in Figure 4.3, where a design named DUT represents the circuit in Figure 4.1, consisting of two NAND gates (with instance names U1 and U3, inputs A and B and output Y) and an inverter (instance U2, input A and output Y). The words NAND and INV refer to the logical gates within the standard cell library from which the design was synthesised, specifying the names of the input and output pins, the logical function of the gate, and the characterised timing and power information. In this netlist the two NAND gates have been synthesised from the same model within the library, and hence reference the same timing and power model.

```verilog
module DUT (A B C N3);
  input A,B,C;
  output N3
  wire N1 N2;
  NAND U1 (.A(A) .B(B) .Y(N1));
  INV U2 (.A(C) .Y(N2));
  NAND U3 (.A(N1) .B(N2) .Y(N3));
endmodule
```

Figure 4.3: Example of a Verilog gate level netlist

The example is continued in Figure 4.4 where the gate level netlist is passed to Prime-Time together with the standard cell library, producing a timing report of the simple design. The timing report compares the propagation delay through a path of standard cells against a timing constraint such as the period of the clock and the setup and hold times of registers. If the path delay is in excess of the requirements then the path is said to have violated timing, and represents a situation where the circuit will not perform at the desired speed. The difference in path delay and the path timing constraint is referred to as slack, where negative slack represents a timing violation and positive slack suggests that the path is performing faster than required.

The input and output nodes of a circuit can be assigned with driving and loading cells (defined within the cell library), allowing realistic operating conditions to be applied to
the circuit, if the interface logic of a connecting circuit is known. A simpler method also allows for a specific transition time to be assigned to input nodes, and specific capacitances to be assigned to output nodes, as shown in Figure 4.5. If an input slew and output load pair are chosen to match indexes within the cell library, then the propagation delay and output transition of the cell, as calculated by STA, will exactly match the values written in the cell library look up table. If values of input slew and output load are chosen between the specified indexes, then the measurements are interpolated from the look up table. For paths with more than a single gate, the output load of a cell is determined by the input capacitances of the cells to which it is connected, combined with any parasitics. Maximum (and minimum) delays between various points in a design can be specified, constraining paths to operate within a certain time frame. For this simple example there are no registers within the design, and so a constraint is simply placed on the time allowed for a signal to pass from the inputs of DUT to the outputs. Clock specifications can be generated for sequential designs, where the name of the clock specification, the clock period, and the name of the input pin is provided. A simple example of a clock specification is given in Figure 4.5 for completeness, even though there are no sequential elements within the example netlist.

An example timing report is provided in Figure 4.6, where all values are given in nanoseconds. The example shows the delay of the timing path from input port A through the
two NAND gates to output port \textit{N}3, including the delay at each point in the path and the increase in delay at each point in the path. PrimeTime combines the user defined definition of the external delay (and transition time) at the input port and the model of the NAND gate within the cell library to predict the length of time taken to propagate a signal to output port \textit{N}3. The \textit{f} and \textit{r} characters represent falling or rising signal transitions respectively. The path delay, or data arrival time, is then compared with the required timing constraints, providing a measure of slack. The required time in this simple example is a user defined constraint from inputs of the design to outputs of the design, but in real circuits the required time will be based on a combination of clock period, clock latency, input/output delays and setup/hold times. PrimeTime can be used to report a number of paths, including the maximum delays (for setup time violations) and minimum delays (for hold time violations), highlighting which sections of a design may require optimisation. The reports can also be customised to show transition times of each cell in the path, as well as clock arrival times, fanout and load capacitance. This extra information can be used by the designer to determine if long delays are caused by weak cells driving large loads, or unbalanced clock trees where the clock pulse is arriving too early at a destination register.

**Figure 4.5:** Example of design constraints

```
set_input_transition 0.1682 all_inputs
set_load 0.25397 all_outputs
set_max_delay -from [all_inputs] -to [all_outputs] 0.27
create_clock -name CLOCK -period 5.0 CLK_INPUT_PIN
```

**Figure 4.6:** Example of a timing report

<table>
<thead>
<tr>
<th>Point</th>
<th>Incr</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>input external delay</td>
<td>0.10</td>
<td>0.10 \textit{f}</td>
</tr>
<tr>
<td>A (in)</td>
<td>0.00</td>
<td>0.10 \textit{f}</td>
</tr>
<tr>
<td>U1/Y (NAND)</td>
<td>0.07</td>
<td>0.17 \textit{r}</td>
</tr>
<tr>
<td>U3/Y (NAND)</td>
<td>0.07</td>
<td>0.24 \textit{f}</td>
</tr>
<tr>
<td>N3 (out)</td>
<td>0.00</td>
<td>0.24 \textit{f}</td>
</tr>
</tbody>
</table>

*data arrival time* 0.24

*data required time* 0.27

*slack* 0.03

Certain timing constraints are estimated by the designer at the initial synthesis stage, such as the amount of time it may take for a clock pulse to arrive at a register or the length of interconnecting wires between two cells on a path. These estimates are replaced
by more accurate values during the floor-planning and placement and routing of a design, where the information is extracted and fed back into the timing analysis tool. If the estimates are omitted from the constraints of the synthesis flow then the performance of the design may significantly decrease when routed, if the constraints are too pessimistic then unnecessarily large amounts of runtime may be consumed synthesising the design for an unrealistic goal. In the simple example of circuit DUT within this section the delays through U1 and U3 share same look up tables within the standard cell library because the same timing model is provided for each gate. The absence of the modelling of statistical process variations means that designers are forced to either omit these factors, or use corner analysis to provide pessimistic estimations, neither of which are ideal solutions.

During static timing analysis the only references to the timing models within the cell library are made within the gate-level netlist of the design. Altering the cell references within a netlist would force PrimeTime to read alternative cell models and produce different timing reports. If each instance of a cell could be altered to reference a separate cell model then it would be possible to model statistical process variations.

### 4.1.3 Monte Carlo Static Timing Analysis

The references to cells within a standard cell library can easily be altered to make use of the multiple cell models within a variation cell library. A simple script can be used to read a variation cell library and determine how many instances of each variation cell are available, this information can then be used to replace each standard cell reference with a variation cell instance between 0 and N (where N is the number of variation cells available -1). Three examples of randomised versions of the example gate level Verilog netlist are given in Figure 4.7, where the design named DUT now consists of three gates each with randomly selected variation cell models. A randomised netlist has exactly the same function and structure as the nominal netlist, with the exception that the cells within the design contain some atomistic differences that were originally modelled at the transistor level. The term randomised in this case refers only to the statistical process models selected to represent each gate.

The multiple randomised netlists can be passed to STA tools, providing timing measurements for each separate netlist. These separate timing reports can be combined to form distributions of delays for each measured path, providing designers with a spread of performance, rather than a single prediction. Examples of variation timing reports are given in Figure 4.8, where reports for the same path through the three variation instances of DUT are compared. The start points and endpoints of the paths are all the same and the same path structure is provided in each report, but the individual gate delay of cell instances U1 and U2 are changed due to the differences in NAND timing models within the variation cell library. The cell timings have different impacts on the
### INSTANCE 1:

module DUT (A B C N3);
    input A,B,C;
    output N3
    wire N1 N2;
    NAND_123 U1 (.A(A) .B(B) .Y(N1));
    INV_47  U2 (.A(C) .Y(N2));
    NAND_93 U3 (.A(N1) .B(N2) .Y(N3));
endmodule

### INSTANCE 2:

module DUT (A B C N3);
    input A,B,C;
    output N3
    wire N1 N2;
    NAND_46 U1 (.A(A) .B(B) .Y(N1));
    INV_178 U2 (.A(C) .Y(N2));
    NAND_23 U3 (.A(N1) .B(N2) .Y(N3));
endmodule

### INSTANCE 3:

module DUT (A B C N3);
    input A,B,C;
    output N3
    wire N1 N2;
    NAND_35 U1 (.A(A) .B(B) .Y(N1));
    INV_69  U2 (.A(C) .Y(N2));
    NAND_11 U3 (.A(N1) .B(N2) .Y(N3));
endmodule

**Figure 4.7:** Example of randomised Verilog gate level netlists

arrival time at the output $N_3$, resulting in different amounts of slack being reported as some gates operate faster and some operate slower. The cell models referenced in randomised circuit instance 3 are slow enough to create a failing timing path where the data arrives after the required time, producing a negative slack report, Figure 4.8(c). This method of timing report allows for the analysis of variation within a single path of a design, comparing the variation in timing of the whole path, and individual gates within the path. If it can be seen that the variation of a gate within a path is a leading contributor to the overall variation within the path then it may be possible to identify this gate and perform a correction. This could involve replacing the cell with one of a greater drive strength, or attempting to reduce the input transition time or output load of the cell, as these are factors which were found to increase variations in cell performance within Section 3.5.
### Chapter 4 Monte Carlo Circuit Analysis

**Figure 4.8:** Example of timing reports for the same path through randomised circuit instances

This process of performing STA on multiple samples of netlists containing statistical variations is referred to as Monte Carlo Static Timing Analysis (MCSTA), a summary of which is given in Figure 4.9.
4.1.4 Repeatability

Randomisation is a key step within both the processes of characterising variation cells and generating variation circuits, it is the method by which models and samples are chosen. This randomisation poses a problem when trying to reproduce data that has been generated from previous runs, or when allowing others to produce the same data within their own experiments. An important step within MCSTA is the recording of the values used to seed the random number generation, as this allows the same seed to be fed into the process at a later date, and the exact same results be produced. The recorded seed values can be used to trace a path from the STA results of the randomised netlist, to the variation cell library models that were used and to the original transistor model cards that were used within RandomSpice. This audit trail can be used to recreate specific results at a later point in time, without storing unwanted excess information from multiple simulation runs. This aids in the process of generating a transparent process to designers, where the variations in the timing path of a circuit can be traced right back to the variations in the performance of a single transistor if necessary.
4.1.5 Trial Results

A proof of concept trial of the proposed MCSTA method was carried out using a section of an IC developed within the SpiNNaker project [73]. The SpiNNaker project aims to produce a neural network architecture of processor cores which mimics the biological structure and functionality of the human brain. The proposed architecture consists of a toroidal mesh of multi-core processors, each comprising of 20 processing cores.

The SpiNNaker project shared academic and industrial partners with the nanoCMOS project and provided the opportunity to combine resources and perform variation analysis on a real world design. These opportunities included access to sections of a SpiNNaker core, and the commercial cell libraries and technology files used for fabricating the cores. The initial fabrication of the SpiNNaker architecture was to take place on a commercial 130nm CMOS process.

A section of a SpiNNaker processor core, referred to as the Communications Controller, was provided to the nanoCMOS project as a gate level netlist with post layout and routing parasitic data. The netlist was originally synthesised from a standard cell library of over 600 cells, but this was reduced to a collection of eighteen cells to allow for a more rapid generation of a variation cell library. The SPICE netlists for the eighteen standard cells were passed to RandomSpice and 200 variation instances were generated for each of the standard cells.

The original 3D atomistic simulations that were carried out by nanoCMOS project members for the generation of RandomSpice transistor model cards (described in Section 3.3) were performed using a 35nm technology, and not for the 130nm technology that was made fully available to the project. This meant that accurate variation models could be used but the standard cells and parasitics were not yet available for circuit analysis. It was decided that the generation of new, simplified, 130nm variation models would provide a suitable method for an initial analysis of circuits using a real 130nm technology, rather than attempting to generate a full custom standard cell library at 35nm.

The original 35nm model cards made use of seven BSIM model parameters to replicate the atomistic simulation data, but the impact of variations at 130nm were found to be less complicated (as expected) by the developers of the atomistic simulator. Instead a single BSIM model parameter was modified, the threshold voltage $V_T$. The variations in threshold voltage were modelled using a normal distribution, and although RandomSpice was initially used in this simple manner, it had the capability to model variability in an arbitrary number of BSIM parameters with generally non-Gaussian parameter distributions.

The effects of statistical process variations on the 130nm technology were artificially injected into the transistor models by sampling values of $V_T$ for each transistor from normal distributions. The mean values for $V_{TN}$ and $V_{TP}$ were approximately 305mV and...
380mV respectively, where $V_{TN}$ is the threshold voltage for NMOS transistors and $V_{TP}$ is
the threshold voltage for PMOS transistors. The standard deviations were initially found
by the 3D atomistic simulator to be approximately 10% of the means, 30.5mV and 38mV,
and the process of generating these distributions was automated by the RandomSpice
tool. The area, footprint and parasitics of the standard cells remained unchanged.
These variation cell instances were characterised using Synopsys Liberty NCX library,
generating a 130nm variation cell library. The re-synthesised Communications Controller
consisted of 4917 instances of the 18 characterised standard cells, including a range of
buffer cells, two forms of d-type flip flops, a 2-input multiplexor, a transparent latch,
and small combinational logic.

The gate level Verilog netlist of the Communications Controller was used to produce 1000
randomised gate level Verilog netlists, where references to cells within the standard cell
library were replaced with randomised references to the variation cell library. These
randomised gate level Verilog netlists were passed to the PrimeTime suite, from which
timing reports were generated for paths that did not meet the timing requirements of
the design.

The SpiNNaker project also provided the nanoCMOS project with a SPICE netlist of the
Communications Controller, where a combination of the SPICE netlists of the standard
cells from which the design was synthesised and resistance and capacitance values from
the post routed design were used to describe the circuit. This SPICE level netlist was
also passed to RandomSpice, and 1000 randomised SPICE netlists were generated for
the entire Communications Controller. This large netlist could not be fully characterised
by using Liberty NCX, and instead a small number of test vectors were used to stimulate
a limited number of logic paths, from which timing measurements were recorded.

4.1.5.1 Critical Path Analysis

The first step within this trial was to assess the impact of randomising the input netlists
on the accuracy of existing STA tools. The timing measurements from the SPICE
level simulations of the Communications Controller were compared with PrimeTime
timing reports for the same logic paths. The range of timing results obtained from
SPICE and MCSTA for the same timing path are compared in Figure 4.10. The
direct comparison of the two forms of measurement in Figure 4.10(a) revealed that
SPICE and STA delays matched within 5% and the correlation plot in Figure 4.10(b)
provides a strong correlation coefficient of 0.97. These comparisons indicate that changes
in transistor variations modelled within the VCL can be accurately used to capture
transistor variations within the SPICE level netlist.

The above comparison between SPICE simulations and MCSTA of the Communications
Controller focused upon the same fixed timing path through the design, as a single
test vector could be used to stimulate the SPICE netlist, and this same path could be extracted from within the STA tool. The fundamental principle of STA is that the static analysis prevents the need for the design and application of test vectors, allowing the longest path delays to be extracted without producing a simulation for each separate path. The use of SPICE simulations to stimulate every single possible timing path and determine the longest is prohibitive, and so while the above test determined the accuracy and correlation between SPICE and STA, it did not represent the critical paths within the design. Figure 4.11 provides a histogram of the longest path delay through the Communications Controller for the 1000 randomised variation netlists, in which a similar skew in the distribution can be seen as from the distributions of delays

![Graph of path delays comparing Hspice and Primetime STA](image)

**Figure 4.10:** Comparison of HSPICE and Prime/Time STA timings for the same single critical path in 1000 randomisations of the source netlist.
within single characterised cells (Figure 3.7). The skew in the distribution of critical path delays of the circuit suggests that the skews in the delay distributions of individual cells are not averaging out to produce normal distributions, and would therefore not be correctly produced from normal distribution models.

![Histogram of Critical Path Delays](image)

**Figure 4.11: Histogram of Critical Path Delays**

The Lilliefors test was used on the distribution of critical path delays, and the null hypothesis that the distribution originated from a normal distribution with unknown mean and standard deviation was rejected with a significance level of 5%. Confirming that non-normal distributions of transistor performance, which lead to non-normal distributions of standard cell delays produce non-normal distributions of critical path delays in circuits. This supports the motivation for the research within this thesis, that simple statistical models and assumptions of Gaussian distributions within transistor, standard cell, or circuit models can produce inaccurate estimations of the extremes of circuit behaviour. MCSTA is an alternative method that can be used to produce estimations of circuit performance without requiring any such assumptions about the distributions of transistor or standard cell distributions.

### 4.1.5.2 Circuit Performance Predictions

The second step within this trial was to obtain estimations of the percentage of circuits that would pass timing requirements when fabricated, and to observe how this percentage altered with changes in clock frequency. This was achieved by performing STA over a range of clock frequencies above and below that at which the original netlist failed timing requirements, for 25000 randomised netlists. Recording the maximum frequency passed
for each randomised netlist in this set of analyses provided the data required to plot the percentage of circuits that passed timing requirements versus the clock period (Yield versus Performance).

![Graph](image)

(a) Histogram of netlists failing timing requirements for a range clock frequencies

![Graph](image)

(b) Yield Vs Clock Frequency

**Figure 4.12:** Probability of the source netlist meeting timing requirements for a given clock frequency. The clock frequency at which traditional STA finds the yield to be 100% is also given

Figure 4.12(a) provides a histogram of the number of randomised netlists failing to meet timing requirements for a range of clock frequencies Figure 4.12(b) is a plot of the same information but as a reversed cumulative distribution function, showing the reduction in percentage yield of the circuit as the clock frequency is increased. These results were obtained for a fixed supply voltage and operating temperature (two of the most commonly used process corners), but include the statistical process variations in threshold
voltage. Timing analysis of the original netlist using the standard cell library produced a nominal operating frequency of 199.8MHz, while this analysis predicts that 95% of circuits would fail at this frequency. A 100% yield was found at operating frequencies of less than 197MHz, which only represents a small decrease in clock frequency of 1.5%. This implies that the statistical process variations modelled within this trial have had little impact on the operating frequency of circuits fabricated with the 130nm process, which is in line with expectations from circuits manufactured by the industry at this technology node.

Having successfully modelled the impact of statistical process variations within a large technology node, it was then important to predict whether this impact would grow significantly for future, smaller technology nodes.

### 4.2 Power Analysis

Power analysis defines the methods of predicting both static and dynamic power consumption of a circuit, within a set of defined operational parameters. These parameters can be used to narrow the analysis to a specific mode of circuit operation, or an average of all possible circuit behaviours. Power consumption information for elements within a circuit that consume power, such as standard cells and interconnecting wires, are read from standard cell libraries and extracted parasitic information. These readings are combined with signal transition patterns generated from simulations, or definitions of input switching activity, generating an estimation of the power consumed by the circuit under the defined behaviour.

Two of the most commonly used formats for describing the behaviour of a circuit for power analysis are the Switching Activity Interchange format (SAIF) and Value Change Dump (VCD) format. SAIF contains counts of the number of times a signal changes during a period of analysis, and the length of time a signal remains at a certain level. VCD files contain the order and timings of signal changes as they occurred during the simulation from which the VCD was generated.

The switching activity information is propagated through the circuit, using a combination of timing and power measurements within a standard cell library. The switching activity simply includes the order in which inputs and outputs transition, the timing information is used to establish when the transitions occur, how long these transitions take to complete, and the length of time between transitions that the circuit remains in a static state.

The leakage power of the cell can be read from a single value for each static state in the operation of a cell. These separate static states may be combined into a single average leakage value for some simple cells. These leakage values are simply read from
the cell library and combined to form the leakage power of the circuit during a period of inactivity.

The dynamic power of a cell is calculated in PrimeTime if switching activity or test vectors are provided. The switching activity is used to sum the energies of each signal transition within the circuit, and the power is then calculated by dividing this energy by the time window over which the power analysis is performed. Cell input transitions that result in an output transition contribute to the total switching power of a circuit, while input transitions that do not result in an output transition count towards internal power consumption.

Generating a VCD file from a test-bench that contains a single input transition for a single gate provides a simple demonstration of how the dynamic energy within the cell library is used. Calculating the power for a single transition over a time window of 1s (for index values of slew and load) returns a measurement of Cell Internal Power that exactly matches the internal power defined within the cell library. The Cell Leakage Power measurement is unaffected by the choice of time window size. The Net Switching Power represents the power required to drive the load capacitance of each cell.

4.2.1 Example of Power Analysis

An example of a power analysis report is given in Figure 4.13, where separate values for switching, internal and leakage power are provided to the user. These values will change depending on the activity file chosen, and the time frame within the activity file used for power analysis. Sections of the activity file with little or no input transitions will be heavily dominated by leakage power measurements, while activity files with unrealistic test patterns will not reflect the power consumption of the circuit as it would be used by the end consumer.

\[
\begin{align*}
\text{Net Switching Power} & = 3.657e-13 \ (0.37\%) \\
\text{Cell Internal Power} & = 1.621e-15 \ (0.00\%) \\
\text{Cell Leakage Power} & = 9.867e-11 \ (99.63\%) \\
\text{Total Power} & = 9.904e-11 \ (100.00\%)
\end{align*}
\]

Figure 4.13: Example of PrimeTime PX Power Report

4.2.2 Monte Carlo Power Analysis

VCD and SAIF files refer to signal and component names within a design, as opposed to the cell library references. This significant fact allows the method described for Monte Carlo Timing Analysis to be used for Monte Carlo Power Analysis, as alterations in the
names of cell library references will still allow the same power analysis constraints to be used during variation analysis.

References to standard cell library cells within a gate level netlist can be altered to reference the multiple cell models within a variation cell library. Multiple netlists with randomised variation cell library references can be passed to power analysis tools such as PrimeTime PX, providing power consumption predictions for the same test patterns for each separate netlist. These separate power reports can then be combined to form distributions of leakage, internal and switching power for the design, allowing designers to assess the range of possible power consumption for a circuit performing different operations.

4.2.3 Trial Results

A VCD file containing a set of signal transitions for a single test vector of the Communications Controller was used during power analysis of the 1000 randomised netlists that were used in the trial timing results (Section 4.1.5). The VCD file contained details of the times at which inputs switched, and the order in which the switches occurred, this information was propagated through each of the variation netlists by PrimeTime PX, producing estimates for the dynamic and static power consumption of the circuit. This is believed to be one of the earliest attempts to model the impact of statistical process variations on the power consumption of a circuit.

Figure 4.14 shows a histogram generated from the distribution of leakage power measurements for the 1000 randomised Communications Controller netlists, in which the same characteristic skew is present as was found within the delay measurements of the circuit. Figure 4.15 provides a similar skewed histogram of the dynamic power consumption of the circuit, but significant gaps are revealed between each bar within the histogram. These gaps are attributed to the number of significant digits that the power consumption estimates are reported to by PrimeTime, which in this analysis forced each value to be rounded to the nearest uW.

The Lilliefors test was used on the distributions of static and dynamic power consumption, and the null hypotheses that the distributions originated from normal distributions with unknown means and standard deviations were both rejected at the 5% significance level. This suggests that the non-normal distributions of transistor performance and standard cell power consumption produce non-normal distributions of static and dynamic power consumption in circuits, and that power consumption predictions based upon normal distributions would be inaccurate. The proposed method of Monte Carlo Circuit Analysis represents a unique approach to predicting the impact of statistical process variations upon the power consumption of circuits, which should generate accurate results for modern and future technology nodes, as no assumptions about the
statistical distributions of standard cell and circuit performance have to be made during this method.

### 4.3 Summary

The standard practice of performing Corner Analysis has been described as being overly pessimistic, especially when modelling intra-die process variations. Such pessimism can
lead to longer, more costly, design stages, or may force design teams to abandon the optimisation of a design before maximum performance is achieved. The process of MCSTA has been introduced as a method of reducing this pessimism, sampling over a range of possible cell delays, rather than assuming that all transistors within a chip simultaneously perform at their worst possible behaviour.

The accuracy of MCSTA was demonstrated as being within 5% of the delays obtained from Monte Carlo SPICE simulations, when performed on a section of a SpiNNaker test chip. This analysis indicates that STA can be used to accurately model SPICE, and that the distribution of a Monte Carlo SPICE simulation can be replicated by a Monte Carlo STA run, but has not yet highlighted the significant improvements that MCSTA offers over Corner Analysis. These improvements are documented in the following chapter.
Chapter 5

Critical Assessment

The process of modelling statistical process variations within circuits was described and tested within Chapter 4, but the necessity for such a process has not yet been demonstrated. This chapter describes the steps taken to verify the accuracy and efficiency of the proposed Monte Carlo Circuit Analysis methods, with comparisons against Monte Carlo SPICE simulations and existing commercial STA and SSTA techniques.

Preliminary testing involved the use of a 130nm standard cell library, as access was provided to the technology at an early stage of the research, which was then used to predict the impact of statistical process variations on future technology nodes. These tests were then repeated using a 35nm technology as the data became available.

5.1 Comparison with Corner Analysis at 130nm

One of the greatest arguments against the continuing use of Corner Analysis for the signing off of the performance of designs, is the potential for large amounts of pessimism generated by simultaneously setting every process corner to its worst possible value. The amount of this pessimism must be quantified in order to assess whether MCSTA can provide a more accurate alternative to the existing industry standard, and so an experiment was established to compare these two methods. The methods will be assessed for their ability to predict circuit delays, power consumption and the percentage of circuits that will meet target constraints, using Monte Carlo SPICE simulations as a reference.

5.1.1 Test Circuit

It was decided that the initial investigations of the effects of MOSFET variability on circuit delays, power consumption and yield should be based upon a simple circuit, such
as a one bit full adder, Figure 5.1. The choice of such a small circuit allowed for very large scale statistical SPICE simulations of the transistor level netlist, including parasitics and full test vector coverage. The ability to perform such comprehensive SPICE simulations of the test circuit allowed for the generation of large amounts of reference data against which both MCSTA and Corner Analysis could be compared. The post place and routed test circuit consisted of 13 gates from 4 standard cells (inverter, NAND, OR and buffer) and contained a total of 52 transistors.

The combined use of a small test circuit and a well understood fabrication process has allowed for the extensive validation of the modelling of random process variations at the transistor, cell and circuit levels.

5.1.2 Method

The effects of statistical process variations on the performance of the test circuit were measured and compared using three methods: large scale Monte Carlo transistor level SPICE simulations, a traditional corner based analysis, and the proposed MCSTA method. The large scale Monte Carlo SPICE simulations were performed as a reference against which the accuracy of Corner Analysis and MCSTA could then be compared. This section describes the steps taken to setup and perform these methods.

5.1.2.1 Monte Carlo SPICE Simulations

The only method of determining the slowest path delay through a SPICE netlist is to stimulate every individual path with input test vectors, measure the propagation delay from each input to output transition, and then compare the separate delays after the simulation is complete. A testbench was created to achieve this for the adder which included each of the 24 possible state transitions, while only a single input was switched at a time, Figure 5.2. A Grey code input pattern would not be sufficient as this would cycle through the 8 possible logic states of the adder, but only in a single direction. The circuit delay and power consumption of transitioning from state $000 \rightarrow 001$ (for inputs $A, B$ and Carry In) would not be the same as for the transition from state $000 \rightarrow 010$.
or \(000\rightarrow 100\). Each state can transition to three other states (if only a single input is switched at a time), producing the 24 possible state transitions required for a complete characterisation of the adder circuit. A 10ns delay was used between each input transition, allowing a large period of time to measure the leakage current before the input transition started and after the output transition had completed and settled to a steady state. Each 10ns window was given a unique set of measurement commands, recording the propagation delay from input to output, the output transition time, the leakage current and the energy consumed from the start of the input transition to the completion of the output transition.

**Figure 5.2:** Input waveforms for Monte Carlo SPICE simulations of 130nm Adder circuit
The transistor level SPICE netlist of the adder was generated by combining the SPICE level netlists of each of the four standard cells in the schematic of Figure 5.1, where each transistor model name was replaced with the ATOMN and ATOMP key words required for use within RandomSpice. RandomSpice was then used to generate 10,000 randomised transistor level netlists of the adder circuit, replacing each MOSFET model instance with BSIM models randomly selected from model cards generated for the 130nm technology node. Although in this particular study RandomSpice was used to randomize only the threshold voltage \( V_T \), it has the capability of injecting variability in an arbitrary number of BSIM parameters with generally non-Gaussian parameter distributions. The 10000 randomised SPICE netlists of the Adder were simulated with the testbench described above, producing a distribution of delays and power consumption for each of the 24 input transitions.

5.1.2.2 Corner Analysis

Performing SPICE level simulations of circuits at the transistor level is exceedingly time consuming, and full test vector coverage becomes prohibitive for circuits any larger than the simple adder tested. The traditional approach to combating this is to perform Corner based Static Timing Analysis (STA) using Standard Cell Libraries (SCLs). Commonly used cells are characterised using SPICE simulations at multiple sets of process and environmental parameters (such as operating temperature and supply voltage) generating multiple SCLs. The combinations of parameters are typically chosen to be at the extremes or corners of each parameter, hence the term Corner Analysis. It is assumed that the behaviour of the circuit is guaranteed at any point within the box created by performing STA at these corners [12].

Statistical process variations within the adder circuit were modelled by generating standard cell libraries at the corners of the \( V_T \) distribution used within the Monte Carlo SPICE simulations. These corners were set at \( \mu + 3\sigma \) and \( \mu - 3\sigma \), as the area under a normal distribution between \( \pm 3\sigma \) represents approximately 99.7% of the possible values. This implies that 99.7% of all possible transistor performance would be represented by the extremes of the performance of transistors at \( \pm 3\sigma \).

Copies of the SPICE level cell netlists of the four standard cells used within the adder test circuit were created and the threshold voltage parameters of the transistor models were set to the \( \pm 3\sigma \) values. Liberty NCX was then used to characterise these cells into two standard cell libraries (+3\( \sigma \) and −3\( \sigma \)), and static timing analysis was performed on the adder circuit using both of these libraries, producing a circuit analysis at each corner. The pattern of input transitions used within the Monte Carlo SPICE simulations was translated into a VCD file so that the same power analysis could be performed at each corner as was used during the Monte Carlo SPICE simulations.
While it is assumed that 99.7% of the statistical process variations would be captured between the ±3\(\sigma\) corners, the probability of every transistor within a cell being simultaneously at ±3\(\sigma V_T\) is exceedingly small. This probability is further reduced by the assumption that every instance of every standard cell within a circuit is simultaneously set at the same \(\sigma V_T\) level. Corner Analysis is therefore likely to produce very pessimistic predictions of circuit yield in comparison with the Monte Carlo SPICE simulations.

5.1.2.3 Monte Carlo Static Timing Analysis

The proposed method of MCSTA is a compromise between the accuracy of Monte Carlo SPICE simulations, and the speed and practicality of STA. This first required the one off generation of a Variation Cell Library (VCL), where RandomSpice was used to generate multiple randomised SPICE netlists of each standard cell, rather than the full adder circuit. The randomised cell netlists are then passed into Liberty NCX, generating a VCL with multiple instances of each cell as described in Section 3.5. These instances reflect the atomistic differences between transistors and provide a mechanism for modelling statistical variation within STA.

It was found that although variations in \(V_T\) were modelled with normal distributions at the transistor level, the distributions of cell delay, output transition and power consumption that these variations create at the cell level are not normal. It was therefore insufficient to assume that small sample sizes of each standard cell within a VCL would provide accurate representations of the overall power and delay distributions, but the use of overly large sample sizes would lead to prohibitive characterisation run times. A trial was created to compare the distributions of power and delay that were represented by the selection of different sample sizes.

A large collection of 40,000 randomised 130nm inverter cells was generated by using RandomSpice, these instances were then characterised using Liberty NCX creating a VCL consisting solely of inverters. The process was then repeated for eleven different sample sizes, ranging from 100 to 10,000 inverters.

The investigation into the number of randomised cells to include in a VCL indicated that the mean and standard deviation of the distributions of cell delays tended to converge for sample sizes of over 400, to an error of less than 1% and 3% for the mean and standard deviation respectively. A plot of this convergence is shown for cell delays with a fixed input transition and output load in Figure 5.3. A similar plot of the percentage error in mean and standard deviations of power distributions, Figure 5.4, reveals that sample sizes of only 100 produce an accurate estimation of the sample mean (within 0.25%), but the error in standard deviation ranges between 0.5% and 3.5% for small to large sample sizes. This seemingly high level of inaccuracy in the standard deviation may simply be due to the standard deviation being measured in femtojoules, with errors between the
Figure 5.3: Reduction of error in the mean and standard deviation of delay distributions with increases in sample sizes

distributions being of the order of less than a quarter of a femtojoule. A comparison between the distributions generated from a large collection (40,000) of inverters and smaller sample sizes was performed by generating quantile-quantile plots of delay and power measurements, these are shown in Figure 5.5 and Figure 5.6. The results from these characterisation runs suggested that a sample size of 500 variation instances for each standard cell would produce distributions that closely resemble the shape (skew and kurtosis) of distributions generated by much larger sample sizes. 500 variation instances of each of the four standard cells within the adder circuit were generated using RandomSpice, and Liberty NCX was used to characterise them forming a VCL.

The gate level Verilog netlist of the adder circuit was then randomised using a simple perl script to replace each standard cell reference with a randomly selected variation instance (of same logic type) within the VCL. 10,000 randomised gate level netlists were then analysed using PrimeTime, producing timing reports of each path through the design. The same input stimuli used within the Monte Carlo SPICE and Corner Analysis methods were used to produce power consumption reports for each of the randomised netlists. The values used to seed the randomisation of each netlist and each cell instance were recorded, creating an audit trail that allowed each individual STA result to be reproduced at a later time. The use of the existing commercial STA tool allowed for the statistical analysis to be combined with the strengths of the industry standard sign-off tool.
5.1.3 Further sigma VTs

Predictions of the impact of further levels of variation were also investigated by varying the level of $\sigma_{VT}$. Seven levels of injected variation were investigated, where $\sigma_{VT}$ was set at 10%, 15%, 20%, 25%, 30%, 40% and 50% of $V_T$. The three methods of analysis (Monte Carlo SPICE simulations, Corner Analysis and MCSTA) were repeated for each of these levels of variation.
5.1.4 Results and Analysis

This section begins with a comparison between the results generated by Corner Analysis and MCSTA, and continues with the analysis of critical paths and predictions of power, performance and yield margins.
5.1.4.1 Circuit Level Statistical Variation

The results from the use of Corner Analysis are shown in Figure 5.7. The green lines represent the maximum power and delay predicted for 99.7% of manufactured circuits at different levels of statistical variability. A scatter plot of the Monte Carlo SPICE simulation results at the highest level of injected variability are also included as a reference. The Corner Analysis prediction for maximum path delay at the highest level of variability is nearly 5.5 times greater than the largest delay generated by Monte Carlo SPICE simulations. The absolute error in yield prediction is nearly 550% which shows that Corner Analysis is incapable of accurately predicting the effects of large levels of statistical variation.

The results from the use of MCSTA are shown in Figure 5.8, which includes scatter plots of power consumption and maximum delay through the adder circuit, for each MCSTA and SPICE run at each level of variation. The same results are presented within Figure 5.9 and Figure 5.10, as histograms of power consumption and path delays. The cumulative distributions of delays are compared in Figure 5.11, from which predictions of yield can be made for different values of the system clock period. A comparison between the predictions made by the three different methods for power consumption and maximum delay required for a 99.7% yield are given in Tables 5.1 and 5.2 respectively. The constraints predicted by MCSTA to meet 99.7% yield are found to be within 1.2%
Figure 5.8: A scatter plot of the average power per input transition against the maximum path delay through the adder. These data were generated from 10,000 MCSTA and SPICE runs for each level of injected variation. The test sequence did not include simultaneous input switching. The shape and location of the distributions match significantly better than the predictions made by Corner Analysis.

and 2.9% of SPICE for power and delay respectively. This is a huge improvement over Corner Analysis where errors reach 782% and 548% for power and delay respectively.

Table 5.1: Power measurements from SPICE, MCSTA and Corner Analysis at 99.7% Yield. Errors are absolute percentage errors with respect to the SPICE predictions

<table>
<thead>
<tr>
<th>$\sigma V_T$</th>
<th>SPICE Power(uW)</th>
<th>Corner Analysis Power(uW)</th>
<th>Error(%)</th>
<th>MCSTA Power(uW)</th>
<th>Error(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>24</td>
<td>25.33</td>
<td>5.54</td>
<td>23.76</td>
<td>1</td>
</tr>
<tr>
<td>15%</td>
<td>24.04</td>
<td>26.82</td>
<td>11.56</td>
<td>23.79</td>
<td>1.04</td>
</tr>
<tr>
<td>20%</td>
<td>24.11</td>
<td>29.12</td>
<td>20.78</td>
<td>23.84</td>
<td>1.12</td>
</tr>
<tr>
<td>25%</td>
<td>24.16</td>
<td>33.25</td>
<td>37.62</td>
<td>23.9</td>
<td>1.08</td>
</tr>
<tr>
<td>30%</td>
<td>24.23</td>
<td>38.42</td>
<td>58.56</td>
<td>23.95</td>
<td>1.16</td>
</tr>
<tr>
<td>40%</td>
<td>24.41</td>
<td>78.18</td>
<td>220.28</td>
<td>24.12</td>
<td>1.19</td>
</tr>
<tr>
<td>50%</td>
<td>24.69</td>
<td>217.8</td>
<td>782.14</td>
<td>24.4</td>
<td>1.17</td>
</tr>
</tbody>
</table>

Statistical analysis of these distributions reveals that MCSTA and SPICE are in close agreement, with errors in the means of the distributions consistently below 1%, Table 5.3. There is a consistent error in the standard deviation of the power distributions of around 30% produced by interpolation and rounding errors within the STA tool when reading power information from the cell libraries. Further sources of error occur between
Figure 5.9: Histograms of the average power per input transition illustrate the close match between the shape and locations of the MCSTA and SPICE Power distributions. The frequency is the number of samples recorded for each power measurement.

Table 5.2: Delay measurements from SPICE, MCSTA and Corner Analysis at 99.7% Yield. Errors are absolute percentage errors with respect to the SPICE predictions.

<table>
<thead>
<tr>
<th>$\sigma V_T$</th>
<th>SPICE Delay (ns)</th>
<th>Corner Analysis Delay (ns)</th>
<th>Error (%)</th>
<th>SPICE Delay (ns)</th>
<th>Error (%)</th>
<th>MCSTA Delay (ns)</th>
<th>Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>0.51</td>
<td>0.64</td>
<td>24.57</td>
<td>0.51</td>
<td>0.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15%</td>
<td>0.52</td>
<td>0.73</td>
<td>41.59</td>
<td>0.52</td>
<td>0.39</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20%</td>
<td>0.53</td>
<td>0.84</td>
<td>58.72</td>
<td>0.53</td>
<td>0.19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25%</td>
<td>0.55</td>
<td>0.98</td>
<td>79.7</td>
<td>0.54</td>
<td>1.28</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30%</td>
<td>0.56</td>
<td>1.2</td>
<td>113.94</td>
<td>0.55</td>
<td>1.96</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40%</td>
<td>0.59</td>
<td>1.97</td>
<td>232.79</td>
<td>0.58</td>
<td>2.36</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50%</td>
<td>0.63</td>
<td>4.08</td>
<td>547.99</td>
<td>0.61</td>
<td>2.86</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MCSTA and SPICE due to the use of different SPICE tools when performing grid based Monte Carlo SPICE simulations and performing standard cell characterisation, these become more significant for delays at the highest variation levels.

MCSTA produces similar results to those of SPICE simulations at a fraction of the CPU time. A comparison of the time taken when using the two methods on a single desktop PC with an Intel Pentium D dual core CPU running at 3.40GHz, with 2GB of RAM is given in Table 5.4. Performance improved by a factor of over 40 for sample sizes of over...
Chapter 5 Critical Assessment

Figure 5.10: Histograms of the maximum path delay through the adder illustrate the close match between the shape and locations of the MCSTA and SPICE Delay distributions. The frequency is the number of samples recorded for each path delay or power measurement.

Table 5.3: Error between MCSTA and SPICE when generating distributions of delay and power consumption

<table>
<thead>
<tr>
<th>$\sigma V_T$</th>
<th>Absolute Percentage Error (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean Delay</td>
</tr>
<tr>
<td>10%</td>
<td>0.58</td>
</tr>
<tr>
<td>15%</td>
<td>0.67</td>
</tr>
<tr>
<td>20%</td>
<td>0.71</td>
</tr>
<tr>
<td>25%</td>
<td>0.94</td>
</tr>
<tr>
<td>30%</td>
<td>0.88</td>
</tr>
<tr>
<td>40%</td>
<td>0.72</td>
</tr>
<tr>
<td>50%</td>
<td>0.52</td>
</tr>
</tbody>
</table>
Chapter 5 Critical Assessment

Figure 5.11: A scatter plot of the average power per input transition against the maximum path delay through the adder. These data were generated from 10,000 MCSTA and SPICE runs for each level of injected variation. The test sequence did not include simultaneous input switching. The shape and location of the distributions match significantly better than the predictions made by Corner Analysis.

Table 5.4: A comparison of the time taken to perform SPICE and MCSTA simulations

<table>
<thead>
<tr>
<th>Samples</th>
<th>SPICE Time (s)</th>
<th>MCSTA Time (s)</th>
<th>Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>43</td>
<td>12</td>
<td>3.58</td>
</tr>
<tr>
<td>10</td>
<td>425</td>
<td>20</td>
<td>21.25</td>
</tr>
<tr>
<td>100</td>
<td>4250</td>
<td>100</td>
<td>42.5</td>
</tr>
<tr>
<td>1000</td>
<td>42500</td>
<td>960</td>
<td>44.27</td>
</tr>
<tr>
<td>10000</td>
<td>425000</td>
<td>9500</td>
<td>44.74</td>
</tr>
</tbody>
</table>

100. Larger circuits with greater numbers of test vectors will require significantly longer SPICE simulations, while MCSTA is performed statically without the need for test vectors and simulations. The benefits of using MCSTA therefore become even greater with larger, more complex, circuits.

5.1.4.2 Variation in Critical Paths

The number of paths identified as most critical for any given analysis increased from two to fifteen at the highest levels of variability tested, these paths are given in Table 5.5. The increase in the number of critical paths is shown in Fig 5.12, where the percentage chance of a path being identified as having the longest delay is given for each level of variation tested. This demonstrates the importance of including statistical information
within circuit timing analysis, as the ‘criticality’ of a path can change even within such a small, simple example as the 1-bit adder. This form of statistical critical path analysis can allow a designer to identify any cells that occur in multiple critical paths, focusing design efforts on areas of high sensitivity to variation.

5.1.4.3 Power, Performance and Yield

The cumulative distribution functions of the power and delay measurements can be combined to create three dimensional plots of power, performance and yield: Figure 5.13. This information can be used by designers to establish what trade-offs can be made between power consumption and maximum clock speed, in order to maximise the number of fabricated devices that perform within the required constraints.
Figure 5.12: The number of paths identified as having the longest delay through the adder circuit, for a given STA run, increased with variability.
Chapter 5 Critical Assessment

(a) Predicted Yield increases as clock and power constraints are relaxed.

(b) A 2D view showing contour lines at every 10% increase in yield.

Figure 5.13: Power Performance Yield plots for the Adder circuit at $\sigma V_t = 50\%$. Equi-yield contour lines are placed at 10% intervals. Predicted Yield increases as the constraints of the Clock Period and Power Consumption are relaxed.
5.2 Comparison with Statistical Static Timing Analysis at 35nm

The analysis of the performance of circuits at 130nm revealed that MCSTA could produce results similar to Monte Carlo SPICE simulations, but represented process variation in a technology node in which statistical process variation does not cause excessive amounts of variation in circuit performance. It was therefore necessary to model statistical process variations at a more significant technology node, for which 35nm was chosen.

The pessimism of Corner Analysis is widely recognised and acknowledged within the industry, and so it was also important to verify the effectiveness of MCSTA against a commercial tool designed to model process variations by performing SSTA.

This section provides details of the analysis of several circuits at 35nm, using Monte Carlo SPICE simulations, Monte Carlo Static Timing Analysis and Statistical Static Timing Analysis.

5.2.1 Test Circuits

Three test circuits were selected to investigate the effects of MOSFET variability on propagation delay, power and yield. The first circuit is the simple one bit full adder used within the previous section, while the second and third circuits are both ISCAS-85 benchmark circuits: c74283 a four bit fast adder, and c2688 a 16x16 bit multiplier. Small circuits allow for comprehensive SPICE simulations of the netlist to be performed. Large SPICE circuits may only be simulated in sections, preventing a comparison of critical paths and power analysis, this was found to be true of the multiplier. The size of the multiplier SPICE netlist exceeded the limits of the simulation tool when separate models were included for each individual transistor, and so only the transistors within the critical path were included. The total power consumption of the multiplier circuit could therefore not be recorded using Monte Carlo SPICE Simulations.

<table>
<thead>
<tr>
<th>Critical</th>
<th>Transistors</th>
<th>Cells</th>
<th>Path Depth</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adder</td>
<td>52</td>
<td>13</td>
<td>7</td>
</tr>
<tr>
<td>Fast Adder</td>
<td>246</td>
<td>64</td>
<td>10</td>
</tr>
<tr>
<td>Multiplier</td>
<td>12426</td>
<td>3390</td>
<td>111</td>
</tr>
</tbody>
</table>

Modern manufacturing technologies were not made available to the project, and so Standard Cell Libraries and transistor layouts were not accessible at a 35nm node. It
was therefore necessary to generate custom scaled 35nm cells, although parasitics, layout and routing information were not included. The parasitic information was omitted as analysis of the 130nm circuits with and without parasitic information revealed that SPICE and STA remained in agreement, and the generation of full custom 35nm cells was not within the scope of this project. Multiple width transistor model libraries were generated for RandomSpice by nanoCMOS project members, where Vth0, U0, rdsw and dsu were varied with Gaussian distributions, with the means and standard deviations of the parameters dependant upon the widths of the transistors.

The test circuits were synthesized using the 35nm technology, the transistor and cell counts are given in Table 5.6. A VCL was generated containing 500 variation instances of each of the following 35nm gates: inverter, buffer, NOR, NAND, OR, and AND. The size of the cell library was deliberately restricted for the purposes of this study. MCSTA runs of 10,000 randomized netlists were performed on each of the three test circuits.

5.2.1.1 4-bit Fast Adder

The size of the fast adder circuit meant that full and exhaustive SPICE simulations of every input transition were impractical for the 4-bit Adder. This demonstrates that it does not take much of an increase in size to remove the possibility of searching for critical paths using SPICE simulations.

An alternative method was to perform a single run of STA on the circuit, and extract the path with the greatest delays. The input stimuli required to excite this path were extracted and used in the Monte Carlo SPICE simulations of the complete circuit.

Comparisons between SPICE and MCSTA could therefore only be performed on a single specific path, rather than the maximum delay of the circuit.

5.2.1.2 16-bit Multiplier

The Multiplier circuit was found to be too large to simulate at the SPICE level. The time required to simulate multiple test vectors was no longer the problem, but instead the physical size of the circuit. The number of transistor models required during the randomisation stage meant that NGSPICE simply failed with a segmentation fault.

This was overcome by performing a single run of STA on the circuit and extracting a SPICE netlist of the path with the greatest delays. This SPICE netlist simply contained the cells along the switching path, and capacitive loads for each of these cells. This dramatically reduced the size of the netlist and allowed SPICE simulations to occur. Unfortunately the removal of the majority of the circuit meant that direct comparisons of power consumption between MCSTA and SPICE can not be made for the Multiplier.
5.2.2 Method

5.2.2.1 Monte Carlo SPICE Simulations

RandomSpice was used to replace each MOSFET model instance within a SPICE netlist with BSIM models randomly selected from the 35nm statistical library. Each individual transistor within a circuit was therefore modelled by a separate atomistic model. The RandomSpice tool was used to generate 10,000 randomized transistor level netlists of each circuit under test, and SPICE simulations were then performed on each circuit instance. The input vectors to the circuits included the stimulation of critical paths that were reported during STA. This allowed for a direct comparison of delays through fixed paths, as well as comparisons of the longest path delay through the circuit. The distributions of static and dynamic power consumption were also recorded.

5.2.2.2 Statistical Static Timing Analysis

Liberty NCX was used to characterize a selection of standard cells using variation aware characterisation. The same transistor model parameters and distributions used within the Monte Carlo SPICE simulations were used during this process. A statistical standard cell library (SSCL) was created using the Synopsys ‘Transistor Mismatch’ model, which allows the performance of the cell to be established for multiple \( \sigma \) levels of each transistor parameter. These results were combined using a method specific to the commercial tool, allowing the variation to be modelled by a single synthetic variation parameter. This process is based on evidence that if each variable has an independent normal distribution then the impact on a given timing parameter (delay, slew or constraints) resembles a normal distribution.

The SSCL was then used within a extension to the PrimeTime tool, referred to as PrimeTime VX, which provides a distribution of delays for paths within the test circuits in a single run. Statistical power analysis was not possible using this method.

The ‘Transistor Mismatch’ method involves altering the transistor model card to define variation parameters as: \( \text{param} = \mu + (\sigma \times \text{SigmaLevel}) \), where \( \mu \) is the mean of the parameter, \( \sigma \) is the standard deviation and \( \text{SigmaLevel} \) is the number of standard deviations to add to the mean. Liberty NCX then performs multiple characterisation simulations with \( \text{SigmaLevel} \) set to different levels for each parameter of each transistor. These sigma levels are determined by the tool and are combined using a ‘proprietary’ method modelling the variation with a single synthetic variation parameter.

Figure 5.14 is an example of a transistor sub-circuit that allows parameters within a transistor model to be specified during instantiation. Figure 5.15 is an example of how the transistor sub-circuit can be instantiated to allow ‘Transistor Mismatch’ characterisation to take place. The mean and standard deviation (SD) of the variation parameters
are defined separately for each transistor, and in this case is dependent upon the width of the transistor. A customisable parameter name is provided for $SigmaLevel$, and Liberty NCX is informed that this parameter name is a mismatch parameter. Liberty reads the netlist and replaces each occurrence of a mismatch parameter with a unique parameter name, so that the variation level of every transistor can be altered independently. Figure 5.16 shows the netlist modified by Liberty, the automatically generated parameter names are included in a separate file. Liberty then performs SPICE based characterisation simulations for multiple combinations of $SigmaLevels$.

```
*Definition of SSTA variation transistor Subcircuit
.SUBCKT PMOS_VARIATION D G S B
*Default Parameter Values
+ W=0 L=0
+ Vth0_Mean=0
+ Vth0_SD=0
+ Vth0_Sigma=0

*Transistor instance
M0 D G S B P L=L W=W

.MODEL P PMOS
*Variation Parameter Definition
+ vth0 = 'Vth0_Mean + ( Vth0_SD * Vth0_Sigma )'

.ENDS
```

Figure 5.14: Transistor model definition which allows Liberty NCX to perform ‘Transistor Mismatch’ cell characterisation

### 5.2.3 Monte Carlo Static Timing Analysis

As before, multiple SPICE netlists were generated for each of the selected standard cells using RandomSpice. The multiple randomized cell netlists were passed into the same commercial cell characterization tool as with SSTA, generating a Variation Cell Library (VCL) where every standard cell has multiple instances. 10,000 randomised netlists were generated for each test circuit and both STA and power analysis were performed on each netlist.

### 5.2.4 Results

Simulations without variation identified margins of error between SPICE and STA. The biases are shown in Tables 5.7 and 5.8 for timing and power measurements respectively, and were removed from the distributions generated by SSTA and MCSTA.
**Definition of Standard Cell Subcircuit**

```
.SUBCKT BUFX2M a vdd vss y
```

*Default values of ‘Transistor Mismatch’ parameters (required as 0 for simulations without variation)*

```
.PARAM PMM_VTH0 = 0
.PARAM NMM_VTH0 = 0
```

*Instances of variation transistor subcircuits (requires the addition of the x to the name of the transistor instance)*

```
* Mean and Standard deviation (SD) depend upon transistor width.
* SIGMA (sigma level) is a parameter that Liberty NCX alters to model increases in variation

*MN1 15 a vss vss nch L=35n W=110e-09 VTH0_MEAN=0.135424 VTH0_SD=0.014413
+VTH0_SIGMA=NMM_VTH0

*MP1 vdd a 15 vdd pch L=35n W=270e-09 VTH0_MEAN=-0.142181 VTH0_SD=0.012526
+VTH0_SIGMA=PMM_VTH0

*MN2 y 15 vss vss nch L=35n W=140e-09 VTH0_MEAN=0.135354 VTH0_SD=0.014063
+VTH0_SIGMA=NMM_VTH0

*MP2 vdd 15 y vdd pch L=35n W=320e-09 VTH0_MEAN=-0.142138 VTH0_SD=0.012381
+VTH0_SIGMA=PMM_VTH0
```

.ENDS

**Figure 5.15:** Example of a cell netlist that instantiates ‘Transistor Mismatch’ transistor models

```
* Modified by Liberty NCX vD-2009.12
* DMG/MOTIVATED BUF X2 35NM SKELETON NETLIST
.SUBCKT BUFX2M a vdd vss y

xMN1 15 a vss vss nch L=35n W=110e-09 VTH0_MEAN=0.135424 VTH0_SD=0.014413
VTH0_SIGMA=mmp0_0_0

xMP1 vdd a 15 vdd pch L=35n W=270e-09 VTH0_MEAN=-0.142181 VTH0_SD=0.012526
VTH0_SIGMA=mmp1_0_0

xMN2 y 15 vss vss nch L=35n W=140e-09 VTH0_MEAN=0.135354 VTH0_SD=0.014063
VTH0_SIGMA=mmp2_0_0

xMP2 vdd 15 y vdd pch L=35n W=320e-09 VTH0_MEAN=-0.142138 VTH0_SD=0.012381
VTH0_SIGMA=mmp3_0_0

.ENDS

* Generated by Liberty NCX vD-2009.12
* 4 devices, 4 mm parameters
.param mmp0_0_0 = 0
.param mmp1_0_0 = 0
.param mmp2_0_0 = 0
.param mmp3_0_0 = 0
```

**Figure 5.16:** Modifications to variation netlist made by Liberty NCX for statistical characterisation

**Table 5.7:** Comparison of Path Delays using SPICE and STA, without variability

<table>
<thead>
<tr>
<th></th>
<th>SPICE Path Delay (ns)</th>
<th>STA Path Delay (ns)</th>
<th>Relative Error(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adder</td>
<td>0.065</td>
<td>0.066</td>
<td>1.30</td>
</tr>
<tr>
<td>Fast Adder</td>
<td>0.086</td>
<td>0.085</td>
<td>1.09</td>
</tr>
<tr>
<td>Multiplier</td>
<td>1.090</td>
<td>1.065</td>
<td>2.28</td>
</tr>
</tbody>
</table>
Table 5.8: Comparison of Power Consumption using SPICE and Power Analysis, without variability

<table>
<thead>
<tr>
<th></th>
<th>SPICE Total Power (uW)</th>
<th>Power Analysis Total Power (uW)</th>
<th>Relative Error(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adder</td>
<td>4.04</td>
<td>3.60</td>
<td>10.90</td>
</tr>
<tr>
<td>Fast Adder</td>
<td>10.67</td>
<td>9.21</td>
<td>13.68</td>
</tr>
</tbody>
</table>

The results of the power and timing analyses of the Adder circuit are shown as a scatter plot in Figure 5.17, where the total power consumption of the circuit is plotted against the delay through the critical path. The distributions of delays generated by MCSTA and SPICE are very similar, with means and standard deviations within 0.09% and 2.4% of one another respectively. The distributions of power have similar means (within 0.7%), but MCSTA produces a noticeably narrower distribution than SPICE. This may be due to approximations in fanout capacitances made during STA, as small changes in load capacitance have a much greater impact on the integration calculations performed in SPICE power measurements than on the look-up-table interpolations used within STA. Further statistical analysis of the SPICE and MCSTA distributions for the Adder revealed that predictions of how 99.97% of the circuits will perform (+3σ) are within 2.5% and 0.2% for power and delay respectively. The statistical analyses of the performance of the three test circuits are given in Tables 5.9 and 5.10. The means, standard deviations and predictions for 0.3 and 99.7 percentiles have been calculated, the percentage errors are with respect to SPICE.

![Figure 5.17: Adder Power Vs Delay Scatter-plot](image)
Table 5.9: Path Delay Statistics

<table>
<thead>
<tr>
<th></th>
<th>SPICE</th>
<th>MCSTA</th>
<th>SSTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ns</td>
<td>ns</td>
<td>%Error</td>
</tr>
<tr>
<td>Adder</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>0.0706</td>
<td>0.0707</td>
<td>0.0880</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.0020</td>
<td>0.0020</td>
<td>2.3688</td>
</tr>
<tr>
<td>$-3\sigma$ (0.3%)</td>
<td>0.0653</td>
<td>0.0656</td>
<td>0.5043</td>
</tr>
<tr>
<td>$+3\sigma$ (99.7%)</td>
<td>0.0764</td>
<td>0.0765</td>
<td>0.1166</td>
</tr>
<tr>
<td>Fast Adder</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>0.0945</td>
<td>0.0942</td>
<td>0.3627</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.0024</td>
<td>0.0023</td>
<td>5.1052</td>
</tr>
<tr>
<td>$-3\sigma$ (0.3%)</td>
<td>0.0884</td>
<td>0.0882</td>
<td>0.2072</td>
</tr>
<tr>
<td>$+3\sigma$ (99.7%)</td>
<td>0.1015</td>
<td>0.1005</td>
<td>0.9786</td>
</tr>
<tr>
<td>Multiplier</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>1.1873</td>
<td>1.1816</td>
<td>0.4857</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.0102</td>
<td>0.0088</td>
<td>13.9391</td>
</tr>
<tr>
<td>$-3\sigma$ (0.3%)</td>
<td>1.1589</td>
<td>1.1581</td>
<td>0.0742</td>
</tr>
<tr>
<td>$+3\sigma$ (99.7%)</td>
<td>1.2165</td>
<td>1.2047</td>
<td>0.9693</td>
</tr>
</tbody>
</table>

Table 5.10: Power Consumption Statistics

<table>
<thead>
<tr>
<th></th>
<th>SPICE</th>
<th>MCSTA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Adder</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>3.93</td>
<td>3.96</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.06</td>
<td>0.03</td>
</tr>
<tr>
<td>$-3\sigma$ (0.3%)</td>
<td>3.80</td>
<td>3.88</td>
</tr>
<tr>
<td>$+3\sigma$ (99.7%)</td>
<td>4.15</td>
<td>4.05</td>
</tr>
<tr>
<td>Fast Adder</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean</td>
<td>10.30</td>
<td>10.40</td>
</tr>
<tr>
<td>Std. Dev.</td>
<td>0.20</td>
<td>0.10</td>
</tr>
<tr>
<td>$-3\sigma$ (0.3%)</td>
<td>10.00</td>
<td>10.30</td>
</tr>
<tr>
<td>$+3\sigma$ (99.7%)</td>
<td>10.60</td>
<td>10.60</td>
</tr>
</tbody>
</table>
A comparison of the cumulative distributions of power and delay reveals that MCSTA provides greater accuracy than SSTA when predicting the tails of the distributions. Figure 5.18 contains the CDFs of critical path delay, where the MCSTA forecast for 99.7% yield is within 1% of SPICE, compared to 6% when using SSTA. The delay distributions generated by SSTA for the Fast Adder are similar to those generated by both SPICE and MCSTA, as the combinations of cell delays along the critical path are successfully modelled by Gaussian distributions. The slightly different combination of these cells and the use of different input transitions within the 1-bit Adder shows that Gaussian distributions are not always suitable. The depth of logic within a critical path of the Multiplier indicates that the Central Limit Theorem can not always be relied upon, as the sum of the 111 cell delay distributions are not accurately modelled, with an error in standard deviation of over 280%. Statistical analyses of the cell delays within the VCL indicated that Generalized Extreme Value distributions offered a closer approximation to the SPICE level cell simulations, Figure 5.19.

![Figure 5.18: Multiplier Delay CDFs](image)

The cumulative distributions of power and delay generated by MCSTA can be combined to form 3D plots of power, performance and yield. This allows a designer to predict what proportion of the devices will perform within boundaries, and make trade offs between changes in power and performance. Figure 5.21 provides a 3D yield plot for the Adder circuit, where contours have been added at 20% increments in yield.

MCSTA produces results that are similar to large scale Monte Carlo SPICE simulations, at a fraction of the CPU time. Table 5.11 shows the dramatic increase in simulation time required for larger SPICE circuits, even when only the critical path is simulated for the Multiplier. The depth of logic within the Adder and the Fast Adder is similar,
**Figure 5.19:** Curve fitting a rising propagation delay through a NAND gate

**Figure 5.20:** Multiplier Delay Histograms
which is reflected in the similar computation times, while the Multiplier has a much
greater depth of logic but only required twice the runtime.

<table>
<thead>
<tr>
<th>CPU Time (h:m:s)</th>
<th>SPICE</th>
<th>MCSTA</th>
<th>SSTA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10000 Simulations</td>
<td>10000 Runs</td>
<td>1 Run</td>
</tr>
<tr>
<td>Adder</td>
<td>08:25:59</td>
<td>07:36:29</td>
<td>00:00:05</td>
</tr>
<tr>
<td>Fast Adder</td>
<td>59:29:39</td>
<td>07:46:48</td>
<td>00:00:02</td>
</tr>
<tr>
<td>Multiplier</td>
<td>315:40:38</td>
<td>14:16:48</td>
<td>00:00:06</td>
</tr>
</tbody>
</table>

While MCSTA is much faster than SPICE the computation times measured are imprac-
tical for commercial use. It was therefore necessary to reduce the number of random
netlists generated and analysed in a run of MCSTA. The process of performing MCSTA
was repeated for each circuit, using a range of sample sizes, and the accuracy of the
results (w.r.t. the large Monte Carlo SPICE simulations) was recorded for each sample
size. It was found that the levels of accuracy in the means, standard deviations and
99.7th percentile predictions converged at sample sizes of around 250 analyses. Figure
5.22 shows a plot of the percentage error in the standard deviation of delays for the
Adder circuit against the sample size used (on a logarithmic scale), the error bars indi-
cate the maximum and minimum errors found during multiple runs of MCSTA at each
sample size. Table 5.12 compares the accuracy of the mean, standard deviation and
99.7th percentile predictions for the delay distributions of the Multiplier circuit, as the
sample size is increased. A sample size of only 250 netlists still provides an accuracy
of within 0.5% and 1% for the mean and $+3\sigma$ predictions respectively, while the error in standard deviation remains at 14%. A MCSTA run of just over 20 minutes can be used to generate a distribution of circuit delays that accurately match the distribution generated by over 315 hours of SPICE simulations, while an MCSTA run of less than a minute still has a greater accuracy than SSTA. This computation time can be reduced even further due to the fact that each of the samples can be analysed independently and in parallel.

![Figure 5.22: Reduction in Absolute Relative Error in the Standard Deviation of Adder Path Delay Distributions, as the sample size used for MCSTA increases](image)

**Table 5.12:** Changes in the accuracy of predicted delay distributions as the MCSTA sample size is reduced for analysis of the 35nm Multiplier circuit. Predictions of $3\sigma$ delays remain to within 2% of SPICE even at a sample size of 10.

<table>
<thead>
<tr>
<th>Samples</th>
<th>CPU Time (s)</th>
<th>Mean Deviation</th>
<th>99.7(^{th}) Percentile</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>58</td>
<td>0.559</td>
<td>23.925</td>
</tr>
<tr>
<td>20</td>
<td>109</td>
<td>0.383</td>
<td>22.081</td>
</tr>
<tr>
<td>50</td>
<td>264</td>
<td>0.509</td>
<td>17.333</td>
</tr>
<tr>
<td>100</td>
<td>564</td>
<td>0.491</td>
<td>13.646</td>
</tr>
<tr>
<td>250</td>
<td>1279</td>
<td>0.493</td>
<td>13.954</td>
</tr>
<tr>
<td>500</td>
<td>2572</td>
<td>0.476</td>
<td>14.531</td>
</tr>
<tr>
<td>10000</td>
<td>52020</td>
<td>0.486</td>
<td>13.939</td>
</tr>
</tbody>
</table>
5.3 Summary

MCSTA was used to assess the performance of a 1-bit full adder implemented on a 130nm technology. This assessment was found to be an effective compromise between the accuracy of a Monte Carlo SPICE simulation and the practicality of abstracted Corner Analysis using STA. Predictions of the 99.7th percentile of circuit performance (3-sigma) were within 1.2% of SPICE for power consumption and 3% of SPICE for maximum path delays, at the highest level of injected variability. Corner analysis was found to have significant errors of 782% and 548% for power and delay at the same level of variability.

The effectiveness of MCSTA was further demonstrated on a series of test circuits implemented on a 35nm process, where greater accuracy (with respect to SPICE) was found when compared to SSTA. MCSTA also has the significant benefit of allowing the user to generate distributions of static and dynamic power consumption.

The performance of some of the test circuits was illustrated by three dimensional plots of Power, Performance and Yield, providing an indication of the percentage of circuits that would meet a specified timing or power constraint. Such analysis allows designers to assess the impact of manufacturing process variations on a circuit, and allows for budgeting and marketing decisions to be made based on the number of fabricated chips that can be sold at different performance points. MCSTA can also be used to assess the probability of individual timing paths becoming critical, and determine if any areas of a design act as a timing bottleneck.

While the accuracy of MCSTA has been demonstrated, there is a significant increase in computation time when compared to a single run of STA or SSTA. An increase in computation time may be acceptable to some designers for the significant increase in accuracy at the sign-off stage of the design process, but MCSTA requires further optimisations before it can be used as an early predictor of process variations through the entire design cycle. Such optimisations are discussed in the next chapter.
Chapter 6

Implementation - Practical MCSTA

The use of Monte Carlo Static Timing Analysis to predict the impact of statistical process variations on the performance of circuits has been discussed and tested within the previous chapters of this document, but the computation times associated with large numbers of cell characterisation and STA runs do not represent a favourable methodology to the IC design industry. The run time of a full Monte Carlo Static Timing Analysis run may be acceptable at the final sign-off stage of the design process, before a circuit is sent for manufacture, but if timing problems are exposed within the design at such a late stage then the cost of re-design may be extremely high. It is therefore important for circuit designers to be able to perform faster forms of MCSTA at earlier stages of the design flow, highlighting any problems as soon as possible when the cost of making design alterations may be trivial. This chapter discusses some more practical methods involving MCSTA that can be used to reduce the need for large analysis runs, saving time without significantly decreasing the accuracy of the analysis.

6.1 Metric of Variability

This document has highlighted some of the inaccuracies and shortfalls in current timing validation tools and methods when used to model circuit behaviour at the nanometre level, which can prevent circuit designers from altering their designs to accommodate random physical intra-die variations. The derivation of a simple but effective metric for the prediction of path delay variability in CMOS logic circuits is presented within this section. This metric allows for a rapid comparison of the variability within logic paths, without the need for extensive SSTA or Monte Carlo simulations. The variability of the delay through a given logic path is defined as $\frac{\sigma_{\text{path}}}{\mu_{\text{path}}}$ where $\mu_{\text{path}}$ and $\sigma_{\text{path}}$ are the mean and standard deviation of the path delay respectively.
The derived metric provides a method of evaluating path delay variability for any synthesis methodology (i.e., based on wire load models or physically based), and strongly simplifies the path timing analysis process for modelling the effects of intra-die variations on delays at an early stage of the design process. The information required for predicting the variability of path delays is minimal, as it depends on very basic parameters of the cells used during circuit synthesis. These parameters can be obtained directly from the knowledge of the chosen cell library, such as the driving strength of the cells and the number of stacked transistors within the cell structure.

A key strength of the proposed metric is that statistical evaluation of the path delay can be performed using standard, deterministic, CAD tools with no requirement for a statistical timing analyser. This dramatically reduces the amount of computations time required for statistical analysis during the early stages of digital circuit design, especially during the comparison of different design options and circuit architectures. Using the proposed metric full statistical static timing analysis can be postponed to validate and sign-off the final design, having already taken design decisions required to reduce the impact of random physical intra-die variations.

This section introduces the metric, describing the assumptions made during the derivation process, and provides an assessment of the effectiveness and limitations of the metric.

### 6.1.1 Description of the proposed metric

Work within [4] highlighted that delay variability caused by random process variations within a given logic gate depends mainly upon the following five factors:

1. Gate topology (the logic function of the gate and the number of stacked transistors used)
2. Transistor size (length and width)
3. Input rise/fall time
4. Output parasitic capacitance of the gate
5. Load capacitance (both from interconnecting wires and the input capacitances of connecting gates)

The following observations are made within [4] about the above sources of variation in order to simplify the process of modelling variations that have a different impact on different transistors within the same die.
1. Topology affects the current $I_{\text{gate}}$ delivered by the gate to the output capacitance. In [4], it was demonstrated that the variations in the on current delivered by a generic logic gate $I_{\text{gate}}$ (and hence the resulting delay variations) are proportional to the reciprocal of the square root of the number of stacked transistors $n_{\text{stack}}$:

$$\frac{\sigma_{\text{delay}}}{\mu_{\text{delay}}} \propto \frac{1}{\sqrt{n_{\text{stack}}}}$$  \hspace{1cm} (6.1)

2. For a given topology, in [4] it was demonstrated that the variability of $I_{\text{gate}}$ is proportional to the reciprocal of the square root of the transistor size, and therefore the driving strength of the cell.

$$\frac{\sigma_{\text{delay}}}{\mu_{\text{delay}}} \propto \frac{1}{\sqrt{\text{strength}}}$$  \hspace{1cm} (6.2)

3. It can be shown that the effect of the input rise/fall time on the delay variability tends to be negligible, compared to the above discussed effects [4]

4. It can be shown that the effect of the output parasitic capacitance on the delay variability is also negligible, compared with the variation of the on current (see point 1) [4]

5. Effect of load capacitance (due to input capacitance of subsequent gates and the in-between wires). The input capacitance is rather insensitive to intra-die process variations [4]. Wire capacitance is well-known to be rather insensitive to random intra-die variations (highly correlated) [74]

From observations 1, 4-5 and equations (6.1)-(6.2), the delay variability of a logic gate due to random intra-die variations is simply estimated within [4] as:

$$\frac{\sigma_{\text{delay}}}{\mu_{\text{delay}}} = \frac{k}{\sqrt{n_{\text{stack}}} \sqrt{\text{strength}}}$$  \hspace{1cm} (6.3)

where $k$ is a technology-dependent constant that is easily evaluated from very few preliminary simulations. From (6.3), the delay variability of a logic gate for a given input transition is characterised very easily from the knowledge of its topology and drive strength. The drive strength of standard cell refers to maximum output load that the cell can drive and is usually identified within the cell version, such as AND_X1 for a single or normal drive strength AND_X2 for twice the drive strength. This information is easy to obtain and is available to the designer as soon as a standard cell library is adopted for the synthesis process.

The above considerations only hold for intra-die process variations, and do not hold for inter-die variations. It is however noted within the literature that inter-die variations are well known to be easier to model and compensate for with feedback adaptive schemes [4] [74].
6.1.1.1 Early estimation of path delay variability

The work within [4] carries on to explain that under random intra-die variations, the delays of all gates belonging to a given path are uncorrelated random variables with variability \( \frac{\sigma_i}{\mu_i} \) given by (6.3); \( i = 1 \ldots N \) where \( N \) is the number of gates belonging to the considered path [74]. Which when assuming that the path delay variability follows a Gaussian distribution results in

\[
\frac{\sigma_{\text{path}}}{\mu_{\text{path}}} = \sqrt{\sum_{i=1}^{N} \left( \frac{\sigma_i}{\mu_i} \right)^2 \left( \frac{\mu_i}{\mu_{\text{path}}} \right)}
\]

which can easily be evaluated within the framework of standard CAD tools without the need for SSTA algorithms. The constant, \( k \), is only required to be extracted once for a given technology, \( n_{\text{stack}} \) and \( \text{strength} \) are defined by the structure of the cells within the path, and the weight \( \frac{\mu_i}{\mu_{\text{path}}} \) represents the fraction of the path delay spent in the \( i \)-th gate which can be deterministically evaluated by standard timing analysers.

The metric of (6.4) is then further simplified within [4] by introducing some approximations, in particular, assuming that the gate delays along the path are comparable,

\[
\frac{\sigma_{\text{path}}}{\mu_{\text{path}}} = \sqrt{k} \sqrt{\frac{1}{n_{\text{stack},i} \cdot \text{strength}_i} \left( \frac{\mu_i}{\mu_{\text{path}}} \right)^2}
\]

(6.5)

where \( n_{\text{stack}} \) is the average number of stacked transistors along the path, \( \text{strength} \) is the average cell strength along the path and \( N \) represents the path logic depth. Many other metrics can be easily derived by considering that the gate delays are not equal, where the above parameters would instead be evaluated as a weighted average.

6.1.2 Evaluating the Design Metrics

The work described within Chapters 3, 4 and 5 was used to perform an investigation on the effectiveness of the metric described within (6.5) using RandomSpice models for a small subset of a 130nm technology library, the method and results of which are presented here.
6.1.2.1 Simulation Framework

Simple combinational logic gates such as inverters, NANDs and NORs provide simple and obvious test cases for the described variability metric, as each gate has either a pull-up or pull-down network of transistors connected solely in series. An inverter has an $n_{\text{stack}}$ value of 1 irrespective of the input transition, while NOR (NAND) gates have an $n_{\text{stack}}$ value equal to the number of inputs to the gate when the input is falling (rising) and an $n_{\text{stack}}$ value of 1 when the input is rising (falling), figures 6.1 and 6.2.

10,000 RandomSpice netlists were generated for each of four paths constructed from combinations of these simple gates shown in Figure 6.3. The configuration of the test-bench used in each case is given in Figure 6.4.

The numerical value for the technology-dependent constant $k$ was obtained from the path variability results of the Inverter chain, as the inverter represents the simplest CMOS logic structure. This was obtained by first generating a prediction of the path delay variability using (6.5) with $k$ set to 1. Results in Figure 6.5 confirm that a linear relationship exists between the predicted and simulated values, adding credibility to the
proposed metric (i.e., the variability is confirmed to be proportional to the right-hand side of equation 6.5 through a constant \( k \)). Then, the numerical value of \( k \) was found from the slope of the plot of the simulated and predicted variability (obtained through linear regression). Figure 6.5 shows a plot of the simulated variability \( \frac{\sigma_{\text{path}}}{\mu_{\text{path}}} \) against the prediction, where \( \sqrt{k} \) was found to be 0.0567. The calculated value of \( k \) was then used to provide predictions of the path variability for each of the remaining test paths. Figure 6.6 is a plot of the simulated path variability against the predicted path variability, which lie within 25% of each other (this accuracy will be shown to be significantly better under appropriate improvements).
6.1.2.2 Analysis of Results

The predicted values of path delay variability for a rising input transition in test path Figure 6.3(c), and for a falling input transition for Figures 6.3(b) and 6.3(d), are on average within 5% of the simulated values. For these input transitions the pull-up and pull-down networks within the cells consist of single switching transistor (inverters), or

---

**Figure 6.5:** A plot of simulated path delay variability ($\sigma_{path}$) against predicted path delay variability (Equation 6.5) for test path Figure 6.3(a). The value of $k$ has been initially set to 1.

**Figure 6.6:** A plot of simulated path delay variability ($\sigma_{path}$) against predicted path delay variability (Equation 6.5) for the test paths in Figure 3.1. $\sqrt{k}$ is set to the calculated value of 0.0567.
a switching transistor arranged in parallel with an inactive transistor (NAND and NOR gates). This suggests that the addition of inactive transistors in parallel within the switching transistor does not greatly increase the variability of the gate, as the proposed metric does not include inactive parallel transistors within the value of $n_{\text{stack}}$ and the generated predictions remain accurate.

In contrast, predictions for the opposite transitions through these paths are on average 20% lower than the simulated values. In these cases the pull-up and pull-down networks within the cells consist of single switching transistors (inverters), or a switching transistor arranged in series with an inactive transistor (NAND and NOR gates). The addition of transistors in series with the switching transistor is included within $n_{\text{stack}}$, but does not reduce the variability as much as predicted. In other words, the approximation made within (6.1) is not very accurate and can be improved by a modification in the calculation of $n_{\text{stack}}$. More specifically, in a generic logic gate with a given $n_{\text{stack}}$, an equivalent number of series transistors $n_{\text{stack}}$ can be derived as the value of $n_{\text{stack}}$ that makes the estimated variability (6.4) equal to the exact variability (which is evaluated with a simple simulation of a path consisting of equal cascaded logic gates). This leads to the following expression for the equivalent number of series transistors in (6.6).

$$n_{\text{stack}} = \left( \frac{\mu_{\text{path}} \sqrt{k}}{\sigma_{\text{path}} \sqrt{N \sqrt{\text{strength}}}} \right)^2$$  \hspace{1cm} (6.6)

Experimental values of $n_{\text{stack}}$ indicated that a single transistor within a stack could be counted as 1 within the metric, for both PMOS and NMOS transistors, but the equivalent $n_{\text{stack}}$ of two series transistors is equal to 1.2 (instead of 2). The corrected values were used to recalculate the predicted variability of the paths with serially stacked transistors, providing a much improved prediction. The results from this corrected value of $n_{\text{stack}}$ are given in Figure 6.7.

The accuracy of the metric increases with a greater depth of logic, and the differences between the predicted values and simulated values have a maximum error of 10% and an average error of 3% for $N > 2$. The increase in accuracy with logic depth may be due to the assumptions within the metric of the Gaussian nature of the delays, as the RandomSpice models are not based on Gaussian distributions of transistor parameters. The accuracy improves because the delays through the path become increasingly Gaussian in nature as the logic depth increases, as predicted by the Central Limit Theorem.

The corrected value of $n_{\text{stack}}$ was verified by repeating the analysis of test path Figure 6.3(b) with three, four and five input NAND gates replacing the previously analysed two input NAND gates. An increase of 0.2 was applied to $n_{\text{stack}}$ for each additional input, and the predicted variability was found to remain within 8%, again with an average error of 3%. Figure 6.8 provides a plot of the path delay variability predictions for multiple
Figure 6.7: A plot of simulated path delay variability ($\sigma_{\text{path}}$) against predicted path delay variability (Equation 6.5) for the test paths in Figures 3.1. The values of $n_{\text{stack}}$ have been changed to 1.2 for cells with two transistors in series.

Figure 6.8: A plot of simulated path delay variability ($\sigma_{\text{path}}$) against predicted path delay variability (Equation 6.5) for the test path in Figure 6.3(b), where the number of inputs to the NAND gates are varied.

input NAND gates, where $n_{\text{stack}}$ was set to 1.2, 1.4, 1.6 and 1.8 for the two, three, four and five input gates respectively.

At this level of accuracy the proposed metric provides a simple and efficient method of predicting the variability of circuits synthesised from simple CMOS logic, especially if the circuit is constructed from simple NAND or NOR based logic.
6.1.3 Summary

A simple metric has been developed which accurately predicts the variability of delays through circuits comprising of simple CMOS logic. Equation 6.5 allows for the variability of different paths to be compared without the need for extensive Monte Carlo or SSTA analysis, allowing circuit variability to be assessed at very early stages within the design process with minimal computational effort. The predictions are based on Gaussian distributions of gate delays, and have been used to accurately predict the variability of paths consisting of non-Gaussian gate delays for logic depths greater than two cells.

The proposed metric provides an accurate and rapid statistical evaluation (always within 10% and usually within 3%) of designs at the first stages of synthesis for circuits consisting of simple CMOS gates. This allows circuit designers to avoid performing large Monte Carlo simulations during design iterations, which thereby dramatically reduces the computational effort required for early circuit analysis. This means that Monte Carlo simulations can be reserved for the sign-off stage of the design cycle if and when greater accuracy is required.

The simplicity of the metric means that it should be possible to include it within the design goals of synthesis tools. Synthesis tools are usually used to focus on the timing of a circuit, with varying degrees of effort spent on optimising the power consumption and logic area of the synthesised design. This metric may be included within these goals, so that synthesis tools automatically aim for designs with low levels of variation on critical timing paths, minimising the range of possible delays and increasing the predictability of the behaviour of the circuit. This would help designers to increase the yield of successfully manufactured designs, and assist in the prediction of the range of performance of manufactured ICs, for the benefit of costing and marketing the finalised product.

Future work should expand on investigation and determine the limitations of the metric when predicting the variability of circuits containing complex cell structures, while verifying the effectiveness of the metric for future technology nodes.

6.2 Statistical Sampling

It has been shown during the previous chapters that Monte Carlo Static Timing Analysis (MCSTA) is a faster method of statistical circuit analysis than Monte Carlo SPICE simulations, and provides acceptable levels of accuracy in the generation of distributions of circuit performance. The results of MCSTA are a large improvement over the inherent pessimism of Corner based STA and SSTA, but are much slower to produce due to the large amounts of analyses required. The Variability Metric described in the previous section provides a rapid prediction of the variability of delays through a timing path.
with accuracy levels suitable for use within the early stages of synthesis, and full MCSTA may be reserved for the sign-off of the final design, but there is a significant gap between initial synthesis and design sign-off where timing analysis still plays a key role. These stages are often referred to as the design implementation stages, where the standard cells of the synthesised circuit are placed within a design footprint or floorplan and the connections between the standard cells are routed. The placement and routing decisions made by designers and automated design tools are usually based on the timing of logic paths, as large gaps between connected cells and excessively long routing paths can add significant delays and reduce circuit performance. The results of STA can be used to order the priority of the the placement and routing tools, focusing larger amounts of effort on the critical timing paths, perhaps allowing longer routing paths to be allocated to less significant timing paths. These results are updated after changes are made to the placement or routing of the design, and it is therefore important to be able to predict the effects of statistical process variations during these stages of the design flow, as the layout of the circuit may need to be altered to incorporate variations in path delays.

A compromise is required between the variability metric that may be used during initial circuit synthesis and full sign-off MCSTA, as accuracy is important, but the runtime of full MCSTA is excessive, especially if an analysis is to be performed whenever the physical position of a standard cell is changed within a design, or routing between cells is adjusted. It is therefore important to reduce the number of analyses performed during MCSTA in order for this approach to be a practical alternative to existing methods.

A full MCSTA run generates distributions of delay and power consumption through paths within a circuit, providing designers with a prediction of the whole range of possible behaviours. In practice designers are interested with the extremes or corners of these behaviours, rather than the entire distribution, or even the mean of the distribution, because design decisions are based on making the design function correctly under as many conditions as possible. This suggests that the vast majority of the distributions of delays and power are not required, especially during the placement and routing stages of the design flow. The nature of a Monte Carlo analysis means that the majority of the samples in an analysis will be near the mean of the distribution, rather than the extremes, so the majority of the computation time during MCSTA would be wasted. A suitable compromise would require the ability to select samples that represented the extremes of a distribution, so that fewer samples and less computation time would be required.

This section provides a description and evaluation of a method of statistical sampling that can be used within MCSTA. This method provides a prediction of the extremes of distributions generated by large MCSTA runs, with a fraction of the CPU time.
6.2.1 Comparing Standard Cell Variation Instances

A Variation Cell Library (VCL) contains look up tables (LUTs) of power consumption, output transition times and cell delays for every variation instance of a standard cell. These look up tables can be compared and used to sort the variation instances of a standard cell into any desired order, for any specified measurement. For example, it is possible to compare the LUTs of every variation instance of an Inverter and determine which model contains the longest output delay for a specific input transition and output load. Another example would be to compare the power consumption LUTs and determine which NAND gate uses, on average, the most energy. The comparison can be performed using simple scripting languages as the LUTs in a .lib file are written in ASCII.

Using this information it is possible to find the extreme limits of the performance of a circuit under variation, or at least the limits at which MCSTA will predict the circuit performance. The slowest variation instances could be found for each standard cell, and STA could be performed on a circuit referencing only these slowest instances. Similarly the variation instances of each cell in a circuit could be selected as those with the highest values for leakage energy, for a worst case evaluation of leakage power consumption. The extreme of the distributions generated by MCSTA could be generated by deliberately targeting the worst or best case of each variation instance, saving the need to analyse thousands of randomised samples. However this would simply be an alternate form of corner analysis, where the worst possible variation of every single standard cell would be modelled simultaneously and would produce an overly pessimistic estimation of circuit performance.

The ability to compare the values of timing and power for each variation instance of a cell in a library allows for the examination of the delay and power distributions of each standard cell within the library. The distribution of possible values for a given input to output delay of a cell can be seen by the designer before any statistical analysis of the circuit is performed and the position of each variation instance within the overall distribution can be found.

The research within this thesis has found that distributions of cell delays and power consumption are not accurately modelled by Gaussian models at the extremes or corners of the distributions, but that the means of the distributions remain relatively accurate. One method of reducing the number of samples required during MCSTA is to find where a random sample of variation instances would fall within a Gaussian model of the circuit, and if these are found to be close to the mean of the distribution then the circuit analysis is not performed, whereas if the instances are found to be at the extremes of the distribution then circuit analysis is performed. If small numbers of samples from the center of the distribution are passed to an STA tool, while large numbers of corner
samples are sent to an STA tool, then the overall distribution of circuit performance can be obtained at a fraction of the computation time.

This method represents a compromise between the use of pure statistical models such as SSTA, and the need for large sample sizes in full MCSTA, as a statistical model is used to predict the significance of the randomly generated samples used in MCSTA. The process of generating a randomised netlist, where each cell instance refers to a variation instance within a VCL, and comparing the netlist against a statistical distribution is much faster than performing full timing and power analysis on the randomised netlist. It is therefore possible to prevent the randomised netlist from being analysed (or even written to disk) if it is found to be within the center of the statistical model, and to allow the netlist to be analysed if it is within a chosen range at the edges of the distribution.

6.2.2 Method

The initial step is to examine the contents of a chosen VCL, reading the contents of LUTs for the variation instances of each standard cell. The mean and standard deviation of each point of each LUT is calculated for each standard cell, so that a separate Gaussian distribution can be used to represent the variation of time and power through each characterised timing and power arc through the cell. Examples of LUTs for the rising propagation delay through three variation instances of an inverter are given in Table 6.1, from which the means and standard deviations are found and stored in a Gaussian model such as that shown in Table 6.2. Elements from the same position within the LUTs are summed, and the mean value and standard deviations are stored within the equivalent positions of the mean and standard deviation LUTs of the Gaussian model, as highlighted in red and blue respectively within the examples. The process of fitting Gaussian distributions to each element within the variation instance LUTs is similar to the analysis described earlier within Section 3.5.1.

The normal probability plot within Figure 6.9 shows a comparison between the distribution of rising propagation delays through 500 variation instances of a 35nm Inverter for a single point within the LUT, against the Gaussian model derived from the mean and standard deviation of the distribution. The distribution matches the model closely between probabilities of 0.1 and 0.9, but is less reliable at the extremes of the distribution. This implies that the Gaussian model could be used to model the average behaviour of the inverter, but not the corner cases that circuit designers may be interested in. The exact fit to a Gaussian model differs between standard cells, between timing paths through a standard cell, and between positions within an LUT, but the overall pattern remains, that the centre of the distribution fits while the extremes are modelled poorly.

The generated Gaussian models can be used to predict the behaviour of the whole circuit, or subsections of the circuit. The distributions of leakage power for each cell
### Table 6.1: Example of Rising Cell Delay through three variation instances of a standard inverter cell

<table>
<thead>
<tr>
<th>Output Load (fF)</th>
<th>INV_1 Rise Delay (ps)</th>
<th>Input Slew (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>0.25</td>
<td>3.34</td>
<td>4.36</td>
</tr>
<tr>
<td>0.50</td>
<td>3.80</td>
<td>4.79</td>
</tr>
<tr>
<td>2.50</td>
<td>6.10</td>
<td>7.66</td>
</tr>
<tr>
<td>5.00</td>
<td>9.55</td>
<td>11.72</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output Load (fF)</th>
<th>INV_2 Rise Delay (ps)</th>
<th>Input Slew (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>0.25</td>
<td>4.72</td>
<td>6.16</td>
</tr>
<tr>
<td>0.50</td>
<td>5.37</td>
<td>6.76</td>
</tr>
<tr>
<td>2.50</td>
<td>8.61</td>
<td>10.82</td>
</tr>
<tr>
<td>5.00</td>
<td>13.48</td>
<td>16.55</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output Load (fF)</th>
<th>INV_3 Rise Delay (ps)</th>
<th>Input Slew (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>0.25</td>
<td>3.21</td>
<td>4.19</td>
</tr>
<tr>
<td>0.50</td>
<td>3.65</td>
<td>4.60</td>
</tr>
<tr>
<td>2.50</td>
<td>5.86</td>
<td>7.35</td>
</tr>
<tr>
<td>5.00</td>
<td>9.17</td>
<td>11.25</td>
</tr>
</tbody>
</table>

### Table 6.2: Translation of the variation instances into look up tables of mean and standard deviation for the Gaussian models of the standard inverter cell

<table>
<thead>
<tr>
<th>Gaussian Model Mean INV Rise Delay (ps)</th>
<th>Input Slew (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Output Load (fF)</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>2.50</td>
</tr>
<tr>
<td></td>
<td>5.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gaussian Model Std. Dev. INV Rise Delay (ps)</th>
<th>Input Slew (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Output Load (fF)</td>
<td>0.25</td>
</tr>
<tr>
<td></td>
<td>0.50</td>
</tr>
<tr>
<td></td>
<td>2.50</td>
</tr>
<tr>
<td></td>
<td>5.00</td>
</tr>
</tbody>
</table>
that is instantiated within a netlist can be combined to produce a prediction of the
distribution of leakage power of the whole netlist. Alternatively the distributions of
delays through standard cells within a critical path could be combined to produce a
prediction of the distribution of delays through the critical path.

Analysis of the nominal version of a netlist (where no statistical process variations are
included) is used to provide the direction of the logic transitions though a critical path,
including input transition times and capacitative loads. These transition times and
capacitative loads can then be used to determine which locations within each of the
standard cell LUTs are used to obtain the correct Gaussian models to predict the vari-
ation of delays through each cell. The sum of these individual Gaussian models is then
used to predict the distribution of delays through the entire path. The mean of the
Gaussian distribution at any depth within the logic path is the sum of the means of each
previous logic gate, while the variance of the Gaussian distribution is the sum of the
variances of each of the previous logic gates and the standard deviation is the square-root
of the variance.

A critical path through the one bit Adder that was analysed within Section 5.1 provides
a good example of how this process can be carried out. Using the component labels
as defined within Figure 5.1 and the dominant critical path (U11 → U10 → U8 → U4 → U3 → U1 → U12), the longest delays through the one bit adder occur via a path
of seven logic gates. Figure 6.10 contains plots of the probability density functions for
the cumulative Gaussian delay models through each of the seven logic gates within the
critical path. The PDF to the farthest right (U12 BUFX2M) represents the distribution

![Figure 6.9: Plot comparing the distribution of propagation delays for the rising trans-
station of a 35nm Inverter (blue) with a fitted Gaussian distribution (red)]
of delays at the output of the final gate within the critical path, and is therefore the Gaussian model for this transition through the entire critical path.

This is a very simple and fast method of predicting the distribution of delays through a logic path, and this method resembles some of the earliest forms of SSTA. It should be noted that although this crude method of statistical modelling produces inaccurate estimations of the extremes of circuit behaviour, as was found with more advanced SSTA techniques within Chapter 5, the purpose of this approach is not to accurately model the behaviour of the circuit by using statistical models, but is instead to use statistical models to reduce the number of samples required by an accurate MCSTA run.

The goal of reducing the runtime of an accurate MCSTA run can be achieved by combining the above Gaussian models with the MCSTA method described within Chapter 4. References to standard cells within a netlist are replaced by randomly selected instances from a variation cell library as normal, and then the delays or power information within the randomly selected variation instances are compared with the Gaussian models that have been generated for the circuit. If the combined behaviours of the randomly selected variation instances lie within the tails of the Gaussian model then the circuit is analysed, but if the predicted behaviour is within the center of the distribution, as the vast majority will be, then the effort of performing a full circuit analysis can be avoided as the results are not expected to be of interest to the designer. The predicted behaviour of the current randomised netlist can be either the total power or total delays of the variation instances of interest to the designer, this predicted behaviour is compared against the Gaussian model by determining the probability of the predicted behaviour being generated by the Gaussian model itself, finding where the predicted behaviour lies within
the CDF of the Gaussian model. If the probability of obtaining the behaviour using the Gaussian model is below a threshold specified by the designer then it may be discarded, otherwise it may be retained for further, more accurate, analysis. A simplified summary of this process is provided in Figure 6.11.

![Figure 6.11: Performing Monte Carlo Static Timing Analysis with Statistical Sampling](image)

For power analysis the designer may select to sum the leakage power of every variation instance within the randomised netlist and compare the total leakage power with the Gaussian model for total leakage power. If the combined leakage power of the variation instances lies within the center of the Gaussian model then netlist analysis may be skipped, but if the combined leakage power is within the tails of the Gaussian model then the netlist can be passed to full power analysis and STA tools.

For analysing critical timing paths the process can be narrowed down to selected timing arcs through specific cells of interest. This may represent a particular set of paths that are known to be problematic, or a set of paths that have been altered by a change in the placement or routing of the design where the designer (or automated design tool) wishes to ensure that the alterations have not caused the paths to become critical paths. The individual cell delays through the path of interest are summed together using the appropriate LUTs of the variation instances and the total is compared against the Gaussian model for the same path. If the predicted path delay is within the center of the
Chapter 6 Implementation - Practical MCSTA

Gaussian model then the randomised netlist may be discarded and a new one generated, if the predicted path delay is within the extremes of the Gaussian model then the netlist can be passed on for full STA.

The Gaussian model that was generated for the 1bit Adder within Figure 6.10 provides a good example of how this form of statistical sampling can save computation time for MCSTA without large reductions in accuracy. The mean and standard deviation of each delay through each cell is obtained from the distributions of delays within a VCL, the appropriate Gaussian models are then combined to estimate the mean and standard deviation of an entire critical path. An example of this is given in Table 6.3, where a Gaussian model is selected for each cell within the path based upon the pins used, the direction of the input transition, and the capacitative loading. The Gaussian models of the standard cells were derived from a 35nm VCL containing 500 variation cell instances of each standard cell.

Table 6.3: The mean and standard deviation of the delay through a critical path can be obtained from the mean and standard deviations of the delays of the cells within the path. The means (\( \mu \)) and standard deviations (\( \sigma \)) of cell delays are calculated from the multiple variation instances of each cell within a variation cell library.

<table>
<thead>
<tr>
<th>Component</th>
<th>Cell</th>
<th>( \mu ) Delay (ps)</th>
<th>( \sigma ) Delay (ps)</th>
<th>Cumulative ( \mu ) (ps)</th>
<th>Cum. ( \sigma ) (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U11</td>
<td>INVX2M</td>
<td>5.36</td>
<td>0.53</td>
<td>5.36</td>
<td>0.53</td>
</tr>
<tr>
<td>U10</td>
<td>OR2X2M</td>
<td>15.05</td>
<td>1.02</td>
<td>20.41</td>
<td>1.15</td>
</tr>
<tr>
<td>U8</td>
<td>NAND2X2M</td>
<td>9.15</td>
<td>0.66</td>
<td>29.56</td>
<td>1.32</td>
</tr>
<tr>
<td>U4</td>
<td>INVX2M</td>
<td>6.85</td>
<td>0.59</td>
<td>36.41</td>
<td>1.45</td>
</tr>
<tr>
<td>U3</td>
<td>OR2X2M</td>
<td>15.05</td>
<td>1.02</td>
<td>51.45</td>
<td>1.77</td>
</tr>
<tr>
<td>U1</td>
<td>NAND2X2M</td>
<td>6.93</td>
<td>0.43</td>
<td>58.39</td>
<td>1.82</td>
</tr>
<tr>
<td>U12</td>
<td>BUFX2M</td>
<td>13.83</td>
<td>0.93</td>
<td>72.22</td>
<td>2.05</td>
</tr>
<tr>
<td>Path</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The process of MCSTA is then modified to allow estimations of critical path delays to be made for each randomised netlist that is generated. Table 6.4 contains examples from two randomised netlists where the LUTs of each randomly selected variation cell have been read for the same output load and input transition as was used to generate the Gaussian model of the critical path delay. The individual cell delays are combined to form an estimation of the critical path delay for each randomised netlist, in this example the second netlist estimated to be nearly 4ps slower than the first netlist. This is an extremely simplistic way of obtaining a delay for the path, as it does not take the output transition times of the variation cells into account, nor does it use constraints, interconnect parasitics or any other detailed analysis that may be performed by an STA tool. The point of this analysis is not to provide the designer with an accurate calculation of the path delay, but instead to see where within the Gaussian model the currently selected variation instances are expected to perform. This is achieved by comparing the
critical path delay for each randomised netlist against the Gaussian model of the critical path delay. In this case the Gaussian model has a mean of 72.22ps and a standard deviation of 2.05ps, and the cumulative distribution function for these values returns a probability of 0.5661 and 0.9792 for obtaining the predicted path delays of the first and second randomised netlists respectively, Table 6.5. This means that there is 56.61% chance of producing a faster netlist than netlist 1 by repeating the randomisation process, and a 97.92% chance of producing a faster netlist than netlist 2. Conversely there is only a 2.08% chance of producing a slower randomised netlist than netlist 2, and so performing a complete STA analysis on netlist 2 will provide a much more accurate delay calculation and this may represent the slowest 2% of circuit performance under statistical process variations. These percentages are represented by the area under the curve for component U12 in Figure 6.12, where the positions of the critical path delay estimates within the Gaussian models are labelled for the two randomised netlists for each cell within the path. The area to the left of an estimate represents the probability of obtaining a faster circuit, while the area to the right of the estimations represents the probability of obtaining a slower circuit. If this model holds true and the designer is only interested in the behaviour of the slowest 2% of circuits then the results of STA on randomised netlist 2 may be used instead of performing full MCSTA with hundreds of samples.

Table 6.4: Estimations of delays through the critical paths of two randomised instances of a 35nm 1-bit Adder Circuit

<table>
<thead>
<tr>
<th>Component</th>
<th>Variation Cell Instance</th>
<th>Cell Delay (ps)</th>
<th>Total Delay (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U11</td>
<td>INVX2M_331</td>
<td>5.33</td>
<td>5.33</td>
</tr>
<tr>
<td>U10</td>
<td>OR2X2M_53</td>
<td>15.61</td>
<td>20.94</td>
</tr>
<tr>
<td>U8</td>
<td>NAND2X2M_200</td>
<td>9.03</td>
<td>29.97</td>
</tr>
<tr>
<td>U4</td>
<td>INVX2M_133</td>
<td>7.04</td>
<td>37.00</td>
</tr>
<tr>
<td>U3</td>
<td>OR2X2M_213</td>
<td>13.12</td>
<td>50.12</td>
</tr>
<tr>
<td>U1</td>
<td>NAND2X2M_471</td>
<td>7.01</td>
<td>57.13</td>
</tr>
<tr>
<td>U12</td>
<td>BUFX2M_130</td>
<td>15.43</td>
<td>72.56</td>
</tr>
<tr>
<td><strong>Path Delay</strong></td>
<td></td>
<td></td>
<td><strong>72.56</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Variation Cell Instance</th>
<th>Cell Delay (ps)</th>
<th>Total Delay (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>U11</td>
<td>INVX2M_435</td>
<td>5.32</td>
<td>5.32</td>
</tr>
<tr>
<td>U10</td>
<td>OR2X2M_268</td>
<td>15.28</td>
<td>20.59</td>
</tr>
<tr>
<td>U8</td>
<td>NAND2X2M_134</td>
<td>9.71</td>
<td>30.30</td>
</tr>
<tr>
<td>U4</td>
<td>INVX2M_329</td>
<td>7.19</td>
<td>37.49</td>
</tr>
<tr>
<td>U3</td>
<td>OR2X2M_474</td>
<td>16.12</td>
<td>53.62</td>
</tr>
<tr>
<td>U1</td>
<td>NAND2X2M_310</td>
<td>7.32</td>
<td>60.94</td>
</tr>
<tr>
<td>U12</td>
<td>BUFX2M_304</td>
<td>15.45</td>
<td>76.39</td>
</tr>
<tr>
<td><strong>Path Delay</strong></td>
<td></td>
<td></td>
<td><strong>76.39</strong></td>
</tr>
</tbody>
</table>
Table 6.5: Probability of obtaining a faster critical path delay than the estimations of randomised netlists.

<table>
<thead>
<tr>
<th>Path Delay (ps)</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Randomised Netlist 1</td>
<td>72.56</td>
</tr>
<tr>
<td>Randomised Netlist 2</td>
<td>76.39</td>
</tr>
</tbody>
</table>

Figure 6.12: The cumulation of the Gaussian models of delays through the critical path of a 35nm 1bit Adder circuit are annotated with the estimated delays of two randomised netlists. The estimated probabilities of obtaining randomised netlists faster than netlist 1 and netlist 2 are 0.57 and 0.98 respectively. Netlist 2 is estimated to be within the top two percent of slowest circuits and may be of some interest to the designer.

It is not important for the delay estimations of the randomised netlists to match with the results of full STA analysis for this method to function effectively. It is important that there is a correlation between the position of the estimated delay within the Gaussian model, and the position of the delay calculated by STA within the distribution of delays generated by a full MCSTA run. If such a correlation exists then it would allow the designer or automated analysis tool to discard the vast majority of samples that are of little or no interest, if there is no such correlation then samples that are viewed as trivial may be discarded when in fact they represent the extremes in circuit behaviour.

Two test circuits were selected for an investigation into whether such a correlation exists between the proposed Gaussian estimations and the actual distribution of delays; a one bit full adder and an ISCAS-85 benchmark four bit fast adder (c74283). These are two of the test circuits that were used within Section 5.2, which were synthesized using a custom scaled 35nm technology. The same VCL was used as within the previous experiments, containing 500 variation instances of an inverter, a buffer and two input NOR, NAND,
OR and AND gates. These variation instances were again based upon multiple width 35nm transistor models libraries within RandomSpice, where $V_{th0}$, $U_0$, $rdsw$ and $dsub$ were varied with Gaussian distributions, and the means and standard deviations of the transistor model parameters were dependant upon the transistor widths.

The Monte Carlo Static Timing Analysis method described in Chapter 4 was adjusted to include the generation of Gaussian models as described within this section. STA was performed on nominal versions of the two test circuits, using SCLs that contained no statistical process variation data, and the critical path delays were reported. The input transitions and output loads at each stage of the critical paths were recorded from these nominal analyses, to determine which LUT elements were read within the SCL by the timing analysis tool. The distributions of delays within the VCL were then read at each of these LUT elements, so that the mean and standard deviation of each relevant cell delay could be calculated. These means and standard deviations were used to generate Gaussian models for the delays through the critical paths of the two test circuits. 50,000 randomised netlists were generated for both of the test circuits, and the variation instances within these randomised netlists were used to calculate a predicted delay and predicted probability for the critical paths. All of the randomised netlists were then passed to PrimeTime for STA, allowing the predicted path delays to be compared with the actual path delays, and more importantly allowing the predicted probabilities to be compared with the shape of the measured delay distributions.

### 6.2.3 Results

This section provides a comparison of the critical path distributions generated by MCSTA and the estimations of delays and significance made by using Gaussian models.

#### 6.2.3.1 35nm 1bit Adder

The estimated delays and probabilities of the 50,000 randomised 1bit Adder netlists were collected and compared against the Gaussian model of the critical path. Figure 6.13 shows the PDF of the Gaussian model, combined with the positions of the 50,000 samples within the distribution, which represents the statistical coverage of the samples used within the MCSTA run. When the CDF of the distribution is viewed, Figure 6.14 the highest estimated probability is 0.99999999206 while the lowest is 0.00005544515, showing that the samples both represent the extremes of the distribution and that the center of the distribution is effectively covered.

A comparison of the distribution of measured delays against the predicted delays reveals that the measured distribution is 8% slower than the estimated distribution, showing as expected that the model used to produce the Gaussian model does not accurately
Chapter 6 Implementation - Practical MCSTA

Figure 6.13: The Gaussian model of the distribution of delays through the critical path of the 1bit Adder circuit, with the points at which each of the 50,000 randomised netlists appear on the distribution. The distribution is well represented by the 50,000 samples.

Figure 6.14: CDF of the Gaussian model of delays through the critical path of the 1bit Adder circuit, with the points at which each of the 50,000 randomised netlists appear on the distribution. 648 of the samples are estimated to be within the slowest 1% of circuit performance.

replicate the STA process, Table 6.6. The differences in delays occur due to the absence of interpolation between LUT elements used by the generation of the Gaussian model, combined with the lack of constraints, parasitics and no propagation of transition times.
The distribution of measured critical path delays is also less symmetrical than the Gaussian model, as observed from previous MCSTA experiments, indicating that improving the method of generating the Gaussian model would simply decrease the difference between the means of the measured and estimated distributions, and not improve the shape of the estimated distribution. Histograms of the measured and estimated critical path delay distributions are shown for comparison in Figure 6.15. A scatter plot of the estimated versus the measured delays of the randomised netlists shows that although there is an average 8% error between the values, there is a strong correlation between the estimated and measured figures, Figure 6.16. This means that a relatively slow estimated delay is likely to correspond to a slow measured delay, which supports the goal of using the Gaussian model for predicting which samples will produce an extreme of circuit performance.

Table 6.6: Comparison of the estimated critical path delays with the measured path delays.

<table>
<thead>
<tr>
<th></th>
<th>Standard</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean (ps)</td>
<td>Deviation (ps)</td>
<td>Maximum (ps)</td>
<td>Minimum (ps)</td>
<td></td>
</tr>
<tr>
<td>Estimated Delays</td>
<td>72.21</td>
<td>2.06</td>
<td>83.78</td>
<td>64.31</td>
<td></td>
</tr>
<tr>
<td>Measured Delays</td>
<td>78.23</td>
<td>2.12</td>
<td>89.80</td>
<td>69.92</td>
<td></td>
</tr>
<tr>
<td>Error (ps)</td>
<td>6.02</td>
<td>0.07</td>
<td>6.02</td>
<td>5.61</td>
<td></td>
</tr>
<tr>
<td>% Error</td>
<td>8.34</td>
<td>3.34</td>
<td>7.18</td>
<td>8.73</td>
<td></td>
</tr>
</tbody>
</table>

Figure 6.15: Histograms of estimated and measured critical path delays for 50,000 randomised samples of the 35nm 1bit Adder circuit. The measured delays are on average 8% slower than the predicted delays.

A CDF of the distribution of measured delays can be generated by arranging the delays in ascending order and assigning each ordered sample a cumulative probability of the
sample number divided by the total number of samples Equation 6.7. This assumes that the measured distribution represents the full and complete range of possible circuit behaviours, as the last sample (the 50,000th slowest delay in this case) would be given a probability of 1. The purpose of this assumption is not to state categorically that 100% of all manufactured circuits will perform faster than the 50,000th slowest sample, but is instead to provide a reference against which the predicted probabilities can be compared. 100% of the estimated probabilities are expected to be lower than the estimated probability of the 50,000th slowest sample.

\[ P_{\text{Sample Number}} = \frac{\text{Sample Number}}{\text{Number of Samples}} \] (6.7)

This expectation can be visually assessed by plotting the probabilities of the measured delays against the predicted probabilities of each of the randomised netlists, Figure 6.17, where in the case of the 1bit Adder, correlation between the estimated and measured probabilities becomes much higher at the highest and lowest extremes of the distribution than in the center. An estimated probability of around 0.5 produces the widest spread in measured probabilities of between 0.2 and 0.8, while an estimated probability of around 0.95 corresponds to a spread of less than 0.2, between 0.8 and 1. This range of accuracy in the predicted probability is best put into context by comparing the CDF of the measured delays with the predicted probabilities, Figure 6.18, where the tails of the probability distribution and delay distribution can be seen simultaneously. In this context it is easier to see how the probability estimations can be used to obtain the longest (or
shortest) delays through the critical path, as a probability estimate of 0.5 corresponds to measured delays within the center of the distribution between probabilities of 0.2 and 0.8, while estimated probability of 0.99 corresponds to measured delays within the slowest 10%. The fact that the Gaussian model predictions are not 100% accurate simply means that the model can not replace MCSTA with a single circuit analysis, but these results show that this very basic method may be used to guide the selection of samples for an MCSTA run. The spread of estimated probabilities is generally symmetric around the measured center line of the measured probability CDF, so the average of a number of samples at an estimated probability point may closely reflect the value that would be found by performing MCSTA.

![Figure 6.17: A scatter plot of estimated probabilities against measured probabilities indicates that there is a weak correlation for predictions within the center of the distribution, while there are strong correlations at the extremes of the distribution.](image)

For the estimated probabilities to be useful it must be possible to select certain areas of the full MCSTA distribution that may be of interest for a designer, which is the case if the designer sets a threshold for full STA against which the estimated probabilities can be compared. Figure 6.19 contains four histograms where the threshold for estimated probabilities has been adjusted, a full MCSTA run (including estimated probabilities between 0 and 1), and estimated probabilities of 0.05 (from 0 to 0.1), 0.50 (from 0.45 to 0.55) and 0.95 (from 0.90 to 1). The distributions obtained from the smaller sample sets within the estimated probability thresholds lie within the appropriate areas of the full MCSTA distribution, i.e. the lowest 10%, middle 10% and highest 10%. The shapes of these sampled distributions do not match perfectly, spilling into the regions outside of the desired thresholds, with the shortest delay from the highest 10% samples overlapping with the longest delay from the middle 10% samples. This is due to the nature of the way
Figure 6.18: The aim of generating a probability estimate with a Gaussian model is to predict where the measured delay appears within the measured distribution. This plot provides a comparison of the cumulative distribution of measured critical path delays through the 1bit Adder against the probability that was estimated for each randomised netlist.

The estimations are calculated, quick and inaccurate, meaning that the average value of the retained samples is of more value than either of the extremes.

A measure of the effectiveness of the probability estimations is to attempt to predict the critical path delays of the adder to the 3-sigma level that is often the goal of statistical analysis, where the designer may wish to be able to predict the behaviour of 99.73% of the distribution of delays, derived from the fact that the mean of a Gaussian distribution plus/minus 3-sigma represents a 99.73% confidence interval. In the case of predicting the worst case delays of a circuit this requires predicting the behaviour of 99.865% of the circuits, which is the probability of obtaining a value at plus 3-sigma of a Gaussian distribution. Figure 6.20 contains a plot of the histogram of critical path delays from select samples, where only those randomised adder netlists whose estimated probability was greater than or equal to 0.99865 were retained; the tail of the distribution of delays generated by the 50,000 MCSTA samples are also provided as a reference. The delay with a probability of 0.99865 in the CDF for the full MCSTA distribution was found to be 85.13ps, while the mean delay of the selected samples was found to be 85.28ps, a relative error of 0.18%. The slowest tail of the 50,000 sample MCSTA distribution is accurately recreated by the selected samples, of which there were 126, requiring 0.25% of the computation time of the full distribution. These results are shown in Table 6.7, where the mean delay of the samples obtained from a range of probability thresholds
is compared with the delay obtained for the same probability threshold from the CDF of the 50,000 sample MCSTA run. A small but significant point to observe from this table is that there is a 0.05% error between the full 50,000 sample MCSTA distribution and the 50,000 samples retained when the probability threshold is set from 0 to 1. This is due to the fact that the median of the distribution is not equal to the mean of the distribution, which is what is being compared in the first entry within the results table. The accuracy of sampled method remains to within 1% of the whole distribution, even when the retained sample size is only 23 for a probability threshold of between 0.9999 and 1 (1 in 10,000 circuits), which requires 0.05% of the computation time required for the 50,000 samples.

The results so far have been based upon a comparison of a 50,000 sample size MCSTA run and the estimated probabilities of each of the 50,000 randomised netlists within the run. For the proposed method to be of use it must be possible for a designer to specify which probability threshold is required for examination, and to specify a limit on the number of samples that must be obtained within the threshold. For this reason the experiment was repeated with a limit on the number of samples that were retained for each probability threshold. In this case randomised netlists and probability estimations were repeatedly generated until 50 of the estimations were found to be within the target threshold, these 50 netlist were retained and STA was performed upon each one. The number of randomised netlists that were discarded was recorded to illustrate how large the full MCSTA run would have to have been to capture the same results. Table 6.8

![Figure 6.19: Histograms of delays for randomised netlists that were selected for their estimated probabilities.](image)
Figure 6.20: A histogram of delays for randomised netlists that were selected with an estimated probability of over 3-sigma, 0.99865. A plot of the tail of the distribution of delays obtained from 50,000 MCSTA samples is also included.

Table 6.7: Comparison between critical path delays from samples selected by their estimated probability thresholds and critical path delays from the CDF of the 50,000 Sample 1bit Adder MCSTA run.

<table>
<thead>
<tr>
<th>Probability Threshold</th>
<th>Full MCSTA Delay (ps)</th>
<th>Mean Sampled Delay (ps)</th>
<th>%Error</th>
<th>No. Samples</th>
<th>%CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00000 to 1.00000</td>
<td>78.19</td>
<td>78.23</td>
<td>0.05</td>
<td>50000</td>
<td>100.00</td>
</tr>
<tr>
<td>0.00000 to 0.00135</td>
<td>72.13</td>
<td>71.64</td>
<td>0.69</td>
<td>42</td>
<td>0.08</td>
</tr>
<tr>
<td>0.00000 to 0.10000</td>
<td>74.82</td>
<td>74.70</td>
<td>0.15</td>
<td>4917</td>
<td>9.83</td>
</tr>
<tr>
<td>0.45000 to 0.55000</td>
<td>78.19</td>
<td>78.23</td>
<td>0.06</td>
<td>4968</td>
<td>9.94</td>
</tr>
<tr>
<td>0.90000 to 1.00000</td>
<td>81.80</td>
<td>81.95</td>
<td>0.18</td>
<td>5025</td>
<td>10.05</td>
</tr>
<tr>
<td>0.98000 to 1.00000</td>
<td>83.40</td>
<td>83.36</td>
<td>0.04</td>
<td>1191</td>
<td>2.38</td>
</tr>
<tr>
<td>0.99865 to 1.00000</td>
<td>85.13</td>
<td>85.36</td>
<td>0.16</td>
<td>97</td>
<td>0.19</td>
</tr>
<tr>
<td>0.99900 to 1.00000</td>
<td>85.39</td>
<td>85.53</td>
<td>0.91</td>
<td>23</td>
<td>0.05</td>
</tr>
<tr>
<td>0.99990 to 1.00000</td>
<td>87.54</td>
<td>86.75</td>
<td>0.18</td>
<td>20</td>
<td>0.25</td>
</tr>
</tbody>
</table>

shows the number of randomised netlists that had to be generated to find 50 probability estimates within the same probability thresholds that were used previously. STA of the 50 retained samples required an average of 65 seconds, including initialising and obtaining licenses for PrimeTime, and this run time is independent of the probability threshold that was chosen.

Smaller probability thresholds required larger numbers of randomised netlist generation, especially when the thresholds are set to the extremes of the distribution. The generation and probability estimations of 137,862 netlists was achieved within 11 seconds, saving
Table 6.8: The number of randomised netlists generated to find 50 netlists with a probability estimate within the target threshold, and a comparison of the time taken to perform STA on the 50 selected samples rather than all of the generated samples

<table>
<thead>
<tr>
<th>Probability Threshold</th>
<th>Samples Generated</th>
<th>Generation Time (s)</th>
<th>50 Sample STA Time (s)</th>
<th>Generated STA Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000000 to 1.00000</td>
<td>50</td>
<td>5</td>
<td>62</td>
<td>62</td>
</tr>
<tr>
<td>0.000000 to 0.00135</td>
<td>62364</td>
<td>8</td>
<td>67</td>
<td>49392</td>
</tr>
<tr>
<td>0.000000 to 0.10000</td>
<td>579</td>
<td>5</td>
<td>65</td>
<td>464</td>
</tr>
<tr>
<td>0.450000 to 0.55000</td>
<td>553</td>
<td>4</td>
<td>62</td>
<td>443</td>
</tr>
<tr>
<td>0.900000 to 1.00000</td>
<td>366</td>
<td>4</td>
<td>61</td>
<td>293</td>
</tr>
<tr>
<td>0.980000 to 1.00000</td>
<td>2402</td>
<td>5</td>
<td>69</td>
<td>1925</td>
</tr>
<tr>
<td>0.998650 to 1.00000</td>
<td>21270</td>
<td>6</td>
<td>63</td>
<td>16846</td>
</tr>
<tr>
<td>0.999000 to 1.00000</td>
<td>24134</td>
<td>6</td>
<td>63</td>
<td>19114</td>
</tr>
<tr>
<td>0.999900 to 1.00000</td>
<td>137862</td>
<td>11</td>
<td>68</td>
<td>109187</td>
</tr>
</tbody>
</table>

approximately 109000 seconds (30 hours) that would be required to perform STA on all of the discarded netlists, rather than simply the 50 that were retained for the probability threshold of between 0.9999 and 1. This probability threshold was only represented by 23 out of 50,000 randomised netlists in the previous experiment, but 50 samples were generated and analysed within 79 seconds providing a significant saving in computation time for the designer. Table 6.9 shows the delays obtained from each of the samples of 50 randomised netlists, compared with the delays obtained from the CDF of the 50,000 sample MCSTA run. The percentage error between the large MCSTA run and the small sample sizes remains very small, with the error only just rising above 1% for the most extreme of the probability thresholds, where twice as many netlists were generated using the probability threshold than within the 50,000 netlist MCSTA run. In this case the distribution of 50,000 delays may not accurately reflect the distribution that would have been obtained by the 137862 netlists generated to find 50 samples within the probability threshold, in which case the 50,000 samples no longer serves as a golden reference. The probability threshold that was chosen to reflect an accuracy of 3-sigma (0.99865 to 1) provides a mean delay that is within 0.2% of delay obtained from the 50,000 sample MCSTA run, saving over 11 hours of computation time for a very acceptable loss of accuracy.

6.2.3.2 35nm 4bit Fast Adder

Gaussian models were generated for the critical path through the 4bit Fast Adder circuit, with an estimated mean of 75.66ps and estimated standard deviation of 2.06ps. STA was performed on 50,000 randomised netlists of the 4bit Fast Adder using PrimeTime. The sample size was found to be large enough to generate estimated delays that covered the entire range of the Gaussian distribution, Figure 6.21, but the mean and spread of
Table 6.9: Comparison between critical path delays from the first 50 samples to be generated within a range of estimated probability thresholds and critical path delays from the CDF of the 50,000 Sample MCSTA run.

<table>
<thead>
<tr>
<th>Probability Threshold</th>
<th>Full MCSTA Delay (ps)</th>
<th>Mean Sampled Delay (ps)</th>
<th>%Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000000 to 1.00000</td>
<td>78.19</td>
<td>78.68</td>
<td>0.63%</td>
</tr>
<tr>
<td>0.000000 to 0.00135</td>
<td>72.13</td>
<td>71.65</td>
<td>0.67%</td>
</tr>
<tr>
<td>0.000000 to 0.10000</td>
<td>74.82</td>
<td>74.61</td>
<td>0.28%</td>
</tr>
<tr>
<td>0.450000 to 0.55000</td>
<td>78.19</td>
<td>78.19</td>
<td>0.00%</td>
</tr>
<tr>
<td>0.900000 to 1.00000</td>
<td>81.80</td>
<td>81.87</td>
<td>0.10%</td>
</tr>
<tr>
<td>0.980000 to 1.00000</td>
<td>83.40</td>
<td>83.37</td>
<td>0.03%</td>
</tr>
<tr>
<td>0.998650 to 1.00000</td>
<td>85.13</td>
<td>85.31</td>
<td>0.20%</td>
</tr>
<tr>
<td>0.999000 to 1.00000</td>
<td>85.39</td>
<td>85.49</td>
<td>0.11%</td>
</tr>
<tr>
<td>0.999900 to 1.00000</td>
<td>87.54</td>
<td>86.6</td>
<td>1.09%</td>
</tr>
</tbody>
</table>

The measured delays were found to be considerably larger than the estimates. There was a 26% increase in the mean delay and a 10% increase in standard deviation when compared to the estimates, providing no overlap between the range of the Gaussian model and the measured distribution of delays, Table 6.10. This can be more clearly seen from a plot of histograms taken of the distributions of estimated and measured delays Figure 6.22.

Figure 6.21: CDF of the Gaussian model of delays through the critical path of the Fast Adder circuit, with the points at which each of the 50,000 randomised netlists appear on the distribution. 595 of the samples are estimated to be within the slowest 1% of circuit performance.
Table 6.10: Comparison of the Gaussian estimations of critical path delays with the 50,000 sampled and measured path delays, for the 35nm Fast Adder Circuit.

<table>
<thead>
<tr>
<th></th>
<th>Mean (ps)</th>
<th>Deviation (ps)</th>
<th>Maximum (ps)</th>
<th>Minimum (ps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Delays</td>
<td>75.66</td>
<td>2.06</td>
<td>84.64</td>
<td>67.57</td>
</tr>
<tr>
<td>Measured Delays</td>
<td>95.56</td>
<td>2.28</td>
<td>106.20</td>
<td>86.29</td>
</tr>
<tr>
<td>Error (ps)</td>
<td>19.90</td>
<td>0.21</td>
<td>21.56</td>
<td>18.71</td>
</tr>
<tr>
<td>% Error</td>
<td>26.3</td>
<td>10.28</td>
<td>25.47</td>
<td>27.69</td>
</tr>
</tbody>
</table>

Figure 6.22: Histograms of estimated and measured critical path delays for 50,000 randomised samples of the 35nm Fast Adder circuit. The measured delays are on average 26% slower than the predicted delays.

A scatter plot of the estimated probabilities against the measured probabilities for the Fast Adder circuit, Figure 6.23, indicated that there was a weaker correlation between the probabilities than was found within the results obtained from the 1bit Adder experiment. The increase in error between the estimations and the measured results after STA is an indication that the Gaussian approach to statistically modelling critical path delays becomes weaker with longer logic paths, but that it is still possible to perform basic statistical sampling. The probability estimations can still be reliably used to predict circuit performance within ranges of desired probability thresholds, such as the slowest, fastest and nominal 10%, Figure 6.24. A range of desired probability thresholds were chosen and were used to assess the validity of the Gaussian modelling process, and it was found that the mean delays of the circuits within the threshold bands matched the delay at the appropriate point within the 50,000 sample CDF to within 0.35%, and less than 0.3% for a prediction of 3-sigma circuit performance, Table 6.11.
Figure 6.23: A scatter plot of estimated probabilities against measured probabilities for the Fast Adder indicates that there is a weak correlation for predictions within the center of the distribution, while there are stronger correlations at the extremes of the distribution.

Figure 6.24: Histograms of delays for randomised netlists of the Fast Adder that were selected for their estimated probabilities.

The ability to predict the performance of a circuit without prior knowledge of the performance is much more important than being able to reproduce a distribution of delays by using sub samples of the distribution. It is therefore vital to be able to achieve similar results using independently generated random samples. The experiment was altered so
Table 6.11: Comparison between critical path delays from samples selected by their estimated probability thresholds and critical path delays from the CDF of the 50,000 Sample Fast Adder MCSTA run.

<table>
<thead>
<tr>
<th>Probability Threshold</th>
<th>Full MCSTA Delay (ps)</th>
<th>Mean Sampled Delay (ps)</th>
<th>%Error</th>
<th>No. Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000000 to 1.000000</td>
<td>95.5</td>
<td>95.56</td>
<td>0.06%</td>
<td>50000</td>
</tr>
<tr>
<td>0.000000 to 0.00135</td>
<td>89.11</td>
<td>88.92</td>
<td>0.22%</td>
<td>47</td>
</tr>
<tr>
<td>0.000000 to 0.100000</td>
<td>91.91</td>
<td>92.03</td>
<td>0.13%</td>
<td>4959</td>
</tr>
<tr>
<td>0.450000 to 0.55000</td>
<td>95.5</td>
<td>95.56</td>
<td>0.06%</td>
<td>4996</td>
</tr>
<tr>
<td>0.900000 to 1.000000</td>
<td>99.39</td>
<td>99.32</td>
<td>0.07%</td>
<td>5009</td>
</tr>
<tr>
<td>0.980000 to 1.000000</td>
<td>101.06</td>
<td>100.71</td>
<td>0.35%</td>
<td>1138</td>
</tr>
<tr>
<td>0.998650 to 1.000000</td>
<td>102.75</td>
<td>102.46</td>
<td>0.29%</td>
<td>109</td>
</tr>
<tr>
<td>0.999000 to 1.000000</td>
<td>102.98</td>
<td>102.77</td>
<td>0.21%</td>
<td>73</td>
</tr>
<tr>
<td>0.999900 to 1.000000</td>
<td>104.19</td>
<td>104.01</td>
<td>0.17%</td>
<td>11</td>
</tr>
</tbody>
</table>

that any randomised netlist with an estimated probability outside of the desired threshold was discarded, and only the first 50 randomised netlists that were found within the threshold were retained and analysed within PrimeTime. The results of using these sets of 50 samples are provided within Table 6.12, where the Full 50,000 sample MCSTA run of the 35nm Fast Adder are retained as a reference against which smaller independent samples are compared. These results indicate that a statistically sampled MCSTA run of just 50 samples can be used replicate the targeted area of a much larger full MCSTA run to within one third of a percent. This method avoids the pessimism of Corner Analysis while dramatically reducing the computation time required to produce accurate delay information at the extremes of circuit behaviour.

Table 6.12: Comparison between critical path delays from the first 50 samples to be generated within a range of estimated probability thresholds and the critical path delays from the CDF of the 50,000 Sample Fast Adder MCSTA run.

<table>
<thead>
<tr>
<th>Probability Threshold</th>
<th>Full MCSTA Delay (ps)</th>
<th>Mean Sampled Delay (ps)</th>
<th>%Error</th>
<th>No. Samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000000 to 1.000000</td>
<td>95.5</td>
<td>95.54</td>
<td>0.04%</td>
<td>50</td>
</tr>
<tr>
<td>0.000000 to 0.00135</td>
<td>89.11</td>
<td>88.96</td>
<td>0.17%</td>
<td>50</td>
</tr>
<tr>
<td>0.000000 to 0.100000</td>
<td>91.91</td>
<td>92.07</td>
<td>0.17%</td>
<td>50</td>
</tr>
<tr>
<td>0.450000 to 0.55000</td>
<td>95.5</td>
<td>95.81</td>
<td>0.32%</td>
<td>50</td>
</tr>
<tr>
<td>0.900000 to 1.000000</td>
<td>99.39</td>
<td>99.4</td>
<td>0.01%</td>
<td>50</td>
</tr>
<tr>
<td>0.980000 to 1.000000</td>
<td>101.06</td>
<td>100.89</td>
<td>0.17%</td>
<td>50</td>
</tr>
<tr>
<td>0.998650 to 1.000000</td>
<td>102.75</td>
<td>102.54</td>
<td>0.20%</td>
<td>50</td>
</tr>
<tr>
<td>0.999000 to 1.000000</td>
<td>102.98</td>
<td>102.66</td>
<td>0.31%</td>
<td>50</td>
</tr>
<tr>
<td>0.999900 to 1.000000</td>
<td>104.19</td>
<td>104.17</td>
<td>0.01%</td>
<td>50</td>
</tr>
</tbody>
</table>
6.2.4 Sample Size Reduction

The ability to accurately predict the behaviour of a specific percentage of circuits will depend upon the probability thresholds and sample sizes specified by the designer. It was therefore important to assess the impact of the sample size and probability threshold selection on the accuracy of the statistically samples MCSTA run. The MCSTA experiment with 50,000 randomised samples of the Fast Adder was repeated, selecting the first \( N \) randomised netlists that were predicted to fall within a chosen probability threshold, where \( N \) is the statistical sample size. The average delay obtained from the \( N \) retained randomised netlists was then compared against the delay obtained from the distribution of 50,000 netlists at the relevant position within the CDF. The results of this experiment are shown within Table 6.13, where the percentage error was never found to be over 1% of the delay obtained from a large MCSTA run of 50,000 samples. This suggests that a sample size of as little as 5 to 10 samples can be used to predict the behaviour of a much larger sample set, by using a simple Gaussian statistical model as a predictor of which samples to select.

Table 6.13: A plot of the percentage error in the average delay obtained by a range of sample sizes for a series of different probability thresholds. The percentage errors are with respect to the delay obtained from a distribution of 50,000 samples.

<table>
<thead>
<tr>
<th>Number Of Samples</th>
<th>Probability Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.00000 to 0.10000</td>
</tr>
<tr>
<td></td>
<td>0.45000 to 0.55000</td>
</tr>
<tr>
<td></td>
<td>0.90000 to 1.00000</td>
</tr>
<tr>
<td></td>
<td>0.99865 to 1.00000</td>
</tr>
<tr>
<td></td>
<td>0.99990 to 1.00000</td>
</tr>
<tr>
<td>5</td>
<td>0.85%</td>
</tr>
<tr>
<td>10</td>
<td>0.11%</td>
</tr>
<tr>
<td>15</td>
<td>0.20%</td>
</tr>
<tr>
<td>20</td>
<td>0.25%</td>
</tr>
<tr>
<td>25</td>
<td>0.30%</td>
</tr>
<tr>
<td>50</td>
<td>0.17%</td>
</tr>
</tbody>
</table>

6.2.5 Summary

Statistical analysis of a circuit under variation must be performed rapidly if the statistical data is to be useful within the design process, rather than simply at the final sign-off stage. Timing analysis is performed at multiple points of the automated design flow, as the delays through a circuit influence the decisions taken by synthesis, placement and routing tools. Any such decision, such as a small change in the routing of a circuit, may require the re-timing of a design, which is a daunting task if multiple large MCSTA runs are required to provide an accurate prediction of the impact of variation upon delays.
The two approaches of Statistical Static Timing Analysis and Monte Carlo Static Timing Analysis can be combined to provide a rapid, accurate prediction of the performance of a circuit that is subject to statistical process variations. Simple Gaussian models can be created from the distributions within a Variation Cell Library, allowing each standard cell to be represented by multiple Gaussian models, one for each position within each delay and power look up table. These Gaussian models are then used to produce a very crude estimate of the distribution of delays through or power consumption of a group of cells. The MCSTA process is then performed for the circuit, and the randomly selected variation instances within the group of cells within each randomised netlist is briefly compared against the estimated distribution of performance. If the comparison indicates that the randomised netlist is within a desired probability threshold then the netlist can be used for full timing and power analysis, otherwise the netlist is discarded.

The generation, assessment, selection and STA of 10 samples of the Fast Adder circuit with a probability threshold of over 3-sigma was achieved over five hundred times faster than performing STA on all of the netlists that were rejected, while the prediction of 3-sigma path delays from the 10 retained samples was found to be within 0.3% of the delay obtained from 50,000 samples.

The significant increase in accuracy that MCSTA provides over the widespread practice of Corner Analysis and modern SSTA techniques can be obtained with a very small sample size when simple statistical sampling methods are introduced. This provides a rapid, accurate and practical approach to modelling the impact of statistical process variations on circuit performance, throughout the design cycle.
Chapter 7

Conclusions And Future Work

The overall aim of this thesis was to predict and analyse the impact of statistical process variations on the performance of digital circuits using nano-scale CMOS technologies. These predictions were required to be accurate when compared to Monte Carlo SPICE simulations and the predictive methods were required to be transparent and repeatable for a circuit designer. To meet these aims a Monte Carlo Static Timing Analysis approach was taken, where the statistical process variations of transistors that were predicted within a 3D Atomistic Simulator were captured within a Variation Cell Library, using industry standard cell characterisation tools. The MCSTA technique was compared with Corner Analysis and Statistical Static Timing Analysis using 130nm and 35nm technology nodes, and was found to provide the most accurate representation of the SPICE distribution of circuit behaviours. MCSTA provides the accurate timing information required by the design industry and provides a novel approach to the prediction of power consumption distributions. The technique was further refined to produce accurate predictions of the extremes, or corners, of circuit performance distributions by using a simple statistical sampling method. A novel, rapid and efficient Metric of Variability was also presented which allows designers to make very early design decisions that may increase the tolerance of a circuit to statistical process variations.

This chapter first provides a summary of the work that was accomplished within this thesis, and continues to suggest possibilities of future work.

7.1 Conclusions and Contributions

The first chapter provided a list of clearly stated objectives:

"The goal of the nano-CMOS project was to provide accurate and efficient predictions of the effects of manufacturing process variations on the power, performance and yield..."
of designs. The objectives and contents of this thesis are strongly aligned with the goals of the nano-CMOS project.

The first objective is to model the impact of statistical manufacturing process variations on standard cells, by providing a transparent and repeatable method of cell characterisation that makes use of commercially available tools and fits within the existing design flow.

The second objective is to use the characterised cells to predict the impact of statistical process variations within circuits, providing design engineers with a clear method of generating the critical information that will allow a design to be signed off for manufacturing.

The third objective is to demonstrate the accuracy of the proposed methods, illustrating the benefits over existing industry standards and alternative statistical modelling methods, and demonstrating ways in which power, performance and yield trade-offs can be made.

The final objective is to illustrate that the proposed methods not only allow for a final design sign-off against process variations, but also allow for practical, rapid predictions to be made throughout the design flow, allowing early design decisions to be made that can increase the tolerance of a design to process variations and reduce the possibility of costly design re-work.”

In Chapter 2 the complexities of designing digital circuits using modern process technologies was discussed. The challenges posed by variability were introduced, and the limitations of the current design strategies and methods were highlighted. A literature review of the methods used to predict the performance of modern digital circuits was provided, concluding that SSTA has not been widely adopted by the industry and a modified Monte Carlo approach could provide a more reliable and transparent approach.

Chapter 3 built upon the concept of using Monte Carlo simulations by introducing Monte Carlo Cell Characterisation. Work performed by other members of the nano-CMOS project was described, including a 3D atomistic transistor simulator and the RandomSpice statistical circuit simulation tool, which provided the source of the statistical process variation models used within this thesis. The process of Cell Characterisation was described, and a method of incorporating statistical variation within the standard practice of cell characterisation was demonstrated. The statistical behaviour of a range of standard cells was analysed and characterised using the industry standard Liberty format, completing the first objective of this thesis by generating Variation Cell Libraries.

Chapter 4 satisfied the second objective of the thesis of propagating the performance distributions of cells within Variation Cell Libraries to gate level netlists. The standard industry practices of timing and power analysis were introduced, demonstrating how
the information within a Standard Cell Library is used to predict the behaviour of a complete circuit. The steps required to use a Variation Cell Library (rather than a Standard Cell Library) were demonstrated by altering the standard cell references within a gate level netlist to address variation instances within the VCL. The accuracy of MCSTA was demonstrated as being within 5% of the delays obtained from Monte Carlo SPICE simulations, when performed on a section of a 130nm SpiNNaker test chip. The analysis indicated that STA can be used to accurately model SPICE, and that the distribution of Monte Carlo SPICE simulations can be rapidly replicated by a Monte Carlo STA run.

The standard practice of performing Corner Analysis is described within the literature as being overly pessimistic, especially when modelling intra-die process variations. Such pessimism can lead to longer, more costly, design stages, or may force design teams to abandon the optimisation of a design before maximum performance is achieved. The process of MCSTA was introduced as a method of reducing this pessimism, sampling over a range of possible cell delays, rather than assuming that all transistors within a chip simultaneously perform at their worst possible behaviour. Chapter 5 provided a critical assessment of MCSTA, verifying whether the claims of reduced pessimism were correct. MCSTA was used to assess the performance of a 1-bit full adder implemented on a 130nm technology, and the technique was found to be an effective compromise between the accuracy of a Monte Carlo SPICE simulation and the practicality of abstracted Corner Analysis using STA. Predictions of the 99.7th percentile of circuit performance were within 1.2% of SPICE for power consumption and 3% of SPICE for maximum path delays, at the highest level of injected variability. Corner analysis was found to have significant errors of 782% and 548% for power and delay at the same level of variability.

The effectiveness of MCSTA was further demonstrated within Chapter 5 on a series of test circuits implemented on a 35nm process, where greater accuracy (with respect to SPICE) was found when compared to the timings generated by SSTA. MCSTA produced delay distributions that more closely matched the shape of the distribution generated by SPICE, and provided an estimation of worst case delay with an average of 4% less pessimism than SSTA. MCSTA also had the significant benefit of allowing the user to generate distributions of static and dynamic power consumption. Three dimensional plots of Power, Performance and Yield, provided an indication of the percentage of circuits that would meet a specified timing or power constraint, allowing designers to assess the impact of manufacturing process variations on a circuit, enabling budgeting and marketing decisions to be made based on the number of fabricated chips that can be sold at different performance points. MCSTA can also be used to assess the probability of individual timing paths becoming critical, and determine if any areas of a design act as a timing bottleneck. The findings within this chapter completed the third objective of the thesis.
In Chapter 6 a practical Metric of Variability was presented, which allows designers (and automated synthesis tools) to make design implementation decisions based on the impact of statistical process variations on path delays. Work within [4] was assessed using the MCSTA method introduced within this thesis, and alterations to the calculation of the number of transistors within the switching path of a gate were suggested.

\[
\frac{\sigma_{\text{delay}}}{\mu_{\text{delay}}} = \frac{k}{\sqrt{n_{\text{stack}}} \sqrt{\text{strength}}}
\]

Where \( N \) is the number of gates belonging to the considered path, \( \text{strength} \) is the driving strength of the cell, \( n_{\text{stack}} \) is a scaling factor for the number of stacked transistors in the switching path, and \( k \) is a technology-dependent constant evaluated from preliminary simulations. The corrected values of \( n_{\text{stack}} \) in the presented trials was found to be \( 1 + (0.2 \times (\text{Number of Transistors in Switching Path} - 1)) \), which produced estimates of variation with an average error of 3% when compared to the generated MCSTA distributions. The speed at which these simple calculations can be performed means that the metric provides a very simple, efficient and novel early predictor of the impacts of statistical process variations.

Chapter 6 also provides a description of how statistical modelling can be used to enhance the process of MCSTA. Simple Gaussian models of standard cell performance were generated and used to predict the probability of the occurrence of the randomised netlists within a MCSTA run. The correlation between the predicted probabilities and actual probabilities was found to be strong within the tails of the generated distributions, which is the key area of interest for circuit designers. Small sample sizes of 10 were found to produce an average delay within 0.3% of the delay obtained from a sample size of 50,000. The proposed statistical sampling method allows a designer to accurately target the area of interest of a circuit performance distribution and perform MCSTA on only the relevant samples, dramatically reducing the analysis time required, providing a very practical alternative to SSTA, and meeting the final objective of this thesis.

The work undertaken within this thesis has described a unique method for modelling statistical process variations at the circuit level, MCSTA. The results of this method have been verified against SPICE simulations, Corner based analysis and SSTA. MCSTA does not require any assumptions of the distributions or correlations of process parameters, and makes use of existing CAD tools. The significant increase in accuracy that MCSTA provides over the widespread practice of Corner Analysis and modern SSTA techniques can be obtained with a very small sample size when simple statistical sampling methods are introduced. This provides a rapid, accurate and practical approach to modelling the impact of statistical process variations on circuit performance, throughout the design cycle.
7.2 Future Work

The research within this thesis was focused on the distributions of delays and power consumption obtained in a 130nm commercial process and a 35nm commercial process. The 35nm process has since been superseded by the regular usage of 28nm process technologies and advanced research into 14nm technologies. These advanced processes represent an even greater challenge to the design industry and so future work will be required by using these technologies to confirm the suitability and effectiveness of MCSTA. A repeat of the comparison between MCSTA and SSTA at such an advanced process node as 14nm would provide a significant illustration of the impact of statistical process variations.

The ultimate aim of this research has been to allow designers to predict what impact statistical process variations will have on their designs, as early as possible, using commercially available, standardised tools. This goal has been achieved by the development of the RandomSpice tool by members of the nano-CMOS project, and by the development of the Monte Carlo Cell Characterisation and Monte Carlo Static Timing Analysis processes within this thesis. The Monte Carlo approaches described within this thesis have been achieved using simple Perl scripts to read the contents of VCLs, randomise gate level verilog netlists and launch multiple STA sessions. Further improvements in computation time and memory usage could be achieved by designing the contents of the Perl scripts into the commercial CAD tools, especially when making use of statistical sampling, as the most significant overhead of MCSTA is the reading of the Variation Cell Libraries and the writing of the randomised netlists to disk so that they may be read in the future by the CAD tool. These overheads could be removed if the randomisation process was incorporated within the CAD tool, as the reading of the Variation Cell Library is already required as part of the STA process, and the randomisation of the netlists could be retained in memory.

Further improvements in the efficiency of the MCSTA process can be achieved by making use of parallel processing, as the STA of each randomised netlist can be performed independently of the other randomised netlists. The combination of the reduced sample sizes when using statistical sampling and parallel processing would allow accurate delay and power distributions to be generated by MCSTA runs that take little longer (real time) than a traditional single STA run. If MCSTA can be achieved with no noticeable increase in the time taken for a designer to analyse a design then this approach may become the standard method for predicting the impact of statistical process variations on sub-nanometre digital circuits.
Appendix A

Publications

The following papers were published during the course of the research conducted within this thesis:


Bibliography


