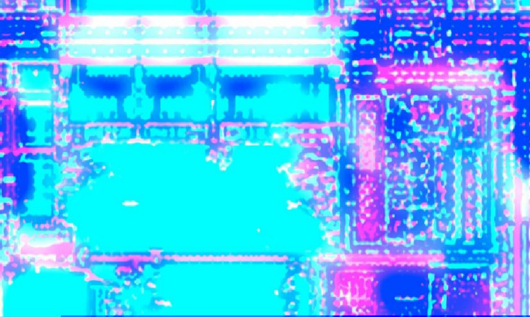


Physical Design Essentials

An ASIC Design Implementation Perspective

Khosrow Golshan

Springer

PHYSICAL DESIGN ESSENTIALS

An ASIC Design Implementation Perspective

PHYSICAL DESIGN ESSENTIALS
An ASIC Design Implementation Perspective

Khosrow Golshan
Conexant Systems, Inc.

Springer

Khosrow Golshan
Conexant Systems, Inc.
Newport Beach, CA

Physical Design Essentials: An ASIC Design Implementation Perspective

Library of Congress Control Number: 2006932950

ISBN 0-387-36642-3 e-ISBN 0-387-46115-9
ISBN 978-0-387-36642-5 e-ISBN 978-0-387-46115-1

Printed on acid-free paper.

Printed in the United States of America.

9 8 7 6 5 4 3 2 1

springer.com

Trademarks

Verilog is a registered trademark of Cadence Design Systems, Inc.

SDF and SPEF are trademarks of Open Verilog International.

All other brand or product names mentioned in this document are trademarks or registered trademarks of their respective companies or organizations.

Disclaimer

The information contained in this manuscript is an original work of the author and intended for informational purposes only. The author disclaims any responsibility or liability associated with the content, the use, or any implementations created based upon such content.

Dedication

To my father, Aziz, my wife, Maury, my son, Troy and his family, Rebecca, his wife and their daughters Madison and Darya

Contents

Preface

The goal of this book is to provide the essential steps required in the physical design of Application Specific Integrated Circuits (ASIC). It is the intention that the book present self-contained material and enough detail so as to give the reader a basic idea of ASIC design implementation.

The first generation of modern electronics is thought to have begun in 1942 with the invention of electronic switches and miniature vacuum tubes. By 1946, a large-scale computing device based on these new inventions was developed. The computing device was known as Electronic Numerical Integration and Calculation (ENIAC). ENIAC could perform thousands of calculations per second and was used in many applications such as scientific research and weather predictions.

With the introduction of the first working transistor in 1948, the second generation of the electronic era began. This era was mainly characterized by the change from vacuum tubes to transistor technology. Vacuum tubes were gradually replaced in the design of switching circuits by discrete transistors.

By 1965, the third generation of electronics began with the development of the Integrated Circuit (IC). The IC started to replace discrete transistor circuits. In addition, semiconductor memories, such the Read Only Memory (ROM) and Random Access Memory (RAM), began to augment the system designs. This resulted in the substantial reduction of the physical size and cost of the systems. This generation propelled the rapid integration of circuit design forms (small, medium, and large) to very large devices that contained millions of transistors.

These tremendous achievements were made possible by the development of IC processing equipment, design tools, and software. In the past fifteen years, the world not only has been witness to the rapid reduction in the feature size of transistors (from 1000 to 45 nanometer), but also to the dramatic innovation of sophisticated physical design automation tools.

The complexities of today's ASIC physical designs require a mix of backgrounds in electrical engineering, computer science, and IC processes. Such diversified knowledge has created a new discipline in engineering – the physical design engineer. Today's physical design engineers are expected to be conversant with all aspects of ASIC design implementation stages that include device processes, library development, place-and-route algorithms, verification and testing.

This book is arranged in a format that follows the industry-common ASIC physical design flow. It begins with the general concept of an ASIC library, then covers floorplanning, placement, routing, verification, and finally, testing. Topics covered include:

- Basic standard cell design, transistor sizing, and layout styles
- Linear, nonlinear, and polynomial characterization
- Physical design constraints and floorplanning styles
- Algorithms used for placement
- Clock Tree Synthesis
- Algorithms used for global and detailed routing
- Parasitic extraction
- Functional, timing, and physical methods of verification
- Functional, scan, parametric, memory and parallel module test

Rather than go into lengthy technical depths, the emphasis has been placed on short, clear descriptions complemented by references to authoritative manuscripts for those desiring further information on each chapter. It is the goal of this book to capture the essence of physical design, and to introduce to the reader the challenging and diversified field of physical design engineering.

Khosrow Golshan
Engineering Division Director, Conexant Systems, Inc.
December 2006

Foreword

In the year 2006, the semiconductor industry is well into its sixth decade of existence. Amazingly, the original Bell Lab invention of the transistor combined a decade later with the refining implementation of an integrated circuit continues to drive exponential economic gain in the form of both improved productivity and growth of new applications. Today, an integrated circuit can perform almost any electronic function on a piece of silicon less than a centimeter square.

From a PC, to a cell phone, to a television, to an internet router, the industry can conceive, design and deliver that function in an extraordinarily cost-effective and convenient form a–single silicon IC. The result looks so deceivingly simple it can be, and is, taken for granted by an entire population.

However, for the industry and design community who must continue to deliver to this value proposition, none of it can be taken for granted. Fully appreciating the science, technology and, dare I say, art that must be mastered to implement an IC that contains tens of millions of individual transistors is a must for those who want to drive semiconductor industry success through the coming decades.

Physical Design Essentials – An ASIC Design Implementation Perspective by Khosrow Golshan is a resource that can be used by all industry participants to help develop just that appreciation.

For the student who is looking to develop a skill and expertise to participate in the industry the book provides exactly what the title suggests a– perspective. I would encourage the student to read the book to gain an understanding of the overall process that drives the physical design community. The student is likely to be studying in depth one of the areas touched on by the book. Whether that be circuit design and analysis, or CAD tool development, the context provided by the book will undoubtedly provide insight that will enhance the learning process.

For the designer who is directly contributing in one or more of the areas of design covered in the book, the book will become a valued reference, complementing and updating the many references with which the designer is already familiar. Mr. Golshan, in fact, has cited many of these well-used references in the chapters of this book and so not only does the work provide an updated perspective within the context of modern design complexity, it also ties many of these works together in a self consistent and clearly articulated framework.

The book is likely to prove most valuable, however, to the design manager. The book, while providing insight on the individual design steps is organized in a way to provide a complete framework through which the design process can be executed.

For the design manager it is essentially a handbook to determine whether a sufficient process for IC design is in place in their organization. More than that, the design manager can use the book to keep an “inventory” of the various skill sets and competencies that their design organization must keep current.

Finally, the book can and should be appreciated by, like me, the business manager within a semiconductor organization. For the business manager, the physical design process may not be focused on as the value added step. From the business managers perspective, value is often perceived to be created in differentiated technology, market selection or product definition. Once these steps have been completed the business manager, like the public, often takes the physical implementation for granted.

I would encourage, however, the business manager to at least peruse Mr. Golshan’s book and become familiar and appreciate the significant process that now forms physical design. This process must be fully understood and carefully executed within the semiconductor business organization or all that supposed value added work that preceded “chip layout and verification” could be for naught.

Matt Rhodes
CEO, Teranetics, Inc.

Acknowledgments

I would like to express my gratitude to a number of individuals who contributed their time and effort towards this manuscript. From Conexant Systems Inc., I especially thank Eric Tan for his technical advice on physical design, Mark Tennyson for sharing his I/O circuit design expertise in the Input-Output pads section, and Himanshu Bhatnagar for his technical recommendations.

In addition, I thank Badih El-Kareh from PIYE, Professor Ping Gui from Southern Methodist University, Scott Peterson from LSI Logic Inc. and Professor Sachin Sapatnekar from the University of Minnesota for reviewing the manuscript and providing and constructive recommendations.

A special thanks to Maury Golshan and Ian Wilson for their help in editing this manuscript. They spent a considerable amount of time and effort in proofreading and revising which significantly improved the clarity and consistency of this manuscript. Without their dedication, it would be almost impossible to have completed the task.

Last, but not least, I would also like to thank Anil Mankar (Vice President of VLSI Engineering, Conexant Systems Inc.) who gave me encouragement and moral support throughout the process.

Khosrow Golshan

Chapter 1

LIBRARIES

"Good order is the foundation of all things." Edmund Burke

Various types of data sets or libraries are required for the physical design of an Application Specific Integrated Circuit (ASIC). Libraries are collections of the physical layout, abstract views, timing models, simulation or functional models, and transistor level circuit descriptions.

As such, libraries are considered one of the most critical parts of the ASIC physical design, and the accuracy of these libraries and their associated views and models has a great impact on the success of the final fabricated ASIC design.

Standard cell libraries and Input-Output pads are typically used in ASIC design. In addition, memories and custom libraries may be used. Memories, such as Random Access Memory (RAM) or Read Only Memory (ROM) and their appropriate layout, abstract, timing and simulation views, are usually obtained from memory compilers.

Custom libraries, which are also referred to as Intellectual Property (IP) libraries, are collections of manually crafted analog function layouts such as Phase Lock Loop (PLL), Analog to Digital Converter (ADC), Digital to Analog Converter (DAC), and Voltage Regulator (VR).

Because standard cells and Input-Out pads are the most basic building blocks of ASIC physical design, a general overview of the physical specification and timing generation of standard cells and Input-Out pads is the focus of this chapter. The techniques that are outlined for standard cells can be extended to memory and IP physical design as well.

1.1 Standard Cells

A standard cell is a specific design for each gate in the library. Special care needs to be taken during the physical design of such libraries in order to obtain optimal ASIC die size and performance. With the advancements in the fabrication process and the increasing complexity of logic designs, the overall area of ASIC designs is becoming more dominated by the routing area rather than by the total area of transistors used. Therefore, it is necessary that the routing area be minimized rather than minimizing the area consumed by standard cells. Since the majority of ASIC routing is performed automatically, it is important to design standard cell sizes so that they are well-suited to the place-and-route tools being used.

The basic step in the physical design of standard cells begins with horizontal and vertical wire track determination. Wire tracks are used to guide place-and-route tools to perform interconnection between standard cells. Manufacturing guidelines such as width and spacing of the first two conducting layers (e.g. metal one and metal two) are used to set proper wire track spacing. Commonly, there are three ways to compute wire track spacing using center-to-center spacing – line-to-line ($d1$), Via-to-line ($d2$), and Via-to-Via ($d3$):

$$d1 = 1/2w + s + 1/2w \tag{1.1.1}$$

$$d2 = 1/2w + s + Via_{overlap} + 1/2Via_{size} \tag{1.1.2}$$

$$d3 = 1/2Via_{size} + Via_{overlap} + s + Via_{overlap} + 1/2Via_{size} \tag{1.1.3}$$

The relationship between these equations is

$$d3 > d2 > d1\,. \qquad (1.1.4)$$

In comparison, line-to-line is the most aggressive for conducting layer compaction. However, line-to-line center spacing will not optimize the overall routing as Via-to-Via and Via-to-line are not considered. Via-to-Via center spacing meets all line-to-line and Via-to-line center requirements, but the overall routing will not be optimum due to large spacing between the conducting layers.

In practice, Via-to-line has shown to be the most optimum. Via-to-line meets all conducting layers spacing rules and exhibits the most compact overall routing. An example of each wire track style is shown in Figure 1-1.

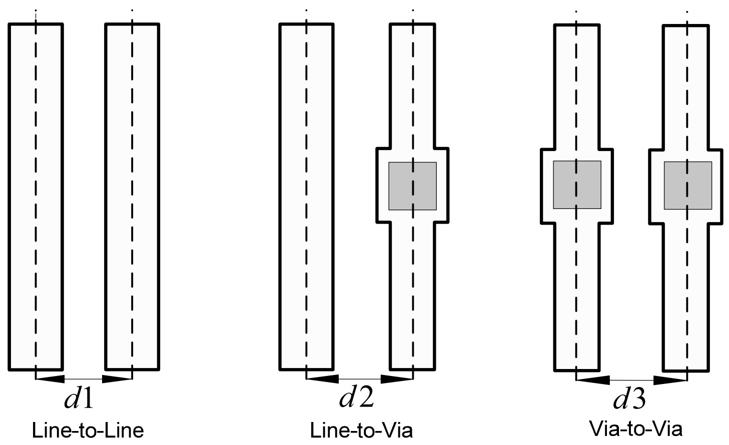


Figure 1-1 Wire Track Center-to-Center Spacing

Most place-and-route tools require that the height and width of a standard cell be an integer multiple of the vertical and horizontal wire track. The height of the standard cell is the same throughout the library, but their widths vary according to their logical functions and drive strengths.

A typical standard cell for the Complementary Metal Oxide Semiconductor (CMOS) process is composed of a row of NMOS (N-type transistors) with channel width W_n, and a row of PMOS (P-type transistors) with channel width W_p separated by the distance of the P and N diffusion (or active) area.

The P and N diffusion area spacing, the channel width of PMOS and NMOS transistors, and the width of power (VDD) and ground (VSS) buses are the key parameters in determining the height of standard cells. Figure 1-2 shows a generalized standard cell height concept.

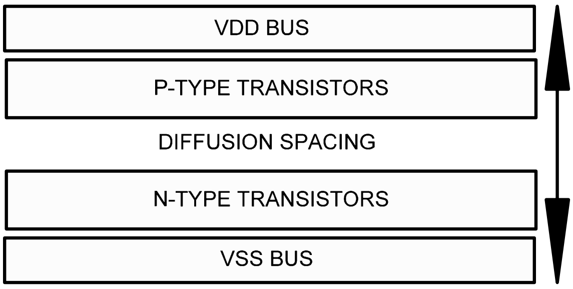


Figure 1-2 Generalized Standard Cell Height

Once vertical and horizontal wire tracks, as well as the height of the standard cell is determined, this information is used to create a wire track template for use during standard cell layout.

Overlaying the wire track template on a standard cell layout during the physical design as a layout guideline insures that the actual physical layout of standard cells and their physical port locations will meet place-and-route tool routing requirements. Figure 1-3 shows a wire track mesh marked by Horizontal Wire Track and Vertical Wire Track.

As mentioned earlier, one of the key parameters in determining the standard cell height is the width of power and ground buses which traverse the top and bottom of the standard cells. If the power (VDD) and ground (VSS) bus layers are the same as the first horizontal routing layer, a limitation on standard cell heights is imposed. This is because the width of VDD and VSS buses has to be wide enough to provide proper current flow capability and this increase in the power and ground line width will affect the standard cell height.

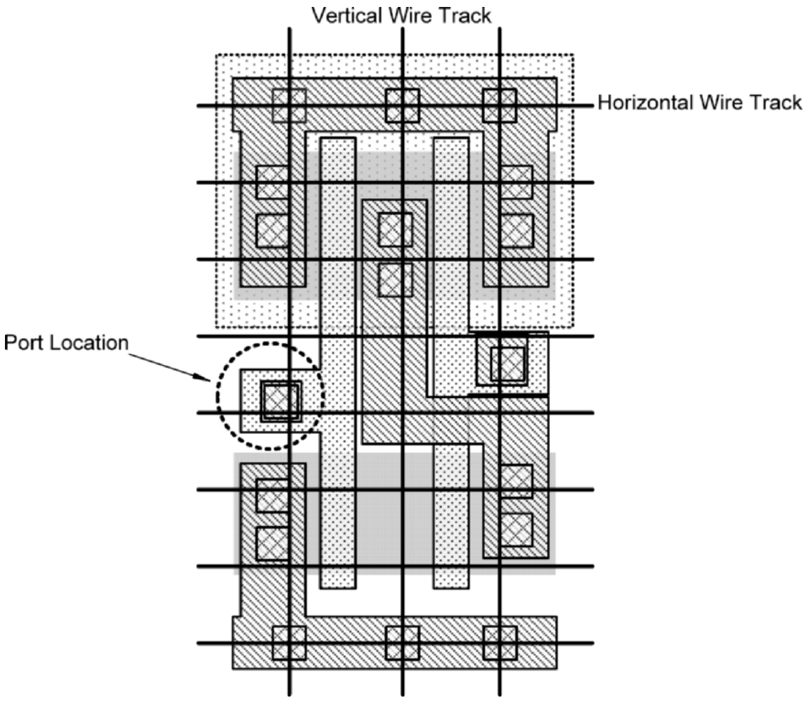


Figure 1-3 Wire Track Mesh

For example, in an NWELL process, the power (VDD) bus must contact the NWELL and the ground (VSS) bus must contact the substrate. In addition, the VDD and VSS should be fully connected or strapped (i.e. using multiple contact cuts). The main advantage of fully strapped VDD and VSS buses to the NWELL and substrate is resistance reduction. This resistance reduction enhances standard cell immunity to internal latch-up phenomena. Because the power and ground minimum widths are dictated by the contact size and overlapping contact of the first conducting layer (e.g. *metal one*), the width of both VDD and VSS buses needs to be enlarged enough to avoid physical design rule violations.

With more metal layers available in today's silicon processes, using an alternate routing approach, such as first metal traverse vertically and second metal horizontally, would be advantageous in standard cell physical design. Using this method, the second layer can be used for power and ground routing over internal standard cell transistors. This technique may provide much better results with respect to area, performance, and power consumption for multimillion gate ASIC designs that require significant amounts of routing resources for power to prevent voltage drop across the chip. To handle a variety of power requirements, the standard cell power and ground buses can vary in widths depending on power requirements without modifying the standard cell heights.

In standard cell layout, it is preferable to use the first conducting layer, such as metal one, as much as possible to make internal connections of NMOS and PMOS transistors within standard. If there is a need to use other conducting layers, such as metal two, use of such layers must be kept to a minimum. This greatly influences the ASIC top level routing. In addition, all internal node capacitance need to be kept at a minimum with the most capacitive nodes close to the VDD and VSS buses in order to reduce body effect impact. The body effect is a dynamic problem that changes the transistor (MOSFET) threshold voltage when the source to well (or body) bias changes.

Another key factor in standard cell physical layout is the location of input and output ports. It is desired to use the first routing (e.g. metal one) layer for standard cell ports, or pins, and place them where the horizontal and vertical wire tracks cross as shown in Figure 1-3. This allows the place-and-route tools to access the ports from both X and Y directions. This is known as port accessibility; it improves execution time during the routing step and produces better quality routing with respect to the overall physical design rule violations.

During standard cell library development, establishing geometrical regularity among the layout of all the standard cells of the same type has two advantages. First, it allows the use of compaction software in order to further reduce the area of standard cells while enabling migration of the standard cell library to another process node (e.g. migrating standard cell library from one design rules to another one) with ease. Secondly, and most importantly, establishing geometrical regularity leads to common electrical characteristics between the standard cells. This electrical uniformity will be useful when dealing with one of the most common limitations and challenges in physical cell design. In addition, it plays an important role in deciding the largest PMOS and NMOS transistor channel widths within the library.

Once channel width and ratio of the PMOS and the NMOS transistors is determined, the standard cell layout can be designed by using a single column of NMOS and PMOS transistors aligned at common connection distances by active area. It is desirable to layout all simple transistors as unbroken columns.

The polysilicon gates need to be ordered to allow maximum connection between them by sharing the source and drain area to form transistors. It is electrically advantageous to place the NMOS transistors as close as possible to the VSS bus and PMOS transistors to the VDD bus.

In connecting the source and drain of transistors to the VDD and VSS buses, single contact cut should be minimized. Minimizing single contact cut and using multiple source and drain contact cut connections will reduce source-drain resistance and enhance electrical performance of the standard cell.

For the logical gates that consist of transistors in series, such as AND logic, the smallest transistor should be placed close to the output, and the size of transistors needs to be increased as they approach the ground (VSS) or power (VDD) supplies. This will improve the overall performance [1], but it has an area penalty.

In the case of complex cells, such as flip-flop or Boolean functions, polysilicon connections can be used for non-critical signals. It is extremely important to avoid PWELL or NWELL routing. There are two main problems with PWELL or NWELL routing – the material is highly resistive, and often its parasitic such as resistance cannot be extracted. This influences the gate characterization accuracy in comparison to the actual silicon.

The current submicron CMOS process is very complex in nature. In addition, it is very difficult to visualize all the mask levels and manufacturing design rules that are used during actual fabrication of an ASIC design. However, in designing standard cell layouts for the CMOS process, a minimal set of design rules shown in Figure 1-4, are adequate. The reason for this minimal set is that most of today’s standard cell are using up to *metal two* in their designs, however, using higher layer such a *metal three* can create a routing obstruction that could lead to local routing congestion during routing.

In the early days of standard cell development, the area of the standard cell was of more concern, thus the physical design objective was to design the standard cell to be as small as possible. This was mainly dominated by the fact the polysilicon feature (two micrometer line width) was larger than the metallization line width. Electrical parameters, such as power and noise, were not an important factor and they did not have a large impact on overall design performance.

Minimum width of a nwell
Minimum space between two nwell of the same potential
Minimum area of nwell

Minimum width of diffusion to define NMOS/PMOS width
Minimum width of diffusion for interconnect
Minimum space between diffusion regions

Minimum overlap of nwell over P+ region inside nwell
Minimum clearance from nwell to P+ region outside nwell
Minimum area of diffusion

Minimum width of a poly for channel length of MOS transistor
Minimum width of a poly for interconnect
Minimum clearance from diffusion to poly on field oxide
Minimum space between two poly on field oxide area
Minimum poly extend into field oxide (end-cap)
Minimum poly area

Contact size
Contact spacing
Minimum space of contact on diffusion to poly
Minimum space of contact on poly to diffusion

Maximum width of metal1
Minimum extension of metal1 over contact
Minimum metal1 space
Minimum metal1 area

Figure 1-4 Minimum Design Rule Set

With today's advanced processes, where the polysilicon line width is becoming very narrow, the overall ASIC chip area is dominated by the level of metallization. Hence, performance is impacted by noise injection and power consumption. It is important to make sure that the channel width of PMOS and NMOS transistors is large enough to account for power dissipation, overall noise immunity, and their ratio is set properly to provide optimum performance. Therefore, one should note that the style of standard cell layout involves optimizing the transistors with respect to noise immunity, power dissipation, and performance, rather than optimizing the transistors to achieve a smaller area.

Another consideration in the physical design of standard cells is the size of the output stage transistors that determine the drive capability of the external capacitive loads. Each gate type needs to have multiple drive strengths. These drive strengths must be uniform across all gate types and need to increase monotonically with the output capacitance.

For process nodes with gate lengths of 130nm and below, classical mask generation for patterning the critical dimensions, such as polysilicon gate and first conducting layer (e.g. metal one), can no longer produce correct results. This is mainly because during wafer printing of very narrow width geometries, the incident light sources will interfere with each other and will cause incorrect exposure. Figure 1-5 shows the basic concept of such a wafer printing problem.

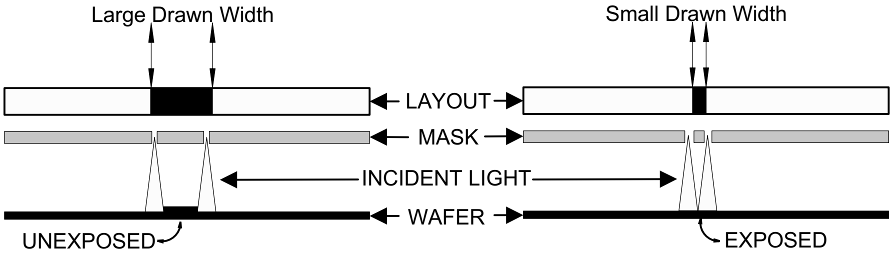


Figure 1-5 Illustration of Problematic Wafer Printing

In order to solve deep submicron wafer printing problems, many of today's semiconductor foundries are utilizing a new way of generating masks called

Phase Shift Mask (PSM). Basically, PSM technology creates two different incident light sources–one with zero degrees and one with a 180-degree–phase shift to print narrow width geometries. Figure 1-6 shows the basic concept of PSM technology.

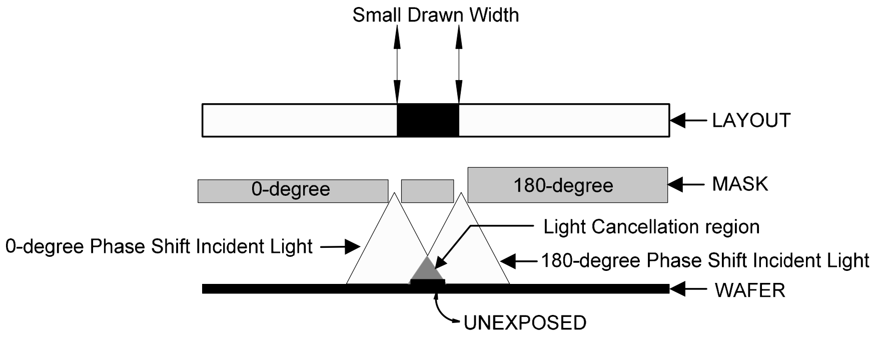

Small Drawn Width
LAYOUT
0-degree
180-degree
MASK
Light Cancellation region
0-degree Phase Shift Incident Light
180-degree Phase Shift Incident Light
WAFER
UNEXPOSED

Figure 1-6 Basic Concept of PSM Technology

Although PSM exposure technology offers significant advantages in the photolithography process for deep submicron processes, it has its own problem – phase conflict. This conflict arises from unintentionally joining two regions that transmit light with opposite phases. Most often this destructive interference of these two lights creates unwanted features on the wafer.

To solve such a problem, there are many variations of PSM available today. Among these alternatives, Complementary Phase Shift Mask (CPSM) technology has demonstrated the power to improve exposure windows significantly, and has been chosen by major foundries for their deep sub-wavelength lithography process [2].

To prevent any wafer printing issues due to the lithographical process in the final inspection of standard cell layout circuits, designers should use viewing software such as the Optical Process Correction (OPC) simulator that presents physical shapes as they will look on the mask and on the wafer. This allows detection of some device manufacturing related problems early in the overall process, where correction is more easily done.

From the circuit design point of view, the objective is to select PMOS and NMOS transistors channel widths W_p and W_n and their corresponding ratio in order to provide maximum performance for a given CMOS process. Proper selection of W_p and W_n has a great impact on parameters such as power dissipation, noise immunity, propagation delay, and the area of standard cells.

One approach to optimizing electrical characteristics and minimizing the area of standard cells is to use preliminary physical layouts of basic functions such as inverter, NAND, and NOR gates to compute the average propagation delay and relate the computed propagation delay to various ratios of the PMOS and NMOS transistor (W_P / W_n) channel widths.

In order to achieve optimal average propagation delay (t_p), accurately defining the channel widths of these transistors and their ratio and obtaining properly balanced low-to-high (t_{pLH}) and high-to-low (t_{pHL}) propagation delay for both NMOS and PMOS transistors is required.

The average propagation delay (t_p) of standard cells is defined as

$$t_p = \frac{t_{pLH} + t_{pHL}}{2}. \tag{1.1.5}$$

In an ideal situation for equally sized PMOS and NMOS transistors, the value of rise time (t_r) and fall time (t_f) are approximated by

$$t_r \approx 3t_{pLH} \tag{1.1.6}$$

$$t_f \approx 3t_{pHL}. \tag{1.1.7}$$

The values of the t_{pLH} and t_{pHL} parameters are considered intrinsic values and are directly related to the topology of standard cells. These parameters determine the system response to an input function and are expressed in time units.

The values of t_r and t_f are counted as extrinsic values and are dependent on the external loads. These parameters determine the slope of system output response in time and are related to the standard cell drive strength capability (the time to charge and discharge unit load capacitance).

1.2 Transistor Sizing

Transistor sizing or selection of the proper W_P / W_n ratio has a direct effect on the intrinsic and extrinsic behavior of any given standard cell. From the electrical characteristic point of view, switching speed of a CMOS device is governed by the time taken to charge and discharge a capacitive load (C_l). Three main timing parameters are associated with CMOS devices – rise time, fall time, and propagation delay. Most often, in discussion with regard to these parameters, the system response of an inverter is used. Figure 1-7 shows the system response of inverter logic.

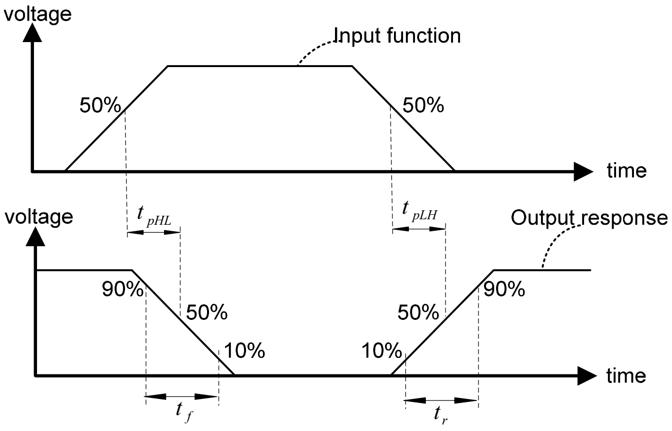


Figure 1-7 System Response of an Inverter

Referring to Figure 1-7, rise time is the time that it takes to charge the output capacitive load. Fall time is the time that it takes for the output capacitive load to discharge. The rise and fall time are usually measured from 10% to 90% and from 90% to 10% of the steady state value of a waveform.

Propagation delay is the time difference between approximately 50% of the input transition and approximately 50% of the output transition. If the input waveform changes from zero to supply voltage (VDD) or from supply voltage (VDD) to zero value, then low-to-high and high-to-low propagation delays can be expressed as [3]

$$t_{pLH} = \frac{C_l VDD}{I_p} = \frac{C_l VDD}{\beta_p (VDD - |V_{tp}|)^2} \tag{1.2.1}$$

and

$$t_{pHL} = \frac{C_l VDD}{I_n} = \frac{C_l VDD}{\beta_n (VDD - V_{tn})^2}, \tag{1.2.2}$$

where the parameters β_n, V_{tn}, β_p, V_{tp} are the gain factors and threshold voltages for NMOS and PMOS transistors.

It is clear from Equations 1.2.1 and 1.2.2, that to improve the propagation delay of a given standard cell, one could

- Increase supply voltage,
- Reduce threshold voltage,
- Increase transistors gain factors, or
- Reduce the load capacitance.

Reducing load capacitance or increasing the supply voltage are external to the standard cell and will be discussed in Chapter 3.

Reduction of the threshold voltage depends on the semiconductor foundry offering (i.e. multiple threshold voltage process) and is part of standard cell characterization rather than the standard cell physical design. Therefore, the only parameter that is available to the circuit designer is to increase the gain factor.

The equations which relate PMOS and NMOS transistor channel width W and channel length L to the gain factor β are:

$$\beta_p = (\mu_p \varepsilon / t_{ox})(W_p / L) \qquad (1.2.3)$$

$$\beta_n = (\mu_n \varepsilon / t_{ox})(W_n / L) . \qquad (1.2.4)$$

The parameters μ_p and μ_n are carrier mobilities associated with PMOS and NMOS transistors and are related as

$$\mu_n \approx 2\mu_p . \qquad (1.2.5)$$

The parameters ε and t_{ox} are the transistors' gate oxide permittivity and thickness.

In the CMOS process, the term $(\mu\varepsilon / t_{ox})$ is frequently referred to as process gain.

Since the channel length L of transistors and process gain are fixed for a given process, to determine a desired rise and fall time of the transistors, the channel width of the transistors needs to be chosen appropriately.

In W_P / W_n ratio determination, it is desired to set

$$\beta_n = k\beta_p . \qquad (1.2.6)$$

Substituting Equations 1.2.3, 1.2.4 and 1.2.5 in the Equation 1.2.6, yields

$$(2\mu_p \varepsilon / t_{ox})(W_n / L) = k(\mu_p \varepsilon / t_{ox})(W_p / L) \qquad (1.2.7)$$

$$(W_p / W_n) = 2 / k . \qquad (1.2.8)$$

In the ideal situation, *k* is equal to 1. This means that for a CMOS inverter to charge and discharge capacitive loads in the same amount of time, the channel width of the PMOS transistor must be twice as large as the channel width of the NMOS transistor. In practice, however, the value of *k* can be more than 1.

Although increasing the value of W_p / W_n reduces the cell propagation delay, it also increases the active area capacitance and gate capacitance. This increase in capacitance adversely affects the gate speed. Therefore, circuit designers must make a tradeoff in determining how large the transistors should be such that their propagation delays are optimal (i.e. optimal *k* value).

Considering an inverter with no external load (i.e. no wire capacitance at the output), then the output load is approximately equal to

$$C_l = C_n + C_p \tag{1.2.9}$$

where C_n and C_p are P and N source and drain active area capacitance.

If the PMOS transistors are made k times larger than the NMOS transistors ($W_p = kW_n$), then the output load can be expressed as

$$C_l = C_n + kC_n = (1+k)C_n . \tag{1.2.10}$$

Substituting Equations 1.2.1, 1.2.2, 1.2.3 and 1.2.4 in Equation 1.1.5, and assuming that the supply voltage is much larger than the threshold voltage, then the propagation delay is approximately equal to

$$t_p = \frac{C_n(1+k)}{2VDD}\left[\frac{t_{ox}L}{k\mu_p \varepsilon W_n} + \frac{t_{ox}L}{\mu_n \varepsilon W_n}\right]. \tag{1.2.11}$$

To find the optimal value of k , set

$$\frac{\partial t_p}{\partial k} = 0 \tag{1.2.12}$$

and solve for k:

$$k_{opt} = \sqrt{\frac{\mu_n}{\mu_p}}\ . \tag{1.2.13}$$

Furthermore, if an inverter with external loads includes the wire capacitance and other standard cell input gate capacitance, the optimal value of k can be computed based on the allowable maximum load capacitance. This maximum load can also be used as a constraint in the standard cell library during logic and physical synthesis to prevent overloading the cell output.

Based on the maximum capacitance $C_{\max}$, then revaluating Equation 1.2.11, yields

$$k_{opt} = \sqrt{\frac{(C_n + C_{\max})\mu_n}{C_n \mu_p}} = \sqrt{(1 + \frac{C_{\max}}{C_n})\frac{\mu_n}{\mu_p}}\ . \tag{1.2.14}$$

If $C_{\max} >> C_n$, then Equation 1.2.12 can be approximated as

$$k_{opt} = \sqrt{\left(\frac{C_{\max}}{C_n}\right)\left(\frac{\mu_n}{\mu_p}\right)}\ . \tag{1.2.15}$$

1.3 Input-Output Pads

One of the most critical elements of ASIC design is the design of the input and output pads. Input-Output (I/O) pad structures require the deepest circuit design expertise and most detailed understanding of the process that is being used to fabricate an ASIC design.

I/O pad circuits translate the signal levels used in the ASIC core to the signal levels used outside the ASIC. Additionally, the I/O pad circuits clamp signals to the power and ground rails to limit the voltage at the external connection to the ASIC I/O pad. This clamping reduces signal overshoot and prevents damage from Electrostatic Discharge (ESD).

General-purpose I/O (GPIO) pads are simple three-state (zero, one, high-impedance) output buffers combined with input receivers and PAD opening (i.e. area of a pad that is used to attach bond wire for external connection), as shown in Figure 1-8.

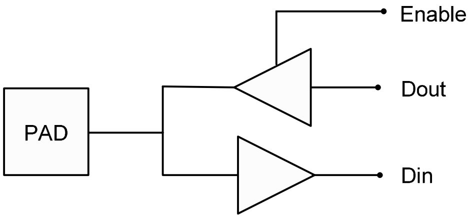


Figure 1-8 General Purpose Input and Output Pads

There are three types of ASIC input and output pads:

- General-purpose input and output pads
- Power and ground pads
- Special purpose input and output pads

Many options are available with a general-purpose I/O pad such as:

- The output buffer voltage may be 3.3V, 2.5V, 5V-tolerant, etc.
- The output buffer current drive capability may be 2mA, 4mA, etc.
- The output buffer may be designed to limit the signal slew rate
- The output buffer may be omitted (input pad)
- The pad may have a pull-up resistor or pull-down resistor
- The input receiver detect level may be TTL 1.4V, CMOS 0.5V, etc.
- The input receiver may have hysteresis (Schmitt trigger)
- The input receiver may be omitted (output pad)

Providing all possible combinations of various options can give rise to a very large number of general purpose I/O pad types. The number of I/O pad types can be reduced by making some of the options selectable and controlled by the ASIC core logic. This reduces the number of I/O pads that need to be modeled, but increases the model complexity.

Power and ground pads provide connections to the various ASIC power and ground busses. The metal connections from the pad to the power or ground bus within the power and ground pad or to the ASIC core are made as wide as possible and on as many metal layers as practical to minimize their resistance and maximize the current carrying capacity.

Because no other circuitry is required, the area in power and ground pads is often used for special ESD clamping circuitry that is associated only with power busses and not with any particular signal pin. Figure 1-9 shows a basic I/O and core power pad with built-in ESD protection circuitry.

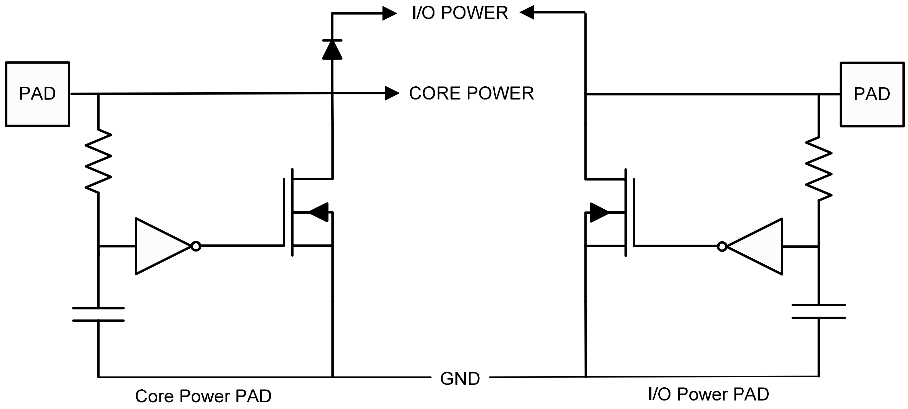


Figure 1-9 Basic Core and I/O Power PAD

Special purpose I/O pads encompass cells with unusual or especially stringent requirements, or cells that are primarily analog. Some examples are crystal oscillator cells, Universal Serial Bus (USB) transceivers, Peripheral Component Interconnect (PCI), Low-Voltage Differential Signaling (LVDS), and isolated analog signal, power, and ground cells. The layout style of the special purpose I/O pad is usually very similar to the general-purpose I/O pad, but the circuit design is more critical.

The output buffer portion of an I/O pad consists of a pre-driver and a driver. The pre-driver defines the pad functionality and the driver is simply a large inverter connected to the PAD.

The pre-driver may contain level shifter circuits to convert the signal voltage from the ASIC core voltage to the I/O pad voltage. Figure 1-10 shows typical I/O pad logic.

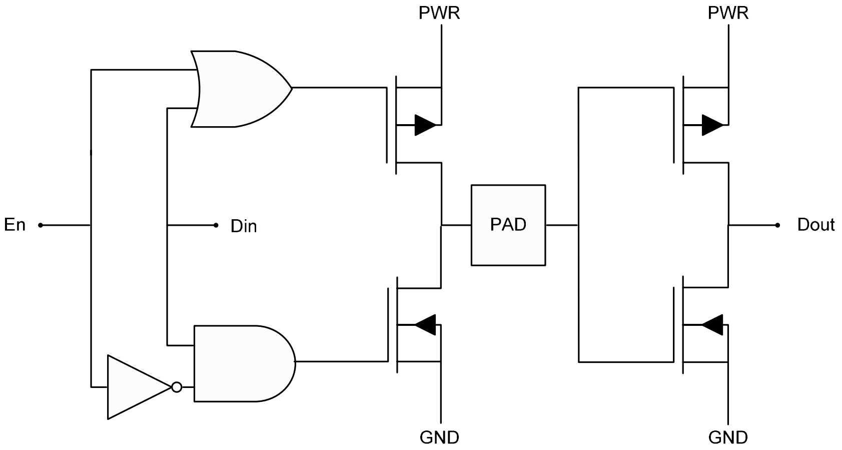


Figure 1-10 Typical Input-Output Pad Logic

In this figure, the data lines from ASIC core logic are connected to **Dout**, data lines to ASIC core logic are connected to **Din**, and control (i.e. input or output direction) is by enable line **En**.

During the design of I/O pads one needs to consider the trade-off between the propagation delay of the pre-driver section and noise sensitivity. High-speed pre-drivers can cause sharp current spikes during operation. These current spikes can inject noise into the power supply especially when a large number of input and output signals are switching simultaneously. This may require a slowing down of the pre-driver by means of slew rate control.

Another potential source of current spikes is if the driver NMOS and PMOS transistors are briefly conducting at the same time. This can be prevented by adjusting the timing of the transistor gate terminals to make sure each transistor is off before the other transistor turns on. Because the supplies connected to the driver transistors are subject to large current spikes and the resulting noise, it is common to isolate these supplies from all the other supplies on the ASIC core.

Typically, the pre-driver is designed to meet both the slowest and fastest ASIC design conditions. In the slow case, if the pre-driver is too slow, then there will be unwanted data dependency jittering.

This data dependency jittering occurs when the pre-driver output voltage does not reach the full power supply voltage level within the data cycle time. On the other hand, in the fast condition, the pre-driver has high current consumption and sharp transients, which lead to simultaneous switching noise. Thus, it is imperative that these two conditions, or constraints, (i.e. fast and slow) be balanced for proper I/O pad operation.

While the circuit design of general-purpose I/O pads is straightforward, the layout is complicated by the need to handle the large currents. Large currents occur during output transitions when external capacitances are rapidly charged and discharged.

Large currents also occur when there is signal overshoot and a clamping diode becomes forward biased. In addition, large currents occur when a high voltage is rapidly discharged through the ESD protection devices.

The various current paths within I/O pads should have metal widths and number of via cuts appropriate for the current based on maximum current densities for acceptable product reliability (i.e. product lifetime).

The overall resistance of the current paths of each I/O pad should be kept low to minimize voltage drops. Since there are at present no automated design rules verification available that will verify I/O pad current paths, careful planning, and good layout practices and manual inspections are necessary.

The layouts of NMOS and PMOS transistors and diodes that are directly connected to a PAD have special design rules. Larger rules for contact-to-gate spacing, PWELL or NWELL overlap of contact, and gate width/length are used. In addition, precise methods of inserting resistance between the PAD and transistor drains as well as general guidelines for placing body ties and latch-up guard rings are used.

Similar to the current path verification, it should be noted that some rules and guidelines such as electrical rules cannot be checked by means of physical verification. Therefore, careful simulation and proper layout style are necessary (e.g. layout dependency of ESC structure). This means that in I/O pad design, simply having a layout that is physically verified does not guarantee success.

Conventionally, I/O pads have a constant length and width. The minimum PAD pitch dictates the I/O pad widths. The pitch is the spacing from a point

on one PAD to the same point on an adjacent PAD.
In practice, the pad pitch and the pad opening size is set by the mechanical limitation of packaging tools and the probe diameter of test equipment used to test an ASIC at wafer level.

Most commonly, the I/O pads are placed at the ASIC core edge either in-line in a single row or staggered in two rows. In both cases, the I/O pad's circuits are placed side by side in a single row. The PAD itself may be part of the I/O pad cell or it may be a separate cell.

For an ASIC that is pad-limited, the pad pitch should be as small as possible to reduce the die size. An ASIC that is core-limited can use a larger pad pitch, which usually allows a shorter I/O pad height.

The I/O pad heights are determined by the size of the NMOS and PMOS transistors and their associated ESD protection devices, the area required for pre-driver and input receiver circuitry, the width of power and ground busses, body ties, guard rings, and other spacing required for latch-up immunity.

The I/O pad size is determined by the cell with the largest circuitry that is to be present in a standard sized I/O pad. It is possible to make some especially large I/O pads with a larger-than-standard I/O pad width if needed.

One important part of any I/O pad design is the design of ESD protection circuits. Figure 1-11 shows a basic ESD protection circuit for a general-purpose I/O pad.

The two primary clamping diodes **D1** and **D2** will turn on if the PAD voltage rises above the power level or drops below the ground level. The resistor **R1** and capacitance at node **N1** create an RC time constant to slow the high-speed ESD spike to the gates of the PMOS and NMOS transistors **M3** and **M4**. This allows the clamping devices to have time to turn on and prevent damage to the gates of **M3** and **M4**. The resistor **R1** layout may use non-silicide polysilicon; PWELL or NWELL will have a value ranging from about a hundred to a thousand ohms. This resistor and associated RC delay create design challenges when dealing with high-speed input pads.

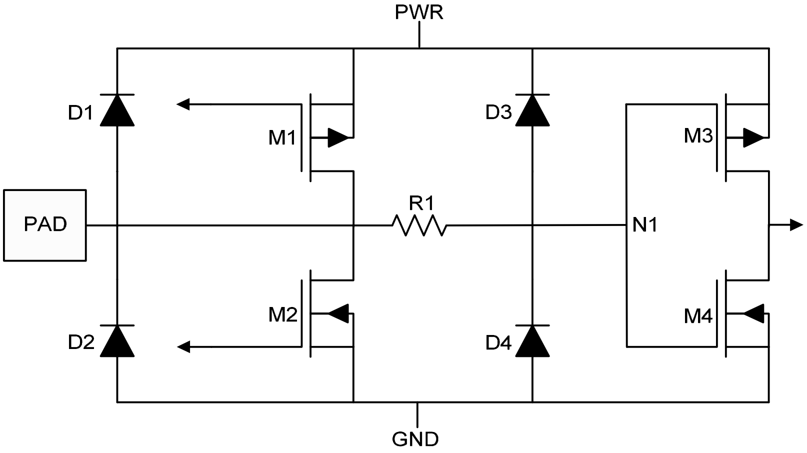


Figure 1-11 Basic ESD Protection Circuit

Very often, the diodes **D1** and **D2** are not explicitly drawn but are still present as the drain to body junctions of transistors **M1** and **M2**. The transistors **M1** and **M2** may be the primary ESD clamping devices. In this case, the drain, body, and source of the transistors form a bipolar transistor known as a snapback device. When the voltage and current of the snapback device reaches a trigger voltage and current, the voltage will snap to a lower level and the resistance of the device becomes very low. This condition can easily damage the transistor when the snapback is localized to only one portion of the entire channel region, otherwise known as a hot spot. For this reason, the resistance from the pad to all portions of the channel region must be as uniform as possible to ensure that the entire channel enters the snapback mode simultaneously. The transistor layouts use multiple fingers with with identical metal widths, contact-to-gate spacing, etc.

In addition to the ESD protection circuitry within the I/O pad, there are other clamping devices coupling the various power and ground busses to each other. These clamping devices may be simple diodes when connecting two supplies having the same voltage – such as digital ground to an isolated analog ground.

For clamps between power and ground, as shown in Figure 1-12, a single transistor may be used as a snapback device or it may be used for more complex circuits such as diode stacks (for low voltage supplies) or large transistors with transient detector circuits.

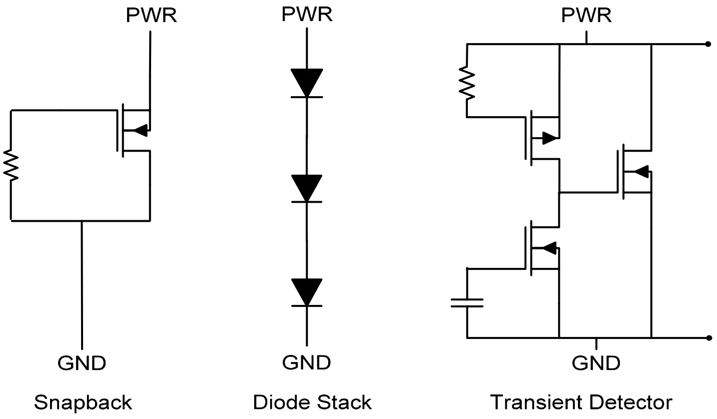


Figure 1-12 Power Supply ESD

ESD events can lead to the failure of poorly designed I/O pads of an ASIC. This failure mechanism primarily is thermally driven and is mainly caused by three physical factors – human body, charge device, and/or mechanical handling of an ASIC device. These physical factors are modeled as [4]:

- The Human Body Model (HBM) that captures the effect of charged human body into an ASIC
- The Charge Device Model (CDM) that captures the effect of self-charging and then self-discharging

- The Machine Model (MM) that captures the effect of charging due to machine handling

Future advancements in process technologies that will produce thinner oxide layers, narrower line widths on conducting layers, and shallower junction transistors will make ASIC devices more susceptible to ESD damage.

Another design aspect of I/O pads is their immunity to the latch-up problem. This problem is more severe in I/O pads than in standard cells. This is because signal overshoot causes the transistor drain-body diode to become forward biased, resulting in current flow through the substrate.

The latch-up phenomenon is well understood and is inherent to bulk CMOS processes. The result of this effect is the shorting between power and ground lines that can lead to ASIC self-destruction and system power failure.

One of the most common practices during I/O pad circuit design is to separate NMOS and PMOS transistors, and encircle them with the appropriate well tie and guard ring.

Figure 1-13 shows the N+ and P+ well ties and guard rings that surround the PMOS and NMOS transistors. The output driver transistors that are directly connected to the PAD are each isolated within their own rings.

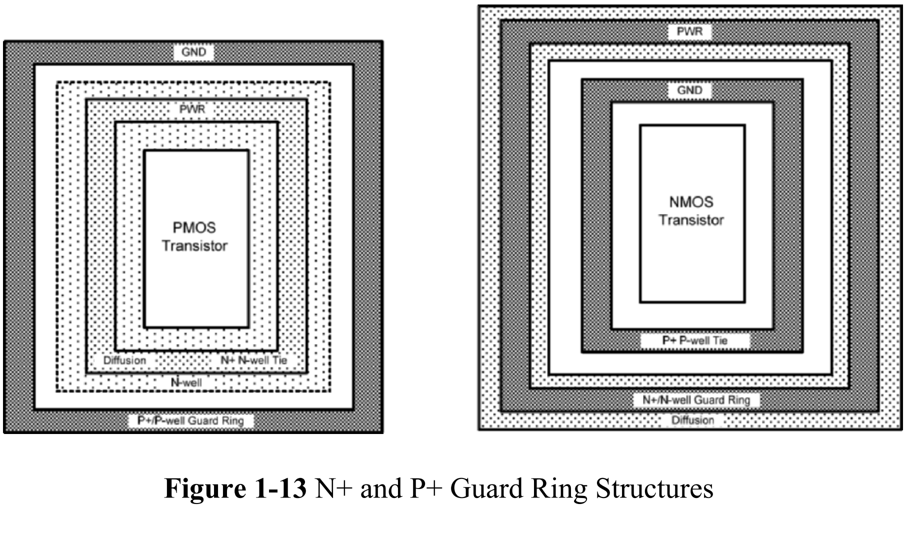


Figure 1-13 N+ and P+ Guard Ring Structures

The pre-driver and input receiver transistors are grouped by PMOS and NMOS, and each type is isolated from the other as well as the output driver transistors. If an NWELL is tied to a signal instead of to power (such as an NWELL resistor) then it should be treated as N-type diffusion during physical verification (e.g. latch-up rule verification).

1.4 Library Characterization

Library characterization refers to the generation of a timing model for standard cells, I/O pads, or any custom library. Most of the time, these timing models are generated by means of automation. Software for this purpose is usually available from EDA companies that are specialized in library characterization.

Characterization tools take user-defined input waveforms and perform various transistor-level circuit simulations under several conditions to generate circuit responses for each element in the library. The generated circuit responses will undergo some data processing to produce timing models for importing to logic synthesis, place-and-route, and logic simulation tools.

During timing model generation, two main components are the subject of interest – propagation and transition delay. Using these two parameters, then the total cell delay can be computed as:

$$d_{LH} = t_{pLH} + t_r \tag{1.4.1}$$

$$d_{HL} = t_{pHL} + t_f \, . \tag{1.4.2}$$

To compute total cell delay d_{LH} and d_{HL} , several methods can be chosen. These are:

- Linear or scalar delay model
- Nonlinear delay model

- Polynomial delay model
- Current source delay model

In the linear or scalar model, the total cell delay is a function only of load capacitance as shown in Figure 1-14.

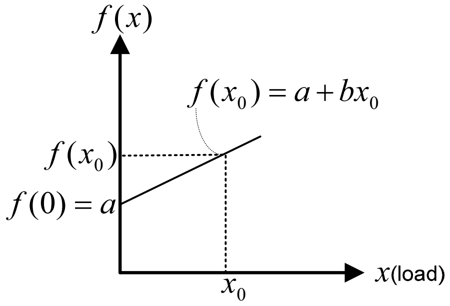


Figure 1-14 Linear Delay Model Graph

In this method, the propagation delay of low-to-high and high-to-low are fixed, and the rise and fall time are linear functions of the output capacitance. This functionality can be expressed as

$$f(x) = a + bx, \tag{1.4.3}$$

where $f(x)$ is the total cell delay as the function of output capacitance load x. The $f(0) = a$ parameter is a representation of total cell delay with zero loads. The slope of the line b is considered as the cell drive strength and the product term bx can be viewed as rise time t_r or fall time t_f.

In a nonlinear delay model, the effect of the input transition of the waveform is included in the cell delay calculation. There are two methods used to express this type of delay model – cell delay and propagation delay.

In the cell delay method, the values of t_{pLH}, t_{pHL}, t_r and t_f are specified separately. For instance, the total cell delay can be expressed as the delay from the input 50% threshold to the output 50% threshold, and the transition delay as the slew time from 10% to 90%.

In this method, the values of t_{pLH} and t_{pHL} represent the entire delay of standard cell from input to output without addition of the values of t_r and t_f. The values of t_r and t_f are only used as variables in the calculation of the next cell's delay.

In the propagation delay method, the total cell delay is specified as the sum of t_{pLH}, t_{pHL}, t_r and t_f. For example, in Figure 1-7, the total cell delay is the delay from the input 50% threshold to the output 10% threshold, and the transition delay is the time required for the output to rise from 10% to 50%. Therefore, the values of t_{pLH} and t_{pHL} are calculated based on the delay of cell from input until the beginning of the output transition.

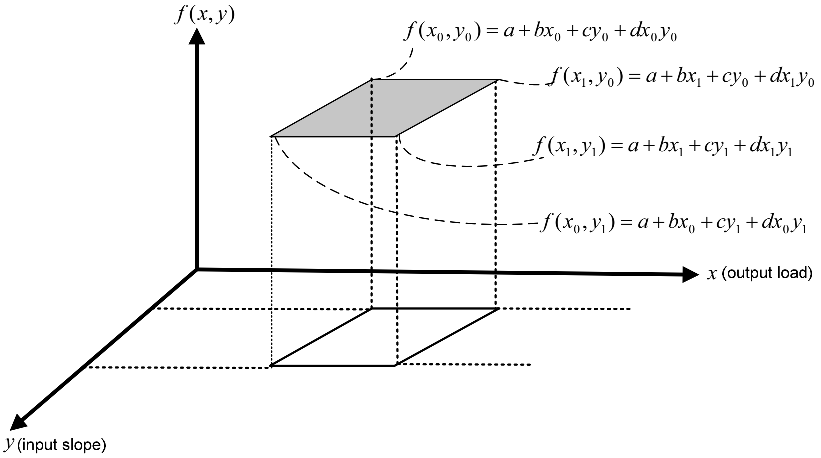

$f(x,y)$
$f(x_0,y_0)=a+bx_0+cy_0+dx_0y_0$
$f(x_1,y_0)=a+bx_1+cy_0+dx_1y_0$
$f(x_1,y_1)=a+bx_1+cy_1+dx_1y_1$
$f(x_0,y_1)=a+bx_0+cy_1+dx_0y_1$
x (output load)
y (input slope)

Figure 1-15 Nonlinear Delay Graph

The rise and fall time is the time required for the output waveform to reach the switching threshold. Incorporating the effect of the input transition at the input of the standard cell, then the total delay of the cell can be expressed as

$$f(x,y)=a+bx+cy+dxy\,, \tag{1.4.4}$$

where $f(x, y)$ is total cell delay, y is the input transition, and x is the output load capacitance. The coefficients a, b, c, and d can be determined by the fact that the propagation delay must cover the four delay corners as shown in Figure 1-15.

It is interesting to note that in discussion of such a model, if the input transition delay is large compared to the total cell delay, then the propagation delay becomes negative. This negative time evaluation might occur when the output changes before the input reaches the logic switching threshold voltage. Although this negative value is alarming, it is not a problem as long as the cell delay is a positive value. Therefore a careful model validation is required when using this type of models.

To generate a timing model for a given standard cell using a nonlinear model, circuit simulations are performed to measure the propagation delay for various input transitions and output capacitance loads.

For example, for six different input transitions and six output capacitance loads, there are thirty-six circuit simulations. These generated numbers are then stored for use by the static timing analysis tool.

The timing analysis tool uses interpolation to calculate the delay number inside the table and extrapolation for the numbers that are located outside the table. This concept is shown in Figure 1-16.

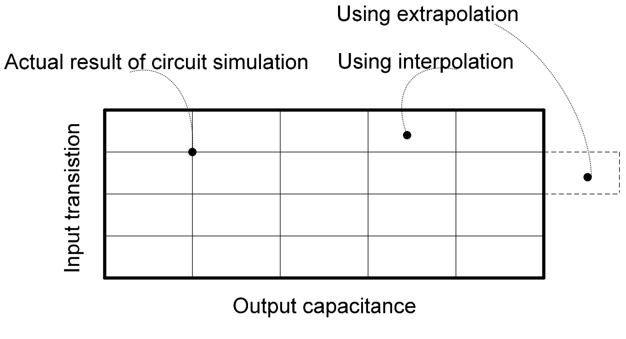


Figure 1-16 Nonlinear Delay Table

The number of rows and columns that correspond to the input transitions and output capacitance loads play an important part during static timing analysis. The larger the table, the more characterization time will be required.

The input transitions and output capacitances used in the table need to be selected to cover most of the real situations to avoid timing analysis engine extrapolation beyond the table. Extrapolation beyond the table range may lead to inaccurate results.

The extrapolation beyond the table range is the fundamental problem with this model as it relies too much on the local information (e.g. entry point of the table) and ignores the wider trends of actual data. The model is not suitable for extrapolation when it is necessary to evaluate the propagation delay outside the area defined by entry points in the table.

It is important to note that the importance of the nonlinear term xy diminishes as the number of entry points or grid size increases. Therefore, the nonlinear Equation 1.4.4 can be approximated by

$$f(x,y) = a + bx + cy\,. \qquad (1.4.5)$$

The equation shown in 1.4.5 has a tendency to extrapolate more accurately outside the table rather than interpolate inside the table (i.e. large grid size),

Both linear and nonlinear models suffer from not being adaptive to variations of external condition variables such as temperature, process, and voltage. Most of today’s standard cell library characterizations are performed for very specific conditions.

For example, for a seven (input transition) by seven (output capacitance) table there are forty-nine circuit simulations required to compute propagation delay for various conditions of temperature, process and voltage (e.g. minimum, nominal, and maximum). This is the simplest form of a timing model that a standard cell library requires for most of today’s ASIC design analysis using minimum, nominal, and maximum conditions.

Owing to the wide range of process and supply voltage variations and the static nature of linear and nonlinear delay models, these models are no longer adequate to model the delay for deep submicron processes. To address this issue, the polynomial delay model has been suggested. This timing delay model takes the form of

$$f(x,y,w,z)=\sum_{d=0}^{o}\sum_{c=0}^{n}\sum_{b=0}^{m}\sum_{a=0}^{l}K_{abcd}x^a y^b w^c z^d, \tag{1.4.6}$$

where x, y, w, and z may represent the total cell delay, input transition, output capacitance, process, and voltage.

Mathematically, two factors complicate the principal computation of such a model. One is the order of the equation (i.e. the values of l, m, n, and o) and the other is the evaluation of constant values of the K_{abcde} coefficient. The values of l, m, n, and o are in order of twenty or more, depending upon desired accuracy.

To compute the values of the K_{abcde} coefficient, least-square error-curve-fitting can be used. This means that one must determine the values of the K_{abcde} coefficient such that the difference between the measured value f_i and the calculated value $f(x_i, y_i, w_i, z_i)$ at the sample point of i are minimized. This can be expressed as

$$\delta=\sum_{i=1}^{n}[f_i-f(x_i,y_i,w_i,z_i)]^2. \tag{1.4.7}$$

In order to minimize the error value δ, the first derivative of Equation 1.4.7 must yield zero with respect to the unknown K_{abcde} coefficient:

$$\frac{\partial\delta}{\partial K}=0. \tag{1.4.8}$$

Expanding Equation 1.4.7 leads to a system of non-linear equations and can be solved by the techniques of numerical computing such as a nonlinear least-squares method.

Since the values of the K_{abcde} coefficient are chosen so that the function δ assumes its minimal value, if the errors follow a normal probability distribution, then the minimization of δ produces a best estimate of the K_{abcde} coefficient [5].

Similar to the polynomial delay model, Current Source Modeling (CSM) is another new equation-based approach to predict the cell timing delay. There are two main advantages of using the CSM delay modeling technique. One advantage is that the CSM is much simpler in comparison to the polynomial delay modeling and provides the same level of accuracy.

The difficulty with polynomial delay modeling is fitting the actual nonlinear nanometer effects in silicon using a polynomial equation with a limited number of variables and coefficients. CSM, on the other hand, is based on the topology and actual construction of the transitors and, thereby, accurately models the silicon nanometer effects by tracking nonlinear transistor switching behavior.

Another advantage is that the CSM models the output drive of a cell as a current source rather than a voltage source. Thus the delay and transition time can be derived from the mean current delivered by the gate into the load. Therefore, using mean current has a tendency to simplify the complex computation of effective capacitance loading by describing the effects of resistive shielding.

It is key to note that in comparison of each of these delay models (linear, nonlinear, polynomial, and current source), the underlying CMOS theory remains the same and the only change is the approximation used.

Apart from process and temperature variations, these models or functions represent the impact of resistance and capacitance on timing. With future ASICs, where the inductance of lines will play an important role, these models will need to be refined so that they can present the cell delay timing more accurately.

Regardless of which model one may choose, a series of circuit simulations need to be performed for various conditions in order to generate the proper models for timing analysis. The most standard simulation conditions are as follows:

- Temperature range
- Voltage range

- Process corners
- Threshold voltages

There are four classes of operational temperature ranges as shown in Figure 1-17. Room temperature (25C) is considered typical, or normal, for the temperature range condition.

CLASS	Lower Limit	Upper Limit
Military	-55C	+125C
Extended	-40C	+125C
Industrial	-40C	+85C
Commercial	0C	+70C

Figure 1-17 Operational Temperature Ranges

However, during physical design it is prudent to use a standard cell library that is characterized for the lower and upper temperature ranges.

The range for supply voltage +/–10% forms the typical value. Depending on the process, the typical value for voltage is determined by transistor's gate oxide thickness (i.e. the amount of voltage applied to a MOSFET device without damaging its gate oxide). As voltage supply technology improves, and transistor's gate oxide become thinner that requires lower typical value for voltage the voltage range can be reduced (e.g. supply voltage variation within +/–5%).

In generating simulations for process corners, one should note that the CMOS process has two steps – transistor formation, and metallization. For the PMOS and NMOS transistor formation, or Front End of the Line (FEOL), four process corners occur:

- Typical NMOS and PMOS
- Fast NMOS and slow PMOS

- Fast PMOS and slow NMOS
- Slow NMOS and PMOS

For the metallization process, or Back End of the Line (BEOL), where all interconnections and dielectric inter layers are formed, the process corners are:

- Best
- Typical
- Worst

These conditions follow a normal distribution where the center is considered as the typical value and the best, or worst, statistically vary by +/–3 σ from the center.

Most of today's semiconductor foundries offer multiple threshold voltage settings. Usually, these settings are standard, low, and high. The advantage of multiple threshold settings is that by mixing them during ASIC design, optimum power and performance can be achieved.

It should be noted that deep submicron ASIC design requires the ability to analyze circuit timing for various voltage, temperature, and process conditions. In both the linear and nonlinear models, characterization would need to be performed for each condition.

To minimize the number of characterizations, the polynomial delay model or current source delay model should be chosen, because the coefficients for the polynomial delay model and the current parameter for the CSM delay model are common for all conditions.

Once the coefficients have been calculated, analysis for temperature, or any other condition in an ASIC design, can be performed by inputting conditions as variables to the equations.

During standard cell library characterization, a combination of all the above conditions is used. However, the minimum requirement of an ASIC timing sign-off is to use the worst and the best of all conditions.

1.5 Summary

In this chapter, we have discussed the principles of ASIC standard cells, Input-Output pads designs, and methods of characterization.

In the standard cell section, we outlined the basic cell structure and briefly discussed the concept of Phase Shift Mask process, or PSM, and its impact on standard cell physical design for deep submicron technology.

In the transistor sizing section, we reviewed the basic equations involved in standard cell circuit design, and discussed the importance of properly selecting the transistor sizes and their effect on timing performance.

In the Input-Output pads section, we reviewed the basic design concept of I/O pads, and discussed the principles of ESD structure and design techniques for latch-up immunity.

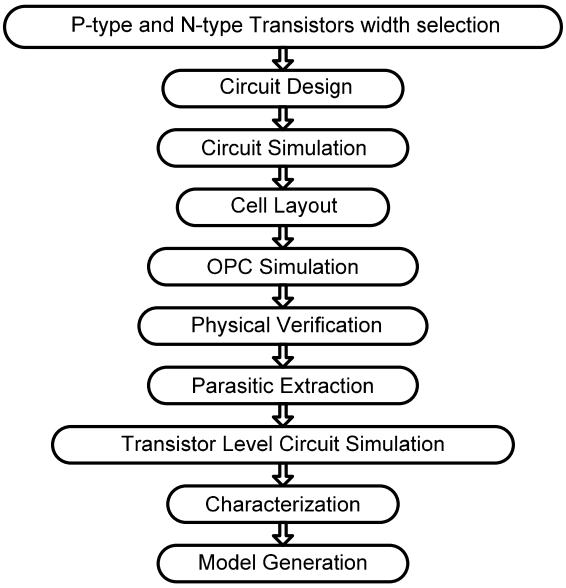


Figure 1-18 Library Development Steps

In the library characterization section, we gave an overview of the basic concepts of standard cell propagation and transition delay. In addition, we outlined and explained, in the simplest form, some of the more widely used methods for the characterization of standard cells.

We should mention that, although it seems to be a very straightforward task, developing high-quality libraries requires a great deal of expertise and knowledge in the area of circuit design, process, and modeling.

Based on the discussion in this chapter, Figure 1-18 shows the general steps that are required for development in the standard cell and I/O pad libraries phase.

References

[1] Neil H. Weste and Kamran Eshraghian, *Principles of CMOS VLSI DESIGN, A Systems Perspective*, Addison-Wesley, 1985.

[2] H.J. Levinsson and William H. Arnold, “Optical Lithography”, in *Handbook of Microlithography, Micromachining and Microfabrication,* SPIE Press, 1997.

[3] Jan M. Rabaey, *DIGITAL INTEGRATED CIRCUITS, a design perspective,* Prentice Hall Electronics and VLSI Series, 1996.

[4] Sanjay Dabral and Timothy J. Maloney, *Basic ESD and I/O Design*, John Wiley & Sons, Inc. Publishing Company, 1998.

[5] Ward Cheney and David Kincaid, *NUMERICAL MATHEMATICS AND COMPUTING*, 2 nd ed., Brooks/Cole Publishing Company, 1985.

Chapter 2

FLOORPLANNING

"Design is neither a form alone or function alone, it is aesthetic synthesis of the both." Ferdinand Porsche

Floorplanning is the art of any physical design. A well thought-out floorplan leads to an ASIC design with higher performance and optimum area.

Floorplanning can be challenging in that it deals with the placement of I/O pads and macros as well as power and ground structures. Before one proceeds with physical floorplanning one needs to make sure that the data used during the course of physical design activity is prepared properly. Proper data preparation is essential to all ASIC physical designs in order to implement a correct-by-construction design.

Entire physical design phases may be viewed as transformations of the representation in various steps. In each step, a new representation of an ASIC is created and analyzed. These physical design steps are iteratively improved to meet system requirements. For example, the placement or routing steps are iteratively improved to meet the design timing specifications.

Another challenge commonly faced by physical designers is the occurrence of physical design rule violations during ASIC design verification. If such violations are detected, the physical design steps need to be repeated to correct the errors. Sometimes these error corrections have a direct impact on the ASIC timing and may require a re-time of the design to meet timing specifications.

Most of the time, the corrections are very time-consuming. Therefore, one of the objectives of physical design is to eliminate or reduce the number of iterations during each step of the design. One of the important keys in reducing the number of iterations is the use of high quality and well-prepared data.

The types of data that are required to start a physical design are:

- Related technology and library files
- Circuit description of the design in the form of netlist representation
- Timing requirements or design constraints
- Floorplan

2.1 Technology File

Almost all physical synthesis and place-and-route tools operate based on the technology file. Technology files contain information or commands that are used to configure structures, parameters (such as physical design rules and parasitic extractions), and limits of an ASIC design targeted to specific process technology.

These commands are used at different stages of ASIC design implementation by physical design tools. One of the objectives is to make sure that all parameters in the technology files are set correctly. Once the initial technology file is created, several trial runs need to be performed and results, such as standard cell placement, routing quality, and accuracy of parasitic extraction, should be carefully analyzed. Based upon the final inspection, technology files may require further refinement for optimal performance.

Technology rule basics are as follows:

- Manufacturing grid

- Routing grid
- Standard cell placement tile
- Routing layer definition
- Placement and routing blockage layer definition
- Via definition
- Conducting layer density rule
- Metal layer slotting rule
- Routing layer physical profile
- Antenna definition

Manufacturing grid is determined by the smallest geometry that a semiconductor foundry can process. All drawn geometries during physical design must snap to this manufacturing grid.

Routing grids or tracks are used by physical synthesis and place-and-route tools during detail routing. The routing tracks can be grid-based, gridless-based, or subgrid-based.

Standard cell placement tile is used during the placement phase. The placement tile is defined by one vertical routing track and the standard cell height.

Routing layer definition is used to define the layers that are used to route the design. These definitions include wire width, routing pitch, and preferred routing direction such as vertical, horizontal, or diagonal.

Placement and routing blockage layer definitions are internal to physical design tools and are used to define "keep-out" regions for standard cell placement and routing.

Via definition defines the layer, size, and type for connection between overlapping geometries of conductor for different conductive layers. This cut layer, or via, can be a single via, stacked via, or array of via.

Conducting layer density rule defines the percentage of area of the chip that is required for processes that are using Chemical Mechanical Polishing (CMP) for each physical layer in the design.

The chemical mechanical polishing process requires limited variation in feature density on conducting layers. This dictates that the density of layout geometries in a given region must be within a certain range. With new silicon processes (i.e. metallization), violation of this rule can have negative impacts on the yield.

Configuration of metal layers slotting rule defines the minimum layer width that may need to have slotting features (i.e. a cut inside a wide routing layer). This rule varies between foundries and is used to limit mechanical stress for a given conducting layer.

Physical profile for each layer is used to define and include conductor thickness, height, and interlayer dielectric thickness. Definition of the electrical interconnect profile includes resistance and dielectric constants.

Antenna definition for each layer configures the physical design tools for automatic antenna repair. Antenna phenomena occur during the metallization process when some wires connected to the polysilicon gates of transistors are left unconnected until the upper conducting layers are deposited. A long wire connected to the gate of MOSFT can act as a capacitor or antenna that collects charges during the plasma-etching step. If this energy build-up on the floating transistor gate is suddenly discharged, the transistor could suffer permanent damage due to gate oxide breakdown. Discussion of how to fix antenna phenomena is presented in Chapter 4.

2.2 Circuit Description

In the early days of ASIC design, logic designers used schematic capture, or circuit editing tools, to implement the design. Once the design was captured, the circuit description was imported from capturing tools in some sort of format for physical design. During this time, the rapid emergence of Computer Aided Engineering (CAE) along with Computer Aided Design (CAD) technology led to the development of a broad range of interchangeable data formats and hardware description languages.

Unfortunately, most of these circuit descriptions were limited to specific companies and data types. ASIC and physical designers who wished to use a combination of different CAD tools were forced to convert among various formats in order to complete the design. This cumbersome and time-consuming translation process drove the need for a standard electronic design interchange format.

To address this issue, the first electronic industry standard format (Electronic Design Interchange Format, or EDIF) was introduced. EDIF is very rich in format and is capable of representing connectivity information, schematic drawings, technology and design rules, and Multi Chip Module (MCM) descriptions, as well as allowing transfer of documentation associated with physical layouts.

During the same time, two major formats were introduced–Very High Speed Integrated Circuit (VHSIC) or Hardware Description Language VHDL and the Verilog description language (commonly referred to as Verilog).

From an ASIC design perspective, VHDL is not a circuit design tool but merely creates an accurate model of circuit designs. This language, however, has been a phenomenal success because it has the utility and ability to meet circuit and system design requirements. VHDL is independent of design tools and is a powerful language for the modeling of hardware timing.

With the introduction of the first logic synthesis tools, the Verilog model that represented the functionality of the circuit designs could be synthesized. This was a major event, as the top-down design methodology could now be used effectively. The design could be modeled down at the Register Transfer Level (RTL) and could then be translated into the gate using synthesis tools. With this event, the use of Verilog modeling increased dramatically.

Use of Verilog simulation for sign-off certification by ASIC designers was the next major trend to emerge. As Verilog became popular with semiconductor vendors' customers, they began to move away from their own proprietary simulators and to use Verilog simulators for timing certification.

However, Verilog remained a closed language and the pressures of standardization eventually caused the industry to shift to VHDL. Once this was recognized, the Open Verilog International (OVI) committee was formed which brought standardization to Verilog. With standardization, Verilog simulators are now available for most computers, with a variety of performance characteristics and features.

The Verilog language is growing faster than any other hardware description language and is more heavily used than ever. It has truly become the standard hardware description language.

In today's ASIC design flow, the design is generated in RTL format, along with design constraints being synthesized and final results in the form of gate level description, and is imported to the physical synthesis tools for physical design implementation.

Verilog gate-level netlists are widely used owing to their ease of understanding and clear syntax. Although the behavioral Verilog language has a vast number of keywords, there are only a few that may be used in structural Verilog to represent the entire circuit function and connectivity.

A structural Verilog netlist consists of keywords, names, literal comments, and punctuation marks. A Verilog structural netlist is case-sensitive, and all its keywords, such as *module, endmodule, input, output, inout, wire,* and *assign* are lowercase.

The most basic element in Verilog is a *module* definition with corresponding input and output ports [1]. It represents a logical entity that is usually implemented in a piece of hardware. A module can be a simple gate or a complex network of gates. The ports in modules can be a single-bit or multiple bits wide, and each port can be defined as an *input*, *output,* or *inout* (i.e. bidirectional) port. The nets that connect the elements inside a module are described by wire statements.

For improved readability, spaces, tabs, and new lines can be used. A single line comment can start with "//". For multiple line comments, start with "/*" and end with "*/".

A module in the Verilog language starts with the keyword *module* followed by the module name, then the list of inputs and outputs. It ends with the keyword *endmodule*. Each module name must be unique.

Figure 2-1 represents a logical entity implemented using the Verilog circuit modeling style.

```
module bottom_level ( D, E, Y1, Y2, Y3, Y4);
    input      D[3:0], E;
    output     Y1,Y2,Y3;
    inout      Y4;

    assign          D[2] = Y2;
    OR_type         I0 ( .A(D[0]), .B(D[1]), .Z(Y1) );
    AND_type        I2 ( .A(D[2]), .B(Y4), .Z(Y3) );
    TRIBUF_type     I3 (. A(D[3]), .ENB(E), .Z(Y4) );
endmodule

module top_level (IN0,IN1,IN2,IN3,EN,OUT0,OUT1,BIDIR);
    input      IN0,IN1,IN2,IN3,EN;
    output     OUT0, OUT1;
    inout      BIDIR;
    wire       NET1,NET2,NET3,NET4,NET5,NET6;

    bottom_level  bottom_level_instance ( .D{net4,net2,IN1,
          net1}, .Y1(net5), .Y2(), .Y3(net6), .Y4(BIDIR) );
    BUF_type     I0 ( .A(IN0), .Z(net1) );
    BUF_type     I1 ( .A(IN2), .Z(net2) );
    BUF_type     I2 ( .A(EN), .Z(net3) );
    BUF_type     I3 ( .A(IN3), .Z(net4) );
    BUF_type     I4 ( .A(net5), .Z(OUT0) );
    BUF_type     I5 ( .A(net6), .Z(OUT1) );
endmodule
```

Figure 2-1 A Logical Entity in Structural Verilog Format

The entity that is shown in Figure 2-1 contains simple OR, AND, TRISTATE BUFFER, and BUFFER gates.

The gate level representation or schematic of a structural Verilog netlist, as shown in Figure 2-1, is illustrated in Figure 2-2.

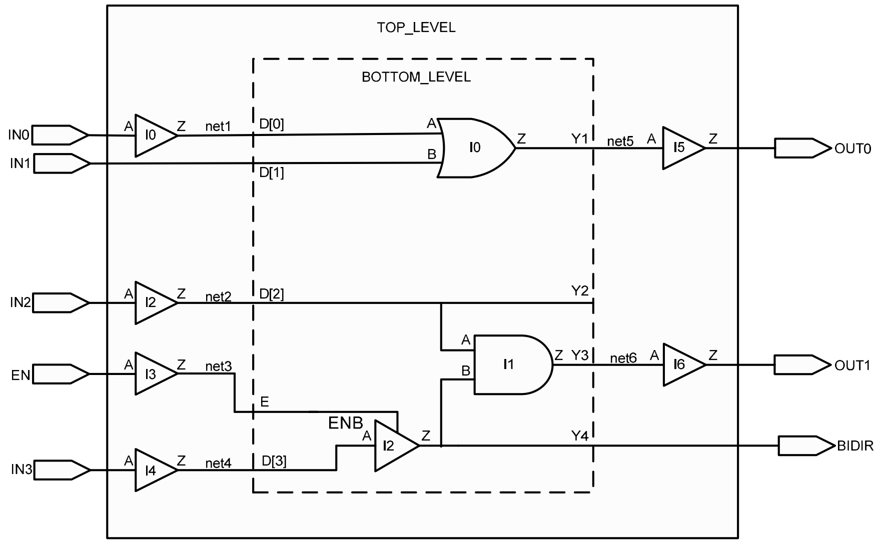


Figure 2-2 Gate Level Representation

Figure 2-1 and its corresponding schematic presentation of that in Figure 2-2 show there are two module definitions – **bottom_level** and **top_level** and their corresponding input and output ports. The structural module **top_level** has one instance of **bottom_level**. In addition, both **top_level** and **bottom_level** modules contain gate types (e.g. OR_type, AND_type, TRIBUF_type, and BUF_type) that are considered cell types. These cell types are built into a standard cell library and have predefined functions.

The **bottom_level** module has a single continuous assignment statement, indicated by the keyword *assign*. This keyword assigns the value of the right-hand side of the "=" to the value on the left-hand side of the "=".

Although the *assign* statement is considered to be a behavioral statement, it is widely used in today's structural Verilog netlists and presents a physical short between two names. This implies having a wire which has two distinct names attached at each end. When dealing with the *assign* keyword, special care must be taken during the physical ASIC verification to avoid misleading results. One of the most frequently used techniques is to replace the *assign* keyword with a buffer.

After processing the imported netlist, the next step is to apply design constraints as explained in the next section.

2.3 Design Constraints

Design constraints are ASIC design specifications that are applied during logic and physical synthesis. Each tool attempts to meet two general design constraints:

- Timing constraints
- Design rule constraints

Timing constraints are user-specified and are related to speed, area, and the power consumption of the ASIC design.

Timing constraints utilized by physical design tools are performance related. The most basic timing constraints are as follows:

- System clock definition and clock delays
- Multiple cycle paths
- Input and output delays
- Minimum and maximum path delays
- Input transition and output load capacitance
- False paths

System clocks, and their delays, are extremely important constraints in ASIC designs. System clocks are typically supplied externally, but can be generated inside an ASIC. All delays, especially in a synchronous ASIC design, are dependent upon the system clocks.

Most logic synthesis tools consider the clock network delays to be an ideal (i.e. a clock with fixed latency and zero skew) and are used during design synthesis. The physical design tools use the system clock definition to perform what is known as Clock Tree Synthesis (CTS) and try to meet the clock networks' delay constraints.

Multiple cycle paths are for ASIC designs that have a non-single cycle clock timing requirement. This directs the physical design tools to avoid optimization of data paths that have non-single clock behavior.

Input and output delays are used to constrain the boundary of external paths in an ASIC design. These constraints specify point-to-point delays from external inputs to the first registers and from registers to the outputs of an ASIC design.

Minimum and maximum path delays provide greater flexibility for physical synthesis tools that have a point-to-point optimization capability. This means that one can specify timing constraints from one specific point (e.g. pin or port) in the ASIC design to another, provided such a path exists between the two specified points.

Input transition and output capacitance loads are used to constrain the input slew rate and output capacitance of an ASIC device input and output pins. These constraints have a direct effect on the final ASIC design timing.

The values of these constraints are set to zero during physical design and place-and-route activity to ensure that the actual ASIC design timing is calculated independent of external conditions and to make sure register-to-register timing is met. Once that is achieved, these external conditions can be applied to the design for input and output timing optimization.

False paths are used to specify point-to-point non-critical timing either internal or external to an ASIC design. Properly identifying these noncritical timing paths has a great impact on physical design tools' performance.

Design rule constraints are imposed upon ASIC designs by requirements specified in a given standard cell library or within physical design tools.

Design rule constraints have precedence over timing constraints because they have to be met in order to realize a functional ASIC design. There are four types of major design rule constraints:

- Maximum number of fan-outs
- Maximum transitions
- Maximum capacitance

- Maximum wire length

Maximum number of fan-outs specify the number of destinations that one cell can connect to for each standard cell in the library. This constraint can also be applied at the ASIC design level during the physical synthesis to control the number of connections one cell can make.

Maximum transition constraint is the maximum allowable input transitions for each individual cell in the standard cell library. Apart from each element in the standard cell library, this constraint can be applied to a specific net or to an entire ASIC design.

Maximum capacitance constraint behaves similarly to maximum transition constraint, but the cost is based on the total capacitance that a particular standard cell can drive any interconnection in the ASIC design. It should be noted that this constraint is fully independent of maximum transition, and therefore, it can be used in conjunction with maximum transition.

Maximum wire length constraint is useful for controlling the length of wire to reduce the possibility of two parallel long wires of the same type. Parallel long wires of the same type may have a negative impact on the noise injection and may cause crosstalk.

These design rule constraints are mainly achieved by properly inserting buffers at various stages of physical design. Thus, it is imperative to control the buffering in an ASIC design during place-and-route to minimize area impact.

2.4 Design Planning

Efficient design implementation of any ASIC requires an appropriate style or planning approach that enhances the implementation cycle time and allows the design goals such as area and performance to be met. There are two style alternatives for design implementation of an ASIC–flat or hierarchical. For small to medium ASIC's, flattening the design is most suited; for very large and/or concurrent ASIC designs, partitioning the design into subdesigns, or hierarchical style, is preferred.

The flat implementation style provides better area usage and requires effort during physical design and timing closure compared to the hierarchical style. The area advantage is mainly due to there being no need to reserve extra space around each subdesign partition for power, ground, and resources for the routing. Timing analysis efficiencies arise from the fact that the entire design can be analyzed at once rather than analyzing each subcircuit separately and then analyzing the assembled design later. The disadvantage of this method is that it requires a large memory space for data and run time increases rapidly with design size.

The hierarchical implementation style is mostly used for very large and/or concurrent ASIC designs where there is a need for a substantial amount of computing capability for data processing. In addition, it is used when subcircuits are designed individually. However, hierarchical design implementation may degrade the performance of the final ASIC. This performance degradation is mainly because the components forming the critical path may reside in different partitions within the design thereby extending the length of the critical path. Therefore, when using a hierarchical design implementation style one needs to assign the critical components to the same partition or generate proper timing constraints in order to keep the critical timing components close to each other and thus minimize the length of the critical path within the ASIC.

In the hierarchical design implementation style, an ASIC design can be partitioned logically or physically.

Logical partitioning takes place in the early stages of ASIC design (i.e. RTL coding). The design is partitioned according to its logical functions, as well as physical constraints, such as interconnectivity to other partitions or subcircuits within the design. In logical partitioning, each partition is place-and- routed separately and is placed as a macro, or block, at the ASIC top level.

Physical partitioning is performed during the physical design activity. Once the entire ASIC design is imported into physical design tools, partitions can be created which combine several subcircuits, or a large circuit can be partitioned into several subcircuits. Most often, these partitions are formed by recursively partitioning a rectangular area containing the design using vertical or horizontal cut lines.

Physical partitioning is used for minimizing delay (subject to the constraints applied to the cluster or managing circuit complexity) and satisfying timing and other design requirements in a small number of subcircuits. Initially, these partitions have undefined dimensions and fixed area (i.e. the total area of cells or instance added to the partition) with their associated ports, or terminals, assigned to their boundaries such that the connectivity among them is minimized. In order to place these partitions, or blocks, at the chip level, their dimensions as well as their port placement must be defined.

One method that is suggested to estimate the perimeter of a macro instance is to use the number of terminals or ports allowed for each block and their associated spacing between each terminal [2]. The relationship between the perimeter of each partition and the number of associated terminals is given by

$$P = NS\,, \qquad (2.4.1)$$

where P is the perimeter of the physical partition block, N is the number of terminals or ports, and S corresponds to spacing between terminals.

The perimeter estimate given by Equation 2.4.1 determines an appropriate width and height for each partition in the hierarchy, based on the wire demand in both vertical and horizontal directions. However, in order to fit each macro instance at the chip top level in an effective manner, the automatic floorplan algorithm needs to have a range of legal shapes that is derived from aspect ratio bounds for each partition in the design.

The aspect ratio bounds that are generated by the hierarchical floorplan algorithm must have the flexibility to ensure that each macro instance shape can be reshaped for optimum placement. Because one requirement for an ASIC design is to fit into the smallest available die size, during the partitioning process for each partition several layout alternatives are considered by modification of the dimensions and terminal placements along the boundary of each partition in the design such that the amount of unused and routing area between each partition is minimized.

Both flat and hierarchical physical design implementation begin with importing technology files, library files, the netlist, and design constraints into the physical design tools. Once this data is imported, the physical design tool performs binding operations on the entire netlist for flat design implementation or for each sub-netlist for hierarchical design implementation.

During the binding, or linking, process, the incoming netlist is flattened and all entities are analyzed to determine their available models. A variety of checks is performed automatically to determine whether the physical synthesis or place-and-route internal data structure is ready to proceed with the rest of the design implementation flow.

Often the checks that detect problems with the physical database are related to the netlist, such as unconnected ports, mismatched ports, standard cell errors, or errors in the library and technology files. Because of these checks, a log file that contains all errors and warnings will be generated. It is important to review the log file and make sure that all reported errors and warnings are resolved before proceeding to the next phase.

Regardless of the physical design implementation style, after physical database creation using the imported netlist and corresponding library and technology files, the first step is to determine ASIC device and core width and height. In addition, standard cell rows and I/O pad sites are created. The rows and I/O pad sites are for standard cell and I/O pad placement. Figure 2-3 shows an initial ASIC design floorplan.

The height of a row is equal to the height of the standard cells in the library. If there are any multiple-height standard cells in the library, they will occupy multiple rows.

Most of the time, standard rows are created by abutment. The standard rows are oriented in alternating 180-degree rotation or are flipped along the X-axis so that the standard cells can share power and ground busses. If the ASIC core has routing congestion owing to the limited number of routing layers, one solution is to create routing channels between rows. These all can be separated individually or as pairs.

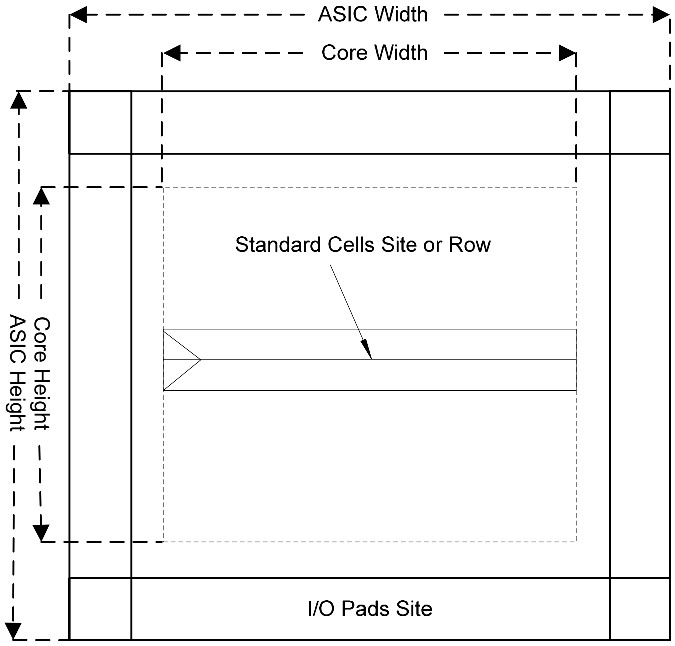


Figure 2-3 ASIC Design Initial Floorplan

2.5 Pad Placement

Correct I/O pad placement and selection is important for the correct function of any ASIC design. As mentioned in Chapter 1, for a given ASIC design there are three types of I/O pads. These pads are power, ground, and signal.

It is critical to functional operation of an ASIC design to insure that the pads have adequate power and ground connections and are placed properly in order to eliminate electromigration and current-switching noise related problems.

Electromigration (EM) is the movement or molecular transfer of metal from one area to another area that is caused by an excessive electrical current in the direction of electron flow (electron “wind”). Electromigration currents exceeding recommended guidelines can result in premature ASIC device failure. Exceeding electromigration current density limits can create voids or hillocks, resulting in increased metal resistance, or shorts between wires, and can impair ASIC performance. Figure 2-4 shows electromigration damage due to excess current captured by Electron Scanning Microscopy (ESM) with 10K magnification.

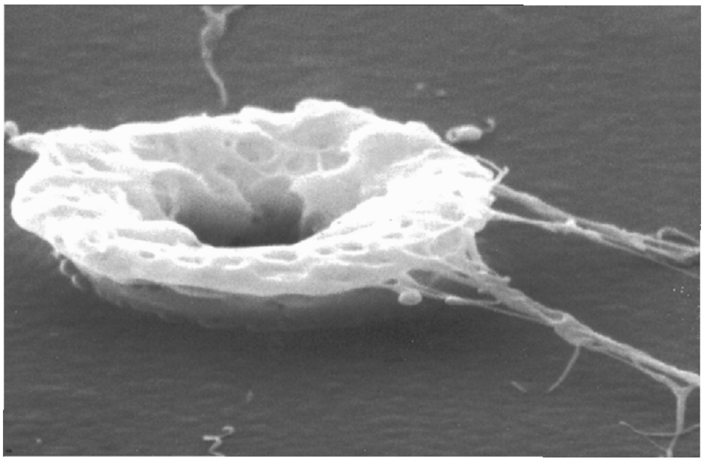


Figure 2-4 Electromigration Damage

Using Equation 2.5.1, one can determine the minimum number of ground pads required to satisfy the electromigration current limits. The required number of power pads is equal to the required number of ground pads and is given by

$$N_{gnd} = \frac{I_{total}}{I_{\max}}, \tag{2.5.1}$$

where N_{gnd} is the number of ground pads; I_{total} is the total current in an ASIC design (which is the sum of its static and dynamic currents in amperes); and $I_{\max}$ is the maximum EM current in amperes per ground pad.

Switching noise is generated when ASIC outputs make transitions between states. An inadequate number of power and ground pads will lead to system data errors due to these switching noise transients. There are two types of mechanisms that can cause noise:

- dv/dt caused by a capacitive coupling effect
- di/dt caused by an inductive switching effect

The capacitive coupling effect is the disturbance on the adjacent package pin caused when switching transients inject pulses via parasitic coupling capacitance.

The maximum $C(dv/dt)$ noise occurs when the ASIC output current nears the maximum current for a given capacitive output load of C. This noise problem can be resolved by proper pad placement, package pin selection, ASIC output pad type and drive current, and input pad type.

To reduce or eliminate capacitive coupling effects during I/O pad placement and selection, one may consider the following guidelines:

- Isolate sensitive asynchronous inputs such as clock or bidirectional pins from other switching signal pads with power or ground pads
- Group bidirectional pads together such that all are in the input or output mode
- Group slow input pads together positioning them on higher inductance package pins
- Use input pads with hysteresis as much as possible

The inductive switching effect is related to simultaneous switching ASIC outputs that induce rapid current changes in the power and ground busses. The inductance in the power and ground pins cause voltage fluctuations in the ASIC internal power and ground level relative to the external system.

These rapid changes in current can change the ASIC input pads' threshold and induce logic errors or may cause noise spikes on non-switching output pads that affect signals connected to other systems.

The maximum $L(di/dt)$ occurs when ASIC output starts to make the transition to another voltage level and its absolute current increases from zero through a wire with inductance of L. Factors such as process, ambient temperature, voltage, location of output pads, and number of simultaneous switching output pads determine the magnitude of inductive switching noise.

To control inductive switching noise, enough power and ground pads must be assigned and placed correctly. This way the noise magnitude will be limited. This noise reduction will prevent inputs of ASIC design from interpreting the noise as valid logic level.

Successful reduction of inductive switching noise can be accomplished by the following:

- Reduce the number of outputs that switch simultaneously by dividing them into groups with each group having a number of delay buffers inserted into their data paths
- Use the lowest rated sink current or low-noise output pads as long as speed is not an issue
- Place the simultaneously switching output or bidirectional pads together and distribute power and ground pads among them according to their relative noise rating
- Assign static and low frequency input pads to higher inductance package pins
- Reduce the effective power and ground pin inductance by assigning as many power and ground pads as possible

2.6 Power Planning

The next step is to plan and create power and ground structures for both I/O pads and core logic. The I/O pads' power and ground busses are built into the pad itself and will be connected by abutment.

For core logic, there is a core ring enclosing the core with one or more sets of power and ground rings. A horizontal metal layer is used to define the top and bottom sides, or any other horizontal segment, while the vertical metal layer is utilized for left, right, and any other vertical segment. These vertical and horizontal segments are connected through an appropriate via cut. The next consideration is to construct the standard cell power and ground that is internal to the core logic. These internal core power and ground busses consist of one or two sets of wires or strips that repeat at regular intervals across the core logic, or specified region, within the design. Each of these power and ground strips run vertically, horizontally, or in both directions. Figure 2-5 illustrates these types of power and ground connections.

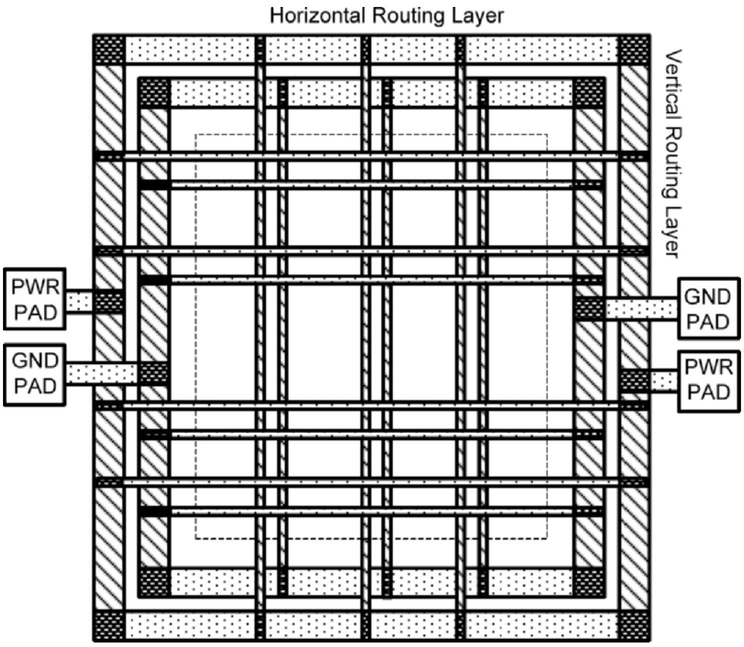


Figure 2-5 Ring and Core Power and Ground

If these strips run both vertically and horizontally at regular intervals, then the style is known as power mesh. The total number of strips and interval distance is solely dependent on the ASIC core power consumption.

As the ASIC core power consumption (dynamic and static) increases, the distance of power and ground strip intervals increases. This increase in the power and ground strip intervals is used mainly to reduce overall ASIC voltage drop, thereby improving ASIC design performance.

In addition to the core power and ground ring, macro power and ground rings need to be created using proper vertical and horizontal metal layers. A macro ring encloses one or more macros, completely or partially, with one or more sets of power and ground rings.

Another important consideration is that when both analog and digital blocks are present in an ASIC design, there is a need for special care to insure that there is no noise injection from digital blocks or core into the sensitive circuits of analog blocks through power and ground supply connections.

Much of this interference can be minimized by carefully planning the power and ground connections for both digital core and analog blocks. There are several methods to improve the noise immunity and reduce interference.

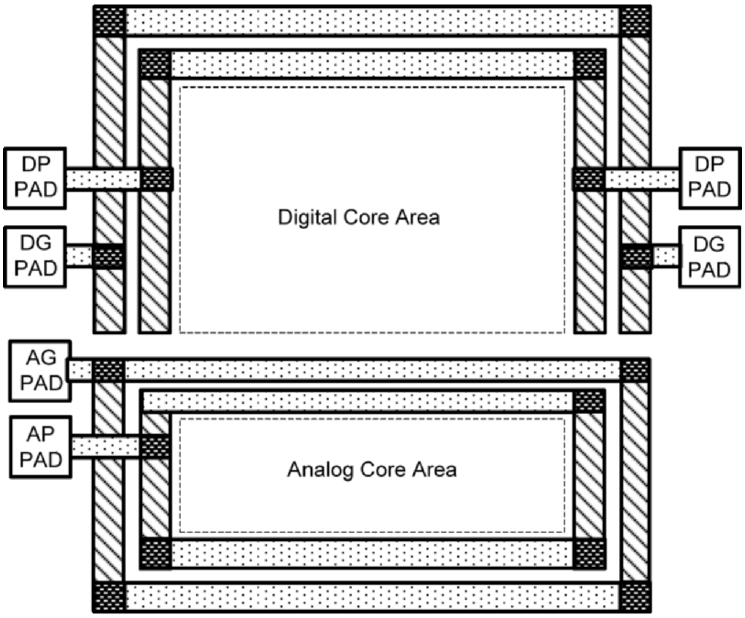


Figure 2-6 Decoupled Analog and Digital Core Power Supply

The most effective method is to decouple the digital and analog power and ground by routing the digital power/ground (DP and DG) and analog power/ground (AP and AG) supply connections separately [3] as shown in

Figure 2-6. However, this decoupling will not be complete if there is ground connectivity from the die substrate through standard cells' ground (i.e. source).

In order to make sure that analog circuits are completely decoupled from digital circuits, one needs to separate the substrate from the ground in the standard cells (e.g. NWELL Process). This is not mandatory but depends on how sensitive the analog circuit is with respect to noise injection from the digital core area.

It is strongly recommended to check for power and ground connectivity and/or any physical design rule violations after construction of the entire power and ground network.

Depending on the ASIC design's power supply requirements, the width of the core macro power and ground ring could exceed the maximum allowable in order to reduce their resistance during floorplanning and power analysis. This increase in power and ground width could be problematic from manufacturing point of view.

The main problem with wide metal (i.e. exceeding manufacturing limits) is that a metal layer cannot be processed with uniform thickness, especially when applied over a wide area. The metal becomes thin in the middle and thick on the edges causing yield and current density problems. To solve this metal density problem, one can either use multiple power and ground busses or use metal slotting.

Most of today's physical design tools are capable of performing slotting on a wire-by-wire or segment-by-segment basis by cutting those wires or segments with widths greater than the specified maximum into multiple, thinner segments of the same type.

These maximum specified widths for different metal layers are determined by semiconductor manufacturers as a slot-width and need to be included in the physical design tool technology file.

2.7 Macro Placement

Typically, the placement of macros takes place after I/O placement, and after power and ground bus structure has been defined. Macros may be memories, analog blocks, or in the case of hierarchical style, an individually placed and routed subcircuit. Proper placement of these macros has a great impact on the quality of the final ASIC design.

Macro placement can be manual or automatic. Manual macro placement is more efficient when there are few macros to be placed and their relationship with the rest of the ASIC design is known. Automatic macro placement is more appropriate if there is not enough information on which to base the initial macro placement and/or the number of macros is large.

During the macro placement step, one needs to make sure that there is enough area between blocks for interconnections. This process (commonly known as channel allocation or channel definition) can be manual or can be accomplished by floorplan tools. The slicing tree is used by the floorplan algorithm for slicing floorplan during macro placement and to define routing channels between the blocks.

Most of today's physical design tools use a global placer to perform the automatic initial macro placement based on connectivity and wire length reduction. Wire length optimization is the most prevalent approach in automatic macro placement.

With increases in the number of embedded blocks such as memories, placing macros of varying sizes and shapes without a good optimization algorithm can result in fragmentation of placement and routing space that can prevent a physical design from being able to complete the final route.

One of the basic algorithms used for automatic macro placement considers that macros are connected to each other by nets and are supposed to exert attractive forces on each other by means of wire length proportional to the distance between these macros.

Automatic macro placement is an iterative process. During the macro placement process, macros are free to move until the equilibrium or optimum wire length is achieved. It is interesting to note that in this algorithm, if there is no relationship between the macros, they tend to repel each other and their placement result may not be optimum.

To improve the placement quality of macros that are not related to each other, one may consider simultaneous standard cell and macro placement provided the physical design tool can deal with both macro and standard cell placement at once. In terms of algorithms, while commercial physical design tools have considerably improved in the past few years, automatic macro placement is still in the early stages of development compared to standard cell placement.

The implementation challenge associated with macro placement is conceptually a time and space problem that needs to be solved simultaneously.

A well-developed macro placement algorithm must be able to handle widely differing shapes and sizes, macro orientation, congestion, and timing-driven placement. Although many improvements have been incorporated into the macro placement algorithms to insure the quality of their placement, one might need to modify the resulting location and/or orientation in order to achieve an optimum floorplan based on the physical quality of measurement.

When it is not an easy task to measure the macro placement quality of an ASIC design containing a large number of blocks, there are some basic physical measurements that one can adopt.

Given a placement solution, the physical measurements could be wire length, data flow direction (e.g. the macro placement relative to each other as well as to standard cell placement), or macro, port accessibility and related timing.

The total wire length for a given placement is a good indicator when comparing different placements of the same physical design. In order to decrease overall wire length, ensure that the chip area is not segmented by the macro's placement.

To avoid area segmentation, macros should be positioned such that the standard cell area is continuous. An area with close to 1:1 aspect ratio is recommended as it allows standard cell placers to utilize the area more efficiently and thereby reduce total wire length.

The segmented floorplan leads to an excess of wire length interconnections from the standard cells located at the bottom of the die to those at the top of the die. Thus, it is necessary that the macros be kept along the ASIC core area in order to avoid a floorplan segmentation problem.

Figure 2-7 shows a problematic segmented floorplan that may lead to long interconnections between the bottom and top of the die.

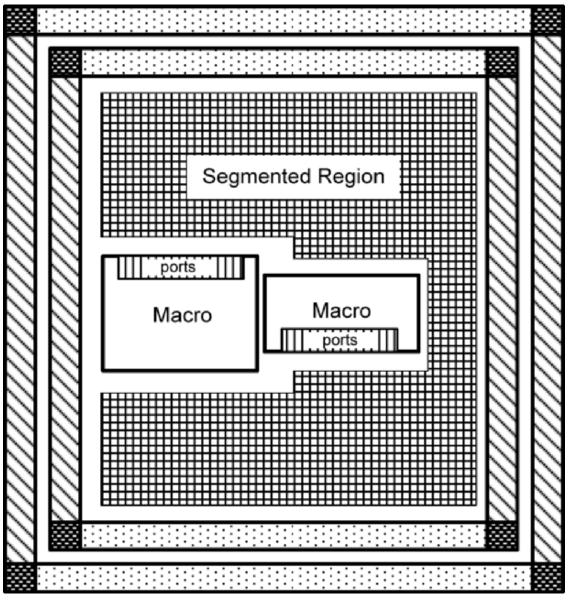


Figure 2-7 Segmented Floorplan

Another aspect of increased wire length is related to macro placement with respect to their orientation and pin placements. Depending on the macro orientation and their actual pin location, the length of the nets connecting to the macros can be different, and can have significant impact on the routing optimization process.

With respect to wire length reduction, macros should be oriented such that their ports are facing the standard cells, or core area, and their orientation should match the available routing layers. Thus, any macro placement algorithm requires computing the macro's interconnect distance by including proper orientation and pin positions.

Figure 2-8 shows a floorplan with macro ports facing the standard cell region, thereby minimizing localized increase in wire length.

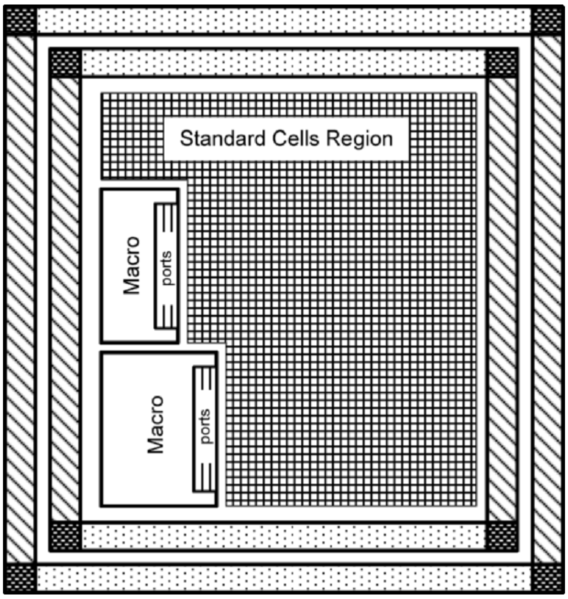


Figure 2-8 Floorplan with Macros Facing Standard Cells Region

Another aspect of physical measure is that of macro placements relative to each other, as well as to the standard cell placement, and the macro port accessibility has a direct impact on the chip's final routing. The macro placement, and thereby the quality measure, can be determined by the analysis of routing congestion produced by a global router.

Most global routers are capable of producing both graphical and text statistical reports. The graphical report, also referred to as the congestion map, provides a visual aid to see where routing congestion exists (e.g. hot spots). The statistical report is a good indication of how much a physical design is congested.

The most common scenarios that cause physical design routing congestion are: there may not be enough space between the macros to provide routing channels–especially for I/O connections and macros (if these macros are placed at the ASIC periphery and over the macro); routing is prohibited; or standard cell trap pockets may be around the edges of macros or within the corners of the floorplan.

Standard cell trap pockets are long, thin channels between macros. If many cells are placed in these channels, routing congestion can result. Therefore, these channels need to be kept free for most standard cells and should be available for repeater or buffer insertion (if this type of insertion is supported by the physical design tool). Figure 2-9 shows a floorplan with a standard cell trap pocket.

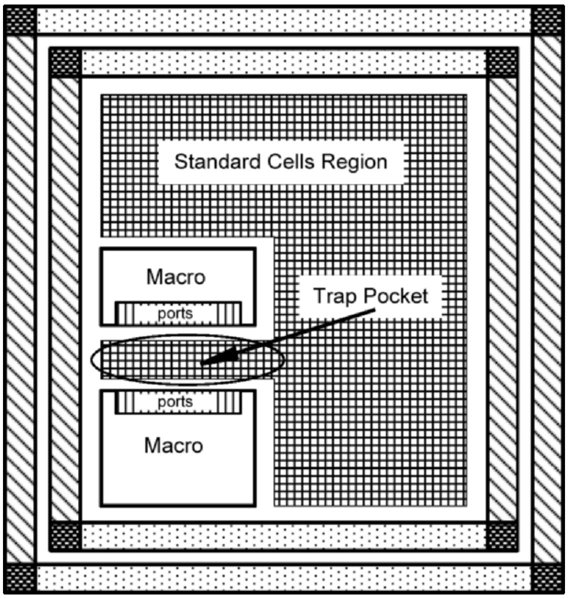


Figure 2-9 Floorplan with Standard Cells Trap Pocket

After macro placement and before performing global routing, most physical designs require keep-out or buffer-only regions to be defined by drawing a blockage layer over an area containing macros to prevent the placer from moving any standard cells into those regions.

Naturally, the wires that are used in keep-out regions have a tendency to be long. By allowing buffer insertion in those areas by using a buffer-only region (or blockage), the placer will taper these long nets and thus avoid the long transition times associated with them.

These blockage layers are created over pre-placed macros such that their power and ground rings are covered. Blockage layers are also used to relieve routing congestion around the macro's corners. When a macro is blocked on many routing layers, wires have a tendency to detour around corners and connect to nearby standard cells thereby creating routing congestion at the corner of the macros.

To reserve more resources for the router, one can draw a blockage layer at these corners. These blockage regions can be simple or gradual as illustrated in Figure 2-10.

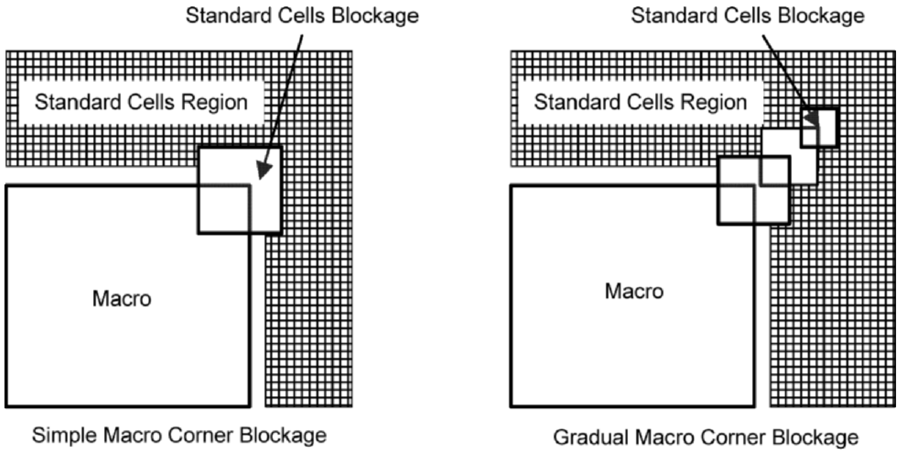


Figure 2-10 Macro Corner Standard Cells Blockage

After refinement of floorplan and macro placement, standard cells are placed and connectivity analysis is performed. Connectivity analysis is the process of studying the connections between macro pairs, macro, I/O pads, and related standard cell instances. Connectivity analysis is also used to identify macros that have substantial direct connectivity and to refine their locations accordingly.

This analysis is conducted by using what is known as fly lines. When fly lines are activated through physical design or place-and-route tools, Graphic User Interface (GUI) displays the lines that mark the connections between currently selected instances (e.g. standard cells, macros, or I/O pads). Using fly lines, one can analyze and identify situations where moving or rotating macros will yield shorter wire lengths that improve the overall ASIC routability during the floorplanning stage of the physical design cycle.

2.8 Clock Planning

Clock planning is considered another part of floorplanning. The idea of the implementation of clock distribution networks is to provide clock to all clocked elements in the design in a symmetrically-structured manner.

Although most ASIC designs use clock tree synthesis, clock tree synthesis may not be sufficient for very high-performance and synchronized designs. In this case, one needs to implement the distributed clock networks manually in order to minimize the skew between communicating elements due to their line resistance and capacitance.

The basic idea of manual implementation of clock distribution networks is to build a low resistance/capacitance grid similar to power and ground mesh that covers the entire logic core area as shown in Figure 2-11.

It is important to note that this type of clock grid does not rely on the matching of components such as clock buffers. However, it can present a systematic clock skew.

In order to minimize such clock skew, a clock tree that balances the rise and fall time at each clock buffer node should be utilized during clock planning. This minimizes the hot-electron effect. The problem of hot-electron occurs when an electron gains enough energy to escape from a channel into a gate oxide.

The presence of hot-electron effect in the gate oxide area causes the threshold voltage of the device to change and thus alters the delay of the clock buffers. This in turn leads to an additional skew. Hence, balancing the rise and fall times of all clock buffers means that hot-electron effect influences the clock buffers at the same rate, minimizing unpredictable skew.

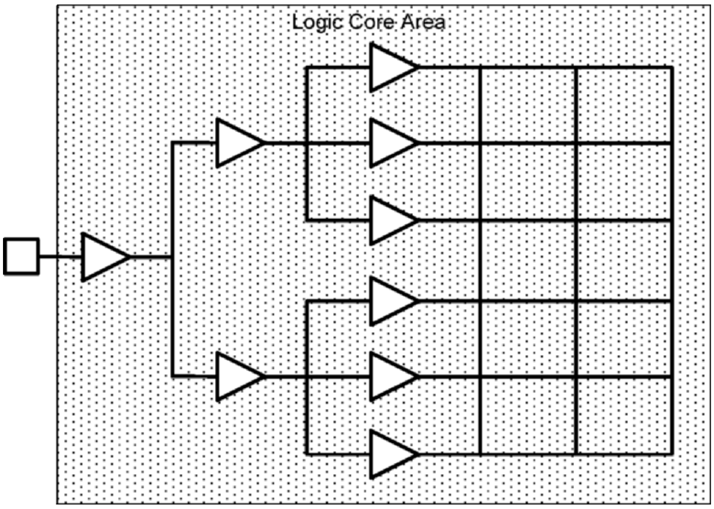


Figure 2-11 Clock Distribution Network

It is essential to realize that clock grid networks consume a great deal of power due to being active all the time and it may not be possible to make such networks uniform owing to floorplanning constraints (e.g. to spread the power dissipation evenly across the chip).

Another aspect of clock planning is that it is well suited to hierarchical physical design. This type of clock distribution is manually crafted at the chip level, providing clock to each sub-block that is place-and-routed individually.

To minimize the clock skew among all leaf nodes, the clock delay for each sub-block must be determined and the design of the clock planned accordingly.

Figure 2-12 illustrates typical hierarchical clock planning.

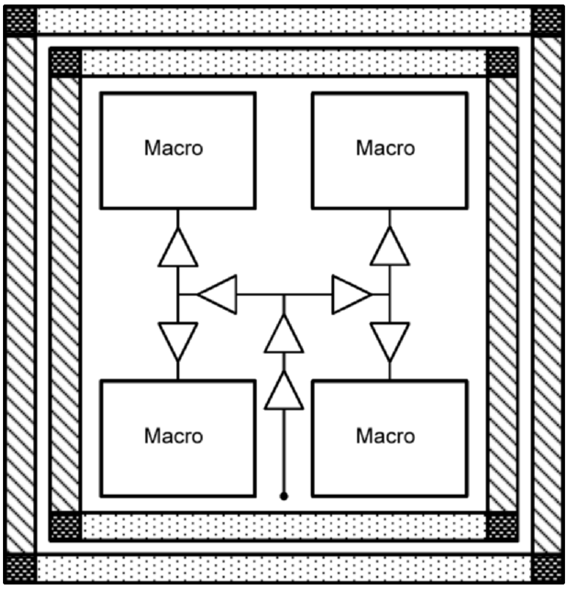


Figure 2-12 Hierarchical Clock Planning

2.9 Summary

In this chapter, we have explained various basic aspects of physical design data preparation and ASIC physical design and floorplanning alternatives.

In the area of data preparation, we have provided a general idea of the technology files that are required by physical design tools, and have provided an example of a Verilog structural netlist with the most common description of its syntax and keywords.

In the design constraint section, we have discussed several important timing and design constraints and their impact on the quality of the ASIC physical design.

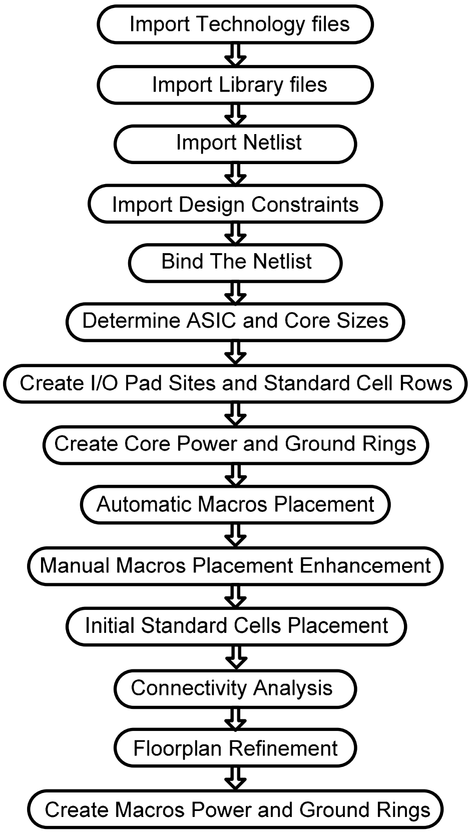

Import Technology files
Import Library files
Import Netlist
Import Design Constraints
Bind The Netlist
Determine ASIC and Core Sizes
Create I/O Pad Sites and Standard Cell Rows
Create Core Power and Ground Rings
Automatic Macros Placement
Manual Macros Placement Enhancement
Initial Standard Cells Placement
Connectivity Analysis
Floorplan Refinement
Create Macros Power and Ground Rings

Figure 2-13 Basic Floorplanning Steps

In the design implementation section, we have outlined the fundamentals of different floorplanning styles, mainly flat and hierarchical, and their advantages, and we have explained basic floorplanning techniques.

In addition, we have outlined the importance of automatic partitioning of a design, and port optimization and minimization for ASIC design. We should note that selection of a floorplanning style depends upon many factors such as the type of ASIC, area, and performance, and relies heavily on one's physical design experience.

In the input-output section, we have explained placement and given basic guidelines with respect to coupling capacitance and inductive switching.

In the power and ground section, we have shown several styles of creating power and ground connections. Again, depending upon the floorplanning style, the power and ground connections need to be designed to meet the ASIC power requirements and can vary from design to design.

In the macro placement section, we have illustrated various macro placements based on industry practices. In addition, we should note that regardless of what style of macro placement is used during floorplanning, a well thought-out floorplan and macro placement leads to a higher quality of the final ASIC design with respect to performance and area.

Finally, in the clock planning section we have briefly discussed the manual clock planning topology and its importance for high-speed design applications.

Figure 2-13 exhibits the basic steps that are involved during the physical design floorplanning phase.

References

[1] Eli Sternheim, Rajvir Singh, Rajeev Madhavan, and Yatin Trivedi, *Digital Design and Synthesis with Verilog HDL*, Automata Publishing Company, 1993.

[2] NaveedSherwani,*Algorithms for VLSI Physical Design Automation, SECOND EDITION,* Kluwer Academic Publishers, 1997.

[3] R. Jacob Baker, Harry W. Li, and David E. Boyce, *CMOS, Circuit Design, Layout, and Simulation,* IEEE Press Series on Microelectronic Systems, 1998.

Chapter 3

PLACEMENT

"The difficulty lies, not in the new ideas, but in escaping the old ones, which ramify, for those brought up as most of us have been, into every corner of our mind." John Maynard Keynes

Standard cell placement and insertion of buffers along clock paths or Clock Tree Synthesis (CTS) are the most important and challenging phases in ASIC physical design.

The goal of standard cell placement is to map ASIC components, or cells, onto positions of the ASIC core area, or standard cell placement region, which is defined by rows. The standard cells must be placed in the assigned region (i.e. rows) such that the ASIC can be routed efficiently and the overall timing requirements can be satisfied. Standard cell placement of an ASIC physical design has always been a key factor for achieving physical designs with optimized area usage, routing congestion, and timing behavior. Almost all of today's physical design tools use various algorithms to place standard cells automatically. Although these placement algorithms are very complex and are being improved frequently, the basic idea has remained the same.

In the early days of physical design, the total area for placing standard cells consisted of the area required for the standard cell rows and the area required for channel routing. With advancement in place-and-route tools, standard cell channel routing has almost vanished because all place-and-route tools today are capable of routing over the standard cells. Over-standard-cell routing utilizes all empty space above the standard cells. This allows physical designers to create an ASIC that is as compact as possible without creating extra channels for routing purposes.

With the disappearance of routing channels, the routing congestion problem has become more important. During standard cell placement, excessive congestion resulting in a local shortage of routing resources must be avoided. In over-standard-cell routing , the objective of most place-and-route tools has been to utilize all the available core area to prevent routing overflow. This routing overflow accounts for an increase in ASIC device size and results in performance degradation.

Standard cell placement may be thought of as an automatic process that requires less physical designer intervention. However, a number of design constraints that can be applied during standard cell placement to achieve optimal ASIC design with respect to area, performance, and power. These constraints can be congestion, timing, power, or any combination thereof.

Most place-and-route tools use a two-step approach to place standard cell instances. These steps are global and detail placement. The objective of the global placement algorithm is to minimize the interconnect wire lengths, whereas the objective of the detail placement algorithm is to meet design constraints such as timing and/or congestion, and to finalize the standard cell placement.

3.1 Global Placement

When the floorplan is first created, standard cells are in a floating state. This means that they are placed arbitrarily in the ASIC core and have not been assigned to a fixed location within the standard cell rows. At this time one can partition the standard cell area and assign a group of cells to these partitions, or simply group a set of standard cells.

Almost every place-and-route tool supports cluster and region options. These two options are used to guide placement algorithms during standard cell placement.

Cluster refers to a group of standard cells that, during placement, are placed near each other. The location of the cluster is undefined until all standard cells have been placed. This option is mainly used to control the closeness of timing-critical components during placement and resembles a module definition in the structural netlist. Since the development of placement algorithms (e.g. interconnect driven), this option has been rarely used – except in very special cases. An example of a standard cell cluster is shown in Figure 3-1.

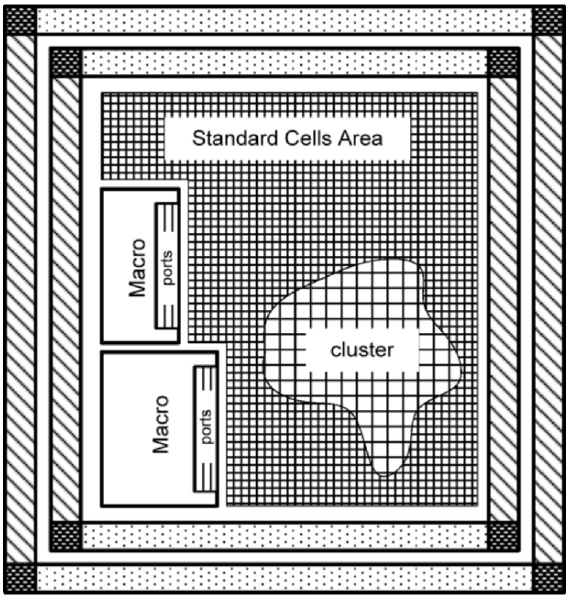


Figure 3-1 Standard Cell Cluster

Region is very similar to the cluster method with the exception that the location of the region is defined prior to standard cell placement. The way this option is implemented is that a cluster or group of standard cells is created and then assigned to a particular area on the core ASIC.

Regions can be soft or hard. Soft region, is physical constraint where logical module is assigned to a location in the core and boundary of the region, it is subject to change during standard cell placement. Hard region, however, is more rigorous than the soft region and defines a physical partition for modular design. It has “hard” boundaries that prevent standard cell crossing during placement. Using the hard region option, one must define the location as well as the shape of the region. This option is used primarily for timing related issues such as grouping clock, voltage, or threshold voltage domains.

In addition, a region can be exclusive or non-exclusive. An exclusive region only allows standard cells assigned to the region to be placed within the region. On the other hand, a non-exclusive region will allow standard cells that do not belong to the region to be placed within it. A hard region (or an exclusive region with a predefined physical boundary) might be used to enforce a floorplan consisting of separate blocks. This approach is useful for dividing the ASIC core area into regions that have different functions or physical aspects.

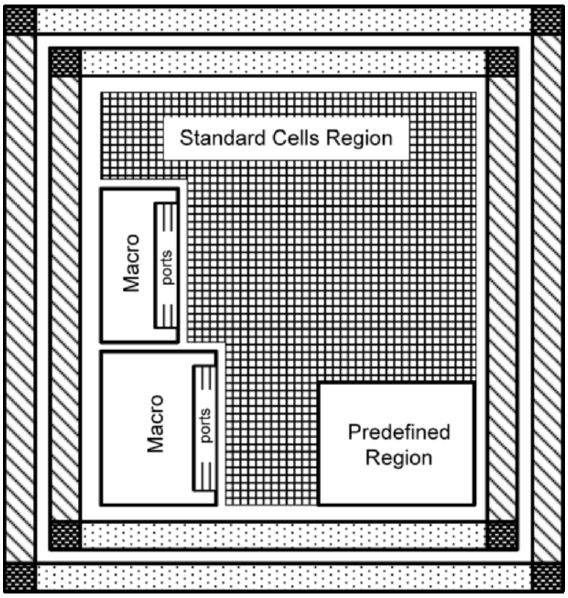


Figure 3-2 Predefined Region

For example, a hard region can be used to partition the ASIC core area so one region has a different voltage from the rest of the design or other regions as shown in Figure 3-2.

After clusters and regions are defined, global placement algorithms begin distributing standard cell instances uniformly across the available ASIC core and use a method of estimation to minimize wire lengths.

During this time, ASIC design is recursively partitioned along alternatively horizontal and vertical cut lines, and standard cell instances are assigned to rectangular bins, or slots, taking partitioning into account. Then, instances in each bin will be moved across each cut line in order to minimize the number of connections between each partition.

The procedure of partitioning and moving standard cell instances across cut lines terminates when certain stop criteria are satisfied (e.g. total number of standard cell instances in each bin). After design partitioning is completed, a legalization step is executed to remove any standard cell overlap and fit current placement into row structures.

These types of global placement algorithms are classified as partition-based. There are two main cost functions associated with partition-based algorithms: to reduce the total wiring or routing length and to distribute the standard cell instances homogeneously in the ASIC core area such that optimal equilibrium among vertical and horizontal routing is achieved.

There are three methods widely used for partitioning an ASIC design during global placement. These methods, which are also known as min-cut algorithms, [1] are:

- Quadrature
- Bisection
- Slice and Bisection

Quadrature placement alternatively partitions the ASIC core area into an equal number of instances in a vertical and horizontal direction, and minimizes the number of interconnections or cut-sizes in each direction. The vertical and horizontal cut lines start from the center of the core and recursively continue partitioning standard cell instances until the number of cut-sizes are minimized and no more horizontal or vertical partitions are possible.

The quadrature algorithm results in a set of blocks, or quadrants, with few standard cell instances in each one. In this way, a very simple placement algorithm can legalize these standard cell instances in each quadrant.

One of the advantages of the quadrant-based placement method is that it produces equilibrium between horizontal and vertical routing without a congestion area. For this reason, most of the place-and-route tools utilize this algorithm during their initial or global placement.

Figure 3-3 shows partitioning of N standard cells using quadrature min-cut.

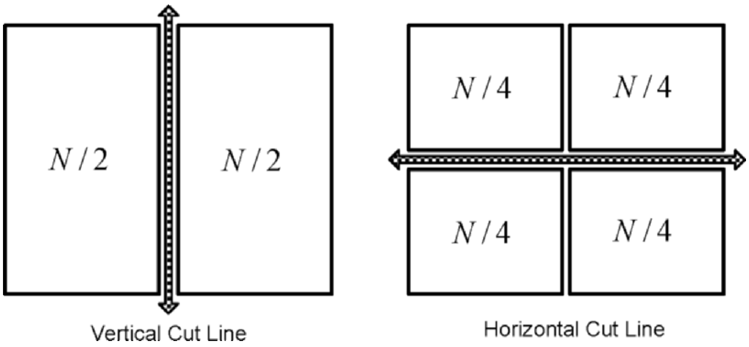


Figure 3-3 Quadrature Min-Cut

Bisection placement repeatedly divides or bisects the design by vertical or horizontal cut lines depending on the standard cell row orientation until the bins or slots contain one or two rows. The position of each standard cell instance will be arbitrary at this point.

To legalize the position of each standard cell instance within each row, the standard cell rows are bisected by vertical or horizontal cut lines recursively to fix or optimize the position of each standard cell. This does not necessarily minimize the number of cut-sizes between the partitions.

Figure 3-4 shows the partitioning of N standard cells using horizontal bisection.

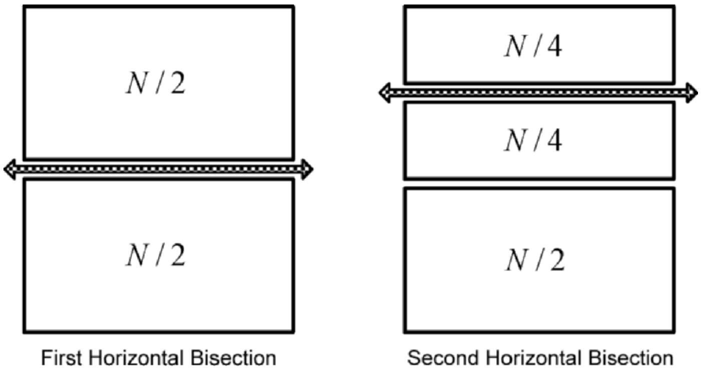


Figure 3-4 Bisection Min-Cut

The slice and bisection algorithm divides N standard cell instances such that the smaller partition N/k can be assigned to a standard cell row by horizontal slicing.

Equation 3.1.1 shows the basic principle of slice and bisection partitioning of N standard cell instances into k groups.

$$\frac{(k-1)N}{k} + \frac{N}{k} = N \tag{3.1.1}$$

The slice and bisection procedure is repeated until all standard cell instances have been assigned to rows. Once all standard cells are assigned to rows, recursive vertical bisecting cut lines are applied such that the interconnection across each cut line is minimized by moving standard cells by columns.

During the bisecting procedure, the placement of all standard cell instances is legalized and cells that overlap are removed.

The basic concept of this procedure for partitioning of N standard cells is shown in Figure 3-5.

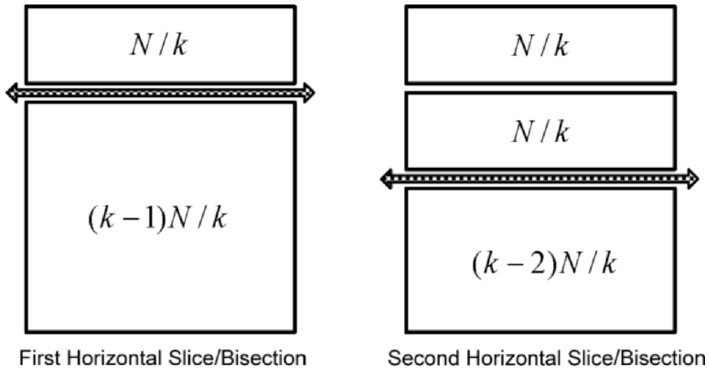

N/k
$(k-1)N/k$
First Horizontal Slice/Bisection
N/k
N/k
$(k-2)N/k$
Second Horizontal Slice/Bisection

Figure 3-5 Slice and Bisection Min-Cut

Apart from partitioning-based placement algorithms, there are other placement algorithms such as clustering, simulation, stochastic evolution, and analytical-based.

Among analytical-based algorithms, quadratic placement techniques [3, 4] have gained wide attraction over the past decade due to their efficiency of handling very large ASIC designs.

The basic concept of quadratic placement is to solve global placement based on two assumptions–one, that all the standard cell instances are point instances and, two, all nets associated with them are two-point nets (i.e. all multiple connections will be preprocessed to form two-pin connections). Thus generation of sparse systems of linear equations, where each system represents a one-dimensional placement problem with a minimum squared wire length objective, is achieved.

Using the minimum squared wire length, the cost function of the distance between instance i and instance j in terms of wire length is

$$\Phi(x,y) = \frac{1}{2}\sum_{i,j=1}^{n} c_{ij}[(x_i - x_j)^2 + (y_i - y_j)^2], \qquad (3.1.2)$$

where (x_i, y_i) and (x_j, y_j) represent the coordinates of instances i and j.

In terms of linear systems, the idea is to introduce $C = [c_{ij}]$ as a symmetric connectivity matrix and a modified connectivity matrix B such that

$$B = D - C\,, \tag{3.1.3}$$

where D is a diagonal matrix with $d_{ii} = \sum_{j=1}^{n} c_{ij}$.

Utilizing a modified connectivity matrix, the cost function of moveable standard cell instances can be computed by

$$\Phi(x, y) = x^T Bx + y^T By\,, \tag{3.1.4}$$

where $x^T = [x_1, x_2, ..., x_n]$ and $y^T = [y_1, y_2, ..., y_n]$.

Although this is a two-dimensional equation, due to the symmetrical and independency between x and y, only the one-dimensional problem needs to be considered (i.e. one placement in X and one placement in Y direction will induce a two-dimensional placement).

It is interesting to note that the main problem with the quadratic algorithm is that it creates a very high cost for long wires and very low cost for short wires. Thus, a highly connected module, or cluster, spread out over the ASIC core area has a tendency to increase wiring congestion, and may reduce the routing resource flexibility required to meet timing or other design constraints.

As mentioned earlier, the global placement algorithm defines the initial, or preliminary, standard cell instance location. During this time, one can use buffering optimization to overcome problems associated with high fan-out nets, long wires, and logic restructuring if the initial timing indicates that the critical timing paths have large timing violation.

In addition, if the ASIC design uses scan testing methodology, the original data output port of a register that connects to the next scan data input port of a register may be reordered, depending upon their locations.

The scan reordering improves the routing congestion if two connected registers are located far from each other. Because this scan-chain reordering is purely connectivity based, it can lead to hold violations at some of the scan data input ports with respect to the input clock. These types of violations are usually resolved after clock tree synthesis.

The problem with nets that have a large number of fan-outs, such as a reset signal, is that one source drives many standard cells across the ASIC core. Although these nets are not critical from a timing perspective, they have a strong impact on the core routing area due to their global nature. Thus, reducing the total number of these fan-outs will improve the overall routing of the ASIC core area. Reducing the number of fan-outs (or connections) to between 40 and 50, to which one standard cell connects, is reasonable.

Long wires are not the same as high fan-out nets as they are not global in nature. They have very small fan-outs, but the driver instance is located far from the receiver. Often this situation is a result of the receiver having very strong connectivity to instances other than the driver.

These types of long wires are highly resistive and can create large input transition times at the input port of the receiver cell. This large input transition time increases the receiver cell's propagation delay. Therefore, it is beneficial to segment these long wires using buffers.

Another timing optimization that is used during global placement is logic restructuring. Logic restructuring is mainly supported by physical synthesis tools that combine several primary logic functions into a few standard cells, decompose functional gates into their equivalent primary logic gates, and/or duplicate combinational logic (cloning).

The logic restructuring algorithm used during global placement mainly focuses on logic reconstruction of critical paths that are not meeting required timing constraints. The objective of the algorithm is to rearrange logic in the critical paths such that the timing constraints are met.

3.2 Detail Placement

Once all standard cell instances are placed globally, a detail placement algorithm is executed to refine their placement based on congestion, timing, and/or power requirements.

Congestion refinement or congestion-driven placement is more beneficial to ASIC designs with very high density, and the objective of the detail placer is to distance standard cell instances from each other such that more routing tracks are created among them.

The quality of congestion placement directly relates to how well the global placer partitions the design and could have a negative impact on the device size and performance.

For minimal device size, one may use more routing layers. In determining the total number of routing layers to use, it is imperative to consider the trade-off between increasing the device size and using extra routing layers. In some instances, it may be more economical to increase the device size rather than adding extra routing layers (i.e. extra mask).

Timing-driven placement algorithms have been classified as either net or path based. Net-based schemes try to control the delay on a signal path by imposing an upper-bound delay or by assigning a weight to each net. Path-based approaches apply constraints to delay paths of small subcircuits (the disadvantage of path-based algorithms is the fact that it is impossible to enumerate all paths within a design).

The major challenge in timing-driven placement is to optimize large sets of path delays without enumerating them in the ASIC design. This optimization is accomplished by interleaving weighted connectivity-driven placements with timing analysis that annotates individual instances, nets, and path delays with design constraint information.

To meet these types of design constraint, various placement techniques have been proposed or used. The most well-known detail placement method is simulated annealing. Not only is simulated annealing efficient, it can also handle complex design constraints.

Simulated annealing is a simulation-based placement technique and is used as an iterative improvement algorithm during the detail placement process. The objective of this procedure is to find an optimal or near-optimal placement for each pre-placed standard cell instance.

This method resembles material science for producing a solid in a low-energy state. For example, to produce a perfect crystal in a low-energy state, material is heated to a high temperature and then slowly cooled down to form a solid. This process of solidification is based on the assumption that at any given temperature, the probability distribution of a change in structure is caused by the energy level that is governed by the Boltzmann distribution theorem (i.e. $\exp(-E/kT)$). When the temperature is high, the system can change radically and many changes that do not lower the energy level are allowed. As the temperature decreases, fewer and fewer non-optimal changes are permitted until an optimal state is achieved.

From a computational point of view, a concept similar to solidification theory (simulated annealing) is applied to placement optimization problems. In this process, a cost function ΔC is used in place of energy, and temperature T is used as the control parameter to iteratively improve the initial pre-placed standard cell instances. The initial pre-placed standard cell instances are chosen and their placement is iteratively optimized.

During the optimization process, all configuration changes that do not increase cost are accepted as in any iterative improvement procedure. The physical presentation of the Boltzmann distribution ($\exp(-\Delta C/T)$) is used as the annealing element to determine the acceptance criteria for configurations with increased cost. The acceptance probability can be given by

$$\begin{aligned} & P = 1 && \text{if} \quad \Delta C \leq 0 \\ \text{and} \\ & P = \exp(-\Delta C/T) && \text{if} \quad \Delta C > 0\,. \end{aligned} \tag{3.2.1}$$

The simulated annealing algorithm begins with a very high temperature for the initial standard cell placement. The temperature slowly reduces according to an annealing schedule so that increases in cost have a progressively lower probability of acceptance.

As the temperature decreases, only the moves that reduce the cost will be accepted. Therefore, fewer non-optimal placements can be made. Fundamentally, a higher initial temperature setting, or cost function, will cause a higher number of moves or trials at the beginning of the process and, in turn, will cause longer optimization time.

Figure 3-6 demonstrates cost versus the total number of trials or moves.

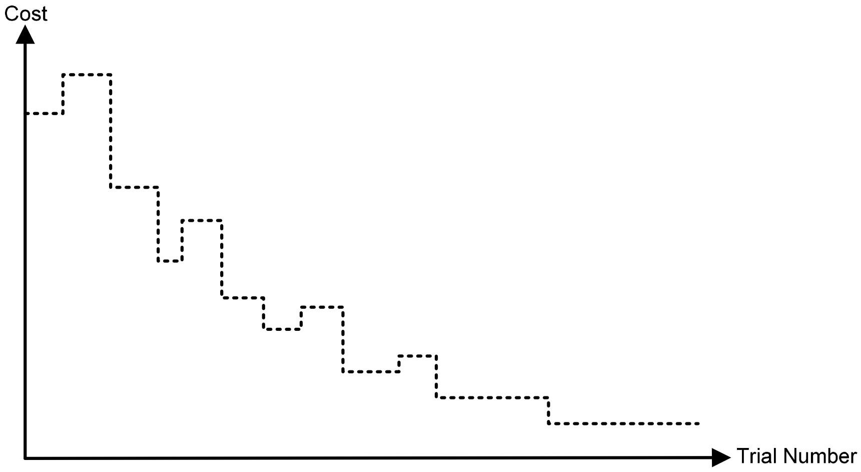


Figure 3-6 Conceptual Simulated Annealing

Simulated annealing optimization can be for timing, congestion, or power. During the optimization process, one of the optimization criteria is chosen and applied to the initial or global placement.

A cost function is used to indicate the desirability of a given placement, and the changes in cost from current placement to new placement are computed. These optimization processes are iteratively repeated until all criteria are met or a specified run time is exceeded.

Timing optimization is one of the most challenging areas of any physical design. This optimization can be load-based or gain-based.

Load-based optimization has been used for some time. During load-based optimization, the wire loads, or interconnect capacitance, are approximated using a wire capacitance model. Then the optimization algorithm makes a decision about the drive strength of the standard cell (i.e. up or down sizing) based on an estimated load that is predicted by the wire capacitance model to meet required timing constraints.

In the early days of physical design, this method was adequate because the intrinsic delays of the cells were dominant. In addition, the effect of the capacitive load on the extrinsic delay was small in comparison and the wire resistance was low due to the large line width. Thus, a wire model could estimate the capacitive loading effect accurately.

With increased design size, unfortunately, these estimated wire load models no longer can predict the actual wire lengths until the physical design is complete. If the logic synthesis tools use the estimated wire load model to select cell drive strength before the actual wire loading is known, a non-terminating iterative process of changing device sizes during timing closure may occur.

Figure 3-7 illustrates a non-terminating iterative process of device up and down sizing during pre-layout and post-layout phases based on the estimated versus actual capacitance loading.

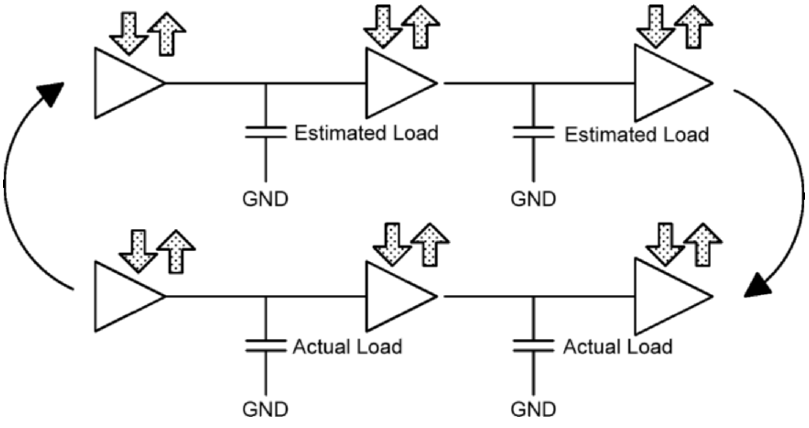


Figure 3-7 Iterative Devices Up and Down Sizing

With larger ASIC sizes and the scaling down of the transistor features, the timing effect of capacitive loading has become more dominant, and wire resistance due to the small line width cannot be neglected. The real wire load differs dramatically from the wire load approximation, making the use of the wire capacitance model inaccurate.

This difference in the wire model and actual wire load (e.g. after routing) can lead to an endless iterative process during ASIC timing closure. To overcome this iterative process, the use of global routing during load-based optimization is suggested, as long as the initial global and final detail routes correlate closely.

For deep submicron and predictability issues associated with wire capacitance models, gain-based optimization is becoming widely used due to the acceptance of the load-independent cell delay [4] concept.

The idea of gain-based optimization relies on the concept of logical effort. The method of logical effort simply reformulates a classical CMOS delay model, and expresses it with some new terminology.

As explained in Chapter 1, the CMOS delay model contains two parameters: intrinsic (which is constant) and extrinsic (which is proportional to the output capacitance load). In the logical effort model, the intrinsic value is known as the parasitic delay, and the extrinsic value is referred to as the effort delay. Using parasitic delay and effort delay, then the total delay through a CMOS gate d is given by

$$d = p + f\,, \tag{3.2.2}$$

where p is parasitic delay and f is effort delay. Both are measured in units of τ. The τ value is based on the characterization of the semiconductor process used for a given ASIC design.

The effort delay can be viewed as having an electrical effect on the output capacitance load as well as having a logical effect that is the ability to drive the capacitive load. Thus, the effort delay is expressed as

$$f = gh, \tag{3.2.3}$$

where g is logical effort and h is electrical effort.

The logical effort determines the standard cell's capability of producing the output current based on its topology and is independent of the size of transistor used in its circuit. In other words, logical effort describes the delay in terms of complexity rather than transistor size. Thus, the more complex the standard cell, or gate, is, the slower it becomes.

Logical effort is defined by the ratio of the total number of transistor units in a standard cell circuit to the total number of transistor units in an inverter circuit (the smallest unit in the library). The ratio is expressed as

$$g = \frac{T_{gate}}{T_{inverter}}, \tag{3.2.4}$$

where T_{gate} is the total number of standard cell transistors and $T_{inverterv}$ is the total number of transistors in an inverter circuit. In a typical inverter with an equal rise and fall delay time with two units of PMOS and one unit of NMOS transistor, the value of $T_{inverterv}$ and T_{gate} is equal to three and is considered to have a logical effort equal to one.

The electrical effort (or gain) captures the electrical behavior and performance of the standard cell associated with its output capacitance load. Essentially, this parameter represents the standard cell's load-driving capability and is given by

$$h = \frac{C_l}{C_i}, \tag{3.2.5}$$

where C_l is the output capacitance and C_i is the input capacitance.

Substituting Equation 3.2.3 into Equation 3.2.2, then the delay through the standard cell d in units of τ is

$$d = gh + p\,. \tag{3.2.6}$$

In Equation 3.2.6, g, h, and p parameters contribute to standard cell delays separately. In the standard cell delay calculation, parameters g and p are independent of the transistor size, while parameter h directly relates to the transistor size. Typically, standard cell delays are measured using τ parameter. The τ parameter is simply the delay through the smallest inverter for a given standard cell library. Typically, an inverter chain is used to measure the value of τ.

During gain-based optimization, the gain for each standard cell along the critical path is computed. Then the algorithm tries to maintain the gains within a given timing path (equal gain for each component in the timing path means optimal timing). In this process if an instance requires more gain due to an increase in its output capacitance loading, then its input capacitance will be increased in order to maintain the original gain. This technique of changing the input capacitance rather than changing the underlying standard cell drive strength exhibits wire capacitance load independency.

Apart from congestion and/or timing optimization, one can consider power minimization during detail placement or after the clock tree synthesis. There are two main sources of power consumption for an ASIC design—dynamic and static.

The dynamic power is given by

$$P_d = V^2 \sum_{i=1}^{N} f_i C_i\,, \tag{3.2.7}$$

where V is the supply voltage and f_i and C_i are the frequency and loading capacitance for each node in the design.

For dynamic power optimization, as suggested by Equation 3.2.7, one can reduce the overall design supply voltage and/or the nodal loading capacitance. Reduction of supply voltage is accomplished by selecting CMOS processes that require less operational power. The overall reduction of nodal capacitance is achieved by limiting the maximum allowable load capacitance during standard cell placement. One should note that limiting maximum allowable load capacitance has negative impact on area as a result of excess buffer insertion.

Another source of dynamic power consumption can be caused by signal transition time at the input of standard cells. This is due to the fact that when a CMOS gate is switched by an input signal (rise and fall) and if the transition time is long enough, it can introduce a condition where both NMOS and PMOS transistors will conduct simultaneously for a short period. When both NMOS and PMOS transistors are on at the same time, there will be a direct path from supply to ground through which current will flow without any contribution to the actual operation of the gate (i.e. charge and discharging the load capacitance), thus causing current leakage. This power consumption is due to the short circuit and can be approximated by

$$P_t = I^2 (R_p + R_n)\,, \tag{3.2.8}$$

where P_t is transition power consumption, I is the supply current and R_p and R_n are the PMOS and NMOS gate resistance.

One method used to reduce the transition power consumption is to control the input maximum transitions during standard cell placement or to specify a maximum allowable transition value for each individual cell in the library.

The next contribution to ASIC power consumption is due to current leakage inherent to standard cells and is given by

$$P_s = V \sum_{j=1}^{N} I_j\,, \tag{3.2.9}$$

where P_s is the total static power consumption, V is the supply voltage and I_j is the leakage current for each component in the design. All standard cells are characterized for their static power consumption and their value is stated in the standard cell library.

Although the static power consumption is very low for designs for mobile applications, this type of optimization for minimizing static power is becoming a primary concern in ASIC physical design in order to extend the operational power period.

Static power optimization is accomplished by replacing the lower threshold standard cells along all non-critical paths in an ASIC design with higher threshold voltage ones (the higher the threshold, the lower the current leakage).

In general, excess static power consumption presents a limiting factor in high-performance CMOS design–especially in deep-submicron technology where the processes inherently have a large amount of current leakage (e.g. smaller transistor features and thinner gate oxide). Thus, it is important to note that after low-power technology mapping, if the placement algorithm ignores power dissipation due to current leakage of an ASIC device, the effort for low power is insignificant. Therefore, it is important having placement algorithm capable of optimizing the power dissipation especially for today's technology trend that indicate the current leakage increase rapidly.

3.3 Clock Tree Synthesis

The concept of clock tree synthesis (CTS) is the automatic insertion of buffers/inverters along the clock paths of the ASIC design to balance the clock delay to all clock inputs. Naturally, clock signals are considered global (or ideal) nets.

These nets exhibit high resistance and capacitance due to their being very long wires. The principle of CTS is to reduce the RC delay associated with these long wires. These long wires can be modeled as distributed networks such that

$$C\frac{dV}{dt} = \frac{(V_{i-1} - V_i)}{R} - \frac{(V_i - V_{i+1})}{R}, \qquad (3.3.1)$$

where V is voltage at a point i in the wire, and R and C are resistance and capacitance of each wire segment as shown in Figure 3-8.

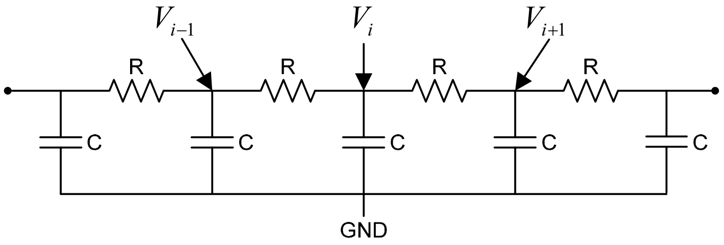


Figure 3-8 Distributed RC Network

As the number of wire segments increase and each wire segment becomes smaller, the differential equation 3.3.1 reduces to a well-known diffusion equation ($t = kx^2$) with distance x from input signal source [5]:

$$rc\frac{\partial V}{\partial t} = \frac{\partial^2 V}{\partial x^2}. \tag{3.3.2}$$

Using discrete analysis of this distributed RC network, the signal delay for wire segment of n can be approximated by

$$t_n = \frac{RCn(n+1)}{2}. \tag{3.3.3}$$

As the number of segments increases, Equation 3.3.3 reduces to

$$t = \frac{rcL^2}{2}, \tag{3.3.4}$$

where L is the length of the wire, and r and c are resistance and capacitance per unit length.

The L^2 term in Equation 3.3.4 shows that the propagation delay of long wire tends to be dominated by the RC effect.

One method to reduce the RC effect is to insert intermediate buffers or repeaters along the wire. Since interconnect is segmented into N equal sections, the wire propagation will be reduced quadratically, which is sufficient to offset the extra buffer delay that is introduced by repeaters in Equation 3.3.5:

$$t = rc(\frac{L}{N})^2 + (N-1)t_b, \tag{3.3.5}$$

where t_b is the propagation delay of the repeaters. To obtain the optimal number of buffers or repeaters, one can let

$$\frac{\partial t}{\partial N} = 0 \tag{3.3.6}$$

and solve for N,

$$N = L\sqrt{\frac{rc}{t_b}}. \tag{3.3.7}$$

In practice, the actual size of these tapering buffers may not be identical. In fact, to obtain optimal propagation delay, these buffers may be selected to monotonically increase in their drive strength d by a factor α for each level of clock tree path, as shown in Figure 3-9.

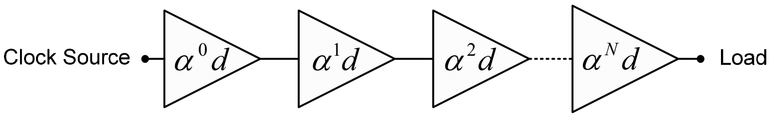


Figure 3-9 Cascaded Buffers

It is recommended for buffers that are used for tapering the clock paths to have equal rise and fall delay time. The main reason for having equal rise and fall time is to maintain the original duty cycle and to insure that there is no clock signal overlap due to any difference in propagation delays.

This overlapping clock signal becomes important when dealing with very high-speed ASIC designs. These types of buffers are known as clock or balance buffers and have different attributes to the normal buffers in the standard cell library.

The proper usage of balance buffers or inverters during clock tree synthesis is extremely important, especially when dealing with very high-speed clocking (i.e. clock pulse with small period) requirements.

In dealing with very high-speed clocking, if the clock buffers or inverters are not selected correctly they may cause the clock pulse width to degrade as the clock propagates through them before reaching the final destination.

Figure 3-10 illustrates typical clock pulse width degradation.

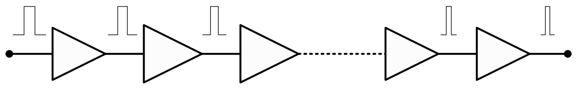


Figure 3-10 Clock Pulse Width Degradation Effect

Often it is difficult to achieve perfect signal rise and fall time balance in the context of an ASIC design during clock tree synthesis. Thus, one remedy to overcome clock pulse narrowing is to use inverters rather than buffers.

Most clock tree synthesis algorithms use either the Sum ($\sum$) or Pi ($\prod$) configuration during the clock buffer insertion along each clock path.

In the Sum configuration, the total number of inserted buffers is the sum of all buffers per level. This type of structure exhibits unbalanced trees due to the different number of buffers and wire lengths. In this method, the systematic skew is minimized through delay matching along each clock path and, due to its non-symmetrical nature, is highly process-corner-dependent.

Figure 3-11 shows a Sum configuration clock tree.

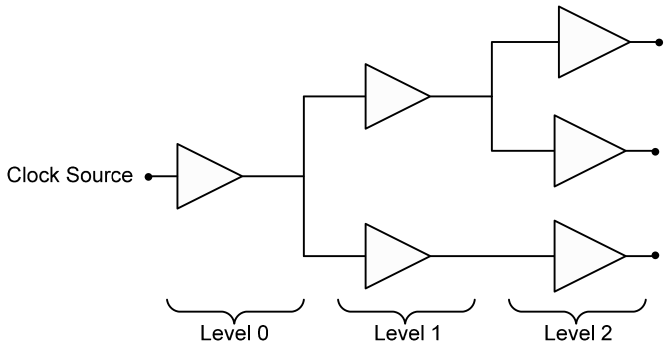


Figure 3-11 Sum ($\sum$) Configuration

In this configuration, the total number of clock buffers is given by

$$N_{total} = n_{level0} + n_{level1} + n_{level2} + ... + n_{leve\ln} . \qquad (3.3.8)$$

In the Pi configuration, the total number of buffers inserted along clock paths is a multiple of the previous level. This type of structure uses the same number of buffers and geometrical wires and relies on matching the delay components at each level of the clock tree.

The Pi structure clock tree is considered to be balanced, or symmetrical, and the total number of buffers is given by

$$N_{total} = (n_{level0} \times n_{level1}) + (n_{level1} \times n_{level2}) + .. + (n_{level(n-1)} \times n_{leve\ln}) . \quad (3.3.9)$$

In contrast to the Sum configuration, in a Pi configuration, the systematic skew is minimized due to symmetry, and the variation in clock skew is determined by process uniformity rather than by process corner.

Figure 3-12 shows a Pi configuration clock tree.

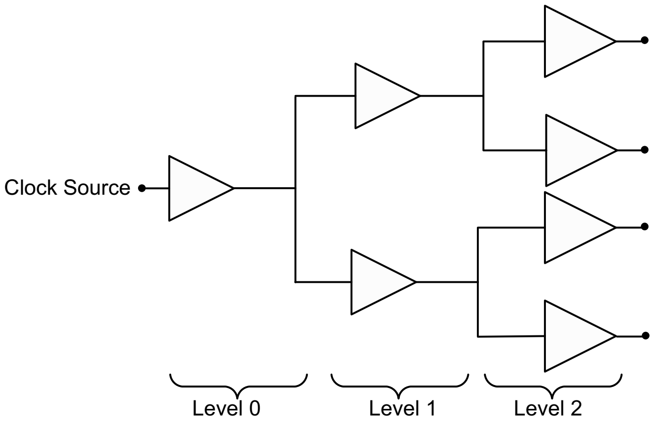


Figure 3-12 Pi ($\prod$) Configuration

In today's ASIC designs where dynamic power consumption plays an important role, Sum configurations are most often used regardless of the number of clock domains. This is because the total number of buffer insertions during clock tree synthesis is less than the total number of buffers used in the Pi configuration.

Regardless of which configuration is used to insert buffers during clock tree synthesis, the objective that governs the clock tree quality is achieving minimal skew while maintaining an acceptable clock signal propagation delay. The difference of the leaf registers' clock or skew with respect to the source of clock can be expressed as

$$\delta = t_2 - t_1, \qquad (3.3.10)$$

where δ is the clock skew between two leaf registers (leaf registers are connected at the last level of clock tree) with the propagation delay clock of t_1 and t_2 along two different clock paths, as shown in Figure 3-13.

For an ASIC design to perform properly, the following condition must be satisfied:

$$\delta \le t_l . \tag{3.3.11}$$

To prevent any erroneous result related to computational time t_l at **Register2** in Figure 3-13, the lower bound value for source clock is given by

$$T \ge t_l - \delta , \tag{3.3.12}$$

where T is the clock source period.

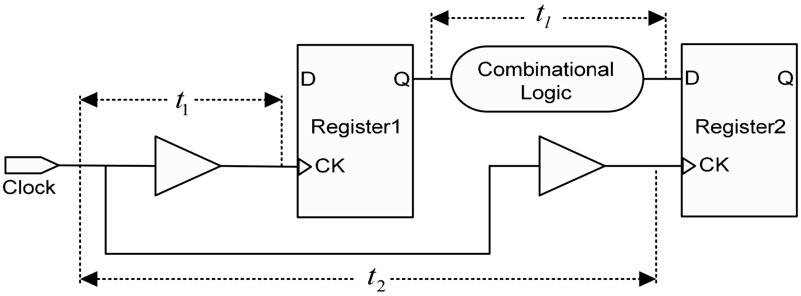


Figure 3-13 Skew Between Two Registers' Clock Ports

Clock skew δ can be local in the case of multiple clock domains or global in the case of a single clock domain. The root cause of clock skew can be systematic due to non-uniformity of clock routing and/or randomly caused by differences in delay components.

The variation in delay components is mainly due to the different device sizes, process gradients, etching effects, supply voltage variation, temperature gradients, or mismatched minimum transistor channel lengths. Any one of these effects can influence the clock propagation delay along each clock path.

In the past, variation in component delay was observed from lot-to-lot (a set of wafer undergoes the process at the same time). As ASIC manufacturing processes advanced, variations became apparent in both wafer-to-wafer and die-to-die. With the current deep submicron process, variation in component delays can also be seen on a single die as on-chip variation, or OCV.

With increase in clock complexity and depth, the need to consider clock tree synthesis to be correct-by-construction during design implementation, and to include the effect of on-chip variations, increases.

One of the common issues that can be affected by on-chip variation on a clock tree is that if the clock tree synthesis algorithm branches the clock paths near the clock source rather than close to the leaf cells, the clock can be skewed.

This means that a CTS algorithm must be able to use a common path during buffer insertion as much as possible. In this situation, the delay differences along each clock path will be local and the delay through the common path will not contribute to clock skews due to on-chip variation.

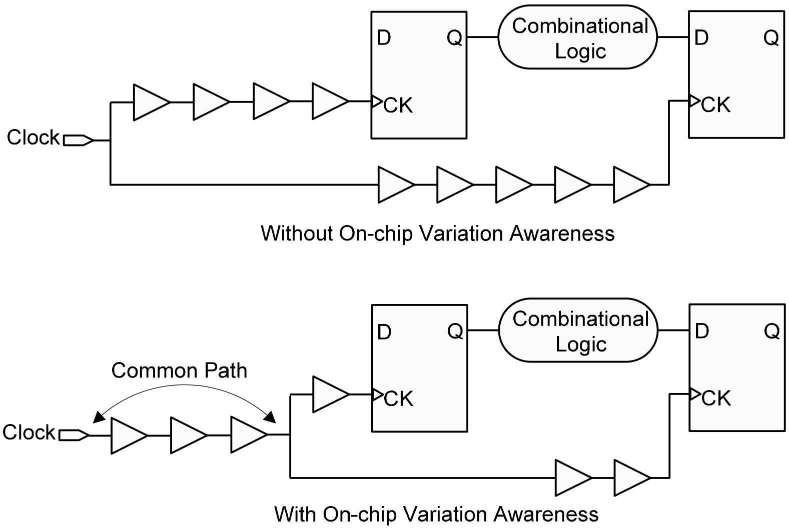


Figure 3-14 Clock Trees With and Without On-chip Variation

The common path that is introduced during clock tree synthesis results in a pessimistic delay that is not desired during the different delay calculations.

Thus, the common path that is also called Common Path Pessimism, or CPP, must be removed during the clock path delay analysis. Figure 3-14 illustrates two different clock trees, one without OCV and one with OCV awareness.

The clock skew can be positive or negative depending upon the routing direction with respect to clock source and data flow.

With positive clock skew, the clock is routed in the same direction as the data flow and increases the performance of any given ASIC design with tighter clock skew constraints.

With negative clock skew, the clock is routed or constructed opposite to the data direction and virtually eliminates any clock skew requirement. Although this clock skew elimination has a tendency to improve sequential elements, hold time within the design as a result may degrade the ASIC design's overall performance.

By default, most CTS algorithms operate based on positive clock skew. To achieve negative clock skew one can set clock delay or latency to a negative number. Setting negative clock latency causes leaf registers to be connected to lower levels (i.e. closer to clock source) of the clock tree.

In practice, most CTS algorithms first identify the leaf registers or sink points (e.g. non-clocking ports of standard cells that are not defined as clock ports) by creating a virtual cluster.

Virtual clustering is accomplished by identifying the location of the leaf cells which are in close proximity to each other. If there are leaf cells that are far from any cluster, they will be moved to the nearest cluster.

The number of leaf cells per cluster is user-defined. Once the clusters and their locations are determined, buffer insertions begin such that the clock propagation delay is equal to each cluster, and clock skew within each cluster is minimized.

Figure 3-15 illustrates the basic concept of virtual leaf cell clustering.

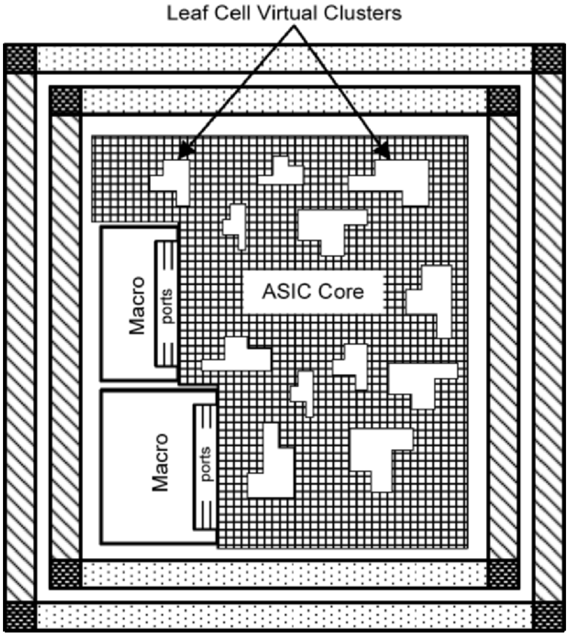


Figure 3-15 Leaf Cell Virtual Clusters

The clock tree's level structuring can be user-defined or defined automatically depending upon physical design tools. One should note that the smaller the cluster, the less the skew, but more clock buffering levels will be required. This has a tendency to increase the overall clock propagation delay.

Upon completion of clock tree synthesis, any timing violations such as data setup and hold with respect to the clock for any register needs to be resolved. The best results are achieved when accompanied by global routing information and appropriate library setting (i.e. slow case library to fix setup and fast case library to fix hold time violations).

Another important aspect of clock tree synthesis to have in mind is power consumption. The clock distribution network accounts for 30% or more of the total dynamic power of current ASIC designs. This is because clock nets operate at the highest switching frequency of any net and often have a considerable amount of capacitive loading. Therefore, designing an optimum clock tree is very important not only for performance, but also for power consumption.

3.4 Power Analysis

With current ASIC designs targeting 130 nanometer and below, proper power analysis not only becomes very important, but also unavoidable. At this nanometer-process scaling, which leads to thinner gate oxides and creates an intense electrical field resulting in increased leakage current having direct impact on performance, it is imperative to be able to estimate and analyze the dynamic power consumption.

This type of analysis relies on knowing the operating frequency of each net and corresponding load capacitance in an ASIC design. In practice, this is very computationally intense (e.g. functional simulation of the entire ASIC to exercise every net) and is becoming almost impossible to obtain meaningful results.

To overcome this dynamic power analysis problem, one method is to perform a static power analysis based on the power distribution systems (i.e. power and ground routing) that are designed to provide needed voltages and currents to the transistors that perform the logical functions.

After clock tree synthesis, one must perform detailed analysis of the ASIC design's power networks to reduce the risk of voltage (IR) drops in power, ground bounce on the ground nets, and electromigration effects due to high current.

Many algorithms available today can perform this type of static power analysis by creating an image of the power grid robustness and the effect of voltage drops, ground rises, and electromigration effects based on anticipated ASIC total power and current consumption.

Most static power analysis algorithms are based on Ohm's and Kirchoff's laws. In these methods, the resistances of power and ground routing are extracted and a resistor network, or matrix, of these nets is built.

Once the resistor matrix is formed, then the average current for each transistor connected to the power net is calculated with the power net replaced by a constant voltage source.

Figure 3-16 illustrates the principle of constructing resistor and transistor networks.

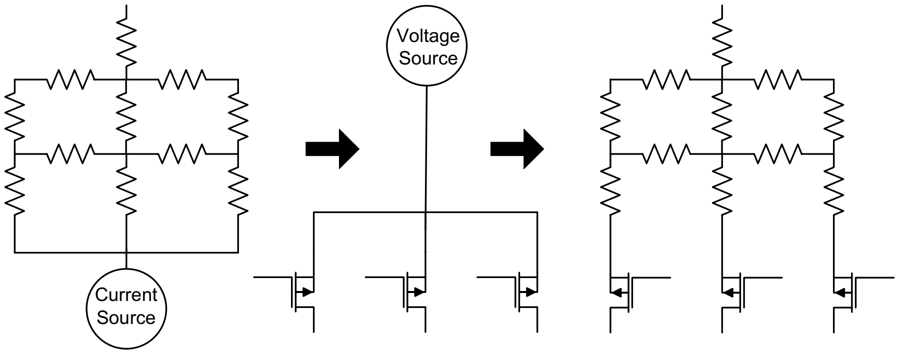


Figure 3-16 Static Power Calculation Method

Next, the average currents are distributed throughout the power net based on the location of each transistor, calculating node voltages and branch current densities using every power source (e.g. power pads). The same process is repeated for calculating the voltage rise on the ground nets.

At the completion of this process, calculated values that exceed the specified user value are collected in percent form showing design goal violations. Figure 3-17 shows conceptual results of a typical power analysis image.

Figure 3-17 illustrates that there are $V_{\max}$ and $V_{\min}$ regions. The $V_{\max}$ region corresponds to the area with the largest voltage drop, whereas the $V_{\min}$ region indicates the lowest voltage drop area of the ASIC core. For an ASIC design to perform correctly, the difference between $V_{\max}$ and $V_{\min}$ plus the voltage due to the ground rise and the external power supply variation must be less than the worst-case voltage that is defined in the standard cell library.

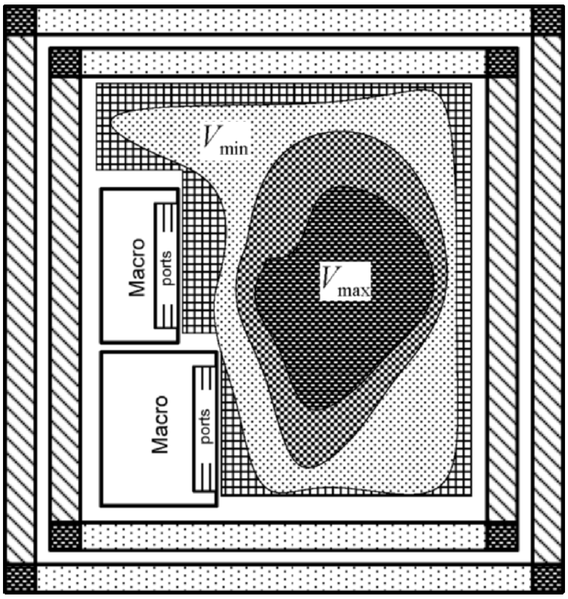


Figure 3-17 Power Analysis Image View

Most of the time, improper power and ground routing can cause the voltage (IR) drop to exceed the specified value in the library. It is critical to resolve the voltage (IR) drop before continuing on to the next stage of the physical design.

It is important to realize that the static power analysis approach approximates the effect of dynamic switching on the power network using the fact that the decoupling capacitance between power and ground smooths out the maximum voltage drops or ground rises, and localized dynamic effects are not included. Thus, to avoid over-optimistic results of the static power analysis, one must insure that there is sufficient decoupling capacitance between power and ground.

Electromigration is another important topic of power analysis. High current densities and narrow metal line widths are root causes of electromigration failure. One of the basic methods used to resolve this problem is to increase the width of the major power and ground connections (i.e. increase the width everywhere that power and ground pads are connected to core power and ground busses) and to make sure that during manual routing there are no notches on the main current-carrying power lines.

3.5 Summary

In this chapter, we discussed the fundamentals of global and detail placement as well as clock tree synthesis and the basics of static power analysis.

In the discussion of global placement, we covered some of the well-known algorithms such as quadrature, bisection, and slice/bisection.

In the detail placement section, we covered the basic principles of simulated annealing and explained load-based and gain-based placement optimization. Although some of these placement methods are considered basic algorithms, most of today's EDA developers involved in designing physical design tools use these classical techniques frequently to develop relatively new placement algorithms. However, we need to recognize that standard cell placement is one of the critical steps in any ASIC design automation flow and has captured the attention of both academic and industrial researchers. The standard cell placement flow that is discussed in this chapter may be viewed as the classical approach. The new trend in ASIC design is to combine logic synthesis with more sophisticated place-and-route algorithms (e.g. layout synthesis) which are then provided in a unified database.

In the clock tree synthesis section we have explored the basic method of systematic buffer insertion along clock paths using different styles such as Pi and Sum models. Also in this section, we explained the concept of different sources of clock skew such as on-chip process variation.

In the power analysis section, we explained the basic concepts of static power analysis and how voltage drops can affect the circuit performance.

Apart from static power analysis, we should recognize that as the performance of modern ASIC increases so does their dynamic power consumption. This increase in power consumption has a local effect on the

ASIC performance, and we should analyze this effect during physical design.

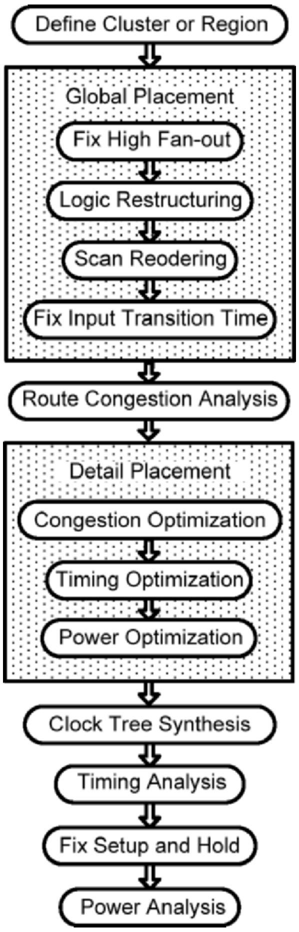


Figure 3-18 Standard Cell Placement and Clock Tree Synthesis Steps

Given the complexity of placement steps involving clock tree synthesis and power analysis, the next generation of layout synthesis tools should not only be capable of performing the place-and-route of an ASIC design, but will also need to be able to create a design-specific standard cell layout automatically and characterize them simultaneously during placement for optimal design performance and power.

Figure 3-18 shows the basic steps that may be required during the placement phase.

References

[1] M.A. Breuer, “A Class of Min-cut Placement Algorithms”, *Proceedings, Design Automation Conference*, pp. 284-290, IEEE/ACM, 1977.

[2] Naveed Sherwani, “Algorithms for VLSI Physical Design Automation”, *Kluwer Academic Publishers,*1997.

[3] R.S. Tasy and E. Kuh, “ A Unified Approach to Partitioning and Placement”, *IEEE Transactions on Circuits and Systems*, Vol. 38, No. 5, pp. 521-633, May, 1991.

[4] J. Kleinhans, G. Sigl, F. Johannes, and K. Antreich, “GORDIAN: VLSI Placement by quadratic programming and slicing optimization”, *IEEE Transactions on Computer-Aided Design*, Vol.10, pp. 356-365 March, 1991.

[4] Sutherland and R. Sproull, “The theory of logical effort: designing for speed on the back of an envelope”, *Advanced Research in VLSI,* 1991.

[5] Neil H. Weste and Kamran Eshraghian, *Principles of CMOS VLSI DESIGN, A Systems Perspective*, Addison-Wesley, 1985.

Chapter 4

ROUTING

"Every item of physical knowledge must therefore be an assertion of what has been or would be the result of carrying out a specified observational procedure." Sir Arthur Eddington

After completion of standard cell placement and power analysis, the next phase is to route the ASIC design and perform extraction of routing and parasitic parameters for the purpose of static timing analysis and simulation.

As ASIC designs are getting more complex and larger (e.g. sea of cells), routing is becoming more difficult and challenging. It is possible for routing to fail to complete, or to take an unacceptable amount of execution run time. Besides the routing algorithms, the factors which influence the routability of a given ASIC are the layout of standard cells style, a well-prepared floorplan, and the quality of standard cell placement as discussed in previous chapters.

Routing algorithms are mainly classified as channel or over-the-cell based routers. Channel-based routing was used in the early days of ASIC physical design. This was because semiconductor factories were not able to process large numbers of metal layers (e.g. two or three). Thus with a limited number of routing layers, all connections were restricted to the area between cells or around macro blocks such as memories.

Fundamentally, channel, or river-based, routers use reserved space between standard cell rows (routing channels) and feed-through (dedicated routing areas inside the standard cell layout) to perform routing between instances as shown in Figure 4-1.

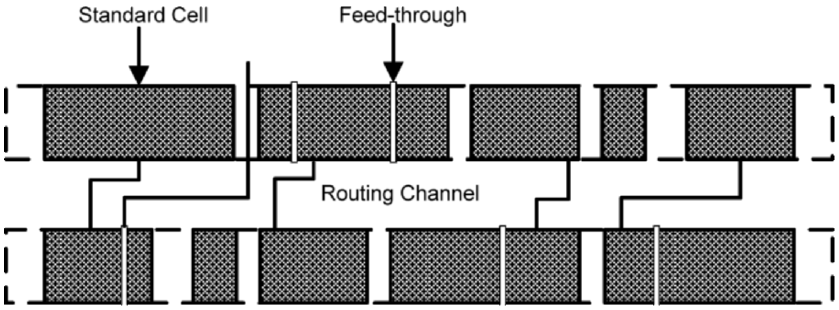


Figure 4-1 Channel Routing Method

With recent improvements in semiconductor processes and increasing numbers of routing layers, the routing channel and standard cells with feed-through have been eliminated, and over-the-cell based routing is widely utilized by many physical synthesis and place-and-route engines during the physical design of ASIC devices.

Due to the inherent complexity of ASIC designs and the very large numbers of interconnections associated with them, the overall routing is performed in three stages: special routing, global routing, and detail routing.

4.1 Special Routing

Special routing is used for standard cells, macro power, and ground connections. Most special routers use line-probe algorithms. The line-probe method uses line segments to connect standard cells, macro power, and ground ports to ASIC power and ground supplies.

Line-probe routers use a generated connectivity list of sources and targets (the generated connectivity list can be port-to-port, port-to-line, or line-to-line) for connection. The line segments are used to perform routing according to the connectivity list starting from the target and tracing the line segment until the source is reached.

These generated line segments do not pass through any obstructions. If they were to pass through an obstruction, they might create an unfixable design rule violation. Therefore, one needs to make sure that there are no obstructions where the power and ground ports are located.

In connecting macro power and ground ports to the main power and ground nets, line-probe routers use the size of the ports to set the width of the power and ground segments. This automatic width setting may not be adequate for current density considerations, and one may need to create more power and ground ports to satisfy the current density requirements.

As mentioned in Chapter 2, excessive current density in an ASIC design can lead to an electromigration problem that reduces the Mean Time To Failure (MTTF) of the device [1].

Most ASIC chips must have an MTTF of at least 10 years. Failure due to current density and electromigration of a given wire is expressed by Black's equation:

$$MTTF = \frac{A}{J^2}\exp(\frac{E_a}{kT}), \tag{4.1.1}$$

where A is the metal constant, J is the current density (i.e. the number of electrons crossing a unit area per unit time), k is the Boltzmann constant, E_a is the activation energy, and T is the temperature.

Based on Equation 4.1.1, the MTTF due to electromigration depends on two parameters–temperature and current density. During power and ground routing, one must focus on current density as a major parameter to address electromigration problems.

4.2 Global Routing

Global routing is the decomposition of ASIC design interconnections into net segments and the assignment of these net segments to regions without specifying their actual layouts. Thus, the first step of the global routing algorithm is to define routing regions or cells (i.e. a rectangular area with terminals on all sides) and calculate their corresponding routing density. These routing regions are commonly known as Global Routing Cells, or GRC, as shown in Figure 4-2.

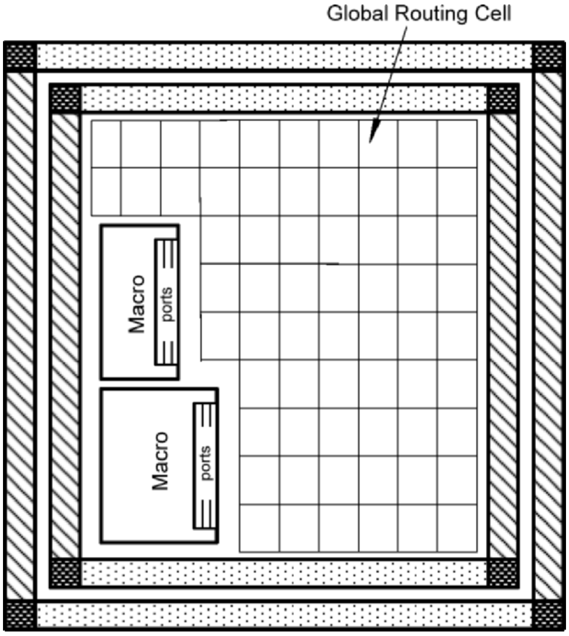


Figure 4-2 Global Routing Cells

The density, or capacity, of these cells is defined as the maximum number of nets crossing the routing regions and is a function of the number of routing layers, cell height in vertical or horizontal direction, minimum width, and spacing of wire as defined in the technology file and is given by

$$C_d = \frac{nh}{w+s}, \tag{4.2.1}$$

where C_d is the global routing cell density, n is the number of routing layers that are available for horizontal or vertical direction, h is the height of the global routing cell length, and w and s correspond to minimum wire width and spacing for vertical or horizontal directions.

Global routing uses a graph to model the interconnection networks. The vertices of the graph denote the standard cell ports. The edges of the graph correspond to connections between two ports within routing cells and among routing cells themselves. The graph is constructed by means of region assignment (i.e. the order that a region will be routed first) and assignment of each net to a pin on the boundary of the routing region.

After global routing is performed, the pin locations will be determined such that the connectivity among all standard cells in the ASIC core area is minimal. Almost all global routers report the design routability statistic using overflow or underflow for Global Routing Cells (GRC), which is the ratio of routing cells' capacity and the number of nets that are required to route a given routing cell for all vertical and horizontal routing layers. The GRC statistic is a very good indication of wiring congestion and shows the number of nets needed to route a region versus the available number of routing layers. For an ASIC design to be routed completely without any design rule violations, this number needs to be less than one.

Global routing algorithms generate a non-restricted route (i.e. not a detail route) for each net in the design and use some method of estimation to compute wire lengths and extract their corresponding parasitics.

Depending on the global routing algorithm, there are several ways to estimate the associated wire length for each net in the design. The most common wire estimation methods are:

- Complete-graph
- Source-to-sink
- Steiner minimal tree
- Minimum spanning tree

- Half-perimeter
- Minimum chain

Complete-graph has connection from each port to every other port. In this method, all the interconnect lengths of the complete-graph connection are added together and then divided by $n/2$, where n is the number of ports. In a graph with n nodes, $(n-1)$ connections will emanate from each node to connect to other $(n-1)$ nodes in a complete-graph connection. Thus, a complete-graph has a total of $n(n-1)$ interconnections that include duplicate connections. Therefore, the total connectivity that requires forming a complete-graph connection is $n(n-1)/2$. Realizing that only $(n-1)$ connections need to connect n nodes, then to estimate reasonable wire length, the total net length of a complete-graph is divided by $n/2$.

Figure 4-3 shows an example of complete-graph wire length estimation based on the number of horizontal and vertical grids from a given source to multiple sink points indicated by dark and clear dots.

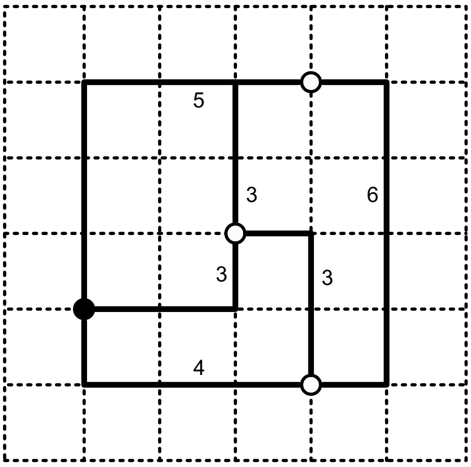


Figure 4-3 Complete Graph Wire Length Estimation

Source-to-sink global routing algorithm connects one port to all other ports for every net in the design. In other words, one wire from a source port to other ports is connected.

It is important to note that for the ports that are located far from each other, this algorithm does not produce an accurate wire estimate. Figure 4-4 illustrates source-to-sink wire length estimation.

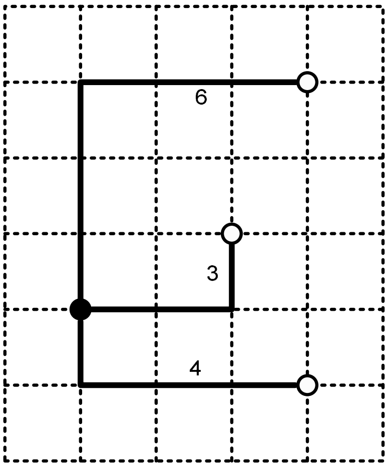


Figure 4-4 Source-to-Sink Wire Length Estimation

The Steiner minimal tree, or SMT, global routing algorithm is well suited for multi-port nets. For a given set of ports in the design, Steiner minimal tree connects these ports through some extra points, called Steiner points, to achieve minimum wire length.

The determination of the Steiner minimal tree is a Nondeterministic Polynomial time, or NP, problem which not only is difficult to solve, but also requires a large amount of computational time.

There are many ways to construct a Steiner minimal tree. The most frequently used is the Rectilinear Steiner minimal tree which is the shortest interconnect using a rectangular grid. The length of the tree is the sum of all interconnections, or the cost of the tree.

The objective of these types of algorithms is to find a tree with minimal tree cost. For a small number of ports, there are several heuristic algorithms based on the minimum cost of the spanning tree. Wire length estimation for Steiner minimal tree is shown in Figure 4-5.

The minimum spanning tree, or MST, algorithm is very similar to SMT. In this procedure, all design ports and interconnections are considered to be a complete graph and the spanning tree is a subgraph that contains all the ports and interconnections.

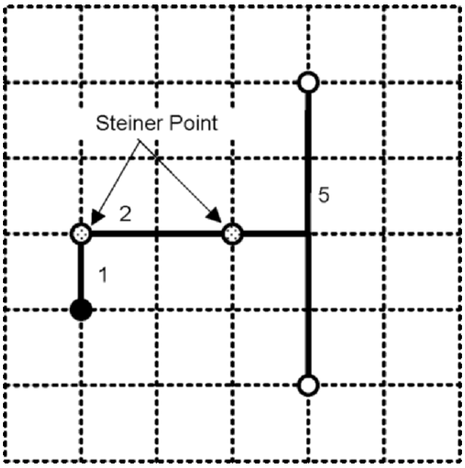


Figure 4-5 Steiner Minimal Tree Wire Length Estimation

The cost of the spanning tree is the sum of the lengths of each interconnection. Thus different trees of the same interconnect network have different lengths.

The objective of the MST algorithm is to find the minimum length of the spanning tree that is to connect each port in the design using the shortest route.

A network of interconnections may have many spanning trees. For example, a four-port interconnection has sixteen spanning trees, as illustrated in Figure 4-6.

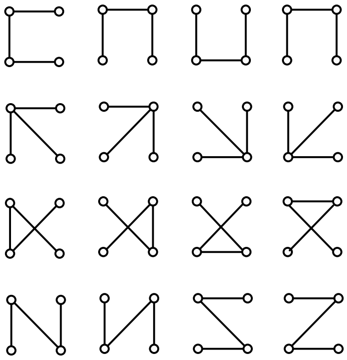


Figure 4-6 Sixteen Possible Spanning Trees

There are many types of algorithms offered today which find the minimal spanning tree. Due to the complexity of these types of algorithms, finding minimal spanning trees still remains the topic of research. Figure 4-7 shows minimum spanning tree wire length estimation.

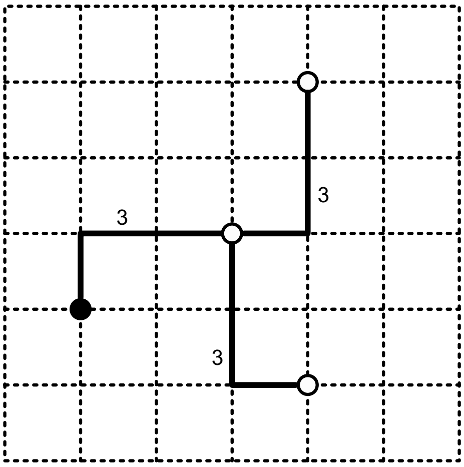


Figure 4-7 Minimum Spanning Tree Wire Length Estimation

The minimum chain connectivity algorithm starts from one port and tries to connect that to the closest point and then to the next closest point in the chain sequence. Figure 4-8 shows wire length estimation for minimum chain connection.

Half-perimeter global router uses a wiring bounding box that is the smallest rectangle that encloses all ports. The half-perimeter wire length estimation is one-half of the wiring bounding box.

In wire length estimation, the size of the wiring bounding box is an important factor in half-perimeter. The smaller the wiring bounding box, the better the correlation to actual routing.

One should note that for ports with fan-out one or two, the wire length estimation using the half-perimeter method is the same as Steiner minimal tree.

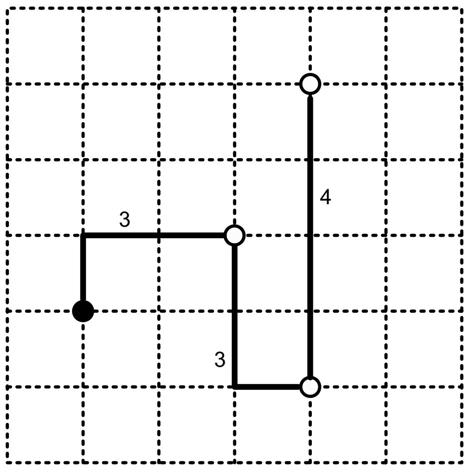


Figure 4-8 Minimum Chain Connection Wire Length Estimation

For ports with a greater number of fan-outs, the Steiner minimal tree wire length estimate is two or more times the half-perimeter wire length estimate. Half-perimeter wire length estimation is shown in Figure 4-9.

It is important to recognize that global routing is a key step in any place-and-route and physical synthesis tool. Global routing is used during placement and clock tree synthesis to estimate the wire lengths and RC delay time constants. Correlation between actual interconnect lengths after detail routing and approximation provided by any global route algorithm should be key for determining the appropriate place-and-route and/or physical synthesis tool to be used.

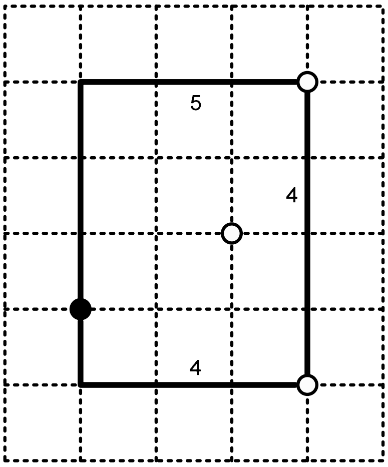


Figure 4-9 Half-Perimeter Wire Length Estimation

4.3 Detail Routing

The objective of detail routing is to follow the global routing and perform the actual physical interconnections of ASIC design. Therefore, the detail router places the actual wire segments within the region defined by the global router to complete the required connections between the ports.

Detail routers use both horizontal and vertical routing grids for actual routing. The horizontal and vertical routing grids are defined in the technology file for all layers that are being used. The detail router can be grid-based, gridless-based, or subgrid-based.

Grid-based routing imposes a routing grid (evenly spaced routing tracks running both vertically and horizontally across the design area) that all routing segments must follow. In addition, the router is allowed to change direction at the intersection of vertical and horizontal tracks as indicated in Figure 4-10.

The advantage of grid-based routing is efficiency. When using a grid-based router, one needs to make sure that the ports of all instances are on the grid. Otherwise, they can create physical design rule errors and will be difficult to resolve with the router.

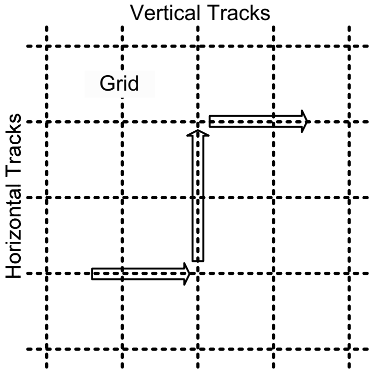


Figure 4-10 Grid-based Routes

Gridless-based (or shape-based) routers do not follow the routing grid explicitly, but are dependent on the entire routing area and are not limited by grid's restrictions. They can use different wire widths and spacing without routing grid requirements. The most fundamental problem with this type of router is that they are very slow and can be very complicated.

The subgrid-based router brings together the efficiency of grid-based routers with the flexibility (of varying the wire width and spacing) of the gridless-based routers. The subgrid-based router follows the normal grid similar to the grid-based router. However, a subgrid-based router considers these grids only as guidelines for routing and is not required to use them, as illustrated in Figure 4-11.

There are many detail routing algorithms available today. One of the most accepted ones is the Maze routing introduced by Lee [2]. In this algorithm, the routing area is represented as a grid by means of vertical and horizontal tracks for each layer that is used in the design. Each track crossing point can be the source or target of connection.

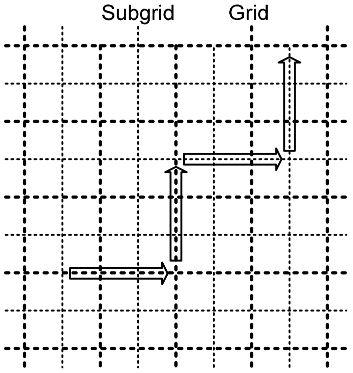


Figure 4-11 Subgrid-based Routes

In searching for the shortest path from source to target connection, the Maze algorithm performs a breadth-first search and labels each track crossing point with its distance from the source (similar to wave propagation). This expansion phase will eventually reach the target node if a connection is possible.

Once the expansion phase is completed, the trace-back phase begins with the connection from target to source by following the path with decreasing labels until the original source is reached. This algorithm is guaranteed to find the shortest path between source and target for a given connection.

Figure 4-12 illustrates the Maze routing (expansion and track back) algorithm.

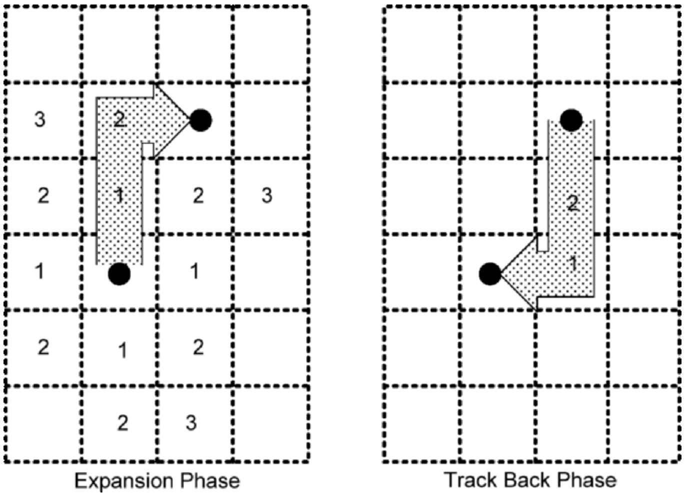


Figure 4-12 Maze Router Algorithm Illustration

Since the introduction of Lee's algorithm four decades ago, many improvements have been made to increase the efficiency of detail router algorithms with respect to memory usage, performance, and area optimization consumed by the routing area [3].

During detail or final routing of any ASIC physical design, total routing area is divided into regions or sub-areas known as switchboxes; these switchboxes are numbered in sequence for subsequent routing by the detail router. These switchboxes have terminals on all four sides (i.e. 2D) or may have connection points inside (i.e. 3D) as shown in Figure 4-13.

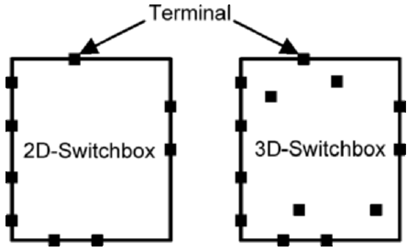


Figure 4-13 Detail Routing Switchbox

If the detail router uses a 3D-switchbox, then this type of routing is commonly referred to as area routing style.

This procedure of detail routing is very similar to global routing. The only difference is that during detail routing, physical wire segments will be used for connection rather than connectivity projections. Thus, it is important to have strong correlation between the detail and global routers with regard to the wire length approximation and actual wire connection. The reason for the close correlation between global and detail routers is that one can determine whether the ASIC design timing meets actual timing requirements by estimating the wire resistance and capacitance early on in the physical design cycle.

Once the switchbox construction is completed, then the detail router starts the track assignments. During track assignment, all routing paths for connection among the entire instance that are placed will be determined for every routing layer.

Depending on the standard cell design, the track assignment algorithm will use HVH or VHV models to assign tracks. Currently, most of the standard cell designs are optimized for HVH (i.e. metal one is routed horizontally; metal two, vertically and so on). If there are more than six metal layers available for routing, then the VHV model (i.e. metal one is routed vertically; metal two, horizontally and so on) provides much better results with respect to area. This routing area reduction is due to the fact that the VHV model has a tendency to produce shorter wire lengths and fewer via insertions.

After completion of track assignment, the detail router starts performing the actual routing of the first switchbox and, in sequence order, tries to complete the routing of all switchboxes. The objective of the detail router during this stage of route is to complete the entire routing area with no open connections between terminals even if connections would cause design rule violations or shorts.

The next phase of detail routing is known as search-and-repair. During this phase, the detail router begins with resolving all types of physical design rule violations such as metal spacing, fill notches, and wire shorts.

Usually, after this stage, the entire ASIC design is completely routed without any physical design rule violations. It is important to realize that a well-prepared ASIC library and proper floorplan and placement have a great impact on how well the detail router will complete this stage.

After completion of route and search-and-repair, via minimization and optimization is recommended–especially for deep submicron physical designs.

As mentioned earlier, most detail router algorithms' objective is to complete all switchbox routing and overall wire length optimization. Thus during detail routing, via optimization is not their major consideration.

Because the impact of large numbers of vias on any ASIC fabrication yield loss, most place-and-route tools offer via minimization and optimization options.

Via minimization refers to the process of removing as many vias as possible by reducing the number of jogs associated with a wire connection. This via minimization not only reduces the wire resistance (i.e. fewer vias), but also provides better yield with respect to via processing (e.g. one of the major reasons yield is lost in deep submicron).

Via optimization or redundancy increases the number of isolated single vias as much as possible as long as there is no impact on the routing area. The benefit of this option is two-fold—one is that it reduces the resistance of the overall via within a path (i.e. two vias in parallel) and two, it provides via yield enhancement by means of redundancy from a statistical point of view. In addition, it provides for better electromigration immunity.

Another step after detail routing is to fix any antenna related problems. Antenna problems as related to metal patterning during ASIC fabrication are described briefly in Chapter 2.

The antenna effect is a common name for the effect of charge accumulation in metal segments that are connected to an isolated transistor gate during the metallization process. This effect is also known as plasma-induced or process-induced damage. Plasma-induced damage has become a major concern for ASIC reliability.

There are several silicon processing steps during which charge accumulation can occur on a given isolated transistor gate. One of the trickiest steps in device processing (in terms of potential damage), is the step in which metal or polysilicon layers are etched. In order to prevent charge accumulation during processing, ASIC manufacturers have a design rule known as the antenna ratio.

The antenna ratio rule applies to any metal segment connected to a gate of transistors. It defines a limit for the ratio between the area of metal segments (i.e. peripheral length) and the area of the gate oxide of the gate to which the metal segment is connected.

Most place-and-route tools that are currently used are capable of checking the topology of overall ASIC wiring for antenna ratio rule violations. These antenna ratio rules are coded in the place-and-route tools technology file for the transistor gate area connected to the standard cell ports, as well as to any transistors connected to the macro port that is being used. More detail on the antenna ratio is given in Chapter 5.

The most common technique used to resolve antenna problems is to reduce the peripheral metal length that is attached to the transistor gates. This is accomplished by segmenting the wire of one type into several segments of different metal types, and connecting these various types of metals through via connections as shown in Figure 4-14.

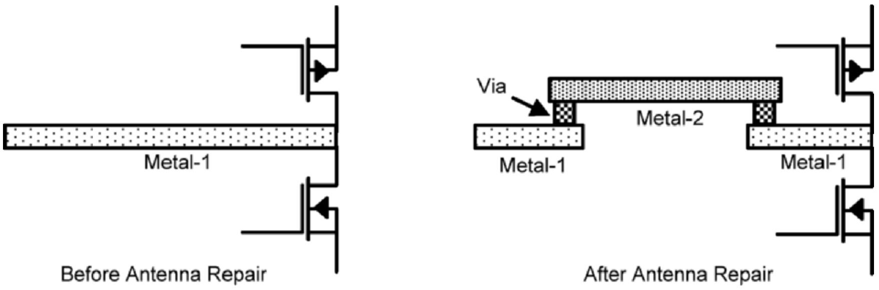


Figure 4-14 Antenna Repair Procedure

It is important to realize that extra via insertions during the antenna repairing procedure increases the wire resistance mainly due to via being very highly resistive. Therefore, it is strongly recommended to extract the routing parasitics after resolving all antenna violations in order to account for extra resistance.

Depending on how well global and detail routing correlate, and the impact of all design rule violation corrections on the final wire parasitics, some signal paths may show negative timing slack that requires optimization along these paths. This type of timing optimization is performed by up- or down-sizing standard cell drive strength along the failure paths as long as these changes do not influence the placement of other instances. These types of timing optimization are local and may not improve path timings that their timing constraints have been violated.

Another important consideration during detail routing is related to routing all clock nets and reducing the effect of crosstalk (as will be explained in Chapter 5). If clock tree synthesis is used for clock insertion, to insure that there is no crosstalk from these nets to other nets in the design, one can shield these nets by ground lines or space them apart from other nets. One of the main disadvantages of clock net shielding is that it increases the sidewall capacitance of the nets and the result is performance degradation.

An alternate approach is to modify the clock net’s minimum width from the default value to a larger one. Setting the clock net’s width to larger values than the default causes the router to skip a grid near these nets to prevent spacing violations. These nets will not only be spaced from other nets, but will also have a lower resistance due to the larger line width and less sidewall capacitance due to spacing.

Regardless of which method is used for clock networks, one could further reduce the resistance and improve the reliability of these clock nets by using redundant vias.

4.4 Extraction

Parasitic extraction is the calculation of all routed net capacitances and resistances for the purpose of delay calculation, static timing analysis, circuit simulation, and signal integrity analysis.

Parasitic extraction is performed by analyzing each net in the design and taking into account the effects (such as dielectric stack) of the net's own topology and proximity to other nets. Currently, most physical design tools are using three-dimensional (3D) models to extract capacitance associated with each net.

The parameters that govern parasitic extraction must set properly in order to yield the most accurate results in terms of measuring how well the extracted data correlates with actual silicon data after the ASIC is manufactured.

The most important parameters that are required to extract routing parasitic are resistance, capacitance, and inductance. Although today's physical design tools are using capacitance and resistance extraction, the importance of inductance extraction for future ASIC design should not be underestimated.

To extract wire capacitance and resistance, the semiconductor foundries provide the capacitance coefficient and sheet resistance values for various drawn layers in their electrical documentation.

These electrical parameters (capacitance coefficient and sheet resistance) are based on measurements performed using test-keys from actual silicon data for best, nominal, and worst process conditions or corners. The change in the value of capacitance coefficient and sheet resistance at each process corner is due to variations in dimension of the ASIC silicon features as well as to the effects of voltage and temperature.

The wire resistance extraction is based on the resistance of a uniform conducting material and can be expressed as

$$R = \rho(l/s), \tag{4.4.1}$$

where ρ is the resistivity, l is the length, and s is the cross-section of the material. Cross-section s is the product of height h and width w of the material. Thus, Equation 4.4.1 can be rewritten as

$$R = (\rho/h)(l/w)\,, \tag{4.4.2}$$

where (ρ/h) refers to the sheet resistance of the material in Ohm per square. Obtaining the resistance of a wire segment using Equation 4.4.2 is done by multiplying the material sheet resistance by the ratio of the segment length and width. To extract the correct value of the segment resistance, the effect of process and temperature must be included.

There are two process-induced topology variations associated with segment resistance—variation in thickness, or height, and variation in the width. Variation in thickness (Δh) results in a change of sheet resistance and variation in the width (Δw) results in a change of calculated resistance as illustrated in Figure 4-15.

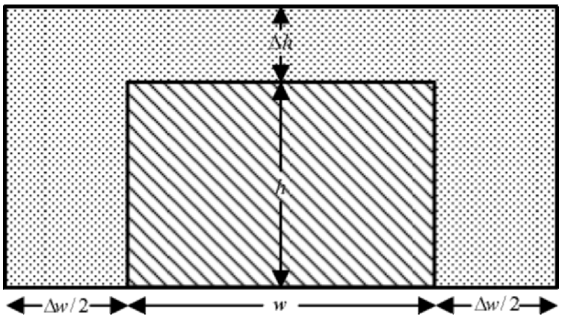


Figure 4-15 Wire Segment Cross-section

Variation in Δw is a result of metal layer patterning during the etching process. Since etching will cause a change in width of wire segment, and therefore its spacing to a neighboring wire segment, one should be able to apply this etching coefficient to the extraction tools for each metal layer.

By definition, a negative etch coefficient increases the width of the wire segment and reduces the spacing between wire segments, while positive etch coefficient decreases the width of wire segment and increases the spacing between the wire segments.

Change in the value of Δh is mainly due to metal layer and interlayer dielectric planarization for eliminating irregular and discontinuous conditions between successive metal layers.

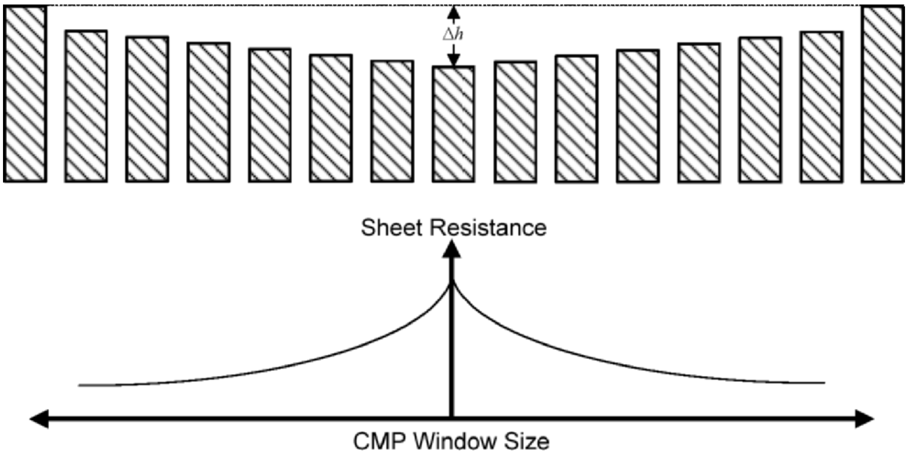


Figure 4-16 Chemical Mechanical Polishing Dishing Effect

Several techniques are used for planarization such as spin-on glass, deposit and etch-back, and Chemical Mechanical Polishing (CMP).

In today's fabrication processes, CMP is considered to be the main technique for the planarization process. During the CMP process, the oxide surface is planarized by rotating a wafer under pressure against the polishing pad in the presence of a polishing solution or slurry. The performance of CMP is greatly determined by polishing parameters such as down-pressure, rotation and speed of pads, and types of abrasives used. One of the main problems encountered during the CMP process (which is caused by the parameter setting such as rotating speed, and pressure of polishing pad) is known as the dishing effect. This effect impacts the interconnect sheet resistance and becomes especially important in deep submicron processes such as 130nm and below. The dishing effect is considered to be a localized problem and is determined by the CMP window size.

Figure 4-16 shows the result of CMP dishing effects as well as the profile of corresponding changes in the sheet resistance. Therefore, it is imperative that extraction tools be able to account for changes of uniform values in Δh due to dishing effect during resistance extraction.

Another consideration during wire segment extraction is the effect of temperature on the resistance value. Since the resistance of a metal segment is dependent on the collision process within the segment, the resistance value is expected to increase with temperature due to the increase in electron collisions with metal atoms. This temperature dependency of resistivity is characterized by a fractional change in resistance which is proportional to the temperature change and is expressed as

$$\frac{\Delta R}{\Delta T} = \beta R_0 , \qquad (4.4.3)$$

where $\Delta T = (T - T_0)$ is the change in temperature T from its initial temperature T_0, $\Delta R = (R - R_0)$ corresponds to the change of resistance R from its initial resistance R_0, and proportionality constant β refers to Temperature Coefficient of Resistance (TCR).

Equation 4.4.3 can thus be rewritten as:

$$\frac{R - R_0}{R_0} = \beta(T - T_0) \qquad (4.4.4)$$

$$R = R_0[1 + \beta(T - T_0)] . \qquad (4.4.5)$$

Although Equation 4.4.5 shows the effect of temperature on resistance, at extremely low temperatures the resistivity of metal layers is dominated by impurities or defects in the material and becomes almost constant with respect to temperature. Equation 4.4.5 assumes that the temperature is uniform across the integrated chip substrate and that during parasitic extraction, a fixed temperature is used to compute the wire resistance.

In the deep submicron process, and for very high-performance ASICs, assumption of uniform temperature across the chip for resistance calculation may not be adequate and non-uniformity of the temperature effects on resistance need to be considered.

Figure 4-17 shows three regions with different temperature profiles: **T1**, **T2**, and **T3**. The non-uniformity of temperature gradient across the chip substrate (e.g. **T1**, **T2**, and **T3)** is often due to the difference of operating frequency of functional blocks and their actual placement within the ASIC device. This different switching activity of different areas of the die and their corresponding non-uniform temperature gradients will create hot-area regions in the substrate. Interconnects across the hot-area will generate different resistances.

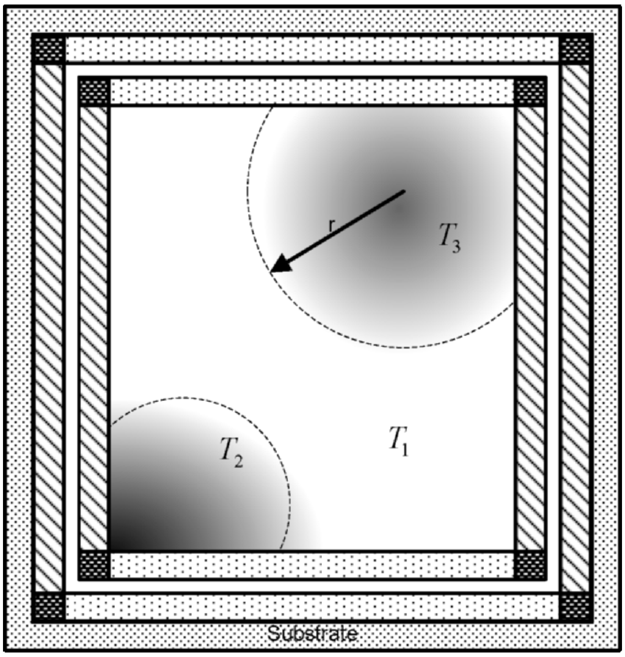


Figure 4-17 Non-uniform Substrate Temperature Profile

The differences in resistance due to the various temperature profiles in an ASIC substrate can lead to unpredictable timing delays, such as clock skew, if not included in the resistance extraction. As a result, determining the wire resistance based on temperature profile in hot-areas is crucial to the proper calculation of the resistance for those interconnects that are exposed to non-uniformities in the substrate.

One method of calculating the effect of non-uniform temperature on the resistance is to use Equation 4.4.6 as a function of the temperature of a circular radiator r centered at a hot-area region:

$$R(r) = R_0[1+\beta T(r)], \tag{4.4.6}$$

where $R(r)$ and $T(r)$ are the resistance and temperature at r location. One method used to express $T(r)$ is to use cylindrical coordinates given by:

$$\frac{\partial T}{\partial t} = k(\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{1}{r^2}\frac{\partial^2 T}{\partial \phi^2} + \frac{\partial^2 T}{\partial z^2}) . \tag{4.4.7}$$

By assuming radial symmetry about the z-axis and assuming $\partial^2 T/\partial\phi^2 = 0$, then Equation 4.4.7 reduces to

$$\frac{\partial T}{\partial t} = k(\frac{\partial^2 T}{\partial r^2} + \frac{1}{r}\frac{\partial T}{\partial r} + \frac{\partial^2 T}{\partial z^2}) , \tag{4.4.8}$$

where T is the temperature at location r and z corresponds to the substrate and routing layer thickness.

Applying various conditions such as process and temperature, the total interconnect resistance is given by

$$R_{total} = \sum_{i=1}^{n} R_i + \sum_{j=1}^{n-1} V_j , \tag{4.4.9}$$

where R_i represents the wire segment resistance and V_j represents the via resistance. It is important to note that most often the via resistance term in

Equation 4.4.9 dominates the total resistance and because of that, it is recommended to use multiple vias along critical paths in order to reduce the overall line resistance.

The next consideration is gate capacitance and routing segment capacitance extraction. Almost all extraction tools today use the 3D capacitance extraction method to extract routing segment capacitance.

The gate capacitance is dictated by the topology of the transistor. Each transistor gate capacitance is actually the sum of capacitances between gate to MOSFET body, gate to source, and gate to drain. Gate to MOSFET body (or substrate) capacitance with gate area of A defined as:

$$C_{gs} = (\frac{\varepsilon_{ox} A}{T_{ox}}) = (\frac{\varepsilon_{ox} WL}{T_{ox}}), \tag{4.4.10}$$

where ε_{ox} is the dielectric constant of the gate oxide, T_{ox} is the gate oxide thickness, L is the transistor drawn gate length, and W is the transistor drawn gate width. The term $\varepsilon_{ox} / T_{ox}$ expresses the oxide capacitance.

In actuality, both source and drain of gate extend below the oxide by amount L_d. This is known as lateral diffusion. Therefore, the affected transistor gate length becomes shorter than the drawn length by a factor of $2L_d$ as shown in Figure 4-18.

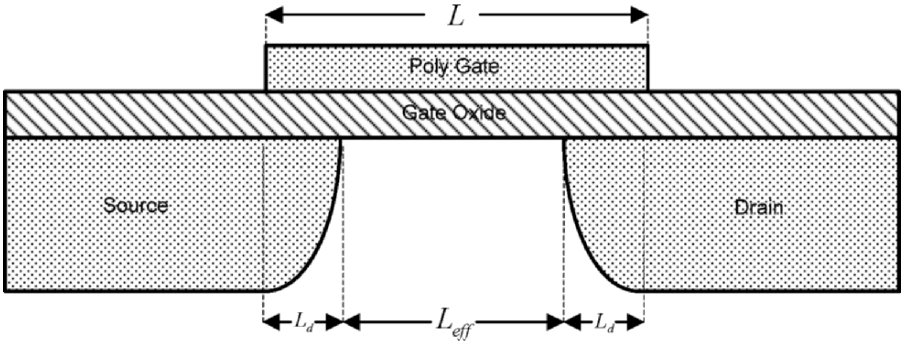


Figure 4-18 Transistor Cross-section

Rewriting Equation 4.4.10 to include the gate length effect results in

$$C_{gs} = \frac{\varepsilon_{ox} W(L - 2L_d)}{T_{ox}}. \qquad (4.4.11)$$

The reduction in the transistor gate length by the lateral diffusion effect gives rise to gate-to-source and gate-to-drain capacitance given by

$$C_s, C_d = \frac{\varepsilon_{ox} W L_d}{T_{ox}}, \qquad (4.4.12)$$

where C_s and C_d are gate-to-source and gate-to-drain capacitance.
The routing segment capacitance is extracted after final routing. During the routing segments' capacitance extraction, the following types of capacitance are considered:

- Area capacitance
- Fringe capacitance
- Sidewall capacitance

Area capacitance is routing segment capacitance to substrate and to other exposed routing segments above and below. The value of area capacitance is given by

$$C_p = \varepsilon \frac{LW}{d}, \qquad (4.4.13)$$

where C_p is the area or plate capacitance, ε is the permittivity of the dielectric, L and W are the overlapping length and width of the routing segment, d is the distance between the routing segments and corresponds to interlayer dielectric, or ILD, thickness, and the term ($\varepsilon W / d$) refers to capacitance per unit length.

Figure 4-19 illustrates the area capacitance for metal and polysilicon layers.

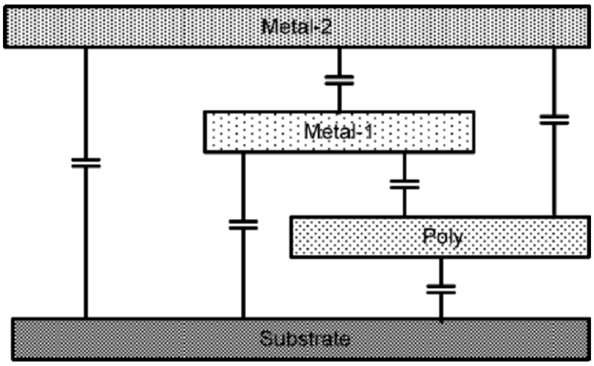


Figure 4-19 Area Capacitance of Two Metal Route

The values of interlayer dielectric thickness and the corresponding capacitance per unit area (ε / d) for various layers are defined by semiconductor foundries for different process corners.

Fringing capacitance is from the sidewall of routing segments to the substrate or other routing segment surfaces. Calculating fringing capacitance is a well-known problem within the electrostatic field theory.

A simple illustration of such an electrostatic field between two conducting plates is shown in Figure 4-20.

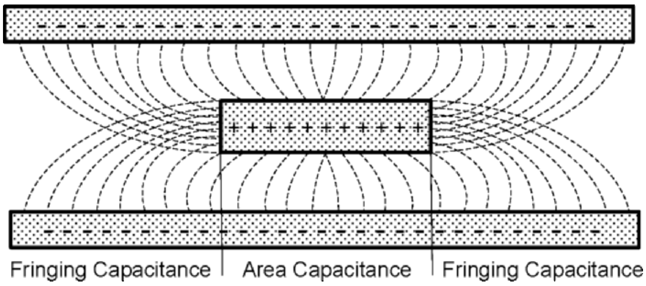


Figure 4-20 Electrical Field Flux Between Three Conducting Layers

Similarly to computing the mutual electrical fields between two conducting layers, analytical or numerical methods can be utilized to determine the fringing capacitance. Both method is computationally expensive and their execution run times become almost prohibitive. Thus, for a reasonably accurate extraction (as close as possible to theoretical value) either a simplified analytical formula or an empirical expression is used.

There are two frequently used methods for fringing capacitance extraction. One is a simple analytic approximation that has a direct physical interpretation [3]. Using this method, the value of fringing capacitance is given by

$$C_f = \frac{2\pi\varepsilon L}{\ln\{1 + \frac{2d}{h} + [\frac{2d}{h}(\frac{2d}{h} + 2)]^{\frac{1}{2}}\}} \quad \text{for } W \geq \frac{d}{2}. \qquad (4.4.14)$$

Second is to use an empirical expression. In this method, several numerical solutions are evaluated and defined. A purely empirical expression allows for all sidewall effects [4] as well as the fringing capacitance of a routing segment, and is given by:

$$C_f = \varepsilon L[0.77 + 1.06(\frac{W}{d})^{0.25} + 1.06(\frac{h}{d})^{0.5}]. \qquad (4.4.15)$$

In Equation 4.4.14 and 4.4.15, C_f is the fringing capacitance, ε is the permittivity of the dielectric, L is the length of the routing layer, h is the layer thickness, d is the dielectric thickness, and W is the width of the routing layer.

The effect of fringing capacitance on the fully routed ASIC design is very complex. For the purpose of illustration, Figure 4-21 shows a simplified fringing capacitance of different conducting layers.

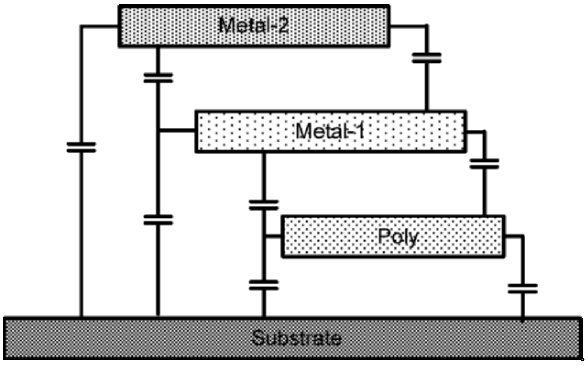


Figure 4-21 Fringing Capacitance of Two Metal Route

Sidewall capacitance is due to the coupling between the sidewalls of routing segments on the same layer and is similar to the area capacitance.
Figure 4-22 illustrates sidewall capacitance associated with different layers.

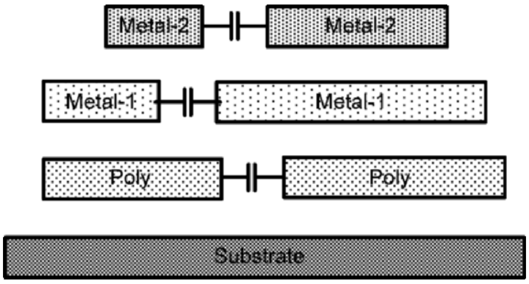


Figure 4-22 Sidewall Capacitance of Two Metal Route

The value of sidewall capacitance is the function of distance separating the routing segments and the length that they extend in parallel. It is given by:

$$C_s = \varepsilon \frac{Lh}{s}, \tag{4.4.16}$$

where C_s corresponds to the sidewall capacitance, ε is the permittivity of the dielectric, L is the length of routing segment, h is the thickness of routing layer, and s is the separation between two adjacent routing layer segments.

As feature size of drawn layers continues to shrink, the process that is used to create these features results in metal layer aspect ratios where height dominates over width and that causes a dramatic increase in the value of sidewall capacitance.

As the separation increases between two adjacent routes, the value of sidewall capacitance approaches zero, and therefore, it is recommended to reduce the coupling capacitance due to sidewall capacitance by increasing spacing between routing tracks in the place-and-route technology file.

Similar to resistance, capacitance values are also subject to process-induced variations. The variation in metal thickness affects fringing and sidewall capacitance, whereas the variation in the dielectrics' thickness impacts the area capacitance value.

Figure 4-23 demonstrates the effect of CMP dishing on the fringing and sidewall capacitance value due to decrease in metal layers' height.

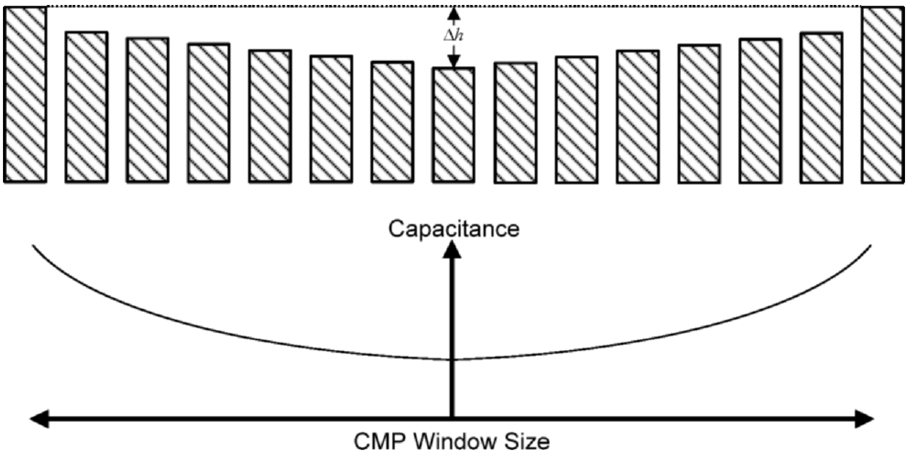


Figure 4-23 Effect of CMP on Capacitance Value

The CMP dishing effect must be considered during extraction–especially for fringing and sidewall capacitance. Fringing and coupling capacitance has a tendency to decrease at the center of the CMP window.

In the final interconnect capacitance calculation, the sum of all transistor gate capacitances associated with such interconnect must be added to the interconnect capacitance.

Including all contributions for a given interconnect, the total interconnect load capacitance is given by

$$C_{total} = \sum_{i=1}^{n} (C_p + C_f + C_s)_i + \sum_{j=1}^{k} (C_g)_j \,, \qquad (4.4.17)$$

where C_{total} is the interconnect total capacitance, C_p is the area capacitance, C_f is the fringing capacitance, C_s is the sidewall capacitance of each segment of the interconnect, and C_g is the transistor's input gate and junction capacitance connected to that interconnect.

With the advent of copper interconnect and its susceptibility to topographical thickness variation (i.e. dishing) due to the CMP process, semiconductor manufacturers require the use of a dummy fill.

Dummy fills are isolated islands (e.g. metal), as shown in Figure 4-24, and are used to create more uniform density that leads to a planar die surface. In the portion of the die, surfaces that are outside of a specified density range will have dummy fills inserted to meet the targeted density.

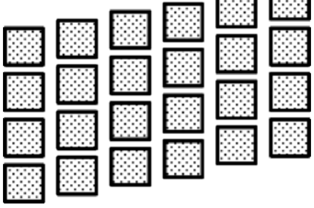


Figure 4-24 Dummy Fill Pattern

For the purpose of timing analysis, the extracted value of resistance and capacitance for each interconnect in the design is exported from the place-and-route tools and then imported to the timing analysis engine.

There are several approximation models used to represent interconnection parasitics. These approximations are:

- Lumped capacitance model
- Lumped resistance and capacitance (RC) model
- Distributed resistance and capacitance (Π) model
- Distributed resistance, capacitance, and inductance (RCL) model

Lumped capacitance is a single order approximation and considers only the total capacitance value of interconnection while ignoring the resistance value. This was used in the early days of process technology as wire delay contributions were negligible. Thus, this model was adequate to calculate the propagation delay through the gate as shown in Figure 4-25.

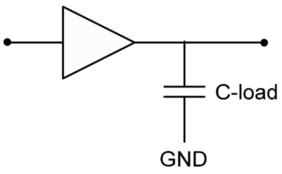


Figure 4-25 Lumped Capacitance Model

Due to the simplicity of this model, some layout synthesis tools still use it to estimate the capacitive loading effects during initial placement and logical restructuring.

This model can provide reasonably accurate results as long as the wire resistance is much smaller than the output driver resistance. However, as the driving gate output resistance decreases and wire resistance increases due to feature size scale-down, this first order approximation will no longer be valid.

Lumped resistance and capacitance (or RC model) is considered to be a second-order approximation and takes into account the effect of loading capacitance as well as the total wire resistance of interconnections as shown in Figure 4-26.

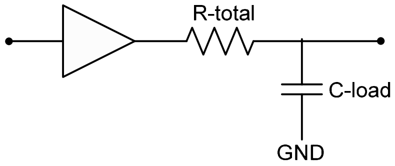


Figure 4-26 Lumped Resistance and Capacitance Model

In the second order approximation, the sum of wire capacitances is used to compute the cell delay based on the characterized model from the library and the product of lumped interconnect capacitance and resistance is used to compute the output slope. Hence, the resistance effects of the loading interconnects are taken into account in a decoupled manner. Using lumped RC to calculate interconnection delay is considered pessimistic and inaccurate for long interconnects. For those interconnections, a third order approximation is more appropriate.

Distributed resistance and capacitance (or so-called Π model) is classified as a third order interconnect delay approximation. In this model, interconnections are segmented into a series of resistor and capacitor network resembling the a transmission line. Figure 4.27 illustrates a simple distributed resistance and capacitance network.

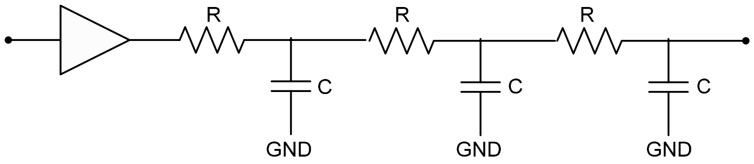


Figure 4-27 Distributed Resistance and Capacitance Model

The method of distributed resistance and capacitance is very effective for computing wire delays of very highly resistive interconnection networks because as wire resistance increases it has a tendency to shield the actual wire capacitance in the presence of driving gate resistance.

Today's physical design tools are capable of generating lumped capacitance, lumped RC, and distributed RC styles for wire delay calculation. Depending upon the style used, the extraction run time could increase. Lumped capacitance extraction has the shortest run time, whereas the distributed RC has the longest run time.

Choice of extraction style is dependent upon the application. For example, if one needs to perform dynamic power analysis, then lumped capacitance would be sufficient. For timing analysis or signal integrity where the effects are due to the resistance value and capacitance values, such as coupling effects to neighboring nets, the distributed RC extraction is more appropriate.

One of the most commonly used formats used to import and export distributed RC parasitic capacitance and resistance values extracted per net based on their actual geometry and layer width and spacing information, is the Standard Parasitic Extended Format (SPEF). SPEF is an Institute of Electrical and Electronics Engineers (IEEE) standard.

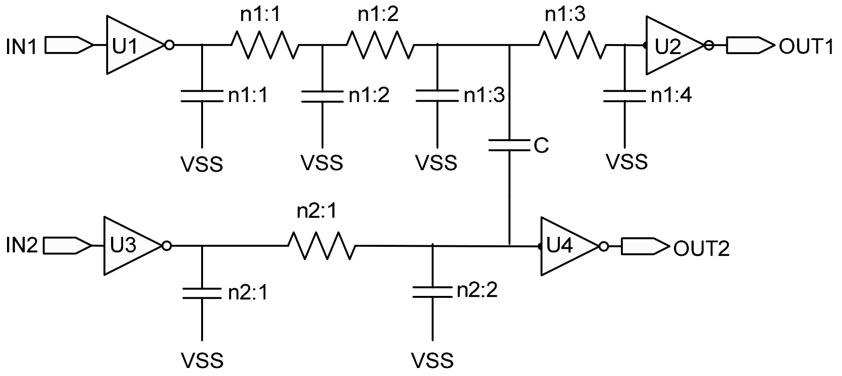


Figure 4-28 Simple Circuit Presentation

Figure 4-28 presents a simple circuit consisting of four inverters and two RC networks to discuss the basic SPEF format constructions (as shown in Figure 4-29).

```
*SPEF "IEEE 1481-1998"
*DESIGN "SAMPLE_SPEF"
*DATE "Thu Dec 22 12:12:24
2004"
*VENDOR    "None"
*PROGRAM   "None"
*VERSION   "1.1.0"
*DESIGN_FLOW   "PIN_CAP
NONE" "FULL_CONNECTIVITY"
"ROUTING_CONFIDENCE 90"
*DIVIDER /
*DELIMITER :
*BUS_DELIMITER [ ]
*T_UNIT  1 PS
*C_UNIT  1 FF
*R_UNIT  1 OHM
*L_UNIT  1 HENRY


*POWER_NETS VDD
*GROUND_NETS VSS

*PORTS
IN1 0 *L 0 *S 0 0
IN2 0 *L 0 *S 0 0
OUT1 0 *L 0 *S 0 0
OUT2 0 *L 0 *S 0 0

*D_NET n1 7.1
*CONN
*I U1:Z I *L 0 D INV
*I U2:A I *L 10

*CAP
1 n1:1 0.2
2 n1:2 0.3
3 n1:3 0.4
4 n1:4 0.1
5 n1:3 n2:2 0.07

*RES
1 n1:1 n1:2 2.2
2 n1:2 n1:3 8.0
3 n1:3 n1:4 3.7
*END


*D_NET n2 11.7
*CONN
*I U3:Z I *L 0 D INV
*I U4:A I *L 10.0

*CAP
1 n1:1 0.9
2 n1:2 0.8
3 n2:2 n1:3 0.07

*RES
1 n2:1 n1:2 5.0
*END

*D_NET IN1 15.1
*CONN
*I U1:A *L 10
*P IN1 O *L 0

*CAP
1 IN1:1 5.1
*END

*D_NET IN2 17.1
*CONN
*I U3:A *L 10
*P IN2 O *L 0

*CAP
1 IN2:1 7.1

*END
```

Figure 4-29 Standard Parasitic Extended Format

In this example, the circuit contains four inverters, **U1**, **U2**, **U3,** and **U4**; net **n1** connects **U1** and **U2**; and net **n2** connects **U3** and **U4**. Net **n1** is extracted as four capacitance segments and three resistance segments, and net **n2** is segmented as two capacitances and one resistance. One should note that the coupling capacitance between the third segment of net **n1** and the second segment of net **n2** is a symmetrical value. This coupling capacitance **C** is used primarily for signal integrity analysis.

It is important to realize that, although most of today's ASIC extractions are based on capacitance and resistance, this may not be adequate for future deep submicron design. Thus one must consider the effect of inductance as well as distributed resistance and capacitance.

Similar to the distributed RC approximation, in this model there are two types of inductance considered--one is the wire segment inductance and the other is the mutual inductance between long segments that run in parallel within the same layer.

The RCL model accounts for resistance, capacitance, and inductance and is a fourth-order interconnect approximation, as shown in Figure 4-30.

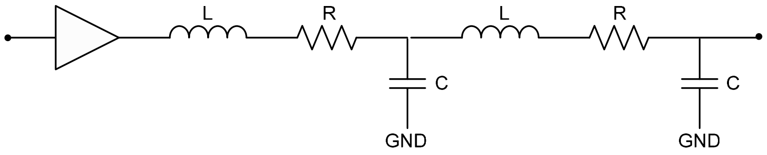


Figure 4-30 Distributed RCL Approximation

As technologies are advanced and individual wire lines and device features become scaled down, operating frequencies are increasing beyond 1 GHz. line inductances can no longer be ignored. This is especially important in analyzing the effect of wire inductance (ωL) in timing and signal integrity analysis. This is mainly due to the fact that as resistance per unit length increases significantly, so does the line inductance. It is important to note that the presence of inductance not only increases signal propagation delay, but also has a tendency to cause voltage overshoot, which results in a reduction of signal rise time, which in turn, is one cause for the increase of crosstalk noise.

4.5 Summary

In this chapter, we have discussed the basic principles of ASIC physical design relevant to global and detail routing as well as methods of parasitic extraction.

In the global routing section, we explored some of the basic global routing techniques for wire length estimation. We should note that the wire length estimation, their associated wire delays, and how well they correlate with the actual routing plays an important role in the final design. Therefore, it is imperative to choose a global routing method which will achieve the best correlation.

In the detail routing section, we discussed the principle of the Maze routing algorithm and the steps required for refining the final route from a manufacturability point of view.

In the extraction section, we gave an overview of today's wire delay calculation based on resistance and capacitance extraction. We also briefly discussed the importance of including the wire inductance in future ASIC design. Moreover, in this section we covered the importance of non-uniform temperature and its effect on the wire resistance.

Looking ahead, it is important to note that with the advent of advances in semiconductor processes and scaling down feature topology, classical assumptions are becoming invalid and, hence, new assumptions must be made in order to insure an accurate link between routing, parasitic extraction, and timing analysis.

Figure 4-31 shows the steps during the routing and parasitic extraction phase.

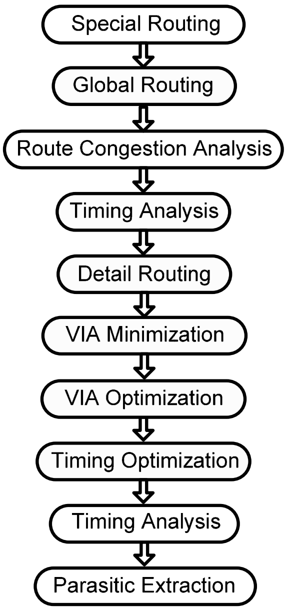


Figure 4-31 Routing and Parasitic Extraction Steps

References

[1] J.R. Black, “Electromigration – a brief survey and some recent results”, *Proceeding IEEE Int. Reliability Physics Symposium*, pp. 338-347, Dec, 1968.

[2] C.Y. Lee, “An algorithm for path connections and its applications”, *IRE Transactions on Electronic Computers*, vol. EC-10, pp. 346-365, 1961.

[3] C.P. Yuan and T.N. Trick, “A simple formula for estimation of the capacitance of two-dimensional interconnects in VLSI circuits”, *IEEE Electron Device Letter*, vol. EDL-3, pp. 391-393, 1982.

[4] N.v.d. Meijs and J.T. Fokkema, “VLSI circuit reconstruction from mask topology”, *Integration, the VLSI journal*, vol. 2, no. 2, pp. 85-119, 1984.

[5] W.C. Elmore, “The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers”, *J. Applied Physics*, 19, pp. 55-63, 1948.

Chapter 5

VERIFICATION

"Even if there is only one possible unified theory, it is just a set of rules and equations." Stephen W. Hawking

Verification is the final phase of any ASIC physical design before submission of the device for fabrication. This phase focuses on the testing of functional correctness and design manufacturability.

As ASIC designs are becoming more complex with respect to increased circuit density and advanced semiconductor processes, verification of these devices is becoming more difficult. The main objective of verification is to insure the functionality of ASIC designs and to minimize the risk of creating a design that cannot be manufactured.

Comprehensive verification is an iterative process and has a time complexity that is linear with respect to the size of the design and sublinear with respect to memory usage. One of today's ASIC design implementation challenges is to manage the iterative verification process, in terms of the time required, and to be able to improve throughput by means of automating the process of verification.

ASIC design verification is becoming an increasingly important phase of any physical design as design complexity increases and traditional verification tools have proven to be insufficient. A comprehensive ASIC design verification process consists of three phases. These verification phases are:

- Functional
- Timing
- Physical

5.1 Functional Verification

Functional verification is performed during the early ASIC design implementation stage. Functional verification that uses mixed tools and technologies for performing logic simulation, simulation acceleration, assertions, in-circuit emulation, and hardware/software co-verification.

Early-stage functional verification consists of three sequential tasks–modification, test, and evaluation. Standard functions and algorithms are used to verify the design behavior and compliance with the specification.

Once the design requirement is satisfied, the behavioral design description (i.e. Register Level Transfer) is transformed into the structural domain, or gate level description, by virtue of logic synthesis. Upon completion of the logic and layout synthesis, functional verification is performed against the behavioral RTL pre-layout and post-layout structural description (netlist) for design validation. The final design verification is accomplished by utilizing either simulation-based, rule-based verification, or both.

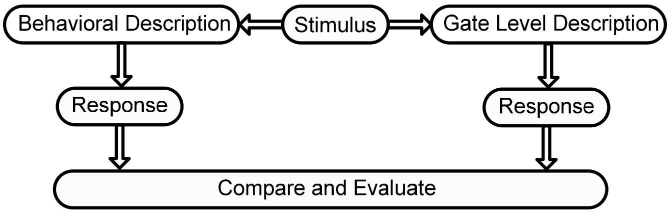


Figure 5-1 Simulation-based Functional Verification Flow

During simulation-based verification, the same stimulus is applied to the design behavioral description as to the final gate level description and their responses are compared and evaluated as shown in Figure 5-1.

Although functional verification through simulation constitutes the current state-of-the-art, this method has two main disadvantages. One is that simulation-based verification requires a long execution time and therefore often becomes impractical when ASIC designs are very complex and large. The other disadvantage is there may not be a comprehensive set of stimuli available to validate the entire design, forcing the designer to rely on some method of random stimulus generation for full design coverage.

To address the growing functional verification requirements and problems associated with simulation-based verification, various techniques have been suggested. Among these techniques, rule-based functional verifications have been adopted for very large and complex ASIC designs.

The most widely accepted rule-based functional verification techniques are assertion-based and formal-based functional verification.

Assertion-based functional verification uses an assertion library containing assertion macros that are used to verify specific design properties. Assertion macros are expressions that, if false, indicate an error, and are instantiated within RTL code that provides diagnostic information during the functional verification.

The levels of visibility and controllability provided by assertion-based verification are becoming ever more important as the size and complexity of ASIC design increases.

Although assertion-based verification has been used for many years (specifically in the Hardware Design Language (HDL) for functional simulation), it has more recently been used as an enabler of much more efficient functional verification methods.

One of the important aspects of assertion-based verification is to be able to reuse the assertion library for multiple verification tools with consistent specification mechanisms. Recently, there has been an Open Verification Library (OVL) initiative that focuses on defining the assertion library for open source standardization. Thus, the idea of assertion-based verification goes beyond functional capability and encompasses usability and methodology.

In contrast with assertion-based functional verification, formal-based functional verification uses equivalency and model checking to verify the RTL versus the structural (gate level netlist) description. There are two methods of formal-based functional verification that are frequently used, model checking and equivalence checking.

In model checking, in order to formally verify a design, the design must first be converted into a verifiable format that obtains a complete Finite State Machine or FSM (a design specified as a set of interacting components; each having a finite number of properties or states; states and transitions between states constitute FSM). Most of the time, design FSM conversion is accomplished by flattening the hierarchical design description into a single network and computing the outputs from the inputs by the network that consists of logic gates, sequential elements, and their interconnections.

The next step in model checking verification is to convert the flattened design description into a functional description that represents the output and the next-state variables as functions of the inputs and current state (creating design FSMs). Once FSMs are created, the formal verification algorithm traverses through each FSM in a recursive fashion and evaluates the values that are stored in all sequential elements for their true values. The network satisfies the algorithm if stated values are true for all states of the network.

Equivalence checking is another method of formal-based functional verification. In this method, two designs are compared to see if they are equivalent. Since equivalence checking uses the RTL description as the reference design, there is no need to determine whether the behavioral design description has been modified during logic synthesis and physical design activity. Therefore, the need for gate level simulation for function correctness can be entirely eliminated. In using equivalence checking, it is important to retain the original design hierarchical structure. If the original hierarchical structure is removed or optimized during logic and physical synthesis, backend function design verification can fail.

5.2 Timing Verification

The objective of timing verification is to insure that the ASIC design meets all timing requirements. Two methods of timing verification are in use:

- Dynamic simulation
- Static timing analysis

Dynamic simulation is performed either at the transistor-level or (logic) gate-level and supports a wide range of design styles while revealing the impact of gate delays and parasitic effect on functionality and performance.

Although gate-level simulation can handle large designs, it does not provide as much accuracy as transistor-level simulation. But with designs comprising of several million transistors, creating vector sets (stimulus) that cover all functional behaviors in an ASIC device for transistor-level simulation it is impossible. Because of that, gate-level simulation has been commonly adopted in the past for timing verification.

During the past decade, however static timing analysis has emerged over dynamic simulation as the preferred method of timing verification. Static timing analysis eliminates the need for vector sets and provides for exhaustive timing analysis across all input and output paths in the design.

Regardless of which method is chosen for timing analysis, both dynamic simulation and static timing analysis require that logic gates and wires are provided with path delays within the design.

The gate delays are provided by the macros and standard cell libraries. Since the timing models in the library give the gate delays, one of the key aspects in path delay computation is the wire delay calculation based on the ASIC final parasitic information such as effective capacitance and resistance for each net in the design.

For accurate calculation of wire delays, many methods have been suggested. Among these methods, the moment-based technique is the most widely adopted.

Moments are the impulse responses of the RLC (resistance, inductance and capacitance) networks (as shown in Figure 5-2) that can be analyzed using time-frequency transformation methods.

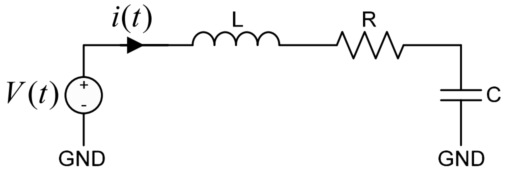


Figure 5-2 RLC Network

One of the well-known time-frequency transformation methods used to compute the impulse response of $f(t)$ at any node of an interconnect modeled as RC, RLC, or distributed-RLC circuit is to use Laplace transformation. This is defined by:

$$F(s) = \int_0^\infty f(t)e^{-st}dt = \sum_{i=0}^{\infty}\frac{(-1)^i}{i}s^i\int_0^\infty t^i f(t)dt\,. \qquad (5.2.1)$$

For example, to compute $i(t)$, (the current flow through a RLC network given in Figure 5-2) one can use a direct mathematical method by solving Equation 5.2.2 (where $V(t)$ is known):

$$L\frac{di(t)}{dt} + Ri(t) + \frac{1}{C}\int_0^t i(\tau)d\tau = V(t)\,. \qquad (5.2.2)$$

However, the direct mathematical method of solving Equation 5.2.2 tends to be complicated, so a time-frequency transformation, such as the Laplace transform can be applied [1].

Applying Laplace transformation to Equation 5.2.2 yields:

$$L[sI(s) - i(0)] + RI(s) + \frac{1}{C}(\frac{I(s)}{s}) = V(s)\,, \qquad (5.2.3)$$

and solving for $I(s)$:

$$I(s) = \frac{V(s) + Li(0)}{sL + R + 1/sC}. \tag{5.2.4}$$

To obtain $i(t)$, the inverse transformation can be applied to $I(s)$.

For commonly occurring input signals in time-continuous networks, Laplace transformation is defined as:

$$F(s) = \frac{a_0 + a_1 s + a_2 s^2 + ...+ a_m s^m}{b_0 + b_1 s + b_2 s^2 + ...+ b_n s^n}, \tag{5.2.5}$$

where the coefficients a_i for $i = 0...m$ and b_i for $i = 0...n$ are real for $n > m$.

Using Taylor series expansion, Equation 5.2.5 can be expressed as

$$F(s) = \sum_{n=0}^{\infty} m_n s^n . \tag{5.2.6}$$

For ease in notation, the coefficients m_n are often referred to as moments, defined as

$$m_i = \frac{1}{i!}\int_0^{\infty} t^i f(t)dt . \tag{5.2.7}$$

One application of using moments for wire delay calculation is to note that for $n = 1$, 2 and 3, $F(s)$ can be considered analogous to the first order (lumped capacitance), second order (RC circuit), and third order (distributed-RC circuit) models. These approximations are expressed as:

$$F(s) = m_1 s \tag{5.2.8}$$

$$F(s) = m_1 s + m_2 s^2 \tag{5.2.9}$$

$$F(s) = m_1 s + m_2 s^2 + m_3 s^3 . \qquad (5.2.10)$$

For lumped C, or first order approximation, as shown in Figure 5-3, using Equation 5.2.8, the approximation of the capacitance value is $C = m_1$.

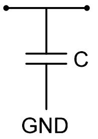


Figure 5-3 First Order Approximation

For lumped RC, or second order approximation, as shown in Figure 5-4, using Equation 5.2.9, the capacitance value is $C = m_1$ and resistance value is $R = -m_2 / m_1^2$.

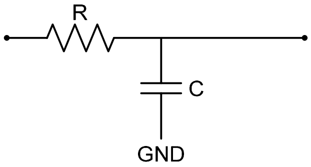


Figure 5-4 Second Order Approximation

For distributed-RC, or third order approximation, as shown if Figure 5-5, using Equation 5.2.10, the capacitance value is $C1 = m_2^2 / m_3$, the capacitance value is $C2 = m_1 - (m_2^2 / m_3)$ and the resistance value is $R = -m_3^2 / m_2^3$.

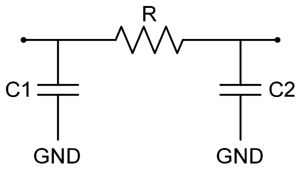


Figure 5-5 Third Order Approximation

Another method of using moment-based interconnect delay calculation is to use the first moment of the impulse response of $f(t)$ that is given by

$$m_1 = \int_0^\infty tf(t)dt\,. \tag{5.2.11}$$

Using the first moment to estimate the wire delay from impulse response of $f(t)$ is known as the Elmore delay model [2].

The Elmore delay model is by far the most popular model used to calculate wire delay due to its use of simple algebraic functions of the wire resistance and capacitance.

Considering an interconnection consisting of N nodes, the Elmore wire delay D_i for node i is given by

$$D_i = \sum_{k=1}^{N} R_{ki} C_k\,, \tag{5.2.12}$$

where R_{ki} is the resistance of the segment of the path between the input and node i that is common with the segment between the input and node k, and C_k is the capacitance at node k.

For example, given a three segment RC network in Figure 5-6,

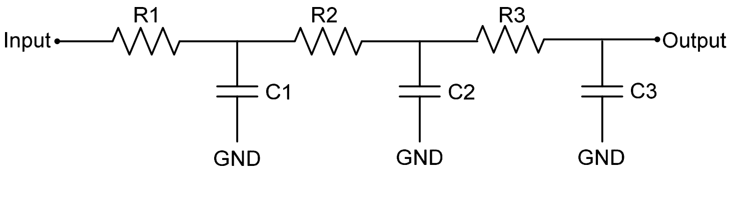


Figure 5-6 Three segment RC Network

then its corresponding Elmore delay is expressed as

$$D_w = R_1C_1 + (R_1 + R_2)C_2 + (R_1 + R_2 + R_3)C_3, \qquad (5.2.13)$$

where D_w is the wire delay from the **Input** node to the **Output** node.

It is important to note that although the Elmore delay model provides an accurate result (true delay) for nodes that are far from the driving point, it can be inaccurate by orders of magnitude for those nodes that are near the driving point.

This inaccuracy in Elmore delay calculation is primarily due to resistive shielding. Resistive shielding refers to the effect that if the resistive component of the wire is comparable to or larger than the driver output resistance, then the propagation delay through the driver is not calculated accurately due to the fact that metal resistance shields the capacitive component of the wire.

It is key to realize that, owing to the increase in total wire resistance with scaling down and the tendency for decreasing output resistance (smaller device size) of the drivers in deep submicron technology, resistance shielding is becoming more dominant.

To capture the gate and load interaction, and to be able to produce an accurate gate delay, many of today's ASIC design tools are using effective capacitance in their gate delay calculation.

Effective capacitance preserves the simplicity and efficiency of the Elmore delay wire calculation and provides for a more accurate capacitive load calculation using k-factors.

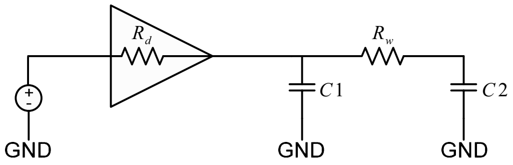


Figure 5-7 Basic Gate and RC Circuit

To illustrate the concept of resistance shielding and effective capacitance, Figure 5-7 shows a Thevenin voltage source driving a simple gate (i.e. a buffer) modeled by its internal resistance connected to a distributed-RC network.

Applying the Elmore wire delay equation to the circuit shown in Figure 5-7, then the total wire delay is given by

$$D_w = R_d C1 + (R_d + R_w)C2\,, \tag{5.2.14}$$

where D_w is the wire delay, R_d and R_w are driver and wire resistance, and $C1$ and $C2$ correspond to the wire capacitance.

Rearranging Equation 5.2.14 yields

$$D_w = R_d (C1 + C2) + R_w C2\,. \tag{5.2.15}$$

The first term in Equation 5.2.15 corresponds to wire delay at the output of the driving cell.

If the driver resistance R_d is much greater than the wire resistance R_w, then the driver delay is accurate and can be characterized as a function of the total load of ($C1 + C2$).

On the other hand, if the wire resistance R_w is equal to or greater than the driver resistance R_d, then the driver delay decreases, thus R_w tends to shield some of the $C2$ capacitance. Because of that, the driver delay could be characterized as a function of total load of ($C1 + kC2$).

The term $kC2$ refers to the effective capacitance (k parameter) that captures the resistance shielding k ranges from zero to one.

The effective capacitance is useful for correctly predicting gate delays during timing analysis. However, since the effective capacitance calculation relies on the value of the driver resistance, an environment independent of the driver model needs to be constructed. Since the environment of the driver model and the effective capacitance are determined in a coupled, iterative manner, the accuracy and simplicity of wire delay calculation is completely dependent on the computation method of effective capacitance (i.e. parameter k).

Although the Elmore delay inaccuracy can be improved by corrective factors (i.e. effective capacitance), there are more accurate methods that use higher-order moments of an RC circuit transfer function. However, these methods are all significantly more expensive to compute than the Elmore delay and that makes them difficult to utilize within ASIC design tools.

One of the more popular and accurate methods used in current physical design and timing analysis environment for estimating wire delay is Asymptotic Waveform Evaluation (AWE).

The AWE method uses higher order moments to construct a pole-residue transfer function $H(s)$ (expanding Equation 5.2.6 into its partial fractions) to approximate the actual transfer function $F(s)$, and is given by

$$H(s) = \sum_{i=1}^{q} \frac{k_i}{s - p_i}, \tag{5.2.16}$$

where p_i is the pole, k_i is the residue, and its corresponding time domain impulse response is

$$h(t) = \sum_{i=1}^{q} k_i e^{p_i t}. \tag{5.2.17}$$

The wire delay calculation using AWE is accomplished through uniquely specifying the poles and residues by matching the first $2q$ moments m_i of the transfer function of $F(s)$, given by Equation 5.2.6, to the corresponding moments of the transfer function $h(t)$.

Depending upon the wire delay calculation accuracy requirement, one can select a higher order of q; however, choosing $q = 2$ or 3 for approximation would be sufficient to capture a reasonably accurate response with reasonable computational complexity.

Once all gate and net delays are computed, the timing verification begins, to determine whether the design meets required performance. As mentioned before, one can use either event-driven (i.e. logic simulation) or static timing analysis.

Event-driven simulation requires that gate and net delays that are computed from routing parasitics are annotated into the simulator for the purpose of compilation. Use of a logic simulator is similar to that of a behavioral simulator. Basic activity is the user application of input signal values, or stimulus, propagation of these values through the circuit, and finally, generation of the circuit activity or responses via waveform display of inputs, outputs, and selected interior signals of the circuit.

Based on the level of abstraction, two types of logic simulators (switch-level and gate-level) are used for ASIC timing verification.

In switch-level simulators, transistors, capacitors, and resistors are modeled as elementary switches based on the flow of charge. During the simulation, equations governing the behavior of each component are approximated rather than manipulating continuous analog data that is computationally intensive. These types of simulators are not only capable of modeling circuit activity in terms of binary values (i.e. 0's and 1's), but can also use strength/value pairs for net values, thus allowing such states as driven or floating 0's, 1's, and unknowns.

Gate-level simulators rely on a higher level of abstraction in comparison with switch-level simulators by replacing the low level devices, such as transistors, capacitors, and resistors, by logic functions (e.g. AND, OR). Effective use of logic functions or gates allows very complex designs to be easily described and subsequently simulated at the gate level rather than at the switch level.

Most of these types of simulators use events during the simulation. Events are defined as incidents that cause the system to change its state during a distinct unit of time during simulation. An occurrence of these events provides an effective dynamic environment in which a circuit is simulated. Although this class of simulator has been used for timing verification for many years, there are only a few that are considered to be "golden" for ASIC timing sign-off by ASIC foundries and designers. Among these golden simulators, Verilog netlist or gate-level simulator is considered to be one of the most widely accepted gate-level simulators that is utilized for functional timing verification.

In order to use Verilog gate-level simulation after completion of the physical design for the purpose of timing analysis, one must export the post-layout structural netlist along with the Standard Delay Format (SDF). The SDF file incorporates the actual timing information such as gate delays, wire delays, and timing checks into an abstracted format. The SDF annotator via Programming Language Interface (PLI) parses the SDF in the simulator.

During parsing of the SDF file, the annotator may report errors that are either fatal or nonfatal. Fatal errors cause the SDF annotator to stop, whereas nonfatal errors will cause the SDF annotator to skip the actions that cause the errors. The root cause of nonfatal errors is mainly found to be due to inconsistencies found during annotation, such as when a condition specified in the SDF file cannot be matched with the Verilog library that is used. These types of errors need to be resolved prior to simulation.

The SDF format uses a keyword (known as a construct) to store timing information. These constructs are typically related to:

- Delays such as interconnect, ports, and devices
- Timing checks for hold, setup, recovery, width, and period
- Timing constraints
- Delay type such as absolute and incremental
- Conditional and unconditional path delays and timing checks
- Data type or library
- Scaling, environmental and technology

For the purpose of post-layout timing simulation, the SDF file contains path constraints such as interconnect, related gate delays, and timing check constructs that are exported from the physical design tools.

Figure 5-8 shows a circuit consisting of two instances connected in a series, and its corresponding SDF is illustrated in Figure 5-9.

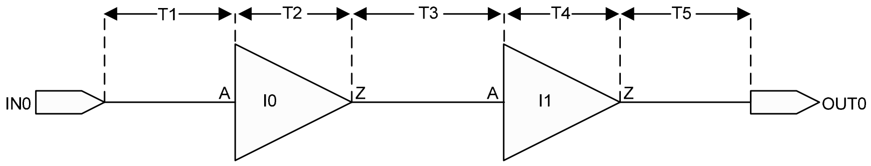
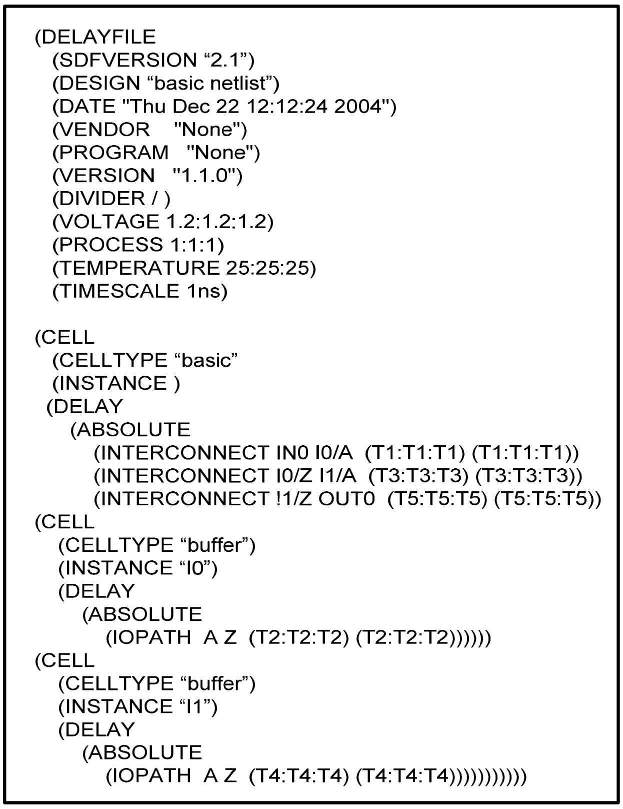


Figure 5-8 Circuit of Two Connecting Instances

Inspecting the illustration of the SDF file as shown in Figure 5-9, the file consists of design specific, or header entries, and delay entries that specify the delay values associated with interconnection between devices and ports.



```
(DELAYFILE
  (SDFVERSION “2.1”)
  (DESIGN “basic netlist”)
  (DATE "Thu Dec 22 12:12:24 2004")
  (VENDOR    "None")
  (PROGRAM   "None")
  (VERSION   "1.1.0")
  (DIVIDER / )
  (VOLTAGE 1.2:1.2:1.2)
  (PROCESS 1:1:1)
  (TEMPERATURE 25:25:25)
  (TIMESCALE 1ns)

(CELL
  (CELLTYPE “basic”
  (INSTANCE )
 (DELAY
    (ABSOLUTE
       (INTERCONNECT IN0 I0/A  (T1:T1:T1) (T1:T1:T1))
       (INTERCONNECT I0/Z I1/A  (T3:T3:T3) (T3:T3:T3))
       (INTERCONNECT !1/Z OUT0  (T5:T5:T5) (T5:T5:T5))
(CELL
   (CELLTYPE “buffer”)
   (INSTANCE “I0”)
   (DELAY
      (ABSOLUTE
         (IOPATH  A Z  (T2:T2:T2) (T2:T2:T2))))))
(CELL
   (CELLTYPE “buffer”)
   (INSTANCE “I1”)
   (DELAY
      (ABSOLUTE
         (IOPATH  A Z  (T4:T4:T4) (T4:T4:T4))))))))))
```

Figure 5-9 SDF Corresponding To Two Connecting Instances Circuit

The header contains information relevant to the entire design environment and its operating conditions such a voltage, process, and time scaling factors. The delay entries that are denoted by the DELAY keyword specify the delay values associated with interconnections between devices and ports, and the gate delays.

Interconnect construct represents the actual delay of the wires between instances that have INTERCONNECT syntax. The timing relation between an input port and an output port of an instance is considered gate delay with IOPATH syntax.

The delays that correspond to interconnect are specified as INTERCONNECT and are followed by the source and destination of wire connections and delay values (low-to-high and high-to-low) between output and input ports. These delay values are calculated by some method of estimation such as Elmore or AWE.

The delay that represents a timing arc on a path from an input to an output port is specified by IOPATH and is followed by unique input and output ports on an instance and their corresponding low-to-high and high-to-low path delays. These path delays are determined by using input transition and output effective capacitance loading. Since effective capacitance is a function of the driver's internal resistance and computation of the k-factor, one must make sure these values are properly calculated.

For both statements, INTERCONNECT and IOPATH, the values that are enclosed by parentheses represent minimum, typical, and maximum process, temperature, and voltage. If these values are different, one must specify which value is to be annotated.

Another option is to generate all these values for the same given minimum, typical, and maximum condition. For instance, in the example shown in Figure 5-8, the values of **T1**, **T2**, **T3**, **T4,** and **T5** can be based on minimum, typical, and maximum conditions using three different SDF files for each circumstance or by using a single SDF file with three different delay entries.

The cell entries that begin with the CELL keyword identify specific design instances that are specific to a design, library, or type, as well as their associated timing checks that determine the constraints between signals and how the signal can change in relation to other signals. For instance, the timing check associated with sequential elements can be SETUP, HOLD (with format that is similar to INTERCONNECT), and IOPATH.

Logic simulation has been part of ASIC design validation methodologies for a long time. However, if every cloud has a silver lining, then the clouds that affect logic simulations of large and complex ASIC is lack of comprehensive stimulus that exercises all paths in the design, simulation performance (e.g. event computation), and memory requirements (e.g. event storage). Because of these limitations, static timing analysis has emerged as the preferred method for timing verification.

Static timing analysis eliminates the need for comprehensive stimulus and provides exhaustive analysis of timing paths by computing the timing arcs between all input to output paths in the design.

Static timing analysis is considered an efficient method for performing case analysis that ensures all critical paths meet their timing requirements for a given design. Owing to the efficiency of static timing analysis, it is used as a core engine in many layout synthesis tools, allowing one to analyze timing requirements as more information becomes available during different stages of the physical design.

In principle, static timing analysis uses arrival time (T_a) which represents the latest time a signal can transition at a node due to the change at the circuit primary inputs and its forward propagation to the primary outputs.

In a similar fashion, static timing analysis uses required time (T_r) which represents the latest time a signal can transition at a node based on timing constraints that are propagated from the circuit primary outputs to the circuit primary inputs.

Using both arrival and required time, the relation that is given by Equation 5.2.18 must hold in order for the circuit to function properly.

$$T_r - T_a \geq 0 \tag{5.2.18}$$

Static timing analysis tools make use of timing graphs to compute the values of T_a and T_r. The timing graph is created automatically and is a directed graph where each port is represented by a node, and nets connecting these ports, by an edge. The start and ending points of interconnection are determined by the netlist, and internal edges of each cell are determined by their timing arcs defined in the library.

Timing graphs are very compact representations of the timing constraints and, at each node, have an associated positive or negative timing slack depending on the value of arrival time and required time.

A positive timing slack indicates that the node is earlier than the timing constraint and implies that the arrival time can be increased without affecting the overall delay of the design.

Negative slack means that the node is late and implies that the arrival time must be decreased in order to meet the required performance. One should note that the total number of timing slacks needed to completely constrain the timing of a given ASIC design is proportional to the number of sequential elements that are used, and have a tendency to grow exponentially as the design size increases. A basic timing graph is shown in Figure 5-10.

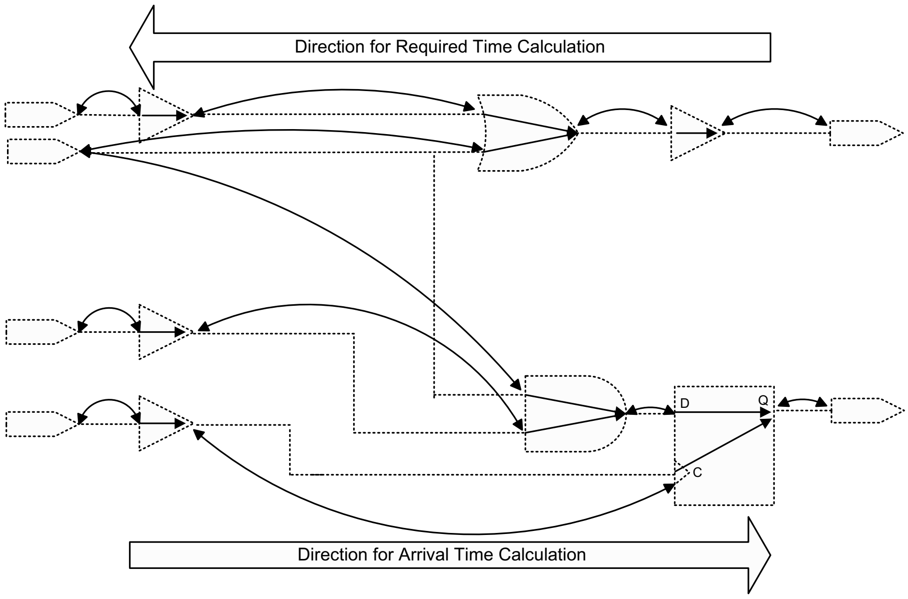

Direction for Required Time Calculation
D
Q
C
Direction for Arrival Time Calculation

Figure 5-10 Timing Graph Representation

Timing analysis is carried out in an input-dependent manner and the objective is to determine all worst-case delays of an ASIC design over all the possible input combinations. This is an exhaustive calculation and, if the circuit functionality is not considered, timing analysis tools may report some paths that are not meaningful. In order to improve efficiency and accuracy of static timing analysis, one must include the circuit states that do not exist in the actual application to distinguish whenever a long path found in the circuit is a true path or a false path.

A true or false path is defined as a set of connected gates that start from one input and end at another output. Creating all the false paths for an ASIC design can be as challenging as developing a set of comprehensive stimulus for logic simulation. Usually an iterative approach is applied during physical design to identify all possible false paths. At each iteration, timing analysis is performed to identify a new set of false paths until all actual critical paths are optimized.

The iterative process of identifying false paths is very time-consuming as the number of false paths identified may become very large. Therefore, a technique that is effective in identifying the false path is needed to reduce the cycle of this iterative process. For this reason, many approaches for determining false paths automatically have been developed or proposed.

One of the methods used for automatic false path generation relies on a process of sensitization. In this process, a so-called false path cannot be sensitized in terms of controlling logic value.

By definition, the controlling logic value is a single input value applied to an input of a gate that forces the output of the gate to a known value independent of the other inputs to the gate. For instance, applying the logic value of zero to an input of AND gate or applying the logic value of one to an input of OR gate will be considered a controlling logic value. Thus, a gate is considered to be sensitized if a signal transition can propagate through it from a particular input to the output as long as the other input has a non-controlling value such as logic value of one at input of AND gate or logic value of zero at input of OR gate.

These sensitization specific conditions are also referred to as sensitization criteria. During the sensitization process, the side inputs provide the controlling logic values associated for a given path, as illustrated in Figure 5-11.

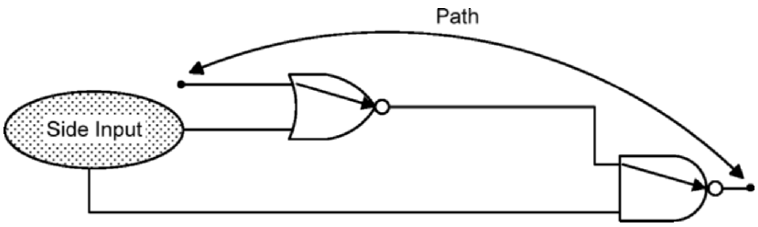


Figure 5-11 Side Input and Path Illustration

The sensitization criteria can be specified statically or dynamically. For a sensitization criteria to be static there must exist a set of input vectors that create non-controlling static to all side inputs along a path as shown in Figure 5-12.

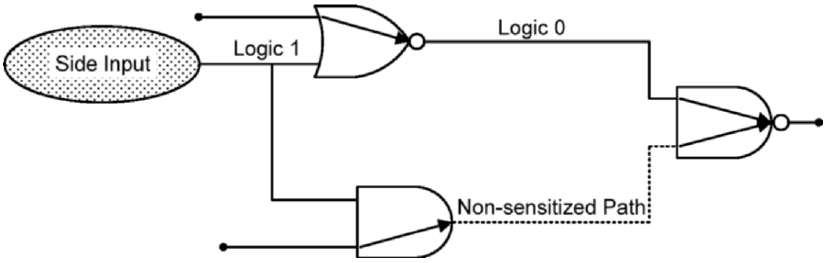


Figure 5-12 Static Sensitizations

The objective of dynamic sensitization criteria is to generate a set of input vectors such that when they are applied at different times they will produce non-controlling logic values at side inputs when the propagating transition signals arrive at a particular gate.

Depending on the timing for a signal transition from logic zero to one, or logic one to zero, from side input one and two, a path could be considered as true or false. Due to these types of timing dependencies, dynamic path sensitization is considered to be a very complex problem and there is much ongoing effort to simplify the process. Figure 5-13 shows the concept of dynamic sensitization.

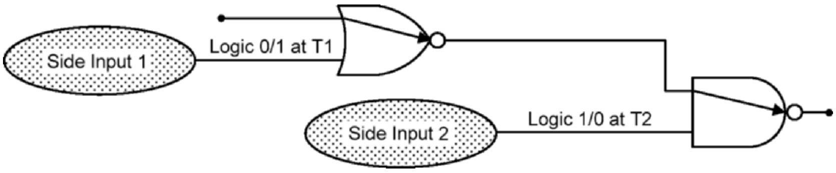


Figure 5-13 Dynamic Sensitizations

It is key to realize that for the determination of sensitization criteria, the path delays are not under- or overestimated. In addition, one must accommodate imperfect timing information such as process parameters and operating conditions.

From a timing analysis point of view, several timing paths are a matter of interest. These types of timing paths are:

- Input-to-register
- Register-to-register
- Register-to-output
- Input-to-output

These various timing paths are illustrated in Figure 5-14.

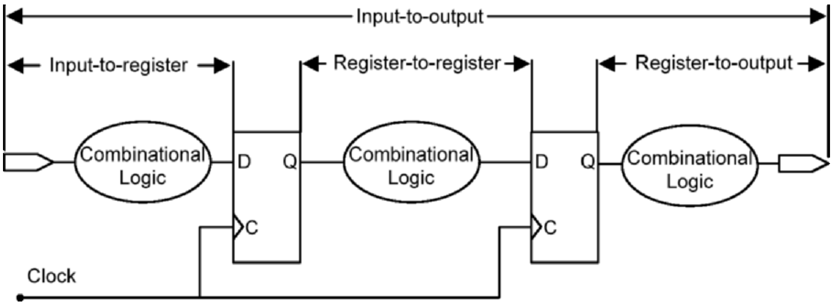


Figure 5-14 Typical Timing Paths

It is essential to have a set of comprehensive constraints and clock scheduling in order to properly analyze timing paths. The design constraints consist of input, output, and input to output delays with adequate timing margins to compensate for variations in design conditions such as process and temperature.

During timing analysis, one must make sure that these delays do not violate any design setup or hold timing requirements with respect to the clock interval. By definition, the setup constraint specifies the time interval before the active edge of the clock, and data must arrive before the interval. The hold time constraint specifies an interval after the active clock edge, and data must be stable in the interval.

Most timing analysis of an ASIC is a function of process, voltage, and temperature (PVT), and is based on the assumption that the entire device is impacted by a single PVT condition (e.g. worst case). In today's submicron process, however, it is imperative to realize that a more complicated PVT needs to be considered.

As mentioned in Chapter 3, this PVT profile complexity (or OCV) has a tendency to degrade the chip performance. Usually, the impact of these types of variation on chip timing cannot be analyzed by standard timing checks using single PVT conditions such as worst-case timing scenarios for setup checks or best-case timing scenarios for hold checks.

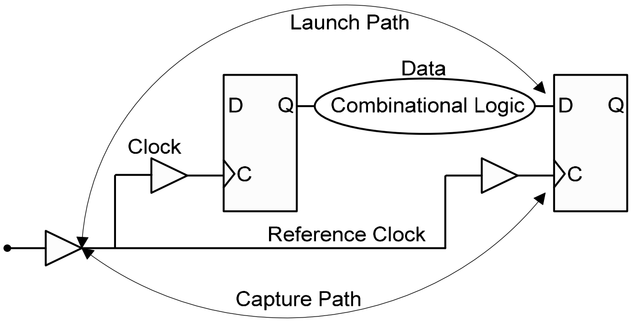


Figure 5-15 Launch and Capture Path

In performing OCV analysis, one must compare a computed pair of timing arc delays. For setup checks, the worst PVT condition is used for clock and data paths (launch path), and the best PVT condition is used for reference clock (capture path). Similarly, for hold timing checks, the best PVT condition is applied to clock and data paths, and the worst PVT condition is applied to the reference clock. An example of launch and capture paths is shown in Figure 5-15.

Another timing analysis consideration is the effect of resistance and capacitance coupling. High coupling resistance and capacitance in deep submicron processes results in both noise and additional delays that greatly affect the functionality and performance of any ASIC design.

Because the number of silicon failures caused by undetected and unresolved resistive-capacitive coupling violations is rising dramatically, it is important to minimize their occurrence. This can be accomplished by various methods such as wire spacing and shielding during the physical design stage and analyzing their impact on signal integrity during design timing closure.

The most common effects of interconnect coupling capacitance are considered to be induced by noise and delay. The crosstalk-induced noise occurs when signals in adjacent wires transition between logic values, and capacitive coupling between wires cause a charge transfer.

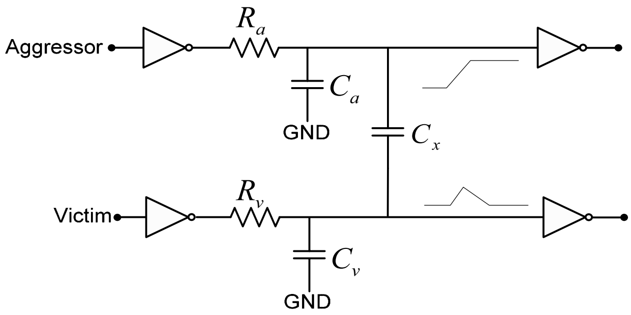


Figure 5-16 Crosstalk-Induced Noise Between Aggressor and Victim

Depending on the rate of change of the signal and the amount of coupling capacitance, there can be a significant possibility of noise injection between the aggressor and victim wires. For a capacitively coupled system, the aggressor is the wire that has the most signal activity, and the victim is the quiet one, as illustrated in Figure 5-16.

In the very simplistic illustration presented in Figure 5-16, a fast transition from the driver of the aggressor wire induces a glitch or noise at the input of the receiver of the victim wire regardless of its voltage level (logic 0 or logic 1). If the voltage of ensuing noises or glitches on the victim wire cross the input of the receiver threshold voltage, then a functional error may occur.

These types of errors can propagate through combinational logic and subsequently alter the state of registers in the design. This can lead to design functional failure.

Commonly, to calculate coupled voltage noise amplitude, voltage charge-sharing models are used. For example, considering Figure 5-16, a closed form expression for the peak amplitude of the coupled noise induced by the aggressor [4] is given by

$$V_n = \frac{Vdd}{1 + \frac{C_v}{C_x} + \frac{R_a}{R_v}(1 + \frac{C_a}{C_v})}, \qquad (5.2.19)$$

where V_n is the voltage noise amplitude voltage, R_a and C_a are the aggressor line resistance and capacitance, R_v and C_v are the victim line resistance and capacitance, C_x is the coupling capacitance between aggressor and victim lines, and Vdd is the aggressor switching voltage value.

It is interesting to note that as (R_a / R_v) approaches zero, Equation 5.2.19 reduces to

$$V_n = \frac{(Vdd)C_x}{C_x + C_v}, \qquad (5.2.20)$$

which is the basic crosstalk noise charge model equation.

Apart from noise injection due to crosstalk capacitance, this capacitance also has a serious impact on the adjacent wire delays. This situation becomes even more complex when signals simultaneously switch on both aggressor and victim interconnects.

For the purpose of illustration, consider two parallel interconnect lines driven by logic gates shown in Figure 5-16. Depending on the activity of these lines, their effective capacitance may be altered by the amount of coupling capacitance.

If the aggressor's line driver changes from high to low and the victim line is in the ground state (logic 0), then the coupling capacitance may cause the effective load capacitance on the aggressor line to be less than ($C_a + C_x$) or ($C_v + C_x$). On the other hand, if the victim's line driver is in the power supply state (logic 1), then the effective capacitance of the aggressor line may exceed ($C_a + C_x$) or ($C_v + C_x$).

The same effect can also be observed when the transition changes in the opposite direction for both aggressor and victim lines. In this situation, the effective load capacitance deviates from ($C_a + 2C_x$) for the aggressor line and ($C_v + 2C_x$) for the victim line.

The situations described here are somewhat simplistic. In practice, many topological possibilities and conditions exist that could contribute to the delay uncertainty for any given ASIC design. Therefore, the availability of an efficient and accurate timing engine to estimate the delay of a coupled system is a crucial element for crosstalk avoidance during physical design timing verification.

There are several design techniques that can be applied to minimize coupling effects. The most effective one is to increase the spacing between neighboring wires of the same metallization as described in the previous chapter. While this technique may be effective for selective interconnects such as clock signals, it is not economically efficient for global application due to its wiring area impact.

Another method that is used to minimize, or even eliminate, the delay unpredictability in a capacitive coupled system is to make sure that the aggressor and the victim driver strength and corresponding capacitive loading are the same. This is an effective method under an in-phase signal transition. In the case of out-of-phase signal transition, to reduce propagation delay due to crosstalk, the engineer needs to adjust the drive strength of the aggressor and/or the victim lines such that there is no transfer signal delay from one path to another path through the effective capacitance for various data paths.

Static timing analysis is an integral part of any ASIC design flow. However, for accurately studying critical paths and crosstalk-induced noise and delays, the use of highly sophisticated timing engines that can account for high order effects using transistor level dynamic analysis is required.

Apart from static timing, one should not underestimate the advantages that dynamic simulation can offer. For a comprehensive timing verification and validation methodology, one should include both static and dynamic timing analysis in their ASIC design flow.

Using dynamic timing simulation as shown in Figure 5-17, analysis can be advantageous in a sense that one can analyze the design timing requirement without breaking any timing loop; it is not possible to study this under static conditions. Moreover, the result of dynamic simulation can be used during the performance testing (or at speed testing) of ASIC devices.

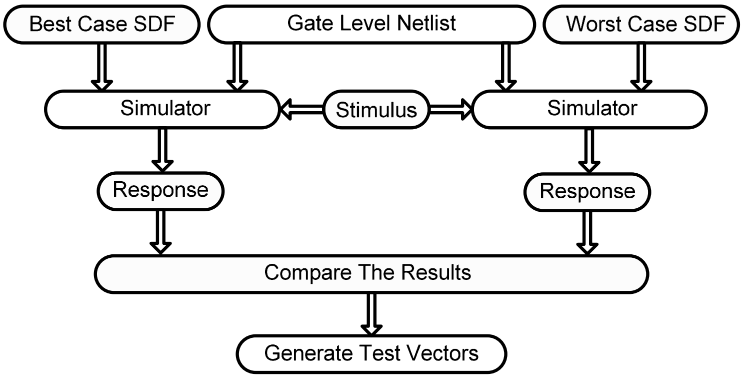


Figure 5-17 Dynamic Simulation Flow

5.3 Physical Verification

The primary objective of physical verification is to check the ASIC layout against process rules provided by semiconductor foundries to insure that it can be manufactured correctly.

The process of physical verification is highly automated and will continue to become more automated in the future. As ASIC designs and their fabrication processes become more complex, physical verification software advance.

Dramatically decreased device sizes, with respect to the number of transistors and routing layers involved in the net result for physical verification of larger layout databases, need to be verified against rules that are more complex. Verifying this amount of data puts strain on the software and computer infrastructure. Therefore, to improve physical verification efficiency and debugging procedures, one recommendation is to make sure that the ASIC is physically designed and implemented by employing a correct-by-construction methodology.

Several checks are performed during physical verification. Mainly these checks are Layout Versus Schematic (LVS), Design Rule Check (DRC), and Electrical Rule Check (ERC).

LVS verification examines two electrical circuits to see if they are equivalent with respect to their connectivity and total transistor count. One electrical circuit corresponds to transistor level schematics, or netlist (reference), and another electrical circuit is the result of the extracted netlist from the physical database. Through the verification process, if the extracted netlist (layout) is equivalent to the transistor level netlist, they should function identically.

LVS verification is critical during physical design and is fast becoming challenging for today's ASIC designs that consist of millions of transistors and enormous amounts of specialized data and deep submicron effects. Although current LVS verification software efficiently provides layout-based error reports, error diagnosis remains a concern.

One of the primary issues with LVS verification is the repeated iteration of design checking needed to find and remove comparison errors between the layout extracted and the transistor level netlist. The cycles involved in the LVS verification consist of data output (Graphical Design System or GDS) from the physical database, transistor level generated netlist, the actual LVS run itself, error diagnosis, and error correction. Thus, one of the objectives during LVS is to keep the length of time needed to complete the verification cycle as short as possible.

Currently, there are two ways to improve the LVS verification cycle time. The first consideration must be given to the capacity and performance of the machine infrastructure along with the type of physical verification software. Physical verification software must execute fast and provide accurate results that can be easily traced in case of error. The second is to use hierarchical verification features rather than flat comparison.

Using hierarchical verification features are very important, not only to minimize the amount of data to check, but also to identify the errors through the usage of hierarchical cells and black box techniques.

It should be pointed out that although the hierarchical verification method is far superior to that of the flat approach, LVS software that incorporates both hierarchical and flat comparison is far better than verification software that is either strictly hierarchical or flat.

By recognizing the components that simply match between the netlist and layout (such as standard cell libraries), memory block and other intellectual property (IP) elements can be compared in a hierarchical manner, while allowing other design elements such as analog blocks and macro cells to maintain a flat representation. In this way, debugging performance is drastically enhanced.

Finally, but most importantly, it is recommended to begin the verification process during an early stage of the design to insure that the physical database is correct. Under the assumption that most of today's physical design tools are capable of producing error-free place-and-routed designs, the most common sources of physical errors arise during the floorplanning stage and most often are related to power and ground connections. Power and ground shorts, and/or opens, impact device (transistor) recognition during LVS verification, which then leads to an extremely long execution time.

DRC is considered a prescription for preparing photo-masks that are used in the fabrication of the ASIC design. The major objective of the layout or design rule checking is to obtain optimal circuit yield without design reliability losses. The more conservative the design rules are, the more likely it leads to correctly manufactured ASIC designs. However, the more aggressive the design rules are, the greater the probability of yield losses.

DRC software uses so-called DRC decks during the verification process. Naturally, DRC decks should contain all manufacturing design rules. As semiconductor manufacturing becomes more complex, so do the DRC decks. These complex DRC decks must be written or composed in an efficient manner. If the layout rule checks are not coded in an optimized way in a DRC deck, they have a tendency to require more time (machine run time) or resources (machine memory) to complete verification.

Reducing DRC verification run time so that the rule deck is efficient from a coding point of view is desired. Similar to LVS verification, one also needs to consider utilizing the hierarchical process.

Hierarchical processing enables the tools to recognize multiple occurring instances throughout the physical database, and checks each instance one at a time, rather than checking all instances concurrently. Another advantage of hierarchical DRC verification is that the amount of data that needs to be checked is reduced together with the machine process and memory requirements.

Another consideration is to use a comprehensive DRC deck during physical verification. If the physical verification is not performed using a comprehensive DRC deck, the result can be low yield or no yield at all. For a DRC verification to be considered comprehensive, one must make sure the DRC deck checks all the rules correctly and accurately and is able to properly identify and address yield-limiting issues. The most common yield-limiting issues are:

- Charge accumulation due to antenna effects
- Inadequate planarity for multiple layers that are required by CMP
- Metal line mechanical stress
- ESD and latch-up

With regard to the antenna problem during the metallization, there are two methods of interest: one is ratio calculation and another one is the wire-charge accumulation. During the DRC verification, ratio calculation in conjunction with wire-charged accumulation is used. In the ratio calculation method, the following ratios can be calculated:

- Wire length to the connecting gate width
- Wire perimeter to the connecting gate area
- Wire area to the connecting gate area

Similar to the ratio calculation method, for wire-charged accumulation, assuming N is the metal layer currently to be etched, the following methods can be considered:

- Layer N connecting to the gate
- Layer N plus all layers below forming a path to the gate
- Layer N plus all layers below layer N

Another aspect of DRC allows one to check for DFM (Design For Manufacturability) such as contact/via overlaps and end-of-line enclosures. The DFM rules are considered optional and are provided by the silicon manufacturer. From a yield perspective, it is beneficial to check the ASIC physical design for DFM rule violations and then correct the errors as much as possible without influencing the overall die area.

ERC verification is intended to verify an ASIC design electrically. In comparison with LVS that verifies the equivalence between the reference and extracted netlists, ERC checks for electrical errors such as open input pins or conflicting outputs. A design can pass LVS verification, but may fail to pass ERC checks. For example, if there is an unused input in the reference netlist, then the extracted (routed) netlist will also contain the same topology. In this case, the result of the LVS process will be correct by matching both circuits, whereas the same circuits will cause an error during ERC verification (based on the fact that a floating gate can lead to excess current leakage).

In the past, ERC verification was used to check the quality of manually captured schematics. Since manual schematic capturing is no longer used for digital designs, ERC verification is mainly performed on the physical data. The ERC verification process is considered to be a custom verification rather than a generic verification. The user can define many electrical rules for the purpose of verification. These rules can be as simple as checking for floating wires, or more complex, such as identifying the number of PWELL- or NWELL-to-substrate contacts, latch-up, and ESD.

5.4 Summary

In this chapter, we have discussed the ASIC functional, timing, and physical verification processes.

In the functional verification section, we briefly explored the various methods such as simulation-based and rule-based verification style. In addition, for the rule-based method, we covered the basic concepts of assertion and formal methods.

In the timing verification section, we outlined the basics of wire delay calculation rather than its extensive mathematical derivation. In this section, we also overviewed the effective and coupling capacitances and their relationship with timing verification. In addition, we discussed methods for generating false paths in order to accurately verify design timing requirements. Furthermore, we explained noise effects on timing and provided some basic methods of preventing their impacts on the timing of the design.

In the physical verification section, we provided the basic steps that are required to verify an ASIC design physically such as LVS, DRC, and ERC. We should realize that the verification steps are critical procedures that integrate all phases of the design process, and, in today’s world, ASIC design verification is severely constrained by human and computer resources.

In addition, we should note that as the number of verification cycles that are required to verify an ASIC design grows exponentially as a function of gate-level complexity, then the solution to ASIC verification productivity problems is to deploy significantly more effective verification methods and tools. This requires a thorough understanding of the system and verification goals, process, and technology.

Figure 5-18 shows the basic steps during final ASIC verification phase.

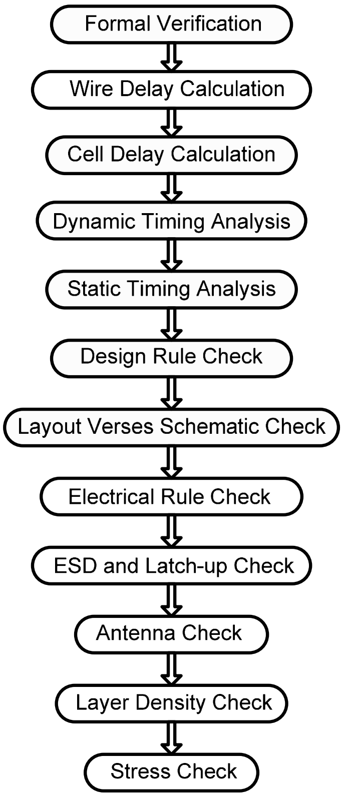


Figure 5-18 ASIC Verification Steps

References

[1] P. O'Brien and T. Savarino, "Modeling the Driving Point Characteristic of Resistive Interconnect for Accurate Delay Estimation", *Proc. IEEE. International Conference on Computer-Aided Design, 1998.*

[2] Roubik Gregorian and Gabor C. Temes, “*ANALOG MOS INTEGRATED CIRCUITS FOR SIGNAL PROCESSING”*, John Wiley & Sons, Inc., 1.986.

[3] W.C. Elmore, “The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers”, *Journal Applied. Phys.* Vol. 19, pp. 55-63, 1948.

[4] A. Vittal and M. Marek-sadowsks, “Crosstalk Reduction for VLSI”, *IEEE Transaction on CAD,* Vol. 16, No. 3, pp. 290-298, March 1997.

[5] Kevin T. Tang and Eby G. Friedman, “Delay and noise estimation of CMOS logic gates driving coupled resistive-capacitive interconnections”, *INTEGRATION, the VLSI Journal,* Vol. 20, pp. 131-165, 2000.

Chapter 6

TESTING

"It is possible to fail in many ways…while to succeed is possible only in one way." Aristotle

The rapid growth in the size of ASIC devices, the complexity of designs, and tighter design geometries are making test methods challenging. Device-level testing consumes a great deal of time and requires sophisticated programs necessitated by heightened reliability and performance standards required by many of today's products and complex Automated Test Equipment (ATE). This results in increasing costs and development time. There are many innovative techniques proposed to address this problem. One of the most profound techniques is to incorporate design-for-test.

The concept of design-for-test of an ASIC design is primarily related to both the controllability and observability of its internal nodes. In general, the circuitry of an ASIC needs to be designed in such a way that the internal nodes of the circuit can be controlled from the primary inputs and observed through the primary outputs. In contrast with board-level testing, where it is often quite easy to probe circuit nodes, with device-level testing, the ASIC must be fully testable from its package pins or I/O pads.

Unfortunately, there is no specific formula that can be applied to provide the answers to all design-for-test issues, as each design will be subject to different circumstances. However, by discussing the various conditions involved, the ASIC and physical designers must decide on the optimal or ideal test philosophy applicable to that particular design. The optimal or ideal test has a precise definition which can be applied with ease and maximum effectiveness (an ability of test programs unambiguously to distinguish faulty devices from those that perform properly over the range of specified operating conditions).

Since a fundamental requirement of any manufacturing process is the ability to verify that the finished product will successfully operate its intended function, any ASIC device should be subjected to sufficiently rigorous testing to demonstrate that the product's design requirements will meet various stages of the design. These testing stages are:

- Circuit and physical design
- Device processing (wafer-level testing)
- Device packaging (package-level testing)
- Device mounted to the circuit board (board-level testing)
- Circuit board insulated in the system (system-level testing)
- Product delivery (field-level testing)

It is important to note that the cost to detect and fix a defect increases exponentially as the product moves from early design to final product delivery. For example, if it would cost N units to fix a problem during the design phase, the same problem might cost $1000N$ (this is an arbitrary scale and depends on the various organization business models) to be fixed during field test. Figure 6-1 illustrates test costs at different ASIC testing stages.

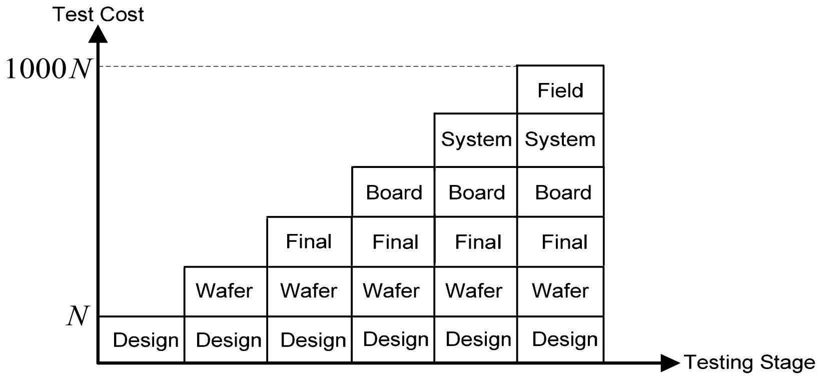


Figure 6-1 Test Cost at Different Testing Stages

The reason for this increase in the cost to fix design defects during field testing is because the only requirement to fix the defects during the design phase is to have a comprehensive test program, whereas fixing the faults in field testing relies on more complex and costly business and engineering structures. Therefore, it is imperative to fix any problems during the early stage of the ASIC design in order to insure system function reliability in the field.

Most of today's ASICs require two types of tests–functional and manufacturability. The following are the most common areas with regard to testing:

- Functional test
- Scan test
- Boundary scan test
- Fault detection
- Parametric test
- Current and very low-level voltage test
- Wafer acceptance test
- Memory test
- Parallel module test

6.1 Functional Test

The primary reason for functional, or at-speed, testing is to determine that the circuit logic is functioning correctly. In addition, it is necessary to establish that both AC and DC performance criteria of the design will be met over the full range of electrical and environmental conditions.

One of the prerequisites of functional testing is to create a set of comprehensive stimuli or test vectors that exercise the logic. These test vectors can be applied at the behavioral, or RTL, level or at the structural gate level. The advantage of simulating logic circuits at the behavioral level is shorter execution time, as no specific timing associated with the design is required. However, because of the lack of specific timing information, the result of behavioral testing cannot be imported to any test equipment for future testing.

For the purpose of creating a set of test vectors that can be imported to ATE, the final netlist from the physical design or place-and-route tools along with its parasitic information is used.

During test pattern generation, the relationship and sequence of input stimuli applied to the ASIC design to test the device performance and functionality are different from those that are used for circuit simulation used to verify the logic design. These differences are mainly due to the system requirement of ATE; thus, test vectors need to be simulated consistently with targeted ATE capabilities and their margin constraints.

Margin constraint refers to a value that is greater than the actual ATE input placement accuracy and is intended to prevent process-induced yield losses due to input margins. Margin constraint is used during input margin testing on a single process lot that cannot represent the full min-delay and max-delay process window. To overcome this limitation input skew margins over min-delay and max-delay conditions, simulation and/or static timing analysis can be used.

Some of the fundamental ATE capabilities are related to their:

- Operating frequency
- On/off clock width
- Number of signal pins
- Loading such as CMOS, TTL, etc.
- Input skew and placement accuracy
- Output strobe timing and stability

Generating a set of test programs that will completely cover the functional blocks in the design is not an easy task, especially if the circuitry was not designed with design-for-test in mind. As mentioned earlier, one aim in the design of ASIC devices is to design the circuit in such a way that its design function allows its internal nodes to be controllable and observable from primary inputs and outputs. This will increase the ability to determine the nature of any fault and can be particularly useful during the prototyping debugging stage.

In the case of complex ASIC designs, incorporating design-for-test features such as proper circuit initialization, asynchronous design avoidance, and use of multiplexers increases controllability and observability of the device and leads to improved testability.

Proper circuit initialization ensures predictable and repeatable operation of each circuit node under test conditions regardless of their power-up condition.

Initialization conditioning may not be as important for combinational logic, however, it is important when dealing with sequential logic–especially if sequential circuits require a complex test pattern or test programs. In that case, one may need to consider inclusion of a master reset from a primary input pad to enable all initialization of all registers in the design.

Using reset lines can also be advantageous during logic simulation, particularly those that include unknown states. Although circuit initialization can be achieved by forcing internal circuit nodes to a known value, such as logic ‘1’ or ‘0’, during logic simulation, it may have a fail to correspond with circuit initialization during actual device testing. This problem occurs because the internal nodes of ASIC devices cannot be initialized externally.

Asynchronous design is another example that may require additional consideration with regard to design-for-test. Although synchronous design techniques are always preferred, some occasions may require asynchronous circuit design.

Asynchronous design, by its nature, is not only difficult to control, but also may lead to other timing problems such as races or glitches in decoding logic. When dealing with asynchronous circuitry, extreme care must be practiced during logic and physical design in order to avoid any timing hazards.

Use of multiplexers may increase both controllability as well as observability during the test phase. An inaccessible circuit has low circuit observability and, therefore, is very difficult to test. One often-used method of test is to connect non-observable nodes directly to a primary output using a multiplexer, as shown in Figure 6-2.

In this example, the non-observable **D** input of the second flip-flop is connected the **A** input of the multiplexer, allowing the **D** input to become an observable node.

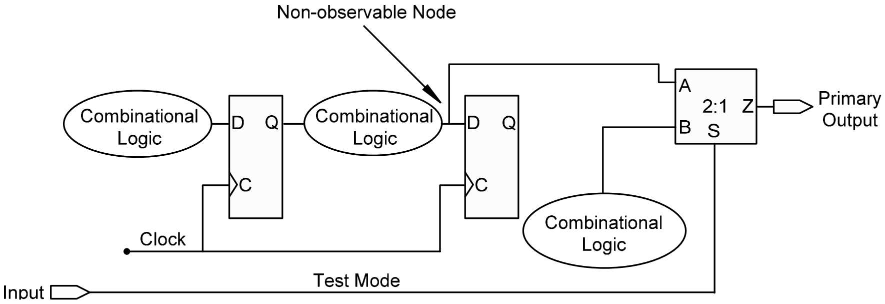


Figure 6-2 Non-observable Node

Multiplexers may also to be added to the circuit for improved controllability. Designing for controllability often reduces the number of test patterns required and should be considered when long dividers or counters are present in the design, or for initialization purposes, as shown in Figure 6-3. In this case, inserting a multiplexer at the **D** input of the flip-flop allows the circuit to be initialized from a primary input.

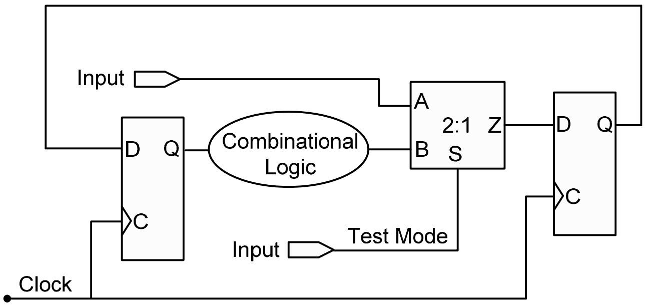


Figure 6-3 Multiplexer Initialization Usage

In addition, multiplexers can be used as bypassing elements when a design contains an on-chip clock oscillator or Phase Lock Loop (PLL) that cannot be controlled externally, as shown in Figure 6-4. This is important if it is not possible to synchronize the ATE to the Device Under Test (DUT) during test.

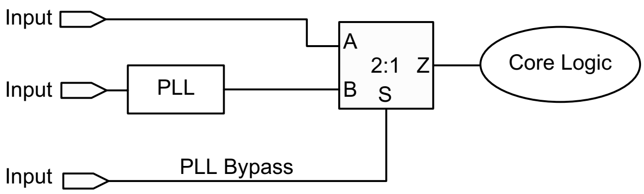


Figure 6-4 Multiplexer Usage as PLL Bypass Mode

6.2 Scan Test

Scan test is classified as a test for manufacturing and is used to detect stuck-at faults (stuck-at-zero and stuck-at-one) that may occur during the fabrication process.

The main objective of scan test implementation is to achieve total, or near total, controllability and observability in sequential circuits. In this method, ASIC design uses scan flip-flops, latches, or both, to operate in either parallel (normal) mode or serial (test) mode.

The scan flip-flop is designed to provide scan-testable features in edge-sensitive ASIC design. These scan flip-flops can be used in place of normal flip-flops when designing registers, counters, state machines, etc.

A scan flip-flop consists of a basic flip-flop and a 2-1 multiplexer as shown in Figure 6-5, with **SI** as scan input, **SE** as scan enable, **DI** as the normal data port, and **CK** as input clock.

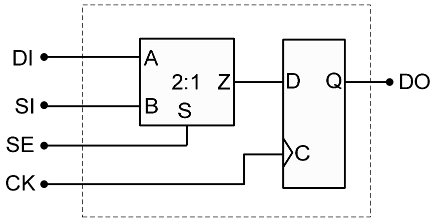


Figure 6-5 Basic Scan Flip-flop Configuration

In the normal or functional mode, the scan flip-flops are configured for parallel operation (or capture mode). In test (or scan) mode, the scan flip-flops are loaded (controlled) by serially clocking in the desired data (or shift mode).

The scan latch is designed to provide scan-testable features in level-sensitive ASIC design. These scan latches can be used in place of normal latches when designing registers, counters, state machines, etc. Level Sensitive Scan Design (LSSD) addresses the problems associated with implementing the scan flip-flop design [1].

LSSD improves the scan design concept by making the scan cells level-sensitive. In a level-sensitive system, the steady-state response to input changes is independent of circuit and wire delays within the system. One of the challenges using this technique is that LSSD imposes constraints on circuit excitation, particularly in handling clocked circuitry.

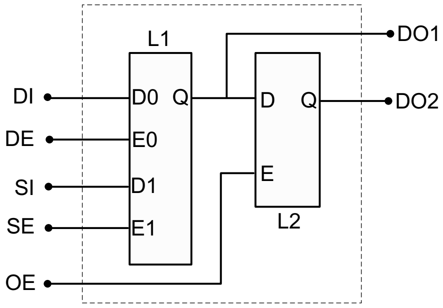


Figure 6-6 Basic Scan Latch Configuration

Scan latches consist of a basic dual-port latch (**L1**) and a basic single-port latch (**L2**) as shown in Figure 6-6.

Multiple enables **DE**, **SE**, and **OE** control the latches. In normal operating (or capture) mode, enable **DE** latches data into **L1** from the data input **DI** while scan enable **SE** is inactive. In the scan (or testing) mode, scan enable **SE** latches in the scan data **SI**, while data enable **DE** is inactive. Output enable **OE** transfers this data from **L1** to **L2**. In this mode of operation, the output data can be taken from either **DO1** of **L1** or **DO2** of **L2**.

In today's complex ASIC designs, scan testing is becoming an integral part of the design process in order to achieve a higher level of fault detection after manufacturing.

In the physical design flow, all functional design constraints must be met in the presence of scan logic. During the placement and clock tree synthesis phases, the physical information (or standard cells' physical locations) of the design is used to minimize routing congestion due to scan chain connectivity as well as minimize functional and scan clock skews.

For minimizing routing congestion during the placement phase, scan chain reordering is performed. It is important to have physical design tools that are capable of optimizing the reordered chain for both timing and power.

Consideration of clock distribution and balancing to minimize clock skew in both functional and scan mode is the next step. This may present some challenges if the ASIC design consists of multilevel clock gating elements, clock dividing, mode switching circuits, and scan clock.

Most of today's CTS tools can generate quality clock trees. However, when clocking logic becomes more complex because of intensive clock gating logic and/or multiple mode-switching logic, these tools may not guarantee construction of high quality clock distribution with regard to skew budgeting and buffer-tree balancing. Thus, it is imperative that physical design tools and their clock tree synthesis algorithms perform multimode clock synthesis and be able to meet both functional and scan timing requirements simultaneously.

6.3 Boundary Scan Test

Boundary scan design extends the scan design methodology for testing primary inputs and outputs of an ASIC design. This technique is suited for solving problems resulting from test equipment costs and testing difficulties with surface-mount technology such as automated probing.

This technique allows designers to place boundary scan cells next to each I/O pad in the design for improved controllability and observability within and between the chips.

At the chip level, boundary scan cells have other terminals through which they can be connected to each other. They then form a shift register path around the periphery of the ASIC, thus making it possible to test the package pins.

Figure 6-7 shows a basic boundary scan cell consisting of two 2:1 multiplexers and one flip-flop.

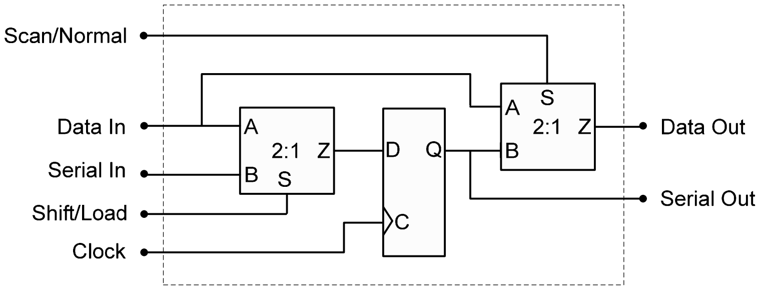


Figure 6-7 Basic Boundary Scan Cell

During normal operation, data passes between the ASIC I/O pins and internal core logic through **Data In** and **Data Out** ports as if the boundary scan cells were transparent. When the ASIC is in test or boundary scan mode, however, the data passes through the boundary scan cells' shift register paths between **Serial In** and **Serial Out** ports. By loading data into the boundary scan cells, the boundary scan cells can prevent the data flow to or from the ASIC I/O pads, so that the ASIC can be tested for either its internal logic or for external chip-to-chip connections.

Using the concept of shift register paths, arbitrary data can be loaded into the boundary scan cells and data can be observed from those boundary scan cells. The advantage of boundary scan testing is that it simplifies the test pattern generation required to create test vectors for an ASIC design.

In order to use the boundary scan cells, additional I/O pads are required, as well as some additional logic on the ASIC design in order to control the boundary scan chain for device testing.

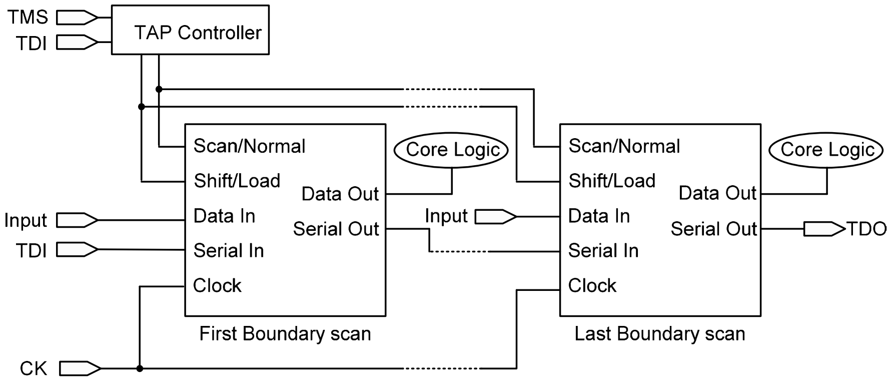


Figure 6-8 Boundary Scan Chain Configurations

IEEE has addressed these boundary scan requirements by setting a standard: ANSI/IEEE Standard 1149.1 IEEE Standard Test Access Port and Boundary-Scan Architecture [2]. This standard defines the I/O pins (Test Access Port, or TAP), control logic, or TAP controller (state machine) that generate the control signals required to operate the ATE for boundary scan testing of ASIC devices.

ANSI/IEEE Standard 114.1 gives details on expansion of the tool set for chip test designs. It also defines a method for communicating test instructions and data from an external processor to the ASIC.

Based on these requirements, the ASIC I/O pads can be configured to meet the IEEE Standard 1149.1 specifications. Figure 6-8 illustrates basic boundary scan chain configurations and basic TAP controller connections.

In Figure 6-8, the additional pins required for an ASIC design to implement boundary scan testing features are as follows:

- The Test Data In, or TDI, that connects to the TAP controller and the first boundary cell **Serial Input** port in the chain
- The boundary scan Test Clock, or TCK, that connects to all boundary scan **Clock** ports in parallel
- The control signal TMS connects to the TAP controller for generation of proper sequencing of test signals and is also connected to the **Scan/Normal** and **Shift/Load** ports of all boundary scan ports
- The Test Data Out, or TDO, that connects to the last boundary scan cell **Serial Out** port in the chain

6.4 Fault Detection

The purpose of testing is to detect defects that occur during ASIC manufacturing. These defects can be categorized either as hard or soft defects.

Hard defects are related to shorted (or bridged) interconnects, gate oxide defects, and open interconnects. Detecting hard defects is based on changes measured as logic-level voltages such as stuck-at-one (SA1) or stuck-at-zero (SA0). Although SA1 and SA0 fault testing has become standard, it has proven insufficient for thoroughly detecting all hard defects, as it only detects nodal defects.

The model for SA1 and SA0 fault testing (e.g. scan) uses zero-delay timing logic-level representation of a circuit at the primitive cell, or gate, level. Using this type of fault model, the flow requires running a set of test patterns that drive each circuit node high and low, and any SA1 and SA0 signal to the output pins. These output signals are then compared to an expected Good Known Signal (GKS) to detect defects. In the case of scan testing, these test patterns are generated automatically using specific software known as Automatic Test Pattern Generation, or ATPG.

ATPG software uses a fault model for the circuit under test and creates a list of all faults or fault dictionary. This fault dictionary lists all the fault information about each node in the design. Often the fault dictionary will be in condensed format, since redundant SA1 and SA0 faults for a given node in the circuit can be removed.

The information listed in the fault dictionary indicates whether the test pattern generated by ATPG tools has detected SA1 and SA0 at each node in the design. In the process of generating ATPG test vectors, at least ninety-seven percent, or better, test coverage of SA0 and SA1 is an industry-common practice for detecting nodal defects.

There are three distinct operations that take place when using ATPG to generate scan test patterns: register path conditioning (or scan mode), data capturing (or normal mode), and data shifting (or scan mode).

Register path conditioning is used to make sure all flip-flops in a given scan path are operating properly and are set to a known value (i.e. initialization state) before the actual scan test operation. During this mode, the circuit is brought up to scan mode and then a series of 1 and 0 data are serially shifted through all registers from the primary scan input until they reach the scan primary output (i.e. the scan chain is fully loaded) of the chip. This operation requires one scan clock for each flip-flop element in the scan chain.

In capture mode, the circuit is brought up to normal mode, and then the data from the combinational logic elements registers is clocked into scan flip-flops using scan clock while applying data at primary inputs of the chip.

In the last, and third, part of the scan operation, or scan mode, the circuit is brought up to scan mode and scan clock is used to unload the scan chain through the scan output. During this operation, captured data shifts out of the scan chain and the new data is loaded from the next test pattern into the scan chain by simultaneously asserting and deasserting the scan-enable signal on the falling edge of the scan clock to ease timing associated with hold time constraints.

Although one can achieve reasonable stuck-at fault coverage using scan techniques with the aid of ATPG, this method does not necessarily detect all possible CMOS defects such as voltage weakening or delay faults. These types of faults are commonly referred to as soft detects and are mainly timing related.

At-speed functional testing is the most often used test approach in detection of soft defects. At-speed functional test patterns provide an excellent defect detection mechanism for any design–if prepared properly.

In contrast with the hard defect model, the soft defect (slow-to-rise and slow-to-fall) fault model uses actual delay models for each element, or gate, in the logic-level representation, or structural netlist, to create a fault dictionary. The fault dictionary is used during the test program development to determine the quality of the test vectors (i.e. test coverage).

One of the advantages of at-speed test vectors is the by-product of design development and functional verification (development time) as discussed in Chapter 5. However, the disadvantage of this type of test vector is the difficulty in measuring fault coverage. The problem of measuring the fault coverage of functional, or at-speed, test vectors has led to the development of software tools that perform testability analysis, otherwise known as fault simulators.

Fault simulation allows testability (or controllability and observability) of the ASIC device to be analyzed during the design phase by identifying those areas of the design that have poor testability thus allowing the designer to act appropriately to improve the design testability.

6.5 Parametric Test

In addition to at-speed, or functional, testing, it is important to ascertain whether the ASIC design will operate within its specified AC timing or DC interface parameters. These types of test programs are known as parametric and test all critical AC and DC parameters to insure that the device will operate successfully over the recommended supply voltage and ambient temperature ranges.

AC performance can be obtained from at-speed, or functional, testing of the ASIC design if the ATE capabilities support this. AC parametric test vectors verify timing requirements such as propagation delay (or path delay) and maximum operating frequency of the design. Other timing related parameters, such as primary input setup and hold times or propagation delaya of the primary outputs, may need to be tested and measured separately. This This can be accomplished by choosing an appropriate location in the test vector and programming the ATE to measure the relevant AC parameters.

As design complexity and gate counts increase with process technology, AC testing for controllability and observability of the critical timing path can present a limitation. Therefore, it is important to be able to expose critical paths automatically (similar to scan ATPG) in generating AC parametric test programs.

DC testing is intended to test for adherence to primary input and output voltage and current requirements. Testing a single DC value often requires a set of test programs--some to set up output pins, others to create the signal value being measured. Thus, the fundamental requirement for DC testing is to be able to measure all DC parameters on each primary input and output as well as power supply currents for a given ASIC design.

It is desirable to have an ATE that is capable of logically operating the device at maximum operating frequency, measuring DC parameters, while continuously varying operating conditions such as voltage and temperature. However, in practice this is impossible; therefore, one may select a few operational conditions, such as minimum and maximum voltage and temperature.

One of the DC test requirements is to set each buffer in the I/O pad to all available states. It is important to note that the DC parametric testing is not performed for missing toggle states, rather it measures the electrical property of each I/O pad in the design. DC specified parameters can only be carried out with a detailed knowledge of the I/O pad circuits and characteristics of the process used to fabricate the ASIC design.

6.6 Current and Very Low-level Voltage Test

Current (or I_{DDQ}) and Very Low-level Voltage (VLV) testing can detect certain defects and degradations that cause parametric failures (or soft defects), but are not detected by functional at-speed test programs. These types of soft defects can lead to timing failures that make a circuit fail to operate at the designed speed while being functional at a lower speed [3].

I_{DDQ} testing and VLV offer alternative testing strategies that can detect many additional defects and reduce defect levels by an order of magnitude. Conventional test programs are the best for detecting some defects such as interconnect shorts and opens, while I_{DDQ} and VLV testing are best for detecting the other defects such as via defects, stress voids, increase in interconnect delay due to electromigration damage, threshold voltage shift, gate oxide shorts, and tunneling effect.

I_{DDQ} testing measures current faults and is mainly used for CMOS processing due to its low quiescent current requirement. In general, current faults (or excessive current) are observed when the low quiescent current of a CMOS device abruptly increases due to an undesired conducting path (or short) between power and ground.

The current fault model for CMOS devices is considered to be a transistor-level model of a circuit with each transistor (PMOS and NMOS) operating with proper quiescent current once they have completed their logical transitions.

I_{DDQ} testing requires executing a set of test patterns on the ATE that drive each circuit node to a high and low value. After applying each test pattern, the current through the power supply pads (or VDD) is measured after allowing the logic transitions to complete. These currents are then measured against limits set by normal static current of the ASIC to detect faults.

The advantage of these types of tests is that they are much less pattern-sensitive in comparison with the scan or functional tests as faults are visible through power pads and do not have to be logically propagated out of a device. Therefore, it is not necessary to create a known good output test program.

I_{DDQ} tests are considered to be more suited for detecting faults such as bridged interconnects, gate oxide shorts, and tunneling opens (a very thin layer of oxide located in via or contact causing transistor gates to be disconnected) that cannot be detected by stuck-at fault techniques.

Stuck-at fault tests, such as scan testing, usually detect SA1 and SA0 faults, but often miss bridged interconnects. This is because the bridged interconnects may have enough resistance to avoid a functional problem and, consequently, avoid detection. I_{DDQ} testing detects bridging faults more efficiently, since it is sensitive to high or low resistance bridges and is not dependent on functional failure.

Similarly, gate oxide shorts and tunneling opens can occur when the device is functioning and, therefore, are undetectable through stuck-at fault test detection. However, that same tunneling open or gate oxide short can cause a pair of PMOS and NMOS transistors to conduct simultaneously, thereby increasing quiescent current and consequently being detectable by I_{DDQ} testing.

VLV test is applied to a circuit at operating voltages much lower than its normal operating voltage in order to detect circuit failures. VLV test programs use a set of test patterns along with a predetermined supply voltage to execute the test. The speed of the test vectors and the value of programming voltage are determined from characterizing a known good device from various manufactured wafers. Although VLV can detect more soft defects as supply voltage reduces, it fails to detect interconnect delay failures because the interconnect delay does not scale down when the supply voltage is reduced.

The minimum supply voltage required by VLV test identifies the lowest supply voltage at which the ASIC design can operate properly. In the CMOS process, this low voltage level is normally two to three times higher than the transistor threshold voltages.

To determine the lowest operating supply voltage (or reference voltage), VLV test begins with a very low supply voltage such that the device cannot function properly (in voltage failure mode), and then the supply voltage is increased by defined intervals until the device passes the test. This procedure is repeated several times using various known good ASIC devices from the manufacturing wafers and lots to construct minimal supply voltage distribution for analysis to determine the minimum supply voltage level.

For today's deep submicron ASIC designs, the interconnect delay is no longer negligible and intensive buffer insertion is used during physical design to reduce long interconnect delay. Using I_{DDQ} and VLV testing in conjunction with traditional stuck-at fault testing can improve manufacturing fault detection because of the ability to detect (slow-to-rise and slow-to-fall) defective buffers.

6.7 Wafer Acceptance Test

Wafer Acceptance Testing, or WAT, is commonly referred to as element evaluation of qualification. It is a sampling method that is used to determine if the manufactured ASIC device will meet the design specifications.

WAT provides an economical means for both the semiconductor manufacturer and ASIC providers to separate wafers that are likely to have higher yield from those that will not.

There are two important aspects of using WAT data. One involves the evaluation of ASIC manufacturer processes and the second monitors the process during ASIC manufacturing.

For process evaluation, WAT data supplied by the semiconductor manufacturer is compared with various process parameters such as transistor characteristics, resistance, capacitance, and noise figures. This allows one to understand the correlation between the actual process and models, as well as to make sure the process meets ASIC design physical and electrical specifications.

Furthermore, the ASIC design team should evaluate the semiconductor process capabilities for wafer acceptance. Often wafer acceptance is accomplished by screening several simultaneously fabricated wafers containing test chips in order to predict if a set of wafers (or lot) will yield an adequate and acceptable percentage of good parts or dice.

By measuring PMOS and NMOS parameters such as delay, and comparing those with the model, one can observe if the process behavior is in line with the model or not. PMOS and NMOS transistors are simulated for various conditions such as fast-PMOS/fast-NMOS, fast-PMOS/slow-NMOS, slow-PMOS/fast-NMOS and slow-PMOS/slow-NMOS using models (e.g. SPICE). The results (as indicated by the indices of the trapezoid shaped region shown in Figure 6-9) are plotted along with measured WAT data.

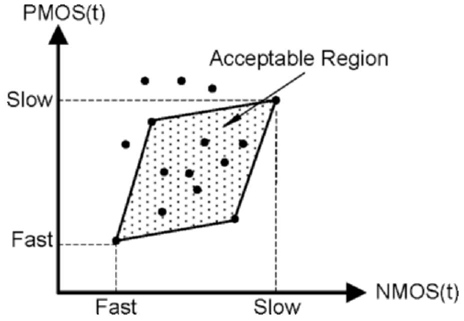


Figure 6-9 Transistor Simulation Results versus WAT Data

In Figure 6-9, the data points inside the acceptable region are an indication of process and model alignment. However, the data residing outside the acceptable region represents mismatches between the process and the models.

From a production point of view, these types of mismatches between process and models need to be resolved. One resolution is to extend the acceptable region in Figure 6-9 so that all the outlying data is covered by adjusting the PMOS and NMOS transistor models. This approach may not be desirable because it requires recharacterization of the entire spectrum of ASIC libraries (standard cells, memories, and intellectual properties) and can have a negative impact on the existing ASIC design. Another solution to the problem is to modify the current process to match existing transistor models. This method is preferred because there are no model or library changes required.

WAT data is generated through visual inspection or parametric testing. The most common process parameters are as follows:

- Critical dimension (e.g. Polysilicon length)
- Oxide thickness
- Thin film thickness
- Sheet resistivity
- Current and voltage shift
- Current leakage
- Mask alignment
- Particle defects
- Under/over etching
- Capacitance/resistance mismatch
- Diode characteristics (PMOS and NMOS performance)

After the ASIC is submitted to the manufacturer for processing, two physical rings will be added to the actual ASIC device before masks are produced. These rings are called the seal ring and scribe line as shown in Figure 6-10.

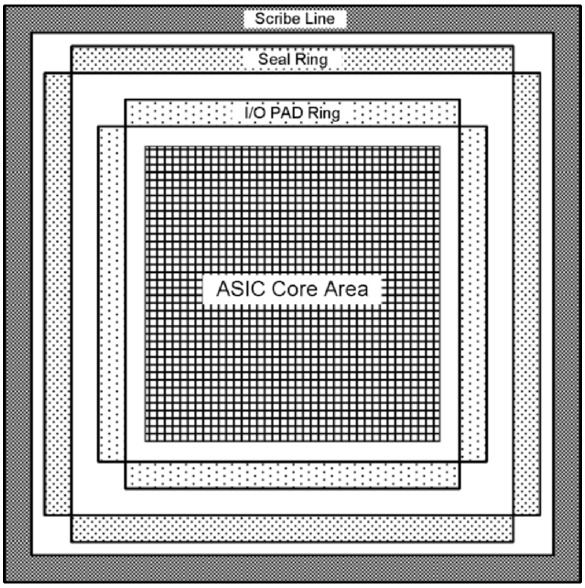


Figure 6-10 ASIC Seal Ring and Scribe Line

The seal ring joins the package case header and substrate to their covers to prevent penetration of moisture into the package. The scribe line is used to separate a processed wafer into individual dice.

Scribe lines, or narrow channels between individual dice, are mechanically weakened by scratching with a scribe (diamond tip), sawing with a diamond blade, or burning with a laser. The scribe lines are then mechanically stressed and broken apart along the scribe lines thus separating the individual dice.

To save silicon area, almost all ASIC manufacturers use scribe lines as a vehicle to add their process-monitoring test structures in order to collect WAT data during the device fabrication process.

In recent years, ASIC and semiconductor industries have intensified their attention to yield issues in order to meet the challenges of manufacturing complex ASIC (at 130nm technology and below); thus, monitoring WAT data during design production is key to insuring high yield.

6.8 Memory Test

Memory Built-In Self-Test (or MBIST) has been proven to be one of the most comprehensive and effective test methodologies widely used for embedded memory testing.

MBIST is very cost-effective because it does not require any external test hardware and requires minimal development effort while allowing in many cases pinpoint location of defects during testing. However, MBIST has an overhead area penalty and may have some negative timing impacts on the memory performance because of its required logic circuitry.

To use MBIST methodology effectively, there are some criteria such as quality and efficiency, which must be considered.

MBIST quality refers to its level of test coverage in detecting stuck-at faults.

If the MBIST testing algorithm does not detect the vast majority of defects, it can influence the quality of the ASIC design and could lead to undesirable engineering and business results. MBIST quality is considered the most important criterion in MBIST methodology.

Efficiency criteria correspond to the area overhead integration of MBIST circuitry in the ASIC design, and its test algorithms' run time. Inefficient MBIST can increase the overall ASIC design area and significantly increase the test time for designs that have many embedded memories.

The basic MBIST configuration consists of multiplexers, data and address generators, control logic, and comparator logic, as illustrated in Figure 6-11.

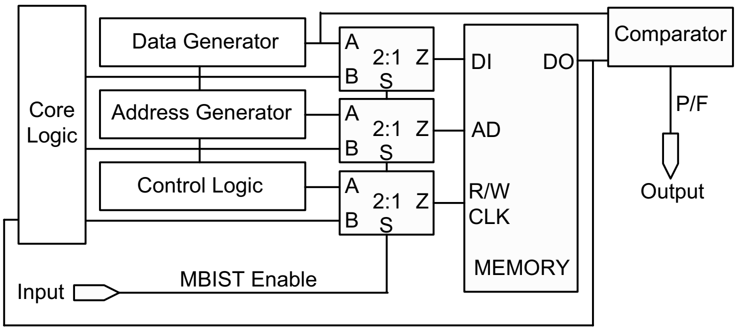


Figure 6-11 MBIST Logical Presentation

During the MBIST operation, the memory is put into a test mode by the use of multiplexers connected to data, address, and control lines (such as read and write lines), to allow MBIST to exercise the memory independent of the rest of the design. Then, a finite state machine that resides inside the control logic provides the necessary states and signals for the data and address generator logic to write a test pattern in a March algorithm to the memory cells, read the data back from the memory, and compare it to the original data. If a mismatch occurs during the operation, a flag is set to show that the memory cell under test has a failure.

It is important to note that although many EDA tools employing MBIST have developed algorithms that are optimized to test a large set of fault types while maintaining reasonable test times, they may not be able to detect faults such as soft defects.

6.9 Parallel Module Test

The Parallel Module Test (PMT) technique allows test vectors to be applied to blocks or modules within the design and observes their response directly using ASIC primary input and output package pins.

PMT uses multiplexers that can be inserted in a series with a module's normal input and output ports to allow reconfiguration of the interconnect routing of a module from its normal functional routing path to a test access path.

Figure 6-12 illustrates PMT configuration of an ASIC design that consists of a memory, hard macro or analog block, and other core logic circuits.

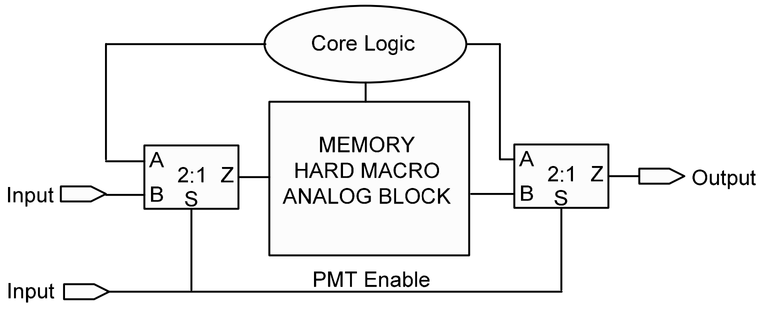


Figure 6-12 Parallel Module Test Logic

During PMT test, the PMT enabled signal sets the multiplexers such that the module under test can be accessed directly from the ASIC primary input and output.

PMT is very useful for memory testing where the area and performance are impacted by MBIST. In addition, PMT can be advantageous from a characterization point of view if an ASIC design has embedded analog modules. By allowing the analog module to be controlled and observed directly via primary input and output pins, PMT reduces test time significantly.

6.10 Summary

In this chapter, we have briefly discussed the most common ASIC testing techniques such as functional, scan, boundary scan, parametric, wafer acceptance, memory, and parallel module test.

Today's ASIC designs without design-for-test can be very expensive and may not meet the high-reliability requirements. We should recognize that all design-for-test techniques require design trade-offs such as area, design cycle time, or extraction processing steps. Economical feasibility of design trade-offs must be carefully evaluated in relation to test development cycle time, test time reduction, as well as the level of fault coverage achieved.

In the functional test section, we outlined the importance of proper circuit initialization, use of multiplexers, and basic design-for-test requirements. Functional testing can involve either manual or automated methods. Although manual testing methods are appropriate for some designs, they are time-consuming and inefficient for complex ASIC requiring shorter test development cycles. Automated testing, on the other hand, can allow increased test coverage and shorter development cycles that may not be possible with manual testing.

In the scan and boundary scan test section, we discussed the concept of such testing techniques and their operation under testing conditions. We should note that although scan and boundary scan improves the stuck-at coverage, it may not be sufficient to cover many timing-based defects. To address this inefficiency, scan-based ATPG solutions for at-speed testing are required. Having such a solution not only ensures high test coverage of detecting timing-based defects, it also reduces testing time.

In the parametric testing section, we covered the concepts of AC and DC testing. These types of tests are performed after wafer manufacturing is completed. During parametric testing, various technological and process parameters are measured and compared against specifications to ensure proper device manufacturing.

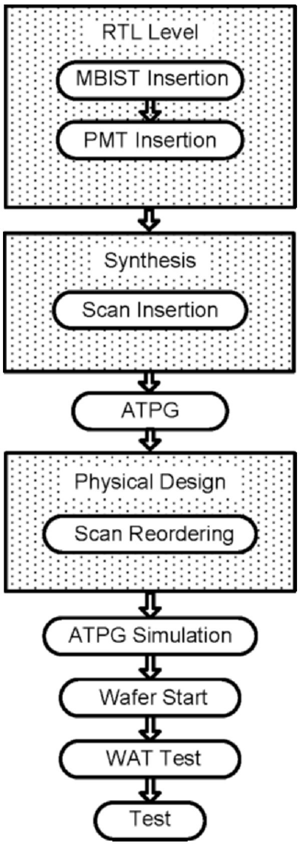


Figure 6-13 Test Development Steps during Design Phase

In the current and very low-level voltage test section, we outlined alternative testing strategies that can detect additional defects such as tunneling opens that cannot be detected by functional or at-speed testing. These types of process-induced defects are major types of defects for submicron processes and we need to be able to detect such defects in order to improve the device reliability.

In the section on wafer acceptance test, we mentioned the importance of wafer acceptance test during the process evaluation as well as process monitoring vehicles for yield enhancement. In addition, we discussed the use of scribe lines as process monitoring elements. We should note that having comprehensive test programs has a profound impact on yield improvement, especially for deep submicron processes.

In the memory and parallel test section, we discussed the advantage of these tests for design-for-test improvement. In addition, we covered automatic test generation for memories. From a physical design point of view, having this testing capability allows one to reduce the test development cycle time especially if the design contains a large number of memories.

Figure 6-13 shows the most common test methodology utilized during the ASIC design test program development phase.

References

[1] E.B. Eichelberger and T.W. Williams, “A logic design structure for LSI testability”, *Journal of Design Automation and Fault Tolerant Computing*, pp. 165-78, 1978.

[2] ANSI/IEEE Standard 1149.1-1990, *IEEE Standard Test Access Port and Boundary-Scan Architecture*, IEEE Standards Board, New York, N.Y, May 1990.

[3] P. Franco “Testing Digital Circuits for Timing Failures by Output Waveform Analysis”, *Center for Reliable Computing Technical Report,* No. 94-9 Stanford University, 1994.

Index