

Veena S. Chakravarthi  
Shivananda R. Koteswar

# SoC Physical Design

A Comprehensive Guide



Springer

# SoC Physical Design

Veena S. Chakravarthi • Shivananda R. Koteswar

# SoC Physical Design

A Comprehensive Guide



Springer

Veena S. Chakravarthi   
Co-founder and Adviser  
Sensesemi Technologies Private Limited  
Bangalore, India

Shivananda R. Koteshwar   
Founder Trustee  
Belakoo Trust  
Bangalore, India

ISBN 978-3-030-98111-2      ISBN 978-3-030-98112-9 (eBook)  
<https://doi.org/10.1007/978-3-030-98112-9>

© The Editor(s) (if applicable) and The Author(s), under exclusive license to Springer Nature Switzerland AG 2022

This work is subject to copyright. All rights are solely and exclusively licensed by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG  
The registered company address is: G  werbestrasse 11, 6330 Cham, Switzerland

*Dedicated to backend designers*

# Foreword

I am pleased to write the foreword for this monograph on the physical design of system on chip (SoC) design. Professors Veena S. Chakravarthi and Shivananda R. Koteswar have put together a very readable summary of the main issues involved in current and future SoC designs.

Almost everyone on the planet now benefits from using CMOS SoCs daily, be they in mobile phones, tablets, PCs, laptops, vehicular electronics, or home entertainment systems. The demand for ever-increasing complexity SoCs is ongoing.

So it is fitting that a text like this exists as a broad introduction to the technology. The narrative is clear and to the point, supported by good illustrative diagrams.

I commend this text to anyone wishing to understand the technology “under the hood” of the electronic devices they use every day. For those considering a career in this area, this is a great text with which to get started.

**Neil H. E. Weste** is an Australian inventor and author, noted for implementation of a Wireless LAN chips and for authoring the textbook Principles of CMOS VLSI Design, which is standard text book for VLSI courses. He was a co-founder of Radiata Communications, which produced the world’s first Wi-Fi 802.11a chip in Australia, before being acquired by Cisco. He is fellow at Morse Micro, Sydney, Australia.



Inventor and Author, CMOS VLSI Design:  
A Circuits and Systems Perspective; Fellow at Morse Micro  
Surry Hills, NSW, Australia

Neil Weste

# Foreword

We are aware of the significant disruptions that the pandemic has caused in both our personal and professional lives. At the same time, the semiconductor industry has been boosted tremendously due to the changing and seemingly irreversible consumer habits during the pandemic. The growth in the industry has been driven by the rise in remote work, distance learning, gaming, entertainment, and internet shopping. This has significantly increased demand for consumer electronic devices to the point that demand for semiconductors has far outpaced production. The industry has assumed more importance than ever before, and there is a positive impact on the investment, growth, and adoption of 5G, artificial intelligence, and the Internet of Things.

This change has put a fresh impetus on the talent required to fulfill the demands of the industry. The widening talent gap is one of the major global issues for the semiconductor industry today. *SoC Physical Design: A Comprehensive Guide* is a very timely book that can help fresh as well as experienced engineers and designers to produce superior work towards the design of system on chip (SoC). The book contains topics pertaining to physical design methodologies, SoC design goals, and details about relevant tools, algorithms, and skills required for the physical design and development of SoC. It ensures that engineers are able to contribute effectively to design projects for the first-time success of complex SoC designs.

Dr. Veena S. Chakravarthi, the author of this book has been working in the semiconductor industry for more than three decades in a range of public sector, service, and product companies. She has managed tapeout projects for complex SoCs in wireless, optical, TV, and communication domains. She also has rich experience in academia as a researcher and professor. With this experience, the kind of insight that she has been able to bring in this book is unique and very valuable for the reader.

Dr. Shivananda R. Koteswar, the co-author of this book is my long-standing friend and an industry veteran with over 25 years of technology leadership in the semiconductor industry with expertise in full-chip and complex IP implementation, verification, and design methodologies to deliver multiple first-time-right silicon ASICs. He wears multiple hats and has a special place in his heart for imparting

education. This passion and knowledge have helped in adding tremendous value to this book.

Today, our everyday lives have been surrounded by semiconductor-enabled smartphones, smart TVs, routers, smart speakers, tablets, laptops, auto, virtual reality headsets, and much more making our lives smarter. The semiconductor industry is poised for significant growth during the next decade. Technology inflections such as 5G wireless, artificial intelligence, Internet of Things, cloud computing, and machine learning are driving up the long-term demand for the chip industry. The innovations that we will see in the future are yet to be imagined and are slated to further fuel the growth of this industry. So, the chip designers need to equip themselves with all the skills required to make that happen and this book is one important step in that direction.

We are indeed heading towards very exciting times!



**Anku Jain** is heading MediaTek's pan India operations with offices in Noida, Bangalore, and Mumbai. MediaTek Inc. is a leading Taiwan-based global fabless semiconductor company that powers nearly two billion devices a year and holds a leading position in several semiconductor verticals.

Over the last decade, Anku has established MediaTek as a key player in the Indian market and helped democratize the face of the Indian mobile ecosystem. Anku's expertise includes business leadership and working with cross-cultural teams across India, the USA, China, and Taiwan in the areas of embedded software, semiconductor, telecom, and mobile phone design. He holds multiple patents in the field of mobile telephony.

He is also an avid angel investor in several Indian tech startups in domains such as AI, Edtech, Fintech, and SaaS in both B2B and B2C businesses. This is with a firm belief that India being a surging economy is going to produce the next lot of unicorns that will compete globally.

Anku brings almost three decades of a rich and unique experience, having worked with the software product industry and in the semiconductor space. Prior to MediaTek, he has worked with technology firms like Sprint, RiverRun, and Network Programs before being part of a startup Pixel Communications as a founding member that got acquired by MediaTek for IP. Anku with multi decades of experience in the Semiconductor Industry understands ground realities and is a visionary leader.

Managing Director, MediaTek India  
Gurgaon, Haryana, India

Anku Jain

# Foreword

It is always an exciting time to be in the semiconductor industry and this is especially true today. AI, IoT, and 5G technologies have proliferated our lives like never before. The appetite for digital consumption is at a record high and is driving technical innovation from huge data centers to smart everything. Every device needs to be smaller, more powerful, self-learning, and smarter than the previous generation. SoCs are at the heart of the advancement in medical transportation, telecommunication, and countless other technologies that will change how we live. Industry challenges, such as the current shortage of chips, which are expected to last until 2023, because for us to find faster, smarter, and more innovative ways and means to develop chips. The semiconductor industry has never been more complex and it is going to keep getting more complicated. With this evolution comes the opportunity for great reward in making our world a better place for all its inhabitants.

The semiconductor industry is at the leading edge of AI technology, and the demand for such intelligent, highly integrated chip-enabled solutions is growing. Any aspiring hardware engineer, whether they want to work for an eager, young startup or an established house of silicon, needs to become fully versed in the art of very large-scale integration (VLSI). In this context, this book is very relevant and practical as it is authored by practicing chip developers themselves.

Veena and Shivoo have been in VLSI chip design from the time chips were hand-crafted to the present, where intelligent automation tools design chips on their own. However, chip design continues to be a complex art and a science.

With cumulative 60 years of experience in semiconductors, both have demonstrated their abilities to design large, complex electronics systems in silicon, developing a baseline, and enabling technologies for several systems. They clearly understand the mindset and skill set needed to accomplish chip design of any complexity, and there aren't any better at teaching these concepts.

This book is a thoughtful guide for any aspiring chip designer. It uses a practical approach that contains straightforward applications of known techniques to create a structure that will help freshman engineers contribute effectively to the SoC physical design and development process.

I appreciate Veena and Shivoo for teaching the next generation of innovators, inventors, and dreamers.

**Naveed Sherwani** is a renowned semiconductor global leader with over 35 years of experience in entrepreneurship, technical engineering, and leadership. He has founded over 10 companies and raised over \$800 M in multiple rounds of fundraising. Dr. Sherwani has authored several books and over 100 articles on various aspects of VLSI physical design automation and ASICs. He earned his PhD from the University of Nebraska-Lincoln and also served as a professor at Western Michigan University where his research focused on ASICs, EDA, combinatorics, graph algorithms, and parallel computing.



Rapid Silicon  
San Jose, CA, USA  
Dec 2021

Naveed Shervani

# Preface

The semiconductor industry is at an inflection point due to two reasons: one being, high technology innovations fueled by AI, ML, and 5G, and the second for the acute global shortage of VLSI chips which has impacted the entire technology supply chain. Multiple markets like automobiles, computing, and gaming are struggling to source chips and other related materials even as prices skyrocket. Another compounding factor to the current semiconductor shortage is the dearth of tools and human resources tasked with designing extremely complex semiconductor chips of the future.

Another trend that has a tremendous impact on the Semiconductor Industry is the evolving nature of physics. The radical shift from the physics-based limits of Moore's era of scale to the high-level solution complexity of new-age systems.

Having worked in the semiconductor design and EDA industry cumulatively for over four decades, the primary objective of the book was to pass on the knowledge of VLSI design to the next generation.

The book presents a comprehensive overview of the design methodology, development environment, tools, and challenges posed by system design, and the skills required for the physical design of complex semiconductor systems. The scope of the book is limited to the physical design or backend design of SoCs with an emphasis on practical design and tool flow. The design methodology dealt with in this book is generic and easily adaptable to any of the preferred standard EDA toolsets.

It ensures that engineers are aware and can contribute effectively to the much-talked AI revolution in complex systems on chips.

While the book is targeted to all engineers (electrical, electronics and communication, computer science, and product design) who aspire to be VLSI designers, it's also a valuable reference guide for professional designers who are part of semiconductor design, development, manufacturing, and fabrication houses.

Typically, most engineers aspire to be VLSI designers after their graduation. Unfortunately for them, they usually don't possess the requisite skills and tool-driven design techniques to circumnavigate the challenges they'll face in implementing the solutions. Meanwhile, young system designers struggle to see the big picture of the SoC design process. This book is our attempt to provide answers to

both groups so that they can plan, understand, and equip themselves with the necessary skillsets for physical design. The highlighted challenges and probable approaches to address them in each phase of SOC design are the leading lights.

This book makes a ready candidate for curriculum offered during engineering courses at all levels to make them industry/research-ready in SoC design.

It would not have been possible to realize this project without the support of many of our friends, colleagues, and family. First, we wish to thank our common instinct and desire to share the acquired knowledge with the coming generation. Our heartfelt thanks to our respective loving families, who encouraged us in this endeavor.

We are indebted to our reviewers, who patiently read each of the book chapters and offered line-by-line reviews.

And we are mighty thankful to Dr. Neil Weste, CSO, Morse Micro, and beloved author for taking time out of his busy schedule to write the foreword for this book. It is a full circle as we were inspired and referred to his work during our studies and at work. We are also thankful to Mr. Anku Jain, Managing Director, MediaTek, India, and Mr. Naveed Sherwani, Chairman, President, and CEO, Rapid Silicon for being kind enough to write the forewords to this book despite their busy schedules.

About the book organization, chapters “Introduction to Design of System on Chips and Future Trends in VLSI” and “SoC Physical Design Flow and Algorithms” set the context by giving an overview of the SoC design methodology and introducing physical design flow in the SoC design context. “Floor Plan and Placement of SoC Design” deals with the floorplanning and placement flow, “Clock Tree Synthesis (CTS) in SoC Physical Design” explains clock tree synthesis, “Routing in SoC Physical Design” covers design routing methods in detail. “System on Chip Design Finishing and Design for Manufacturability DFM” covers the design finishing and tape outflow. Physical design verification and design signoff are dealt with in “Physical Design Verification of SoC”. “Advanced Packages and 3D-SoC Designs” explains the chip packaging and introduces the concept of 3D IC design. The last chapter on “Current trends of semiconductor systems and physical design” is a short chapter to motivate engineers to help push the envelope of technological advancement in the development of future systems. Care is taken to ensure that with minimal understanding of the VLSI design flow, physical design is understood in the most comprehensive and practical way. At the end, a list of self-assessment questions is given for readers to check their understanding of the subject. As a bonus, a short notes on current trends and future direction is presented.

We will be very happy if users find each chapter useful and subsequently turn out to be VLSI engineers. We are curious about your feedback and criticisms. We welcome them.

Bangalore, India

Veena S. Chakravarthi  
Shivananda R. Koteshwar

# About the Book

This book presents a comprehensive overview of a physical design of system on chip (SoC). Practical physical design process with the required fundamentals help engineers to be effective at work. A short note on challenges of SoC physics design in the chapters help everyone to visualise real life design scenarios.

Veena S. Chakravarthi  
Shivananda R. Koteswar

# Contents

<b>Introduction to Design of System on Chips and Future Trends in VLSI .</b>	1
System on Chip (SoC) .....	1
SoC Trends .....	2
Components of SoC .....	5
Analog Block .....	5
Digital Cores .....	7
Memory Blocks.....	7
Macros.....	8
Design Abstractions of SoC Design .....	8
Abstractions .....	8
Integration of SoC Design Components .....	8
Trends in SoC Design Methodology.....	8
SoC Design Styles.....	10
Physical Design (PD) of SoC .....	11
Fundamentals of Physical Design.....	11
Future Trends of Systems .....	18
References .....	20
<b>SoC Physical Design Flow and Algorithms .</b> .....	21
System on Chip (SoC) Physical Design .....	21
SoC Core Design and Input-Output (IO) Design .....	21
Physical Design Flow of SoC .....	22
Floor Planning of SoC Design .....	22
Placement of SoC Design .....	27
Clock Tree Synthesis.....	28
Global and Detailed Routing .....	29
Physical Design Verification .....	29
Design Rule Check (DRC) .....	29
Layout vs. Schematic (LVS).....	30
Logical Equivalence Check (LEC).....	30
EDA Tools for Physical Design .....	30

<b>Floor Plan and Placement of SoC Design</b>	31
Floor Planning of SoC Design	31
Design Partitioning	31
SoC Floor Plan	34
SoC Power Plan	36
Placement Considerations	38
Two-Step Synthesis of SoC Design	38
Placement of SoC Design	38
Placement Methods	40
Design Verification Checklist for Final Floor Plan	40
Input for Floor Plan	40
Floor Plan and Placement Validation	46
EDA Tools in Physical Design	47
Challenges in SOC Design Floor Plan and Placement	48
References	48
<b>Clock Tree Synthesis (CTS) in SoC Physical Design</b>	49
Digital Logic in SoC Design	49
Sequential Cells in Digital Logic Design	50
Pipelining	50
Need for Clock Distribution	52
Clock Parameters	52
Clock Tree Synthesis	53
Design Constraints	54
Maximum Delay Constraint	54
Minimum Delay Constraint	56
Quality of Design	56
CTS Topologies and PPA	57
Types of Clock Trees	59
Clock Tree Algorithms	61
Design Challenges in the Clock Synthesis	62
<b>Routing in SoC Physical Design</b>	65
Design Routing	65
Major Challenges in Signal Routing	67
Design Readiness for Signal Routing	67
Signal Routing Steps	68
Signal Routing Algorithms	71
Maze Router or Lee Router Algorithm	72
Routers in Physical Design Tools	74
Design Signal Routing for Good Timing Performance	76
Signal Routing for Signal Integrity in Designs	76
Design Challenges in Signal Routing in SoC Designs	79
References	80

**System on Chip Design Finishing and Design for Manufacturability**

<b>DFM</b> .....	81
Reliability and Design for Manufacturability (DFM) of SoC .....	81
Signal Latch-Up in CMOS Standard Cells .....	81
Antenna Effect .....	85
Cross Talk .....	86
<i>IR</i> Drop .....	88
Electromigration (EM) Issue .....	91
Fabrication Issues Leading to Signal Integrity and Reliability Issues .....	92
Tap Cells .....	93
Inserting Boundary Cap/End Cap Cells .....	94
TiHi and TiLo Cells .....	95
Decap Cells .....	97
Spare Cells in Physical Design .....	99
References .....	100

**Physical Design Verification of SoC** .....

Physical Design Verification .....	101
Static Timing Analysis (STA) .....	101
Design Sign Off .....	103
ECO for Design Fix .....	103
EDA Tools for Sign Off .....	105
Physical Design Verification by Formal Methods .....	105
Model Checking .....	105
Logic Equivalence Checking (LEC) .....	107
Electromigration .....	110
<i>IR</i> and Cross Talk Analysis .....	111
Gate Level Simulation .....	112
Electrical Rule Check (ERC) .....	112
Design Rule Check (DRC) .....	114
Design Rule Violation (DRV) Checks .....	118
Design Tape Out .....	119
SoC Design Verification Challenges .....	119
Available Solutions .....	120
Machine Learning in Physical Design Verification .....	120

**Advanced Packages and 3D-SoC Designs** .....

SoC Packaging .....	121
Trends in Packaging .....	121
Concept of Stacked Dies .....	122
3 D Integration Schemes .....	123
3D IC Constituents .....	124
EDA Tools for 3D ICs .....	126
Early Adopters of 3D-IC .....	128
Future Trends of 3D IC Technology .....	129
Reference .....	129

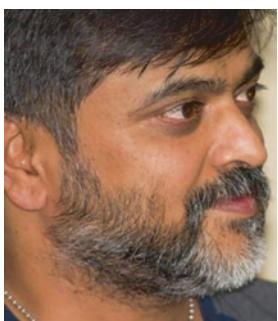
<b>Current Trends of Semiconductor Systems and Physical Design . . . . .</b>	131
Current Trends in Design Methodologies . . . . .	131
Optimal Logic Structure to Achieve PPA Goal . . . . .	132
Optimal Macro Placement . . . . .	132
Route Aware CTS, Pre-route to Post-route Correlation (Timing, Congestion) . . . . .	132
System-Level Technology Trends . . . . .	135
Future Trends of Systems [3] . . . . .	135
References . . . . .	136
<b>Question Bank . . . . .</b>	137
Answers . . . . .	146
<b>Index . . . . .</b>	151

## About the Authors



**Veena S. Chakravarthi** is a Bangalore-based technologist, system on chip architect, and educator. Over a career spanning three decades, she has spawned several VLSI Design and Incubation centers and managed several high-performance tech-teams at ITI Limited and across various MNCs like Mindtree Consulting Pvt. Ltd., Centillium India Pvt. Ltd., Transwitch India Pvt. Ltd., Ikanos Communications Pvt. Ltd., Pereira Ventures, Asarva Chips and Technologies, and Saankhya Labs and Prodigy Technovations Pvt. Ltd. She has been the Research Head and Adjunct Professor in the

Department of Electronics and Communication Engineering, Bangalore. She is passionate about promoting VLSI research and development in academics. She has mentored many fresh engineers and guided them to pursue their career in Semiconductor Industry. She has guided two research scholars for their PhD degree. She holds a PhD from Bangalore University for her work in low power VLSI research and is an alumnus of IIM, Bangalore. She is the author of books *A Practical Approach to VLSI System on Chip (SoC) Design* and *Internet of Things and M2M Communication Technologies*.



**Shivananda R. Koteshwar** has over 25 years of experience in the semiconductor industry. He is currently with Synopsys India as the Silicon Realization and Custom Design Manufacturing Group Site Leader and R&D Head. Shivoo is a Mentor of Change—ATAL Innovation Mission, NITI Aayog, Government of India, SIG Member ESSCI—Electronics Sector Skills Council of India, Startup Mentor in MEITY (Ministry of Electronics and Information Technology), and IET Skill Development Working Group member. He is a Bangalore chapter member at Singularity University.

and the founder Trustee of Belakoo Trust focusing on rural education, skill development, and experiential learning. He is one of the Trustees in Aurinko Trust with a flagship product, the Aurinko Academy, a progressive K12 school in Sarjapur Road with a career-focused programs in all streams. Dr. Koteswar has received a doctorate in Education Management and is an alumnus of the Indian Institute Of Management–Bangalore. He has a Postgraduate Diploma in Innovation and Design Thinking, a collaborative program by MIT Sloan, Columbia Business School, and Tuck University a Master’s degree in Electrical Engineering from the Oregon Graduate Institute of Science & Technology (OGI), a BTech in E&C from Mysore University, and has completed various certification courses from Tier 1 business schools in the United States and India.

# Abbreviations

AI	Artificial intelligence
API	Application programming interface
BG	Background
BS	Boundary scan
CDO	Clock and data path optimization
CTS	Clock tree synthesis
DRC	Design rule check
EM	Electromigration
FG	Front ground
GPIO	General purpose input-output
HAL	Hardware abstraction layer
HDR	High definition range
HMD	Head mounted devices
HMI	Human-machine interface
HTML	Hypertext markup language
HTTP	Hypertext transfer protocol
<i>IR</i>	Current-Resistance
JTAG	Joint Test Action Group
LWM2M	Lightweight Machine-to-Machine
M2M	Machine-to-machine
MAN	Short form for manual
ML	Machine learning
OSI	Open systems interconnection
PDK	Process design kit
QoS	Quality of Service
RAM	Random access memory
RPL	Routing
SG	Study group
SMS	Short message service
SOC	System on chip
Rx-Tx	Receiver-transmitter

UE	User equipment
UDP	User datagram protocol
UDP	User datagram protocol
UPM	Useful packages and modules
UI	User interface
UI and ML	User interface and machine learning
URI	User resource identifier
URL	Uniform resource locator
URN	Uniform resource name
USP	Unique selling proposition
USB	Universal serial bus
VLSI	Very large-scale integration

# Introduction to Design of System on Chips and Future Trends in VLSI



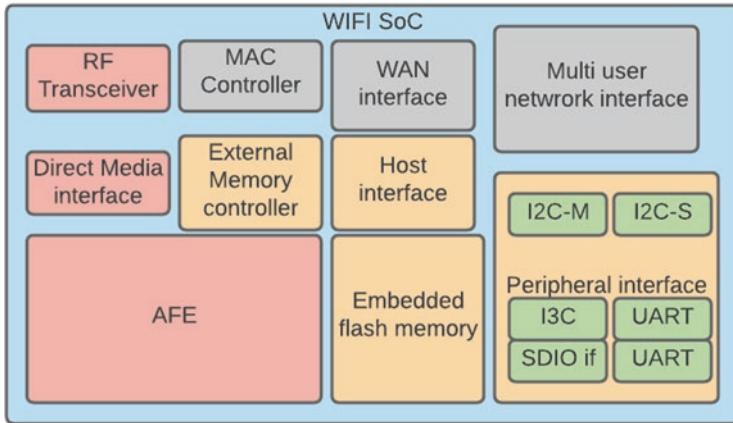
## System on Chip (SoC)

System on Chip (SoC) is an integral part of any electronic product today. All the electronic products ranging from large systems like data servers to mobile and sensor tags will have systems on chip (SoC) in it. They are used to process incoming signals, both in analog and digital forms, store them for future use or further process and analysis. The data stored is used to generate the output to control other systems or to derive meaningful information for the user. SoCs perform almost all functions of electronic products making the reliable and miniaturised.

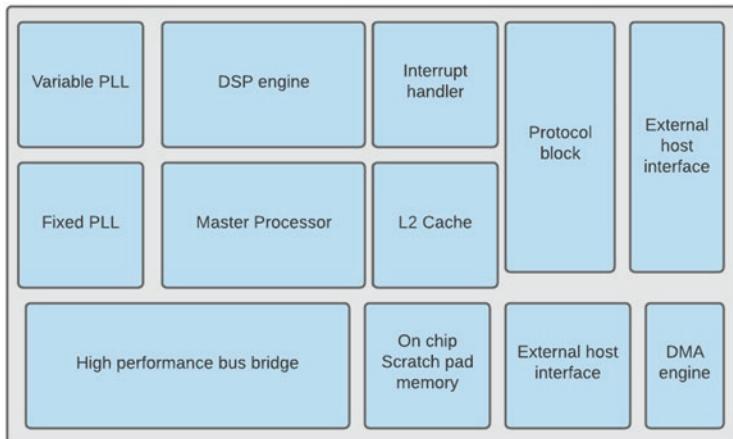
System on chip (SOC) contains logic blocks with many functions of the system except for a few interface blocks. These exempted functions are not realizable by the CMOS or CMOS compatible technologies [1]. CMOS and CMOS compatible technologies are MEMs, Bi-CMOS technology, Memory technology, etc. Typical interface functional blocks which currently are not part of any System on chip include the display screens, Keypads, Battery circuitry, some types of Antennas, etc. Figures 1, 2 and 3 are a few System on chips (SOC)s.

There are advanced SOCs which are far more complex than the ones shown in Figure 3, having hundreds of functional cores, peripheral interfaces, on-chip memories, and external expandable memories. SoCs are designed and fabricated using many advanced CMOS VLSI technologies which are nanotechnologies.

The process of designing these SoCs uses a combination of digital design techniques, mixed signal custom design, and FPGA design techniques.



**Fig. 1** WI-FI SoC



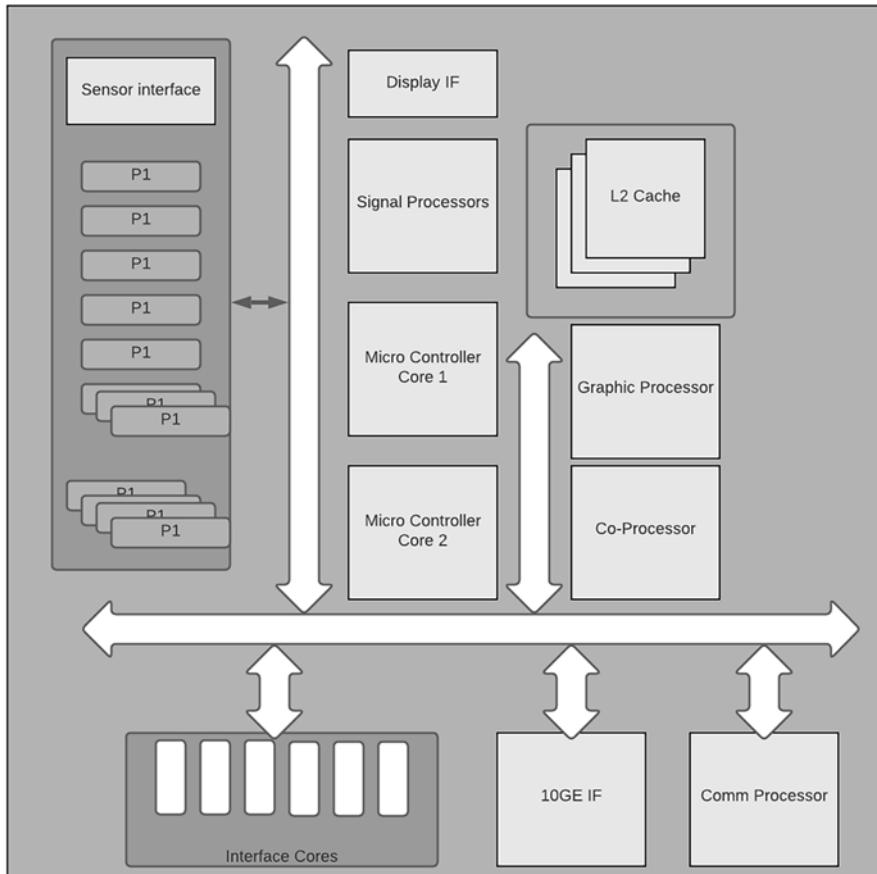
**Fig. 2** Microcontroller chip

## SoC Trends

Following trends are observed in the System on chip solutions:

### 1. SoC trends with continued device scaling

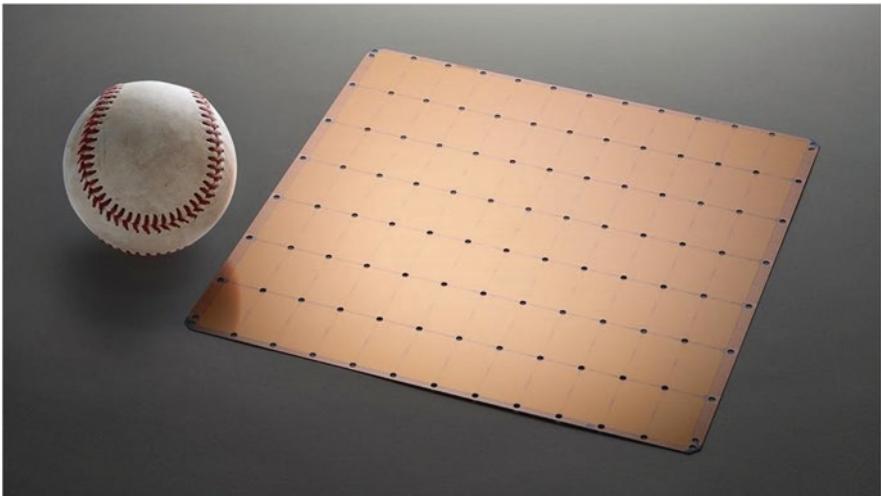
Moore's law of scaling continues to miniaturize the devices with advancement in lithography and other process technologies. Extreme ultraviolet (EUV) lithography double patterning techniques and advanced front end of line (FEOL) device architectures ensure the feature size is reduced further beyond 3 nm. This makes SoCs of higher packing densities possible. Packing density is number of transistors in square millimeter of silicon. This require very sophisticated design flow and EDA tools with advanced features to carry out design and verifications of complex SoC designs to achieve functional, performance specifications considering application, safety, security, fabrication, packaging and environment conditions and product reliability.



**Fig. 3** Cloud Processor SoC with Network interface

2. Denhard Scaling {move the next sentence to new line} Device feature scaling has continued over the decades and the kind of advantage seen in performance has slowed down in SoCs in recent transitions. This was predicted as Denhard scaling. When technology node to node changes, the performance characteristics of the SoCs are not scaling up at lower nodes below 5 nm. This has triggered new innovations in advanced devices, new device structures, new process technologies, newer integration methods, and advanced design methodologies and advancement in design platforms.
3. SoC designs with integration of heterogeneous functional cores

More and more SoCs are designed using heterogeneous integration of digital cores, denser memories, macros, analog, RF cores, on-chip antenna arrays and on-chip sensors developed using CMOS or CMOS compatible processes. These also use advanced interconnect and packaging technologies.



**Fig. 4** Wafer-scale large chip. (Courtesy: Cerebras systems)

#### 4. Growth of nonvolatile memory trends

Advent of magnetic RAM memories in embedded systems and traditional NAND and NOR technologies pushed to limit, the development of SoC demanded newer methods of integration. It is seen that various kinds of magnetic random-access memories (MRAM) including spin-transfer torque (STT)-MRAM, spin-orbit torque (SOT)-MRAM, and voltage controlled magnetic anisotropy (VCMA)-MRAM to potentially replace some of the traditional L1, L2, and L3 SRAM-based caches in the systems.

#### 5. Demand for edge AI SoCs

As opposed to cloud-based AI, the need for faster inference algorithms for real-time safety critical applications, AI SoCs for edge devices are seen to be trending technologies in years to come. The growth in IoT and smart communication devices with 5G technologies in the next few decades is expected to drive innovative SoC architectures. Analog compute-in-memory architecture is seen to be a promising SoC design trend in recent years. This will make every edge devices more powerful, intelligent and sophisticated.

6. Large compute and memory requirement in systems with AI and ML in datacenter applications occupy a large silicon area. Existing compute systems still require off-chip integration to build large systems required for such applications. An innovation in this direction is building large size system on chip. A recent example is a wafer-scale AI system on chip which is of palm size as shown in Figure 4.
7. Scalable CPU systems: With never diminishing demand for computation and large data processing in systems, horizontal scalability of the compute systems is much in need. Flexibility on need based compute resource utilization in cloud applications is enablers for large, modular, flexible compute architectures of unimaginably large systems. The compute resources like memory, Disk IO,

network IO, and CPU are the resources generally scaled up/down depending on the load. This often requires support in the system architecture which offers the flexibility of easy partitioning of data compute cores, cache, processing elements, and flexible need based add/remove interfaces. These systems are mainly used for cloud computing applications in data farms

8. 3D-IC: With ever growing need for integrating more and more heterogeneous features in system, another trend seen is integrating and packaging stacked dies fabricated using different fabrication technologies. Individual chips are referred as *chiplets*. Such chiplets are vertically stacked one above the other and interfaced using silicon *interposer*. Interconnection across dies are achieved using redistribution layers (RDL) which is another silicon chip in its own. This is the concept of 3D-IC. 3D-IC enables higher level of integration of memories systems and RF, Sensors, compute cores to develop systems of any specification with any complexity and more flexibility.

## Components of SoC

A typical SOC consists of one or more general purpose RISC processors which are digital dominant, one or more DSP processors, dense on-chip memory, protocol block, interfaces to external expandable memories, one to many external standard interfaces for off chip integration, house keeping functions like clock generation and stabilization blocks, power management blocks, Analog interfaces, User interface functions and Sensor/Radio interfaces to the external world depending on the applications [1]. In addition, a SOC invariably houses software functions like bootloader and embedded firmware and application softwares. Each of the constituents of SOC is developed independently using appropriate design techniques such as full custom design flow (Analog, mixed signal blocks, Phase Locked Loop (PLL) circuit, Pad circuits), automatic cell based design flow (digital SOC core), structured array-based design flow (embedded memory), and are integrated as single chip or packaged together with multiple dies stacked. A simple SoC with internal constituents is shown in Figure 5.

### Analog Block

Analog blocks process analog signals directly interacting with sensors or signal conditioning circuits or transceivers. Transceivers, Data converters, Filters, PLLs, Serdes, analog pads are examples of analog components in System. Physical layer functions of seven-layer OSI architectures, analog front end (AFE) of standard technologies like Ethernet, Wi-Fi, PCIe use analog cores. In earlier times, these circuits were realized using off-chip discrete components but with the advancement in process technologies, more and more analog circuits are integrated into systems on chip. They are high performance, low power, and small area (PPA) cores. These are much more susceptible to noise, distortion, and other errors. The design of analog

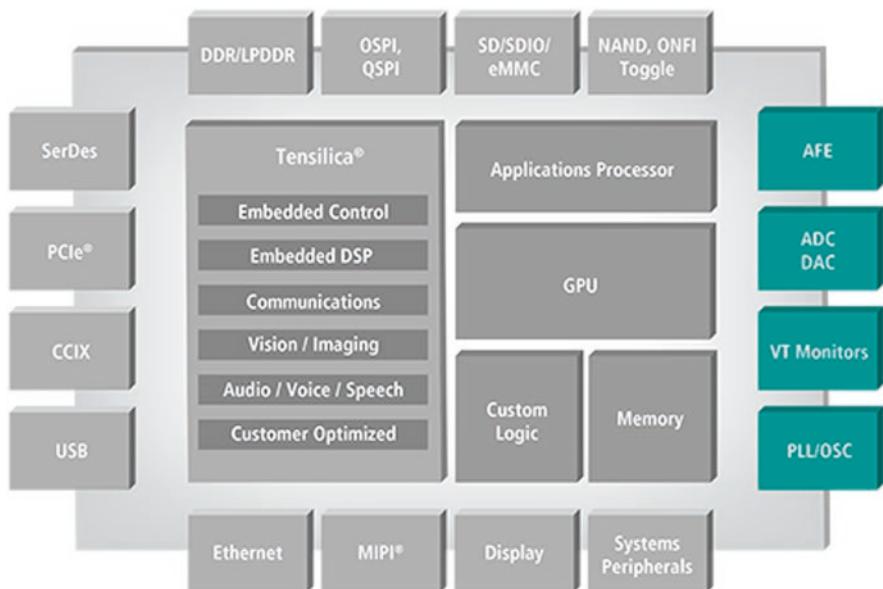


Fig. 5 Types of SoC constituents

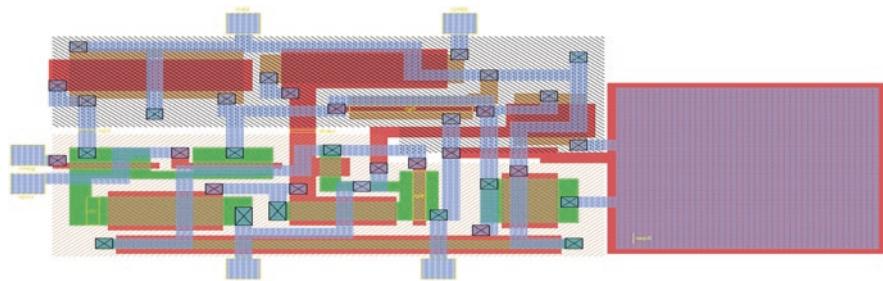


Fig. 6 Layout of OP-AMP. (Credit: Atropos235 at English Wikipedia. [CC BY-SA 2.5 (<https://creativecommons.org/licenses/by-sa/2.5/>])]

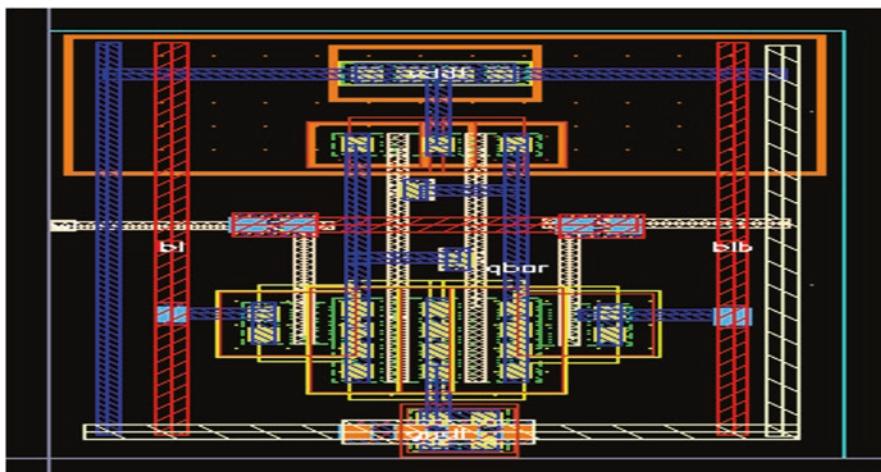
cores is considered more intuitive art than science. Analog power supplies, special noise handling methods are used in the design of these blocks as they process very small amplitude, noise-prone physical signals. Designing the analog blocks is very tedious multiple trial processes as they are high-performance blocks. Analog blocks like data converters interact with analog signal domain on one side and digital processing circuits on the other side. The analog and mixed mode design flow is called custom design flow. The design considerations of these blocks involve sampling rate, bit resolution, target technology nodes and their size. Design technique involves drawing the schematic circuit at transistor levels with interconnections of proper sizes, placing and routing them manually. Design verification for design equivalence, timing, and power is done at every stage as the design phase progress. Simple layout of OP-AMP is shown in Figure 6.

## Digital Cores

Digital cores in SoC are available in many forms. Different design representations of digital cores are netlist, GDS, and synthesizable RTL cores of different complexities. They are also referred as hard, soft cores or macros. Processor subsystems, interface cores, and protocol cores are available in these forms. These are predesigned, pre-verified, and sometimes targeted to a particular CMOS process technology. Digital cores are available as IP cores. These can be integrated with other cores in a SoC design. Digital design cores in multiples of instances are integrated at RTL/gate or in physical design stage and design process is continued till standard ASIC design flow is followed from synthesis till GDS generation.

## Memory Blocks

Memory blocks for SoC integration are available as configurable cores of different sizes in different configurations. These are integrated as on-chip SRAMs, Flash memories and one time programmable (OTP) memories in SoC. Register arrays, RAM, ROM, EPROM, and FLASH memories are different types of memories which are embedded as on chip macros for SoC designs. These memories are optimised for the target technology and are generated using memory compilers with configurations of size and structures and address multiplexers and decoder combinations. Memory compilers generate all design representations for automated cell based design flow. Memory cores also come with test and repair options such as built in self repair (BISR) feature with built in self analysis (BISA). BISA gets enabled automatically when a fault is detected to find redundant memory cells for replacement. A simple layout of memory macro is shown in Figure 7.



**Fig. 7** Layout of memory macro cell

## ***Macros***

Design Macros are predesigned functional cores available in soft or hard cores for SoC integration. Soft macros are synthesizable RTL cores and Hardcores are macros which are integrated at the physical design stage of SoC design. These macros can be instantiated as many times as required for the SoC application. Macros come with full fledged implementation as functional core with defined interfaces and design guidelines for integration.

## **Design Abstractions of SoC Design**

### ***Abstractions***

Different design abstraction levels of SoC design in consideration are the following:

- Design Model as system C or system Verilog.
- RTL level Top hierarchy.
- RTL level block level which could be analog or digital or cell or transistor level.
- Structural level as design Netlist which is HDL file with standard cells/Macrocells and Memory cells and their interconnections.
- Design layout level in GDS format cell dimensions.

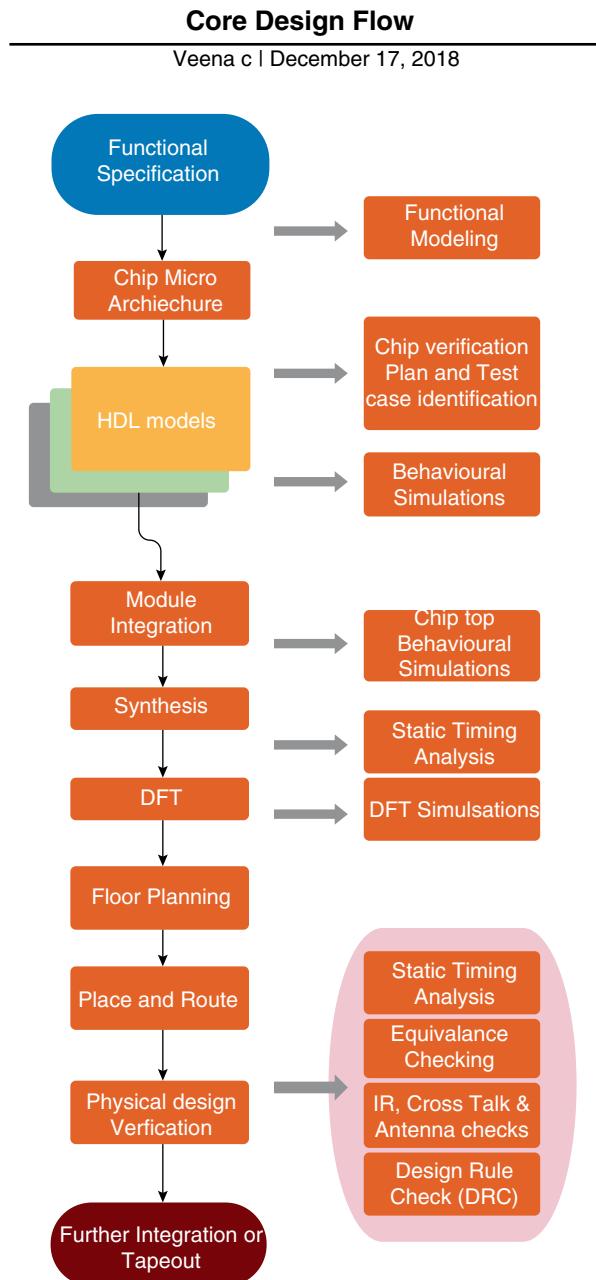
Standard cells, functional macro cells like PLL, Memory macro, analog cores, and digital cores can be integrated into SoC design at any level of abstraction. If the design abstraction is a synthesizable RTL file, it is called softcore and if the design abstraction is GDS level it is called hard macro. If the design block is soft macro, it can be modified but if the design is hard macro, it cannot be modified.

## **Integration of SoC Design Components**

SoC design is traditionally realized by integrating analog, digital, memory macros, and standard cells using automated cell based design technique. According to the sub block design type, the cores are designed using different design flows such as custom layout, mixed system or standard cell based design flow.

## **Trends in SoC Design Methodology**

A typical SoC design flow is shown in Figure 8. SoC design methodology is dealt in detail in [1].



**Fig. 8** SoC Design flow

The demands of future day SoC designs are substantial. The convergence of multiple heterogeneous SoC technologies in sophisticated package requires holistic analysis of the entire solution. The current SoC design methods will not work for these requirements. Future SoC designs require hyper convergent design flow to address the complexities of systems and technologies [3].

## SoC Design Styles

As the design complexity increases, the SoC designers find means and ways to reduce design time. This is achieved in representing design at different stages. Availability of EDA tools today makes a complex SoC design with multiple millions of gates possible. Also, there are many design styles adopted to implement SoC design in full or part. This section discusses VLSI design styles used to design systems on chip.

**Custom design style:** If the volumes of the chips required are large and it is essential to achieve high Performance, Power, and Area goals, custom design method is followed. Most appropriate candidates for this style are processor cores and high-speed analog cores where the design is manually tweaked to take complete advantage of silicon area to operate at highest speed. This is the most complex design style.

**Semi-custom design style:** This design style uses predesigned standard cells and macros yet offers a good degree of freedom to SoC designers to define and realize the functionalities. There are two types of semi-custom design styles. Gate array and standard cell design.

**Gate array design:** In this style, the sea of logic gates or unconnected logic cells are predesigned and laid out. The system designer need not know the design knowledge of these gates as required in full custom design. The semiconductor manufacturer gives the designers a set of gates and instructions to use these gates. Designers will customize these gates and interconnections of these gates to realize the systems. Special type of gate array design style is Field programmable gate arrays (FPGA) where the array of gates is available for designers to customize them through programming an EPROM whose contents control the state of the logic gates and pass transistors. The cells provided consist of I/O buffers and arrays of configurable logic blocks (CLBs) which typically consist of an assembly of programmable multiplexers, logic gates, and one or more bistable devices. Uncommitted logic arrays or sea-of-gates devices consist of groups of transistors which can be interconnected to form other structures by applying a custom layer of metal to interconnect them. Gate arrays are now very popular since they offer low cost and fast turnaround design. Systems designed using FPGAs are very popular today as complex systems can be implemented on these devices. FPGA-based systems are also used in safety critical and mission critical applications as they can be reprogrammed and reused for changed design requirements.

**Standard cell design:** Standard cells-based design style is the most used design technique to realize the systems on chip. PPA goals for the system are easily achievable compared to other design styles. Standard cells are the generic logic cells that are used to design any complex SoC functionality. This design does not have repeated regular structures as in gate arrays. The complete information on standard

cells is given by the fabrication house as a process design kit (PDK). The standard cell library has design description models, timing information, geometric information of cells in the format which are processed by EDA tools and process. EDA tools are very important for standard cell designs. EDA tools are used for design synthesis, Physical design, Simulation, and Timing analysis. Specialized tools are used for each of these processes in SoC design flow.

## Physical Design (PD) of SoC

Physical design of SoC involves converting the design netlist to Physical design layout in VLSI design is also referred as PNR design flow. Physical design methodology of system on Chips (SoC) adopted depends on the type, complexity, and the level of integration (module/block or SoC top level) of design. The analog/mixed signal design blocks are designed using custom design flow. Complex digital cores like processor subsystems are designed following standard EDA tools-based PNR stages. The SoC physical design is carried out using standard PNR flow. This book discusses in detail the physical design flow of the design at a block and a full SoC level.

### *Fundamentals of Physical Design*

To understand the theory behind a SoC physical design, it is recommended to refer to Chap. 8 of the book “A Practical Approach to VLSI System on Chip (SoC) Design” [1]. The Prerequisites to understand physical design like stick diagrams, Re-convergent models, and graph theories are reproduced here for ready reference.

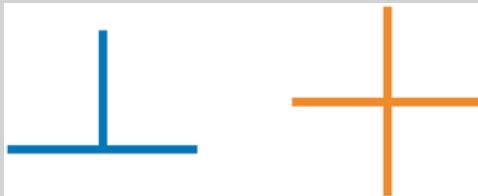
#### Stick Diagrams

Stick diagram captures CMOS VLSI circuit topology and colored device layer information in a simple diagram. Stick diagram is a indicative layout map of symbolic circuit. Notations and a few important rules followed to draw stick diagram of a circuit are shown in Figure 9. The colored lines depict the device layers which are also represented by lines of different styles in monochrome diagram. Rules define the interconnection methods in stick diagram.

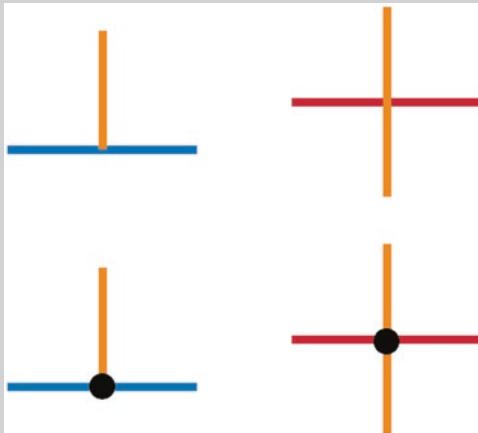


Fig. 9 Color coding of stick diagram

Rule 1. When two or more sticks of the same color touch or cross each other form a contact.



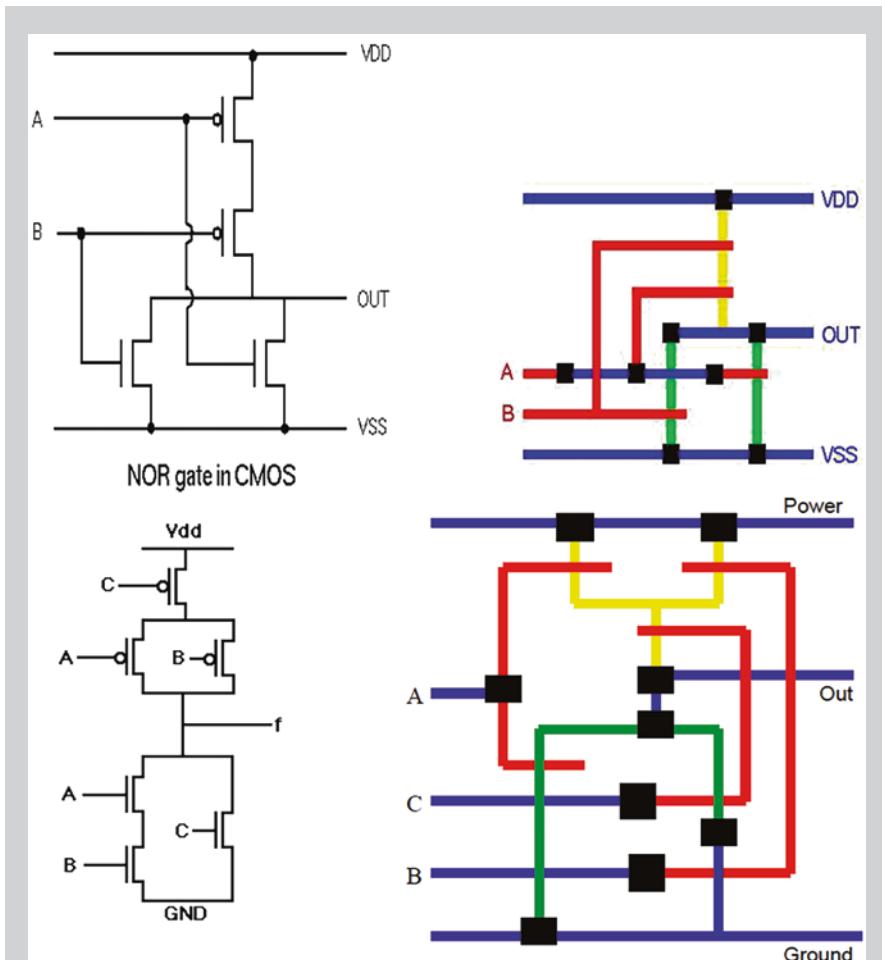
Rule 2. When two or more “sticks” of different types cross or touch each other there is no electrical contact. If contact is to be represented, it must be shown explicitly by a filled small circle.



Rule 3. When two or more “sticks” of different types cross or touch each other there is no electrical contact. If contact is to be represented, it has to be shown explicitly by a filled small circle.

Rule 4. In CMOS a demarcation line is drawn to avoid touching p-diffusion with n-diffusion. All pMOS must lie on one side of the line and all nMOS will have to be on the other side.



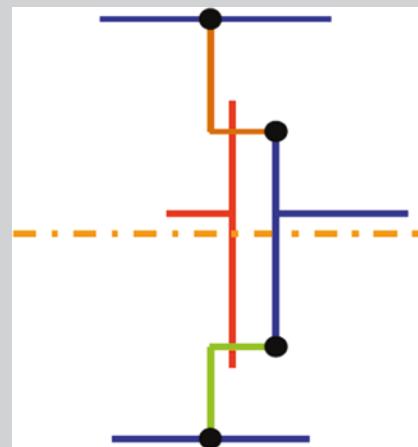
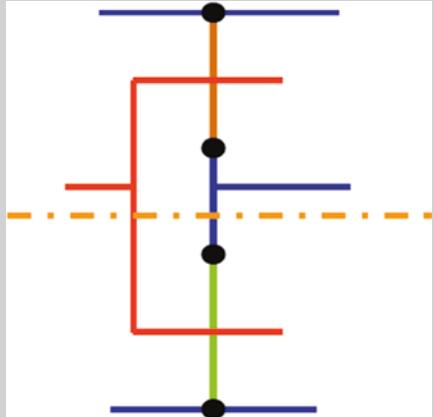


**Fig. 10** Examples of stick diagrams

Few examples of stick diagrams for circuits are shown in Figure 10.

The Stick diagrams do not have the exact coordinates of the device structures and interconnects. The design database after the physical design has complete information of device structures, placement coordinates within the die, vias across the layers, and their interconnections. Design layout database is used to make Masks or Reticles that are used in VLSI fabrication. Mask or Reticle facilitate processing of selective parts of the die to different processes in IC fabrication. Important processes in VLSI fabrication are Doping, Diffusion, Etching, ION implantation, and metallization.

Re-convergent models:



VLSI SoC design flow involves stages where the design is converted to different forms till the time it is taped out to the fabrication houses. In SOC design, specification in document format is converted into RTL behavioral model and through the process called synthesis, it is converted into design netlist and through physical design, it is converted into physical structures. The SOC design flow can be considered as the re-convergent model with multiple transformations. At every stage design can be extracted from the design database for verification and analysis.

The final design database is taped out (design file transferred to the fabrication house for further processing) in GDSII file format. It forms the input for the mask making process of the fabrication. This GDSII file contains information regarding the different structures that are used for Mask making. Masks are used in the fabrication process. Processes such as doping, ion implantation, chemical vapor deposition (CVD), physical vapor deposition (PVD) in fabrication is selectively exposed through the mask to different parts of the die, on the silicon wafer. Design flow converts the specification in the document format to a design layout represented in GDSII format. The SOC design starts with capturing the system requirements as specifications in a document, which is modeled as Register transfer language (RTL) design using hardware description languages (HDL) like Verilog, System verilog/VHDL, then synthesized to a gate level netlist which is then converted to physical layout of structures written out in GDSII format.

## File Formats

During SoC design flow, different design information is stored in different file formats. Table 1 lists various design information and their file formats with their relevance in design flow.

## Process Design Kit (PDK)

The fabrication houses develop the process design kits consisting of many standard logic cells, macros, power, and signal pads which are used for designing system on chip (SoC). The PDK is a physical design kit which is a set of design files representing different views and characterization data of standard universal logic cells, generic pad cells and memory cells. A PDK contains files with timing information, size information, and physical layer information as per the fabrication process. NAND, NOR INV, Flip-flop are a few of the standard cells in the PDK. There are some process specific special cells which are required for reliable fabrication of the VLSI circuits in PDK. Corner cells, Tie cells, Tap cells are examples of such cells. Technology Library file in liberty format (Lib file) contains timing-related information like pin capacitance, port capacitance, net delays, drive strengths. This information in PDK contains information like input capacitance, output capacitance for cell structures from which the cell propagation delay is computed for circuit design analysis. There can be many cells of the same type with different electrical characteristics. For example, in PDK libraries there will be NAND gates of different drive strengths, NOR gates of different number of inputs and drive strengths etc. PDK contains files with the cell structure details of a target CMOS fabrication process. The file format containing these information is called layout extraction file (LEF). The EDA tools use these files during

**Table 1** SoC Design file formats and their relevance in design flow

S No.	Design stage	Format	Description
1	Requirement capture, Marketing requirement document, Architecture document, or High-Level Design Document	Document in docs, doc, XLS	Chip architecture is documented from market requirement, standard, feature list.
2	Design Modeling using hardware description language	Verilog/VHDL files; .v, .vhdl formats	The SOC functional behavior is modeled using HDL.
3	Synthesis	Gate level file in Verilog/VHDL file containing logic gates and interconnections; .vg, formats. lib files	The SOC design is converted to gate level netlist by the process called synthesis using synthesis tool. Synthesis tool can also write out liberty timing files in the form of .lib. Liberty timing file is the ASCII representation of timing and power parameters associated with the cell at various conditions. It contains timing models and data to compute input-output path delays, Timing requirements (for timing checks), and interconnect delays.
4	Static timing analysis and Signal Integrity checks	SPEF file	Standard parasitic exchange format (SPEF) file is the IEEE standard format for representing parasitic data in ASCII format on interconnect in the design. This is used by the static timing analysis tool to compute path delays and for interconnect data for signal integrity checks.
5	Static timing analysis/ dynamic timing analysis	SDF	Standard delay format (SDF) or Synopsis delay format is the representation of timing delays.
4	Floor plan and Placement; Global Routing; Clock Tree Synthesis	DEF, LEF files	Design Exchange Format files written as .def file by place and route tool contains die size, logical connectivity, and physical location in the die. Hence, it contains floor planning information of standard cells, modules, placement and routing blockages, placement constraints and power boundaries. Layer Exchange Format (LEF) provides technical information, such as metal layer, via layer information and via geometry rules. The LEF file contains all the physical information for the design. DEF file is used in conjunction with LEF file to describe the physical layout of the VLSI design.

(continued)

**Table 1** (continued)

S No.	Design stage	Format	Description
5	Power routing	Layout file. LEF file, DEF File, Lib file	
6	Detail routing	Layout file. LEF file, DEF File, Lib file	
7	Tape out	Layout file in GDSII format	Industry standard database file format for data exchange for layout artwork. It is a binary file format representing planar geometric shapes, text labels, and other information about the layout in hierarchical form. GDSII files contain all the information related to SOC design. Once the design meets all the constraints for Timing, SI, Power Analysis and DRC and LVS it means that the design is ready for Tape out. This GDSII file is used by Fabrication house for mask/reticle making.

design flow of SoC and computes SoC design-specific timing and placement information. Fabrication foundries provide the PDK to design houses. There are open source fabricatable PDKs which are accessible to the designers.

### Physical Design Flow for Analog and Mixed Signal Designs

Physical design of analog and mixed signal blocks use a custom design flow where the schematic circuit is drawn in schematic editor using each of the design elements is designed individually. The transistor sizes such as length, and width are computed for a specific electrical specification using fundamental design equations. From the schematic editor, the layout is drawn manually using corresponding layers corresponding to process layers. The design verification of custom design include circuit simulation using test bench, schematic vs. layout, design rule check (DRC), Electrical rule check (ERC). This requires a basic understanding of device characteristics, second order effects, circuit behavior, design methods.

### Physical Design Tools

Physical design phase of SoC top level of the hierarchy is predominantly EDA tool dependent. The EDA tools for physical design are often referred as Physical design (PD) tools. The PD processes are compute intensive requiring advanced high-performance (Workstations with multi-core computer with high MIPS, high speed, display monitor with Good graphic resolution and large memory) machines.

The PD tools include modules for schematic editor, floor planner, placer, and router modules with good graphics user interface. In addition, the PD verification engines and analysers are integrated with these modules for design analysis on the go and optimization. There are design extraction features to write out design netlist at every stage, parasitic electrical parameters of the component or interconnect or of design subset or in the top level design. Tools also check any violation of fabrication rule set. The input design files used for physical design are SoC design netlist, design constraint, and PDK. Design process is carried out on the design files for design planning on silicon die (floor plan), placement of design blocks, and interconnections of them as per the design specification and set of PPA goals.

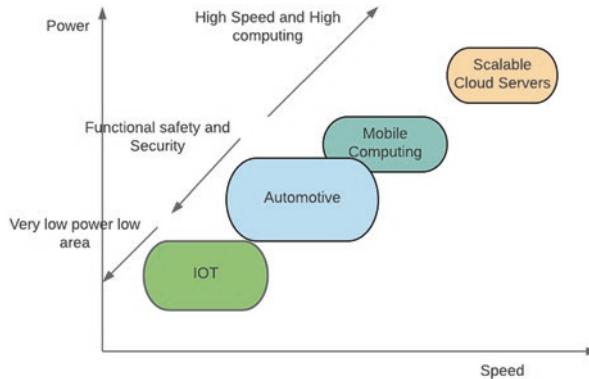
### **Future of EDA Tools**

System on chip design using EDA tools is a very complex process. This requires large computing machines for tools to run on them. The quality of system design depends on the EDA tools and the design flow used to meet the design goals. The PD tools use advanced algorithms, that are more effective and efficient to deal with extremely large search design space with low latency. Many times, these algorithms use iterations of similar runs to choose at optimal PPA for the design. Although the use of machine learning (ML) traces back its history to the 1990s, recent development in ML and increased complexity of system design have aroused interest in adapting them into them again. ML methods show great potential to generate high-quality systems for many NP-complete (NPC) problems, that are common in the EDA field, while traditional methods lead to huge time and resource consumption to solve these problems. Advanced PD tools learn from old design runs and chosen options to guide the designer. It is ongoing efforts from the tool vendors to make ML as an integral part of EDA offerings for system designs faster and thus reducing the time to market. Major EDA vendors have already provided ML- and AI-enabled EDA platforms for either AI SOC design or complex SOC designs with massive processing and deep learning features. Another trend seen in EDA tools is unified design platform to support 3D IC to further the integration of heterogeneous chiplets to develop feature rich systems of the future.

### ***Future Trends of Systems***

Aligning to the trends of the electronic industry in areas of AI, IoT, automotive, and 5G technologies, functional blocks on semiconductor chips are to be developed spanning from devices to systems and systems to devices. New professional body called the international roadmap of devices and systems (IRDS) is defining the roadmap of future systems [2]. IRDS predicts the following:

- Reduction of feature size will reach final limits around 7–8 nm towards the end of this decade.



**Fig. 11** Power speed performance of different applications

- With AI features, system capability over-performs human errors as seen by vision processing systems.
- New devices continue to be invented beyond CMOS.
- Hundreds of NAND memory layers are stacked to realize dense memories.
- Quantum computing and quantum communication promise unprecedented increase in functionality.
- Several technologies to reduce the storage cost are becoming practical.
- Stacked transistors are realizable.
- Future trends will be integration of CMOS and NON-CMOS devices on chip. These are being done in silicon in package (SIP) technology as of today. Another trend seen in complex SoC with many cores operating with different clock frequency is simplified by making each IP core to operate in its isolated timing domain. This ensures on-chip interconnect to be treated as an additional IP core.

Considering the above trends, it is very evident that future systems will demand application-specific power performance and area (PPA) goals as shown in Figure 11.

Applications demand system integration of many functional cores and integration of cores at deep submicron nodes pose different challenges. The resistance of metal, vias is increased exponentially making PPA convergence very difficult. Power problems at advanced nodes need advanced techniques for reliability and variability modeling at ultra-low voltages. Heterogeneous integration of functional cores needs techniques which are low cost and high-performance solutions like 3D packaging.

Fully automatic and smart EDA tools capable of advanced automatic techniques of design flows with via pillars are required to solve high resistance problems beyond 7 nm process technologies. The systems in future demand time space explorations beyond human possible iterations and that demands highly scalable, on-demand high computing platforms to self-learn, adapt, and guide designers to achieve set PPA goals. EDA tools need ML and AI capabilities to solve deep submicron systems of high physical design complexity.

## References

1. Veena S. Chakravarthi, A Practical Approach to VLSI System on Chip (SoC) Design, Springer-Nature publication 2019
2. International roadmap of devices and systems 2020
3. <https://www.forbes.com/sites/marcochiappetta/2021/05/25/chips-designed-by-ai-are-the-future-of-semiconductor-evolution-beyond-moores-law/?sh=25f5a296430f>

# SoC Physical Design Flow and Algorithms



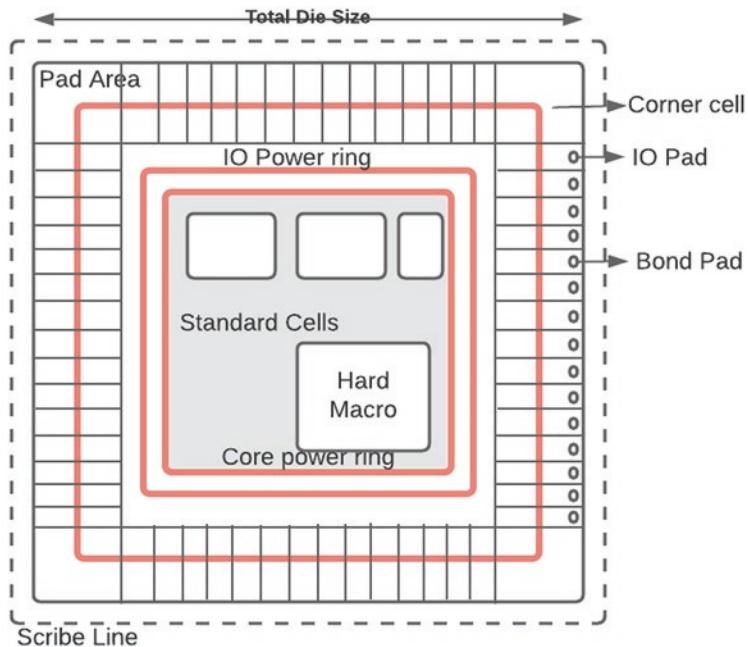
## System on Chip (SoC) Physical Design

Physical design of SoC begins after the successful completion of the logic synthesis. The output of logic synthesis is the structural representation of SoC design in netlist form. SoC design netlist is converted into chip design layout by Physical design process. SoC design netlists are of two different types: flat and hierarchical. A flat SoC netlist consists of all the design objects in the same level of hierarchy. Hierarchical netlists consist of design blocks instantiated at different levels building to a bigger design blocks, which form the SoC top level. Number of design hierarchy levels for a complex SoC is typically five to six. Physical design of SoC is carried out at block level or at flat level. Design blocks can be inserted into the physical design process depending on the type of design. Physical design tool reads in the design blocks depending on the file formats.

## SoC Core Design and Input-Output (IO) Design

SoC physical design is divided into core design and Input-output (IO) design. The core design holds all the logic components defining the core functionality of the system. IO design is the area of the die where the IO pads and Power pads are placed in the die. Typical SoC die is shown in Figure 1. As in the figure, SoC design will be confined to the scribe line on the die. Scribe line also serves as a marker to dice the die on the silicon wafer after fabrication.

SoC Die has an aspect ratio which is the ratio of die width to die length. SoC dies are square or rectangular which are easy to dice automatically by step and repeat machine tools on wafer post fabrication and very rarely of custom shape. The die boundary is defined by a *scribe line*. All the functional elements of the SoC design



**Fig. 1** SoC Die Physical design terminology

are constrained within the scribe line. During the physical design, the scribe line acts as a design boundary within which the functional blocks are placed.

SoC design netlist in the form HDL file format or .db file, design constraint file with standard design constraint file (sdc), and the technology library file in liberty format (lib) are the inputs to physical design.

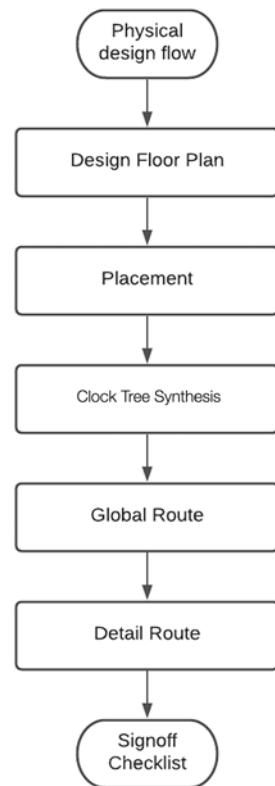
## Physical Design Flow of SoC

The physical design flow of VLSI SoC design is shown in Figure 2.

### *Floor Planning of SoC Design*

SoC core area and SoC IO area are two distinctions to identify in any SoC design. Functional blocks are placed in the SoC core area and the input-output (IO) pads are placed in the IO area of the SOC design as shown in Figure 1. There are three types of IO pads depending on how they are connected. They are input pad, output pad and bidirectional pad. Special pads such as Bonding pads are pads which get

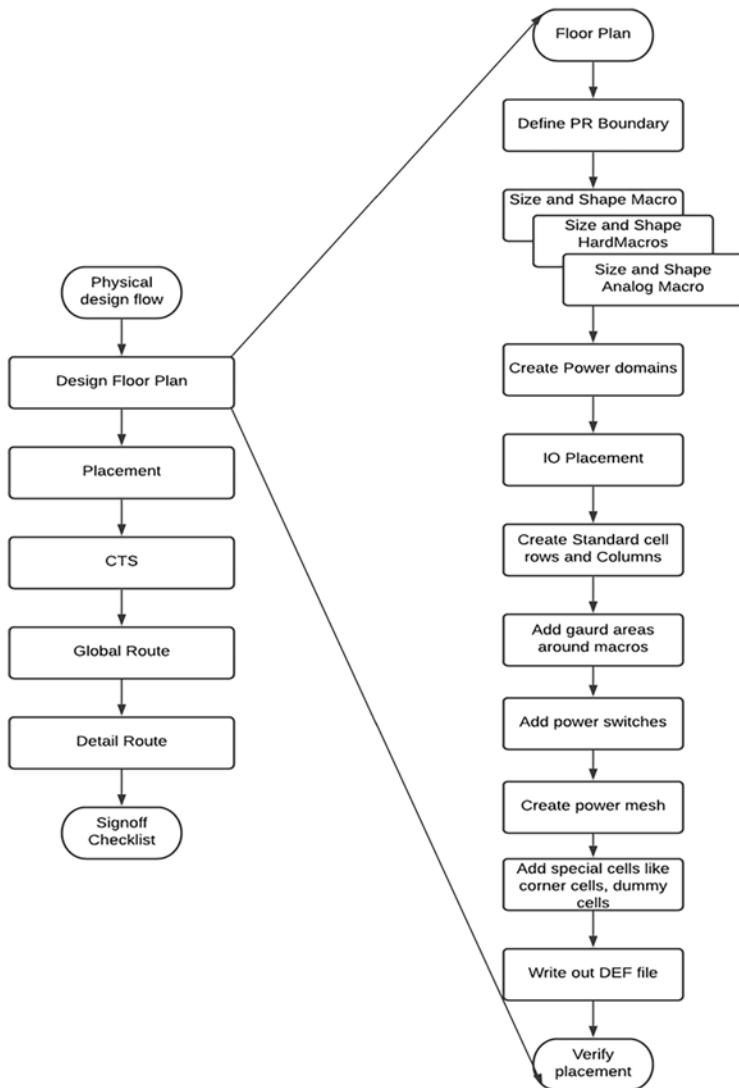
**Fig. 2** Physical design flow



permanently bonded to the ground or power in the package, IO pads are connected to the Pins or ports on the package. Tap cells, corner cells are dummy cells on the die offering mechanical stability to the die. Power is supplied to all the components in the core area as core power through the core power grid drawn from the core power ring surrounding the boundary. Similarly, the IO pads get their power from the IO ring around the boundary. SoCs of today have multiple power rings corresponding to many power supplies they require. The SoC core has SoC design netlist that consists of the following components:

- Standard Cells
- Memory Macros
- Digital macros
- Analog cores

SoC design is planned for physical placement of the design blocks in the process called *floor plan*. *Floor plan in SoC design flow is shown in Figure 3*. Analog and digital functional blocks, macros, and standard cells use different power supply voltages and can be in different design representations. Some design blocks are hard and may even be protected from viewing their internal logic cells. Deciding on the *Die size* is the first design decision taken for SoC floor planning. It is taken based on



**Fig. 3** Floor planning

optimum fitment of all the design elements considering ease of interconnecting nets also called *interconnects*. Interconnects connect logic cells in the design. The length of interconnects connecting the two design elements must be estimated to determine the optimal fitment of logic in the die size.

## Cell Characterization and Wire Load Model

Wire load model is the model consisting of delay and power characteristics of a library cell. Delay characteristics are mentioned as resistance and capacitance values in each of the circuit paths of the standard cell. This information is used to calculate the cell delay for different fanout conditions. Specific resistance and specific capacitance values of the interconnect are used for estimating interconnect delay after routing (process of interconnecting the signals) the design. These values are stored as a look-up table in a file of the cell library. The look-up table is developed by characterizing each of the standard cells for a particular CMOS process at different operational conditions. Model is written out as a file in liberty format. Liberty format is an industry standard format used to describe the library cells of a particular technology. Liberty is an ASCII format, usually represented in a text file with the extension “.lib”. Figure 4 shows sample timing data of the model for delay computation. The values are the percentages of power supply Vdd.

Every standard cell is characterized by simulating its behavior under different conditions it may encounter in the design. For example, an inverter is characterized for changing input transitions and output load capacitances as they are the parameters that can change its electrical (delay) behavior when used in a circuit design. The delays are of two types: cell delay and interconnect delays. The cell characterization data is written in a liberty file in the form of a look-up table representing delay values at different transition times and load values. It is a part of a technology library developed by the fabrication house. *Zero wire load model* is the liberty (lib) file that has only cell delay information. It does not contain the delay due to interconnections. A zero wire load model is used for preliminary synthesis as the cells in the design netlist that are logically connected by cell and net names. They are not physically interconnected in the design netlist. Similarly, the standard cell library file contains characterization data for estimating power dissipation in cells and in interconnects. Timing and Power analysis tools use this file for analyzing the delay and power of the design and to identify design violations if it exceeds the expected thresholds.

**Fig. 4** Liberty file sample  
for timing of the cell

```
# threshold point of input falling edge


# threshold point of input rising edge


#threshold point of output falling edge


#threshold point of output rising edge

```

However, to determine the boundary for the physical design of the SoC, a rough estimate of the interconnect delays is used. Interconnect delay of the cells is estimated by extrapolating or interpolating average delay values in the library file. These baseline values in the wire load model liberty file for interconnects are derived by the characterization data collected from historical designs fabricated using the process.

## Die Size Decision

The boundary for the SoC design is determined by estimating the silicon size required to optimally fit all the functional elements of the design and their interconnections. Every SoC design has a performance, power, and Area (PPA) specification in addition to intended functional requirement. Optimal fitment must meet the timing performance, power, and Area (PPA) specification for a SoC. Design netlist, liberty file, and design constraint are used to arrive at the best fit of a floor plan. Floor plan identifies physical positions for placing analog, digital functional blocks, macros, embedded memories, IOs and standard cells listed in design netlist, within the die boundary. All the design elements except the IOs form the functional core which are planned to be placed following the design and geometric rules defined by the technology process to meet the specified design goals. Optimizing the placement of functional blocks within the identified PR boundary of the die is an iterative process.

## Power Plan and Power Strategy

Power planning is another activity during floor planning to distribute the power to all the design elements. Power mesh is decided as per the requirement of the functional blocks, macros, and standard cells for an even distribution of power to all cells in the design. It is also called *power network synthesis* or pre-routing as it is done before the clock and signal routing.

## Floor Plan Verification

- The floor plan of the design is verified for intended functions, targeted PPA and compliance to design rules. This is done by functional simulations, power, delay and IR drop analysis and checking for the violations of the specified design rules. It is also verified for the correctness of floor plan rules. Any violation resulted due to wrong planning resulting in wrong or overlapping placements, crisscrossing of interconnects and modules are corrected (fixed). Power mesh is verified such that every cell is adequately powered. *IR* analysis is used to check this on the floor plan. Some of the design guidelines for a good floor plan are the following:
- Macros blocks must be placed such that there are no overlaps and connections are done with optimal interconnections. Using *fly line analysis* macros that are connected are placed closer.

- Macros are placed close to IO pins. The spacing between the macro and IOs, connected Macros are maintained so that interconnections are done easily.
- Block-level macros are placed near the module or block boundaries.
- Considerable spacing called guard space or Halo space around all sides of the macros is provided so that no standard cells are placed there which can cause timing violations.
- Proper blockages are provided around all sides of the macroblocks to avoid cell placement in that area which may affect the macro functions violations.

The output of the Floor plan of the SoC design is the floor plan of the design with power mesh that meets design goals.

### ***Placement of SoC Design***

The design core, macros, IOs are placed in identified positions in the placement stage. Standard cells are placed in the rows identified in the floor plan. In addition to these, tap cells, antenna cells, and spare cells are inserted in the placement stage. *High fanout nets* are identified in the design and synthesized by placing the buffers in their signal interconnect paths to carry signals reliably in the design. High fanout nets (HFNs) are design nets connected to many nodes. *Scan chain reordering* is carried out in the design as the floor plan just plans the placement without considering the order of scan chains. Following are the sequence of operation in the placement stage:

#### **Pre Placement Stage**

In this stage, special cells such as antenna diodes and buffers are added near the IO ports of the functional blocks and the macros. Tap cells are inserted to avoid any latch-up issues in the design. Design macros and standard cells and IO pads are placed without any placement violations of design rules. Spare cells are inserted into the design.

#### **Course Placement**

During this stage, all the design elements and standard cells are placed considering macros and blockages. This is done to analyze any placement violation and possible congestions during signal routing.

#### **Placement Legalization**

Placement violations are fixed in placement legalization stage. This is automated process using PD tool. The tool moves all the design elements and cells such that there are no design rule violations and overlaps. This process may result in new timing violations on analyzing which are to be fixed. It is an iterative process.

## **High Fanout Net (HFN) Synthesis**

High fanout nets are nets that have a large fanout connection. HFN synthesis inserts buffers in the paths of HFNs. Clock nets are not considered as HFN at this stage. They are considered an ideal net.

## **Placement Optimization**

Placement optimization is carried out to reclaim any unused silicon space and avoid any probable routing congestions. Signal routing is done using specific paths in the design called routing channels. Routing congestion is seen in the design if the number of routing channels available is less than the number of channels required for signal routing. This is avoided by honoring placement guidelines for placing the macros, blocks, IOs, and standard cells. Blockages both Soft and hard, halo spaces around all sides of the macros are the ways to avoid signal routing congestion. Cell padding provides required cell-to-cell spacing which also avoids congestion.

## **Scan Chain Reordering**

Scan testing chains of scan flip flops in design netlist will not be optimum as they are placed using auto placement in PD placement. The scan chains are also not in one order. This is reordered in this stage of design placement. This provides optimal routing of scan chains avoiding congestion in the design.

## **Clock Tree Synthesis**

The clock is a very important control signal in any SoC design. All the synchronous logic elements require a clock signal input without any non idealities or minimum or balanced skew and jitter, to be specific. Clock tree synthesis (CTS) adds clock buffers or inverters in the special tree structures such as H-tree or mesh for the clock signal distribution. The clock signal for the logic cells and macros are connected from the nearest point on the tree structure. This ensures the uniform distribution of clock signals to all sequential cells in the SoC design. In any design, a clock network consumes around half the dynamic power in SoC design. This is minimized by a process called clock gating. The control gates are added to generate enable-disable the clock signal to the logic blocks whenever it is not operational. The control logic consists of clock gates that are added during CTS. This reduces power consumption in the clock circuitry. The clock is also one of the high fanout nets and the CTS process reduces insertion delay and minimum skew on the clock signal path. The CTS process uses non default clock routing rules (NDR) for clock signal paths. Clock signal routes are of double or triple the default widths of metal in the design. SoC design after CTS is verified for legality of cell placements, setup, and hold times violations, and for CTS/ NDR rule violations.

## ***Global and Detailed Routing***

After the CTS stage, the design is ready for signal routing. Routing is a process of interconnecting design signals using interconnect segments of metal layers. It is done in two steps: *Global and detailed*. Interconnect routing is done for standard cells, macros, pads of block boundaries, and IO Pins of chip boundaries. Routing is the process of connecting all the nets in the design in the routing space following routing and via rules defined for the process technology. Vias are vertical paths or through holes which will be filled with metal segments used to connect two metal layers stacked one above the other separated by a dielectric in a design layer stack. Signal routing is done as two step process as follows:

Global routing divides the routing regions into tiles or rectangular sectors and the routing channels are created for interconnect routing. These tiles are called global routing cells or gCells. gCells connected to metal interconnect paths are identified to connect gCells as per the SoC design requirement, following the routing rules of the technology process. Actual metal routing is not done in this step but the optimal path for interconnecting is identified for physical connection. Routing capacity depends on routing channels, blockages defined in the design. Global routing is also called coarse routing. After iterations of estimation optimal routing paths, tracks are assigned to connect gCells. Tracks are vertical and horizontal depending on the identified paths.

Detailed Routing is the process of physically filling the metal layer in the identified tracks of the gCells. Filler cells and metal fills are added to comply with the design rules of the fabrication process. Routing is verified for design rules, timing analysis, and LVS checks.

## **Physical Design Verification**

The design layout is ready once the signal routing is verified completely. But the function of the SoC design has to be confirmed with a few additional checks as intended. These are called *Design sign-offs*. They are:

### ***Design Rule Check (DRC)***

Any violation in complying with to design rule check must be resolved as this affects the processing of the circuit resulting in failure of the chips. These rules are to be satisfied by the layout as these are extracted from the resolution of the processing equipment. Some of the rules are the thickness of the metal runs, spacing between two interconnects, the pitch of the IO pads, etc. DRC is the physical rule checks for metal widths, spacings, pitch, etc., defined for a particular fabrication process.

### ***Layout vs. Schematic (LVS)***

The tool writes out a netlist from the layout which is compared with the netlist generated from the schematic of the same stage. The two netlists are compared for equivalence. The two must be equivalent to proceed to tape out. Typical LVS errors are shorts, opens, or parameter mismatches.

### ***Logical Equivalence Check (LEC)***

The netlist generated from layout is compared for equivalence with the golden netlist from synthesis as reference. Golden reference netlist is a pre-layout netlist. The two are compared for functionality equivalence. The difference between the post layout netlist will be it contains the clock tree network, power networks, and additional buffers inserted as timing fixes.

In addition to the above, some signal integrity checks are carried out to avoid cross-talk, electromigration, and antenna effects.

## **EDA Tools for Physical Design**

The physical design of SoC is almost fully automated and is very much tool-dependent flow. Physical design tools include the following modules:

- Floor planner
- Placer
- CTS tool
- Router
- Parametric extractor
- Netlist Extractor
- Layout editor
- DRC checker
- LVS equivalence checker

In addition, PD uses verification, extraction and analyser modules like timing analyzers, power engines to verify and optimize the system design layout.

# Floor Plan and Placement of SoC Design

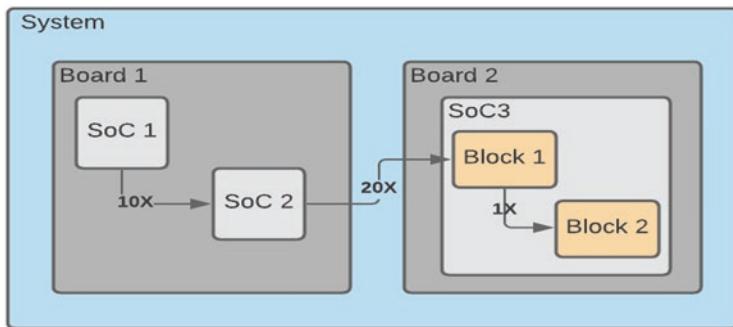


## Floor Planning of SoC Design

Floor planning is the first and crucial step in SoC physical design. Subsequent steps of placement, Clock Tree Synthesis (CTS), routing are dependent on a quality of floor plan. A good floor plan reduces the time to market for SoC designs by avoiding many iterations. Floor planning is planning the position and orientation of each of the design objects for easy interconnection with optimum wires. Floor planning targets no compromise on required Performance, Power and Area (PPA) set for SoC.

## Design Partitioning

The SoC design database will have different design elements such as analog cores, digital cores, and memory arrays in the form of macros or independent entities. Some functional blocks in micro architecture of system are designed as blocks with standard functions and interfaces. SoC Design partitioning is a process of dividing it into smaller design objects for the physical design tools to process. The designer can independently understand, analyse the processed design objects and fix any violations or deviations from the design requirements. It is done hierarchically top-down by dividing the design into subblocks repeatedly until the smallest design element is reached. Different design levels in SoC design are transistor level, logic gate level, sub-block level, block level, sub system level and SoC level (often referred as SoC top level). Design partitioning re organises the level of SoC design, by partitioning at transistor level, gate level, subblock level, block level, and SoC level for a system. It also considers board level and multiple board level usage of the SoC in a final application. Input signals, data traverse through these design elements as they get processed to generate desired outputs. Such paths traversed by signals in SoC



**Fig. 1** Critical path in system

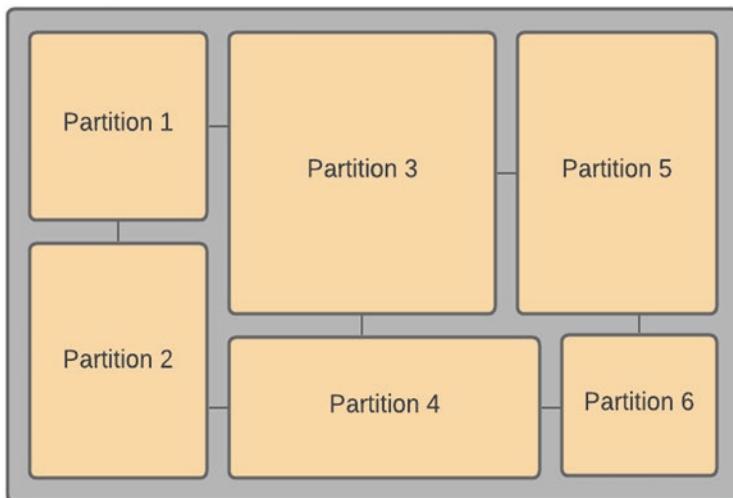
design are called delay path. Delay path is critical delay path if the signal arrives just in time for correct processing by the design element, failing which the processing results into undesired output resulting into functional failure. The critical delay paths are affected by the system partitioning. Path delay of signal path crossing chip boundary is ten times the delay within the SoC. When the signal crosses the board boundary to another board in the system, it increases twenty times. Path delays in SoC, Board and across boards in a system is shown in Figure 1. The goal of design partitioning of SoC is to reduce the path delays as much as possible inside the chip.

Physical design is a tool based automatic process in which the designer feeds input for the tool to process as per the designer's requirements. Each of the design partition is a design netlist which can be independently planned to be placed by the designer using a physical design tool. Hard macros are design objects that is retained as it is. Some of the main considerations in design netlist partitioning for floor planning are:

- Design netlist is decomposed into smaller netlists without affecting the overall design functionality.
- Each block gets connected to other blocks with minimum signals. This ensures that the interacting blocks are planned to be placed nearby.
- Maintain minimum delay of circuit paths when timing paths run across multiple blocks meeting the timing requirements like input-output delays at the interfaces of interacting blocks.
- Maintain minimum number of signals interfacing blocks and limit them to less than the set limit of maximum number of signals.
- Area of subblocks is almost equal and each of the subblocks is of estimated sizes bound by the boundary around them. Large number of subblocks result in easy physical design but increases the cost of fabrication. It is necessary to arrive at a trade-off between physical design complexity and fabrication cost of the SoC design.

Design netlist is partitioned using PD tool for most of the non critical design blocks and manually for time critical part of the SoC design. The PD tool uses

iterative probabilistic algorithms for dividing the netlist into smaller blocks. PPA is assessed for a Design with a baseline cost goal that is improved upon incrementally again and again till the design database is manageable for design analysis by the designer. Partitioning algorithm of the PD tool must be fast enough so that it is a small fraction of the total physical design time. This enables multiple iterations of partitioning the design to reach satisfactory design partition. Design partition must follow standard design rules for its manufacturability. The standard design rules are defined by the fabrication houses where the design is targeted to be manufactured. Design rules are important part of SoC design process to ensure it is fabricatable. Design rules are part of standard cell library often given by foundry. Few terms used in the design partitions are *terminal pitch* and *terminal count*. *Terminal pitch* is the minimum spacing between two successive terminals of the design block. Terminals are design input output pins on the interface. Terminal pitch depends on the target technology. Each design object or block has defined size. The design size defined by area is the cumulative value of areas of standard logic cells and their interconnections used in the design. The *terminal count* is the ratio of perimeter of the partitioned block to its terminal pitch. The number of *signal nets* which connect one partition to other partitions should not be more than terminal count of the design partition. The PD tool places all the critical components in the same level of hierarchy and in the same block. If that is not possible, due to lack of flexibility, they are at least placed in close proximity. The inputs used for design partitioning include design netlist, area constraint and terminal constraint. The cost function for the partition algorithm is the number of the interface signals that cross partition boundaries and number of times the signals cross partition boundaries. Figure 2 shows an example of a design partition.



**Fig. 2** Design Partition example

The SoC physical design can be *pad limited* or *core limited*. If number of input-output pads are too many, it is pad limited. On the other hand, if number of pads are too small and there is space between pads, it is core limited. Floor planner module in the PD tool decomposes the SoC netlist into partitions which can be managed by the placer module of the Physical design tools. The designer requirement is provided by proper design constraints. This results in minimizing the interface signals across the partitions of the design netlist. Design planner module uses 70% of the design layout area for all the modules with 30% space reserved for interconnect routes which connect different blocks, clock routes for timing elements and test features. The 70% area used for planning design objects is called utilization which is the input to the auto planner are design netlist, utilization and design constraint specifying preferred placement positions of any design objects.

Die size utilization is the ratio of the area which can be used to plan the design structures to estimated die size expressed in percentage. Aspect ratio is the ratio of die height to die width of the design layout. A number of instances are a number of partitions in SoC design netlist. The partition tool will restructure the design netlist to be able to structurally partition the netlist without breaking the functionality. After the design partition, the tool writes out the partitioned design netlist. Partitioned design netlist is checked for equivalence to the input design netlist (synthesised netlist) to confirm that the functionality is retained.

## **SoC Floor Plan**

Floor planning is a process of arranging design components (macros, standard cells, complex cells, I/O cells, etc.). Design components are planned to be placed close to each other for easy interconnection as per the functionality in design netlist and PPA goals specified in design constraint. The main objective of floor planning is the optimum size and shape of die layout that meets functional and performance specifications. At this stage, design components are not physically placed and interconnected. The smaller the SoC die size, the lower will be the chip cost. Once the design partition is done, floor planning involves placing design partitions nonoverlapping with each other and interconnected with minimum length interconnect wires within the die boundary. Die boundary is created using approximate area estimates of design blocks, macros, and Input-Output pins and additional 45% of die area. The additional 15% of the diesize is reserved for Scan flops replacement, CTS, buffer insertions used for fixing timing violations of design paths, Electronic Change Order (ECO)s. Remaining 30% more die size is reserved for the interconnect routing across the design components as per the design netlist and power routing of the instances. The following case study demonstrates the estimation of diesize. This example can be used as a template to estimate approximate die size of any partitioned SoC design.

Normally, the metal layers used for interconnection of design blocks in the automated cell based designs are classified as vertical layers and horizontal layers.

Following parameters are considered from the technology cell library data of targeted fabrication process provided by the foundry where the design is planned to be fabricated:

Gate Density per sq. mm =  $D$

Number of Horizontal Layers =  $H$

Number of Vertical Layers =  $V$

SoC design Inputs:

Gate count of the SoC design (excluding memories, macros and sub-blocks) =  $G$

# This is considered from the synthesis area report

IO area, in sq. mm =  $IO$

Memory + Macros + Subblock area, in sq. mm =  $M$

Target Utilization, in percentage =  $U\%$

Additional gate count for Scan flop replacement, CTS, buffer additions, sizing for fixing timing violation etc., in percentage =  $T\%$

Additional gate count for ECOs, in percentage =  $E\%$

Die area calculation:

$$\text{Die Area in sq.mm} = \left\{ \left[ (G + T + E) / D \right] + IO + M \right\} / U$$

$$DieArea = \left\{ \left[ (G + T + E) / D \right] + IO + M \right\} / U$$

Die aspect ratio, width, height calculation:

Aspect Ratio

$$AR = height / width$$

$$= Number of horizontal resources / Number of vertical resources$$

$$AR = H / V$$

Height

$$AR = W / H$$

$$W = H * AR \quad (1)$$

$$Area = W * H$$

$$= H * H * AR \left( Expressing W in terms of H from (1) \right)$$

$$H_2 = Area / AR$$

$$H = \text{SQRT}(\text{Die Area} / \text{AR})$$

Width

$$W = H * \text{AR}$$

The die boundary and shape are fixed by designer's preference of Die shape and estimating die height and width. This automatically determines the number of resources available in horizontal and vertical channels for the die area. The IO cells, Macros, standard cells are placed as per the partitions to minimize the interconnections wire lengths. While floor planning, the spacing guidelines for macros and memories if any are to be considered. These guidelines, rules and restrictions are provided by the macro or memory vendors. The design structures are not placed during floor plan, but placeholders are provided for each of them. Figure 3 shows an example of a floor plan of a design.

## ***SoC Power Plan***

Floor planning the SoC design involves power planning. Power supply and corresponding ground wires are to be connected to all design components for proper operation. Power VDD and Ground GND supply to the SoC design are fed from the external source through one or more primary input pads of the design. These Pins are interconnected to the supply terminals of all the design elements. The power supply provides necessary bias voltage to all the design elements. The power ground interconnects are to be uniformly distributed across the die layout. This is done by connect lines to planning the power grid. Distributed power grid has taps on them which are connected to the power points of the design elements. The power grid runs all around in the reserved channels on the die so that the power supply to the design structure is tapped from the nearest tap on the grid. The power grid ensures the even distribution of power supply to all the design elements. The power grid consists of *rings*, *stripes*, and *rails*. The *ring* carries power lines VDD and GND around the die. Stripes carry VDD and GND from rings all around the die and rails connect the VDD and GND signals to standard cells or design element (including macro or memory) as shown in Figure 4.

SoC designs with many power supplies use many power grids. Power distribution planning is critical part of SoC design as it is the lifeline of the SoC. Utmost care is taken to ensure that every design cell is connected to VDD and GND signals before other signals are interconnected in the physical design.

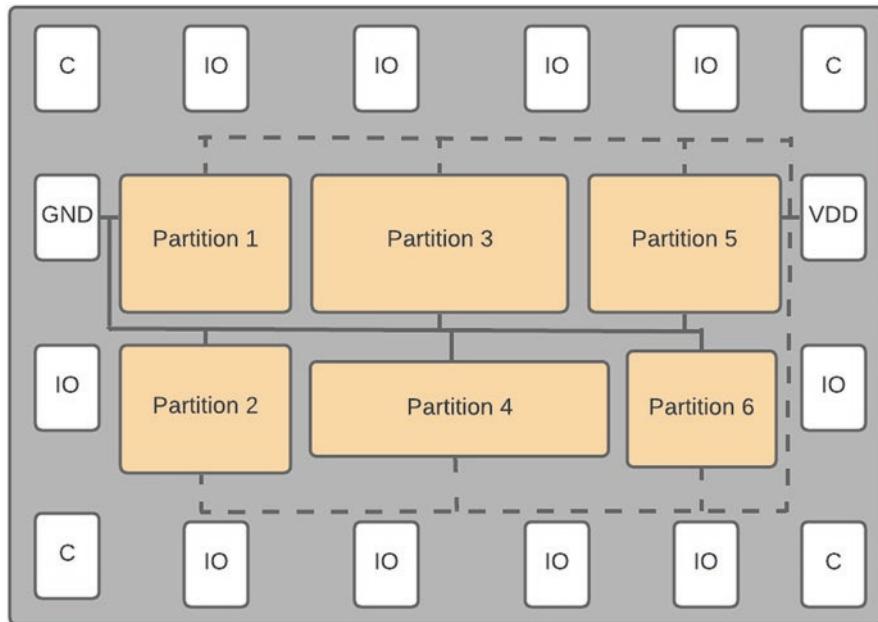


Fig. 3 Example floor plan

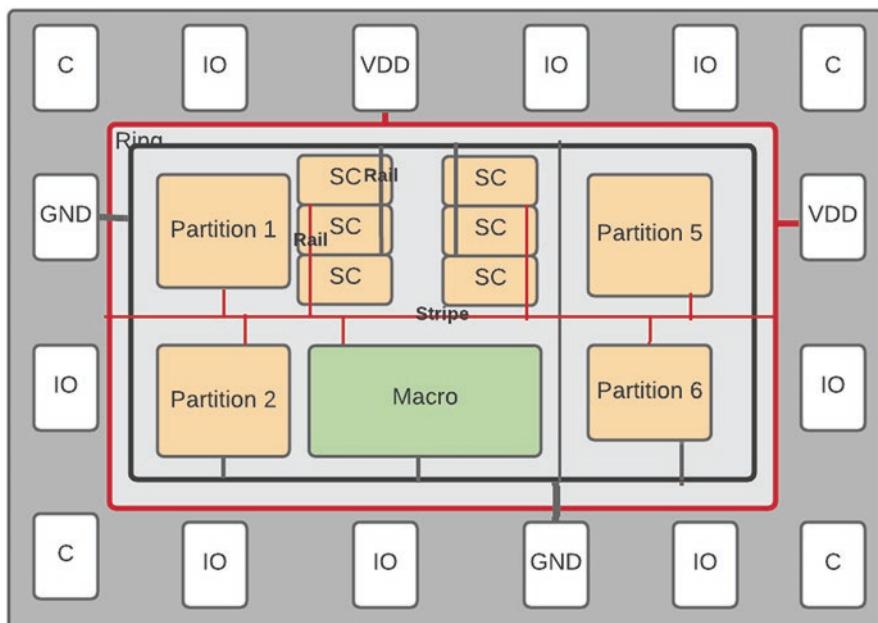


Fig. 4 Power Grid plan

## ***Placement Considerations***

Following are the other placement considerations:

1. All design elements using the same power supply voltage are combined in one module.
2. The macros and design elements following same design guidelines and interacting with each other are grouped together. Some examples of design guidelines are same module spacings, and guard bands around the macro for isolation.
3. The memories following similar guidelines are clustered into the same partition.

## **Two-Step Synthesis of SoC Design**

The SoC design database with design netlist and constraints are written out using PD tool after floor planning. The design netlist is used for logic equivalence check with the synthesis netlist to ensure that the process of the floor plan has not disturbed the functionality. Any nonequivalence must be corrected.

In two-step syntheses, the design constraint with floor plan information is used for resynthesizing the design. The design synthesis process accepts planned design netlist, Placement constraints and timing constraint to update optimum placement aware design netlist. Two step synthesis helps to meet the PPA goals of SoC design easily. The floor plan is repeated with the newly synthesized design netlist. The SoC design after successful floor plan has all standard cells, macros, IO pads placed in identified areas in the design layout.

## **Placement of SoC Design**

Once the floor plan is done, the Macros are placed in the identified floor plan following all the necessary guidelines. In this stage, the IOs are placed in the IO site or IO area, standard cells are placed in the rows created during the floor plan for the standard cell area. Design placement is continued on the same PD tool environment. Tool support many user friendly help for successful placement. One of many such help features is Macro orientation identifier called *fly lines*. Fly lines show the connection of the macro or design component under consideration with the rest of the logic in the SoC design. This gives visual help to find and optimize the interconnect lengths by correct placement.

The steps involved in placement are the following:

*Preplacement:* In this stage buffers and antenna diodes are added at the macro and block-level IO ports. *Tap cells* are added to avoid latch-up problems in the design.

*Spare cells* are added to make the design failproof with metal corrections required in the further stage of the design process. Spare cells are used for last-stage changes in design as a part of electronic change order flow (ECO) to fix timing or functional defects.

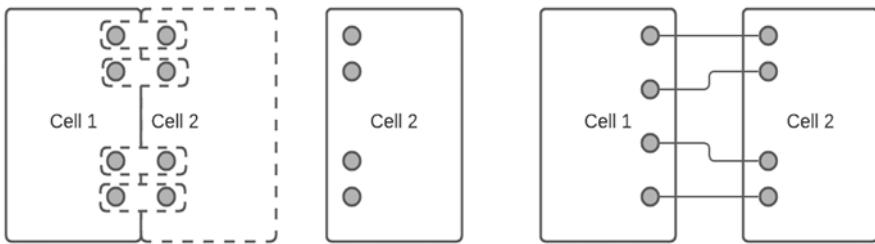
*Coarse placement:* Coarse placement identifies congestions due to preliminary routing and probable timing violations. In this stage, you identify the correct location for each of the design cells.

*The legalization* stage of placement moves the cells to the identified positions considering the geometric and layout rules of the SoC design. This process may result into timing violations. These violations must be fixed by buffer insertions. This is an iterative process. Incremental optimization of design placement takes place until the design is fully placed in the core area correctly without any timing or design rule violations.

*Next stage in design flow is synthesizing logic with appropriate drive strengths for driving the high fanout net (HFN) in the design.* Clock and reset signals in the design are excluded as they need special consideration during the design. Chip enable signals in processor-based subsystems, scan-enable signals, are examples of the HFNs. HFN is a signal which drives a larger number of loads. There is a limit set on the maximum number of loads a cell output can drive in any process technology. If a signal exceeds this number, a net is considered to be a HFN. The PD tool in this stage automatically adds the buffers in the path of these signals. If a designer wants to take care of these nets himself, HFN synthesis can be overruled. However, it is necessary to add buffers manually for HFNs to avoid functional failures. System clock signal is a high fanout net which requires clock buffers in its interconnect path to drive large load. During physical design clock signal is routed to all design objects during clock tree synthesis stage and not in placement stage. To avoid PD tool considering clock as HFN you must declare it to be excluded event as *idle* signal. If the clock is not declared as idle, it is considered as HFN, and buffers are added to its paths in this stage.

In the Placement optimization stage, all the placed cells are incrementally adjusted to avoid any further violations. In this stage, congestion analysis for design objects is carried out. *Blockages* come in handy during this design analysis. *Placement blockages* are called Hard blockages, and soft blockages are used for placement optimization. No design objects or cells can be placed in blockages. The hard blockage is an area on the core area where standard cell placement is forbidden. Soft blockages are areas restricted to only a certain number of standard cells or certain cells. Even areas around standard cells can be a blockage.

*Scan chain reordering:* Design netlist from synthesis will have scan flops connected but in the placement process, the scan chains order will get disturbed. So, scan chain reordering is done after placement of design elements. This optimizes scan chain lengths in a design.



**Fig. 5** Abutted and non-abutted cells and interconnections

### Placement Methods

*Abutted*, *non-abutted*, and *partially abutted* are three methods of placing design components, cells, and macros. In Abutted placement, the two connected cells are placed adjacent to each other. It is also called *channel-less placement*. Interconnects are not required to interface the two design components. The interface IO pins of two macros or standard cells are aligned and abutted so that connection happens on their IO contacts. This requires the cells to be carefully designed such that their IO pins are aligned such that their placement close to each other forms connections. *The pitch matching process* is used to design standard cells and other design components wherever possible. Pitch matching is done by stretching and shrinking of cells so that the bounding boxes of connected cells align and can be connected. For example, if an adder module is to be connected to a subtractor module in the design, the adder cell or subtractor cell is stretched so that the pitch of their IO pins match and when they are placed adjacent to each other they are connected by large contacts.

In non-abutted placement, the two macros are separated apart and interconnect connects the IO pins of the two design elements. Figure 5 shows the cells in abutted and non-abutted methods. These standard cells are part of a technology library obtained from the fabrication house which must be used during the synthesis stage of design to generate the SoC design netlist.

*Partially abutted* macros placement uses a combination of both abutted and non-abutted macros to place them. Appropriate methods for placing the macros must be decided in the floor plan stage itself.

## Design Verification Checklist for Final Floor Plan

### *Input for Floor Plan*

Following are the input to a design floor plan process:

- SOC Design netlist in (.v or .vg) format
- Technology file in (.tlf) format
- Timing library (.lib) format

- Physical library (.lef) format
- SoC design constraint file (.SDC) format

The die area and other terminologies for SOC physical design are shown in Figure 6. The IO pads are placed in the *I/O pad area or I/O Site*, logic function blocks are placed in the *Core area*, Standard cells of the design blocks are placed in the *standard cell row or site*.

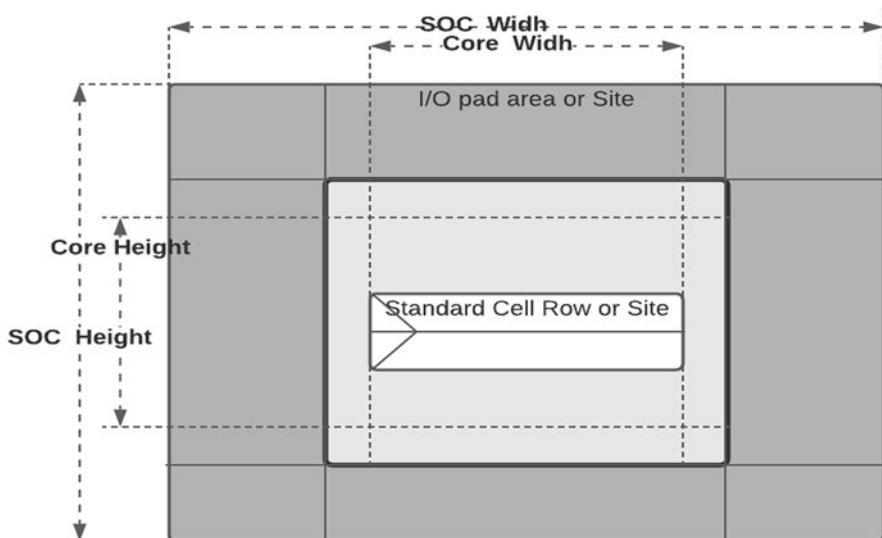
Core height can accommodate only a limited number of standard cell rows and columns. These are called horizontal and vertical resources. It depends on the cell density which is dependent on the chosen process technology. This is dictated by the process capability of the fabrication house.

Following are the design steps of Floor planning:

- SOC design netlist is imported onto the Physical design tool. Design can be in Verilog or VHDL file. It can be flat or hierarchical. A flat netlist is the design file that has only one level of hierarchy.
- As per the estimated Die size, logic core height, and width of die layout are created within the die boundary. Core area depend on the *Aspect ratio*. Aspect ratio is the ratio of horizontal routing resources to vertical routing resources. The shape of the die depends on the aspect ratio.

$$\text{Aspect ratio} = \text{height} / \text{width}$$

- *Core utilization* is the ratio of the core area occupied by a SoC Core (standard cells, Macros, memories) to the total core area. If the core utilization is 70%, the



**Fig. 6** Die boundary and area definitions

Core components placed in the core area uses 70% of the area and 30% of the core area is available for interconnect routing.

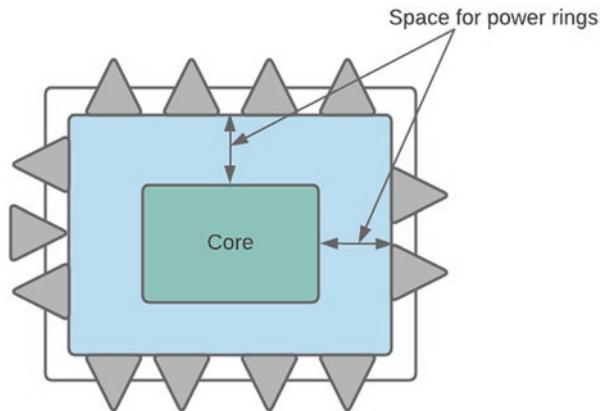
$$\text{Core utilization} = \frac{\text{Area occupied by (standard cells + macros + memories + special cells)}}{\text{Total core area}}$$

- Typical core functional blocks are placed in core area as shown in Figure 7.
- IO pad sites are created to place IO cells. IO pads are assigned to the IO box. Place and Route tools support placer modules with layout editors wherein preferred pin placement is read into the placer module. EDA tool support scripts wherein IO pin details are mentioned. Typically, the details include the shape of the IO ring, spacing between two pins, starting coordinates, metal layers for connecting the wire, pin dimensions, skipping distance from the chip corners, etc. Figure 7 shows an example placement of IOs.

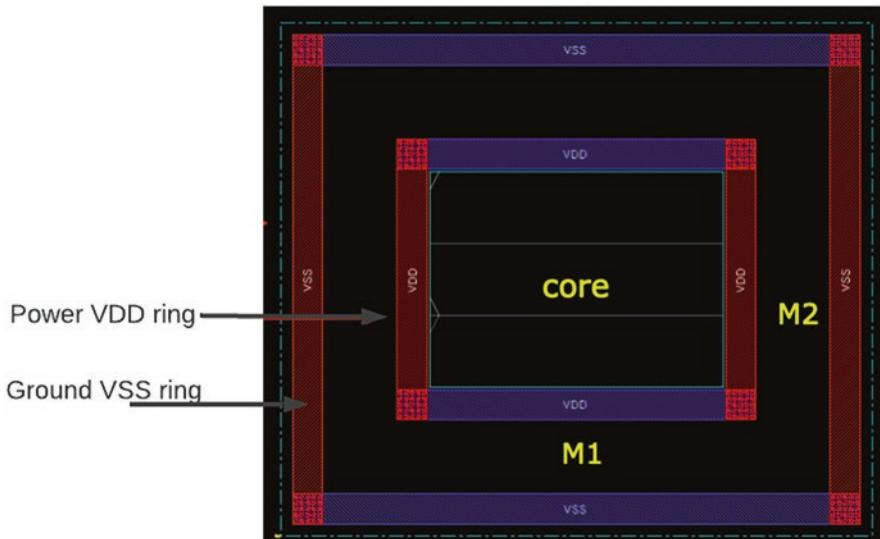
Part of IO pin placement script file for pin placement is as shown in code below:

```
(globals version = 3
io_order = clockwise ##place pins in this order
total_edge = 4 ## 4 edges on the IO box
space = 2 ## global spacing of 2um between pins
)

(iopin
## start pin definitions
(left ## pins on left side
)
(right ## pins on right side
(pin name = "O[0]" ## pin name
layer = 3 ## metal layer for connecting wire
width = 0.5 ## pin dimensions
depth = 0.6
skip = 2 ##skip 2 positions to get away from corner
place_status = fixed
)
(pin name = "O[1]"
layer = 3
width = 0.5
depth = 0.6
place_status = fixed
)
....
.... ## Continue for other pins, including right and bottom sides
```

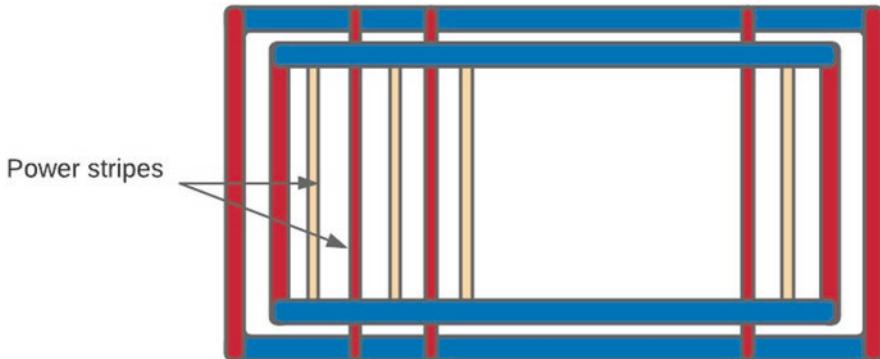


**Fig. 7** IO Placement in the IO area of the layout



**Fig. 8** Power Ground rings

- Power pad, Ground pad, and Signal pads are three types of I/O pads. There can be many pairs of power and ground pads in SoC design. A minimum number of power ground pads needed for SoC design is estimated by estimating the current consumption of the design. The number of power ground rings, width and dimensions, and metal layers used are defined in the script which is read by the placer module of the EDA tool. Common power ground rings are shown in Figure 8.
- Once the power ground rings are made, power stripes are drawn to the power supply ring so that all the core cells are powered well. The power ring with power stripes is shown in Figure 9.



**Fig. 9** Power Ground rings

- Technology files in library exchange format (LEF) include process technology-specific characterization parameters of metal layers, vias, and so on. Standard cell library defines an abstract view of each standard cell (box, pins, obstructions) with defined metal layers for pins.
- Macros are memories, analog, and digital blocks available as hard macros. Macros are placed in the core area considering the necessary defined guidelines. The placement of macros has a great impact on the quality and performance of the SoC design. Macro placement can be manual or automatic. When the number of macros are small, they are placed manually in the die core area. Macros are manually placed based on their connectivity information to IO pin/pads and macro to macro. Macros are automatically placed using tool feature if there are large number of them. Macros are placed near the core boundary leaving good space all around the macro. The space is intentionally marked around the macros called a *halo* or *keep out margin* where no other macros or standard cells are placed. Halos are associated with macros. Halos move with a macro if the macro is moved. Macros in the same hierarchy of design are placed close to each other. Macros are placed such that there are sufficient routing channels for interconnections through metal layers with the macros or other logic it interacts. Consider two macros with 25 IO pins each and with the pitch of 0.6 to be interfaced. If in the chosen target technology, six horizontal metal channels (M0, M2, M4, M6, M8, M10) and six vertical metal channels (M1, M3, M5, M7, M9, M11) are used for interconnects, then the channel width between two macros to be planned is computed as follows:

Channel width =  $(\text{Total number of pins} * \text{pitch}) / \text{Number of horizontal channels or number of vertical channels}$

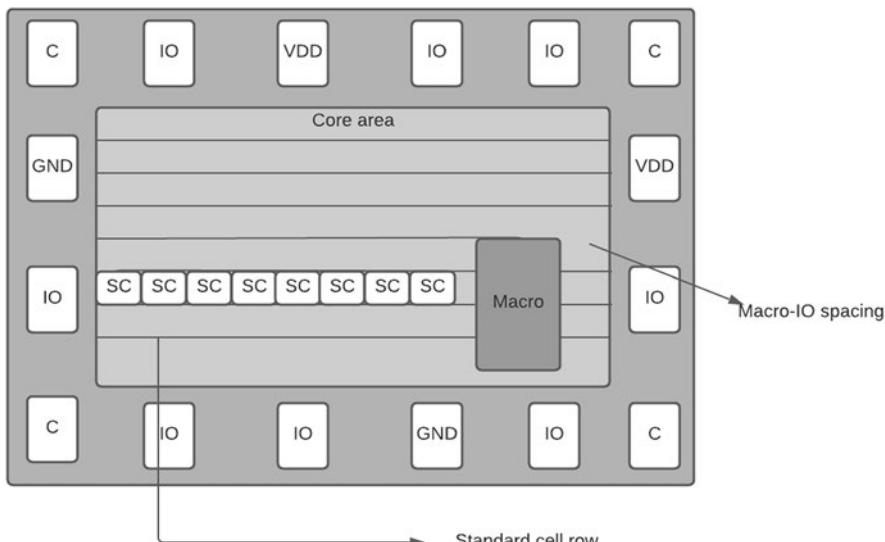
$$= (25 + 25) * 0.6 / 6$$

$$= 30 / 6 = 5$$

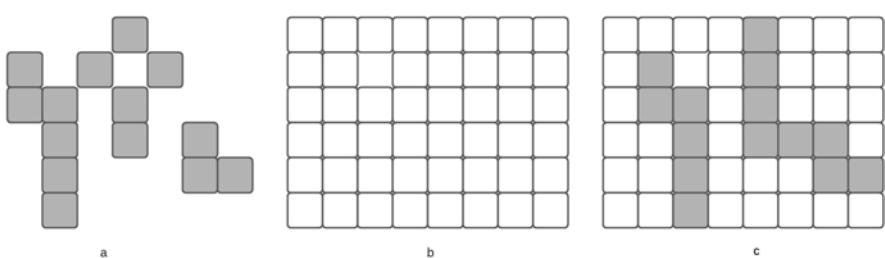
- The standard cells in the design netlist are placed in the areas marked as rows. The standard cell rows are of fixed height ( $x$ ) and varying width to accommodate different logic. The standard cell rows are multiples of base heights ( $2x, 4x$ ) which can be defined to accommodate special standard cells of bigger heights. Once the row is chosen, all cells placed in the row are of the same height. Figure 10 shows the standard cell rows in a die area.

There are some restricted areas created where the cell is avoided. This is to avoid signal interference from neighboring blocks. These are called *blockages*. Blockages are hard or soft. Hard blockage areas are restricted totally from cell placement and the soft blockage permits certain types of cells or a smaller number of standard cells to be placed in the blocked areas.

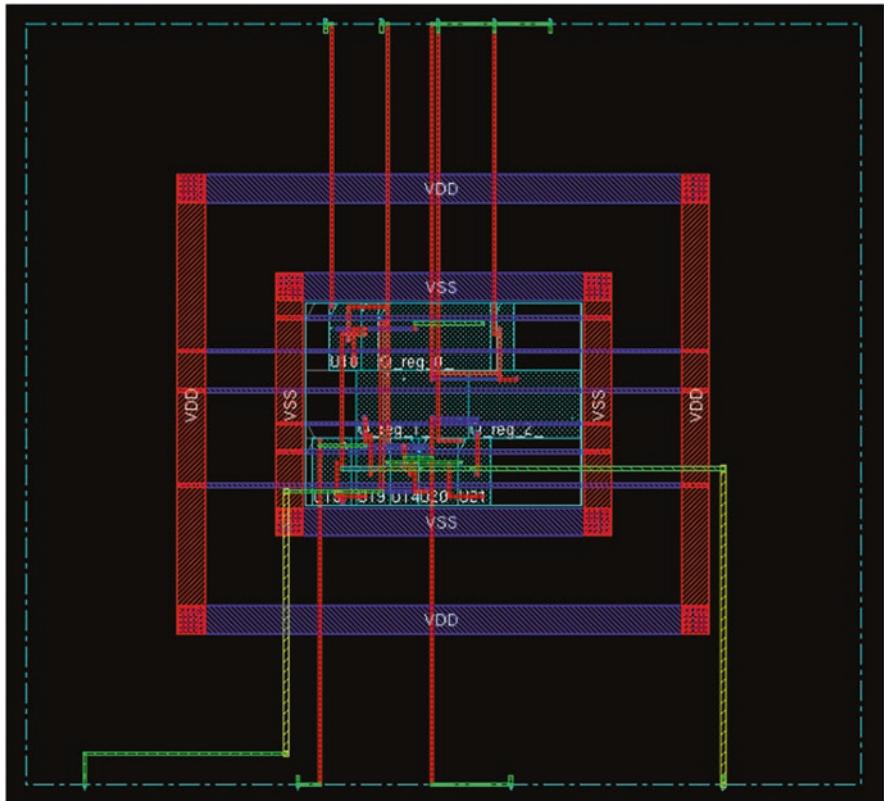
- Placement of power grid is planned across the die according to the power requirements of SoC design. There can be one or many power grids in SoC depending on its power requirement. The fully placed design is shown in Figures 11 and 12.



**Fig. 10** Standard cell rows in die area



**Fig. 11** (a) Design components (b) Tiles. (c) Placed design



**Fig. 12** Placed design layout

- Core cell area is planned for some additional physical only cells which are needed for reasons other than functionality of the SOC. More on this will be explained in Chapter “Clock Tree Synthesis (CTS) in SoC Physical Design”.

## Floor Plan and Placement Validation

Once all the IO cells, standard cells, macros, and power pads are placed, the parasitic delays are extracted from the layout to compute cell and interconnect delays for timing analysis. The advanced extraction tools execute the trial route and estimate the delays of interconnecting paths in the design. The PD tool automatically improve the performance by swapping pins, moving cell placements, and resizing the cells with different drive strength-delay parameters. Physical design tools can extract the cells and interconnect delays and report timing of the design paths.

The placed design netlist file, the timing and physical constraint files are written out from the physical design tool which is used for placement aware resynthesis of the design for incremental optimization and performance expectation of the SoC. The placed netlist is checked with a golden reference netlist (which is used as input) for logical equivalence. Once the placement is found satisfactory, the design database is taken up for clock tree synthesis (CTS) which is the next step on VLSI physical design.

## EDA Tools in Physical Design

EDA tools play a very important role considering the fast-growing complexity of the system designs. Over the past couple of decades, semiconductor process technology has become very complex as device physics changes so much in nano scale or deep submicron processes. Transistors are now three-dimensional devices with many fins that exhibit counterintuitive behaviors. The feature size in nanometer scale of advanced process technologies demands multiple exposures (multi-patterning) using masks or reticles to accurately reproduce these layout structures of design on a silicon wafer. This has added substantial complexity to the design process.

The growing VLSI complexity has “slowed down” Moore’s law scaling but porting the design to a new process node is still an option, with increased challenges and exorbitant cost. It is observed that the new process nodes do not offer the expected rate of increased density, performance, and power reduction. The evolution of semiconductor process technology is reaching molecular limits, and this is slowing the exponential benefits of Moore’s law. In this context, the innovation has continued to explore new ways to compute adopting artificial intelligence (AI) techniques. At the chip level, the single-chip approach to design is getting replaced by multiple chips/dies of silicon for specific applications, each with a specific purpose and integrated into a single package. System in package (SIP) is seen to be again an option going forward. It is also believed that previous innovation was driven by scaling down the feature size and future innovation is predicted by a solution complexity. EDA tool vendors have recognized this trend and offer hyper convergent design flows with tools and platforms addressing the design challenges of system. Leading EDA tool vendors are upgrading the EDA tools to address challenges of system complexities and the process technologies with miniaturization using advanced AI methods of reinforcement learning (RL) [1]. The AI techniques train the design algorithms to result in counterintuitive, placements of design elements in unconventional shapes, achieving excellent design PPA goals. Arriving at the optimal and right floor plan for the placement process of VLSI design is an excellent early adopter of this concept.

There are many advanced EDA tools and platforms used for SoC design from different vendors [2].

## Challenges in SOC Design Floor Plan and Placement

A good floor plan leads to a good layout in VLSI SoC designs. It is more challenging as there can be many possible floor plans for the desired PPA goals. Sometimes, the chosen floor plan can be overdesign or killer design demanding more time and effort to meet for the design goals. This could be expensive in terms of design effort, time to market, or fabrication challenges. Sometimes, the decision fatigue may lead to arriving at finalizing the option which is barely optimum for the design goals. The only way to address this problem is to understand from design experiences of the past manually which can be the limiting factor. Artificial intelligent techniques which can automatically scan through the past design database and learn itself to converge to meet the design goals is a promising method. It is a complex problem to address requiring the huge correct design databases and the machine runs several times to make it learn from the database. Again, the risk is that the quality of learning, which will be as good as the reference design dataset used to learn.

## References

1. Keynote addresses at Hot Chips 2021
2. <https://eda.sw.siemens.com/en-US/ic/calibre-design/physical-verification/>

# Clock Tree Synthesis (CTS) in SoC Physical Design



## Digital Logic in SoC Design

Digital logic forms a major part of the system today, in addition to analog blocks and memory macros. Digital logic is a combination of both *sequential* and *combinational logic* circuits realizing the function. The *combinational logic* cells like *NAND*, *NOR*, *INVERTOR*, *MUX*, and *DECODER* cells form the building blocks of major combinational logic which do not need clock. This combinational logic design processes data which is stored and fed to the next stage of data processing in data flow architecture of the system RTL design. *Flip-flops*, *latches*, *synchronous memories* are the major logic cells or design elements in the sequential logic design using clock. Clock is the lifeline of the digital *sequential logic cells* in the system design. The data flow control and storage functions of the system are realized using sequential logic circuits. Most Memory blocks are sequential logic of different circuit configurations, which many times require a clock. In a complex system on chip, there will be hundreds of thousands of standard cells out of which typically the ratio of combinational to sequential logic cells will be around 10:1. If the design has one million combinational cells, there will be thousand sequential cells. There are asynchronous circuits which promise higher speed performance but are extremely difficult to design and verify. This do not require clock signal to operate.

Most digital designs are synchronous in nature. Clock synchronizes functions with proper timing in a digital circuit designs. It uses single reference clock and one or more derived clocks. Derived clocks are generated using reference clock. Clock signal is the main timing control signal for the design in data flow architecture.

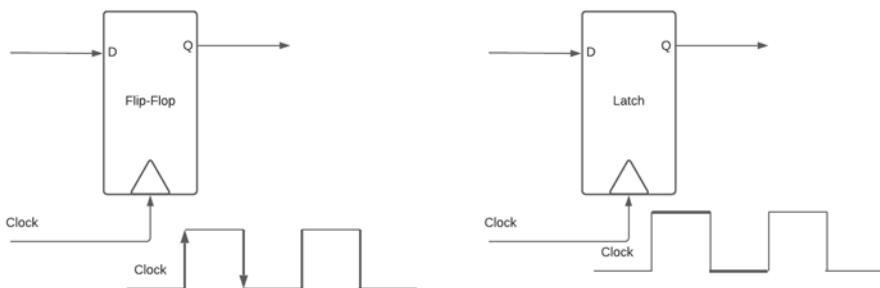
## Sequential Cells in Digital Logic Design

**Sequential cells:** Latches and Flip-flops are sequential cells predominantly used to realize digital logic design. These store system states, configurations, and data in the design. These are also called re-convergent logic cells or feedback cells. The output of these cells depend not only on current inputs and historical data. Figure 1 shows the storage elements. In flip-flop the data D is stored at the rising or falling edge of the clock signal and in Latch the data is stored when the clock signal is high or low. When the storage element is active at the negative logic of the clock, it is represented by a bubble in the clock signal in the symbol.

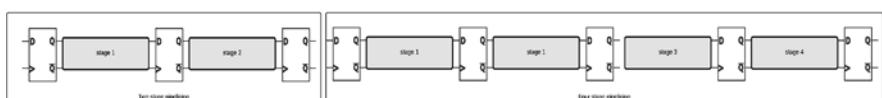
## Pipelining

Concept of *Pipelining* is used in most synchronous digital logic to increase the speed of the processes. It is a process of partitioning the data path of the design by mutually exclusive functional operations so that they operate independently but in order. Each of the partition is called a stage. In a way, design data path is a set of pipeline stages separated by a storage element. In the processor core, pipelining increases the number of instructions processed. Figure 2 shows two-stage and four-stage pipelining with a gray shaded box representing process of a stage separated by storage elements.

Speed performance of the digital system increases by the number of pipeline stages used in the logic. Pipeline stages are a combinational logic design. It is separated by a storage element. Therefore, the digital part at the lowest level circuit will



**Fig. 1** Storage elements in design



**Fig. 2** Two- and Four-stage pipelining

essentially be two storage elements separating the combinational logic cells. Same concept is extended to the SoC architecture. Consider a processor with several instructions NI with each instruction taking TPI number of clocks of period CP units, the total execution time of the processor is given by

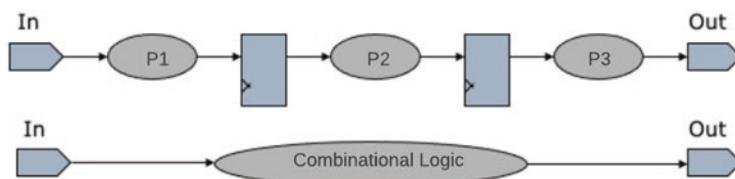
$$\text{Execution time} = \text{NI} + \text{TPI} + \text{CP}$$

Processor speed is increased by reducing execution time which can be done in three ways:

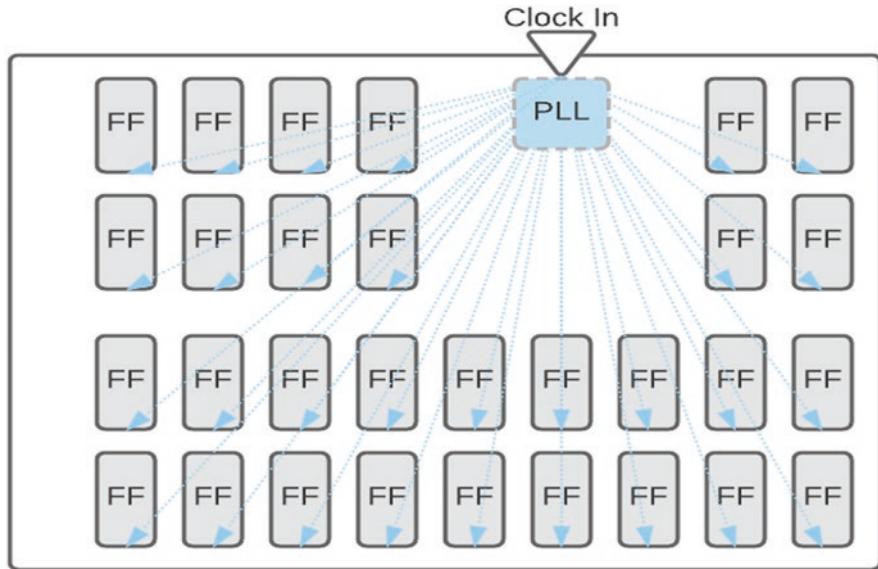
- *By reducing the number of instructions which is instruction set architecture (ISA).*
- *By reducing the number of clocks used per instructions using concepts like pipelining in SoC architecture.*
- *By reducing clock period which is done by a combination of choice of process technology and logic design.*

Right clocking strategy with clock tree synthesis (CTS) increases speed performance *by logic design*. The simplest timing path in digital design is pipeline logic between two sequential elements shown in Figure 3. In the figure,  $P1$ ,  $P2$ , and  $P3$  are pipeline segments.

For the pipeline stage to work correctly, it is necessary that the logic delay in the combinational block must be less than the arrival clock edge. This should repeat for every such pipeline logic segment. As the data progresses from one pipeline stage to the next in the data path of the design, there is subsequent data that will occupy the current segment of the pipeline. It is necessary to synchronize the pipeline stages so that no data gets corrupted because of pipeline delays. Controlling the flow of data for data synchronization is done by a clock signal. There must be enough time for combinational logic to process the data before it gets latched by the next stage pipeline else it results in functional failures. A clock fed to sequential elements of the entire design are generated and driven by a clock source like DLL or PLL internal to the design. Sometimes, the clock signal is fed through a input pin of the SoC design from an external clock source. For proper functioning of the sequential elements, it is necessary that the timing requirements of all the sequential cells in the design are met. This requires proper control of the arrival times of clock and data signals at the inputs of the



**Fig. 3** Pipeline stages and design timing paths



**Fig. 4** Clock network for synchronous elements before clock tree synthesis

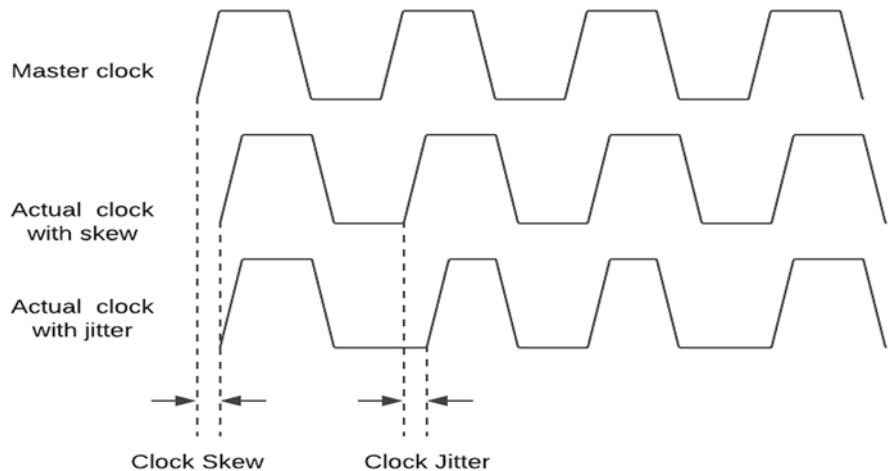
sequential elements. These sequential cells in the design can be anywhere in the physical design layout. However, practically it is not possible to get the clock signal arrive at the same time at the clock input points of all the sequential cells. This is because of differing lengths of interconnects on the clock paths from the clock source to the clock input signal of the sequential cells as in Figure 4.

## Need for Clock Distribution

The clock signal is not routed to all the flip-flops directly due to the signal loss which could occur that affect the functionality of the design. In today's deep submicron technologies, the interconnect delays impact the functionality equally as design elements. For the clock signal to get routed to sequential cells placed at the other corner of the core area in layout will have a substantial signal drop. Also, the clock signal is connected to thousands of sequential elements in the design causing heavy load on the clock input pin. Clock signal is also high fan out net in the design.

## Clock Parameters

The clock signal is distributed during physical design, to all sequential cells scattered across the chip layout. Clock distribution manifests into two major physical parameters of the clock signal called *Clock Skew* and *Clock Jitter*. Clock



**Fig. 5** Clock skew and clock jitter

distribution must be done such that these effects on clock are controlled to be minimum and is within acceptable limits for the design to work correctly.

*Clock Skew* is the maximum difference in delays in the arrival times of the clock at two leaf nodes called the launch sequential cell and the capture sequential cell of the clock distribution network. In simple words, it is the maximum difference in arrival times of any two sequential cells from the same clock source in the design. This is shown in Figure 5.

*Clock Jitter* is the maximum cycle-to-cycle delays of the clock signal at the same leaf node. It is the variation of clock cycle time at the same leaf node as shown in Figure 5.

## Clock Tree Synthesis

A clock distribution network consists of a series of buffers/inverters placed in the clock signal path starting from clock source point to clock inputs of sequential cells. CTS while distributing the clock must consider:

1. Minimizing the clock skew between launch sequential and capture sequential element and Jitter.
2. Drive the entire buffer/inverter set used in the paths of the clock en route to leaf nodes.

The process of distribution of clock to all sequential cells of the design to minimize clock skew is called *clock tree synthesis (CTS)*. Clock tree synthesis (CTS) inserts inverters/buffers in the clock path starting from the clock input pin to sequential cells with a minimum skew or balanced skew. CTS is carried out by different

methods for different SoC designs demanding different PPA goals. For SoC designs working with clock frequency less than 1 GHz, a physical design tool is used to automatically synthesize clock tree but for SoC designs working with clock frequency more than 1 GHz, the CTS process is done manually on the tools platform as it is very challenging to achieve PPA goals.

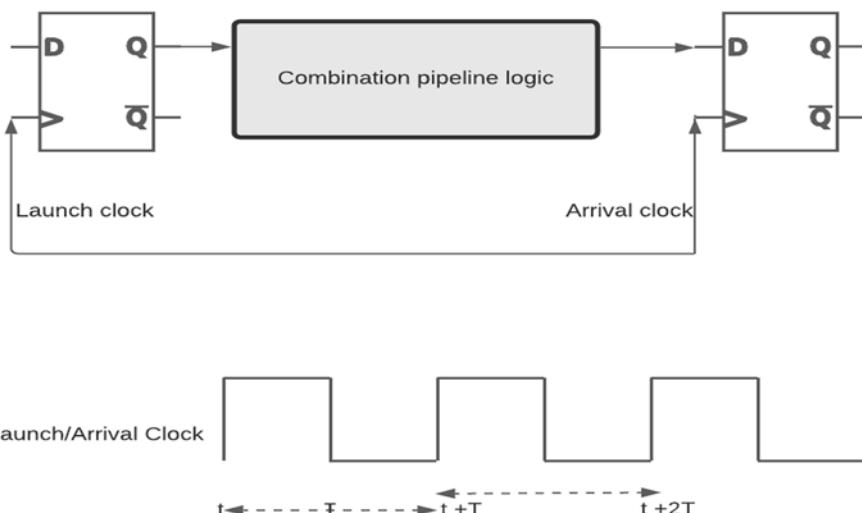
## Design Constraints

In addition to skew and jitter manifestations, for correct functionality of the design, it is necessary to meet the *Maximum* path delay (Setup time) and *Minimum* path delay (Hold time) requirements of the SoC design timing paths.

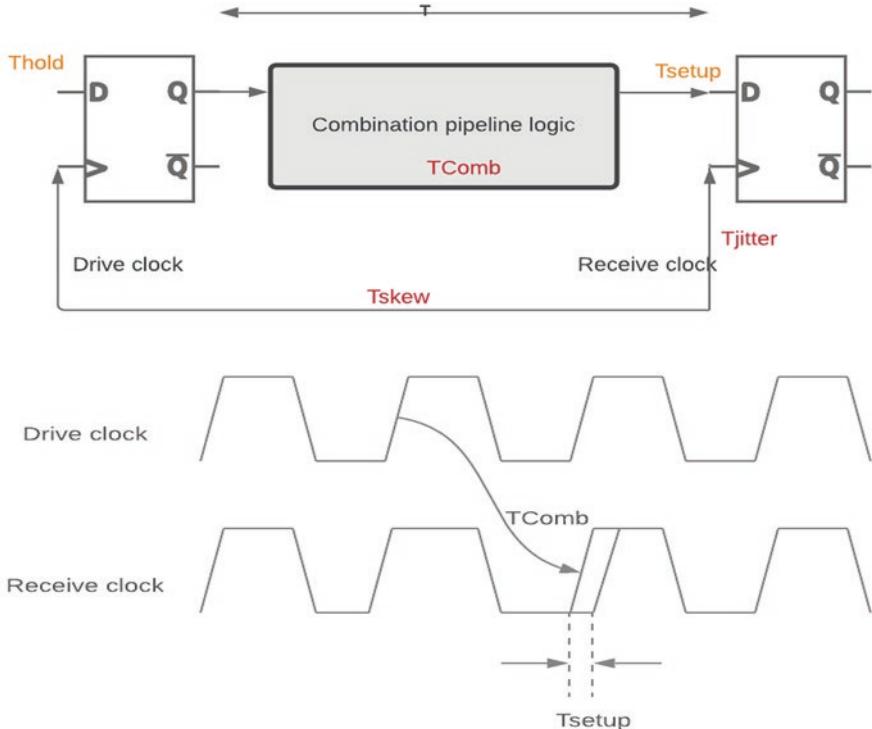
### Maximum Delay Constraint

Consider the design path shown in Figure 6.

Data processing in combinational logic begins once the data is available at its inputs. Data is available at the input when the data is latched by the drive clock correctly. In every technology node, data latching will take some time after the clock edge is called *hold time*. That means for data to correctly get latched, data must be stable for hold time  $T_{\text{hold}}$  time after the clock edge. The hold time is the characteristic of the sequential cell used from the technology library. The processing time for combinational logic for the valid data is called  $T_{\text{Comb}}$ . The processed data will be



**Fig. 6** Simplest pipeline path



**Fig. 7** Max and Min path delay constraint

available at the receiving flip-flop with receive clock only after  $T_{\text{hold}} + T_{\text{Comb}}$  time units. For data to be latched correctly by a sequential element, it must be stable for some time called setup time  $T_{\text{setup}}$  before the arrival of clock edge. Hence, the processed data gets latched with receive clock only after  $T_{\text{Comb}} + T_{\text{setup}}$  time units. This is possible only if the minimum clock period  $T$  is greater than  $T_{\text{Comb}} + T_{\text{setup}}$  time units as shown in Figure 7.

$$T_{\min} \geq T_{\text{Comb}} + T_{\text{setup}} \quad (1)$$

Considering the clock skew and clock jitter, the equation is written as:

$$T_{\min} \geq T_{\text{Comb}} + T_{\text{setup}} + \text{RX}_{\text{setClockSkew}} + \text{RX}_{\text{ClockJitter}} \quad (2)$$

Equation (2) is putting the limit on the frequency of operation of the design. For any design, the maximum frequency of operation achievable is only  $\frac{1}{T_{\text{Comb}} + T_{\text{hold}} + T_{\text{setup}} + \text{RX}_{\text{ClockSkew}} + \text{RX}_{\text{ClockJitter}}}$ .

In the equation,  $T_{\text{hold}}$  and  $T_{\text{setup}}$  are technology parameters that will be fixed by the selected processing technology.  $T_{\text{Comb}}$  is determined by logic function in the design. This is done by controlling skew and jitter parameters of the clock. The goal of the

clock distribution or CTS in the design is therefore to design the clock network in the design distributed to all the synchronous cells (leaf nodes) with minimum possible clock skew and jitter.

### ***Minimum Delay Constraint***

Minimum delay constraint for the design to work is given by:

$$T_{\text{Comb}} + T_{\text{hold}} > T_{\text{skew}} + T_{\text{jitter}}$$

### ***Quality of Design***

The maximum operating frequency of the design with skew and jitter is given by:

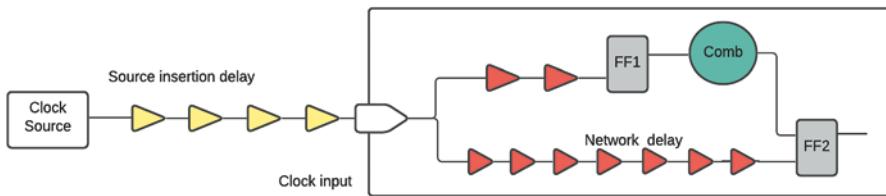
$$F_{\max_{\text{skew-jitter}}} = \frac{1}{T_{\text{Comb}} + T_{\text{skew}} + T_{\text{jitter}} + T_{\text{setup}}}$$

In the absence of skew or jitter,

$$\begin{aligned} F_{\max_{\text{without skewjitter}}} &= \frac{1}{T_{\text{Comb}} + T_{\text{setup}}} \\ \text{Frequency cost due to skew and jitter } \Delta f &= \left( \frac{F_{\max_{\text{without skewjitter}}} - F_{\max_{\text{skew-jitter}}}}{F_{\max_{\text{without skewjitter}}}} \right) \\ &= \frac{(T_{\text{skew}} + T_{\text{jitter}})}{(T_{\text{comb}} + T_{\text{setup}} + T_{\text{skew}} + T_{\text{jitter}})} \end{aligned}$$

Major design constraints used are upper bound limits for parameters like transition delay, fan-out, load capacitance, and buffer stages with the CTS goal to achieve minimum skew, jitter, and min/max insertion delays. Few definitions relevant for CTS are as below:

1. Clock latency: Clock latency is the total time taken by the clock signal to reach the clock input of the register. It is addition of clock generation latency at the source (clock source latency or source insertion delay) and the network latency of the interconnects through which signal reaches the input point on the register.



**Fig. 8** Clock insertion delay

2. Insertion delay: Insertion delay is actual delay seen from the clock generation point to the input pin of register as seen practically. It is post clock route parameter compared to clock latency. It is the target goal set as latency for the CTS engine to achieve and Insertion delay is achieved delay parameter. Clock insertion delay is shown in Figure 8.

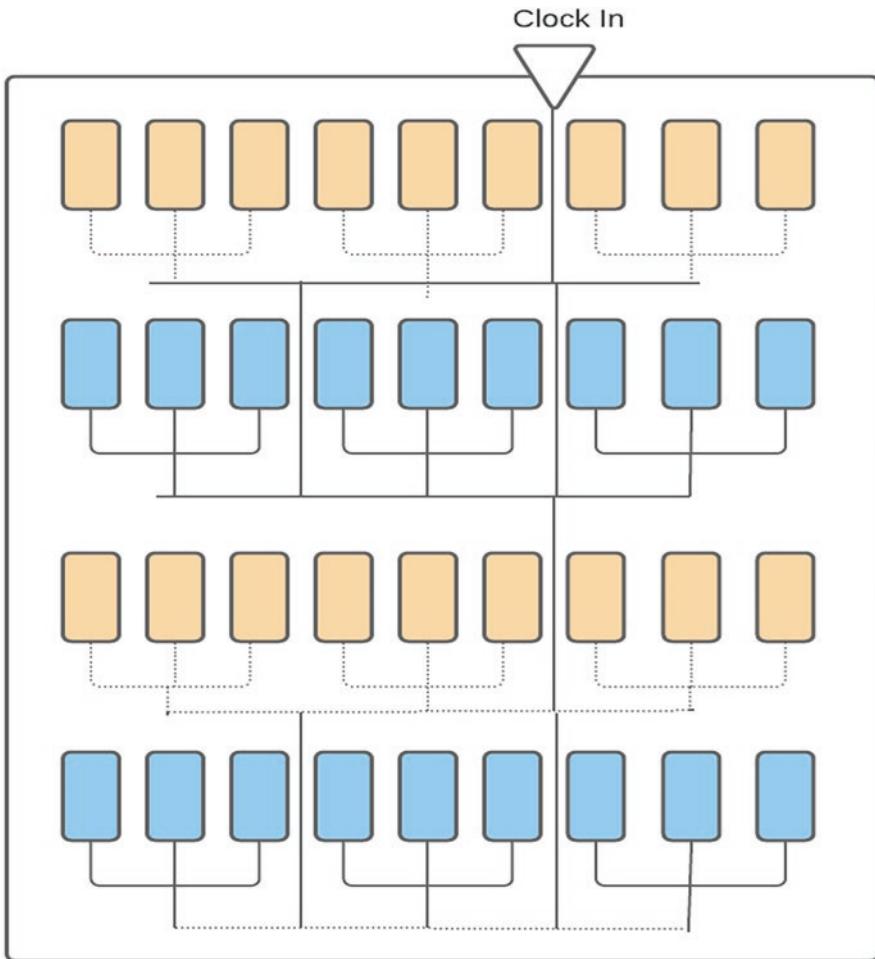
Clock width: clock width is the time of the clock signal when it is logic high. Few flip-flops need the minimum clock width needed for proper operation.

## CTS Topologies and PPA

The sequential cells placement and the clock distribution in any design will look like the one shown in Figure 9.

Clock tree is built such that the arrival time of the clock at the farthest sequential cell and the nearest sequential cell must be within the skew specification and latency specifications of the design. Clock routing is done during CTS before the signal routing which is planned in the next step as a good clock tree boosts the performance helping the design closure. In complex SoC designs, the clock tree depends on the following parameters:

- Several functional clocks are present in current day designs. They are either generated and distributed by system clock frequency or any derived clocks from primary system clock.
- The complexity of SoC design in terms of the number of sequential cells determines the size of the tree. The size of the tree determines the number of buffer stages needed to clock all the leaf nodes depending on the fan-out specification of buffer/Inverters. The clock tree depth of five buffer stages can feed the clock to 50K sequential cells within the skew spec of the design. But when the number of sequential cells is 200K, deeper clock trees must be built. The clock trees use all the routing resources which may further challenge the critical signal routing. The higher the number of stages of clock buffer/inverters, the larger will be the *insertion delays* and it is challenging to limit it below the delay specification of

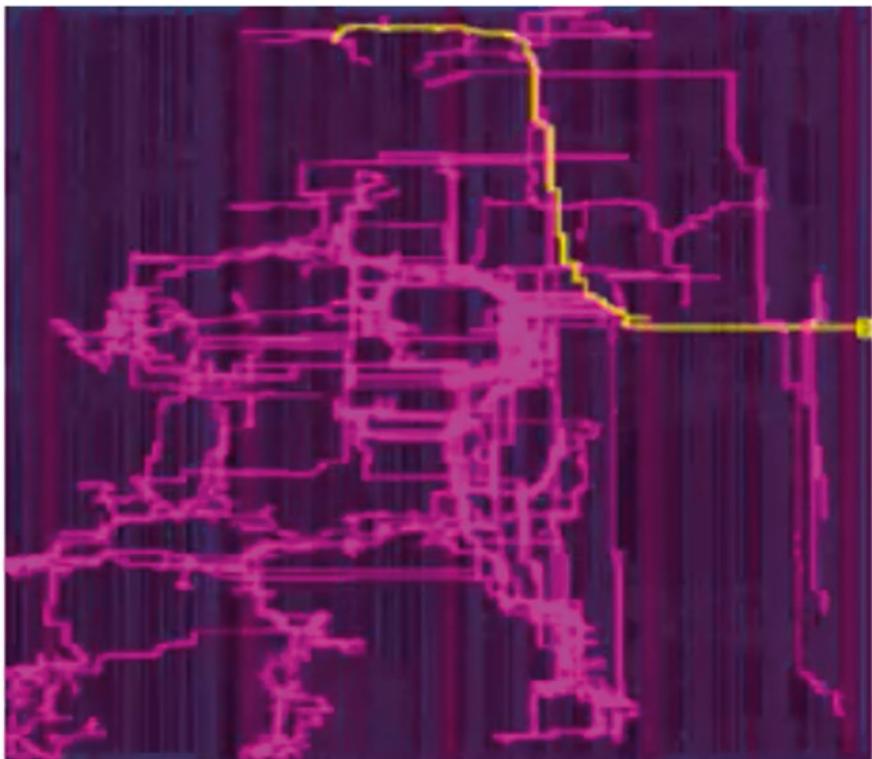


**Fig. 9** Design after Clock tree synthesis

the design. Insertion delay is the time the clock takes from the source pin of the SoC design to the farthest sequential cells in the design. This in turn affects the PPA of the design.

- Clock requirement such as frequency of operation and amplitude with clock jitter/skew parameters for different blocks in the SoC design must be considered for synthesizing clock trees. Different trees are to be built by identifying correct clock groups to avoid routing congestion and with power considerations.
- Variation in arrival times of clock at the inputs of design elements causes duty cycle distortions which could result in functional failures affecting the reliability of the chip.

Figure 10 shows the clock tree in a design layout.



**Fig. 10** CTS of design layout

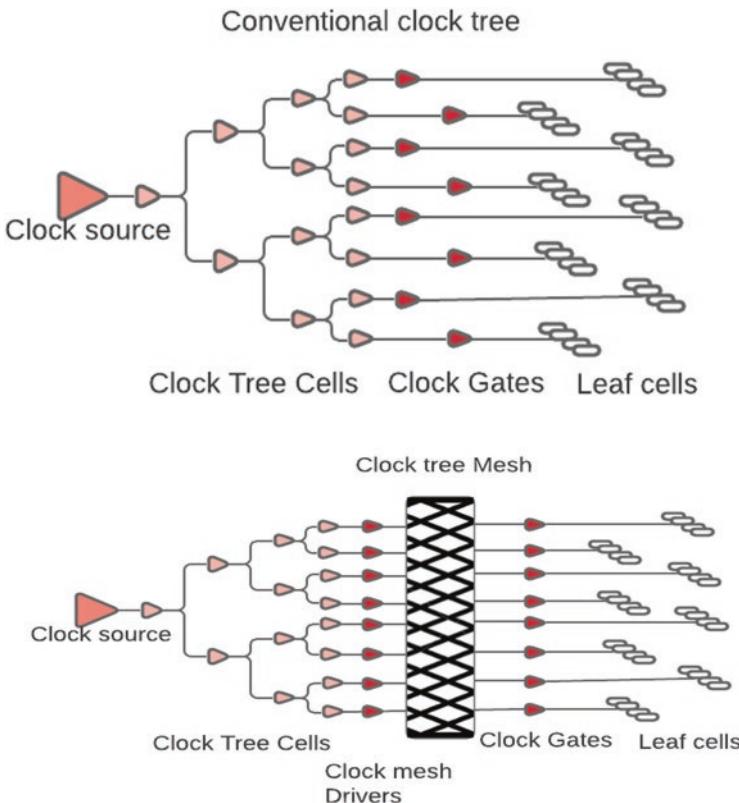
## Types of Clock Trees

SoC designs of different complexity require different clock trees for clock signal distribution. For example, clock signal of frequency of < 1 GHz for SoC designs of 50K gate complexity require a simple clock tree. This can be automatically synthesized using a physical design tool. But Clock signal of frequency greater than 1Ghz for a complex SoC designs will require special types of trees like clock mesh or clock grids to meet the design requirements. Apart from the clock tree type, these use higher metals of layer stack for routing to meet stringent design timing requirements. Conventional CTS, Multi-source clock tree, and Clock Mesh are three types of clock trees generated for the SoC designs. Figure 11 shows conventional CTS and Clock mesh-based trees for complex SoC designs.

Clock tree synthesis is one of the critical stages of VLSI SoC design which effects the PPA goals. CTS distributes and balances the load in the design by inserting inverters/buffers in the interconnect paths of clock/data signals to minimize clock skew and clock jitter. The clock tree consists of clock drivers (buffers/inverters), clock frequency dividers, level shifters for clock domain crossing, one or many

clocks insertion pins, clock muxes, and clock enable logic gates. CTS differs from high fan-out nets (HFN)s in that it uses buffer/inverter cells of equal rise and fall times. HFNs are routed by inserting buffer/ inverters with differing rise/fall times also. HFNs are basically defined for control signals like reset, scan\_en, chip\_en, etc., where balancing and power performance is not much critical. Power optimization is an important goal in CTS as it is observed that clock switching network constitute 30–40% of SoC dynamic power consumption. Criticality of CTS specifications are addressed using Non default rules (NDR) for interconnect routing in clock trees. CTS uses special cells called clock buffers where the switching point is at the center of the voltage transfer characteristic (VTC) curve with equal rise and fall times. This ensures that input clock signal with 50:50 duty cycle produces clock output of duty cycle 50:50.

Complex SoCs may require contain Ip cores or macros which have clock trees synthesized and optimized for performance. Clock trees in such design objects must be excluded from CTS in system design. This is done by the feature to exclude synthesizing tree in PD tools. Presynthesized clock trees are retained with the options like “don’t touch clock\_tree” in the PD tool as in Figure 12.



**Fig. 11** The types of clock trees generated in SoC designs

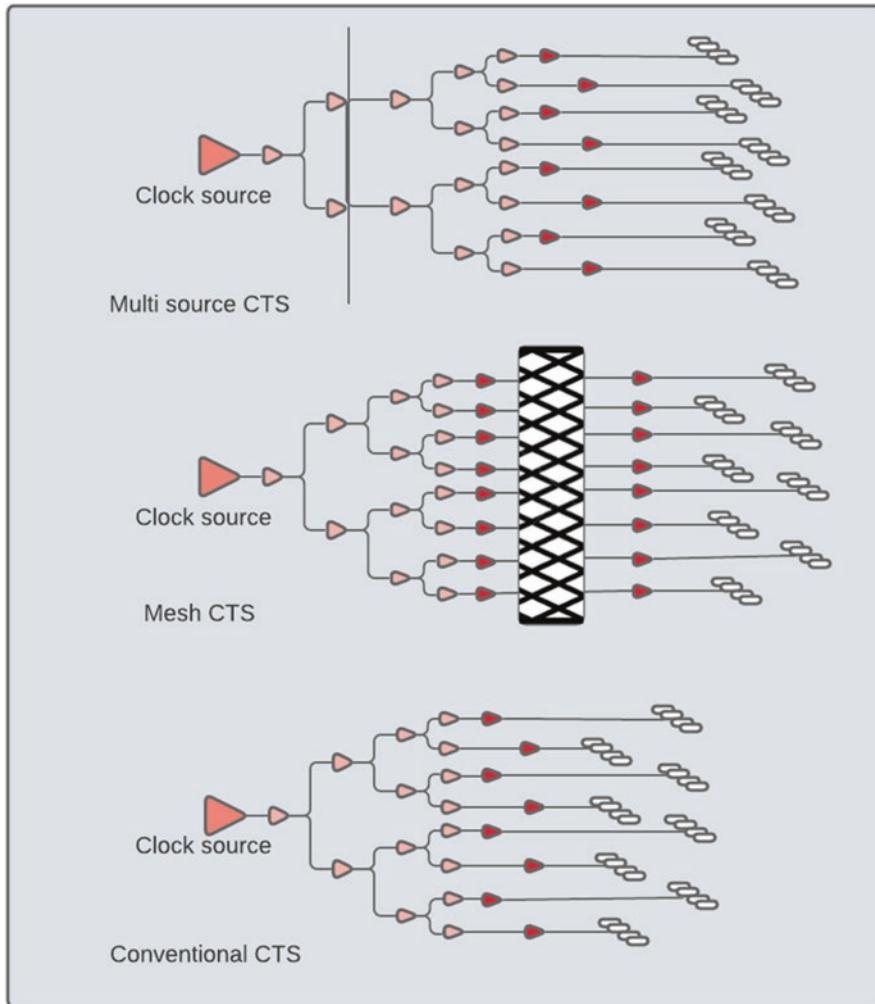
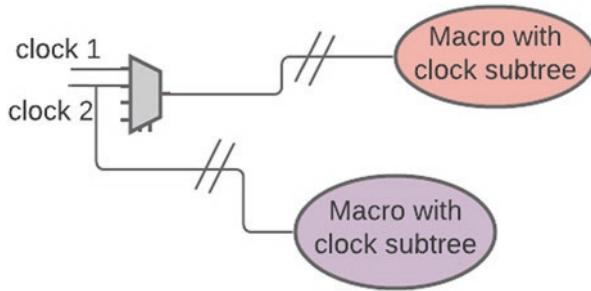


Fig. 11 (continued)

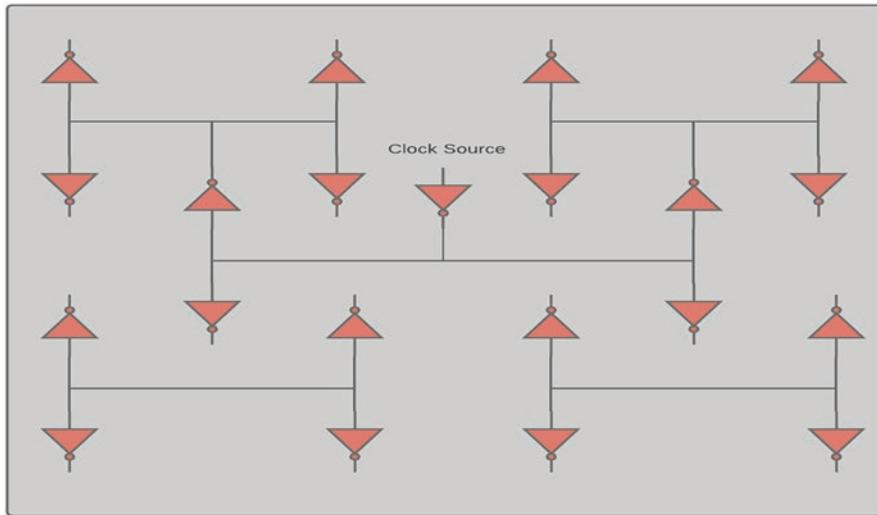
## Clock Tree Algorithms

The Clock tree synthesis (CTS) uses different algorithms listed below:

- RC-based clock tree
- H-Tree algorithm
- X-Tree algorithm
- Method of mean and medium (MMM)
- Geometric matching algorithm
- Pi algorithm



**Fig. 12** CTS don't touch for Macros with pre-synthesized clock subtree



**Fig. 13** H-Tree CTS

Out of the above, H-Tree algorithm shown in Figure 13 is most used in Clock tree synthesizers. These algorithms run to generate required clock tree when a corresponding tool configuration is enabled. The choice of clock tree depends on the design requirement and complexity in terms of levels of design hierarchy.

## Design Challenges in the Clock Synthesis

Design QOR depends on the quality of synthesized clock tree. Major challenges of clock are power optimization and minimizing spatial and temporal variations of clock signal.

**Power optimization:** Clock distribution network is the most power-hungry part of any SoC design as it keeps switching all the time. Low power design techniques such as multiple frequency clocks, gated clocks are used in complex designs. A good CTS ensures that the clock trees are properly clustered and synthesized to ensure that the low power design.

**CTS for timing optimization:** In deep submicron process technologies the system design timing closure becomes a major challenge. This is due to the increase in on-chip process and surface variations (OCV) leading to changes in interconnect and cell delays. It is a difficult task for the clock to reach every flop at almost the same instance of time to avoid timing violations. OCV affects path delays in design largely than signal and other paths. Also, clock distribution network consumes more power in system design affecting electrical properties. So, it is important to minimize the effect of OCV in the clock network by creating proper clock distribution structure to effectively reduce timing violations and variation effects while meeting the clock skew and latency requirements of the design. It is to be noted that once CTS optimization is done, the clock network is fixed, frozen, and routed. Hence, clock-related optimizations are addressed before it is finalized for signal routing. Clock path optimization includes resizing of the cells, swapping high threshold voltage (HVT), Low threshold voltage (LVT) cells in the design, and routing clock nets with appropriate sizes overruling default signal routing rules. Post CTS, only data signal paths are available for fixing design rule violations (DRV), timing violations (setup, hold, max trans, max cap, and max fan-out), and routing congestion reduction. This many times poses limitation for design closure. Also, the conventional CTS may result in worsening the timings of critical paths at the leaf level by over fixing few paths and under fixing few others. This is avoided by optimizing data and clock paths of design concurrently. This is called concurrent clock and data synthesis. This promises better QOR for the design but consumes more design time. It optimizes both data and clock path simultaneously.

**Clock tree optimization for cross talk:** Cross talk in design is unwanted signal variation seen on a net due to the switching activity of neighbouring nets. This results into variation in functional and timing characteristics of the design paths. It has unpredictable design behavior sometime resulting into fatal system failures. Clock is seen as a major aggressor signal influencing the signals in neighbouring nets due to cross talk effect in the design. It is necessary to address cross talk effect at this stage. This is done by proper shielding for clock nets and critical signal nets. Also, designer use non default rules (NDR) while fixing any of the cross talk issues found during analysis.

The design convergence for desired QOR depends on the quality of CTS and it is of utmost importance to adopt all possible ways and means in generating correct clock tree for the design and optimization.

# Routing in SoC Physical Design

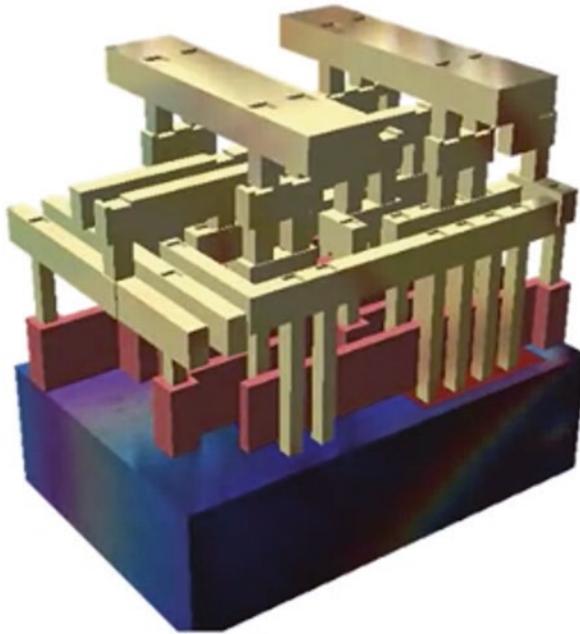


## Design Routing

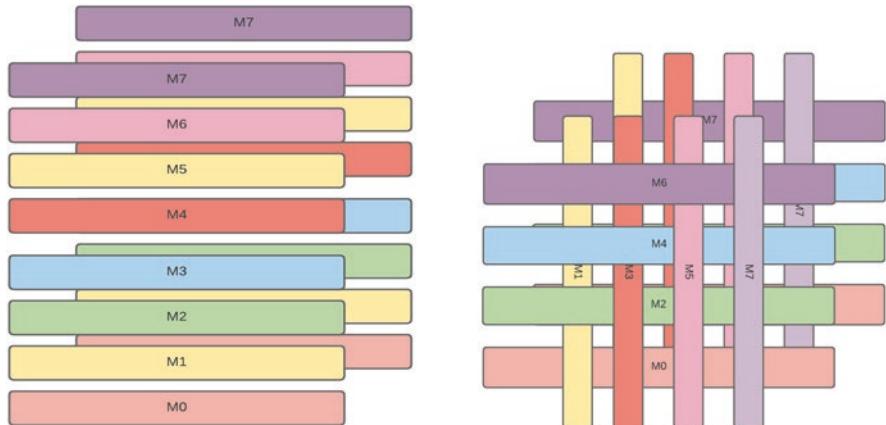
Correct operation of SoC design is when the operations of all its design constituents are correct. This requires proper interconnection of design elements and proper biasing by powering of every design elements. Power connections to every design element and macros are done in Clock signal is distributed in CTS stage of the physical design. The signal interconnections of design elements, macros and input output pins are done in routing stage. It is necessary that every signal/data path has an interconnect route in the design as defined in the design netlist. In SoC design of millions of gates there will be tens of millions of signal interconnections. Routing is a design stage where these interconnections are done. This is called routing of signals. This is done according to the fabrication rules, which ensures signal integrity when the design is fabricated. Interconnections of signals use one or many of the metal layers in the layer stack as per the targeted fabrication technology process. The metal layers make horizontal and vertical connections. Signal routing in standard cells are already done during cell designs in the base layers as shown in Figure 1. This will form first two layer of the physical design stack which are active device layer and the first M0 metal layer. Routing of signals in automatic cell based design flow is restricted to routing of IO signals of standard cells and macros and on chip memories.

Interconnecting functional blocks and macros cells in design layout use metal layers M1 to upper layers in layer stack of target technology. Interconnections are done by using metal layers stacked one above the other as stacked layers. Signals can be connected across metal layers as per the netlist through silicon vias. There are fabrication technologies which permit routing interconnections using 9–12 layers of metal stack.

The metal layers can be stacked in left to right direction or alternate horizontal and vertical layers. This is shown in Figure 2.



**Fig. 1** Base layer standard cell interconnections. (Source: Wikipedia)



**Fig. 2** Metal stack in layout

The routing signals involve interconnecting clock signals and data signals. Major design complexity in routing stage is to interconnect a large number of connections ranging in tens to thousands in number. This is impossible for designer to do manually. This requires sophisticated and reliable tool which understands design requirements, follows design rules supplied by the foundry, and achieves 100% signal

routing reasonable design time. The current EDA tools have router modules which can do this job, and this has become the industry standard way of chip design.

### ***Major Challenges in Signal Routing***

Signal routing is connecting signal interconnects which is different from power signal connection by routing. Power routing involves designing power ground rings, stripes, and rails. The interconnect width depends on the current carrying capacity of the wires in Rings, Stripes, and Rails. Power ground routing must be done for each standard cell, IO pads, and Macros in the design. Power Ground routing follows non default design rules (NDRs) as dealt in the earlier chapter. Clock signal routing involves synthesizing clock trees (CTS) or clock mesh which distributes clock signal along with buffers with correct drive strengths driving all the sequential cells of the SoC design. The goal of design of CTS goal is to ensure clock signal reaches clock input of every sequential cell at the same time or balanced time with minimum skew as dealt in Chapter 4. The *challenge of the signal routing* is connecting every IO signal of each of the standard cells, macros, and sequential cells to the connections as in SoC netlist. Scale of interconnections is the greater challenge of design. In the SoC design involving millions of gates hundreds of macros and several thousands of millions of signal connections within few square millimeters of silicon space, every signal must be connected and connected such that they carry signals with minimum delays and without any cross talk effects so that functionality is not broken. So, the challenge of Routing is identifying horizontal and vertical wiring patterns to route signals in SoC design with a fixed number of metal layers. Getting *100% connectivity* in the system design layout terminals of cells whose locations are fixed with minimum area and wirelength is the goal of the routing process. Constraints of routing the SoC design are fixed number of metal layers, design rules, limit on the signal delay and permittable cross talk between signal tracks. This is PD tool specific design activity. Sometimes, routers give up in between due to unavailability of routing resources such as channels, space for interconnects, and design element unreachable due to wrong placements or orientation. In such conditions, designer intervenes and commands the tool to complete the routing. Designer at this stage, changes placement of the related cells, or directs the router to use alternative channel or path for such signal route.

### **Design Readiness for Signal Routing**

No mistake is acceptable during physical design as it increases time to market and hence has serious business impact. Routing is done right first time. Checking the design inputs for correctness before initiating signal routing is very important. The checklists for design inputs at routing stage are:

- Make sure the design elements, functional blocks, IOs and macros in the design are placed correct. To check if there is enough space for routing around the design elements. Design elements are re-placed by adjusting its orientation such that signal routing of it is feasible. This is verified at design placement stage by virtually routing and checking correct routability of signals. The design log files from the placement process must be thoroughly checked to see any violation in design rules during placement and all are fixed.
- Power Ground signals are connected to all the design cells. The power routing is the most critical stage of placement. It is to be ensured that every cell of the design is correctly fed with the power and there is no violation, error, or warning in the log files of the design placement. The normal and non default (NDR) design rules are followed without any violation. The metal interconnect sizes have defined permissible sizes and length enough to connect the source and destination interface signals. Interconnects of VDD and Ground signals are sized correct for the specified current they carry to the design elements. This is to be ensured by even using NDR design rules.
- Clock tree is correctly synthesized connecting clock signal to all sequential cells in the design as per the design netlist. This is done by checking all the log files of CTS runs of the design. The timing reports are analyzed and confirmed the skew and latencies are balanced at all the leaf nodes.
- Design is verified for congestions of interconnect routes. Design congestions do not permit correct routing of signals. So, it is very important to check if there are enough routing channels in the design for signal routing.
- Static timing reports of the SoC design is checked for zero timing violations. STA analysis is a crucial process to take up design for signal routing. If there is any timing violations in placement and CTS stage in the design, signal routing will only enhance violations. It is necessary to fix any timing violations before proceeding to signal routing.

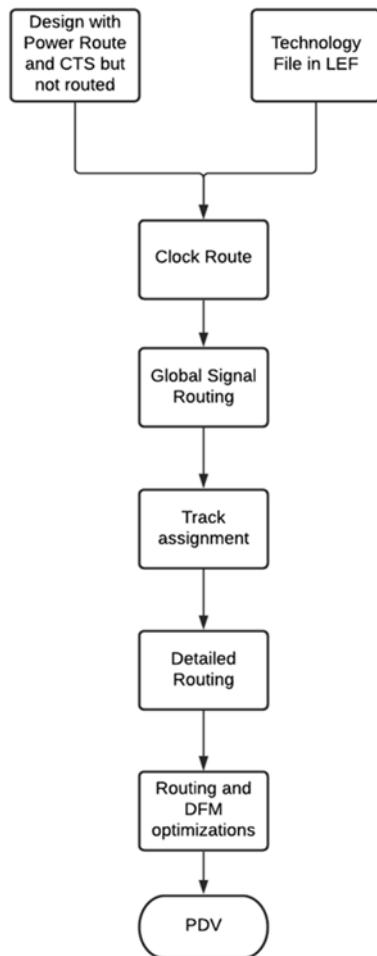
## Signal Routing Steps

Once the routability check on the SoC design is satisfactory, signal routing is initiated. The signal routing in the design flow is shown in Figure 3.

Signal routing begins with the clock routing for connecting clock nets for synthesized clock trees in the design.

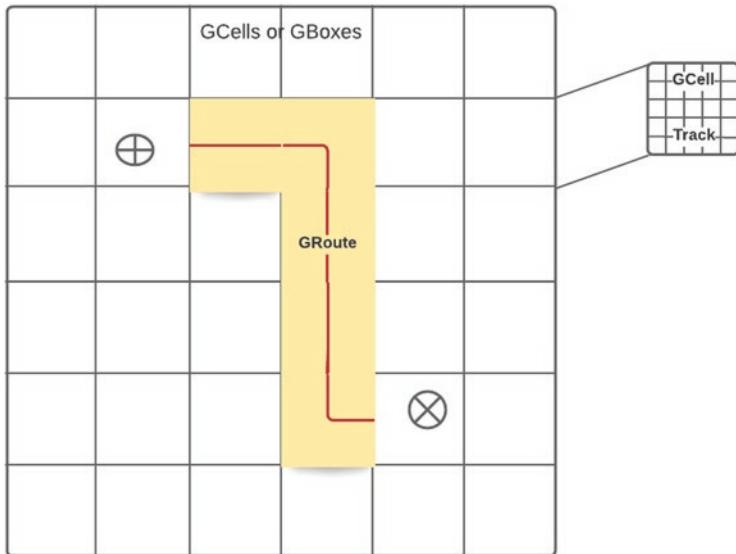
Next step in signal routing is global routing where routes are identified for signals considering congestions, design rules, and delay constraints. This is coarsely done. The design is divided into coarse global routing regions called Gboxes or GCells with hundreds of possible routing tracks in them. Correct Signal routes are identified by a sophisticated tool algorithm called Maze routing algorithm to find routes between GBoxes. Figure 4 shows the GBoxes with tracks and a global route for signal.

**Fig. 3** Design Routing Flow



Next step in the signal routing is identifying signal track for the signal nets in the GCells. This is done based on design rules and signal delays. Track assignment consider horizontal or vertical tracks depending on the metal layer capability for each design partition. The global routes are then replaced with metal routes. This stage of design routing result in many DRC violations, Signal integrity, and Timing violations. Horizontal and vertical metal routes are defined in the technology library. Typically, in any target technology, if metal layers are defined to support alternative horizontal and vertical routing. That is if metal 1 supports horizontal signal routing, metal 2 supports vertical routing of signals. This is to avoid electromagnetic issues for the reliable functioning of the design. This is discussed in further sections of this chapter.

After track assignment, *detail routing* is done for signal nets in the design. In detailed routing stage, actual metal wires are drawn using colored pallets in the

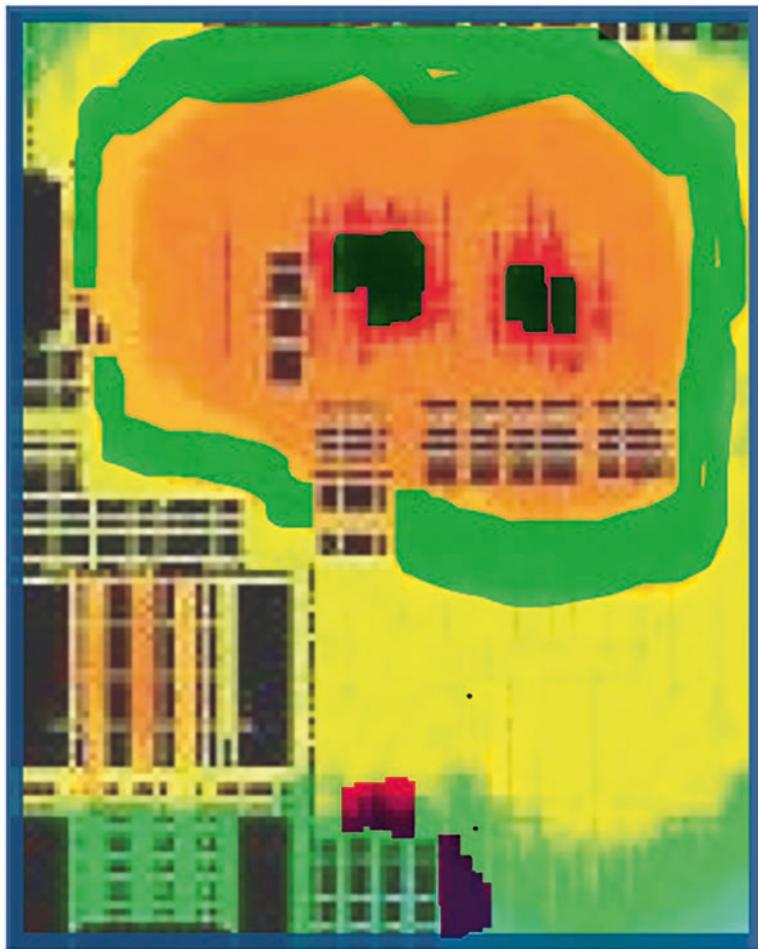


**Fig. 4** GBoxes, Tracks, and global route

tracks to connect the different signals, nets to pins. Detail routing stage uses appropriate design rules from the technology library wherever required to achieve design goals set in the design constraint. Detailed router follows the same pathways and tracks identified in global routing stage. It divides each of the GCell into smaller squares called *switch boxes* or *SCells*. Detail routing algorithm is same as global router algorithm but uses assigned tracks and Groutes from global routing stage. When a detailed route results in violation, it disconnects and reroutes the net in a way to automatically fix a violation. Algorithms in detail routing consider design rules, advanced phenomenon such as antenna effects, to optimize wire length and via count for connected net. It is an iterative tool oriented process. DRC is run and any violation is fixed inside every GCell and it is repeated iteratively. Routing optimization is done considering design constraints for specified delay performance.

Design is said to be routed completely if interconnections are done for all the signals as per the design netlist. The SoC design is now ready for physical design finishing and verification.

Design Routing is carried out virtually for assessing routing congestion early in the physical design stage during floor planning to ensure cleaner flow. Virtual signal routing is similar to actual signal routing but it is used to get the best floor plan. Figure 5 shows an example layout with routing congestion. The routing congestion results in *IR* drop issue in the region indicated by red color as shown in the layout. The congestion analysis using virtual signal routing is used to arrive at right floor plan of the design. Once the floor plan is accepted, the procedure for Placement, CTS, and Routing is repeated for actual Physical design. As described, getting optimum design is an iterative process going back and forth from floor plan, placement, CTS and Routing until specified quality of results(QOR) is obtained.



**Fig. 5** Routing congestion analysis

## Signal Routing Algorithms

As discussed earlier, the major challenge of routing is the scale of the problem. Routing problem involves routing the signal nets of nanoscale lengths of metal for thousands of millions of signals within a few square millimeter (mm) of die area. Completing interconnecting all signal nets in a liable manner within the specified design constraint itself is the main goal of signal routing. Only possible way is to automate it using the PD tool. Router modules of the physical design tools are used to route the design signals automatically. Here we discuss the algorithms used in routers of PD tool. The discussion is based majorly on Lecture 11 of “*Logic to layout*” by Rob Rutenbar [1]. The design die area is considered as number of small square grids of nanoscale size. The geometry rules of metals are complex at

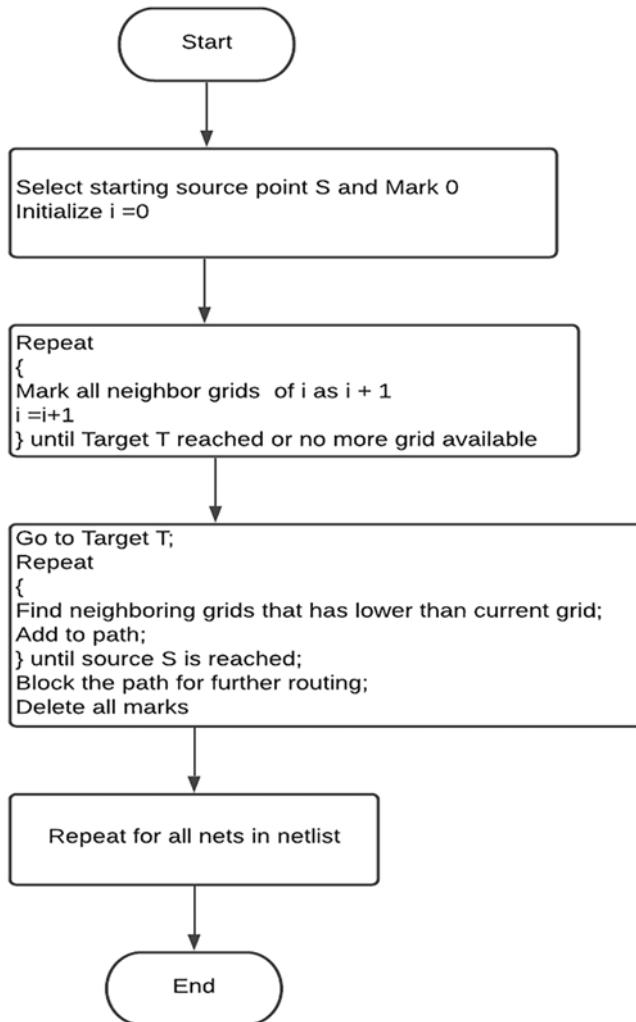
nanometer scale. The design layout consists of a number of different metal layers arranged one above the other used for signal routing. Inter layer connections are made using small holes filled with metal called vias. Apart from logistical complexity, routing interconnects with such structure at nanometer scale is electrically complex. Signal Routing is not just interconnections, but connections optimized for signal path delays. It is also necessary to address material properties effect such as cross talk effect. The problem definition of signal routing is finding a valid pattern of horizontal and vertical wires that connect the terminals of the signal nets in the design where design elements are placed( called placed design) using segments of fixed number of metal layers. The input for the routing function is the placed design netlist and cell locations in the die layout and the output is the geometric layouts of each net connecting standard cells and design elements with complete connectivity as in design netlist using optimum wire lengths and layout area. The design constraints are a number of metal layers, timing delay specification, and the design rules.

### ***Maze Router or Lee Router Algorithm***

For automatic Routing of the signal nets, optimized Maze router algorithm is used in the tools. Lee maze algorithm, Soukup Algorithm, Hadlock algorithm, Mikami Tabuchi algorithm, and Hightower algorithms are few examples of the maze algorithm. All these algorithms determine path between two points in the maze if there is one. But only the paths determined by Lee Maze algorithm and Hadlock algorithm find the guaranteed optimized path. Most of the VLSI routers use Lee algorithm [2]. The SoC design layout is assumed to be divided into many square grids and an optimized path between two connected points source point S and Target location T from the netlist is the optimized path between S and T which is determined by Lee's algorithm or modified Lee algorithm. The algorithm depends on *wave propagation* and *breadth first search*. The algorithm is three step process called *expand, back trace, and clean-up* to get the optimal route for the interconnect. The allowed direction of movement in identifying the path is Manhattan which is traced in horizontal and vertical directions. Trace directions are also referred as North-South and East-West. The flow chart of the algorithm is given in Figure 6.

As shown in the flow chart, during *expansion*, the grid in which the source net S is located is initialized to zero. The neighboring grids are marked with number one more than the current grid starting from source grid. The process is repeated till the target grid with T is reached. The resultant layout with marked grids is shown in Figure 7. All the paths with same numbers are called *wavefront*. Target is approached through these wavefronts.

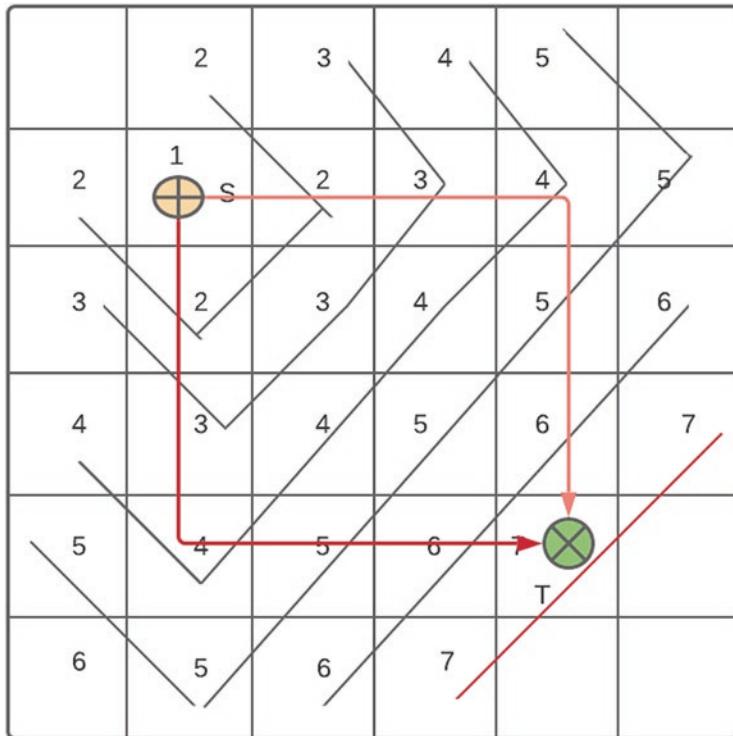
The next step is *back trace* wherein, from the target grid the paths are retraced by adding the numbers and storing the grids traced back till source grid is reached. These paths are the probable shortest paths between the source and Target. This is shown in Figure 8.



**Fig. 6** Breadth first Lee algorithms for maze routing

The third step is the *clean-up* which is the optimization stage. The shortest path between source and Target is the path with the smallest cumulative number. If multiple optimal paths are found as in this case, the grids are marked with weighted numbers in the expansion phase depending on preferences specified by the user and the path with the least cumulative weights is chosen as the route.

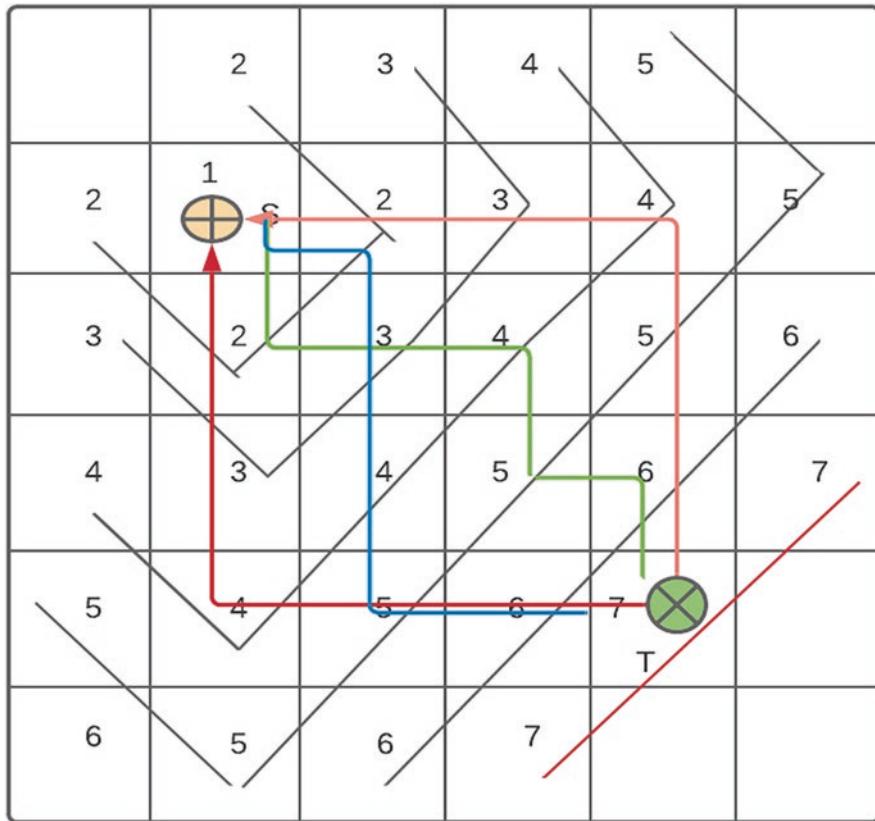
To improve the speed of execution of this algorithm, wavefronts are generated from both target and source grids simultaneously during wave expansion until the two wavefronts intersect. The back trace and clean-up procedure is the same as original algorithm. This makes the execution faster than the original maze algorithm.



**Fig. 7** Maze routing algorithm: Expansion for wavefront formation

## Routers in Physical Design Tools

Maze routers with modified Lee algorithms are used in physical design tools. In SoC design layout there are blockages specified by the user where routing is not to be done. Such blockages are avoided in the algorithm. Also, the routes must go on multiple layers of metal through *vias*. Such paths in multiple layers are also found out by means of weighted grid markings in the layout. Vias in layout are considered as routing options only when routes are not found on the same layers. Paths through vias spreading in multiple layers are found by using algorithms with weighted grid-based wavefront formation. In this algorithm, the grids reached through vias are numbered ten times the previous number and the path with smallest cumulative weights of the back trace is the shortest path. In SoC design consisting of thousands of millions of nets to be traced, one can imagine the computational complexity of these algorithms. Hence, routing process takes hours to days to route a complex design even on high-end computing systems.



**Fig. 8** Maze routing algorithm: Back trace

Routers in physical design tools have the following capabilities based on user preferences:

- Restricted use of metal layers for signal routing
- Restricted direction of routing in a metal layer
- Controlling off-grid routing of nets
- Preventing routing of selected signals
- Restricting routing in some defined regions
- Prioritizing routing regions
- Restricting scope of rerouting
- Restricted pin routing
- Setting stringent spacing and route sizes
- Reserving spaces for top level routing
- Controlling routing density
- Following non default routing rules
- Assigning shielding specific rules

Physical design guidelines consider technology and user preferences to give best quality results for SoC design.

## Design Signal Routing for Good Timing Performance

Quality of results (QOR) is the term used to assess the extent to which the design meets PPA goals. The Router is configured to meet the required QOR for the design. The SoC design is optimized for any one of the QOR as primary goal. Primary goal can be timing performance or signal integrity and Power performance. Rest of the QOR goals will also be met during physical design. The PD tool is configured to enable analyser and optimizer algorithms according to the selected mode. The necessary design information from the project database is imported for signal routing and optimization. For example, if signal Routing is to be done for timing as a critical goal, the configuration is set such that net delays are computed first and then the router uses the delay information to route and optimize the signal nets. The tool computes the net delays from the library information and previous executions, present in the design directory for design optimization. If the design timing file does not exist in the design database, it executes virtual routing to generate the delay information and uses it to route and optimise the design. This technique is called *timing-driven global routing*. Depending on the designer's confidence level, PD tool configuration is chosen.

Timing-driven routing aims to minimize the following:

- Maximum sink delay: delay from the source to any sink in a net
- Total wirelength: routed length of the net

For a signal route  $n1$ , let

$s_0$  be the source node

sinks =  $\{s_1, \dots, s_n\}$  be the sinks

$G = (V, E)$  be a corresponding weighted graph where:

$V = \{v_0, v_1, \dots, v_n\}$  represents the source and sink nodes of *net*, and

the weight of an edge  $e(v_i, v_j) \in E$  represents the routing cost between  $v_i$  and  $v_j$

The advanced spanning tree *algorithm is used for signal routing in PD tools*. Let  $\text{RouteLength}(T)$  be the length of the longest source-sink path in  $T$ , and  $\text{cost}(T)$  be the total weights of edges of  $T$ , the *routing for timing optimization* is a trade-off between *shallow Tree and Light tree*. “Shallow” trees have minimum  $\text{RouteLength}$  with shortest-paths tree constructed by Dijkstra’s Algorithm and “Light” trees have minimum cost with Minimum spanning tree (MST) constructed by Prim’s Algorithm.

## Signal Routing for Signal Integrity in Designs

Global routing of nets is also done to meet the DRC goals of the SoC design. The Cross talk, *IR* issues, and electromagnetic issues are of serious concerns of SoC designs. *Cross talk* is the effect of the neighboring signal nets when the signal on a particular net toggles fast. The effect could be a change in logic level, signal rise/fall

times, and frequency of the signal. The signal which effects the neighboring net is called *aggressor net* and the signal which gets affected is called *victim net*. Cross talk results in unpredictable behavior resulting in reliability issues of the design including fatal failures. This must be considered and corrected during physical design only. The techniques to avoid cross talk affects are spacing the signals apart so that they do not affect each other, shielding the nets which carried high-frequency signals, increasing the net width so that cross talk effect is negligible.

*IR* issue is caused when the nets carrying signals especially power signals are long enough and has narrow tracks. The signal strength Vdd reduces due to the resistance of long tracks and hence by the time they reach the destination nets, the signal amplitude drops considerably resulting in functional failures. Major causes of *IR* issues in SoC design are the following:

- Improper placement of power and ground pads.
- Smaller than required width of nets to carry Vdd and Gnd signals.
- Insufficient width of core ring, power straps, and rails.
- Small number of power straps.
- Disconnects in the power signal nets due to missing vias.

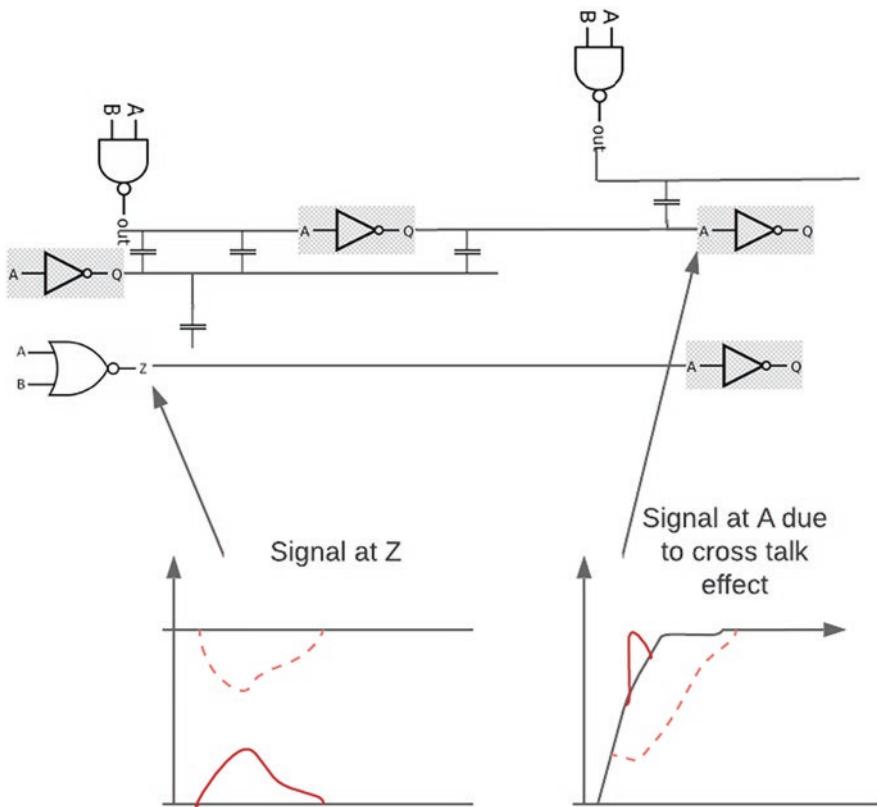
*IR* issue is addressed by checking the above-listed probable root causes and fixing them. It is necessary to set the safe route lengths, correct vias, widths of power ground rings, straps, rails, and nets for power ground signal tracks during physical design.

Electromagnetic issue is a long-term process and can result in failures after SoC working for many years. This arises when a continuously large current flows in one direction displacing the metal ions causing physical damage to the tracks, pins, pads, and ports. Design rules like track signal and signal direction are to be considered to avoid EM issues in long run. Cross talk effects manifest into two types of failures. Functional noise refers to noise that occurs on a victim net which is being held quiet by a driver. Cross talk noise on such a victim causes a glitch (Figure 9) which may propagate to a dynamic node or a latch, changing the circuit state and causing a functional failure [3, 4].

*IR*, crosstalk, and EM issues are to be addressed by design during physical design stage of SoC designs. Some of the techniques used to avoid cross talk effect are:

1. Controlling the length of nets running in parallel.
2. Shielding the critical signals if they are running in parallel to long nets or close to highly switching signals.
3. Maintaining safe inter nets distances.
4. Controlling the slew rate values and adding buffers in paths where the slew rate violations are found early in design.
5. Adding buffers globally as a strategy will to a greater extent avoid cross talk effect.

All the above are set in the design rules during routing stage. Different metal layers have different track resistances. Typically, the upper metal layers have lesser resistivity than the lower metal layers. When the design is routed with signal integrity as basis of routing, interconnections are carried out in compliance with these

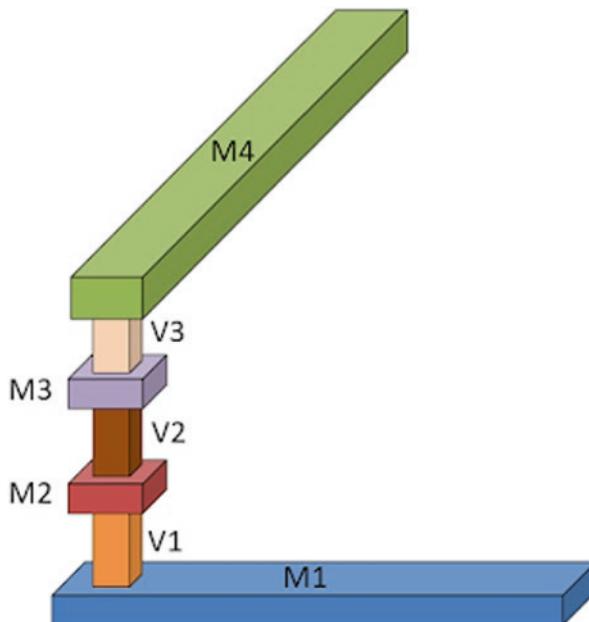


**Fig. 9** Functional and timing effect of cross talk

design rules. The router is configured to comply with specific design rules by setting blockage areas, restricting the use of layers, reserving layers for routing certain critical signals, defining custom rules with increased interconnect spacing, etc.

Advanced signal auto Router sometimes uses the concept of via *ladders* to avoid certain metal layers and the signal routing is done in a particular pattern. Via ladder is a stack of vias used for connecting target net in one layer and source nets in different layer. Via ladder is shown in Figure 10.

All the techniques discussed in this section also avoid EM issues and fix them by design. To exclusively invoke this, Router engines in physical design EDA tools are configured to route the signals with signal integrity as a criterion for routing optimization. By default, the design rules do consider avoiding these issues even in timing as a critical factor for routing optimization, for critical blocks where the SI issues are foreseen by designer, special rules are used, and routing is carried out with SI as a main factor for optimization. But understanding signal integrity issues of the SoC design and deciding routing options are in the hands of a good designer.



**Fig. 10** Stacked vias or via ladder

## Design Challenges in Signal Routing in SoC Designs

Practical challenges faced by the designers in the stage of signal routing of system designs are the following:

1. Routing process is a compute intensive process requiring machines of large computational power due to routing of many billions of signals. Many times, the process stops due to the lack of processing resource when algorithm is searching for the optimal net from large number of routing options in a congested part of the design and trying to optimize the design for multiple corner timing and signal integrity. The processing power needed to route the signal differs in such a scenario of the chip design compared to routing a simple non-congested block of the chip. This calls for a flexibility adding and removing the processing elements or CPUs dynamically as needed by the process. This is addressed in recent times by an option called *elastic CPU or scalable CPUs* by which one can add or remove CPUs dynamically during run times. User sets the option of available processor cores for the routing run and algorithm automatically uses them based on need.
2. Signal routing with the multiple optimization goals is another challenge. Design is optimized for timing in multiple design corners, consider signal integrity issues, comply with other design constraints and DRC rules. Some of the techniques used to address this are setting correct effort level in finding correct signal route, depending on the design complexity. This is done by defining correct

constraints. Signal routing complexity is addressed by routing of critical blocks separately, and specifying the important design goal whether is signal integrity or timing or area. Accordingly, tool options are set. Signal routing is highly compute intensive process. Use of scalable CPU platform, cloud computing are other options designers use for signal routing in designs.

## References

1. Logic to Layout, Lecture 11 by Rob Rutenbar, 2013.
2. An algorithm for path connections and its applications, C.Y Lee, IRE Trans. Electronic Computer, EC-10, 1961
3. S. Alwar, D. Blaauw, A. Dasgupta, A. Grinshpon, R. Levy, C. Oh, B. Orshaw, S. Sirichotiyakul, and V. Zolotov, "Clarinet: A Noise Analysis Tool for Deep Submicron Design," in Proceedings of Design Automation Conference DAC, June 2000, pp. 233-238.
4. K. L. Shepard and V. Narayanan, "Noise in Deep submicron Digital Design," in Proceedings of the IEEE International Conference on Computer Aided Design, November 1996, pp. 524-531.

# System on Chip Design Finishing and Design for Manufacturability DFM



## Reliability and Design for Manufacturability (DFM) of SoC

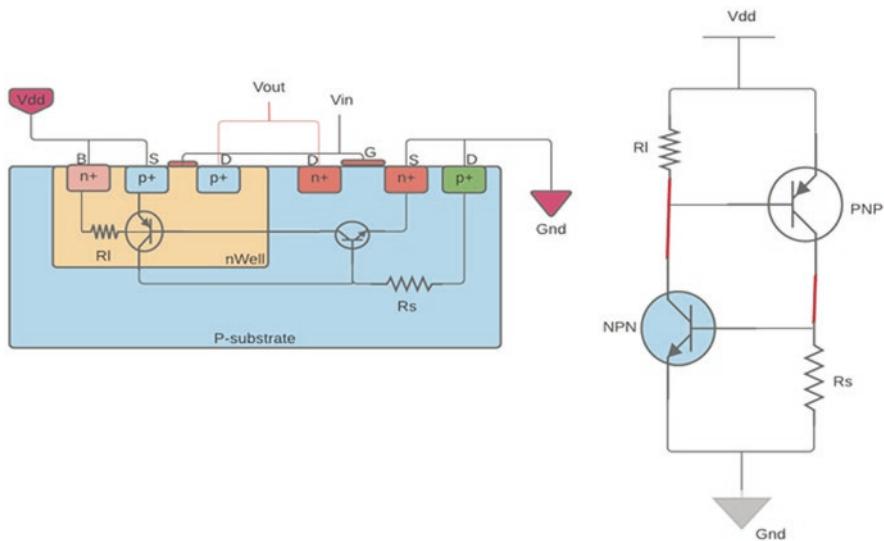
For reliable functioning of system on chips design as intended, it is essential to understand and address the device fabrication issues, exploit material properties that ensures device reliability. Device processing challenges are considered during physical design of SoC. The issues which are fixed during physical design are:

- Signal latch-up
- Antenna effect
- *IR* drop issue
- Electromigration effects
- Fabrication issues leading to signal integrity and reliability issues

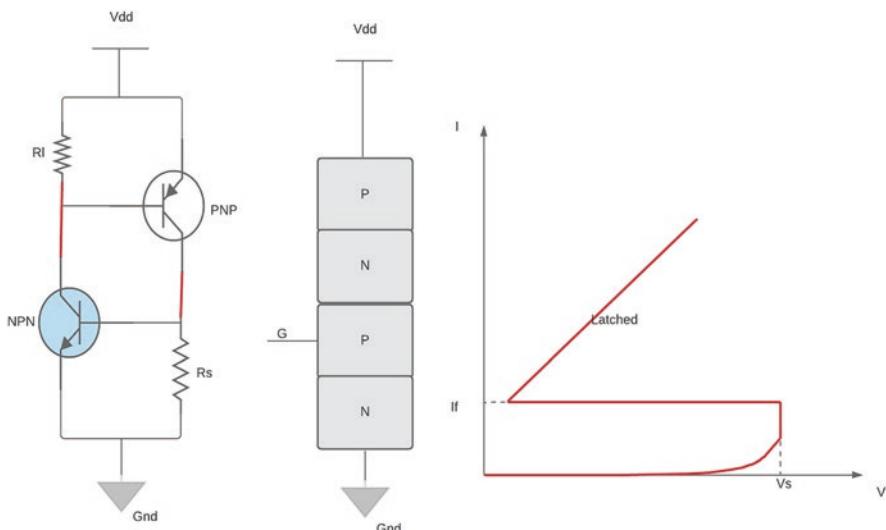
To address the above technological problems, some special cells are inserted in the design which are not part of functional design elements. Design process structures are adjusted considering manufacturing special rules which avoid failures while device fabrication process or to an extent in long term. Advanced design rules are part of Design Rule Check (DRC) rule deck.

### ***Signal Latch-Up in CMOS Standard Cells***

Latch-up in standard cell is a condition which results in large leakage current from power supply ( $V_{DD}$ ) to Ground leading to circuit failures. This condition arises when the signals at input-output have voltage values greater than supply voltage fed to the circuit. By structure of standard cell, there is a direct path from  $V_{DD}$  to Ground (Gnd) terminals as shown in Figure 1. This path is due to an unintended PNPN junction



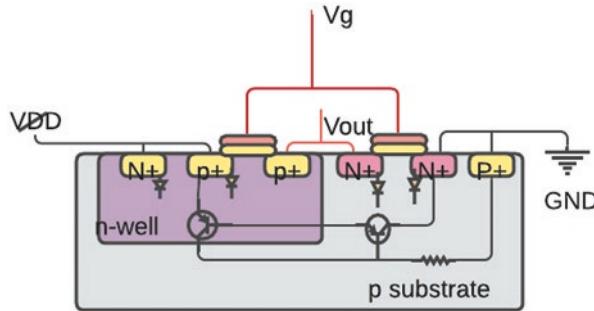
**Fig. 1** PNPN structure in CMOS Circuit



**Fig. 2** Parasitic circuit latch-up characteristics

device or SCR (Silicon Controlled Rectifier) formed by CMOS layer structure of the cell.

Standard circuit of CMOS cell has a pMOS device is in the nWell and a nMOS device in the p-substrate. Parasitic PNPN circuit is constituted by PNP and NPN BJT transistors which are formed by the structure as in Figure 2.



**Fig. 3** Latch-up circuit caused due to the junctions formed because of the structure

A PNPN device is *off* by default with no current flowing in it. PNPN device turns on when the input signal voltage is larger than power supply  $V_{DD}$  at gate terminal with a large current flowing through it. Signal at gate terminal can be higher than power supply voltage due to voltage fluctuation or noise in the signal path, Electrostatic discharge, or radiation effects. This turns a PNPN device on. Once turned on, the current from power to ground continues to flow even if the gate signal is removed. Figure 2 shows the terminals of the PNPN device and its characteristics.

This is called a latch-up in the standard cell where the parasitic BJT circuit turns *On* and gives a low impedance path to the current to flow from power supply and ground terminals as shown in Figure 3. These large current heats up the SoC causing permanent damage.

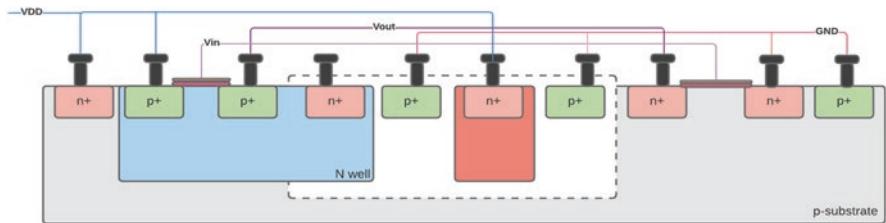
When output voltage rises beyond the  $V_{DD}$  due to noise, the junction PNP device in nWell turns ON which further turns on the device in p+ region. With this, current from the drain of pMOS starts flowing to the body of nMOS causing voltage drop between the source terminal of nMOS to the substrate which will forward bias the PN junction between substrate and source of nMOS, which starts conducting. This continues causing both the BJTs to turn on permanently causing latch-up which is self-sustaining. In latch-up current flows from  $V_{DD}$  to GND causing localized heating which will eventually result in chip failure.

Following are some popular design techniques used to prevent latch-up in chip design.

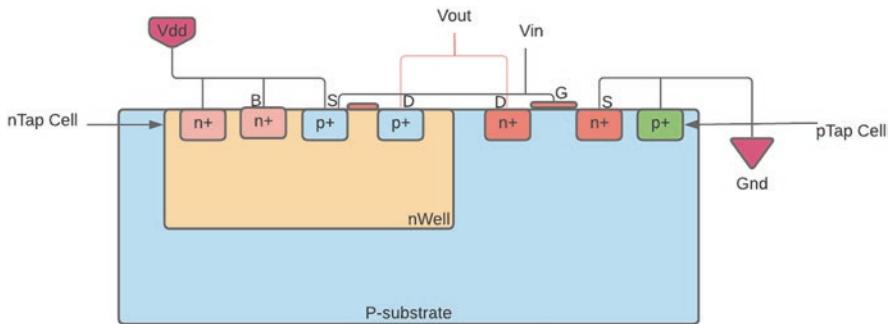
- Adding Guard ring by additional implants
- Adding well tap cells
- Creating isolation trench around the device structures

### Guard Ring

The short circuit path developed due to parasitic BJTs formed are cut by adding extra p+ implant in nWell and n+ implant in the pSubstrate. This prevents the current flow from drain on nMOS to the body of pMOS. This technique is called



**Fig. 4** Guard ring



**Fig. 5** Tap cells in CMOS standard cell

adding guard ring in between pMOS and nMOS transistors in CMOS standard cells circuits to prevent latch-up. Guard ring in a design is shown in Figure 4.

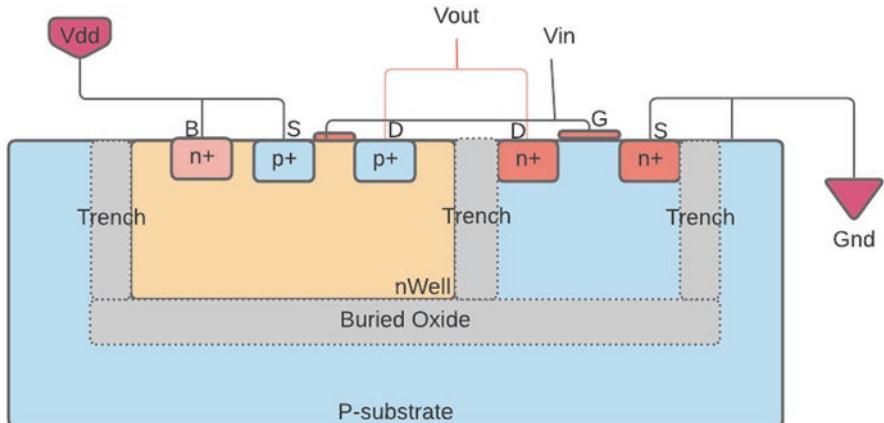
### Well Tap Cells

Another technique to avoid latch-up in CMOS is to connect the  $V_{DD}$  signal to nWell and Gnd signal to p-substrate. This is done using special cells called *well tap* cells. *nWell tap cell* taps the *nWell* to  $V_{DD}$  and *pWell tap cell* p-sub to  $V_{SS}$  or Gnd as in Figure 5.

The tap cells are part of standard cell library. They are placed in the standard cell rows during physical design at the regular spacing as per the design rules of the technology library.

### Creating Oxide Isolation Trenches

Oxide trench is buried between nMOS and pMOS transistors in standard cell circuit to break the latch-up path. The oxide trenches isolate the MOS transistors and avoid formation of the parasitic PNPN device. A cross-section of oxide trench isolation is shown in Figure 6.



**Fig. 6** Buried oxide trench

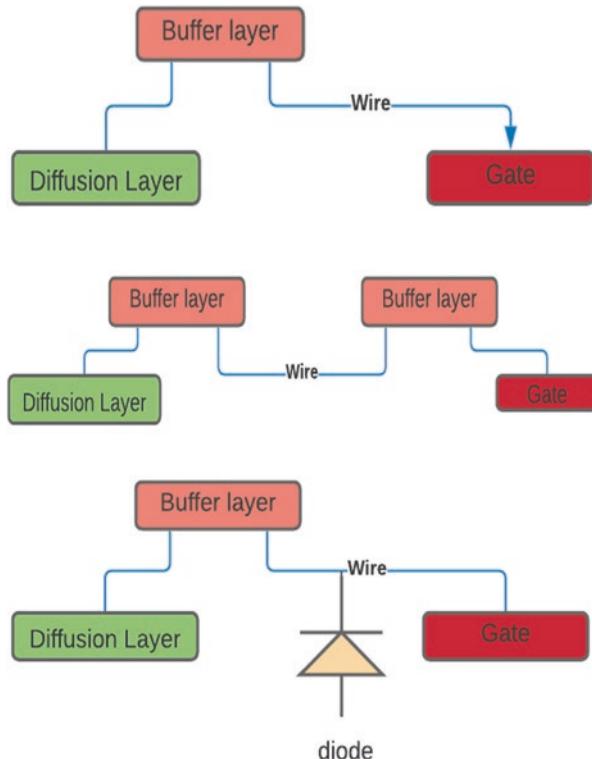
Apart from these techniques which are used in physical design process, special processed wafers like *epitaxial wafers*, *retrograde wafers* are used to avoid latch up in designs. These wafers are processed with controlled well doping profiles using advanced technologies. Another most used technology is Silicon on Insulator (SOI) technology, which is the preferred over CMOS technology for the avoidance of latch-up.

### ***Antenna Effect***

During CMOS device fabrication processes such as plasma etching, there is a chance that large amount of charge gets accumulated in the gate region of the transistor if there is extra metal connected. During these processes, there is a possibility of gate oxide getting easily damaged by electrostatic discharge. The static charge collected during multilayer metallization process can damage the device leading to chip failure. The charge accumulated is conducted by the diodes formed in source and drain diffusion regions, which will change transistor behavior. This is called *charge collecting antenna effect* or *antenna problem* in VLSI chips. Antenna effect transistors in the following ways:

- Reduces threshold voltage.
- Changes the *IV* characteristics.
- Reduces life expectancy of the SoC.
- Increases gate leakage.

The changed electrical behavior of transistors due to antenna effect can considerably cause functional deterioration of SoC and many times results in functional device failures. Antenna effect can be avoided by defining the maximum size



**Fig. 7** Techniques to avoid antenna effect

of metal which can be connected to gate. The design rules for antenna fix indicate maximum area of metal that can be connected to a gate. Depending on the break-through voltage of the transistor, maximum ratio of the area of interconnect metal wire to the area of gate vary from 100:1 to 5000:1. Thicker the gate of the transistor, higher the ratio. There are many design techniques to nullify an antenna effect. One is by splitting the metal interconnect wire connected to the gate into two segments that are connected to each other by a buffer layer. This is done practically by defining the antenna rules for metal layers during design. The second way is to connect a *reverse-biased diode* to the long wire connected to gate. This is called diode protection mode. Both the methods are shown in Figure 7.

## Cross Talk

Electromagnetic cross talk is another effect in SoC designs of nanometer CMOS technologies. Increased cell densities and increased interconnects in complex SoCs are subjected to parasitic effects with electromagnetic cross talk. The lateral capacitance between the adjacent nets act like a mutual capacitance that affects the signal

integrity. Crosstalk is effect in which signal transition in one net creates an undesired effect on the signal carried by the neighboring nets. The effects can be the change in signal amplitude or timing. A net that effects the signal carried by the neighboring net is called the *aggressor net* and a net in which signal is affected is called *victim net*. This leads to functional or timing failure in the SoC devices.

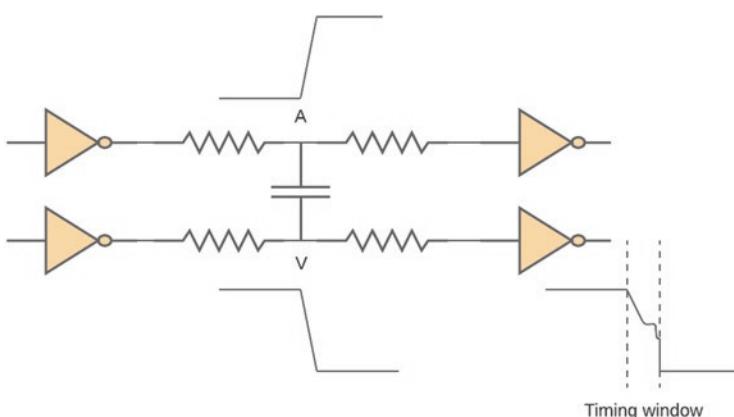
### Cross Talk Timing Window Analysis

Cross talk effect is intermittent. SoC design is analysed for Crosstalk effect during static timing analysis. In the timing analysis window under consideration, an aggressor influences a victim net. Analysis by window method of crosstalk analysis is more accurate and optimistic compared to the infinite arrival window that assumes an aggressor switches any time. Figure 8 shows the switching window of a victim net. If a switching window of an aggressor net overlaps with that of a victim net, then only the delay of a victim net will get affected else it is ignored.

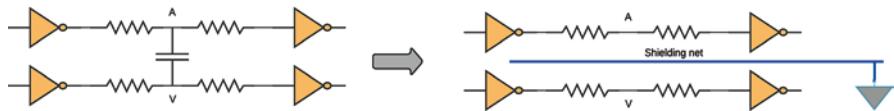
Effect of a cross talk can be either increase the design path delay or decrease in delay. Timing widow concept is also applicable in the cross talk noise analysis. Analysis is carried out considering the effect of multiple aggressors on a victim net. Cross talk effects the functionality by changing the signal delay or generating signal *glitch*. Both can cause function failure which sometimes can be fatal. Cross talk delay can affect the *timing* of the design. It could unbalance a balanced clock tree, and could result into the setup and hold timing violations.

### Cross Talk Prevention Techniques

There are various design techniques used to prevent the cross talk. Some well-known techniques followed to avoid crosstalk are as follows:



**Fig. 8** Timing window analysis



**Fig. 9** Shielding victim net

1. Increasing the spacing between crosstalk suspected signal net and the neighboring nets. If the spacing between aggressor and victim net is increased, the coupling capacitance reduces, reducing the cross talk.

2. Shielding of nets:

Figure 9 shows the shielding technique used to prevent cross talk. Generally, we insert a shielding net between the victim and the aggressor net. The shielding net is connected to  $V_{DD}$  or  $V_{SS}$  or Gnd

Shielding net avoids cross talk and keeps the signal nets in the neighborhood silent as it is connected to the ground. This however increases design area.

3. Upsizing a victim cell:

Another technique to avoid cross talk is to replace the victim cell with a cell of higher drive strength so that cross talk effect by the aggressor is ignorable.

4. Downsize the aggressor cell:

Alternatively, an aggressor cell is downsized by replacing it with the cell of lower drive strength but however it must be assessed for connected fanout. This reduces any effect of causing cross talk to neighboring signal nets.

## IR Drop

Metals like copper and Aluminum are considered ideal for interconnections of different design elements in CMOS VLSI technologies. However, these metals despite being good conductors have small resistance in them. Since metals are used for contacts and interconnects in fabricating CMOS devices, it is necessary to know the resistivity of the metal layers in layout stack. Typical resistivity of layer stack for 90 nm CMOS technology is shown in Table 1. *IR* drop is more pronounced in nanometer CMOS technologies where the interconnect lengths are comparable to device sizes. *Resistivity* which is *resistance per unit square* is the parameter used to define the property of a metal. When the length of interconnect is large while routing source net to destination net, the signal amplitude at the source may drop considerably when it reaches the destination due to metal resistance. This changes the signal voltage in the net carrying it at the destination input point of the cell. *IR* problem is more relevant for Power signal routes. *IR* problem in advanced technology nodes is more critical resulting in chip failures like functional and timing failures. Power supply to all the standard cells, macros is distributed in SoC design by means of *power distribution network* (PDN)s formed by  $V_{dd}$  and  $G_{nd}$  Rings, Rails, and Stripes. These are interconnects wires using metal layers in design layout. Large current

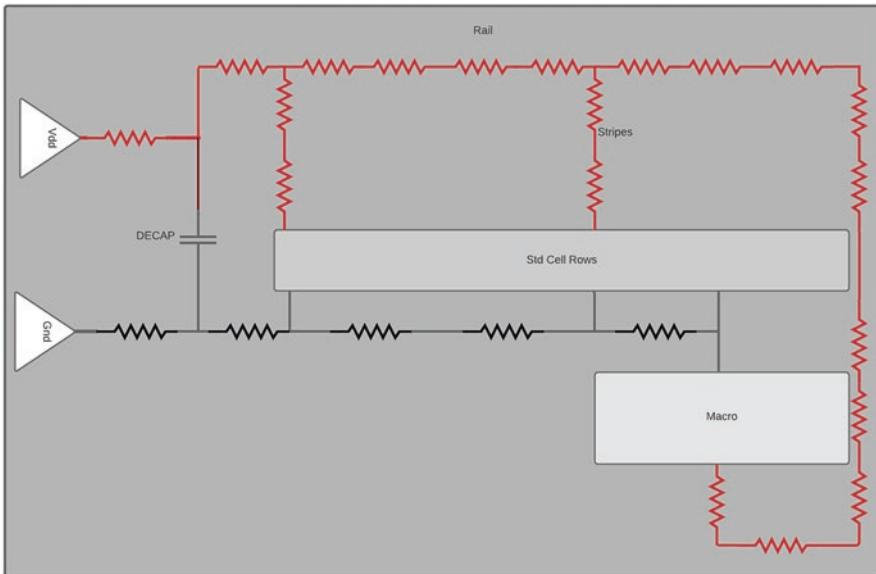
**Table 1** Layer wise resistivity in 90 nm technology

Layer	Resistivity in ohms per square unit
Metal	0.1
N and P plus contact	10–100
NWell	$1000 \pm 40\%$
Low Doped polysilicon	10K
Silicide poly resistor	$1-10 \pm 30\%$
Non silicide poly resistor	$50-1000K + 120\%$

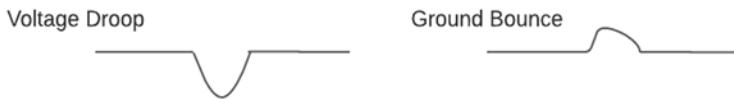
flowing in the metal interconnect of power net results in voltage drop due to the resistance of the net. A significant amount of voltage will be dropped in the PDN which will make less power supply voltage available for the standard cells. If  $V_{dd}$  is the supply voltage at power pin and voltage applied at the standard cell  $V_{sc}$  is given by  $V_{sc} = V_{dd} - IR$ , where  $I$  is the current flowing in the net and  $R$  is interconnect resistance. The voltage drop in the metal tracks of PDN is proportional to  $V_{drop} = IR$ . Figure 10 shows the voltage drop in one of the power nets of PDN.

The effect of  $IR$  drop is reduced power supply voltage actually applied to standard cell or Macros than intended voltage. Sometimes, the power supplied to the standard cells or macro will be less than the minimum operating voltage needed by them for the correct function. This will affect the performance of the chip by affecting setup/hold time of the input signals to these cells or functional failure. The  $IR$  drop can be *static* or *dynamic*.  $IR$  drop on the PDN net is static when signals to functional blocks/standard cells are static.  $IR$  drop on the PDN net is dynamic when the input-output signals to the standard cells/macros toggle. Signals toggle in the functional mode of the SoC. Hence, the dynamic  $IR$  drop is more than static  $IR$  drop. With an increase in signal toggling  $IR$  effect can result in *ground bounce* or *voltage droop*. Voltage droop is a small instantaneous decrease in the power supply voltage value. Ground bounce is a small instantaneous increase in voltage level on the ground net. Ground bounce and Voltage droop are shown in Figure 11. Both, when it happens beyond a certain threshold can result in cell not working correctly resulting in functional failures sometimes fatal to SOC.

Static  $IR$  drop reduces the power supply to the standard cells. Reduced power supply to the cell circuit increases the standard cell delay. Increase in cell delay affects the functionality or signal setup and hold times of signals in design paths. It can also introduce noise on power supply voltage on power nets. The  $IR$  effect can be seen in signal, power, or clock routes. Effect of power supply noise on Clock route is taken care of by the clock tree synthesis. There are many causes of  $IR$  problem in design layout. Some of the causes of IR problem are bad PDN network design, smaller metal width of power interconnects, larger spacing between the power stripes, inadequate *decap* cells on power supply ports or concentrated high density of cells in some parts of layout, higher signal switching in some parts of die area in the chip, occasional high current flow, and insufficient number of voltage sources.



**Fig. 10** *IR* Drop in power net of PDN



**Fig. 11** Voltage Droop and Ground bounce

All of these must be considered during the physical design stage by doing *IR* analysis. The design analysis for *IR* problem involve extraction of parasitics from the design layout and analysing the design timing by back annotating the extracted parasitics.

*IR* problem in physical design is addressed by many design techniques. Some of them are:

- It is to be noted that in deep submicron process technologies, the metals of higher layers have lesser resistance than the metal layers in lower layers of the physical design layer stack. One obvious way of addressing *IR* issue is by routing the affected interconnects using the higher layers of metal.
- *IR* issue is addressed by stacking metal layers through vias so that the resultant resistivity is reduced.
- *IR* issue is avoided by redefining the floor plan by placing the affected design elements closer to each other so that the interconnect length is reduced.

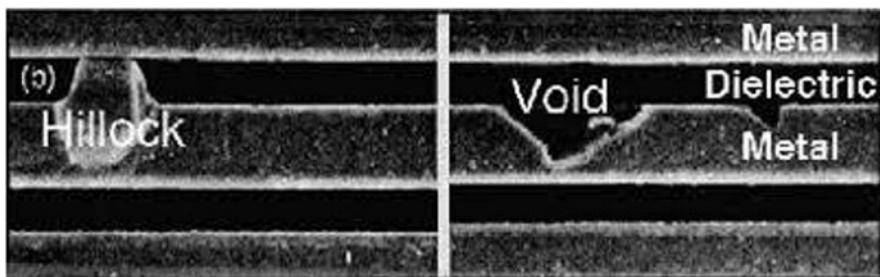
In areas where there is high density of cells, they are spread out in that region to decrease the *IR* effect. Proper PDN with increased width of metal, adding decoupling capacitors in power nets reduce effect of *IR* drop on power nets. The smaller the interconnect length, the smaller the resistance, and hence lesser the voltage drop in the interconnects.

### ***Electromigration (EM) Issue***

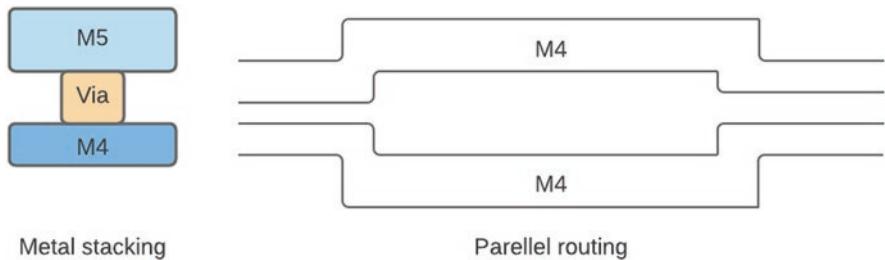
Electromigration is the gradual displacement of metal ions over time in the semiconductor device due to high current density in the direction of electron flow and ion flux. This depends on many factors like current carrying capacity of the metal, crystal size, force of electron-ion that tend to displace them which intern depends on current density, temperature, and mechanical stress encountered. When the high current due to some reason flows in the metal which has the current carrying capacity of  $i_m$ , the temperature in the metal net increases which increases the current. This continues till the metal ions get displaced from their original locations. This can cause *hillock* or *voids* in the metal layers. This will either cause open circuit or short circuit in the circuits resulting in circuit failures. Figure 12 shows the effect of EM effect in metal layer.

EM effect is taken care of during physical design by following techniques:

1. Increasing width of metal interconnect.
2. By using metal of higher layers as they have lower resistivity and high current carrying capacity.
3. Metal stacking with vias.
4. By parallel routing technique as shown in Figure 13, which increases the width of metal interconnect.



**Fig. 12** EM effect in metal interconnect



**Fig. 13** Methods to fix EM issues

### ***Fabrication Issues Leading to Signal Integrity and Reliability Issues***

In deep submicron processes, the system failure can be because of variabilities in processes and dynamic behaviors. Process variability and nonuniformities in the fabrication processes can cause functional, performance, and timing issues in the systems. Hence, it calls for well-designed layouts to guarantee good reliable functioning of the SoCs. This can cause serious performance issues for SoCs where the legacy IP cores targeted to different technology nodes are integrated during physical design. Major effects which fall in this category are *well proximity effect* and *stress-strain effect* which can be controlled effectively in layout design.

#### **Well Proximity Effect (WPE)**

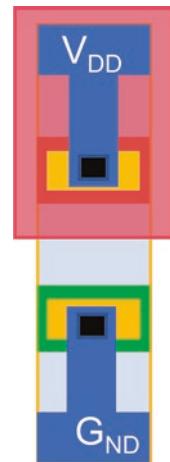
Transistors that are close to the edge of the well have different performance characteristics than ideally placed (centered in the well) transistors. This is due to the different threshold voltage  $V_t$  compared to normal transistors. This results in variation in propagation delay of these transistors to the extent of 10% [1].

#### **Stress/Strain Effect**

Process variation changes the mobility of the charge carriers of transistors. This changes on current of the transistor, which will further affect the speed of the transistors. The change in speed can be to an extent of  $\pm 30\%$  [1]. This variation is controlled by using design rules to compensate the process variations due to layout or structural irregularities.

The structural design rules are defined considering the issues described above. By complying to these design rules and by adding some special cells like well tap cells, well tie cells, boundary cells, corner cells, the effect on electrical characteristics are minimised.

**Fig. 14** Layout of well tap cell



## ***Tap Cells***

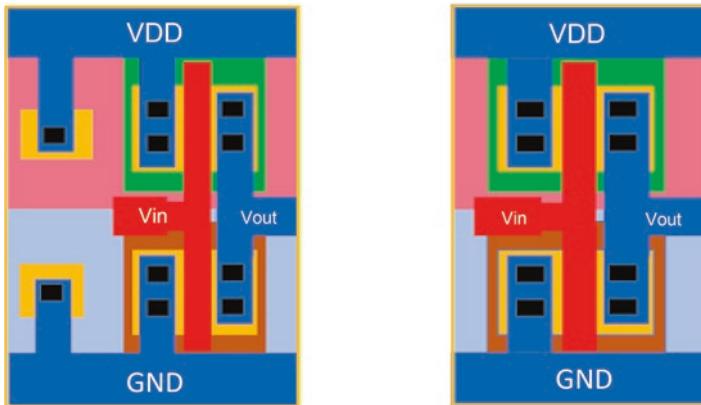
Well tap cells connect the nWell to VDD and p-substrate to VSS. Apart from tapping wells to the power points, these cells do not affect the functionality of the design. Therefore, well tap cell is also called a *physical-only* cell. There are two types of Well tap cells one to connect nWell to VDD and other to connect p-substrate to GND.

A structure of well tap layout of a cell is shown in Figure 14.

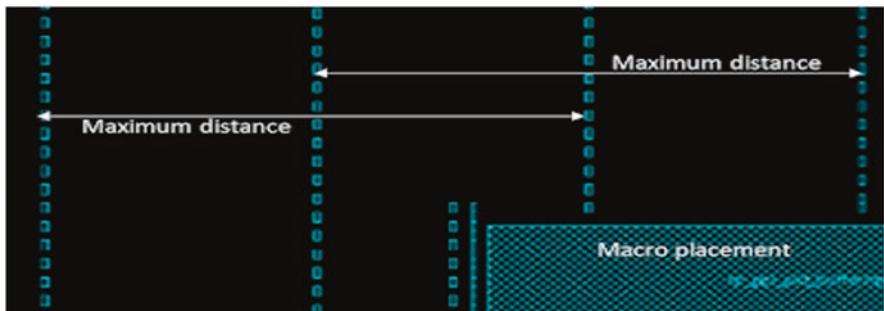
Well tap cells are part of tapless cell library available in advanced technology nodes. Tapless cell library is a library in which the standard cells does not have wells and substrates connected to the power and ground. This library is used to get area advantage as it gives designer to add tap cells where ever needed. Tap cells are added to a set of standard cells and not individually to achieve area optimization in physical design. Figure 15 shows both standard cell with and without tap cells.

Well tap cells are added at regular intervals in the standard cell rows. They are added after the standard cell and macro placement and power rail generation in placement step. The maximum distance between two well tap cells is defined in DRC rule set of that technology. A typical placement of well tap cells is shown in Figure 16.

Well tap cells are generally placed in a straight column in the alternate row as shown in Figure 16 and such a pattern is called *checkerboard pattern* to provide power tapping for wells and substrates of all standard cells. If a macro comes in the path of vertical columns, then the placement of vertical column is shifted adjacent to macro.



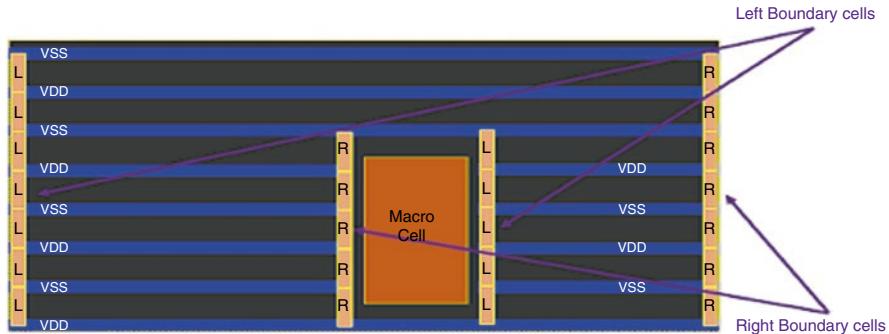
**Fig. 15** Traditional and Tapless standard cell structure



**Fig. 16** Well tap cell placement

### ***Inserting Boundary Cap/End Cap Cells***

Gates of the standard cells placed at the boundaries of the design layout generally get damaged during fabrication process. To prevent such damages of cells at the boundary, an additional cell called *end cap cell* or *boundary cell* is added to the standard cell rows at the boundary of design layout. Boundary cell not only protects the designs from gate damage at the boundary, but it also serves many other purposes like relaxed design rules for (DRC) for base cells, and an alignment of rows. Boundary cell is placed at both the ends of each row and at the top and bottom of the row. Placing the boundary cell at the end of the standard cell row terminates it. Placement of boundary cell at the top and bottom of the row enable easy integration to other blocks. Some standard cell library has *corner end cap cells* which are placed



**Fig. 17** Placement of end cap cell at the end of rows

at the corner of the blocks. Boundary cells have fixed shape attribute suitable for placement at the corners and boundaries. A typical placement of end cap cells at the end of the row is shown in Figure 17.

*Special end cap cells* which also serve as decap cell is found in some standard cell library. The boundary cell is a physical-only cell and has no logical functions. Boundary cells have Nwell, implant, poly layer, and metal rails as shown in Figure 18.

*Boundary cells* are placed just after the macro placement and standard cell row creation before standard cell placement during placement stage. It is also called a pre-placed cell.

### **TiHi and TiLo Cells**

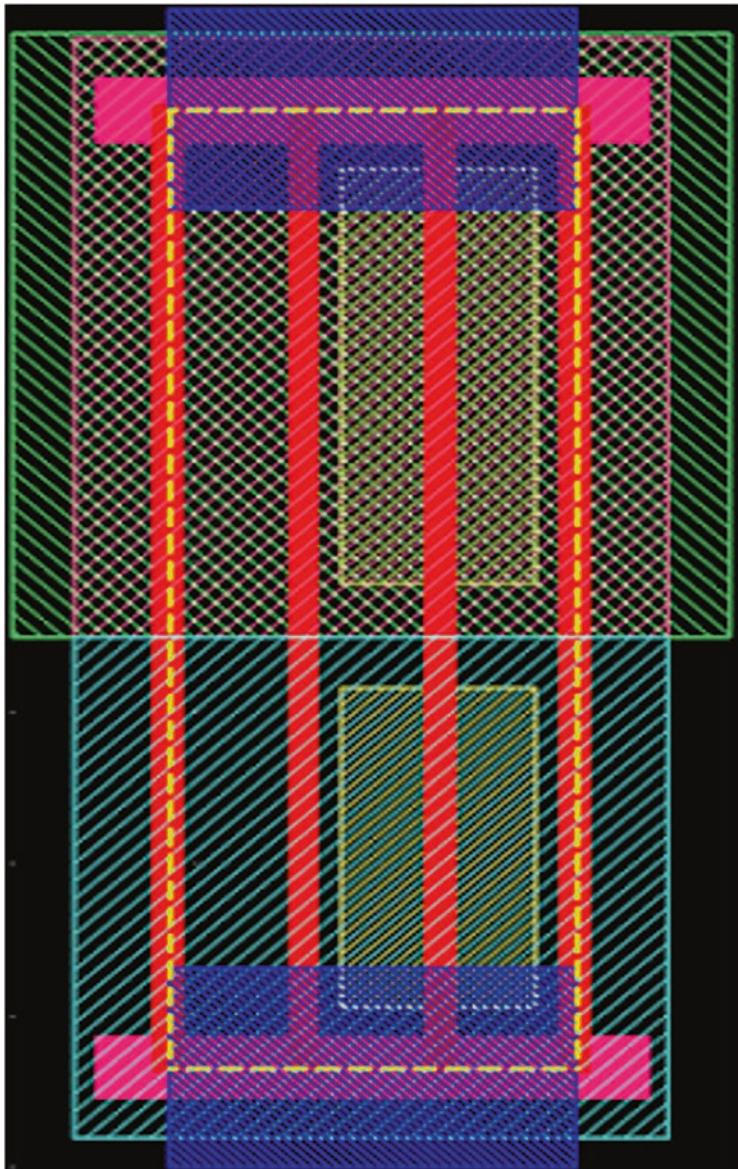
Generally, in designs, many times, constant values are stored in registers or flip flops. This require gates of the transistors in flip flop circuits to be driven logic high or low or VDD or GND respectively. The gates of the transistors in such standard cells are not directly connected to power signals VDD or VSS. This is to avoid any surge currents/glitch voltages on power lines damaging the gate oxide in transistors of the cells. Such cells are connected through special cells called TiHi and TiLo cells, available in the cell library. Two types of tie cells called *Tie-high cell* and *Tie-low cell* are available in cell library.

The tie cells have only one output pin to connect to VDD or VSS depending on the type. The schematic of tie-high cell and tie-low cell is shown in Figure 19.

In Tie-high cell, the drain and gate of nMOS transistor are connected so that the transistor operates in linear region and the pMOS transistor is biased in saturation.

$$\text{So } V_g = V_d \Rightarrow V_{gs} = V_{ds}$$

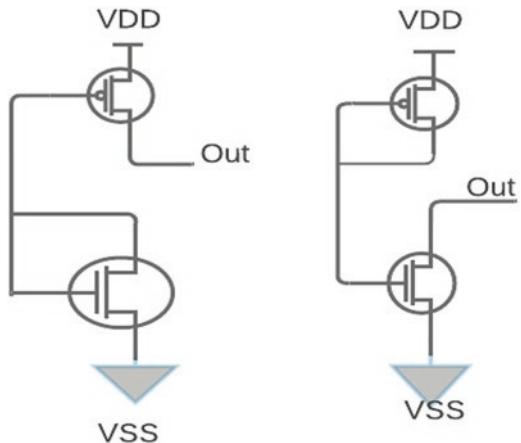
Therefore,  $V_{ds} > V_{gs} - V_t$  to keep nMOS in the saturation region. The configuration of MOS where drain and gate are shorted is also called diode-connected transistor. And when nMOS is behaving like a diode here, the gate of pMOS is



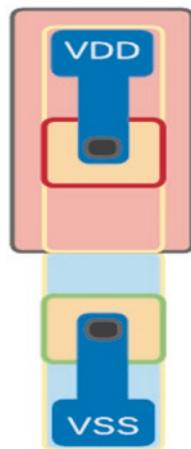
**Fig. 18** Dummy rails

always low and so pMOS is always in on state. When pMOS is in on state its drain which is output will always be high. Similarly, for the tie-low cell, the pMOS is always in saturation so the gate of nMOS is always high and hence the drain of nMOS will always be at the low logic. The sudden spike in VDD or VSS will be not propagated to the output of the tie cell. Figure 20 shows the layout of the cells.

**Fig. 19** Tie-high and Tie-Low cells schematic circuits



**Fig. 20** Tihi cell layout



### Decap Cells

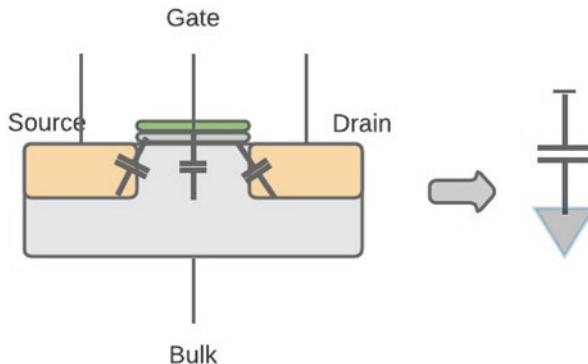
Decap cells are the charge storing capacitors and are used to supply the instant current in the power network. Sometimes in SoC design, circuits draw large current which causes power droop or ground bounce in power supply, which affect the delay of standard cells. Such sudden power requirements of the circuits are fed by *decap cells* which are inserted into the design.

Figure 21 shows the various capacitances inside the MOS transistor and if we connect the source, drain, and body terminal together then all these capacitances will be configured as a parallel capacitance and a single equivalent capacitance as shown. Figure 22 shows the layout of a simplest decap cell.

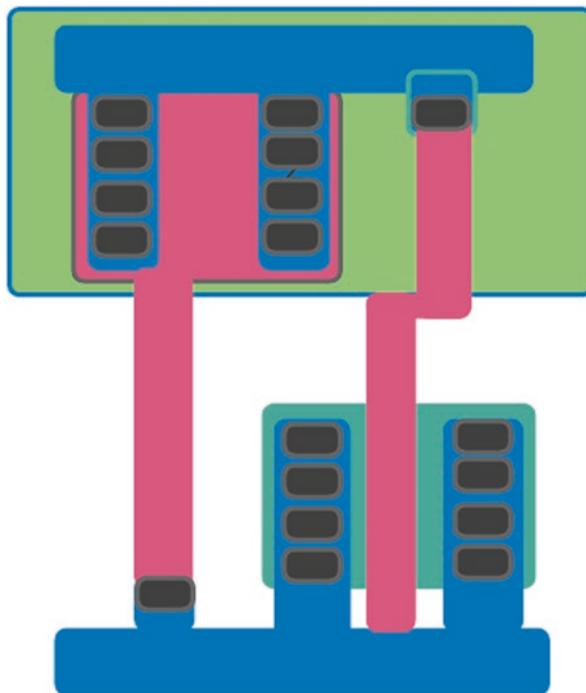
Source and drain of pMOS transistor shorted together and connected to VDD and the Gate is connected to VSS. Similarly, the source and drain of the nMOS transistor are connected to the VSS and gate is connected to VDD.

When CMOS gate is switching, there is a region of input transition where both the nMOS and pMOS transistors conduct together as shown in Figure 23a. This results into short circuit current  $I_{sc}$  flow from VDD to VSS for that instant. Simultaneous switching of such large number of cells causes large current as shown in Figure 23b. This large current requirement drops the VDD or increases the ground voltage which is called voltage droop or ground bounce as shown in Figure 23c.

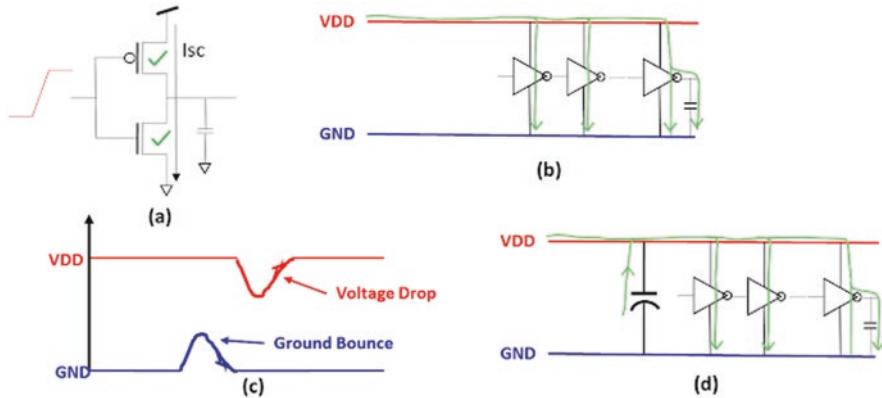
Voltage droop or ground bounce results in the change in the connected standard cells delay. Variation in supply voltage affect cell delay. Change in delay may further



**Fig. 21** MOS capacitors



**Fig. 22** Layout of decap cell

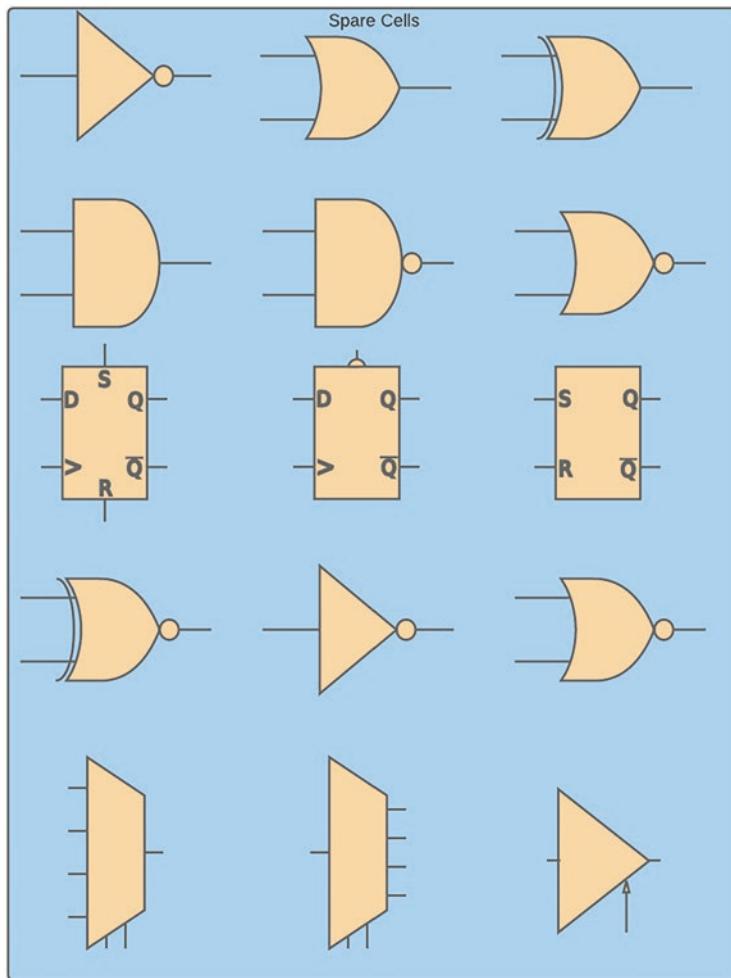


**Fig. 23** Instances requiring large current from Power supply in design causing Ground bounce and Voltage drops

affect the timing of design. If the supply voltage drop is high it affects the functionality of the standard cell. Decap cells supply this extra current needed avoiding ground bounce. Decap cells work as charge reservoirs and support the power delivery network making power network robust as shown in Figure 23d. Decap cells are added after the power planning and before the standard cell placement, that is, in the pre-placement stage. These cells are placed uniformly throughout the design in this stage. Decap cells can also be placed in the post route stage if required. Decap cells are leaky and increases the leakage power of design, so must be used judiciously.

### Spare Cells in Physical Design

If a functionality issue is found in a chip after fabrication, or some simple enhancement is required, which is feasible with few standard cells, then this is implemented using *spare cells*. Spare cells enable us to modify/improve the functionality of a chip with minimal changes in the mask. We can use already placed spare cells from the nearby location and just need to modify the metal interconnect. There is no need to make any changes in the base layers. Using metal ECO we can modify the interconnect metal connection and make use of spare cells. We only need to change some metal mask, not the base layer masks. Spare cells are a group of basic standard cells consisting of inverter, buffer, nand, nor, and, or, xor, mux, flip-flops, and maybe some specially designed configurable spare cells. Spare cells do not perform any logical operation in the design and act as a filler cell only. A group of spare cells is shown in Figure 24.



**Fig. 24** Spare cells

The inputs of spare cells are tied either VDD or VSS through the tie cell and the output is left floating. Input signals of the gates are left floating as a floating input is prone to be affected by noise resulting in unnecessary switching which leads to extra power dissipation. Spare cells are added either in the netlist or by PnR tool. These cells are added before the placement of standard cells throughout the design.

## References

1. <https://sage-da.com/analyze.html>

# Physical Design Verification of SoC



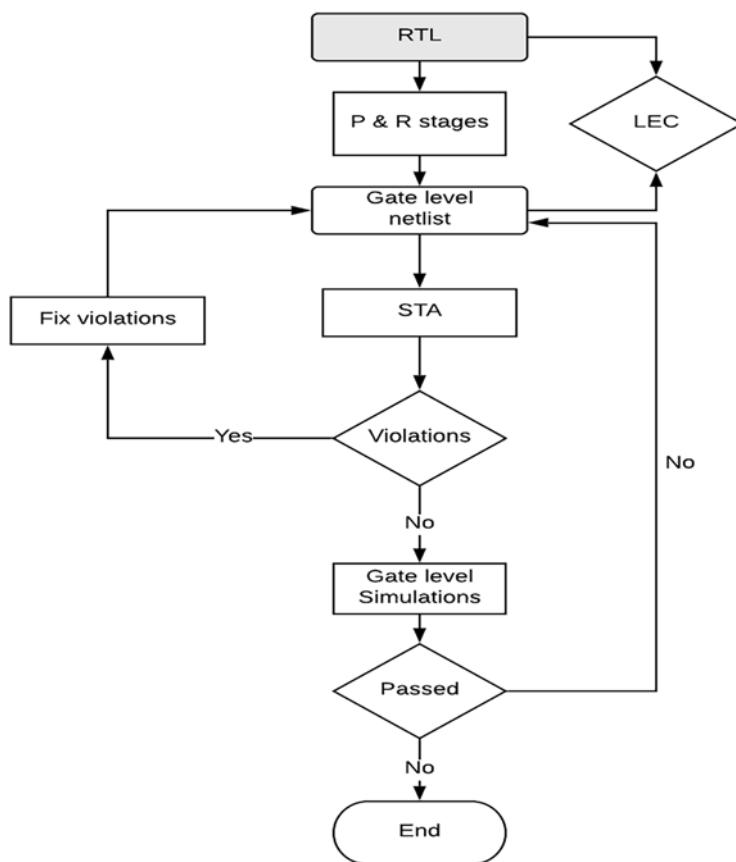
## Physical Design Verification

As SoC design approaches tape out deadline, every physical design engineer involved will be in a mindset to do everything which is needed to ensure the design works as intended when it is fabricated. This is finally confirmed to a great extent by the physical design verification process. *Physical design verification* is carried out at many intermediate stages of physical design apart from ECO and final design sign off stage. Design verification methods are Static timing analysis, verification involving Design rule check (DRC), Logic equivalence check (LEC)/Layout vs Schematic (LVS), and Static current (*IR*) drop analysis. Apart from the intended functionality, every design transformation processes in physical design like placement, CTS, clock/power routing, and signal routing must be done right for the SoC design to work first time. This is verified for correctness at every stage of design transformation and is an iterative process. Any design issue found in this stage must be fixed and even if it is an incremental design change, complete physical design verification is carried out to ensure that nothing is broken. This activity is carried out by identified physical design verification engineers and finally by sign off engineers.

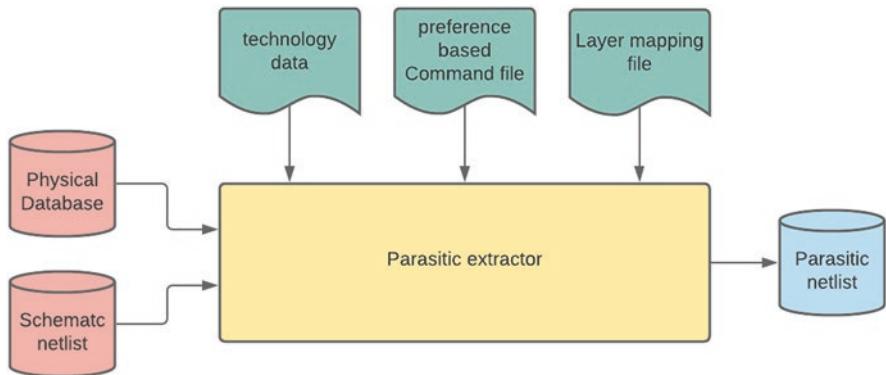
### ***Static Timing Analysis (STA)***

STA is carried out whenever the design netlist is changed in terms of cells or interconnects. This is done after placement, CTS, Routing, Engineering change order (ECO) implementation, and during sign off. Apart from basic timing analysis, it is good to analyze the design for skew, pulse width, duty cycle, and latency. STA is carried out at the block level of changed design or at the design top level of hierarchy. Analysis is carried out using STA tool which can consider all the relevant

design inputs and derive timing values and write the reports with the signal delay values for every timing paths in the SoC design. Design netlist, Physical library, constraint file, delay database, logic library, and extracted timing models are input to a STA tool. STA tool writes out updated design constraint file and violation reports. These output files are used again for design optimization most design tools automatically perform. Most timing violations are fixed automatically by the tools or by providing Electronic change order (ECO) to the tool for design changes. Basically, design violations are fixed by adjusting the signal path or clock path timing either by up/downsizing the cells or by inserting/removing the buffer cell in gate level netlist paths. Once the timing fixes are rolled in, STA is carried out again to confirm that there are no violations in the design. Once all violations are fixed, the path delays in standard delay format (SDF) is written out. The design with parasitics are written out by STA tool to use it in the design simulation in gate level hierarchy for dynamic timing analysis. The gate level simulations are run to confirm basic functionality of the design considering its actual design path timings. The STA and gate level timing verification flow is shown in Figure 1.



**Fig. 1** STA and Gate level simulation during PDV



**Fig. 2** Parasitic extraction flow

The quality of STA is dependent on how accurately the layout parasitics are modeled and extracted. Parasitic extraction must be accurate on the design netlist generated by a PD tool. Parasitic extraction tool is used at any stage of physical design to extract accurate parasitics. The parasitics written out in this stage is used further for static and dynamic timing analysis. Parasitic extraction tool uses input files which contain design layout netlist, Layer mapping file, and the parasitic extraction command file.

Figure 2 shows how parasitic extractions are carried out during physical design verification.

Parasitic netlist is used further by analysis tools during static timing analysis, gate level simulation.

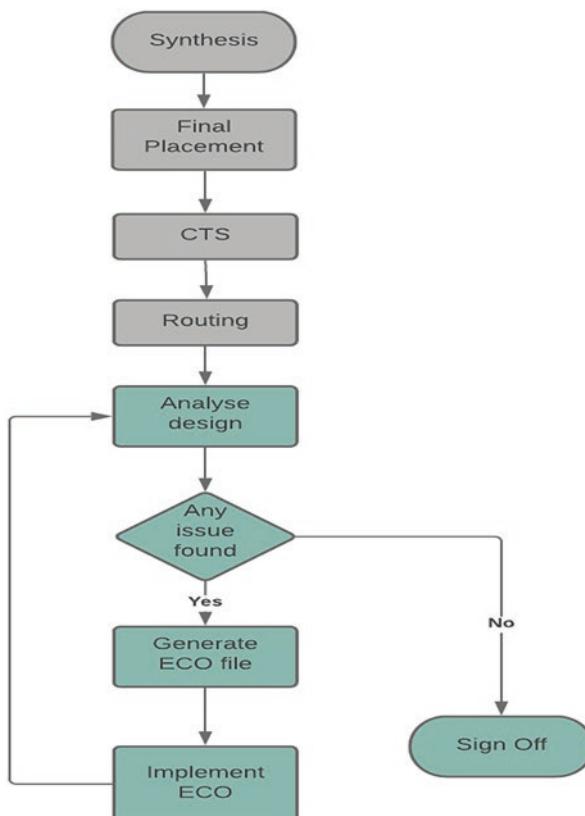
## Design Sign Off

There are many sign offs like physical sign off, timing sign off, and *IR* sign off which are done before layout of the design with other relevant design data files are sent for fabrication. The design changes at this stage are critical and is called engineering change order (ECO). ECOs are last stage fixes for any of the issue found both functional, timing, or reliability of the SoC design are implemented as engineering change order (ECO).

## ECO for Design Fix

ECO is the process of implementing design corrections for issues found in last stage sign off checks done in physical design. These fixes are for timing, DRC, and *IR* issues. ECO is small design changes implemented very carefully in the design

without compromising on the already working and verified part of design. In this phase, all physical design implementation activities are completed, and most of the design issues are resolved. Before entering the ECO phase of the design, design timing goals are met, DRC are passed, and good level of confidence is attained for fixing a small number of pending design issues which are small design change but critical to the design functionality. Design changes for a open issue are documented in an ECO file and incrementally implemented in the design netlist. Almost all the design checks for physical design are passed and the possible small design change to fix any new issue found are recorded in the ECO file. ECO file contains a series of commands of physical design tool corresponding to the changes in the design needed for correcting the issue. Sign off tools support the generation of ECO file when design change is fed in. If the change in design as ECO it is acceptable fix for lately found issue, it is directly implemented in the final design netlist. Typically, these are minor changes like disconnecting the identified a signal net and rerouting it with the identified correct net or resizing the standard cell or addition or removal of the buffer cells in the design path. ECOs are simple design changes but can be as



**Fig. 3** ECO Flow

many as thousands in number for lately found design issues in the final stage of the design. Hence, some PD tools support implementing ECOs in batch mode. Some of the Physical design tools also support implementing ECOs in a dedicated ECO design flow with special commands. This makes sure that only identified corrections are incorporated without disturbing the rest of the finished design.

There are very advanced design check and analysis tools which help us in uncovering any issue in this flow. The ECO file is thoroughly reviewed for the possible side effects and if found good, the design changes are implemented in the design. This is implemented one by one until all ECOs are rolled in the design. The updated design database is finally verified for correctness of the issues. When all the ECOs are implemented, final design database is written out by the tool. The ECO flow is shown in Figure 3.

## EDA Tools for Sign Off

There are many sophisticated EDA tools for physical design verification and design sign off. These tools can analyze design database thoroughly and if any issue is found at later stage of design process, the tool writes out probable fix as ECO file. Physical design is analyzed for detailed timing, *IR* issues, and physical implementations like ERC and DRC considering the ECO. Depending on the analysis type, specific sign off tool is used for analysis and fix. An ECO flow using a particular tool is certified and accepted by the fabrication foundry.

## Physical Design Verification by Formal Methods

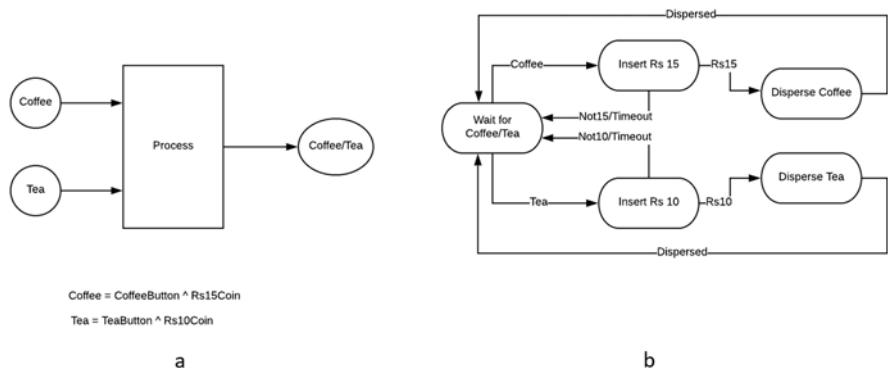
Physical design flow involves transformation of the design. There is change in SoC design netlist at every step of physical design which need to be verified for functionality and timing. But it is very difficult to simulate the gate level design because of high execution time, cumbersome debug mechanism. At the same time, it is absolutely needed to confirm that the design intent is preserved during the physical design. This objective is achieved by formal verification methods. Formal verification methods are model checking and equivalence checking. Verification by formal methods complements simulation based verification during physical design very well.

### *Model Checking*

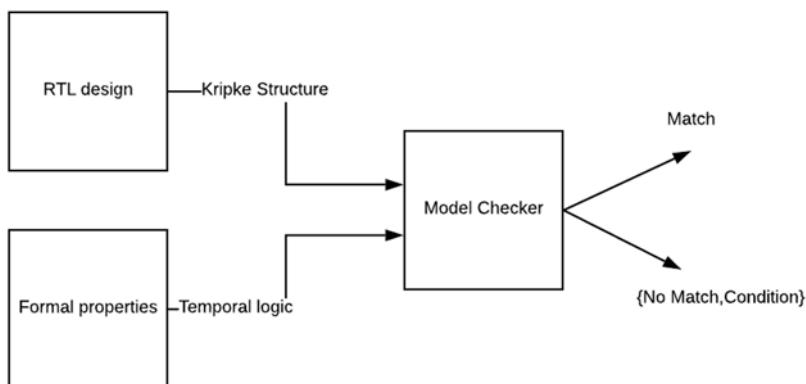
Design modeling is a process of identifying the system properties and representing them as a set of mathematical equations and verifying the conformance. For example, to verify a coffee/tea vending machine, it is required to model its properties or

specifications. The block diagram of coffee/Tea vending machine is shown in Figure 4a. The vending machine disperses coffee when coffee button is pressed and Rs15 is inserted, and tea is dispersed when tea button is pressed and Rs10 inserted in the coin slot. The vending machine system is mathematically represented, and the state diagram of the vending system is given in Figure 4b.

The functional requirement of vending system is represented by formal properties. The actual design and the formal properties are read in to the model checker for comparison to check the functional equivalence as shown in Figure 5. The design is extracted into Kripke structure, and the properties are represented into temporal structure which are input to the Model Checker, and both are compared for equivalence. A Kripke structure is a variation of the transition system, originally proposed by Saul Kripke, to represent the behavior of a design. It is a graph whose nodes represent the states of the system and whose edges represent state transitions.



**Fig. 4** Coffee/Tea vending Machine, State diagram and formulae representing formal properties



**Fig. 5** Model checking

## ***Logic Equivalence Checking (LEC)***

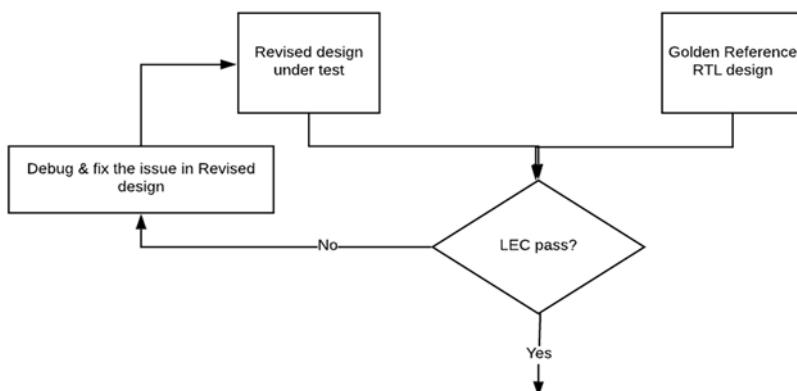
Logic equivalence checking (LEC) involves a comparison of synthesized design netlist and extracted design netlist from physical design stages (placed design, design after CTS, design layout after routing) for logical equivalence. The concept is shown in Figure 6.

Design comparison for logic equivalence is carried out by establishing correspondence and deriving equivalence. The step-by-step process is shown in Figure 7.

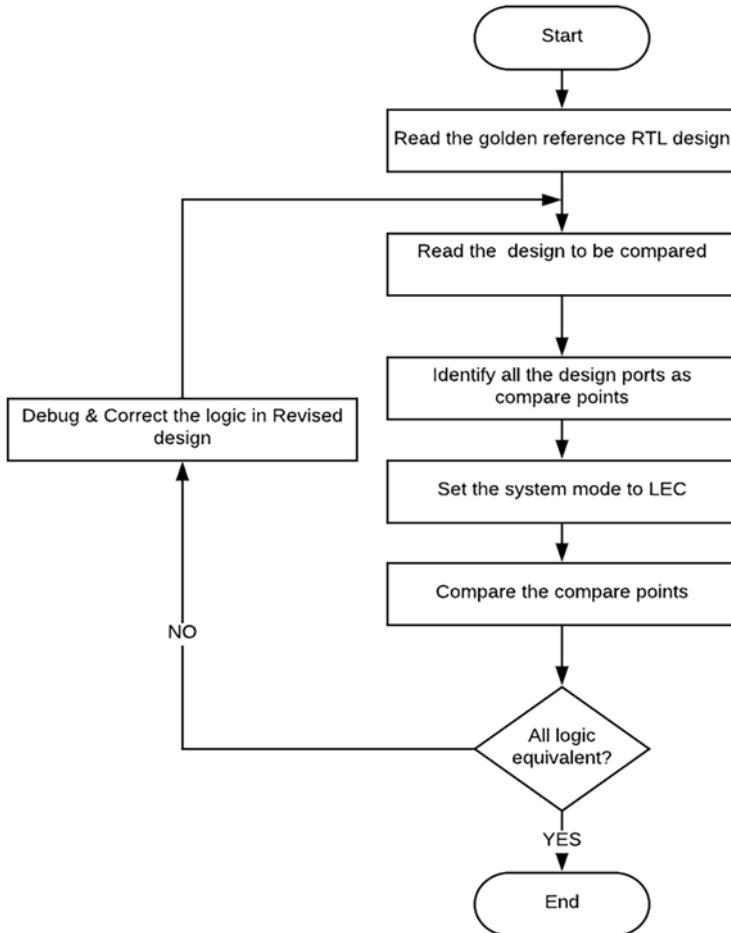
Logic equivalence check (LEC) of the design is done using a LEC tool. The formal tool typically has a debug environment where the nonequivalent points are highlighted in the design. The user interface exactly shows the design logic which is non equivalent to the reference design logic such that user can trace the logic cone and fix the logic for achieving equivalence. LEC tool is be guided by the designer by mapping the compare points manually with some naming conventions in a tool configuration runscript. LEC is run on synthesized netlist vs. placed netlist, synthesized netlist vs. routed netlist.

## ***Layout Versus Schematic (LVS)***

LVS is one of the traditional sign off techniques for the SoC design. Physical layout is checked for retaining the functionality of the design during PD processes. Design netlist is extracted from the design layout by identifying logic elements and interconnections in metal layers, base layer structures. Extracted design netlist in SPICE format is verified by checking for opens, shorts, and overlapping base cells. Design extraction is very critical in this stage of design. The extraction tool extracts the layer structures (polygons) to determine components like transistors, diodes, capacitors, resistors, and their connectivity information by identifying the layers of



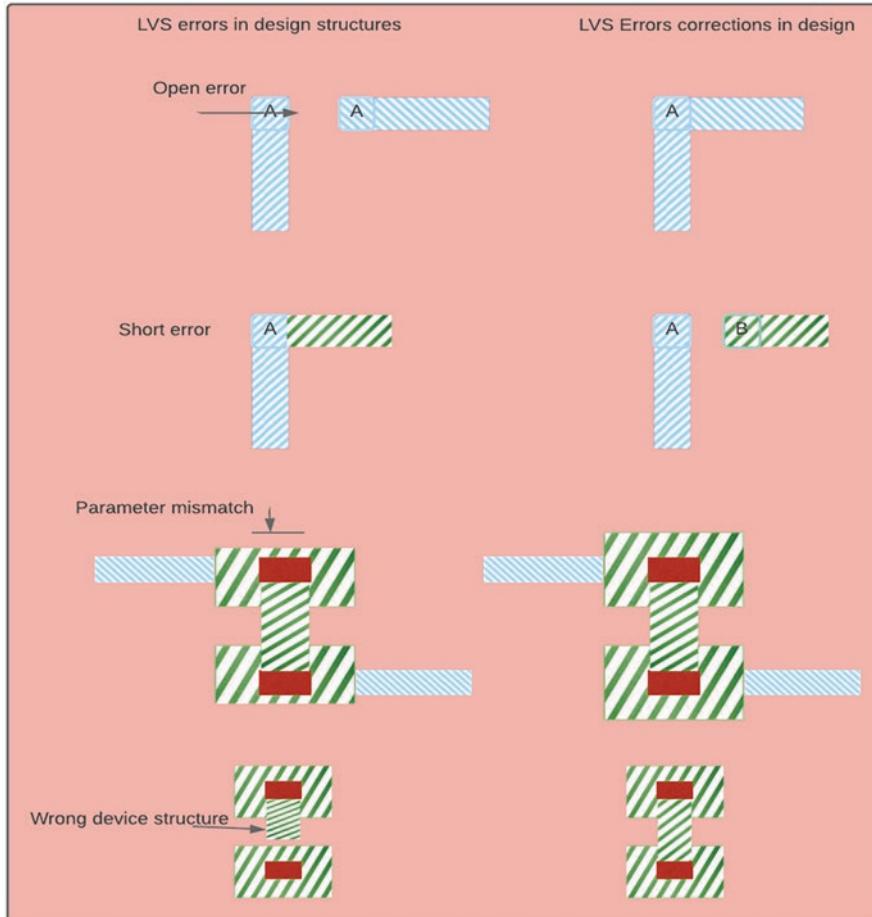
**Fig. 6** Logic equivalence



**Fig. 7** Logic equivalence check flow

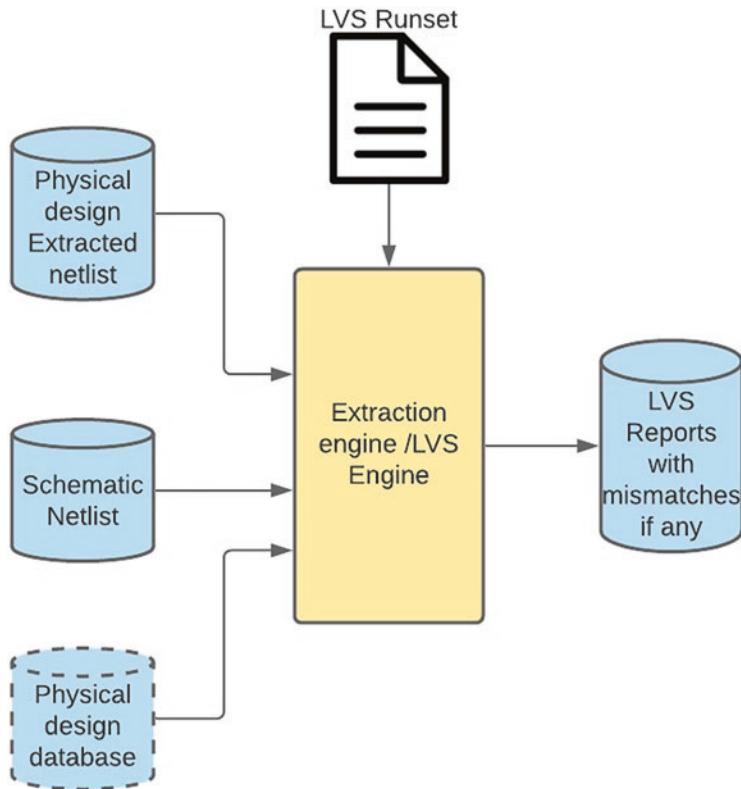
constructions. Device layers, terminals of the devices, size of devices, nets, vias, and the pin locations are defined and will be identified by the unique number.

Finally, the layout netlist is compared with a reference netlist of the design used as input to PD for preserving the design functions. LVS uses a rule deck to extract design netlist from design layout. In this stage the number of instances, nets, and ports are compared. All the mismatches such as opens, shorts, and pin mismatches are written out in the LVS report. The two netlists are compared for number of devices, their correspondence, interconnected ports, other circuit topologies and device sizes. Errors in LVS check are mismatches in number of devices, opens and shorts, additional nets found in the layout netlist, mismatch in components, and transistor sizes and so on. Some of the LVS errors are open net, short net errors, mismatch in device parametric values, device mismatches, and their corrections are shown in Figure 8.



**Fig. 8** LVS errors in design and their corrections

There are dedicated LVS checking tools that are used for design sign off. The difference between inbuilt LVS module integrated in P and R tool and the stand-alone is that the latter uses all expanded details of physical structures of the design, but P and R tools use abstract layout structures. The set of commands are written as LVS rule set, design netlist which is input to the LVS checking tool, layer assignments, physical database checks like SNAP, and GRID checks. Schematic netlist provides complete cell level information along with nets. LVS flow is shown in Figure 9.



**Fig. 9** LVS Flow

### ***Electromigration***

Aluminium and Copper is used for Interconnection inside the chip. Aluminum and its alloy interconnect lines exhibit a phenomenon called electromigration. These are typically found in the supply and ground rails which always carry unidirectional current. Electromigration occurs after years of usage of the design. When constant current flows through the power and ground interconnects for long time, ions get knocked out by electrons from one place to the other creating piles of ions at one side called hillocks and consequently voids at the other end. This results in open/short faults on the interconnects. This can lead to reliability issues in VLSI designs. Electromigration rules are added as electrical rules which must be adhered to avoid such failures. There are three types of electromigration rules: DC, time-varying unidirectional flow, and bidirectional AC. These are considered while designing power grids. The verification of these design rules is to be passed before the chip tape out. Electromigration is not seen much in copper interconnects and hence gaining importance in today's designs. Copper as interconnect does not exhibit problem of electromigration.

## IR and Cross Talk Analysis

Due to high operating frequency of the designs, at multi-Gigahertz, it is very essential to perform signal integrity (SI) and power integrity (PI) checks like *IR* drop, cross talk effects, and noise analysis and noise to ensure first time success of designs. Noise effect on design can be due to the following reasons:

- Technology scaling resulting in high transistor density.
- Power supply voltage reduction less than 1 V.
- Increased switching and power density.
- Power supply noise due to resistance on power nets, spatial variations on power grids, temporal variations of power supply voltage.
- Cross talk due to one signal interfering with another signal, Capacitive cross talk between RC lines floating and/or drive nets on a chip Floating, signal coupling between nets due to LC transmission lines effect.
- Inter-symbol interference.
- Thermal and shot noise.
- Parameter variation.

Static and Dynamic *IR* analyses are done to check if there are any hotspots in the design. Hotspot in design is self heating power and thermal state of an interconnect which manifests into circuit failure causing reliability issue when a chip is in use. This is due to the occasional high current flowing in denser parts of design with lot of interconnections. If not addressed properly during design, these can render themselves as noise leading to “hard to find” intermittent errors at current switching frequencies. Good practices in layout design guidelines are followed during physical design to avoid them. One of the layout Guidelines is to avoid floating nets which results in capacitive cross talk, picking up signals from neighboring nets. Layout guidelines will be stringent for sensitive circuits like low swing on chip buses, dynamic memories, low swing pre-charge circuitry near supply lines. Inductive effects are more pronounced in input-output circuitry of mixed signal and analog circuitry and are not major problem in digital circuits. Congestion analysis is carried out during physical design. Design congestion is relaxed by suitably replacing them or by distributing them by new placement. High current carrying nets are sized up sometimes using NDR in design enough to carry high currents. Cross talk effect is reduced by adding level restoring circuits called keeper cells in dynamic switching circuits. Some of the layout guidelines to deal with cross talk issues are:

1. To avoid the floating nets.
2. Sensitive circuits like pre-charge circuits are supported by keeper cells.
3. Sensitive nodes are separated by fast switching nets.
4. Avoiding long interconnects on the same layer and sufficient gaps are included between interconnect nets with sufficient gaps.
5. Shielding wires are inserted between the nets. Vdd and Gnd nets are drawn between two parallel long nets which shield the power nets from signal and

clock nets. Dynamic IR analysis shows up hotspots due to cluttering of clock buffers in some parts due to high switching activity. Design congestion is addressed by evenly distributing the clock buffers across the die. The parasitic values from the routed design are extracted from the design. Static timing analysis is carried out using the extracted netlist in spf format using STA tool. The timing reports are analysed at all the design corners and considering on chip variation( OCV) in all the design functional and test modes. PD tools automatically fixes any violations automatically by either resizing or adding/ removing buffers in data paths. However there will still be violating paths which are to be fixed by the designer manually considering the criticality on the functionality.

## Gate Level Simulation

Gate level simulation for some of the functional test scenarios are carried out by back annotating the extracted parasitics in test bench. The gate level simulation is a tedious and very time consuming process. The simulation test bench is updated to feed the input stimulus considering input delays and clock/reset uncertainties. The response from the design under test (DUT) are adjusted considering the output pad delays which are available now. In SoC which has hundreds of primary input- output signals, this task is very difficult and hence a gate level simulation is run on only a small set of critical tests in all modes of the SoC. The design netlist under test now has all the timing parameters like setup, hold times. Debugging a DUT response for a test scenario require design knowledge, timing consideration and more importantly the signal tracing to the actual issue site is done buy back tracing the signal path from primary input-outputs. The design netlist will have internal net names modified by tool specific which will not be debug friendly. Figure 10 shows the flow.

## Electrical Rule Check (ERC)

Electrical rule check (ERC) is a static verification. It is run to check the correctness of electrical connectivity of devices. Design layout is checked for power ground connections, opens/shorts in nets, well/tap connections, floating nets and devices and missing connections. It also checks any design rule violations regarding placement of tap cells, cell densities which is indicative of device reliability issues. Dynamic IR analysis is carried out to detect IR drop bottlenecks, violations of Electromigration (EM) rules, connectivity, and reliability checks to identify weak spots in the power grid, resistance bottlenecks (through short path tracing), missing vias, and current hot spots. Tool provides what-if scenario analysis on IR and EM by using region-specific power assignment. IR map is also called heat map. A typical heat map is shown in Figure 11.

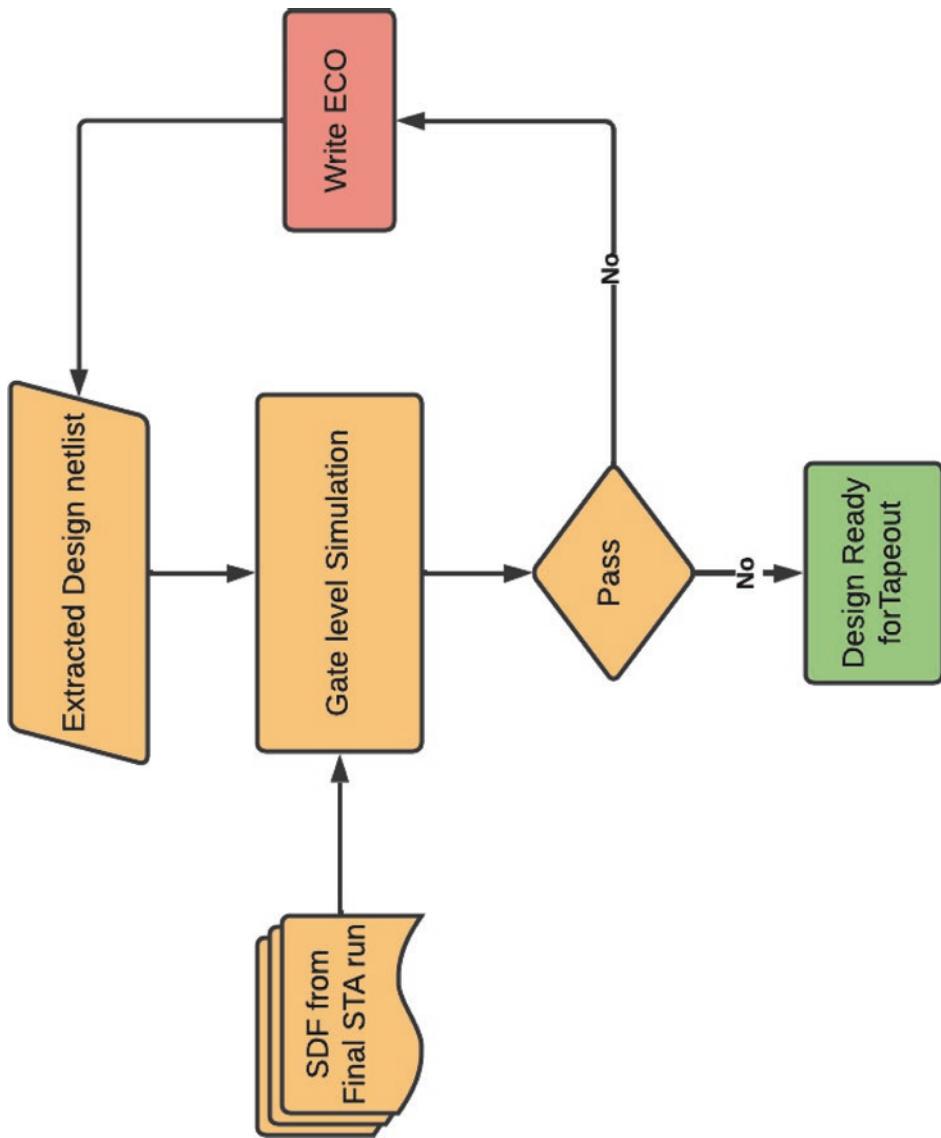
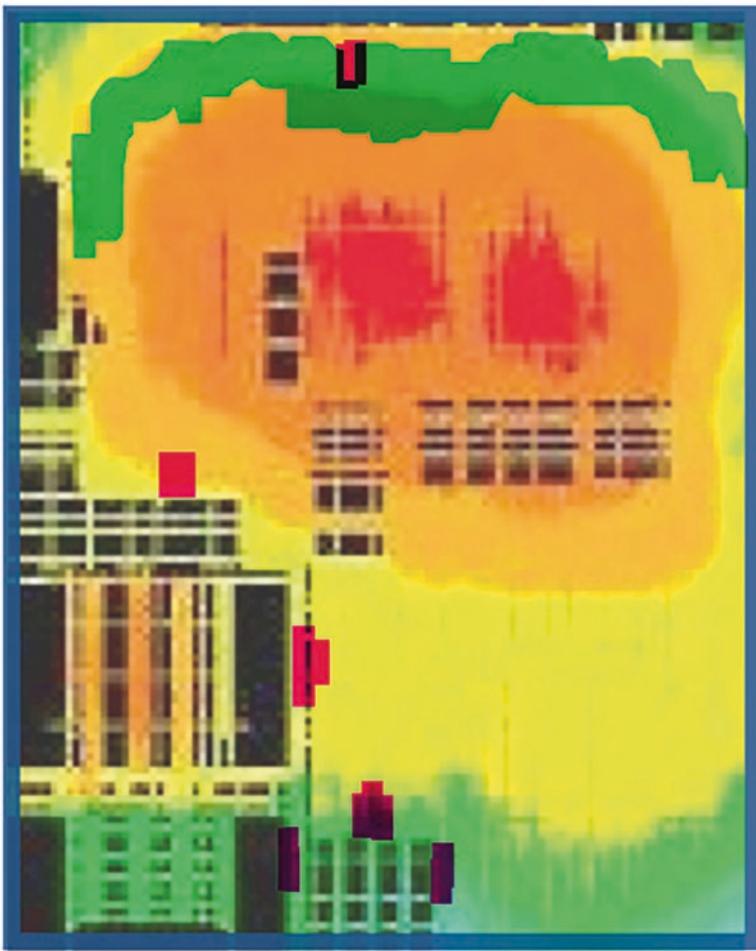


Fig. 10 Gate level simulation flow



**Fig. 11** IR Map

## Design Rule Check (DRC)

Physical design is verified for design rule violations (DRV). This is done by the process called *design rule check (DRC)*. It is a process of checking physical design layout data against the fabrication-specific rules supplied by the process vendor to ensure successful fabrication. Process design rules are related to X-Y dimensions of layout structures and not vertical dimensions of the layers. Example rules from foundries for a few technologies are shown in Table 1.

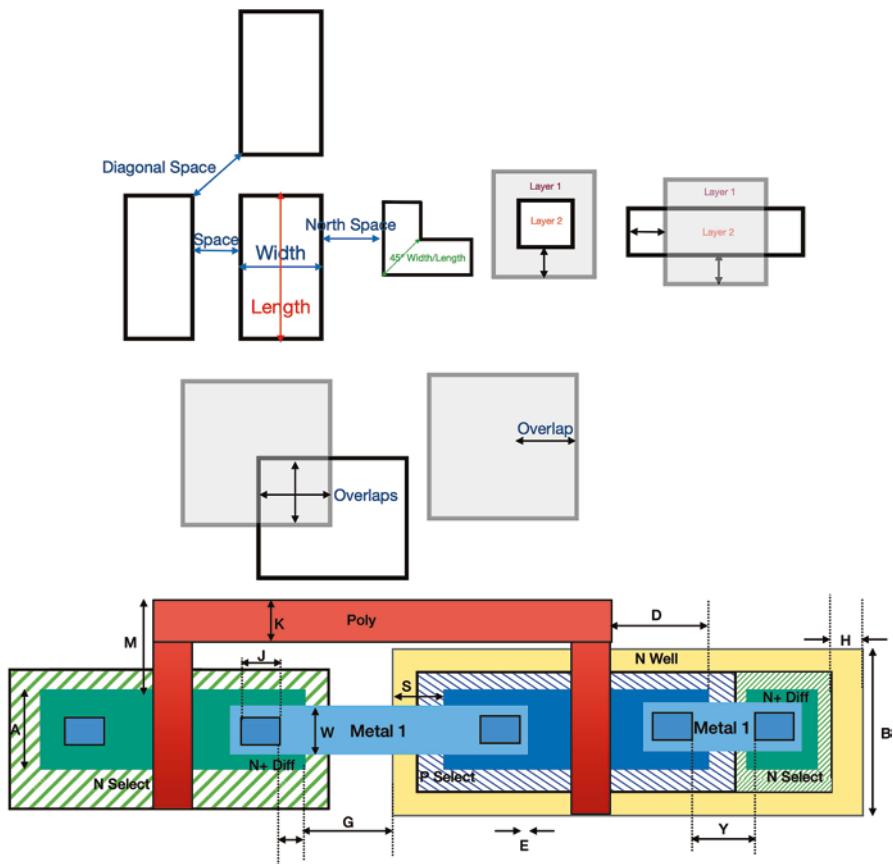
Typical design rules for a particular technology node look like Figure 12.

Designs with violations for any of the rules will result in wrong structural dimensions during fabrication and will result in bad yield and functional failures in chips. Some of the design rules are listed below:

- Maximum metal rules: Lengthy interconnecting metals structures are prone to stress cracks. So, the design rules for long route metals are made to have holes or slots in them to avoid cracks.

**Table 1** Sample DRC rules for different technologies

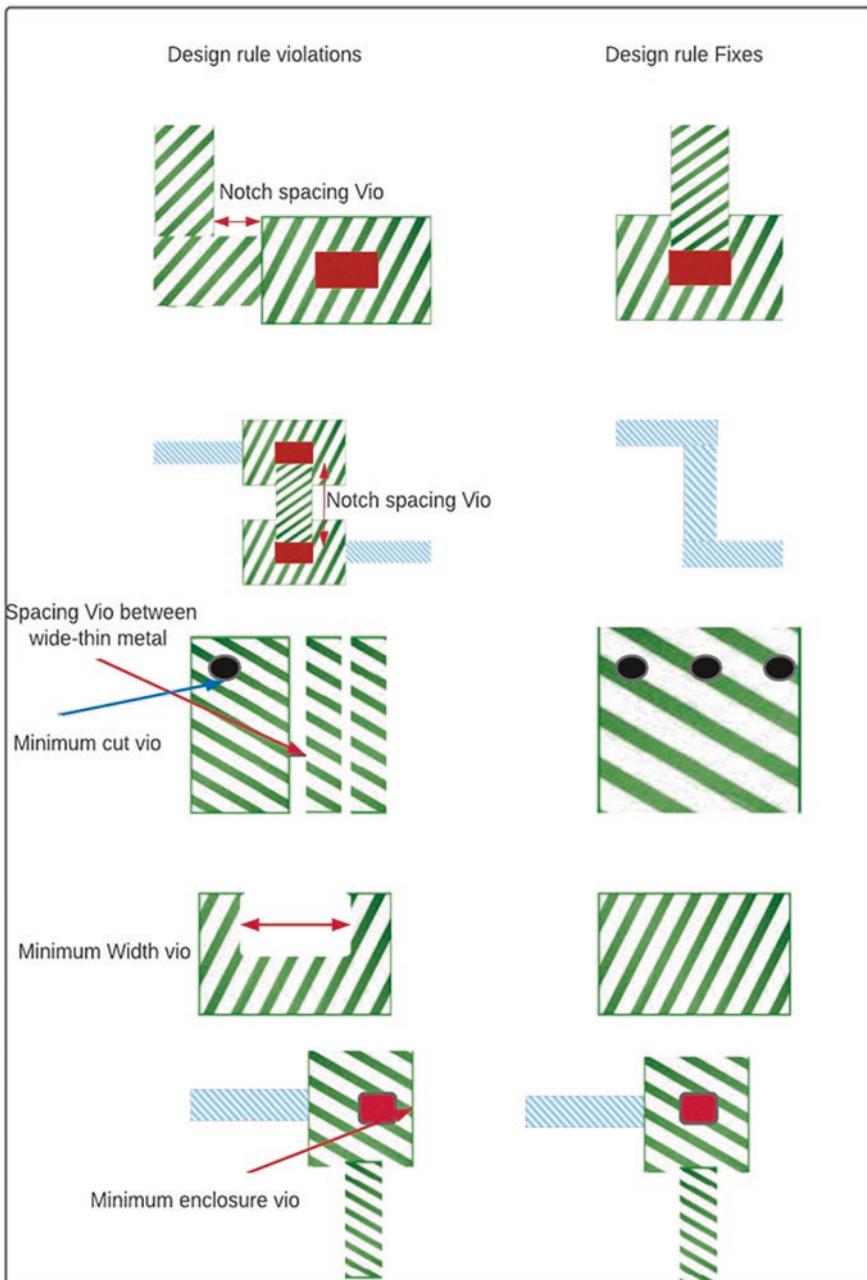
DRC rule	130 nm	90 nm	65 nm	45 nm
Width-based spacing	1–2	2–3	3–5	7
Min area rule	1 pitch	2 pitch	3 pitch	5 pitch
Cut number (Via)	N/A	1–2	4–5	5–6
Dense EoL (OPC)	N/A	N/A	M1/M2	All layers
Min step (OPC)	N/A	1	5	5

**Fig. 12** Design rules

- The structures are to be only multiple of 45° inclinations.
- The corners of structures are to be snapped to minimum resolution grids in the layout.
- Minimum spacing is expected to be maintained between structures within the same layer or across the layers.
- There is a requirement of minimum overlaps when two structures in same layers or different layers are to be connected.

- There is notch rule which specifies minimum spacing rule for objects on same net. Minimum notch size defined in same layer or merged objects.
- Minimum number of cuts permitted in vias in metal interconnects.

Some of the design rule violations and fixes in the designs are shown in Figure 13.

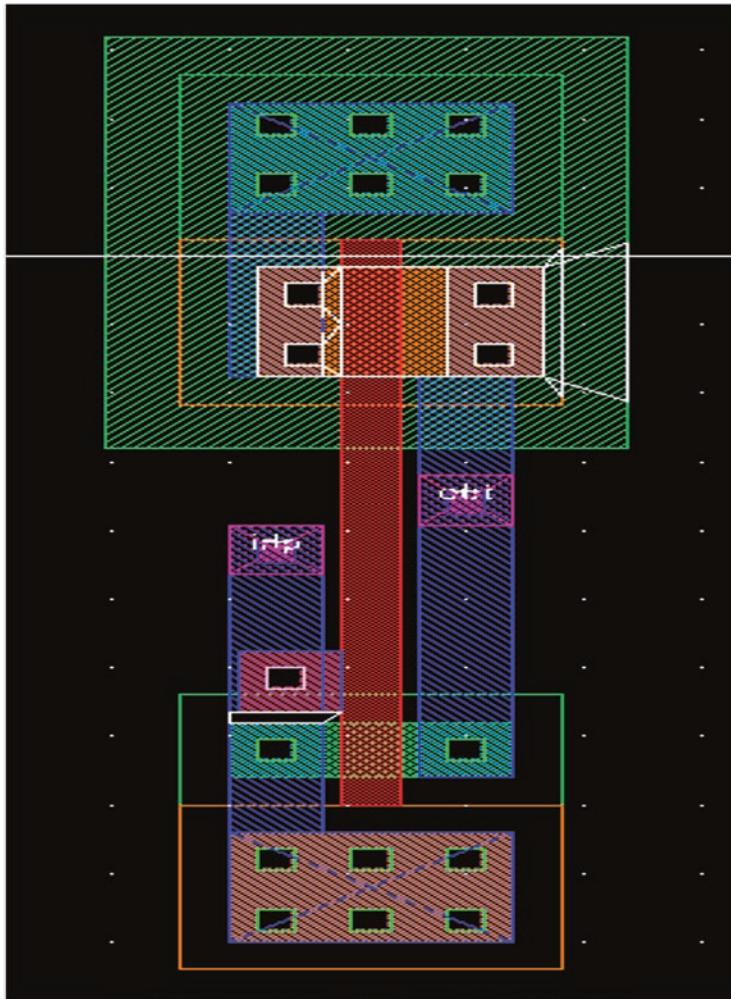


**Fig. 13** DRC violations and fixes

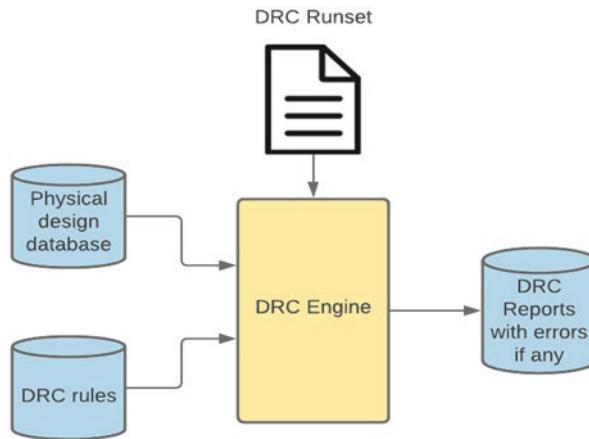
Design rules are classified as *Base layer DRCs* and *Metal layer DRCs*. Base layer design rules define base layer structures upto metal 1 (M1). They define spacing rules for transistor structures like Well spacing, Poly spacing, Poly width, etc. Tap cell requirements, well continuity to be maintained with filling empty spaces, etc. Most times Base DRCs will be through if the design is placed legally without any structural overlaps and no gaps.

Metal Rules include metal minimum and maximum widths permitted in metal layers, minimum spacing required to be maintained, Via enclosures for minimum cut, via to via spacing in multi-cut vias, etc.

DRC is carried out in physical design tool by generating computational geometry from design layout and checking the relation of overlap or distance between polygons of either the same or the different layers. The resulting geometric data



**Fig. 14** Screenshot of DRC



**Fig. 15** DRC flow

from the design layout is verified against rules run set file provided by the fabrication house. A screenshot of DRC run is shown in Figure 14.

The DRC engines accept DRC rule set, design layout, and the runset (runset is a script containing the ruleset file location, design files, and directory structure) to check the design against rules for any violations. The DRC flow is shown in Figure 15.

## Design Rule Violation (DRV) Checks

DRV is typically performed during Physical design after CTS, routed, and as the final design completion. The typical steps involve the following:

- Perform RC extraction of the clock nets and compute accurate clock arrival time.
- Adjust the I/O timings.
- After implementing the clock tree, the tool extracts the input and output delays to reflect the actual clock arrival time.
- Perform power optimization of the design.
- Use a large/max clock gating fanout during insertion of the ICG cells.
- Merge ICG cells that have the same enable signal.
- Perform power-aware placement of integrated clock gate (ICG) and registers.
- Check and fix any congestion hotspots.
- Optimize the scan chain.
- Fix the placement of the clock tree buffers and inverters.

- Perform placement and timing optimization.
- Check for major hold time violation.

## Design Tape Out

SoC Design is ready for tape out to the foundry after the physical design verification is completed to the satisfaction of the designer. Final physical design database is saved in GDS II file or OASIS file format, and is sent to a fabrication house through file transfer protocol (FTP) process. This is called design tape out. Final SoC design consists of design database, design netlist, design reports including DRC reports, final the design constraints. Fabrication house verifies the Design database and design reports before it is accepted for fabrication.

## SoC Design Verification Challenges

With continuous scaling and increase in complexity, the design rules have become extremely complex making design convergence a major challenge. Design convergence is typically used for the process of achieving specified performance, power and the size goals specified for the design with meeting DRC rules. The mask generation data set is derived from Design layout database in advanced nodes like 7 nm and below. This require further processing involving necessary corrections considering mask generation process which is specific to foundry. Earlier, in higher technology nodes, the technique called optical proximity correction (OPC) was applied to design data to improve the fidelity of corners and edges of the layout structures. In subsequent processes, a set of *sub resolution assist features (SRAF)* is applied to mask data to provide constructive or destructive interference to illumination intensity while exposing the photoresist coated wafers. Existing deep submicron processes use extremely complex algorithm called *source mask optimization (SMO)* to derive mask data and the nonuniform exposing illumination pattern during this process.

Layout rules have become extremely critical and complex because of small resolution of lithography exposure requirement and non uniform fabrication process. Additional rules like forbidden pitches, extra dummy shapes to be inserted and limited range of allowed cell sizes, contact, vias, and nets for both improved exposure and net uniformity in designs. Sometimes, even if the design database is DRC clean, there can be an extreme challenge to generate SMO solution for mask data generation. Hence, in deep submicron technologies, foundries verify design by Lithography Process Checking (LPC).

## Available Solutions

Foundries and EDA tool vendors collaborate to incorporate the LPC requirement during physical design verification. Source mask optimization (SMO) algorithm for advanced technology from foundry is incorporated in PD tools to automatically verify the mask data generation early in the design stage. This helps to define high-risk layout topologies checking LPC patterns in the design. The combination of rule checking, pattern matching, and SMO-like algorithm are applied at the design stage to ensure correct mask generation. With design complexity increasing, complying to these requirements is extremely compute-intensive activity. For advanced nodes, LPC services are offered as separate services to accept tapeouts for fabrication. Integrating LPC-based pattern matching along with physical design verification addresses to a major extent tapeout feasibility.

Close association with the foundry for reviewing any DRC waiver is another method to ensure fast design convergence. Design database saves PDK version and any DRC waiver granted by the foundry for design reuse. This requires close coordination of design houses with the foundries and very robust data management systems.

## Machine Learning in Physical Design Verification

Physical design verification require high end computing systems. It is in this context, advanced PDV tools adopt machine learning techniques to improve the execution time and machine utilization.

Typically, PDV and design sign off executions for complex SoC design requires computing machines with hundreds of CPUs and effective management is the key for faster designs.



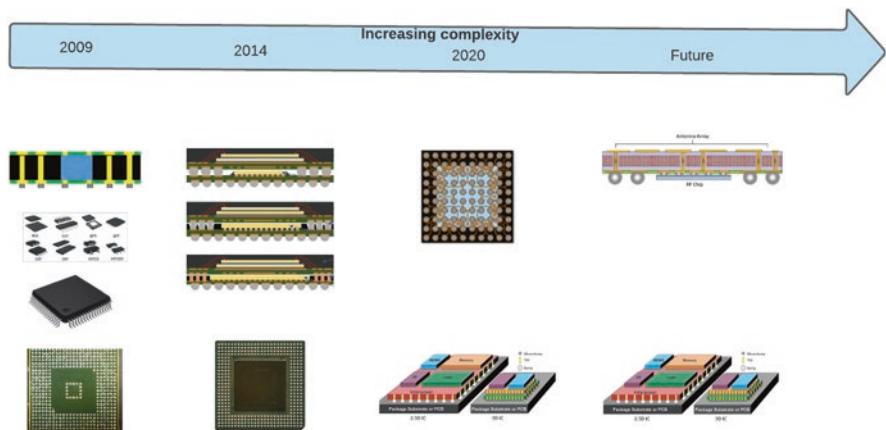
## SoC Packaging

VLSI SoC die is packaged to interface it to the rest of the world in a product. The dies are protected from mechanical stress, environment (humidity, pollution), and electrostatic discharge during handling. Functional reliability in dynamic environmental conditions by burn in tests and other safety tests are carried out on packaged ICs before they are used in product design. SoC Package provides high-yield assembly for the next level of integration or interconnection on circuit boards. Important functions of packaging are the following:

1. Protects the inside system on chip circuitry.
2. Protects the fabricated silicon die from environment.
3. Provides path for heat dissipation from chip to the ambience.
4. Provides reliable electrical connectivity to the neighboring systems through interface pins.
5. Handles further reliability tests on the system on chips.

## Trends in Packaging

VLSI technology has piggybacked on packaging technology for meeting never-ending demand of increased integration of functionalities in multi-chip system solutions. This has kept alive “More than Moore” vision so far. Implementing more complex systems on chip as a monolithic IC has become super expensive and soon will not be commercially viable. This was predicted by Gordon Moore who stated that “It may prove to be more economical to build large systems out of smaller functions, which are separately packaged and interconnected.” Technologies like system



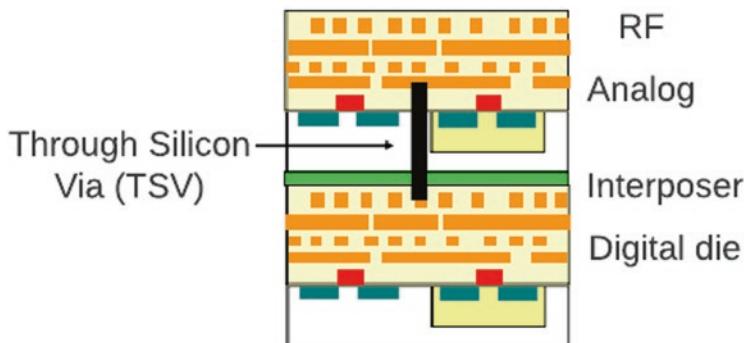
**Fig. 1** Evolution of packaging technologies

in package (SiP) for such systems have worked out well as an alternative to designing an advanced-node monolithic SoC for heterogenous ICS. Exploring such alternative is continued in parallel till date with Moore miniaturization. Evolution of packaging technologies for such integration is shown in Figure 1.

With growing need for integration of many more heterogenous die-based architecture, advanced semiconductor package design methodologies are required [1]. Traditional packaging poses limitations and makes SoC designs in less than 7nm technologies impossible to meet PPA goals. Sub-ten nanometer process technologies introduce new difficulties, to realize nonplanar finFET transistor structures, needing complex double-patterning lithography for critical layers, and perhaps even new substrates. Newer systems demand integration of chips realized in diverse technologies. This has led to exploration of stacked die integration technologies.

### ***Concept of Stacked Dies***

Over the years, designers constantly strived to catch up with processing technologies to make Moore's law true first to bridge the design-productivity gap and second to overtake it with innovative packaging technologies. Innovation in interconnect technologies such as multi-chip modules, silicon in package, and package-on-package schemes have played a major role in this endeavor. The current 3D IC concept is believed to be promising technology to stack multiple dies through a layer of *interposer*. An interposer is a silicon layer used as a bridge or a conduit to allow IC signals to pass through it from one die to another die or die to board with varying signal pitches. Typical 3D IC structure is shown in Figure. 2.



**Fig. 2** 3D IC structure

### ***3D Integration Schemes***

Few commonly used schemes for integration of 3D ICs are

#### **2.5D-IC**

In this integration concept, two or more dies are placed adjacent to each other facing downwards toward the silicon interposer. Dies are supported by micro bumps on their active surfaces which get connected to the silicon interposer. Sometimes, the redistribution layer (RDL) is added to align the micro bumps to the interposer pads or signals are routed on the RDL to connect them. Connection from RDL to interposer is achieved with *through silicon vias* (TSV)s. There is a possibility of multiple RDLs to the package for complex integration.

#### **3D-IC**

In 3D-IC concept, dies are stacked one above the other and they are interconnected vertically using *through silicon vias* (TSV)s. The stacked dies can be similar dies such as memory dies or they can be processed for different functionalities fabricated using different compatible technologies. Integrating similar dies as a die stack is called homogenous 3D technologies. Integrating dissimilar dies in a stack is called heterogeneous 3D technology. 3D heterogeneous integration technology is very complex but most promising to achieve higher levels of integration. 3D Technology offers different approaches to integrate multiple dies on single 3D chip, which are discussed below.

**Face-face:** Dies are connected using micro bumps wherein lower die also has TSV through active layers and substrate to metallization on its back surface. Back

surface of the die acts as RDL using pads on to which pads are laid to connect them to package substrate.

Face to back: Two or more dies are placed one above the other and connections are made through TSV and DSL and micro bumps and to the package substrate.

5.5 D-IC: In this integration approach, the 3D-IC is connected to 2.5D-IC silicon interposer to enhance further integration to develop high bandwidth, compute-intensive solution on chips.

## Chiplets

Chiplet is a fully functional system component with an interface that is designed to work with other chiplets to form a system on chip. Integration of chiplets uses advanced packaging information during chip design. The concept of multiple chips in a single package is an old technique. Multi-chip modules (MCM)s, System in package (SIP) existed in semiconductor technologies as early as 1980. Assembling chiplets side by side on the same substrate and interconnecting them is termed as 2.5D technology. Applications like high-performance computing, AI, systems with incredibly large memories of high-bandwidth and low-latency demand integration of multi-die fabricated by different technologies like MEMS(Sensors), RFCMOS, Analog, and many others for multi-die integration. This has forced designers to look for advancements beyond 2.5D technologies to 3-D IC technologies and heterogeneous integration.

## 3D IC Constituents

Following are different constituents of 3D-ICs each of which are discussed in brief:

- Bumps and balls
- Ball grid arrays
- C4 (controlled collapse chip connection) bumps
- Micro bumps
- Through silicon vias
- Redistribution layers
- Silicon interposer

Bumps and balls: Bumps and balls serve to match the interconnect spacings and dimensions to enable the connection between two technologies. The dies are connected to PCBs through these bumps and balls.

Ball grid arrays: A package-level interconnect that connects a packaged device to a PCB.

C4 (controlled collapse chip connection) bumps Solder balls arranged as an array of grids used to connect bare dies to PCBs. These bumps have a pitch of 180  $\mu\text{m}$ .

Micro bumps: Micro bumps are small solder balls of the pitch of less than 10  $\mu\text{m}$ , that are used to connect two dies face to face.

Through silicon vias (TSV): TSVs use the CMOS etching process to create vias from active to top of the die and through the backside of the substrate. They are

further filled with copper or tungsten to make interconnections from the circuit to the top of the die to the backside. The typical diameter of the TSV is 12 µm with a 180 µm pitch. Filling these long vias is very challenging and time-consuming. Hence, the wafers are trimmed to 50 µm thickness from 300 to 350 µm original thickness. Another main challenge is that these vias travel through active layers of the wafer and the designers must take care of the ESD issues that originate from them during the physical design of the chip planned for 3D with TSV. TSV can also result in mechanical stress which also needs to be addressed.

**A redistribution layer (RDL):** The redistribution layer is shown in Figure 3 consists of metallization on the surface of a die, either on its active face or on the back of the substrate, which is then patterned to redistribute connections from one part of the die to another, or to match the pitch of two interconnection technologies. Redistribution layers have landing bumps on which micro bumps are formed to make connections.

**Silicon Interposer:** An interposer is a silicon layer used as a bridge or a conduit to allow electrical signals to pass through it from one die to another die or die to board with varying signal pitches. Silicon interposer is shown in Figure 4.

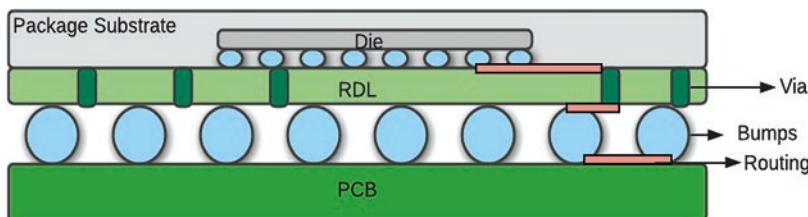


Fig. 3 Redistribution layer (RDL)

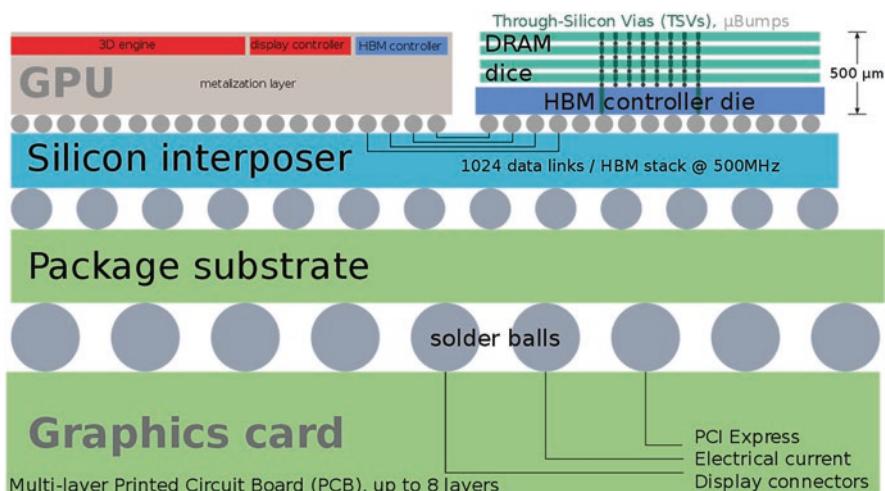


Fig. 4 Silicon interposer. (Source: Wikipedia)

## ***EDA Tools for 3D ICs***

Conventionally, design packaging flow is independent of SoC design flow. EDA tools do not support multiple die chiplet integration issues during physical design. For systems of the future, it is necessary that the SoC physical design tools to support off-chip integration, boards, and beyond to get optimum PPA as systems can be designed in isolation. This requires design tools to support features which help designers to virtually model inter chip interconnects, perform electrical and thermal analysis in a multi chip environments. This requires forward integration of package design features into SoC physical design tools. This requires chip designer to consider package design challenges and verifying them to make it work in an integrated environment. Advanced EDA tools already have cross-domain analysis features targeted to the design of 3D SoCs. The scope of Physical design tools get extended beyond SoC design layout to System in package design. PD tools must possess capabilities of designing layouts from transistor level to system with multiple chiplets. Leading EDA platforms of today are a complete, end-to-end solution for efficient, 2.5D, and 3D multi-die system integration with many transformative, multi-die analytic capabilities. Design platform features include advanced features with immersive 2D and 3D visualization, cross-hierarchy exploration, solution planning, design, implementation, solution level validation and sign off analysis. The 3D IC design platform is expected to have capabilities of data aggregation from the SoC designer, the package designer, the board designer, and the product designer for system-level PPA optimization. They provide top-level netlist for connectivity verification and solution level design sign off. Other challenges include:

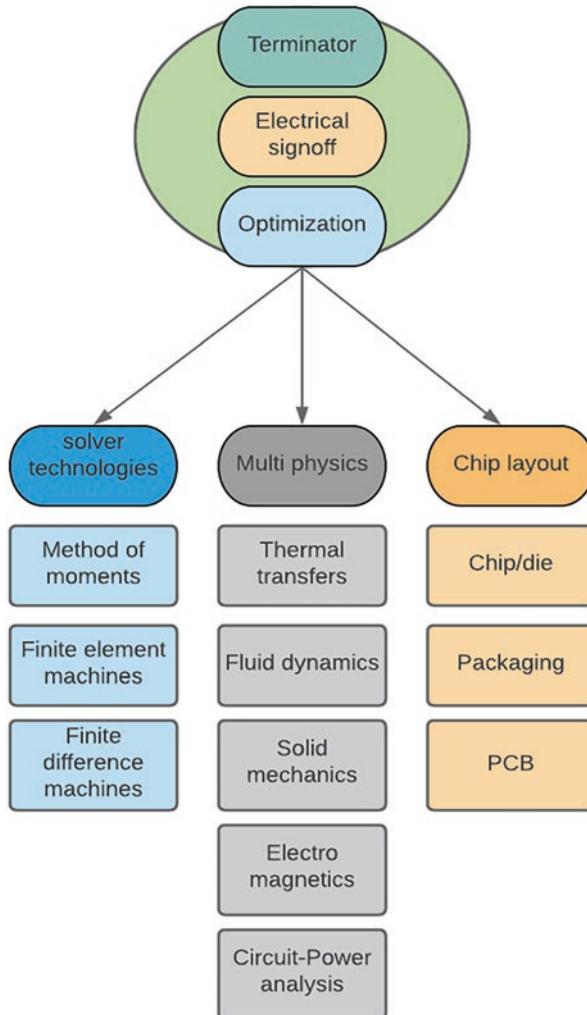
- Package aware physical design verification
- Solution sign off criteria
- PPA goals at the solution level

Design tools for electrical and thermal analysis and support tools must support in-design and electrothermal signoff, for on-chip and off-chip dies and their interconnections. Modeling tools to support cross-domain coupling effects for 3 D SoC designs. The tool feature will be a convergence of Multiphysics analysis (EM, Electrical Power, Thermodynamics, fluid dynamics, and solid mechanics), design layout platforms (Chip design, package design, and board design), and solver technologies (Finite element method, Method of Moments, Boundary Element Method) as shown in Figure 5.

The system design verification strategy must be conducted bottom-up at chip-level, off-chip level, and finally at system-level considering both off-chip and on-chip specifications for integration.

The physical design toolset should include the following features to support future 3D IC designs:

- Circuit design and schematic capture for redistribution layers, interposer signal routing, shielding, and power.



**Fig. 5** System sign off capabilities for 3D technologies

- Place-and-route support, including TSV, micro bump, redistribution layer, and signal routing, power mesh creation, and interconnect checks.
- Design-for-test (DFT) for stacked die and TSV.
- Integrated memory test, diagnostics, and repair systems.
- Support for parasitic extraction for TSV, micro bump, interposer RDL, and signal routing metal.
- support for circuit simulation for multi-die interconnect analysis.
- Thermo-mechanical stress analysis of TSVs and micro bumps in multi-die stacks.
- Physical verification (LVS, DRC).
- Extended design rule checking.

- Chip-level functional verification.
- Static timing analysis.
- IR/EM/SI and other electrical analysis.
- Assembly and yield analysis support.

Factors to be considered for 3D IC design are

1. System-level exploration: It is necessary to consider the right silicon technology for each die that goes into the 3D IC.
2. Physical design EDA design tools:
  - (a) 3D floor planners should consider a three-dimensional floor plan.
  - (b) PD flow should be thermal aware. The tool algorithm should identify hotspots in the circuits and move them close to the heat sinks and apart from each other.
  - (c) Need to insert thermal vias appropriately. Thermal vias are heat discharging guides from high-temperature regions to low-temperature regions.
3. The router in physical design should consider the thermal effect on interconnect delays.
  - (a) Routing critical nets should avoid hot regions.
  - (b) Interlayer vias are to follow new design rules including interposer vias and keep-out regions.
  - (c) Interlayer vias are critical resources that are to be used efficiently.
4. Physical design tools should be capable of doing TSV stress analysis, coupling, and keep-out area considerations.
5. Extraction tools must consider RLC parameters for TSVs, micro bumps, and interposer routing.
6. Physical design tools should have capabilities for the design features of the package.

3-D chips can be manufactured in multiple ways:

- Die to die
- Die to wafer
- Wafer to wafer

## ***Early Adopters of 3D-IC***

There are early adopters of 3D IC technologies in FPGA, advanced processor chips, gaming chips, and for high-density memory solutions. Memory vendors have stacked multiple dies and used 3D packaging to achieve high-density memory chips since many years. Considering 3D technology for heterogeneous systems is the recent development.

## Future Trends of 3D IC Technology

The future of further integration and miniaturization of systems rely on how the ecosystem gets developed for 3D IC technology in terms of design methodology, supporting features in EDA tools, fabrication, and assembly ecosystems. Nevertheless, 3D IC technology is a promising technology.

## Reference

1. [https://en.wikipedia.org/wiki/Three-dimensional\\_integrated\\_circuit](https://en.wikipedia.org/wiki/Three-dimensional_integrated_circuit)

# Current Trends of Semiconductor Systems and Physical Design



Semiconductor Industry was hit with a “*Cannot afford to design chips*” crisis. This is due to high cost of EDA tools, limitation of design resources, high development time, and high risks compounded with poor return on investment for new technology. Cost to migrate to new technology node in nano meter technologies that offer much lesser benefit than before is exorbitant, challenging further semiconductor technology scaling down. Beyond CMOS innovation such as *buried interconnects, backside power delivery, super vias*, next-generation device architectures such as *vertical gate all around (VGAA) FETs* has continued to offer great promise as a good alternative to complex system needs. Innovative design technology co-optimizations, and use of vertical dimension-heterogeneous, multi-die integration, monolithic 3D VLSI all offer potential continuity to technology scaling trajectory. In addition, various “rebooting computing” paradigms—quantum, approximate, stochastic, adiabatic, neuromorphic, etc.—are being actively explored. *Machine Learning (ML)* techniques in SoC Physical Design is not a new for VLSI and was explored to counter the challenges of process technologies below 10 nm SoC Designs.

## Current Trends in Design Methodologies

SoC Physical Design problems are complex. Conventional design flow and tools are not sufficient to realize the complex system on chips. Following section reiterates few examples discussed in the earlier chapters.

### ***Optimal Logic Structure to Achieve PPA Goal***

Placement aware Synthesis is design technique to achieve faster design time for SoCs. The placement information of the design elements is used to generate optimal good quality design netlist by the design synthesis process for faster PPA convergence. This require synthesis process to predict based on the placement information and predicted interconnections.

### ***Optimal Macro Placement***

It is chicken-egg problem for auto placer to find out optimal macro placement. Design Placement process require good quality design netlist and to generate the design netlist, PD information is required. However the two step synthesis process addresses this challenge to a certain extent. Design placement is not automated process though it is tool dependent. It requires good design experience to arrive at optimal placement of design objects. In spite of this, there can be alternative optimal placements possible. Many times, this process may be over kill to the design as it is iterative and designer specific. Manual placement of design elements need to virtually route the design to check routing congestion and other routability checks. This exploration takes large amount of time during design cycle.

### ***Route Aware CTS, Pre-route to Post-route Correlation (Timing, Congestion)***

Pre-route stage must predict how much impact routing topology on the final DRC passed design layout. It also has dilemma between showing congestion and detouring. Mandate at pre-route stage is to show congestion than detouring.

Pre-CTS stage needs to predict an area overhead due to CTS error corrections and timing fixes in the design eventually.

The EDA tools are getting enhanced to address these challenges. However the designer will still continue to assess the quality of SoC the tool has placed intelligently.

Automation for optimal routing solution adhering to all complex DRC rules including lithography process verification (LPV) is still a work in progress. The design sign off verification done on the SoC designs are to be upgraded for more intense and advanced lithography process challenges. The lithography process verification (LPV) is carried out for SoC design are limited. Advancement in process technologies and rapid scaling has reduced lithography exposure wavelengths so drastically for high volume wafer fabrication. The mask data of device and interconnect structures differs from the layout structures requiring corrections in them

before using for mask making. In earlier process nodes, correction technique called optical proximity correction (OPC) was applied to design data to adjust the structures at the corners and edges. This improves exposure fidelity of the design structures at nanoscale. In later technologies, subresolution assist feature (SRAF) enhances process window of exposure of structures of isolated and partially isolated structures and during litho. But this also cause yield issues in large scale manufacturing at nodes less than 7nm. This demands selective application of SRAF for design data predicting SRAF printing characteristics. litho aware process window need total control on the process windows and exposure to structures. Prediction algorithms need regression techniques to be incorporated for design data correction to generate mask data for both scattering and reverse scattering adjustments selectively on structures. Thus design tapeout at final GDS II require further processing before use in mask data generation in current and future nodes. New technique called source mask optimization (SMO ) is followed for advanced nodes to address this to an extent. Another challenge is that the design rules have become extremely complex considering so many process variations but still not enough to consider design data for guaranteed processing.Increased set of rules for litho corrections, fabrication uniformities and process abnormalities like guard areas, addition shapes, need for additional dimensions of device structures, vias, interconnects, contacts and etch uniformities are needed to ensure first time success. This introduces enough risk to challenge the design process of existing design sign off methods. The new sign off methods demand both pattern checking algorithms for rule checking and model analysis for predicting process variations which are compute intensive and need a holistic approach.

There are many more such problems which cannot be handled only by conventional algorithms. With every node, every design, tools need to be tuned to get the best possible results. This is where a good designer plays a role. Even after that, there will be still some suboptimal results if you look very closely. It becomes even more complex as earlier in the design cycle some logic cores are bought off the shelf. This demands core evaluation process in the design flow which many times results into some wrapper development to interface third party core in the system. These are some of the problems where AI can be deployed. With new technology node, existing designs are fed to train the models to learn the process of arriving at optimal solution. Again, it is not as simple as stated above because reinforced learning needs very big problem set and its solution available for tool to learn from it. This is not practical for physical design, more so when you are at cutting edge and technology is not widely used. Access to limited set of design data distributed across different design houses is another major challenge to be addressed. This needs a lot of innovative ways technologically and logically to make AI work. In nutshell, AI and ML will certainly play big role in future but they will not make human job redundant. The recent article from Google captioning “Google is using AI to design chipsets in just six hours,” “Google is using Artificial Intelligence (AI) to design the next generation chipset” And “the AI is taking only six hours to design, faster than human designers who take months,” statement from Google “in under six hours, our method automatically generates chip floorplans,” and “The new chips are said to be

superior or comparable to those produced by humans in all key metrics including power consumption, performance and chip area” are to be taken as motivation to innovate ways and means to adopt AI and ML to chip designs. This method undoubtedly utilizes human experiences to become better and faster at solving new instances of the problem. This allows artificial agents producing the chip to gain more experience than any human designer.

Machine learning (ML) is a powerful computer science technique which can derive knowledge from big data and provide prediction and matching. Since nanometer VLSI design and manufacturing have extremely high complexity and gigantic data, there has been a surge recently in applying and adapting machine learning and pattern matching techniques in VLSI physical design (including physical verification). ML can raise the level of abstraction from detailed physics-based simulations and provide reasonably good quality-of-result. Many designers have explored ML Techniques both inside and around the SoC Physical Design tools and flows. Many complex steps in the design flow made designers think “Can we predict outcomes better?” and complex loops within these flows, made them ask the question—“How to initialize?”—Both these problems are NP-Hard. Problem types solved with Machine Learning are Classification, Regression, Dimensionality reduction, Structured prediction, and Anomaly detection. Researchers have successfully used ML in Yield modeling (Anomaly detection and classification), Lithography hotspot detection (classification), Identification of data path-regularity (classification), Noise and process-variation modeling (regression), Performance modeling for analog circuits (regression), Design- and implementation-space exploration (regression).

Areas where Machine Learning can be immediately applied are Design modeling, prediction, and correlation during physical design. When Suboptimality is expensive, i.e., 10% of (power, speed, area) is half of the benefit from new node and Iteration is expensive, Conservatism (“margin”) is expensive, Machine learning can be used effectively in improving PPAs and some of the applications include:

- Achieving faster design convergence through predictors of downstream flow outcomes that understand both tools and design
- Removing unnecessary design margins through correlation techniques
- Clustering
- Dimensionality reduction
- Density estimation
- Lithography hotspot detection
- Data/pattern-driven physical design

Physical design of complex SoC becomes unpredictable when it comes to optimization and design convergence for specified QoR goals. Higher need for flexibility in EDA implementation tools (Synthesis, DFT, and STA) make design convergence unpredictable and hence more iterations [1]. Higher the iterations, higher the cost of design. As the SoC QoR is compromised, the design capability gap widens. The design capability gap gets reduced when the design convergence becomes more predictable. But with larger design partitions, predictability being less requires a large number of iterations to converge to desired PPA goals. Increasing

number of design partitions reduces the number of iterations, improving predictability and hence shorter design convergence times. This chapter suggests five areas where ML can be applied to improve design cost for a set QoR [1].

## System-Level Technology Trends

The key trends seen in system-level technologies are the following:

1. This shift from proprietary to open source system architectures. This is driven by the combination of open source software frameworks and the rise of AI machine learning frameworks capable of creation of very effective models based on statistical inference than programming the general purpose processing elements.
2. One Physics to many: Earlier couple of decades have been spent in building systems based on transistor scaling but however today many technologies like Carbon nanotubes, Adiabatic and reversible computing, Neuromorphic and brain-inspired computing, Networks of organic and inorganic materials are paving ways for interesting solutions in most cost-effective and innovative ways.
3. Transition from data center to data everywhere [2] shift focus from data center solutions.
4. From scarce e-memory to memory of abundance.
5. Centralized power full systems to distributed systems.
6. Different IP cores to operate as self timed cores interfaced to a interconnect IP core in system. This simplifies time domain cross over issues drastically.

## Future Trends of Systems [3]

Following trends are clearly seen in systems of the future:

- Security and privacy are key requirements of future systems.
- AI and AR becoming key technology drivers for system development.
- Advanced packaging is a key technology for enabling architectural diversity.
- Open source hardware initiatives with flexible and more open architectures are seen trending. However, managing this extreme heterogeneity will present difficult application development and system software challenges.
- As data grows disproportionately at the edge, computation will follow it, with increasingly demanding workloads in increasingly challenging space, weight, power, and costs envelopes creating opportunities for nonconventional architectures.

With ever-growing thrust for data consumption, the future systems show clear requirements for Security and Privacy, general purpose to build for purpose, AI and augmented reality features, proprietary to open system architectures. The complete ecosystem will cater to enable the realization of these advancements.

## References

1. <https://vlsicad.ucsd.edu/Publications/Conferences/356/c356.pdf>
2. <https://ieeexplore.ieee.org/document/7059020>
3. <https://www.quora.com/How-Will-AI-impact-on-VLSI-industry>

# Question Bank



## Chapter 1

1. Scaling down of feature size in VLSI technology due to Moore's law resulted in \_\_\_\_\_.  
(A) Smart Architectures (B) Process technologies (C) Smart designers  
(D) Advancement in lithography and other process technologies
2. More than Moore trend involves \_\_\_\_\_.  
(A) Further scaling of feature size (B) Designing Large systems by reuse  
(C) Integration of CMOS and other CMOS compatible technologies (D)  
Moore's law less than 50 nm
3. Chiplets are small chips with full functionality ready to integrate to larger  
chips on PCB.  
(A) True (B) False
4. Silicon Interposer is used to \_\_\_\_\_.  
(A) protect chips (B) integrate packages (C) connect 2 PCBs (D) To intercon-  
nect two silicon dies in stacked ICs
5. Design of analog and mixed signal cores are carried out by \_\_\_\_\_.  
(A) Advanced digital technologies (B) Hand layouts (C) Full custom design  
techniques (D) Standard cell design
6. Standard cell based design methodology is used to design \_\_\_\_\_.  
(A) Large digital cores (B) RF chips (C) Sensors (D) Cells
7. Physical design is a process of converting design netlist to a testable netlist.  
(A) True (B) False
8. Physical design is a process of converting VLSI design netlist to Design layout.  
(A) True (B) False
9. Basic method of capturing circuit topology and layer information in a simple  
diagram is called \_\_\_\_\_.  
(A) Physical diagram (B) Stick diagrams (C) Structural design (D) Synthesis

10. Design rules are defined to address \_\_\_\_\_.  
 (A) Interconnect rules (B) Manufacturing limitations (C) Good Artwork of design (D) To get working design
11. \_\_\_\_\_ forms the basis of physical design.  
 (A) Netlist (B) Stick diagram (C) Layout (D) Silicon real estate
12. Timing delays of design paths are represented in \_\_\_\_\_ file format.  
 (A) SDF (B) ASCII (C) Verilog (D) LEF
13. Die size, logical connectivity, physical location in design layout are defined in \_\_\_\_\_.  
 (A) LIB files (B) DEF file (C) LEF file (D) SDF file
14. Process design kit (PDK) is supplied by \_\_\_\_\_.  
 (A) Foundry vendors (B) Front end designers (C) Backend designers (D) System architects
15. PPA in VLSI design is to ensure \_\_\_\_\_.  
 (A) Design works (B) Design Works fast (C) Design works with lowest power (D) Design is realized with specified Performance, power and Area

## **Chapter 2**

16. Two main regions of VLSI physical design layout are \_\_\_\_\_ and \_\_\_\_\_.  
 (A) Core design, IO design (B) Macro, Standard Cell (C) Power, Logic (D) Soft and Hard
17. \_\_\_\_\_, \_\_\_\_\_, \_\_\_\_\_ and \_\_\_\_\_ are four main stages of physical design.  
 (A) Synthesis, STA, DFT, Routing (B) Floor plan, Placement, CTS, Routing  
 (C) Logic design, verification, synthesis, STA (D) RTL design, verification, validation DFT
18. Power planning and power routing is carried out in \_\_\_\_\_ stage in VLSI physical design.  
 (A) Routing (B) Placement (C) CTS (D) Floor plan
19. A SoC design after the floor plan include power mesh for power distribution of design objects.  
 (A) True (B) False
20. Antenna diodes and buffers are added in \_\_\_\_\_ stage of physical design.  
 (A) Floor plan (B) Preplacement (C) Routing (D) Design finish
21. Placement legalization is a process in Floor plan and placement stage where \_\_\_\_\_.  
 (A) Standard cells are fully placed (B) Macros are placed (C) All placement violations are fixed (D) Placement of IO pads are done
22. Chip Utilization depends on  
 (A) Only on standard cells (B) Macros (C) Standard cell and Macros (D) Standard cells, macros, IO pads
23. Soft blockages are restricted areas in physical design layout where \_\_\_\_\_ are placed.  
 (A) Only buffers and inverters (B) Standard cells (C) No Cells (D) Any cells

24. Scan chains are removed before placements because  
(A) They are set of flip-flops (B) They are not timing critical paths (C) They are not critical (D) They are chained
25. HFN synthesis is done in CTS stage.  
(A) True (B) False
26. Scan chain reordering is done in \_\_\_\_\_ stage.  
(A) PrePlacement (B) After Placement (C) CTS (D) Synthesis
27. CTS and HFN are addressed simultaneously.  
(A) True (B) False
28. Tiles are minimum cell unit with unit length and width where a single cell is placed.  
(A) True (B) False
29. Major physical design verification techniques are \_\_\_\_\_, \_\_\_\_\_ and \_\_\_\_\_.  
(A) Logic simulation, LEC, LVS (B) Simulation, STA, DRC (C) DRC, ERC, LVS (D) Fault simulation, DRC, STA, Signoff
30. Circuit extractor extracts \_\_\_\_\_ from Layout.  
(A) Schematic (B) Layout (C) Timing and coordinates (D) Netlist with parasitics
31. NDR is used for only for clock signal nets.  
(A) True (B) False

### Chapter 3

32. Floor planning of physical design is not that important in physical design.  
(A) True (B) False
33. Design partitioning aims to \_\_\_\_\_.  
(A) Split designs in equal number of blocks (B) IO, and Core (C) Smaller manageable blocks to be placed such that they interface with other functional blocks minimal delays (D) Get best power performance
34. Terminal pitch is defined as \_\_\_\_\_.  
(A) Spacing between two IO pads (B) Two standard cells (C) Two macros (D) Minimum spacing between two successive terminals of the design block
35. \_\_\_\_\_ is the ratio of perimeter of the partitioned block to its terminal pitch.  
(A) Terminal pitch (B) Pin Count (C) Utilization factor (D) Terminal count
36. The number of *signal nets* which connect one partition to other partitions should not be more than terminal count of the partition.  
(A) True (B) False
37. Physical design can be \_\_\_\_\_ and \_\_\_\_\_.  
(A) Big, Small (B) Pad Limited and IO limited (C) Pad limited, Core limited (D) Bare or packaged
38. Die size estimate is not required during physical design as it is only required after the design.  
(A) True (B) False

39. Physical design is pad limited if number of IO pads are \_\_\_\_\_.  
 (A) More (B) Less (C) Pad area same as Core area (D) Core area more than Pad area
40. Physical design is core limited if pads are \_\_\_\_\_ in number and there is \_\_\_\_\_ space between pads.  
 (A) More, no (B) Less, more (C) Small, overlapping (D) More, nonoverlapping
41. Aspect ratio in physical design is the \_\_\_\_\_.  
 (A) Ratio of number of pads to number of modules (B) Ratio of die width to die height (C) Ratio of die height to die width (D) Ratio of number of macros to number of pads
42. Aspect ratio in physical design is given by \_\_\_\_\_ where  $V$  is number of vertical resources and  $H$  is number of horizontal resources.  
 (A)  $\frac{V}{H}$  (B)  $\frac{V/2}{H}$  (C)  $\frac{H}{V}$  (D)  $\frac{H}{V/2}$
43. Rings, Rail, and stripes are part of \_\_\_\_\_ network.  
 (A) Clock (B) Macro (C) IO (D) Power
44. \_\_\_\_\_ carries power and ground around the die in power distribution network.  
 (A) Stripes (B) VDD (C) Rail (D) Ring
45. \_\_\_\_\_ carry power and ground to standard cells in power distribution network.  
 (A) Stripes (B) VDD (C) Rails (D) Ring
46. \_\_\_\_\_ carry power and ground to across the die in power distribution network  
 (A) Stripes (B) VDD (C) Rails (D) Ring
47. \_\_\_\_\_ is the placement aware design process of converting RTL design to gate level netlist.  
 (A) Two step synthesis (B) Synthesis (C) Physical synthesis (D) Multistage synthesis
48. Buffers and antenna diodes are added at the macro and block-level IO ports in \_\_\_\_\_ stage.  
 (A) Preplacement stage (B) Floor plan stage (C) Routing stage (D) Post placement stage
49. \_\_\_\_\_ are added to avoid latch up problem in the SoC designs  
 (A) Spare cells (B) Decap cells (C) Antenna diodes (D) Corner pads
50. Reset signal is an example of \_\_\_\_\_ net.  
 (A) HFN (B) SSN (C) LFN (D) Control net
51. Legalization of the placement of design elements in physical design \_\_\_\_\_.  
 (A) Move cells and Macros to an extent as per geometric and layout rules (B) Snaps to grids (C) Do not disturb macros (D) Removes overlaps of macros
52. Clock net is HFN which is routed in placement stage.  
 (A) True (B) False
53. Scan chain reordering is done in \_\_\_\_\_ stage.  
 (A) Preplacement stage (B) Post placement stage (C) CTS stage (D) Routing stage

54. \_\_\_\_\_, \_\_\_\_\_ and partially abutted are three methods of placing design components.  
(A) Butted, Abutted (B) Abutted, non-Abutted (C) Non-abutted, partially abutted
55. Number of standard cells which can be accommodated in core height depends on \_\_\_\_\_.  
(A) Design height (B) Standard cells in the library (C) Die size (D) Chosen technology
56. If the core utilization is 70%, remaining 30% is available for the \_\_\_\_\_.  
(A) CTS (B) Spare cells and special cells (C) Decap cells (D) Routing
57. If the number of IO signals of two macros are 25 each with the pitch of 0.6, the number of horizontal resources is 6, then the channel width is \_\_\_\_\_.  
(A) 5 (B) 6 (C) 25 (D) 50
58. Only few important special cells can be placed in Hard blockages.  
(A) True (B) False

## Chapter 4

59. Pipelining concept is used to increase the \_\_\_\_\_ of the design.  
(A) Power (B) Speed (C) Area (D) Processing
60. CTS increases the speed performance by \_\_\_\_\_.  
(A) Process technology (B) By processing more instructions (C) Logic design  
(D) Pipelining
61. \_\_\_\_\_ is the maximum difference in delays in the arrival times of the clock at two leaf nodes called the launch sequential cell and the capture sequential cell of the clock distribution network.  
(A) Clock Skew (B) Clock jitter (C) Setup (D) Hold
62. *Clock Jitter* is the maximum cycle-to-cycle delays of the clock signal at the same leaf node. It is the variation of clock cycle time at the same leaf node.  
(A) Clock Skew (B) Clock jitter (C) Setup (D) Hold
63. CTS distributes clock to all \_\_\_\_\_ elements with minimum skew.  
(A) Sequential cells (B) Combinational cells (C) Clock buffers (D) Memories
64. \_\_\_\_\_ time is the duration when data must be stable before the arrival of clock edge for a sequential element to latch the data correct.  
(A) Hold (B) Setup (C) Rise (D) Fall
65. \_\_\_\_\_ time is the duration when data must be stable after the arrival of clock edge for a sequential element to latch the data correct.  
(A) Hold (B) Setup (C) Rise (D) Fall
66. For the digital circuit to work properly, in a timing path with combinational logic between two sequential elements, the minimum clock period should be \_\_\_\_\_ the sum of combinational path delay and setup delay.  
(A) less than (B) greater than (C) equal to (D) greater than or equal to
67. The equation  $T_{\min} \geq T_{\text{Comb}} + T_{\text{setup}}$  imposes the limit on \_\_\_\_\_ for a design.  
(A) Maximum frequency of operation (B) Minimum frequency of operation (C)  
Minimum power consumption (D) Number of standard cells

68. \_\_\_\_\_ is the total time taken by the clock signal to reach the clock input of the register.  
(A) Setup time (B) Time Period (C) Clock latency (D) Input delay
69. \_\_\_\_\_ is actual delay seen from the clock generation point to the input pin of register as seen practically.  
(A) Insertion delay (B) Time Period (C) Clock latency (D) Input delay
70. The goal of a CTS in SoC physical design is to achieve clock distribution such that the arrival time of the clock at the farthest sequential cell and the nearest sequential cell must be within the \_\_\_\_\_.  
(A) Skew and latency specifications of the design (B) Meet setup and hold times (C) Highest frequency of operation (D) Minimum delay
71. Size of the clock tree depends on number of buffer stages needed and \_\_\_\_\_.  
(A) Complexity of design (B) Number of buffer cells in library (C) Drive strength of buffer cell (D) Fanout specification of buffer cell
72. Conventional clock tree, Multi Source clock tree, and Clock tree mesh are three types of CTS adopted in designs.  
(A) True (B) False
73. For complex digital designs \_\_\_\_\_ is generated.  
(A) Conventional clock tree (B) Multi Source clock tree (C) Clock tree mesh (D) H-Tree
74. CTS uses \_\_\_\_\_ buffers.  
(A) Ratioed buffers (B) Inverter based (C) Buffers with equal rise and fall times (D) Large Buffer
75. CTS differs from HFN in the type of buffers used.  
(A) True (B) False
76. NDR is abbreviation for \_\_\_\_\_.  
(A) Non default rules (B) Not followed rule (C) Not required to be followed rule
77. HFN path use buffers with relaxed rise and fall times.  
(A) True (B) False
78. Delay between shortest and longest clock distribution paths is called \_\_\_\_\_.  
(A) Useful skew (B) Local skew (C) Global skew (D) Jitter
79. Clock net is strong \_\_\_\_\_ with reference to cross talk.  
(A) Aggressor (B) Victim (C) Driver (D) Load
80. Cross talk in SoC physical design is avoided by \_\_\_\_\_.  
(A) Shielding the nets (B) Decreasing the spacing between two metal tracks (C) Using long parallel nets
81. Buffers which produce output signal with 50:50 duty cycle for input signal with 50:50 duty cycle has switching point at the \_\_\_\_\_ of the transfer curve.  
(A) Center (B) Lower part (C) Higher part (D) {0,0}
82. CTS must be done even for macros having pre-synthesized CTS in physical design.  
(A) True (B) False

83. Most used algorithm for CTS is \_\_\_\_\_.  
(A) RC-based clock tree (B) MMM algorithm (C) X-Tree algorithm (D) H-Tree algorithm
84. CTS optimization is a critical stage of Physical design in deep submicron technologies due to \_\_\_\_\_.  
(A) Feature size (B) Large trees (C) OCV (D) Large number of Flip-Flops
85. Applying NDR is one of the optimization techniques used in CTS.  
(A) True (B) False
86. NDR is also applied to HFN.  
(A) True (B) False
87. CTS routing is done in routing stage of physical design.  
(A) True (B) False
88. Virtual routing means \_\_\_\_\_.  
(A) Routing partially (B) Routing few nets (C) Routing clock only (D) Routing to assess the congestion but not drawing interconnects
89. Resistance of Metal layer is maximum in \_\_\_\_\_ layer, if M4 is higher than M3 is higher than M2 is higher than M1 layers.  
(A) M1 (B) M2 (C) M3 (D) M4
90. Goal of Clock tree synthesis is \_\_\_\_\_.  
(A) To optimize clock signals (B) To create an interesting routing pattern (C) To minimize clock skew and Jitter (D) To connect clock and reset signals
91. Main aim of CTS is \_\_\_\_\_.  
(A) Minimum slack (B) Minimum fan out (C) Minimum slack (D) Minimum skew
92. Better timing is achieved by choosing \_\_\_\_\_ cells.  
(A) HVT cells (B) LVT cells (C) HVT and LVT cell alternatively (D) RVT cells

## Chapter 5

93. Routing is predominantly \_\_\_\_\_ routing.  
(A) Signal (B) Clock (C) Power (D) All interconnections
94. Major challenge of Routing is interconnecting billions of signals and achieving 100% routed signals in complex designs.  
(A) True (B) False
95. Design constraint for routing is mainly the restrictions in the Metal layers.  
(A) True (B) False
96. For routing design layout is divided into hundreds of course regions called \_\_\_\_\_ or \_\_\_\_\_.  
(A) Grids, GCells (B) Grids, Boxes (C) Channels, Grids (D) Gboxes, GCells
97. In any VLSI technology node, if a metal layer M1 supports horizontal routing, metal layer M2 supports vertical signal routing. Maintaining opposite directions for signal routing in alternate layers of metal is to avoid \_\_\_\_\_.  
(A) Avoid signal interference (B) Cross talk (C) Electromigration (D) For no reason

98. Smaller Cells in design layout used for routing the signals are called \_\_\_\_\_.  
 (A) gCells (B) GBoxes (C) SCells (D) Grids
99. Routes go to multiple metal layers through Vias through weighted grid-based wavefront formation.  
 (A) True (B) False
100. The extent to which the design meets its PPA goals is assessed by \_\_\_\_\_ of the design.  
 (A) Design Convergence (B) Timing Closure (C) PPA (D) Power closure
101. Leakage power is inversely proportional to \_\_\_\_\_.  
 (A) Frequency (B) Supply voltage (C) Load capacitance (D) Threshold Voltage
102. Filler cells are added \_\_\_\_\_.  
 (A) Before placement of standard cells (B) After CTS (C) Before detailed Routing (D) After placing standard cells
103. Maximum current density is available in \_\_\_\_\_ file.  
 (A) .lib file (B) .sdc file (C) tf file (D) .def file
104. Large *IR* drop in design layout are caused by \_\_\_\_\_.  
 (A) Hot spots (B) Leakage (C) short circuits (D) Routing congestion
105. *IR* drop increases in \_\_\_\_\_.  
 (A) Increase in metal width (B) Increase in metal length (C) Decrease in metal length (D) Long metal lengths
106. Lee Maze algorithm is used in most routers in PD tools because \_\_\_\_\_.  
 (A) It is easy (B) Finds optimal path between source and destination (C) Of easy integration (D) It is complex
107. Minimum height and width a standard cell can occupy is called \_\_\_\_\_.  
 (A) Grid (B) A tile (C) LVT cell (D) Multi-Vt cell
108. Routing considers *IR* and EM analysis and accordingly optimizes route by one or many techniques like shielding, NDR, spacing between routes, etc., during routing optimization.  
 (A) True (B) False
109. \_\_\_\_\_ is the way to connect the signals passing across restricted metal.  
 (A) Using same layers (B) Skip routing (C) Not routing the signal by throwing error (D) Via ladder
110. Flexible computing resource needed to route the signals in a large scale is termed \_\_\_\_\_.  
 (A) Cloud computing (B) Elastic CPU or Scalable computing (C) Networked computing (D) Load sharing
111. Routers use Spanning tree algorithms like Dijkstra algorithm and Prim's algorithm to find the shortest route for interconnects.  
 (A) True (B) False
- Chapter 6**
112. \_\_\_\_\_ cells are added in the design to address reliability and manufacturability of the design.  
 (A) Special cells (B) Spare cells (C) Guard cells (D) gCells

113. Reliability issues in VLSI chips are analyzed in design stage by Si analysis. Antenna effect, IR analysis, and EM effects.  
(A) True (B) False
114. Well tap cells are added as per the design rules to optimize \_\_\_\_\_.  
(A) Power (B) Speed (C) Area (D) Signal bounce
115. Antenna effect is avoided in designs by adding \_\_\_\_\_ cells.  
(A) Decap cells (B) Corner cell (C) Shielding the net and compliance to Rules  
(D) Antenna diodes
116. Shielding a net, up/downsizing cells are ways to avoid Cross talk issues  
(A) True (B) False
117. Resistance of Metal layers increases from M1 to M7; M1 being lower most and M7 the higher most.  
(A) True (B) False
118. \_\_\_\_\_ are added to avoid latch-up issues in designs.  
(A) Decap cells (B) Diodes (C) Spare cells (D) Well tap cells
119. Physical only cells are called so due to \_\_\_\_\_.  
(A) They lack logic functionality (B) They address manufacturability (C)  
They are smaller than standard cells (D) They address manufacturability
120. \_\_\_\_\_ cells are added to avoid damage of devices at the edge.  
(A) Corner cells (B) Boundary cap or end cap cells (C) Spare cells (D) Diodes
121. Logic cell driven by constant VDD in design layout by \_\_\_\_\_.  
(A) Cap cell (B) Corner cell (C) End cap cell (D) TiHi cell
122. Logic cell is driven by constant GND in design layout by TiHi cell.  
(A) True (B) False
123. \_\_\_\_\_ Cells added to avoid ground bounce or voltage droop in designs.  
(A) DeCap cell (B) Corner cell (C) End cap cell (D) TiHi cell
124. \_\_\_\_\_ Cells are used to correct small issue due to functional or timing after routing.  
(A) Antenna diode (B) Space cell (C) DeCap cell (D) TiHi cell

## Chapter 7

125. LEC and DRC fixing in the design are major activities during physical design stage of tapeout.  
(A) True (B) False
126. Design layout rules are extremely critical for checking the design for manufacturability.  
(A) True (B) False

## Chapter 8

127. Interposer connects multiple dies in 3D-ICs as per the netlist.  
(A) True (B) False
128. Stacked dies are integration technologies of future.  
(A) True (B) False

129. Dies are connected using micro-bumps wherein lower die also has \_\_\_\_\_ through active layers and substrate to metallization on its back surface.  
(A) Vias (B) via ladder (C) TSVs (D) Metal M7
130. The 3D-IC integration connecting 2.5D-IC silicon interposer to enhance further integration to develop high bandwidth, compute intensive solutions using stacked IC. This is called \_\_\_\_\_.  
(A) 2D (B) 2.5D (C) 3D (D) 5.5D

## Answers

### Chapter 1

1. (A)
2. (C)
3. (B) chiplets are functional dies used to develop large systems by stacking in 3D-ICs
4. (D)
5. (C)
6. (A)
7. (B)
8. (A)
9. (B)
10. (B)
11. (B)
12. (A)
13. (B)
14. (A)
15. (D)

### Chapter 2

16. (A)
17. (B)
18. (B)
19. (A)
20. (B)
21. (C)
22. (B)
23. (A)
24. (B)
25. (B)
26. (B)
27. (B)
28. (A)

29. (C)

30. (D)

31. (A)

### **Chapter 3**

32. (B) Floor planning is very important as QOR of a design and design convergence for PPA goals depend on quality of Floor plan

33. (C)

34. (D)

35. (D)

36. (A)

37. (C)

38. (B) is required to generate PR boundary for design layout during physical design

39. (A)

40. (B)

41. (C)

42. (C)

43. (D)

44. (D)

45. (C)

46. (A)

47. (A)

48. (A)

49. (C)

50. (A)

51. (A)

52. (B)

53. (B)

54. (B)

55. (D)

56. (D)

57. (A)

58. (B)

### **Chapter 4**

59. (B)

60. (C)

61. (A)

62. (B)

63. (A)

64. (B)

65. (A)

66. (D)

67. (A)

68. (C)

- 69. (A)
- 70. (A)
- 71. (D)
- 72. (A)
- 73. (C)
- 74. (C)
- 75. (A)
- 76. (A)
- 77. (A)
- 78. (C)
- 79. (A)
- 80. (A)
- 81. (A)
- 82. (B)
- 83. (D)
- 84. (C)
- 85. (A)
- 86. (B)
- 87. (B)
- 88. (D)
- 89. (A)
- 90. (C)
- 91. (D)
- 92. (B)

**Chapter 5**

- 93. (A)
- 94. (A)
- 95. (A)
- 96. (D)
- 97. (C)
- 98. (C)
- 99. (A)
- 100. (C)
- 101. (D)
- 102. (C)
- 103. (C)
- 104. (D)
- 105. (B)
- 106. (B)
- 107. (B)
- 108. (A)
- 109. (D)
- 110. (B)
- 111. (A)

**Chapter 6**

- 112. (A)
- 113. (A)
- 114. (C)
- 115. (C) and (D)
- 116. (A)
- 117. (B)
- 118. (D)
- 119. (A)
- 120. (B)
- 121. (D)
- 122. (B)
- 123. (A)
- 124. (C)

**Chapter 7**

- 125. (A)
- 126. (C)

**Chapter 8**

- 127. (A)
- 128. (A)
- 129. (C)
- 130. (D)

# Index

## A

Abutted, non-Abutted, 141  
A clock tree in CTS, 142  
Aggressor cell, 88  
Aggressor net, 77  
Analog, 124  
Analog blocks process  
    design, 5  
Analog cores in SoC process, 5  
Antenna affects transistors, 85  
Antenna diodes, 140  
Antenna effect, 145  
Applications demand system, 19  
Area, 145  
Artificial intelligence (AI), 133–135  
Aspect ratio, 140  
A tile, 144

## B

Backside power delivery, 131  
Ball grid arrays, 124  
Boundary cap/end cap cells, 145  
Boundary cells, 95  
Buffers with equal rise and fall times, 142  
Buried interconnects, 131  
Buried oxide trench, 85

## C

Checkerboard pattern, 93  
Chiplets, 124, 137  
Chosen technology, 141  
Clock Jitter, 53

Clock latency, 142  
Clock Mesh, 59  
Clock net, 140  
Clock routing, 57  
Clock signal, 49, 52  
Clock skew, 53  
Clock tree mesh, 142  
Clock tree optimization, 63  
Clock tree synthesis (CTS), 28, 47, 51, 53, 58,  
    59, 61, 139, 142, 143  
    clock latency, 56  
    and clock mesh-based trees, 59  
    clock width, 57  
    design layout, 59  
    insertion delay, 57  
    routing, 143  
Cloud-based AI, 4  
CMOS cell, 82  
CMOS compatible technologies, 137  
CMOS etching process, 124  
Coarse placement, 39  
Coarse routing, 29  
Controlled collapse chip connection (C4)  
    bumps, 124  
Conventional clock tree, 142  
Core design, IO design, 138  
Core height, 41  
Core limited, 139  
Core utilization, 41  
Course placement, 27  
Cross talk effect, 87  
Cross talk noise, 77  
Custom design flow, 6  
Custom design style, 10

**D**

- Data processing, 54
- Decap cells, 97, 99, 145
  - CMOS gate, 98
  - layout, 97, 98
  - voltage droop, 98
- Deep submicron processes, 92
- DEF file, 138
- Default clock routing rules
  - (NDR), 28
- Design for manufacturability (DFM)
  - design rules, 81
  - functioning, 81
  - latch-up, 83
  - physical design, 81
  - technological problems, 81
- Design layout rules, 145
- Design macros, 8
- Design modeling, 105
- Design netlist file, 47
- Design process, 18
- Design Quality of Results (QOR), 62
- Design readiness
  - elements, 68
  - static timing, 68
- Design routing, 70
  - CTS, 67
  - design, 67
  - input-output, 65
  - metal layers, 65
  - signals, 66
- Design rule check (DRC), 29
  - ERC, LVS, 139
  - flow, 118
  - violations and fixes, 116
  - waiver, 120
- Design rule violation (DRV), 114
  - design layout rules, 119
  - steps, 118
- Design sign-offs, 29
- Design technique, 6
- Detailed routing, 29
- Die size, 139
- Digital cores, 7
- Digital designs, 49
- Digital system, 50
- Dijkstra algorithm, 144
- Diode protection mode, 86
- Driver, 142
- Dynamic IR analysis, 112

**E**

- EDA tools, 18, 47, 67, 105, 126, 127
  - automatic and smart, 19
  - vendors, 47, 120
- Elastic CPU/scalable computing, 144
- Electrical rule check (ERC), 112
- Electromagnetic cross talk, 86
- Electromagnetic issue, 77
- Electromigration (EM), 91, 110, 143
  - effects, 145
  - rules, 112
- Electronic change order flow (ECO), 39
  - file, 104, 105
  - flow, 104
- Electronic industry, 18
- Electrostatic discharge, 83
- End cap cell/boundary cell, 94

**F**

- Fabricating CMOS devices, 88
- Fabrication technologies, 65
- Fanout specification of buffer cell, 142
- Field programmable gate arrays (FPGA), 10
- File transfer protocol (FTP) process, 119
- Filler cells, 144
  - 5.5 D, 146
  - 5.5 D-IC, 124
- Floor plan, Placement, CTS, Routing, 138
- Floor planning, 34, 36, 41, 139
- Floor planning stage, 138
- Foundry vendors, 138
- Full custom design techniques, 137
- Functional blocks, 22
- Functional noise, 77

**G**

- Gate array design, 10
- Gboxes/GCells, 143
- Global routing, 29, 68, 76
- Global skew, 142
- Ground bounce, 145

**H**

- Hadlock algorithm, 72
- Hard blockages, 141
- Heterogeneous multi-die integration, 131
- Heterogenous 3D technology, 123
- HFN synthesis, 139

High fanout net (HFN), 28, 39, 139, 140, 142, 143  
Hold time, 141  
Homogenous 3D technologies, 123  
H-Tree algorithm, 143

**I**

Input-output (IO) pads, 22, 42, 140  
Insertion delay, 142  
Interconnecting functional blocks, 65  
Interconnects, 24  
International roadmap of devices and systems (IRDS), 18  
IO pin placement script file, 42  
IO signals, 141  
IR analysis, 111, 145  
IR and EM analysis, 144  
IR drop, 144

**L**

Large digital cores, 137  
Latch-up in standard cell, 81  
Layout *vs.* Schematic (LVS), 30, 107  
    errors, 108  
    flow, 110  
    rule deck, 108  
    tools, 109  
LEC and DRC fixing, 145  
Lee Maze algorithm, 72, 144  
Lee's algorithm, 72  
Liberty format, 25  
Library exchange format (LEF), 44  
Lithography process verification (LPV), 132  
Logical equivalence check (LEC), 30  
Logic design, 141  
Logic equivalence, 107  
Logic equivalence checking, 107  
Logic functionality, 145  
LVT cells, 63, 143

**M**

Machine learning (ML), 131, 133–135  
Macro placement, 44  
Macros, 138  
Manufacturing process limitations, 138  
Maximum frequency of operation, 141  
Maze routers, 74  
Maze routing algorithm, 68  
Memory macro, 7  
MEMS (sensors), 124

Metal layer, 143, 145  
Micro bumps, 124  
Minimum delay constraint, 56  
Minimum skew, 143  
Monolithic IC, 121  
Moore's law, 2, 47  
MOS transistor, 97  
Multi-chip modules (MCM), 124  
Multiple chips, 124  
Multiple optimal paths, 73

**N**

Netlist, 137  
Netlist with parasitics, 139  
Non default rules (NDR), 60, 63, 67, 139, 142, 143  
NP-complete (NPC) problems, 18

**O**

On-chip variation (OCV), 63, 143  
Only buffers and inverters, 138  
Optical proximity correction (OPC), 119

**P**

Package-on-package schemes, 122  
Pad limited, 139  
Parasitic extraction flow, 103  
Parasitic extractions, 103  
Parasitic netlist, 103  
Physical design, 140  
    file formats, 16–17  
    flow, 22, 105  
    tools, 30, 105  
    verification, 101, 120  
Pipeline stage, 51  
Pipelining, 50  
Pitch matching process, 40  
Placed design layout, 46  
Placement, 138  
    blockages, 39  
    optimization, 28  
    violations, 27  
PNPN device, 83  
Post placement stage, 140  
Power, 140  
    ground signals, 68  
    mesh, 26  
    network synthesis, 26  
    optimization, 63  
    planning, 26  
    routing, 67

Power performance and area (PPA)

    goals, 19, 144

Pre placement stage, 27

Preplacement, 138

Preplacement stage, 140

Prim's algorithm, 144

Process design kit (PDK)

    EDA tools, 15

    libraries, 15

Processor speed, 51

## **Q**

QoR goals, 134

Quality of results (QOR), 76

## **R**

Rails, 140

Ratio of die height to die width, 140

Redistribution layer (RDL), 123, 125

RFCMOS, 124

Ring, 140

Route aware CTS, 132–135

Router engines, 78

Routers, 78, 144

Routing, 67, 72, 143

    congestion, 144

    problem, 71

    process, 79

Routing algorithms

    Lee's algorithm, 72

    Maze router, 72

    router, 71

## **S**

Scalable CPU systems, 4

Scan chain reordering, 27, 39, 139

Scan chains, 139

SCells, 144

Semiconductor package design

    methodologies, 122

Semiconductor process technology, 47

Semiconductor systems and physical design

    post-route stage, 132–135

    pre-route stage, 132–135

    route aware CTS, 132–135

    trends, 135

Semi-custom design style, 10

Sequential cells, 50, 141

Setup time, 141

Shielding a net, 145

Shielding technique, 88

Shielding the nets, 142

Shielding wires, 111

Signal, 143

    nets, 139

    routers, 75

    routes, 68

    routing, 67, 141

Silicon in package, 122

Silicon interposer, 125

Skew and latency specifications of the design, 142

Smaller manageable blocks, 139

Smart Architectures, 137

SoC design styles

    EDA tools, 10, 11

    EPROM, 10

SoC packaging

    fabricated SoC die, 121

    functions, 121

    in product design, 121

    trends, 121–129

SoC physical design, 34

Soft blockages, 39

Spanning tree algorithms, 144

Spare cells, 99, 145

    inputs, 100

Special cells, 144

Speed, 141

Stacked die

    chiplets, 124

    face-face, 123

    face to back, 124

    5.5 D-IC, 124

    innovation, 122

    RDL, 125

    silicon interposer, 125

    3D IC (*see* 3D IC)

Stacked dies, 145

Standard cell, 10, 25

Standard delay format (SDF), 138

Static timing analysis (STA), 101

    quality, 103

    sign offs, 103

Stick diagram, 13, 137, 138

Stripes, 140

Super vias, 131

Synchronization, 51

System in package (SiP), 122, 124

System on chip (SoC), 1

    abstractions, 8

    architectures, 4

    constituents, 5

    current, 10

    design, 3, 8, 36

- design database, 31  
design flow, 8, 9  
design partitioning, 31  
digital cores, 7  
memory cores, 7  
physical design, 11, 17  
3D-IC, 5  
trends, 2  
types, 6  
typical, 5  
System-level technologies, 135
- T**
- Tap cells, 93, 145  
Terminal count, 139  
Terminal pitch, 139  
tf file, 144  
3-D chips, 128  
3D IC, 145  
    ball grid arrays, 124  
    bumps and balls, 124  
    C4 bumps, 124  
    concept, 122, 123  
    constituents, 124, 125  
    design, 128  
    designs, 126  
    EDA tools, 126, 127  
    heterogenous 3D technology, 123  
    homogenous 3D technologies, 123  
    integration schemes, 123  
    micro bumps, 124  
    physical design toolset, 126  
    structure, 122, 123  
    TSV, 124, 125  
3D SoCs, 126  
3D VLSI, 131  
Threshold voltage, 144  
Through silicon vias (TSV), 123–125, 146  
Tie cells, 95  
Tie-high cell, 95  
TiHi cell, 145  
Tihi cell layout, 97  
Tiles, 139  
Timing data, 25  
Timing-driven global routing, 76
- Timing-driven routing, 76  
Timing window analysis, 87  
Track assignment, 69  
Transistors, 92  
Trends in packaging  
    evolution, 122  
    SiP, 122  
    stacked die (*see* Stacked die)  
    VLSI technology, 121
- 2D designs, 126  
2.5D designs, 126  
2.5D-IC, 123, 146  
2.5D technologies, 124  
Two step synthesis, 140  
Two-step syntheses, 38
- V**
- Vertical dimensions, 131  
Vertical gate all around (VGAA), 131  
Via ladder, 144, 146  
Victim net, 77  
Virtual routing, 143  
VLSI, 134  
    chips, 145  
    design, 137  
    SoC designs, 48  
    SoC die, 121  
    technology, 121  
VLSI SoC design flow, 14  
    PPA convergence, 132  
Voltage droop, 145
- W**
- Wafer-scale large chip, 4  
Wavefront, 72  
Well proximity effect (WPE), 92  
Well tap cells, 93, 145  
Well tap layout, 93  
WI-FI SoC, 2  
Wire load model, 25
- Z**
- Zero wire load model, 25