


POLITECNICO DI TORINO

Master Degree course in NANOTECHNOLOGIES FOR ICTs
(NANOTECNOLOGIE PER LE ICT)

Master Degree Thesis

Integrated circuit Back-End of line analysis and modeling for future node pathfinding

Supervisors
Prof. Guido Masera
Ph.D. Anita Farokhnejad

Candidate
Francesco Dell'Atti

Academic Year 2022-2023

Acknowledgements

To curiosity, to embrace a better life,
To my lovely family and close friends,
Thanks to all the minds I met along this special path,

Let's uphold Gordon Moore's vision.

Contents

List of Figures

Chapter 1

Introduction

Downscaling transistors over thousand folds have been the working horse for miniaturization and improving power performance area (PPA) over decades. However, this trend seems to be limited by other factors than only the size of front end of line (FEOL) elements [5]. By moving from one technology node to the next one the nanoscale interconnects, the so-called back end of line (BEOL), shrinks as well. Reduction in width of the lines (CD of the line) results in resistance increment. By reducing the spacing between metal lines capacitance increases. Hence, delay of BEOL in each node further increased and negatively impacts the performance of the design [3].
The first step to tackle the aforementioned issues is to gain a solid grasp of the nature problem. To do so, in-depth understanding of routing and its dependency on design criteria is necessary.

Ring Oscillator simulations are widely used in industry for performance evaluation or comparison of different logic and interconnect technologies. Electronic design automation (EDA) software are powerful tools which are essential to design a chip [1]. EDA is also used for Place and Route (PnR) purposes to simulate if and how all electronic components of a circuit can get connected fulfilling both space and speed restrictions. Considering the targeted frequency for an arbitrary design, there will be paths that are slower. Those are known as critical paths (CPs) or paths with negative slack. Slack refers to the difference between the desired arrival time of a signal and the actual time the signal has reached the destination. In each design, CPs limit the chip performance. These paths show a signature of different designs and routing criteria. It is expected that by analyzing CPs, a pattern can be found that shows and also predicts the correlation between design ground rules and characteristics of CPs. The latter are also used to construct the RO so that its frequency and power shift best mimics the actual shift obtained when a new technology is employed. Thus they represent a compass to drive technology to achieve the best optimization result.
Results of PnR include all information about physical and electrical details about each path such as path delay, metal layers used in a path, number of vias, type of instances and many more. However, accessing that information is not straightforward. To extract the required data an extraction flow is required. An extraction flow refers to a script that enables accessing specific types of information from the PnR database.

The aim of this project is to improve, extend and increase the features of the existing extraction flow which captures and connects the exact topology of paths from Place and Route database to other circuitry features and physical parameters of a given chip design, node, and technology.
Integrating an Elmore delay calculator along the flow, it is possible to compute the delay due to interconnects and so the R and C contributions, to the total path delay, of the metal layers and the vias in between. Enriching the Elmore delay calculator with structural and electrical statistical analysis, a novel mapping method called enhanced ring oscillator (eRO) is proposed.
To understand how the BEOL influences and determines cell delays, a comprehensive study, and analysis of the liberty (.lib) files is undertaken. By conceptualizing the propagation of slew along the chain of cells as a feedback loop, specific scaling factors are formulated and calculated. These scaling factors serve as crucial connectors, bridging cell delays to BEOL and FEOL variations.
Being part of a long-term plan, the flow represents the bearing structure for the development of an automated model which enables the possibility to predict the power-performance response to new disruptive technologies such as new boosters, new interconnect concepts or even new devices. This model can then be utilized to offer the required guidance for technological advancements.

In the following, an overview of the basic information, software tools, and used program languages is provided to face the high specificity of the topic. The existing extraction flow is introduced as well as the Elmore delay computation. Afterward, the improvements in the existing extraction flow are described. Results are discussed and a sanity check is done by using other features of the PnR tool as a reference to validate the work.
Next, the final table-like database which is constructed along the flow is used as a high-performance database. From it is then possible to perform statistical analysis which allows capturing correlation between different design-dependant elements and the trade-offs the EDA tool considers during routing optimization. These data are then mapped into the enhanced RO (eRO) utilized as a rapid instrument for accurate benchmarking.
Finally, the cell delay dependency on signals and circuitry characteristics is investigated, and a methodology to predict how these characteristics impact the overall achieved frequency of a design is proposed.

This work has been carried out under the supervision of the Physical Design Research (PDR) group at Imec within 6 months. Considering the level of complexity and uniqueness of this project, the involvement of a tutor and teamwork with the other members of the group was essential. The knowledge and information which have been transferred by the team were useful to align with the work to be done and the future perspective. On this background is superimposed the understanding of the required tools needed to perform the job, such as the PnR tool from Cadence, the Unix environment to access Imec servers, and the basics of some program languages such as Tcl and Python which are essential to developing an automated extraction flow.

Chapter 2

Background

To establish a solid foundation for the project and develop a comprehensive model, it is essential to begin with a literature review and an exploration of the VLSI theory underlying the current state of the art. This chapter aims to provide an understanding of the project's objectives and to establish a base for further development. Additionally, it explains the key concepts addressed in the project and provides an overview of the work that has already been done. This discussion serves to identify the limitations of previous extraction flows and highlight the need for further expansion and advancement in the field.

2.1 Back-End of line

BEOL refers to the network of nanoscale wires that establish connections between billions of logic gates within an integrated circuit. These wires are arranged in multiple layers of metal stacked on top of each other, with an oxide layer serving as a separator between them. Each metal layer has a unidirectional flow, allowing signals to travel vertically or horizontally within that layer. In modern technology nodes, low-resistivity materials, referred to as low-κ materials, are employed to isolate metal lines within a layer, thereby minimizing the intra-layer capacitance of the lines [12]. Vias, which are metal connections, are used to link wires from one layer to another, either above or below. The width of metal lines varies between different layers, with lower metal layers (Mx layers, typically from layer 0 to 4) having narrower widths compared to upper metal layers (My and Mz). The distance between the center of a metal line and the center of its adjacent line within the same layer is known as the metal pitch.

When scaling down to smaller nodes in terms of the back-end-of-line (BEOL) perspective, it involves adopting a tighter metal pitch, narrower metal lines, and reduced spacing between them. This downsizing process has two main effects. On one hand, it increases the resistance and capacitance of nanoscale wires, resulting in higher BEOL delay. On the other hand, a smaller metal pitch provides more routing resources, allowing for a greater number of metal tracks to fit within the same design area. Consequently, reducing the pitch of a metal layer used in the cell design enables further scalability and denser layouts, particularly in the case of M0 (Mint), M1, or even M2 in advanced nodes. As complexity

increases in each node, there is a higher utilization of standard elements or cells, while simultaneously maintaining or reducing the chip's overall area. Consequently, connecting all these circuit elements becomes increasingly challenging, necessitating adherence to both physical and electrical restrictions, such as area and frequency.

A path refers to a connection between the output of a register to the input of the next register. Usually, several standard cells interpose between two registers. The set of interconnects between each of two consequent instances of a path is referred to as a net. A net contains all the wires connecting the source instance with all the load instances. A path is called critical when its delay exceeds a predefined time criteria which strongly depends on the clock target frequency. Slack refers to the difference between the desired arrival time of a signal and the actual time the signal has reached the destination and it is computed by 4.1, as shown later. CPs limit the performance of the whole design. In order to achieve a functional design, all the slacks must be positive so that all the paths fulfill their actual time constraints. Typically, in a new design, there are several critical paths with negative slack. They need to be speeded up to meet the required timing. A critical path can be affected at four main levels [15]:

- The architectural/micro architectural level
- The logic level
- The circuit levels
- The layout levels

In the context of BEOL, the term "fan-out" (FO) refers to the number of loads that are connected to the output of a cell. Essentially, it represents the number of gates that a cell drives. The wire paths connecting the sink to the load towards which the signal is intended to travel is called 'main' net part. The wire paths connecting the 'main' net part to one or more loads are called branches. The delay of a path is influenced by its nets, and the delay of a net is determined by various factors in both BEOL and FEOL elements. These factors include the length of the net, metal distribution, number of vias, fan–out (FO), starting positions of branching, load of each branch, load of the main sink where the signal is intended to travel, and the drive strength of the net's source. As the wire resistance and capacitance increase with length, the wire delay grows quadratically with length. Repeaters, also known as buffers, are employed to reduce the net length, thereby enabling the overall delay to become a linear function of the length.
Understanding these factors highlights the importance of considering net topology when constructing a reliable model for BEOL. This approach allows for an accurate assessment of net characteristics in relation to the aforementioned factors.
By tracking the net topology, it becomes possible to perform Elmore delay computation, which serves two purposes: firstly, to estimate the net delay, and secondly, to estimate the contributions of resistance (R) and capacitance (C) of all the metal layers, vias and pins to that delay.

The subsequent paragraph presents the PnR flow, which combines and enhances all these parameters to attain a design that is ready for tape-out.

2.2 Place and Route flow

ASIC digital chips require a significant level of automation to achieve cost-effectiveness and expedite the overall development process. This high level of automation is attained by integrating EDA (Electronic Design Automation) tools, TCAD (Technology Computer-Aided Design) tools, and simulation tools into a unified automated flow, commonly known as semi-custom digital flow.

The part of semi-custom flow which here is exploited, begins with a Register Transfer Level (RTL) description of the CPU's functionalities. Together with cell timing libraries, cell layout geometries, and Back-End-of-Line (BEOL) stack geometries, this information is fed into a synthesis tool to generate a netlist that meets various constraints, such as target frequency and target utilization. The target frequency determines the clock frequency, while the target utilization specifies the percentage of area occupied by cells.

The synthesis process involves several steps: first, the functionality is transformed into generic gates, then these gates are mapped to standard cells, and finally, buffers are inserted to speed up the signals and meet the timing constraints. Since logical synthesis lacks physical awareness, physical synthesis comes into play. It includes wire load models and additional steps that enable interaction between the synthesis tool and the PnR tool. This ensures that the netlist is optimized with physical considerations in mind.

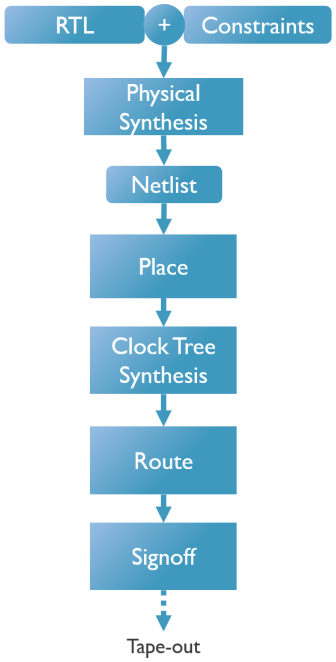

Figure 2.1. PnR flow, from RTL to signoff.

With the netlist and technology files containing BEOL stack characteristics, the PnR tool performs the following steps: placement, clock tree synthesis, routing, and signoff.

Buffer insertion is carried out between each step to improve timing performance. The PnR steps rely on heuristics, and multiple optimization algorithms are utilized to minimize wire lengths and achieve optimal performance with reduced power consumption [15].

In summary, the digital semi-custom flow shown in figure 2.1 involves transforming RTL functionalities into a netlist through logical synthesis, which is then optimized with physical considerations using the PnR tool. This process aims to achieve the desired target frequency and area utilization.

The following paragraph is dedicated to the specific PnR tool used to generate and collect the data discussed in this thesis.

2.2.1 Innovus Implementation System - Cadence

Innovus, a Place-and-Route (PnR) tool provided by Cadence, is accompanied by its user interface called Stylus [1]. By utilizing a recursive mechanism, Innovus optimizes chip routing to achieve the best possible solution that satisfies all time constraints and ensures connectivity between instances, standard cells, and registers. Various optimization algorithms are performed, relying on cost functions to achieve the optimal balance between power and performance while considering physical constraints such as wiring resources and Design Rule Check (DRC) rules [6]. The PnR optimization process depends on the target period (which defines the desired frequency) and the target utilization area (which determines the percentage of chip surface area dedicated to the Front-End-Of-Line, or FEOL). Due to the time-consuming nature of the optimization process, typically taking weeks depending on design complexity and size, there is a need to develop a new model that can efficiently preselect a subset of designs for the tool to run on. In other words, this new model should rapidly predict, for example, which are the most promising new technologies, or how the achieved frequency evolves when some characteristics of the FEOL or BEOL stack are changing.

After the completion of chip routing using the PnR tool, specific commands like *report-timing* or *report-wire-paths* can be used to extract reports from the tool's database. The *report-timing* command, for instance, provides a table of path information including pins, instances, cell types, delays, slack, clock frequency, slew, and more. The output of the report-timing command is typically presented in a tabular format, as shown in Figure 2.2 for reference. This report provides valuable insights into the timing characteristics of various paths based on the given number of inputs.

The delay of a signal along each path is calculated at the timing pins within the VLSI design. In VLSI, the timing relationship between two pins is defined as a timing arc, which represents the signal propagation through logic gates or nets. By knowing the timing pin names, it is possible to extract the names of all the nets and wires associated with each critical path. This allows for the retrieval of net attributes such as fan-out (FO) or delay from the net name. These attributes are stored within the Innovus database as part of the net object. Furthermore, electrical information such as total net resistance or capacitance can be extracted, serving as references for subsequent sanity checks.

Path 1: VIOLATED (ns) Clock Gating Setup Check with Pin <pin name>
Startpoint: (R) <pin name>
Clock: (R) clk
Endpoint: (R) <pin name>
Clock: (R) clk

	Capture	Launch
Clock Edge:+	(ns)	(ns)
Src Latency:+	- (ns)	- (ns)
Net Latency:+	(ns) (P)	(ns) (P)
Arrival:=	(ns)	(ns)
Uncertainty:-	(ns)	
Cppr Adjust:+	0.000	
Required Time:=	(ns)	
Launch Clock:=	(ns)	
Data Path:+	(ns)	
Slack:=	(ns)	

# Timing Point	Cell	Trans (ns)	Load (pf)	Delay (ns)	Required Time (ns)	Arrival (ns)	Incr Delay	Aocv Stage Count	User Derate	Instance	Instance Location
<clock pin name>	(arrival)	(ns)	0.024	-	(ns)	(ns)	-	-	-	<instance name>	(73.63,69.84)
<pin name>	DFFD1	(ns)	0.002	0.028	(ns)	(ns)	0.000	-	1.000	<instance name>	(73.63,69.84)
<pin name>	BUFFD8	(ns)	0.007	0.016	(ns)	(ns)	0.000	-	1.000	<instance name>	(69.05,70.20)
<pin name>	XOR2D1	(ns)	0.000	0.009	(ns)	(ns)	0.000	-	1.000	<instance name>	(58.55,70.02)
<pin name>	DFFD1	(ns)	0.002	0.028	(ns)	(ns)	0.000	-	1.000	<instance name>	(74.63,68.84)

Figure 2.2. Report-timing output format.

2.3 Ring Oscillator benchmark

RO simulations are commonly employed to study circuit behavior by modifying BEOL or FEOL elements [5]. Instead of simulating the entire complex design repeatedly to assess minor modifications, a simplified equivalent circuit is used to capture the impact of the implemented change. A RO circuit consists of an odd number of stages, with each stage represented by a logic cell, typically an inverter, and an RC load representing the metal interconnects. The RC load can be constructed in various ways, including representing all the metal layers used in a real design, where each layer is modeled by an equivalent RC model constructed in different ways [10]. The choice of an equivalent model influences the Spice simulation results, particularly the cell and interconnects delays and steady-state frequency at which the system oscillates.

Calling d the stage delay and n the number of stages in the RO chain, the oscillating frequency of a chain of inverters is given by [8]:

$$f = \frac{1}{2dn} \tag{2.1}$$

When an RC load is inserted between each inverter, d represents the stage delay, thus the sum of the cell and RC load delays, representing the delay from the input of one cell to the input of the next cell.

To evaluate the impact of BEOL on core performance, a 23-stage RO circuit (refer to Figure 2.3) is designed. Based on statistics obtained from the previous extraction flow

for PnR data of a given RTL design, the number of stages is selected as the odd number closest to the average logic depth of all critical paths in the design, and an equivalent π shape BEOL load is designed as illustrated in Figure 2.4.
This RC load is then incorporated into the RO circuit between each stage to mimic the overall BEOL of the chip [5]. It is assumed that each inverter has a fan-out (FO) of three, meaning that the output of each inverter is connected to the input of three other inverters. Additionally, it is assumed that the branching occurs at the end of the net, which represents the worst-case scenario as it results in the highest net resistance up to the branching point, and subsequently the biggest path delay.

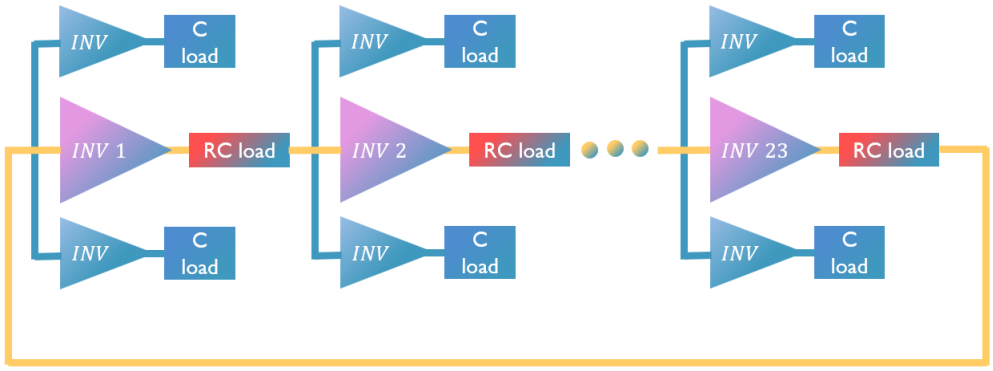

Figure 2.3. 23–stages RO designed to evaluate the impact of the BEOL on circuit performances.

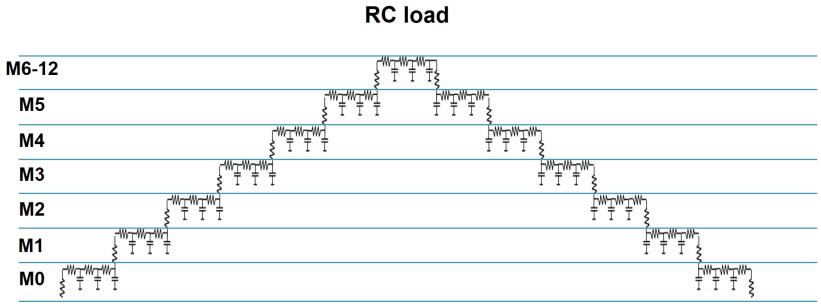

Figure 2.4. Considering the length of each metal layer, their resistance, and capacitance, an equivalent π shape RC load is designed to mimic the BEOL of the whole circuit.

Through repeated simulations and by varying the branching points, it was observed that the starting position of the branch, when deviating from the main path, has a significant impact on the results. However, the previous extraction flow lacked the ability to provide information regarding this aspect because it relied on simple built-in commands capable of extracting general net statistics. These commands did not distinguish between the branching metals and those belonging to the 'main' net part. As a result, the metal distribution obtained from the old flow was a combination of all metals in both the main

path and the branches. Consequently, the values used in the RO model for net length were inaccurate. For example, the extracted data showed an average net length of $0.5\mu m$, with 20% ($0.1\mu m$) attributed to the M2 line. However, this value did not pertain to the M2 length of the main part of the net but rather included all the branches as well.

Given these observations, it became evident that a new extraction flow was necessary to enhance the model and obtain meaningful and reliable estimations.
The subsequent section provides a detailed explanation of how the new flow was developed, along with the challenges encountered along the way to achieve the desired goal.

2.4 Net topology extraction flow

In order to extract the topology of the nets of critical paths (CPs) from a design, a custom automated script was developed. This script utilizes the *report-wire-path* command to traverse the net and construct the net topology. By analyzing and comparing all the net parts with the 'main' net part, the script identifies the branching points of the net in correspondence with a mismatch in the metal or via sequence. This mechanism used to reconstruct the net topology within the script operates similarly to a Depth-First Search (DFS) algorithm. The resulting net topology is then printed in a nested format and saved in a .txt file. Together with the net topology, many other cell, net, or path related information are extracted.

Differentiating between the branching metals and those belonging to the main path is crucial for the subsequent Elmore delay computation, which will be discussed in the section 2.5. It is worth noting that the script's traversal algorithm for recognizing branching points aligns with the algorithm used for performing the Elmore delay computation. This allows for a relatively straightforward implementation of the Elmore delay calculation within the flow.

2.5 Elmore delay

The chip interconnections in integrated circuits are commonly modeled as lumped or distributed RC circuits. When computing the delay of these RC networks, it is important to strike a balance between high accuracy and computational efficiency, particularly in modern ICs with billions of transistors. The use of highly accurate methods can be impractical.
For tree-structured networks that lack resistive loops, such as signal nets within an integrated circuit, the Elmore delay metric is often used to approximate the signal travel time [4]. The Elmore delay metric leverages the first moment of the impulse response (or the first-order time constant) of the RC distributed network, which limits its accuracy to this characteristic [11].
While the Elmore delay metric is most accurate for step-like input signals, it provides a simple analytical function that can be easily integrated into automation tools for digital or analog design. This integration allows for efficient estimation while maintaining a high degree of fidelity [2]. By employing the Elmore delay model to estimate path delays, simulation tools can prioritize critical paths that define the upper bound speed

of the circuit. This approach enables more exhaustive simulations to be performed on these critical paths while reducing the computational burden associated with non-critical paths.
When dealing with a complex network consisting of numerous capacitors and resistors, accurately describing its behavior becomes exceedingly complex. This requires a set of differential equations that consider the network's many time constants, poles, and zeros, which can only be solved through a comprehensive SPICE simulation [7]. In such scenarios, the Elmore delay model offers significant advantages.

To calculate the propagation delay from a source cell output to a specific branching sink, the approach involves summing up the products of capacitance and resistance in a series of RC segments. Each capacitance C_i at the i_{th} node is multiplied by the total sum of resistances R_{si} from the source to that node 2.2. In simpler terms, the delay of each segment is determined by multiplying the capacitance by the resistance upstream.

$$\tau_{\text{Elmore}} = \sum_i R_{si} C_i \tag{2.2}$$

In the example 2.5 is analyzed a simple net where the Elmore delay formula is applied. It

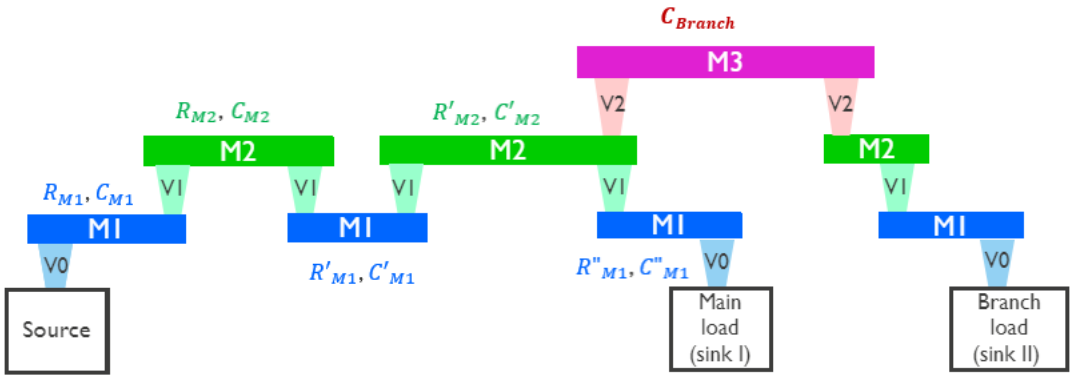

Figure 2.5. Topology example of a net with FO = 2.

is a resistor-capacitor tree with a single source node, where all capacitors are referenced to ground. This network also does not contain any resistive loops. Notably, a distinct resistive path exists between the source and any other sink node within these tree-like topologies.
To quantify the resistance associated with the net, denoted as main net-part resistance, we can refer to the resistance between the source and the main sink. This resistance is equal to 2.3.

$$R_{\text{main}} = R_{V0} + R_{M1} + R_{V1} + R_{M2} + R_{V1} + R'_{M1} + R_{V1} + R'_{M2} + R_{V1} + R''_{M1} + R_{V0} \tag{2.3}$$

Assuming that a step input is applied at the source node, and all other nodes in the network are initially discharged to ground, the Elmore delay at the main load (sink 1)

node is calculated using 2.4. This equation provides a means to determine the Elmore delay for each specific node in the network.

$$\begin{aligned}\tau_{\text{Elmore}} =&(R_{V0}+R_{M1})C_{M1}+(R_{V0}+R_{M1}+R_{V1}+R_{M2})C_{M2}\\&+(R_{V0}+R_{M1}+R_{V1}+R_{M2}+R_{V1}+R'_{M1})C'_{M1}\\&+(R_{V0}+R_{V1}+R_{M2}+R_{V1}+R'_{M1}+R_{V1}+R'_{M2})(C'_{M2}+C_{\text{Branch}})\\&+(R_{V0}+R_{M1}+R_{V1}+R_{M2}+R_{V1}+R'_{M1}+R_{V1}+R'_{M2}\\&+R_{V1}+R''_{M1})C''_{M1}\\&+(R_{V0}+R_{M1}+R_{V1}+R_{M2}+R_{V1}+R'_{M1}+R_{V1}+R'_{M2}\\&+R_{V1}+R''_{M1}+R_{V0})C^{\text{pin}}_{\text{Main load}}\end{aligned} \tag{2.4}$$

By observing that R=rL and C=cL, the delay of a wire is a quadratic function of its length. This implies that doubling the length of the wire results in a quadrupling of its delay. As a consequence, in modern integrated circuits (ICs), where cell delays are continuously decreasing, the delay of long wires has started to become a dominant factor. In highly compact CPUs, where wiring resources are not a limiting factor, routing optimization algorithms tend to prioritize shorter wires. This ensures that the delay introduced by the wires is comparable to the delay of the individual cells, resulting in a well-balanced design.
However, in larger designs that integrate multiple macros, such as CPUs integrated with memory blocks, the data paths can be significantly longer, with wire lengths ranging in millimeters. In such cases, the delay caused by these long wires dramatically outweighs the delay of the individual cells, leading to a significant impact on the overall path delay.

In this study, the metal segments are characterized using the L-shape model, where the resistance (R) of the metal is followed by its capacitance (C). However, alternative models such as the T-segments or the π-shaped segments can also be used to represent the metal segments. In the T-segment model, the resistance is divided into two parts, and the capacitance is placed in between them. Similarly, in the π-shaped segment model, the capacitance is divided into two parts, with the resistance in between them. These alternative models result in delay estimates that are less pessimistic compared to the L-shape model [10].

Chapter 3

Leveraging net topology extraction flow - Elmore delay computation

A signoff database refers to a digital chip design that has undergone specific stages of the design process. After logical or physical synthesis, the instances are placed and routed using a Place-and-Route (PnR) tool. Subsequently, static timing analysis is performed to optimize the timing and ensure that all critical paths meet the specified timing constraints. Various reports are generated to verify the timing requirements are satisfied. Additionally, the signoff database incorporates physical optimizations to enhance the chip's manufacturability and yield. These optimizations may include steps such as metal fills to achieve planarization of the chip's layout. The signoff database represents a final, validated design ready for tape-out, which is the stage of preparing the design for manufacturing.

The design under study is a 64-bit Arm core design that does not include any macros or memories. Nanosheets are used as cell technology with a transistor gate pitch of 45 nm, corresponding to node A14. The target frequency has been pushed towards the highest frequency and area utilization percentage. The design's maximum operating frequency (f_{max}) was obtained by identifying the Pareto points where there are more than 1000 critical paths when the target frequency or utilization area is not met. Additionally, there are no DRC rule violations, indicating successful routing.

The extraction flow, implemented in Tcl (Tool Command Language), is designed to extract critical paths from the signoff database based on predefined criteria, such as slack being less than or equal to zero. The script captures various characteristics of the nets in each critical path, including topology, fan-out (FO), source and load cell types, and cell (FEOL) and net (BEOL) delays.

This chapter focuses on the updates made to the Tcl script, which not only enable the computation of the Elmore delay but also facilitate the determination of the individual resistance (R) and capacitance (C) contributions to the overall delay. These contributions are obtained by rearranging and segmenting the Elmore delay formula.

To ensure the accuracy of the computed results, various sanity checks are discussed towards the end of the chapter. These checks involve comparing the computed values with the data that can be extracted from the database, providing further validation of the extraction and calculation processes.

3.1 Elmore delay computation on extracted net topology

Although the extraction flow in the previous implementation correctly handled the extraction of metal segments, it was found that the extraction of vias was not accurate. The previous method relied on comparing consecutive metal layers to determine the presence of a via. However, it was discovered that this derivative approach was not entirely reliable in certain special cases, as mentioned below.
To address this issue, a more robust method was devised, leveraging information from the *report-wire-path* command in Innovus. This command provides detailed information about wire paths, including the presence of vias. Using this command, the new script ensures more accurate and reliable extraction of vias. The previous script missed some vias when the metal layers of a cell were used for routing. This was because the *report-wire-path* command does not examine the cells and only reports the routing wires. Consequently, in cases where there were vias to enter or exit a cell, but no change in metal layers, the previous script failed to detect them.
Additionally, the previous script also missed some vias when the routing involved jumping from one metal layer to another non-consecutive metal layer using two different consecutive vias. These limitations and shortcomings were addressed and rectified in the updated script by utilizing the more comprehensive vias information provided by the *report-wire-path* command.

The accurate extraction of the number of vias plays a crucial role in computing the Elmore delay, particularly in the case of very short nets (less than or equal to 0.1 μm), where the presence of vias dominates both the resistance and the delay. This step is vital to improving the overall fidelity of the extraction flow and enables intermediate sanity checks by comparing the total net resistance and capacitance with the values extracted from the database for each net. When the exact number of vias is captured, the full extraction flow becomes more solid and reliable. It ensures that the computed Elmore delay and other characteristics align closely with the actual net properties, providing a higher level of accuracy and confidence in the results.

To implement the Elmore delay computation within the algorithm used for net topology extraction, several techniques are employed. These tricks ensure the accurate modeling of resistances and capacitances within the net.

1. Cumulative Resistance - The resistances of the metal layers along the main net path are accumulated up to each encountered branching point.

2. Branch Capacitance - Since branches contribute only as capacitive loads to the Elmore delay, the lumped capacitance of a branch is computed.

3. L-shape Metal Segment Modeling - Each metal segment is represented as a series resistance (R) followed by a grounded parallel capacitance (C).

4. Via Representation - Vias are modeled only as series resistances due to the complexity of accurately calculating their (coupling) capacitance.

5. Resistance Calculation - The resistance (R) of each metal segment is computed by multiplying the metal layer's resistances per unit length. Each via is assigned a constant resistance value.

6. Capacitance Estimation - Extracting the exact capacitance of each metal segment from the Innovus database is not possible using the *get-db* command. Instead, metal layer capacitances per unit length are used. Although resistances are exact, the capacitance values introduce some uncertainty in the delay computations when compared to those computed by the Cadence tool.

7. Variability Considerations - In real-case scenarios, metal segment capacitances can vary due to factors such as local variability and additional coupling capacitances. These variations are more pronounced for very short nets (less than or equal to 0.1 μm). However, when considering longer nets (several dozens of μm), these differences tend to average out.

These techniques collectively enable accurate estimation of delays.
Once the Elmore delay is computed, it can be segmented to determine the relative contribution to the delay from each metal and via layer, and pins.

3.2 Path delay Resistance and Capacitance contributions

The Elmore delay formula can be expressed as a linear sum of RC couples, representing the contributions from each metal and via layer. By grouping specific resistance (R) or capacitance (C) factors, it is possible to estimate the relative contribution of each metal layer, via, and pins to the overall path delay. These delay contributions are referred to as RC contributions, pointing out that they stem from the summation of RC pairs that define the Elmore delay computation. The path delay is computed by summing the BEOL and FEOL delays of all the gates and nets within each path 3.1.

$$\text{Path delay} = \sum_{i}^{\#\text{path stages}} (\text{BEOL RC delay}_i + \text{FEOL RC delay}_i) \tag{3.1}$$

The formulas 3.2, 3.3, and 3.4 provide expressions for the relative contributions of metal resistance (R) and capacitance (C) to the path delay, as well as the contribution of via resistance ($R_$via). Thus allowing for a detailed analysis of the impact of each metal

layer and vias on the overall delay.

$$RC_{n-\text{metal layer R}} = \frac{1}{\text{Path delay}} \cdot \left\{ \sum_{i}^{\#\text{path nets}} \sum_{j}^{\#\text{main metals of the n-layers}} \left[R_{i,j-\text{wires}} \cdot \left(\sum_{k}^{\#\text{metals after the j-metal}} C_{i,j,k-\text{metal}} + \sum_{k}^{\#\text{pins after the j-metal}} C_{i,j,k-\text{pin}} \right) \right] \right\} \tag{3.2}$$

$$RC_{n-\text{metal layer C}} = \frac{1}{\text{Path delay}} \cdot \left[\sum_{i}^{\#\text{path nets}} \sum_{j}^{\#\text{metals of the n-layers}} \left(C_{i,j-\text{wires}} \cdot \sum_{k}^{\#\text{main metals/vias up to the j-metal}} R_{i,j,k-\text{metal/via}} \right) \right] \tag{3.3}$$

$$RC_{n-\text{via layer R}} = \frac{1}{\text{Path delay}} \cdot \left\{ \sum_{i}^{\#\text{path nets}} \sum_{j}^{\#\text{main vias of the n-layers}} \left[R_{i,j-\text{vias}} \cdot \left(\sum_{k}^{\#\text{metals after the j-metal}} C_{i,j,k-\text{metal}} + \sum_{k}^{\#\text{pins after the j-metal}} C_{i,j,k-\text{pin}} \right) \right] \right\} \tag{3.4}$$

Figure 3.1 presents the R and C contributions obtained by applying the Elmore delay computation to all the critical paths with negative slack of a given design with the BEOL stack assumed in the article [5] and depicted in tables 3.2 and 3.3.

This distinction is crucial when determining the optimal achievable optimization for a wiring booster technology.
As an example, hybrid height is a technological solution that involves a trade-off between the resistance and capacitance of metal layers, making it important to consider the R and C delay contributions to make informed decisions [5].

In addition to the contributions of R_{wire}, C_{wire}, and R_{via}, various other C statistics can be derived by manipulating the Elmore delay formula. Table 3.1 presents several contributions to the path delay obtained for pins and branches. It is worth noticing that C_{pins} contributes around 38% to the BEOL delay. This very high contribution is explained by the fact that the average FO is around 4 and the maximum FO has been forced, through a design constraint, to be limited to 40. Moreover, since the database is dominated by very short nets, below 1 μm, knowing that the maximum capacitance per μm is about 0.4 fF/μm and the pin capacitance of a cell with drive strength 1 is around 0.2 fF, this result is reasonable. The same considerations explain why the C contribution of the main pins is about 1/5 of the total pin contribution. Comparing this value with the average FO of 4 can lead to the fact that, on average, cells with higher pin capacitance,

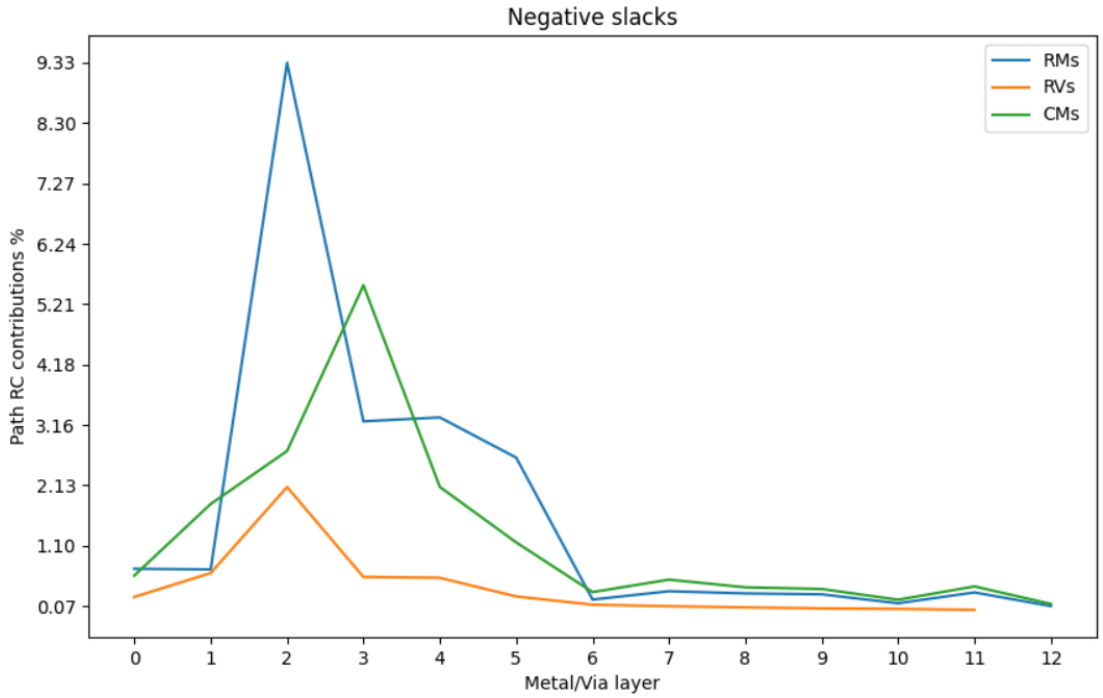

Figure 3.1. Average R and C metal contributions and R vias contributions to the path delay. Average computed on all the critical paths with negative slack of the design under study.

RC parameters	RC contribution to the path delay (%)
$C_{\text{main pins}}$	1.92
$C_{\text{branching pins}}$	7.69
C_{pins}	9.62
$C_{\text{branching wires}}$	4.60
C_{branches}	12.30
BEOL delay	25.61
FEOL delay	74.39

Table 3.1. More RC contributions to path delay (%)

that is, higher drive strength, statistically belong to the branches. This argumentation is strengthened considering that the branching pins always have an upstream resistance of the main net part which is smaller or at most equal to the one related to the main pin. Thus, their contribution is even subject to this downscale effect.

The values of the branching wire and branching pins show that 62% of the branching capacitance comes from the pins. Thus when a sink needs to be reached, branching wires represent an additional delay cost, about half of the branching pin cap, which is summed up.

One fourth of the path delay is on average due to the interconnects because the CPU design is dominated by short paths but also by longer ones when the FO gets big.
The analyzed design exhibits a Back-End-of-Line (BEOL) delay, caused by interconnects,

Metal Layer	Material	Pitch (nm)	Width (nm)	Spacing (nm)	Thickness (nm)	R per µm ($\Omega/\mu m$)	C per µm (fF/μm)
0	Ru	18	10	8	27	518	0.319
1	Ru	28	16	12	49	155	0.391
2	Ru	18	10	8	27	518	0.344
3	Ru	28	16	12	49	155	0.388
4	Cu	28	16	12	28	245	0.270
5	Cu	28	16	12	28	245	0.237
6	Cu	80	40	40	80	16	0.203
7	Cu	80	40	40	80	16	0.202
8	Cu	80	40	40	80	16	0.202
9	Cu	80	40	40	80	16	0.202
10	Cu	80	40	40	80	16	0.202
11	Cu	80	40	40	80	16	0.202
12	Cu	80	40	40	80	16	0.193

Table 3.2. BEOL metal stack.

Via Layer	Material	Thickness (nm)	Area (nm x nm)	R (Ω)	C (aF)
0	Ru	20	10 x 16	23.5	3.2
1	Ru	20	16 x 10	14.2	4.6
2	Ru	28	10 x 16	52.7	4.4
3	Ru	28	16 x 16	26.7	4.6
4	Cu	28	16 x 40	26.7	3.5
5	Cu	80	40 x 40	16.1	5.2
6	Cu	80	40 x 40	6	6
7	Cu	80	40 x 40	6	6
8	Cu	80	40 x 40	6	6
9	Cu	80	40 x 40	6	6
10	Cu	80	40 x 40	6	6
11	Cu	80	40 x 40	6	6

Table 3.3. BEOL via stack.

which on average contributes 25% to the path delay of the worst critical paths with $slack \leq 0$, as extracted from a signoff database. In the table 3.2 and 3.3 are listed the resistances and capacitances per unit length used in the computation of the values R_{wire} and C_{wire} and R_{via}.

The primary factor contributing to the delay from the BEOL is the R_{M2} and the C_{M3}. This outcome is reasonable given that M2 is the most frequently utilized metal layer, and its resistance is considerably higher compared to M1 and M3. M2 sees greater usage compared to M1 due to its increased wiring resources stemming from a more compact metal pitch. Additionally, M1, similar to M0, is allocated for cell layout, further limiting the available resources for routing.
Moreover, the Capacitance of M3 is contributing more that the one of M2 because of a higher capacitance per unit length. Together with that, the M3 is also more used as a branching wire than as the main wire, so its contribution is statistically most impacting as the C value.

Upon examination of the vias, it appears that their contribution to the path delay is not statistically significant. Overall, they contribute 5% to the path delay. Vias play a crucial role in connecting cells that use different metal layers, ranging from M0 to M2. The PnR optimization tool treats instances as black boxes and connects them by contacting their pins. Contacts can be made using vias to reach higher metal layers of the cell or without vias if the routing metal layer is at the same level as the instance pin. V0, V1, and V2 vias are commonly used to access cells and are essential components due to the structure of the BEOL stack. Starting from V3, vias are utilized for two main reasons: to access faster metal layers for longer distances while maintaining reasonable timing by using less resistive metal layers (such as Mz) with larger cross-sections and relaxed metal pitch, or to avoid pre-routed tracks where signals were already routed in previous iterations of the PnR algorithm.

The introduction of this new feature clearly highlights the impact of wiring interconnections on path delay, which was previously reliant on general statistics regarding wiring lengths. The previous approach was unable to differentiate between the contributions of R and C to the path delay. In section 5.1 we explain why the previous methodology is lacking in accuracy and a new mapping method is proposed in 5.2.

3.3 Checking R, C, and RC

The Elmore delay computation explained in 2.5 becomes increasingly challenging when performed manually, even for short nets. It requires addressing the exact net topology, accurately determining the values of R_{wire} and C_{wire} and the R_{via} treating branches as capacitance-only contributions, and properly calculating the resistance up to the branching point.

This section presents the systematic checks conducted to ensure the reliability of the computed Elmore delay. The goal of this work is not to obtain an identical value to the one computed by the Cadence tool, as the values also depend on the RC structure considered as a model of a single segment, and our computation neglects Via capacitances. While achieving the same value would be desirable, the main objective is to predict the same frequency shifts that a design would experience when changes are made to the BEOL stack, such as variations in resistance and capacitance.

In this section, additional data extracted from the database is explained to enable

essential sanity checks. A statistical comparison is made between our Elmore computation and the one performed by the Cadence tool, and the alignment of predictions with reality is discussed.

3.3.1 Net R and C reliability values

The Elmore delay, being composed of a summation of RC pairs, heavily relies on the accurate computation of the correct R and C values. Unfortunately, there is no built-in Innovus command specifically designed to extract the exact R and C values of a net segment.
The closest available information is the total resistance and total capacitance of a net. When comparing resistance values, only nets with $FO = 1$ are considered since the script ignores branch resistances as they do not contribute to the Elmore delay. In the case of metal layers, the resistance depends solely on the material properties and dimensions (height, width, and length). For vias, the resistance value remains constant. Therefore, if the extraction is implemented properly, there should be no mismatch between the computed and extracted resistance values.

On the other hand, capacitances are more complex as they depend on the surrounding environment of one conductor with respect to another. The implemented code computes capacitance using a capacitance per unit length, following a similar approach as with resistances.

To compare the extracted values with the calculated values, a relative mismatch is computed using 3.5, which quantifies the difference between the two data sets.

$$\text{Relative mismatch} = \frac{\text{expected value} - \text{computed value}}{\text{expected value}} \tag{3.5}$$

Reliability is defined as the average of the relative mismatch 3.6.

$$\text{Reliability} = \overline{\text{Relative mismatch}} \tag{3.6}$$

The reliability values computed for the R and C are shown in the table 3.4.
Similarly to R and C, the reliability value of the RC delay can be computed by using Equation 3.6 to compare the Elmore delay with the delay value calculated by the PnR tool of Cadence during Static Timing Analysis. By averaging these values, the reliability value of the RC is obtained. The histogram of relative mismatch is shown in figure 3.2.

	Reliability mean (%)
RC	20.5
C	12.8
R	0.1

Table 3.4. R, C, and RC mean reliability values (%).

It shows an asymmetrical Gaussian-like distribution

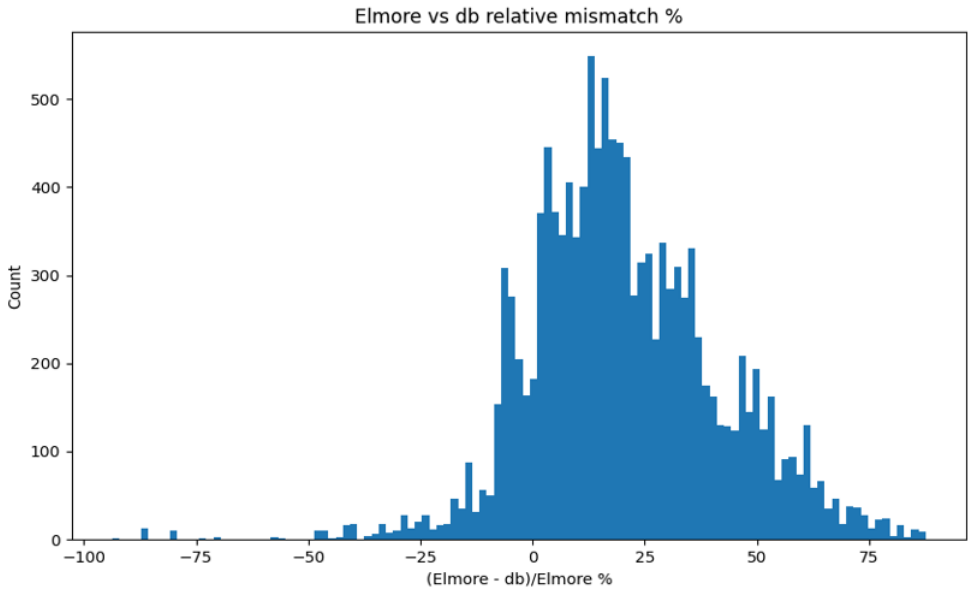

Figure 3.2. Histogram of the relative mismatch of Elmore computation compared with the values calculated by the static timing analysis of the PnR tool.

3.3.2 Elmore predictions vs static timing analysis

After noticing the significant discrepancy between the Elmore delay computation and the results obtained from the tool, it is necessary to verify the predictions of the R and C contributions to the path delay. To conduct this verification, one metal resistance or capacitance should be modified at a time, followed by running the static timing analysis again.

The .qrc file contains all the R and C data, along with parasitics, which were computed for the given BEOL stack defined in the .ict file. In the .ict file, the BEOL stack is defined, specifying the dielectrics with their thickness and dielectric constant. Metal layers are defined as conductors with specific spacing, width, thickness, and ρ value. Vias are defined with a constant resistance based on their area. To make changes to the .qrc file, the .ict file needs to be modified accordingly, and then the .qrc file needs to be updated to reflect the modification.

To perform the Rs scaling check, the ρ values in the .ict file need to be scaled. Reducing the ρ values by 50% is a reasonable reduction factor to ensure a reasonable ρ while achieving a noticeable reduction.
An automated flow has been implemented to systematically change the ρ of each metal layer in the .ict file, update the .qrc file through a TCAD simulation, and run the static timing analysis using the updated database. The extraction flow is performed on the same critical paths ($slack \leq 0$) as before the modification in the .ict file.

The predictions obtained from the Elmore delay computation are expressed relative to the path delay. Therefore, the shift in the path delay needs to be extracted and compared with the predictions.

It is assumed that the FEOL delay would not have changed due to the R modification in the BEOL stack. However, it should be noted that the signal's slew is changing, leading to slight variations in the FEOL delays. EDA tools utilize lookup tables to define the cell (FEOL) delay, considering the input slew and the total output capacitance load. These lookup tables are generated using a library characterization system tool.

To compare the predictions while eliminating the impact of FEOL delay changes, the metal contributions to the path delay are expressed relative to the interconnect (BEOL) delay. To maintain consistency in the computations, the delay contributions are scaled by the ratio of the average BEOL delay to the average path delay, which is calculated as $< \frac{\text{BEOL delay}}{\text{Path delay}} >= 0.2598$. This scaling ensures that the relative contributions of Rs to the path delay remain consistent throughout.

Furthermore, it is crucial to ensure consistency between the computations. The predictions are obtained by averaging the relative contribution of Rs to the path delay, as indicated by equation 3.2. Therefore, for each path, the recalculated BEOL delay is compared to the original BEOL delay (using the database with an unchanged .ict file).

The comparison between the recalculated and original BEOL delays is expressed as a relative shift, as described in equation 3.7. This relative shift allows for an assessment of the changes in BEOL delay resulting from the modification in the .ict file and provides insight into the impact on the overall timing of the path.

$$\text{Relative BEOL delay shift} = \frac{\text{recomputed BEOL delay} - \text{original BEOL delay}}{\text{original BEOL delay}} \cdot 100 \tag{3.7}$$

Then the average of these shifts is computed

$$\text{Average relative BEOL delay shift} = \overline{\text{Relative BEOL delay shift}} \tag{3.8}$$

Table 3.5 presents the Average BEOL shift and the Predictions used to calculate the BEOL reliability for each new database created from a different .ict file, where the metal layer Rs have been scaled. The Average BEOL shift provides information on the average change in BEOL delay relative to the original database (with an unchanged .ict file). The Predictions in the table are utilized to determine the BEOL reliability, taking into account the modified R values and their impact on the path delays.

$$Reliability(\%) = \frac{Prediction - Shift}{Prediction} \cdot 100 \tag{3.9}$$

$$Predictions = -\frac{\text{RC contributions}}{< \frac{\text{BEOL delay}}{\text{Path delay}} >} \cdot 0.5 \tag{3.10}$$

where $< \frac{\text{BEOL delay}}{\text{Path delay}} >= 0.2598$ for the design under study.

The reliability values of the BEOL delay shifts and the Average BEOL delay shift are plotted in figure 3.3. Figure 3.4 illustrates the comparison between the predictions (top) and the actual shifts (below) experienced by each database. Qualitatively, the two plots show good agreement between the predictions and the actual shifts. However, quantitatively, there are some mismatches, particularly for the lower metal layers, with discrepancies of up to 20%. This mismatch can be attributed to the assumption of

Modified database	Average BEOL shift (%)	Predictions (%)	RC contributions (%)	BEOL reliability (%)
M0	-1.58	-1.36	0.71	-16.11
M1	-1.47	-1.34	0.70	-9.19
M2	-16.92	-18.05	9.38	6.24
M3	-5.49	-6.23	3.23	11.93
M4	-4.95	-6.36	3.30	22.10
M5	-3.96	-5.03	2.61	21.21
M6	-0.45	-0.35	0.18	-27.25
M7	-0.71	-0.62	0.32	-14.27
M8	-0.55	-0.55	0.29	0.21
M9	-0.52	-0.52	0.27	-0.73
M10	-0.21	-0.23	0.12	8.32
M11	-0.46	-0.58	0.30	21.46
M12	-0.12	-0.13	0.07	9.64

Table 3.5. Modified Database: 50% R Scaling of n-Layer.

using constant capacitance values in the Elmore calculations. In reality, due to coupling effects, capacitances are not constant and depend on the local neighborhood, making them unpredictable with the current approach. Finding a way to extract these capacitance values from the database using a built-in command is necessary.

The R predictions show the correct trend, although the absolute values are not predicted with high accuracy.
Additionally, it is important to consider that when applying a new technology to a design, another PnR flow needs to be executed, and the actual frequency improvement achieved could potentially exceed the predictions. Thus, further analysis is required to incorporate these factors when building a predictive model that aims to anticipate the impact of a new technology when implementing a change in the BEOL stack.

3.4 Application: BEOL stack geometries and booster technologies

The results presented in Figure 3.4 hold potential for various applications, such as booster technologies or optimization of BEOL stack geometry. The BEOL geometry entails a significant trade-off between R and C, as well as the number of tracks available within a given chip area.

For instance, modifying the spacing between metal layers can increase track density but also increase the capacitance. Similarly, increasing the line thickness reduces resistance but increases capacitance. The introduction of air gap technology offers a low-k dielectric option, although it may not be feasible to implement it on all metal layers due to mechanical reliability and cost considerations. Therefore, it is crucial to identify

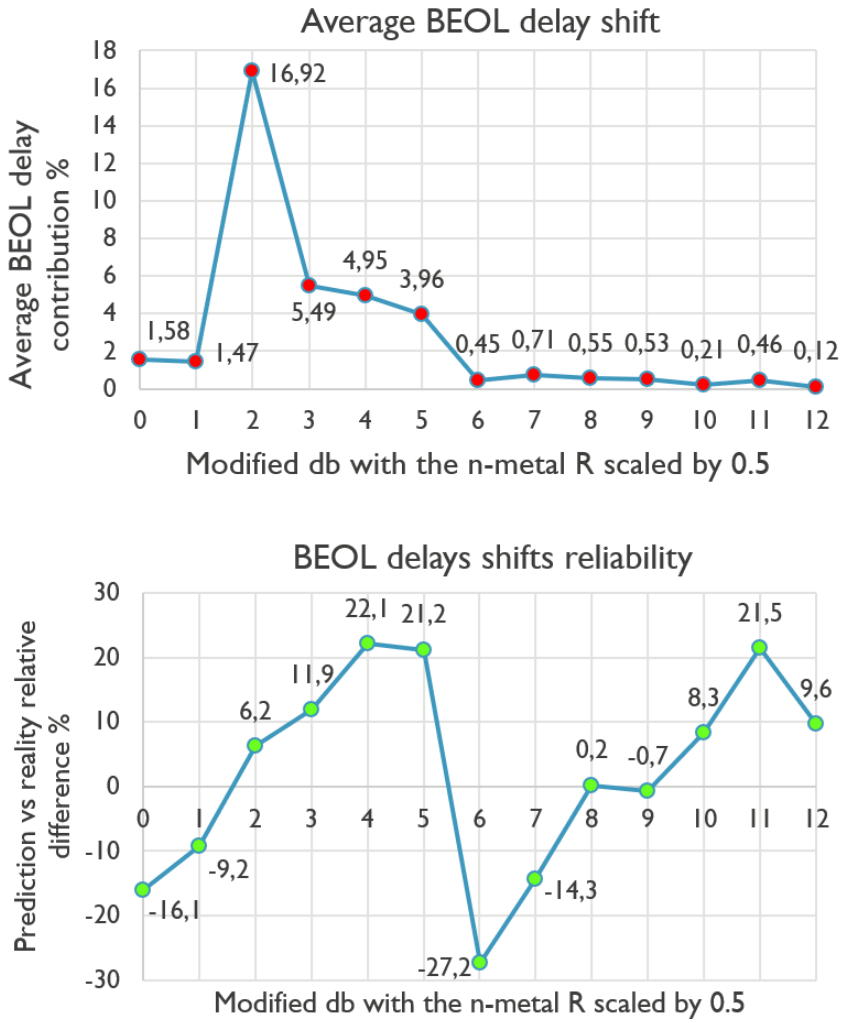

Figure 3.3. Average BEOL delay shift and reliability.

the metal layer that has the most significant impact on BEOL delay, as this layer can maximize the benefits of new technologies.

Hybrid height solutions involve trading off the resistance of one metal layer with the capacitance of another layer. Identifying the optimal layer for implementing such boosters becomes essential.

Graphene capping is another technology that can reduce resistance by adding a few layers on top or on the sides of the metal. However, challenges related to manufacturing, such as delamination and reliability, need to be addressed [5].

Analyzing the R and C contributions to the path delay can serve as a valuable tool for quickly assessing improvements brought by new booster technologies and guiding research activities toward investigating promising future technologies. By increasing research yield and reducing costs, this approach can contribute to advancements in the field.

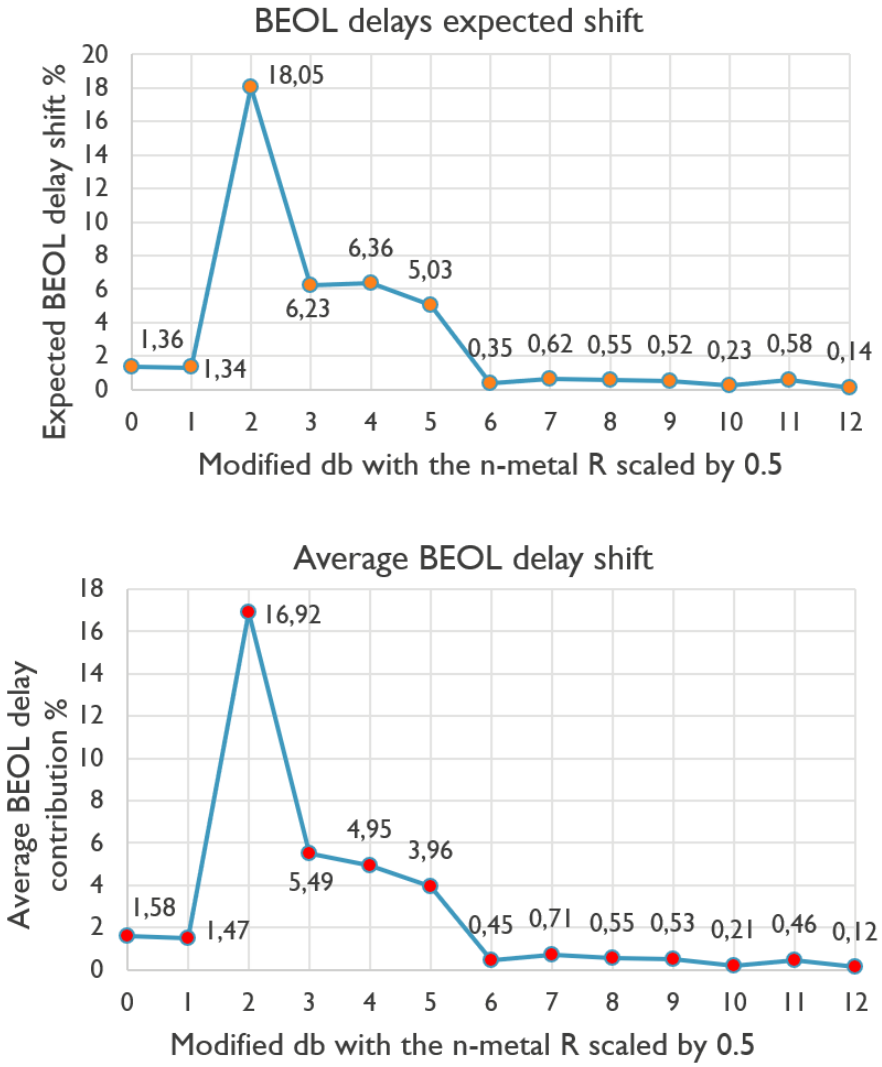

Figure 3.4. Comparison: predictions vs. actual shifts in each database containing a different .qrc file.

Chapter 4

Statistical data analysis of critical paths

Following the extraction process, pertinent information and topology of all critical paths are stored in a .txt file. This file is structured so that each row contains information related to paths, nets, metals, or vias.

To facilitate further statistical analysis, a Python script is employed to convert this ASCII database into a high-performance .csv (comma-separated values) format. The tabular arrangement of this database allows for the application of the 'pandas' library [14], simplifying tasks such as reading, analyzing, filtering, grouping, and manipulating data. This library also equips you with functions for fundamental statistical analyses like mean, median, and standard deviation. Furthermore, it allows for the generation of distinct tables that can be saved in separate files and Excel sheets. This practice enables the isolated analysis of various design characteristics.

The automation of data analysis streamlines comprehensive statistical evaluation for each .signoff database. Consequently, the complete statistical data analysis can be effortlessly conducted for every .signoff database. This approach ensures that only pertinent figures and plots are produced using scripts, thus honing in on the essential data among the vast amount extracted.

By leveraging the predefined and fixed structure of each line, the script extracts and collects the necessary information. It utilizes the pandas library to create two-dimensional, size-mutable, potentially heterogeneous tabular data. Counters and flags are employed to keep track of the script's position along the net during scanning, enabling nested printing. Two databases are created and updated: one for the path of the main sink and another for the main branches of the net. Headers are added to enhance readability and ease the recognition of values by the user.
During this process, some post-processed information is extracted to facilitate future analysis. For example, it computes the normalized branching position relative to the main sink path length and saves all the metals belonging to the path of the main sink up to the branching point.

In the main csv output, each row represents a main net part, while in the branch

csv output, each row represents a branch of the main net part. The database contains information such as metal lengths in micrometers (µm), capacitances in picofarads (pF), resistances in ohms (Ω), delays in picoseconds (ps), and slacks in nanoseconds (ns). Each path, net, or branch is labeled with its respective ID.

Once the database is prepared, it becomes possible to perform extensive data analysis on it. The subsequent figures are presented and discussed, offering valuable insights for predicting the impact of BEOL boosters and gaining a better understanding of final routing optimization and the statistical behavior of optimizations performed by the PnR (Place and Route) tool. To analyze the csv database effectively, a Python script is employed to transform it into an array that can be easily scanned. The csv database contains various information, so it is important to select the appropriate parameters for meaningful dependencies.

4.1 Gate sizing

Optimizing the sizing of logic gates within a circuit can significantly improve performance in terms of speed and power consumption. Gate sizing involves adjusting the dimensions of the transistors within the gate, either by making them narrower or wider based on a scale factor.
Each gate has an optimal scale factor that considers the trade-off between increased transistor size, which enhances their ability to handle loads and speeds up signal switching, and the disadvantages of larger gate size. These disadvantages include reduced output resistance and increased input capacitance, which can impose higher capacitive loads on the driver.
A widely recognized approach for determining the optimal gate dimensions within a series of stages is referred to as "logical effort" [13].
Capacitive loads in gate sizing consist of wiring loads and pin cell loads. By carefully optimizing the gate sizing, designers can strike a balance between improved performance and the potential drawbacks associated with increased transistor dimensions.

The presence of interconnects also plays a role in the problem of logical effort which aims at the best sizing for minimum delay [13]. Optimally, to achieve the minimum path delay in a real case scenario where interconnects are also considered, the best gate sizing is achieved when the delay component $(R^{\text{eff}}_{\text{cell}_{i-1}} + R_{\text{wires}_{i-1}}) \cdot C^{\text{input}}_{\text{cell}_i}$ due to the input capacitance of the gate of the i^{th} cell is equal to the delay component $R^{\text{eff}}_{\text{cell}_i} \cdot (C_{\text{wires}_i} + C^{\text{input}}_{\text{cell}_{i+1}})$ due to the effective resistance of the i^{th} cell [9]. When incorporating interconnects in the model, the minimum path delay condition deviates from the equal distribution of delay or effort across each stage of the path, as observed in the logical effort model without interconnects [13]. This disparity arises due to fixed capacitances in interconnects, which do not correlate with the characteristics of the gates.

The subsequent passages delve into the examination of gate sizing alongside key influencing factors, including capacitive load and wire lengths. The resulting delays in both

BEOL and FEOL aspects are scrutinized to comprehend the outcomes of the optimization's effectiveness. These findings will subsequently undergo deeper exploration in the following sections to enhance our comprehension.

Capacitive load

Figure 4.1 shows the cumulative density function (CDF) of the capacitive load for all the drive strengths (DS).

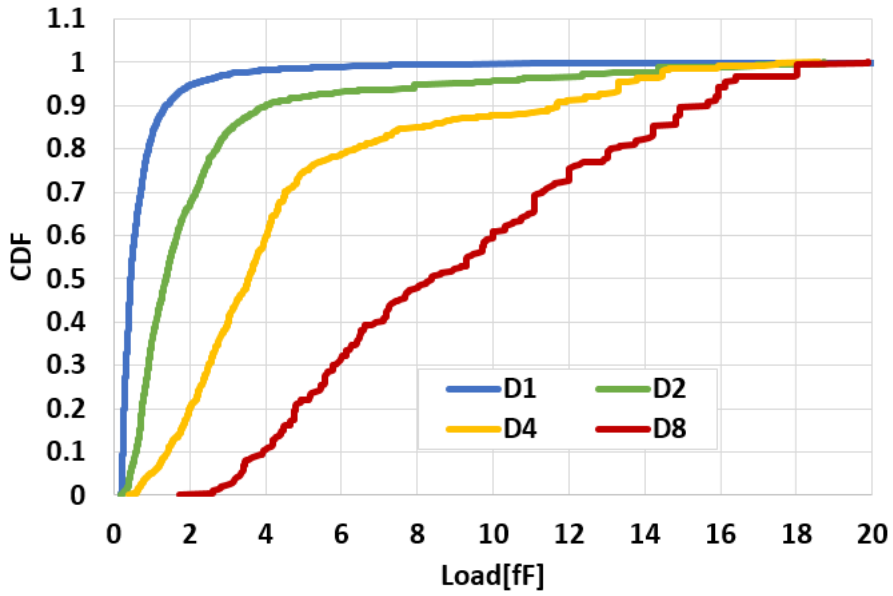

Figure 4.1. Capacitive load cumulative density function (CDF) for different drive strengths (DS).

In general, larger capacitive loads typically require larger driver gates. This is especially true for scenarios involving large nets with a high fan-out (FO) or long lengths leading to high capacitance. In such cases, higher drive strengths (DS) are often chosen to ensure effective signal transmission and compensate for the increased load.

Net lengths and metal distributions

Figure 4.2 shows the CDF of the gate sizing with respect to the net length (NL).

Higher drive strength settings often result in increased variability in net length. For a fixed drive strength, the distribution of metal layers closely follows the pattern of net length. Examining the D1 statistics, it becomes apparent that D1 mostly utilizes Mx layers. On the other hand, the D4 statistics reveal a greater diversity in metal distribution, indicating that D4 experiences variations in the utilization of different metal layers.

Shorter nets tend to make greater use of Mx layers, while longer nets tend to utilize My and Mz layers more extensively. Additionally, cells with higher drive strength settings typically employ upper metal layers for their output interconnections. These observations

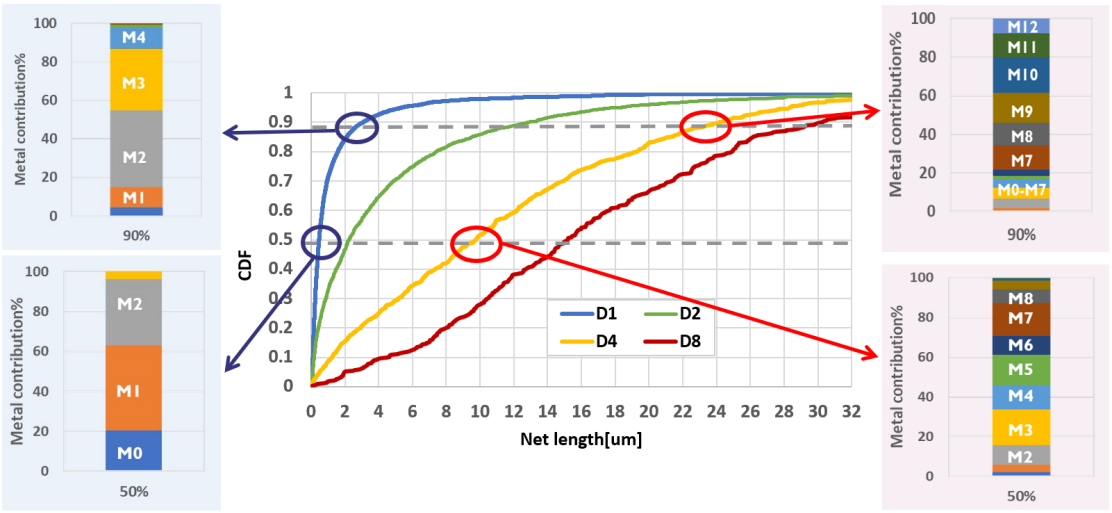

Figure 4.2. Gate sizing dependency on net length and metal distributions.

highlight the relationship between drive strength, net length, and metal distribution in circuit designs.

BEOL vs FEOL delays

The path delay in a circuit consists of both cell delays and interconnect delays. Analyzing how the delay is distributed between the front-end of line (FEOL) and back-end of line (BEOL) components can provide valuable insights for optimization.

In Figure 4.3, the FEOL and BEOL delays are plotted in picoseconds (ps) as a function of the cell drive strength. The number of nets is also counted for each drive strength. The findings reveal that for low drive strengths, the FEOL components dominate the delay. This is because the nets are relatively short, allowing signals to quickly reach their destination sinks.

Conversely, high drive strengths result in BEOL dominance. Even though the cells are faster due to wider channels and higher currents, the delay is significantly influenced by the larger net size.

For the D4 drive strength, a more balanced distribution between FEOL and BEOL delays is observed.

Understanding the relationship between FEOL and BEOL delays and selecting appropriate drive strengths based on net lengths can greatly optimize circuit performance. This analysis provides valuable insights for such optimization efforts.

4.2 Unraveling the criticality of critical paths

Finding a universal fingerprint for negative slack critical paths is challenging due to the varying characteristics of the nets that constitute them. There are several reasons why a critical path is critical (negative slack). The main ones are the followings:

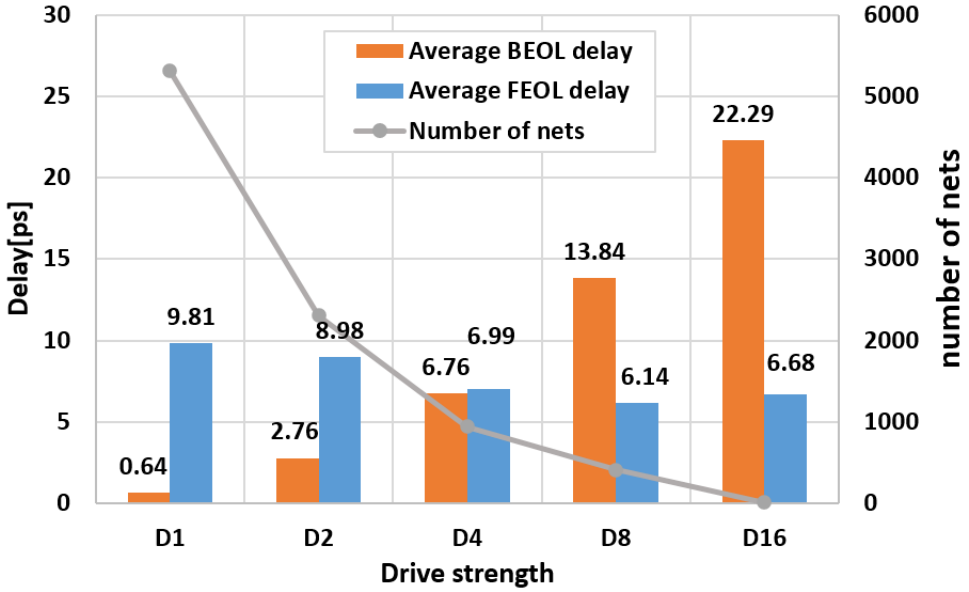

Figure 4.3. Analysis of FEOL and BEOL delays with cell drive strength and net count.

- Clock skew;
- High logic depth;
- High BEOL and/or FEOL delay nets

Clock skew

The clock skew is the difference in the arrival time of a clock signal at two different registers. The slack is computed by 4.1. The *required time* is defined as the minimum time needed for the signal to travel from the timing start point to the endpoint and is computed as 4.2. The *Other End Arrival time* is the moment when the capture clock edge reaches the clock pin of the capture flop. The *Phase shift* refers to the adjustment made from the ideal clock edge to properly align with the moment when the data should be captured. The *Clock Path Pessimism Removal* (CPPR) is the process of identifying and removing the pessimism introduced in the slack reports for clock paths when the clock paths have a segment in common because the common clock path is considered to be both late and early at the same time. The *setup* constraint, from the timing library for the flop, specifies the minimum amount of time that data must be stable and valid before the clock edge arrives at the clock pin of the flop so that the data is reliably captured by the register. The *uncertainity* is the clock uncertainty of the launch clock. The *Arrival time* refers to the time it takes for a signal to propagate from the clock port to the D (data) pin of the capture flip-flop 4.3. The *Beginpoint Arrival time* or *Launch clock* is the clock edge at the clock pin of the flop, relative to the leading edge of the ideal clock waveform. It includes the clock network latency. The *Data path* is the time it takes for

the data signal to travel from the Q (output) pin of the launch flop to the D (Data) pin of the capture flop.

$$\text{Slack} = \text{Required time} - \text{Arrival time} \tag{4.1}$$

where

$$\text{Required time} = \text{Other End Arrival time} - \text{Phase shift} - \text{Cppr Adjus} - \text{Setup} - \text{Uncertainity} \tag{4.2}$$

$$\text{Arrival time} = \text{Beginpoint Arrival time} + \text{Data path} \tag{4.3}$$

The clock skew can cause the difference between the *Other End Arrival time* and the Beginpoint Arrival time to be small so that can cause a fast path, with low *Data path* time, to have a negative slack.

High logic depth

The logic depth (LD) represents the number of stages (or nets) characterizing the path between two registers. Alternatively, it can be seen as the number of cells + 1, between two registers. The logic depth is directly influenced by the RTL (Register Transfer Level) and is a direct outcome of the logic synthesis process, apart from the buffer insertion done for timing optimization purposes. The synthesis could also perform buffer insertion but in the current PnR flow, those buffers are deleted and added in subsequent steps such as post placement, post clock tree synthesis, and post route. Figure 4.6 shows that the highest LD values are spanning from the average of 21 up to the value of 33. Knowing that the average cell delays are around 8 ps as shown in the figure 4.3, the minimum path delay should range from 168 ps ($21 \cdot 8$) up to 264 ps ($33 \cdot 8$) only for what concerns the cell delays. Thus, the LD is intrinsically representing the maximum frequency at which a path can run if we suppose all D1 cells and ideally a negligible BEOL delay. Realistically speaking the BEOL delay usually counts from 10% to 25% of the path delay for a CPU design with no macros or memories.

High BEOL and/or FEOL delay nets

For every path, the logic depth is defined from synthesis. In the specific scenario where FO always equals 1, and the pin capacitance of a register (D flip-flop) closely matches that of a D1 cell, the delays between stages along that path should exhibit a uniform distribution because every cell should see the same load if supposing the same wiring load. Thus we expect every stage to have, on average, a delay equal to $\frac{T_{clock}}{LD}$. In the design under study, looking into the critical path characteristics, they show fast stages and very slow ones. In particular, there are stages with very high BEOL delay, FEOL delay, or both. The FEOL delay can be up to 10 times bigger than the average case and the BEOL delay can vary from a few decimals up to several dozens of picoseconds. The reason behind the non-uniform distribution of delays between stages is attributed to the variability in FO. As FO increases, the net's load also increases proportionally, resulting from both the cumulative pin capacitances of the fan-out sinks involved and the added capacitances from the wiring branches.

Assuming that the cell rise and cell fall (FEOL) delays are identical, they depend on the input slew (transition time) of a cell and the output total capacitance which is loading the cell. The slew or transition time represents the steepness of the switching signal, thus it is the time that it takes for the signal to go from $30\%V_{DD}$ to $70\%V_{DD}$ during a rising transition or from $70\%V_{DD}$ to $30\%V_{DD}$ during a falling transition. The cell delay is big when one or both the input slew or the load capacitance are high. Also, the output slew is determined in the same way. This implies that the best way to minimize the FEOL delay is to keep low the capacitance in front of the cells. To do so, the FO should be limited to a maximum number and since the wire capacitance per unit length doesn't vary much between layers, also the net lengths should be kept as low as possible, which can also be seen as placing all the cells, connected to the same net, close enough.

For example, a D1, to be fast, should not drive a load of more than 1.2 fF, which means, a maximum fan-out of 2 or 3 if considering that 0.5 fF of pin and wiring capacitances on average are added for every additional sink. Since the current design is synthesized with a fan-out that is almost given as a free parameter, limited to 40, design performances can be improved by limiting the FO to a more realistic value. This constraint should be determined based on factors like the BEOL stack, floorplan area, and the maximum available drive strength. However, by constraining the FO, the LD starts to increase, thus an optimal average FO exists depending on the design and the technology.

Similarly to the previous discussion about high cell delays, the BEOL (net) delay is big for long nets with high FO. The BEOL delay dependency with respect to the FO is shown in figure 4.4. Normally, delays of up to 20 ps are maintained when the FO is

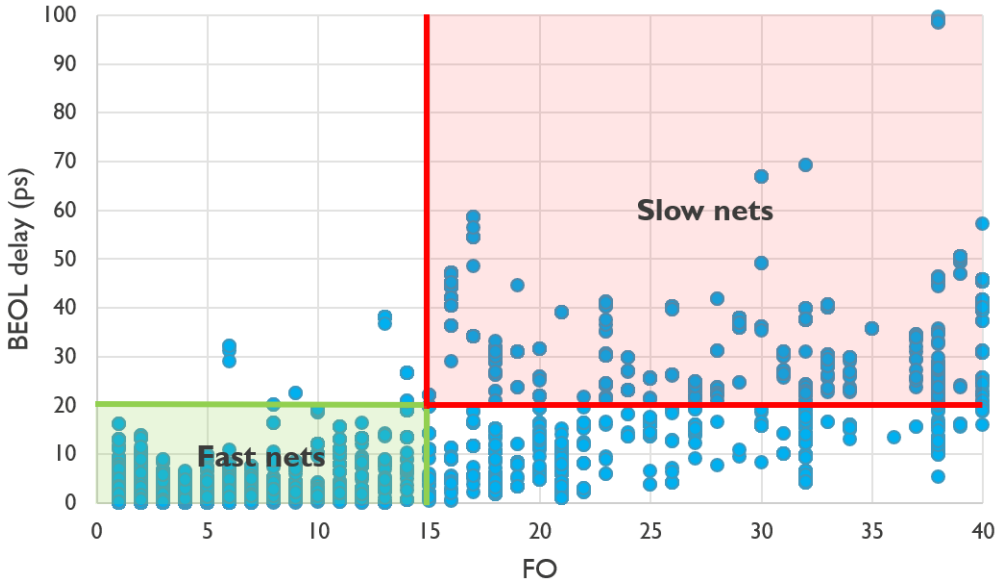

Figure 4.4. Interconnects (BEOL) delay as a function of the FO. When the FO is smaller than 15, nets are likely to be fast (BEOL delay $<$ 20 ps). As the FO increases, the nets start to slow down.

less than 15. However, once the FO exceeds 15, the speed of connections decreases due to longer interconnects. The optimization tool encounters challenges in positioning cells from high FO categories in close proximity to maintain adequate swift connections.
Adding a fan-out sink, not only increases the net capacitance through the pin capacitance, thus adding an additional delay contribution due to the branching pin, but also adds branching wires to the net. Branching wires only contribute as capacitance to the Elmore delay but depending on the branching position, their contribution increases as the branching position gets closer to the end of the net. Moreover, as already mentioned, by adding branching wires and sinks, the overall capacitive load increases, thus a stronger cell is required to drive the net in order to keep a low cell delay.

In summary, identifying the patterns that are making a path to be critical, steps can be taken to limit such cases and keep the BEOL delay below a certain threshold. This can be achieved by refining the design space during the definition of the PnR (Place and Route) flow.

An optimum between logic depth and fan-out exists for a given design and it should be found to achieve the best optimization result. This trade-off can be also seen as the challenge to increase the resource sharing by trading it with a more difficult placing optimization, which should make high FO stages have short wiring connections to ensure both low FEOL delay due to wire C and BEOL delay due to wiring RC.

4.3 Logic depth vs slack

To construct a ring oscillator that accurately mimics the analyzed design, it is necessary to determine the number of stages required. This information is crucial for assessing power consumption and the steady-state frequency of the ring oscillator.

The contribution of FEOL and BEOL delays in the ring oscillator's performance depends on the logic depth (LD). Paths with a higher number of cells (higher LD) will have a greater contribution to the FEOL delay, as the total delay in the path must be limited by the target period.

Figure 4.6 illustrates the relationship between logic depth and slack (in nanoseconds). It can be observed that an average value of LD is commonly used for all paths, regardless of the slack. The distribution of logic depths tends to spread more as the slack gets closer to 0. This is because the group of paths with slack closer to 0 is richer, as shown in figure 4.5, and more variability is allowed.

A noteworthy observation can be made by combining the findings from Figures 4.3 and 4.6. By multiplying the maximum logic depth by the average FEOL delay, it becomes possible to compute the minimum achievable frequency. This information can assist designers in comparing the maximum frequency achievable for a specific technology, considering a given RTL design.
The logic depth is influenced by the RTL synthesis and the buffer insertion performed subsequently. Assuming that the average FEOL delay remains consistent regardless of the logic depth, the minimum delay will be constrained by the longest path with the

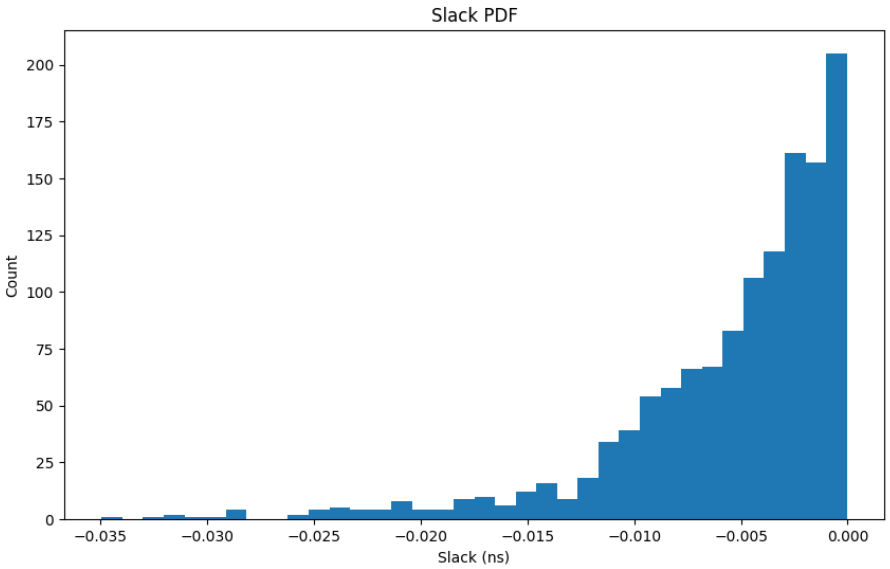

Figure 4.5. Slack histogram.

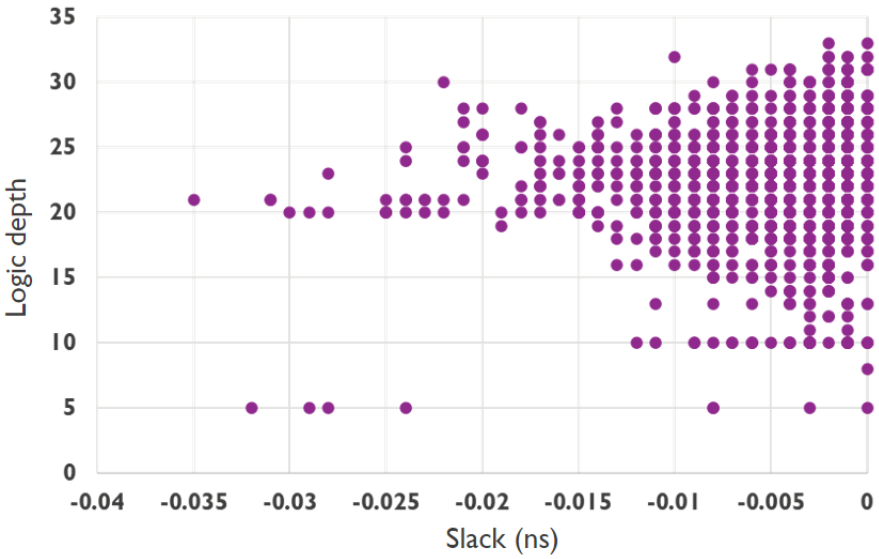

Figure 4.6. Logic depth vs slack.

highest logic depth. In an ideal scenario where interconnect delays between cells are negligible, the path delay will solely be determined by the sum of the FEOL delays, thus defining the maximum achievable frequency of the design. In reality, also interconnects play a role, contributing to the path delay.

Since the target frequency is constrained, while the logic depth increases, the length of the nets should decrease to keep the sum of the FEOL and BEOL delay to be below the clock frequency threshold. This implies that the optimization tool is forced to place the cells closer, trying to ensure a smaller FEOL delay due to the wiring C as well as a contained interconnects delay.

4.4 Slew vs BEOL delay

The signal transition time, also known as slew, is a crucial parameter in digital circuits, representing the time it takes for a signal to cover the transition between 30% and 70% of its amplitude during rising or falling edges. In Electronic Design Automation (EDA) tools, slew is typically measured in nanoseconds (ns), and an ideal slew value is about 0 ns.

Figure 4.7 illustrates the relationship between BEOL delay and slew degradation. Slew degradation refers to the difference in slew between the end and start points of

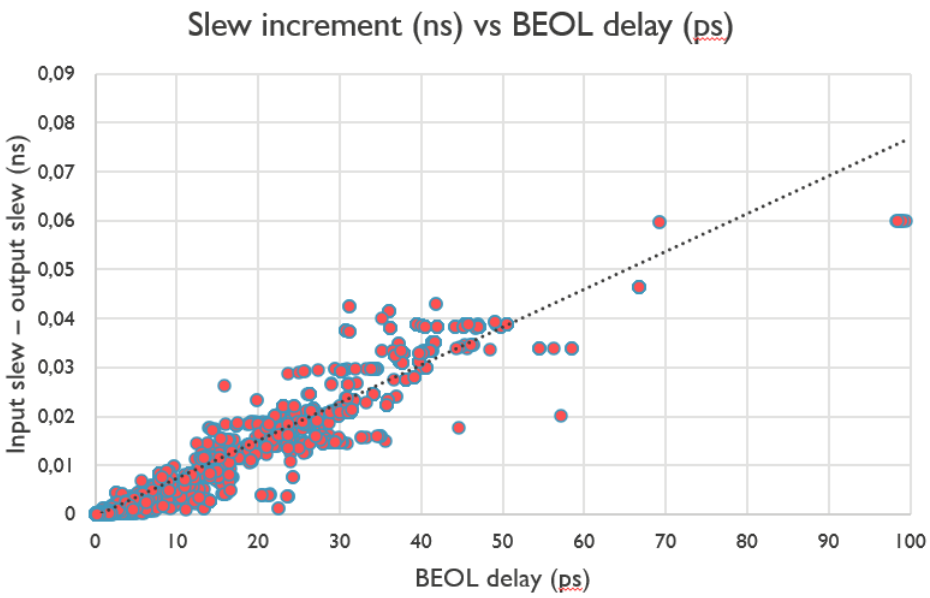

Figure 4.7. Slew degradation along interconnects compared to BEOL delay. Input slew refers to the signal slew at the load logic gate's input pin, while output slew is computed at the source logic gate's output pin. Their difference illustrates signal slew increase during interconnect travel.

the interconnects. This graph visually presents how the slew degradation changes with varying levels of BEOL delay. This degradation along the BEOL can be likened to a signal traveling through a series of RC low-pass filters, gradually reducing the signal's steepness.

To mitigate this degradation effect and prevent slower switching speeds in subsequent cells, it is important to avoid using large RC nets. To improve slew once it has been

degraded, two approaches can be considered: using cells with very small loads to reduce the load capacitance and speed up the signal transition; utilizing cells with high drive strength, which can overcome the increased load capacitance and achieve faster switching times.
Buffers are commonly employed to enhance signal speed by leveraging the first method. In a two-stage configuration, comprising two inverters arranged in series, the first inverter enhances the input slew of the subsequent cell. This is achieved because the first inverter only perceives the pin capacitance of the connected inverter as its load. The heightened input transition time enables the second inverter to switch more swiftly, even when faced with a substantial load.
A detailed study about the cell transition time (slew) is presented in chapter 6.

Chapter 5

Enhanced Ring Oscillator (eRO)

This chapter presents a new methodology to map statistical delay contributions into a Ring Oscillator (RO) simulation. The use of the new statistics leads to a new mapping concept which has been called enhanced Ring Oscillator (eRO) thanks to its improved accuracy and reliability in describing the delay contributions of the BEOL.

5.1 RO - Structural mapping

Traditionally, ring oscillator simulations were built by mapping metal length distributions into a π shape symmetric topology as shown in the figure 2.4.

In the simple case of 1 or 2 metal layers with roughly the same R and C values per unit lengths, representing the BEOL stack in the very first technology nodes, mapping the metal lengths was corresponding to mapping the delay. This is because the delay is RC and either R and C are proportional to the metal lengths. Building a simple R and C circuit, we know that changing the lengths is equal to change the delays by a proportional factor, keeping in mind that the delay scales with the square of the length.

When the BEOL stack started to become more complex with metal layers with different R and C values, the problem of representing critical paths BEOL statistics into a Spice simulation which can well mirror the path delay contributions due to each metal layer started to become challenging. The π shape topology, previously shown in figure 2.4, was needed to get closer to the fact that the signal is typically going up and down along the metal layers, starting from the output pin of the source cell and reaching the input pin of the next sink cell as also shown in the example 2.5.

Typically higher metal layers (Mz) are a lot less resistive than lower metal layers (Mx) as shown in 3.2 where, for the chosen geometries, the R per μm can vary up to 30 times. This leads to the consequence that the metal utilization is varying a lot depending on the net length so that different R and C values can correspond to every net length depending on its metal distribution.
To address this issue, different ring oscillators were built, binning the metal distribution for different net lengths. Longer the net lengths and the higher the metal layers used for routing.

The π BEOL topology was also built splitting the lengths of the same metal layers into two equal parts, half for the way up and the other half for the way down.
To complete the picture, a FO 3 was chosen and it was represented by 2 branching sinks at the end of the net, just before the 'main' sink.

Studying the RO frequency shift with the change of the branching position, it is showing relevant differences(i.e. frequency shift varying from 5% to 100% improvement for a given change of R and C). Thus, the importance of the branching position choice for the building of a reliable model which is well representing the statistics of the analyzed paths was clear.

A possible solution may be to sweep the branching position, similarly as done for the net lengths, building different ROs based on different subgroups.
But then another problem arises which is how to properly average the results obtained by the different ring oscillators. Recalling that the final purpose of performing ring oscillator simulations in our case is for benchmarking new technologies, theoretically only one value is needed to evaluate and compare two solutions.

It becomes evident that further statistical studies are needed to properly weigh the different ring oscillator simulations in a faithful way.

Looking for trends and dependencies of the available parameters, it became clear that there are too many correlations involved in the optimization and so the majority of the design statistics seem to be highly subject to casualties. This is reasonable if one thinks that the PnR optimization is exploiting heuristics and several optimization techniques, including the use of cost functions which are defining ways to prioritize the net to be optimized first or the ones which should not be touched anymore [15].

To go around these difficulties, a new RO mapping concept is discussed in the next section 5.2. It exploits the new extraction flow to build a BEOL equivalent circuit which is drastically more accurate in expressing the delay R and C contributions of the real interconnects of a digital design routed database thanks to the new extracted statistics of delay contributions.

5.2 eRO - Mathematical mapping

In this section, we discuss a methodology to map the BEOL RC contributions shown in figure 3.1 and table 3.1 into an equivalent ring oscillator.

The RC contributions to the BEOL (net) delay contain all the dependencies of the delay on:

- Metal lengths and number of vias
- Metal and via positions
- Net topology with branching positions and FO
- 'Main' R and C factors and branch C factors

Mapping these RCs into an equivalent RC load can lead to a simple Spice simulation which is intrinsically representing all the characteristics of the extracted paths and nets. This mapping method is not including fundamental assumptions. The only contribution which cannot be assessed yet is the via C. The via C contributions are considered negligible since typically of the order of few aF per via.

It is worth noticing that only one average is performed just on the initial sample based on the chosen paths and nets analyzed. Instead, in the case of the previous RO mapping methodology, a first metal distribution average was done on the net lengths and then a weighted average of the results of the RO simulations is required to have only one value which should be used for benchmarking.

The eRO represents how in average the delays of the analyzed paths or nets are shifting when the technology changes.
Using these data it is clear that the delay contributions are properly tackled, leading to a more accurate mapping.

As previously mentioned, since the RC contributions to the BEOL (net) delay contain all the dependencies of the delay on the net characteristics, a complex RC structure is not needed anymore. Thus a simple RC load 5.1 is chosen with R_{tot} and C_{tot} being

$$R_{\text{tot}} = \sum_{i=1}^{\#\text{ metal layers}} R_{\text{metal}_i} + \sum_{i=1}^{\#\text{ metal layers - 1}} R_{\text{via}_i} \tag{5.1}$$

$$C_{\text{tot}} = \sum_{i=1}^{\#\text{ metal layers}} C_{\text{metal}_i} + C_{\text{branch pins}} + C_{\text{main pin}} \tag{5.2}$$

where

$$R_{\text{metal}_i} = R_{\text{metal}_i}^{\text{per unit length}} \cdot L_{\text{metal}_i}^{\text{R}} \tag{5.3}$$

$$R_{\text{via}_i} = R_{\text{via}_i}^{\text{per via}} \cdot L_{\text{via}_i}^{\text{R}} \tag{5.4}$$

$$C_{\text{metal}_i} = C_{\text{metal}_i}^{\text{per unit length}} \cdot L_{\text{metal}_i}^{\text{C}} \tag{5.5}$$

The resistance (or capacitance) corresponding to each metal layer is computed by multiplying the resistance (or capacitance) per unit length with a coefficient that represents its length. The resistance of the vias is the product of the resistance per unit of via and a length coefficient representing the number of vias. Since this is an equivalent delay circuit, the lengths and the number of vias are not physically representative in absolute value since they will depend only on the choice of R_{tot} and C_{tot}.

The L coefficients are the unknown values to be determined through a system of equations written by computing the RC contributions to the (Elmore) BEOL delay as done for all the nets of the extracted critical paths. The RC BEOL delay contributions

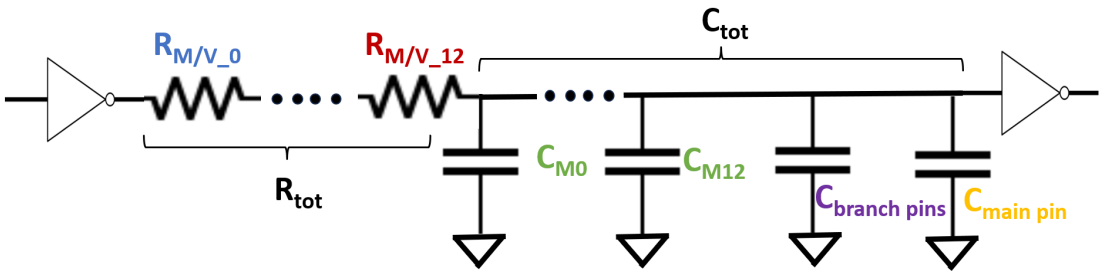

Figure 5.1. Mathematical mapping - Ring oscillator i^{th} stage RC load .

for the topology in 5.1 can be written as:

$$RC_{\text{R metal}_i} = \frac{R_{\text{metal}_i} \cdot C_{\text{tot}}}{\text{BEOL delay}} = \frac{R_{\text{metal}_i} \cdot C_{\text{tot}}}{R_{\text{tot}} \cdot C_{\text{tot}}} = \frac{R_{\text{metal}_i}}{R_{\text{tot}}} \tag{5.6}$$

$$RC_{\text{R via}_i} = \frac{R_{\text{via}_i} \cdot C_{\text{tot}}}{\text{BEOL delay}} = \frac{R_{\text{via}_i} \cdot C_{\text{tot}}}{R_{\text{tot}} \cdot C_{\text{tot}}} = \frac{R_{\text{via}_i}}{R_{\text{tot}}} \tag{5.7}$$

$$RC_{\text{C metal}_i} = \frac{R_{\text{tot}} \cdot C_{\text{metal}_i}}{\text{BEOL delay}} = \frac{R_{\text{tot}} \cdot C_{\text{metal}_i}}{R_{\text{tot}} \cdot C_{\text{tot}}} = \frac{C_{\text{metal}_i}}{C_{\text{tot}}} \tag{5.8}$$

$$RC_{\text{branch/main pins}} = \frac{R_{\text{tot}} \cdot C_{\text{b/m pins}}}{\text{BEOL delay}} = \frac{R_{\text{tot}} \cdot C_{\text{b/m pins}}}{R_{\text{tot}} \cdot C_{\text{tot}}} = \frac{C_{\text{b/m pins}}}{C_{\text{tot}}} \tag{5.9}$$

$RC_{\text{R metal}_i}$ is the delay contribution due to the resistance of the i^{th} metal layer. Instead $RC_{\text{C metal}_i}$ is the delay contribution due to the capacitance of the i^{th} metal layer. Similarly $RC_{\text{R via}_i}$ is the delay contribution due to the resistance of the i^{th} via layer. $RC_{\text{branch/main pins}}$ are the delay contributions due to the pins of the branching sinks and the pin of the main sink. In total, 40 equations are written: $13 \cdot 2$ equations for the metal layers R and C; 12 equations for the via layers R; 1 equation for the C of branching pins; 1 equation for the C of the main pin. These linear systems of 40 equations lead to the ratio of L factors which are needed to build an equivalent ring oscillator that is experiencing the same delay contributions as the average of the extracted critical paths with negative slack of a given design. In this way, the frequency shifts which is experiencing a ring oscillator should be, in average, the same as the ones experiencing the extracted critical paths. In this way, the accuracy of the RO results should be much more accurate than the ones obtained using the structural mapping explained in the section 5.1. It is important to note that the 40 equations are fixing the relative R and C values with respect to R_{tot} and C_{tot}. Thus R_{tot} and C_{tot} are free parameters and regardless of their choice, the delay contributions will be kept.

5.2.1 Total R and C choice

The choice of R_{tot} and C_{tot} doesn't determine the ability of the RO to experience the same BEOL delay shifts as the ones experiencing the extracted sample of paths and nets which, in any case, are tackled. Nonetheless, C_{tot} needs to be fixed to ensure a realistic capacitive net load that is sitting in front of a cell. The total capacitance loading of a

cell determines the cell delay and the output slew of a cell for a given input slew. Bigger the load and higher the cell delay and the slew degradation for a cell with a given drive strength. R_{tot} can be chosen to achieve an RC load that is structurally representative so that the slew degradation along the interconnects is tackled as well. It is essential to keep consistent cell delays. To do so, it can be chosen based on the average of the total main of the nets.

Another option is to choose Rtot so as to include another kind of estimation obtainable as a result of the ring oscillator simulation. For example, one could opt to include predictions of power consumption. Typically is very difficult to predict power consumption trends because they include all the chip paths. The total power consumption is depending on all the chip characteristics which are contributing to it. No direct correlation with critical paths is known yet to determine the power consumption of a chip only from the statistics of critical paths. Thus an indirect correlation, depending on the given design, needs to be researched to achieve that aim. Power represents an important concern in modern semiconductor technology research activities.

The research is pushing for more sustainable solutions which are able to keep high performances or improve battery lifetime for low-power applications.

5.2.2 Selective sampling of critical paths and nets

In this paragraph, the potentialities of different groups of critical paths and nets are commented on to achieve different types of studies, depending on the need of a designer. The non-uniform characteristics of the nets composing the critical paths make it difficult to find a general fingerprint of CPs.

Negative slack CPs

The extraction flow can be performed over any collection of critical paths. Each collection group is created using the report_timing built-in command for a definable group of critical paths which are satisfying certain timing constraints. In this case, for the group of critical paths from register to register, all critical paths with slack ≤ 0 are extracted. In particular, all the timing and structural information of all the nets and timing arcs (delay between two timing points) are saved into a structured text file. Sampling the worst critical paths leads to statistics which is describing the maximum performances of the design under study. By analyzing the characteristics of those paths it is possible to understand the behavior of the current design and which are the bottlenecks that should be improved to achieve better performances or the trade-offs to take into account to reach the best optimization attainable through the use of a new boosting technology or new BEOL geometries. Moreover, predictions on the impact of a new device technology on the max performances could be performed on a given system design.

Long nets

The CPU design being studied does not incorporate macros or memories and is characterized by the presence of short nets. The average length of these nets is approximately

4.2 micrometers, with variations ranging from fractions of micrometers to several dozen micrometers.

Figure 4.2 illustrates the Cumulative Density Function of net lengths in the main portion of the design, with the data organized based on the drive strength used. As expected, larger nets necessitate higher drive strengths due to their higher capacitance. Analyzing the statistics of long nets allows for insights into how the optimization tool handles such cases and enables predictions regarding the behavior of more substantial designs dominated by long nets. This is valuable because, within a specific technology and with a constant availability of wiring resources, the optimization tool should consistently find the most efficient way to route two distant cells. This may involve utilizing higher metal layers with lower resistance to reduce the BEOL delay, minimizing wiring lengths to keep the wiring load low, and selecting appropriate drive strengths to restrain the delay of source cells.

Developing such a model could potentially predict the behavior of larger designs exploiting the current technology, even without access to the fully routed database. However, it remains crucial to validate the model using actual large designs.

In practical scenarios, during the tape-out process, companies assign a specialized group of engineers to manage very long routed nets. These engineers prioritize routing such nets with highly designed metal layers (Mz) tailored for enhanced speed. These specific metal layers may have relaxed dimensions, such as wider and taller lines with a more lenient metal pitch, ensuring exceptionally fast operation.

High FO stages

As previously discussed, the placing and routing optimization of high FO stages is critical to keep the delay of these stages low by both reducing the wiring length and using low resistive metal layers.

The high FO characteristics also imply that these stages belong, at least, to a number of critical paths which is equal to the fan-out number. Moreover, more is the sharing of resources, meaning that the same stage belongs to multiple paths, more important it is to optimize that stage which will impact the delay of many paths. It should be intuitive that high FO stages should be prioritized during placement and routing steps. This turns out to not to be really the case because high FO stages have big nets which lead to a high cell and interconnects delay due to the reason explained in the previous paragraph regarding long nets. Studying the statistics of these stages one could find an optimal FO which leads to the best tradeoff between placement difficulties and delay minimization. In other words, for a fixed floorplan, letting the tools to chose its optimal FO and LD combination, may lead to a suboptimal solution because of the lack of good optimization in connecting many cells. To improve the result, EDA tools should prioritize the placement of high FO stages so to ensure affordable BEOL and FEOL delays.

Developing a model which could mirror the EDA tool behavior in optimizing these cases is essential to understand the link between the RTL output, being the FO, LD, and the total number of cells, with the placing optimization which at the end will leads to wire length minimization for load minimization. In this way, the metal length could be used as a placement figure of merit without entering into a detailed physical distances analysis.

Nonetheless, the latter, if compared with the net lengths, could lead to understanding how well the tool is able to find the shortest path to connect two cells.

However, some RTL designs may be more parallelized and others may be more sequential. Thus some would likely require more FO stages and others less. The idea is to compare how the distribution of wire lengths changes as a function of the FO of different RTL designs.

Chapter 6

Scaling factors - Modeling .lib and .ict scaling effects on PnR

The aim of this chapter is to investigate how the PnR flow globally reacts to input changes, specifically to device characteristics changes such as cell delays and transition times (slew), keeping constant all other parameters and trying to make the tool experience the same optimization effort. First, the operating conditions are defined. Then an in-depth understanding of the parameters present in the liberty (.lib) file is discussed.
A methodology to predict the actual target frequency is presented based on operating points which are chosen with respect to the average statistics of the design under study. The computed predictions are compared with the actual changes in the target frequency. Moreover, a proof of concept about how to correlate the delay shift coming from C and R changes is introduced. Relevant comparisons between the statistics of the obtained databases are shown and discussed.

6.1 Impact of cell delay and transition time scaling

When the impact of a new device technology needs to be addressed, the full PnR flow needs to be performed with all the new device-related files. One of these is the .lib file. It contains the timing and power parameters associated with any cell which will be available for the router. The values are obtained through cell simulations under different operating conditions.

To evaluate a new cell technology, the trade-off between power, performance, and area must be analyzed. The performances depend on the electrical and timing parameters. The most important electrical parameter which is still present in the .lib file is the cell pin capacitance. A higher drive strength in a cell leads to increased pin capacitance. This drive strength determines the cell's capability to deliver current for charging or discharging the capacitive load in front of it. Achieving this is possible either by widening the cell or employing multiple minimum-width transistors in parallel. The choice between these methods depends on the cell technology, with the selection based on achieving the minimum area and, consequently, the lowest parasitics from a layout perspective.

The timing parameters are the cell rise/fall delays and the rise/fall transition times. The idea is to scale these timing values so that to see how the full optimization reacts to these changes. A downscaling factor of 20% will be applied and three new databases will be generated running a new full PnR flow from synthesis to placement to routing. The first database will have as input the .lib file where cell delays are scaled by 20% the second flow has as input the lib file where transition times are scaled and the last flow will be run with a liberty file where both cell delays and transition times are scaled by 20%. This scaling methodology is not realistic from a device standpoint of view but all the values will track to some degree the changes in device characteristics. Nonetheless, the aim of this study is to analyze how the PnR flow reacts to changes in the individual device characteristics than being faithful to the underlying physics.

To properly perform this study, the target frequency of the new designs needs to be carefully adjusted. This experiment is looking for reaching the same optimization effort performed by the EDA tool which means it is looking for roughly the same number of critical paths of the reference database where the liberty file is not scaled. The optimization engine behaves differently depending on how pushed the design is.

6.2 Liberty file

In this section, the timing parameters of the liberty (.lib) file are analyzed in depth. Each instance in the file is described with specific timing, power, and pin capacitance parameters. While the pin capacitance is represented by a single value, the timing and power parameters are expressed using look-up tables (LUTs).

For cell delays and transition times, the LUTs are presented in the form of 7x7 matrices. An example of this can be observed in Figure 6.1, which displays the LUTs for the cell rise delay and rise transition time of a buffer with drive-strength 1.

Cell delay (ps)

Input slew (ps) \ Load (fF)	0.5	1.15725	2.67848	6.19936	14.3485	33.2097	76.8641
1	5.1179	7.22979	12.0569	23.2069	49.0037	108.704	246.884
2.48541	5.98733	8.09524	12.9229	24.0746	49.8723	109.578	247.766
6.17725	7.50464	9.6448	14.4914	25.6048	51.4051	111.107	249.296
15.353	9.93295	12.0864	16.8903	28.0418	53.828	113.532	251.706
38.1584	14.0759	16.326	21.1765	32.3215	58.0651	117.777	255.917
94.8393	21.0925	23.6927	28.6929	40.0465	65.9496	125.659	263.871
235.714	33.6245	36.8567	42.481	54.2991	81.1267	141.49	279.654

delay < 10 ps
10 ps < delay < 20 ps
20 ps < delay

→ Max load ≃ 1.2 fF
→ Max FO ≃ 3
(considering 0.5 fF per sink cell)

Cell transition time (ps)

Input slew (ps) \ Load (fF)	0.5	1.15725	2.67848	6.19936	14.3485	33.2097	76.8641
1	1.90674	3.68539	7.85131	17.5083	39.8388	91.5372	211.131
2.48541	1.91499	3.68665	7.85196	17.503	39.84	91.5419	211.193
6.17725	1.98536	3.72739	7.85443	17.5027	39.8397	91.5847	211.193
15.353	2.12926	3.77751	7.90722	17.5365	39.8404	91.5422	211.193
38.1584	2.49766	3.97781	7.99349	17.5895	39.8967	91.5422	211.193
94.8393	3.28038	4.61527	8.39876	17.9768	40.0558	91.668	211.192
235.714	4.61485	6.04428	9.41411	19.0449	41.4022	92.2407	211.42

slew < 2.5 ps
2.5 ps < slew < 15 ps
15 ps < slew

Figure 6.1. Cell delay and transition times of a BUFF D1.

These LUTs are used to replace non-linear functions describing the cell delay and transition time variations as a function of the input slew (transition time) and the total capacitive load at the output pin of the cell.

Both cell delays and transition times exhibit an increase as the load and the input transition time increase. Notably, cell delay values below 10 ps are highlighted in green, indicating fast cells. However, when the cell delay is up to 20 ps, it suggests that the cell is loaded with a relatively large capacitance, possibly requiring a cell with higher drive strength for optimal timing. Delays exceeding 20 ps may imply that the net is only present in critical paths with positive slack, thus not requiring further optimization. Alternatively, the optimization tool may have abandoned efforts to optimize it with a cell of higher drive strength, possibly due to space constraints in inserting a beneficial buffer.

For optimal performance, it is recommended to choose a maximum load of about 1.2 fF for these cells. This corresponds to a fan-out between 2 and 3, considering that the average load per sink cell is around 0.5 fF/cell. The consideration of capacitance per cell is essential when the fan-out is increased, as it accounts for the additional capacitance contributed by branching wires connecting to the main net part.

Similarly, cell transition times are highlighted in green when they are smaller than 2.5 ps, ensuring that subsequent stages will also have low delays when the capacitive load is low. This value is somewhat arbitrarily chosen to maintain low delays in the next stage.

It is important to consider the degradation of the slew while traveling along the BEOL, as shown in Figure 4.7. Consequently, orange values in the LUTs are considered sub-optimal, and red values are instances where the optimization tool has given up on further optimization due to diminishing returns compared to the required effort.

Upon closer analysis, it becomes evident that the absolute differences along the columns and rows in the LUTs are independent of each other. This independence leads to the later discussion of a linearization method, which assumes that both delays and transition times are independent functions of delay and slew. In the case of delays, the difference in input slew seems to be translated into an almost constant additional delay contribution which is added to the cell delay. In the case of output transition times, their dependency on the input slew diminishes as the output load increases. Moreover, it is slightly increasing with the input slew.
This is the case of multi stages instances in which the input transition time is strongly decoupled with the output transition time thanks to the presence of a node in between the two inverters composing the buffer.

Analyzing the timing characteristics of the BUFF D16 and comparing them with the BUFF D1, it becomes evident that the performance of BUFF D16 relies less on output capacitance while maintaining a relatively consistent sensitivity to input slew. This observation reaffirms that in the context of buffers, variations in slew can lead to an almost constant additional delay contribution. The higher drive strength can be appreciated seeing that the green area is bigger in both tables. The same considerations as before are still valid making them to be generalized to all the drive strengths of buffers.

In this case, the maximum load that doesn't lead to a very high delay is about 22 fF which corresponds to a maximum FO of 37. A load per cell of 0.6 fF has been used in this computation because the average load per sink cell increases with the load. These

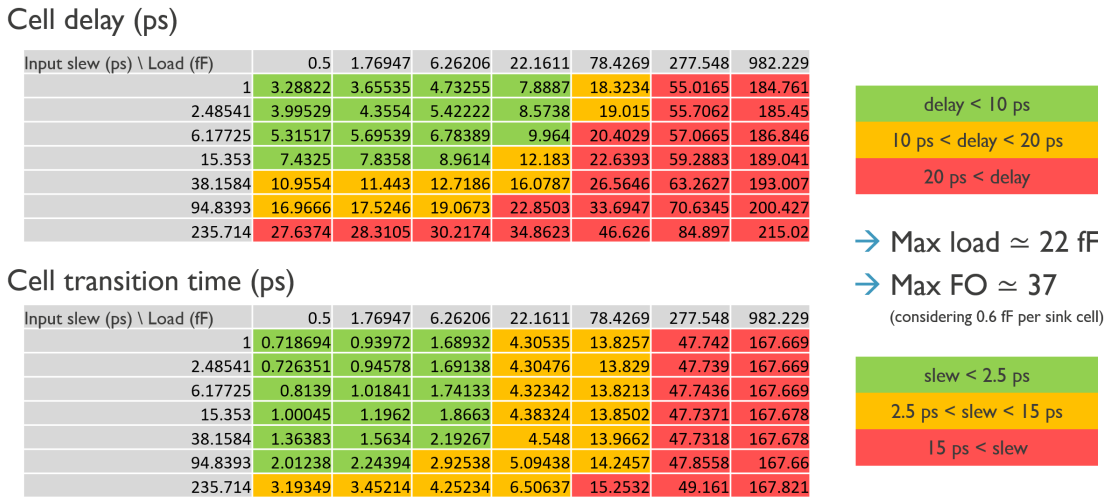

Figure 6.2. Cell delay and transition times of a BUFF D16.

considerations are consistent with the plot previously shown in figure 4.1 where is clear that higher drive strengths are chosen for higher loads to keep a low BEOL delay.

Knowing that the slew is important to determine the cell delays and that it is degraded along the BEOL, it must be restored along the cells to avoid forcing the following stages to be slower. In the figure 6.3 are shown the transition times LUTs for the BUFF D1 and D16. In green are highlighted the transitions in which the output transition time is smaller than the input transition time. In orange when both transition times are very similar and in red when the output transition times are worse than the input ones. The load capacitance plays a crucial role in determining the output transition time. A bad transition time can be easily buffered exploiting a stage with a very low load. This buffering cell will dramatically improve the slew paying a cost in terms of delay which is proportional to the input slew. On the contrary, a very high load leads to bad output transition times. Higher drive strengths are more likely to improve the slew paying a cost in area and power consumption. In multi-level logic, such as in the case of buffers, the output slew is highly independent of the input slew. Moreover, as the DS increases, the correlation between Ys and C weakens, while the correlation between Ys and Si strengthens. Optimally the slew should be kept constantly low along all the stages of a path to have the best cell performances.

In summary, the output slew and the cell delay are functions of the input slew and total capacitive load. The input slew depends on the output slew of the previous stage and the slew degradation experienced along the interconnects. The degradation is proportional to the RC BEOL delay. The capacitive load is heavily impacting slew and delay values. It is the sum of loading pins and wires. The number of sinks is defined from the FO and the pic capacitance increases with the DS of the sink. The wire capacitance can be considered proportional to the wiring length because, in the BEOL stack under study,

Cell transition time (ps)

BUFF D1

Input slew (ps) \ Load (fF)	0.5	1.15725	2.67848	6.19936	14.3485	33.2097	76.8641
1	1.90674	3.68539	7.85131	17.5083	39.8388	91.5372	211.131
2.48541	1.91499	3.68665	7.85196	17.503	39.84	91.5419	211.193
6.17725	1.98536	3.72739	7.85443	17.5027	39.8397	91.5847	211.193
15.353	2.12926	3.77751	7.90722	17.5365	39.8404	91.5422	211.193
38.1584	2.49766	3.97781	7.99349	17.5895	39.8967	91.5422	211.193
94.8393	3.28038	4.61527	8.39876	17.9768	40.0558	91.668	211.192
235.714	4.61485	6.04428	9.41411	19.0449	41.4022	92.2407	211.42

BUFF D16

Input slew (ps) \ Load (fF)	0.5	1.76947	6.26206	22.1611	78.4269	277.548	982.229
1	0.718694	0.93972	1.68932	4.30535	13.8257	47.742	167.669
2.48541	0.726351	0.94578	1.69138	4.30476	13.829	47.739	167.669
6.17725	0.8139	1.01841	1.74133	4.32342	13.8213	47.7436	167.669
15.353	1.00045	1.1962	1.8663	4.38324	13.8502	47.7371	167.678
38.1584	1.36383	1.5634	2.19267	4.548	13.9662	47.7318	167.678
94.8393	2.01238	2.24394	2.92538	5.09438	14.2457	47.8558	167.66
235.714	3.19349	3.45214	4.25234	6.50637	15.2532	49.161	167.821

Slew out < Slew in
Slew out ≃ Slew in
Slew out > Slew in

Figure 6.3. Transition times of a BUFF D1 and D16.

the capacitances per unit length are not dramatically varying between the metal layers. Finally, the wire length depends on the placement, the physical distance between the sinks, and the wiring congestion which can lead to obstacles to get around.

6.3 Achievable frequency predictive models

In this section a methodology for predicting how the achievable frequency changes as a function of the timing parameters present in the lib file is proposed. The idea relies on the linearization of the LUTs around an average operating point of input slew and output slew. The operating point is composed of the average total capacitance and the average input slew of all the cells belonging to the critical paths with negative slack. Sampling the latter is crucial to determine how on average the worst critical paths will shift because most of them will remain slow so to determine the new target frequency of the database.

The proposed method is based on the scaling factors (SF) defined in 6.1

$$\mathrm{SF}_{\frac{Y}{X}} = \frac{\frac{Y_{\mathrm{high}} - Y_{\mathrm{low}}}{Y_{\mathrm{high}}}}{\frac{X_{\mathrm{high}} - X_{\mathrm{low}}}{X_{\mathrm{high}}}} \tag{6.1}$$

where $Y = \{\text{cell delay (D)}, \text{output slew } (S_{\mathrm{out}})\}$ and $X = \{\text{C load (C)}, \text{input slew } (S_{\mathrm{in}})\}$.

The scaling factors represent how the Y variable will relatively change when the X variable is scaled. In other words, these coefficients are multiplied by the scaling coefficient that is applied to the input. This multiplication aids in recovering the scaling coefficient of the desired output variable. When dealing with predictions of target frequencies, the focus is on how the delay of the cells scales in relation to the other scaled variables. For simplicity here all the Y(X) are supposed to be independent functions. For example, the

scaling factor delay/slew for a BUFF D1 at the operating point of 2.2 fF, 6.2 ps is:

$$\mathrm{SF}_{\frac{D}{S_{\mathrm{in}}}} = \frac{\frac{D_{\mathrm{high}} - D_{\mathrm{low}}}{D_{\mathrm{high}}}}{\frac{S_{\mathrm{in,\ high}} - S_{\mathrm{in,\ low}}}{S_{\mathrm{in,\ high}}}} = \frac{\frac{9.64-8.09}{9.64}}{\frac{6.17-2.5}{6.17}} = 0.27 \tag{6.2}$$

Computing the scaling factors for all the combinations of Y/X parameters, the following results shown in 6.1 are obtained for buffers and inverters of different drive strengths.

	INV D1	INV D8	BUFF D1	BUFF D16	Chosen
$\mathrm{SF}_{\frac{D}{S_{\mathrm{in}}}}$	0.4	0.41	0.27	0.36	0.35
$\mathrm{SF}_{\frac{D}{C}}$	0.66	0.58	0.65	0.51	0.65
$\mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}}$	0.2	0.52	0.02	0.12	0.2
$\mathrm{SF}_{\frac{S_{\mathrm{out}}}{C}}$	0.78	0.54	0.95	0.64	0.85

Table 6.1. Scaling factors for INV and BUFF cells and ones chosen for predictions

It is evident that the scaling factors are varying from cell to cell. The choice of the scaling factors to use for target frequency predictions is determined knowing that the most used cells in the current design are D1 and D2 cells as already shown in 4.3. They are shown in the last column of the table 6.1.

So, knowing the scaling factors we can predict how the target period shifts when the timing parameters of the liberty file are scaled down by, for example, 20% by multiplying this value with all the scaling factors until when the delay scaling is reached for each parameter. The scaling factors correlate the outputs of the LUTs as a function of the input of the LUTs.

For the slew, when it is scaled, not only the output slew is reduced, but in real case scenario where the output slew travels along the RC and then enters the next cell, it becomes an input slew, reducing the output slew even more, thus a secondary effect should be considered as a result of the slew propagation along a path. To complete the picture, here is proposed a method to predict how the input slew of the cells scales, overall, when the output slew of the cells is scaled.

As already mentioned, the slew at the output of a cell will propagate along the BEOL RC network until reaching the input pin of the next cell. The degradation along the RC is adding to the initial slew an absolute amount proportional to the BEOL delay of the RC 6.3.

$$\text{input slew} = \text{output slew}(72\%) + \text{slew degradation}(28\%) \tag{6.3}$$

When the transition time LUTs are scaled, the RC is not supposed to change, so only the contribution coming from the output slew of the previous cell is scaled and not the degradation contribution. On average, 72% of the input slew of the cells is coming from the output slew contribution, the remaining 28% comes from the slew degradation along the RC.

To predict how much the cell delays are changing when the output slew is scaled down, the input slew needs to be studied because it bridges the output slew with the cell delay. To model the slew chain effect dependency, a feedback loop is modeled as in the picture 6.4 In this circuit, all the signals represent a relative scaling of a quantity. 'K' is

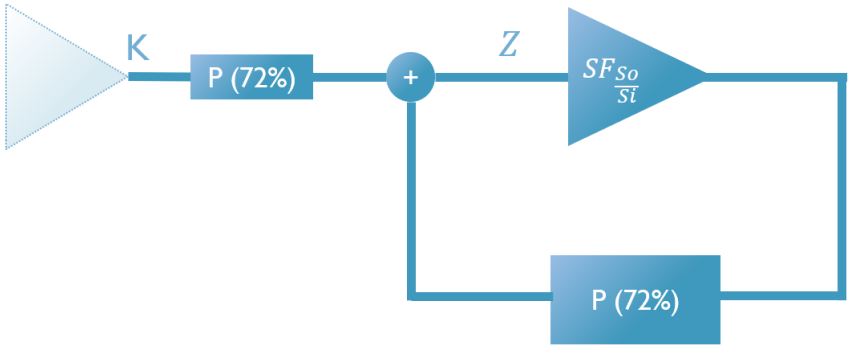

Figure 6.4. Feedback loop model for the propagation of slew variations due to the improved output slews.

the scaling applied to the LUTs of the transition times. 'P' is a block that multiplies its input by a factor, in this case 0.72. The inverter is represented by the scaling factor of the slews. It converts the relative variation of the input slew into a relative variation of the output slew. The feedback represents the ideal case of an infinite number of stages. This model is described by the equation:

$$Z = K \cdot P + Z \cdot \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P \tag{6.4}$$

Rewriting the equation explicating Z as a function of K, one gets

$$Z = \frac{K \cdot P}{1 - \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P} \tag{6.5}$$

Z(K) expresses how the input slew scales when the output slew is scaled. Thus the scaling factor $\mathrm{SF}_{\frac{S_{\mathrm{in}}}{S_{\mathrm{out}}}}$ of the input slew with respect to the output slew scaling is:

$$\mathrm{SF}_{\frac{S_{\mathrm{in}}}{S_{\mathrm{out}}}} = \frac{P}{1 - \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P} = \frac{0.72}{1 - 0.2 \cdot 0.72} = 0.84 \tag{6.6}$$

The delay scaling case doesn't need any scaling factor because it has a direct impact on the target period. One only has to remember which is the average FEOL delay contribution over the path delay contribution which is around 25% as shown in 3.1.

The achieved value of the actual target period is the one which leads to roughly the same number of critical paths as the original database. By doing so, one assumes that the effort in optimizing the design would be roughly the same so that the results are not skewed by other factors. So the new $f_{achieved}$ corresponds to $\frac{1}{T_{achieved}}$.

	Delay	Slew	Delay & Slew
# CPs	1508	896	1538
$T_{achieved}$	$T_{ref} \cdot 0.83$	$T_{ref} \cdot 0.94$	$T_{ref} \cdot 0.77$
$T_{predicted}$	$T_{ref} \cdot 0.85$	$T_{ref} \cdot 0.956$	$T_{ref} \cdot 0.81$

Reference
1272
T_{ref}

Table 6.2. Critical paths (CPs) with negative slack, achieved and predicted target periods for each new PnR run compared to the original reference database

The results obtained by running the full PnR flow after scaling by 20% the cell delays, the transition times, and both together in the liberty file, are shown in the table 6.2.

The $T_{predicted}$ are computed as shown in 6.7,6.8 and 6.9.

$$\begin{aligned} T^{Delay}_{predicted} &= Tref - T_{ref} \cdot \left(\text{lib delay scaling} \cdot \left\langle \frac{\text{FEOL delay}}{\text{Path delay}} \right\rangle \right) = \\ &= T_{ref} \cdot (1 - 0.2 \cdot 0.75) = T_{ref} \cdot 0.85 \end{aligned} \tag{6.7}$$

$$\begin{aligned} T^{Slew}_{predicted} &= T_{ref} \cdot \left(1 - \text{lib slew scaling} \cdot \text{SF}_{\frac{S_{\text{in}}}{S_{\text{out}}}} \cdot SF_{\frac{D}{S_{\text{in}}}} \cdot \left\langle \frac{\text{FEOL delay}}{\text{Path delay}} \right\rangle \right) = \\ &= T_{ref} \cdot (1 - 0.2 \cdot 0.84 \cdot 0.35 \cdot 0.75) = T_{ref} \cdot 0.956 \end{aligned} \tag{6.8}$$

$$T^{\text{Slew \& Delay}}_{predicted} = T_{ref} \cdot \left(\frac{T^{Delay}_{predicted}}{T_{ref}} \cdot \frac{T^{Slew}_{predicted}}{T_{ref}} \right) = T_{ref} \cdot (0.85 \cdot 0.956) = T_{ref} \cdot 0.81 \tag{6.9}$$

$T^{\text{Slew \& Delay}}_{predicted}$ is obtained by multiplying the predicted values from the other two databases under the assumption that the shifts caused by the delay and the slew are independent of each other. The predictions are very close to the results. The predicted periods are larger than the achieved ones, possibly because the PnR flow can achieve even better optimization when re-executed with improved inputs. However, the uncertainty is inevitable because of the intrinsic causality and heuristic performed by the tool during the optimization.

6.4 R and C contributions to cell delay

Following the same methodology, one can compute the theoretical impact of a capacitance or resistance change on the cell delays and so also on the path delays. At the end of this study one could merge the information about the BEOL delay contributions to the path delay with the impact of the BEOL RC on the cell delays to have a complete overview on how much all the R and C circuitry elements are contributing to the path delay. This knowledge can lead to a complete picture of the design under study needed to benchmark a new BEOL or FEOL technology or improve the performances of the design through target optimizations. When going through research or market, a cost analysis needs to be assessed to determine which solution is more worth it.

The LUTs in 6.3 show that the cell delays and the output cell transition times are strongly dependent on the capacitive load in front of the cell and this dependence decreases with the increase of the drive strength. To perform the computation of the impact of R or C here we suppose that the BEOL delay is computed as a simple RC product so it will linearly scale with the scaling of R or C. Again the FEOL delays are contributing 75% to the path delay and the BEOL RC delay is contributing 25% to the path delay.

6.4.1 R scaling

The BEOL R scaling impacts the cell delays through the reduced slew degradation along the BEOL. Summing this effect with the scaling effect of the BEOL delay one could theoretically predict the global scaling of the path delay.

Again, it is necessary to model the slew chain effect dependency due to the improvement of the slew degradation along the BEOL RC. In this case the feedback loop is modeled as in figure 6.5

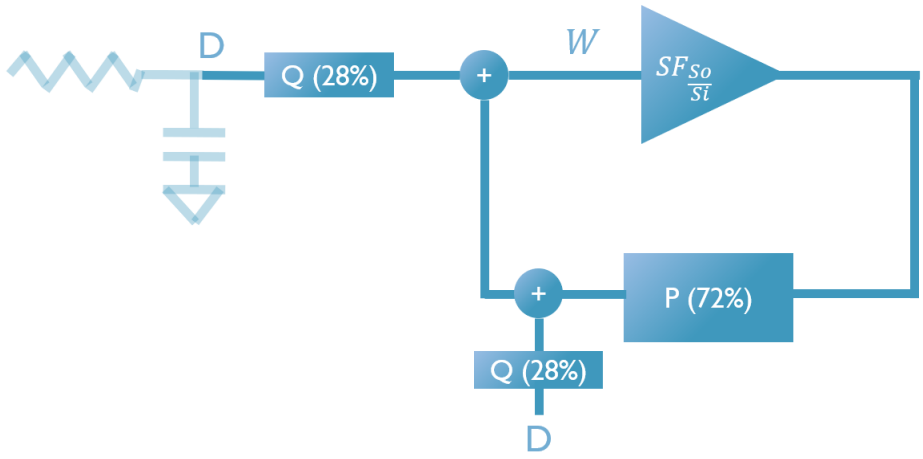

Figure 6.5. Feedback loop model for the propagation of the slew variations due to the scaled slew degradation.

Again, in this circuit, all the signals represent a relative scaling of a quantity. 'D' is the scaling applied to the R, so to the slew degradation effect. 'Q' and 'P' are blocks that multiply their input by a factor, 0.28 and 0.72 respectively. similarly, as before, the equation describing this model is 6.10

$$W = D \cdot Q + W \cdot \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P + D \cdot Q \tag{6.10}$$

Rewriting the equation explicating W as a function of D, one gets

$$W = \frac{2 \cdot D \cdot Q}{1 - \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P} \tag{6.11}$$

W(D) expresses how the input slew scales when the slew degradation is scaled.

Thus the scaling factor $\mathrm{SF}_{\frac{S_{\mathrm{in}}}{R}}$ of the input slew with respect to the slew degradation scaling due to the R scaling is:

$$\mathrm{SF}_{\frac{S_{\mathrm{in}}}{R}} = \frac{2 \cdot Q}{1 - \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P} = \frac{2 \cdot 0.28}{1 - 0.2 \cdot 0.72} = 0.65 \tag{6.12}$$

Now it is possible to write the total scaling factor which correlates the R scaling with the scaling of the path delay as 6.15:

$$\mathrm{SF}_{\frac{\text{BEOL delay}}{R}} = 1 \tag{6.13}$$

$$\mathrm{SF}_{\frac{\text{FEOL delay}}{R}} = \mathrm{SF}_{\frac{S_{\mathrm{in}}}{R}} \mathrm{SF}_{\frac{D}{S_{\mathrm{in}}}} = 0.65 \cdot 0.35 = 0.23 \tag{6.14}$$

$$\begin{aligned} \mathrm{SF}_{\frac{\text{Path delay}}{R}} &= \mathrm{SF}_{\frac{\text{FEOL delay}}{R}} \cdot \left\langle \frac{\text{FEOL delay}}{\text{Path delay}} \right\rangle + \mathrm{SF}_{\frac{\text{BEOL delay}}{R}} \left\langle \frac{\text{BEOL delay}}{\text{Path delay}} \right\rangle = \\ &= 0.23 \cdot 0.75 + 1 \cdot 0.25 = 0.42 \end{aligned} \tag{6.15}$$

6.4.2 C scaling

The C scaling directly impacts the cell delays and the slews of the LUTs, the cell delays through the slew, and the cell delay through the reduced slew degradation along the BEOL RC. As the case of the R scaling, the BEOL delay is also proportionally scaled with C.

To model the slew chain effect dependency due to both, the improvement of the output slew and the improvement of the slew degradation along the BEOL RC, the feedback circuit shown in 6.6 is used.

Remembering that all the signals represent a relative scaling of a quantity, 'G' is the scaling applied to the C, so to the slew degradation effect. 'Q' and 'P' are multiplier blocks. The model is described by the equation

$$Y = G \cdot \mathrm{SF}_{\frac{S_{\mathrm{out}}}{C}} \cdot P + G \cdot Q + (Y \cdot \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} + G \cdot \mathrm{SF}_{\frac{S_{\mathrm{out}}}{C}}) \cdot P + G \cdot Q \tag{6.16}$$

making explicit Y as a function of G:

$$Y = \frac{G \cdot 2 \cdot (\mathrm{SF}_{\frac{S_{\mathrm{out}}}{C}} \cdot P + Q)}{1 - \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P} \tag{6.17}$$

Thus the scaling factor $\mathrm{SF}_{\frac{S_{\mathrm{in}}}{C}}$ of the input slew with respect to the output slew improvement and the slew degradation scaling due to the C scaling is:

$$\mathrm{SF}_{\frac{S_{\mathrm{in}}}{C}} = \frac{2 \cdot (\mathrm{SF}_{\frac{S_{\mathrm{out}}}{C}} \cdot P + Q)}{1 - \mathrm{SF}_{\frac{S_{\mathrm{out}}}{S_{\mathrm{in}}}} \cdot P} = \frac{2 \cdot (0.85 \cdot 0.72 + 0.28)}{1 - 0.2 \cdot 0.72} = 2.08 \tag{6.18}$$

So, the total scaling factor which correlates the C scaling with the scaling of the path delay is 6.21

$$\mathrm{SF}_{\frac{\text{BEOL delay}}{C}} = 1 \tag{6.19}$$

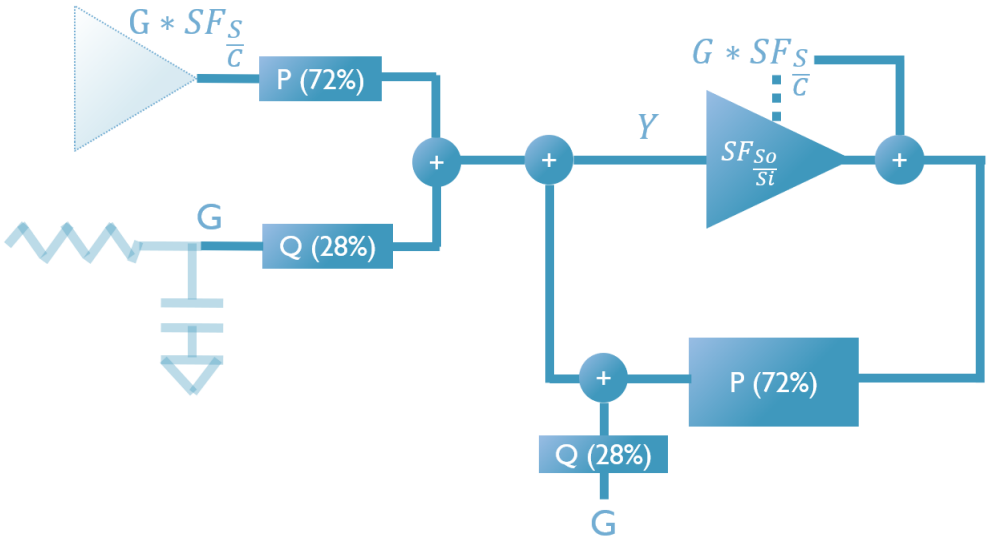

Figure 6.6. Feedback loop model for the propagation of the slew variations due to the improved output slews and the scaled slew degradation.

$$\mathrm{SF}_{\frac{\text{FEOL delay}}{C}} = \mathrm{SF}_{\frac{D}{C}} + \mathrm{SF}_{\frac{S_{\text{in}}}{C}} \mathrm{SF}_{\frac{D}{S_{\text{in}}}} = 0.65 + 2.08 \cdot 0.35 = 1.38 \tag{6.20}$$

$$\begin{aligned} \mathrm{SF}_{\frac{\text{Path delay}}{C}} &= \mathrm{SF}_{\frac{\text{FEOL delay}}{C}} \cdot \left\langle \frac{\text{FEOL delay}}{\text{Path delay}} \right\rangle + \mathrm{SF}_{\frac{\text{BEOL delay}}{C}} \left\langle \frac{\text{BEOL delay}}{\text{Path delay}} \right\rangle = \\ &= 1.38 \cdot 0.75 + 1 \cdot 0.25 = 1.28 \end{aligned} \tag{6.21}$$

Observe that this result is > 1, indicating that reducing the C by a certain factor results in a path delay reduction of more than that factor. This is expected because scaling both FEOL and BEOL delays through capacitance scaling combines with the chain effect driven by slew improvements, leading to a higher overall cell delay improvement. This cumulative effect is amplified due to the historical dependency of slew rates on preceding cells in the chain.

It's important to keep in mind that in this context, C represents the overall net capacitance. It encompasses both the interconnect capacitance and the pin capacitance, thus including both BEOL and FEOL characteristics. In contrast, R relies solely on BEOL factors. Consequently, when assessing how path delays change concerning C, one must either uniformly scale both interconnect and pin capacitances or scale one while maintaining the other constant. It's worth noting that, on average, 56% of the total net capacitance originates from pin capacitance, while the remaining 44% comes from wire capacitance. Thus, $\mathrm{SF}_{\frac{\text{Path delay}}{C}}$ can practically be divided into two components:

$$\mathrm{SF}_{\frac{\text{Path delay}}{C}} = \mathrm{SF}_{\frac{\text{Path delay}}{\text{C wires}}} + \mathrm{SF}_{\frac{\text{Path delay}}{\text{C pins}}} \tag{6.22}$$

where, using the table 3.1

$$\begin{aligned}\mathrm{SF}_{\frac{\text{Path delay}}{\text{C wires}}} &= \left\langle \frac{\text{total wires capacitance}}{\text{total net capacitance}} \right\rangle \cdot \mathrm{SF}_{\frac{\text{FEOL delay}}{C}} \cdot \left\langle \frac{\text{FEOL delay}}{\text{Path delay}} \right\rangle + \\ &+ \mathrm{SF}_{\frac{\text{BEOL delay}}{C}} \cdot \left\langle \frac{\mathrm{RC}_{\text{wires C}}}{\text{BEOL delay}} \right\rangle \cdot \left\langle \frac{\text{BEOL delay}}{\text{Path delay}} \right\rangle = \\ &= 0.44 \cdot 1.38 \cdot 0.75 + 1 \cdot 0.62 \cdot 0.25 = 0.61\end{aligned}$$

$$\begin{aligned}\mathrm{SF}_{\frac{\text{Path delay}}{\text{C pins}}} &= \left\langle \frac{\text{total pins capacitance}}{\text{total net capacitance}} \right\rangle \cdot \mathrm{SF}_{\frac{\text{FEOL delay}}{C}} \cdot \left\langle \frac{\text{FEOL delay}}{\text{Path delay}} \right\rangle + \\ &+ \mathrm{SF}_{\frac{\text{BEOL delay}}{C}} \cdot \left\langle \frac{\mathrm{RC}_{\text{pins C}}}{\text{BEOL delay}} \right\rangle \cdot \left\langle \frac{\text{BEOL delay}}{\text{Path delay}} \right\rangle = \\ &= 0.56 \cdot 1.38 \cdot 0.75 + 1 \cdot 0.38 \cdot 0.25 = 0.67\end{aligned}$$

The values of these two components show similar results. They are significantly influenced by the FO constraint and the characteristics of the BEOL stack. This implies that in the present design, using the assumed specific cell technology, RTL, and BEOL stack, the path delay contributions are well split between the interconnect capacitance and the pin capacitance.

6.5 Scaling factors summary and further validation steps

All the scaling factors of delay, slew, net C, and net R on the path delay are summarized in table 6.3.

	Scaling factors
$\mathrm{SF}_{\frac{\text{path delay}}{\text{output slew}}}$	0.22
$\mathrm{SF}_{\frac{\text{path delay}}{\text{R wires}}}$	0.42
$\mathrm{SF}_{\frac{\text{path delay}}{\text{C wires}}}$	0.61
$\mathrm{SF}_{\frac{\text{path delay}}{\text{C pins}}}$	0.67
$\mathrm{SF}_{\frac{\text{path delay}}{\text{cell delay}}}$	0.75
$\mathrm{SF}_{\frac{\text{path delay}}{\text{C}}}$	1.28

Table 6.3. Summary of path delay scaling factors.

As demonstrated in this chapter, these scaling factors are values that depend on the operating point of a design around which the LUTs of the liberty file are linearized. They depend on how much of the path delay is due to the cell delays and how much is coming from the interconnects delay. These pieces of information are required to calculate the scaling factors in a given design.

The scaling factors, $\mathrm{SF}_{\frac{\text{path delay}}{\text{cell delay}}}$ and $\mathrm{SF}_{\frac{\text{path delay}}{\text{output slew}}}$, have been verified by examining the frequency shift achieved in completely new PnR workflows. This evaluation involves

modifying the cell delay and output slew LUTs within the liberty files during the digital design process. Other constraints remain constant.

Similarly, to confirm the accuracy of $SF_{\frac{\text{path delay}}{R}}$ and $SF_{\frac{\text{path delay}}{C}}$ in representing the behavior of the most critical paths, it is necessary to execute PnR workflows in which the total resistance and capacitance of the nets are reduced by a certain factor.

For resistance, the .ict file, used as input in the PnR flow, can be scaled. This will provide insights into the overall reaction of optimization tools to these modifications. Alternatively, a separate static timing analysis could be conducted on the already routed design, with adjustments to the .ict and .qrc files. By comparing the observed average delay changes with predicted values, the model's accuracy can be assessed.

Regarding pin capacitance, a fresh PnR workflow is required. In this process, the input pins of all cells in the .lib file are scaled uniformly. By examining the achieved frequency shift, the alignment of results with predictions can be evaluated.

Analyzing wire capacitance is more intricate, as it involves linearly scaling the capacitances of the BEOL stack. The complexity and nonlinearities of the BEOL stack present challenges, and proper experimental design is crucial to mitigate the impact of coupling capacitance and achieve accurate results.

Furthermore, this context underscores the connection between cell delays and the BEOL stack. The BEOL stack's influence extends to both capacitive load and slew degradation. Consequently, the significance of statistically significant C_{tot} and R_{tot}, as suggested in section 5.2.1, becomes evident. These parameters are essential for effective eRO mapping, ensuring proper average capacitive load and slew degradation outcomes. This reflection also emphasizes the merits of the traditional structural RO mapping of Resistance and Capacitance. Despite its limited ability to precisely emulate R and C influences on BEOL delay, this approach guarantees an accurate representation of the total capacitive load and, consequently, cell delay. This outcome is achieved by leveraging metal lengths in the mapping process, which effectively captures the overall capacitance of the networks. It's worth noting that a straightforward average approach to R and C load can yield accurate cell delays in ring oscillator simulations. However, this method may fall short of effectively tracking the BEOL delay contributions of the finalized routed design.

Chapter 7

Conclusion and future perspectives

This thesis introduces and analyzes the improvements made to the previous extraction flow. It focuses on extracting and connecting the topology of paths to other circuitry features and physical parameters. The computation of metal and via resistance and capacitance contributions to path delay is achieved by integrating the Elmore delay computation into the topology extraction flow. The accuracy of Elmore predictions is compared with static timing analysis results in various ways to validate the proposed computation and align with the Place and Route (PnR) database.

The thesis discusses the potential applications of the R and C extraction tool, particularly in optimizing the Back-End of Line (BEOL) stack. Statistical data analysis studies are conducted to gain an in-depth understanding of the PnR output and its working mechanism. This analysis aims to develop new descriptive methods for characterizing a PnR design, understanding its features, limitations, and optimization potential.

Crucial criteria used by the EDA tool to achieve routing optimizations within timing and area constraints are deduced from the statistical analysis. Additionally, a novel mapping method is proposed to integrate electrical and statistical analysis into an enhanced ring oscillator which aims to be more accurate for benchmarking than the usual ring oscillator.
This eRO model turns out to be essential to accurately predict the impact of new BEOL booster technologies or BEOL stack optimizations.

A comprehensive assessment of the impact of the BEOL on cell delays provides insights into the contributions of resistance and capacitance components within the circuit. This evaluation facilitates the determination of overall signal path delays, leading to the calculation of the circuit's achieved frequency of operation.

Despite the topic's high specificity and the short working period, the project successfully achieved its predefined objectives. The support of the tutor and the collaboration with industry partners, such as Intel and Arm, provided valuable insight and expertise throughout the research process.

This short-term work lays the foundation for the long-term mission of developing a predictive model for a pathfinding methodology. This model aims to efficiently benchmark

future technology options, including new booster technologies, through a comprehensive analysis of extracted data.

However, the research path for developing a complete predictive model is not yet clear. The complexity and breadth of the topic may require advanced data analysis tools to better understand noisy data that are difficult to analyze using standard methods. The potential application of machine learning models is proposed, along with the discussion of design-dependent enhanced ring oscillator models. The need for predictive abilities to project current design statistics onto more complex designs, where databases are not yet available, is also emphasized.

7.1 Wiring congestion evaluation

Routing congestion significantly impacts achieving the best frequency-area optimization, which the PnR tool aims to accomplish with the given BEOL stack. Originally, only a few metal layers were employed for routing purposes. However, with downscaling, the demand for more routing resources becomes crucial to avoid violating DRC (Design Rule Check) rules and achieve designs with no violations.

To qualitatively evaluate routing congestion, designers often examine DRC rule violations. Overlapping wires carrying different signals may indicate that the router failed to place and route while satisfying all optimization constraints. A well-designed circuit should have no DRC violations, and all paths must meet timing constraints, meaning all critical paths should have $slack \geq 0$.

While the congestion map can provide a rough understanding of the congested areas, this approach heavily relies on the placement of macros and blocks. Some blocks may have a high density of pins along the edges to be routed, which can create an impression of congestion even if there are no real routing congestion issues. A more reliable figure of merit is required to quantitatively address routing congestion as a limiting factor in the design. Optimally, the fastest routing is obtained for a signal traveling along the least resistive path which exists between two pins. Thus it is supposed to reach the highest metal layer used in the net and go down to reach the cell. Thus, an idea can be to count how many times the main net part of a stage with FO equal to 1 is going up and down more than once. Checking when more than 2 vias of the same layer are used to route a net, should highlight routing congestion cases. Thus counting how many times it happens, can lead to a quantitative estimation of the routing congestion among the extracted critical paths.

7.2 Machine learning for topology analysis

The high-performance database, discussed in section 4, contains comprehensive extracted information for every net, creating a multidimensional database. However, analyzing this complex data poses significant challenges, as it requires filtering and manipulation to identify trends and patterns. To address these challenges, machine learning algorithms prove to be powerful tools for complex data analysis.

Machine learning algorithms enable researchers to uncover hidden patterns, make accurate predictions, and gain deeper insights into multidimensional data. They are particularly effective in identifying patterns, correlations, and trends that may not be apparent using traditional analysis techniques. While the extraction flow is written in Tcl, the data management code is implemented in Python, allowing for seamless integration of machine learning libraries into the workflow.

For example, machine learning algorithms can be applied to study net topology or net metal usage. After cells are routed and timing constraints are met, net metal usage and topology are influenced by cell placement and routing congestion, which refers to the availability of unused metal tracks for signal routing. Since these data can be noisy, extracting meaningful information from them using conventional methods can be challenging. Machine learning approaches can effectively handle such noisy data and uncover previously unknown dependencies.

Overall, integrating machine learning algorithms into the analysis of the high-performance database provides a powerful means to extract valuable insights and tackle the complexities of multidimensional data.

7.3 Design dependent eRO models

The eRO model, presented in chapter 5, is constructed by mapping the statistics of a routed design. The final routing optimization of a chip is highly dependent on the specific system being studied. For systems with multiple blocks, the macro placement plays a crucial role in determining the routing optimization, as it is influenced by factors such as macro size, shape, and placement.

In designs that involve memory paths, these paths often represent critical paths and are characterized by longer metal paths, making timing criticality more pronounced. When constructing a routing optimization model that aims to closely mimic a real design, it is essential to consider the dependencies of macro cells. Longer nets are likely to better represent the realistic scenario. Alternatively, multiple ring oscillators can be built to cover a range of net lengths. However, this introduces the challenge of averaging the results from a set of ring oscillators.

To enhance the reliability of the ring oscillator model and effectively address appropriate design optimization techniques for a specific design, models need to account for the design's dependence on various parameters. This is crucial to accurately predict the quantitative impact of new technologies. Additionally, different application fields, such as low-power or high-performance, can significantly influence the RTL (Register Transfer Level) structure and, consequently, the statistics of the final routing optimization. Therefore, it is necessary to select appropriate figures of merit to capture the design-dependent correlations with physical and electrical parameters that characterize a chip.

Furthermore, technology choice can greatly impact performance. Low-power cell designs may result in more compact and timing-limited designs. On the other hand, high-performance cell designs can be larger and more difficult to route, and they may face reliability issues related to joule heating mechanisms and power limitations.

In summary, expanding the application of the eRO model requires considering design-dependent correlations with physical and electrical parameters, accounting for various application fields, and addressing the influence of technology choices on performance, size, and reliability.

7.4 Evaluation of CPM's predictive ability and statistical analysis of new PnR runs

To improve and evaluate the critical path modeling (CPM) methodology, extensive analyses of multiple databases are necessary.

Firstly, the model can be studied for its dependence on the target period and area utilization in a given design. By pushing these values towards Pareto points, optimization algorithms will employ different techniques, including those based on static timing analysis. The characteristics of critical paths will reveal where the optimization algorithm struggled to find a suitable solution for timing-driven routing. In relaxed designs, the paths are expected to reflect the results of standard optimization techniques employed by the tool. For moderately pushed designs, critical paths should show the best outcomes achieved by the tool in challenging scenarios. In heavily pushed designs, the critical paths will indicate where, why, and how the tool failed.

Similarly, analyzing specific paths can provide valuable insights. Negative slack paths represent bottlenecks where the tool failed to meet timing constraints. Paths with positive slack close to zero demonstrate an optimal balance between the system, the design, and the optimization. Very positive paths, with low logic depth and small nets, are rarely critical at a logic level.

Another analysis can focus on how RC predictions vary by rerunning the full routing flow with a new BEOL technology included in the .ict file. The expectation is that the final routing optimization will achieve even better performance than predicted by computing the R and C contributions to the Elmore delay while keeping the same topology. This is because the optimal topology configuration is likely to change when the resistance or capacitance of a BEOL element is modified. However, further investigation is needed, particularly if placing optimization occurs prior to timing optimization. In such cases, standard cells are placed regardless of the characteristics of the metal layers, potentially resulting in less dramatic changes to the topology. This indicates a direct link between standard cell libraries and metal layer usage distributed between horizontal and vertical layers. Routing optimization algorithms typically rely on cost functions, where nets with more connected cells are given higher priority. Additionally, these nets are more critical for the optimization of paths that include them.

When searching for trends, a scaling factor can be included to estimate the quantitative impact of rerouting on predictions. This accounts for modifications in the BEOL stack characteristics resulting from the use of booster technology or other factors.

By conducting these various analyzes, the quality and reliability of the model developed can be improved and a deeper understanding of the routing optimization process and its effects on timing can be gained.

7.5 Power estimations

In addition to other parameters, the extraction code also captures power-related metrics such as leaking power, internal power, switching power, dynamic power, and total power for each instance. Power estimations and power-related analyses have gained increasing importance in recent times to better assess chip performance.

Currently, ring oscillator simulations are commonly used to benchmark the power characteristics of different technologies. However, it is important to note that while mapping a ring oscillator for frequency evaluations can be effective, it may not provide an accurate evaluation of power consumption. The power consumption of a design is influenced by the characteristics of all paths within the design, and it cannot be directly assessed through a ring oscillator alone.

Looking at the relationship between average delay and average power consumption of paths. It shows that higher delays correspond to higher power consumption by the paths. However, there appears to be a limit beyond which the power does not increase significantly. This limit may be influenced by the constraints specified in the design definition. Additionally, the switching power tends to stabilize among critical paths.

To overcome the limitations of using a ring oscillator for power evaluations, alternative approaches are needed. One possible approach is to employ an equivalent transmission line model, which can provide a more accurate power analysis compared to a ring oscillator. Alternatively, power predictions can be mapped into equivalent ring oscillator simulations that align with the power consumption of a specific design.

By exploring these alternative methods, more accurate and reliable power evaluations can be performed, leading to a better understanding of power consumption characteristics in chip designs.

Imec environment

Imec, the Interuniversity Microelectronics Centre, is an international research & development organization, world leader in nanoelectronics research. It has 12,000 square meters of cleanroom capacity and hosts more than 5000 researchers from more than 96 countries. It is a highly stimulating and inclusive environment.

During the internship, I had the opportunity to work with the Physical Design Research (PDR) group, which perfectly matched my fields of interest. I have also participated in intergroup meetings, being able to see the other research activities around my project. This environment can provide a lot of very interesting challenges where creativity, expertise, and passion are required and deeply involved. Its relaxed atmosphere allows to stimulate creativity and enables the possibility to collaborate and make contact with senior researchers. Moreover, I had the opportunity to actively collaborate with Imec partners, such as Intel and Arm.

I have really enjoyed working there, improving my soft and technical skills, which will definitely help and guide me through my future career.

I would like to thank Anita Farokhnejad for her loving support and tenacious guide. She fed my curiosity and provided satisfactory answers to all my questions.
Moreover, I am glad to have had the possibility to work in the PDR group, especially with Odysseas Zografos and Giuliano Sisto for their technical support, Peter Weckx and Julien Ryckaert for their guide and inspiring mindset.

Bibliography

[1] https://www.cadence.com/ko_kr/home/tools/digital-design-and-signoff/soc-implementation-and-floorplanning/innovus-implementation-system.html.

[2] Mutlu Avci and Serhan Yamacli. An improved elmore delay model for vlsi interconnects. *Mathematical and Computer Modelling*, 51:908–914, 04 2010.

[3] Ivan Ciofi, Philippe J. Roussel, Rogier Baert, Antonino Contino, Anshul Gupta, Kristof Croes, Christopher J. Wilson, Dan Mocuta, and Zsolt Tokei. Rc benefits of advanced metallization options. *IEEE Transactions on Electron Devices*, 66(5):2339–2345, 2019.

[4] W. C. Elmore. The Transient Response of Damped Linear Networks with Particular Regard to Wideband Amplifiers. *Journal of Applied Physics*, 19(1):55–63, January 1948.

[5] Anita Farokhnejad, Simone Esposto, Ivan Ciofi, Odysseas Zografos, Pieter Weckx, Julien Ryckaert, Pieter Schuddinck, Yang Xiang, and Zsolt Tokei. Evaluation of beol scaling boosters for sub-2nm using enhanced-ro analysis. pages 136–138, 2022.

[6] Andrew B. Kahng, Jens Lienig, Igor L. Markov, and Jin Hu. *VLSI Physical Design: From Graph Partitioning to Timing Closure*. Springer Publishing Company, Incorporated, 1st edition, 2011.

[7] Sun Lingling, Yan Xiaolang, Wang Junhu, and Cai Miaohua. The comparison of delay modeling for basic interconnect net topologies. In *1998 5th International Conference on Solid-State and Integrated Circuit Technology. Proceedings (Cat. No.98EX105)*, pages 468–471, Oct 1998.

[8] Mrinal Mandal and Bishnu Charan Sarkar. Ring oscillators: Characteristics and applications. *Indian Journal of Pure and Applied Physics*, 48:136–145, 02 2010.

[9] Arkadiy Morgenshtein, Eby G. Friedman, Ran Ginosar, and Avinoam Kolodny. Unified logical effort a method for delay evaluation and minimization in logic paths with *rc* interconnect. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 18(5):689–696, 2010.

[10] Sudhakar Muddu and Andrew B. Kahng. Optimal equivalent circuits for interconnect delay calculations using moments. Washington, DC, USA, 1994. IEEE Computer Society Press.

[11] Jan M. Rabaey. *Digital Integrated Circuits: A Design Perspective*. Prentice-Hall, Inc., USA, 1996.

[12] Jing Shang, Jianxia Hao, Qi Deng, Tao Hang, and Ming Li. Interface adhesion study of cu interconnection and low-k organic materials. In *2016 17th International Conference on Electronic Packaging Technology (ICEPT)*, pages 686–689, 2016.

[13] Ivan Sutherland, Bob Sproull, and David Harris. *Logical Effort: Designing Fast CMOS Circuits*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[14] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[15] Neil H. E. Weste and Kamran Eshraghian. *Principles of CMOS VLSI Design: A Systems Perspective*. Addison-Wesley Longman Publishing Co., Inc., USA, 1985.