Circuits and Systems
Mekelweg 4,
2628 CD Delft
The Netherlands
http://ens.ewi.tudelft.nl/

CAS-MS-2009-09

M.Sc. Thesis

45nm Extraction and verification flow with SPACE

Lin Li.

Abstract

As CMOS technology progresses to the 45nm generation and below, lots of changes are developed on material, process and structure, such as, metal gates, high-κ gate dielectrics, increased mobility, low-κ wiring dielectrics, and multiple layers of circuitry (3D) [1]. These new changes provide performance improvements and economic benefit, however face challenges on gate leakage, short channel effect, power dissipation and the various (parasitic) phenomena influences the accuracy and efficiency of analog and digital design. Consequently, extraction and verification of the design under deep submicron level is a big challenge.

The main objective of this project is to evaluate the capabilities of the SPACE layout to circuit extractor for extraction of the 45nm technology. Firstly, technology files for the 45nm virtual technology are developed based on the FreePDK 45nm hypothetical technology from Nangate 45nm Open Cell Library. After building the technology files successfully, several suitable examples are developed for testing the technology files and demonstrating the most important SPACE features. Moreover, SPACE and Calibre verification flows are also compared in this project. Finally, the trade-off between SPACE 2D and 3D interconnect capacitance extraction is discussed.

45nm Extraction and verification flow with SPACE

Thesis

submitted in partial fulfillment of the
requirements for the degree of

Master of Science

in

Microelectronics

by

Lin Li.
born in Datong, China

This work was performed in:

Circuits and Systems Group
Department of Microelectronics & Computer Engineering
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology

Delft University of Technology

Delft University of Technology
Department of
Microelectronics & Computer Engineering

The undersigned hereby certify that they have read and recommend to the Faculty of Electrical Engineering, Mathematics and Computer Science for acceptance a thesis entitled **"45nm Extraction and verification flow with SPACE"** by **Lin Li.** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: June 30th 2009

Chairman:

prof.dr.ir. Edoardo Charbon

Advisor:

dr. ir. Nick van der Meijs

Committee Members:

dr. ir. Arjan van Genderen

ir. Kees-Jan van der Kolk

Abstract

As CMOS technology progresses to the 45nm generation and below, lots of changes are developed on material, process and structure, such as, metal gates, high-κ gate dielectrics, increased mobility, low-κ wiring dielectrics, and multiple layers of circuitry (3D) [1]. These new changes provide performance improvements and economic benefit, however face challenges on gate leakage, short channel effect, power dissipation and the various (parasitic) phenomena influences the accuracy and efficiency of analog and digital design. Consequently, extraction and verification of the design under deep submicron level is a big challenge.

The main objective of this project is to evaluate the capabilities of the SPACE layout to circuit extractor for extraction of the 45nm technology. Firstly, technology files for the 45nm virtual technology are developed based on the FreePDK 45nm hypothetical technology from Nangate 45nm Open Cell Library. After building the technology files successfully, several suitable examples are developed for testing the technology files and demonstrating the most important SPACE features. Moreover, SPACE and Calibre verification flows are also compared in this project. Finally, the trade-off between SPACE 2D and 3D interconnect capacitance extraction is discussed.

Acknowledgments

With the final version of this thesis, 358 days of hard work came to an end. I would like to acknowledge several people who were involved in this project.

First and foremost, I would like to express my gratitude to Nick van der Meijs. He is the most excellent supervisor and communicator I have ever known. He has strongly improve the quality of this project, and has guided me to the right direction. Meanwhile, he has been fully supporting me and teaching me to find the scientific methods to solve the problem on this project.

Moreover, he has been constantly encouraging me to pursue the (science) truth. Without his trust and encouragement, I could not finish this project quickly and efficiently.

Thanks for the time he spent every week to discuss the problem that met in my project. Thanks for his supporting and guiding to my project. Thanks for his constantly encouragement and trust during my studying period.

I would like to thank Edoardo Charbon for his time that spent in my thesis defence as the committee chairman.

I would like to acknowledge Simon de Graaf for his suggestions on my project. He is my consultant on SPACE, and provide project supporting during this work. I am very appreciate the wonderful days with Simon.

I would like to thank Arjan van Genderen for his suggestion on the choice of the suitable large demo example in my project.

I would like to thank Kees-Jan van der Kolk for his supporting in comparison of interconnect capacitance extraction. And I am grateful to his time and effort in reviewing on my thesis. Last, thanks for his magic Salsa showing.

I would like to thank Tao Xu for his supporting and suggestions on the synthesis and place&route flows.

I would like express my gratitude to Yu Bi, Qin Tan, Yiying Wang and Gert Jan Schoneveld. Thanks for their time and efforts that spent in my thesis revising.

I would like to acknowledge Antoon Frehe and Duan Zhao for their supporting on hardware and software management. Without their help, I can not finish this project efficiently.

I would like to thank Secretary Laura Bruns for organizing my final defence.

I would like to thank all colleagues in CAS labs. I really appreciate the wonderful days with all of you.

On a personal level, I would like express my gratitude to my parents for their supporting at anytime and anywhere. I really thank them to give me freedom and let me find the life by myself. I cannot forget my two roommates Kai Lin and Yong Zhao because they were always present when I need them.

I would like to thank all my friends in Delft, without them I could not have survived here for two years. Thank you all so much!

Finally, I would like to share my experience: Success Belongs to the Persevering!

Lin Li.

Delft, The Netherlands

June 30th 2009

Contents

List of Figures

List of Tables

1 Introduction

1.1 Motivation

As Complementary Metal Oxide Semiconductor (CMOS) technology progresses to the 45nm generation and below, lots of changes are developed on material, process and structure, such as, metal gates, high-κ gate dielectrics, increased mobility, low-κ wiring dielectrics, and multiple layers of circuitry (3D) [1]. These new changes provide performance improvement and economic benefit. However, they also face several challenges on gate leakage, short channel effect and power dissipation. The aggressive scaling of CMOS technology also leads to the big change of the MOSFET models. At 45nm technology and beyond, the reduction of supply voltage and scaling geometry lead to the physical effects that dominate the designs performances [3]. In order to predict the characteristics of nanoscale CMOS accurately, one new Predictive Technology Model (PTM) [4] is developed. This model includes more physical parameters into the prediction and can capture process sensitivities correctly [5].

Furthermore, new changes and new CMOS model of 45nm CMOS technology and below lead to the challenges for the designs and Electronic design automation (EDA) tools. Due to the on-going increase of VLSI integration density, the various (parasitic) phenomena is going to influence analog and digital designs. Consequently, extracting and verifying the design under deep submicron level accurately and efficiently is a big challenge.

SPACE is an advanced layout-to-circuit extractor for analog and digital integrated circuits [6]. It engages in the accurate and efficient layout-to-circuit extraction for deep submicron technology. It uses an efficient finite-element method to accurately extract interconnect resistances [6]. A fast and comprehensive interpolation method and boundary-element method (BEM) for capacitance extraction are used in SPACE to improve efficiency and accuracy. Meanwhile, SPACE can also extract substrate resistances to verify the substrate coupling effect. Besides efficiency and accuracy, SPACE has lots of features, for example, it is capable of extracting 45 degree polygonal layout, it is technology independent, it can read and write various layout and circuit formats, etc[6].

For SPACE, it is necessary to evaluate the capabilities to extract new 45nm technologies as a layout-to-circuit extractor. Meanwhile, the most important features of SPACE should be demonstrated based on the 45nm technology.

1.2 The related work

Some universities, organizations and companies take effort to develop an Open-Access-based process design kit (PDK) for the 45nm technology node and the Predictive Technology Model at and below 45nm node. This FreePDK is presented and available for users to utilize in research and education. One 45nm Open Cell Library is released by Nangate Inc., in order to provide for the purposes of testing and exploring EDA flows [7].

Nangate 45nm Open Cell Library is generated using Nangate's Library Creator [7]. It uses the 45nm FreePDK Base Kit from North Carolina State University (NCSU) [8]. And its characterization is done using the Predictive Technology Model (PTM) from Arizona State University (ASU) [4].

Nangate 45nm Open Cell Library contains the following files [7]:

- Geometric library in Library Exchange Format (LEF)
- Simulation libraries in Verilog and Spice (pre and post parasitic extracted netlists)
- Cell layouts in GDSII
- Liberty (.lib) formatted libraries with CCS Timing, ECSM Timing and NLDM/NLPM data (fast, slow and typical corners)
- Schematics
- Library databook in HTML/XML format
- OpenAccess database containing layouts and netlists

Meanwhile, some reference processes have been successfully developed in SPACE, such as *tsmc0.25*, *dimes03* and *scmos_n*, etc.

These libraries can be utilized as the references to develop FreePDK45 process with SPACE.

1.3 Project target

The main objective of this project is to evaluate the capabilities of the SPACE layout-to-circuit extractor for extraction of the 45nm technology and to create tutorial examples based on such a technology.

First step of this project is to develop the SPACE technology files for the 45nm virtual technology from the FreePDK45 hypothetical technology from NCSU and Nangate 45nm Open Cell Library. The technology files should enable a complete extraction flow, including Spice simulation.

Moreover, suitable examples are built for demonstrating the most important SPACE features after 45nm technology files development. These examples can also be used for testing the technology files. Three examples are considered in this project:

1. A demo design for validation of the technology files.
2. A ring oscillator is developed to show the substrate resistance extraction capabilities.
3. A large design is built to demonstrate the extraction speed and memory requirements of SPACE.

Furthermore, this project compares SPACE verification flow with Calibre verification flow. Finally, SPACE 2D and 3D capacitances extraction is discussed.

1.4 Synopsis

The organization of this thesis is:

Chapter 2 introduces the methodology of this project briefly.

Chapter 3 presents the SPACE technology files development procedure for the FreePDK 45nm process. The technology files of the 45nm process are the foundation of this project.

Chapter 4 presents several suitable examples which are built for verifying the validation of 45nm technology files. Meanwhile, these examples will be used to demonstrate the important SPACE features.

Chapter 5 compares the Calibre and SPACE verification flow, and discusses the capacitance extraction of SPACE 2D and 3D.

Chapter 6 summarizes the work achieved and suggests the future work.

2 Methodology of this project

This project contains three important factors:

1. SPACE—Platform for development
2. FreePDK 45nm technology files—Foundation of this project
3. Extraction & verification flow—Flow for validation of technology files as well as for SPACE features demonstration

This chapter is going to describe the methodology of this project based on the above factors.

2.1 Brief introduction of SPACE

The function structure of the SPACE is shown in Figure 2.1

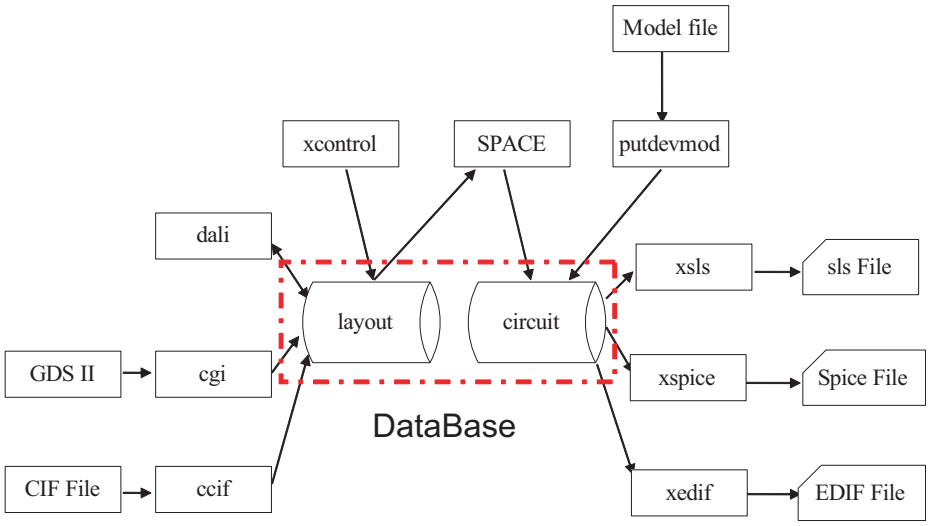

Figure 2.1: Basic structure of the SPACE

SPACE operates in a database that is called a project. In this project, it finds the layout data and stores the extracted circuit data [9]. In one project, the particular process is set and every layout is designed in this particular process; all cells in this project share the same process. It means that the design is related with the certain process. The characteristics of process are specified in the related technology files. Hence, the technology files of new process is the foundation of SPACE.

2.2 FreePDK45 technology files development

The foundation of SPACE is the technology files of new process. So that, in this project, the first and the most important step is technology files development. We use the Nangate 45nm Open Cell Library as the reference process. The Nangate 45nm Open Cell Library provides an open-source 45nm technology in order to help universities and organizations in developing flows, circuits and algorithms [7]. According to the reference process, the characteristics of 45nm process can be extracted. And then, these parameters can be used to build SPACE technology files. Finally, the source technology files of FreePDK45 should be compiled into the input element file of SPACE and put it into SPACE. Figure 2.2 indicates the whole flow of technology files development. Chapter 3 is going to introduce more details of technology files development.

Figure 2.2: Simple flow of technology files development

2.3 Extraction & verification flow

The simple flow of top-down design and verification flow with SPACE is shown in Figure 2.3. Firstly, the synthesizable VHDL models are developed with related test-benches. These models are validated through simulation. Then, RTL synthesis is going to be implemented. It infers a possible gate-level realization of the input RTL that meets user-defined constraints, such as area, timing, etc. Synthesis step generates several outputs: gate level VHDL or Verilog netlist, SDF includes the delay information for simulation. Place & Route step infers the layout of the gate-level netlist. The outputs of this step are: layout description in GDSII format; SDF description includes interconnect delay; Verilog gate-level netlist. At the end, SPACE is going to use the GDSII files to generate the circuit description. Meanwhile, parameters like, substrate resistance, interconnect capacitance, can also be extracted by SPACE.

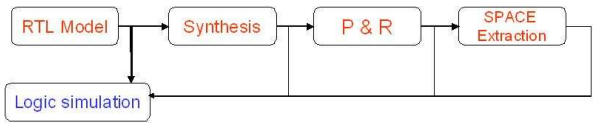

Figure 2.3: Simple flow of top-down design and verification with SPACE

The flow in Figure 2.3 is going to be utilized for validation of technology files and for demonstrating the features of SPACE. More details of this flow can be found at Chapter 4 .

According to the above methodology, the goal of this project is going to be achieved. Here, the reference process in this project is Nangate 45nm Open Cell Library.

45nm Process technology tiles development

3

As the process scaling downto 45nm and below, the increasing of chip dimensions make the interconnect parasitics tend to dominate the chip performance. Hence, it is important to fully take into account all parasitic elements as the layout simulation. SPACE is a candidate to achieve this goal.

From a mask-level layout, SPACE can generate a circuit netlist, which contains active devices, instances of cells, terminals and parasitic elements [6]. Moreover, accuracy and efficiency of extraction is one of the most important features of SPACE.

At the 45nm technology, the capabilities of SPACE for extraction of the deep submicron level technology should be evaluated. Meanwhile, the most important features of SPACE should be demonstrated.

This chapter focus on explaining the FreePDK45nm process technology files development based on the Nangate 45nm Open Cell Library. The demo process technology files, such as *tsmc0.25*, *dimes01* and *scmos_n*, will be used as the reference.

3.1 Flow of techfiles development

This section explains the development flow of SPACE FreePDK 45nm process technology files (Figure 3.1). The graphical technology interface tool—**spock** is used to build the technology files.

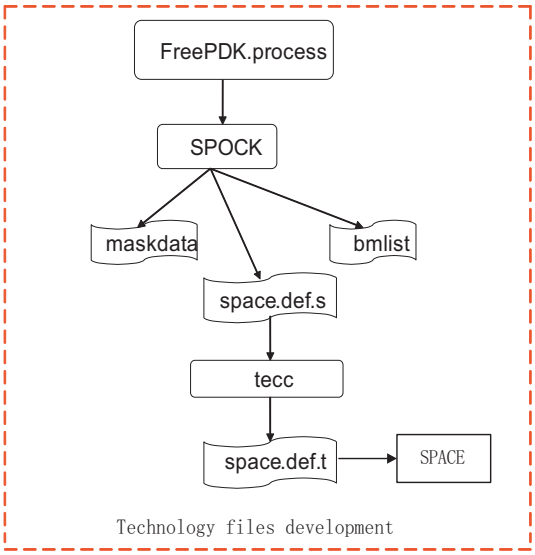

Figure 3.1: Techfiles Development Flow

Firstly, the characteristics of the new process are extracted, such as the layer specification, conductor specification, etc. Then these parameters are developed into the new technology files with **spock**. Technology files present the details of the new process and parameters according to the extraction [10], and it contains several files:

- **bmlist**: Maps GDS mask numbers to process mask names, generated by **spock**
- **maskdata**: Defines the masks used in the process and the colors of each mask in the layout editor **dali**, generated by **spock**
- **space.def.s**: Element definition source file with detailed process information, generated by **spock**
- **space.def.p**: Defines parameters used during layout to circuit extraction, generated by **spock**
- **epslay.def**: Defines parameters used during layout-to-circuit extraction, generated by **spock**
- **space.def.t**: Compiled element definition file, in binary format, and used as the input file for SPACE
- **xspicerc** : Control file that specifies models used for devices and other components, and the location of the library files that contain description of these models, generated manually
- **Free45nm.lib**: PTM models description for 45nm, generated manually

The element definition source file—*space.def.s* defines the circuit elements that can be recognized from the layout description. And it describes the detailed process information. It also contains the values for the different capacitances and for the different sheet-resistances [6].

After generating the element definition source file *space.def.s*, it can be compiled into the tabular element definition file—*space.def.t*. This tabular element definition specifying how the different elements can be identified from the different mask combinations, and which values should be used for [6]. Finally it can be imported into the SPACE system.

Table 3.1: Function of elements definitions

Element definition	Functions
Layer Specification	Identify the mask elements
Vertical dimension	Specifies the different conductors in 3D extraction
Dielectrics	Specifies the dielectric structure of the chip in 3D extraction
Conductor list	Define the conducting layers
Contact list	Define the contact elements between different conductors
FETS list	Identify the MOSFET

Table 3.1 describes the function of element specifications. These specifications include the layer elements, vertical dimensions, the conductor elements, the contact elements and the transistor elements, etc.

Following sections are going to discuss the development steps of element definitions.

3.2 Layer specification

Nangate 45nm Open Cell Library provides the Cadence technology file with extension ".tf". The layout and process information of the Cadence technology file can be extracted manually. These information will be used to develop SPACE technology files.

We can find the layers information from "layerDefinitions" section. And the information can be used in CACD process description. Normally, the layer numbers defined in this section is *NOT* the numbers used in the GDSII stream, however in this process, Nangate 45nm Open Cell Library defines the layer numbers as the same with the GDSII numbers, we can find the related information from Appendix B.1.

According to the layer definition and GDSII stream numbers, the layer specification and GDSII file—bmlist.gds, can be developed with **spock**. Table 3.2 presents the layers in the FreePDK 45nm technology process:

Table 3.2: 45nm Technology layers list

Layer	nWell	pWell	nImplant	pImplant	diffusion	poly	contact	metal1...10	via1...9
Material	nwell	pwell	nimplant	pimplant	diffusion	poly	contact	metal1...10	via1...9

45nm process contains the 26 layers definition totally, such as 10 metal layers, 9 contact vias, nwell & pwell, diffusion region, poly and n-type & p-type implant region. More detail information of layers definitions can be found in technology files Appendix D (line 4—39).

3.3 Vertical dimension

Vertical dimension is used for 3D extractions [10]. As the technology goes down to the sub-micron level, 3D numerical techniques are used to compute the interconnect capacitances.

For SPACE 3D extraction, the vertical dimensions of the conductors and the dielectric structure of the circuit are added into the element definition file. The vertical dimension list specifies the distance between the substrate and the bottom of each conductor, and the thickness of each conductor. The dielectric structure specifies the dielectric structure of the chip [11].

Here, based on the Calibre techfile "CalibrexRC.rul" (Appendix B.2) from Nangate 45nm Open Cell Library, the vertical dimension and the dielectric structure are developed.

Figure 3.2 indicates the vertical dimension of FreePDK 45nm technology process. Vertical dimensions list in SPACE can be found in the FreePDK 45nm technology files Appendix D (line 139–153).

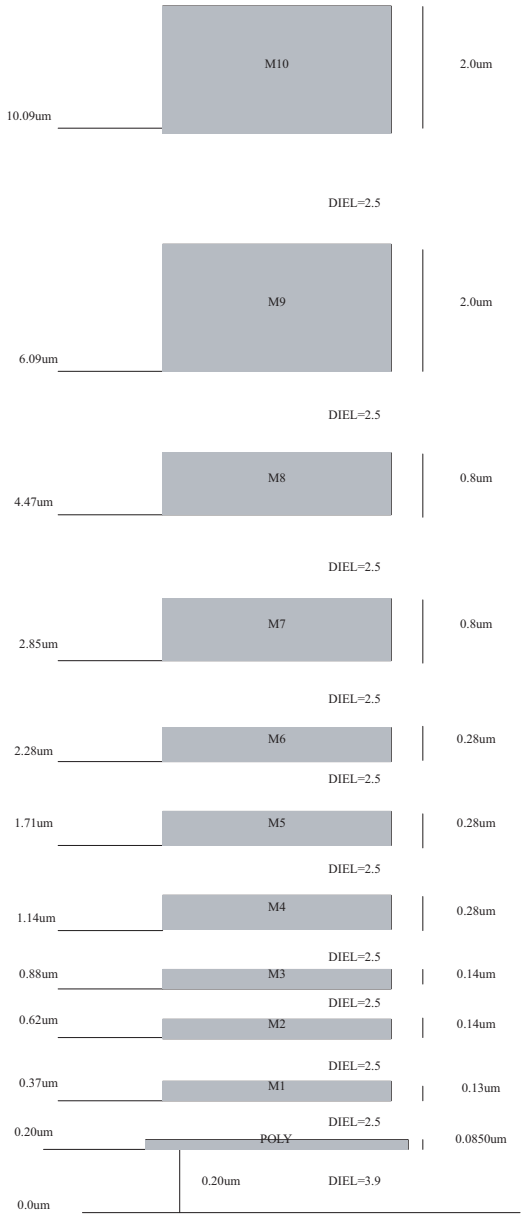

Figure 3.2: Vertical dimension of 45nm process

Table 3.3 lists the dielectric structure of the 45nm process. Note that, the lowest dielectric is specified firstly in the list, then the second lowest, etc. The first dielectric layer bottom **MUST** be zero, this dielectric layer is field-based-dielectric; the top of the last dielectric—air is at infinity; and the middle dielectric layer is the standard dielectric.

Table 3.3: Dielectric structure spreadsheet

Material	Relative permittivity	Bottom at(e-6 m)	Comment
field_base	3.9000	0.0	between poly&sub
stadiel	2.500	0.2000	between different metals
air	1	14.090	air

3.4 Field-effect transistor list

The Field-effect transistor list identifies different mask combinations in the layout that define FETS for extraction [10]. In this specification, the gate mask and drain/source mask are defined firstly in element definition. Additionally, the drain/source regions are specified behind the drain/source mask of the transistor definition. Moreover, the bulk connection can also be specified for the transistor. Based on the definition of drain/source area, the area/perimeter parameters *ad*, *as*, *pd*, *ps*, *nrs*, and *nrd* for the transistor can be extracted.

More information about FETS definition of FreePDK 45nm technology process can be found in Appendix D (line 110-113).

3.5 Conductor list

The conductor list defines the conducting layers in the circuit for resistance extraction. For each conductor specification, the actual conductor mask and the sheet-resistivity (in ohms) are required. The type of each conductor can be specified as *n* doped conductors, *p* doped conductors and *metal* [6]. As extracting resistances, linear resistances will be extracted for a conductor as default.

The key factor for this element definition is sheet-resistivity of each layers. The values of sheet-resistivity influences the extraction accuracy of SPACE. Nangate 45nm Open Cell Library and FreePDK website [8] have provided the sheet-resistivity of each conductor layers (Table 3.4):

Table 3.4: Conductor list spread sheet

Name	Sheet Resistivity(ohms)	Name	Sheet Resistivity(ohms)
cond_nw	933.0	cond_m4	0.21
cond_na	0	cond_m5	0.21
cond_pa	0	cond_m6	0.21
cond_p	7.8	cond_m7	0.075
cond_m1	0.38	cond_m8	0.075
cond_m2	0.25	cond_m9	0.03
cond_m3	0.25	cond_m10	0.03

Note that, the sheet-resistivity for "cond_nw" can be found from technology file *FreePDK.tf*, for *cond_na* and *cond_pa*, sheet-resistivity values are **zero** as default. And the sheet-resistivity values of poly conductor layer and metal conductor layers can be found in FreePDK website [12].

Table 3.5: Contacts list spread sheet

Name	Mask1 Mask2	Res. per area($\Omega.\mu m^2$)	Name	Mask1 Mask2	Resistivity($\Omega.\mu m^2$)
cont_su	@sub metal1	0	cont_3	metal3 metal4	0.0588
cont_nw	nwell metal1	0	cont_4	metal4 metal5	0.0588
cont_an	active metal1	0	cont_5	metal5 metal6	0.0588
cont_ap	active metal1	0	cont_6	metal6 metal7	0.0588
cont_ps	poly metal1	0.0338	cont_7	metal7 metal8	0.16
cont_1	metal1 metal2	0.02535	cont_8	metal8 metal9	0.16
cont_2	metal2 metal3	0.0245	cont_9	metal9 metal10	0.32

3.6 Contact list

The contact elements connect different conductors that are on top of each other [10] (Figure 3.3).

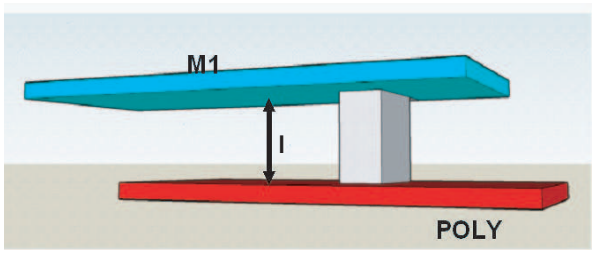

Figure 3.3: Contact structure

Two different conductor masks are connected by the contact. The resistivity parameter (Resistivity per area) is required in this specification, the contact resistance in *ohm square meter* for a contact. Note that the resistivity parameter (Resistivity per area) for each contacts are computed based on Equation 3.1. And the value of resistivity parameter (Resistivity per area) is the result of Resistivity × Area.

$$\rho \times l = R \times A \tag{3.1}$$

where
ρ is the static resistivity (measured in ohm meters, Ωm);
R is the resistance of vias (measured in ohms, Ω);
A is the area of contact vias (measured in meter square, m^2);
l is the distance between two layers (measured in meters, m);

According to the FreePDK 45nm process layers sizes, which are published in FreePDK website [8], the contact resistances are calculated based on the Equation 3.1. Table 3.5 shows the information of contacts list.

Note that, here we assume the *substrate contact*, *nwell contact* and *active region contact* resistivities are **zero**. This specification can be used as the resistance extraction.

3.7 Conclusion of technology files development

According to the technology files development flow (Figure 3.1), the FreePDK45 technology files had been developed for SPACE. The technology files contained the information about layer, contacts, conductors, MOSFETS, etc. The validation of the technology files is going to be verified at the next chapter. After that, the most important features of SPACE should be demonstrated.

Technology files verification and large demo design

4

After development of FreePDK45 technology files, the characteristics of 45nm process are clearly described in this technology files. And then SPACE is going to read the information that provided by the technology files. In the technology files, we can find the layer definition, vertical dimensions, contact lists, etc. However, the technology files should be validated before publication.

This chapter is not only going to focus on the 45nm technology files verification, but also the large circuit demo design to show the extraction speed and memory of SPACE.

According to the Figure 2.1, SPACE can extract several netlists based on the technology files, such as the Spice netlist, the VHDL netlist, etc. These netlists contain the layout information like drain/source area and perimeter, MOSFET sizes and interconnect capacitances, etc. These information can be utilized to verify the FreePDK45 technology files according to the reference netlists that are provided by Nangate 45nm Open Cell Library. Meanwhile, the function validation of the netlists can also be discussed based on the simulation results.

Two ways will be considered in this chapter to verify the FreePDK 45nm technology files:

1. Netlist comparison: According to the Spice netlist extracted by SPACE, the netlist connection can be verified based on the reference netlist from Nangate 45nm Open Cell Library. Meanwhile, netlist parameters, like drain/source area, drain/source perimeters can be compared with Calibre extracted results.

2. Function verification: Attempts to determine if the design operates as specified.

The organizations of this chapter are: Firstly, the standard cells that provided by Nangate are used as the demo in this chapter. We randomly choose several cells as the demo designs, in order to show the netlists and function validation. Moreover, 9-stages ring oscillator is developed to show the SPACE capabilites of substrate resistance extraction and substrate noise capturing. Finally, one large demo design is developed in this chapter to show the extraction speed and memory requirements of SPACE.

Here, one 8-bit 8051 with 128-bytes RAM microprocessor is used as the demo design. This microprocessor is an open IP core, from the Dalton project (University of California) [13]. It provides synthesizeable VHDL code and models the actual Intel implementation rather closely. It is capable of addressing 64K of program memory and 64K of data memory.

4.1 Standard cells Verification

Nangate Open Cell Library provides over 100 different standard cells, functions ranging from buffers to scan flip-flops [7]. These standard cells can be used as the demo designs to show the validation of the technology files.

4.1.1 Standard cells netlist verification

Nangate 45nm Open Cell Library provides the 128 standard cells, like *NAND, INV, FA,* etc. In the library, we can find the GDSII files, pre-layout and post-layout netlists for all standard cells. These data can be utilized for technology files verification.

In this part, we focus on the layout connection verification. The extracted layout netlist by SPACE is compared to the netlist taken from the Nangate Open Cell Library circuit schematic.

Figure 4.1 shows the flow of this procedure. Firstly, tool **cgi** (convert GDSII files to an ICD project) is used to convert the GDSII file of basic cell from Nangate 45nm Open Cell Library into the project. Then, **space** is used to extract the layout to the circuit. Meanwhile, reference netlist can be convert into the database (DB) by **cspice** (convert netlist into the database). Finally, **match** (a net work comparison program) is utilized to compare the netlist connection between extracted netlist and reference netlist.

Pre-simulation netlists from Nangate 45nm Open Cell Library are used as the reference netlist. SPACE layout extracted netlists are compared with the reference netlists. The match reports indicate the netlist connection of SPACE extracted netlist.

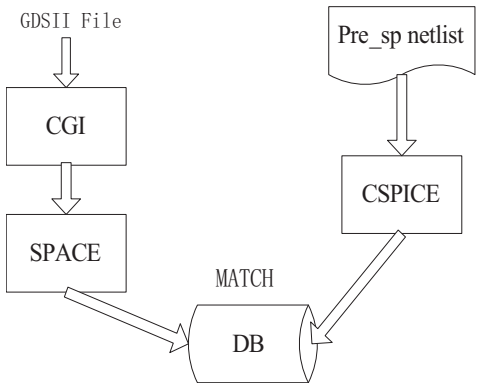

Figure 4.1: SPACE layout vs netlist comparison flow

Due to the amounts of the standard cells, one program *domatch* is needed to match every standard cell with reference circuit netlist automatically. Figure 4.2 indicates the flowchart of the *domatch* programme. The *domatch* program can be found in Appendix A.1.

After layout vs netlist flow, we can randomly open one match report to check weather the layout and netlist matched successfully. For this project, connection of standard cells successfully match with reference circuits. Here, the NOR4-X2 cell is used as an example to show the match result (Figure 4.3).

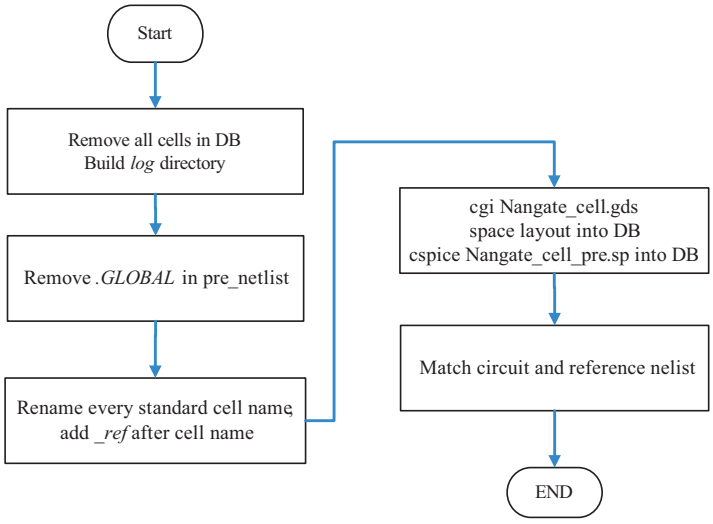

Figure 4.2: MATCH flowchart

```
Unconnected (Nom) net: 4
Unconnected (Nom) net: SUBSTR
Unconnected (Nom) net: GND

match: Succeeded.

+--------------+
| MATCHING     |
+-------+-------+----------------------+-----------------------------------+
|   P   | Network NOR4_X2          (Nom)| Network NOR4_X2_ref          (Act)|
+-------+-------+----------------------+-----------------------------------+
|     1 | VDD        (terminal)         | VDD        (terminal)             |
|     2 | _T6        (penh)             | M_instance_87 (penh)              |
|     3 | _T8        (penh)             | M_instance_91 (penh)              |
|     4 | 3          (net)              | net_002    (net)                  |
|     5 | A2         (terminal)         | A2         (terminal)             |
|     6 | 2          (net)              | net_001    (net)                  |
|     7 | _T5        (nenh)             | M_instance_66 (nenh)              |
|     8 | ZN         (terminal)         | ZN         (terminal)             |
|     9 | VSS        (terminal)         | VSS        (terminal)             |
|    10 | _T1        (nenh)             | M_instance_53 (nenh)              |
|    11 | _T3        (nenh)             | M_instance_60 (nenh)              |
|    12 | _T7        (nenh)             | M_instance_72 (nenh)              |
|    13 | A3         (terminal)         | A3         (terminal)             |
|    14 | _T4        (penh)             | M_instance_83 (penh)              |
|    15 | 1          (net)              | net_000    (net)                  |
|    16 | A4         (terminal)         | A4         (terminal)             |
|    17 | A1         (terminal)         | A1         (terminal)             |
|    18 | _T2        (penh)             | M_instance_78 (penh)              |
+-------+-------+----------------------+-----------------------------------+
```

Figure 4.3: NOR4-X2 Match log result

Note that Nangate GDSII file does not define the contact to the bulk, so that in the result, these terminals are not connected.

4.1.2 Random standard cells verification

Above section, we verified the SPACE extracted netlist connection. Accuracy of extracted layout parameters are also important factors for this project. So that we discuss the layout characteristics in this part. Moreover, logic function validation of standard

cells should also be considered in this part. We randomly select two standard cell: *INV-X1* and *AOI22-X2*, so as to test layout parameters like drain/source area, perimeter, and verify the logic function of the cells. The most common approach to function validation involves the use of a logic simulator software. Here, the *Spectre* simulator is used to show the cells logic results based on the SPACE extraced netlists. Since SPACE can only extract the Spice netlist. Hence, the *SPP* command (Figure 4.4) is utilized to convert the Spice netlist into Spectre format.

The flow for the standard cells verification is shown in Figure 4.4. Firstly, the GDSII file of standard cell is imported into the SPACE database. Then **space** is used to extract the layout to the circuit. Finally, the standard cell netlist can be extracted for comparison or simulation.

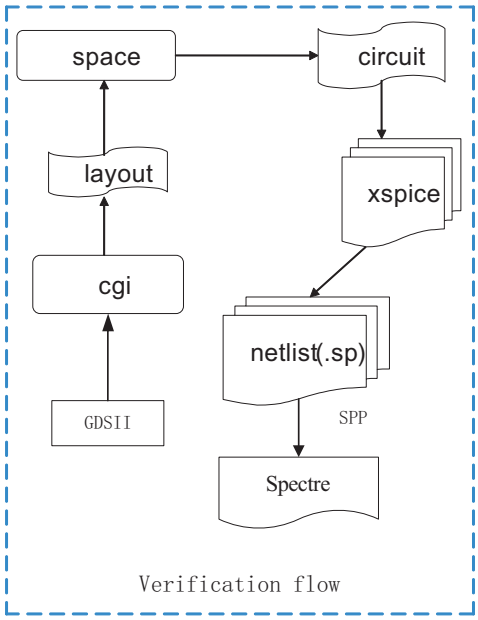

Figure 4.4: Technology files verification flow

4.1.2.1 INV-X1

Using one NMOS transistor and one PMOS transistor, CMOS invert is built directly. Figure 4.5 indicates the schematic of invert. It is a very simple basic cell, and we can recognize the layout parameters information conveniently. Moreover, it is widely used in digital design. So that, we use the invert as the first demo design in our project.

Comparison between Calibre and SPACE layout netlist extraction is discussed firstly. Table 4.1 indicates the parameters that extracted by SPACE and Calibre. It can be clearly seen that Calibre extracted drain/source areas are same compared with SPACE extraction results. However, the perimeters of drain/source between Calibre and SPACE extraction are different.

In Calibre, the drain/source perimeters are equal to the sum of four edges of drain/-source active region (Figure 4.7–left); nevertheless, SPACE use the junction capacitance

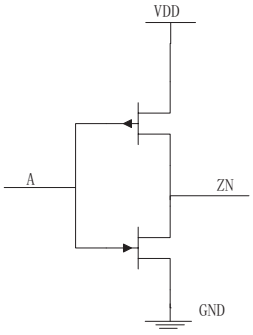

Figure 4.5: INV-X1 circuit

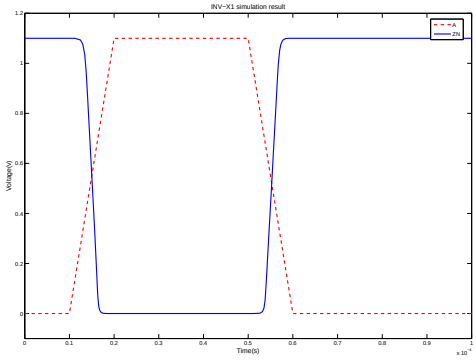

Figure 4.6: Simulation result of Invert

to extract the drain/source area and perimeter values, which are equivalent to the sum of three active region edges without the edge nearby the poly (Figure 4.7–right). The blue lines indicate the parameters calculation region for Calibre, and the pink lines show the SPACE extraction region.

Table 4.1: Comparison between Calibre and SPACE

	Calibre-NMOS	Calibre-PMOS	SPACE-NMOS	SPACE-PMOS
W	90n	135n	90n	135n
L	50n	50n	50n	50n
AS	9.45f	14.175f	9.45f	14.175f
AD	9.45f	14.175f	9.45f	14.175f
PS	390n	480n	300n	345n
PD	390n	480n	300n	345n

Here,
W is the width of the transistor;
L is the length of the transistor;
AS is the source area;
AD is the drain area;
PS and PD are source/drain perimeters;

Operation function of INV-X1 can be verified by the simulation results of SPACE extracted netlist (Figure 4.6). The input signal (red line) is the ideal wave which is defined in the simulation control file. The blue line shows the output signal of the invert.

4.1.2.2 AOI2-X22

After the simple inverter verification, we select one complex cell AOI2-X22 as the second demo design. AOI (AND-OR-Invert) logic and AOI gates are two-level compound (or complex) logic functions constructed from the combination of one or more AND gates followed by a NOR gate. AOI gates perform one or more AND operations followed by

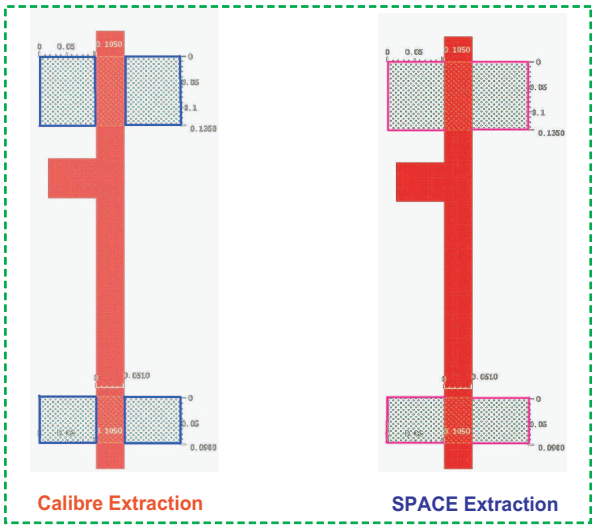

Figure 4.7: Comparison of Calibre and SPACE extraction

OR operation and then an inversion. For instance, a 2-2 AOI gate can be represented by the boolean equation Equation (4.1), circuit is shown in Figure 4.8 and truth table is shown in Figure 4.9:

$$ZN = \overline{(A1 \bullet A2) + (B1 \bullet B2)} \tag{4.1}$$

AOI (And or Invert) circuit is checked in this part based on the layout comparison and function validation. Figure 4.10 shows the simulation result.

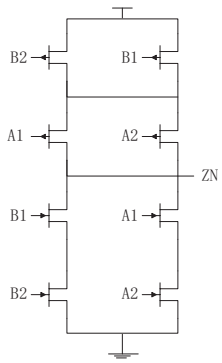

Figure 4.8: AOI circuit

B1	B2	A1	A2	ZN
0	x	x	0	1
x	0	x	0	1
0	x	0	x	1
x	0	0	x	1
1	1	x	x	0
x	x	1	1	0

Figure 4.9: Truth table of AOI

Layout comparison is also considered here. The netlist extracted by Calibre, indicates that Calibre reuse the middle region of active region as both drain and source; moreover, for perimeter calculation, Calibre add the four edges of the drain/source region (Figure 4.11).

However, netlist extracted by SPACE shows the different result. SPACE divides the middle part between two poly-gate, and one half is drain, another half is source. SPACE adds three active region edges without considering the edge nearby the poly to calculate the first or the last transistor's drain/source perimeters; but for the internal

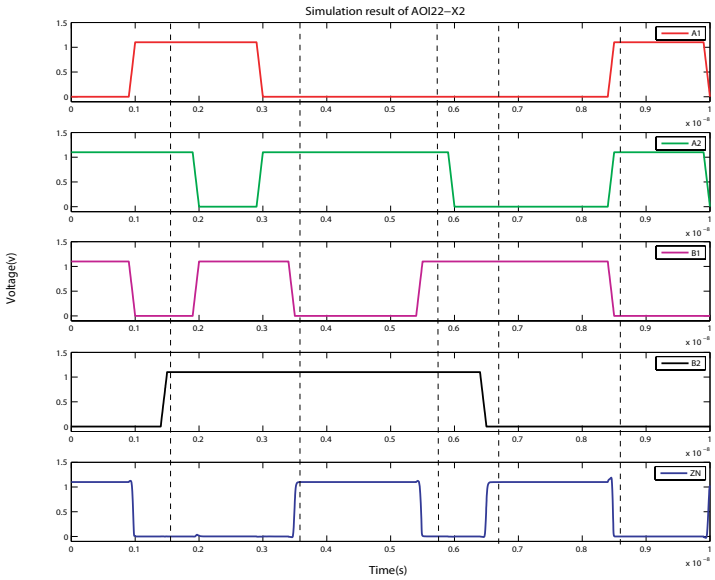

Figure 4.10: AOI Simulation result

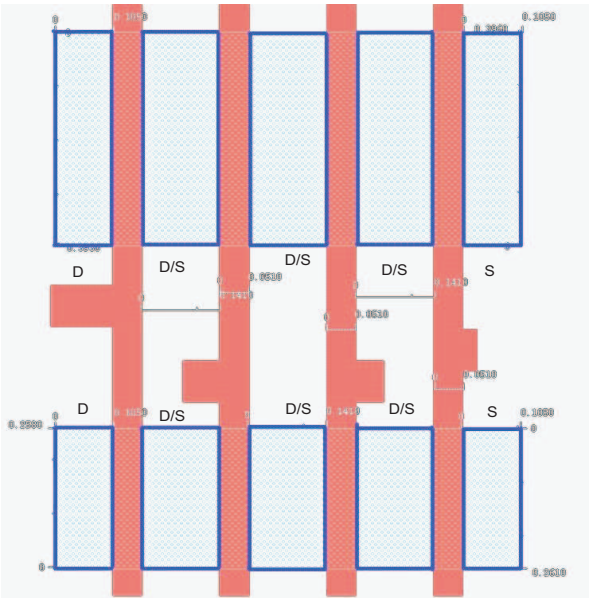

Figure 4.11: AOI layout extracted by Calibre

active region, SPACE only adds the top and the bottom edges of the active region to calculate the drain/source perimeters (Figure 4.12).

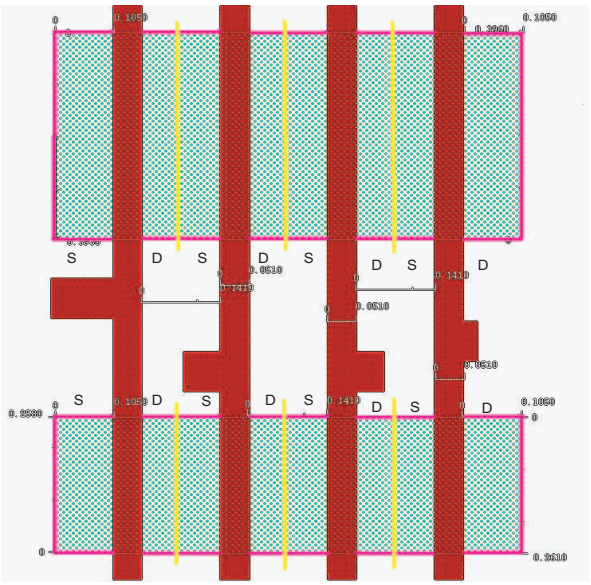

Figure 4.12: AOI layout extracted by SPACE

4.1.3 Conclusion of standard cells verification

According to the analysis of above parts, it can be concluded that: (1). The basic cells netlists connection, which are extracted by SPACE, match the reference netlists successfully. (2). The logic function of two random demo examples are verified based on the SPACE extracted netlists. The FreePDK 45nm technology files are validated, but the layout parameters extracted by SPACE and Calibre are different. These deviations indicate that extraction methodology between SPACE and Calibre are different.

For SPACE, in the perimeter of the drain and source, the edge that coincides with the transistor channel, is not included. If more than one transistor is connected to a drain/source region, the area value that is obtained for the total drain/source region is subdivided over the different transistors in proportion to the widths of the transistors. The perimeter value is split in a similar way [6].

For Calibre, in the perimeter calculation of the drain and source, the edges which coincide with the transistor channel are computed. When more than one transistor is connected to a drain/source region, the area of the drain/source is reused by each transistor. The same way is used in the perimeter calculation.

4.2 Ring oscillator

Ring oscillators are simple circuits with which you can easily compare different circuit technologies. A ring oscillator is often used to demonstrate a new hardware technology, analogous to the way a *hello world* program is often used to demonstrate a new software technology. Ring oscillators can be used to measure the effects of voltage and temperature on a chip. And it can measure the new frequency of the oscillator. This gives you a good base point for measuring and comparing the performance of the new technology. Ring oscillators can also be used to measure the effects of voltage and temperature on a chip.

The purpose of this example in this project, is to test the substrate resistance extraction abilities and to capture the substrate noise, so as to detect the substrate coupling effects.

4.2.1 Noise coupling in substrate

Mixed-signal designs become the main trend of IC industry as the process technology scaling at ultra-deep submicron range. Lots of noticeable advantages accompanied by noise coupling through substrate in mixed-signal circuits lead to new challenge for circuit design. Substrate coupling, as the most important effects, degrades the performance of the analog circuits located on the same chip. An example of a substrate coupling problem is given below:

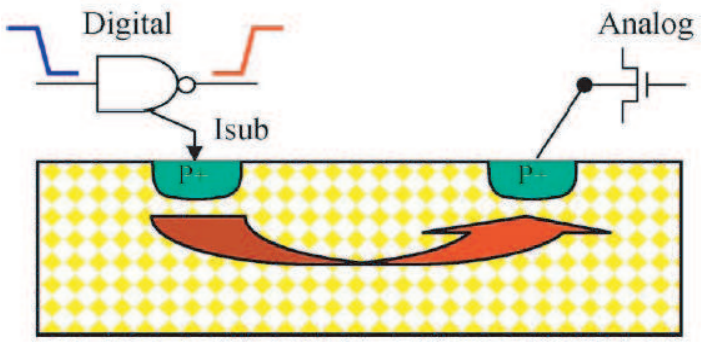

Figure 4.13: Noise coupling between analog and digital circuits [2]

Figure 4.13 shows that the switching noise due to large swing signal swings in the digital part can propagate through the substrate and corrupt sensitive analog part [14]. This is just one effect in the mixed-signal circuit, substrate coupling problem gives lots of effects in modern circuit design. Hence, we should accurately detect the substrate coupling effects at 45nm and below.

SPACE as an advanced layout-to-circuit extractor, can accurately extract parasitics parameters. SPACE 3D is updated version of SPACE, it is a 3D layout-to-circuit extractor for 45 degree poly geometries [6]. SPACE 3D can perform accurate substrate resistance and capacitance extraction by using 3D boundary element method BEM (Boundary Element Method) that has been used in SPACE for substrate resistance

and interconnect capacitances calculation. More information can be found in [15], [16], [17].

4.2.2 Ring Oscillator simulation

A ring oscillator is a device composed of an odd number of NOT gates whose output oscillates between two voltage levels, representing true and false [18]. The NOT gates, or inverters, are attached in a chain, the output of the last inverter is feed backed into the first. The last output of a chain of an odd number of inverters is the logical NOT of the first input. This final output is asserted a finite amount of time after the first input is asserted; the feedback of this last output to the input causes oscillation.

Here, one 9-stages ring oscillator is used to test the SPACE substrate resistance extraction capability of 45nm technology process, and the structure of this ring oscillator is shown in Figure 4.14.

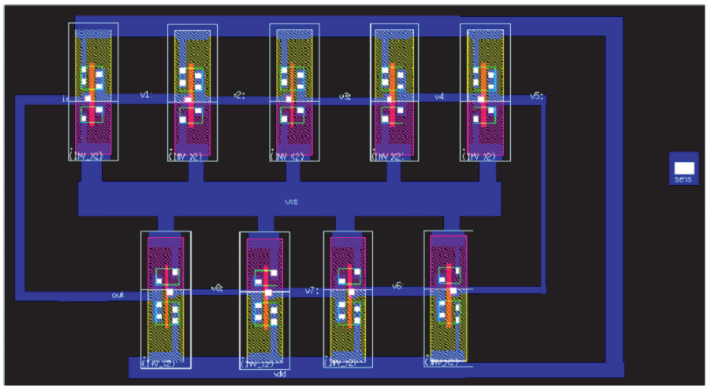

Figure 4.14: 9-stages ring oscillator in SPACE

The outer ring is the ground line which is composed of metal1 layer. It is also connected to the substrate through a via. Metal1 layer is also used in inner ring to carry the actual oscillating signal. The label "sens" is substrate sensor to detect the substrate noise. The whole structure is situated on top of a p-type substrate with a conductivity of S/m. The dimension of the p-MOSFETS are 270(nm)/50(nm), and they connect to the inner supply line (VDD). These transistors are embedded in an n-well. They have a connection to the substrate through the junction capacitance between n-well and p-substrate. The dimension of the n-MOSFET are 180(nm)/50(nm). They connect to the outer ring and have a direct connection to the substrate underneath their gates.

Meanwhile, the reference schematics simulation of the 9-stages ring oscillator is simulated by Cadence. Figure reffig:cadenceringoscillator shows the 9-stages ring oscillator structure in Cadence. The sizes of the PMOS and CMOS are w/L=50/270, w/L=50/180. And the final frequency equals *3.6GHz*.

Figure 4.16 and Figure 4.17 indicate the simulation results of the 9-stages Ring Oscillator (*freq=3.367GHz*) in SPACE and in Cadence (*freq=3.6GHz*) separately.

P.S. Due to the unfound reason, the layout of Nangate 45nm Open Cell Library can not be edited in Cadence. Hence, we can not get the layout simulation result here.

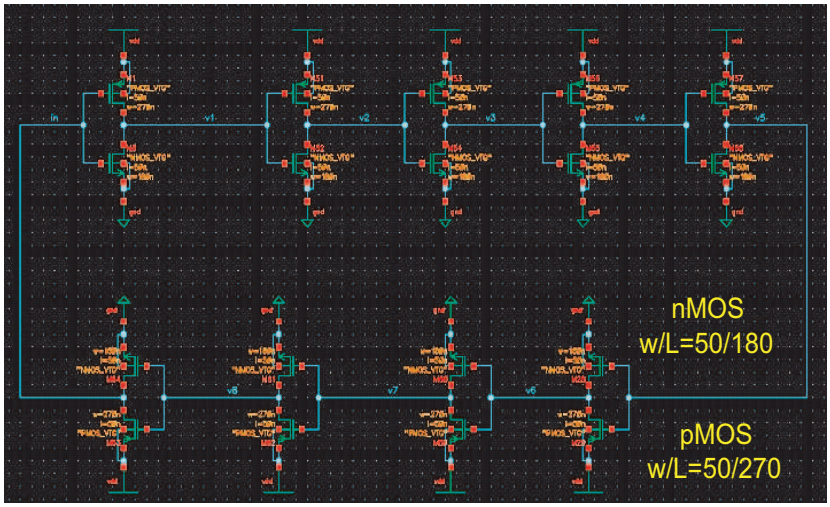

Figure 4.15: 9-stages ring oscillator in Cadence

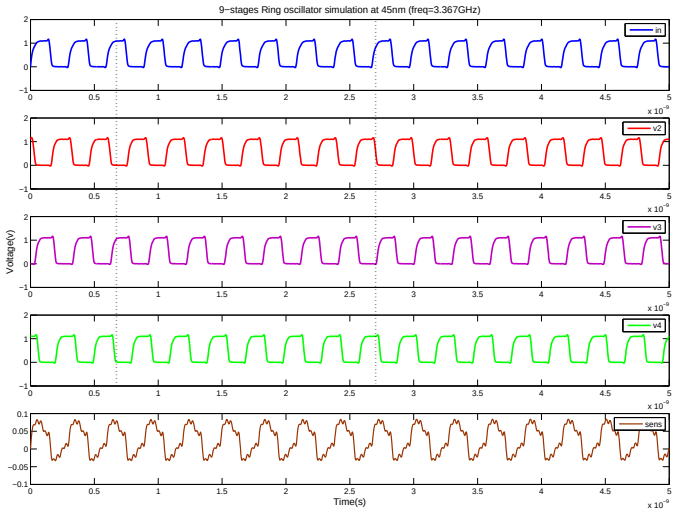

Figure 4.16: Substrate noise simulation of 45nm

4.2.3 Substrate resistance extraction

According to the BEM method, SPACE 3D extraction is used to perform the 3D substrate resistance extraction of cell ring oscillator. Figure 4.18 indicates the summary of the 3D substrate resistance extraction, and the complete parasitics netlist can be found in Appendix C.

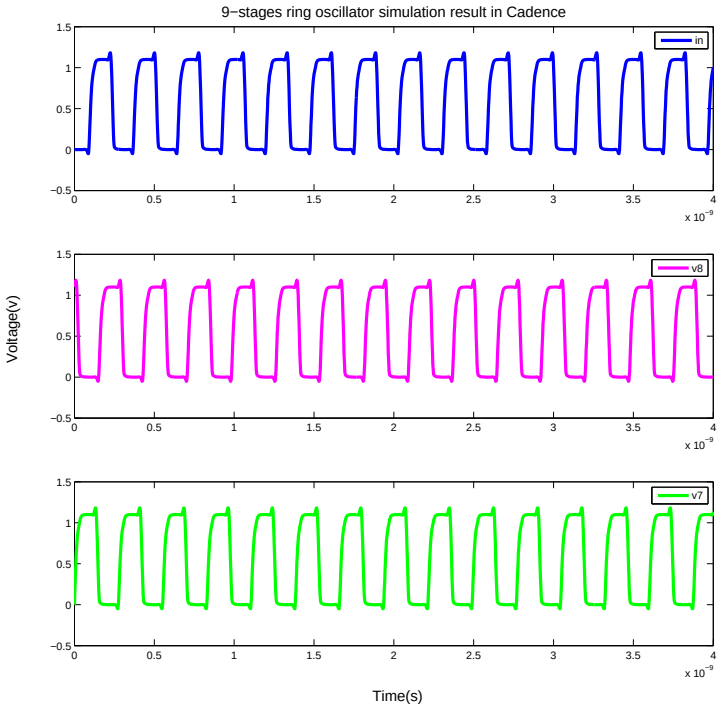

Figure 4.17: Cadence simulation result of 9-stages ring oscillator

```
extraction statistics for layout osc_final_ver2:
        capacitances        : 100
        resistances         : 351
        nodes               : 49
        mos transistors     : 18
        bipolar vertical    : 0
        bipolar lateral     : 0
        substrate terminals : 26
        substrate nodes     : 26

overall resource utilization:
        memory allocation  : 2813.411 Mbyte
        user time          :         0.0
        system time        :         0.0
        real time          :         5.8   1.0%
```

space 3D extracton summary

Figure 4.18: Ring oscillator 3D substrate resistance extraction summary

4.3 Large demo design

After the FreePDK 45nm technology files are validated, one large demo design is developed to demonstrate the extraction speed and memory requirements of SPACE. One synthesizable VHDL model of 8051 [13] is chosen as the example in this project. It is an 8-bit micro-controller, which is capable of addressing 64K of program memory and 64K of data memory.

There are several good reasons for choosing this VHDL Model. Firstly, it is an open IP core, which is provided by Dalton project of University of California. We can freely use this model; Secondly, it provides synthesizeable VHDL code and models the actual Intel implementation rather closely [13];

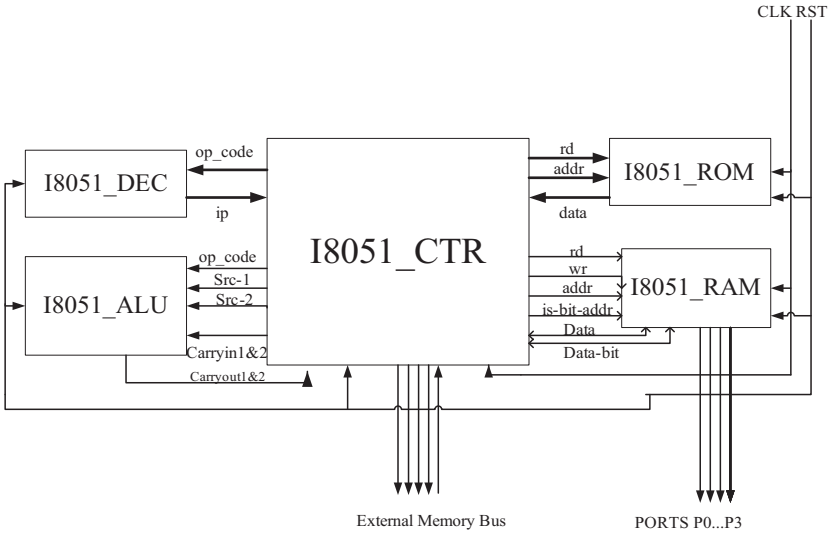

Figure 4.19: Dalton Intel 8051 structure diagram

Following list indicates the description of 8051 microprocessor related files [13]:

- *I*8051_*LIB.vhd*: Defines a package that is used in all the VHDL files of the 8051 model. This package defines commonly used constants.
- *I*8051_*ALU.vhd*: Model of an ALU that performs 8051 specific arithmetic. This model is described behaviorally as a combinational logic block.
- *I*8051_*DEC.vhd*: Model of a decoder that decodes the non-uniform 8051 instructions into uniform representations.
- *I*8051_*RAM.vhd*: Model of 128 bytes of SRAM, specific to 8051.
- *I*8051_*ROM.vhd*: Model of up to 64K bytes of ROM, specific to 8051. This model is automatically generated.
- *I*8051_*CTR.vhd*: Model of the core 8051 processor. This model is described behaviorally as a sequential logic block.

- *I*8051_*mkr.c*: Program to convert an Intel 8051 HEX file into a ROM model.
- *I*8051_*All.vhd*: Model of a complete 8051 micro-processor. This model structurally combines the above logic blocks.

Dalton Intel 8051 model also has the limitations: interrupt handling is not currently implemented, peripheral devices are not currently implemented [13]. However, for our project, these limitations can be ignored.

According to the design and verification flow (Figure 4.20), one 8051 processor is developed and verified. The files mentioned above are used here to develop the whole 8051 processor. This is very simple 8051 structure, and the instructions are generated from 8051 ROM. This ROM is automatically generated behaviorally using *I*8051_*mkr.c*.

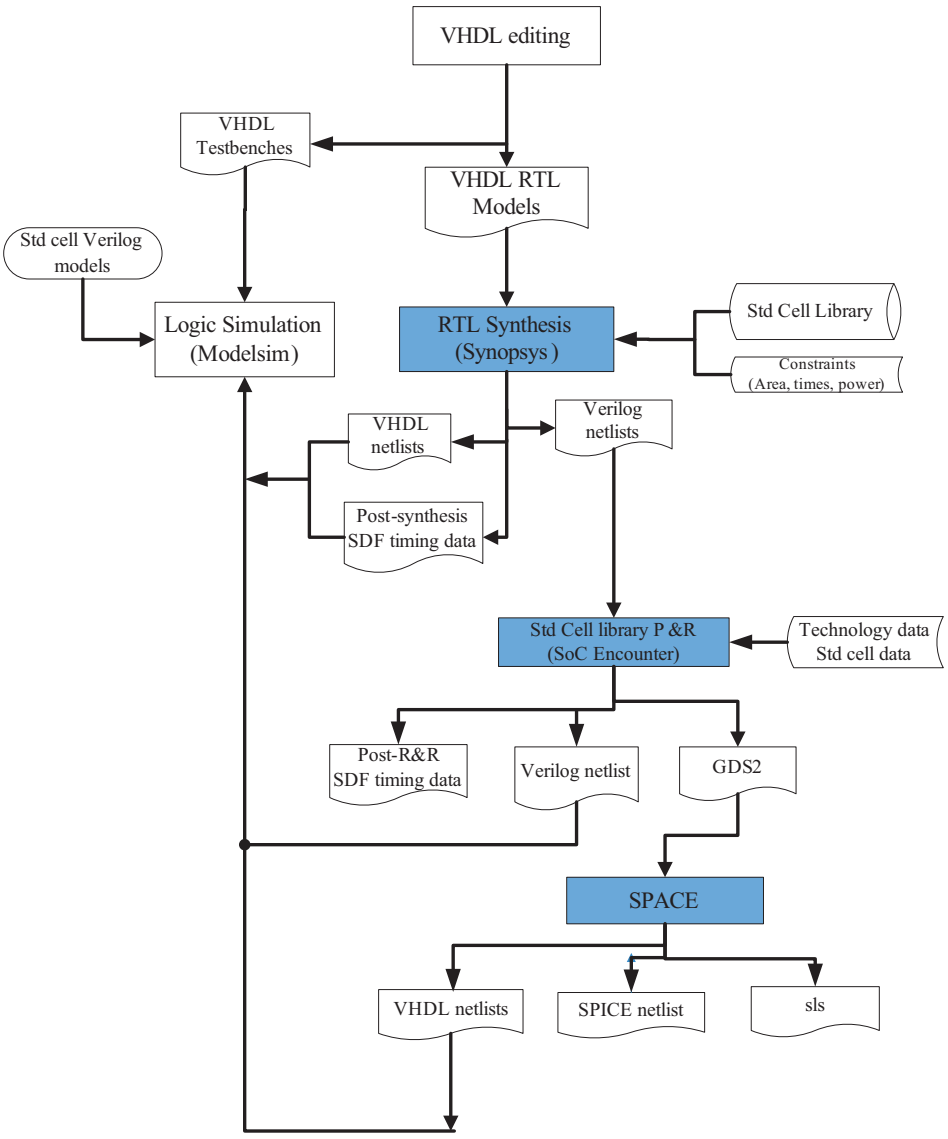

Figure 4.20: Top-Down design and verification Flow

In Dalton project, there are some examples can be used to generate 8051 ROM model. Here, we use the *negcnt.c* as the example. According to the instructions of this example, the microprocessor generates the signals in *port P0* (Figure 4.19), and the signal trace of the *port P0* is shown in Table 4.2, the 8051 microprocessor is counting from 64 to 73 in decimal. This example is very simple and convenient to test function operation of 8051 microprocessor.

Table 4.2: Signal trace for 8051 microprocessor in port P0

P0	64	65	66	67	68	69	70	71	72	73

In this project, design and verification flows can be divided into three parts: 1. Synthesis flow; 2. Place & Route flow; 3. SPACE extraction and verification flow. Note that, the function validation should keep consistent along three steps.

4.3.1 Synthesis and P & R process flow

In this part, two EDA tools are used: *Design Compiler* (Synopsys), *SoC Encounter* (Cadence). These tools can help us to finish design synthesis and floorplan, place & route processes.

Figure 4.21 shows the synthesis flow:

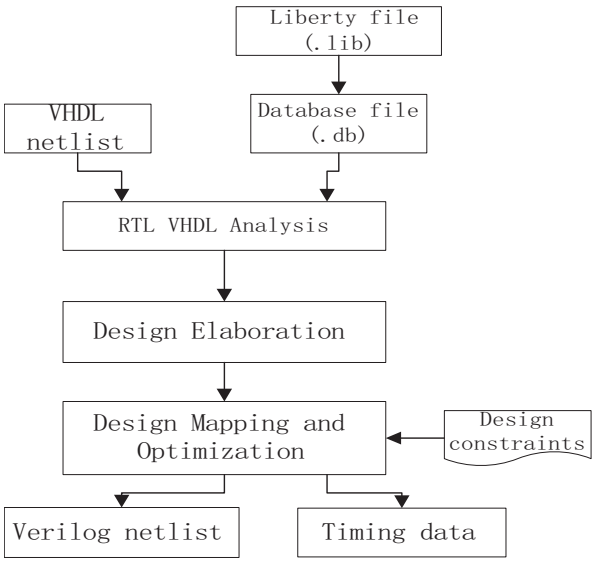

Figure 4.21: Synthesis flow

The first step of synthesis is loading Synopsys database file. Unfortunately, Nangate Open Cell Library does not contain Synopsys database file, we need to create the "**.db**" (Synopsys database file) based on the "**.lib**" (Liberty file), using Library Compiler (read liberty and then write the db file). After library translation, we compile the VHDL Model based on the scripts, which is list in Appendix E.1.

Note that, there should not be *negative slack* (set the maximum area is zero) in timing report, otherwise, we have to optimize our design again.

Figure 4.22 and Figure 4.23 show the Modelsim Simulation of the original and post-synthesis of 8051 microprocessor. The same testbench is used here. It is clearly seen that results are matchable (**P0 = 64 65 ... 73**).

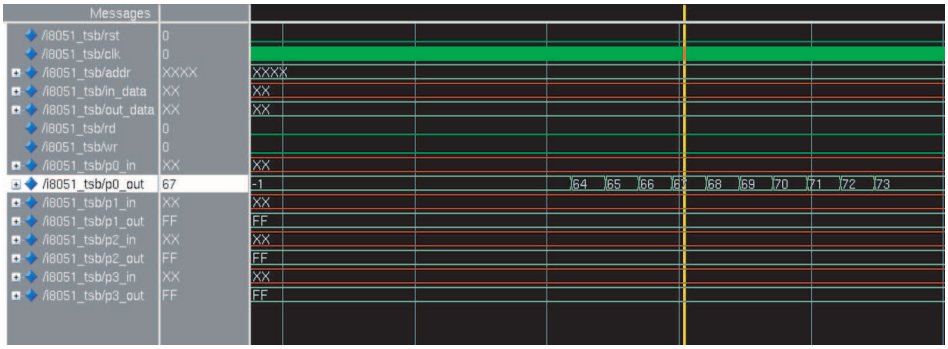

Figure 4.22: Simulation result of original VHDL model

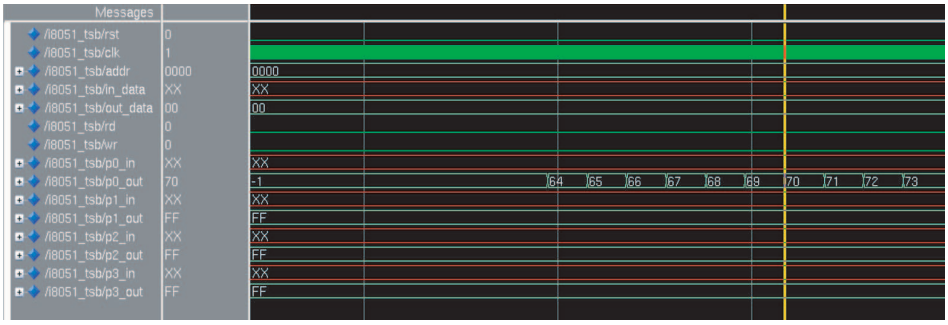

Figure 4.23: Simulation result after synthesis

4.3.2 Place & Route flow

Figure 4.24 shows the steps of Place & Route with SoC Encounter. After synthesis, the verilog netlist, *.sdc* timing file, *LEF* file and common timing library file which are generated by Design Compiler are loaded into the SoC Encounter, to run P&R process. The whole procedure is based on the Encounter scripts, which is offered in Appendix E.2.

Figure 4.25 indicates the Modelsim Simulation results of the 8051 processor post-SoC Encounter. It matches to the original VHDL Model simulation result.

Figure 4.26 shows the physical layout view and floorplan layout view post P&R process.

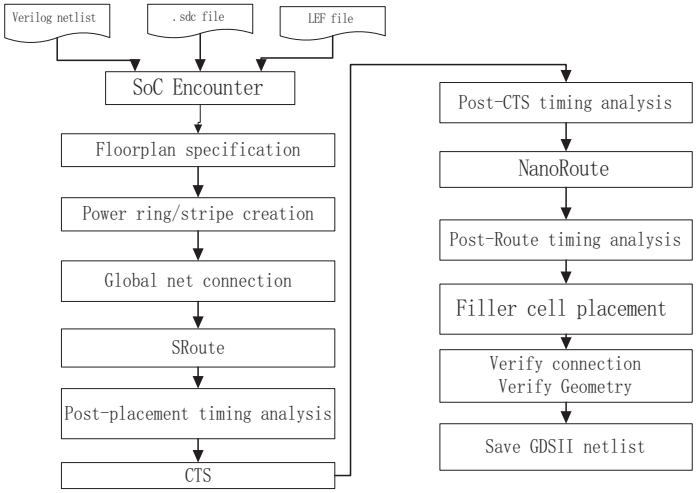

Figure 4.24: Place and Route flow

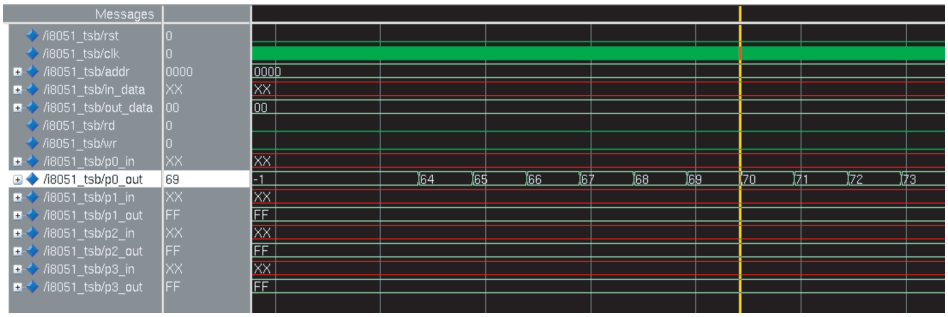

Figure 4.25: Simulation result after encounter

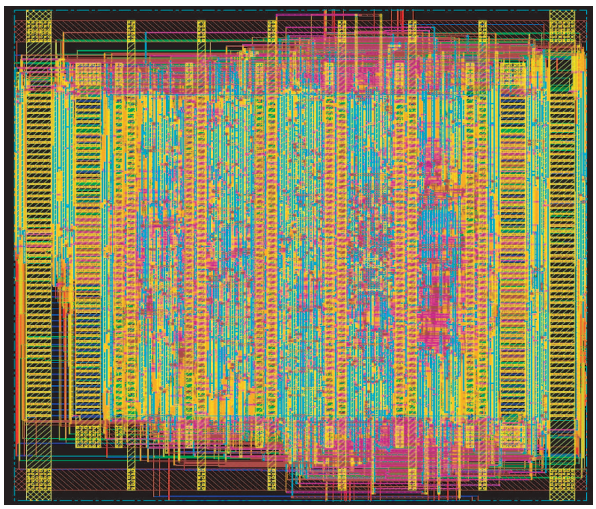

Figure 4.26: Physical layout view

4.3.3 SPACE and Calibre extraction and verification flow

In this part, extraction and verification flow with SPACE are implemented. Figure 4.27 shows the flow of SPACE extraction and verification.

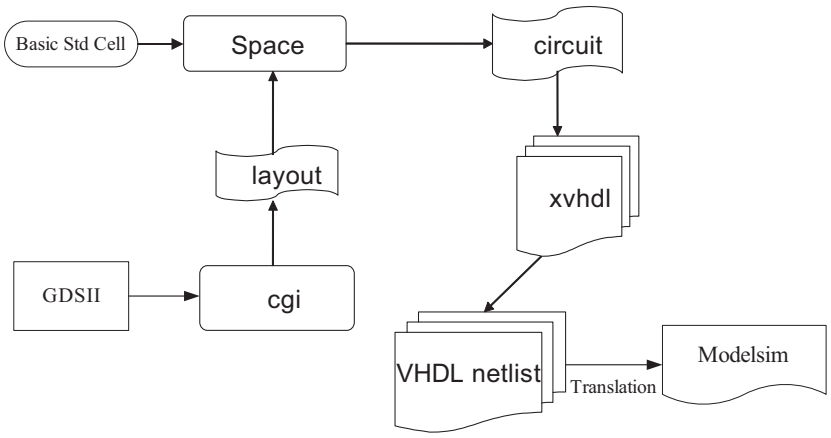

Figure 4.27: SPACE extraction and verification flow

Here, we firstly import 8051 processor GDSII file that generated by SoC Encounter into the SPACE database. Meanwhile, we add the Nangate Open Cell Library standard cells into our design project. After that, we can use **cgi** (convert GDSII files to an ICD project) command with encounter *bmlist.gds* file to generate the layout of 8051 processor in SPACE. The circuit can be achieved based on **space** (layout to circuit extractor) command. Successfully generated layout and circuit of the design, VHDL netlist can be extracted in order to test the logic function operation.

Note that, extracting VHDL file in SPACE can only give the bit signals, and the output of the SPACE VHDL netlist is not the standard VHDL netlist, hence, *Translation* (Figure 4.27) program is used to change this VHDL into the standard format to test the logic operation easily and conveniently in Modelsim. The *Translation* program is shown in Appendix A.2.

Figure 4.28 indicates the Modelsim simulation results after the design is imported into SPACE. Figure 4.29 shows the layout view in **dali** (an interactive layout editor). Compared with original, post-synthesis and post-encounter results, logic function is matchable during the whole verification flow.

To sum up, the function validation of the large demo keeps consistent during the synthesis flow, Place & Route flow and SPACE extraction & verification flow. Meanwhile, according to the Figure 4.20 and Figure 4.27, the capability of SPACE to extract large demo design is proven. Table 4.3 lists the extraction statistics for 8051 microprocessor.

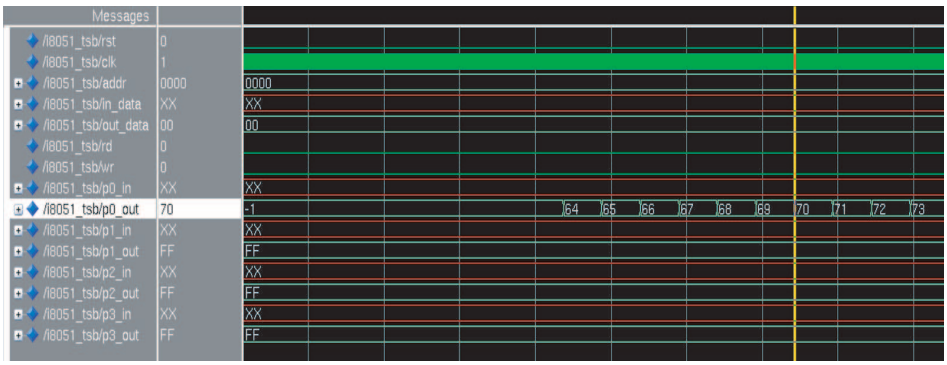

Figure 4.28: 8051 processor simulation result after SPACE

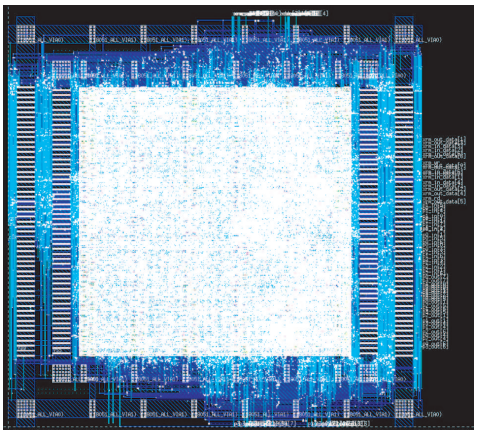

Figure 4.29: 8051 processor layout view in SPACE

Table 4.3: SPACE extraction statistics of 8051 microprocessor

Name	Statistics
cells	13891
user time	1:22.9 (min. : sec.)
system time	0.7 (sec.)
real time	1:54.2 (min. : sec.) 73.2% (system utilization)

Verification flow comparison and SPACE 2D & 3D discussion

5

After technology files development and verification with SPACE. We are going to compare the verification flow of SPACE and Calibre, in order to find the strong and weak points of these two EDA tools. Moreover, the trade-off between SPACE 2D and 3D extraction is discussed here according to the interconnect capacitance extraction.

5.1 Comparison of SPACE & Calibre verification flow

Layout Versus Schematic (LVS) is one of the most important steps in verification flow that determines whether a particular integrated circuit layout corresponds to the original design schematic. A successful Design rule check (DRC) ensures that the layout conforms to the rules required for faultless fabrication. And LVS guarantees if it really represents the circuit of your design. The LVS contains three steps, shown in Figure 5.1:

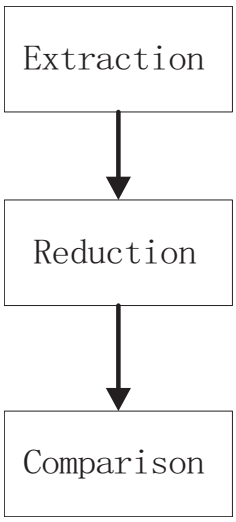

Figure 5.1: LVS flow

1. Extraction: Program takes a database file containing all the layers drawn to represent the circuit during layout [19]. It firstly determines the semiconductor components represented in the layout drawing by their construction. And then it finds the way of metal layers connection.

2. Reduction: Program combines the extracted components into series and parallel combinations. Meanwhile, it generates layout netlist [19].

3. Comparison: Comparison between layout extracted netlist and schematic is going to show the results if the two netlists match [19].

Several commercial LVS softwares are used in modern circuit design, such as, $L-EditLVS$ by Tanner, *Calibre* by Mentor Graphics, $QuartzLVS$ by Magma, etc. Here, we will compare Calibre LVS flow with SPACE $Match$ flow based on the 45nm Nangate OpenCell Library. The platform is: Intel(R)4 CPU 3.00GHz, Memory 1010.3MiB.

Two verification flows are shown in Figure 5.2 and Figure 5.3

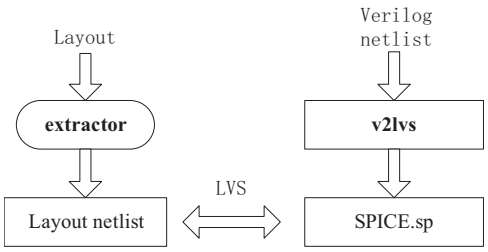

Figure 5.2: Calibre LVS flow

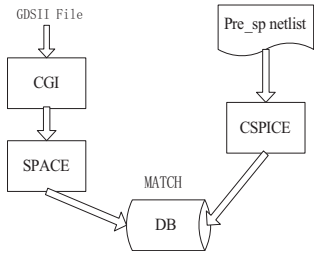

Figure 5.3: SPACE Match flow

According to the Figure 4.20, after synthesis and place&route, the circuit netlist and the GDSII file can be imported into the Calibre. Running Calibre LVS, getting layout to match its corresponding schematic. The same netlist and the GDSII file can be added into SPACE database through command *cgi* and *cspice*. SPACE Match gives the summary report.

Note that, SPACE can not convert HDL description into Spice netlist, so that, the reference Spice netlist can also be extracted by *v2lvs* or other third part software.

Based on the flows Figure 5.2 and Figure 5.3, layout extraction netlist and LVS summary can be printed. We can compare these two flows according to the netlist and LVS summary. In the following part, we are going to discuss the netlist difference between SPACE and Calibre extraction. The methodology for SPACE extraction is based on the FET condition list, the parameters *ad*, *as*, *pd*, *ps*, *nrs* and *nrd* for the transistor can be calculated as capacitance extraction is enable. However, Calibre use different methodology to extract these parameters.

5.1.1 Simple layout extraction comparison

In this section, we still use INV-X1 as the demo example to show the disparity in netlist extraction between Calibre and SPACE.

Table 5.1 indicates the comparison results among SPACE, Calibre and Nangate reference netlist. It can be clearly seen that the netlist extracted by SPACE and Calibre are the same except peripheries extraction.

Here, w is the transistor; l is the length of the transistor; AS is the source area and AD is the drain area; PS and PD are source/drain perimeters; *nrs* and *nrd* are the equivalent source/drain squares.

Table 5.1: Comparison of INV-X1

	SPACE		Calibre		Nangate-ref	
	NMOS	PMOS	NMOS	PMOS	NMOS	PMOS
W	90n	135n	90n	135n	90n	135n
L	50n	50n	50n	50n	50n	50n
AS	9.45f	14.175f	9.45f	14.175f	9.45f	14.175f
AD	9.45f	14.175f	9.45f	14.175f	9.45f	14.175f
PS	300n	345n	390n	480n	390n	480n
PD	300n	345n	390n	480n	390n	480n
nrs	1.16667	1.16667				
nrd	0.777778	0.777778				

Due to the different perimeter calculation methodology between SPACE and Calibre, netlist extraction shows the different results.

For SPACE, in the perimeter of the drain and source, the edge that coincides with the transistor channel, is not included. If more than one transistor is connected to a drain/source region, the area value that is obtained for the total drain/source region is subdivided over the different transistors in proportion to the widths of the transistors(Figure 5.4–right). The perimeter value is split in a similar way [6].

For Calibre, in the perimeter calculation of the drain and source, the edges which coincide with the transistor channel are computed. When more than one transistor is connected to a drain/source region, the area of the drain/source is reused by each transistor. The same way is used in the perimeter calculation (Figure 5.4–left).

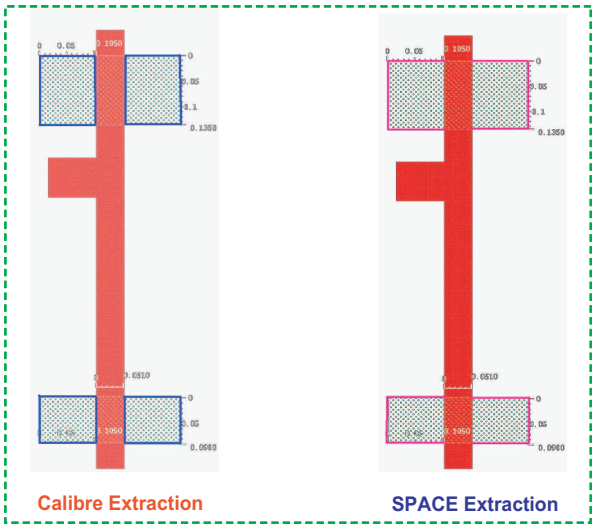

Figure 5.4: Comparison of Calibre and SPACE extraction

5.1.2 Complicated layout extraction comparison

In order to understand the difference of netlist extraction completely, one complicated (AOI22-X2) layout extraction is considered in this section, to indicate the SPACE and Calibre layout extraction difference.

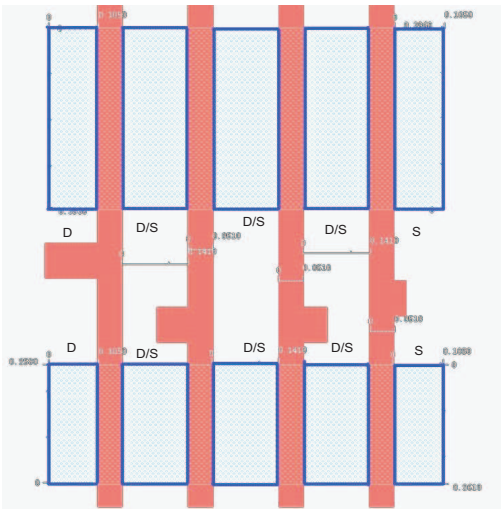

Figure 5.5: AOI layout extracted by Calibre

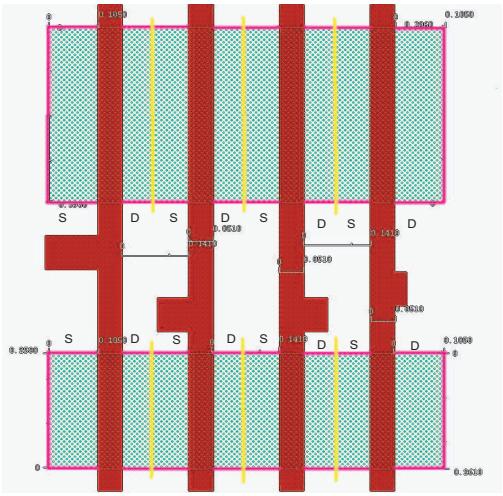

Figure 5.6: AOI layout extracted by SPACE

Figure 5.5 (blue lines) indicates the Calibre extracted region. It can be clearly seen that as two or more transistors connect to the same drain/source region, Calibre reuse the middle active region as both the drain and the source of each transistor. Moreover, for perimeter calculation, Calibre adds the four edges of the drain/source region.

However, netlist extracted by SPACE that indicated in Figure 5.6 (pink lines) gives the different results. SPACE divides the middle source/drain region, one half part is the drain, another half part is the source. SPACE adds three active region edges without considering the edge nearby the poly. Based on this way, SPACE calculates the first or the last transistors drain/source perimeters. But for the internal active region, SPACE only adds the top and the bottom edges of the active region to calculate the drain/source perimeters (Figure 5.6).

There are various S/D diode model in different commercial circuit simulators. Hence, even users use the same BSIM model for the intrinsic part of the MOSFET, they still get the different results from different simulators [20]. Moreover, the calculation for circuit parameters AS, AD, PS, PD according to users own definitions, or following the same definitions of AS, AD, PS, PD as those given in simulators.

As examples here, we demonstrate the AS, AD, PS, PD calculation according to two simulator, HSPICE and ELDO. For ELDO S/D diode models, Figure 5.7 indicates the geometry information of one device. The values of AS, AD, PS, PD are calculated based on the Equation 5.1 to Equation 5.4.

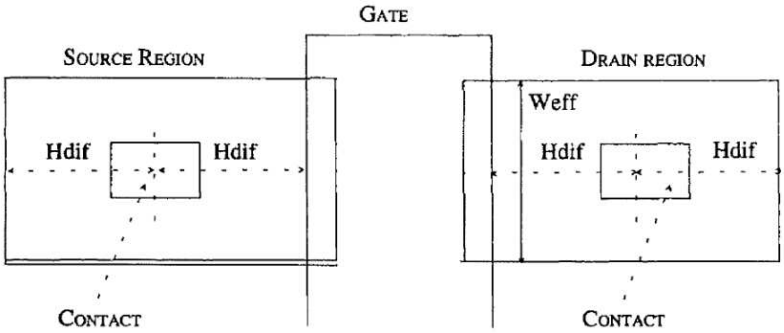

Figure 5.7: A simple device layout plot

$$AD = 2 \times H_{dif} W_{eff} \tag{5.1}$$

$$AS = 2 \times H_{dif} W_{eff} \tag{5.2}$$

$$PD = 4 \times H_{dif} + 2 \times W_{eff} \tag{5.3}$$

$$PS = 4 \times H_{dif} + 2 \times W_{eff} \tag{5.4}$$

It can be clearly found that the ELDO methodology for S/D diode model extraction is the same with Calibre diode model extraction.

Moreover, HSPICE includes four S/D diode model, which can be selected with the model parameter **ACM** (Area Calculation Method) [20].

1. ACM=0 SPICE model, parameters determined by element areas
2. ACM=1 ASPEC model, parameters function of element width
3. ACM=2 META model, combination of ACM=0, 1 and provisions for lightly doped drain technology
4. ACM=3 Extension of ACM=2 model that deals with stacked devices (shared source/drains) and source/drain periphery capacitance along gate edge.

In HSPICE, the definitions of AS, AD, PS and PD are defined by GEO parameter as ACM=3 [20]. As the ACM=3, the effective areas and peripheries are calculated differently, depending on the value of GEO (element parameter). GEO can be specified on the MOS element description [21]. It can have the following values:

- GEO=0: indicates the drain and source of the device are not shared by other devices (default).
- GEO=1: indicates the drain is shared with another device.

- GEO=2: indicates the source is shared with another device.
- GEO=3: indicates the drain and source are shared with another device.

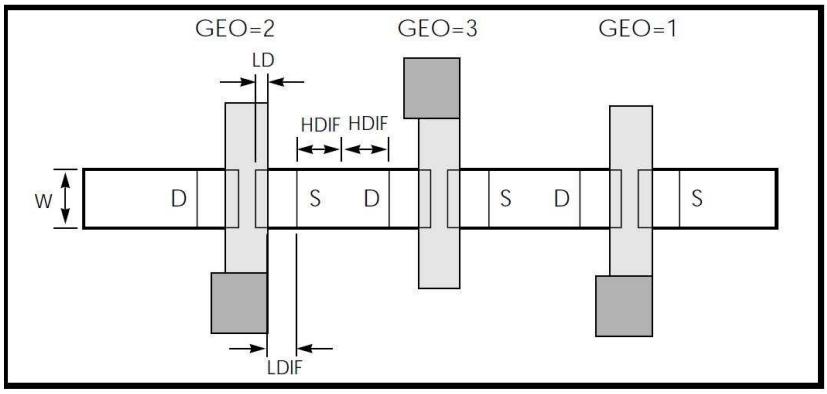

Figure 5.8: Stacked devices layout in HSPICE S/D diode model

Figure 5.8 shows the layout of the stacked devices. Based on the different GEO values, we can get the different methodology for calculation of AD, AS, PD, PS [21].

1. If AD is not specified, then,
For GEO=0 or 2,

$$AD_{eff} = 2 \times HDIF_{eff} \times W_{eff}$$

For GEO=1 or 3,

$$AD_{eff} = HDIF_{eff} \times W_{eff}$$

2. If AS is not specified, then,
For GEO=0 or 1,

$$AS_{eff} = 2 \times HDIF_{eff} \times W_{eff}$$

For GEO=2 or 3,

$$AS_{eff} = HDIF_{eff} \times W_{eff}$$

3. If PD is not specified, then,
For GEO=0 or 2,

$$PD_{eff} = 4 \times HDIF_{eff} + W_{eff}$$

For GEO=1 or 3,

$$PD_{eff} = 2 \times HDIFeff$$

4. If PS is not specified, then,
For GEO=0 or 1,

$$PS_{eff} = 4 \times HDIF_{eff} + W_{eff}$$

For GEO=2 or 3,

$$PS_{eff} = 2 \times HDIF_{eff}$$

We can clearly found that the HSPICE S/D diode model extraction is similar with the SPACE extraction.

To sum up, due to the simulator-related source and drain diode models, sometimes we may get different results from different simulators. In this project, we found out that Calibre and ELDO use the same S/D diode model; however, the S/D diode models for SPACE and HSPICE (as ACM=3) are similar.

5.2 SPACE 2D & 3D capacitance extraction discussion

SPACE 2D is a 2D hierarchical layout to circuit extractor. It can accurately extract lateral and crossover coupling capacitance [6] with the cost of the accuracy. Space 3D is a 3D hierarchical layout to circuit extractor for 45 degree polygonal geometries. Moreover, SPACE 3D is used to accurately and efficiently compute 3D interconnect capacitances of integrated circuits based upon their mask layout description [11]. However, the speed of SPACE 3D is slower than SPACE 2D.

Here, we want to verify the validation of 2D capacitance rules that generated by **tabs** (technology abstraction tool). **Tabs** is used to generate the capacitance rules automatically for the 45nm technology [9]. Firstly, it computes the area capacitances that are the capacitances associated with the overlap area of two conductors. Meanwhile, it calculates the vertical edge-edge capacitances that are defined as the capacitances between the edges of two conductors on different layers. Furthermore, it computes the edge-surface capacitances that defined as the capacitances between a side-wall of one conductor, and the top/bottom of a conductor on a lower/higher plane. Finally, it computes the lateral capacitances—the capacitances between conductors in the same layer.

According to the capacitance rules that generated by **tabs**, the interconnect capacitance can be extracted by SPACE 2D. If the extraction values keep consistent with SPACE 3D, the capacitance rules can be validated.

Here, we use the INV-X1 as the demo design. Based on the technology files, the SPACE 3D extraction can be easily implemented. However, for SPACE 2D extraction, **tabs** is used firstly to add the capacitance rules into technology files automatically and then extract capacitances. Table 5.2 lists the capacitance extraction results:

Due to the original technology files do not contain the capacitance definition, SPACE 2D can not run the capacitance extraction. In this project, **tabs** (technology abstraction tool) is used to generate the capacitance rules automatically for the 45nm technology [9]. Firstly, it computes the area capacitances that are the capacitances associated with the overlap area for two conductors. Meanwhile, it calculates the vertical edge-edge capacitances that are defined as the capacitances between the edges of two conductors on different layers. Furthermore, it computes the edge-surface capacitances that defined as the capacitances between a side-wall of one conductor, and the top/bottom of a conductor on a lower/higher plane. Finally, it computes the lateral capacitances—the capacitances between conductors in the same layer.

Based on these capacitance rules, SPACE 2D extraction can be used to calculate the interconnect capacitances of the circuit. Here, we are going to prove the validation of the capacitance rules that generated by **tabs**. The way is to compare the SPACE 2D and 3D interconnect capacitance extraction results. SPACE 2D can accurately extract lateral and crossover coupling capacitance [6]. Moreover, SPACE 3D is used to

accurately and efficiently compute 3D interconnect capacitances of integrated circuits based upon their mask layout description [11]. If these two results keep consistent, the 2D capacitance rules are verified.

Table 5.2: Comparison of SPACE 2D & 3D Extraction results of INV-X1

	SPACE 2D extracted capacitance	SPACE 3D extracted capacitance
ZN $\Longrightarrow$ VSS	26.71e-18	41.5e-18
A $\Longrightarrow$ VSS	30.139e-18	26.7e-18
A $\Longrightarrow$ ZN	8.81e-18	43.63e-18

From Table 5.2, we can find out that there is a large difference of A $\Rightarrow$ ZN between SPACE 2D and 3D capacitance extraction. We are going to use the reduced layout of INV-X1 (Figure 5.9) to find out the reason. Figure 5.9 indicates the reduced layout, here we just consider the two conductors—poly and metal1, without any diffusion or implantation.

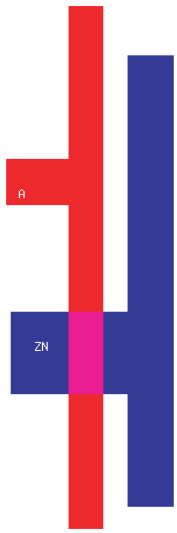

Figure 5.9: Reduced layout of INV-X1

Using SPACE 2D and 3D to extract the reduced layout capacitance values again. The equivalent circuit is shown in Figure 5.10:

It can be clearly found that the difference between node A and ZN. Due to SPACE 2D only calculates the capacitances of the overlapping areas, so that, the 2D extracted capacitances are smaller than 3D extraction.

Furthermore, we also consider the case of lateral capacitances extraction of this reduced layout (Figure 5.9). Firstly, we need to add the lateral capacitance definition of this reduced layout into the technology files. The lateral capacitance defined in Equation 5.5:

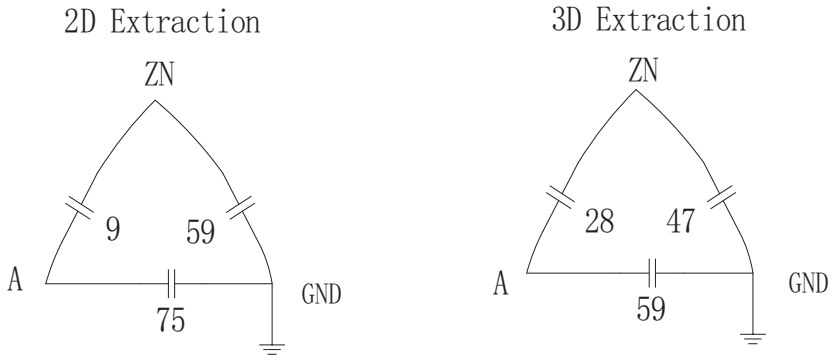

Figure 5.10: Equivalent capacitance view before adding lateral cap. rules (Unit: e-18 F)

$$C_l = c_l \frac{l}{d} \tag{5.5}$$

l is the length of the conductor and d is the space between two conductors. c_l corresponds to the capacitance for a configuration where the distance between both wires is equal to their length [6].

Here, two parallel conductors, *poly* and *metal1*, are used to compute the value of c_l (Figure 5.11). And the length of two conductors are 1.5μm, the space between two conductors are 1.5μm.

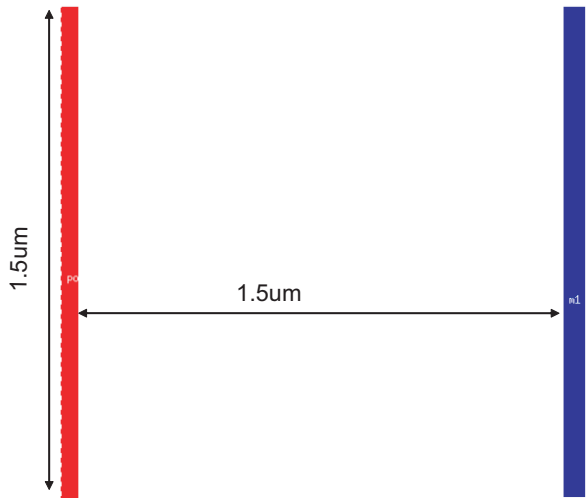

Figure 5.11: Two conductors structure

Firstly, we use SPACE 3D extraction to extract c_l value.

$$c_l = 0.001072\ fF \tag{5.6}$$

We add c_l into the technology file (Appendix D line(159–160)), in order to calculate the lateral capacitances of this reduced layout based on SPACE 2D. Figure 5.12

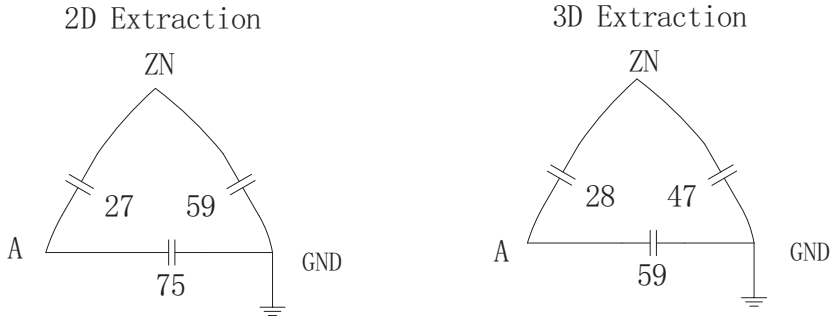

Figure 5.12: Equivalent capacitance view after adding lateral cap. rules (Unit: e-18 F)

indicates the SPACE 2D and 3D lateral capacitance. It can be clearly seen that, the lateral capacitance of node A ⇒ ZN is almost the same.

Moreover, we are going to compare the short-circuit capacitance of SPACE 2D and 3D extraction. For a multiconductor system, the relationship between potential and charge can be expressed:

$$V = G \times Q \quad Q = C_s \times V \tag{5.7}$$

Here, G is coefficients of potential, and C_s is the short-circuit matrix.

C_{sii} is the element of short-circuit capacitance matrix C_s represents the total capacitive load a conductor forms its driving stages as all other conductors are shorted to GND; C_{sij} is defined as the charge on conductor i as the j-th conductor is held at unit potential and all other conductors are short-circuited to GND. For this reduced model (Figure 5.9), we can get the short-circuit capacitance matrix:

$$\begin{aligned} C_s &= \begin{pmatrix} C_{sAA} & C_{sAB} \\ C_{sBA} & C_{sBB} \end{pmatrix} \\ C_{sAA} &= C_{AGND} + C_{AB} \\ C_{sBB} &= C_{BA} + C_{BGND} \end{aligned} \tag{5.8}$$

However, the total short-circuit capacitances of node A and node ZN are showing unmatchable (Table 5.3)

Table 5.3: Short-circuit capacitances of node A → ZN after adding lab-cap. rules

	SPACE 2D extraction	SPACE 3D extraction
C_{sAA}	102e-18 F	87e-18 F
C_{sBB}	86e-18 F	75e-18 F

It can be concluded that the capacitance rules are generated by **tabs** are not really accurate in this case. We still need to improve the capacitance rules. Moreover, SPACE 2D provides one more efficient way to extract the circuit than 3D, at the cost of some accuracy. However, extraction results of SPACE 3D are more accurate than 2D with the cost of speed. So that, choice of these two methods depend on the different projects and requirements.

6 Conclusion

6.1 Summary of results

In this project, we successfully built the FreePDK 45nm technology files from Nangate 45nm OpenCell Library. The element definition like *Layer specification*, *Vertical Dimension*, *Dielectric Structure*, *Field-effect Transistor*, *Conductor* and *Contact* were discussed and added in the technology files.

Several suitable examples were developed for technology files verification and demonstrating the important SPACE features. (1). Every basic cells were used as the demo design for validation of the technology files; (2). A ring oscillator is developed according to the 45nm technology to show the substrate resistance extraction capabilities on 45nm process; (3). A large design were built to demonstrate the extraction speed and memory requirements of SPACE.

Furthermore, Calibre and SPACE verification flow were compared based on some demo examples. Difference between these two tools like extraction parameters, extraction methodology were discussed in this project according to the reference ELDO and HSPICE S/D diode model. To sum up, since the differences of simulator-related source and drain diode models, we may get different results from different simulators.

Finally, The validation of SPACE 2D capacitance rules were verified based on the lateral capacitances of reduced INV-X1 layout. Meanwhile, the trade-off between SPACE 2D & SPACE 3D was discussed based on the interconnect capacitance extraction.

6.2 Further work

Although FreePDK 45nm technology files were successfully built for SPACE and demo design were developed, there are few recommendations for further research:

1. Substrate noise analysis at 45nm technology node;
2. SPACE extraction capability of super-huge demo design

Code for MATCH and Translation

A.1 domatch program

This python program demonstrates the **domatch** program, which is used to Match every standard cell automatically.

```
#!/usr/bin/env  python
#this is my exchang name of match  programe.

#######################################################
## Here, we should change the pre.sp file into the suitable
## format for "cspice", and then import it into DB, meanwhile
##import '.gds' into DB. Finally, match them
#######################################################

import os
os.system ('rmdb -af')
os.mkdir  ('./matchlog')
os.mkdir  ('./spicelog')

#################################################################
## Change cell name into cell_ref# ##########################
#################################################################
fin = open('NangateOpenCellLibrary_PDKv1_2_v2008_10_pre.sp')
fout = open('tt.txt', 'w')      #---middle variable1

import re
f=fin.readlines()
f2="".join(f)
p=re.compile(r'(\W+)')
f3=p.split(f2)

##loops for every parameters to change the cell name to cell_ref name
for i in range(0, len(f3)):
   if f3[i]=='SUBCKT':
     f3[i+2]+="_ref"

f4="".join(f3)

for line in f4:
   fout.write(line)                                  #write to tt.txt
fin.close()
fout.close()


#######################################################
```

```
## remove .GLOBAL ###################################
#####################################################
fi = open('tt.txt', 'r')
fo = open('ttnew_final.txt', 'w')                    #middle variable2

##delete the .GLOBAL for the correct syntax of match
for line in fi:

   if '.GLOBAL VDD' in line:
     line=line.replace('.GLOBAL VDD','*')
   if '.GLOBAL VSS' in line:
     line=line.replace('.GLOBAL VSS','*')

   fo.write( line )                              #write to ttnew_final.txt
fi.close()
fo.close()

#####################################################
##change  NMOS&PMOS model names into nenh&penh  ###
## and remove middle variables ####################
#####################################################
import os
import re
os.remove ("tt.txt")                      #remove middle variable1

## change the NMOS&PMOS model names into nenh&penh
os.system ('cpp -DNMOS_VTL=nenh -DPMOS_VTL=penh ttnew_final.txt >
   NangateOpenCellLibrary_PDKv1_2_v2008_10_pre2.spc')

os.remove ("ttnew_final.txt")#remove middle variable2

#############################################################
# start the SPACE operation ################################
#############################################################

#import gds into DB
os.system('cgi NangateOpenCellLibrary_PDKv1_2_v2008_10.gds')
#import ref.sp into DB
os.system('cspice NangateOpenCellLibrary_PDKv1_2_v2008_10_pre2.spc')
#read cell name
finl=os.popen("dblist -l").readlines()
#read reference cell name
finc=os.popen('dblist -c').readlines()

#change list into string
fl1="".join(finl)
fc1="".join(finc)

#change string back into list,but without "\n"
p=re.compile(r'\W+')
fl2=p.split(fl1)
fc2=p.split(fc1)
```

```
#define length of the list
length=range(len(fl2)-1)
for i in length:
    space='space '+fl2[i]              #space all cell in list
# match cell_ref and cell
    match='match '+fl2[i]+' '+fc2[i]+' -fullbindings'
    os.system (space)
    os.system (match+ '>&./matchlog/'+fl2[i]+'.malog' )
    spice='xspice '+'-aou '+fl2[i]
    os.system (spice+ '>&./spicelog/'+fl2[i]+'.splog')
```

A.2 Translation programme

This python programme is used to exchange the verilog netlist which is extracted by SPACE, into the suitable format for Modelsim simulation.

```
#!/usr/bin/env  python
### This is my new version exchange syntax of SPACE vhdl program ###
### It will replace VDD, VSS and "_numbers"                       ###
### However, fist you should change your input file name into "input.vhd"
    ###

fin = open('input.vhd')
fout = open('Xchanged.vhd', 'w')
f=fin.readlines()
library='library IEEE;\n' +'use IEEE.STD_LOGIC_1164.all;\n' +'use IEEE.
   STD_LOGIC_ARITH.all;\n'
f.insert(3, library)#insert new library definition

entity='ENTITY I8051_ALL IS\n'+
'   PORT (  rst: INOUT STD_LOGIC;\n'+'   clk: INOUT STD_LOGIC;\n'+
'   xrm_addr     : INOUT UNSIGNED (15 downto 0);\n'+
'   xrm_out_data : INOUT UNSIGNED (7 downto 0);\n' +
'   xrm_in_data  : INOUT  UNSIGNED (7 downto 0);\n' +
'   xrm_rd       : INOUT STD_LOGIC;\n'  +
'   xrm_wr       : INOUT STD_LOGIC;\n' +
'   p0_in        : INOUT  UNSIGNED (7 downto 0);\n'  +
'   p0_out       : INOUT UNSIGNED (7 downto 0);\n' +
'   p1_in        : INOUT  UNSIGNED (7 downto 0);\n'  +
'   p1_out       : INOUT UNSIGNED (7 downto 0);\n'  +
'   p2_in        : INOUT  UNSIGNED (7 downto 0);\n' +
'   p2_out       : INOUT UNSIGNED (7 downto 0);\n'  +
'   p3_in        : INOUT  UNSIGNED (7 downto 0);\n'  +
'   p3_out       : INOUT UNSIGNED (7 downto 0));\n'+'END I8051_ALL;\n'

for line in f:
   if 'LIBRARY' in line:
      line=line.replace('LIBRARY','--')
   if 'VDD: INOUT STD_LOGIC;' in line:
      line = line.replace('VDD: INOUT STD_LOGIC;' , '')
   if 'VDD: INOUT STD_LOGIC);' in line:
      line = line.replace('VDD: INOUT STD_LOGIC);' , ');' )
```

```
        if 'VSS: INOUT STD_LOGIC;' in line:
            line = line.replace('VSS: INOUT STD_LOGIC;' , '')
        if 'VSS: INOUT STD_LOGIC)' in line:
            line = line.replace('VSS: INOUT STD_LOGIC)' , ')' )
        if 'VDD, VSS,' in line:
            line = line.replace( 'VDD, VSS,', '' )
        if ', VDD, VSS);' in line:
            line = line.replace( ', VDD, VSS);', ');' )
        if ', VDD,\n VSS,' in line:
            line = line.replace( ', VDD,\n VSS,', ',' )
        if 'VDD,' in line:
             line=line.replace('VDD,','')
        if 'VSS,' in line:
             line=line.replace('VSS,', '')


    ##############################################################
    # Here, we changed the name of ports, since different
    # design will have different input/ouput names
    ##############################################################

        if 'addr_15' in line:
            line = line.replace('addr_15','addr(15)')
                  ... ... ...
              ... ... ...
        if 'addr_0' in line:
            line = line.replace('addr_0','addr(0)')
        if 'data_0' in line:
            line = line.replace('data_0','data(0)')
              ... ... ...
              ... ... ...
        if 'data_7' in line:
            line = line.replace('data_7','data(7)')
        if 'in_0' in line:
            line = line.replace('in_0','in(0)')
              ... ... ...
              ... ... ...
        if 'in_7' in line:
            line = line.replace('in_7','in(7)')
        if 'out_0' in line:
            line = line.replace('out_0','out(0)')
              ... ... ...
              ... ... ...
        if 'out_7' in line:
            line = line.replace('out_7','out(7)')

        fout.write( line)

    fin.close()
    fout.close()
```

45nm Nangate OpenCell Library GDSII number

B

B.1 GDSII number

GDSII number of 45nm Nangate OpenCell Library is shown in this section. Note that, Nangate provides the same GDSII number with layer number. It is from NangateOpenCellLibrary-PDKv1.2-v2008-05/openaccess/layer.map file.

```
#################################################################
#          Copyright (c) 2004-2008 Nangate Inc.                 #
#                All rights reserved.                           #
#################################################################

# Format: OpenAccess
#
# oa        oa         gds         gds         oa                oa
# LName     LPurpose   Layer       Purpose     Material          MaskNumber
NW          drawing        3           0           nWell          3
PW          drawing        2           0           pWell          2
NIMP        drawing        4           0           nImplant       4
PIMP        drawing        5           0           pImplant       5
DIF         drawing        1           0           diffusion      1
POLY        drawing        9           0           poly           9
CO          drawing        10          0           cut            10
M1          drawing        11          0           metal          11
M1T         pin            11          1           metal          11
V1          drawing        12          0           cut            12
M2          drawing        13          0           metal          13
M2T         pin            13          1           metal          13
V2          drawing        14          0           cut            14
M3          drawing        15          0           metal          15
V3          drawing        16          0           cut            16
M4          drawing        17          0           metal          17
V4          drawing        18          0           cut            18
M5          drawing        19          0           metal          19
V5          drawing        20          0           cut            20
M6          drawing        21          0           metal          21
V6          drawing        22          0           cut            22
M7          drawing        23          0           metal          23
V7          drawing        24          0           cut            24
M8          drawing        25          0           metal          25
V8          drawing        26          0           cut            26
M9          drawing        27          0           metal          27
V9          drawing        28          0           cut            28
M10         drawing        29          0           metal          29
BOUND       drawing        183         0           other          183
```

B.2 Vertical dimension definition

The Vertical dimension information is used for 3D extractions. It lists specify for the distance between the substrate and the bottom of a conductor, and the thickness of conductor. It is from file *calibrexRC.rul* of NangateOpenCellLibrary-PDKv1-2-v2008-10-SP1.

```
#########################################################################
##           Generated on Fri Aug 31 02:40:41 2007                     ##
##     by xCalibrate v2006.1_19.19 Tue Mar  7 12:29:27 PST 2006        ##
#########################################################################

   Background dielectric = 1

   Profile = field

 Layers =  (field_base field_base_diel poly poly_diel metal1
metal1_diel metal2  metal2_diel metal3 metal3_diel metal4
metal4_diel metal5 metal5_diel metal6 metal6_diel metal7
metal7_diel metal8 metal8_diel metal9 metal9_diel metal10 metal10_diel)

      Z-COORD              NAME TYPE THICKNESS   DIEL   WIDTH   SPACE  ENC

  --- 14.0900 ---------------------------------------------------------
  |               metal10_diel   D    2.0000   2.5000
  --- 12.0900 ---------------------------------------------------------
  |                    metal10   C    2.0000           0.8000  0.8000
  --- 10.0900 ---------------------------------------------------------
  |                metal9_diel   D    2.0000   2.5000
  --- 8.0900  ---------------------------------------------------------
  |                    metal9   C    2.0000           0.8000  0.8000
  --- 6.0900  ---------------------------------------------------------
  |                metal8_diel   D    0.8200   2.5000
  --- 5.2700  ---------------------------------------------------------
  |                    metal8   C    0.8000           0.4000  0.4000
  --- 4.4700  ---------------------------------------------------------
  |                metal7_diel   D    0.8200   2.5000
  --- 3.6500  ---------------------------------------------------------
  |                    metal7   C    0.8000           0.4000  0.4000
  --- 2.8500  ---------------------------------------------------------
  |                metal6_diel   D    0.2900   2.5000
  --- 2.5600  ---------------------------------------------------------
  |                    metal6   C    0.2800           0.1400  0.1400
  --- 2.2800  ---------------------------------------------------------
  |                metal5_diel   D    0.2900   2.5000
  --- 1.9900  ---------------------------------------------------------
  |                    metal5   C    0.2800           0.1400  0.1400
  --- 1.7100  ---------------------------------------------------------
  |                metal4_diel   D    0.2900   2.5000
  --- 1.4200  ---------------------------------------------------------
  |                    metal4   C    0.2800           0.1400  0.1400
  --- 1.1400  ---------------------------------------------------------
  |                metal3_diel   D    0.1200   2.5000
```

```
47   ─── 1.0200 ──────────────────────────────────────────────────────────
     │                  metal3   C    0.1400          0.0700  0.0700
     ─── 0.8800 ──────────────────────────────────────────────────────────
     │               metal2_diel   D    0.1200  2.5000
     ─── 0.7600 ──────────────────────────────────────────────────────────
52   │                  metal2   C    0.1400          0.0700  0.0700
     ─── 0.6200 ──────────────────────────────────────────────────────────
     │               metal1_diel   D    0.1200  2.5000
     ─── 0.5000 ──────────────────────────────────────────────────────────
     │                  metal1   C    0.1300          0.0650  0.6500
57   ─── 0.3700 ──────────────────────────────────────────────────────────
     │                poly_diel   D    0.0850  2.5000
     ─── 0.2850 ──────────────────────────────────────────────────────────
     │                    poly   C    0.0850          0.0500  0.0750
     ─── 0.2000 ──────────────────────────────────────────────────────────
62   │           field_base_diel   D    0.2000  3.9000
     ─── 0.0000 ──────────────────────────────────────────────────────────
     │                field_base   B    1.0000
     ─── −1.0000 ─────────────────────────────────────────────────────────
```

Oscillator extraction result

C

3D substrate resistance extraction results of 9-stages ring oscillator is shown in this part.

```
osc_final_ver2

* Generated by: xspice 2.51 06-May-2009
* Date: 25-Jun-09 12:33:09
* Path: /u/01/01/windhoos/home/goollia/intern/SPACE/osc_final
* Language: SPICE

* circuit osc_final_ver2 sens v8 v7 v6 v5 v4 v3 v2 v1 si VSS VDD
m1 v1 si VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+  nrs=0.583333 nrd=0.583333
m2 VDD si v1 9 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+  nrs=0.388889 nrd=0.388889
m3 v2 v1 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+  nrs=0.583333 nrd=0.583333
m4 VDD v1 v2 8 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+  nrs=0.388889 nrd=0.388889
m5 VDD v8 si 7 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+  nrs=0.388889 nrd=0.388889
m6 si v8 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+  nrs=0.583333 nrd=0.583333
m7 v3 v2 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+  nrs=0.583333 nrd=0.583333
m8 VDD v2 v3 6 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+  nrs=0.388889 nrd=0.388889
m9 VDD v7 v8 5 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+  nrs=0.388889 nrd=0.388889
m10 v8 v7 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+   nrs=0.583333 nrd=0.583333
m11 v4 v3 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+   nrs=0.583333 nrd=0.583333
m12 VDD v3 v4 4 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+   nrs=0.388889 nrd=0.388889
m13 VDD v6 v7 3 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+   nrs=0.388889 nrd=0.388889
m14 v7 v6 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+   nrs=0.583333 nrd=0.583333
m15 v5 v4 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+   nrs=0.583333 nrd=0.583333
m16 VDD v4 v5 2 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+   nrs=0.388889 nrd=0.388889
m17 VDD v5 v6 1 PMOS_VTL w=270n l=50n ad=28.35f as=28.35f pd=480n ps=480n
+   nrs=0.388889 nrd=0.388889
```

```
m18 v6 v5 VSS 11 NMOS_VTL w=180n l=50n ad=18.9f as=18.9f pd=390n ps=390n
+   nrs=0.583333 nrd=0.583333

c1 1 v5 56.31494e-18        c31 v6 11 105.7063e-18      c62 v4 v5 8.814578e
   -18
c2 1 v6 14.31477e-18        c32 v6 v7 8.814577e-18      c63 v4 26 3.52162e
   -18
c3 1 VDD 45.39855e-18       c33 si v8 8.814577e-18      c64 v4 27 26.96235e
   -18
c4 2 v5 14.4595e-18         c34 si 16 309.9881e-18      c65 v4 11 63.63161e
   -18
c5 2 VDD 44.96559e-18       c35 si 34 3.52162e-18       c66 v4 VSS 5.127501
   e-18
c6 2 v4 56.21653e-18        c36 si v1 8.814578e-18      c67 v5 12 3.52162e
   -18
c7 3 v6 51.63966e-18        c37 si 11 63.63161e-18      c68 v5 VSS 5.127501
   e-18
c8 3 v7 14.5476e-18         c38 si VSS 5.127501e-18     c69 v5 11 292.533e
   -18
c9 3 VDD 44.80689e-18       c39 v1 v2 8.814578e-18      c70 VSS 17 169.3489
   e-21
c10 4 v4 20.05665e-18       c40 v1 32 3.52162e-18       c71 VSS 21 169.3489
   e-21
c11 4 VDD 44.96559e-18      c41 v1 33 15.17522e-18      c72 VSS 18 169.3489
   e-21
c12 4 v3 56.50535e-18       c42 v1 11 63.63161e-18      c73 VSS 22 169.3489
   e-21
c13 5 v7 56.40059e-18       c43 v1 VSS 5.127501e-18     c74 VSS 19 169.3489
   e-21
c14 5 v8 19.84269e-18       c44 v8 v7 8.814577e-18      c75 VSS 23 169.3489
   e-21
c15 5 VDD 45.65328e-18      c45 v8 14 26.79796e-18      c76 VSS 20 169.3489
   e-21
c16 6 v3 19.84269e-18       c46 v8 15 3.52162e-18       c77 VSS 24 169.3489
   e-21
c17 6 VDD 44.96559e-18      c47 v8 VSS 5.127501e-18     c78 VSS 25 169.3489
   e-21
c18 6 v2 56.25394e-18       c49 v2 v3 8.814578e-18      c79 VSS 11 582.0669
   e-18
c19 7 v8 56.49371e-18       c50 v2 30 3.52162e-18
c20 7 si 19.62874e-18       c51 v2 31 27.06325e-18
c21 7 VDD 45.65328e-18      c52 v2 11 63.63161e-18
c22 8 v2 20.0063e-18        c53 v2 VSS 5.127501e-18
c23 8 VDD 44.96559e-18      c54 v7 13 3.52162e-18
c24 8 v1 56.40506e-18       c55 v7 VSS 5.127501e-18
c25 9 v1 19.9245e-18        c56 v7 11 93.24863e-18
c26 9 si 56.1838e-18        c57 v3 v4 8.814578e-18
c27 9 VDD 47.50738e-18      c58 v3 28 3.52162e-18
c28 VDD 10 1.402471f        c59 v3 29 24.34772e-18
c29 v6 v5 8.814577e-18      c60 v3 11 63.63161e-18
c30 v6 VSS 5.127501e-18     c61 v3 VSS 5.127501e-18
```

```
    r1 10 35 1.0476meg          r340 31 34 297.8961meg
 80 r2 10 17 18.15534meg        r341 31 SUBSTR 2.223355meg
    r3 10 12 10.16353meg        r342 32 35 2.114185g
    r4 10 26 9.872117meg        r343 32 33 16.75738meg
    r5 10 21 21.64214meg        r344 32 34 683.9997meg
    r6 10 27 1.247291meg        r345 32 SUBSTR 17.38612meg
 85 r7 10 18 25.16364meg        r346 33 35 422.6875meg
    r8 10 28 10.41829meg        r347 33 34 47.12924meg
    r9 10 22 24.43952meg        r348 33 SUBSTR 3.23578meg
    ... ... ...                 r349 34 35 2.297474g
    r324 29 35 180.8385meg      r350 34 SUBSTR 15.67699meg
 90 r325 29 30 41.06297meg      c80 sens 35 32.44835e-18
    r326 29 31 15.38114meg      r351 35 SUBSTR 490.2302k
    r327 29 32 423.9719meg      r335 30 34 4.413218g
    r328 29 33 123.0065meg      r336 30 SUBSTR 17.98669meg
    r329 29 34 1.131787g        r337 31 35 230.6317meg
 95 r330 29 SUBSTR 2.339295meg  r338 31 32 41.67318meg
    r331 30 35 1.674024g        r339 31 33 19.62957meg
    r332 30 31 15.08776meg
    r333 30 32 1.103542g
    r334 30 33 382.1902meg
100
    * end osc_final_ver2
```

FreePDK45 technology files

D

FreePDK45nm technology files of SPACE are list in this part, including Element definition source file—space.def.s; GDSII maps—bmlist; Mask definition—maskdata.

D.1 Element definition source file

```
# rcsid = "$Id: space.def.s,v 1.1 2009/04/22 13:37:05 simon Exp $"
## This file was generated by SPOCK on Wed Apr 22 2009 at 14:16:09.
#####
# FreePDK45 - FreePDK 45nm
#
# Masks:
# nwell - n-type well implant
# pwell - p-type well implant
# nimplant  - n-type s/d doping
# pimplant  - p-type s/d doping
# active    - active area
# poly  - poly-silicon
# contact   - contact
# metal1    - metal1
# via1  - via1
# metal2    - metal2
# via2  - via2
# metal3    - metal3
# via3  - via3
# metal4    - metal4
# via4  - via4
# metal5    - metal5
# via5  - via5
# metal6    - metal6
# via6  - via6
# metal7    - metal7
# via7  - via7
# metal8    - metal8
# via8  - via8
# metal9    - metal9
# via9  - via9
# metal10   - metal10
# sblock    - salicide block
# thkox - thick-oxide implant
# vthl  - low threshold implant
# vthg  - general threshold implant
# vthh  - high threshold implant
# nodrc - nodrc
```

```
# bb    - bbox
#
# See also the maskdata file.
#####

unit resistance    1    # ohm
unit c_resistance  1e-12    # ohm m^2
unit s_resistance  1    # ohm ...
unit capacitance   1e-15    # F
unit a_capacitance 1e-6 # F/m^2
unit e_capacitance 1e-12    # F/m
unit distance      1e-6 # m
unit resize        1e-6 # m
unit vdimension    1e-6 # m
unit shape         1e-6 # m

#maxkeys 12

colors:
    nwell   @6901aa
    pwell   @ffff00
    nimplant    @aa00ff
    pimplant    @55ff00
    active  @008700
    poly    @ff0000
    contact @ffffff
    metal1  @0000ff
    via1    @e6e6e6
    metal2  @55ffff
    via2    @c7c7c7
    metal3  @4ab4ff
    via3    @a7a7a7
    metal4  @55aaff
    via4    @8d8d8d
    metal5  @55aaff
    via5    @7a7a7a
    metal6  @55aaff
    via6    @797979
    metal7  @55aaff
    via7    @717171
    metal8  @55aaff
    via8    @636363
    metal9  @55aaff
    via9    @585858
    metal10 @55aaff
    sblock  @ffaa7f
    thkox   @ffaaff
    vthl    @aa0000
    vthg    @aa5500
    vthh    @aaaa00
    nodrc   glass
    bb  glass
    @sub    @aa5500
```

```
conductors :
#   name:condition_list:mask:sheet_resistivity:carrier_type
    cond_nw:( nwell ):nwell:933:n       # n-well area
    cond_an:( !nwell nimplant active !poly ):active:0:n     # active n+
       area
    cond_ap:( nwell pimplant active !poly ):active:0:p      # active p+
       area
    cond_ps:( poly ):poly:7.8:m     # poly-silicon
    cond_m1:( metal1 ):metal1:0.38:m        # 1st metal
    cond_m2:( metal2 ):metal2:0.25:m        # 2nd metal
    cond_m3:( metal3 ):metal3:0.25:m        # 3rd metal
    cond_m4:( metal4 ):metal4:0.21:m        # 4th metal
    cond_m5:( metal5 ):metal5:0.21:m        # 5th metal
    cond_m6:( metal6 ):metal6:0.21:m        # 6th metal
    cond_m7:( metal7 ):metal7:0.075:m       # 7th metal
    cond_m8:( metal8 ):metal8:0.075:m       # 8th metal
    cond_m9:( metal9 ):metal9:0.03:m        # 9th metal
    cond_m10:( metal10 ):metal10:0.03:m     # 10th metal

fets:
#   name:condition_list:mask_g mask_ds [dsarea] [:mask_b]
    nenh:( !nwell pwell nimplant active poly ):poly active ( !nwell
       nimplant active !poly ):@sub
    penh:( nwell !pwell pimplant active poly ):poly active ( nwell
       pimplant active !poly ):nwell

bjts:
#   name:condition_list:type:mask_em mask_ba mask_co [:mask_b]

connects:
#   name:condition_list:type:mask1 mask2

contacts :
#   name:condition_list:type:mask1 mask2:resitivity
    cont_su:( !nwell pimplant active contact metal1 ):@sub metal1:0     #
        substrate to m1
    cont_nw:( nwell nimplant active contact metal1 ):nwell metal1:0     #
        nwell to m1
    cont_an:( !nwell nimplant active !poly contact metal1 ):active
       metal1:0     # active n+ to m1
    cont_ap:( nwell pimplant active !poly contact metal1 ):active
       metal1:0      # active p+ to m1
    cont_ps:( poly contact metal1 ):poly metal1:0.0338      # poly to m1
    cont_m1:( metal1 via1 metal2 ):metal1 metal2:0.02535    # m1 to m2
    cont_m2:( metal2 via2 metal3 ):metal2 metal3:0.0245     # m2 to m3
    cont_m3:( metal3 via3 metal4 ):metal3 metal4:0.0245     # m3 to m4
    cont_m4:( metal4 via4 metal5 ):metal4 metal5:0.0588     # m4 to m5
    cont_m5:( metal5 via5 metal6 ):metal5 metal6:0.0588     # m5 to m6
    cont_m6:( metal6 via6 metal7 ):metal6 metal7:0.0588     # m6 to m7
    cont_m7:( metal7 via7 metal8 ):metal7 metal8:0.16       # m7 to m8
    cont_m8:( metal8 via8 metal9 ):metal8 metal9:0.16       # m8 to m9
    cont_m9:( metal9 via9 metal10 ):metal9 metal10:0.32     # m9 to m10
```

```
vdimensions:
#   name:condition_list:mask:bottom thickness [spacing]
    vdim_an:( !nwell nimplant active !poly ):active:0.196 0
    vdim_ap:( nwell pimplant active !poly ):active:0.196 0
    vdim_ps:( poly ):poly:0.200 0.085 0.075
    vdim_m1:( metal1 ):metal1:0.370 0.130
    vdim_m2:( metal2 ):metal2:0.620 0.140
    vdim_m3:( metal3 ):metal3:0.880 0.140
    vdim_m4:( metal4 ):metal4:1.140 0.280
    vdim_m5:( metal5 ):metal5:1.710 0.280
    vdim_m6:( metal6 ):metal6:2.280 0.280
    vdim_m7:( metal7 ):metal7:2.850 0.800
    vdim_m8:( metal8 ):metal8:4.470 0.800
    vdim_m9:( metal9 ):metal9:6.090 2.000 0.800
    vdim_m10:( metal10 ):metal10:10.090 2.000 0.800

eshapes:
#   name:condition_list:mask:dxb dxt

capacitances:
# lateral capacitances
  lcap_poly_metal1 : !poly -poly  =metal1 !metal1 : -poly  =metal1 : 0
      .001072

cshapes:
#   name:condition_list:mask:xb1 xt1 xb2 xt2

dielectrics:
#   name permittivity bottom
    field_base 3.9 0    # field_base diel
    stddiel    2.5 0.200    # standard diel
    air        1  14.090    # no diel

sublayers:
#   name conductivity top
    substrate 6.7 0

subcaplayers:
#   name permittivity top

selfsubres:
#   area perim value rest

coupsubres:
#   area1 area2 dist. value decr.

#EOF
```

D.2 GDSII maps

```
# rcsid = "$Id: bmlist.gds,v 1.3 2009/04/23 17:39:17 simon Exp $"
## This file was generated by SPOCK on Thu Apr 23 2009 at 19:31:22.
#####
# FreePDK45 - FreePDK 45nm
nwell        3
pwell        2
nimplant     4
pimplant     5
active       1
active:term  1
poly         9
poly:term    9
contact      10
metal1       11
metal1:term  11
via1         12
metal2       13
metal2:term  13
via2         14
metal3       15
metal3:term  15
via3         16
metal4       17
metal4:term  17
via4         18
metal5       19
metal5:term  19
via5         20
metal6       21
metal6:term  21
via6         22
metal7       23
metal7:term  23
via7         24
metal8       25
metal8:term  25
via8         26
metal9       27
metal9:term  27
via9         28
metal10      29
metal10:term     29
vthg         6
vthh         7
thkox        8
nodrc        80
bb       63
bb2      183
```

D.3 Mask definition

```
# rcsid = "$Id: maskdata,v 1.2 2009/04/22 15:44:14 simon Exp $"
```

```
## This file was generated by SPOCK on Wed Apr 22 2009 at 17:32:53.
#------------------------------------------------------------------
#                M A S K D A T A   I N F O
#
#  Layer fields (2):     field  1: layer name
#                        field  2: layer type
#                                  type = 0: normal layer
#                                  type = 1: interconnect layer
#                                            (terminals/labels may be
#                                             defined for this layer).
#                                  type = 2: symbolic layer
#
#  Pattern-Generate (2): field  3: job number
#  (Only used by         field  4: mask type
#   PG-tape programs               type = 0: negative
#                                  type = 1: positive
#
#  CMask Terminals (2):  field  5: color number
#  (Obsolete)            field  6: fill style
#
#  (Sea)Dali (2):        field  7: color number
#                                  0=black, 1=red, 2=green,
#                                  3=yellow, 4=blue, 5=violet,
#                                  6=aqua, 7=white
#                        field  8: fill style
#                                  0=hashed, 1=solid, 2=hollow
#                                  3,4,5 = 12,25,50% hash+outline
#                                  6,7,8 = idem, no outline
#
#  Plotter (2):          field  9: pen number
#  (Obsolete)                      1=black, 2=red, 3=yellow,
#                                  4=green, 5=brown, 6=violet,
#                                  7=blue, 8=aqua
#                        field 10: fill style
#
"FreePDK45" "FreePDK 45nm"
# name   type     PGtape  CMask   Dali    Plot    comment
nwell    0    0 1 0 1 5 2 1 0 "n-type well implant"
pwell    0    0 1 0 1 3 2 1 0 "p-type well implant"
nimplant 0    0 1 0 1 5 4 1 0 "n-type s/d doping"
pimplant 0    0 1 0 1 3 4 1 0 "p-type s/d doping"
active   1    0 1 0 1 2 2 1 0 "active area"
poly     1    0 1 0 1 1 1 1 0 "poly-silicon"
contact  0    0 1 0 1 7 1 1 0 "contact"
metal1   1    0 1 0 1 4 1 1 0 "metal1"
via1     0    0 1 0 1 7 5 1 0 "via1"
metal2   1    0 1 0 1 6 5 1 0 "metal2"
via2     0    0 1 0 1 7 5 1 0 "via2"
metal3   1    0 1 0 1 6 6 1 0 "metal3"
via3     0    0 1 0 1 7 5 1 0 "via3"
metal4   1    0 1 0 1 4 3 1 0 "metal4"
via4     0    0 1 0 1 7 5 1 0 "via4"
metal5   1    0 1 0 1 4 3 1 0 "metal5"
```

```
    via5     0    0 1 0 1 7 5 1 0 "via5"
    metal6   1    0 1 0 1 4 3 1 0 "metal6"
57  via6     0    0 1 0 1 7 5 1 0 "via6"
    metal7   1    0 1 0 1 4 3 1 0 "metal7"
    via7     0    0 1 0 1 7 5 1 0 "via7"
    metal8   1    0 1 0 1 4 3 1 0 "metal8"
    via8     0    0 1 0 1 7 5 1 0 "via8"
62  metal9   1    0 1 0 1 4 3 1 0 "metal9"
    via9     0    0 1 0 1 7 5 1 0 "via9"
    metal10  1    0 1 0 1 4 3 1 0 "metal10"
    sblock   0    0 1 0 1 7 0 1 0 "salicide block"
    thkox    0    0 1 0 1 5 3 1 0 "thick-oxide implant"
67  vthl     0    0 1 0 1 1 3 1 0 "low threshold implant"
    vthg     0    0 1 0 1 1 3 1 0 "general threshold implant"
    vthh     0    0 1 0 1 7 3 1 0 "high threshold implant"
    nodrc    2    0 1 0 1 7 2 1 0 "nodrc"
    bb   2   0 1 0 1 7 2 1 0 "bbox"
72  bb2  2   0 1 0 1 7 2 1 0 "bbox2"
```

45nm technology Synthesis and Place & Route flow

E

This part indicate the 45nm design Synthesis flow and Place&Route flow. Two EDA tools are used: Design Compiler (Synopsys), SoC Encounter (Cadence). These tools help us to finish design synthesis and floorplan, place&route processes. Section E.1 indicates the Design Compile flow script. Section E.2 shows the SoC Encounter P&R flow script.

E.1 45nm technology Synthesis flow

```
#########################################################
### Compile Script for Synopsys                       ##
##                                                    ##
### dc_shell-t -f compile_dc_all.tcl                  ##
### Nangate FreePDK 45nm                              ##
#########################################################

remove_design -all

set LANGUAGE vhdl               ;# 'vhdl' or 'verilog'
set STAGE  "_lin"
set STAGE1 "_lin1"

#set the target library and search path
set Nangate_FREEPDK [format "%s%s"  [getenv "PDK_DIR"] "#Design8051#lib"]
set search_path [concat  $search_path $Nangate_FREEPDK ]
set alib_library_analysis_path $Nangate_FREEPDK
set link_library [set target_library [concat  [list
    NangateOpenCellLibrary_PDKv1_2_v2008_10.db ] [list dw_foundation.sldb
    ]]]
set target_library "NangateOpenCellLibrary_PDKv1_2_v2008_10.db"

# Setup the parameters for Compiling
set fsm_export_formality_state_info true
set fsm_auto_inferring true
set find_ignore_case true
set suppress_errors "VHDL -2285 OPT -150 TIM -111 TIM -112"
set high_fanout_net_threshold 0
set bus_naming_style %s\[%d\]

# Set work library
define_design_lib Work -path LIB#SNPS#Work  ;# define work library
# Define language file extension
if { $LANGUAGE == "vhdl" } then { set ext .vhd } else { set ext .v }

```

```
#############################################################
####  Define the top_level design and analyze your VHDL code
####          the topest in the last
#############################################################
set TOP_LEVEL I8051_ALL         ;# name of core instance file

# Start RTL analysis,
echo "Analyze started."
sh mkdir -p log

set res [ analyze -f $LANGUAGE -lib Work  [ list i8051_dec.vhd
i8051_alu.vhd i8051_ctr.vhd  i8051_rom.vhd
i8051_lib.vhd i8051_ram.vhd  i8051_all.vhd ] ]

# Check for errors
if { $res == 1 } then {
  echo "Analyze successful."
} else {
  echo "Analyze generated at least one error."
  set strings [ exec grep "Error:" log/analyze$STAGE.log ]
  echo "Error mesage(s) from Analyze command:"
  puts $strings
  echo "Terminating due to error(s)..."
  exit }

echo "Elaborate started"
redirect log/elaborate$STAGE.log { set res [ elaborate $TOP_LEVEL -lib
   Work -update ] }

# Check for errors
if { $res == 1 } then {
  echo "Elaborate successful."
} else {
  echo "Elaborate generated at least one error."
  set strings [ exec grep "Error:" log/elaborate$STAGE.log ]
  echo "Error mesage(s) from Elaborate command:"
  puts $strings
  echo "Terminating due to error(s)..."
  exit }

# Check for combinatorial loops
redirect log/loops$STAGE.log { report_timing -loops }

# Check for latches
redirect log/latches$STAGE.log { all_registers -level_sensitive }

current_design $TOP_LEVEL
set fsm_auto_inferring true
set_fsm_encoding_style binary

# Resolve design references
redirect log/link$STAGE.log { link }
```

```
# check the design for errors such as missing module definisions
check_design > log/check_design$STAGE.log

#current_design $TOP_LEVEL
# Setup the parameters for Compiling
set fsm_export_formality_state_info true
set fsm_auto_inferring true
set find_ignore_case true
set suppress_errors "VHDL -2285 OPT -150 TIM -111 TIM -112"
set high_fanout_net_threshold 0
#set bus_naming_style %s[%d]-->output name use []
set bus_naming_style %s\[%d\]
set dw_prefer_mc_inside true
#set_ultra_optimization true
set_ultra_optimization -f
set fsm_auto_inferring true
set_fsm_encoding_style binary
set gen_show_created_symbols true
set enable_recfinaly_removed_arcs true

set_fix_multiple_port_nets -all -buffer_constants
set_fix_multiple_port_nets -feedthroughs

#################################################################
# Define environment
#################################################################
set OPERATING_COND typical
set_operating_conditions $OPERATING_COND


####Define constraints
echo "Applying constraints"

source constraint_lin.tcl   ;# Constraint file to this design
redirect log/constraints0$STAGE.rpt { report_constraint -nosplit }


# Map design to gates
echo "Compile started."
redirect log/uniquify$STAGE.log { uniquify }
redirect log/compile1$STAGE.log { set res [ compile -map_effort medium
    -area_effort medium ] }
# Check for errors
if { $res == 1 } then {
  echo "Compile successful."
} else {
  echo "Compile generated at least one error."
  set strings [ exec grep "Error:" log/compile1$STAGE.log ]
  echo "Error mesage(s) from Compile command:"
  puts $strings
  echo "Terminating due to error(s)..."
  exit }
```

```
set vhdlout_dont_create_dummy_nets true
remove_unconnected_ports [get_cells -hier #]
remove_unconnected_ports -blast_buses [get_cells -hier  # ]
sh mkdir -p netlist
# Save mapped design
echo "Writing mapped design."
write -hierarchy -format ddc -output netlist/$TOP_LEVEL$STAGE.mapped1.ddc

# Save netlist
echo "Writing netlist."
write -format $LANGUAGE -hierarchy -output netlist/
   $TOP_LEVEL$STAGE.netlist1$ext
# For reference verilog
set verilogout_no_tri true
set verilogout_single_bit false
set verilogout_show_unconnected_pins true
change_names -rule verilog -hierarchy
write -format verilog -hierarchy -output netlist/
   $TOP_LEVEL$STAGE.netlist1.v
write_sdf netlist/$TOP_LEVEL$STAGE.sdf
write_sdc netlist/$TOP_LEVEL$STAGE.sdc

# Generate reports
echo "Generating reports."
redirect log/area1$STAGE.rpt { report_area }
redirect log/timing1$STAGE.rpt { report_timing -path full -max_paths 1
   -nosplit }
redirect log/timingwst1$STAGE.rpt { report_timing -nworst 10}
redirect log/timingloops1$STAGE.rpt { report_timing -loops}
redirect log/hierarchy1$STAGE.rpt { report_hierarchy}
redirect log/constraints1$STAGE.rpt { report_constraint}
redirect log/resources1$STAGE.rpt { report_resources -nosplit -hierarchy
   }
redirect log/references1$STAGE.rpt { report_reference -nosplit }
redirect log/cell1$STAGE.rpt { report_cell -nosplit }
redirect log/fsm1$STAGE.rpt { report_fsm -nosplit }
redirect log/design1$STAGE.rpt { report_design -nosplit }
redirect log/power1$STAGE.rpt { report_power -analysis_effort high
   -nosplit}

# Flatten hierarchy and optimize
echo "Optimize started."
redirect log/compile2$STAGE.log { set res [ compile_ultra -incremental] }

# Check for errors
if { $res == 1 } then {
  echo "Optimize successful."
} else {
  echo "Optimize generated at least one error."
  set strings [ exec grep "Error:" log/compile2$STAGE.log ]
  echo "Error mesage(s) from Optimize command:"
  puts $strings
  echo "Terminating due to error(s)..."
```

```
    exit }

# Save mapped design
echo "Writing mapped design."
write -hierarchy -format ddc -output netlist /
    $TOP_LEVEL$STAGE.mapped2.ddc
# Save netlist
echo "Writing netlist."

write -format $LANGUAGE -hierarchy -output netlist/
    $TOP_LEVEL$STAGE.netlist2$ext

# For reference verilog
set verilogout_no_tri true
set verilogout_single_bit false
set verilogout_show_unconnected_pins true
change_names -rule verilog -hierarchy
write -format verilog -hierarchy -output netlist/
    $TOP_LEVEL$STAGE.netlist2.v

write_sdf netlist/$TOP_LEVEL$STAGE1.sdf
write_sdc netlist/$TOP_LEVEL$STAGE1.sdc

# Generate reports
echo "Generating reports."
redirect log/area2$STAGE.rpt { report_area }
redirect log/timing2$STAGE.rpt { report_timing -path full -max_paths 1
    -nosplit }
redirect log/timingwst2$STAGE.rpt { report_timing -nworst 10}
redirect log/timingloops2$STAGE.rpt { report_timing -loops}
redirect log/hierarchy2$STAGE.rpt { report_hierarchy}
redirect log/constraints2$STAGE.rpt { report_constraint}
redirect log/resources2$STAGE.rpt { report_resources -nosplit -hierarchy
    }
redirect log/references2$STAGE.rpt { report_reference -nosplit }
redirect log/cell2$STAGE.rpt { report_cell -nosplit }
redirect log/fsm2$STAGE.rpt { report_fsm -nosplit }
redirect log/design2$STAGE.rpt { report_design -nosplit }
redirect log/power2$STAGE.rpt { report_power -analysis_effort high
    -nosplit}
sh date
echo "Done."
exit
```

F

```
##############################################
##        Define constraints
##############################################

set CLK_PERIOD 5; # 5ns
set CLK_UNCERTAINTY  0.2; # 200ps

set DFF_CKQ  0.2; # Clk to Q in technology time units
```

```
set DFF_SETUP  0.01; # Setup time in technology time units

set LIB_DFF_D DFFX1/Q
set DFF_CELL DFFX1
set LIBNAME NangateOpenCell45nm

set CLK  "clk"
set RST  "rst"

create_clock $CLK -period $CLK_PERIOD
set_clock_uncertainty $CLK_UNCERTAINTY [all_clocks]
set_dont_touch_network [all_clocks]

remove_driving_cell $RST
set_drive 0 $RST
set_dont_touch_network $RST
set_ideal_network [get_ports $RST] ;# Before Layout

set_output_delay -max $DFF_SETUP -clock $CLK [all_outputs]
set all_inputs_worst_clk [remove_from_collection
       [remove_from_collection [all_inputs]
          [get_ports $CLK]] [get_ports $RST]]
set_input_delay -clock $CLK -max $DFF_CKQ $all_inputs_worst_clk

set_load [expr [load_of $LIBNAME/BUFX1/O] * 4] [all_outputs]
set_load 0.35 [all_outputs]

set_max_area 0
```

E.2 45nm technology Place&Route flow

```
####################################
# Run the design through Encounter
####################################

# Create Initial Floorplan
floorplan -r 1.0 0.6 20 20 20 20

# Create Power structures
addRing -spacing_bottom 5 -width_left 5 -width_bottom 5 -width_top 5 -
    spacing_top 5 -layer_bottom metal9 -width_right 5 -around core -center
     1 -layer_top metal9 -spacing_right 5 -spacing_left 5 -layer_right
    metal10 -layer_left metal10 -nets { VSS VDD }

####################################################
# Define global Power nets - make global connections
####################################################
clearGlobalNets
globalNetConnect VDD -type pgpin -pin VDD -inst * -module {} -verbose
globalNetConnect VSS -type pgpin -pin VSS -inst * -module {} -verbose
globalNetConnect VDD -type tiehi -module {}
globalNetConnect VSS -type tielo -module {}
```

```
applyGlobalNets

# Route power nets
sroute -noBlockPins -noPadRings -noPadPins
# Verify the PG nets connectivity
verifyConnectivity -nets {VDD VSS} -type special -error 1000 -warning 50
violationBrowser -all -no_display_false

# Placement of the Std-Cells
setPlaceMode -timingDriven 1 -modulePlan 1 -doCongOpt true -congEffort
    medium
placeDesign -prePlaceOpt
setDrawView place
refinePlace -checkRoute 0 -honorSoftBlockage 1 -preserveRouting 0 -
    rmAffectedRouting 0 -swapEEQ 0 -checkPinLayerForAccess 1
verifyGeometry -allowSameCellViols -allowRoutingBlkgPinOverlap -
    allowRoutingCellBlkgOverlap -report verifyGeometry.log
violationBrowser -all -no_display_false
placeDesign -incremental
verifyGeometry -allowSameCellViols -allowRoutingBlkgPinOverlap -
    allowRoutingCellBlkgOverlap -report verifyGeometry.log
violationBrowser -all -no_display_false
checkDesign -place -outdir checkDesign/afterplacement
#######################################################
## Perform trial route and get initial timing results
#######################################################
trialroute

# Time Analyse after Placement and before CTS, here the Trail-route done
    automaticly
timeDesign -preCTS -idealClock -pathReports -drvReports -slackReports -
    numPaths 50 -prefix FIR5_withPAD_preCTS -outDir timingReports/
    preCTS_setup
#######################################################
# opt the timing of the design before CTS
#######################################################
setOptMode -effort high -fixFanoutLoad true
setOptMode -setupTargetSlack 0.6
getOptMode
optDesign -preCTS -outDir timingReports/preCTS_setupOpt0
setOptMode -setupTargetSlack 0.9
getOptMode
optDesign -preCTS -outDir timingReports/preCTS_setupOpt1
timeDesign -preCTS -outDir timingReports/preCTS_postsetupOpt0drc

getOptMode
clearClockDomains

setClockDomains -fromType input -toType output
optDesign -preCTS -incr -outDir timingReports/preCTS_setupOpt1_in2out
#Increment
clearClockDomains
setClockDomains -fromType register -toType output
```

```
optDesign -preCTS -incr -outDir timingReports/preCTS_incrOpt_reg2reg
#Decrease the DRC errors  fixDRCViolation
clearClockDomains
setClockDomains -all
optDesign -preCTS -drv
timeDesign -preCTS -outDir timingReports/preCTS_OptafterDRC

############################################################
#CTS
############################################################
timeDesign -postCTS -outDir timingReports/postCTS_preSetupOpt
#Analyse the hold time
timeDesign -postCTS -hold -outDir timingReports/preNano_postSetupOpt_hold
setOptMode -setupTargetSlack 0.2
getOptMode
optDesign -postCTS -outDir timingReports/postCTS_setupOpt0

setOptMode -setupTargetSlack 0.2 -holdTargetSlack 0.5
getOptMode
optDesign -postCTS -hold -outDir timingReports/postCTS_holdOpt
timeDesign -postCTS -outDir timingReports/postCTS_OptafterDRC

clearClockDomains
setClockDomains -fromType input -toType output
optDesign -postCTS -incr -outDir timingReports/preCTS_incrOpt_reg2reg
############################################################
# Route design (Nanoroute)
############################################################
if { 1 } {
setNanoRouteMode -quiet -timingEngine CTE
setNanoRouteMode -quiet -routeWithTimingDriven true
setNanoRouteMode -quiet -routeTdrEffort 0
setNanoRouteMode -quiet -routeBottomRoutingLayer 1
setNanoRouteMode -quiet -routeTopRoutingLayer 6
}
setNanoRouteMode -quiet route_selected_net_only false
setNanoRouteMode -quiet routeWithTimingDriven true
setNanoRouteMode -quiet routeTdrEffort 1
setNanoRouteMode -quiet drouteFixAntenna true
setNanoRouteMode -quiet routeInsertAntennaDiode true
setNanoRouteMode -quiet routeAntennaCellName "ANT"
setNanoRouteMode -quiet routeWithSiDriven true
setNanoRouteMode -quiet routeSiLengthLimit 200
setNanoRouteMode -quiet routeSiEffort normal
setNanoRouteMode -quiet drouteUseMinSpacingForBlockage false
#setNanoRouteMode -envNumberFailLimit 11
globalDetailRoute

setDrawView place
Redraw
timeDesign -postRoute -outDir timingReports/postNano_Setup
timeDesign -postRoute -hold -outDir timingReports/postNano_hold
setOptMode -setupTargetSlack 0.5
```

```
optDesign -postRoute -outDir timingReports/afterroute_setupOpt
setOptMode -setupTargetSlack 0.2 -holdTargetSlack 0.5
optDesign -postRoute -hold -outDir timingReports/afterroute_holdOpt

verifyGeometry -allowSameCellViols -allowRoutingBlkgPinOverlap -
    allowRoutingCellBlkgOverlap -report verifyGeometry.log
violationBrowser -all -no_display_false
#Fix the Geometry Violations by hand and then reRun globalDetailRoute
globalDetailRoute

################################################
# Add Core Filler
################################################
addFiller -cell FILLCELL_X8 FILLCELL_X4 FILLCELL_X32 FILLCELL_X2
    FILLCELL_X16 FILLCELL_X1 -prefix FILLER -honorPrerouteAsObs true -
    doDRC true
fillNotch -report notch.log
verifyGeometry -allowSameCellViols -allowRoutingBlkgPinOverlap -
    allowRoutingCellBlkgOverlap -report verifyGeometry.log
violationBrowser -all -no_display_false
timeDesign -postRoute -outDir timingReports/postNano_Setup_afterfix
timeDesign -postRoute -hold -outDir timingReports/postNano_hold_afterfix
verifyConnectivity -type regular -error 1000 -warning 50 -report
    verifyConnectivity_regu.log

################################################
##      Connect all new cells to VDD/GND
################################################
globalNetConnect VDD -type tiehi -module {}
globalNetConnect VSS -type tielo -module {}

# Run global Routing
globalDetailRoute

################################################
## Final Verification
################################################
verifyGeometry -allowSameCellViols -allowRoutingBlkgPinOverlap -
    allowRoutingCellBlkgOverlap -report verifyGeometry.log
violationBrowser -all -no_display_false
timeDesign -postRoute -outDir timingReports/postNano_Setup_afterfix
timeDesign -postRoute -hold -outDir timingReports/postNano_hold_afterfix

verifyConnectivity -type regular -error 1000 -warning 50 -report
    verifyConnectivity_regu.log

################################################
# Output GDSII and netlist
################################################
streamOut final.gds2 -mapFile gds2_encounter.map -stripes 1 -units 1000 -
    mode ALL
saveNetlist -lvs final.v
```

Bibliography

[1] D.J. Frank, R. Puri, and D. Toma, "Design and cad challenges in 45nm cmos and beyond," in *Computer-Aided Design, 2006. ICCAD '06. IEEE/ACM International Conference on*, Nov. 2006, pp. 329–333.

[2] Zhuoxiang Ren, H. Hegazy, and N. Kurt-Karsilayan, "Characterization of dynamic substrate macro-model in mixed signal ic systems using 3-d finite element method," *Magnetics, IEEE Transactions on*, vol. 44, no. 6, pp. 1466–1469, June 2008.

[3] Yu Cao and C. McAndrew, "Mosfet modeling for 45nm and beyond," in *Computer-Aided Design, 2007. ICCAD 2007. IEEE/ACM International Conference on*, Nov. 2007, pp. 638–643.

[4] Predictive Technology Model, "http://www.eas.asu.edu/∼ptm/," July. 2009.

[5] Wei Zhao and Yu Cao, "New generation of predictive technology model for sub-45 nm early design exploration," *Electron Devices, IEEE Transactions on*, vol. 53, no. 11, pp. 2816–2823, Nov. 2006.

[6] N.P. van der Meijs, A.J. van Genderen, F. Beeftink, and P.J.H. Eleas, *SPACE USER'S MANUAL*, 2005.

[7] Nangate-Design Optimization Company, "http://www.nangate.com/index.php," July. 2009.

[8] FreePDK-NCSU EDA Wiki, "http://www.eda.ncsu.edu/wiki/freepdk," July. 2009.

[9] N.P. van der Meijs and A.J. van Genderen, *SPACE TUTORIAL*, 2003.

[10] OptEM, *SPOCK USER'S MANUAL*, 2002.

[11] N.P. van der Meijs and A.J. van Genderen, *Space 3D Capacitance Extraction USER'S MANUAL*, 2008.

[12] NCSU FreePDK45, "http://www.eda.ncsu.edu/wiki/freepdk45:contents," July. 2009.

[13] Synthesizable VHDL Model of 8051, "http://www.cs.ucr.edu/∼dalton/i8051/i8051syn/," July. 2009.

[14] R. Murgai, S.M. Reddy, T. Miyoshi, T. Horie, and M.B. Tahoori, "Sensitivity-based modeling and methodology for full-chip substrate noise analysis," in *Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings*, Feb. 2004, vol. 1, pp. 610–615 Vol.1.

[15] E. Schrik, "A combined BEM/FEM method for IC substrate modeling," Ph.D. dissertation, TU Delft, Dept. EEMCS, September 2006.

[16] E. Schrik and N.P. van der Meijs, "Combined bem/fem substrate resistance modeling," in *Proc. 39th Design Automation Conference*, New Orleans, LA, June 2002, pp. 771–776.

[17] R. Gharpurey and E. Charbon, "Substrate coupling: modeling, simulation and design perspectives," in *Quality Electronic Design, 2004. Proceedings. 5th International Symposium on*, 2004, pp. 283–290.

[18] Ring Oscillator, "http://en.wikipedia.org/wiki/ring_oscillator," July. 2009.

[19] Layout Versus Schematic, "http://en.wikipedia.org/wiki/layout-versus-schematic," July. 2009.

[20] Yuhua Cheng and Chenming Hu, *MOSFET Modeling and Bsim3 User's Guide*, Kluwer Academic Publishers, Norwell, MA, USA, 1999.

[21] AVANT, *Star-Hspice Manual*, July 1998.