2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE 10 YEAR ANNIVERSARY

Low-Power Validation Framework for Standard Cell Library Including Front-End and Back-End Implementation

J. Lee, H. Bak, S. Do, T. Yoo,

Foundry Division, Samsung Electronics,

Hwaseong-si, Republic of Korea (jinhoo.lee@samsung.com)

Gowrishankar Srinivasan, Vishw Mitra Singh Bhadouria,

Samsung Semiconductor India Research (SSIR), Bengaluru, India

***Abstract*—In a foundry ecosystem, libraries are key components because customers proceed with chip design under the assumption that the library is completed. In the worst-case scenario, designers restart the entire design process if there is an issue with the library. As chip sizes have reduced in recent years, the significance of low-power design has come to light. As a result, libraries are enhanced to support low-power design. To assure the low-power support of libraries, the power-intent strategy and the verification methodology through the establishment of a multi-voltage design platform were previously presented for standard cell library verification. However, there was no verification process for strategies and platforms to validate the accuracy of the results. In this paper, we introduce a more complete verification environment with a verification process for strategies and platforms that were not available in the existing framework. Our study extended verification from simulation-based verification to design verification with more diverse scenarios, and covered not only front-end flow but also back-end flow to implement a more complete methodology.**

***Keywords*—standard cell; UPF; low-power verification**

I. Introduction

In the foundry ecosystem, library is a key object. Various customers design their IPs or chips with the library provided from the foundry. Among them, standard cells account for more than 50 percent of the total chip area and are used the most in the chip. As the chip size reduces, the importance of low-power design increases. Therefore, designers now build their low-power design strategies as Unified Power Format (UPF). The importance of library power technology verification has also begun to emerge because UPF can be used accurately only when it matches the library information.

Validation framework for library verification is important. If customers use unverified library, they proceed with the design without knowing that the required cell is omitted or inaccurate information is used. In the worst case, this can lead to redesign of customers, which affects the customer's trust for foundry. Most chip designers do not concern about this verification because they are designing assuming that the library is fully prepared. To this end, Samsung Foundry is making great efforts to provide a fully verified library for right design of customers.

In this paper, we introduce a more upgraded library validation framework. Using datasheet parser, we created an input file including cell information and truth table by using the necessary information in the datasheet, and checked whether there is any issue in the flow through low-power rule check, low-power equivalence check, and low-power simulation for library verification. In addition, we are able to automatically proceed with the back-end flow regardless of the process node and extract the post-netlist with power/ground to perform the entire design flow verification from register transfer level (RTL) to pre-netlist and post-netlist.

II. Related Work

Previous studies have suggested a validation methodology through power-intent strategy and multi-voltage design platform construction for standard cell library verification as illustrated in Figure 1. However, the study proceeded with the creation of strategy and platform, and only simulation-based verification. Therefore, it was impossible to identify any issue that would have occurred during validation with the framework. For example, if a specific cell is missing from the cell list for using the framework, then verification of that cell is not performed. In addition, we had to overlook inaccurate information if the strategy described in UPF was incorrect and no error occurred in the synthesis process. Besides, verification could not be performed for power netlist with power/ground connection because only front-end flow was proceeded. In the worst case, coverage for a specific cell was reduced, which could lead to significant issues during process of using the cell. To solve this problem, we develop the upgraded validation framework by using various methods as follows:

- Eliminate possibility of missing data by parsing the cell data from the published datasheet document.
- UPF validation with library by enabling VC-LP EDA tool.
- Crosscheck the power intent information between each view by enabling VCS EDA tool.
- Crosscheck the power & non-power functions by enabling Formality EDA tool.
- Automated back-end flow for power-aware netlist generation.

The power management kit and data retention logic kit require not only function verification but also checking whether the power intent information is properly included. This information cannot be confirmed using UPF only. We need to use the verification tool additionally. A more detailed description of the library verification method using low power verification used in design methodology is illustrated in Section III.

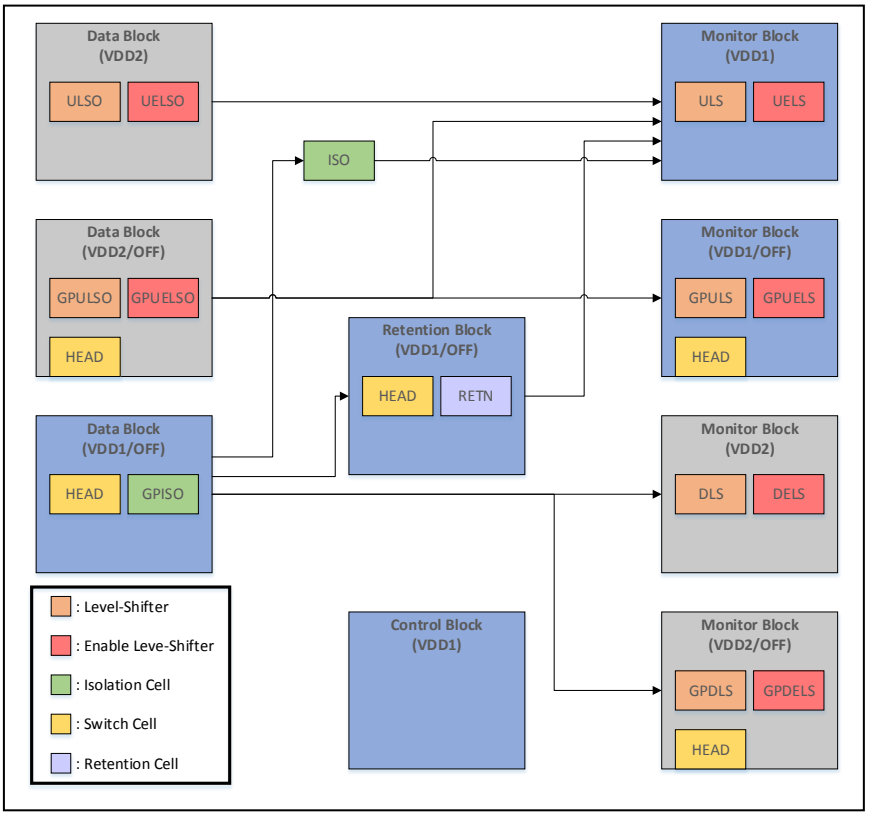

Figure 1. Block diagram of synthesized power-aware gate-level netlist

III. Low-Power Validation Framework

In general, the use of verification tools in design flow allows identification of any problem in the design. Designers can use verification tools to check whether the design is based on low-power rules or whether unintended changes have been applied to the design. However, we use the design verification tool for library verification. Library verification using the verification tool can identify any issue in the library. Unlike design verification, library verification verifies libraries on the assumption that the design netlist with UPF is correct. To make sure the correctness of design netlist and UPF, input files should guaranteed. To achieve this correctness, we use datasheet parser to create an input file with the contents in the datasheet so that there is no problem with the UPF and netlist. Through this, we can also check whether the contents described in the datasheet are correct. Below Figure 2 is illustration of our proposed verification flow. Following sub-sections describe detail procedure of this verification flow.

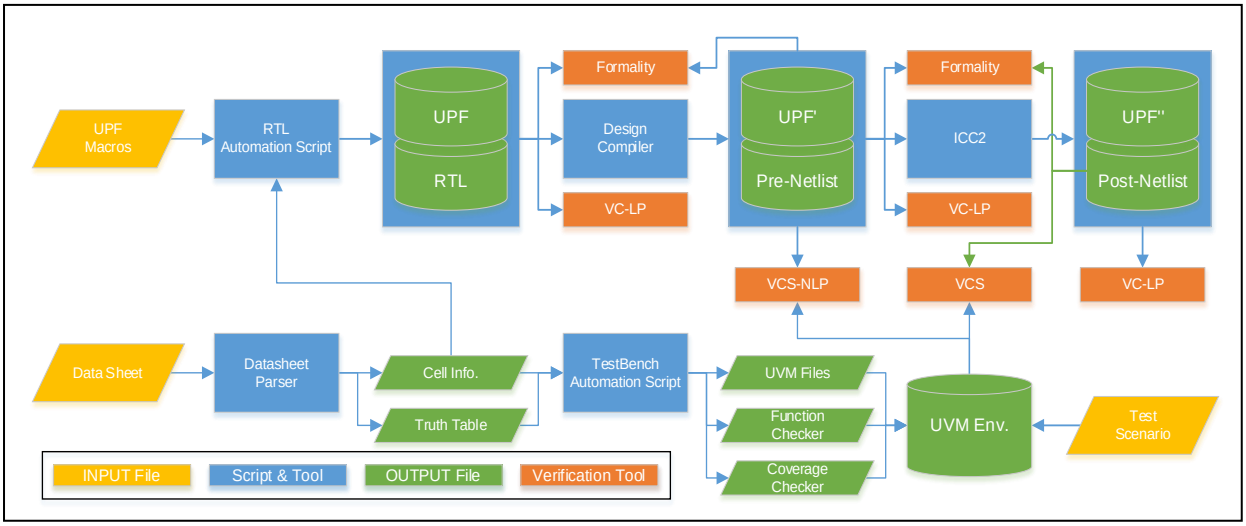

Figure 2. Automation flow diagram

A. *Datasheet parsing*

In the existing framework, the use of XLS file containing cell information presented many issues such as manual generation of information and truth tables. Not only does it take a lot of time to write, but it can also cause human errors such as incorrect input and omission of specific cells. We solved this problem by parsing the datasheet.

In general, the datasheet contains complete cell information, such as cell location and cell type. We can parse the information and extract it in JSON format. Figure 3 and Figure 4 illustrates the example for isolation cell. We can parse cell type, cell name, cell location, isolation enable condition, and clamp value. These contents can be extracted in JSON format, and XLS file can be generated using this file. In addition, since the truth table includes an expected output according to an input, it can also be used to generate a checker and a cover-group used for simulation.

1. GPA2ISO

Cell Description
- Source-side AND-type isolation cell, high enable with global power.

Truth Table

A	EN	VDDG	Y
0	X	1	0
X	0	1	0
1	1	1	1

Figure 3. Datasheet template

```
JSON
CELL_TYPE: isolation cell
CELL_NAME: GPA2ISO
CELL_CLAMP_VAL: 0
CELL_SENSE: 1
CELL_LOCATION: source
```

Figure 4. JSON file template

B. *Low-power rule check*

Low-power rule check is a power verification sign-off step for the designs employing advanced low-power design techniques. This is mandatory for checking power specification, such as power gating, multi-VDD, body bias, and so on.

We use this method in reverse to verify the library based on the design and UPF generated from datasheet. There are "*map_**" and "*use_interface_cell*" commands in UPF. These commands specify the functional model and a list of implementation targets for cells. They map the strategy described in the UPF to a target cell. Therefore, if there is a difference between the cell and the strategy, errors such as "*ISO_MAP_MISMATCH*" and "*LS_MAP_MISMATCH*" occur and we can check it. Because we think of UPF as golden, we can know that the specific attribute is incorrect or missing. In this way, attributes in liberty such as "*is_isolation_cell*" and "*level_shifter_type*" can be checked using this method.

C. *Low-power equivalence check*

Low-power equivalence check is an alternative to verification. It uses mathematical techniques to compare the logic to be verified against a logical specification or a reference design. It does not require input vectors and considers only logical functions during comparison.

This process can also be used for library verification. We can check whether the cells described in the UPF are used according to the function described in the specification. This method is particularly efficient in verifying retention cells because general functions such as 'clear' and 'preset' are described in RTL and retention functions are described in UPF. Other cells are not defined in RTL but are used in the synthesis process using UPF. If the RTL and UPF functions are not same as the liberty function of the cell mapped in the UPF, failing point occurs. Through this method, we can check attributes in liberty such as "*clear_preset_var1/2*" and "*clear/preset*".

D. *Low-power simulation*

Low-power simulation is to simulate the power intent, low-power design logics, and as the impact on the functionality of the design. The power intent describes the power domain partition of the design, the low-power control signals, and the power states of the power domains.

Verilog simulation can be performed using pre-netlist and post-netlist. If the functions of the netlist used in each simulation are same, the two results must be same. We can check two library verifications here; (1) whether the function of non-power-aware and power-aware Verilog model are the same or (2) the power down functions of Liberty and power-aware Verilog model are the same. It can be checked because non-power-aware simulation uses Liberty and non-power-aware Verilog model and power-aware simulation use power-aware Verilog model. To do this, we use checker and cover-group made from datasheet and we use them in both non-power and power-aware simulation for consistency. In addition, the test scenario made it possible to check the function according to power on/off.

E. *Back-end flow*

Normally, back-end flow needs to consider many things such as design rule and metal layer. However, because our purpose is only post-netlist extraction, we consider only the minimum back-end element. So the key points in focus are physical intact and independent of the process.

For physical intact, in our low-power design, the generated RTL also has multiple power domains and P&R block. The total block consists of top block and several sub-blocks. Top block has main power domain and several sub-blocks have head cells for power gating. All blocks are surrounded by layout finishing cells and guard rings because these blocks should be independent to each other and we can ensure that there is no short path between each domain.

For general use, to cover various process nodes we used standard cell's unit such as CPP(Contacted Poly

Pitch), single height, offset and space of the lowest vertical metal of power mesh, and latch-up distance as variables. For example, width and height of sub-block are multiple of standard cell's unit. Considering latch-up distance, head cells are located in 1/4 and 3/4 horizontal point of each sub-block.

IV. Experimental Result

The low-power validation framework we proposed has enhanced the validation flow compared to previous works. We are focusing on the verification tools that have not been confirmed before, and the results are as follows.

Low power rule check stage check that the low-power description in both Liberty and UPF are same. Figure 5 illustrates the examples for isolation cell and level-shifter cell. In case of isolation cell, if "*is_isolation_cell*" attribute is not defined in Liberty but this cell is used in "*use_interface_cell*" UPF command, VC-LP detect "*ISO_MAP_MISMATCH*" error. In case of level-shifter cell, if "*level_shifter_type*" attribute in Liberty is "*LH*" but UPF is described as high-to-low, VC-LP detect "*LS_MAP_MISMATCH*" error.

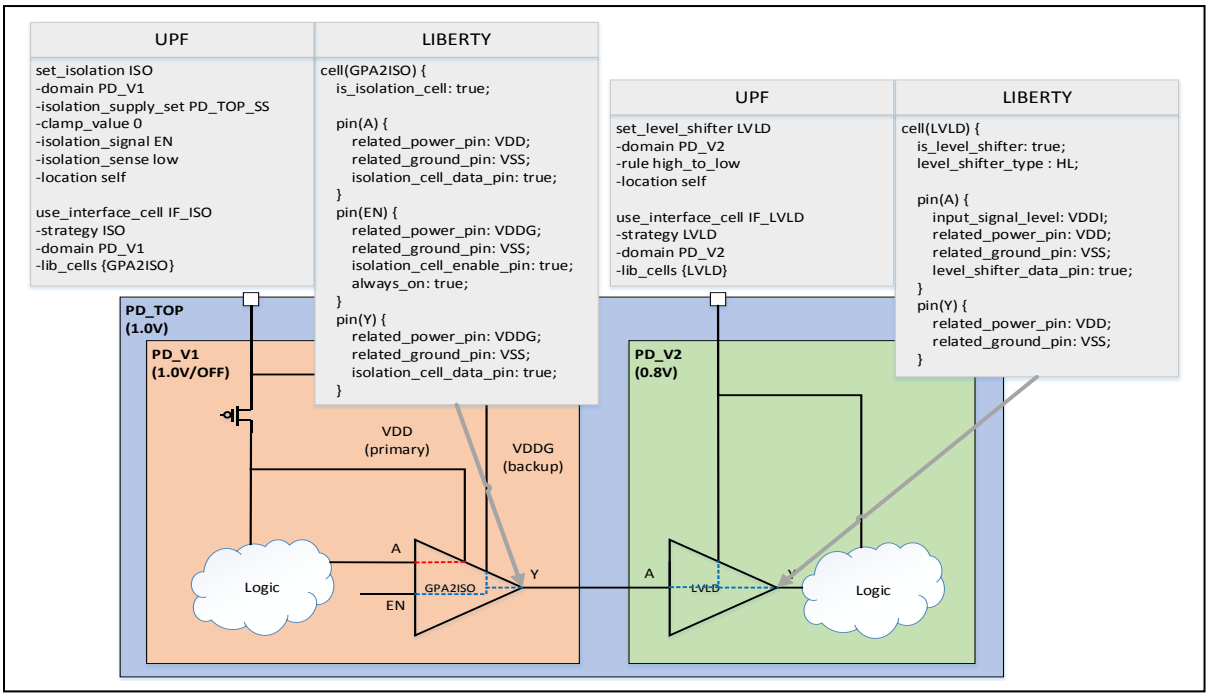

Figure 5. Low-power rule check for isolation cell and level-shifter cell

At low-power equivalence check stage, we can check the consistency between Liberty versus RTL and UPF. Figure 6 illustrates example for retention flip-flop with both set and reset. If preset priority is higher than clear, "*clear_preset_var1*" attribute in Liberty should be "*H*". RTL also describes the priority between preset and clear. The condition described in if statement has higher priority than those described in else-if statement. Therefore, "*SN*", which is preset, has higher priority than "*R*", which is clear. Through this process, we can check if "*clear_preset_var1/2*" attribute is properly described.

LIBERTY	RTL	UPF

ff(IQ,IQN) {
clocked_on: CK;
next_state: (D * !SE + SI * SE);
clear: (R * RETN) + (!RETN * !IQ2);
preset: (!SN * RETN) + (!RETN * IQ2);
clear_preset_var1: H;
clear_preset_var2: L;
}
latch(IQ2,IQN2) {
enable: RETN;
data_in: IQ;
}
	always @ (posedge clk, negedge SN, posedge R)
	begin
	if (!SN)
	Q <= 1'b1;
	else if (R)
	Q <= 1'b0;
	else
	Q <= D;
	end
		set_retention CELL_NUM
		-domain RET_DOMAIN
		-retention_supply AON_POWER
		-elements {Q}
		-save_signal {RETN high}
		-restore_signal {RETN low}
		
		map_retention_cell CELL_NUM
		-domain RET_DOMAIN
		-lib_cells {cell}

Figure 6. Low-power equivalence check for retention flip-flop

At low-power simulation stage, we can check the consistency between Verilog model and Liberty. Because there is no power/ground connection in the pre-netlist, it uses the information in the liberty.

In enable level-shifter, power_down_function of Liberty and Verilog model should be same. We can check it to compare the result of the two. Figure 8 and Figure 9 illustrate the simulation results when output is incorrect and correct due to power_down_function. If EN condition is omitted from the power_down_function of the Liberty or Verilog as illustrate in Figure 7, Y will not be "*0*" but "*x*" at the time of isolation enable. Because the result is different from the expected output in that case, we can check if there is an issue by checking for the error in the log.

LIBERTY	VERILOG
... power_down_function : "!VDD + (!VDDI & EN) + VSS + VPW + !BIASNW" ; ...	... and I0 (out_temp, A, EN); assign Y = ((VDD === 1'b1) && (BIASNW === 1'b1) && (VPW === 1'b0) && (VSS === 1'b0) && (!EN|VDDI === 1'b1))? out_temp : 1'bx; ...

Figure 7. Correct Liberty and power-aware Verilog model of the enable level-shifter

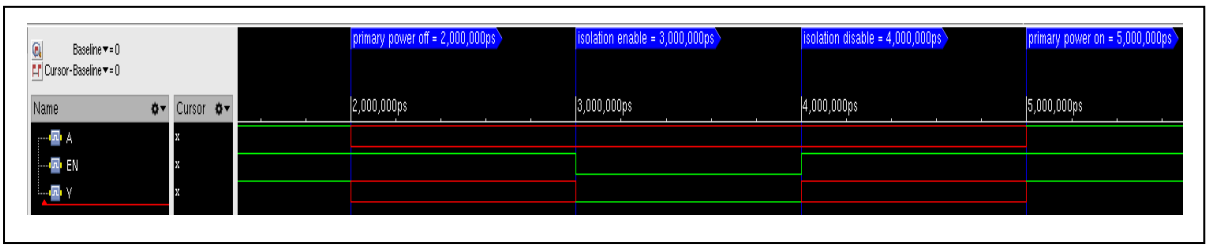

Figure 8. Correct simulation waveform snapshot of the enable level-shifter

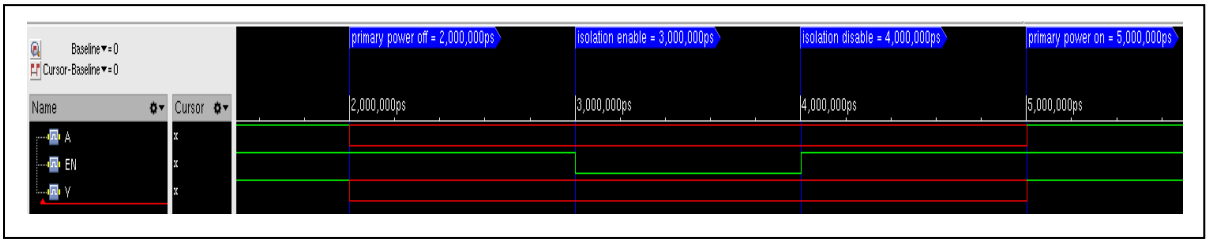

Figure 9. Incorrect simulation waveform snapshot of the enable level-shifter

In retention flip-flop, when primary power is turned off and on in retention mode, non-power and power-aware model should be same for the interval in which output x appears. Figure 10 and Figure 11 illustrate that x-propagation is different due to mismatch of retention modeling. If retention modeling is incorrect and x appears in

the output even after the primary power is turned on, we can check through checker that the retention value does not come out.

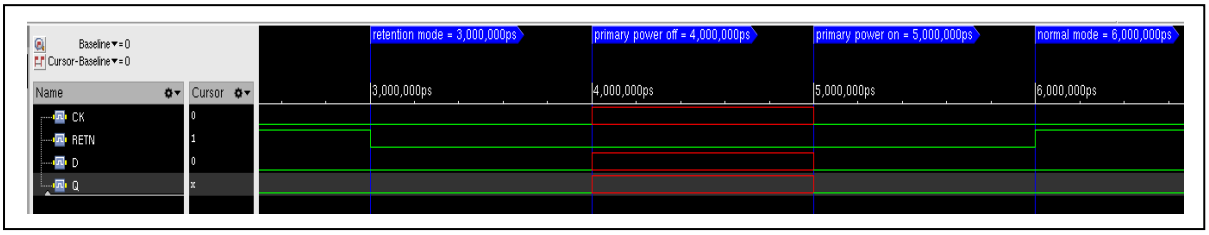

Figure 10. Correct simulation waveform snapshot of the retention flip-flop

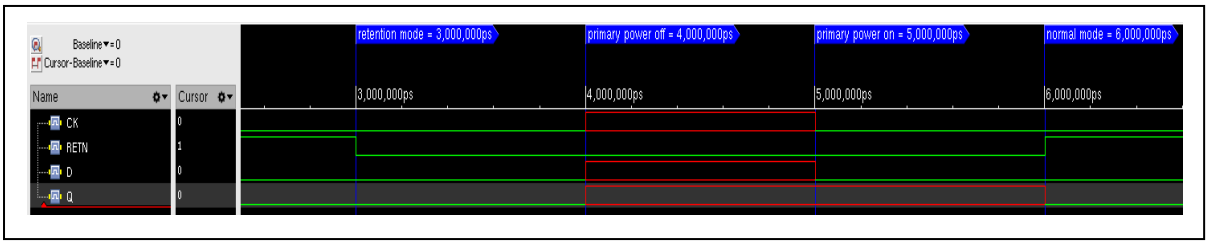

Figure 11. Incorrect simulation waveform snapshot of the enable level-shifter

Figure 12 and Figure 13 illustrate the P&R block using 3nm process and 4nm process, respectively. Using the variables mentioned in back-end flow, top and several sub-blocks were created in two different process nodes. Size of sub-blocks is decided by CPP and single height, so 4nm process sub-blocks are larger than that of 3nm.

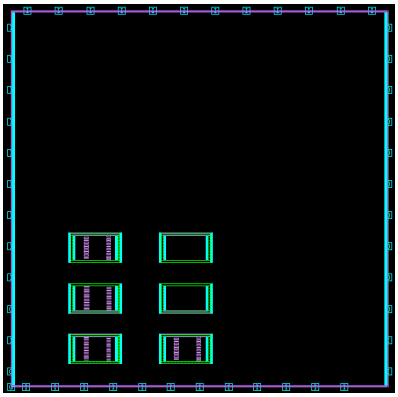

Figure 12. 3nm power domain floorplan

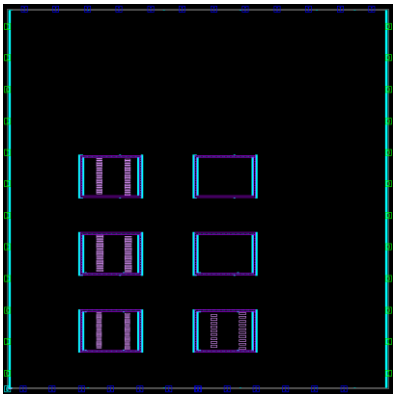

Figure 13. 4nm power domain floorplan

V. Conclusion

The most important thing in the foundry business is customer's trust. Therefore, we are putting a lot of effort to strengthen library quality. One of the methods is to check view-to-view or view verification at the cell level. However, it is impossible to completely verify the design kit by verifying only at the cell level. Therefore, we proceed with the design to identify problems that customers may experience in the design process in advance. Unlike the reason designers proceed, we use the design verification tool for library verification. In addition, we have increased the coverage of design methodology by proceeding not only front-end flow but also back-end flow. Samsung Foundry uses this validation framework to enhance the integrity of library and achieve highly qualified libraries for the general foundry customers.

References

[1] Kamath, A., Kumar, B., Aggarwal, S., Parameswaran, S., Lonkar, P., Prasanna, D., Sreenath, S., “An automated validation framework for power management and data retention logic kits of standard cell library”, DVCON, USA, 2021.

[2] Kamath, A., Kumar, B., Aggarwal, S., Parameswaran, S., Goyal, M., Lonkar, P., & Sreenath, S., "A comprehensive multi-voltage design platform for system-level validation of standard cell library," 2021 22nd International Symposium on Quality Electronic Design (ISQED), Santa Clara, CA, USA, 2021, pp. 285-291, doi: 10.1109/ISQED51717.2021.9424350.

[3] Unified Power Format, IEEE Standard for Design and Verification of Low-Power, Energy-Aware Electronic Systems, 27, September 2018.

[4] V. Gourisetty et al., "Low power design flow based on Unified Power Format and Synopsys tool chain," 2013 3rd Interdisciplinary Engineering Design Education Conference, Santa Clara, CA, USA, 2013, pp. 28-31, doi: 10.1109/IEDEC.2013.6526754.

[5] “VC-LP” Synopsys, Inc., https://spdocs.synopsys.com/dow_retrieve/qsc-s/vg/vc_lp/S-2021.09-SP1/vclp_olh/htmlhelp/index.html#page/VC_LP/Chapter1.1.4.htm

[6] “VCS Native Low Power(NLP)” Synopsys, Inc., https://spdocs.synopsys.com/dow_retrieve/qsc-u/vg/VCS/U-2023.03-SP1/vcs_olh/index.htm#page/VCS_LCA_Features_Guide/mvsim_native_features.20.01.htm

[7] “Formality Equivalence Checking.” Synopsys, Inc., www.synopsys.com/implementation-and-signoff/signoff/formality-equivalence-checking.html.