ICIC Express Letters
Volume **12**, Number **3**, March **2018**
 ISSN 1881-803X
pp. **213–220**

DEVELOPMENT OF NDA FREE VLSI DESIGN FLOW FOR 0.6μM COMMERCIAL FABRICATION

Naohiko Shimizu

School of Information and Telecommunication Engineering
Tokai University
2-3-23, Takanawa, Minato-ku, Tokyo 108-8619, Japan
nshimizu@tokai.ac.jp

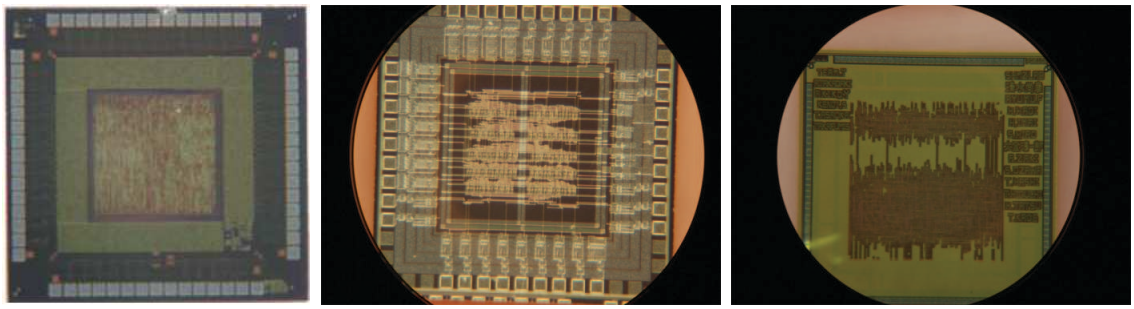
Received September 2017; accepted December 2017

Abstract. *We will present the development of NDA free VLSI design flow for 0.6μm commercial fabrication. Our design flow will help users to democratize VLSI design. The flow consists of two parts. One is the layout design with a scalable open design rule. The other is to translate the scalable design rule to the specific rules provided by fabrication company under NDA. Ordinary user does not have to care on the second flow but stay at the first flow which will be enough to fabricate their VLSI chip. The chip with this method was fabricated in success.*
Keywords: VLSI design flow, NDA free, EDA, Fabrication

1. **Introduction.** The Very Large Scale Integration (VLSI) chip is the fundamental technology in modern world. To fabricate VLSI chip, usually a user has to use the Process Design Kit (PDK) provided by the fabrication company with signing up a Non-Disclosure Agreement (NDA). The PDK consists of the process design rules, device models, cell libraries, etc., and it will designate the ability of the fabrication company. The NDA will prevent users to share their knowledge and/or experiences between them even in universities. In US, Mead and Conway had changed the situation with scalable MOS design [2]. Their method provides an open design rule based on a unit lambda and applies the same rule to every fabrication process only with changing the lambda value. It worked well during 1980s to 1990s, and many universities fabricated their original VLSI chips through MOSIS service [3]. Nanometer technology breaks the MOS scalability and the scalable MOS method became obsolete. MOSIS still provide fabrication opportunity with scalable CMOS (SCMOS) rules, but in 2017, they only provide 0.5 micron ON Semiconductor chips. Prof. Akita, Kanazawa Univ. started MakeLSI community [1] to share the knowledge and experiences among users for democratizing VLSI design. We started with Kitakyushu FIAS process because FAIS will not request us to sign up NDA [8].

The author has fabricated several VLSI chips with open source Electronic Design Automation (EDA) tool set Alliance by LIP6, UPMC, France [6,7]. Figure 1 shows the LSI chips we designed with Alliance. The author created own PDKs for Alliance from the NDA based foundry PDKs.

With these experiences, we decided to introduce a virtual layout layer based on MOSIS SCMOS rules and convert the layout against a real one with a translation method. This method will not require users to sign up NDA, because users only read the MOSIS open rules. In this paper, we present the method of the translation with a virtual layout based on MOSIS SCMOS DEEP submicron rules against Phenitec 0.6 micron process rules and the fabrication process.



(a) (b) (c)

FIGURE 1. VLSI chips which were designed with Alliance: (a) Rohm 0.35μm, (b) ON Semiconductor 1.2μm, (c) Rohm 0.18μm

2. **Problem Statement and Preliminaries.**

2.1. **Alliance symbolic layout.** Alliance uses symbolic rules internally. The famous symbolic rule in VLSI design is the lambda rule which is proposed by Mead and Conways in 1980. The lambda rule defines every dimension as multiples of a unit size lambda. The lambda is selected as a half of the gate width of a minimum transistor. It works only for over micron technologies and the simple method had problems for sub micron layout rules.

Alliance uses more sophisticated method to convert symbolic layout to real one. Figure 2 shows an example of transistor conversion. In this example, a symbolic transistor with 1 unit for width and 3 unit for length is provided by user. The transistor will be converted to the required layers such as poly silicon, active, and N select (implant). Not only the layers, but we are also able to convert the size of each layer with a conversion database. The conversion database must be written for each fabrication process to follow the design rules. In [2], we developed a cell library whose target is MOSIS SCMOS DEEP and also developed the conversion database and design rule check database for the technology. We succeeded in building the chip layout design flow for MOSIS SCMOS DEEP with Alliance tool set. MOSIS open design rules are not directly applicable to any commercial fabrication processes. We expect that it must be compatible with TSMC CL018 for which MOSIS claims to be able to fabricate the chip on their web site.

Figure 3 shows our logic VLSI design flow with Alliance and NSLCore. We use NSL for the design entry because the VHDL used in Alliance has some obscured restrictions and is a little difficult to manage. The NSL compiler is able to generate Alliance compatible

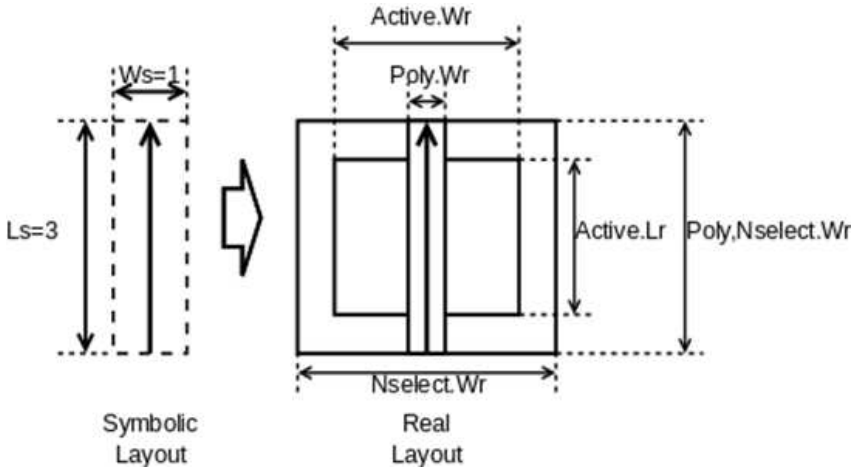


FIGURE 2. Symbolic to real layout conversion on Alliance

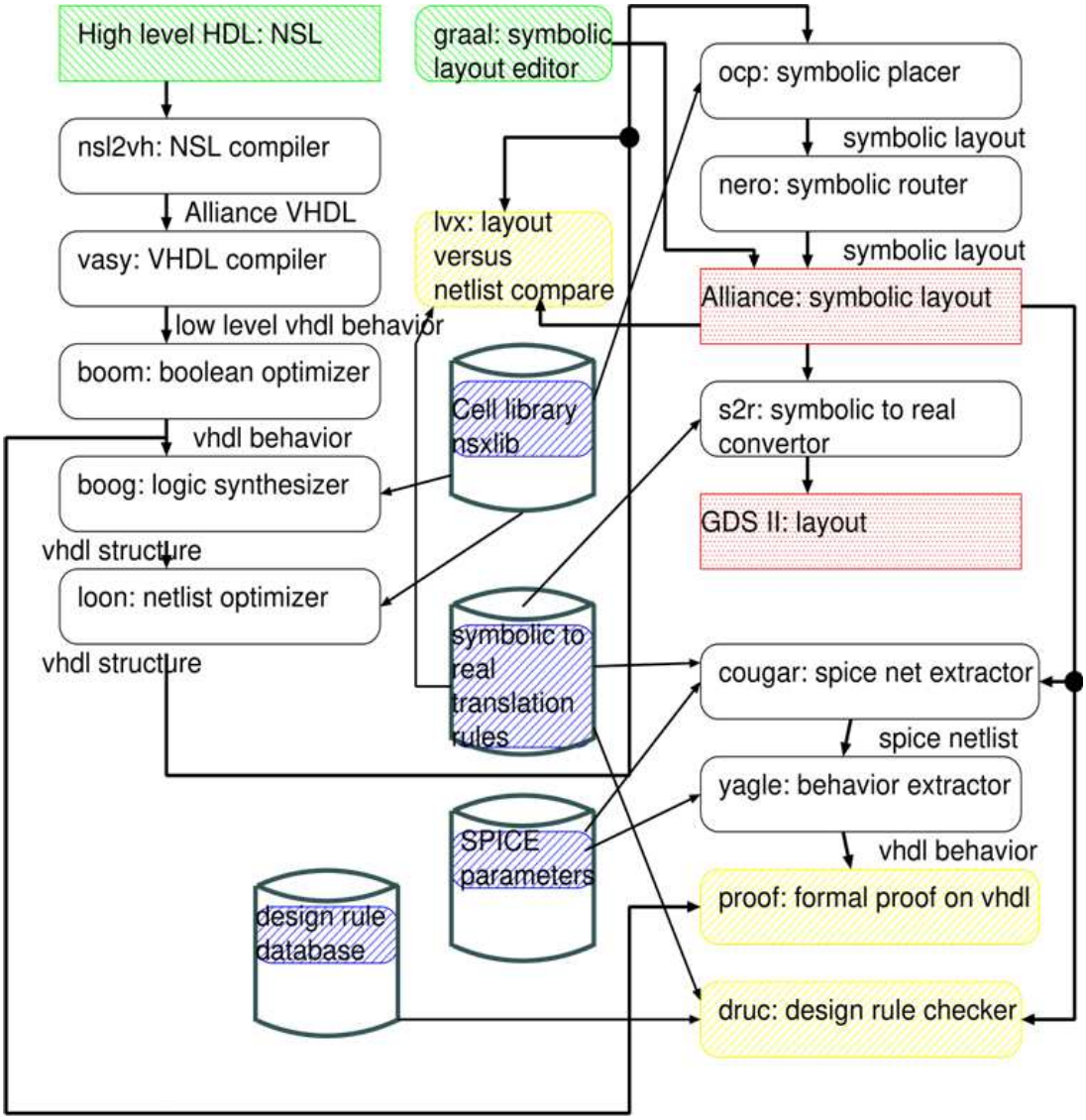


FIGURE 3. VLSI design flow with Alliance

VHDL with "-vasy" option that is specially designed for Alliance. Users have to write their logic in NSL or they can use symbolic editor graal to write the symbolic layout directly. As in Figure 3, we use three checking methods for our design: 1) design rule check with druc, 2) layout versus schematic with lvx and 3) formal proof on behavioral vhdl with proof.

There are four databases which users have to prepare. In [2], we developed these databases for MOSIS SCMOS DEEP: 1) cell library that includes all logic cells to make their layout, the symbolic layout of each logic cells, the vhdl behavior file for each logic cell, and optionally real layout of cells which is written in GDSII or CIF; 2) symbol to real layout conversion database which is the most important and perhaps most difficult database which the users have to write; 3) SPICE parameters to generate SPICE netlist and to calculate the delay, power consumption, etc.; 4) design rule checking database and messages catalog.

2.2. **Using layout which follows MOSIS SCMOS DEEP as a virtual layer.** Users have to develop the databases for a specific process with the data which is provided by the fabrication company under NDA. This development will prevent users from sharing their design and/or experiences among other users. Therefore, we develop a new method to use MOSIS SCMOS process as a virtual layer for the Phenitec 0.6μm process. Unfortunately,

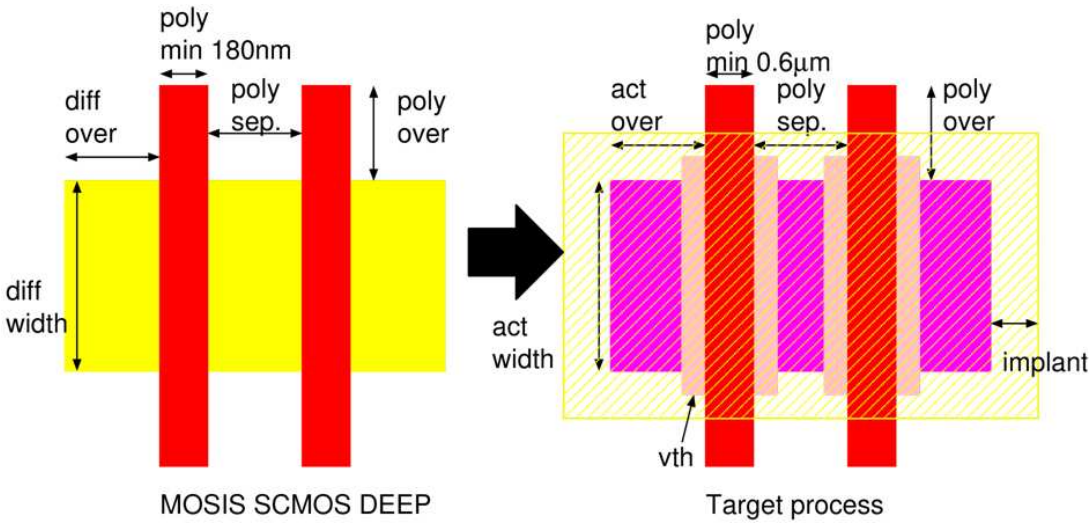


FIGURE 4. Translation of segments

the SPICE parameter is not able to scale and we will not use the parameter for MOSIS SCMOS DEEP. However, we will extract parameters from test and evaluation chip. The cell library has been developed under the symbolic rules. We expect that the library will work without modification with 0.6μm process. Therefore, we should care about the symbolic to real translation database and the design rule check database.

Figure 4 shows the translation example from MOSIS SCMOS DEEP to a target process. The real dimensions are under NDA and will not be able to disclose in our paper then we will not show detailed dimensions in the figure.

As shown in this figure, the virtual layers will not directly be corresponding to the real layers. Some virtual layer will be converted to a set of real layers such as the diffusion which will be converted into the active, the implant, the vth layers, respectively. We can adjust the segments length and width and the overhangs on the layout are easily modified with the symbolic to real translation database. The separation requirements are much difficult to manage. The origin of each segment in the layout is defined on the layout file in symbolic coordinate and when we fix the basic unit for the translation, what we can do to make more separation between segments is to change the width or offset of the segments. It will not be flexible as required. Sometimes the only thing we can do is to expand the unit length on this translation. Even if we expand the unit length, we can keep the gate width as the minimum with the translation database, because we can adjust the width of segments.

We have to write the design rule check database following the design rules of the fabrication company. The rule itself is covered under NDA and we cannot disclose the rules in this paper. Instead of the design rules for the fabrication process. Figure 5 shows a part of the design rule checking database for MOSIS SCMOS DEEP. We will develop the corresponding database for the target process.

Figure 6 shows the design flow we develop. With the MOSIS layer, we will have a symbolic layout and netlist and behavioral description in VHDL. We intend ordinary designer to use only the MOSIS rules because one has to sign up NDA to see the target rules. After the ordinary designer completes his design with MOSIS rules, our method translates the deign to the target rules with our original translation table.

We applied the design flow for all of the logic cells in the cell library and the target design itself. The formal proof of the whole design usually is not mandatory because we proved the functionality of each cell with formal verification and the connection between the cells is proved with the layout versus schematic checking.

```
#-------------------------
caracterise RDS_POLY (
regle 310 : largeur >= 0.18 ;
regle 311 : longueur_inter >= 0.18 ;
regle 320 : notch >= 0.27 ;
);
relation RDS_POLY , RDS_POLY (
regle 321 : distance axiale min 0.27 ;
);

# Check the CHANNEL shapes
#-------------------------
caracterise CHANNEL (
regle 322 : notch >= 0.27 ;
);
relation CHANNEL , CHANNEL (
regle 323 : distance axiale min 0.27 ;
);
```

FIGURE 5. Part of the design rule checking database for MOSIS SCMOS DEEP process with lambda = 90nm

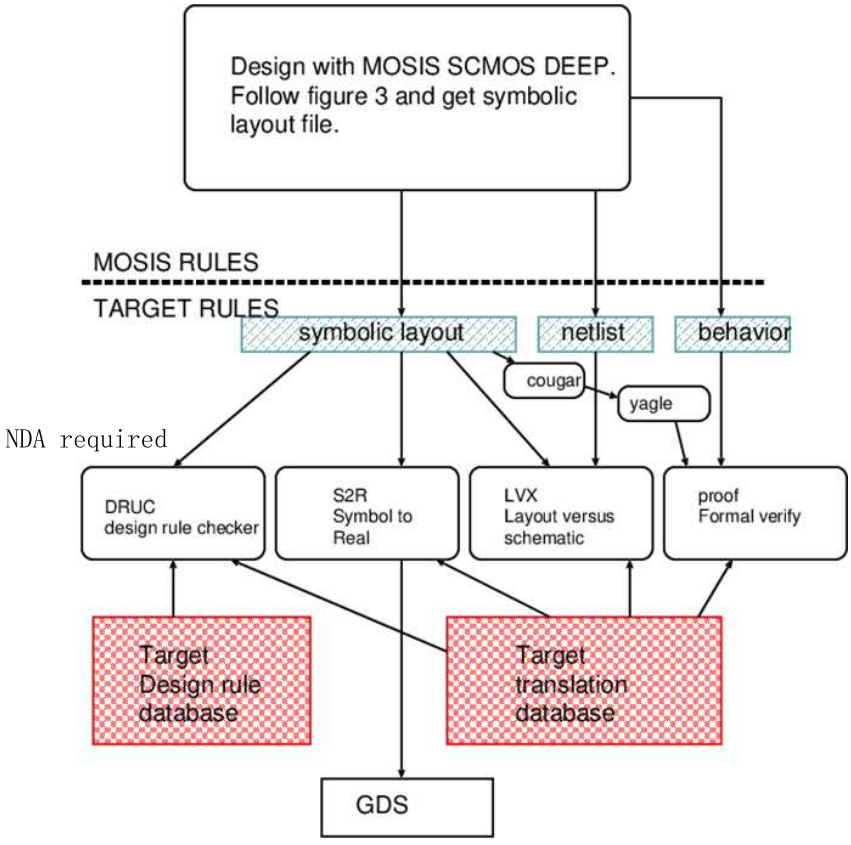


FIGURE 6. Our VLSI design flow with MOSIS virtual layer

Alliance uses symbolic rules for the place and routing. Unlike lambda based design, Alliance with the translation database modifies not only the width and length of segments but the layers which consist of the segments. If the translation database has some problems, it is very possible that we will have some cells or design which will not meet the design rules. Therefore, the target design rule database is very important for this flow, because we have to verify the target translation database with the design rule checker on the way to develop the translation database.

3. **Main Results.** With our target specific symbolic to real conversion database, we successfully converted the symbolic layout of 8-bit CPU which was produced with MOSIS

SCMOS DEEP design rules to the layout of the target rules. Also, we check our layout with the design rule checker with the target rules. There is no design rule error in the converted layout as Figure 7.

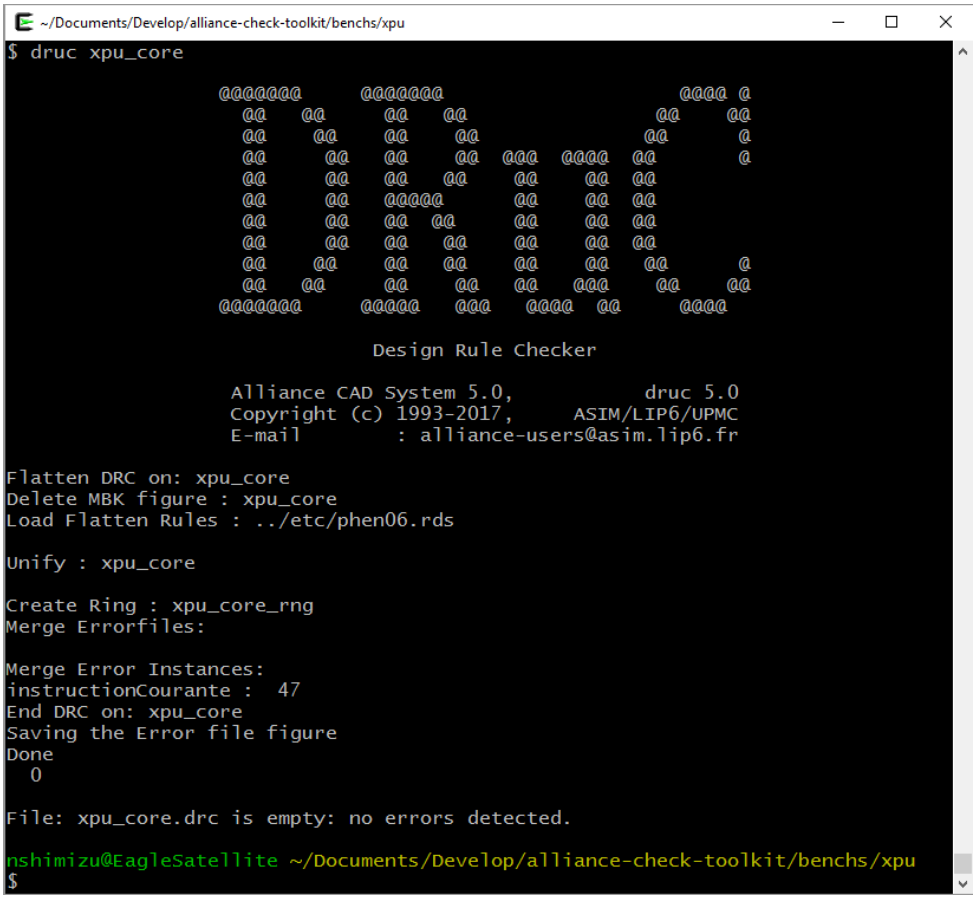
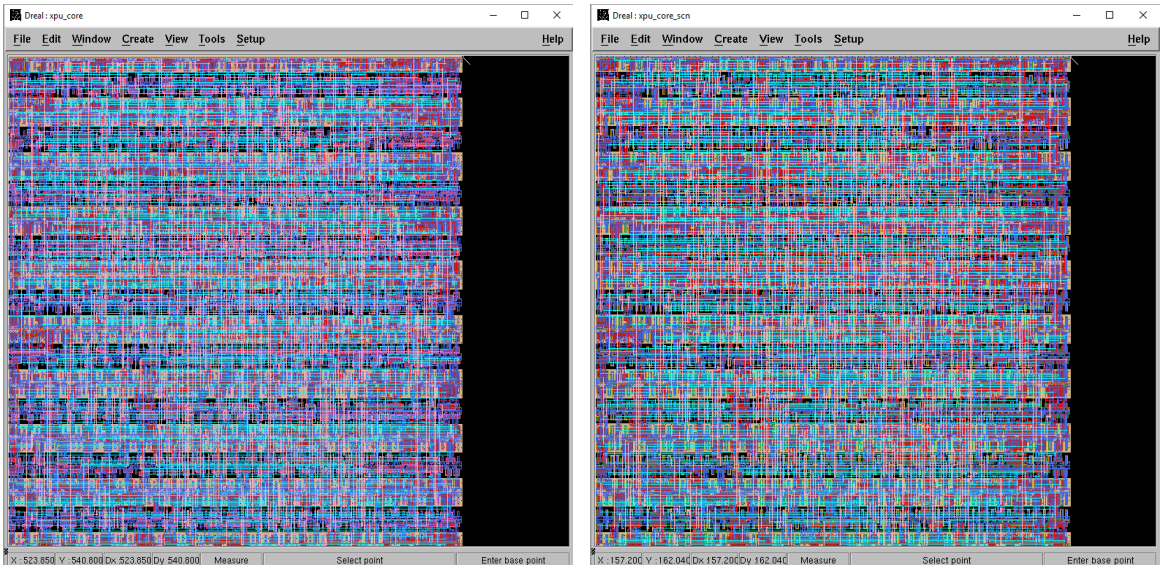


FIGURE 7. Design rule check results in no error



(a) (b)

FIGURE 8. GDS-II layout of 8-bit CPU: (a) 0.6μm rule, 522 $\times$ 540μm, (b) 0.18μm rule, 157 $\times$ 162μm

Figure 8 shows the two GDS layouts which are produced from the same symbolic layout of a simple 8-bit CPU. The CPU is written in 56 lines of NSL. It has 16 instructions. We follow Figure 3 design flow to make the symbolic layout. The symbolic layout contains 3190 transistors. Figure 8(a) is the layout on our target 0.6μm technology, and Figure 8(b) is on MOSIS SCMOS DEEP with lambda equal to 90nm. They are very similar because the symbolic layout is just the same and only symbolic to real conversion database is different between them.

In addition to the 8-bit CPU core, we designed test circuits to check our standard cell library and a ring oscillator to check the circuit performance. We also designed input, output, power supply cells to meet the dimension requirement of the foundry chip. As noted above, all the cells were designed with SCMOS DEEP design rules and converted against the foundry specific rules.

Our PDK has the Design Rule Check (DRC) database and we checked our design before fabrication. In addition to our own DRC, we also used foundry provided DRC database for Mentor Graphics' Caribre which is the standard DRC tool for the foundry. Our design had no problem with them either. The logic design will be merged with the chip frame provided by foundry, and the merged GDSII data was sent to the foundry.

Figure 9 shows the picture of fabricated chip.

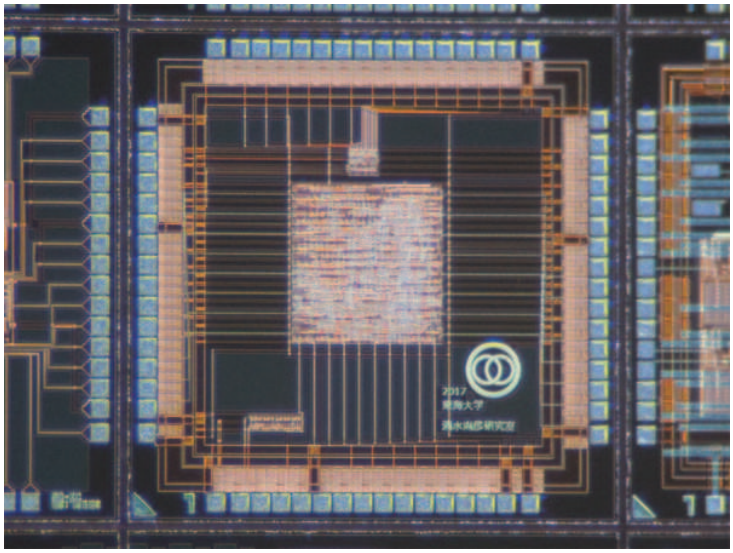


FIGURE 9. Fabricated VLSI chip

4. **Conclusions.** We successfully convert our NDA free design to a commercial fabrication process with Alliance. The converted layout is DRC clean and we will be able to fabricate the layout with Phenitec 0.6μm process. Now when we write this paper, we are on the way to tape out our design, and we have to finish our layout with I/O cells. I/O cells in Alliance are a little big to fit in our design, and then we will rewrite the cells to shrink and fit in our dimension.

We believe with this methodology, ordinary people do not have to look at the NDA papers because they only use MOSIS SCMOS DEEP design rules and open source symbolic cell libraries. All the required conversion can be done on symbolic layout and only the licensed person will process the conversion. Therefore, we will achieve our objective to democratize LSI making without NDA on our methodology.

Acknowledgment. We express special thanks to Phenitec Semiconductor Corp. for their support and providing an opportunity to fabricate the experimental chip. We also express special thanks to Mr. Akihiro Yamada who carries the shuttle service on the technology and informed us for this opportunity. We express special thanks to Prof. Junichi Akita who started MakeLSI community. We also express special thanks to Marie-Minerve Louerat and Jean-Paul Chaput in LIP6, UPMC, France who have exchanged the basic ideas for these topics and have deeply supported Alliance and related tools.

REFERENCES

[1] J. Akita, Open source LSI design & fabrication project for distributed IP development, *International Conference on Analog VLSI Circuits*, Boston, USA, 2016.
[2] C. Mead and L. Conway, *Introduction to VLSI Systems*, Addison Wesley, 1980.
[3] *MOSIS*, https://www.mosis.com/.
[4] J. E. Stine et al., FreePDK v2.0: Transitioning VLSI education towards nanometer variation-aware designs, *IEEE International Conference on Microelectoronic Systems Education*, pp.100-103, 2009.
[5] G. A. Sanca et al., Development of a sCMOS interoperable process design kit and an open set of standard digital cells, *Argentine Conference of Micro-Nanoelectronics, Technology and Applications*, pp.6-10, 2016.
[6] N. Shimizu, J.-P. Chaput and M.-M. Louerat, Development of a NDA free cell library for 180nm CMOS technology with FOSS EDA tools, *The 5th Asia Symposium on Engineering and Information*, Hanoi, Vietnam, pp.63-71, 2017.
[7] T. Hosokawa, T. Imai, T. Higuchi and N. Shimizu, The application scalable cell libraries to SCMOS with ROHM0.18μm process and the learning method, *ITC-CSCC2011*, pp.895-898, 2011.
[8] *FAIS*, http://www.ksrp.or.jp/e/info/fais.html.