World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:9, No:10, 2015

Physical Verification Flow on Multiple Foundries

R. Abdul Wahab, R. Mohd Fuad Tengku Aziz, N. Othman, S. Saleh, N. Razali, M. Al Baqir Zinal Abidin, M. Hanif Md Nasir

Abstract**—**This paper will discuss how we optimize our physical verification flow in our IC Design Department having various rule decks from multiple foundries. Our ultimate goal is to achieve faster time to tape-out and avoid schedule delay. Currently the physical verification runtimes and memory usage have drastically increased with the increasing number of design rules, design complexity, and the size of the chips to be verified. To manage design violations, we use a number of solutions to reduce the amount of violations needed to be checked by physical verification engineers. The most important functions in physical verifications are DRC (design rule check), LVS (layout vs. schematic), and XRC (extraction). Since we have a multiple number of foundries for our design tape-outs, we need a flow that improve the overall turnaround time and ease of use of the physical verification process. The demand for fast turnaround time is even more critical since the physical design is the last stage before sending the layout to the foundries.

Keywords**—**Physical verification, DRC, LVS, XRC, flow, foundry, runset.

I. Introduction

THE Department of Integrated Circuit Research and Development in MIMOS Berhad uses Electronics Design Automation (EDA) tool from Mentor Graphics to run verification to check whether the designs are clean from DRC and LVS violations. Another function for physical verification is parasitic capacitance extraction (XRC) for post layout simulation. We have designs from different technologies such as 0.18 micron and 0.35 micron and the layout (in GDSII) are sent to different foundries. These foundries cater for different technology i.e. MIMOS cater for 0.35 micron technology whereas XFAB and Silterra cater for 0.18 micron technology. Since physical verification is the last stage before tapeout and requires iteration between front-end design engineers (schematic and transistor level) and physical layout engineers, physical verification flow need to be optimized to fasten the turnaround time.

While a physical verification tool has many more applications, such as layout post processing and rule based optical proximity correction for manufacturing, software runtime always depends heavily on both the size of the layout to be verified (LVS) and number of rules to be checked (DRC)–two parameters that are increasing with every new design [1].

Rohaya Abdul Wahab, Raja Mohd Fuad Tengku Aziz, Nazaliza Othman, Sharifah Saleh, Nabihah Razali, Muhammad Al Baqir Zinal Abidin and Md Hanif Md Nasir are with MIMOS Berhad, Department of Integrated Circuit Development, Technology Park Malaysia, 57000 Bukit Jalil, Kuala Lumpur, Malaysia (Phone: 603-89955000; Fax: 603-89914234; e-mail: rohaya@mimos.my).

Fig. 1 shows how the physical verification task can be thought of as several runset commands running on a layout (full chip or block) producing error output and modified layout (in cases where physical verification is used for layout modification as in rule based optical proximity correction) [1]. To reduce the turnaround time, an optimal processing platform should be able to determine the runset command and portion of the layout to which the command is applicable, and then send these combinations to separate CPUs for processing. The results can be reassembled at the end [1].

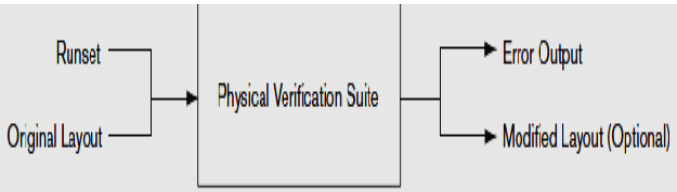

Fig. 1 System view of physical verification problem

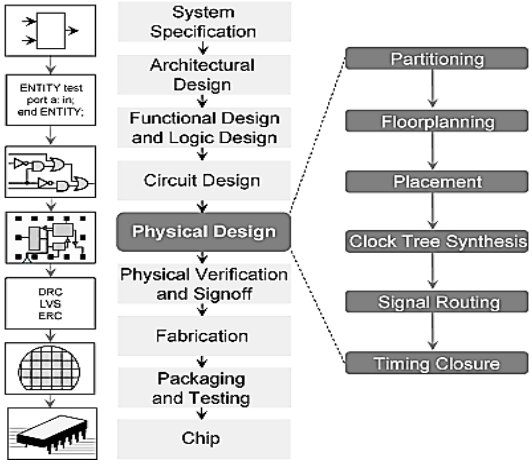

Fig. 2 The diagram of a physical verification flow

II. Background

Since 1992, our department had been using EDA tools with sign-off checks to perform physical verification on the layout before taped out. This implies a series of iteration involving incremental fixes between front-end designers and physical designers. With technology moving to nanometer geometries, the design becomes more complex and time consuming. Poor communication among engineers and between engineers and CAD support has resulted in major layout rework that adds significant time to the design cycle. Risk of design problems is also increased because much of this rework is hastily done in the final days prior to tapeout [2]. Worse yet, since some communication has no permanent record, this can lead to

completely missed requirements that result in nonparametric performing silicon that requires a costly second run through the fab [2]. Fig. 2 shows a physical verification design flow.

III. Implementation

The biggest challenge for physical designers is how to verify and deliver a design that is free of DRC violations while meeting their tapeout schedule [3]. Currently we send our tapeouts to three different foundries but use the same physical verification tool to fix DRC, LVS and extract the layout (XRC). Previously physical designers copy the rule decks from previous foundries to their home directories. Sometimes they do not aware with new updated rule decks thus end up using the same old rule decks. With this flow, latest version of rule decks will be updated by the system administrator at pdk folders. We started by creating runsets for each rule deck. All runsets are located at /pdk/master_lib folder.

First, we will discuss how we organize the runsets for all the rule decks that come from different foundries to make sure smooth workflow because users will be using runsets from the same source. Pdk (process design kit) is a set of files used within the semiconductor industry to model transistors for a certain technology for a certain foundry.

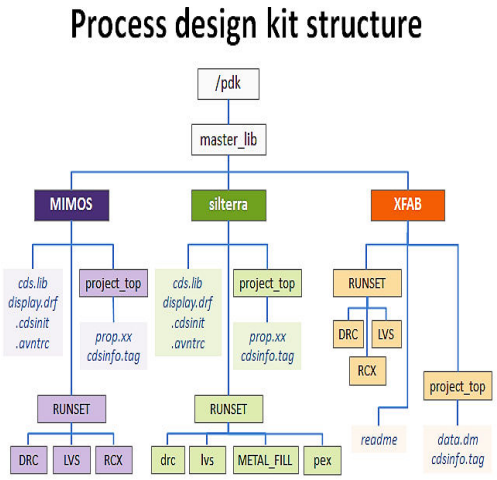

Fig. 3 Structure of our pdk for three different foundries

Fig. 3 shows the structure of our pdk for three different foundries. We store all runsets under the same path (i.e. /pdk/master_lib). Under master_lib we separate the three foundries which are our own in-house foundry (MIMOS), Silterra and XFAB. The runsets are used to run physical verification using Calibre from Mentor Graphics.

Second, we will discuss about the significance of the runset and how it helps to reduce overall turnaround time to meet tapeout schedule. The physical verification solution needs to provide DRC error information as early as possible in the design process, to minimize iterations, and to give designers more control over how much of the design is verified at a given time [3].

The overall runtime of the physical verification flow is impacted by the size of the design, the number of design rules and the complexity of the rules [3]. Fig. 4 shows how larger design size impacts the overall physical verification run time [3].

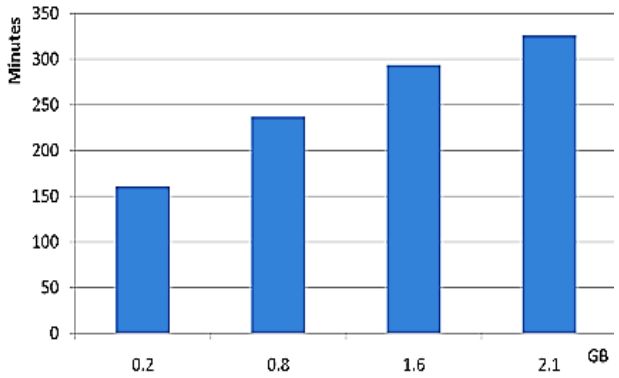

Fig. 4 Increasing Single CPU Runtime as Design Size Increases

Creating Runset

Finally we will discuss how we create the runsets and locate them for physical design engineers. We use MIMOS runset as an example. All runsets for both digital and analog are stored in the directory /pdk/master_lib/MIMOS.

Fig. 5 shows how to load runset and run DRC automatically using Calibre from Mentor Graphics. If the design is big, user can change from single-threaded to multi-threaded. Single thread performance is the amount of work completed by some software that runs as a single stream of instructions in a certain amount of time whereas multi thread is the ability of the software to manage multiple simultaneous requests without the need to have multiple copies of program running within the computer.

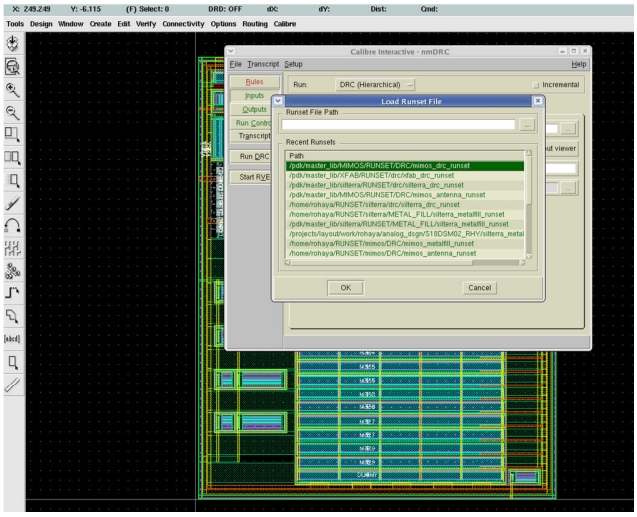

Fig. 5 Load DRC Runset at the directory/pdk/master_lib/MIMOS/RUNSET/DRC

Fig. 6 shows how to load runset to run LVS using Calibre. User needs to fill in input file in CDL format for the software to compare between schematic and layout. If there are any shorts or opens nets, it will inform user if there are any

discrepancies. Troubleshooting of any discrepancies needs to be done manually.

After DRC and LVS are clean, we proceed with parasitic extraction (XRC). Parasitic extraction is a process where parasitic resistors and capacitors are extracted from the layout. The goal of the extraction is to identify and characterize parasitic devices within a layout so that designers can simulate the behavior of the circuit drawn. After running XRC in Calibre, design engineers will run post simulation from the extraction files generated from this process. For XRC process with MIMOS foundry, there is a different between analog flow and digital layout flow.

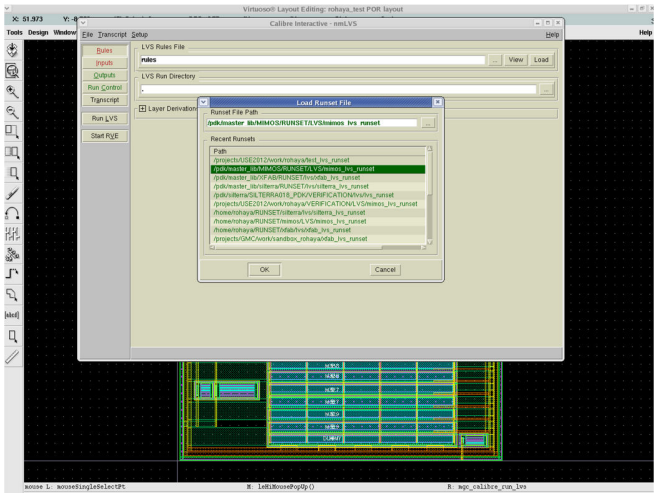

Fig. 6 Load LVS Runset at the directory/pdk/master_lib/MIMOS/RUNSET/LVS

Fig. 7 shows how to load XRC runset for an analog layout. After loading the runset, run XRC and the extraction files will be generated automatically. For digital layout, extraction is done using batch mode. spef file needs to be generated using batch mode. PEX_ADD is a file that contents all the necessary info before the script is run on UNIX command. An example of the flow to run XRC using digital flow is shown below:

1) Copy PEX_ADD and run_spef from /pdk/master_lib/MIMOS/RUNSET/RCX/DIGITAL/PEX_ADD to own directory.
2) Run run_spef script using below command.
3) % run_spef (run script in UNIX command).

After metal fill is inserted on the layout and verification had passed, the layout can be converted to GDS2 format for taped out and send to the foundry. Our automated flow is to ease the end users which are the physical design engineers. In today's demanding design environment, traditional physical verification after design closure is severely running out of steam, causing late-stage surprises and leading to an increasing number of time-consuming and error-prone manual fixes [4]. Using this automated flow, we are hoping to create an environment which is user friendly, hassle-free and fuss-free.

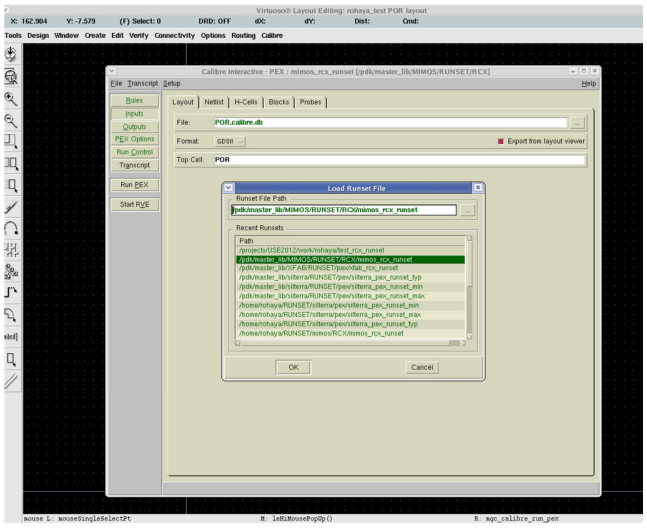

Fig. 7 Load XRC for Analog Runset at the directory /pdk/master_lib/MIMOS/RUNSET/RCX

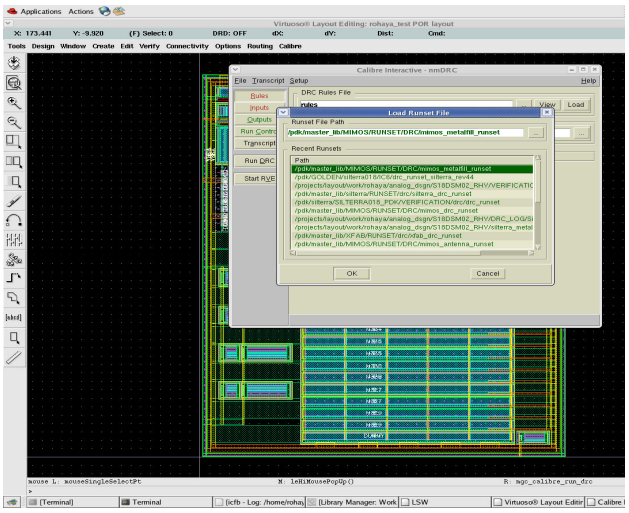

Fig. 8 Load Metal Fill Runset at the directory /pdk/master_lib/MIMOS/RUNSET/DRC

Fig. 9 shows the flow chart to summarize physical verification for a digital layout. The layout for digital design is generated from a place and route tool.

Fig. 10 shows the flow chart to summarize physical verification for an analog layout. The layout for analog design is a full custom layout drawn with reference from the schematic passed by design engineers. For the other two foundries which are Silterra and XFAB, physical layout engineers can load from /pdk/master_lib in RUNSET folders. Step by step flow can be read in README files.

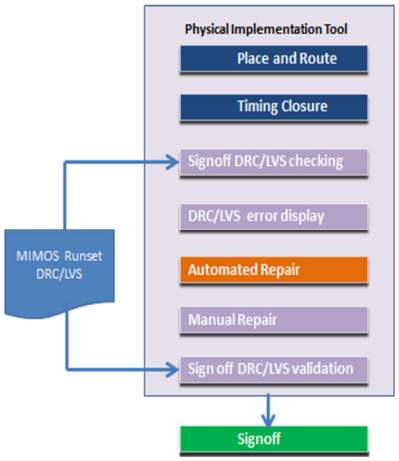

Fig. 9 Physical Verification flow for digital design

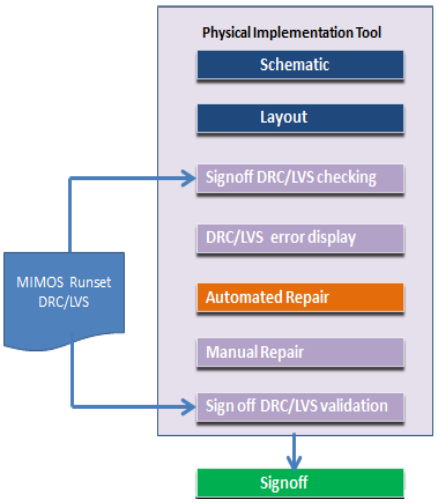

Fig. 10 Physical Verification flow for analog design

IV. Conclusion

As today's designs continue to get larger and more complex, schedules for physical designs are staying the same or even getting shorter [5]. This automation flow for verification will be a continuous project because rule files from foundries will be updated from time to time. Hence the runset might need some adjustments besides changes in startup files due to latest versions updated. Furthermore, as the technology advances, older version of the EDA tools will become obsolete and latest tools with new features to cater deep submicron technologies will be purchased. Runsets from different EDA tools might be different if we purchase different tool. Hence new structure needs to be developed from time to time. The utmost priority is to avoid late schedule for taped out if current flow is not fully automated. Also, we want to avoid rule files, source and setup files scattered in users /home folders.

Acknowledgment

Authors thank all the engineers in Physical Layout group who tested and gave feedbacks to improve the flow.

References

[1] Rahul Kapoor, Marilyn Adan and Louis Schaffer, "Achieving Optimal Performance Scalability for Physical Verification," *Synopsys, Inc. 2004.*
[2] Ahmed Arafa, Hend Wagieh, Rami Fathy, John Ferguson, Doug Morgan, Mohab Anis, Mohamed Dessouky, "Schematic-Driven Physical Verification: Fully Automated Solution for Analog IC design," Mentor Graphics Corporation, *IEEE 2012.*
[3] Elango Velayuthamt, "Accelerating Physical Verification with an In-Design Flow," *Synopsys, Inc. May 2009.*
[4] Paul Friedberg, "In-Design Physical Verification-Automatic DRC Repair (ADR)," *Synopsys, Inc. March 2011.*
[5] Kister, Steve, Minimizing The Time Needed To Complete a Hierarchical Design," *Synopsys, Inc. February 2011.*