iDRM:
Fixing the broken interface between design and manufacturing

Sage Design Automation, Inc.
Santa Clara, California, USA

Abstract

This paper reviews the industry practice of using the design rule manual (DRM) for documenting semiconductor manufacturing limitations and how these are translated to design rule check (DRC) decks. The fundamental flaws in the current paradigm are shown as well as the resulting negative impact on the industry. We will then describe iDRM, a brand new technology and new paradigm that eliminates these flaws and closes the loop between design and manufacturing. We will conclude with a few suggested applications of iDRM.

Industry background: The roles of DRM and DRC in interfacing manufacturing and design

Design Rules constitute the interface between semiconductor design and manufacturing. Design rules are written by fab process engineers and reflect the fabrication limitations for each drawn layer or combination of layers. Design rules are captured in the Design Rule Manual (DRM). The DRM is a book published by the foundry that represent a contract between the fab (or foundry) and the design house: if the designers keep all physical design features "legal" according to the DRM description, the resulting device can be fabricated successfully with good yield.

To fulfill the contract on the design end, designers have to make sure that physical design shapes adhere to the DRM rules, while trying to make the design as area efficient as possible. To verify that the design conforms to the rules, designers use Design Rule Check (DRC) tools. These are software tools that run a process-specific program called DRC deck or DRC runset. The DRC deck is a software code that implements checks based on the DRM rules. Unlike the DRM which uses descriptive form to define each design rule, the DRC deck is composed of long sequences of illegible low level programming code that manipulates the geometry in search of rule violations.

© 2013 Sage Design Automation

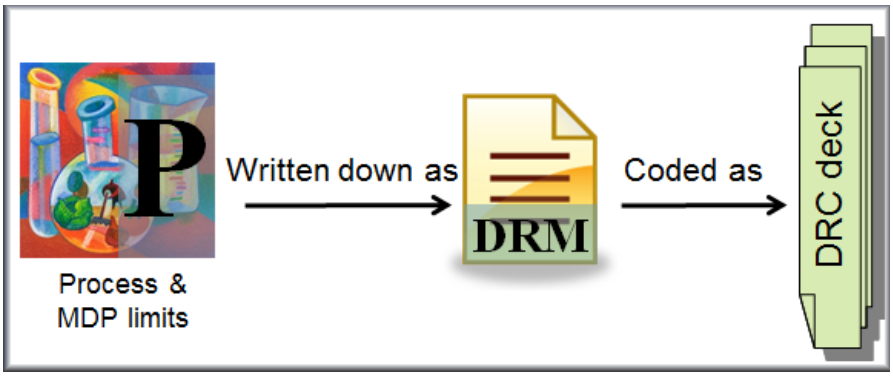

Figure 1: The roles of DRM and DRC deck

The DRM/DRC problem

The DRM/DRC pair paradigm has been in place since the 1980s. However in recent years it has been severely challenged and it currently suffers from fundamental issues that have a tremendous impact on process ramp-up time and effort, production schedules and overall product cost and yield.

1. The DRM: The problem starts with the DRM: It is a hardcopy book or a pdf file. The DRM is written by process folks who describe in free form English (or other language) the geometric limitation of shapes. Since it is free language, with no formal semantics and syntax, the rule descriptions are often unclear, incomplete and ambiguous.

This problem propagates to the DRC deck where it also becomes much more severe. The deck is programming code written manually based on the programmer's interpretation of the design rule description in the DRM book. The programmer reads each design rule and tries to program a sequence of low level geometry manipulations that attempt to identify violations of each design rule and flag them. This programming is done based on his or her subjective interpretation of the design rule description in the DRM, which is already unclear and ambiguous to begin with.

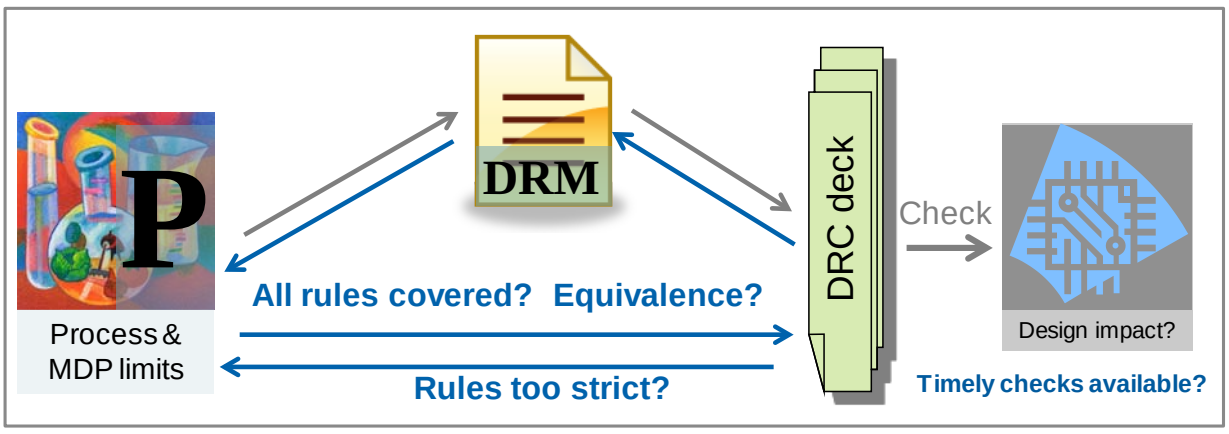

Figure 2: Validation problem between process/DRM/DRC-deck, and impact on design

Today there is no formal way to compare and verify that each piece of code in the DRC deck implementation fully and correctly represents the design rule intent it is based on. In fact, in the case of a new technology, the first versions of the DRC deck are known to have errors and inaccuracies.

© 2013 Sage Design Automation

Finding such errors and cleaning up the DRC deck code is done through an iterative trial and error process which takes a long time and effort, and even then few errors often remain after a DRC deck is released to customers. This long and winding road from process to DRC deck causes the impact on a design to be fully understood only after a too long time, possibly leading to low yield, re-designs and/or wasted area (if rules were interpreted too strictly).

2. Different viewpoints - different domains

The DRM and DRC deck represent two very different functions, view points and expertise domains. The DRM is a rule specification written by process people. It captures the process limitations of physical design and is usually described by drawings that depict specific layout patterns and allowed distances or other geometrical properties that correspond to them. It is descriptive in nature so that designers can understand what configurations and distances are legal so they know how to build their layout correctly.

By contrast, the DRC code is a program that manipulates the layout in an attempt to generate geometrical markers where the rules are violated. The DRC code has no notion of what is the allowed rule – it is focused on specific manipulations that will generate error marker polygons in specific locations where the programmer thinks such errors will manifest themselves.

The DRC programmer is a CAD person skilled in a specific DRC programming language that manipulates polygons. He or she does not know the original design rule intent and therefore can misinterpret it or miss some aspects of it. Because of tool or human limitations, in some cases it may be hard to program a complete and accurate check for a specific rule description, and the DRC code checks only an approximation of the rule, but not exactly the rule. Again, such compromises are common but hard to detect.

3. Process complexity

The most significant factor that has changed in recent years with advanced process technologies is the number and complexity of design rules. Advanced lithography limitation, OPC requirements, double patterning, new materials, new layers, and new device structures and constraints have all added new types of rules and complexities that have not existed before. Current process DRMs hold thousands of design rules and each rule may have multiple variables and conditions. In past technologies the DRM book had only a few hundred simple design rules, so implementing DRC checks for them was relatively straightforward. Today with DRMs that hold thousands of rules, and more than ten thousand variables and conditions, it is simply impossible. This tenuous process of figuring out intricate and ambiguous rule intents and writing complex code to check against them simply breaks down under the sheer number, complexity and ambiguity of the new design rules.

As a result, current DRC decks take a very long time to implement and debug, and it may take years before a DRC deck is complete and is solid.

© 2013 Sage Design Automation

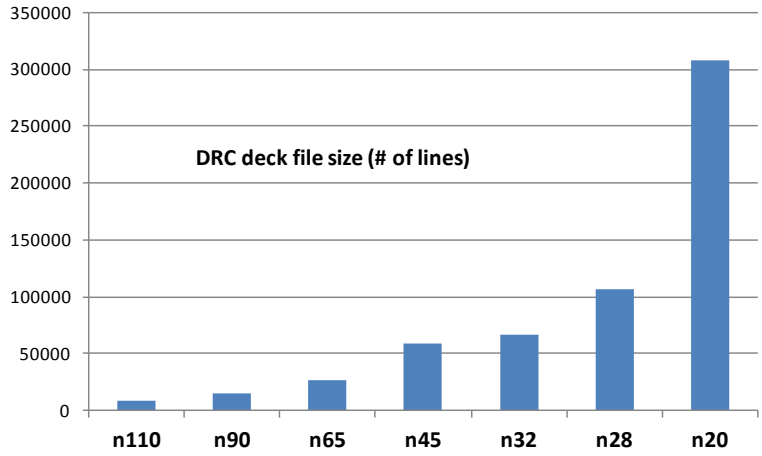

Figure 3: complexity of DRC deck vs. process node

Impact on the industry

Using the current broken paradigm comes with a significant cost and pain for the foundry and the entire semiconductor product chain:

- Many iterations between DRC deck and DRM: many man-years wasted at the foundries between process integration, PDK, CAD and customer support teams
- Delayed implementation of IP, library and first designs in a new technology: delaying first designs for design houses, and unnecessarily stretching out process ramp-up time for foundries.
- Each rule change or update in DRC deck takes very long to implement and verify: both in calendar time and man-years lost
- DRC decks that don't cover well manufacturing limitation and can end up in yield issues, costly re-spins and production delays
- Impacted silicon area and cost due to overdesign: physical design exceeds the required rules since complex rules were implemented inaccurately

The desired solution:

A new paradigm: Formal, graphical and executable design rule capture, which establishes a direct and verifiable interface between process limitations and design checks.

An effective and lasting solution to these problems should offer the following qualities and capabilities:

- Design rules should be entered and captured consistently in an environment understood and easily used by process engineers.
- The captured description should be formal, clear, complete and unambiguous. This will resolve any problems caused by communication issues between the two worlds of DRM development and DRC deck development.

© 2013 Sage Design Automation

- Graphical: The design rule description should use graphical capture where possible. This will enhance user friendliness and will allow non programmers to describe the rules in a succinct and accurate way
- Executable: The design rule description should generate a correct-by-construction rule check, putting an end to subjective interpretation and complex, error-prone programming. This allows rule developers to immediately try their definition of the rule on test layout and verify it against the rule intent. It shortens the iteration loop and since all is done by the same person, errors are avoided.

For this solution to succeed it must start at the source: design rule definition and capture.
Sage-DA provides such a solution with its iDRM (integrated Design Rule Management) system.

iDRM technology primer

iDRM is a design rule compiler technology that offers the following unique capabilities:

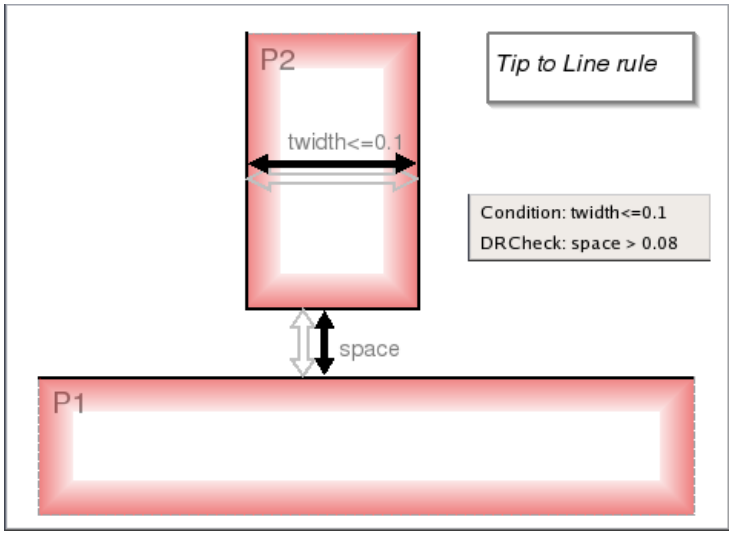

Figure 4: Tip to line space rule in iDRM

a. Capture design rules: Using the graphical Design Rule Editor, users draw the layout topology or pattern that describes the design rule. The drawing is simple and intuitive and can be made from scratch or jumpstarted by clipping a snippet from existing layout. Users can add arrows marking proximities and distances between shapes, edges or corners, and assign parameter names to them. e.g. *spacing*, *width1*, *poly-pitch*, etc.

Users can then add logical expressions and conditions, using these parameters as variables, to express statements that further qualify the topologies or patterns that are associated with a design rule. For example in a tip to line rule they can set a condition that this specific rule only applies if the tip width <= 100nm, and for all such cases define a rule that the space should be bigger than 80nm. (see Fig 4) .

© 2013 Sage Design Automation

These expressions use simple mathematical and logical operators that are self explanatory. This is all performed using a friendly GUI system and the description is clear, visual, unambiguous, very readable and yet complete and formal.

Executable: (here comes the magic part!)

Once a rule has been captured it serves not only as a clear visual description and documentation, but it also becomes an executable program for doing the following:

b. Manufacturing side: Validation of design rule intent

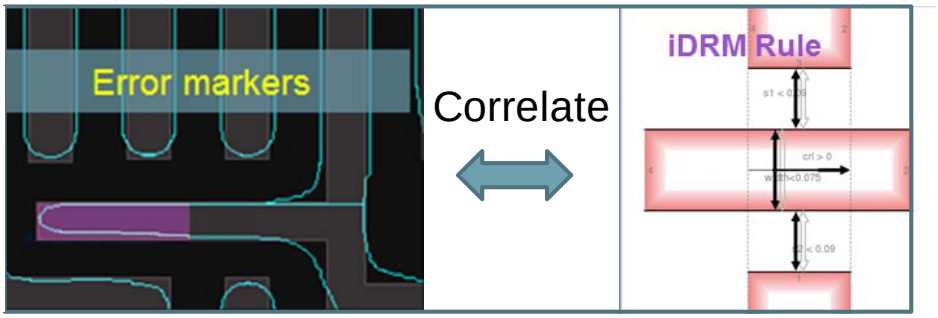

Figure 5: Correlating iDRM rule with imported fabrication/litho failure data

Process engineers can execute (run) the rule description they just captured on test layout examples that were used to identify and characterize specific process limitations. The tool will scan these examples, will find all similar patterns to the one that was captured in the design rule description and for each instance will measure all the parameters used in that definition. Based on the logical expression in the rule, the tool will determine a pass or fail result for each instance. Process engineers can compare and correlate the iDRM pass/fail indications with the actual process induced issues, e.g. lithography hotspots, and verify that their iDRM rule description is accurate and completely aligned with the actual physical process related issues. If the description is not exactly aligned, i.e. the iDRM run missed a few violations or flagged a few extra false violations; the process engineers can quickly modify the statements or fine-tune the parameter values in the rule description. This is done interactively and quickly until an accurate rule description is reached. iDRM has a specific *rule correlation* feature that provides automatic comparison between its own results and hot-spot (or other violation) markers and also helps in figuring out what needs to be adjusted. Using iDRM rule capture, process engineers can immediately verify that the design rule they entered matches accurately with the process limitation it is trying to capture.

© 2013 Sage Design Automation

c. Design side: validation of DRC deck using pass/fail QA test structures

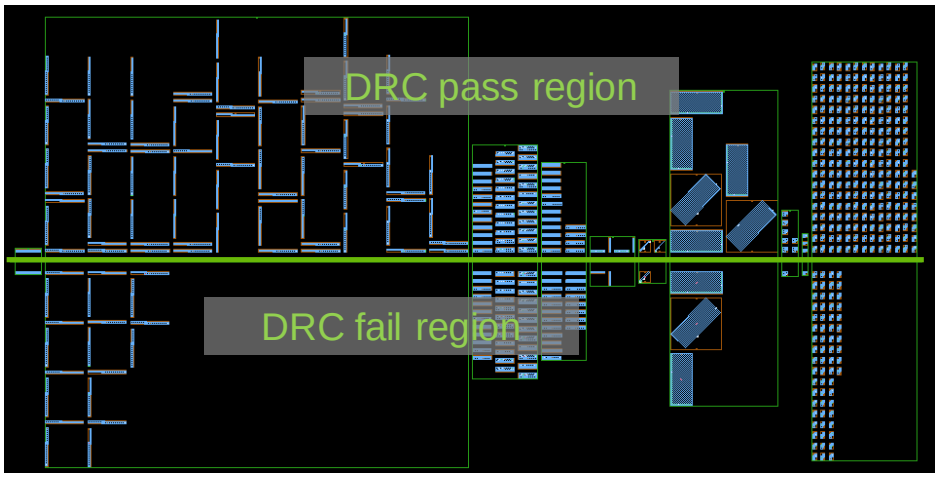

Figure 6: DRC deck QA test patterns generated by iDRM

iDRM can automatically generate QA pass/fail test patterns for each captured rule. These QA structures are then used to verify the correctness and completeness of the DRC code of a third party tool DRC deck. Using the drawn topology and the variables associated with the design rule expression, the tool will toggle pass/fail values for each parameter and logical condition and create pattern variations for different sets of values for each variable or parameter. Using this feature enables automatic generation of large sets of QA test structure with maximum coverage, a task that when done manually takes a very long time, a great deal of polygon pushing effort and in most cases cannot provide the degree of coverage that this automatic approach does. Since the test generation and pass/fail tagging is based on the source captured rule definition, it is complete, accurate and consistent.

d. Physical Design Scan (PD-Scan): Data-mining and analysis of physical design.

Users can run iDRM executables on any existing physical design, IP or test layout. The tool will scan through the layout and will look for all instances in the layout that use a similar pattern to the one captured in this design rule and take all the relevant measurements of the parameters (variables) that were used in the rule definition. The result of this scan is a complete list of all such instances, each with complete information of all the relevant measurements, orientations, etc.

The PD-Scan can be considered as a superset of a DRC check, as it finds all relevant patterns, measures all of them, and provides full geometrical information on each one. PD-Scan can output the results in tables or various graph formats (e.g. 1d histograms, 2d graph, Pareto charts, etc.) The iDRM integrated layout viewer provides a one-click hop from each table or graph entry to the layout location where this specific instance is found.

© 2013 Sage Design Automation

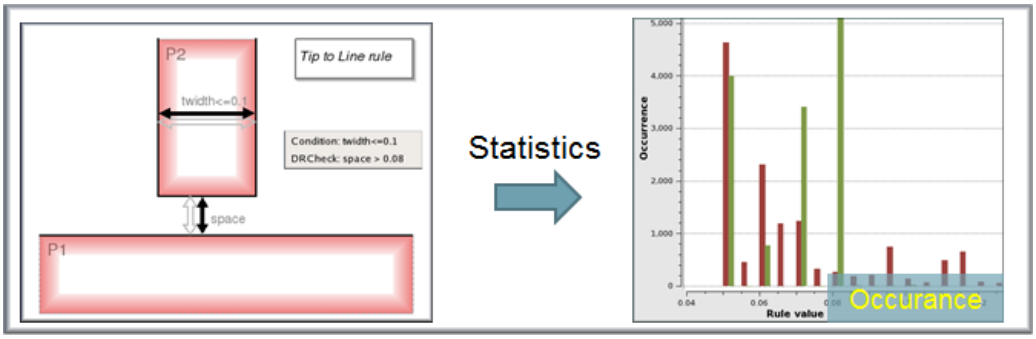

Figure 7: Data mining generating occurrence graphs

e. Future extension: DRC check synthesis

iDRM is a design rule compiler: once a design rule has been captured, it already is an executable object that can be run on test data to verify rule intent. In the near future this capability will also be extended to enable automatic synthesis of DRC code for third party commercial DRC programs. [This feature is not available yet]

Summary

The iDRM technology platform closes the loop between manufacturing and design and provides a much needed solution to the currently broken physical verification paradigm.

With iDRM, process engineers can capture process limitation with a clear and formal DRM description; they can validate it against yield issues in actual or simulated test layout and immediately verify that the description accurately represents the process limitations. They can also run the executables on existing designs and find possibly interesting cases of these design rules they may want to further investigate and characterize. CAD engineers that write DRC deck programs are now able to use a clear and formal rule definition and automatically generate QA pass/fail test cases for their code that are correct and consistent with the rule definition, and verify the DRC code against them. In the near future, engineers will be able to also synthesize a correct-by-construction third party DRC program from the iDRM rule description.

© 2013 Sage Design Automation