
An Error Comparison Scheme for Design Rule Checking Flows

**Dibyendu Goswami,
Swami Gangadharan**



Agenda

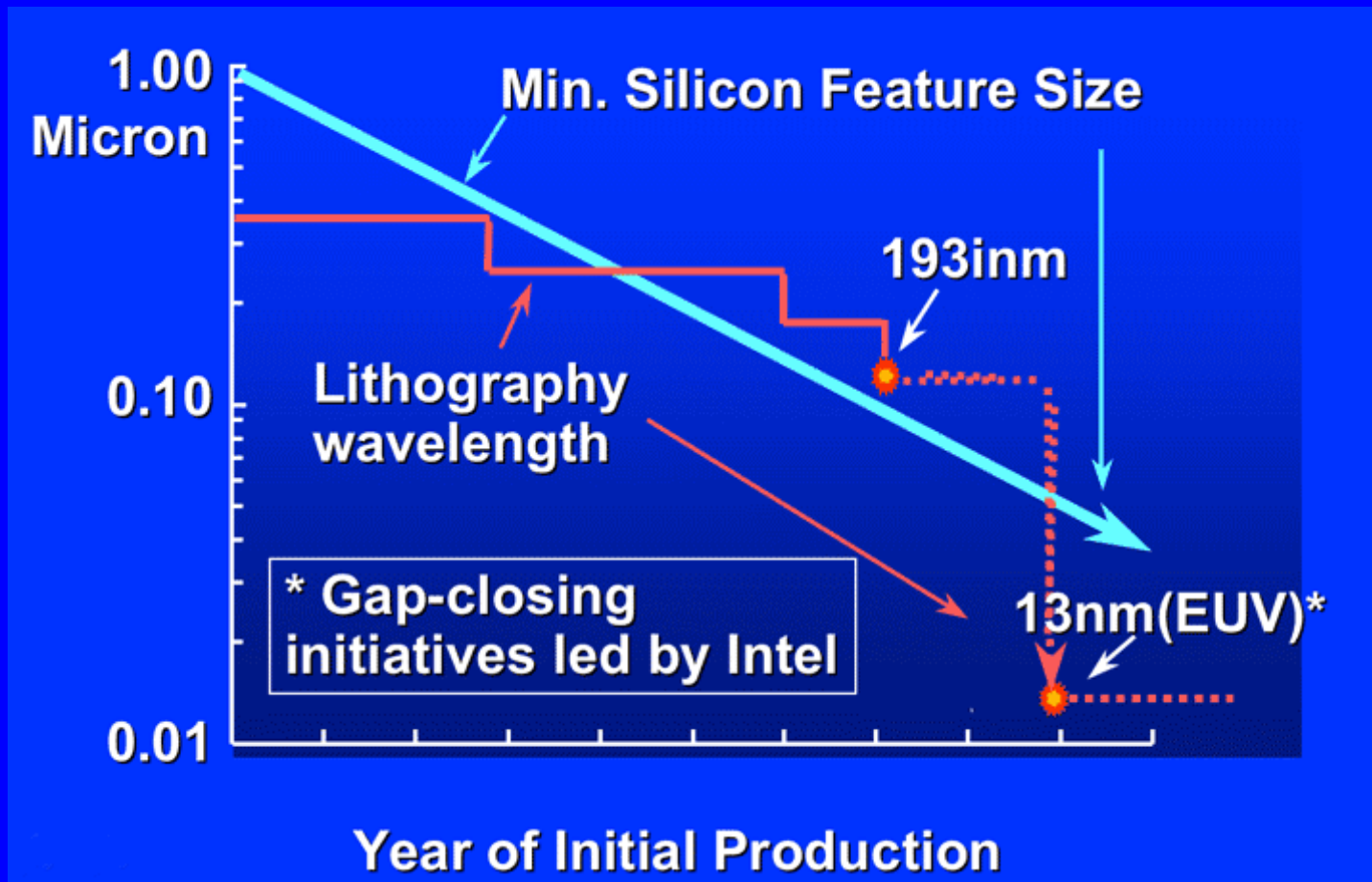
- Introduction
- Why DRC Error Comparison?
- Layout Verification Tools
- DRC Error Comparison Scheme:
 - Polygon-based tool
 - Edge-based tool
 - Polygon-based tool and Edge-based tool
- Results
- Conclusion

Introduction

- **DRC stands for Design Rule Checks**
- **The number of design rules to be checked on a design has grown multifold**
- **Sub-wavelength Lithography: contributes to complex design rules:**
 - **OPC Rules**
 - **PSM Rules**
 - **DFM Rules**

Introduction, Contd..

- Initiatives to close Litho Gap:



Why DRC Error Comparison?

- To cope with ever increasing DRC complexity EDA vendors provide faster and more accurate layout verification solutions
- Thorough validation of the new tool version with respect to the POR (Plan Of Record) tool version is a must
- “Error found, error missed discrepancy” – nightmare to the physical verification engineer
- What if the tool is brand new?
- “Error number count” method is misleading
- Analyzing the mismatches can be very tedious in manual approach

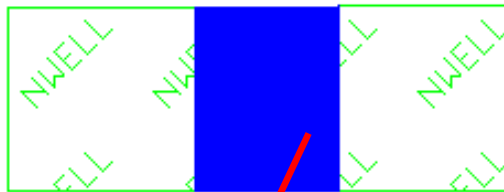
Layout Verification Tools

Two classes of layout verification tools:

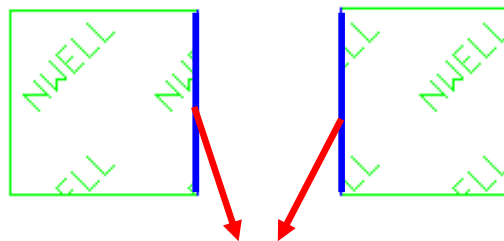
- 1. Polygon-based tools: Entire polygons are considered during dimensional rule checks. Example: Hercules from Synopsys**
- 2. Edge-based tool: edges of polygons are considered during dimensional rule checks. Example: Calibre from Mentor Graphics**

Layout Verification Tools

- **Error Reporting:**



Error marker by polygon-based tool



Error markers by edge-based tool

DRC Error Comparison Scheme

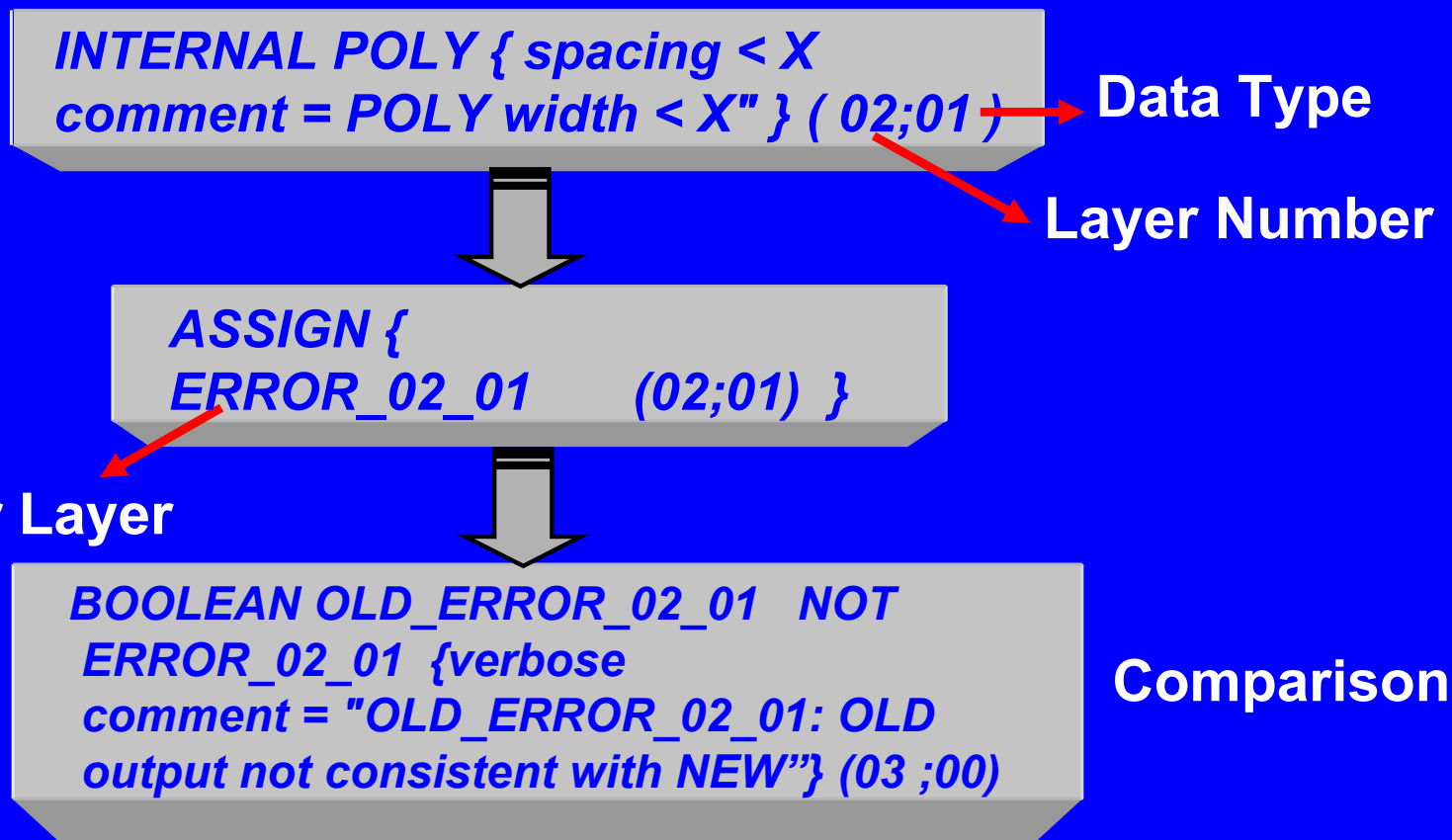
- **Two Steps for Comparison:**
 - **Creation of Error DBs**
 - **Comparison of Error DBs**
- **Four Types of Comparison:**
 1. **Two versions of a polygon-based tool**
 2. **Two versions of an edge-based tool**
 3. **Two runsets of a polygon/edge-based tool**
 4. **Polygon-based tool and Edge-based tool**
- **Note: “runset” – DRC deck understood by LV Tools**

Two versions of polygon-based tool

- Error DBs are created for two versions
- Dynamic creation of 'ASSIGN' file by parsing the input runset
 - Concatenation of data type and layer number to create an unique error layer
- Two error DBS are superimposed – Comparison error is flagged if there is a mismatch
- On-the-fly generation of “auto-compare” runset based on the input runsets
- Auto-compare runsets retain individual identity of each “error marker”

Two versions of polygon-based tool

- Creating Unique Error Layer and Comparison:

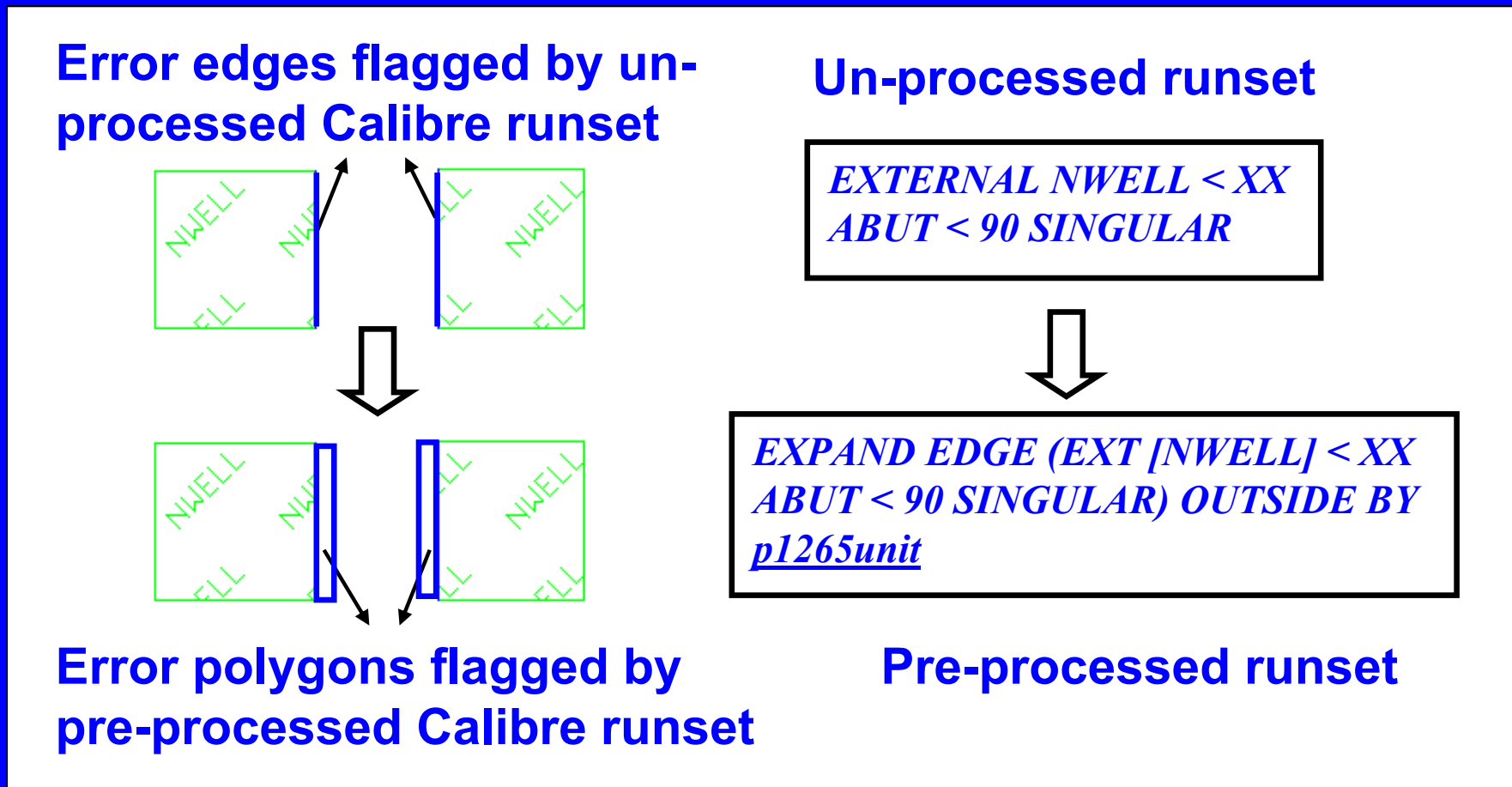


Two versions edge-based tool

- Errors DBs are created for two versions
- Pre-processing of input runset to expand Error Edges by “tiny amount” into Error Polygons
 - “Tiny amount”: $1/5^{\text{th}}$ or $1/10^{\text{th}}$ of the manufacturing grid
- Special technique to distinguish between error layers from two different versions: *Layer Bump*
- Superimpose Error DBs
 - Error flagged in case of mismatch
- Auto-compare runsets generated on-the-fly retain individual identity of each “error marker”

Two versions edge-based tool

- Converting error edges into error polygons



Two runsets of a polygon/edge-based tool

- Alternative runset implementing may be required for the same set of design rule checks
- “runset comparison” is done by running the same tool on the same input database using two different runsets being compared
 - Runset pre-processing for edge based tool
- Error DBs are superimposed and error is flagged in case of a mismatch

Polygon-based tool and Edge-based tool

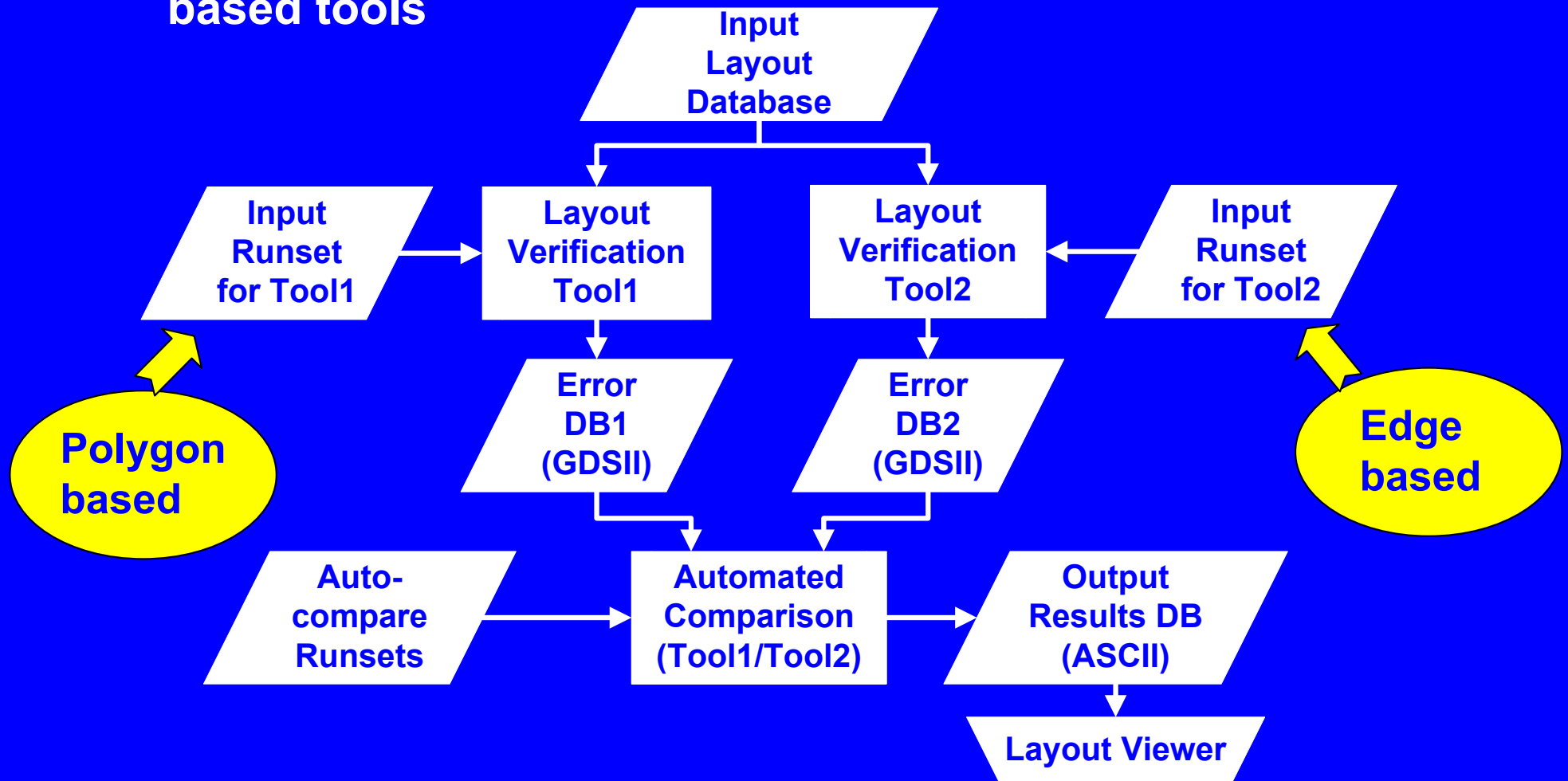
- Most Challenging among all comparisons
- Superficial differences between the error markers generated by the tools
- False mismatches to be avoided:
 1. For a given rule check, both tools write out the errors into a common layer in terms of GDSII layer number and data-type
 2. Runset pre-processing for edge based tool
 3. No Mismatch: For a given rule check, the error marker generated by one tool “interacts” with the other in some form
 4. Mismatch: No interaction between the error markers

Polygon-based tool and Edge-based tool

- Large “Dirty” database is required for robust comparison
- Manual comparison is practically impossible
- Key aspects in our approach:
 - Locations of the mismatches can be browsed using one of the LV tools
 - Very useful for large testcases
 - From the final error comment it is very easy to realize which tool is causing the mismatch
 - Simpler debugging by back-tracking the error markers

Polygon-based tool and Edge-based tool

- DRC error comparison scheme between polygon and edge-based tools



Polygon-based tool and Edge-based tool

- Layer identification in auto-compare runset in Calibre

```
INTERNAL POLY {  
spacing <XX  
comment = "(C1) POLY width < XX" } (  
02;20 )
```

```
LAYER MAP 02  
DATATYPE 20 7001  
LAYER HERC_ERROR_C1 7001
```

Hercules runset

```
C1 {  
@(C1) POLY width < XX  
INTERNAL POLY < XX }  
DRC CHECK MAP C1 GDSII 1 20
```

```
LAYOUT BUMP2 500  
LAYER MAP 02  
DATATYPE 20 1  
LAYER CAL_ERROR_C1 1
```

Calibre runset

Polygon-based tool and Edge-based tool

- Simpler debugging by back-tracking the error markers
- Comparing Calibre and Hercules Error markers in Calibre Auto-compare runset:

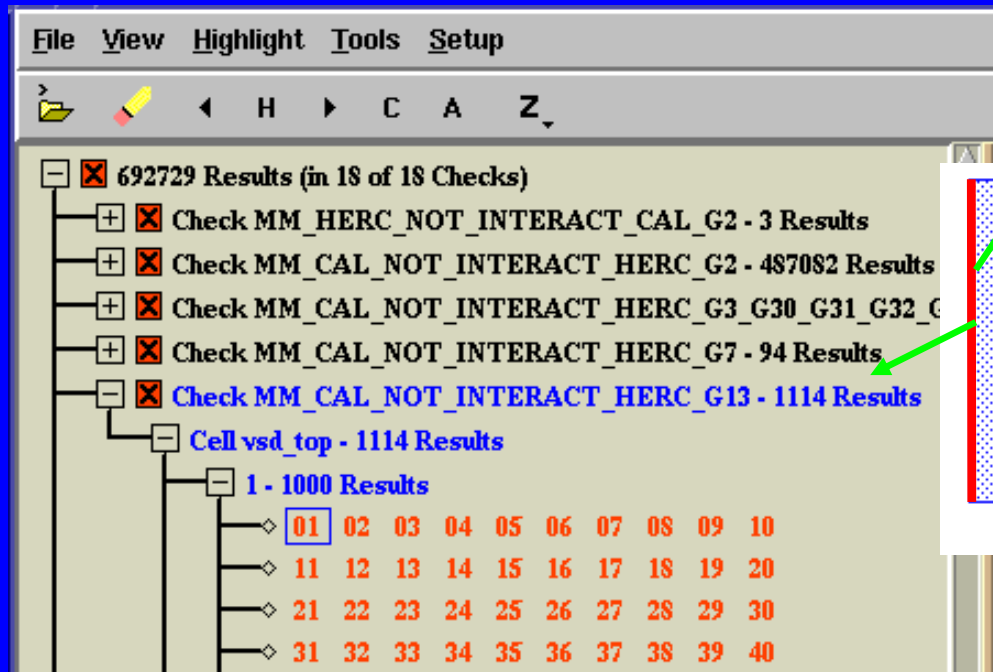
```
MM1_C1 = HERC_ERROR_C1 NOT INTERACT SINGULAR ALSO CAL_ERROR_C1  
MM2_C1 = CAL_ERROR_C1 NOT INTERACT SINGULAR ALSO HERC_ERROR_C1
```

```
MM_HERC_NOT_INTERACT_CAL_C1 {  
@comp_error_herc_not_cal_C1: herc DRC errors not reported by cal  
  COPY MM1_C1 } //Code to flag Hercules markers that don't match with Calibre
```

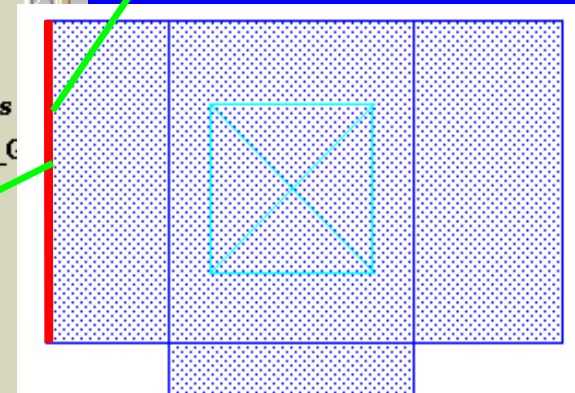
```
MM_CAL_NOT_INTERACT_HERC_C1 {  
@comp_error_cal_not_herc_C1: cal DRC errors not reported by herc  
  COPY MM2_C1 } //Code to flag Calibre markers that don't match with Hercules
```

Results

- **G13 rule (met enclosure of via):**
 - No of violations reported by Hercules: 7150
 - No of violations reported by Calibre: 8264
 - No of Calibre errors that don't match with Hercules violations: 1114

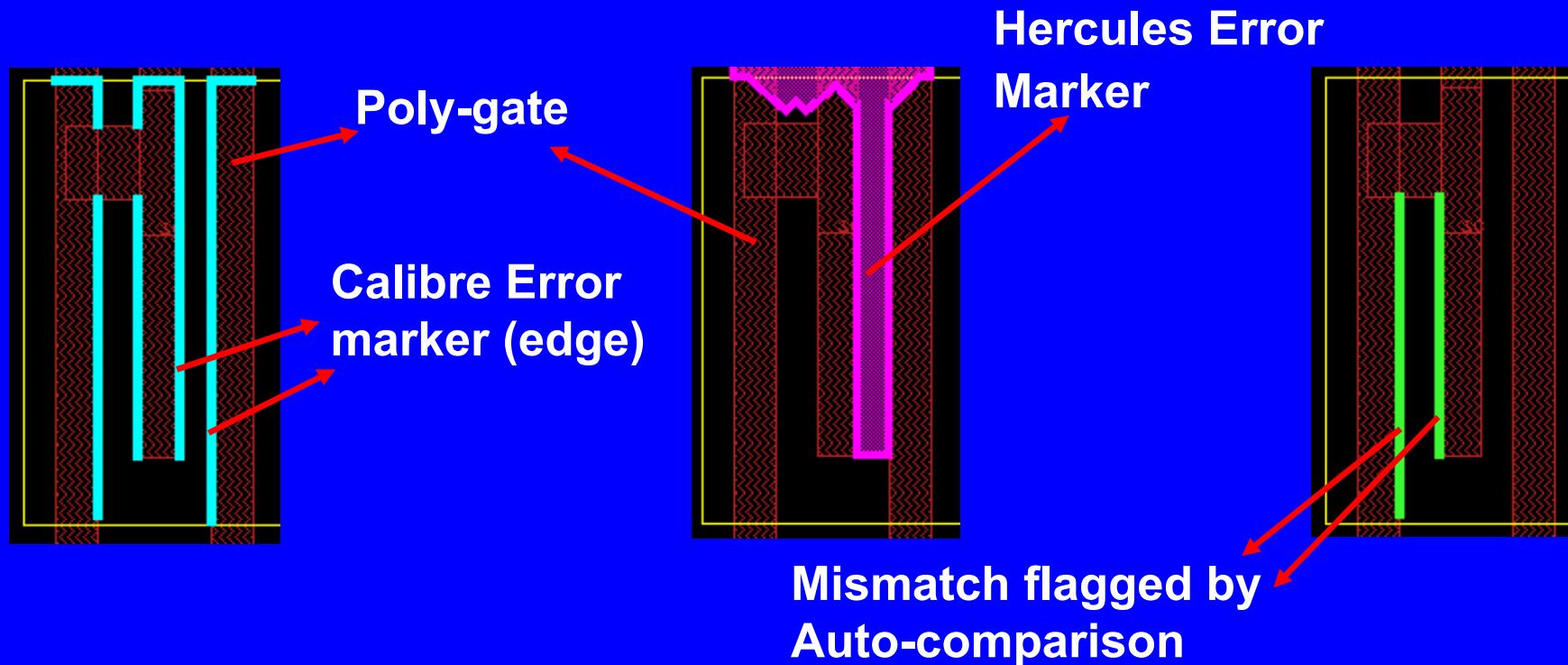


G13 Calibre error not matching with Hercules



Results

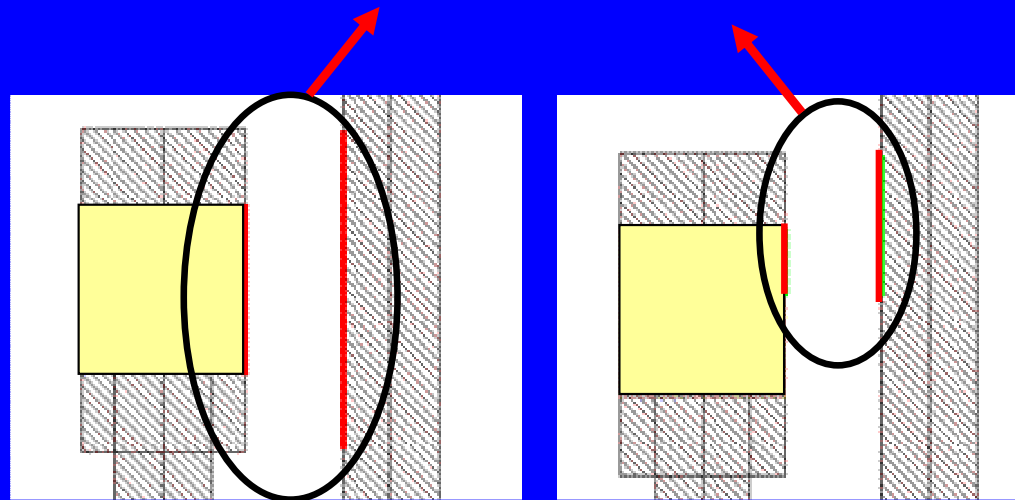
- Gate Spacing:
 - Errors are not flagged by Hercules (not a tool limitation, it's a sunset issue) for Gate edges that are part of the same polygon



Results

- via to metal spacing rule:
 - Mismatch in error count
 - Duplicate error reporting by Calibre

Error between the same edges reported twice by Calibre



Conclusion

- **Simplification of the task of comparing DRC errors between two tools/tool versions/runsets**
- **At least 5X productivity gain**
- **Generic concepts, can be deployed for the validation of any other layout verification tools across all process technologies**



Thank You

