



**JUNE 23-27, 2024**

MOSCONE WEST CENTER  
SAN FRANCISCO, CA, USA





JUNE 23-27, 2024

MOSCONE WEST CENTER  
SAN FRANCISCO, CA, USA

# Automated Design Scenario Extraction From A Large Design For Faster Debug Of Static Verification Tools

Sanjay Gulati, Gaurav Pratap, Sachin Bansal, Vishal Keswani

**SYNOPSYS®**

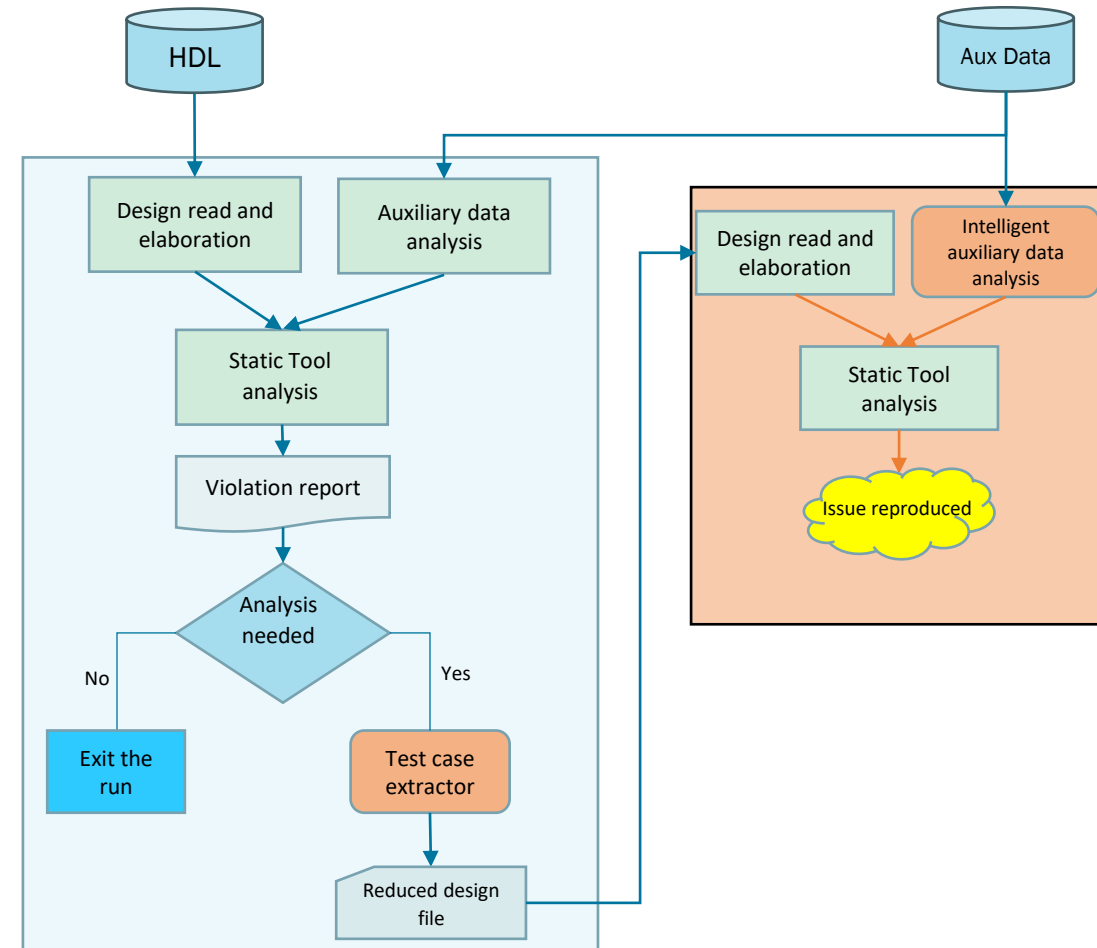


# Motivation

- Customers report issues related to false violation or missing violation, hang, crash etc. on their *large* designs
- During the sign-off stage, a quick fix is expected for any tool issue
- Finding the root cause of the issue and provide quick fix takes time as:-
  - R&D, AE may not have direct access to design
  - Shipping design to a secure network is takes time or may not be possible at all
  - High number of debug prints make it difficult to find root cause
  - Attaching debuggers on large design is cumbersome and slow
  - Debug fields in violations and other reports provide limited information about the design scenarios making it difficult to create a unit reproducer
- **Need an automated method to create a unit reproducer with only minimum gates and using existing constraints (UPF, others)**

# Testcase Extractor

- The Utility creates a small testcase extracted out from the original design with below properties:
  - The created testcase shows the same issue as reported by customer
  - No modification of auxiliary data (UPF/SDC ...) is required
  - The created testcase contains minimal number of design elements
  - It is an automated process which requires minimal human intervention
- Main Features of utility
  1. **Relevant design object identification** - Tool extracts it from debug fields in a false violation or from user input for a missing violation
  2. **Design tree extraction** - Based on the relevant design object and their type, this utility identifies all the design objects which need to be extracted for reproducer.
  3. **Dumping reduced design tree** - A custom design tree dumper dumps the extracted design tree in either plain Verilog or Binary dump
  4. **Intelligent readback of auxiliary data** - It ensures the original auxiliary data is read back seamlessly on the extracted reproducer with no errors



# Testcase Extractor

Viol Name: ISO\_VIOL\_TYPE1

Debug fields:

Debug 1

Debug 2

Debug 3

Debug 4

.....

.....

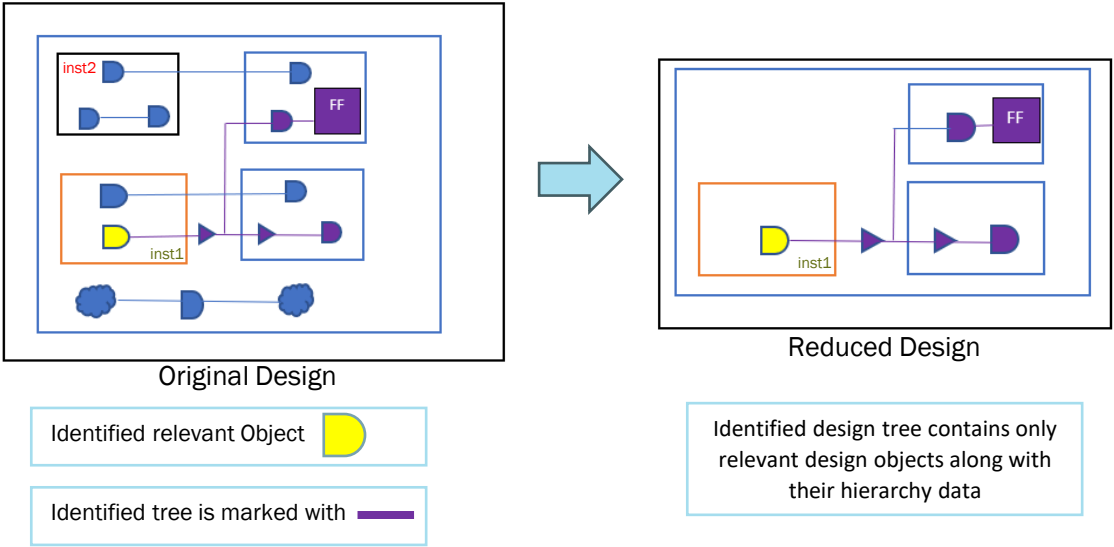
Debug n

Viol Name: ISO\_VIOL\_TYPE1

Debug field	Priority	Field Type
Debug 1	2	source domain
Debug 2	2	intermediate point
Debug 3	1	source object
Debug 4	3	sink object
.....		
.....		
Debug n	3	sink domain

Relevant debug field: Debug 3

Debug field type: source object



## Extract and Dump reduced design tree

```
for each object(obj) in relevant object list
{
    add obj node
    get the connected object list(conlist) from original design graph
    for each object(newObj) in conlist
    {
        if(newObj is not present) add newObj in relevant Object list
        connect the nodes of obj and newObj
    }
    remove obj from relevant object list
}
dump this reduced design graph
```

Intelligent readback of auxiliary data

```
create_power_domain TOP -include_scope
create_power_domain PD2 -elements {inst2}
create_power_domain PD1 -elements {inst1/U1}
set_retention RET1 -domain PD2 -elements {inst2/ff1}
    -retention_power_net VDD2_net -save_signal {save1 high}
    -restore_signal {res1 low }
set_isolation ISO -domain PD1 -elements { inst1/U1/out}
    -isolation_power_net VDD_net -isolation_ground_net VSS
    -isolation_signal Enable2 -location self
    -isolation_sense high -clamp_value 0
```

Design object removed from the original design.

Errors will not be flagged for these objects

Design object present in the reduced design.

UPF attributes will be annotated on them

# Results

## Design size reduction

Design name	Type of violation	Flat Instance count(Original)	Flat Instance count(Reproducer)
Design 1(Memory Design)	ISO_VIOL_TYPE1 (Electrical violation)	945,128	16(~0%)
Design 2(Server Design)	ISO_VIOL_TYPE2 (Electrical violation)	270,987,823	681(~0%)
Design3 (Modem Design)	PSW_VIOL_TYPE1 (Functional violation)	82,481,822	1,389(~0%)

## Run time gain

Design name	Type of violation	Run Time in seconds (Original)	Run Time in seconds (Reproducer)
Design 1(Memory Design)	ISO_VIOL_TYPE1 (Electrical violation)	112	28(4X)
Design 2(Server Design)	ISO_VIOL_TYPE2 (Electrical violation)	17340	196(88.5X)
Design3 (Modem Design)	PSW_VIOL_TYPE1 (Functional violation)	22610	1148(19.6X)

# Summary

- Quick debug of customer reported issues are possible
- Utility is compatible with all the design stages(RTL/Netlist/PG-Netlist)
- Design integrity is maintained by dumping extracted design in binary format
- Utility can help with enhancing **regression coverage for EDA tools**
- **Future work:**
  - To get better run time reduction, auxiliary data reduction will be explored
  - AI models can be introduced to dump unique Netlist/RTL which replicate user scenario, but different from the user design

# Q&A

# Thank You!