Autoliblist: Synergize DV–PD Flow for Increased Efficiency of Gate Level Simulations

Ruchika Sapra, Staff Engineer, Qualcomm India Pvt Ltd, Bangalore, India
(*ruchikas@qti.qualcomm.com*)

Abstract—Gate Level Simulations (GLS) are an integral part of the chip design verification flow, for a successful first pass silicon. With increased challenges of IP designs i.e technology nodes, multiple projects, external dependency on netlists, analog-digital interactions, GLS timelines are always a big bottleneck to converge with limited time and resource constraints.

We developed and deployed a stress free, push-button solution to reduce the time lines of the GLS flow with much better quality in the DV world. Typically it takes about 5-8 iterations before the PD netlist is finalized. It is a nightmare for the DV guy to find the deltas in the iterative netlist and get the setup working again. The deltas are a function of cell addition/deletion, new library versions, different cells with varying drive strengths, different cell library being used , zero delay , sdf based runs etc .

Keywords—GLS, PD, Standard Cell libraries, Netlist

I. INTRODUCTION

Gate level simulations (GLS) are an essential step before going into the last stage of chip manufacturing. It can help verify dynamic circuit behavior, which cannot be verified accurately by static methods. They also give confidence in verification of low power structures, absent in (Register transfer Level) RTL verification and added during synthesis. GLS has to pass through various stages before sign-off and serves to check both functionality and timing. The setup of GLS starts off when the prelim netlist is released. Since this netlist is prone to functional and timing bugs, the GLS brings up user selected functional tests with zero/unit delay simulations. This helps in setting up the flow for GLS and confirm that the netlist is properly hooked up. This paper talks in detail about the methodology to reduce the multiple iterations between netlist, liblist versions thereby reducing the overall turn-around time. We introduce some level of automation here to resolve the iteration's. This paper talks about automating the DV flow for GLS by categorizing and automating the leaf netlist cells selection.

II. MOTIVATION:: PROBLEM STATEMENT/CHALLENGES

Simulations are an important part of the verification cycle in the process of hardware designing. Gate level simulations are a very significant and important step to ensure the quality of netlist.

GLS simulations are run to gain confidence on the design footprint post addition of DFT structures, power ground pins,clock domain crossing syncronizers. Aim of GLS to verify the power up and reset operation of the design and also to check that the design does not have any unintentional dependencies on initial conditions. To give confidence in verification of low power structures, absent in RTL and added during synthesis. GLS is also performed to catch multicycle paths and do Power estimation is done on netlist for the power numbers. It also aims to verify DFT structures eg scan chains, atpg patterns, tester patterns. Glitch detections across various PVT corners, need for synchronizers.

Gate-level simulations are done at various stages of netlist as given below:

1. zero-delay netlist after logic (standard cells) synthesis (Z-delaynetlist)
2. netlist generated after place and route (PnR) clock tree insertion (CTS) (PnRnetist)
3. final netlist with static timing closure and SDF (Standard Delay Format) annotation (TOnetlist)

GLS setup differs from the RTL simulation setup due to handling of additional ports (mostly DFT), Power related cells, input delays and all cells/macros have their model available. Also, PnR/TOnetlist ports are bit-blasted which requires additional setup up modification. Every verification engineer doing GLS spends a week or two getting the setup working and is error-prone if done manually. The major challenge being the initial compile of the environment. With a charter to improve the efficiency and efficacy of the GLS setup, we attempted to address the pain points in GLS and accelerate the verification process.

Most of the times, it is observed that the cells used by the netlist folks did not match the cells available with the verification folks. The verification team initially has access to Z-delaynetlist library only. Since the libraries

keep evolving during the course of the project, the cells used in Zero-delay netlist and the PnR netlist/TO netlist differ. There is a communication gap leading to considerable time being wasted in getting the PnR/TOnetlist to compile. For eg, During ECO verification – eco_filler cells are introduced in the PnR/TOnetlist, though the remaining cells are Z-delaynetlist. Similarly power aware related cells are introduced which need an update in PnR/TOnetlist.

Fig.1 Netlist

This requires modification to existing design verification (DV) framework to support multistage GLS. This was achieved by proposing a new methodology to create liblists required at each stage of GLS. We rolled out this methodology within our organization using our home-grown auto-liblist tool

III. OBJECTIVE & IMPLEMENTATION

With advancements in project timelines and maturity of netlists, the DV infrastructure iteratively increments to the need of the project deliverables. The evolution of the DV environment to GLS environment is depicted in Fig 2.

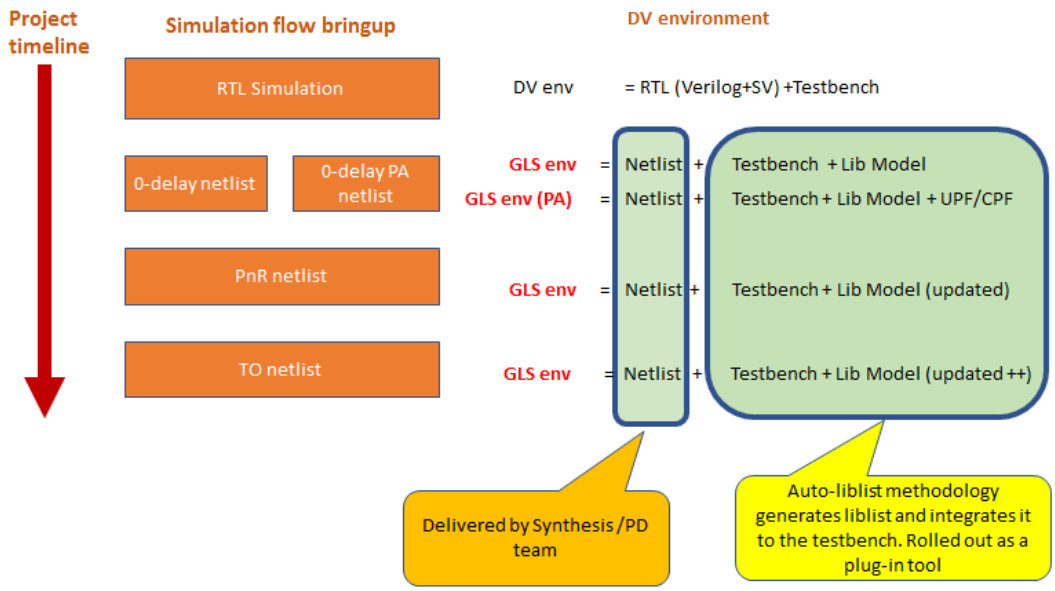

Fig.2: Evolution of the DV environment

Every RTL design comprises of cells – memory cells, CDC cells, standard cells, custom cells from technology specific nodes. To meet synthesis closure, Designers use cells oblivious to the PD requirements. As the designers handover the design to the back end PD teams, new constraints get added, apart from the PnR (Placement and Route) delays and clock tree insertion delays.

We studied all the cells being used in the design and could bucketize the cells as design specific and PD specific (refer Fig3). Memory cells, cdc cells, technology cells were found to be design specific and would not be changed during backend processing.

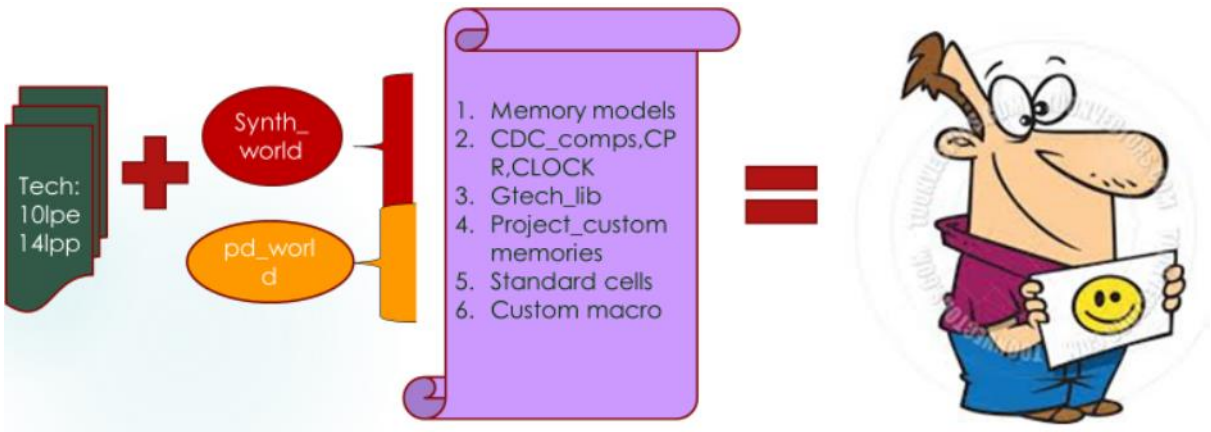

Fig.3: Implementation

Standard cells and Custom memory cells would undergo over period of the design cycle with respect to the layouts, parasitic extraction model, DRC rule deck etc. These cells would be constantly updated to fine tune the final GDSII blueprint to match the ip/chip operating numbers at all PVT conditions. These cells also contained the annotated values. Let's call these the variable cells.

We further divided our variable cells into PLL based, reg_arrays, sensor cells (cpr cells), level shifter cells etc. Once the classification of the cells was streamlined, we started to focus on the database which was used by both the fixed and variable cells.

We worked with the designers and the PD team to provide us a labeled tag for the cell libraries used with every netlist drop. This helped us to overcome the issues of drive strength mismatch, annotated delays mismatch etc. Also, this helped as the dependency on the PD engineer reduced and the communication was structured and correct.

For the final library creation from both the worlds – RTL world (including synthesis) and the Physical Design.

IV. RESULTS WIH PROPOSED METHODOLOGY

This was deployed on two different product lines and helped us find issues in a new netlist drop in significantly less amount of time. Missing cells were detected within 2 hours which usually took 2-4 days in the past depending on the communication. In another instance the drive strength mismatch did not allow the compile to go through, such issues were uncovered in the first pass. We did a left shift where we could focus on real GLS problems of booting and initialization.

As the netlist matured, the number of cells getting changed and the lib issues reduced. The success of this tool lies in getting to 0 lib issues as is evident from the tabe .

Project Details			Cells Added %	Cells Changed %	Lib issue (old Methodology)	Lib issue (proposed methodology)	GLS Setup Time (old)	GLS Setup time(new)
PRODUCT A	Project 1	Zero Delay	100%	100%	70	4	2weeks	4hours
		Post PnR	30%	20%	40	2	1week	2hours
		SDF Annotation	5%	10%	10	1	1week	1 hour
	Project 2 (Derivative , Tech node changed)							
		Zero Delay	100%	100%	58	0	2weeks	3hours
		Post PnR	25%	18%	36	0	1week	1hour
		SDF Annotation	5%	10%	6	0	0.6 week	1hour
PRODUCT B	Project 3							
		Zero Delay	100%	100%	100	0	3weeks	8 hours
		Post PnR	35%	15%	57	1	1.5week	3 hours
		SDF Annotation	5%	8%	38	0	1week	2hours
	Project 4 (Derivative , Tech node changed)							
		Zero Delay	100%	100%	45	3	2weeks	4 hours
		Post PnR	28%	15%	22	0	1.5week	1 hours
		SDF Annotation	4%	5%	17	1	1week	0.75hours

Table.1 Results

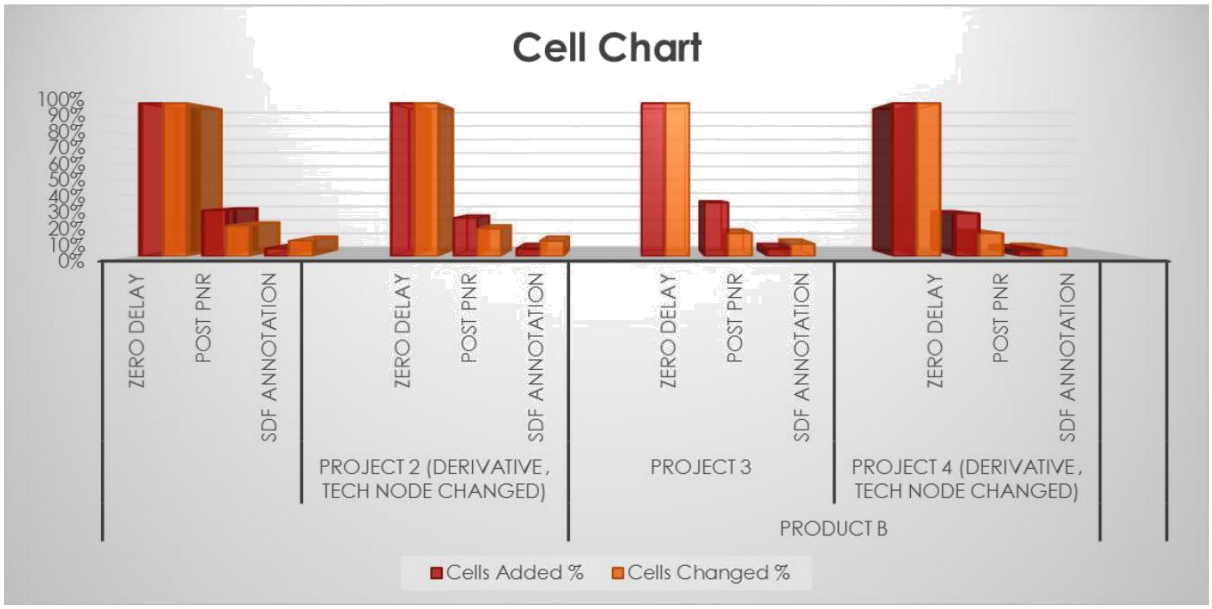

Table.2: Cell Chart

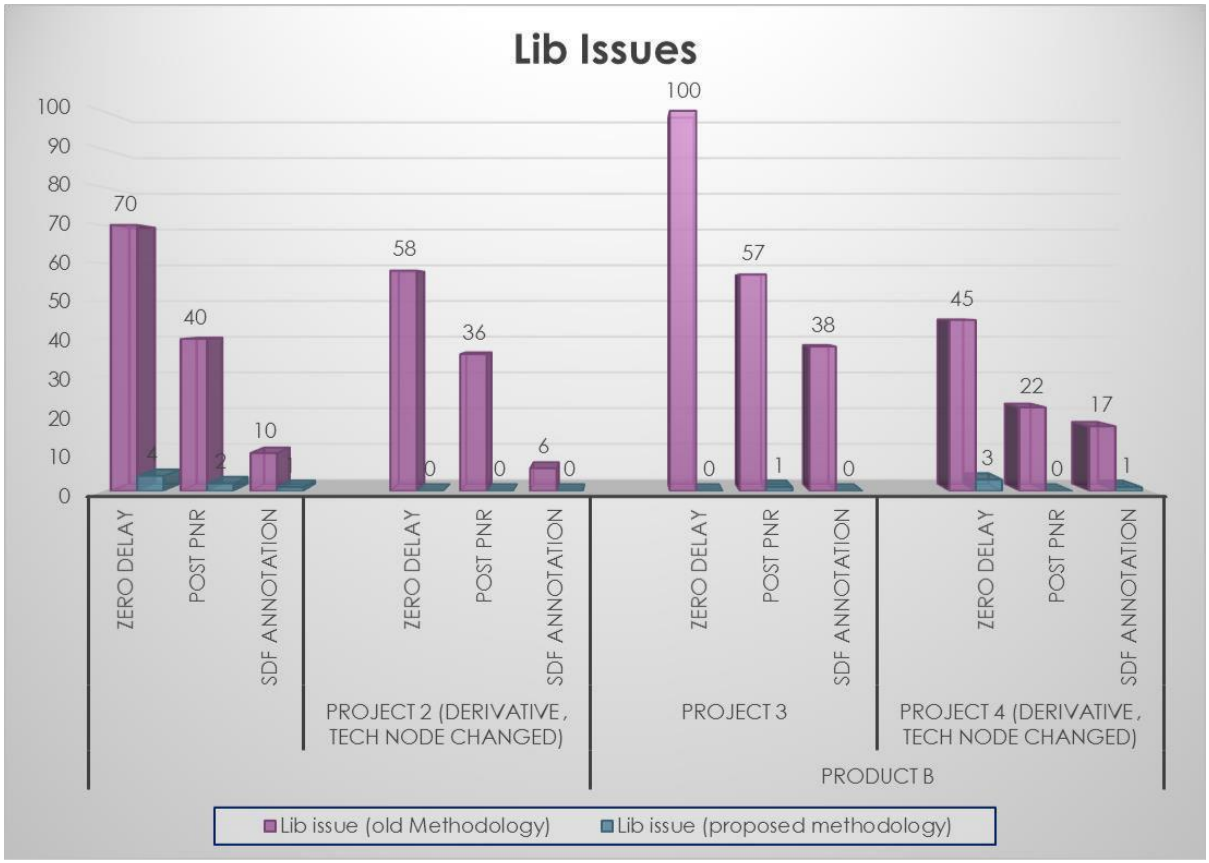

Table.3: Lib Issues

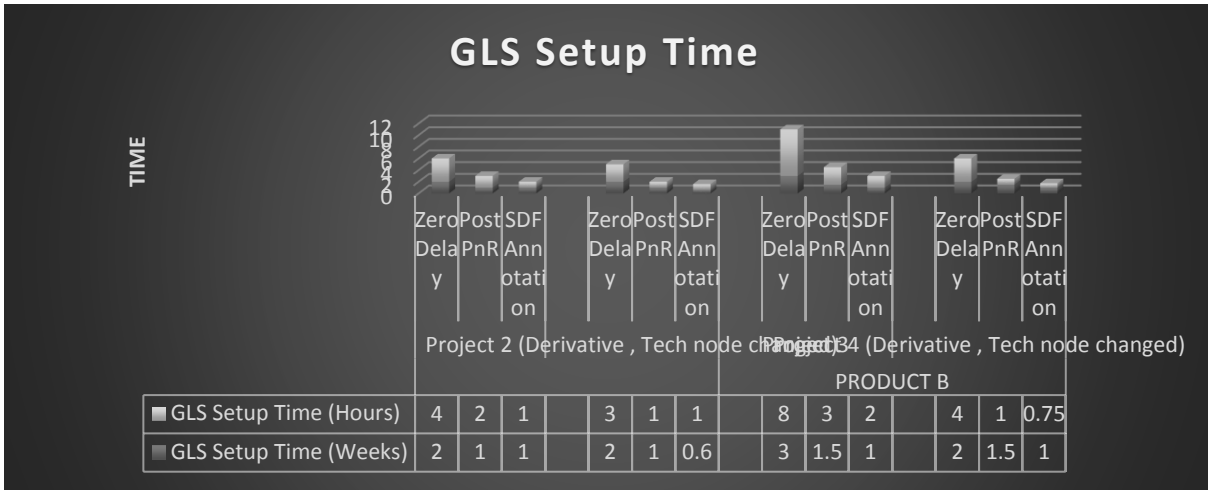

Table.4: GLS SetupTime

V. DEPLOYMENT OF PROPOSED METHODOLOGY & CUSTOMER FEEDBACK

Since the methodology was created with the intent of Seamless Integration across Projects and Derivatives, we could Quickly Deploy and reuse this methodology across many projects in Qualcomm with 99% accuracy and no extra overhead and resource.

The response we got back from various teams was very encouraging.

“Excellent efforts in delivering Starhawk Metis (Met80) VCDs with in very short span, which helped PD in delivering WCSS BTO data without risk for Dynamic IR. “

“Simple yet elegant solution to every netlist problem.”

“Great bridge between PD and DV”

“This would help ungate all risk to BTO/TO deliverables and uncover real issues”

VI. LIMITATIONS & FUTURE SCOPE

Currently this tool is used to solve the most time consuming step in setting up GLS. We would like to expand this methodology for auto-connection of additional netlist ports and auto-calculation of input delays. To conclude, with the proposed environment, we saved 5 days in the DV side without any stress or long nights to DV engineer. The saving of 5 days helped PD and Program Management teams declare Tapeout before the scheduled date.

VII. CONCLUSION

To conclude, with the proposed environment, we saved 5 days in the DV side without any stress or long nights to DV engineer. The saving of 5 days helped PD and Program Management teams declare Tapeout before the scheduled date.

ACKNOWLEDGMENTS

Would like to thank Qualcomm Management, Pankaj Saxena and Basha Sardar for encouraging to deep dive into the issues and find out a right solution.

REFERENCES

[1]. “Synopsys PTPX 12.3 Version User Guide”

[2]. “Absolute GLS Verification , DVCON INDIA 2015”