How Not to "Wing It" on Your Timing Constraints

Pradeep Singh, Xilinx, Inc.
Alex Warshofsky, Xilinx, Inc.
Nigel Herron, Xilinx, Inc.
Kenneth Chang, Cadence Design Systems, Inc.

Abstract

During any standard product design cycle large amounts of time and money are spent on timing verification using simulations. Not enough resources are allocated for SDC timing constraints verification – most engineers just "wing it". Like the gate level netlist, design constraints are equally important for a successful tapeout. Using incorrect or poorly written constraints can cause unnecessary rework which would impact the overall cost budget and schedule. Also, with increasing design complexity the design constraints have also increased at a very rapid rate and in some cases become unmanageable. To verify the change in constraints through design cycle stages there are not many options available today to automatically manage and verify the constraints. Most of today's method makes use of visual SDC checks, manual modification of SDC and scripting which may not be re-usable and time consuming.

In this paper we discuss about the Encounter Conformal Constraint Designer (CCD) tool which helped the design team to perform SDC Quality Checks – to ensure that constraints reflect the correct design intention. The paper will show a proven methodology for SDC Quality Check rather than relying on manual review, spot checks, etc. CCD helped us to delivers results that were used to automate our SDC management. As we continue our work, we expect that the quality of our SDC will result in increasing our design development cycle efficiencies by instilling SDC quality checks everywhere in our design flows.

Additional important features of CCD which will be highlighted in the paper are: (1) Formal Validation of false path constraints, (2) False Path constraint generation through formal proof, (3) Multi-cycle path constraints validation through formal proof, (4) SDC integration of partial constraints into single a complete SDC constraints set, and (5) Hierarchical SDC Check for consistency check to confirm that constraints get propagated across different hierarchical levels.

The goal is to demonstrate how users can apply CCD for constraints handoff and to show better alternatives to iterative, manual methods that are in use today.

I. Introduction

Validating and modifying design constraints conventionally are time consuming manual and inefficient processes. Our product development team at Xilinx has found CCD to be helpful in verifying and validating the SDC constraints very efficiently with respect to our PowerPC platform projects. CCD is useful for verifying and validating the constraints before synthesis as well as after synthesis. In this paper we will discuss our experience at Xilinx with CCD tool for verification of the constraints provided by the synthesis team to the physical design team for the place and route purposes. We will discuss how the tool helped us to check the completeness of the constraints, capture syntax errors, inform missing constraints and improve the delivered constraints for the place and route purposes.

II. Issues and Solution to SDCs driven by Physical Design Requirements

Due to the design complexity and the floorplanning constraints the physical design development team at Xilinx has to change the physical implementation methodology. The impact on the design flow is shown in Figure 1. Note that the physical design constraints like module placements, IO definitions, power grid implementation are more governed by FPGA Field Programmable Gate Array) design requirements. It was a hierarchical design with flat physical implementation. The sheer size of the design and the design complexity lead to definition of very large complex constraints file. The end result was that a constraint file with more than 100k lines, multiple clock definitions was made available to the backend team for the place and route purpose. It was impossible to verify such a large constraint file manually line by line. The amount of resources and energy that would have needed to verify the constraint file would have been tremendous. This is where the CCD solution helped the design team to efficiently and effectively verify the constraints delivered for the place and route stage.

Based on the experimental place and route runs completed for the particular design the methodology followed by the backend team was to make use of the restructuring capability of the place and route tool during timing optimization stage. Being a hierarchical design different engineers were working on different blocks and were defining there own block level constraints.

These block-level constraints were than combined to form a single top-level constraints file for place and route and static timing analysis. The block designers defined the block level constraints with respect to the block IO ports. However, the same block level IO ports now became the hierarchical module port definitions. Due to restructuring during timing optimization inside the place

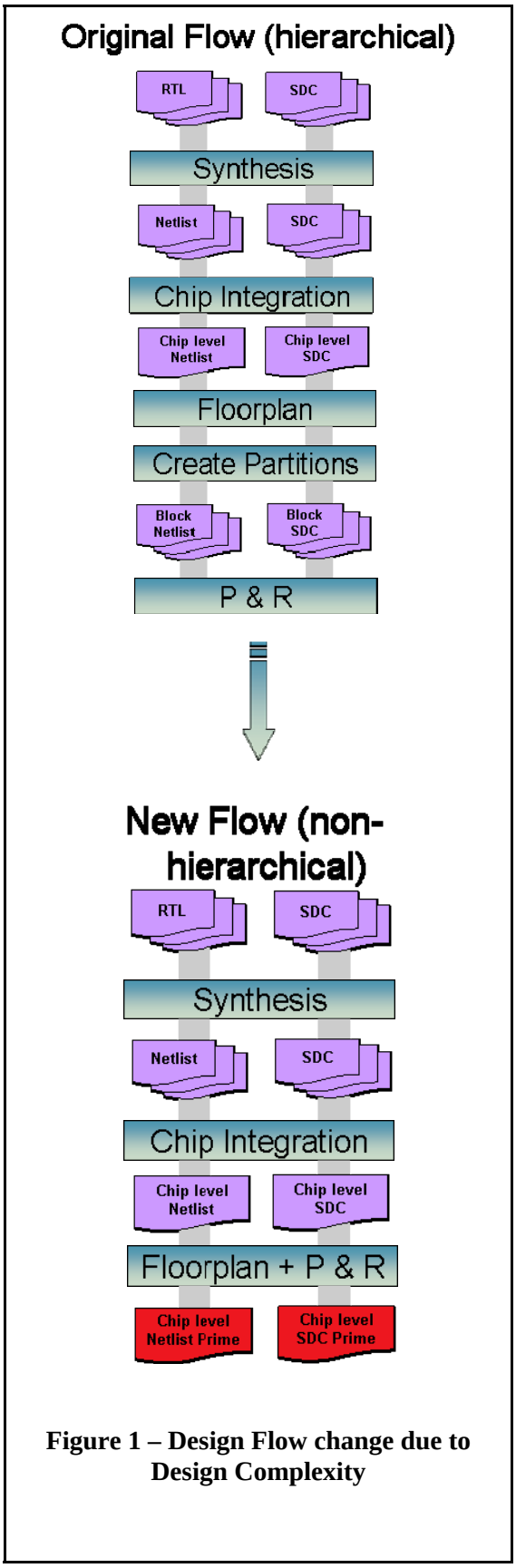

Figure 1 – Design Flow change due to Design Complexity

and route tool a large number of the constraint definition at the module hierarchy were either lost or dropped which added a large overhead on the place and route tool during further optimization down the place and route process. This it was imperative to review the constraint definition on these module port definitions and redefine constraints so that the constraints are not lost during the place and route optimization process. The CCD tool provided a fast mechanism for the top level designer to find out the constraint definition at the module ports.

III. Issues and Solutions to SDC Constraints in General for Hand-off to Physical Design Team

The version of the CCD tool we were using had more than 150 SDC rule definitions. CCD helped us to verify items such as:

- the clock definitions to ensure consistency
- input and output constraint definitions for completeness and correctness
- false path definitions
- multi cycle path definitions
- redundant definitions
- overlapping exceptions (such as false path and multi-cycle paths)

We have found the majority of SDC rule checks to be useful and therefore have been using 100% of the checks. CCD run times have been fast for our +1M gate designs, less than 10 minutes. For the PPC design CCD version 6.1 was used.

Since the design size was large and has complex clocking structures (over 10 clocks), it was not possible to verify all the input output constraints manually. In the past designers have used there internally developed scripts to resolve such errors, but this process is not always error free.

By resolving the constraint issues before the start of place and route process helped us to reduce the required number of design iterations and close our projects on schedule.

From our experiments on live designs, we have estimated an approximate 20% savings in complete cycle time. As shown in Figure 2, if the SDC quality is maintained throughout the flow, it is possible to reduce the number of iterations that the design must go through, as far back as the RTL stage, and hence reduce the total design cycle time from RTL to GDSII delivery.

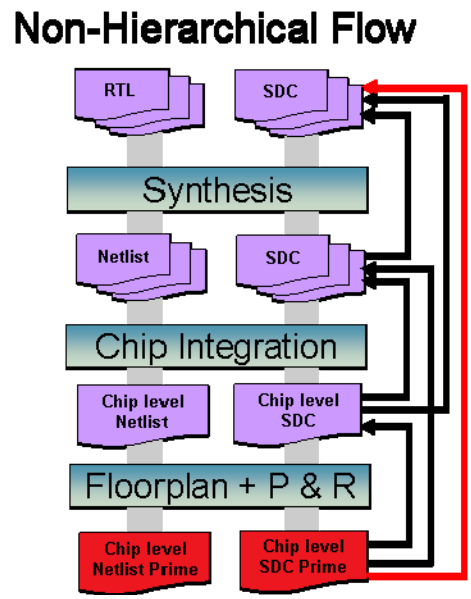

Figure 2 – Possible Vicious Cycles of Iteration (when SDCs are incorrect and/or incomplete)

IV. Summary

It was proven that after we integrated CCD to help clean our SDC constraints, the ability to reduce our total design cycle time (from RTL to GDSII) provided a great enough value to the overall Xilinx business, there after warranting further work to explore the

inclusion of CCD in other parts of the design flow.

Given a design cycle of approximately one year, this has equated to at least 2 to 3 months savings/reduction in total design cycle time. These schedule savings are important to company due to time-to-market criticality. As well, we recognize that these formal SDC verification steps help to achieve first time silicon success.

Based on these positive results, our designers plan to make use of CCD for SDC verification at the RTL level and ensure that SDCs are verified early and often, as shown in Figure 3 below.

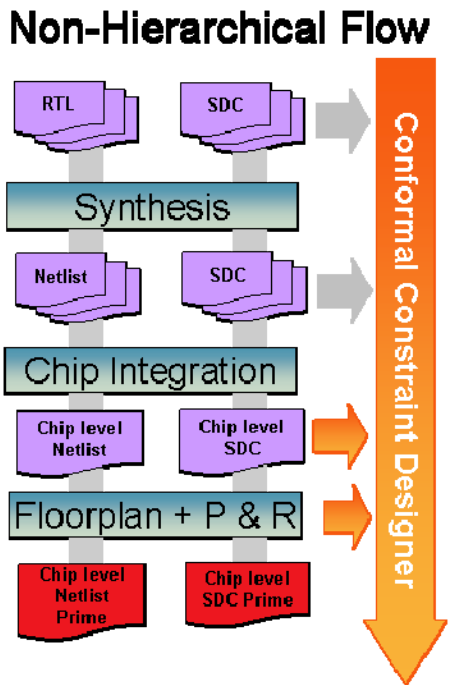

Figure 3 –
Design flow include formal SDC verification using CCD
**gray arrows denote planned activities*

V. Future Work

Based on the success of implementing "Stage 1" - SDC Quality Sign-off Checks, the design team is looking further to make use of the advanced capabilities of the CCD tool such as generation and validation of false, multi-cycle path validation, SDC integration, and hierarchical SDC consistency check.

Details on the value of each of these CCD solutions are described as follows:

- False Path Validation – false path exceptions require special attention due to the potential impact a single incorrect constraint can bring (i.e. dead silicon). Understanding this as an integral critical check, sign-off of all false path statements in the SDC at various points in the design flow where constraints transform is desired using CCD's formal validation engine.

- False Path Generation – false path generation through formal methods is recommended from at least the gate level netlist. Due to past design experience where negative slack was consistently experienced early in the design flow, a solution to solving this issue (in addition to Section II) is by timing report validation, i.e. verify if a path is truly a true or false functional path based on the environment variables. For each discovered path which is a false path, the solution is to generate a SDC false path constraint which can be used to help timing optimization in different stages of the design flow.

- Multi-Cycle Path (MCP) Validation – Due to the complexity of certain IPs and timing requirements, automated multi-cycle path validation through formal methods is desired due to the difficulty in verifying these constraints solely by manual or other means. Much like false paths, an incorrect MCP specification

can result in non-functional silicon. Using CCD's formal engine, we desired to remove the uncertainty of existing MCPs and move towards a more formal MCP validation sign-off strategy.

- SDC Integration – Due to the natural flow of building a single set of full chip SDC constraints from multiple block level SDCs (including SDCs from IPs), an automated SDC integration method would be desired to remove human error. This includes resolving clock definitions, and indicating conflicting SDC statements. Using priority driven methods, we desire a formal process for merging and propagating these types of constraints.

- Hierarchical SDC Consistency Check – As we manually merge SDCs into one chip level SDC, we desire an automated method for verifying the final SDC versus the initial block SDCs to remove human error. This method will be repeatable and require minimal manual effort.

All of these solutions are provided by CCD.

VI. Acknowledgements

We would like to thank the Xilinx management within the APD group and Cadence engineers and management for their support on the SDC constraint refinement efforts with CCD as well as for allowing this paper and presentation to be published.