Quality Assured SoC Design Using Crossfire

A Fractal whitepaper

Introduction

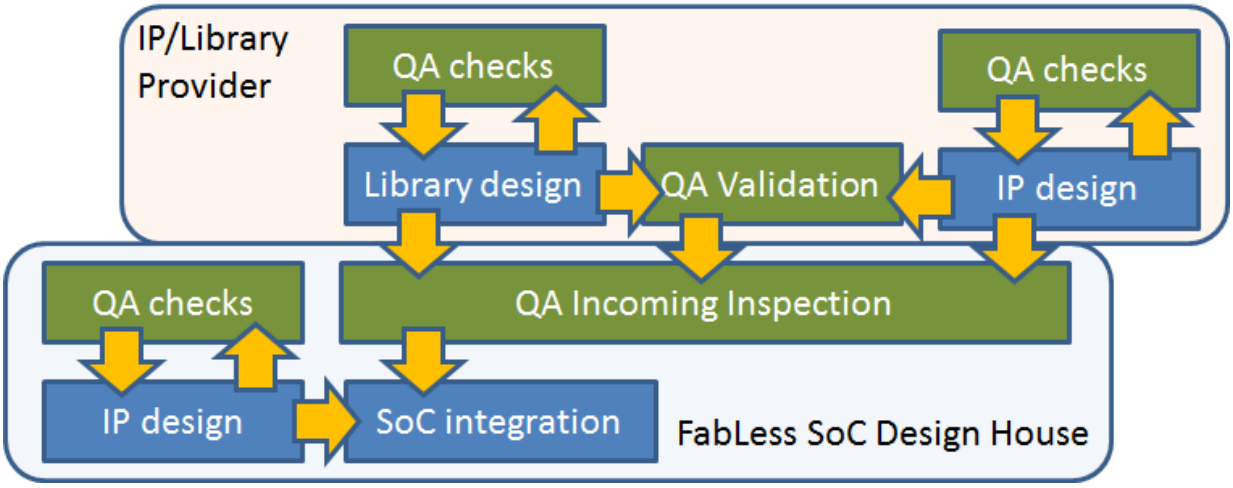
There is no industry where the need for early bug-detection is more paramount than in SoC design. Consequences like design-re-spins missed tape outs and hence missed market opportunities make the cost of late bug-detection prohibitive. Where earlier generations of SoC designs could be crafted by a team of limited size that could oversee the entire design process, design in the latest process nodes requires a different strategy. Designer productivity is lagging behind Moore's law that drives the increase of transistor density. Thus design teams are becoming larger and are comprised of multiple groups spread over the globe. Outsourcing of design-tasks by integrating third-party IP is mandatory to get the job done but reduces oversight of the SoC design process and leaves SoC design companies at the mercy of the quality strategy implemented by their suppliers. At the same time modelling of new physical effects like current-driver models for CCS and ECSM models needs to be taken into account.

It is clear that QA needs to be a shared responsibility by all partners in the SoC design flow, from library and IP providers to foundry and SoC integrators. Each of these partners needs an integrated QA solution in their part of the design flow; never should QA be an afterthought to be checked off right before IP delivery.

This paper describes how the Fractal Crossfire product can address these QA challenges. We will show how Crossfire has its place in every stage of the SoC design flow, serving different purposes to different users. Because of the breadth of deep-submicron quality checks and the rich set of supported EDA formats and databases Crossfire can cover any design flow and quality requirements.

The SoC ECO System

The figure below depicts the typical situation of a Fabless SoC design house: the design process being a mixture of custom, in-house, IP design and integration of IP blocks acquired from third party vendors. Also cell and IO libraries are externally sourced and need to be checked for integrity and compatibility with the chosen design flow.



The boundaries as depicted are more or less arbitrary. An IDM, for instance, may develop all these components in-house. The other end of the spectrum consists of SoC designers that only integrate external IP and put all their added value in the software that runs on their device. What are common though are the four categories of QA checks:

- **On-going QA checks** throughout the design process to ensure that the creator of the IP or library cells detects any issues while still modifying the IP. This ensures the smallest overhead for QA in the entire design flow by having the issues fixed by the best-qualified expert – the original designer.
- **A validation checks** are done at the end of a design-task, before shipping IP or cell-library to the customer. These checks involve more integration style checks involving compatibility between different parts and models of the deliverable.
- **QA Incoming Inspection** is again required at the receiving end and in part duplicates the checks already performed by the supplier. Their inspection's main purpose is to validate compatibility of the delivered component with the tools in the SoC design flow.
- **QA Requirements Forwarding** is the process where an IP consumer draws up a QA check set to be implemented by the IP provider, thereby greatly reducing the need for an elaborate incoming inspection task.

It should be noted that in-house activity these tasks are no different. Design-teams, often in different divisions and geographical locations can be more "alien" to each other than an IP supplier that is tightly linked with the SoC design company.

QA Aspects of Cell Library Design

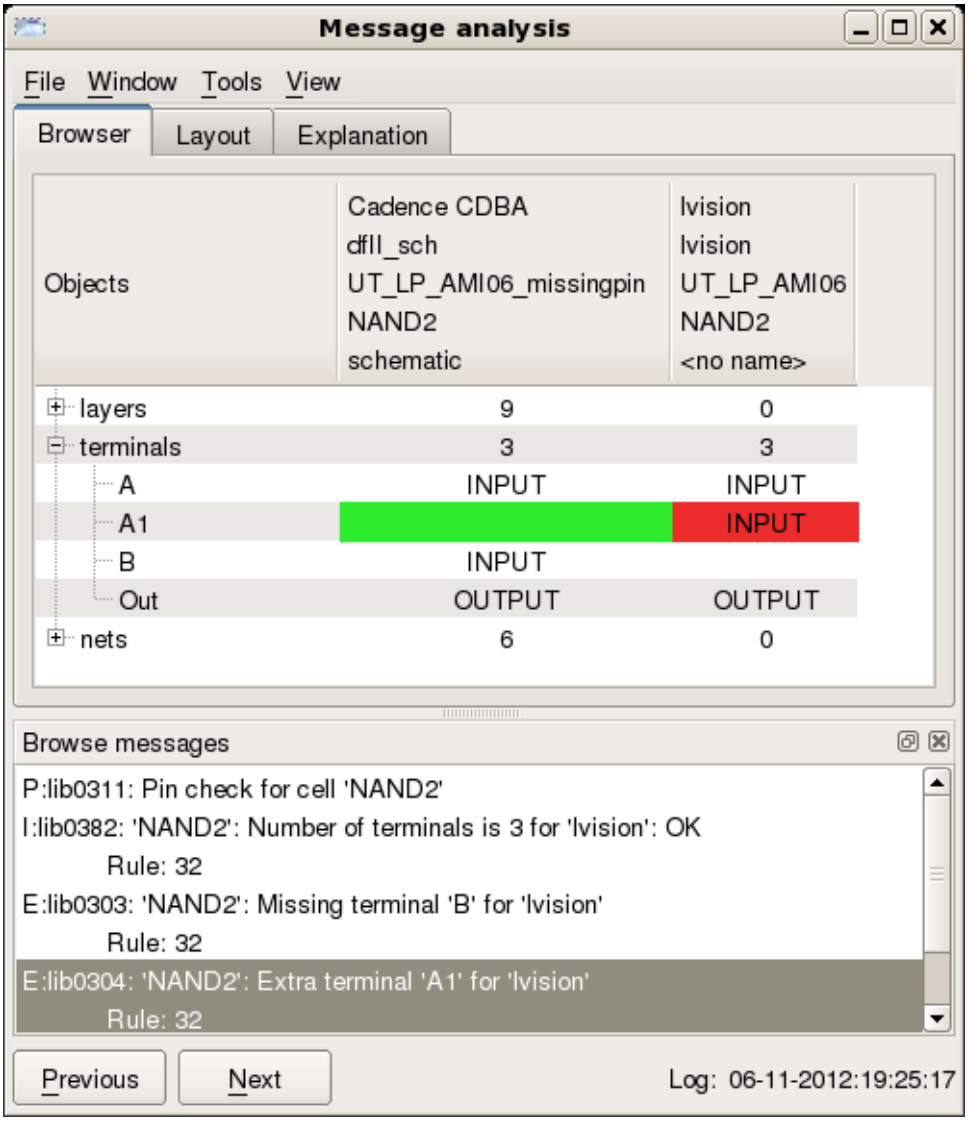
The main characteristic of cell library is volume: the large amount of individual cells (easily reaching into 1,000's) and the large amount of different models to be created for them. Not only is Crossfire capable of dealing with this large variety of different models and checks, at least as essential is its capability of presenting the results in an easy-to-understand and accessible format, and its options for automating regression-style checking.

To elaborate on the rich set of input formats, the following formats are supported:

- Cadence DFII layout & schematic views
- Open Access layout & schematic views
- Milky-Way CEL, FRAM & CON views
- Verilog
- SystemVerilog (*)
- Verilog AMS (*)
- Tetramax
- VHDL
- Liberty NLDM,NLPM
- Liberty CCS, CCSN
- Liberty ECSM
- PLIB
- TLF (Timing Library Format)
- LEF
- DEF
- SLIB
- GDSII
- Oasis
- SPICE, HSPICE, CDL
- Fastscan
- STIL/CTL (Core Test Language)
- PDF (**)
- HTML
- All ASCII user Defined Formats

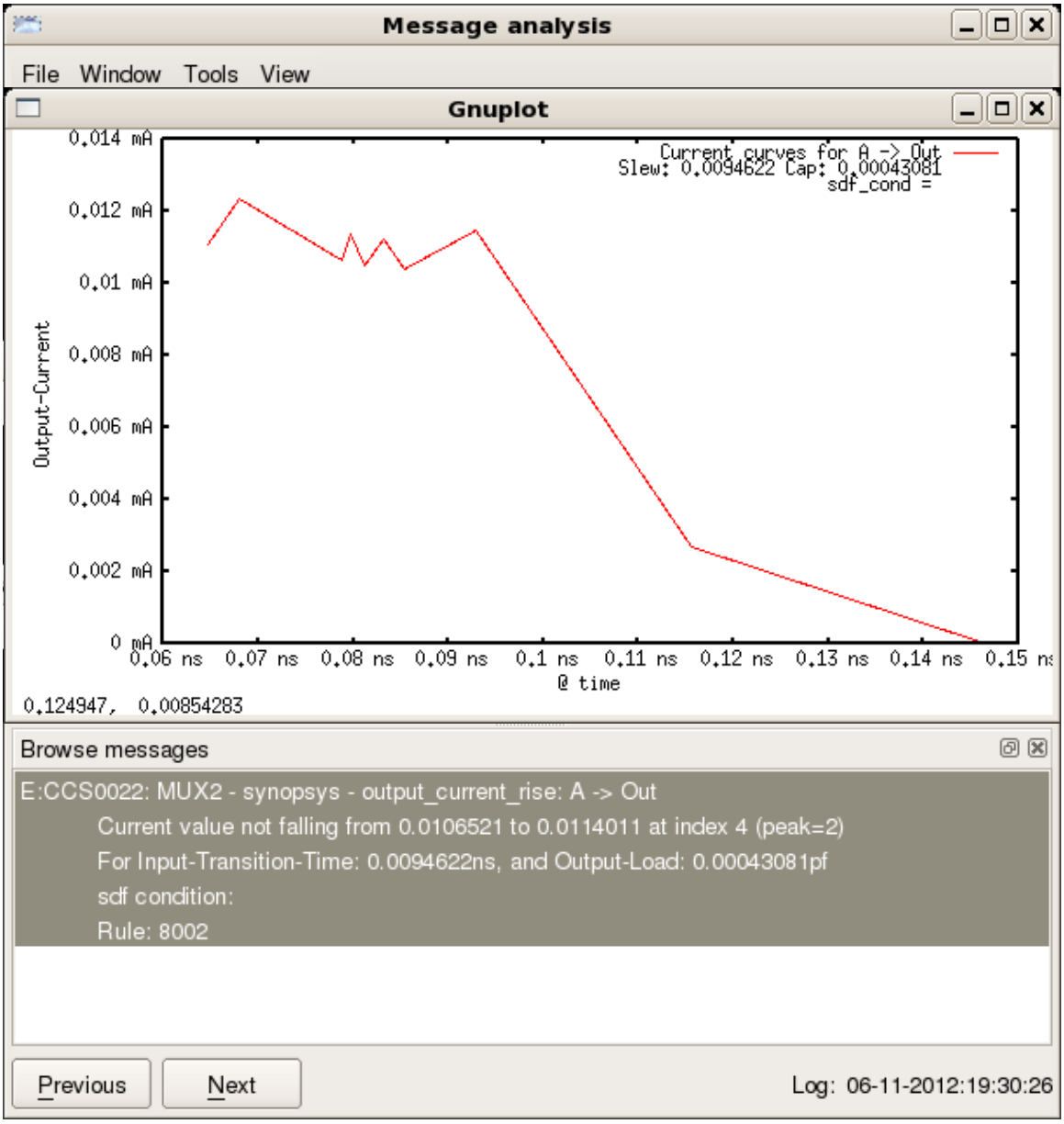
(*) No functionality check for this format
(**) After PDF to TEXT conversion

Crossfire internally constructs a unified data model covering all of these formats, making sure that the different cell models are all equivalent is a straightforward task. Any mismatch, reaching from mismatches on simple terminal names to conditional timing arcs and Boolean output terminal functionality are captured and flagged for the end-user with a graphical illustration where possible.



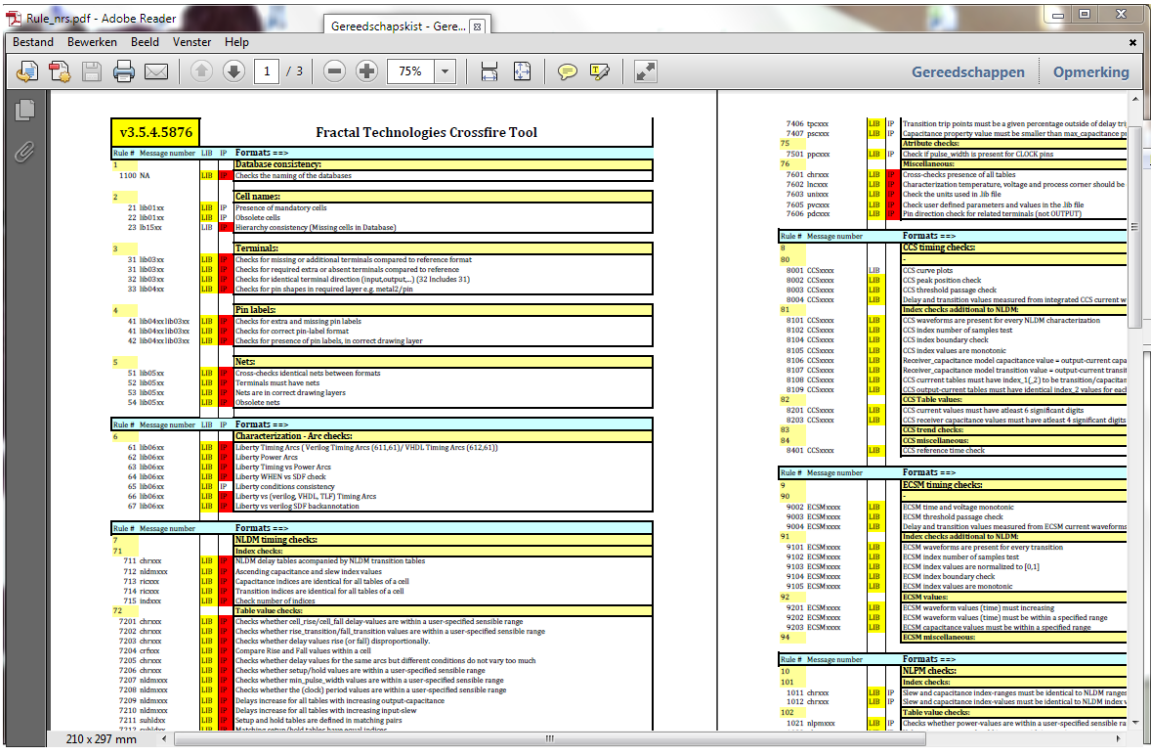
The unified data model allows users to also feed proprietary data formats into Crossfire for checking. Examples include library datasheets and proprietary cell-specification formats that describe pins, function, and number of tracks as input to cell-synthesis and characterization design flows.

The comprehensiveness of checks has already been mentioned and may be illustrated from the following screenshot from the CCS-specific checks.



For CCS and any other format all checks required to validate a cell-library are provided with the tool out-of-the-box. This allows users to quickly configure a suitable test set that covers the essentials of a particular database. Where necessary, the Crossfire API allows users to add their own customized checks to the check-set.

Of course, ease-of-use and automation are essential for a successful adaptation of Crossfire in any cell-library QA methodology. Ease-of-use for end-users has already been illustrated above; as important is that end-users are supported by partial check-sets suited for the different design-stages. Such partial check-sets are created by the CAD support team. Crossfire assists here with the automatic recognition of the tests necessary for each format. This allows the entire QA task to be sub-divided to accompany each completed step in the cells' creation process. This avoids needless backtracking, for instance by first checking on the pin-compatibility of layout, schematic and Liberty formats before starting lengthy SPICE simulations for creating ECSM models.

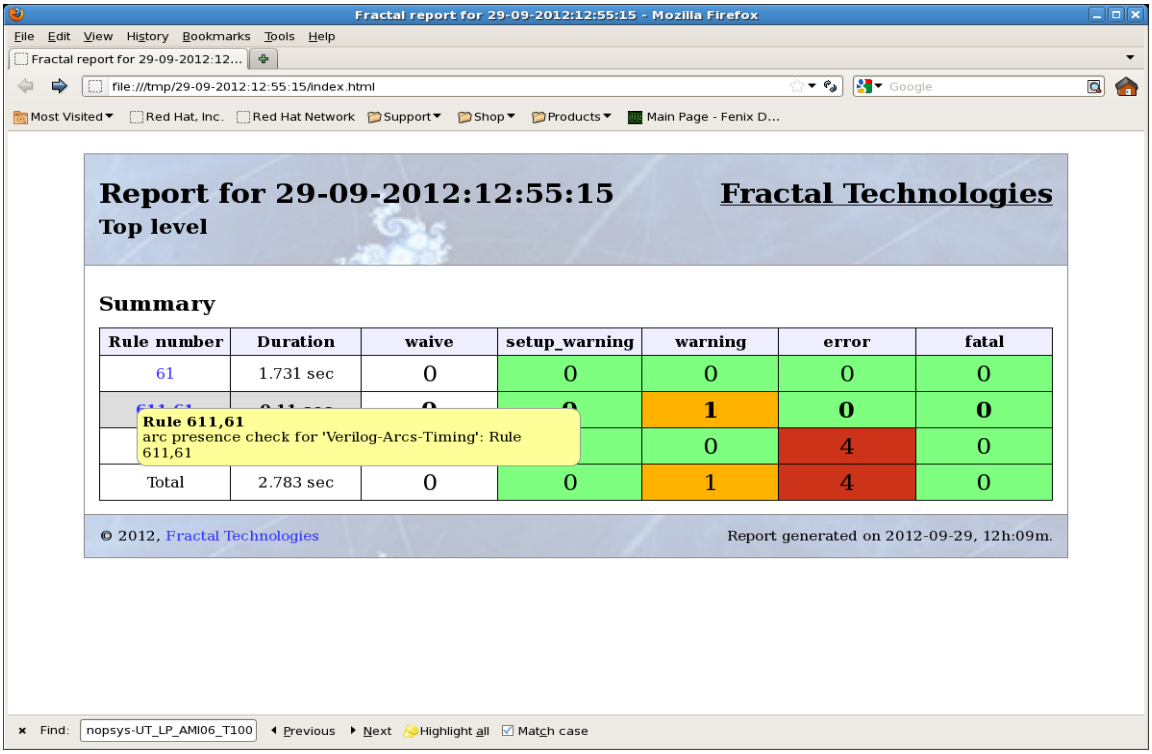


QA Aspects of IP Design

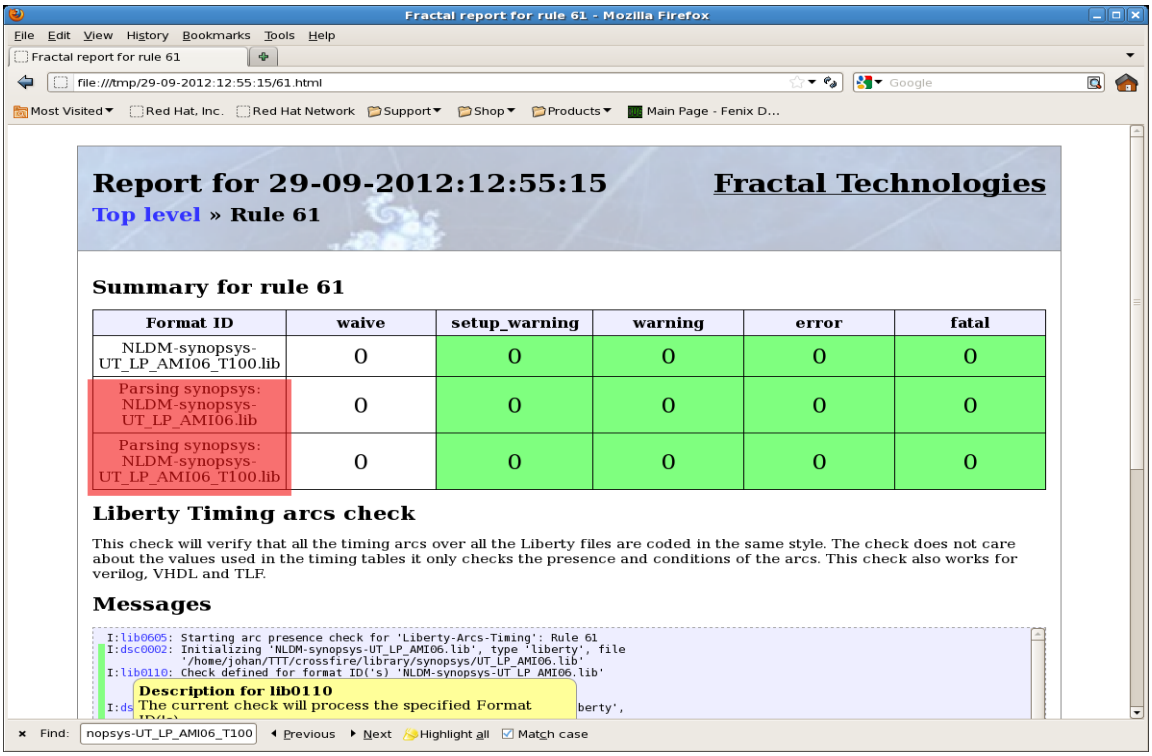
IP designers can rely on Crossfire for checking consistency and compatibility of their IP models similar to library developers. Notable differences are of course the size of the data (one large block versus many small ones) and the different types of models. Crossfire is optimized to deal also with large GDS or Verilog files, and is able to check compatibility with the different language dialects used for modelling IP such as Verilog-A.

IP can be delivered in many different ways, for instance as synthesizable digital IP blocks or as a hard analog macro GDS file. For each type, different check-sets make sure that Crossfire only checks those requirements that actually make sense. Routability checks for instance (i.e. are pins on grid and can they be reached by DRC-clean tracks from the IP boundary) apply to the various layout-related views only (GDS, OASIS, Milky-Way, OA and dfII databases). For an RTL-level IP, compatibility with the target design flow can be checked by running small fragments of the design-flow as part of the Crossfire check set. Only by running the original target synthesis or analysis tools to be used by the IP customer can we be sure that the delivered IP description is indeed valid input to the design-flow.

Crossfire also has a role to play in the final delivery of the IP to the customer. The QA reports generated by Crossfire in HTML format can be directly delivered to the IP-customer to demonstrate the integrity of the delivered IP block. An example of such a report is shown in the following figure:



The report itself far transcends the notion of a simple summary of checks performed with a pass/fail status. For each check detailed descriptions of what is checked can be directly obtained from the report itself. Likewise for checks that don't pass detailed view-comparisons or other explanations serve to demonstrate the issue on the IP-data itself. Unfamiliar as this may seem to QA-outsiders, a final QA report may indeed contain failed checks. Crossfire offers the functionality of waiving checks that fail, provided that the designer offers a motivation for this particular waive. This allows an IP blocked to pass the entire mandatory check set, while at the same time offering to the IP-customer all the necessary insight into which checks were waived and why.



QA Aspects of SoC Integration – Trust in God, but check your IP

The dominant QA aspect of SoC integration is validating the IP that is received from the various suppliers and their compatibility. Even though IP suppliers will provide proof-of-concepts with the data they deliver, SoC integrators will always need to perform incoming inspection on the libraries and IP they receive before plugging them into their design flow. The motivation is always to reduce the time-to-discovery for bugs found in the incoming models. Anything found during incoming inspection can be directly reported to the supplier and does not have the opportunity to disrupt the IP design at later stages, closer to the tape out deadline.

Crossfire is the enabling tool for quickly reaching a decision on whether a new IP or library shipment can be accepted. Once provided with the root-folder of the IP delivery, Crossfire will automatically recognize the delivered databases and formats, recognize the IP classification and select the appropriate checks that apply to these formats. A click-of-a-button later and the IP-receiver are presented with an elaborate incoming inspection report. As this report is specific to the IP delivered, evaluating the contents (in essence going over the failed tests and deciding to waive or fix them) is a straightforward process. Where needed the Crossfire API may be used to extend the built-in Crossfire checks with checks particular to the design-style deployed during SoC integration – think of naming conventions for different clock-domains.

The typical next step in an IP-supplier relationship is to move the qualification responsibility to the supplier. The IP or library supplier will take on the responsibility of proving that the delivered IP is indeed conforming to the standards agreed with the IP-consumer. Crossfire provides the ideal framework for such an engagement as the check-set for the IP qualification can be created by the IP-consumer and passed on to the IP-supplier. The incoming inspection work is then reduced to reviewing the Crossfire IP qualification report that was shipped with the IP delivery.
Such an arrangement allows for very rapid adaptation of new or modified IP blocks into the SoC design flow, making the entire SoC design process more predictable and manageable.

Conclusions

We have presented the QA requirements present in today's SoC design flow, which are made particularly challenging by large distributed design teams and the need to model deep submicron physical effects. It has been shown how the Fractal Crossfire product can serve as the QA backbone for an integrated approach to Quality Assurance. Crossfire can assist designers during their daily IP creation work and implements sign-off checks and incoming inspection checks surrounding the transfer of IP. Crossfire-check sets allow to substantially reducing the SoC integration time by transferring the qualification responsibility to IP suppliers through IP inspection check sets.
Thus Crossfire enables a truly Quality Assured SoC design flow across multiple partners, allowing predictable completion of SoC design projects.