A Fully Qualified Top-Down and Bottom-Up Mixed-Signal Design Flow for Non Volatile Memories Technologies

Pierluigi Daglio - Carlo Roma
STMicroelectronics N.V. - Agrate Brianza - Milan - Italy
E-mail: pierluigi.daglio@st.com - carlo.roma@st.com

Abstract

The wide range and rapid increase in the complexity of EDA tools demand proven and safe design flows. This paper presents a complete and fully qualified mixed-signal top-down design flow for non volatile memory applications. It has been successfully applied to an Embedded Flash Macrocell based design as well as to a 14-bit analog/digital converter with digital non linearity compensation manufactured in 0.18um proprietary flash technology. One remarkable feature of the proposed methodology is the high level of integration among EDA tools from different vendors and internally developed solutions. Mixed-signal domain has been really explored at any level: functional, behavioural, vhdl/schematic and post layout with parasitic components. Furthermore, we propose a bottom-up methodology to generate and validate VHDL-AMS models for IP analog cells. All the illustrated features are integrated in a design flow which provides full compatibility and flexibility between analog and digital design steps to cut down time-to-design, improve time-to-market and streamline design quality.

Introduction

Often designers can not completely rely on a well-defined methodology throughout the entire project development. For this reason, sometimes designers of the same organization use different approaches to mixed-signal designs without following a standardized flow. This implies an extra cost and a lower quality of results, including difficulties in cross-fertilization as far as mixed-signal design methodology is concerned.

The proposed **Mixed-Signal Design Flow (MSDF)** suggests a methodology for mixed-signal circuit development and analysis promoting the usage of state-of-the-art EDA tools whose functionalities have been completely validated on Non Volatile Memory (NVM) technologies. The set of tools that make up the system configuration for MSDF either belong to major EDA vendors or have been developed internally to our company. Cross-link utilities, translators and scripts are also used in the flow.

Notably, nine major design steps will be accounted for in this paper:

- ***Mathematical Functional Verification***
- ***Behavioural Simulation***
- ***VHDL-AMS IP Models Validation***
- ***Analog Electrical Simulation & Optimization***
- ***Analog Layout & Verification***
- ***Digital VHDL & Gate Level Simulation***
- ***Digital Front-to-Back***
- ***Mixed-Signal Simulation (VHDL/Schematic)***
- ***Full Chip Flat and Hierarchical Post Layout Simulation with Parasitics***

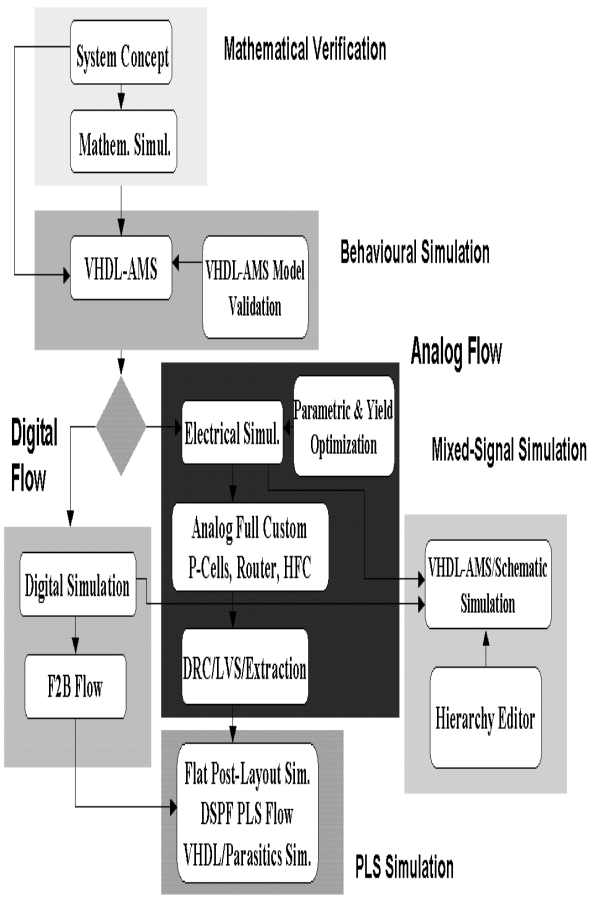

Figure 1. Top-Down Mixed-Signal Design Flow

Figure 1 shows the flowchart related to the top-down Mixed-Signal Design Flow: the main design steps have been highlighted in different shades of gray. MSDF has been qualified on various circuits in a number of technologies. In this paper, an Embedded Flash Macrocell based design (EFM) in 0.18um flash technology has been mainly examined.

The Embedded Flash Macrocell

The Embedded Flash Macrocell based design (EFM) considered as test circuit to validate the MSDF is shown in figure 2. The circuit consists of three main regions: a finite state machine (digital portion), several sense amplifiers and the programming/erasing circuitry (analog portion) and, finally, a flash cell array (non-volatile memory portion).

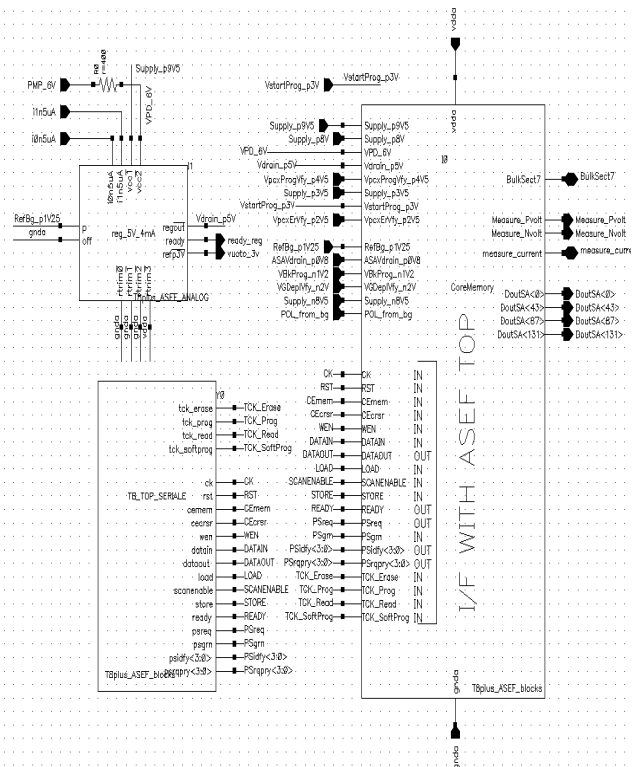

Figure 2. The Embedded Flash Macrocell testbench

The simulation of flash cells is a non-trivial task with traditional mixed-signal simulation methodologies because the threshold voltages change according to the potential applied to the cell terminals. In our case, using the IEEE VHDL-AMS standard language, it was possible to implement a behavioural model of the flash cell, electrically compatible with the traditional analog components. Furthermore, such a language allowed to process digital blocks, implemented in standard VHDL, together with analog circuitry even at schematic level, making therefore possible to run a real full chip simulation.

1. Mathematical Functional Verification

Mathematical functional verification has been based upon a commercial tool that allows designers to describe circuit behaviour with mathematical functions and, consequently, to simulate it. Libraries containing analog and digital blocks with different functionalities are also supplied by the simulation environment. Once the capabilities of the application have been verified, designers can switch to the behavioural simulation step that, in the proposed flow, is based on the VHDL-AMS facilities.

Sometimes, this kind of verification can be skipped and designers enter mixed-signal flow directly from the behavioural simulation step. The purpose of the functional verification is to understand, in a fast way, if your idea works as you conceived it. Circuit functionalities are checked only from a mathematical point of view, that is: supposing that the blocks have a given transfer function, at a certain set of input signals corresponds an expected set of output signals. The main advantage of this kind of approach is the possibility to run very long time frame simulations very quickly.

2. Behavioural Simulation

Once designers have checked circuit capabilities through the mathematical functional verification, they can model any block using VHDL-AMS behavioural language to perform mixed-signal simulation at behavioural level. Analog blocks are implemented in VHDL-AMS while digital blocks in pure VHDL or Verilog (standard cells).

When models are ready, designers can import them in a design framework automatically generating the corresponding symbols. At the end of the importing operation, a library of cells with *vhdl-ams* and *symbol* views has been created. Using those cells, designers can now draw the schematic of the mixed-signal application.

Then, from the schematic window, the mixed-signal simulation environment can be entered. There, circuit simulation at behavioural level can be set up and run. Such an environment supplies the designer with a powerful *hierarchical editor*. This kind of tool allows to easily choose, for any block, the view to netlist for the next simulation (*vhdl-ams*, *schematic*, *extracted*, *textual*, ...) without modifying the properties of the instanced cells.

Figure 3 shows the results of the EFM mixed-signal behavioural simulation related to a programming cycle. When writing data in the memory, the internal state machine executes a set of programming and verifying pulses which take care of all the programming phase. The figure presents the evolution of the threshold voltage of one flash cell along with the interface signals that triggered the

programming command (WEN, CEMEM) and the response signal from the chip (READY) which indicates when the operation is on-going (low level) and when it is completed (high level). At the end of the programming operations, the threshold voltage sets to the correct value so the algorithm ends up and the READY signal goes high.

Once the functionalities of the application have been verified at behavioural level so that circuit is working as expected, the flow splits into analog and digital portions.

At this stage, analog designers need to implement analog blocks at schematic level, to perform electrical simulation and, if necessary, to carry out performance and yield optimization of critical analog blocks. Then, they have to produce the full custom layout of the analog parts of the whole application.

Likewise, digital designers need to synthesize VHDL code to produce gate level schematic and, in case, simulate it. Then, they will have to go through the digital front-to-back sub-flow to generate, verify and optimize, according to the specifications, the functionalities of the digital part.

Analog and digital specific sub-flows are taken into account in the following paragraphs of this paper.

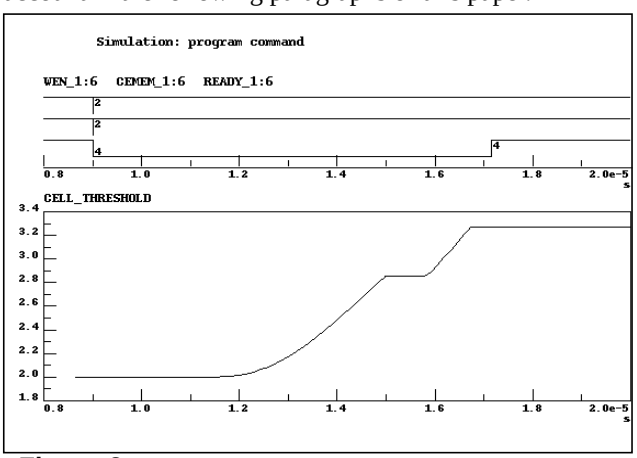

Figure 3. Mixed-signal behavioural simulation of the macrocell related to a programming cycle. Flash cell threshold voltage and digital control signals are visible

3. Bottom-Up Methodology to Generate and Validate VHDL-AMS Models

A specific bottom-up methodology to generate and validate accurate VHDL-AMS models is becoming more and more important for analog designers to rely on precise behavioural models for IP analog cells. This approach is very useful because chip size is always growing and, consequently, full chip transistor level simulation is becoming more and more difficult or even impossible due to memory limitations and huge simulation time. For this reason, mixed-signal simulation with accurate models, replacing transistor level blocks, is mandatory.

At this purpose, first step is to characterize the transistor level blocks using our golden reference simulator. Then, by means of a mathematical post-processor, the target is to search the best polynomial fitting each magnitude (dependent variable) versus parameters (independent variables). Polynomials can be of any order (quadratic, cubic, 4th order, ...) reproducing as well as possible circuit responses. Today, this methodology is partially based on internally developed solutions and partially on commercial tools but, in the future, we plan to switch totally to a standard product, obviously as soon as its reliability will be certified. At this point, polynomial functions can be implemented in VHDL-AMS format to be used as accurate models. Electrical and physical equations to monitor specific variables can be written inside the VHDL-AMS models too.
So doing, the behavioural model for any IP analog block comes out well characterized and accurate, allowing a full chip simulation sharing a good trade-off between speed and accuracy. Moreover, a VHDL-AMS model library would be available for IP reuse purposes. An example of VHDL-AMS model validation related to an operational amplifier is shown in figure 4.

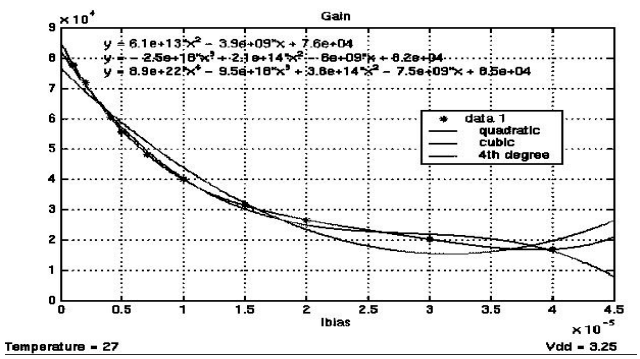

Figure 4. Example of AMS model validation related to an operational amplifier. The curves represent the gain magnitude versus the biasing current (for a fixed temperature and a fixed Vdd) and the different polynomial results.

4. Analog Electrical Simulation & Optimization

Analog electrical simulation and block optimization are achieved by means of a specific Analog Design Flow in place in our company.

After schematic entry, based upon a commercial framework schematic editor and on Design Kit (technology dependent set of data), designers can enter the analog simulation environment. There, the circuit electrical simulation can be set up and run. Such a commercial environment has been enhanced with company tools (ArtistKit and StatKit)

to fit designer needs and requests.

Furthermore, they also provide the entry point to the advanced analog tools for circuit nominal optimization and statistical design analysis. Parametric optimization is achieved using commercial solutions of different vendors. Window forms to define an optimization problem, to set the variables to trim, to fix constraints, to track parameters and to run the selected optimizer have been implemented. Output data from the optimizer can be directly back-annotated to the schematic composer. In this way, designers will get the optimized circuit directly available in their library.

5. Analog Layout & Verification

Two different commercial tools are supported to layout the circuit: one to instantiate design kit parametric cells and another one to route automatically the devices.

The former, starting from the schematic view, generates, using the *parametric cells* mechanism, the layout of any single component. Then, devices should be placed inside a pre-defined box and eventually routed. Regrettably, at present, the automatic placement of analog devices is not fully supported yet. In addition, an internally developed tool and an analog library of *matched cells*, named respectively as HFCKit and HFClib, help designers in achieving a highly performing layout. The layout of the whole analog part of EFM project is shown in figure 5.

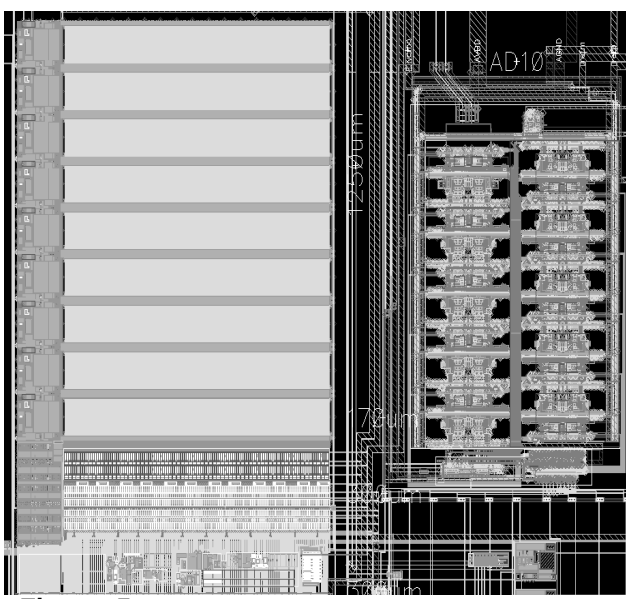

Figure 5. Layout of the whole analog portion of EFM: parametric cells and automatic routing have been used.

Layout verification is carried out by a commercial tool. It supports DRC and LVS on the gds2 layout format and also manages to prepare (with a dedicated LVS run) the input data deck to the parasitics extractor.

Finally, an internally developed tool, named as HFC-Check helps designers to set and launch correctly such a commercial verification tool.

6. Digital VHDL & Gate-Level Simulation

Digital simulation can be performed both at VHDL and gate levels. In the first case, VHDL code of digital blocks is compiled and simulation is run using a commercial simulator which can be also linked to the VHDL-AMS simulator for mixed-signal purposes. VHDL code can be synthesized to generate a gate level schematic which, at its turn, can be netlisted and simulated too.

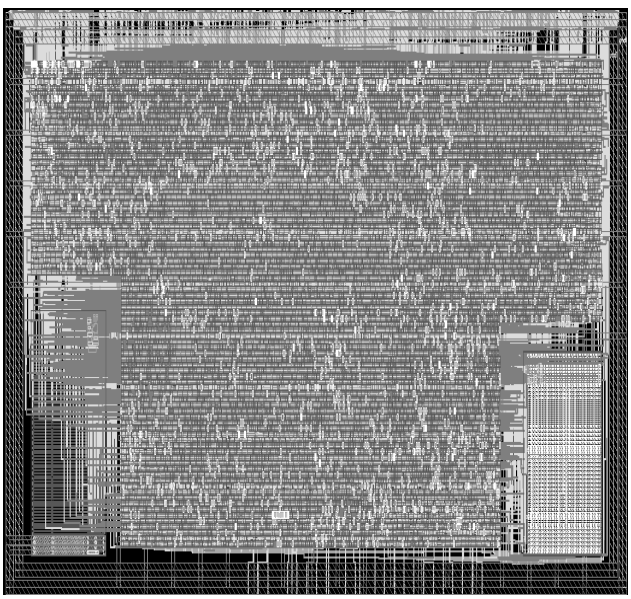

Figure 6. Layout of the digital portion of EFM project obtained with the Front-to-Back Digital Flow

7. Digital Front-to-Back

The layout of the digital portion of EFM is achieved by means of a dedicated flow, named as Front-to-Back (F2B) flow. It allows, starting from the netlist of the digital top level, to automatically generate the layout of the digital parts keeping into account performances and area constraints. Layout optimization is achieved looping between layout and synthesis stages. An internally developed set of tools, named as AsicKit, integrated with commercial tools for synthesis and physical design allows designers to perform incremental synthesis, placement, routing, timing analysis and verification. The layout of the digital portion of EFM project, obtained through the digital F2B flow, is shown in figure 6.

Due to the number of pages limitation, we can not go in

details about F2B flow in this paper, which is more oriented to analog aspects of mixed-signal designs.

8. Mixed-Signal VHDL/Schematic Level Simulation

Along the designing phase, usually, there is the need to run a mixed-signal simulation coupling the digital portion of the circuit at VHDL level with the analog portion of the circuit at schematic level to investigate about undesired effects at the analog/digital interfaces. Sometimes, designers set the digital part at VHDL level, some analog non-critical blocks at VHDL-AMS level and analog critical blocks at schematic level. This “real-life” mixed-signal simulation can be achieved by means of a commercial tool which allows to combine together all those different kinds of circuit representations. By using the *hierarchical editor*, it is quite easy to switch from behavioural to schematic representation of any single block, so that different configurations can be exploited. After choosing the desired configuration, designers can enter the simulation environment and run the netlister. Blocks will be netlisted according to the selected view (VHDL/Verilog, VHDL-AMS and schematic). Then, designers can invoke the simulation. VHDL-AMS, digital and analog simulators start together loading the corresponding portion of the circuit according to the partitioning set through the hierarchical editor. All the three simulators run in parallel in a synchronized way to produce the corresponding output waveforms. Combining those engines together, it is possible to simulate faster than traditional approaches processing systems that other simulators would not be able to handle.

9. Flat and Hierarchical Post-Layout Simulation with Parasitic Components

Once the layout of analog and digital portions of the circuit are available, designers can put them together and route them launching the automatic router. Obviously, they should have performed a floorplanning at the beginning of the project to reserve the needed space in the whole chip for the different portions of layout. After that step, the mixed-signal application top level layout is ready.

DRC and LVS full chip operations are performed using the same commercial tool as for the analog blocks.

At this point, designers need to extract parasitics to perform the post layout full chip simulation with parasitic components. An internally developed tool, named as DKCust, allows designers to tailor the technology basic extraction rules to their needs. Then, they can easily run the extractor to obtain a netlist including parasitic components.

Netlist is quite large and flat, as the extractor in use is not capable to generate a hierarchical netlist. At this point, a set of scripts helps to generate a fully compatible input deck to the chosen simulator.

Today, the only way to simulate such kind of large flat netlist is to use *lookup table* simulators. In our company, mainly three different lookup table simulators are in use and their effectiveness depends on several factors like circuit topology, percentage of analog/digital portions, needed accuracy and speed. They can be used to simulate a large analog top level or a mixed-signal design where critical analog blocks come into play. Some engines are almost fully compatible with electrical simulator netlist syntax whereas others have been integrated through internally developed utilities. Those programs automatically bring a standard netlist into the right format, grab the right models, run the lookup table simulator, take the output results and convert them into the required output format. In this way, the output from any lookup table simulator can be loaded inside the waveform display normally used for wave comparison.

In our design flow, we propose two more interesting possibilities for full chip post-layout simulation. The first option is to keep the digital portion of the circuit in pure VHDL, set the non-critical analog blocks to VHDL-AMS description and replace the critical analog blocks with their netlist containing parasitic components. Everything can be easily managed through the hierarchical editor. In this way, designers reach a good trade-off between simulation accuracy and speed, taking into account only the parasitics that really could cause problems to the circuit performances.

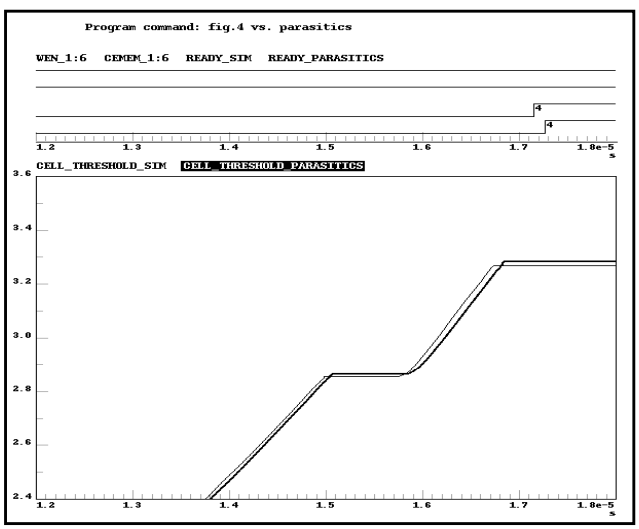

Figure 7. Simulation outputs of the pre-layout simulation at behavioural level and of the post-layout simulation with parasitics in a programming cycle

The second option is known as *DSPF flow* and consists of using the pre-layout hierarchical netlist back-annotated

with device parameters (AS, AD, PS, PD, ...) taken from the layout plus the parasitics computed by the extractor in DSPF format. In this way, the netlist is hierarchical so that simulation can take advantage from lookup tables and hierarchical array reduction algorithms.

Post-layout output signals can now be compared against pre-layout ones using the same graphical waveform display. It enables to gather signals coming from different simulations (pre-layout and post-layout) in the same strip. Figure 7 shows the comparison between pre-layout and post-layout simulations. It can be noticed that the threshold evolution is slightly slower in the post-layout case (thicker line) and, accordingly, the READY signal (second digital strip) goes high later than in the pre-layout simulation (first digital strip). Furthermore, it is worth noting once again how the changes in analog voltages influence the response of the digital state machine.

Design Quality Enhancements

Major benefits to the design quality derive from the usage of advanced tools for mixed-signal simulation that enable both to simulate very large circuits by means of behavioural language and to really check undesired effects at the interfaces between analog and digital portions of the chip, where problems are often found when silicon is already out. Furthermore, the fact that designers can simulate the whole circuit at different mixed-signal levels (mathematical, behavioural, digital VHDL together with analog schematics and post-layout) always provides a good trade-off between accuracy and speed. In this way, designers can privilege either speed (mathematical and behavioural simulations) when they need to check system consistency or accuracy when they need to check critical functionalities. Moreover, they can also control exactly what happens at the analog/digital interfaces of the circuit blocks (VHDL/schematic and lookup table simulations).

Mixing together all these possibilities allows designers to use the right tool at the right moment, highly increasing the quality of their projects with the purpose of having the first silicon working good.

Conclusions

A top-down Mixed-Signal Design Flow integrating tools from the major EDA vendors and internally developed utilities has been presented in this paper. MSDF allows the designer community to rely on a sound and well-proven standard methodology when designing large mixed-signal circuits or systems-on-chip so that they can really focus on design while reducing the time minding EDA tool issues.

The quality of the circuits can be highly improved by means of advanced mixed-signal tools which allow to simulate the whole application at any level of abstraction and to detect interface errors prior to go to the silicon.

The philosophy implemented in MSDF is to integrate commercial solutions, when available, to standardize the flow at the most. Internally developed tools and sets of script files have been conceived to link the various commercial tools and to replace missing features when necessary. The major benefits related to MSDF are the standardization of the methodology, the possibility to check the whole circuit responses at any design stage and the enhancement in terms of speed brought about to the overall design cycle avoiding unpleasant surprises at the first silicon out.

Acknowledgments

Thanks to all the people of Design Support & Flows, Design Kit and ISDG groups for their remarkable contribution to this paper.

References

[1] “ADVance MS User’s Manual”, Revision 1.5_1, Mentor Graphics Corporation, U.S.A., December 2001

[2] “Artist Link User’s Manual”, Revision 4.1_2, Mentor Graphics Corporation, U.S.A., December 2001

[3] “Hierarchy Editor User’s Manual”, Product Version 4.4.3, Cadence Design System, U.S.A., December 1998

[4] “AsicKit User’s Manual”, Unicad 2.4, Unicad2 documentation, May 2001

[5] “Virtuoso Layout Editor and Virtuoso XL”, Product Version 4.4.3, Cadence Design System, U.S.A., March 1999

[6] “Arcadia Reference Guide”, Release 5.6, Synopsys Inc., U.S.A., June 2001

[7] “Arcadia User Guide”, Release 5.6, Synopsys Inc., U.S.A., June 2001