Projet ANR VALMEM

Délivrable : D1.1

Titre : *State of art in eSRAM design and validation flow*

Auteurs : R. Chevallier

Version : 1

Date : 30 Mai 2007

VALMEM : Validation fonctionnelle et temporelle des mémoires embarquées décrites au niveau transistor par des méthodes formelles

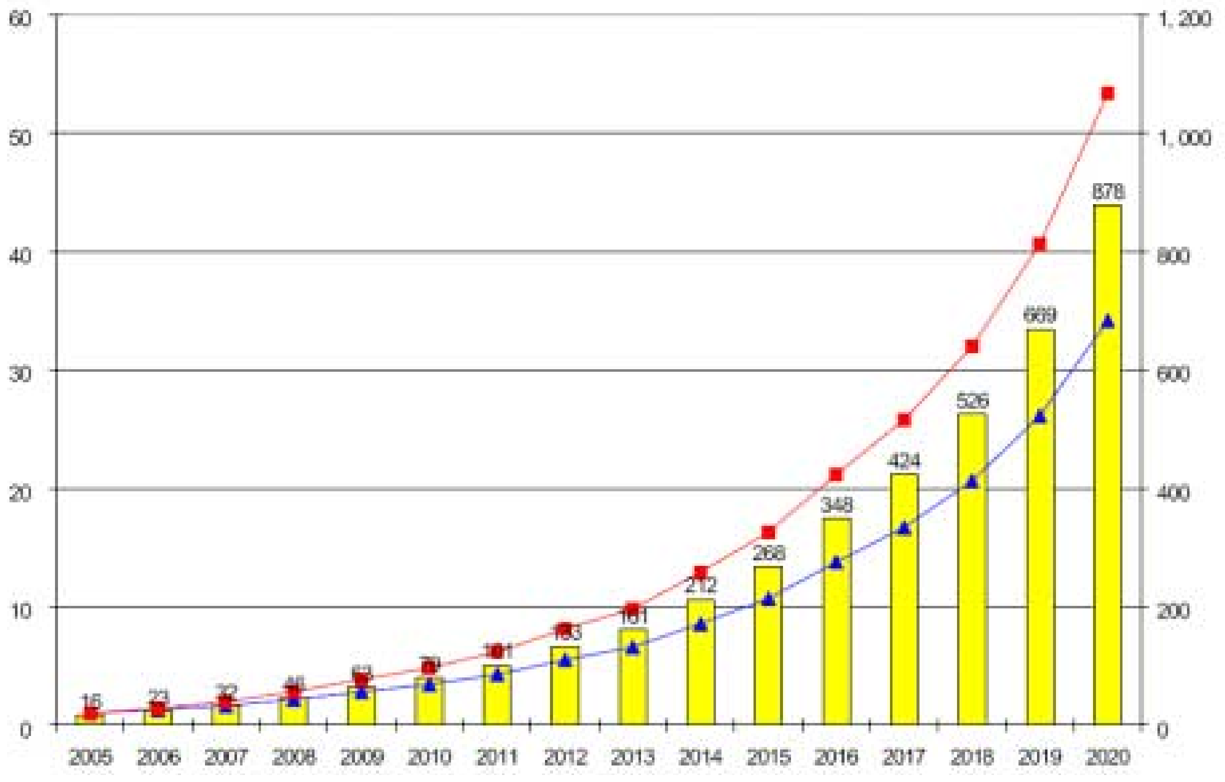
State of Art in eSRAM design and validation flow

Author	date	Release	Comment
Remy Chevallier	13 February 2007	0.1	First release
Remy Chevallier	30 March 2007	1	Maturity section updated

1. Introduction

1.1. Market overview and forecast

With the improvement of the technology, more and more functionalities are embedded inside the chips. More functionality induces more embedded memories to save the data.



🟨 Number of processor customized for a specific function (right axis)

🟥 Total memory size normalized 2005 (left axis)

🟦 Total logic size normalized 2005 (right axis)

ITRS 2005 System Drivers

According to the ITRS forecasts, the total size of the memory needed in a chip will continue to grow during the coming years.
Moreover, the functionalities and the customer constraints are exploding. Wireless market is looking for low consumption and high density memory, and at the opposite, Wire line market is looking for high speed memories.
These two points induces the growth of the embedded SRAM requests, and an explosion of the different kind of requests.
On the top of that, the memories are always into the critical path of the designs: customers put high pressures to the memory team to optimize their performances.

1.2. eSRAM design flow

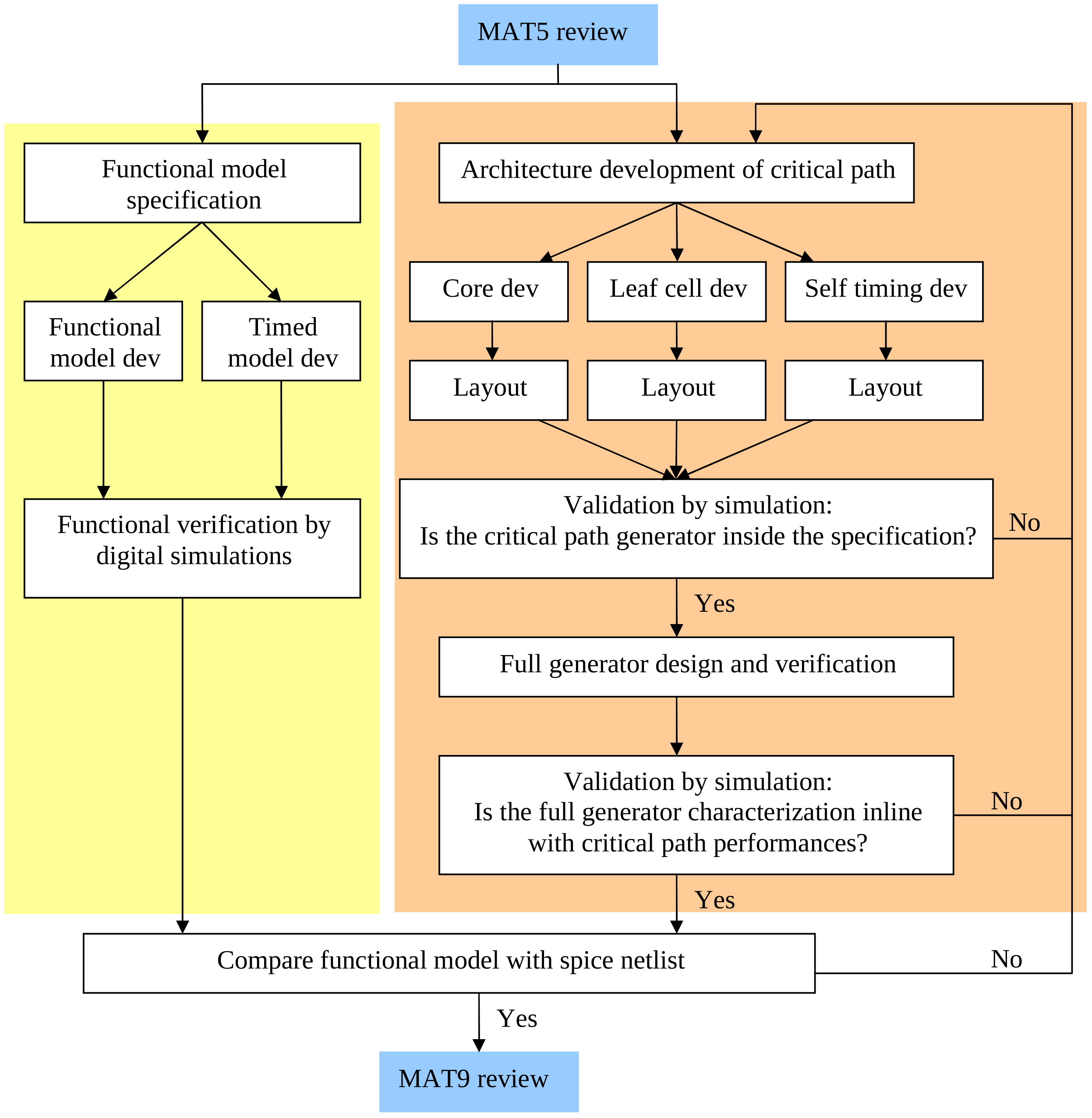
Today the eSRAM requests are split into markets (low-power, high speed, high density) and capacity ranges. For each defined areas, a memory compiler must be developed; A generator handles a range of possible memory size and performance characteristics, and it generates under customer demands all the needed view for his needs.
The design process for a memory compiler is very costly and follows a complex verification and quality steps.
The quality process is split into maturity (MAT) items described in the next table.

Quality step name	Description
MAT5	Memory specification frozen Customers identified
MAT9	Memory compiler ready Primary views built and checked
MAT10	Additional derived views are generated
MAT20	Design successfully checked on silicon
MAT30	Design successfully checked on all temperature ranges.

🟨	Simulation and CAD-views
🟧	Silicon needed

The design and the verification of the compiler are performed between the MAT5 and the MAT9 items. MAT9 included primary views: schematic (spice), layout (gds), timings (.lib). The MAT10 included additional views needed by customers to include easily the memory inside their development flow. These views are directly generated from the primary views generated for the MAT9 step.
In this document, the processes between the MAT5 and the MAT9 will be described. In a first step, a global overview of the process will be presented before zooming into transistor based design and into HDL model design

2. MAT5 to MAT9 development overview

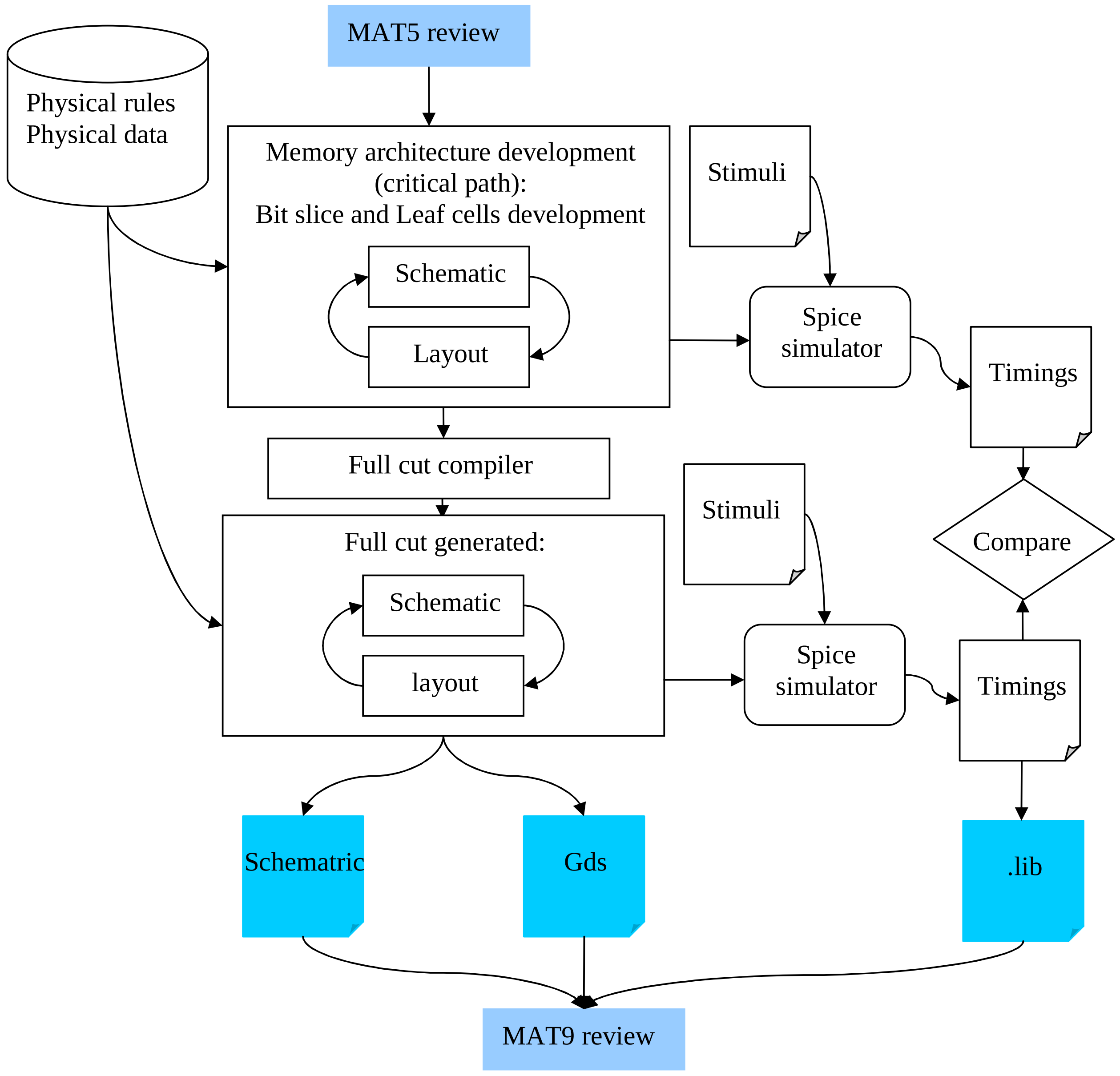

MAT5 review
Functional model specification
Functional model dev
Timed model dev
Functional verification by digital simulations
Architecture development of critical path
Core dev
Leaf cell dev
Self timing dev
Layout
Layout
Layout
Validation by simulation: Is the critical path generator inside the specification?
No
Yes
Full generator design and verification
Validation by simulation: Is the full generator characterization inline with critical path performances?
No
Yes
Compare functional model with spice netlist
No
Yes
MAT9 review

🟨 Digital model (HDL) development follow

🟧 Transistor based development flow

3. Transistor based design flow

This section zooms on the development flow used to provide MAT9 data from MAT5 data for transistor based design. The memory generator is a first phase developed and validated on a critical path (simplified memory)
The real memory is generated in a second phase. The performances computed on the critical path are compared with the real memory performances.

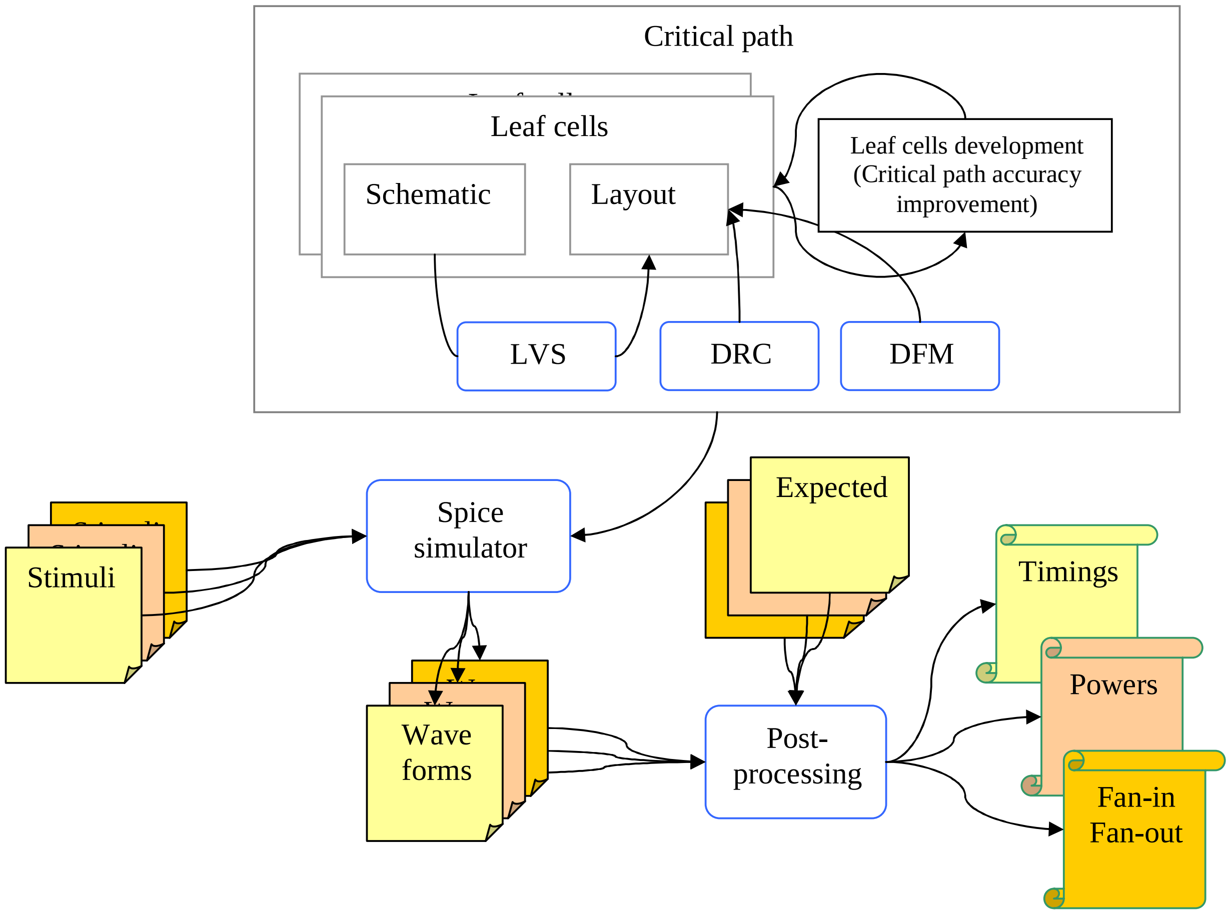


The '.lib' view describe the timing arcs between inputs and outputs of the memory.

3.1. Step1: Build critical path

3.1.1. Design flow

The critical path is a simplified view of the memory: only the longest and the shortest paths to the memory points inside the memory are fully described. The other part of the memory is modeled as passive components (resistors and capacitors).
The memory point is provided to the design team by a dedicated team.
The design is built by elementary bricks called leaf cells. Leaf cells are plugs to provide the full functionality of the memory.
The critical path includes hands computed capacitors and resistors in order to represent the real behavior of the main wires.



The layout of the full memory is generated by its memorycompiler. The configuration of this tool has to be performed for the project.

3.1.2. QA methodologies

3.1.2.1. Static verification methods

- LVS (Layout versus Schematic)

The LVS verification is applied for each leaf cells, one by one during the development flow.

- DRC (Design Rules Check), DFM (Design For Manufacturing)

The DRC/DFM checkers assures that the layout shapes are well supported by the silicon process.

3.1.2.2. Simulation methodology

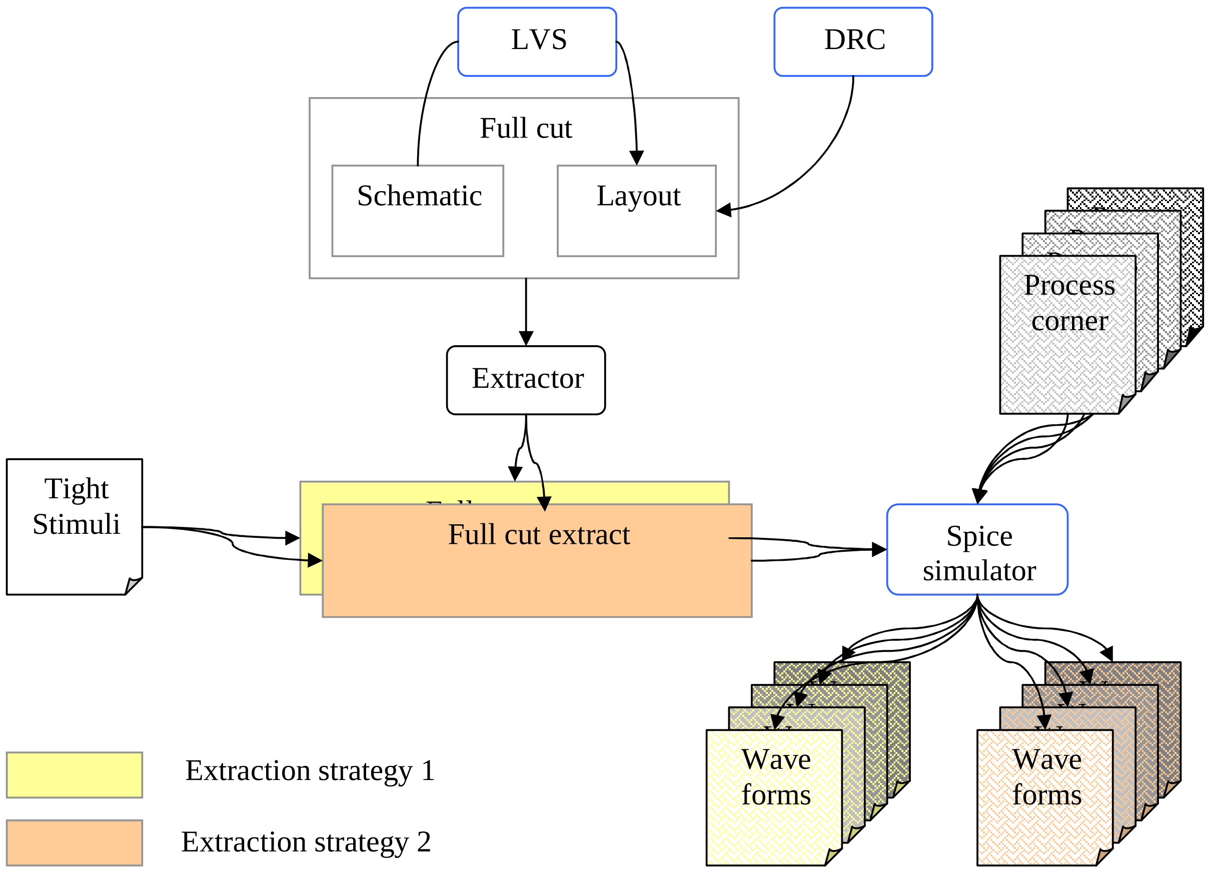
The stimuli must be built according to the behavior to check. The challenge is to check all the behavior of the design with a minimum of clock cycle in order to reduce the simulation run time.
The expected behavior of the signals is developed and read by a post-processing tool.

3.2. Conclusion

The critical path is linked to the designs of leaf cells. Indeed, if the simulation of the critical path does not fit to the designer request, leaf cell layouts are optimized, or, the schematic itself is updated. The critical path development and the leaf cell development cannot be separated.
When the transistor model is updated, all the verification flow is ran again, and the critical path optimized.
Moreover, the characterization of memory timings is based on designer assumptions about signal races. These assumptions are never checked in the CAD flow.

3.3. Phase2: Full cut verification

When the full cut is finished, the performances computed at the critical path point of view is compared with the data provided by the full cut simulation.



3.3.1. QA methodologies

3.3.1.1. Static verification methods

The LVS and DRC verification methods are applied in the full cut to check if the design is well built.

3.3.1.2. Simulation methodology

The full cut is checked with a dedicated stimuli call tight stimuli.
This stimuli is using the setup and hold timings computed on the critical path. If the simulation is succeeded, the designer concludes that the full cut timings are equal or smaller that the timings computed on the critical path.

The full cut verification is very complex because the parasitic of the full cut must be computed before the run of the simulation. The extraction phase is not easy and use amount of disc space and run time. The simulation run time is very huge too.

Cut	Extract run time	Disk space needed/used by the tool	Simulation run time by simulation	Disk space used by simulation
eSRAM 64x16	1h	200Mo / 500Mo (x2)	4h (x2) (x4)	50Mo (x2) (x4)
eSRAM 256x106	2h	500Mo/2Go	12h (x2) (x4)	50Mo (x2) (x4)

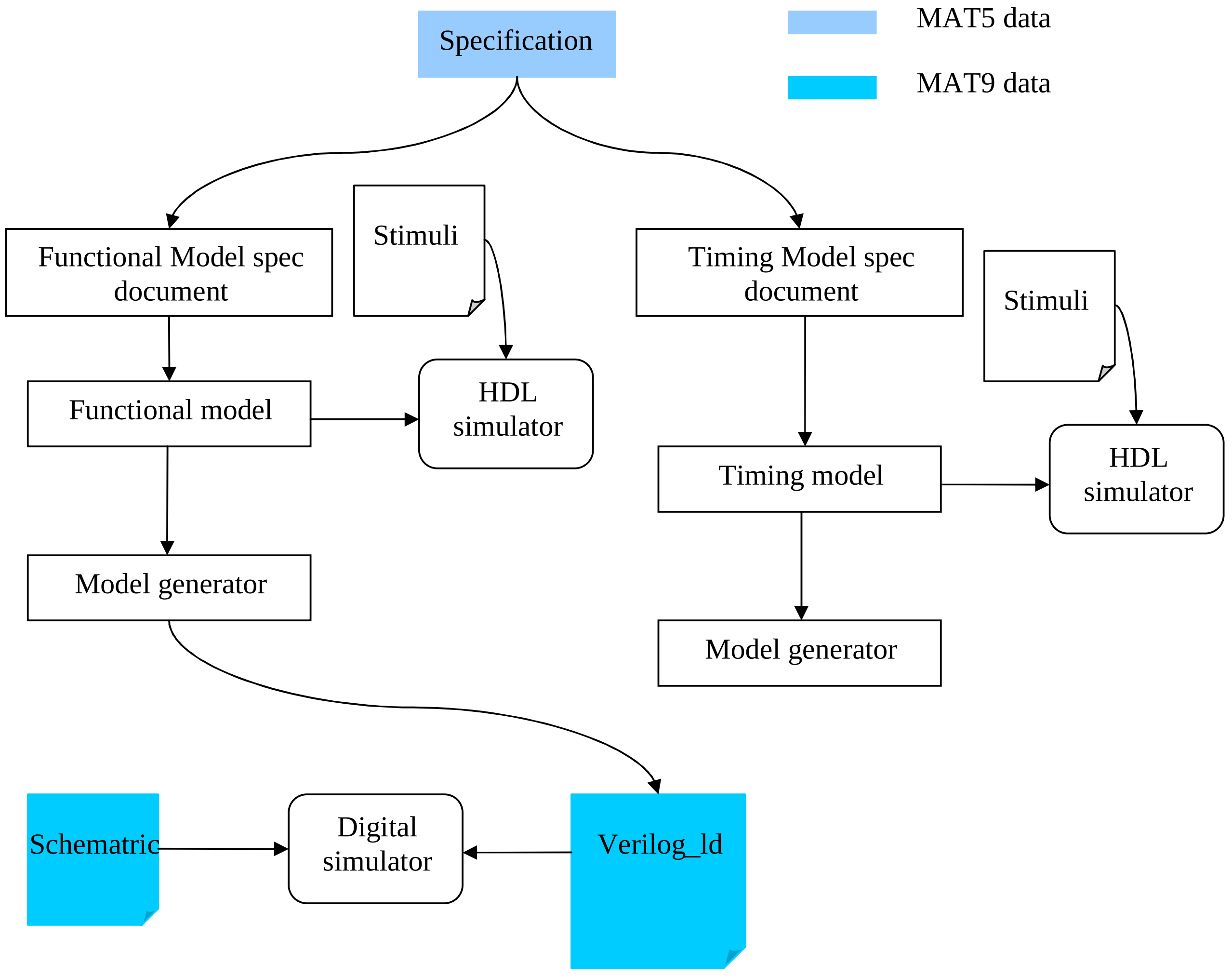
(x2): extraction and simulation performed for RC min and RC max.
(x4): Number of process corner simulated

3.3.2. Conclusion

The verification of the full cuts verification is not obvious.
Firstly, the development of the vectors is complex.
Secondly, the extraction and simulation steps use many time and disk space.
The other issue is that the full cut verification cannot be performed on biggest cuts. A strategy should be developed if the verification of these cuts is needed.

4. HDL models design

The flow used to build the verilog models is the following:



4.1. Functional view development

Functional views are developed to allow the digital functional simulations of the SoC with the memory by the customer.
There are two models developed:

- Full functional model
- Functional model with timings

4.2. QA Methodologies

The functional model is developed and checked with several simulations. The behavior of the design is compared automatically at the end of the simulation.
When the model is finished, the functional model is compared with the full cut schematic by a digital simulation: vectors are applied at the input of the functional view and at the spice view. The outputs of both descriptions are compared.
The timed model is only checked by simulation with expected results and expected timing violations.

4.3. Conclusion

The simulation between transistor netlist and verilog netlist is not obvious: The tool translates the netlist into an internal format in order to run the simulations. This step is very complex and difficult to debug: the accuracy of the internal delays of the memory does not always allow the right behaviour of the design. Manual modifications are sometimes needed to have the right behaviour of the memory in timing point of view.

5. General conclusion

The development of a new memory compiler is very expensive. The development process needs many resources, and the compiler has to be check on silicon to reach the MAT30. I underline the fact that the development of Silicon is more and more expensive (the mask set manufacturing cost has been multiplied by a factor 10 in about 3 technologies).
Moreover, the debugging of silicon is very complex and only few signals can be studied.
On the top of that, leak into CAD process development and verification could only be detected either at the full cut verification or at least on the silicon.
As explained in the introduction, the memories are key items for products and a delay due to memory issue could kill the project.

Another point is that today, the verifications are only done by transistor based simulations which are very accurate but very costly in term of run-time and never exhaustive. Hence the verification engineers must optimize vectors in order to win time to the detriment of the verification coverage.
The expected values at the output of the memory are built by the verification engineer, and it is not checked before the run of the BIST on the silicon, and sometimes, the run of the final product.

At this time, an automate way to check the timings and the functionality of the memory could, at least improves the performances of the memory (decrease the unused margin) and assures that the computed timing and the memory behaviour is safe in all cases. The designer will be in this case more confident in CAD verification process.