



IST-2001-34561

MUMOR

D1.3

Design environment

Document Number: IST-2001-34561/LETI/WP1/R/Pub/001/05

| | |
|---|-------------------------------|
| Contractual Date of Delivery to the CEC: | September 2002 |
| Actual Date of Delivery to the CEC: | 30.08.2002 |
| Author(s): | LETI |
| Participant(s): | LETI, NOKIA, UNIS, STM |
| Workpackage: | WP1 |
| Est. person months: | 15 |
| Security: | Public |
| Nature: | Report |
| Version: | v05 |
| Total number of pages: | 18 |

Abstract:

This deliverable presents the design flow and methodology for the MUMOR project. It addresses both baseband and RF design environment that will enable to seamlessly move from high-level algorithmic or architectural mono-mode designs to a unique re-configurable multi-mode architecture that will be refined down to the demonstration environment.

Keyword list: re-configurability, multi-mode, methodology, design flow, architecture, front-end, baseband

Table of contents

| | | |
|----------|--|-----------|
| 1 | INTRODUCTION | 5 |
| 2 | HIGH LEVEL MONO-MODE REFERENCE MODELS | 5 |
| 3 | BUILDING THE RE-CONFIGURABLE REFERENCE MODEL | 7 |
| 3.1 | FUNCTIONAL BLOCK SPLIT..... | 7 |
| 3.2 | FUNCTIONAL GROUP ANALYSIS | 8 |
| 3.3 | INTRODUCING IMPLEMENTATION SCENARIOS..... | 9 |
| 3.4 | RE-CONFIGURABLE REFERENCE MODEL IMPLEMENTATION | 10 |
| 4 | RF FUNCTIONAL BLOCKS OF THE REFERENCE MODEL..... | 10 |
| 4.1 | BICMOS7 BASED DESIGN..... | 10 |
| 4.2 | MEMS BASED DESIGN | 11 |
| 5 | BASE-BAND FUNCTIONAL BLOCKS OF THE REFERENCE MODEL..... | 12 |
| 5.1 | HW/SW PARTITIONING..... | 12 |
| 5.2 | BASE-BAND MODULE DESCRIPTION | 13 |
| 5.3 | HW DIGITAL MODULE DESIGN FLOW | 14 |
| 5.4 | SW MODULE DESIGN FLOW..... | 14 |
| 6 | DEMONSTRATOR INCREMENTAL INTEGRATION..... | 15 |
| 7 | LIST OF DESIGN TOOLS AND VERSIONS | 17 |
| 7.1 | SYSTEM LEVEL SIMULATION..... | 17 |
| 7.2 | FRONT END SIMULATION AND DESIGN | 17 |
| 7.3 | BASE-BAND SIMULATION AND DESIGN | 18 |

List of figures

| | |
|--|----|
| Figure 1: design flow overview | 5 |
| Figure 2: reference model description for each mode addressed by the reconfigurable architecture | 6 |
| Figure 3: definition of the reference model for mode #i..... | 6 |
| Figure 4: introducing re-configurability | 8 |
| Figure 5: high level multi-modes matrix block diagram..... | 8 |
| Figure 6: Modelling of different configuration appearances | 9 |
| Figure 7: High-level re-configuration (left) replaced by re-configurable design (right)..... | 10 |
| Figure 8: RF MEMS simulation flow | 11 |
| Figure 9: HW/SW partitioning | 12 |
| Figure 10: module description refinement..... | 13 |
| Figure 11: HW module design flow for FPGA..... | 14 |
| Figure 12: SW module description for DSP or micro-controller..... | 15 |
| Figure 13: module integration in the demonstrator..... | 16 |

Abbreviations / Terminology

| | |
|------|---|
| BER | Bit Error Rate |
| CCS | Code Composer Studio |
| CCSS | Co-Centric System Studio |
| DSP | Digital Signal Processor |
| FPGA | Field Programmable Gate Array |
| HW | Hardware |
| IP | Intellectual Property |
| PAR | Place And Route |
| RT | Real Time |
| SW | Software |
| VHDL | Very high speed Hardware Description Language |

1 Introduction

The definition, description and design of the architecture of MuMor will be based on a common methodology in order to facilitate data, building blocks and algorithms/architectures exchange within the project. The re-configurability of one part of the design may impact the overall architecture performance. Therefore, a system level simulation model is required to analyse how blocks interact and how re-configurability control messages are passed between blocks.

Figure 1 gives an overview of the system level methodology that will be detailed in this document. Re-configurability will be introduced in both the RF front end and the baseband. This implies that these two sections will cooperate to make a re-configurable system as a whole. Considering several mono-mode models, re-configurability will be introduced to build up a re-configurable unique model that will be flexible enough to address these modes. The re-configurability impact is then analysed to select the best strategy for re-configurability. This leads to a re-configurable and multi-mode reference model that will be refined down to a description that will fit in the demonstrator. The introduced “Common FE/BB simulation” does not necessarily mean simultaneous simulation. Beside the difficulty in putting together RF and baseband simulation tools those simulations would take long time due to the high accuracy of the RF simulation part. However for the simulation of the baseband receiver the data input could be from a stimuli file, which has been written out by a RF simulation before, thus containing data modified by all effects applied in this RF simulation. For baseband simulations, which do not require such an accurate modelling of the RF behaviour a SystemC modelling of RF effects is also introduced in this document.

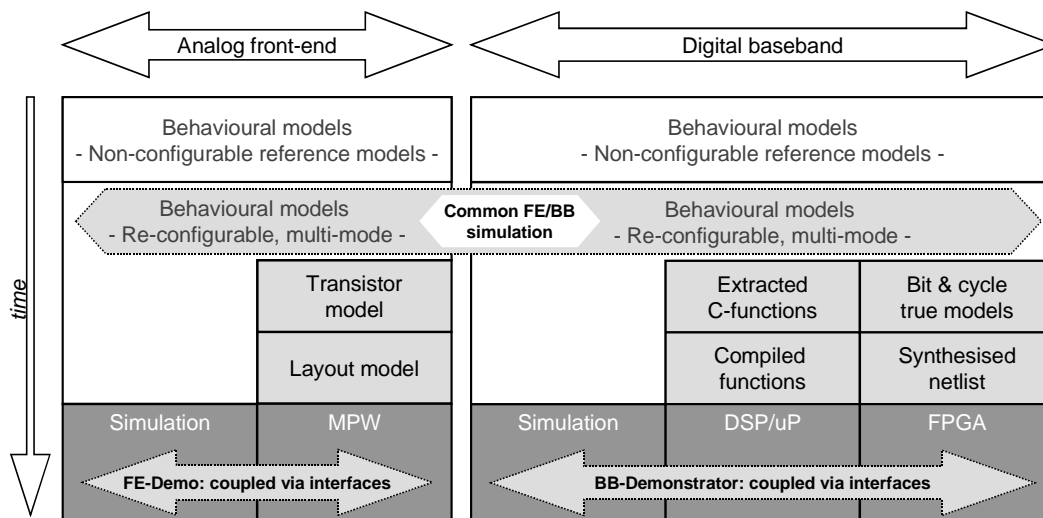


Figure 1: design flow overview

2 High level mono-mode reference models

- *Task of concern* : T1.2, T3.1
- *INPUTS*: project description, standards (from T1.1), block library
- *OUTPUTS*: provide with the first system description model for each mode and with the identification of the block which will take advantage of re-configurability (high level)

This first step aims at collecting information from the standards and project requirements in order to define a system block diagram for each mode addressed by MUMOR. The requirements will be

analysed to build a reference model for each mode consisting of a high-level system description and a testbench as illustrated by the flow of Figure 2.

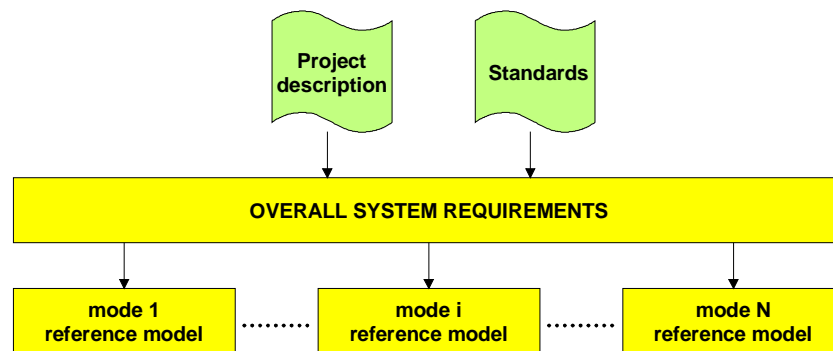


Figure 2: reference model description for each mode addressed by the reconfigurable architecture

Figure 3 illustrates the design flow to obtain a mono-mode reference model. A behavioural model of the RF section should be given in SystemC to enable overall system simulation. This simulation model will consist of a high level description, which focuses on the imperfection of the RF section (additional noise, non linearity modelling), without taking the actual structure of the front end into account. The characteristics of the RF section have been derived from accurate RF simulations or actual measurements. The goal of this simulation is to provide the baseband section with “realistic” stimuli without dramatically increasing the duration of the simulations.

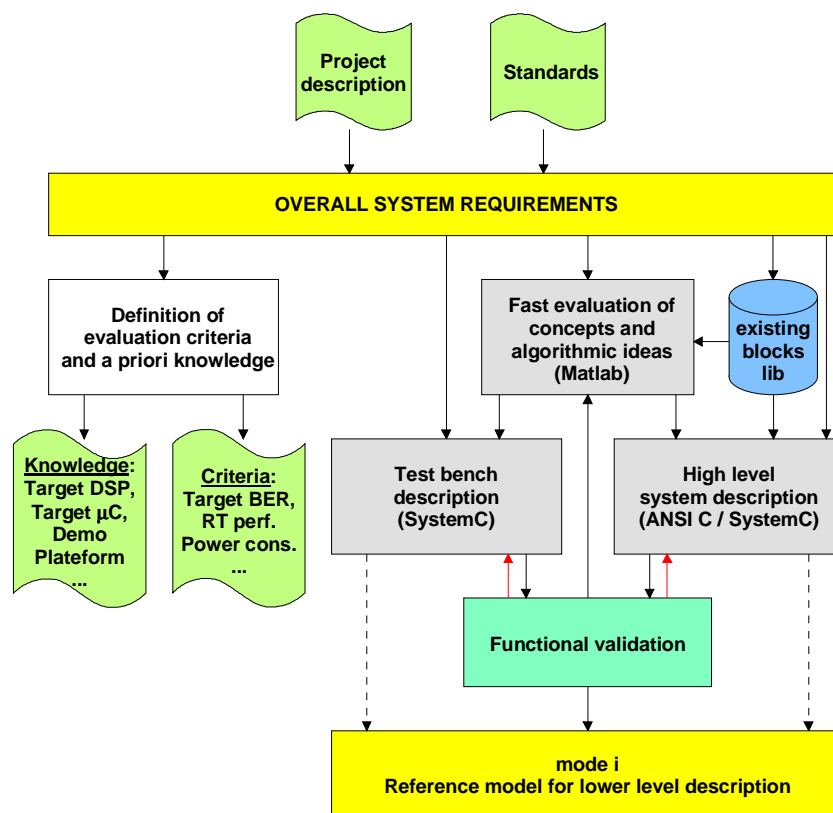


Figure 3: definition of the reference model for mode #i

Also coming from project requirements and standards, a list of evaluation criteria and a list of a priori knowledge are established for each mode. The evaluation criteria will include all the parameters that

will enable to measure the model “quality”: re-configurability, BER, timing requirements, etc that will be used at all levels of description. At this step, the lists will be roughly established and will be refined afterwards as the modelisation goes along. All these requirements will not be met by the reference model but will be used as input guidelines for the lower level models.

The high level model provides a means to perform fast simulation to provide with a reference model for the corresponding mode. SystemC will be used for this level of description since this language has useful features to describe software/hardware and testbench blocks.

3 Building the re-configurable reference model

- *Task of concern : T1.2, T2.1, T3.2, T3.4*
- *INPUTS: high level reference model for each mode*
- *OUTPUTS: re-configurable multi-mode reference model*

3.1 Functional block split

Bearing in mind the different single mode reference model, the multi-mode re-configurable model will be defined.

At the RF level, parameterizable blocks and switchable blocks will be analyzed. New technologies such as MEMS will be studied in order to evaluate how they could contribute to the re-configurability of the front end.

For the baseband part, an analysis will be carried out at a system level in order to state how re-configurability methodology will be inserted and how the re-configurable RF front end will be re-configured. This will be the job of the master control unit, which will configure the whole system for re-configurability.

Introducing re-configurability into the system model is likely to impact the overall system performance. This impact must be evaluated according to some criteria and the re-configurable model may be modified to fit performance tradeoffs.

Figure 4 illustrates this step of the design methodology.

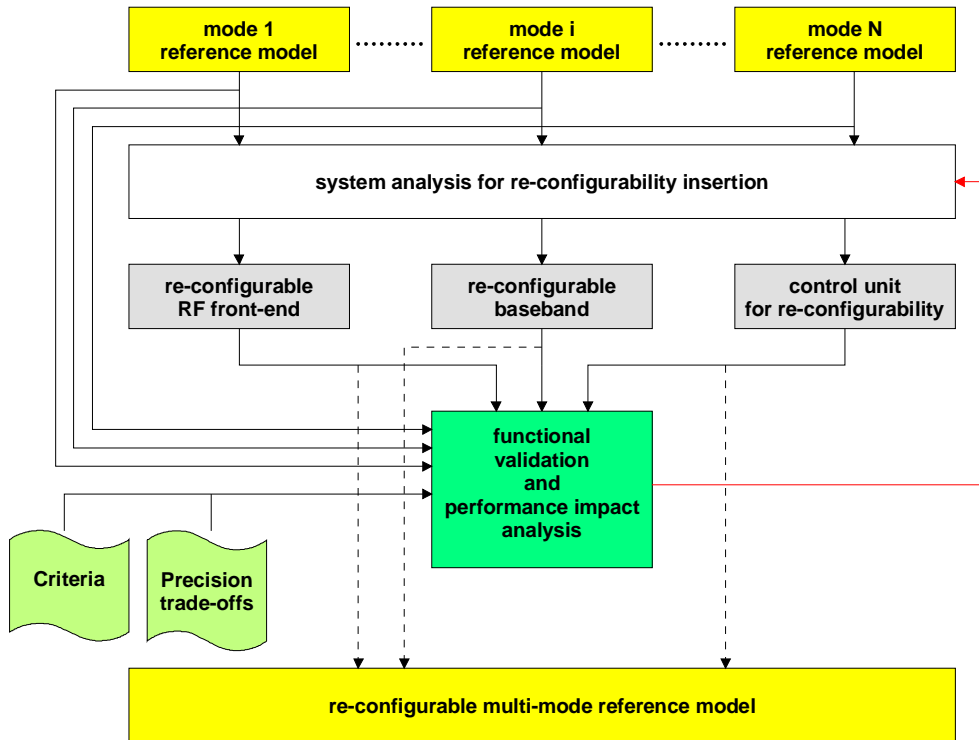


Figure 4: introducing re-configurability

In order to define how re-configurability can be used between the different mode reference models, a matrix representation of the architecture is introduced. In this matrix like block diagram, the rows represent the different modes addressed whereas columns consist of functions that are similar in some aspects. In this representation, a column represents a functional group, i.e. a group of blocks belonging to different modes but having similar or identical functionalities. This matrix block diagram is illustrated by Figure 5.

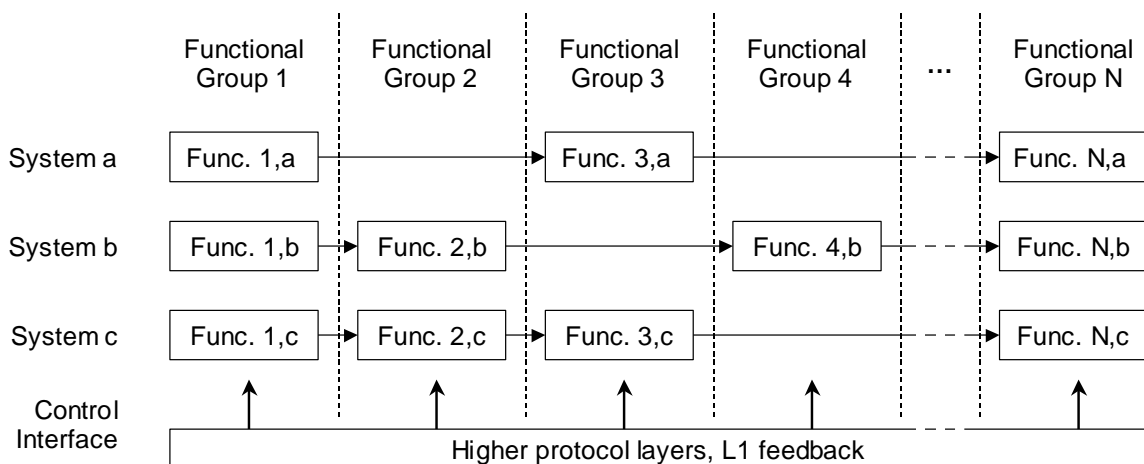


Figure 5: high level multi-modes matrix block diagram

3.2 Functional group analysis

Blocks will be investigated to determine the type and possibility of their re-configuration. Each “vertical” functional group is inspected in terms of the similarity of functions. Flags are given to each “functional column” to indicate the category of re-configurability :

- Red: No (inner) re-configurability is possible at least not at reasonable cost in terms of power and/or complexity. To suit all modes in the final implementation these blocks will be framed by switches or multiplexers, which will be controlled via the configuration interface. The blocks themselves are not addressable via the configuration interface.
- Red-and-Amber (for baseband blocks only): If this block were implemented in HW, re-configuration would not be reasonable. However this flag indicates that this block will be implemented in SW, in a re-configurable way.
- Amber: Hardware re-configurability via a configuration interface is reasonable.
- Green: No re-configurability required, because these blocks all have identical functionality in *all applicable modes* without modification.

3.3 Introducing implementation scenarios

To enable further analysis of the re-configurability possibilities, the way functional blocks are expected to be implemented is introduced. This is particularly the case of baseband blocks where either HW or SW can be envisaged.

Depending on their respective re-configurability flag, the blocks will be changed from ‘red’ to ‘amber’ – for HW-reconfiguration – or to ‘red-and-amber’ for the SW re-configurable blocks. In the end all blocks in the same column of the architecture matrix must have the same colour. If this is not the case additional column have to be installed to put the blocks with different colours into different columns to emphasize the obvious differences in their functions or their different properties with respect to re-configurability.

The simulation of the system model at this stage will provide with a rough performance and feasibility feedback that will be balanced against the colour modification to check if this colour change is reasonable.

At this stage, all functional groups (columns), which are not green, but any other category will be implemented framed by a pair of de-/multiplexer or switches. The MUXes are configured by the 3-state (or 3-valued) configuration signal, which indicates the current operating mode (Figure 6, column i+1). The same will have to be done for all the blocks, which are green, but not having *all* functionality in *one* branch, i.e. a branch with no functionality for at least one mode (Figure 6, column i+2 and i+3). Only the green columns, which really cover all three modes in *one* functional block without any exception (Figure 6, column i), do not need any additional de-/multiplexer.

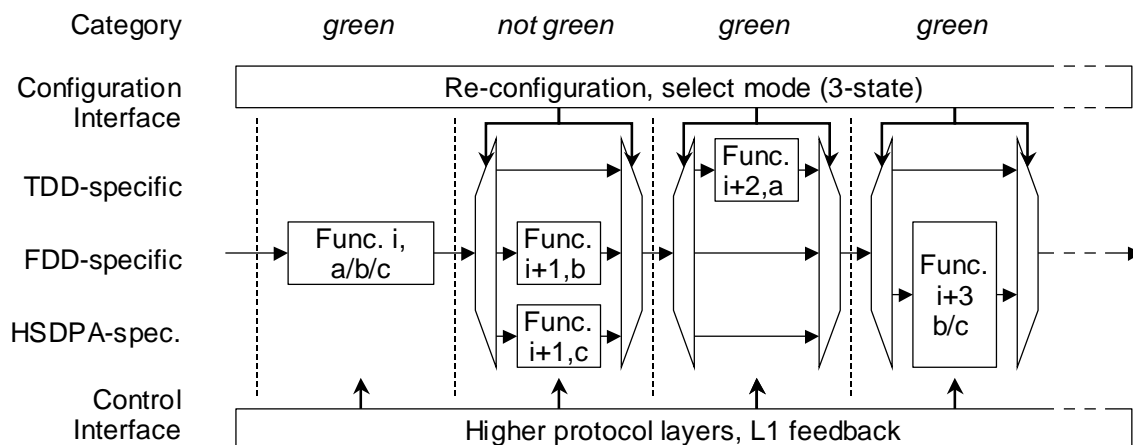


Figure 6: Modelling of different configuration appearances

3.4 Re-configurable reference model implementation

At this step of the design flow the multiplexed functional groups (columns), which are declared neither with the red flag nor with the green flag, will be replaced by a re-configurable design with the appropriate configuration logic translating the high-level 3-valued re-configuration information into a more detailed signalling for the multi-mode (hybrid) design block (Figure 7). The interfaces (data, control, configuration) of the entire block do not change while replacing.

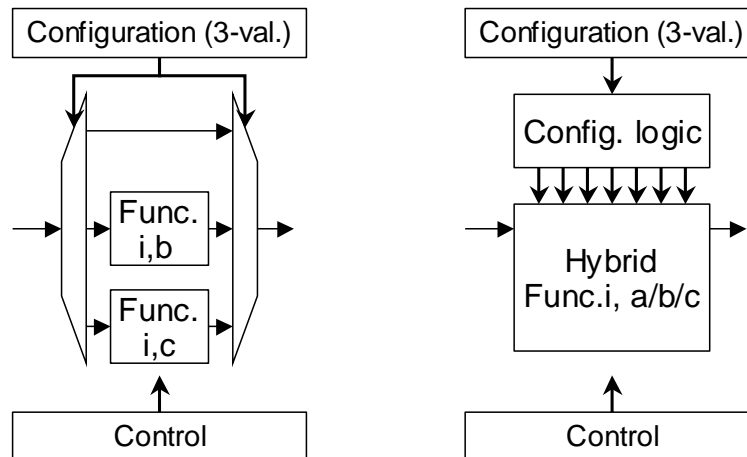


Figure 7: High-level re-configuration (left) replaced by re-configurable design (right)

Extending this method to all column lead to the re-configurable reference model that will be analysed for implementation.

4 RF functional blocks of the reference model

4.1 BiCMOS7 based design

- *Task of concern* : T2.2, T2.3, T2.4, T4.1
- *INPUTS*: behavioral description and design kit
- *OUTPUTS*: refined simulation models and silicon

First of all the complete front-end has to be modeled at a behavioral level. Multi-mode is achieved through parameterization of the models. Models, which will be implemented on silicon, have to be developed further on at a transistor level and then at a layout level.

Implementation will be based on BiCMOS7 design kits provided by ST Microelectronics. Two design kits will be provided:

- for Cadence Analog Artist giving access to Spectre and Eldo simulators,
- for ADS simulator. Refer to table below for available versions and required environment.

Transistor level simulations can be performed with Spectre, Eldo or ADS as needed.

Cadence Virtuoso is the only tool supported for drawing layouts.

For physical verification purposes (DRC, LVS) Cadence DIVA can be used but it is strongly recommended to use Calibre from Mentor. However using Calibre is mandatory for the layout finishing and the sign-off verification (DRC, LVS, antenna).

Parasitics from interconnections may be extracted with DIVA but it is strongly recommended to use Arcadia (based on Calibre) instead for it provides more accurate results.

4.2 MEMS based design

- *Task of concern* : T2.1
- *INPUTS*: characterized devices
- *OUTPUTS*: simulation models and libraries

The microelectromechanical systems (MEMS) offer many benefits in radio frequency (RF) applications. These benefits include better RF performances than solid state devices (in many cases), low power consumption, simplicity of principle, very good linearity, compatibility with standard IC technologies and potential low costs manufacturing into a variety of substrates.

The RF MEMS technology is today at an inflection point, several MEMS devices with new levels of performances have been achieved and now the goal is to propagate the device-level benefits up to the system level to attain unprecedented levels of system performances.

The designer need models of MEMS basic elements in order to incorporate them into their applications. In the project MUMOR, LETI will provide efficient electromagnetic models of different MEMS devices that have been already characterized (such as switches) or that will be derived from our knowledge. The final goal is to give tools to designer to enable them to imagine and simulate new concept of front-end architectures that would exploit to the highest degree possible the advantages of MEMS.

The flow of conception and the tools employed are summarized in Figure 8.

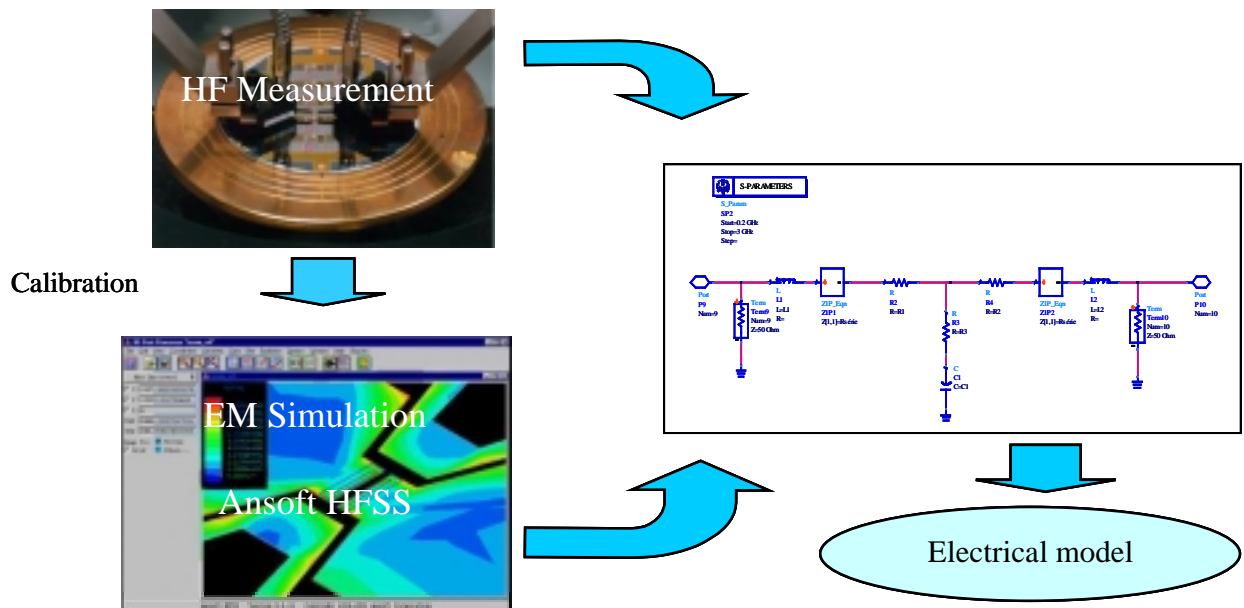


Figure 8: RF MEMS simulation flow

The first step is to obtain the S-parameter of each MEMS device. They can be obtained from measurements if the device has been already fabricated or from a full wave electromagnetic simulation. Two simulators will be used : HFSS from Ansoft and Momentum from Agilent.

Then, the parameters of the equivalent circuit model are extracted and optimised to fit the S-parameters. By repeating this procedure for different dimensions (for instance different switch height), we can extract a parametric scalable model.

Finally, the lumped elements model is implemented into ADS from Agilent and will be available for system level simulations.

- Validity of the models

This flow of conception has already been validated in the LETI.

5 Base-band functional blocks of the reference model

5.1 HW/SW partitioning

- *Task of concern : T3.3*
- *INPUTS: high level reference model*
- *OUTPUTS: bit true reference model*

This stage takes the reference re-configurable model as an input. The reference model should not be modified at this step.

First a rough coarse grain HW/SW partitioning considering a priori knowledge on the target and RT performance tradeoffs (Figure 9). This partitioning leads to some description models that may include fixed-point operators in order to obtain a “bit true” reference model of the system. At this step, no target dependent code is used and the output reference model is a refined view of the floating point reference model that includes fixed point operators. The simulation of the “bit true” model will be balanced against the reference model simulation to analyse the precision loss and to decide if the partitioning should be modified (e.g. use floating point operators instead of fixed point at some stage, etc). The new reference model will be optimised in the next steps of the flow.

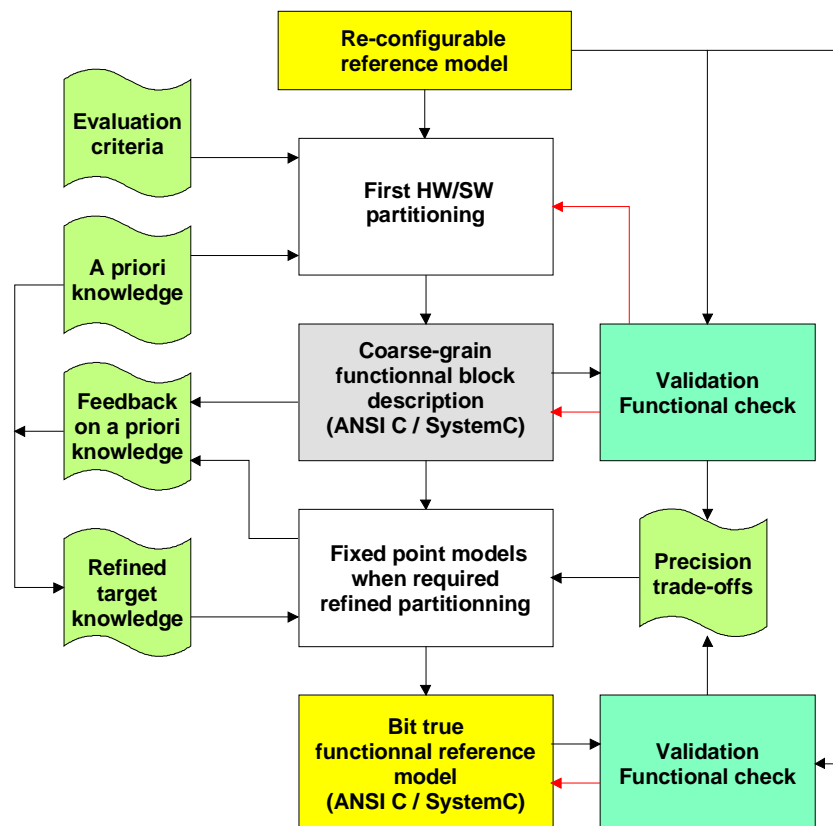


Figure 9: HW/SW partitioning

5.2 Base-band module description

- *Task of concern* : T3.4
- *INPUTS*: bit true model
- *OUTPUTS*: SW block model and VHDL/RTL block model

At this stage, the HW/SW partitioning has been defined and each module description must be refined. First, interface between blocks is refined to enable unary simulation of each module. Transaction level model gives a more detailed model of the functional bus protocol. This step might be omitted if bus transactions are simple. In that case, a bus cycle accurate model is directly designed.

Once the modules' interface have been properly defined, it is possible to derive two separate design flows for HW and SW modules, including the testbench for each unary module (Figure 10). HW and SW blocks will be considered separately as long as synthesis tools do not provide with an efficient SystemC¹ entry. In this case, VHDL will be used for HW block description. It may also occur that some partners would like to reuse VHDL IP inside some blocks, which would also imply a VHDL design flow.

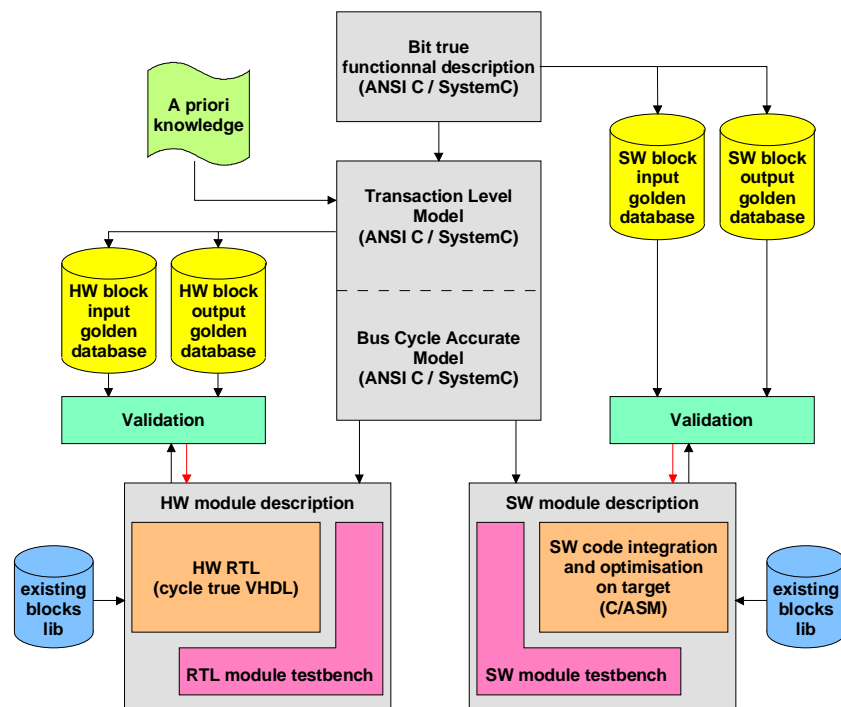


Figure 10: module description refinement

Block validation will be obtained using golden reference data files derived from higher level description models or using some design environment that enable SystemC/VHDL co-simulation (e.g. CCSS from Synopsys).

For SW modules, a “stand-alone” version of the code implemented on the target device will be derived from the bus cycle accurate model (local constant definition, memory mapping...).

¹ If SystemC entry is available, the design flow is more simple since only SystemC models are considered for both HW and SW blocks.

The module partitioning at the end of this step should correspond to the module repartition on the demonstrator platform to facilitate the translation to the demonstrator.

5.3 HW digital module design flow

- *Task of concern* : T3.5, T4.2
- *INPUTS* : unitary RTL description
- *OUTPUTS (GOAL)* : HW block in situ validation

At the beginning of the project, RTL VHDL is the most realistic description language to address synthesis tools (SystemC synthesis tools are expected but not wide spread and technical background is missing on such an approach).

PAR will be achieved using standard FPGA vendor suites (e.g.: Xilinx ISE...). Post PAR simulation using VITAL library will give a strong validation of the block (Figure 11).

In situ validation will then show that the module is working properly in its actual demonstrator environment. If some interface troubles are detected here, this means that the bus cycle accurate is not consistent with the platform environment and must be modified.

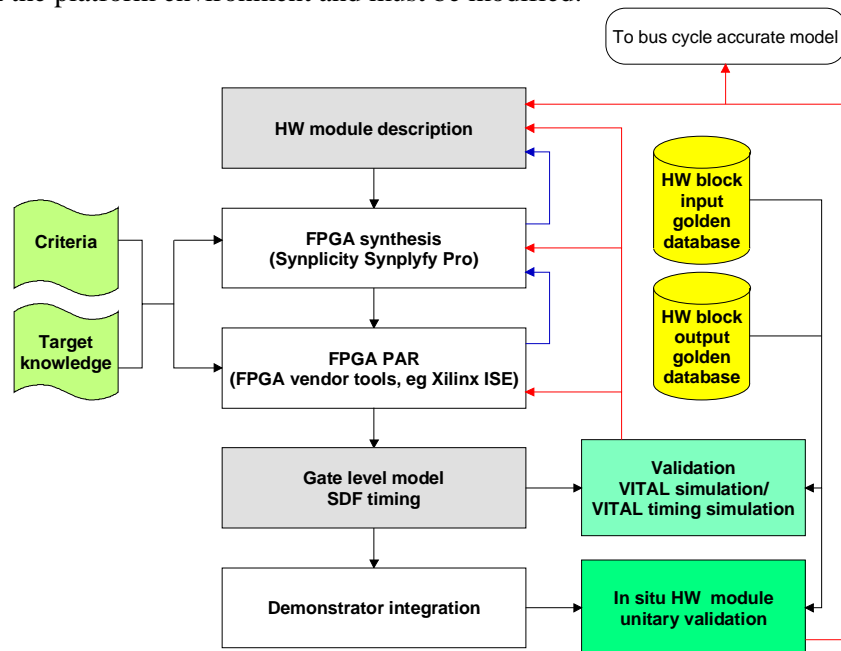


Figure 11: HW module design flow for FPGA

5.4 SW module design flow

- *Task of concern* : T3.5, T4.2
- *INPUTS*: unary SW description
- *OUTPUT*: in situ SW validation

In order to increase RT performance, SW code might be optimized. ASM and target dependent code will be used as scarcely as possible to preserve reusability and readability of the code (Figure 12).

In situ validation helps to provide with a good performance evaluation feedback. DSP vendor tool suites (e.g.: Texas Instrument CCS) include in situ simulation and debug options to make integration

easier. This gives an opportunity to use in situ simulation as soon as possible and accelerates simulation time.

As for HW blocks, SW block interface may be refined in the block cycle accurate description if description errors are revealed at this step.

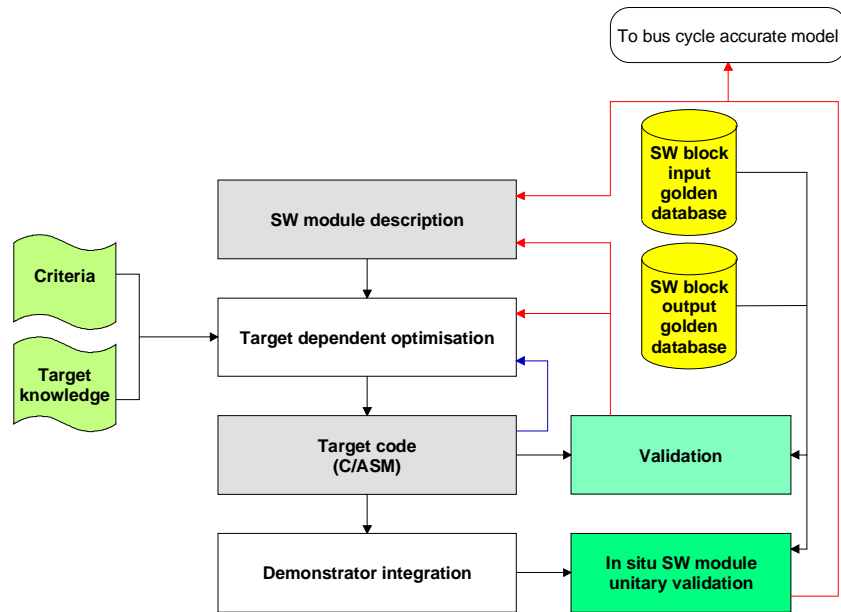


Figure 12: SW module description for DSP or micro-controller

6 Demonstrator incremental integration

- *Task of concern* : T4.2
- *INPUTS*: unary SW and HW descriptions
- *OUTPUTS*: global in situ validation

In order to move seamlessly from block validation to system demonstrator, the integration of the blocks will be done incrementally as shown in Figure 13. Test pattern will be used to validate the communication between the modules. A correct functionality within each module is assumed at this stage.

As detailed above, any inconsistency with higher description level shall be reported and integrated in a revised version of these descriptions. Of course, only slight modifications are tolerated at this stage.

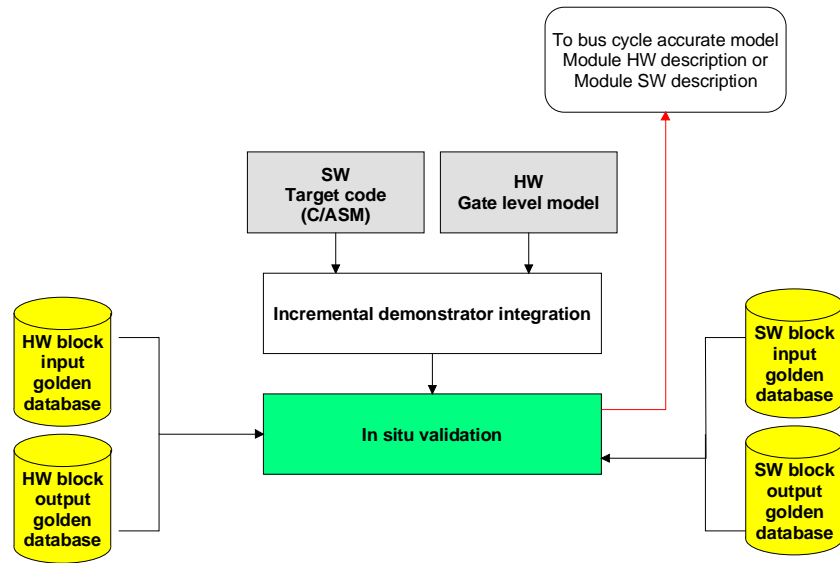


Figure 13: module integration in the demonstrator

7 List of design tools and versions

7.1 System level simulation

| Tools | Vendor | Function |
|---------------------------|--------------------------------|--|
| System Design Flow | | |
| CoCentric System Studio | Synopsys | Behavioral modeling of digital baseband and front-end (complex baseband representations) functions. The front-end models should be easily generated out of the ADS environment. |
| Matlab | MathWorks | Matlab may be needed if models are available already. |
| SystemC | OSCI (Open SystemC Initiative) | SystemC provides with a good description language for system description. |

7.2 Front end simulation and design

| Tools | Vendor | Version | Function |
|---|----------|--|--|
| ADS | Agilent | 2002 | Front-end system simulation |
| Design Framework II | Cadence | 4.4.6 (QSR2 - SunOs 2.7) ² | Used for transistor and layout modeling. ADS may also be used if applicable. |
| BiCMOS7 Design Kit for Cadence Analog Artist (Spectre and Eldo) | ST | 4.0 | Release 4.4.6.100.29_57_ADS15 (QSR2 - SunOs 2.7) of Cadence is required. |
| BiCMOS7 Design Kit for ADS | ST | 4.0 | Release 1.5 (Service Pack 1A) of ADS is required. |
| AMS | Mentor | 2001.3 | Eldo, ModelSim,... |
| Arcadia | Synopsys | 5.5-P81 | Interconnection parasitics extraction |
| Calibre | Mentor | 2001.8 (v8.8_13.1 Aug2001) | Physical verification, layout finishing and sign-off. |
| SubstrateStorm ³ | Simplex | 3.5 | Substrate coupling |

² The needed version is dependent on the requirements from the Design Kit.

³ Product not supported yet but assumed to be in the near future.

7.3 Base-band simulation and design

| Tools | Vendor | Version | Function |
|---|------------|---------|--|
| Digital Baseband Design Flow | | | |
| CoCentric System Studio with SystemC functionality | Synopsys | | SystemC allows high level behavioral system modeling as well as bit and cycle true modeling |
| Design Compiler with SystemC functionality ⁴ | Synopsys | | This will allow the development of netlists used for the FPGA environment without writing VHDL code. |
| VHDL | | | VHDL will be used to give a RTL description of the HW blocks that will be synthesized on FPGA. |
| Modelsim | Model Tech | | Modelsim will be used for simulating building blocks described in VHDL. |
| ISE | Xilinx | 4.1i | FPGA design environment and PAR |
| DSP/uP compiler (gcc, would be preferable) | DSP vendor | | C will be used as much as possible for DSP implementation. Generation of DSP specific code. DSP code should be generated from the SystemC models easily if the core functionality is encapsulated in external C functions. |

⁴ If available.