



FP7 – 216693 - MULTICUBE Project
 MULTI-OBJECTIVE DESIGN SPACE EXPLORATION OF MULTI-PROCESSOR SOC
 ARCHITECTURES FOR EMBEDDED MULTIMEDIA APPLICATIONS

**Deliverable D4.2.3: Demonstrator of use cases:
 Multimedia Applications**

Revision [8]

Delivery due date: M30 (June 2010)
Actual submission date: September 24, 2010
Lead beneficiary: IMEC

Dissemination Level of Deliverable		
PU	Public	X
PP	Restricted to other program participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	
Nature of Deliverable		
R	Report	
P	Prototype	X
D	Demonstrator	
O	Other	



Author(s):	Prabhat Avasare (IMEC), Hector Posadas (UC)		
Reviewer(s):	Vittorio Zaccaria (POLIMI) and Carlos Kavka (ESTECO)		
WP/Task No:	WP4/T4.2	Number of pages:	26
Identifier:	D4.2.3_v8	Dissemination level:	Public
Issue Date:	Sept. 24, 2010		

Keywords: Prototype, Open-Source, Modeling, Multimedia, Abstraction levels

Abstract: This deliverable (issued at month 30) represents the accompanying report of the **multimedia demonstrator**. More in detail, this deliverable describes how to run the multimedia demonstrator which is available in MULTICUBE virtual machine (released as **D1.4.2 “Integrated Design Tool Prototype”**) and also available on the web as a *public demonstrator* for dissemination purposes.

Being public, this demonstrator has been used to validate the MULTICUBE open-source design flow and related tools. In particular, this demonstrator exploits M3-SCoPE simulator combined with M3Explorer tool to perform extensive Design Space Explorations (DSE) for a multimedia application. The open-source tool-chain has been tested on an industrial use case provided by IMEC including the MPEG4 encoder. To enable the public distribution of the demonstrator including the IMEC proprietary application code, the MPEG4 encoder IP has been distributed in binary form (under acceptance from the user of the IMEC license agreement reported in the Appendix I of this document).

The multimedia demonstrator has also been used to run DSE by using simulators at several abstraction levels providing some tradeoffs in terms of simulation speed and accuracy.

Overall, the demonstrator has proved the ease integration and interoperability of the tools composing the MULTICUBE design flow thanks to the validity of the XML interface defined in **D1.4.1. “Definition of the specification of design flow integration”**.

Approved by the Project Coordinator:



Date:

September 24, 2010



Index

1	Executive Summary	4
2	Introduction.....	5
2.1	Exploration flow	6
2.2	Use Case Description.....	7
2.3	HW platform description	8
2.4	Design space and metrics	9
3	Modeling with M3-SCoPE	10
3.1	Modeling of SW components	10
3.2	Modeling HW platform components	11
3.2.1	Model of the communication assist	12
3.2.2	Model of the communication network	12
3.2.3	Bus and memory models.....	12
3.3	System modeling and link to M3Explorer.....	12
4	How to access the demonstrator	14
4.1	System Requirements	14
4.2	Downloading and installation	14
4.3	Execution and exploration results.....	15
4.3.1	Demonstration of the integration between the simulator and MULTICUBE Explorer	15
4.3.2	Demonstration of the HTML reporting capabilities of MULTICUBE Explorer	15
4.3.3	Exploration results in the web page.....	16
4.4	License for the MPEG4 encoder IP.....	16
5	Exploration Results	17
6	Conclusions.....	22
7	References.....	23



1 Executive Summary

This deliverable describes how to run the Multimedia demonstrator which is available in MULTICUBE virtual machine (released as **D1.4.2 Integrated Design Tool Prototype**) and also as a **public demonstrator**. The demonstrator is based on the MPEG4 encoder application described in **D1.3: Definition of the specification for industrial use cases** [1]. .

This demonstrator shows the open-source exploration flow developed in MULTICUBE project for Design Space Exploration (DSE). This open-source flow combines the use of M3-SCoPE simulator and M3Explorer DSE tool. It is important to note that the open-source quality is only related to the design tools. The Multimedia application code is an industrial use case provided by IMEC, of a high importance for the company, and thus, proprietary, only distributed in source-code form inside the Consortium, under the MULTICUBE Consortium Agreement. To allow the distribution of the open-source demonstrator to third parties for dissemination purposes, the application code is distributed in binary form. To enable the combination of proprietary code and open-source tools M3-SCoPE and M3explorer are released under LGPL and BSD licenses respectively. Additionally, the ARP library used to provide the ADRES processor execution times to M3-SCoPE is also proprietary and it is distributed in binary form.

The deliverable is organized as follows. Section 2 presents brief description of Multimedia use case i.e. MPEG4 encoder application. Section 3 explains how to perform DSE for multimedia use case using MULTICUBE-SCoPE simulator with M3explorer DSE tool. Section 4 describes interesting results that can be obtained by running this demonstrator. Section 5 concludes this report.



2 Introduction

To cover a broad range of application domains for validating MULTICUBE design flow, the application use cases have been selected from various domains e.g. multimedia applications, advanced computing (multi-core architecture and low-power processor). For multimedia domain, MPEG4 encoder is chosen as one of the use-cases because it is one of widely known/used real-life multimedia application. The encoding scheme inside an MPEG4 encoder has a high computational complexity and at the same time there is lot of scope for exploiting parallelism. Amount of parallelism available makes it an interesting case for doing a design space exploration for targeting a multi-processor platform. Thus MPEG4 encoder is a good candidate to drive the development of the performance analysis and design space exploration tools developed during MULTICUBE project.

This use case has been applied within the project to demonstrate three main results of MULTICUBE :

First of all, it has been used to demonstrate the open-source MULTICUBE design flow. First, this means that this use case checks the work done mainly in Task 3.1: “Definition of the open-source framework for design space exploration” and in Task 2.1: “Open-source performance and power consumption estimation framework” resulting in the open source tools M3Explorer and M3-SCoPE respectively. As based on open-source tools, this is a public demonstrator and will be used for dissemination purposes. It has been prepared for public download from the web.

Second, the use case has been used to demonstrate the cross accuracy of the estimations provided by different simulators developed within the MULTICUBE project scope. This allows analyzing the effects of using simulators at different abstraction levels, in terms of accuracy, exploration speed, etc. HLSim, M3-SCoPE and a CoWare TLM tools have been used for this purpose. As a consequence, the use case is more centered on modeling than in exploration issues, being the design space relatively small. The use of simulators at different abstraction levels implies that the use case must be adapted to run slow simulators as cycle accurate models. In order to obtain feasible exploration times, a design space with a reduced number of possible combinations is required. This document includes only the information related to the use of M3-SCoPE for the exploration of the Multimedia application. Information about the application of the use case with other two simulators (HLSim and CoWare) and the comparisons between the three simulators in terms of accuracy are included in D4.2.4: “Final report on the application of MULTICUBE design flow to the demonstrators” [3].

Third, as the use case has been defined to enable its application to different simulators, this use case enables the possibility of checking the validity of the XML interfaces for easy tool integration. Using the use case it is possible to check that the same DSE tool (M3Explorer) can perform an exploration with any simulator that accomplishes the requirements of the XML interface. The use case demonstrates that simulator used for the exploration can be changed without requiring any modification in the DSE tool, the XML files or the environment.



2.1 Exploration flow

The flow used for the exploration of the best configurations in the Multimedia Use Case, follows the general flow defined in the MULTICUBE project. The following figure shows the flow.

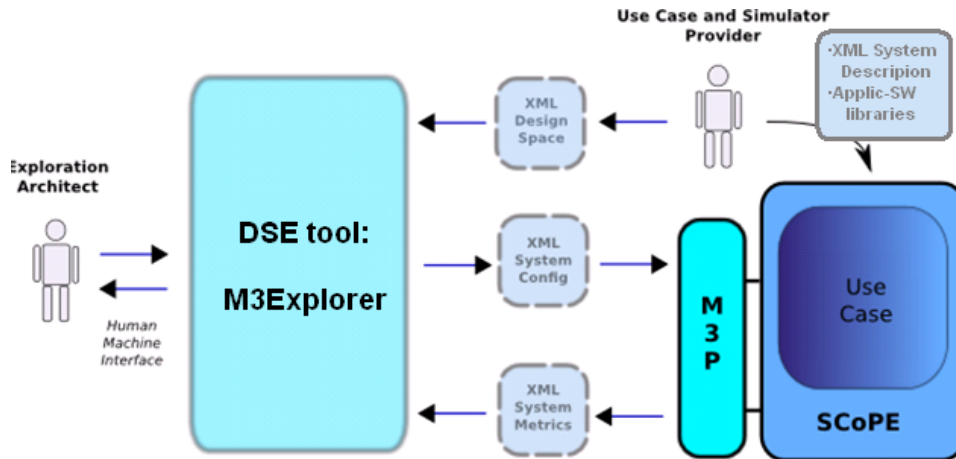


Figure 1. Multimedia use case exploration flow

As the figure shows, the use case provider generates a XML description of the system, a XML file defining the Design Space, and all the libraries with the specific code of the use case. In this example, some specific HW components are required, such as a communication assist and a switched network, so libraries containing both the application SW, compiled following the SCoPE requirements, and adequate HW component models are provided.

Then, the DSE tool, M3Explorer analyzes the design space and starts a loop where different possible configurations are selected and analyzed in order to find the set of best solutions, which conforms the Pareto front. To analyze each configuration, M3Explorer calls M3SCoPE simulator. The simulator, composed by SCoPE and the M3Plugin, takes the information provided by the use-case provider, and the information describing the selected configuration, creates the corresponding model, simulates the system and obtains the resulting metrics (execution time, power consumption, ...).

2.2 Use Case Description

This section briefly describes multimedia use case MPEG4 encoder used as a demonstration of MULTICUBE design flow. For detailed description of this use-case, please refer [1]. .

The MPEG-4 encoder is an industry-standard, block-based, hybrid video encoder consisting of a motion-estimation and compensation phase followed by transformation and entropy coding phases [4]. . The implementation used is based on the MPEG-4 Simple Profile reference code, which has been pruned and cleaned for the intended purposes. The MPEG4 encoding scheme is standardized and it has a high computational complexity.

The structure of an MPEG-4 encoder is shown in Figure 2. For more details see [5]. .

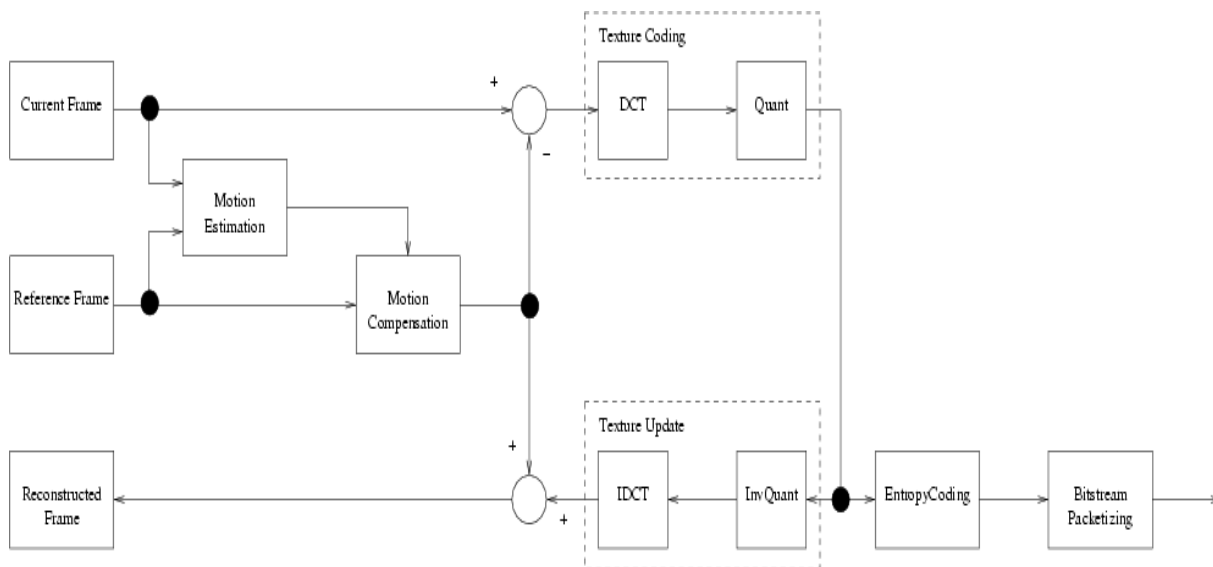


Figure 2: Overview of the MPEG-4 encoder application

Main functional blocks which can be recognized in MPEG4 encoder application are: MotionEstimation (ME), MotionCompensation (MC), TextureCoding (TC), TextureUpdate (TU), EntropyCoding (EC) and BitstreamPacketizing (BP). Major percentage of application execution is spent in computation-intensive nested loops inside these functional blocks. The computation-intensive loops, called kernels, have been optimized for compilation for VLIW architectures, more specifically the ADRES processor [6]. .

Six different parallel implementations of the MPEG-4 application have been created from the original sequential source code. These parallel versions range from 2 to 8 threads. The sequential version can serve as a reference for the other implementations. Performance figures (execution time, energy consumption, etc) are obtained by executing these multiple versions on different platform simulators. These obtained figures are validated against the cycle-accurate transaction-level simulator (Figure 3).



2.3 HW platform description

As specified in D1.3, the platform architecture is depicted in Figure 3. It is composed of 8 processor nodes and 1 memory node. The processor nodes are composed of an ADRES with and scratch-pad memory, for storing the data, L1 memory for storing instructions and a communication assist. The memory nodes are composed of a memory and a communication assist. A communication infrastructure connects all nodes together. A switch-based network has been applied for that purpose.

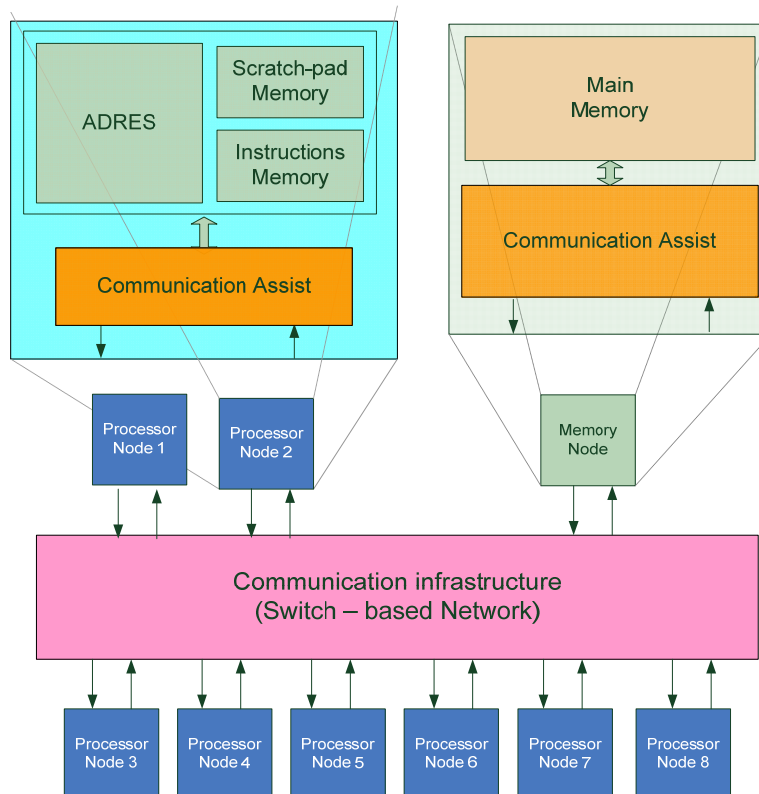


Figure 3. Multi-processor platform architecture for the multi-media use case

The main memory is only used for data transfer and storage as instructions are located in local memory in the processor nodes. Data transfer between the main memory and the processors scratch-pad memories are done through explicit instructions in the application code. Summarizing, the processor has no conventional caches, which means that no cache misses are produced.

As instructions are located inside the processor nodes, no task migration is possible among nodes. Task allocation is fixed. Furthermore, as there is only one task per node, no scheduling services are required.

All these characteristics enable the modeling of the use case at different abstraction levels, from an accurate, slow cycle accurate level model to a high-level model without OS support or cache estimation.

2.4 *Design space and metrics*

Following the exploration requirements defined in document D1.4.1: “Definition of the Specification of the Design Flow Integration” [2]. , in the design space exploration definition, the following four input parameters can be distinguished:

- **Threads:** indicates the number of threads used by the (parallelized) multimedia application. Hence, this parameter also defines which application code base is to be used. Since it is currently assumed that every processor can only run one single thread, this parameter also indicates the (minimal) number of processors required in the simulation model of the platform. Since all parallelizations require at least one master and one slave, the minimal amount of processors is two. And due to the limited set of available parallelizations, the maximum amount of threads is eight (one master and seven slave threads).
- **Freq:** This parameter describes the frequency of all active processors in the system. Thanks to the parallelization of the application, it will be possible to reduce the frequency of the processors while maintaining the application performance of the sequential application code, reducing also the overall power consumption. For this exploration a discrete set of frequencies was chosen between 20MHz and 200MHz (with steps of 20MHz).

The result of the design space exploration will be the optimal parallelization in terms of number of threads, where the performance is still maintained compared to the sequential code, but with minimal power consumption. Hence, there are only two system metrics required for this exploration:

- **Execution_Time:** the total time required by the application, which allows determining the performance of the application.
- **Power_Consumption:** total power consumption, which allows searching for the minimal power consumption design.



3 Modeling with M3-SCoPE

SCoPE and its extension for DSE M3-SCoPE are generic simulators. This means that the simulation infrastructure provides a set of generic components that can be used to create the system model to be simulated. Additionally, user-defined models can be easily integrated to model application-specific HW components. This chapter describes how the M3-SCoPE capabilities have been used in order to create an accurate model of the multimedia use case. First, the modeling of all the system components is described. Finally, the integration of all the components to create the final system simulation is shown.

3.1 Modeling of SW components

SCoPE infrastructure is based on a native co-simulation technique. The system is simulated combining native execution of annotated SW code with loosely timed TLM models of the HW components. As a consequence, the main difference with respect to other TLM modeling technologies is that there is no processor model, as an Instruction Set Simulator (ISS), or just-in-time binary translation, like in Qemu.

SW modeling and its interaction with the HW platform model is based on source-code annotation and the use of a RTOS model (Figure 4). The annotation extends the original functional SW code with information about the time, power consumption and cache misses required to execute each basic block of SW in the target platform. During execution, the kernel of the RTOS model controls the overall simulation, interleaving the execution of the application SW in order to adequately model the time advance and the bus traffic. Additionally, the RTOS model provides all the functionality required for SW/SW and SW/HW communication, automatically performing the required access to the HW platform model.

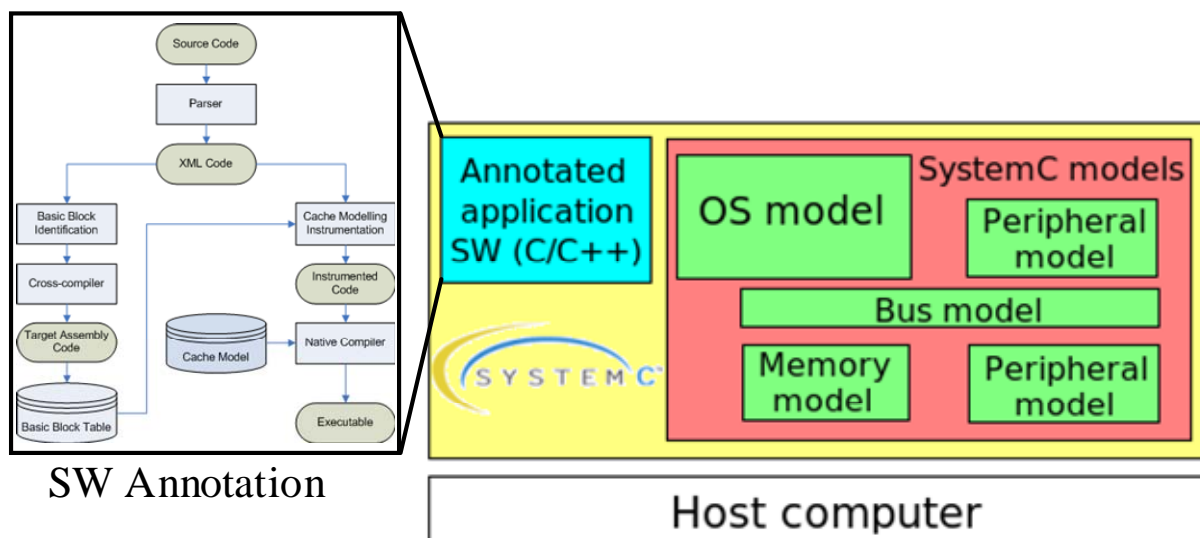


Figure 4. Basic structure of native co-simulation



In the Multimedia use case, there is not a real operating system, as there is only one thread per processor. The SW infrastructure is limited to some functions for data transfer among tasks and between remote L2 memory and scratch-pad L1 memories. Thus, to make compatible the original SW code and M3-SCoPE, it is required a layer to transform the original interface (MPS) to POSIX. As a consequence, a MPS2POSIX library has been developed by IMEC.

Nevertheless, an RTOS model is indispensable for SW modeling. It has been supposed that the eight processor nodes make up a symmetric multi-processor system (SMP). Thus, a single OS is instantiated, controlling all the processors. This solution enables the use of inter-processor communication (IPC) mechanisms for SW/SW communication. Additionally, it enables automatic exploration of fixed task allocation and dynamic task re-allocation by the OS. Nevertheless, these options have not been considered in the use case definition, as they are not supported by the other two simulators, and subsequently results of these options are not included in the present document.

About performance modeling of the SW code, most of the annotations added by the parser before compilation have been disabled. The parser transforms the SW code in three ways:

- Transform the system calls in order to call the OS model functions instead of the host OS. To do so, the “uc_” prefix is added to all these functions.
- Add SW execution times.
- Add cache miss calls.

As presented in chapter 2, the ADRES processor included in the platform has no caches. A scratch-pad and a small memory are used to store data and instructions respectively. Thus, no cache miss annotation is required.

Additionally, the automatic SW time estimation system of SCoPE has been disabled. Instead, the ARP library from IMEC provides the estimated time for each code segment. A pair of SW functions has been developed to interconnect M3-SCoPE and the ARP library.

Summarizing, only the first transformation of the parser is required. To indicate that, the compilation instruction included in the “makefile” files uses the “none” estimation method; which combines the flags “--scope-notime”, “--scope-noicache” and “--scope-nocache”.

It is important to note that, although the real system has no caches and no miss annotation has been performed, in the platform model all processors have caches associated, to model the power consumption of the local memories (static power and processor accesses).

Finally, in order to enable exploring the 6 different codes for the different multimedia parallelizations, dynamic libraries have been used. The code for each parallelization is encapsulated in a different “.so” file, opening the adequate one depending on the configuration selected by the DSE tool (M3explorer) for the experiment.

3.2 *Modeling HW platform components*

To model the HW platform described in the use case, models of the Communication Assist, the network, the memory and the busses has been used. To do so, generic models provided by SCoPE and new models created ad-hoc has been applied. In all the component models, the main goal has been to obtain the faster models possible while maintaining the estimation accuracy.



3.2.1 Model of the communication assist

In the real platform, this component has two purposes: acts as a DMA and as a bridge, connecting the local nodes to the network. SCoPE provides a generic DMA model, but a new model for the Communication Assist (CA) has been developed. The reason is the following. First, the generic DMA is prepared to be connected to a single bus, not to two different communication media. Thus, a DMA model and a bridge model are required for the platform modeling, slowing down the simulation time. Additionally, the DMA is a complex component, but the use case proposed does not use all its communication capabilities. DMAs show all their power in multithreaded systems, where some tasks runs while the other tasks wait for the DMA transfers. However, in the use case, there is no concurrency inside the processors, so the use of a complex DMA model implies an unnecessary simulation overhead.

As a consequence, a new component has been developed that acts as a bridge between the local bus and the network, providing the minimal DMA functionalities required for correct use case modeling. This bridge acts as a master of the network and slave of the bus in the processor nodes, and in the memory node in the opposite configuration (slave of the network and master of the bus).

Additionally, power consumption values for static power and data transfers have been applied.

3.2.2 Model of the communication network

The network proposed in the use case is a switch that connects all the processor nodes to the memory node. As a consequence, the modeling of a complete, complex switch device is avoidable. The model is not a “many-to-many” communication device, but a “many-to-one” device. That is, their operation can be easily modeled starting from a bus model, which in fact is a “many masters to one channel” device. Thus, a switch model has been developed inheriting the characteristics of the SCoPE generic bus model.

3.2.3 Bus and memory models

To interconnect the processor and the CA in the processor nodes and CA to memory in the memory nodes, instances of the SCoPE generic bus have been used. As the models connect one master and one slave, no complex, specific bus models are required to obtain an accurate simulated behavior.

With respect to the memory, the SCoPE generic memory model has been used, applying power values for static power and data accesses.

3.3 *System modeling and link to M3Explorer*

Based on the XML tool interface defined in D1.4.1, automatic exploration by using M3Explorer has been applied. A XML file with the system architecture described above is provided to M3-SCoPE to build the configurable model, fixing the parameters with the XML parameter files provided by the DSE tool (M3Explorer) for each experiment. Finally, the XML report file, with the information about execution time and power consumption is generated by the simulator and analyzed by the DSE tool in order to generate the Pareto Fronts.



Finally, to simplify the command required to execute the simulator, and enable running several experiments in parallel and easy changing the execution paths, a script that automatically set the adequate environment variables and calls the M3-SCoPE tool has been developed.



4 How to access the demonstrator

The public demonstrator can be downloaded from the M3-SCoPE web, hosted in the public web-site of the University of Cantabria. The M3-SCoPE web site is accessible at <http://www.teisa.unican.es/gim/en/scope/multicube.html>, following the corresponding link from the SCoPE web site (<http://www.teisa.unican.es/scope>) or from the MULTICUBE web site (<http://www.multicube.eu>). Once in the server you can access the tab “Demonstrator” and follow the instructions.

To simplify the process, a simple “bash” script can be downloaded. This script automatically downloads and installs all the required components, executes the exploration flow and generates the result files.

4.1 System Requirements

The open-source exploration flow has been developed for a Linux, 32 bits system, with

- gcc/g++ version 4.x.
- bison and flex
- libxml2 library and libxml2-devel
- Gnuplot software

4.2 Downloading and installation

To install the demonstrator it is required to perform the following steps:

1.- SystemC

- Download and decompress SystemC version 2.2 from the M3SCoPE web-site or from www.systemc.org
- Execute the sequence “configure”, “make all” and “make install” as described in the README file of the SystemC distribution.
- Set the environment variable “SYSTEMC” with the SystemC installation path

2.- SCoPE

- Download and decompress SCoPE version 1.1.5 from the SCoPE web-site
- Set the environment variable “SCOPE_HOME” with the SCoPE installation path
- Execute “make all” in the main SCoPE folder.
- Add to the PATH variable “SCOPE_HOME/bin” folder

3.- M3Plugin for SCoPE

- Download and decompress M3Plugin version 1.0.5 from the M3SCoPE web-site
- Set the environment variable “SCOPE_XML_PLUGIN” with the M3Plugin installation path
- Execute “make all” in the main M3Plugin folder.



4.- M3explorer

- Download and decompress M3explorer from the web page of the Polytechnic of Milan. http://home.dei.polimi.it/zaccaria/multicube_explorer_v1/Home.html
- Execute the sequence “configure”, “make all” and “make install” as described in the user guide of the M3explorer distribution.
- Add to the PATH variable the bin folder

5.- Demonstrator

- Download and decompress the use-case files from the M3-SCoPE web page.
- Execute “make”.

4.3 *Execution and exploration results*

The demonstrator associated with the integration of the Multimedia-M3SCoPE architecture and MULTICUBE Explorer consists of two parts.

- 1- Demonstration of the integration between the simulator and MULTICUBE Explorer
- 2- Demonstration of the HTML reporting capabilities of MULTICUBE Explorer

4.3.1 **Demonstration of the integration between the simulator and MULTICUBE Explorer**

For running the first part of the demonstrator, open a shell and follow the following steps:

```
> cd [use-case installation path]
> make clean
> make run_integration
```

This will run a full search over a restricted design space of the Multimedia-M3SCoPE architecture. Since a partial database is loaded at the beginning of the MULTICUBE script, this will trigger the simulator only in the case of two missing (one) architectures (2 and 8 cores).

4.3.2 **Demonstration of the HTML reporting capabilities of MULTICUBE Explorer**

For running the second part of the demonstrator, open a shell follow the following steps:

```
> cd [use-case installation path]
> make clean
> make run_report
```



The M3Explorer tool will create a 'report' directory that contains m3_index.html that is the document root for the HTML report on the design space.

4.3.3 Exploration results in the web page

The results of the exploration process can be shown directly from the example web page, without requiring from a visitor to perform the complete exploration. This can give a potential user a better idea of what can be obtained from the MULTICUBE open-source exploration flow with a minimal effort.

It can be directly accessed at : http://www.teisa.unican.es/gim/en/scope/report/m3_index.html

4.4 *License for the MPEG4 encoder IP*

To protect MPEG4 encoder code IP (Intellectual Property) provided by IMEC, the MPEG4 code is distributed in a binary format as dynamic libraries. This approach enables showing potential benefits of M3-SCoPE on a real-life use-case and at the same time protecting an IP. Please read the license file attached in Appendix I (also distributed with the public demonstrator) for further information on licensing issues concerning use of MPEG4 encoder IP in the MULTICUBE public demonstrator.



5 Exploration Results

The application of the design flow shown in Figure 1 results in an analysis of the different configurations of the system. In the multimedia use case, the effects in power consumption and latency resulting of modifying the frequency and selecting different parallelizations are analyzed. A final report in HTML format is obtained with all the results and it can be directly accessed at : http://www.teisa.unican.es/gim/en/scope/report/m3_index.html

This section summarizes the most significant exploration results with some discussion about them. First, the report shows a graphic with all the possible configurations and its effects in terms of latency and power.

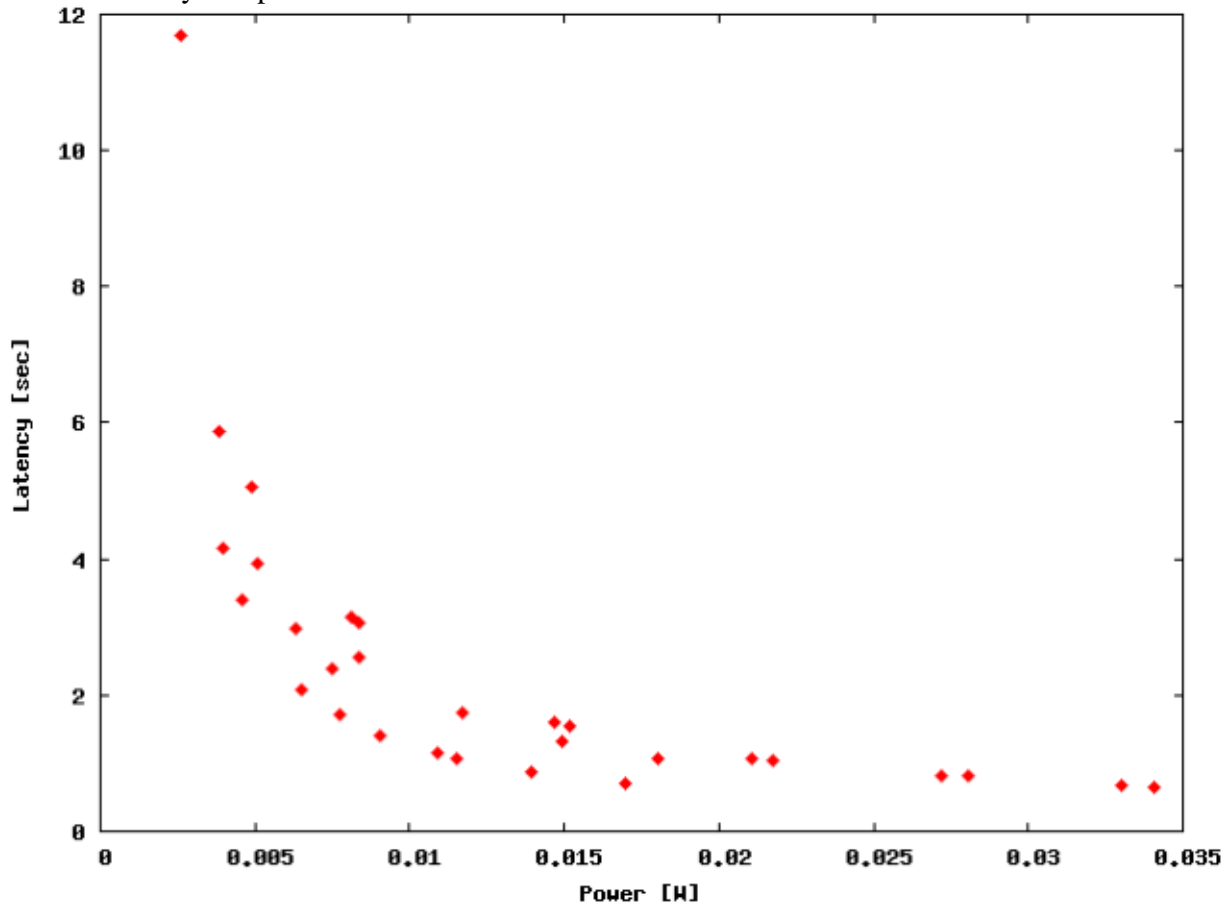


Figure 5. Analysis of the entire design space

Starting from the information obtained from a complete analysis of the design space, the best configurations can be identified. To do so, the Pareto front with the optimal configurations is obtained (Figure 6).



num_cpus	freq	Power[W]	Latency[sec]
2	40	0.002616	11.6941
2	80	0.003857	5.879
2	160	0.006293	2.97604
5	40	0.003964	4.15676
5	80	0.00653	2.09356
5	120	0.009052	1.40727
5	160	0.011532	1.06413
5	200	0.013959	0.859115
6	40	0.004581	3.38714
6	80	0.007756	1.70555
6	120	0.010878	1.14616
6	160	0.01395	0.866462
6	200	0.016959	0.699334
7	200	0.033026	0.67227
8	200	0.034084	0.653604

Figure 6. Composition of the Pareto set of the objective space

For this use case, a front analyzing the effects of the different parallelizations and another front analyzing the effect of using different frequencies for the execution of the Multimedia application have been extracted. These results are shown in Figure 7 and 8 respectively.

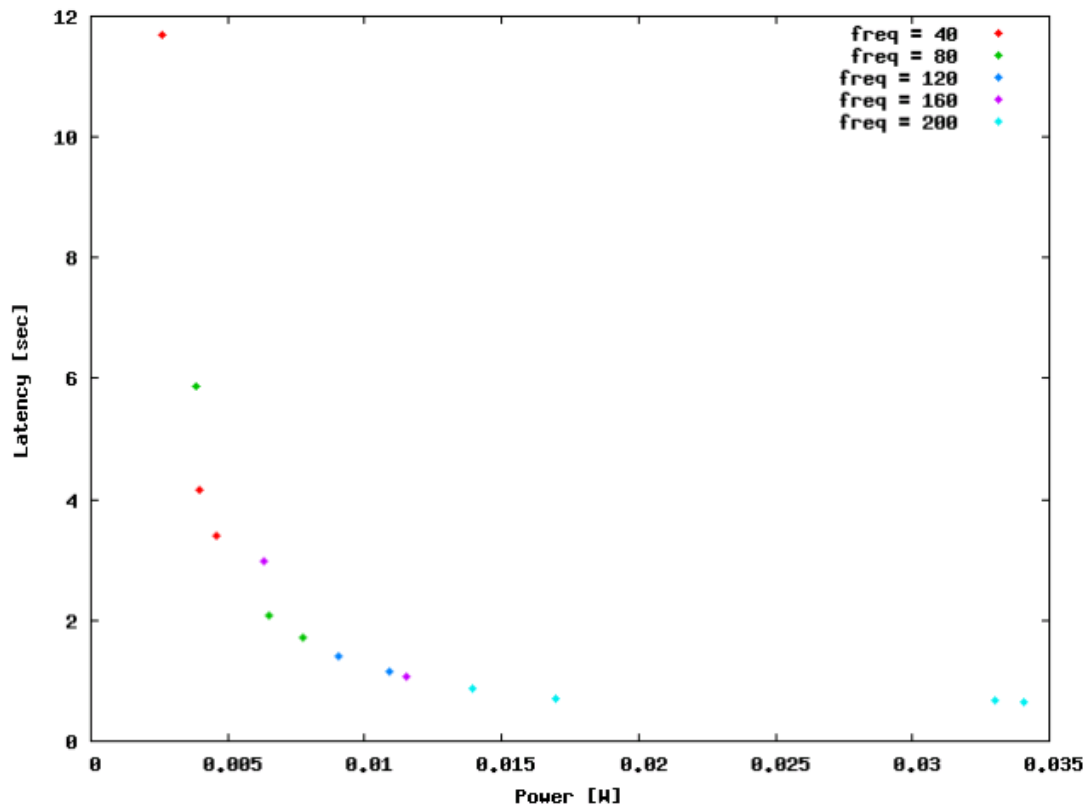


Figure 7. Pareto front resulting analyzing the effect of the different frequencies

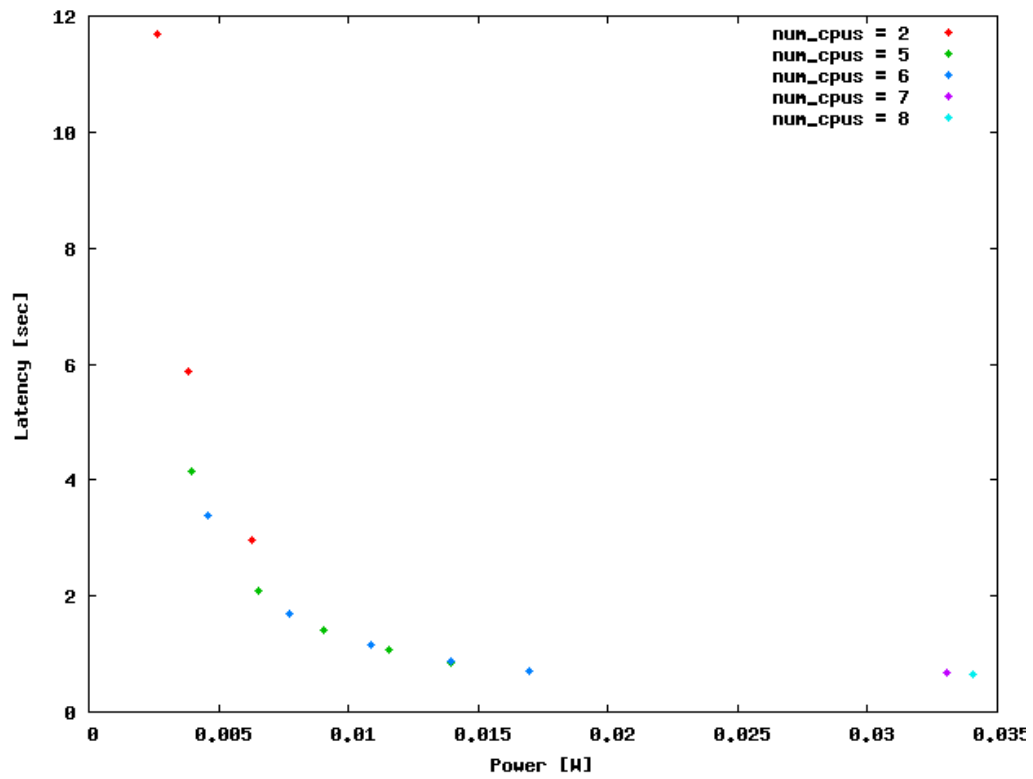


Figure 8. Pareto front resulting analyzing the effect of the different parallelizations



Additionally, the effects of both configuration parameters have been analyzed in Figure 9 and Figure 10. The interaction between parameters is measured as the difference in slope between the associated effects. Several graphics show the interaction effects for objective power consumption and latency. Parallel lines represent no interaction between parameters. Diverging lines represent synergistic effects while converging lines represent antagonistic effects between parameters.

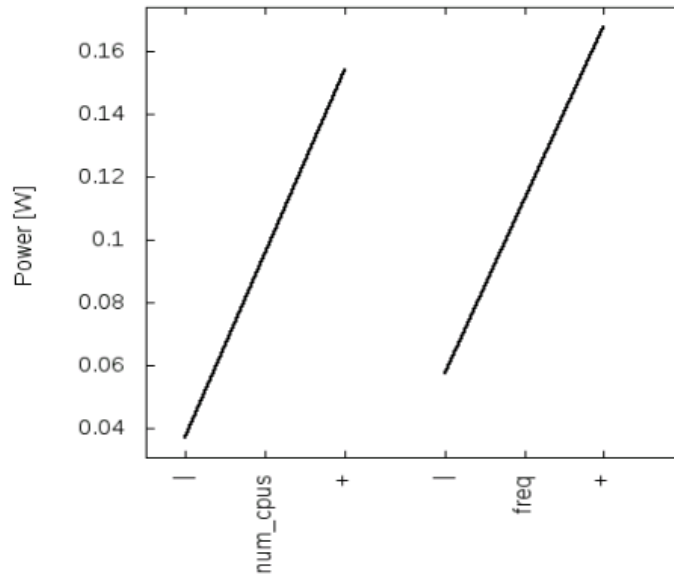


Figure 9. Effect of the number of parallelizations and frequency in power consumption

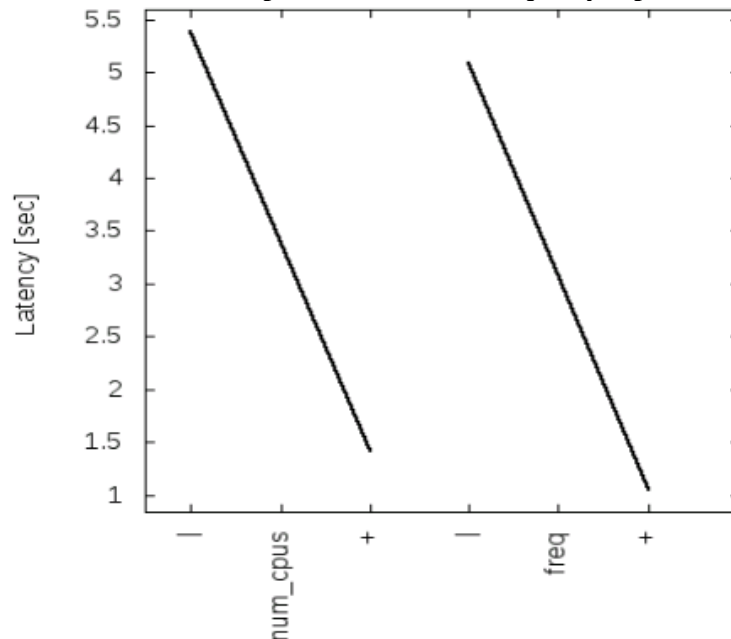


Figure 10. Effect of the number of parallelizations and frequency in system latency

As can be shown, when increasing the frequency and the parallelism, which implies increasing the number of active processors, the system computes faster, but the power consumption is increased.



Additionally, the results of M3Explorer can also be shown in other formats, to allow making complete analysis of the exploration results. Box plots (see Figure 11), surface plots (see Figure 12) and contour maps are provided by M3Explorer.

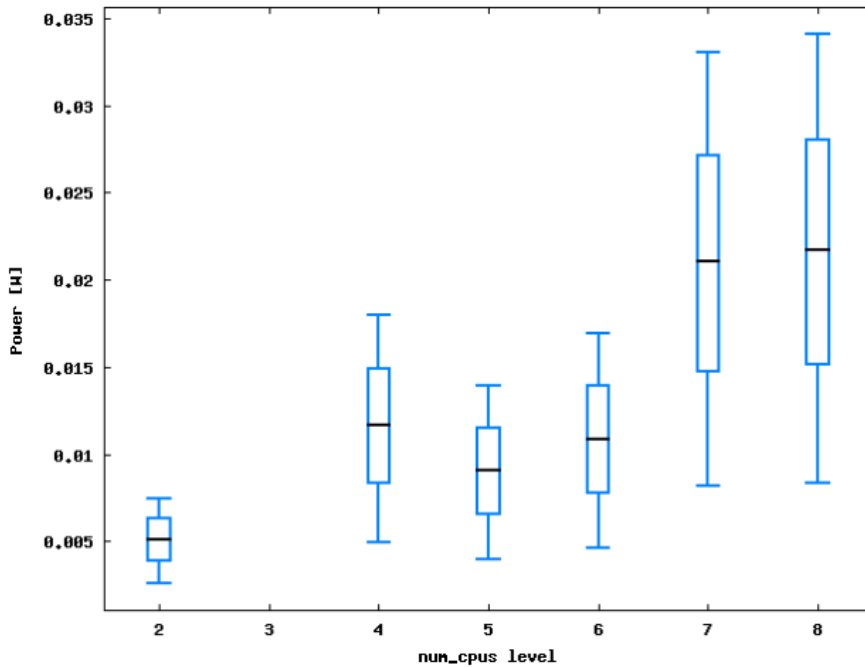


Figure 11. Effect of the number of parallelizations in power consumption

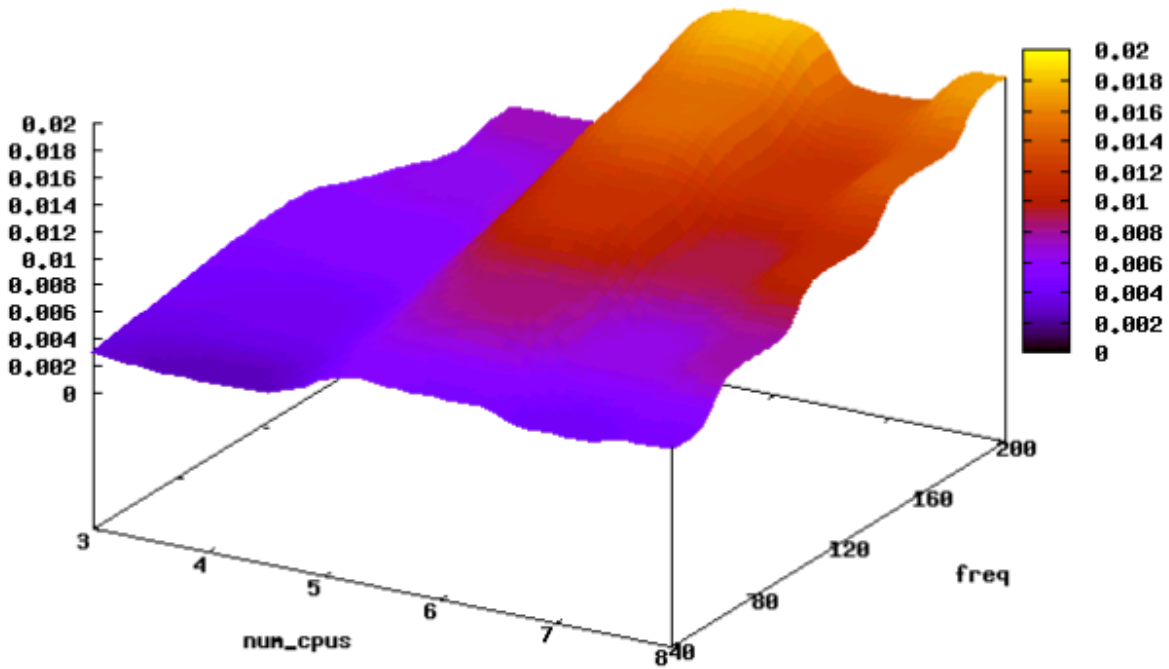


Figure 12. Surface plot for 'Power' considering 'num_cpus'-'freq' parameters



6 Conclusions

The Multimedia use-case represents a common industrial case to demonstrate three main results of the project:

Firstly, it has been used to demonstrate the open-source design flow developed in Task 3.1 combining M3Explorer, and M3SCoPE. Additionally, as it is based on open-source tools, this is a public demonstrator and enables easy dissemination. In fact, it has been prepared for public download from the web.

Secondly, the use case has been used to demonstrate the accuracy of the estimations provided by different simulators developed within the MULTICUBE project scope: HLSim, M3SCoPE and a CoWare TLM tools. As a consequence, the use case is more centered on modeling than in exploration issues. This means that the design space proposed is relatively small, which is required to obtain feasible exploration times.

Finally, as the use case has been defined to enable its application to different simulators, this use case enables the possibility of checking the validity of the XML interfaces for easy tool integration. The use case has allowed verifying that simulators used for the exploration can be changed without requiring any modification in the DSE tool, the XML files or the environment.

Nevertheless, the limitation in the use case complexity does not imply a lack in the verification of the quality of the open-source tools developed in the project. In fact, M3SCoPE has been deeply analyzed in the Power-line use case, where a less flexible, but much more complex system has been modeled and analyzed. In the same way, the M3Explorer exploration power has been proved in the Advanced Computing use case, where an extremely wide design space has been analyzed.



7 References

- [1]. MULTICUBE Deliverable D1.3: Definition of the specification for the industrial use-cases, June 2008.
- [2]. MULTICUBE Deliverable D1.4.1: Definition of the Specification of the Design Flow Integration, December 2008.
- [3]. MULTICUBE Deliverable D4.2.4: Final report on the application of MULTICUBE design flow to the demonstrators, June 2010.
- [4]. ISO/IEC 14496-2, “Coding of Audio-Visual Objects – Part 2: Visual”, 2001.
- [5]. I.E.G. Richardson, “H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia”, John Wiley and Sons, 2003.
- [6]. B. Mei, A. Lambrechts, J.-Y. Mignolet, D. Verkest, and R. Lauwereins, “Architecture Exploration for a Reconfigurable Architecture Template”, IEEE Design & Test of Computers, 22(2):90-101, 2005.



Appendix I: IMEC license for MPEG4 encoder IP

SOFTWARE LICENSE AGREEMENT

IMPORTANT-READ CAREFULLY: *This is an agreement between you (either an individual or a single entity) and the INTERUNIVERSITAIR MICRO-ELECTRONICA CENTRUM vzw, with its registered office at Kapeldreef 75, 3001 Leuven, Belgium, Register of Legal Entities Leuven VAT BE 0425.260.668, (hereinafter "Licensor"). Please read this Software License Agreement (further referred to as the 'Agreement') carefully before installing or using the proprietary computer software package identified in Appendix A attached hereto and further referred to as the "Program". For your perfect understanding note that the proprietary computer software package, any associated software, services, media, printed material, electronic documentation and related methods and techniques shall be considered as being part of the Program . By the installation and the use of the Program: (i) you are accepting and you are bound by the terms of this Agreement; (ii) you acknowledge to have read, understood and accepted the terms and conditions contained in this Agreement. This Agreement represents the entire agreement concerning the Program between you and Licensor, and it supersedes any prior proposal, representation or understanding between the parties.*

The Program is protected by applicable laws, including copyright laws and international copyright treaties, as well as other intellectual property right laws and treaties. The Program is distributed free of charge, upon acceptance of this Agreement .

Article 1 - Scope of the License

1.1. *The Licensor hereby grants to you, and you accept, for the duration of this Agreement, free of charge, a non-exclusive and non-transferable license on the Program, without the right to grant sub-licenses to install, display and use the Program for solely non-commercial demonstrational use of the Program within the scope of the **Multicube Project** (hereinafter referred to as the "Project");*

1.2. *You do not have the right to use the Program in any other way or for any other purpose than those set forth in article 1.1 of this Agreement, and shall not in any way dispose of or transfer the Program, in whole or in part to any other person or entity.*

1.3. *You may NOT (attempt to) (i) translate, decompile, alter, reverse-compile or disassemble all or any part of the Program or derive source code from the Program; (ii) distribute, rent, lease, sell, offer to sell, license or transfer the Program to any third party or use the Program for the performance of any commercial service; (iii) use the Program for any commercial purpose ; or (iv) benchmark the performance of the Program against any other existing software program.*

1.4. *Licensor may equip the Program with a key or mechanism to limit its use to a specific time or to otherwise restrict its use, and you agree not to circumvent such mechanism.*

Article 2 – Proprietary rights

2.1. *You acknowledge and agree that the Program is subject to Licensor's proprietary rights and constitutes confidential, proprietary and trade secret information of Licensor.*

2.2. *You further acknowledge and agree that Licensor is and shall remain the sole owner of all rights in and to the Program and all updates, upgrades, patches, improvements, modifications or derivative works thereof and you agrees to take whatever action is necessary to perfect Licensor's ownership thereof.*



2.3. *In relation to the Program, no rights are assigned or transferred to you. This Agreement does not convey to you an interest in or to the Program, but only a limited and revocable right to use the Program in accordance with the terms of this Agreement.*

Article 3 - Term and Termination

3.1 *This Agreement shall have effect as from date of installation or use of the Program, whichever is earlier.*

3.2. *You or Licensor may terminate this Agreement at any time by means of a written notice to the other party. Upon such termination, you agree to immediately stop using the Program and return all copies and portions thereof to Licensor.*

Article 4 – Liability

4.1. *THE PROGRAM (AND UPON LICENSORS SOLE DECISION ANY UPDATES, UPGRADES AND PATCHES THEREOF) ARE PROVIDED TO YOU ON AN “AS IS” BASIS WITHOUT ANY WARRANTY OF ANY KIND, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR ANY PARTICULAR PURPOSE OR THE ABSENCE OF INFRINGEMENT ON THIRD PARTY PATENTS OR OTHER PROPRIETARY RIGHTS.*

4.2. *LICENSOR DISCLAIMS ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, EXEMPLARY OR PUNITIVE DAMAGES OR LOST PROFITS OF ANY KIND WHATSOEVER, ARISING OUT OF THE USE OR THE INABILITY TO USE THE PROGRAM EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.*

4.3. *LICENSOR FURTHER DISCLAIMS LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, EXEMPLARY OR PUNITIVE DAMAGES OR LOST PROFITS OF ANY KIND WHATSOEVER, ARISING OUT OF THE IMPLEMENTATION OF THIS AGREEMENT, BREACH OF CONTRACT, TORTUOUS CONDUCT OR OTHERWISE EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.*

4.4. *You agree to indemnify and hold harmless Licensor from any claim by a third party arising out of or related to your use of the Program. You shall indemnify Licensor for any damages, losses and all costs and expenses resulting from or in any manner arising out of or in connection with the disclosure of the Program by you, your employees, consultants and/or students to persons not authorised under this Agreement.*

Article 5 – Invalidity

5.1. *If any provision, or part of any provision of this Agreement, or the Appendixes hereto, is invalidated by operation of law or otherwise, that provision or part will to that extent be deemed omitted and the remainder of this Agreement, or applicable Appendix, will remain in full force and effect. In place of any such invalid provision or part thereof, the Parties undertake to agree on a similar but valid provision the effect of which is as close as possible to the original intention of the parties.*

5.2. *The rights and obligations arising out of this Agreement may not be assigned or transferred by you to any third party, without Licensor’s prior written consent, and any attempt of assignment or transfer in contradiction with the provisions set forth herein shall be considered null and void.*

Article 6 - Governing law and venue

This Agreement shall be governed by Belgian law. All disputes between the Parties in connection with this Agreement shall first be discussed in good faith between the Parties, in order to try to find an amicable solution. However should a dispute relating to this Agreement arise which cannot be solved amicably, then the Parties hereby agree the courts of Brussels, Belgium, to be the only competent courts to take notice of this dispute.



Appendix A

Licensed Software

Description of the licensed Program “MPEG4 encoder” :

The licensed MPEG4 software consists of dynamically linking libraries executing six different versions of MPEG4 encoder software.

This MPEG4 encoder software is available on following Linux distributions: Ubuntu 9.x, RedHat Enterprise Linux 4.x and Fedora 13. Support for other platforms is not available. Further, the Licensor would not be providing additional updates on these platforms.

