



# MORPHEUS

---

## D2.3.2 Revised toolset modules report

### Publishable abstract

---

CONTRACT NO	MORPHEUS IST 027342
TYPE OF DOCUMENT	Publishable Abstract
DATE	19/11/2008
AUTHOR, COMPANY	Philippe Bonnot, TRT
CONTRIBUTORS	Loic Lagadec, Damien Picard, UBO (in section 6) Philippe Millet, Arnaud Grasset, TRT (in section 6) Florian Thoma, UK (in section 5) Marius Schoorel, ACE (in section 4) Koen Bertels, TUD (in section 4) Richard Taylor, CBlue (in section 6)
WORKPACKAGE	WP2
CONFIDENTIALITY LEVEL	PU
FILING CODE	MORPHEUS-D2.3.2-TRT-Toolset_modules_report-R03.doc

### Context

This deliverable is part of the MORPHEUS project which is a european initiative financed under the 6th FP and addresses innovative solutions for embedded computing based on dynamically reconfigurable platform and tools.

MORPHEUS project aims at satisfying embedded systems new demanding requirements in terms of computing performance, cost-efficient development, functional flexibility and sustainability by developing a global solution based on a modular heterogeneous SOC platform providing dynamically reconfigurable computing completed by a software oriented design flow and a consistent toolset.

MORPHEUS is a 3 and a half -year project started in 2006 and gathering all the required expertises from several countries: academics, industrials, SMEs.

### Aim of the deliverable

The D2.3.2 document provides a report on the toolset module description for phase 2 integrated toolset. It follows the revised toolset specification that provided a definition for these modules. The main objectives of these evolutions are the optimization of performances and the programming easiness. It therefore completes the D2.3.1 deliverable which presented the initial detailed description of these modules.

## Content of the deliverable

The deliverable quickly reminds the global toolset flow showing how the different modules interact: The global application program is coded in C with pragmas to identify the functions to be accelerated. This C code is provided to the Retargetable Compilation module that generates calls towards the services of the Dynamic Reconfiguration RTOS. The accelerated function design on the various units is performed thanks to the Spatial Design module.

Regarding Retargetable Compilation module, extensions compliant to OpenMP are described. Notably the algorithm proposed for the optimization of the static allocation based on area information is presented. Scenario groups are defined for all kernels. The objective is to minimize the total execution time under area constraints. An Integer Linear Programming approach is used to solve the problem. Then the Configuration Call Graphs construction is detailed with explanation of the different options made possible by the graph generator. The 2 steps are presented where first a Control Flow Graph is generated. The format of this intermediate level is described. The programmer has to write a configuration file for the identification of the Heterogeneous Reconfigurable Engine on which the accelerated functions can be implemented. The format of this file is given.

On the Dynamic Reconfiguration RTOS side, the document mentions the porting on the VHDL model since only the SystemC model had been addressed in phase 1. This notably required complementary drivers since the VHDL model is more complete than the SystemC one. It is reminded that the hardware Predictive Configuration Manager plays a role is the reconfiguration control. The new control functions related to Transfer Registers are introduced. More importantly, the dynamic allocation is presented together with improved scheduling strategies, including presentation of their advantages and disadvantages. A weighted score solution offers a tailored strategy tuneable according to application characteristics and requirements.

The Spatial Design module updates are presented for each of its 3 main components: the SPEAR tool where the accelerated function is captured and that generates the global graph of this function, then the CASCADE tool performing graph generation for the kernels that are involved in the accelerated function, and finally the MADEO tool that performs the code synthesis for the various targets.

On SPEAR tool, the main evolutions concern the implementation of a function on a chain of several Heterogeneous Reconfigurable Engines. The operation itself is captured as a graph of Elementary Functions (operative kernels) and this graph is segmented on the chosen HRE. The communication parameters corresponding to the stream on the HRE are generated including the management of the ping-pong buffering that permits the pipeline behaviour. This also includes the DMA synchronization aspect. A second evolution concerns the simplification of the function capture. For example, the specific XML code describing the kernels IO is here automatically generated from the C code. Another evolution is the testbench generation that can be used to verify the code generated for fine-grain target, which provides the user with a way to perform HDL simulations independently of the complete architecture of the chip.

Regarding CASCADE tool, in complement to the generation of kernel graphs required for the global function graph explained above, VHDL synthesis of these graphs is described. This VHDL synthesis work was planned as a backup solution for the MADEO synthesis solution. Since the MADEO solution feasibility is now acquired, the VHDL synthesis with CASCADE is simply used for comparison. First step was to generate a model including a coprocessor approach but this has been proven to be too expensive in term of area in regard of the size of the available target. Another approach based on intrinsic synthesis is therefore presented. It permits to generate a very compact datapath whose FIFO interface is compliant to the approach adopted in the Spatial Design approach.

For MADEO tool, the overall concept of Spatial Design is reminded: it is based on synthesis of the accelerated function including communications with the main internal data memory. Also the genericity of the approach is reminded: It is based on standard description language to describe and generate an API for the high abstraction level of the Control Data Flow Graphs. The synthesis on M2000 fine-grain target is described including notably the scheduling aspect of the computing datapath and the management of the interface Data Exchange Buffers and the handshake mechanism with the DMA. Also the behavioural simulator of the CDFG included can be used to perform early checks on the CDFG. The code synthesis for DREAM HRE is described, including

code generated for the control processor and code generated for the PICOGA array. Some detailed elements are presented such as the control of the loops, the generation of the connectors between kernels, specific cases with several dimensions as well as the data buffering within the elementary function.

The objectives of the toolset modules presented (performance increase, programmability increase) will have to be verified in the integrated version. It is already clear that they greatly contribute to simplify the complexity of the programming of such heterogeneous platform.