# SYNDEX EXECUTIVE KERNELS FOR FAST DEVELOPMENTS OF APPLICATIONS OVER HETEROGENEOUS ARCHITECTURES

*M. RAULET[1,2], C. MOY[3], F. URBAN[1], JF NEZAN[1], O. DEFORGES[1]*

(1) IETR/Image group Lab
UMR CNRS 6164/INSA
20, av des Buttes de Coësmes,
35043 RENNES, France
{jnezan, odeforge,furban}@insa-rennes.fr

(2) Mitsubishi Electric ITE
Telecommunication Lab
1 Allée de Beaulieu,
35 000 RENNES, France
raulet@tcl.ite.mee.com

(3) IETR/Automatic & Communication Lab
UMR CNRS 6164/Supelec-SCEE team
Avenue de la Boulaie - BP 81127
35511 Cesson-Sévigné, France
christophe.moy@rennes.supelec.fr

## ABSTRACT

Future generations of mobile phones, including advanced video and digital communication layers, represent a great challenge in terms of real-time embedded systems. Programmable multicomponent architectures can provide suitable target solutions combining flexibility and computation power. The aim of our work is to develop a fast and automatic prototyping methodology dedicated to signal processing application implementation onto parallel heterogeneous architectures, two major features required by future systems. This paper presents the whole methodology based on the SynDEx CAD tool, that directly generates a distributed implementation onto various platforms from a high-level application description, taking real-time aspects into account. It illustrates the methodology in the context of real-time distributed executives for applications based on video codecs and telecommunication physical layers.

## 1. INTRODUCTION

New embedded multimedia systems require more and more computation power. They are increasingly complex to design and have a shorter time to market. Computation limits of systems (i.e. video processing, telecommunication physical layer) are often overcome thanks to specific circuits. Nevertheless, this solution is not compatible with short time designs or the system's growing need for reprogramming and future capacity improvements. An alternative can be provided by programmable software (DSP, RISC, CISC processors) or programmable hardware (FPGA) components since they are more flexible. The parallel aspect of multicomponent architectures and possibly its heterogeneity (different component types) raise new problems in terms of application distribution: handmade data transfers and synchronizations quickly become very complex and result in lost time and potential deadlocks. A suitable design process solution consists of using a rapid prototyping methodology. The aim is then to go from a high-level description of the application to its real-time implementation on a target architecture as automatically as possible. The aim is to avoid disruptions in the design process from a validated system at simulation level (monoprocessor) to its implementation on a heterogeneous multicomponent target.

The methodologies generally rely on a description model which must match the application behavior. These applications are a mixture of transformation and reactive operators [1]. For the first class of system (including signal, image and communication applications), DFG (Data Flow Graphs) have proven to be an efficient representation model. This paper presents a rapid prototyping methodology based on the SynDEx tool, suitable for transformation-oriented systems and heterogeneous multicomponent architectures.

SynDEx automatically generates synchronized distributed executives from both application and target architecture description models. These specific executives are expressed in an intermediate generic language. These executives have to be transformed to be compliant with the type of component and communication media so that they automatically become compilable codes. In this article, we will focus on this mechanism based on the concept of SynDEx kernels, and detail new developed kernels enabling automatic code generation on various multicomponent platforms.

The paper is organized as follows: section 2 introduces the SynDEx tool and the AAA methodology. The prototyping platforms and the executive kernels are described in section 3. Application implementations according to the AAA methodology are explained in section 4. Finally conclusions are given in section 5.

## 2. SYNDEX OVERVIEW

SynDEx[1] is a free academic system level CAD tool developed in INRIA Rocquencourt, France. It supports the AAA methodology (Adequation Algorithm Architecture [2]) for distributed real-time processing.

### 2.1 Adequation Algorithm Architecture (AAA)

A SynDEx application (Fig.1) comprises an algorithm graph (operations that the application has to execute) which specifies the potential parallelism, and an architecture graph (multicomponent target, i.e. processors and specific integrated circuits), which specifies the available parallelism. "Adequation" means efficient mapping, and consists of manually or automatically exploring the implementation solutions with optimization heuristics [2]. These heuristics aim to minimize the total execution time of the algorithm running on the multicomponent architecture, taking the execution time of operations and of data transfers between operations into account. An implementation consists of both performing a distribution (allocating parts of the algorithm on components) and scheduling (giving a total order for the operations distributed onto a component) the algorithm on the architecture. SynDEx provides a timing graph which includes simulation results of the distributed application and thus enables SynDEx to be used as a virtual prototyping tool.

[1] www.syndex.org

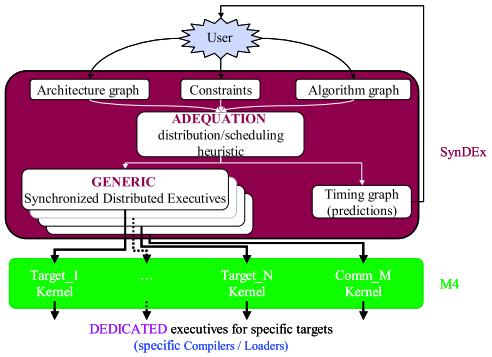


**Fig. 1**. SynDEx utilization global view

### 2.2 Automatic Executive Generation

The aim of SynDEx is to directly achieve an optimized implementation from a description of an algorithm and of an architecture. SynDEx automatically generates a *generic executive*, which is independent of the processor target, into several source files (Fig.1), one for each processor. These generic executives are static and are composed of a list of macrocalls. The macro processor transforms this list of macro-calls into compilable code for a specific processor target. It replaces macro-calls by their definition given in the corresponding *executive kernel*, which is dependent on a processor target and/or a communication medium. In this way, SynDEx can be seen as an off-line static operating system that is suitable for setting data-driven scheduling and so for signal processing applications.

### 2.3 Design Process

SynDEx is basically a CAD tool for distribution/scheduling and code generation. Thanks to our kernels presented in section 3, SynDEx can also be directly used as the front-end of the process for functional checking. The design process is based on a single tool and is therefore simple and efficient. SynDEx therefore enables full rapid prototyping from the application description (DFG) to final multiprocessor implementation.

Automatic code generation provides a standard C code for a single PC implementation (SynDEx PC kernel). In this way, the user can design and check each C function associated with each vertex of its DFG, and can check the functionalities of the complete application with any standard compilation tools. With automatic code generation, visualization primitives or binary error rate computation can be used for easy functional checking of algorithms. Next the user can easily check his own DFG on a cluster of PCs interconnected by TCP Buses. This cluster can emulate the topology of an embedded platform.

The developed DFG is then used for automatic prototyping on monoprocessor embedded targets where chronometric reports are automatically inserted by SynDEx code generator. Each duration associated with each function (i.e vertex) executed on each processor of the architecture graph is automatically estimated using dedicated temporal primitives.

These chronometric durations characterize the algorithm graph. Then SynDEx tool executes an adequation (optimized distribution/scheduling) and generates a real-time distributed and optimized executive according to the target platform. Several platform configurations can be simulated.

The main advantage of this prototyping process is its simplicity because most of the tasks performed by the user concern the description of an application. Only a limited knowledge of SynDEx and compilers is required. All complex tasks (adequation, synchronization, data transfers and chronometric reports) are executed automatically, thus reducing the "time to market". Exploring several design alternatives can rapidly be done by modifying the architecture graph.

## 3. SYNDEX EXECUTIVE KERNELS

As described above, the SynDEx generic executive will be translated into a compilable language. The translation of SynDEx macros into the target language is contained in library files (also called kernels). The final executive for a processor is static and composed of one computation sequence and one communication sequence for each medium connected to this processor.

### 3.1 Development Platforms

Different hardware providers (Sundance, Pentek) were chosen to validate automatic executive generation. Multicomponent platform manufacturers must insert additional digital resources between processors to make communication possible. Thus, SynDEx kernels depend on specific platforms. Many component and inter-component communication links are used in their platforms, ensuring accuracy and the generic aspect of the approach. The use of several hardware architectures guarantees generic kernels developments.

**Sundance[2] platform:** a typical Sundance device is made up of a host (PC) with one or more motherboards, each supporting one or more TIMs (Texas Instrument Module). A TIM is a basic building block from which you build your system. It contains one processing element which is not necessarily a DSP but an I/O device, or a FPGA. A TIM also provides mechanisms to transfer data from module to module. These mechanisms, such as SDBs (200MB/s), ComPorts (CP 20MB/s), or a global bus (to access a PCI bus up to 40MB/s), are implemented on the TIMs using FPGAs.

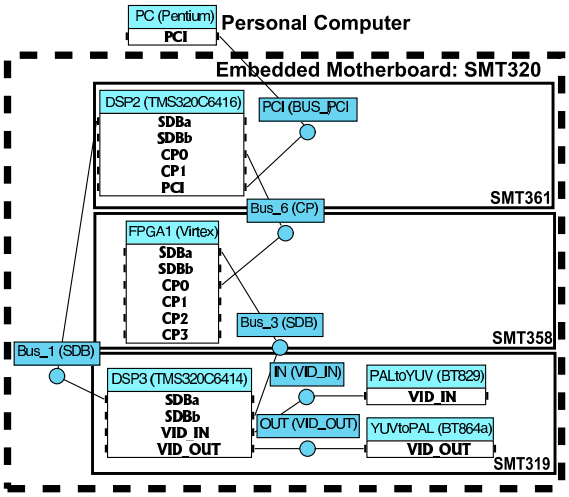


**Fig. 2**. Example of Sundance architecture topology

[2] **http://sundance.com/**

The sundance motherboards (e.g. SMT310q or SMT320)(Fig.2) are modular, flexible and scalable. Up to four different modules can be plugged into the motherboard and connected using CP or SDB cables. The SMT361 TIM module with a C6416 (400Mhz) is very suitable for image processing solutions as the C64xx has special functions for handling graphics. The SMT319 is a framegrabber, which includes a C6416 (600Mhz) and two non programmable devices: a BT829 PAL to YUV encoder and a BT864 YUV to PAL decoder. These two devices are connected to the C6414 DSP thanks to two FIFOs, which are equivalent to SDBs with the same data rate. An SMT358 is composed of a programmable Virtex FPGA (XCV600) which integrates specific communication links and specific IP blocks (computation).

**Pentek**[3] **platform:** The Pentek p4292 platform is made up of four C6203 DSPs. Each DSP has three communication links: two bi-directional (300Mhz) inter-DSP links and one for the I/O interface. The four DSPs are already connected to each other in a ring structure. Some daughter boards may be added to the p4292 thanks to the VIM bus, such as analog to digital converters (ADC p6216), digital to analog converters (DAC p6229), or FPGAs (XC2V3000).

This stand-alone Pentek platform is connected to an Ethernet network. This allows TCP/IP (effective 1.5MB/s) communications between DSPs and any computer in the network. However this bus throughputs will not authorize for instance the transfer of uncompressed images.

### 3.2 Software component kernels

*Most of the kernels are developed in C language* so that they can be reused for any C software programmable device (e.g. Texas Instrument C64x DSP). These kernels are quite similar for the host (PC) and the embedded processors (DSPs). The generated executive is composed of a sequential list of function calls (one for each DFG operation). This kind of executive and the fact that the adapted C compiler for DSPs has really improved in term of resource use mean that the gap between an executive written in C and an executive written in assembly language is narrow.

SynDEx creates a macrocode made of several interleaved schedulers allowing parallelism: one for computation and the others for communications. In [3] transfers were executed by DMA. Here an improvment of TI C64x DSP is multi-channel eDMA which maximize parallelism and have better timing performance, however eDMA mechanisms have many differences with DMA in term of programmation and synchronizations.

The development of an application on TI processors can be hand-coded with TI RTOS called DSP/BIOS. DSP/BIOS is well-suited for multithread monoprocessor applications. Several processors must be connected to improve computational performances and reach real-time performances. In this case, the multithread multiprocessor 3L diamond[4] RTOS is more appropriate for this situation than DSP/BIOS. Applications are built as a collection of inter-communicating tasks. The mapping and scheduling of each task are chosen manually. Then data transfers and synchronizations are implemented by the RTOS using precompiled libraries and data polling techniques.

[3] **http://www.pentek.com/**

[4] http://www.shen.myby.co.uk/threel/

Data transfers in a signal processing application are generally well-defined both in terms of type and number so that their description with a DFG is suitable. The execution of DFG operations is also well-defined so that data transfers can be implemented with static processes. As static processes are faster than dynamic ones, *SynDEx kernels are developed without any RTOS.* That is to say that the SynDEx generic executive is not transformed into dynamic RTOS functions but into specific static optimized functions.

### 3.3 Communication link kernels

With AAA methodology, two different models [4] are possible for communication links between processors: the SAM (Single Access Memory) and RAM (Random Access Memory, shared memory) models.

**The SAM model** corresponds to FIFOs in which data are pushed by the producer if it is not full, and then pulled by the receiver if the FIFO is not empty. Synchronizations between the two processors are hardware signals (empty and full flags) and are not handled by SynDEx semaphores. The data must be received in the same order as it is sent. Most of our kernels are designed according to this model. SDBs, CPs and BIFO_DMAs enable parallelism between calculations and communications, whereas TCP and BIFO do not enable it (data polling mechanism).

**The RAM model** corresponds to an indexed shared memory. A memory space is allocated and an interprocessor synchronization semaphore is created for each item of data that has to be transferred. This mechanism allows the destination processor to read data in a different order to which it has been written by the source processor. Inter-processor synchronizations are handled by SynDEx.

PCI transfer kernels for communications between a DSP on Sundance platforms and the host computer have been developed with these two models. The PCI bus is the first real implementation of the SynDEx RAM model giving higher data rates. Theorical data rates of PCI bus are up to 132MB/sec. Transfer data rates are shown in Table 1 with and without model synchronizations.

| | *SMT320* | *SMT310Q* |
|---|---|---|
| SAM Model | 16 | 30 |
| RAM Model | 40 | 70 |
| without Synchronizations | 50 | 100 |

**Table 1**. PCI bus Transfer rates (MBytes/sec)

### 3.4 Hardware component kernel

Moreover a FPGA kernel for programmable hardware components, that could be considered as a coprocessor in order to speed up a specific function of the algorithm, has been developed in HDL. This kernel handles automatic integration of inter-component communication syntheses and instantiates a specific IP block.

Programming of a communication link depends on its type but also on the processor. Previous work has already validated these libraries however they need to evolve with processors or communication links (depending on provider's additional logic).

### 3.5 Kernel organization

The libraries are classified to make developments easier and to limit modifications when necessary. As shown in Figure 3, these files are organized in a hierarchical way. An application-dependent library contains macros for the application, such as the calls of the algorithm's different functions. A generic library contains macros used regardless of the architecture target (basic macros). The others are architecture-dependent: processor or communication type dependent. Processor-dependent libraries contain macros related to the real-time kernel, such as memory allocations, interrupt handling or the calculation sequence. Communication type-dependent libraries contain macros related to communications: send, receive and synchronization macros, communication sequences. As different processor types (with different programming of the link) can be connected by the same communication type, one part per processor type can be found in one library. The right part of the file is used during the macro-processing.

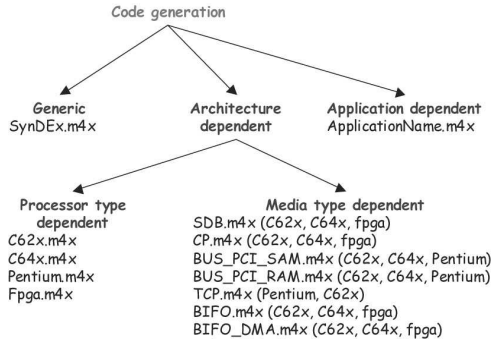


**Fig. 3**. SynDEx kernel organization

Kernels have been developed for every component of the platforms described in section 3.1. When SynDEx is used for a new application, only the application-dependent library needs to be modified by the user. Architecture-dependent libraries are added or modified when a new architecture is used (a processor or a medium that does not have its kernel).

## 4. RESULTS

We have designed and implemented a real-time multi-layer application composed of video codecs (LAR [5], Mpeg-4 [6]) and a digital communication layer (UMTS, MC-CDMA [7]). A coding application (Mpeg-4 or LAR) feeds a video coded bitstream in the transmetter (UMTS or MC-DMA) whereas the associated video decoder is connected to the receiver (UMTS or MC-CDMA). The result is a complete demonstration application with automatic code generation over several kinds of processors and communication media, or several kind of platforms. The multi-layer application has been first implemented on Pentek platform with several C62x DSPs and FPGAs. Then the prototyping of the same application has been immediately done on Sundance platform with several C64x DSPs.

## 5. CONCLUSIONS

The design process proposed in this paper covers every steps of digital signal application development, from simulation to integration. Compared with a manual approach, the use of our fast prototyping process ensures easy reuse, reduced time to market, design security, flexibility, virtual prototyping, efficiency and portability.

The kernels enable recent multiprocessor platforms to be used and also enable the process to be extended to heterogeneous platforms. It was tested on several different architectures composed of Texas Instruments TMS320C6201, TMS320C6203, TMS320C6416 DSPs, Xilinx Virtex-E and Virtex-II FPGAs, and PCs.

The calculations and data transfers are executed in parallel. RAM and SAM communication models have been tested for PCI transfers. Higher transfer rates are reached using the RAM model enabling real-time video transfers between a PC and a DSP platform.

Several complex tasks are performed automatically, such as distribution/scheduling, code generation of data transfers and synchronizations. So the development of a new application is limited to the algorithm description and to the adaptation of kernels for platforms or components. Furthermore, as the C language is used and there is a large number of tested topologies, developed DSP kernels can easily be adapted to any other DSP and communication media.

## REFERENCES

[1] T.A. Henzinger, and M.A. Sanvido C.M. Kirsch, and W. Pree, "From control models to real-time code using Giotto," *IEEE Control Systems Magazine*, vol. 23, no. 1, pp. 50–64, 2003.

[2] T. Grandpierre, C. Lavarenne, and Y. Sorel, "Optimized Rapid Prototyping for Real-Time Embedded Heterogeneous Multiprocessors," in *CODES'99*, Rome, Italy, May 1999, pp. 74–78.

[3] Y. Le Méner, M. Raulet, J.-F. Nezan, A. Kountouris, and C. Moy, "SynDEx Executive Kernel Development for DSP TI C6x Applied to Real-Time and Embedded Multiprocessors Architectures," in *XI European Signal Processing Conference (EUSIPCO)*, Toulouse, France, September 3-6 2002.

[4] T. Grandpierre and Y. Sorel, "From Algorithm and Architecture Specification to Automatic Generation of Distributed Real-Time Executives: a Seamless Flow of Graphs Transformations," in *First ACM and IEEE International Conference on Formal Methods and Models for Codesign, MEMOCODE'03*, Mont Saint-Michel, France, June 2003.

[5] M. Raulet, M. Babel, J.-F. Nezan, O. Déforges, and Y. Sorel, "Automatic Coarse Grain Partitioning and Automatic Code Generation for Heterogeneous Architectures," in *SIPS*, Seoul, Korea, August 27-29 2003.

[6] N. Ventroux, J.-F. Nezan, M. Raulet, and O. Déforges, "Rapid Prototyping for an Optimized Mpeg-4 Decoder Implementation over a Parallel Heterogeneous Architecture," in *ICASSP*, Hong-Kong, April 06-10 2003, Conference cancelled - Invited paper, ICME 2003.

[7] S. Le Nours, F. Nouvel, and J.F. Helard, "Example of a Co-Design approach for a MC-CDMA transmission system implementation," *Journees Francophones Adequation Algorithme Architecture (JFAAA)*, December 2002.