# IP Based Design 2002

**Session 9B: Interconnect and communication**

## Design of communication interface based on configuration for system on chip

*Issam MAALEJ[1-2*], Guy GOGNIAT[2], Mohamed ABID[1], Jean Luc PHILIPPE[2]*

[1] ECOLE NATIONALE D'INGENIEURS DE SFAX
DEPARTEMENT DE GENIE ELECTRIQUE, *B.P : W 3038* SFAX TUNISIE

[2]L.E.S.T.E.R., UNIVERSITE DE BRETAGNE SUD, FRANCE ; RUE SAINT MAUDE - 56100 LORIENT
* Email *: maalej@ iuplo.univ-ubs.fr*

## Abstract:

Communication synthesis is an essential step in hardware/software co-synthesis: many embedded systems use automatic generation of interfaces for point to point communication or use external supports of communication as standard bus or micro network. In this paper, we consider the problem of hardware – software interface design in a codesign approach for real time processor embedded system. We refer to the hardware component as hardware accelerator and the software component as processor. In this paper we only target what we have called FGDA (Fine Granularity Determinist Accelerator) accelerators. These types of accelerators are directly integrated into the processor core in order to optimize the transfers of data between the two units. Furthermore, our method is based on a configuration approach to design a communication interface.

### 1. Introduction

Thanks to the micro-electronic advances in integration and the progress made by silicon manufacturers, implementation of embedded systems is observing a quick shifting from systems on boards to Systems on Chips (SoC) where all components are on the same die. Hardware/Software SoC co-design is one of the most important electronic application areas in future-oriented designs. As time to market pressures and product complexities climb, the pressure to reuse complex building (also known as Intellectual Property (IP) or Virtual Component (VC)) also increases. These components represent functions of specific domains like signal processing (DCT, FFT), telecommunication and multimedia (VLC, Turbo codes) etc. In this area, Intellectual Property management requires new concepts and innovative breakthroughs, in order to introduce a new quality in Electronic Design Automation.

Communication synthesis is a key step in the integration of the system. Interfacing between hardware and software is the bottleneck in many embedded systems, because communication links add both chip and board costs.

Today, different approaches attempt to ease hardware-software communication by automatic generation of interface for point-to-point communication or by defining architecture using external supports of communication as standard bus or micro network. In this paper, we propose a communication synthesis approach in order to ease the interface design for SoC. This approach considers a typical architecture of system on chip (SoC), which consists on the integration of hardware accelerator in the core of the processor. This architecture integrates hardware accelerators on the internal bus of the processor. The advantage of such architecture is to have a specific processor without designing an ASIP (Application Specific Instruction set Processor). The accelerators are directly integrated into the processor core in order to optimise the transfers of data between the two units. This approach is based on the module configuration to design a communication interface. On our work we consider two types of accelerators: Fine Granularity Determinist Accelerator (FGDA) and Variable Granularity Non-Determinist Accelerator (VGNDA). We focalise in this paper only the first type of accelerators.

The rest of the paper is organised as follows. Section 2 gives a brief description of previous and related work; section 3 describes the target architecture and formulates the problem that we consider, while section 4 presents our approach to communication synthesis; some experiments and results are reported in section 5 and finally section 6 concludes the paper and presents some line of future work.

## 2. Related work

The problem of interface synthesis has been addressed in the literature. The majority of published work can be classified in two categories. The first treats the problem during high level synthesis thanks to tools or languages of specification like Cosmos [Ben95], Polis [Bal97], SystemC [Kje01] and CoWare [Ver98], which allow the automatic generation of interfaces of communications. For example, in [Zit02] the communication synthesis approach is basically based on the synthesis of the communication structures (interfaces, arbitration schemes). The second is interested by standard components that have incompatible protocols. For example, the IMEC proposes a solution for the design of communication interfaces [Lin97]. This solution is interesting when all IPs used have the same communication protocol. [Lys02] uses "wrappers" in order to ensure the communication between the hardware accelerators and the bus. [Son02] proposes an architecture micro-network in SoC in order to ensure the communication between different software and hardware components.

These approaches require having a perfect knowledge of the protocols of both sender and receiver. Also, they represent a dedicated approach: for each new interface it is entirely necessary to specify this interface. This is why we propose an approach based on configuration of predetermined interfaces that makes the designers a way to obtain very quickly and easily an optimised interface in time and area.

## 3. Target architecture

Architecture of embedded system includes generally many types of components as processor core, hardware and software components. For these systems, designer can target three types of processors according to constraints of the system: processor specific to an application as ASIP, processor specific to a domain as DSP or VSP (Digital/Video Signal Processor) and processor of a general nature as RISC processor (Reduced Instruction Set Computer). The First and the second types present better performances and flexibility but they require a significant cost of design. Contrary, the last type presents the most short design time and presents an acceptable performance. Always, different applications require adding some hardware components to processor core in order to make a specific system in the application domain.

Target architecture considered in this work is depicted in fig.1. Such architecture includes hardware accelerators that represent computing units that haven't enough control logic to be autonomous. These accelerators need others control units that command the communication transfers. Thus, they are connected to the internal bus of the processor and may be considered as slave while processor core may be master of the system.

Also, the accelerators that we consider in this work are characterized by a constant computing time, no memorization on the level of the interface (only transfers of data), a fine granularity and a constant flow and size of input and output data. In the following, we note such accelerator FGDA (Fine Granularity Determinist Accelerator). Thus, their characteristics are perfectly deterministic and independent of the input data. In this case, the proposed communication synthesis approach is based on configuration of predetermined interfaces in order to allow the designers to obtain quickly and easily an optimised interface in term of time and area. This module is presented in the next section.

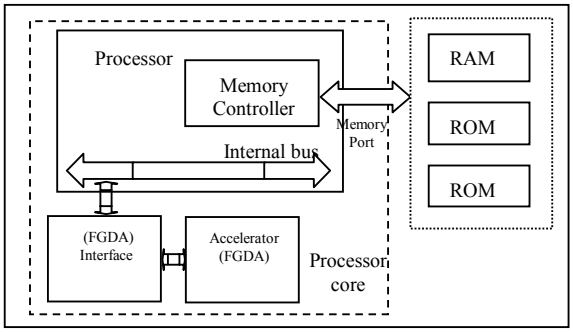


Figure 1 : SoC architecture

## 4. Communication synthesis approach based on configuration

Designers committee agrees that the key approach to enhance the reuse and integration of IPs into a wide range of applications is the one separating communication from behaviour. This model called *interface-based design* has two main advantages. The first is that VC creators can easily update their products to customer's architecture by limiting the modifications to IP interfaces, while keeping the IP cores untouched. The second advantage is to promote the definition and the development of standards and design methodologies for IP

interfaces. Today, these concerns are being addressed by many industry as SoCiations like Virtual SoCket Initiative Alliance (VSIA) and Virtual Component Exchange (VCX). Our approach considers this model and this section presents successively the FGDA interface description and its configuration.

### 4.1 Interface description

The objective of FGDA interface module is to tail the communication interface between processor core and FGDA. Accelerators protocols are of start-busy type while bus protocols vary from one to another. Each bus protocol considers a particular delay and form of transfer. The use of protocol or another is according to the application. Thus, in order to design an interface which is able to adapt to any kind of protocols, we consider a description of interface module that makes communication with a bus independently of the protocol used. This module will be configured depending on the bus and the accelerated used.

The interface operation is described as follows. During the first clock cycle, the address is presented on the bus. The interface decodes if the address is intended for the accelerator or not. In this case, it decodes the control signal in order to prepare the rest of the transfer (decode the type of transfer: Read or Write). During the second clock the interface reads (writes) the data. If the sequence is correct, it transmits the data towards the accelerator and activates the signal start so that it can begin treatment.

This approach based on the configuration of this predetermined interface module allows designing communication interface between processor core and accelerator without need to specify a dedicated interface.

### 4.2 Interface spécification

The interface is described in VHDL language and is performed in two files. The first is a package that describes the types necessary for the interface. It contains one type describing the size of the buses of a given processor, another describing the size of the data bus of the hardware accelerator, and two others describing the signals entered and exits of the bus of the processor. The second file constitutes the functional description of the interface. The interface is ensured by 5 processes grouped in one "process" which is sensitive to the clock signal "clk". Each process is controlled by specific conditions to its operation. The first process constitutes the procedure of launching of the transfer of the data. It detects the beginning of the transfer operation when all the operating conditions are validated. This process has as a function to validate the transfer operations in writing and reading. These two operations constitute successively the second and the third process. The operating conditions of these processes are ordered by the first process, the fourth and the fifth process which correspond successively to the processes of reset of writing and reading.

### 4.3 Interface configuration

The considered interface represents a generic module and allows to be connected with various types of buses and different accelerators. Indeed, this interface is able to ensure the transfer of data between processor bus and accelerators bus having different sizes. This property is performed in the code-description using notion of package in the VHDL language. Using this approach, we don't have to use an automatic generation of interface but the configuration of communication interface. This approach has the advantage of making possible the communication without detailed knowledge of protocols and sequencing of the tasks.

With proposed approach, communication synthesis is summarised in the three following steps:

1. Configure the data types of the buses in the package.
2. Configure the signals of the buses in the package.
3. Update the code description.

## 5. Experimental results

The interface module is described using VHDL language. This code is optimised in term of area and frequency. Note that the processor and the hardware accelerators can work with different frequencies. Different examples were done, in particular communication between DCT (Discrete Cosinus Transform) as hardware accelerator and LEON as processor core. Some results are shown in table 1.

The DCT has 64 bits as data size and start and busy as control signals. This accelerator is considered as a FGDA. LEON processor has a data bus of 32 bits. The used FPGA have a maximum frequency as 100Mhz. After synthesis, as shown in table 1 the interface designed uses a minimum area and the frequency of the accelerators is on maximum.

Similar results are obtained when we use the interface to ensure communication between the APB and PCI buses (also exist on LEON processor) and the same DCT. These buses have also 32 bits as data bus size but have different protocol

complexity. This diversity can confirm the reliability of the interface module designed.

| | Area map ( % of total area) | High frequency (Mhz) | FPGA |
|---|---|---|---|
| Interface | 1 | 100 | Virtex XCV800 HQ 240 |
| LEON processor | 28 | 35 | Virtex XCV800 HQ 240 |
| DCT | 6 | 100 | Virtex XCV800 HQ 240 |

Table 1: Experimental results

**6. Conclusion and future work**

This article proposes a configurable generic interface model allowing the refinement of the communication between processor core and various FGDA hardware accelerators. This approach allows the designer to simplify the communication integration step. This model is sufficiently generic in order to be able to adapt to various bus protocols and accelerators data size. The experiment results show that this method is efficient for the hardware accelerators of FGDA type.

Since this approach is limited to the accelerators of FGDA type, this interface is currently extended to be able to ensure communication between processor and more general accelerator. We called this accelerator VGNDA (Variable Granularity Non Determinist Accelerator). These works are on done.

**References**


[Ben94] T. Ben Ismail, M. Abid, and A.A. Jerraya, “COSMOS: A CoDesign Approach for Communicating Systems”, Proc. Third IEEE Int'l Wshp on Hardware/Software Codesign (CODES/CASHE), Grenoble, France, IEEE CS Press, pp. 17-24, Sept. 1994.

[Bal97] Balarin F., Chiodo M., Lavagno L., Passerone C., Sangiovanni-Vincentelli A., Sentovich E., Suziki K., Tabbra B., Hardware-Software Co-Design of embedded system : The Polis Approach, Kluwer 1997.

[Gon97] Gongniat G., Auguin M., Belleudy C., "Interface Synthesis in Embedded Hw/Sw Systems", Carlo Delgado KLOOS Hardware description langages and their applications, P 80-82, Aduard Cerny (editor), Chapman & hall, 1997

[Kje01] Kjetil Svarstad, Gabriela Nicolescu, Ahmed A. Jerraya "A model for describing communication between aggregate objects in the specification and design of embedded systems", DATE 2001.

[Lin97] : Bill Lin, Vercauteren Steven, Hardware/Software Communication and system Integration For Embedded Archtectures.

[Lys02] Lysecky Roman, Vahid Frank "Prefetching for improve bus wrapper performance in cores" ACM-Transaction 2002

[Son02] http://www.sonicscorp.com

[Ver98] Vercauteren S., ‘Hardware/Software co-design of application-spécific Heteregeneous architectures’, IMEC, rapport de these, Déc. 1998.

[Vsi02] Virtual Socket Inetrface Alliance, http://www.vsi.org.

[Zit02] A. Zitouni, M. Abid, K. Torki, R. Tourki, "Communication synthesis techniques for multiprocessor systems”, Journal : International Journal of Electronic 2002, VOL. 89, NO. 1, 55-76.