Chapter 8
High-Level Programming of Dynamically Reconfigurable NoC-Based Heterogeneous Multicore SoCs

Wim Vanderbauwhede
University of Glasgow, UK

ABSTRACT
With the increase in System-on-Chip (SoC) complexity and CMOS technology capabilities, the SoC design community has recently observed a convergence of a number of critical trends, all of them aimed at addressing the design gap: the advent of heterogeneous multicore SoCs and Networks-on-Chip and the recognition of the need for design reuse through Intellectual Property (IP) cores, for dynamic reconfigurability and for high abstraction-level design. In this chapter, we present a solution for High-level Programming of Dynamically Reconfigurable NoC-based Heterogeneous Multicore SoCs. Our solution, the Gannet framework, allows IP core-based Heterogeneous Multicore SoCs to be programmed using a high-level language whilst preserving the full potential for parallelism and dynamic reconfigurability inherent in such a system. The required hardware infrastructure is small and low-latency, thus adding full dynamic reconfiguration capabilities with a small overhead both in area and performance.

INTRODUCTION
Networks-on-Chip (NoCs) provide a scalable, efficient and performant communication medium to interconnect complex IP cores. To facilitate inter-operation between IP cores, a number of standards have been proposed (e.g. VSIA, OCP/IP) [Kogel et al., 2005] regarding the interface between the IP cores and the communication medium. The purpose of such standards is to facilitate design reuse and as such they are aimed primarily at IP core developers. By their very nature, IP cores are agnostic of the system in which they are deployed. Consequently, the datapaths between the cores are not governed by the IP cores. Thus, a system consisting solely of IP cores and a communication medium is not dynamically reconfigurable. A Dynamically Reconfigurable SoC therefore requires an additional infrastructure to support reconfiguration of the cores and the datapaths. Moreover, to reduce design time,
it is essential that a Dynamically Reconfigurable SoC can be programmed at high level.

In this chapter we analyse the requirements of such an infrastructure, in particular in view of allowing high-level programmability. We present our architecture of a dynamic reconfiguration infrastructure (DRI), the Gannet framework.

The chapter starts with the background to this work. We then analyse the requirements for NoC-based SoC infrastructures to support dynamic reconfiguration. The subsequent sections discuss the key components of the Gannet framework:

- The Gannet Dynamic Reconfiguration Infrastructure,
- The Gannet language used to program the system,
- The Gannet Machine Model, a formal model used to explain how the Gannet DRI executes Gannet programs.

Section “Examples of Dynamic Reconfiguration” presents examples of dynamic reconfiguration of communication (data path) and computation (IP core). In section “Implementation of the Dynamic Reconfiguration Infrastructure” we discuss the implementation of the Gannet DRI. The chapter concludes with an overview of the current status of the project and avenues for future research.

BACKGROUND

Reconfigurable architecture platforms are gaining increasing popularity as flexible, low-cost solutions for a variety of media processing applications. They contribute to bridging the gap between general purpose processors and application specific circuits. As the application domain for reconfigurable platforms further expands, it becomes imperative that these systems provide a high degree of flexibility and adaptability, while at the same time being extremely cost-efficient. For many applications, fine-grained reconfigurable platforms such as FPGAs are not an acceptable choice because of their high area overhead and reduced performance compared to ASIC solutions. As a result, recent times have seen more focused research on coarse-grained reconfigurable architectures (CGRA), which constitute a middle ground between ASICs and FPGAs in terms of performance-versus-flexibility.

Using the terminology from the review paper on CGRAs by [ul Abdin and Svensson, 2008], the Gannet platform [Vanderbauwhede, 2008, Vanderbauwhede et al., 2008, Vanderbauwhede, 2007] is an Array of Functional Units - as opposed to Hybrid Architectures and Arrays of Processors. The former are typically using a combination of an ordinary processor with a reconfigurable fabric, e.g. [Singh et al., 2000, Mishra et al., 2006] in the latter the reconfigurable fabric consists of fully-featured processors, e.g. [Butts, 2007, Taylor et al., 2004]). Within its category, Gannet bears some similarity to MATRIX [Mirsy and Deon, 1996], Silicon Hive [Cocco et al., 2004] and MORA [Lanuzza et al., 2007, Profit et al., 2008], as it shares with these architectures the local-memory processing model, i.e. every functional unit has its own local memory. However, Gannet distinguishes itself from the other architectures by providing a generic interface layer to third-party processing elements where all other architectures discussed in [ul Abdin and Svensson, 2008] use either dedicated functional units or ordinary processors.

In contrast to most CGRAs, the Gannet platform was specifically designed as a reconfiguration infrastructure for SoCs using packet-switched NoCs. From that perspective, Gannet has similar goals as the work done at IMEC [Nollet et al., 2005, Nollet et al., 2004, Marescaux et al., 2004] on operating system control of NoC routing tables and support for run-time reconfiguration of functional units. The two approaches can be considered complementary as Gannet focuses on providing an interface layer which facilitates high-level programming of the interactions between the