Chapter 3.10
Implementation of FFT on General-Purpose Architectures for FPGA

Fabio Garzia
Tampere University of Technology, Finland

Roberto Airoldi
Tampere University of Technology, Finland

Jari Nurmi
Tampere University of Technology, Finland

ABSTRACT

This paper describes two general-purpose architectures targeted to Field Programmable Gate Array (FPGA) implementation. The first architecture is based on the coupling of a coarse-grain reconfigurable array with a general-purpose processor core. The second architecture is a homogeneous multi-processor system-on-chip (MP-SoC). Both architectures have been mapped onto two different Altera FPGA devices, a StratixII and a StratixIV. Although mapping onto the StratixIV results in higher operating frequencies, the capabilities of the device are not fully exploited. The implementation of a FFT on the two platforms shows a considerable speed-up in comparison with a single-processor reference architecture. The speed-up is higher in the reconfigurable solution but the MP-SoC provides an easier programming interface that is completely based on C language. The authors' approach proves that implementing a programmable architecture on FPGA and then programming it using a high-level software language is a viable alternative to designing a dedicated hardware block with a hardware description language (HDL) and mapping it on FPGA.

DOI: 10.4018/978-1-61350-456-7.ch3.10

INTRODUCTION

During recent years the popularity of FPGAs has grown increasingly because of the reduced cost and development time of FPGA-based solutions. It is well known that the FPGA implementation is more convenient for smaller market volumes and this is generally the case of embedded systems (Sangiovanni-Vincentelli, 1993). In addition, the boundary line that defines when it is better to use an Application-Specific Integrated Circuit (ASIC) is moving toward larger volumes with the advancement of silicon technologies, allowing the exploration of FPGA-based architecture for different application fields.

However, a FPGA-based approach still has some drawbacks: a larger area, less performance, and higher power consumption when compared to an ASIC approach (Kuon & Rose, 2006). Even though this trend is also changing in favor of FPGAs, they are roughly one order of magnitude slower than standard cell ASIC implementations and two orders of magnitude compared with full custom approaches (Chinnery & Keutzer, 2000).

FPGA implementation is usually associated with hardware design. A digital circuit is designed following a RTL flow but its synthesis is based on FPGA resources instead of standard cell library components as in the case of ASIC implementation. The main advantage is that the hardware synthesis is faster and the manufacturing costs follow the trend that we mentioned above. However, it is possible to implement a general-purpose system on a FPGA as well, e.g. a single-processor system. In this case we can also reduce the cost of the initial design, which is implemented programming the processor in software rather than creating a dedicated hardware architecture using a HDL. Processor-based architectures can be programmed using C language or similar programming languages, which are more popular than HDLs. This approach has a drawback: a digital system based on a processor is usually slower than a system based on dedicated hardware. However, in many situations the flexibility of a general-purpose system is an unavoidable requirement.

This paper proposes two solutions based on general-purpose hardware for FPGAs. In these two solutions, we try to overcome the performance degradation due to the choice of FPGA implementation and the adoption of a general-purpose approach. The evaluation of these two solutions is based on the implementation of an N-point FFT algorithm. The FFT is used in various multimedia applications, for example image processing, audio processing, and video compression. In addition, Software Defined Radio (SDR) applications use the FFT for OFDM-based wireless protocols such as IEEE802.11a/g and WiMAX. The development of SDRs pushes towards novel architectural solutions because it needs flexibility and high performance at the same time.

The most common way to map an FFT on a general-purpose system is to use a software implementation on a digital signal processor (DSP). It is not a case that this approach was already under study in the 90s (Meyer & Schwarz, 1990). A DSP provides a general-purpose architecture optimized for signal-processing algorithms.

An alternative that is very popular nowadays is the use of coarse-grain reconfigurable hardware. The programmability of these devices is comparable with CPUs and DSPs, even though they require a better knowledge of the architecture to achieve the best results. These devices are based on arrays of homogeneous processing elements (PE). These PEs can be as simple as an ALU or as complex as a processor. The interconnections can be fixed or based on a network approach. Typically it is possible to define at run-time the functionality of each single PE as well as the interconnections between them.

The FFT has been used as a case study in the proposal of such solutions. One of the most successful devices is Morphosys, Kamalizad, Pan, and Bagherzadeh's (2003) showing that the mapping of the FFT on their Morphosys2 was the most efficient. Very good results were achieved by XPP