Chapter 1

Introduction

1.1 Motivation

Over the last decades, numerous improvements in processing performance of digital systems have significantly increased the data-flow requirements. Unfortunately, the system's data-flow hardware has not gone through similar advances, resulting in a growing performance gap between the two. This growing gap can partly be related to the used process technology, and partly to the actual architecture. Specifically, an optimal solution to the latter is closely related to the application being implemented. For example, for concurrent memory accesses the access locality will influence the optimal data splitting and memory architecture. Due to the increasing significance of this performance gap, the area of memory optimisation has seen an increased research interest in recent years.

In addition, the electronic industry is currently facing an increase in development and Non-Recurrent Engineering (NRE) costs of Application Specific Integrated Circuits (ASIC). The demand for lower costs per device in combination with the reduced time-to-market has forced manufacturers to use reconfigurable devices as alternative implementation platforms to ASICs and/or micro-processors. Their flexibility, allow

them to be used in a wide range of applications. In comparison to ASICs they are more price-efficient for small to medium volumes. Therefore, they are often used to launch the product even before an ASIC implementation has been created.

Designing hardware has long been an exclusive area for hardware design engineers. As a result of the increasing number of products that need to be developed in a shorter time frame, it would be useful if the design community can be extended and/or the productivity of the existing design teams improved. Increasingly, digital integrated circuits are designed in higher level hardware description languages and mapped to hardware through synthesis. In developing an application for a reconfigurable device, a large amount of time is devoted to the memory architecture implementation. If the design subsequently needs to be re-implemented as an ASIC, then this design effort might need to be repeated. This could be overcome by providing memory architecture design abstractions which could offer platform-independent implementation.

1.2 Objective

The primary objective of the presented work is to investigate the possible improvements in using memory for structured data accesses. The behaviour of the structured data accesses is assumed to be known at compile/synthesis time and therefore not data-dependent. The present work comprises the actual hardware necessary for this structured address generator as well as the software framework to support these structured accesses as abstractions.

1.3 Statement of Originality

The work presented in this thesis contributes to several areas. It proposes the extension of reconfigurable architectures to improve the data-flow for structured memory accesses. Moreover, this has been extended towards the higher degrees of parallelism

often found within applications implemented on these devices. As a consequence, these memory architecture solutions can be used as design abstractions, significantly influencing synthesis. The presented memory architectures have been applied to two case studies, revealing the large variety of possible architectures and the respective benefits of each of them.

The main contributions made are summarised below. Several other contributions in relation to a specific chapter are then listed in the introduction section of the respective chapter.

- A dedicated address generator for extending memory in reconfigurable architectures with a flexible address generator that provides a number of structured access modes, consequently, resulting in an Autonomous Memory Block (AMB) (Section 3.2) [69]. This dedicated address generator demonstrates an area advantage and performance improvement of up to 18 and 2.6 times, respectively, in comparison to using the reconfigurable resources. Alternatively, these modes can be offered as a library of modules, suitable for a platform-independent approach.

- Examine the scalability of memory blocks with localised address generation to deal with larger storage requirements as well as parallelism within applications (Section 4.3) [70]. A design exploration, comparing different memory and address generation architectures for a 2D filter, displays an area and performance variation up to 55 and 2.7 times, respectively.

- Investigate the possible memory architectures in relation to the different amounts of data buffering for two model applications with large data-flow demands (Sections 5.2 and 5.3).

1.4 Overview

A literature review of related areas is presented in Chapter 2. The topics of: data-flow optimisation within a computing system, memory architectures, address generation, synthesis towards memory and reconfigurable systems have been discussed as much as necessary for later presentation of the experimental work and discussion.

Chapter 3 presents abstractions for memory design. These abstractions offer a set of frequently occurring data access patterns, commonly found within Digital Signal Processing (DSP) applications. Each of these modes of access share the same architecture for implementation, which reduces the required resources. These structured addressing modes can then be provided as dedicated hardware or implemented using reconfigurable resources. The former links an address generator to each memory block, which is essential for concurrent accesses with the data split across several memory blocks.

The scalability of these memory blocks for larger storage requirements as well as high degrees of parallelism is discussed in Chapter 4. Storage for increased parallelism requires data splitting to allow simultaneous accesses and is dependent on the locality of the items accessed for processing. Different methods of concatenation to extend the storage capacity as well as to provide higher degrees of parallelism are studied and compared.

In Chapter 5 the usefulness of the techniques presented in Chapters 3 and 4 is demonstrated with two case studies. The first application presents an extension to an image registration application with large data-flow demands. The implementation is studied with respect to different sizes of on-chip buffering, the respectively possible memory architectures and achieved performance. The second example is the well-known Joint Photographic Expert Group (JPEG) encoding algorithm, for which a memory architecture for implementation is presented.

Finally, Chapter 6 summarises the main contributions of this thesis and argues

possible future research directions.