The Journal of Supercomputing, 26, 167–184, 2003

PACT XPP—A Self-Reconfigurable Data Processing Architecture

V. BAUMGARTE
PACT Informationstechnologie AG, Muthmannstr. 1, D-80939 München, Germany

G. EHLERS
PACT Informationstechnologie AG, Muthmannstr. 1, D-80939 München, Germany

F. MAY
PACT Informationstechnologie AG, Muthmannstr. 1, D-80939 München, Germany

A. NÜCKEL
PACT Informationstechnologie AG, Muthmannstr. 1, D-80939 München, Germany

M. VORBACH
PACT Informationstechnologie AG, Muthmannstr. 1, D-80939 München, Germany

M. WEINHARDT
PACT Informationstechnologie AG, Muthmannstr. 1, D-80939 München, Germany

Abstract. The eXtreme Processing Platform (XPP^{TM}) is a new runtime-reconfigurable data processing architecture. It is based on a hierarchical array of coarsegrain, adaptive computing elements, and a packet-oriented communication network. The strength of the XPP^{TM} technology originates from the combination of array processing with unique, powerful run-time reconfiguration mechanisms. Parts of the array can be configured rapidly in parallel while neighboring computing elements are processing data. Reconfiguration is triggered externally or even by special event signals originating within the array, enabling self-reconfiguring designs. The XPP^{TM} architecture is designed to support different types of parallelism: pipelining, instruction level, data flow, and task level parallelism. Therefore this technology is well suited for applications in multimedia, telecommunications, simulation, signal processing (DSP), graphics, and similar stream-based application domains. The anticipated peak performance of the first commercial device running at 150 MHz is estimated to be 57.6 GigaOps/sec, with a peak I/O bandwidth of several GByte/sec. Simulated applications achieve up to 43.5 GigaOps/sec (32-bit fixed point).

Keywords: reconfigurable processor, adaptive computing, run-time reconfiguration, partial reconfiguration, XPP

1. Introduction

The limitations of conventional processors are becoming more and more evident. The growing importance of stream-based applications makes reconfigurable architectures an attractive alternative [17]. They combine the performance of ASICs with the flexibility of processors.

This paper presents the eXtreme Processing Platform (XPP™), a new family of runtime- and self-reconfigurable IP cores (XPP™ Cores) and processors [1, 15]. A prototype device, the XPU128-ES, has been produced in silicon by PACT. The XPP™ architecture is designed to support different types of parallelism: pipelining, instruction level, data flow, and task level parallelism. Therefore this technology is well suited for applications in multimedia, telecommunications, simulation, signal processing (DSP), graphics, and similar stream-based application domains.

1.1. The XPP idea

This section gives a brief introduction to XPP™ data processing. The main idea is to combine data-stream processing in an array configuration with sophisticated run-time reconfiguration mechanisms. Configurations are parallel computation modules derived from a data-flow graph of an algorithm. Nodes of the data-flow graph are mapped to fundamental machine operations such as multiplication, addition etc. The operations are implemented by configurable ALUs. The ALUs communicate via an automatically synchronizing, packet-oriented communication network. Figure 1 shows a small configuration performing a parallel matrix-matrix multiplication. In this algorithm, data packets continuously stream through a single configuration. During the computation, the graph remains static, i.e., no operators or connections are changed. No reconfiguration or instruction decoding is required, and all operators and input and output ports are active in each cycle, resulting in the optimal performance [9].

Several of these configurations can be executed sequentially on an XPP™ Core. Since each configuration processes long data streams, the reconfiguration overhead is amortized over many parallel operations. Results of computations are stored in distributed memories or FIFOs for use by subsequent configurations [10]. We call this programming paradigm configuration flow, as opposed to the instruction flow in a classical Von-Neumann architecture. The difference is illustrated in Figure 2. This programming paradigm is highly suited to computation-intensive applications since many of them can be separated into smaller, inherently parallel phases.

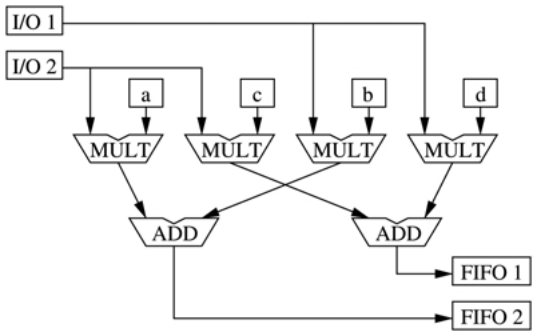

Figure 1. A small configuration: matrix-matrix multiplication.