Loading $\rho\mu$-Code: Design Considerations

Georgi Kuzmanov, Georgi Gaydadjiev, and Stamatis Vassiliadis

Computer Engineering Laboratory, 
Electrical Engineering Dept., EEMCS, TU Delft, The Netherlands
{G.Kuzmanov,G.N.Gaydadjiev,S.Vassiliadis}@ET.TUDelft.NL
http://ce.et.tudelft.nl/

Abstract. This article investigates microcode generation, finalization and loading in MOLEN $\rho\mu$ processors. In addition, general solutions for these issues are presented and implementation for Xilinx Virtex-II Pro platform FPGA is introduced.

Keywords: Reconfigurable architectures, MOLEN, implementation, loading microcode.

1 Introduction

Reconfigurable hardware extensions of general purpose processors (GPP) have indicated considerable potentials for speed-ups of computationally demanding algorithms. Numerous design concepts and organizations have been proposed to support the Custom Computing Machine (CCM) paradigm from different perspectives [2–4, 6]. An example of a detailed classification of CCMs can be found in [5]. Recently, the MOLEN $\rho\mu$—processors for CCM organizations have been proposed [7]. The MOLEN concept provides a flexible and easily extendable framework for hardware/software co-design of complex computing systems by extending the traditional microcode. The presented paper addresses some specific issues related to the microcode design and maintenance within the MOLEN processors. More specifically, we investigate the problems related to the generation, memory alignment and loading of configuration microcodes.

Hereafter, the discussion is organized as follows. Section 2 gives a brief background on the MOLEN organization. Section 3 introduces the FPGA configuration format for the targeted Xilinx technology. In Section 4, problems related to generation, alignment and loading of reconfigurable microcodes are discussed. Section 5 proposes solutions to different problems with respect to efficient hardware implementations. Finally, the discussion is concluded in Section 6.

2 The MOLEN Organization

This section presents the MOLEN $\rho\mu$-coded Custom Computing Machine organization, introduced in [7] and illustrated in Figure 1. The ARBITER performs a partial decoding on the input instructions flow in order to determine where they should be issued. The arbiter controls the proper co-processing of the GPP and the reconfigurable units. Figure 2 depicts a general design of a MOLEN arbiter. It is closely connected to three major components of the CCM: the GPP,
the memory and the $\rho\mu$-unit. Instructions implemented in fixed hardware are issued to the core processor (GPP). Instructions for custom execution are redirected to the reconfigurable unit, referred to as $\rho\mu$-unit. The reconfigurable unit consists of a custom computing unit (CCU) and the $\rho\mu$-code unit. An operation, executed by the reconfigurable unit, is divided into two distinct phases: set and execute. The set phase is responsible for reconfiguring the CCU hardware enabling the execution of the operation. This phase may be divided into two subphases - partial set ($\text{pset}$) and complete set ($\text{cset}$). In the $\text{pset}$ phase the CCU is partially configured to perform common functions of an application (or group of applications). Later, the $\text{cset}$ sub-phase only reconfigures those blocks in the CCU, which are not covered in the $\text{pset}$ sub-phase in order to complete the functionality of the CCU.