Simultaneous Multiprocessing on a FPGA+CPU Heterogeneous System-On-Chip

Jose NUNEZ-YANEZ [a] Mohammad HOSSEINABADY [a]
Andrés RODRÍGUEZ [b] Rafael ASENJO [b] Angeles NAVARRO [b]
Rubén GRAN-TEJERO [c] Darío SUÁREZ-GRACIA [c]
[a] *University of Bristol, United Kingdom*
[b] *Universidad de Málaga, Spain*
[c] *Universidad de Zaragoza, Spain*

Abstract. In this paper, we investigate how to enhance an existing software-defined framework to reduce overheads and enable the parallel utilization of all the programmable processing resources present in systems that include FPGA-based hardware accelerators. To remove overheads, a new hardware platform is created based on interrupts, which removes spin-locks and frees the processing resources. Additionally, instead of simply using the hardware accelerator to offload a task from the CPU, we propose a scheduler that dynamically distributes the tasks among all the resources to minimize load unbalance. The experimental evaluation shows that the interrupt-based heterogeneous platform increases performance by up 22% while reducing energy requirements by 15%. Additionally, we measure between 50% to 25% reduction in execution time when the CPU cores assist FPGA execution at the same level of energy requirements depending on hardware speed-ups.

Keywords. FPGAs, heterogeneous, interrupts, dynamic scheduler, performance improvement, energy reduction.

1. Introduction

Heterogeneity is seen as a path forward for computers to deliver the energy and performance computing improvements needed over the next decade. In heterogeneous architectures, specialized hardware units accelerate complex tasks. A good example of this trend is the introduction of GPUs (Graphics Processing Units) for general purpose computing combined with multicore CPUs. FPGAs (Field Programmable Gate Arrays) are an alternative high performance technology that offer bit-level parallel computing in contrast with the word-level parallelism deployed in GPUs and CPUs. In a typical configuration, the host CPU employs the FPGA accelerator to offload the work and then remains idle. In this research, we investigate a cooperative strategy applied to compute intensive applications in which both the CPU and FPGA perform the same task on different regions of the input data. The proposed scheduling algorithm dynamically distributes different