Abstract

In the era of big data, advanced sensing, and artificial intelligence, the required computation power is provided mostly by multicore and manycore architectures. However, the performance demand keeps growing. Thus the computer architectures need to continue evolving and provide higher performance. The applications, which are executed on the manycore architectures, are divided into several tasks to be mapped on separate cores and executed in parallel. Usually these tasks are not identical and may be executed more efficiently on different types of cores within a heterogeneous architecture. Therefore, we believe that the heterogeneous manycores are the next step for the computer architectures. However, there is a lack of knowledge on what form of heterogeneity is the best match for a given application or application domain. This knowledge can be acquired through designing these architectures and testing different design configurations. However, designing these architectures is a great challenge. Therefore, there is a need for an automated design method to facilitate the architecture design and design space exploration to gather knowledge on architectures with different configurations. Additionally, it is already difficult to program manycore architectures efficiently and this difficulty will only increase further with the introduction of heterogeneity due to the increase in the complexity of the architectures, unless this complexity is somehow hidden. There is a need for software development tools to facilitate the software development for these architectures and enable portability of the same software across different manycore platforms.

In this thesis, we first address the challenges of the software development for manycore architectures. We evaluate a dataflow language (CAL) and a source-to-source compilation framework (Cal2Many) with several case studies in order to reveal their impact on productivity and performance of the software. The language supports task level parallelism by adopting actor model and the framework takes CAL code and generates implementations in the native language of several different architectures.

In order to address the challenge of custom hardware development, we first evaluate a commercial manycore architecture namely Epiphany and identify its demerits. Then we study manycore architectures in order to reveal possible uses of heterogeneity in manycores and facilitate choice of architecture for software and hardware development. We define a taxonomy for manycore architectures that is based on the levels of heterogeneity they contain and discuss the benefits and drawbacks of these levels. We finally develop and evaluate a design method to design heterogeneous manycore architectures customized based on application requirements. The architectures designed with this method consist of cores with application specific accelerators. The majority of the design method is automated with software tools, which support different design configurations in order to increase the productivity of the hardware developer and enable design space exploration.

Our results show that the dataflow language, together with the software development tool, decreases software development efforts significantly (25-50%), while having a small impact (2-17%) on the performance. The evaluation of the design method reveal that the performance of automatically generated accelerators is between 96-100% of the performance of their manually developed counterparts. Additionally, it is possible to increase the performance of the architectures by increasing the number of cores and using application specific accelerators, usually with a cost on
the area usage. However, under certain circumstances, using accelerator may lead to avoiding usage of large general purpose components such as the floating-point unit and therefore improves the area utilization. Eventually, the final impact on the performance and area usage depends on the configurations. When compared to the Epiphany architecture, which is a commercial homogeneous manycore, the generated manycores show competitive results. We can conclude that the automated design method simplifies heterogeneous manycore architecture design and facilitates design space exploration with the use of configurable parameters.