A Study on Efficient Application Mapping on Parallel Computing Accelerators

Graduate School of Engineering, Nagasaki University
Keisuke Dohi

Since the invention of electronic computers, their performance has been constantly advanced. The recent progress of micro processors in performance has been mainly achieved by increasing the number of cores on a device, instead of increasing working frequency. In addition, because of increasing of density of semiconductors, not only computational performance but also density of power consumption has been steadily growing, and this trend is expected to restrict the performance of computers as a monolithic system. Therefore, increasing energy efficiency of the computers along with computational performance is now a critical issue. Against a backdrop of these trends, both hardware and software have new challenges; a) suppressed operating frequency; b) large amount of but poor processors; c) power consumption wall; and d) narrow bandwidth technologies for external memory. Both Graphics Processing Unit (GPU) and Field Programmable Gate Array (FPGA) have considerable performance/ cost benefits because these are massively manufactured as consumer devices, and thus these are expected to be applicable as parallel computing accelerators in various fields. In addition, this dissertation focuses high-level synthesis tools that generate desired hardware logic using high abstraction layer descriptions. Although parallel computing accelerators include GPUs and FPGAs are available, efficient mapping of desired applications on the accelerators is still an important and difficult issue. Therefore, this dissertation addresses efficient application mapping on parallel computing accelerators using on-chip memories. This dissertation focuses stencil computation and stream-oriented process as design patterns on the accelerators, and demonstrate concrete application mappings from the design patterns.

After describing motivation and background of this study in Chapter 1 to Chapter 3, in Chapter 4, implementation of Smith-Waterman algorithm on GPUs is presented. Operations of the algorithm are massively parallelizable. In addition, the computation requires relatively high ratio of integer arithmetic to memory access. Therefore, costs of synchronizing arithmetic cores and on-chip memory access impact computational performance significantly. Implementing the algorithm on a GPU shows the importance of effective utilization of the warp-level parallelism on a GPU. Central to this technique is a divide and conquer approach to alignment matrix calculation in which a whole pairwise alignment matrix is subdivided. This leads to the efficient calculation of data by 32 threads, and the reduction in the number of load/ stores to/ from external memory. As a result of evaluation, the implementation of the algorithm achieved a throughput ranging between 9.09 Giga Cell Updates

Per Second (GCUPS) and 12.71 GCUPS on a single-GPU version, and a throughput between 29.46 GCUPS and 43.05 GCUPS on a quad-GPU platform, which was the world fastest GPU implementation of the algorithm at that time.

In Chapter 5, an implementation of electro-magnetical simulation by a finite-deference time-domain (FDTD) method for analysis of micro strip antenna characteristics on a GPU is presented. The algorithm is known as a kind of stencil computation that has a high degree of parallelism. The implementation uses Perfectly Matched Layer (PML) as a boundary condition, hence memory accessing pattern and update-equations are changed depend on location of grids. The transformation technique of update-equations for partial boundary cells is also proposed. The empirical experiment showed the memory bandwidth of 62.5 GB/ s which corresponds to 55.8 % of the peak of the target GPU.

In Chapter 6, implementation of human detection from video image using Histograms of Oriented Gradients (HOG) feature on an FPGA is presented. Since the HOG features are extracted from luminance gradient of an image they have high robustness for lighting condition and are widely used to detect a human in an input video image. In order to cope a lot of computation costs of the HOG feature extraction, the architecture is designed in a data stream oriented deep pipelined manner. As a result of evaluation, the throughput of 62.5 FPS was achieved without using any external memory modules.

In Chapter 7, implementation of 3-D heat spreading simulation using MaxCompiler, which is a high-level synthesis tool based on Java on an FPGA is presented. This chapter aims to establish estimation models for computational performance and resource utilization using user parameters on high abstraction layer description. In addition, energy consumption of accelerator is measured for various parameter configurations.

As a result of evaluation, the best configuration achieved about six times faster than CPU implementation in performance, and the proposed estimation model provided us reasonable estimation about performance and resource utilization.

In Chapter 8, implementation of ellipse estimation from video image using RANSAC algorithm on an FPGA is presented. The algorithm needs to solve simultaneous equations as much as possible to get a reasonable solution. To solver simultaneous equations, three types of algorithms were implemented. As a result of evaluation, the throughput of 62.5 FPS was achieved with 3.34 W power consumption. While the optimal algorithm needs to be chosen depending on the amount of resources on FPGAs and required criteria, the FPGA based system that consists of streamed structure is promised as a better solution for the application.

The results of implementation showed both GPUs and FPGAs have advantages over existing microprocessor architectures in computational performance and energy consumption. At the same time, modification of existing algorithms played a significant role in achieving a high degree of computing efficiency with the parallel computing accelerators. Especially, a view of making the best use of on-chip memory with a deep pipelined manner was crucial.