Understanding How OpenCL Parameters Impact on Off-Chip Memory Performance of FPGA Platforms

Yingyi Luo,[1] Zheming Jin,[2] Kazutomo Yoshii,[2] Seda Ogrenci-Memik[1]

[1]Northwestern University
[2]Argonne National Laboratory

1 INTRODUCTION

Reconfigurability has strong potential to achieve higher performance and energy efficiency in the post-Moore era. Many studies show that field-programmable gate arrays (FPGAs), the most practical reconfigurable architecture today, outperform their counterparts. In such studies, FPGA designs are written by experienced hardware engineers using hardware description language (HDL) such as Verilog and VHDL. Unfortunately, developing HDL-based designs is too costly for high-performance computing and lacks proper abstraction to express numerical computing algorithms. Now emerging FPGA high-level synthesis (HLS) technology could open up an opportunity for software developers. Most notably, FPGA vendors started supporting OpenCL for FPGA platforms, and some OpenCL-based codes have been ported to FPGAs. However, OpenCL offers no guarantee for performance portability; optimal OpenCL parameters such as global size and local size are different between platforms [1], which could lead to unfair comparisons. In successful FPGA demonstrations, data may be placed in FPGA internal memory to avoid off-chip memory accesses, but off-chip memory accesses are unavoidable for HPC workloads. In this study, our objective is two folds: 1) to understand how OpenCL parameters impact off-chip memory access performance of the current generation of OpenCL-FPGA platforms and 2) to find effective OpenCL parameters empirically from microbenchmark results.

2 EXPERIMENTS

We choose two distinct memory access patterns to find the optimal OpenCL parameters. The OpenCL parameters we consider in this study include global and local size, data type, single-instruction multiple data (SIMD) size, and the number of compute units (CUs). Global and local size are common OpenCL parameters and can be changed by the host code at runtime. While SIMD size and the number of CUs are basically fixed for instruction-set architecture such as CPU and GPU, these parameters on reconfigurable FPGAs can be changed via OpenCL attribute specifiers, thus expanding the parameter search space.

The two microbenchmarks we use are described below.

VectorAdd: performs a single-precision floating-point vector add in a data-parallel manner, which yields two loads and one store. This regular memory access pattern appears commonly in many HPC workloads. The size of each vector is 512 M (6 GB of memory usage in total).

RandomIndex: iterates a load operation that reads data from a prefilled array and uses the read data for the next load index, which resembles pointer chasing and appears in irregular algorithms such as breadth-first search. This kernel is expressed in a hybrid task-parallel (loop) and data-parallel model. The number of iterations per work item is determined by the array size and the global size. The total memory usage for this kernel is 6 GB.

2.1 Experimental Setup

Our target platform is the Nallatech 385A acceleration card, which includes one Altera Arria10 [2] 1150 GX FPGA chip and two channels of 4 GB DDR3L-2133 memory (8 GB in total). The theoretical DDR3 memory bandwidth limit of the card is approximately 34 GB/s (2133 MT/s * 8 bytes * 2). We use Intel FPGA SDK for OpenCL to compile our OpenCL benchmark codes. For measuring the FPGA board power consumption, the Nallatech OpenCL board support package (BSP) provides memory-mapped device (MMD) library functions that can be called to monitor the board power consumption in real time. The idle power of the FPGA board measured from this tool is approximately 29 W.

2.2 Results

For *VectorAdd*, we set the globe size to be equal to the number of array elements due to the data-parallel kernel. Finding the optimal local size is the first step. We empirically determine that the local size should be greater than 128; the performance with local size of 1 is approximately 60 times slower. Figures 1 and 2 show the impact of SIMD size and the number of CUs on the performance and energy efficiency, respectively. The larger SIMD size performs better because the compiler can combine memory operations to form wider load or store requests. More CUs can exploit more memory bank-level parallelism, but diminishing returns occur at 16

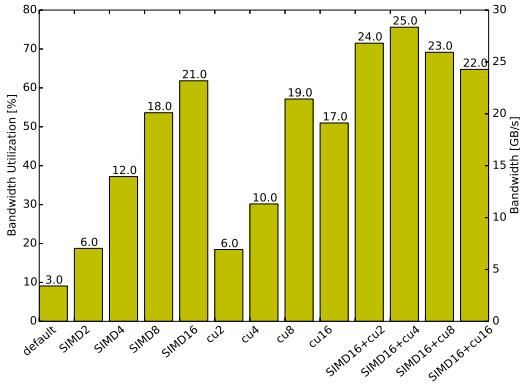

Figure 1: VectorAdd: Impact of SIMD/CU on Performance

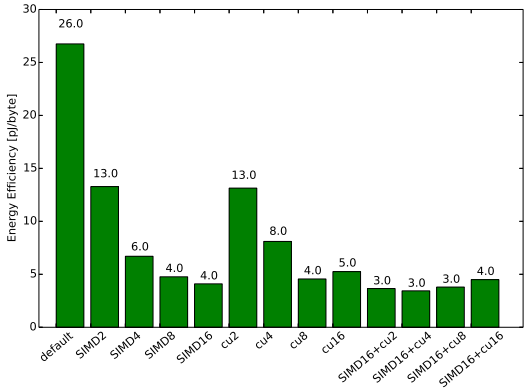

Figure 2: VectorAdd: Impact of SIMD/CU on Energy

CUs. The combination of SIMD16 and CU4 yields the best performance.

For *RandomIndex*, we vary the global size. Since the kernel requires a loop iteration, the kernel cannot be expressed by pure data parallelism; the local size is always set to 1. Figures 3 and 4 illustrate that 128 or more work groups are required to maximize both the bandwidth utilization and energy efficiency. When the number of work items is small, the generated kernel pipeline will be severely underutilized. *RandomIndex* has no opportunity for SIMDization. The only way to improve the bandwidth is to increase the data access size. Up to *ulong8*, both the bandwidth utilization and the energy efficiency improve linearly compared with smaller data types. Since the 6 GB array size cannot fit into on-chip local memory and the order of the array element access is random, each request of the array element may require a global memory transaction. We also note that the performance improvement by increasing the number of CUs for RandomIndex is negligible compared with the VectorAdd case. The reason is that all the compute units share the global memory interconnect. For memory bounded tasks, more compute units do not increase performance.

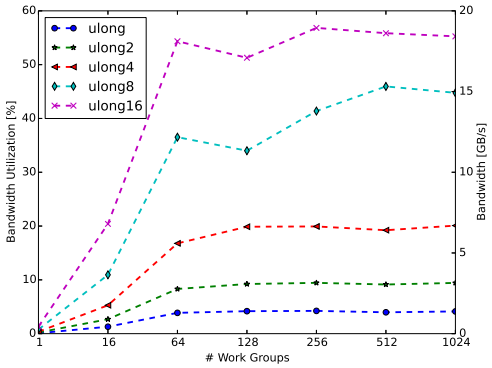

Figure 3: RandomIndex: Impact of # of Work Groups

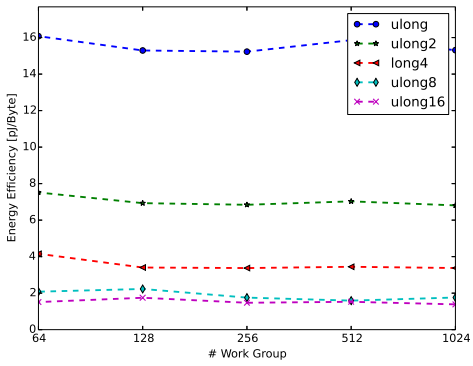

Figure 4: RandomIndex: Impact of # of Work Groups

3 CONCLUSION

In this study, we design microbenchmarks to understand how OpenCL parameters impact on off-chip memory performance and measure the bandwidth and energy efficiency of two distinct memory access patterns on an Arria10-based board. Finding right global size and local size is the first important step; the single work-item kernel performs poorly on both VectorAdd and RandomIndex (up to 60x slower than the best parameters). Our experimental results show that larger SIMD size and larger data type, which yield wider memory operations, improve both memory bandwidth utilization and energy efficiency. OpenCL FPGAs design can achieve 75% (VectorAdd) and 55% (RandomIndex) of the memory bandwidth compared to the theoretical peak. We plan to continue this study, targeting next generation FPGA platforms, and develop a prediction model to predict both performance and energy efficiency.

REFERENCES

[1] Sangmin Seo, Jun Lee, Gangwon Jo, and Jaejin Lee. 2013. Automatic OpenCL work-group size selection for multicore CPUs. In *Proceedings of the 22nd international conference on Parallel architectures and compilation techniques*. IEEE Press, 387–398.

[2] Jeffrey Tyhach, Mike Hutton, Sean Atsatt, Arifur Rahman, Brad Vest, David Lewis, Martin Langhammer, Sergey Shumarayev, Tim Hoang, Allen Chan, and others. 2015. Arriaffl 10 device architecture. In *Custom Integrated Circuits Conference (CICC), 2015 IEEE*. IEEE, 1–8.