Stream Computing of Lattice-Boltzmann Method on Intel Programmable Accelerator Card

Takaaki Miyajima, Tomohiro Ueno and Kentaro Sano
Processor Research Team, RIKEN Center for Computational Science, Kobe, Hyogo, 650-0047, Japan
{*takaaki.miyajima, tomohiro.ueno, kentaro.sano*}*@riken.jp*

***Abstract*—Intel Programmable Accelerator Card (Intel-PAC) and Open Programmable Acceleration Engine (OPAE) aim at saving developers time and enabling code re-use across multiple FPGA platforms. We implemented a Lattice-Boltzmann Method (LBM) computing core, a computational fluid dynamics application, on Intel-PAC. We evaluated four designs of LBM core. In our evaluation of four LBM cores, sustained performance of each design is the same as the theoretical peak performance.**

I. Intel-PAC and OPAE

Intel-PAC is a PCIe-connected FPGA board for data centers. It consists of an Arria 10 GX FPGA (10AX115N2F40E2LG), PCIe port and two 4GB DDR4-2133 memory banks (17 GB/s for input/output), as depicted in Figure 1. It also has four QSFP+ interfaces, flash memory and so on. The users can utilize conventional tools but must have the know-how to implement custom logic. In our design, SPGen, our in-house domain specific language, is used to implement LBM core[1]. Intel Platform Designer is used to connect Direct Memory Access Controllers (DMACs) and LBM core. The OPAE is a lightweight user-space library, which includes drivers, application programming interfaces (APIs), and an FPGA interface manager. In our design, C/C++ OPAE APIs are used to manage FPGA resource, control TX/RX DMAC and set the initial condition of LBM core. Note that control software is still under development.

II. Design and Evaluation

Figure 1 depicts an overview of our design with one LBM core. Accelerator Function Unit (AFU) includes two DMACs ($RXDMAC$ and $TXDMAC$), one computing core of LBM (LBM_core) and two width converters ($width\_converter$). Pipeline depth of LBM core is 830. Input width of LBM core is 40 bytes and the width converters change the input width of 64 bytes to the 40 bytes. Details of LBM core is explained in [1]. We set working frequency F at 200 MHz. Thus, one LBM core requires 40 bytes $\times$ 200 MHz = 8 GB/s memory bandwidth. Our design works as follows. Input stream first comes from FPGA memory, stream goes to the LBM core and then goes to another memory bank. We implemented four designs to evaluate the scalability. Each design cascaded one, two, four and eight LBM cores. We can utilize temporal parallelism by cascading LBM cores. For example, the two cascaded LBM cores operate as a computing pipeline to take an input stream and output a data stream as computational results for two time steps. Required bandwidth remains the same even if eight cores are cascaded.

Our evaluation environment uses Intel Quartus Prime Pro 17.1.1, Intel OPAE C/C++ 1.1.0, SPGen 2.3, g++ 4.8.5 and gcc 4.8.5. Resource utilization is described in Table I. The number of maximum cascades is limited by RAM Blocks since eight LBM cores consume 88 % of total resource.

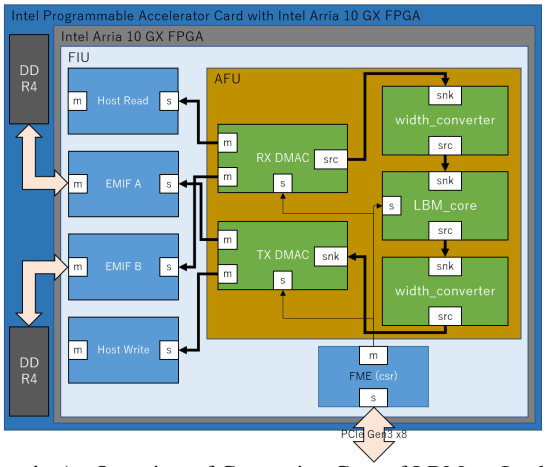

Figure 1: An Overview of Computing Core of LBM on Intel PAC. (Bold line is data path and thin line is MMIO.)

Table I: Total Resource and Utilization of Entire Design

# of cascaded core	1	2	4	8
Logic utilization (427,200)	12 %	15 %	21 %	31 %
Registers	81,206	97,579	130,095	195,239
Block memory bits (55,562,240)	3 %	4 %	7 %	12 %
RAM Blocks (2,713)	18 %	28 %	48 %	88 %
DSP Blocks (1,518)	6 %	11 %	22 %	44 %

We can obtain the theoretical peak performance FN_{ops} of one LBM core by multiplying F with the number of single-precision floating point operations per cycle N_{ops}. In our design, F = 200 MHz and N_{ops} = 131 for 70 adders, 60 multipliers, and 1 divider in the LBM core. As a result, FN_{ops} of one, two, four, and eight cascaded design is 26.2, 52.4, 104.8 and 209.6 GFlops, respectively. We inserted a clock cycle counter to measure the stall rate r_{stall} of LBM core to measure the sustained performance. Sustained performance can be obtained by using Eq (15) in [1] and r_{stall}. When the size of input stream data is 1.92 MB, r_{stall} for all the implementation design becomes 1.04 e-05. Thus, sustained performance for each implementation is the same as theoretical peak performance when the size of input stream is longer than than the pipeline depth of LBM core.

References

[1] K. Sano and S. Yamamoto, "Fpga-based scalable and power-efficient fluid simulation using floating-point dsp blocks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 10, pp. 2823–2837, Oct 2017.