Analysis Of Benchmarks On The Enzian Research Computer

Noah Berner

Introduction

It is widely known, that the rate at which memory speed increases is smaller than the pace at which microprocessors improve. This has been true for several decades and is unlikely to change drastically in the future. Thus, memory bandwidth is expected to be an even bigger bottleneck in computation than it already is today.
A trend in recent years has been to look further than traditional multiprocessing and to explore the possibilities of stream processing on FPGAs. Intel, for example, acquired Altera, a leading FPGA manufacturer, in 2015 and NVIDIA has included Tensor Cores in their Volta architecture, which are streaming processors for matrix multiplication.
It is unlikely that FPGAs will replace traditional CPUs as their size limits their capabilities and running complete programs on FPGAs would make the architecture too complicated and thus ruining the benefits seen when running small kernels on them. They are more likely going to serve as separate accelerators which take over tasks that benefit from being offloaded to FPGAs, much like vector processors are used in today's architectures.
What is missing today however, is the possibility to compile an existing program in a high level language to a bitstream for a computer that houses an FPGA and a CPU such that the performance of the overall program is increased when comparing it to a traditional CPU-only run. The Enzian research computer [1] is the perfect platform to study the behaviour of such a hybrid architecture, as it allows for fast reconfiguration of the hardware and thus is much more versatile than other systems.

Existing solutions

A great number of HLS tools exist, that allow a user to compile high-level source code to hardware circuits on FPGAs [2]. HLS tools are great for fast design exploration and accessible hardware generation, but most of them do not create code or hardware circuits for a CPU-FPGA hybrid architecture.

There are a few compliers that do generate code for hybrid architectures, like CHiMPS [3] and LegUp [4]. CHiMPS [3] compiles ANSI-C to the Xilinx Accelerated Computing Platform (ACP), a hybrid CPU-FPGA platform, that was still under development when the paper was published in 2008. While there were performance gains when using CHiMPS, the practical use of it was limited, as the ACP never caught on. LegUp uses a different strategy by configuring part of the FPGA to be a RISC CPU and thus creating a CPU-FPGA hybrid architecture on one FPGA.

Objectives and goals

The main goal of the thesis is to get closer to a solution for easy programming of a CPU and FPGA hybrid architecture.

Firstly, the two SPEC-benchmarks lbm [5] and cactusADM [6] are analysed when running on a traditional CPU. This serves as a performance baseline and to find the data movements and data size for later FPGA design decisions.

In a second phase, the compute kernel of the benchmarks is converted into a hardware design on an FPGA using HLS. Several designs are explored, especially which data is stored where (Registers, BRAM/UltraRAM).

Depending on whether or not the Enzian data architecture allows it, a next step is to run the two benchmarks completely on Enzian and to optimize the data transfer between the CPU and the FPGA.

As a stretch goal, it is possible to start automating the process for similar programs so that they can be compiled to run on Enzian.

References

[1] Cock et al. (24 February 2019). *Why Enzian?* Retrieved from `http://www.enzian.systems/why-enzian.html`

[2] Meeus, W., Van Beeck, K., Goedemé, T., Meel, J. & Stroobandt, D. (2012). An overview of today's high-level synthesis tools. *Design Automation for Embedded Systems, 16*(3), 31-51. doi:10.1007/s10617-012-9096-8.

[3] Putnam, A., Bennett, D., Dellinger, E., Mason, J., Sundararajan, P. & Eggers, S. (2008). CHiMPS: A C-level compilation flow for hybrid CPU-FPGA architectures. *2008 International Conference on Field Programmable Logic and Applications*, Heidelberg, 2008, (pp. 173-178). doi:10.1109/FPL.2008.4629927.

[4] Canis, A., Choi, J., Aldham, M., Zhang, V., Kammoona, A., Czajkowski, T., Brown, S. & Anderson, J. (2013). LegUp: An Open-Source High-Level Synthesis Tool for FPGA-Based Processor/Accelerator Systems. *ACM Transactions on Embedded Computing Systems (TECS), 13*(2). doi:10.1145/2514740.

[5] Thomas, P. (23 January 2008). *470.lbm: SPEC CPU2006 Benchmark Description File.* Retrieved from `https://www.spec.org/cpu2006/Docs/470.lbm.html`

[6] Malcolm, T. (16 August 2011). *436.cactusADM: SPEC CPU2006 Benchmark Description.* Retrieved from `https://www.spec.org/cpu2006/Docs/436.cactusADM.html`

Timeline and To-Do List

- **February 2019:**
  Hand in proposal.
- **March 2019:**
  Start analysis of benchmarks on CPU,
  Start writing background section, for example infos about benchmarks, how HLS works and so on.
- **April 2019:**
  Work on analysis on CPU, find data movement and size
  Start thinking about design on FPGA,
  Write down FPGA design specific info into background section.
- **May 2019:**
  Finish analysis of benchmarks on CPU,
  Write down the findings of the CPU analysis,
  Use HLS to compile kernel into bitstream for FPGA,
  Analyse kernel performance on FPGA,
  move data into different places (Register, BRAM/UltraRAM).
- **June 2019:**
  Start of Lernphase (01 June 2019)
  Finish analysis of benchmarks on FPGA,
  Write down the findings of the FPGA analysis,
  Try running the whole benchmark on CPU and FPGA,
  Port the running program to Enzian and figure out data-transfer between CPU and FPGA.
- **July 2019:**
  Reserve two weeks for studying for exams,
  In the other two weeks, finish Enzian port,
  Write down Enzian related findings.
- **August 2019:**
  Prüfungsession (05 - 30 August 2019) with two core subject exams,
  Reserve two weeks for studying for exams and writing them,
  Get feedback on current thesis and improve it,
  Hand in thesis at 12 August 2019 ($\sim$3 weeks before actual deadline).
- **September 2019:**
  Hold presentation before 11 September 2019 (Notenkonferenz).
- **Write thesis in parallel to analysis and other work.**