Eithne: A framework for benchmarking micro-core accelerators

Maurice Jamieson
EPCC
University of Edinburgh
Edinburgh, UK
maurice.jamieson@ed.ac.uk

Nick Brown
EPCC
University of Edinburgh
Edinburgh, UK
n.brown@epcc.ed.ac.uk

1 INTRODUCTION

The free lunch is over and the HPC community is acutely aware of the challenges that the end of Moore's Law and Dennard scaling [4] impose on the implementation of exascale architectures due to the end of significant generational performance improvements of traditional processor designs, such as x86 [5]. Power consumption and energy efficiency is also a major concern when scaling the core count of traditional CPU designs. Therefore, other technologies need to be investigated, with micro-cores and FPGAs, which are somewhat related, being considered by the community.

Micro-core architectures look to address this issue by implementing a large number of simple cores running in parallel on a single chip and have been used in successful HPC architectures, such as the Sunway SW26010 of the Sunway TaihuLight (#3 June 2019 Top500 [3]) and the 2048 core PEZY-SC2 of the Shoubu system B (#1 June 2019 Green500 [2]). Micro-cores are also being used for new HPC architectures, for example the RISC-V based European Processor Initiative (EPI) [1], is developing a CPU for the next generation of European exascale machines, and their accelerator will use very many simple RISC-V based cores.

FPGAs are another technology being investigated with respect to the performance and energy efficiency challenges that the HPC community will face in the near future. It has already been demonstrated that, by programming the hardware at the electronics level, one can obtain significantly reduced power consumption over CPUs or GPUs. This efficiency advantage has increased the interest in FPGA-based accelerators, particularly re-configurable architectures for HPC [8]. However, a major limitation of FPGAs is their programming challenge - not just the physical effort required but also the bitstream build time, often many hours for non-trivial kernels. Soft-cores, the configuration of an FPGA to appear like traditional CPU core(s), are an interesting alternative, resulting in the development of new accelerators, such as the GRVI Phalanx [6], which combine very many simple soft-cores onto a single chip e.g. micro-cores. The use of soft-cores allows researchers to experiment with CPU core design, for example hardware acceleration for Posits, an alternative to floating point arithmetic, as plug-ins to soft-core(s).

There are hundreds of physical or soft-core micro-core designs, with over 40 implementations of RISC-V alone; the ability to assess competing designs simply and quickly is crucial.

1.1 How to choose a micro-core architecture?

The key micro-core selection criteria are: core performance, power consumption, chip area and code density. The specifics and order of importance of these depend on the exact application but the choices are nuanced, for instance it might not initially be apparent that the first and last criteria are closely linked. An example of this is the benefit of reduced chip resource usage when configuring without hardware floating point support, but there is a 50 times performance impact on LINPACK due to the software emulation library required to perform floating point arithmetic. By understanding the implications of different configuration decisions, the user can make the most appropriate choice, in this case trading off how much floating point arithmetic is in their code vs the saving in chip resource.

Soft-core	MFLOPs/core
MicroBlaze (integer only)	0.120
MicroBlaze (floating point)	5.905

Table 1: LINPACK performance of the Xilinx MicroBlaze on the Zynq-7020 @ 100MHz

Therefore, it is important to consider not only different micro-core instruction set architectures (ISA) but also variants within a particular processor ISA, especially for RISC-V based designs due its very rich micro-architecture eco-system. For instance, when selecting a RISC-V CPU there is a choice between many important aspects such as pipelined vs non-pipelined, superscalar vs non-superscalar, hardware floating point support, large register sets. Without hard numbers to detail the impact of these choices, it is easy to make the wrong decision and doing so in just one of these will have a significant impact.

One approach would be to run a number of the very many currently available common benchmarks on these micro-cores. However, there are a number of architectural features common to micro-cores that makes them significantly different from traditional CPUs and difficult to benchmark:

- Tiny amounts on-chip RAM for code / data (c. 32KB)
- Low-level knowledge specific to each device:
  - C memory map / linker files
  - Core control mechanisms (reset / halt, interrupts)
  - Host / device communications interface
- High communications latency and limited bandwidth
- Complex or no device debugging environment

The result is that running existing HPC benchmarks as-is on micro-cores is at best difficult and most often impossible. In order to compare and contrast the different micro-core options, a benchmark framework is required to abstract much of this complexity.

In this poster and abstract we introduce the Eithne[1] framework that supports comparison of a number of micro-cores. The framework separates the actual benchmark from the details of how this is executed on the different technologies. We then illustrate some sample benchmarking results for the Adapteva Epiphany, two RISC-V

[1] Eithne (/ɛnʲə/ "enya"): Gaelic for "kernel" or "grain".

implementations, ARM, and Xilinx MicroBlaze in terms of performance and power consumption, using the LINPACK benchmark.

2 EITHNE BENCHMARK FRAMEWORK

Eithne supports the benchmarking of micro-core architectures, whether physical chips or soft-cores running on FPGAs, by providing a framework that abstracts over the tricky architectural differences. It enables a single benchmark codebase to be deployed to multiple devices by targeting the required devices at compile time. Running the benchmark suite is as simple as starting execution from the host, with the framework managing all communications / data transfer and kernel execution. The same interaction model is used for all devices to minimise the impact of communications bandwidth and latency across devices.

A wide variety of different host / device links are supported, from on-chip communications for Zynq FPGAs, to on-board communications in the case of the Epiphany, to serial links for embedded devices such as the RM32M1 (see Table 2). Host and device bandwidth can be measured if required. This can be especially useful for instance with soft-cores on FGPAs, one example is the performance difference between having the FPGA fabric (soft-cores) and host ARM CPU cores on the same package, and a board mounting an FPGA with an external link to the CPU.

Eithne currently supports the Xilinx MicroBlaze, PicoRV32 (RISC-V) and VectorBlox Orca (RISC-V) soft-cores, the Adapteva Epiphany III, NXP RV32M1, Cortex-A9 and threads running on the host.

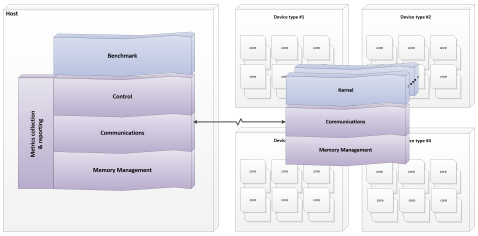

Figure 1: Eithne framework architecture

2.1 Framework extensibility

Whilst a number of different benchmarks, communication mechanisms and micro-core devices are currently supported, users will want to use Eithne to benchmark future devices or kernels and integrate these themselves. As such, we developed the benchmark with extensibility in mind and provide it as a modular framework. Figure 1 illustrates the framework architecture, where Eithne is provided as a stack of functionality; to extend with new functionality, such as a new benchmark, device or communication mechanism, only that level of the framework need be replaced, with all other levels running unchanged. For example, the benchmark codes on the host, which call framework APIs to allocate buffer space etc., can select different devices through the same API calls with no code change.

The kernel running on each device is typically written in C and compiled for that device. Whilst moving a kernel from one device to another is often fairly trivial, again due to the stack of functionality and standard API, the user might want to specialise that, either at the code level or compiler options for the specific device which the framework supports.

3 SAMPLE RESULTS

Whilst the intention of this extended abstract and poster is to introduce the Eithne framework, we provide a sample set of results here from the framework to illustrate some example metrics that can be generated. For example, single-core power consumption vs. performance may be of particular interest but Eithne provides APIs to support multi-core devices. Crucially, can add other metrics but even with the simple run presented here, one can see the benefits of a framework that supports a number of different micro-cores and communication links.

Table 2 outlines the performance and power consumption for the currently supported set of micro-cores running the single-precision LINPACK benchmark. It should be noted that the figures are for software floating point versions of the cores and there are a number of other parameters, including pipeline depth, that can impact the results. All soft-cores are running on an Zynq-7020 at 100MHz with 8 cores. The MicroBlaze and Orca have been configured with an 5 stage pipeline and the PicoRV32 has no pipleline. Even with these simple results, one can see the performance difference between soft-cores and physical chips, along with the impact of different ISAs and configuration options.

Technology	MFLOPs	Watts	KB
PicoRV32 (soft-core)	0.32	0.18	8.92
Orca (soft-core)	0.32	0.11	47.26
MicroBlaze (soft-core)	0.96	0.19	78.93
MicroBlaze + FPU (soft-core)	47.20	0.18	73.30
Epiphany III	1508.16	0.90	8.27
Cortex-A9	33.20	0.60	13.40
RV32M1	27.02	0.19	36.57

Table 2: Performance, power consumption and kernel size for LINPACK benchmark.

4 CONCLUSIONS AND FUTURE WORK

The purpose of the Eithne framework is to reduce the time taken to explore, compare and contrast the different micro-core technologies that are currently, and will be in the future, available to the HPC community. The nature of this technology means that the wealth of benchmarks that are available for traditional HPC machines, such as HPL, NPB, HPCG, will simply not run as-is on these cores. As such, a plug-in architecture which separates the behaviour of the benchmark from the complexities of how it is launched, data transferred and timed, for each target device is highly useful. The plug-in architecture has allowed the comparison between hard and soft-cores where the communications interface is significantly different (shared memory vs. UART), and to very quickly add new processor ISAs where the underlying hardware is already supported by the framework.

Even these initial results highlight the benefit of hardware floating point support and a pipelined architecture for performance, with a minimal impact on FPGA resources.

An alternative to floating point emulation is to use other approaches such as fixed point arithmetic [7] or relax the accuracy of calculation [5], the Eithne benchmark supports future exploration of architectures and codes making this choice. Future work includes implementing additional benchmarks, adding support for additional RISC-V soft-cores e.g. RI5CY, SweRV, implementing kernels using OpenMP and adding support for MPI-based communications.

REFERENCES

[1] 2019. European Processor Initiative Accelerator. https://www.european-processor-initiative.eu/accelerator/

[2] 2019. June 2019 | GREEN500 Supercomputer Sites. https://www.top500.org/green500/lists/2019/06/

[3] 2019. June 2019 | TOP500 Supercomputer Sites. https://www.top500.org/lists/2019/06/

[4] R. H. Dennard, F. H. Gaensslen, V. L. Rideout, E. Bassous, and A. R. LeBlanc. 1974. Design of Ion-Implanted MOSFET's with Very Small Physical Dimensions. 9, 5 (1974), 256–268. https://doi.org/10.1109/JSSC.1974.1050511

[5] Hadi Esmaeilzadeh, Emily Blem, Renée St. Amant, Karthikeyan Sankaralingam, and Doug Burger. 2013. Power Challenges May End the Multicore Era. 56, 2 (2013), 93–102. https://doi.org/10.1145/2408776.2408797

[6] Jan Gray. 2016. GRVI Phalanx: A Massively Parallel RISC-V FPGA Accelerator Accelerator. In *Field-Programmable Custom Computing Machines (FCCM), 2016 IEEE 24th Annual International Symposium On* (2016). IEEE, 17–20.

[7] Cherry Hinton and Cambridge CB. 1996. Document Number: ARM DAI 0033A: Application Note 33 Fixed Point Arithmetic on the ARM.

[8] Martin Kaiser, René Griessl, and Jens Hagemeyer. 2017. A Reconfigurable Heterogeneous Microserver Architecture for Energy-Efficient Computing. (2017), 2.