RISC-V based GPGPU on FPGA: A Competitive Approach for Scientific Computing ?

Eric Guthmuller[1] and Jérôme Fereyre[1] *

[1]Univ. Grenoble Alpes, CEA, List,
F-38000 Grenoble, France

Abstract

FPGA architectures include increasingly complex arithmetic operators and optimized hard IPs, such as memory subsystems and Networks-on-Chip (NoC). This evolution leads to higher compute density also linked with high memory bandwidth. It represents an opportunity to tailor an architecture to niche application needs while being competitive with a costly ASIC implementation. More specifically, scientific computing requires high precision (> 32 bits) floating point computation. However, GPU vendors are progressively favoring low precision performance for AI needs, and are even phasing out support for 64-bit floating point compute. We present an analytical study motivating the need to investigate the implementation of an open source 64-bit GPGPU architecture on a state of the art FPGA, as an alternative to GPUs for scientific computing.

Introduction

Scientific computing relies on linear algebra kernels that are sensitive to compute precision. Support for 64-bit floating precision precision is a requirement. At the same time, Artificial Intelligence (AI) targets much lower precision, down to single bit weights. Processors and accelerators are thus increasingly targeting high performance at low computing precision, to the detriment of high precision operations.

At the same time, FPGA architectures are evolving as feature-rich System-on-Chips (SoCs). They include complex arithmetic blocks called Digital Signal Processors (DSPs) which can compactly implement floating point operators. They also embed hard IPs: memory controllers (DDR or HBM), application processors, PCIe controllers, and NoCs, for example. The computing density and memory bandwidth now available to FPGA designers make implementing a computing accelerator a possibility worth investigating. Real scientific use cases are also often limited by memory bandwidth rather than arithmetic throughput as they work on sparse data. The integration of HBM memories in some FPGAs is thus an opportunity for FPGA architectures to be competitive with ASICs.

Vortex [1], was first introduced in 2021, as a fully fledged open source GPGPU with several FPGA reference implementations. Vortex is based on the RISC-V ISA extended with custom instructions to handle the Single-Instruction-Multiple-Threads (SIMT) programming model of GPGPUs. It is still actively developed and supports the OpenCL API through the open source PoCL [2] framework. The software stack includes a modified LLVM with support for Vortex extended ISA.

*Corresponding author: eric.guthmuller@cea.fr

We investigate the suitability of using Vortex as a foundation for high performance FPGA-based GPGPU targeting scientific computing kernels. We target the AMD Alveo V80 [3] board as it integrates the largest FPGA in AMD's Versal HBM series. It features 32 GB of HBM (819 GB/s) and 32 GB of DDR4 (25 GB/s). Details about the AMD Versal XCV80 HBM Series FPGA are given in Table 1.

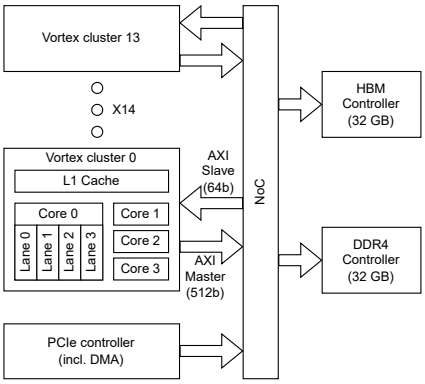

Figure 1: *Integration of Vortex clusters in XCV80 FPGA*

Analytical Study

We will first estimate memory bandwidth per operation of some linear algebra kernels used in scientific computing. We then present our FPGA board target before stating our performance objectives to be relevant for our use cases.

As a typical scientific computing application, the Conjugate Gradient (CG) iterative solver can be decomposed into successive linear algebra kernels, such as matrix-vector multiplication or vector dot product, for example. In fact, CG uses only matrix-vector and

Table 1: *FPGA Implementation Results*

Configuration	Total LUTs	FFs	RAMB36	RAMB18	URAM	DSP Blocks	Frequency (MHz)	Peak Gflops (FP32)
4 cores	132,397 (5.14%)	148,541 (2.89%)	166 (4.44%)	10 (0.13%)	0	32 (0.29%)	300	9.60
56 cores	1,804,955 (70.12%)	1,999,442 (38.84%)	2,324 (62.12%)	140 (1.87%)	0	448 (4.13%)	282	126.36
AMD Versal HBM XCV80	2,574,208	-	132 Mb		541 Mb	10,848	≈ 800	≈ 17,500

vector-vector kernels that exhibit a low arithmetic density. Moreover, in most cases the matrix is sparse putting even higher pressure to memory.

In all cases, compute performance is generally much lower than theoretical peak GFLOPS and the upper limit can be computed using the roofline model [4]. Authors of [5] show that FP32 Sparse Matrix-Vector multiplication (SpMV) running on the Nvidia P100 GPU has an upper bound of around 170 GFLOPS (limited by memory bandwidth), far from the P100's 10 TFLOPS peak FP32 performance. Variability is also high, with some matrices tested achieving lower than 10 GFLOPS.

Designing an architecture always requires balancing compute throughput with memory bandwidth. In our case, memory bandwidth is given by the FPGA itself, so we wish to determine the minimum compute capacity required to fully use the given memory bandwidth. We computed maximum theoretical throughput given by the roofline model computed for vector-vector operations and given by [5] for SpMV with N=1,000,000 and nnz=100,000,000 (CSR format, 64-bit matrix elements and 32-bit column indexes are assumed) : DOT at 205 GFLOPS, AXPY at 68, SCAL at 102 GFLOPS and SpMV at 135 GFLOPS. This model assumes 100% memory efficiency, so real figures would be lower (a 70% memory bandwidth utilization is generally considered quite good). It shows that a performance target between 100 and 200 GFLOPS is a realistic range and should provide performance competitive with GPGPUs for this kind of applications.

Early Results

Figure 1 shows the overall architecture where we integrate a varying number of 4-lanes 4-threads 4-core Vortex clusters, implementing the extended RV64IMAF RISC-V ISA. Each cluster implements a shared 16 kB L1 I cache and 16 kB L1 D cache. Instead of using the provided hierarchical Vortex organization, we connected these 4-core clusters directly to the FPGA NoC through a 512-bit AXI master port for memory accesses and a 64-bit AXI slave port for configuration. The 512-bit bandwidth is sufficient to feed the four cores.

Table 1 shows different implementation results with 1 to 14 clusters synthesized on the AMD Versal HBM XCV80 and a target frequency of 300 MHz. DSP usage is low, so there is room to use more of them for higher precision floating point operators. Finally, as our configuration does not include L2 or L3 caches, no Ultra RAM (URAM) is used, leaving a significant amount of embedded memory available for future use.

Conclusion

As FPGAs are morphing into fully fledged System-on-Chips with high performance integrated IPs and memory controllers, reaching competitive computing performance is now very much possible. As scientific computing applications are memory intensive, integration of HBM in FPGAs is an opportunity to provide an alternative to GPGPUs that are focusing on AI workloads. Using the roofline mode, we show analytically that such an approach is an avenue worth to pursue. We show early results of Vortex open source GPGPU implementation on AMD Versal XCV80 HBM Series with up to 56 cores and 126 GFLOPS peak FP32 throughput.

References

[1] Blaise Tine et al. "Vortex: Extending the RISC-V ISA for GPGPU and 3D-Graphics". In: *MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture*. MICRO '21. Virtual Event, Greece: Association for Computing Machinery, 2021, pp. 754–766. ISBN: 9781450385572. DOI: 10.1145/3466752.3480128.

[2] Pekka Jääskeläinen et al. "pocl: A Performance-Portable OpenCL Implementation". In: *Int. J. Parallel Program.* 43.5 (Oct. 2015), pp. 752–785. ISSN: 0885-7458. DOI: 10.1007/s10766-014-0320-y.

[3] *AMD Alveo V80 product page*. https://www.amd.com/en/products/accelerators/alveo/v80.html.

[4] Samuel Williams, Andrew Waterman, and David Patterson. "Roofline: an insightful visual performance model for multicore architectures". In: *Commun. ACM* 52.4 (Apr. 2009), pp. 65–76. ISSN: 0001-0782. DOI: 10.1145/1498765.1498785.

[5] Changwan Hong et al. "Efficient sparse-matrix multi-vector product on GPUs". In: *Proceedings of the 27th International Symposium on High-Performance Parallel and Distributed Computing*. HPDC '18. Tempe, Arizona: Association for Computing Machinery, 2018, pp. 66–79. ISBN: 9781450357852. DOI: 10.1145/3208040.3208062.