Harnessing Hardware Acceleration with RISC-V and the EU Processor

Juan Fumero¹, Athanasios Stratikopoulos¹, Mehdi Goli², Ruymán Reyes², Konstantinos Nikas³, Dionisios Pnevmatikatos³, Nectarios Koziris³ and Christos Kotselidis³

¹Department of Computer Science, The University of Manchester, UK
²Codeplay Ltd, UK
³ICCS, National Technical University of Athens, Greece

AERO: Accelerated European Cloud
An open-source software ecosystem for the EPI hardware

Enriching the software ecosystem for Cloud deployment

ARM/RISC-V based EU hardware designs are being developed

How to efficiently use it from high level programming languages?

AERO Hardware/Software Stack

Enabling Language Frontends
- TornadoVM: a Java parallel programming framework and a JVM plugin for transparent hardware acceleration on multi-core CPU, GPUs and FPGAs.

Enabling RISC-V Backends
- ComputeAorta, from Codeplay, enables implementation of open standards such as OpenCL. It includes tooling to convert OpenCL C and SPIR-V into target ISA using existing LLVM backends.

Enabling Vectorization for RISC-V
- Data parallel programs written in Java with TornadoVM can be accelerated using ComputeAorta’s vector units via RISC-V RVV ISA instructions generated from Java scalar code.

Example of leveraging the AERO Stack:
Accelerate Java/Cloud workloads on RISC-V

Input: Java Sequential Code

Compiler IR

Drivers

Managed Programming Languages
- OpenJDK, GraalVM, TornadoVM, Quarkus

Native Programming Languages
- Open-source heterogeneous programming languages & runtimes (SYCL, OpenCL, DPC++/OneAPI)

Runtimes (JVM, C++, SYCL, CUDA, etc.)

Orchestration (Kubernetes, Maestro)

Virtualization (KVM, Docker, rust-vmm, Firecracker)

Operating Systems (Red Hat Linux, Suse Linux)

Digital Twins IoT/Cloud

HPC/Cloud Database Acceleration

OS, drivers & virtualization
- Optimized Linux distribution
- Docker, KVM (targeting CPU & RISC-V coprocessors)

HW acceleration
- Leverage HW components of Rhea for performance & security

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the HaDEA. Neither the European Union nor the granting authority can be held responsible for them. Project number: 101092850. In addition, this work is funded by UK Research and Innovation (UKRI) under the UK government’s Horizon Europe funding guarantee for grant numbers 10048318 and 10048915.