

MSc Topics in the Customized Parallel Computing group



[0922-1] Offloading Low Latency Control Loop Decisions from Nano-drones

Background: Nano-drones can be used in indoor facilities to perform important missions, such as search and rescue surveillance. Their small physical form factor is ideal for navigating small indoor compartments safely. However, running complex algorithms such as neural network inference onboard is challenging due to their limited computational resources, weight, and battery capacity. A potential solution is offloading the heavy computation to a more powerful server over the wireless network.

Project: The task for the student is to implement neural network inference such as in [1] on an AI deck [2] of the Crazyflie nano-drone [3]. Computation would be performed with OpenCL and offloaded to a remote GPU server using [4]. The key research question is if we can reduce the impact of the wireless network to the latency of the control decisions by utilizing low-latency image compression to reduce the size of the camera images sent to the remote server.

Basis: The starting point of the project is a proof-of-concept drone firmware that can offload OpenCL commands to a server and use the results to control the drone's hovering altitude and orientation.

Keywords: nano-drones, AI, neural networks, GPU computing, edge computing, embedded development

Material/Links

[1]<https://arxiv.org/abs/2103.10873>

[2]<https://store.bitcraze.io/collections/decks/products/ai-deck-1-1>

[3]<https://www.bitcraze.io/products/crazyflie-2-1>

[4]http://dx.doi.org/10.1007/978-3-031-04580-6_6

Supervisors and Advisors

Jakub Žádník, Jan Solanti, Pekka Jääskeläinen

Why Work in a Research Group?

- Paid research work for thesis
- One month of paid **writing** time*
- Part-time option for more flexibility in combining work and studying
- Contracts generally 6-12 months, extending is possible

* If the material is reusable in research papers or project reports



[0922-2] Automatic Networked Resource Discovery for Compute Roaming

Background: Offloading computation from a low-power client device to another machine across network has many benefits simply from additional performance to using less energy locally on a battery-powered client device. *Edge computing* is a special case of this, where the computer doing the actual work is located physically close both to the source of the processed data and to a network access point to keep network latency to a minimum. However, when the edge computing device is *mobile*, the client device may *roam* from one access point to another, necessitating automatic discovery and use of compute resources close to a nearby access point for a form of opportunistic cycle scavenging from the powerful network-accessible compute units.

Project: The task for the student would be to implement automatic discovery of compute servers on the local network in PoCL-R [1] and demonstrate it with a real-world *compute roaming* setup.

Basis: PoCL-R as described in [1] already provides basic OpenCL cross-network offloading capabilities, but it requires applications to explicitly define the remote devices to be used at initialization time and lacks the capability to alter the set of devices at runtime.

Keywords: networking, edge computing, service discovery, cycle scavenging

Material/Links

[1]http://dx.doi.org/10.1007/978-3-031-04580-6_6

Supervisors and Advisors

Jan Solanti, Pekka Jääskeläinen

Interested?

Send a freeform application by e-mail to Pekka Jääskeläinen pekka.jaaskelainen@tuni.fi.

Include at least these with your application:

- Study record
- CV
- Links to previous projects, if publically available

Positions will be filled as soon as suitable applicants are found!



tuni.fi/cpc

MSc Topics in the Customized Parallel Computing group

Why Work in a Research Group?

- Paid research work for thesis
- One month of paid **writing** time*
- Part-time option for more flexibility in combining work and studying
- Contracts generally 6-12 months, extending is possible

* If the material is reusable in research papers or project reports



[0922-5] Porting a Vendor-independent OpenCL Implementation to Intel FPGAs

Background: Currently, programming heterogeneous systems requires the use of vendor-specific tools and languages. Open standards such as OpenCL provide a standardized way for creating software libraries and applications that are portable to any hardware platforms claiming support for OpenCL. However, the OpenCL implementations for FPGAs provided by the two major FPGA manufactures (Intel/Altera and AMD/Xilinx) are non-conformant “vendor islands”, where the source code of the program still needs to be modified and thus the cross-device functionality enabled by OpenCL is not properly realized.

To tackle this, in previous work [2] we augmented an open-source multi-vendor implementation PoCL [1] with support for FPGA devices. This work proposed a unified interface which enables implementing the underlying computation kernel with multiple different techniques including RTL, soft processors, and proprietary High-Level Synthesis (HLS) tools. While our methodology described in the published work [2] was claimed to be vendor-independent, it is **not so far ported and evaluated on Intel FPGAs**.

Project: The task for the student would be to port our open-source FPGA OpenCL implementation to Intel FPGA devices. Additionally, there are interesting future avenues to continue the work towards efficient data movement, multi-device systems and partial reconfiguration, depending on the pace of the student.

Basis: There already exists an implementation of the framework targeted at Xilinx FPGA devices which will be used and adapted for Intel FPGA platforms.

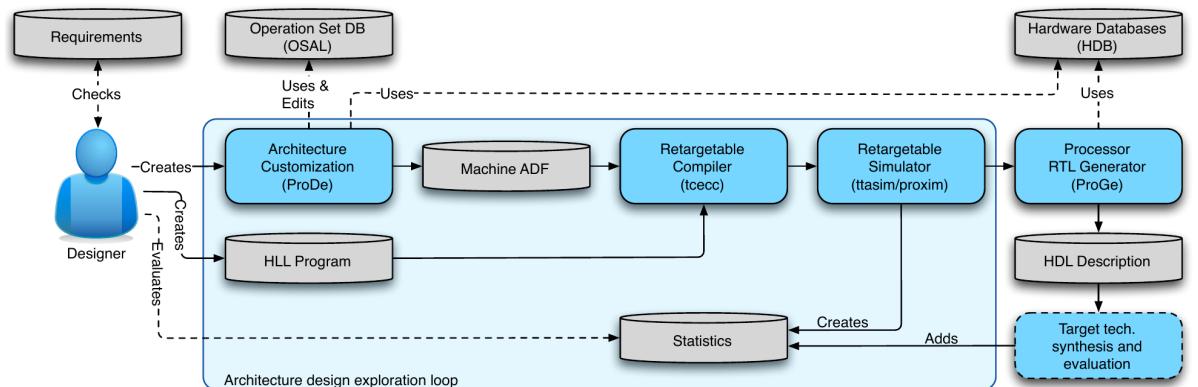
Keywords: FPGA, OpenCL

Material/Links

- [1] <http://portablecl.org/>
 [2] <https://urn.fi/URN:NBN:fi:tuni-202111298778>

Supervisors and Advisors

Topi Leppänen, Pekka Jääskeläinen



[0922-3] Standards Conformance Validation for an Open Source OpenCL Implementation

Background: Multi-vendor programming standards such as OpenCL greatly simplify interoperability between systems and adapting devices and software to new environments, as well as helps constructing new environments such that as many existing tools and applications can be reused in them as possible. This requires that the users of the interface can trust that various implementations of a standard interface behave the same way (at least for the most crucial aspects). Standards conformance test suites are created as an indicator for users and implementors to see whether a given implementation of a programming standard meets the requirements and expectations laid out in the specification.

Project: The task for the student is to develop, validate and report the conformance of PoCL [1] with regards to the OpenCL 3.0 specification [2]. The primary platform to make conformant is a state-of-the-art ARM based CPU device, which would make PoCL the first conformant ARM CPU OpenCL 3.0 implementation. The second goal would be to make RISC-V CPUs similarly conformant. Third goal would be to make an AMD CPU using the x86 instruction-set architecture pass the conformance tests. However, passing the conformance on a subset of the targets suffices for a master's thesis in case unforeseen challenging technical problems appear.

Basis: The ARM CPU port works relatively well, there are some missing functions in the bitcode library (we could reuse libclc or ocml). The conformance test suite of Khronos should pass for Intel x86 in the conformance mode with the minimal OpenCL 3.0 feature set, which means AMD CPUs should work as well. The RISC-V port is likely the most technically challenging one; there was initial effort from RISC-V community long time ago, but it disappeared, with some spurious publications popping every now and then where PoCL is used on RISC-V (but not claiming OpenCL compliance).

Keywords: standards, testing, validation, OpenCL

Material/Links

- [1] <http://portablecl.org/>
 [2] <https://www.khronos.org/opencl/>

Supervisors and Advisors

Jan Solanti, Pekka Jääskeläinen

[0928-4] Integration of System-level Simulation to a Processor Customization Toolset

Background: Designing customized hardware has become increasingly popular in the recent years due to declining performance and energy efficiency improvements acquired from newer process technologies. While customized hardware can achieve both better energy efficiency and performance, proper tooling is required to simulate, program and generate customized hardware efficiently in order to minimize engineering time of customized processors. In addition to designing the customized processor IP itself, attention must be paid to the system level integration and to its effects on the performance and energy efficiency of the computing platform.

Project: The task for the student would be to integrate a popular system-level simulator gem5 [1] to the OpenASIP toolset [2] so processors designed with OpenASIP can be easily tested in systems with various data memory hierarchies and other processors. The goal is to be able to generate performance and possibly energy consumption estimates from the system-level components such as complex cache hierarchies.

Basis: Currently, the OpenASIP toolset offers an instruction set simulator that can be used for functional verification, as well as to produce cycle-accurate profiling data of program runs. In addition, the simulator can be hooked into SystemC system-level simulations, thus there is already a proof-of-concept simulation integration case.

Keywords: system design, computer architecture, hw/sw co-design, ASIP

Material/Links

- [1] <https://www.gem5.org/>
 [2] <http://openasip.org>

Supervisors and Advisors

Topi Leppänen, Kari Hepola, Joonas Multanen, Pekka Jääskeläinen

Interested?

Send a freeform application by e-mail to Pekka Jääskeläinen pekka.jaaskelainen@tuni.fi. Include at least these with your application:

- Study record
- CV
- Links to previous projects, if publically available

Positions will be filled as soon as suitable applicants are found!

