

APPENDIX

ARTIFACT DESCRIPTION APPENDIX: OPENACC TO FPGA: A DIRECTIVE-BASED HIGH-LEVEL PROGRAMMING FRAMEWORK FOR HIGH-PERFORMANCE RECONFIGURABLE COMPUTING

A. Abstract

This artifact describes the hardware devices and configurations, software environment, computational benchmarks, and performance experiments used in the evaluations of the optimizations developed for the OpenACC to FPGA framework. These evaluations were used to generate the figures in the bottom-right panel of the poster.

B. Description

B1. Check-list (artifact meta information):

- **Algorithm:** Benchmarks from the Rodinia Benchmark Suite 3.1, and the Intel FPGA OpenCL High-Performance Computing Platform Examples
- **Compilation:**
 - **OpenACC Front-end:** OpenARC V0.11 (openarc)
 - **OpenACC Back-end FPGA:** Intel FPGA SDK for OpenCL V16.1.0 (aocl)
 - **OpenACC Back-end GPU:** CUDA V8.0 (nvcc)
 - **OpenMP CPU:** GCC 4.8.5, `-O2` flag, 32 OpenMP Threads (g++)
- **Data set:** Data provided as part of benchmarks or generated during benchmark initialization phase
- **Run-time environment:** Red Hat 4.4.7-18 (FPGA, GPU experiments), Red Hat 4.8.5-16 (CPU experiments)
- **Hardware:**
 - **FPGA Experiments:** Stratix V 5SGSD5 FPGA (172K ALMs, 690K registers, 2014 M20K blocks, 1590 DSP blocks, 8GB DDR3 device memory), and Intel (R) Xeon (R) E5520 CPU (2.27 GHz)
 - **GPU Experiments:** NVIDIA Tesla K40c GPU and Intel (R) Xeon (R) E5520 CPU (2.27 GHz)
 - **CPU Experiments:** 16-core Intel(R) Xeon(R) E5-2683 v4 CPU (2-way hardware multithreading)
- **Execution:**
 - **Runtime Measurements:** C++ API calls (`clock_gettime()`)
 - **FPGA Power Measurements:** Quartus Prime PowerPlay Power Analyzer (2.34 W added to power estimations to account for FPGA memory modules)
 - **GPU Power Measurements:** NVIDIA NVML Library (`nvmlDeviceGetPowerUsage()`)
 - **CPU Power Measurements:** Intel Running Average Power Limit (RAPL) interface
- **Publicly available?:** No; access can be granted per request.

B2. How software can be obtained (if available):

```
git clone https://gitlab.com/lamb/ACC2FPGA.git
```

B3. Hardware dependencies:

- **FPGA experiments:** Our framework can be evaluated on any FPGA supported by the Intel OpenCL FPGA SDK. The presented results were performed on a Stratix V FPGA, but we have also successfully tested using an Arria 10 FPGA. The FPGA used can be specified in the appropriate Makefile.
- **GPU experiments:** GPU comparisons can be performed on any GPU supported by OpenARC, which includes both CUDA-compatible and OpenCL-compatible GPUs.
- **CPU experiments:** CPU comparisons can be performed on any OpenMP-compatible CPU.

B4. Software dependencies: The OpenACC compilations depend on the OpenARC v0.11 compiler, described here:

<http://ft.ornl.gov/research/openarc>

OpenARC is still undergoing development, but access can be granted from the poster authors by request.

OpenACC FPGA compilations require the backend Intel FPGA SDK for OpenCL.

<https://www.intel.com/content/www/us/en/software/programmable/sdk-for-opencl/overview.html>

The OpenACC GPU comparisons were tested using the CUDA 8.0 installation, available from the NVIDIA website, although an OpenCL to GPU compiler could also be used for GPUs that support OpenCL.

The OpenMP CPU comparisons were tested using GCC 4.8.5, although other OpenMP compilers could also be used for comparisons.

B5. Benchmarks and Datasets: We used benchmarks from both the *Intel FPGA SDK for OpenCL High Performance Design Examples* (Sobel and FD3D) and the *Rodinia Benchmark Suite v3.1* (HotSpot, SRAD, and NW) in our evaluations:

<https://www.intel.com/content/www/us/en/programmable/products/design-software/embedded-software-developers/opencl/developer-zone.html#design-examples>

http://lava.cs.virginia.edu/Rodinia/download_links.htm

OpenACC implementations of the Rodinia v1.0 benchmarks, available in the examples directory of the OpenARC repository, were updated with changes introduced in Rodinia v3.1. This allowed for a more fair comparison with the FPGA-optimized OpenCL implementations of the Rodinia v3.1 benchmarks developed by H. R. Zohouri et. al:

<https://ieeexplore.ieee.org/abstract/document/7877113>

For the CPU benchmark experiments, we use the OpenMP implementations from the benchmark suite without modification. Test data sets are provided with the benchmarks.

For the Intel benchmarks, OpenACC and OpenMP implementations were done by removing OpenCL kernels and logic, and replacing them with serial logic and the appropriate directives. For FD3D, test data is generated as part of the benchmark initialization phase. For Sobel, a test image is provided, although any 8-bit PPM image can be used.

C. Installation

OpenARC, the Intel FPGA SDK for OpenCL, CUDA, and GCC can be installed according to the instructions in the corresponding documentation. OpenARC must be installed with FPGA support, which can be enabled by setting the appropriate flags in the provided `openarc/make.header` file. No additional software installation is required.

D. Experiment workflow

We do not currently have scripts to automate experiments. However, experiments can be manually verified by compiling individual examples and comparing results.

- **FPGA experiments:** Each FPGA experiment can be performed by navigating to the appropriate directory, `ACC2FPGA/openacc/fpga/benchmark`, and building the example. OpenACC programs can be compiled by the following:

```
cd benchmark_directory
./02Build.script
make
```

To recreate the sliding window experiment, the window optimization can be enabled or disabled by simply commenting out the `#openarc` transform window directives in the `benchmark/src/kernel.c` file. The OpenCL FPGA directories used for baseline comparisons can be found in `ACC2FPGA/opencl`. Each directory contains a README with compilation instructions.

- **CPU and GPU experiments:** Similarly, GPU and CPU experiments can be run by navigating to the appropriate directory, `ACC2FPGA/openacc/gpu/benchmark` or `ACC2FPGA/openmp/benchmark`. The OpenACC GPU benchmarks can be compiled as noted above, and the OpenMP benchmarks directories contain either a *compile* file with the compilation command or a *Makefile*.

E. Evaluation and expected result

E1. Runtimes: Each benchmark source code currently has embedded timers, built on the `clock_gettime()` function of *time.h*. These timers are designed to record execution time of offloaded code, and to omit initialization and memory transfer times.

E2. Power: For FPGA power measurements, we use the Quartus PowerPlay Power Analyzer application, which comes bundled with the Intel FPGA SDK for OpenCL. To get FPGA power estimates, first fully compile the target benchmark with OpenARC (OpenACC) or AOCL (OpenCL), and then run the PowerPlay Analyzer. For example, for OpenACC benchmarks perform the following after compiling:

```
cd cetus_output/openarc_kernel
quartus_pow top.qpf
vim top.pow.summary
```

For GPU power measurements, we use the NVML-power API for measuring average power that serves as a wrapper API to the NVIDIA NVML library:

<https://github.com/kajalv/nvml-power>

For CPU power measurements, we query the Intel Running Average Power Limit (RAPL) interface for CPU Power usage. See `ACC2FPGA/openmp/energy.c` for an example of the implemented function calls.

F. Experiment customization

The experiments can be customized in several ways. For the OpenACC FPGA benchmarks, different unroll factors can be used, and optimizations can be enabled or disabled by modifying the applied directives. Also different input sizes can be tested for most benchmarks by providing different input files and command line arguments. Similarly, the number of iterations for the iterative algorithms can be modified via command line arguments.

G. Notes

Please feel free to email the authors with any problems or questions regarding this Artifact Description or the experimental process.