Ref. Ares(2019)4139823 - 30/06/2019

D5.1

Heterogeneous Execution Prototype v1

The University of Manchester

D5.1

Heterogeneous Execution Prototype v0.1

DOCUMENT IDENTIFIER:	D5.1
DUE DATE:	30/06/2019
DELIVERY DATE:	30/06/2019
CLASSIFICATION:	Public
EDITORS:	Christos Kotselidis
DOCUMENT VERSION:	1.0

CONTRACT START DATE:	01/01/2018
CONTRACT DURATION:	36 months

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

(Page intentionally blank)

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

CONTRIBUTORS	
Name	**Organisation**
Christos Kotselidis	The University of Manchester

PEER REVIEWERS	
Name	**Organisation**
Christos Tsalidis	NEUROCOM LUXEMBOURG
Vassilis Spitadakis	NEUROCOM LUXEMBOURG

REVISION HISTORY		
Version	**Date**	**Modifications**
0.1	15/06/2019	Initial version
0.2	24/06/2019	Incorporated feedback from NCOM
0.3	26/06/2019	Added roadmap for v0.3 release
1.0	30/06/2019	Final revision after reviews

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

Executive Summary

This document describes the Prototype v1.0 of the Heterogeneous Execution framework of E2Data; namely, TornadoVM.

The source code of TornadoVM can be found here: https://github.com/beehive-lab/TornadoVM

Alternatively, a forked (mirror) repository can be found here: https://github.com/E2Data/TornadoVM

In addition, this deliverable highlights the software components developed up to M18 of the duration of the E2Data project that corresponds to v0.2 of TornadoVM.

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

Table of Contents

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

1. Introduction

TornadoVM is the heterogeneous execution platform of E2Data that can accelerate arbitrary Java code on OpenCL compatible devices such as multicore CPUs, GPUs, and FPGAs.

TornadoVM follows a multi-layer software architecture with its various layers orchestrating different parts of the execution. Ongoing publications [1-7] describe in detail each particular component of the framework. In a nutshell, the main components of TornadoVM that will be briefly discussed in this deliverable are:

1. Tornado API
2. Tornado Runtime
3. Tornado Drivers
4. Tornado Assembly
5. Tornado Benchmark and Unit test Suites

The license model of each component can be found here: https://github.com/beehive-lab/TornadoVM#license

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

2. TornadoVM

The following subsections outline the current state of the framework which corresponds to v0.2. For completeness, the release notes for both releases (v0.1 and v0.2) can be found here:

- TornadoVM v0.1: https://github.com/beehive-lab/TornadoVM/blob/master/CHANGELOG.md#tornado-010
- TornadoVM v0.2: https://github.com/beehive-lab/TornadoVM/blob/master/CHANGELOG.md#tornadovm-02

2.1 Current Version: 0.2

As of v0.2 TornadoVM support transparent heterogeneous execution on Intel/AMD CPUs, NVDIA/INTEL/AMD GPUs, and Intel FPGAs. Ongoing work is currently undertaken to provide coverage for ARM Mali GPUs, and Xilinx FPGAs (scheduled for release in v0.3). In the following link we list the currently supported devices and platforms: https://github.com/beehive-lab/TornadoVM/blob/master/INSTALL.md

Regardless of the target device, TornadoVM follows the same execution path through its various components with the exception of the Drivers layer and some specialized compiler optimizations. The specialized optimizations are in the form of optimization phases that are applied to the compiler graph during the compilation process; most notably loop auto-parallelization, automatic reductions, data transfer optimizations, loop unrolling, etc.

2.1.1 TornadoVM API

This component is visible to the developers who wish to use TornadoVM to accelerate their Java applications. An additional package which semantically belongs to the API component but is separated for licensing purposes is the Matrices package. This package contains high-performance Matrix implementations for GPUs custom to TornadoVM.

These packages can be found here:

TornadoVM API : https://github.com/beehive-lab/TornadoVM/tree/master/tornado-api

Matrices: https://github.com/beehive-lab/TornadoVM/tree/master/matrices

Descriptions of how to use the TornadoVM API can be found in the following publications [1-7] as well as the live whitepaper document [8] maintained by the University of Manchester.

2.1.2 TornadoVM Runtime

This component is not visible to the developers and it is part of the core TornadoVM execution engine. The TornadoVM Runtime is responsible for compiling and optimizing Java code for heterogeneous execution, memory management, TornadoVM bytecode generation and execution, scheduling and execution coordination.

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

The root folder of the TornadoVM Runtime can be found here: https://github.com/beehive-lab/TornadoVM/tree/master/runtime

Descriptions of how to use the TornadoVM Runtime can be found in the following publications [1-7].

2.1.3 TornadoVM Drivers

This component includes all the device-specific associated drivers for executing the auto-generated OpenCL code on the heterogeneous devices. It also contains the secondary Drivers-OpenCL-Headers package.

The root folder of the TornadoVM Drivers can be found here: https://github.com/beehive-lab/TornadoVM/tree/master/drivers

2.1.4 TornadoVM Assembly

This package contains auxiliary helper functions (scripts, documentation, etc.) for building and running TornadoVM.

The root folder of the TornadoVM Assembly can be found here: https://github.com/beehive-lab/TornadoVM/tree/master/assembly/src

2.1.5 TornadoVM Benchmark and Unit Test Suites

This component is a collection of different packages for unit tests, benchmarks, and examples for utilizing TornadoVM assessing its performance and functional correctness. These subcomponents are part of the continues integration framework UNIMAN uses for guaranteeing functional and performance correctness of incoming changesets.

The root folder of the TornadoVM Benchmark and Unit Test Suites can be found here:

https://github.com/beehive-lab/TornadoVM/tree/master/benchmarks

https://github.com/beehive-lab/TornadoVM/tree/master/examples

https://github.com/beehive-lab/TornadoVM/tree/master/unittests

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

3. Additional software components

TornadoVM-Docker repository: https://github.com/beehive-lab/docker-tornado

KFusion-TornadoVM repository: https://github.com/beehive-lab/kfusion-tornadovm

4. Installation and usage instructions

To install TornadoVM you can follow one of the following procedures:

1. **Installation through Docker image**
   If the workstation upon which TornadoVM will be installed has an NVDIA GPU you can use the provided docker image following the instruction here: https://github.com/beehive-lab/docker-tornado

2. **Manual built and installation:**
   If you wish TornadoVM from scratch please follow the steps listed here: https://github.com/beehive-lab/TornadoVM/blob/master/INSTALL.m

After successful installation of TornadoVM, you can run all unit tests by issuing:

```
$ tornado-tests.py -V
```

To start using TornadoVM to accelerate your Java applications please follow the instructions included in the TornadoVM whitepaper here: https://www.dropbox.com/s/rbb2qv0q2wicgvy/main.pdf

Finally, you can refer to the following screencast explaining how to utilize FPGA acceleration through TornadoVM and the IntelliJ IDE here:
https://www.youtube.com/watch?v=lYtpi7zcp1g&feature=YouTube

5. TornadoVM roadmap

TornadoVM v0.3 release is currently scheduled for 07/2019.

Features for this release include:

- Implementation of fine-grain performance profiler
- Advanced reduction support for GPUs and FPGAs
- Implementation of large data processing on GPUs (data that exceed GPUs' physical memory)
- Initial support for Xilinx FPGAs
- Stability and performance fixes

Currently these features are in the code review and regression testing phase.

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

Conclusions

This short deliverable lists the current software components of TornadoVM v0.2 as well as the features scheduled for the next release in M19 of the project.

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov

Bibliography

[1] Juan Fumero, Michail Papadimitriou, Foivos S. Zakkak, Maria Xekalaki, James Clarkson, Christos Kotselidis. ***Dynamic Application Reconfiguration on Heterogeneous Hardware.*** In 15th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE), 2019.

[2] James Clarkson, Juan Fumero, Michail Papadimitriou, Foivos S. Zakkak, Maria Xekalaki, Christos Kotselidis, and Mikel Luján. ***Exploiting High-Performance Heterogeneous Hardware for Java Programs using Graal.*** In 15th International Conference on Managed Languages and Runtimes (ManLang), 2018.

[3] Juan Fumero, Christos Kotselidis. ***Using Compiler Snippets to Exploit Parallelism on Heterogeneous Hardware: A Java Reduction Case Study.*** In the 10th International Workshop on Virtual Machines and Intermediate Languages (VMIL '18), November 2018.

[4] Maria Xekalaki, Juan Fumero, Christos Kotselidis. ***Challenges and proposals for enabling dynamic heterogeneous execution of Big Data frameworks***. In 1st International Workshop on Next Generation Clouds for Extreme Data Analytics (Xtreme-Cloud), December 2018.

[5] James Clarkson, Juan Fumero, Michail Papadimitriou, Maria Xekalaki, Christos Kotselidis. ***Towards Practical Heterogeneous Virtual Machines***. In MoreVMs 2018 Workshop on Modern Language Runtimes, Ecosystems, and VMs, April 2018.

[6] Christos Kotselidis, James Clarkson, Andrey Rodchenko, Andy Nisbet, John Mawer, Mikel Luján. ***Heterogeneous Managed Runtime Systems: A Computer Vision Case Study*.** In 13th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments (VEE), 2017.

[7] Michail Papadimitriou, Juan Fumero, Athanasios Stratikopoulos, Christos Kotselidis. **Towards prototyping and Acceleration of Java programs onto Intel FPGAs**, In FCCM 2019.

[8] TornadoVM White paper: https://www.dropbox.com/s/rbb2qv0q2wicgvy/main.pdf

Co-funded by the Horizon 2020 Framework Programme of the European Union under Grant Agreement nº 780245.

Partners
EXUS (Coordinator), UNIMAN, ICCS, DFKI, NEUROCOM, KALEAO, CTI DIOPHANTUS, SPARKS, iProov