D5.7 Prototype demonstration of performance analysis tools on a system with multiple ARM boards Version 1.0

Document Information

Contract Number	288777
Project Website	www.montblanc-project.eu
Contractual Deadline	M24
Dissemination Level	PU
Nature	Prototype
Author	Marc Schlütter (JUELICH)
Contributors	Harald Servat (BSC), Judit Gimenez Lucas (BSC),Petar Radojkovic (BSC), Markus Geimer (JUELICH)
Reviewer	Chris Adeniyi-Jones (ARM)
Keywords	performance analysis tools, Scalasca, Score-P, Extrae, Paraver, Dimemas

Notices:
The research leading to these results has received funding from the European Community's Seventh Framework Programme [FP7/2007-2013] under grant agreement n° 288777.

Change Log

Version	Description of Change
V0.1	First Template for the contributing partners
V0.2	Combined input from BSC and JUELICH
V0.3	Changes from internal review
V1.0	Last changes with respect to the review/ Version to send to the EC

Table of Contents

Executive Summary

In this deliverable, we present the current status of the prototype versions of the performance analysis tools, considered in the Mont-Blanc project. This includes the community instrumentation and measurement system Score-P, the performance analysis toolset Scalasca with its result browser CUBE, developed by Juelich Supercomputing Centre, and the Barcelona performance tool-suite, containing the instrumentation library Extrae, the analysis tool Paraver and the simulation tool Dimemas. For all of these tools, we describe the current status of the porting to the Mont-Blanc platform as well as the implemented extensions for supporting the OmpSs programming model.

1 Introduction

The main objective of the Mont-Blanc project is to build the first high-performance computing system based on low-power processors using the ARM architecture. In this context, it is essential that application developers are provided with tools supporting their efforts in porting their codes to the ARM platform and optimizing their performance on this architecture.
In this document we present the work done on porting and extending the performance tools of the Mont-Blanc partners to support the ARM platform. In this context, Score-P and Scalasca have been ported to the ARM platform, with the Tibidabo prototype as a test system. Also, Score-P has been extended with prototypical support of the OmpSs programming model. The instrumentation library Extrae has been ported to ARM as well and allows us to gather measurement results for use with Paraver and Dimemas.

2 Score-P, Scalasca and CUBE

Scalasca[1] is a free software tool, developed at the Juelich Supercomputing Centre, which supports the performance optimization of parallel programs by measuring and analyzing their runtime behavior. In the latest release the previous custom instrumentation and measurement system has been replaced by the community measurement system Score-P[2], allowing measurement of applications implemented with MPI, OpenMP and CUDA. As part of the tool suite, the CUBE result browser is responsible for the visualization of performance profiles and trace analysis results.

2.1 Score-P

The newest release of Score-P, version 1.2, has been tested on the Tibidabo test platform. New features, which are of interest in context of Mont-Blanc and OmpSs, are the generic threading extension and the multi-lib approach. The generic threading extension introduced an abstraction to the Score-P measurement system, allowing for other threading models than OpenMP as the underlying basis for the process-local measurement. In the case of OmpSs, a new threading subsystem using Pthreads has been introduced to manage thread local information when dealing with OmpSs threads. With this new subsystem, multithreaded and multi-process measurements of OmpSs applications are now possible.
The multi-lib approach is a solution for the problem of increasing numbers of supported programming models at both the intra- and inter-node level. Instead of supplying and

maintaining a fixed set of monolithic combinations, the multi-lib system chooses the necessary partial libraries and dependencies for the models and systems used at instrumentation time. This now supports arbitrary combinations of systems and models, which becomes useful when adding new threading models like OmpSs.

2.2 Scalasca

Scalasca 2.0 has been released together with Score-P 1.2 as the first Scalasca version using Score-P as underlying measurement system supporting traces in the Open Trace Format 2 (OTF2). With regard to Mont-Blanc, Scalasca 2.0 is able to perform the automatic analysis of MPI measurements on the ARM platform, but currently there are no OmpSs-specific patterns of inefficient behavior detected by Scalasca's automatic trace analyzer. Figure 1 shows an exemplary analysis result for a trace measurement run of a Jacobi benchmark on 256 MPI processes, two per node, on Tibidabo.

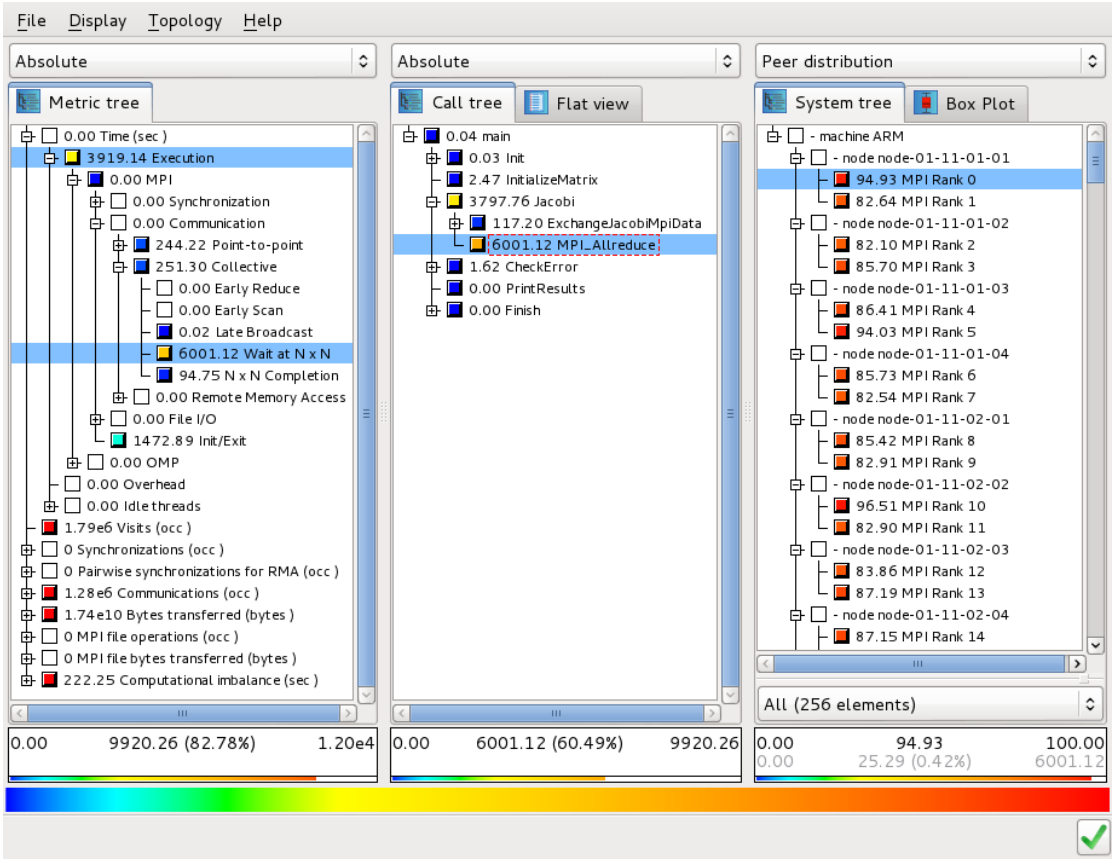

Figure 1: Distribution of wait times at a MPI_Allreduce as a case for the Scalasca pattern "Wait at N x N" in an automatic trace analysis result of a Jacobi benchmark run with 256 processes on Tibidabo using Score-P 1.2 and Scalasca 2.0.

The OmpSs support for Score-P and Scalasca consists of two components. For the connection between the OmpSs runtime and the Score-P measurement system an instrumentation plugin has been created using the provided API. This plugin identifies and translates OmpSs events for the Score-P adapter, which integrates these events into the Score-P task and region model and manages the thread handling by applying the recently introduced generic threading model.

The profiling in CUBE visualizes OmpSs tasks by using the representation of the generic task model used in Score-P, e.g., for OpenMP tasks. For this, an additional root node has been introduced as a sibling of “main” for the display of the task executions as can be seen in Figure 2. The OmpSs adapter now supplies all events and regions for OmpSs runtime functions, like creating, scheduling and executing tasks, synchronization and barriers.

Due to the processing power of the individual ARM nodes, only the CUBE command line tools are used on the platform itself. The graphical user interface will be run on a local computer to review the measurement results.

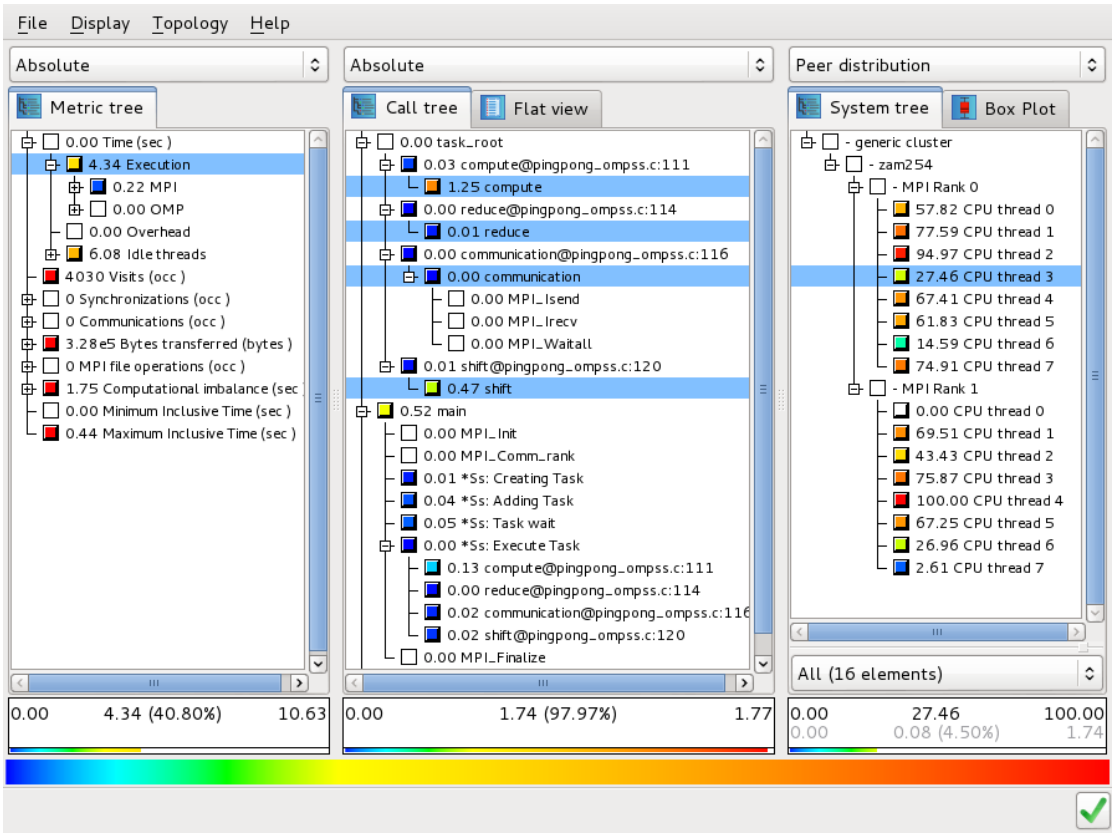

Figure 2: Hybrid MPI/OmpSs example test case illustrating task model representation and OmpSs specific management regions in the main call tree.

3 Extrae, Paraver and Dimemas

The performance tool-suite developed at Barcelona Supercomputing Center includes Extrae, Paraver and Dimemas. Extrae is the instrumentation package that generates time-stamped trace-files for multiple parallel programming paradigms (including MPI, OpenMP, and Pthreads, among many others). Paraver is a very powerful performance visualization and analysis tool based on traces that can be used to analyze any information that is expressed in its input trace format. Finally, Dimemas is a simulation tool for the parametric analysis of the behavior of message-passing applications on a configurable parallel platform.

3.1 Extrae

Extrae has been successfully ported to two ARM system prototypes (to the Tibidabo system and to an Arndale board). The instrumentation package supports most of the functionalities. Besides supporting manual instrumentation, the instrumentation package automatically gathers performance information for many parallel programming models, such as MPI, OpenMP, OmpSs, Pthreads, CUDA, and since recently also OpenCL. However, there is no support for dynamic instrumentation using DynInst because this library has not been ported to ARM.

With respect to the performance data gathered at the instrumentation points, Extrae collects the timestamps for the events using the POSIX clock., which provides nanosecond resolution in this architecture Extrae provides information of the call-stack at MPI entries using the libunwind library. Additionally, the instrumentation library captures the performance counters of the Cortex processors using the PAPI library. PAPI required some operating system kernel changes in the Tibidabo system, but worked out-of-the-box for the Arndale board.

Extrae has been used to instrument multiple applications on the ARM prototype under different conditions. Among the applications instrumented, we can name HPL Linpack, OpenMX [3] and GTC[4]. We have also successfully tested simple OpenCL and hybrid OpenCL+MPI applications on the Arndale board, as depicted in Figure 3.

3.2 Paraver

Paraver is a very flexible data browser that is part of the CEPBA-Tools toolkit developed by BSC and is available for download under an LGPL open-source license. The tool allows a detailed and powerful exploration of trace data. Programmable through configuration files, Paraver can visualize performance data via time-line displays (showing metrics per process or thread over time) or histogram displays (showing statistical data), independently from the programming model used.

While Extrae is required to be executed on the target architecture, and therefore had to be ported, Paraver is typically executed on a desktop machine. Hence, we did not port this part of the tool-suite.

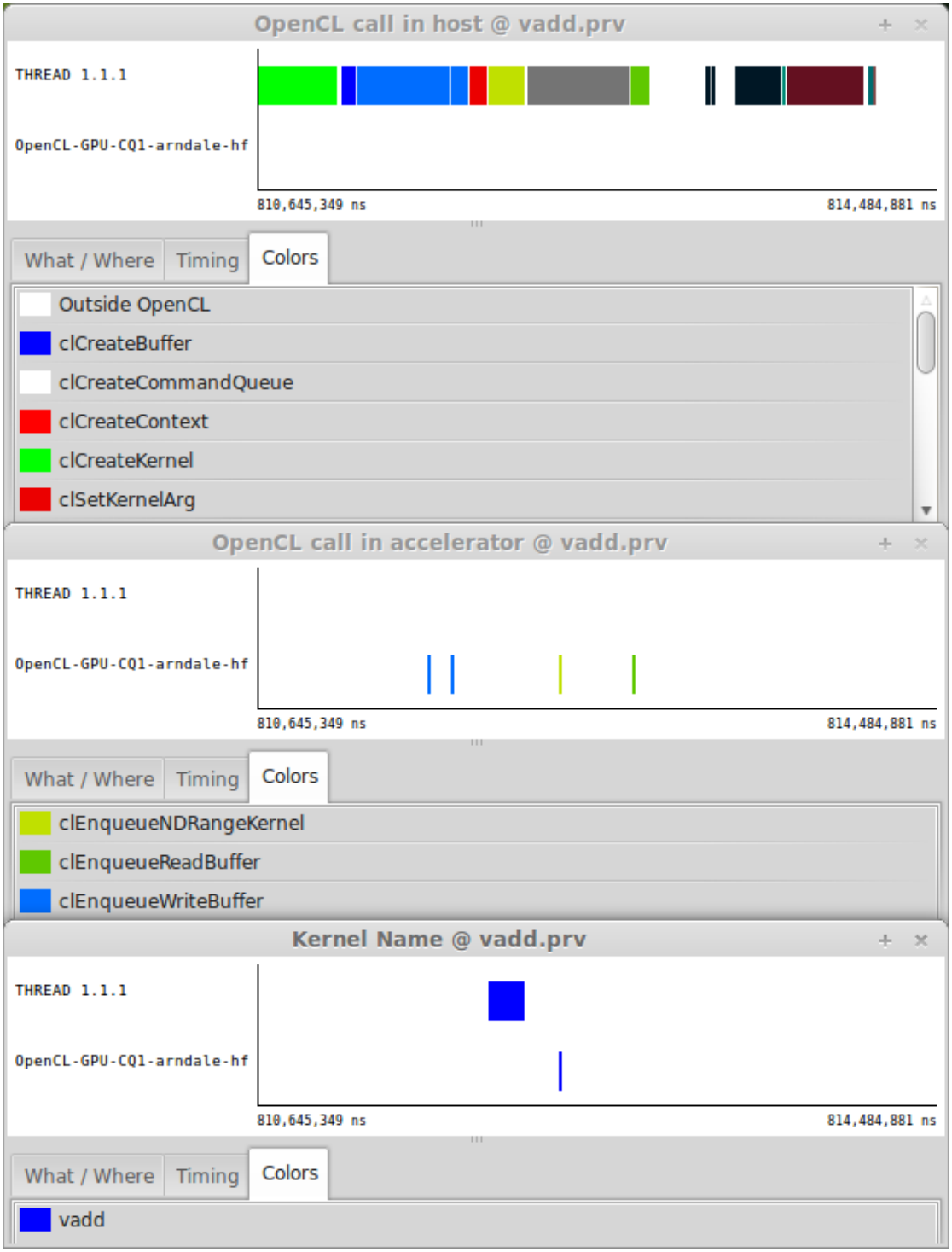

Figure 3 Synchronized Paraver time-lines showing a simple OpenCL application. The windows show OpenCL calls shown on the host-side (at the top), the OpenCL calls executed in the accelerator-side (in center) and the kernel names executed in both sides (at bottom).

3.3 Dimemas

Dimemas is a performance analysis tool for message-passing programs. The Dimemas simulator reconstructs the time behavior of a parallel application trace-file based on a machine

modeled by key factors influencing the performance. With a simple model, Dimemas allows us to simulate complete parametric studies in a very short time frame. Dimemas generates as part of its output a Paraver trace-file, enabling the user to conveniently examine the simulator run in Paraver.

As with Paraver, Dimemas is typically executed in a desktop machine, and therefore we did not port the simulator to the architecture.

Acronyms and Abbreviations

- MPI : Message Passing Interface
- OpenMP : Open Multi-Processing
- OmpSs : OpenMP Super-Scalar
- CUDA: Compute Unified Device Architecture
- OpenCL: Open Computing Language
- DynInst: Dynamic Instrumentation
- PAPI: Performance Application Programming Interface
- POSIX: Portable Operating System Interface uniX
- OTF2: Open Trace Format 2

References

[1] Markus Geimer, Felix Wolf, Brian J. N. Wylie, Erika Ábrahám, Daniel Becker, Bernd Mohr: The Scalasca performance toolset architecture. Concurrency and Computation: Practice and Experience, 22(6):702–719, April 2010.
[2] Andreas Knüpfer, Christian Rössel, Dieter an Mey, Scott Biersdorff, Kai Diethelm, Dominic Eschweiler, Markus Geimer, Michael Gerndt, Daniel Lorenz, Allen D. Malony, Wolfgang E. Nagel, Yury Oleynik, Peter Philippen, Pavel Saviankou, Dirk Schmidl, Sameer S. Shende, Ronny Tschüter, Michael Wagner, Bert Wesarg, Felix Wolf: Score-P – A Joint Performance Measurement Run-Time Infrastructure for Periscope, Scalasca, TAU, and Vampir. In Proc. of 5th Parallel Tools Workshop, 2011, Dresden, Germany, pages 79-91, Springer Berlin Heidelberg, September 2012.
[3] http://www.openmx-square.org/
[4] http://phoenix.ps.uci.edu/GTC/