Juurlink, B.; Lucas, J.; Mammeri, N.; Bliss, M.; Keramidas, G.; Kokkala, C.; Richards, A.

The LPGPU2 Project: Low-Power Parallel Computing on GPUs

The LPGPU2 Project - Low-Power Parallel Computing on GPUs

Extended Abstract

Ben Juurlink
TU Berlin
Einsteinufer 17
Berlin, Germany 10587
b.juurlink@tu-berlin.de

Jan Lucas
TU Berlin
Einsteinufer 17
Berlin, Germany 10587
j.lucas@tu-berlin.de

Nadjib Mammeri
TU Berlin
Einsteinufer 17
Berlin, Germany 10587
mammeri@tu-berlin.de

Martyn Bliss
Samsung Electronics
Communications House
Staines, UK
martyn.bliss@samsung.com

Georgios Keramidas
Think Silicon
Patras Science Park
Rion Achaias, Greece 26504
g.keramidas@think-silicon.com

Chrysa Kokkala
Think Silicon
Patras Science Park
Rion Achaias, Greece 26504
c.kokkala@think-silicon.com

Andrew Richards
Codeplay
Argyle House
Edinburgh, Scotland
andrew@codeplay.com

ABSTRACT

The LPGPU2 project is a 30-month-project (Innovation Action) funded by the European Union. Its overall goal is to develop an analysis and visualization framework that enables GPU application developers to improve the performance and power consumption of their applications. To achieve this overall goal, several key objectives need to be achieved. First, several applications (use cases) need to be developed for or ported to low-power GPUs. Thereafter, these applications need to be optimized using the tooling framework. In addition, power measurement devices and power models need to be developed that are 10x more accurate than the state of the art. The project consortium actively promotes open vendor-neutral standards via the Khronos group. This paper briefly reports on the achievements made in the first half of the project, and focuses on the progress made in applications; in power measurement, estimation, and modelling; and in the analysis and visualization tool suite.

KEYWORDS

GPUs, low power, embedded, power modeling, performance counters, microbenchmarks, visualization, framework

1 INTRODUCTION

Consumers today expect to be able to carry a supercomputer around with them, that has detailed graphical displays, is easy to use and lasts for more than a day on a single battery charge. Not only that, but now users expect their devices to see, listen and understand the world around them. This applies to devices from smartphones that can understand human spoken requests to all the way to cars that can drive themselves. To deliver on this capability requires very power efficient graphics processors ("GPUs").

Power consumption is important to processing, smart power management algorithms such as dynamic voltage frequency scaling (DVFS) are used as a mitigation for high power consumption but this usually results in a less compelling user experience as the CPU and GPU are clocked down to conserve power resulting in less raw processing power. The strict power limitations means that these demands cannot be met through hardware improvements alone, the software must better exploit the available resources. Unfortunately, programmers are hindered when creating low-power GPU software by the quality of current performance analysis tools. As software becomes more complex it becomes increasingly unmanageable for programmers to optimize the software for low-power devices.

The LPGPU^2 consortium has come together to work as a diverse team on delivering low-power GPUs from all the range of angles

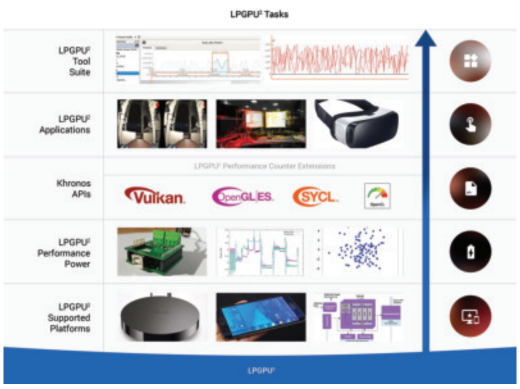

Figure 1: General diagram of the LPGPU^2 framework

required. The consortium combines commercial tools, applications, platform and GPU designers with academic researchers to analyse GPU power and performance, define standard interfaces to reliably measure the power and performance, and create a tool chain to provide clear information and insights to software developers. The companies in the consortium are world leaders in power-efficient GPU design (Think Silicon), GPU and compute tools (Codeplay), graphics standards and applications (Samsung), video codecs and media players (Spin Digital), and the university in the consortium (Technical University of Berlin TUB) has leading experts on parallel applications and multi-core architectures.

This project proposes to aid the programmer in creating software for low-power GPUs by building on the results of the first LPGPU project [2] to provide a complete performance analysis process for programmers. It will address all aspects of performance analysis, from hardware power and performance counters, to a tool chain that processes and visualizes information from these counters. The main objectives of this project are:

1. To help programmers to improve the energy efficiency of their applications, the LPGPU^2 tool chain will provide hints and suggestions to GPU programmers showing ways to reduce power.

2. To enable programmers to be able to write their software once and run it on a variety of different low-power GPUs. The LPGPU^2 project will work on standardizing power analysis and power-efficient programming models.

3. To increase the productivity in GPU software development, we propose an approach in which there are layers of technologies, that all work together via open standards, or open source software.

4. To bring technologies to market in a commercializable form, including productizing and commercializing the technologies developed in the previous LPGPU project [2]. This includes bringing the SYCL™ standard into real-world AI applications and putting the LPGPU video decoders into commercial video systems.

2 APPLICATIONS

As part of LPGPU^2 several applications are being developed and ported to low-power GPU environments. These applications combine graphics and compute parts, and require the use of existing and emerging APIs such as OpenGL ES™ [4], OpenCL™ [7, 11], and Vulkan™ [13], in order to deliver the required performance and functionality. These applications will be used as benchmarks for the power and performance analysis tool being developed as part of the project. Figure 1 illustrates the role of the applications in LPGPU^2 .

The applications under consideration are important commercial applications for individual partners in the area of their core businesses:

- Spin Digital develops video codecs for the next generation of ultra high quality media. In LPGPU^2 Spin Digital is developing a media player based on the H.265 video codec. The main contribution has been the development of a high performance and high quality multi-API video rendering engine. When combined together with the H.265 video decoder, the new video rendering engine allows the creation of state-of-the art media playback applications in areas such as UHD TV, Virtual Reality (VR), and large screen display.
- Samsung graphics team in the UK is responsible for android mobile graphics in their platform. Their applications are related to Virtual Reality (VR), Augmented Reality (AR) and font rendering. Any output from this work will help in improving Samsungs' mobile graphics platform where their mobile phones could be used for VR using GearVR, their mobile camera can used for AR and their mobile phone screen could be used for displaying text using font rendering.
- To realize the possibilities of computational photography, applications must access the hardware at low level. This would allow to control individual ISP blocks, as well as send and receive detailed information from the hardware, including exposure settings, flash, focus point, timestamps, and raw image sensor data. Using the GPU for post-camera processing is a promising direction since the bulk of the memory traffic can be eliminated. Think Silicon has built three different implementations of a set of ISP algorithms (in C, in NEMA|gfx, and in Vulkan). An ISP demonstrator based on NEMA|gfx [10], a proprietary low-level graphics API of Think Silicon, was developed and presented at industrial exhibitions with the goal of exploiting commercial opportunities of executing ISP and post-camera processing algorithms on embedded GPUs.
- TensorFlow™[1], is an artificial intelligence framework that can be used for executing machine learning algorithms. While a computation expressed using TensorFlow can be executed across heterogeneous systems, support has so far been limited to NVIDIA® processors using CUDA® [6]. In order to enable developers to access a wider range of processors, we are working to bring support for OpenCL devices to the TensorFlow framework using SYCL. OpenCL is a framework for writing programs that execute across heterogeneous platforms, and SYCL is a royalty-free, cross-platform C++ abstraction layer that builds on the underlying concepts, portability and efficiency of OpenCL, while adding the ease-of-use and flexibility of modern C++14. Parallelization is also important from a processing and power management perspective. Since tensors are n-dimensional

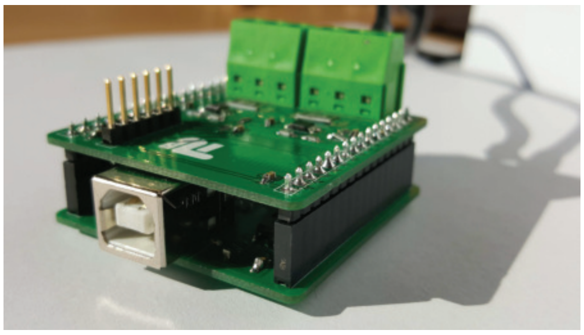

Figure 2: LPGPU^2 power measurement testbed

vectors, having access to parallelization of the TensorFlow code is important not just at the training stage, but also when performing inference on new data sets and this could well be happening on embedded hardware with low power requirements.

3 POWER MEASUREMENT, ESTIMATION & MODELING

Another large area of the project is the measurement, estimation and modeling of the power consumption of embedded low power GPUs, as well as the SoCs that employ these GPUs. A main activity of the LPGPU^2 project is the development of a highly-accurate power models for embedded GPUs. These power models will be integrated in the LPGPU^2 toolchain and will act as a valuable means to locate the most power consuming parts of the executed applications.

3.1 LPGPU2 Power Measurement Testbed

In order to verify and calibrate our power models the LPGPU^2 project also developed its own power measurement testbed for embedded SoCs. The first LPGPU project also used a power measurement testbed, but relied on a CotS USB DAQ and custom signal processing circuits. While this setup worked, it was cumbersome to use due to several issues: Closed source drivers prevented the use of regular up to date Linux distributions, sample rate and resolution could be improved and the wiring between custom signal conditioning was prone to loose contacts. The LPGPU^2 power measurement testbed, shown in Figure 2, improves upon the old testbed: The complete circuit, firmware and host software was designed within the project, sample rate and resolution was improved and new software was written to support measurements of embedded Android based platforms.

3.2 Data-Dependent Power Consumption

The switching activity of CMOS circuits depends on the processed data. As CMOS dynamic power depends on the switching activity this also influences the energy consumption. Our initial experiments showed a large influence of data values on the energy consumption of commercial GPUs. In one experiment the power consumption increased by 65%, when changing the processed data without changing the number of executed instructions or memory accesses patterns. Conventional architectural power models for

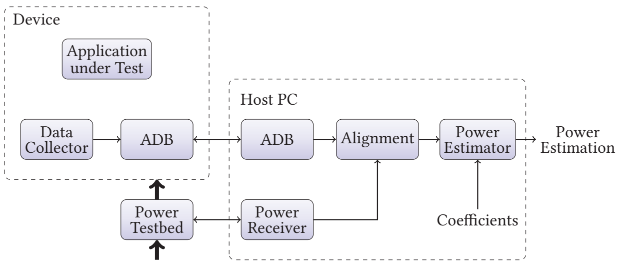

Figure 3: Overview of the Power Model and Evaluation Platform

GPUs do not consider the influence of data on power. The LPGPU^2 project developed a novel power model for GPU ALUs, that takes the data-dependence into account [5]. By considering data dependent metrics such as hamming distances, the power consumption of the data path could be predicted with 85.6% smaller errors than previous models.

3.3 Android Power Model & Microbenchmarks

One result of the project is the development of a flexible power model for Android based systems. The power model collects performance counters suitable for power estimation. A set of microbenchmarks is executed on the device to discover the influence of the different performance counters on the power consumption. This is used to calibrate the power model. After the calibration has been performed for a platform, power consumption can be predicted from the performance counters without requiring extra measurement hardware. An overview over this setup is shown in Figure 3. Android Debug Bridge (ADB) is used as communication channel between a lightweight performance counter collection software running device and the power model running on a host PC. A special alignment procedure is used to ensure that performance counter data and measured power data is synchronized.

3.4 GPU Power Model Verified at Netlist-level

While the previous power model relied on microbenchmarks and public architectural information. We also wanted to discover how accurate a power model can be, if full knowledge of the architecture is knowen as well as access to the hardware description is available. To this end, a fully parameterized power model is created and a methodology for selecting the suitable set of hardware performance counters is developed. The proposed methodology attempts to reduce the set of required hardware counters for a given upper bound or maximum error in estimating the power consumption of embedded GPUs. The outcome of this activity is validated in Think Silicon GPUs and in particular in the 3D Nema|t GPU (multi-threaded and multi-core) [9] assuming various hardware configurations (altering the number of GPU hardware threads, the number of cores), technology nodes (two process technologies; TSMC and FDSOI), and operating configurations (two voltage/frequency levels).

The calibration and validation of the power model is done using fine-grain, netlist-level power measurements using the IC power compiler tools of Synopsis [12] (version VCS-MX K-2015.09). Figure 4 depicts a high level overview of the derived methodology. The

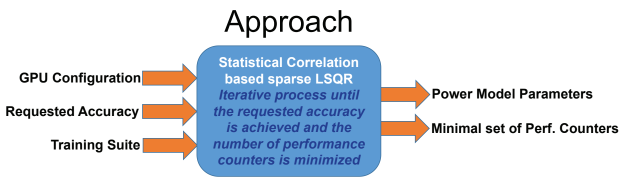
Approach
GPU Configuration
Requested Accuracy
Training Suite
Statistical Correlation based sparse LSQR
Iterative process until the requested accuracy is achieved and the number of performance counters is minimized
Power Model Parameters
Minimal set of Perf. Counters

Figure 4: Power model methodology LPGPU^2 framework

inputs of the power model are: i) the GPU configuration, ii) the requested accuracy of the power model, and iii) the training suite (the testbench suite of the company is used). The outputs of the model are: i) the model parameters (i.e., the weight factors of a parameterized equation which takes also as input the performance counter values), ii) the minimal set of required performance counters for the input accuracy. The heart of the power model is a statistical correlation model based on a least square linear regression (LSQR) algorithm [8].

The development of the power model is divided into two main phases: the training phase and the validation phase. During the training phase, the initial power model is created using the testbenches and the netlist-level power measurements as input. The validation phase includes an iterative phase in which the number of selected hardware performance counters is progressively reduced based on a try-and-error algorithm until the predefined input accuracy is achieved. The experimental results of the validation phase are illustrated in Figure 5 in which a set of ISP and post-camera processing computation kernels is used (developed also as part of the LPGPU^2 project). As Figure 5 indicates, the average error when the full set of performance counters is utilized is well below 1.5% for all studied GPU configurations. Obviously, the error will increase when a meager set of performance counter is used.

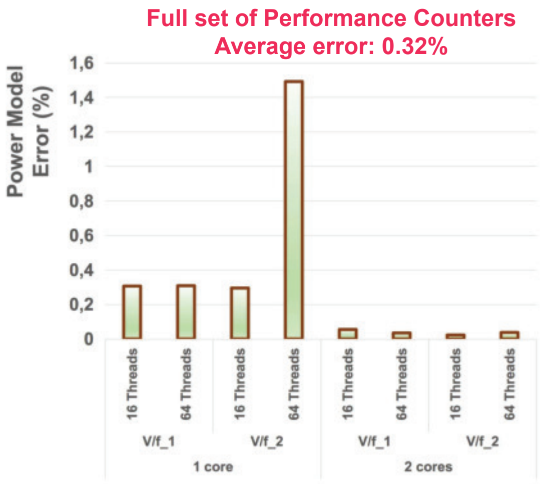

Figure 5: Accuracy of the power model when the full set of the performance counters is used

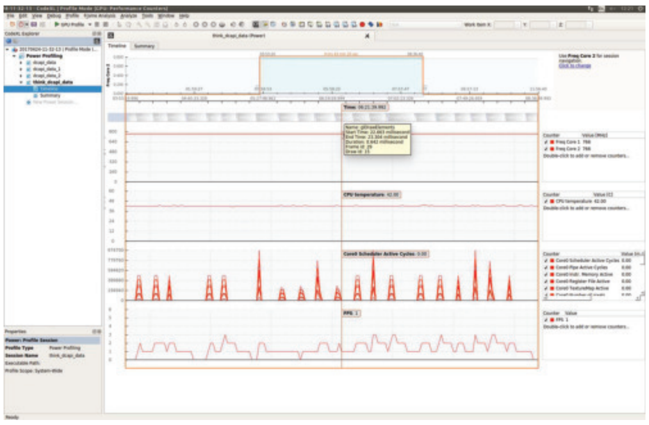

Figure 6: CodeXL displaying data captured from Think Silicon's Nema

4 TOOL SUITE

The tool suite provides the infrastructure and mechanisms to allow the visualization of collected API traces along with hardware counters (performance & power) that are collected in parallel with the API traces. Figure 6 shows a representative example of the data collected being displayed. The tool suite consists of:

- CodeXL
  LPGPU^2 created a fork of the open source CodeXL [3] tool from AMD and has extended this to provide the visualization and processing capabilities required by the project. These include the addition of OpenGL ES trace processing, simultaneous timeline and counter views, extending the database schema to support additional types of data, etc.
- DC API (Data Collection API)
  DC API is a graphics and hardware counter vendor neutral API. It was designed to sit between higher level API's such as OpenGL ES, EGL, Vulkan, etc and the modules that interface with hardware counters.
  By exposing a data driven API, DC API has proven itself flexible enough to support Samsung mobile devices, Think Silicon's Nema system and other android devices without deviating from the initial API definition.
- Interposer (Shim)
  The Shim is a c++ module that is automatically generated from Khronos (or other compliant) API definitions in XML format. Additional boiler plate code is added as part of the build process resulting in an entity that can be used to hook all function calls in a single or multiple API and then perform various actions such as: tracing function names, execution times, monitoring state etc.
- Collector
  The collector is a python module that is responsible for installing the shim, starting and stopping data collection, hosting the DC API module, converting data collected on the target into a form that CodeXL can process and allowing reliable and simple configuration of hardware counters.

The tool suite has been used by Samsung extensively when testing the applications to be delivered as part of the project, and due to the multi-platform capabilities of the tool on Think Silicon's Nema system as well. Support for a third platform will also be achieved by the end of the project.

5 CONCLUSION AND FUTURE WORK

GPUs are becoming mainstream in accelerating many applications, thanks to their parallel processing capabilities. However, power consumption remains a concern especially for applications where there are strict power limitations.

The LPGPU^2 project aims to deliver low power efficient GPU processing not by hardware improvements alone but by leveraging enhancements at the software level. LPGPU^2 aims to enable GPU programmers with a framework that allows them to improve the performance and power consumption of their applications. In this paper, we showed how the LPGPU^2 project addresses the problem from different perspectives; from power models and tool development to applications and algorithm optimizations.

The project devised a toolchain encompassing a GUI interface, APIs, interposer and data collectors. The toolchain equips developers with the infrastructure and mechanisms allowing the visualization an interaction with the collected data (API traces along with hardware counters) for the sake of improving the performance and power consumption of their applications.

Counter-based power models, that can be calibrated to different hardware platforms, were developed and will be integrated within the LPGPU^2 toolchain. These models will be used in estimating power consumption of the running applications based on the collected counter data. Initial results show that average error is well below 1.5% when a full set of performance counters is used.

Several applications combining graphics and compute parts were developed. These applications will be used as benchmarks for the power and performance analysis tool being developed as part of the project. A wide range of applications were developed showcasing font rendering, augmented reality, virtual reality, ISP algorithms, deep learning using the Tensorflow framework as well as an H.265 codec [14] and a new high-performance video rendering engine.

ACKNOWLEDGMENT

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 688759.

REFERENCES

[1] M. et al. Abadi. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). http://tensorflow.org/ Software available from tensorflow.org.

[2] Ben Juurlink et al. 2012. LPGPU: Low Power Parallel Computing on GPUs. (2012). http://lpgpu.org/wp/reports/reports-lpgpu1/ FP7 STREP project co-financed by the EU under the 7th Framework Programme Grant Agreement No.288653.

[3] Advanced Micro Devices Inc. 2016. CodeXL: A comprehensive tool suite that enables developers to harness the benefits of CPUs, GPUs and APUs. (2016). https://github.com/GPUOpen-Tools/CodeXL

[4] John Kessenich, Dave Baldwin, and Randi Rost. 2010. The OpenGL ® Shading Language. *Language* 1 (2010), 1–29. http://www.opengl.org/documentation/specs/

[5] Jan Lucas and Ben Juurlink. 2016. ALUPower: Data Dependent Power Consumption in GPUs. In *2016 IEEE 24th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*. 95–104. https://doi.org/10.1109/MASCOTS.2016.21

[6] NVIDIA. 2015. NVIDIA CUDA C Programming Guide v7.5. http://docs.nvidia.com/cuda/pdf/CUDA_C_Programming_Guide.pdf. (Sep. 2015).

[7] Khronos Opencl. 2009. OpenCL Specification. (2009). https://www.khronos.org/opencl/

[8] Peter Rousseeuw. 1984. Least Median of Squares Regression. *J. Amer. Statist. Assoc.* 79, 388 (1984), 871–880. https://doi.org/10.1080/01621459.1984.10477105

[9] Think Silicon. 2016. 3D Nema|t GPU. (2016). http://think-silicon.com/products/hardware/nema-tiny/

[10] Think Silicon. 2016. Nema|gfx API. (2016). http://think-silicon.com/products/software/nemagfx-api/

[11] John E. Stone, David Gohara, and Guochun Shi. 2010. OpenCL: A parallel programming standard for heterogeneous computing systems. *Computing in Science and Engineering* 12, 3 (2010), 66–72. https://doi.org/10.1109/MCSE.2010.69

[12] Synopsys. 2015. IC Power Compiler. (2015). https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/power-compiler.html

[13] Khronos Vulkan. 2016. Vulkan Specification. (2016). https://www.khronos.org/registry/vulkan/specs/1.0/html/vkspec.html

[14] B. Wang, D. F. de Souza, M. Alvarez-Mesa, C. C. Chi, B. Juurlink, A. Ilic, N. Roma, and L. Sousa. 2017. GPU Parallelization of HEVC In-Loop Filters. *International Journal of Parallel Programming* (Jan 2017), 1–21.