VSI/Pro®-GPU: Commercial VSIPL Support for Single and Multkernel GP-GPU Accelerated Signal and Image Processing based on CUDA and Fermi

Anthony Skjellum, PhD, Jennifer H. Skjellum

RunTime Computing Solutions, LLC
1500 1st Avenue North, Suite C112/U19, Birmingham, AL 35203, USA
{tony,jennifer}@runtimecomputing.com

Overview[1]

In this poster, the VSI/Pro-GPU commercial VSIPL product is introduced. This commercial offering of the VSIPL standard builds on more than thirteen years' experience with commercial VSIPL in the VSI/Pro product family originally designed by MPI Software Technology, Inc and its subsidiaries during the late 1990's. The VSI/Pro-GPU product leverages a commercial license of the academic software library GPU VSIPL for NVIDIA CUDA developed by Georgia Tech by Campbell, Richards, et al [1], which has provided many end-users with an excellent starting point for VSIPL development on NIVIDIA/CUDA-based GP-GPUs, as well as a newly designed architecture and technology and a specific merging of multithreaded VSI/Pro for the multicore processing with GPU accelerated kernels.

The design, scope of functionality in terms of VSIPL profiles, and currently available product feature set are described. Benchmarks with NVIDIA Fermi via the GTX-480 family of NVIDIA graphics cards are offered, front-ended with an Intel multicore Nehalam 920 multicore processor. Demonstration with the earlier TESLA and GTX-295 cards is also demonstrated briefly, in which only single kernels are supported. The use of two GPUs simultaneously is also supported

VSI/Pro-GPU is capable of being employed into defense programs as of now. With the emergence of GPUs for embedded applications, such as the GE NPN 240 [6], use of VSIPL in high performance embedded computing is immediately feasible both in terms of hardware and software support. Embedded GPUs may initially offer less functionality than their desktop counterparts, but such differences may lessen over time.

Programming Model and Library Model

In order to define a production-class VSIPL for GPU's, the VSIPL programming model has had to be slightly extended, with added concepts for heterogeneous memory, multiple computation units, and for mapping to these units as a function of problem instance. Such extensions result in no-ops on non-GPU instances of VSIPL, providing continued portability of the standard and programs written in VSI/Pro-GPU.

Separately, we have proposed strategies that cover multithreaded VSIPL for multicore processors [5,8]. All of the programming model concerns associated with multicore VSIPL discussed in [5,8] have to be addressed in VSI/Pro-GPU, which emphasizes support for NVIDIA CUDA, in which multiple cores can drive the GPU via pthread-type concurrency, and multiple kernels can be simultaneously scheduled on the GPU. In particular, admit-release concepts are extended for multiple memory spaces.

The CUDA driver API is essential to the fine-grain control of GP-GPU programs with non-trivial scheduling goals and with the composition of multiple users of the GPUS in the systems. Unlike the academic GPU VSIPL upon which VSI/Pro-GPU is based in part, users of VSI/Pro-GPU have an underlying driver architecture which allows for far greater control of the system, including dynamic compilation of kernels at runtime. This supports a number of online/runtime optimization possibilities.

Features

The VSI/Pro-GPU product is described in terms of its compliance to the VSIPL standard 1.3. VSI/Pro-GPU accelerated portion provides these capabilities:

- GP-GPU Acceleration of the CoreLite functionality for signal processing, with both single and double precision support.
- Multithreaded VSIPL program execution, where Pthread-type threads work with optimistic locking when they operate on independent data and VSIPL objects, providing their own locks when simultaneously sharing objects or memory between threads.
- Additional functionality and hinting to help establish the use of both multiple GPUs and multiple kernels per GPU.
- The remainder of the Core Full operations are not GPU accelerated as of the September release, but are SIMD accelerated under the VSI/Pro, with plans for future GP-GPU acceleration.
- Extensions to deal with memory and objects bound to different parts of the heterogeneous memory space during execution of an application, with need for explicit or implict copy are described. Reducing explicit copies is also discussed.
- Exploiting multiprocessor concurrency to hide latency is also discussed in terms of the overall use of VSI/Pro-GPU.

[1]

In addition, we demonstrate the ability for the user program or an alternative library to use the Fermi GPU collaboratively with VSI/Pro-GPU.

Importantly, the approach taken by VSI/Pro-GPU makes compliant VSIPL programs work immediately and unlocks availability opportunities for greater performance and scalability by optional use of extended notations and options. Furthermore VSI/Pro-GPU provides an easy upward compatibility path for users of GPU VSIPL.

Image Processing Proposed Standard

As with the VSI/Pro product family for G4/e600, G5, and x86 processor families, RunTime Computing Solutions offers the image-processing draft standard functionality originally promulgated by Stan Ahalt and others. These functions map naturally to GPUs, which are exceptionally well designed and familiar to image processing algorithms. These concepts have been part of the discussion of VSIPL standardization and commercial support for several years (e.g., [7]).

At the September 2010 product release schedule, these functions are demonstration only, but we present their current capability set, and describe how VSI/Pro-GPU can be turned to trade-off using multicore implementations vs. GPU instances depending on image size.

VSIPL++ Options

VSIPL++ offers a convenient way to express VSIPL from C++. Use of the VSIPL++ layerable implementation with VSI/Pro-GPU is discussed, and plans for integrated optimization are described briefly.

Conclusions

VSI/Pro-GPU offers an architecture, implementation, and roadmap for complete VSIPL and VSIPL image processing capability on single and multiple GPUs supporting CUDA, and when exploiting Fermi GPUs, exploiting simultaneous multi-kernel capabilities. This product is ready for use in high performance embedded computing applications.

Current product capability and performance are presented. Needed extensions to the VSIPL standard to exploit this hierarchical processing and memory hierarchy are shown, with a view toward future standardization of these concepts to ensure global portability.

Importantly, the decisions of when to use and not to use the GPU are tunable by the user, or to the library, depending on user's preference, and the ability to capture performance profiling information is discussed. In this sense, optimized library functions have both “SIMD” and “GPU” modes, analogous to scalar and vector modes in a classical vector computer.

VSI/Pro-GPU technology offers defense programs with the ability to move their VSIPL-based applications to GPUs with minimal code rewrite for functionality, and the ability to exploit multprocessing and GPU kernel-based performance immediately, with continued enhancement of performance as the product matures. Continued portability between GPU and non-GPU instances of embedded systems will remain possible with straightforward use of VSI/Pro-GPU, avoiding application “code forks.”

References

[1] D Campbell and M. Richards et al, GPU VSIPL home page, http://gpu-vsipl.gtri.gatech.edu/, accessed May 20, 2010,

[2] VSIPL Standards, URL: www.vsipl.org ; accessed May 19, 2010.

[3] Cain, Kenneth, and Sroka, Brian, “Experiences in Porting an Existing Application to the VSIP API,” MITRE Corporation, July 22, 1997, presented at the VSIP Meeting, URL: http://www.vsipl.org/VSIP_Exp.pdf , accessed May 19, 2010.

[4] Exascale Software Study, Kogge et al, URL: http://users.ece.gatech.edu/mrichard/ExascaleComputingStudyReports/ECSS%20report%20101909.pdf , accessed May 19, 2010.

[5] A. Skjellum, “Multicore, Multithreaded, and/or Multi-GPU-Kernel VSIPL Standardization: Implementation and Programming Impacts: Syntax, Semantics, and Models, short paper submitted to *HPEC 2010*, May 2010.

[6] GE NPN 240 Product home page, URL: http://www.embeddedstar.com/weblog/2010/03/19/npn240-gpgpu/ , accessed May 27, 2010.

[7] Anonymous, VSIPL image processing compatibility, http://www.hpec-si.org/private/HPEC%20-%20Verari%20Image%20Processing.ppt, August 2006, accessed May 27, 2010.

[8] A. Skjellum, “Multicore, Multithreaded VSIPL Standardization: Implementation and Programming Impacts: Syntax, Semantics, and Models,” short briefing, HPEC-SI 2010 meeting, June 2010.