Center for Embedded Systems (CES)
Request for Proposals Template – YEAR 6

TITLE:	Multi-Core Simulator using GPGPU Platforms				
PI:	Spyros Tragoudas	**EMAIL:**	spyros@engr.siu.edu	**TEL:**	(618) 453-7027
DEPT:	Electrical and Computer Eng.	**SCHOOL:**	Southern Illinois University		

ABSTRACT: (250 OR FEWER WORDS)

The proposed project is a continuation of previous work for a multicore simulator implemented using the OpenMPI interface. Previous development in the project was focused on implementation of a simulator which can process data, and estimate timing information for a set multi-core architectures. This allows for an exploration of design choices of the underlying hardware to optimize the performance of the algorithms running on the platform.

It has been found that on typical general computing platforms such as the x86, performance bottlenecks exist. This is primarily caused by the limited number of cores available on the host's platform compared to the many simulated cores. The proposed project seeks to migrate the simulator to a General-Propose computing on Graphics Processing Units (GPGPU) platform, and provide metrics for the order of speed up that is achieved. These platforms have significantly more cores available and may allow for near real-time simulation of multi-core architectures.

PROBLEM:

Current performance measurements of the multi-core simulator have shown that processing a single 720p video stream using simple image processing algorithms was unable to keep up with a real-time feed on a Core i7 platform. The performance bottleneck stems from the attempt to simulate many more cores than is available on a typical CPU. This then places the requirement that the CPU cores must have significantly more throughput then the simulated platform's cores which is often not the case.

While the multi-core simulator supports cluster computing, this is not ideal for real-time simulation of image processing since round trip transmission of data over a network has high latency. Given a frame rate of a 30 fps, each frame must be loaded, processed, and stored within 33ms. To achieve this the data should be kept locally. While the number of cores in x86 architectures has increased, most of the emphasis has been increasing the performance of each core, and increasing the performance of the integrated graphics processor.

RATIONALE:

Many image processing functions relay on local pixels to generate an output. This property has allowed for near optimal parallelism for many of the algorithms. Furthermore, the use of image sensors in embedded systems has increased dramatically in recent years. Cameras, cars, drones, planes and many other platforms have deployed these sensors for data collection and safety critical systems. As such, it has been a primary goal for the simulator to support real-time image processing. This allows for the algorithms to be tested and performance data generated using live video streams. The performance data can then be matched to an optimal platform so that hardware resources are not wasted, and costs are kept to a minimal.

APPROACH:

In current form, the multi-core simulator has been implemented using Open-MPI, which is a high performance message passing interface. Each simulated core is run as a separate process, and intra-core communication is simulated using message cues and MPI's protocol. While running the cores in the open-MPI environment in itself does not accurately depicted the timing of the cores, the simulator tracks the sequence of instructions and records it in a log. A post process timing analysis algorithm then gives an estimate of the timing of the simulated

hardware platform. The multi-core simulator is scalable and can use cluster computing to speed up simulations. The simulator also has the ability to stream live data to the cores allowing greater flexibility in verifying the parallel algorithms.

Open-MPI supports dispatching tasks to a GPGPU environment. As such, the simulator is likely to continue to use the developed MPI interface as it is migrated to the new environment. There are two major GPGPU platforms that can be targeted for migration of the tool, CUDA [1] and OpenCL [2]. It has been show there are trade-offs between using CUDA which generally has higher performance, and OpenCL which has greater hardware support and considered more portable [3]. A further survey of the platforms will be conducted to identify which GPGPU platform will be selected.

As the Open-MPI environment is migrated to the GPGPU environment, special consideration will be made for finding efficient methods for inter-core communication. Tasks are typically dispatched to the GPGPU as threads, and blocks. It is likely that a simulated core may require communication with another that resides in another block. While intra-block synchronization is easy, inter-block synchronization is more challenging, and will likely require resolution using either MPI calls or leveraging global memory. Minimization of data transfers over the graphics bus will also be considered since transferring data is often a bottleneck to GPGPU performance [4] [5].

After completion of migrating the platform to the GPGPU environment, the performance of several image processing algorithms in the environment will be measured [6] [7]. The data collected will be used to determine the speed up achieved over the original computing environment. Furthermore, stress tests will be performed to determine how many cores and video streams can be executed using current GPGPU hardware.

NOVELTY:

Current multi-core simulators model low level architecture timing. While these simulators give very precise timing information, they require significant processing power just to simulate a few instructions. Thus real-time simulation of data is not feasible using such a method. The proposed simulator reduces the accuracy of software timing estimates in exchange for increasing the simulated data throughput. This allows for companies to quickly implement and verify parallel algorithms targeted for a particular architecture. In conjunction, the solution space of the number of cores in the hardware, and the architecture can be explored to find the optimal ASIC design.

POTENTIAL BENEFITS TO INDUSTRY MEMBERS:

The use of multicore platforms in industry has become common place. However, as a developer it is often unclear as to what the optimal hardware platform is. The variety of embedded system platforms is increasing, and developers must choose a platform with the ideal number of cores. Unfortunately it is difficult to understand the performance of the system prior implementation.

DELIVERABLES:

- Survey of GPGPU Platforms and Trade-Offs
- Multi-Core Simulator Running on a GPGPU Platform
- Benchmarks on the a Multi-Core Simulator using Common Image Processing Algorithms
- Documentation of software tool use

TIMELINE/MILESTONES: (PER QUARTER)

- Select a GPGPU Platform – 9/30/2014
- Migrate Multi-Core Tool To GPGPU – 02/28/2015
- Performance Analysis Using Image Processing Algorithms – 04/30/2015
- Transfer Multi-Core Tool and Documentation – 07/31/2015

TECHNOLOGY TRANSFER:

Biweekly teleconferences with the industrial liaisons will occur throughout the duration of the project to ensure deliverables are met and within scope.

BUDGET:

$25,000 is requested to support the PIs and two graduate students, and to purchase demonstration hardware and software tools.

BIBLIOGRAPHY: (ATTACH IN IEEE CONFERENCE OR JOURNAL FORMAT)

[1] "CUDA Toolkit Documents," 2014. [Online]. Available: http://docs.nvidia.com/cuda/.

[2] "The open standard for parallel programming of heterogeneous systems, OpenCL 2.0," 2014. [Online]. Available: https://www.khronos.org/opencl/.

[3] Jianbin Fang, A. Varbanescu and H. Sips, "A Comprehensive Performance Comparison of CUDA and OpenCL," in *International Conference on Parallel Processing (ICPP)*, 2011.

[4] Wei Yi, Guibin Wang and Xudong Fang, "A Case Study of SWIM: Optimization of Memory Intensive Application on GPGPU," in *Third International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, 2010.

[5] C. Gregg and K. Hazelwood, "Where is the data? Why you cannot debate CPU vs. GPU performance without the answer," in *2011 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, 2011.

[6] E. Monteiro, M. Maule, F. Sampaio, B. Zatt and S. Bampi, "Real-time block matching motion estimation onto GPGPU," in *19th IEEE International Conference on Image Processing (ICIP)*, 2012.

[7] R. Kutil, E. Peter and M. Watzl, "SIMD parallelization of common wavelet filters," in *In Parallel Numerics*, 2005.

PI INFORMATION: (ATTACH 2-PAGE CV)

SPYROS TRAGOUDAS

PROFESSIONAL AFFILIATION AND CONTACT INFORMATION

Electrical & Computer Engineering Department, Southern Illinois University, Carbondale, IL 62901, ENGR-E, Rm 113, MC 6603, tel: (618) 453 7027, e{mail: spyros@engr.siu.edu, fax (618) 453 7972

EDUCATION

1986 Diploma (5 years), Computer Engineering and Informatics Department, University of Patras, Greece

1988 M.S., Erik Jonsson School of Engineering and Computer Science, Computer Science Program, The University of Texas at Dallas, Richardson, TX 75083-0688.

1991 Ph.D., Erik Johnson School of Engineering and Computer Science, Computer Science Program, The University of Texas at Dallas, Richardson, TX 75083-0688.

PROFESSIONAL EXPERIENCE

07/01/12 – current Professor and Chair, Electrical & Computer Eng. Dept., Southern Illinois University Carbondale

03/01/09-current Director, NSF IUCRC on Embedded Systems, SIUC-site.

07/16/99- current Professor, Electrical & Computer Eng. Dept., Southern Illinois University Carbondale.

08/16/98-07/15/99 Associate Professor, Electrical and Computer Engineering Department, University of Arizona.

08/16/91-08/15/98 Associate Professor, Computer Science Department, Southern Illinois University Carbondale (Assistant Professor until 6/30/96).

07/01/97-08/15/98 Graduate Program Director, Computer Science Department, Southern Illinois University Carbondale.

01/03/87-08/14/91 Research/Teaching Assistant, Computer Science Program, School of Engineering and Computer Science, The University of Texas at Dallas, Richardson, TX 75083-0688.

08/15/86-01/02/87 Systems Analyst, Computer Technology Institute, Patras, Greece.

RESEARCH INTERESTS

Design and Test Automation for VLSI, Embedded Systems

RESEARCH SPONSORS

Direct support: National Science Foundation, US Navy, SAIC, Intel, Qualcomm, Synopsys

NSF IUCRC: NSF, NAVSEA Crane, Rockwell Collins, United Technologies Aerospace Systems, SAIC, Intel, Caterpillar, TSI, EMAC, Wildlife Materials

PROFESSIONAL SERVICE

Editorial Board: IEEE Transactions on Computers, VLSI Design journal, Journal of electrical and Computer Engineering, Universal Computer Science, Research Letters in Electronics.

General Chair of IEEE DFTS 2010, Program Committee Chair of DFTS 2009, Program Committee member of many International Conferences

Has graduated 14 PhD students and supervised over 60 MS theses. Currently advising 11 PhD students

PUBLICATIONS

Over 70 journal papers and over 130 articles in peer-reviewed conference proceedings

Ten recent journal publications

• A.K. Palaniswamy and S.Tragoudas, An Efficient Heuristic to Identify Threshold Logic Functions, ACM Journal on Emerging Technologies in Computing (JETC), to appear in 2012.

• M.N. Skoufis, S. Tragoudas, An on-line Failure Detection Method for Data Buses using Multi-threshold Receiving Logic, IEEE Transactions on Computers, vol. 61, no. 2, pp. 187-198, Feb. 2012

• K. Stewart, Th. Haniotakis, and S. Tragoudas, Securing sensor networks: A novel approach that combines encoding, uncorrelation, and node disjoint transmission, Ad Hoc Networks, vol. 10, issue 3, May 2012, pp. 328-328, Elsevier.

• M.N. Skoufis, K. Karmakar, S. Tragoudas, and T. Haniotakis, A data capturing method for buses on chip, IEEE Transactions on Circuits and Systems I, vol. 57, no. 7, pp.1631-1641, July 2010.

• D. Jayaraman, R. Sethuram, and S. Tragoudas, Scan Shift Power Reduction by Gating Internal Nodes. J. Low Power Electronics 6(2): 311-319 (2010).

• E. Flanigan, S, Tragoudas, Path Delay Measurement Techniques using Linear Dependency Relationships, IEEE Transactions on VLSI Systems, vol. 18, issue 6, pp.1011-1015, June 2010.

• R. Adapa, S. Tragoudas, Techniques to Prioritize Paths for Diagnosis, IEEE Transactions on VLSI Systems, vol. 18, issue 4, pp. 658-661, April 2010.

• K. Christou, M. K. Michael, and S. Tragoudas, On the Use of ZBDDs for Implicit and Compact Critical Path Delay Fault Test Generation, Journal of Electronic Testing: Theory and Applications, 2008.

• A. Abdulrahman and S. Tragoudas, Low-Power Multi-Core ATPG to Target Concurrency, Integration, the VLSI Design Journal, vol. 41, issue 4, pp. 459-473, July 2008.

• C. Song, S. Tragoudas, Identification of Critical Executable Paths at the Architectural Level, IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems (TCAD), vol. 27, no. 12, pp. 2291-2302, December 2008

I/UCRC Executive Summary - Project Synopsis		Date: 3/31/2014
Project Title: Multi-Core Simulator using GPGPU Platforms		
Center/Site: **Southern Illinois University Carbondale**		
Principle Investigator: Dr. Spyros Tragoudas		**Type: Continuing**
Tracking No.:	**Phone :** (618) 453 - 7027	**E-mail : spyros@engr.siu.edu**
		Proposed Budget: $25,000

Abstract:
The proposed project is a continuation of previous work for a multicore simulator implemented using the OpenMPI interface. Previous development in the project was focused on implementation of a simulator which can process data, and estimate timing information for a set multi-core architectures.

It has been found that on typical general computing platforms such as the x86, performance bottlenecks exist. This is primarily caused by the limited number of cores available on the host's platform compared to the many simulated cores. The proposed project seeks to migrate the simulator to a General-Propose computing on Graphics Processing Units (GPGPU) platform, and provide metrics for the order of speed up that is achieved.

Problem:
- Current simulator experiences bottlenecks on high resolution images due to the lack of cores available on traditional desktop platforms
- Bottlenecks erode the ability to process simulation data in real-time, which is a key feature of the simulator

Rationale / Approach:
- The simulator will be converted from a CPU platform to a GPGPU platform
- GPGPUs offer many more processing cores than traditional CPUs allowing more scalable implementations

Novelty:
- Through estimations the current simulator platform is able to rapidly simulate timing of a multi-core system
- Allows developers to explore the design space prior to committing to a hardware platform

Potential Member Company Benefits:
- Allows members to choose a platform which maximizes performance and minimizes cost
- Increase in simulator performance allows for real-time data simulation giving better indication on the simulated platforms performance

Deliverables for the proposed year:
- Survey of GPGPU Platforms and Trade-Offs
- Multi-Core Simulator Running on a GPGPU Platform
- Benchmarks on the a Multi-Core Simulator using Common Image Processing Algorithms
- Documentation of software tool use

Milestones for the proposed year:
- Select a GPGPU Platform – 9/30/2014
- Migrate Multi-Core Tool To GPGPU – 02/28/2015
- Performance Analysis Using Image Processing Algorithms – 04/30/2015
- Transfer Multi-Core Tool and Documentation – 07/31/2015

Progress to Date:
- Implemented high level multicore simulator
- Added capability to stream real-time data to the simulator

Estimated Start Date: 8/1/2014	**Estimated Knowledge Transfer Date**:7/31/2015