

## Gregory F. Damos

---

1911 Dry Creek Road. Campbell, CA. 95008  
gregory.damos@gatech.edu (404) 693-4094

### OBJECTIVE

To address challenging and novel problems in computer architecture, compilers, and systems with a particular interest in heterogeneous many core architectures, dynamic compilers, and locality-aware execution models.

### EDUCATION

**Georgia Institute of Technology:** Atlanta, Georgia – September 2004 to Present

- PhD in Electrical Engineering – June 2011.
- MS in Electrical Engineering – May 2008  
*4.0 CS and Computer Architecture, 3.83 cumulative GPA.*
- BS in Electrical Engineering – December, 2006 - *135 credit hours in 2.5 years. 3.95 ECE and 3.74 GPA.*

### SKILLS

**Languages** (Lines in Largest Program Written):

- C/C++ (**74922**), VHDL (22361), Python(7549), Verilog(1000), PTX, Cell SPU, **LLVM**

**Libraries:**

- Flex, Bison, Boost, STL, C++ TR1, **NVIDIA CUDA**, OpenCL, MPI, Pthreads, OpenMP

**Open Source Projects** (Code Samples):

- Ocelot ( <http://code.google.com/p/gpuocelot/> )
- Harmony ( <http://code.google.com/p/harmonyruntime/> )

### EXPERIENCE

**NVIDIA Corporation:** Santa Clara, California — September 2010 to Present

- Consulting on topics related to DARPA UHPC project Echelon.

**Intel Corporation:** Folsom, California — May 2010 to September 2010

- Contributed to the OpenCL compiler for the GEN6 and GEN7 Intel GPUs.

**LogicBlox Corporation:** Atlanta, Georgia — May 2009 to August 2009

- Developed a compilation chain from a relational database language to GPUs.

**NVIDIA Corporation:** Santa Clara, California — May 2008 to August 2008

- Evaluated the performance and functionality of Fermi GPUs for general purpose applications.

**Sandia National Labs:** Livermore, California — May 2007 to August 2007

- Designed and synthesized a flash memory controller and NoC as part of a SoC with a PCIe interface, flash memory banks, and embedded PowerPC processor.

### AWARDS

**NVIDIA Research Fellowship**

- 2 consecutive years. 9 selected out of 120 PhD applicants in 2008.

### SELECTED PUBLICATIONS

Gregory Damos, Andrew Kerr, Sudhakar Yalamanchili, and Nathan Clark. "Ocelot: A Dynamic Compiler for Bulk-Synchronous Applications in Heterogeneous Systems." In *PACT'10*, Vienna, Austria, September 2010. ACM.

Gregory Damos and Sudhakar Yalamanchili. "Speculative Execution in Multi-GPU Systems." In *IPDPS'10*, Atlanta, Georgia, USA, April 2010. ACM.

Andrew Kerr, Gregory Damos and Sudhakar Yalamanchili. "Modeling GPU-CPU Workloads and Systems." In *GPGPU'10*, Pittsburgh, Pennsylvania, USA, March 2010. ACM.

Andrew Kerr, Gregory Damos, and Sudhakar Yalamanchili. "A Characterization and Analysis of PTX Kernels." In *IISWC'09*, Austin, Texas, USA, October 2009. IEEE.

Gregory Damos and Sudhakar Yalamanchili. "Harmony: An Execution Model and Runtime for Heterogeneous Many Core Systems." In *HPDC'08*, Boston, Massachusetts, USA, June 2008. ACM.

Gregory Damos, Andrew Kerr, and Mukil Kesavan. "Translating GPU Binaries to Tiered SIMD Architectures With Ocelot." Technical Report 0901, January 2009.

Sudnya Padalikar and Gregory Damos. "GPU-RPC: Exploiting The Latency Tolerance of CUDA Applications." In *NVIDIA Research Summit*, San Jose, California, USA, September 2009.

Gregory Damos, Sudhakar Yalamanchili, and Jose Duato. "Stars: A System for Actively Tuning and Reconfiguring SOC Links." In *DATE07*. ACM, 2007.

## PHD THESIS TOPIC

### **Harmony: An Execution Model and Runtime for Heterogeneous Many-Core Systems**

- An execution model is proposed that allows applications to be specified as a sequence of control and compute kernels with associated metadata. Each kernel is dynamically generated for target architectures.
- A parallel schedule maps kernels to available processors. Regression models use dynamically gathered profiling data to predict the execution time of kernels on specific processors.
- Speculation and dynamic memory renaming improve concurrency, while kernel combining/splitting reduces overheads.

## MAJOR PROJECTS

### **Ocelot PTX to LLVM Translator: Backend compiler from PTX 1.4-2.1 to LLVM 2.9**

- Each PTX instructions is translated into an equivalent series of LLVM instructions.
- Lightweight user-level threading library handles the possibly thousands of PTX threads efficiently, context switching as infrequently as possible.
- PTX transformation passes reverse if-conversion and add context switch points at PTX barriers.
- Runtime component maps one pthread to each x86 core, and assigns many PTX threads to each pthread.

### **Ocelot: GPU Emulator Implementing PTX 1.4**

- Complete reimplementaion of the PTX virtual machine model including texture interpolation and complex floating point rounding modes. The PTX SIMD model is mapped onto a single CPU thread. The GPU memory hierarchy is emulated in host memory.
- CUDA runtime reimplementaion intercepts calls to the CUDA driver and emulates them on the host CPU.

### **PTX to CELL SPU Just in Time Compiler**

- A parser converts PTX assembly files to an internal control flow graph representation.
- Data and control flow analysis are used to determine register live ranges, while dominance analysis is used to find synchronization points.
- Individual PTX instructions are translated to equivalent SPU instructions.
- The generated binary is linked against a runtime component that maps the PTX thread hierarchy to individual SPUs in Cell.

### **Compiler From Datalog<sub>LB</sub> to GPU Relational Algebra Operators**

- High performance implementations of relational algebra operators (Union, Intersection, Difference, Project, Select, Rename) in CUDA
- Micro-Benchmarks of each operator, obtaining between 20%-80% of peak performance on an NVIDIA 285GTX GPU
- Manual compilation from Datalog to Relational Algebra to prove that it is possible. 60-80x speedup of a financial risk analysis application using this library over an equivalent C++ implementation

### **Parallel Discrete Event Simulator for Multiprocessor Networks and Network On Chips**

- Optimistic, conservative, and stand alone simulator kernels with identical interfaces.
- Models for 5-stage pipelined routers with virtual channels, pipelined links, network interfaces, and packet generators.
- Topology descriptions with visualization support through graph viz. Generators for meshes, tori, fat trees, and k-ary-n-cubes.

### **Architecture Techniques for Controlling PVT Variations**

- Process, voltage, and thermal variations at future technology nodes make worst-case design margins impractical.
- Real time measurements of link delays are used to dynamically tune the system clock frequency.

- All Digital PLL and delay detectors, implemented in TSMC .13um and verified with MOSIS HSPICE models.

#### **Scalable Interconnect for Flash Memory Controller Arrays**

- Script for automatically generating a K-dimension, adaptively routed, torus network with a variable number of routers that is synthesizable in VHDL.
- Memory controllers for individual flash memory banks are connected via the torus network which scales to an arbitrary number of memory banks and external devices with no VHDL code modifications.
- Synthesized and verified at 100 MHz on a Xilinx VirtexII-Pro FPGA.

#### **Memory Controller for Micron MT29F16G08 NAND Flash Memory**

- Support for basic read/write/erase operations as well as burst read/writes. Automatic detection of burst operations.
- DMA engine for low overhead memory copies.
- Timing parameters specified as generics to support design reuse.
- Implemented in VHDL. Synthesized and verified at 100 MHz on a Xilinx VirtexII-Pro FPGA.

#### **Hardware Accelerator for iDCT and Inter-Frame Window Synthesis in MPEG Audio Decoding**

- Tightly coupled SoC system consisting of a general purpose RISC processor and a VLIW accelerator.
- VLIW accelerator was optimized for 32x32 iDCT operations and Kaiser Windowing.
- An open source software MPEG decoder was compiled for the RISC processor with the iDCT and Windowing operations moved to the VLIW accelerator.

#### **Locally Synchronous Globally Asynchronous Interconnection Network for SoCs**

- Separate clock domains for each device connected to the network as well as the network itself.
- Pipelined, asynchronous FIFO buffers for crossing clock domains.
- Intelligent monitoring and control logic used to adjust the network clock frequency and link channel width to meet the current performance requirement while consuming the lowest possible power.

#### **Group Communication Protocols for Wireless Sensor Networks**

- Provides acknowledged message passing layer for reliable communication.
- Provides group membership service for dynamically adding and removing nodes from a group and total ordered message delivery within a group.
- Automated methods for backing up critical data across multiple nodes and restoring the state of a specific node after a crash.

# RELATED COURSEWORK

## Computer and Network Architecture

- ECE 8833 Polymorphic Many-Core Architectures
- CS 8803 DCM Dynamic Compilation Techniques
- ECE 8813a Multiprocessor Network Architectures
- ECE 7142 Fault Tolerant Computing
- CS 8803 HPC High Performance Network Architectures
- CS 6230 High Performance Parallel Architectures
- CS 8803 AMA Advanced Micro Architecture
- ECE 6102 Dependable Distributed Systems
- ECE 6100 Advanced Computer Architecture
- ECE 8863 Wireless Sensor Networks
- ECE 4170 Hardware Design Languages VHDL and Verilog
- ECE 3076 Network Protocols and Coding Schemes
- ECE 3055 Computer Architecture and Operating Systems

## Signal Processing

- ECE 6258 Digital Image Processing
- ECE 6254 Statistical Signal Processing
- ECE 4720 Fundamentals of Digital Signal Processing
- ISYE 3720 Statistical Analysis
- ECE 3075 Random Signal Analysis
- ECE 2025 Intro to Signal Processing

## Digital Logic

- ECE 6130 Advanced VLSI Layout
- ECE 4420 Digital MOS Circuits
- ECE 2031 Digital Logic Lab
- ECE 2030 Intro to Computer Engineering

## Semiconductors

- ECE 4751 Semiconductor Lasers and Optical Principles
- ECE 3042 Microelectronic Circuits Lab
- ECE 3040 Solid State Physics and Microelectronic Devices
- PHYS 3143 Quantum Physics
- MSE 2001 Basic Materials Science Principles

## Analog Circuits

- ECE 3025 Electromagnetic Theory
- ECE 3050 Analog Semiconductor Devices
- ECE 3041 Circuits Lab
- ECE 2040 Circuit Analysis

## REFERENCES

1. Prof. Sudhakar Yalamanchilli (sudha@ece.gatech.edu)  
Professor, School of Electrical and Computer Engineering  
Georgia Institute of Technology, Atlanta, GA
2. Prof. Karsten Schwan (schwan@cc.gatech.edu)  
Professor and CERCS Director, College of Computing  
Georgia Institute of Technology, Atlanta, GA