MADHAVI KRISHNAN

4553 Guadalupe Apt B310
Austin, TX- 78751

E-mail: madhavi@cs.utexas.edu
Phone: 512-913-9370

OBJECTIVE: Seeking a full time industry position in Computer Sciences.

EDUCATION:

MA, Computer Science, (Aug 2008)
The University of Texas, Austin
GPA: 3.78/ 4

BE, Computer Science (Apr 2005)
College of Engineering, Guindy
Anna University, India
GPA: 9.11/10

RESEARCH INTERESTS:

Computer Architecture, High Performance Computing
Parallel Computing - Architecture, Programming and Applications

AWARDS AND HONORS:

- Intel Kudos during Summer 2006 internship at Intel for work on micro-op placement
- TCS Best Student Award 2005 (Anna University)
- JNCASR (**J**awaharlal **N**ehru **C**enter for **A**dvanced **S**cientific **R**esearch) Summer Research Fellowship 2004.
- HiPC Scholarship and Second Best Poster Award in the International Conference on High Performance Computing 2004 (HiPC 04).

COMPUTER SKILLS:

Languages Known: C, C++, Java, Perl, Assembly (x86), Visual Basic 6.0.
Application programs: MATLAB, Mathematica

GRADUATE COURSES:

CS382M Advanced Computer Architecture
CS388G Algorithms: Theory and Techniques
CS395T High performance and Parallel Computing
CS395T Fine Grained Parallelism
CS380C Compilers
CS394N Neural Networks
CS389R Recursion and Induction I
CS380L Advanced Operating Systems

EXPERIENCE:

Jan '07 - present *Research Assistant for Prof. Stephen W.Keckler of Dept.* of *Computer Sciences, The University of Texas at Austin.* Developed MPI library for TRIPS prototype chip. Now working on an integrated approach to fine grained parallelism focusing on interaction between hardware primitives, programming language abstraction and parallel applications.

Aug '06 - Dec'06 *Teaching Assistant, Dept. of Computer Sciences, The University of Texas at Austin, Computer Systems Architecture (Fall 2007)*, instructor Prof. Stephen W.Keckler. My duties as TA included discussion sections, holding office hours, grading homework programming assignments and exams, and preparing solution sets.

May '06 - Jul '06 *Summer Intern at Intel*. Worked with the microcode team (Manager Alan. G Lee) on optimization of micro-op placement algorithm and verification of FP transcendental functions.

Jan '06 - May '06 *Teaching Assistant, Dept. of Computer Sciences, The University of Texas at Austin, Computer Systems Architecture (Spring 2006)*, instructor *Prof.* Stephen W.Keckler.

Aug '05 - Dec '05 *Teaching Assistant, Dept. of Computer Sciences, The University of Texas at Austin, Computer Organization and Programming (Fall 2005)*, Instructor Chris Edmondson-Yurkanan..

May '04 - Jul '04 *JNCASR Summer Research Fellowship.* Worked at the Institute of Mathematical Sciences (IMSc), India, on "**OpenMp implementation in MPI environment**" on the Linux cluster, KABRU, India's second fastest supercomputer, which hit the Top 500 List.

May '03 - Dec '04 *Research Assistant for Dr.K.Murali of the Physics department, Anna University*.

PUBLICATIONS:

Dipak Krishnamani, **Madhavi Krishnan**, Sriram Shankar, and Ranjani Parthasarathy, "Basic Block Architecture for Power Saving (B_2APS)", Poster at **International Conference on High Performance Computing (**HiPC'04), Bangalore, India, December 2004

K. Satish, T. Jayakar, Charles Tobin, **Madhavi Krishnan,** and K. Murali, 'Chaos based Spread Spectrum Image Steganography', **IEEE Transactions on Consumer Electronics**, vol 50, No.2, May. 2004, p 587-590

PROJECTS (REFER PROJECT DETAILS):

- "**MPI on TRIPS**" - developed MPI (Message Passing Interface) library on TRIPS; evaluated mapping of hardware primitives like DMA, scratch pad memory to MPI programming model.
- "**A Trace-Driven File System Workload Characterization of Laptops**" as part of the graduate "Advanced Operating Systems" course.
- "**Compiling streaming programs to Silicon**" as part of the graduate "Fine Grained Parallelism" course
- **"Modeling Dataflow Execution in ACL2**" as part of the graduate "Recursion and Induction" course.
- "**Evolving Test Programs for Verifying microprocessor**" as part of the graduate "Neural Network" course.
- "**Cholesky Factorization using OpenMP**" as part of the graduate "High performance and parallel computing" course.
- "**Profile guided top or bottom predication"** as part of graduate "Compilers" course
- **"Pattern Matching on TRIPS SVM System**", as part of the graduate "Advanced Computer Architecture" course.
- Undergraduate Project in Final Year: "**Basic Block Architecture for Power Saving (B_2APS)**"
- "**Thread selection for SMT processor based on Function Unit usage in hardware**" as a part of undergraduate "Computer Architecture" course.
- "**Implementation Of Pthreads and OpenMP In An MPI Environment**" as a part of summer research at IMSc, Chennai, India.
- "**Linux Parallel Port Printer driver**"- Programmed the kernel and user level modules for a Parallel Port Printer Driver in Linux.

PERSONAL INFORMATION:

Date of Birth: March 27, 1984
Citizenship: India
Visa status: F1

AVAILABILITY: SEPTEMBER 1, 2008

REFERENCES: AVAILABLE ON REQUEST

Project Details

1. **MPI on TRIPS**
   In this work, we implemented MPI on TRIPS architecture to study how a subset of primitives, namely DMA and configurable memory can be mapped to MPI programming and communication model. We developed a full MPI library for TRIPS. We evaluated the hardware primitives using microbenchmarks and the NAS benchmarks. The library is being used at ISI and AFRL for TRIPS application studies.

2. **A Trace-Driven File System Workload Characterization of Laptops**
   Laptops now constitute a significant portion of the personal computing market and buyers are no longer exclusively computer science professionals. In this project, we studied the file system usage of average users' laptops. We used ProcMon - a monitoring tool for file systems - to capture and analyze four user traces. Our analysis of the traces showed: (1) web-browsers contribute significantly to overall file system activity, (2) file access patterns are highly polarized, and (3) most file accesses are less than 8KB in size.

3. **Compiling streaming programs to Silicon**
   In this project, we designed and implemented a framework for generating custom hardware to execute streaming applications. The framework accepts streaming applications written in StreamIt and generates synthesizable Verilog code. The Verilog code is synthesized using the Synopsis Verilog compiler to produce RTL. This is equivalent to a customized hardware that performs the function. We use bitonic sort and autocorrelation kernels for evaluation. We compared and analyzed the area and performance tradeoffs of customized hardware to a general purpose machine.

4. **Modeling Dataflow Execution in ACL2**
   In this project, we modeled in ACL2 - an automatic theorem prover - a dataflow machine and executed programs written in dataflow language. We modeled various states of the architecture as lists and use recursive functions to execute the program in dataflow manner. Simple properties about the validity of the architectural state and functions that modify them were proved. We proposed some interesting properties of program correctness, termination and bounds on execution time and execution resources, that could be proved using this model.

5. **Evolving Test Programs for Verifying microprocessor**
   The state of art in verification of processors is to use randomized test program generators (RTPG) to generate programs that tests the entire architecture. It is still a challenge to design sophisticated RTPGs for new processors since they must be highly customized for the processor to be effective. In this work, we proposed an evolutionary approach to generate high quality test programs for new dataflow architecture called TRIPS. We implemented a genetic algorithm to evolve test programs and mutation operators for the architecture. The initial results indicated that the test programs generated by our approach are as good as a customized RTPG.

6. **Profile guided top or bottom predication**
   Scale, the dataflow predicating compiler for TRIPS, can eliminate most of the fan-out of conventional predicates by predicating the top or bottom of the dependence tree. Predication at the bottom of a dependence chain can improve performance by increasing speculation and concurrency in a block but can also hurt power and performance by increasing useless work and resource contention. Predication at the top of a dependence chain can save power but might reduce performance by delaying the execution of predicated instructions. We use branch probability, pre-fetching loads and predicate fan-out minimization as compiler and showed that these heuristics are effective in reducing instruction count and improving performance for a subset of benchmarks but it is a challenge to come up with a single strategy that is effective on a broad range of benchmarks.

7. **Basic Block Architecture for Power Saving (B_2APS)**
   This project introduces a new approach towards power saving with no compromise in performance. The principal aim of this approach is to handle the entire architecture in terms of basic blocks with compiler support and restructure the underlying architecture to handle blocks and implement deterministic clock gating techniques. The proposed area of focus includes cache, register files and pipeline units and functional units.

8. Implementation Of Pthreads and OpenMP In An MPI Environment

The KABRU cluster consists of 144 nodes with 2 CPUs each. The aim of the project is to reduce communication overheads by using **Pthread/OpenMP** for intra node communication and **MPI** for inter node communication. We evaluated the system and programming model and concluded: **(1) Optimal No. of Threads**: Running with 2-4 threads is as efficient as running the program with more threads as only 2 threads are effectively scheduled to the processors and the execution of other threads stall for architectural resources**. (2) Compiler optimizations**: It was observed that OpenMP parallelization is not well integrated with the compiler optimizations of Intel C compiler. Hence, the desired performance is not achieved at –O2 level. **(3) Comparison of MPI and OpenMP**: On comparing the performance of MPI and OpenMP by running the process in i) one node with two processors and ii) multiple nodes (Mixed mode programming) , it is found that the performance of MPI and OpenMP is comparable. In most cases, the OpenMP overhead of thread creation and synchronization matches the lower latency MPI communication overhead, made possible by the high speed communication network.