

# CANS Master's Degree Program, Spring 2008-2009

## Visiting Professors' Seminars

### SEMINAR 1

Issues in Computer Architecture and Microarchitecture for Future Computing Machines (3 ECTS)

Prof. Yale N. Patt (University of Texas at Austin, USA)

### DATES:

*8<sup>th</sup> June to 17<sup>th</sup> June and 29<sup>th</sup> and 30<sup>th</sup> June 2009*

June 8<sup>th</sup>: 15:00 – 19:00

June 9<sup>th</sup> to 12<sup>th</sup>: 15:00 – 18:00

June 15<sup>th</sup> to 17<sup>th</sup>: 15:00 – 18:00

June 29<sup>th</sup>: 14:00 – 19:00

June 30<sup>th</sup>: 15:00 – 16:00

Every day in the room **C6-E101** except the 12th of June 2009 that the class will be in the room **C6-220**

### Description:

This course will consist of 30 hours, spread over the month of June. The first 25 hours will consist of interactive lectures (lectures nominally by me, but interrupted regularly by questions from students and the professor). The last day will be devoted to student presentations (on June 29), and a one hour written exam on the material in the course (on June 30). Each student will be responsible for evaluating a computer architecture problem and presenting his critical analysis to the class on June 29.

The objective in this course is to add to your foundation that will prepare you to do meaningful research in microarchitecture. That means (a) understanding at a deep level the major issues confronting microarchitects in the future, (b) thinking critically about the accomplishments of the past, and (c) putting both (a) and (b) in perspective. Given the choice of covering more material or understanding more deeply less but critical material, I will always opt for the latter.

The schedule will be as follows:

June 8: 1500-1900 Introduction and focus on the major issues.

June 9: 1500-1800 A science of tradeoffs (Architecture, Microarchitecture)

June 10: 1500-1800 Branch Prediction: Why? How? Including new results

June 11: 1500-1800 Run time: SMT, SSMT, CMP, Trace Cache, Runahead

June 12: 1500-1800 Compile time: BS-ISA, Predication, Wish branch, Braids

June 15: 1500-1800 Implications of multi-core and many-core multiprocessors

June 16: 1500-1800 Case Studies

June 17: 1500-1800 Future microprocessors -- Challenges, potential solutions

June 29: 1400-1600, 1700-1900 Paper presentations/critiques.

June 30: 1500-1600 One hour exam

If there is reason to reschedule any particular session, and we have unanimous agreement, then we will reschedule accordingly.

## SEMINAR 3

Profiling and Optimization (3 ECTS)

Prof David Kaeli (Northeastern University (Boston, USA))

### **DATES:**

May, Wed 13<sup>th</sup>: 15:00-18:00 (room C6-E106)

May, Thu 14<sup>th</sup>: 15:00-18:00 (room C6-E101)

May, Fri 15<sup>th</sup>: 9:00-12:00 (room C6-E101)

May, Mon 18<sup>th</sup>: 11:00-14:00 and 15:00-18:00(lab) (room C6-E101)

May, Tue 19<sup>th</sup>: 17:00-20:00 room (C6-E101)

May, Wed 20<sup>th</sup>: 15:00-18:00 room (C6-E106)

May, Thu 21<sup>st</sup>: 15:00-18:00 room (C6-E101)

May, Fri 22<sup>nd</sup>: 9-12(lab) C6-E101 and 15:00-18:00 C6-E106

### **Description:**

#### **Overview:**

This PhD-level course will discuss how to capture program static and dynamic characteristics of program execution using both hardware and software techniques. The course will also discuss how to utilize these characteristics in the areas of architectural modeling, program optimization, binary translation, power modeling, software fault recovery, security, software testing, and runtime monitoring. This year's course will also focus on issues of security, and will run experiments on X86 and embedded processors.

The subjects that will be covered in the course include:

a.) A historical perspective of profiling and instrumentation

b.) Instrumentation

Hardware techniques

- Performance counters
- ICE and pods
- Trace arrays
- Bus analyzers
- LLATTs

Software techniques

- source modifications
- object modifications
- executable modifications
- emulators

c.) Applications

- Performance

- o Hardware/architecture
    - o Software
  - Recovery
    - o Trace buffers
    - o Dumps
  - Security
    - o Anti-virus scanning
    - o Trojan horses
    - o Spam filtering
  - Testing
    - o Clear-box testing
  - Power modeling
    - o Instruction-based modeling
  - Monitoring
  - JITs/Binary Translators
- d.) Optimization
- Static
  - Dynamic
- e.) Selected topics
- Hot-cold code layout
  - Heap layout
  - Cloning and Specialization
  - Aspect-oriented programming/profiling
  - Power/energy efficiency
  - Anti-virus studies
  - Def-use testing
  - Cachability
- f.) software/hardware interface - how does software interface to hardware
- g.) program debugging and testing - tools and strategies

## Course Details

### Homework:

There will be one lab assignment that will ask you to run an experiment on a profiling system. You will have the choice of using either the Pin toolset or the ATOM toolset. The goal is to use a tool to obtain interesting runtime information about a program.

### Paper Review:

You need to submit a review of a selected paper during the course. The review will consist of a 1-page summary of the material presented in the selected paper, followed by 2-3 pages of critical opinion of the material presented. Your review should rigorously address the following questions:

1. What is the problem being studied?
2. Is this an important problem?
3. What are the main points or results presented?
4. What quantitative methods were used in this paper? Were these the best methods to be used?
5. What did you learn from this paper?

6. Compare this paper to similar papers on the subject? (you should find and read one or two related papers, and use them to compare to this paper)  
In the last class, you will be asked to stand up and discuss your paper review with the class.

**Grading:**

Homework: 40%

Paper review: 40%

Class discussion: 20%

**Topics to be covered:**

Class 1a: Introduction

Course overview

Definitions of common terminology

Historical perspective

Issues with profiling: overhead, completeness and correctness

Class 1b: Tools

Papers – ATOM, Simics, Pin, Liberty, DCPI

Review of tools and techniques

Hardware Tools, Performance Counters

Class 2: Workload Characteristics

Papers – OO-Calder, Values, Uhlig, AS/400

Instruction Profiling, Memory Profiling, VisualDSP++

Class 3: Sampling Technology, AOP

Papers – Profile-Me, Trace Stitching,

Sampling theory

Sampled workloads during simulation

Incomplete profiles

Class 4: Profile-guided Compilation

Papers – PLDI97, WMPI01, PLTO, IOICS03

Program graphs (PCG, TRG)

Code layout, Data layout

Instruction layout

Class 5: Static Optimization

Papers – Ball/Larus, Static Branch Prediction

Code Layout

Class 6: Static/Dynamic Optimization

Papers – Shade, Spike, FX!32

Overhead

Optimization techniques

Class 7: Dynamic Optimization  
Papers – Dynamo, JITs, replay

Class 8: Translation, Software Testing  
Papers – Dixie, UQBT, Def-Use, Dataflow

Class 9: Power Profiling, Security  
Paper Review Discussion

## **SEMINAR 8**

HIPEAC Summer School Seminars (3 ECTS)

Local organizer contact: Prof. Nacho Navarro (UPC)

ACACES 2009

Fifth International Summer School on Advanced Computer Architecture and  
Compilation for Embedded Systems

July 12<sup>th</sup> to July 18<sup>th</sup> 2009 L'Aquila, Italy

Organized by the HiPEAC Network of Excellence

<http://www.hipeac.net/summerschool/>

## SEMINAR 15

### Fault Tolerant Computing (3 ECTS)

Prof. Israel Koren (University of Massachusetts at Amherst, USA)

#### **DATES:**

*18<sup>th</sup> to 29<sup>th</sup> May 2009*

Monday, Wednesday and Thursday: 9:00 – 11:00.

Tuesday: 15:00 – 17:00.

Friday: 12:00 – 14:00.

Friday May 22<sup>nd</sup>: Also 16:00 – 18:00.

Thursday May 28<sup>th</sup>: Also 15:00 – 18:00.

Every day the room **C6-E101**, except 19/05/2009 and 26/05/2009 that will be in room **C6-E106**. The day 22/05/2009 the class will be in room **C6-220**.

#### **Description:**

The purpose of the short course is to provide a solid introduction to the rich field of fault-tolerant computing. With the increasing complexity of computing systems and the reliance on their error-free operation, fault tolerance techniques are necessary to guarantee high levels of reliability and availability. Therefore, a course covering the principles of fault tolerant computing is a must for computer engineers, software developers and system designers.

The students taking the course are expected to have a basic knowledge of computer organization, principles of software development, and probability theory.

The course will cover the following topics:

1. Introduction - Types of faults, redundancy techniques and measures of fault tolerance.

1 hour

2. Hardware fault-tolerance - Failure rates, reliability and availability, resilient hardware structures, Markov models, processor-level techniques.

6 hours

3. Information redundancy - Error detecting and correcting codes and their use in contemporary memory circuits, arithmetic circuits and storage systems.

Information redundancy in distributed systems and algorithm-based fault-tolerance.

6 hours

4. Fault-tolerant networks - Resilience measures for networks, distributed network topologies and their resilience, fault tolerant routing.

4 hours

5. Software fault-tolerance - Acceptance tests, single and multiple version fault tolerance, recovery blocks, software reliability models.

4 hours

6. Checkpointing - Single processor checkpointing, distributed systems checkpointing, other uses of checkpointing.

3 hours

7. Case studies - Real-life examples of fault-tolerant systems illustrating the usage of the previously covered fault tolerance techniques.

3 hours

8. Defect-tolerance in VLSI circuits - Manufacturing defects, critical area, yield models, yield enhancement techniques for memory and logic.

3 hours

9. (Time permitting) Fault detection in cryptographic devices - Symmetric and public key cyphers, security attacks through fault injection, countermeasures against these attacks.

4 hours