BEAST Lab Preliminary Meeting

LRZ
Dr. Josef Weidendorfer, josef.weidendorfer@lrz.de

LMU
Minh Thanh Chung, minh.thanh.chung@ifi.lmu.de
Dr. Karl Fürlinger, karl.fuerlinger@ifi.lmu.de

TUM
Vincent Bode, vincent.bode@tum.de,
Dennis-Florian Herr herrd@in.tum.de
Bengisu Elis, bengisu.elis@tum.de
Table of Contents

Course Organization

Introduction to BEAST
Weekly Schedule

Lab Meetup
15:00-18:00
+ Assignment Presentations
+ Theory/Vendor Presentation
+ Assignment Introduction
Weekly Schedule

Lab Meetup
15:00-18:00
+ Assignment Presentations
+ Theory/Vendor Presentation
+ Assignment Introduction

Assignment/Project

Mon Tue Wed Thu Fri Sat Sun
Weekly Schedule

Announcement (E-Mail)
2-3 groups chosen to present

Lab Meetup
15:00-18:00
+ Assignment Presentations
+ Theory/Vendor Presentation
+ Assignment Introduction
Tentative Semester Overview

Weeks

1. Triad
2. MatMul
3. Caching
4. GPUs
5. GPUs 2
6. Profiling
7. Project 1
8. Project 2
9. 28.4
10. 5.5
11. 12.5
12. 19.5
13. 26.5
14. 2.6
15. 9.6
16. 16.6
17. 23.6
18. 30.6
19. 7.7
20. 14.7
21. 21.7

Organization

- Note: This is preliminary based off of last semester and is subject to improvements
- 6 Assignments
  - 1 week each (except on holidays)
- 2 bigger Projects
  - 2 weeks each
- Student groups of 3 (Bachelor) or 2 (Master)

Previous Vendor Talks

V. Bode, D. Herr, B. Elis (TUM)
Deliverables/Grading

Git Repository
- **Assignment/Project Report** in Markdown
- Your Code
- CI Jobs (not graded)

Presentation
- No slides. Go through the report
- Talk about what you learned
- Get feedback from advisors
Next Steps

Register on Matching System

• We will prioritize you if you attended today
• Open until 15.02.2022
• Wait for announcement of matching results (24.02.2022)

Group Preferences

• Only after matching has ended
• Send us by e-mail (bengisu.elis@tum.de)
• No preferences submitted → we will match you

Attend Course Kickoff

• At university if everything goes according to plan
• We hope to see you there :)
Up Next: Introduction to BEAST
Collaboration among 3 institutions

LMU
TUM
LRZ

TUM – CAPS/Prof. Schulz
(Bengisu Elis, Vincent Bode)

LRZ - Future Computing Group
(Josef Weidendorfer)
Focus: Experimental Evaluation

We want you to learn about performance properties of modern architectures
• Be able to understand and explain performance effects seen from measurements
• Get a deeper understanding of current system designs (CPU / GPU)

Part 1: get started with small codes across systems
• We show key hardware design concepts + a parallel programming model (OpenMP)
• We give you typical small HPC code examples
• You run measurements of different scenarios across systems, compare / discuss results
• We all discuss results in the weekly meetings, from presentations of 2 groups

Structure:
Memory on CPU (Triad / Traversal) ➔ Compute on CPU (MM) ➔ … on GPU ➔ Tools
Focus: Experimental Evaluation

We want you to learn about **performance properties of current architectures**

- Be able to understand and explain performance effects seen from measurements
- Get a deeper understanding of current system designs (CPU / GPU)

Part 2: make use of gained knowledge

- We assign randomly one system to each group
- We give you some larger typical HPC code examples
- You tune the code to get best single-node performance (2 weeks time)
- We all discuss results in the weekly meetings
Evaluation of Single-Node Performance

Target Architectures for the Lab

CPUs
• Intel Icelake (ISA: x86-64 + AVX512)
• AMD Rome (ISA: x86-64 + AVX2)
• Marvell ThunderX2 (ISA: ARM AArch64 + Neon)
• Fujitsu A64FX (ISA: ARM AArch64 + SVE)

GPUs
• NVidia V100
• AMD MI-50
Organization

- Work in student groups
  - we expect you to split up the work equally
- Assignments
  - at start every week, later more time
  - code / reports (MarkDown) via Gitlab repos, CI feedback when it makes sense
- Weekly meetings (Thursday afternoon)
  - talks around assignment tasks (microarchitecture, parallel prog. models, …)
  - student group presentations for every assignment (randomly selected)
  - discussions around results
Prerequisites

• Good knowledge of C (C++) on Linux
• Basic knowledge of computer architecture. You should know terms such as
  • Multi-core, L1/L2/L3 caches, TLB, pipelining, SIMD, SMT
• Interest in computer architecture, benchmarking, low-level code optimization