Chair of Computer Architecture and Parallel Systems
Department of Informatics
Technical University of Munich

BEAST Lab Preliminary Meeting

LRZ

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

LMU

Sergej Breiter, sergej.breiter@nm.ifi.lmu.de

Minh Thanh Chung, minh.thanh.chung@ifi.lmu.de

Dr. Karl Fürlinger, karl.fuerlinger@ifi.lmu.de

TUM

Bengisu Elis, bengisu.elis@tum.de

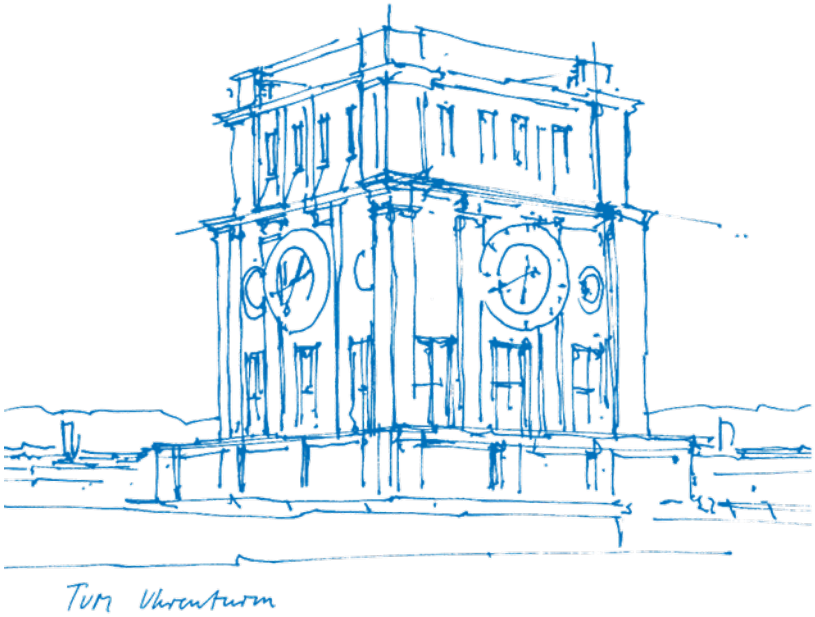
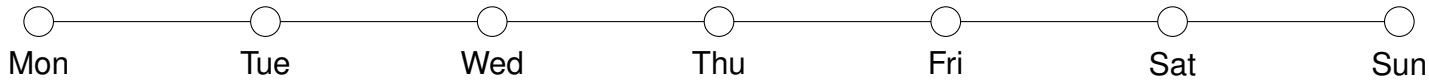


Table of Contents

Course Organization

Introduction to BEAST

Weekly Schedule

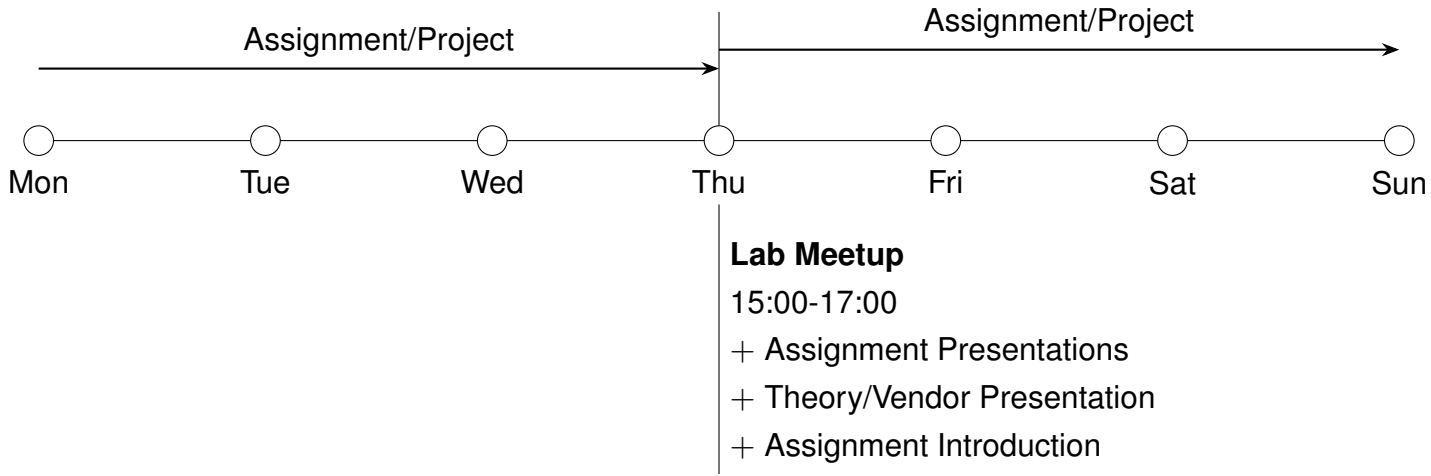


Lab Meetup

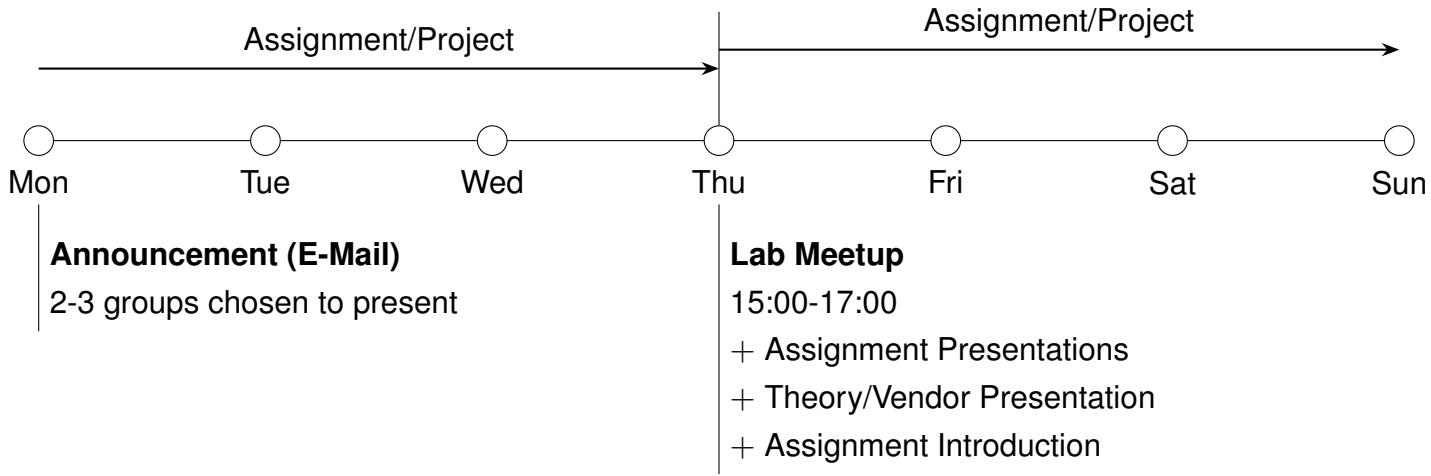
15:00-17:00

+ Assignment Presentations
+ Theory/Vendor Presentation
+ Assignment Introduction

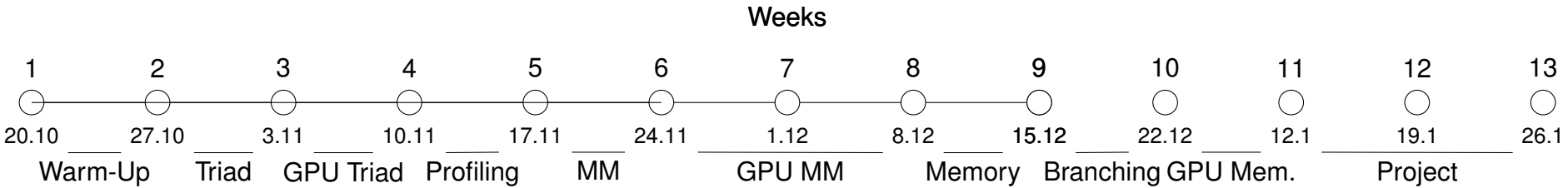
Weekly Schedule



Weekly Schedule



Tentative Semester Overview - Last Year



Organization

- Note: This is preliminary based off of last semester and is subject to improvements
- 8 Assignments
- Assignment 0 (of today) is not optional
  - 1 week each (except on holidays)
- Project
  - 2 weeks
  - Student groups of 3 (Bachelor) or 2 (Master)

Previous Vendor Talks

AMD, intel, arm, FUJITSU, CRAY a Hewlett Packard Enterprise company, MARVELL

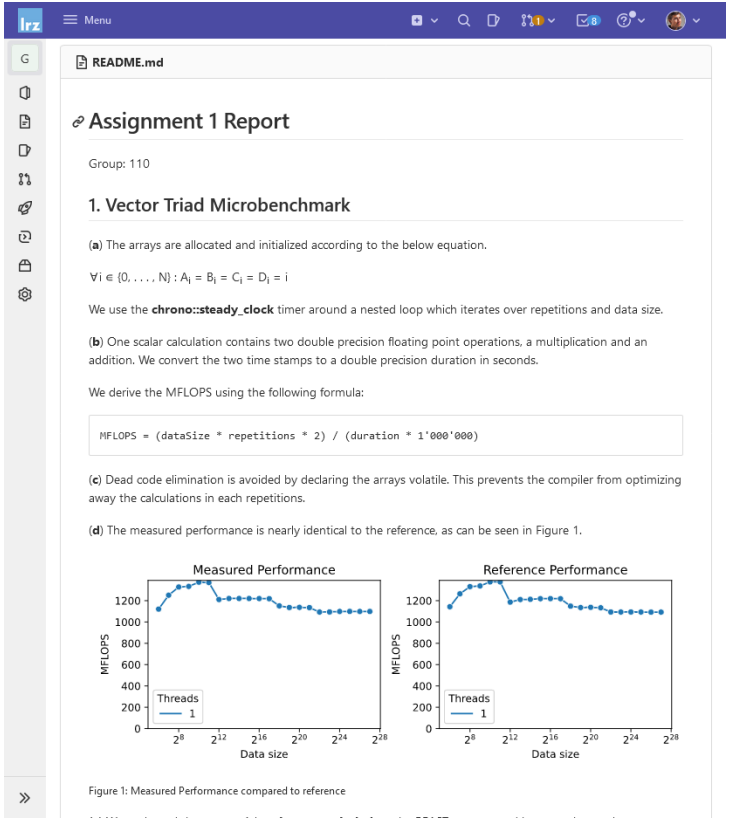
Deliverables/Grading

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

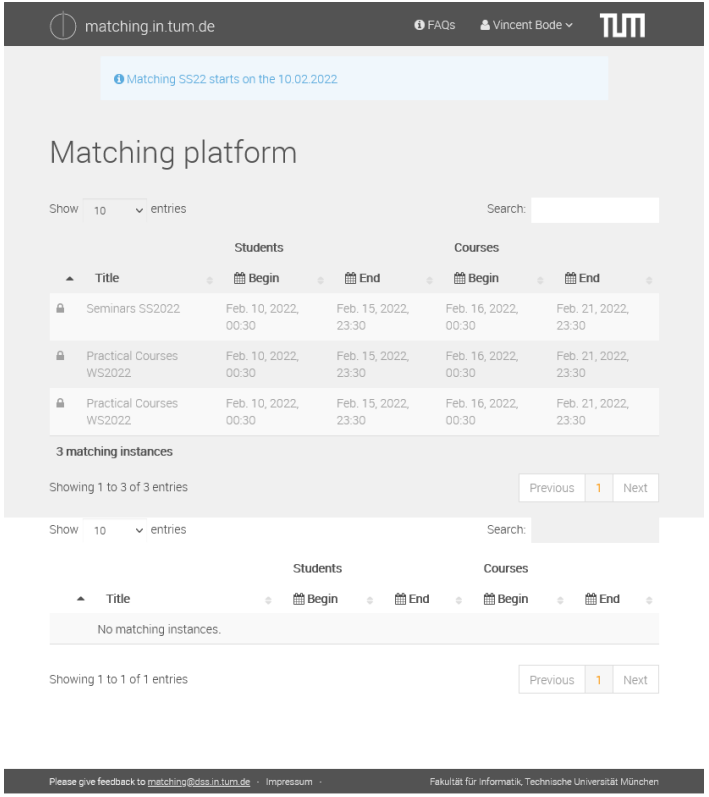
- We will prioritize you if you attended today
- Open until 14.02.2023
- Wait for announcement of matching results (23.02.2023)

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

BEAST Lab Summer Term 2023

Preliminary Meeting

February 9, 2023

Collaboration among 3 institutions

LMU
TUM
LRZ

TUM – CAPS/Prof. Schulz
(Bengisu Elis)

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
- You tune the code to get best single-node performance (2 weeks time)
- We all discuss progress in weekly meeting

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-100

Organization

- Work in student groups
  - we expect you to split up the work equally
- Assignments
  - every week (Part 2: 2 weeks)
  - code / reports (MarkDown) via Gitlab repos, partially CI feedback in the beginning
- 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