

# Testing thermal-aware scheduling for avionics applications

OSPM, 2020-05-12

**Michal Sojka**, Czech Technical University in Prague  
(**Cláudio Maia**, Instituto Superior de Engenharia do Porto)



This project has received funding from the Clean Sky 2 Joint Undertaking under the European Union's Horizon 2020 research and innovation programme under grant agreement No 832011.

# THERMAC project overview

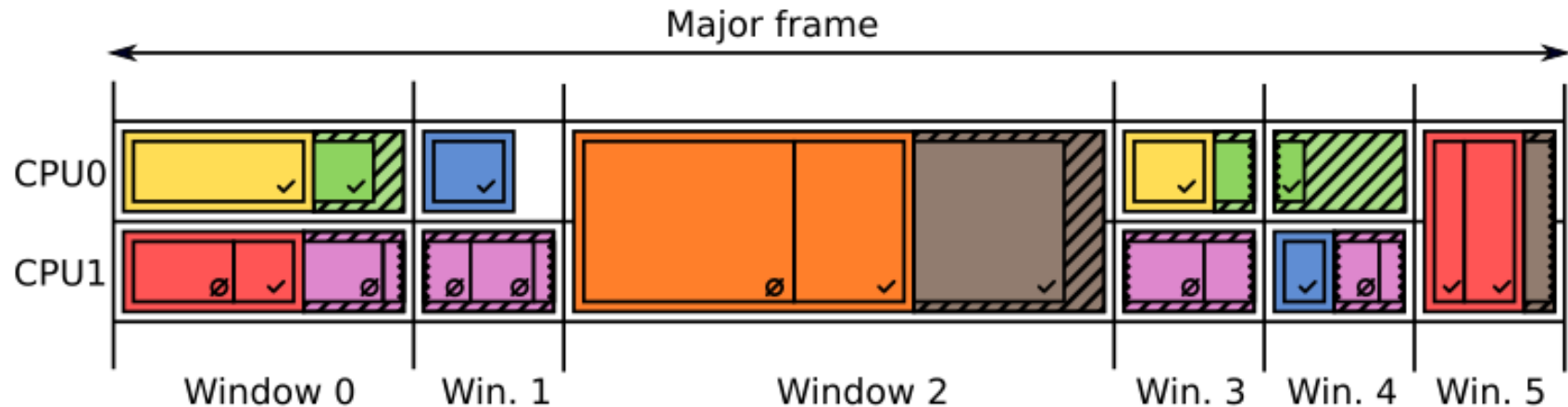


- Collaboration with avionics company
- Their requirements
  - **embedded** systems
  - no **fans**, no heavy **heat sinks**
  - as much **performance** as **thermal** constraints allow
  - no deadline misses for **safety critical** workload
  - **best effort** workload runs when **temperature** and **time** allows



# Operating system & scheduling

- Safety-critical avionics applications **cannot use Linux**
- Special-purpose OSES with time-partitioned scheduling
- We simulate such scheduling on Linux via cgroups
- Safety-critical workload is **well known** in advance



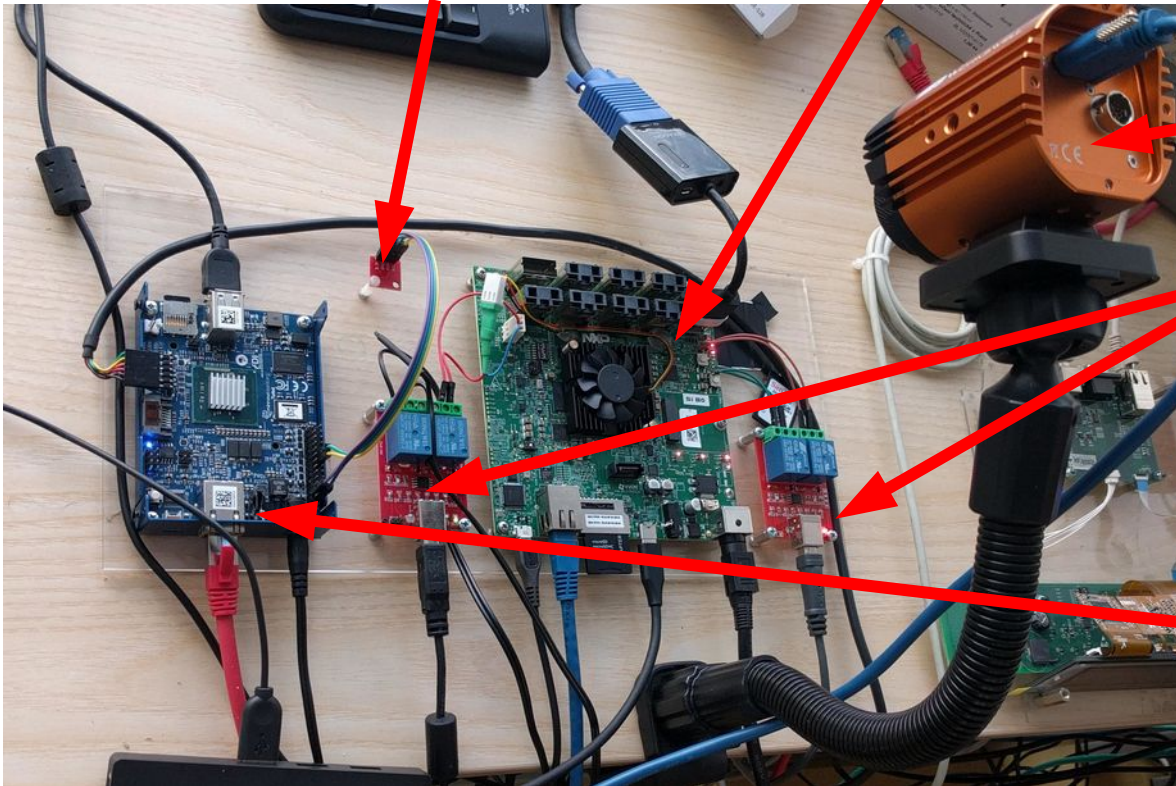
# Goals

- Prepare methods and benchmarks to evaluate thermal-aware scheduling strategies
- Measure thermal properties of the HW
  - **NXP i.MX8 QuadMax**, NVIDIA Tegra X2/Xavier
- Come up with thermal-aware scheduling strategies to control the peak temperature
- Evaluate the scheduling strategies

# Testbed

Board under test  
**NXP i.MX8 QuadMax MEK**  
4× Cortex A53, 2× A72, 2× GPU

Ambient temperature sensor

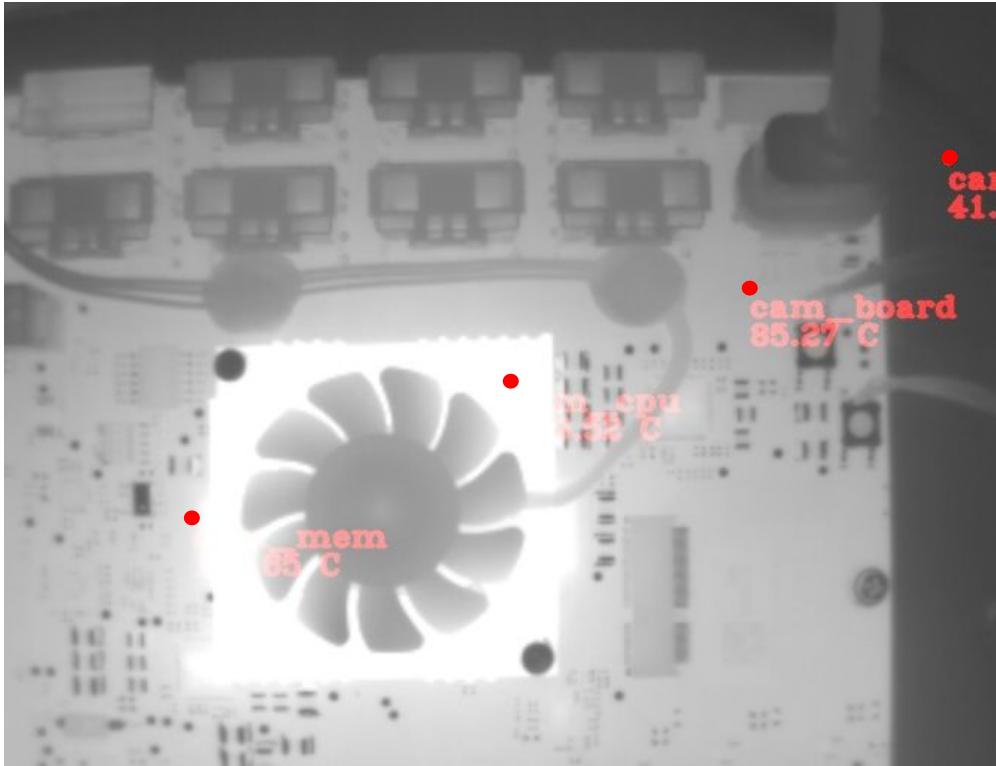


Thermal camera

Relays for remote control and automation

MinnowBoard – management

# Thermo camera



- Measures temperature of various points on the board
- 336×256px, 9 Hz
- Results over HTTP

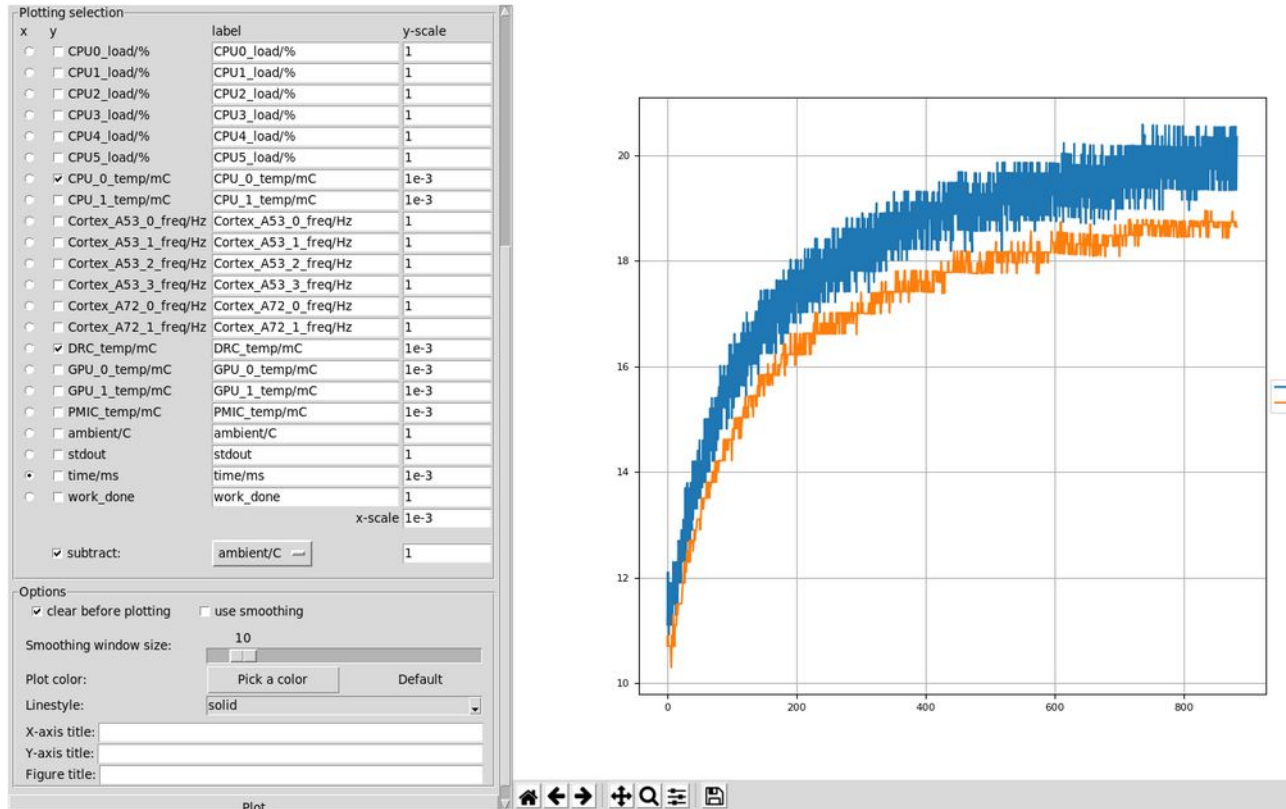
# Thermobench tool

<https://github.com/CTU-IIG/thermobench>

- Fancy .csv file generator (C++)
- Controls cooling devices (fan)
- Runs a benchmark
- Periodically captures
  - thermal-zone temperatures
  - CPU frequency
  - CPU load
  - benchmark stdout
  - other commands stdout (thermocam, ambient temp.)
  - etc.
- GPL

The screenshot shows the GitHub repository page for `CTU-IIG / thermobench`. The repository has 175 commits, 18 branches, 0 packages, 0 releases, and 6 contributors. The main branch is `master`. A list of files is displayed, including `benchmarks`, `data_visualizer`, `report_gen`, `src`, `subprojects`, `test`, `.clang-format`, `.dir-locals.el`, `.gitmodules`, `Makefile`, `README.md`, `aarch64.txt`, `meson.build`, `thermobench.cflags`, `thermobench.config`, `thermobench.creator`, `thermobench.cxxflags`, `thermobench.files`, and `thermobench.includes`. The `README.md` file is selected, showing the title `Thermobench` and the section `Prerequisites`. The prerequisites section states: "We use the Meson build system. On Debian-based distro, it can be installed by:".

# Interactive data visualizer (Python)

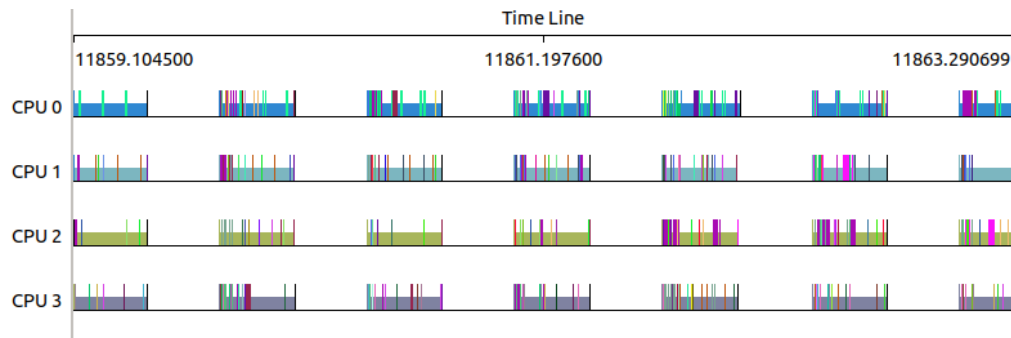
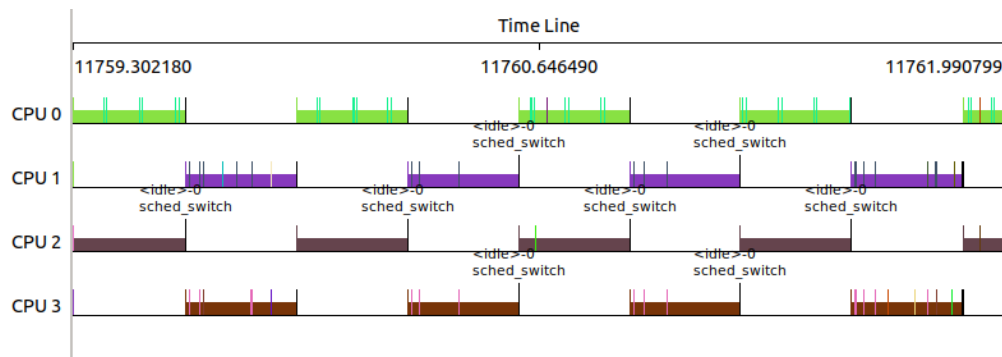
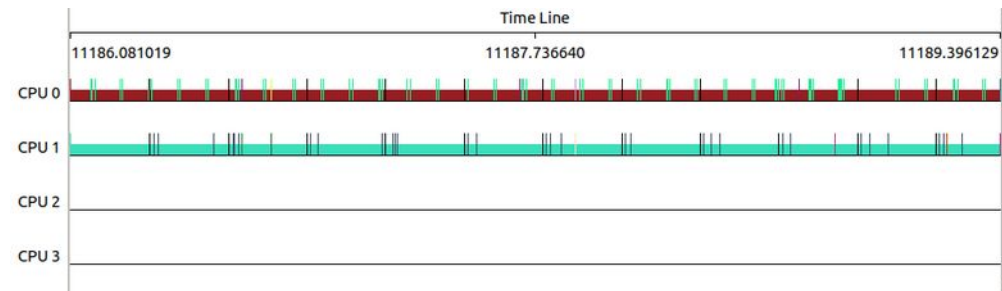


- Easily plot data produced by Thermobend
  - temp. vs. time
  - temp. vs. work\_done
  - subtract ambient temperature
  - and more...



# Benchmarks

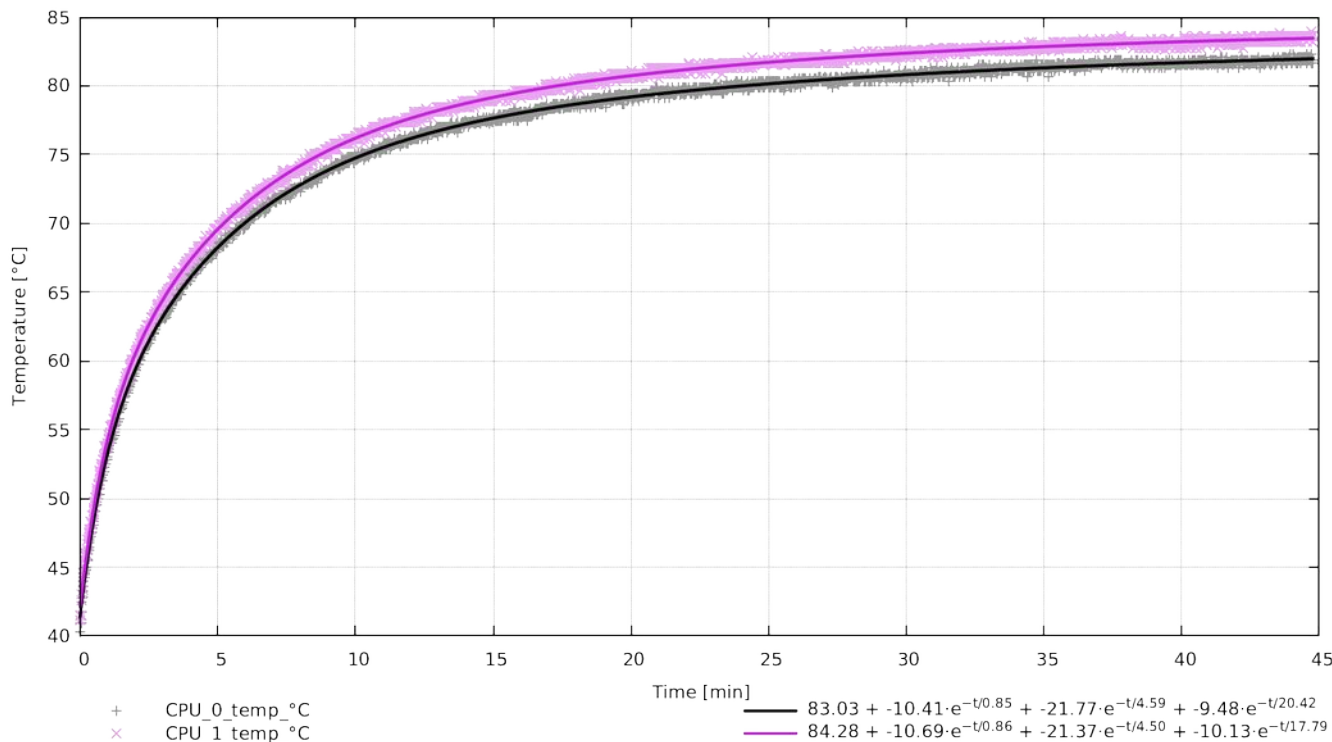
- Instruction  $\mu$ -benchmarks
- Various sleep patterns
- Memory-/CPU-bound
- etc.



# Thermal model fitting in Julia

$$84.28 - 10.69 \cdot e^{-\frac{t}{0.86}} - 21.37 \cdot e^{-\frac{t}{4.50}} - 10.13 \cdot e^{-\frac{t}{17.79}}$$

/home/wsh/thermac/devel/experiments/cl-mem/cl-mem-read.csv\*

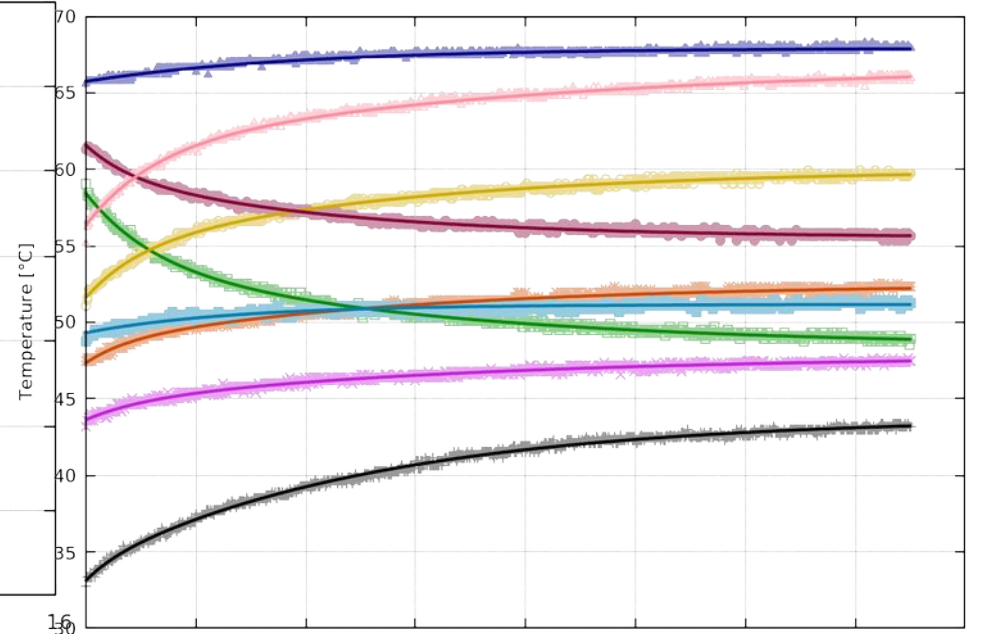
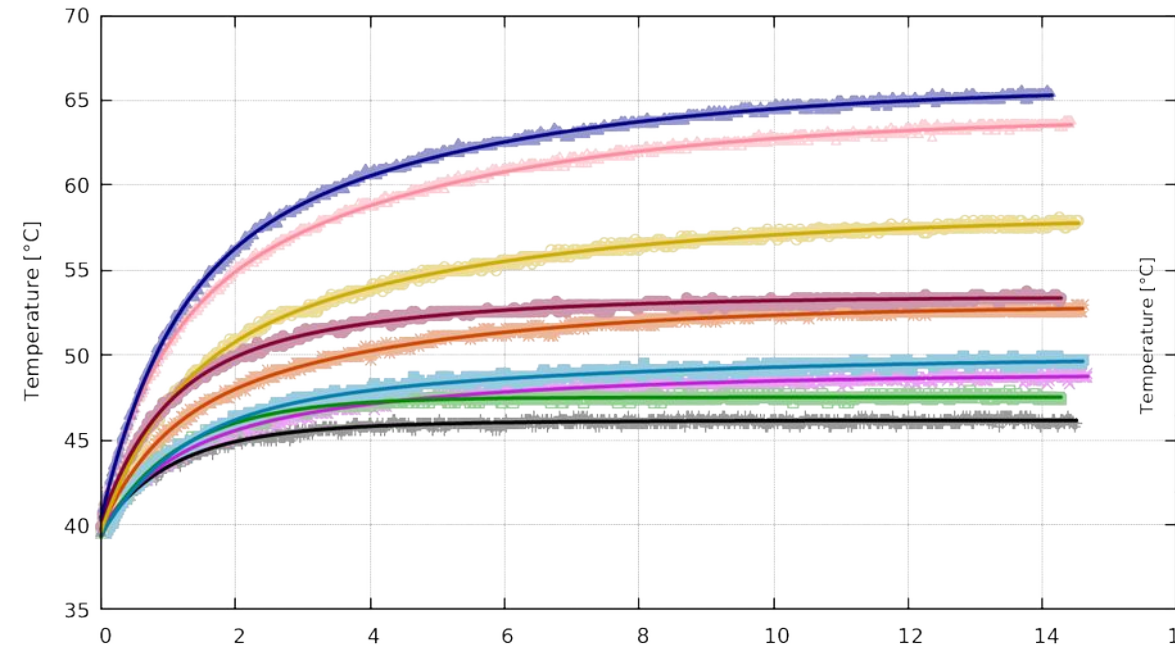


- Fits n-th order dynamic system model to data from Thermobench
- $n=1,2,3$
- 1<sup>st</sup> coeff. =  $T(\infty)$
- Coefficients in exponents = “time constant”
  - How long to run experiments?

# Comparison of various experiments

memory-bandwidth/data-fan/seq-a\*

memory-bandwidth/data-nofan/seq-a\*



- + \*53-t1-s16k.csv: CPU\_0\_temp\_°C
- × \*53-t1-s256k.csv: CPU\_0\_temp\_°C
- ✱ \*53-t1-s4M.csv: CPU\_0\_temp\_°C
- \*72-t1-s16k.csv: CPU\_0\_temp\_°C
- \*72-t1-s256k.csv: CPU\_0\_temp\_°C
- \*72-t1-s4M.csv: CPU\_0\_temp\_°C
- \*ll-t6-s16k.csv: CPU\_0\_temp\_°C
- △ \*ll-t6-s256k.csv: CPU\_0\_temp\_°C
- ▲ \*ll-t6-s4M.csv: CPU\_0\_temp\_°C

Time [min]

- $46.14 + -5.00 \cdot e^{-t/0.97} + -1.43 \cdot e^{-t/2.40}$
- $48.85 + -4.63 \cdot e^{-t/0.95} + -4.49 \cdot e^{-t/4.07}$
- $52.90 + -5.74 \cdot e^{-t/0.83} + -7.36 \cdot e^{-t/3.86}$
- $47.51 + -0.60 \cdot e^{-t/0.34} + -7.61 \cdot e^{-t/1.24}$
- $49.90 + -6.88 \cdot e^{-t/1.13} + -3.63 \cdot e^{-t/5.68}$
- $58.18 + -8.84 \cdot e^{-t/0.96} + -9.74 \cdot e^{-t/4.60}$
- $53.40 + -7.65 \cdot e^{-t/0.83} + -5.49 \cdot e^{-t/3.03}$
- $64.00 + -10.27 \cdot e^{-t/0.81} + -13.19 \cdot e^{-t/4.24}$
- $65.83 + -12.92 \cdot e^{-t/0.95} + -12.48 \cdot e^{-t/4.47}$

- + \*53-t1-s16k.csv: CPU\_0\_temp\_°C
- × \*53-t1-s256k.csv: CPU\_0\_temp\_°C
- ✱ \*53-t1-s4M.csv: CPU\_0\_temp\_°C
- \*72-t1-s16k.csv: CPU\_0\_temp\_°C
- \*72-t1-s256k.csv: CPU\_0\_temp\_°C
- \*72-t1-s4M.csv: CPU\_0\_temp\_°C
- \*ll-t6-s16k.csv: CPU\_0\_temp\_°C
- △ \*ll-t6-s256k.csv: CPU\_0\_temp\_°C
- ▲ \*ll-t6-s4M.csv: CPU\_0\_temp\_°C

Time [min]

- $43.78 + -0.91 \cdot e^{-t/0.51} + -9.74 \cdot e^{-t/5.22}$
- $47.80 + -1.09 \cdot e^{-t/1.01} + -3.09 \cdot e^{-t/6.73}$
- $52.49 + -1.27 \cdot e^{-t/0.72} + -3.88 \cdot e^{-t/5.66}$
- $48.44 + 4.32 \cdot e^{-t/1.17} + 5.67 \cdot e^{-t/5.92}$
- $51.17 + -54.55 \cdot e^{-t/2.72} + 52.71 \cdot e^{-t/2.72}$
- $60.00 + -2.95 \cdot e^{-t/0.96} + -5.44 \cdot e^{-t/5.43}$
- $55.53 + 1.85 \cdot e^{-t/0.78} + 4.19 \cdot e^{-t/4.29}$
- $66.84 + -4.83 \cdot e^{-t/1.22} + -5.66 \cdot e^{-t/7.11}$
- $67.91 + 6.51 \cdot e^{-t/3.85} + -8.68 \cdot e^{-t/3.85}$

# Feedback/Discussion

- Can this be useful for something else?
  - Linux PM/scheduler work?
- Preliminary results:
  - Execute on all CPUs in parallel + sleep
  - Execute on a small part of GPU continuously, let the rest of the GPU sleep
- What other benchmarks would you find useful?
- Power measurements vs. temperature measurements
- Experience with oil-based cooling