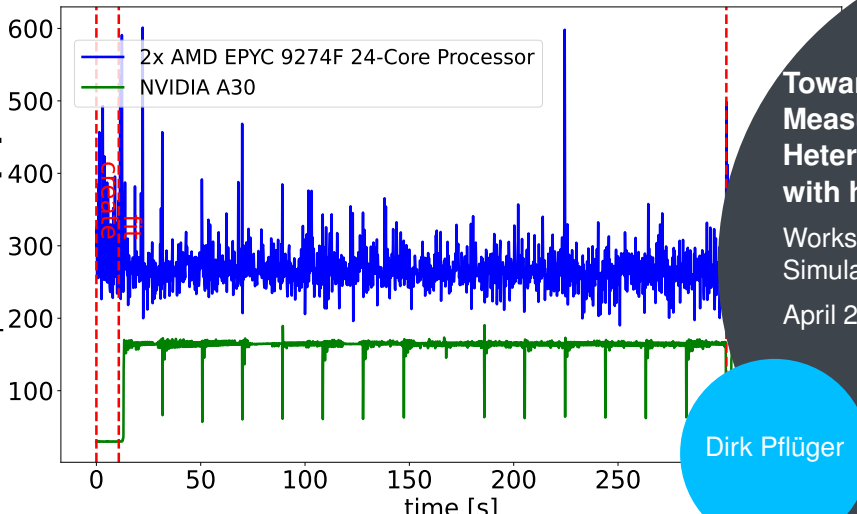
University of Stuttgart
Germany



Towards Energy-Aware HPC: Measuring Efficiency Across Heterogeneous Hardware with hws

Workshop on Sustained Simulation Performance

April 24, 2026

Dirk Pflüger

Scientific Computing at IPVS

HPC

- **Algorithms and Methods**
  - Scalability and parallel efficiency
  - ABFT and resilience
  - Education and training
  - Parallelization of applications
  - Superfacility study
- **Performance Portability**
  - Heterogeneous computing
  - GPU and accelerator integration
  - SYCL, OpenCL, (HIP) and oneAPI
- **Task-Based Parallelism**
  - HPX and Kokkos
  - Dynamic load balancing

Further Topics

- **Scientific Machine Learning**
  - PINNs
  - Data-driven methods
  - LLMs for hardware testing
- **High-Dimensional Approximation**
  - UQ and surrogate modeling
  - Sparse Grids
  - Rare events
- **Music & Sleep :-)**
  - Personalization
  - Applied ML

Why energy-aware HPC?

Core drivers

- Time-to-solution ⇒ Energy-to-solution
- System cost ⇒ Lifecycle cost
- Sustainability
  ⇒ Environmental impactrechten,
- Exascale constraints
  ⇒ Power constraints

Triggering scenario

- First and only multi-node multi-vendor Support Vector Machine: PLSSVM
- . . . and fair comparison to DNNs

Why is measurement hard?

- Heterogeneous hardware
- Vendor-specific tools
- No standardized CPU/GPU interface
- Limited access to data
- Overhead and accuracy

Challenges and a hws Solution

General Task

- gather information at runtime

Challenges

- gather various information (not only energy)
- high-level overview
- support for all major hardware platforms
- easy-to-use, uniform counters

```
1  import HardwareSampling as hws
2  import tensorflow as tf
3
4  sampler = hws.SystemHardwareSampler()
5  sampler.start()
6
7  sampler.add_event("init")
8  A = tf.random.uniform((2**14, 2**14))
9  B = tf.random.uniform((2**14, 2**14))
10
11 sampler.add_event("matmul")
12 C = A @ B
13
14 sampler.stop()
15 sampler.dump_yaml("samples.yaml")
```

Comparison to Other Libraries

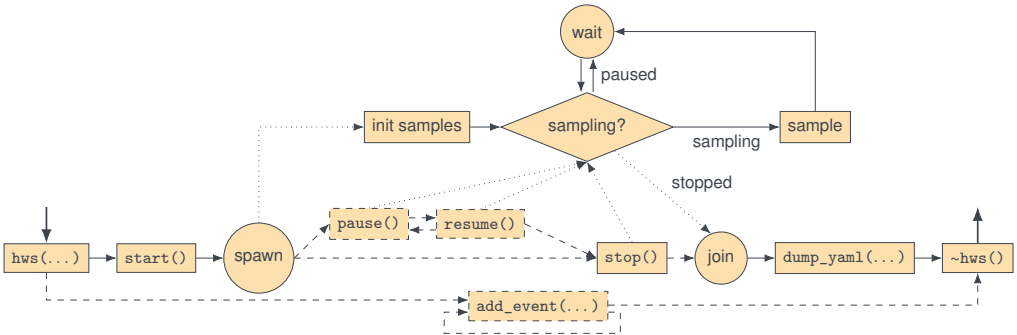
	NVIDIA GPUs	AMD GPUs	Intel GPUs	x86 CPUs	ARM CPUs	Power9 CPUs	C++	Python	autom. sampling
PAPI	✓	✓	✗	✓	✓	✓	✓	○ (unof.)	○ (marker API)
likwid	✓	✓	✗	✓	✓	✓	✓	✓	○ (marker API)
variorum	✓	✓	✓	✓	✓	✓	✓	✓	✗
hws	✓	✓	✓	✓	○ (mem only)	○ (mem only)	✓	✓	✓

hws: A Hardware Sampling Library

- open-source available at
  `https://github.com/SC-SGS/hardware_sampling`
- any combination possible: sample CPU and (multi-)GPU information simultaneously
- custom events to mark specific points in the program execution
- C++ API with Python bindings
- sampling interval can be changed
- can easily save samples as YAML file
- selectively only sample specific categories
- support for all major hardware platforms:
  - CPUs via turbostat, lscpu, and free
  - NVIDIA GPUs via NVML
  - AMD GPUs via ROCm SMI lib
  - Intel GPUs via Level Zero
- Single heterogeneous node

GitHub repo

How Does the Sampling Process Work?



Case Study: Semiconductor Process Node Comparison with hws

Application Scenario

Gaussian Process Regression (GPR)

- Non-parametric Bayesian ML
- Key computational kernel:
  Cholesky decomposition of large dense matrix

Compute-bound solver

- Portable implementation via SYCL
- GPU acceleration of FP64 Cholesky decomposition with SYCL
- Comparison of energy efficiency across different GPU architectures and semiconductor process nodes

Alternative: memory-bound CG

General

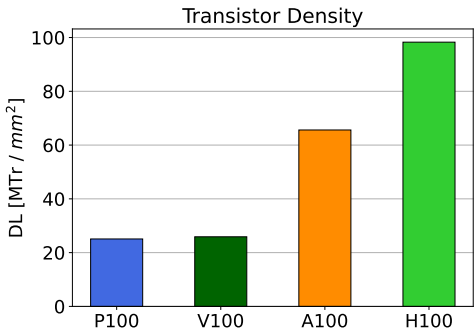
- Comparison of all NVIDIA top-tier datacenter GPU models 2016–2022:
  - P100 (Pascal $\approx$ 2016)
  - V100 (Volta $\approx$ 2018)
  - A100 (Ampere $\approx$ 2020)
  - H100 (Hopper $\approx$ 2022)
- Workload:
  GPU-only FP64 Cholesky decomposition of 60,000$\times$60,000 SPD matrix
- Watt and Energy (where possible) measurements with hws
  - P100: calculation of energy from Watt samples
- Results averaged over 20 runs

GPU Specs

	P100 PCIe 16GB	V100 SXM2 32GB	A100 PCIe 80GB	H100 SXM5 94GB
Architecture	Pascal	Volta	Ampere	Hopper
Year (architecture)	$\approx$ 2016	$\approx$ 2018	$\approx$ 2020	$\approx$ 2022
FP64 TFLOPS	5.3	7.8	9.7	34
Memory	16GB HBM2	32GB HBM2	80GB HBM2e	94GB HBM3
TDP	250W	300W	300W	415W
TSMC Process	16nm FinFET	12nm FinFET NVIDIA (FFN)	7nm N7	4N (NVIDIA-customized)
Die size	610 mm^2	815 mm^2	826 mm^2	814 mm^2
Transistors (billion)	15.3	21.1	54.2	80
DL	25.08 MTr / mm^2	25.89 MTr / mm^2	65.62 MTr / mm^2	98.28 MTr / mm^2

A100 and H100 at bwUniCluster

GPU Specs - Density of Logic Transistors (DL)



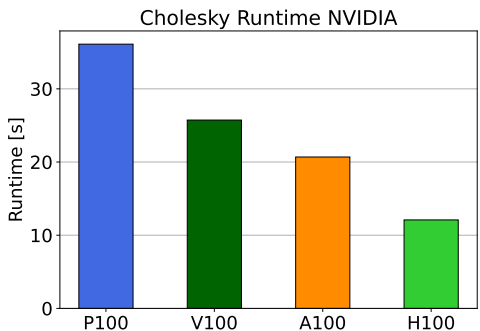
Fabrication process

- TSMC process names raise expectations
- Closer look: P100 and V100 more similar than implied

Density of Logic Transistors (DL)

- Better metric to capture architectural advancements of different semiconductor processes

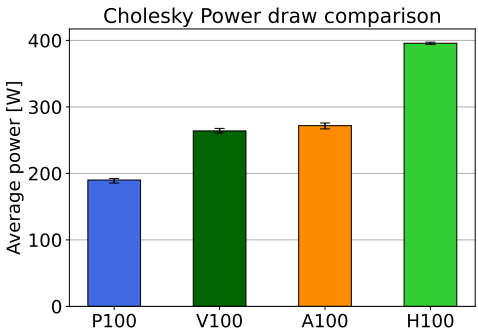
Cholesky Results - Runtime



Runtime

- Uniform SYCL implementation, no tensor cores
- GPU-only load
- Performance gain essentially as expected

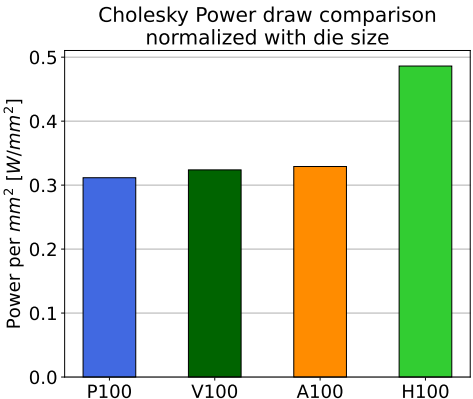
Cholesky Results - Power Draw

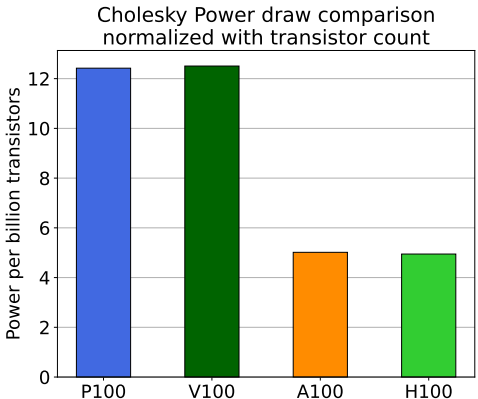


Power draw

- Disclaimer: Exact values depend on GPU configuration; here: H100 with 415W TDP
- Error bars: minimum and maximum observed average power observed for the 20 runs

Cholesky Results - Normalized Power Draw





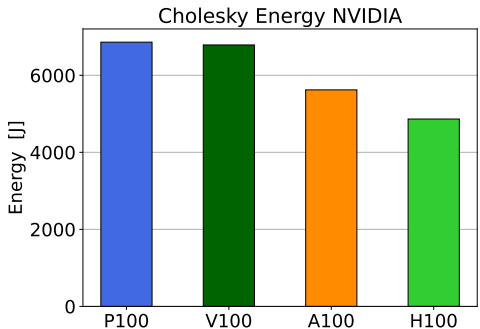
- Normalized to compensate for growth of GPUs

Cholesky Results - Power Draw

Observations for normalized power draw

- Absolute power draw increased from P100 to V100
- However, die size also increased
  - Power per mm^2 similar
  - Power per billion transistors similar, too!
  - Indicates no real architectural improvement has happened between the two manufacturing processes (as opposed to what the TSMC name suggests)
- Different picture comparing P100 and A100: architectural improvement

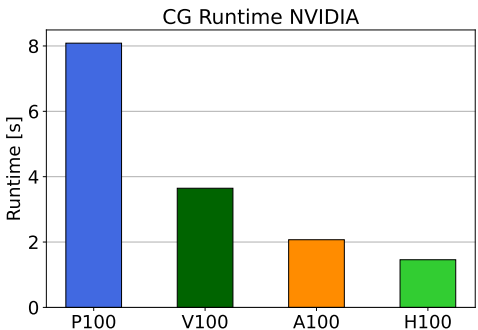
Cholesky Results - Total Energy Consumption



Total energy consumption

- Confirmed: newer GPUs more energy efficient
- Evolution roughly correlates inversely to density of logic transistors (DL)
- P100 and V100 have a similar DL and energy consumption

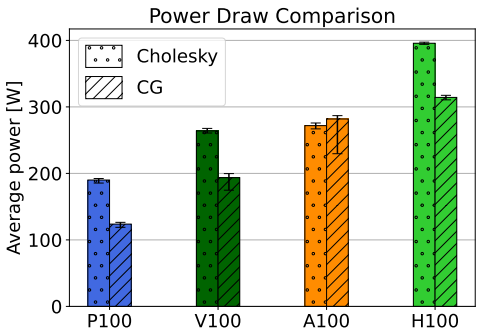
CG Results - Runtime Comparison



Runtime comparison

- Now memory-bound CG
- 100 CG iterations
- Memory speed more important than peak FLOPs
- Not sufficient to fully explain jump from P100 to V100

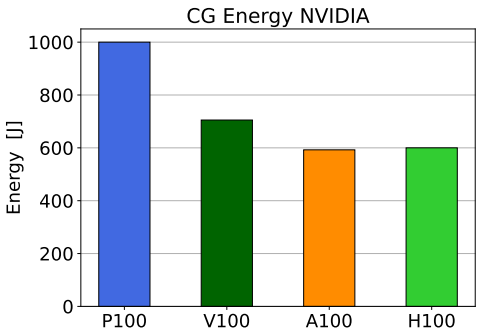
CG Results - Power Draw Comparison with Cholesky



Cholesky – CG power draw comparison

- CG draws less power than Cholesky for 100 iterations
- Behavior for A100 unclear

CG Results - Total Energy Consumption

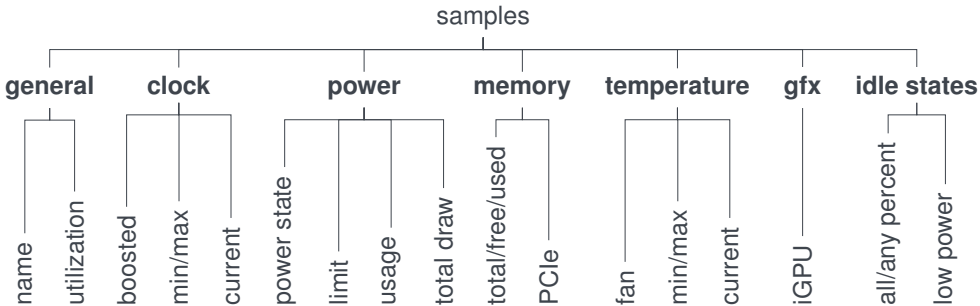


CG energy consumption

- in contrast to Cholesky, not correlated to DL
- A100 and H100 very similar
  - H100 is only slightly faster but draws more power, cancels out

hws — Beyond Energy and Power Sampling

hws Sampling

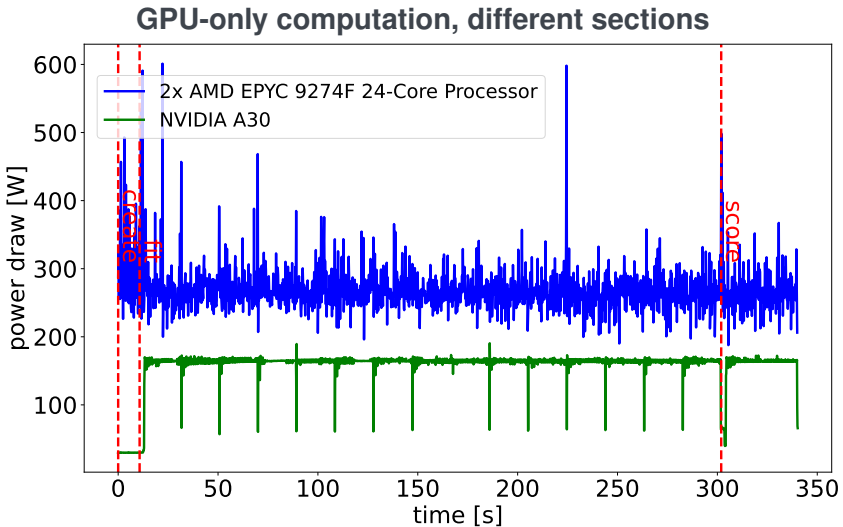


Two categories

fixed: gathered only once at the beginning (e.g., minimum/maximum clock frequencies)

sampled: gathered during the whole sampling period (e.g., current clock frequencies)

Fast Insight: E.g. Power Draw of CPU and GPU with PLSSVM



Summary

Summary

Towards Energy-Aware HPC

- considering energy more tricky than mere time-to-solution
- dependent on many hidden factors
- requires tool support for sampling energy and power on modern hardware

hws — A Hardware Sampling Library

- easy to use[1], high-level API
- low overhead, especially for GPU sampling
- vendor-independent, vendor-comprehensive
- supports all major hardware platforms, custom events
- hws is open-source, contributions welcome!

[1] if sampling permitted

University of Stuttgart
Germany

Thank You!

Dirk Pflüger
Scientific Computing / IPVS

Mail dirk.pflueger@ipvs.uni-stuttgart.de
Phone +49 711 685 88447
Internet https://www.ipvs.uni-stuttgart.de