HPCSE 2026 | dare | OPTKIT profiling & energy optimization toolkit | EuroHPC Joint Undertaking

OPTKIT is a C++11 library for energy measurement, performance bottleneck identification, and dynamic hardware adjustment. Adhering to a metrics-first approach and the RAII idiom, it ensures safe resource management and low overhead. It serves both as a diagnostic tool during development and as a lightweight runtime engine in production for on-the-fly hardware optimization. The framework provides cross-platform monitoring for CPUs, GPUs, and I/O subsystems across Intel, AMD, ARM, NVIDIA, and RISC-V (upto some level) architectures. Supporting over 60 metrics, OPTKIT enables real-time frequency scaling and energy tracking. The suite also includes the *optkit-cli* and *optkit-setenv* utilities, as well as language extensions for *C and Python*

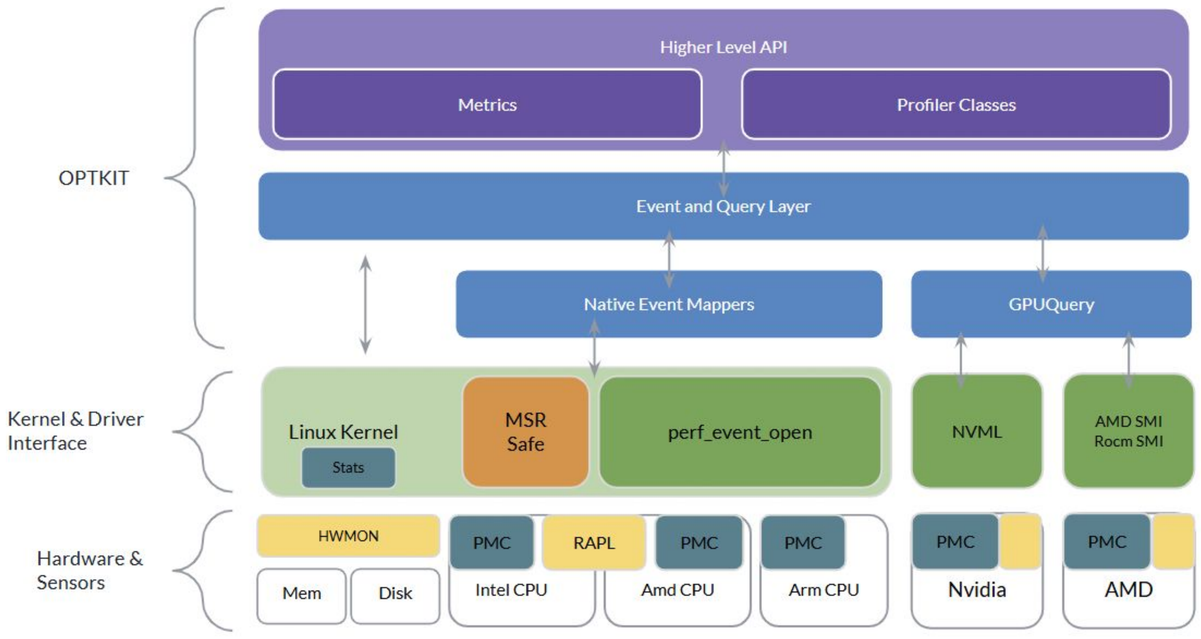
Backend and Telemetry Integration

Internally, OPTKIT uses perf event open for process-scoped PMCs and RAPL energy counters, with automatic child-thread aggregation. For architectures lacking RAPL, it integrates support for HWMON, GPU APIs (NVML, ROCm SMI), and Power Distribution Units (PDUs) for node-level power/energy measurements and profiling. To extract low-level telemetry inaccessible via perf event open syscall, OPTKIT supports the msr-safe kernel module for secure, system-wide machine-specific register (MSR) access without root privileges. GPU performance is monitored via Nvidia-GPM [18], with broader accelerator support reserved for future work. Finally, OPTKIT utilizes the ONNX Runtime C++ API [2] for portable high-speed model inference.

```cpp
static const MetricBuilder<uint64_t>& ipc() {
    static const MetricBuilder<uint64_t> metric = [] {
        auto inst_str = to_string(CoreEvents::INST_RETIRED);
        auto cyc_str  = to_string(CoreEvents::UNHALTED_CORE_CYCLES);

        return MetricBuilder<uint64_t>{}
            .add(inst_str, amd::EventMapper::get(CoreEvents::INST_RETIRED))
            .add(cyc_str, amd::EventMapper::get(CoreEvents::
    UNHALTED_CORE_CYCLES))
            .build("ipc", [inst_str, cyc_str](const auto& counts) -> double {
                uint64_t inst = get_event_count(counts, inst_str);
                uint64_t cyc  = get_event_count(counts, cyc_str);

                if (cyc == 0)return std::numeric_limits<double>::quiet_NaN();
                return static_cast<double>(inst) / static_cast<double>(cyc);
            });
    }();
    return metric;
}
```



Event Abstraction and Metric Construction

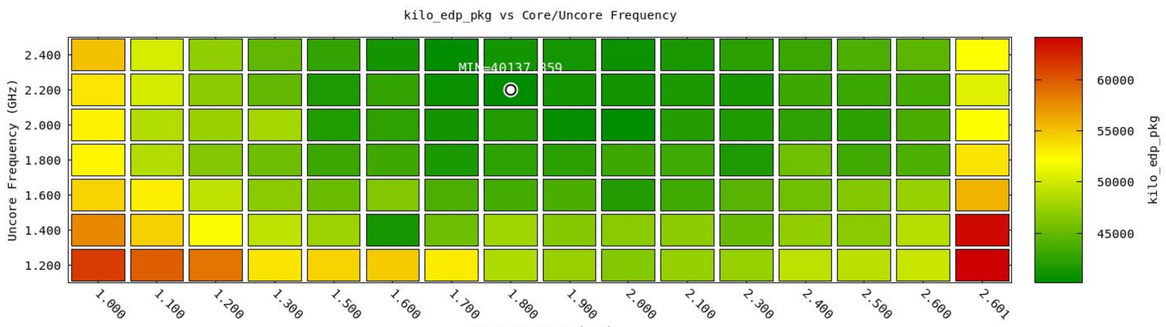
Event Layer abstracts hardware-specific PMCs into a unified naming convention re-solved by Native Event Mappers. OPTKIT provides over 60 predefined metrics, including Topdown L1/L2, Cache MPKI, and K-EDP. Users can define custom metrics via the MetricBuilder class by registering events and providing a calculation lambda; the framework then handles all underlying event configuration and runtime math.

OPTKIT Compatibility Matrix

	Intel	AMD	ARM Neoverse	RISC-V	Nvidia GPU	AMD GPU
PMU	P6, WSM, NHM, SNB, IVB, HSW, BDW, SKL, KBL, CFL, CML, ICL, TGL, RKL, ADL, RPL, MTL, SPR, EMR, GRN	Zen+	N1, V1, N2, V2	SG2042/4	GPM	-
Energy	RAPL	RAPL	HWMON	PDU	NVML	AMD/ROCm SMI
Temperature	HWMON	HWMON	HWMON	HWMON	NVML	AMD/ROCm SMI
Frequency Scaling	SYSFS - MSR	SYSFS	SYSFS	SYSFS	NVML	AMD/ROCm SMI

Use Case

Developed an AI model that analyzes live hardware telemetry to dynamically adjust core and uncore frequencies, optimizing the Energy-Delay Product (EDP) without requiring manual profiling. To train our AI model, we utilized OPTKIT to sample hardware events during the execution of the NAS benchmarks, performed frequency sweep to constitute outputs and tested the model on CoMD and LULESH benchmarks.



Hardware Specifications of Cross-Architecture Test Environments

Architecture	Processor	Clocks (Base/Turbo)	TDP	Cores	Memory	Vector
RISC-V	2× Sophgo SG2042	2.0 GHz / n/a	120 W	2×64	250 GB DDR4	RVV v0.7.1
NVIDIA Grace	2× Neoverse-V2	3.3 GHz / n/a	500 W	2×72	470 GB LPDDR5X	SVE2, NEON
Intel Cascade Lake	2× Xeon 6240	2.6 GHz / 3.9 GHz	150 W	2×18	187 GB DDR4	AVX-512
AMD Zen4	1× Ryzen 7950X	4.5 GHz / 5.7 GHz	170 W	1×32	64 GB DDR5	AVX-512

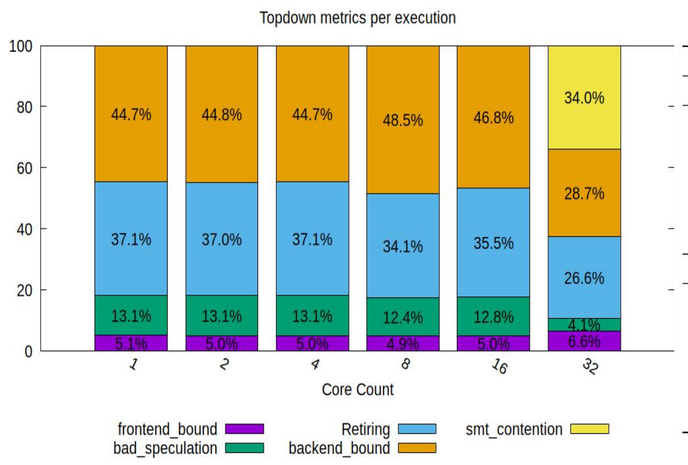
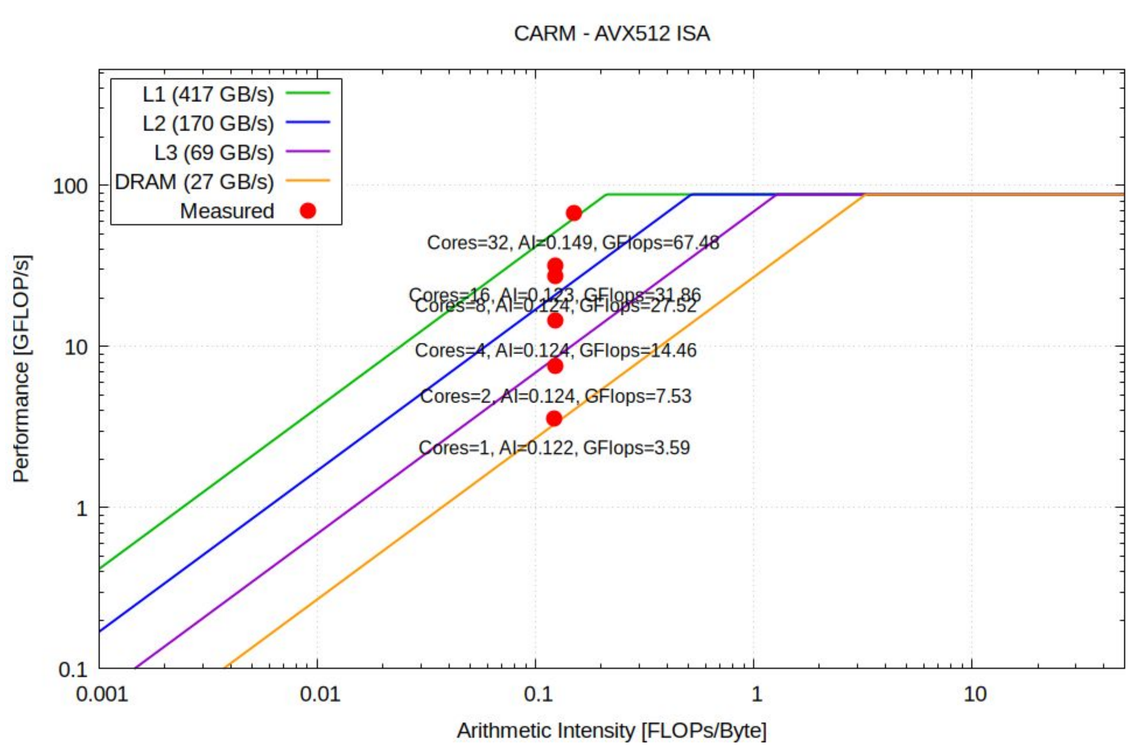


Table: Performance and K-EDP Comparison for CoMD and LULESH Benchmarks

	AMD	Intel	ARM	RISC-V
CoMD Benchmark [8]				
Baseline [1]				
Exec Time(s)	83	105.18	51.06	1347.09
K-EDP	955.59	3128.72	612.80	329142.32
OPTKIT [2]				
Exec Time(s)	97	122.20	72.13	–
K-EDP	564.47	3532.22	500.74	–
Savings	**40.92%**	**-11.42%**	**18.28%**	–
LULESH Benchmark [11]				
Baseline [1]				
Exec Time(s)	348.10	248.36	78.13	5228.39
K-EDP	12386.97	14358.47	1727.27	5025260.13
OPTKIT [2]				
Exec Time(s)	376.15	306.39	106.23	–
K-EDP	5312.68	15748.82	1243.08	–
Savings	**57.11%**	**-9.68%**	**28.03%**	–



This work was supported by the DARE SGA1 project under grant agreement No 101202459, European High-Performance Computing Joint Undertaking and its members (including top-up funding by the Ministry of Education, Youth and Sports of the Czech Republic ID: MC2502) and the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90254).