inesc id lisboa

The CARM Tool

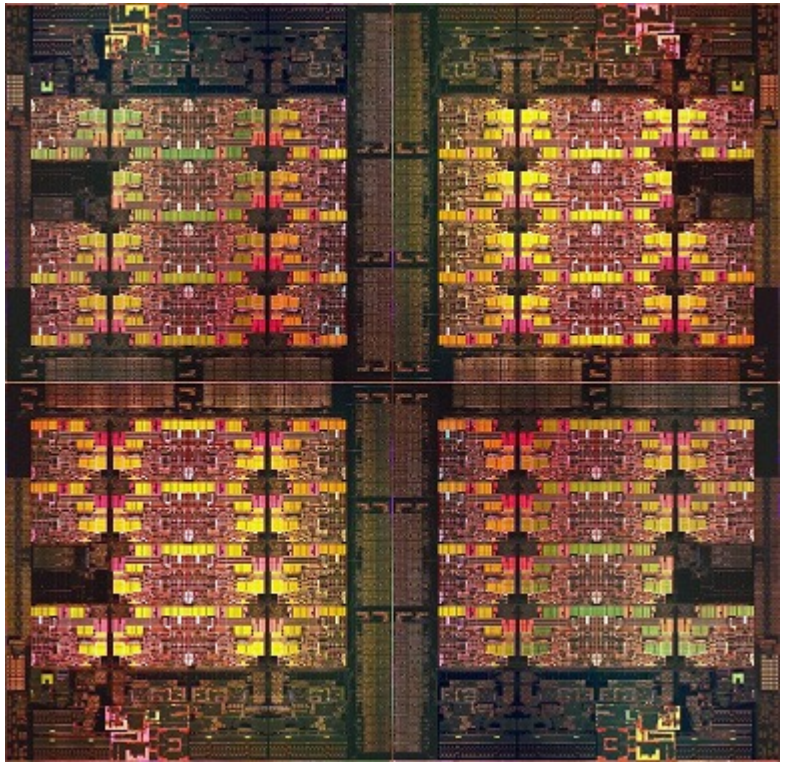
Cache-aware Roofline Model for HPC=CPU+GPU

Aleksandar Ilic
Leonel Sousa
José Morgado
Alexandre Rodrigues
Diogo Matos

48TH VI-HPS TUNING WORKSHOP
13 February 2026

Motivation

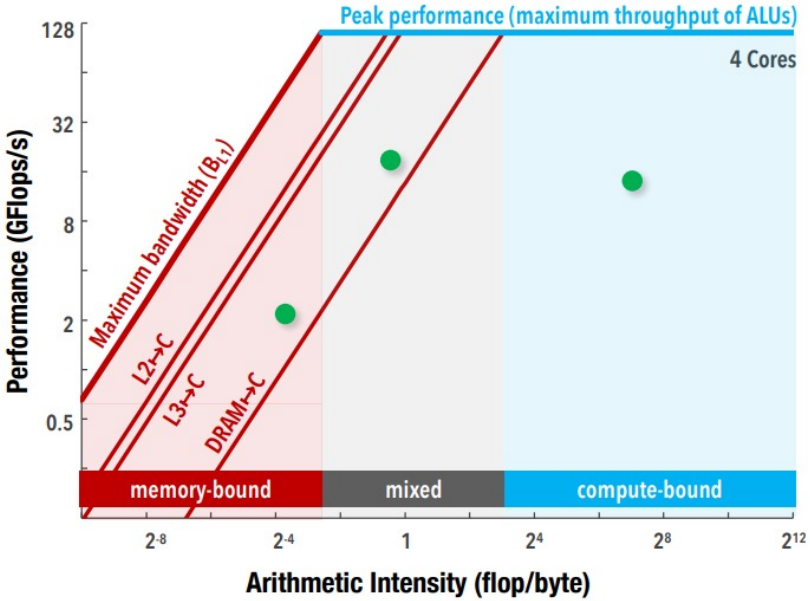
- Modern HPC systems and applications are **complex** and **heterogeneous**
  - Hard to model
  - Hard to optimize
- The CARM as a solution



Intel Sapphire Rapids CPU

Motivation

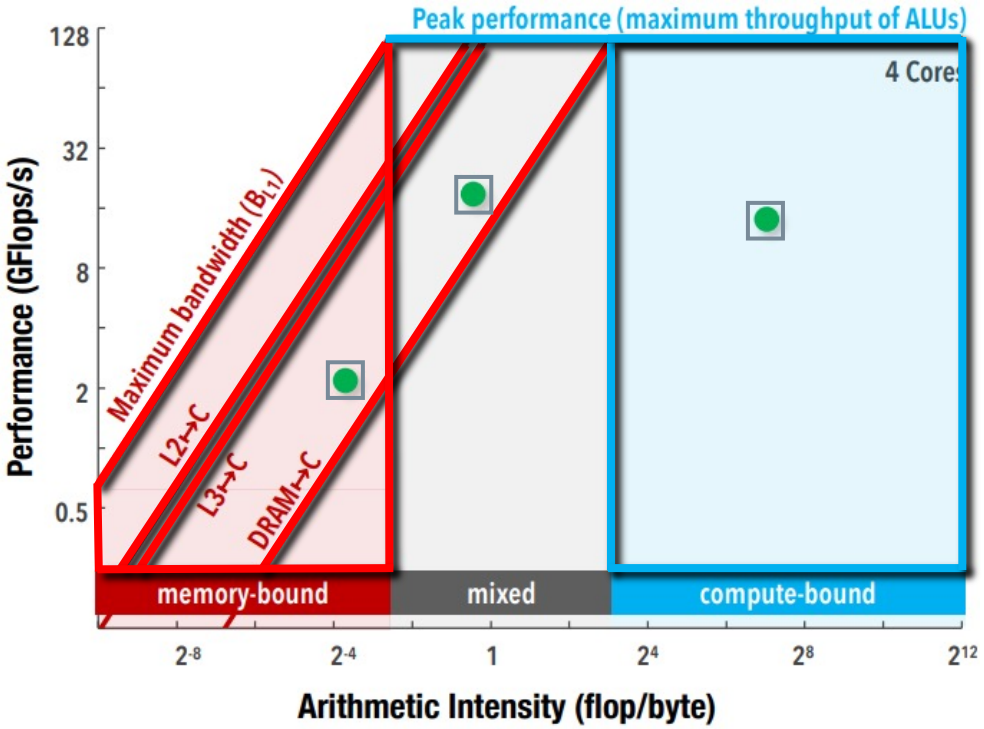
- Modern HPC systems and applications are **complex** and **heterogeneous**
  - Hard to model
  - Hard to optimize
- The CARM as a solution
  - Easy to understand
  - Accurate performance overview
  - Good optimization hints
- CARM was originally supported only by Intel Advisor (commercial tool)



The Cache-Aware Roofline Model

The Cache-aware Roofline Model

- How does CARM work?
  - Sloped Roof
  - Flat Roof
- How to generate CARM?
  - Floating-Point Microbenchmarks
  - Memory Microbenchmarks
- The CARM Tool
  - Automatic Benchmarking
  - Automatic CARM Generation



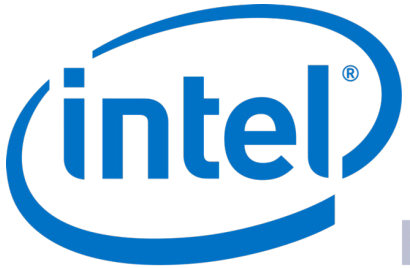
A. Ilic, F. Pratas and L. Sousa, "Cache-aware Roofline model: Upgrading the loft", IEEE Computer Letters, 13 (1), 2014

State of the art – Roofline Tools

Tools / Features	Intel Advisor	AMD uProf	ERT	CARM Tool
Supported Architectures	Intel	AMD	Intel | A	Intel | AMD ARM | RISC-V
Supported Roofline	CARM	ORM	? CARI	CARM
Application Analysis	DBI	PMUs	No sup	DBI | PMUs
Open-Source	No	No	Yes	Yes

- And about ARM and RISC-V?

  and GPUs?



The CARM Tool – Supported ISAs

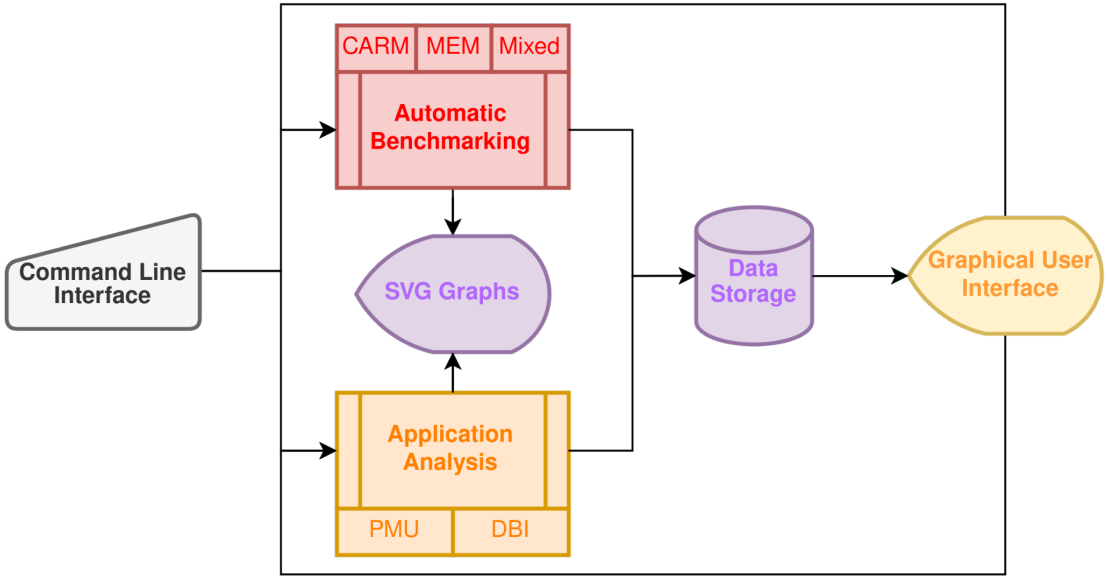
intel	AMD	arm	RISC-V
Scalar	Scalar	Scalar	Scalar
SSE	SSE	Neon	RVV 0.7.1 | 1.0
AVX2	AVX2	SVE	
AVX512	AVX512		

CONTENTS

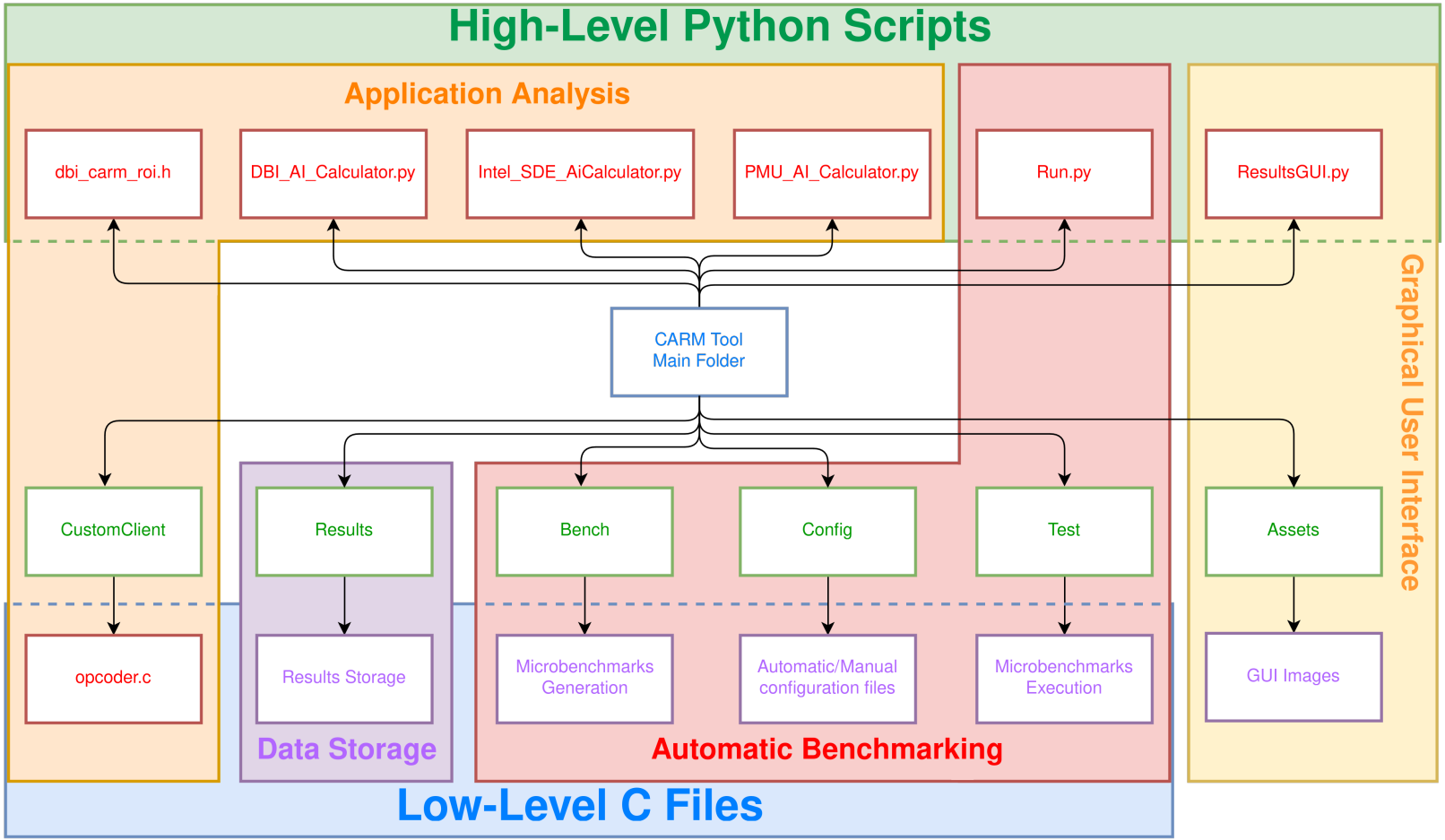
- **The CARM Tool**
- Extension to GPU
- Integration: HPC Tools, Centres and Codesign

The CARM Tool – High Level

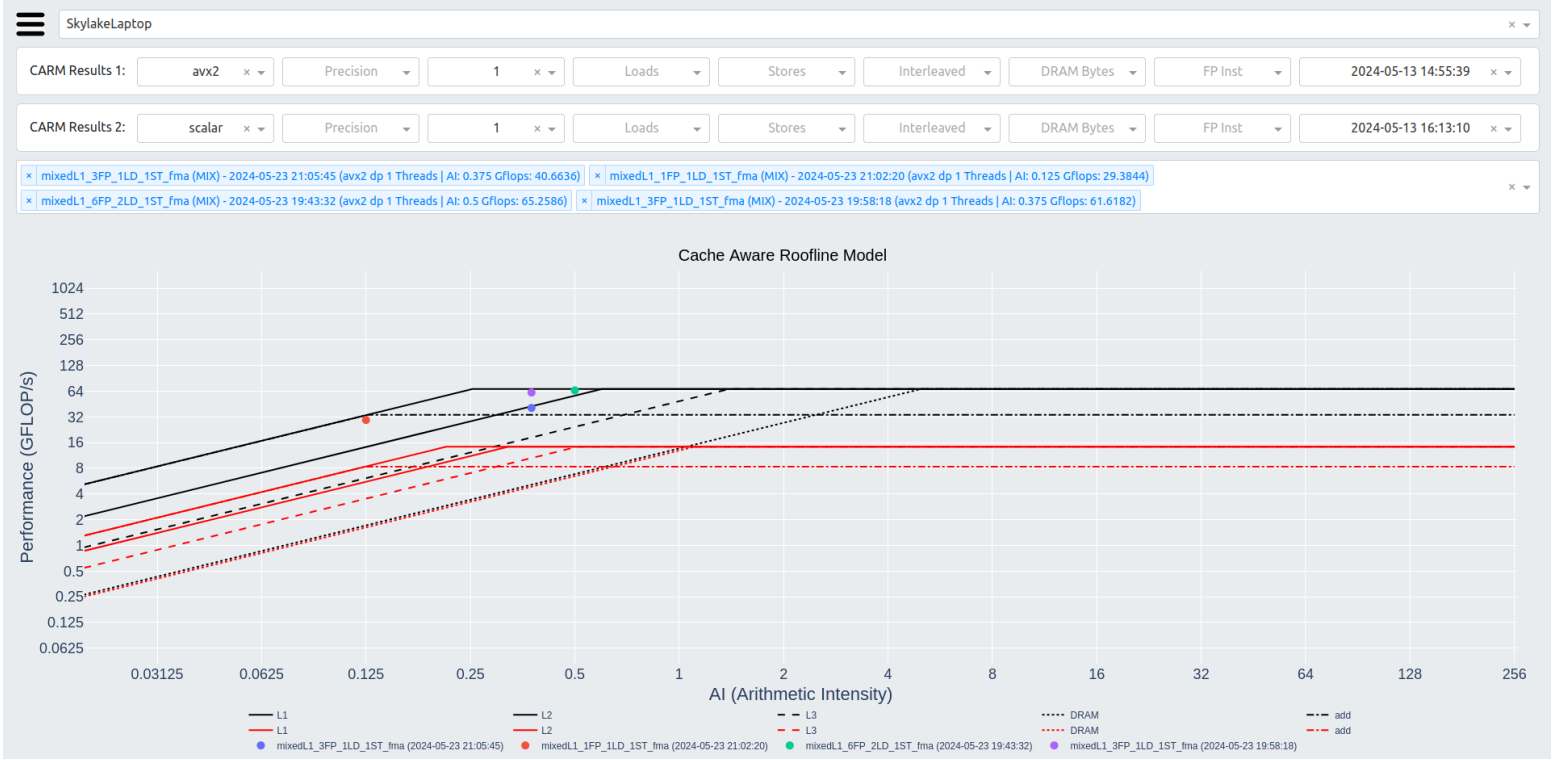
- Interfacing
  - Command Line Interface
  - Graphical User Interface
- Automatic Benchmarking
  - CARM Benchmarks
  - Memory / FP / Mixed Benchmarks
- Application Analysis
  - PMU Analysis
  - DBI Analysis



The CARM Tool – High Level



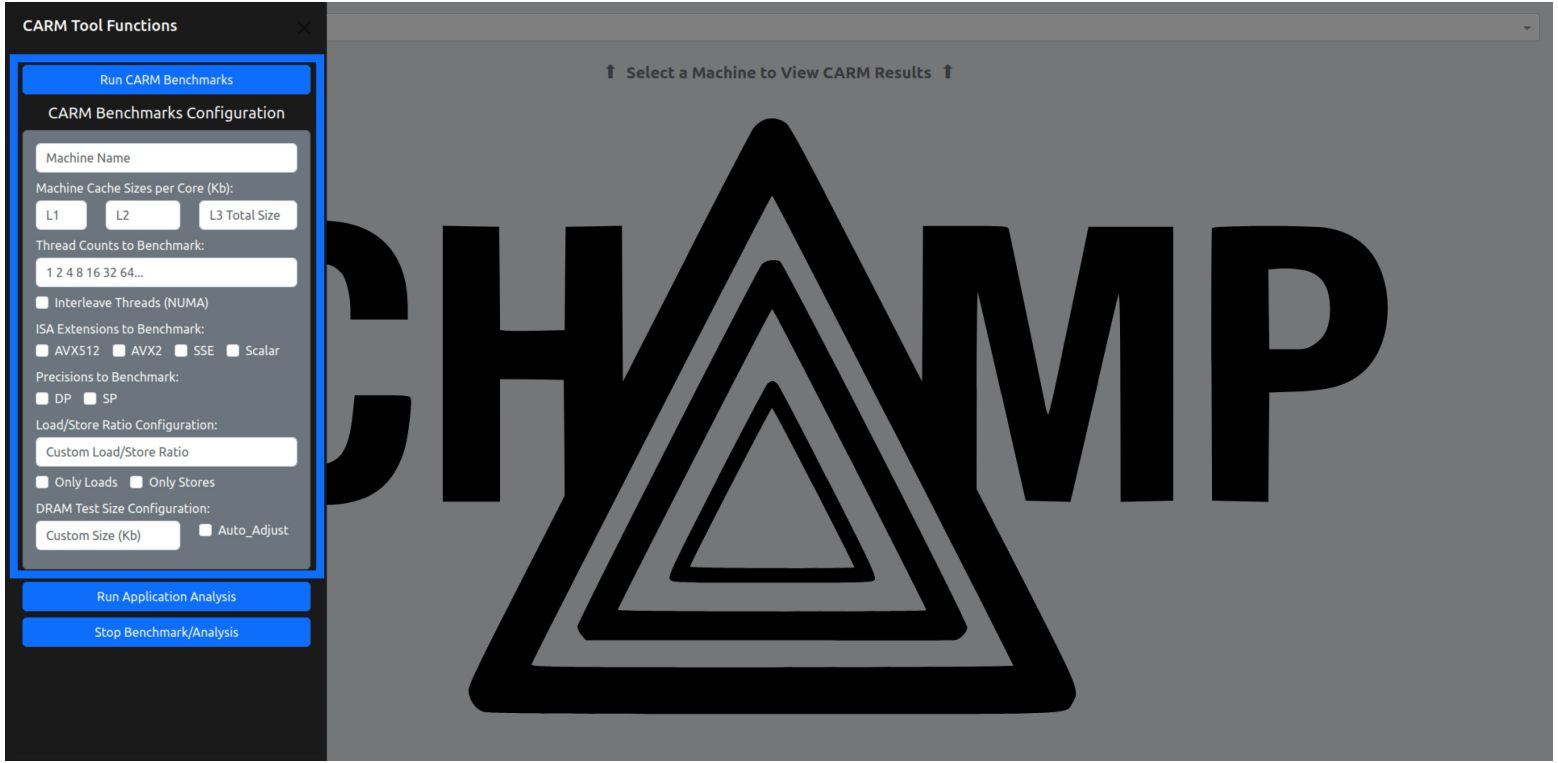
Results Visualization



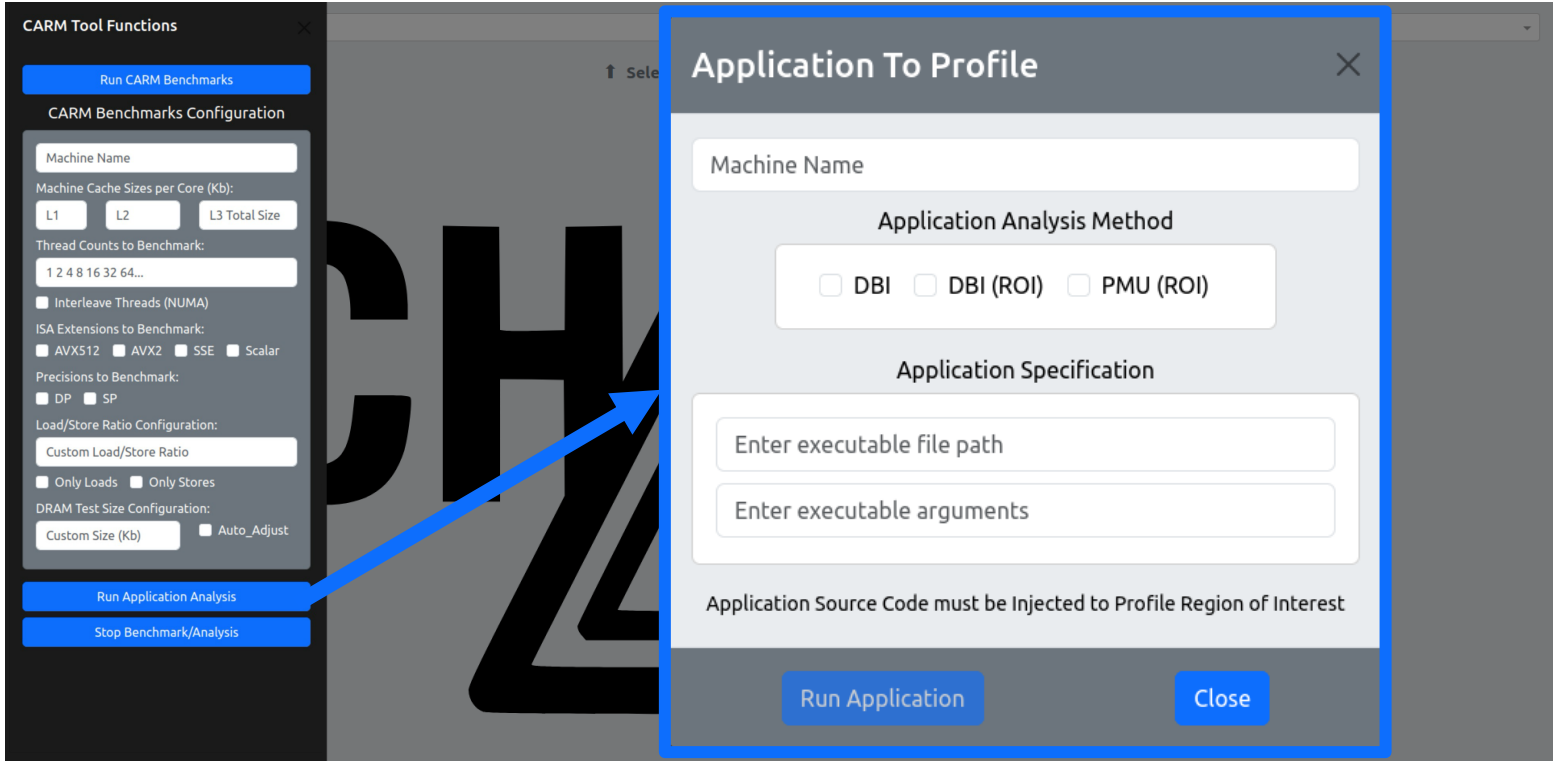
Graphical User Interface

- ResultsGUI.py
- Assets
- GUI Images

Benchmark Execution



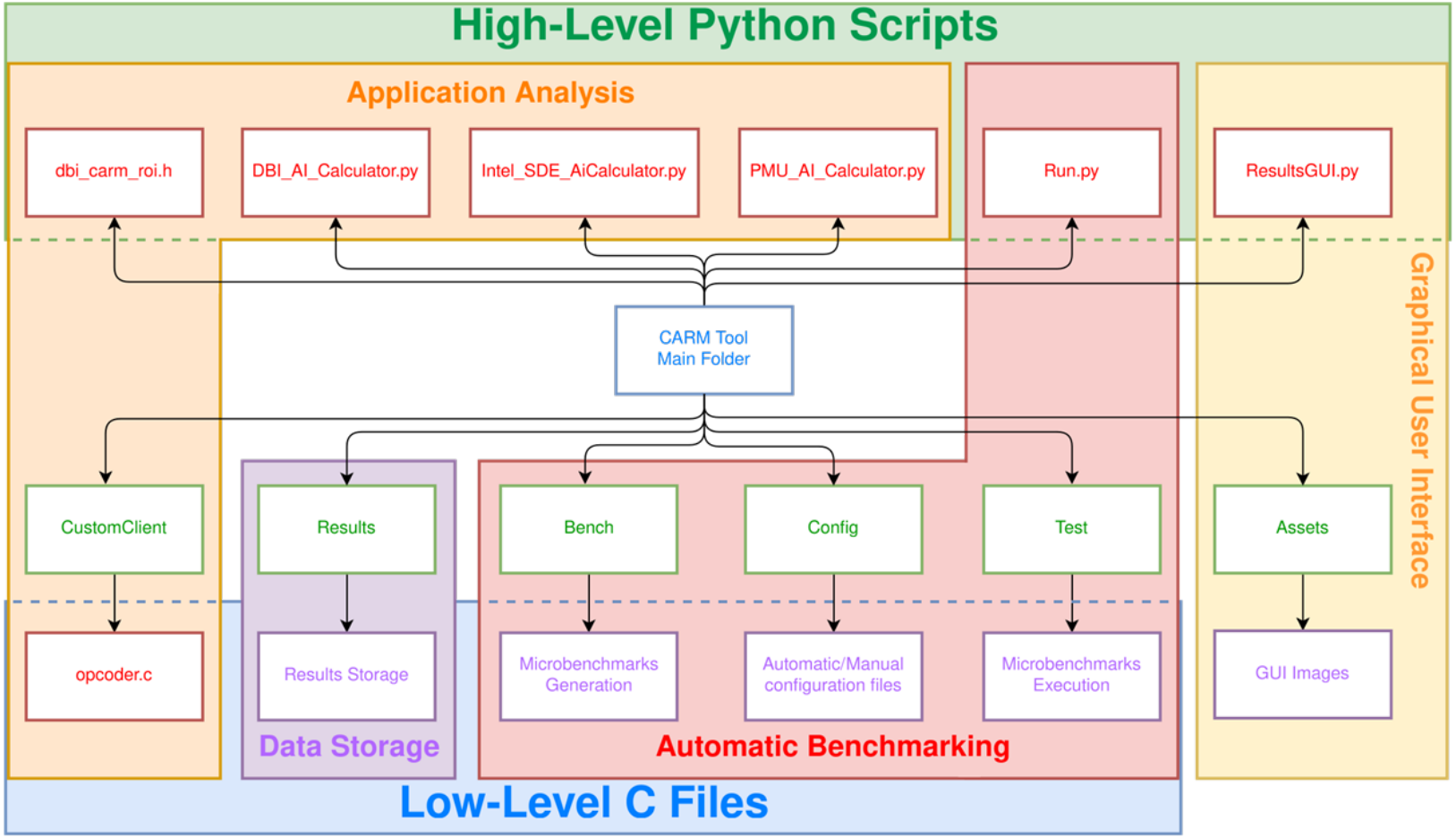
Application Profiling



Graphical User Interface

- ResultsGUI.py
- Assets
- GUI Images

The CARM Tool – Automatic Benchmarking



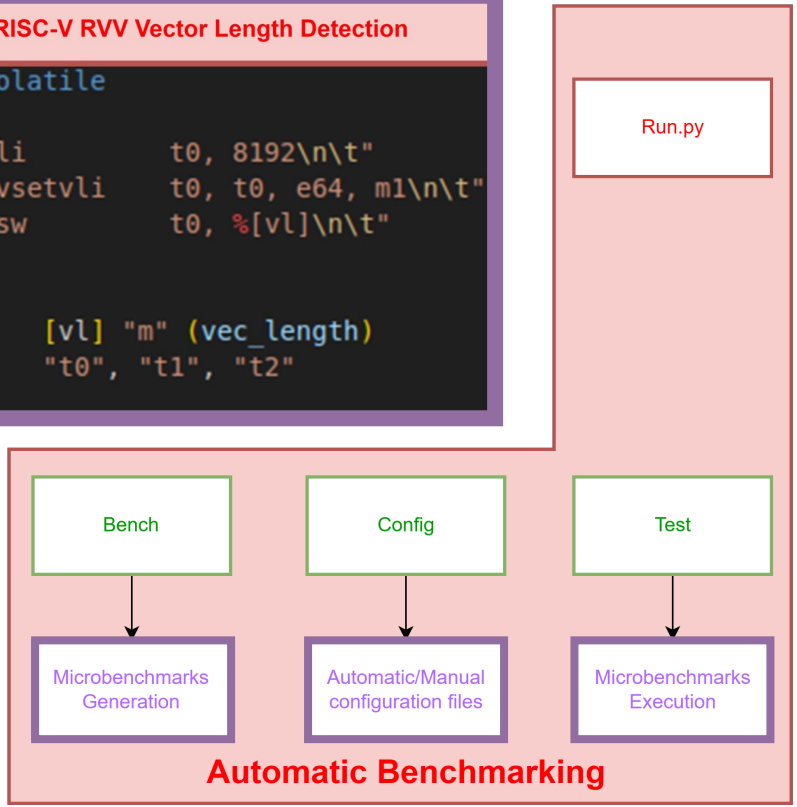
The CARM Tool – Automatic Benchmarking

AVX512 Microbenchmarks Pseudo-Code

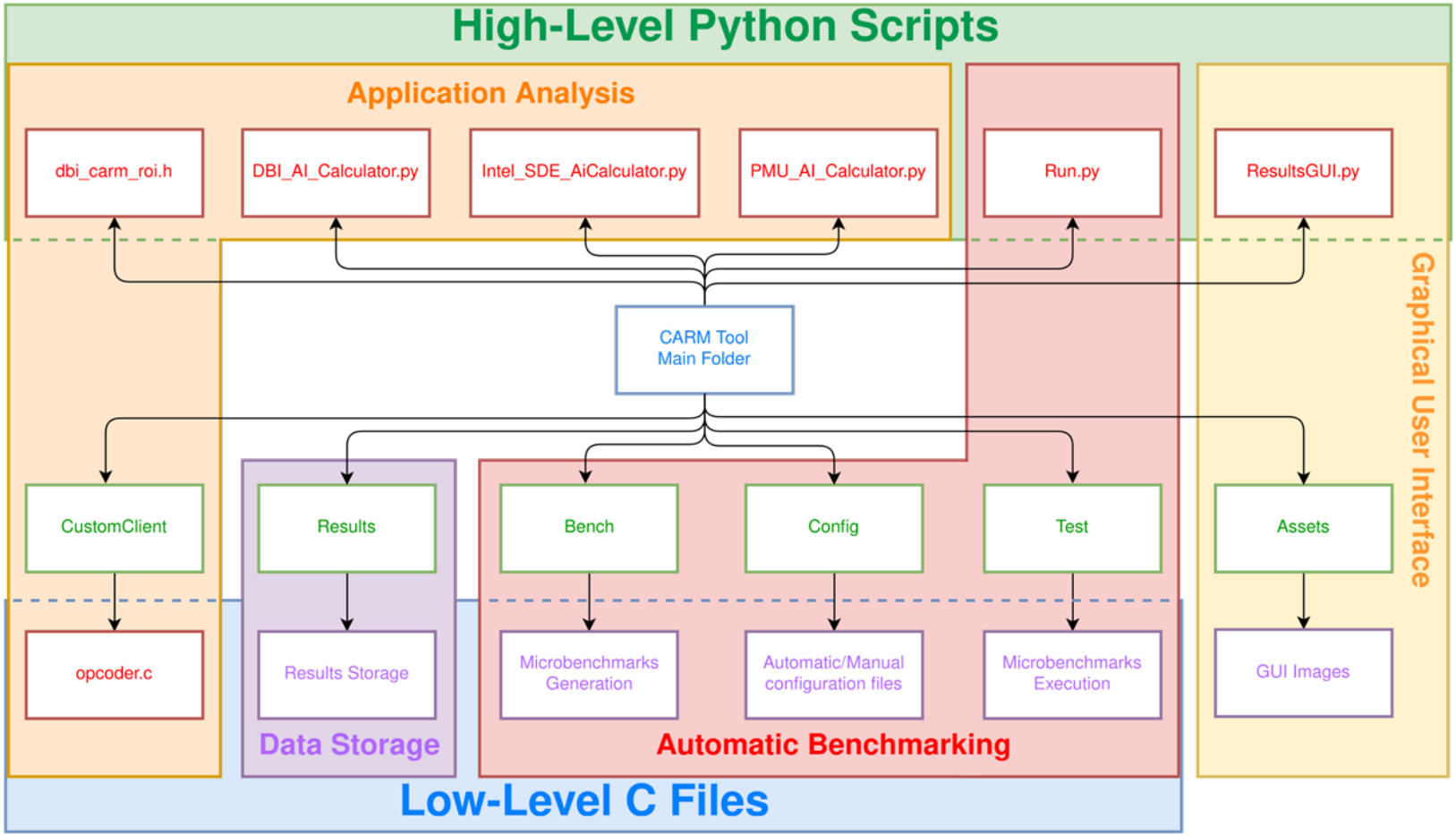
```
__asm__ __volatile__ (
    "movq %0, %%r8" //Outer Loop Variable Iterations
    "Loop2_%=:"
    "movq %1, %%rax" //Pointer to Test Data Array
    "movq $388, %%rdi" //Inner Loop Iterations
    "Loop1_%=:"
    "vmovapd 0(%%rax), %%zmm0" //Vector Load
    "vmovapd %%zmm1, 64(%%rax)" //Vector Store
    "vfmadd132pd %%zmm2, %%zmm2, %%zmm2" //Vector FMA
    //................................//
    "vmovapd 16384(%%rax), %%zmm29"
    "vmovapd %%zmm30, 16448(%%rax)"
    "vfmadd132pd %%zmm31, %%zmm31, %%zmm31"
    "addq $16512, %%rax" //Pointer Bump
    "subq $1, %%rdi"
    "jnz Loop1_%=" //Inner Loop End
    "vmovapd 0(%%rax), %%zmm0"
    "vmovapd %%zmm1, 64(%%rax)"
    "vfmadd132pd %%zmm2, %%zmm2, %%zmm2"
    //................................//
    "subq $1, %%r8"
    "jnz Loop2_%=" //Outer Loop End
    :"r"(num_reps_t),"r" (test_var)
    :"rax","rdi","r8","zmm0-31"
);
```

RISC-V RVV Vector Length Detection

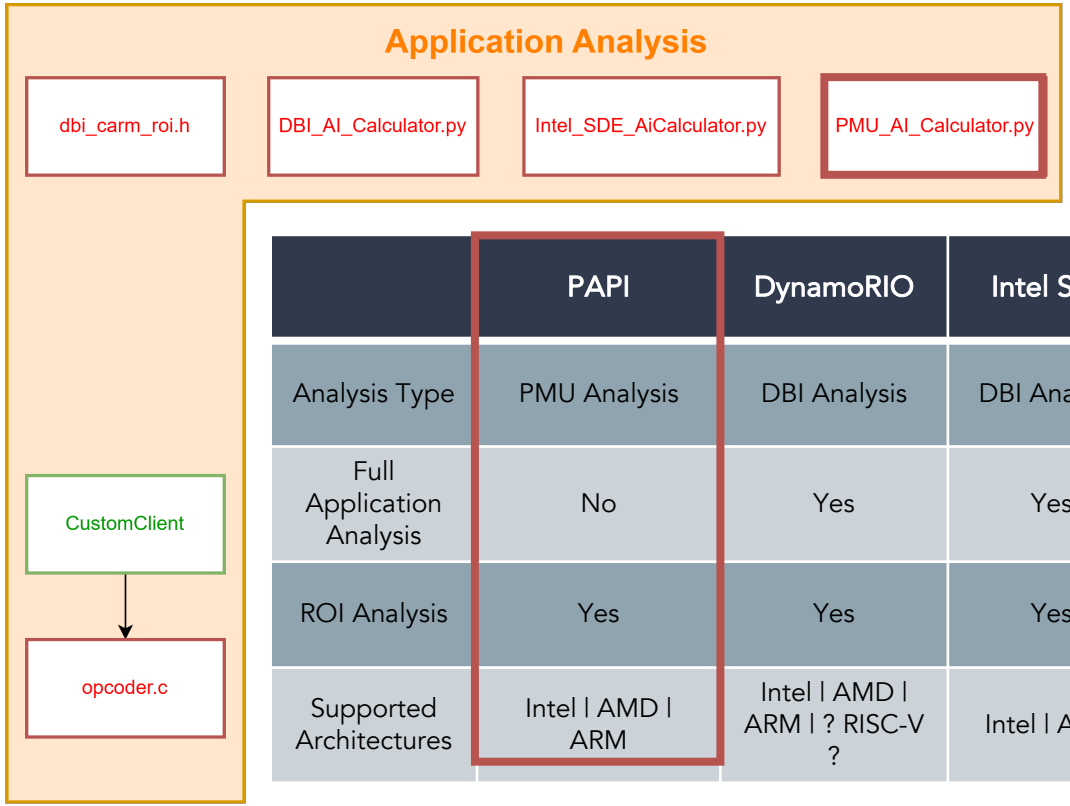
```
asm volatile
(
    "li          t0, 8192\n\t"
    "vsetvli     t0, t0, e64, m1\n\t"
    "sw          t0, %[vl]\n\t"

    :
    :   [vl] "m" (vec_length)
    :   "t0", "t1", "t2"
);
```



The CARM Tool – Application Analysis



The CARM Tool – Application Analysis



	PAPI	DynamoRIO	Intel SDE
Analysis Type	PMU Analysis	DBI Analysis	DBI Analysis
Full Application Analysis	No	Yes	Yes
ROI Analysis	Yes	Yes	Yes
Supported Architectures	Intel | AMD | ARM	Intel | AMD | ARM | ? RISC-V ?	Intel | AMD

PAPI

DynamoRIO

intel

The CARM Tool – Application Analysis

Application Analysis

dbi_carm_roi.h

DBI_AI_Calculator.py

Intel_SDE_AiCalculator.py

PMU_AI_Calculator.py

CustomClient

opcoder.c

Link PAPI library during compilation

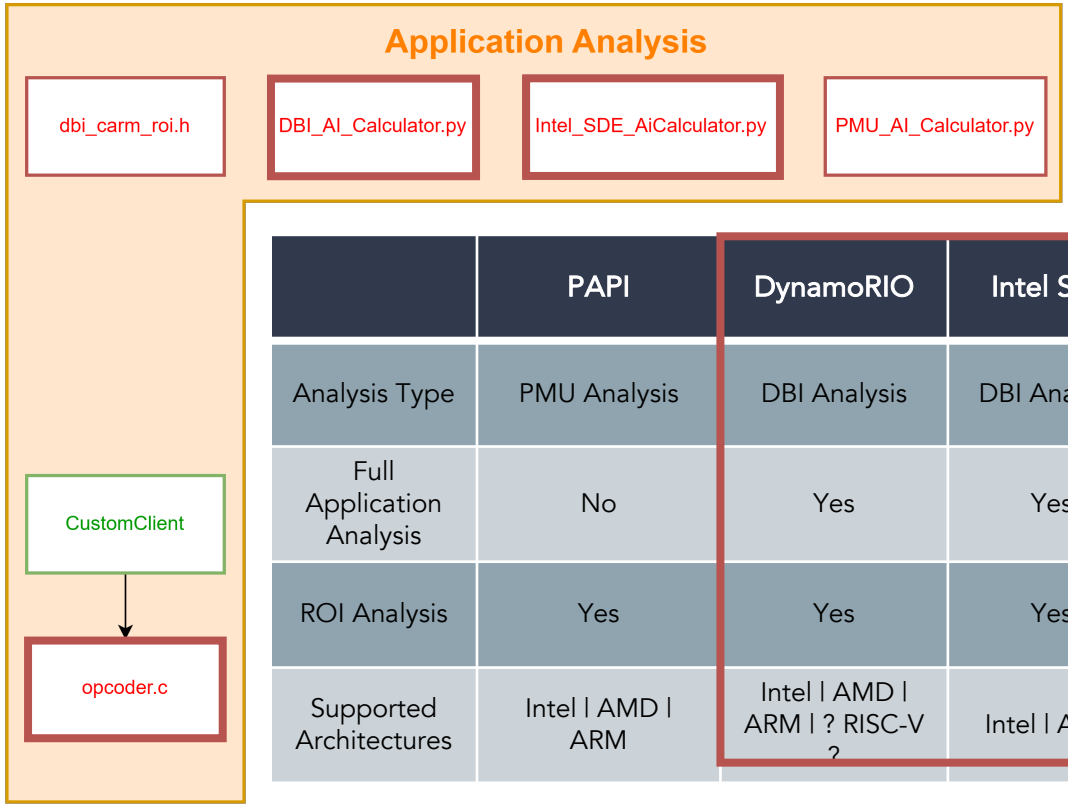
```c
#include <papi.h> //PAPI API header file

int main(int argc, char* argv[]) {

    PAPI_hl_region_begin(""); //PAPI ROI begin API function
    for (i=0; i<num_runs; i++){
        matrix_multiply(a, b, c, msize);
    }
    PAPI_hl_region_end(""); //PAPI ROI end API function

    return 0;
}
```

PAPI

DynamoRIO

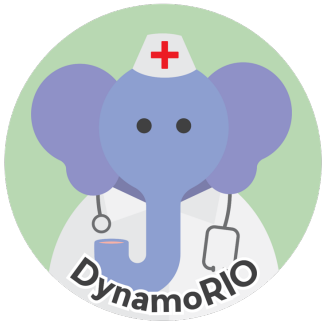
intel

The CARM Tool – Application Analysis



	PAPI	DynamoRIO	Intel SDE
Analysis Type	PMU Analysis	DBI Analysis	DBI Analysis
Full Application Analysis	No	Yes	Yes
ROI Analysis	Yes	Yes	Yes
Supported Architectures	Intel | AMD | ARM	Intel | AMD | ARM | ? RISC-V ?	Intel | AMD

PAPI



intel

The CARM Tool – Application Analysis

Application Analysis

dbi_carm_roi.h

DBI_AI_Calculator.py

Intel_SDE_AiCalculator.py

PMU_AI_Calculator.py

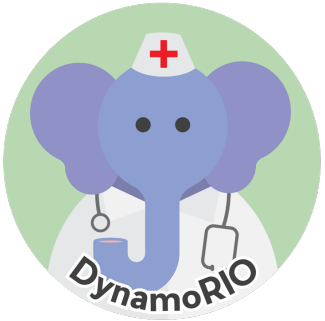
CustomClient

opcoder.c

Place dbi_carm_roi.h in the same folder as the source code

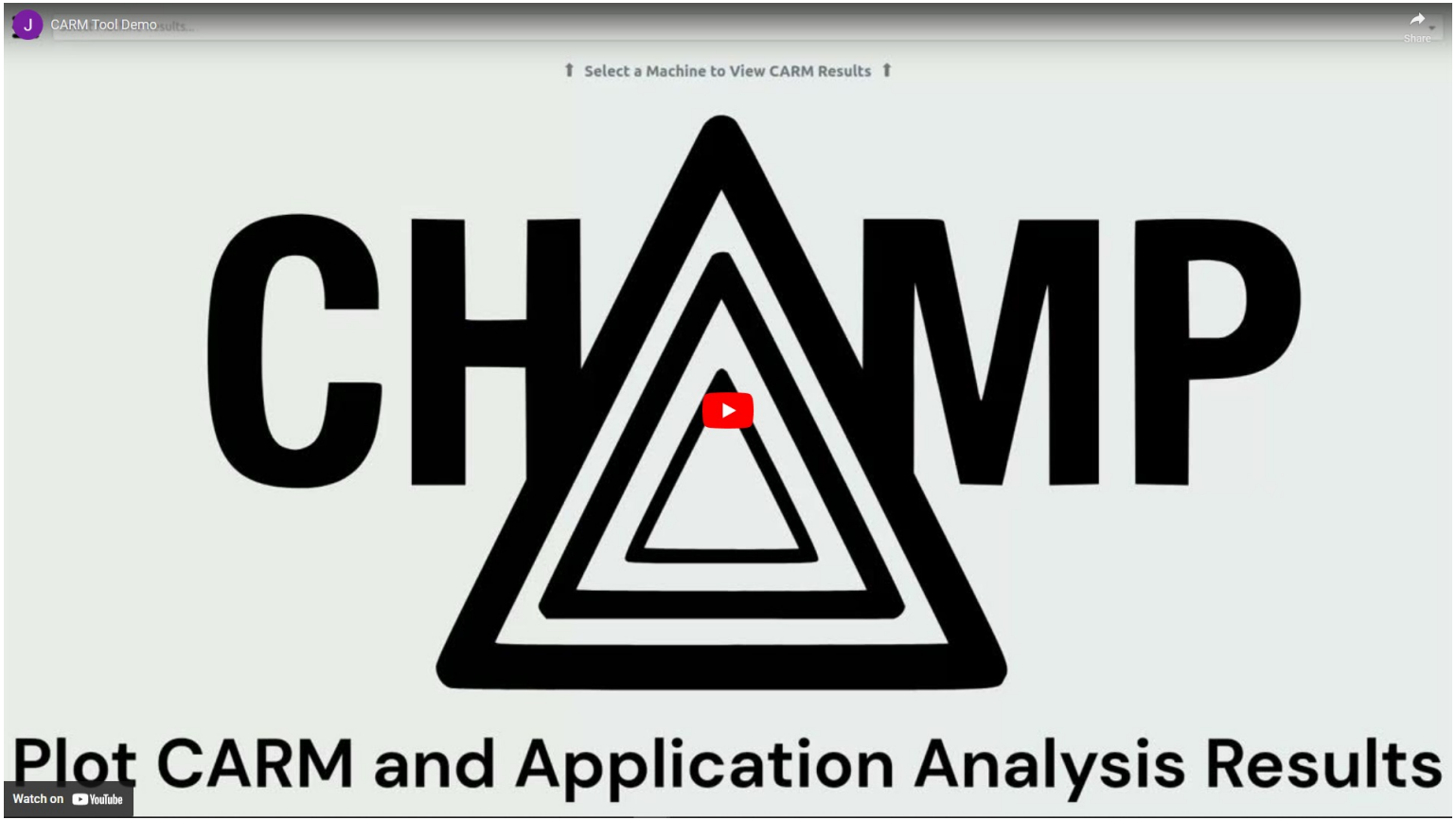
```c
#include "dbi_carm_roi.h" //DBI ROI API header file

int main(int argc, char* argv[]) {

    CARM_roi_begin(); //DBI ROI begin API function
    for (i=0; i<num_runs; i++){
        matrix_multiply(a, b, c, msize);
    }
    CARM_roi_end(); //DBI ROI end API function

    return 0;
}
```

PAPI



intel

The CARM Tool – Demo



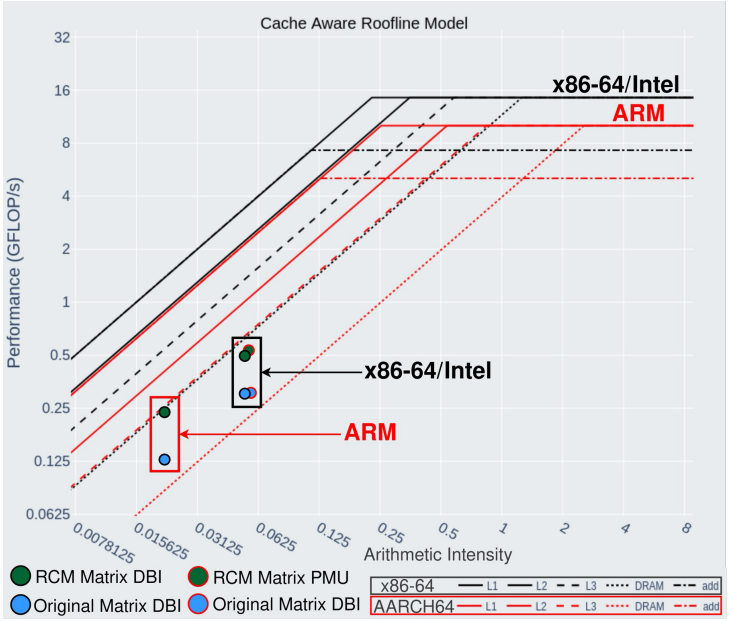
Results – Application Analysis

Cross-Architecture SpMV Analysis

- Using the Eigen library
- SpMV performance comparison

SpMV Performance

- RCM Re-Ordering improves performance



CONTENTS

- The CARM Tool
- **Extension to GPU**
- Integration: HPC Tools, Centres and Codesign

GPU integration in the CARM Tool

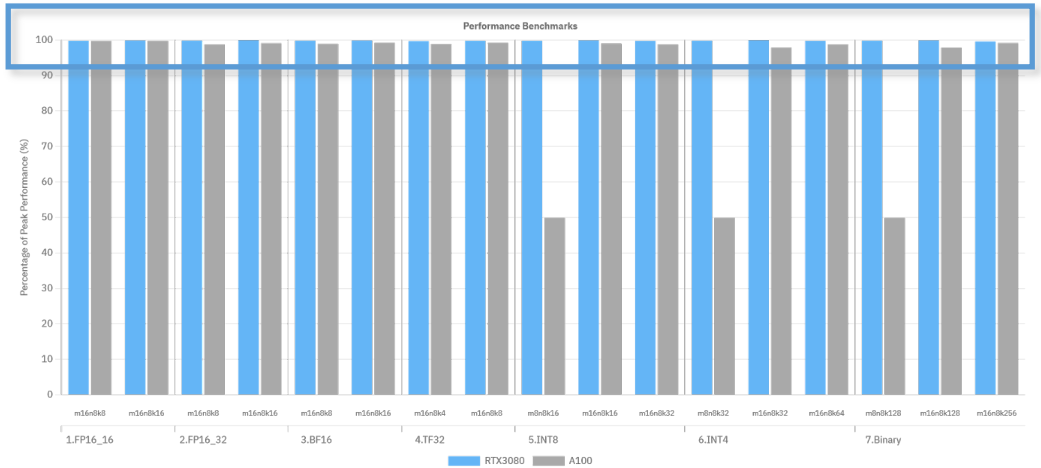
- Nvidia GPU extension + Tensor CARM
  - Functionality to test both CUDA Cores and Tensor Cores
  - Functionality to test Shared Memory, L2 cache and Global Memory
- Allows for the profiling of GPU applications using Nsight Compute
  - Functionality to profile entire applications or target individual kernels
  - Potential shown in the optimization of CNNs
- AMD GPU extension
  - Complete implementation for AMD MI GPUs using HIP
  - Supporting benchmarking using AMD RocProf

Performance benchmarks

- Maximum measured performance
  - Target instruction in PTX assembly
  - Thousands of iterations, thousands of threads
- Different data precisions
  - Set of different instruction shapes for mat-mul

```
// data preparation in CUDA C++
// ...
mov.u64 %time_start, %%globaltimer;
for i in 0 to Iter do:                                  D = A x B + C
    mma.sync.aligned.shape.row.col.precision C, A, B, C;
    // loop unrolling
end
mov.u64 %time_end, %%globaltimer;
```

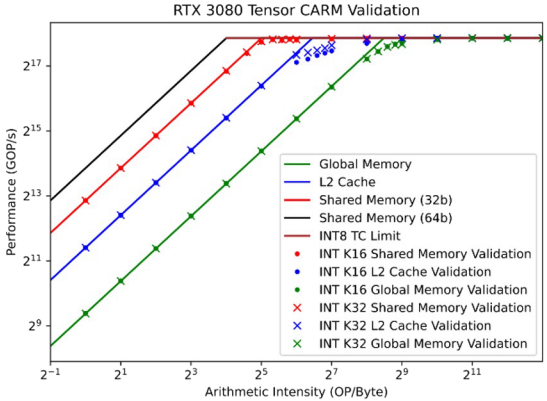
precision: { s32.s8.s8.s32 | f32.f16.f16.f32 | f32.tf32.tf32.f32 | ... } → int8 shape: { m8n8k16 | m16n8k16 | m16n8k32 }

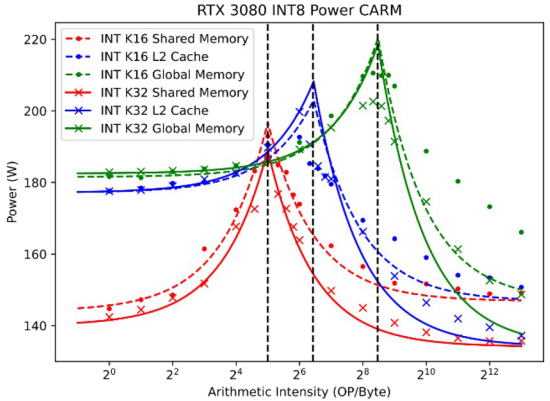


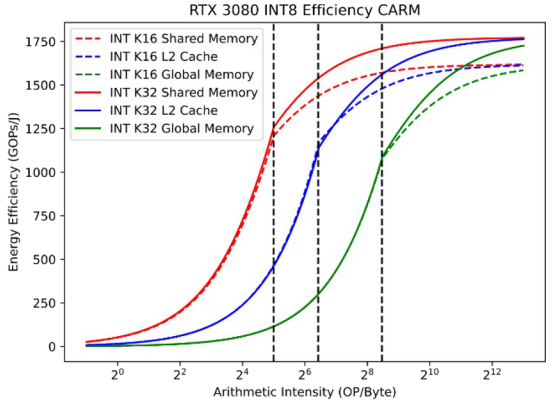
Within 0.5% on the RTX 3080

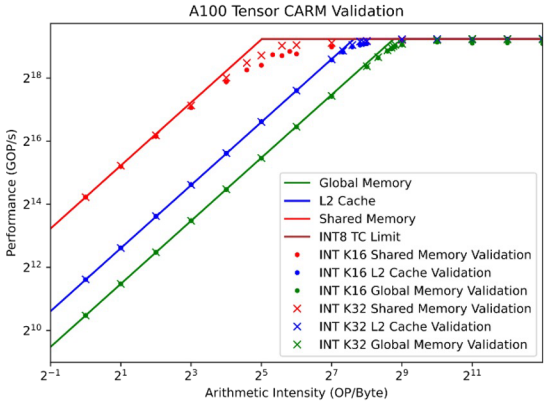
2.5% on the A100 (some exceptions)

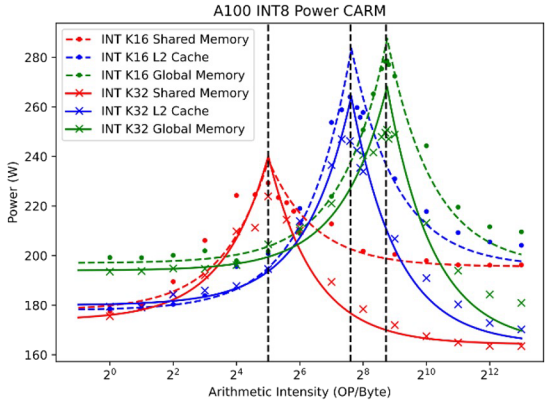
Tensor CARM Construction & Validation

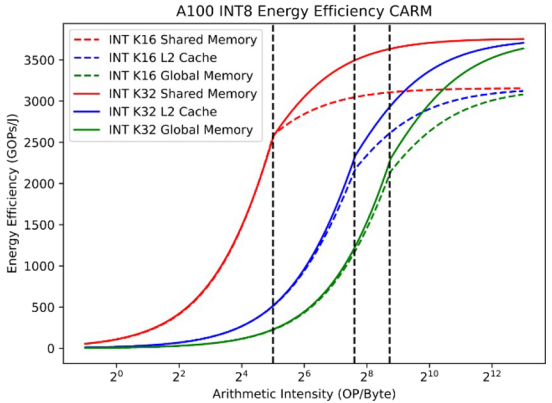










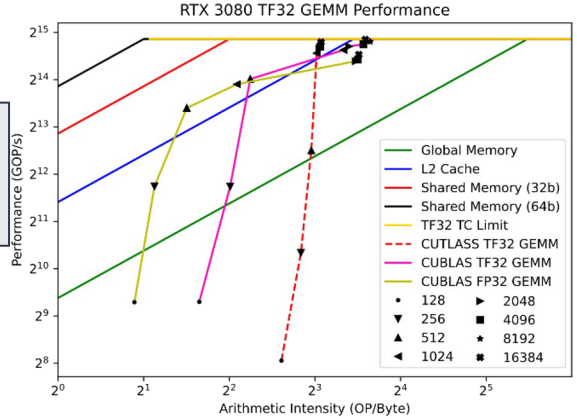


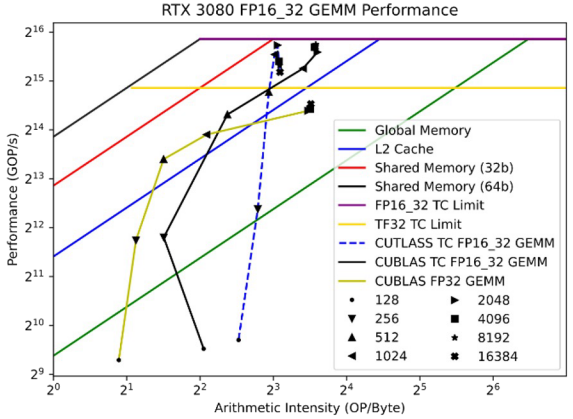
Characterization of DL GPU Kernels

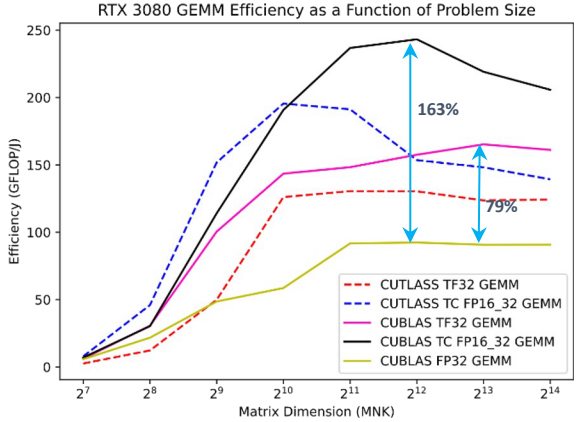
- Use CARM to evaluate GPU kernels
  - GEMM kernels provided by cuBLAS and CUTLASS
  - Deep learning kernels provided by cuDNN
- Nsight Compute as profiler tool
  - Bytes transferred, FLOPs executed and execution time
  - Cannot measure integer operations

GEMM and Convolutional Kernels

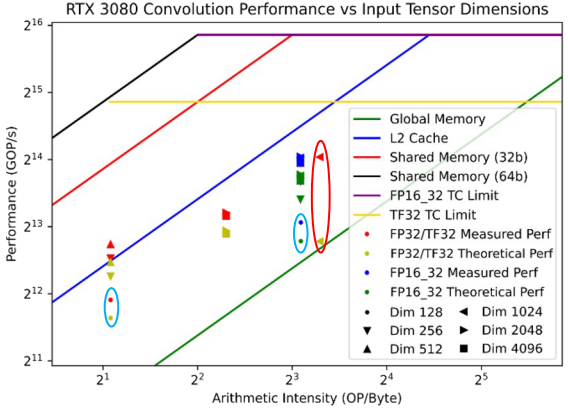
GEMM

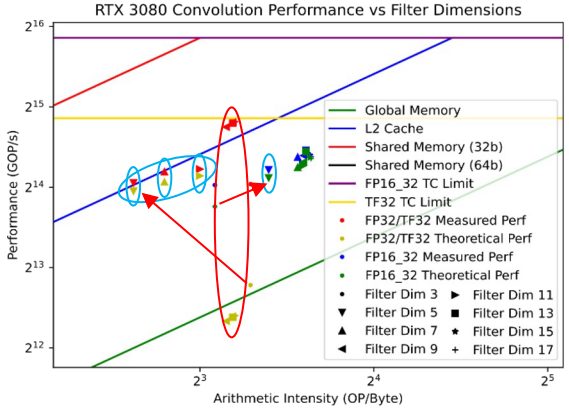


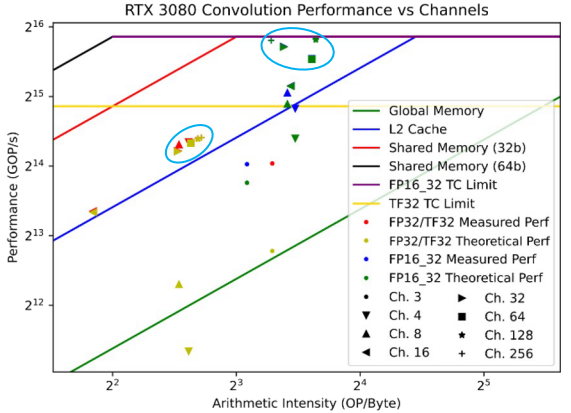




Convolve





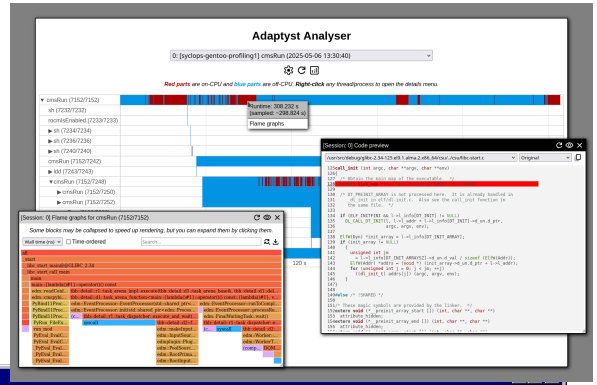


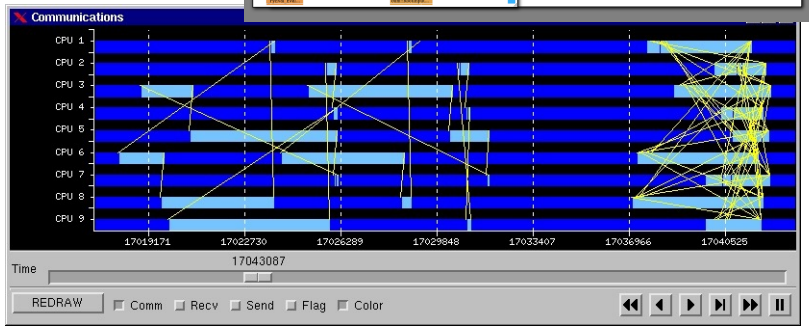
CONTENTS

- The CARM Tool
- Extension to GPU
- **Integration: HPC Tools, Centres and Codesign**

Integration into the HPC Centers

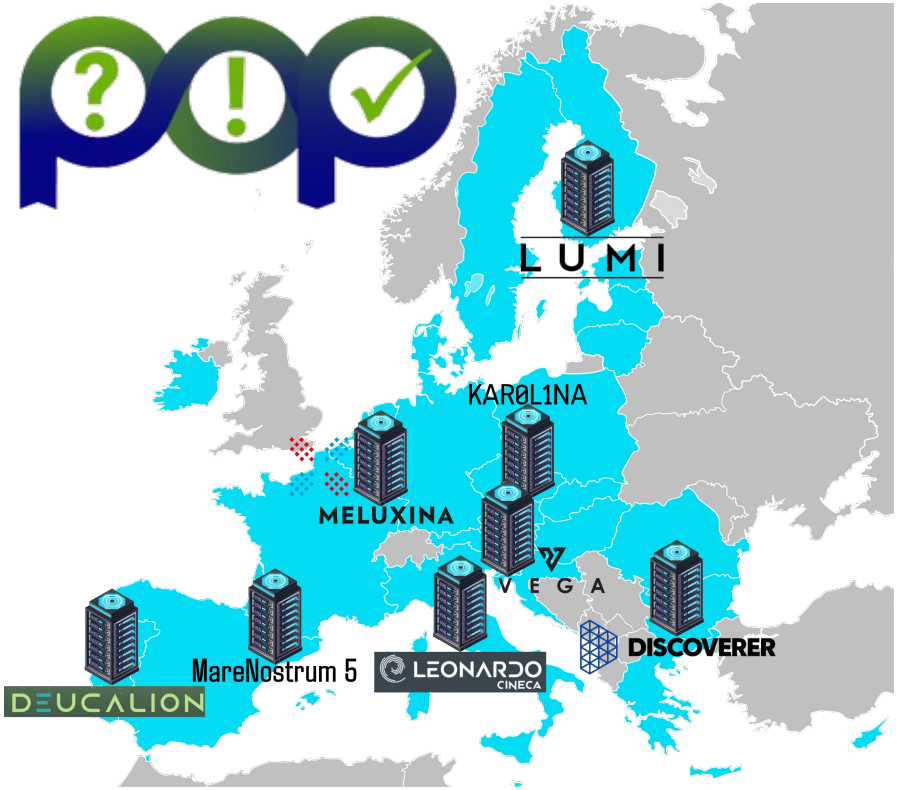
- Tools in HPC Centers are comprehensive, but mostly oriented to performance analysis not architecture modelling
- Close collaboration and extensions with SOTA tools
  - Paraver at BSC
  - GPP Codesign at Codasip
  - Adaptyst at CERN
- Great to integrate computer architecture
  - models such as CARM!





Integration into HPC Centres

- CARM Based Application Profiling
  - Via the CARM Tool
  - Via Extrae traces and Paraver integration
- Deployment to HPC Centres
  - Karolina automated deployment complete
  - Available on EuroHPC centres soon
- CARM Dissemination Initiatives
  - POP3 Webinar
  - VI-HPS Tools Guide and Blog
    - https://www.vi-hps.org/tools/carm.html
    - https://pop-coe.eu/blog/28th-pop-webinar-the-carm-tool-cache-aware-roofline-model-for-hpc



CARM - Paraver Integration

CARM Based Application Profiling in Paraver

- Via analysis of Extrae traces
- CARM Tool provides the CARM results
- Integrated with the Paraver Tool as a separate module

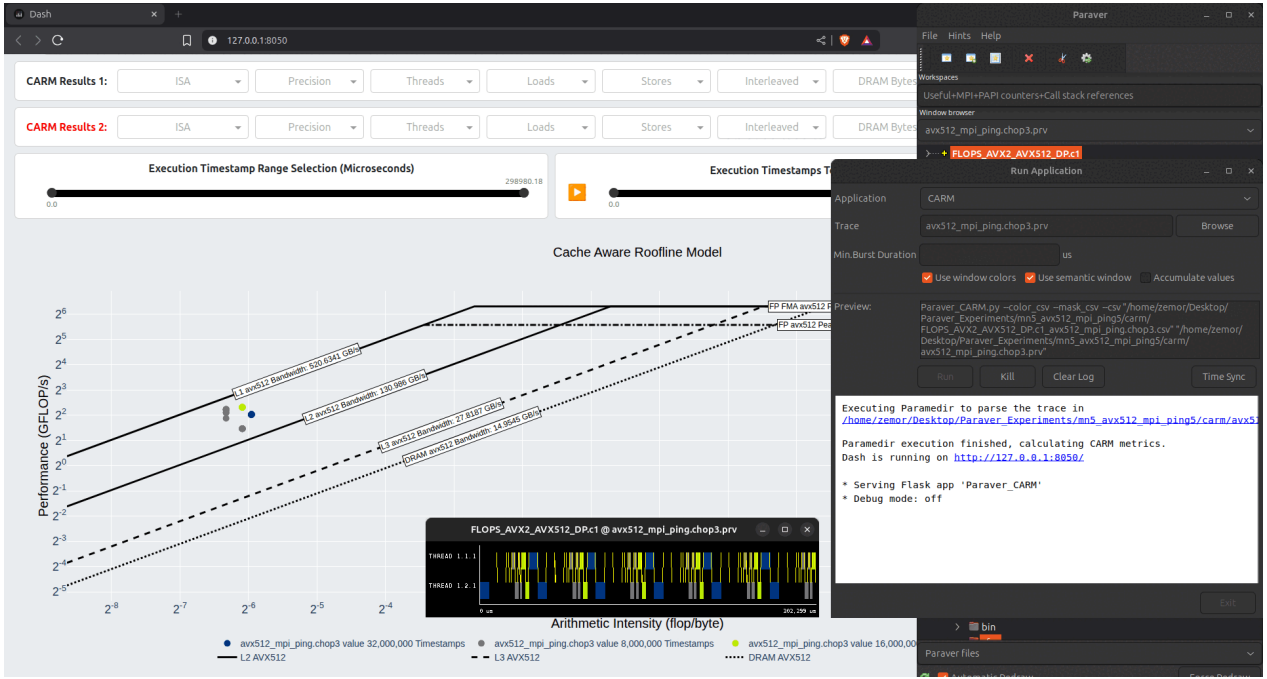
CARM-Paraver Integration Features

- Syncronization with Paraver timelines
- Exporting Paraver timestamp coloring and masks to the CARM GUI
- Exporting CARM related metrics to Paraver

CARM - Paraver Integration

Syncronization with Paraver Timelines

- Synchronization point defined via Paraver GUI
- Re-sync possible via CARM GUI



CARM-driven Codesign

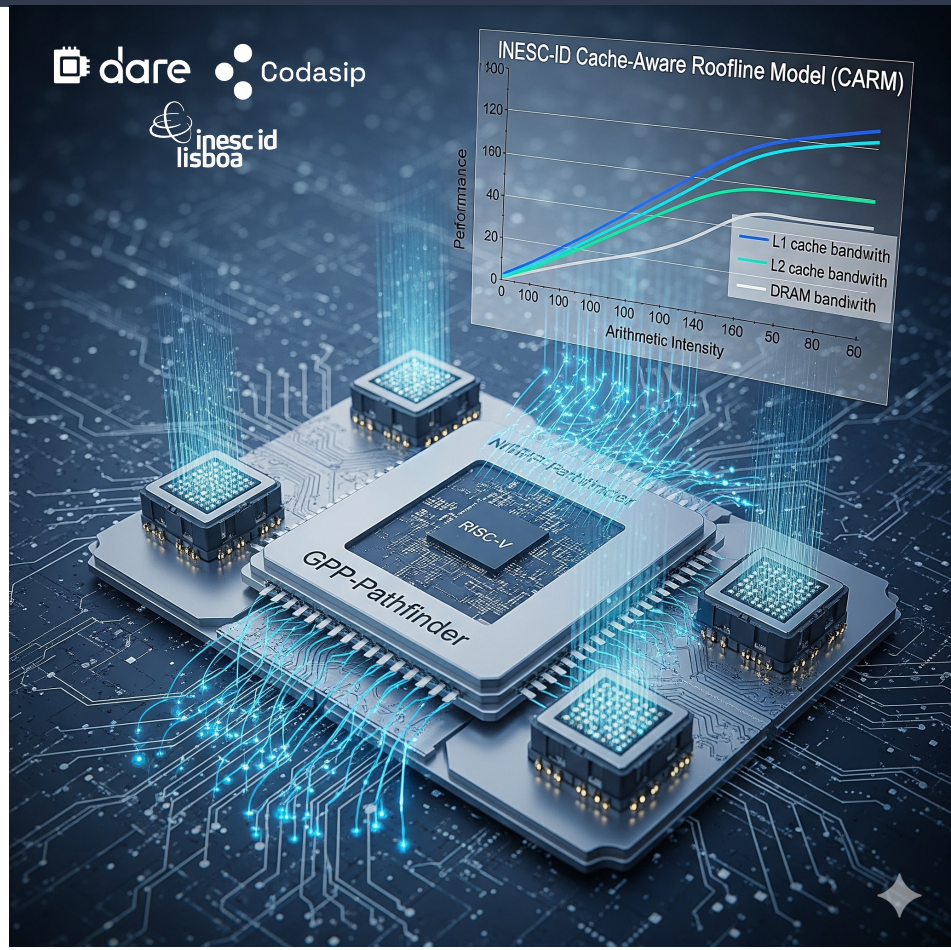
DARE SGA1 Project

- Driving Europe's next-generation HPC systems
- Focus on performance, efficiency, and sovereignty

Codasip: GPP Pathfinding

- Low-level architecture-aware support for CARM
- Help optimising the apps running on the GPP core
- Insights about characteristics of the architecture

Collaborate with the DARE SW TA



CARM Tool – Available on Github

- The CARM Tool is open source and available on Github
  - https://github.com/champ-hub/carm-roofline

- The first of many tools to be developed in the scope of CHAMP hub

CHAMP

Heterogeneous Computing and Performance Modeling Hub

Hub para Computação Heterogénea e Modelação de Performance

TÉCNICO LISBOA

Thank You

Any questions?

Aleksandar Ilic
Leonel Sousa
José Morgado
Alexandre Rodrigues
Diogo Matos

48TH VI-HPS TUNING WORKSHOP
13 February 2026