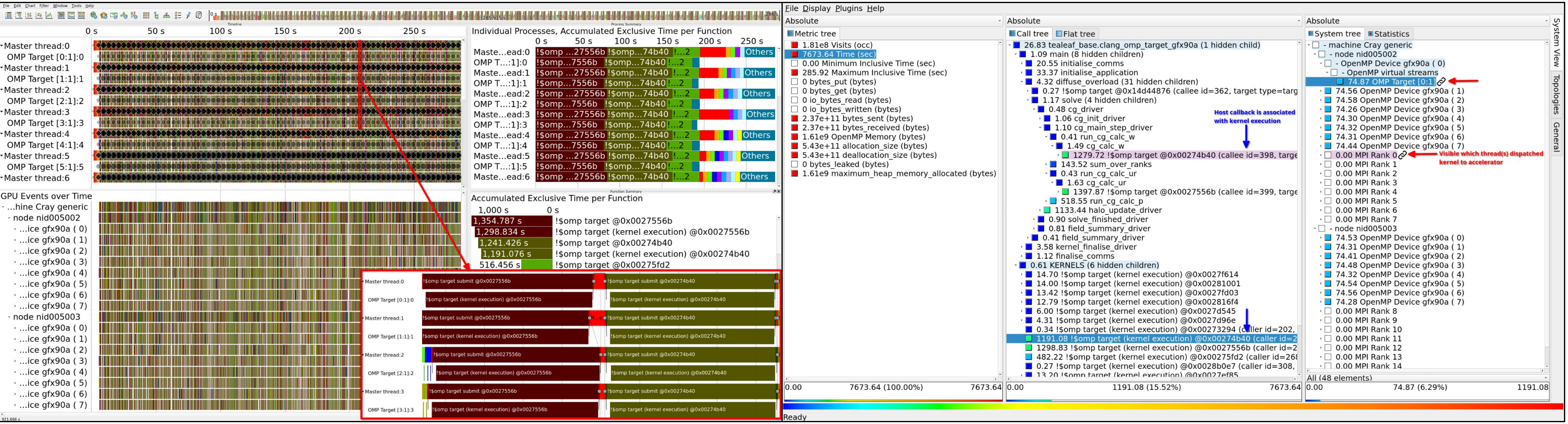

File Edit Chart Filter Window Tools Help
Timeline
0 s 50 s 100 s 150 s 200 s 250 s
Master thread:0
OMP Target [0:1]:0
Master thread:1
OMP Target [1:1]:1
Master thread:2
OMP Target [2:1]:2
Master thread:3
OMP Target [3:1]:3
Master thread:4
OMP Target [4:1]:4
Master thread:5
OMP Target [5:1]:5
Master thread:6
GPU Events over Time
...hine Cray generic
node nid005002
...ice gfx90a ( 0) ... ...ice gfx90a ( 7)
node nid005003
...ice gfx90a ( 0) ... ...ice gfx90a ( 7)
Process Summary
Individual Processes, Accumulated Exclusive Time per Function
Others
Function Summary
Accumulated Exclusive Time per Function
!$omp target submit @0x0027556b
!$omp target (kernel execution) @0x0027556b
!$omp target submit @0x00274b40
!$omp target (kernel execution) @0x00274b40

```
Accumulated Exclusive Time per Function
1,354.787 s   !$omp target @0x0027556b
1,298.834 s   !$omp target (kernel execution) @0x0027556b
1,241.426 s   !$omp target @0x00274b40
1,191.076 s   !$omp target (kernel execution) @0x00274b40
  516.456 s   !$omp target @0x00275fd2
```

```
File  Display  Plugins  Help

Absolute                                  Absolute                                              Absolute
Metric tree                               Call tree | Flat tree                                 System tree | Statistics

1.81e8 Visits (occ)                       26.83 tealeaf_base.clang_omp_target_gfx90a (1 hidden child)   - machine Cray generic
7673.64 Time (sec)                          1.09 main (8 hidden children)                                - node nid005002
0.00 Minimum Inclusive Time (sec)             20.55 initialise_comms                                       - OpenMP Device gfx90a ( 0)
285.92 Maximum Inclusive Time (sec)           33.37 initialise_application                                   - OpenMP virtual streams
0 bytes_put (bytes)                           4.32 diffuse_overload (31 hidden children)                       74.87 OMP Target [0:1]
0 bytes_get (bytes)                             0.27 !$omp target @0x14d44876 (callee id=362, target type=targ  74.56 OpenMP Device gfx90a ( 1)
0 io_bytes_read (bytes)                         1.17 solve (4 hidden children)                               74.58 OpenMP Device gfx90a ( 2)
0 io_bytes_written (bytes)                        0.48 cg_driver                                             74.26 OpenMP Device gfx90a ( 3)
2.37e+11 bytes_sent (bytes)                         1.06 cg_init_driver                                      74.30 OpenMP Device gfx90a ( 4)
2.37e+11 bytes_received (bytes)                     1.10 cg_main_step_driver                                 74.32 OpenMP Device gfx90a ( 5)
1.61e9 OpenMP Memory (bytes)                          0.41 run_cg_calc_w                                     74.31 OpenMP Device gfx90a ( 6)
5.43e+11 allocation_size (bytes)                        1.49 cg_calc_w                                       74.44 OpenMP Device gfx90a ( 7)
5.43e+11 deallocation_size (bytes)                        1279.72 !$omp target @0x00274b40 (callee id=398, targe  0.00 MPI Rank 0
0 bytes_leaked (bytes)                                143.52 sum_over_ranks                                  0.00 MPI Rank 1
1.61e9 maximum_heap_memory_allocated (bytes)          0.43 run_cg_calc_ur                                    0.00 MPI Rank 2
                                                        1.63 cg_calc_ur                                      0.00 MPI Rank 3
                                                          1397.87 !$omp target @0x0027556b (callee id=399, targe  0.00 MPI Rank 4
                                                      518.55 run_cg_calc_p                                   0.00 MPI Rank 5
                                                    1133.44 halo_update_driver                               0.00 MPI Rank 6
                                                  0.90 solve_finished_driver                                 0.00 MPI Rank 7
                                                  0.81 field_summary_driver                                - node nid005003
                                                0.41 field_summary_driver                                    74.53 OpenMP Device gfx90a ( 0)
                                              3.58 kernel_finalise_driver                                    74.31 OpenMP Device gfx90a ( 1)
                                              1.12 finalise_comms                                            74.41 OpenMP Device gfx90a ( 2)
                                            0.61 KERNELS (6 hidden children)                                 74.48 OpenMP Device gfx90a ( 3)
                                              14.70 !$omp target (kernel execution) @0x0027f614              74.32 OpenMP Device gfx90a ( 4)
                                              14.00 !$omp target (kernel execution) @0x00281001              74.54 OpenMP Device gfx90a ( 5)
                                              13.42 !$omp target (kernel execution) @0x0027fd03              74.56 OpenMP Device gfx90a ( 6)
                                              12.79 !$omp target (kernel execution) @0x002816f4              74.28 OpenMP Device gfx90a ( 7)
                                              6.00 !$omp target (kernel execution) @0x0027d545               0.00 MPI Rank 8
                                              4.31 !$omp target (kernel execution) @0x0027d96e               0.00 MPI Rank 9
                                              0.34 !$omp target (kernel execution) @0x00273294 (caller id=202,  0.00 MPI Rank 10
                                              1191.08 !$omp target (kernel execution) @0x00274b40 (caller id=2  0.00 MPI Rank 11
                                              1298.83 !$omp target (kernel execution) @0x0027556b (caller id=2  0.00 MPI Rank 12
                                              482.22 !$omp target (kernel execution) @0x00275fd2 (caller id=26  0.00 MPI Rank 13
                                              0.27 !$omp target (kernel execution) @0x0028b0e7 (caller id=308,  0.00 MPI Rank 14
                                              13.20 !$omp target (kernel execution) @0x0027ef85
                                                                                                   All (48 elements)
0.00        7673.64 (100.00%)      7673.64   0.00       1191.08 (15.52%)      7673.64   0.00      74.87 (6.29%)      1191.08
Ready
```

Host callback is associated with kernel execution

Visible which thread(s) dispatched kernel to accelerator

# 27th POP Webinar

## Performance Analysis of OpenMP Target Offloading in Score-P

2024-05-28 I Jan André Reuter

Mitglied der Helmholtz-Gemeinschaft

Score-P Scalable performance measurement infrastructure for parallel codes

JÜLICH Forschungszentrum

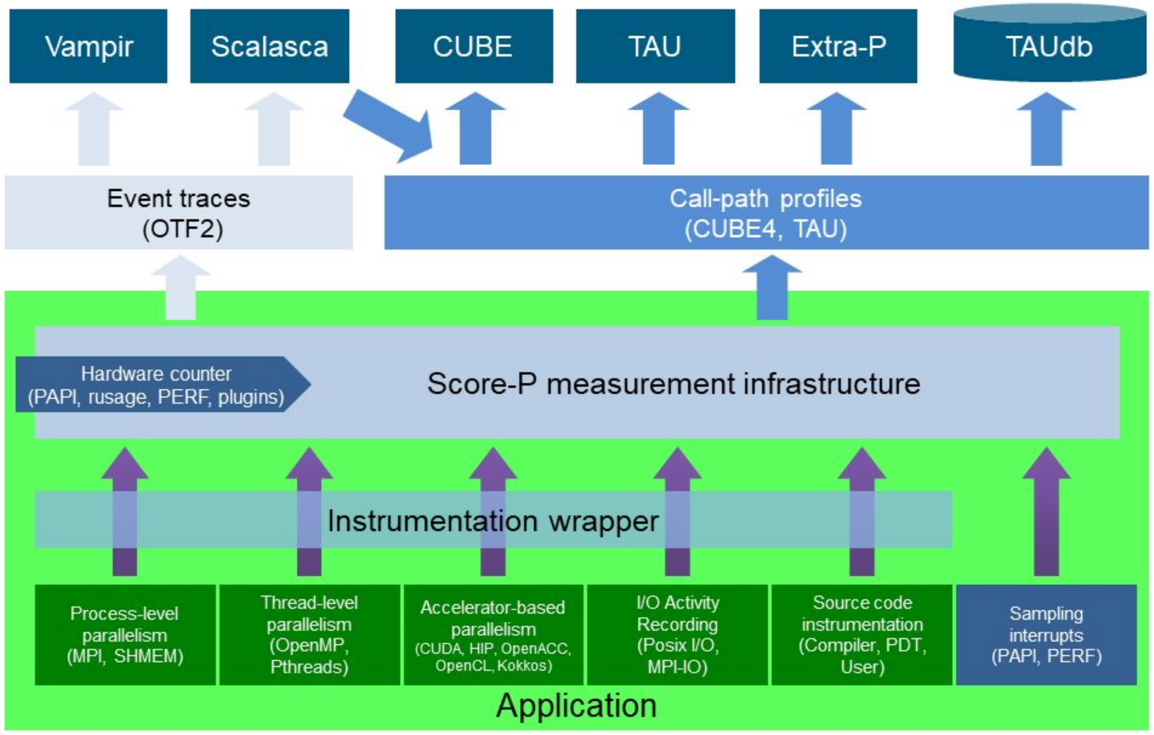
# AGENDA

- What is the Score-P infrastructure?
- The OpenMP Tools Interface and our support of it
- OpenMP and offloading
- Handling offloading events on the host
- Handling offloading events from accelerators
- Results
- What if my runtime has only limited support?
- Final words

# WHAT IS SCORE-P?

## The Score-P instrumentation and measurement infrastructure

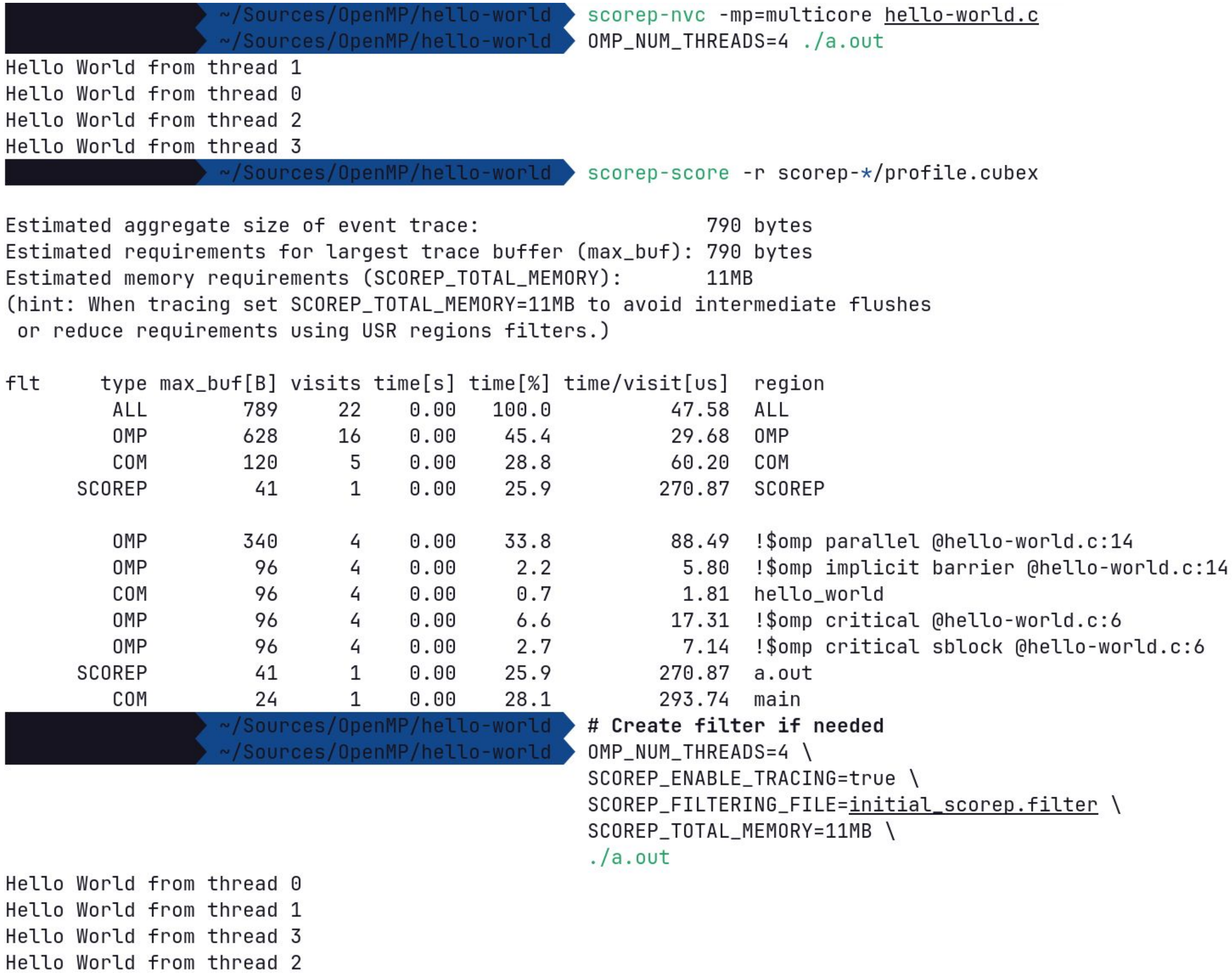
- Score-P is a highly scalable instrumentation tool
- Support for multi-process, thread-parallel and accelerator-based paradigms
- Support for additional metrics (I/O, HW counters, …)
- Flexible measurement without re-compilation:
  - Profile generation (CUBE4 .cubex format)
  - Event trace recording (OTF2 format)
- Support for C, C++, Fortran and Python


Vampir
Scalasca
CUBE
TAU
Extra-P
TAUdb
Event traces
(OTF2)
Call-path profiles
(CUBE4, TAU)
Hardware counter
(PAPI, rusage, PERF, plugins)
Score-P measurement infrastructure
Instrumentation wrapper
Process-level parallelism (MPI, SHMEM)
Thread-level parallelism (OpenMP, Pthreads)
Accelerator-based parallelism (CUDA, HIP, OpenACC, OpenCL, Kokkos)
I/O Activity Recording (Posix I/O, MPI-IO)
Source code instrumentation (Compiler, PDT, User)
Sampling interrupts (PAPI, PERF)
Application

# INSTRUMENTING APPLICATIONS

## A very high level overview on how to use Score-P

Score-P
Scalable performance measurement infrastructure for parallel codes



```
[redacted] ~/Sources/OpenMP/hello-world  scorep-nvc -mp=multicore hello-world.c
           ~/Sources/OpenMP/hello-world  OMP_NUM_THREADS=4 ./a.out
Hello World from thread 1
Hello World from thread 0
Hello World from thread 2
Hello World from thread 3
[redacted] ~/Sources/OpenMP/hello-world  scorep-score -r scorep-*/profile.cubex

Estimated aggregate size of event trace:                   790 bytes
Estimated requirements for largest trace buffer (max_buf): 790 bytes
Estimated memory requirements (SCOREP_TOTAL_MEMORY):       11MB
(hint: When tracing set SCOREP_TOTAL_MEMORY=11MB to avoid intermediate flushes
 or reduce requirements using USR regions filters.)

flt     type max_buf[B] visits time[s] time[%] time/visit[us]  region
         ALL        789     22    0.00   100.0          47.58  ALL
         OMP        628     16    0.00    45.4          29.68  OMP
         COM        120      5    0.00    28.8          60.20  COM
      SCOREP         41      1    0.00    25.9         270.87  SCOREP

         OMP        340      4    0.00    33.8          88.49  !$omp parallel @hello-world.c:14
         OMP         96      4    0.00     2.2           5.80  !$omp implicit barrier @hello-world.c:14
         COM         96      4    0.00     0.7           1.81  hello_world
         OMP         96      4    0.00     6.6          17.31  !$omp critical @hello-world.c:6
         OMP         96      4    0.00     2.7           7.14  !$omp critical sblock @hello-world.c:6
      SCOREP         41      1    0.00    25.9         270.87  a.out
         COM         24      1    0.00    28.1         293.74  main
[redacted] ~/Sources/OpenMP/hello-world  # Create filter if needed
           ~/Sources/OpenMP/hello-world  OMP_NUM_THREADS=4 \
                                         SCOREP_ENABLE_TRACING=true \
                                         SCOREP_FILTERING_FILE=initial_scorep.filter \
                                         SCOREP_TOTAL_MEMORY=11MB \
                                         ./a.out
Hello World from thread 0
Hello World from thread 1
Hello World from thread 3
Hello World from thread 2
```

Build systems like CMake may need additional steps. See `scorep-wrapper --help` for more info.

## More information

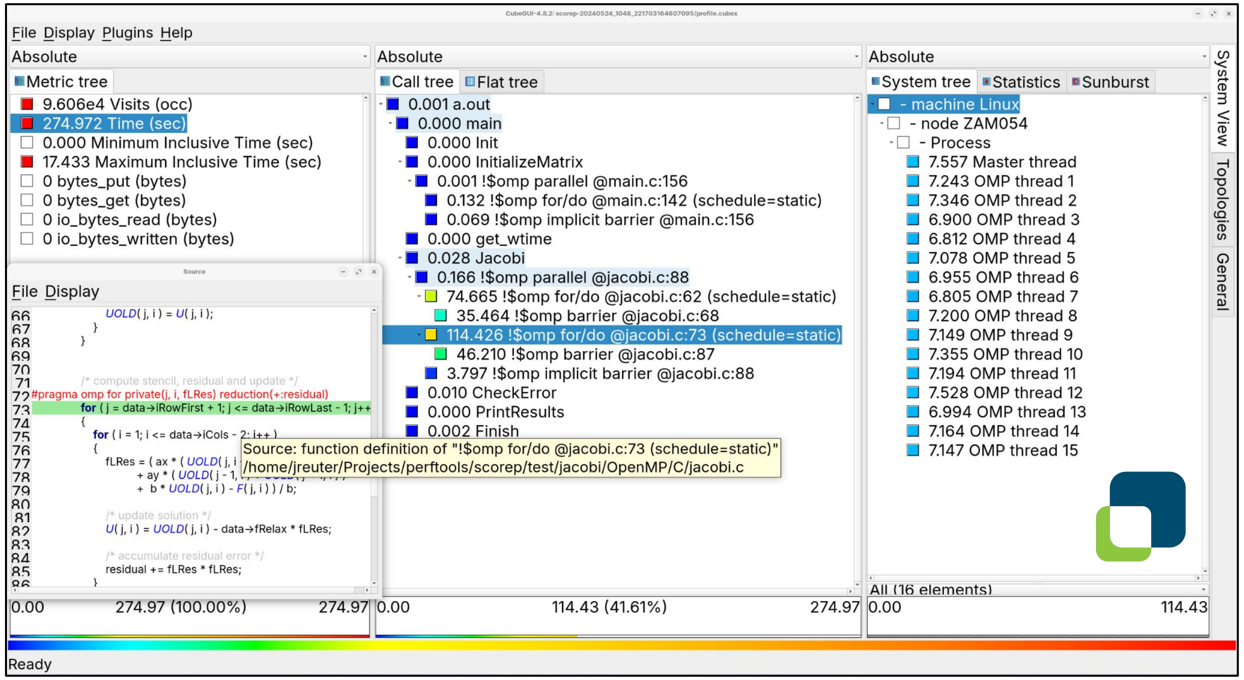
Using POP Tools: Score-P, Scalasca

Bernd Mohr

JÜLICH
Forschungszentrum

# VIEW YOUR RESULTS

## A short look at Cube and Vampir



Metric tree:

- 9.606e4 Visits (occ)
- 274.972 Time (sec)
- 0.000 Minimum Inclusive Time (sec)
- 17.433 Maximum Inclusive Time (sec)
- 0 bytes_put (bytes)
- 0 bytes_get (bytes)
- 0 io_bytes_read (bytes)
- 0 io_bytes_written (bytes)

Call tree:

```
0.001 a.out
  0.000 main
    0.000 Init
    0.000 InitializeMatrix
      0.001 !$omp parallel @main.c:156
        0.132 !$omp for/do @main.c:142 (schedule=static)
        0.069 !$omp implicit barrier @main.c:156
    0.000 get_wtime
    0.028 Jacobi
      0.166 !$omp parallel @jacobi.c:88
        74.665 !$omp for/do @jacobi.c:62 (schedule=static)
          35.464 !$omp barrier @jacobi.c:68
        114.426 !$omp for/do @jacobi.c:73 (schedule=static)
          46.210 !$omp barrier @jacobi.c:87
        3.797 !$omp implicit barrier @jacobi.c:88
    0.010 CheckError
    0.000 PrintResults
    0.002 Finish
```

System tree:

```
- machine Linux
  - node ZAM054
    - Process
      7.557 Master thread
      7.243 OMP thread 1
      7.346 OMP thread 2
      6.900 OMP thread 3
      6.812 OMP thread 4
      7.078 OMP thread 5
      6.955 OMP thread 6
      6.805 OMP thread 7
      7.200 OMP thread 8
      7.149 OMP thread 9
      7.355 OMP thread 10
      7.194 OMP thread 11
      7.528 OMP thread 12
      6.994 OMP thread 13
      7.164 OMP thread 14
      7.147 OMP thread 15
```

All (16 elements)

Source:

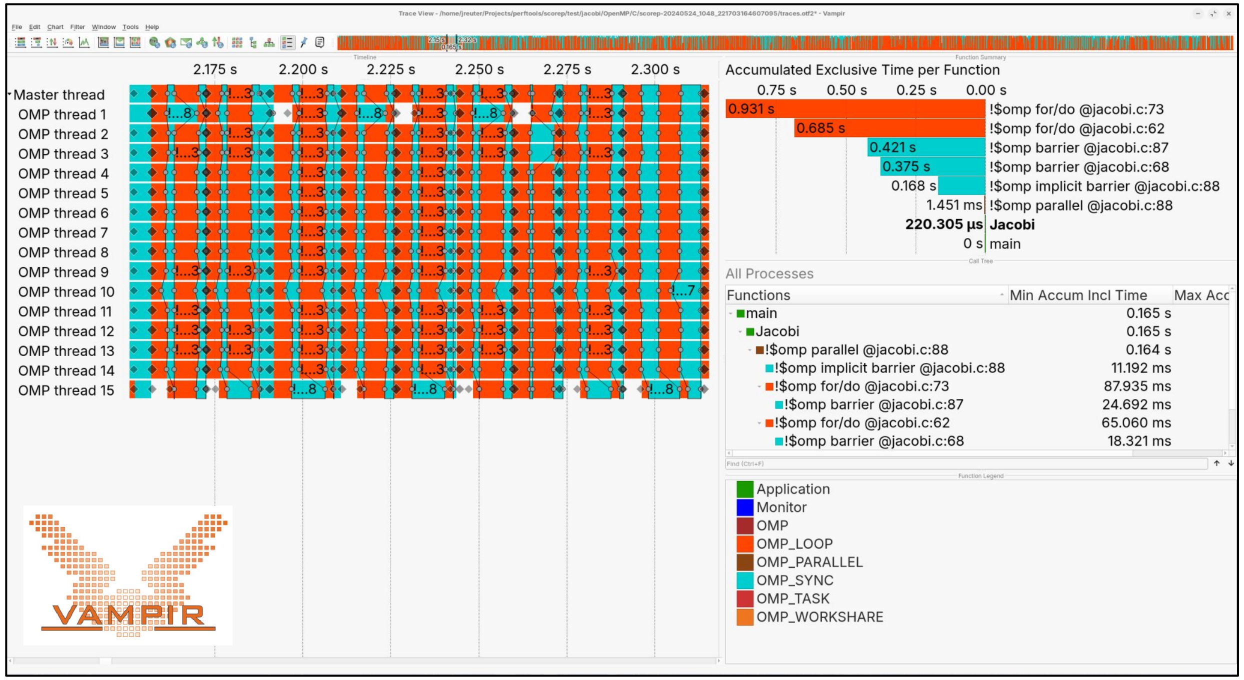
```
            UOLD( j, i ) = U( j, i );
        }
    }


    /* compute stencil, residual and update */
#pragma omp for private(j, i, fLRes) reduction(+:residual)
    for ( j = data->iRowFirst + 1; j <= data->iRowLast - 1; j++
    {
        for ( i = 1; i <= data->iCols - 2; i++ )
        {
            fLRes = ( ax * ( UOLD( j, i
                    + ay * ( UOLD( j - 1,
                    +  b * UOLD( j, i ) - F( j, i ) ) / b;

            /* update solution */
            U( j, i ) = UOLD( j, i ) - data->fRelax * fLRes;

            /* accumulate residual error */
            residual += fLRes * fLRes;
        }
```

Source: function definition of "!$omp for/do @jacobi.c:73 (schedule=static)"
/home/jreuter/Projects/perftools/scorep/test/jacobi/OpenMP/C/jacobi.c

| 0.00 | 274.97 (100.00%) | 274.97 |
|---|---|---|
| 0.00 | 114.43 (41.61%) | 274.97 |
| 0.00 |  | 114.43 |

- Open source viewer for .cubex profiles
- More information:

  `https://www.scalasca.org/scalasca/software/cube-4.x/`



Accumulated Exclusive Time per Function

| Time | Function |
|---|---|
| 0.931 s | !$omp for/do @jacobi.c:73 |
| 0.685 s | !$omp for/do @jacobi.c:62 |
| 0.421 s | !$omp barrier @jacobi.c:87 |
| 0.375 s | !$omp barrier @jacobi.c:68 |
| 0.168 s | !$omp implicit barrier @jacobi.c:88 |
| 1.451 ms | !$omp parallel @jacobi.c:88 |
| **220.305 µs** | **Jacobi** |
| 0 s | main |

All Processes

| Functions | Min Accum Incl Time | Max Acc |
|---|---|---|
| main | 0.165 s | |
| Jacobi | 0.165 s | |
| !$omp parallel @jacobi.c:88 | 0.164 s | |
| !$omp implicit barrier @jacobi.c:88 | 11.192 ms | |
| !$omp for/do @jacobi.c:73 | 87.935 ms | |
| !$omp barrier @jacobi.c:87 | 24.692 ms | |
| !$omp for/do @jacobi.c:62 | 65.060 ms | |
| !$omp barrier @jacobi.c:68 | 18.321 ms | |

- Application
- Monitor
- OMP
- OMP_LOOP
- OMP_PARALLEL
- OMP_SYNC
- OMP_TASK
- OMP_WORKSHARE

- Commercial viewer for .otf2 traces
- More information:

  `https://vampir.eu`

# OPENMP SUPPORT IN SCORE-P

## Two ways to collect information about OpenMP

opari2

- Source-to-source instrumentation tool
- Independent from compiler used
- Instrumentation up to OpenMP 3.x
- Various limitations
  - Code sometimes has to be prepared for OPARI2

# OPENMP SUPPORT IN SCORE-P

## Two ways to collect information about OpenMP

opari2

- Source-to-source instrumentation tool
- Independent from compiler used
- Instrumentation up to OpenMP 3.x
- Various limitations
  - Code sometimes has to be prepared for OPARI2

OpenMP

OpenMP Tools Interface

- Standardised tool interface since OpenMP 5.0
- Enables development of tools using any implementation of the OpenMP API
- Support for the latest and greatest OpenMP features
- Continuously expanded with new versions

# WHAT IS THE OPENMP TOOLS INTERFACE?

- Interface for *first-party* tools, linked or loaded into the OpenMP program

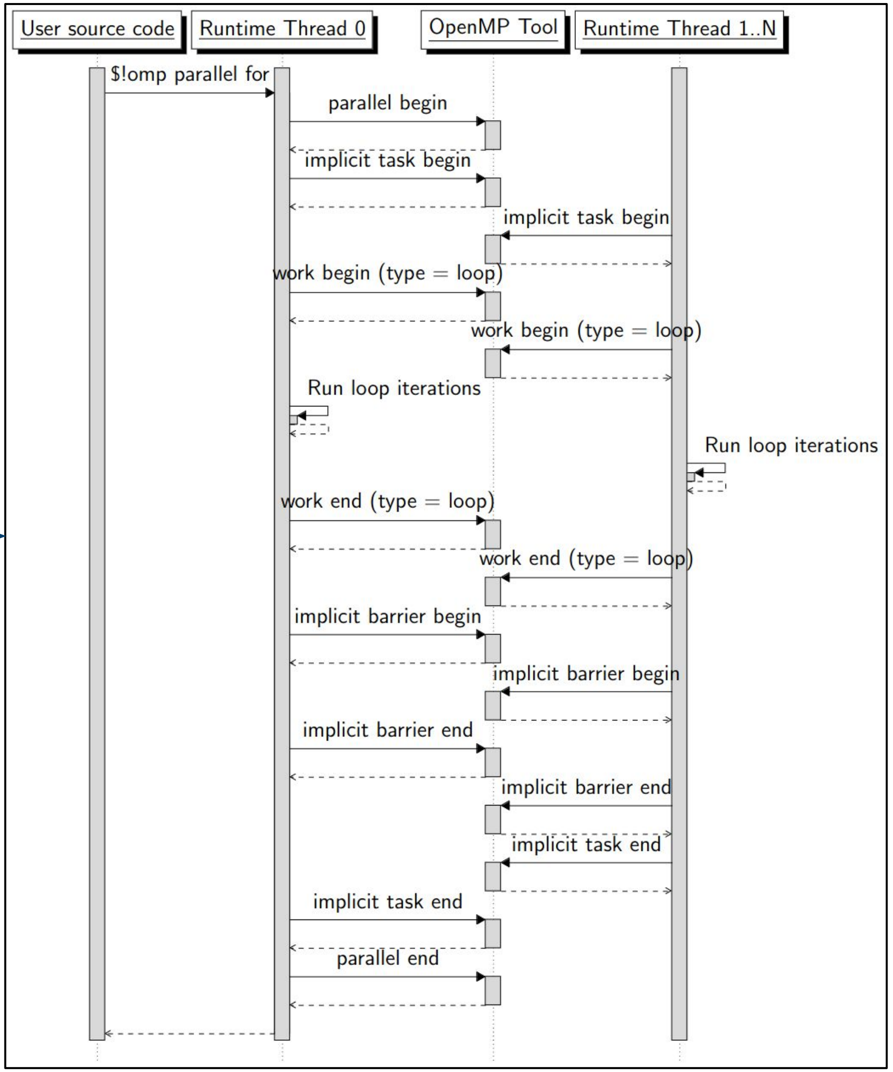
```
$ clang -fopenmp my-program.c -lmy-tool
$ clang -fopenmp my-program.c my-tool.c
$ OMP_TOOL_LIBRARIES=my-tool.so ./my-program
```

- Defined by the OpenMP standard, implemented by OpenMP runtimes
- Tools have to implement functions to interact with OMPT (e.g. `ompt_start_tool`)
- One interaction method: *callbacks*, invoked for runtime events

# OPENMP TOOLS INTERFACE CALLBACKS

## Just a part of what tools see for user code

```
int main( void )
{
    const int N = 100;
    #pragma omp parallel for
    for( int i = 0; i < N; ++i)
    {
        // Some code...
    }
    return 0;
}
```



- Tools receive a lot of events for user code
- With this, we are able to record events for OpenMP directives

# SCORE-P AND OMPT: THE PRESENT …

Score-P
Scalable performance measurement infrastructure for parallel codes

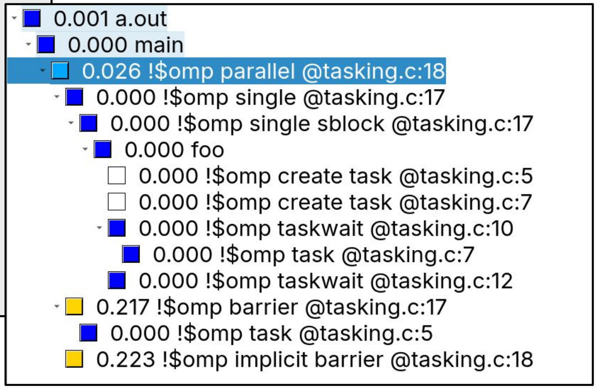
- First support in version 8.0 (released in December 2022)
- Selectable via `scorep --thread=omp:ompt`
- Support tried to match the available features in OPARI2 (focused on OpenMP 3.x)
- Small feature additions and several bug fixes in 8.x:
  - Recording loop schedules
  - Recording `omp_test_lock` events

| OpenMP directive | Support |
|---|---|
| `OpenMP 3.x` | yes |
| `cancel` | no |
| `task depend` | no |
| `task detach` | no |
| `taskgroup` | yes |
| `taskloop` | no |
| `teams` | no |
| `scope` | no |

# … AND THE FUTURE (V9.0)

- Current OpenMP features planned for Score-P v9.0
  - Improvements to support of `task` directives

```c
#include <stdio.h>

void foo() {
   int x = 0, y = 2;
   #pragma omp task depend( inout: x ) shared( x )
   x++;                                         // 1st child task
   #pragma omp task shared( y )
   y--;                                         // 2nd child task
   #pragma omp taskwait depend( in: x )         // 1st taskwait
   printf( "x=%d\n", x );
   #pragma omp taskwait                         // 2nd taskwait
   printf( "y=%d\n", y );
}

int main() {
   #pragma omp parallel
   #pragma omp single
   foo();
   return 0;
}
```

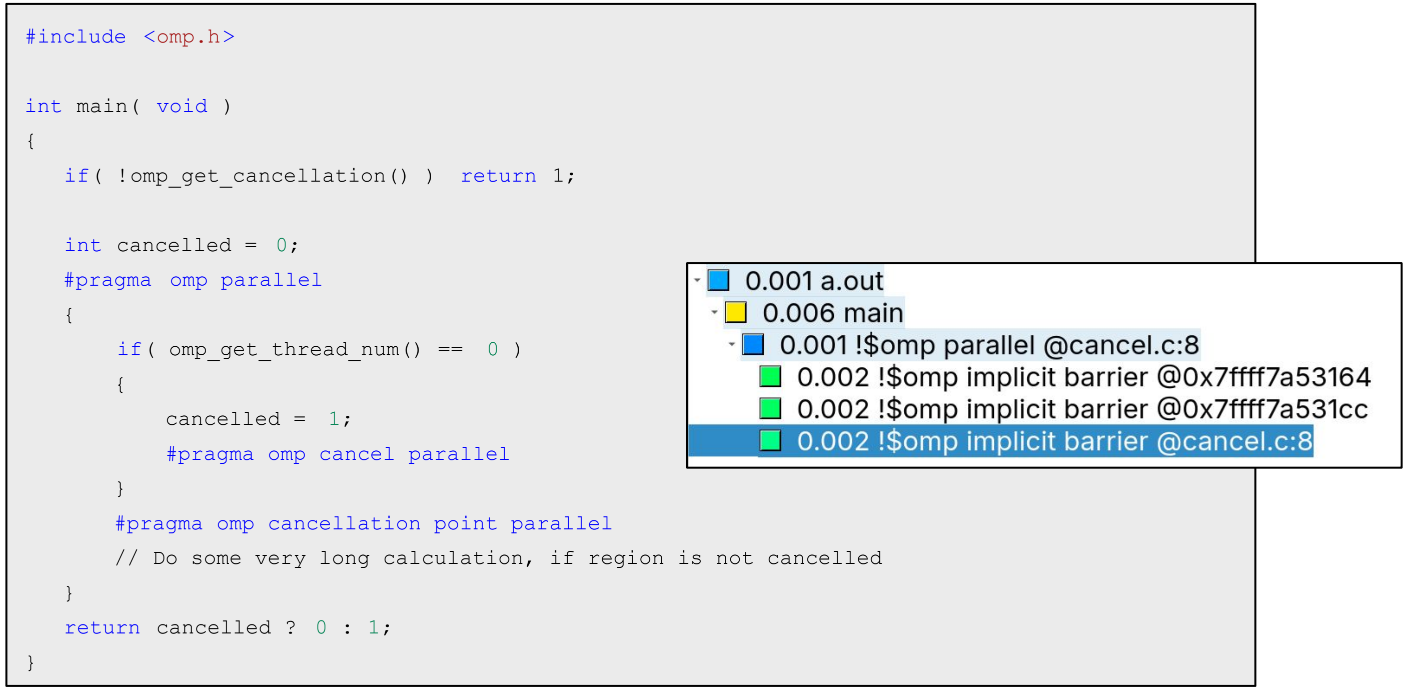


```
■ 0.001 a.out
  ■ 0.000 main
    ■ 0.026 !$omp parallel @tasking.c:18
      ■ 0.000 !$omp single @tasking.c:17
        ■ 0.000 !$omp single sblock @tasking.c:17
          ■ 0.000 foo
            □ 0.000 !$omp create task @tasking.c:5
            □ 0.000 !$omp create task @tasking.c:7
            ■ 0.000 !$omp taskwait @tasking.c:10
              ■ 0.000 !$omp task @tasking.c:7
            ■ 0.000 !$omp taskwait @tasking.c:12
      🟨 0.217 !$omp barrier @tasking.c:17
        ■ 0.000 !$omp task @tasking.c:5
      🟨 0.223 !$omp implicit barrier @tasking.c:18
```

| OpenMP directive | Support |
|---|---|
| `OpenMP 3.x` | yes |
| `cancel` | partial |
| `task depend` | partial |
| `task detach` | yes |
| `taskgroup` | yes |
| `taskloop` | yes |
| `teams` | yes |
| `scope` | no |

# … AND THE FUTURE (V9.0)

- Current OpenMP features planned for Score-P v9.0
  - Improvements to support of `task` directives
  - Improved support for `cancel` directive



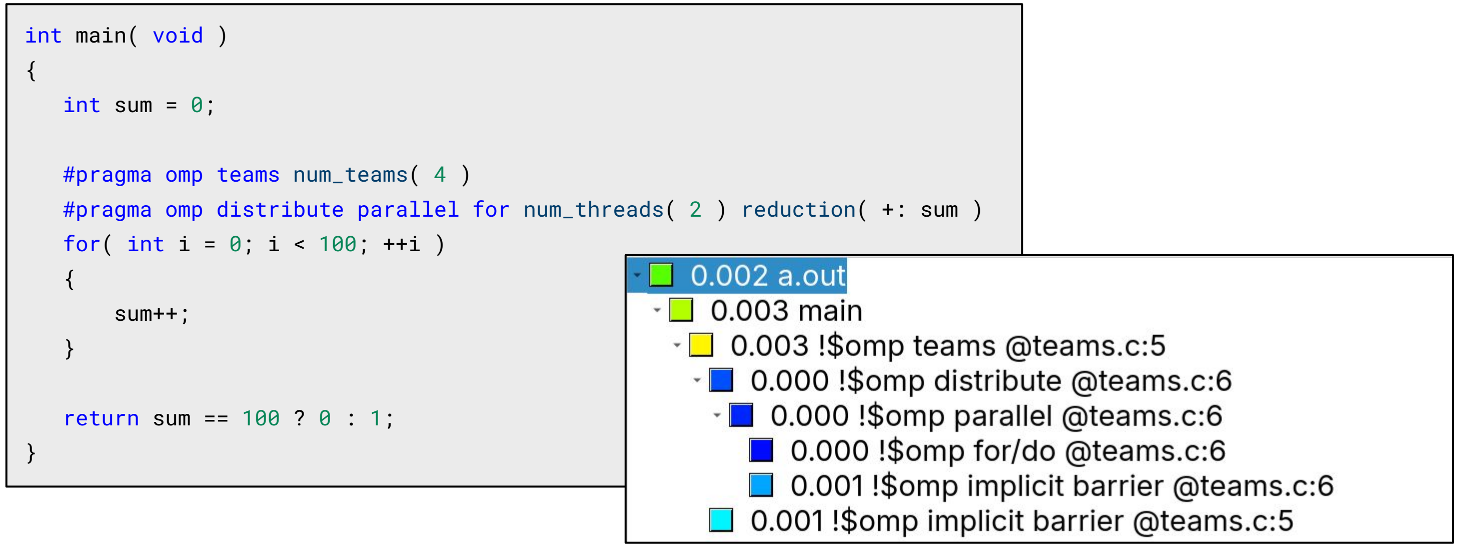
```
#include <omp.h>

int main( void )
{
   if( !omp_get_cancellation() )  return 1;

   int cancelled = 0;
   #pragma omp parallel
   {
       if( omp_get_thread_num() ==  0 )
       {
           cancelled =  1;
           #pragma omp cancel parallel
       }
       #pragma omp cancellation point parallel
       // Do some very long calculation, if region is not cancelled
   }
   return cancelled ? 0 : 1;
}
```

```
0.001 a.out
  0.006 main
    0.001 !$omp parallel @cancel.c:8
      0.002 !$omp implicit barrier @0x7ffff7a53164
      0.002 !$omp implicit barrier @0x7ffff7a531cc
      0.002 !$omp implicit barrier @cancel.c:8
```

| OpenMP directive | Support |
|---|---|
| `OpenMP 3.x` | yes |
| `cancel` | partial |
| `task depend` | partial |
| `task detach` | yes |
| `taskgroup` | yes |
| `taskloop` | yes |
| `teams` | yes |
| `scope` | no |

# … AND THE FUTURE (V9.0)

- Current OpenMP features planned for Score-P v9.0
  - Improvements to support of `task` directives
  - Improved support for `cancel` directive
  - Support for `teams` directive



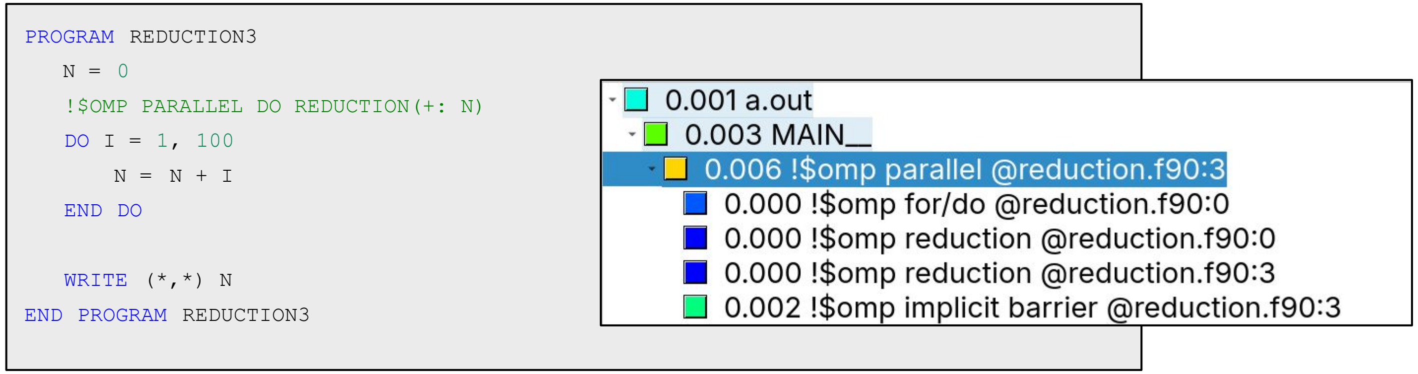
```
int main( void )
{
   int sum = 0;

   #pragma omp teams num_teams( 4 )
   #pragma omp distribute parallel for num_threads( 2 ) reduction( +: sum )
   for( int i = 0; i < 100; ++i )
   {
       sum++;
   }

   return sum == 100 ? 0 : 1;
}
```

```
0.002 a.out
  0.003 main
    0.003 !$omp teams @teams.c:5
      0.000 !$omp distribute @teams.c:6
        0.000 !$omp parallel @teams.c:6
          0.000 !$omp for/do @teams.c:6
          0.001 !$omp implicit barrier @teams.c:6
      0.001 !$omp implicit barrier @teams.c:5
```

| OpenMP directive | Support |
|---|---|
| `OpenMP 3.x` | yes |
| `cancel` | partial |
| `task depend` | partial |
| `task detach` | yes |
| `taskgroup` | yes |
| `taskloop` | yes |
| `teams` | yes |
| `scope` | no |

# … AND THE FUTURE (V9.0)

- Current OpenMP features planned for Score-P v9.0
  - Improvements to support of `task` directives
  - Improved support for `cancel` directive
  - Support for `teams` directive
  - Recording of `reduction` clause



```
PROGRAM REDUCTION3
   N = 0
   !$OMP PARALLEL DO REDUCTION(+: N)
   DO I = 1, 100
       N = N + I
   END DO

   WRITE (*,*) N
END PROGRAM REDUCTION3
```

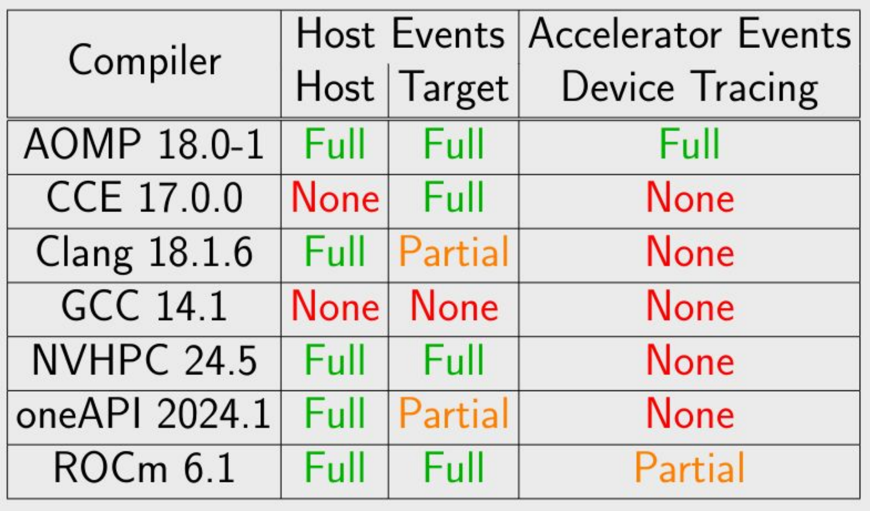
```
▾ 🟦 0.001 a.out
  ▾ 🟩 0.003 MAIN__
    ▾ 🟨 0.006 !$omp parallel @reduction.f90:3
        🟦 0.000 !$omp for/do @reduction.f90:0
        🟦 0.000 !$omp reduction @reduction.f90:0
        🟦 0.000 !$omp reduction @reduction.f90:3
        🟩 0.002 !$omp implicit barrier @reduction.f90:3
```

| OpenMP directive | Support |
|---|---|
| `OpenMP 3.x` | yes |
| `cancel` | partial |
| `task depend` | partial |
| `task detach` | yes |
| `taskgroup` | yes |
| `taskloop` | yes |
| `teams` | yes |
| `scope` | no |

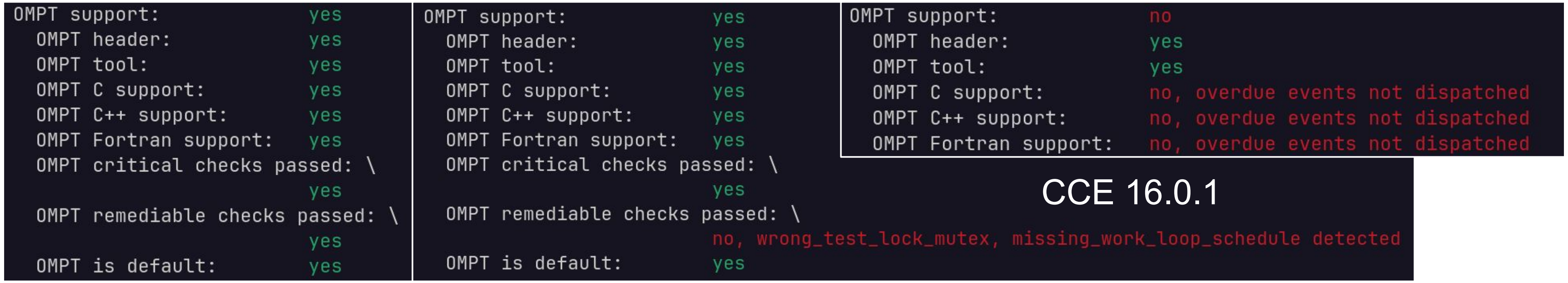
# COMPILER SUPPORT FOR OMPT

## We do have strict requirements

- OpenMP runtimes have to implement the tools interface
- However, runtimes may have bugs or features not fully implemented
- We test the OpenMP runtime during configuration to prepare for known issues



| Compiler | Host Events | | Accelerator Events |
|---|---|---|---|
| | Host | Target | Device Tracing |
| AOMP 18.0-1 | Full | Full | Full |
| CCE 17.0.0 | None | Full | None |
| Clang 18.1.6 | Full | Partial | None |
| GCC 14.1 | None | None | None |
| NVHPC 24.5 | Full | Full | None |
| oneAPI 2024.1 | Full | Partial | None |
| ROCm 6.1 | Full | Full | Partial |



```
OMPT support:                 yes
  OMPT header:                yes
  OMPT tool:                  yes
  OMPT C support:             yes
  OMPT C++ support:           yes
  OMPT Fortran support:       yes
  OMPT critical checks passed: \
                              yes
  OMPT remediable checks passed: \
                              yes
  OMPT is default:            yes
```

NVHPC 24.5

```
OMPT support:                 yes
  OMPT header:                yes
  OMPT tool:                  yes
  OMPT C support:             yes
  OMPT C++ support:           yes
  OMPT Fortran support:       yes
  OMPT critical checks passed: \
                              yes
  OMPT remediable checks passed: \
                              no, wrong_test_lock_mutex, missing_work_loop_schedule detected
  OMPT is default:            yes
```

oneAPI 2024.1

```
OMPT support:                 no
  OMPT header:                yes
  OMPT tool:                  yes
  OMPT C support:             no, overdue events not dispatched
  OMPT C++ support:           no, overdue events not dispatched
  OMPT Fortran support:       no, overdue events not dispatched
```

CCE 16.0.1

# RIGOROUS TESTING

## Ensuring that Score-P is able to work with compiler runtimes

- Large internal test suite, based on OpenMP examples and additional smoke tests
- Score-P is regularly tested with GCC, LLVM/Clang, NVHPC, ROCm & oneAPI
- Allows easy testing of new compilers as soon as they release
  - NVHPC 24.5, released on May 22nd, did show differences in runtime for example
- Contributed to several bug reports to compiler vendors (~65 bugs since Dec. 2022) and workarounds in Score-P

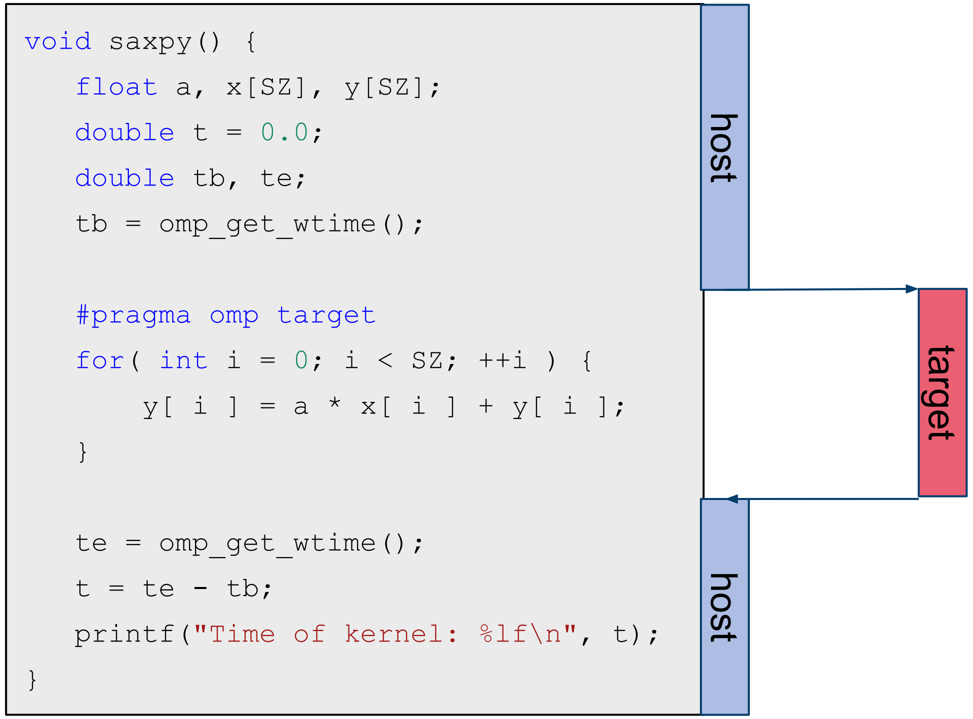
| | C | C++ | Fortran |
|---|---|---|---|
| Host Examples | 138 | 7 | 165 |
| Offload Examples | 63 | 6 | 63 |
| Teams Smoke Tests | 30 | 0 | 0 |
| General Smoke Tests | 42 | 4 | 0 |

# HOW TO HANDLE OFFLOADING TO ACCELERATORS?

# OFFLOADING WORK TO ACCELERATORS

## What is offloading in OpenMP?

- OpenMP introduced `target` directives in OpenMP 4.0
- Expanded in later spec. versions, including functions like `omp_target_alloc`



```
void saxpy() {
   float a, x[SZ], y[SZ];
   double t = 0.0;
   double tb, te;
   tb = omp_get_wtime();

   #pragma omp target
   for( int i = 0; i < SZ; ++i ) {
      y[ i ] = a * x[ i ] + y[ i ];
   }

   te = omp_get_wtime();
   t = te - tb;
   printf("Time of kernel: %lf\n", t);
}
```

More information:

26th POP Webinar – Asynchronous GPU Programming in OpenMP

Tuesday, 30 April 2024, 15:00 CEST

Christian Terboven

RWTH Aachen

Michael Klemm

AMD

# OFFLOADING WORK TO ACCELERATORS

## Splitting data transfers and kernels

```
void saxpy( float a, float* x, float* y, int n ) {
   #pragma omp target teams distribute parallel  for
   for( int i = 0; i < n; ++i ) {
       y[ i ] = a * x[ i ] + y[ i ];
   }
}

void vecadd( float* x, float* y, float* data_out, int n )
{
   #pragma omp target teams distribute parallel  for
   for( int i = 0; i < n; ++i ) {
       data_out[ i ] = x[ i ] + y[ i ];
   }
}

void example() {
   float a, x[N], y[N], b[N], data_out[N];

   #pragma omp target data map ( to: x[:N], y[:N], b[:N], a ) \
                            map( from: data_out[:N] )
   {
       saxpy( a, x, y, N );
       vecadd( y, b, data_out, N );
   }
}
```

Simple optimizations:

- Reduce numbers of data transfers
- Reuse data, if possible
- Transfer data beforehand, maybe asynchronously

More information:

26th POP Webinar – Asynchronous GPU Programming in OpenMP

Tuesday, 30 April 2024, 15:00 CEST

Christian Terboven

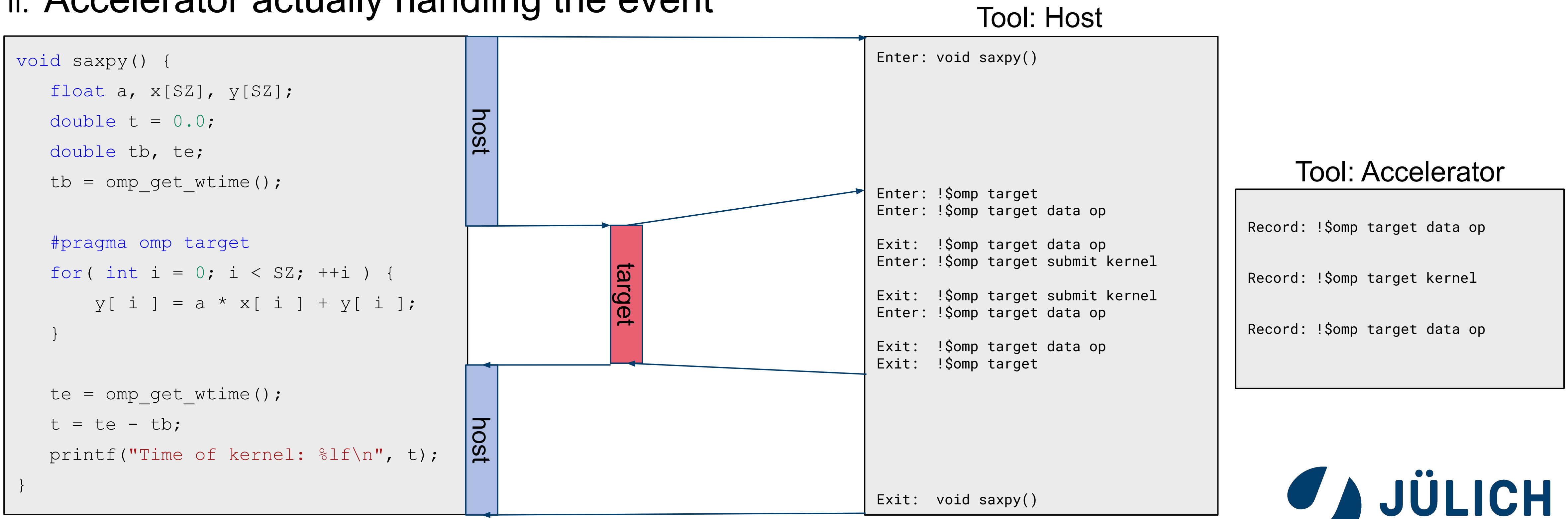
RWTH Aachen

Michael Klemm

AMD

# ACCELERATOR EVENTS AND TOOLS

- Unlike host directives, offloading needs to be handled twice to get all information
  1. Host side dispatching events and waiting for completion
  2. Accelerator actually handling the event



```c
void saxpy() {
   float a, x[SZ], y[SZ];
   double t = 0.0;
   double tb, te;
   tb = omp_get_wtime();

   #pragma omp target
   for( int i = 0; i < SZ; ++i ) {
       y[ i ] = a * x[ i ] + y[ i ];
   }

   te = omp_get_wtime();
   t = te - tb;
   printf("Time of kernel: %lf\n", t);
}
```

Tool: Host

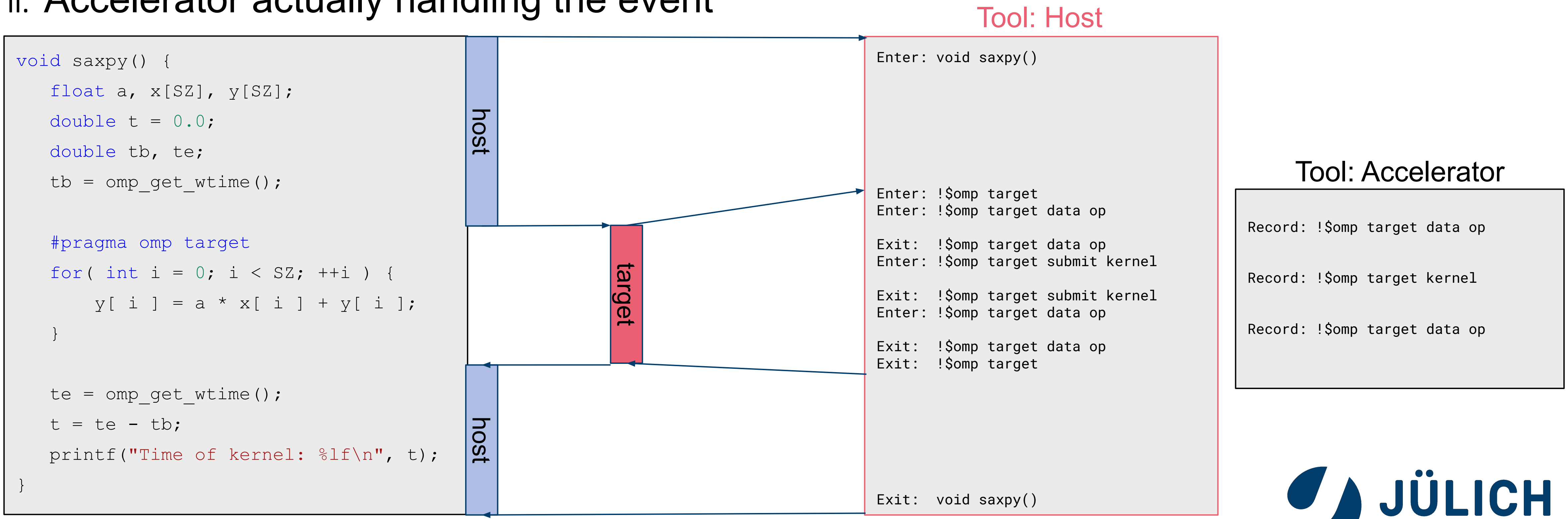
```
Enter: void saxpy()




Enter: !$omp target
Enter: !$omp target data op

Exit:  !$omp target data op
Enter: !$omp target submit kernel

Exit:  !$omp target submit kernel
Enter: !$omp target data op

Exit:  !$omp target data op
Exit:  !$omp target




Exit:  void saxpy()
```

Tool: Accelerator

```
Record: !$omp target data op

Record: !$omp target kernel

Record: !$omp target data op
```

# ACCELERATOR EVENTS AND TOOLS

- Unlike host directives, offloading needs to be handled twice to get all information
  1. Host side dispatching events and waiting for completion
  2. Accelerator actually handling the event



```
void saxpy() {
   float a, x[SZ], y[SZ];
   double t = 0.0;
   double tb, te;
   tb = omp_get_wtime();

   #pragma omp target
   for( int i = 0; i < SZ; ++i ) {
       y[ i ] = a * x[ i ] + y[ i ];
   }

   te = omp_get_wtime();
   t = te - tb;
   printf("Time of kernel: %lf\n", t);
}
```

Tool: Host

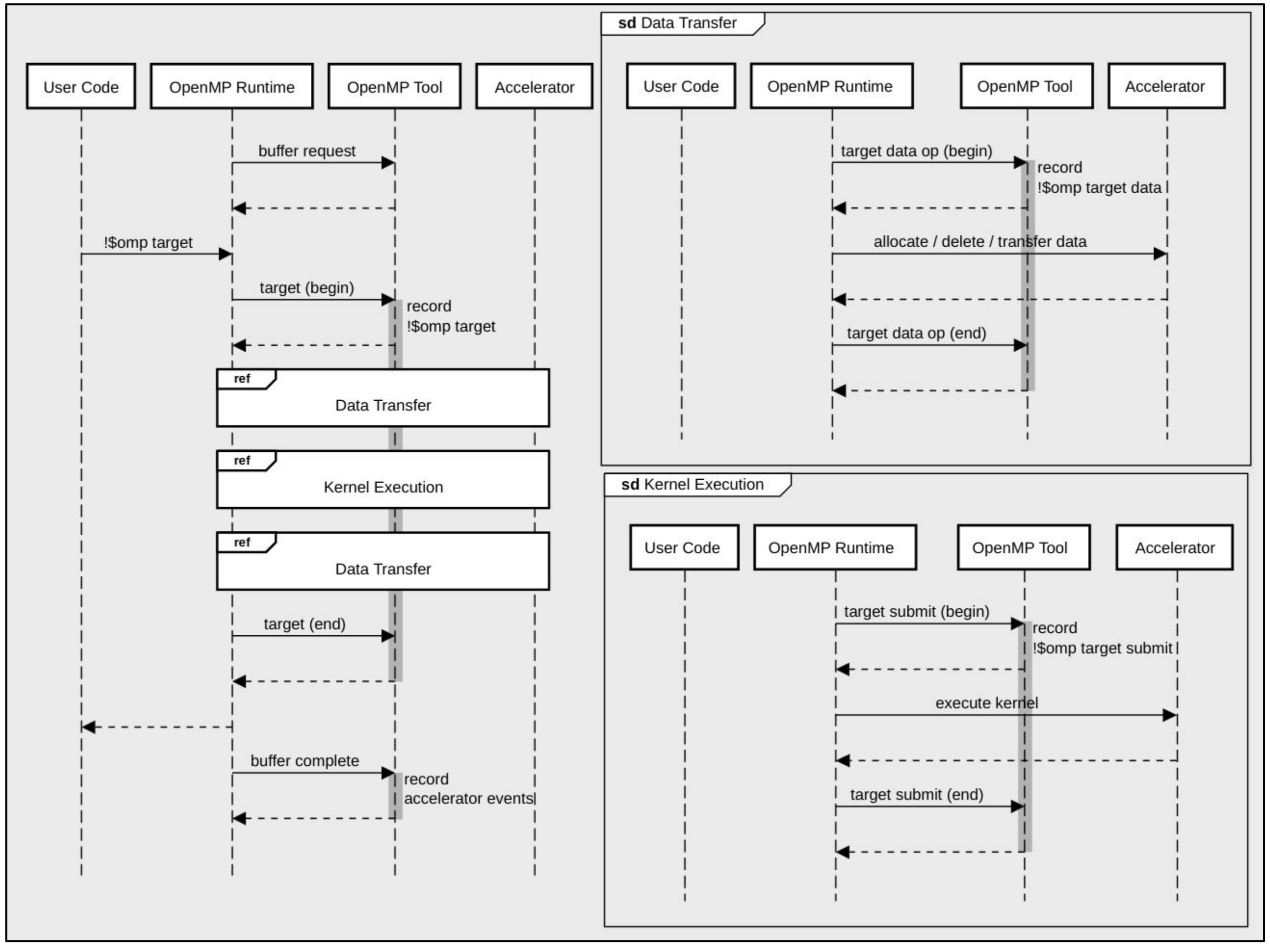
```
Enter: void saxpy()

Enter: !$omp target
Enter: !$omp target data op

Exit:  !$omp target data op
Enter: !$omp target submit kernel

Exit:  !$omp target submit kernel
Enter: !$omp target data op

Exit:  !$omp target data op
Exit:  !$omp target

Exit:  void saxpy()
```

Tool: Accelerator

```
Record: !$omp target data op

Record: !$omp target kernel

Record: !$omp target data op
```

# TARGET DIRECTIVES AND CALLBACKS

- Host side events are similar to existing ones (e.g. `parallel_begin`)
- We receive one begin and end event for the following scenarios:
  - `target` directives
  - data transfers
  - submit of a kernel
- This includes directives and function calls



User Code
OpenMP Runtime
OpenMP Tool
Accelerator
buffer request
!$omp target
target (begin)
record
!$omp target
ref Data Transfer
ref Kernel Execution
ref Data Transfer
target (end)
buffer complete
record
accelerator events

sd Data Transfer
User Code
OpenMP Runtime
OpenMP Tool
Accelerator
target data op (begin)
record
!$omp target data
allocate / delete / transfer data
target data op (end)

sd Kernel Execution
User Code
OpenMP Runtime
OpenMP Tool
Accelerator
target submit (begin)
record
!$omp target submit
execute kernel
target submit (end)

# TARGET DIRECTIVES AND CALLBACKS

## Getting more into the details

```
typedef void (*ompt_callback_target_emi_t) (
    ompt_target_t         kind,
    ompt_scope_endpoint_t endpoint,
    int                   device_num,
    ompt_data_t*          task_data,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    const void*           codeptr_ra );
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

```
typedef void (*ompt_callback_target_data_op_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    ompt_target_data_op_t optype,
    void*                 src_addr,
    int                   src_device_num,
    void*                 dest_addr,
    int                   dest_device_num,
    size_t                bytes,
    const void*           codeptr_ra );
```

- Three callbacks, giving us the important information

# TARGET DIRECTIVES AND CALLBACKS

## Getting more into the details

```
typedef void (*ompt_callback_target_emi_t) (
    ompt_target_t         kind,
    ompt_scope_endpoint_t endpoint,
    int                   device_num,
    ompt_data_t*          task_data,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    const void*           codeptr_ra );
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

```
typedef void (*ompt_callback_target_data_op_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    ompt_target_data_op_t optype,
    void*                 src_addr,
    int                   src_device_num,
    void*                 dest_addr,
    int                   dest_device_num,
    size_t                bytes,
    const void*           codeptr_ra );
```

- Three callbacks, giving us the important information
- Start / end of the operation

# TARGET DIRECTIVES AND CALLBACKS

## Getting more into the details

```
typedef void (*ompt_callback_target_emi_t) (
    ompt_target_t          kind,
    ompt_scope_endpoint_t endpoint,
    int                   device_num,
    ompt_data_t*          task_data,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    const void*           codeptr_ra );
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

```
typedef void (*ompt_callback_target_data_op_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    ompt_target_data_op_t optype,
    void*                 src_addr,
    int                   src_device_num,
    void*                 dest_addr,
    int                   dest_device_num,
    size_t                bytes,
    const void*           codeptr_ra );
```

- Three callbacks, giving us the important information
- Start / end of the operation
- Information about what exactly is done

# TARGET DIRECTIVES AND CALLBACKS

## Getting more into the details

```
typedef void (*ompt_callback_target_emi_t) (
    ompt_target_t         kind,
    ompt_scope_endpoint_t endpoint,
    int                   device_num,
    ompt_data_t*          task_data,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    const void*           codeptr_ra );
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

```
typedef void (*ompt_callback_target_data_op_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    ompt_target_data_op_t optype,
    void*                 src_addr,
    int                   src_device_num,
    void*                 dest_addr,
    int                   dest_device_num,
    size_t                bytes,
    const void*           codeptr_ra );
```

- Three callbacks, giving us the important information
- Start / end of the operation
- Information about what exactly is done
- Source code position

# TARGET DIRECTIVES AND CALLBACKS

## Getting more into the details

```
typedef void (*ompt_callback_target_emi_t) (
    ompt_target_t          kind,
    ompt_scope_endpoint_t  endpoint,
    int                    device_num,
    ompt_data_t*           task_data,
    ompt_data_t*           target_task_data,
    ompt_data_t*           target_data,
    const void*            codeptr_ra );
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

```
typedef void (*ompt_callback_target_data_op_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    ompt_target_data_op_t optype,
    void*                 src_addr,
    int                   src_device_num,
    void*                 dest_addr,
    int                   dest_device_num,
    size_t                bytes,
    const void*           codeptr_ra );
```

- Three callbacks, giving us the important information
- Start / end of the operation
- Information about what exactly is done
- Source code position
- Unique information per target region and operation

# CORRELATION OF EVENTS

- We do need to correlate two things:
  - Host callbacks between each other
  - Host callbacks to accelerator events
- Done via `target_data` and `host_op_id`
- What needs to be transferred?

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

# CORRELATION OF EVENTS

- We do need to correlate two things:
  - Host callbacks between each other
  - Host callbacks to accelerator events
- Done via `target_data` and `host_op_id`
- What needs to be transferred?

```
typedef struct scorep_ompt_target_data_t {
    const void* codeptr_ra;
    ompt_id_t   target_id;
    bool        supports_device_tracing;
} scorep_ompt_target_data;
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

# CORRELATION OF EVENTS

- We do need to correlate two things:
  - Host callbacks between each other
  - Host callbacks to accelerator events
- Done via `target_data` and `host_op_id`
- What needs to be transferred?

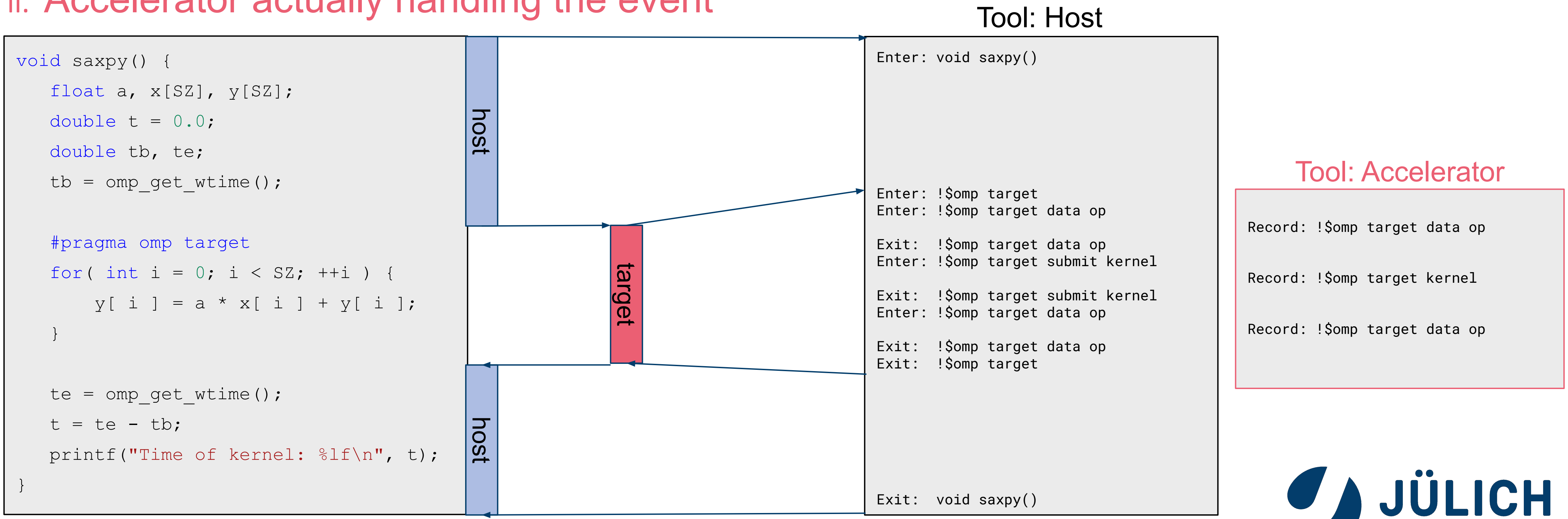
```
typedef struct scorep_ompt_target_data_t {
    const void* codeptr_ra;
    ompt_id_t   target_id;
    bool        supports_device_tracing;
} scorep_ompt_target_data;
```

```
/* Use the following 64-bit layout for mapping host_op_id:
 * --------------------------------------------------------------------------
 * 00000000 00000000 00000000 | 00000000 00000000 00000000 00000000 00000000
 * hostLocationId             | hostOpId
 * --------------------------------------------------------------------------
 */
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

# ACCELERATOR EVENTS AND TOOLS

- Unlike host directives, offloading needs to be handled twice to get all information
  i. Host side dispatching events and waiting for completion
  ii. Accelerator actually handling the event



```
void saxpy() {
   float a, x[SZ], y[SZ];
   double t = 0.0;
   double tb, te;
   tb = omp_get_wtime();

   #pragma omp target
   for( int i = 0; i < SZ; ++i ) {
       y[ i ] = a * x[ i ] + y[ i ];
   }

   te = omp_get_wtime();
   t = te - tb;
   printf("Time of kernel: %lf\n", t);
}
```

Tool: Host

```
Enter: void saxpy()







Enter: !$omp target
Enter: !$omp target data op

Exit:  !$omp target data op
Enter: !$omp target submit kernel

Exit:  !$omp target submit kernel
Enter: !$omp target data op

Exit:  !$omp target data op
Exit:  !$omp target







Exit:  void saxpy()
```

Tool: Accelerator

```
Record: !$omp target data op


Record: !$omp target kernel


Record: !$omp target data op
```

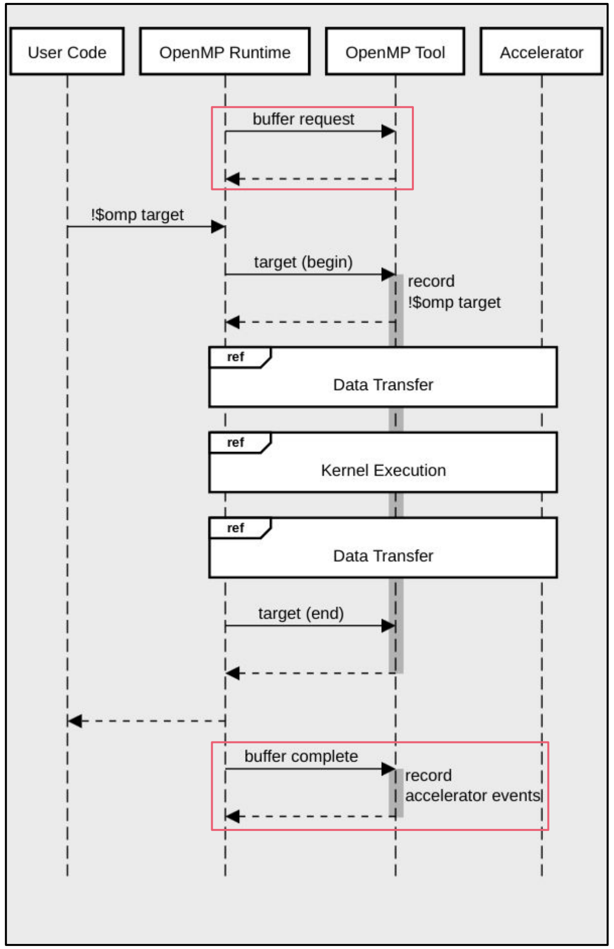
# THE DEVICE TRACING INTERFACE

## What is it?

- Buffer-based handling of accelerator events (similar to CUPTI, rocTracer)
- When a device is initialized, we can enable this interface
- Runtime will ask for buffers, record events, flush full buffers

What tools need to do:

- Sort buffers, as runtimes are not required to sort buffer
- Convert timestamps, either manually or via `ompt_translate_time`
- Iterate through buffer and write events



# THE DEVICE TRACING INTERFACE

## What does a record contain?

- Each record contains:
  - What type of event is recorded
  - When the record was recorded
  - Which thread recorded the record
  - The mapped `target_data` of the tool
  - The actual record of the callback

- Actual records may contain more information, like end timestamp and our set `host_op_id`

```
typedef struct ompt_record_ompt_t {
   ompt_callbacks_t    type;
   ompt_device_time_t time;
   ompt_id_t           thread_id;
   ompt_id_t           target_id;
   union {
      [...]
      ompt_record_target_t         target;
      ompt_record_target_data_op_t target_data_op;
      ompt_record_target_kernel_t  target_kernel;
   } record;
} ompt_record_ompt_t;
```

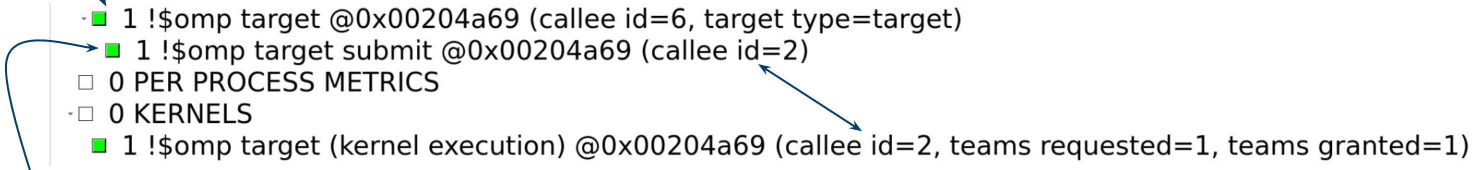
```
typedef struct ompt_record_target_kernel_t {
    ompt_id_t host_op_id;
    unsigned int requested_num_teams;
    unsigned int granted_num_teams;
    ompt_device_time_t end_time;
} ompt_record_target_kernel_t;
```

# BRINGING BOTH TOGETHER

## What can we record with callbacks and the device tracing interface?

```
typedef void (*ompt_callback_target_emi_t) (
    ompt_target_t         kind,
    ompt_scope_endpoint_t endpoint,
    int                   device_num,
    ompt_data_t*          task_data,
    ompt_data_t*          target_task_data,
    ompt_data_t*          target_data,
    const void*           codeptr_ra );
```

```
typedef struct ompt_record_ompt_t {
   ompt_callbacks_t   type;
   ompt_device_time_t time;
   ompt_id_t          thread_id;
   ompt_id_t          target_id;
   union {
       [...]
       ompt_record_target_t         target;
       ompt_record_target_data_op_t target_data_op;
       ompt_record_target_kernel_t  target_kernel;
   } record;
} ompt_record_ompt_t;
```



```
🟩 1 !$omp target @0x00204a69 (callee id=6, target type=target)
  🟩 1 !$omp target submit @0x00204a69 (callee id=2)
□ 0 PER PROCESS METRICS
□ 0 KERNELS
🟩 1 !$omp target (kernel execution) @0x00204a69 (callee id=2, teams requested=1, teams granted=1)
```

```
typedef void (*ompt_callback_submit_emi) (
    ompt_scope_endpoint_t endpoint,
    ompt_data_t*          target_data,
    ompt_id_t*            host_op_id,
    unsigned int          requested_num_teams );
```

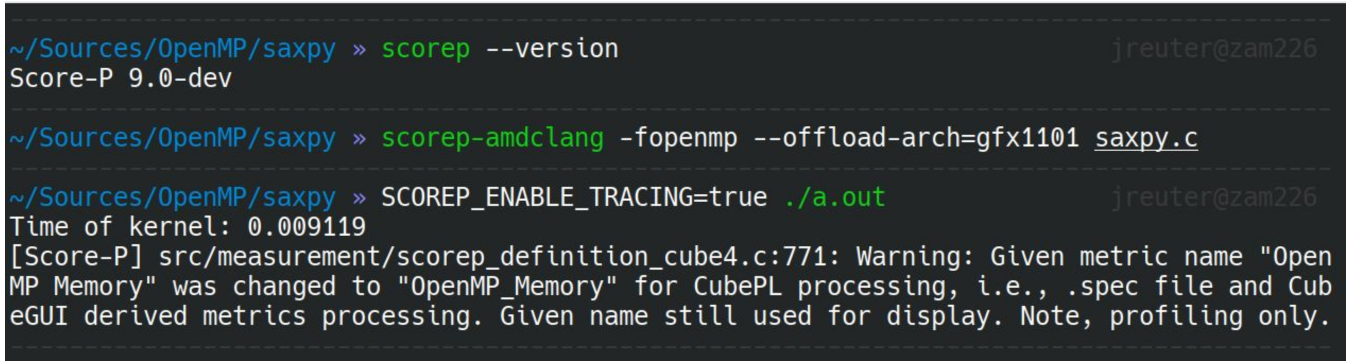
```
typedef struct ompt_record_target_kernel_t {
    ompt_id_t host_op_id;
    unsigned int requested_num_teams;
    unsigned int granted_num_teams;
    ompt_device_time_t end_time;
} ompt_record_target_kernel_t;
```

# HOW DO I USE THE NEW FEATURES?

# HOW TO USE THE NEW FEATURES?

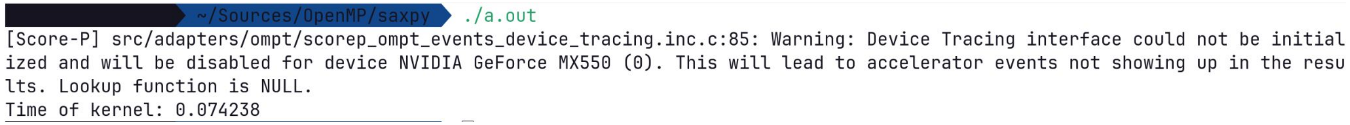
## There are no additional steps needed!

- The OpenMP Tools Interface will be the default with Score-P v9.0
- Example:

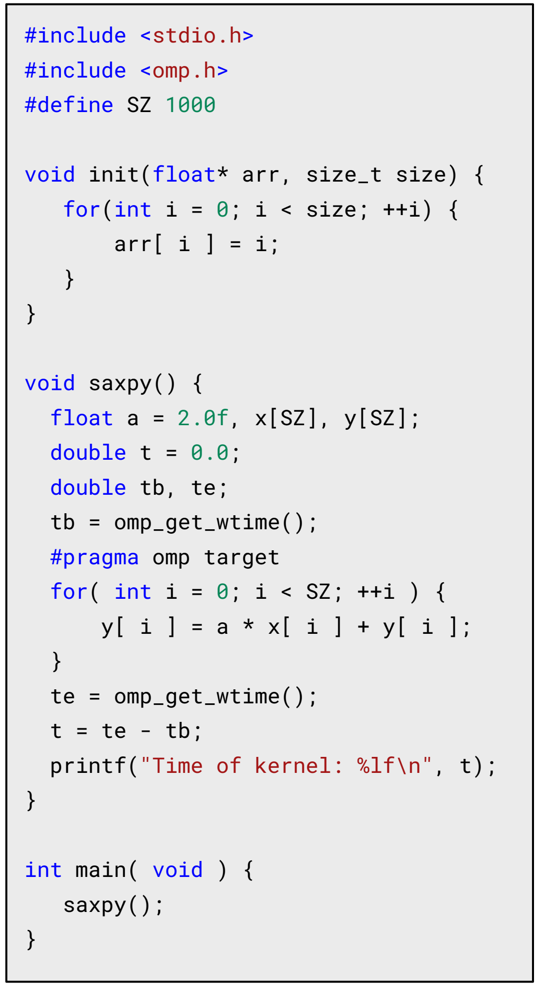


```
~/Sources/OpenMP/saxpy » scorep --version                                     jreuter@zam226
Score-P 9.0-dev

~/Sources/OpenMP/saxpy » scorep-amdclang -fopenmp --offload-arch=gfx1101 saxpy.c

~/Sources/OpenMP/saxpy » SCOREP_ENABLE_TRACING=true ./a.out                   jreuter@zam226
Time of kernel: 0.009119
[Score-P] src/measurement/scorep_definition_cube4.c:771: Warning: Given metric name "Open
MP Memory" was changed to "OpenMP_Memory" for CubePL processing, i.e., .spec file and Cub
eGUI derived metrics processing. Given name still used for display. Note, profiling only.
```

- If device tracing cannot be activated, a message is shown

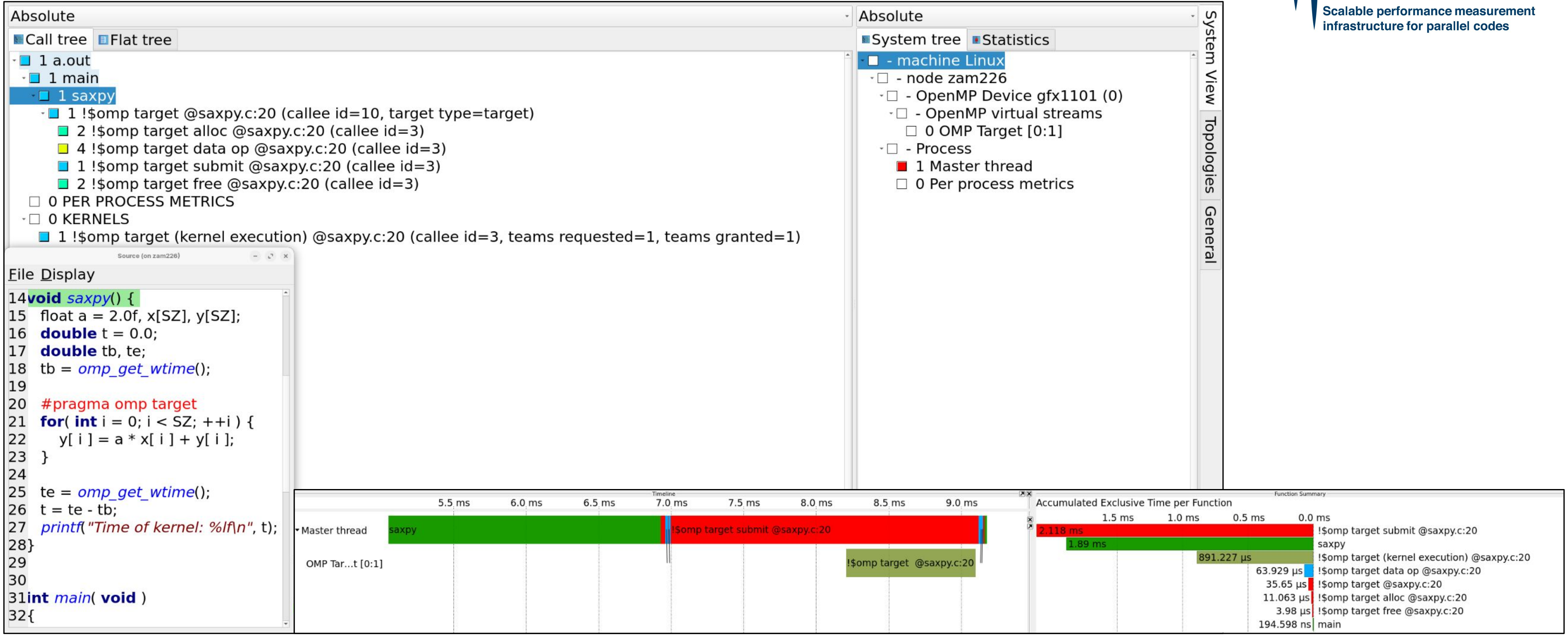


```
[redacted] ~/Sources/OpenMP/saxpy ./a.out
[Score-P] src/adapters/ompt/scorep_ompt_events_device_tracing.inc.c:85: Warning: Device Tracing interface could not be initial
ized and will be disabled for device NVIDIA GeForce MX550 (0). This will lead to accelerator events not showing up in the resu
lts. Lookup function is NULL.
Time of kernel: 0.074238
```



```c
#include <stdio.h>
#include <omp.h>
#define SZ 1000

void init(float* arr, size_t size) {
   for(int i = 0; i < size; ++i) {
       arr[ i ] = i;
   }
}

void saxpy() {
  float a = 2.0f, x[SZ], y[SZ];
  double t = 0.0;
  double tb, te;
  tb = omp_get_wtime();
  #pragma omp target
  for( int i = 0; i < SZ; ++i ) {
      y[ i ] = a * x[ i ] + y[ i ];
  }
  te = omp_get_wtime();
  t = te - tb;
  printf("Time of kernel: %lf\n", t);
}

int main( void ) {
   saxpy();
}
```

# RESULTS: LETS START SMALL …



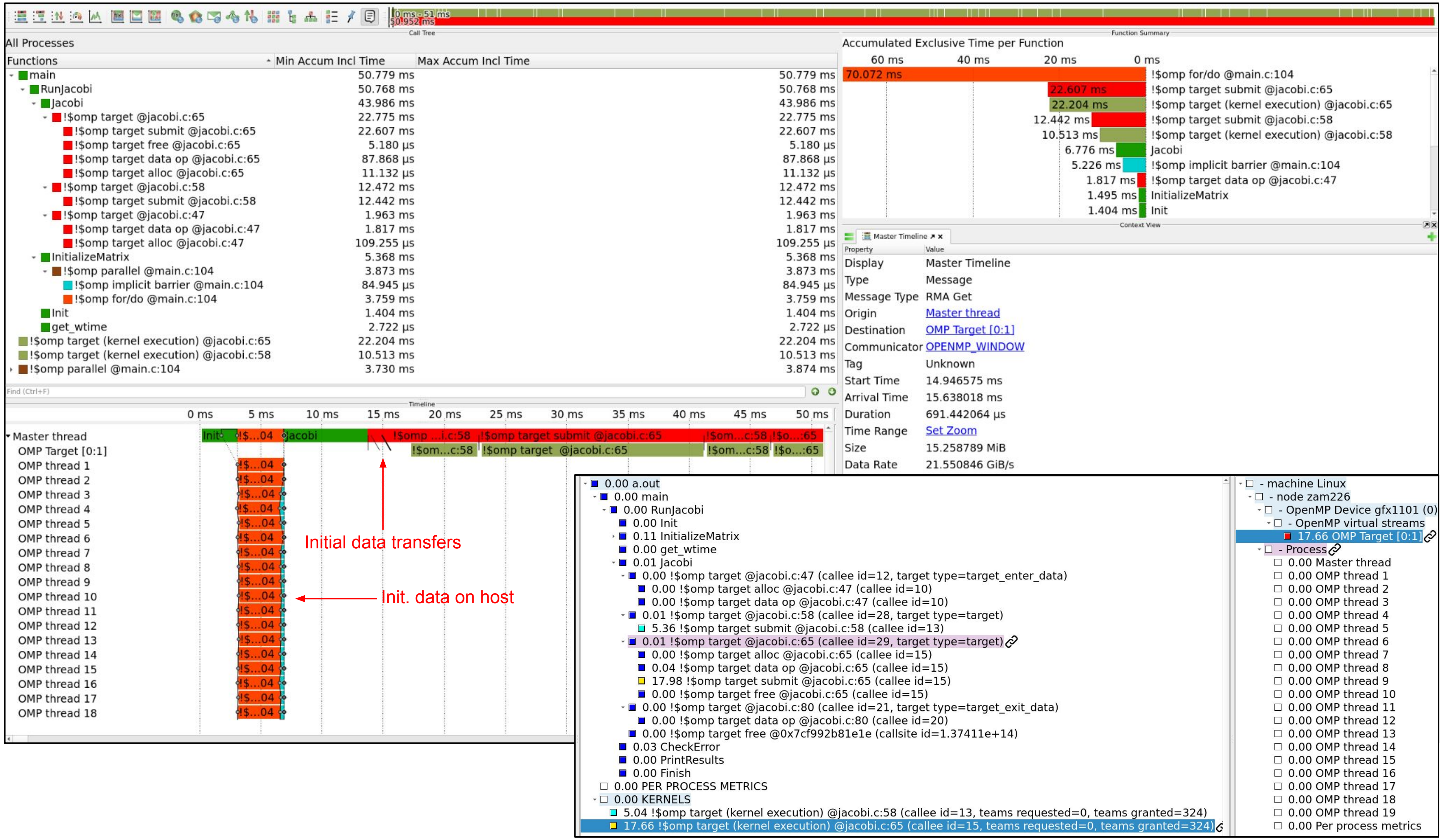
```
1 a.out
  1 main
    1 saxpy
      1 !$omp target @saxpy.c:20 (callee id=10, target type=target)
        2 !$omp target alloc @saxpy.c:20 (callee id=3)
        4 !$omp target data op @saxpy.c:20 (callee id=3)
        1 !$omp target submit @saxpy.c:20 (callee id=3)
        2 !$omp target free @saxpy.c:20 (callee id=3)
  0 PER PROCESS METRICS
  0 KERNELS
    1 !$omp target (kernel execution) @saxpy.c:20 (callee id=3, teams requested=1, teams granted=1)
```

```
- machine Linux
  - node zam226
    - OpenMP Device gfx1101 (0)
      - OpenMP virtual streams
        0 OMP Target [0:1]
    - Process
      1 Master thread
      0 Per process metrics
```

```c
14void saxpy() {
15  float a = 2.0f, x[SZ], y[SZ];
16  double t = 0.0;
17  double tb, te;
18  tb = omp_get_wtime();
19
20  #pragma omp target
21  for( int i = 0; i < SZ; ++i ) {
22     y[ i ] = a * x[ i ] + y[ i ];
23  }
24
25  te = omp_get_wtime();
26  t = te - tb;
27  printf("Time of kernel: %lf\n", t);
28}
29
30
31int main( void )
32{
```

```
Accumulated Exclusive Time per Function
     1.5 ms     1.0 ms     0.5 ms     0.0 ms
2.118 ms                                  !$omp target submit @saxpy.c:20
   1.89 ms                                saxpy
                    891.227 µs            !$omp target (kernel execution) @saxpy.c:20
                               63.929 µs  !$omp target data op @saxpy.c:20
                                35.65 µs  !$omp target @saxpy.c:20
                               11.063 µs  !$omp target alloc @saxpy.c:20
                                 3.98 µs  !$omp target free @saxpy.c:20
                              194.598 ns  main
```

# … INCREASE THE COMPLEXITY …



```
All Processes
Functions                                    Min Accum Incl Time   Max Accum Incl Time
- main                                               50.779 ms      50.779 ms
  - RunJacobi                                        50.768 ms      50.768 ms
    - Jacobi                                         43.986 ms      43.986 ms
      - !$omp target @jacobi.c:65                    22.775 ms      22.775 ms
          !$omp target submit @jacobi.c:65           22.607 ms      22.607 ms
          !$omp target free @jacobi.c:65              5.180 µs       5.180 µs
          !$omp target data op @jacobi.c:65          87.868 µs      87.868 µs
          !$omp target alloc @jacobi.c:65            11.132 µs      11.132 µs
      - !$omp target @jacobi.c:58                    12.472 ms      12.472 ms
          !$omp target submit @jacobi.c:58           12.442 ms      12.442 ms
      - !$omp target @jacobi.c:47                     1.963 ms       1.963 ms
          !$omp target data op @jacobi.c:47           1.817 ms       1.817 ms
          !$omp target alloc @jacobi.c:47           109.255 µs     109.255 µs
    - InitializeMatrix                                5.368 ms       5.368 ms
      - !$omp parallel @main.c:104                    3.873 ms       3.873 ms
          !$omp implicit barrier @main.c:104         84.945 µs      84.945 µs
          !$omp for/do @main.c:104                    3.759 ms       3.759 ms
      Init                                            1.404 ms       1.404 ms
      get_wtime                                       2.722 µs       2.722 µs
  !$omp target (kernel execution) @jacobi.c:65       22.204 ms      22.204 ms
  !$omp target (kernel execution) @jacobi.c:58       10.513 ms      10.513 ms
> !$omp parallel @main.c:104                          3.730 ms       3.874 ms
```

```
Accumulated Exclusive Time per Function
60 ms   40 ms   20 ms   0 ms
70.072 ms   !$omp for/do @main.c:104
22.607 ms   !$omp target submit @jacobi.c:65
22.204 ms   !$omp target (kernel execution) @jacobi.c:65
12.442 ms   !$omp target submit @jacobi.c:58
10.513 ms   !$omp target (kernel execution) @jacobi.c:58
 6.776 ms   Jacobi
 5.226 ms   !$omp implicit barrier @main.c:104
 1.817 ms   !$omp target data op @jacobi.c:47
 1.495 ms   InitializeMatrix
 1.404 ms   Init
```

```
Property          Value
Display           Master Timeline
Type              Message
Message Type      RMA Get
Origin            Master thread
Destination       OMP Target [0:1]
Communicator      OPENMP_WINDOW
Tag               Unknown
Start Time        14.946575 ms
Arrival Time      15.638018 ms
Duration          691.442064 µs
Time Range        Set Zoom
Size              15.258789 MiB
Data Rate         21.550846 GiB/s
```

```
0.00 a.out
  0.00 main
    0.00 RunJacobi
      0.00 Init
      0.11 InitializeMatrix
      0.00 get_wtime
      0.01 Jacobi
        0.00 !$omp target @jacobi.c:47 (callee id=12, target type=target_enter_data)
          0.00 !$omp target alloc @jacobi.c:47 (callee id=10)
          0.00 !$omp target data op @jacobi.c:47 (callee id=10)
        0.01 !$omp target @jacobi.c:58 (callee id=28, target type=target)
          5.36 !$omp target submit @jacobi.c:58 (callee id=13)
        0.01 !$omp target @jacobi.c:65 (callee id=29, target type=target)
          0.00 !$omp target alloc @jacobi.c:65 (callee id=15)
          0.04 !$omp target data op @jacobi.c:65 (callee id=15)
          17.98 !$omp target submit @jacobi.c:65 (callee id=15)
          0.00 !$omp target free @jacobi.c:65 (callee id=15)
        0.00 !$omp target @jacobi.c:80 (callee id=21, target type=target_exit_data)
          0.00 !$omp target data op @jacobi.c:80 (callee id=20)
        0.00 !$omp target free @0x7cf992b81e1e (callsite id=1.37411e+14)
      0.03 CheckError
      0.00 PrintResults
      0.00 Finish
  0.00 PER PROCESS METRICS
  0.00 KERNELS
    5.04 !$omp target (kernel execution) @jacobi.c:58 (callee id=13, teams requested=0, teams granted=324)
    17.66 !$omp target (kernel execution) @jacobi.c:65 (callee id=15, teams requested=0, teams granted=324)
```

```
- machine Linux
  - node zam226
    - OpenMP Device gfx1101 (0)
      - OpenMP virtual streams
          17.66 OMP Target [0:1]
    - Process
        0.00 Master thread
        0.00 OMP thread 1
        0.00 OMP thread 2
        0.00 OMP thread 3
        0.00 OMP thread 4
        0.00 OMP thread 5
        0.00 OMP thread 6
        0.00 OMP thread 7
        0.00 OMP thread 8
        0.00 OMP thread 9
        0.00 OMP thread 10
        0.00 OMP thread 11
        0.00 OMP thread 12
        0.00 OMP thread 13
        0.00 OMP thread 14
        0.00 OMP thread 15
        0.00 OMP thread 16
        0.00 OMP thread 17
        0.00 OMP thread 18
        0.00 OMP thread 19
        0.00 Per process metrics
```

Initial data transfers

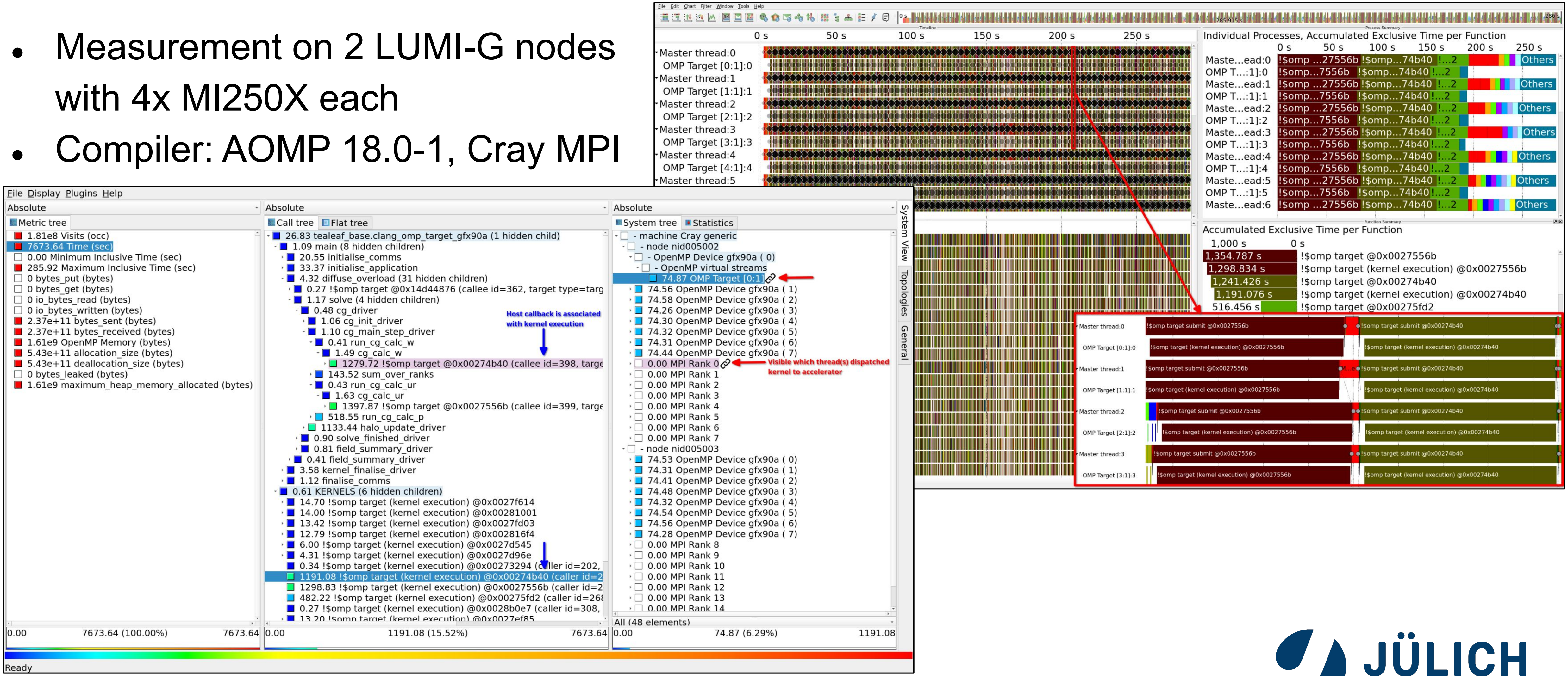
Init. data on host

- Jacobi example used in Score-P testing
- System info:
  - Ubuntu 22.04
  - ROCm 6.1.0
  - RX 7700 XT

# … AND SCALE IT UP!

## Running the SPEC HPC tealeaf benchmark on multiple LUMI-G nodes

- Measurement on 2 LUMI-G nodes with 4x MI250X each
- Compiler: AOMP 18.0-1, Cray MPI



File Edit Chart Filter Window Tools Help
0 s
50 s
100 s
150 s
200 s
250 s
Master thread:0
OMP Target [0:1]:0
Master thread:1
OMP Target [1:1]:1
Master thread:2
OMP Target [2:1]:2
Master thread:3
OMP Target [3:1]:3
Master thread:4
OMP Target [4:1]:4
Master thread:5

Individual Processes, Accumulated Exclusive Time per Function

0 s
50 s
100 s
150 s
200 s
250 s
Maste...ead:0 !$omp ...27556b !$omp...74b40 !...2 Others
OMP T...:1]:0 !$omp...7556b !$omp...74b40 !...2
Maste...ead:1 !$omp ...27556b !$omp...74b40 !...2 Others
OMP T...:1]:1 !$omp...7556b !$omp...74b40 !...2
Maste...ead:2 !$omp ...27556b !$omp...74b40 !...2 Others
OMP T...:1]:2 !$omp...7556b !$omp...74b40 !...2
Maste...ead:3 !$omp ...27556b !$omp...74b40 !...2 Others
OMP T...:1]:3 !$omp...7556b !$omp...74b40 !...2
Maste...ead:4 !$omp ...27556b !$omp...74b40 !...2 Others
OMP T...:1]:4 !$omp...7556b !$omp...74b40 !...2
Maste...ead:5 !$omp ...27556b !$omp...74b40 !...2 Others
OMP T...:1]:5 !$omp...7556b !$omp...74b40 !...2
Maste...ead:6 !$omp ...27556b !$omp...74b40 !...2 Others

Accumulated Exclusive Time per Function

| 1,000 s   0 s | |
|---|---|
| 1,354.787 s | !$omp target @0x0027556b |
| 1,298.834 s | !$omp target (kernel execution) @0x0027556b |
| 1,241.426 s | !$omp target @0x00274b40 |
| 1,191.076 s | !$omp target (kernel execution) @0x00274b40 |
| 516.456 s | !$omp target @0x00275fd2 |

Master thread:0 !$omp target submit @0x0027556b !$omp target submit @0x00274b40
OMP Target [0:1]:0 !$omp target (kernel execution) @0x0027556b !$omp target (kernel execution) @0x00274b40
Master thread:1 !$omp target submit @0x0027556b M...e !$omp target submit @0x00274b40
OMP Target [1:1]:1 !$omp target (kernel execution) @0x0027556b !$omp target (kernel execution) @0x00274b40
Master thread:2 !$omp target submit @0x0027556b !$omp target submit @0x00274b40
OMP Target [2:1]:2 !$omp target (kernel execution) @0x0027556b !$omp target (kernel execution) @0x00274b40
Master thread:3 !$omp target submit @0x0027556b !$omp target submit @0x00274b40
OMP Target [3:1]:3 !$omp target (kernel execution) @0x0027556b !$omp target (kernel execution) @0x00274b40

File Display Plugins Help
Absolute
Metric tree

```
■ 1.81e8 Visits (occ)
■ 7673.64 Time (sec)
□ 0.00 Minimum Inclusive Time (sec)
■ 285.92 Maximum Inclusive Time (sec)
□ 0 bytes_put (bytes)
□ 0 bytes_get (bytes)
□ 0 io_bytes_read (bytes)
□ 0 io_bytes_written (bytes)
■ 2.37e+11 bytes_sent (bytes)
■ 2.37e+11 bytes_received (bytes)
■ 1.61e9 OpenMP Memory (bytes)
■ 5.43e+11 allocation_size (bytes)
■ 5.43e+11 deallocation_size (bytes)
□ 0 bytes_leaked (bytes)
■ 1.61e9 maximum_heap_memory_allocated (bytes)
```

0.00 7673.64 (100.00%) 7673.64

```
- ■ 26.83 tealeaf_base.clang_omp_target_gfx90a (1 hidden child)
  - ■ 1.09 main (8 hidden children)
    ■ 20.55 initialise_comms
    ■ 33.37 initialise_application
    - ■ 4.32 diffuse_overload (31 hidden children)
      ■ 0.27 !$omp target @0x14d44876 (callee id=362, target type=targ
      - ■ 1.17 solve (4 hidden children)
        - ■ 0.48 cg_driver
          ■ 1.06 cg_init_driver
          - ■ 1.10 cg_main_step_driver
            - ■ 0.41 run_cg_calc_w
              - ■ 1.49 cg_calc_w
                ■ 1279.72 !$omp target @0x00274b40 (callee id=398, targe
              ■ 143.52 sum_over_ranks
            - ■ 0.43 run_cg_calc_ur
              - ■ 1.63 cg_calc_ur
                ■ 1397.87 !$omp target @0x0027556b (callee id=399, targe
              ■ 518.55 run_cg_calc_p
          ■ 1133.44 halo_update_driver
        ■ 0.90 solve_finished_driver
        ■ 0.81 field_summary_driver
      ■ 0.41 field_summary_driver
    ■ 3.58 kernel_finalise_driver
    ■ 1.12 finalise_comms
  - ■ 0.61 KERNELS (6 hidden children)
    ■ 14.70 !$omp target (kernel execution) @0x0027f614
    ■ 14.00 !$omp target (kernel execution) @0x00281001
    ■ 13.42 !$omp target (kernel execution) @0x0027fd03
    ■ 12.79 !$omp target (kernel execution) @0x002816f4
    ■ 6.00 !$omp target (kernel execution) @0x0027d545
    ■ 4.31 !$omp target (kernel execution) @0x0027d96e
    ■ 0.34 !$omp target (kernel execution) @0x00273294 (caller id=202,
    ■ 1191.08 !$omp target (kernel execution) @0x00274b40 (caller id=2
    ■ 1298.83 !$omp target (kernel execution) @0x0027556b (caller id=2
    ■ 482.22 !$omp target (kernel execution) @0x00275fd2 (caller id=26
    ■ 0.27 !$omp target (kernel execution) @0x0028b0e7 (caller id=308,
    ■ 13.20 !$omp target (kernel execution) @0x0027ef85
```

Host callback is associated with kernel execution

0.00 1191.08 (15.52%) 7673.64

```
- □ - machine Cray generic
  - □ - node nid005002
    - □ - OpenMP Device gfx90a ( 0)
      - □ - OpenMP virtual streams
          ■ 74.87 OMP Target [0:1]
      ■ 74.56 OpenMP Device gfx90a ( 1)
      ■ 74.58 OpenMP Device gfx90a ( 2)
      ■ 74.26 OpenMP Device gfx90a ( 3)
      ■ 74.30 OpenMP Device gfx90a ( 4)
      ■ 74.32 OpenMP Device gfx90a ( 5)
      ■ 74.31 OpenMP Device gfx90a ( 6)
      ■ 74.44 OpenMP Device gfx90a ( 7)
      □ 0.00 MPI Rank 0
      □ 0.00 MPI Rank 1
      □ 0.00 MPI Rank 2
      □ 0.00 MPI Rank 3
      □ 0.00 MPI Rank 4
      □ 0.00 MPI Rank 5
      □ 0.00 MPI Rank 6
      □ 0.00 MPI Rank 7
  - □ - node nid005003
      ■ 74.53 OpenMP Device gfx90a ( 0)
      ■ 74.31 OpenMP Device gfx90a ( 1)
      ■ 74.41 OpenMP Device gfx90a ( 2)
      ■ 74.48 OpenMP Device gfx90a ( 3)
      ■ 74.32 OpenMP Device gfx90a ( 4)
      ■ 74.54 OpenMP Device gfx90a ( 5)
      ■ 74.56 OpenMP Device gfx90a ( 6)
      ■ 74.28 OpenMP Device gfx90a ( 7)
      □ 0.00 MPI Rank 8
      □ 0.00 MPI Rank 9
      □ 0.00 MPI Rank 10
      □ 0.00 MPI Rank 11
      □ 0.00 MPI Rank 12
      □ 0.00 MPI Rank 13
      □ 0.00 MPI Rank 14
```

Visible which thread(s) dispatched kernel to accelerator

All (48 elements)

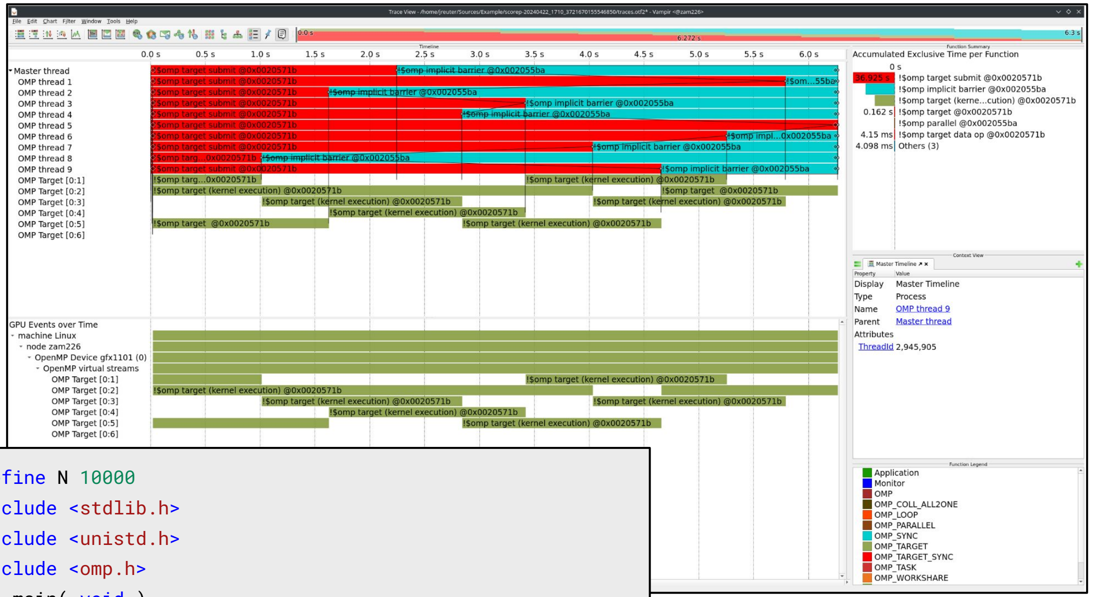
0.00 74.87 (6.29%) 1191.08

Ready

# LIMITATIONS

## The OpenMP Tools Interface only offers so much… (as of spec. 5.2)

- The device tracing interface doesn’t give us any information about the low-level stream for any event
    - To handle overlapping events, we create `OpenMP virtual streams`
    - May lead to more `OpenMP virtual streams` than actually created by runtime



Trace View - /home/jreuter/Sources/Example/scorep-20240422_1710_3721670155546850/traces.otf2* - Vampir <@zam226>

Accumulated Exclusive Time per Function

| | |
|---|---|
| 0 s | |
| 36.925 s | !$omp target submit @0x0020571b |
| | !$omp implicit barrier @0x002055ba |
| | !$omp target (kerne...cution) @0x0020571b |
| 0.162 s | !$omp target @0x0020571b |
| | !$omp parallel @0x002055ba |
| 4.15 ms | !$omp target data op @0x0020571b |
| 4.098 ms | Others (3) |

| Property | Value |
|---|---|
| Display | Master Timeline |
| Type | Process |
| Name | OMP thread 9 |
| Parent | Master thread |
| Attributes | |
| ThreadId | 2,945,905 |

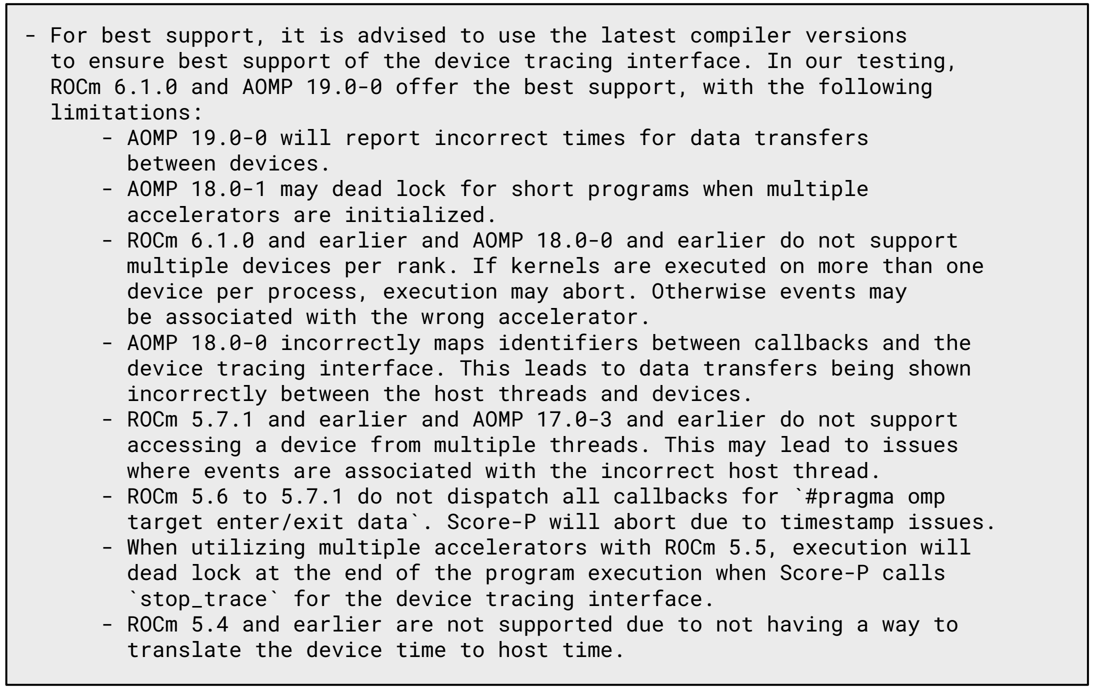
- 🟩 Application
- 🟦 Monitor
- 🟥 OMP
- ■ OMP_COLL_ALL2ONE
- 🟧 OMP_LOOP
- 🟫 OMP_PARALLEL
- ■ OMP_SYNC
- 🟩 OMP_TARGET
- 🟥 OMP_TARGET_SYNC
- 🟥 OMP_TASK
- 🟧 OMP_WORKSHARE

```c
#define N 10000
#include <stdlib.h>
#include <unistd.h>
#include <omp.h>
int main( void )
{
    #pragma omp parallel
    {
        size_t* sum = malloc( sizeof( size_t ) * N );
        #pragma omp target teams \
                  distribute parallel for simd \
                  map(tofrom: sum[:N])
        for( size_t i = 0; i < N; ++i )
        {
            sum[i] = i;
        }
        free( sum );
    }
}
```

# LIMITATIONS

## Compiler and runtime limitations

- The device tracing interface doesn't give us any information about the low-level stream for any event
  - To handle overlapping events, we create `OpenMP virtual streams`
  - May lead to more `OpenMP virtual streams` than actually created by runtime
- OpenMP runtimes still have runtime issues
  - We generally recommend the latest releases, as they are the most stable



```
- For best support, it is advised to use the latest compiler versions
  to ensure best support of the device tracing interface. In our testing,
  ROCm 6.1.0 and AOMP 19.0-0 offer the best support, with the following
  limitations:
      - AOMP 19.0-0 will report incorrect times for data transfers
        between devices.
      - AOMP 18.0-1 may dead lock for short programs when multiple
        accelerators are initialized.
      - ROCm 6.1.0 and earlier and AOMP 18.0-0 and earlier do not support
        multiple devices per rank. If kernels are executed on more than one
        device per process, execution may abort. Otherwise events may
        be associated with the wrong accelerator.
      - AOMP 18.0-0 incorrectly maps identifiers between callbacks and the
        device tracing interface. This leads to data transfers being shown
        incorrectly between the host threads and devices.
      - ROCm 5.7.1 and earlier and AOMP 17.0-3 and earlier do not support
        accessing a device from multiple threads. This may lead to issues
        where events are associated with the incorrect host thread.
      - ROCm 5.6 to 5.7.1 do not dispatch all callbacks for `#pragma omp
        target enter/exit data`. Score-P will abort due to timestamp issues.
      - When utilizing multiple accelerators with ROCm 5.5, execution will
        dead lock at the end of the program execution when Score-P calls
        `stop_trace` for the device tracing interface.
      - ROCm 5.4 and earlier are not supported due to not having a way to
        translate the device time to host time.
```

Snippet from our OPEN_ISSUES for AMD (others to follow)

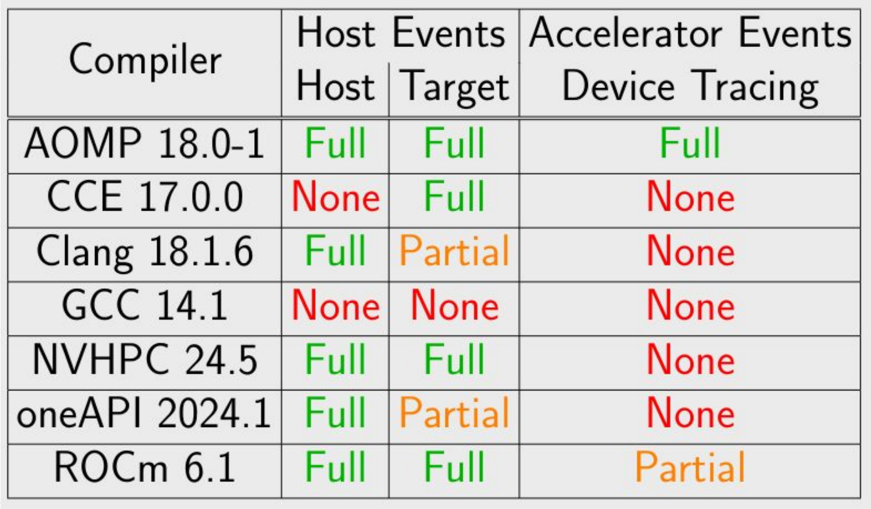
# WHAT ABOUT OTHER RUNTIMES?

## Not all runtimes do support device tracing

- Score-P will output a warning, reminding that no accelerator data will be collected by OMPT
- However, host callbacks are still recorded!

**Solution:** Use native GPU adapters, if possible!

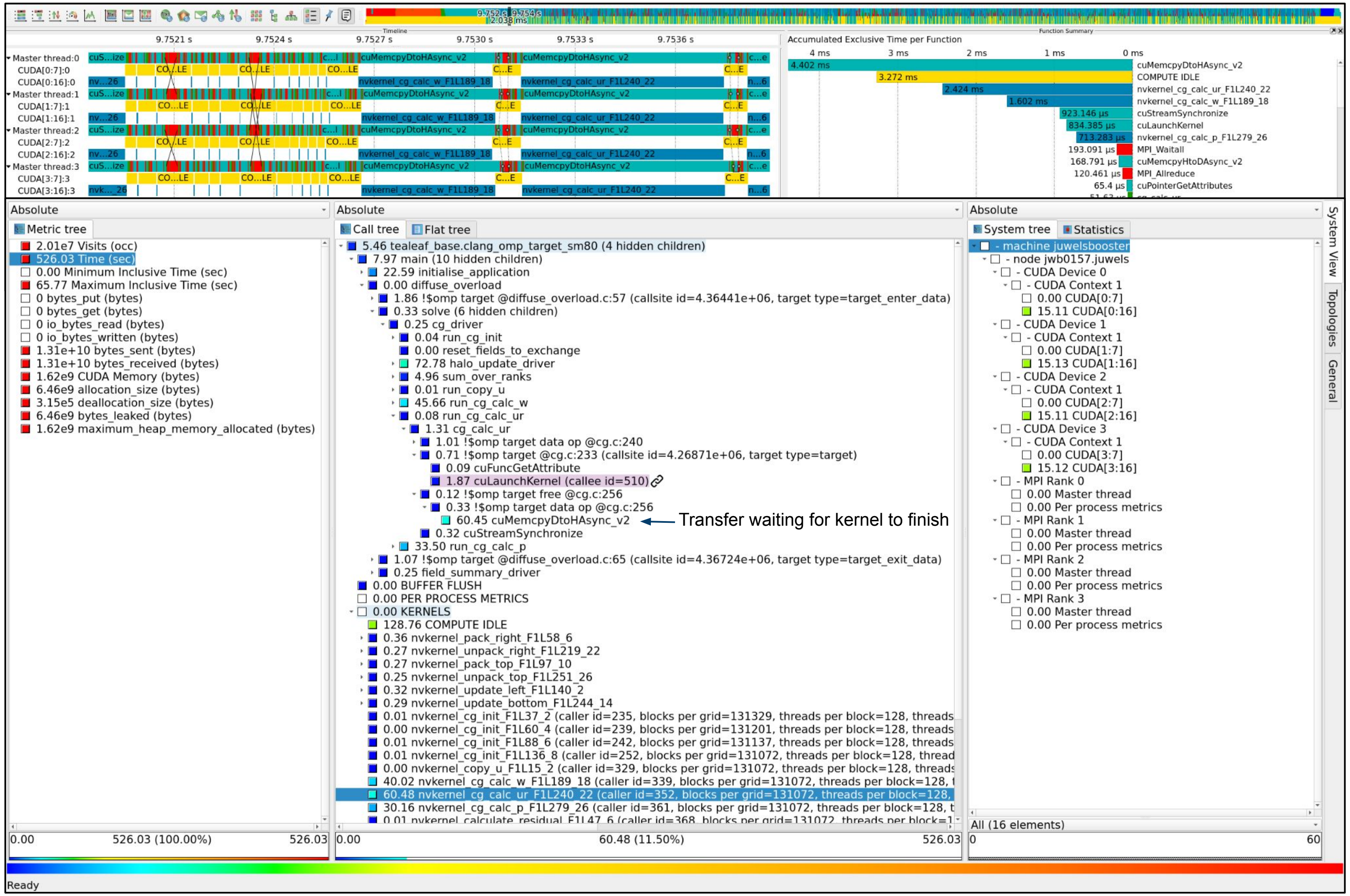
```
$ scorep --thread=omp:ompt –cuda nvc -mp=gpu \
  my-code.c -o my-code.out
$ SCOREP_ENABLE_TRACING=true ./my-code.out
```



| Compiler | Host Events | | Accelerator Events |
|---|---|---|---|
| | Host | Target | Device Tracing |
| AOMP 18.0-1 | Full | Full | Full |
| CCE 17.0.0 | None | Full | None |
| Clang 18.1.6 | Full | Partial | None |
| GCC 14.1 | None | None | None |
| NVHPC 24.5 | Full | Full | None |
| oneAPI 2024.1 | Full | Partial | None |
| ROCm 6.1 | Full | Full | Partial |

# COMBINING OMPT AND CUPTI

## Showing off results on JUWELS Booster with NVHPC 23.7 on one node with 4x A100



Absolute

Metric tree

```
2.01e7 Visits (occ)
526.03 Time (sec)
0.00 Minimum Inclusive Time (sec)
65.77 Maximum Inclusive Time (sec)
0 bytes_put (bytes)
0 bytes_get (bytes)
0 io_bytes_read (bytes)
0 io_bytes_written (bytes)
1.31e+10 bytes_sent (bytes)
1.31e+10 bytes_received (bytes)
1.62e9 CUDA Memory (bytes)
6.46e9 allocation_size (bytes)
3.15e5 deallocation_size (bytes)
6.46e9 bytes_leaked (bytes)
1.62e9 maximum_heap_memory_allocated (bytes)
```

0.00 526.03 (100.00%) 526.03

Absolute

Call tree | Flat tree

```
5.46 tealeaf_base.clang_omp_target_sm80 (4 hidden children)
  7.97 main (10 hidden children)
    22.59 initialise_application
    0.00 diffuse_overload
      1.86 !$omp target @diffuse_overload.c:57 (callsite id=4.36441e+06, target type=target_enter_data)
      0.33 solve (6 hidden children)
        0.25 cg_driver
          0.04 run_cg_init
          0.00 reset_fields_to_exchange
          72.78 halo_update_driver
          4.96 sum_over_ranks
          0.01 run_copy_u
          45.66 run_cg_calc_w
          0.08 run_cg_calc_ur
            1.31 cg_calc_ur
              1.01 !$omp target data op @cg.c:240
              0.71 !$omp target @cg.c:233 (callsite id=4.26871e+06, target type=target)
                0.09 cuFuncGetAttribute
                1.87 cuLaunchKernel (callee id=510)
              0.12 !$omp target free @cg.c:256
                0.33 !$omp target data op @cg.c:258
                  60.45 cuMemcpyDtoHAsync_v2
              0.32 cuStreamSynchronize
          33.50 run_cg_calc_p
      1.07 !$omp target @diffuse_overload.c:65 (callsite id=4.36724e+06, target type=target_exit_data)
      0.25 field_summary_driver
  0.00 BUFFER FLUSH
  0.00 PER PROCESS METRICS
  0.00 KERNELS
    128.76 COMPUTE IDLE
    0.36 nvkernel_pack_right_F1L58_6
    0.27 nvkernel_unpack_right_F1L219_22
    0.27 nvkernel_pack_top_F1L97_10
    0.25 nvkernel_unpack_top_F1L251_26
    0.32 nvkernel_update_left_F1L140_2
    0.29 nvkernel_update_bottom_F1L244_14
    0.01 nvkernel_cg_init_F1L37_2 (caller id=235, blocks per grid=131329, threads per block=128, threads
    0.00 nvkernel_cg_init_F1L60_4 (caller id=239, blocks per grid=131201, threads per block=128, threads
    0.01 nvkernel_cg_init_F1L88_6 (caller id=242, blocks per grid=131137, threads per block=128, threads
    0.01 nvkernel_cg_init_F1L136_8 (caller id=252, blocks per grid=131072, threads per block=128, thread
    0.00 nvkernel_copy_u_F1L15_2 (caller id=329, blocks per grid=131072, threads per block=128, threads
    40.02 nvkernel_cg_calc_w_F1L189_18 (caller id=339, blocks per grid=131072, threads per block=128, t
    60.48 nvkernel_cg_calc_ur_F1L240_22 (caller id=352, blocks per grid=131072, threads per block=128,
    30.16 nvkernel_cg_calc_p_F1L279_26 (caller id=361, blocks per grid=131072, threads per block=128, t
    0.01 nvkernel_calculate_residual_F1L47_6 (caller id=368, blocks per grid=131072, threads per block=1
```

← Transfer waiting for kernel to finish

0.00 60.48 (11.50%) 526.03

Absolute

System tree | Statistics

```
- machine juwelsbooster
  - node jwb0157.juwels
    - CUDA Device 0
      - CUDA Context 1
        0.00 CUDA[0:7]
        15.11 CUDA[0:16]
    - CUDA Device 1
      - CUDA Context 1
        0.00 CUDA[1:7]
        15.13 CUDA[1:16]
    - CUDA Device 2
      - CUDA Context 1
        0.00 CUDA[2:7]
        15.11 CUDA[2:16]
    - CUDA Device 3
      - CUDA Context 1
        0.00 CUDA[3:7]
        15.12 CUDA[3:16]
    - MPI Rank 0
      0.00 Master thread
      0.00 Per process metrics
    - MPI Rank 1
      0.00 Master thread
      0.00 Per process metrics
    - MPI Rank 2
      0.00 Master thread
      0.00 Per process metrics
    - MPI Rank 3
      0.00 Master thread
      0.00 Per process metrics
```

All (16 elements)

0 60

- OMPT adapter is still able to record host events
- Some host events might show longer times than expected
  - Synchronization points of low-level runtime
- Native accelerator adapter records kernels and data transfers

# FINAL WORDS

## A short overview of what was shown in this talk

- With Score-P v9.0, we will expand our OpenMP support in several ways
- Most important: Users will be able to record OpenMP target events
  - AMD compilers sufficiently support the OpenMP Tools Interface
  - For other compilers: Native accelerator adapters required to get events
- Some compromises had to be made, partially because of the 5.2 specification
- Available implementation already works on several different systems and on small and large scale

# OBTAIN SCORE-P AND GET IN CONTACT

- Visit our web page:

  https://score-p.org

- Check out our public GitLab mirror:

  https://gitlab.com/score-p/scorep

- Available on several different platforms:

openHPC

EASYBUILD.io

building software with ease

Ubuntu

`ppa:score-p/releases`

https://go.fzj.de/scorep-ompt-device-tracing

Get the Score-P development version

# THANKS FOR YOUR ATTENTION!
# QUESTIONS?