Debugging and optimizing Idefix

the art of hunting segfaults and NaNs

Geoff Lesur

Before we start

Bigfoot environment

- To connect to bigfoot and choose your team, follow the tutorial in « env/Readme.md » of the idefix-days repo
- All of the steps described here are available in `tutorials/debugging/readme.md` of the idefix-days repo. You can directly copy-paste the commands from there.

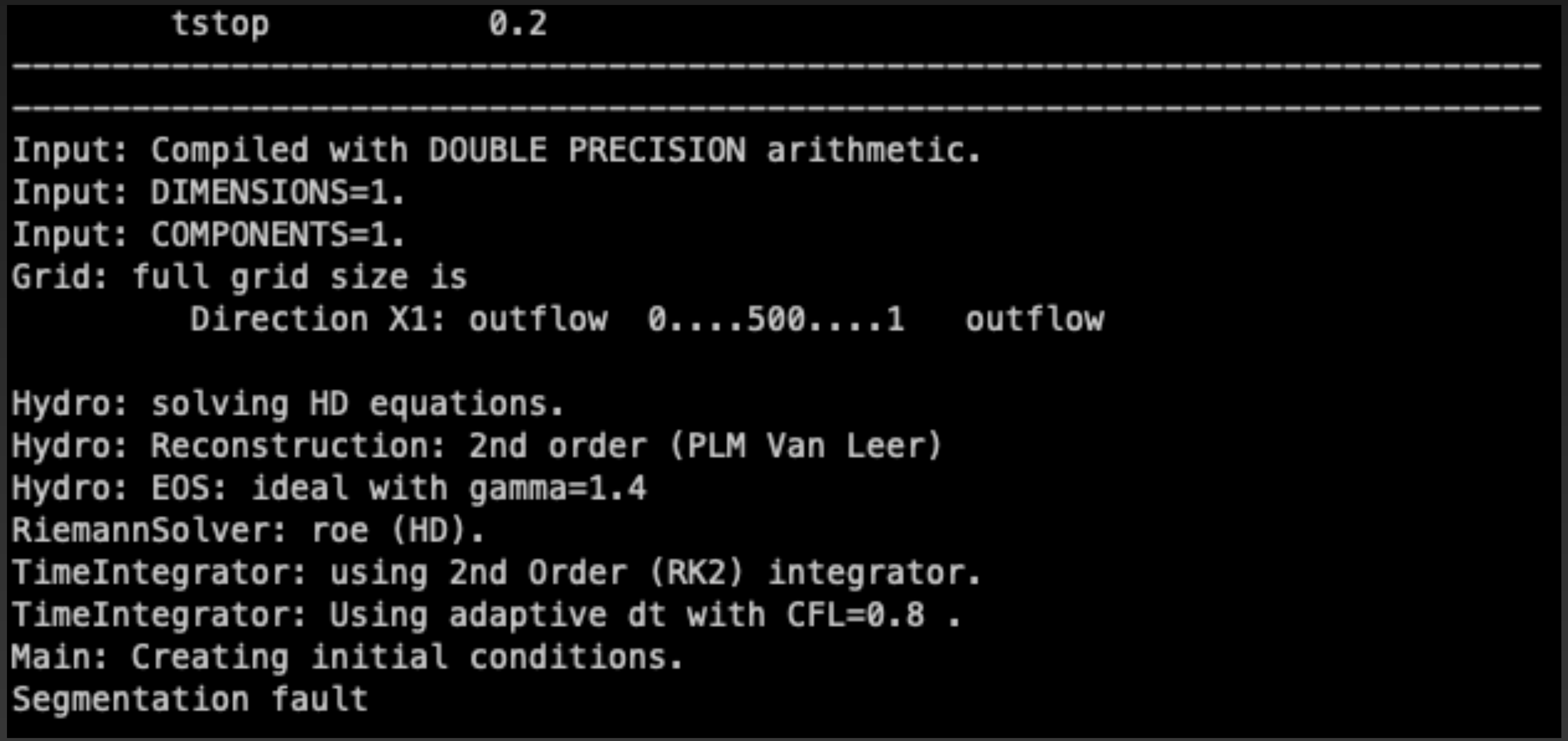
What you will learn

- Use Idefix_DEBUG to trace where you are in the code
- Use Kokkos debugging features (Kokkos Bound checks, Kernel logger),
- Use debuggers
- The most common mistakes™
- Asynchronous error reporting on GPUs
- Use kokkos profiling tools to look at performances

Case study I

A CPU-domain segmentation fault

- go to tutorials/debugging/problem1 (it is the sod tube problem)
- Compile and run
- Surprise!

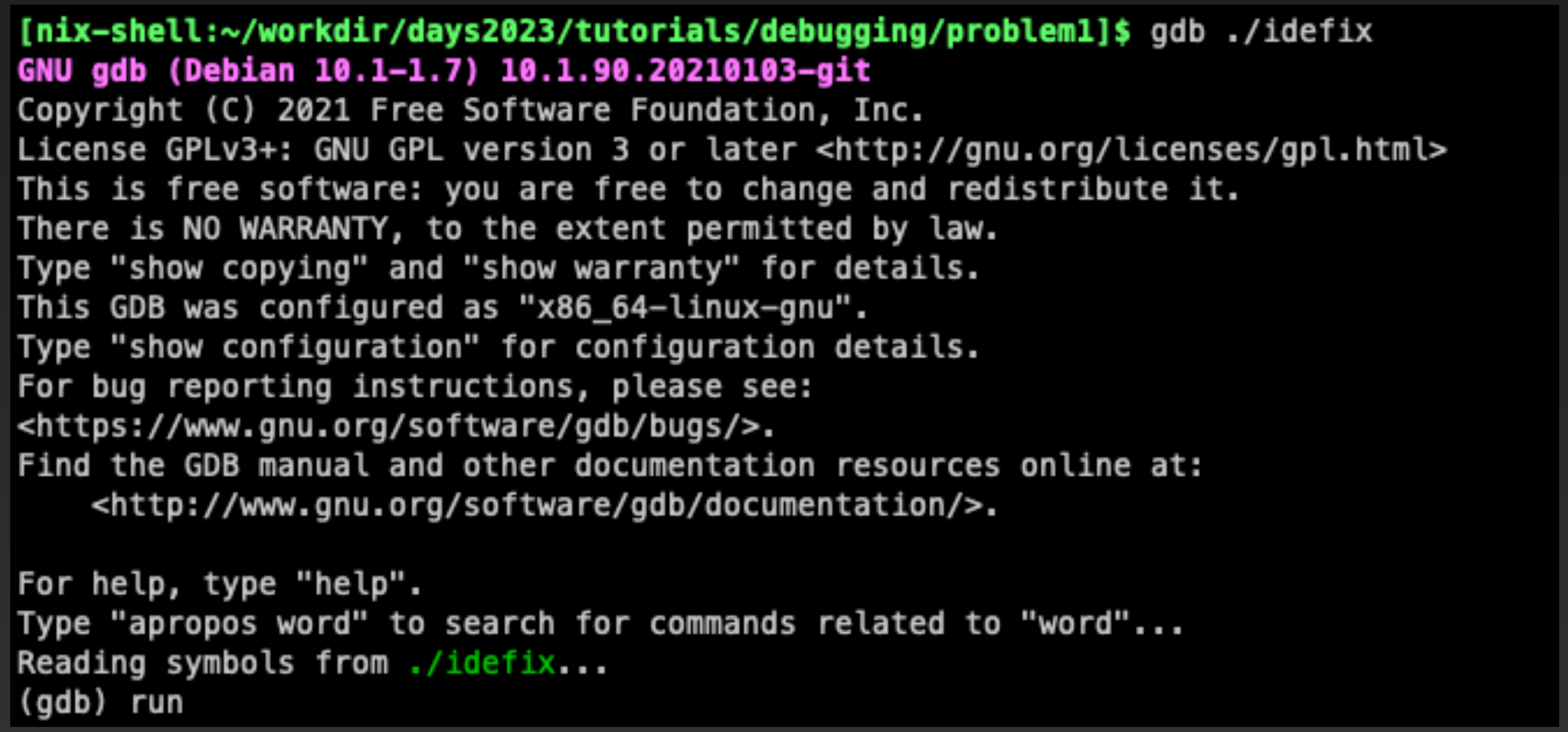


```
       tstop           0.2
-----------------------------------------------------------------------------
-----------------------------------------------------------------------------
Input: Compiled with DOUBLE PRECISION arithmetic.
Input: DIMENSIONS=1.
Input: COMPONENTS=1.
Grid: full grid size is
         Direction X1: outflow  0....500....1   outflow

Hydro: solving HD equations.
Hydro: Reconstruction: 2nd order (PLM Van Leer)
Hydro: EOS: ideal with gamma=1.4
RiemannSolver: roe (HD).
TimeIntegrator: using 2nd Order (RK2) integrator.
TimeIntegrator: Using adaptive dt with CFL=0.8 .
Main: Creating initial conditions.
Segmentation fault
```

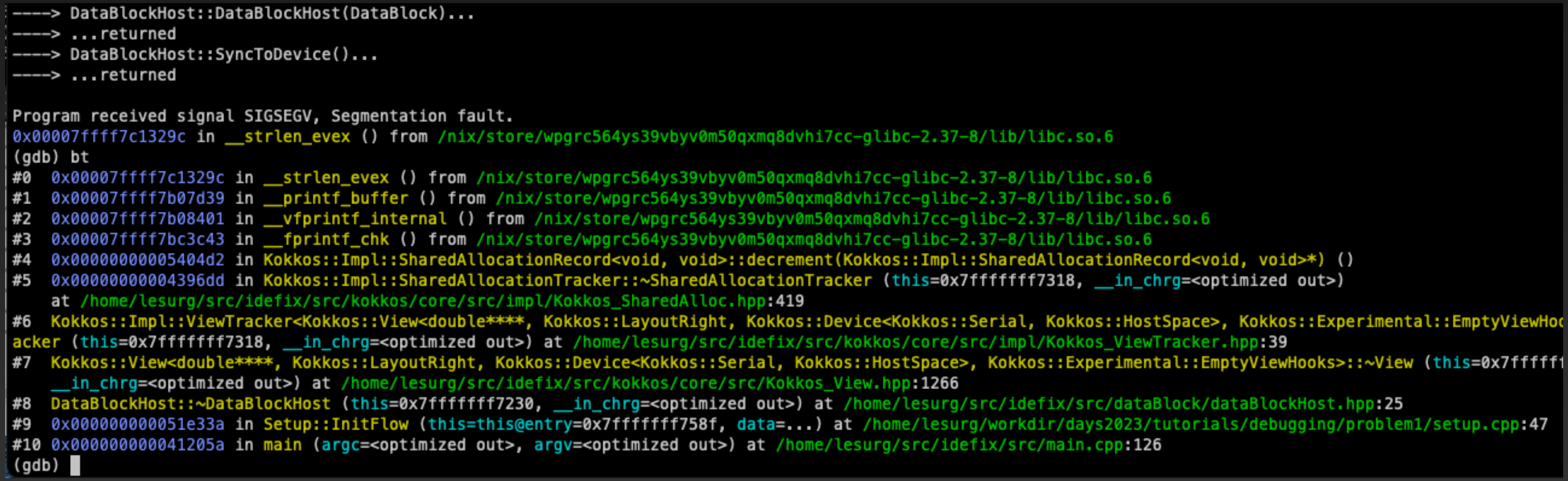
Case study I

A CPU-domain segmentation fault

- Add `Idefix_DEBUG=ON` to your configuration (using `cmake` or `ccmake`) & recompile & run
  - The error happens in `Setup::InitFlow` after SyncToDevice() (cool!)
- Use gdb to debug the code and backtrace the problem



```
[nix-shell:~/workdir/days2023/tutorials/debugging/problem1]$ gdb ./idefix
GNU gdb (Debian 10.1-1.7) 10.1.90.20210103-git
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./idefix...
(gdb) run
```



```
----> DataBlockHost::DataBlockHost(DataBlock)...
----> ...returned
----> DataBlockHost::SyncToDevice()...
----> ...returned

Program received signal SIGSEGV, Segmentation fault.
0x00007ffff7c1329c in __strlen_evex () from /nix/store/wpgrc564ys39vbyv0m50qxmq8dvhi7cc-glibc-2.37-8/lib/libc.so.6
(gdb) bt
#0  0x00007ffff7c1329c in __strlen_evex () from /nix/store/wpgrc564ys39vbyv0m50qxmq8dvhi7cc-glibc-2.37-8/lib/libc.so.6
#1  0x00007ffff7b07d39 in __printf_buffer () from /nix/store/wpgrc564ys39vbyv0m50qxmq8dvhi7cc-glibc-2.37-8/lib/libc.so.6
#2  0x00007ffff7b08401 in __vfprintf_internal () from /nix/store/wpgrc564ys39vbyv0m50qxmq8dvhi7cc-glibc-2.37-8/lib/libc.so.6
#3  0x00007ffff7bc3c43 in __fprintf_chk () from /nix/store/wpgrc564ys39vbyv0m50qxmq8dvhi7cc-glibc-2.37-8/lib/libc.so.6
#4  0x00000000005404d2 in Kokkos::Impl::SharedAllocationRecord<void, void>::decrement(Kokkos::Impl::SharedAllocationRecord<void, void>*) ()
#5  0x00000000004396dd in Kokkos::Impl::SharedAllocationTracker::~SharedAllocationTracker (this=0x7fffffff7318, __in_chrg=<optimized out>)
    at /home/lesurg/src/idefix/src/kokkos/core/src/impl/Kokkos_SharedAlloc.hpp:419
#6  Kokkos::Impl::ViewTracker<Kokkos::View<double****, Kokkos::LayoutRight, Kokkos::Device<Kokkos::Serial, Kokkos::HostSpace>, Kokkos::Experimental::EmptyViewHoo
acker (this=0x7fffffff7318, __in_chrg=<optimized out>) at /home/lesurg/src/idefix/src/kokkos/core/src/impl/Kokkos_ViewTracker.hpp:39
#7  Kokkos::View<double****, Kokkos::LayoutRight, Kokkos::Device<Kokkos::Serial, Kokkos::HostSpace>, Kokkos::Experimental::EmptyViewHooks>::~View (this=0x7fffff
    __in_chrg=<optimized out>) at /home/lesurg/src/idefix/src/kokkos/core/src/Kokkos_View.hpp:1266
#8  DataBlockHost::~DataBlockHost (this=0x7fffffff7230, __in_chrg=<optimized out>) at /home/lesurg/src/idefix/src/dataBlock/dataBlockHost.hpp:25
#9  0x000000000051e33a in Setup::InitFlow (this=this@entry=0x7fffffff758f, data=...) at /home/lesurg/workdir/days2023/tutorials/debugging/problem1/setup.cpp:47
#10 0x000000000041205a in main (argc=<optimized out>, argv=<optimized out>) at /home/lesurg/src/idefix/src/main.cpp:126
(gdb) 
```

- The problem is definitely in DataBlockHost and Idefix is crap… Or ?

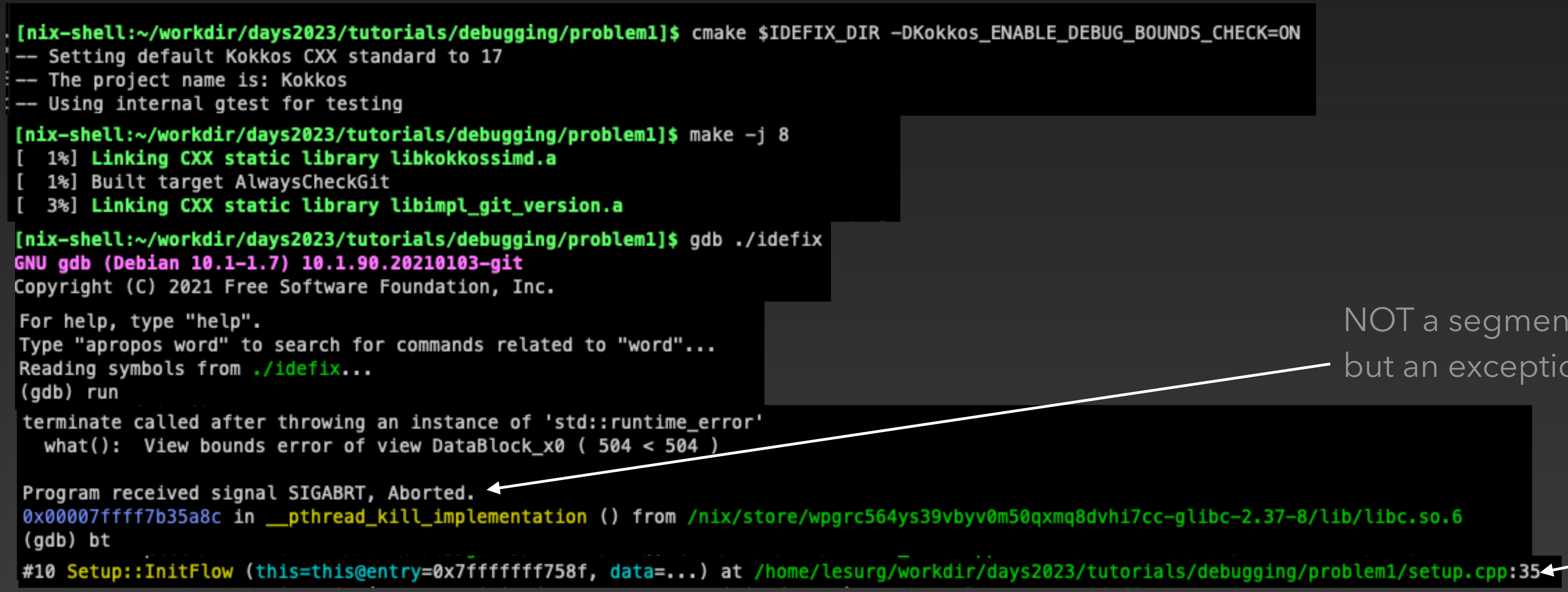
Case study I

A CPU-domain segmentation fault

RULE I: A segmentation fault always shows up *after* it actually happens

Sometimes, it can be 1000s of lines later

- Most likely problem: overflow of an array
- Need a way to check that the indices are always consistent with the array size: Kokkos_ENABLE_DEBUG_BOUNDS_CHECK!



```
[nix-shell:~/workdir/days2023/tutorials/debugging/problem1]$ cmake $IDEFIX_DIR -DKokkos_ENABLE_DEBUG_BOUNDS_CHECK=ON
-- Setting default Kokkos CXX standard to 17
-- The project name is: Kokkos
-- Using internal gtest for testing
[nix-shell:~/workdir/days2023/tutorials/debugging/problem1]$ make -j 8
[  1%] Linking CXX static library libkokkossimd.a
[  1%] Built target AlwaysCheckGit
[  3%] Linking CXX static library libimpl_git_version.a
[nix-shell:~/workdir/days2023/tutorials/debugging/problem1]$ gdb ./idefix
GNU gdb (Debian 10.1-1.7) 10.1.90.20210103-git
Copyright (C) 2021 Free Software Foundation, Inc.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./idefix...
(gdb) run
terminate called after throwing an instance of 'std::runtime_error'
  what():  View bounds error of view DataBlock_x0 ( 504 < 504 )

Program received signal SIGABRT, Aborted.
0x00007ffff7b35a8c in __pthread_kill_implementation () from /nix/store/wpgrc564ys39vbyv0m50qxmq8dvhi7cc-glibc-2.37-8/lib/libc.so.6
(gdb) bt
#10 Setup::InitFlow (this=this@entry=0x7fffffff758f, data=...) at /home/lesurg/workdir/days2023/tutorials/debugging/problem1/setup.cpp:35
```

NOT a segmentation fault,
but an exception raised by Kokkos!

line # is different

Case study I

A CPU-domain segmentation fault

Conclusions

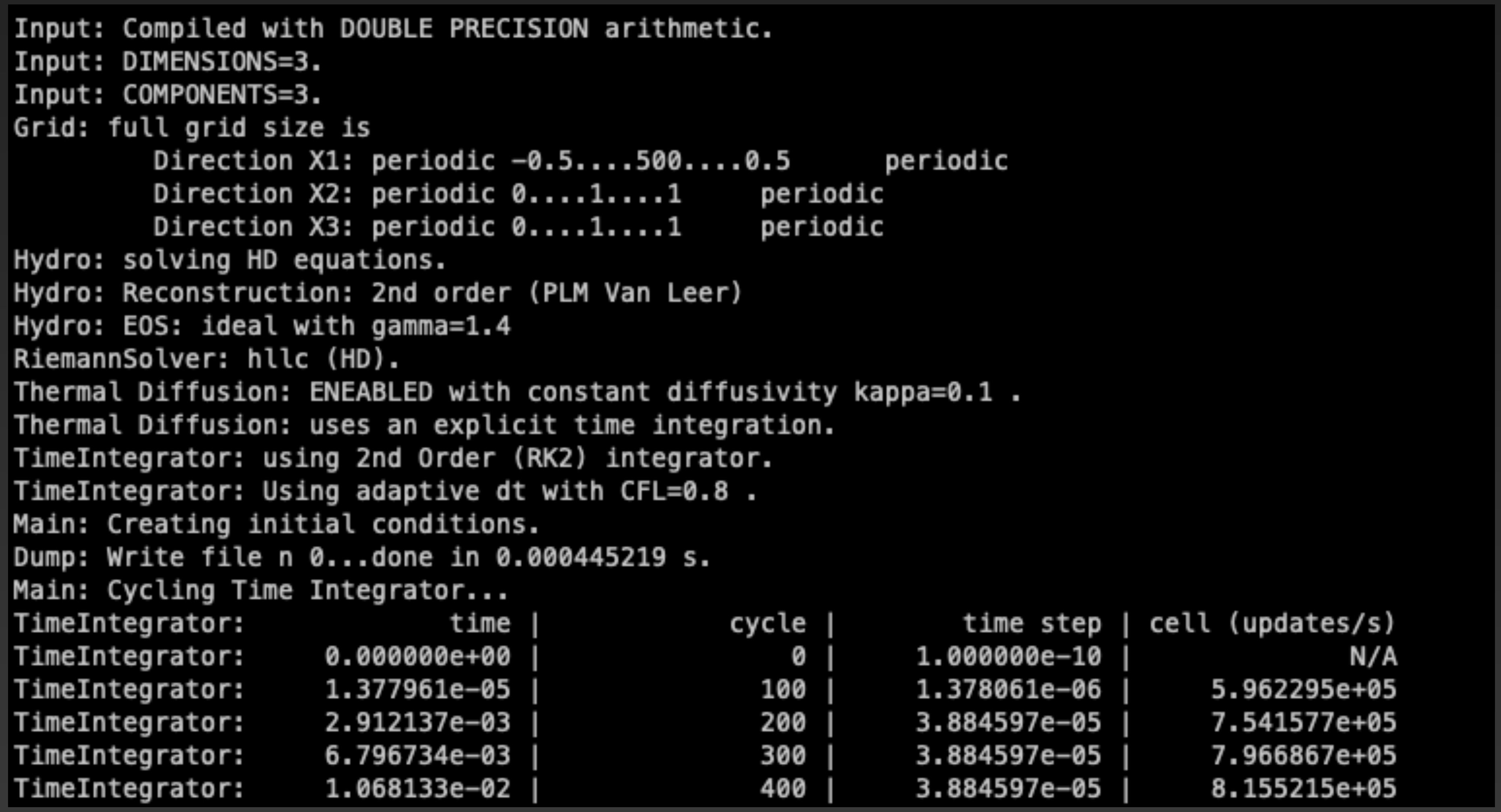
- The error is in the loop bounds in InitFlow (should be <, not <=)
- Good idea to check your code using `Kokkos_ENABLE_DEBUG_BOUNDS_CHECK` (even if it works!)
- When the system detects a segmentation fault, it is usually too late, the error happened before
- Use a debugger (lldb on macs, gdb on linux, cuda-gdb on nvidia, see later)
- Note the importance of `IdefixArray` labels when debugging, and `idfx::pushRegion()`.

Case study II

A GPU-domain segmentation fault

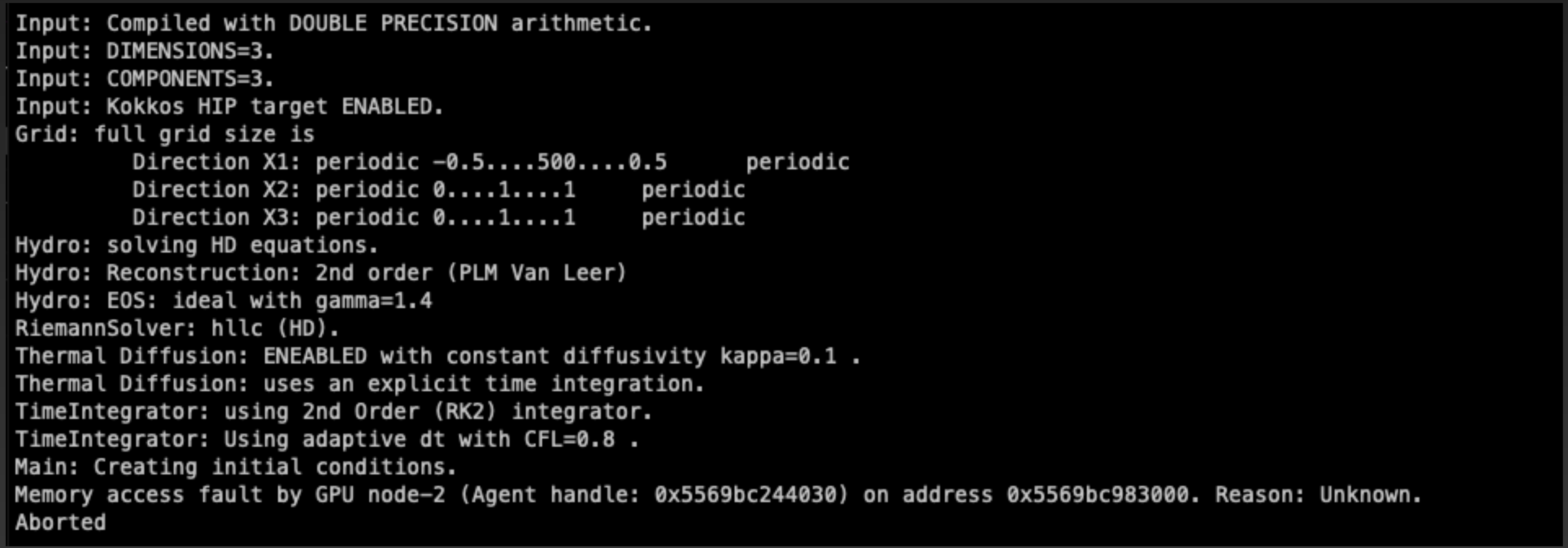
- go to tutorials/debugging/problem2 (thermal diffusion problem)
- Configure it for CPU (cmake without any option will do), run it
- Same on GPU…

CPU



```
Input: Compiled with DOUBLE PRECISION arithmetic.
Input: DIMENSIONS=3.
Input: COMPONENTS=3.
Grid: full grid size is
          Direction X1: periodic -0.5....500....0.5      periodic
          Direction X2: periodic 0....1....1     periodic
          Direction X3: periodic 0....1....1     periodic
Hydro: solving HD equations.
Hydro: Reconstruction: 2nd order (PLM Van Leer)
Hydro: EOS: ideal with gamma=1.4
RiemannSolver: hllc (HD).
Thermal Diffusion: ENEABLED with constant diffusivity kappa=0.1 .
Thermal Diffusion: uses an explicit time integration.
TimeIntegrator: using 2nd Order (RK2) integrator.
TimeIntegrator: Using adaptive dt with CFL=0.8 .
Main: Creating initial conditions.
Dump: Write file n 0...done in 0.000445219 s.
Main: Cycling Time Integrator...
TimeIntegrator:            time |             cycle |        time step | cell (updates/s)
TimeIntegrator:    0.000000e+00 |                 0 |     1.000000e-10 |              N/A
TimeIntegrator:    1.377961e-05 |               100 |     1.378061e-06 |     5.962295e+05
TimeIntegrator:    2.912137e-03 |               200 |     3.884597e-05 |     7.541577e+05
TimeIntegrator:    6.796734e-03 |               300 |     3.884597e-05 |     7.966867e+05
TimeIntegrator:    1.068133e-02 |               400 |     3.884597e-05 |     8.155215e+05
```

GPU (AMD/Rocm)

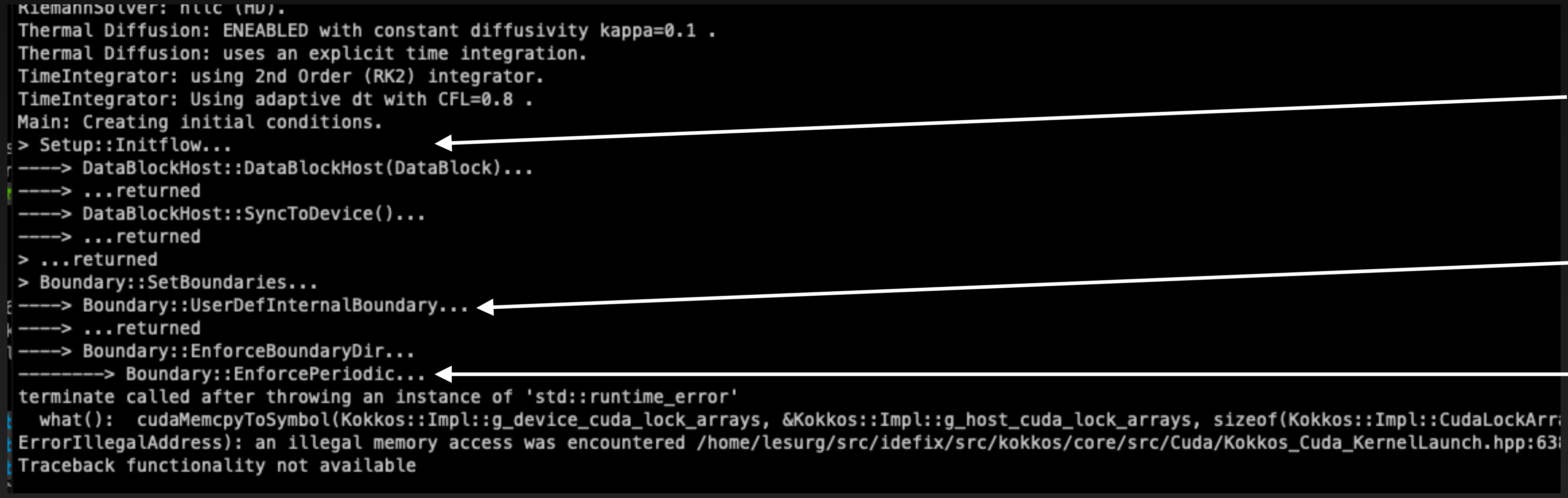


```
Input: Compiled with DOUBLE PRECISION arithmetic.
Input: DIMENSIONS=3.
Input: COMPONENTS=3.
Input: Kokkos HIP target ENABLED.
Grid: full grid size is
          Direction X1: periodic -0.5....500....0.5      periodic
          Direction X2: periodic 0....1....1     periodic
          Direction X3: periodic 0....1....1     periodic
Hydro: solving HD equations.
Hydro: Reconstruction: 2nd order (PLM Van Leer)
Hydro: EOS: ideal with gamma=1.4
RiemannSolver: hllc (HD).
Thermal Diffusion: ENEABLED with constant diffusivity kappa=0.1 .
Thermal Diffusion: uses an explicit time integration.
TimeIntegrator: using 2nd Order (RK2) integrator.
TimeIntegrator: Using adaptive dt with CFL=0.8 .
Main: Creating initial conditions.
Memory access fault by GPU node-2 (Agent handle: 0x5569bc244030) on address 0x5569bc983000. Reason: Unknown.
Aborted
```

Case study II

A GPU-domain segmentation fault

- same as case I: start with Idefix_DEBUG



```
RiemannSolver: hllc (HD).
Thermal Diffusion: ENEABLED with constant diffusivity kappa=0.1 .
Thermal Diffusion: uses an explicit time integration.
TimeIntegrator: using 2nd Order (RK2) integrator.
TimeIntegrator: Using adaptive dt with CFL=0.8 .
Main: Creating initial conditions.
> Setup::Initflow...
----> DataBlockHost::DataBlockHost(DataBlock)...
----> ...returned
----> DataBlockHost::SyncToDevice()...
----> ...returned
> ...returned
> Boundary::SetBoundaries...
----> Boundary::UserDefInternalBoundary...
----> ...returned
----> Boundary::EnforceBoundaryDir...
--------> Boundary::EnforcePeriodic...
terminate called after throwing an instance of 'std::runtime_error'
  what():  cudaMemcpyToSymbol(Kokkos::Impl::g_device_cuda_lock_arrays, &Kokkos::Impl::g_host_cuda_lock_arrays, sizeof(Kokkos::Impl::CudaLockArr
ErrorIllegalAddress): an illegal memory access was encountered /home/lesurg/src/idefix/src/kokkos/core/src/Cuda/Kokkos_Cuda_KernelLaunch.hpp:63
Traceback functionality not available
```

Our initial conditions

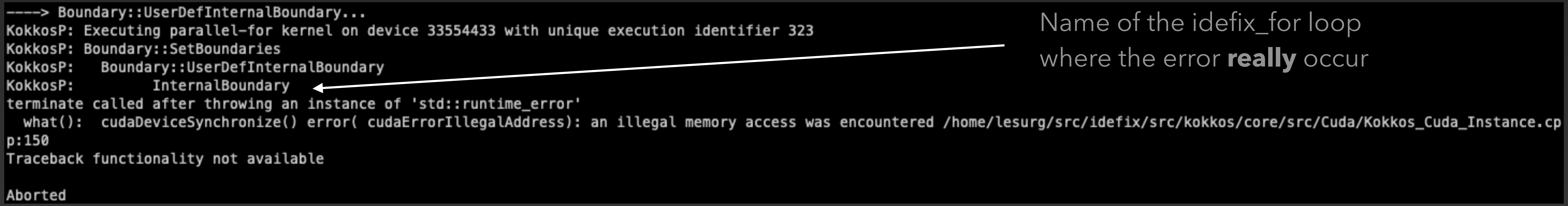
Our internal boundaries

So, it is in the periodic boundary conditions implemented in Idefix?!

- Enable Kokkos kernel logger:

`export KOKKOS_TOOLS_LIBS=$KOKKOS_TOOLS_DIR/debugging/kernel-logger/libkp_kernel_logger.so`

… and execute again idefix



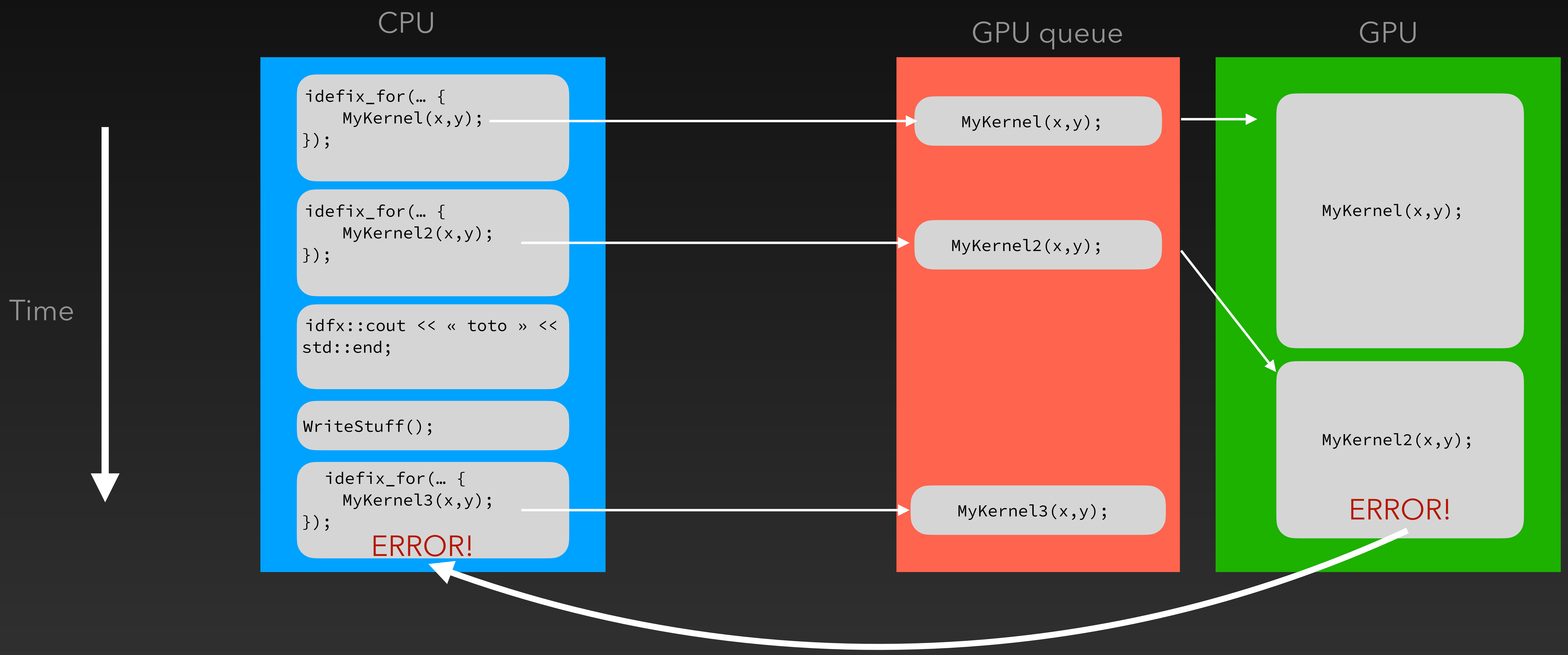
```
----> Boundary::UserDefInternalBoundary...
KokkosP: Executing parallel-for kernel on device 33554433 with unique execution identifier 323
KokkosP: Boundary::SetBoundaries
KokkosP:   Boundary::UserDefInternalBoundary
KokkosP:         InternalBoundary
terminate called after throwing an instance of 'std::runtime_error'
  what():  cudaDeviceSynchronize() error( cudaErrorIllegalAddress): an illegal memory access was encountered /home/lesurg/src/idefix/src/kokkos/core/src/Cuda/Kokkos_Cuda_Instance.cp
p:150
Traceback functionality not available

Aborted
```

Name of the idefix_for loop where the error **really** occur

Case study II

A GPU-domain segmentation fault

- The problem with GPU execution is called « non blocking dispatch »: each idefix_for (=« kernel ») is sent to the GPU for it to be executed, but there *is no guarantee* it has been executed when the call to idefix_for ends on the CPU! (similar to non-blocking MPI instructions)

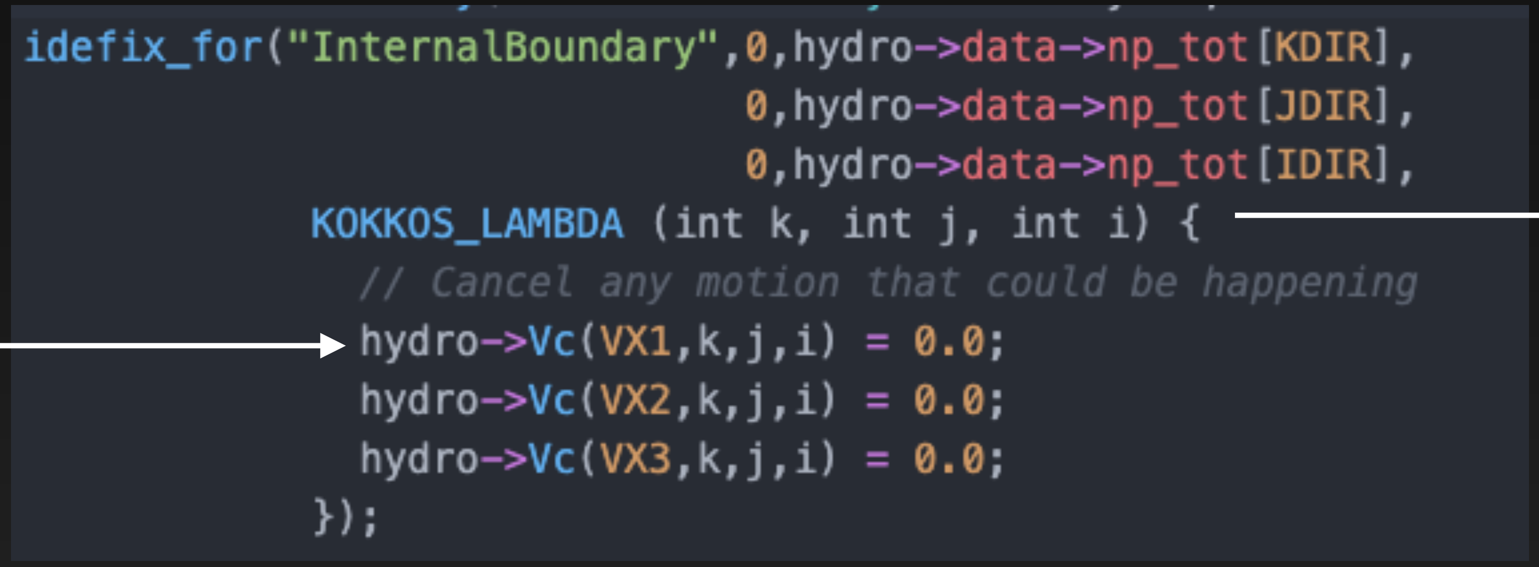


- Kokkos kernel logger forces synchronisation between CPU & GPU by introducing Kokkos::fence at the end of each idefix_for

Case study II

A GPU-domain segmentation fault

- Origin of the problem



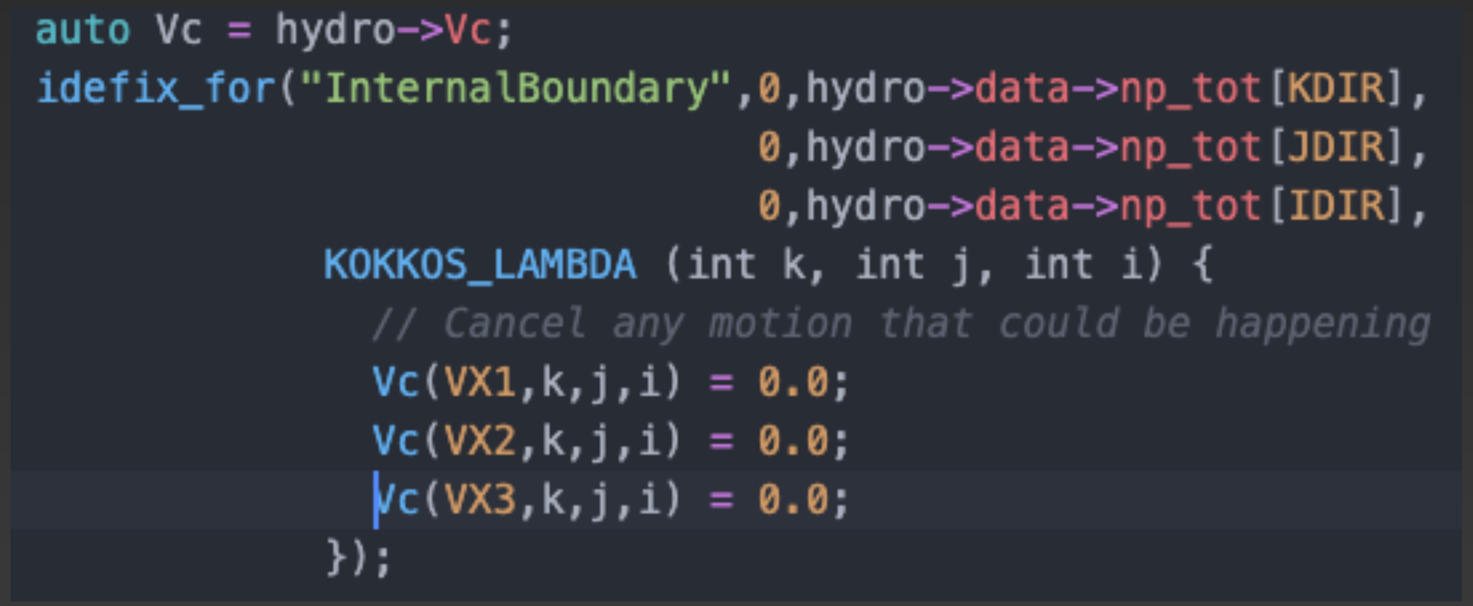
```
idefix_for("InternalBoundary",0,hydro->data->np_tot[KDIR],
                              0,hydro->data->np_tot[JDIR],
                              0,hydro->data->np_tot[IDIR],
            KOKKOS_LAMBDA (int k, int j, int i) {
              // Cancel any motion that could be happening
              hydro->Vc(VX1,k,j,i) = 0.0;
              hydro->Vc(VX2,k,j,i) = 0.0;
              hydro->Vc(VX3,k,j,i) = 0.0;
            });
```

pointer to a hydro object in *Host memory*

Whatever is inside that loop executes on the GPU!

RULE II: Never use pointers to Host space in idefix_for loops
RULE III: Always shallow copy whatever you need.

- Solution: shallow copy the array needed before calling `idefix_for`



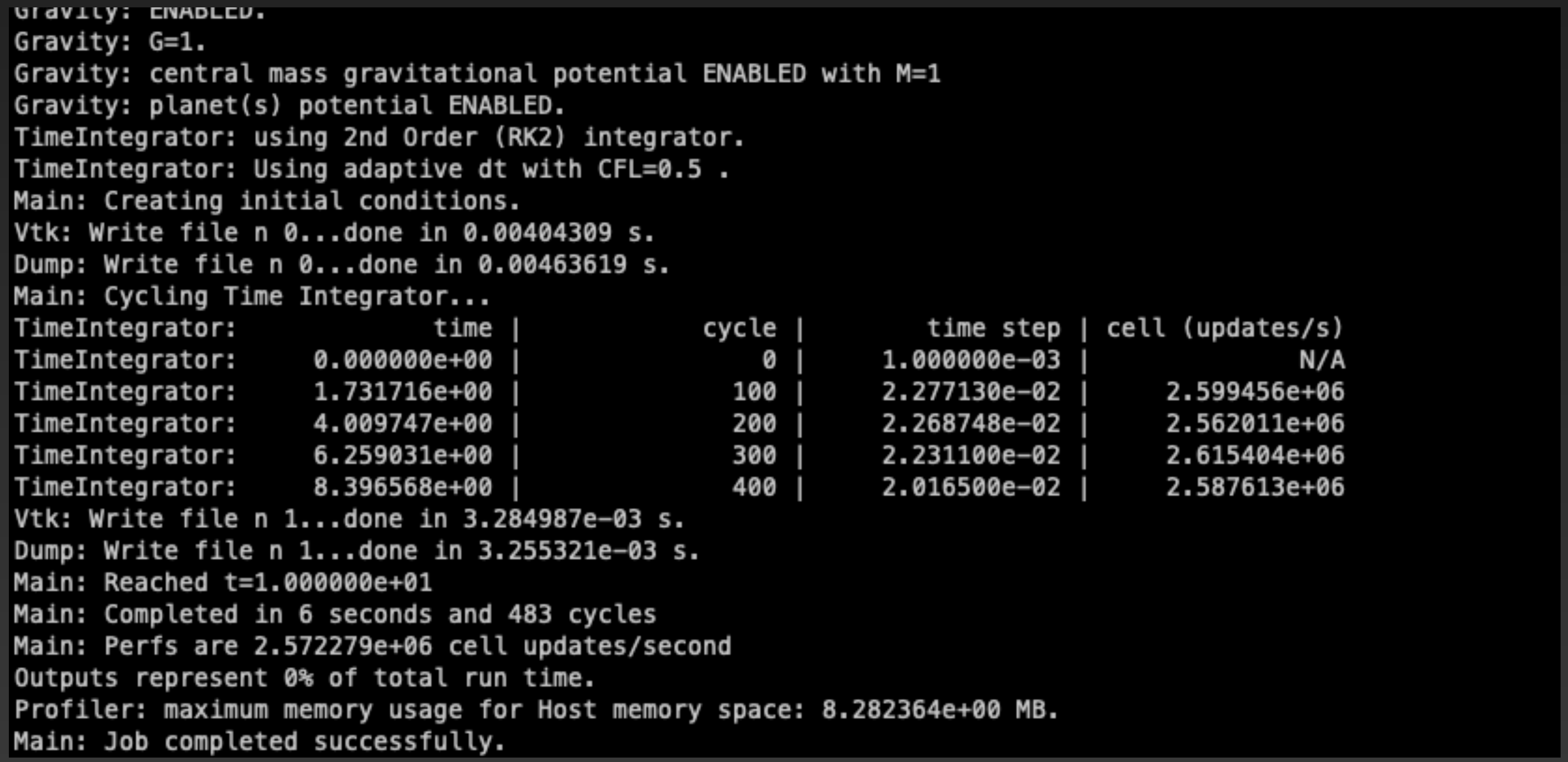
```
auto Vc = hydro->Vc;
idefix_for("InternalBoundary",0,hydro->data->np_tot[KDIR],
                              0,hydro->data->np_tot[JDIR],
                              0,hydro->data->np_tot[IDIR],
            KOKKOS_LAMBDA (int k, int j, int i) {
              // Cancel any motion that could be happening
              Vc(VX1,k,j,i) = 0.0;
              Vc(VX2,k,j,i) = 0.0;
              Vc(VX3,k,j,i) = 0.0;
            });
```

Case study III

A GPU-domain segmentation fault in a C++ class

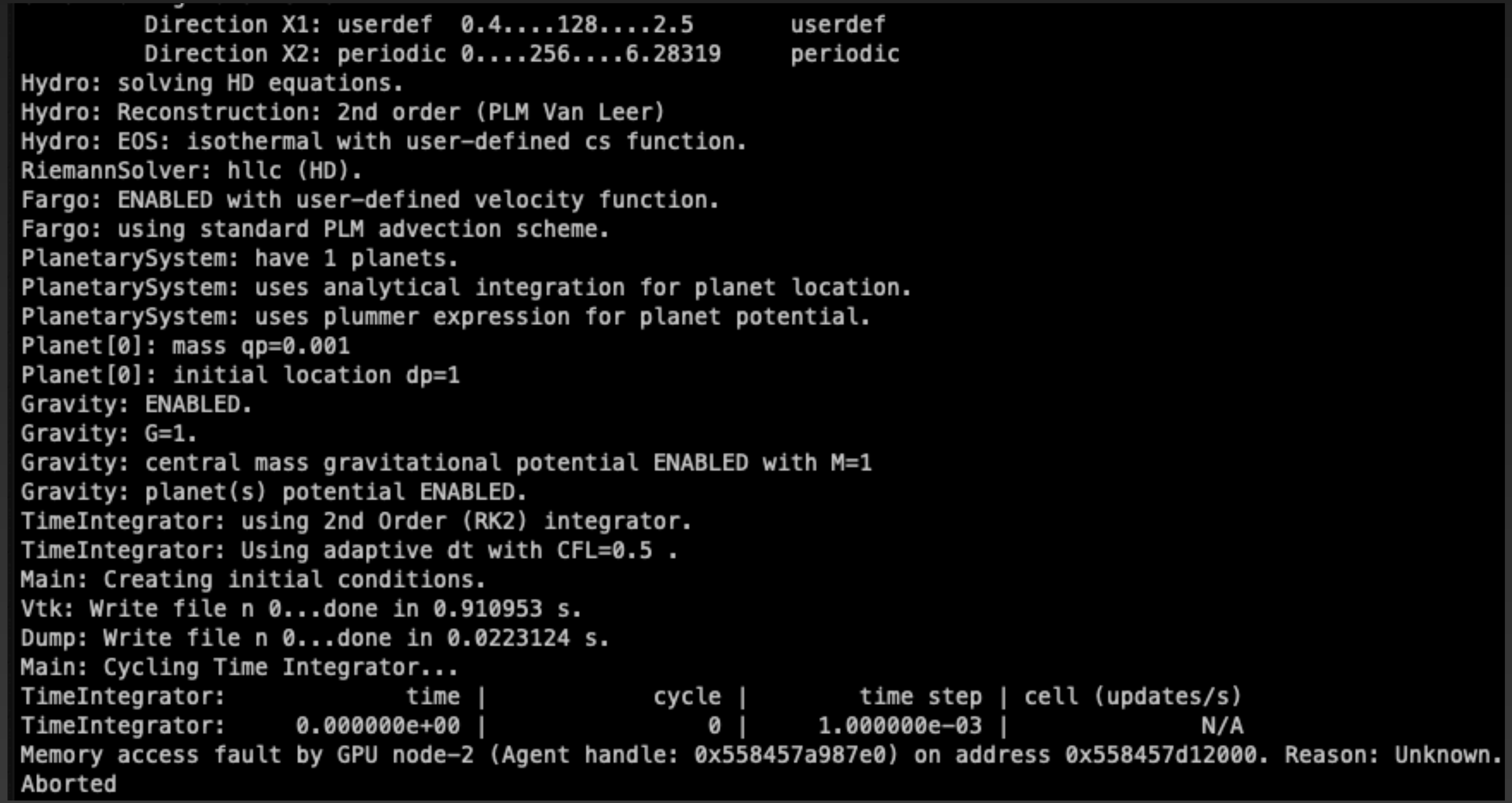
- go to tutorials/debugging/problem3 (disk+planet problem)
- Configure it for CPU (cmake without any option will do), run it
- Same on GPU…

CPU



```
Gravity: ENABLED.
Gravity: G=1.
Gravity: central mass gravitational potential ENABLED with M=1
Gravity: planet(s) potential ENABLED.
TimeIntegrator: using 2nd Order (RK2) integrator.
TimeIntegrator: Using adaptive dt with CFL=0.5 .
Main: Creating initial conditions.
Vtk: Write file n 0...done in 0.00404309 s.
Dump: Write file n 0...done in 0.00463619 s.
Main: Cycling Time Integrator...
TimeIntegrator:              time |              cycle |        time step | cell (updates/s)
TimeIntegrator:     0.000000e+00 |                  0 |     1.000000e-03 |              N/A
TimeIntegrator:     1.731716e+00 |                100 |     2.277130e-02 |     2.599456e+06
TimeIntegrator:     4.009747e+00 |                200 |     2.268748e-02 |     2.562011e+06
TimeIntegrator:     6.259031e+00 |                300 |     2.231100e-02 |     2.615404e+06
TimeIntegrator:     8.396568e+00 |                400 |     2.016500e-02 |     2.587613e+06
Vtk: Write file n 1...done in 3.284987e-03 s.
Dump: Write file n 1...done in 3.255321e-03 s.
Main: Reached t=1.000000e+01
Main: Completed in 6 seconds and 483 cycles
Main: Perfs are 2.572279e+06 cell updates/second
Outputs represent 0% of total run time.
Profiler: maximum memory usage for Host memory space: 8.282364e+00 MB.
Main: Job completed successfully.
```

GPU (AMD/Rocm)

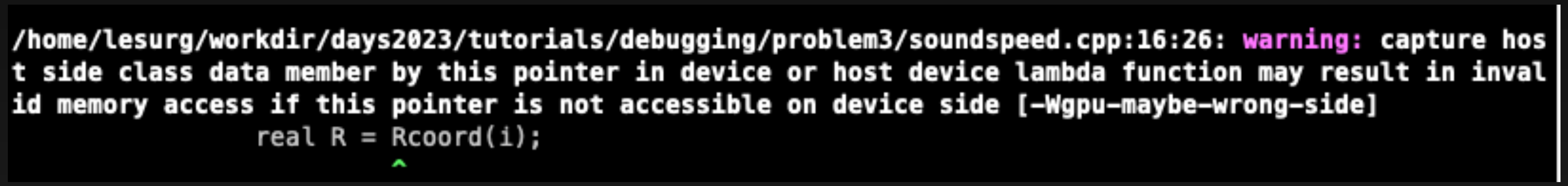


```
        Direction X1: userdef  0.4....128....2.5       userdef
        Direction X2: periodic 0....256....6.28319     periodic
Hydro: solving HD equations.
Hydro: Reconstruction: 2nd order (PLM Van Leer)
Hydro: EOS: isothermal with user-defined cs function.
RiemannSolver: hllc (HD).
Fargo: ENABLED with user-defined velocity function.
Fargo: using standard PLM advection scheme.
PlanetarySystem: have 1 planets.
PlanetarySystem: uses analytical integration for planet location.
PlanetarySystem: uses plummer expression for planet potential.
Planet[0]: mass qp=0.001
Planet[0]: initial location dp=1
Gravity: ENABLED.
Gravity: G=1.
Gravity: central mass gravitational potential ENABLED with M=1
Gravity: planet(s) potential ENABLED.
TimeIntegrator: using 2nd Order (RK2) integrator.
TimeIntegrator: Using adaptive dt with CFL=0.5 .
Main: Creating initial conditions.
Vtk: Write file n 0...done in 0.910953 s.
Dump: Write file n 0...done in 0.0223124 s.
Main: Cycling Time Integrator...
TimeIntegrator:              time |              cycle |        time step | cell (updates/s)
TimeIntegrator:     0.000000e+00 |                  0 |     1.000000e-03 |              N/A
Memory access fault by GPU node-2 (Agent handle: 0x558457a987e0) on address 0x558457d12000. Reason: Unknown.
Aborted
```

Case study III

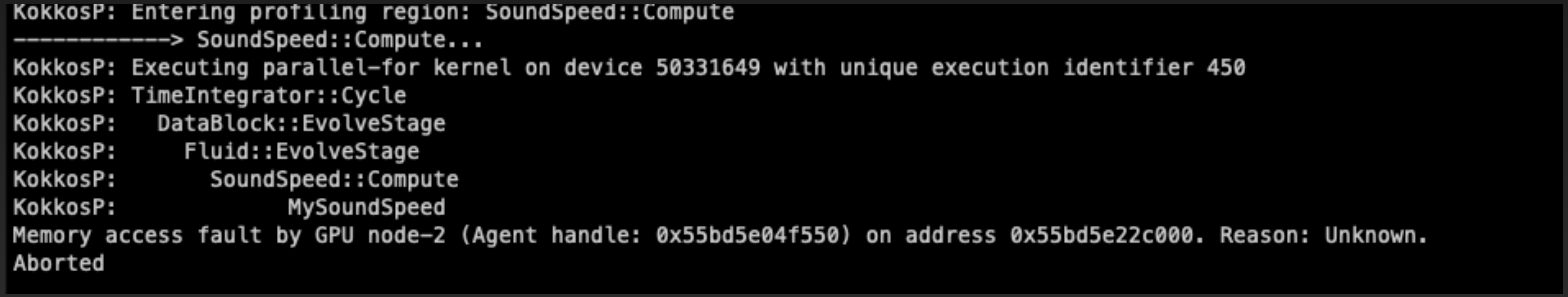
A GPU-domain segmentation fault in a C++ class

- Notice the warning on GPU (not always…)



```
/home/lesurg/workdir/days2023/tutorials/debugging/problem3/soundspeed.cpp:16:26: warning: capture hos
t side class data member by this pointer in device or host device lambda function may result in inval
id memory access if this pointer is not accessible on device side [-Wgpu-maybe-wrong-side]
                real R = Rcoord(i);
                         ^
```

- Enable kokkos-kernel-logger+IDEFIX_GPU

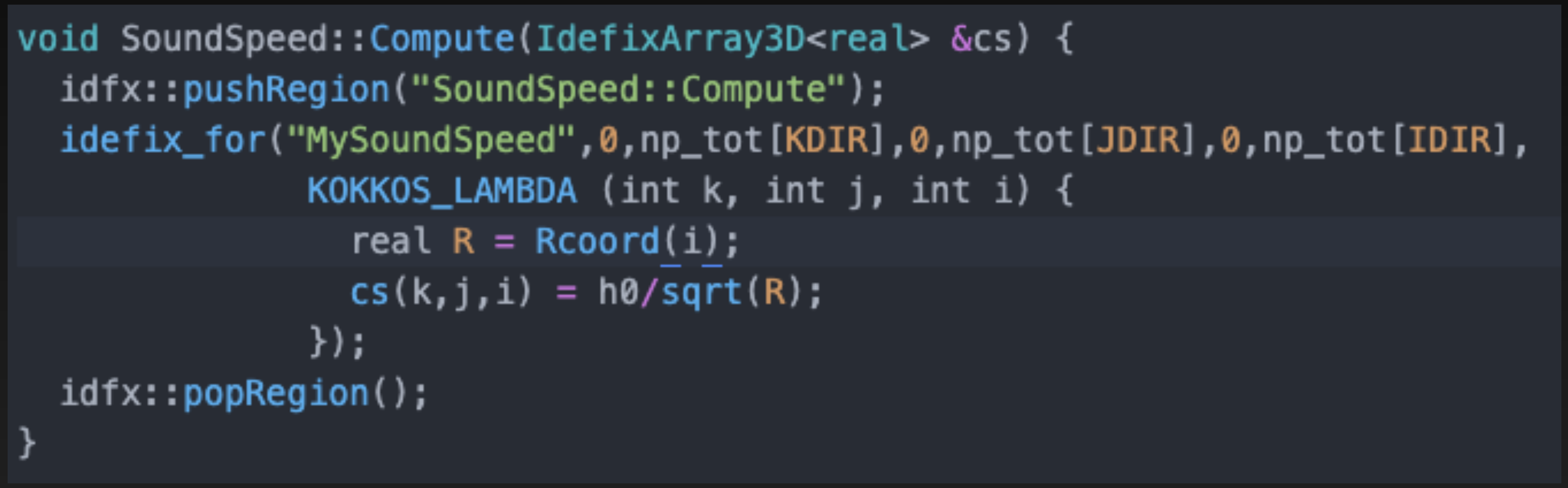


```
KokkosP: Entering profiling region: SoundSpeed::Compute
------------> SoundSpeed::Compute...
KokkosP: Executing parallel-for kernel on device 50331649 with unique execution identifier 450
KokkosP: TimeIntegrator::Cycle
KokkosP:   DataBlock::EvolveStage
KokkosP:     Fluid::EvolveStage
KokkosP:       SoundSpeed::Compute
KokkosP:             MySoundSpeed
Memory access fault by GPU node-2 (Agent handle: 0x55bd5e04f550) on address 0x55bd5e22c000. Reason: Unknown.
Aborted
```

-> Problem is clearly in the idefix_for called in SoundSpeed::Compute

Case study III

A GPU-domain segmentation fault in a C++ class



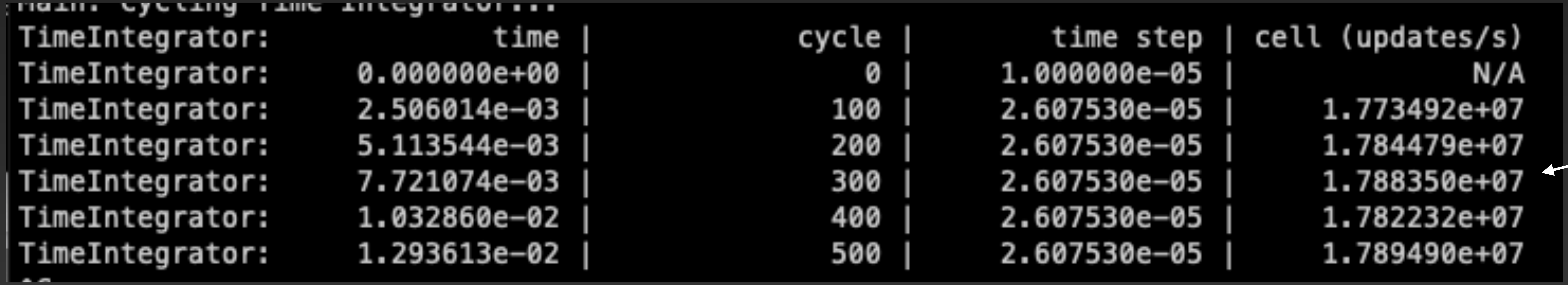
```
void SoundSpeed::Compute(IdefixArray3D<real> &cs) {
  idfx::pushRegion("SoundSpeed::Compute");
  idefix_for("MySoundSpeed",0,np_tot[KDIR],0,np_tot[JDIR],0,np_tot[IDIR],
              KOKKOS_LAMBDA (int k, int j, int i) {
                real R = Rcoord(i);
                cs(k,j,i) = h0/sqrt(R);
              });
  idfx::popRegion();
}
```

- What is `Rcoord`? what is `h0`?
- For the compiler, these are member variables of the `SoundSpeed` class
- Hence, they are transformed into `this->Rcoord` and `this->h0`
- The pointer `this->` is in host space! `Back to rule II and III`
- Very common bug, check https://github.com/kokkos/kokkos/issues/695

Case Study 4

A slow execution

- go to tutorials/debugging/problem4 (disk+planet problem, again)
- Compile & run on your favorite GPU
- Checkout the code performances (NB: expect at least 1e8 cell/sec per GPUs!)



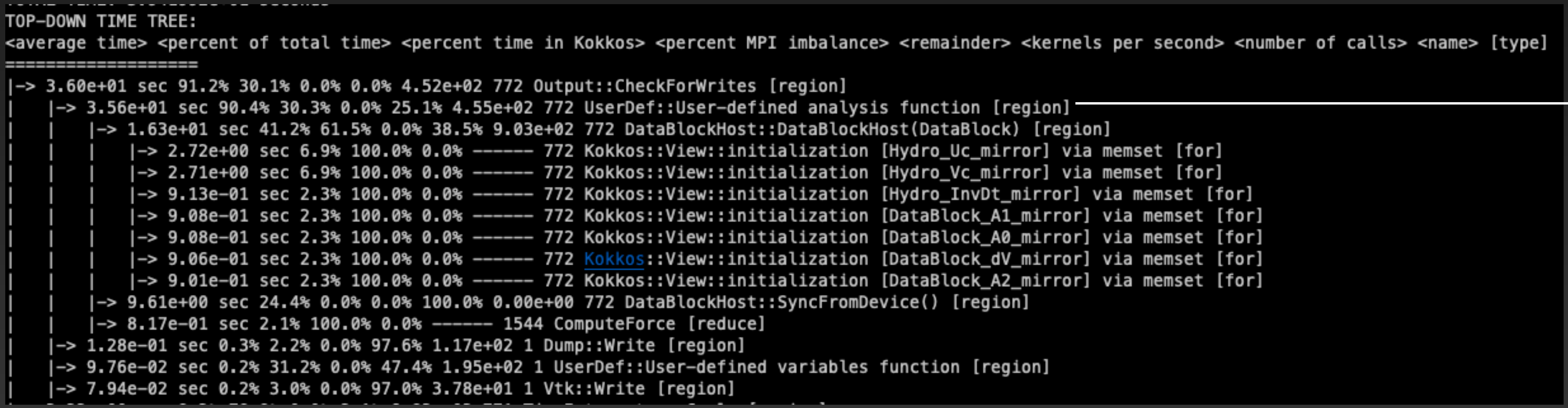
```
TimeIntegrator:            time |          cycle |      time step | cell (updates/s)
TimeIntegrator:    0.000000e+00 |              0 |   1.000000e-05 |              N/A
TimeIntegrator:    2.506014e-03 |            100 |   2.607530e-05 |     1.773492e+07
TimeIntegrator:    5.113544e-03 |            200 |   2.607530e-05 |     1.784479e+07
TimeIntegrator:    7.721074e-03 |            300 |   2.607530e-05 |     1.788350e+07
TimeIntegrator:    1.032860e-02 |            400 |   2.607530e-05 |     1.782232e+07
TimeIntegrator:    1.293613e-02 |            500 |   2.607530e-05 |     1.789490e+07
```

low!

Case Study 4

A slow execution

- Use the space-time-stack provided by Kokkos:
  `export KOKKOS_TOOLS_LIBS=$KOKKOS_TOOLS_DIR/profiling/space-time-stack/libkp_space_time_stack.so`
- No need to recompile, just re-run the same executable. Profiling is performed on the fly…



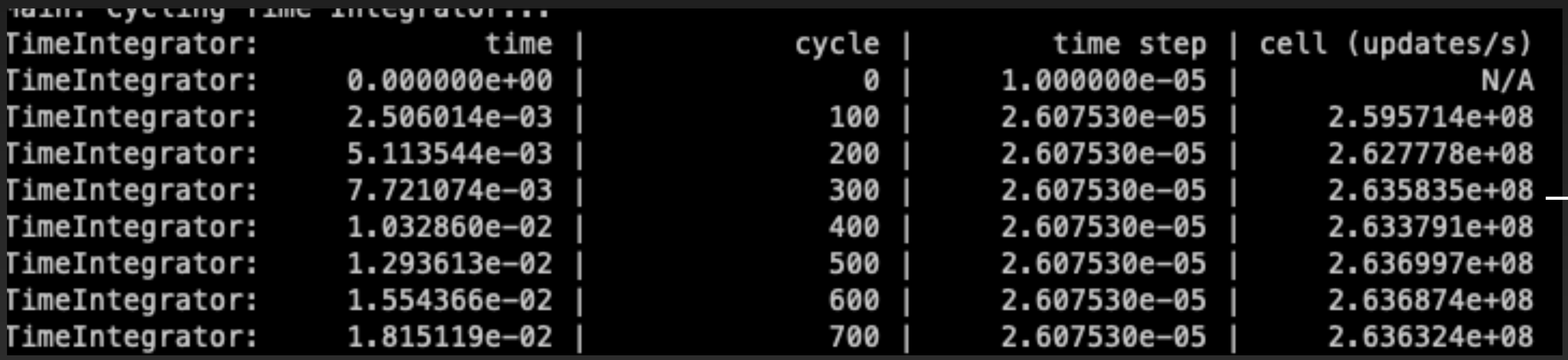
```
TOP-DOWN TIME TREE:
<average time> <percent of total time> <percent time in Kokkos> <percent MPI imbalance> <remainder> <kernels per second> <number of calls> <name> [type]
===================
|-> 3.60e+01 sec 91.2% 30.1% 0.0% 0.0% 4.52e+02 772 Output::CheckForWrites [region]
|   |-> 3.56e+01 sec 90.4% 30.3% 0.0% 25.1% 4.55e+02 772 UserDef::User-defined analysis function [region]
|   |   |-> 1.63e+01 sec 41.2% 61.5% 0.0% 38.5% 9.03e+02 772 DataBlockHost::DataBlockHost(DataBlock) [region]
|   |   |   |-> 2.72e+00 sec 6.9% 100.0% 0.0% ------ 772 Kokkos::View::initialization [Hydro_Uc_mirror] via memset [for]
|   |   |   |-> 2.71e+00 sec 6.9% 100.0% 0.0% ------ 772 Kokkos::View::initialization [Hydro_Vc_mirror] via memset [for]
|   |   |   |-> 9.13e-01 sec 2.3% 100.0% 0.0% ------ 772 Kokkos::View::initialization [Hydro_InvDt_mirror] via memset [for]
|   |   |   |-> 9.08e-01 sec 2.3% 100.0% 0.0% ------ 772 Kokkos::View::initialization [DataBlock_A1_mirror] via memset [for]
|   |   |   |-> 9.08e-01 sec 2.3% 100.0% 0.0% ------ 772 Kokkos::View::initialization [DataBlock_A0_mirror] via memset [for]
|   |   |   |-> 9.06e-01 sec 2.3% 100.0% 0.0% ------ 772 Kokkos::View::initialization [DataBlock_dV_mirror] via memset [for]
|   |   |   |-> 9.01e-01 sec 2.3% 100.0% 0.0% ------ 772 Kokkos::View::initialization [DataBlock_A2_mirror] via memset [for]
|   |   |-> 9.61e+00 sec 24.4% 0.0% 0.0% 100.0% 0.00e+00 772 DataBlockHost::SyncFromDevice() [region]
|   |   |-> 8.17e-01 sec 2.1% 100.0% 0.0% ------ 1544 ComputeForce [reduce]
|   |-> 1.28e-01 sec 0.3% 2.2% 0.0% 97.6% 1.17e+02 1 Dump::Write [region]
|   |-> 9.76e-02 sec 0.2% 31.2% 0.0% 47.4% 1.95e+02 1 UserDef::User-defined variables function [region]
|   |-> 7.94e-02 sec 0.2% 3.0% 0.0% 97.0% 3.78e+01 1 Vtk::Write [region]
```

Most of the time is spent in the user-defined analysis!

Case Study 4

A slow execution

- The culprit is the frequency of the analysis routine in idefix.ini, which is set to 0 (i.e. every integration loop!)
- Set it to 0.01, and things get back to normal



```
TimeIntegrator:            time |            cycle |        time step | cell (updates/s)
TimeIntegrator:    0.000000e+00 |                0 |     1.000000e-05 |              N/A
TimeIntegrator:    2.506014e-03 |              100 |     2.607530e-05 |     2.595714e+08
TimeIntegrator:    5.113544e-03 |              200 |     2.607530e-05 |     2.627778e+08
TimeIntegrator:    7.721074e-03 |              300 |     2.607530e-05 |     2.635835e+08
TimeIntegrator:    1.032860e-02 |              400 |     2.607530e-05 |     2.633791e+08
TimeIntegrator:    1.293613e-02 |              500 |     2.607530e-05 |     2.636997e+08
TimeIntegrator:    1.554366e-02 |              600 |     2.607530e-05 |     2.636874e+08
TimeIntegrator:    1.815119e-02 |              700 |     2.607530e-05 |     2.636324e+08
```

Almost x20!

RULE IV: always check that the performances are what you expect