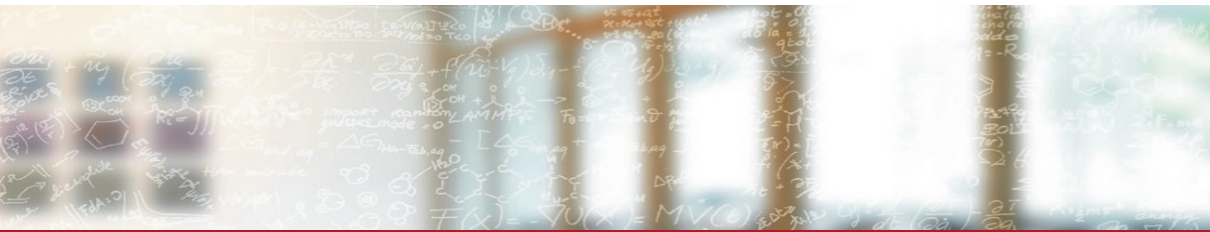




**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich



## Profiling and Debugging

Directive Based GPU programming course 2018

Vasileios Karakasis, CSCS

May 14–15, 2018

# Overview

- Why and where my code crashes?
- Why my code does not perform as “expected”?

# Debugging

OpenACC translates to CUDA code, so you may use the corresponding tools:

- `cuda-memcheck`: Check for memory errors and race conditions
- `cuda-gdb`: Debug the generated CUDA kernels
- `nvprof`+`nvvp`: Detailed performance profiling

Other CUDA-aware tools:

- Allinea DDT: Debug MPI, CUDA, OpenMP applications + memory checking

Compiler-related diagnostics:

- Code generation diagnostics
- PGI debugger (`pgdb`)
- PGI profiler (`pgprof`)
- CrayPAT profiler

# Debugging

## cuda-memcheck

```
$ srun -n1 -Cgpu cuda-memcheck ./blur.openacc
```

```
===== Invalid __global__ read of size 8
=====      at 0x00000098 in blur_twice_gpu_nocopies_84_gpu(double*, double*, int, int)
=====      by thread (66,0,0) in block (8192,0,0)
=====      Address 0x10253e00210 is out of bounds
=====      Saved host backtrace up to driver entry point at kernel launch time
=====      Host Frame:/opt/cray/nvidia/default/lib64/libcuda.so (cuLaunchKernel + 0x2cd) [0x23ce3d]
=====      Host Frame:/apps/common/UES/pgi/18.4/linux86-64/18.4/lib/libaccn.so (__pgi_uacc_cuda_launch3 + 0x10) [0x10]
=====      Host Frame:/apps/common/UES/pgi/18.4/linux86-64/18.4/lib/libaccn.so [0x17950]
=====      Host Frame:/apps/common/UES/pgi/18.4/linux86-64/18.4/lib/libaccn.so (__pgi_uacc_cuda_launch + 0x10) [0x10]
=====      Host Frame:/apps/common/UES/pgi/18.4/linux86-64/18.4/lib/libaccg.so (__pgi_uacc_launch + 0x1ac) [0x1ac]
=====      Host Frame:./blur.openacc [0x52a5]
=====      Host Frame:./blur.openacc [0x57df]
=====      Host Frame:/lib64/libc.so.6 (__libc_start_main + 0xf5) [0x206e5]
=====      Host Frame:./blur.openacc [0x3489]
```

# Debugging

cuda-gdb

Compile with `-g -Mcuda=debug`

```
$ srtn -n1 -u -Cgpu cuda-gdb ./blur.openacc

(cuda-gdb) b blur_openacc.cpp:86
Breakpoint 1 at 0x40542b: file ./blur_openacc.cpp, line 86.
(cuda-gdb) r
Starting program: /users/karakasv/Devel/openacc-training/solutions/shared/./blur.openacc
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib64/libthread_db.so.1".
dispersion 1D test of length n = 1048580 : 8.00003MB
[New Thread 0x2aaab52b2700 (LWP 17350)]
[New Thread 0x2aaab54b3700 (LWP 17351)]
[Switching focus to CUDA kernel 0, grid 3, block (1,0,0), thread (0,0,0), device 0, sm 2, warp 0, lane 0]

Breakpoint 1, blur_twice_gpu_nocopies_84_gpu<<<(8193,1,1),(128,1,1)>>> (
    out=0x10253200000, in=0x10252800000) at blur_openacc.cpp:87
87      for (auto i = 0; i < n; ++i) {
```

# Debugging

## Using DDT

Arm DDT - Arm Forge 18.1.1

File Edit View Control Tools Window Help

Current Group: All Focus on current: C Group Process Thread Step Threads Together Step CUDA threads by: Warp (default)

Create Group

CUDA Threads (Process 1. diffusion\_gpu\_17\_gpu) Block 0 Thread 0 Go Grid size: 128x1x1 Block size: 128x1x1

Project Files

diffusion2d\_openacc\_mpi.cpp diffusion2d.hpp diffusion2d.hpp

Application Code

Headers

diffusion2d.hpp

Utilities

diffusion2d\_openacc\_mpi.c

main(int argc, char\*\* ar

External Code

```
2 #include <iostream>
3 #include <fstream>
4
5
6 void diffusion_gpu(const double *x0, double *x1, int nx, int ny, double dt)
7 {
8     int i, j;
9     auto width = nx+2;
10
11 #ifdef OPENACC_DATA
12     #pragma acc parallel loop collapse(2) private(i,j) \
13     present(x0[0:nx*ny], x1[0:nx*ny])
14 #else
15     #pragma acc parallel loop deviceptr(x0,x1) collapse(2) private(i,j)
16 #endif
17 for (j = 1; j < ny+1; ++j) {
18     for (i = 1; i < nx+1; ++i) {
19         auto pos = i + j*width;
20         x1[pos] = x0[pos] + dt * (-4.*x0[pos]
21                                 + x0[pos-width] + x0[pos+width]
22                                 + x0[pos-1] + x0[pos+1]);
23     }
24 }
25
26
27 template<typename T>
28 void copy_gpu(T *dst, const T *src, int n)
29 {
30     int i;
31
32 #ifdef OPENACC_DATA
33     #pragma acc parallel loop present(dst[0:n], src[0:n])
34 #else
35     #pragma acc parallel loop deviceptr(dst,src)
36 #endif
37 for (i = 0; i < n; ++i) {
38     dst[i] = src[i];
39 }
```

Locals

Current Line(s)

Current Stack

GPU Devices

Variable Name	Value
dt	0.10000000000000001
x0	0x102160000000
x1	0
x1	0x10216021200
x1	0

Type: none selected

Input/Output

Breakpoints Watchpoints Stacks Kernel Progress View Tracepoints Tracepoint Output Logbook

Input/Output

arm: job 7450576 queued and waiting for resources

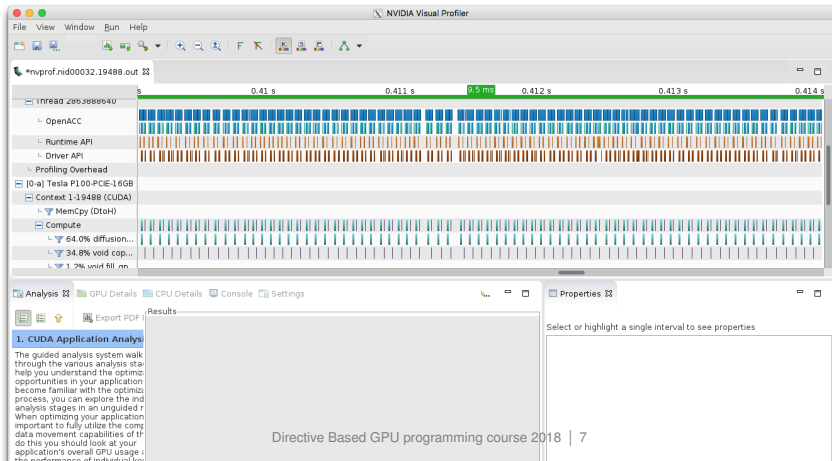
arm: job 7450576 has been allocated resources

## 2 MPI rank(s)

# Profiling

Using nvprof & nvvp

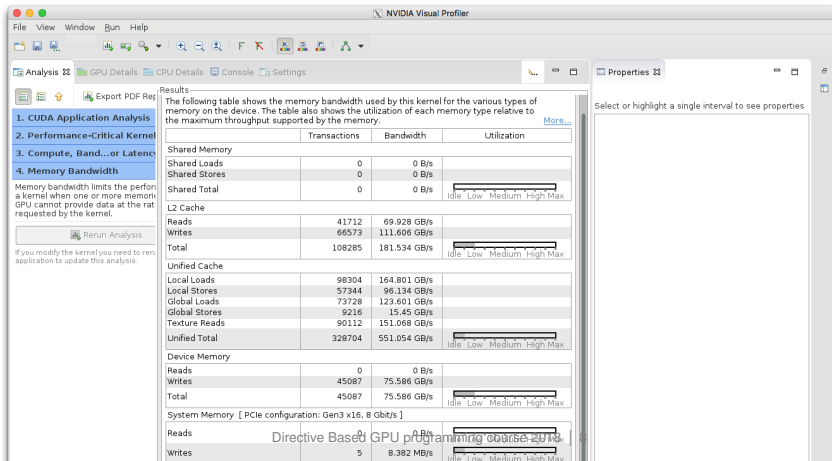
```
srunk -N2 -Cgpu nvprof -o nvprof.%h.%p.out  
./diffusion2d.openacc.mpi
```



# Profiling

Using nvprof & nvvp – Detailed analysis

```
srunk -N2 -Cgpu nvprof --analysis-metrics -o nvprof.%h.%p.out  
./diffusion2d.openacc.mpi
```





# Compiler diagnostics

## GPU kernel information

- Compile with special flags (-Minfo=accel for PGI, -hmsg for Cray)

```
diffusion_gpu(const double *, double *, int, int, double):
    6, include "diffusion2d.hpp"
    17, Generating present(x0[:nx*ny],x1[:nx*ny])
        Accelerator kernel generated
        Generating Tesla code
    17, #pragma acc loop gang, vector(128) collapse(2) /* blockIdx.x threadIdx.x */
    18, /* blockIdx.x threadIdx.x collapsed */

main:
    72, Generating create(x0[:buffer_size])
        Generating copyout(x1[:buffer_size])
void fill_gpu<double>(T1 *, T1, int):
    6, include "diffusion2d.hpp"
    53, Generating present(v[:n])
        Accelerator kernel generated
        Generating Tesla code
    53, #pragma acc loop gang, vector(128) /* blockIdx.x threadIdx.x */
```

More during the hands-on!

