Отладка: терминология

- *Фрейм* (stack frame) – “кадр” состояния функции в момент останова или передачи управления во вложенную функцию
- *Трасса* – полная иерархия фреймов от точки входа в программу до положения останова
- *Breakpoint* – точка останова в коде программы
- *Watchpoint* – точка останова по условию достижения заданного состояния

Отладка: терминология

- Просмотр значений переменных и вычисление несложных выражений
- Назначение точек и условий останова
- Перемещение по стеку фреймов
- Отладка нескольких потоков
- Анализ отладочного дампа (core dump) в offline-режиме

Исполняемый формат ELF

- Секции – разделы исполняемого файла, в которых информация сгруппирована по типу:

.text – исполняемый код
.bss – неинициализированые данные
.data – инициализированные данные
…
.debug – **отладочная информация**

Отладочная информация

- Таблица соответствия обычных имён символов декорированным (mangled)
- Таблица соответствия адресов исполняемого кода и строк исходного кода
- Любая дополнительная информация

Отладчик: ограничения

- Отладчики могут работать и без отладочной информации, но возможностей будет меньше
- Если код содержит оптимизации, то отладка может быть менее точна (позиционирование по коду, недоступность значений исключённых оптимизацией переменных, …)

GDB: рюшки

- Сокращённые имена команд: b – breakpoint, wa – watchpoint, …
- В командной строке GDB в любом месте можно нажать <TAB>, чтобы показать варианты автозаполнения (или заполнить сразу, если возможен только один вариант)

GDB: компиляция

- Флаг -g для включения отладочной информации

```
marcusmae@T61p samples]$ cd multigpu/pthread/pthread_cuda/
[marcusmae@T61p pthread_cuda]$ make
gcc -g -std=c99 -I/opt/cuda/include -c pthread_cuda.c -o pthread_cuda.o
nvcc -g -c pattern2d.cu -o pattern2d.o
gcc pthread_cuda.o pattern2d.o -o pthread_cuda
-lpthread -L/opt/cuda/lib64 -lcudart -lm
```

GDB: запуск

- Запуск отладчика одновременно с запуском приложения

```
[marcusmae@T61p pthread_cuda]$ gdb ./pthread_cuda
GNU gdb (GDB) Fedora (7.2-51.fc14)
Copyright (C) 2010 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-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/marcusmae/Samples/cuda-
training/samples/multigpu/pthread/pthread_cuda/pthread_cuda...done.
(gdb)
```

GDB: запуск (attach)

- Подключение к ранее запущенному приложению

```
[marcusmae@T61p pthread_cuda]$ gdb pthread_cuda 3346
Reading symbols from /home/marcusmae/Samples/cuda
-training/samples/multigpu/pthread/pthread_cuda/pthread_cuda...done.
Attaching to program: /home/marcusmae/Samples/cuda
-training/samples/multigpu/pthread/pthread_cuda/pthread_cuda, process 3346
(gdb)
```

GDB: breakpoint

- Установка breakpoint по имени функции

```
(gdb) b pattern2d_cpu
Breakpoint 1 at 0x401675: file pattern2d.cu, line 34.
(gdb) r
Starting program: /home/marcusmae/Samples/cuda
-training/samples/multigpu/pthread/pthread_cuda/pthread_cuda
[Thread debugging using libthread_db enabled]
1 CUDA device(s) found
[New Thread 0x7ffff6f44700 (LWP 3349)]

Breakpoint 1, pattern2d_cpu (bx=1, nx=128, ex=1, by=1, ny=128, ey=1,
    in=0x7ffff6f65010, out=0x7ffff6f55010, id=1) at pattern2d.cu:34
34   size_t size = nx * ny * sizeof(float);
(gdb)
```

GDB: breakpoint

- Установка breakpoint по номеру строки
- Удаление – d <номер>

```
(gdb) b 36
Breakpoint 2 at 0x40169b: file pattern2d.cu, line 36.
(gdb) b pattern2d.cu:37
Note: breakpoint 2 also set at pc 0x40169b.
Breakpoint 3 at 0x40169b: file pattern2d.cu, line 37.
(gdb) c
Continuing.

Breakpoint 2, pattern2d_cpu (bx=1, nx=128, ex=1, by=1, ny=128, ey=1,
    in=0x7ffff6f65010, out=0x7ffff6f55010, id=1) at pattern2d.cu:37
37   for (int j = by; j < ny - ey; j++)
(gdb)
```

GDB: watchpoint

- Установка watchpoint по адресу в памяти

```
(gdb) b 38
Breakpoint 4 at 0x4016a6: file pattern2d.cu, line 38.
(gdb) c
Continuing.

Breakpoint 4, pattern2d_cpu (bx=1, nx=128, ex=1, by=1, ny=128, ey=1,
    in=0x7ffff6f55010, out=0x7ffff6f65010, id=1) at pattern2d.cu:38
38     for (int i = bx; i < nx - ex; i++)
(gdb) p &j
$1 = (int *) 0x7fffffffdf3c
(gdb) wa *(int*)0x7fffffffdf3c
Hardware watchpoint 5: *(int*)0x7fffffffdf3c
```

GDB: watchpoint

Достижение watchpoint

```
(gdb) d 4
(gdb) c
Continuing.
Hardware watchpoint 5: *(int*)0x7fffffffdf3c

Old value = 1
New value = 2
0x00000000004017a2 in pattern2d_cpu (bx=1, nx=128, ex=1, by=1, ny=128, ey=1,
    in=0x7ffff6f55010, out=0x7ffff6f65010, id=1) at pattern2d.cu:37
37   for (int j = by; j < ny - ey; j++)
```

GDB: контекст

- list – просмотр исходного кода в месте останова

```
(gdb) list
32   float* in, float* out, int id)
33{
34   size_t size = nx * ny * sizeof(float);
35   memset(out, 0, size);
36
37   for (int j = by; j < ny - ey; j++)
38     for (int i = bx; i < nx - ex; i++)
39       OUT(i,j) = sqrtf(fabs(IN(i,j) + IN(i-1,j) + IN(i+1,j) -
40          2.0f * IN(i,j-1) + 3.0f * IN(i,j+1)));
41   return 0;
(gdb)
```

GDB: контекст

- Ctrl-x-a – “оконный” TUI-режим

```
    ----pattern2d.cu-------------------------------------------------------------------------
    |33  {                                                                                  |
B+  |34      size_t size = nx * ny * sizeof(float);                                        |
    |35      memset(out, 0, size);                                                         |
B+  |36                                                                                    |
B+> |37      for (int j = by; j < ny - ey; j++)                                            |
    |38        for (int i = bx; i < nx - ex; i++)                                          |
    |39            OUT(i,j) = sqrtf(fabs(IN(i,j) + IN(i-                                   |
    |40                2.0f * IN(i,j-1) + 3.0f * IN(                                       |
    |41      return 0;                                                                     |
    |42  }                                                                                 |
    |43                                                                                    |
    |44  // GPU device kernel.                                                            |
    |45  __global__ void pattern2d_gpu_kernel(i,j+1)                                       |
    ----------------------------------------------------------------------------------------
multi-thre Thread 0x7ffff In: pattern2d_cpu       Line: 37 PC: 0x
(gdb)
```

GDB: многопоточность

- info th – информация о запущенных потоках
- t <номер> - переключение в поток

```
(gdb) info th
  2 Thread 0x7ffff6f44700 (LWP 3349)  0x0000003befed8997 in ioctl ()
   from /lib64/libc.so.6
* 1 Thread 0x7ffff786c740 (LWP 3346)  0x00000000004017a2 in pattern2d_cpu (
    bx=1, nx=128, ex=1, by=1, ny=128, ey=1, in=0x7ffff6f55010,
    out=0x7ffff6f65010, id=1) at pattern2d.cu:37
(gdb) t 2
[Switching to thread 2 (Thread 0x7ffff6f44700 (LWP 3349))]#0  0x0000003befed8997
in ioctl () from /lib64/libc.so.6
(gdb)
```

GDB: трасса

bt – трасса

```
(gdb) bt
#0  0x0000003befed8997 in ioctl () from /lib64/libc.so.6
#1  0x00007ffff710b923 in ?? () from /usr/lib64/libcuda.so
...
#10 0x00007ffff70e7b41 in ?? () from /usr/lib64/libcuda.so
#11 0x00007ffff7daa82e in ?? () from /opt/cuda/lib64/libcudart.so.4
#12 0x00007ffff7daad7b in ?? () from /opt/cuda/lib64/libcudart.so.4
#13 0x00007ffff7dab764 in ?? () from /opt/cuda/lib64/libcudart.so.4
#14 0x00007ffff7da09b6 in ?? () from /opt/cuda/lib64/libcudart.so.4
#15 0x00007ffff7d98659 in ?? () from /opt/cuda/lib64/libcudart.so.4
#16 0x00007ffff7dbdf88 in cudaMalloc () from /opt/cuda/lib64/libcudart.so.4
#17 0x0000000000400dea in thread_func (arg=0x611cf0) at pthread_cuda.c:68
#18 0x0000003bf0206ccb in start_thread () from /lib64/libpthread.so.0
#19 0x0000003befee0c2d in clone () from /lib64/libc.so.6
(gdb)
```

GDB: переключение фрейма

- f <номер> – переключить фрейм

```
(gdb) f 17
#17 0x0000000000400dea in thread_func (arg=0x611cf0) at pthread_cuda.c:68
68   cuda_status = cudaMalloc((void**)&config->in_dev, size);
(gdb) list
63   }
64
65   size_t size = config->nx * config->ny * sizeof(float);
66
67   // Create device arrays for input and output data.
68   cuda_status = cudaMalloc((void**)&config->in_dev, size);
69   if (cuda_status != cudaSuccess)
70   {
71     fprintf(stderr, "Cannot allocate CUDA input buffer on device %d, status
72       idevice, cuda_status);
(gdb)
```

CUDA-GDB

- Вариант GDB с расширениями для отладки ядер CUDA
- Поддержка отладки на одном GPU или multi-GPU
- Следуя GPL, доступен в исходниках:
  ftp://download.nvidia.com/CUDAOpen64/

CUDA-GDB: требования

- Требует один свободный GPU

fatal: All CUDA devices are used for X11 and cannot be used while debugging. (error code = 24)

➡ в GNU/Linux в текстовом режиме можно работать с cuda-отладчиком на машине с одним GPU

- Можно запустить не более одного сеанса cuda-gdb

Found an already running instance of cuda-gdb. If you believe you are seeing this message in error, try deleting /tmp/cuda-gdb.lock

CUDA-GDB: компиляция

- Флаг -g для включения отладочной информации на хосте, -G – на GPU

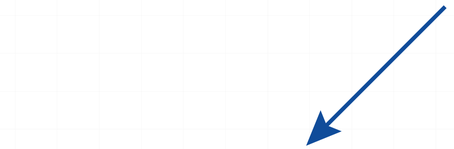
```
[marcusmae@T61p pattern2d]$ make
nvcc -g -G -c pattern2d.cu
gcc -g -std=c99 -c -I/usr/include/ImageMagick draw_diffs.c
nvcc -g pattern2d.o draw_diffs.o -o pattern2d -lMagickCore
```

CUDA-GDB: запуск

```
(cuda-gdb) b pattern2d_gpu_kernel
Breakpoint 1 at 0x4032ba: file pattern2d.cu, line 38.
(cuda-gdb) r
Starting program: /home/dmikushin/Programming/cuda
-training/samples/multigpu/pthread/pthread_cuda/pthread_cuda
[Thread debugging using libthread_db enabled]
[New process 26320]
[New Thread 139849021003584 (LWP 26320)]
8 CUDA device(s) found
[New Thread 139849009342208 (LWP 26324)]
[New Thread 139849000949504 (LWP 26325)]
[New Thread 139848992556800 (LWP 26326)]
[New Thread 139848984164096 (LWP 26327)]
[New Thread 139848975771392 (LWP 26328)]
[New Thread 139848899884800 (LWP 26329)]
[New Thread 139848891492096 (LWP 26330)]
[New Thread 139848883099392 (LWP 26331)]
Device 6 initialized
Device 0 initialized
Device 7 initialized
```

CUDA-GDB: запуск

Events



```
Device 5 initialized
Device 3 initialized
Device 1 initialized
[Launch of CUDA Kernel 0 (pattern2d_gpu_kernel) on Device 6]
Device 4 initialized
Device 2 initialized
[Launch of CUDA Kernel 1 (pattern2d_gpu_kernel) on Device 1]
[Termination of CUDA Kernel 0 (pattern2d_gpu_kernel) on Device 6]
[Launch of CUDA Kernel 2 (pattern2d_gpu_kernel) on Device 2]
[Launch of CUDA Kernel 3 (pattern2d_gpu_kernel) on Device 7]
[Launch of CUDA Kernel 4 (pattern2d_gpu_kernel) on Device 4]
[Termination of CUDA Kernel 3 (pattern2d_gpu_kernel) on Device 7]
[Launch of CUDA Kernel 5 (pattern2d_gpu_kernel) on Device 0]
[Launch of CUDA Kernel 6 (pattern2d_gpu_kernel) on Device 7]
[Switching to CUDA Kernel 5 (<<<(0,0),(0,0,0)>>>)]

Breakpoint 1, pattern2d_gpu_kernel<<<(3,7),(32,16,1)>>> (bx=1, nx=128,
    ex=1, by=1, ny=128, ey=1, block_length=32, block_height=16,
    in=0x100000, out=0x110000, id=0) at pattern2d.cu:52
52   int i = blockIdx.x * block_length + threadIdx.x + bx;
```

CUDA-GDB: расширения

Отладка ядра: вывод индексов и размерностей

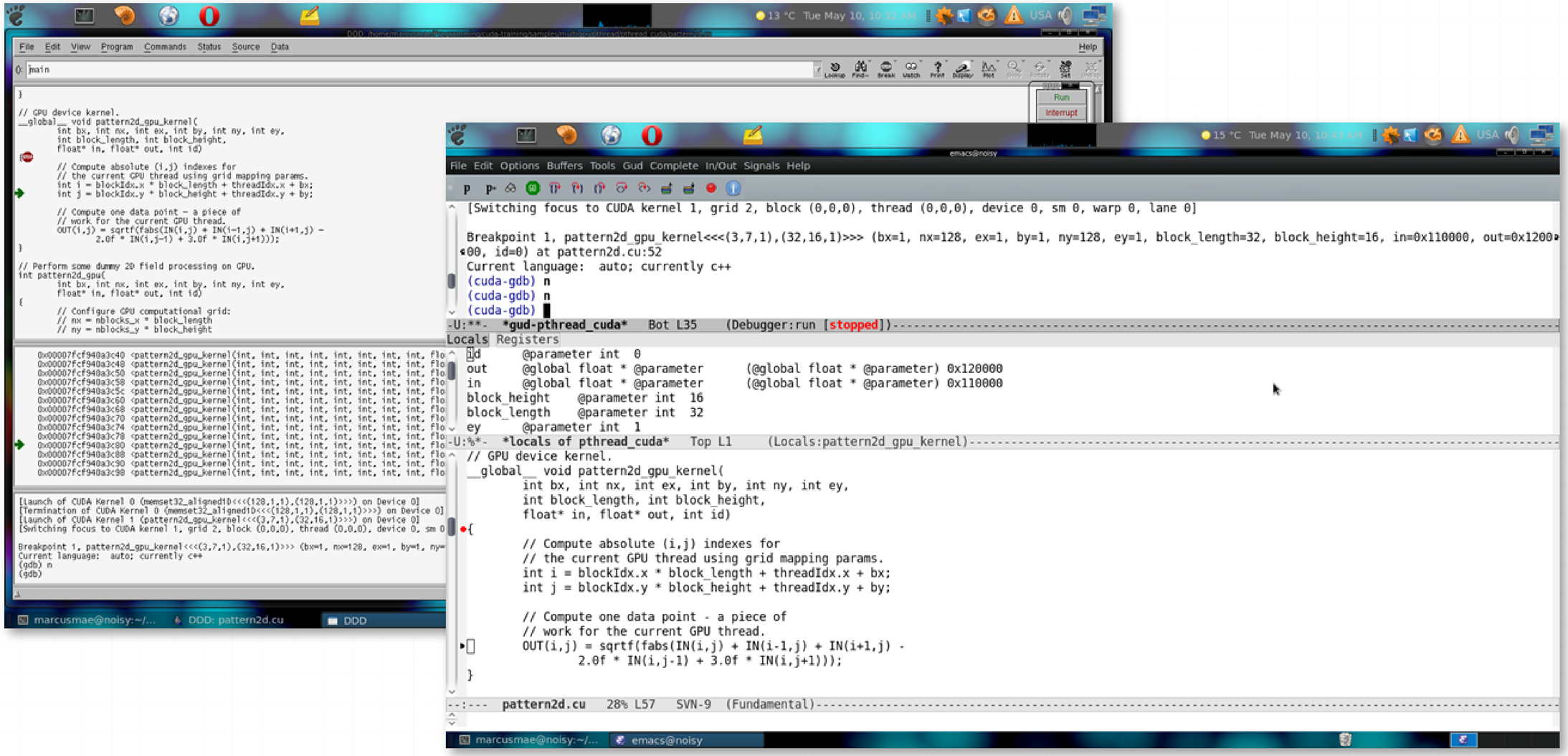
```
(cuda-gdb) b 57
Breakpoint 2 at 0x7fdbac085e98: file pattern2d.cu, line 57.
(cuda-gdb) c
Continuing.
[Termination of CUDA Kernel 8 (pattern2d_gpu_kernel) on Device 4]

Breakpoint 2, pattern2d_gpu_kernel<<<(3,7),(32,16,1)>>> (bx=1, nx=128,
    ex=1, by=1, ny=128, ey=1, block_length=32, block_height=16,
    in=0x100000, out=0x110000, id=0) at pattern2d.cu:57
57   OUT(i,j) = sqrtf(fabs(IN(i,j) + IN(i-1,j) + IN(i+1,j) -
(cuda-gdb) p blockIdx
$1 = {x = 0, y = 0}
```

CUDA-GDB: расширения

- Отладка ядра: переключение текущего блока/потока

```
(cuda-gdb) cuda block 2,1 thread 15,4,0
[Switching to CUDA Kernel 1 (device 0, sm 15, warp 4,
lane 15, grid 2, block (2,1), thread (15,4,0))]
57   OUT(i,j) = sqrtf(fabs(IN(i,j) + IN(i-1,j) + IN(i+1,j) -
(cuda-gdb) p i
$2 = 80
(cuda-gdb) p j
$3 = 21
```

CUDA-GDB: DDD и emacs



Заключение

- Представленный метод отладки универсально полезен, так как кроме GPU применим к широкому классу встраиваемых систем

- Чем раньше Вы столкнётесь с достаточно серьёзной проблемой, чтобы пришлось освоить отладчик, тем лучше! :-)

CUDA-MEMCHECK

- Диагностика ошибок в CUDA-ядрах
- Выявление конкретной причины “Kernel launch failure”
- Может быть запущен отдельно или внутри отладчика

```
cuda-gdb) set cuda memcheck on
```

CUDA-MEMCHECK

- Адрес вне допустимых границ

```
[marcusmae@T61p pthread_cuda]$ cuda-memcheck ./pthread_cuda
========= CUDA-MEMCHECK
1 CUDA device(s) found
Device 0 initialized
Cannot execute CUDA kernel #2 by device 0, status = 4
Cannot execute pattern 2d on device 0, status = 4
========= Invalid __global__ read of size 4
=========     at 0x00000170 in pattern2d_gpu_kernel
=========     by thread (0,0,0) in block (1,0,0)
=========     Address 0x00100ef4 is out of bounds
=========
========= ERROR SUMMARY: 1 error
```