COMPREHENSIVE GENERAL LUMI COURSE
WARSAW, POLAND

INTRODUCTION TO ROCGDB

SUYASH TANDON, SAMUEL ANTAO, BOB ROBEY

JAKUB KURZAK - PRESENTER

ADVANCED MICRO DEVICES, INC.

AMD
together we advance_

slides on LUMI in /project/project_465000644/Slides/AMD/

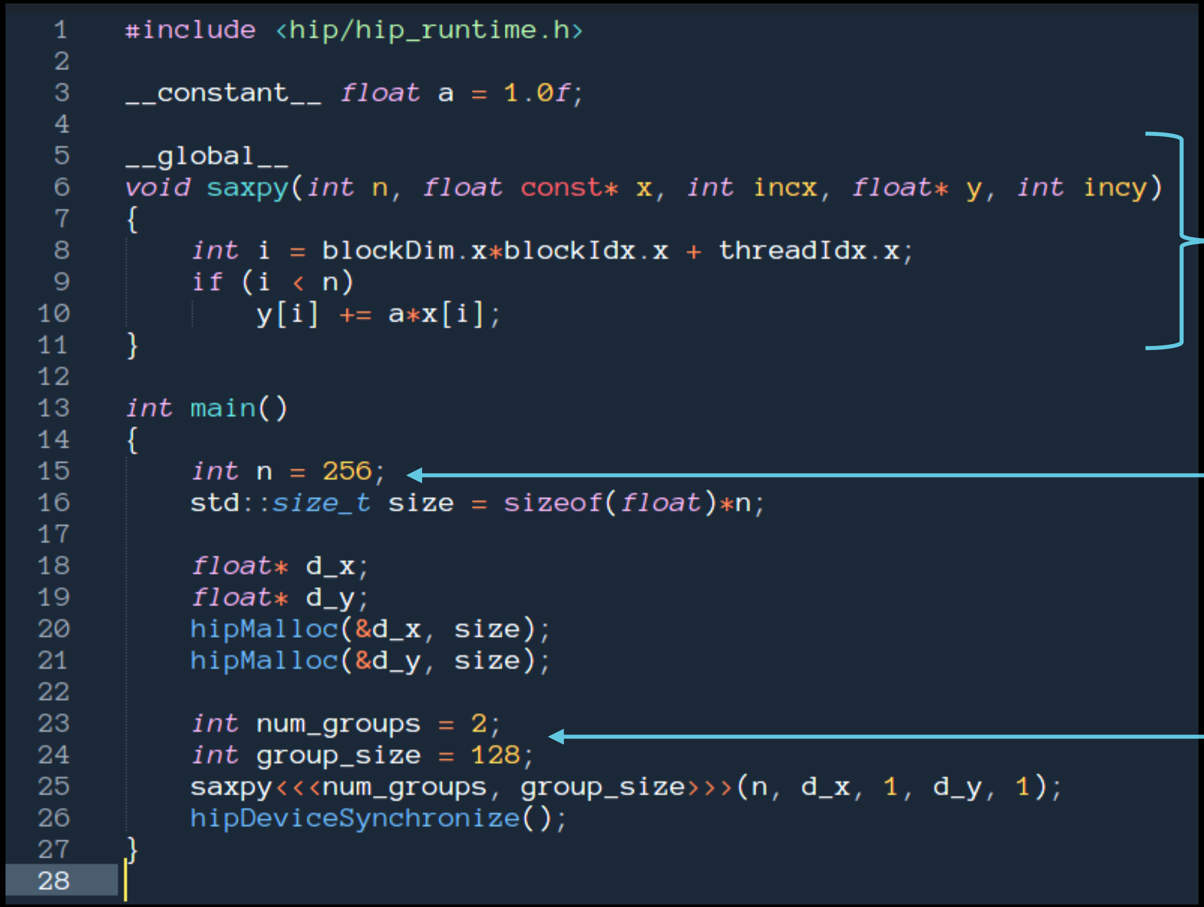
hands-on exercises: https://hackmd.io/@sfantao/H1QU6xRR3

hands-on source code: /project/project_465000644/Exercises/AMD/HPCTrainingExamples/

rocgdb

- AMD ROCm source-level debugger for Linux®
- based on the GNU Debugger (GDB)
  - tracks upstream GDB master
  - standard GDB commands for both CPU and GPU debugging
- considered a prototype
  - focus on source line debugging
  - no symbolic variable debugging yet

simple saxpy kernel

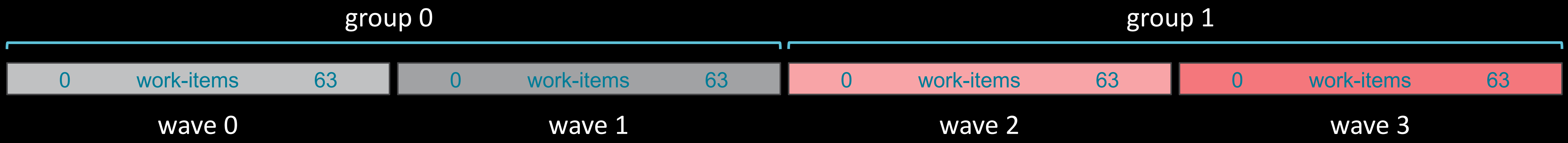


```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n, d_x, 1, d_y, 1);
    hipDeviceSynchronize();
}

```

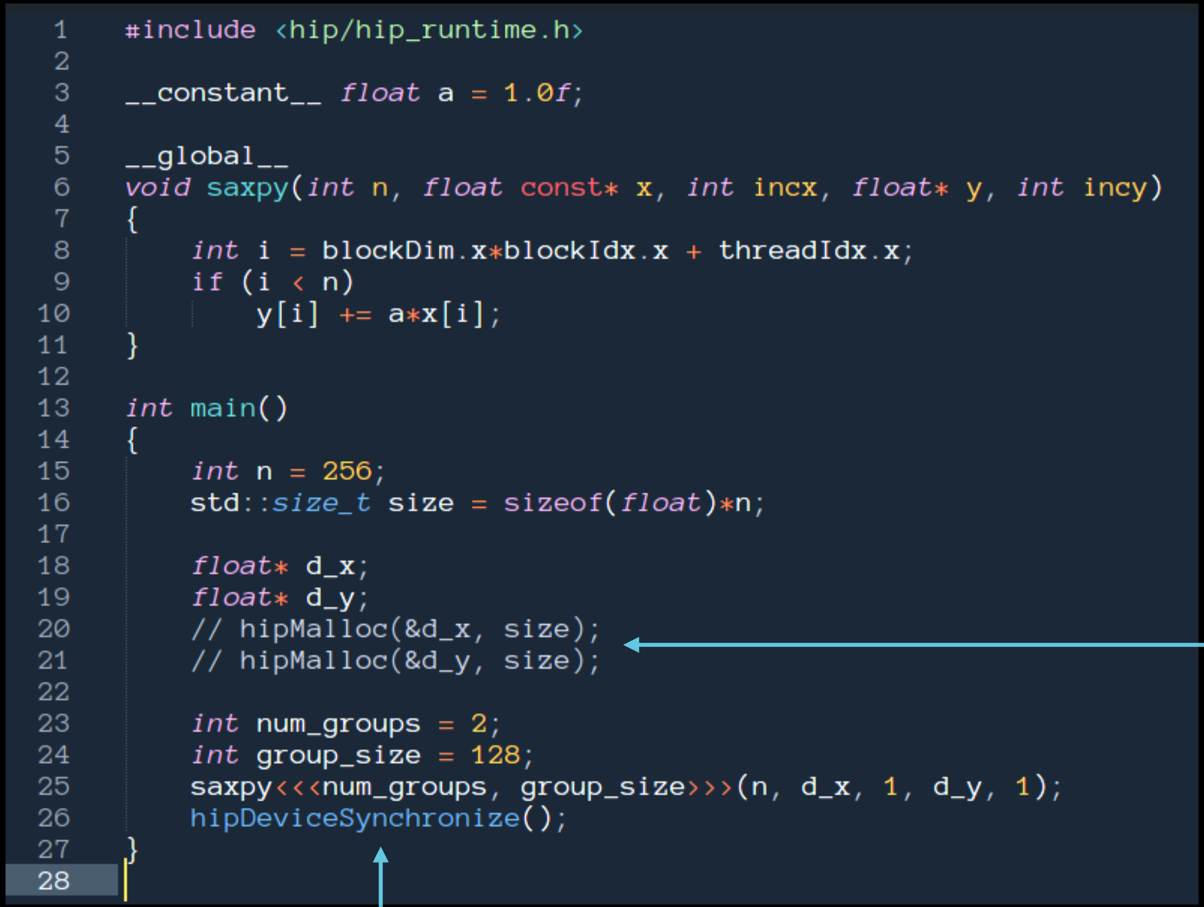
classic saxpy operation
one array index = one work-item

size of arrays = 256

two groups
each 128 work-items



cause a page fault

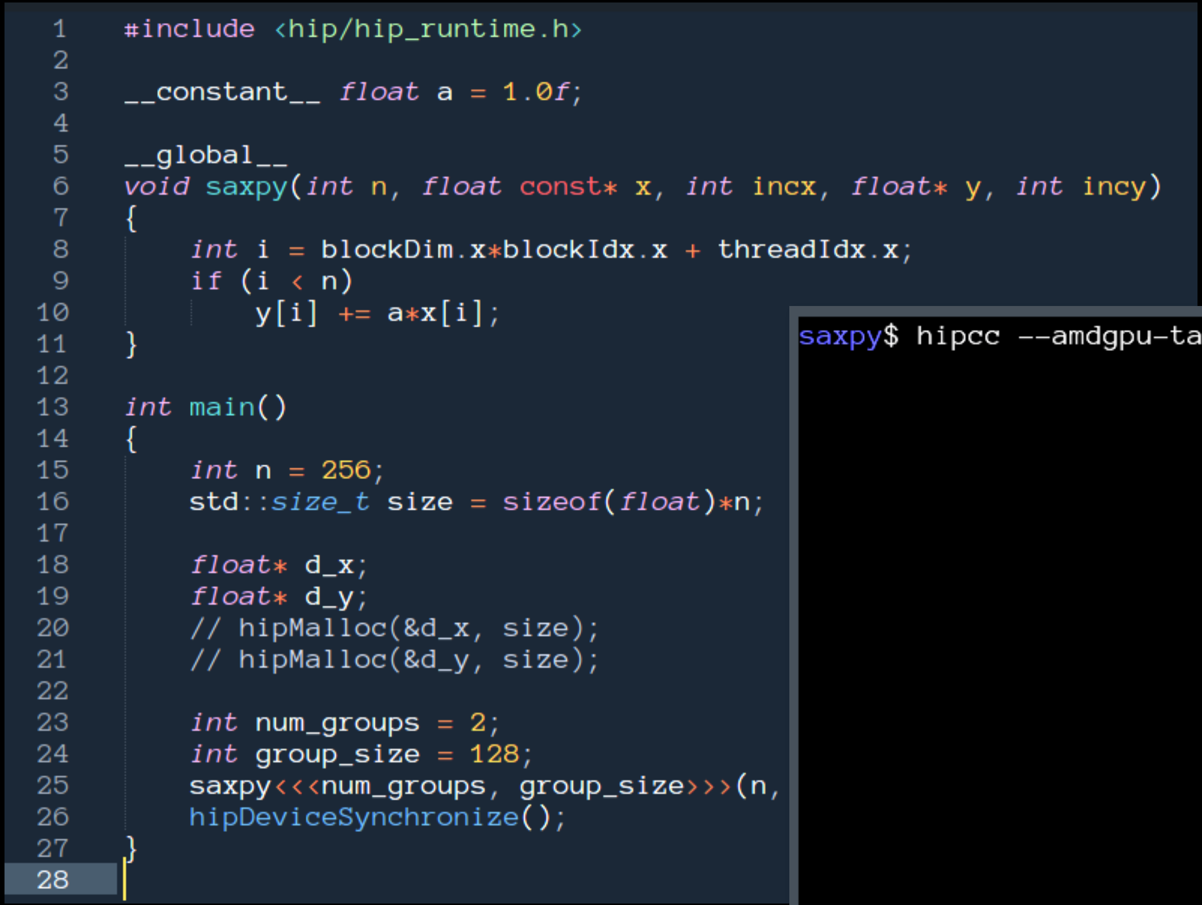


```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n, d_x, 1, d_y, 1);
    hipDeviceSynchronize();
}

```

Break it by commenting out the allocations.
(better to initialize the pointers to nullptr)

It’s important to synchronize before exit.
Otherwise, the CPU thread may quit before the GPU gets a chance to report the error.

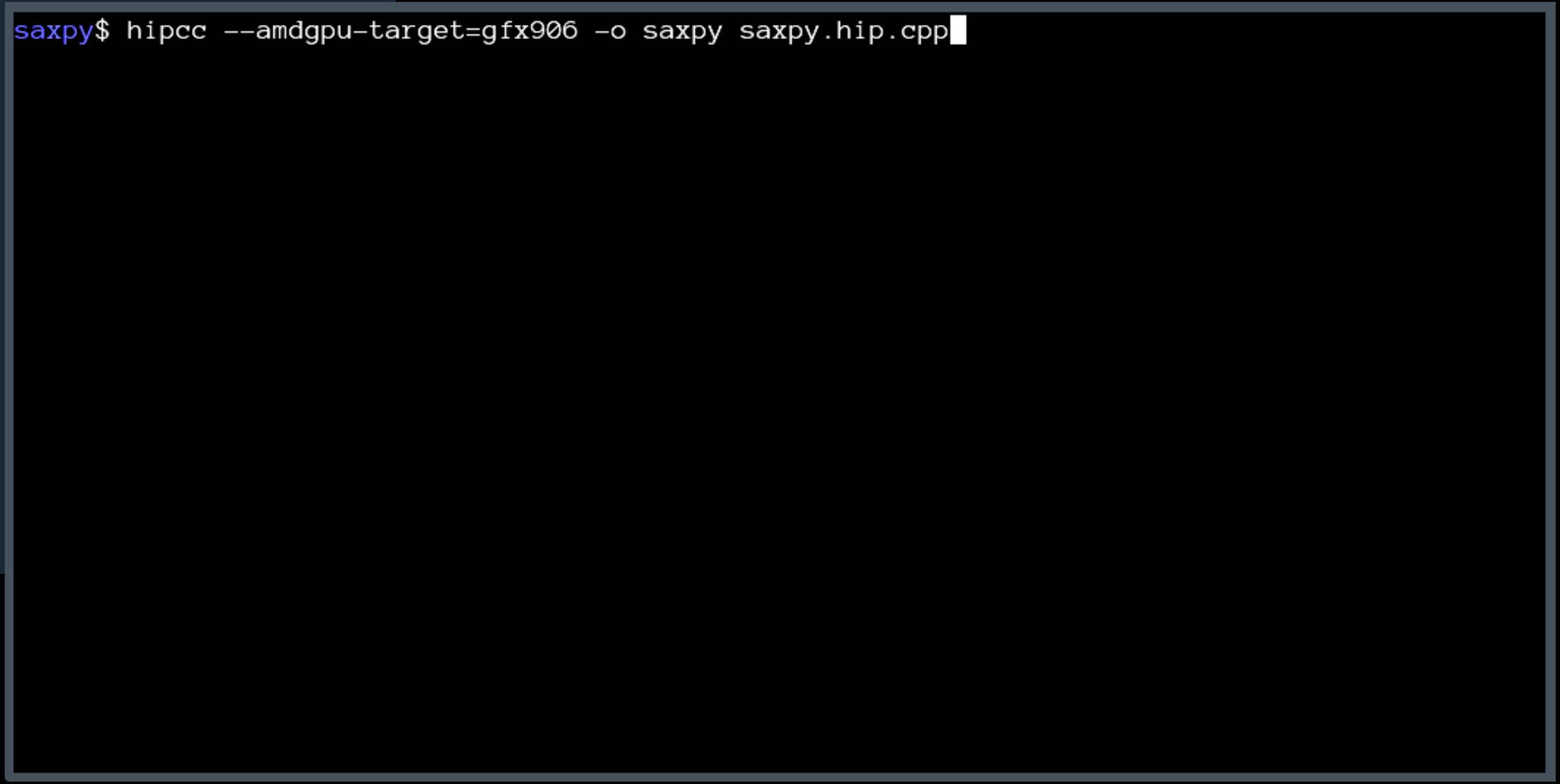
compile with hipcc



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}
```

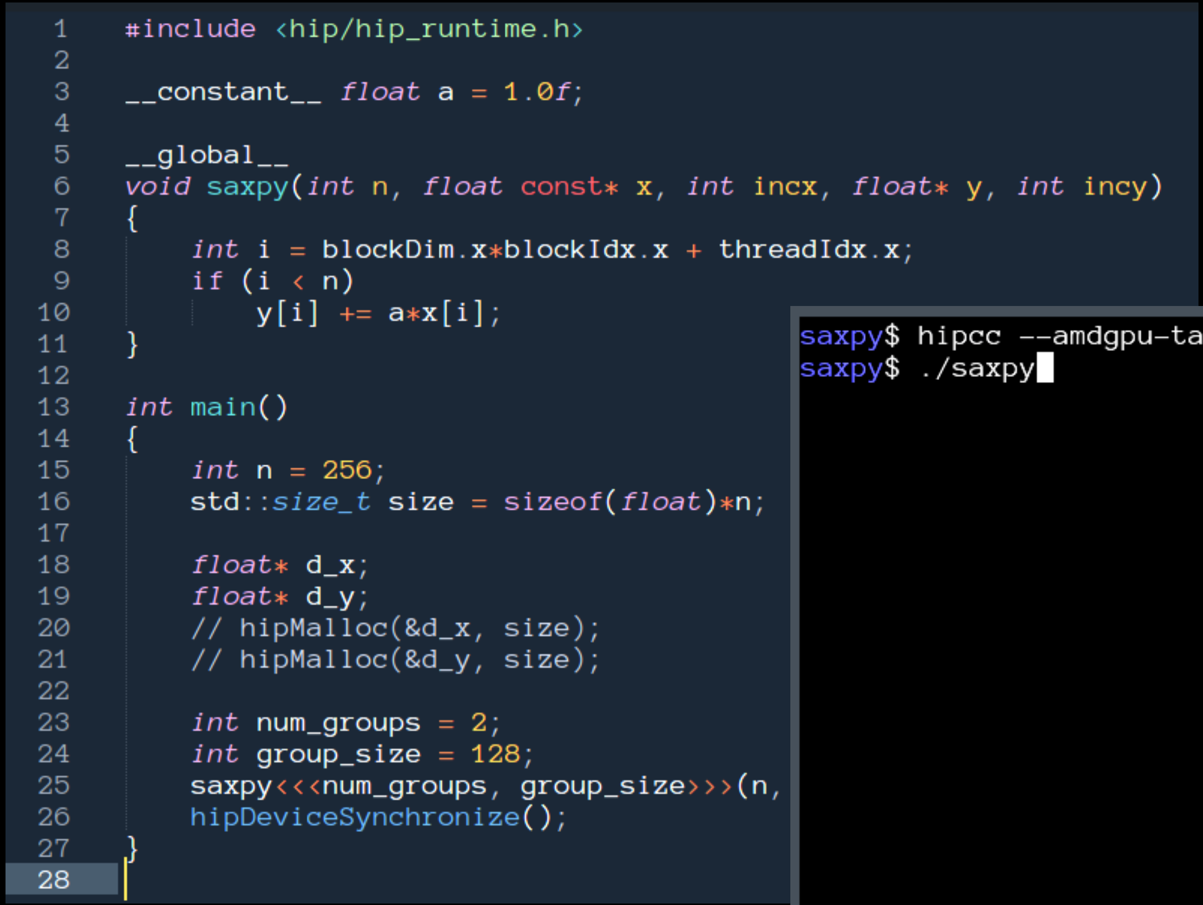
Need be, set the target

- gfx906 – MI50, MI60, Radeon 7
- gfx908 – MI100
- fgx90a – MI200

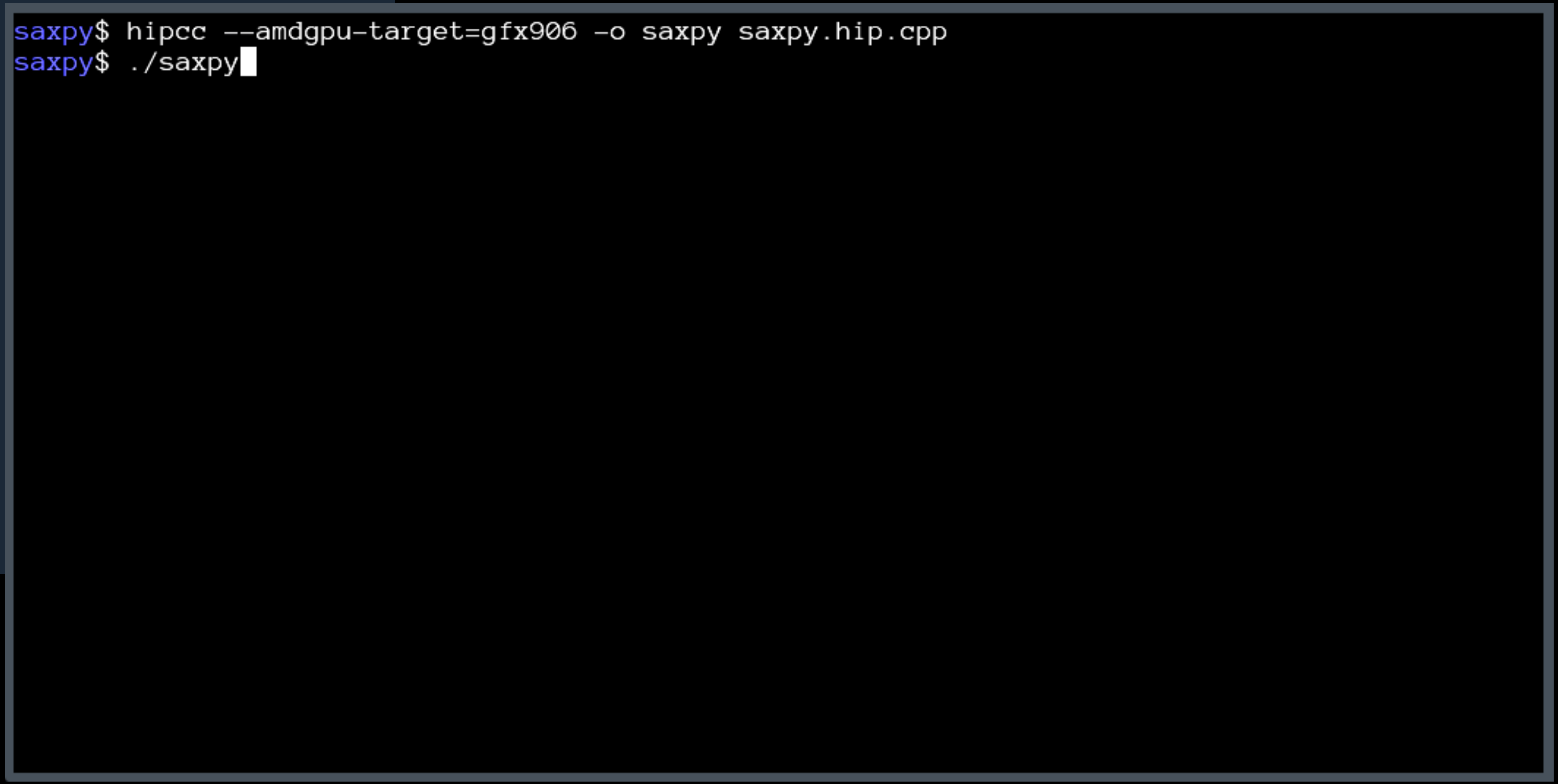


```
saxpy$ hipcc --amdgpu-target=gfx906 -o saxpy saxpy.hip.cpp
```

run



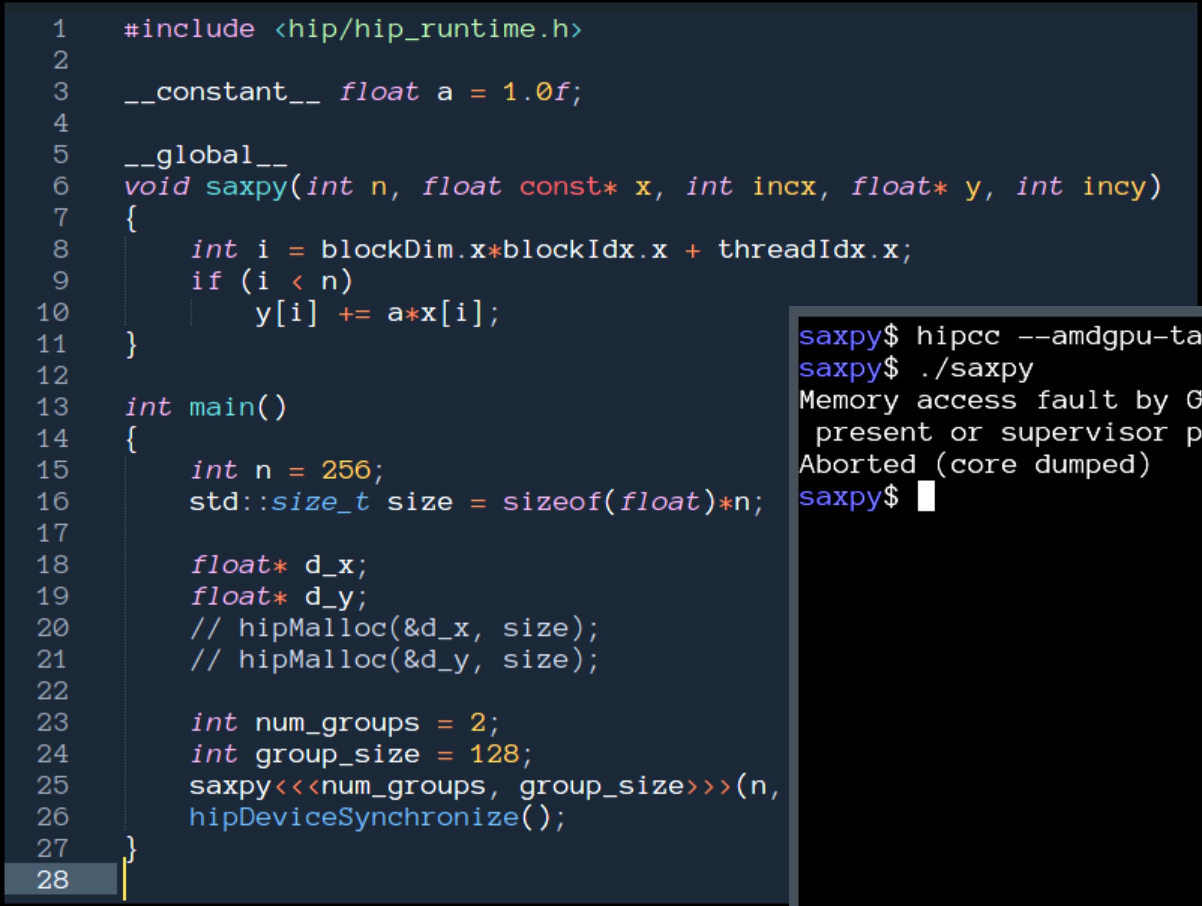
```cpp
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}
```



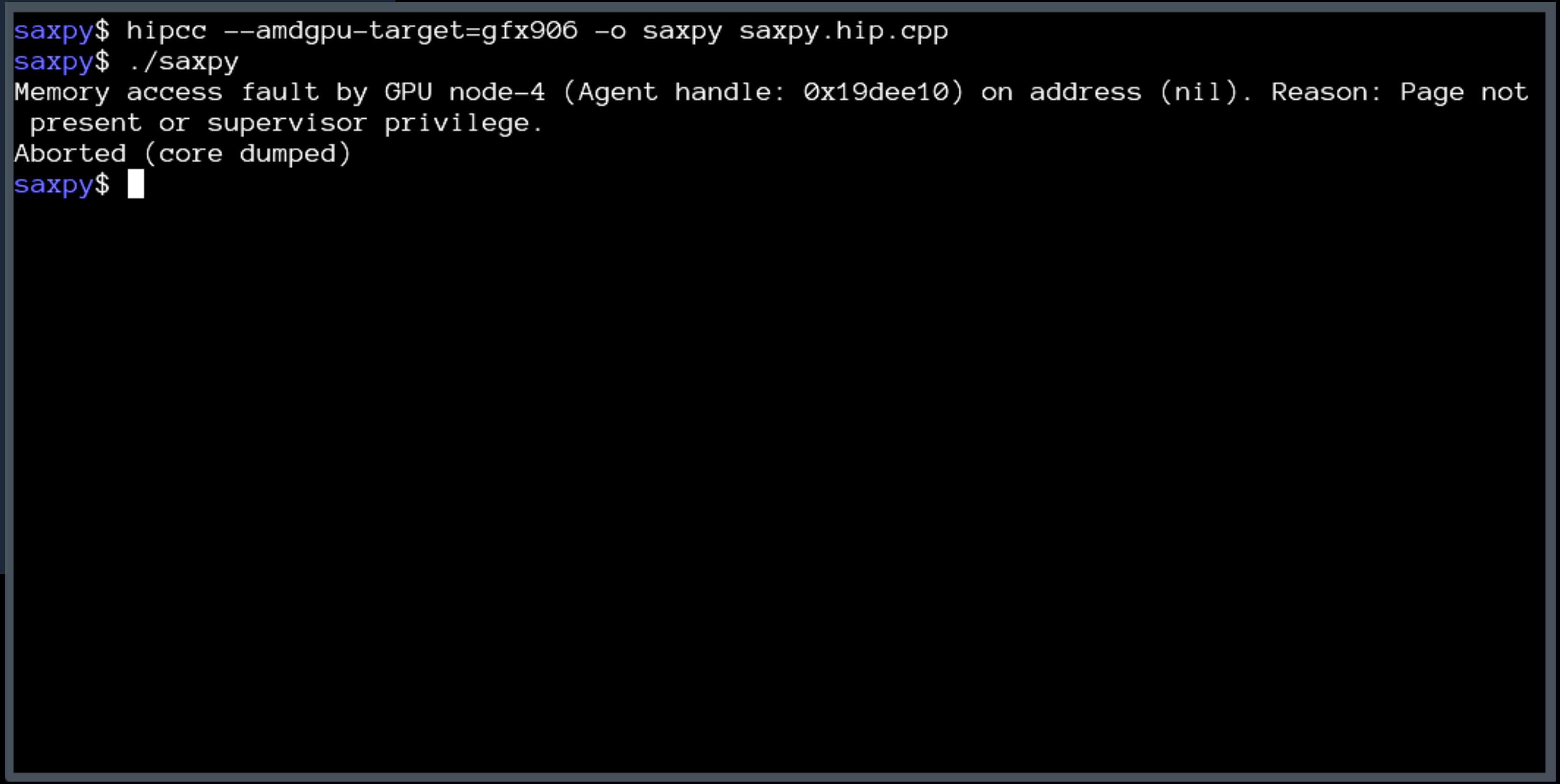
```
saxpy$ hipcc --amdgpu-target=gfx906 -o saxpy saxpy.hip.cpp
saxpy$ ./saxpy
```

AMD
together we advance_

get a page fault

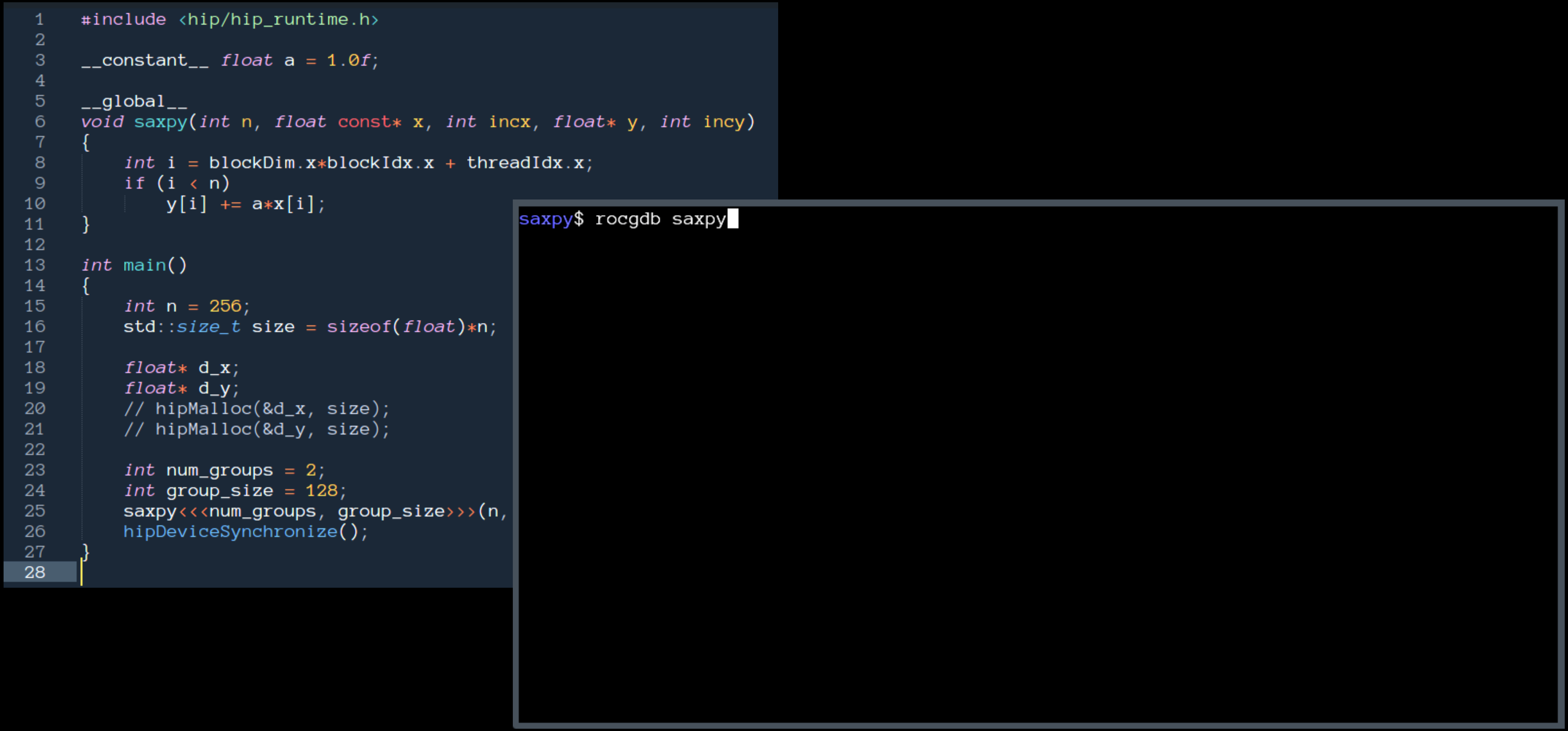


```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}

```



```
saxpy$ hipcc --amdgpu-target=gfx906 -o saxpy saxpy.hip.cpp
saxpy$ ./saxpy
Memory access fault by GPU node-4 (Agent handle: 0x19dee10) on address (nil). Reason: Page not
 present or supervisor privilege.
Aborted (core dumped)
saxpy$
```

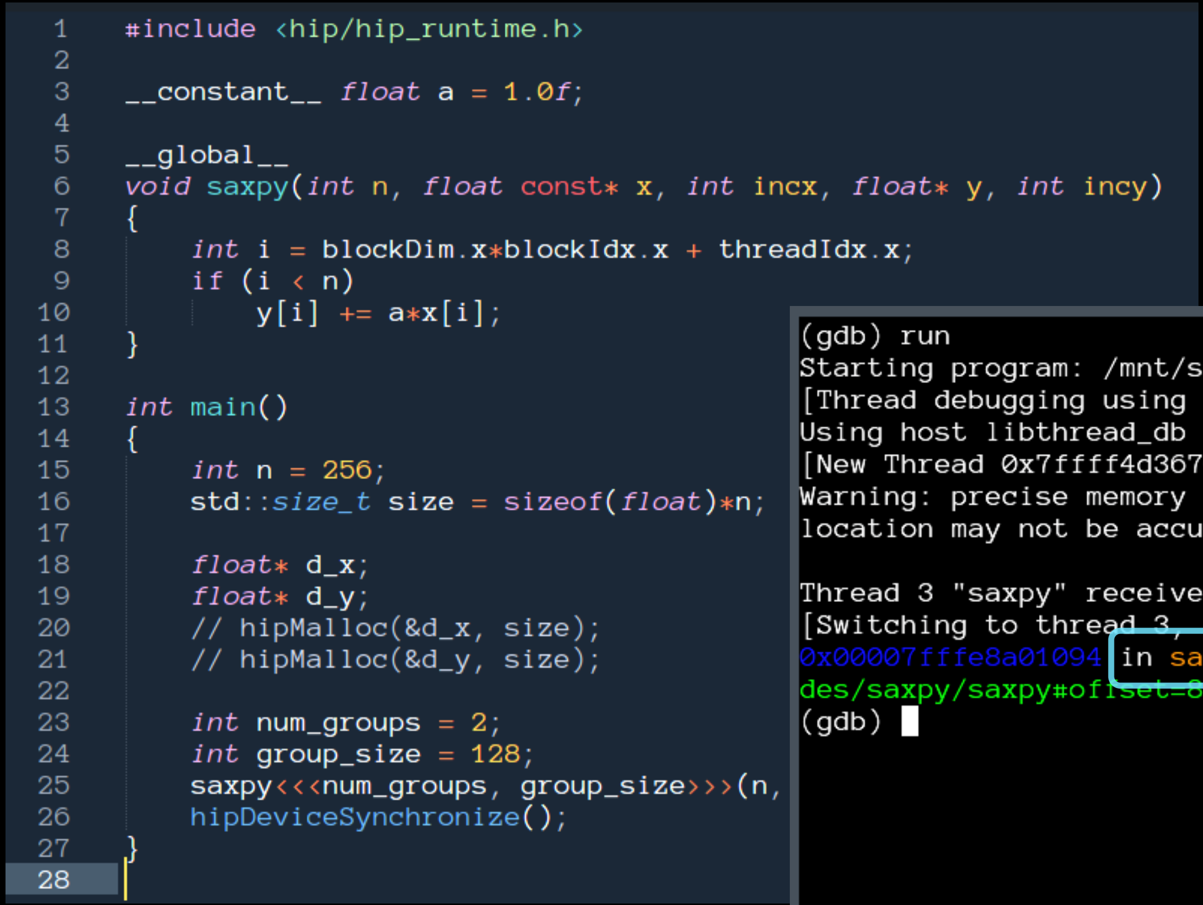
run with rocgdb



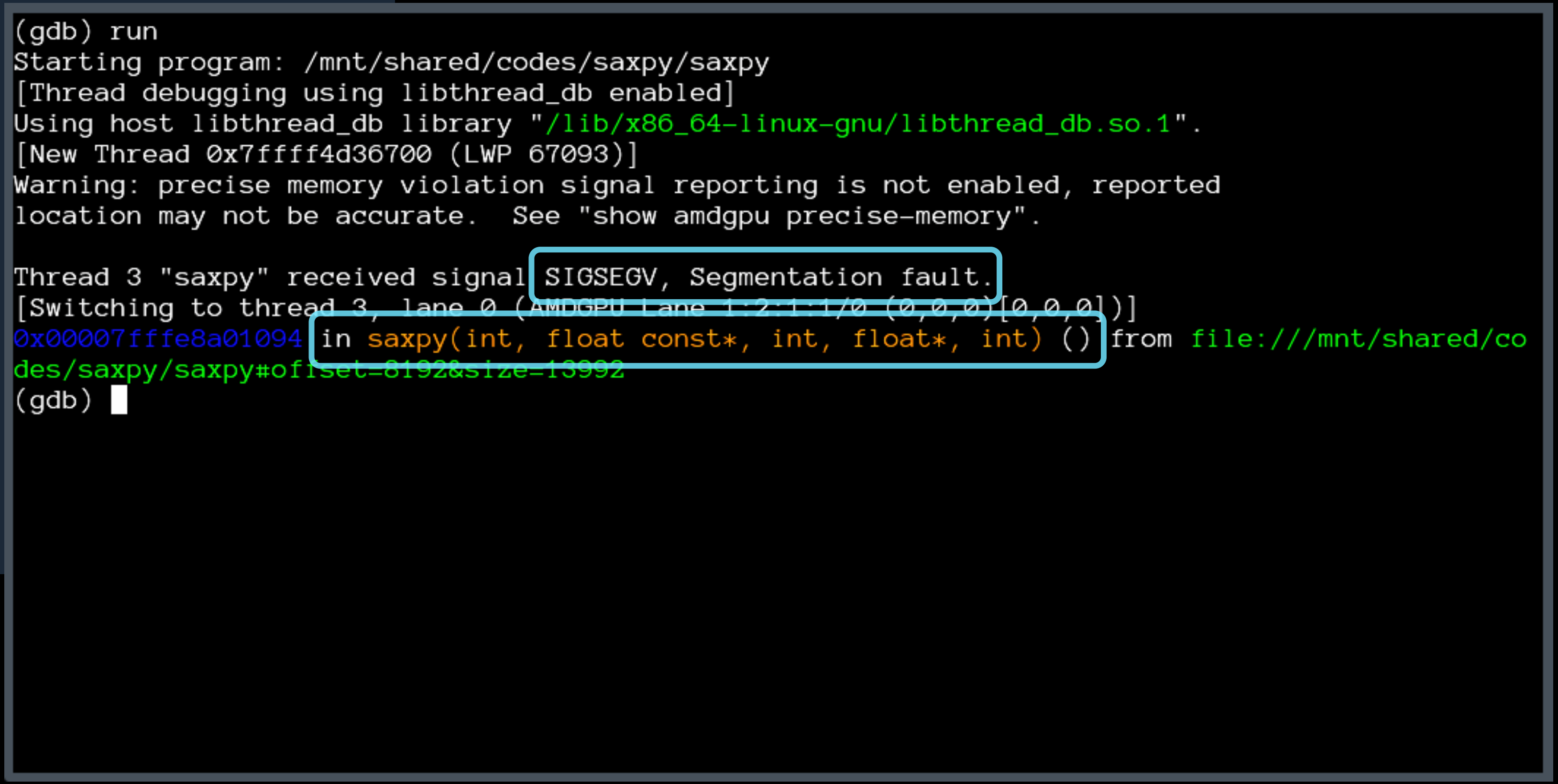
```cpp
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}
```

```
saxpy$ rocgdb saxpy
```

get more info



```cpp
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}

```

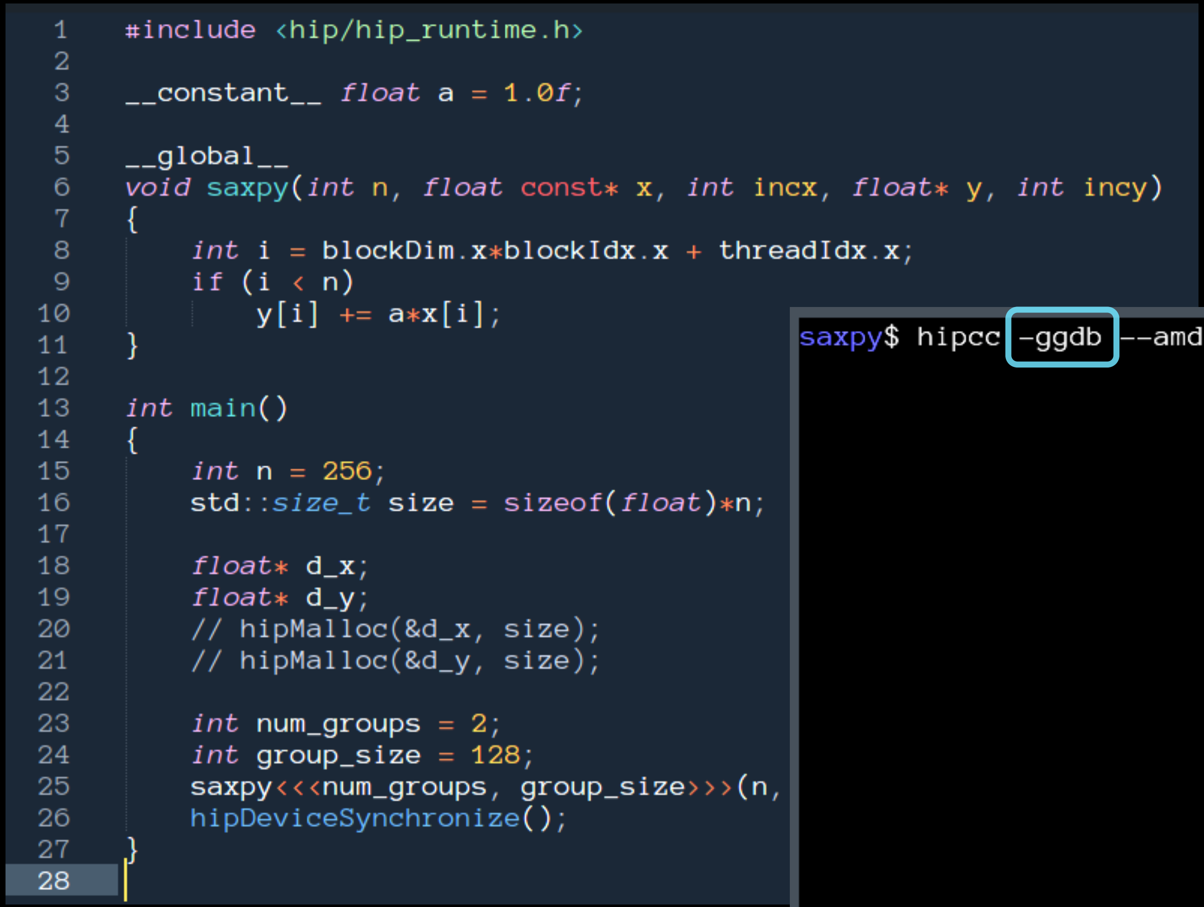


```
(gdb) run
Starting program: /mnt/shared/codes/saxpy/saxpy
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
[New Thread 0x7ffff4d36700 (LWP 67093)]
Warning: precise memory violation signal reporting is not enabled, reported
location may not be accurate.  See "show amdgpu precise-memory".

Thread 3 "saxpy" received signal SIGSEGV, Segmentation fault.
[Switching to thread 3, lane 0 (AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0])]
0x00007fffe8a01094 in saxpy(int, float const*, int, float*, int) () from file:///mnt/shared/co
des/saxpy/saxpy#offset=8192&size=13992
(gdb) 
```

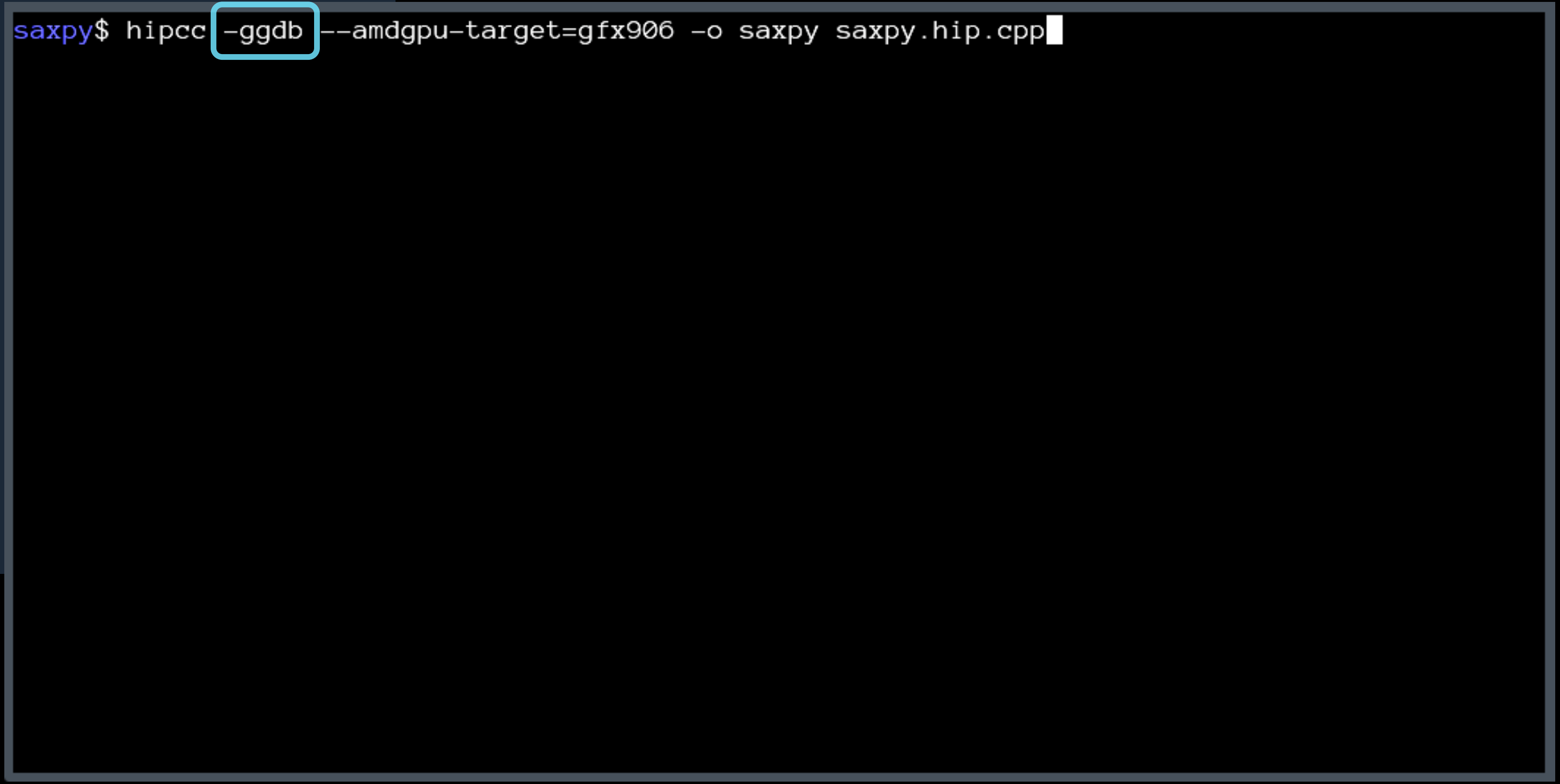
Reports segmentation fault in the saxpy kernel.

AMD
together we advance_

compile with -ggdb

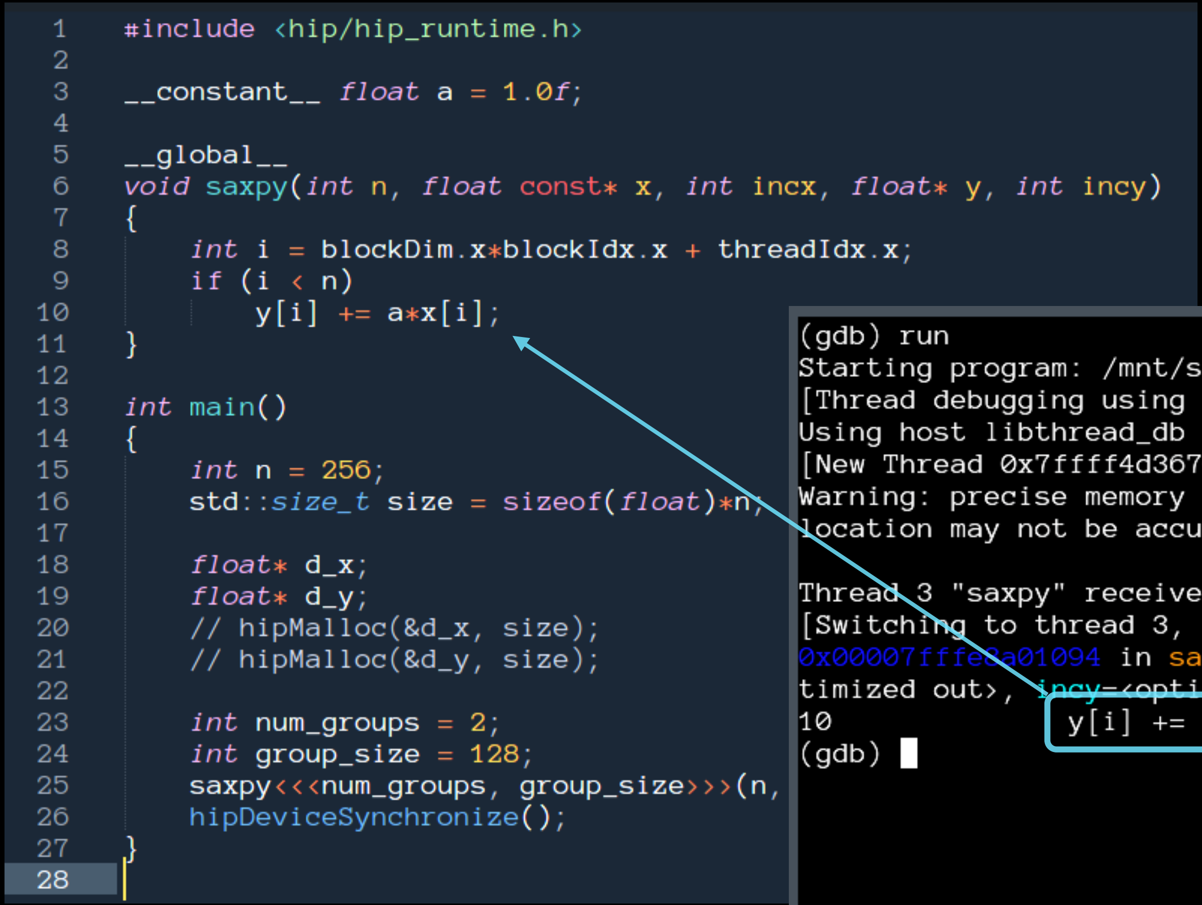


```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}
```

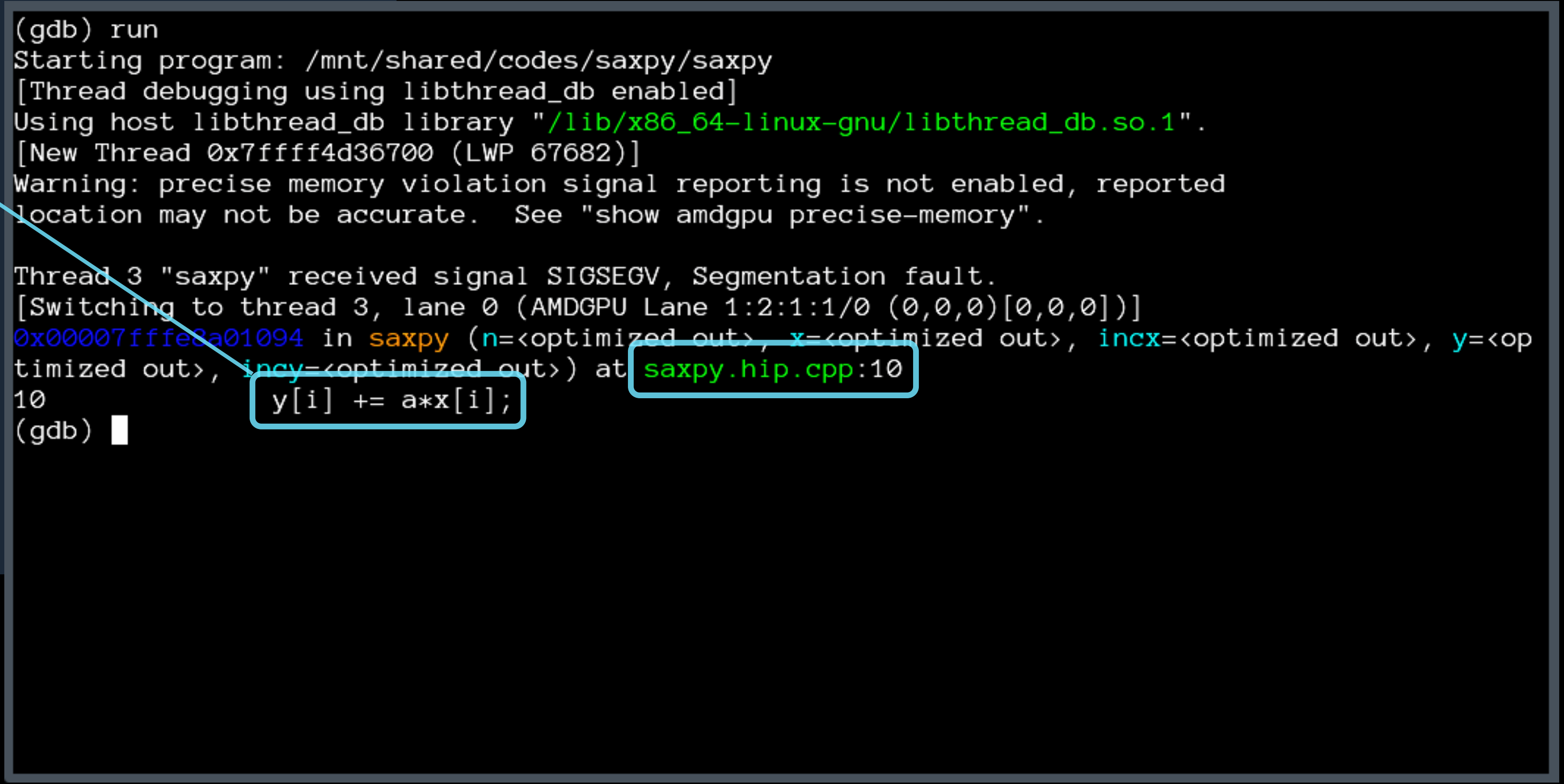


```
saxpy$ hipcc -ggdb --amdgpu-target=gfx906 -o saxpy saxpy.hip.cpp
```

get more details



```
 1  #include <hip/hip_runtime.h>
 2
 3  __constant__ float a = 1.0f;
 4
 5  __global__
 6  void saxpy(int n, float const* x, int incx, float* y, int incy)
 7  {
 8      int i = blockDim.x*blockIdx.x + threadIdx.x;
 9      if (i < n)
10          y[i] += a*x[i];
11  }
12
13  int main()
14  {
15      int n = 256;
16      std::size_t size = sizeof(float)*n;
17
18      float* d_x;
19      float* d_y;
20      // hipMalloc(&d_x, size);
21      // hipMalloc(&d_y, size);
22
23      int num_groups = 2;
24      int group_size = 128;
25      saxpy<<<num_groups, group_size>>>(n,
26      hipDeviceSynchronize();
27  }
28
```



```
(gdb) run
Starting program: /mnt/shared/codes/saxpy/saxpy
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
[New Thread 0x7ffff4d36700 (LWP 67682)]
Warning: precise memory violation signal reporting is not enabled, reported
location may not be accurate.  See "show amdgpu precise-memory".

Thread 3 "saxpy" received signal SIGSEGV, Segmentation fault.
[Switching to thread 3, lane 0 (AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0])]
0x00007fffe8a01094 in saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, y=<op
timized out>, incy=<optimized out>) at saxpy.hip.cpp:10
10          y[i] += a*x[i];
(gdb) 
```

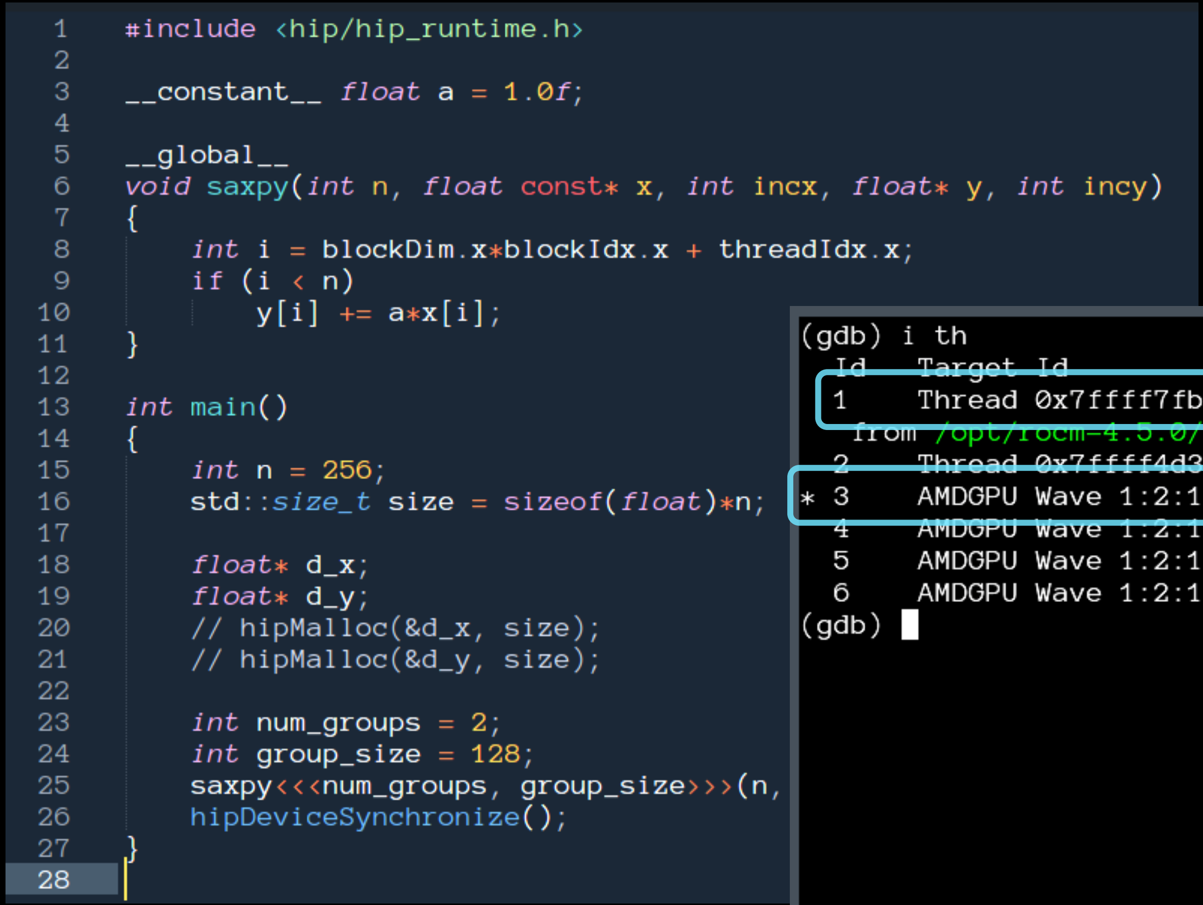
more details

- what kernel
- what file:line

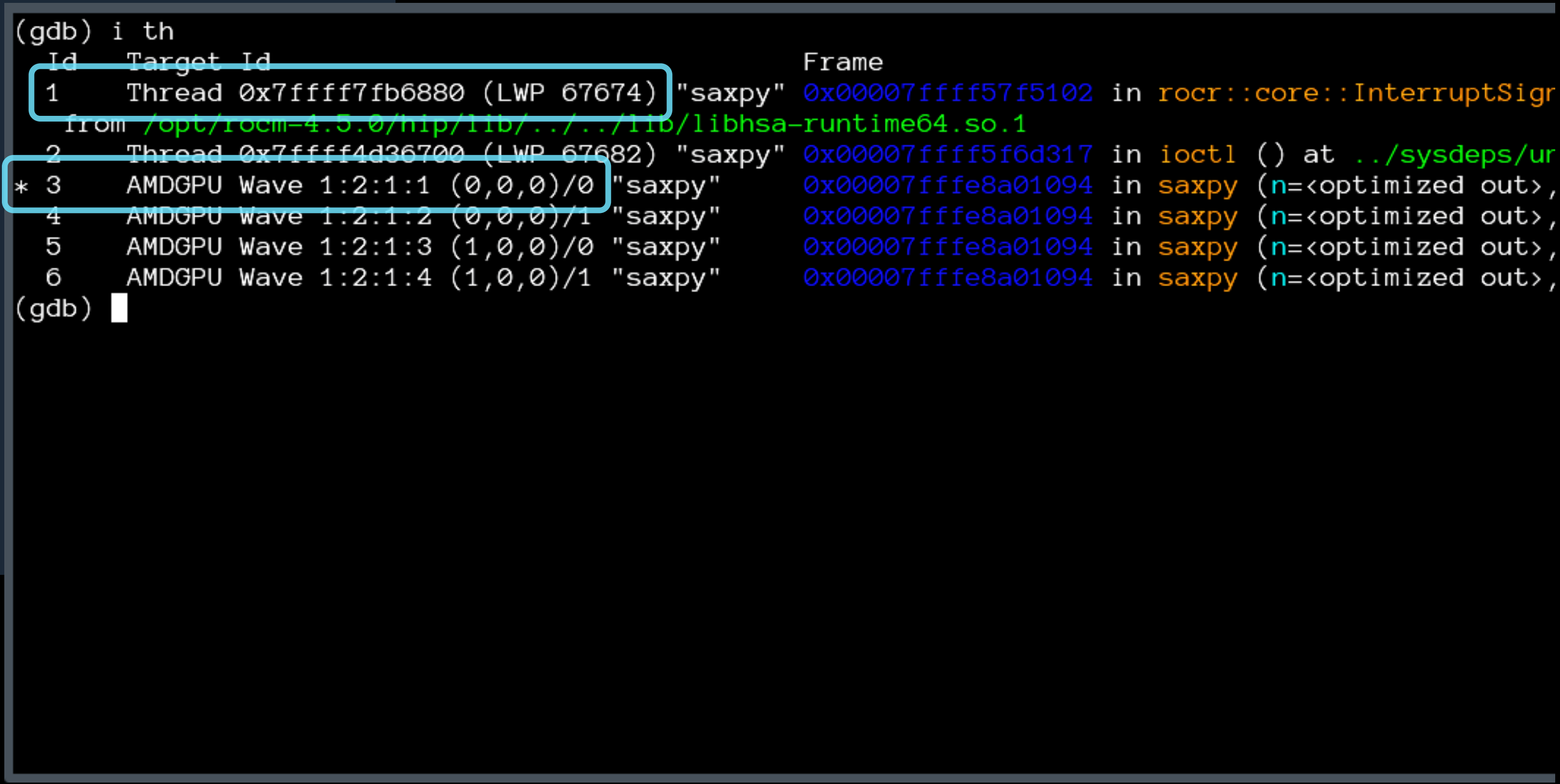
But where's my stack trace?

To get exceptions reported precisely: `set amdgpu precise-memory on`

list threads



```cpp
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}
```

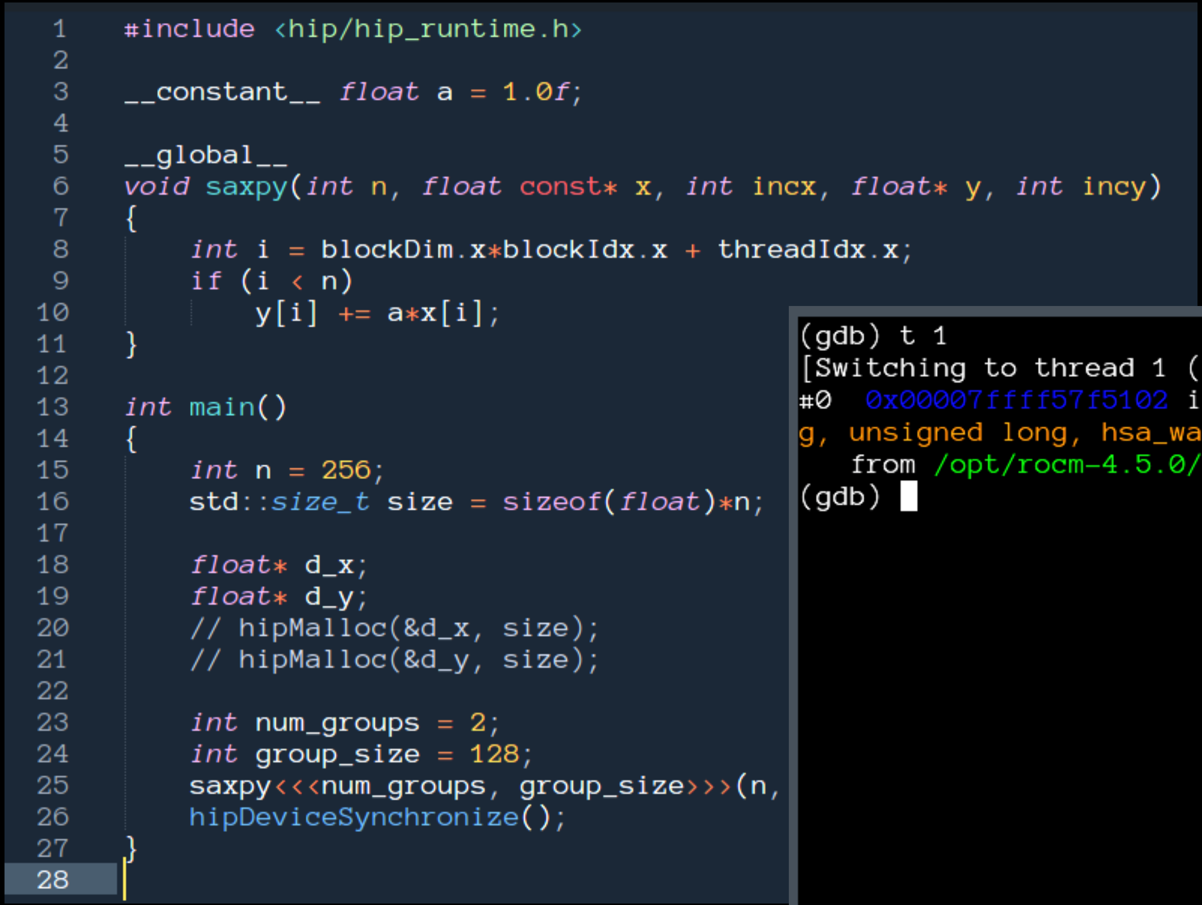


```
(gdb) i th
  Id   Target Id                                  Frame
  1    Thread 0x7ffff7fb6880 (LWP 67674) "saxpy" 0x00007ffff57f5102 in rocr::core::InterruptSigr
   from /opt/rocm-4.5.0/hip/lib/../../lib/libhsa-runtime64.so.1
  2    Thread 0x7ffff4d36700 (LWP 67682) "saxpy" 0x00007ffff5f6d317 in ioctl () at ../sysdeps/ur
* 3    AMDGPU Wave 1:2:1:1 (0,0,0)/0 "saxpy"     0x00007fffe8a01094 in saxpy (n=<optimized out>,
  4    AMDGPU Wave 1:2:1:2 (0,0,0)/1 "saxpy"     0x00007fffe8a01094 in saxpy (n=<optimized out>,
  5    AMDGPU Wave 1:2:1:3 (1,0,0)/0 "saxpy"     0x00007fffe8a01094 in saxpy (n=<optimized out>,
  6    AMDGPU Wave 1:2:1:4 (1,0,0)/1 "saxpy"     0x00007fffe8a01094 in saxpy (n=<optimized out>,
(gdb)
```

What segfaulted is a GPU wave.
It does not have your CPU stack.
List threads to see what's going on.

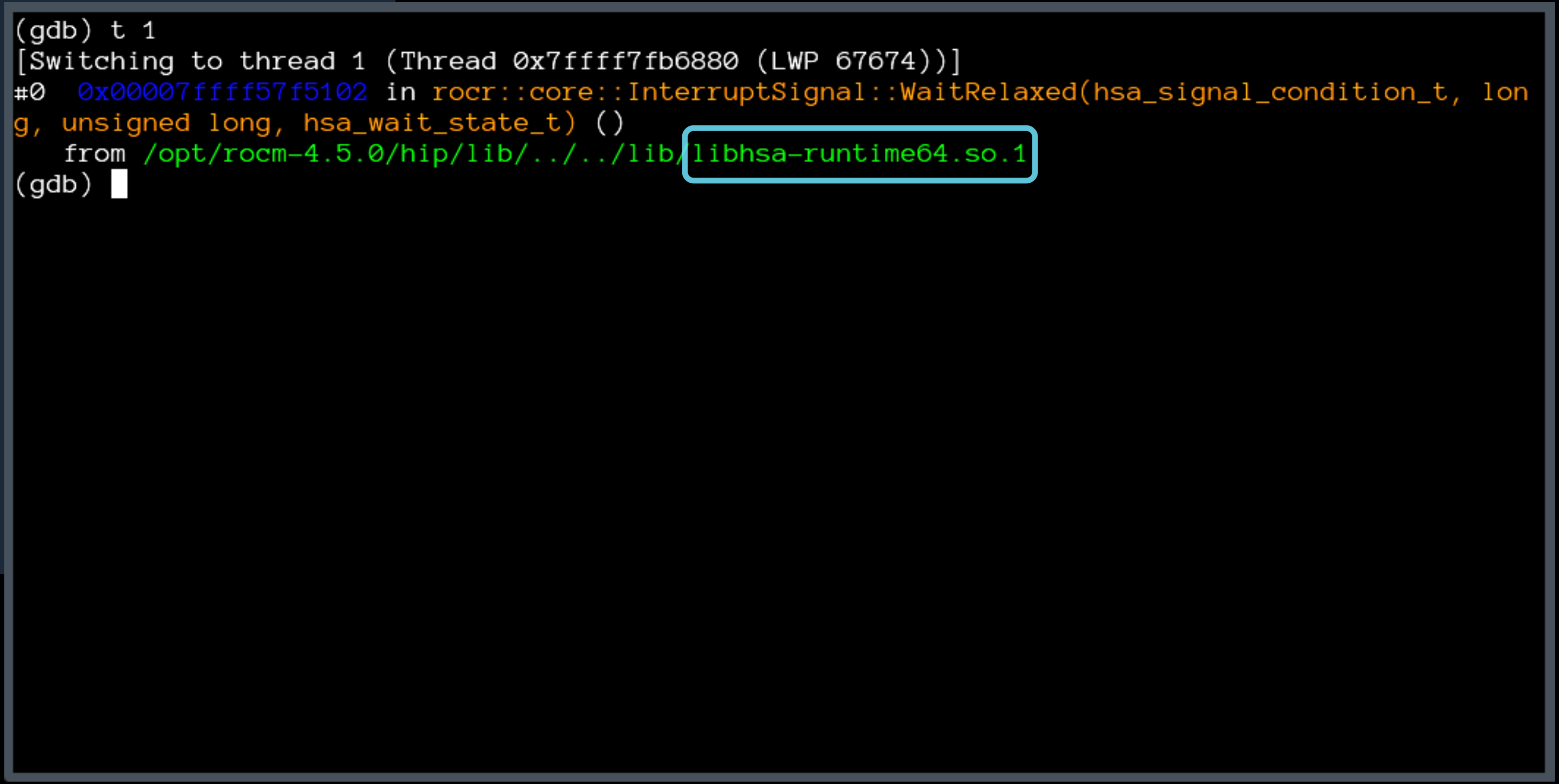
AMD together we advance_

switch to the CPU thread



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    // hipMalloc(&d_x, size);
    // hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n,
    hipDeviceSynchronize();
}

```

t 1
(thread 1)
It’s in the HSA runtime.

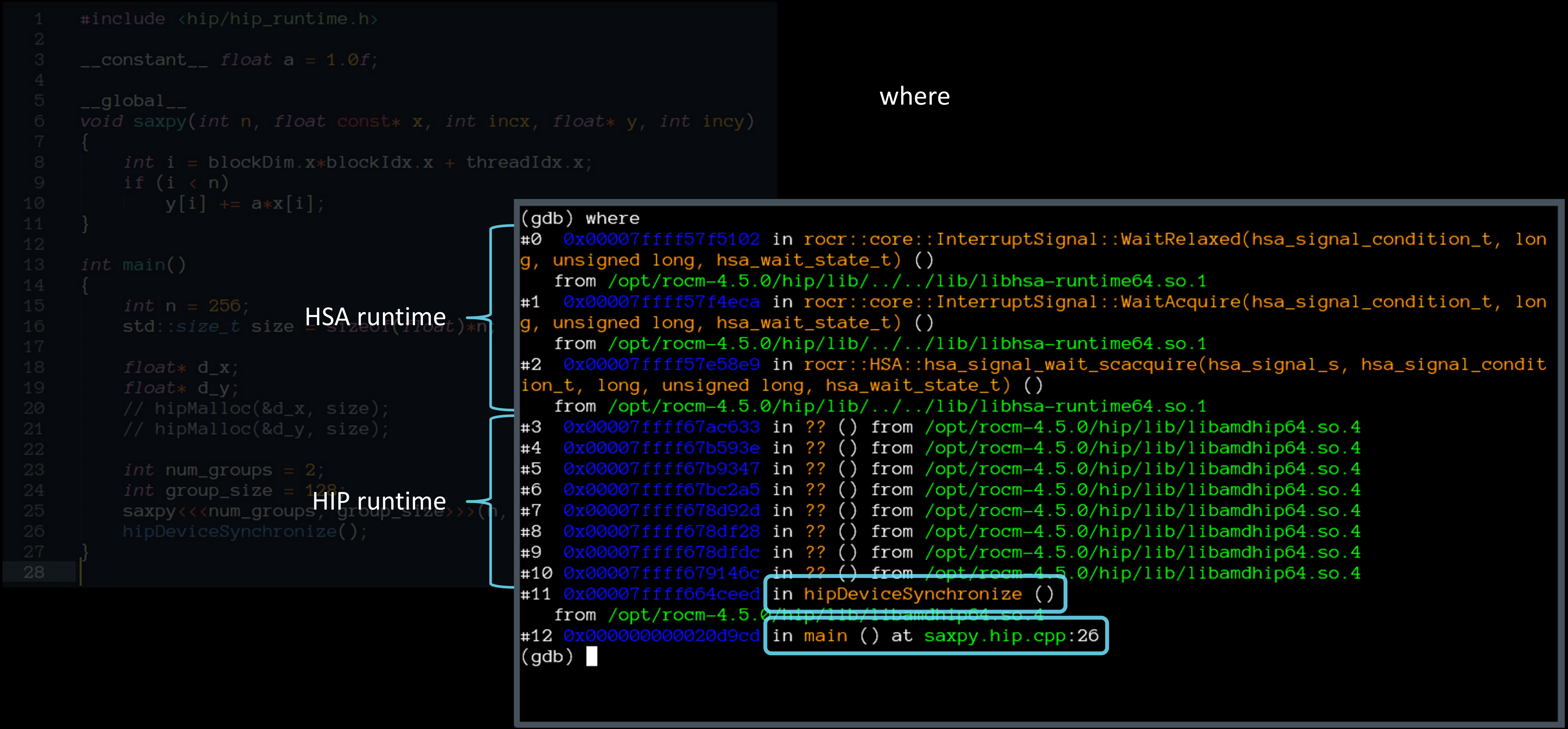


```
(gdb) t 1
[Switching to thread 1 (Thread 0x7ffff7fb6880 (LWP 67674))]
#0  0x00007ffff57f5102 in rocr::core::InterruptSignal::WaitRelaxed(hsa_signal_condition_t, lon
g, unsigned long, hsa_wait_state_t) ()
   from /opt/rocm-4.5.0/hip/lib/../../lib/libhsa-runtime64.so.1
(gdb) 
```

But how did it get there?

see the stack trace of the CPU thread

where



```
 1  #include <hip/hip_runtime.h>
 2
 3  __constant__ float a = 1.0f;
 4
 5  __global__
 6  void saxpy(int n, float const* x, int incx, float* y, int incy)
 7  {
 8      int i = blockDim.x*blockIdx.x + threadIdx.x;
 9      if (i < n)
10          y[i] += a*x[i];
11  }
12
13  int main()
14  {
15      int n = 256;
16      std::size_t size = sizeof(float)*n;
17
18      float* d_x;
19      float* d_y;
20      // hipMalloc(&d_x, size);
21      // hipMalloc(&d_y, size);
22
23      int num_groups = 2;
24      int group_size = 128;
25      saxpy<<<num_groups, group_size>>>(n,
26      hipDeviceSynchronize();
27  }
28
```

```
(gdb) where
#0  0x00007ffff57f5102 in rocr::core::InterruptSignal::WaitRelaxed(hsa_signal_condition_t, lon
g, unsigned long, hsa_wait_state_t) ()
   from /opt/rocm-4.5.0/hip/lib/../../lib/libhsa-runtime64.so.1
#1  0x00007ffff57f4eca in rocr::core::InterruptSignal::WaitAcquire(hsa_signal_condition_t, lon
g, unsigned long, hsa_wait_state_t) ()
   from /opt/rocm-4.5.0/hip/lib/../../lib/libhsa-runtime64.so.1
#2  0x00007ffff57e58e9 in rocr::HSA::hsa_signal_wait_scacquire(hsa_signal_s, hsa_signal_condit
ion_t, long, unsigned long, hsa_wait_state_t) ()
   from /opt/rocm-4.5.0/hip/lib/../../lib/libhsa-runtime64.so.1
#3  0x00007ffff67ac633 in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#4  0x00007ffff67b593e in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#5  0x00007ffff67b9347 in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#6  0x00007ffff67bc2a5 in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#7  0x00007ffff678d92d in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#8  0x00007ffff678df28 in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#9  0x00007ffff678dfdc in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#10 0x00007ffff679146c in ?? () from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#11 0x00007ffff664ceed in hipDeviceSynchronize ()
   from /opt/rocm-4.5.0/hip/lib/libamdhip64.so.4
#12 0x00000000020d9cd in main () at saxpy.hip.cpp:26
(gdb)
```

AMD
together we advance_

quick tip

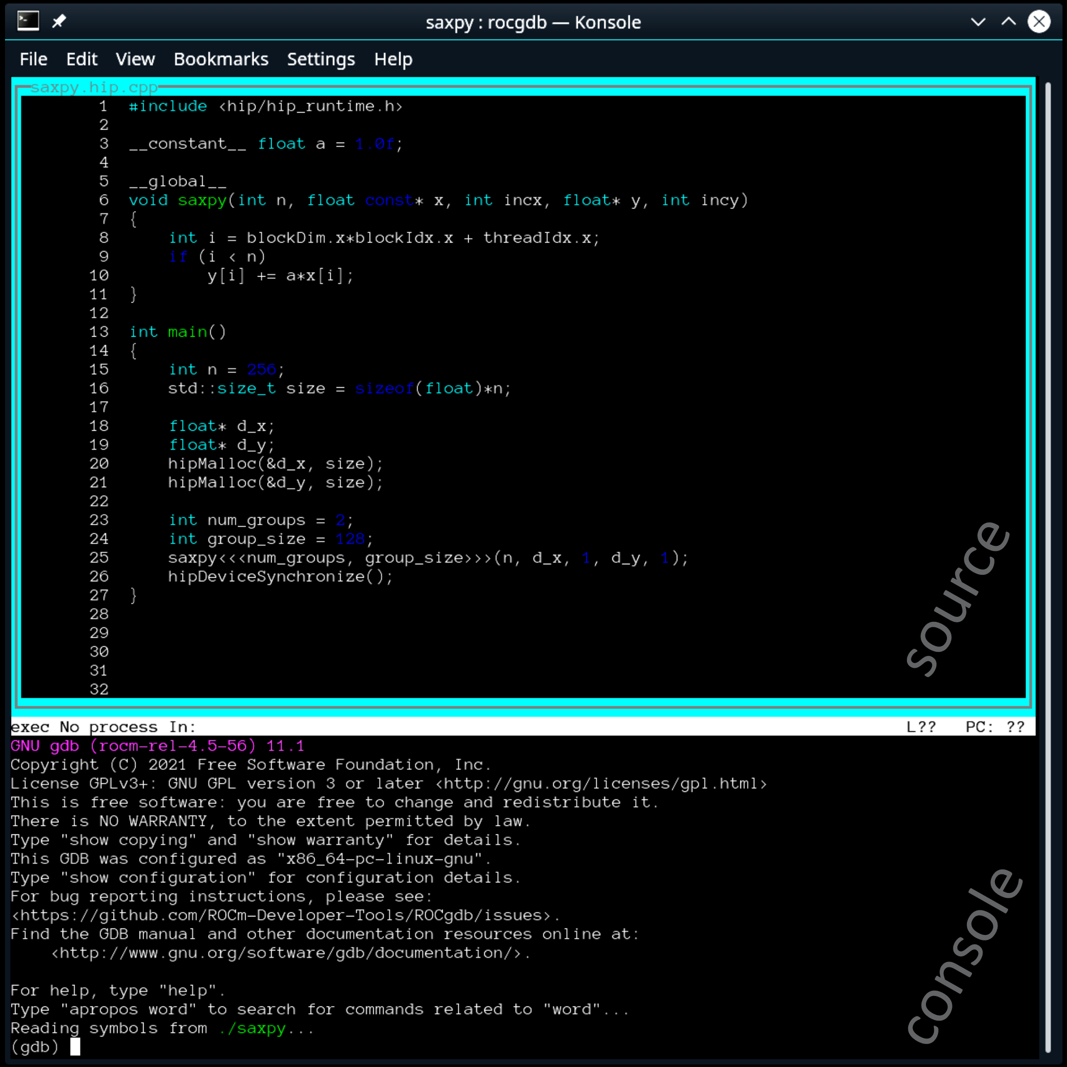
LUMI and Frontier CPUs have 64 cores / 128 threads.

If you’re debugging an app with OpenMP® threading
and OMP_NUM_THREADS is not set
you will see 128 CPU threads in rocgdb.

Set OMP_NUM_THREADS=1 when debugging GPU codes.

"GUIs"

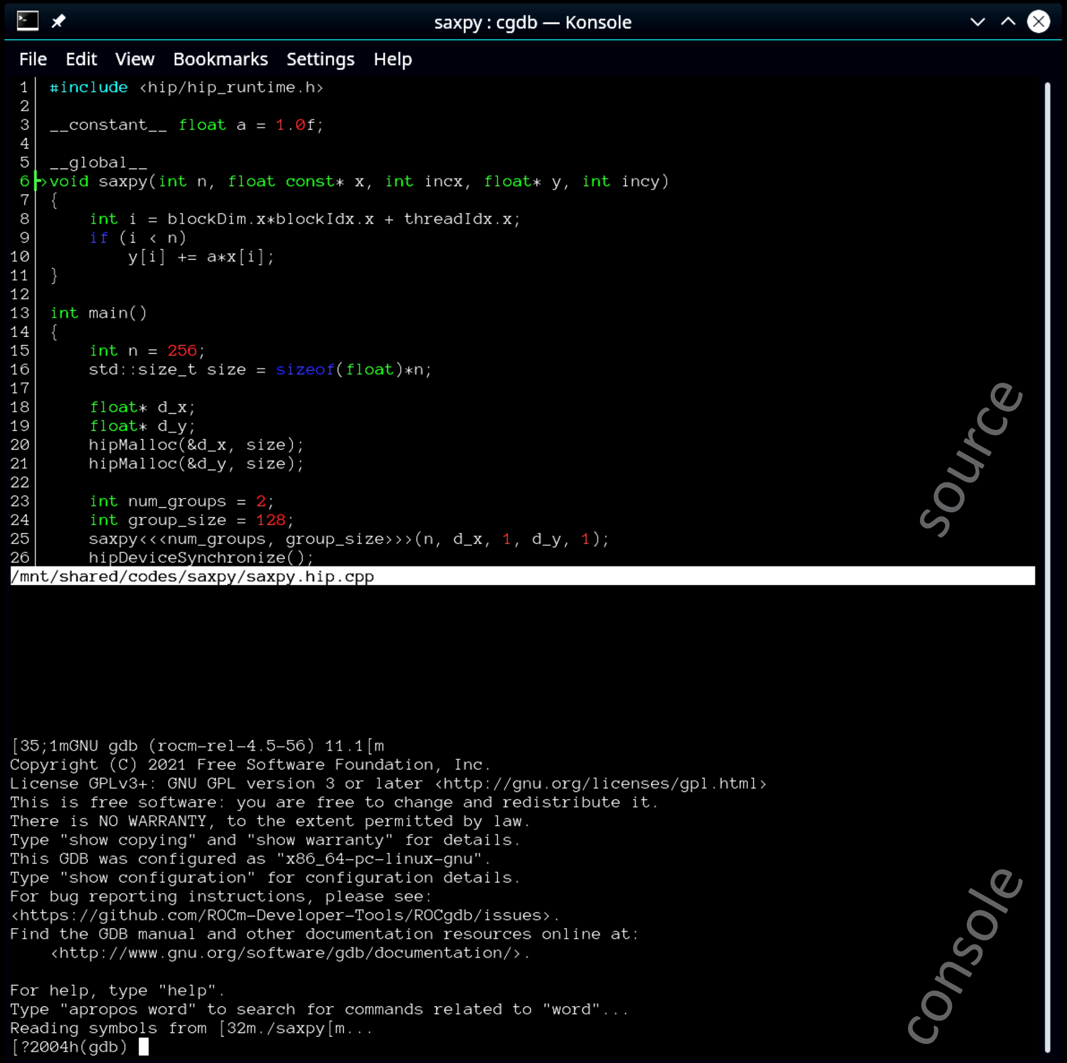
rocgdb -tui saxpy



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n, d_x, 1, d_y, 1);
    hipDeviceSynchronize();
}
```

```
exec No process In:                                                           L??   PC: ??
GNU gdb (rocm-rel-4.5-56) 11.1
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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://github.com/ROCm-Developer-Tools/ROCgdb/issues>.
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 ./saxpy...
(gdb)
```

cgdb -d rocgdb saxpy



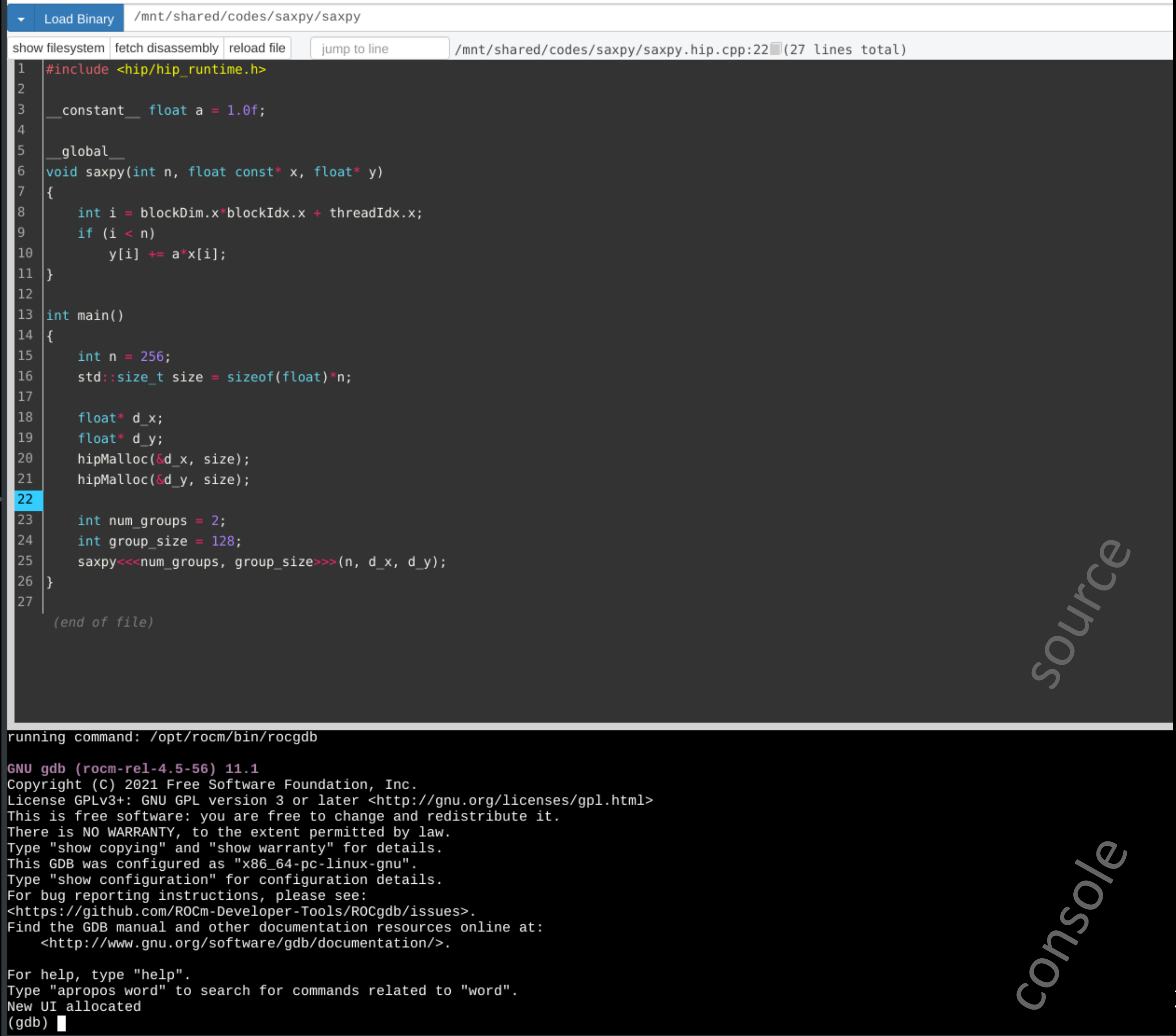
```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, int incx, float* y, int incy)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
    saxpy<<<num_groups, group_size>>>(n, d_x, 1, d_y, 1);
    hipDeviceSynchronize();
/mnt/shared/codes/saxpy/saxpy.hip.cpp
```

```
[35;1mGNU gdb (rocm-rel-4.5-56) 11.1[m
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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://github.com/ROCm-Developer-Tools/ROCgdb/issues>.
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 [32m./saxpy[m...
[?2004h(gdb)
```

AMD
together we advance_

rocgdb + gdbgui

breakpoint in CPU code



```
/mnt/shared/codes/saxpy/saxpy
/mnt/shared/codes/saxpy/saxpy.hip.cpp:22 (27 lines total)
```

```
1  #include <hip/hip_runtime.h>
2
3  __constant__ float a = 1.0f;
4
5  __global__
6  void saxpy(int n, float const* x, float* y)
7  {
8      int i = blockDim.x*blockIdx.x + threadIdx.x;
9      if (i < n)
10         y[i] += a*x[i];
11 }
12
13 int main()
14 {
15     int n = 256;
16     std::size_t size = sizeof(float)*n;
17
18     float* d_x;
19     float* d_y;
20     hipMalloc(&d_x, size);
21     hipMalloc(&d_y, size);
22
23     int num_groups = 2;
24     int group_size = 128;
25     saxpy<<<num_groups, group_size>>>(n, d_x, d_y);
26 }
27
   (end of file)
```

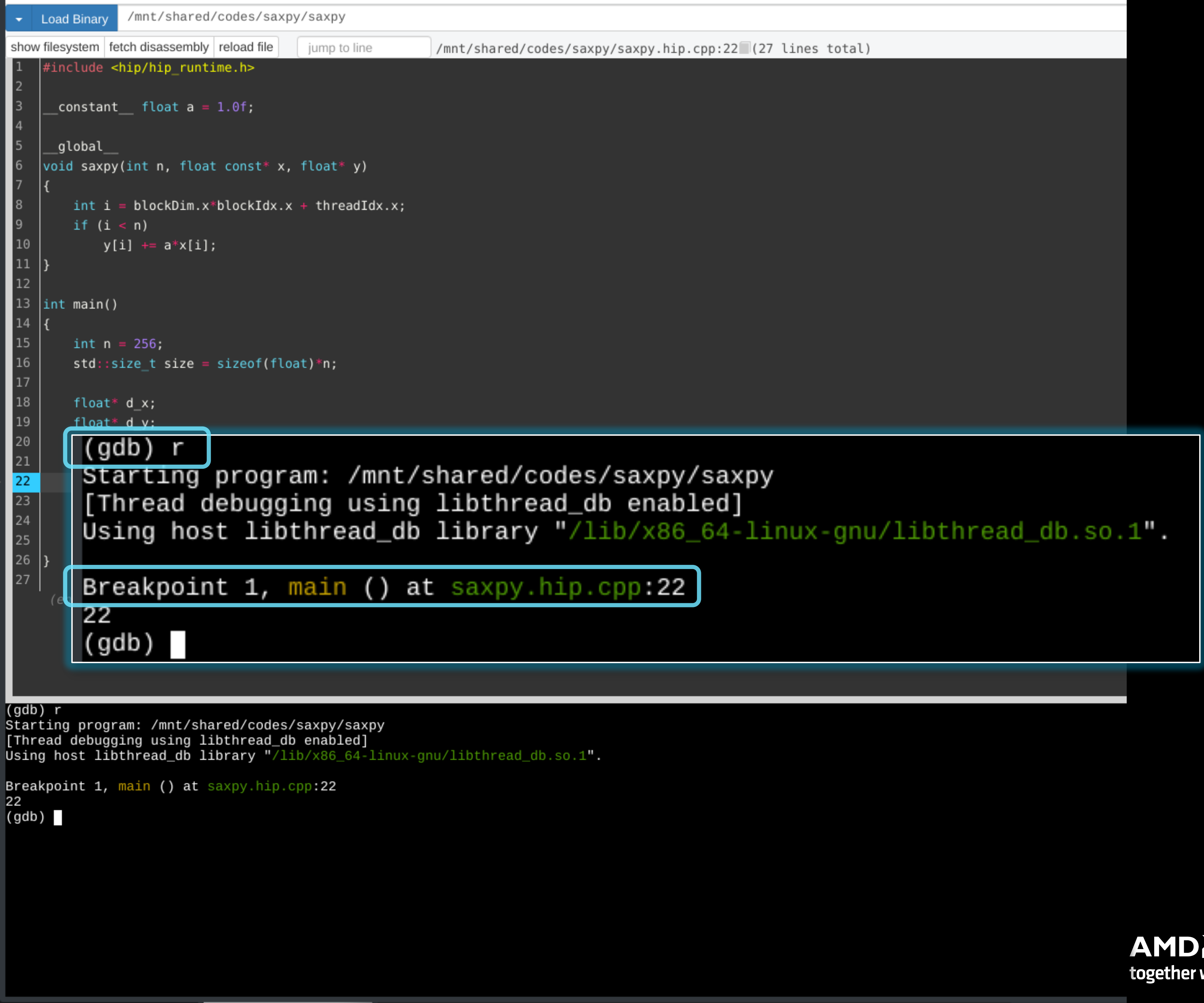
```
running command: /opt/rocm/bin/rocgdb

GNU gdb (rocm-rel-4.5-56) 11.1
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-pc-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://github.com/ROCm-Developer-Tools/ROCgdb/issues>.
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".
New UI allocated
(gdb)
```

[Public]

hit CPU breakpoint

r
(run)

hit the breakpoint in CPU code



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | jump to line | /mnt/shared/codes/saxpy/saxpy.hip.cpp:22 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
```

```
(gdb) r
Starting program: /mnt/shared/codes/saxpy/saxpy
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at saxpy.hip.cpp:22
22
(gdb)
```

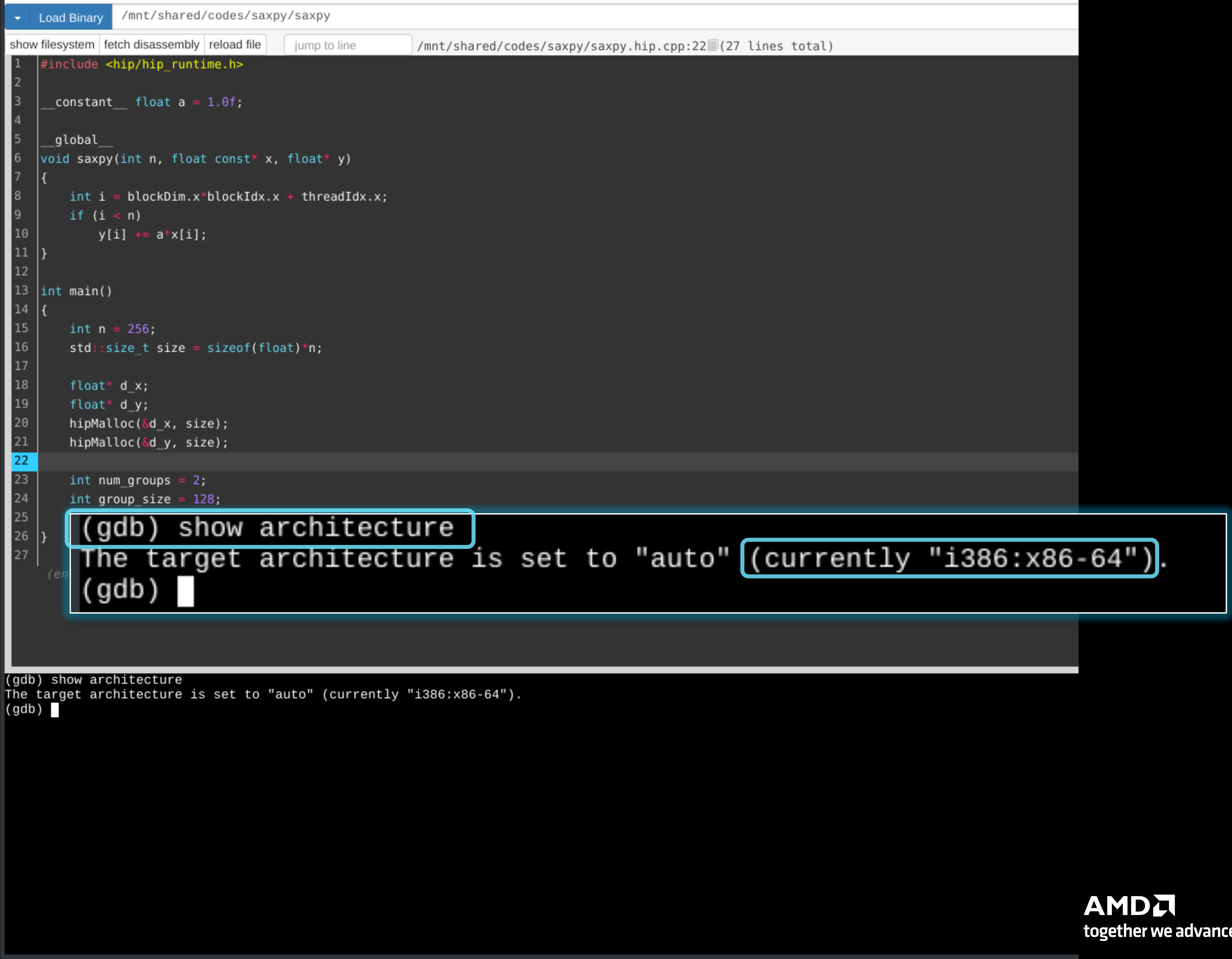
```
(gdb) r
Starting program: /mnt/shared/codes/saxpy/saxpy
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, main () at saxpy.hip.cpp:22
22
(gdb)
```

AMD
together we advance_

show CPU arch

show architecture

- ➢ x86-64



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | jump to line | /mnt/shared/codes/saxpy/saxpy.hip.cpp:22 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;

}
```

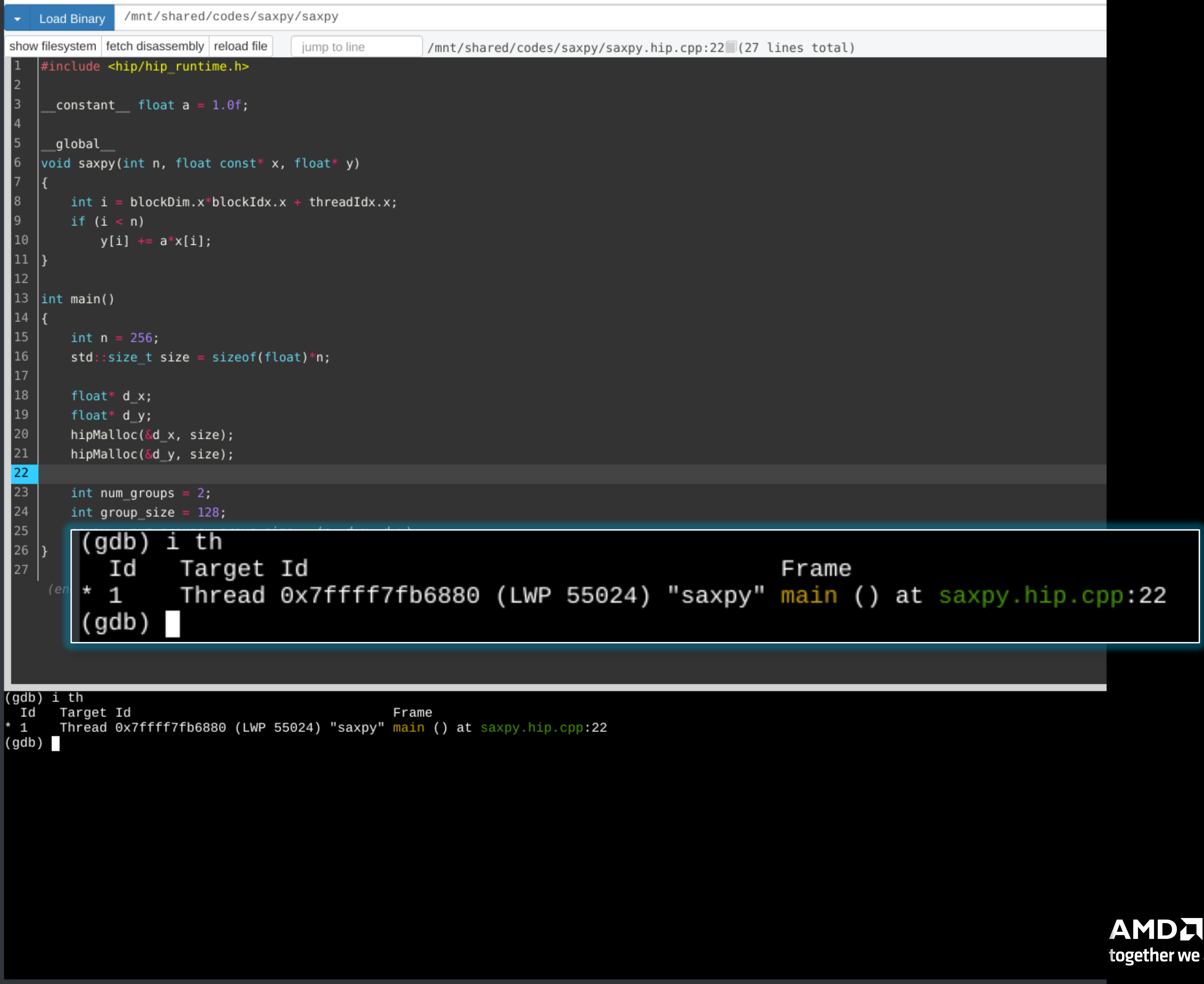
```
(gdb) show architecture
The target architecture is set to "auto" (currently "i386:x86-64").
(gdb)
```

```
(gdb) show architecture
The target architecture is set to "auto" (currently "i386:x86-64").
(gdb)
```

AMD
together we advance_

show CPU thread

i th
(info threads)
➢ one CPU thread in main()


Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | jump to line | /mnt/shared/codes/saxpy/saxpy.hip.cpp:22 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;

}
```

```
(gdb) i th
  Id   Target Id                                   Frame
* 1    Thread 0x7ffff7fb6880 (LWP 55024) "saxpy" main () at saxpy.hip.cpp:22
(gdb) 
```

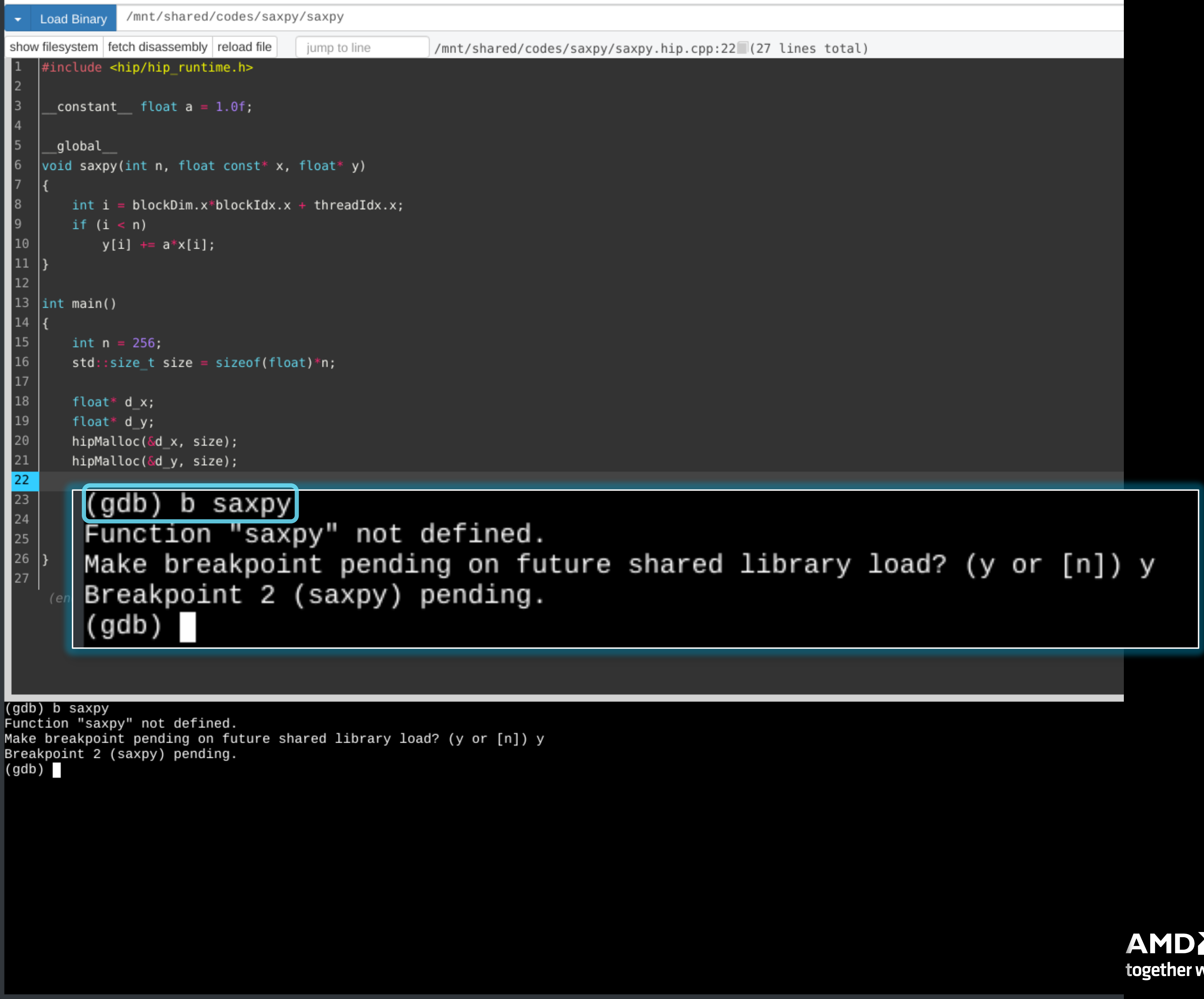
```
(gdb) i th
  Id   Target Id                                   Frame
* 1    Thread 0x7ffff7fb6880 (LWP 55024) "saxpy" main () at saxpy.hip.cpp:22
(gdb) 
```

AMD
together we advance_

set GPU breakpoint

b saxpy
(breakpoint saxpy)

- ➢ set a breakpoint in saxpy



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | jump to line /mnt/shared/codes/saxpy/saxpy.hip.cpp:22 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);
```

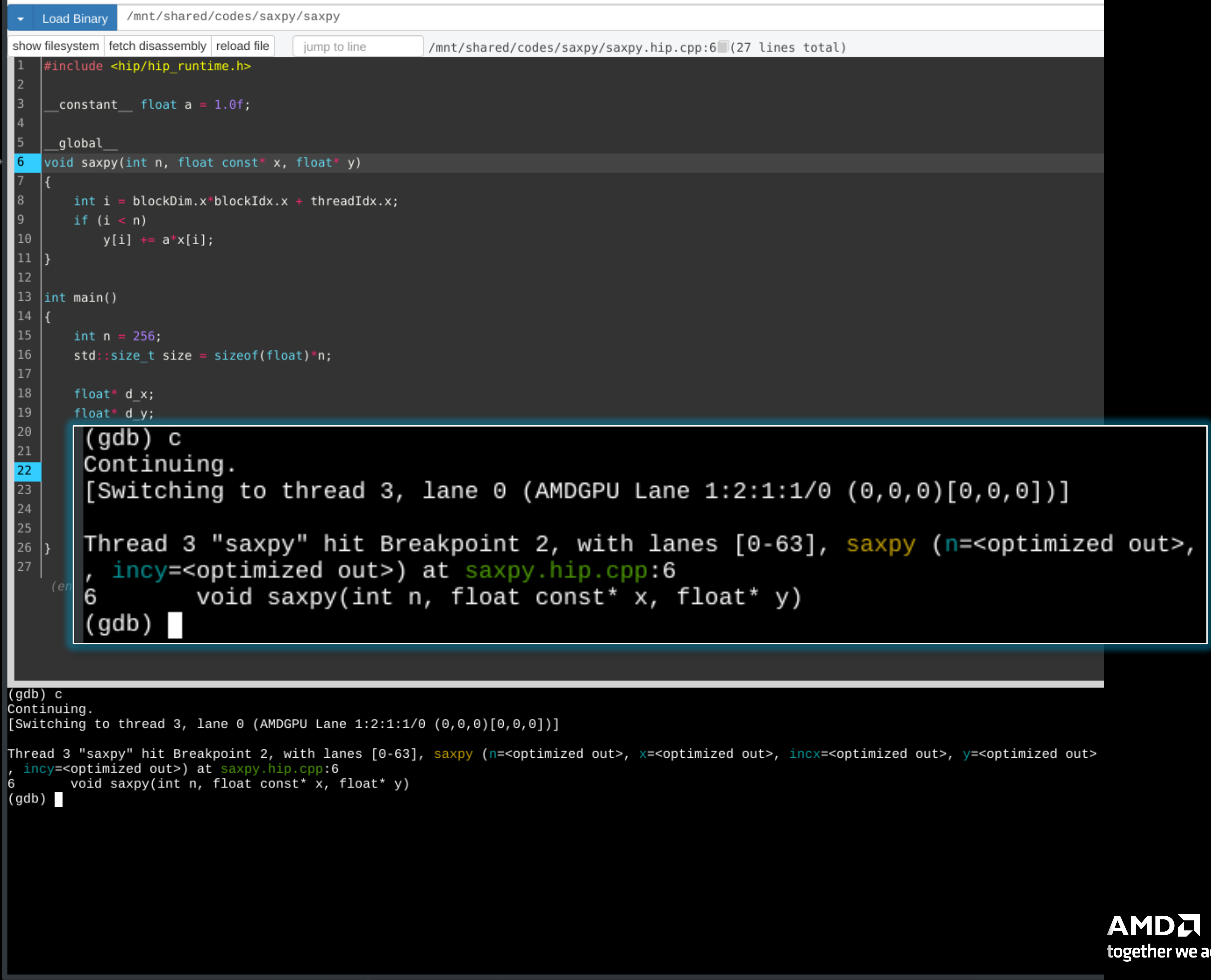
```
(gdb) b saxpy
Function "saxpy" not defined.
Make breakpoint pending on future shared library load? (y or [n]) y
Breakpoint 2 (saxpy) pending.
(gdb)
```

```
(gdb) b saxpy
Function "saxpy" not defined.
Make breakpoint pending on future shared library load? (y or [n]) y
Breakpoint 2 (saxpy) pending.
(gdb)
```

AMD together we advance_

hit GPU breakpoint

c
(continue)
➢ hit the kernel breakpoint



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | jump to line | /mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
```

```
}
```

```
(gdb) c
Continuing.
[Switching to thread 3, lane 0 (AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0])]

Thread 3 "saxpy" hit Breakpoint 2, with lanes [0-63], saxpy (n=<optimized out>,
, incy=<optimized out>) at saxpy.hip.cpp:6
6         void saxpy(int n, float const* x, float* y)
(gdb) 
```

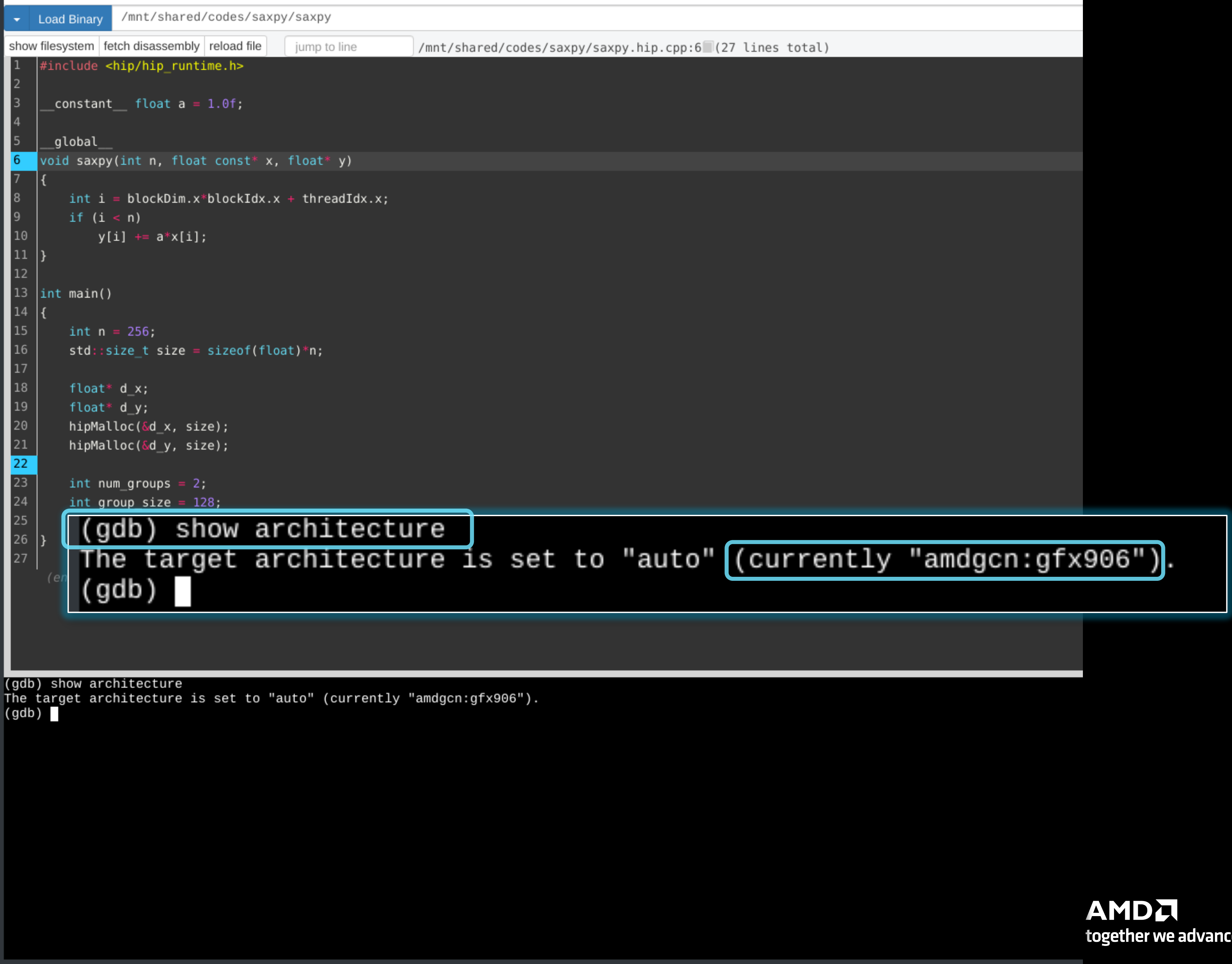
```
(gdb) c
Continuing.
[Switching to thread 3, lane 0 (AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0])]

Thread 3 "saxpy" hit Breakpoint 2, with lanes [0-63], saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, y=<optimized out>
, incy=<optimized out>) at saxpy.hip.cpp:6
6         void saxpy(int n, float const* x, float* y)
(gdb) 
```

AMD
together we advance_

show GPU arch

show architecture

- ➢ gfx906

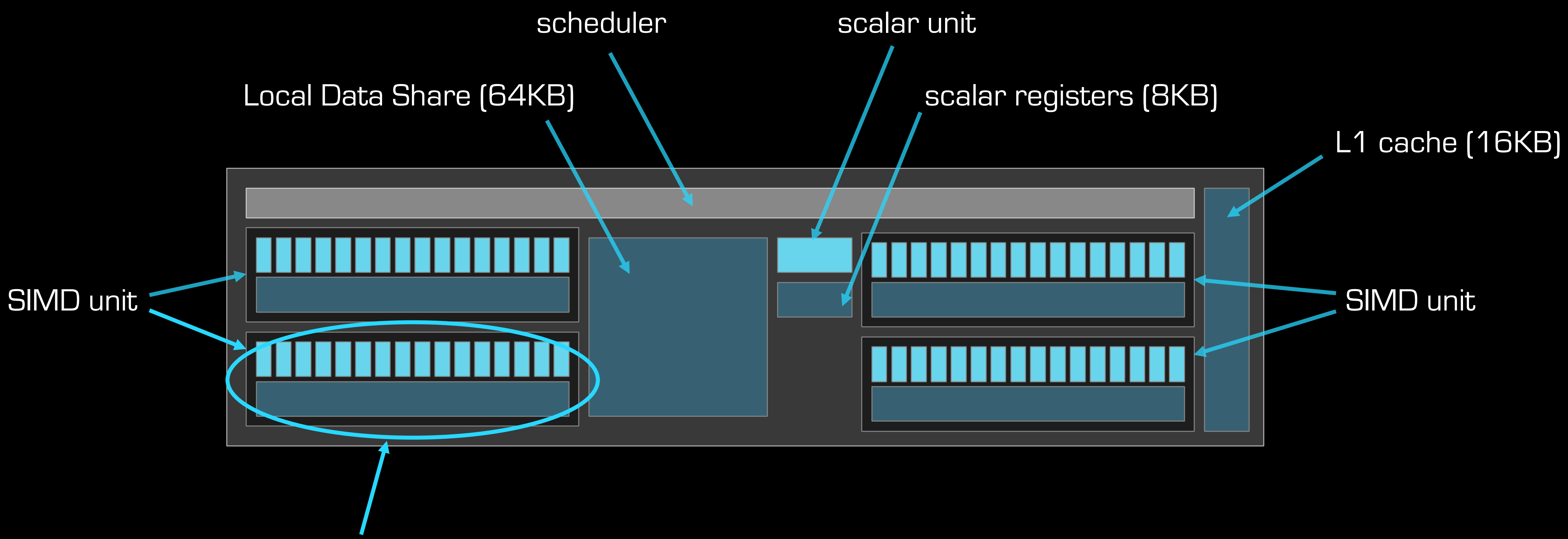

Load Binary
/mnt/shared/codes/saxpy/saxpy
show filesystem
fetch disassembly
reload file
jump to line
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;

    float* d_x;
    float* d_y;
    hipMalloc(&d_x, size);
    hipMalloc(&d_y, size);

    int num_groups = 2;
    int group_size = 128;
```

```
(gdb) show architecture
The target architecture is set to "auto" (currently "amdgcn:gfx906").
(gdb) 
```

```
(gdb) show architecture
The target architecture is set to "auto" (currently "amdgcn:gfx906").
(gdb) 
```

MI200 compute unit

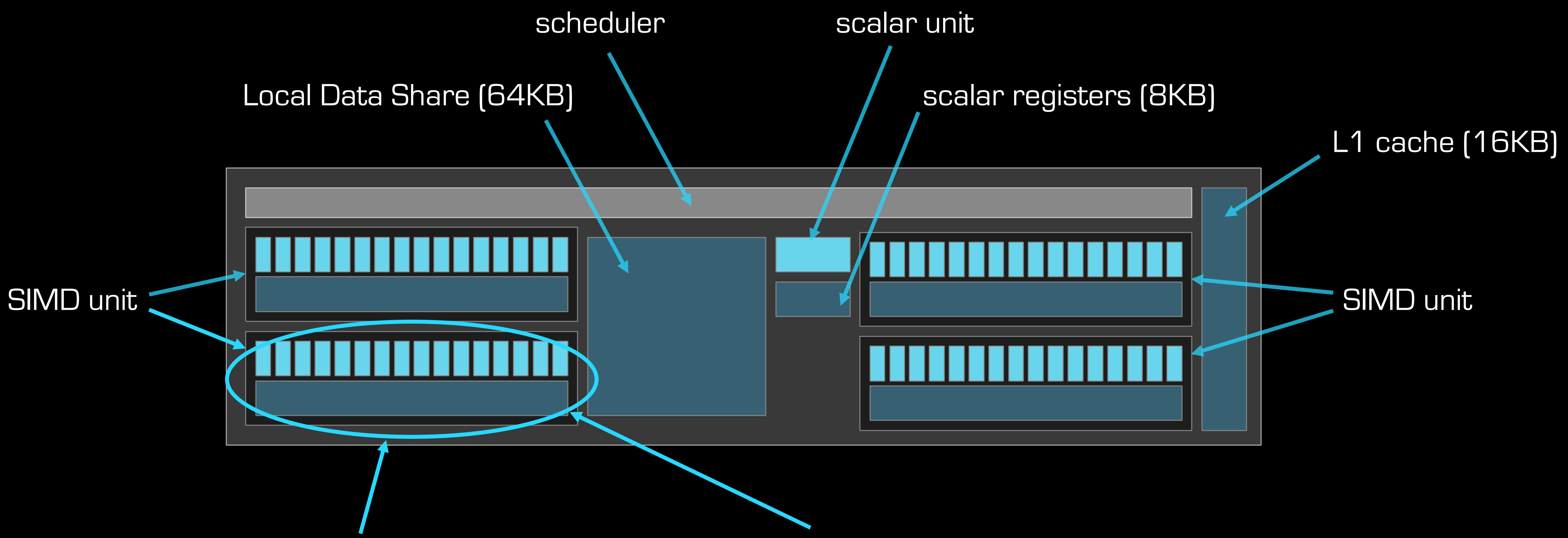


typically described as

- a 16-way SIMD unit
- with 64KB of registers

AMD
together we advance_

MI200 compute unit



typically described as

- a 16-way SIMD unit
- with 64KB of registers

from the standpoint of rocGDB

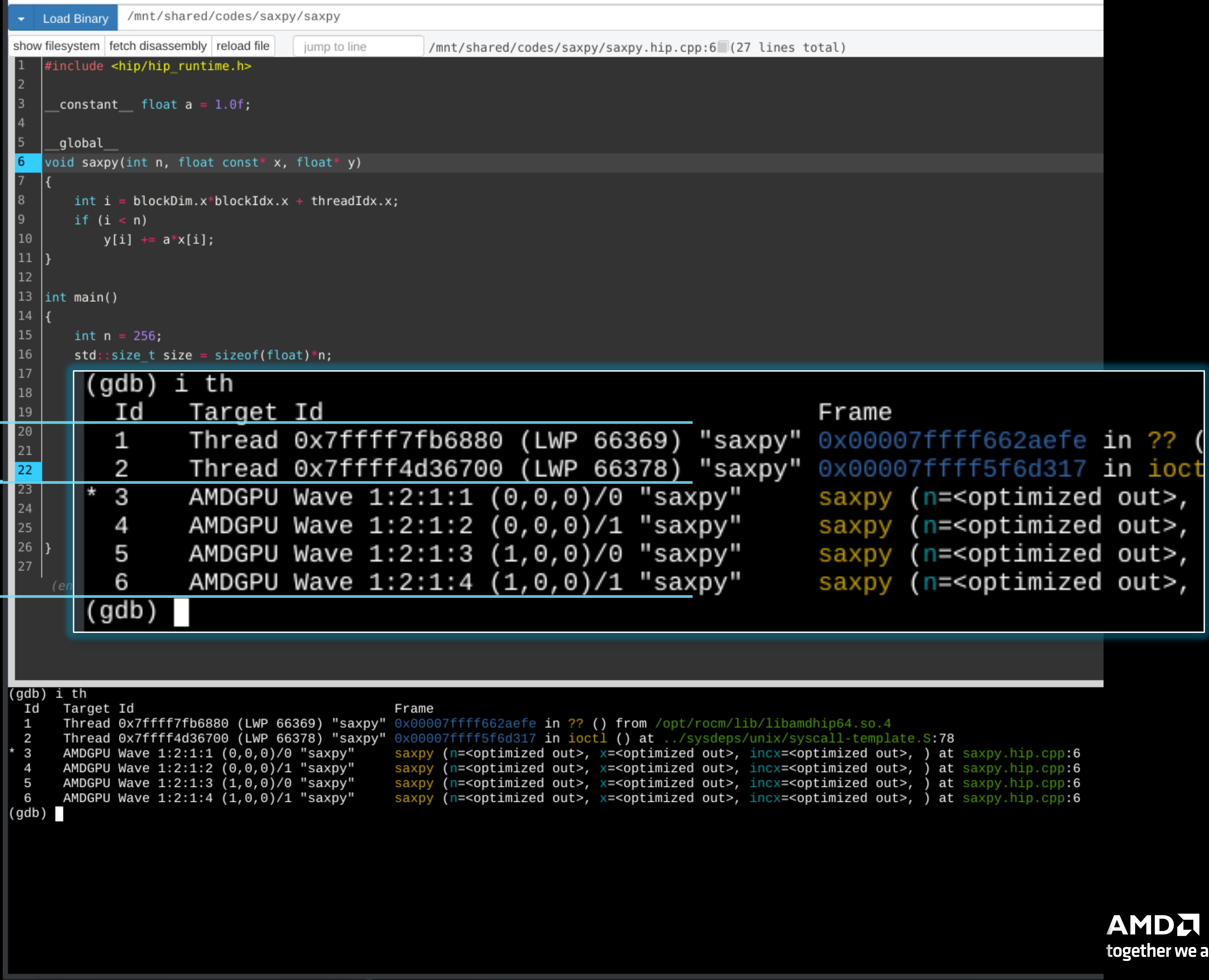
- a **core**
- executing up to 10 **threads**
- with vector length of 64 **lanes**
- and containing 256 vector **registers**

list threads / waves

i th
(info threads)

some CPU threads

4 GPU “threads” (waves)


Load Binary
/mnt/shared/codes/saxpy/saxpy
show filesystem
fetch disassembly
reload file
jump to line
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

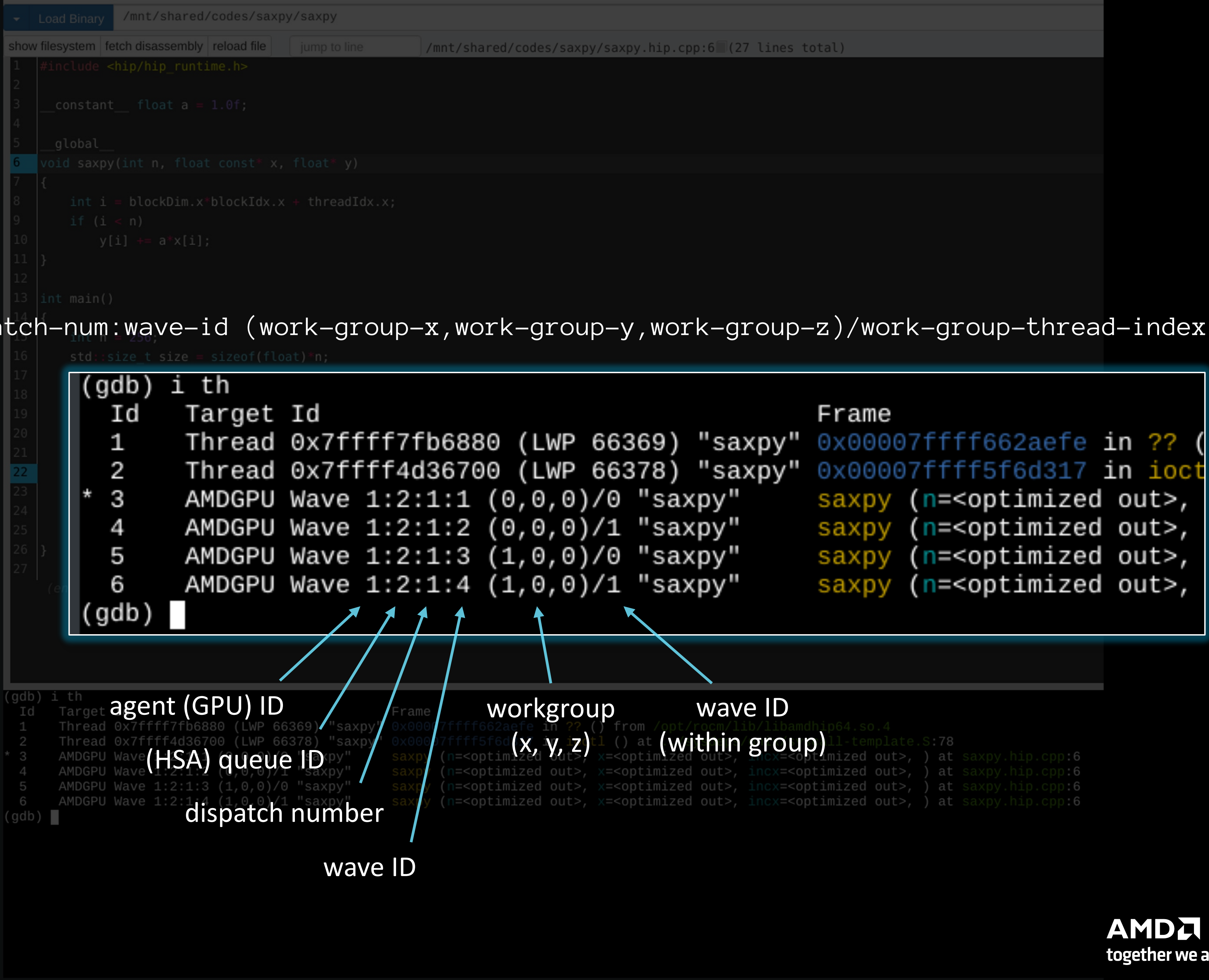
```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
    int n = 256;
    std::size_t size = sizeof(float)*n;
```

```
(gdb) i th
  Id   Target Id                                   Frame
  1    Thread 0x7ffff7fb6880 (LWP 66369) "saxpy" 0x00007ffff662aefe in ?? () from /opt/rocm/lib/libamdhip64.so.4
  2    Thread 0x7ffff4d36700 (LWP 66378) "saxpy" 0x00007ffff5f6d317 in ioctl () at ../sysdeps/unix/syscall-template.S:78
* 3    AMDGPU Wave 1:2:1:1 (0,0,0)/0 "saxpy"     saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, ) at saxpy.hip.cpp:6
  4    AMDGPU Wave 1:2:1:2 (0,0,0)/1 "saxpy"     saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, ) at saxpy.hip.cpp:6
  5    AMDGPU Wave 1:2:1:3 (1,0,0)/0 "saxpy"     saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, ) at saxpy.hip.cpp:6
  6    AMDGPU Wave 1:2:1:4 (1,0,0)/1 "saxpy"     saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, ) at saxpy.hip.cpp:6
(gdb)
```

AMD
together we advance_

wave details

`agent-id:queue-id:dispatch-num:wave-id (work-group-x,work-group-y,work-group-z)/work-group-thread-index`



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{

    std::size_t size = sizeof(float)*n;
```

```
(gdb) i th
  Id   Target Id                                         Frame
  1    Thread 0x7ffff7fb6880 (LWP 66369) "saxpy" 0x00007ffff662aefe in ?? (
  2    Thread 0x7ffff4d36700 (LWP 66378) "saxpy" 0x00007ffff5f6d317 in ioct
* 3    AMDGPU Wave 1:2:1:1 (0,0,0)/0 "saxpy"     saxpy (n=<optimized out>,
  4    AMDGPU Wave 1:2:1:2 (0,0,0)/1 "saxpy"     saxpy (n=<optimized out>,
  5    AMDGPU Wave 1:2:1:3 (1,0,0)/0 "saxpy"     saxpy (n=<optimized out>,
  6    AMDGPU Wave 1:2:1:4 (1,0,0)/1 "saxpy"     saxpy (n=<optimized out>,
(gdb) 
```

together we advance_

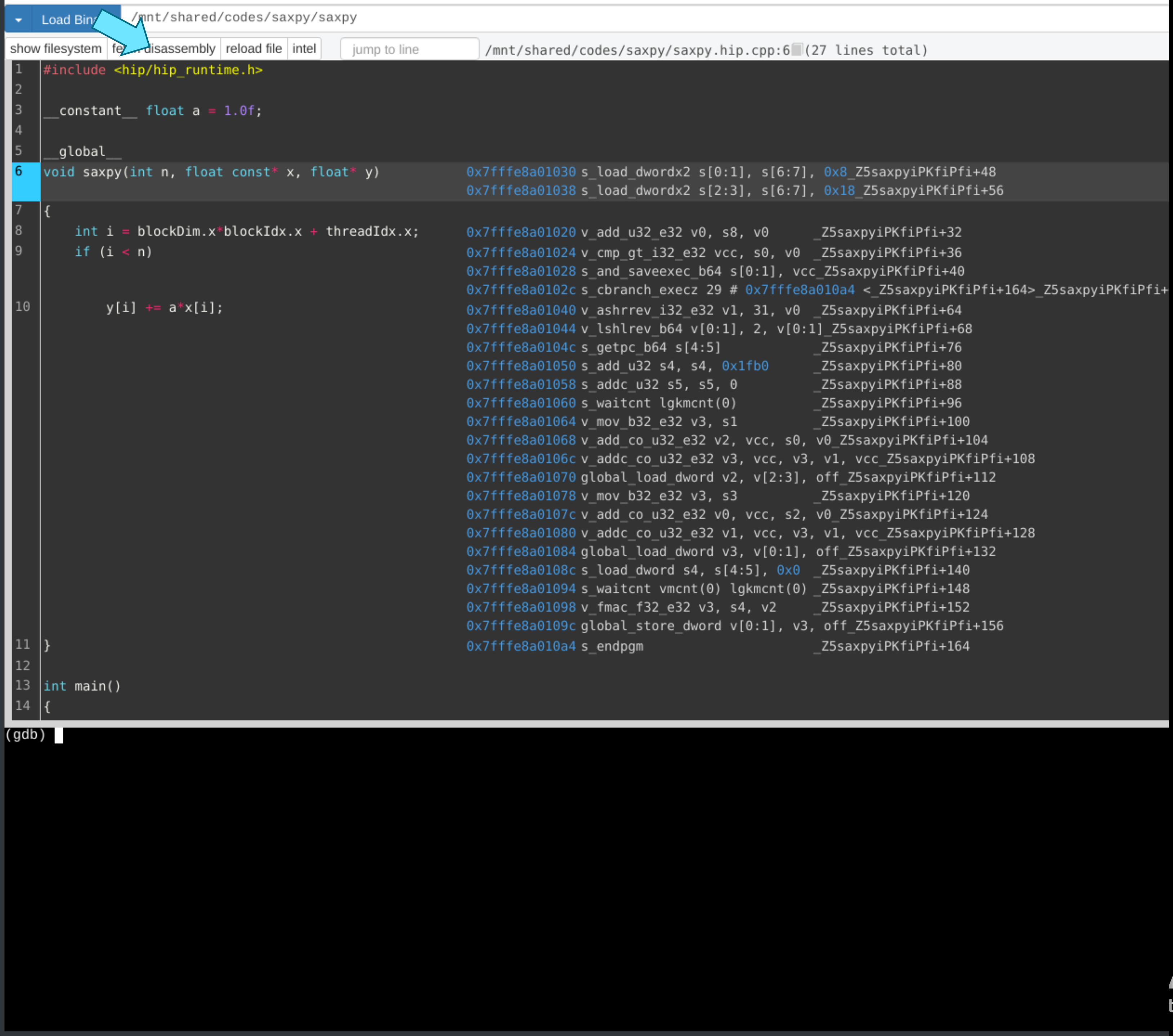
show assembly

fetch disassembly

- ➢ assembly next to source

-O3 is the default for device code

-O0 gives better ISA-source correlation

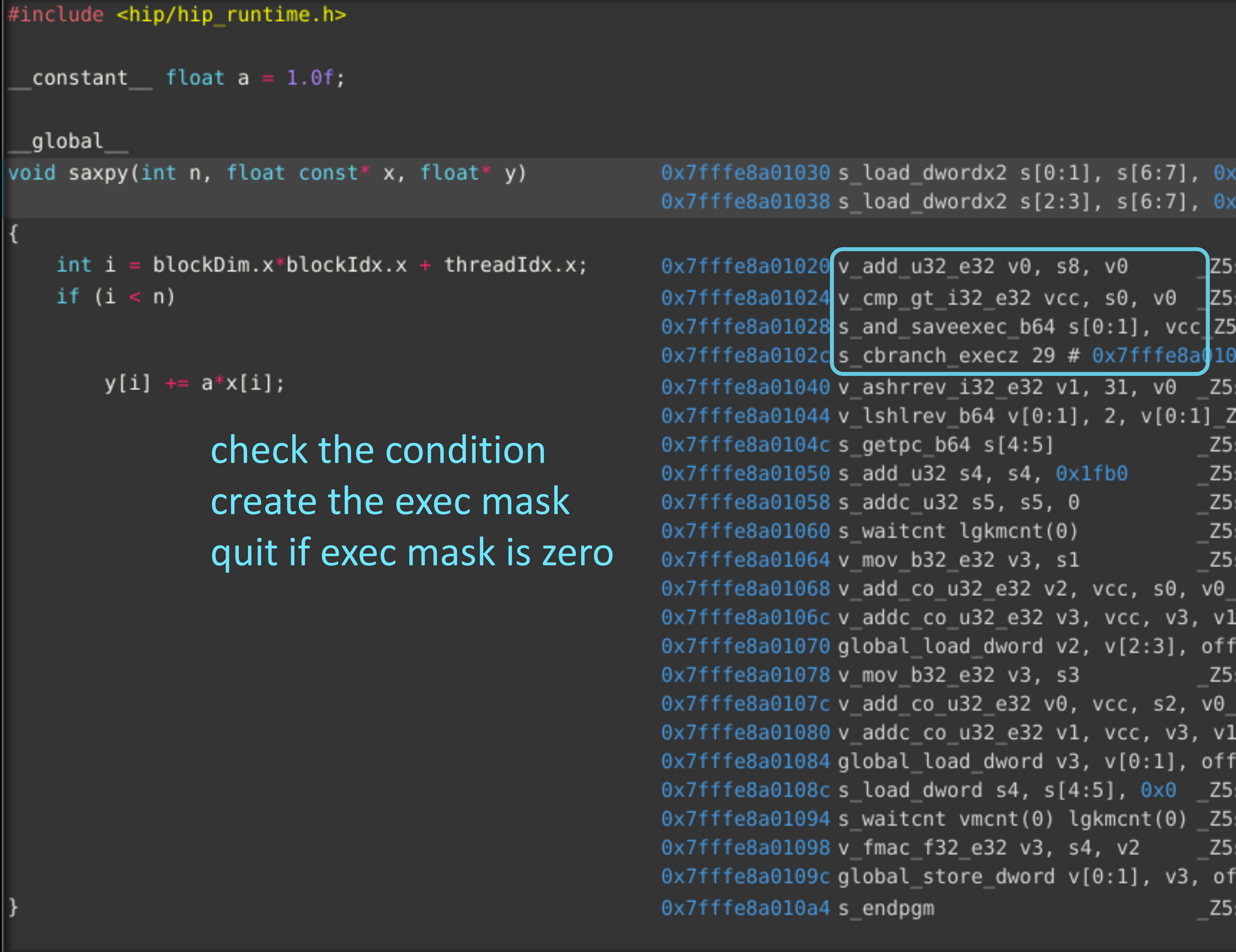

Load Bin
/mnt/shared/codes/saxpy/saxpy
show filesystem
fe disassembly
reload file
intel
jump to line
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)              0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
                                                         0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;         0x7fffe8a01020 v_add_u32_e32 v0, s8, v0      _Z5saxpyiPKfiPfi+32
    if (i < n)                                           0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0  _Z5saxpyiPKfiPfi+36
                                                         0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
                                                         0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164> _Z5saxpyiPKfiPfi+
        y[i] += a*x[i];                                  0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0  _Z5saxpyiPKfiPfi+64
                                                         0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
                                                         0x7fffe8a0104c s_getpc_b64 s[4:5]            _Z5saxpyiPKfiPfi+76
                                                         0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0      _Z5saxpyiPKfiPfi+80
                                                         0x7fffe8a01058 s_addc_u32 s5, s5, 0          _Z5saxpyiPKfiPfi+88
                                                         0x7fffe8a01060 s_waitcnt lgkmcnt(0)          _Z5saxpyiPKfiPfi+96
                                                         0x7fffe8a01064 v_mov_b32_e32 v3, s1          _Z5saxpyiPKfiPfi+100
                                                         0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
                                                         0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
                                                         0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
                                                         0x7fffe8a01078 v_mov_b32_e32 v3, s3          _Z5saxpyiPKfiPfi+120
                                                         0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _Z5saxpyiPKfiPfi+124
                                                         0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+128
                                                         0x7fffe8a01084 global_load_dword v3, v[0:1], off _Z5saxpyiPKfiPfi+132
                                                         0x7fffe8a0108c s_load_dword s4, s[4:5], 0x0  _Z5saxpyiPKfiPfi+140
                                                         0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0) _Z5saxpyiPKfiPfi+148
                                                         0x7fffe8a01098 v_fmac_f32_e32 v3, s4, v2     _Z5saxpyiPKfiPfi+152
                                                         0x7fffe8a0109c global_store_dword v[0:1], v3, off _Z5saxpyiPKfiPfi+156
}                                                        0x7fffe8a010a4 s_endpgm                      _Z5saxpyiPKfiPfi+164

int main()
{
```

```
(gdb)
```

AMD
together we advance_

inspect assembly



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)

{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)


        y[i] += a*x[i];

}
```

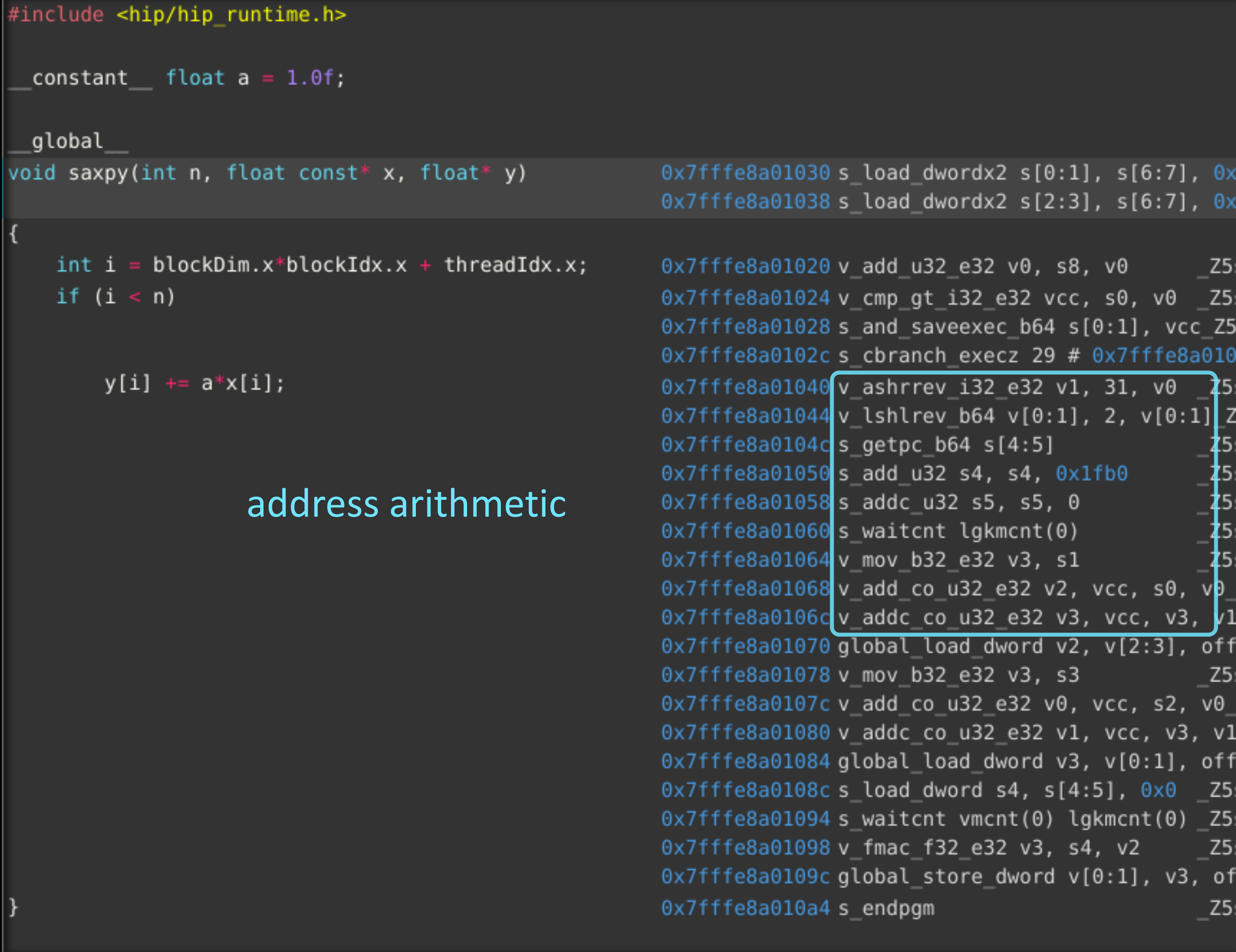
```
0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x
0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x

0x7fffe8a01020 v_add_u32_e32 v0, s8, v0        _Z5
0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0    _Z5
0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc  _Z5
0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010
0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0    _Z5
0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z
0x7fffe8a0104c s_getpc_b64 s[4:5]              _Z5
0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0        _Z5
0x7fffe8a01058 s_addc_u32 s5, s5, 0            _Z5
0x7fffe8a01060 s_waitcnt lgkmcnt(0)            _Z5
0x7fffe8a01064 v_mov_b32_e32 v3, s1            _Z5
0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _
0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1
0x7fffe8a01070 global_load_dword v2, v[2:3], off
0x7fffe8a01078 v_mov_b32_e32 v3, s3            _Z5
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _
0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1
0x7fffe8a01084 global_load_dword v3, v[0:1], off
0x7fffe8a0108c s_load_dword s4, s[4:5], 0x0    _Z5
0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0)   _Z5
0x7fffe8a01098 v_fmac_f32_e32 v3, s4, v2       _Z5
0x7fffe8a0109c global_store_dword v[0:1], v3, of
0x7fffe8a010a4 s_endpgm                        _Z5
```

check the condition

create the exec mask

quit if exec mask is zero

inspect assembly

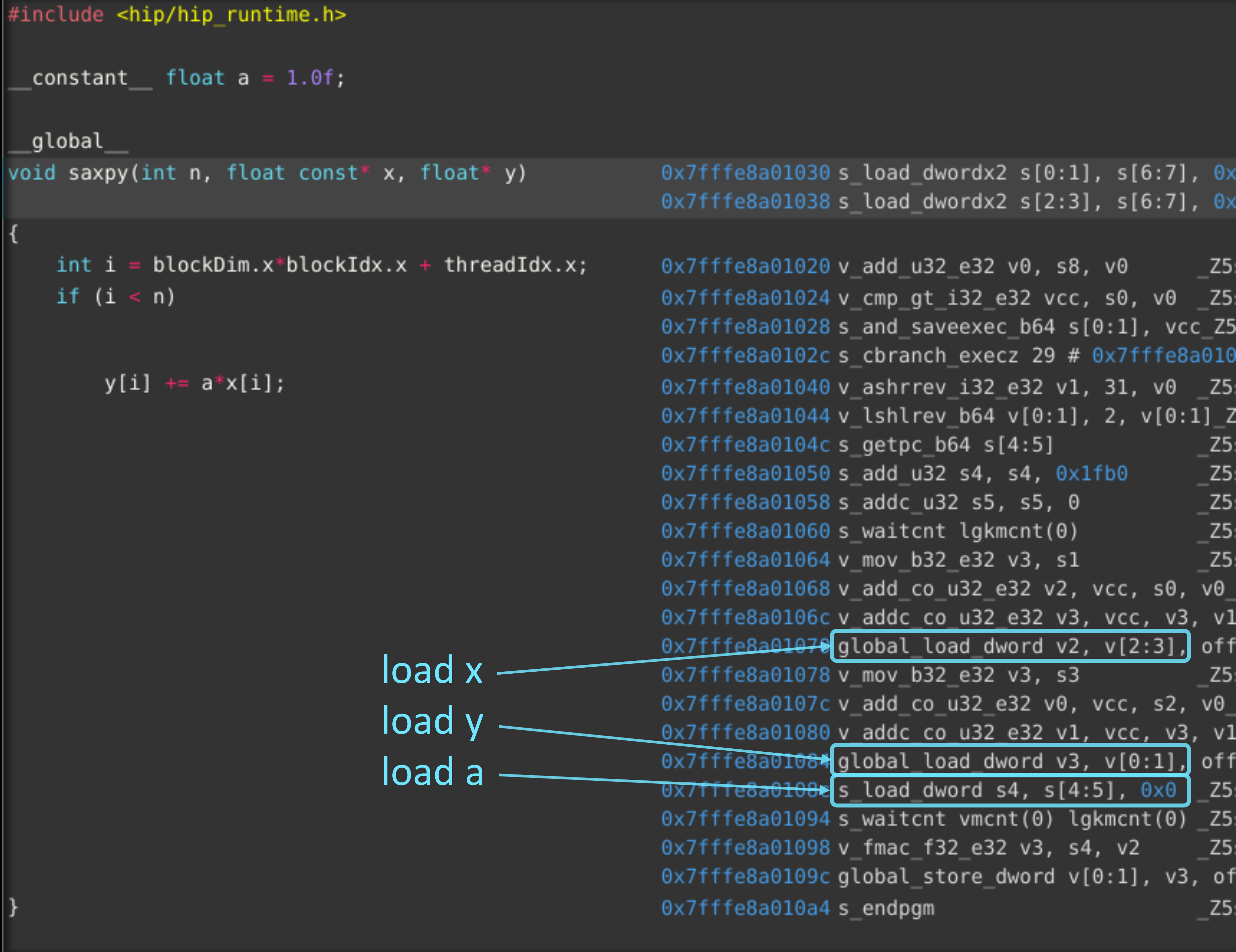


```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)

{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)


        y[i] += a*x[i];
}
```

```
0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x
0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x

0x7fffe8a01020 v_add_u32_e32 v0, s8, v0       _Z5
0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0   _Z5
0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5
0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010
0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0   _Z5
0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z
0x7fffe8a0104c s_getpc_b64 s[4:5]             _Z5
0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0       _Z5
0x7fffe8a01058 s_addc_u32 s5, s5, 0           _Z5
0x7fffe8a01060 s_waitcnt lgkmcnt(0)           _Z5
0x7fffe8a01064 v_mov_b32_e32 v3, s1           _Z5
0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _
0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1
0x7fffe8a01070 global_load_dword v2, v[2:3], off
0x7fffe8a01078 v_mov_b32_e32 v3, s3           _Z5
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _
0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1
0x7fffe8a01084 global_load_dword v3, v[0:1], off
0x7fffe8a0108c s_load_dword s4, s[4:5], 0x0   _Z5
0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0)  _Z5
0x7fffe8a01098 v_fmac_f32_e32 v3, s4, v2      _Z5
0x7fffe8a0109c global_store_dword v[0:1], v3, of
0x7fffe8a010a4 s_endpgm                       _Z5
```

address arithmetic

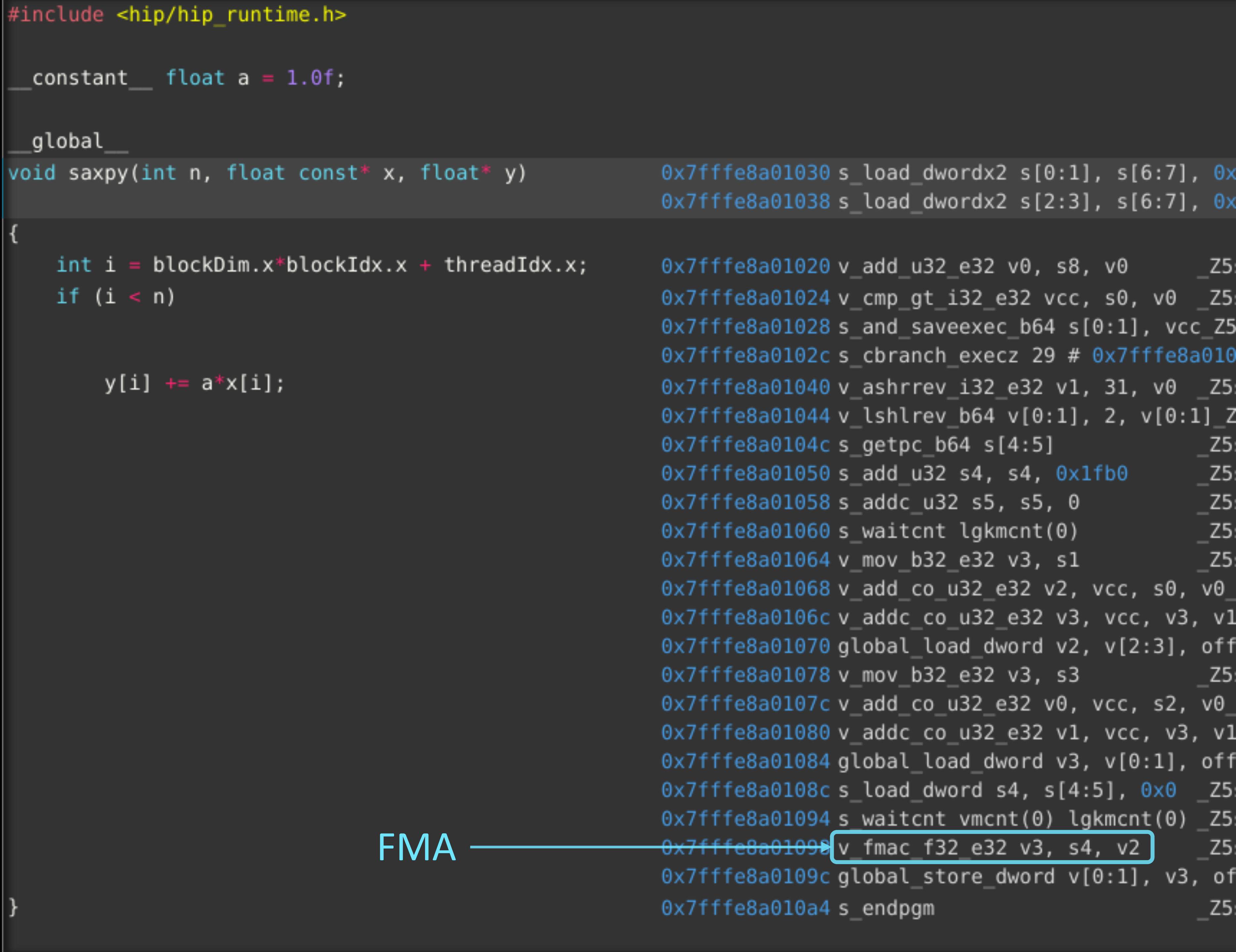
inspect assembly



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)

{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)


        y[i] += a*x[i];

}
```

```
0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x
0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x

0x7fffe8a01020 v_add_u32_e32 v0, s8, v0       _Z5
0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0   _Z5
0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5
0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010
0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0   _Z5
0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z
0x7fffe8a0104c s_getpc_b64 s[4:5]             _Z5
0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0       _Z5
0x7fffe8a01058 s_addc_u32 s5, s5, 0           _Z5
0x7fffe8a01060 s_waitcnt lgkmcnt(0)           _Z5
0x7fffe8a01064 v_mov_b32_e32 v3, s1           _Z5
0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _
0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1
0x7fffe8a01070 global_load_dword v2, v[2:3], off
0x7fffe8a01078 v_mov_b32_e32 v3, s3           _Z5
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _
0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1
0x7fffe8a01084 global_load_dword v3, v[0:1], off
0x7fffe8a01088 s_load_dword s4, s[4:5], 0x0   _Z5
0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0)  _Z5
0x7fffe8a01098 v_fmac_f32_e32 v3, s4, v2      _Z5
0x7fffe8a0109c global_store_dword v[0:1], v3, of
0x7fffe8a010a4 s_endpgm                       _Z5
```

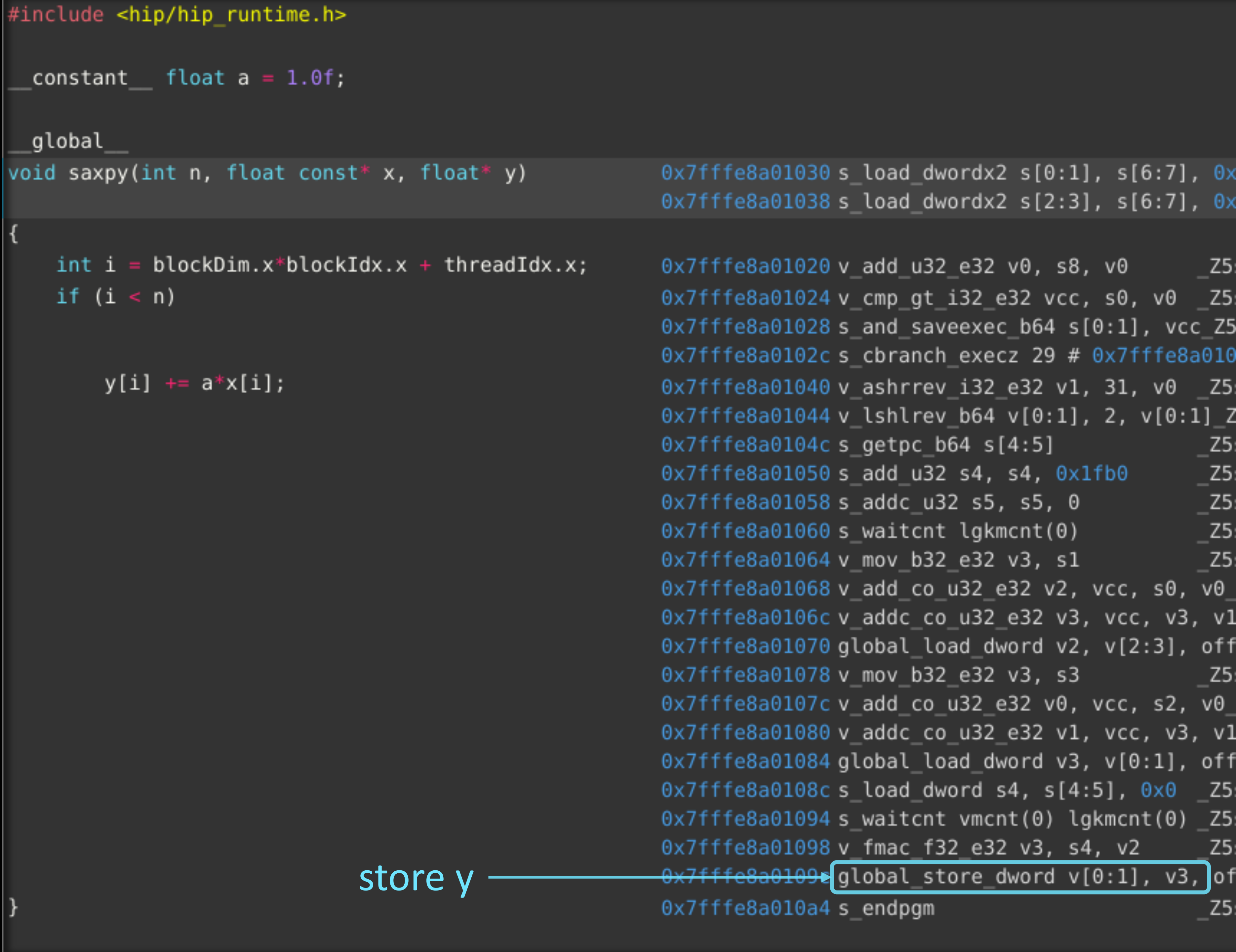
inspect assembly



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)

        y[i] += a*x[i];
}
```

```
0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x
0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x
0x7fffe8a01020 v_add_u32_e32 v0, s8, v0        _Z5
0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0    _Z5
0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5
0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010
0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0    _Z5
0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z
0x7fffe8a0104c s_getpc_b64 s[4:5]              _Z5
0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0        _Z5
0x7fffe8a01058 s_addc_u32 s5, s5, 0            _Z5
0x7fffe8a01060 s_waitcnt lgkmcnt(0)            _Z5
0x7fffe8a01064 v_mov_b32_e32 v3, s1            _Z5
0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _
0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1
0x7fffe8a01070 global_load_dword v2, v[2:3], off
0x7fffe8a01078 v_mov_b32_e32 v3, s3            _Z5
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _
0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1
0x7fffe8a01084 global_load_dword v3, v[0:1], off
0x7fffe8a0108c s_load_dword s4, s[4:5], 0x0    _Z5
0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0)   _Z5
0x7fffe8a01098 v_fmac_f32_e32 v3, s4, v2       _Z5
0x7fffe8a0109c global_store_dword v[0:1], v3, of
0x7fffe8a010a4 s_endpgm                        _Z5
```

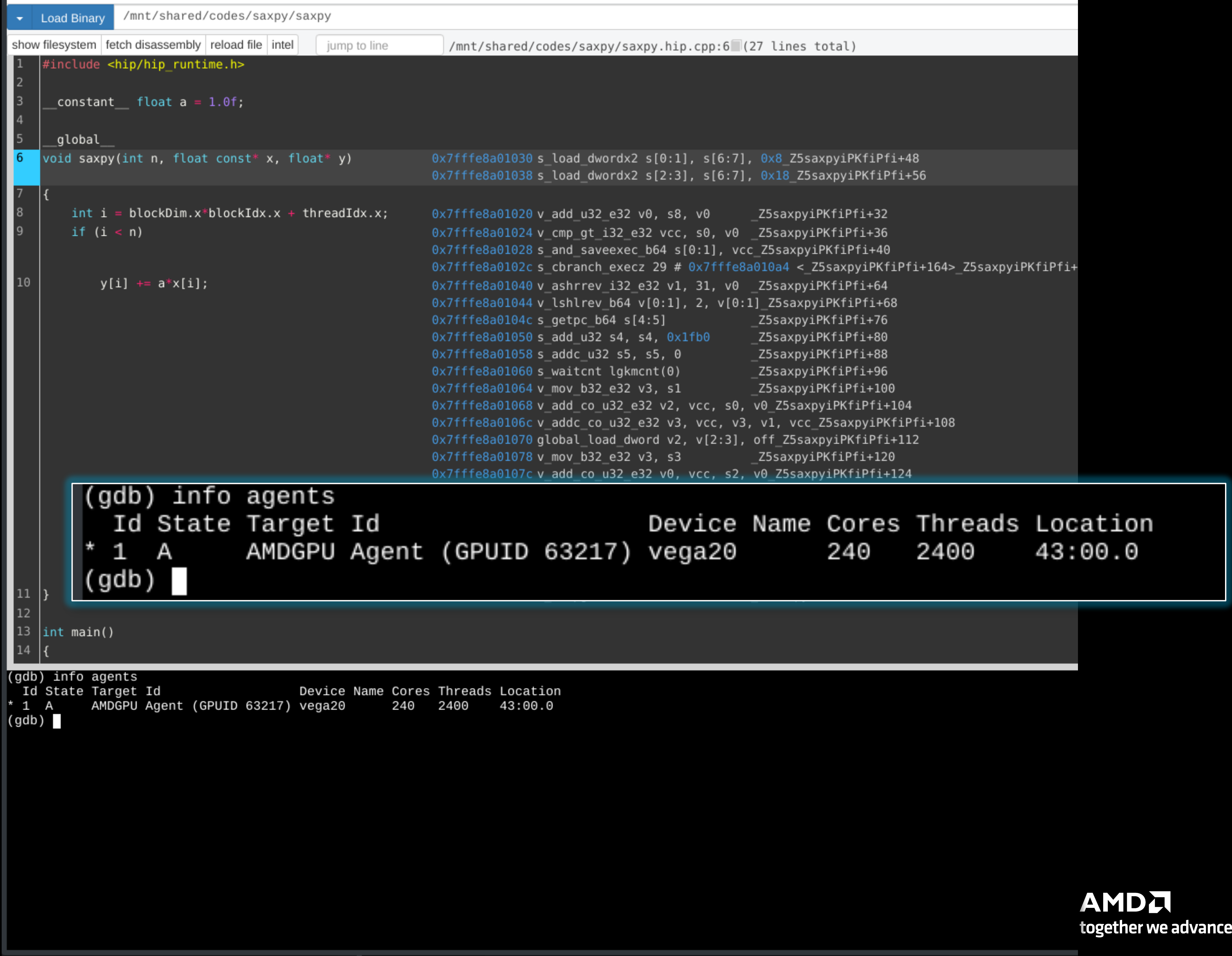
inspect assembly



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)       0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x
                                                  0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;  0x7fffe8a01020 v_add_u32_e32 v0, s8, v0       _Z5
    if (i < n)                                    0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0   _Z5
                                                  0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5
                                                  0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010
        y[i] += a*x[i];                           0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0   _Z5
                                                  0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z
                                                  0x7fffe8a0104c s_getpc_b64 s[4:5]             _Z5
                                                  0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0       _Z5
                                                  0x7fffe8a01058 s_addc_u32 s5, s5, 0           _Z5
                                                  0x7fffe8a01060 s_waitcnt lgkmcnt(0)           _Z5
                                                  0x7fffe8a01064 v_mov_b32_e32 v3, s1           _Z5
                                                  0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _
                                                  0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1
                                                  0x7fffe8a01070 global_load_dword v2, v[2:3], off
                                                  0x7fffe8a01078 v_mov_b32_e32 v3, s3           _Z5
                                                  0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _
                                                  0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1
                                                  0x7fffe8a01084 global_load_dword v3, v[0:1], off
                                                  0x7fffe8a0108c s_load_dword s4, s[4:5], 0x0   _Z5
                                                  0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0)  _Z5
                                                  0x7fffe8a01098 v_fmac_f32_e32 v3, s4, v2      _Z5
                                       store y ─> 0x7fffe8a0109c global_store_dword v[0:1], v3, of
}                                                 0x7fffe8a010a4 s_endpgm                       _Z5
```

list agents

info agents
- ➢ shows devices + properties


Load Binary
/mnt/shared/codes/saxpy/saxpy
show filesystem
fetch disassembly
reload file
intel
jump to line
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)
        y[i] += a*x[i];
}

int main()
{
```

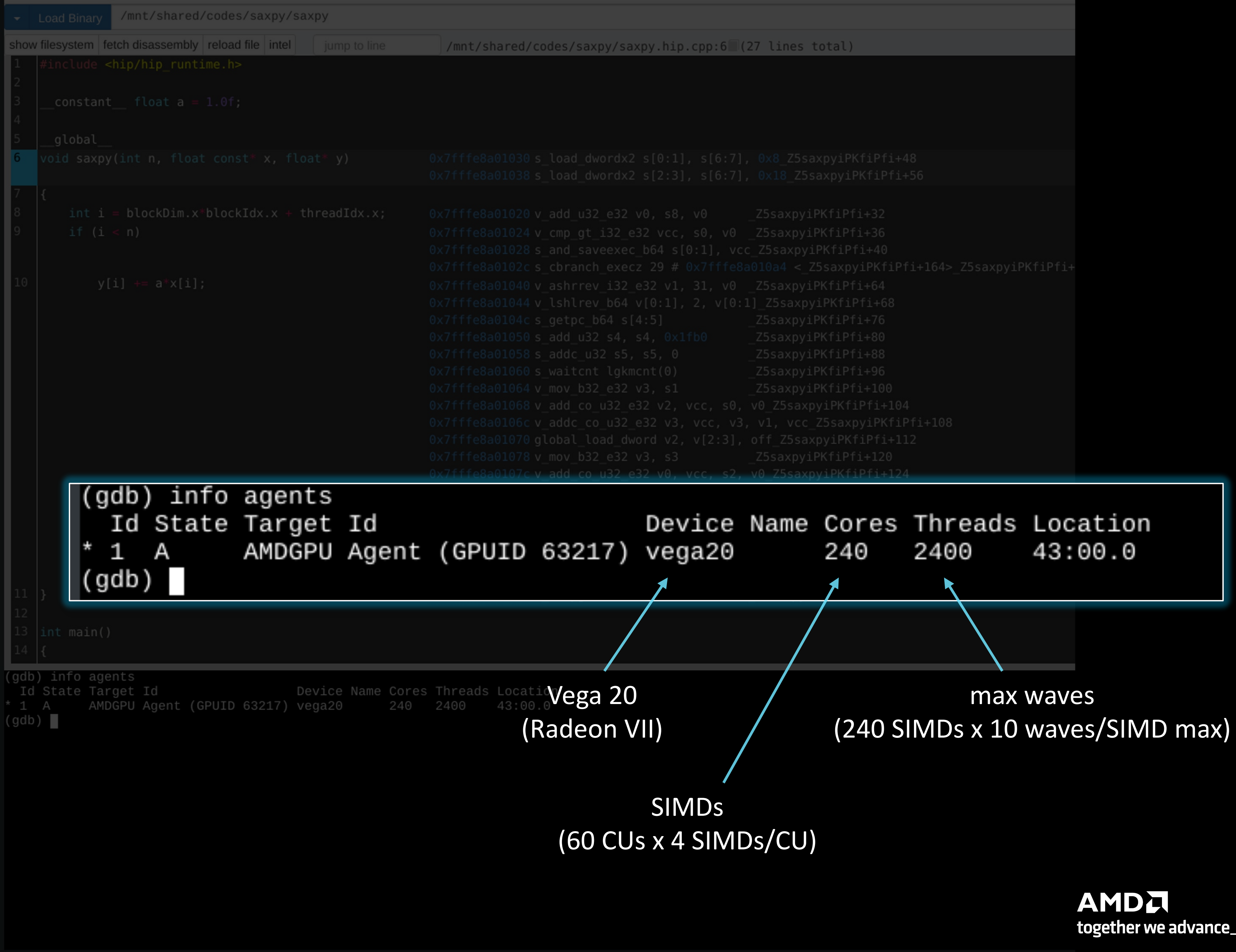
```
0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
0x7fffe8a01020 v_add_u32_e32 v0, s8, v0      _Z5saxpyiPKfiPfi+32
0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0  _Z5saxpyiPKfiPfi+36
0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164> _Z5saxpyiPKfiPfi+
0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0  _Z5saxpyiPKfiPfi+64
0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
0x7fffe8a0104c s_getpc_b64 s[4:5]            _Z5saxpyiPKfiPfi+76
0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0      _Z5saxpyiPKfiPfi+80
0x7fffe8a01058 s_addc_u32 s5, s5, 0          _Z5saxpyiPKfiPfi+88
0x7fffe8a01060 s_waitcnt lgkmcnt(0)          _Z5saxpyiPKfiPfi+96
0x7fffe8a01064 v_mov_b32_e32 v3, s1          _Z5saxpyiPKfiPfi+100
0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
0x7fffe8a01078 v_mov_b32_e32 v3, s3          _Z5saxpyiPKfiPfi+120
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _Z5saxpyiPKfiPfi+124
```

```
(gdb) info agents
  Id State Target Id                   Device Name Cores Threads Location
* 1  A     AMDGPU Agent (GPUID 63217) vega20      240   2400    43:00.0
(gdb)
```

```
(gdb) info agents
  Id State Target Id                   Device Name Cores Threads Location
* 1  A     AMDGPU Agent (GPUID 63217) vega20      240   2400    43:00.0
(gdb)
```

AMD
together we advance_

agent details



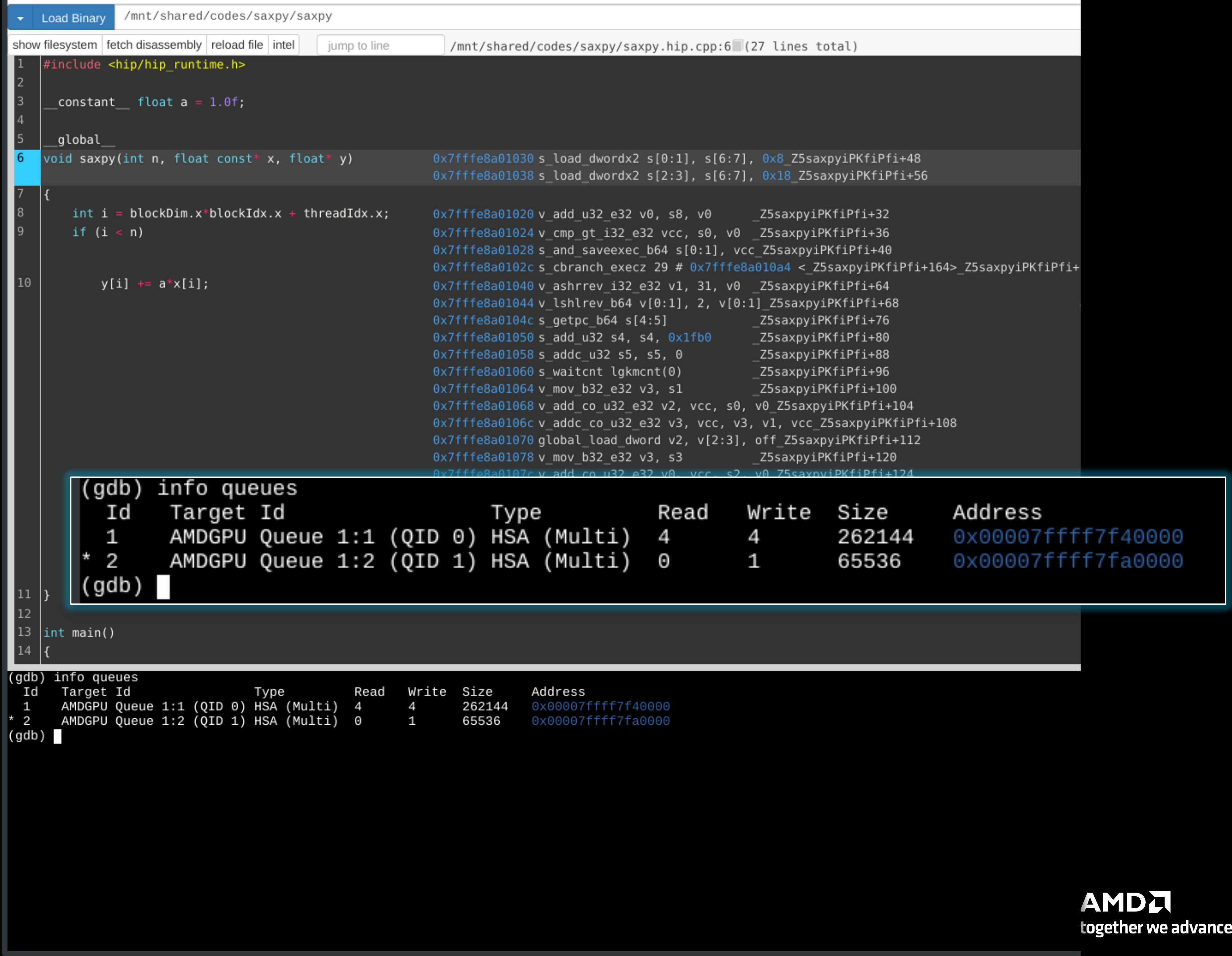
```
/mnt/shared/codes/saxpy/saxpy
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)
```

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)       0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
                                                  0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;  0x7fffe8a01020 v_add_u32_e32 v0, s8, v0      _Z5saxpyiPKfiPfi+32
    if (i < n)                                    0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0  _Z5saxpyiPKfiPfi+36
                                                  0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
                                                  0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164> _Z5saxpyiPKfiPfi+
        y[i] += a*x[i];                           0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0  _Z5saxpyiPKfiPfi+64
                                                  0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
                                                  0x7fffe8a0104c s_getpc_b64 s[4:5]            _Z5saxpyiPKfiPfi+76
                                                  0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0      _Z5saxpyiPKfiPfi+80
                                                  0x7fffe8a01058 s_addc_u32 s5, s5, 0          _Z5saxpyiPKfiPfi+88
                                                  0x7fffe8a01060 s_waitcnt lgkmcnt(0)          _Z5saxpyiPKfiPfi+96
                                                  0x7fffe8a01064 v_mov_b32_e32 v3, s1          _Z5saxpyiPKfiPfi+100
                                                  0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
                                                  0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
                                                  0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
                                                  0x7fffe8a01078 v_mov_b32_e32 v3, s3          _Z5saxpyiPKfiPfi+120
                                                  0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _Z5saxpyiPKfiPfi+124
}

int main()
{
```

```
(gdb) info agents
  Id State Target Id                   Device Name Cores Threads Location
* 1  A     AMDGPU Agent (GPUID 63217) vega20      240   2400    43:00.0
(gdb) █
```

```
(gdb) info agents
  Id State Target Id                   Device Name Cores Threads Locatio
* 1  A     AMDGPU Agent (GPUID 63217) vega20      240   2400    43:00.0
(gdb) █
```

list queues



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | intel | jump to line | /mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)      0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
                                                 0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
{
    int i = blockDim.x*blockIdx.x + threadIdx.x; 0x7fffe8a01020 v_add_u32_e32 v0, s8, v0      _Z5saxpyiPKfiPfi+32
    if (i < n)                                   0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0  _Z5saxpyiPKfiPfi+36
                                                 0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
                                                 0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164> _Z5saxpyiPKfiPfi+
        y[i] += a*x[i];                          0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0  _Z5saxpyiPKfiPfi+64
                                                 0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
                                                 0x7fffe8a0104c s_getpc_b64 s[4:5]            _Z5saxpyiPKfiPfi+76
                                                 0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0      _Z5saxpyiPKfiPfi+80
                                                 0x7fffe8a01058 s_addc_u32 s5, s5, 0          _Z5saxpyiPKfiPfi+88
                                                 0x7fffe8a01060 s_waitcnt lgkmcnt(0)          _Z5saxpyiPKfiPfi+96
                                                 0x7fffe8a01064 v_mov_b32_e32 v3, s1          _Z5saxpyiPKfiPfi+100
                                                 0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
                                                 0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
                                                 0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
                                                 0x7fffe8a01078 v_mov_b32_e32 v3, s3          _Z5saxpyiPKfiPfi+120
                                                 0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _Z5saxpyiPKfiPfi+124
}

int main()
{
```

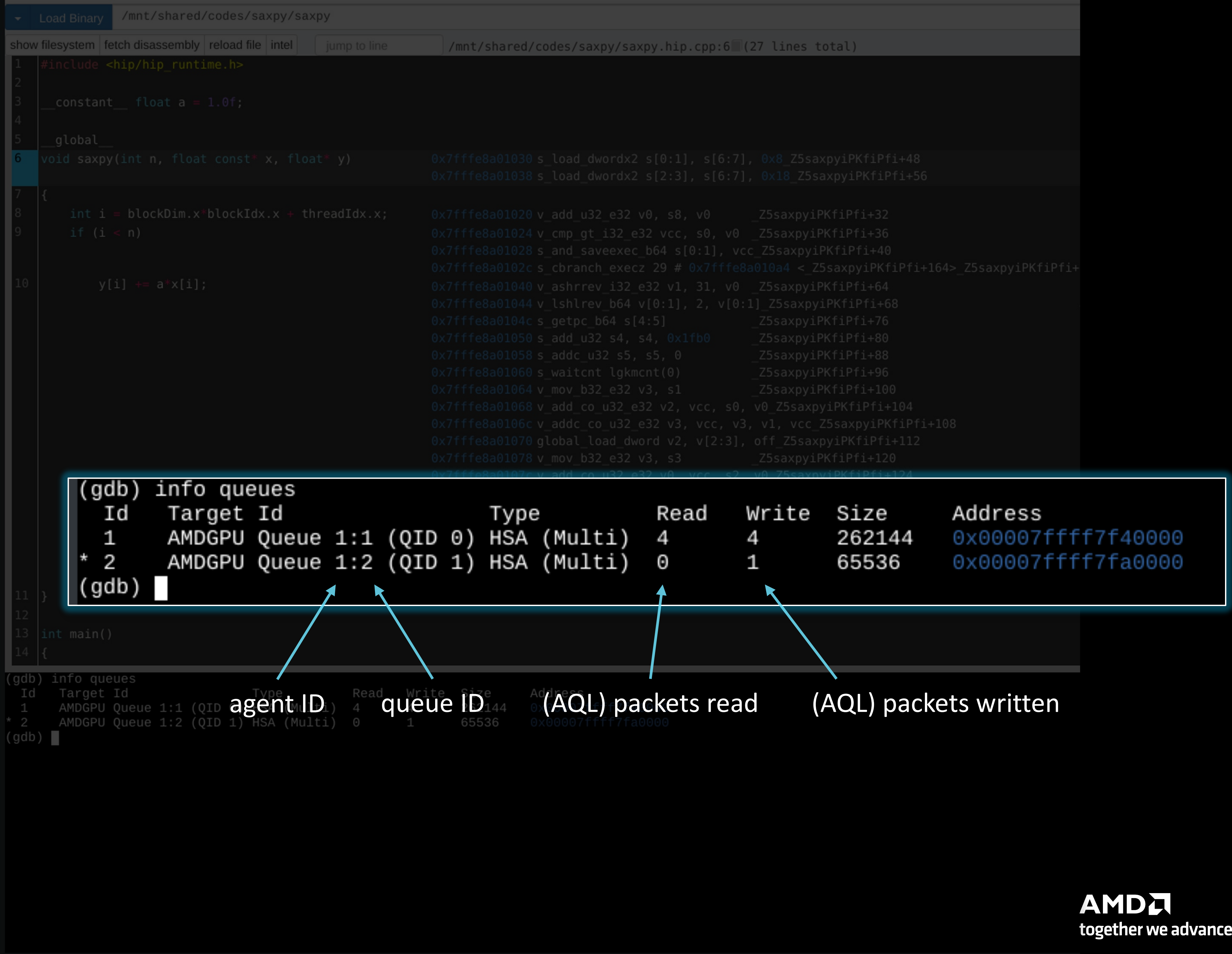
```
(gdb) info queues
  Id   Target Id                   Type          Read    Write   Size      Address
  1    AMDGPU Queue 1:1 (QID 0) HSA (Multi)  4       4       262144    0x00007ffff7f40000
* 2    AMDGPU Queue 1:2 (QID 1) HSA (Multi)  0       1       65536     0x00007ffff7fa0000
(gdb)
```

```
(gdb) info queues
  Id   Target Id                   Type          Read    Write   Size      Address
  1    AMDGPU Queue 1:1 (QID 0) HSA (Multi)  4       4       262144    0x00007ffff7f40000
* 2    AMDGPU Queue 1:2 (QID 1) HSA (Multi)  0       1       65536     0x00007ffff7fa0000
(gdb)
```

info queues
- shows HSA queues

AMD together we advance_

queue details



```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)          0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
                                                     0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;     0x7fffe8a01020 v_add_u32_e32 v0, s8, v0       _Z5saxpyiPKfiPfi+32
    if (i < n)                                       0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0   _Z5saxpyiPKfiPfi+36
                                                     0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
                                                     0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164> _Z5saxpyiPKfiPfi+
        y[i] += a*x[i];                              0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0   _Z5saxpyiPKfiPfi+64
                                                     0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
                                                     0x7fffe8a0104c s_getpc_b64 s[4:5]             _Z5saxpyiPKfiPfi+76
                                                     0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0       _Z5saxpyiPKfiPfi+80
                                                     0x7fffe8a01058 s_addc_u32 s5, s5, 0           _Z5saxpyiPKfiPfi+88
                                                     0x7fffe8a01060 s_waitcnt lgkmcnt(0)           _Z5saxpyiPKfiPfi+96
                                                     0x7fffe8a01064 v_mov_b32_e32 v3, s1           _Z5saxpyiPKfiPfi+100
                                                     0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
                                                     0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
                                                     0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
                                                     0x7fffe8a01078 v_mov_b32_e32 v3, s3           _Z5saxpyiPKfiPfi+120
}

int main()
{
```

```
(gdb) info queues
  Id    Target Id                  Type          Read    Write   Size      Address
  1     AMDGPU Queue 1:1 (QID 0)  HSA (Multi)   4       4       262144    0x00007ffff7f40000
* 2     AMDGPU Queue 1:2 (QID 1)  HSA (Multi)   0       1       65536     0x00007ffff7fa0000
(gdb) █
```

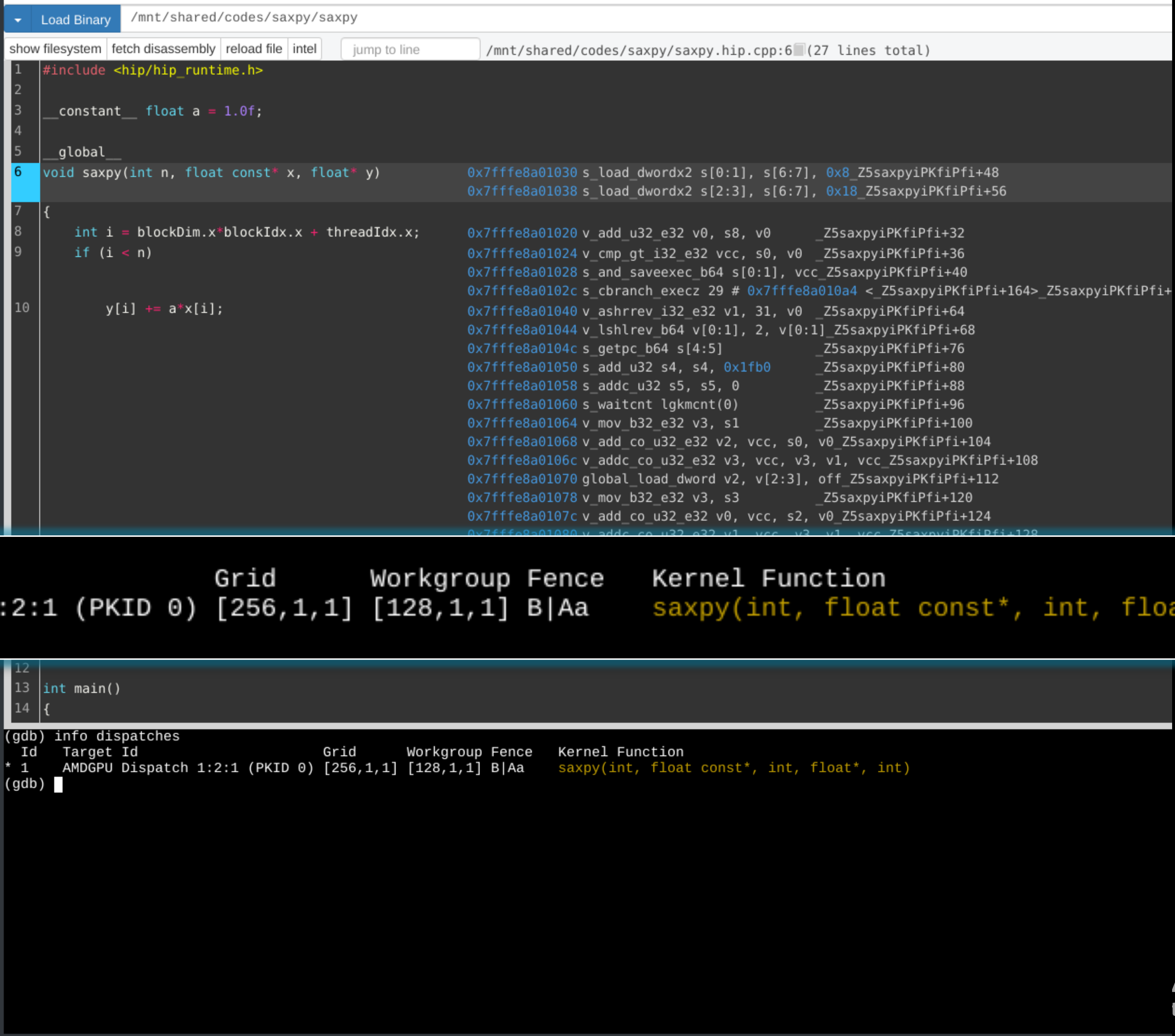
```
(gdb) info queues
  Id   Target Id                 Type         Read  Write  Size    Address
  1    AMDGPU Queue 1:1 (QID 0) HSA (Multi)  4     4      262144  0x00007ffff7f40000
* 2    AMDGPU Queue 1:2 (QID 1) HSA (Multi)  0     1      65536   0x00007ffff7fa0000
(gdb) █
```

AMD
together we advance_

list dispatches

info dispatches

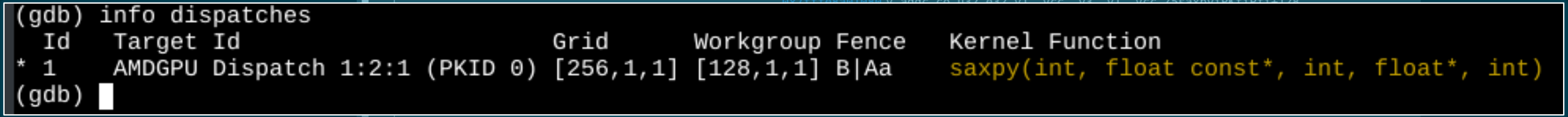
- ➢ shows kernel dispatches



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | intel | jump to line /mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)          0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
                                                     0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;    0x7fffe8a01020 v_add_u32_e32 v0, s8, v0      _Z5saxpyiPKfiPfi+32
    if (i < n)                                       0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0  _Z5saxpyiPKfiPfi+36
                                                     0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
                                                     0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164> _Z5saxpyiPKfiPfi+
        y[i] += a*x[i];                              0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0  _Z5saxpyiPKfiPfi+64
                                                     0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
                                                     0x7fffe8a0104c s_getpc_b64 s[4:5]            _Z5saxpyiPKfiPfi+76
                                                     0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0      _Z5saxpyiPKfiPfi+80
                                                     0x7fffe8a01058 s_addc_u32 s5, s5, 0          _Z5saxpyiPKfiPfi+88
                                                     0x7fffe8a01060 s_waitcnt lgkmcnt(0)          _Z5saxpyiPKfiPfi+96
                                                     0x7fffe8a01064 v_mov_b32_e32 v3, s1          _Z5saxpyiPKfiPfi+100
                                                     0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
                                                     0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
                                                     0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
                                                     0x7fffe8a01078 v_mov_b32_e32 v3, s3          _Z5saxpyiPKfiPfi+120
                                                     0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _Z5saxpyiPKfiPfi+124

int main()
{
```

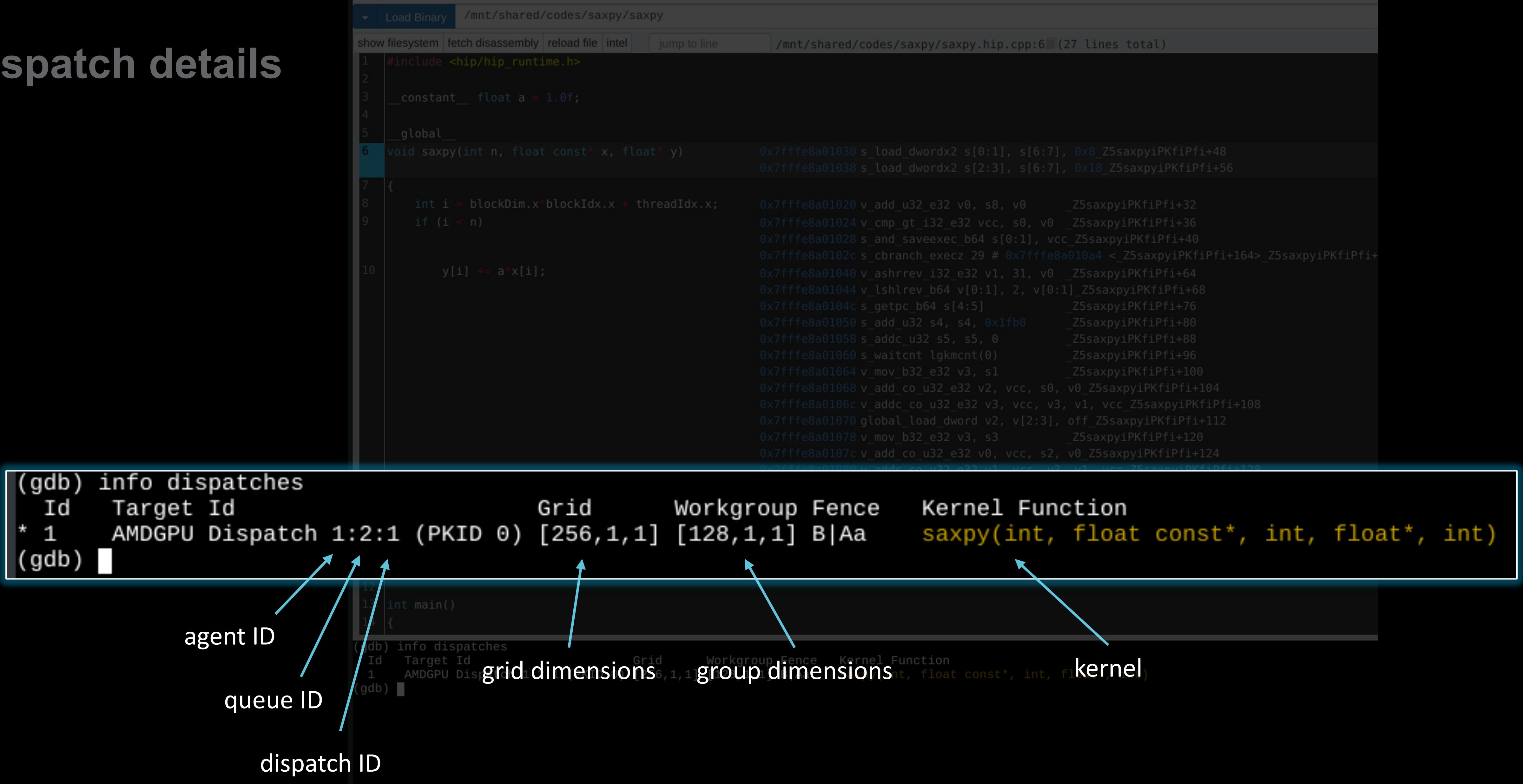


```
(gdb) info dispatches
  Id   Target Id                          Grid      Workgroup Fence   Kernel Function
* 1    AMDGPU Dispatch 1:2:1 (PKID 0) [256,1,1] [128,1,1] B|Aa    saxpy(int, float const*, int, float*, int)
(gdb)
```

```
(gdb) info dispatches
  Id   Target Id                          Grid      Workgroup Fence   Kernel Function
* 1    AMDGPU Dispatch 1:2:1 (PKID 0) [256,1,1] [128,1,1] B|Aa    saxpy(int, float const*, int, float*, int)
(gdb)
```

AMD together we advance_

dispatch details



```
/mnt/shared/codes/saxpy/saxpy
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;
    if (i < n)

        y[i] += a*x[i];

int main()
{
```

```
0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
0x7fffe8a01020 v_add_u32_e32 v0, s8, v0       _Z5saxpyiPKfiPfi+32
0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0   _Z5saxpyiPKfiPfi+36
0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164>_Z5saxpyiPKfiPfi+
0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0   _Z5saxpyiPKfiPfi+64
0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
0x7fffe8a0104c s_getpc_b64 s[4:5]             _Z5saxpyiPKfiPfi+76
0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0       _Z5saxpyiPKfiPfi+80
0x7fffe8a01058 s_addc_u32 s5, s5, 0           _Z5saxpyiPKfiPfi+88
0x7fffe8a01060 s_waitcnt lgkmcnt(0)           _Z5saxpyiPKfiPfi+96
0x7fffe8a01064 v_mov_b32_e32 v3, s1           _Z5saxpyiPKfiPfi+100
0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
0x7fffe8a01078 v_mov_b32_e32 v3, s3           _Z5saxpyiPKfiPfi+120
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0 _Z5saxpyiPKfiPfi+124
```

```
(gdb) info dispatches
  Id   Target Id                      Grid        Workgroup Fence    Kernel Function
* 1    AMDGPU Dispatch 1:2:1 (PKID 0) [256,1,1]  [128,1,1] B|Aa     saxpy(int, float const*, int, float*, int)
(gdb) █
```

AMD
together we advance_

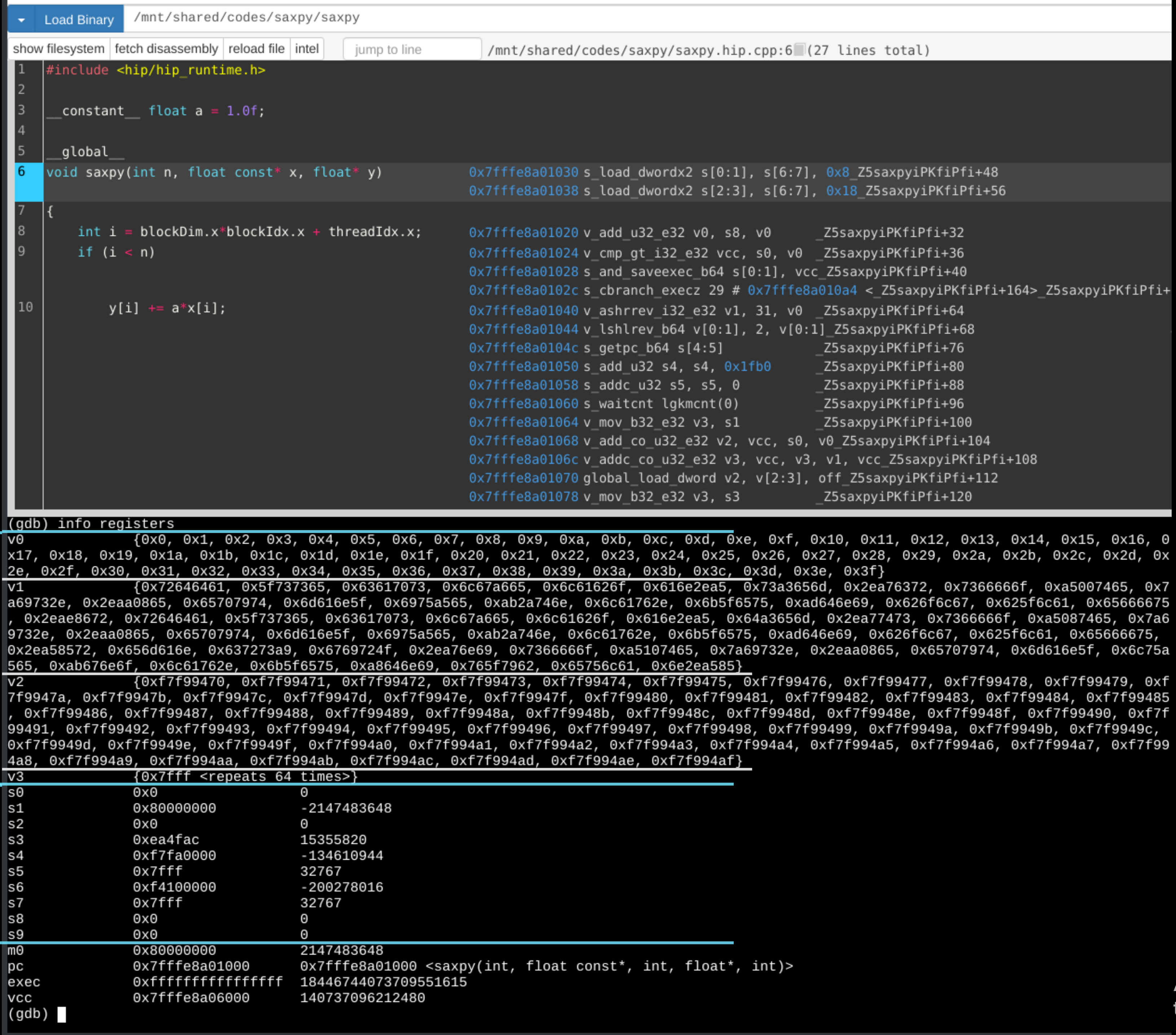
list registers

info registers
- shows registers in use

vector registers

scalar registers

program counter, exec mask, …



Load Binary /mnt/shared/codes/saxpy/saxpy

show filesystem | fetch disassembly | reload file | intel | jump to line | /mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)

```
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
void saxpy(int n, float const* x, float* y)        0x7fffe8a01030 s_load_dwordx2 s[0:1], s[6:7], 0x8 _Z5saxpyiPKfiPfi+48
                                                   0x7fffe8a01038 s_load_dwordx2 s[2:3], s[6:7], 0x18 _Z5saxpyiPKfiPfi+56
{
    int i = blockDim.x*blockIdx.x + threadIdx.x;   0x7fffe8a01020 v_add_u32_e32 v0, s8, v0      _Z5saxpyiPKfiPfi+32
    if (i < n)                                     0x7fffe8a01024 v_cmp_gt_i32_e32 vcc, s0, v0  _Z5saxpyiPKfiPfi+36
                                                   0x7fffe8a01028 s_and_saveexec_b64 s[0:1], vcc _Z5saxpyiPKfiPfi+40
                                                   0x7fffe8a0102c s_cbranch_execz 29 # 0x7fffe8a010a4 <_Z5saxpyiPKfiPfi+164>_Z5saxpyiPKfiPfi+
        y[i] += a*x[i];                            0x7fffe8a01040 v_ashrrev_i32_e32 v1, 31, v0  _Z5saxpyiPKfiPfi+64
                                                   0x7fffe8a01044 v_lshlrev_b64 v[0:1], 2, v[0:1] _Z5saxpyiPKfiPfi+68
                                                   0x7fffe8a0104c s_getpc_b64 s[4:5]            _Z5saxpyiPKfiPfi+76
                                                   0x7fffe8a01050 s_add_u32 s4, s4, 0x1fb0      _Z5saxpyiPKfiPfi+80
                                                   0x7fffe8a01058 s_addc_u32 s5, s5, 0          _Z5saxpyiPKfiPfi+88
                                                   0x7fffe8a01060 s_waitcnt lgkmcnt(0)          _Z5saxpyiPKfiPfi+96
                                                   0x7fffe8a01064 v_mov_b32_e32 v3, s1          _Z5saxpyiPKfiPfi+100
                                                   0x7fffe8a01068 v_add_co_u32_e32 v2, vcc, s0, v0 _Z5saxpyiPKfiPfi+104
                                                   0x7fffe8a0106c v_addc_co_u32_e32 v3, vcc, v3, v1, vcc _Z5saxpyiPKfiPfi+108
                                                   0x7fffe8a01070 global_load_dword v2, v[2:3], off _Z5saxpyiPKfiPfi+112
                                                   0x7fffe8a01078 v_mov_b32_e32 v3, s3          _Z5saxpyiPKfiPfi+120
```

```
(gdb) info registers
v0             {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7, 0x8, 0x9, 0xa, 0xb, 0xc, 0xd, 0xe, 0xf, 0x10, 0x11, 0x12, 0x13, 0x14, 0x15, 0x16, 0
x17, 0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2a, 0x2b, 0x2c, 0x2d, 0x
2e, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3a, 0x3b, 0x3c, 0x3d, 0x3e, 0x3f}
v1             {0x72646461, 0x5f737365, 0x63617073, 0x6c67a665, 0x6c61626f, 0x616e2ea5, 0x73a3656d, 0x2ea76372, 0x7366666f, 0xa5007465, 0x7
a69732e, 0x2eaa0865, 0x65707974, 0x6d616e5f, 0x6975a565, 0xab2a746e, 0x6c61762e, 0x6b5f6575, 0xad646e69, 0x626f6c67, 0x625f6c61, 0x65666675
, 0x2eae8672, 0x72646461, 0x5f737365, 0x63617073, 0x6c67a665, 0x6c61626f, 0x616e2ea5, 0x64a3656d, 0x2ea77473, 0x7366666f, 0xa5087465, 0x7a6
9732e, 0x2eaa0865, 0x65707974, 0x6d616e5f, 0x6975a565, 0xab2a746e, 0x6c61762e, 0x6b5f6575, 0xad646e69, 0x626f6c67, 0x625f6c61, 0x65666675,
0x2ea58572, 0x656d616e, 0x637273a9, 0x6769724f, 0x2ea76e69, 0x7366666f, 0xa5107465, 0x7a69732e, 0x2eaa0865, 0x65707974, 0x6d616e5f, 0x6c75a
565, 0xab676e6f, 0x6c61762e, 0x6b5f6575, 0xa8646e69, 0x765f7962, 0x65756c61, 0x6e2ea585}
v2             {0xf7f99470, 0xf7f99471, 0xf7f99472, 0xf7f99473, 0xf7f99474, 0xf7f99475, 0xf7f99476, 0xf7f99477, 0xf7f99478, 0xf7f99479, 0xf
7f9947a, 0xf7f9947b, 0xf7f9947c, 0xf7f9947d, 0xf7f9947e, 0xf7f9947f, 0xf7f99480, 0xf7f99481, 0xf7f99482, 0xf7f99483, 0xf7f99484, 0xf7f99485
, 0xf7f99486, 0xf7f99487, 0xf7f99488, 0xf7f99489, 0xf7f9948a, 0xf7f9948b, 0xf7f9948c, 0xf7f9948d, 0xf7f9948e, 0xf7f9948f, 0xf7f99490, 0xf7f
99491, 0xf7f99492, 0xf7f99493, 0xf7f99494, 0xf7f99495, 0xf7f99496, 0xf7f99497, 0xf7f99498, 0xf7f99499, 0xf7f9949a, 0xf7f9949b, 0xf7f9949c,
0xf7f9949d, 0xf7f9949e, 0xf7f9949f, 0xf7f994a0, 0xf7f994a1, 0xf7f994a2, 0xf7f994a3, 0xf7f994a4, 0xf7f994a5, 0xf7f994a6, 0xf7f994a7, 0xf7f99
4a8, 0xf7f994a9, 0xf7f994aa, 0xf7f994ab, 0xf7f994ac, 0xf7f994ad, 0xf7f994ae, 0xf7f994af}
v3             {0x7fff <repeats 64 times>}
s0             0x0                 0
s1             0x80000000          -2147483648
s2             0x0                 0
s3             0xea4fac            15355820
s4             0xf7fa0000          -134610944
s5             0x7fff              32767
s6             0xf4100000          -200278016
s7             0x7fff              32767
s8             0x0                 0
s9             0x0                 0
m0             0x80000000          2147483648
pc             0x7fffe8a01000      0x7fffe8a01000 <saxpy(int, float const*, int, float*, int)>
exec           0xffffffffffffffff  18446744073709551615
vcc            0x7fffe8a06000      140737096212480
(gdb) 
```

AMD
together we advance_

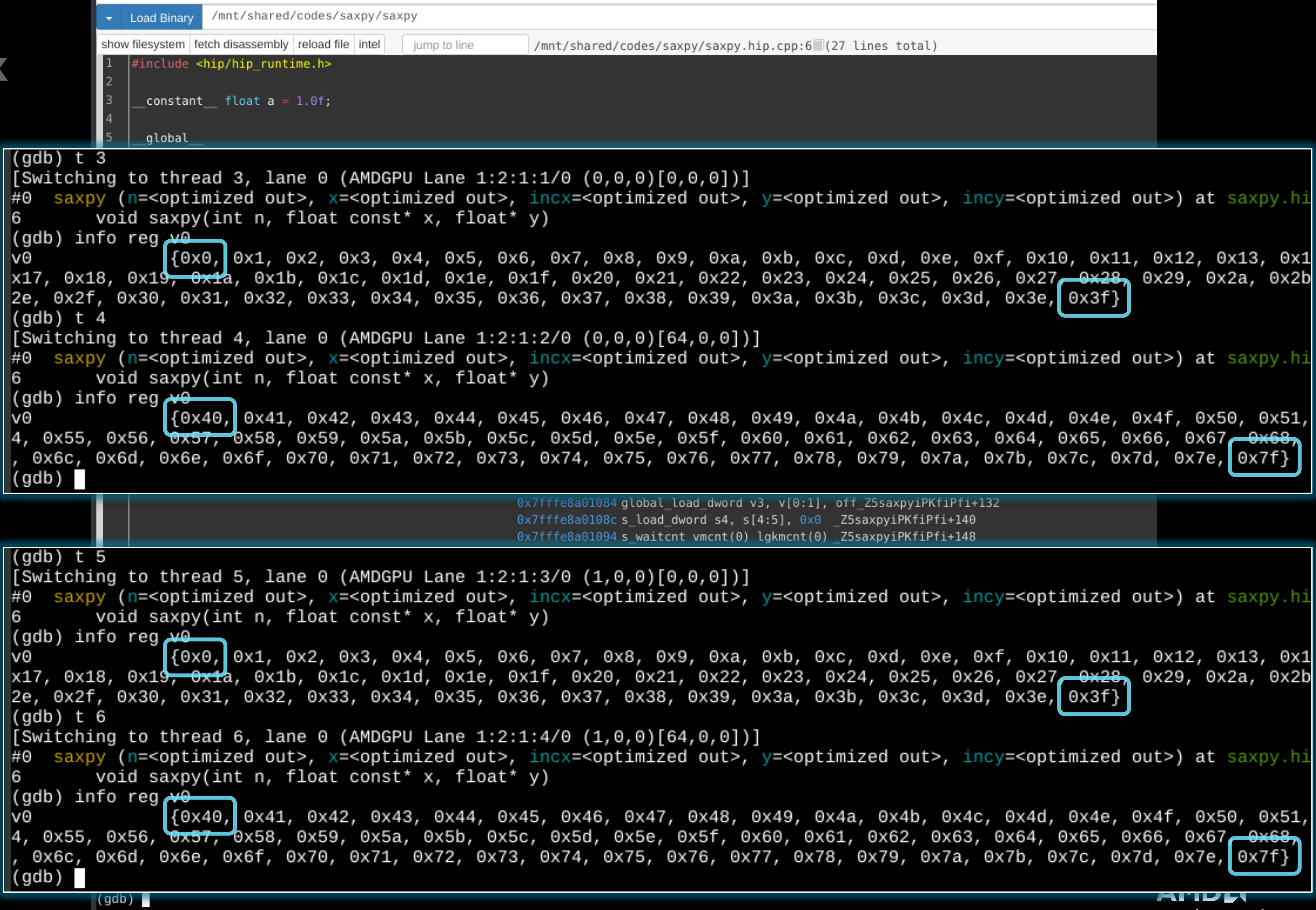
v0 = theadIdx.x

t 3
info reg v0
- ➢ values from 0 to 63

t 4
info reg v0
- ➢ values from 64 to 127

t 5
info reg v0
- ➢ values from 0 to 63

t 6
info reg v0
- ➢ values from 64 to 127



```
/mnt/shared/codes/saxpy/saxpy
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;

__global__
```

```
(gdb) t 3
[Switching to thread 3, lane 0 (AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, y=<optimized out>, incy=<optimized out>) at saxpy.hi
6           void saxpy(int n, float const* x, float* y)
(gdb) info reg v0
v0             {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7, 0x8, 0x9, 0xa, 0xb, 0xc, 0xd, 0xe, 0xf, 0x10, 0x11, 0x12, 0x13, 0x1
x17, 0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2a, 0x2b
2e, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3a, 0x3b, 0x3c, 0x3d, 0x3e, 0x3f}
(gdb) t 4
[Switching to thread 4, lane 0 (AMDGPU Lane 1:2:1:2/0 (0,0,0)[64,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, y=<optimized out>, incy=<optimized out>) at saxpy.hi
6           void saxpy(int n, float const* x, float* y)
(gdb) info reg v0
v0             {0x40, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4a, 0x4b, 0x4c, 0x4d, 0x4e, 0x4f, 0x50, 0x51,
4, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5a, 0x5b, 0x5c, 0x5d, 0x5e, 0x5f, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68
, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77, 0x78, 0x79, 0x7a, 0x7b, 0x7c, 0x7d, 0x7e, 0x7f}
(gdb)
```

```
0x7fffe8a01084 global_load_dword v3, v[0:1], off _Z5saxpyiPKfiPfi+132
0x7fffe8a0108c s_load_dword s4, s[4:5], 0x0  _Z5saxpyiPKfiPfi+140
0x7fffe8a01094 s_waitcnt vmcnt(0) lgkmcnt(0) _Z5saxpyiPKfiPfi+148
```

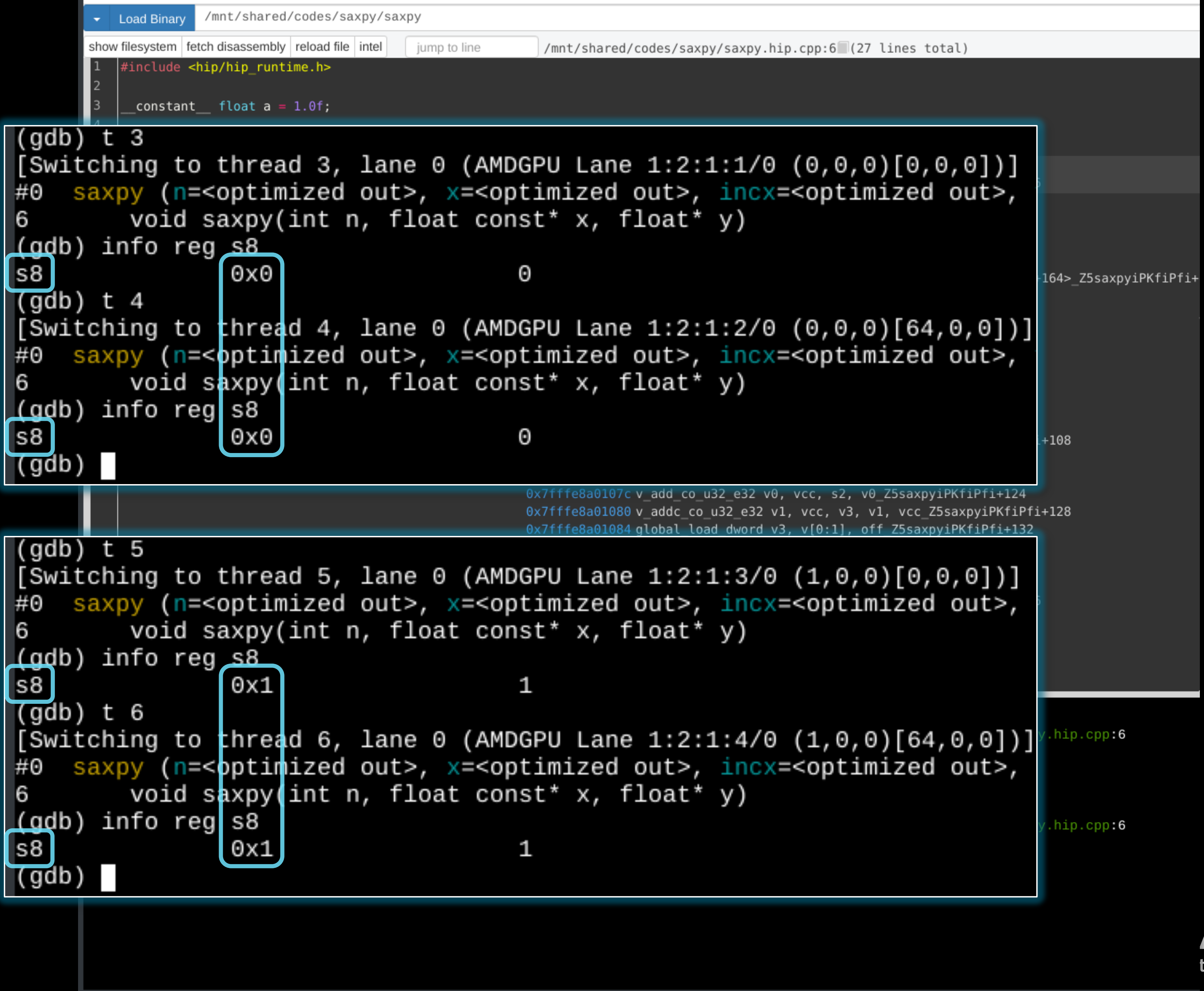
```
(gdb) t 5
[Switching to thread 5, lane 0 (AMDGPU Lane 1:2:1:3/0 (1,0,0)[0,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, y=<optimized out>, incy=<optimized out>) at saxpy.hi
6           void saxpy(int n, float const* x, float* y)
(gdb) info reg v0
v0             {0x0, 0x1, 0x2, 0x3, 0x4, 0x5, 0x6, 0x7, 0x8, 0x9, 0xa, 0xb, 0xc, 0xd, 0xe, 0xf, 0x10, 0x11, 0x12, 0x13, 0x1
x17, 0x18, 0x19, 0x1a, 0x1b, 0x1c, 0x1d, 0x1e, 0x1f, 0x20, 0x21, 0x22, 0x23, 0x24, 0x25, 0x26, 0x27, 0x28, 0x29, 0x2a, 0x2b
2e, 0x2f, 0x30, 0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38, 0x39, 0x3a, 0x3b, 0x3c, 0x3d, 0x3e, 0x3f}
(gdb) t 6
[Switching to thread 6, lane 0 (AMDGPU Lane 1:2:1:4/0 (1,0,0)[64,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>, y=<optimized out>, incy=<optimized out>) at saxpy.hi
6           void saxpy(int n, float const* x, float* y)
(gdb) info reg v0
v0             {0x40, 0x41, 0x42, 0x43, 0x44, 0x45, 0x46, 0x47, 0x48, 0x49, 0x4a, 0x4b, 0x4c, 0x4d, 0x4e, 0x4f, 0x50, 0x51,
4, 0x55, 0x56, 0x57, 0x58, 0x59, 0x5a, 0x5b, 0x5c, 0x5d, 0x5e, 0x5f, 0x60, 0x61, 0x62, 0x63, 0x64, 0x65, 0x66, 0x67, 0x68
, 0x6c, 0x6d, 0x6e, 0x6f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77, 0x78, 0x79, 0x7a, 0x7b, 0x7c, 0x7d, 0x7e, 0x7f}
(gdb)
```

```
(gdb)
```

s8 = blockIdx.x

t 3/4
info reg s8
➢ blockIdx.x = 0

t 5/6
info reg s8
➢ blockIdx.x = 1



```
/mnt/shared/codes/saxpy/saxpy
/mnt/shared/codes/saxpy/saxpy.hip.cpp:6 (27 lines total)
#include <hip/hip_runtime.h>

__constant__ float a = 1.0f;
```

```
(gdb) t 3
[Switching to thread 3, lane 0 (AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>,
6       void saxpy(int n, float const* x, float* y)
(gdb) info reg s8
s8             0x0                 0
(gdb) t 4
[Switching to thread 4, lane 0 (AMDGPU Lane 1:2:1:2/0 (0,0,0)[64,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>,
6       void saxpy(int n, float const* x, float* y)
(gdb) info reg s8
s8             0x0                 0
(gdb)
```

```
0x7fffe8a0107c v_add_co_u32_e32 v0, vcc, s2, v0_Z5saxpyiPKfiPfi+124
0x7fffe8a01080 v_addc_co_u32_e32 v1, vcc, v3, v1, vcc_Z5saxpyiPKfiPfi+128
0x7fffe8a01084 global_load_dword v3, v[0:1], off_Z5saxpyiPKfiPfi+132
```

```
(gdb) t 5
[Switching to thread 5, lane 0 (AMDGPU Lane 1:2:1:3/0 (1,0,0)[0,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>,
6       void saxpy(int n, float const* x, float* y)
(gdb) info reg s8
s8             0x1                 1
(gdb) t 6
[Switching to thread 6, lane 0 (AMDGPU Lane 1:2:1:4/0 (1,0,0)[64,0,0])]
#0  saxpy (n=<optimized out>, x=<optimized out>, incx=<optimized out>,
6       void saxpy(int n, float const* x, float* y)
(gdb) info reg s8
s8             0x1                 1
(gdb)
```

AMD
together we advance_

other things you can do

- inspect / modify registers
- inspect / modify memory
- inspect / modify LDS
- step through the assembly one instruction at a time

more info

- /opt/rocm<-version>/share/doc/rocgdb/
  - **rocgdb.pdf**
    - basically GDB manual +
      - section 20 “*Debugging Heterogeneous Programs*”
      - section 22.4.10 “*AMD GPU*”
  - rocrefcard.pdf

- https://www.olcf.ornl.gov/wp-content/uploads/2021/04/rocgdb_hipmath_ornl_2021_v2.pdf
  - ROCgdb presentation by Justin Chang
- https://lpc.events/event/11/contributions/997/attachments/928/1828/LPC2021-rocgdbdemo.pdf (https://youtu.be/IGWFph4SlpU?si=zxFMVAWG8JKVLowV)
  - debugging video by Andrew Stubbs

gdb cheat sheet

Start GDB (GNU Debugger)

gdb <program> [core dump]

gdb –args <program> <args>

gdb -help

Run commands

r[un] - Runs the program until a breakpoint or error

c[ontinue] - Continues running the program until the next breakpoint or error

q[uit] or kill - Quits gdb

fin[ish] - Runs until current function or loop is finished

n[ext] - Runs the next line of the program

- **n N** - Runs the next N lines of the program

s[tep] - Runs the next line of the program, stepping into any called routines

until N - Runs until you get N lines after the current line

Breakpoint commands

b[reakpoint] <where> – set breakpoint

- **b main** - Puts a breakpoint at the beginning of the program
- **b** - Puts a breakpoint at the current line
- **b N** - Puts a breakpoint at line N
- **b +N** - Puts a breakpoint N lines down from the current line
- **b fn** - Puts a breakpoint at the beginning of function "fn”
- **b/w <where> if <condition** – conditional breakpoint or watch

i[nfo] b[reak] - list breakpoints

dis[able] N - disable breakpoint number N

en[able] N – enables breakpoint number N

d[elete] N – delete breakpoint number N

clear – clear all breakpoints

Print commands

[h]elp <command>

[p]rint var - Prints the current value of the variable "var”

[l]ist – list lines

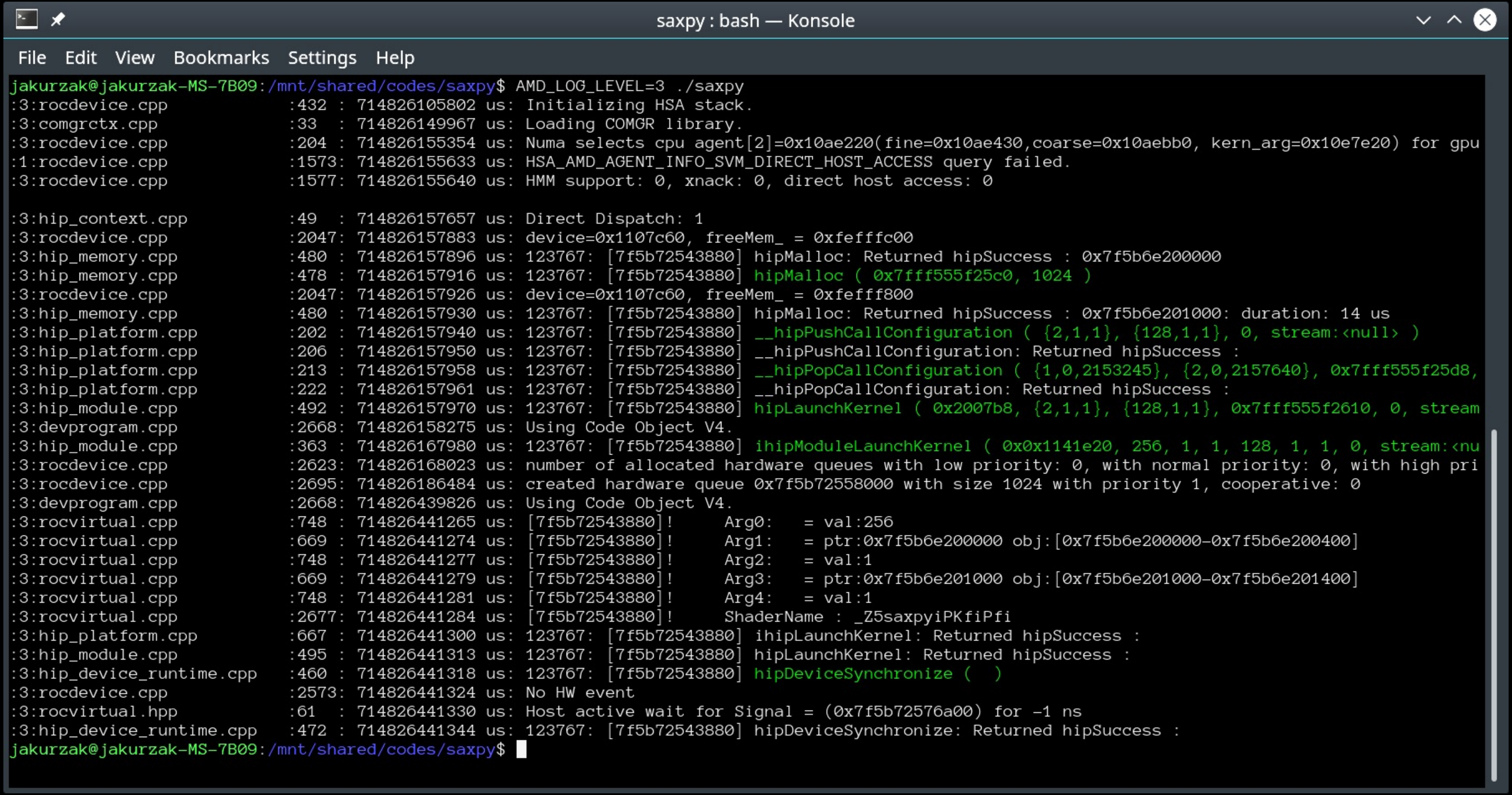
bt (backtrace) - Prints a stack trace

Movement

up - Goes up a level in the stack

[do]wn - Goes down a level in the stack

AMD_LOG_LEVEL=3



```
jakurzak@jakurzak-MS-7B09:/mnt/shared/codes/saxpy$ AMD_LOG_LEVEL=3 ./saxpy
:3:rocdevice.cpp            :432 : 714826105802 us: Initializing HSA stack.
:3:comgrctx.cpp             :33  : 714826149967 us: Loading COMGR library.
:3:rocdevice.cpp            :204 : 714826155354 us: Numa selects cpu agent[2]=0x10ae220(fine=0x10ae430,coarse=0x10aebb0, kern_arg=0x10e7e20) for gpu
:1:rocdevice.cpp            :1573: 714826155633 us: HSA_AMD_AGENT_INFO_SVM_DIRECT_HOST_ACCESS query failed.
:3:rocdevice.cpp            :1577: 714826155640 us: HMM support: 0, xnack: 0, direct host access: 0

:3:hip_context.cpp          :49  : 714826157657 us: Direct Dispatch: 1
:3:rocdevice.cpp            :2047: 714826157883 us: device=0x1107c60, freeMem_ = 0xfefffc00
:3:hip_memory.cpp           :480 : 714826157896 us: 123767: [7f5b72543880] hipMalloc: Returned hipSuccess : 0x7f5b6e200000
:3:hip_memory.cpp           :478 : 714826157916 us: 123767: [7f5b72543880] hipMalloc ( 0x7fff555f25c0, 1024 )
:3:rocdevice.cpp            :2047: 714826157926 us: device=0x1107c60, freeMem_ = 0xfefff800
:3:hip_memory.cpp           :480 : 714826157930 us: 123767: [7f5b72543880] hipMalloc: Returned hipSuccess : 0x7f5b6e201000: duration: 14 us
:3:hip_platform.cpp         :202 : 714826157940 us: 123767: [7f5b72543880] __hipPushCallConfiguration ( {2,1,1}, {128,1,1}, 0, stream:<null> )
:3:hip_platform.cpp         :206 : 714826157950 us: 123767: [7f5b72543880] __hipPushCallConfiguration: Returned hipSuccess :
:3:hip_platform.cpp         :213 : 714826157958 us: 123767: [7f5b72543880] __hipPopCallConfiguration ( {1,0,2153245}, {2,0,2157640}, 0x7fff555f25d8,
:3:hip_platform.cpp         :222 : 714826157961 us: 123767: [7f5b72543880] __hipPopCallConfiguration: Returned hipSuccess :
:3:hip_module.cpp           :492 : 714826157970 us: 123767: [7f5b72543880] hipLaunchKernel ( 0x2007b8, {2,1,1}, {128,1,1}, 0x7fff555f2610, 0, stream
:3:devprogram.cpp           :2668: 714826158275 us: Using Code Object V4.
:3:hip_module.cpp           :363 : 714826167980 us: 123767: [7f5b72543880] ihipModuleLaunchKernel ( 0x0x1141e20, 256, 1, 1, 128, 1, 1, 0, stream:<nu
:3:rocdevice.cpp            :2623: 714826168023 us: number of allocated hardware queues with low priority: 0, with normal priority: 0, with high pri
:3:rocdevice.cpp            :2695: 714826186484 us: created hardware queue 0x7f5b72558000 with size 1024 with priority 1, cooperative: 0
:3:devprogram.cpp           :2668: 714826439826 us: Using Code Object V4.
:3:rocvirtual.cpp           :748 : 714826441265 us: [7f5b72543880]!     Arg0:    = val:256
:3:rocvirtual.cpp           :669 : 714826441274 us: [7f5b72543880]!     Arg1:    = ptr:0x7f5b6e200000 obj:[0x7f5b6e200000-0x7f5b6e200400]
:3:rocvirtual.cpp           :748 : 714826441277 us: [7f5b72543880]!     Arg2:    = val:1
:3:rocvirtual.cpp           :669 : 714826441279 us: [7f5b72543880]!     Arg3:    = ptr:0x7f5b6e201000 obj:[0x7f5b6e201000-0x7f5b6e201400]
:3:rocvirtual.cpp           :748 : 714826441281 us: [7f5b72543880]!     Arg4:    = val:1
:3:rocvirtual.cpp           :2677: 714826441284 us: [7f5b72543880]!     ShaderName : _Z5saxpyiPKfiPfi
:3:hip_platform.cpp         :667 : 714826441300 us: 123767: [7f5b72543880] ihipLaunchKernel: Returned hipSuccess :
:3:hip_module.cpp           :495 : 714826441313 us: 123767: [7f5b72543880] hipLaunchKernel: Returned hipSuccess :
:3:hip_device_runtime.cpp   :460 : 714826441318 us: 123767: [7f5b72543880] hipDeviceSynchronize (  )
:3:rocdevice.cpp            :2573: 714826441324 us: No HW event
:3:rocvirtual.hpp           :61  : 714826441330 us: Host active wait for Signal = (0x7f5b72576a00) for -1 ns
:3:hip_device_runtime.cpp   :472 : 714826441344 us: 123767: [7f5b72543880] hipDeviceSynchronize: Returned hipSuccess :
jakurzak@jakurzak-MS-7B09:/mnt/shared/codes/saxpy$ 
```

how to use rocgdb + gdbgui + Chrome

test if X forwarding works

```
ssh -X USERNAME@home.ccs.ornl.gov
ssh -X login1._______.olcf.ornl.gov
srun -A VEN113 -N 1 -n 1 -c 64 --x11 --pty bash
xmessage -center hello!
```

install gdbgui

```
python3 -m pip install --user pipx
python3 -m userpath append ~/.local/bin
pipx install gdbgui
```

install Chrome

- Go to https://www.google.com/chrome/
- Click *Download Chrome*
- Click *64 bit .rpm (For Fedora/openSUSE)*
- Click *Accept and Install*

```
scp google-chrome-stable_current_x86_64.rpm USERNAME@home.ccs.ornl.gov:
ssh -X USERNAME@home.ccs.ornl.gov
mkdir ~/chrome
cd ~/chrome
rpm2cpio ../google-chrome-stable_current_x86_64.rpm | cpio -id
```

run rocgdb with gdbgui in Chrome

```
ssh -X USERNAME@home.ccs.ornl.gov
ssh -X login1._______.olcf.ornl.gov
srun -A VEN113 -N 1 -n 1 -c 64 --x11 --pty bash
gdbgui -g /opt/rocm/bin/rocgdb --no-browser &
~/chrome/opt/google/chrome/google-chrome 2>/dev/null &
```

- In Chrome, go to: http://127.0.0.1:5000
- Click *Load Binary* to load your binary (compiled with -ggdb)
- Step into a kernel
- Click *fetch disassembly*

```
show architecture
info threads
info queues
info dispatches
info registers
info reg vcc
info reg exec
s
si
n
ni
...
```

AMD
together we advance_

DISCLAIMERS

AMD
together we advance_

ATTRIBUTIONS

AMD