

HIP Training Workshop – Day 3

ROCgdb & HIP math libraries

Justin Chang

1 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved



Part 1: Intro to ROCgdb

2 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved

Getting started

What is ROCgdb, from the tin:

The ROCm Debugger (ROCgdb) is the ROCm source-level debugger for Linux, based on the GNU Debugger (GDB). It enables heterogenous debugging on the ROCm platform of an x86-based host architecture along with AMD GPU architectures supported by the AMD Debugger API Library (ROCdbgapi). The AMD Debugger API Library (ROCdbgapi) is included with the ROCm release.

The current ROCm Debugger (ROCgdb) is an initial prototype that focuses on source line debugging and does not provide symbolic variable debugging capabilities. The user guide presents features and commands that may be implemented in future versions.

So... cuda-gdb? Yes, and mostly no -- rocgdb *is* (or will be) gdb, that is it tracks upstream GDB master.

What can it do?

In addition to your usual host-debugging capabilities, a very brief overview of rocgdb's current functionality:

- Switching between and seeing info about wavefronts
- Read/write to hardware registers, global memory, and LDS/scratch
- Breakpoints
- Watchpoints
- ISA-level debugging, mapping of ISA to source lines

Before we can take a test drive, let's talk about the code we're going to use it on.

Jacobi Example

Our example code:

- Solves a Laplacian problem
- 5-point finite difference stencil
- May be distributed over multiple MPI ranks
- Performs several iteration of a Jacobi method

Laplacian equation

Laplace equation:

$$-\nabla^2 q = f$$

In two dimensions, with a 5-point finite difference stencil we discretize as:

$$\frac{-q_{i-1,j} + 2q_{i,j} - q_{i+1,j}}{\Delta x^2} + \frac{-q_{i,j-1} + 2q_{i,j} - q_{i,j+1}}{\Delta y^2} = f_{i,j}$$

On a rectangular lattice one point in the x-y plane:



6 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved

Jacobi Example

 The finite difference Laplacian operator is a sparse matrix operator on a vector of unknowns:

Aq = f

- When the whole domain is distributed with MPI, each process will need to receive halo data to evaluate Aq
- Can compute Laplacian at interior stencil points while the halo data is being exchanged
- Jacobi Iterative method is the iteration:

 $\boldsymbol{q}^{k+1} = \boldsymbol{q}^k + D^{-1} (\boldsymbol{f} - A \boldsymbol{q}^k)$

where D is the diagonal of A.



Jacobi Structure

- Initialize MPI
- Setup Jacobi object
 - ApplyTopology()
 - CreateMesh()
 - InitializeData()
- Execute Jacobi run method

2 //* Copyright (c) 2019, Advanced Micro Devices, Inc. All rights reserved. **** 5 #include "Jacobi.hpp" int main(int argc, char ** argv) 8 MPI_Init(&argc, &argv); 10 MPI_Comm comm = MPI_COMM_WORLD; 12 13 grid_t grid; mesh_t mesh; 15 16 // Extract topology and domain dimensions from the command-line arguments ParseCommandLineArguments(argc, argv, comm, grid, mesh); Jacobi_t Jacobi(grid, mesh); 23 Jacobi.Run(); 25 26 // Finalize the MPI process MPI_Finalize(); return STATUS_OK; 28 29 }

Jacobi Structure

- Initialize MPI
- Setup Jacobi object
 - ApplyTopology()
 - Establish MPI neighbors
 - Set GPU devices
 - CreateMesh()
 - InitializeData()
- Execute Jacobi run method

```
//select GPU
    int devCount = 0;
    SafeHipCall(hipGetDeviceCount(&devCount));
    if (devCount == 0) {
      fprintf(stderr, "Error: MPI rank %d detected no GPUs."
                      "Terminating...\n", grid.rank);
      MPI_Abort(grid.comm, STATUS_ERR);
    char name[255];
   gethostname(name, 255);
    long int hostId = gethostid();
    long int* hostIds = (long int*) calloc(size, sizeof(long int));
   MPI_Allgather(&hostId,1,MPI_LONG,hostIds,1,MPI_LONG,grid.comm);
76
   //count how many ranks are on this node
    int local_rank = 0;
    for (int r=0;r<grid.rank;r++)</pre>
     if (hostIds[r]==hostId) local_rank++;
   //select gpu via round-robin
   grid.device_id = local_rank % devCount;
    SafeHipCall(hipSetDevice(grid.device_id));
```

Jacobi Structure

- Initialize MPI
- Setup Jacobi object
 - ApplyTopology()
 - Establish MPI neighbors
 - Set GPU devices
 - CreateMesh()
 - Set local domain
 - Create halo exchange buffers
 - Create streams
 - InitializeData()
- Execute Jacobi run method

//coordinates (including boundary points) 114 mesh.x = (dfloat *) malloc((mesh.Nx+2)*sizeof(dfloat)); 115 116 mesh.y = (dfloat *) malloc((mesh.Ny+2)*sizeof(dfloat)); 117 118 for (int i=0;i<mesh.Nx;i++)</pre> mesh.x[i] = mesh.dx + grid.mycol*(DX+mesh.dx) + i*mesh.dx; 119 120 for (int j=0;j<mesh.Ny;j++)</pre> 121 mesh.y[j] = mesh.dy + grid.myrow*(DY+mesh.dy) + j*mesh.dy; 122 123 //halo exchange storage SafeHipCall(hipHostMalloc(&(mesh.sendBuffer), mesh.Nhalo*sizeof(dfloat),0)); 124 SafeHipCall(hipHostMalloc(&(mesh.recvBuffer), mesh.Nhalo*sizeof(dfloat),0)); 125 SafeHipCall(hipMalloc(&(mesh.d_haloBuffer), mesh.Nhalo*sizeof(dfloat))); 126 SafeHipCall(hipMemset(mesh.d_haloBuffer, 0, mesh.Nhalo*sizeof(dfloat))); 127 128 for (int i=0;i<2*NSIDES;i++) mesh.requests[i] = MPI_REQUEST_NULL;</pre> 129 130 131 //number of entries for each side in send/recv buffers 132 mesh.sideLength[SIDE_DOWN] = mesh.Nx; 133 mesh.sideLength[SIDE_UP] = mesh.Nx; 134 mesh.sideLength[SIDE_LEFT] = mesh.Ny; 135 mesh.sideLength[SIDE_RIGHT] = mesh.Ny; 136 137 //offets of halo side data in send/recv buffers mesh.sideOffset[SIDE_DOWN] = 0; 138 139 mesh.sideOffset[SIDE_UP] = mesh.Nx; mesh.sideOffset[SIDE_LEFT] = 2*mesh.Nx; 140 mesh.sideOffset[SIDE_RIGHT] = 2*mesh.Nx + mesh.Ny; 141 L42 SafeHipCall(hipStreamCreate(&(computeStream))); SafeHipCall(hipStreamCreate(&(dataStream)));

Jacobi Structure

- Initialize MPI
- Setup Jacobi object
 - ApplyTopology()
 - Establish MPI neighbors
 - Set GPU devices
 - CreateMesh()
 - Set local domain
 - Create halo exchange buffers
 - Create streams
 - InitializeData()
 - Generate initial data
 - Copy data to device
- Execute Jacobi run method

153 //host buffers 154 h_U = (dfloat*) malloc(mesh.N*sizeof(dfloat)); 155 h_AU = (dfloat*) malloc(mesh.N*sizeof(dfloat)); 156 h_RHS = (dfloat*) malloc(mesh.N*sizeof(dfloat)); 157 h_RES = (dfloat*) malloc(mesh.N*sizeof(dfloat)); 158 159 for (int j=0;j<mesh.Ny;j++) { 160 for (int i=0;i<mesh.Nx;i++) { 161 int id = i+j*mesh.Nx; 162 h_U[id] = 0.0; //initial guess 163 h_RHS[id] = 0.0; //forcing 164 } 165 }

// Setup boundary contributions

```
:
```

9	//device buffers
0	SafeHipCall(hipMalloc((void **)&d_U, mesh.N*sizeof(dfloat)));
1	<pre>SafeHipCall(hipMalloc((void **)&d_AU, mesh.N*sizeof(dfloat)));</pre>
2	<pre>SafeHipCall(hipMalloc((void **)&d_RHS, mesh.N*sizeof(dfloat)));</pre>
3	SafeHipCall(hipMalloc((void **)&d_RES, mesh.N*sizeof(dfloat)));
4	
5	//send data to device
6	SafeHipCall(hipMemcpy(d_U, h_U, mesh.N* <mark>sizeof</mark> (dfloat), hipMemcpyHostToDevice));
7	SafeHipCall(hipMemcpy(d_AU, h_U, mesh.N*sizeof(dfloat), hipMemcpyHostToDevice));
8	<pre>SafeHipCall(hipMemcpy(d_RHS, h_RHS, mesh.N*sizeof(dfloat), hipMemcpyHostToDevice));</pre>
9	<pre>SafeHipCall(hipMemcpy(d_RES, h_RHS, mesh.N*sizeof(dfloat), hipMemcpyHostToDevice));</pre>
0	
1	Safallin Call (hin Event Cheat of Blacobil ocal Stant));

1 SafeHipCall(hipEventCreate(&JacobiLocalStart));

2 SafeHipCall(hipEventCreate(&JacobiLocalEnd))

Iterative Loop

while(){ LocalLaplacian() HaloExchange() HaloLaplacian() JacobiIteration() Norm()

Note:

}

- Two streams: computeStream and dataStream
- hipEventRecord timings on each respective stream (omitted for clarity)

26 27 28 29		<pre>hile_((iterations < JACOBI_MAX_LOOPS) && (residual > JACOBI_TOLERANCE)) { //queue local part of Laplacian to compute stream LocalLaplacian(grid, mesh, computeStream, d_U, d_AU); }</pre>
30		//Extract data off GPU exchange Halo with MPI
31		HaloExchange(arid, mesh, dataStream, d_U);
32		
33		//use halo data to complete Laplacian computation
34		HaloLaplacian(grid, mesh, computeStream, d_U, d_AU);
35		
36		//Jacobi iterative method
37		$// U = U + D^{-1}*(RHS - AU)$
38		JacobiIteration(grid, mesh, computeStream, d_RHS, d_AU, d_RES, d_U);
39		
40		<pre>//residual = U </pre>
41		residual = Norm(grid, mesh, computeStream, d_RES);
42		
43		//finish everything on device
44		hipDeviceSynchronize();
45		
46		++iterations;
47	- 7	

Preparing the code for the debugger

Preparing your HIP code for debugging:

- Use any optimization level you like, we'll use -O3
- Have ROCm load code objects at initialization:
 - export HIP ENABLE DEFERRED LOADING=0
- Add –ggdb to your flags:
- Optionally print even more useful information on API calls
 - export AMD_LOG_LEVEL=3

Example of what the compile options may look like...

```
mpic++ -I/usr/lib/x86_64-linux-gnu/openmpi/include/openmpi \
-L/usr/lib/x86_64-linux-gnu/openmpi/include -pthread -O3 -g -ggdb -fPIC \
-std=c++11 \ -march=native -Wall -I/opt/rocm/roctracer/include \
-I"/opt/rocm-4.2.0/hip/include" -I"/opt/rocm/llvm/bin/../lib/clang/12.0.0" \
-I/opt/rocm/hsa/include -I/opt/rocm/roctracer/include \
-c JacobiSetup.cpp -o JacobiSetup.o
```

Diving in

Launching the debugger, is the same as gdb:

```
rocgdb --args ./Jacobi_hip -g 1 1
```

For this demo, I will be using a program called <u>cgdb</u> alongside rocgdb, this gives a nice curses-based interface, but is by no means required. In this case:

cgdb -d rocgdb --args ./Jacobi hip -g 1 1

```
Copyright (c) 2019, Advanced Micro Devices, Inc. All rights reserved.
   #include "Jacobi.hpp"
7
    * @file JacobiMain.cpp
9
10
    * Obrief This contains the application entry point
11
12
13
    * @brief The application entry point
14
     * @param[in] argc The number of command-line arguments
15
16
    * @param[in] argv The list of command-line arguments
17
18
    int main(int argc, char ** argv)
19
20
     MPI_Init(&argc, &argv);
21
22
     MPI_Comm comm = MPI_COMM_WORLD;
23
24
     grid_t grid;
25
     mesh_t mesh;
26
27
     // Extract topology and domain dimensions from the command-line arguments
/home/nick/Documents/software_repos/hiptutorial/hip/JacobiMain.cpp
GNU gdb (rocm-rel-3.5-30) 9.1
Copyright (C) 2020 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 ./Jacobi hip...
(gdb) s
```

Setting a breakpoint

Let's step into one of our kernels, LocalLaplacianKernel computes the finite difference for one of our MPI domains.





Setting a breakpoint in host code

Here we setup a breakpoint in the host code. We can inspect the device pointer and its values:

F	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip Q = - 0 8
23 24 25 26 27	<pre>const int id_l = id - 1; const int id_r = id + 1; const int id_d = id - stride; const int id_u = id + stride;</pre>	For help, type "help". Type "apropos word" to search for commands related to "word" Reading symbols from ./Jacobi_hip (gdb) b Laplacian.cpp: 40
29 30 31 32 33	AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) + (-U[id_d] + 2*U[id] - U[id_u])/(dy*dy); } }	Breakpoint 1 at 0x415080: file Laplacian.cpp, line 40.(gdb) run Starting program: /home/jychang48/Downloads/hiptutorial/nip/lacobi_hip -g 1 1 [Thread debugging using libthread_db enabled] Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1". [Detaching after fork from child process 129946]
34 35 36 37 38	<pre>void LocalLaplacian(grid_t& grid, mesh_t& mesh,</pre>	[New Thread 0x7f2900108700 (LWP 129953)] [New Thread 0x7f28ff74a700 (LWP 129954)] Topology size: 1 x 1 Local domain size (current node): 4096 x 4096 Global domain size (all nodes): 4096 x 4096
39 40 41 42 43	<pre>//there are (Nx-2)x(Ny-2) node on the interior of the mesh > int localNx = mesh.Nx-2; int localNy = mesh.Ny-2; int xthreads = 16;</pre>	[New Thread 0x7f28f7f53700 (LWP 129955)] Rank 0 selecting device 0 on host jychang48-workstation [New Thread 0x7f28f71ff700 (LWP 129956)] [Thread 0x7f28f71ff700 (LWP 129956) exited] [New Thread 0x7f28fc07f700 (LWP 129957)]
44 45 46 47	<pre>int ythreads = 16; dim3 threads(xthreads,ythreads,1); dim3 blocks((localNx+xthreads-1)/xthreads,</pre>	[New Thread 0x7f28f7752700 (LWP 129958)] [New Thread 0x7f28f757f700 (LWP 129959)] Starting Jacobi run. Iteration: 0 - Residual: 0.022108
48 49 50 51	<pre>(localNy+ythreads-1)/ythreads, 1); hipLaunchKernelGGL(LocalLaplacianKernel,</pre>	Thread-1 Jacobi_hip hit_Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0 x69a6e0, d_U=0x7f27a7e00000, d_4U=0x7f279fc000000) at Laplacian.cpp:40 (gdb) p d_U
52 53 54 55	threads, 0, stream, localNx, localNy, mesh.Nx, mesh.dx, mesh.dy, d.W.d.AW);	<pre>\$1 = (double *) 0x7f27a7e00000 (gdb) p d_U[0] \$2 = 0 (gdb) p d_U[0]@10 *2 = [0 - 0 - 0 - 0 - 0 - 0]</pre>
/home	<pre>2/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp</pre>	ss = {0, 0, 0, 0, 0, 0, 0, 0, 0} (gdb)

Setting a breakpoint in host code

Typing 'step' enables one to iterate through the stack trace:

F	jychang48@jychang48-workstati	ion: ~/Downloads/hiptutorial/hip
33 34 35	<pre>void LocalLaplacian(grid_t& grid, mesh_t& mesh, hipStream_t stream,</pre>	Type "apropos word" to search for commands related to "word" Reading symbols from ./Jacobi_hip (gdb) b Laplacian.cpp: 40
36 37 38	dfloat* d_U, dfloat* d_AU) {	Breakpoint 1 at 0x415080: file Laplacian.cpp, line 40.(gdb) run Starting program: /home/ivchang48/Downloads/hiptutorial/hip/Jacobi hip -g 1 1
39 40	<pre>//there are (Nx-2)x(Ny-2) node on the interior of the mesh int localNx = mesh.Nx-2;</pre>	[Thread debugging using libthread_db enabled] Using host libthread db library "/lib/x86 64-linux-gnu/libthread db.so.1".
41 42	<pre>int localNy = mesh.Ny-2;</pre>	[Detaching after fork from child process 129946] [New Thread 0x7f2900108700 (LWP 129953)]
43 44	<pre>int xthreads = 16; int ythreads = 16;</pre>	[New Thread 0x7f28ff74a700 (LWP 129954)] Topology size: 1 x 1
45 46	<pre>dim3 threads(xthreads,ythreads,1); dim2 blacks(lace)Ny wthreads 1)(ythreads</pre>	Local domain size (current node): 4096 x 4096 Global domain size (all nodes): 4096 x 4096
47 48 49	(localNy+ythreads-1)/ythreads, 1);	[New Thread 0x712617155700 (LWP 129955)] Rank 0 selecting device 0 on host jychang48-workstation [New Thread 0x7f28f71ff700 (LWP 129956)]
50	hipLaunchKernelGGL(LocalLaplacianKernel,	[Thread 0x7f28f71ff700 (LWP 129956) exited]
51 52	threads,	[New Thread 0x7f28f7752700 (LWP 129957)] [New Thread 0x7f28f7752700 (LWP 129958)]
53 54	U, stream, localNx, localNy, mesh.Nx,	[New Inread 0x/1281/5/1/00 (LWP 129959)] Starting Jacobi run.
55 56	mesh.dx, mesh.dy, d_U, d_AU);	Iteration: 0 - Residual: 0.022108
57 58	}	Thread 1 "Jacobi_hip" hit Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0 x69a6e0, d_U=0x7f27a7e00000, d_AU=0x7f279fc00000) at Laplacian.cpp:40
59 60	// $AU_i, j = (-U_i+1, j + 2U_i, j - U_i-1, j)/dx^2 +$	(gdb) p d_U \$1 = (double *) 0x7f27a7e00000
61 62	// (-U_1,]+1 + 2U_1,] - U_1,]-1)/dy^2 global void HaloLaplacianKernel(const int Nx,	(gdb) p d_U[0] \$2 = 0
63 64	const int Ny, const int stride,	(gdb) p d U[0]@10 \$3 - {0, 0, 0, 0, 0, 0, 0, 0}
65 66	const dfloat dx, const dfloat dy,	(jdb) step (jdb) step
/hom	e/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	(ցեխ)

17 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved

Setting a breakpoint in device kernel

Invoke 'b' or 'break' to the device kernel of interest:

F	jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip 🛛 🔤 – 🗉 😣
1 2 3	//************************************	Topology size: 1 x 1 Local domain size (current node): 4096 x 4096 Global domain size (all nodes): 4096 x 4096
4 5 6	<pre>#include "Jacobi.hpp"</pre>	[New Thread 0x7f28f7f53700 (LWP 129955)] Rank 0 selecting device 0 on host jychang48-workstation [New Thread 0x7f28f71ff700 (LWP 129956)]
7	// AU_i,j = (-U_i+1,j + 2U_i,j - U_i-1,j)/dx^2 + (-U_i,i+1 + 2U_i,j - U_i,j-1)/dv^2	[Thread 0x7f28f71ff700 (LWP 129956) exited] [New Thread 0x7f28fc07f700 (LWP 129957)]
9	-> glopal void LocalLaplacianKernel(const int localNx,	[New Thread 0x7f28f7752700 (LWP 129958)]
10	const int localNy,	[New Thread 0x7f28f757f700 (LWP 129959)]
11	const int stride,	Starting Jacobi run.
12	const dfloat dx,	Iteration: 0 - Residual: 0.022108
13	const dfloat * restrict U	Thread 1 "Jacobi bin" bit Breaknoint 1 JocallanJacian (grid- mesh- stream-A
15	dfloat * restrict AU) {	x_{69a6e0} , d $II=0x_{7}f_{27a}^{2}e_{00000}$, d $AII=0x_{7}f_{279}f_{c}_{00000}$ at Laplacian cpp:40
16		(qdb) p d U
17	<pre>const int i = threadIdx.x+blockIdx.x*blockDim.x;</pre>	\$1 = (double *) 0x7f27a7e00000
18	<pre>const int j = threadIdx.y+blockIdx.y*blockDim.y;</pre>	(gdb) p d_U[0]
19		\$2 = 0
20	וֹל ((ו <localnx) &&="" (j<localny))="" td="" {<=""><td></td></localnx)>	
21	const int id = $(i,1)$, $(i,1)$ *strido:	\$3 = {0, 0, 0, 0, 0, 0, 0, 0, 0}
22	$const int id = (i+i) + (j+i) \cdot stride,$	(gdb) step
24	const int id l = id - 1:	(gdb) b LocalLaplacianKernel
25	const int id r = id + 1;	Function LocalLaplacianKernel" not defined.
26	<pre>const int id_d = id - stride;</pre>	Make breakpoint pending on future shared library load? (y or [n]) y
27	<pre>const int id_u = id + stride;</pre>	
28		Breakpoint 2 (LocalLaplacianKernel) pending.(gdo) continue
29	$AU[10] = (-U[10_{l}] + 2^{*}U[10] - U[10_{r}])/(dx^{*}dx) + (U[10_{r}] + 2^{*}U[10] - U[10_{r}])/(dx^{*}dx)) + (U[10_{r}] + 2^{*}U[10] - U[10_{r}])/(dx^{*}dx$	Continuing.
30		
32	}	Thread 9 "Jacobi hip" hit Breakpoint 2. LocalLaplacianKernel (localNx= <optimized out<="" td=""></optimized>
33		>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimized< td=""></optimized<></optimized></optimized></optimized>
34	<pre>void LocalLaplacian(grid_t& grid, mesh_t& mesh,</pre>	d out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
/hon	ne/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	(gdb)

What happens when you type 'step'? Another thread hit the same breakpoint! GDB will switch context to the new thread:

F	jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip 🛛 🔤 – 🗉 😣
1 2 3 4 5 6 7 8 9 10	<pre>//***********************************</pre>	<pre>\$2 = 0 (gdb) p d U[0]@10 \$3 = {0, 0, 0, 0, 0, 0, 0, 0, 0} (gdb) step (gdb) step (gdb) b LocalLaplacianKernel Function "LocalLaplacianKernel" not defined. Make breakpoint pending on future shared library load? (y or [n]) y Breakpoint 2 (localLaplacianKernel) pending (gdb) continue</pre>
10 11 12 13 14 15 16 17 18	<pre>const int totativy, const int stride, const dfloat dx, const dfloat dy, const dfloat *restrict U, dfloat *restrict AU) { const int i = threadIdx.x+blockIdx.x*blockDim.x; const int j = threadIdx.y+blockIdx.y*blockDim.y;</pre>	Continuing. [Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0] Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out<br="">>, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimized d out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1287 (65,1,0)/2]</optimized></optimized></optimized </optimized></optimized></optimized></optimized>
20 21 22 23 24 25	<pre>if ((i<localnx) &&="" (j+1)*stride;="" (j<localny))="" +="" -="" 1;="" 1;<="" const="" id="" id_l="id" int="" pre="" r="id" {=""></localnx)></pre>	Thread 1295 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out>, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim ized out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1269 (61,1,0)/0]</optimized></optimized></optim </optimized></optimized></optimized></optimized
26 27 28 29 30 31 32	<pre>const int id_d = id - stride; const int id_u = id + stride; AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +</pre>	Thread 1277 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out>, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim ized out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:363 (90,0,0)/2] Thread 371 "Jacobi hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized o<="" td=""></optimized></optimized></optimized></optim </optimized></optimized></optimized></optimized
33 34 <mark>/ho</mark> m	void LocalLaplacian(grid_t& grid, mesh_t& mesh, me/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	ut>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimi zed out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb)</optimized></optimized></optimi </optimized></optimized></optimized>



Setting a breakpoint in device kernel

[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]

Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out >, localNy=<optimized out>, stride=<optimized out>, dx=<optimized out>, dy=<optimize d out>, U=<optimized out>, AU=<optimized out>) at Laplacian.cpp:9

Setting a breakpoint in device kernel

• agent-id = Agent Target ID

[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]

Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out >, localNy=<optimized out>, stride=<optimized out>, dx=<optimized out>, dy=<optimize d out>, U=<optimized out>, AU=<optimized out>) at Laplacian.cpp:9

Setting a breakpoint in device kernel

- agent-id = Agent Target ID
- queue-id = Queue Target ID

[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]

Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out >, localNy=<optimized out>, stride=<optimized out>, dx=<optimized out>, dy=<optimize d out>, U=<optimized out>, AU=<optimized out>) at Laplacian.cpp:9

Setting a breakpoint in device kernel

- agent-id = Agent Target ID
- queue-id = Queue Target ID
- dispatch-num = Dispatch Target ID how many kernels have been launched

[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]

Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out >, localNy=<optimized out>, stride=<optimized out>, dx=<optimized out>, dy=<optimize d out>, U=<optimized out>, AU=<optimized out>) at Laplacian.cpp:9

Setting a breakpoint in device kernel

- agent-id = Agent Target ID
- queue-id = Queue Target ID
- dispatch-num = Dispatch Target ID how many kernels have been launched
- wave-id = Wavefront ID index of wavefront of kernel

[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]

Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out >, localNy=<optimized out>, stride=<optimized out>, dx=<optimized out>, dy=<optimize d out>, U=<optimized out>, AU=<optimized out>) at Laplacian.cpp:9

Setting a breakpoint in device kernel

- agent-id = Agent Target ID
- queue-id = Queue Target ID
- dispatch-num = Dispatch Target ID how many kernels have been launched
- wave-id = Wavefront ID index of wavefront of kernel
- (z, y, x) work-group/block index

[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]

Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out >, localNy=<optimized out>, stride=<optimized out>, dx=<optimized out>, dy=<optimize d out>, U=<optimized out>, AU=<optimized out>) at Laplacian.cpp:9

To look at the progress of a single wavefront, disable the breakpoint by typing 'disable <*num*>', and then 'step'

Ē	jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip
Image: 1 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27	<pre>jychang48@jychang48-workstati //***********************************</pre>	<pre>ion:-/Downloads/hiptutorial/hip Q = - • </pre> \$3 = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0} (gdb) step (gdb) step (gdb) step (gdb) b LocalLaplacianKernel Function "LocalLaplacianKernel" not defined. Make breakpoint pending on future shared library load? (y or [n]) y Breakpoint 2 (LocalLaplacianKernel) pending.(gdb) continue Continuing. [Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0] Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out="">, localNy=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1287 (65,1,0)/2] Thread 1295 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1269 (61,1,0)/0] Thread 1277 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1269 (61,1,0)/0] Thread 1277 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1269 (61,1,0)/0] Thread 1277 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, Stride=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">, dx=<optimized out="">, dx=<optimized out="">, dx=<optimized out=""></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized>
28 29 30 31	AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) + (-U[id_d] + 2*U[id] - U[id_u])/(dy*dy);	[Switching to AMDGPU Thread 1:5:1:363 (90,0,0)/2] Thread 371 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out=""></optimized>
32 33 34 /hor	} void LocalLaplacian(grid_t& grid, mesh_t& mesh, me/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	zed out>, U= <optimized out="">, AU=<optimized out="">, at Laplacian.cpp:9 (gdb) disable 2 (gdb) step (gdb)</optimized></optimized>

Use 'stepi' which enables the user to dive deeper into the HIP API

FI	jychang48@jychang48-workstati	ion: ~/Downloads/hiptutorial/hip
215	} while (0)	(gdb) b LocalLaplacianKernel
216		Function "LocalLaplacianKernel" not defined.
217	<pre>#define hipLaunchKernelGGL(kernelName,) hipLaunchKernelGGLInternal((kernelN</pre>	Make breakpoint pending on future shared library load? (y or [n]) y
218	#endlt	
219	Hinelude this (his mustime and he	Breakpoint 2 (LocalLaplacianKernel) pending.(gdb) continue
220	<pre>#include <nip nip_runtime_api.n=""> ovtern "C" deviceattribute ((const)) size tsckl get local id(uipt);</nip></pre>	Continuing. [Switching to AMDCDU Thread $1.5.1.1$ (0.0.0)(0]
221	extern "C" deviceattribute ((const)) size tockl get group id(uint);	
223	extern "C" deviceattribute ((const)) size tockl get local size(uint);	Thread 9 "Jacobi hip" hit Breakpoint 2. LocalLaplacianKernel (localNx= <optimized out<="" td=""></optimized>
224	extern "C" deviceattribute ((const)) size tockl get num groups(uint);	>. localNv= <optimized out="">. stride=<optimized out="">. dx=<optimized out="">. dv=<optimized< td=""></optimized<></optimized></optimized></optimized>
225	<pre>struct HIP BlockIdx {</pre>	d out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
226	device	(gdb) step
227	<pre>std::uint32_t operator()(std::uint32_t x) const noexcept { returnockl_get_g</pre>	[Switching to AMDGPU Thread 1:5:1:1287 (65,1,0)/2]
228	3;	
229	<pre>structHIP_BlockDim {</pre>	Thread 1295 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized< td=""></optimized<>
230		out>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim< td=""></optim<></optimized></optimized></optimized>
231	<pre>std::uint32_t operator()(std::uint32_t x) const noexcept {</pre>	ized out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
232	<pre>> returmockl_get_local_size(x);</pre>	(gab) Step [Switching to AMDCDU Throad 1.5.1.1260 (61 1 0)/0]
233	ر ۲.	[Switching to Amboro Inteau 1.5.1.1209 (01,1,0)/0]
235	struct HIP GridDim {	Thread 1277 "Jacobi hip" hit Breakpoint 2. LocalLaplacianKernel (localNx= <optimized< td=""></optimized<>
236	device	out>. localNv= <optimized out="">. stride=<optimized out="">. dx=<optimized out="">. dv=<optimized< td=""></optimized<></optimized></optimized></optimized>
237	<pre>std::uint32 t operator()(std::uint32 t x) const noexcept {</pre>	ized out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
238	<pre>return ockl get num groups(x);</pre>	(gdb) step
239	}	[Switching to AMDGPU Thread 1:5:1:363 (90,0,0)/2]
240	};	
241	<pre>structHIP_ThreadIdx {</pre>	Thread 371 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized o<="" td=""></optimized>
242	device	ut>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimi< td=""></optimi<></optimized></optimized></optimized>
243	<pre>std::uint32_t operator()(std::uint32_t X) const noexcept { roturnoskl got local id(x);</pre>	zed out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
244	lecurnocki_gei_tocai_iu(x),	(gub) utsable 2
246	};	(udb) stepi
247		0x00007f28f722f010 in HIP BlockDim::operator() (this= <optimized out="">, x=<optimized< td=""></optimized<></optimized>
248	template <typename f=""></typename>	out>) at /opt/rocm-4.2.0/hip/include/hip/amd detail/hip runtime.h:232
/opt	/rocm-4.2.0/hip/include/hip/amd_detail/hip_runtime.h	(gdb)



Type 'step' again and you can now freely step through the device kernel

F	jychang48@jychang48-workstati	ion: ~/Downloads/hiptutorial/hip
1	//*************************************	Make breakpoint pending on future shared library load? (y or [n]) y
2	<pre>//* Copyright (c) 2019, Advanced Micro Devices, Inc. All rights reserved.</pre>	
3	//*************************************	Breakpoint 2 (LocalLaplacianKernel) pending.(gdb) continue
45	#include "lacobi hnn"	[Switching to AMDGPU Thread 1:5:1:1 (0 0 0)/0]
6	#include Successings	
7	// AU_i,j = (-U_i+1,j + 2U_i,j - U_i-1,j)/dx^2 +	Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out<="" td=""></optimized>
8	// (-U_i,j+1 + 2U_i,j - U_i,j-1)/dy^2	>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimize< td=""></optimize<></optimized></optimized></optimized>
9	global void LocalLaplacianKernel(const int localNx,	d out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
11	const int stride	(gdd) Step [Switching to AMDGPU Thread 1:5:1:1287 (65 1 0)/2]
12	const dfloat dx.	
13	const dfloat dy,	Thread 1295 "Jacobi hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized< td=""></optimized<>
14	<pre>const dfloat *restrict U,</pre>	out>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim< td=""></optim<></optimized></optimized></optimized>
15	dfloat *restrict AU) {	ized out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
17	const int i - threadIdy y+blockIdy y*blockDim y:	(gab) step [Switching to AMDCPU Thread 1:5:1:1269 (61 1 0)/0]
18-	-> const int j = threadIdx.v+blockIdx.v*blockDim.v;	
19		Thread 1277 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized< td=""></optimized<>
20	<pre>if ((i<localnx) &&="" (j<localny))="" pre="" {<=""></localnx)></pre>	out>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim< td=""></optim<></optimized></optimized></optimized>
21	a	ized out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
22	const int id = $(1+L) + (j+L)^{stride};$	(gab) Step [Switching to AMDGPU Thread 1:5:1:363 (90 0 0)/2]
24	const int id l = id - 1;	
25	<pre>const int id_r = id + 1;</pre>	Thread 371 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized o<="" td=""></optimized>
26	<pre>const int id_d = id - stride;</pre>	ut>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimi< td=""></optimi<></optimized></optimized></optimized>
27	const int id_u = id + stride;	zed out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
20	+ (xh*xh)/([r hi]] - [[hi]]*(+ [[hi]]]-) = [[hi]]	(gdb) disable z
30	(-U[id d] + 2*U[id] - U[id u])/(dv*dv);	(gdb) stepi
31	}	0x00007f28f722f010 inHIP_BlockDim::operator() (this= <optimized out="">, x=<optimized< td=""></optimized<></optimized>
32	}	out>) at /opt/rocm-4.2.0/hip/include/hip/amd_detail/hip_runtime.h:232
33	world Legellenlegien (wrid +C wrid work +C work	(gub) step
34 /hom	void LocalLaplaClan(grid_t& grid, mesn_t& mesn,	
, 110111	c, jyenang o, bown cours, nipracoriac, nipracoriac, nipracorian cop	



Examining the ISA

Several ways one can view the ISA. Using cgdb, type

ESC -> :set dis -> ENTER

F	jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip
3	9global void LocalLaplacianKernel(const int localNx,	Make breakpoint pending on future shared library load? (y or [n]) y
4	0x00007f28f722f000 <+0>: s_load_dwordx4 s[0:3], s[6:7], 0x0	
5		Breakpoint 2 (LocalLaplacianKernel) pending.(gdb) continue
6	/opt/rocm-4.2.0/hip/include/hip/amd_detail/hip_runtime.h:	Continuing.
7	<pre>232 returnockl_get_local_size(x);</pre>	[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]
8	0x00007f28f722f008 <+8>: s_load_dword s10, s[4:5], 0x4	
9	0x00007f28f722f010 <+16>: s_waitcnt lgkmcnt(0)	Thread 9 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out<="" th=""></optimized>
10	0x00007f28f722f014 <+20>: s_lshr_b32 s3, s10, 16	>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimize< p=""></optimize<></optimized></optimized></optimized>
11	0x00007f28f722f018 <+24>: s_and_b32 s4, s10, 0xffff	d out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
12	0x00007f28f722f020 <+32>: s_mul_i32 s8, s8, s4	(gdb) step
13	0x00007f28f722f024 <+36>: s_mul_i32 s9, s9, s3	[Switching to AMDGPU Thread 1:5:1:1287 (65,1,0)/2]
14		
15	Laplacian.cpp:	Thread 1295 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized< th=""></optimized<>
16	<pre>17 const int i = threadIdx.x+blockIdx.x*blockDim.x;</pre>	out>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim< th=""></optim<></optimized></optimized></optimized>
17	0x0000712817221028 <+40>: v_add_u32_e32 v0, s8, v0	ized out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
18		(gdb) step
19	<pre>l8 const int j = threadIdx.y+blockIdx.y*blockDim.y;</pre>	[Switching to AMDGPU Thread 1:5:1:1269 (61,1,0)/0]
20	> 0x0000/1281/221026 <+44>: v_add_u32_e32 v1, s9, v1	
21		Inread 12// "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized< th=""></optimized<>
22		out>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimized out="">, dy=<optided out="">, dy=<optimized out="">, dy=<optimized out="">, dy=<optimized< th=""></optimized<></optimized></optimized></optided></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized>
23	$\frac{1}{10} \left(\frac{1}{1000} \left(\frac{1}{1000} \left(\frac{1}{1000} \right) \right) \left(\frac{1}{1000} \left(\frac{1}{1000} \right) \right) \left(\frac{1}{1000} \right$	ized out>, U= <optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized>
24	0x00000/12817221030 <+48>: v_cmp_gt_132_e32 vcc, s0, v0	
25	0x0000/1281/221034 <+52>: V_cmp_gt_132_e64 s[6:1], s1, V1	[Switching to AMDGPU Inread 1:5:1:363 (90,0,0)/2]
26	0x0000/1281/221036 <+60>: s_and_bb4 s[0:1], vcc, s[0:1]	
27	0x0000/1281/221040 <+64>: s and saveexec bo4 s[4:5], s[0:1]	Inread 3/1 "Jacobi nip" nit Breakpoint 2, LocalLaplaciankernel (localNx= <optimized o<="" th=""></optimized>
28	0x0000/1281/221044 <+08>: 5_CDrancn_exec2 104	ut_{x} , tocative-optimized out _x , stride=coptimized out _x , dx =coptimized out _x , dy =coptimi
29		(adb) displant out>, de= <optimized out="">, Au=<optimized out="">) at Laptacian.cpp:9</optimized></optimized>
30	21	(gdb) disable 2
22	22 constant $10 = (1+1) + (1+1)^{-5}$ (100)	(gdb) step
32	0x0000712817221048 < +725; V adu usz esz VI, I, VI	(gdd) stepi ovonovatastazatoin in UTD BlockDim.concreter() (this continized out, vecontinized
20	0,0000712877227654 < +703; v mut to us2 v1, v1, s2 0,00075654 < +703; v mut to us2 v1, v1, s2 0,00075654 < +703557275654 < +703557275772757574 < +703557777577577775777777777777777777777	auts) = t (opt/solver) A = 0/bin/include/bin/amd detail/bin subtimized outs, x=optimized
25	$(0,0000,1201,2210,34,<+04>; S_(0,000,000,23); S_(0,1); S_(0,7); 0,0000,1201,2210,34,<+04>; S_(0,000,000,000,000,000,000,000,000,000,$	(adb) stop
36	• dlobal woid locallanlacianKernel(const int localNx	(gdb) step
50 Sde	for function	(gdb) step
Pue		



Examining the ISA

Let's say I want to see what the value of "i" is for my wavefront:

F	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip Q = - 🛛 😣
3 4 5	<pre>9global void LocalLaplacianKernel(const int localNx, 0x00007f28f722f000 <+0>: s_load_dwordx4 s[0:3], s[6:7], 0x0</pre>	(gdb) step [Switching to AMDGPU Thread 1:5:1:1287 (65,1,0)/2]
6 7 8 9 10 11	<pre>/opt/rocm-4.2.0/hip/include/hip/amd_detail/hip_runtime.h: 232</pre>	Thread 1295 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out>, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim ized out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:1269 (61,1,0)/0]</optimized></optimized></optim </optimized></optimized></optimized></optimized
12 13 14	0x00007f28f722f020 <+32>: s_mul_i32 s8, s8, s4 0x00007f28f722f024 <+36>: s_mul_i32 s9, s9, s3	Thread 1277 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized out>, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optim ized out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized></optim </optimized></optimized></optimized></optimized
15 16	Laplacian.cpp: 17 const int i = threadIdx.x+blockIdx.x*blockDim.x:	(gdb) step [Switching to AMDGPU Thread 1:5:1:363 (90.0.0)/2]
10 17 18 19 20 21 22 23 24 25	<pre>17</pre>	Thread 371 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localNx= <optimized o<br="">ut>, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimi zed out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) disable 2 (gdb) step (gdb) stepi 0x00007f28f722f010 inHIP_BlockDim::operator() (this=<optimized out="">, x=<optimized out>) at /opt/rocm-4.2.0/hip/include/hip/amd_detail/hip_runtime.h:232</optimized </optimized></optimized></optimized></optimi </optimized></optimized></optimized></optimized>
26 27 28	0x00007f28f722f03c <+60>: s_and_b64 s[0:1], vcc, s[0:1] 0x00007f28f722f040 <+64>: s_and_saveexec_b64 s[4:5], s[0:1] 0x00007f28f722f044 <+68>: s_cbranch_execz 104	(gdb) step (gdb) step (gdb) info v0
29 30 31 32 33	21 22 const int id = (i+1) + (j+1)*stride; 0x00007f28f722f048 <+72>: v_add_u32_e32 v1, 1, v1 0x00007f28f722f04c <+75>: v_mul_lo_u32 v1_v1_s2	Guderined into command: vo. Try netp into. (gdb) info reg v0 v0 {0x5a0, 0x5a1, 0x5a2, 0x5a3, 0x5a4, 0x5a5, 0x5a6, 0x5a7, 0x5a8, 0x5a9 , 0x5aa, 0x5ab, 0x5ac, 0x5ad, 0x5ae, 0x5af, 0x5a0, 0x5a1, 0x5a2, 0x5a3, 0x5a4, 0x5a5 0x5a6, 0x5a7, 0x5a8, 0x5a9, 0x5aa, 0x5ab, 0x5a6, 0x5a4, 0x5a9, 0x5a5
34 35 36 >de	9 global void LocalLaplacianKernel(const int localNx, for function Z20LocalLaplacianKerneliiiddPKdPd: (7f28f722f000 - 7f28f722f1e8) **	, 0x5a0, 0x5a7, 0x5a8, 0x5a8, 0x5a8, 0x5aa, 0x5ab, 0x5ac, 0x5ad, 0x5ae, 0x5ar, 0x5ab, 0x5ar, , 0x5a2, 0x5a3, 0x5a4, 0x5a5, 0x5a6, 0x5a7, 0x5a8, 0x5a9, 0x5aa, 0x5ab, 0x5ac, 0x5ad , 0x5ae, 0x5af, 0x5a0, 0x5a1, 0x5a2, 0x5a3, 0x5a4, 0x5a5, 0x5a6, 0x5a7, 0x5a8, 0x5a9 , 0x5aa, 0x5ab, 0x5ac, 0x5ad, 0x5ae, 0x5af} (gdb)

Examining the ISA

This also works for scalar registers, for instance here we can check the conditional:

(i < localNx) && (j < localNy)

which is stored in s[0:1], one bit corresponding to each thread in the wavefront:

Æ	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip	Q = - 🛛 🙁
15 16 17 18	Laplacian.cpp: 17 const int i = threadIdx.x+blockIdx.x*blockDim.x; 0x00007f28f722f028 <+40>: v_add_u32_e32 v0, s8, v0	out>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized o<br="">ized out>, U=<optimized out="">, AU=<optimized out="">) at Laplacian.cpp:9 (gdb) step [Switching to AMDGPU Thread 1:5:1:363 (90,0,0)/2]</optimized></optimized></optimized></optimized></optimized>	ut>, dy= <optim< th=""></optim<>
19	<pre>18 const int j = threadIdx.y+blockIdx.y*blockDim.y;</pre>		
20 21 22	0x00007f28f722f02c <+44>: v_add_u32_e32 v1, s9, v1	Thread 371 "Jacobi_hip" hit Breakpoint 2, LocalLaplacianKernel (localN ut>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized ou<br="">zed out>. U=<optimized out="">. AU=<optimized out="">) at Laplacian.cpp:9</optimized></optimized></optimized></optimized></optimized>	x= <optimized o<br="">t>, dy=<optimi< th=""></optimi<></optimized>
23	<pre>20 if ((i<localnx) &&="" (j<localny))="" pre="" {<=""></localnx)></pre>	(gdb) disable 2	
24	0x00007f28f722f030 <+48>: v cmp gt i32 e32 vcc, s0, v0	(gdb) step	
25	0x00007f28f722f034 <+52>: v_cmp_gt_i32_e64 s[0:1], s1, v1	(gdb) stepi	
26	0x00007f28f722f03c <+60>: s_and_b64 s[0:1], vcc, s[0:1]	<pre>0x00007f28f722f010 inHIP_BlockDim::operator() (this=<optimized out=""></optimized></pre>	, x= <optimized< th=""></optimized<>
27	0x00007f28f722f040 <+64>: s_and_saveexec_b64 s[4:5], s[0:1]	out>) at /opt/rocm-4.2.0/hip/include/hip/amd_detail/hip_runtime.h:232	
28	0x00007f28f722f044 <+68>: s_cbranch_execz 104	(gdb) step	
29		(gdb) step	
30	ZI	(GGD) 1NTO VU	
31	$22 \qquad \text{const int id} = (1+1) + (1+1)^{\text{stride}};$	Undefined into command: "VU". Try "nelp into".	
32	> 0X0000/1281/221048 <+/2>: V_d00_u32_e32_V1, 1, V1	((gab) into reg v⊍ \versiling (av5a) av5a1 av5a2 av5a2 av5a4 av5a5 av5a6 av5a7	015-29 015-20
24	$0 \times 0000077287722704C < 702; v mul (0 usz vi, vi, sz 0 \times 0000077287722704C < 802; s load dwordy2 s[6:1] s[6:7] 0 × 20$	100 (0x3a0, 0x3a1, 0x3a2, 0x3a3, 0x3a4, 0x3a3, 0x3a0, 0x3a7 0x5aa 0x5ab 0x5ac 0x5ad 0x5ae 0x5af 0x5a0 0x5a1 0x5a2 0x5a3	0x5a0, 0x5a9
35	3_COAU_AWOTUA2 S[0.1], S[0.7], 0.20	γ ,	0x5a0 $0x5a1$
36	9 global void LocalLaplacianKernel(const int localNx.	, 0x5a2, 0x5a3, 0x5a4, 0x5a5, 0x5a6, 0x5a7, 0x5a8, 0x5a9, 0x5aa, 0x5ah	. 0x5ac. 0x5ad
37	$0 \times 000077287722705c <+92>: s load dwordx4 s[8:11], s[6:7], 0x10$, 0x5ae, 0x5af, 0x5a0, 0x5a1, 0x5a2, 0x5a3, 0x5a4, 0x5a5, 0x5a6, 0x5a7	. 0x5a8. 0x5a9
38		0, Jag, 0, Jab, 0, Jac, 0, Jad, 0, Jac, 0, Jac	, ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
39	21	(qdb) info reg s0	
40	<pre>22</pre>	s0 0xffe 4094	
41	0x00007f28f722f064 <+100>: v_add_u32_e32 v4, v1, v0	(gdb) info reg sl	
42		s1 0xffe 4094	
43	28	(gdb) step	
44	29 $AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +$	(gdb) step	
45	0x0000712817221068 <+104>: v_ashrrev_i32_e32 v5, 31, v4	(gdb) into reg s0	
46	0x0000/1281/22106c <+108>: v_lshlrev_b64 v[5:6], 3, v[4:5]		
4/	0X0000/T28T/22T0/4 <+11b>: s_waltcnt lgkmcnt(0)	(gab) into reg si	
48 >de	for function _720LocalLaplacianKernelijijddPKdPd; (7f28f722f000 _ 7f28f722f108) **		
-ue			

Switching wavefronts

Now that I've been stepping only my wavefront, are the others still at the beginning of the kernel? Use info threads. This enables us t o see the location of both host threads and GPU wavefronts ...

F	jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip	. = 😣
15	Laplacian.cpp:	(adb) info threads	[110/112]
16	17 const int i = threadIdx.x+blockIdx.x*blockDim.x:	Id Target Id Frame	
17	0x00007f28f722f028 <+40>: v add u32 e32 v0. s8. v0	1 Thread 0x7f291814d8c0 (LWP 129937) "Jacobi hip" 0x00007f2918b7a89	9b in sched v
18		ield () from /lib/x86 64-linux-gnu/libc.so.6	·····_/
19	<pre>18 const int i = threadIdx.y+blockIdx.y*blockDim.y;</pre>	2 Thread 0x7f2900108700 (LWP 129953) "Jacobi hip" 0x00007f2918b8aa	ff in poll ()
20	0x00007f28f722f02c <+44>: v add u32 e32 v1, s9, v1	from /lib/x86 64-linux-gnu/libc.so.6	· · · · · · · · · · · · · · · · · · ·
21	` `	3 Thread 0x7f28ff74a700 (LWP 129954) "Jacobi hip" 0x00007f2918b9750	ce in epoll w
22	19	ait () from /lib/x86 64-linux-gnu/libc.so.6	' -
23	20 if ((i <localnx) &&="" (j<localny))="" td="" {<=""><td>4 Thread 0x7f28f7f53700 (LWP 129955) "Jacobi hip" 0x00007f2918b8c50</td><td>9b in ioctl (</td></localnx)>	4 Thread 0x7f28f7f53700 (LWP 129955) "Jacobi hip" 0x00007f2918b8c50	9b in ioctl (
24	0x00007f28f722f030 <+48>: v cmp gt i32 e32 vcc, s0, v0) from /lib/x86 64-linux-gnu/libc.so.6	
25	0x00007f28f722f034 <+52>: v cmp qt i32 e64 s[0:1], s1, v1	6 Thread 0x7f28fc07f700 (LWP 129957) "Jacobi hip" 0x00007f2918b7a89	9b in sched y
26	0x00007f28f722f03c <+60>: s and b64 s[0:1], vcc, s[0:1]	ield () from /lib/x86 64-linux-gnu/libc.so.6	_*
27	0x00007f28f722f040 <+64>: s and saveexec b64 s[4:5], s[0:1]	7 Thread 0x7f28f7752700 (LWP 129958) "Jacobi hip" 0x00007f2918b8c50	🥑 in ioctl (
28	0x00007f28f722f044 <+68>: s cbranch execz 104) from /lib/x86 64-linux-gnu/libc.so.6	
29		8 Thread 0x7f28f757f700 (LWP 129959) "Jacobi hip" 0x00007f291a2f467	78 in do fute
30	21	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0	_
31	<pre>22 const int id = (i+1) + (j+1)*stride;</pre>	<pre>* 371 AMDGPU Thread 1:5:1:363 (90,0,0)/2 "Jacobi_hip" LocalLaplacianKer</pre>	rnel (localNx
32	> 0x00007f28f722f048 <+72>: v_add_u32_e32 v1, 1, v1	<pre>=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dx=<optized out="">, dx=<optimized out="">, dx=<optimized ou<="" td=""><td>optimized out</td></optimized></optimized></optized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></pre>	optimized out
33	0x00007f28f722f04c <+76>: v_mul_lo_u32 v1, v1, s2	, dy= <optimized out="">, U=<optimized out="">, AU=<optimized out="">) at Laplac:</optimized></optimized></optimized>	ian.cpp:22 🛛 🖌
34	0x00007f28f722f054 <+84>: s_load_dwordx2 s[0:1], s[6:7], 0x20	(Gdb)	
35		Id Target Id Frame	
36	9global void LocalLaplacianKernel(const int localNx,	1 Thread 0x7f291814d8c0 (LWP 129937) "Jacobi_hip" 0x00007f2918b7a89	<pre>9b in sched_y</pre>
37	0x00007f28f722f05c <+92>: s_load_dwordx4 s[8:11], s[6:7], 0x10	ield () from /lib/x86_64-linux-gnu/libc.so.6	
38		2 Thread 0x7f2900108700 (LWP 129953) "Jacobi_hip" 0x00007f2918b8aat	ff in poll ()
39		from /lib/x86_64-linux-gnu/libc.so.6	
40	22 const int id = (i+1) + (j+1)*stride;	3 Thread 0x7f28ff74a700 (LWP 129954) "Jacobi_hip" 0x00007f2918b9750	ce in epoll_w
41	0x00007f28f722f064 <+100>: v_add_u32_e32 v4, v1, v0	ait () from /lib/x86_64-linux-gnu/libc.so.6	
42		4 Thread 0x7f28f7f53700 (LWP 129955) "Jacobi_hip" 0x00007f2918b8c50	Əb in ioctl (
43	28) from /lib/x86_64-linux-gnu/libc.so.6	
44	$\frac{29}{AU[1d]} = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) + \frac{1}{2}$	6 [hread 0x/f28fc07f700 (LWP 129957) "Jacobi_hip" 0x00007f2918b7a8	9b in sched_y
45	0x00000/1281/221068 <+104>: v_ashrrev_i32_e32 v5, 31, v4	leld () from /lib/x86_64-linux-gnu/libc.so.6	
46	0x0000/1281/22106c <+108>: v_lshlrev_b64 v[5:6], 3, v[4:5]	/ Inread 0x/f28f//52/00 (LWP 129958) "Jacobi_hip" 0x00007f2918b8c50	b in loctl (
4/	0X0000/T28T/22T0/4 <+110>: s_waitcht lgkmcht(0)) Trom /llD/X86_64-llnux-gnu/llbc.so.6	
48	0X0000/1281/2210/8 <+120>: V mov b32 e32 V15, S1	8 Inread 0x/f28f/5/f/00 (LWP 129959) "Jacobi hip" 0x0000/f291a2f46.	/8 in do_fute
>de	Tor function _220LocalLaplacianKernelliiddPKdPd: (/f28f/22f000 - /f28f/22f1e8) **	<pre>TX_Wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0</pre>	l



Switching wavefronts

.. but where did all the other threads go? **Step allows other wavefronts to advance**, hence the rest of the waves in our kernel have already completed!:

F	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip Q	= - • 😣
15	Laplacian.cpp:	(adb) info threads	[110/112]
16	<pre>17 const int i = threadIdx.x+blockIdx.x*blockDim.x:</pre>	Id Target Id Frame	
17	0x00007728f722f028 <+40>: v add u32 e32 v0. s8. v0	1 Thread 0x7f291814d8c0 (LWP 129937) "Jacobi hip" 0x00007f2918b7a89	b in sched v
18		ield () from /lib/x86 64-linux-gnu/libc.so.6	
19	18 const int i = threadIdx v+blockIdx v*blockDim v:	2 Thread 0x7f2900108700 (JWP 129953) "Jacobi hip" 0x00007f2918b8aaf	f in noll ()
20	0x00007f28f722f02r <+44>: v add u32 a32 v1 s9 v1	from /lib/x86 64-libux-gnu/libc so 6	· 11 poet ()
21		3 Thread 07728ff74a700 (UWP 129954) "lacobi hin" 0x00007f2918b975c	e in enoll w
22		ait () from /lib/x86.64_libux_gnu/libc_so_6	
22	$if ((i < local Nx)) \delta \delta (i < local Ny)) \delta$	4 Thread 0×75287753700 (LWP 120055) "lacobi hin" $0 \times 0000752918h8c50$	h in ioctl (
23	$0 \times 0.000772877227030 < +485$ y cm at 132 p32 ycc s0 y0) from /lib/y86.64_lipuy_gpu/libc so 6	D IN IOCCC (
25	$a_{y}a_{0}a_{0}f_{2}a_{1}f_{2}f_{3}f_{3}f_{4}(z_{1}z_{2})$, $y_{c}a_{0}a_{1}f_{2}a_{1}f_{3}(z_{1}z_{2})$	6 Thread $0\sqrt{7}(2876075700)$ (LWP 120057) "lacobi bin" $0\sqrt{0}00075201807380$	h in schod v
25	0x00007f20f722f02c +f02; c = 0 b64 c[0:1] vc = [0:1]	iald () from /lib/v86 6/lipux anu/libc co 6	b in sched_y
20	$0_{0}0000712917221040 < t < t < > s = and - bay so (s, t), vec, s [0, 1]$	7 Thread 0x7f28f752700 (WP 120058) "lacobi bin" 0x00007f2018b8c50	h in ioctl (
27	0x0000712017221040 <+04>. s_allu_saveexec_bu4 s[4.3], s[0.1]) from (lib(x)66 lipux apu/libc co.6	D TH TOCCC (
20	5_CDT alcli_execz_104	// Thom / (II/ A00_04- (IMA-900/CIDC.S0.0	o in do futo
29		o Infedu 0x/1201/3/1/00 (LWF 12993) Jacobi int 0x0000/1291a2140/	
20	22	$X = \frac{1}{2}$ AMPCON Thread 1.5.1.262 (0.0.1/2) and 1.1.2.3 (1.1.2.3)	
22	22 constant in $= (1+1) + (1+1)^{-2}$ string;	* 3/1 AMDGPU INTEAU 1:3:1:305 (90,0,0)/2 Jacobi Nip LocalLaplaClainker	net (tocathx
32		= <optimized data<="" outs,="" stride="<optimized" td="" tocativy="<optimized"><td>ptimized out</td></optimized>	ptimized out
33	0x0000772877227046 <+76>: V mut to us2 V1, V1, S2	, dy= <optimized out="">, U=<optimized out="">, AU=<optimized out="">) at Laplaci autorial</optimized></optimized></optimized>	an.cpp:22
34	0x0000/f28f/22f054 <+84>: s_toad_dwordx2 s[6:1], s[6:7], 0x20		
35		Id larget Id Frame	
30	global Void LocalLaplaciankernel(const int localNx,	1 Inread 0x/f29181408c0 (LWP 129937) "Jacobi_nip" 0x0000/f2918b7889	b in sched_y
3/	0x0000/f28f/22f05c <+92>: s_load_dwordx4 s[8:11], s[6:7], 0x10	<pre>leld () from /llb/x86_64-linux-gnu/llbc.so.6</pre>	<pre></pre>
38		2 Thread 0x/f2900108700 (LWP 129953) "Jacobi_hip" 0x0000/f2918b8aaf	f in poll ()
39	21	from /lib/x86_64-linux-gnu/libc.so.6	
40	22 const int id = (i+1) + (j+1)*stride;	3 Thread 0x/f28ff/4a/00 (LWP 129954) "Jacobi_hip" 0x00007f2918b975c	e in epoll_w
41	0x0000772877227064 <+100>: v_add_u32_e32 v4, v1, v0	ait () from /lib/x86_64-linux-gnu/libc.so.6	
42		4 Thread 0x7f28f7f53700 (LWP 129955) "Jacobi_hip" 0x00007f2918b8c50	b in ioctl (
43) from /lib/x86_64-linux-gnu/libc.so.6	
44	29 $AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +$	6 Thread 0x7f28fc07f700 (LWP 129957) "Jacobi_hip" 0x00007f2918b7a89	b in sched_y
45	0x00007t28t722t068 <+104>: v_ashrrev_i32_e32 v5, 31, v4	ield () from /lib/x86_64-linux-gnu/libc.so.6	
46	0x000077287722706c <+108>: v_lshlrev_b64 v[5:6], 3, v[4:5]	7 Thread 0x7f28f7752700 (LWP 129958) "Jacobi_hip" 0x00007f2918b8c50	b in ioctl (
47	0x00007f28f722f074 <+116>: s_waitcnt lgkmcnt(0)) from /lib/x86_64-linux-gnu/libc.so.6	
48	0x0000772877227078 <+120>: v_mov_b32_e32 v15, s1	8 Thread 0x7f28f757f700 (LWP 129959) "Jacobi_hip" 0x00007f291a2f467	8 in do_fute
>de	<pre>for function _Z20LocalLaplacianKerneliiiddPKdPd: (7f28f722f000 - 7f28f722f1e8) **</pre>	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0	

Rerunning rocgdb with scheduler-locking

Use 'set scheduler-locking on' will not continue the other waves.

Æ	jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip
1	//*************************************	
2	<pre>//* Copyright (c) 2019, Advanced Micro Devices, Inc. All rights reserved. //***********************************</pre>	For help, type "help". Type "apropos word" to search for commands related to "word" Deading symbols from (lesschi bin
4		(all) b a ball all all all all all all all al
5	#include "Jacobi.hpp"	(gdb) b LocalLaplaciankernel
6		Function "LocalLaplacianKernel" not defined.
/	$// AU_1, j = (-U_1+1, j + 2U_1, j - U_1-1, j)/dx^2 +$	Make breakpoint pending on future shared library load? (y or [n]) y
	// (-0_1,)+1+20_1,) - 0,)////////	Prosknoint 1 (local anlacianKornol) ponding (adh) run
10		Starting program, (home (iveband 19 m) and (hintutaria) (hin (lacabi hin a 1.1
11	const int otatiny,	Starting program: /nome/jychang4o/bowncoads/nip/utoriat/nip/Jacobi_nip -g i i
12	const dfloat dy	Using best lighthrough the light way will be the set of
12		Determine of the form child process 121611
13	const diloat dy,	[Detaching after fork from child process islooi]
14	dilact * restrict 0,	[New Inread 0x76050005700 (LWP 131008)]
10	artoat *restrictAU) {	[New Inread 0x/16C3d00/00 (LWP 131009)]
10		Topology Size: 1 X 1
1/	const int 1 = threadiax.x+blockIax.x+blockUim.x;	Local domain size (current node): 4090 x 4090
18	const int j = threadidx.y+blockidx.y+blockDim.y;	Global domain size (all nodes): 4096 X 4096
19		[New Inread 0x/ToCS8T02/00 (LWP 1310/0)]
20	וד ((1 <localnx) (j<localny))="" td="" {<="" אא=""><td>Rank @ selecting device @ on host jychang48-workstation</td></localnx)>	Rank @ selecting device @ on host jychang48-workstation
21		[New Inread 0x/f6c5253e/00 (LWP 1316/1)]
22	const int $Id = (1+1) + (1+1) * stride;$	[Ihread 0x/t6c2253e/00 (LWP 1316/1) exited]
23		[New Ihread 0x/f6c590ff/00 (LWP 1316/2)]
24	$const int id_{L} = id - i;$	[New Ihread 0x/f6C58/ff/00 (LWP 1316/3)]
25	$const int id_r = id + 1;$	[New Thread 0x/f6c585ff700 (LWP 131674)]
26	const int id_d = id - stride;	Starting Jacobi run.
27	<pre>const int id_u = id + stride;</pre>	Iteration: 0 - Residual: 0.022108
28		[Switching to AMDGPU Thread 1:5:1:1 (0,0,0)/0]
29	$AU[1d] = (-U[1d_l] + 2*U[id] - U[id_r])/(dx*dx) +$	
30	(-U[id_d] + 2*U[id] - U[id_u])/(dy*dy);	Thread 9 "Jacobi_hip" hit Breakpoint 1, LocalLaplacianKernel (localNx= <optimized out<="" td=""></optimized>
31		>, localNy= <optimized out="">, stride=<optimized out="">, dx=<optimized out="">, dy=<optimize< p=""></optimize<></optimized></optimized></optimized>
32		d out>, 8optimized out>, A8-coptimized out>) at Laplacian.cpp:9
33		(tdb) set scheduler-locking on
34	void LocalLaplacian(grid_t& grid, mesh_t& mesh,	(gdb) dicable 1
/hor	ne/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	(gdb)

Rerunning rocgdb with scheduler-locking

Now we see the rest of the GPU threads. Can also type "thread <tid>" to examine one particular thread

	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip	Q = 8
1 2 3 4	//************************************	(gdb) set scheduler-locking on (gdb) disable 1 (gdb) step (gdb) info threads	[60/2103]
4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24	<pre>#include "Jacobi.hpp" // AU_i,j = (-U_i+1,j + 2U_i,j - U_i,j-1)/dx^2 + // (-U_i,j+1 + 2U_i,j - U_i,j-1)/dy^2global void LocalLaplacianKernel(const int localNx,</pre>	<pre>(gdb) info threads Id Target Id Frame 1 Thread 0x7f6c737088c0 (LWP 131652) "Jacobi hip" 0x00007f6c73 er debug_state() () from /opt/rocm/lib/libhsa-runtime64.so.1 2 Thread 0x7f6c5b6c3700 (LWP 131668) "Jacobi_hip" 0x00007f6c74 () from /lib/x86_64-linux-gnu/libc.so.6 3 Thread 0x7f6c5ad05700 (LWP 131669) "Jacobi_hip" 0x00007f6c74 wait () from /lib/x86_64-linux-gnu/libc.so.6 4 Thread 0x7f6c58fd2700 (LWP 131670) "Jacobi_hip" 0x00007f6c74 () from /lib/x86_64-linux-gnu/libc.so.6 6 Thread 0x7f6c590ff700 (LWP 131672) "Jacobi_hip" 0x00007f6c75 tex_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 7 Thread 0x7f6c5877f700 (LWP 131673) "Jacobi_hip" 0x00007f6c74 () from /lib/x86_64-linux-gnu/libc.so.6 8 Thread 0x7f6c585ff700 (LWP 131674) "Jacobi_hip" 0x00007f6c75 tex_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 * 9 AMDGPU Thread 1:5:1:1 (0,0,0)/0 "Jacobi_hip" LocalLaplaci Nx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">) at La 10 AMDGPU Thread 1:5:1:2 (0,0,0)/1 "Jacobi_hip" LocalLaplaci Nx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">) at La 10 AMDGPU Thread 1:5:1:2 (0,0,0)/1 "Jacobi_hip" LocalLaplaci</optimized></optimized></optimized></optimized></optimized></optimized></pre>	Ba95da6 in _load 4145aff in poll 41525ce in epoll 414750b in ioctl 58af678 in do_fu 414750b in ioctl 58af678 in do_fu 58af678 in do_fu canKernel (local dx= <optimized o<br="">aplacian.cpp:17 .anKernel (local dx=<optimized o<="" td=""></optimized></optimized>
25 26 27 28	<pre>const int id_r = id + 1; const int id_d = id - stride; const int id_u = id + stride;</pre>	<pre>ut>, dy=<optimized out="">, locally soptimized out>, AU=<optimized out="">) at La 11 AMDGPU Thread 1:5:1:3 (0,0,0)/2 "Jacobi_hip" LocalLaplaci Nx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">, ut> dy=<optimized out="">, l=<optimized out="">, AU=<optimized out="">) at La</optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></pre>	aplacian.cpp:9 anKernel (local dx= <optimized o<="" td=""></optimized>
29 30 31 32 33 34 /hom	<pre>AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +</pre>	<pre>12 AMDGPU Thread 1:5:1:4 (0,0,0)/3 "Jacobi_hip" LocalLaplaci Nx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">, ut>, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">) at La 13 AMDGPU Thread 1:5:1:5 (1,0,0)/0 "Jacobi_hip" LocalLaplaci Nx=<optimized out="">, localNy=<optimized out="">, stride=<optimized out="">, ut>, dy=<optimized out="">, localNy=<optimized out="">, AU=<optimized out="">, 14 AMDGPU Thread 1:5:1:6 (1,0,0)/1 "Jacobi_hip"</optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></optimized></pre>	anKernel (local dx= <optimized o<br="">aplacian.cpp:9 anKernel (local dx=<optimized o<br="">aplacian.cpp:9 anKernel (lo<u>cal</u></optimized></optimized>



Other tricks: export AMD_LOG_LEVEL=3

By setting the above environment variable, we can get a print of all API calls and more happening:

F	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip	🛛
23 24 cons 25 cons	t int id_l = id - 1; t int id_r = id + 1;	Reading symbols from ./Jacobi_hip (gdb) b Laplacian.cpp: 40	[41/144]
26 cons	t int id d = id - stride;	Breakpoint 1 at 0x415080: file Laplacian.cpp, line 40.(gdb) run	
27 cons	t int id u = id + stride;	Starting program: /home/jychang48/Downloads/hiptutorial/hip/Jacobi hip -g 1 1	L 🛛
28		[Thread debugging using libthread_db enabled]	
29 AU[i	d] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +	Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".	
30	(-U[id_d] + 2*U[id] - U[id_u])/(dy*dy);	[Detaching after fork from child process 151584]:3:rocdevice.cpp :	:457 : 4
31 }		35438640/8 us: Initializing HSA stack.	
32 }		:3:comgreix.cpp :33 : 435438/904/ us: Loduing Cumber Library.	-0x55o1a
34 void Loc	allaplacian(grid t& grid, mesh t& mesh.	.3.10cdevice.cpp	for anu
35	hipStream t stream.	agent=0x7ffa27a13769	or gpu
36	dfloat* d ⁻ U,		
37	dfloat* d_AU) {	[New Thread 0x7f39eab37700 (LWP 151591)]	
38		[New Thread 0x7f39ea179700 (LWP 151592)]	
39 //ther	e are (Nx-2)x(Ny-2) node on the interior of the mesh	Topology size: 1 x 1	
40 -> int lo	calNx = mesh.Nx-2;	Local domain size (current node): 4096 x 4096	
41 INT LO	calny = mesn.ny-2;	GLODAL GOMAIN SIZE (ALL NOGES): 4090 X 4090	
$\frac{42}{43}$ int yt	hreads $= 16$		
44 int vt	hreads = 16 :	:3:comgretx.cpp :33 : 43544267424 us: Loading COMGR library.	
45		:3:rocdevice.cpp :200 : 43544292744 us: Numa selects cpu agent[0]=	=0x88ab4
46 dim3 t	hreads(xthreads,ythreads,l);	0(fine=0x84b9c0,coarse=0x8a39a0, kern arg=0x8a2900) for gpu agent=0x7f3a03a43	31d9
47 dim3 b	<pre>locks((localNx+xthreads-1)/xthreads,</pre>	:3:hip_context.cpp :124 : 43544298171 us: 151575: [7f3a02b7c8c0] hip	oInit: R
48	<pre>(localNy+ythreads-1)/ythreads, 1);</pre>	eturned hipSuccess :	
49		:3:hip_device_runtime.cpp :497 : 43544298229 us: 151575: [7f3a02b7c8c0] hip	oGetDevi
50 nipLau	ncnKernelGGL(LocalLaplacianKernel,	Celount (0X/TTd1902Cla4)	CotDovi
52	DLUCKS, threads	;;;nip_device_runtime.cpp ::499 : 45544296250 us: 151575; [715d02D7C6C0] nip	DecDevi
53	0. stream	'3'hin device runtime.con :512 : 43544298264 us: 151575: [7f3a02b7c8c0] hir	SetDevi
54	localNx. localNv. mesh.Nx.	ce(0)	
55	mesh.dx, mesh.dy,	:3:hip device runtime.cpp :517 : 43544298269 us: 151575: [7f3a02b7c8c0] hip	SetDevi
56	d_U, d_AU);	ce: Returned hipSuccess :	
/home/jychang	48/Downloads/hiptutorial/hip/Laplacian.cpp	Rank 0 selecting device 0 on host jychang48-workstation	



Other tricks: export AMD_LOG_LEVEL=3

By setting the above environment variable, we can get a print of all API calls and more happening:

J.F.	jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip Q = - 0 😣
23		Rank 0 selecting device 0 on host jychang48-workstation [61/144]
24	<pre>const int id_l = id - l;</pre>	:3:hip_memory.cpp :289 : 43544298319 us: 151575: [7f3a02b7c8c0] hipHostMal
25	<pre>const int id_r = id + 1;</pre>	loc (0x7ffd1902c6f8, 131072, 0)
26	<pre>const int id_d = id - stride;</pre>	:3:hip_memory.cpp :318 : 43544298463 us: 151575: [7f3a02b7c8c0] hipHostMal
27	<pre>const int id_u = id + stride;</pre>	loc: Returned hipSuccess : 0x7f39e85c0000: duration: 144 us
28		:3:hip_memory.cpp :289 : 43544298473 us: 151575: [7f3a02b7c8c0] hipHostMal
29	AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +	loc (0x7ffd1902c700, 131072, 0)
30	(-U[id_d] + 2*U[id] - U[id_u])/(dy*dy);	:3:hip_memory.cpp :318 : 43544298589 us: 151575: [7f3a02b7c8c0] hipHostMal
31	}	loc: Returned hipSuccess : 0x7f39e8580000: duration: 116 us
32		:3:hip_memory.cpp :283 : 43544298607 us: 151575: [7f3a02b7c8c0] hipMalloc
33		(0x7ffd1902c708, 131072)
34 \	/oid LocalLaplacian(grid_t& grid, mesh_t& mesh,	:3:rocdevice.cpp :2065: 43544298694 us: device=0x8d2890, freeMem_ = 0xfef
35	hipStream_t stream,	e0000
36	dfloat* d_U,	:3:hip_memory.cpp :285 : 43544298702 us: 151575: [7f3a02b7c8c0] hipMalloc:
37	dfloat* d_AU) {	Returned hipSuccess : 0x7f39e1a00000: duration: 95 us
38		:3:hip_memory.cpp :1877: 43544298712 us: 151575: [7f3a02b7c8c0] hipMemset
39	<pre>//there are (Nx-2)x(Ny-2) node on the interior of the mesh</pre>	(0x7f39e1a00000, 0, 131072)
40 ->	<pre>> int localNx = mesh.Nx-2;</pre>	:3:rocdevice.cpp :2566: 43544299081 us: number of allocated hardware queu
41	<pre>int localNy = mesh.Ny-2;</pre>	es with low priority: 0, with normal priority: 0, with high priority: 0, maximum per
42		priority is: 4
43	<pre>int xthreads = 16;</pre>	[New Thread 0x7f39e19ff700 (LWP 151594)]
44	<pre>int ythreads = 16;</pre>	[Thread 0x7f39e19ff700 (LWP 151594) exited]
45		[New Thread 0x7f39e857f700 (LWP 151595)]
46	dim3 threads(xthreads,ythreads,l);	:3:rocdevice.cpp :2638: 43544316233 us: created hardware queue 0x7f39e88c
47	dim3 blocks((localNx+xthreads-1)/xthreads,	6000 with size 1024 with priority 1, cooperative: 0
48	(localNy+ythreads-1)/ythreads, 1);	:3:devprogram.cpp :2463: 43544489762 us: Using Code Object V4.
49		:3:rocvirtual.cpp
50	hipLaunchKernelGGL(LocalLaplacianKernel,	7f39e1a00000 obj:[0x7f39e1a00000-0x7f39e1a20000] threadId : 7f39e857f700
51	blocks,	
52	threads,	:3:rocvirtual.cpp
53	0, stream,	x7f39e876e080 obj:[0x7f39e876e000-0x7f39e876f000] threadId : 7f39e857f700
54	localNx, localNy, mesh.Nx,	
55	mesh.dx, mesh.dy,	:3:rocvirtual.cpp :2521: 43544495591 us: [7f39e857f700]! ShaderName :
56	d_U, d_AU);	amd_rocclr_fillBuffer
/home/	/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	



Other tricks: export AMD_LOG_LEVEL=3

By setting the above environment variable, we can get a print of all API calls and more happening:

	jychang48@jychang48-workstatio	on: ~/Downloads/hiptutorial/hip
23 24 25 26	<pre>const int id_l = id - 1; const int id_r = id + 1; const int id_d = id - stride;</pre>	LaunchKernel (0x0x949de0, 128, 1, 1, 128, 1, 1, 0, stream:0x789710, 0x7ffd[144/144] char array: <null>, event:0, event:0, 0, 0) :3:rocvirtual.cpp :2521: 43544718510 us: [7f39e8446700]! ShaderName : _Z11NormKernel1iddPKdPd</null>
27 28 29 30 31 32	<pre>const int id_u = id + stride; AU[id] = (-U[id_l] + 2*U[id] - U[id_r])/(dx*dx) +</pre>	:3:hip_platform.cpp :649 : 43544718514 us: 151575: [7f3a02b7c8c0] ihipLaunch Kernel: Returned hipSuccess : :3:hip_module.cpp :433 : 43544718523 us: 151575: [7f3a02b7c8c0] hipLaunchK ernel: Returned hipSuccess : :3:rocvirtual.cpp :572 : 43544718524 us: ! arg0: = ptr:0x7f39e1a29000
33 34 35 36 37 38	void LocalLaplacian(grid_t& grid, mesh_t& mesh, hipStream_t stream, dfloat* d_U, dfloat* d_AU) {	obj:[0x7f39e1a29000-0x7f39e1a29400] threadId : 7f39e8446700 :3:hip_memory.cpp :328 : 43544718529 us: 151575: [7f3a02b7c8c0] hipMemcpy (0x7f39e8706000, 0x7f39e1a2a000, 8, hipMemcpyDeviceToHost) :3:rocvirtual.cpp :572 : 43544718534 us: ! argl: = ptr:0x7f39e1a2a000 obj:[0x7f39e1a2a000-0x7f39e1a2a008] threadId : 7f39e8446700
39 40 → 41 42	<pre>//there are (Nx-2)x(Ny-2) node on the interior of the mesh > int localNx = mesh.Nx-2; int localNy = mesh.Ny-2;</pre>	:3:rocvirtual.cpp :2521: 43544718546 us: [7f39e8446700]! ShaderName : _Z11NormKernel2iPKdPd
42 43 44 45	<pre>int xthreads = 16; int ythreads = 16;</pre>	:3:hip_memory.cpp :331 : 43544719486 us: 151575: [7f3a02b7c8c0] hipMemcpy: Returned hipSuccess : : duration: 957 us Iteration: 0 - Residual: 0.022108
46 47 48	<pre>dim3 threads(xthreads,ythreads,1); dim3 blocks((localNx+xthreads-1)/xthreads,</pre>	:3:hip_device_runtime.cpp :458 : 43544719507 us: 151575: [7f3a02b7c8c0] hipDeviceS ynchronize () :3:hip_device_runtime.cpp :470 : 43544719512 us: 151575: [7f3a02b7c8c0] hipDeviceS
49 50 51	hipLaunchKernelGGL(LocalLaplacianKernel, blocks,	<pre>ynchronize: Returned hipSuccess : :3:hip_event.cpp :310 : 43544719523 us: 151575: [7f3a02b7c8c0] hipEventRe cord (event:0x948d00, stream:0x789710) :2:hip_event.com</pre>
52 53 54 55 56	(jychang48/Downloads(hiptutorial(hip(lanlacian, cpp	<pre>cord: Returned hipSuccess : Thread 1 "Jacobi_hip" hit Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0 x789710, d_U=0x7f3893e00000, d_AU=0x7f388bc00000) at Laplacian.cpp:40 (adb)</pre>
nome	/ Jychang40/ Down toads/ htp://toi/ia//htp//tap/actan.cpp	(gub)

Other tricks: switching between host threads

Type 'i th' to see a list of all active host threads. Currently viewing thread 1 (default).

ا ي jychang48@jychang48-workstatio	on: ~/Downloads/hiptutorial/hip Q = - 0 😣
<pre>23 24</pre>	:3:rocvirtual.cpp :2521: 43544718546 us: [7f39e8446700]! ShaderName : _Z11NormKernel2iPKdPd
27 27 28 29 AU[id] = (-U[id l] + 2*U[id] - U[id r])/(dx*dx) +	:3:hip_memory.cpp :331 : 43544719486 us: 151575: [7f3a02b7c8c0] hipMemcpy: Returned hipSuccess : : duration: 957 us Iteration: 0 - Residual: 0.022108
30 (-U[id_d] + 2*U[id] - U[id_u])/(dy*dy); 31 } 32 }	:3:hip_device_runtime.cpp :458 : 43544719507 us: 151575: [7f3a02b7c8c0] hipDeviceS ynchronize () :3:hip device runtime.cpp :470 : 43544719512 us: 151575: [7f3a02b7c8c0] hipDeviceS
33 34 <mark>void</mark> LocalLaplacian(grid_t& grid, mesh_t& mesh, 35 hipStream t stream,	<pre>ynchronize: Returned hipSuccess : :3:hip_event.cpp :310 : 43544719523 us: 151575: [7f3a02b7c8c0] hipEventRe cord (_event:0x948d00, stream:0x789710)</pre>
36 dfloat* d_U, 37 dfloat* d_AU) { 38	:3:hip_event.cpp :352 : 43544719530 us: 151575: [7f3a02b7c8c0] hipEventRe cord: Returned hipSuccess :
<pre>39 //there are (Nx-2)x(Ny-2) node on the interior of the mesh 40 -> int localNx = mesh.Nx-2; 41 int localNx = mesh.Nx -2;</pre>	Thread 1 "Jacobi_hip" hit Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0 x789710_d_U=0x7f3893e00000_d_AU=0x7f388bc000000) at Laplacian cpp:40
<pre>41 int tocathy = mesh.hy-2; 42 43 int xthreads = 16; 44 int ythreads = 16; 45</pre>	Id Target Id * 1 Thread 0x7f3a02b7c8c0 (LWP 151575) "Jacobi_hip" LocalLaplacian (grid=, mes h=, stream=0x789710, d_U=0x7f3893e00000, d_AU=0x7f388bc00000) at Laplacian.cpp:40 2 Thread 0x7f39eab37700 (LWP 151591) "Jacobi_hip" 0x00007f3a035b9aff_in_poll ()
<pre>dim3 threads(xthreads,ythreads,1); dim3 blocks((localNx+xthreads-1)/xthreads,</pre>	<pre>from /lib/x86_64-linux-gnu/libc.so.6</pre>
50 hipLaunchKernelGGL(LocalLaplacianKernel, 51 blocks, 52 threads, 53 0, stream.	<pre>) from /lib/x86_64-linux-gnu/libc.so.6 6 Thread 0x7f39e857f700 (LWP 151595) "Jacobi hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 7 Thread 0x7f39e8446700 (LWP 151596) "Jacobi hip" 0x00007f3a04d23678 in do_fute</pre>
54 localNx, localNy, mesh.Nx, 55 mesh.dx, mesh.dy, 56 d U, d AU); /home/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0

Other tricks: switching between host threads

Switching between the threads gives different stack traces

Ē			jychang48@jychang48-workstati	on: ~/Downloads/hiptutorial/hip Q = - • 8
5	0x00007f3a035b9abe <+14>:	jne	0x7f3a035b9ad0 <poll+32></poll+32>	
6	0x00007f3a035b9ac0 <+16>:	mov	\$0x7,%eax	:3:hip memory.cpp :331 : 43544719486 us: 151575: [7f3a02b7c8c0] hipMemcpy:
7	0x00007f3a035b9ac5 <+21>:	syscal	l	Returned hipSuccess : : duration: 957 us
8	0x00007f3a035b9ac7 <+23>:	cmp	\$0xffffffffffff000,%rax	Iteration: 0 - Residual: 0.022108
9	0x00007f3a035b9acd <+29>:	ja	0x7f3a035b9b20 <poll+112></poll+112>	:3:hip device runtime.cpp
10	0x00007f3a035b9acf <+31>:	ret		ynchronize ()
11	0x00007f3a035b9ad0 <+32>:	sub	\$ <mark>0x28</mark> ,%rsp	:3:hip_device_runtime.cpp :470 : 43544719512 us: 151575: [7f3a02b7c8c0] hipDeviceS
12	0x00007f3a035b9ad4 <+36>:	mov	%edx, <mark>0x1c</mark> (%rsp)	ynchronize: Returned hipSuccess :
13	0x00007f3a035b9ad8 <+40>:	mov	%rsi, <mark>0x10</mark> (%rsp)	:3:hip event.cpp :310 : 43544719523 us: 151575: [7f3a02b7c8c0] hipEventRe
14	0x00007f3a035b9add <+45>:	mov	%rdi, <mark>0x8</mark> (%rsp)	cord (event:0x948d00, stream:0x789710)
15	0x00007f3a035b9ae2 <+50>:	call		:3:hip_event.cpp :352 : 43544719530 us: 151575: [7f3a02b7c8c0] hipEventRe
16	0x00007f3a035b9ae7 <+55>:	mov	<pre>0xlc(%rsp),%edx</pre>	cord: Returned hipSuccess :
17	0x00007f3a035b9aeb <+59>:	mov	0x10 (%rsp),%rsi	
18	0x00007f3a035b9af0 <+64>:	mov	%eax,%r8d	Thread 1 "Jacobi_hip" hit Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0
19	0x00007f3a035b9af3 <+67>:	mov	0x8(%rsp),%rdi	x789710, d_U=0x7f3893e00000, d_AU=0x7f388bc00000) at Laplacian.cpp:40
20	0x00007f3a035b9af8 <+72>:	mov	\$ <mark>0x7</mark> ,%eax	(gdb) i th
21	0x00007f3a035b9afd <+77>:	syscal	l	Id Target Id Frame
22	> 0x00007f3a035b9aff <+79>:	cmp	\$0xffffffffffff000,%rax	* 1 Thread 0x7f3a02b7c8c0 (LWP 151575) "Jacobi_hip" LocalLaplacian (grid=, mes
23	0x00007f3a035b9b05 <+85>:	ja	0x7f3a035b9b32 <poll+130></poll+	h=, stream=0x789710, d_U=0x7f3893e00000, d_AU=0x7f388bc000000) at Laplacian.cpp:40
24	0x00007f3a035b9b07 <+87>:	mov	%r8d,%edi	2 Thread 0x7f39eab37700(LWP 151591)"Jacobi_hip" 0x00007f3a035b9aff in poll()
25	0x00007f3a035b9b0a <+90>:	mov	%eax, <mark>0x8</mark> (%rsp)	from /lib/x86_64-linux-gnu/libc.so.6
26	0x00007f3a035b9b0e <+94>:	call		3 Thread 0x7f39ea179700(LWP 151592)"Jacobi_hip" 0x00007f3a035c65ce in epoll_w
27	0x00007f3a035b9b13 <+99>:	mov	0x8(%rsp),%eax	ait () from /lib/x86_64-linux-gnu/libc.so.6
28	0x00007f3a035b9b17 <+103>:	add	\$ <mark>0x28</mark> ,%rsp	4 Thread 0x7f39e253e700(LWP 151593)"Jacobi_hip" 0x00007f3a035bb50b in ioctl(
29	0x00007f3a035b9b1b <+107>:	ret) from /lib/x86_64-linux-gnu/libc.so.6
30	0x00007f3a035b9b1c <+108>:	nopl	<mark>0x0</mark> (%rax)	6 Thread 0x7f39e857f700(LWP 151595)"Jacobi_hip" 0x00007f3a04d23678 in do_fute
31	0x00007f3a035b9b20 <+112>:	mov	0xd5349(%rip),%rdx # 0x7f3a0368ee70	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
32	0x00007f3a035b9b27 <+119>:	neg	%eax	7 Thread 0x7f39e8446700 (LWP 151596) "Jacobi_hip" 0x00007f3a04d23678 in do_fute
33	0x00007f3a035b9b29 <+121>:	mov	%eax,%fs:(%rdx)	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
34	0x00007f3a035b9b2c <+124>:	mov	\$0xffffffff,%eax	8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute
35	0x00007f3a035b9b31 <+129>:	ret		x wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
36	0x00007f3a035b9b32 <+130>:	mov	0xd5337(%rip),%rdx	(gdb) t 2
37	0x00007f3a035b9b39 <+137>:	neg	%eax	[Switching to thread 2 (Thread 0x7f39eab37700 (LWP 151591))]
38	0x00007f3a035b9b3b <+139>:	mov	%eax,%ts:(%rdx)	#0 0x00007†3a035b9aff in poll () from /lib/x86_64-linux-gnu/libc.so.6
** Du	mp of assembler code for funct	ion pol	l: (7†3a035b9ab0 - 7f3a035b9b43) **	(gdb)

Other tricks: switching between host threads

Switching between the threads gives different stack traces

F			jychang48@jychang48-workstat	ion: ~/Downloads/hiptutorial/hip
9	0x00007f3a035c658a <+26>:	cmp	\$0xffffffffffff000,%rax	Iteration: 0 - Residual: 0.022108
10	0x00007f3a035c6590 <+32>:	ja	0x7f3a035c65f0 <epoll wait+128=""></epoll>	:3:hip device runtime.cpp :458 : 43544719507 us: 151575: [7f3a02b7c8c0] hipDeviceS
11	0x00007f3a035c6592 <+34>:	ret	· _	ynchronize ()
12	0x00007f3a035c6593 <+35>:	nopl	<pre>0x0(%rax,%rax,1)</pre>	:3:hip device runtime.cpp :470 : 43544719512 us: 151575: [7f3a02b7c8c0] hipDeviceS
13	0x00007f3a035c6598 <+40>:	sub	\$ <mark>0x28</mark> ,%rsp	ynchronize: Returned hipSuccess :
14	0x00007f3a035c659c <+44>:	mov	%edx, 0x18(%rsp)	:3:hip event.cpp :310 : 43544719523 us: 151575: [7f3a02b7c8c0] hipEventRe
15	0x00007f3a035c65a0 <+48>:	mov	%rsi, <mark>0x10</mark> (%rsp)	cord (_event:0x948d00, stream:0x789710)
16	0x00007f3a035c65a5 <+53>:	mov	%edi, <mark>0xc</mark> (%rsp)	:3:hip event.cpp :352 : 43544719530 us: 151575: [7f3a02b7c8c0] hipEventRe
17	0x00007f3a035c65a9 <+57>:	mov	%ecx, 0xlc(%rsp)	cord: Returned hipSuccess :
18	0x00007f3a035c65ad <+61>:	call		
19	0x00007f3a035c65b2 <+66>:	mov	<pre>0xlc(%rsp),%r10d</pre>	Thread 1 "Jacobi hip" hit Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0
20	0x00007f3a035c65b7 <+71>:	mov	0x18(%rsp),%edx	x789710, d U=0x7f3893e00000, d AU=0x7f388bc00000) at Laplacian.cpp:40
21	0x00007f3a035c65bb <+75>:	mov	%eax,%r8d	(gdb) i th
22	0x00007f3a035c65be <+78>:	mov	0x10 (%rsp),%rsi	Id Target Id Frame
23	0x00007f3a035c65c3 <+83>:	mov	Oxc(%rsp),%edi	* 1 Thread 0x7f3a02b7c8c0 (LWP 151575) "Jacobi hip" LocalLaplacian (grid=, mes
24	0x00007f3a035c65c7 <+87>:	mov	\$ <mark>0xe8</mark> ,%eax	h=, stream=0x789710, d U=0x7f3893e00000, d AU=0x7f388bc00000) at Laplacian.cpp:40
25	0x00007f3a035c65cc <+92>:	syscal	.1	2 Thread 0x7f39eab37700 (LWP 151591) "Jacobi hip" 0x00007f3a035b9aff in poll ()
26>	> 0x00007f3a035c65ce <+94>:	cmp	\$ <mark>0xfffffffffffff000</mark> ,%rax	from /lib/x86 64-linux-gnu/libc.so.6
27	0x00007f3a035c65d4 <+100>:	ja	0x7f3a035c6602 <epoll_wait+146></epoll_wait+	3 Thread 0x7f39ea179700 (LWP 151592) "Jacobi_hip" 0x00007f3a035c65ce in epoll_w
28	0x00007f3a035c65d6 <+102>:	mov	%r8d,%edi	ait () from /lib/x86_64-linux-gnu/libc.so.6
29	0x00007f3a035c65d9 <+105>:	mov	%eax, <mark>0xc</mark> (%rsp)	4 Thread 0x7f39e253e700 (LWP 151593) "Jacobi_hip" 0x00007f3a035bb50b in ioctl (
30	0x00007f3a035c65dd <+109>:	call) from /lib/x86_64-linux-gnu/libc.so.6
31	0x00007f3a035c65e2 <+114>:	mov	<code>0xc(%rsp),%eax</code>	6 Thread 0x7f39e857f700(LWP 151595)"Jacobi_hip" 0x00007f3a04d23678 in do_fute
32	0x00007f3a035c65e6 <+118>:	add	\$ <mark>0x28</mark> ,%rsp	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
33	0x00007f3a035c65ea < <u>+122>:</u>	ret		7 Thread 0x7f39e8446700 (LWP 151596) "Jacobi_hip" 0x00007f3a04d23678 in do_fute
34	0x00007f3a035c65eb <+123>:	nopl	0x0(%rax,%rax,1)	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
35	0x00007f3a035c65f0 <+128>:	mov	0xc8879 (%rip),%rdx	8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute
36	0x00007f3a035c65f7 <+135>:	neg	%eax	x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
37	0x00007f3a035c65f9 <+137>:	mov	%eax,%fs:(%rdx)	(gdb) t 2
38	0x00007f3a035c65fc <+140>:	mov	\$ <mark>0xffffffff</mark> ,%eax	[Switching to thread 2 (Thread 0x7f39eab37700 (LWP 151591))]
39	0x00007f3a035c6601 <+145>:	ret		#0 0x00007f3a035b9aff in poll () from /lib/x86_64-linux-qnu/libc.so.6
40	0x00007f3a035c6602 <+146>:	mov	0xc8867 (%rip),%rdx	(gdb) t 3
41	0x00007f3a035c6609 <+153>:	neg	%eax	[Switching to thread 3 (Thread 0x7f39ea179700 (LWP 151592))]
42	0x00007f3a035c660b <+155>:	mov	%eax,%fs:(%rdx)	#0 0x00007f3a035c65ce in epoll_wait () from /lib/x86_64-linux-gnu/libc.so.6
** Dur	np of assembler code for func	ion epo	oll_wait: (7f3a035c6570 - 7f3a035c6613) **	(gdb)

Other tricks: switching between host threads

Switching between the threads gives different stack traces

F	jychang48@jychang48-workstatio	on: ~/Downloads/hiptutorial/hip 🛛 🔍 = 💷 😣
1 1 1 1 1 1		:3:hip_device_runtime.cpp :470 : 43544719512 us: 151575: [7f3a02b7c8c0] hipDeviceS ynchronize: Returned hipSuccess : :3:hip_event.cpp :310 : 43544719523 us: 151575: [7f3a02b7c8c0] hipEventRe cord (event:0x948d00, stream:0x789710) :3:hip_event.cpp :352 : 43544719530 us: 151575: [7f3a02b7c8c0] hipEventRe cord: Returned hipSuccess :
~ ~ ~ 12345 678910111213 ~ ~ ~	<pre>Dump of assembler code for function ioctl: 0x00007f3a035bb500 <+0>: endbr64 0x00007f3a035bb504 <+4>: mov \$0x10,%eax 0x00007f3a035bb509 <+9>: syscall > 0x00007f3a035bb50b <+11>: cmp \$0xffffffffff001,%rax 0x00007f3a035bb511 <+17>: jae 0x7f3a035bb514 <ioctl+20> 0x00007f3a035bb513 <+19>: ret 0x00007f3a035bb514 <+20>: mov 0xd3955(%rip),%rcx # 0x7f3a0368ee70 0x00007f3a035bb51b <+27>: neg %eax 0x00007f3a035bb51d <+29>: mov %eax,%fs:(%rcx) 0x00007f3a035bb51d <+29>: mov %eax,%fs:(%rcx) 0x00007f3a035bb52d <+30>: ret End of assembler dump.</ioctl+20></pre>	Thread 1 "Jacobi_hip" hit Breakpoint 1, LocalLaplacian (grid=, mesh=, stream=0 x789710, d_U=0x7f3893e00000, d_AU=0x7f388bc00000) at Laplacian.cpp:40 (gdb) i th Id Target Id Frame * 1 Thread 0x7f3a02b7c8c0 (LWP 151575) "Jacobi_hip" LocalLaplacian (grid=, mes h=, stream=0x789710, d_U=0x7f3893e00000, d_AU=0x7f388bc00000) at Laplacian.cpp:40 2 Thread 0x7f39eab37700 (LWP 151591) "Jacobi_hip" 0x00007f3a035b9aff in poll () from /lib/x86_64-linux-gnu/libc.so.6 3 Thread 0x7f39ea179700 (LWP 151592) "Jacobi_hip" 0x00007f3a035c65ce in epoll_w ait () from /lib/x86_64-linux-gnu/libc.so.6 4 Thread 0x7f39e253e700 (LWP 151593) "Jacobi_hip" 0x00007f3a035bb50b in ioctl () from /lib/x86_64-linux-gnu/libc.so.6 6 Thread 0x7f39e857f700 (LWP 151595) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 7 Thread 0x7f39e8446700 (LWP 151596) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0 8 Thread 0x7f39e827f700 (LWP 151597) "Jacobi_hip" 0x00007f3a04d23678 in do_fute x_wait.constprop () from /lib/x86_64-linux-gnu/libpthread.so.0
* * * * * * *	Dump of assembler code for function ioctl: (7f3a035bb500 - 7f3a035bb524) **	<pre>#0 0x00007f3a035b9aff in poll () from /lib/x86_64-linux-gnu/libc.so.6 (gdb) t 3 [Switching to thread 3 (Thread 0x7f39ea179700 (LWP 151592))] #0 0x00007f3a035c65ce in epoll wait () from /lib/x86_64-linux-gnu/libc.so.6 (gdb) t 4 [Switching to thread 4 (Thread 0x7f39e253e700 (LWP 151593))] #0 0x00007f3a035bb50b in ioctl () from /lib/x86_64-linux-gnu/libc.so.6 (gdb)</pre>

42 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved

What to do if your program returns an error?

In a perfect world, all code should run successfully like this ...

Я	jychang48@jychang48-workstation: ~/Downloads/hiptutorial/hip	Q =	- 1	• 😣
jychang48@jychang48-workstation:~	-/Downloads/hiptutorial/hip\$./Jacobi hip -g 1 1			
Topology size: 1 x 1				
Local domain size (current node):	4096 x 4096			
Global domain size (all nodes): 4	4096 x 4096			
Rank 0 selecting device 0 on host	jychang48-workstation			
Starting Jacobi run.				
Iteration: 0 - Residual: 0.0221	108			
Iteration: 100 - Residual: 0.0006	525			
Iteration: 200 - Residual: 0.0003	871			
Iteration: 300 - Residual: 0.0002	274			
Iteration: 400 - Residual: 0.0002	221			
Iteration: 500 - Residual: 0.0001	87			
Iteration: 600 - Residual: 0.0001				
Iteration: 700 - Residual: 0.000				
Iteration: 800 - Residual: 0.000				
Iteration: 900 - Residual: 0.000				
Iteration: 1000 - Residual: 0.000				
Stopped after 1000 iterations wit				
Total Jacobi run time: 2.0454 Sec	(-1)			
Measured ELOPS, 130 44 CELOPS (to	L_{0} (Local), 0.20 dL()/S (per process)			
Measured device bandwidth: 787 42	GB(s (tata)) 787 A2 GB(s (per process))			
ivchang/8@ivchang/8_workstation:	(Downloads/), 101.42 (D)s (per process)			
Jychang+o@Jychang+o-workstatton.~	/ bown coaus, mip catch i ac/ mip s			

What to do if your program returns an error?

... but every so often we come across GPU errors like this

A	jychang48@jychang48-workstation: ~/Downloads/hiptutorial/hip	Q =		ı 😣
<pre>in jychang48@jychang48-workstation:~ Topology size: 1 x 1 Local domain size (current node): Global domain size (all nodes): 4 Rank 0 selecting device 0 on host Starting Jacobi run. Iteration: 0 - Residual: 0 0221 Memory access fault by GPU node-1 (jychang48-workstation:133929) si [jychang48-workstation:133929] Si [jychang48-workstation:133929] [jychang48-workstation:133929] ** Aborted (core dumped) jychang48@jychang48-workstation:</pre>	<pre>/bownloads/hiptutorial/hip\$./Jacobi_hip -g 1 1 4096 x 4096 (b) 06 x 4096 (c) 19 ychang48-workstation // (Agent handle: 0x121b750) on address 0x7f2c6fe00000. Reason: Page not present or supervisor privilege. ***********************************</pre>	Q =	_ E	

What to do if your program returns an error?

Launch rocgdb and invoke the options shown in the picture:

المعالي jychang48@jychang48-workstation: ~/Downloads/hiptutorial/hip		Q = _ = &	
1 2 3 4 5 6 7 8 9 0 0 1 1 2 2 3 4 5 6 7 8 9 0 0 1 1 2 2 3 4 4 5 5 6 7 7 8 9 0 0 1 1 2 2 3 3 4 4 5 5 6 7 7 8 9 0 0 1 1 2 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 2 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 2 7 3 3 4 5 5 6 7 7 8 9 0 0 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1 3 1	<pre>//***********************************</pre>	<pre>GNU gdb (rocm-rel-4.2-21) 19.1 Copyright (C) 2020 Free Software Foundation, Inc. License GPLV3+: GNU GPL version 3 or later <http: gg<br="" gnu.org="" licenses="">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="" issues="" rocgdb="" rocm-developer-tools="">. Find the GDB manual and other documentation resources online at: <http: documentation="" gdb="" software="" www.gnu.org=""></http:>.</https:></http:></pre> For help, type "help". Type "apropos word" to search for commands related to "word" Reading symbols from _/lacobi hip (gdb) set pagination off (gdb) set non-stop on (gdb) b abort Function "abort" not defined. Make breakpoint 1 (abort) pending. (gdb) ■	ol.html>
	c, j j enang ic, com coaco, it pracor tac, it p, bacobtilatin opp		

What to do if your program returns an error?

We now see the offending line which invokes the error. Uncommenting out lines 20 and 31 will fix this particular issue.

jychang48@jychang48-workstation: ~/Downloads/hiptutorial/hip		on: ~/Downloads/hiptutorial/hip 🛛 🔤 – 🗉 😣
12 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43	<pre>const dfloat dx,</pre>	For help, type "help". Type "apropos word" to search for commands related to "word" Reading symbols from ./lacobi_hip (gdb) set pagination off (gdb) set non-stop on (gdb) b abort Function "abort" not defined. Make breakpoint pending on future shared library load? (y or [n]) y Breakpoint 1 (abort) pending. (gdb) run Starting program: /home/jychang48/Downloads/hiptutorial/hip/Jacobi_hip -g 1 1 [Thread debugging using libthread_db enabled] Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1". [Detaching after fork from child process 134151] [New Thread 0x7f3d1c486700 (LWP 134159)] Topology size: 1 x 1 Local domain size (current node): 4096 x 4096 Global domain size (current node): 4096 x 4096 [New Thread 0x7f3d152d700 (LWP 134160)] Rak 0 selecting device 0 on host jychang48-workstation [New Thread 0x7f3d15544700 (LWP 134161)] [Thread 0x7f3d5564700 (LWP 134161)] [Thread 0x7f3d15544700 (LWP 134161)] [Thread 0x7f3d15544700 (LWP 134161)] [New Thread 0x7f3d15544700 (LWP 134161)] [New Thread 0x7f3d15364700 (LWP 134162)] [New Thread 0x7f3d1548f700 (LWP 134161)] [Thread 0x7f3d15544700 (LWP 134162)] [New Thread 0x7f3d15544700 (LWP 134162)] [New Thread 0x7f3d1548f700 (LWP 134162)] [New Thread 0x7f3d15544700 (LWP 134162)] [New Thread 0x7f3d1548f700 (LWP 134162)] [New Thread 0x7f3d1548f700 (LWP 134162)] [New Thread 0x7f3d148f700 (LWP 134162)] [New Thread 0x7f3d1548f700 (LWP 134162)] [New Thread 0x7f3d148f700 (LWP 134162)] [New Thread
44 45 /home	e/jychang48/Downloads/hiptutorial/hip/Laplacian.cpp	_64-linux-gnu/libc.so.6 (gdb)

And more...

ROCgdb has several other features and capabilities not covered in this presentation. See the following for much more:

- https://github.com/RadeonOpenCompute/ROCm/blob/master/ROCm_Debugger_User_Guide_v4.2.pdf
- /opt/rocm-4.2.0/share/doc/rocgdb/rocannotate.pdf
- /opt/rocm-4.2.0/share/doc/rocgdb/rocgdb.pdf
- /opt/rocm-4.2.0/share/doc/rocgdb/rocrefcard.pdf
- /opt/rocm-4.2.0/share/doc/rocgdb/rocstabs.pdf



Part 2: Math Libraries

48 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved

Decoder ring: Math library equivalents

CUBLAS	ROCBLAS	Basic Linear Algebra Subroutines
CUFFT	ROCFFT	Fast Fourier Transforms
THRUST	ROCTHRUST	Deep Learning Library
CUB	ROCPRIM	Optimized Parallel Primitives
EIGEN	EIGEN	C++ Template Library for Linear Algebra

MORE INFO AT: GITHUB.COM/ROCM-DEVELOPER-TOOLS/HIP → HIP_PORTING_GUIDE.MD

49 | ROCgdb and HIP Math Libraries | ORNL Hackathon, May 24 – 26, 2021 | © Advanced Micro Devices Inc., All Rights Reserved

AMD GPU Math Libraries

- A note on naming conventions:
 - roc* -> AMGCN library usually written in HIP
 - cu* -> NVIDIA PTX libraries
 - hip* -> usually interface layer on top of roc*/cu* backends
- hip* libraries:
 - Can be compiled by hipcc and can generate a call for the device you have:
 - hipcc->clang->AMD GCN ISA
 - hipcc->nvcc (inlined)->NVPTX
 - Just a thin wrapper that marshals calls off to a "backend" library:
 - corresponding roc* library backend containing optimized GCN
 - corresponding cu* library backend containing NVPTX for NVIDIA devices
 - E.g., hipBLAS is a marshalling library:



AMI)

AMD GPU Libraries: BLAS

- rocBLAS `sudo apt install rocblas`
 - Source code: <u>https://github.com/ROCmSoftwarePlatform/rocBLAS</u>
 - Documentation: <u>https://rocblas.readthedocs.io/en/latest/</u>
 - Basic linear algebra functionality
 - axpy, gemv, trsm, etc
 - Use hipBLAS if you need portability between AMD and NVIDIA devices
- hipBLAS `sudo apt install hipblas`
 - Documentation: <u>https://github.com/ROCmSoftwarePlatform/hipBLAS/wiki/Exported-functions</u>
 - Use this if you need portability between AMD and NVIDIA
 - It is just a thin wrapper:
 - It can dispatch calls to rocBLAS for AMD devices
 - It can dispatch calls to cuBLAS for NVIDIA devices



AMD GPU Libraries: rocBLAS example

- rocBLAS
 - Documentation: <u>https://rocblas.readthedocs.io/e</u> <u>n/latest/</u>
 - Level 1, 2, and 3 functionality
 - axpy, gemv, trsm, etc
 - Note: rocBLAS syntax matches BLAS closer than hipBLAS or cuBLAS
 - Use hipBLAS only if you need portability between AMD and NVIDIA devices
 - Link with: -lrocblas

#include <rocblas.h>

```
int main(int argc, char ** argv) {
   rocblas_int N = 500000;
```

```
// Allocate device memory
double * dx, * dy;
hipMalloc(&dx, sizeof(double) * N);
hipMalloc(&dy, sizeof(double) * N);
```

```
// Allocate host memory (and fill up the arrays) here
std::vector<double> hx(N), hy(N);
```

```
// Copy host arrays to device
hipMemcpy(dx, hx.data(), sizeof(double) * N, hipMemcpyHostToDevice);
hipMemcpy(dy, hy.data(), sizeof(double) * N, hipMemcpyHostToDevice);
```

```
const double alpha = 1.0;
rocblas_handle handle;
rocblas_create_handle(&handle);
rocblas_status status;
status = rocblas_daxpy(handle, N, &alpha, dx, 1, dy, 1);
rocblas_destroy_handle(handle);
```

```
// Copy result back to host
hipMemcpy(hy.data(), dy, sizeof(double) * N, hipMemcpyDeviceToHost);
hipFree(dx);
hipFree(dy);
return 0;
```

AMD GPU Libraries: FFT

- rocFFT `sudo apt install rocfft`
 - Source code: <u>https://github.com/ROCmSoftwarePlatform/rocFFT</u>
 - Documentation: <u>https://rocfft.readthedocs.io/en/latest/</u>
 - Implementation of Discrete Fourier Transforms (DFT), leveraging mathematical symmetries to reduce algorithmic complexity from O(N²) to O(N log N)
 - Use hipFFT (`sudo apt install hipfft`) if you need portability between AMD and NVIDIA devices
- Basic steps in rocFFT:
 - 1. Initialize the library by calling rocfft_setup()
 - 2. Create a plan for the FFT needed
 - 3. Optionally allocate a work buffer
 - 4. Execute the plan
 - 5. Free the work buffer if needed
 - 6. Destroy the plan
 - 7. Terminate the library by calling rocfft_cleanup()



AMD GPU Libraries: rocFFT example

```
#include <iostream>
#include <vector>
#include "hip/hip_runtime_api.h"
#include "hip/hip_vector_types.h"
#include "rocfft.h"
```

int main()

```
{
```

rocfft_setup();

```
// Create HIP device buffer
size_t N = 16;
size_t Nbytes = N * sizeof(float2);
float2 *x;
hipMalloc(&x, Nbytes);
```

```
// Initialize data
std::vector<float2> cx(N);
for (size_t i = 0; i < N; i++) {
    cx[i].x = 1; cx[i].y = -1;
}</pre>
```

// Copy data to device hipMemcpy(x, cx.data(), Nbytes, hipMemcpyHostToDevice);

```
// Create rocFFT plan
rocfft_plan plan = nullptr;
size_t length = N;
rocfft_plan_create(&plan, rocfft_placement_inplace,
    rocfft_transform_type_complex_forward, rocfft_precision_single,
    1, &length, 1, nullptr);
```

```
// Check if the plan requires a work buffer
size_t work_buf_size = 0;
rocfft_plan_get_work_buffer_size(plan, &work_buf_size);
void* work_buf = nullptr;
rocfft_execution_info info = nullptr;
if(work_buf_size) {
   rocfft_execution_info_create(&info);
   hipMalloc(&work_buf, work_buf_size);
   rocfft_execution_info_set_work_buffer(info, work_buf, work_buf_size);
}
```

```
// Execute plan
rocfft_execute(plan, (void**) &x, nullptr, info);
hipDeviceSynchronize();
```

```
// Clean up work buffer
if(work_buf_size) {
    hipFree(work_buf);
    rocfft_execution_info_destroy(info);
}
```

```
// Destroy plan
rocfft_plan_destroy(plan);
```

// Copy result back to host
std::vector<float2> y(N);
hipMemcpy(y.data(), x, Nbytes, hipMemcpyDeviceToHost);

```
// Free device buffer
hipFree(x);
rocfft_cleanup();
return 0;
```

}

Some Links to Key Libraries

- BLAS
 - rocBLAS (<u>https://github.com/ROCmSoftwarePlatform/rocBLAS</u>)
 - hipBLAS (<u>https://github.com/ROCmSoftwarePlatform/hipBLAS</u>)
- FFTs
 - rocFFT (<u>https://github.com/ROCmSoftwarePlatform/rocFFT</u>)
- Random number generation
 - rocRAND (<u>https://github.com/ROCmSoftwarePlatform/rocRAND</u>)
 - hipRAND (<u>https://github.com/ROCmSoftwarePlatform/hipRAND</u>)
- Sparse linear algebra
 - rocSPARSE (<u>https://github.com/ROCmSoftwarePlatform/rocSPARSE</u>)
 - hipSPARSE (<u>https://github.com/ROCmSoftwarePlatform/hipSPARSE</u>)
- Iterative solvers
 - rocALUTION (<u>https://github.com/ROCmSoftwarePlatform/rocALUTION</u>)
- Parallel primitives
 - rocPRIM (<u>https://github.com/ROCmSoftwarePlatform/rocPRIM</u>)
 - hipCUB (<u>https://github.com/ROCmSoftwarePlatform/hipCUB</u>)

More links to key libraries

Machine Learning libraries and Frameworks

- Tensorflow: <u>https://github.com/ROCmSoftwarePlatform/tensorflow-upstream</u>
- Pytorch: <u>https://github.com/ROCmSoftwarePlatform/pytorch</u>
- MIOpen (similar to cuDNN): <u>https://github.com/ROCmSoftwarePlatform/MIOpen</u>
- Tensile: https://github.com/ROCmSoftwarePlatform/Tensile
- RCCL (ROCm analogue of NCCL): <u>https://github.com/ROCmSoftwarePlatform/rccl</u>

DISCLAIMER

The information presented in this document is for informational purposes only and may contain technical inaccuracies, omissions, and typographical errors. The information contained herein is subject to change and may be rendered inaccurate for many reasons, including but not limited to product and roadmap changes, component and motherboard version changes, new model and/or product releases, product differences between differing manufacturers, software changes, BIOS flashes, firmware upgrades, or the like. Any computer system has risks of security vulnerabilities that cannot be completely prevented or mitigated. AMD assumes no obligation to update or otherwise correct or revise this information. However, AMD reserves the right to revise this information and to make changes from time to time to the content hereof without obligation of AMD to notify any person of such revisions or changes.

THIS INFORMATION IS PROVIDED 'AS IS." AMD MAKES NO REPRESENTATIONS OR WARRANTIES WITH RESPECT TO THE CONTENTS HEREOF AND ASSUMES NO RESPONSIBILITY FOR ANY INACCURACIES, ERRORS, OR OMISSIONS THAT MAY APPEAR IN THIS INFORMATION. AMD SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. IN NO EVENT WILL AMD BE LIABLE TO ANY PERSON FOR ANY RELIANCE, DIRECT, INDIRECT, SPECIAL, OR OTHER CONSEQUENTIAL DAMAGES ARISING FROM THE USE OF ANY INFORMATION CONTAINED HEREIN, EVEN IF AMD IS EXPRESSLY ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

AMDJ