ROCgdb Documentation

Release 16.3

Advanced Micro Devices, Inc.

Mar 09, 2026

INSTALL

This is the documentation for AMD ROCm Debugger (ROCgdb) for Linux, which is the AMD source-level debugger based on the GNU Debugger (GDB). For documentation on ROCgdb for Windows, see AMD ROCm debugger for Windows. ROCgdb enables heterogeneous debugging on the ROCm software that consists of an x86-based host architecture along with commercially available AMD GPU architectures supported by the AMD Debugger API Library (ROCdbgapi). ROCdbgapi is included with ROCm.

ROCgdb provides the following features:

- Debugs ROCm applications running on AMD GPU-supported hardware.
- Debugs applications without the potential variations introduced by simulation and emulation environments.
- Offers a seamless debugging environment that allows simultaneous GPU and CPU code debugging within the same application, just like programming in HIP, which is a seamless extension of C++ programming.
- Additional features to support debugging ROCm device code on top of the existing GDB debugging features, which are inherently present for debugging the host code.
- Supports HIP kernel debugging.
- Allows you to set breakpoints, single-step ROCm applications, and inspect and modify the memory and variables of any given thread running on the hardware.

The code is open source and hosted at: https://github.com/ROCm/ROCgdb

Install

- *Installation*

Quick reference

- *Quick start*
- *Commands for key operations*

How to

-
- *Setting up third-party tools*
- *Debugging Python kernel code*

To contribute to the documentation, refer to Contributing to ROCm.

You can find licensing information on the Licensing page.

CHAPTER

ONE

INSTALLING ROCGDB

This topic provides information required to build and install ROCgdb.

1.1 System requirements

- A system supporting ROCm. See the supported operating systems.
- A C++17 compiler such as GCC 9 or Clang 5.
- AMD Debugger API Library (`ROCdbgapi`) that can be installed as part of the ROCm release using the `rocm-dbgapi` package.
- Install the required packages according to the OS:

Ubuntu

```
apt install bison flex gcc make ncurses-dev texinfo g++ zlib1g-dev \
libexpat-dev python3-dev liblzma-dev libgmp-dev libmpfr-dev
```

RHEL

```
yum install -y epel-release centos-release-scl bison flex gcc make \
texinfo texinfo-tex gcc-c++ zlib-devel expat-devel python3-devel \
xz-devel gmp-devel ncurses-devel mpfr-devel
```

SLES

```
zypper in bison flex gcc make texinfo gcc-c++ zlib-devel libexpat-devel \
python3-devel xz-devel gmp-devel ncurses-devel mpfr-devel
```

Note: ROCgdb might become unresponsive in SELinux-enabled distributions. To learn more about this issue, see installation troubleshooting.

1.2 Building ROCgdb

An example command line to build ROCgdb on Linux:

```
cd rocgdb
mkdir build
cd build
../configure --program-prefix=roc \
--enable-64-bit-bfd --enable-targets="x86_64-linux-gnu,amdgcn-amd-amdhsa" \
--disable-ld --disable-gas --disable-gdbserver --disable-sim --enable-tui \
--disable-gdbtk --disable-gprofng --disable-shared --with-expat \
--with-system-zlib --without-guile --without-babeltrace --with-lzma \
--with-python=python3
make
```

If `ROCdbgapi` is not installed in the system's default location, specify `PKG_CONFIG_PATH` to make the correct build configuration available to `pkg-config`. If `ROCdbgapi` is installed in `/opt/rocm-$ROCM_VERSION` (default for ROCm packages), use `PKG_CONFIG_PATH=/opt/rocm-$ROCM_VERSION/share/pkgconfig`.

If the system's dynamic linker is not configured to locate `ROCdbgapi` where it is installed, configure and build ROCgdb using `LDFLAGS="-Wl,-rpath=/opt/rocm-$ROCM_VERSION/lib"`. Alternatively, use `LD_LIBRARY_PATH` at runtime to indicate where `ROCdbgapi` is installed.

You can find the built ROCgdb executable in `build/gdb/gdb` and the user manual in `build/gdb/doc/gdb.info`.

1.3 Installing ROCgdb

To install ROCgdb, use:

```
make install
```

This installs ROCgdb in `<prefix>/bin/rocgdb`.

1.4 Installing libraries

To execute ROCgdb, you must install the `ROCdbgapi` library and its dependent `Comgr` library. These can be installed as part of the ROCm release using the `rocm-dbgapi` package:

- `librocm-dbgapi.so.0`
- `libamd_comgr.so`

To generate the ROCgdb user guide as a PDF, use:

```
make pdf
```

This generates the PDF in `build/gdb/doc/gdb.pdf`.

Note: For ROCgdb user guide in HTML format, see ROCgdb user guide.

CHAPTER

TWO

ROCGDB QUICK START

After *installing ROCgdb*, follow the *setup* to start debugging your application.

2.1 Source compilation

Before debugging, compile your software with debug information.

To compile your source with debug symbols, use:

```
$ hipcc -ggdb -O0 saxpy.cpp -o saxpy
```

Or, compile using amd-llvm:

```
amdclang++ -ggdb -O0 -x hip --offload-arch=native saxpy.cpp -o saxpy
```

Adding the `-g` flag to your compilation command generates debug information even when optimizations are turned on. Note that higher optimization levels make debugging more difficult, so it might be helpful to turn off these optimizations with the `-O0` compiler option.

For saxpy source code, see main.hip.

2.2 Debugging using ROCgdb

You can either launch and run your application under debugger control or attach the debugger to a running process and continue execution.

To start debugging your application under debugger control, follow these steps:

1. Launch your application under debugger control:

   ```
   $ rocgdb ./saxpy
   [...]
   ```

   At this point, the application is not running, but you'll have access to the debugger console. On the console, you can use any *gdb option* for host debugging along with ROCgdb-specific features for device debugging.

2. Set a breakpoint before running the application with debugger.

   ```
   tbreak my_app.cpp:458
   ```

   This places a temporary breakpoint at the specified line. To start your application, use:

```
(gdb) run
```

If the breakpoint is in the device code, the debugger shows the device and host threads. The device threads are not individual work items; instead, they represent a wavefront on the device. You can switch between the device wavefronts as you can between the host threads.

To attach the debugger to a running process and continue execution, use:

```
$ rocgdb -pid <process_id>
[...]
(gdb) continue
```

Use `ps` command to get the <process_id> of the running application, to which the debugger needs to be attached.

You can also switch between layouts, which allows you to use different layouts for different situations while debugging.

```
layout src
layout asm
```

The `src` layout is the source code view, while the `asm` is the assembly view. For more layouts, see TUI-specific commands.

After starting or attaching your application with the debugger, you can utilize these *ROCgdb commands for key operations* to perform further operations.

2.3 ROCgdb user guide

The ROCgdb user guide provides detailed information about using ROCgdb. This user guide is also installed in the following directories when you install ROCm:

- `/opt/rocm/share/info/rocgdb/gdb.info` as a texinfo file
- `/opt/rocm/share/doc/rocgdb/rocgdb.pdf` as a PDF file

For specific information about debugging heterogeneous programs on ROCm software, refer to the following chapters in the ROCgdb user guide:

- **Debugging Heterogeneous Programs:** Provides general information about debugging heterogeneous programs. It also discusses features and commands that are not currently implemented but provisionally planned for future versions.
- **Configuration-Specific Information > Architectures > AMD GPU:** Provides specific information about debugging heterogeneous programs on ROCm software with supported AMD GPU hardware. This section also lists the implementation status and known issues of the current version.

You can use the standard GDB commands for both CPU and GPU code debugging.

CHAPTER

THREE

ROCGDB COMMANDS FOR KEY OPERATIONS

This topic summarizes the ROCgdb commands for key operations.

3.1 Inspecting kernel state

Here are the commands used to inspect the kernel state:

3.1.1 View kernel code

```
(gdb) list
```

Sample output:

```
1  #include <hip/hip_runtime.h>
2  #include <algorithm>
3  #include <iostream>
4  #include <numeric>
5  #include <vector>
6  #include <cstddef>
8  __global__ void saxpy_kernel(const float a, const float* d_x, float* d_y, const unsigned int size)
9  {
10      // Compute the current thread's index in the grid.
11      const unsigned int global_idx = blockIdx.x * blockDim.x + threadIdx.x;
12      // The grid can be larger than the number of items in the vectors. Avoid out-of-bounds addressing.
13      if(global_idx < size)
14      {
15          d_y[global_idx] = a * d_x[global_idx] + d_y[global_idx];
16      }
17  }
18  int main()
19  {
....
99  }
```

3.1.2 View disassembly

```
(gdb) disassemble
```

Sample output:

```
Dump of assembler code for function _ZL3bari:
    0x00007ffff608e2b0 <+0>:     s_waitcnt vmcnt(0) expcnt(0) lgkmcnt(0)
    0x00007ffff608e2b4 <+4>:     s_mov_b32 s25, s33
    0x00007ffff608e2b8 <+8>:     s_mov_b32 s33, s32
    0x00007ffff608e2bc <+12>:    s_xor_saveexec_b64 s[16:17], -1
    0x00007ffff608e2c0 <+16>:    buffer_store_dword v36, off, s[0:3], s33 offset:52
    .....
    0x00007ffff608e92c <+1660>:  s_mov_b64 exec, s[4:5]
    0x00007ffff608e930 <+1664>:  s_mov_b32 s33, s25
    0x00007ffff608e934 <+1668>:  s_waitcnt vmcnt(0)
    0x00007ffff608e938 <+1672>:  s_setpc_b64 s[30:31]
End of assembler dump.
```

3.1.3 View system information

The following commands are related to heterogeneous debugging:

- **Agents:**

  The following command lists the information shown in the sample output for each heterogeneous agent:

  ```
  (gdb) info agents
  ```

  Sample output:

  ```
    Id State Target Id                    Architecture Device Name       Cores Threads␣
  ↪Location
  * 1  A      AMDGPU Agent (GPUID 35090) gfx90a       AMD Instinct MI210 416    3328     ␣
  ↪0000:4a:00.0
    2  A      AMDGPU Agent (GPUID 34915) gfx90a       AMD Instinct MI210 416    3328     ␣
  ↪0000:09:00.0
    3  A      AMDGPU Agent (GPUID 56224) gfx90a       AMD Instinct MI210 416    3328     ␣
  ↪0000:0c:00.0
    4  A      AMDGPU Agent (GPUID 33385) gfx90a       AMD Instinct MI210 416    3328     ␣
  ↪0000:11:00.0
  ```

  For more information, see info agents command.

- **Queues:**

  The following command lists the information shown in the sample output for each heterogeneous queue:

  ```
  (gdb) info queues
  ```

  Sample output:

  ```
    Id   Target Id                Type         Read   Write  Size     Address
    1    AMDGPU Queue 1:1 (QID 0) HSA          2      2      4096    ␣
  ```

(continues on next page)

(continued from previous page)

```
↪0x00007ffff626e000
* 2    AMDGPU Queue 1:2 (QID 1) HSA          0      2      1048576 ␣
↪0x00007fffe5800000
```

For more information, see info queues command.

- **Dispatches:**

  The following command lists the information shown in the sample output for each heterogeneous dispatch:

```
(gdb) info dispatches
```

  Sample output:

```
 Id   Target Id                         Grid    Workgroup Fence   Kernel Function
* 1    AMDGPU Dispatch 1:2:1 (PKID 0) [1,1,1] [1,1,1]   B|Aa|Ra kern()
```

  For more information, see info dispatches command.

- **Threads:**

  In some operating systems where a single program might have more than one thread of execution, the threads are akin to multiple processes with a shared address space but individual registers, execution stack, and perhaps private memory.

  To facilitate debugging such multi-thread programs, the following command lists the threads created on all heterogeneous agents:

```
(gdb) info threads
```

  Sample output:

```
  Id   Target Id                                   Frame
  1    Thread 0x7ffff6288180 (LWP 645917) "nosimple" 0x00007ffff207d586 in ?? ()␣
↪from /opt/rocm-7.1.0/lib/libhsa-runtime64.so.1
  2    Thread 0x7fffe81ff6c0 (LWP 645924) "nosimple" __GI___ioctl (fd=3,␣
↪request=3222817548) at ../sysdeps/unix/sysv/linux/ioctl.c:36
  4    Thread 0x7fffe61ff6c0 (LWP 645926) "nosimple" __GI___ioctl (fd=3,␣
↪request=3222817548) at ../sysdeps/unix/sysv/linux/ioctl.c:36
  6    Thread 0x7ffff5fff6c0 (LWP 645930) "nosimple" __GI___ioctl (fd=3,␣
↪request=3222817548) at ../sysdeps/unix/sysv/linux/ioctl.c:36
* 7    AMDGPU Wave 1:2:1:1 (0,0,0)/0 "saxpy"      kern () at /home/user/saxpy.cpp:7
```

  For more information, see Debugging programs with multiple threads.

- **Lanes:**

  On some heterogeneous systems there can be heterogeneous agents that support Single Instruction Multiple Data (SIMD) or Single Instruction Multiple Threads (STMT) machine instructions. On these target architectures, a single machine instruction can operate in parallel on multiple heterogeneous lanes.

  To facilitate debugging heterogeneous programs, the following command displays information about individual source language threads of execution that are mapped to SIMD-like lanes of a thread.

```
(gdb) info lanes
```

  Sample output:

```
  Id   State Target Id                                    Frame
* 0    A     AMDGPU Lane 1:2:1:1/0 (0,0,0)[0,0,0] kern () at /home/user/saxpy.cpp:7
```

For more information, see Debugging heterogeneous programs.

3.1.4 View back trace

```
(gdb) backtrace
```

Sample output:

```
#0  saxpy (tid=0) at /home/oogunbow/saxpy.cpp:33
#1  0x00007ffff608ee40 in kern () at /home/user/saxpy.cpp:4
```

3.1.5 View stack frames

```
(gdb) info frame
```

Sample output:

```
Stack level 0, frame at private_wave#0x800:
pc = 0x7ffff608e3bc in bar (/home/user/saxpy.cpp:33); saved pc = 0x7ffff608ee40
called by frame at private_wave#0x0
source language c++.
Arglist at private_wave#0x800, args: tid=0
Locals at private_wave#0x800, Saved registers:
v36 at private_wave#0x1500, v37 at private_wave#0x1600
```

3.1.6 View frame arguments

```
(gdb) info args
```

Sample output:

```
tid = 0
```

3.1.7 View frame local variables

```
(gdb) info locals
```

Sample output:

```
No locals.
```

3.1.8 View GPU registers

```
(gdb) info registers
```

This command dumps the content of the current wavefront's registers.

Sample output:

```
v0              {0x30 <repeats 64 times>}
....
s41             0x0                 0
m0              0x1008              4104
pc              0x7ffff608e3bc      0x7ffff608e3bc <saxpy(int)+268>
exec            0x5555555555555555  6148914691236517205
vcc             0xffffffffffffffff  18446744073709551615
```

This command dumps only the general-purpose registers, which provide all-inclusive data about the state of the current wavefront.

To get data for all registers, use:

```
(gdb) info all-registers
```

3.1.9 View GPU data @ address spaces

```
(gdb) x/nfu global#0xdeadbeef
(gdb) x/nfu local#0xdeadbeef
(gdb) x/nfu generic#0xdeadbeef
(gdb) x/nfu private_wave#0xdeadbeef
(gdb) x/nfu private_lane#0xdeadbeef
```

For more information, see AMD GPU address spaces.

3.1.10 View CPU/GPU threads

```
(gdb) info threads
```

Sample output:

```
  Id   Target Id                                      Frame
  1    Thread 0x7ffff648bf80 (LWP 1981864) "saxpy" 0x00007ffff5a6c9ef in ?? () from /opt/
↪rocm-7.1.0/lib/libhsa-runtime64.so.1
  2    Thread 0x7ffff55ff6c0 (LWP 1981871) "saxpy" __GI___ioctl (fd=3,␣
↪request=3222817548) at ../sysdeps/unix/sysv/linux/ioctl.c:36
  4    Thread 0x7fffeffff6c0 (LWP 1981873) "saxpy" __GI___ioctl (fd=3,␣
↪request=3222817548) at ../sysdeps/unix/sysv/linux/ioctl.c:36
  6    Thread 0x7ffff5dff6c0 (LWP 1981877) "saxpy" __GI___ioctl (fd=3,␣
↪request=3222817548) at ../sysdeps/unix/sysv/linux/ioctl.c:36
* 7    AMDGPU Wave 1:2:1:1 (0,0,0)/0 "saxpy"       saxpy_kernel () at saxpy.cpp:8
```

3.1.11 Switch threads

```
(gdb) thread <id>
```

3.2 Printing kernel data

Commands to print the kernel data:

3.2.1 Print variable

```
(gdb) print foo
```

3.2.2 Print array

```
(gdb) print *foo[2]@8     -- i.e. (gdb) print *<address>@<count>
```

3.2.3 Print expressions

```
(gdb) print foo[4] >> 1
```

3.2.4 Print formats

```
// (gdb) p/<code> <value>
(gdb) p/x foo[1]
```

The values for <code> are:

- x - hexadecimal
- d - decimal
- u - unsigned decimal
- o - octal
- t - binary (two)
- a - address (hex + offset)
- c - character
- f - float
- s - string
- z - hexadecimal with leading zeros
- r - raw (skips pretty printing)

3.3 Modifying kernel data

The commands to modify the kernel data:

3.3.1 Using set command

Use the `set` command to modify kernel data directly.

```
(gdb) set var foo[1]=45
```

3.3.2 Using print command

The `print` command is an indirect way to modify the kernel data.

```
(gdb) print foo[3]=3
```

3.4 Changing kernel focus

Commands to change the kernel thread, lane, or frame:

3.4.1 Change thread

```
(gdb) thread 9
```

3.4.2 Change lane

```
(gdb) lane 5
```

3.4.3 Change frame

```
(gdb) frame <index>
(gdb) up <count>
(gdb) down <count>
```

3.5 Controlling kernel execution

Commands to control kernel execution:

3.5.1 Set breakpoints

```
(gdb) break saxpy.cpp:47
(gdb) break func_foo
(gdb) break *0x01234567
```

3.5.2 Set temporary breakpoints

```
(gdb) tbreak saxpy.cpp:47
(gdb) tbreak func_foo
(gdb) tbreak +24
```

3.5.3 Set conditional breakpoints

```
(gdb) break func_foo if idx == 9
(gdb) break func_foo if $_agent == 2
(gdb) break func_foo if $_queue == 1
(gdb) break func_foo if $_dispatch == 6
(gdb) break func_foo if $_thread == 7
(gdb) break func_foo if $_lane == 15
(gdb) break func_foo if $_thread_workgroup_pos == 3
(gdb) break func_foo if $_lane_workgroup_pos == "[0,0,0]"
```

3.5.4 Set watchpoints

```
(gdb) watch foo[4]
```

3.5.5 Set catchpoints

```
(gdb) catch load            -- Catch loads of shared libraries (debug dynamic linking).
(gdb) catch unload          -- Catch unloads of shared libraries (track cleanup/
↪unloading).
(gdb) catch rethrow         -- Catch an exception, when rethrown (trace exception␣
↪propagation).
(gdb) catch signal SIGSEGV  -- Catch signals by their names and/or numbers (debug␣
↪crashes or signals).
(gdb) catch syscall open    -- Catch system calls by names, groups, or numbers (trace␣
↪system-level calls).
(gdb) catch throw           -- Catch an exception, when thrown (trace exception origins).
(gdb) catch vfork           -- Catch calls to vfork (monitor child process creation).
```

3.5.6 Set scheduler locking (waves)

```
(gdb) set scheduler-locking on
```

For more information, see Scheduler locking mode.

3.5.7 Set scheduler non-stop (waves)

```
set non-stop non
```

For more information, see Non-stop mode.

3.5.8 Set scheduler all-stop (waves)

```
set non-stop off
```

For more information, see All-stop mode.

3.5.9 Disable breakpoint, watchpoint, catchpoint

```
disable 4
```

3.5.10 Enable breakpoint, watchpoint, catchpoint

```
enable 4
```

3.5.11 Delete breakpoint, watchpoint, catchpoint

```
// delete <list>
delete 4
```

3.5.12 Step execution (source line)

```
(gdb) step
(gdb) next
```

3.5.13 Step execution (multiple source lines)

```
(gdb) step 3
(gdb) next 3
```

3.5.14 Step execution (stack frame)

```
(gdb) until
(gdb) until 0x0000ffffdeadbeef
(gdb) finish
```

3.5.15 Step execution (machine instruction)

```
(gdb) stepi
(gdb) nexti
```

3.5.16 Resume execution

```
(gdb) continue
```

Command sequence:

```
(gdb) break saxpy.cpp:47
command BREAKPOINT_NUMBER
continue
end
```

CHAPTER

FOUR

SETTING UP THIRD-PARTY TOOLS

This topic discusses how to configure third-party tools or plugins such as the GDB dashboard and Visual Studio (VS) Code GUI for debugging applications using ROCgdb.

Note: AMD is not responsible for providing any support for issues or bugs reported on these third-party tools. To report such issues, visit the GitHub or webpage for these third-party tools. AMD doesn't guarantee that these third-party tools will work seamlessly across ROCm releases.

4.1 Setting up GDB dashboard TUI

The GDB dashboard is a Text User Interface (TUI). It's a standalone `.gdbinit` file written using the Python API, that provides a modular interface for showing relevant information about the program being debugged.

4.1.1 Installation

To install the GDB dashboard, download the .gdbinit file and move it to your home directory.

4.1.2 Layout setup

During debugging, the default dashboard layout setup appears automatically every time the inferior program stops. The GDB dashboard's purpose is to reduce the number of GDB commands needed to inspect the current program's status, allowing you to focus on the control flow.

To display the default set of views, use this command:

```
(gdb) dashboard -layout
```

Sample output:

```
Dashboard    (default TTY)
assembly     (default TTY)
breakpoints  (default TTY)
expressions  (default TTY)
history      (default TTY)
memory       (default TTY)
registers    (default TTY)
source       (default TTY)
```

(continues on next page)

(continued from previous page)

```
stack        (default TTY)
threads      (default TTY)
variables    (default TTY)
```

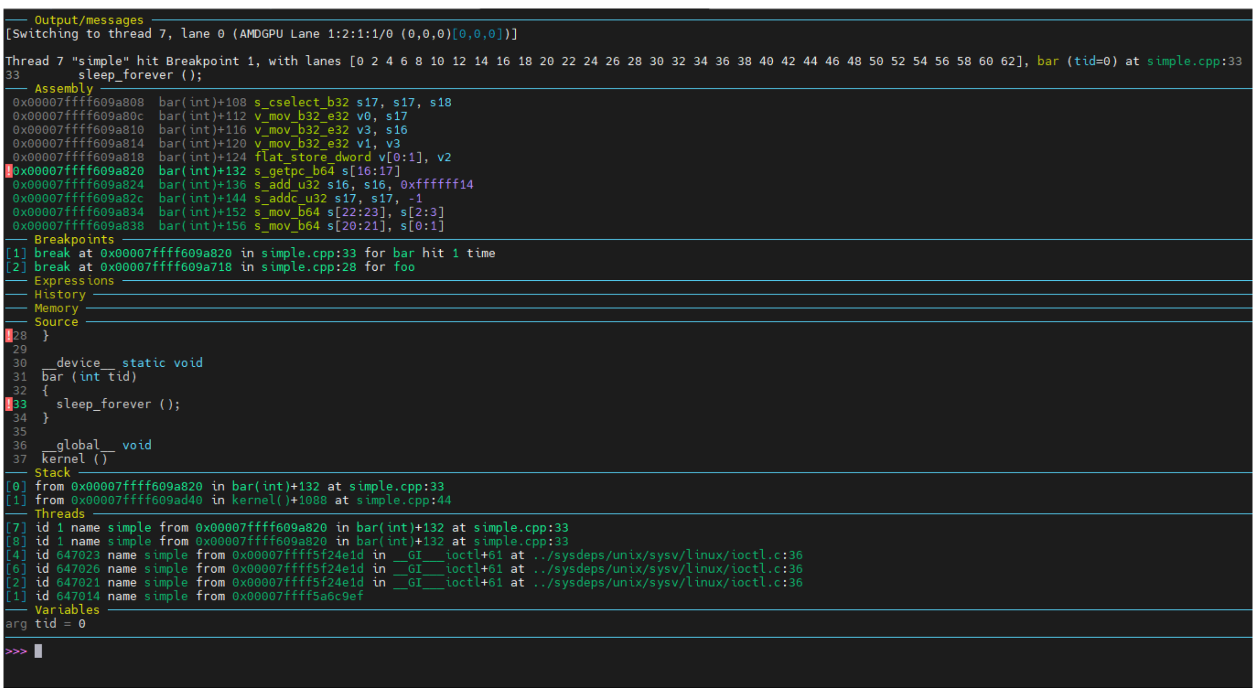
4.1.3 Customizing the dashboard

The GDB dashboard TUI is customizable. For example, you can customize the TUI to exclude less commonly used views from the default display during a debug session, such as **Expressions**, **History**, and **Memory** views.

To avoid a cluttered display with many AMD GPU registers displaying constantly on the dashboard, you can omit the **Register** view from the default dashboard using the following commands:

```
(gdb) dashboard registers
registers module disabled
(gdb) dashboard expressions
expressions module disabled
(gdb) dashboard history
history module disabled
(gdb) dashboard memory
memory module disabled
```

Here is how compact the customized dashboard will look:



Furthermore, the dashboard offers several stylable attributes that can be modified via the `-style` command, which applies to both the dashboard and individual modules. For example, the height of the **Source view** can be increased using the following command:

```
(gdb) dashboard source -style height 35
```

4.1.4 Dashboard command-line options

The following table lists the `dashboard` command-line options:

Table 4.1: dashboard cli options

Option	Description
`configuration`	Dumps or saves the dashboard configuration.
`enabled`	Enables or disables the dashboard.
`layout`	Sets or shows the dashboard layout.
`output`	Sets the output file or TTY for the whole dashboard or individual module.
`style`	Configures the stylable attributes.
`assembly`	Configures the assembly module. Using without arguments toggles its visibility.
`breakpoints`	Configures the breakpoints module. Using without arguments toggles its visibility.
`expressions`	Configures the expressions module. Using without arguments toggles its visibility.
`history`	Configures the history module. Using without arguments toggles its visibility.
`memory`	Configures the memory module. Using without arguments toggles its visibility.
`registers`	Configures the registers module. Using without arguments toggles its visibility.
`source`	Configures the source module. Using without arguments toggles its visibility.
`stack`	Configures the stack module. Using without arguments toggles its visibility.
`threads`	Configures the threads module. Using without arguments toggles its visibility.
`variables`	Configures the variables module. Using without arguments toggles its visibility.

To see the complete list of `dashboard` subcommands, you can also use `help`:

```
help dashboard
```

- For full documentation of a subcommand, use `help dashboard` followed by the subcommand name.
- To search for commands related to a "word", use `apropos <word>`.
- For full documentation of commands related to a "word", use `apropos -v <word>`.

You can also pass command name abbreviations as "word", if unambiguous.

For more information on GDB dashboard, see GDB dashboard wiki.

4.2 Setting up VS Code GUI

This section provides information on configuring Visual Studio (VS) Code GUI for debugging applications using ROCgdb.

4.2.1 Installing extensions

To use ROCgdb within the VS Code, you need to install some VS Code extensions. Only two extensions are required from external vendors while the rest are provided by Microsoft. These extensions are grouped into three categories:

- Must-have extensions. These are required for HIP debugging.
- Extra extensions for Python tracing.
- Optional extensions.

Must-have extensions

- C/C++ for VS Code by Microsoft
- C/C++ for Extension Pack by Microsoft
- C/C++ Themes by Microsoft
- Remote SSH by Microsoft
- Remote Explorer by Microsoft
- Remote Development by Microsoft
  - This installs Dev Containers and Remote Tunnels by Microsoft, which is necessary for tracing under Docker.
- Docker by Microsoft

Extra extensions for Python tracing

- Pylance by Microsoft
- Python by Microsoft
- Python Debugger by Microsoft
- Python C++ Debugger by BeniBenj

Note: VS Code requires you to install the extensions on the remote system as well.

Optional extensions

- Jupyter by Microsoft
- GitHub Pull Request by Microsoft

4.2.2 Configuring the Remote Debugger settings

After installing the VS Code extensions, you need to configure the Remote Debugger settings. The settings help VS Code to connect (Attach) to the machine hosting the HIP program to be debugged and execute the program under ROCgdb.

Follow these steps to configure the Remote Debugger settings:

1. Select **Remote Explorer** and add the new remote:
   - Add the ssh command line `ssh <user_name>@<remote_server_url>`.
2. Connect to the remote system.
3. Open the repo folder on the remote system. You can use a previously cloned CLR repo from the public GitHub.
4. Click on **Run and Debug** button on the left panel.
5. Click on **Create a launch.json** file.
6. Select **GDB** in the drop out menu and add these two configurations: **(gdb) Attach** and **(gdb) Launch**.
   - Attach doesn't require any extra setup.
   - Launch requires the environment variable `LD_LIBRARY_PATH` to point to the debug build of runtime.

- If required, set the debugger path to `rocgdb` installation path. For example, `miDebuggerPath: /opt/rocm-7.2.0/bin/rocgdb`.

4.2.3 Configuration file: launch.json

The launch.json configuration file contains information required by VS Code to Launch or Attach to a program for debugging. This information includes path information for the debugger and the program including the arguments and environment variables.

Here is a sample launch.json file:

```
{
   "version": "0.2.0",
   "configurations": [
      {
         "name": "(gdb) Attach",
         "type": "cppdbg",
         "request": "attach",
         "processId": "${command:pickProcess}",
         "program" : "/usr/bin/python3",
         "miDebuggerPath": "/opt/rocm-6.4.0/bin/rocgdb",
         "MIMode": "gdb",
         "setupCommands": [
         {
            "description": "Enable pretty-printing for gdb",
            "text": "-enable-pretty-printing",
            "ignoreFailures": true
         },
         {
            "description": "Set Disassembly Flavor to Intel",
            "text": "-gdb-set disassembly-flavor intel",
            "ignoreFailures": true
         }
         ]
      },
      {
         "name": "(gdb) Launch",
         "type": "cppdbg",
         "request": "launch",
         "program": "/home/test_dir/graph/graph",
         "args": [
         "Unit_hipMemcpy_MultiThread-AllAPIs"
         ],
         "stopAtEntry": true,
         "cwd": "/home/test_dir/graph/",
         "environment": [
         {
            "name": "LD_LIBRARY_PATH",
            "value": "/home/test_dir/udp/clr/build/install/lib/:/opt/rocm/lib"
         },
         {
            "name": "DEBUG_HIP_MEM_POOL_VMHEAP",
            "value": "1"
```

(continues on next page)

(continued from previous page)

```
            }
            ],
            "externalConsole": false,
            "MIMode": "gdb",
            "setupCommands": [
            {
               "description": "Enable pretty-printing for gdb",
               "text": "-enable-pretty-printing",
               "ignoreFailures": true
            },
            {
               "description": "Set Disassembly Flavor to Intel",
               "text": "-gdb-set disassembly-flavor intel",
               "ignoreFailures": true
            }
            ]
        }
    ]
}
```

4.2.4 Launching the debugger

After the debugger settings are configured, the **Run and Debug** tab will show these two options:

- **(gdb) Attach option:** This option is used to connect the debugger to a running process.
- **(gdb) Launch option:** This option is used to start a process under debugger control.

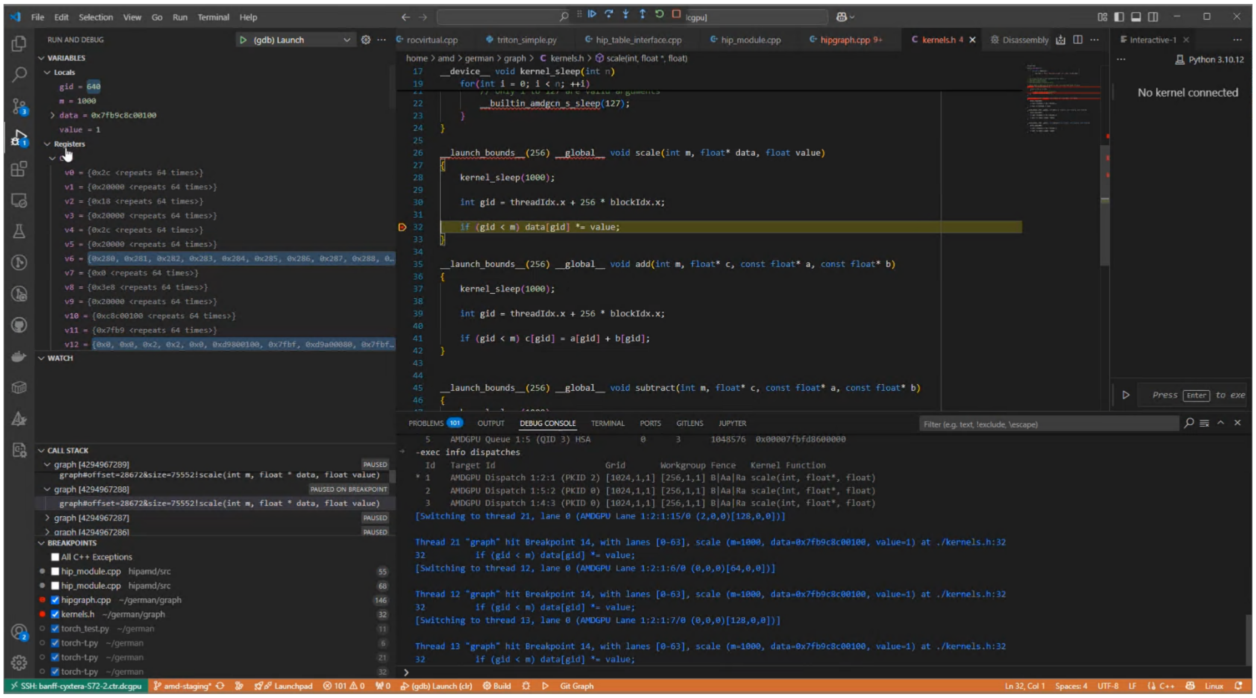
To start remote debugging, follow these steps:

1. Click on the **Launch** option to start the application under debugger control:
   - `stopAtEntry: true` stops the application on `main()`.
2. Navigate in the repo and set breakpoints in the application or runtime source code.
3. VS Code enables pretty printers by default.
   - STL classes are easily modifiable like regular data sets.
   - ROCgdb might require `~/.gdbinit` for pretty printers:

   ```
   python
   import sys
   sys.path.insert(0, '/usr/share/gcc/python')
   from libstdcxx.v6.printers import register_libstdcxx_printers
   register_libstdcxx_printers (None)
   end
   ```
4. ROCgdb also facilitates device kernel tracing. Breakpoints, variables, and registers work automatically.

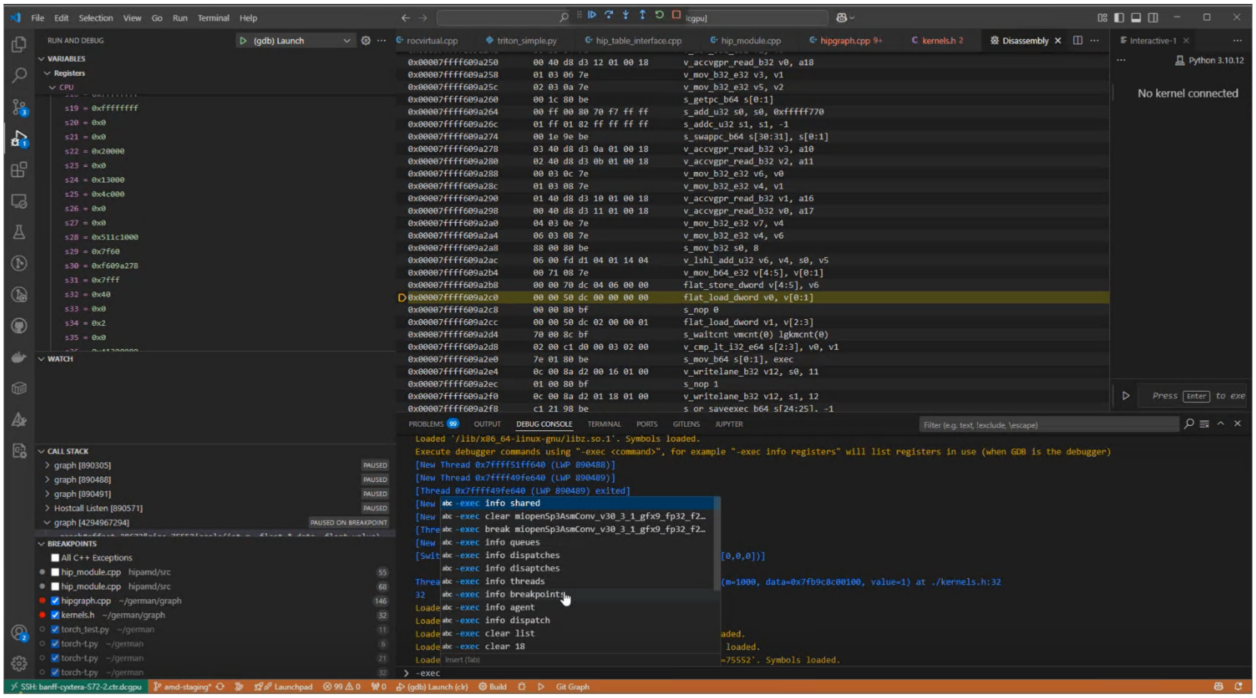
4.2.5 Debugger displays

During an active debug session, several tabs are available for displaying the running program and kernel states. These include tabs to display kernel variable, call stack frame, GPU registers, and source code breakpoint locations.



4.2.6 Debug console

During a debug session when the inferior is stopped, you can enter ROCgdb commands in the Debug console. All such commands must be entered with a `-exec` prefix. For example, all GPU threads can be displayed using `-exec info threads`.



CHAPTER

FIVE

DEBUGGING PYTHON KERNEL CODE

This topic provides information on debugging Python (Pytorch) kernel code that offloads HIP compute kernels to AMD GPU.

5.1 Installing extensions

To debug the Python kernel code, you must install the **Extra Additions For Python tracing** extensions on the remote system. These extensions help you to set a breakpoint at the Python layer, C/C++ layer, and/or HIP kernel layer simultaneously.

5.2 Getting started

To start debugging Python code, follow these steps:

1. Install Python extensions on the remote system or Docker.
2. Add Python Debugger configuration.
3. Specify the Python version using interpreter (Ctrl+Shift+P).
4. Select **Python C++ Debugger Custom** option in the configurations. It launches the Python Debugger and attaches gdb to the Python process.
5. Verify the (gdb) Attach configuration. You might need to run `echo 0|sudo tee /proc/sys/kernel/yama/ptrace_scope` to allow attach.
6. You can now see the Python Debugger and gdb threads in the **CALL STACK** window. The **Breakpoint** window works for both and is easy to navigate.

5.2.1 Key considerations

- The Python extension doesn't support gdb for Python and C++ tracing.
- Breaks in the Python script don't activate gdb automatically.
- Ensure that Python picks the correct runtime libraries. To see the location of the loaded libraries, use `-exec info shared`.
- There are multiple terminal windows. Switch as required.

5.3 Configuration file: launch.json

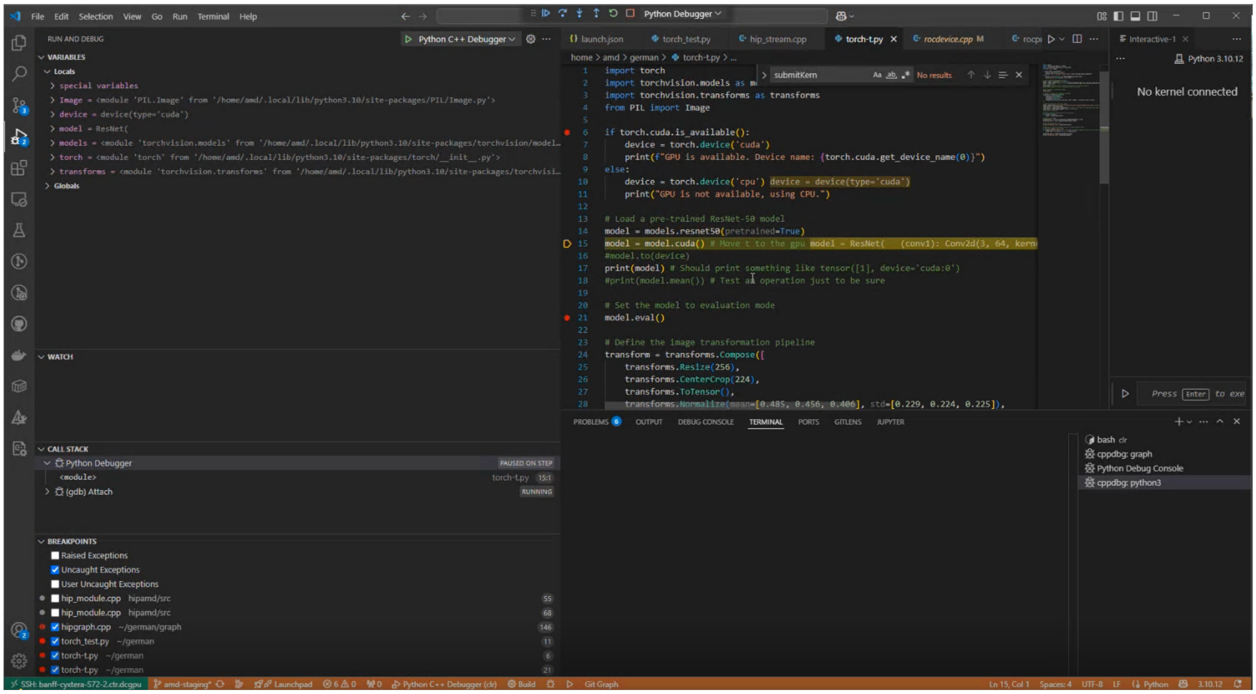
The launch.json configuration file contains the information required by VS Code to debug the Python program. This information includes the path information and environment variables required for the Python program.

See the configuration items in the following sample launch.json configuration file:

```
{
    "name": "Python C++ Debugger",
    "type": "pythoncpp",
    "request": "launch",
    "pythonLaunchName": "Python Debugger",
    "cppAttachName": "(gdb) Attach"
},
{
    "name": "Python Debugger",
    "type": "debugpy",
    "request": "launch",
    "program": "/home/test_dir/test.py",
    "console": "integratedTerminal",
    "cwd": "/home/test_dir/",
    "env": {
        "PYTHONPATH":"${PYTHONPATH}:/opt/rocm/bin:/opt/rocm/lib:/home/myenv-py311/lib/
↪python3.11/site-packages",
        "LD_LIBRARY_PATH":"/home/test_dir/udp/clr/build/install/lib/:/opt/rocm/lib",
        "AMD_LOG_LEVEL":"4"
}
},
```

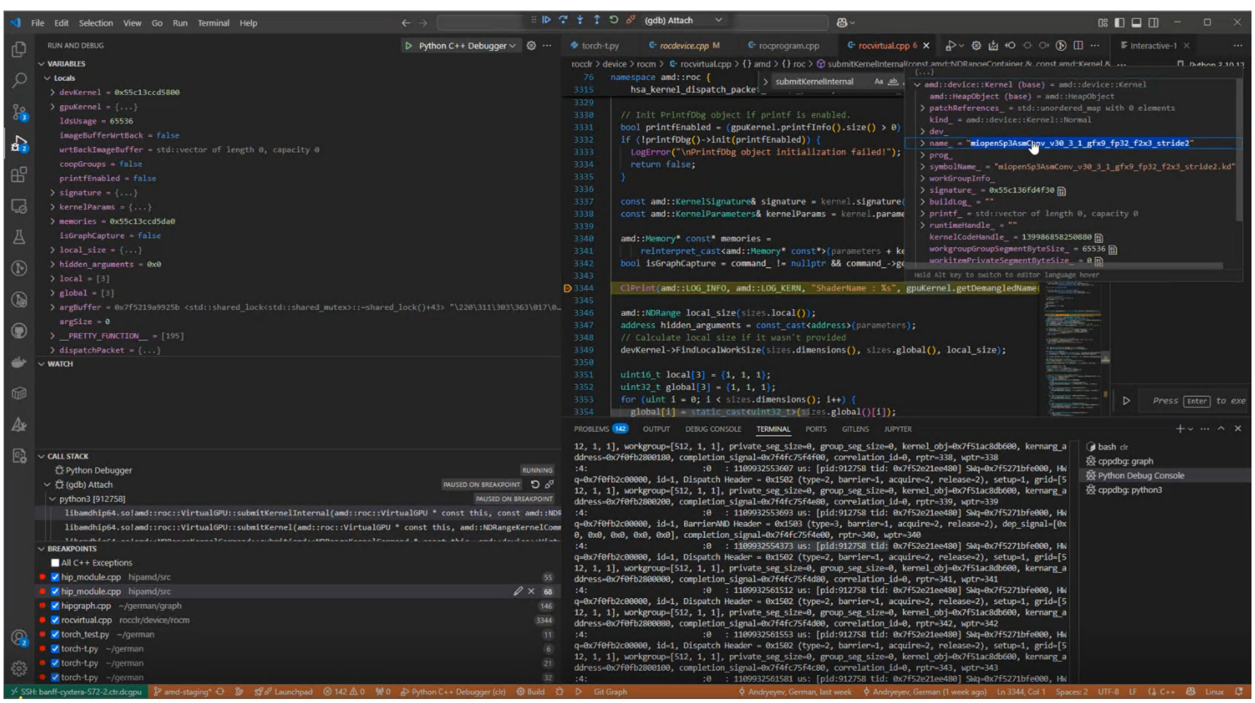
5.4 Python and C++ breakpoints

Running the launch configuration for the Python C++ Debugger starts program execution in the Python code and stops it at any preset breakpoint. As shown in the following image, there will be two entries in the **CALL STACK** window: one for **Python Debugger** and another for **(gdb) Attach**:



Whenever Python code execution pauses at a breakpoint, additional breakpoints can be set in the C/C++ code layer so that when Python code execution resumes and calls down to the C/C++ layer, program execution will stop at that C/C++ layer breakpoint.

Under the **CALL STACK** window in the following image, see the call stack hierarchy under the (gdb) Attach inferior. If the C/C++ layer breakpoint is set at the `amd::roc::VirtualGPU::submitKernel()` function, the name field for the `amd::device::kernel` object shows the name of the HIP kernel about to be submitted.



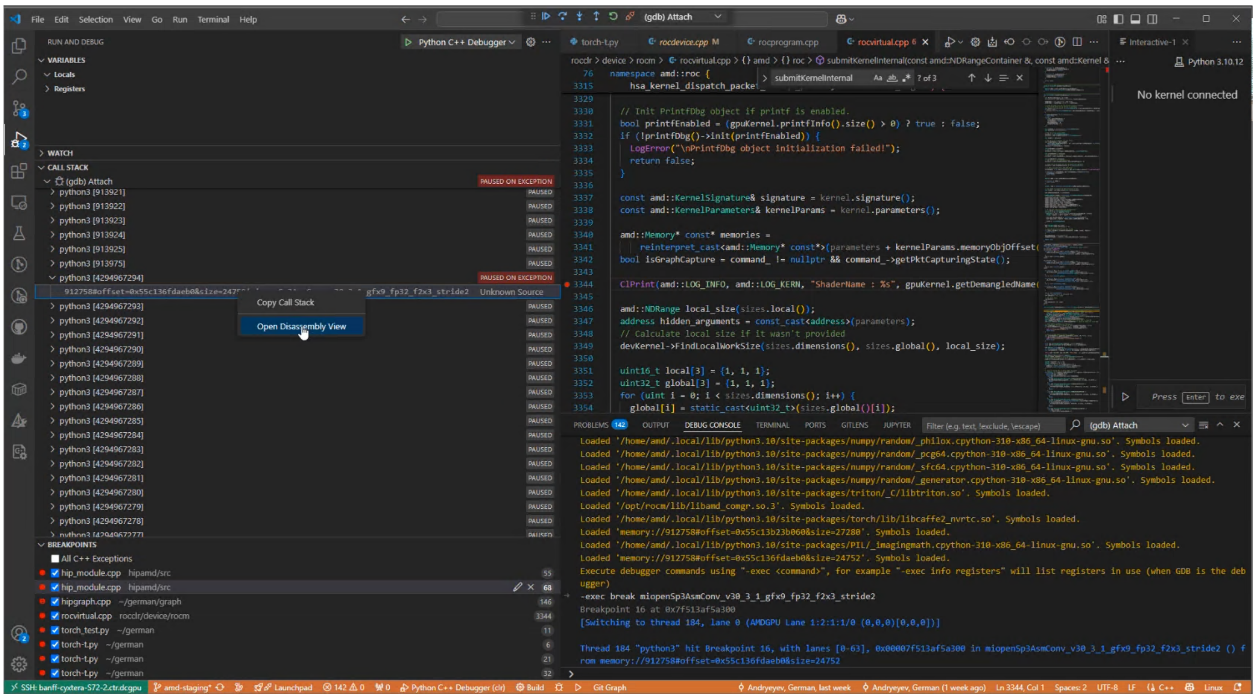
5.5 C++ and HIP kernel breakpoints

Once a breakpoint in the C/C++ layer is hit, a HIP kernel breakpoint can be set in the DEBUG CONSOLE using `-exec break <kernel_name>`. Once execution continues and the specified kernel is executed, Python will be **PAUSED ON EXCEPTION**.

To inspect the HIP kernel state:

1. Navigate to the **CALL STACK** window.
2. Search for the python3 [ID] with the **PAUSED ON EXCEPTION** status.
3. Selecting this python3 [ID] will change the debugger focus.
4. To view its disassembly, right click on this python3 [ID] and select **Open Disassembly View**.

The following image demonstrates the HIP kernel breakpoints:



CHAPTER

SIX

LICENSE

```
                GNU GENERAL PUBLIC LICENSE
                   Version 3, 29 June 2007
```



```
                        Preamble
```

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

```
                    TERMS AND CONDITIONS
```

0. Definitions.

"This License" refers to version 3 of the GNU General Public License.

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

```
a) The work must carry prominent notices stating that you modified
it, and giving a relevant date.

b) The work must carry prominent notices stating that it is
released under this License and any conditions added under section
7.  This requirement modifies the requirement in section 4 to
"keep intact all notices".

c) You must license the entire work, as a whole, under this
License to anyone who comes into possession of a copy.  This
License will therefore apply, along with any applicable section 7
additional terms, to the whole of the work, and all its parts,
regardless of how they are packaged.  This License gives no
permission to license the work in any other way, but it does not
```

(continues on next page)

(continued from previous page)

```
invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display
Appropriate Legal Notices; however, if the Program has interactive
interfaces that do not display Appropriate Legal Notices, your
work need not make them do so.
```

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

```
a) Convey the object code in, or embodied in, a physical product
(including a physical distribution medium), accompanied by the
Corresponding Source fixed on a durable physical medium
customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product
(including a physical distribution medium), accompanied by a
written offer, valid for at least three years and valid for as
long as you offer spare parts or customer support for that product
model, to give anyone who possesses the object code either (1) a
copy of the Corresponding Source for all the software in the
product that is covered by this License, on a durable physical
medium customarily used for software interchange, for a price no
more than your reasonable cost of physically performing this
conveying of source, or (2) access to copy the
Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the
written offer to provide the Corresponding Source.  This
alternative is allowed only occasionally and noncommercially, and
only if you received the object code with such an offer, in accord
with subsection 6b.

d) Convey the object code by offering access from a designated
place (gratis or for a charge), and offer equivalent access to the
Corresponding Source in the same way through the same place at no
further charge.  You need not require recipients to copy the
Corresponding Source along with the object code.  If the place to
copy the object code is a network server, the Corresponding Source
may be on a different server (operated by you or a third party)
that supports equivalent copying facilities, provided you maintain
clear directions next to the object code saying where to find the
Corresponding Source.  Regardless of what server hosts the
Corresponding Source, you remain obligated to ensure that it is
available for as long as needed to satisfy these requirements.
```

(continues on next page)

(continued from previous page)

```
e) Convey the object code using peer-to-peer transmission, provided
you inform other peers where the object code and Corresponding
Source of the work are being offered to the general public at no
charge under subsection 6d.
```

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

```
a) Disclaiming warranty or limiting liability differently from the
terms of sections 15 and 16 of this License; or
```

(continues on next page)

(continued from previous page)

```
b) Requiring preservation of specified reasonable legal notices or
author attributions in that material or in the Appropriate Legal
Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or
requiring that modified versions of such material be marked in
reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or
authors of the material; or

e) Declining to grant rights under trademark law for use of some
trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that
material by anyone who conveys the material (or modified versions of
it) with contractual assumptions of liability to the recipient, for
any liability that these contractual assumptions directly impose on
those licensors and authors.
```

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

```
                 END OF TERMS AND CONDITIONS

        How to Apply These Terms to Your New Programs
```

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <http://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w' and` show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see http://www.gnu.org/licenses/.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read http://www.gnu.org/philosophy/why-not-lgpl.html.