Gdbgui

Contents

Gdbgui

Stable: 09.10.2019 - 15:48 / Revision: 04.09.2019 - 11:17

A quality version of this page, accepted on *9 October 2019*, was based off this revision.

Contents

1 Article purpose

This article provides the basic information needed to start using the **gdbgui**[1] host PC tool.

2 Introduction

The following table provides a brief description of the tool, as well as its availability depending on the software packages:

: this tool is either present (ready to use or to be activated), or can be integrated and activated on the software package.

: this tool is not present and cannot be integrated, or it is present but cannot be activated on the software package.

Tool			STM32MPU Embedded Software distribution			STM32MPU Embedded Software distribution for Android™		
Name	Category	Purpose	Starter Package	Developer Package	Distribution Package	Starter Package	Developer Package	Distribution Package
		gdbgui[1] runs GDB in the background to build			*			

Tool			distribution			distribution for Android™		
Name	Category	Purpose	Starter Package	Developer Package	Distribution Package	Starter Package	Developer Package	Distribution Package
gdbgui	Debugging tools	an easy-to-use graphical user interface. It is a browser-based debugger that operates through a web browser page. Since it is based on GDB, it can be used through a JTAG or an ethernet link with gdbserver.				⊗	⊗	⊗

*: The Developer Package is required to run a GDB debug session that includes all dependencies.

3 Installing the trace and debug tool on your host PC (Ubuntu Linux distribution)

To avoid variable conflicts, the following steps must not be executed from a console where the SDK environment is set.

If it is not present on your Ubuntu Linux® machine, install gdbgui via the Python package installer (pip):

```
PC $> pip install --trusted-host pypi.python.org --cert /etc/ssl/certs/ --proxy
$https_proxy --upgrade gdbgui
```

Note1: If pip is not already installed, install it as follows:

```
PC $> sudo apt-get install python-pip
```

Note2: Depending on your environment, the --proxy option is not mandatory; if required, please check that $https_proxy is well defined, or replace it by your environment variable in the command line.

The gdbgui binary is then installed under *$HOME/.local/bin/gdbgui*. You can check it by using ths following command:

```
PC $> which gdbgui
```

4 Getting started with gdbgui

4.1 Running gdbgui using gdbserver

This environment is mainly used to debug applications. The Developer Package context (SDK) is used for setup.

- Start gdbserver on your target board for the userland program you want to debug:

```
Board $> gdbserver host:1234 hello_world_example
```

- Go to the source directory path:

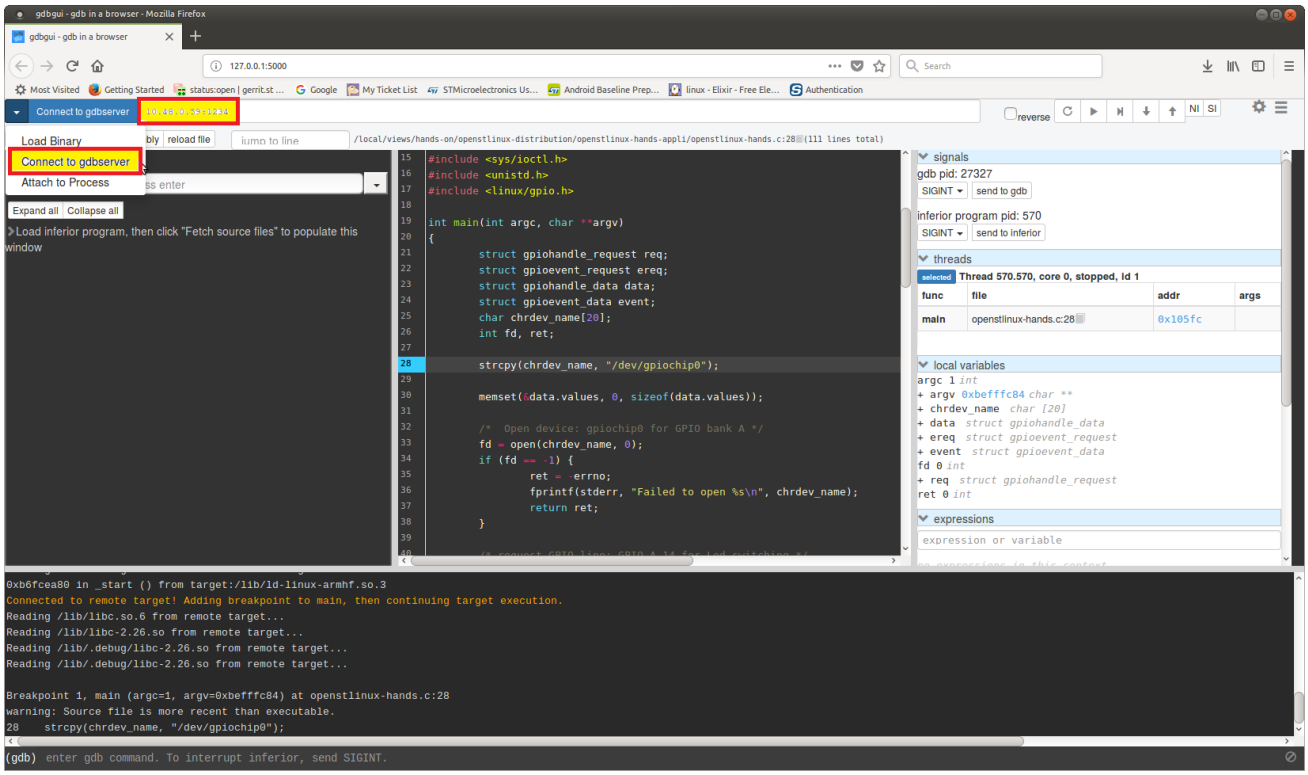
```
PC $> cd <source_file_folder_path>
```

- Set the SDK environment and start **gdbgui**:

```
PC $> source <Your_SDK_path>/environment-setup-cortexa7hf-neon-vfpv4-
openstlinux_weston-linux-gnueabi
PC $> gdbgui -g $OECORE_NATIVE_SYSROOT/usr/bin/arm-openstlinux_weston-linux-gnueabi
/arm-openstlinux_weston-linux-gnueabi-gdb
```

Note: In case of issue when starting gdbgui, please refer to the Error cases chapter.

- Then:
  - Select the "Connect to gdbserver" option from the top left menu.
  - Fill in the ip address of the remote target including the port.



4.2 Running gdbgui using ST-LINK

In that case, GDB with openOCD environment is used in a Developer Package context (SDK). Please refer to Running OpenOCD and GDB.

Only GDB console command will be changed:

- OpenOCD console

No change for OpenOCD. Please refer to the command described in the above link.

- GDB console

gdbgui must be started with an argument providing the path to the GDB version available on your target board, as well as the config setup file:

```
PC $> source <Your_SDK_path>/environment-setup-cortexa7hf-neon-vfpv4-
openstlinux_weston-linux-gnueabi
PC $> gdbgui -g $OECORE_NATIVE_SYSROOT/usr/bin/arm-openstlinux_weston-linux-gnueabi
/arm-openstlinux_weston-linux-gnueabi-gdb --gdb-args="-x <Your_full_path_to>/Setup.gdb"
```

Note: Please refer to GDB page for information on Setup.gdb file and how to configure it: Attach on running target or Attach on boot.

5 To go further

5.1 Execution and debug

A button to program the execution (continue, step, next), and a panel showing all debug information (signals, thread, variables, memory) can be found on the top right of the gdbgui web page.

Please refer to the **gdbgui**[1] web page for details.

5.2 Error cases

- *socket.error: [Errno 98] Address already in use: ('127.0.0.1', 5000)*

  This error occurs when an instance of gdbgui is already running in background.
  In that case, the connection to the gdbserver fails in gdbgui, and the message "remote 'g' packet reply is too long" is displayed.
  Just kill the gdbgui instance running in background to solve the issue.

6 References

- 1.01.11.2 https://gdbgui.com/

debug and test protocol, named from the Joint Test Action Group that developed it

GNU dedugger, a portable debugger that runs on many Unix-like systems

Software development kit (A programming package that enables a programmer to develop applications for a specific platform.)

Gdbgui

Contents

1 Article purpose

This article provides the basic information needed to start using the **gdbgui**[1] host PC tool.

2 Introduction

The following table provides a brief description of the tool, as well as its availability depending on the software packages:

: this tool is either present (ready to use or to be activated), or can be integrated and activated on the software package.

: this tool is not present and cannot be integrated, or it is present but cannot be activated on the software package.

Tool			STM32MPU Embedded Software distribution			STM32MPU Embedded Software distribution for Android™		
Name	Category	Purpose	Starter Package	Developer Package	Distribution Package	Starter Package	Developer Package	Distribution Package
gdbgui	Debugging tools	**gdbgui**[1] runs GDB in the background to build an easy-to-use graphical user interface. It is a browser-based debugger that operates throug			*			

Tool			STM32MPU Embedded Software distribution			STM32MPU Embedded Software distribution for Android™		
Name	Category	Purpose	Starter Package	Developer Package	Distribution Package	Starter Package	Developer Package	Distribution Package
		h a web browser page. Since it is based on GDB, it can be used through a JTAG or an ethernet link with gdbserver.	*: The Developer Package is required to run a GDB debug session that includes all dependencies.					

3 Installing the trace and debug tool on your host PC (Ubuntu Linux distribution)

To avoid variable conflicts, the following steps must not be executed from a console where the SDK environment is set.

If it is not present on your Ubuntu Linux® machine, install gdbgui via the Python package installer (pip):

```
PC $> pip install --trusted-host pypi.python.org --cert /etc/ssl/certs/ --proxy
$https_proxy --upgrade gdbgui
```

Note1: If pip is not already installed, install it as follows:

```
PC $> sudo apt-get install python-pip
```

Note2: Depending on your environment, the --proxy option is not mandatory; if required, please check that $https_proxy is well defined, or replace it by your environment variable in the command line.

The gdbgui binary is then installed under *$HOME/.local/bin/gdbgui*. You can check it by using ths following command:

```
PC $> which gdbgui
```

4 Getting started with gdbgui

4.1 Running gdbgui using gdbserver

This environment is mainly used to debug applications. The Developer Package context (SDK) is used for setup.

- Start gdbserver on your target board for the userland program you want to debug:

```
Board $> gdbserver host:1234 hello_world_example
```

- Go to the source directory path:

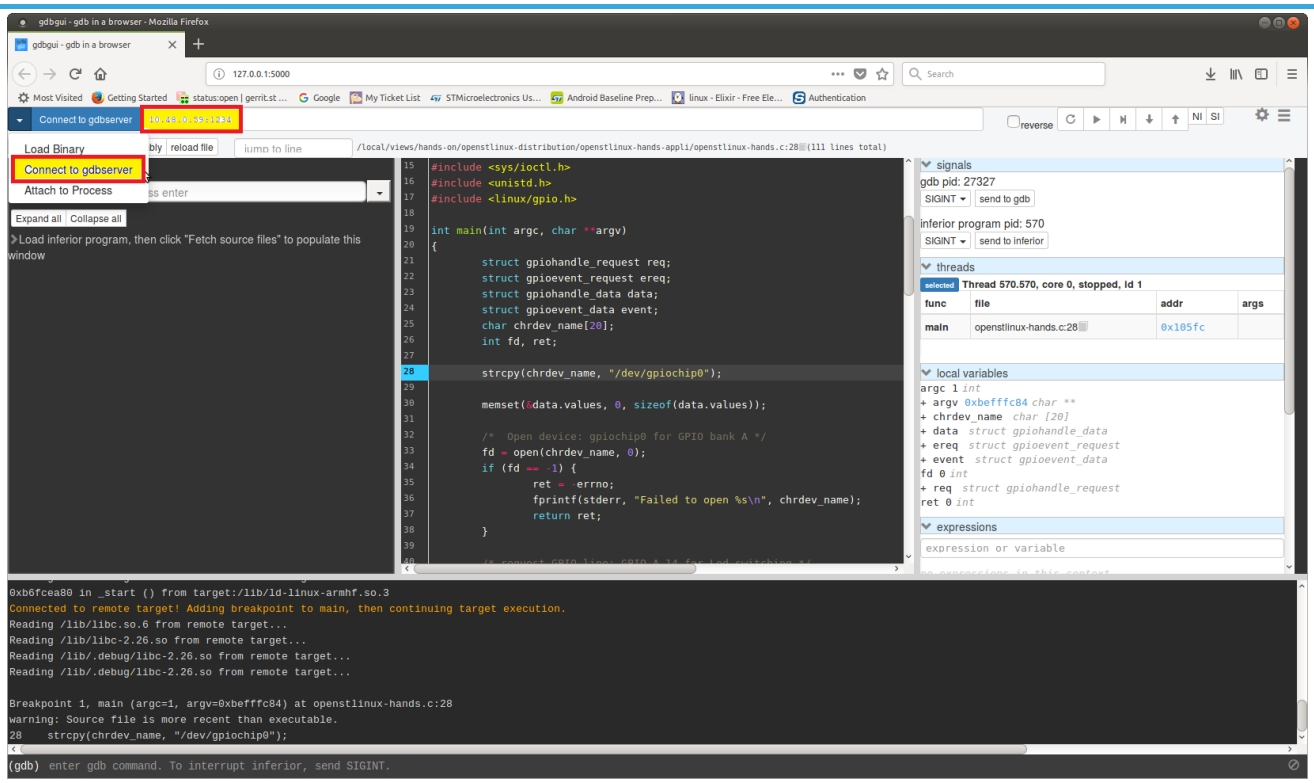
```
PC $> cd <source_file_folder_path>
```

- Set the SDK environment and start **gdbgui**:

```
PC $> source <Your_SDK_path>/environment-setup-cortexa7hf-neon-vfpv4-
openstlinux_weston-linux-gnueabi
PC $> gdbgui -g $OECORE_NATIVE_SYSROOT/usr/bin/arm-openstlinux_weston-linux-gnueabi
/arm-openstlinux_weston-linux-gnueabi-gdb
```

Note: In case of issue when starting gdbgui, please refer to the Error cases chapter.

- Then:
  - Select the "Connect to gdbserver" option from the top left menu.
  - Fill in the ip address of the remote target including the port.



4.2 Running gdbgui using ST-LINK

In that case, GDB with openOCD environment is used in a Developer Package context (SDK). Please refer to Running OpenOCD and GDB.

Only GDB console command will be changed:

- OpenOCD console

No change for OpenOCD. Please refer to the command described in the above link.

- GDB console

gdbgui must be started with an argument providing the path to the GDB version available on your target board, as well as the config setup file:

```
PC $> source <Your_SDK_path>/environment-setup-cortexa7hf-neon-vfpv4-
openstlinux_weston-linux-gnueabi
PC $> gdbgui -g $OECORE_NATIVE_SYSROOT/usr/bin/arm-openstlinux_weston-linux-gnueabi
/arm-openstlinux_weston-linux-gnueabi-gdb --gdb-args="-x <Your_full_path_to>/Setup.gdb"
```

Note: Please refer to GDB page for information on Setup.gdb file and how to configure it: Attach on running target or Attach on boot.

5 To go further

5.1 Execution and debug

A button to program the execution (continue, step, next), and a panel showing all debug information (signals, thread, variables, memory) can be found on the top right of the gdbgui web page.

Please refer to the **gdbgui**[1] web page for details.

5.2 Error cases

- *socket.error: [Errno 98] Address already in use: ('127.0.0.1', 5000)*

  This error occurs when an instance of gdbgui is already running in background.
  In that case, the connection to the gdbserver fails in gdbgui, and the message "remote 'g' packet reply is too long" is displayed.
  Just kill the gdbgui instance running in background to solve the issue.

6 References

- 1.01.11.2 https://gdbgui.com/

debug and test protocol, named from the Joint Test Action Group that developed it

GNU dedugger, a portable debugger that runs on many Unix-like systems

Software development kit (A programming package that enables a programmer to develop applications for a specific platform.)

Subcategories

This category has the following 3 subcategories, out of 3 total.

A

- Android debugging tools (1 P)

H

- HW probes (1 P)

L

- Linux debugging tools (3 P)

Pages in category "Debugging tools"

The following 7 pages are in this category, out of 7 total.

G

- GDB
- GDB commands
- Gdbgui

I

- IDE

O

- OP-TEE - How to debug

T

- TF-A - How to debug

U

- U-Boot - How to debug

Gdbgui

Stable: 15.10.2019 - 09:33 / Revision: 15.10.2019 - 09:31

Contents

1 Article purpose

This article provides the basic information needed to start using the **gdbgui**[1] host PC tool.

2 Introduction

The following table provides a brief description of the tool, as well as its availability depending on the software packages:

✓: this tool is either present (ready to use or to be activated), or can be integrated and activated on the software package.

✗: this tool is not present and cannot be integrated, or it is present but cannot be activated on the software package.

Tool			STM32MPU Embedded Software distribution			STM32MPU Embedded Software distribution for Android™		
Name	Category	Purpose	Starter Package	Developer Package	Distribution Package	Starter Package	Developer Package	Distribution Package
gdbgui	Debug	**gdbgui**[1] runs GDB in the background to build an easy-to-use graphical user	✗	✓	✗*	✗	✗	✗

Tool			STM32MPU Embedded Software distribution			STM32MPU Embedded Software distribution for Android™		
Name	Category	Purpose	Starter Package	Developer Package	Distribution Package	Starter Package	Developer Package	Distribution Package
	ging tools	interface. It is a browser-based debugger that operates through a web browser page. Since it is based on GDB, it can be used through a JTAG or an ethernet link with gdbserver.	*: The Developer Package is required to run a GDB debug session that includes all dependencies.					

3 Installing the trace and debug tool on your host PC (Ubuntu Linux distribution)

To avoid variable conflicts, the following steps must not be executed from a console where the SDK environment is set.

If it is not present on your Ubuntu Linux® machine, install gdbgui via the Python package installer (pip):

```
PC $> pip install --trusted-host pypi.python.org --cert /etc/ssl/certs/ --proxy
$https_proxy --upgrade gdbgui
```

Note1: If pip is not already installed, install it as follows:

```
PC $> sudo apt-get install python-pip
```

Note2: Depending on your environment, the --proxy option is not mandatory; if required, please check that $https_proxy is well defined, or replace it by your environment variable in the command line.

The gdbgui binary is then installed under *$HOME/.local/bin/gdbgui*. You can check it by using ths following command:

```
PC $> which gdbgui
```

4 Getting started with gdbgui

4.1 Running gdbgui using gdbserver

This environment is mainly used to debug applications. The Developer Package context (SDK) is used for setup.

- Start gdbserver on your target board for the userland program you want to debug:

```
Board $> gdbserver host:1234 hello_world_example
```

- Go to the source directory path:

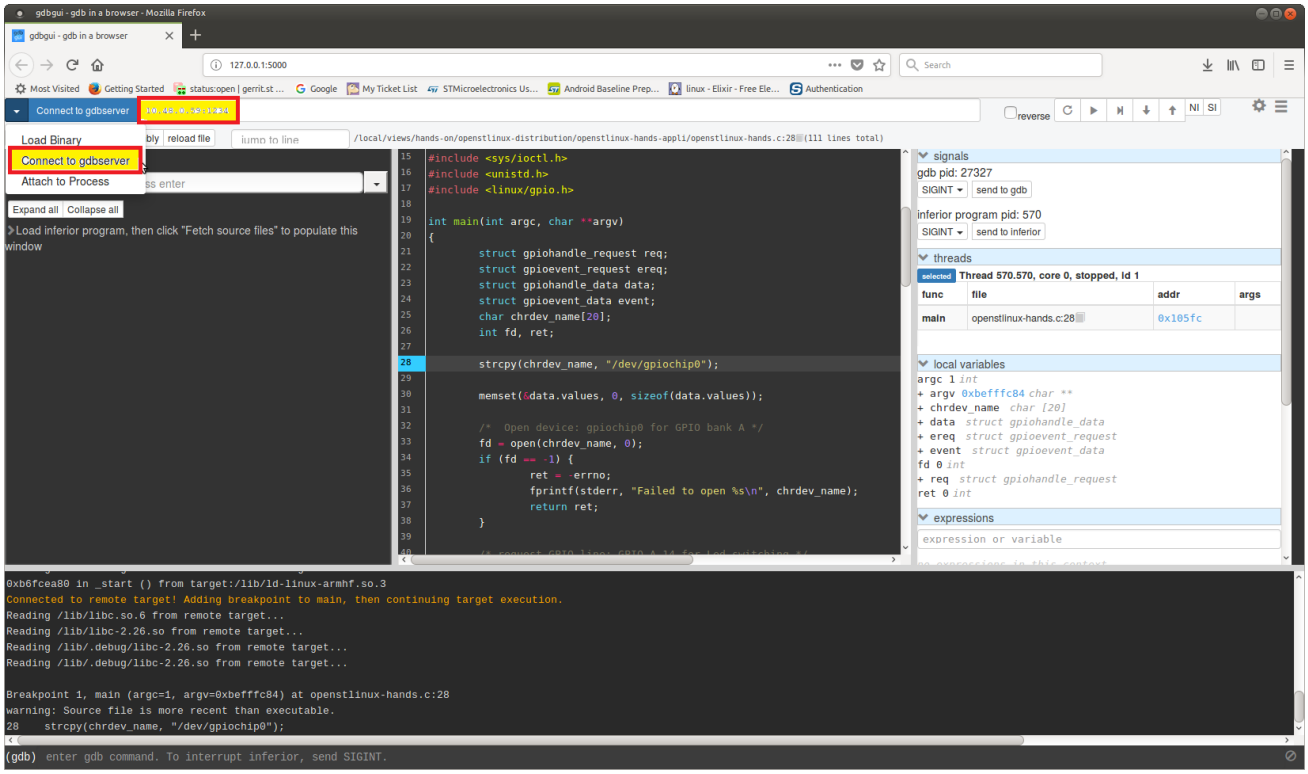
```
PC $> cd <source_file_folder_path>
```

- Set the SDK environment and start **gdbgui**:

```
PC $> source <Your_SDK_path>/environment-setup-cortexa7hf-neon-vfpv4-
openstlinux_weston-linux-gnueabi
PC $> gdbgui -g $OECORE_NATIVE_SYSROOT/usr/bin/arm-openstlinux_weston-linux-gnueabi
/arm-openstlinux_weston-linux-gnueabi-gdb
```

Note: In case of issue when starting gdbgui, please refer to the Error cases chapter.

- Then:
  - Select the "Connect to gdbserver" option from the top left menu.
  - Fill in the ip address of the remote target including the port.



4.2 Running gdbgui using ST-LINK

In that case, GDB with openOCD environment is used in a Developer Package context (SDK). Please refer to Running OpenOCD and GDB.

Only GDB console command will be changed:

- OpenOCD console

No change for OpenOCD. Please refer to the command described in the above link.

- GDB console

gdbgui must be started with an argument providing the path to the GDB version available on your target board, as well as the config setup file:

```
PC $> source <Your_SDK_path>/environment-setup-cortexa7hf-neon-vfpv4-
openstlinux_weston-linux-gnueabi
PC $> gdbgui -g $OECORE_NATIVE_SYSROOT/usr/bin/arm-openstlinux_weston-linux-gnueabi
/arm-openstlinux_weston-linux-gnueabi-gdb --gdb-args="-x <Your_full_path_to>/Setup.gdb"
```

Note: Please refer to GDB page for information on Setup.gdb file and how to configure it: Attach on running target or Attach on boot.

5 To go further

5.1 Execution and debug

A button to program the execution (continue, step, next), and a panel showing all debug information (signals, thread, variables, memory) can be found on the top right of the gdbgui web page.

Please refer to the **gdbgui**[1] web page for details.

5.2 Error cases

- *socket.error: [Errno 98] Address already in use: ('127.0.0.1', 5000)*

This error occurs when an instance of gdbgui is already running in background.
In that case, the connection to the gdbserver fails in gdbgui, and the message "remote 'g' packet reply is too long" is displayed.
Just kill the gdbgui instance running in background to solve the issue.

6 References

- 1.01.11.2 https://gdbgui.com/

debug and test protocol, named from the Joint Test Action Group that developed it

GNU dedugger, a portable debugger that runs on many Unix-like systems

Software development kit (A programming package that enables a programmer to develop applications for a specific platform.)