
Debugging the PineCone

Tobias Tefke¹, Prof. Ralf C. Staudemeyer¹

¹ Schmalkalden University of Applied Sciences, Schmalkalden, Germany

October 9, 2025

Additional requirements

To debug errors on the PineCone you need the following additional hardware:

- USB-JTAG adapter (like in Figure 1)
- Five female-male wires

For debugging purposes, the PineCone implements the JTAG protocol. This is a serial protocol that was developed to debug integrated circuits. The USB-JTAG adapter allows us to connect our PC with a device understanding the JTAG protocol.

Debugging with GDB and OpenOCD

To debug the PineCone using JTAG, we must connect the PineCone to your computer using USB. This cable is used to power the PineCone and can be used to flash applications on it. Furthermore, you must connect the USB-JTAG-adapter to your PC. Then, connect the following JTAG pins exposed by the adapter to the following GPIO pins of the PineCone:

- The JTAG pin TDO is connected to the IO pin 11.
- JTAG pin TMS is connected to IO pin 12.
- JTAG pin TCK is connected to IO pin 14.
- JTAG pin TDI is connected to IO pin 17.
- The ground pin of the JTAG adapter is connected to a ground pin of the PineCone.

Now your setup should similar to the one in Figure 2.

Besides setting up this connection, you need one more software tool, OpenOCD. This tool is used to remotely debug SoCs. Download OpenOCD from GitHub¹ and place it under `toolchain/openocd`.

Once you downloaded OpenOCD and the PineCone is connected to the PC via USB and JTAG we can start debugging:

¹<https://github.com/xpack-dev-tools/openocd-xpack/releases/tag/v0.12.0-6>



Figure 1: USB-JTAG adapter.

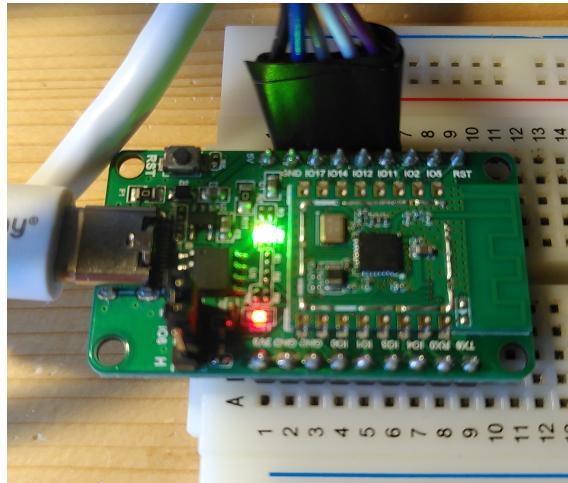


Figure 2: PineCone connected to a PC via JTAG.

1. Start OpenOCD:

```
./toolchain/openocd/bin/openocd -f ./tools/debug/openocd.cfg
```

2. Create a configuration for GDB (GNU Debugger):

```
cat > /tmp/gdb_tgt.cfg << EOF
target extended-remote :3333
EOF

cat ./tools/debug/602.init /tmp/gdb_tgt.cfg > /tmp/602.init
```

3. Start GDB:

```
./toolchain/compiler/bin/riscv32-unknown-elf-gdb -x /tmp/602.init --se=<ELF
    ↪ file>
```

Where the ELF file is the file <project name>.elf inside the *build_out* directory of your project.

If you run Debian Trixie, you can also use the *debug.sh* script located in the *tools* directory. In this case, run the script from your project's root (*customer_app/project*):

```
../../tools/debug.sh
```

Then, you should have two terminal windows: one running OpenOCD and another one running GDB (similar to Figure 3). In the OpenOCD window you can see the connection state and with GDB you debug your program. The most important commands for GDB are the ones listed in Table 1 [2, pp. 7–10]. You can find more information about OpenOCD and GNU GDB in their manuals [1, 2].

<code>break <function></code>	Sets a breakpoint for the given function, this means, the program is interrupted when the specified function is called. Then, you can enter further commands.
<code>run</code>	Starts running the specified program.
<code>c</code>	Continues running the program (e.g. after a breakpoint).
<code>n</code>	Runs the next line of code.
<code>s</code>	Steps into the next subroutine.
<code>l</code>	Shows up to the next 10 lines of code.
<code>bt</code>	Shows the backtrace (displays the stack for the currently running program part).
<code>p <variable></code>	Prints the specified variable.
<code>p <variable> = <expression></code>	Sets the variable to the given expression.
<code>lay next</code>	Switch layout of GDB.
<code>q</code>	Quits the session.

Table 1: Most important GDB commands.

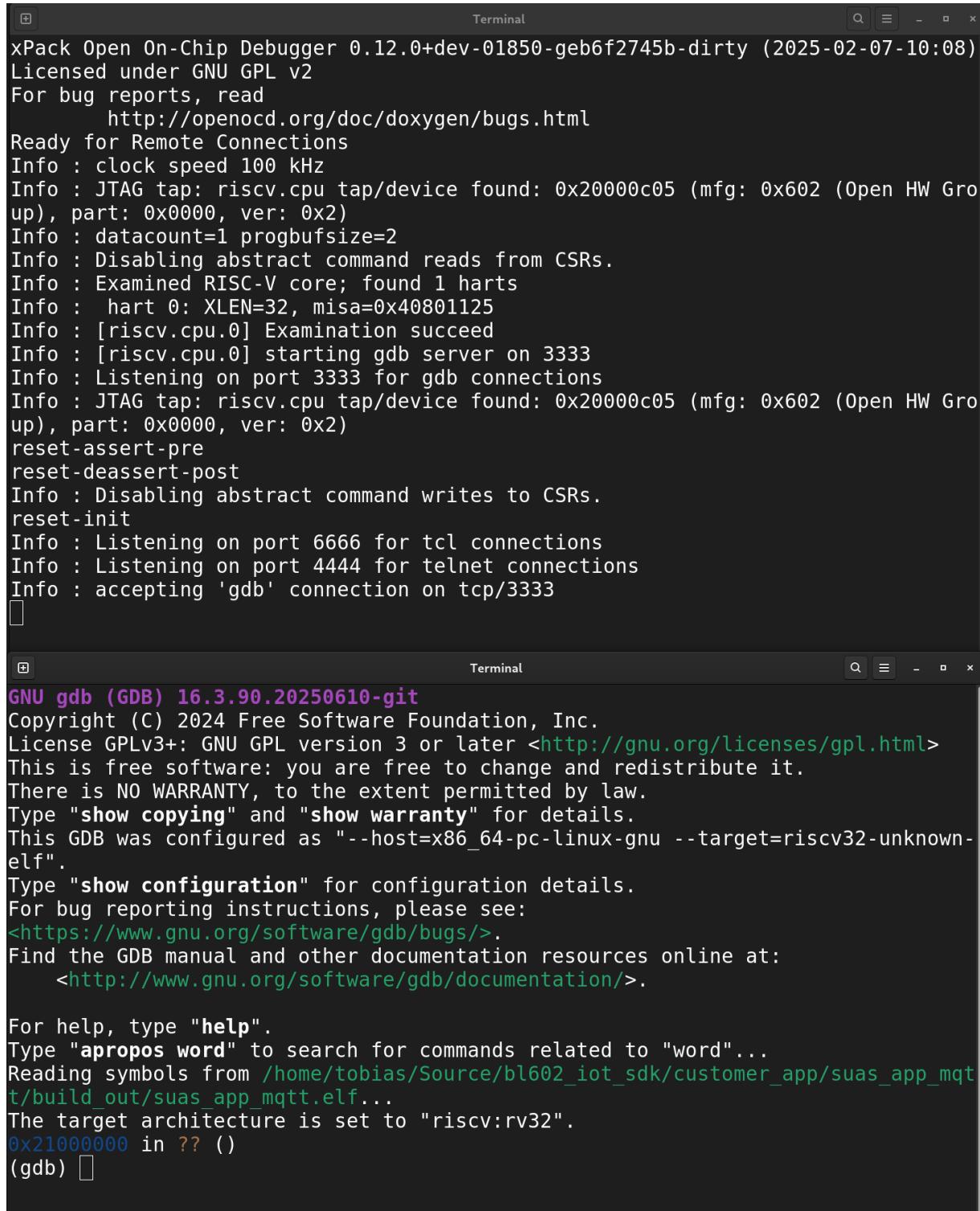
Notes

Please note that the PineCone's LEDs are used by OpenOCD to indicate the connection status (a green LED means everything is alright). Consequently, you can not debug applications making use of the PineCone's internal LEDs. To debug those applications you either have to disable the LED usage by your application or you must remap the pins of the PineCone².

References

- [1] The OpenOCD Project. *Open On-Chip-Debugger: OpenOCD User's Guide*. online. Dec. 2022. URL: <https://openocd.org/doc-release/pdf/openocd.pdf>.
- [2] Richard Stallman, Roland Pesch, and Stan Shebs. *Debugging with GDB*. online. 2025. URL: OpenOCD2022.

²More information about pin remapping: <https://lupyuen.org/articles/openocd>



The image shows two terminal windows side-by-side. The top window is titled "Terminal" and displays the output of the xPack Open On-Chip Debugger. It includes information about the debugger version (0.12.0+dev-01850-geb6f2745b-dirty), license (GNU GPL v2), bug reporting instructions, and details about the JTAG tap found (riscv.cpu, clock speed 100 kHz, part 0x0000, ver 0x2). It also shows the gdb server starting on port 3333 and accepting connections. The bottom window is also titled "Terminal" and displays the GNU gdb (GDB) 16.3.90.20250610-git. It provides copyright information, license terms (GPLv3+), and details about the configuration (host x86_64-pc-linux-gnu, target riscv32-unknown-elf). It also lists symbols from a file named suas_app_mqtt.build_out/suas_app_mqtt.elf and the target architecture (riscv:rv32).

```
xPack Open On-Chip Debugger 0.12.0+dev-01850-geb6f2745b-dirty (2025-02-07-10:08)
Licensed under GNU GPL v2
For bug reports, read
    http://openocd.org/doc/doxygen/bugs.html
Ready for Remote Connections
Info : clock speed 100 kHz
Info : JTAG tap: riscv.cpu tap/device found: 0x20000c05 (mfg: 0x602 (Open HW Group), part: 0x0000, ver: 0x2)
Info : datacount=1 progbufsize=2
Info : Disabling abstract command reads from CSRs.
Info : Examined RISC-V core; found 1 harts
Info : hart 0: XLEN=32, misa=0x40801125
Info : [riscv.cpu.0] Examination succeed
Info : [riscv.cpu.0] starting gdb server on 3333
Info : Listening on port 3333 for gdb connections
Info : JTAG tap: riscv.cpu tap/device found: 0x20000c05 (mfg: 0x602 (Open HW Group), part: 0x0000, ver: 0x2)
reset-assert-pre
reset-deassert-post
Info : Disabling abstract command writes to CSRs.
reset-init
Info : Listening on port 6666 for tcl connections
Info : Listening on port 4444 for telnet connections
Info : accepting 'gdb' connection on tcp/3333
[...]
GNU gdb (GDB) 16.3.90.20250610-git
Copyright (C) 2024 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 "--host=x86_64-pc-linux-gnu --target=riscv32-unknown-elf".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
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 /home/tobias/Source/bl602_iot_sdk/customer_app/suas_app_mqtt/build_out/suas_app_mqtt.elf...
The target architecture is set to "riscv:rv32".
0x21000000 in ?? ()
(gdb) [...]
```

Figure 3: PineCone connected to OpenOCD and GCB. Top: OpenOCD. The address 0x20000c05 refers to the PineCone. Bottom: GNU GDB.