AN14680

Debugging Kernel Space Using GDB and KGDB

Rev. 1.0 — 16 May 2025 Application note

Document information

Information	Content
Keywords	AN14680, Kernel GNU Debugger, i.MX 8M, Ubuntu 22.04
Abstract	This document provides a step-by-step guide for setting up Kernel GNU Debugger (KGDB) on a development board (target).

1 Introduction

This document provides a step-by-step guide for setting up GNU Debugger (GDB) on a Linux machine (host). GDB is a powerful debugging tool for programs written in various programming languages, including C and C++. The host further connects to Kernel GNU Debugger (KGDB) on a development board (target).

KGDB is an extension of GDB designed specifically for debugging the Linux kernel, including device drivers and kernel modules. This document includes instructions on configuring the kernel, sharing the serial port, and establishing a debugging session.

2 Prerequisites

Following are the prerequisites for the setup:

- NXP development board (target)
- Linux host machine with the appropriate cross-compile toolchain for the target
- Serial connection between the host and target

Set up the development board that is being debugged with KGDB (target) and a Linux machine with GDB (host). Connect the target to a serial console with host. Install the cross-compiled toolchain for your target, which includes the GDB, on the host.

This document has been tested on the following setup:

- Host machine: Running Ubuntu 22.04
- Target device: NXP i.MX 8M Plus Evaluation Kit (imx-8mp-evk) with BSP and toolchain version 6.6.3-1.0.0

3 Setting up serial port sharing with kdmx

Kdmx is a tool used along with KGDB which multiplexes connections. This tool allows both standard serial console and debugging session to share the serial port.

The KGDB requires a dedicated serial connection. Also, a console is needed for the system interaction. Since the board has only one serial port, the kdmx tool is used to multiplex the serial port to satisfy these requirements.

3.1 Install kdmx

Run the following commands on the host to install kdmx.

```
$ git clone https://kernel.googlesource.com/pub/scm/utils/kernel/kgdb/agent-
proxy
$ cd agent-proxy/kdmx/
$ make
$ gcc -Wall -Wunreachable-code -D_XOPEN_SOURCE -c -o kdmx.o kdmx.c
$ gcc -o kdmx kdmx.o
```

3.2 Run kdmx

To share the serial port, execute the following command:

```
./kdmx -p /dev/ttyUSB<X> -b 115200
```

X is the port number for the serial console.

All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

4 Configure and compile the kernel with KGDB support

This section explains the procedure to complete the KGDB installation.

4.1 Enable kernel configuration

Reconfigure and recompile the kernel with the following options:

- Enable the following configurations in the *.config* file:

```
CONFIG_KGDB=y
CONFIG_KGDB_SERIAL_CONSOLE=y
CONFIG_MAGIC_SYSRQ=y
CONFIG_DEBUG_INFO=y
CONFIG_FRAME_POINTER=y
```

- Disable the following configurations in the *.config* file:

```
CONFIG_DEBUG_INFO_REDUCED
CONFIG_RANDOMIZE_BASE
```

To build the kernel, refer to “How to build U-Boot and kernel in a standalone environment” of the *i.MX Linux User Guide* (document UG10163).

4.2 Verify debug symbols

Ensure the compiled vmlinux binary contains the debugging symbols by running the command given below:

```
$ file vmlinux
```

The expected output of running the above command is as below:

```
vmlinux: ELF 64-bit LSB pie executable, ARM aarch64, version 1 (SYSV),
 statically linked, BuildID[sha1]=2e57df8b4e05509036e463d820f9cc4d87304061, with
 debug_info, not stripped
```

Remember to search for ‘with debug_info, not stripped’ message.

4.3 Compile and flash the kernel

After compiling the kernel with debugging symbols, flash the kernel image to the target device. The kernel image is created in *<build_folder>/arch/arm64/boot/Image*. Copy and overwrite the image on the bootpartition of the SD card.

4.4 Verify KGDB installation

Check if KGDB is available on the target by running the command given below:

```
$ ls /sys/module/kgdboc/
```

The expected output is `parameters uevent`.

All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

5 Entering debug mode on the target

You can enter the KGDB debugging mode either at boot time or during runtime. The following sections explain both scenarios.

5.1 Entering debug mode at boot time

To enable KGDB, it is necessary to modify the U-Boot environment variables. Steps for this process are given below:

1. Enter U-Boot and modify the boot arguments as below:

```
u-boot=>  editenv mmcargs
```

Add `kgdboc=${console} kgdbwait` at the beginning of the arguments.

```
u-boot=>  printenv mmcargs
mmcargs=setenv bootargs kgdboc=${console} kgdbwait ${jh_clk} ${mcore_clk}
 console=${console} root=${mmcroot}
u-boot=> boot
```

2. The system pauses during boot. It displays the output as below:

```
[1.458834] KGDB: Registered I/O driver kgdboc
[1.463339] KGDB: Waiting for connection from remote gdb...
```

3. Now, the kernel is in debug mode. Proceed to connect GDB from the host.

5.2 Entering debug mode at runtime

To start KGDB while the system is running, follow the steps as given below:

1. Set the serial port for debugging.

```
$ echo <tty> > /sys/module/kgdboc/parameters/kgdboc
```

`tty` is the serial console for the target, which can be seen after the board boot in the first line as below:

```
NXP i.MX Release Distro 6.6-nanbield imx8mp-lpddr4-evk ttymxc1
imx8mp-lpddr4-evk login:
```

2. Trigger KGDB as below:

```
$ echo g > /proc/sysrq-trigger
```

3. Now, the kernel freezes and displays the below message:

```
[28.710921] KGDB: Entering KGDB
```

4. The kernel is now in debug mode. Proceed to connect GDB from the host.

6 Connecting to KGDB from the host

Once the target is in debugging mode, connect to KGDB using GDB from the host. Follow the steps for this process as below:

1. Switch to the root user.

```
$ sudo su
```


All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

2. Load the toolchain environment.

```
$ source /path/to/environment/toolchain
```

3. Launch GDB for vmlinux.

```
$ aarch64-poky-linux-gdb vmlinux
```

4. Inside GDB, configure the serial connection and connect to the target.

```
(gdb) set serial baud 115200
(gdb) target remote /dev/pts/X
```

`X` is the number for the GDB connection from kdmx.

Once connected, begin the debugging.

7 Debug a kernel module

To debug an external kernel module, you can either debug an already probed module or debug it during the probing process. The probing process is also referred to as the initialization function of the module.

Ensure that the module is compiled with debug flags. After compilation, copy the module to the Linux build folder of the host machine. For this example, use the module 'test.ko'.

7.1 Debug an already loaded kernel module

1. Load the module on the target with the command as given below:

```
$ modprobe test
```

2. Retrieve the address of the section namely ‘.text’, ‘.data’, and ‘.bss’ (optional).

```
$ cd /sys/module/test/sections/
$ cat .text .data .bss
```

Example output is as below:

```
0xffff80007a376000 - .text
0xffff80007a38c360 - .data
0xffff80007a38cf00 - .bss
```

3. Enter the debug mode on target (Refer to Section 5.2).
4. Connect to KGDB from the host machine (Refer to Section 6).
5. Add module addresses to the GDB with the command as given below:

```
(gdb) add-symbol-file <path-to-module> <address of module’s .text section> \
-s .data <address of module’s .data section> \
-s .bss <address of module’s .bss section>
```

For the test module, the command is given below.

```
(gdb) add-symbol-file test.ko 0xffff80007a376000 \
-s .data 0xffff80007a38c360 \
-s .bss 0xffff80007a38c360
```

6. Set breakpoints anywhere in the module sources and start the debugging process.

7.2 Debug a kernel module during loading

1. Connect to the target machine (Refer to Section 6).

All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

2. Locate the **load_module()** function in 'kernel/module/main.c' and identify the following lines:

```
2925 /*
2926 * Now we've got everything in the final locations, we can
2927 * find optional sections.
2928 */
2929 err = find_module_sections(mod, info);
2930 if (err)
2931     goto free_unload;
```

3. Set a breakpoint at `if (err)` (line 2930 for linux-imx lf-6.12.y), referred to as breakpoint 1.

```
(gdb) breakpoint kernel/module/main.c:2930
(gdb) continue
```

4. The target unfreezes. Load the module on it as below:

```
# insmod <path-to-module>
```

5. The GDB stops at breakpoint 1 as can be seen below:

```
Thread 186 hit Breakpoint 1, load_module (info=info@entry=0xffff80000adfbd30,
 uargs=uargs@entry=0xaaaaaab8ede0 "", flags=flags@entry=0) at kernel/module/
main.c:2803
2803            if (err < 0)
```

6. Retrieve the module sections addresses with the command as given below:

```
(gdb) print mod->init_layout.base (.init.text section address)
 0xffff800001345000
(gdb) print mod->core_layout.base (.text section address)
 0xffff800001340000
```

7. Add addresses to the GDB as below:

```
(gdb) add-symbol-file test.ko -s .init.text 0xffff800001345000 -s .text
 0xffff800001340000
```

8. Set a breakpoint in the initialization function of the module as below:

```
(gdb) breakpoint custom_init
```

9. Delete the first breakpoint as below. This step prevents the repeated GDB stops.

```
(gdb) delete breakpoint 1
(gdb) continue
```

10. The GDB stops at the second breakpoint, entering the initialization function of the module.

8 Note about the source code in the document

Example code shown in this document has the following copyright and GPL-2.0-only license:

Copyright 2025 NXP

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; version 2.

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, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.

All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

9 Revision history

Table 1 summarizes revisions to this document.

Table 1. Revision history

Document ID	Release date	Description
AN14680 v.1.0	16 May 2025	• Initial public release

All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

All information provided in this document is subject to legal disclaimers.

© 2025 NXP B.V. All rights reserved.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, µVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

Contents

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

© 2025 NXP B.V. **All rights reserved.**