RZ/T2H and RZ/N2H Group

Linux Start-up Guide for RZ/T2H and RZ/N2H EVKIT

Introduction

This document provides a guide to prepare RZ/T2H or RZ/N2H evaluation board to boot up with the Verified Linux Package.

This guide provides the following information:

- Building procedure
- Preparation for use
- Boot loader and U-Boot
- How to run this Linux package on the target board
- How to create a software development kit (SDK)

Target Reference Board

- RZ/T2H or RZ/N2H Evaluation Board

Target Software

- RZ/T2H or RZ/N2H Verified Linux Package Rev.5.00 or later. (hereinafter referred to as "VLP")

Contents

1. Environmental Requirement

The environment for building the Verified Linux Package (hereinafter referred to as "VLP") is listed in Table 1-1. Please refer to the documents below for details about setting up the environment:

Figure 1-1 shows the recommended environment for this package.

A Linux PC is required for building the software.

A Windows PC can be used as a serial terminal interface with software such as TeraTerm.

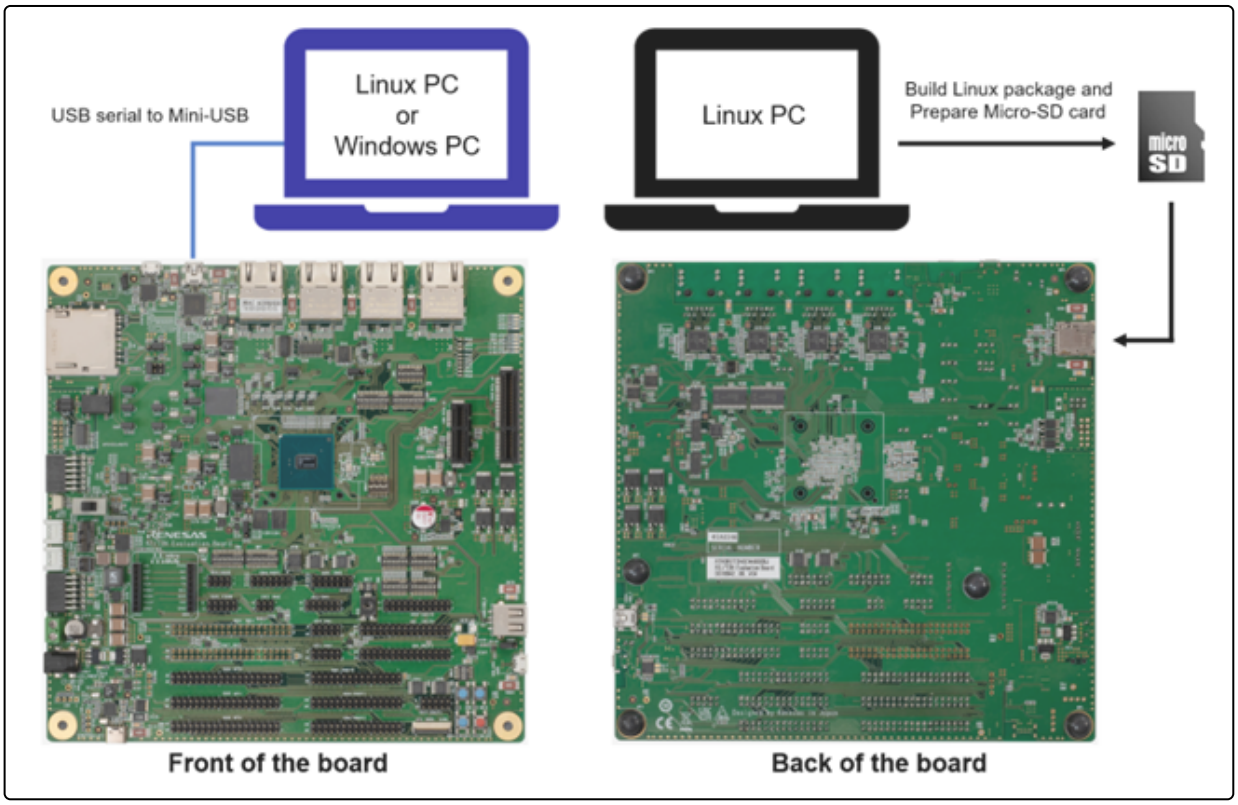


Figure 1-1: Recommend Environment

Table 1-1 Equipment and Software for Developing Environments of Linux Platform

Equipment		Description
Linux Host PC	-	Used as build/debug environment 100GB free space on HDD or SSD is necessary
	OS	Ubuntu 22.04 LTS 64 bit OS must be used. 22.04 inside a docker container also OK.
Windows Host PC	-	Used as debug environment, controlling with terminal software
	OS	Windows 11
	Terminal software	Used for controling serial console of the target board Tera Term (latest version) is recommended Available at: https://github.com/TeraTermProject/teraterm/releases
	VCP Driver	Virtual COM Port driver which enables to communicate Windows Host PC and the target board via USB which is virtually used as serial port. Available at: http://www.ftdichip.com/Drivers/VCP.htm
USB serial to mini-USB Cable		Serial communication (UART) between the Evaluation Board Kit and Windows PC. The type of USB serial connector on the Evaluation Board Kit is USB type mini-B.
micro.SD Card		Use to boot the system, and store applications

2. Build Instructions

2.1 Required Host OS

Ubuntu 22.04 is required to build the VLP. This is because it was the only host operating system tested and is a specific requirement for Yocto 5.0 (Scarthgap).

2.2 Building images

This section describes the instructions to build the Verified Linux Package.

Before starting the build, run the command below on the Linux Host PC to install packages used for building the VLP.

```
$ sudo apt-get update
$ sudo apt-get install build-essential chrpath cpio debianutils diffstat file gawk gcc git iputils-ping libacl1 \
liblz4-tool libssl-dev libyaml-dev locales python3 python3-git python3-jinja2 python3-pexpect python3-pip \
python3-subunit socat texinfo tmux unzip wget xterm xz-utils zstd bmap-tools libsdl1.2-dev p7zip-full fdisk tmux
```

Please refer to the URL below for detailed information:

- https://docs.yoctoproject.org/5.0.11/brief-yoctoprojectqs/index.html

Run the commands below and set the username and email address before starting the build procedure. Without this setting, an error occurs when building procedure runs git command to apply patches.

```
$ git config --global user.email "you@example.com"
$ git config --global user.name "Your Name"
```

Copy all files obtained from Renesas into your Linux Host PC prior to the steps below. The directory which you put the files in is described as in the build instructions.

2.2.1 Set the environment variable of the package

Set the version number of the package you are using as an environment variable. The following is an example, so please rewrite it to match your package.

```
$ PACKAGE_VERSION=5_0_0
```

2.2.2 Create a working directory at your home directory, and decompress Yocto recipe package

Run the commands below. The name and the place of the working directory can be changed as necessary.

```
$ mkdir ~/rz_vlp_v${PACKAGE_VERSION}
$ cd ~/rz_vlp_v${PACKAGE_VERSION}
$ cp ../<package download directory>/*.zip .
$ unzip ./RTK0EF0045Z0042AZJ-rz-vlp-v${PACKAGE_VERSION}.zip
$ tar zxvf ./ RTK0EF0045Z0042AZJ-rz-vlp-v${PACKAGE_VERSION}/rz_vlp_v${PACKAGE_VERSION}.tar.gz
```

NOTE:

Please note that your building environment must have 100GB of free hard drive space to complete the minimum build. The Yocto VLP build environment is very large. Especially in case you are using a Virtual Machine, please check how much disk space you have allocated for your virtual environment.

2.2.3 Build Initialize

Initialize a build using the 'oe-init-build-env' script in Poky and point TEMPLATECONF to platform conf path.

```
$ TEMPLATECONF=$PWD/meta-renesas/meta-rz-boards/conf/templates/vlp-v5-conf/ source poky/oe-init-build-env build
```

2.2.4 Decompress OSS files to “build” directory (Optional)

Run the commands below. This step is not mandatory and able to go to the next step in case the "offline" environment is not required. All OSS packages will be decompressed with this '7z' command.

```
$ cp ../../<package download directory>/*.7z .
$ 7z x oss_pkg_rz_v${PACKAGE_VERSION}.7z
```

NOTE:

If you skip this step, the bitbake command will download all source codes from the repositories of each OSS over the internet. Please be aware that if you are not in an "offline" environment, the building might fail due to unexpected changes in the OSS repositories.

Open source software packages include all the source codes of the OSS components. These are the exact versions of the OSS used during the verification of VLP. If you are simply evaluating VLP and the RZ/T2H and RZ/N2H group, using these open source software packages are not necessary. Generally, all the software can be built without these files if your building machine has an internet connection.

Open source software packages are necessary for an "offline" environment. An "offline" environment is defined as an isolated environment without any network connection. VLP can consistently build images in this "offline" environment using these packages, unaffected by any modifications in the original OSS repositories.

Furthermore, this "offline" environment always produces the same images that Renesas verified. Keep in mind that building without these open-source software packages could lead to the use of different source codes than those used by Renesas, due to potential unexpected changes in the OSS repositories.

After the above procedure is completed, the "offline" environment is ready. If you want to prevent network access, please change the line in the "~/rz_vlp_v${PACKAGE_VERSION}/build/conf/local.conf" as below:

```
BB_GENERATE_MIRROR_TARBALLS = "1"
BB_GENERATE_SHALLOW_TARBALLS = "1"
BB_GIT_SHALLOW = "1"
BB_GIT_SHALLOW_DEPTH = "1"
BB_NO_NETWORK = "1"
INHERIT += "own-mirrors"
SOURCE_MIRROR_URL = "file://<package download directory>/own-mirror"
```

Then other build PC also can refer same "package download directory" instead decompress the OSS file by themselves.

2.2.5 Start a build

Run the commands below to start a build. Building an image can take up to a few hours depending on the user's host system performance.

Build the target file system image using bitbake.

```
$ MACHINE=<board> bitbake core-image-minimal
```

board can be selected by referring to the following table.

Table 2-1 List of platforms and the board

Renesas MPU	board
RZ/T2H	rzt2h-dev
RZ/N2H	rzn2h-dev

After the building is successfully completed, a similar output will be seen, and the command prompt will return.

```
NOTE: Tasks Summary: Attempted 6018 tasks of which 0 didn't need to be rerun and all succeeded.
Summary: There were 52 WARNING messages.
```

All necessary files listed in the below table will be generated by the bitbake command and will be in the "build/tmp/deploy/images" directory.

Table 2-2 Image files

RZ/T2H	Linux kernel	Image-rzt2h-dev.bin
	Device tree file	r9a09g077m44-dev.dtb
	root filesystem	core-image-minimal-rzt2h-dev.rootfs.tar.bz2
	Boot loader	bl2_bp_xspi0-rzt2h-dev.srec fip-rzt2h-dev.srec
	SD image	core-image-minimal-rzt2h-dev.rootfs.wic.gz core-image-minimal-rzt2h-dev.rootfs.wic.bmap
RZ/N2H	Linux kernel	Image-rzn2h-dev.bin
	Device tree file	r9a09g087m44-dev.dtb
	root filesystem	core-image-minimal-rzn2h-dev.rootfs.tar.bz2
	Boot loader	bl2_bp_xspi0-rzn2h-dev.srec fip-rzn2h-dev.srec
	SD image	core-image-minimal-rzn2h-dev.rootfs.wic.gz core-image-minimal-rzn2h-dev.rootfs.wic.bmap

If you want to know the components installed to the root filesystem, please check the manifest file. The manifest file is created to the following path after building the images:

- ~/rz_vlp_v${PACKAGE_VERSION}/build/tmp/deploy/images/rzt2h-dev/core-image-minimal-rzt2h-dev.manifest
- ~/rz_vlp_v${PACKAGE_VERSION}/build/tmp/deploy/images/rzn2h-dev/core-image-minimal-rzn2h-dev.manifest

2.3 Notes for building

2.3.1 Install ethtool command

"ethtool" command is a command for configuring network interfaces. If you want to install the ethtool command on Linux, add the following line to "~/rz_vlp_v${PACKAGE_VERSION}/build/conf/local.conf" and rebuild the VLP.

```
IMAGE_INSTALL:append = " ethtool"
```

2.3.2 Patch for Changing Ethernet port configuration on the evaluation board

The ethernet ports on the evaluation board are configured as follows:

Table 2-3 Default Port Configuration

Ethernet Port Number (*)	Configuration	Note
ETH0	MII	Ethernet port or Ethernet Switch port
ETH1	MII	Ethernet port or Ethernet Switch port
ETH2	RGMII	Ethernet port
ETH3	RGMII	Ethernet port

(*) The ethernet port number is printed on the evaluation board as follows.

Figure 2-1: Example Image of Ethernet port

If you want to change the configuration, run the following commands to apply a patch, add a line in "~/rz_vlp_v${PACKAGE_VERSION}/build/conf/local.conf" according to your board configuration, and build VLP as previous section.

Please note that 2 patterns of configuration are supported now.

For both RZ/T2H and RZ/N2H

```
$ cd ~/rz_vlp_v${PACKAGE_VERSION}/meta-renesas
$ patch -p1 < ../extra/0001-rz-bsp-Add-support-changing-ports-to-MII-RGMII-for-R.patch
$ cd ..
```

Pattern 1: Change ETH0,1 to RGMII

Add the following line to local.conf.

```
MACHINE_FEATURES:append = " eth01_rgmii"
```

Pattern 2: Change ETH2,3 to MII

Add the following line to local.conf.

```
MACHINE_FEATURES:append = " eth23_mii"
```

After applying a patch and adding the line to local.conf, start the build according to previous section.

3. Preparing the SD Card

This chapter describes how to prepare the micro-SD card. This is used to boot the evaluation board, so please follow the instructions.

RZ/T2H is explained as an example.

3.1 Insert the micro-SD card into your Linux PC

3.2 Check the device name and unmount

Run the command below to determine the device name assigned to the micro-SD card. In this example, it's "/dev/sdb". If necessary, replace "/dev/sdb" with the correct device name for your system. After that, unmount this device. Example is as below:

```
$ sudo fdisk -l
$ umount /dev/sdb1
$ umount /dev/sdb2
```

NOTE:

After run "sudo fdisk -l", below log would output:

```
Disk /dev/sdb: 3.74 GiB, 3997171712 bytes, 7806976 sectors
Disk model: Storage Device
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0xxxxxxxxxx
```

3.3 Expand disk image

Run the command below to expand the disk image onto the micro-SD card. When you complete the following steps, the micro-SD card is prepared correctly.

```
$ cd ~/rz_vlp_v${PACKAGE_VERSION}
$ sudo bmaptool copy ./build/tmp/deploy/images/rzt2h-dev/core-image-<target>-rzt2h-dev.wic.gz /dev/sdb
```

Table 3-1 File and directory in the micro-SD card

Type/Number	Size	Filesystem	Contents
Primary #1	500MB (minimum 128MB)	FAT32	bl2_bp_xspi0-rzt2h-dev.srec fip-rzt2h-dev.srec
Primary #2	All remaining	Ext4	See directory structure below.

Directory structure (Ext4 partition):

```
./
├── bin
├── boot
│   ├── Image
│   └── r9a09g077-dev.dtb
├── dev
├── etc
├── home
├── lib
├── media
├── mnt
├── opt
├── proc
├── root
├── run
├── sbin
├── srv
├── sys
├── tmp
├── usr
└── var
```

4. Reference Board Setting

4.1 Prerequisites

Chapter 4 introduces the switches, jumper pin settings, and other configuration options on the evaluation board. Thesechanges are based on the factory default settings of the evaluation board and are intended for running Linux. Please note that changing settings other than those described here may cause Linux to malfunction.

If you want to switch ethernet ports on the evaluation board to RGMII mode or MII mode, you will need to modify the board. Please refer to the manual below for your specific board for instructions on how to modify it. You can also find more detailed information about board settings in the manual for your specific board.

- RZ/T2H Evaluation Board User's Manual

4.2 Overview of the evaluation environment

The following environment of Hardware and Software is used in the evaluation steup.

Hardware preparation (Users should purchase the following equipment.):

- USB Type-C cable compatible with USB PD. (e.g. AK-A8485011 (manufactured by Anker))
- USB PD Charger 45W (15V 3.0A) or more. (e.g. PowerPort III 65W Pod (manufactured by Anker))
- USB Type-A miniB cable (Any cables)
- micro HDMI cable (Any cables) (Optional)
- PC Installed FTDI VCP driver and Terminal software (Tera Term) (*1)

(*1) Please install the FTDI driver that can be following website (Releases TeraTermProject/teraterm(github.com)).

(*2) Display Output module is avalable for rent. Please refer to section "How to enable display output (console ouput)".

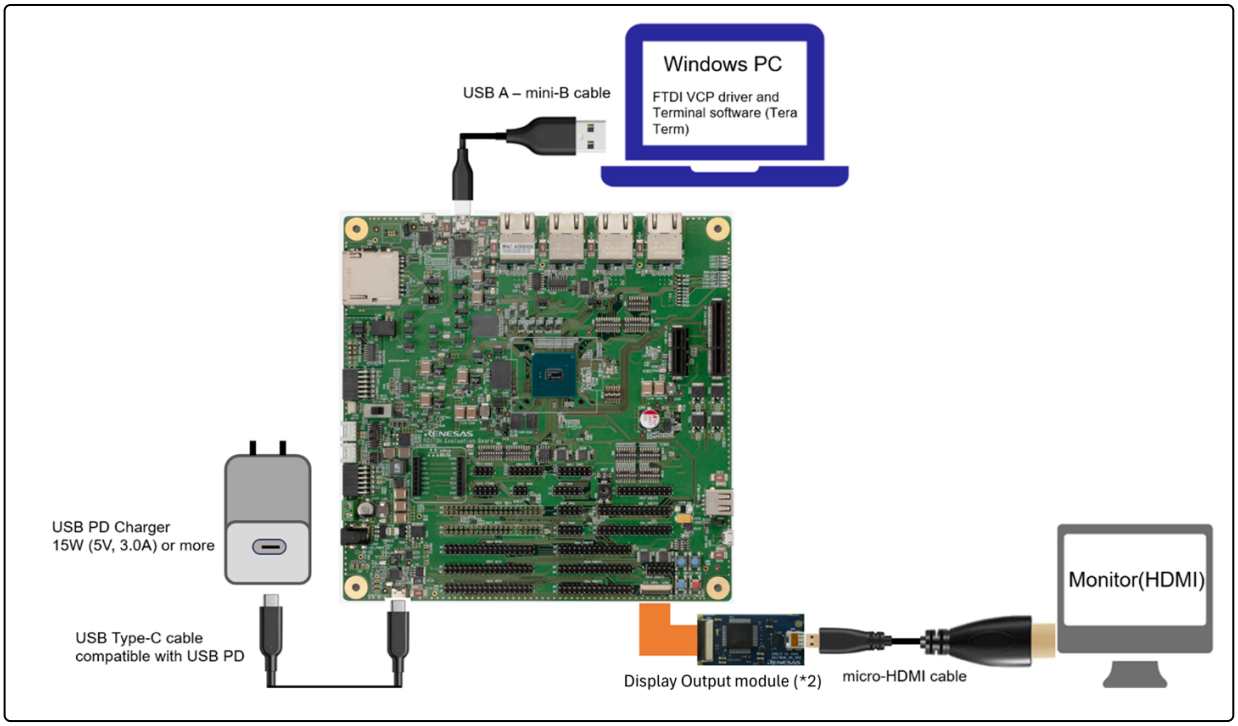


Figure 4-1: Recommend Environment

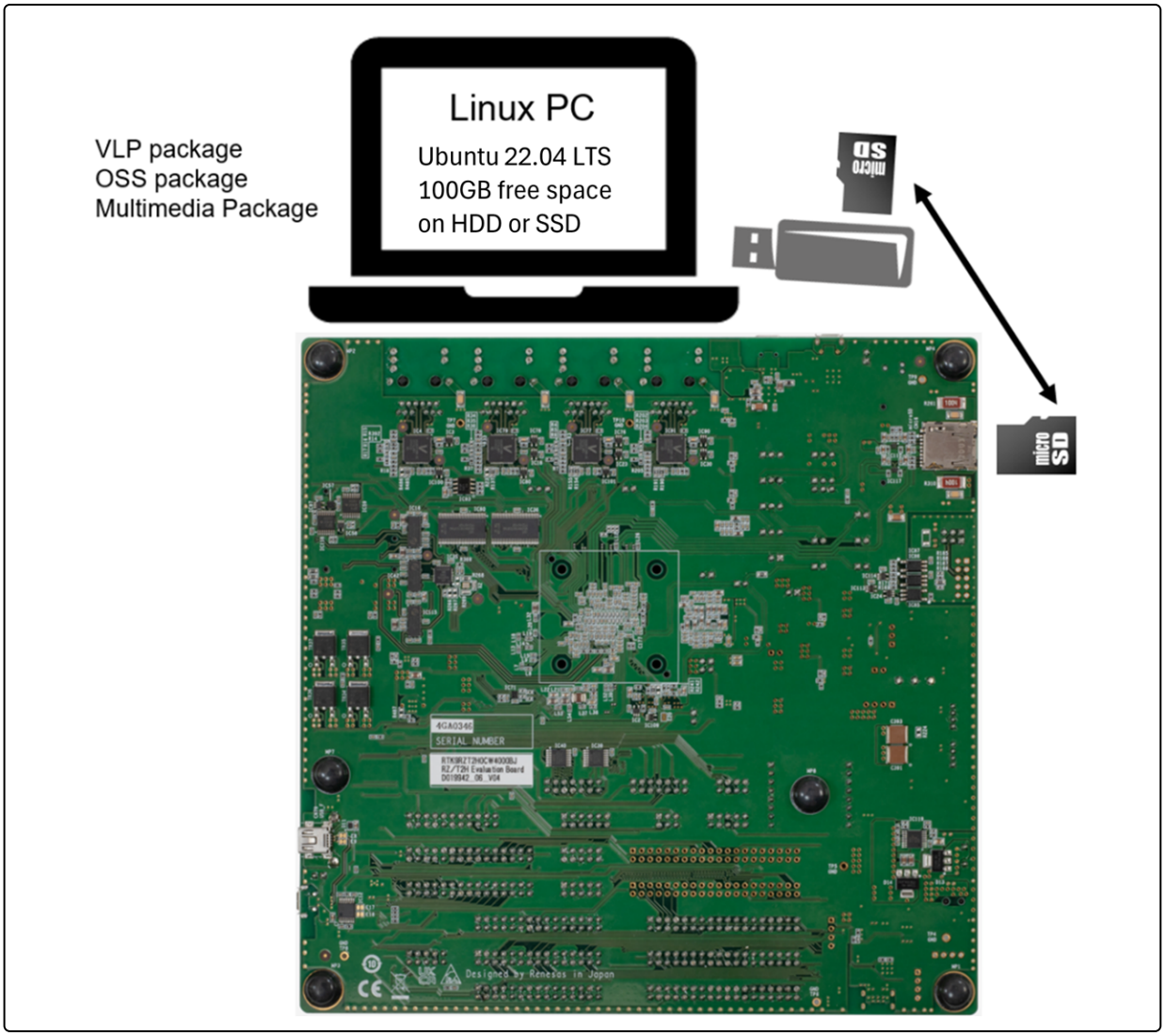


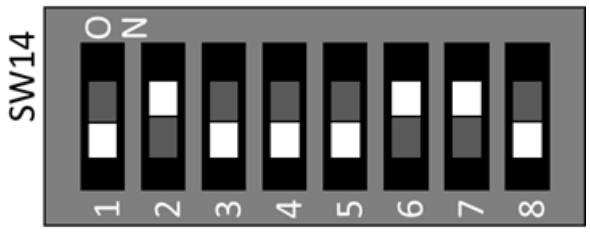
Figure 4-2: Recommend Environment (2)

4.3 Board Settings of RZ/T2H Evaluation Board

4.3.1 How to set boot mode of RZ/T2H

SCIF Download Mode

Set the SW14 settings as follows. This mode is used to write bootloaders.

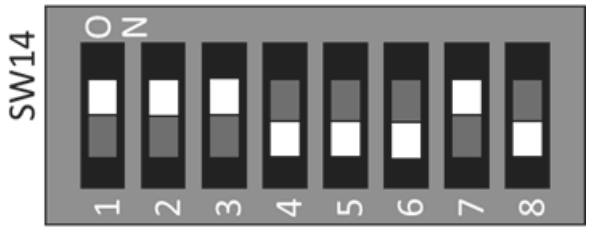


SW14-1	OFF *
SW14-2	ON
SW14-3	OFF *
SW14-4	OFF
SW14-5	OFF
SW14-6	ON *
SW14-7	ON
SW14-8	OFF

Figure 4-3: SCIF Download MODE(SW14)

xSPI Boot Mode

Set the SW14 settings as follows. This mode is used to boot Linux.



SW14-1	ON *
SW14-2	ON
SW14-3	ON *
SW14-4	OFF
SW14-5	OFF
SW14-6	OFF *
SW14-7	ON
SW14-8	OFF

Figure 4-4: xSPI Boot MODE(SW14)

4.3.2 How to set CN77

Set CN77 As follows.

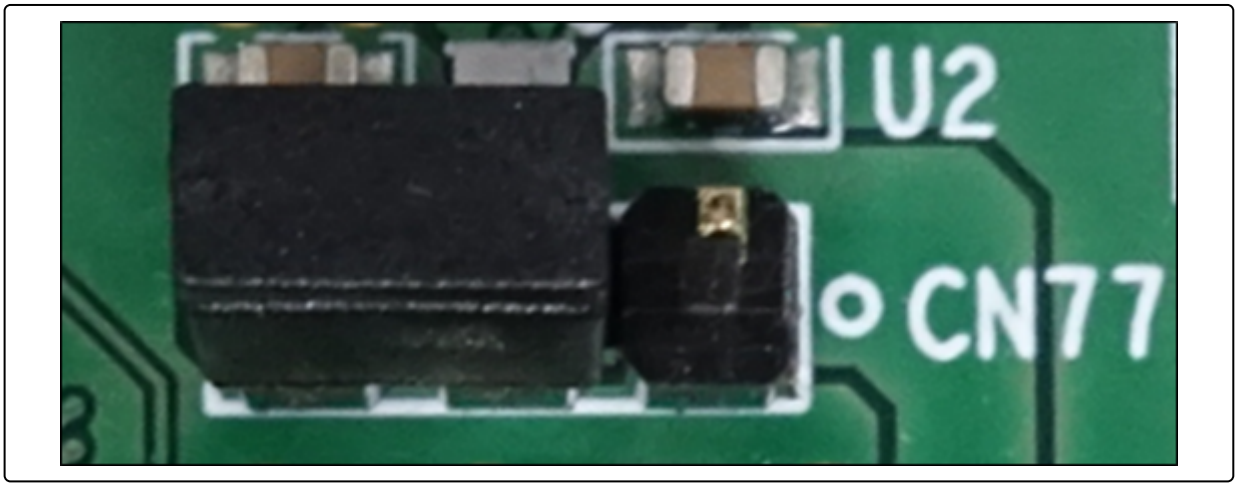


Figure 4-5: CN77 Setting

4.3.3 How to use debug serial (console ouput)

Connect USB Type Mini-B cable to CN34: USB Type Mini-B Connector.

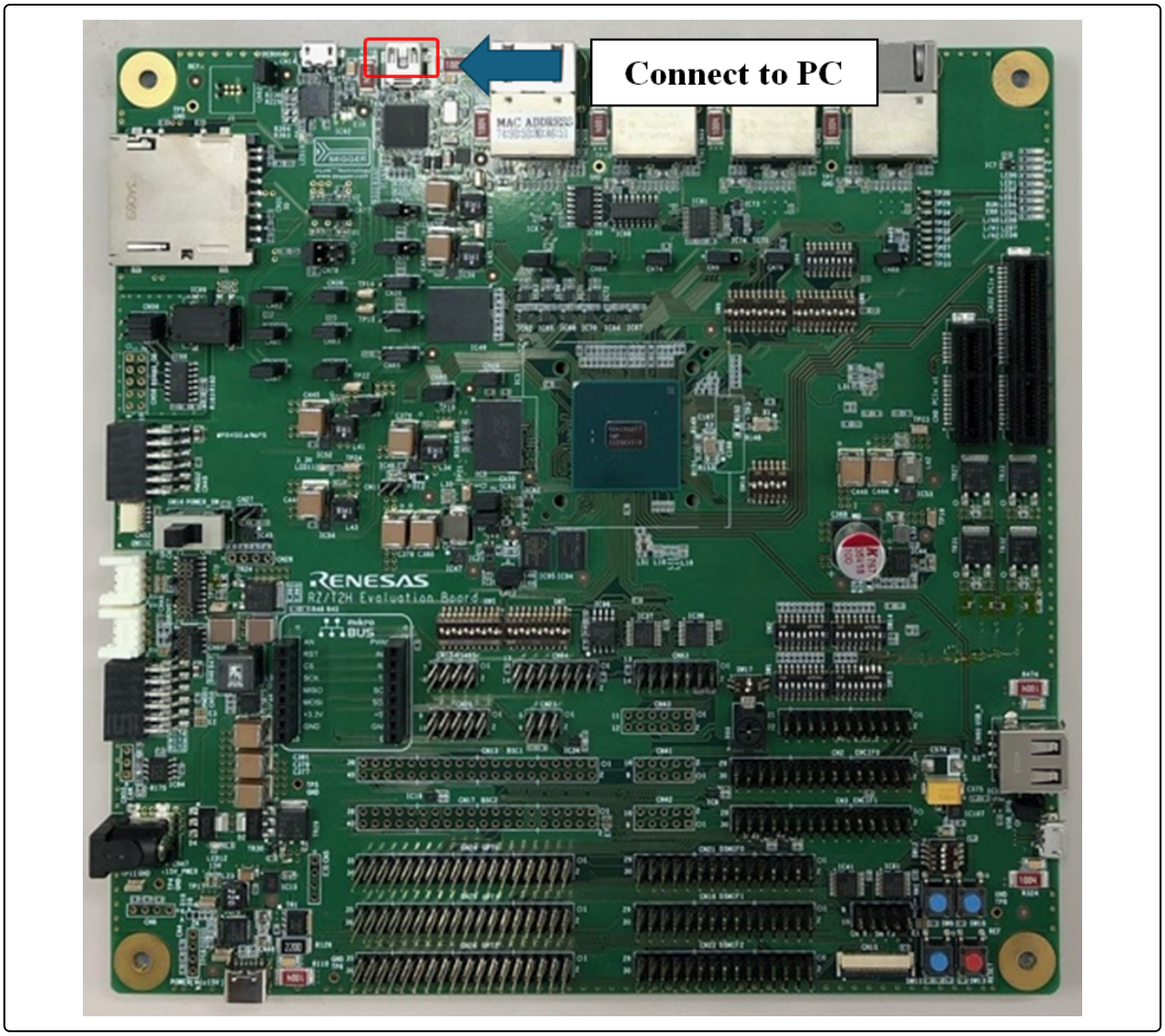


Figure 4-6: Connecting console for debug

4.3.4 How to enable the ethernet port of ETH2 CN45 (Optional)

The ethernet port of ETH3 CN1 is enabled as default. If you want to use ETH2 CN45 as well, refer to the below settings.

Set SW2-6 OFF

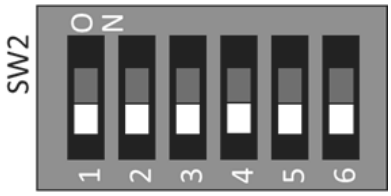


Figure 4-7: SW2-6 OFF

Connect pins 2 and 3 on CN9 with a jumper.

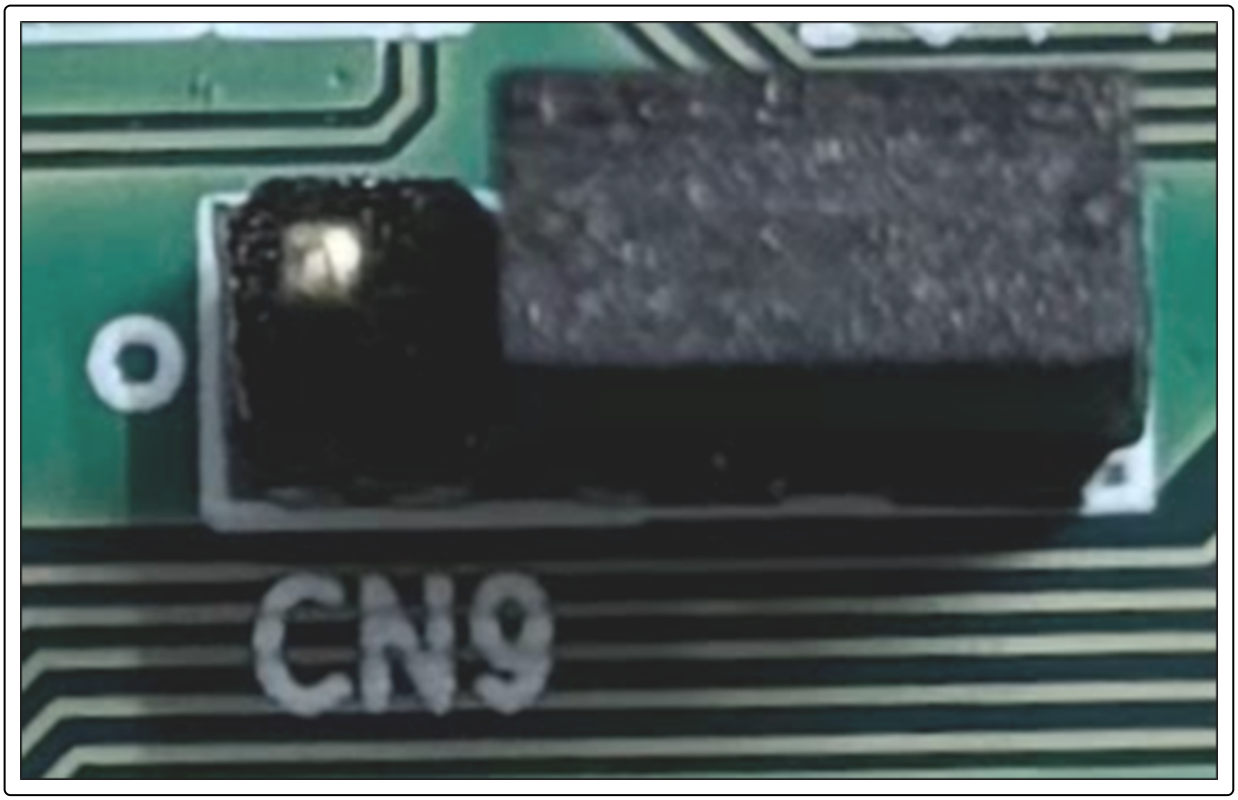


Figure 4-8: Pins 2 and 3 on CN9

Set SW6-2 ON and SW6-3 OFF

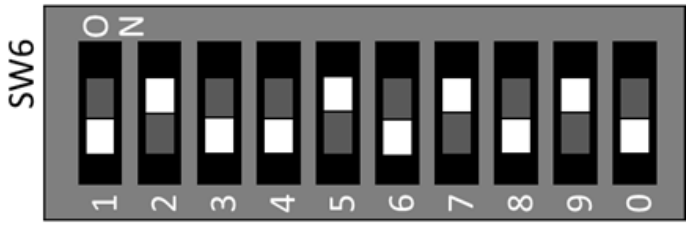


Figure 4-9: SW6-2 and SW6-3

4.3.5 How to set for Ethernet Port as RGMII and MII (console ouput)

When switching the Ethernet ports to RGMII or MII modes, you need to change some board configuration. Please refer to the manual below for instructions on how to modify it.

- RZ/T2H Evaluation Board User's Manual

4.3.6 How to enable display output (console ouput)

The steps in this section are optional. Skip this step when you don't use the display output.

Please note that the display output module (and the flexible cable) in this section is not included in the evaluation board kit.

The display output module (and the flexible cable) is available for rent. For details, please contact the retailer from which you purchased the evaluation board kit or your local Renesas Electronics sales office or distributor.

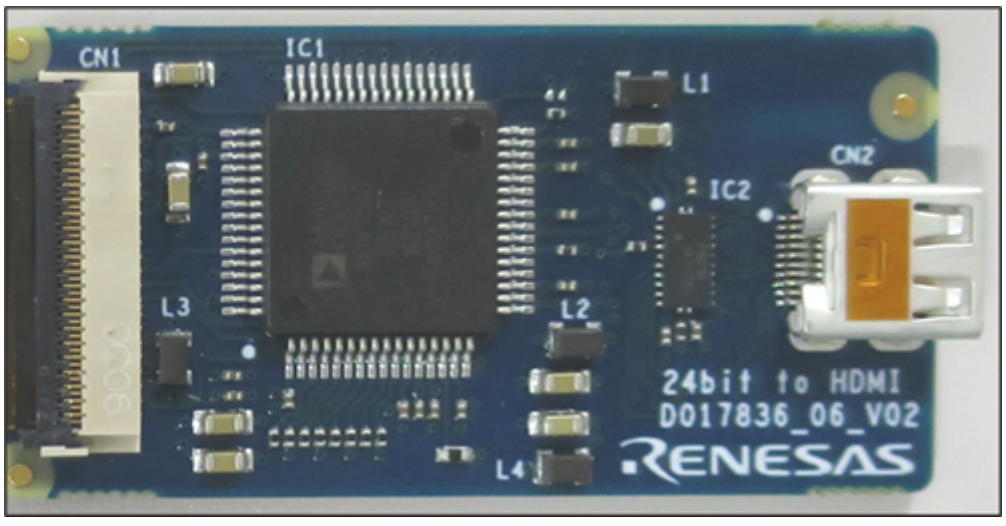


Figure 4-10: Display output module

How to connect the display output module to the evaluation board.

Please connect the flexible cable and the connector correctly as shown below. When the step is wrong, the output fails. This step is optional. Skip this step when you do not use the display output.

Connect the flexible cable to the CN1 on the display output module. Install it with the side that does not have any silk facing upwards.

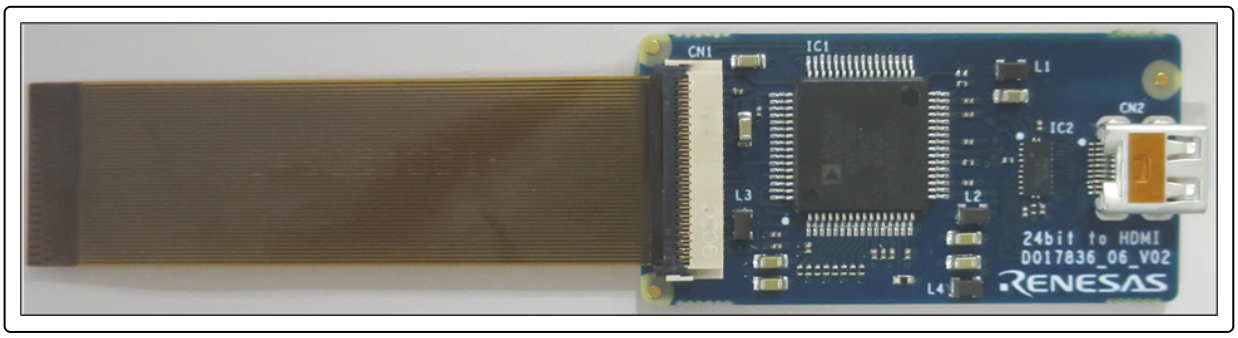


Figure 4-11: Connection of the display output module and flexible cable

Connect the flexible cable to CN15 on the evaluation board as the image below. When installing it, ensure that you do not install it in the wrong orientation.

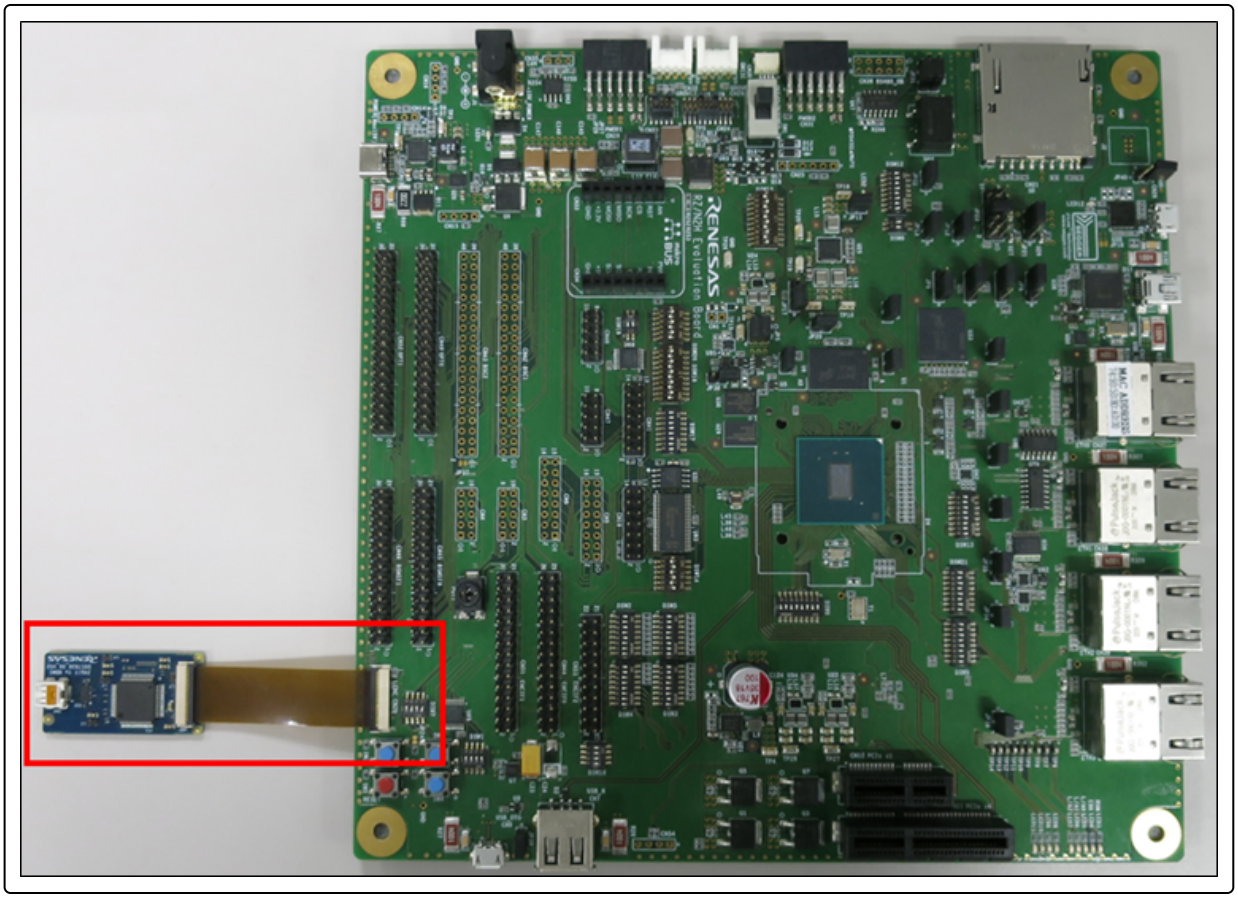


Figure 4-12: Connection of the display output module and Evaluation Board

How to configure DIP Switch to enable LCDC

Change the DIP switches as the following table.

SW2-1	ON
SW2-2	OFF
SW2-3	ON

SW6-3	OFF
SW6-4	ON
SW6-5	OFF

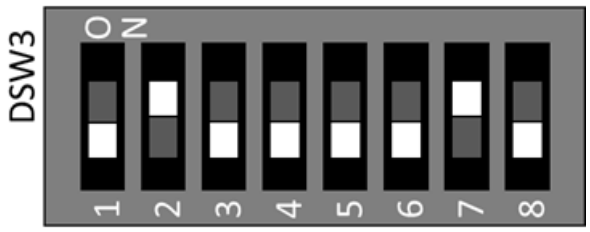
SW8-1	OFF
SW8-2	ON
SW8-3	OFF
SW8-4	ON

4.4 Board Settings of RZ/N2H Evaluation Board

4.4.1 How to set boot mode of RZ/N2H

SCIF Download Mode

Set the DSW3 settings as follows. This mode is used to write bootloaders.

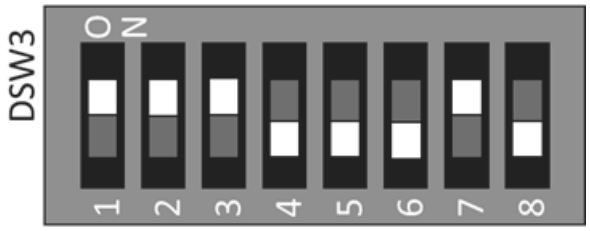


DSW3-1	OFF *
DSW3-2	ON
DSW3-3	OFF *
DSW3-4	OFF
DSW3-5	OFF
DSW3-6	OFF
DSW3-7	ON
DSW3-8	OFF

Figure 4-13: SCIF Download MODE(DSW3)

xSPI Boot Mode

Set the DSW3 settings as follows. This mode is used to boot Linux.



DSW3-1	ON *
DSW3-2	ON
DSW3-3	ON *
DSW3-4	OFF
DSW3-5	OFF
DSW3-6	OFF
DSW3-7	ON
DSW3-8	OFF

Figure 4-14: xSPI Boot MODE(DSW3)

4.4.2 How to use debug serial (console ouput)

Connect USB Type Mini-B cable to CN34: USB Type Mini-B Connector.

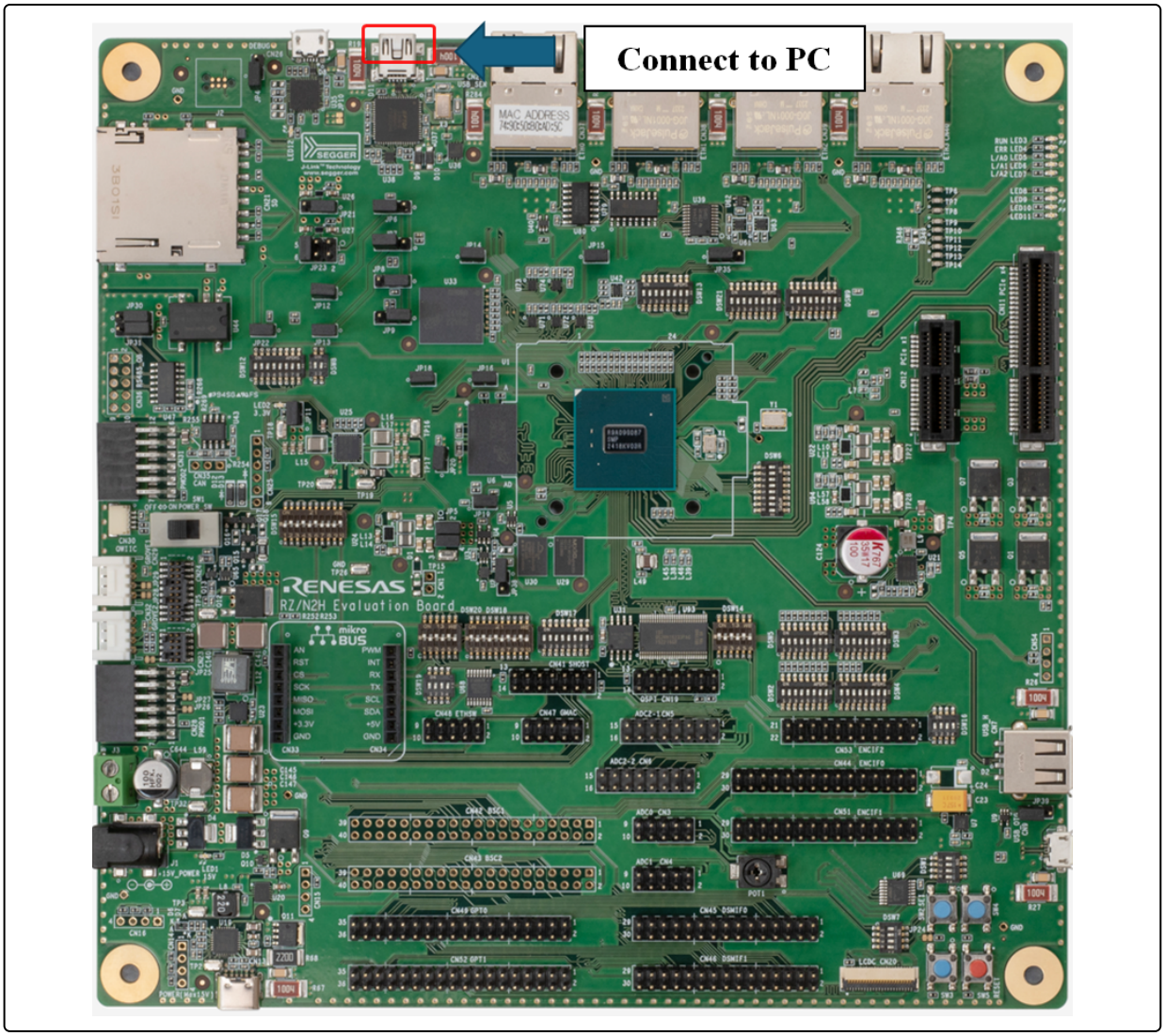


Figure 4-15: Connecting console for debug

4.4.3 How to enable the ethernet port of ETH2 CN39 (Optional)

The ethernet port of ETH3 CN40 is enabled as default. If you want to use ETH2 CN39 as well, refer to the below settings.

Set DSW5-6 OFF

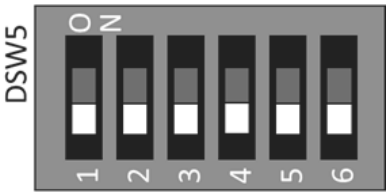


Figure 4-16: DSW5-6 OFF

Connect pins 2 and 3 on JP35 with a jumper.

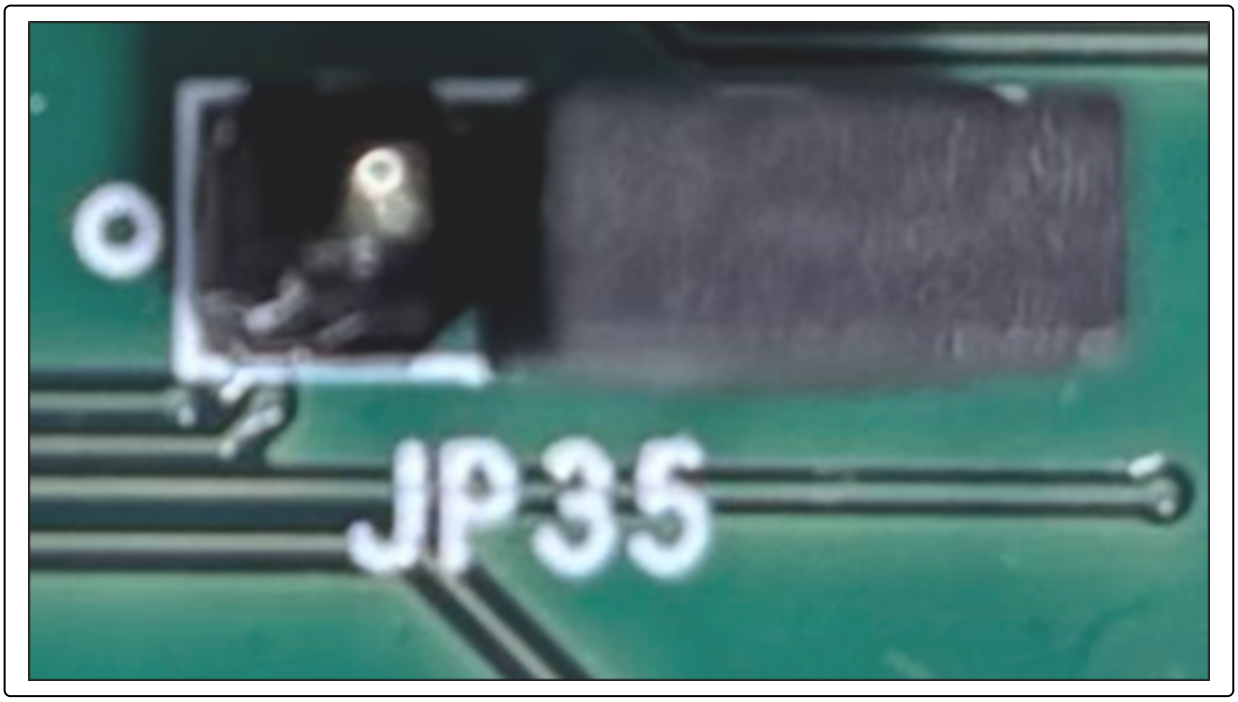


Figure 4-17: Pins 2 and 3 on JP35

4.4.4 How to set for Ethernet Port as RGMII and MII (console ouput)

When switching the Ethernet ports to RGMII or MII modes, you need to change some board configuration. Please refer to the manual below for instructions on how to modify it.

- RZ/N2H Evaluation Board User's Manual

4.4.5 How to enable display output (console ouput)

The steps in this section are optional. Skip this step when you don't use the display output.

Please note that the display output module (and the flexible cable) in this section is not included in the evaluation board kit.

The display output module (and the flexible cable) is available for rent. For details, please contact the retailer from which you purchased the evaluation board kit or your local Renesas Electronics sales office or distributor.

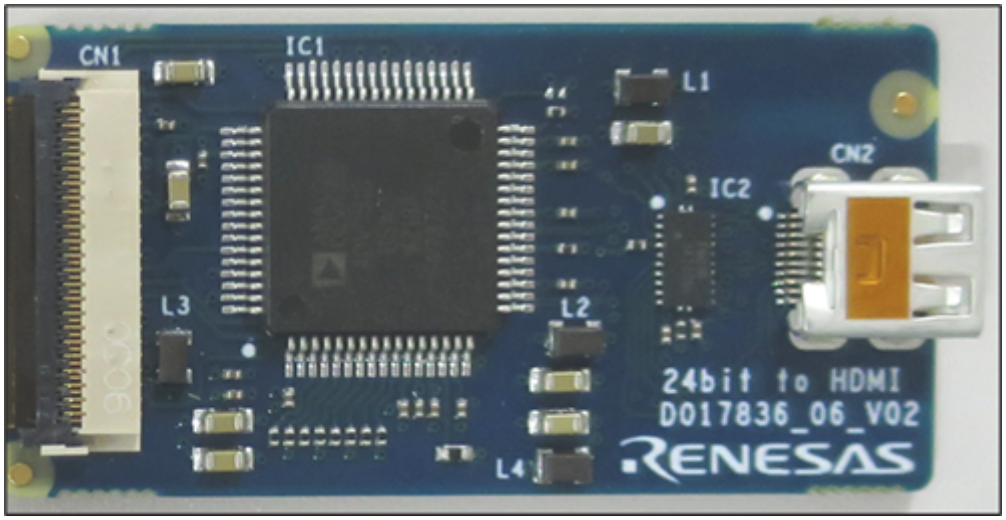


Figure 4-18: Display output module

How to connect the display output module to the evaluation board.

Please connect the flexible cable and the connector correctly as shown below. When the step is wrong, the output fails. And set DIP switch DSW15-8 to ON. This step is optional. Skip this step when you do not use the display output.

Connect the flexible cable to the CN1 on the display output module. Install it with the side that does not have any silk facing upwards.

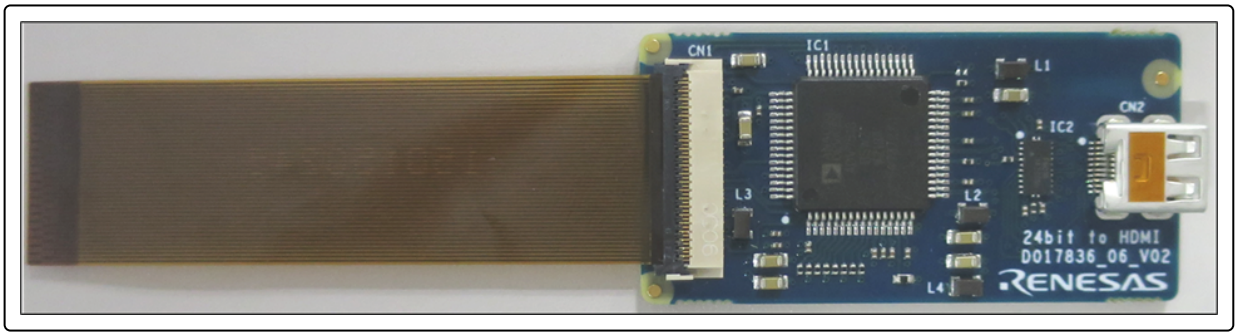


Figure 4-19: Connection of the display output module and flexible cable

Connect the flexible cable to CN20 on the evaluation board as the image below. When installing it, ensure that you do not install it in the wrong orientation.

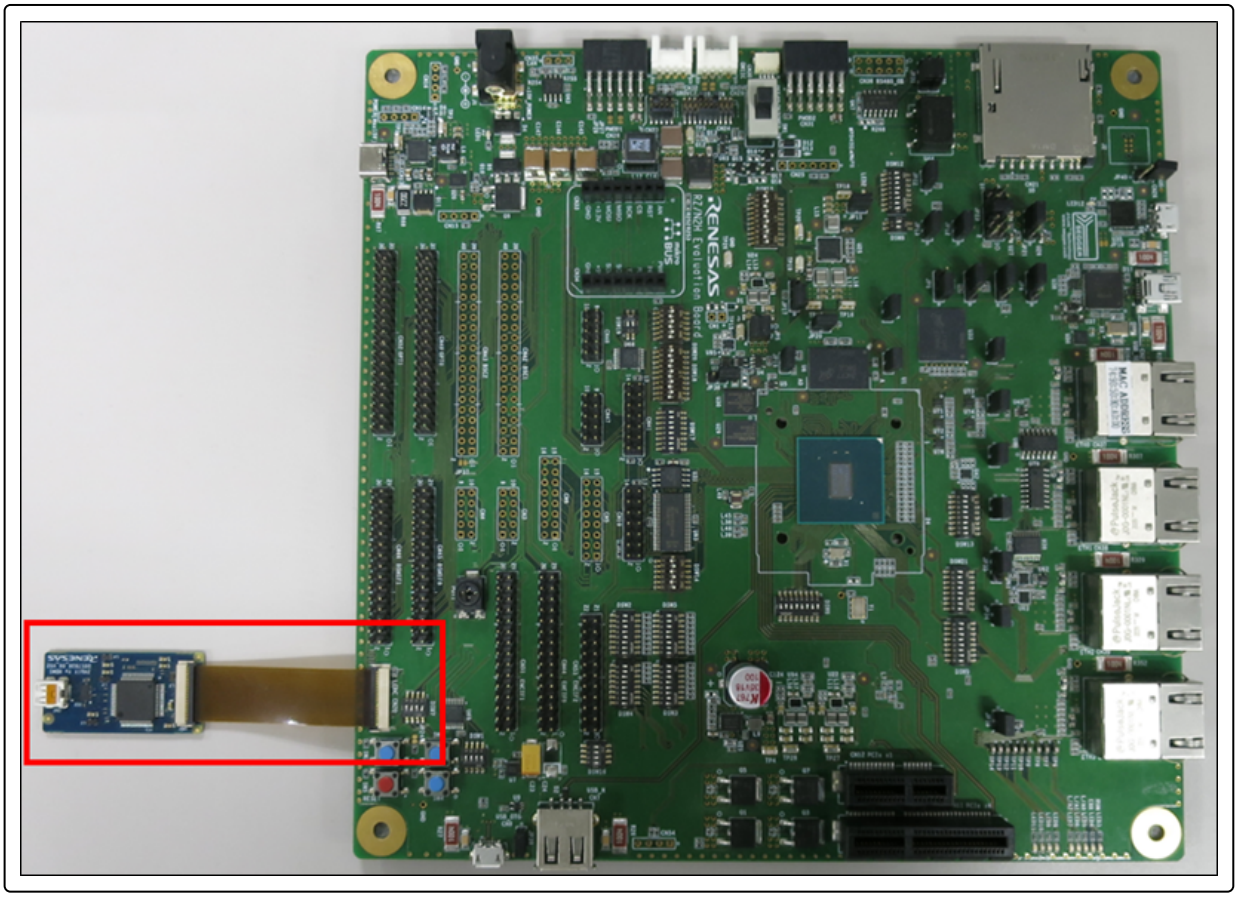


Figure 4-20: Connection of the display output module and Evaluation Board

How to configure DIP Switch

Change the DIP switches as the following table.

DSW5-3	ON
DSW7-1	ON
DSW7-2	OFF
DSW7-3	ON
DSW7-4	OFF

DSW12-3	ON
DSW12-4	OFF
DSW15-8	ON
DSW15-9	OFF
DSW15-10	OFF

DSW18-5	OFF
DSW18-6	ON
DSW18-9	OFF
DSW18-10	ON

4.5 Startup Procedure of RZ/T2H and RZ/N2H

4.5.1 Power Supply

1. Connect USB-PD Power Charger to USB Type-C Connector (RZ/T2H: CN46, RZ/N2H: CN13).
2. LED lights up. (RZ/T2H: LED12, RZ/N2H: LED1)
3. Turn ON the power switch to supply the power. (RZ/T2H: SW16, RZ/N2H: SW1)

4. LED lights up. (RZ/T2H: LED11, RZ/N2H: LED2)

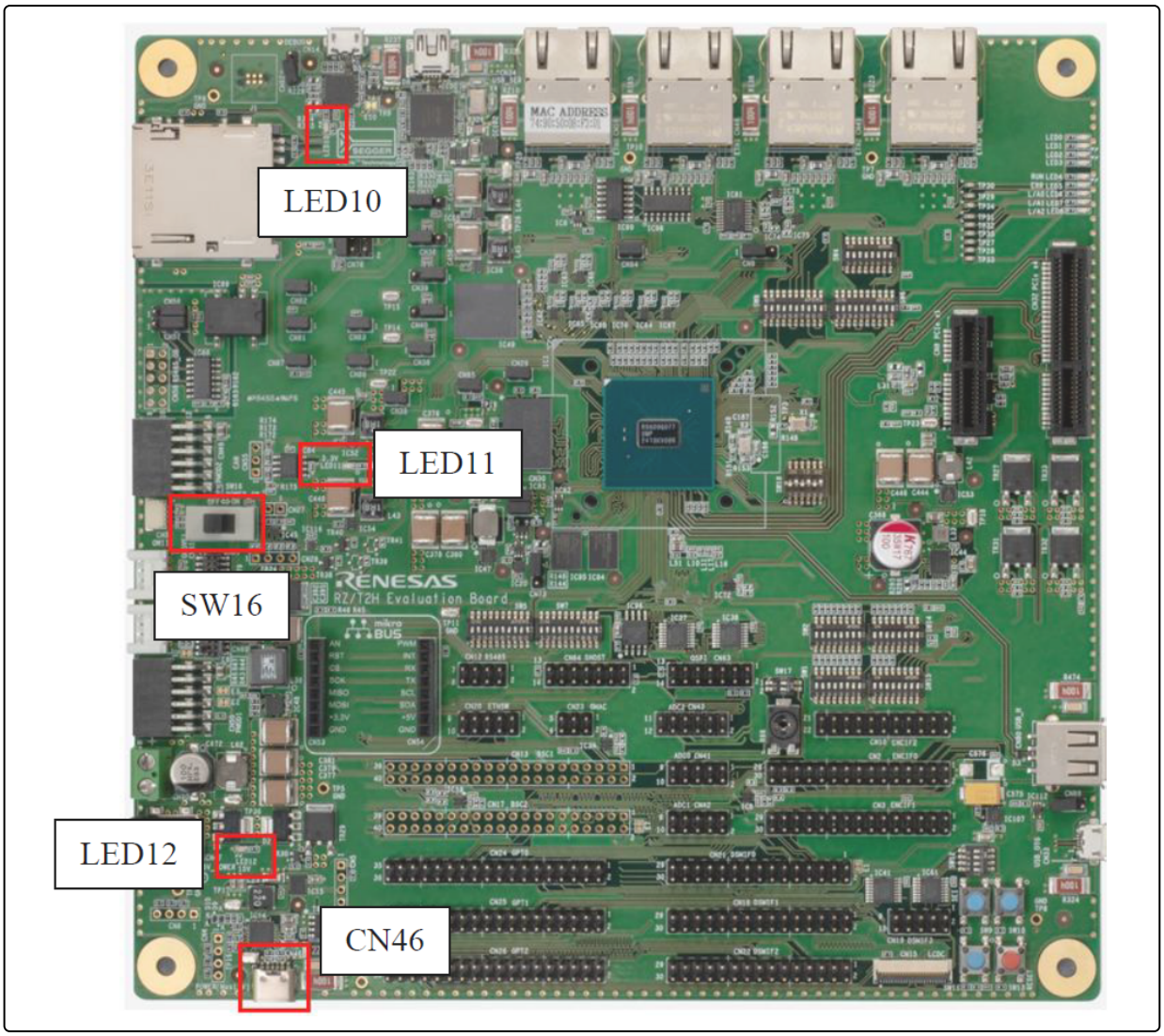


Figure 4-21: RZ/T2H Board example

4.5.2 Building files to write

The evaluation board uses the files in the below Table as the boot loaders. The boot loaders files are generated by the build instructions in Chapter 2. Once the build is finished, copy these files to a PC which runs serial terminal software.

Table 4-1 File names of Boot loader

Board	File name of Boot loader
RZ/T2H Evaluation Board	bl2_bp_xspi0-rzt2h-dev.srec fip-rzt2h-dev.srec
RZ/N2H Evaluation Board	bl2_bp_xspi0-rzn2h-dev.srec fip-rznW2h-dev.srec

4.5.3 Settings of Tera Term and evaluation board

Connect between the board and a control PC by USB serial cable.

Bring up the terminal software and select the "File" > "New Connection" to set the connection on the software.

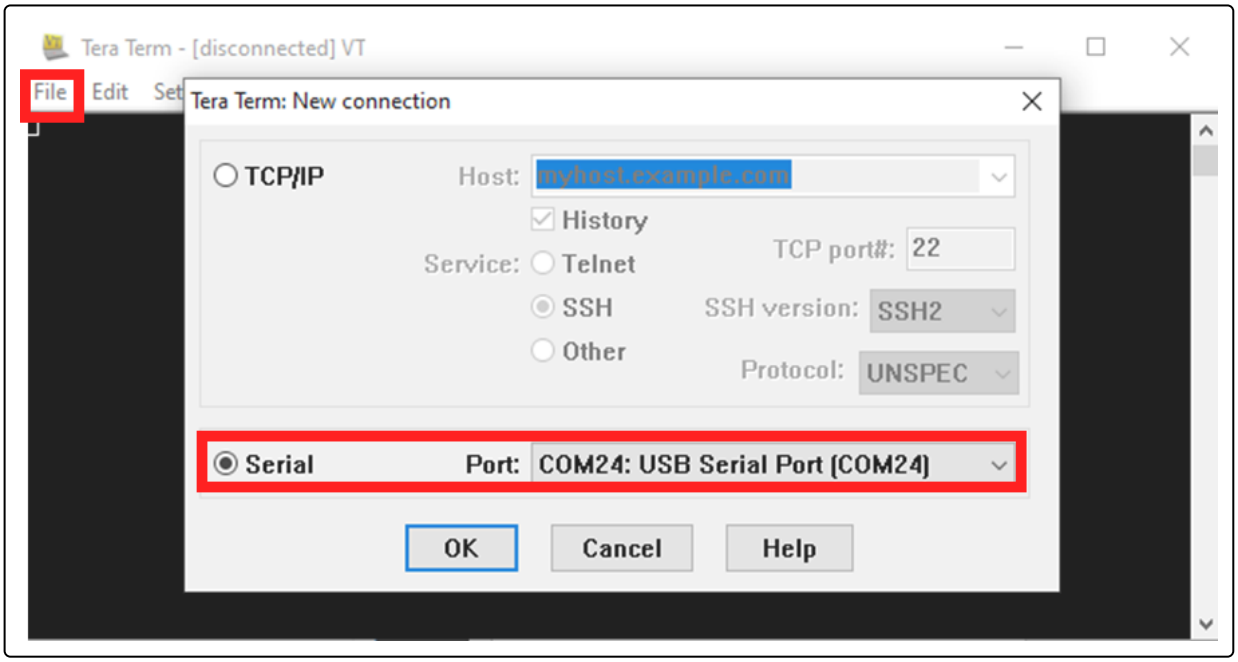


Figure 4-22: Set "Serial"

Set the serial port

Select the "Setup" > "Serial port" to set the settings about serial communication protocol on the software. Set the settings about serial communication protocol on a terminal software as below:

- Speed: 115200 bps
- Data: 8bit
- Parity: none
- Stop bit: 1 bit
- Flow control: none
- Trasmit delay: 0 msec/line

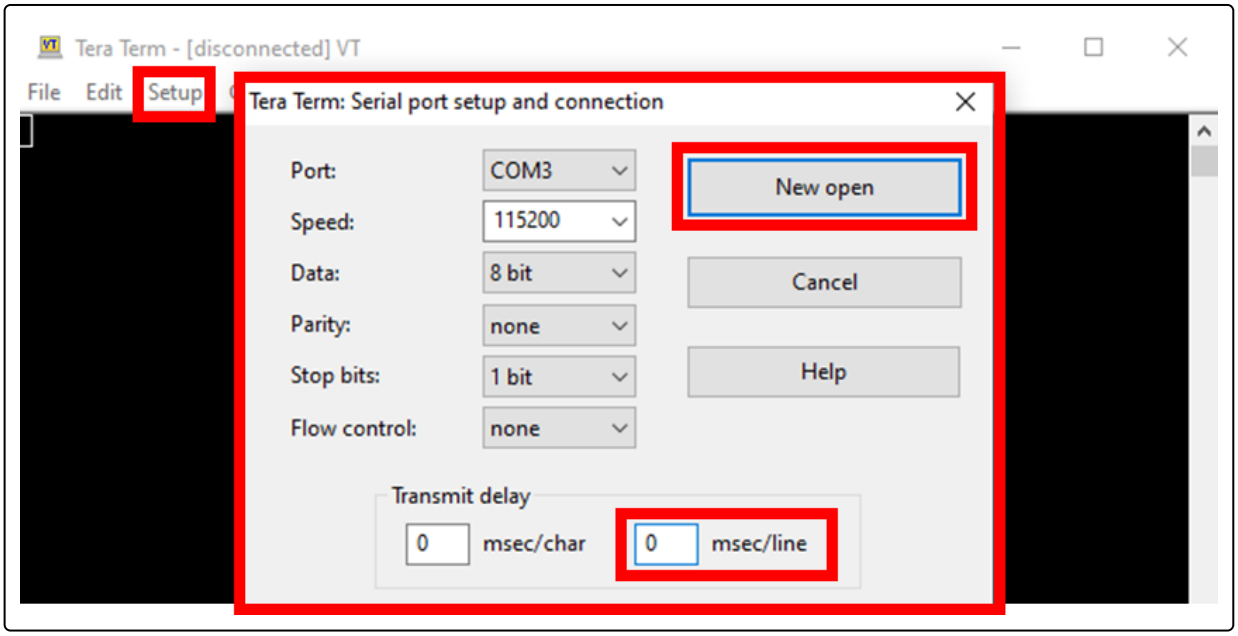
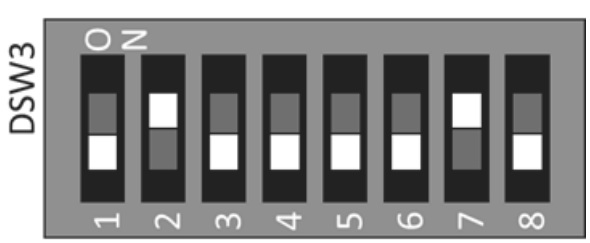


Figure 4-23: Terminal Setting

To set the board to SCIF Download mode, set the W14/DSW3 as below (please refer to previous section):

SW14-1	OFF *
SW14-2	ON
SW14-3	OFF *
SW14-4	OFF
SW14-5	OFF
SW14-6	ON *
SW14-7	ON
SW14-8	OFF

Figure 4-24: SCIF Download MODE(SW14) on RZ/T2H



DSW3-1	OFF *
DSW3-2	ON
DSW3-3	OFF *
DSW3-4	OFF
DSW3-5	OFF
DSW3-6	OFF
DSW3-7	ON
DSW3-8	OFF

Figure 4-25: SCIF Download MODE(DSW3) on RZ/N2H

After finishing all settings, turn on power (RZ/T2H: SW16, RZ/N2H: SW1), the messages below are displayed on the terminal.

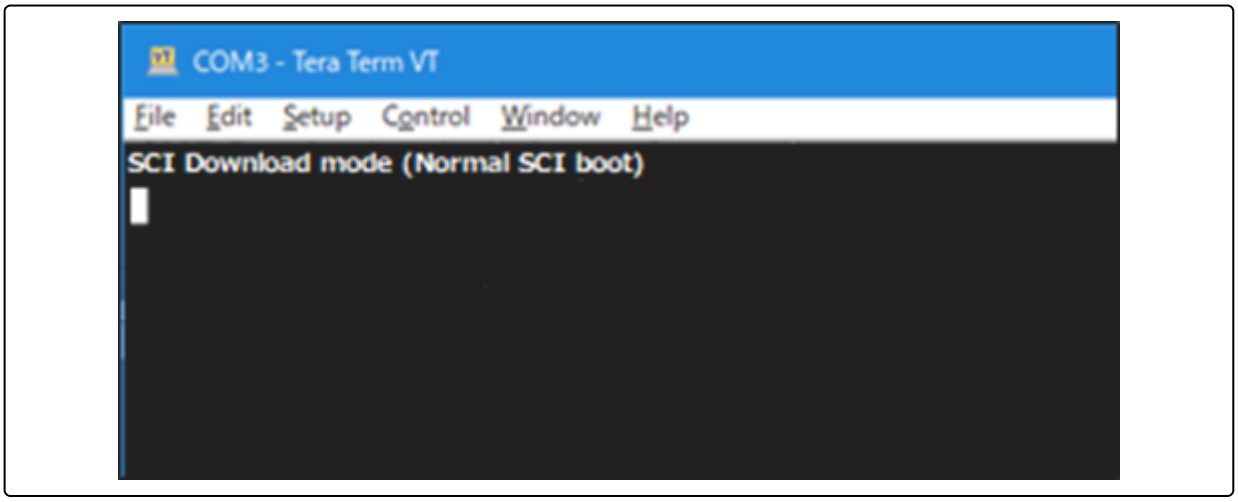


Figure 4-26: Terminal log of SCIF Download mode

4.6 Download Flash Programmer to RAM

Turn on the power switch of the board (RZ/T2H: SW16, RZ/N2H: SW1). The messages below are shown on the terminal.

```
SCI Download mode (Normal SCI boot)
```

Send 2 images of Flash Programmer using terminal software. Send "HDR NM" firstly, then send "Flash_Programmer_SCIF_CR52_RZT2H_EVK.mot" secondary.

Below is a sample procedure with Tera Term.

Drag "HDR NM" file and drop the file to the TeraTerm window, then click "OK".

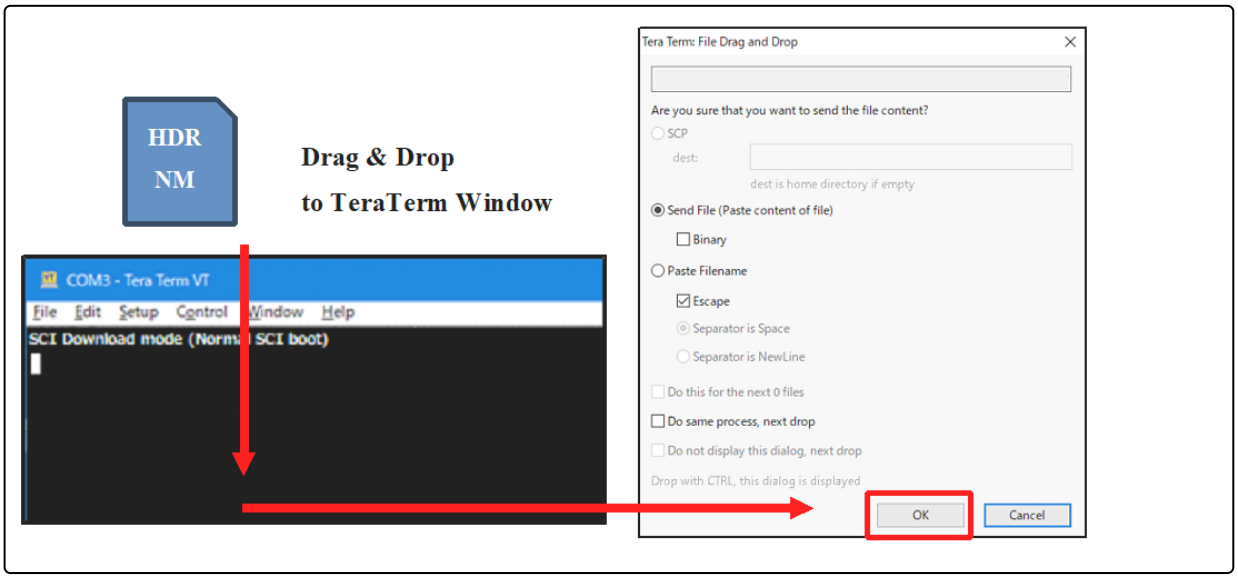


Figure 4-27: Send File

The image will be sent to the board via serial connection. After completing that, send "Flash_Programmer_SCIF_CR52_RZT2H_EVK.mot" as same as "HDR NM". Send it as soon as possible.

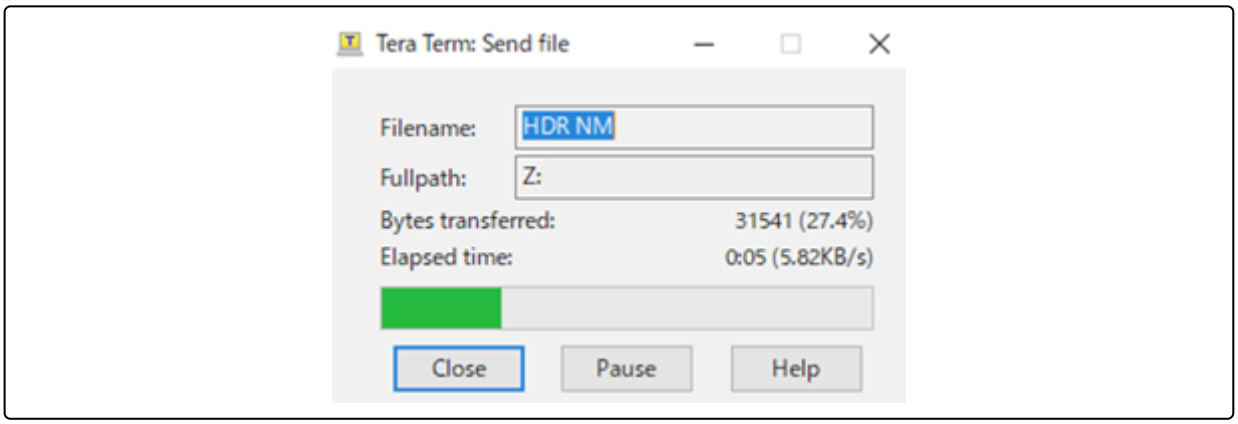


Figure 4-28: Sending File

When the transfer is complete, please transfer the file of "Flash_Programmer_SCIF_CR52_RZT2N_EVK.mot" using the same procedure within 30 seconds. If not completed within 10 seconds, please restart the transfer from the "HDR NM" file.

After successfully downloading the binary, Flash Programmer starts automatically and shows a message like below on the terminal.

```
SCI Download mode (Normal SCI boot)
-- Load Program to RAM ----------
-- Start Boot Program on RAM ---------

Flash Programmer V1.06 (RZT2H)(CR52)
```

4.7 Write the Bootloader

For the boot operation, two boot loader files need to be written to the target board. When updating the VLP version, make sure to replace the bootloader.

4.7.1 XSPIW command to write boot loader

"XSPIW" command of Flash Programmer is used to write boot loader binary files. This command receives binary data from the serial port and writes the data to a specified address of the Flash ROM with information where the data should be loaded on the address of the main memory.

```
$ XSPIW 0x0 0x0 0x0
send file
```

4.7.2 Send BL2 file

When the "send file" prompt appears, transmit the "bl2_bp_xspi0-xxx.srec" file using the terminal software.

Once the binary is successfully downloaded, messages similar to the following will be displayed on the terminal

```
Erased
Writen
```

4.7.3 XSPIW command to write boot loader again

Run the following command to write another boot loader file.

```
$ XSPIW 0x0 0x060000 0x0
send file
```

4.7.4 Send FIP file

When the "send file" prompt appears, transmit the "fip-xxx.srec" file using the terminal software.

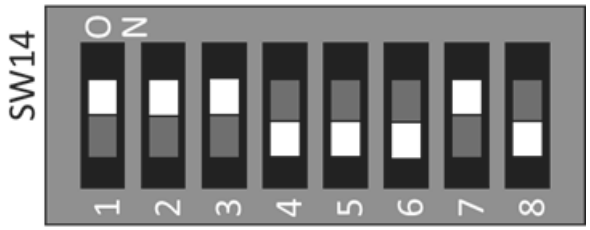
Once the binary is successfully downloaded, messages similar to the following will be displayed on the terminal

```
Erased
Writen
```

After writing two loader files normally, turn off the power switch of the board (RZ/T2H: SW16, RZ/N2H: SW1).

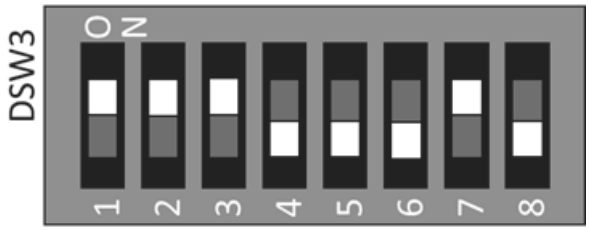
4.8 Change Back to Normal Boot Mode

To set the board to SPI Boot mode, set the RZ/T2H: SW16, RZ/N2H: DSW3 as below:



SW14-1	ON *
SW14-2	ON
SW14-3	ON *
SW14-4	OFF
SW14-5	OFF
SW14-6	OFF *
SW14-7	ON
SW14-8	OFF

Figure 4-29: xSPI Boot MODE(SW14)



DSW3-1	ON *
DSW3-2	ON
DSW3-3	ON *
DSW3-4	OFF
DSW3-5	OFF
DSW3-6	OFF
DSW3-7	ON
DSW3-8	OFF

Figure 4-30: xSPI Boot MODE(DSW3)

Turn on the power switch of the board.

```
U-Boot 2024.07 (Jul 01 2024 - 18:07:18 +0000)

CPU: Renesas Electronics RZ/T2H
Model: Renesas Development EVK based on r9a07g076
DRAM:  960 MiB (effective 1.9 GiB)
Core:  34 devices, 17 uclasses, devicetree: separate
MMC:   mmc@92080000: 0, mmc@92090000: 1
Loading Environment from MMC... Reading from MMC(0)... *** Warning - bad CRC, using default environment

In:    serial@80005400
Out:   serial@80005400
Err:   serial@80005400
Net:
Warning: ethernet@92000000 (eth0) using random MAC address - da:17:6e:81:2f:2a
eth0: ethernet@92000000Get shared mii bus on ethernet@92010000

Warning: ethernet@92010000 (eth1) using random MAC address - b2:6d:db:1d:05:10
, eth1: ethernet@92010000
Hit any key to stop autoboot:  0

=>
```

Following the messages above, many warning messages will be shown. These warnings are eliminated by setting correct environment variables. Please set default value and save them to the Flash ROM.

```
=> env default -a
## Resetting to default environment
=> saveenv
Saving Environment to MMC... Writing to MMC(0)...OK
=>
```

Now, the evaluation board is ready to boot Linux.

5. Booting and Running Linux

This chapter explains how to boot Linux on the evaluation board.

The explanation is based on RZ/T2H as an example.

5.1 Set micro-SD card

Set micro-SD card to slot CN16 on the back of the evaluation board.

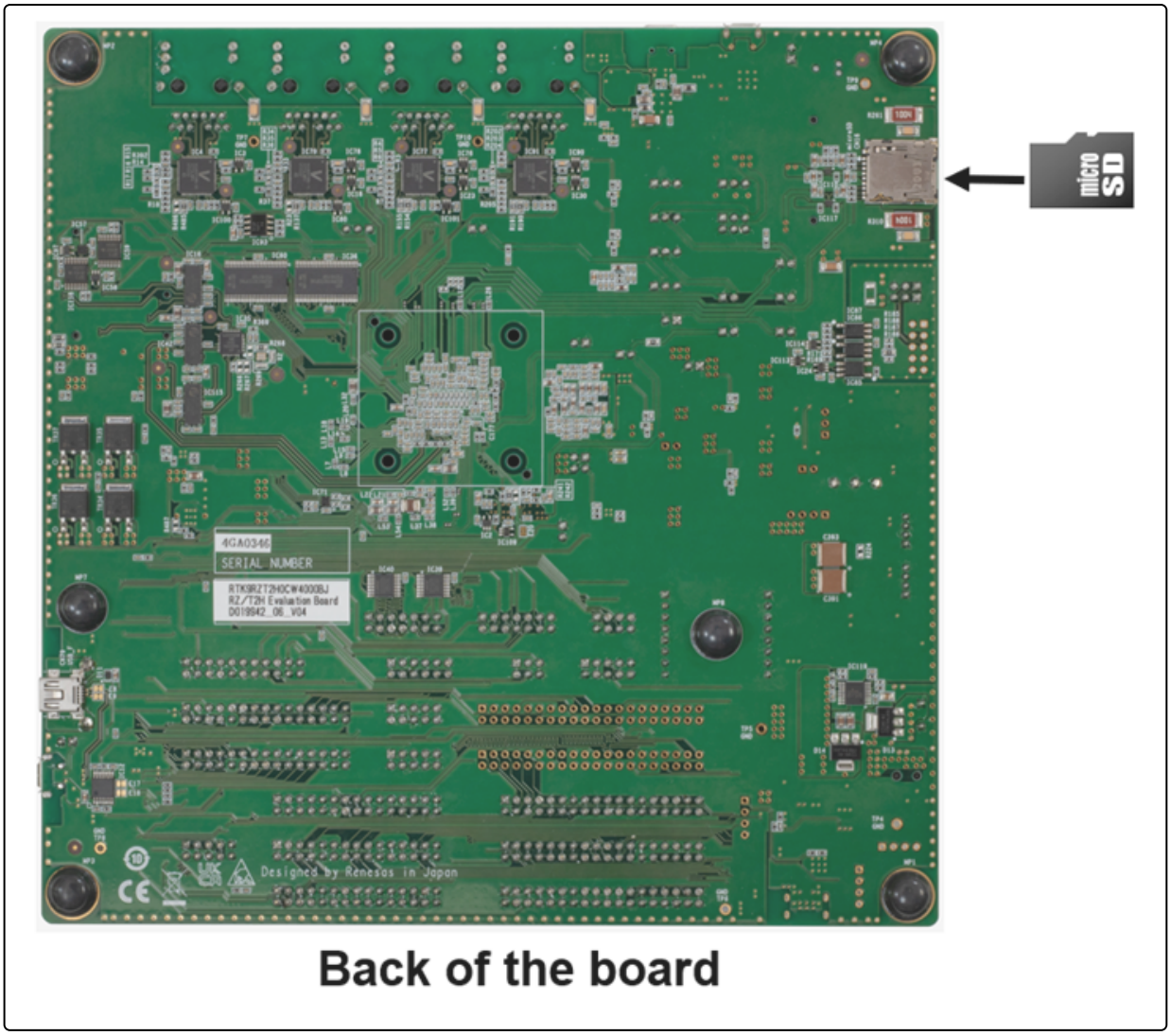


Figure 5-1: Set micro-SD card to the evaluation board

Now the board can boot up normally. Please turn off and on the power again to boot up the board.

5.2 Power on the board and Startup Linux

After obtaining your reference board, please be sure to follow the document and write the bootloaders to the Flash ROM before starting the evaluation.

Before booting the board, please be sure to confirm the bootloaders which are built with your VLP are written to your board.

```
U-Boot 2024.07 (Jul 01 2024 - 18:07:18 +0000)

CPU: Renesas Electronics RZ/T2H
Model: Renesas Development EVK based on r9a09g077
DRAM:  960 MiB (effective 1.9 GiB)
Core:  34 devices, 17 uclasses, devicetree: separate
MMC:   mmc@92080000: 0, mmc@92090000: 1
Loading Environment from MMC... Reading from MMC(0)... OK
In:    serial@80005400
Out:   serial@80005400
Err:   serial@80005400
Net:
Warning: ethernet@92000000 (eth0) using random MAC address - 76:f7:28:49:00:be
eth0: ethernet@92000000Get shared mii bus on ethernet@92010000

Warning: ethernet@92010000 (eth1) using random MAC address - 5a:4f:46:f5:c6:89
, eth1: ethernet@92010000
Hit any key to stop autoboot:  0
switch to partitions #0, OK
mmc1 is current device
46602752 bytes read in 1846 ms (24.1 MiB/s)
29043 bytes read in 3 ms (9.2 MiB/s)
## Flattened Device Tree blob at c7000000
   Booting using the fdt blob at 0xc7000000
Working FDT set to c7000000
   Loading Device Tree to 00000000cfff5000, end 00000000cffff172 ... OK
Working FDT set to cfff5000
Overlap found: 0x240000000..0x27fffffff / 0x240000000..0x2fffffffff

Starting kernel ...

[    0.000000] Booting Linux on physical CPU 0x0000000000 [0x412fd050]
[    0.000000] Linux version 6.12.9-yocto-standard
:
:
Poky (Yocto Project Reference Distro) 5.0.8 rzt2h-dev ttySC1
rzt2h-dev login:root
WARNING: Poky is a reference Yocto Project distribution that should be used for
testing and development purposes only. It is recommended that you create your
own distribution for production use.
root@rzt2h-dev:~#
```

NOTE:

As a reference, this log is written with the VLP v5.0.0.

5.3 Shutdown the Board

To power down the system, follow the step below.

5.3.1 Run shutdown command

Run shutdown command on the console as below. After that, the shutdown sequence will start.

```
$ shutdown -h now
```

NOTE:

Run this command during the power-off sequence on rootfs.

5.3.2 Confirm that the power is off and turn off the power switch

After executing the shutdown command, you will see the "reboot: Power down" message. Then, turn off the "POWER" SW.

6. Building the SDK

To build Software Development Kit (SDK), run the commands below after the steps (1) – (3) of the Chapter 2.1 are finished.

The SDK allows you to build custom applications outside of the Yocto environment, even on a completely different PC. The results of the commands below are 'installer' that you will use to install the SDK on the same PC, or a completely different PC.

For building general applications:

```
$ cd ~/rz_vlp_v${PACKAGE_VERSION}/build

$ MACHINE=rz<board>-dev bitbake core-image-minimal -c populate_sdk
```

The resulting SDK installer will be in "build/tmp/deploy/sdk/"

The SDK installer will have the extension .sh

To run the installer, you would execute the following command:

```
$ sudo sh ~/rz_vlp_v${PACKAGE_VERSION}/build/tmp/deploy/sdk/rz-vlp-glibc-x86_64-\

core-image-minimal-cortexa55-rz<board>-dev-toolchain-<version>.sh
```

7. Application Building and Running

This chapter explains how to make and run an application with this package.

The explanation is based on RZ/T2H as an example.

7.1 Make an application

Here is an example of how to make an application running on VLP. The following steps will generate the "Hello World" sample application.

NOTE:

Must build (bitbake) a core image for the target and prepare SDK before making an application.

7.1.1 How to extract SDK

Install toolchain on a Host PC:

```
$ cd ~/rz_vlp_v${PACKAGE_VERSION}
$ sudo sh ./build/tmp/deploy/sdk/rz-vlp-glibc-x86_64-core-image-minimal-cortexa55\
-rzt2h-dev-toolchain-<version>.sh
```

NOTE:

sudo is optional in case user wants to extract SDK into a restricted directory (such as: /opt/).

If the installation is successful, the following messages will appear:

```
/mnt/host$ sudo sh ./build/tmp/deploy/sdk/rz-vlp-glibc-x86_64-core-image-minimal-cortexa55-rzt2h-dev-toolchain-x.x.x.sh
Poky (Yocto Project Reference Distro) SDK installer version x.x.xx
==========================================================================
Enter target directory for SDK (default: /opt/rz-vlp /x.x.x):
You are about to install the SDK to "/opt/rz-vlp/x.x.x". Proceed [Y/n]? Y
Extracting SDK.....................................................................
.....................................................done
Setting it up...done
SDK has been successfully set up and is ready to be used.
Each time you wish to use the SDK in a new shell session, you need to source the environment setup script e.g.
$ . /opt/rz-vlp/x.x.xx/environment-setup-cortexa55-poky-linux
```

Set up cross-compile environment:

```
$ source /<Location in which SDK is extracted>/environment-setup-cortexa55-poky-linux
```

NOTE:

User needs to run the above command once for each login session.

7.1.2 How to build Linux application

Make a work directory for the application on the Linux host PC.

```
$ mkdir ~/hello_apl
$ cd ~/hello_apl
```

Make the following two files (an application file and Makefile) in the directory for the application. Here, the application is made by automake and autoconf.

The code of file "main.c":

```
#include <stdio.h>
/* Display "Hello World" text on terminal software */
/* RZ/T2N Linux Start-up Guide */
/* R01US0780EJ0030 Rev.0.20 Page 23 of 27 */
/* Jul 24, 2024 */
int main(int argc, char** argv)
{
 printf("\nHello World\n");
 return 0;
}
```

The code of file "Makefile":

```
APP = linux-helloworld
SRC = main.c
all: $(APP)
CC ?= gcc
# Options for development
CFLAGS = -g -O2 -Wall -DDEBUG_LOG
$(APP):
<-tab->$(CC) -o $(APP) $(SRC) $(CFLAGS)
install:
<-tab->install -D -m755 $(APP) $(DESTDIR)/home/root/$(APP)
clean:
    rm -rf $(APP)
```

Make the application by the generated makefile.

```
$ make
```

```
$ make
aarch64-poky-linux-gcc  -mcpu=cortex-a55+crypto -mbranch-protection=standard -fstack-protector-strong  -O2 -
D_FORTIFY_SOURCE=2 -Wformat -Wformat-security -Werror=format-security --sysroot=/opt/rz-vlp/x.x.xx/sysroots/cortexa55-poky-
linux -o linux-helloworld main.c -g -O2 -Wall -DDEBUG_LOG

$ ls -l
total 36
-rwxrwxr-x 1 renesas renesas 15320 Mar 28 09:28 linux-helloworld
-rw-rw-r-- 1 renesas renesas   148 Mar 28 09:28 main.c
-rw-rw-r-- 1 renesas renesas   246 Mar 28 09:28 Makefile
```

After making, confirm that the execute application (the sample file name is "linux-helloworld") is generated in the hello_apl folder. Also, this application must be cross compiled.

The sample application could be written by the following procedure. The application should be stored in the ext3 partition.

```
$ sudo mount /dev/sdb2 /media/
$ cd /media/usr/bin
$ sudo cp ~/hello_apl/linux-helloworld .
$ sudo chmod +x linux-helloworld
```

NOTE:

1, "sdb2" may depend on using system. 2, "hello_apl" is an optional directory name to store the application.

7.2 Run a sample application

Power on the evaluation board kit and start the system. After booting, run the sample application with the following command.

```
rzt2h-dev login:root
root@rzt2h-dev:~# /usr/bin/linux-helloworld

Hello World
root@rzt2h-dev:~#
```

Appendix

A Ethernet port

A.1 Check the Ethernet port on the evaluation board and in Linux

Please note that the Ethernet ports on the evaluation board are assigned in Linux as shown in the table below.

Table Appendix-1 Port name on Linux

Ethernet Port on the evaluation board	Channel	Interface on Linux
ETH0	ETHSW0	lan0
ETH1	ETHSW1	lan1
ETH2	ETHSW2	eth1
ETH3	GMAC1	eth0

If you want to check it on Linux, run the ifconfig command after booting Linux. The following is an example.

A.2 Enable ETH0 (lan0) and ETH1 (lan1) on Linux

"ETH0" and "ETH1" on the evaluation board are enabled by the build steps in Chapter 2.3. To use them on Linux, you need to enable the network interfaces and obtain an IP address.

The following commands provide examples of how to enable interfaces and acquire an IP address.

```
$ ifconfig lan0 up
$ udhcpc -i lan0
$ ifconfig lan1 up
$ udhcpc -i lan1
```

"ETH2" and "ETH3" on the evaluation board are enabled on Linux as "eth1 (=ETH2)" and "eth0(=ETH3)" by default.

NOTE:

Running "udhcpc" command on one network interface causes other unrelated interfaces to enter the link-down state. To reuse the previously affected interface, they must be brought down once (e.g., ifconfig xxx down).

A.3 Check Ethernet port on Linux

To check the ethernet port, run the following command as a reference.

```
$ ping -I eth1 <IP Address>
```

To check the performance of the ethernet port, iperf3 command can be used for one of the checking tools. To install the command, add the following line to "~/rz_vlp_v${PACKAGE_VERSION}/build/conf/local.conf".

```
IMAGE_INSTALL:append = " iperf3 "
```

B Docker

This section explains how to build the VLP with Docker enabled and how to obtain and run the "hello-world" image.

B.1 Add layers to enable Docker

After completing Chapter 2.2, run the following commands.

```
$ bitbake-layers add-layer ../meta-openembedded/meta-networking
$ bitbake-layers add-layer ../meta-openembedded/meta-filesystems
$ bitbake-layers add-layer ../meta-virtualization
```

Modify the below line in the "~/rz_vlp_v${PACKAGE_VERSION}/build/conf/local.conf" to enable the Docker feature.

```
 # Un-comment a following line to enable the Docker Engine.
- # DISTRO_FEATURES:append = " virtualization docker"
+ DISTRO_FEATURES:append = " virtualization docker"
```

Once executed, proceed until Chapter 2.2 last step to build the VLP with Docker enabled.

B.2 Boot the evaluation board

Follow Chapters 4 and 5, boot the evaluation board, and proceed to the next step.

Please rewrite the U-boot Environment Variable as below commands:

For T2H

```
=> setenv bootimage 'booti 0xc4200000 - 0xC7000000'
=> setenv bootcmd 'run bootcmd_check;run bootimage'
=> setenv emmcload 'ext4load mmc 0:2 0xc4200000 boot/Image;ext4load mmc 0:2 0xC7000000 boot/r9a09g077m44-dev.dtb;run
prodemmcbootargs'
=> setenv sd1load 'ext4load mmc 1:2 0xc4200000 boot/Image;ext4load mmc 1:2 0xC7000000 boot/r9a09g077m44-dev.dtb;run
prodsdbootargs'
=> setenv bootimage 'booti 0xc4200000 - 0xC7000000'
=> saveenv
```

For N2H

```
=> setenv bootimage 'booti 0xc4200000 - 0xC7000000'
=> setenv bootcmd 'run bootcmd_check; run bootimage'
=> setenv emmcload 'ext4load mmc 0:2 0xc4200000 boot/Image; ext4load mmc 0:2 0xC7000000 boot/r9a09g087m44-dev.dtb; run
prodemmcbootargs'
=> setenv sd1load 'ext4load mmc 1:2 0xc4200000 boot/Image;ext4load mmc 1:2 0xC7000000 boot/r9a09g087m44-dev.dtb;run
prodsdbootargs'
=> bootimage 'booti 0xc4200000 - 0xC7000000'
=> saveenv
```

NOTE:

U-boot's command-line parser does not support line continuation using backslashes () like a typical Linux shell. All commands must be written as a single, complete line.

B.3 Hello world

Run the commands below to get the docker image of hello-world and run the image.

```
$ docker pull hello-world
$ docker run hello-world
```

```
$ docker run hello-world
... skip ...
Hello from Docker!
This message shows that your installation appears to be working correctly.
... skip ...
```

C How to boot from eMMC

This section explains how to boot Linux from eMMC.

C.1 Writing Bootloader for eMMC Boot

For the boot operation, EXT_CSD register of eMMC needs to be modified and two boot loader files need to be written to the target board. Modifying register address and value information are described in the following table.

After booting the Flash Programmer (Refer to Chapter 4 "Download Flash Programmer to RAM").

Then, "EMMCW" command of Flash Programmer is used to write boot loader binary files. This command receives binary data from the serial port and writes the data to a specified address of the eMMC with information where the data should be loaded on the address of the main memory

Table Appendix-2 Address of EXT_CSD register of eMMC for eMMC boot

Address	Value to write
0xB1	0x02
0xB3	0x08

Run "EMMCWECSD" and "EMMCW" commands as below:

```
USB Download mode (Normal USB boot)
-- Load Program to RAM ----------
-- Start Boot Program on RAM ---------
Flash Programmer V1.06 (RZT2N)(CR52)
> EMMCWECSD 0xb3 0x9
EXT_CSD[179] = 0x9
> EMMCW 0x1 0x0
send file
```

Send the data of "bl2_bp_emmc-xxx.srec" from terminal software after the message "send file" is shown.

After successfully downloading the binary, messages like below are shown on the terminal.

```
EMMCW Complete!
```

Next, write another loader file by using "EMMCW" command again.

```
> EMMCW 0x300 0x0
send file
```

Send the data of "fip-xxx.srec" from terminal software after the message "send file" is shown.

After successfully downloading the binary, messages like below are shown on the terminal.

```
EMMCW Complete!
```

Then modify EXT_CSD register by running "EMMCWECSD" and "EMMCW" commands.

```
> EMMCWECSD 0xb3 0x8
EXT_CSD[179] = 0x8
> EMMCWECSD 0xb1 0x2
EXT_CSD[177] = 0x2
>
```

After writing two loader files normally, turn off the power switch of the board (RZ/T2H: SW16, RZ/N2H: SW1).

C.2 Create a microSD card to boot Linux for eMMC boot

Create a microSD card by Chapter 3 "Preparing the SD Card".

C.3 Set eMMC Boot mode

Please Set the SW14 (RZ/T2H) and DSW3 (RZ/N2H) settings as follows. This mode is used to boot Linux.

Table Appendix-3 SW14 settings on RZ/T2H

SW14-1	ON
SW14-2	ON
SW14-3	OFF
SW14-4	OFF
SW14-5	OFF
SW14-6	ON
SW14-7	ON
SW14-8	OFF

Table Appendix-4 DSW3 settings on RZ/N2H

DSW3-1	ON
DSW3-2	ON
DSW3-3	OFF
DSW3-4	OFF

DSW3-5	OFF
DSW3-6	ON
DSW3-7	ON
DSW3-8	OFF

Set CN78 (RZ/T2H) and JP23 (RZ/N2H) as follows.

Figure Appendix-1: CN78 Setting on RZ/T2H

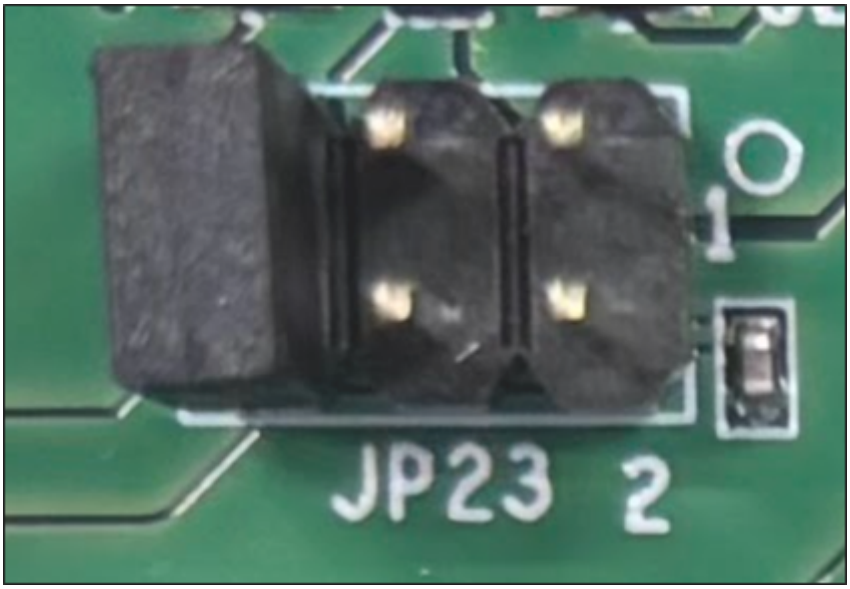


Figure Appendix-2: JP23 Setting on RZ/N2H

C.4 Set U-boot for eMMC boot

Reset the board and interrupt the boot process by pressing the Enter key.

```
U-Boot 2024.07 (Jul 01 2024 - 18:07:18 +0000)
CPU: Renesas Electronics RZ/T2H
Model: Renesas Development EVK based on r9a09g077
DRAM: 960 MiB (effective 1.9 GiB)
Core: 18 devices, 11 uclasses, devicetree: separate
MMC: mmc@92080000: 0, mmc@92090000: 1
Loading Environment from MMC... Reading from MMC(0)....OK
*** Warning - No block device, using default environment
In: serial@80005400
Out: serial@80005400
Err: serial@80005400
Net: No ethernet found.
Hit any key to stop autoboot: 0
=>
```

D How to boot from eSD

This section explains how to boot Linux from eSD mode.

D.1 Create a microSD card to boot Linux for eSD boot

Create a microSD card by following Chapter 3.

D.2 Setting the U-boot and boot with eSD boot mode

Please Set the SW14 (RZ/T2H) and DSW3 (RZ/N2H) settings as follows. This mode is used to boot Linux.

Table Appendix-5 SW14 settings on RZ/T2H

SW14-1	OFF
SW14-2	OFF
SW14-3	ON
SW14-4	OFF
SW14-5	OFF
SW14-6	OFF
SW14-7	ON
SW14-8	OFF

Table Appendix-6 DSW3 settings on RZ/N2H

DSW3-1	OFF
DSW3-2	OFF
DSW3-3	ON
DSW3-4	OFF
DSW3-5	OFF

DSW3-6	OFF
DSW3-7	ON
DSW3-8	OFF

Then, turn on the power of the board. After booting Linux, please login as root.

E How to change the U-boot Environment Variable Storage Location

U-boot Environment Variable can be stored in eMMC, eSD and SPI Flash. This section explains how to change the UBoot Environment Variable Storage Location.

The explanation is based on RZ/T2H as an example.

E.1 Store in eMMC

The default setting is storing boot environment variables stored to eMMC. No need to change anything.

When storing boot environment variables stored to eMMC, the log should be as below:

```
U-Boot 2024.07 (Jul 23 2025 - 12:51:51 +0000)
CPU: Renesas Electronics RZ/T2H
Model: Renesas Development EVK based on r9a09g077
DRAM: 960 MiB (effective 7.9 GiB)
Core: 75 devices, 21 uclasses, devicetree: separate
MMC: mmc@92080000: 0, mmc@92090000: 1
Loading Environment from MMC... Reading from MMC(0)... OK
In: serial@80005000
Out: serial@80005000
Err: serial@80005000
Net:
Warning: ethernet@92010000 (eth1) using random MAC address - 8a:60:42:8a:01:12
RZ/T2N Evaluation Board Linux Start-up Guide
R01US0780EJ0040 Rev. 0.40 Page 37 of 40
Sep. 30, 2025
Warning: ethernet@92000000 (eth0) using random MAC address - 82:92:15:39:03:14
eth0: ethernet@92000000, eth1: ethernet@92010000
Hit any key to stop autoboot: 0
=>
=> saveenv
Saving Environment to MMC... Writing to MMC(0)... OK
```

E.2 Store in eSD

Run the following commands to Open the following config file.

```
$ TEMPLATECONF=$PWD/meta-renesas/meta-rz-distro/conf/templates/vlp-v5-conf/ source poky/oe-init-build-env build

$ MACHINE=rzt2h-dev bitbake u-boot -c devshell
$ vi configs/rzt2h-dev_defconfig
```

Modify the config file as described below to change boot environment variables stored position from eMMC (default setting) to eSD.

```
- CONFIG_ENV_OFFSET=0x3FFFE0000
+ CONFIG_ENV_OFFSET=0x200000
...
...
- CONFIG_SYS_MMC_ENV_DEV=0
- CONFIG_SYS_MMC_ENV_PART=2
+ CONFIG_SYS_MMC_ENV_DEV=1
+ CONFIG_SYS_MMC_ENV_PART=0
```

Then, you can use the following command to re-build the U-Boot.

```
$ MACHINE=rzt2h-dev bitbake u-boot -C configure
```

When storing boot environment variables stored to eSD, the log should be as below:

```
U-Boot 2024.07 (Jul 23 2025 - 12:51:51 +0000)
CPU: Renesas Electronics RZ/T2H
Model: Renesas Development EVK based on r9a09g077
DRAM: 960 MiB (effective 7.9 GiB)
Core: 75 devices, 21 uclasses, devicetree: separate
MMC: mmc@92080000: 0, mmc@92090000: 1
Loading Environment from MMC... Reading from MMC(1)... OK
In: serial@80005000
Out: serial@80005000
Err: serial@80005000
Net:
Warning: ethernet@92010000 (eth1) using random MAC address - 8a:60:42:8a:01:12
Warning: ethernet@92000000 (eth0) using random MAC address - 82:92:15:39:03:14
eth0: ethernet@92000000, eth1: ethernet@92010000
Hit any key to stop autoboot: 0
=>
=> saveenv
Saving Environment to MMC... Writing to MMC(1)... OK
```

E.3 Store in SPI Flash

Run the following commands to Open the following config file.

```
$ TEMPLATECONF=$PWD/meta-renesas/meta-rz-distro/conf/templates/vlp-v5-conf/ source poky/oe-init-build-env build

$ MACHINE=rzt2h-dev bitbake u-boot -c devshell
$ vi configs/rzt2h-dev_defconfig
```

Modify the config file as described below to change boot environment variables stored position from eMMC (default setting) to SPI Flash.

```
- CONFIG_ENV_OFFSET=0x3FFFE0000
+ CONFIG_ENV_OFFSET=0x001E0000
...
...
- CONFIG_ENV_IS_IN_MMC=y
+ CONFIG_ENV_IS_IN_SPI_FLASH=y
+ CONFIG_ENV_SECT_SIZE=0x20000
```

Run the following commands to open the following c file. (T2H and N2H are using same c file.)

```
$ TEMPLATECONF=$PWD/meta-renesas/meta-rz-distro/conf/templates/vlp-v5-conf/ source poky/oe-init-build-env build
$ MACHINE=rzt2h-dev bitbake u-boot -c devshell
$ vi plat/renesas/rz/soc/t2h/plat_security.c
```

Modify the c file as described below.

```
- .nsaid_permissions = PLAT_TZC_REGION_ACCESS_S_PRIV
+ .nsaid_permissions = PLAT_TZC_REGION_ACCESS_NS_PRIV
```

Then, you can use the following command to re-build the U-Boot.

```
$ MACHINE=rzt2h-dev bitbake u-boot -C configure
```

When storing boot environment variables stored to eSD, the log should be as below:

```
U-Boot 2024.07 (Jul 23 2025 - 12:51:51 +0000)
CPU: Renesas Electronics RZ/T2H
Model: Renesas Development EVK based on r9a09g077
DRAM: 960 MiB (effective 7.9 GiB)
Core: 75 devices, 21 uclasses, devicetree: separate
MMC: mmc@92080000: 0, mmc@92090000: 1
Loading Environment from SPIFlash... SF: Detected mx25lw51245g with page size 256 Byte
s, erase size 4 KiB, total 64 MiB
In: serial@80005000
Out: serial@80005000
Err: serial@80005000
Net:
Warning: ethernet@92010000 (eth1) using random MAC address - 8a:60:42:8a:01:12
Warning: ethernet@92000000 (eth0) using random MAC address - 82:92:15:39:03:14
eth0: ethernet@92000000, eth1: ethernet@92010000
Hit any key to stop autoboot: 0
=>
=> saveenv
Saving Environment to SPIFlash... Erasing SPI flash...Writing to SPI flash...done
```

F GUI Application Framework (LVGL)

This section explains how to enable the GUI application framework. LVGL is option for the GUI application framework, enabling you to display a graphical user interface on the monitor. The steps outlined in this section are optional, so skip this section if it is not required for your purposes.

F.1 Build Initialize

Initialize a build using the 'oe-init-build-env' script in Poky and point TEMPLATECONF to platform conf path.

```
$ cd ~/rz_vlp_v${PACKAGE_VERSION}/build
$ TEMPLATECONF=$PWD/meta-renesas/meta-rz-distro/conf/templates/vlp-v5-conf/ source poky/oe-init-build-env build
```

Run the following commands to apply a patch.

```
$ cd ~/rz_vlp_v${PACKAGE_VERSION}/meta-renesas
$ patch -p1 < ../extra/0001-rz-distro-lvgl-Use-lv_tick_set_cb-for-LVGL-timing.patch
$ cd ..
```

F.2 Add the LVGL to the local.conf

Add below line to "~/rz_vlp_v${PACKAGE_VERSION}/build/conf/local.conf" to the LVGL demo.

```
IMAGE_INSTALL:append = " lvgl-demo-fb "
LVGL_CONFIG_DRM_CARD = "/dev/dri/card0"
```

F.3 Edit device tree source for enabling LCDC

Run the following command and launch the development shell. A new terminal is opened, and you are placed in the source directory of kernel.

```
$ MACHINE=<board> bitbake linux-renesas -c devshell
```

In the new terminal, modify the device tree source as below.

For RZ/T2H:

```
$ vi ./arch/arm64/boot/dts/renesas/r9a09g077m44-dev.dts
```

For RZ/N2H:

```
$ vi ./arch/arm64/boot/dts/renesas/r9a09g087m44-dev.dts
```

And change the following setting from the default. The setting is the same for both RZ/T2H and RZ/N2H.

```
#define SEL_ETHSW_LCDC 1
```

Then exit from the terminal with the following command.

```
$ exit
```

F.4 Build images

Firstly, compile the kernel to apply the modification of the device tree source. And build the images.

```
$ MACHINE=<board> bitbake linux-renesas -C compile
$ MACHINE=<board> bitbake bitbake core-image-minimal
```

F.5 Prepare the SD card and the board environment

Follow section 3 and section 4 to prepare the board environment and the micro-SD card.

F.6 Boot and Run Linux

Refer to section 5 and power on your evaluation board.

F.7 Run the LVGL demo

Run the below commands to launch the LVGL demo.

```
root@rzt2h-dev:~# lvgl
```

When you boot Linux with Weston, run the below command to stop Weston. When you launch the LVGL demo in the Weston environment, the demo performace is slow.

```
root@rzt2h-dev:~# systemctl stop weston@root
```

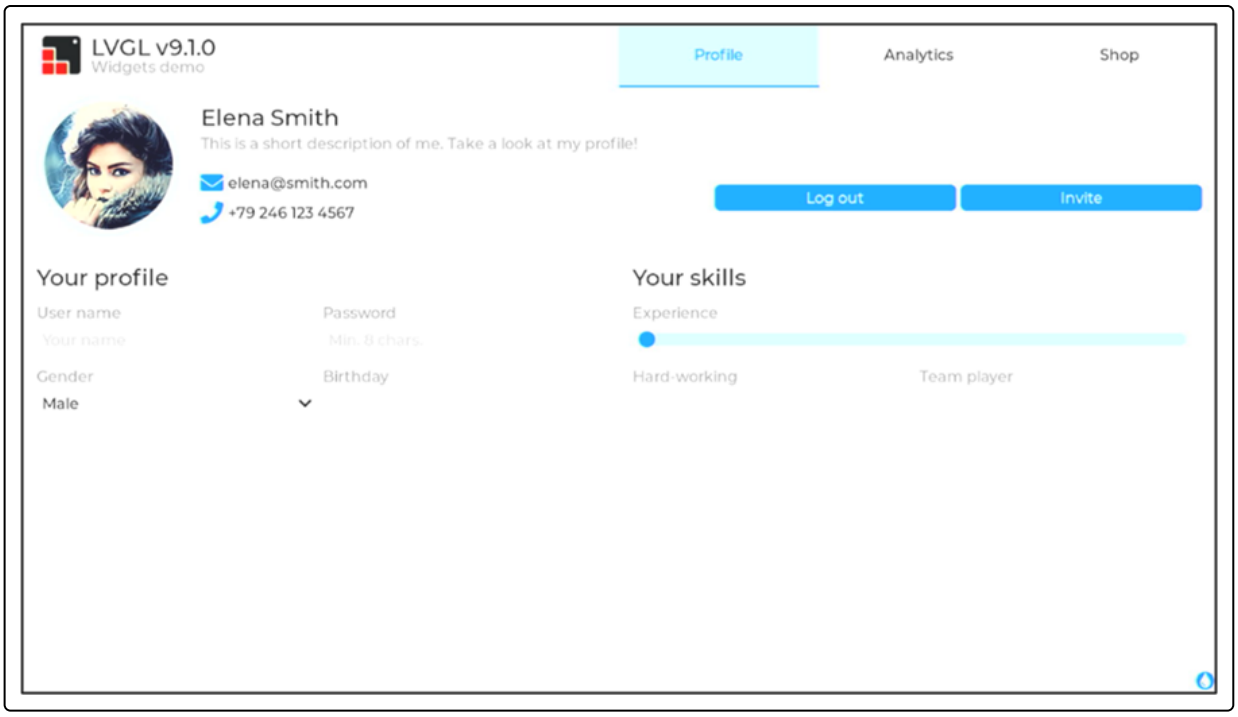


Figure Appendix-3: LVGL Demo

G Device drivers

The following drivers are supported: For detailed information on how to use them, please refer to these documents in the VLP manual set.

File	Description
RTK0EF0177Z90001ZJ-bsp-manuals-v5_x_x.zip	BSP Manual Set for RZ/T2H and RZ/N2H.

NOTE:

"x" is the version of the file. Please refer to the latest one.

Table Appendix-7 Support device drivers

Device Driver	Documents
Kernel Core	r01us0823ej0xxx-rz-t2h-n2h_Kernel_Core_UME.pdf
Direct Memory Access Controller (DMAC)	r01us0824ej0xxx-rz-t2h-n2h_DMAC_UME.pdf
GPIO	r01us0825ej0xxx-rz-t2h-n2h_GPIO_UME.pdf
Multi-function Timer Unit 3a (MTU3a)	r01us0826ej0xxx-rz-t2h-n2h_MTU3_UME.pdf
General PWM Timer (GPT)	r01us0827ej0xxx-rz-t2h-n2h_GPT_UME.pdf
Watchdog Timer (WDT)	r01us0828ej0xxx-rz-t2h-n2h_WDT_UME.pdf
Real Time Clock (RTC)	r01us0829ej0xxx-rz-t2h-n2h_RTC_UME.pdf
Ethernet Subsystem	r01us0830ej0xxx-rz-t2h-n2h_Ethernet_Subsystem_UME.pdf
USB 2.0 Host	r01us0831ej0xxx-rz-t2h-n2h_USB2.0_Host_UME.pdf
USB 2.0 Function	r01us0832ej0xxx-rz-t2h-n2h_USB2.0_Function_UME.pdf
Serial Communication Interface (SCI)	r01us0833ej0xxx-rz-t2h-n2h_SCI_UME.pdf
I2C Bus Interface (I2C)	r01us0834ej0xxx-rz-t2h-n2h_I2C_UME.pdf
SD/MMC Host Interface	r01us0835ej0xxx-rz-t2h-n2h_SD_MMC_UME.pdf
CAN-FD Interface (CANFD)	r01us0836ej0xxx-rz-t2h-n2h_CANFD_UME.pdf
Renesas Serial Peripheral Interface (RSPI)	r01us0837ej0xxx-rz-t2h-n2h_RSPI_UME.pdf
Serial Peripheral Interface (SPI)	r01us0838ej0xxx-rz-t2h-n2h_xSPI_UME.pdf
LCD Controller (LCDC)	r01us0839ej0xxx-rz-t2h-n2h_LCDC_UME.pdf
A/D Converter	r01us0840ej0xxx-rz-t2h-n2h_AD_Converter_UME.pdf
SD/MMC Host Interface Port Output Enable 3 (POE3)	r01us0841ej0xxx-rz-t2h-n2h_POE3_UME.pdf
Thermal Sensor Unit (TSU)	r01us0842ej0xxx-rz-t2h-n2h_TSU_UME.pdf
PCI Express 3.0 Interface (PCIe)	r01us0843ej0xxx-rz-t2h-n2h_PCIe_UME.pdf
Power Management	r01us0844ej0xxx-rz-t2h-n2h_Power_Management_UME.pdf
Compare Match Timer W (CMTW)	r01us0845ej0xxx-rz-t2h-n2h_CMTW_UME.pdf

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Nov.28.25	-	First Edition issued for VLP v5.0.0.

Trademarks

- Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries.
- Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.
- Other company names and product names mentioned herein are registered trademarks or trademarks of their respective owners.
- Registered trademark and trademark symbols (® and ™) are omitted in this document.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.
   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION ("Vulnerability Issues"). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user's manuals, application notes, "General Notes for Handling and Using Semiconductor Devices" in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.
(Note2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on

   The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state

   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. Handling of unused pins

   Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

5. Clock signals

   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

6. Voltage application waveform at input pin

   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between VIL (Max.) and VIH (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between VIL (Max.) and VIH (Min.).

7. Prohibition of access to reserved addresses

   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

8. Differences between products

   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.