**Renesas RA family MCUs with Zephyr RTOS**

Software Guide

Contents

[1. Overview 3](#_Toc64987151)

[2. Tools and Software Versions required 3](#_Toc64987152)

[3. Project setup 4](#_Toc64987153)

[3.1 Install Zephyr dependencies for Windows 4](#_Toc64987154)

[3.1.1 Chocolatey installation 4](#_Toc64987155)

[3.1.2 Install dependencies 4](#_Toc64987156)

[3.2 Get Zephyr source code 4](#_Toc64987157)

[3.3 Install toolchain 5](#_Toc64987158)

[3.4 Test Zephyr installation 5](#_Toc64987159)

[3.5 Eclipse IDE plugin 5](#_Toc64987160)

[3.6 Integration of Renesas FSP into Zephyr HAL definitions 6](#_Toc64987161)

[4. Setup of Renesas RA4W1 SoC and development board 6](#_Toc64987162)

[4.1 RA4W1 device tree definition 6](#_Toc64987163)

[4.2 Driver/HAL interface definition 6](#_Toc64987164)

[4.3 SoC definition 6](#_Toc64987165)

[4.4 EK-RA4W1 development board definition 7](#_Toc64987166)

[5. EK-RA4W1 – Bluetooth sample application 7](#_Toc64987167)

[5.1 Zephyr Application 7](#_Toc64987168)

[5.2 Building errors that may occur 9](#_Toc64987169)

[5.3 Debug setup 10](#_Toc64987170)

[5.3.1 E2Studio 10](#_Toc64987171)

[5.3.2 Eclipse 11](#_Toc64987172)

[6. Renesas EK-RA6M3 development board with Zephyr OS 12](#_Toc64987173)

[7. Renesas EK-RA6M3 ethernet definition 12](#_Toc64987174)

[7.1 Peripheral definition for SoC 12](#_Toc64987175)

[7.2 EK-RA6M3 development board Ethernet definition 13](#_Toc64987176)

[7.3 Ethernet driver 13](#_Toc64987177)

[7.3.1 How to define a new driver 13](#_Toc64987178)

[7.3.2 Define an interface to the Renesas FSP package 14](#_Toc64987179)

[References 14](#_Toc64987180)

[Revision History 15](#_Toc64987181)

# Overview

This document describes the setup for Renesas EK-RA4W1 and EK-RA6M3 development board in Zephyr-RTOS and provides the description of Bluetooth sample application with EK-RA4W1.

RA4W1 Bluetooth hardware access is provided by *rble\_lib*, that also includes Bluetooth stack protocol. This cannot be used together with Zephyr Bluetooth stack.

*Rble\_library* uses FSP configuration files that will be included in the Bluetooth sample project.

**Note:** Zephyr project is in continuous work and may suffer changes that could impact the process described in this document.

# Tools and Software Versions required

Software versions required to start a Zephyr OS using Renesas RA devices:

* Toolchain installation required to build Zephyr applications.
* E2Studio or Eclipse IDE
* Eclipse IDE plugin for Zephyr project *https://github.com/zephyrproject-rtos/eclipse-plugin*

Difference from zephyrproject startup guide, the Zephyr-RA package depend on the following versions:

* Zephyr OS revision 2.4.0
* CMake 3.18.4
* GNU Arm Embedded Toolchain 9.3.1

**Note**: Make sure that you have installed [**Eclipse IDE for C/C++ Developers**](https://www.eclipse.org/downloads/packages/eclipse-ide-cc-developers/oxygen2). If not, the package can be downloaded from [*https://www.eclipse.org/downloads/packages/*](https://www.eclipse.org/downloads/packages/). This version has included the JLink Debugging plugin that will be used later.

# Project setup

Please follow the instructions from: *https://docs.Zephyrproject.org/latest/getting\_started/index.html*

The setup of the Zephyr project can be tested by building a sample project using one of the supported boards provided by Zephyr.

**Note**: for Windows machine use PowerShell for the entire setup process.

## Install Zephyr dependencies for Windows

The instructions are as described at <https://docs.zephyrproject.org/latest/getting_started/index.html#install-dependencies> with additional observations.

### Chocolatey installation

Follow instructions at <https://chocolatey.org/install> .

Install chocolatey using Windows PowerShell (powershell.exe) as administrator.

Check execution policy: Run “*Get-ExecutionPolicy*”. If it returns *Restricted*, then run *Set-ExecutionPolicy AllSigned* or *Set-ExecutionPolicy Bypass -Scope Process*.

To install chocolaty run the following command in powershell:

*>Set-ExecutionPolicy Bypass -Scope Process -Force; System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))*

### Install dependencies

Continue using Windows PowerShell as administrator.

Install *cmake* and *ninja* with the following commands:

*>choco install cmake –version 3.18.4 --installargs 'ADD\_CMAKE\_TO\_PATH=System'*

*>choco install ninja gperf python git*

Install *west* using the following command:

*>pip3 install west*

Restart your PC to set the Environment Variables

## Get Zephyr source code

Set you work/desired directory to check out the zephyr source code.

*>cd c:\work*

Execute the following commands to check out and update zephyr project:

*>west init --mr zephyr-v2.4.0 zephyrproject*

*>cd zephyrproject*

*>west update*

*>west zephyr-export*

Install additional dependencies for zephyr

>*pip3 install -r zephyr\scripts\requirements.txt*

## Install toolchain

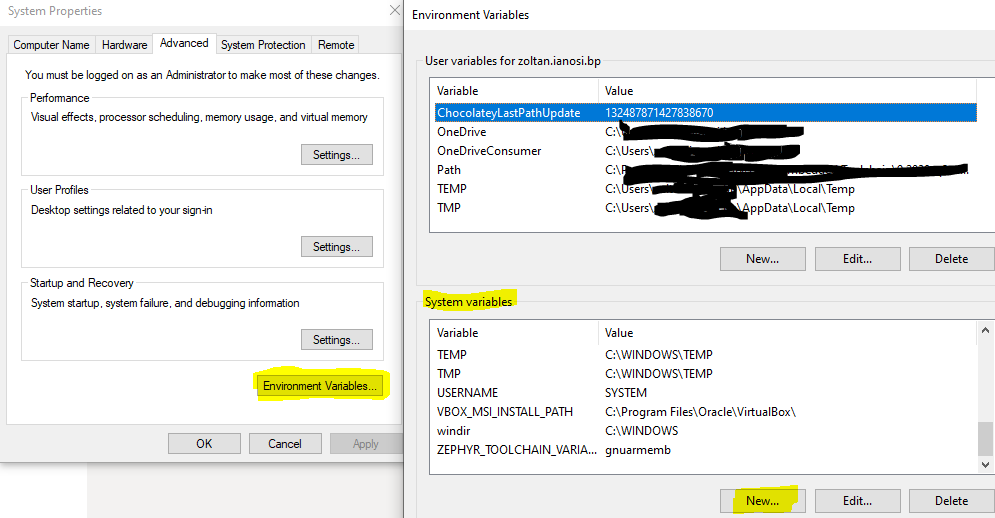
As a precaution install the toolchain in a directory without spaces, ex: *C:\gnu\_arm\_embedded*

Install GNU Arm Embedded Toolchain from: <https://developer.arm.com/tools-and-software/open-source-software/developer-tools/gnu-toolchain/gnu-rm/downloads>

For details please see: <https://docs.zephyrproject.org/latest/getting_started/toolchain_3rd_party_x_compilers.html#gnu-arm-embedded>

In the download section scroll down and select: *GNU Arm Embedded Toolchain: 9-2020-q2-update*

Add the following variable in Windows Environment variables:



Variable name: *ZEPHYR\_TOOLCHAIN\_VARIANT*

Variable value: *gnuarmemb*

Variable name: *GNUARMEMB\_TOOLCHAIN\_PATH*

Variable value: *c:\gnu\_arm\_embedded*

Restart the PC to ensure that the variables are registered/loaded.

## Test Zephyr installation

Build a sample application with a supported board using command line in PowerShell, ex:

Blinky sample application on supported board from c:\work\ zephyrproject\zephyr\boards\arm\

*>cd work\zephyrproject\zephyr*

*>west build -p auto -b <board\_directory> samples\basic\blinky*

## Eclipse IDE plugin

Install Eclipse IDE plugin for Zephyr project following the instructions from: [*https://github.com/Zephyrproject-rtos/Eclipse-plugin*](https://github.com/Zephyrproject-rtos/Eclipse-plugin)

The process is the same for Eclipse and E2Studio.

**Note**: Debug on hardware target will not work as described, as OpenOCD has no support for Renesas RA MCU’s. Debug on hardware target will be set up later in this document.

## Integration of Renesas FSP into Zephyr HAL definitions

Renesas FSP is included in the Zephyr\_RA package, and this can be updated if necessary, regarding Zephyr-project specific files.

Copy the folder *modules\hal\renesas* from Zephyr\_RA package to *Zephyrproject\modules\hal\.*

Zephyr ‘*hal’* modules are declared in:

* *Zephyrproject\Zephyr\west.yml*
* *Zephyrproject\Zephyr\modules\Kconfig*
* *Zephyrproject\Zephyr\modules\Kconfig.renesas* – contains HAL specific configuration definitions.

These files need to be updated by comparing(Beyond Compare, for instance) and edited with the ones provided in Zephyr\_RA package.

# Setup of Renesas RA4W1 SoC and development board

## RA4W1 device tree definition

SoC device tree definition (\*.dtsi) contains SoC specific peripheral definitions. Zephyr device tree documentation:

*https://docs.Zephyrproject.org/latest/application/index.html#devicetree-definitions*

*https://docs.Zephyrproject.org/latest/guides/dts/index.html*

Peripheral definition requirements are described in peripheral specific binding files (<peripheral>.yaml).

The Zephyr-RA4W1 sample provides peripheral definition for *gpio*, *pinctr* and serial(*sci*).

Device tree SoC include file:

*Zephyrproject\Zephyr \dts\arm\renesas\ ra4w1.dtsi*

Binding files:

*Zephyrproject\Zephyr \dts\arm\bindings*

IO port configurations used by the device tree are defined in:

*Zephyr\_RA package \Zephyr\include\dt-bindings\pinctrl\ra-pinctrl.h*

RA MCU family specific ROM register mapping is defined in the linker file:

*Zephyr\_RA package \Zephyr\include\arch\arm\aarch32\cortex\_m\scripts\linker.ld*

## Driver/HAL interface definition

Drivers for *gpio* and *serial* are rewritten based on the drivers provided by FSP as the architecture between FSP and Zephyr are not compatible.

Driver files:

*Zephyrproject\Zephyr\drivers*

Copy the provided files from *Zephyr\_RA* package to *Zephyrproject* folder, update(by comparing) *Cmake* and *Kconfig* files.

## **SoC definition**

*Zephyrproject\Zephyr\soc* provides SoC specific initialization function and configurations.

Copy “*renesas\_ra*” folder and its contents to *Zephyrproject\Zephyr\soc\arm*.

**Configurations for RA family**:

*Zephyrproject\Zephyr\soc\arm\renesas\_ra\ Kconfig.defconfig*

*Zephyrproject\Zephyr\soc\arm\renesas\_ra\ Kconfig.soc*

*Zephyrproject\Zephyr\soc\arm\renesas\_ra\ Kconfig*

**Configurations for RA4W1:** *Zephyrproject\Zephyr\soc\arm\renesas\_ra\ra\ra4w1\Kconfig.defconfig.series*

*Zephyrproject\Zephyr\soc\arm\renesas\_ra\ra\ra4w1\Kconfig.soc*

*Zephyrproject\Zephyr\soc\arm\renesas\_ra\ra\ra4w1\Kconfig.series*

**RA4W1 initialization:**

Initialization of RA4W1 is done using *SystemInit()* provided by Renesas FSP. The implementation is done in:

*Zephyrproject\Zephyr\soc\arm\renesas\_ra\ra\ra4w1\soc.c*

## EK-RA4W1 development board definition

EK-RA4W1 development board definition is provided in:

*Zephyr\_RA package Zephyr\boards\arm\ek\_ra4w1*

Copy the folder into:

*Zephyrproject\Zephyr\boards\arm*.

Detailed description for defining a custom board (board support is not provided by Zephyrproject):

*https://docs.Zephyrproject.org/latest/application/index.html#custom-board-devicetree-and-soc-definitions*

# EK-RA4W1 – Bluetooth sample application

Bluetooth sample application feature:

* LED blink rate change between 100 – 25500 ms, based on the value sent from Renesas GATTBrowser mobile application 0x01 – 0xFF.
* EK-RA4W1 – User Button press notification Renesas GATTBrowser mobile application.

User *LED gpio* is controlled via Zephyr OS GPIO interface.

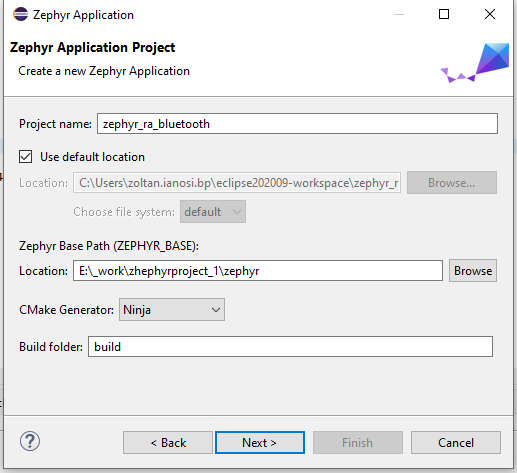
User *Button gpio* is set up as interrupt using Renesas FSP api and driver. BLE library interrupt is defined using Renesas Configurator, making Zephyr OS interrupt interface incompatible.

## Zephyr Application

Zephyr application setup is the same for both Eclipse and E2Studio.

Create a new Zephyr Application project “New->Project->Zephyr Application”.

Zephyr base: c:\work\zephyrproject\zephyr



**Fig.1 Eclipse project creation window**

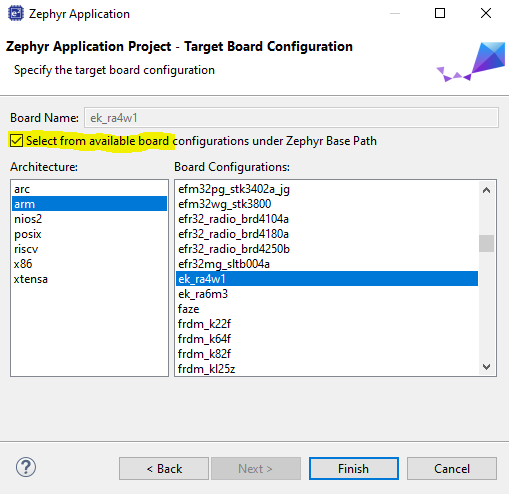
For Windows machine, select GNU ARM Embedded Toolchain and set toolchain path if required.

The toolchain path must be without “\bin”.

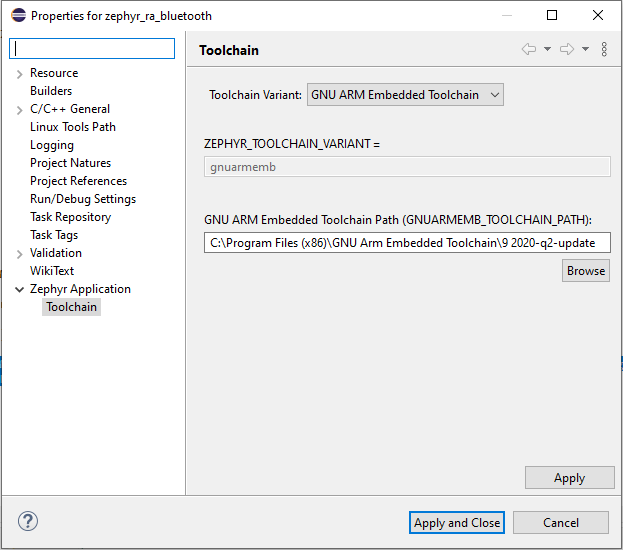
\

The board can be written or selected from the list provided by zephyrproject.

In the case that ek\_ra4w1 or eg\_ra6m3 boards don’t appear in the list please recheck chapter: *4.4 EK-RA4W1 development board definition*



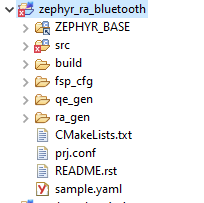
**Fig.2 Board selection**



**Fig.3 Eclipse Toolchain window**

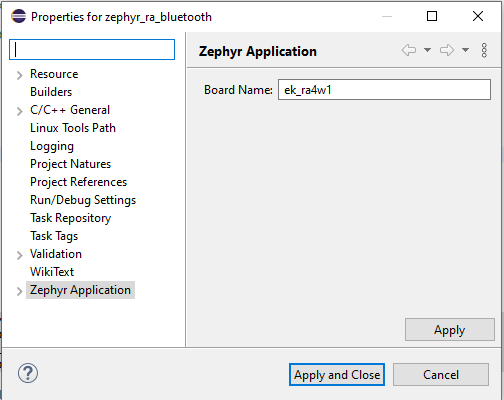
Copy and overwrite the sample application files provided in Zephyr\_RA package in the new project folder.

The sample application project should look like this:



**Fig.3 Project tree**

Check that board name (*ek\_ra4w1*) is correctly set for the sample application in the project properties.



**Fig.4 Eclipse sample application window**

Build the project by right clicking on the project folder and build it. If there are errors, try to clean the project or delete the content of build folder.

## Building errors that may occur

When building the project:

* **CMake Deprecation Warning** **(cmake\_minimum\_required):**

**Compatibility with CMake < 2.8.12 will be removed from a future version of CMake. Update the VERSION argument <min> value or use a ...<max> suffix to tell CMake that the project does not need compatibility with older versions.”** occurs then it is required to be changed the version of **cmake\_minimum\_required(VERSION 2.8)** to **cmake\_minimum\_required (VERSION 2.8...3.13)** in the indicated CMakeLists.txt.

* **“modules/Kconfig.nordic not found”:**

Edit *zephyrproject/zephyr/modules/Kconfig* according to [*https://github.com/zephyrproject-rtos/zephyr/blob/251648afb458b5593bbd8216e6fc169b0a870e92/modules/Kconfig*](https://github.com/zephyrproject-rtos/zephyr/blob/251648afb458b5593bbd8216e6fc169b0a870e92/modules/Kconfig). Do not forget to add **source "modules/Kconfig.renesas"** if it is not included.

* **Phyton missing module**:

There can be multiple versions of phyton installed on the PC. Zephyr will search for phyton3.x. Depending on the phyton3.x that Zephyr has found there can be missing packages in the phyton installation.

Preferred solution:

* try to install the missing module using: *>pip install <module name>*
* uninstall all the phyton instances and rerun *>choco install ninja gperf python git*

Alternate solutions:

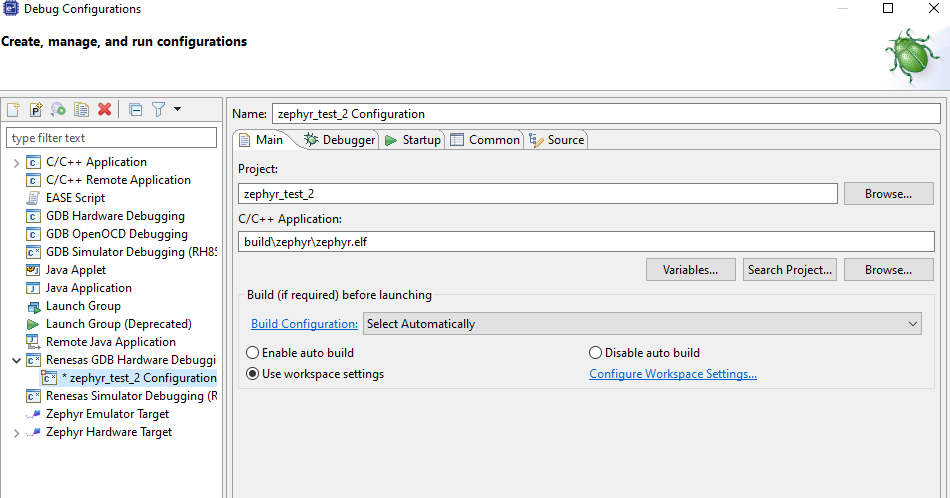
- set windows phyton environment variable to the path used by Zephyr (in general this can be C:/Phyton3.x) and install missing modules using *>pip install <module name>*

- if the module is present in another phyton instance, manually copy the module to the phyton instance used by Zephyr

## Debug setup

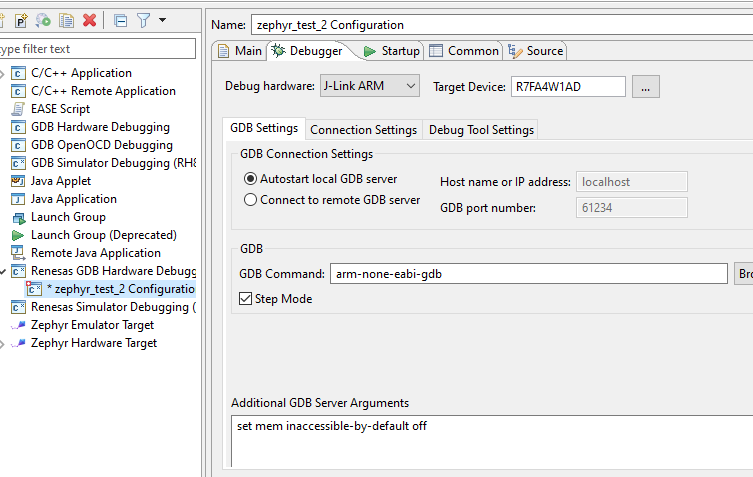
### E2Studio

Create a new debug configuration using “Renesas GDB Hardware debugging”.



**Fig.5 E2Studio debug configuration window**

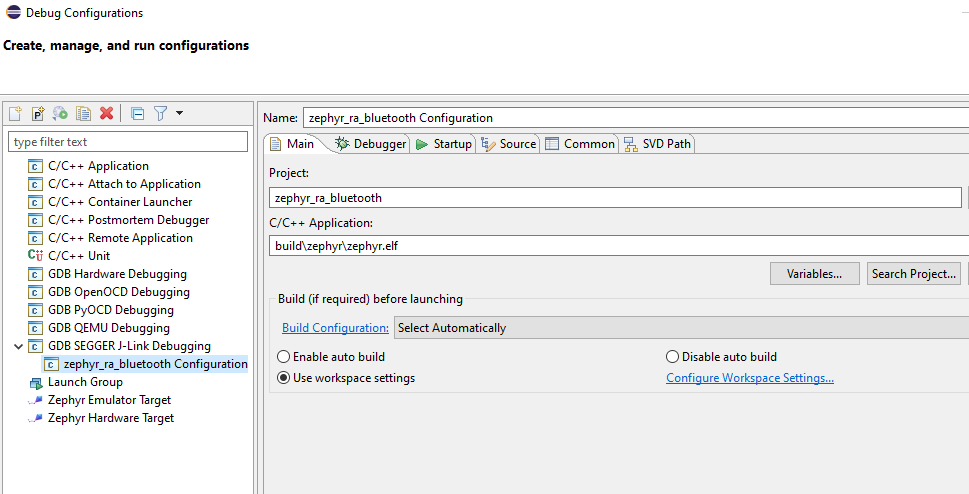
Setup the hardware debugger according to the image bellow:



**Fig.6 E2Studio debug configuration window**

### Eclipse

Create a new debug configuration using “*GDB SEGGER J-Link Debugging*”.

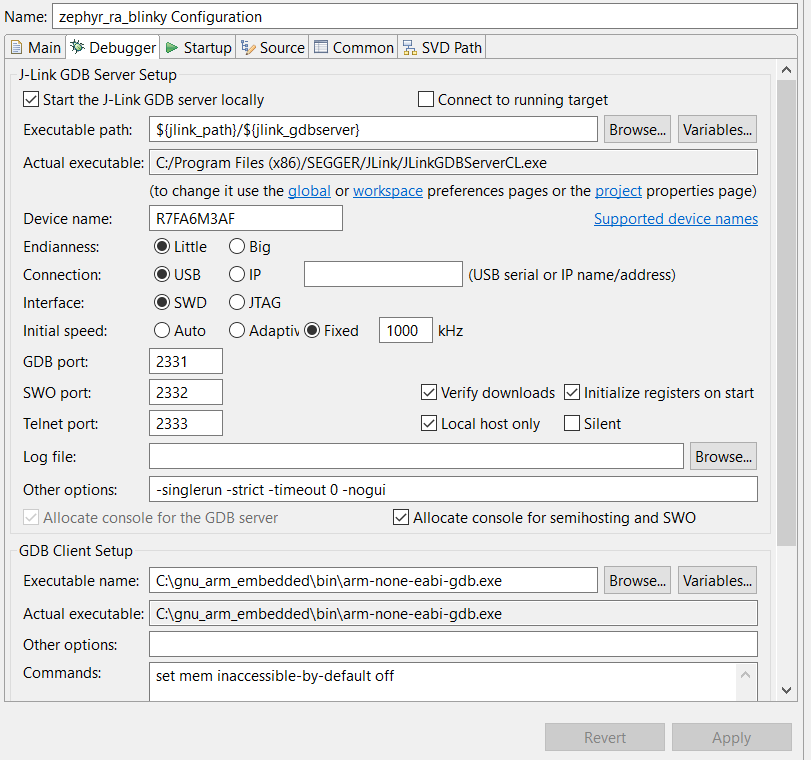


**Fig.7 Eclipse debug configuration window**

Under “Debugger” tab set the device name from [*https://www.segger.com/downloads/supported-devices.php/*](https://www.segger.com/downloads/supported-devices.php/)where are listed the supported devices for JLink.

Set the “Executable name” and “Actual executable” under GDB Client Setup (Debugger section) to the location *https://www.segger.com/downloads/supported-devices.php/* where GNU ARM Embedded Toolchain has been installed.

Bluetooth sample application can be flashed and debugged using this configuration.



**Fig.8 Eclipse debug configuration window**

# Renesas EK-RA6M3 development board with Zephyr OS

Integration of EK-RA6M3 development board into Zephyr OS project follows the same steps as above for EK-RA4W1 Chapter 4.

The setup can be tested with the Blinky sample application provided by the Zephyr project.

* Create a new Zephyr application project in Eclipse, specifying *ek\_ra6m3* development board.
* Copy Blinky sample application from “*Zephyrproject\Zephyr\samples\basic\blinky\*” into the newly create Eclipse project.
* Build and debug.

# Renesas EK-RA6M3 ethernet definition

## Peripheral definition for SoC

The ethernet peripheral definition for the SoC is described in the device tree include file: Zephyr\dts\arm\renesas\ra6m3.dtsi

Example from other dtsi:

mac: ethernet@40028000 {

compatible = "<soc provide>,<soc family>-ethernet";

reg = <0x40028000 0x8000>;

label = "ETH\_0";

interrupts = <61 0>;

clock-names = "maceth", "mac-clk-tx",

"mac-clk-rx", "mac-clk-ptp";

clocks = <&rcc CLOCK\_BUS\_AHB1 0x02000000>,

<&rcc CLOCK\_BUS\_AHB1 0x04000000>,

<&rcc CLOCK\_BUS\_AHB1 0x08000000>,

<&rcc CLOCK\_BUS\_AHB1 0x10000000>;

status = "disabled";

};

Peripheral definition required fields are defined in the corresponding binding *\*.yaml* file. The binding files must be defined for each new peripheral. Fields can be added, removed as required by the peripheral and driver.

Example of binding *\*.yaml* file:

***description:*** *<provide peripheral description>*

***compatible:*** *"<soc provide>,<soc family>-ethernet"*

***include:*** *ethernet.yaml*

***properties:***

***reg:***

***required:*** *true*

***interrupts:***

***required:*** *true*

***clocks:***

***required:*** *true*

***clock-names:***

***required:*** *true*

***pinctrl-0:***

***type:*** *phandles*

***required:*** *false*

***description:*** *|*

GPIO pin configuration for Ethernet signals. We expect that the *phandles*

will reference *pinctrl* nodes, e.g.

pinctrl-0 = <&eth\_ref\_clk\_pa1 &eth\_mdio\_pa2 ...>;

## EK-RA6M3 development board Ethernet definition

Board device tree shall activate the specific peripheral and specify the HW interfaces used (in the case of *ek-6m3* the definition of the pins used to interface to physical layer).

Example of ethernet definition in board device tree:

*&mac {*

*status = "okay";*

*pinctrl-0 = <&eth\_mdc\_pc1*

*&eth\_rxd0\_pc4*

*&eth\_rxd1\_pc5*

*&eth\_ref\_clk\_pa1*

*&eth\_mdio\_pa2*

*&eth\_crs\_dv\_pa7*

*&eth\_tx\_en\_pg11*

*&eth\_txd0\_pg13*

*&eth\_txd1\_pb13>;*

*};*

Pin definition is the same as used for UART-console:

*p206\_uart\_urxd: p206\_uart\_urxd {*

*renesas,pins = <&port2 6 (IOPORT\_CFG\_PERIPHERAL\_PIN | IOPORT\_PERIPHERAL\_SCI0\_2\_4\_6\_8)>;*

*};*

## Ethernet driver

There are two implementation possibilities for the ethernet driver:

* Define a new driver in *\Zephyr\drivers\ethernet* for *ra6m3* and the physical layer *KSZ8091RNB* for *ek-ra6m3.*
* Define an interface to the Renesas FSP package provided ethernet driver in *\Zephyr\drivers\ethernet.*

The ethernet driver shall use data from the device trees defined above in driver define structure ex: “**struct** eth\_hal\_dev\_cfg eth0\_config”. Data is passed through macros defined in *devicetree.h*.

### How to define a new driver

Below one can find the definition of a new driver that shall implement the functionalities of the ethernet peripheral without using functionality from the Renesas FSP package.

Zephyr OS ethernet stack requires driver implementations described in *\Zephyr\include\net\ethernet.h* in **struct** ethernet\_api.

Minimum functionality Zephyr requirement for the driver:

* *iface\_api.init* - initialization
* *get\_capabilities* – link speed capabilities and other if needed
* *set\_config* – ethernet specific configuration, ex: mac address if it must be set
* *send* – ethernet tx function

Received data is handled by a receiving thread “**static** **void** **rx\_thread**(**void** \*arg1, **void** \*unused1, **void** \*unused2)” that shall be defined also in the driver.

### Define an interface to the Renesas FSP package

The driver interface shall implement the minimum requirements as described in “7.3.1. Define new driver”.

Inclusion of the FSP package ethernet driver into Zephyr project will be required. This is shall be achieved using as example for *ek-ra4w1* that uses FSP package driver (ex: *r\_gpt*).

* Definition of *Kconfig* symbol is done in *Zephyr\soc\arm\renesas\_ra\Kconfig.defconfig*
  + *Ex:*

*config RENESAS\_DRIVER\_GPT*

*bool "Enable Renesas GPT driver"*

* Include FSP package driver source code based on defined symbol
  + *Ex for r\_gpt:*

*Add directory to Zephyr project:*

*\Zephyrproject\modules\hal\renesas\fsp\src\CMakeLists.txt*

*Add source file:*

*“Zephyr\_library\_sources\_ifdef(CONFIG\_RENESAS\_DRIVER\_GPT r\_gpt.c)”:*

*\Zephyrproject\modules\hal\renesas\fsp\src\r\_gpt\CMakeLists.txt*

*For ethernet driver there shall be two symbols one for “r\_ether” and one for “r\_ether\_phy”.*

The functionalities shall be interfaced with the FSP package ethernet driver with the necessary adaptations.

# References

1. Renesas Electronics, “Zephyr Project Documentation v2.4.99” [Zephyr Project Documentation — Zephyr Project Documentation](https://docs.zephyrproject.org/latest/#zephyr-project-documentation).

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Rev. | Date | Description | |
| Page | Summary |
| 0.1 | 08.01.2021 |  | Initial version. |
| 0.2 | 28.01.2021 |  | Update version |
| 0.3 | 18.02.2021 |  | Detailed setup process and tool versioning E2Studio setup |
|  |  |  |  |

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.

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 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.

5. 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.

6. 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.

7. 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.

8. 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.

9. 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.

10. 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.

11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.

12. 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.4.0-1 November 2017)

|  |  |  |
| --- | --- | --- |
| Corporate Headquarters |  | Contact information |
| TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan  [www.renesas.com](https://www.renesas.com/) |  | 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/](https://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. |  |  |