Quick Start Guide

Renesas RUHMI Framework

Quick Start Guide

Renesas RA Family
RA8P1 Series

Rev.1.00 July 2025

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.

Renesas EK-RA8P1

By using this EK-RA8P1, the User accepts the following terms, which are in addition to, and control in the event of disagreement, with Renesas' General Terms and Conditions available at https://www.renesas.com/en-us/legal/disclaimer.html.

The EK-RA8P1 is not guaranteed to be error free, and the entire risk as to the results and performance of the EK-RA8P1 is assumed by the User. The EK-RA8P1 is provided by Renesas on an "as is" basis without warranty of any kind whether express or implied, including but not limited to the implied warranties of good workmanship, fitness for a particular purpose, title, merchantability, and non-infringement of intellectual property rights. Renesas expressly disclaims any implied warranty.

Renesas does not consider the EK-RA8P1 to be a finished product and therefore the EK-RA8P1 may not comply with some requirements applicable to finished products, including, but not limited to recycling, restricted substances and electromagnetic compatibility regulations. Refer to Certifications section, for information about certifications and compliance information for the EK-RA8P1. It is the kit User's responsibility to make sure the kit meets any local requirements applicable to their region.

Renesas or its affiliates shall in no event be liable for any loss of profit, loss of data, loss of contract, loss of business, damage to reputation or goodwill, any economic loss, any reprogramming or recall costs (whether the foregoing losses are direct or indirect) nor shall Renesas or its affiliates be liable for any other direct or indirect special, incidental or consequential damages arising out of or in relation to the use of this EK-RA8P1, even if Renesas or its affiliates have been advised of the possibility of such damages.

Renesas has used reasonable care in preparing the information included in this document, but Renesas does not warrant that such information is error free nor does Renesas guarantee an exact match for every application or parameter to part numbers designated by other vendors listed herein. The information provided in this document is intended solely to enable the use of Renesas products. No express or implied license to any intellectual property right is granted by this document or in connection with the sale of Renesas products. Renesas reserves the right to make changes to specifications and product descriptions at any time without notice. Renesas assumes no liability for any damages incurred by you resulting from errors in or omissions from the information included herein. Renesas cannot verify, and assumes no liability for, the accuracy of information available on another company's website.

Precautions

This Evaluation Kit is only intended for use in a laboratory environment under ambient temperature and humidity conditions. A safe separation distance should be used between this and any sensitive equipment. Its use outside the laboratory, classroom, study area, or similar such area invalidates conformity with the protection requirements of the Electromagnetic Compatibility Directive and could lead to prosecution.
The product generates, uses, and can radiate radio frequency energy and may cause harmful interference to radio communications. There is no guarantee that interference will not occur in a particular installation. If this equipment causes harmful interference to radio or television reception, which can be determined by turning the equipment off or on, you are encouraged to try to correct the interference by one or more of the following measures:
• Ensure attached cables do not lie across the equipment.
• Reorient the receiving antenna.
• Increase the distance between the equipment and the receiver.
• Connect the equipment into an outlet on a circuit different from that which the receiver is connected.
• Power down the equipment when not in use.
• Consult the dealer or an experienced radio/TV technician for help.
Note: It is recommended that wherever possible shielded interface cables are used.
The product is potentially susceptible to certain EMC phenomena. To mitigate against them it is recommended that the following measures be undertaken:
• The user is advised that mobile phones should not be used within 10 m of the product when in use.
• The user is advised to take ESD precautions when handling the equipment.
The Evaluation Kit does not represent an ideal reference design for an end product and does not fulfill the regulatory standards for an end product.

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 V_{IL} (Max.) and V_{IH} (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 V_{IL} (Max.) and V_{IH} (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.

Renesas RUHMI Framework – Quick Start Guide

Using EK-RAP1

Contents

1. Introduction

This Quick Start Guide(QSG) provides a practical introduction to the Renesas RUHMI (Robust Unified Heterogeneous Model Integration) framework, in which AI compiler is by powered by EdgeCortix® MERA™. and it is optimized for Renesas RA microcontrollers.

The guide covers:

- An overview of RUHMI Framework's key features, including model quantization, compilation, and C source code generation.
- Step-by-step instructions for installing and running RUHMI Framework in both Linux and Windows environments.
- Usage examples for deploying TensorFlow Lite, ONNX and PyTorch models with optional acceleration using Arm® Ethos-U55.
- A description of the integrated tools that support calibration, reference data generation, and runtime testing.

RUHMI Framework is tailored for developers aiming to deploy machine learning models to low-power, memory-constrained Arm-based embedded systems. It supports both floating-point and quantized inference, enables automatic operator partitioning between CPU and NPU, and integrates seamlessly with common model formats. With its flexible design and efficient toolchain, RUHMI Framework enables rapid prototyping and deployment of edge AI applications such as computer vision and speech processing.

1.1 Assumptions and Advisory Notes

To use RUHMI Framework effectively, the following conditions and knowledge are assumed:

1. **Tool experience:** Users are expected to have prior experience with embedded development and AI model training using frameworks such as TensorFlow or PyTorch..
2. **Model access:** It is assumed that users already possess or have access to pre-trained models in TensorFlow Lite (.tflite),ONNX (.onnx) or PyTorch(.pte) format.
3. **Subject knowledge:** Familiarity with microcontroller-based systems, C programming, and Renesas Smart Configurator tools are required.
4. **Supported platforms:** RUHMI Framework targets Renesas RA family MCUs. Optional support for the Arm® Ethos-U55 NPU is available through the Arm Vela compiler. Verify hardware compatibility before use.
5. **Software versioning:** Always use the latest version of RUHMI Framework to ensure compatibility with supported models, libraries, and runtime tools.
6. **Screen references:** Figures and UI examples in this guide are for reference only. Actual screen layouts or paths may differ depending on the software version.

2. RUHMI Framework Overview

RUHMI (Robust Unified Heterogeneous Model Integration) Framework is a model deployment toolchain developed to enable AI model deployment on embedded processors. It enables efficient and scalable deployment of machine learning models on edge devices with limited memory and power budgets. RUHMI Framework's first backend iteration has been optimized to support Renesas RA8 series of microcontrollers by leveraging a modified version of MERA™2.0 from Renesas partner EdgeCortix®.

RUHMI Framework supports widely used model formats including TensorFlow Lite, ONNX, and PyTorch, and is capable of deploying models to both CPU-only and CPU+NPU (Arm® Ethos-U55) configurations.

RUHMI Framework is ideal for real-time applications such as:

- Object detection
- Speech recognition
- Industrial and consumer edge AI use case

It is designed to minimize hardware-specific complexity while offering developers a flexible, framework-agnostic toolchain.

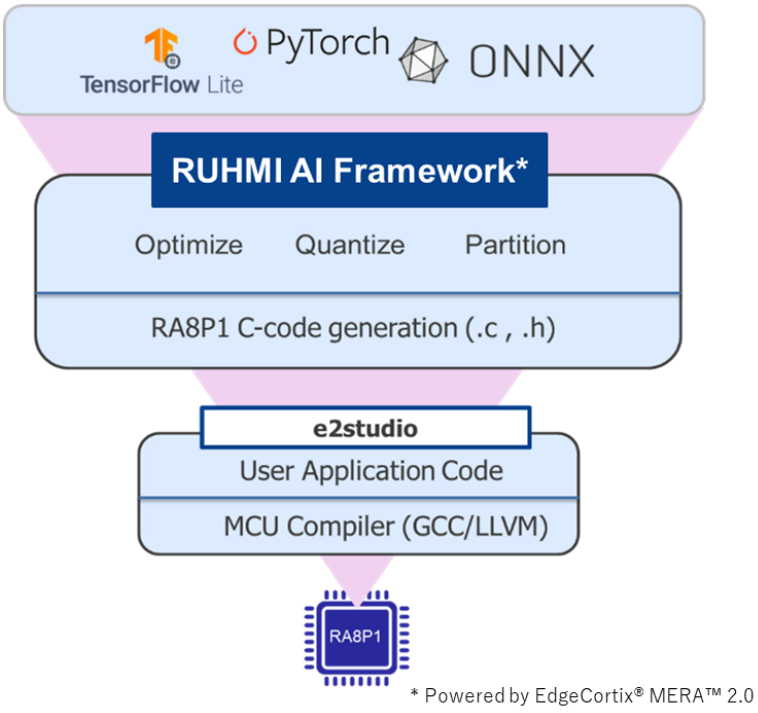

Figure 1 RUHMI Framework Overview

Key Features of RUHMI Framework

- Supports FP32 (floating-point) and INT8 (quantized) inference
- Automatically partitions operators between CPU and NPU based on hardware constraints
- Generates optimized C/C++ source code for integration into e² studio projects
- Uses CMSIS-NN and soft-float libraries for portable MCU inference
- Supports external memory for large models beyond SRAM size
- Built-in quantizer with PSNR-based validation for post-training quantization
- Integrated utilities for reference data generation, runtime testing, and MCU verification

- Seamless integration with TensorFlow Lite and ONNX model formats
- Compatible with both Linux CLI and Windows CLI / GUI(AI Navigator) environments

The sample project included in this guide demonstrates the following: Importing pre-trained AI models in TensorFlow Lite or ONNX format.

(1) Import of a pre-trained AI model in TensorFlow Lite or ONNX format

(2) Post-training quantization using RUHMI Framework's built-in quantizer

(3) Compilation into optimized C source code for Renesas RA8P1 platform

(4) Generation of input/output reference data for validation

(5) Deployment and execution on an evaluation board via Renesas e² studio

3. Running the Quick Start Example Project

This section lists the requirements and instructions to power up the EK-RA8P1 board and run the Quick Start Guide project. Refer to the latest release notes on GitHub to confirm compatibility

Hardware Requirements

- EK-RA8P1 Evaluation Kit
- USB-C to USB-C cable
- A PC with at least one available USB port

Software Requirements

- Windows® 10 or 11 operating system or Ubuntu 22.04 LTS.
- Renesas e² studio 2025-04.1 or later
- FSP 6.0 or later
- AI Navigator v2.0.0 for GUI

3.1 Downloading and Installing Software and Development Tools

Before modifying or running the Quick Start example project, it is necessary to download and install the required software and development tools on the host PC. RUHMI Framework supports installation and usage in the following environments:

Platform	Interface Type	Description
Linux (Ubuntu 22.04)	CLI	Full-featured environment. Recommended for advanced users and automation.
Windows	CLI	Suitable for scripting and model testing. Requires Python + VC Redistributable.
Windows	GUI	Integrated into e² studio. Ideal for beginners and visual, step-by-step setup.

3.1.1 Linux CLI Installation

Get the installation file (mera-2.3.1+pkg.1503-cp310-cp310-manylinux_2_27_x86_64.whl) from GitHub and place it in your PC.

GitHub repository: ruhmi-framework-mcu/install at main · renesas/ruhmi-framework-mcu

Update the package index to ensure your system has the latest information about available software packages. and installs essential development tools such as GCC, and Make (build-essential), along with CMake, which is a tool used to manage build processes for compiling source code. Run the following commands:

```
sudo apt update
sudo apt install build-essential cmake
```

Installs compression, encryption, database, and terminal libraries (libssl-dev, zlib1g-dev, libbz2-dev, libreadline-dev, etc.). Includes tools like curl and git for downloading and version control.

Provides headers and runtime libraries required for compiling Python from source and for tools like PyEnv.

```
sudo apt install libssl-dev zlib1g-dev libbz2-dev libreadline-dev libsqlite3-
dev curl git libncursesw5-dev xz-utils tk-dev libxml2-dev libxmlsec1-dev
libffi-dev liblzma-dev
```

Install PyEnv and Python 3.10.15

```
curl https://pyenv.run | bash
sudo add-apt-repository ppa:ubuntu-toolchain-r/test
sudo apt-get update
sudo apt-get install gcc-13 g++-13 libstdc++6
```

Add to ~/.bashrc or ~/.zshrc:

```
echo 'export PYENV_ROOT="$HOME/.pyenv"' >> ~/.bashrc
echo '[[ -d $PYENV_ROOT/bin ]] && export PATH="$PYENV_ROOT/bin:$PATH"' >>
~/.bashrc
echo 'eval "$(pyenv init - bash)"' >> ~/.bashrc
eval "$(pyenv virtualenv-init -)" >> ~/.bashrc
source ~/.bashrc
```

Then run:

```
pyenv install 3.10.15
pyenv virtualenv 3.10.15 ruhmi-env
pyenv activate ruhmi-env
```

Install Python Dependencies

```
(ruhmi-env) pip install --upgrade pip
(ruhmi-env) pip install decorator typing_extensions psutil attrs pybind11
```

Install Python Package

```
(ruhmi-env) cd install
(ruhmi-env) pip install ./ mera-2.4.0+pkg.1702-cp310-cp310-
manylinux_2_27_x86_64.whl
```

Verify Python Package

```
(ruhmi-env)  python -c "import mera; print(mera.__version__)"
2.4.0+pkg.1503
```

Now you can deactivate pyenv by “pyenv deactivate” and activate again by “pyenv activate ruhmi-env”

3.1.2 Windows CLI Installation

Download the Microsoft Visual C++ Redistributable

Note:

- You must download vc_redist.x64.exe since RUHMI Framework's MERA™2.0 backend only supports X64 architecture PCs
- Run the download exe file to install Redistributable package.

  Microsoft Visual C++ Redistributable latest supported download

Install Python 3.10.15

Download Python installer (python-3.10.5-amd64.exe) from **PYTHON RELEASE PYTHON 3.10.15**
And install it.

Open PowerShell. Then you can verify your Python version.

```
PS C: >python  --version
Python 3.10.15
```

Install MERA backend and dependencies

Get the installation file (mera-2.4.0+pkg.168-cp310-cp310-win_amd64.whl) from GitHub and place it in your PC.

GitHub repository: ruhmi-framework-mcu/install at main · renesas/ruhmi-framework-mcu

Assuming the working location as "C:\git\ruhmi-framework-mcu".

The installation file shold be here.

```
PS C:\git\ruhmi-framework-mcu\install
```

Before installing the software, you shall make the virtual environment on Python.

```
PS C:\git\ruhmi-framework-mcu > py -3.10 -m venv .venv
PS C:\git\ruhmi-framework-mcu > .venv\Scripts\Activate.ps1
```

You may need to change the execution policy for running the shell.

```
PS C:\git\ruhmi-framework-mcu > Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -
Scope Process
PS C:\git\ruhmi-framework-mcu > .venv\Scripts\Activate.ps1
```

In PowerShell, run the following commands to install MERA™.

```
(.venv) PS C:\git\ruhmi-framework-mcu > python -m pip install .\ mera-2.4.0+pkg.168-
cp310-cp310-win_amd64.whl
(.venv) PS C:\git\ruhmi-framework-mcu > python -m pip install onnx==1.17.0
tflite==2.18.0
(.venv) PS C:\git\ruhmi-framework-mcu >
[Environment]::SetEnvironmentVariable('CONVERSION_TOOL_E2STUDIO_PLUGIN_PYTHON_VENV_LO
C', "$(Get-Location)", 'User')
```

Please check that all your path settings of your environment are correct. After installation you should be able to successfully complete the following commands.

```
(.venv) PS C: \git\ruhmi-framework-mcu > vela --version

4.2.0

(.venv) PS C: \git\ruhmi-framework-mcu > python -c "import mera;print(dir(mera))"

['Deployer', 'InputDescription', 'InputDescriptionContainer', 'Layout', 'MERADeployer', 'MeraModel',
'MeraTvmDeployment', 'MeraTvmModelRunner', 'MeraTvmPrjDeployment', 'ModelLoader',
'ModelQuantizer', 'Platform', 'PowerMetrics', 'QuantizationQualityMetrics', 'Quantizer', 'TVMDeployer',
'Target', '__builtins__', '__cached__', '__doc__', '__file__', '__loader__', '__name__', '__package__',
'__path__', '__spec__', '__version__', 'calculate_quantization_quality', 'deploy', 'deploy_project',
'get_mera_dna_version', 'get_mera_tvm_version', 'get_mera_version', 'get_versions',
'load_mera_deployment', 'mera_deployment', 'mera_model', 'mera_platform', 'mera_quantizer',
'metrics', 'model', 'quantization_quality', 'quantizer', 'version']
```

3.2 Using RUHMI Framework Deployment Scripts for Linux and Windows CLI

RUHMI Framework provides Python-based command-line tools to help users deploy AI models on Renesas RA8 MCUs with or without acceleration by Arm® Ethos-U55. The main scripts are:

- mcu_deploy.py: for deploying non-quantized (FP32) models
- mcu_quantize.py: for quantizing models and deploying optimized C code
  Both scripts support additional options for reference data generation and memory configurations.

.

3.2.1 Compiling Non-Quantized TFLite Models

To deploy a pre-trained **TFLite model** without quantization:

```
python scripts/mcu_deploy.py --ethos --ref_data models/ output_dir/
```

Option	Description
--ethos	Enables code generation for Arm Ethos-U55
--ref_data	Generates input/output .npy files for testing
models/	Directory containing .tflite model files
output_dir/	Output folder where C code and test data are saved

Example output:

A folder named output_dir/model_name_no_ospi/ containing:

- MCU-compatible C source files (.c, .h)
- Reference data (inputs.npy, outputs.npy)
- Deployment logs

3.2.2 Quantizing and Compiling Models

To quantize a TFLite, ONNX, or Executorch (.pte) model and compile it for MCU:.

Example command:

```
python scripts/mcu_quantize.py models_fp32/ compiled_models/ -c 5 --ethos --ref_data
```

Option	Description
-c, --calib_num	Number of random samples used for calibration (default: 5)
--ethos	Target Arm Ethos-U55 for accelerated subgraphs
--ref_data	Generate C99-format reference input/output for validation
models_fp32/	Input folder with FP32 models (.tflite, .onnx, .pte)
compiled_models/	Output folder for quantized and compiled results

Example output folders:

```
compiled_models/
└─── model_001_mobilenet/
    ├─── quantization/        # Quantized .mera file
    ├─── deploy_qtz/          # Ref input/output (.npy) from quantized model
    ├─── deploy_mcu/          # Final C source for deployment
    └─── reference_data/      # (Optional) test data from FP32 model
```

3.3 Deployment Output Structure

After running the deployment scripts, the following folder and file structure will be generated for each model. These represent the C source code, headers, and supporting files necessary to integrate the model into an MCU project.

```
deploy/
├── person-det_no_ospi
│   ├── build
│   │   └── MCU
│   │       ├── compilation
│   │       │   ├── mera.plan
│   │       │   ├── src
│   │       │   │   ├── CMakeLists.txt
│   │       │   │   ├── compare.cpp
│   │       │   │   ├── compute_sub_0000.c
│   │       │   │   ├── compute_sub_0000.h
│   │       │   │   ├── ethosu_common.h
│   │       │   │   ├── hal_entry.c
│   │       │   │   ├── kernel_library_int.c
│   │       │   │   ├── model.c
│   │       │   │   ├── model.h
│   │       │   │   ├── model_io_data.c
│   │       │   │   ├── model_io_data.h
│   │       │   │   ├── python_bindings.cpp
│   │       │   │   ├── sub_0001_command_stream.c
│   │       │   │   ├── sub_0001_command_stream.h
│   │       │   │   ├── sub_0001_invoke.c
│   │       │   │   └── sub_0001_invoke.h
│   │       ├── deploy_cfg.json
│   │       ├── ir_dumps
│   │       │   ├── person-det_can.dot
│   │       │   └── ...
│   │       ├── person-det_after_canonicalization.dot
│   │       └── person-det_subgraphs.dot
│   ├── logs
│   ├── model
│   │   └── input_desc.json
│   └── project.mdp
```

Descriptions of Key Files and Directories:

File Name		Description
mera.plan		Deployment plan file describing subgraph partitioning and compilation sequence.
CMakeLists.txt		CMake configuration for compiling the generated C source code.
compare.cpp		C++ utility to compare model output vs. reference data (for validation).
compute_sub_*.c/.h	✓✓	Source and header files for subgraphs executed on CPU.

File Name		Description
ethosu_common.h	✓	Shared Ethos-U55 header with data types and macros.
hal_entry.c		MCU entry point source file for application runtime.
kernel_library_int.c	✓✓	Common CPU operator implementations used in the model.
model.c/.h	✓	Model execution stub and interface for invoking inference.
model_io_data.c/.h	✓✓	Input/output buffer data declarations for the model.
python_bindings.cpp		Python binding support for on-host simulation and testing.
sub_*_command_stream.c/.h	✓	Command stream source and headers for Ethos-U55 execution.
sub_*_invoke.c/.h	✓	Function wrappers for launching Ethos-U55 command streams.
deploy_cfg.json		Deployment configuration summary in JSON format.
ir_dumps/*.dot		Graph visualization files for different compilation stages.
input_desc.json		JSON description of input tensor names and shapes.
project.mdp		e2studio project metadata file.

✓ Output file used in CPU + Ethos U55

✓✓ Output file used in CPU only and CPU + Ethos U55

Blank Output file for debug / test and not used in MCU compilation

3.4 Example: Inference API (Ethos-U Enabled)

```
void RunModel(bool clean_outputs);      // Runs the full model
float* GetModelInputPtr_input0();       // Access to input buffer
float* GetModelOutputPtr_out0();        // Access to output buffer
```

3.5 Example: Inference API (CPU-Only)

```
void compute_sub_0000(
    uint8_t* main_storage,    // Intermediate buffer (use size defined in enum)
    const int8_t input[],     // Model input
    int8_t output[]           // Model output
);
Size of main_storage is defined in:

enum BufferSize_sub_0000 {
    kBufferSize_sub_0000 = <bytes_needed>
};
```

4. Running RUHMI Framework with AI Navigator (Windows GUI)

RUHMI Framework also supports a GUI-based model deployment workflow via the AI Navigator in Renesas e² studio. This section explains how to import and convert an AI model (e.g. TFLite) using the GUI, and how to integrate it into your Renesas RA8P1 project.

4.1 Introduction

AI Navigator is a set of plugins for e² studio that makes it easy to develop the edge AI application with Renesas devices. One of the plugins, AI Model Conversion Tool Plugin uses RUHMI Framework to provide AI model conversion feature.
The development steps using AI Navigator are shown below. All steps can be controlled by the AI Navigator.

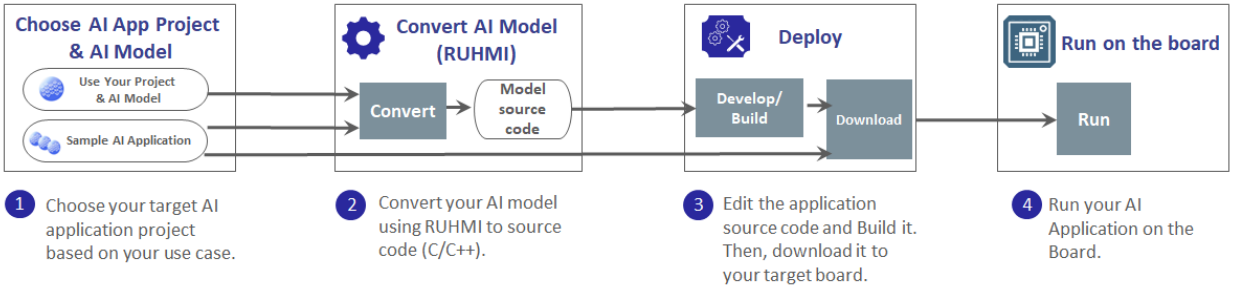

Figure 4.1 AI Navigator development flow

4.2 Preparation

4.2.1 Supported version

- **AI Navigator v2.0.0**

  Visit the AI Navigaor web page (AI Navigator: IDE for AI Applications) and download the release note. It includes the features, changes, additional notes, and so on.

4.2.2 Software Requirements

- Windows 10 or 11.
- Renesas e² stuido 2025-04.1
- RA Flexible Software Package (FSP) v6.0.0 or later
  Refer to the webpage (RA Flexible Software Package Documentation) for the details.

4.2.3 Installation

- **New e² studio users**

  Download e² studio 2025-04.01 from the following link:
  e² studio 2025-04.1 installer for Windows

  After downloading, install e² studio. Select the following each item when installing.

  - Device Family: RA
  - Additional Software: Renesas AI Navigator, AI Model Conversion Tool Plugin, Renesas FSP v6.0.0 or later

- **e² studio users**

  Download and install FSP v6.0.0. Then, add the necessary plugins by following the steps below.:

  1. Launch e² studio.
  2. Click **[Help]** > **[Install Renesas IDE Features...]**.
  3. Select **[Renesas AI Navigator]** and **[AI Model Conversion Tool Plugin]**. Click **[Finish]**.
  4. Confirm that these plugins are selected in the "install" dialog box and click **[Next]**.
  5. Confirm these plugins are selected as the installation target and click **[Next]**.
  6. Read the license agreements and select **"I accept the terms of the license agreement"** if you agree. Then, click **[Finish]** to start the plugins installation.
  7. If the dialog of the trust certificate dialog appears during the installation, check the certificate and click **[OK]**.
  8. After the installation is finished, e² studio will prompt you to restart. Click **[Restart Now]**.

  *Note: If [Install Renesas IDE Features...] is not listed in [Help], please follow the steps below instead.

  1. Click [Help] > [Install New Software...].
  2. Click "Add..." and enter the following URLs in Location.
     - AI Navigator Plugin: https://tool-support.renesas.com/e2studio/ai/ai-navi
     - AI Model Conversion Tool Plugin: https://tool-support.renesas.com/e2studio/ai/rz_tvm

4.3 Getting Started

This section explains how to integrate an AI model into your project with AI Navigator, including model conversion using RUHMI Framework on GUI.

The following figure shows the workflow from opening to running on the board. It includes AI model conversion using RUHMI Framework.

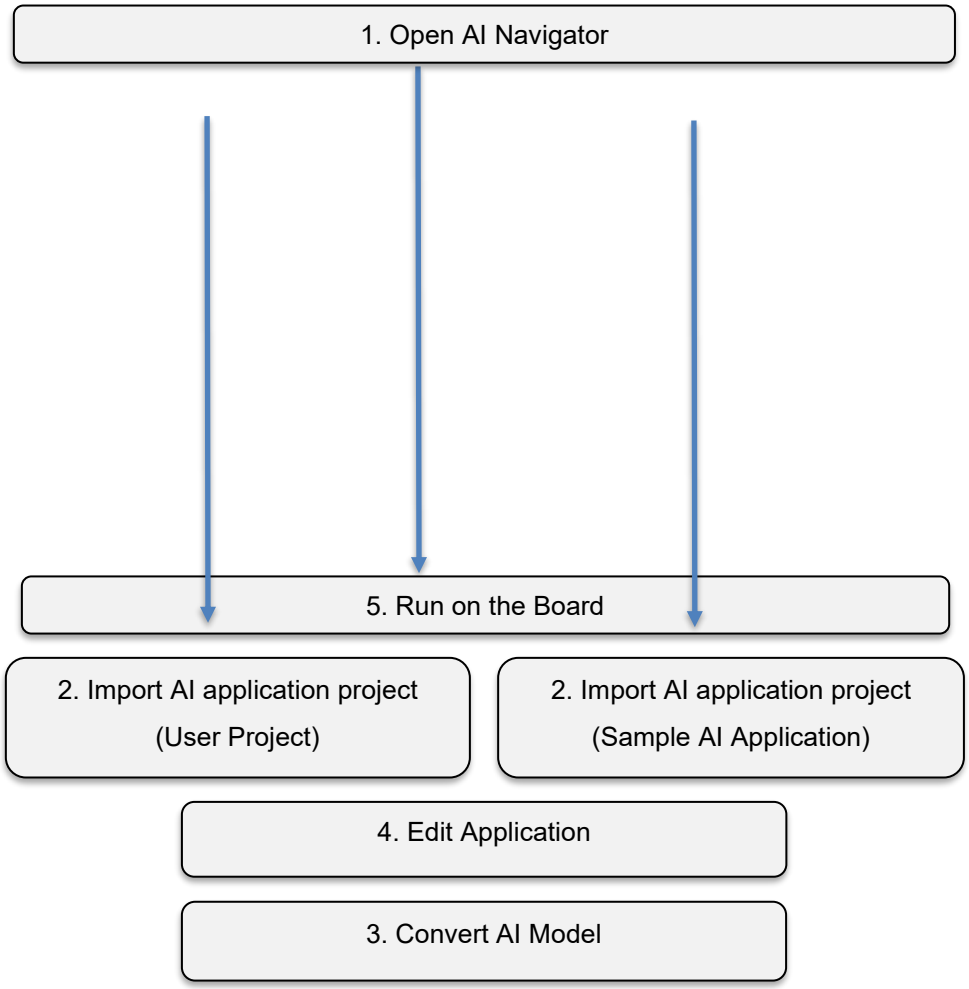

Figure 4.2 AI Navigator Workflow

4.3.1 Open AI Navigator

Open e^2 studio and your workspace.

Click **[Renesas Views] > [Renesas AI] > [AI Navi]** as shown in the figure below to launch AI Navigator.

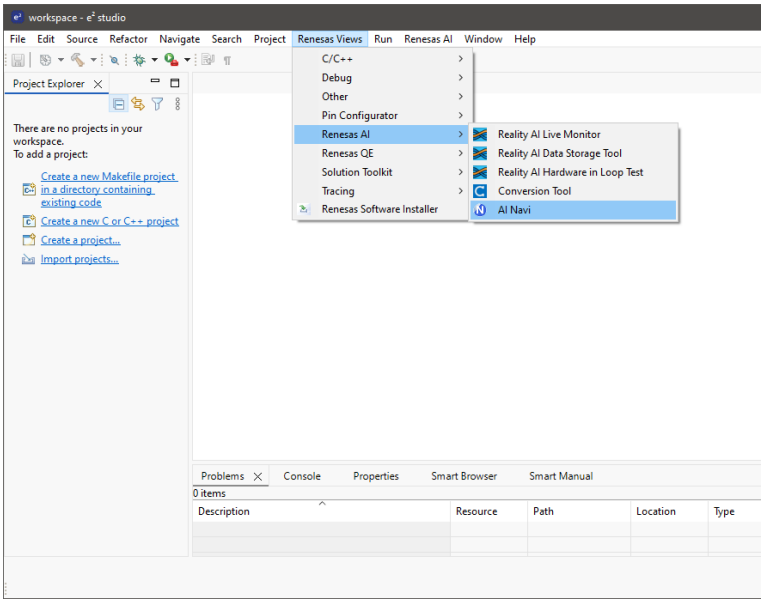

Figure 4.3 Opening AI Navigator

A dialog box may appear asking you to switch perspectives. If you accept, click **"Switch"** to continue.

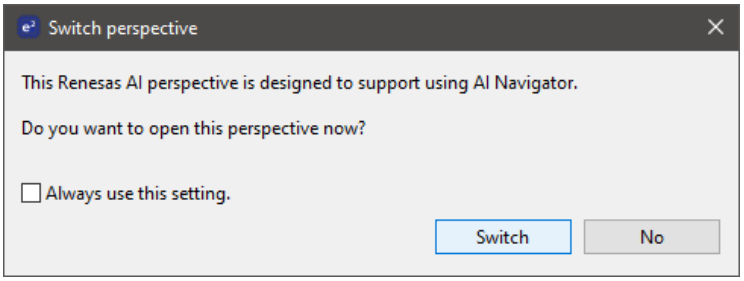

Figure 4.4 Switch perspective dialog box

AI Navigator will open.

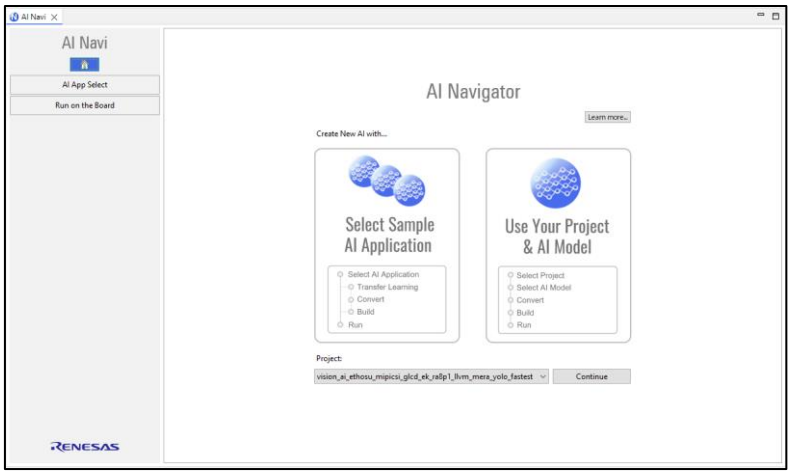

Figure 4.5 AI Navigator Welcome View

Note: If the Appearance Theme is set to "Light (Preview)", some GUI components such as buttons may not be displayed correctly.
Please change Theme to "**Light**" or "**Dark**" from **[Window]** -> **[Preferences]** -> **[General]** -> **[Appearance]**.

4.3.2 Import AI Application Project

There are two ways to import a project.

- **Select Sample AI Application**
- **Use Your Project & AI Model**

4.3.2.1 Select Sample AI Application

RA8P1 provides the following two AI Application.

- Image classification
- Face detection

Using AI Navigator, you can import them into your workspace and run them onto the target board. Follow the instructions below to import.

1. Click **[Select Sample AI Application]**.
2. Select a category of AI application based on your case.
3. Choose an AI Application.
4. Click **[Import]** and AI application.
5. The Project and AI information view will appear if the import is successful.

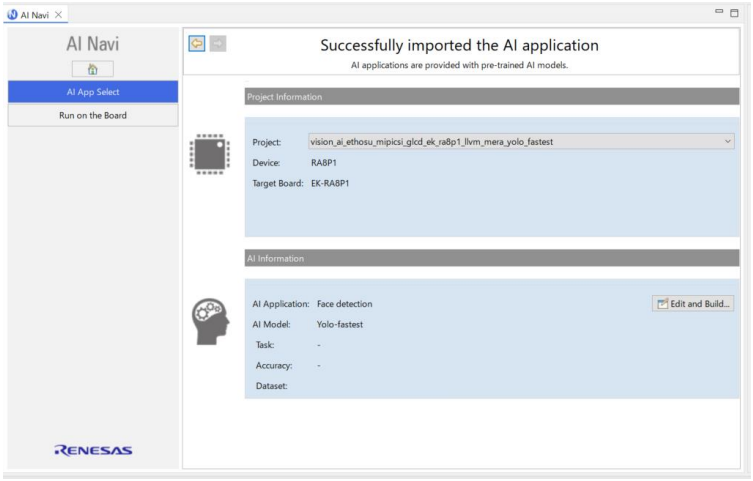

Figure 4.6 Project and AI information view (Sample AI application)

Note: If you just want to run the sample AI application, click **[Run on the Board]** in the AI Navigator menu. (See 4.3.5 Run on the Board for details.)
If you want to convert an AI model, click **[Edit and Build…]** in AI information to appear **[Convert AI Model]** in the menu.

4.3.2.2 Use Your Project & AI Model

AI Navigator also provides the user project feature that allows users to develop AI Applications with their own projects, including their own AI model. Follow the instructions below to prepare and import your project and AI model.

1. Create or import an RA8P1 e^2 studio project into your work space.
   *Note: Refer to RA Flexible Software Package Documentation for information about FSP project.
   *Note: Add the following stacks if you use Ethos:
   - Google TFLM Core Lib
   - Google TFLM CMSIS-NN Kernel

   Once added, the required dependent stacks (such as ARM Ethos-U Core Driver, ARM CMSIS NN Library Source, ARM CMSIS DPS Library Source) will be included automatically.
2. Click **[Use Your Project & AI Model]** in the AI Navigator Welcome View.
3. Select your project prepared in 1. and click **[Finish]**.

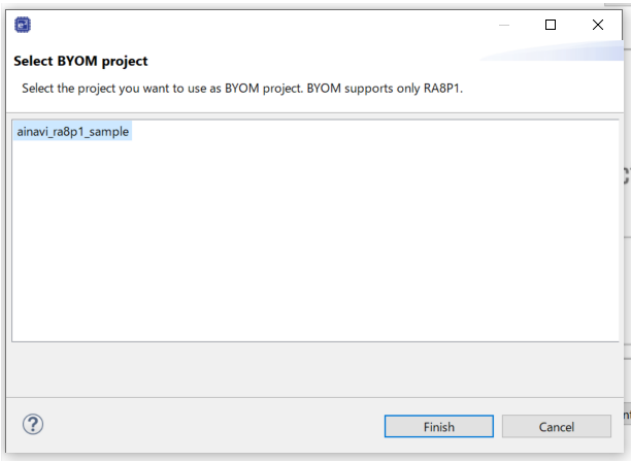

Figure 4.7 Select your project window

4. Click **[Use AI Model on Your PC]**.

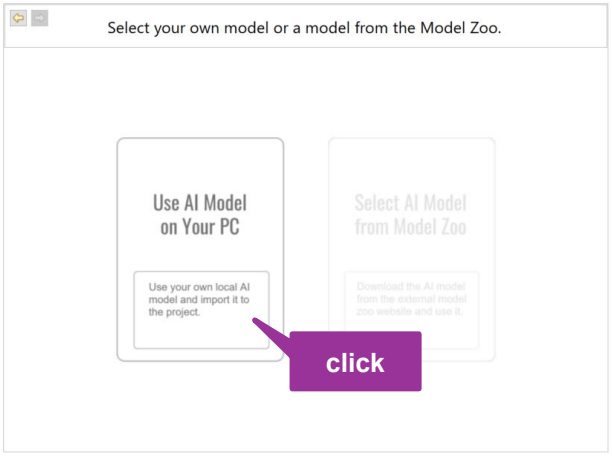

Figure 4.8 Use AI Model on Your PC

5. Select your AI model (*.tflite / *.onnx).
   *Note: Please avoid using spaces in your AI model name. The AI model conversion may not succeed.
6. The Project and AI information view will appear when the project and AI model are imported successfully.

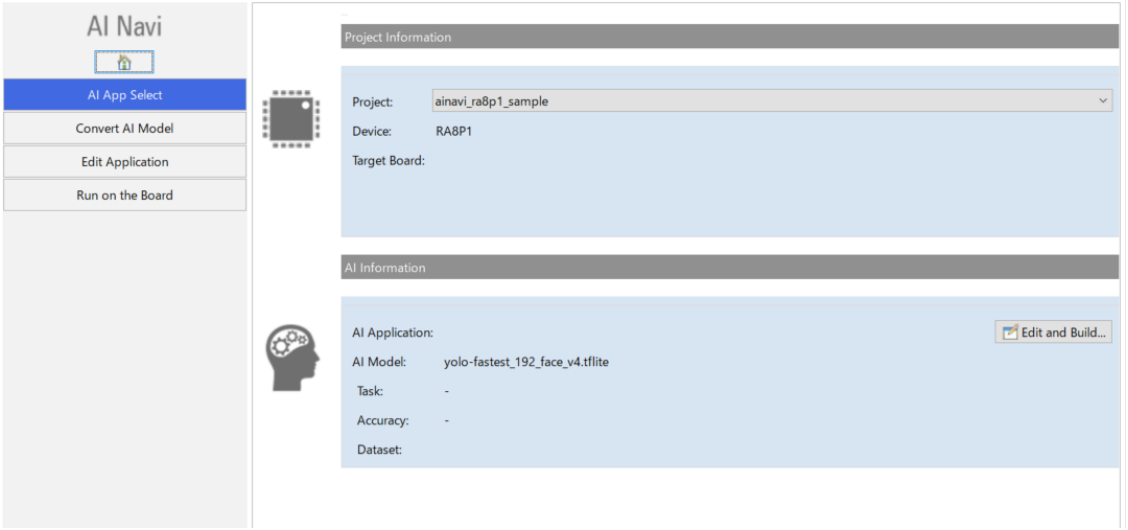

Figure 4.9 Project and AI information view (Use your project & AI model)

4.3.3 Conversion

After the setup, click **[Convert…]** and Conversion Tool will open.

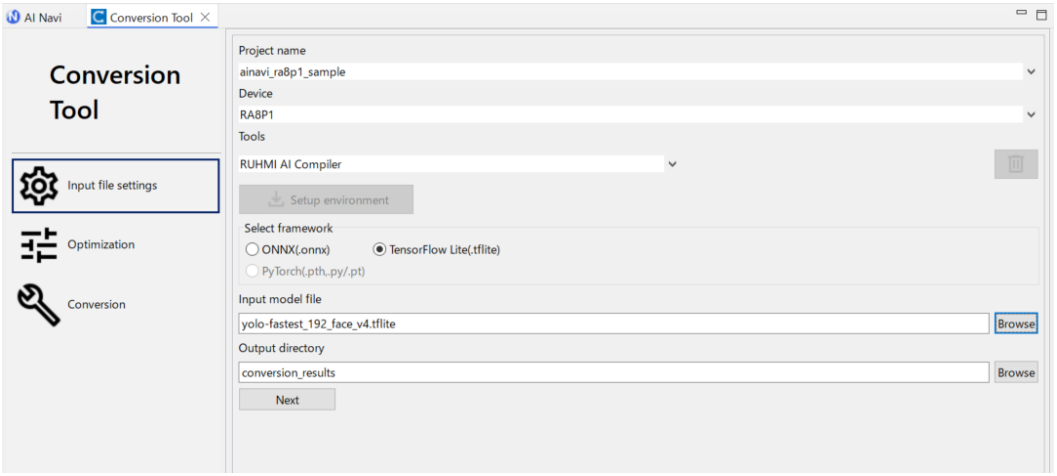

Figure 4.10 Conversion Tool

The AI model conversion follows the steps below. These steps are also shown in the left-side menu in the figure above.

(1) Input file settings

(2) Optimization

(3) Conversion

(1)Input file settings

Please set the following settings in this view. After setting, click **[Next]**.

- **Project name**: Select your project.
- **Device**: RA8P1
- **Tools**: RUHMI Framwork
- **Select framework**: ONNX (.onnx) or TensorFlow Lite (.tflite)
- **Input model file**: Click **[Browse]** and select your target AI model.
- **Output result**: Specify the output directory for the conversion result.

(2)Optimization

In this view, you can perform quantization as needed to prepare your AI model into an optimized format.

If you selected a quantized model as the input model file in the previous view, confirm that the message indicating a quantized AI model was loaded appears in Quantization result, and click **[Next]**.

If you selected a non-quantized model, perform quantization in this view. Set each parameter and click **[Start quantization]**. If quantization is successful, a message as shown in Figure 4.13 will appear. Click [Next] to continue.

Note: Click **[Learn more]** in this view and open AI Model Conversion Tool Help.
Refer to View 2: Optimization (Quantization) for the details about quantization.

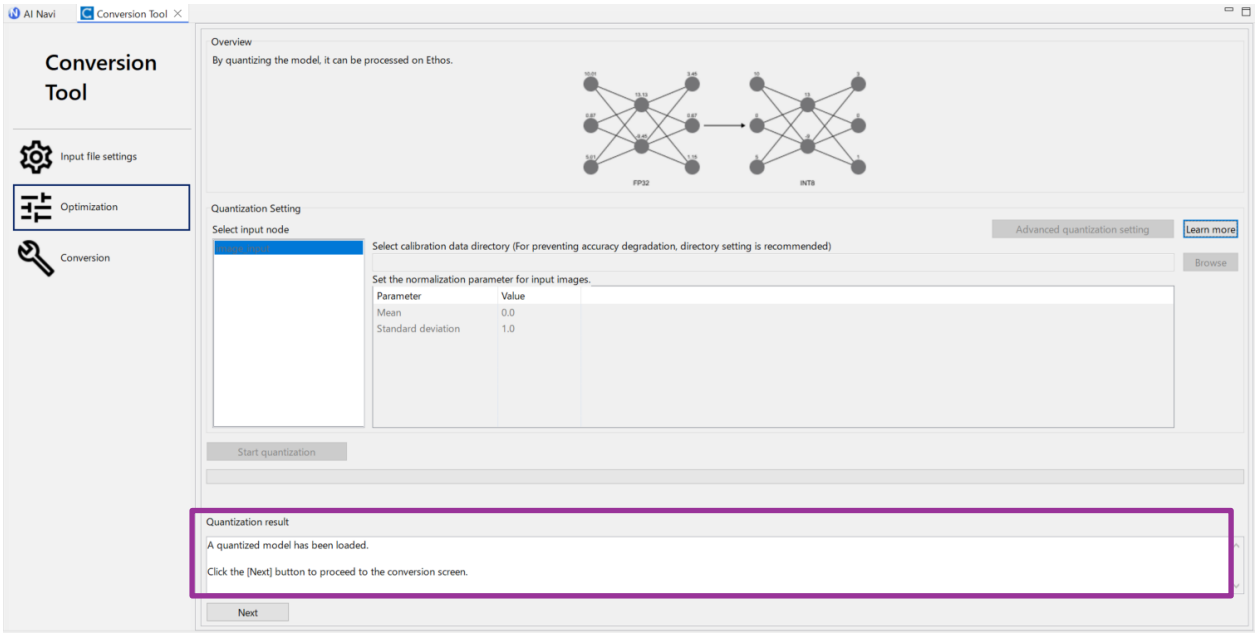

Figure 4.11 Optimization (input file: a quantized model)

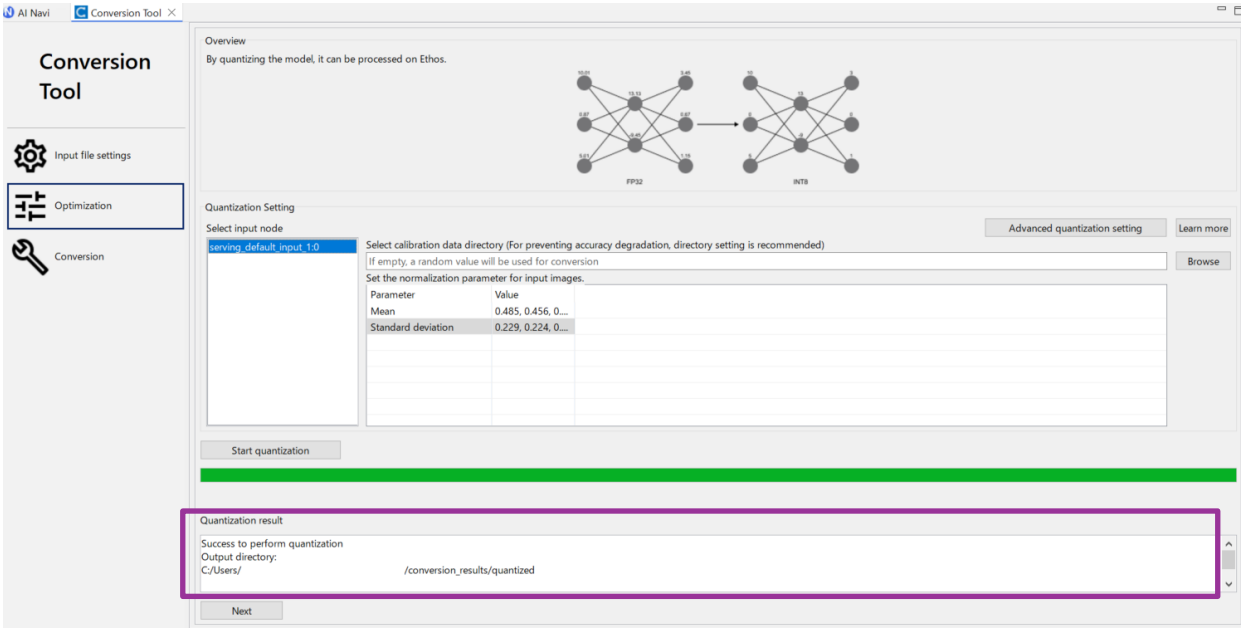

Figure 4.12 Optimization (input file: a non-quantized model)

(3)Conversion

Configure the following options as needed before conversion.

- **Optimize mode**:
  - Performance: (default) Optimize for maximum performance.
  - Size: Optimized for minimal RAM usage.
- **Memory mode**:
  - Sram_Only: (default) Specify when placing weights in the internal ROM.
  - Shared_Sram: Specify when placing weights in the external ROM.
- **Use only CPU**:
  - unchecked: Perform inference using Ethos and CPU.(default)
- - checked: Perform inference using CPU only.
- **Weight location (*for using CPU only)** :
  - ROM: (default) Neural network weight data is stored in ROM.
  - RAM: Neural network weight data is stored in RAM.
- **Additional options**:
  Click **[Browse]** and select your target AI model.

After setting options, click **[Start conversion]**. If the conversion is successful, a message indicating success will appear in the Conversion Result section, as shown in the figure below.

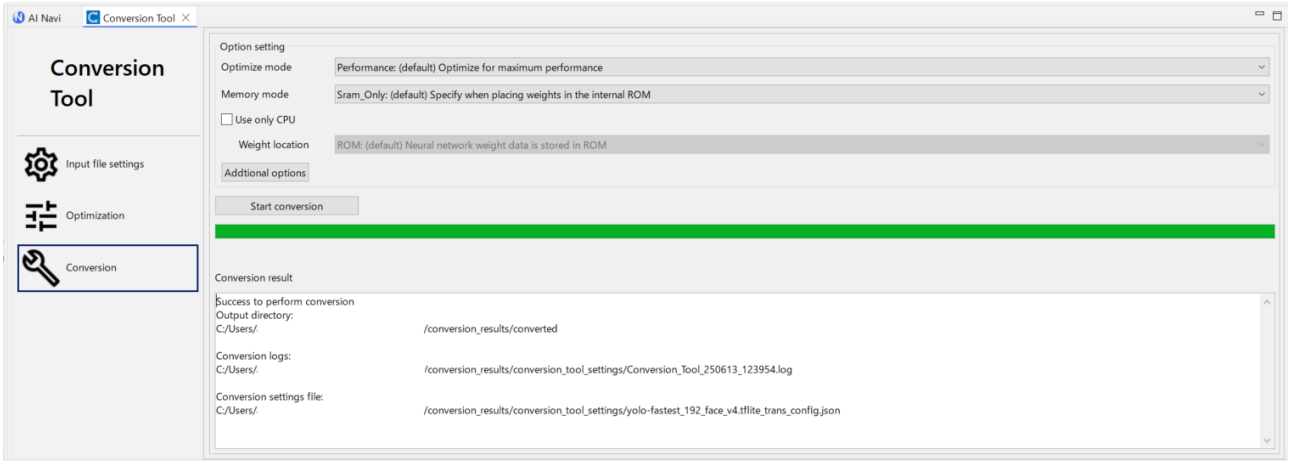

Figure 4.13 Conversion

4.3.4 Edit your application

After conversion, edit your application source code including the implementation of AI model (C/C++), and build your project.

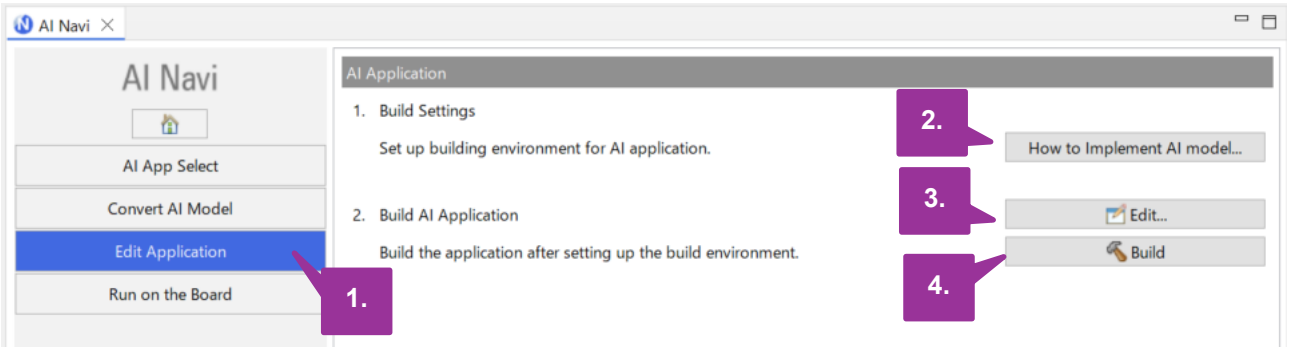

Figure 4.14 Edit Application

1. Click **[Edit Application]** in the AI Navigator menu.
2. Click **[How to Implement AI Model…]** to open “How to Implement AI model”. This document describes the steps from integrating the converted output code via RUHMI Framework to executing AI inference.
3. Modify the source code. When clicking **[Edit…]**, hal_entry.c will open. Integrate the AI model.
4. Click **[Build]**. The build result will be shown in the console.

4.3.5 Run on the Board

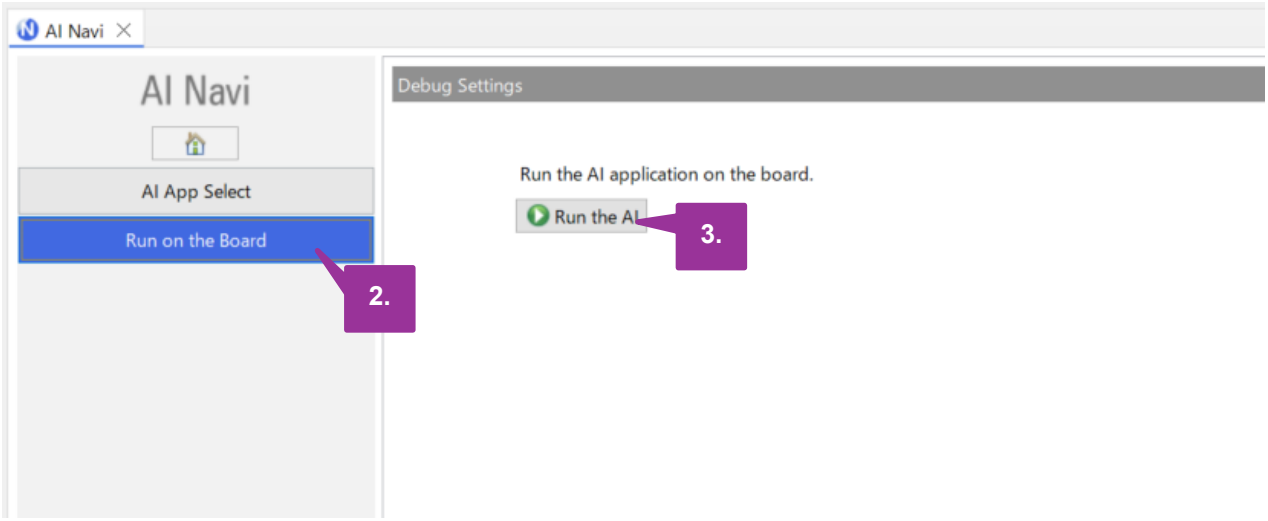

Figure 4.15 Run on the AI

To run your AI Application on the target board,

1. Connect EK-RA8P1.
2. Click **[Run on the Board]** in the AI Navigator menu.
3. In the Debug AI view, click **[Run the AI]** to start inference.

Note: When you try to run an RA8P1 sample AI application via the [Run the AI] button, the build process will start automatically before debugging. To skip this auto build:
1. Open Debug Configurations.
2. Select your project in Renesas GDB Hardware Debugging.
3. Go to Main tab.
4. Choose “Disable auto build” or “Use workspace settings” in Build (if required) before launching.

5. Next Steps

To learn more about the RUHMI Framework, refer to the user's manual and application examples available in RUHMI Github

6. Website and Support

Visit the following URLs to learn about the kit and the RA family of microcontrollers, download tools and documentation, and get support.

Renesas Artificial Intelligence (AI)	renesas.com/ai
AI Navigator	renesas.com/software-tools/ai-navigator
RA Product Information	renesas.com/ra
MCU Evaluation Kit	renesas.com/ra-kits
RA Product Support Forum	renesas.com/ra/forum
Renesas Support	renesas.com/support

Provide Feedback/ Request a Feature

Renesas aims to provide the best microcontroller kit experience to help jumpstart customer innovation with RA, RX & RL78 family of microcontrollers and take products to market faster. The Renesas microcontroller kits have been designed with a lot of attention-to-detail and customer-centric thinking in every aspect of design. Renesas aims to exceed customer expectations.

Renesas looks forward to hearing your feedback and knowing how we can enhance your experience. Please share your feedback at renesas.com/ra/kitfeedback & renesas.com/rx/kitfeedback.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jul. 01 2025	—	Initial release

Renesas RUHMI Framework – Quick Start Guide

Publication Date: July 1st 2025

Published by: Renesas Electronics Corporation

Renesas RUHMI AI Compiler – Quick Start Guide

Renesas Electronics Corporation

R11QS0065EJ0100