RN00543

Ara SDK Release Notes

Rev. 1.0 — 20 August 2026 **Release notes**

Document information

Information	Content
Keywords	RN00543, Ara, Ara240 DNPU, Ara SDK
Abstract	This document provides important information about the Ara SDK and Runtime release 2.1.1.

1 Overview

This document provides important information about the Ara SDK and Runtime release 2.1.1. It includes release highlights, supported features, known issues, and limitations.

2 Features

Ara SDK release 2.1.1 along with Runtime provides the software tools required to compile, optimize, deploy, and run AI models on the Ara240 Discrete Neural Processing Unit (DNPU). The release supports both computer vision and generative AI workloads through the following components:

- CNN Compiler
- GenAI Compiler
- GenAIQuant
- Runtime for Ara240
- Optimum-Ara (GenAI API interface to Runtime)

2.1 CNN compilation

A Convolutional Neural Network (CNN) compilation process converts a trained AI model from frameworks such as PyTorch, TensorFlow, or ONNX into an optimized binary. The binary runs directly on a target hardware accelerator, such as the Ara240 DNPU. The Ara SDK compiler converts trained AI models from PyTorch, TensorFlow, ONNX, and other supported frameworks into optimized binaries that run directly on the Ara240 DNPU.

2.1.1 Compilation pipeline

The Ara SDK compiler transforms a trained model into a hardware-optimized binary through following stages:

1. **Front-end processing**: Converts a model from supported frameworks (PyTorch, TensorFlow, ONNX) into the Ara240 compatible format for internal processing. Performs compatibility checks for supported layers, tensor shapes, and dimensions, preprocesses the network, and generates metadata for subsequent compilation stages.
2. **Quantizer**: Converts FP32 weights and activations to INT8 or INT4 to reduce memory usage and computational requirements. Supports the quantizer, pre-quantized PyTorch, and selective FP32 execution for precision-sensitive layers.
3. **Graph planner**: Schedules layer by layer by tiling the problem and mapping them to NNP cores with the objective of optimizing data movement and overall latency.
4. **Emitter**: Translates the scheduled graph and generates Model DVM, a hardware-optimized representation of the compiled network that can be deployed on Ara240.

Table 1. Release folder contents for CNN compilation and description

File name	Responsibility
`dvrun`	Python script invoking docker tool for compilation of vision models.
`ara_docker`	`ara_docker` folder containing docker file and binaries needed to build the compiler environment.
`sample.yaml`	Sample YAML file for CNN model compilation.

2.1.2 Key features (quantizer)

- Post-conversion bias correction to improve quantized model accuracy.
- Hardware-aware quantization optimized for Ara240 architectures.
- Automatic generation of hardware-consumable quantization parameters, including shift values, scales, and offsets and LUT tables.
- Dual-network architecture with parallel FP32 and fixed-point execution for layer-by-layer accuracy validation.
- Detailed per-layer error analysis and reporting, including MSE, MAP, and SNR metrics.
- Enhanced visibility into quantization trade-offs through accuracy comparison between floating-point and quantized models.

2.2 GenAI compilation

The Ara SDK compiler transforms a generative AI model into a hardware-optimized binary through following stages:

1. **Front-end processing**: Ingests transformer-based models (LLMs, VLMs) from supported sources including Hugging Face, converting them into an internal graph representation with handling for transformer-specific operators.
2. **GenAIQuant**: A dedicated quantization tool purpose-built for large generative AI models. Converts weights and activations to efficient integer representations while preserving output quality. Supports high-accuracy quantization and direct FP32 for precision-sensitive layers.
   Note: *GenAIQuant is not a mandatory requirement for compilation, but highly recommended for best results on Ara240 DNPU.*
3. **Graph planner**: Schedules layer by tiling the problem and mapping them to NNP cores with the objective of optimizing data movement and overall latency.
4. **Emitter**: Generates hardware instructions optimized for autoregressive inference, where each output token depends on all previously generated tokens. Model DVM (binary optimized for execution on Ara240) is a representation of a compiled network that can be deployed on Ara240.

Table 2. Release folder contents for GenAI compilation and its description

File name	Responsibility
`llm_model_gen`	Python script invoking docker tool for compilation of GenAI models.
`ara_docker`	`ara_docker` folder containing docker file and binaries needed to build the compiler environment.
`llm_model_config.yaml`	Sample YAML file for compilation of GenAI model.

2.3 GenAIQuant

GenAIQuant is a specialized quantization tool designed to optimize LLMs and VLMs for deployment on the NXP Ara240 DNPU.

Table 3. Release folder contents for GenAIQuant and its description

File name	Responsibility
GenAIQuant	Python scripts to quantize supported GenAI models. For details, see Section 4.

2.3.1 Key features

GenAIQuant offers a comprehensive suite of capabilities designed to streamline the quantization workflow for generative AI models:

- **Outlier reduction**: QuaRot rotation-based optimization to improve quantization stability by mitigating the impact of extreme activation values.
- **State-of-the-art quantization methods**: Support for GPTQ, eQAT (efficient quantization-aware training), and Modality-Balanced Quantization (MBQ) for vision-language models.
- **Bit allocation**: ShortGPT-based mixed-precision bit allocation to intelligently distribute bit-widths across model layers for optimal performance.
- **Multi-GPU support**: Distributed quantization pipeline via torchrun to accelerate processing across multiple GPUs.
- **Flexible calibration**: Support for custom calibration datasets for both text-only and multimodal models.

2.3.1.1 Supported algorithms for GenAIQuant

- Outlier reduction
  - QuaRot (offline rotations)
- Bit allocation
  - ShortGPT
- Post-training quantization (PTQ)
  - GPTQ
- Quantization-aware training (QAT)
  - LLMs: EQAT (Block-AP + End-to-End)
  - VLMs: Block-AP + knowledge distillation

2.4 Runtime for Ara240

Ara240 Runtime is the software component that enables deployment and execution of AI models on the Ara240 DNPU. It provides the APIs and runtime services required to load compiled models, manage inference execution, communicate with Ara240 devices, and monitor device health and performance.

Ara240 Runtime consists of the below listed components:

- Client library
- Proxy
- Kernel driver

For 2.1.1, the runtime has been integrated in the i.MX Embedded Linux version 6.18.20_2.0.0.

For more information, refer to https://github.com/nxp-imx/rt-sdk-ara2 and Ara SDK page.

2.4.1 Key features

- Multi-client support with session management APIs over unix socket, TCP/IP socket, and Shared Memory (SHM) IPC mechanisms.
- Multi-endpoint support for both PCIe and USB-connected Ara240 devices.
- Built-in load balancing within the Proxy for automatic workload distribution based on device availability and minimal wait time.
- Centralized device DDR memory management handled by Proxy.
- Multi-model execution support for CNN, LLM, and VLM workloads, including parallel and cascaded execution flows.

- Dynamic device power-state management through firmware-controlled power-state switching.
- Host suspend and resume support from Linux user-space environments.
- Integrated fault detection, recovery, and error-handling mechanisms.
- Runtime compatibility validation across Client Library, Proxy, firmware, and deployed models.
- Robust thermal monitoring and failure management capabilities.
- Support for automatic re-execution of failed inference requests.
- Automatic temperature throttling mechanisms to protect devices operating outside defined thermal limits.
- Comprehensive endpoint statistics, health monitoring, and diagnostic reporting.
- Detailed per-inference performance statistics and execution diagnostics.
- Prompt caching support for LLM workloads to improve Time to First Token (TTFT) for repeated prompts.

3 Supported devices

This section provides the supported configurations for the Ara SDK and runtime.

3.1 Host requirements for CNN/LLM compilation

Table 4. Host requirements for CNN/LLM compilation

Platforms	CNN	LLM
Operating system	Linux (Tested: Ubuntu, Cent OS)	Linux (Tested: Ubuntu, Cent OS)
Processor architectures	Intel x86	Intel x86
RAM	>64 GB	>128 GB
Disk space	>=40 GB	>=500 GB
Python	3.12	3.12
Shell	Bash or equivalent	Bash or equivalent
Docker	>=24.x	>=24.x

3.1.1 GenAIQuant memory requirements

- **QuaRot + eQAT with MBQ:** Uses approximately 120 GB of VRAM.
- **QuaRot + GPTQ:** Uses approximately 48 GB of VRAM.

3.2 Supported platforms and modules for Runtime for Ara240

Validated host and Ara240 hardware

- FRDM i.MX 95 and FRDM i.MX 95 Pro boards
- FRDM i.MX 8M Plus
- Ara240 16GB M.2 Module
- Ara240 16GB USB Module (preproduction stage)

Note: *Supported in i.MX Embedded Linux Version: 6.18.20_2.0.0*

4 Supported models

Below is a list of validated models for Ara SDK and Runtime. Some of these models are available in DVM format on Hugging Face.

4.1 GenAI

Table 5. Llama family

Model	Type	Task	Parameters (B)
Llama-2-7B	LLM	Text Generation	7
Llama-3.1-8B-Instruct	LLM	Text Generation	8
Llama-3.2-3B-Instruct	LLM	Text Generation	3

Table 6. Qwen family

Model	Type	Task	Parameters (B)
Qwen2-7B-Instruct	LLM	Text Generation	7
Qwen2.5-3B-Instruct	LLM	Text Generation	3
Qwen2.5-7B-Instruct	LLM	Text Generation	7
Qwen2.5-Coder-1.5B-Instruct	LLM	Text Generation	1.5
Qwen2.5-VL-3B-Instruct	VLM	Image-Text-to-Text	3
Qwen2.5-VL-7B-Instruct	VLM	Image-Text-to-Text	7

4.2 CNN

Table 7. Ultralytics YOLO series (v5, v8, 11, 26)

Model	Task	Parameters (M)
Ultralytics YOLOv8n	Object Detection	3.2
Ultralytics YOLOv8s	Object Detection	11.2
Ultralytics YOLOv8m	Object Detection	25.9
Ultralytics YOLOv8l	Object Detection	43.7
Ultralytics YOLOv8x	Object Detection	68.2
Ultralytics YOLOv8n-seg	Instance Segmentation	3.4
Ultralytics YOLOv8s-seg	Instance Segmentation	11.8
Ultralytics YOLOv8m-seg	Instance Segmentation	27.3
Ultralytics YOLOv8l-seg	Instance Segmentation	46
Ultralytics YOLOv8x-seg	Instance Segmentation	71.8
Ultralytics YOLOv8n-pose	Pose Estimation	3.3
Ultralytics YOLOv8s-pose	Pose Estimation	11.6
Ultralytics YOLOv8m-pose	Pose Estimation	26.4
Ultralytics YOLOv8l-pose	Pose Estimation	44.4
Ultralytics YOLOv8x-pose	Pose Estimation	69.4
Ultralytics YOLO11n	Object Detection	2.6
Ultralytics YOLO11s	Object Detection	9.4
Ultralytics YOLO11m	Object Detection	20.1

Table 7. Ultralytics YOLO series (v5, v8, 11, 26)*...continued*

Model	Task	Parameters (M)
Ultralytics YOLO11l	Object Detection	25.3
Ultralytics YOLO11x	Object Detection	56.9

Table 8. YOLO series (X)

Model	Task	Parameters (M)
YOLOX-Tiny	Object Detection	5.06
YOLOX-s	Object Detection	9.0
YOLOX-m	Object Detection	25.3
YOLOX-l	Object Detection	54.2
YOLOX-x	Object Detection	99.1

Table 9. ResNet series

Model	Task	Parameters (M)
ResNet18	Image Classification	11.69
ResNet18d	Image Classification	11.71
ResNet34	Image Classification	21.8
ResNet34d	Image Classification	21.82
ResNet50-V1	Image Classification	25.5
ResNet50-V2	Image Classification	25.5
ResNet50d	Image Classification	26
ResNet101-v2	Image Classification	44.5
ResNet152-v2	Image Classification	60.1

Table 10. MobileNet series

Model	Task	Parameters (M)
MobileNet-v1	Image Classification	4.24
MobileNet-v2	Image Classification	3.47
MobileNet-v1-SSD	Object Detection	6.85
MobileNet-v2-SSD	Object Detection	16.88

Table 11. EfficientNet series

Model	Task	Parameters (M)
EfficientNet-Lite0	Image Classification	4.7
EfficientNet-Lite1	Image Classification	5.4

Table 11. EfficientNet series...*continued*

Model	Task	Parameters (M)
EfficientNet-Lite2	Image Classification	6.1
EfficientNet-Lite3	Image Classification	8.2
EfficientNet-Lite4	Image Classification	13.0

Table 12. Detection and specialty models

Model	Task	Parameters (M)
SCRFD-500M	Face Detection	0.57
SCRFD-2.5G	Face Detection	0.82
SCRFD-10G	Face Detection	4.23

5 Known issues and limitations

This section summarizes the known limitations for this release.

5.1 CNN compilation

- Pre-quantized networks are not supported.
- Accuracy for dynamic quantization is heavily dependent on the calibration dataset used.

5.2 GenAI compilation

- Hidden size must be a multiple of 64.
- Number of hidden layers ≤ 36.
- Maximum context length ≤ 8192 tokens.
- No speculative decoding compilation support — pre-compiled DVMs providing higher token rates are available for some of the supported models. For details, refer to Section 4.

5.3 GenAIQuant

- The SpinQuant algorithm is currently unsupported.
- Online rotations are not supported for QuaRot.
- QuaRot can introduce degradation on models, which do not exhibit big outliers.

5.4 Runtime for Ara240

Validated environments: This runtime release has been verified under standard operating workloads on primary target devices and single-client configurations.

The following deployment scenarios are outside the validation scope of this release:

- Concurrent multi-model execution
- High-density multi-client stress testing

6 Revision history

Table 13 summarizes the revisions to this document.

Table 13. Revision history

Document ID	Release date	Description
RN00543 v.1.0	20 August 2026	Initial public release

7 Appendix

Note: *This only applies to customers who are upgrading from a previous confidential release of the software to the 2.1.1 release of Ara SDK and runtime.*

7.1 Compiler

The section lists the enhancements as compared to previous confidential release of compilers in Ara SDK.

7.1.1 Enhancements

- Added support for networks with optimized LayerNorm implementations.
- Improved LLM output accuracy.

7.2 Runtime for Ara240

The section lists the enhancements & bug fixes as compared to the previous confidential release of the Runtime for Ara240.

7.2.1 Bug fixes

- Fixed memory allocation issue to address small models being overwritten.
- Fixed stability issues that could occur when disconnecting and reconnecting application sessions.
- Improved runtime stability during sustained inference workloads. During thermal throttling, APP IPS recovery is fixed when the temperature cools down.
- Fixed device idle-time configuration handling by updating the device idle time configuration unit from microsecond to millisecond.

7.2.2 Enhancements

- Updated API header information in `dvapi.h`.
- Updated third-party software dependencies, `yhirose/cpp-httplib` version from v0.30.2 to v0.38.0 used by proxy.

7.2.3 Configuration changes (w.r.t release v2.0.4)

- Renamed Proxy configuration parameter:
  - `dm_mcp_throttle_disable` → `dm_mcp_throt_enable`
    - Default value: 1 (MCP throttling enabled).

  - Set to 0 to disable MCP throttling.
- `dm_mcp_throttle_delay` → `dm_mcp_throt_delay`
  - Specifies the delay, in microseconds, applied before submitting an inference request when MCP throttling is active.
  - Default value: 1000000.
- `device_idle_wait_time_us` → `device_idle_wait_time`
  - `device_idle_wait_time`: 0 wait time before automatically putting the device in low power mode (milliseconds). This value is non-negative. If the config value is < 5, the configuration is disabled. Setting this config (>=5) enables dynamic power switching.

- Updated the default firmware throttling temperature thresholds, as shown in Table 14.

Table 14. Default firmware throttling thresholds

Parameter	Commercial part (°C)	Industrial part (°C)
`Throttle_High`	83	103
`Throttle_High_Ret`	78	98
`Throttle_Low`	5	5
`Throttle_Low_Ret`	10	10

7.3 Optimum-Ara

The section lists the enhancements for Optimum-Ara.

7.3.1 Enhancements

- Added multi-device support for Python applications.
- Reset version numbering to 1.0.0.

Legal information

Definitions

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

Disclaimers

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Trademarks

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

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

Intel, the Intel logo, Intel Core, OpenVINO, and the OpenVINO logo — are trademarks of Intel Corporation or its subsidiaries.

PyTorch, the PyTorch logo and any related marks — are trademarks of The Linux Foundation.

TensorFlow, the TensorFlow logo and any related marks — are trademarks of Google Inc.

Contents