

RN00161

Real-time Edge Software Release Notes

Rev. 3.2 — 29 July 2025

Release notes

Document information

Information	Content
Keywords	RN00161, Real-time Edge software, release notes, Industrial Networking, Real-time System, Industrial Protocols, i.MX, QorIQ, Layerscape, Heterogeneous Multicore Framework, Heterogeneous Multi-SoC Framework, EtherCAT, TSN, High-availability Seamless Redundancy (HSR), NXP hardware platforms
Abstract	This document contains important information about Real-time Edge software contents, supported features, known issues and limitations in this release.



1 Feature introduction

'Real-time Edge Software' is a software suite provided by NXP Semiconductors for the industrial and IoT domains. It offers Real-time System, Heterogeneous Multicore Framework, Heterogeneous multi-SoC Framework, and Industrial Networking components. The software provides comprehensive support for control of IO devices, selection of operating systems with different schedule latencies, and intercore communication. It allows expansion to multiple SoCs, construction of industrial networks, and network redundancy. Real-time Edge software can be used for various use cases in industrial and IoT domains. Factory Automation, Process Automation, Building and Energy, Home Appliances, and EV chargers are some of these applications.

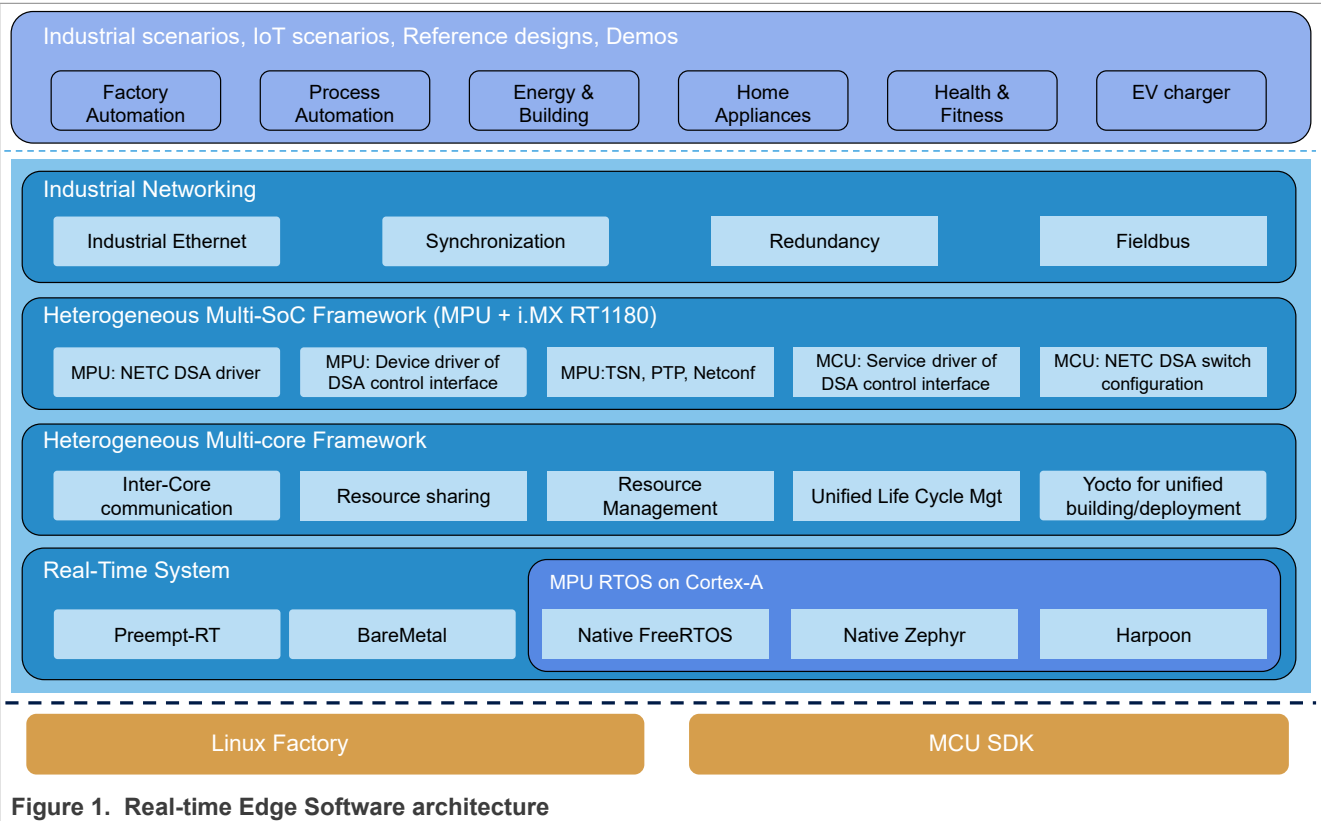


Figure 1. Real-time Edge Software architecture

1.1 Real-time System

For industrial or IoT development, customers first select the appropriate Real-time operating system based on the application scenario. To meet various schedule and latency requirements, the real-time system provides Preempt-RT Linux, native RTOS on Cortex-A core and Cortex-M core, RTOS on Cortex-A core with Jailhouse and BareMetal. It also offers a flexible combination of different cores running Preempt-RT Linux and RTOS/ BareMetal.

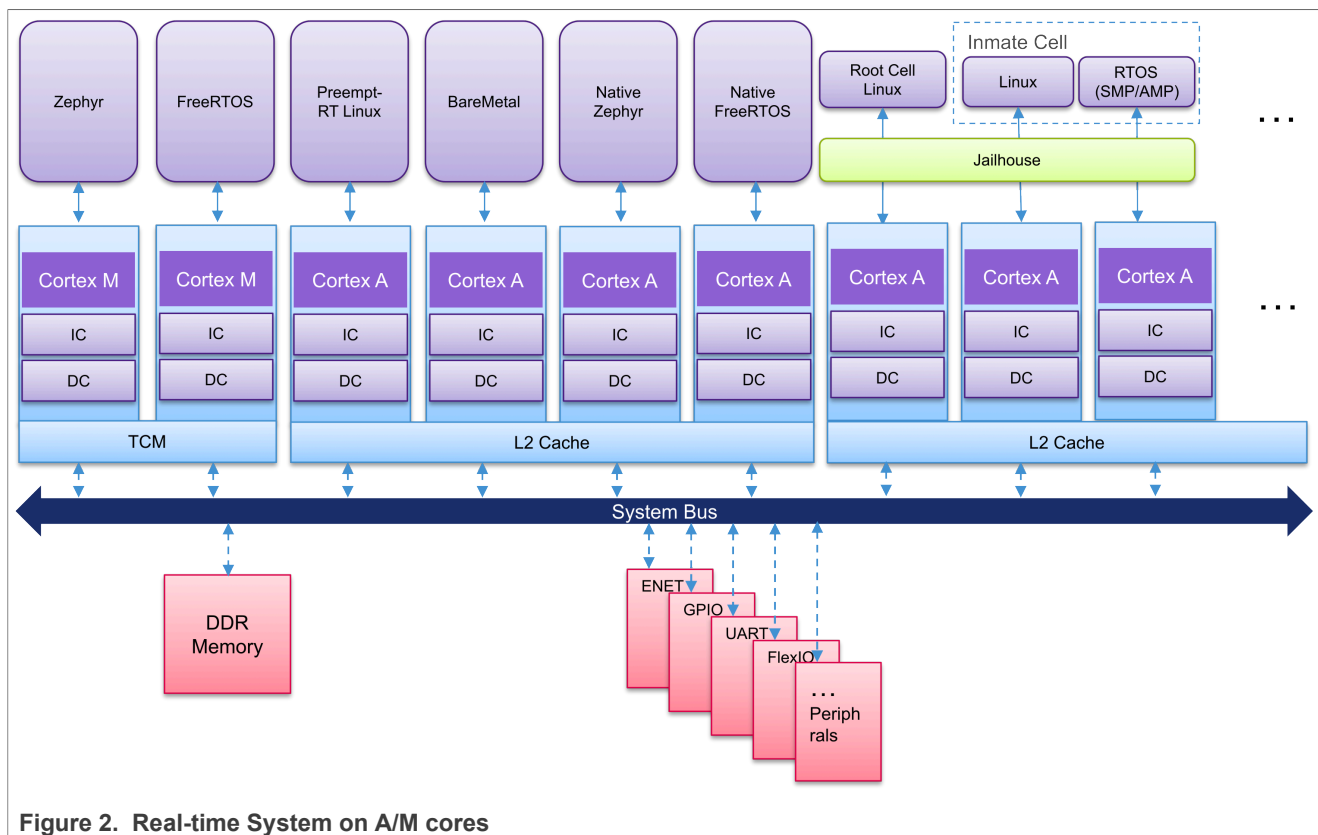


Figure 2. Real-time System on A/M cores

1.1.1 Preempt-RT Linux on Cortex-A core

PREEMPT-RT introduces preemptible and low-latency features to the Linux kernel, transforms Linux into a real-time system. Preempt-RT Linux is suitable for time-critical applications.

1.1.2 BareMetal on Cortex-A core

Real-time BareMetal uses U-Boot to provide a driver-rich BareMetal environment. Single or multiple BareMetal instances run on Cortex-A cores with zero schedule latency.

1.1.3 Native RTOS SMP/AMP on Cortex-A Core

Native RTOS (FreeRTOS or Zephyr) is kicked to one or more Cortex-A Core from U-Boot. Native RTOS does not use Jailhouse and targets lower latency and higher performance as compared to RTOS with Jailhouse.

1.1.4 RTOS SMP/AMP on Cortex-A Core with Jailhouse

RTOS (FreeRTOS or Zephyr) runs in Jailhouse inmate with hardware resource isolation on the Cortex-A core. The Harpoon RTOS provides an environment to develop real-time demanding applications on an RTOS running on one (or several) Cortex-A core(s) in parallel with a Linux distribution. Harpoon applies Jailhouse to partition the hardware and run the RTOS as a Linux guest.

1.1.5 RTOS and Baremetal on Cortex-M Core

Cortex-M cores exhibit deterministic behavior, ensuring consistent and predictable response times when handling real-time tasks. Both RTOS and BareMetal on Cortex-M cores are suitable for deterministic real-time applications.

1.2 Heterogeneous Multicore Framework

When customers choose multiple systems to run on different cores, they encounter problems, such as inter-core communication and resource sharing. Heterogeneous Multicore Framework provides a general software framework to help customers solve these issues. It enables AMP to be interconnected and provides a unified resource management and life-cycle management. It provides the functions illustrated in the [Figure 3](#) to help users accelerate solution development based on multicore platforms.

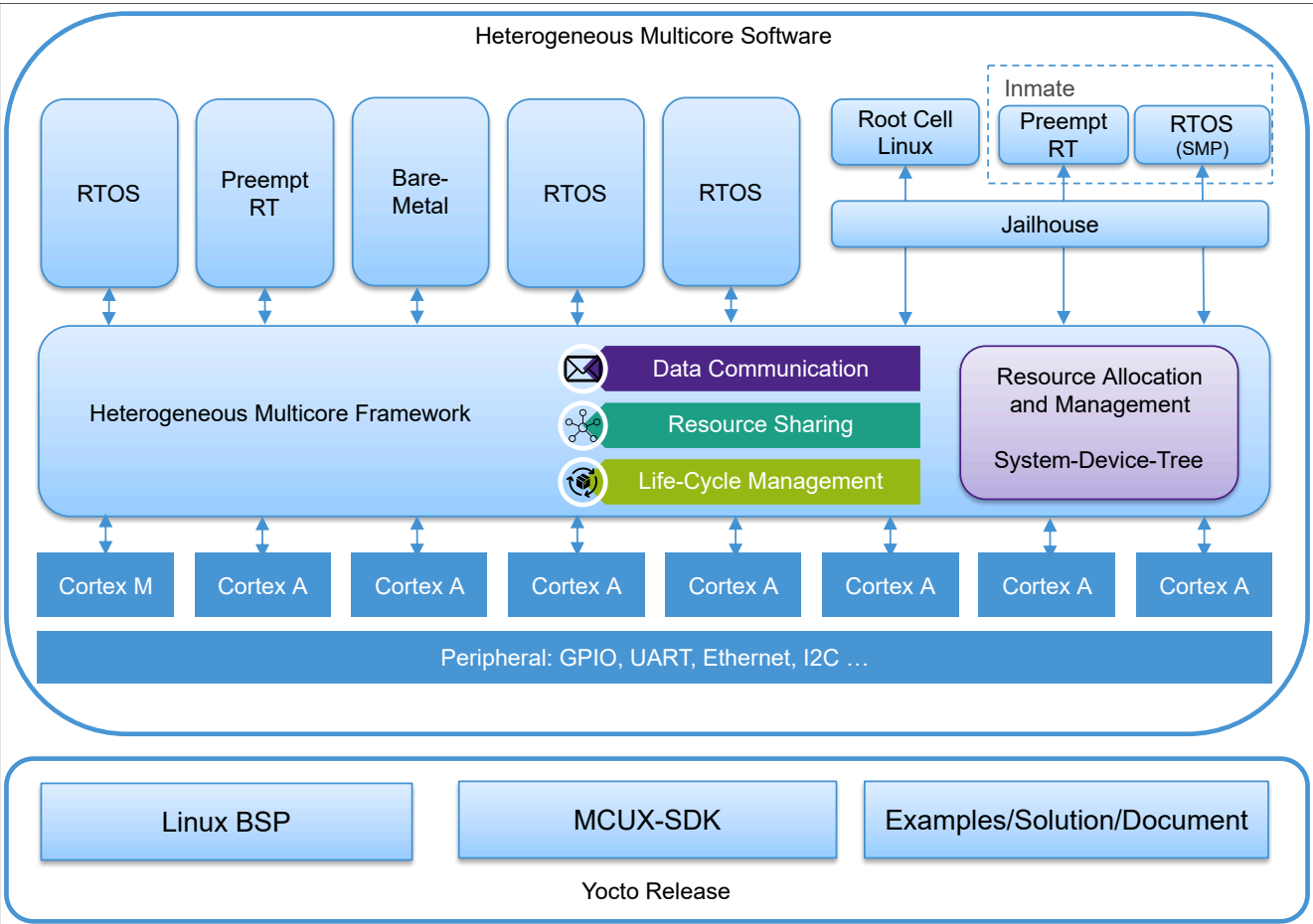


Figure 3. Heterogeneous Multicore Framework

1.2.1 Flexible Real-time System

NXP MPU platforms support Flexible Real-time System. Such a system can run single or multiple RTOS on Cortex-M Core and Cortex-A Core with or without running Linux on Cortex-A core simultaneously. The system provides a RAM console to make it easy to debug multiple OSES in case enough physical UART consoles are not used. The Flexible Real-time System also provides a few common software stacks, such as lwIP networking stack on Cortex-A or Cortex-M cores.

1.2.2 Data communication between different operating systems

The following techniques can be used for data communication between different operating systems. The communication can be between Cortex-M Core and Cortex-A Core, or between different Cortex-A Cores, or between multiple CPU cores simultaneously.

- **RPMSG:** RPMSG is a standard inter-core communication protocol supported on Linux and RTOS.
- **Heterogeneous Multicore VirtIO:** Heterogeneous Multicore VirtIO applies para-virtualization VirtIO technology to build high performance inter-core data path, customized data path is defined according to different use cases.

1.2.3 Resource sharing between different operating systems

Resource sharing enables sharing physical resources between different operating systems. In general, one OS owns and controls the hardware resources while the other OS uses a virtual device. The following mechanism is followed to build the control path and data path to access the physical resource.

- **RPMsg**

Use RPMsg to build control and data path crossing OS, physical resource is shared with another OS in terms of virtual device. The Simplified Real-time Messaging (SRTM) protocol provided in Real-time Edge is an implementation based on RPMSG. It is used to share the physical resources of Cortex-M core with Cortex-A core in terms of virtual device in Linux.

- **Heterogeneous Multicore VirtIO**

Heterogeneous Multicore VirtIO has a better performance than RPMSG, and it can also be used for resource sharing. POSIX compatible API can be used to access virtual device, and some existing VirtIO device drivers in Linux can be reused. Networking sharing is provided in Real-time Edge to share the same networking interface between multiple OSes.

1.2.4 Unified Life-Cycle Management

Heterogeneous Multicore Framework provides Unified Life Cycle Management for both Cortex-A and Cortex-M cores.

1.2.5 Unified SW release and development environment

Separate Yocto meta layers for Cortex-A and Cortex-M cores help define compilation methods with different cross-tool chain. With a single command, all applications on Cortex-A and Cortex-M cores can be compiled and installed into the rootfs.

1.3 Heterogeneous Multi-SoC framework

Using a single SoC has few limitations in certain complex scenarios. In such scenarios, users frequently combine multiple SoCs such as an MPU and MCU to jointly complete the control task. Customers can use the real-time processing capability, industrial protocols, and TSN switch support of MCUs in addition to the high compute power and rich software of MPUs. Heterogeneous Multi-SoC Framework helps customers to implement this support. It flexibly extends an MCU as an MPU's hardware extension.

A typical use case is a combination of i.MX MPU with i.MX RT1180 MCU as the industrial TSN switch. Using this architecture, the i.MX RT1180 MCU acts as an extension that supports not only the switch functionality but also the TSN functionality and different industrial protocols. In this use case, i.MX RT1180 runs real-time tasks such as industrial protocols, in the real-time domain. The MPU takes up the processing of compute-heavy tasks in the non-real-time domain. The external Ethernet ports of i.MX RT1180 MCU are exposed to the MPU side

as standard Ethernet interfaces in the data path. Different interfaces, such as LPSPI or I2C can be used for the control path between MPU and i.MX RT1180.

The Linux Distributed Switch Architecture (DSA) framework is used to expose the i.MX RT1180 NETC switch ports to MPU side. In this architecture, one of the NETC switch ports or ENETC port is used as the data interface. Different interfaces (for example, LPSPI, I2C or message unit) can be used as management interfaces.

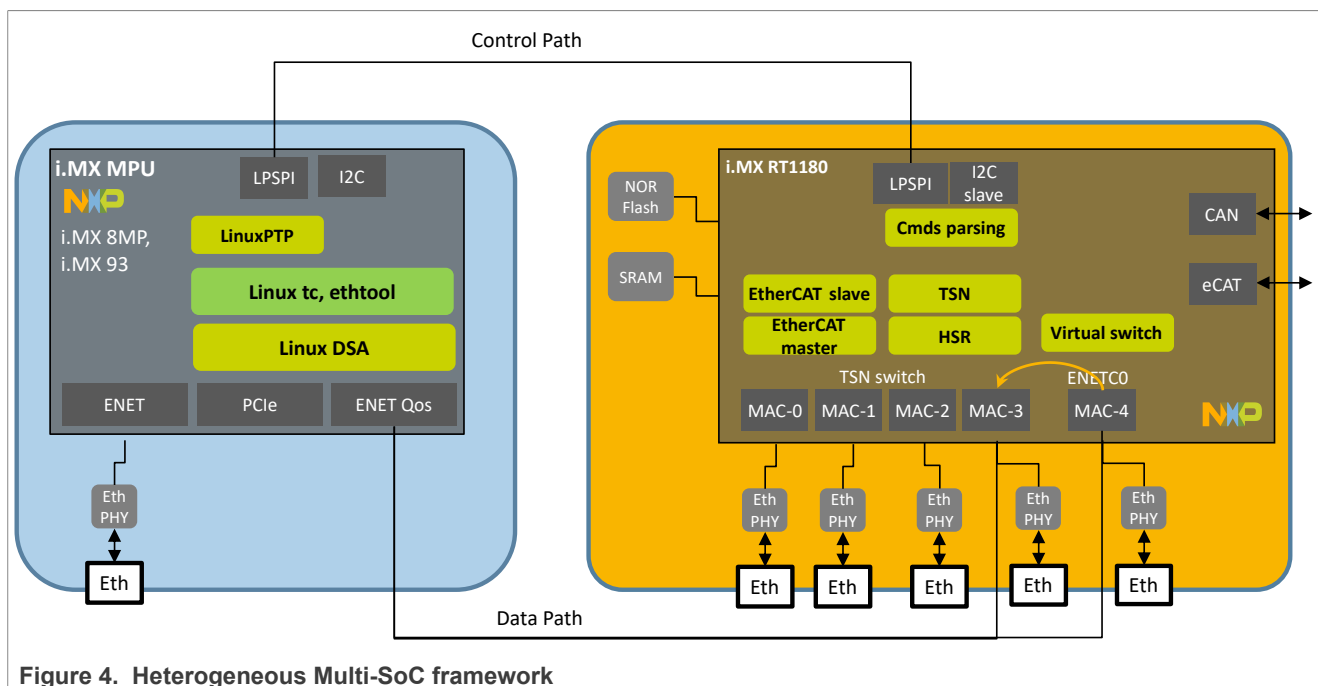


Figure 4. Heterogeneous Multi-SoC framework

- On the MPU Linux side, the standard Ethernet port can use Linux configuration tools such as `ethtool`, `tc` to control the TSN switch port of i.MX RT1180 MCU.
- On the MCU RTOS side, a service driver is added to parse the command to support the Linux configuration tool. The virtual switch forwards packets between the NETC switch and ENETC port0 and the DSA CPU port.

1.4 Industrial networking

1.4.1 EtherCAT

Real-time Edge software provides full stack support for EtherCAT (Ethernet for Control Automation Technology). Real-time Edge integrates IGH and Codesys for EtherCAT master on Cortex-A core and SOEM on Cortex-M core. This feature helps users get rid of the low-level development directly on EtherCAT protocol and provides a common API for real-time applications. i.MX RT1180 MCU supports EtherCAT slave stack.

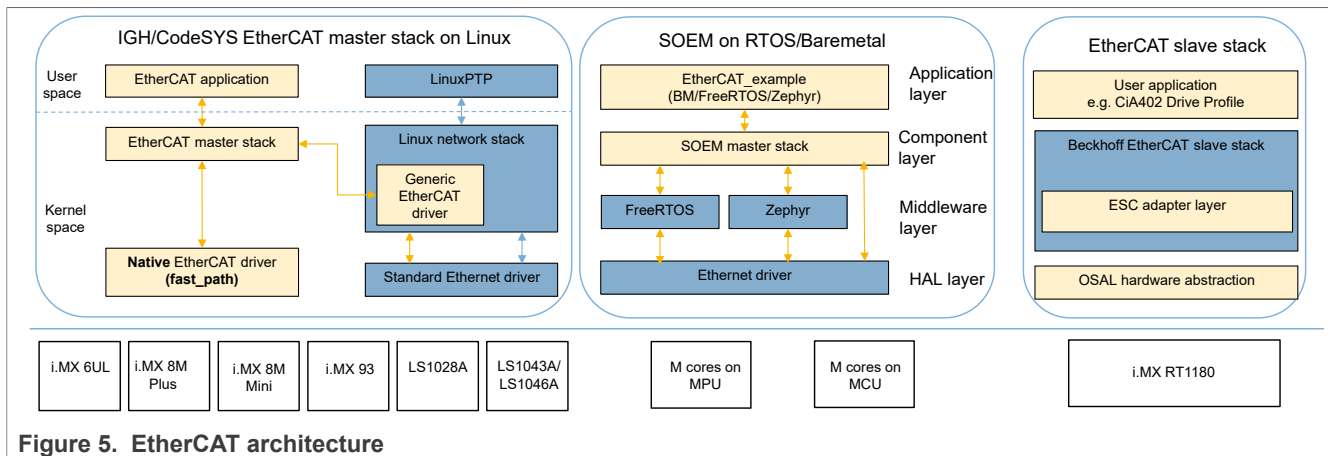


Figure 5. EtherCAT architecture

1.4.1.1 IGH EtherCAT

Real-time Edge integrates IGH EtherCAT master protocols. The Generic driver uses the lower layers of the Linux network stack to connect to the hardware, independent of the actual hardware driver. Therefore, all the hardware platforms that Real-time Edge supports can use the generic driver. However, the performance by using a generic Ethernet Device driver is a bit worse than the native driver.

To improve IGH performance, the native Ethernet Device drivers are added. Native Ethernet Device Drivers allow the EtherCAT master directly and exclusive access to the Ethernet hardware. This implies that the network device must not be connected to the kernel's stack as usual. In Real-time Edge software, there are three native Ethernet drivers:

1. `ec_fec`: the native driver `ec_fec` can be used for the FEC MAC on i.MX 8DXL LPDDR4 EVK, i.MX 8M Mini LPDDR4 EVK, i.MX 8M Plus LPDDR4 EVK, and i.MX 93 EVK.
2. `ec_enetc`: the native driver "`ec_enetc`" is used for ENETC MAC on the LS1028ARDB platform.
3. `ec_dpaa1`: the native driver "`ec_dpaa1`" is used for DPAA1 MAC on the LS1043ARDB and LS1046ARDB.
4. `ec_enetc4`: the native driver "`ec_enetc4`" is used for ENETC4 MAC on the i.MX 95 and i.MX 943 platforms.

1.4.1.2 User Space IGH EtherCAT

The user space IGH EtherCAT stack feature introduced in the Real Time Edge software, further improves performance. The default IGH EtherCAT stack has both code in the kernel space and the user space. In the kernel space, it integrates multiple drivers for different networking devices. In the user space, it supplies a library (`libetherat.so/libetherat.a`) for the user's application. Users can set/get the running parameters of the device using the API of this library. The API reads or writes data via the `ioctl` mechanism of the Linux kernel.

The user space IGH EtherCAT stack runs wholly in the user space. Therefore, it does not need `ioctl` to call the kernel part. Users can invoke the functions of the master module directly. This step can avoid system call, which can decrease the system jitter.

From the perspective of threads, there is only one main thread in the whole system, which avoids context switching. This feature also improves the performance, such as jitter, system latency, and other such parameters.

User space IGH EtherCAT stack has no code that runs in the kernel space, which makes it entirely independent of the kernel version.

User space IGH EtherCAT supports the below reference boards:

- i.MX 8M Mini LPDDR4 EVK

- i.MX 8M Plus LPDDR4 EVK
- i.MX 93 EVK
- i.MX 95 19x19 EVK
- i.MX 943 19x19 EVK

1.4.1.3 Real-time Edge Servo Stack

real-time-edge-servo is a CiA402 (also referred to as DS402) profile framework based on IGH CoE interface. It abstracts the CiA 402 profile and provides an easily usable API for the Application developer.

The real-time-edge-servo project consists of a basic library *libnservo* and several auxiliary tools.

The application developed with *libnservo* offers flexibility to change the CoE network by modifying the *xml* config file, which is loaded when the application starts. The *xml* config file describes the necessary information, which includes EtherCAT network topology, slave and master configurations, and definitions of all the axes.

The stack has been tested on the below CoE servo production: DELTA ASDA-B3, HCFA SV-X6EB, SV-X3EB, Just Motion Control 2HSS458-EC, and INOVANCE InoSV680N.

1.4.1.4 SOEM EtherCAT

Simple Open EtherCAT master (SOEM) is a library that provides user applications a means to send and receive EtherCAT frames. The application provides for:

- Reading and writing process data to be sent/received by SOEM
- Keeping local IO data synchronized with the global IO map
- Detecting errors reported by SOEM
- Managing errors reported by SOEM

On Real-time Edge software, SOEM is supported on Cortex-M core on the below platforms:

- i.MX 8M Plus LPDDR4 EVK
- i.MX 8M Mini LPDDR4 EVK
- i.MX 93 EVK
- i.MX 943 19x19 EVK

SOEM runs on FreeRTOS or without an operating system (baremetal).

There are two SOEM demos provided in this release. Both of them have the same function, but are based on different layers. The demo "freertos_soem_gpio_pulse" is based on FreeRTOS, while the demo "soem-gpio-pulse" is based on bare metal (without an operating system).

1.4.1.5 CODESYS EtherCAT

Based on Real-time Edge yocto project, a new yocto distro named "nxp-real-time-edge-plc" is added, which is specific to the PLC use case.

Features are as follows:

- **Optimized native driver**

Typical industrial software, for example CODESYS or SOEM EtherCAT master stack, work on user space and communicate using Linux standard network interface. To reduce the latency when Ethernet raw packets pass from user space through Linux standard network, the network driver is optimized to avoid memory reallocation and copying, task rescheduling. The latency of the critical path is reduced for packet transmitting and receiving.

- **Light root filesystem**

Lighter root filesystem saves CPU cycles that could be used by the PLC user program.

- **Supported platforms**

- imx6ull14x14evk
- imx8mm-lpddr4-evk
- imx8mp-lpddr4-evk
- imx93evk
- i.MX 95 19x19 EVK
- i.MX 95 15x15 EVK
- i.MX 943 19x19 EVK

1.4.2 TSN

Time Sensitive Networking (TSN) is an extension to traditional Ethernet networks, providing a set of standards compatible with IEEE 802.1 and 802.3. These extensions intend to address the limitations of standard Ethernet in sectors ranging from industrial and automotive applications to live audio and video systems. Applications running over traditional Ethernet must be designed to be very robust in order to withstand corner cases such as packet loss, delay, or even reordering. TSN aims to provide guarantees for deterministic latency and packet loss under congestion. Therefore, it allows critical and non-critical traffic to be converged in the same network.

1.4.2.1 TSN hardware capability

Table 1. TSN hardware capability on different platforms

Platform	802.1Qbv (Enhancements for Scheduled Traffic)	802.1Qbu and 802.3br (Frame Preemption)	802.1Qav (Credit Based Shaper)	802.1AS (Precision Time Protocol)	802.1CB (Frame Replication and Elimination for Reliability)	802.1Qci (Per Stream Filtering and Policing)
ENETC (LS1028 A)	Y	Y	Y	Y	N	Y
Felix switch (LS1028A)	Y	Y	Y	Y	Y	Y
Stmmac (i.MX 8DXL, i.MX 8M Plus, i.MX 93)	Y	Y	Y	Y	N	N
enetc4 - i.MX 943 and i.MX 95	Y	Y	Y	Y	Y	Y

1.4.2.2 TSN configuration

Real-time Edge provides multiple configuration tools to set TSN hardware and network.

The table below describes the TSN configuration tools support on different platforms.

Linux traffic control(tc), ethtool, tsntool are integrated into Real-time Edge to configure TSN device.

Table 2. TSN configuration tool support on different hardware platforms

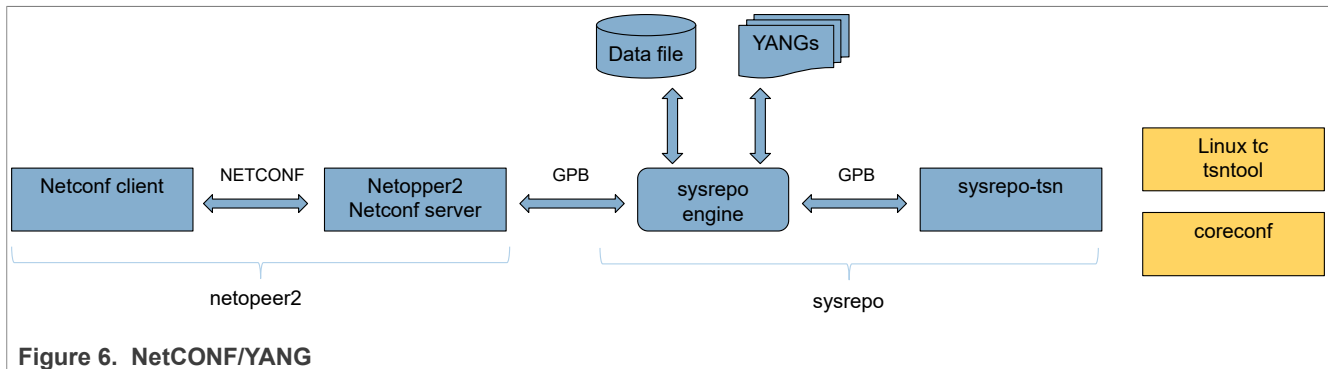
Platform	802.1Qbv (Enhancements for Scheduled Traffic)	802.1Qbu and 802.3br (Frame Preemption)	802.1Qav (Credit Based Shaper)	802.1AS (Precision Time Protocol)	802.1CB (Frame Replication and Elimination for Reliability)	802.1Qci (Per Stream Filtering and Policing)
ENETC (LS1028 A)	tc-taprio tsntool	ethtool tsntool	tc-cbs tsntool	ptp4l	N/A	tc-flower tsntool
Felix switch (LS1028A)	tc-taprio tsntool	ethtool tsntool	tc-cbs tsntool	ptp4l, GenAVB/ TSN stack	tsntool	tc-flower tsntool
Stmmac (i.MX 8DXL, i.MX 8M Plus, i.MX 93)	tc-taprio	ethtool	tc-cbs	ptp4l, GenAVB/ TSN stack	N/A	N/A
enetc4 (i.MX 943 and i.MX 95)	tc-taprio	ethtool	tc-cbs	ptp4l, GenAVB/ TSN stack	N/A,	tsntool, tc-flower

1.4.2.3 Remote configuration using NETCONF/YANG

NETCONF/YANG is also provided for remote configuration.

The NETCONF protocol defines a mechanism for device management and configuration retrieval and modification. It enables a client to adjust to the specific features of any network equipment by using a remote procedure call (RPC) paradigm and a system to expose device (server) capabilities.

YANG is a standards-based, extensible, hierarchical data modeling language. YANG is used to model the configuration and state data used by NETCONF operations, RPCs, and server event notifications.



1.4.2.4 Web-based configuration

The Web UI allows the remote control of the YANG model and also get devices information by websockets.

The user can connect to the http server and input TSN parameters on the web UI. Click "Yes, confirm" button to send the parameters to the board.

The dynamic TSN configuration is used for the TSN configuration dynamically. Users do not need to log into each TSN node to specify the TSN parameters for TSN configuration. They only need to select the path, the base time, and then specify the cycle time. Then, the schedule mapping component calculates the TSN

configuration parameters according to the user input and the path selected. The configuration parameters are delivered to each node by YANG models.

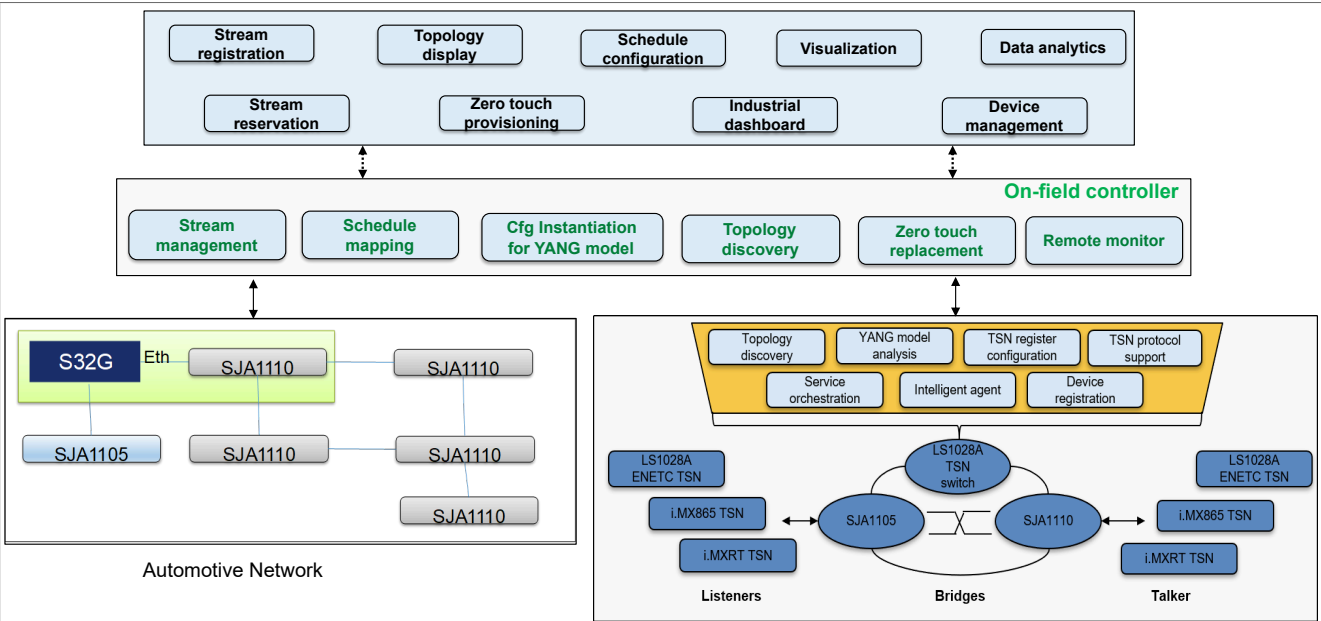


Figure 7. Dynamic TSN network configuration

1.4.3 GenAVB/TSN Stack

The GenAVB/TSN Stack provides advanced implementation for Audio Video Bridging (AVB) and Time-Sensitive Networking (TSN) functionalities on NXP SoCs and hardware platforms, for both Endpoints and Bridges.

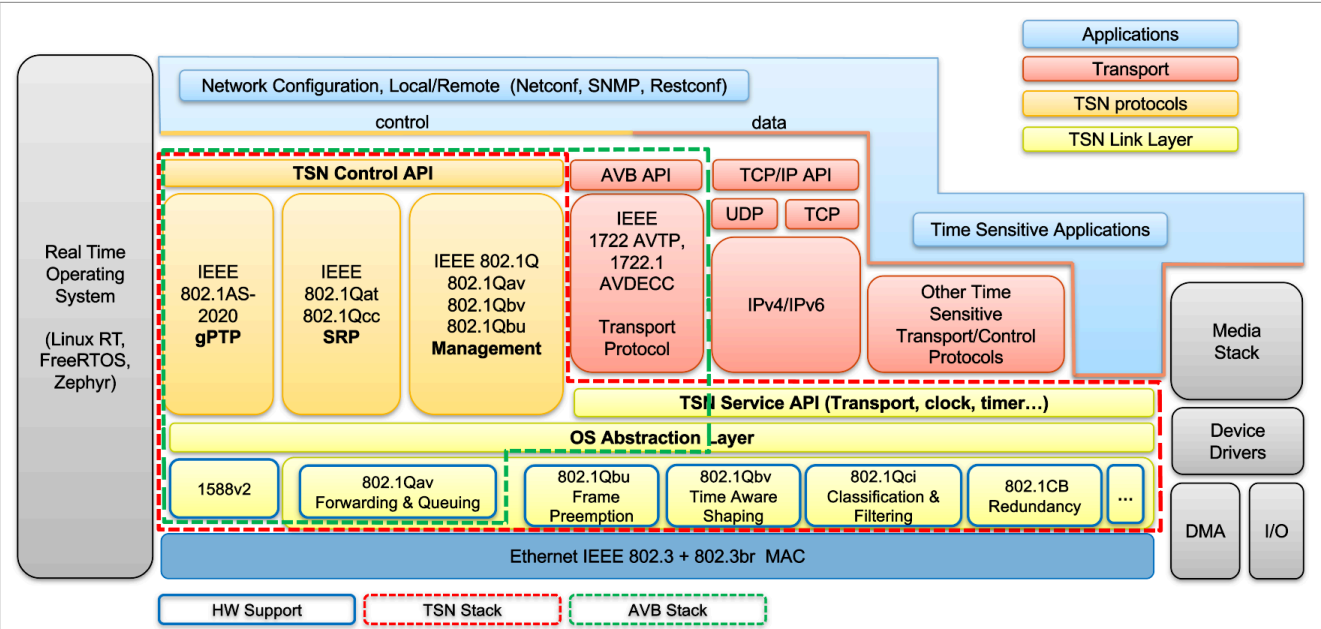


Figure 8. GenAVB/TSN Stack Conceptual Diagram

The following sections provide an overview of supported runtime roles and features.

1.4.3.1 GenAVB/TSN Roles

The GenAVB/TSN stack supports the following roles:

- **AVB Endpoint**

AVB Endpoint functionality is provided in i.MX 93, i.MX 8M Plus, i.MX 8DXL, i.MX 8M Mini, and i.MX 6ULL SoCs (using hardware support if available).

- **AVB Bridge**

AVB Bridge functionality requires AVB hardware support, available in LS1028A SoC and SJA1105Q-EVB (paired with i.MX 93 14x14 EVK or i.MX 8DXL EVK).

- **AVB Hybrid**

AVB Hybrid functionality requires AVB hardware support, available in i.MX 93 14x14 EVK SoC paired with a SJA1105Q-EVB.

- **TSN Endpoint**

TSN Endpoint functionality requires TSN hardware support, available on i.MX 95, i.MX 93, i.MX 943, i.MX 8M Plus, and i.MX 8DXL SoCs.

- **TSN Bridge**

TSN Bridge functionality requires TSN hardware support, available in LS1028A SoC.

1.4.3.2 GenAVB/TSN Protocols

The GenAVB/TSN stack includes the following protocol stacks running in standalone userspace processes.

1.4.3.2.1 gPTP Stack

The gPTP stack implements IEEE 802.1AS-2020 standard, and supports both time-aware Endpoint and Bridge systems. The stack runs fully in userspace, using Linux socket APIs for packet transmit, receive, and timestamping. Linux clock APIs are used for clock adjustment. Configuration files are used to configure the stack at initialization time and extensive logging is available at runtime.

1.4.3.2.2 SRP stack

The SRP stack implements MRP, MVRP, and MSRP defined in IEEE 802.1Q-2022, sections 10, 11, and 35, and supports both Endpoint and Bridge systems. The stack runs fully in userspace, using Linux socket APIs for packet transmit and receive. Linux tc and bridge netlink APIs are used to update Multicast FDB entries and FQTTSS Credit Based Shaper (CBS) configuration. Configuration files are used to configure the stack at initialization time and extensive logging is available at runtime.

1.4.3.2.3 AVTP Stack

The AVTP stack implements IEEE 1722-2016 standard, supporting both AVB Talker/Listener end stations and multiple Audio/Video stream formats. The stack runs in userspace, in combination with a Linux AVB kernel module, providing low latency network packet processing and AVTP packet encapsulation/decapsulation. The stack provides an API for external media applications through a run time library. The API allows external applications to act as sources of AVTP Talker streams/sinks of AVTP Listener streams.

1.4.3.2.4 AVDECC/Milan Stack

The AVDECC stack implements IEEE 1722.1-2013 standard, and supports Talker, Listener and Controller entities. The stack also implements "Milan Specification v1.2", which can be enabled at initialization time. AVDECC entity definitions are loaded from the filesystem and can be created based on a C header file definition. The stack provides an API for external media applications through a run time library.

1.4.3.2.5 MAAP Stack

The MAAP stack implements IEEE 1722-2016, Annex B. The stack provides an API for external media applications through a run time library, but it mainly serves the AVDECC/Milan stack.

1.4.3.3 User Library

The GenAVB/TSN stack provides a public C API through a user space dynamic library that can be linked into customer applications. These API's cover initialization of the stack, run time configuration as well as AVB and TSN endpoint functionality (AVTP Listener/Talker streaming, Layer 2 network sockets, ...).

1.4.3.4 AVB Kernel Module

Integration of AVB functionality in the Linux kernel is provided by a dedicated kernel module. The main role of this module is to add real time guarantees to the processing of AVTP packets.

1.4.3.5 Supported configurations

GenAVB/TSN stack currently supports the following boards and the associated roles:

- LS1028ARDB: gPTP Time-aware Bridge and SRP Bridge
- i.MX 95 19x19 LPDDR5 EVK: gPTP Time-aware Endpoint station and TSN Endpoint.
- i.MX 93 EVK: gPTP Time-aware Endpoint station, TSN Endpoint, and AVB Endpoint stack/applications.
- i.MX 93 9x9 LPDDR4 QSB: gPTP Time-aware Endpoint station and TSN Endpoint.
- i.MX 93 14x14 LPDDR4X EVK: gPTP Time-aware Endpoint station, TSN Endpoint and AVB Endpoint stack/applications. Along with AVB bridge and AVB hybrid stacks when connecting SJA1105Q-EVB Hardware.
- i.MX 8M Plus LPDDR4 EVK: gPTP Time-aware Endpoint station, TSN Endpoint, and AVB Endpoint stack/applications.
- i.MX 8M Mini LPDDR4 EVK: gPTP Time-aware Endpoint station and AVB Endpoint stack/applications.
- i.MX 8DXL LPDDR4 EVK: gPTP Time-aware Endpoint station, TSN Endpoint and AVB Endpoint stack/applications. Along with AVB bridge stack when connecting SJA1105Q-EVB Hardware.
- i.MX 6ULL 14x14 EVK: gPTP Time-aware Endpoint station and AVB Endpoint stack/applications.
- i.MX 943 19x19 LPDDR4/5 EVK: gPTP Time-aware Endpoint station and TSN Endpoint.

1.4.3.6 AVB endpoint example applications

The stack provides extensive example applications for media playback/capture and server. Please refer to “*GenAVB/TSN Stack Evaluation User Guide*” for detailed information.

1.4.3.7 TSN endpoint example application

The TSN example application provides example code and reusable middleware exercising the GenAVB/TSN API. It can be used to exercise and verify the real time behavior of the local system and TSN properties of the network between endpoints. [Figure 9](#) shows the TSN application cycle.

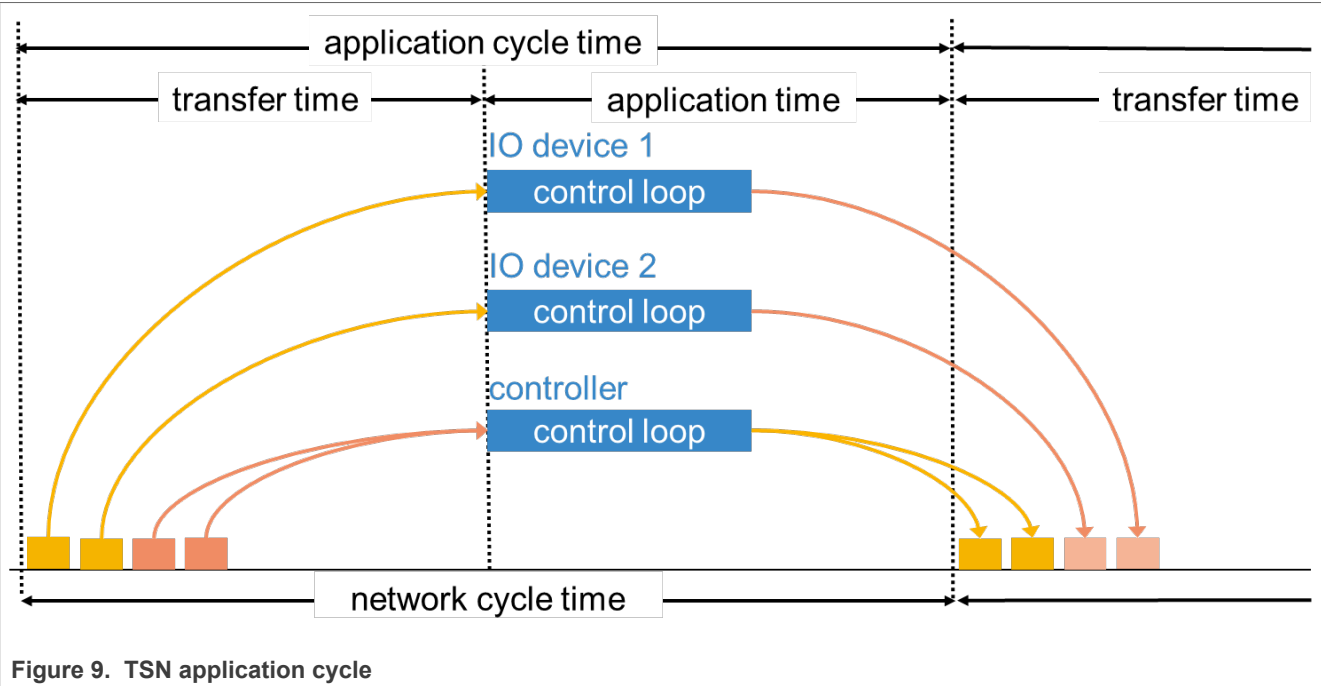


Figure 9. TSN application cycle

The TSN example application implements a control loop similar to industrial use cases requiring cyclic isochronous exchanges over the network.

The TSN endpoints run their application synchronized to a common time grid in the same gPTP domain so that they can send and receive network traffic in a cyclic isochronous pattern (the application cycle time is equal and synchronous to the network cycle time as shown in [Figure 9](#)). Currently the cycle is configured with a period of 2 ms, and periods as low as 1 ms have been confirmed to work as well. When the application is scheduled, frames from other endpoints are ready to be read and at the end of the application time frames are sent to other endpoints.

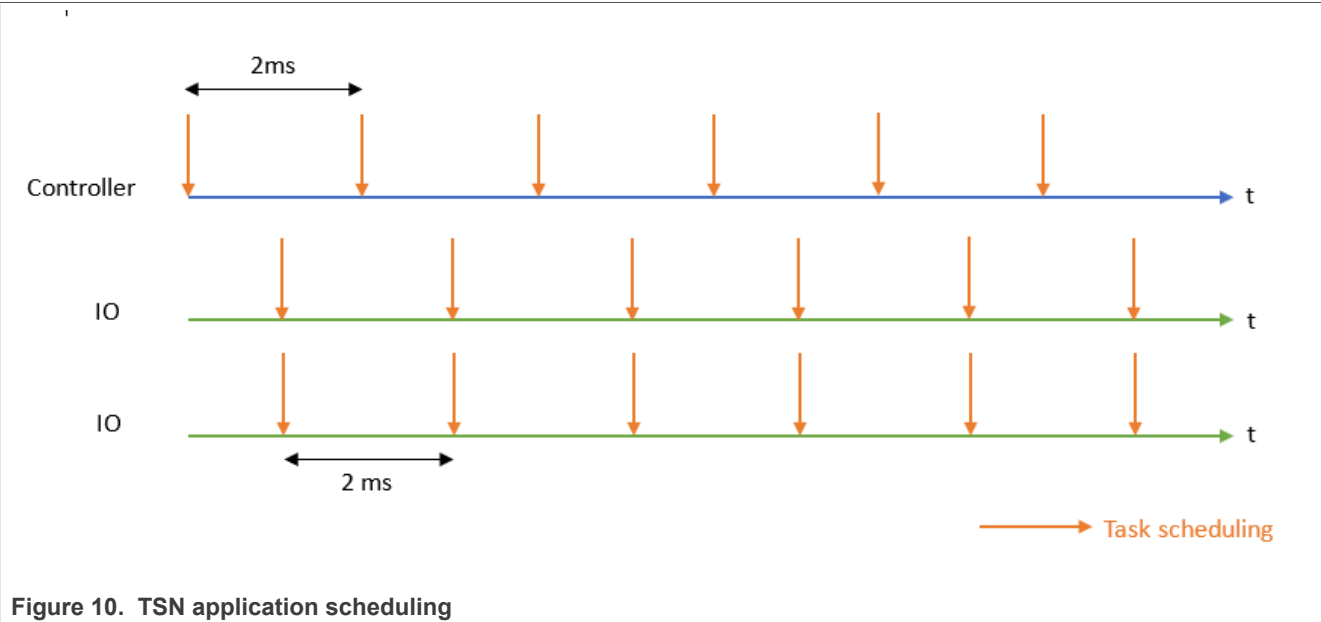


Figure 10. TSN application scheduling

As shown in [Figure 10](#), the controller and the IO devices are scheduled with a half cycle offset to reduce the processing latency.

The time sensitive traffic is layer 2 multicast with VLAN header and proprietary EtherType. Its priority is defined using the PCP field of the VLAN header.

In addition, the TSN application provides detailed logs and time sensitive traffic timing statistics. These are based on hardware timestamping of packets and allow the characterization of an entire real time distributed system.

Finally, a OPCUA server is implemented and offer the possibility to browse and retrieve the TSN application statistics exposed as OPCUA objects. The OPCUA server runs over TCP and allows access to any OPCUA client.

1.4.4 IEEE 1588/802.1AS

IEEE 1588 is the IEEE standard for a precision clock synchronization protocol for networked measurement and control systems. IEEE 802.1AS is the IEEE standard for local and metropolitan area networks – timing and synchronization for time-sensitive applications in bridged local area networks.

NXP's i.MX and Layerscape platforms provide hardware assist for 1588 compliant time stamping with the 1588 timer module. The software components provided to run IEEE 1588/802.1AS protocol utilizing the hardware feature are listed below:

- Linux PTP Hardware Clock (PHC) driver
- Linux Ethernet controller driver with hardware timestamping support
- A software stack application for IEEE 1588/802.1AS

1.4.5 Network redundancy

Real-time Edge supports the below protocols to achieve network redundancy.

Table 3. Network redundancy

Parameter name	STP	RSTP	ERPS	FRER (802.1 CB)	HSR
Recovery time	30-60 s	1-2 s	50 ms	Zero-delay recovery (seamless redundancy)	Zero-delay recovery (seamless redundancy)
Hardware support	No	No	No	Yes	Yes
Network topologies	Ring, mesh	Ring, mesh	Ring	Stream based	Ring
Protocol	IEEE 802.1D	IEEE 802.1w	ITU G.8032	IEEE 802.1CB	IEC 62439-3:2016
Layer 2	Yes	Yes	Yes	Yes	Yes
SoC capability	SoC with Ethernet ports	SoC with Ethernet ports	SoC with Ethernet ports	LS1028A, i.MX RT1180	i.MX RT1180

i.MXRT TSN bridges can be configured to enable High-availability Seamless Redundancy (HSR). It can be used as HSR DANH, REDBOX. Once HSR is enabled, a duplicated stream flows in the opposite direction on the ring. Communication can be maintained by disconnecting any link on the ring. It supports unicast, multicast, Redbox and quadbox for setup/verification. It also supports the mode-H, mode-N, mode-U, mode-T and mode-M HSR modes at runtime.

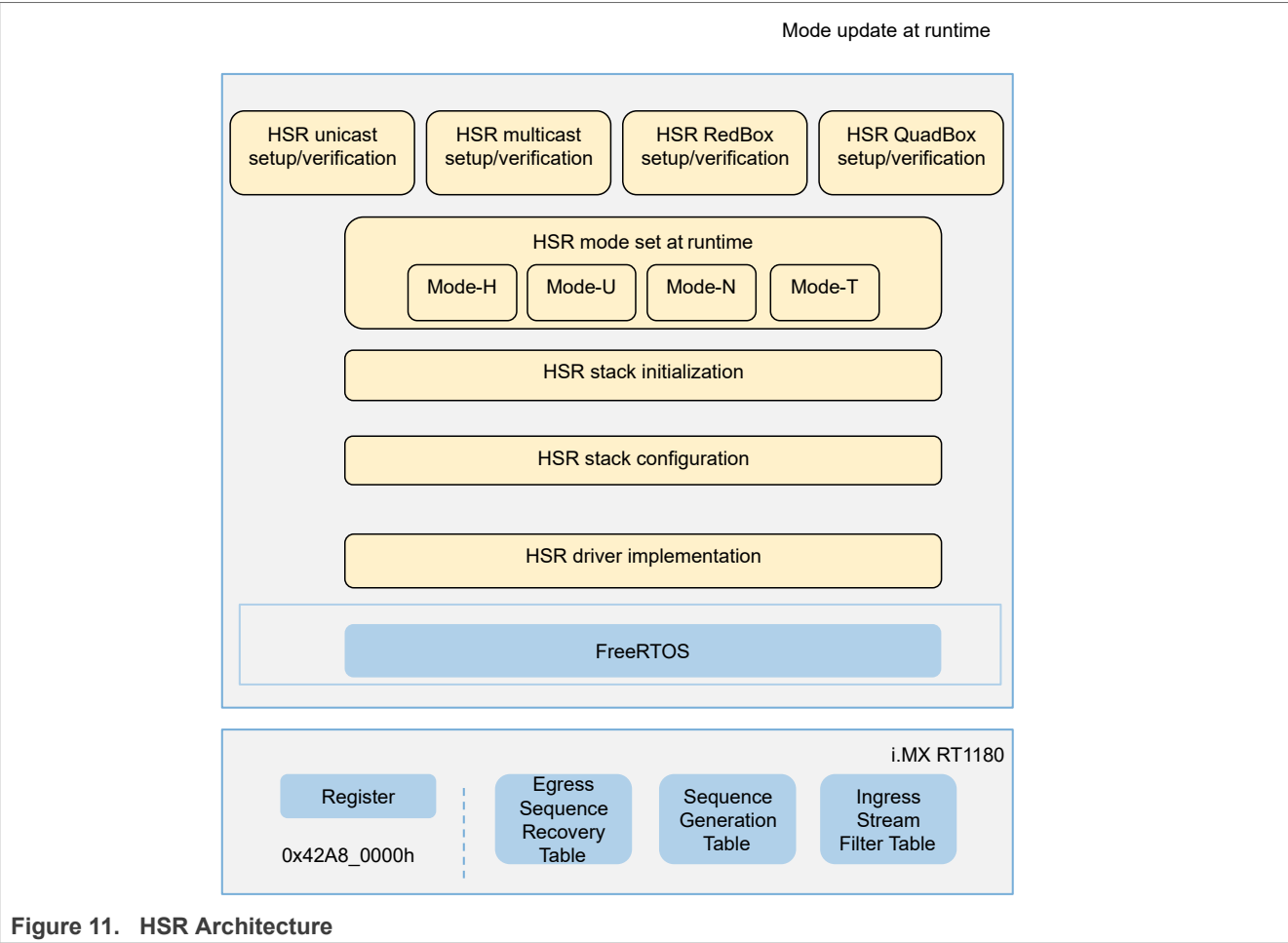


Figure 11. HSR Architecture

Two i.MXRT TSN bridges can be coupled as an HSR Quadbox. The Quadbox can connect two peer HSR rings, which behaves as an HSR node in each ring and is able to filter the traffic and forward it from ring to ring.

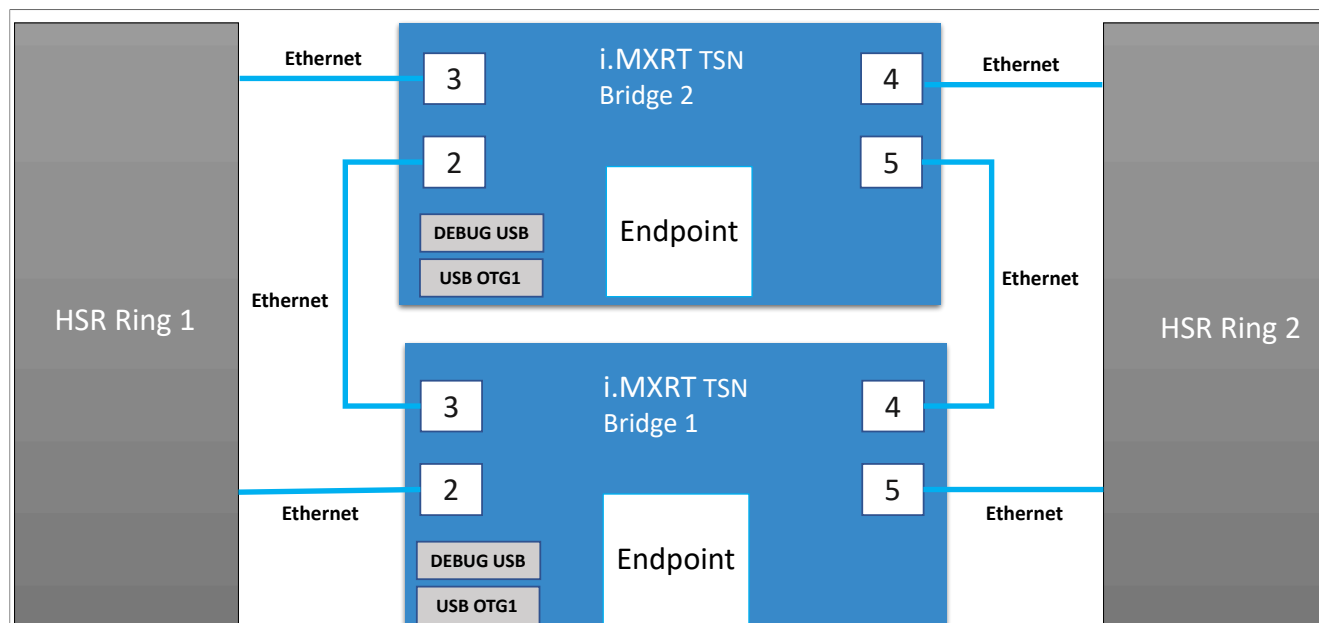
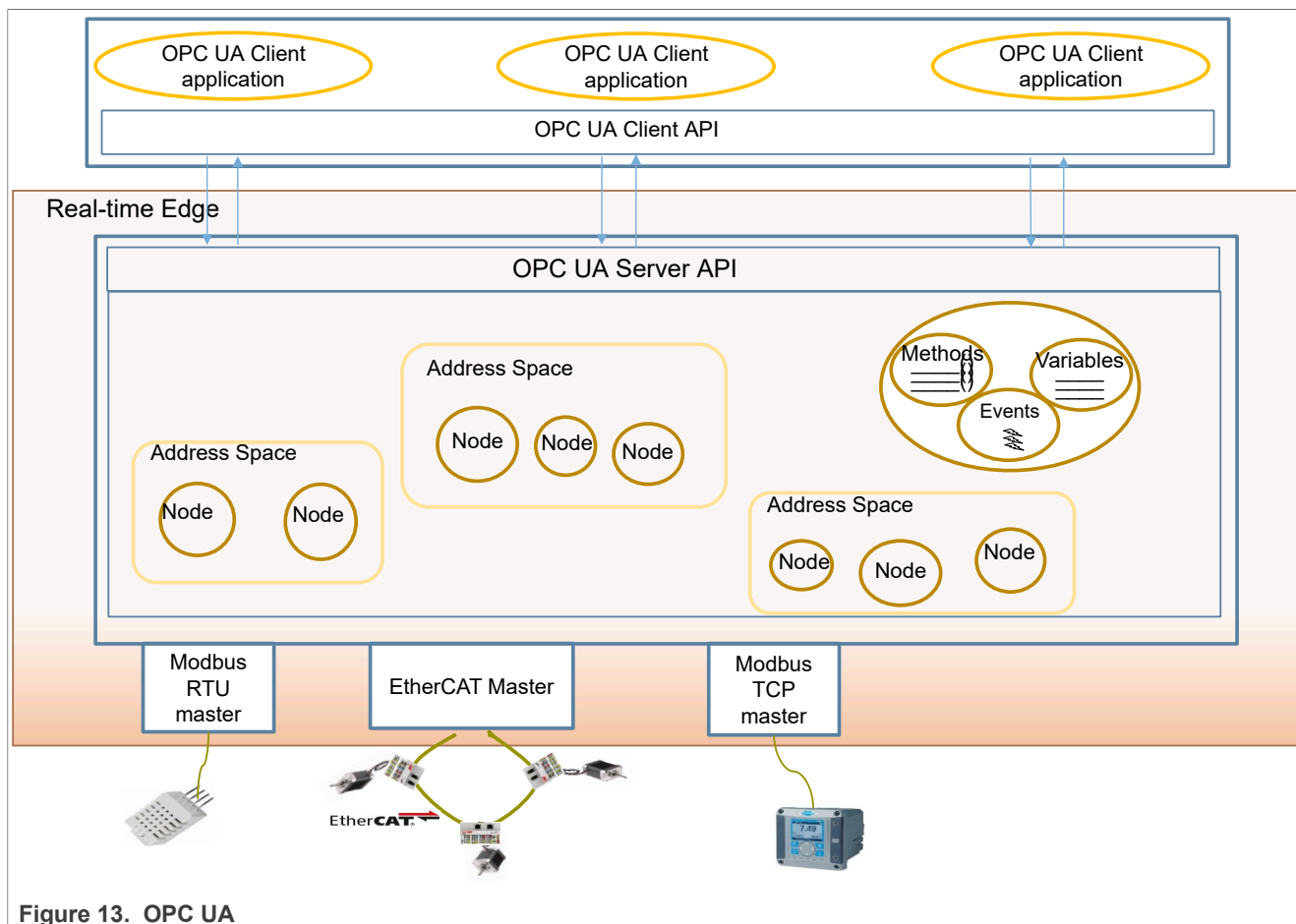


Figure 12. HSR Quadbox formed by coupling two i.MXRT TSN bridges

1.4.6 OPC-UA

OPC UA is a platform-independent standard through which various kinds of systems and devices can communicate by sending request and response Messages between Clients and Servers or Network Messages between Publishers and Subscribers over various types of networks. Multiple samples are provided by Real-time Edge software.

The typical case is two i.MX 8M Plus LPDDR4 EVK boards connected to LS1028ARDB TSN switch.



1.4.6.1 Open62541

Real-time Edge integrates the Open62541 software stack (<https://open62541.org/>). This supports both server-side and client-side API for OPC UA applications. Only server-side capabilities of open62541 are shown here.

Open62541 is distributed as a C-based dynamic library (`libopen62541.so`). The services run on pthreads, and the application code runs inside an event loop.

Enable open62541 in Real-time Edge file `./recipes-nxp/packagegroups/packagegroup-real-time-edge-industrial.bb`:

```
libopen62541 \
```

In order to install Open62541 example application, file `"meta-real-time-edge/conf/distro/include/libopen62541.inc"` is included in the distro configuration.

The following Open62541 example applications are included in the target image:

- `open62541_access_control_client`
- `open62541_access_control_server`
- `open62541_client`
- `open62541_client_async`
- `open62541_client_connect`
- `open62541_client_connectivitycheck_loop`

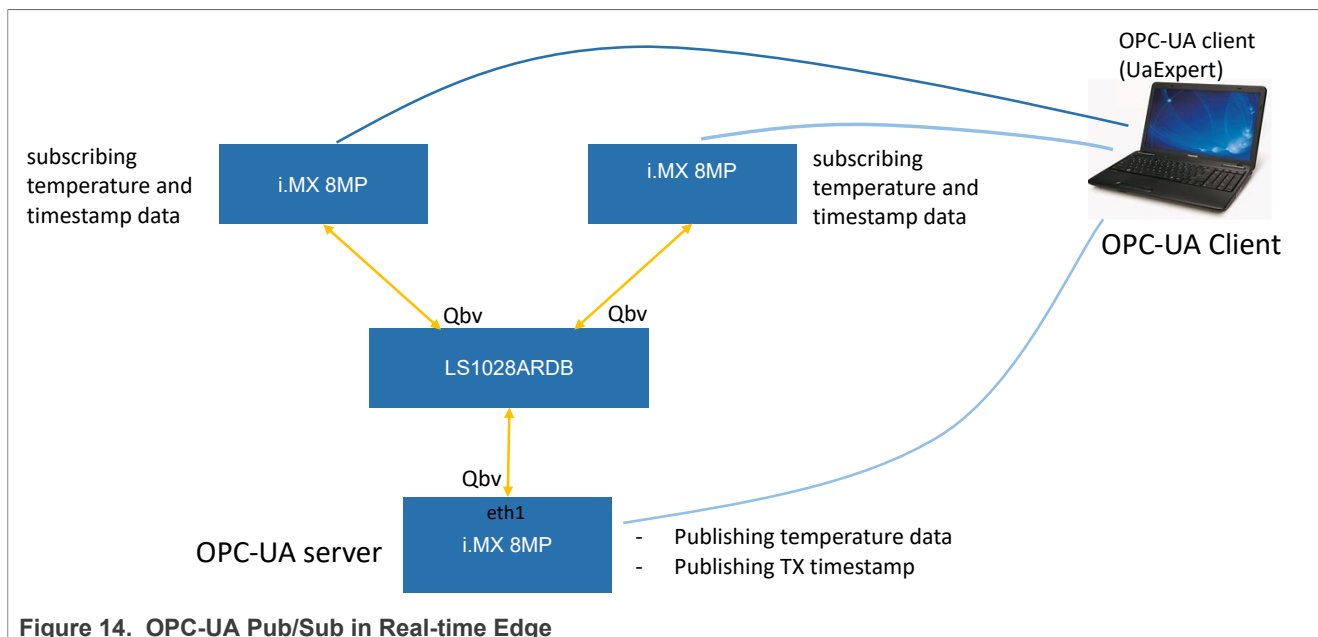
- open62541_client_connect_loop
- open62541_client_subscription_loop
- open62541_custom_datatype_client
- open62541_custom_datatype_server
- open62541_server_ctt
- open62541_server_inheritance
- open62541_server_instantiation
- open62541_server_loglevel
- open62541_server_mainloop
- open62541_server_nodeset
- open62541_server_repeated_job
- open62541_tutorial_client_events
- open62541_tutorial_client_firststeps
- open62541_tutorial_datatypes
- open62541_tutorial_server_datasource
- open62541_tutorial_server_firststeps
- open62541_tutorial_server_method
- open62541_tutorial_server_monitoreditems
- open62541_tutorial_server_object
- open62541_tutorial_server_variable
- open62541_tutorial_server_variabletype

1.4.6.2 OPC UA PubSub over TSN

In general, OPC UA operates at the upper layers of the OSI reference model for networking, whereas TSN is a Layer 2 protocol. TSN adds real-time capability to standard Ethernet. Operating at different layers, TSN and OPC UA PubSub complement each other, yielding a complete communication stack for the IIOT. OPC UA standardizes the protocols that applications use to exchange data and TSN enables this exchange to meet timing requirements of factories.

One of the key things is to define a mechanism for OPC UA nodes to tell the TSN layers how to prioritize data streams. This cross-layer control is essential to enabling operations technology (OT) using the OPC UA framework to get the data they need when they need it. It also enables time-sensitive OT to coexist on the same network as information technology (IT) functions. In this section, standard Linux tools (that is, tc) are used to map packets from different sources to different traffic classes in order to use TSN features like IEEE 802.1AS and IEEE 802.1Qbv.

The performance: 1 ms cycle time, 500 μ s for OPC-UA pub packets, 500 μ s for PTP packets and best effort packets.



1.4.6.3 OPC UA Frame Summation

OPC UA FX (Field eXchange) is a manufacturer-independent communication protocol that extends the core OPC UA functionality for field-level communications. It defines connections using OPC UA PubSub for data exchange, including the TSN usage. In this scenario, decreasing the network latency and increasing bandwidth usage is essential.

The i.MX 943 hardware platform provides the hardware acceleration feature of Frame Modification on its internal TSN switch. This feature allows insertion of one or more blocks of DataSetMessage (an OPC UA PubSub data structure) into a OPC UA PubSub NetworkMessage (a container for DataSetMessage) on the fly when it is passing the switch. This feature benefits the frame summation process, which combines data sets from various device nodes in one Ethernet frame. It not only improves bandwidth usage, but also lowers the network latency.

1.4.7 MODBUS

MODBUS is an application layer messaging protocol positioned at level 7 of the OSI model. It provides client/server communication between devices connected on different types of buses or network.

1.4.7.1 Libmodbus

Real-time Edge integrates libmodbus library. Libmodbus is a free software library used to send or receive data with a device that conforms to the Modbus protocol. It contains various backends to communicate over different networks (for example, serial in RTU mode or Ethernet in TCP IPv4/IPv6).

All the i.MX series and Layerscape series of boards support modbus. It can be used to write both:

- Client applications that reads/writes data from various devices.
- Server applications that provide data to several clients.

1.4.7.2 Libmodbus introduction

Modbus-Simulator is a Modbus tool based on libmodbus library. It contains a modbus client and a modbus device simulator.

Modbus-device-Simulator supports both **TCP** and **RTU** modes, and each mode supports the following functions.

Features supported by TCP:

- Gets CPU temperature of a device
- Gets the status of the LED light of a device
- Modifies the status of the LED light of a device

Features supported by RTU:

- Gets CPU temperature of a device
- Gets the status of the LED light of a device
- Modifies the status of the LED light of a device
- Modifies the slave address of the device
- Modifies the baud rate of the device

Note: Only the i.MX 8M Mini and i.MX 8M Plus boards support LED light function and CPU temperature function, other boards define a dummy variable. By default Modbus-Simulator is disable except i.MX 8M Mini and i.MX 8M Plus, taking I.MX 93 as an example, add the following line to `meta-real-time-edge/conf/distro/include/real-time-edge-base.inc` to enable it.

```
DISTRO_FEATURES:append:mx93-nxp-bsp = " modbus-simulator"
```

1.4.8 FlexCAN and CANopen

The FlexCAN module is a communication controller implementing the CAN protocol according to the CAN 2.0 B protocol specification. The main sub-blocks implemented in the FlexCAN module include an associated memory for storing message buffers, Receive (RX) Global Mask registers, Receive Individual Mask registers, Receive FIFO filters, and Receive FIFO ID filters. The LS1028ARDB board has the FlexCAN module.

1.4.9 UART 9-bit Multidrop mode (RS-485) support

The UART provides a 9-bit mode to facilitate multidrop (RS-485) network communication. When 9-bit RS-485 mode is enabled, UART transmitter can transmit the ninth bit (9th bit) set by TXB8. The UART receiver can differentiate between data frames (9th bit = 0) and address frames (9th bit = 1).

Two examples are provided to demo UART RS485 9-bit multidrop support:

- `9bit-uart-interrupt-transfer` for interrupt mode
- `9bit-uart-polling` for polling mode.

The i.MX 8M Mini LPDDR4 EVK board supports these two demos.

2 Yocto building

You can use Real-time Edge Yocto to build all images including Linux, U-Boot, Rootfs, RTOS, and BareMetal.

Real-time Edge Yocto Project Release directory contains a `sources` directory. This directory contains the recipes used to build one or more build directories, and a set of scripts used to set up the environment.

The following example shows how to download the Real-time Edge recipe layers. For this example, a directory called `yocto-real-time-edge` is created for the project. Any other name can also be specified instead of this name.

```
$ mkdir yocto-real-time-edge
$ cd yocto-real-time-edge
$ repo init -u https://github.com/nxp-real-time-edge-sw/yocto-real-time-edge.git \
-b real-time-edge-walnascar \
-m real-time-edge-3.2.0.xml
$ repo sync
```

When this process is completed, the source code is checked out into the directory `yocto-real-time-edge/sources`. You can perform repo synchronization, with the command `repo sync`, periodically to update to the latest code.

Real-time Edge provides the script `real-time-edge-setup-env.sh`, which simplifies the setup for both i.MX and Layerscape boards. To use the script, the name of the specific machine to be built for and the desired distro must be specified. The script sets up a directory and the configuration files for the specified machine and distro.

Each build folder must be configured in such a way that it uses only one distro. Each time the variable `DISTRO_FEATURES` is changed, a clean build folder is needed. Distro configurations are saved in the `local.conf` file in the `DISTRO` setting and are displayed when the `bitbake` command is run. Here is the list of `DISTRO` configurations:

- `nxp-real-time-edge`: This is the normal image including Real-time and industrial package without BareMetal support.
- `nxp-real-time-edge-baremetal`: This is the BareMetal image (some boards do not support this distro).
- `nxp-real-time-edge-emmc`: This is the normal image to be deployed in eMMC device (for ls1028ardb and ls1046ardb only).

The syntax for the `real-time-edge-setup-env.sh` script is shown below:

```
$ DISTRO=<distro name> MACHINE=<machine name> source real-time-edge-setup-env.sh
-b <build dir>
```

For example:

```
$ DISTRO=nxp-real-time-edge MACHINE=imx8mp-lpddr4-evk source real-time-edge-
setup-env.sh -b build-imx8mpevk-real-time-edge
```

- `DISTRO=<distro configuration name>` is the distro that configures the build environment.
- `MACHINE=<machine configuration name>` is the machine name that points to the configuration file in `conf/machine` in `meta-freescale` and `meta-imx`.
- `-b <build dir>` specifies the name of the build directory created by the `real-time-edge-setup-env.sh` script.

When the script is run, it prompts the user to accept the End User License Agreement (EULA). Once the EULA is accepted, the acceptance is stored in the `local.conf` file within each build folder. The EULA acceptance query is no longer displayed for that build folder.

Yocto Project build uses the `bitbake` command. For example, `bitbake <component>` builds the named component. Each component build has multiple tasks, such as fetching, configuration, compilation, packaging, and deploying to the target rootfs. The `bitbake` image build gathers all the components required by the image and builds in the order of the dependency per task. The first build is the toolchain along with the tools required for the components to build.

The following command is an example of how to build a Real-time Edge image:

```
$ bitbake nxp-image-real-time-edge
```

3 What's new

The following sections describe the new features for each release.

3.1 What's new in Real-time Edge software v3.2

- **Real-time system**
 - Linux PREEMPT_RT 6.12.20
 - RTOS on Cortex-A core
 - Zephyr v4.1
 - Native RTOS Zephyr, FreeRTOS (i.MX 95 19x19 15,15 B0)
 - RTOS with Jailhouse (Harpoon v3.4)
- **Heterogeneous Multicore Framework**
 - Inter-core communication
 - RPMSG communication between Zephyr on Cortex-A core and Linux (i.MX 93, i.MX 8MP)
 - RPMSG communication between FreeRTOS on Cortex-A core and Linux (i.MX 95)
 - RPMSG communication between FreeRTOS on Cortex-A core and FreeRTOS on Cortex-A core (i.MX 93)
 - Benchmarking
 - Refined rt_latency to support run all testcases in one booting
- **Heterogeneous Multi-SoC Framework**
 - LinuxPTP Multi-domain support (i.MX 943, i.MX 93, i.MX 8MP)
 - HSR (i.MX 943, i.MX 93, i.MX 8MP)
- **Industrial Protocols**
 - NETCONF/YANG: plugin mode support
 - EtherCAT Main Device
 - IGH, CODESYS: optimized native driver (i.MX 95, i.MX 943)
 - IGH-User-Space (i.MX 95, i.MX 943)
 - SOEM (i.MX 943)
 - Modbus(TCP/RTU) (i.MX 95, i.MX 943)
 - GenAVB/TSN Milan Network Redundancy (conformance)
 - GenAVB/TSN Endpoint Application with AF_XDP (i.MX 95)
- **NPI Platform**
 - i.MX 943 19x19
 - PREEMPT_RT
 - EtherCAT Main/Sub Devic, Modbus, Digital Encoder
 - Heterogeneous Multi-SoC DSA : TSN, PTP, Qbv, Qbu, Qci, 802.1CB, HSR
 - NETC endpoint: GenAVB/TSN, gPTP multi-domain
 - NETCONF/YANG: LLDP, PTP, TSN YANG model
 - i.MX 95 19x19, 15x15 B0
- **Bases**
 - If-6.12.20-2.0.0
 - Yocto Walnascar
 - Zephyr 4.1

3.2 What's new in Real-time Edge software v3.1

- **Real-time system**
 - RTOS with Jailhouse
 - Harpoon v3.3
- **Heterogeneous Multicore Framework**
 - Unified life cycle management
 - Linux remoteproc auto-boot on A-core
 - Linux RPMSG migrate to remoteproc
 - RPMSG str echo between A-Core Linux and A-Core RTOS on i.MX 93
- **Heterogeneous Multi-SoC Framework**
 - TSN configuration: Qbu on [i.MX RT1180](#)
 - TSN configuration: Qbu on Linux DSA
- **Industrial Protocols**
 - NETCONF/YANG upgrade
 - AVB/TSN
 - GenAVB Milan Network Redundancy (interoperability)
 - GenAVB/TSN IEEE 802.1AS multi-domain time sync on linux virtual PHC on i.MX 95
- **NPI Platform**
 - i.MX 95 (both 19x19 and 15x15 packages) (PRC1)
 - remoteproc support
 - CAN application
 - Real Time latency optimizations
 - i.MX 91 (both 11x11 and 9x9 packages) (PRC)
 - FreeRTOS and Zephyr (hello_world, rt_latency examples)
- **Bases**
 - LF 6.6.36-2.1.0
 - Yocto Scarthgap 5.0
 - MCUX SDK 2.16.0

3.3 What's new in Real-time Edge software v3.0

- **Real-time system**
 - Preempt-RT Linux 6.6.36-rt35 (tag lf-6.6.36-rt-2.1.0)
 - BareMetal (U-Boot 2024.04)
 - RTOS on Cortex-A core
 - Uprev: Zephyr v3.7, FreeRTOS Kernel v11.0.1
 - Native RTOS on i.MX95
 - FreeRTOS and Zephyr with hello_world/rt_latency
 - RTOS with Jailhouse
 - Harpoon v3.2
- **Heterogeneous Multicore Framework**
 - Unified life cycle management
 - Linux remoteproc start/stop
 - U-Boot CPU disable/status
 - Enhanced ram console with dumping tools
- **Heterogeneous Multi-SoC Framework**
 - TSN configuration: CB/Qci/Qbv on [i.MX RT1180](#)

- TSN configuration: CB/Qci/Qbv on Linux DSA
- **Real-time Networking**
 - Linux PTP4L v4.1 (Virtual clocks)
- **Industrial Protocols**
 - AVB/TSN
 - TSN Endpoint on i.MX 95
 - AVB Avnu Milan conformance improvements
- **NPI Platform**
 - i.MX 95 19x19 15x15 (EAR)
 - i.MX 91 11x11 9x9 (EAR)
- **Bases**
 - LF 6.6.36-2.1.0
 - Yocto Scarthgap 5.0
 - MCUX SDK 2.16.0

3.4 What's new in Real-time Edge software v2.9

- **Real-time system**
 - Preempt-RT Linux 6.6.3-rt16 (tag lf-6.6.3-rt-1.0.0)
 - BareMetal - example and driver change to DM mode: DPAA1, DPAA2
 - Harpoon 3.1
- **Heterogeneous Multicore Framework**
 - Network common API on backend for packet processing
- **Heterogeneous Multi-SoC Framework**
 - PTP processing stack on [i.MX RT1180](#)
 - PTP support in Linux DSA
- **Protocols**
 - EtherCAT master
 - SOEM CiA402 profile on M-core [i.MX 8M Mini](#), [i.MX 8M Plus](#) and [i.MX 93](#)
 - SOEM on A core [i.MX 8M Mini](#), [i.MX 8M Plus](#) and [i.MX 93](#)
 - AVB/TSN: GenAVB/TSN stack 6.2
 - AVB Avnu Milan conformance improvements
- **NPI Platform**
 - i.MX93 A1 14*14 EVK
 - – Preempt-RT Linux
 - Heterogeneous Multicore Framework example
 - AVB/TSN
- **Based on lf-6.6.3-1.0.0**
 - U-Boot v2023.04
 - LTS 6.6.3
 - MCUX SDK 2.14.0
 - Yocto Nanbield 4.3

3.5 What's new in Real-time Edge software v2.8

- **Real-time System**
 - Preempt-RT Linux 6.1.55-rt16
 - eMMC boot support on i.MX 8M Mini/ i.MX 8M Plus/ i.MX 93

- Harpoon 3.0
- **Heterogeneous multicore Framework**
 - lwIP on Cortex-A Core (ENET on i.MX 93 and i.MX 8M Mini)
 - RPMSG performance evaluation tools (on i.MX 8M Plus)
 - Native Zephyr on Cortex-A Core
- **Heterogeneous Multi-SoC Framework**
 - Software virtual switching stack on i.MX RT1180
 - NETC DSA switch configuration on both i.MX RT1180 and MPU
 - Using ENETC port as DSA CPU port
- **Protocols**
 - EtherCAT master
 - User space IGH EtherCAT master stack and user space native driver on i.MX 8M Mini, i.MX 8M Plus and i.MX 93.
 - AVB/TSN
 - Bridge and hybrid mode (endpoint + bridge) on i.MX93 14x14 / SJA1105 using external port
- **NPI**
 - i.MX93 A1 14x14 (PRC)
- **Based on If-6.1.55-2.2.0**
 - U-Boot v2023.04
 - LTS 6.1.55
 - MCUX SDK 2.14.0
 - Yocto mickledore 4.2
- **Documentation**
 - Real-time Edge RN (Release Notes)

3.6 What's new in Real-time Edge software v2.7

- **Real-time System**
 - Preempt-RT Linux 6.1.36-rt12
 - BareMetal: math library extended to all platforms
 - Harpoon 2.5
- **Heterogeneous multicore Framework**
 - lwIP on Cortex-A Core (ENET on i.MX8MP)
 - RPMSG between two FreeRTOS
 - RPMSG performance evaluation tools
 - RAM console on FreeRTOS
 - Flexible bootstraps with application
- **Heterogeneous Multi-SoC Framework**
 - NETC DSA switch driver on Linux
 - Device driver of DSA control interface on Linux DSA
 - Service driver of DSA control interface on i.MX RT1180

- NETC DSA switch configuration on i.MX RT1180
- **Protocols**
 - AVB bridge with SJA1105
- **Benchmarking**
 - Heterogeneous Multicore VirtIO performance optimization
- **NPI**
 - i.MX93 A1 9*9
 - Preempt RT, BareMetal, Jailhouse, Heterogeneous Multicore (RPMSG, UART sharing), TSN web-UI configuration
- **Based on If-6.1.36-2.1.0**
 - U-Boot v2023.04
 - LTS 6.1.36
 - MCUX SDK 2.13.1
 - Yocto mickledore 4.2
- **Documentation**
 - Real-time Edge QRG (Quick Reference Guide)

3.7 What's new in Real-time Edge software v2.6

- **Real-time system**
 - Preempt-RT Linux 6.1.22-rt8
 - Baremetal
 - LS1028A
 - Preempt-RT Linux + Baremetal
 - All Cortex-A running under Baremetal
 - Math library support
 - DM mode for Baremetal example and driver: I2C, QSPI
 - Harpoon 2.4.0
- **Heterogeneous multicore framework**
 - VirtIO Ethernet sharing RFP
- **Protocols**
 - EtherCAT master CodeSYS networking optimization
 - i.MX 8M Plus, i.MX 8M Mini, i.MX93, i.MX 6ULL
 - TSN
 - Enhancements for Avnu Alliance Conformance Test for IEEE802.1Qbu/IEEE802.3br
 - AVB Milan 1.1 Test Suite conformance
- **Benchmark**
 - Heterogeneous multicore performance: networking
 - CodeSYS EtherCAT master stack benchmarking
- **NPI**
 - i.MX93 A0 9*9
 - Preempt RT, TSN, TSN stack, and config tool
 - i.MX8DXL: AVB Media Clock Recovery
- **Based on If-6.1.22-2.0.0**
 - U-Boot v2023.04
 - LTS 6.1.22
 - MCUX SDK 2.13.1
 - Yocto mickledore 4. 2

3.8 What's new in Real-time Edge software v2.5

- **Real-time system**
 - Heterogeneous multi-core
 - RPMSG Vring buffer increasing from 256 KB to 8 MB
 - VirtIO network sharing with performance optimization
 - Baremetal improvements on LS1046A
 - All Cortex-A cores running under Baremetal
 - Flextimer
 - Baremetal example and driver change to DM mode: GPIO
 - Integration of Harpoon 2.3
 - Support for AVB Talker in FreeRTOS audio app
 - Support for RPMsg control (FreeRTOS, all boards)
 - Support for Virtual Ethernet
 - Basic support for i.MX 93 ("hello world")
- **Protocols**
 - EtherCAT master
 - Basic CodeSYS PLC control support and native driver optimization
 - i.MX 8M Plus, i.MX 8M Mini, i.MX 93, i.MX 6ULL
- **NPI**
 - i.MX93 A0 11*11
 - Baremetal, RPMSG based UART sharing
 - AVB Media Clock Recovery
 - i.MX8DXL: AVB audio talker/listener
- **Platform**
 - eMMC booting on LS1028ARDB and LS1046ARDB
 - Removal of testing and documentation support, keeping code inclusion for:
 - LS1021AIOT, LS1021ATSN, LS1021ATWR, LS1012ARDB
 - Based on lf-5.15.71-rt-2.2.0
 - LTS 5.15.71
 - Yocto Kirkstone 4.0
 - U-Boot v2022.04

3.9 What's new in Real-time Edge software v2.4

- **Real-time system**
 - Preempt-RT Linux-5.15.52-rt
 - Heterogeneous multi-core
 - Inter-core communication between Cortex-A core and Cortex-A/Cortex-M core on i.MX 8M Plus and i.MX 8M Mini
 - UART 9-bit Multidrop mode (RS-485) support
 - RPMSG between Cortex-A cores
 - Linux SGI mailbox driver on Linux
 - RPMSG Lite with SGI mailbox on RTOS
 - Loading binaries on i.MX 8M Mini and i.MX 8M Plus to the Cortex-M from Linux
 - Baremetal extensions on LS1046A
 - Single hardware interrupt routed to multiple cores
 - Newlib math library

- Integration of Harpoon 2.2
- Audio SMP pipeline (Zephyr)
- RPMsg-based IPC through Linux control application
- audio AVB pipeline (FreeRTOS)
- Support for AVB Listener in FreeRTOS audio app
- **Real-time Networking**
 - TSN
 - Dynamic TSN configuration of Qci for bandwidth limitation
 - Qbu: added preemption TLV on LLDP package and preemption verification support
 - AVB
 - AVB integration improvements
- **Protocols**
 - AVB Milan extensions
 - EtherCAT master stacks
 - EtherCAT master multiple axes control system
 - HMI: LS1028A and i.MX 8M Plus
 - HTML5/chromium
 - Modbus
 - Libmodbus package integration
 - Modbus-simulator client and server
 - WiFi enabled on i.MX 8DXL
- **Reference design**
 - EtherCAT master multiple axes control system
 - HCFA 60-axes servo using CSP mode
- **NPI**
 - i.MX93 A0 11*11: Preempt-RT, EtherCAT master, AVB/TSN, TSN stack and config tools, TSN performance, OPC-UA Pub/Sub
 - i.MX8DXL: Preempt-RT, EtherCAT master, TSN stack and config tools, OPC-UA Pub/Sub
- **Based on If-5.15.52-2.1.0**
 - LTS 5.15.52
 - Yocto Kirkstone 4.0
 - U-boot v2022.04

3.10 What's new in Real-time Edge software v2.3

- **Real-time Networking**
 - TSN
 - Dynamic TSN configuration (EAR)
 - Qci configuration
 - CAF configuration based on 802.1 Qch

- YANG modules updating to latest version
- AVB
 - Endpoint support on i.MX 6ULL, i.MX 8M Plus, and i.MX 8M Mini
- **Real-time System**
 - PREEMPT-RT Linux-5.15.5-rt22
 - Heterogeneous AMP software
 - Yocto based unified delivery for Cortex-A and Cortex-M
 - Resource sharing
 - RPMSG-based UART sharing
 - Virtual UART to physical UART 1:1 mapping
 - Virtual UART to physical UART n:1 mapping
 - Virtual UART to physical UART flexible mapping
 - Harpoon (RTOS on Cortex-A)
 - Zephyr integration on i.MX 8M Plus and i.MX 8M Mini
 - Audio Application
 - Sine wave playback
 - Record and playback (loopback)
 - Audio pipeline
 - Industrial applications:
 - TSN over Ethernet test application
 - CAN test application
- **Protocols**
 - EtherCAT master stack
 - IGH EtherCAT master native driver on LS1043A and LS1046A
 - Multiple EtherCAT masters
 - Flexible port selection for EtherCAT and Ethernet
 - SOEM EtherCAT master stack enablement (PRC):
 - RTOS on Cortex-M on i.MX 8M Plus
 - RTOS on Cortex-M on i.MX 8M Mini
 - FreeRTOS or without an operating system
- **Benchmark**
 - Scheduling latency on Preempt_RT and Harpoon RTOS
 - Inter-core communication bandwidth of Baremetal
 - Packet processing time of TSN
 - Packet processing time of EtherCAT
- **Based on If-5.15.5-1.0.0**
 - Linux 5.15.5-rt22
 - U-Boot v2021.04
 - Yocto Honister 3.4

3.11 What's new in Real-time Edge software v2.2

- **Real-time Networking**
 - TSN
 - 802.1AS: PHY delay correction calibration
 - AF_XDP performance improvements

- IEEE 1588 PTP UDP on LS1028ARDB TSN switch
- Real-time system
 - PREEMPT-RT Linux-5.10.72-rt53
 - Harpoon (RTOS on Cortex-A)
 - Integration of Harpoon on i.MX 8M Plus and i.MX 8M Mini
- Protocols
 - EtherCAT master stack
 - IGH EtherCAT master native driver on LS1043A and LS1046A
 - Multiple EtherCAT masters
 - Flexible port selection for EtherCAT and Ethernet
 - SOEM EtherCAT master stack enablement (EAR):
 - RTOS on Cortex-M on i.MX 8M Plus
 - FreeRTOS
 - or without an operating system
- Based on If-5.10.72-2.2.0
 - Linux 5.10.72-rt
 - U-Boot v2021.04
 - Yocto Hardknott 3.3

3.12 What's new in Real-time Edge software v2.1

What's New:

- Real-time Networking
 - TSN
 - 802.1AS-2020
 - CMLDS (generic interface to PTP stack)
 - TSN application
 - TSN application with AF_XDP data path
 - TSN configuration
 - Path selection for Qbv
 - Schedule mapping for Qbv
- Real-time system
 - PREEMPT-RT Linux-5.10.52-rt47
 - Jailhouse
 - GPIO in non-root cell Linux support on LS1028ARDB
 - ENETC in non-root cell Linux support on LS1028ARDB
- Protocols
 - Native EtherCAT-capable network driver module on ENETC (LS1028ARDB)
 - Native EtherCAT-capable network driver module on FEC (i.MX 8M Plus EVK)
 - EtherCAT: CoE 6-8 axis control
 - OPC UA PubSub
 - OPC UA PubSub over TSN
- Based on i.MX L5.10.52_2.1.0
 - Linux 5.10.52-rt
 - U-Boot v2021.04
 - Yocto Hardknott 3.3

3.13 What's new in Real-time Edge software v2.0

- **Based on Yocto project 3.2 (Gatesgarth)**
- **Real-time System**
 - PREEMPT-RT Linux
 - Heterogeneous architecture
 - Baremetal: PREEMPT-RT Linux on A core + Baremetal architecture on A core
 - i.MX 8M Plus EVK, i.MX 8M Mini EVK, LS1028ARDB, LS1046ARDB, LS1043ARDB, LS1021A-IoT
 - Jailhouse: PREEMPT-RT Linux on A core + Jailhouse + PREEMPT-RT Linux on A core
 - i.MX 8M Plus EVK, LS1028ARDB, LS1046ARDB
- **Real-time Networking**
 - TSN
 - TSN Standards
 - IEEE 802.1Qav
 - IEEE 802.1Qbv
 - IEEE 802.1Qbu
 - IEEE 802.1Qci
 - IEEE 802.1CB
 - IEEE 802.1AS-2020 (gPTP)
 - IEEE 802.1Qat-2010 (SRP)
 - TSN Configurations
 - Linux tc command and tsntool
 - NETCONF/YANG
 - Dynamic TSN configuration - web-based TSN configuration, dynamic topology discovery
 - TSN Applications
 - Example for real-time traffic processing
 - Networking
 - 802.1 Q-in-Q
 - VCAP tc flower chain mode
 - Priority set, VLAN tag push/pop/modify, Policer Burst and Rate Configuration, drop/trap/redirect
- **Industrial**
 - EtherCAT master
 - IGH EtherCAT master stack
 - Native EtherCAT-capable network driver module (i.MX 8M Mini EVK)
 - FlexCAN
 - SocketCAN on Linux kernel
 - CANOpen
 - CANOpen master and slave example code
 - CoE: CANOpen over EtherCAT
 - CiA402(DS402) profile framework based on IGH CoE interface
 - EtherCAT CoE 6-8 axis control (i.MX 8M Mini EVK)
 - OPC UA/OPC UA PubSub
 - open62541
 - Modbus
 - Modbus master and slave
 - Modbus-RTU
 - Modbus-TCP

- Modbus-ASCII
- **New Added Platform**
 - i.MX 6ULL EVK

3.14 What's new in OpenIL v1.11

What's New:

- **TSN**
 - 802.1AS-2020
 - Initial support for multi-domain on i.MX 8M Plus and LS1028A
- **Hardware**
 - i.MX 8M Plus silicon A1
- **Linux Kernel**
 - LTS 5.4.70 for i.MX 8 Series
- **U-Boot**
 - v2020.04 for i.MX 8 Series
- **Baremetal**
 - v2020.04 for Layerscape and i.MX 8 Series
 - i.MX 8M Plus EVK

3.15 What's new in OpenIL v1.10

What's New:

- **TSN**
 - VCAP chain mode
 - GenAVB/TSN stack
- **Real-time**
 - PREEMPT-RT 5.4 on i.MX 8M Mini
 - Ethernet
 - PCIe
 - GPIO
 - DSI
- **Baremetal**
 - i.MX 8M Mini EVK (A core to A core)
 - ICC
 - Ethernet
 - GPIO
- **OpenIL framework**
 - Board
 - i.MX 8M Mini platform
 - GPU: OpenGL ES
 - Display: OpenGL ES, Weston, DSI-MIPI, CSI-MIPI

3.16 What's new in OpenIL v1.9

What's New:

- **TSN**
 - tc flower support for Qbu and Qci
 - 802.1 QinQ
 - Multi-ports TSN switch solution
 - i.MX 8M Plus - TSN
- **Real-time**
 - PREEMPT-RT 5.4 on i.MX 8M Plus
- **Baremetal**
 - LX2160ARDB rev2 support and ICC
- **OpenIL framework**
 - linuxptp uprev to 3.0
 - Board
 - i.MX 8M Plus EVK
 - TSN: Qbv, Qbu, Qav
 - GPU: OpenGL ES, OpenCL
 - Display: OpenGL ES, Weston
 - LS1028ARDB
 - Display: OpenGL ES, Weston
 - GPU: OpenGL ES, OpenCL
 - LX2160ARDB Rev2

3.17 What's new in OpenIL v1.8

What's New:

- TSN
 - tc VCAP support for VLAN-retagging
 - tc VCAP support for police
 - tc support for Qav and Qbv
 - SJA1105 DSA Support and clock synchronization
 - YANG modules for network config (IP, MAC, and VLAN)
- Real time
 - PREEMPT-RT 5.4
- Baremetal
 - LX2160A rev1 ICC
- OpenIL framework
 - buildroot uprev to 2020.02
 - Kernel/U-Boot
 - Linux upgraded to LSDK20.04 - Linux-5.4.3
 - U-Boot upgraded to LSDK20.04 - U-Boot 2019.10
 - Board
 - i.MX 8M Mini
 - Foxconn LS1028ATSN board with SJA1105

3.18 What's new in OpenIL v1.7

What's New:

- TSN
 - BC-based 802.1AS bridge mode
 - Netoppper2 support based on sysrepo. Support Qbv, Qbu, Qci configuration
 - VLAN-based tc flower policer
 - Web-based TSN configuration tool - available for Qbv, Qbu, and Qci configuration
- Real time
 - Xenomai
 - Xenomai l-pipe uprev to 4.19
 - Baremetal
 - SAI support on LS1028
 - i.MX6Q Baremetal ICC
- Industrial protocols
 - CANopen over EtherCAT
- OpenIL framework
 - Kernel/U-Boot
 - Linux upgraded to LSDK1909 - 4.19
 - U-Boot upgraded to U-Boot-2019.04
 - Boards
 - LX2160ARDB SD boot
 - LX2160ARDB XSPI boot
 - LS1028ARDB XSPI boot
 - LS1046ARDB eMMC boot

3.19 What's new in OpenIL v1.6

What's New:

- TSN
 - Web-based TSN configuration tool - available for Qbv and Qbu configuration
 - TSN driver enhancement
- Real time
 - Baremetal
 - i.MX6Q-sabresd Baremetal support
- NETCONF/YANG
 - NETCONF/YANG model for Qbu and Qci protocol
- Industrial protocols
 - LS1028A - BEE click board

3.20 What's new in OpenIL v1.5

What's New:

- TSN
 - Web-based TSN configuration tool - available for Qbv and Qbu configuration

- 802.1AS endpoint mode for LS1028A TSN switch
- Real time
 - Xenomai
 - LS1028 ENETC Xenomai RTNET support
 - Baremetal
 - LS1028 Baremetal ENETC support
- NETCONF/YANG
 - NETCONF/YANG model for Qbv protocol
- Industrial protocols
 - LS1028A - BLE click board

3.21 What's new in OpenIL v1.4

What's New:

- TSN
 - ENETC TSN driver: Qbv, Qbu, Qci, Qav
 - ENETC 1588 two steps timestamping support
 - SWITCH TSN driver: Qbv, Qci, Qbu, Qav, 802.1CB support
- Real time
 - Xenomai
 - LS1028ARDB
 - Baremetal
 - LS1021AIoT, LS1043ARDB, LS1046ARDB
 - LS1028 Baremetal basic Baremetal support
- Industrial protocols
 - LS1028A - NFC click board
 - QT5.11
- OpenIL framework
 - boards: LS1028ARDB

4 Supported NXP platforms

The [Table 4](#) lists the NXP hardware SoCs and boards that support the Real-time Edge software.

Table 4. Supported NXP platforms

Platform	Architecture	Boot
i.MX 6ULL EVK	Arm v7	SD
i.MX 8DXL LPDDR4 EVK	Arm v8	SD, eMMC
i.MX 8M Mini LPDDR4 EVK	Arm v8	SD, eMMC
i.MX 8M Plus LPDDR4 EVK	Arm v8	SD, eMMC
i.MX 93 EVK	Arm v8	SD, eMMC
i.MX 93 9x9 QSB	Arm v8	SD, eMMC
i.MX 93 14x14 EVK	Arm v8	SD, eMMC
i.MX 95 19x19 LPDDR5 EVK	Arm v8	SD, eMMC
i.MX 95 15x15 LPDDR4X EVK	Arm v8	SD, eMMC
i.MX 91 11x11 LPDDR4X EVK	Arm v8	SD, eMMC
i.MX 91 9x9 LPDDR4 QSB	Arm v8	SD, eMMC
i.MX 943 19x19 LPDDR5 EVK	Arm v8	SD, eMMC
i.MX 943 19x19 LPDDR4 EVK	Arm v8	SD, eMMC
LS1028ARDB	Arm v8	SD, eMMC
LS1043ARDB	Arm v8	SD
LS1046ARDB	Arm v8	SD, eMMC
LX2160ARDB Rev 2	Arm v8	SD

5 Feature support matrix

[Table 5](#) shows the features that are supported in this release.

Table 5. Key features

Feature			i.MX 6ULL 14x14 EVK	i.MX 8DXL LPDDR4 EVK	i.MX 8M Mini LPDDR4 EVK	i.MX 8M Plus LPDDR4 EVK	i.MX 93 EVK	i.MX 93 9x9 LPDDR4 QSB	i.MX 93 14x14 LPDDR4x EVK	i.MX 943 19x19 LPDDR4/5 EVK	i.MX 95 19x19 LPDDR5 EVK	i.MX 95 15x15 LPDDR4 EVK	i.MX 91 11x11 LPDDR4 EVK	i.MX 91 9x9 LPDDR4 QSB	i.MX RT1180 EVK	LS1028 ARDB	LS1043 ARDB	LS1046 ARDB	LX2160 ARDB
Boot mode	SD		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y
	eMMC				Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y		Y	
Real-time System	Preempt-RT Linux		Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y		Y	Y	Y	Y
	BareMetal	ICC			Y	Y	Y	Y								Y	Y	Y	Y
		PCIe														Y	Y	Y	
		Ethernet			Y	Y	Y									Y	Y	Y	
		GPIO			Y	Y												Y	
		IPI			Y	Y	Y	Y								Y	Y	Y	Y
		UART			Y	Y	Y	Y								Y	Y	Y	Y
		USB															Y	Y	
		SAI														Y			
		CAN																	
		I2C														Y	Y	Y	
		QSPI																Y	
		IFC																Y	
		Flextimer																Y	
		Linux (communication with BareMetal)	ICC		Y	Y	Y	Y								Y	Y	Y	Y
			IPI		Y	Y	Y	Y								Y	Y	Y	Y
		Single HW Interrupt to multiple cores																Y	
		Newlib Math library			Y	Y	Y	Y								Y	Y	Y	Y
		All Cortex-A cores running under Baremetal														Y		Y	
	Native RTOS on Cortex-A	FreeRTOS			Y	Y	Y		Y		Y	Y	Y	Y					
		Zephyr			Y	Y	Y		Y		Y	Y	Y	Y					
	Jailhouse				Y	Y		Y			Y	Y				Y	Y	Y	
	Harpoon RTOS	FreeRTOS			Y	Y	Y				Y	Y							
		Zephyr			Y	Y	Y				Y	Y							
Heterogeneous Multicore Frame- work	Flexible Real-time System	Flexible Real-time System	Free RTOS		Y	Y	Y		Y		Y	Y							
			Zephyr		Y	Y	Y		Y		Y	Y							

Table 5. Key features...continued

Feature				i.MX 6ULL 14x14 EVK	i.MX 8DXL LPDDR4 EVK	i.MX 8M Mini LPDDR4 EVK	i.MX 8M Plus LPDDR4 EVK	i.MX 93 EVK	i.MX 93 9x9 LPDDR4 QSB	i.MX 93 14x14 LPDDR4x EVK	i.MX 943 19x19 LPDDR4/5 EVK	i.MX 95 19x19 LPDDR5 EVK	i.MX 95 15x15 LPDDR4 EVK	i.MX 91 11x11 LPDDR4 EVK	i.MX 91 9x9 LPDDR4 QSB	i.MX RT1180 EVK	LS1028 ARDB	LS1043 ARDB	LS1046 ARDB	LX2160 ARDB
		RAM Console	Free RTOS			Y	Y	Y		Y		Y	Y							
			Zephyr			Y	Y	Y		Y		Y	Y							
		SMP RTOS	Zephyr				Y	Y		Y										
		Networking stack on A-Core RTOS	Free RTOS			Y	Y	Y		Y										
	Unified Life Cycle Management	U-Boot booting Native RTOS A-Core Image	Free RTOS			Y	Y	Y		Y		Y	Y	Y	Y					
			Zephyr			Y	Y	Y		Y		Y	Y	Y	Y					
		Linux booting Native RTOS A-Core Image	Free RTOS			Y	Y	Y		Y		Y	Y							
			Zephyr			Y	Y	Y		Y		Y	Y							
		U-Boot booting Native RTOS M-Core Image	Free RTOS			Y	Y	Y	Y	Y										
			Free RTOS			Y	Y	Y	Y	Y										
	RPMSG	RPMSG between Linux and M-Core RTOS	Free RTOS			Y	Y	Y	Y	Y										
			Free RTOS			Y	Y	Y		Y		Y	Y							
		RPMSG between Linux and A-Core RTOS	Zephyr				Y	Y		Y										
			Free RTOS				Y	Y		Y										
		RPMSG between 2 A-Core RTOS	Free RTOS				Y	Y		Y										
		RPMSG between Linux and M-core RTOS with enhanced 8 MB buffer	Free RTOS			Y														
		RPMSG Performance Evaluation	Free RTOS				Y													
	Heterogeneous Multicore VirtIO	UART Sharing based on RPMSG	Free RTOS			Y		Y	Y	Y										
		Heterogeneous Multicore VirtIO Performance Evaluation	Free RTOS			Y	Y													
		Heterogeneous Multicore VirtIO Network Sharing	Free RTOS			Y	Y	Y		Y										

Table 5. Key features...continued

Feature				i.MX 6ULL 14x14 EVK	i.MX 8DXL LPDDR4 EVK	i.MX 8M Mini LPDDR4 EVK	i.MX 8M Plus LPDDR4 EVK	i.MX 93 EVK	i.MX 93 9x9 LPDDR4 QSB	i.MX 93 14x14 LPDDR4x EVK	i.MX 943 19x19 LPDDR4/5 EVK	i.MX 95 19x19 LPDDR5 EVK	i.MX 95 15x15 LPDDR4 EVK	i.MX 91 11x11 LPDDR4 EVK	i.MX 91 9x9 LPDDR4 QSB	i.MX RT1180 EVK	LS1028 ARDB	LS1043 ARDB	LS1046 ARDB	LX2160 ARDB
	Industrial Applications	SOEM running on A-core or M-core	Free RTOS			Y	Y	Y												
Heterogeneous Multi-SoC Framework	NETC TSN Switch	DSA single port mode					Y	Y			Y					Y				
		DSA bridge mode					Y	Y			Y					Y				
		MTU configuration					Y	Y			Y					Y				
		VLAN configuration					Y	Y			Y					Y				
		FDB configuration					Y	Y			Y					Y				
		Port statistics					Y	Y			Y					Y				
		PTP stack					Y	Y			Y					Y				
		802.1CB					Y	Y			Y					Y				
		Qci					Y	Y			Y					Y				
		Qbv					Y	Y			Y					Y				
		Qbu					Y	Y			Y					Y				
	HSR					Y	Y			Y					Y					
	Virtual Switch					Y	Y			Y					Y					
Real-time Networking	TSN Standards	Qbv			Y		Y	Y	Y	Y	Y						Y			
		Qbu			Y		Y	Y	Y	Y	Y						Y			
		Qci									Y						Y			
		Qav			Y	Y	Y	Y	Y	Y	Y						Y			
		802.1AS			Y	Y	Y	Y	Y	Y	Y						Y	Y	Y	Y
		802.1CB															Y			
		VCAP chain mode															Y			
		802.1 Q-in-Q															Y			
	TSN Configurations	Linux tc command			Y		Y	Y	Y	Y	Y						Y			
		TSN tool															Y			
		NETCONF/YANG	Qbv		Y		Y	Y	Y	Y	Y	Y					Y			
			Qbu		Y		Y	Y	Y	Y	Y	Y					Y			
			Qci														Y			
			IP		Y		Y	Y	Y	Y	Y	Y					Y			
			MAC		Y		Y	Y	Y	Y	Y	Y					Y			
			VLAN config		Y		Y	Y	Y	Y	Y	Y					Y			
			PTP		Y		Y	Y	Y	Y	Y	Y								
LLDP			Y		Y	Y	Y	Y	Y	Y										
Web-based configuration	Qbv		Y		Y	Y	Y	Y							Y					

Table 5. Key features...continued

Feature				i.MX 6ULL 14x14 EVK	i.MX 8DXL LPDDR4 EVK	i.MX 8M Mini LPDDR4 EVK	i.MX 8M Plus LPDDR4 EVK	i.MX 93 EVK	i.MX 93 9x9 LPDDR4 QSB	i.MX 93 14x14 LPDDR4x EVK	i.MX 943 19x19 LPDDR4/5 EVK	i.MX 95 19x19 LPDDR5 EVK	i.MX 95 15x15 LPDDR4 EVK	i.MX 91 11x11 LPDDR4 EVK	i.MX 91 9x9 LPDDR4 QSB	i.MX RT1180 EVK	LS1028 ARDB	LS1043 ARDB	LS1046 ARDB	LX2160 ARDB
GenAVB/TSN Stack		Qbu			Y		Y	Y	Y								Y			
			Qci														Y			
		Dynamic topology discovery			Y		Y	Y	Y								Y			
		Dynamic TSN configuration	Qci														Y			
			CQF														Y			
			Qbv		Y		Y	Y	Y								Y			
	GenAVB/TSN Stack	gPTP 802.1AS-2020 with multi-domain support			Y	Y	Y	Y	Y	Y	Y	Y					Y	Y	Y	Y
		SRP IEEE 802.1Q-2022		Y	Y	Y	Y	Y	Y	Y		Y					Y	Y	Y	Y
		AVB Standards	AVTP IEEE 1722-2016	Y	Y	Y	Y	Y		Y										
			ATDECC 1722.1-2021	Y	Y	Y	Y	Y		Y										
			Milan v1.2	Y	Y	Y	Y	Y		Y										
			MAAP IEEE 1722-2016	Y	Y	Y	Y	Y		Y										
		AVB Endpoint		Y	Y	Y	Y	Y		Y										
		AVB Bridge			Y ^[1]					Y ^[1]							Y			
		AVB Hybrid								Y ^[1]										
		TSN Endpoint			Y		Y	Y	Y	Y	Y	Y								
		TSN Bridge															Y			
		Media clock recovery		Y	Y	Y ^[2]	Y	Y ^[2]		Y ^[2]										
Industrial Protocol	EtherCAT Main Device	IGH EtherCAT MainDevice stack		Y	Y	Y	Y	Y			Y	Y	Y				Y	Y	Y	Y
		IGH native Ethernet device driver			Y	Y	Y	Y			Y	Y	Y				Y	Y	Y	
		SOEM				Y	Y	Y			Y									
		CodeSYS EtherCAT Main Device stack		Y		Y	Y	Y			Y	Y	Y							
	FlexCAN										Y					Y				
	CANopen										Y					Y				
	OPC UA	open62541		Y	Y	Y	Y	Y	Y	Y		Y					Y	Y	Y	Y
		OPC UA PubSub over TSN			Y		Y	Y	Y	Y		Y	Y				Y			
		OPC UA Frame Summation									Y									

Table 5. Key features...continued

Feature		i.MX 6ULL 14x14 EVK	i.MX 8DXL LPDDR4 EVK	i.MX 8M Mini LPDDR4 EVK	i.MX 8M Plus LPDDR4 EVK	i.MX 93 EVK	i.MX 93 9x9 LPDDR4 QSB	i.MX 93 14x14 LPDDR4x EVK	i.MX 943 19x19 LPDDR4/5 EVK	i.MX 95 19x19 LPDDR5 EVK	i.MX 95 15x15 LPDDR4 EVK	i.MX 91 11x11 LPDDR4 EVK	i.MX 91 9x9 LPDDR4 QSB	i.MX RT1180 EVK	LS1028 ARDB	LS1043 ARDB	LS1046 ARDB	LX2160 ARDB
	BEE (Mikroe Click board)														Y			
	Bluetooth Low Energy (Mikroe Click board)														Y			
	NFC (Mikroe Click board)														Y			
	Modbus	Modbus-RTU	Y	Y	Y	Y	Y	Y	Y		Y							
		Modbus-TCP	Y	Y	Y	Y	Y	Y	Y		Y				Y	Y	Y	Y
	Digital Encoder	BiSS							Y									
		EnDat2.2							Y									
		EnDat3.0							Y									
		HIPERFACE DSL							Y									
		Nikon A-Format							Y					Y				
		Tamagawa T-Format							Y					Y				

[1] Using a SJA1105Q-EVB

[2] Media clock recovery is implemented through a software-based sampling.

6 Related documentation

All documentation related to Real Time Edge software is available on the link: [REALTIME EDGE Documentation](#). The following documents are available:

- **Real-time Edge User Guide** (*REALTIMEEDGEUG*) (provides detailed user guide)
- **Real-time Edge Yocto Project User Guide** (*RTEDGEYOCTOUG*) (provides information for using Yocto build environment)
- **GenAVB/TSN Stack Evaluation User Guide** (*GENAVBTSNUG*) (provides information on how to set up Audio Video Bridging evaluation experiments of the GenAVB/TSN Stack on NXP platforms)
- **Harpoon User's Guide** (*HRPNUG*) (provides information to build Harpoon Yocto images)
- **NXP MPU Cortex-A Core Zephyr User Guide** (*UG10199*) (explains the hardware setup, use cases, and commands to run Zephyr on the supported NXP MPU Cortex A hardware platforms)
- **i.MX6ULL EVK GenAVB/TSN Rework Application Note** (*AN13678*)
- For details about the graphics feature available in i.MX 8M Plus and i.MX 8M Mini boards, refer to the i.MX Graphics User's Guide (<https://www.nxp.com/docs/en/user-guide/UG10159.pdf>).

To boot up and set up the boards mentioned in this document, refer to the instructions available in the following user guides:

- [i.MX 6ULL EVK Quick Start Guide](#)
- [i.MX 8M Mini LPDDR4 EVK Quick Start Guide](#)
- [i.MX 8M Plus LPDDR4 EVK Quick Start Guide](#)
- [i.MX 8X Lite EVK Quick Start Guide](#)
- [LS1028ARDB Quick Start Guide](#)
- [LS1043ARDB Getting Started Guide](#)
- [LS1046ARDB Getting Started Guide](#)
- [LX2160A/LX2160A-Rev2 RDB Quick Start Guide](#)
- [i.MX93EVK Quick Start Guide](#)

7 Open, fixed, and closed issues

This section contains three tables that describe Open, Fixed, and Closed issues.

- Open issues do not currently have a resolution. Workaround suggestions are provided where possible. Refer [Table 6](#).
- Fixed issues have a software fix that has been integrated into the '*Fixed In*' Release. Refer [Table 7](#).
- Closed issues are issues where the root cause and fix are outside the scope of Real-time Edge Software. Disposition is to provide the explanation. Refer [Table 8](#).

Table 6. Open issues in Real-time Edge software v3.2

ID	Description	Opened In	Workaround
INDLINUX-3750	Logout after listing services by running the command 'service --status-all' on LS1028ARDB	Real-time Edge software v2.6	None
INDLINUX-5034	Unable to boot the LS boards with BareMetal image	Real-time Edge software v3.2	None
INDLINUX-5036	i.MX 8MM board goes into a 'hang' state with BareMetal image	Real-time Edge software v3.2	None

Table 7. Fixed Issues in Real-time Edge Software v3.2

ID	Description	Opened In	Fixed In
None			

Table 8. Closed Issues in Real-time Edge Software v3.2

ID	Description	Opened In	Disposition
None	-	-	-

8 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

Copyright 2024-2025 NXP Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials must be provided with the distribution.
3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

9 Revision history

[Table 9](#) summarizes the revisions to this document.

Table 9. Document revision history

Document ID	Release date	Description
RN00161 v.3.2	29 July 2025	Updated for Real Time Edge Software Rev 3.2
RN00161 v.3.1	26 March 2025	Updated for Real Time Edge Software Rev 3.1
RN00161 v.3.0	17 December 2024	Updated for Real Time Edge Software Rev 3.0
RN00161 v.2.0	25 July 2024	Updated for Real Time Edge Software Rev 2.9
RN00161 v.1.0	29 March 2024	Initial release for Real Time Edge Software Rev 2.8

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.

Evaluation products — This evaluation product is intended solely for technically qualified professionals, specifically for use in research and development environments to facilitate evaluation purposes. It is not a finished product, nor is it intended to be a part of a finished product. Any software or software tools provided with an evaluation product are subject to the applicable license terms that accompany such software or software tools. This evaluation product is provided on an "as is" and "with all faults" basis for evaluation purposes only and is not to be used for product qualification or production. If you choose to use these evaluation products, you do so at your risk and hereby agree to release, defend and indemnify NXP (and all of its affiliates) for any claims or damages resulting from your use. NXP, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this evaluation product remains with user.

In no event shall NXP, its affiliates or their suppliers be liable to user for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out of the use of or inability to use the evaluation product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that user might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP, its affiliates and their suppliers and user's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by user based on reasonable reliance up to the greater of the amount actually paid by user for the evaluation product or five dollars (US\$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose and shall not apply in case of willful misconduct.

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.

Amazon Web Services, AWS, the Powered by AWS logo, and FreeRTOS — are trademarks of Amazon.com, Inc. or its affiliates.

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

I2C-bus — logo is a trademark of NXP B.V.

Layerscape — is a trademark of NXP B.V.

Oracle and Java — are registered trademarks of Oracle and/or its affiliates.

QorIQ — is a trademark of NXP B.V.

Contents

1	Feature introduction	2	3.1	What's new in Real-time Edge software v3.2	24
1.1	Real-time System	2	3.2	What's new in Real-time Edge software v3.1	25
1.1.1	Preempt-RT Linux on Cortex-A core	3	3.3	What's new in Real-time Edge software v3.0	25
1.1.2	BareMetal on Cortex-A core	3	3.4	What's new in Real-time Edge software v2.9	26
1.1.3	Native RTOS SMP/AMP on Cortex-A Core	3	3.5	What's new in Real-time Edge software v2.8	26
1.1.4	RTOS SMP/AMP on Cortex-A Core with Jailhouse	3	3.6	What's new in Real-time Edge software v2.7	27
1.1.5	RTOS and Baremetal on Cortex-M Core	4	3.7	What's new in Real-time Edge software v2.6	28
1.2	Heterogeneous Multicore Framework	4	3.8	What's new in Real-time Edge software v2.5	29
1.2.1	Flexible Real-time System	4	3.9	What's new in Real-time Edge software v2.4	29
1.2.2	Data communication between different operating systems	5	3.10	What's new in Real-time Edge software v2.3	30
1.2.3	Resource sharing between different operating systems	5	3.11	What's new in Real-time Edge software v2.2	31
1.2.4	Unified Life-Cycle Management	5	3.12	What's new in Real-time Edge software v2.1	32
1.2.5	Unified SW release and development environment	5	3.13	What's new in Real-time Edge software v2.0	33
1.3	Heterogeneous Multi-SoC framework	5	3.14	What's new in OpenIL v1.11	34
1.4	Industrial networking	6	3.15	What's new in OpenIL v1.10	34
1.4.1	EtherCAT	6	3.16	What's new in OpenIL v1.9	34
1.4.1.1	IGH EtherCAT	7	3.17	What's new in OpenIL v1.8	35
1.4.1.2	User Space IGH EtherCAT	7	3.18	What's new in OpenIL v1.7	35
1.4.1.3	Real-time Edge Servo Stack	8	3.19	What's new in OpenIL v1.6	36
1.4.1.4	SOEM EtherCAT	8	3.20	What's new in OpenIL v1.5	36
1.4.1.5	CODESYS EtherCAT	8	3.21	What's new in OpenIL v1.4	37
1.4.2	TSN	9	4	Supported NXP platforms	38
1.4.2.1	TSN hardware capability	9	5	Feature support matrix	39
1.4.2.2	TSN configuration	9	6	Related documentation	44
1.4.2.3	Remote configuration using NETCONF/YANG	10	7	Open, fixed, and closed issues	45
1.4.2.4	Web-based configuration	10	8	Note about the source code in the document	46
1.4.3	GenAVB/TSN Stack	11	9	Revision history	47
1.4.3.1	GenAVB/TSN Roles	11		Legal information	48
1.4.3.2	GenAVB/TSN Protocols	12			
1.4.3.3	User Library	13			
1.4.3.4	AVB Kernel Module	13			
1.4.3.5	Supported configurations	13			
1.4.3.6	AVB endpoint example applications	13			
1.4.3.7	TSN endpoint example application	13			
1.4.4	IEEE 1588/802.1AS	15			
1.4.5	Network redundancy	15			
1.4.6	OPC-UA	17			
1.4.6.1	Open62541	18			
1.4.6.2	OPC UA PubSub over TSN	19			
1.4.6.3	OPC UA Frame Summation	20			
1.4.7	MODBUS	20			
1.4.7.1	Libmodbus	20			
1.4.7.2	Libmodbus introduction	20			
1.4.8	FlexCAN and CANopen	21			
1.4.9	UART 9-bit Multidrop mode (RS-485) support	21			
2	Yocto building	22			
3	What's new	24			

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