AN15056

NETC Switch Latency Test on i.MX RT1180

Rev. 1.0 — 21 July 2026 Application note

Document information

Information	Content
Keywords	AN15056, i.MX RT1180, switch latency test, MIMXRT1180-EVK board, NETC
Abstract	This application note introduces the NETC TSN switch latency test with Spirent on i.MX RT1180, and describes the test environment, test cases, and measurement procedure.

1 Introduction

This application note introduces the NETC TSN switch latency test with Spirent on i.MX RT1180, and describes the test environment, test cases, and measurement procedure.

The hardware environment is the MIMXRT1180-EVK board. The test environment is the Spirent M1 appliance and the corresponding Spirent Test Center (STC) tool.

This application note includes software for reproducing the latency test. The software is delivered as a patch file. Apply this patch file on top of the standard MCUXpresso SDK v26.03 example project for i.MX RT1180.

2 NETC overview

NETC is a TSN capable Ethernet IP, which enables a comprehensive portfolio of Ethernet solutions. As shown in Figure 1, NETC provides a full range of Ethernet capabilities including Ethernet Controller (ENETC) and switch functionality.

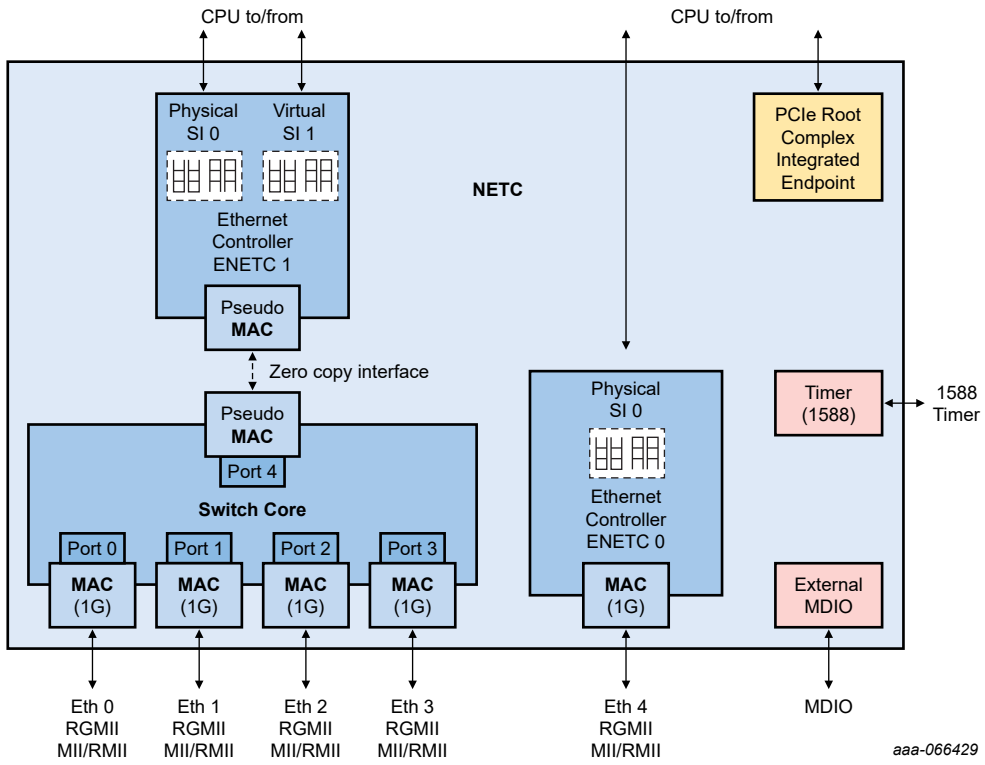
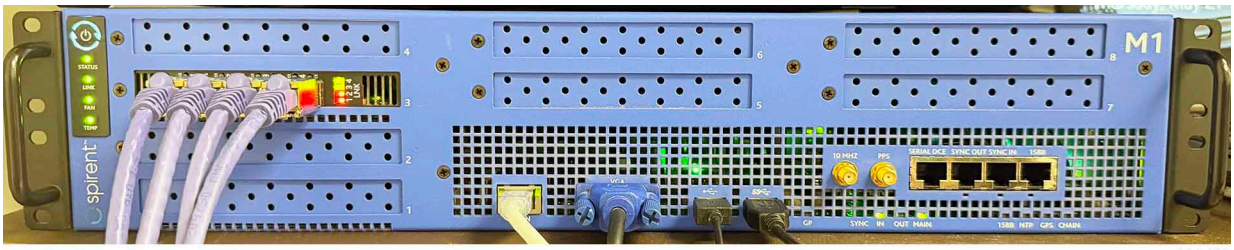


Figure 1. RT1180 NETC block diagram

3 Test environment

The following are the test environments covered in this document:

1. Spirent M1 appliance
   The Spirent M1 appliance combines Layer 2–3 traffic generation and analysis with network emulation and application-layer protocol support to emulate a wide range of devices, users, and traffic scenarios. It runs STC, enabling comprehensive test packages for functional, performance, and network verification.



aaa-066430

Figure 2. Spirent M1 appliance

2. STC
STC is an end-to-end test solution used as a traffic generator and analyzer. It provides configurable test packages to create controlled Ethernet traffic and measure system latency.

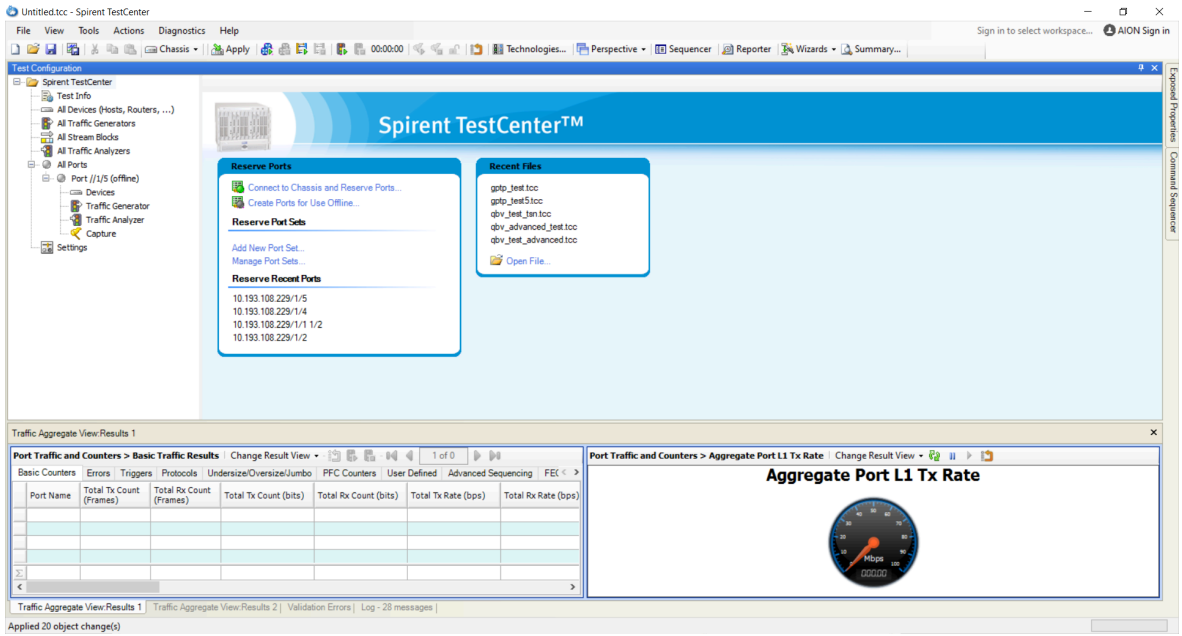


Figure 3. STC

3. MIMXRT1180-EVK board
The MIMXRT1180-EVK board is the hardware used to run NETC application and test performance, which is identified as the Device Under Test (DUT) in Figure 4.

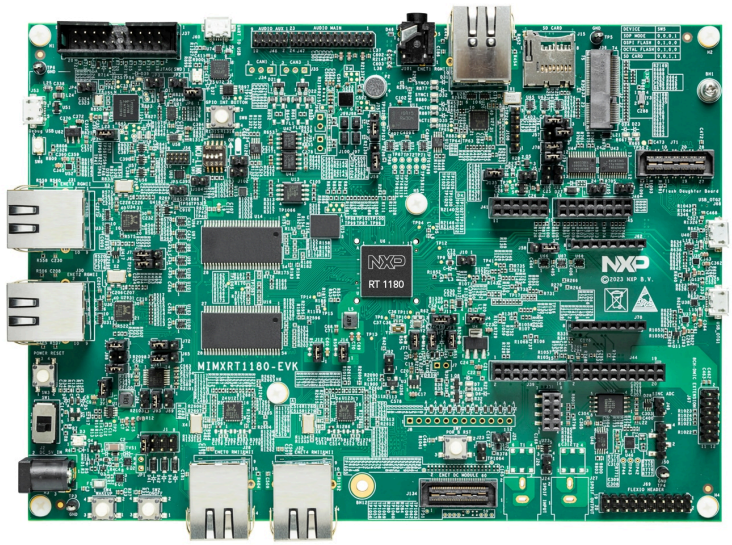


Figure 4. MIMXRT1180-EVK board

4. Hardware setup
   Connect the ETH2/ENET2 (J30) and ETH3/ENET3 (J31) of MIMXRT1180-EVK (DUT) to Spirent M1 port 3 and port 4 respectively. Connect the MCU-link USB connector (J53) of the EVK to the USB interface of a Windows laptop (that also runs the STC and software for DUT). Also, connect the LAN of Spirent M1 to the network so that it is accessible via STC.

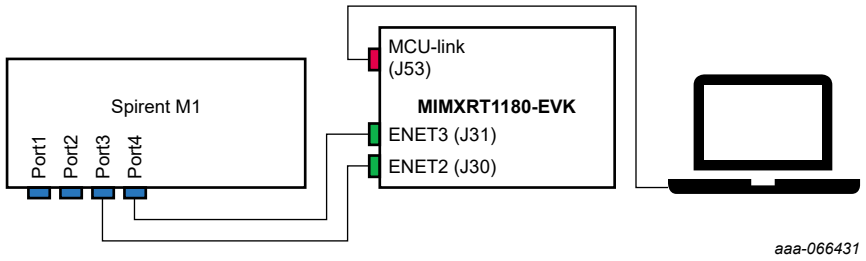


Figure 5. Hardware setup

4 NETC latency test

This section describes the latency test for the NETC switch via ETH ports ENET2 and ENET3. The test part is shown in red box in Figure 6.

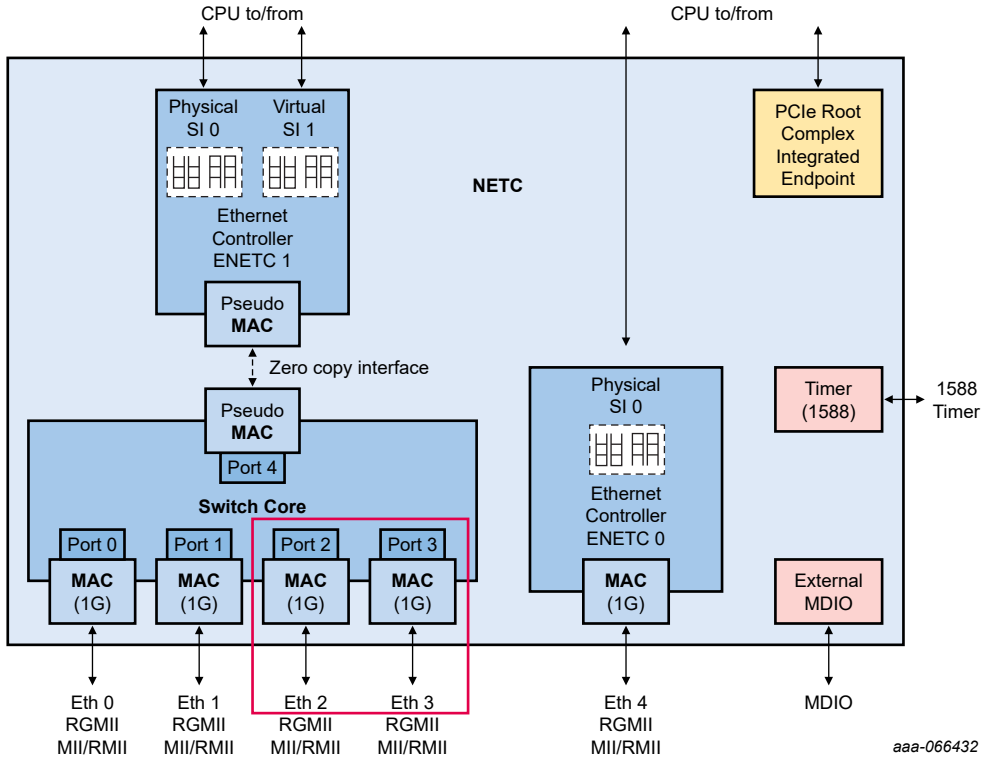


Figure 6. Latency test

4.1 Test cases

The NETC switch test cases summary is shown in Table 1.

Table 1. NETC switch ETH 2-3 latency test cases

Switch mode	Cut-Through	Store-and-Forward	
Latency type	FIFO	LIFO	FIFO
Speed mode	1000 Mbit/s	1000 Mbit/s	1000 Mbit/s
Interface type	RGMII	RGMII	RGMII
EVK port tested	Two ports (ENET2 and ENET3)	Two ports (ENET2 and ENET3)	Two ports (ENET2 and ENET3)
Frame size (byte)	64, 65, 129, 512, 1518, iMIX (combined with 64, 594 and 1518)		

To evaluate Store-and-forward switching both FIFO and LIFO latency measurements can be used. FIFO latency represents end-to-end forwarding delay. This delay includes the time required to receive and store the frame prior to transmission, and is therefore dependent on frame length. LIFO latency removes this frame-length dependency, which provides a measure of the switching latency that is independent of frame size. Reporting both metrics quantifies the impact of frame buffering while enabling a direct comparison between store-and-forward and cut-through switching modes.

4.2 DUT configuration

To configure the DUT, start with installing VS Code and MCUXpresso for VS Code extension. For more details on using the MCUXpresso installer to install and configure dependencies, see the MCUXpresso for VS Code documentation.

4.2.1 Import SDK repository and example project

Once the MCUXpresso for VS Code is ready to use, import MCUXpresso SDK repository v26.03.00.

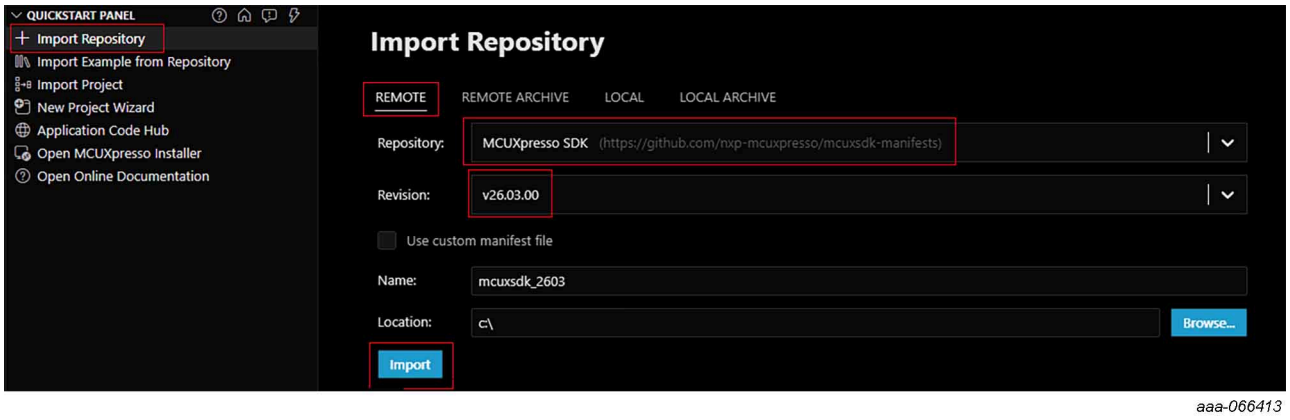
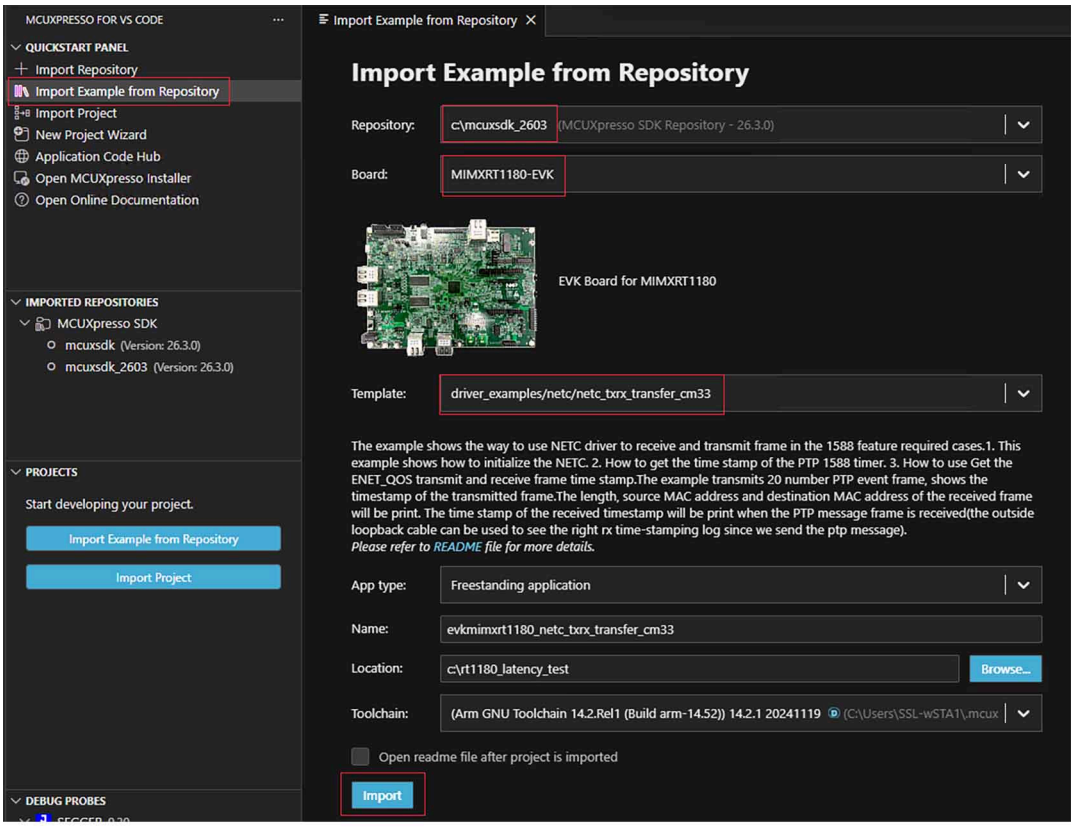


Figure 7. Import repository

Then import example **netc_txrx_transfer_cm33** for the MIMXRT1180-EVK board.



aaa-066414

Figure 8. Import example from repository

4.2.2 Apply the patch

Navigate to your imported `evkmimxrt1180_netc_txrx_transfer_cm33` project directory. Download the patch `switch_latency_test.patch` from the associated file of this application note and apply it. For Windows, a command-line utility such as Git Bash can be used to run commands to apply the patch.

```
patch -p1 < switch_latency_test.patch
```

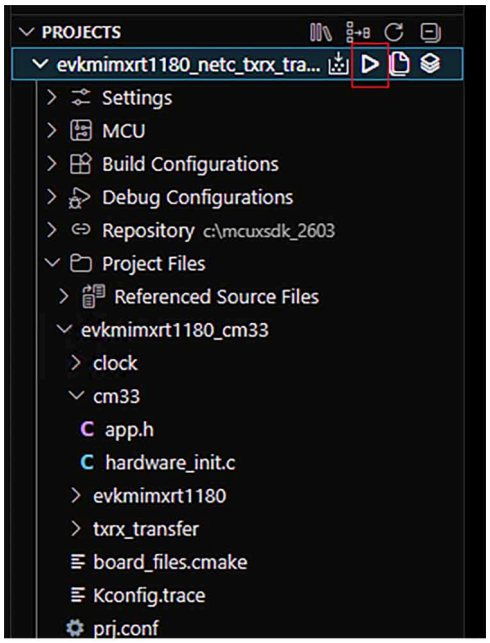
This patch modifies two project files to configure the NETC switch hardware for external traffic generator-based latency measurement rather than software-driven loopback.

`hardware_init.c` is modified such that the PHY local loopback is disabled to allow the frames to travel out on the physical wire. Also, auto-negotiation is enabled on the RTL8211F (1 Gbit/s) PHYs so that they properly establish a link with external test equipment such as Spirent.

`netc_txrx_transfer.c` is modified to hardcode the switch port configuration to RGMII at 1 Gbit/s full duplex on ports 2 and 3. Hardcoding the switch port bypasses the original dynamic PHY negotiation. The port membership and forwarding behavior are set to flood frames between these two ports. After the switch is initialized, the CPU halts in an infinite loop so that the software no longer sends or receives frames itself. Instead, the Spirent traffic generator drives the traffic through the switch and measures the port-to-port forwarding latency directly.

4.2.3 Build and run the project

To build and download the project to the flexspi_nor on the board, click the debug (play button) icon.



aaa-066415

Figure 9. Build and run the project

Once the execution halts in main, click the “continue” button to continue debugging/running the program.

Alternatively, you can build a persistent bootable image to start running the demo from POR. For more details, see MCUXSPTUG_26.03: Secure Provisioning Tool User Guide 26.03.

4.2.4 Configuration for cut-through Switch mode

The SDK example and the patch enable the store-and-forward Switch mode by default. To enable cut-through Switch mode, uncomment the following lines of the code in `netc_txrx_transfer.c`.

```
SWT_GetDefaultConfig(&g_swt_config);

for (uint32_t i = 2U; i < 4u; i++)
{
    g_swt_config.ports[i].ethMac.miiMode    = kNETC_RgmiiMode;
    g_swt_config.ports[i].ethMac.miiSpeed   = kNETC_MiiSpeed1000M;
    g_swt_config.ports[i].ethMac.miiDuplex = kNETC_MiiFullDuplex;
    g_swt_config.ports[i].inCutThrough = true; //enable for cut through mode
    g_swt_config.ports[i].outCutThrough = true; //enable for cut through mode
    g_swt_config.ports[i].commonCfg.parser.enableL4Parser = false; //Optimization: Disable parsing Layer 4 (TCP, UDP) - PPCR[L4HFP] = 0
    g_swt_config.ports[i].commonCfg.parser.enableL3Parser = false; //Optimization: Disable parsing Layer 3 (IPv4, IPv6) - PPCR[L3HFP] = 0
    g_swt_config.ports[i].commonCfg.parser.l2PloadCount = 0; //Optimization: Disable parsing Layer 2 payload - PPCR[L2PFS] = 0
    g_swt_config.ports[i].bridgeCfg.isRxVlanAware = false;
}
```

aaa-066734

Figure 10. Configuration for cut-through Switch mode

4.2.5 NETC switch configuration optimizations

To characterize the lowest achievable switching latency, some optimizations can be applied to the switch configuration. They mainly pertain to the parser, which is part of the ingress processing pipeline in the NETC Switch datapath. The parser parses the frames and extracts the required packet header fields in preparation for classification up to L4 protocol headers within the first 144 bytes of the frame.

By default, the parser parses and extracts fields from each layer. To configure the parser to end parsing at L2 or L3 regardless of the content of the frame, set the correct value of the Port Parser Configuration Register

(PPCR). The parser also extracts the payload (up to 24 bytes by default) of the last layer parsed (L2, L3, or L4). The maximum amount of payload to extract is configurable for each layer via PPCR.

The parameters of the parser can be adjusted by setting appropriate values of the PPCR bits as given below to optimize the switching latency. For more details on the PPCR register and bit field descriptions, see the *i.MX RT1180 Reference Manual* (IMXRT1180RM).

- Disable parsing Layer 4 (TCP, UDP)
  PPCR[L4HFP] = 0
- Disable parsing Layer 3 (IPv4, IPv6)
  PPCR[L3HFP] = 0
- Disable parsing Layer 2 payload
  PPCR[L2PFS] = 0

Note: *The above settings reduce parser processing overhead to improve latency. However, this optimization limits the ability of the parser to extract payload information for subsequent packet processing functions. Features that depend on the extracted payload information, such as stream filtering, are affected.*

To apply the optimizations as described above, uncomment the following lines of code in the netc_txrx_transfer.c.

```
SWT_GetDefaultConfig(&g_swt_config);

for (uint32_t i = 2U; i < 4u; i++)
{
    g_swt_config.ports[i].ethMac.miiMode    = kNETC_RgmiiMode;
    g_swt_config.ports[i].ethMac.miiSpeed   = kNETC_MiiSpeed1000M;
    g_swt_config.ports[i].ethMac.miiDuplex = kNETC_MiiFullDuplex;
    g_swt_config.ports[i].inCutThrough = true; //enable for cut through mode
    g_swt_config.ports[i].outCutThrough = true; //enable for cut through mode
    g_swt_config.ports[i].commonCfg.parser.enableL4Parser = false; //Optimization: Disable parsing Layer 4 (TCP, UDP) - PPCR[L4HFP] = 0
    g_swt_config.ports[i].commonCfg.parser.enableL3Parser = false; //Optimization: Disable parsing Layer 3 (IPv4, IPv6) - PPCR[L3HFP] = 0
    g_swt_config.ports[i].commonCfg.parser.l2PloadCount = 0; //Optimization: Disable parsing Layer 2 payload - PPCR[L2PFS] = 0
    g_swt_config.ports[i].bridgeCfg.isRxVlanAware = false;
}
```

aaa-066735

Figure 11. NETC switch configuration optimizations

4.2.6 Configuration for larger frame size

The NETC switch driver in the SDK sets the default value for maximum frame size to 0x600 or 1536 bytes. To test for a larger frame size, for example, a frame size of 2000 bytes, the value can be updated in the fsl_netc_switch.c file in the SDK repository. The path to the file is mcusdk > drivers > netc > fsl_netc_switch.c.

C: > mcuxsdk_2603 > mcuxsdk > drivers > netc > C fsl_netc_switch.c > SWT_GetDefaultConfig(swt_config_t *)

```
180   status_t SWT_GetDefaultConfig(swt_config_t *config)
185       for (uint32_t i = 0U; i < NETC_SOC_SWT_PORT_NUM; i++)
207           config->ports[i].commonCfg.txPpduBco                = 20U;
208           config->ports[i].commonCfg.timeGate.holdSkew        = 64;
209           config->ports[i].commonCfg.parser.l2PloadCount      = 24;
210           config->ports[i].commonCfg.parser.l3PayloadCount    = 24;
211           config->ports[i].commonCfg.parser.enableL3Parser    = true;
212           config->ports[i].commonCfg.parser.l4PayloadCount    = 24;
213           config->ports[i].commonCfg.parser.enableL4Parser    = true;
214           config->ports[i].ethMac.PreemptionConfig.preemptMode     = kNETC_PreemptDisable;
215           config->ports[i].ethMac.PreemptionConfig.enMergeVerify   = false;
216           config->ports[i].ethMac.PreemptionConfig.mergeVerifyTime = 10U;
217           config->ports[i].ethMac.PreemptionConfig.raf_size        = kNETC_RafSize64;
218           config->ports[i].ethMac.enTxPad                     = true;
219           config->ports[i].ethMac.rxMinFrameSize              = 64U;
220           config->ports[i].ethMac.rxMaxFrameSize              = 0x7D0U;
221           config->ports[i].ethMac.maxBackPressOn              = 3036U;
222           config->ports[i].ethMac.minBackPressOff             = 20U;
223           config->ports[i].enTxRx                             = true;
224           for (uint32_t j = 0U; j < 8U; j++)
225           {
226               config->ports[i].txTcCfg[j].enTcGate          = true;
227               config->ports[i].txTcCfg[j].sduCfg.maxSduSized = 0x7D0U;
228               config->ports[i].txTcCfg[j].sduCfg.sduType     = kNETC_MPDU;
229               config->ports[i].ipvToTC[j]                    = (uint8_t)j;
230           }
231       }
```

aaa-066733

Figure 12. Configuration for larger frame size

4.3 STC configuration

The RFC 2544 protocol is an international standard proposed by the RFC organization for evaluating network interconnection devices (firewall, IDS, Switch, and so on). It specifies many parameters for testing different network devices.

Based on this, the RFC 2544 test package in STC is used to test latency of the NETC switch. The RFC 2544 test package provides four types of network tests in Figure 13. In this application note, the focus is on the latency test.

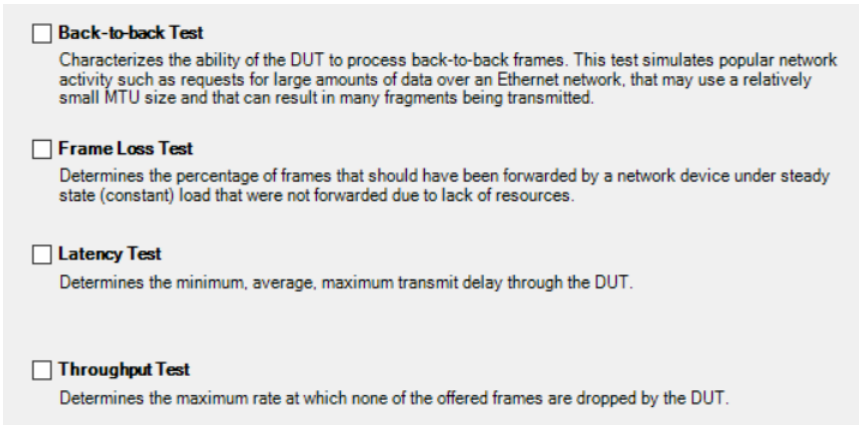


Figure 13. RFC 2544 test

This section introduces the entire testing process in STC in detail for the Store and Forward LIFO latency type. The basic method remains the same for other test cases as well. To test with iMIX configuration, see Section 4.3.1.

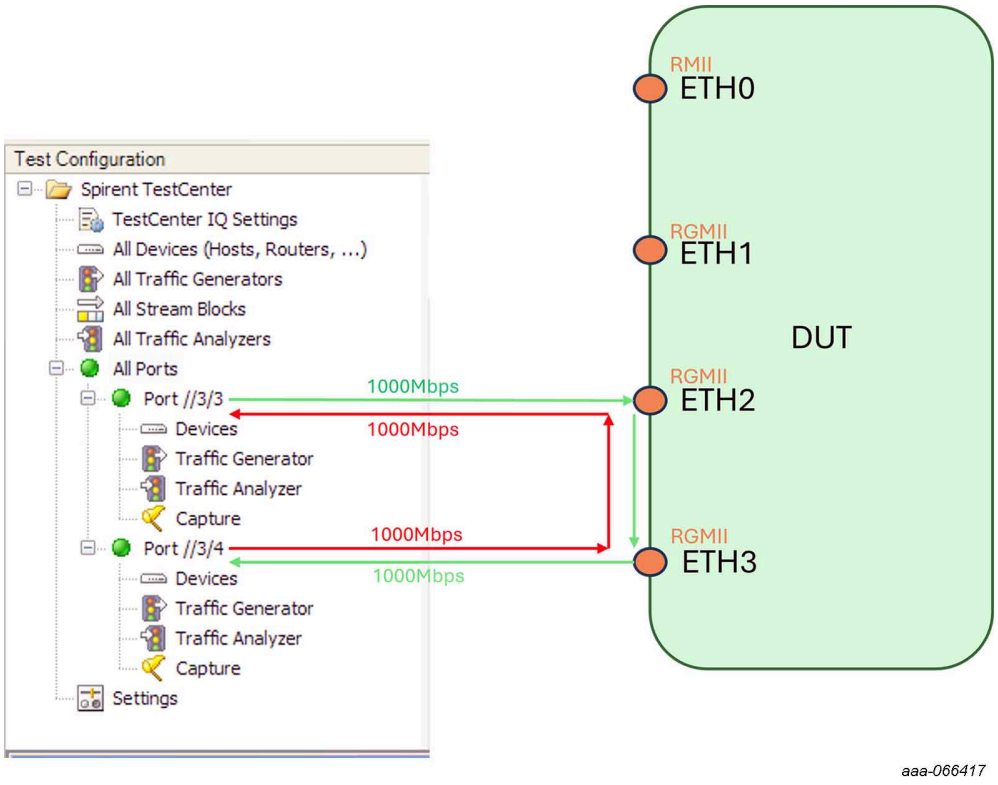


Figure 14. Test case diagram

The main steps of the above test case with STC RFC 2544 are as follows:

1. Open the STC application and connect to the M1 appliance.

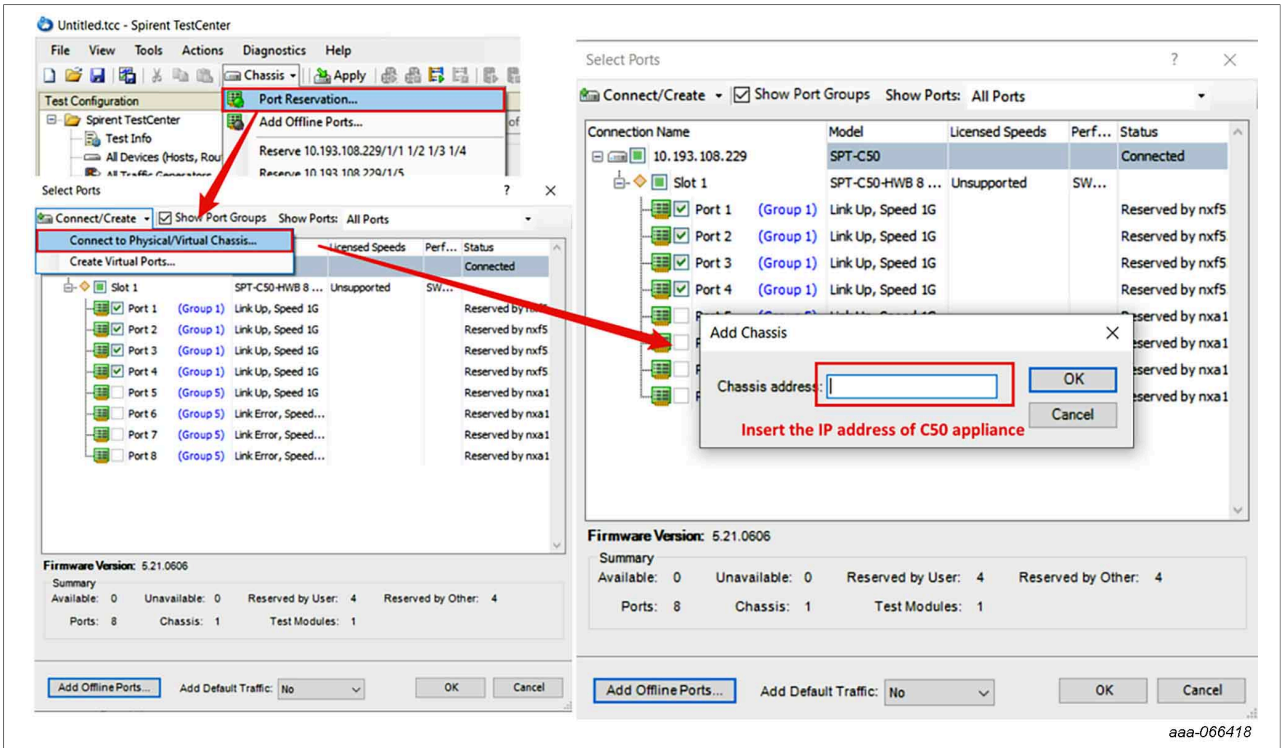


Figure 15. STC configuration

2. Select test ports connected to the ports of DUT.

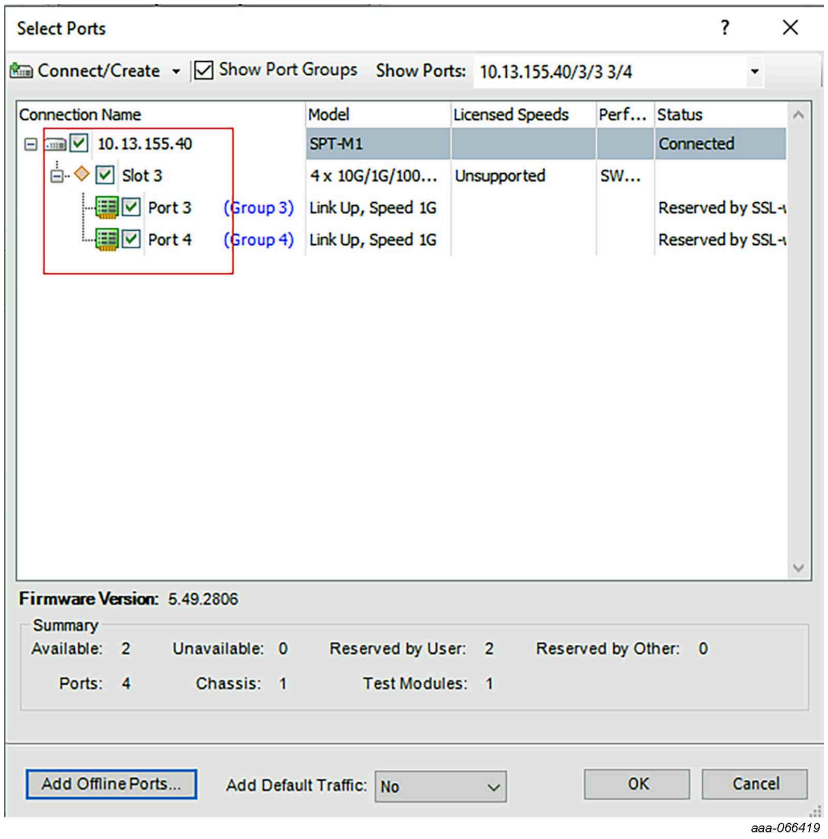


Figure 16. Select test ports

3. Select RFC 2544 test package.

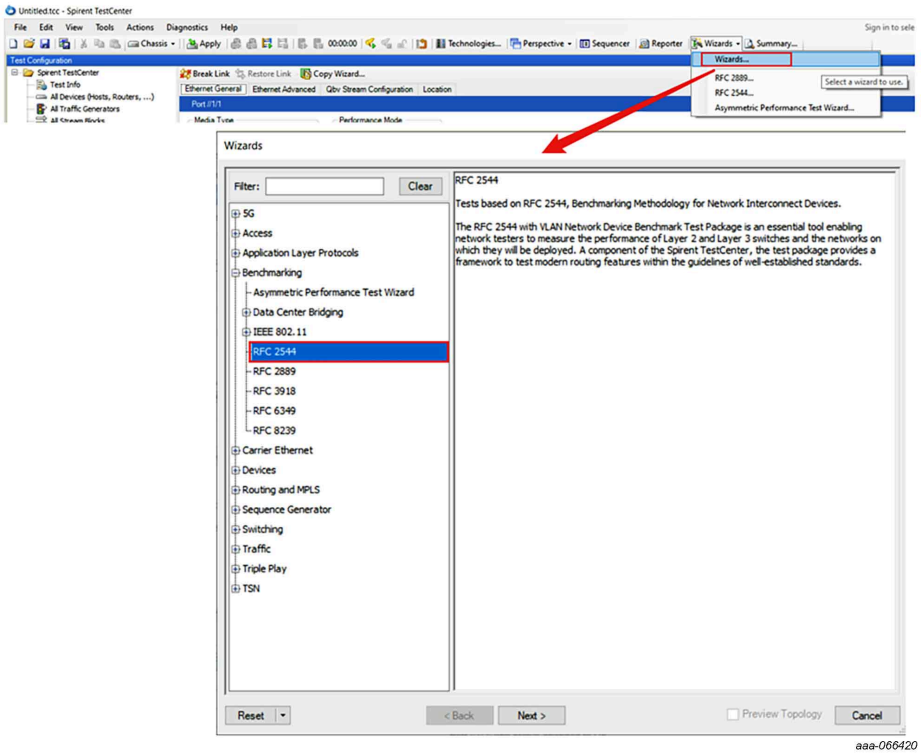
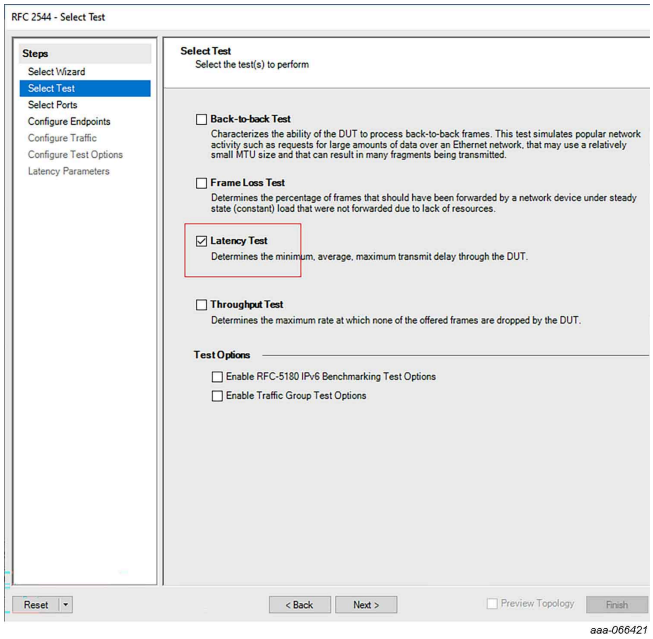


Figure 17. Select a test package

4. Select latency test and ports.



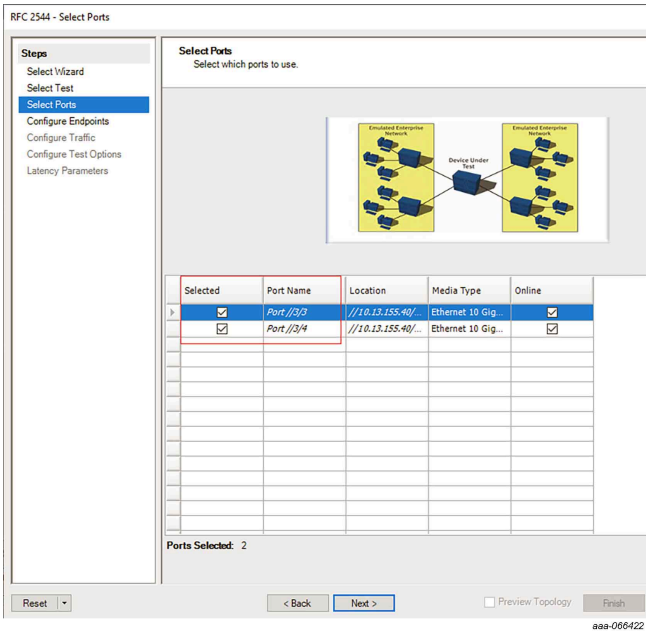


Figure 18. Select latency test and ports

5. Create and configure devices.

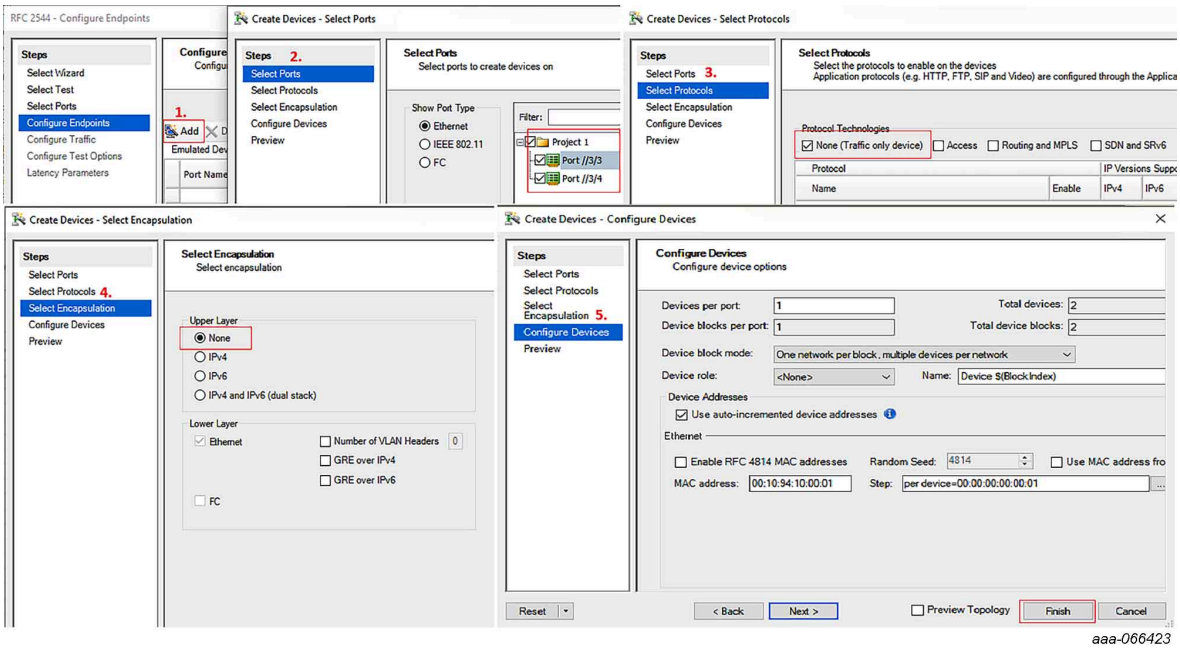
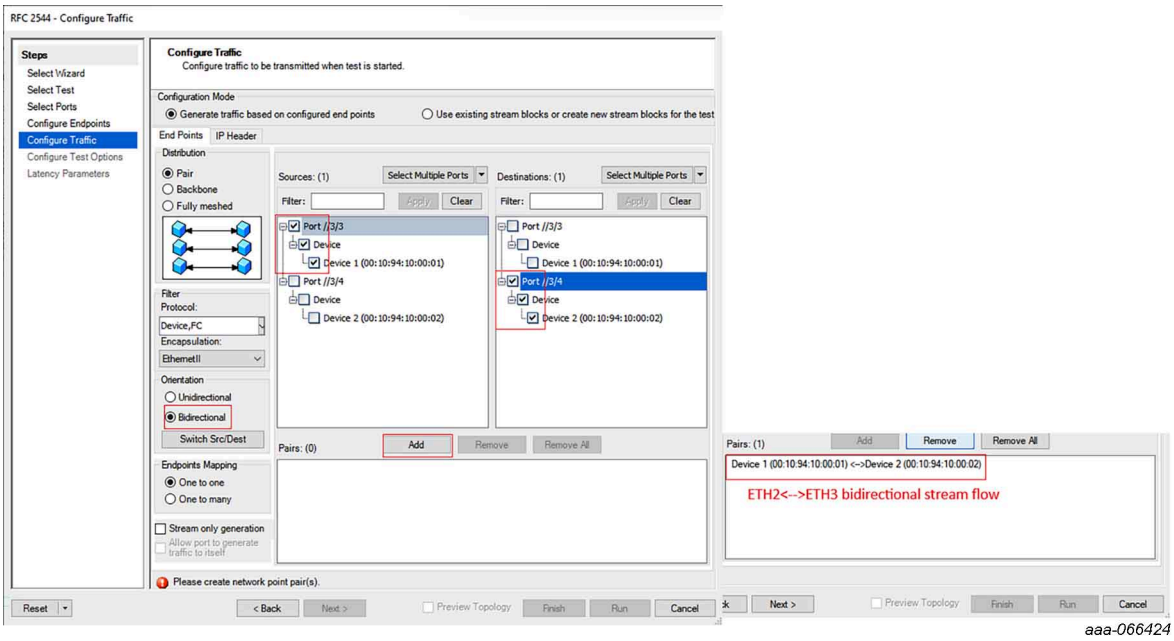


Figure 19. Create and configure devices

6. Create a bidirectional stream flow between the switch ports 2 and 3.



aaa-066424

Figure 20. Create a stream flow

7. Test options configuration
   Select the **Latency Type** as **LIFO (Store and Forward)** for this test case, the switch mode of DUT is store-and-forward. You can change the latency type from this GUI window based on the test case.

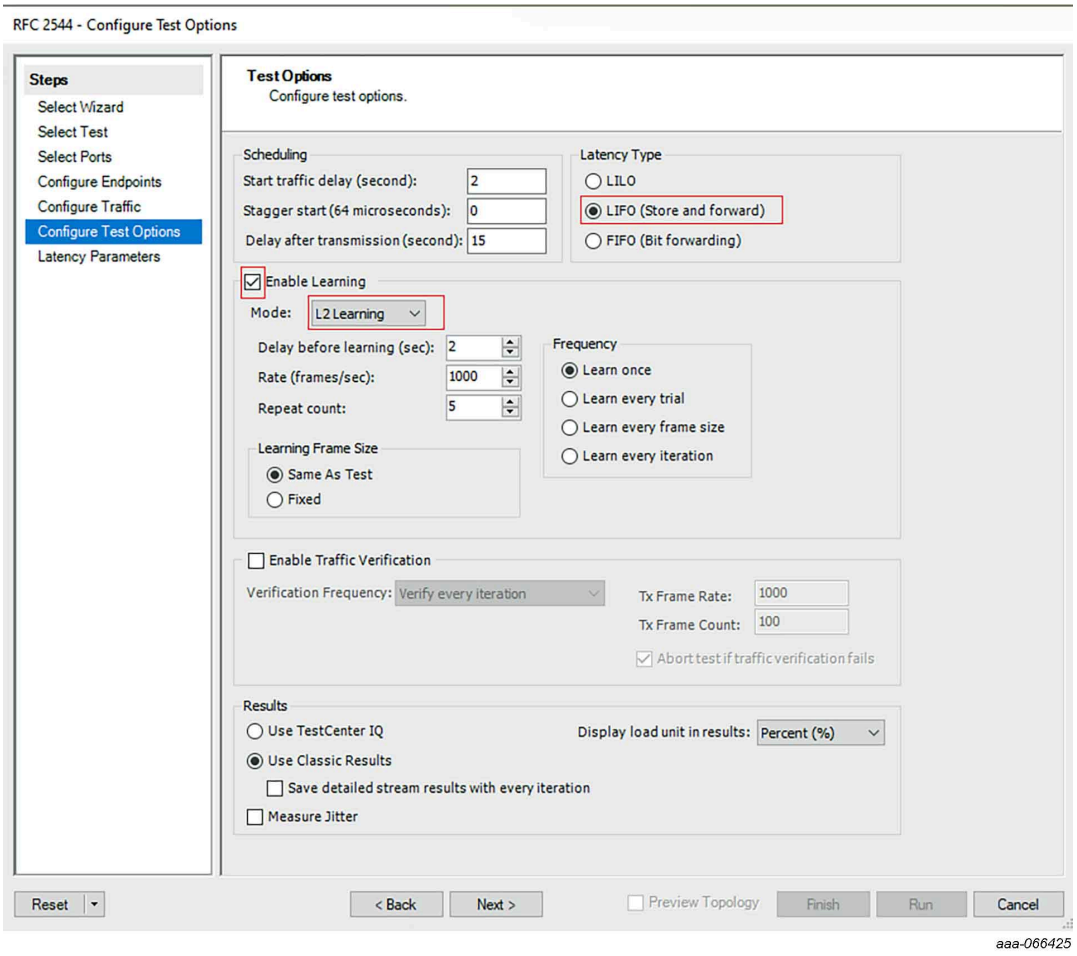


Figure 21. Configure test options

8. Latency parameters
   Configure the parameters of latency test as given in Figure 22. these parameters can be modified as needed.
   For example, add frame size 2000 to the configuration if the max frame size is increased as given in Section 4.2.6.

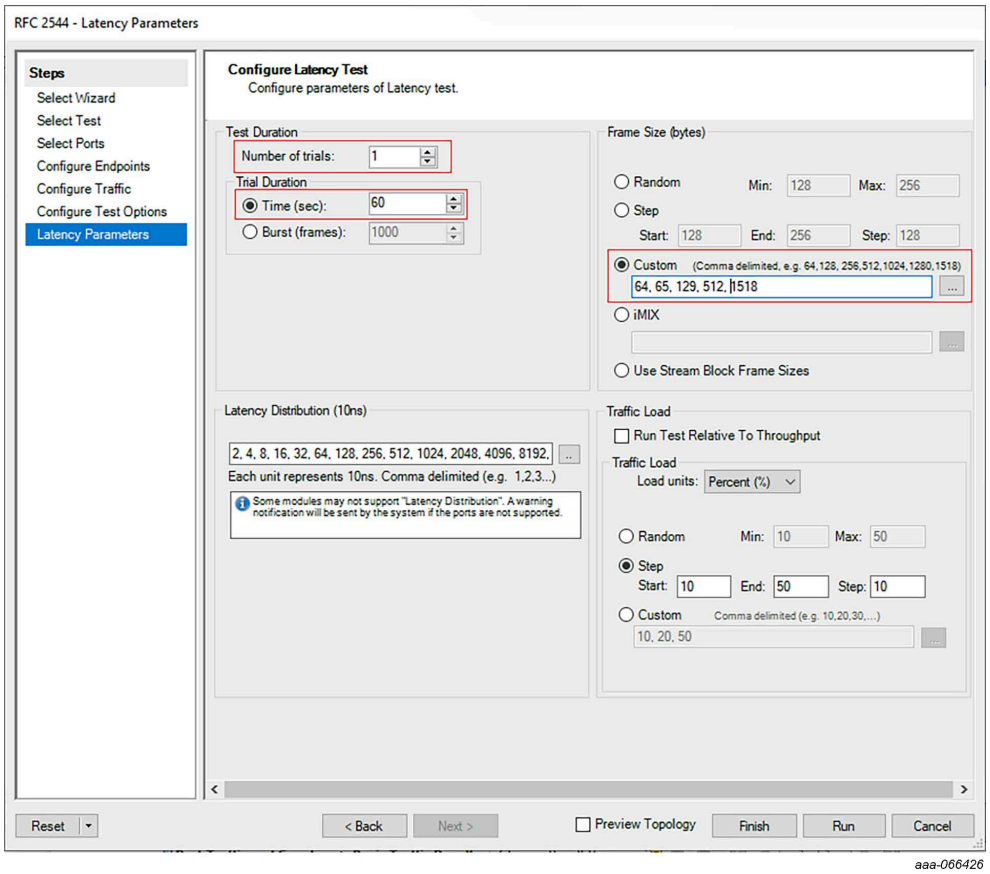


Figure 22. Configure latency test parameters

9. After clicking the finish button, RFC 2544 test for this test case is created as given in Figure 23. To start the test, click the 'start sequencer' button.

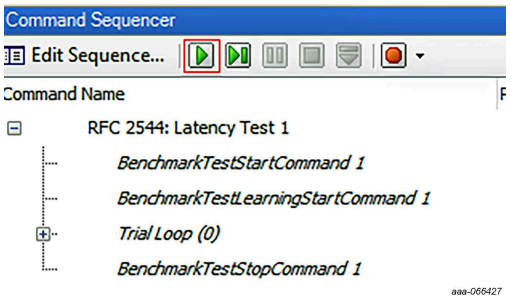


Figure 23. Start the test

10. After the test is finished, choose reporter on the menu to view the test results.

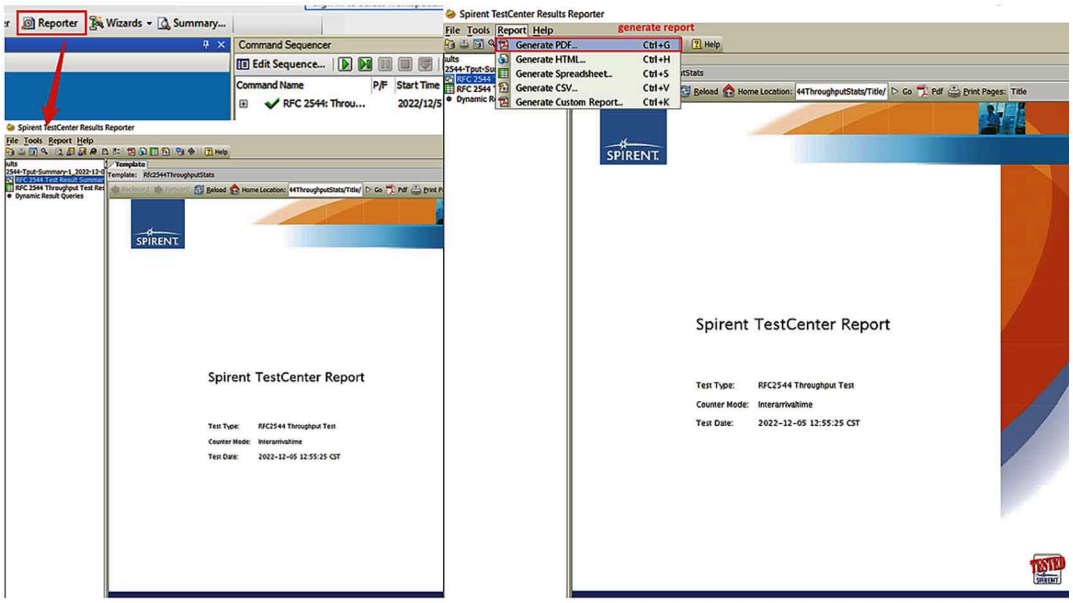
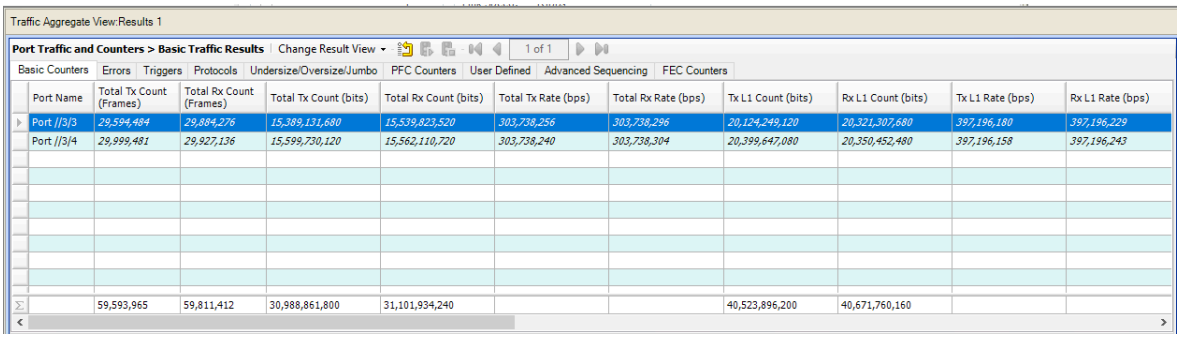


Figure 24. View test results

11. During the test process, live results can be observed in the **Traffic Aggregate View: Results 1**, which can be opened from the view tab.



Port Name	Total Tx Count (Frames)	Total Rx Count (Frames)	Total Tx Count (bits)	Total Rx Count (bits)	Total Tx Rate (bps)	Total Rx Rate (bps)	Tx L1 Count (bits)	Rx L1 Count (bits)	Tx L1 Rate (bps)	Rx L1 Rate (bps)
Port //3/3	29,594,484	29,884,276	15,389,131,680	15,539,823,520	303,738,256	303,738,296	20,124,249,120	20,321,307,680	397,196,180	397,196,229
Port //3/4	29,999,481	29,927,136	15,599,730,120	15,562,110,720	303,738,240	303,738,304	20,399,647,080	20,350,452,480	397,196,158	397,196,243
Σ	59,593,965	59,811,412	30,988,861,800	31,101,934,240			40,523,896,200	40,671,760,160		

Figure 25. Traffic aggregate view

4.3.1 iMIX configuration

To configure iMIX traffic:

1. Go to **Tools**.
2. Select **iMIX Editor** from the **Tools** menu.

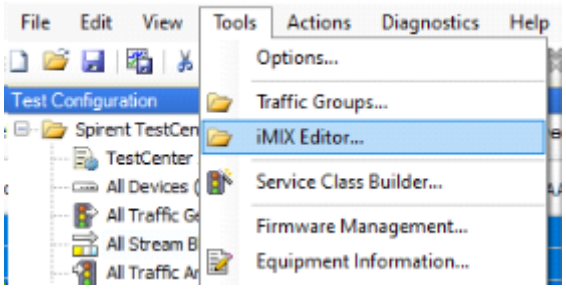


Figure 26. Configure iMIX traffic

Add an iMIX set called NXP. Configure the IP total length as shown in Figure 27.

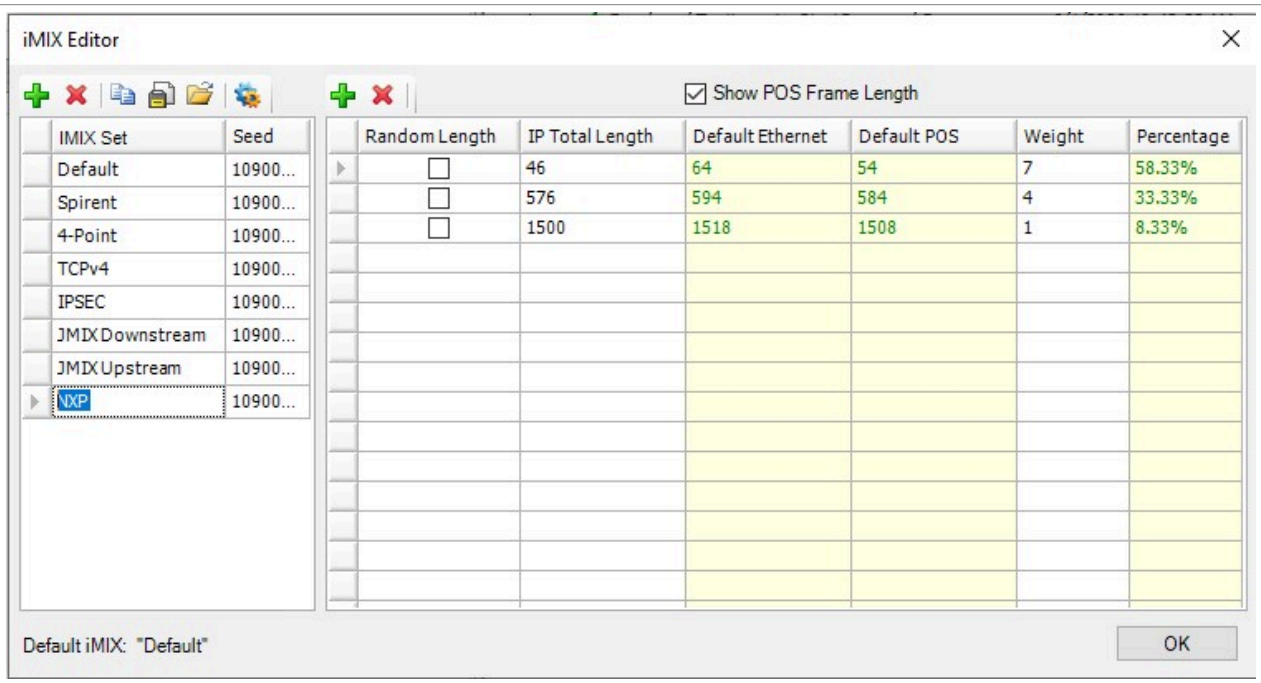


Figure 27. Add iMIX set

For testing with iMIX:

1. Select the **iMIX** option in the **Frame Size (bytes)** section.
2. Select **NXP** in the **Select iMIX Distributions** window.

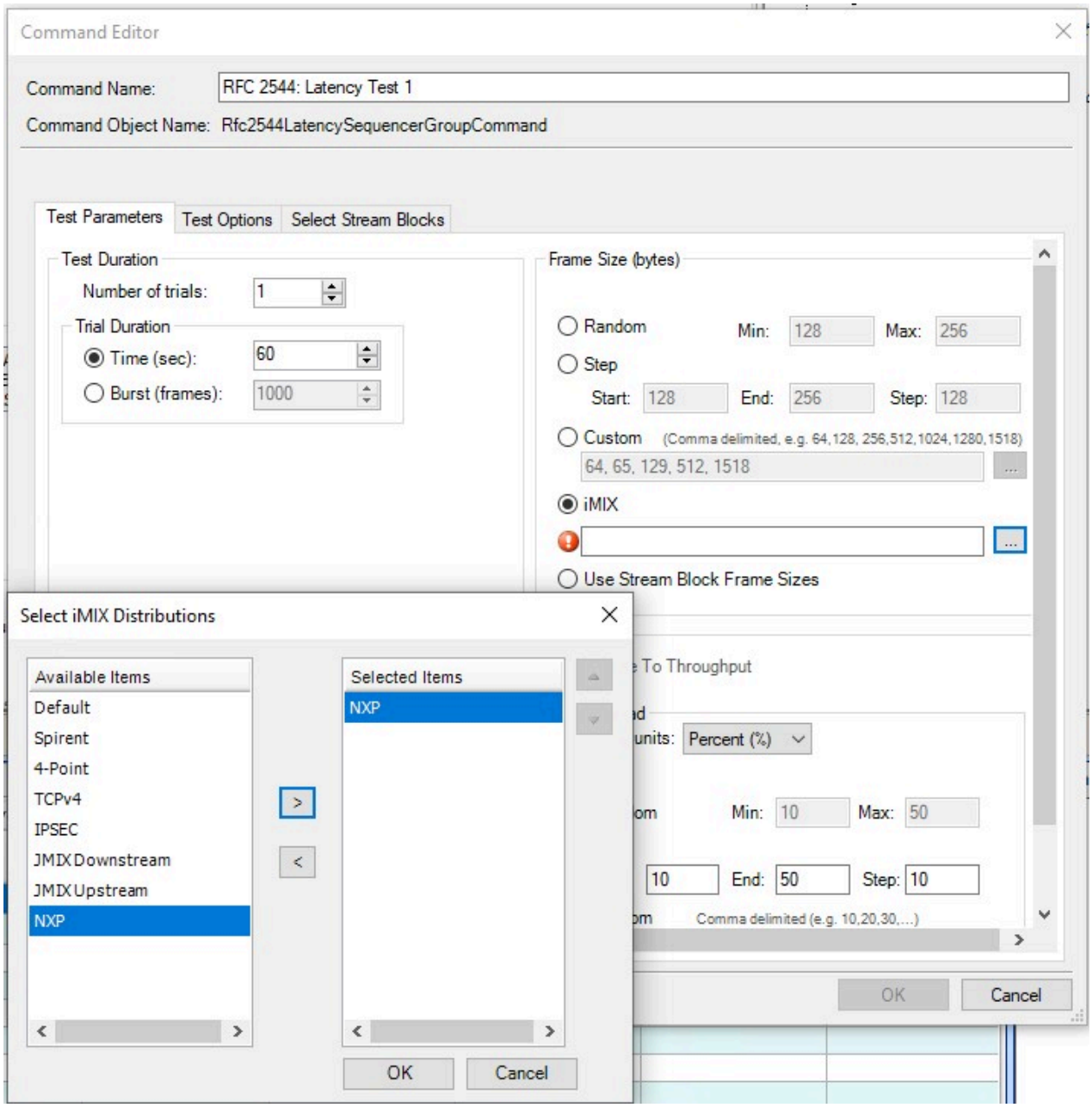


Figure 28. Testing with iMIX

5 Acronyms

Table 2 lists the acronyms used in this document.

Table 2. Acronyms

Acronym	Description
CPU	Central Processing Unit
DUT	Device Under Test
ENET	Ethernet

Table 2. Acronyms...*continued*

Acronym	Description
ENETC	Ethernet Controller
EVK	Evaluation Kit
FIFO	First In First Out
iMIX	Internet MIX
IP	Intellectual Property
LIFO	Last In First Out
NETC	Network Controller
PHY	Physical Layer
POR	Power-On Reset
PPCR	Port Parser Configuration Register
RGMII	Reduced Gigabit Media Independent Interface
SDK	Software Development Kit
STC	Spirent Test Center
TSN	Time-Sensitive Networking

6 Revision history

Table 3 summarizes the revisions to this document.

Table 3. Revision history

Document ID	Release date	Description
AN15056 v.1.0	21 July 2026	Initial public release

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.

Contents