Renesas Synergy™ Platform

Communications Framework on NX Module Guide

R11AN0248EU0100
Rev.1.00
Aug 14, 2018

Introduction

This Module Guide will enable the reader to effectively use a module in their own design. On completion of this guide, the reader will be able to add this module to their own design, configure it correctly for the target application and write code, using the included Application Project code as a reference and an efficient starting point. References to more detailed API descriptions and suggestions of other Application Projects that illustrate more advanced uses of the module are available in the Renesas Synergy Knowledge Base, as described in the References section at the end of this document, and should be valuable resources for creating more complex designs.

The Communications Framework on NX is a high-level API for Communications Framework application and is implemented on NetX Telnet server. The Communications Framework uses the Ethernet peripheral on the Synergy MCU.

Contents

1. Communications Framework on NX Module Features

- High level connectivity is supported on Ethernet but is easily changed to UART and USB connectivity without API modification.
- Supports channel locking for exclusive access. Thread aware implementation uses mutex and event flags internally.

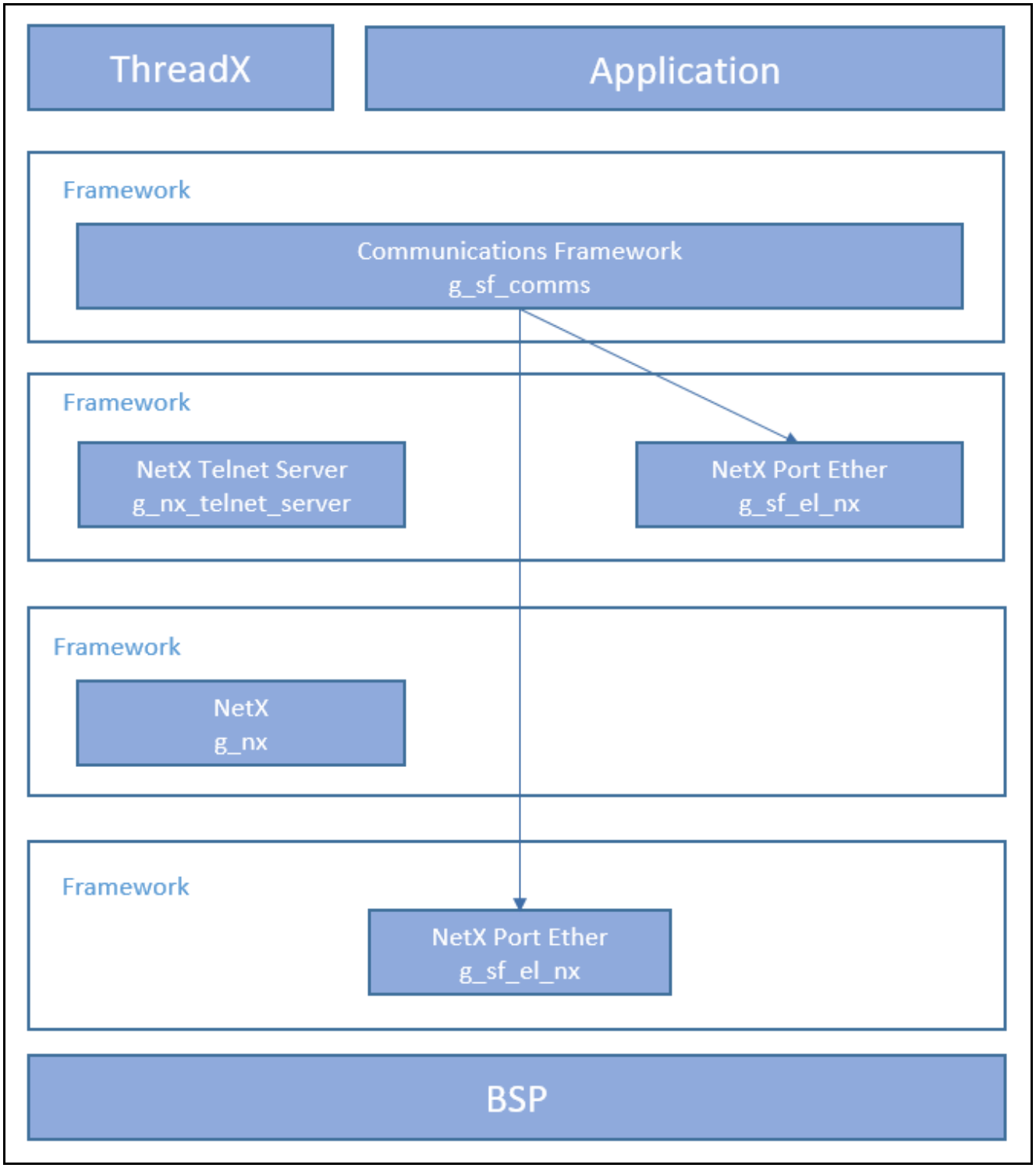

Figure 1 Communications Framework on NX Module Block Diagram

2. Communications Framework on NX Module APIs Overview

The Communications Framework on NX defines APIs for opening, closing, reading and writing over the USB connection. A complete list of the available APIs, an example API call and a short description of each can be found in the following table. A table of status return values follows the API summary table.

Table 1 Communications Framework on NX Module API Summary

Function Name	Example API Call and Description
.open	`g_sf_comms0.p_api->open(g_sf_comms0.p_ctrl, g_sf_comms0.p_cfg);` Initialize communications driver.
.close	`g_sf_comms0.p_api->close(g_sf_comms0.p_ctrl);` Clean up communications driver.
.read	`g_sf_comms0.p_api->read(g_sf_comms0.p_ctrl, &destination, bytes, timeout);` Read data from communications driver. This call will return after the number of bytes requested is read or if a timeout occurs while waiting for access to the driver.
.write	`g_sf_comms0.p_api->write(g_sf_comms0.p_ctrl, &source, bytes, timeout);` Write data to communications driver. This call will return after all bytes are written or if a timeout occurs while waiting for access to the driver.
.lock	`g_sf_comms0.p_api->lock(g_sf_comms0.p_ctrl, lock_type, timeout);` Lock the communications driver. Reserve exclusive access to the communications driver.
.unlock	`g_sf_comms0.p_api->unlock(g_sf_comms0.p_ctrl, lock_type);` Unlock the communications driver. Release exclusive access to the communications driver.
.versionGet	`g_sf_comms0.p_api->version(&version);` Store the driver version in the provided version pointer.

Note: For more complete descriptions of operation and definitions for the function data structures, typedefs, defines, API data, API structures and function variables, review the associated Express Logic User's Manual.

Table 2 Status Return Values

Name	Description
SSP_SUCCESS	Channel opened successfully.
SSP_ERR_IN_USE	Channel already in use.
SSP_ERR_ASSERTION	Pointer to UART control block or configuration structure is NULL.
SSP_ERR_INTERNAL	Internal error occurs.
SSP_ERR_TIMEOUT	Timeout error.
SSP_ERR_NOT_OPEN	Module is not opened.

Note: Lower level drivers may return common error codes. Refer to the *SSP User's Manual* API References for the associated module for a definition of all relevant status return values.

3. Communications Framework on NX Module Operational Overview

The Communications Framework on NX provides an easy to use connection over the Ethernet port. The high-level APIs are compatible with other connection implementations, such as UART and USB, so that it is easy to switch from one implementation to another without changing APIs.

3.1 Communications Framework on NX Module Important Operational Notes and Limitations

3.1.1 Communications Framework on NX Module Operational Notes

The Ethernet peripheral can use either RMII or MII depending on MCU capabilities.

3.1.2 Communications Framework on NX Module Limitations

Refer to the most recent SSP Release Notes for any additional operational limitations for this module.

4. Including the Communications Framework on NX Module in an Application

This section describes how to include the Communications Framework on NX module in an application using the SSP configurator.

Note: This section assumes you are familiar with creating a project, adding threads, adding a stack to a thread and configuring a block within the stack. If you are unfamiliar with any of these items, refer to the first few chapters of the SSP User's Manual to learn how to manage each of these important steps in creating SSP-based applications.

To add the Communications Framework on NX module to an application, simply add it to a thread using the stacks selection sequence given in the following table. The default name for the Communications Framework on NX is g_sf_comms0 and this is shown in the following table. This name can be changed in the associated Properties window.

Table 3 Communications Framework on NX Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
g_sf_comms0 Communications Framework on sf_el_nx_comms	Threads	New Stack> Framework> Connectivity> Communications Framework on sf_el_nx_comms

When the Communications Framework on NX on sf_el_nx_comms module is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level drivers. Any drivers that need additional configuration information will be boxtext highlighted in red. Modules with a gray band are individual modules that stand alone. Modules with a blue band are shared or common and need only be added once and can be used by multiple stacks. Modules with a pink band can require the selection of lower-level drivers; these are either optional or recommended. (This is indicated in the block with the inclusion of this text.) If the addition of lower-level drivers is required, the module description will include **Add** in the text. Clicking on any pink banded modules will bring up the **New** icon and then display the possible choices.

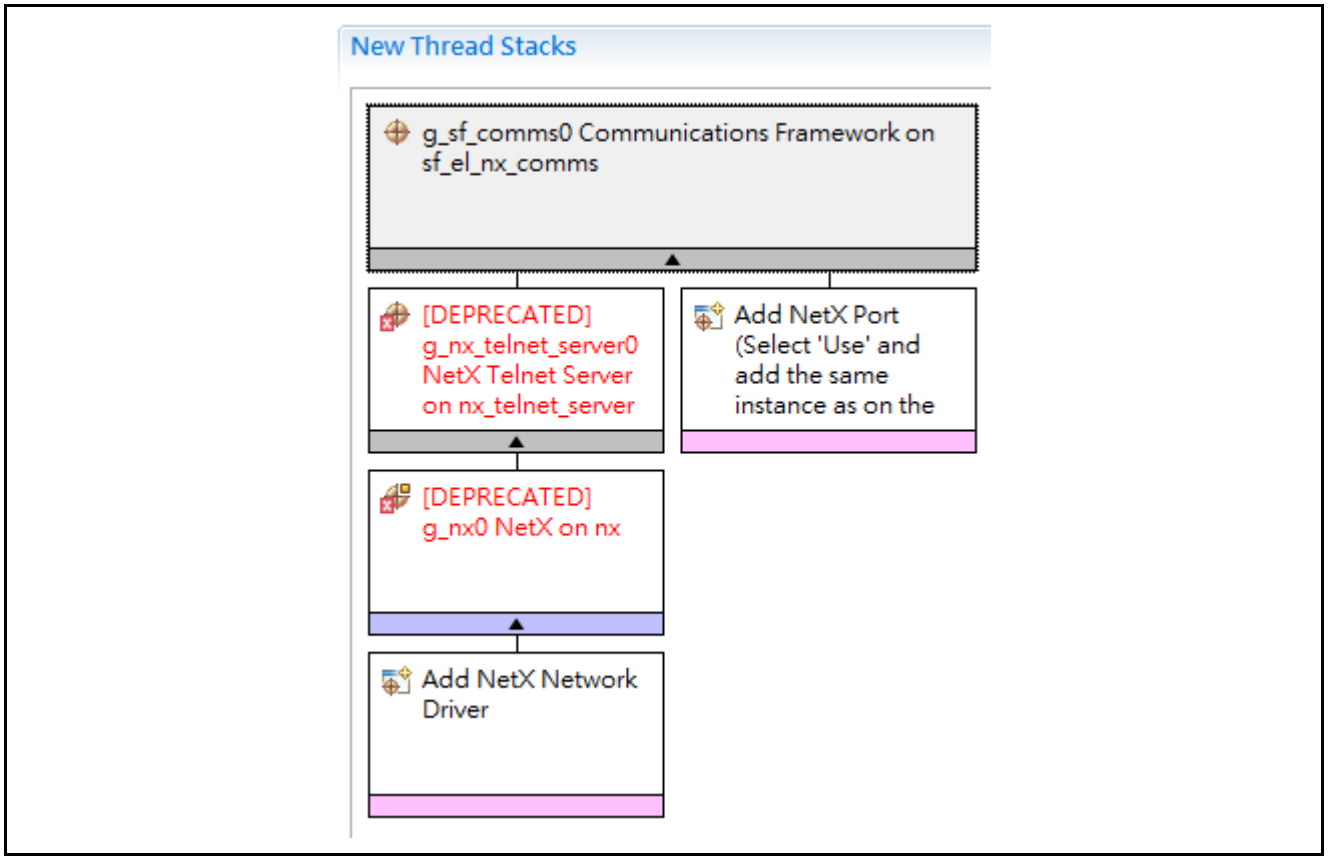

Figure 2 Communications Framework on NX Module Stack

5. Configuring the Communications Framework on NX Module

The Communications Framework on NX module must be configured by the user for the desired operation. The SSP configuration window will automatically identify (by highlighting the block in red) any required configuration selections, such as interrupts or operating modes, which must be configured for lower-level modules for successful operation. Furthermore, only those properties that can be changed without causing conflicts are available for modification. Other properties are 'locked' and are not available for changes, and are identified with a lock icon for the 'locked' property in the Properties window in the ISDE. This approach simplifies the configuration process and makes it much less error-prone than previous 'manual' approaches to configuration. The available configuration settings and defaults for all the user-accessible properties are given in the properties tab within the SSP Configurator and are shown in the following tables for easy reference.

One of the properties most often identified as requiring a change is the interrupt priority; this configuration setting is available within the Properties window of the associated module. Simply select the indicated module and then view the Properties window; the interrupt settings are often toward the bottom of the properties list, so scroll down until they become available. Also note that the interrupt priorities listed in the Properties window in the ISDE will include an indication as to the validity of the setting based on the targeted MCU (CM4 or CM0+). This level of detail is not included in the following configuration properties tables, but is easily visible within the ISDE when configuring interrupt-priority levels.

Note: You may want to open your ISDE, create the module and explore the property settings in parallel with looking over the following configuration table settings. This will help orient you and can be a useful 'hands-on' approach to learning the ins and outs of developing with SSP.

Table 4 Configuration Settings for the Communications Framework on NX Module

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled Default: BSP	Enable or disable the parameter checking
Name	g_sf_comms0	Module name
Channel	0	Underlying channel used by Ethernet driver
IP Address Byte 1	192	IP Address Byte 1 selection
IP Address Byte 2	168	IP Address Byte 2 selection
IP Address Byte 3	0	IP Address Byte 3 selection
IP Address Byte 4	0	IP Address Byte 4 selection
Subnet Mask Byte 1	255	Subnet Mask Byte 1 selection
Subnet Mask Byte 2	255	Subnet Mask Byte 2 selection
Subnet Mask Byte 3	255	Subnet Mask Byte 3 selection
Subnet Mask Byte 4	0	Subnet Mask Byte 4 selection
Name of generated initialized function	sf_comms_init0	Name of generated initialized function
Auto Initialization	Enable, Disable Default: Enable	Auto initialization selection

Note: The example values and defaults are for a project using the Synergy S7G2 Family. Other MCUs may have different default values and available configuration settings.

In some cases, settings other than the defaults for stack modules can be desirable. For example, it might be useful to select different IP Addresses depending on the application. The configurable properties for the lower level stack modules are given in the following sections for completeness, and as a reference.

Note: Most of the property settings for lower-level modules are intuitive and usually can be determined by inspection of the associated properties window from the SSP configurator.

5.1 Configuration Settings for the Communications Framework on NX Lower-Level Modules

Typically, only a small number of settings must be modified from the default for lower-level modules as indicated by the red text in the thread stack block. Notice that some of the configuration properties must be set to a certain value for proper framework operation and will be locked to prevent user modification. The following table identifies all the settings within the properties section for the module.

Table 5 Configuration Settings for the NetX Telnet Server

ISDE Property	Value	Description
Name	g_nx_telnet_server0	Module name
Show deprecation warning	Enabled, Disabled Default: Enabled	Show deprecation warning selection

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs may have different default values and available configuration settings.

Table 6 Configuration Settings for the NetX on nx

ISDE Property	Value	Description
Name	g_nx0	Module name
Show deprecation warning	Enabled, Disabled Default: Enabled	Show deprecation warning selection

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs may have different default values and available configuration settings.

Table 7 Configuration Settings for the NetX Port ETHER

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled Default: BSP	Enable or disable the parameter checking
Channel 0 Phy Reset Pin	IOPORT_PORT_09_PIN_03	Channel 0 Phy reset pin selection
Channel 0 MAC Address High Bits	0x00002E09	Channel 0 MAC address high bits selection
Channel 0 MAC Address Low Bits	0x0A0076C7	Channel 0 MAC address low bits selection
Channel 1 Phy Reset Pin	IOPORT_PORT_07_PIN_06	Channel 1 Phy reset pin selection
Channel 1 MAC Address High Bits	0x00002E09	Channel 1 MAC address high bits selection
Channel 1 MAC Address Low Bits	0x0A0076C8	Channel 1 MAC address low bits selection
Number of Receive Buffer Descriptors	8	Number of receive buffer descriptors selection
Number of Transmit Buffer Descriptors	32	Number of transmit buffer descriptors selection
Ethernet Interrupt Priority	Priority 0 (highest), Priority 1:2, Priority 3 (CM4: valid, CM0+: lowest- not valid if using ThreadX), Priority 4:14 (CM4: valid, CM0+: invalid), Priority 15 (CM4 lowest - not valid if using ThreadX, CM0+: invalid) Default: Disabled	Ethernet interrupt priority selection
Name	g_sf_el_nx	Module name
Channel	0	Channel selection
Callback	NULL	Callback selection

Note: The example values and defaults are for a project using the Synergy S7G2. Other MCUs may have different default values and available configuration settings.

The Communication Framework on NX is implemented using telnet server on ethernet physical layer. Cellular and WiFi framework are not supported by the current communication framework.

5.2 Communications Framework on NX Module Clock Configuration

The ETHERC peripheral module uses PCLKA as its clock source. The PCLKA frequency is set by using the SSP configurator clock tab prior to a build, or by using the CGC Interface at run-time.

5.3 Communications Framework on NX Module Pin Configuration

The ETHERC peripheral module uses pins on the MCU to communicate to external devices. I/O pins must be selected and configured as required by the external device. The following table illustrates the method for selecting the pins within the SSP configuration window and the subsequent table illustrates an example selection for the I2C pins.

Note: The operation mode selection determines what peripheral signals are available and thus what MCU pins are required.

Table 8 Pin Selection for the ETHERC Module

Resource	ISDE Tab	Pin selection Sequence
ETHERC	Pins	Select Peripherals > Connectivity: ETHERC > ETHERC1.RMII

Note: The selection sequence assumes ETHERC1 is the desired hardware target for the driver.

Table 9 Pin Configuration Settings for the ETHERC1

Pin Configuration Property	Value	Description
Operation Mode	Disabled, Custom, RMII Default: Disabled	Select RMII as the Operation Mode for ETHERC1
Pin Group Selection	Mixed, _A only Default: _A only	Pin group selection
REF50CK	P701	REF50CK Pin
TXD0	P700	TXD0 Pin
TXD1	P406	TXD1 Pin
TXD_EN	P405	TXD_EN Pin
RXD0	P702	RXD0 Pin
RXD1	P703	RXD1 Pin
RX_ER	P704	RX_ER Pin
CRS_DV	P705	CRS_DV Pin
MDC	P403	MDC Pin
MDIO	P404	MDIO Pin

Note: The example values are for a project using the Synergy S7G2 and the SK-S7G2 Kit. Other Synergy Kits and other Synergy MCUs may have different available pin configuration settings.

6. Using the Communications Framework on NX Module in an Application

The typical steps in using the Communications Framework on NX Module in an application are:

1. Initialize the Communications Framework on NX using the `open` API.
2. Lock the channel for continuous communications using the `lock` API if needed.
3. Receive data using the `read` API.
4. Send data using the `write` API.
5. Unlock the channel from continuous communication using the unlock command if needed.
6. Close the channel using the `close` API.

These common steps are illustrated in a typical operational flow diagram in the following figure:

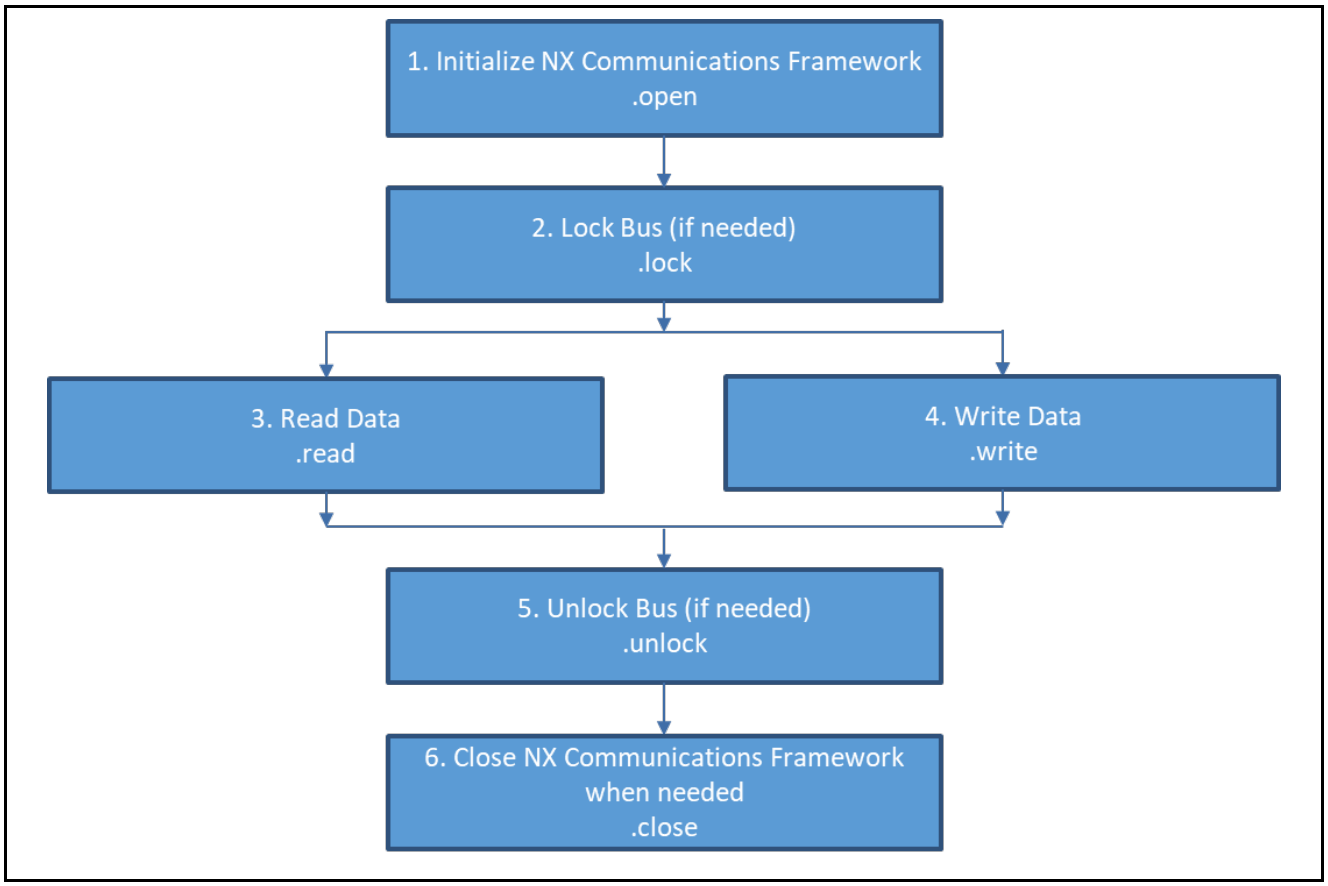

Figure 3 Communications Framework on NX Module

7. The Communications Framework on NX Module Application Project

The Application Project demonstrates the above steps in an example application. You may want to import and open the Application Project within ISDE and view the configuration settings for the Communications Framework on NX module. You can also read over the code, in `nx_comms.c`, which is used to illustrate the Communications Framework on NX Module APIs in a complete design.

The application project demonstrates the typical use of the Communications Framework APIs. It initializes the Communication Framework on NX, sends a welcome message through the framework, and listens for user input. Each time the user presses a proper key (1, 2, or 3), a corresponding LED toggles, and all LED statuses are sent in a message over the framework. A terminal application is required to handle communication with the board and display messages, such as the Tera Term terminal, for instance.

Table 10 Software and Hardware Resources Used by the Application Project

Resource	Revision	Description
e² studio	5.4.0.023	Integrated Solution Development Environment
SSP	1.3.3	Synergy Software Platform
IAR EW for Renesas Synergy	7.71.3	IAR Embedded Workbench for Renesas Synergy
SSC	5.4.0.023	Synergy Standalone Configurator
SK-S7G2	v3.0 to v3.1	Starter Kit

A simple flow diagram of the application project is given in the following figure:

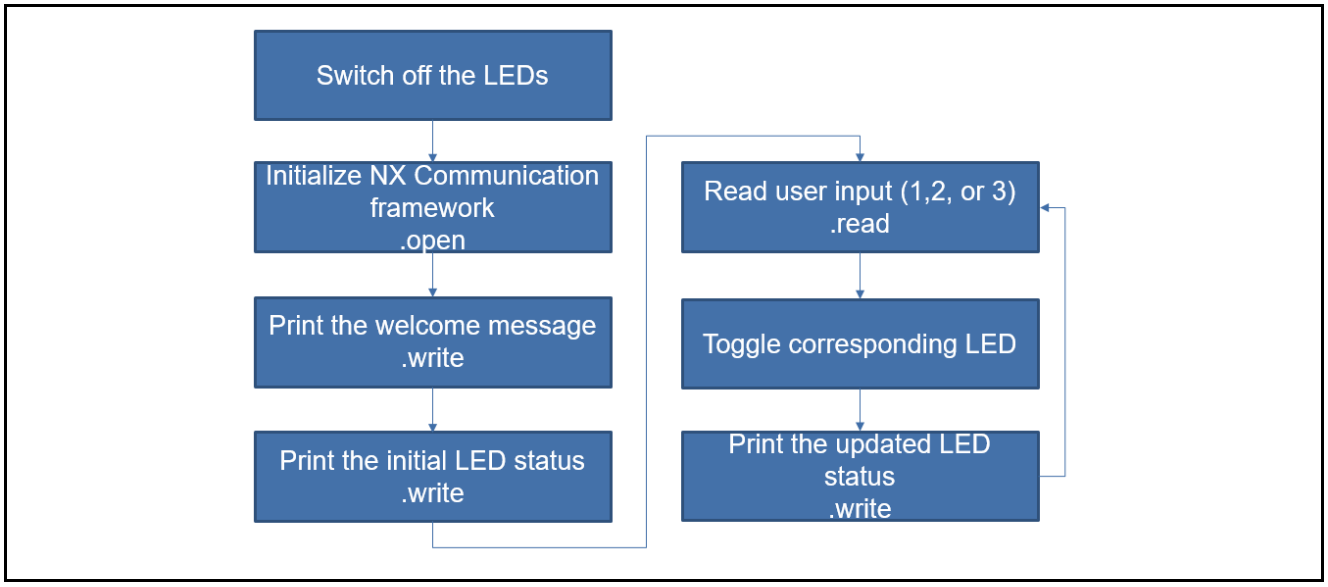

Figure 4 Communication Framework Module Application Project Flow Diagram

The `nx_comms.c` file is located in the project once it has been imported into the ISDE. You can open this file, within the ISDE, and follow along with the below description, to help identify key uses of APIs.

The first section of `nx_comms_ap.c` has the header files which references Communications Framework instance structure and code section with function prototypes and global variables. The next section is `nx_comms_ap()` function which implements the main program control logic. At first, the LED information structure is initialized and all LEDs are turned off. And then the Communication Framework is initialized using `open` API. And the its underlying Telnet Server is set up as well. Note that if auto-initialization of the framework is enable in the configurator, avoid calling the `open` API again. After that, the application sends the welcome message and the initial LED status message through the Communication Framework using the `write` API. The timeout argument of this write API is NX_WAIT_FOREVER. Therefore, the program will wait here until the user establishes a telnet connection and the message is sent to telnet client. The program enters an infinite loop afterwards and in that loop, a key is read using the Communications Framework `read` API. The input is then parsed and translated into a command to toggle LED 1, 2, or 3 in response to receiving keys 1, 2 or 3, respectively. After an LED is toggled, the application sends an LED updated status message using the `write` API.

The following sections are the functions to toggle LED levels, update LED pin levels, and generate and send LED status messages.

A few key properties are configured in this Application Project to support the required operations and the physical properties of the target board and MCU. Below are the properties with the values set for this specific project. You can also open the Application Project and view these settings in the property window as a hands-on exercise.

Provided are tables of settings required to configure the AP for the target MCU and Kit. Use the existing table for an example of standard format and content.

Table 11 Communication Framework on NX Module threads list and parameters

Symbol	Name	Stack Sizes (bytes)	Priority
nx_comms_thread	NX COMMS Thread	1024	3

Table 12 Communication Framework on NX Module Configuration Settings

ISDE Property	Value Set
Name	g_sf_comms
Channel	1
IP Address Byte 1	192
IP Address Byte 2	168
IP Address Byte 3	1
IP Address Byte 4	1
Subnet Mask Byte 1	255
Subnet Mask Byte 2	255
Subnet Mask Byte 3	255
Subnet Mask Byte 4	0
Name of generated initialization function	sf_comms_init0
Auto Initialization	Disable

Current SSP version has yet support the module stacking of the Communication framework on new NetX Telnet Server component. Therefore, user need to use the existing one which is marked as DEPRECATED in red text. This warning can be disable from below configuration setting.

Table 13 NetX Telnet Server Configuration Settings

ISDE Property	Value Set
Name	g_nx_telnet_server0
Show deprecation warning	Disable

Table 14 NetX Telnet Server Configuration Settings

ISDE Property	Value Set
Name	g_nx0
Show deprecation warning	Disable

Table 15 NetX Port ETHER Configuration Settings

ISDE Property	Value Set
Name	g_sf_comms
Channel 0 Phy Reset Pin	IOPORT_PORT_09_PIN_03
Channel 0 MAC Address High Bits	0x00002E09
Channel 0 MAC Address Low Bits	0x0A0076C7
Channel 1 Phy Reset Pin	IOPORT_PORT_08_PIN_06
Channel 1 MAC Address High Bits	0x00002E09
Channel 1 MAC Address Low Bits	0x0A0076C7
Number of Receive Buffer Descriptors	8
Number of Transmit Buffer Descriptors	32
Ethernet Interrupt Priority	Priority 3
Channel	1 (Lock)
Callback	NULL

This project also requires several pin configurations for ethernet. The proper configuration of those pin is shown in the tables below.

Table 16 Pin Selection for the ETHERC Module

Resource	ISDE Tab	Pin selection Sequence
ETHERC	Pins	Select Peripherals > Connectivity: ETHERC > ETHERC1.RMII

8. Customizing the Communications Framework on NX Module for a Target Application

Some configuration settings will normally be changed by the developer from those shown in the Application Project. For example, the user can easily change the IP address of telnet server.

9. Running the Communications Framework on NX Module Application Project

To run the Communications Framework on NX Module application project and to see it executing on a target kit, you can simply import it into your ISDE, compile and run debug.

Note: The below steps are described in sufficient detail for someone experienced with the basic flow through the Synergy development process. If these steps are not familiar, refer to the first few chapters of the SSP User's Manual for a description of how to accomplish these steps.

To run the application project simply follow these steps:

1. Import and build the example project included with this module guide according to the Synergy Project Import Guide, r11an0023eu0120-synergy-ssp-import-guide.pdf.
2. Click on the **Generate Project Content** button.
3. Compile the project.
4. Connect to the host PC through a micro USB cable to J19 on the SK-S7G2 board.
5. Start to debug the application.
6. Setup the local area network connection properties as shown in Figure 5.

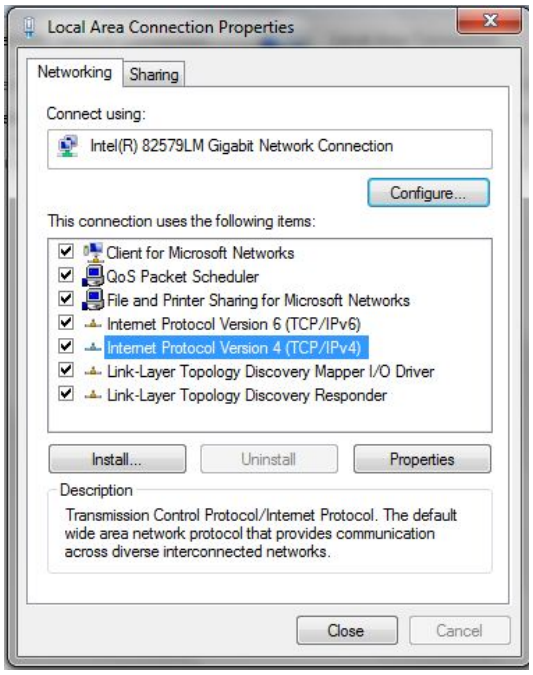

Figure 5 The Procedure to Configure Local Network Properties

7. Start the Tera Term application and establish a connection on the pre-configured IP address as shown in Figure 6.

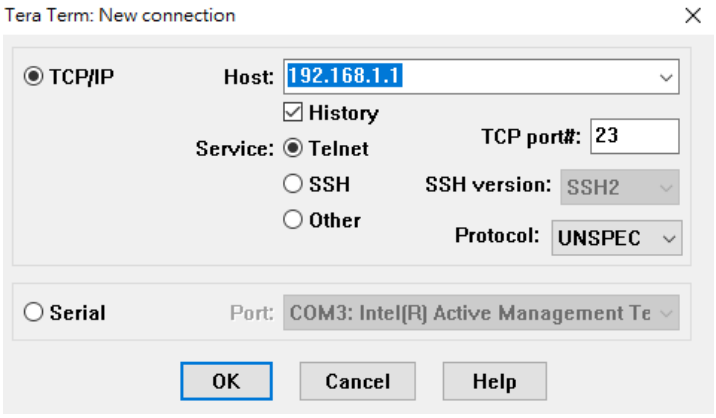

Figure 6 The Procedure to Configure Terminal Tool Settings

8. Press 1, 2 or 3 on the PC keyboard. The output can be viewed in the Tera Term application as shown in the following figure 7.

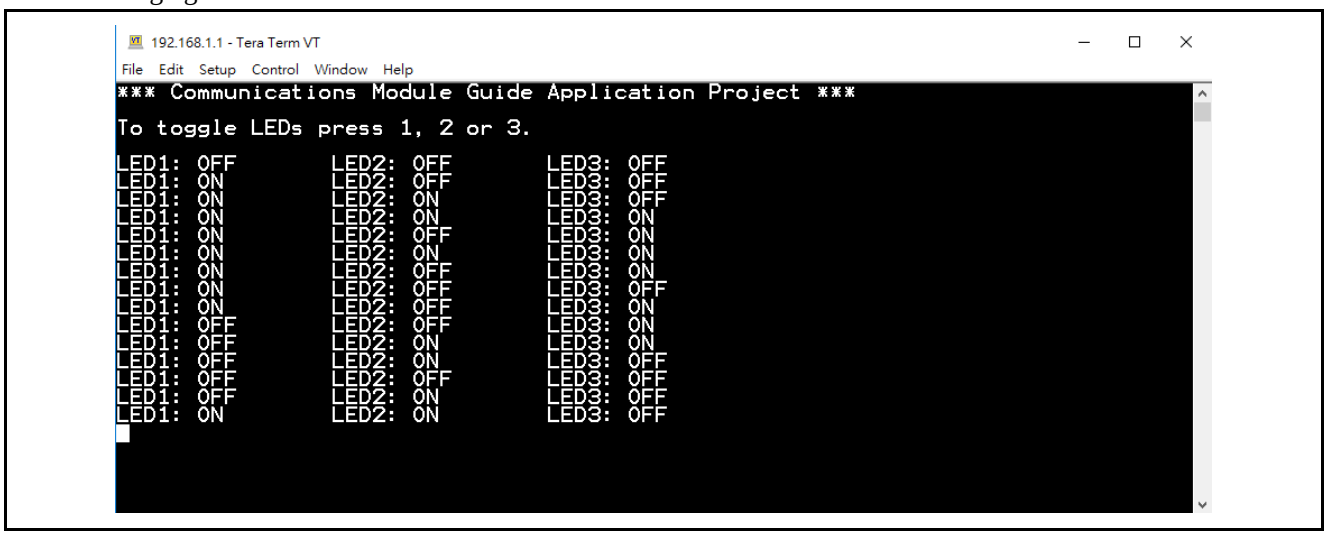

Figure 7 Example Output from NX Communication Module Application Project

10. Communications Framework on NX Module Conclusion

This Module Guide has provided all the background information needed to select, add, configure and use the module in an example project. Many of these steps were time consuming and error prone activities in previous generations of embedded systems. The Renesas Synergy Platform makes these steps much less time consuming and removes the common errors, like conflicting configuration settings or incorrect selection of low level drivers. The use of high level APIs, as demonstrated in the Application Project illustrate additional development time savings by allowing work to begin at a high level, avoiding the time required in older development environments to use or, in some cases, create low level drivers.

11. Communications Framework on NX Module Next Steps

After you have mastered a Communications Framework on NX Module project, you may want to review alternative Communications Framework implementations on USBX or UART.

12. Communications Framework on NX Module Reference Information

SSP User Manual: Available in html format in the SSP distribution package and as a pdf from the Synergy Gallery.

Links to all the most up-to-date Communications Framework on NX Module reference materials and resources are available on the Synergy Knowledge Base: https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy%E2%84%A2_Platform/Renesas_Synergy_Knowledge_Base/sf_el_nx_comms_Module_Guide_Resources

Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

Synergy Software	renesassynergy.com/software
Synergy Software Package	renesassynergy.com/ssp
Software add-ons	renesassynergy.com/addons
Software glossary	renesassynergy.com/softwareglossary
Development tools	renesassynergy.com/tools
Synergy Hardware	renesassynergy.com/hardware
Microcontrollers	renesassynergy.com/mcus
MCU glossary	renesassynergy.com/mcuglossary
Parametric search	renesassynergy.com/parametric
Kits	renesassynergy.com/kits
Synergy Solutions Gallery	renesassynergy.com/solutionsgallery
Partner projects	renesassynergy.com/partnerprojects
Application projects	renesassynergy.com/applicationprojects
Self-service support resources:	
Documentation	renesassynergy.com/docs
Knowledgebase	renesassynergy.com/knowledgebase
Forums	renesassynergy.com/forum
Training	renesassynergy.com/training
Videos	renesassynergy.com/videos
Chat and web ticket	renesassynergy.com/support

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Aug 14, 2018	—	Initial release

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

RENESAS

SALES OFFICES Renesas Electronics Corporation http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.

Renesas Electronics America Inc.
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

Renesas Electronics Canada Limited
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

Renesas Electronics Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700

Renesas Electronics Europe GmbH
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

Renesas Electronics (China) Co., Ltd.
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

Renesas Electronics (Shanghai) Co., Ltd.
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

Renesas Electronics Hong Kong Limited
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

Renesas Electronics Taiwan Co., Ltd.
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

Renesas Electronics Singapore Pte. Ltd.
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

Renesas Electronics Malaysia Sdn.Bhd.
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

Renesas Electronics India Pvt. Ltd.
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338

Colophon 7.1