Renesas Synergy™ Platform

Telnet Communications Framework on sf_comms_telnet for SSP v1.4.0 Module Overview

R11AN0340EU0100
Rev.1.00
Sep 29, 2018

Telnet Communications Framework Introduction

The Communications Framework on NX provides a high-level APIs for communications framework using the ThreadX RTOS. The framework is currently implemented on the Telnet Server on NetX Duo. The Telnet communications Framework uses the Ethernet peripheral on the Synergy MCU.

Telnet Communications Framework Features

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

Contents

1. Telnet Communications Framework Module APIs Overview

The Telnet Communications Framework module defines APIs to open, read from, write to, lock, unlock, and close the module. 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.

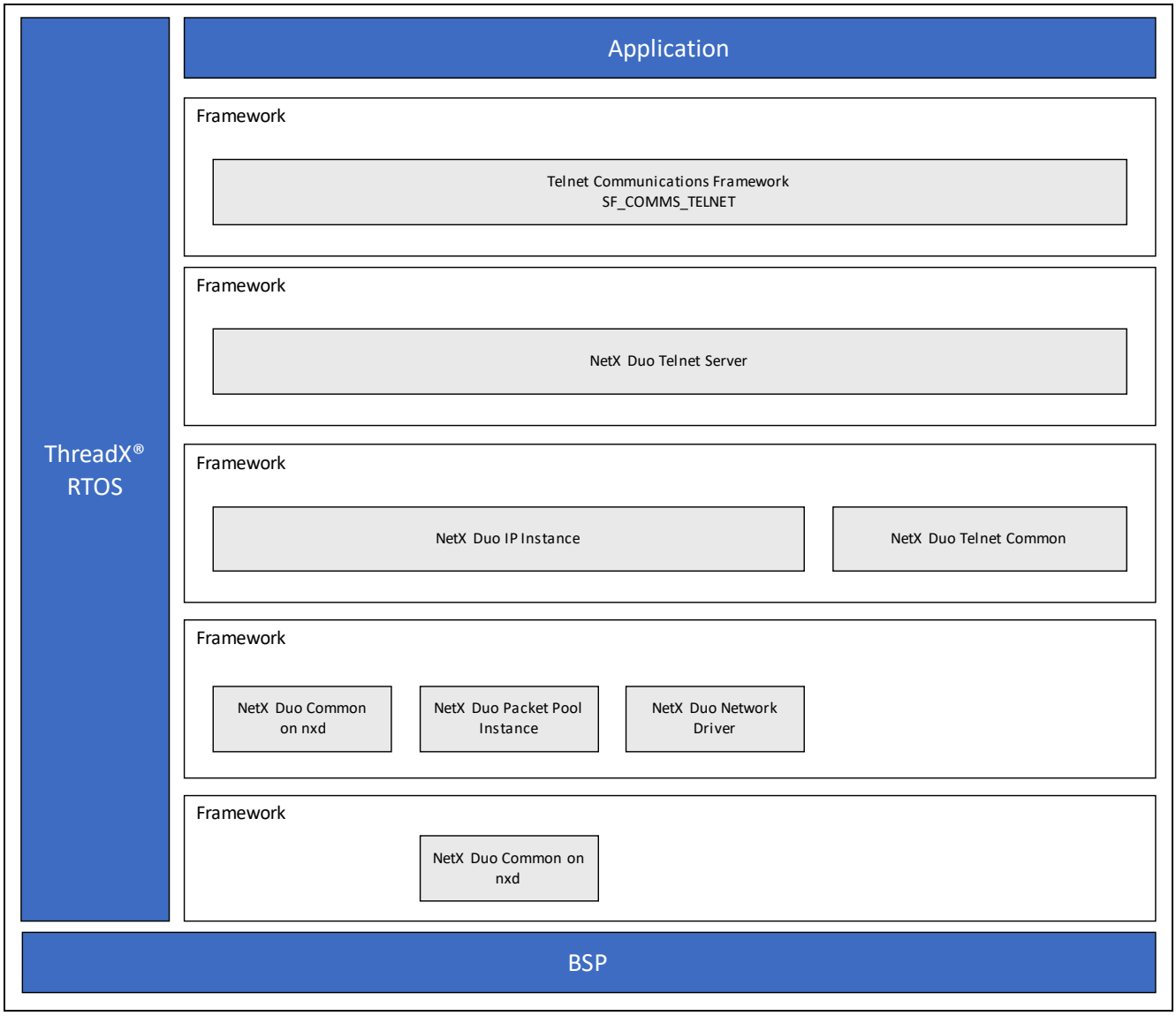

Figure 1 Telnet Communications Framework Module Block Diagram

Table 1 Telnet Communications Framework Module API Summary

Function Name	Example API Call and Description
.open	g_sf_comms_telnet0.p_api->open (g_sf_comms_telnet0.p_ctrl, g_sf_comms_telnet0.p_cfg); Initialize module.
.read	g_sf_comms_telnet0.p_api->read (g_sf_comms_telnet0.p_ctrl, p_dest, bytes, timeout); Read a number of bytes of data into the destination.
.write	g_sf_comms_telnet0.p_api->write (g_sf_comms_telnet0.p_ctrl, p_src, bytes, timeout); Write a number of bytes from the source.
.lock	g_sf_comms_telnet0.p_api->lock (g_sf_comms_telnet0.p_ctrl, locktype, timeout); Acquire lock type for the Telnet comms instance.
.unlock	g_sf_comms_telnet0.p_api->unlock (g_sf_comms_telnet0.p_ctrl, locktype); Release the lock type for the Telnet comms instance.
.close	g_sf_comms_telnet0.p_api->close (g_sf_comms_telnet0.p_ctrl); Disconnect Telnet server and clean up resources.
.versionGet	g_ sf_comms_telnet0.p_api->versionGet(&version); Gets version and stores it in 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 *SSP User's Manual* API References for the associated module.

Table 2 Status Return Values

Name	Description
SSP_SUCCESS	API Call Successful.
SSP_ERR_INVALID_ARGUMENT	Parameter has invalid value.
SSP_ERR_INTERNAL	An internal TheadX or NetX error has occurred.
SSP_ERR_NOT_OPEN	Unit is not open
SSP_ERR_ASSERTION	A parameter is NULL.
SSP_ERR_IN_USE	Peripheral is still running in another mode; perform Close first.
SSP_ERR_UNSUPPORTED	Command not supported.
SSP_ERR_OUT_OF_MEMORY	Can't allocate pool memory.
SSP_ERR_TIMEOUT	An event timed out.

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.

2. Telnet Communications Framework Module Operational Overview

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

Operations supported by the framework include initializing the module using the `open` API, and closing the module using the `close` API. A communications read is implemented by the `read` API and a communications write by the `write` API. The `read` and `write` lock the module only while the API is in action.

The `lock` API locks the module until the `unlock` API is called on the same module instance. This helps ensure processing is completed before moving to the next API function call.

The underlying NetX driver supports the configuration of the IP address, the Network mask, and the Ethernet channel. When a different communications implementation is used (like USB), different low-level modules configuration settings are used to define its interface. Thus, no code needs to be changed in the application, just changes to configuration settings. The same API calls are retained at the application level.

2.1 Telnet Communications Framework Module Important Operational Notes and Limitations

2.1.1 Telnet Communications Framework Module Operational Notes

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

2.1.2 Telnet Communications Framework Module Limitations

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

3. Including the Telnet Communications Framework Module in an Application

This section describes how to include the Telnet Communications Framework 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 Telnet Communications Framework module to an application, simply add it to a thread using the stack's selection sequence given in the following table. (The default name for the Telnet Communications Framework module is sf_comms_telnet0. This name can be changed in the associated **Properties** window.)

Table 3 Telnet Communications Framework Module Selection Sequence

Resource	ISDE Tab	Stacks Selection Sequence
sf_comms_telnet0 Communications Framework on sf_comms_telnet	Threads	New Stack> Framework> Connectivity> Communications Framework on sf_comms_telnet

When the Communications Framework module on sf_comms_telnet is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level modules. Any modules needing additional configuration information have the box text highlighted in Red. Modules with a Gray band are individual modules that stand alone. Modules with a Blue band are shared or common; they need only be added once and can be used by multiple stacks. Modules with a Pink band can require the selection of lower-level modules; these are either optional or recommended. (This is indicated in the block with the inclusion of this text.) If the addition of lower-level modules is required, the module description include Add in the text. Clicking on any Pink banded modules brings up the New icon and displays possible choices.

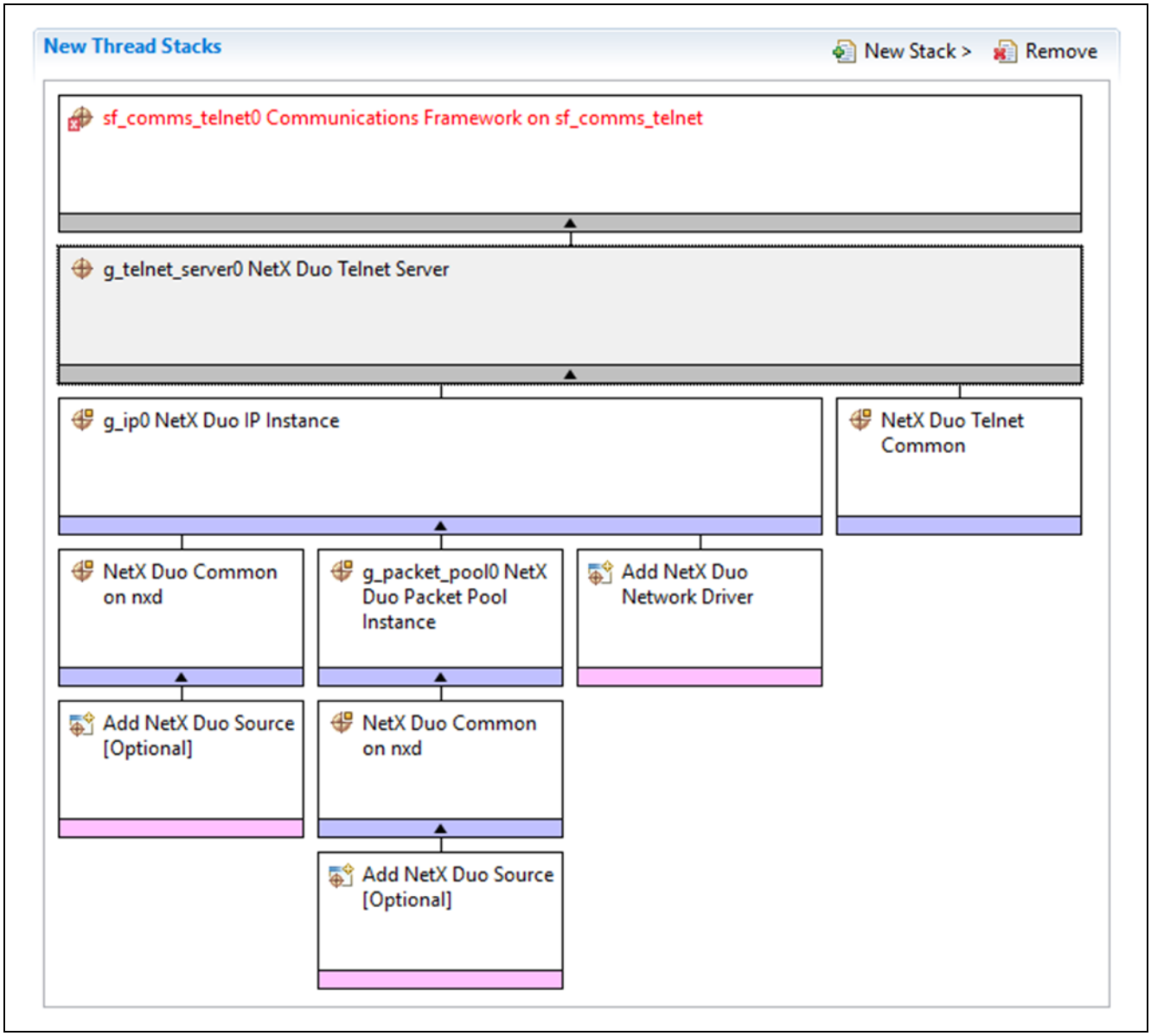

Figure 2 Telnet Communications Framework Module Stack

4. Configuring the Telnet Communications Framework Module

The Telnet Communications Framework 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 to ensure 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 the SSP.

Table 4 Configuration Settings for the Telnet Communications Framework Module on sf_comms_telnet

ISDE Property	Value	Description
Parameter Checking	BSP, Enabled, Disabled Default: BSP	Enable or disable the parameter checking
Packet size in pool memory (bytes)	1536	Packet size in pool memory selection
Packets to allocate in pool memory (units)	5	Packets to allocate in pool memory selection
Timeout for internal options (ticks)	10	Timeout for internal options selection
Maximum number of instances	4	Maximum instances that can be open at any given time
Name	sf_comms_telnet0	Module name
Name of generated initialization function	sf_comms_telnet_init0	Name of generated initialization selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization 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.

In some cases, settings other than the defaults for lower-level modules can be desirable. 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 fairly intuitive and can usually be determined by inspection of the associated **Properties** window from the SSP configurator.

4.1 Configuring the Telnet Communications Framework Lower-Level Modules

Typically, only a small number of settings must be modified from the default for lower-level drivers as indicated with 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/NetX Duo Telnet Server

ISDE Property	Value	Description
Internal thread priority	16	Internal thread priority selection
Maximum clients to serve simultaneously	4	Maximum clients to serve simultaneously selection
Socket window size (bytes)	2048	Socket window size selection
Server time out (seconds)	10	Duration internal services will suspend for
Client inactivity timeout (seconds)	600	Client inactivity duration for disconnection
Timeout check period (seconds)	60	Client activity timeout check interval
Option negotiation	Enable, Disable Default: Enable	Option negotiation selection
Use application packet pool	Enable, Disable Default: Disable	Use application packet pool selection
Packet size in the pool (bytes)	300	Telnet Server only creates this packet pool if 'Option negotiation' is enabled
Total packet pool size (bytes)	2048	Telnet Server only creates this packet pool if NX_TELNET_SERVER_OPTION_DISABLE
Name	g_telnet_server0	Module name
Thread Stack Size (bytes)	2048	Thread stack size selection
Name of Client Connect Callback Function	NULL	Name of client connect callback function selection
Name of Receive Data Callback Function	NULL	Name of receive data callback function selection
Name of Client Disconnect Callback Function	NULL	Name of client disconnect callback function selection
Name of generated initialization function	telnet_server_init0	Name of generated initialization function selection
Auto Initialization	Disable	Auto initialization 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/NetX Duo IP Instance

ISDE Property	Value	Description
Name	g_ip0	Module name
IPv4 Address (use commas for separation)	192,168,0,2	IPv4 Address selection
Subnet Mask (use commas for separation)	255,255,255,0	Subnet Mask selection
**IPv6 Global Address (use commas for separation)	0x2001, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x1	IPv6 global address selection
**IPv6 Link Local Address (use commas for separation, All zeros means use MAC address)	0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0, 0x0	IPv6 link local address selection
IP Helper Thread Stack Size (bytes)	2048	IP Helper Thread Stack Size (bytes) selection
IP Helper Thread Priority	3	IP Helper Thread Priority selection
ARP	Enable	ARP selection
ARP Cache Size (bytes)	512	ARP Cache Size in Bytes selection
Reverse ARP	Enable, Disable Default: Disable	Reverse ARP selection
TCP	Enable	TCP selection
UDP	Enable, Disable Default: Enable	UDP selection
ICMP	Enable, Disable Default: Enable	ICMP selection
IGMP	Enable, Disable Default: Enable	IGMP selection
IP fragmentation	Enable, Disable Default: Disable	IP fragmentation selection
Name of generated initialization function	ip_init0	Name of generated initialization function selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization selection
Link status change callback	NULL	Link status change 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.

** Indicates properties that are only available in NetX Duo.

Table 7 Configuration Settings NetX/NetX Duo Telnet Common

ISDE Property	Value	Description
Type of Service for TCP requests	Normal, Minimum delay, Maximum data, Maximum reliability, Minimum cost Default: Normal	Type of service TCP requests selection
Fragmentation option	Don't fragment, Fragment okay Default: Don't fragment	Fragment option selection
Server TCP port number	23	Server TCP port number selection
Time to live	128	Time to live 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 8 Configuration Settings for the NetX/NetX Duo Common on nx/nxd

ISDE Property	Value	Description
Name of generated initialization function	nx_common_init0	Name of generated initialization function selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization 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 9 Configuration Settings for the NetX/NetX Duo Packet Pool Instance

ISDE Property	Value	Description
Name	g_packet_pool0	Module name
Packet Size (bytes)	640	Packet size selection
Number of Packets in Pool	16	Number of packets in pool selection
Name of generated initialization function	packet_pool_init0	Name of generated initialization function selection
Auto Initialization	Enable, Disable Default: Enable	Auto initialization 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 10 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 1:14, Priority 15 (lowest - not valid if using ThreadX) Default: Priority 12	Ethernet interrupt priority selection
Link status monitoring method	PHY Interrupt (Uses LINKSTA Pin), PHY Polling Default: PHY Polling	Link status monitoring method selection
Name	g_sf_el_nx	Module name
Channel	0	Channel selection
MAC address change callback	NULL	MAC address change callback selection
Unknown packet receive Callback	NULL	Unknown packet receive callback selection

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

4.2 Telnet Communications Framework Module Clock Configuration

The Telnet Communications Framework module uses the Ethernet peripheral which uses the PCLKA as its clock source.

To change the clock frequency at run-time, use the CGC Interface.

4.3 Telnet Communications Framework Module Pin Configuration

To use the Telnet Communications Framework module, the port pins for the peripheral inputs and outputs must be set in the pin configurator in the ISDE. The following table illustrates the method for selecting the pins within the ISDE configuration window:

Table 11 Pin Selection Sequence for the Telnet Communications Framework Module

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

Note: The selection sequences are examples for selected implementations. Others are also possible depending on the target hardware.

Table 12 Pin Configuration Settings for EHTERC1

Property	Value	Description
Pin Group Selection	Mixed, _A only (Default: _A only)	Pin group selection
Operation Mode	Disable, Custom, RMII (Default: _A only)	Select RMII as the operation Mode for EHTERC1
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

5. Using the Telnet Communications Framework Module in an Application

The typical steps in using the Telnet Communications Framework module in an application are:

1. Initialize the Telnet Communications Framework 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` API if needed
6. Close the channel using the `close` API

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

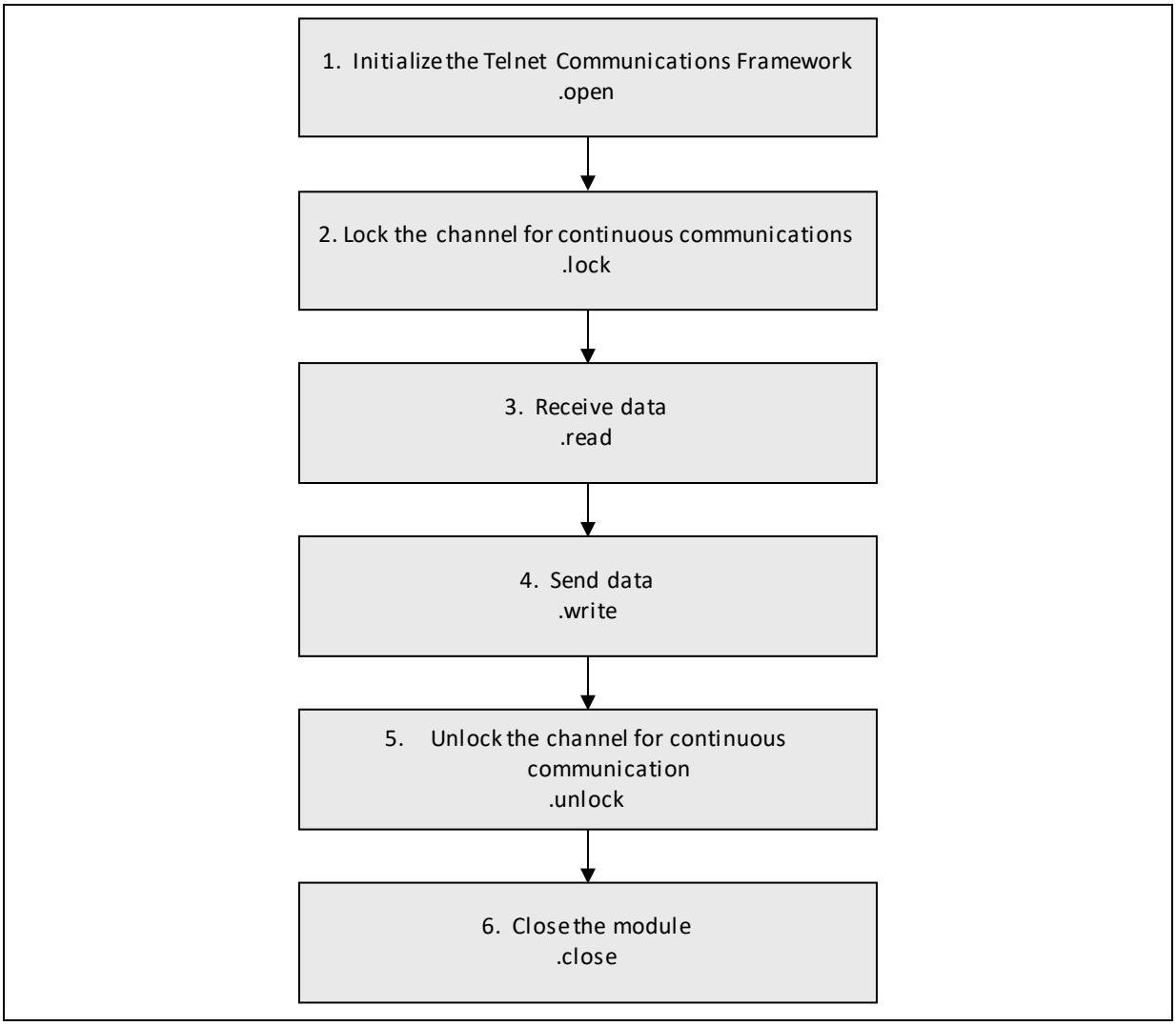

Figure 3 Flow Diagram of a Typical Telnet Communications Framework Module Application

6. Telnet Communications Framework Module Application Project

The application project associated with this module guide demonstrates the steps in a full design. The project can be found using the link provided in the References section at the end of this document. You may want to import and open the application project within the ISDE and view the configuration settings for the Telnet Communications Framework module. You can also read over the code in `com_telnet_thread_entry.c` and `com_telnet_fw_mg.c` which are used to illustrate the Telnet Communications Framework module APIs in a complete design.

The application project demonstrates the typical use of the Telnet Communications Framework module APIs. The application project Communication Telnet Thread creates the NetX Duo IP instance, NetX Duo Telnet Server instance, as well as the Telnet communication instance automatically. Then the project initializes the Telnet Communications Framework module, and send a welcome message to a Telnet client. Lock and unlock function can be called if necessary. After that, it enters an infinite while loop, Telnet Communications Framework module is waiting for character sent from Telnet client and echo client with received character. If ‘q’ is received, Telnet Communications Framework module is closed and the project jumps out of the infinite loop. Then all process will be re-started from the initialization step.

The following table identifies the target versions for the associated software and hardware used by the application project:

Table 13 Software and Hardware Resources Used by the Application Project

Resource	Revision	Description
e^2 studio	6.2.0 or later	Integrated Solution Development Environment
SSP	1.4.0 or later	Synergy Software Platform
IAR EW for Renesas Synergy	8.21.1 or later	IAR Embedded Workbench for Renesas Synergy
SSC	6.2.0 or later	Synergy Standalone Configurator
SK-S7G2	v3.0 to v3.3	Starter Kit

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

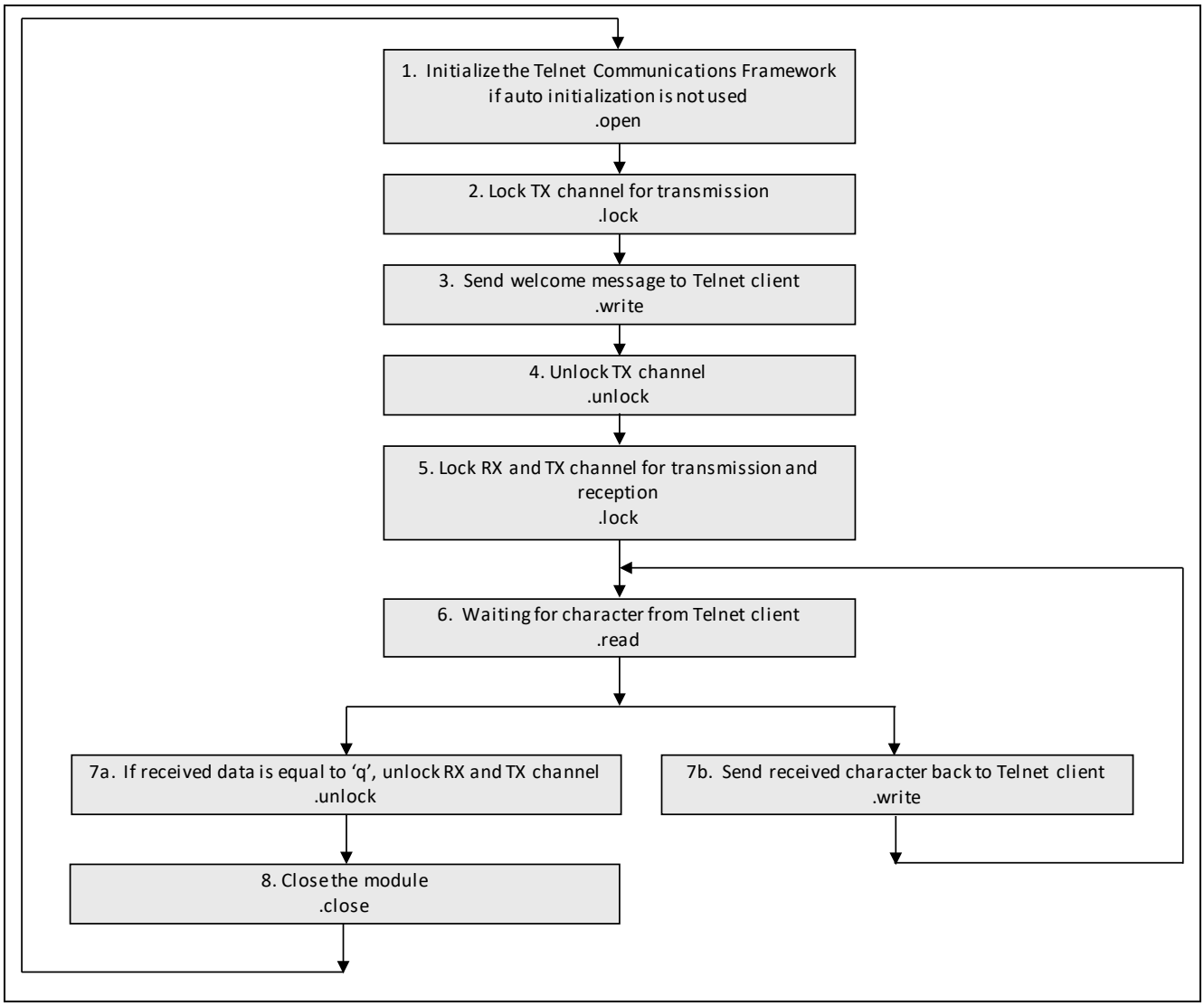

Figure 4 Telnet Communications Framework Module Application Project Flow Diagram

The first section of `com_telnet_thread_entry.c` has the header file that references the Telnet comms instance. The following section is the entry function which calls the main program-control function `telnet_start_process()` from the `com_telnet_fw_mg.c` file. The Telnet Communications Framework module is initialized using the `open` API when Auto Initialization properties item in the configuration is set to Disable. For exclusive use of the TX or RX function, the `lock` or `unlock` API is used to acquire or release lock type for the Telnet comms instance. The welcome message will be sent to Telnet client using the `write` API after successfully opening the Telnet Communications Framework module. After connecting with the Telnet client, Telnet Communications Framework module will enter an infinite loop, which is waiting for character from client and echo with same character received from client. If 'q' is received, Telnet Communications Framework will be closed, and the

project jump out of this infinite loop. Then the process will start over from the beginning. All error or correct operation messages can be output to Debug Virtual Console if SEMI_HOSTING is defined.

Note: This description assumes you are familiar with using `printf()` with the Debug Console in the Synergy Software Package. If you are unfamiliar with this, refer to the "How do I Use Printf() with the Debug Console in the Synergy Software Package" Knowledge Base article, available as described in the References section at the end of this document. Alternatively, the user can see results via the watch variables in the debug mode.

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. The properties with the values set for this specific project are listed in the following table. You can also open the application project and view these settings in the **Properties** window as a hands-on exercise.

Table 14 Telnet Communications Framework Module Configuration Settings for the Application Project

ISDE Property	Value Set
Properties of Com Telnet Thread	
Stack size (bytes)	2048
Properties of sf_comms_telnet0 Communications Framework on sf_comms_telnet	
Auto Initialization	Disable
Properties of g_telnet_server0 NetX Duo Telnet Server	
Maximum clients to serve simultaneously	1
Properties of g_sf_el_nx NetX Port ETHER on sf_el_nx	
Channel 1 Phy Reset Pin	IOPORT_PORT_08_PIN_06
Channel	1

7. Customizing the Telnet Communications Framework Module for a Target Application

The developer will normally change some configuration settings from those shown in the application project. For example, you can easily change the configuration settings for the NetX IP Instance. You can use a NetX DHCP Client to obtain an IP address provided by a DHCP server in the network to replace a static IP in this project. You can also set Auto Initialization item in Communications Framework Properties to automatically initialize module instead of calling `open` API.

8. Running the Telnet Communications Framework Module Application Project

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

Note: The following 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 create and run the Telnet Communications Framework Module 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* 11an0023eu0116-synergy-ssp-import-guide.pdf.
2. Connect to the host PC using the USB cable (use J19 DEBUG_USB connector)
3. Connect to the host PC using the USB cable to J5 on the SK-S7G2 board
4. Connect the host PC via an Ethernet cable to J11 on the SK-S7G2 board
5. Start to debug the application.
6. Open a terminal application (PuTTY, TeraTerm or others) and connect to a Telnet Server using the chosen IP address. In case of a failure, make sure you have a statically defined IP address on your client machine and your firewall is disabled.
7. The output can be viewed in the Renesas Debug Console.

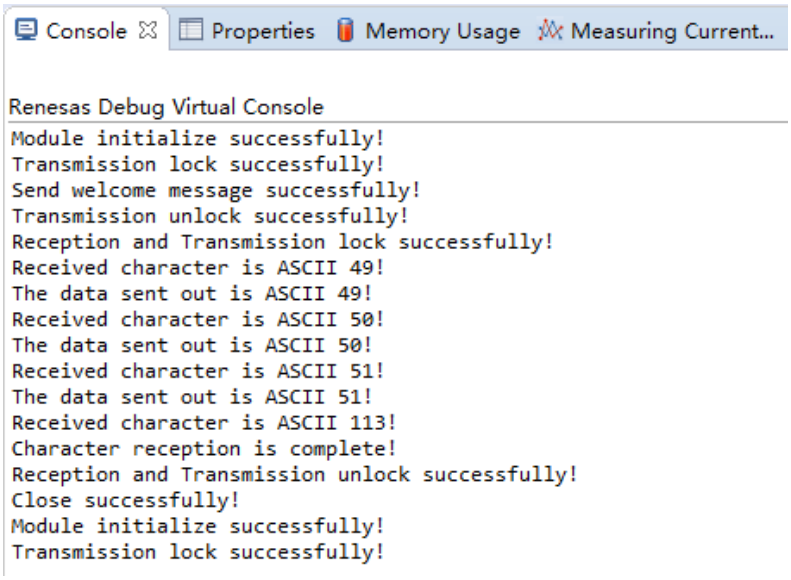

Figure 5 Debug Console Output

8. The welcome message can be viewed in the terminal application window. If you press 1, 2, and 3 on the telnet client, the echoed characters also can be viewed in this window.

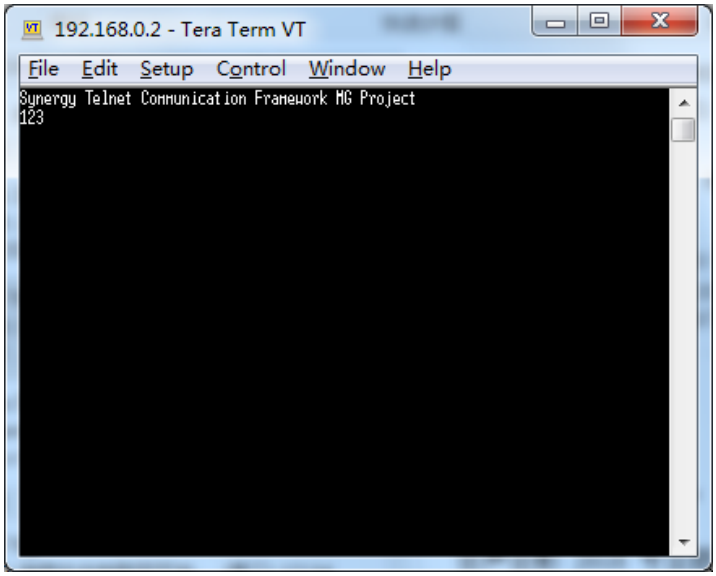

Figure 6 Example Output from Telnet Communications Framework Module Application Project

9. Telnet Communications Framework 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 lower-level drivers. The use of high-level APIs (as demonstrated in the application project) illustrates additional development-time savings by allowing work to begin at a high level and avoiding the time required in older development environments to use, or, in some cases, create, lower-level drivers.

10. Telnet Communications Framework Module Next Steps

After you have mastered a simple Telnet communication framework module project, you may want to review a more complex example. The use of the Communications Framework implemented on other interfaces such as USBX and

UART can be found on Renesas website. Other application projects and application notes that demonstrate NetX/NetX Duo module use can be found as described in the following References section.

You may find that the setting a DHCP Client is a better fit for your target application than setting a static IP. The NetX DHCP Client Module Guide illustrates how to use it to obtain a dynamic provided IP address from a DHCP Server.

11. Telnet Communications Framework 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 Telnet Communications Framework 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/R_ADC_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	Sep 29, 2018	—	First release document

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 Corporation
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

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 7.2