RH850/U2B Group

R01AN6571EJ0100
Rev.1.00

RHSIF Application Note

1 Introduction

The Renesas High-Speed Interface (RHSIF) is a point-to-point high speed serial communication between two devices.

It supports autonomous handling of the physical layer and the transfer of application data on higher communication layers. In this application note the basic initialization of the interface and simple transmission and reception operations are shown with two RH850 MCU devices. The necessary steps can be easily adapted to other Renesas RH850 MCU with a build-in RHSIF IP.

1.1 Target Device

- RH850/U2Bx Series

The SW examples in this application note are implemented on RH850/U2Bx devices.

2 Table of Contents

3 Table of Figures

4 Table of Tables

5 The RHSIF Module

5.1 General Interface Overview

The RHSIF interface is a full duplex asynchronous interface. It is used as a direct communication interface between two devices. In this document the two devices will be named ‘communication partners’ or ‘partners’.

On the physical level the RHSIF uses five signals:

- Reference clock: RHSIF0_REFCLK
- Receive data differential input: RHSIF0_RXDP
RHSIF0_RXDN
- Transmission data differential output: RHSIF_TXDP
RHSIF_TXDN

The reference clock is a 10/20MHz signal with the full voltage swing of the port supply voltage (e.g. 3.3V or 5V). The differential signals operate at an offset voltage of (typ.) 1.2V with a voltage swing of (typ.) ±100mV. The bus speed can be up to 320MHz (depending on the device specification).

The reference clock is used for the generation of the bus clock / baud rate of the interface. A dedicated PLL is available for that purpose. The reference clock is output by one of the two communication partners and is received by the other communication partner. The outputting partner is called the *master device*, the receiving partner is called the *slave device*. This master/slave terminology is only important for the configuration of the reference clock.
For the later communication both devices can be considered as equal partners. Therefore the terminology of *initiator node* and *target node* will be used.

The interface feature of *full duplex mode* means that data can be transmitted (using the TXD signals) and received (using the RXD signals) at the same time.
The feature of *asynchronous* interface means that the transmission and the reception of data is fully independent of each other. [1]

5.2 Module Block Diagram

The following figure shows the basic block diagram of a RHSIF module with its port structure and internal signal routing.

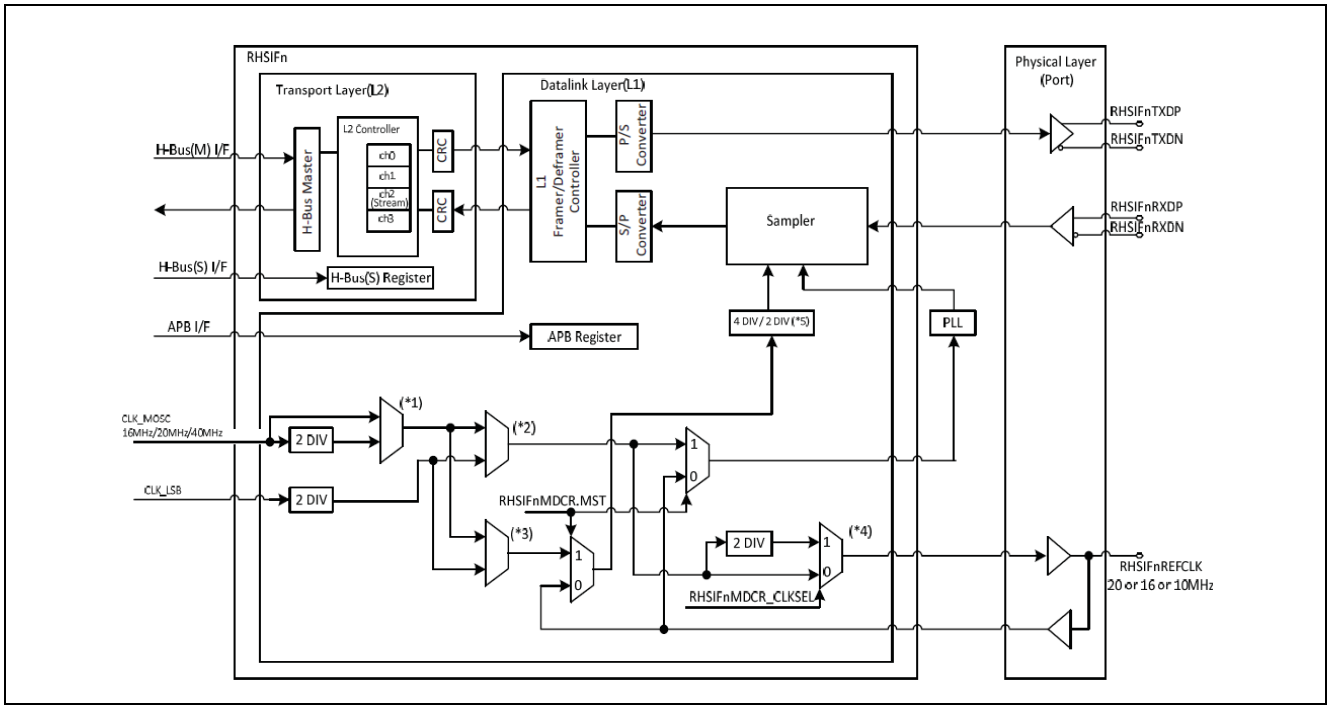

Figure 1: RHSIF Module Block Diagram [1]

5.3 Device Interconnection with RHSIF

The following figure shows the interconnection of two devices which are communicating via the RHSIF.

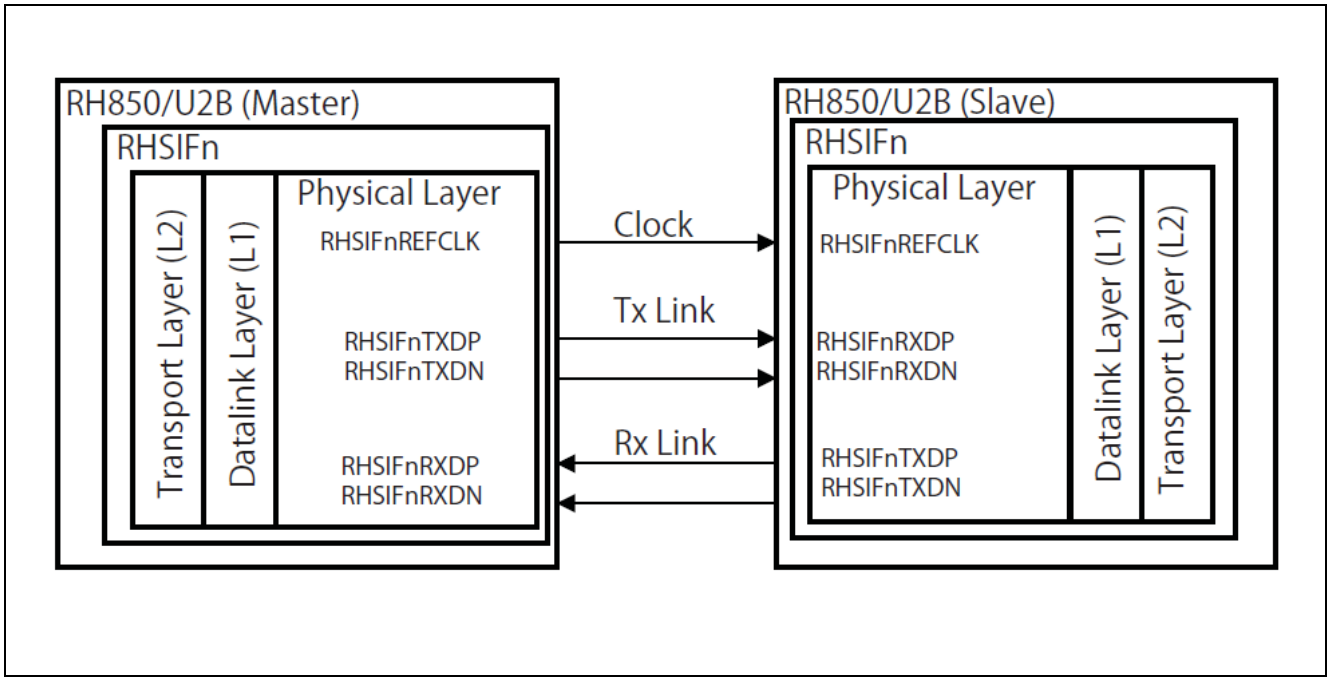

Figure 2: Interconnection between RHSIF Devices [1]

5.4 Message Frame Format

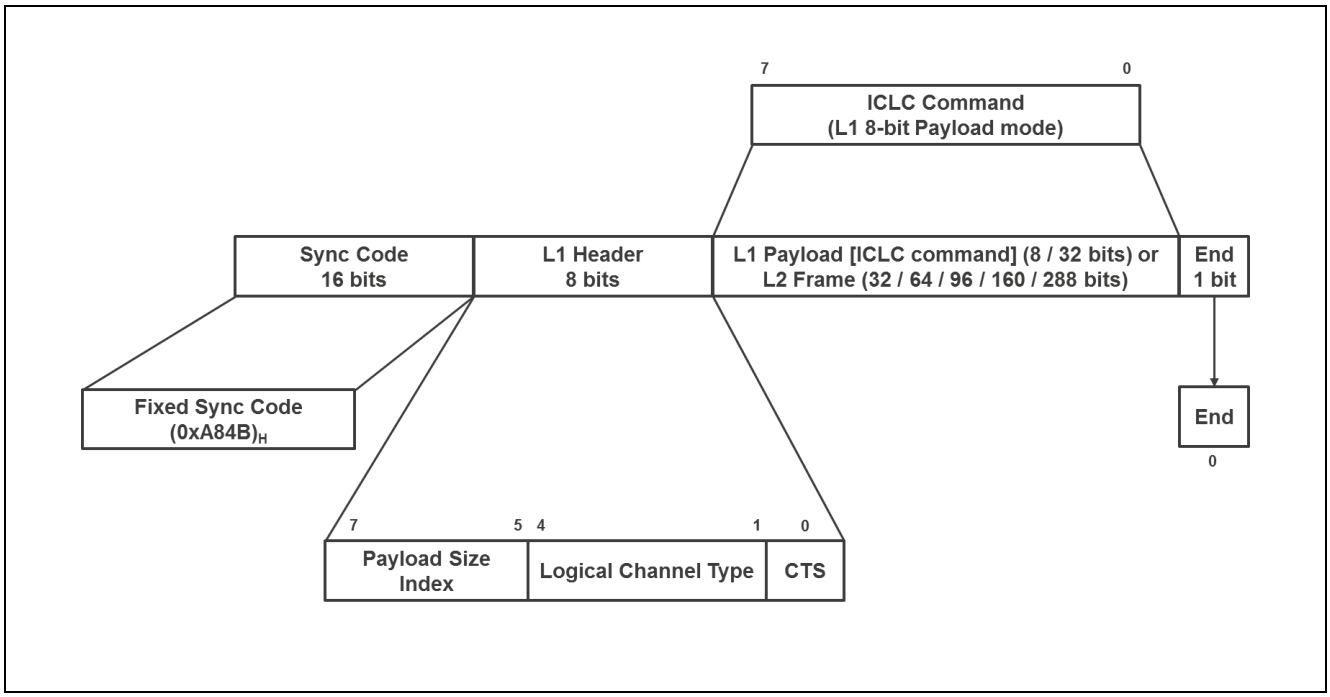

Figure 3: RHSIF L1 Frame detail [2]

The RHSIF module consist of the two sub modules. The "**Transport Layer L2**" and the "**Datalink Layer L1**". The final transmitted L1 Frame can also carry the information of the L2 module. The Datalink Layer performs the integration of L2 data into its final message frame by replacing the L1 Payload with the L2 message frame. The detailed composition of the final message frame is shown in the previous figure.

The L2 frame format consist of the Header and CRC information. The size of the Payload can be 0, 32, 64, 128 or 256 bits. Within the Header the Transaction ID, Command ID and the Channel No. is transmitted. The detailed composition of the L2 frame is shown in the next figure.

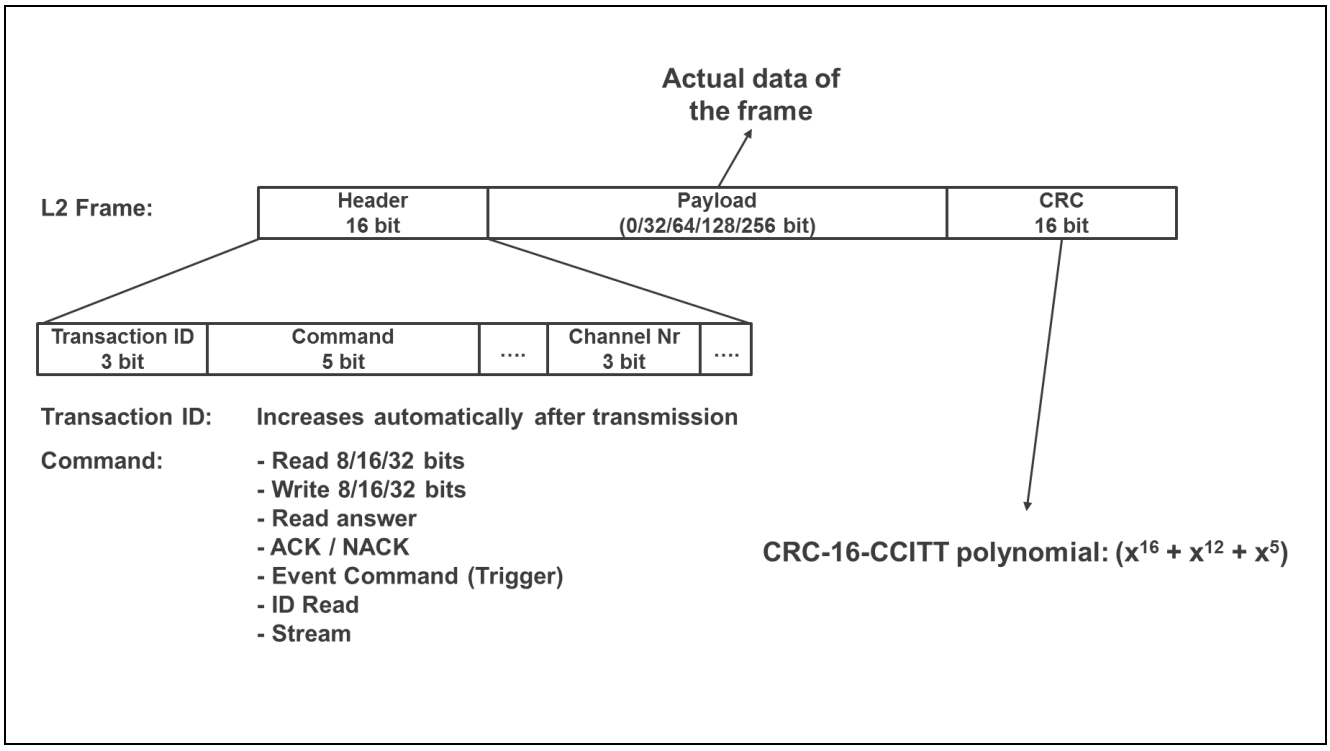

Figure 4: RHSIF L2 Frame detail [2]

The next two chapters provide a general overview of both sub modules to give the user a basic understanding for their interaction. For a more detailed explanation please refer to the RHSIF chapter in the latest user manual which fits to the RH850 MCU device that will be used in the application.

5.5 Datalink Layer L1

The Datalink Layer couples the RHSIF macro to the physical interface and integrates the L1 frame information into the final transmitted L1 message frame. The following functions are implemented by this sub module.

- Generation and reception of ICLC commands (e.g. READ,WRITE or PING)
- Generation and transmission of CTS Commands (“Clear To Send” command)
- Interrupt generation
- Frame arbitration
- Flow control
- Operation modes for:
  - Loopback testing
  - Pattern Testing

The following figure shows the block diagram of the L1 sub module.

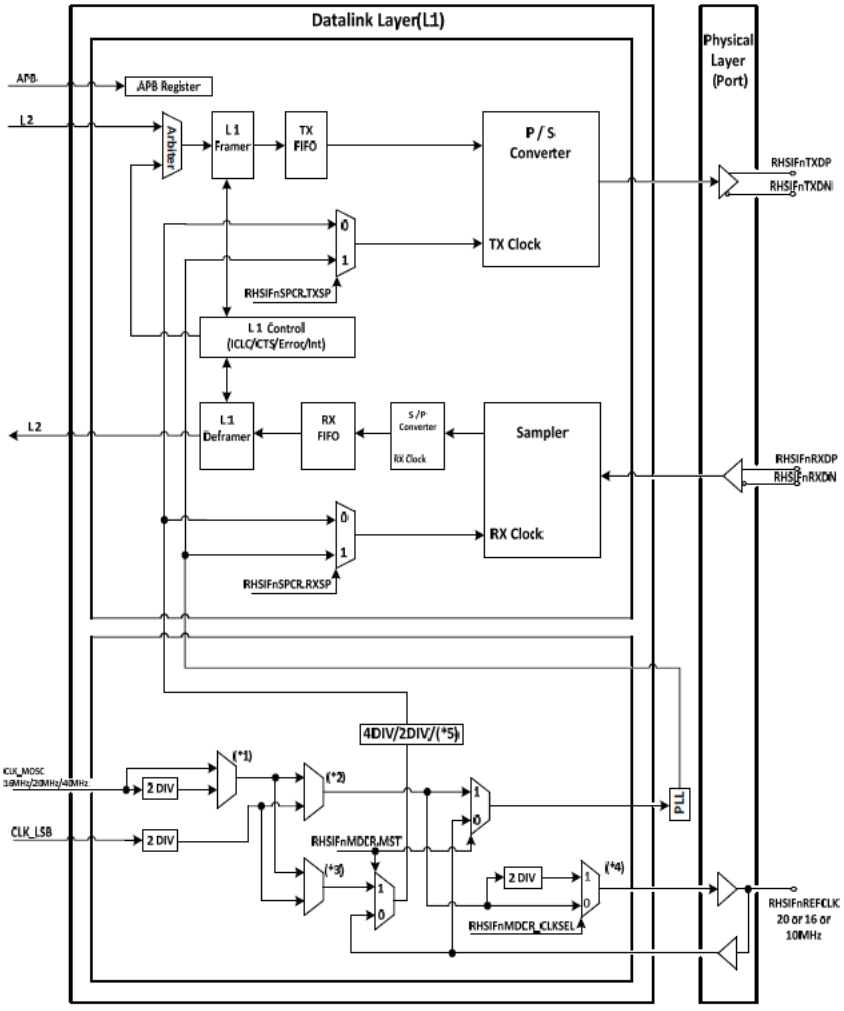

Figure 5: Datalink Layer Block Diagram [1]

5.6 Transport Layer L2

The Transport Layer L2 performs the segmentation of the data into the L2 message frame, which will be send via the RHSIF interface within the L1 frame. Beside the data preparation the L2 module also controls the process communication between the RHSIF modules of the initiator and target node.

The Transport Layer offers the following major features:

- Support of 4 RHSIF channels:
  - With stream transfer capability for channel 2
  - Channel arbitration
- Generation of the Header:
  - Manage transaction ID
  - Support of channel number code table I and II
- Generation and validation of CRC-16
- Support of initiator and target node mode with their dedicated features

The following block diagram provides a simplified overview of L2 sub module.

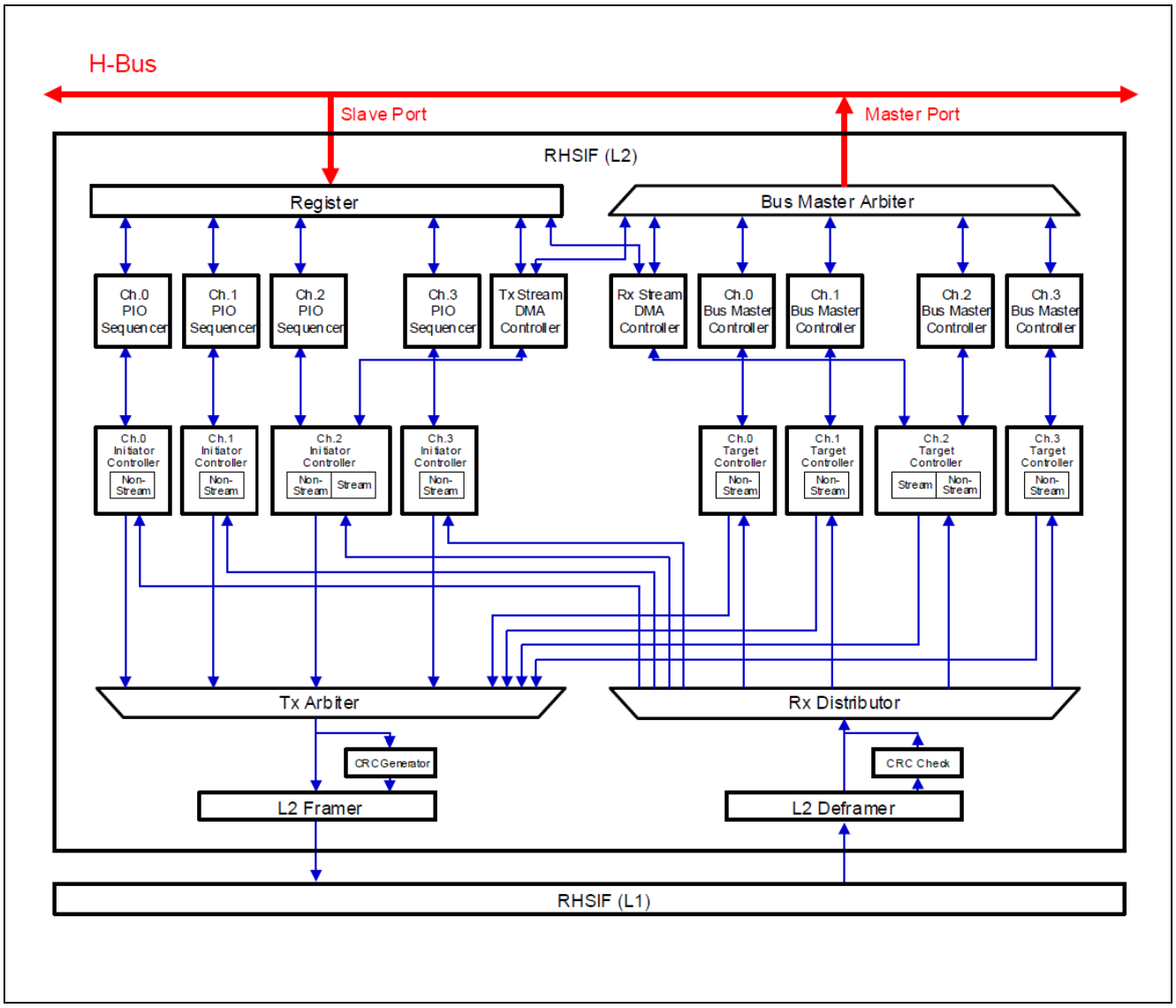

Figure 6: Transport Layer simplified Block Diagram

5.7 Security and Safety Features

The RHSIF interface supports the following Security and Safety features, which can be activated via configuration setting in the RHSIF configuration registers or via Option Byte settings.

- CRC-16 protected transfers
- Programmable timeouts
- Dropped frame detection with automatic transfer ID generation
- Access protection from an external master with window settings
  - Four access windows are configurable
  - Each window can have its own access rule (R/W, W, R)
- Two authentication methods at the initial time of communication:
  - ID based authentication
  - Challenge and response authentication in cooperation with the ICUM
- The RHSIF module is implemented in the overall protection guard security system of the MCU
  - Guard protected access to MCUs memory or registers is possible if necessary
- Build-In self-test functions

This application note implements only the "**ID based authentication**" process in the software example. To provide a basic understanding for the procedure the following chapter will explain the detailed processes for a successful authentication. It is possible to enhance the security of the authentication with a second procedure called "**Challenge & Response Authentication**". The configuration and activation of the involved **guard systems** is also implemented in the software example as further security options.

5.7.1 ID-Based Authentication

The fist authentication is carried out via a 256-bit key ID which is stored in the RHSIF Authentication ID Registers (RHSFInAID0...7). Data can only be read or written to those registers via the master port of the L2 sub module (that means only from the externally connected partner device). When another H-Bus master device (that means another macro or CPU of the device) access those registers the bits can't be modified and are read only for this device.

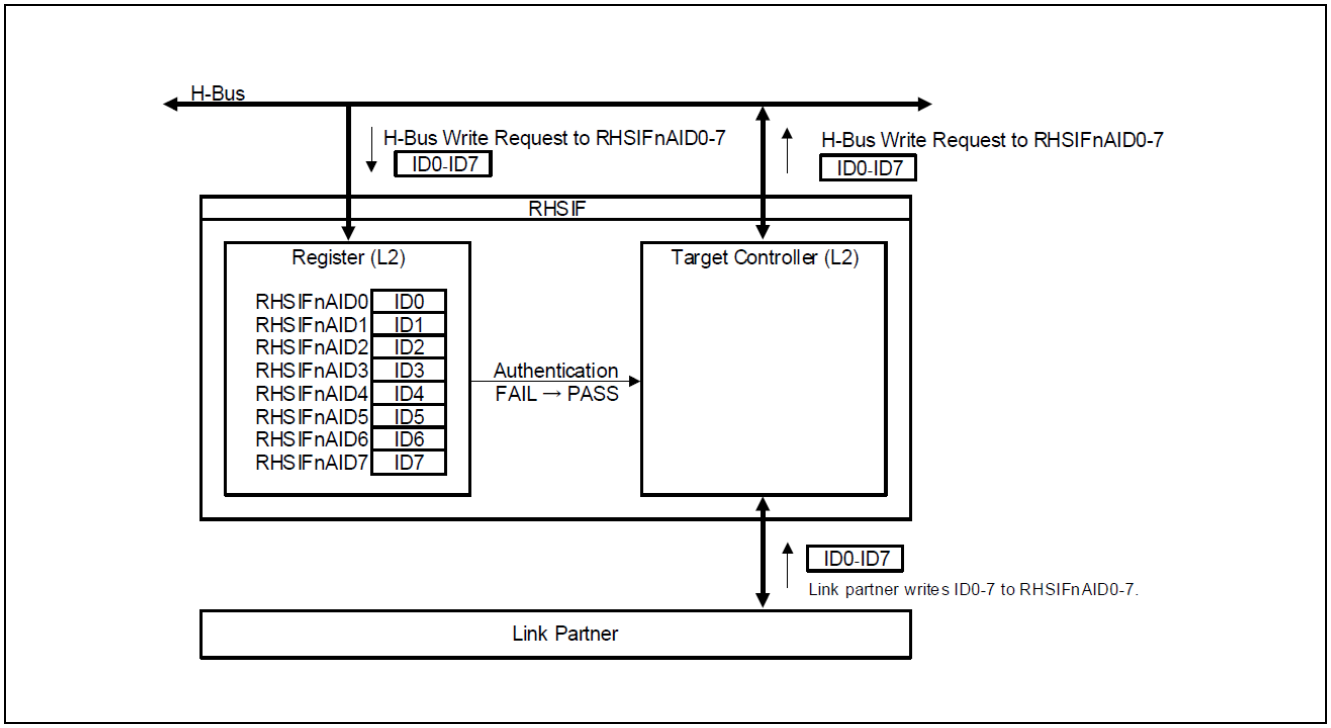

Figure 7: ID-Based Authentication Process -Overview- [2]

When a connected RHSIF partner device wants to perform the initial communication, it must issue a write command addressed to the RHSIFnAID0…7 register in the targeted L2 sub module. The data which is sent with the command is the 256-bit key ID. Therefore the partner device must know the correct ID the partner device expects.

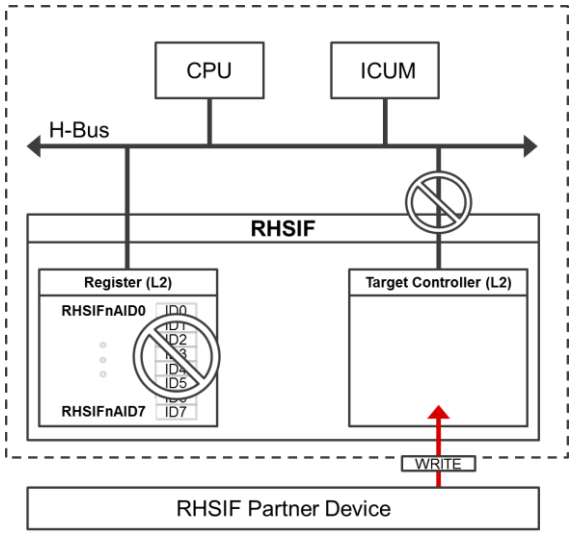

Figure 8: ID-Based Authentication Process Step -1- [2]

The Target Controller of the passive RHSIF L2 sub module receive the commands with the key ID segments and performs an address check on the received frames. When the address matches with the address of the registers of L2 which holds the key ID information the Target Controller of the passive L2 sub module will grant access to the H-Bus of the device.

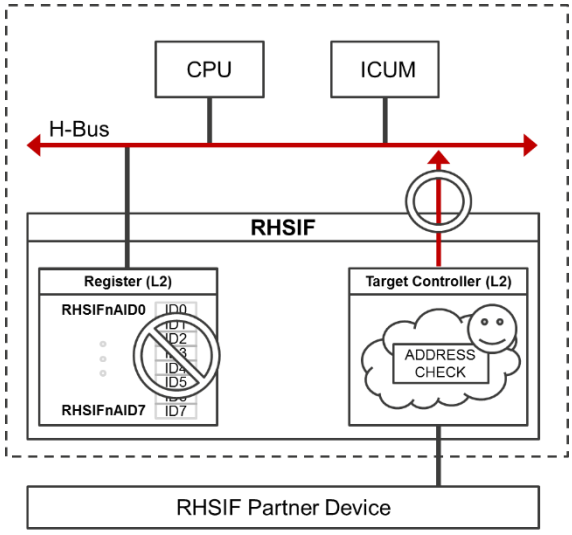

Figure 9: ID-Based Authentication Process Step -2- [2]

When the RHSIF module receive the H-Bus write request to its own registers the L2 sub module checks the Master ID in the write request. Only when the Master ID is identical to the ID of the L2 master port the write operation will be permitted.

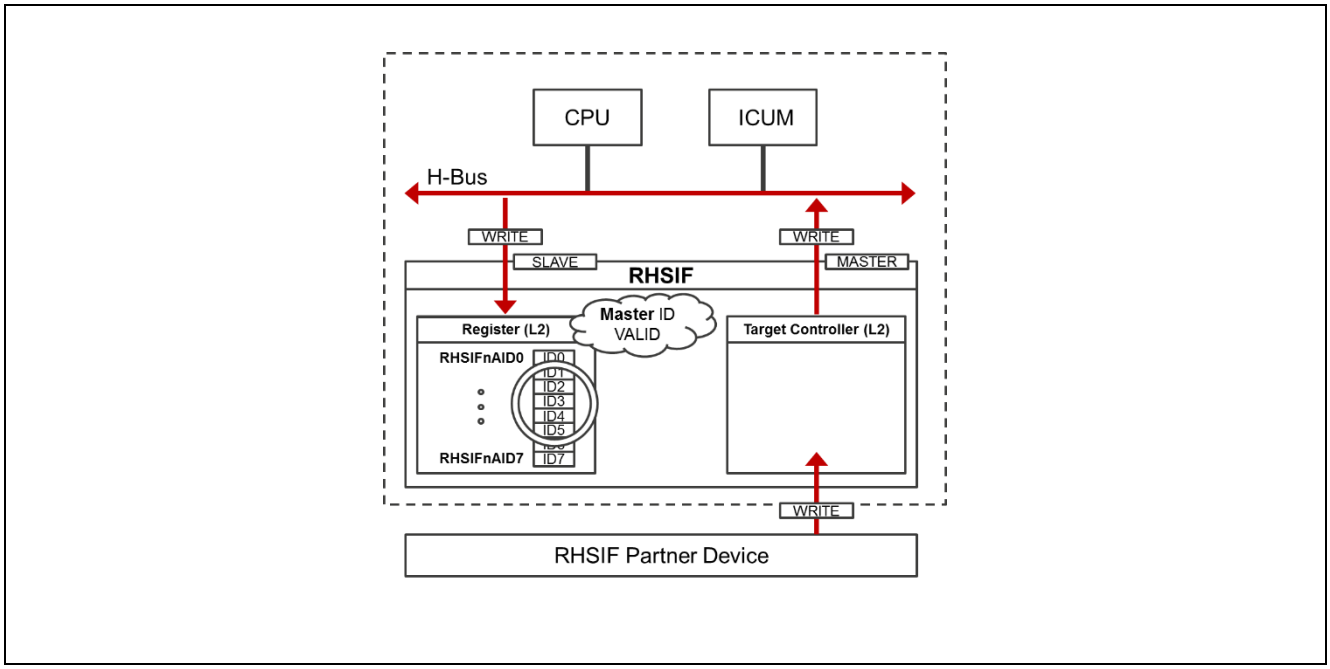

Figure 10: ID-Based Authentication Process Step -3- [2]

In the last step the L2 sub module of the targeted RHSIF module will compare the written key ID with the programmed ID inside the **Option Bytes**. When the written ID is identical to the stored ID inside the flash memory, the RHSIF partner device passed the 1st authentication.

Two options are possible now:

- Further communication is possible without any further authentication checks.
- The **Challenge & Response** authentication can be carried out to enhance the security level.

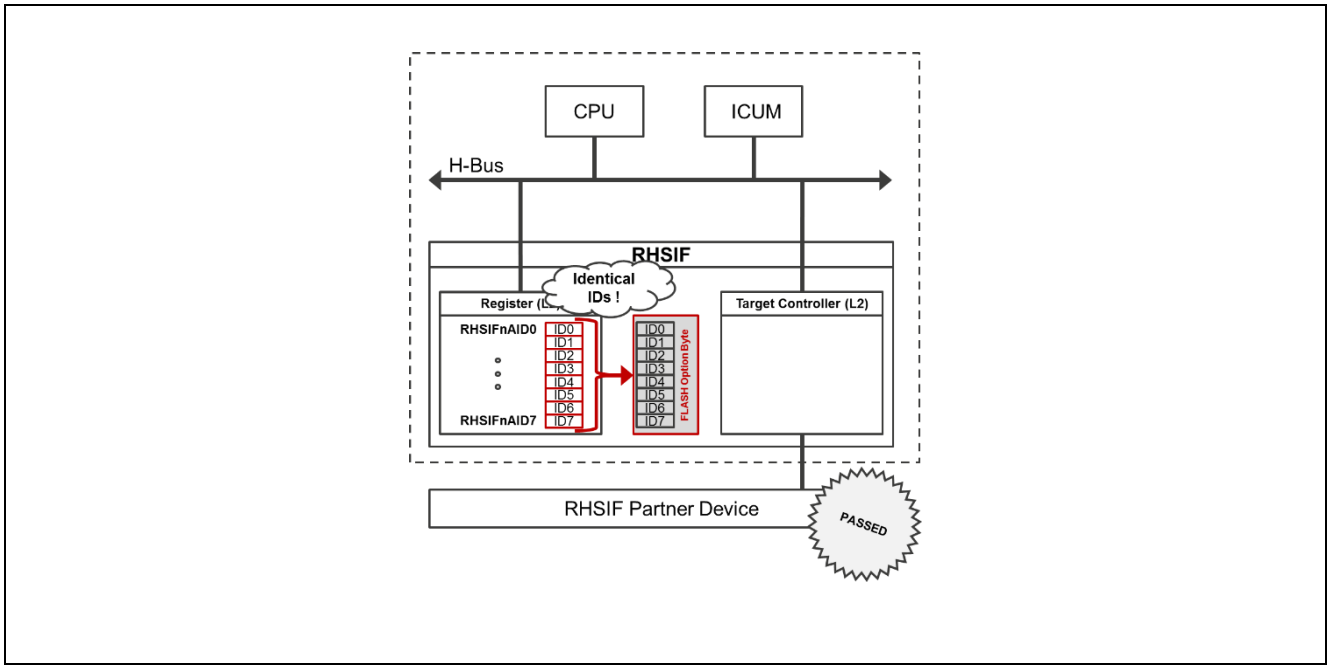

Figure 11: ID-Based Authentication Process Step -4- [2]

5.7.2 Challenge & Response Authentication

After the first ID based authentication is successfully completed, it is possible to perform a second level of authentication. This authentication follows the concept of a challenge & response process by involving the ICUM of the both devices. The challenge & response authentication is not implemented in the attached software example. The provided explanations should only give a basic overview of the RHSIF security features and enhance the understanding for the module interconnection with the other MCU modules and busses.

To start this authentication, the device (active) which wants to access the other (passive) device, prepares a set of challenge data whit its ICUM. The ICUM sends the prepared data to the RHSIF IP controller. The controller finally transmit the prepared challenge data to the link partners RHSIF controller.

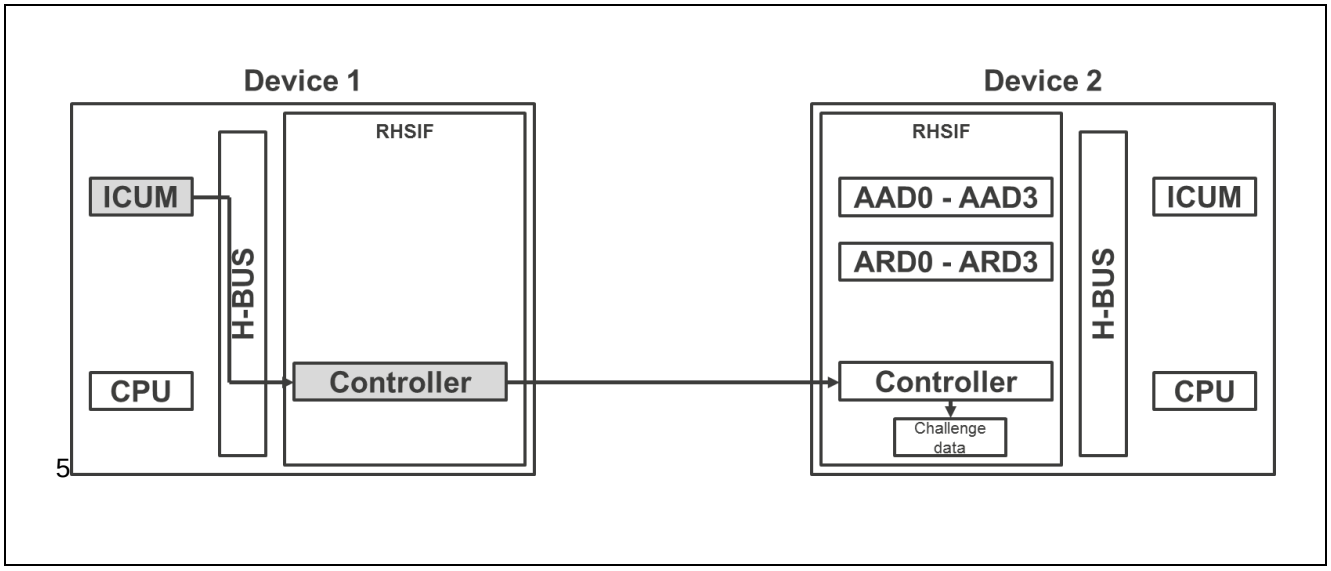

Figure 12: Challenge & Response Authentication Step -1- [2]

After receiving the data the passive RHSIF controller activates the ICUM located on the targeted device. The received challenge data is hand over to the ICUM via the internal H-Bus for further processing.

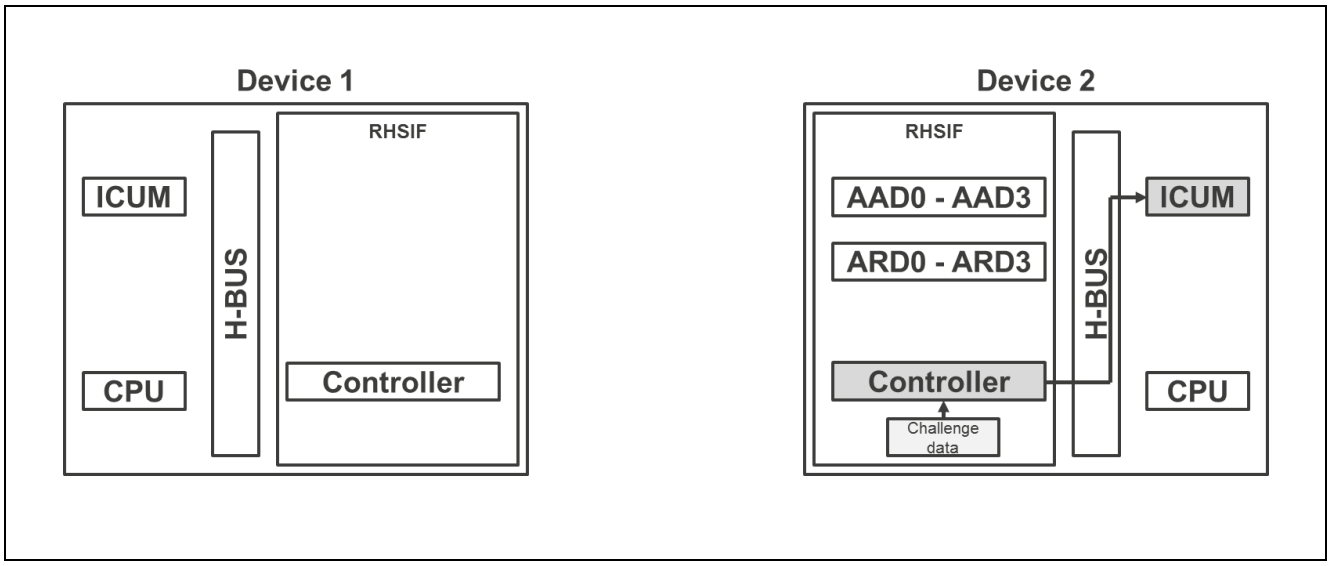

Figure 13: Challenge & Response Authentication Step -2- [2]

The ICUM of the passive device processes the challenge data and write the response data to the AAD0…3 registers of the RHSIF module. Only the build in ICUM can write to those registers.

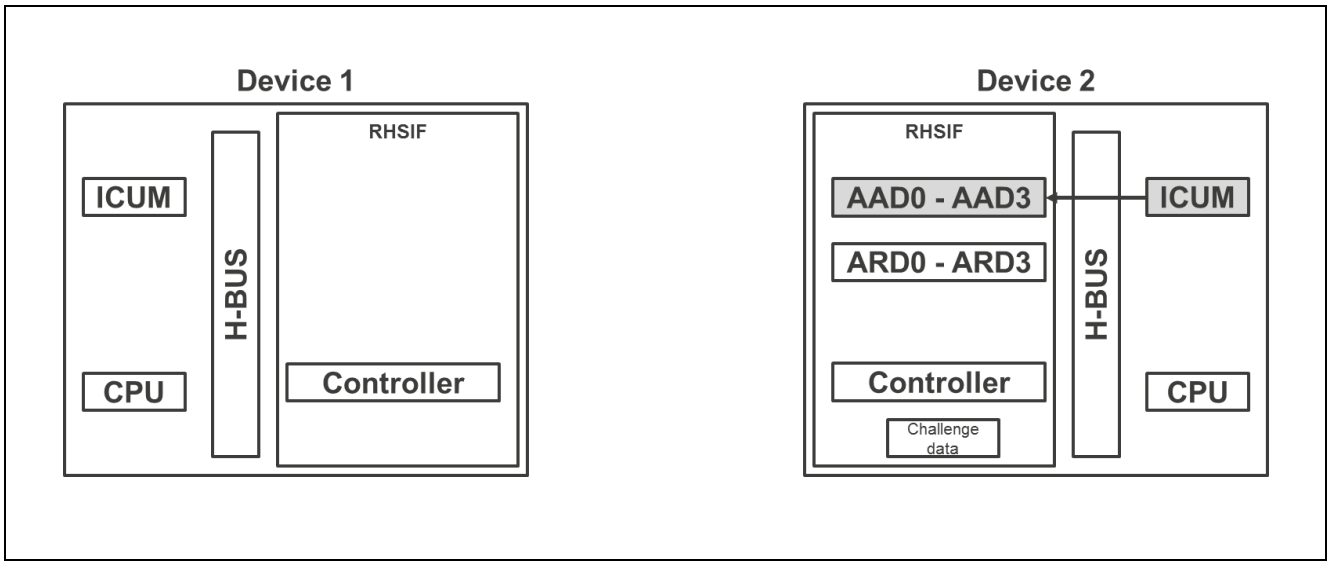

Figure 14: Challenge & Response Authentication Step -3- [2]

The ICUM of the active device calculates his own set of response data in the meantime. After the calculation is finished, the data will be sent to the passive RHSIF controller which will write the received response data to the ARD0…3 registers via the internal H-Bus. Only the device own RHSIF module can write to those registers via the RHSIF master port of the H-Bus.

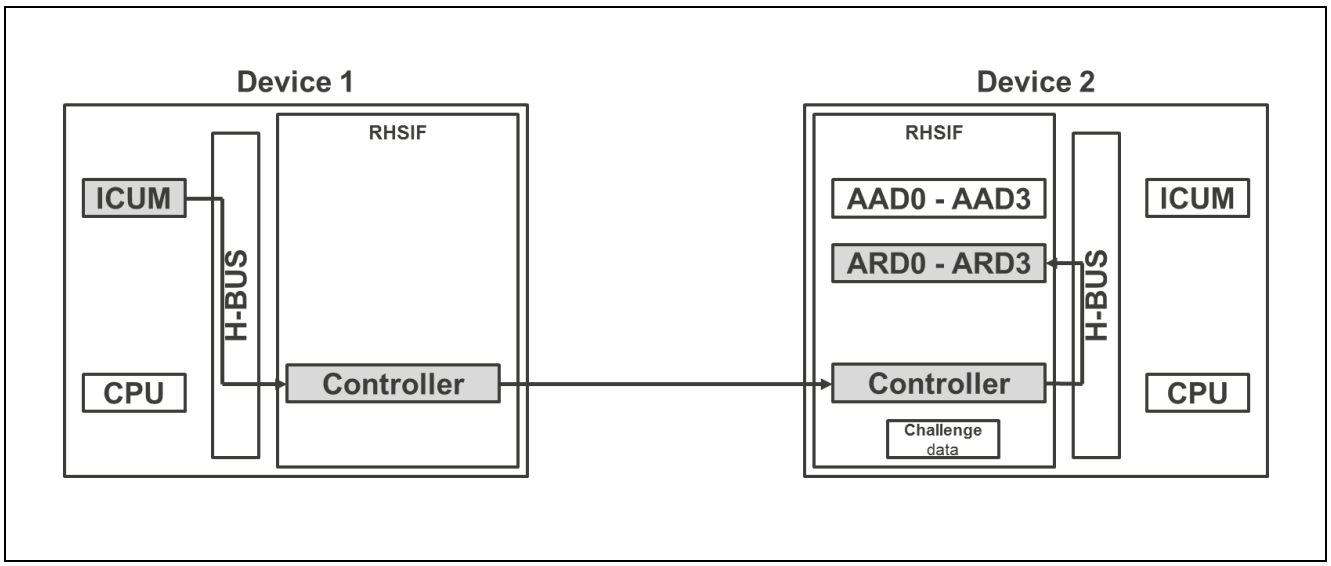

Figure 15: Challenge & Response Authentication Step -4- [2]

In the last step the passive RHSIF module compares the both sets of response data located in the AADn and ARDn registers. When the data matches each other, the passive RHSIF module will grant the access to the H-Bus of the link partner device own RHSIF module.

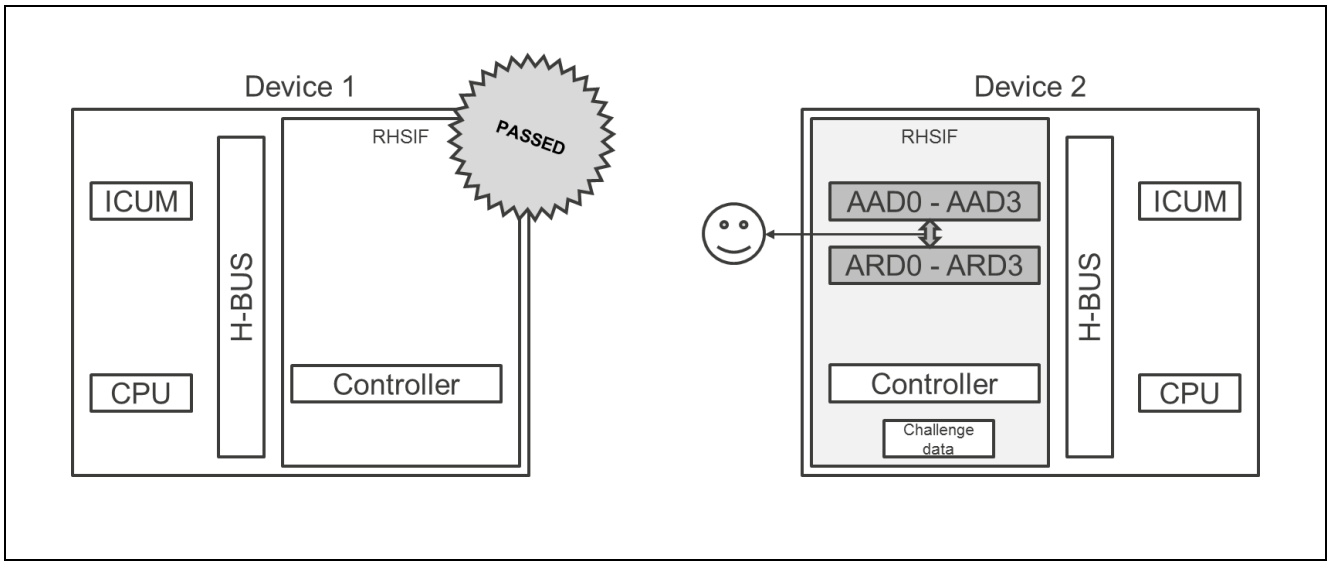

Figure 16: Challenge & Response Authentication Step -5- [2]

5.7.3 Memory Address Windows

When the RHSIF operates as a target node, it is possible to access the memory of the targeted node by the initiators RHSIF interface. To avoid security issues when an unspecific link partner access the memory of the partner device RHSIF implements the concept of the **Memory Address Windows**.

With these windows the memory map of the targeted node can be separated into up to four different zones which are independent in their size and access behavior. When the initiator node wants to access a memory address which is not windowed for him, the targeted node denies the access attempt and generates an error which will be indicated in the RHSFInAEST register.

When you set a memory address window you have to declare its start address in the memory map and the size of the window. The windows should not overlap each other. The setting of the windows is possible during the initial setting mode of the RHSIF interface, or after disabling all channels in the target- and initiator-node function.

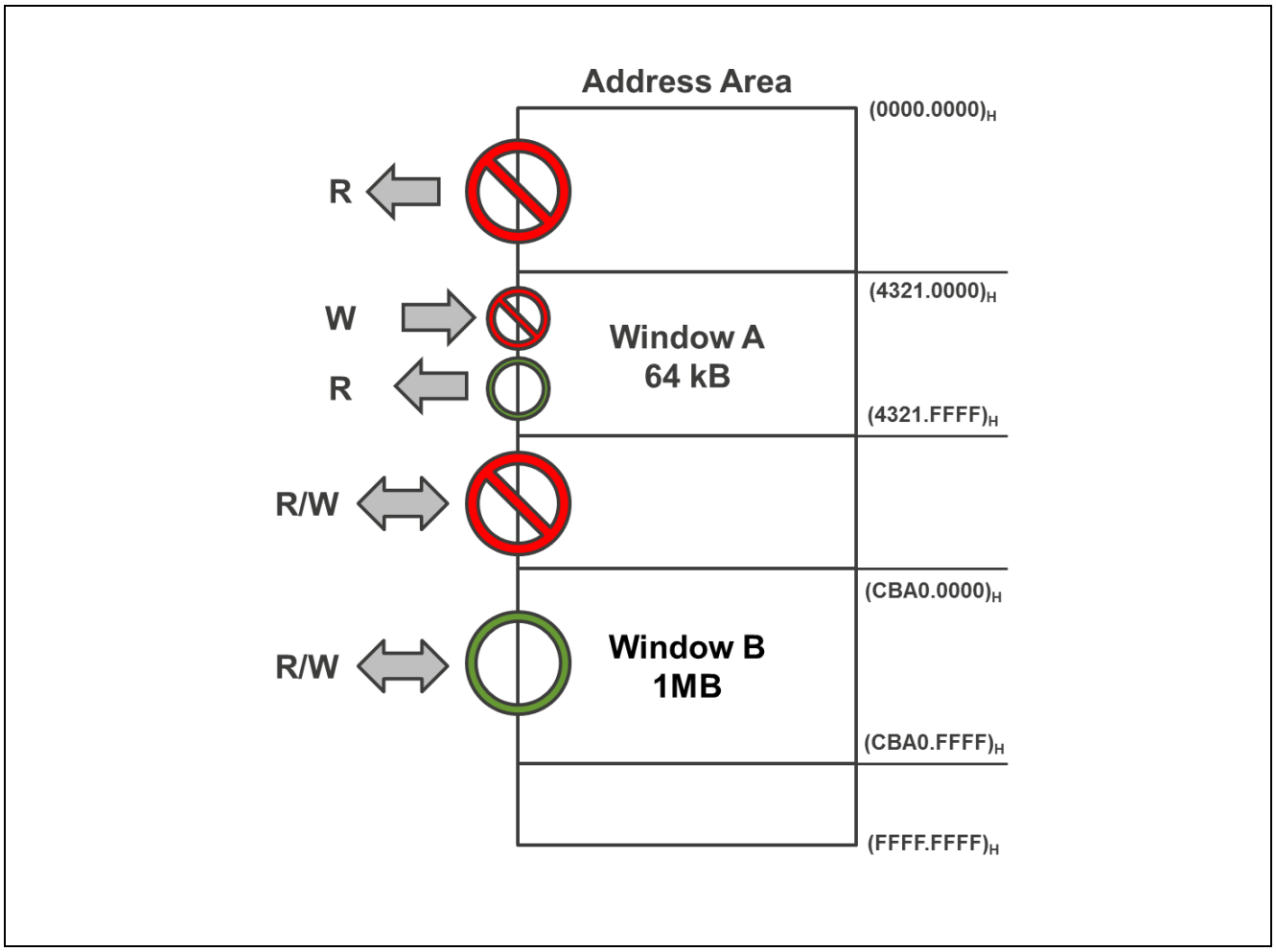

Figure 17: Memory Address Windows [2]

5.7.4 Protection Guards

An access to the internal memory by the high-speed peripherals like RHSIF is protected by tree different guard systems which are important for this application note.

- H-Bus Guard (HBG)
- Cluster RAM Guard (CRG)
- Peripheral Bus Guard (PBG)

Please keep in mind, that there usually are other guard systems like PEG (Processor Element Guard) which protects the CPUs local RAM from unhallowed access. When you are adapting this software to your dedicated use case you maybe have to configure other guard systems as well.

H-Bus Guard

The H-Bus interconnects the high-speed peripheral IPs and have a slave guard system (HBG) that controls which core can access the H-Bus. The HBG controls the read and write access. The region of the protection is a peripheral instance. Each region has its own registers for protection configuration. These registers are protected with LOCK-Bits against illegal changes.

The HBG uses two filters in sequence to check if a core is allowed to access the H-Bus. The first one checks the PEID and the second one compares the SPID with the SPID filter. When an illegal access is detected the violation will be noticed to the ECM module.

Cluster RAM Guard

This guard systems controls the access to the Cluster RAM. The Cluster RAM Guard can separate the Cluster RAM into eight different protection zones with independent access right behavior. A Bus master can only access one of the declared cluster RAM zones, if the access permission is granted by the Cluster RAM Guard. When an illegal access attempt is detected, the ECM will be noticed.

Peripheral Bus Guard

The Peripheral Bus Guard offer several **PBG Groups** to the user. Each of these groups have 16 protection channels. A Single PBG channel can control the access to a single peripheral circuit. If an access attempt is rejected by the PBG the information related to this access is stored into the error registers of the involved PBG group.

6 RHSIF Interface Example - Hardware Setup

The following components where used to implement the RHSIF application on both MCU devices.

Software:

- Green Hills Software MULTI® ver. 7.1.6 / Compiler V2019.5.5
- Renesas Flash Programmer (use the latest version)
- Latest “EXEC” and “ESERV” files which are necessary for the GHS Compiler to operate with the RH8520/U2Bx

Debugging Tools:

- 2x Renesas E2 Emulator
- 2x PiggyBack board V1 RH850-U2Bx Evaluation Platform (486 pin version)

Hardware components:

- Microcontroller:
  — 2x RH850/U2B24-FCC 486pin R7F702Z23 (WS 1.0)
- Cable to interconnect the two PiggyBack boards RHSIF interface connectors

Note: For best readability in the source code editor a TAB spacing of 4 characters should be applied.

The two microcontrollers are applied to the PiggyBack board device sockets. The used version of the PiggyBack boards are offering the RHSIF interface via a separate connector. These two connectors are combined with a suitable cable.

As an alternative it is also possible to connect the two devices via the pin headers, but be aware of possible EMC issues when using higher Baud rates due to this insufficient cabling method. If you want to use this method please make sure that the dedicated pins are connected to the MCU device ports on the PiggyBack board. In some cases the ports are not connected to the pin header on the board by factory preset, so that the user has to establish the routing manually by soldering the necessary connections.

The used E2 Emulator offers the possibility to provide the necessary supply voltage to the PiggyBack board. Depending on the implemented use case and your additional soft- and hardware application for the device, it might be necessary to apply an external voltage supply. The E2 Emulator is limited to its power output by around 200mA.

The GHS multi environment is capable to perform the simultaneous independent debugging for two connected E2 Emulators on one single Host system. To implement this you have to add the following command to your connection string in the “Connection Organizer” of the GHS MULTI. Please add the individual serial number of the used debugger before adding the command.

- -ice E2:_<SERIAL No. OF E2>

The following figure shows the hardware setup with the PiggyBack boards in detail.

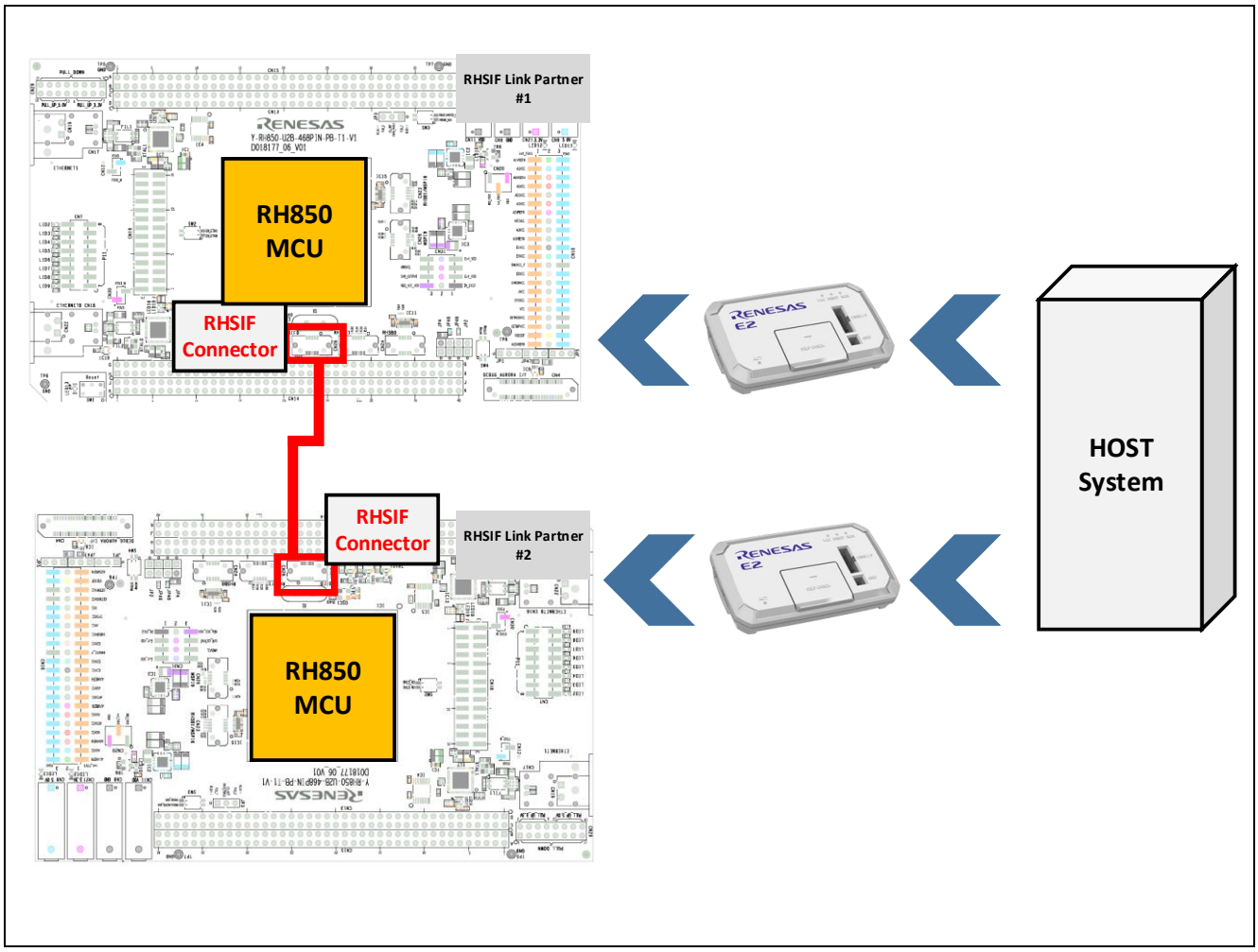

Figure 18: PiggyBack Board Hardware Interconnection

7 RHSIF Interface Example - Software

The attached software example is implemented on a RH850/U2Bx MCU device. When you are going to use other Renesas MCU devices which are equipped with the RHSIF IP you have to adapt the software. The basic steps for configuration and operation are identical but e.g. the register names might be different.

For the operation of the interface the following modules of the device must be configured before an RHSIF interconnection is possible.

- The RHSIF interface
- The RHSIF related ports of the RH850/U2Bx
- The memory access windows and the involved bus guards

Details for each configuration are shown in the next chapters. In order to make the described source code sections more recognizable for the user, the start of the single sections are marked with the related numbers of the description in this application note. The used number formatting in the source code is highlighted with a yellow marking for the following short example:

```
/*#1*/
  if ( SYSCTRL.MSR_RHSIF.BIT.MSR_RHSIF_0 == 1 )
  {
    SYSCTRL.MSRKCPROT.UINT32 = 0xa5a5a501; // Unlock the MSTBMS_RHSIF0 register
    SYSCTRL.MSR_RHSIF.UINT32  = 0;         // Put the RHSIF to operating mode
    SYSCTRL.MSRKCPROT.UINT32 = 0xa5a5a500; // Lock the MSTBMS_RHSIF0 register
    }
```

7.1 Security Option Byte Settings for Authentication

On the RH850/U2Bx the Key-ID based authentication must be activated by the user as shown in this chapter. The user can check the status of a selected activation with the Security Option Byte **S_OPBT_6**. The default value for the register is as follows:

- S_OPBT6 = $(FFFE\ FFFF)_H$

The bit 16 (S_OPBT6.HSIF_IDAUTH_NEED) defines if a link partner authentication is needed or not.

When the ICUM is activated on the MCU device, the Key-ID based 1st level authentication is upgraded with the 2nd level Challenge & Response authentication procedure as shown in the previous chapter. The descried software example only implements the Key-ID based authentication because the ICUM is not used.

To perform the change of the security option byte you have to use the Renesas Flash Programmer (RFP). With the current implementation of the RFP it is required to load the project file into the programmer tool. You have to read out the memory settings by clicking on the “Device Information” button and then “Read Memory Information”. Select only the **SECURITY** area in the selection and store the **.mot** file to your local drive.

In order to modify the necessary settings you need to change the content of the file with e.g. an “S-Record” modification program. Optional you can change the settings by using a usual editor, but then you need to calculate the correct checksums manually by yourself.

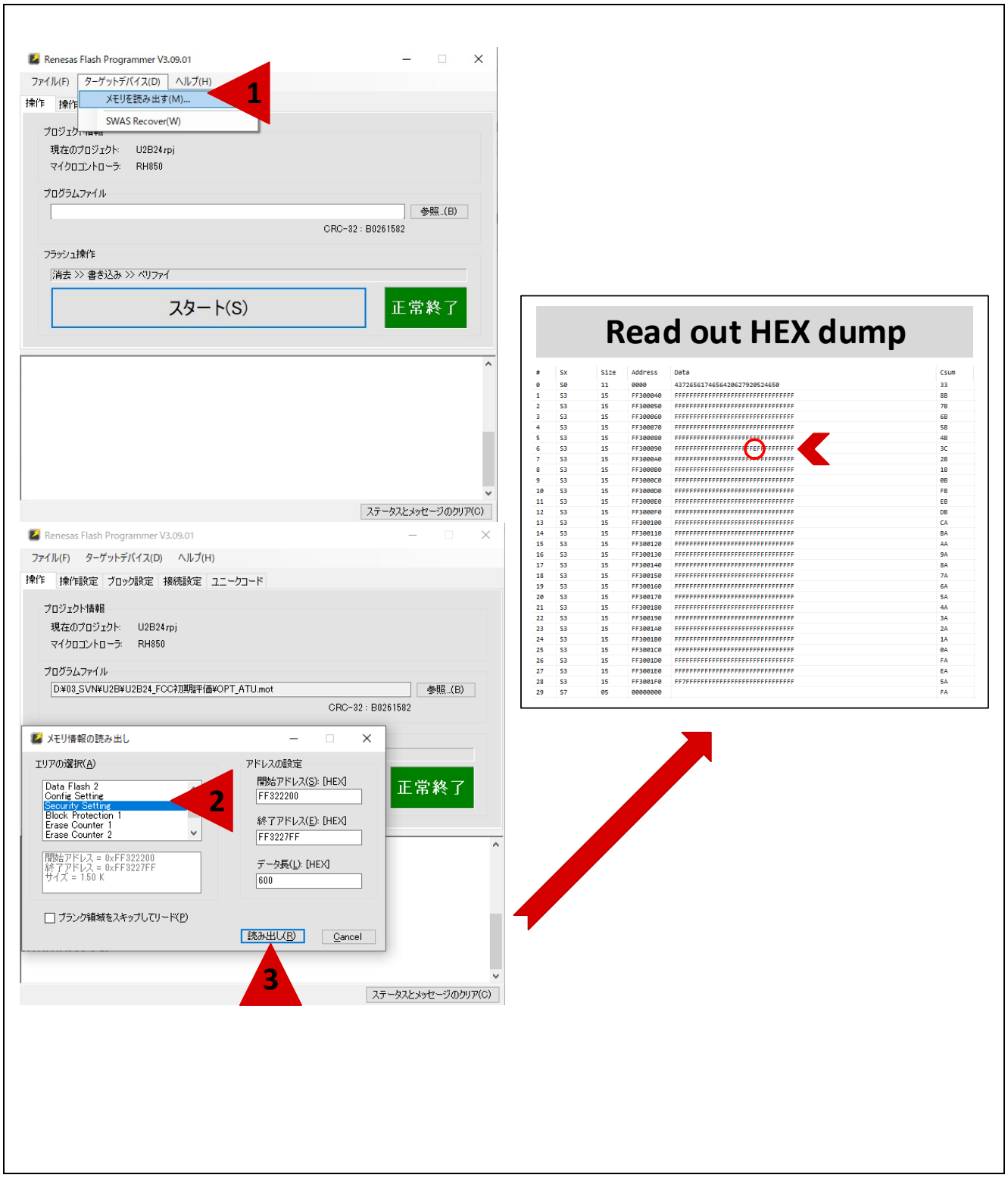

Figure 19: Memory Read with the RFP Tool

Finally you need to load the modified **.mot** file into the Microcontroller. This is carried out also with the RFP by hitting the “Browse” button and select the file you have modified in the previous step. Be sure that you have add the checkmarks on the “Operation Setting” section of the RFP for **ERASE**, **PROGRAM** and **VERIFY**. You start the programming process by hitting the START button on the “Operation” section.

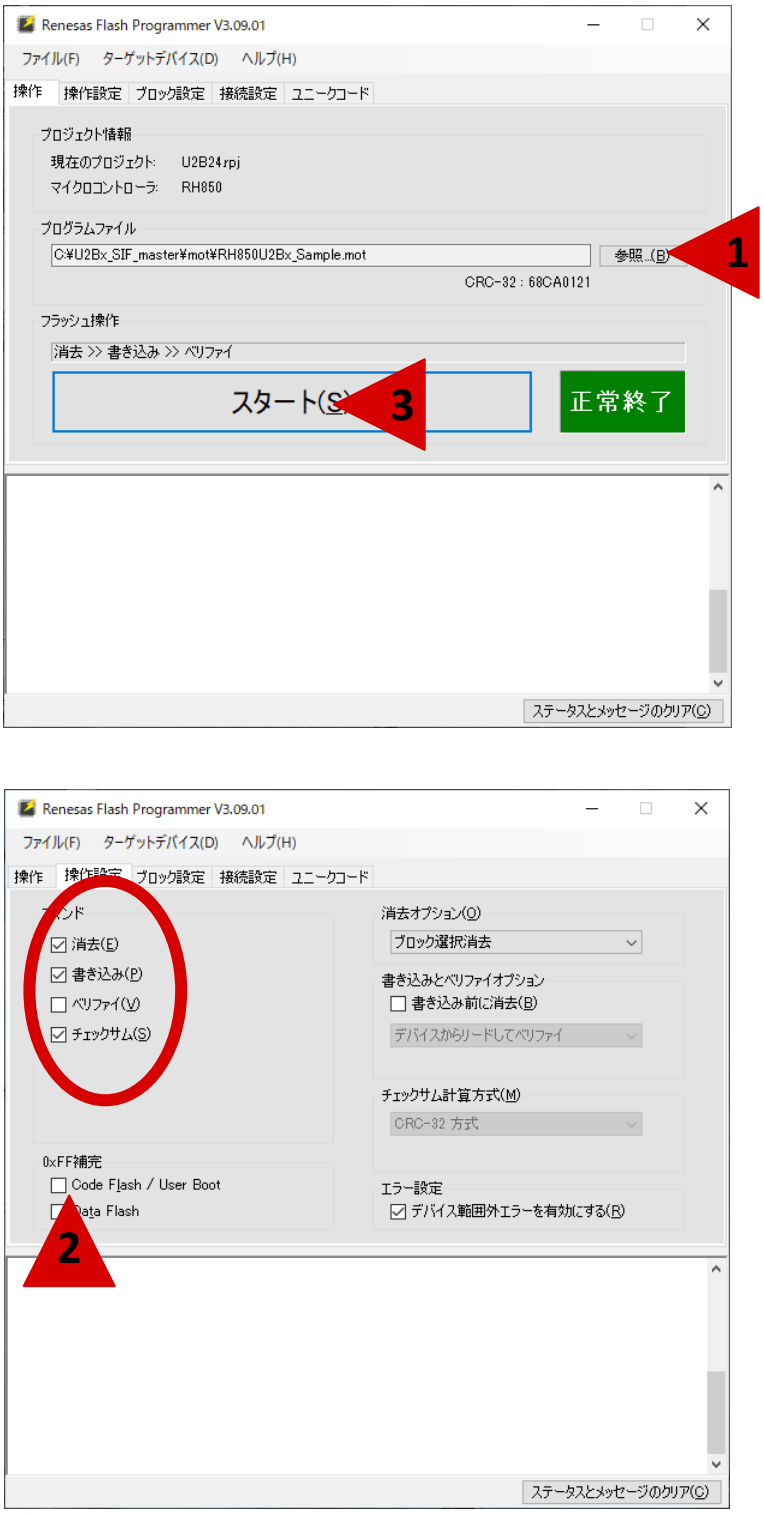

Figure 20: Memory Write with the RFP Tool

7.2 Structure of the Demonstration Software

The RHSIF demo project was realized on a dual core RH850/U2Bx. The important code lines for the RHSIF operation are located in the following source code and header files:

- RHSIF.h
- main_pe0.c
- RHSIF.c

7.2.1 C Header File “RHSIF.h”

This header file defines several type definitions used in the project. The definitions are as follows.

- **RHSIF_State_t**
  - Used to show the current state of the RHSIF driver SW
- **L1_Payload_Size_t**
  - Defines different possible payload sizes for L1
- **L1 Logical_Channel_Type_t**
  - Provides definitions for the four channels, CTS and for the interface control
- **RHSIF_Mode_t**
  - Defines the two modes: MASTER/SLAVE
- **RHSIF_L2Command_t**
  - Defines the different types of L2 commands

Note: Not all available ICLC-, L1- and L2-commands are used in the demonstration software

After all type definitions the used sub functions for the RHSIF.c file are declared with their used variables.

7.2.2 C Code file “RHSIF.c”

This function defines all RHSIF related functions used for this application note. The following functions are implemented

- **RHSIF_Init**
  — for the initialization of the interface
- **RHSIF_L2_Send1stAuth**
  — for the Key-ID based authentication sending process
- **RHISF_L2_wait1stAuth_Check**
  — To check if the authentication is carried out in the right way
- **RHSIF_L1_SendPing**
  — which will issue a PING command to the L2 sub module via ICLC command
- **RHSIF_L2_setWindow**
  — to configure the Memory Access Windows
- **RHSIF_L2_read32**
  — which performs a read operation
- **RHSIF_L2_write32**
  — to prepare a write transmission
- **loc_RHSIF_portInit**
  — to initialize the RHSIF related ports groups and pins
- **RHSIF_L2_stream256_read**
  — which performs a stream read operation
- **RHSIF_L2_stream256_write**
  — to prepare a stream write transmission

7.2.3 C Code file “main_pe0.c“

This is the main function for the processor element 0 (PE0). Within the source code those RHSIF related events are implemented.

- **For the MASTER device:**
  — Function call for RHSIF IP initialization as a Master device
  — Function call 1st level authentication
  — Function call 1st level authentication check
  — Function call for the READ function
  — Function call for the WRITE function
  — Function call for the SEND PING function
  — Function call for the STREAM WRITE function
  — Write data preparation

- **For the SLAVE device:**
  — Function call for RHSIF IP initialization as a Slave device
  — Function call 1st level authentication check
  — Function call 1st level authentication
  — Function call for the READ function
  — Function call for the WRITE function
  — Function call for the STREAM READ function
  — Write data preparation

The following figure shows the RHSIF related flow chart of the “main_pe0.c” source file.

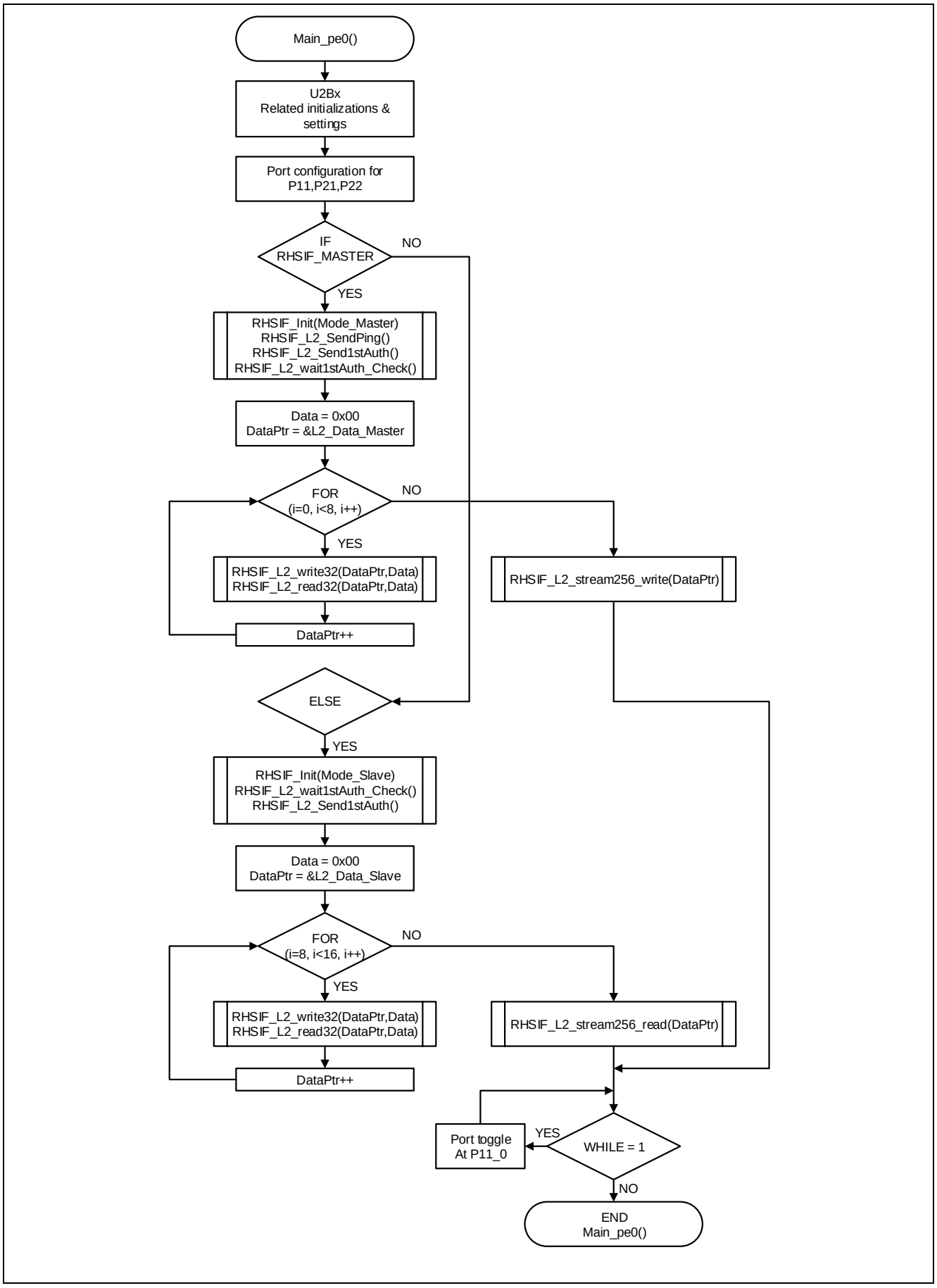

Figure 21: Flowchart “main_pe0.c”

7.3 Software Process Chart

The following figure shows the overall software process and the individual dependencies of the two devices in their combined workflow.

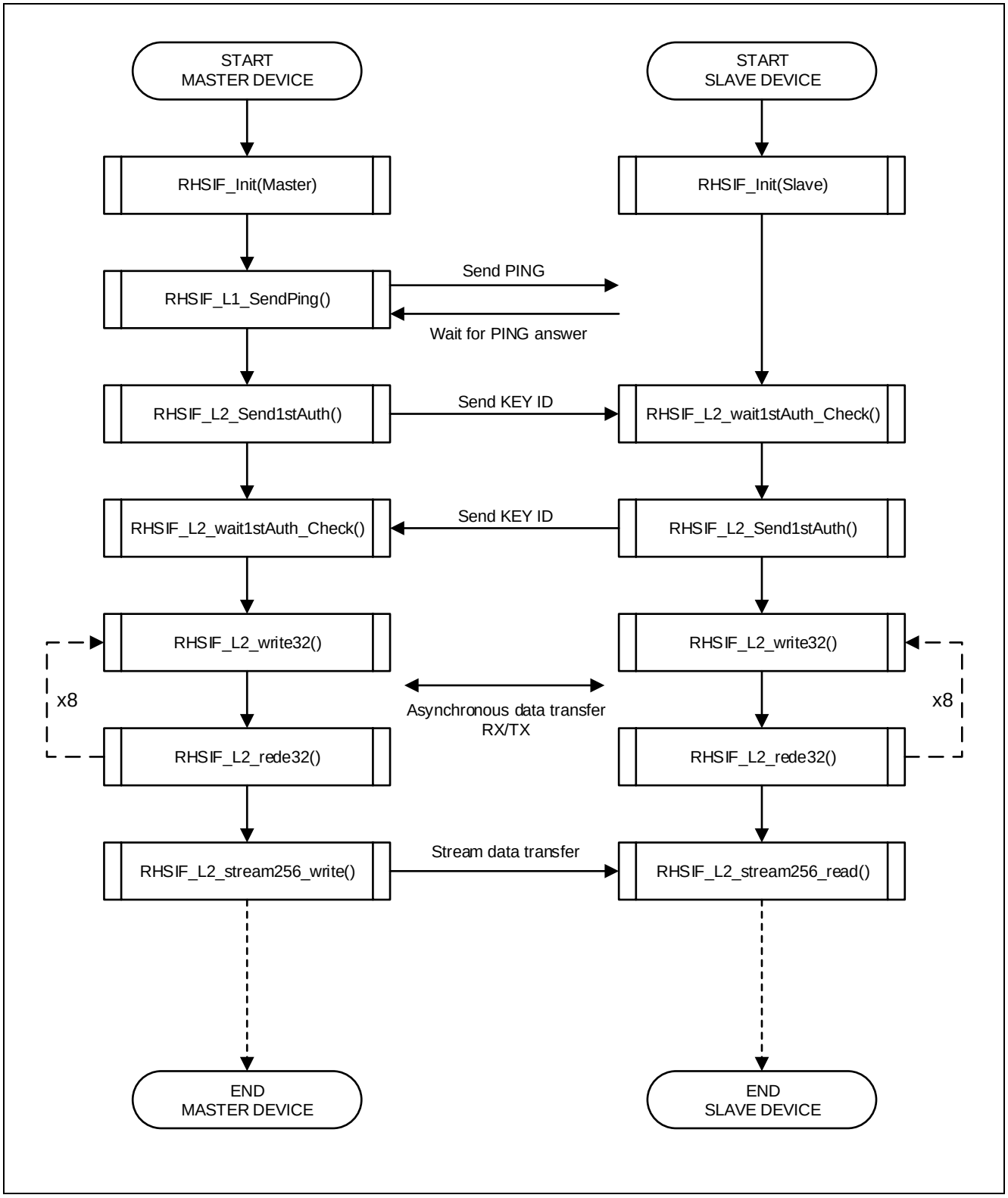

Figure 22: Software Process Chart

7.4 RHSIF Module Initial Configuration

The initial configuration of the RHSIF interface is carried out inside the source code file "**RHSIF.c**". The related function is named with "**RHSIF_Init**".

#1. Release the RHSIF from standby. This is done by clearing the bit MS_RHSIF0 in the register MSR_RHSIF to 0. As the register is protected from inadvertent write access, it must be unlocked via the MSRKCPROT register before the write access.

#2. Disable the PLL of the RHSIF for the configuration of the interface:
Set the RHSIFnPCR.PLLSTBY bit to 1.

#3. Set the RHSIF0MDCR register according to the application requirements:

   a. Configure the device as either the RHSIF master or as the RHSIF slave.
   For this the RHSIF0MDCR.MST bit must be set accordingly.

   b. The CLKSEL configures the REFCLK frequency to the 10MHz or 20MHz

   c. The CTSV bit configures the CTS value of the L1 headers.

   d. The CTSEN configures the automatic flow control

#4. Set the RHSIFnTXRXCR register to enable reception and transmission of the RHSIF.

#5. Set the RHSIFnSPCR register according the application requirements:

   a. The RXSP and TXSP bits set the interface to slow mode or fast mode for reception/transmission.

   b. In case the fast mode is selected the FMBR1 bits the actual baud rate.

#6. Activate the PLL by clearing the RHSIFnPCR.PLLSTBY bit to 0.

#7. Finally configure the RHSIF related port pins

7.4.1 Protection Guards Configuration

The Guard configuration is also located inside the function "**RHSIF_Init**". The Guards are not activated after power on reset as default. If you want to use the Guard system you need to uncomment the commented lines in the demonstration code.

#8. Disable the Cluster RAM Guard according to the setting procedure shown in the user manual.

   a. Unprotect the control registers of the Cluster RAM guard to make a configuration change possible

   b. Make the address range settings

   c. Enable the guard (OPTIONAL)

   d. Protect the control registers of the Cluster RAM guard.

#9. Disable the H-Bus Guard according to the setting procedure in the user manual

   a. Unprotect the H-Bus guard control registers

   b. Enable the H-Bus guard via the control register (OPTIONAL)

   c. Protect the control register by writing the dedicated bit sequence to the protection register

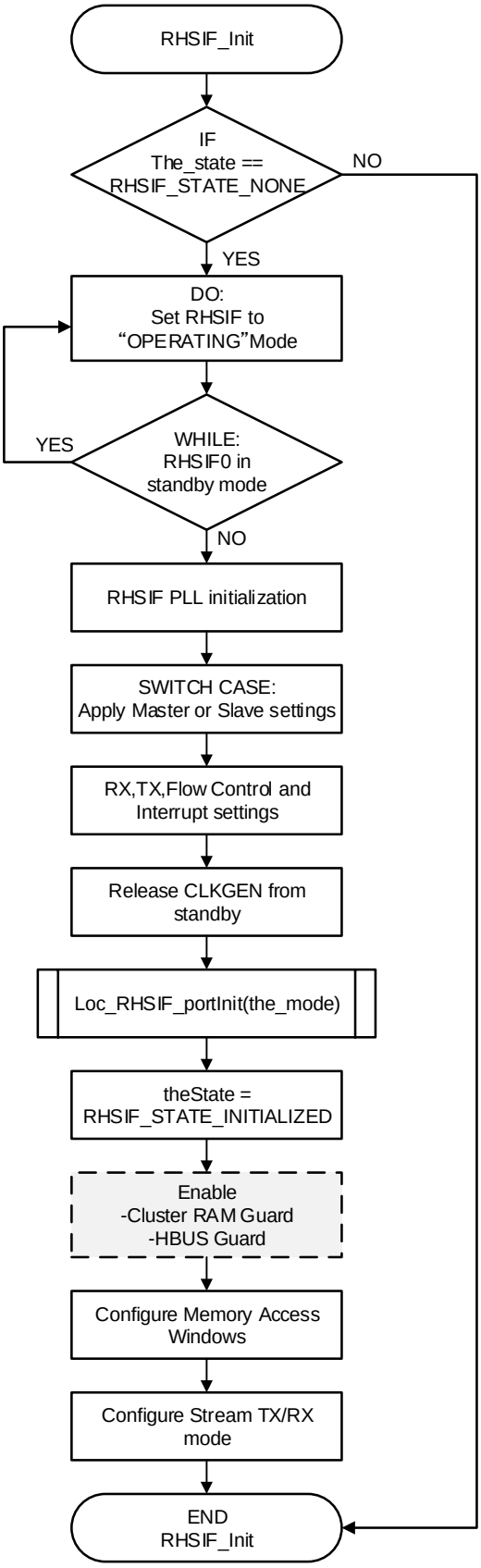

Figure 23: Flowchart “RHSIF_Init” Function

7.4.2 Memory Access Windows Configuration

The configuration of a single memory access window is shown inside the function "**RHSIF_Init**".

#10. Set the RHSIF start address and size for the memory access windows via the "**RHSIF_L2_setWindow**" function.

a. The function calculates the Start, Size and Window No. parameters based on the values the user handover together with the function call inside the "RHSIF_Init" function.

7.4.3 Stream TX/RX Configuration

This is the initial setting for stream transfer.

#11. Configure for stream Tx mode

a. Set the RHSIFnMRT register for Module Reply Timeout.

b. Set bit STPS of the RHSIFnSTMD register for data payload size to the 128bit or 256bit.

c. Set the RHSIFnSTBC register for Stream Tx byte count.

#12. Configure for stream Rx mode

a. Set bit SRPS of the RHSIFnSRMD register for data payload size to the 128bit or 256bit.

b. Set the RHSIFnSRDS register for Stream Rx Destination Area Size.

c. Set the RHSIFnSRBC register for Stream Rx byte count.

7.5 Microcontroller Port Configuration

The configuration of the necessary port settings for the RHSIF interface is carried out inside the source code file "**RHSIF.c**". The related function is named with "**loc_RHSIF_portInit**".

#1. The Port control registers are protected against changes in their settings. To make a configuration possible you have to enable the write permission by setting the correct bit sequence to the Port Access Protection register PORT0.PKCPROT.

a. After the previous step the write access to the port groups P20 and P21 have to be enabled by writing to the PORT0.PWE register.

#2. For the Port group 21 the pins 2 and 3 supports the RHSIF RX function in LVDS mode

a. The PORT0.LVDSCTRLA register configures the pins to 3.3V LVDS mode as input pins.

b. With the PORT0.PFC21 and PORT0.PM21 registers the pin P21_2 and P21_3 are set into the HSIF_RXDN and HSIF_RXDP mode.

#3. For the Port group 21 the pins 4 and 5 supports the RHSIF TX function in LVDS mode

a. The PORT0.LVDSCTRLA register enables the LVDS output for P21_4 and P21_5

b. With the PORT0.PFC21, PORT0.PMC21, PORT0.PFC21 and PORT0.PSFSC21 register the pins P21_4 and P21_5 are set into output mode with HSIF_TSDN and HSIF_TXDP.

#4. The RHSIF clock signal is supplied with the pin P22_3. The pin is configured via the PORT0.PFC22 and PORT0.PFCAE22 and PORT0.PMC22 registers.

a. Depending on the selected RHSIF mode the P22_3 pin will be configured to input or output by the PORT0.PM22 register.

#5. After all port registers are set to the necessary settings we have to enable the Port Access Protection by disabling the write access in the PORT11.PWE register. Finally the PORT11.PKCPROT register needs to be set to with the activation sequence value.

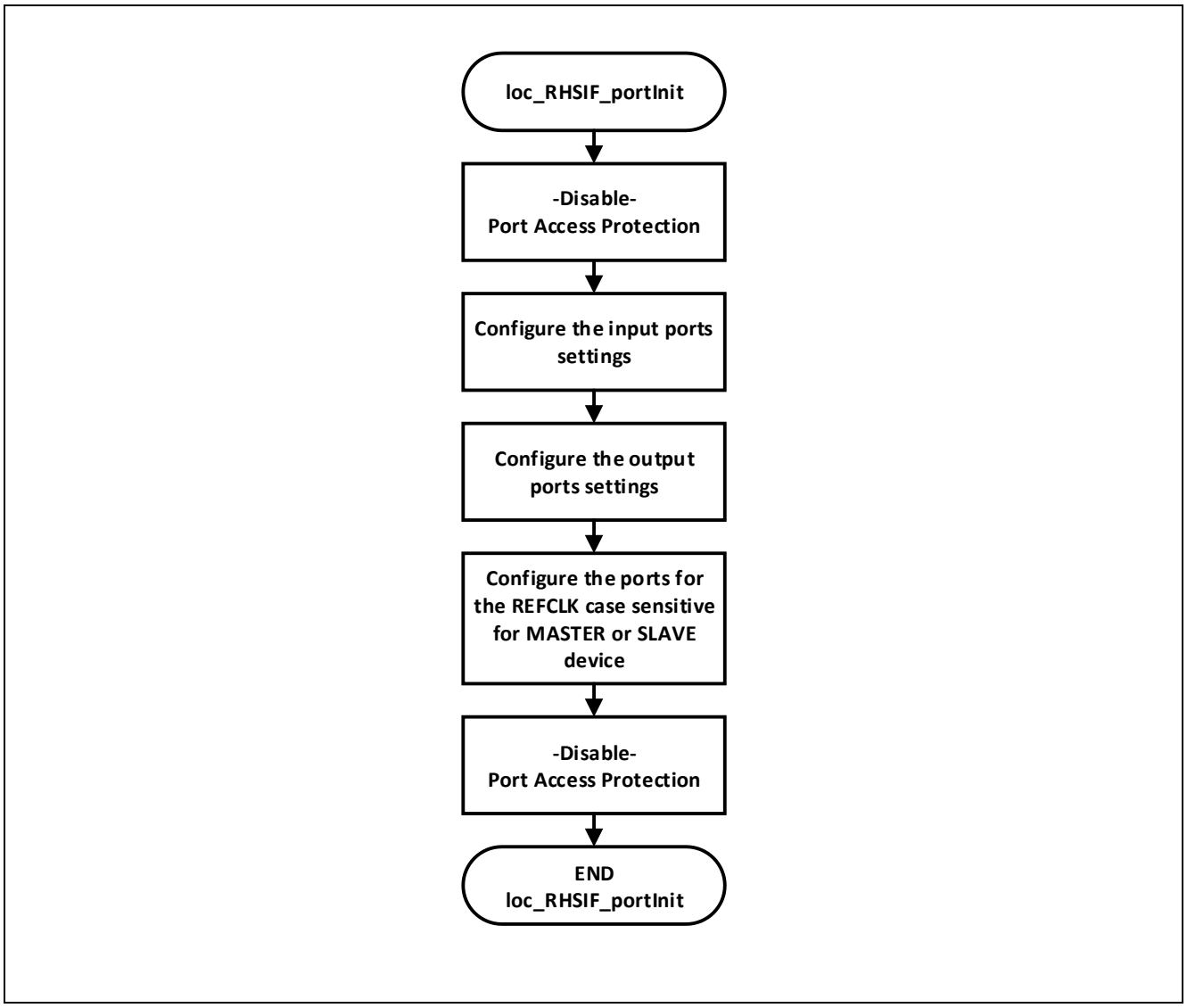

Figure 24: Flowchart "loc_RHSIF_portInit" Function

7.6 RHSIF L1 communication

7.6.1 Introduction

The available configuration and command for the L1 sub module are very limited. For the example we implement a PING command in the software by using one of the different ICLC commands. The software is implemented according to the workflow shown in the manual.

7.6.2 PING command

The RHSIF L1 sub module offers an automatic PING function. This function is implement within the "**RHSIF.c**" source code file with the function "**RHSIF_L1_SendPing**".

#1. First the function check if the RHSIF driver is initialized. When this statement is true the program performs the PING transmission.

   a. The program stops further execution until the bit ITRG in the RHSIF0.ICCR indicates that no current ICLC Command is triggered.

   b. Then all error and status register will be cleared via their related status clear registers.

   c. Afterwards a PING command (indicated by the chosen payload $(0x00)_H$) and the ICLC command TX trigger is issued by writing to the RHSIF0.ICCR register.

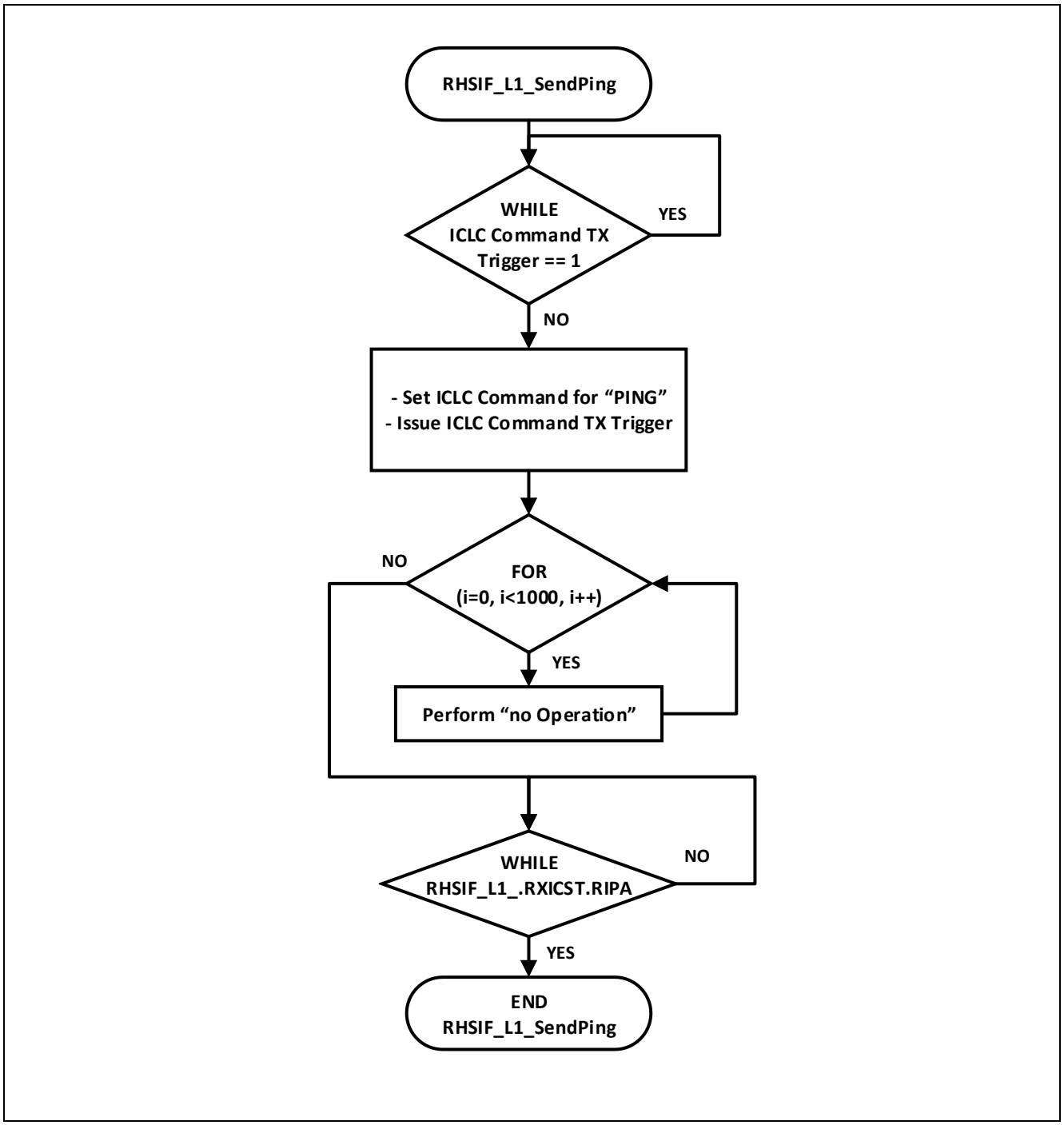

Figure 25: Flowchart “RHSIF_L1_SendPing” Function

7.7 RHSIF L2 communication

7.7.1 Introduction

With the L2 related registers the user can control the general RHSIF communication. For the example a READ and WRITE command is implemented and triggered by the SEND command. To unlock and establish the interconnection between the two link partners the 1st level authentication (Key-ID based) has to be carried out before the first transmission.

7.7.2 1st Authentication

The Key-ID based 1st level authentication is shown inside the "**RHSIF.c**" file in the function "**RHSIF_L2_Send1stAuth**"

#1. The initiator RHSIF node command writes the defined key-ID into the targeted RHSIF node. The write sequence is carried out with the RHSIF_L2_write32 function, which will be also explained in this chapter.

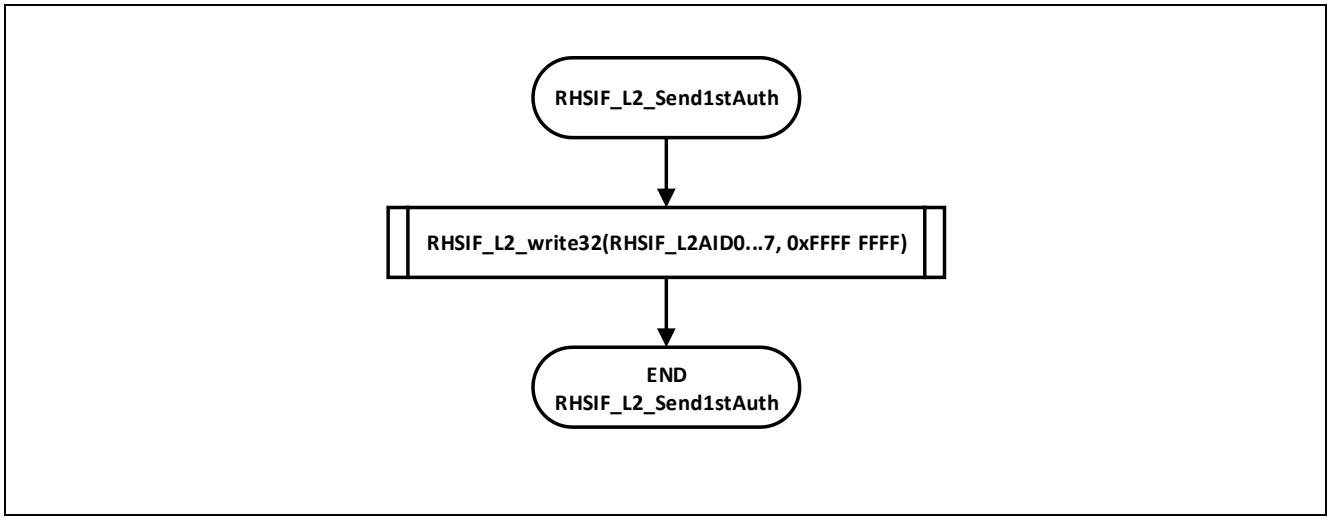

Figure 26: Flowchart "RHSIF_L2_Send1stAuth" Function

7.7.3 1st Authentication Check

The check for a valid Key-ID based 1st level authentication is carried out inside the "**RHSIF.c**" file with the function "**RHSIF_L2_wait1stAuth_Check**"

#1. The function starts with a while loop which is polling the status of the RHSIF0_L2MST.AUTS0 bit. When the bit becomes "1" a valid authentication is detected, the variable "the_state" changes to the status *RHSIF_STATE_AUTHENTICATED*.

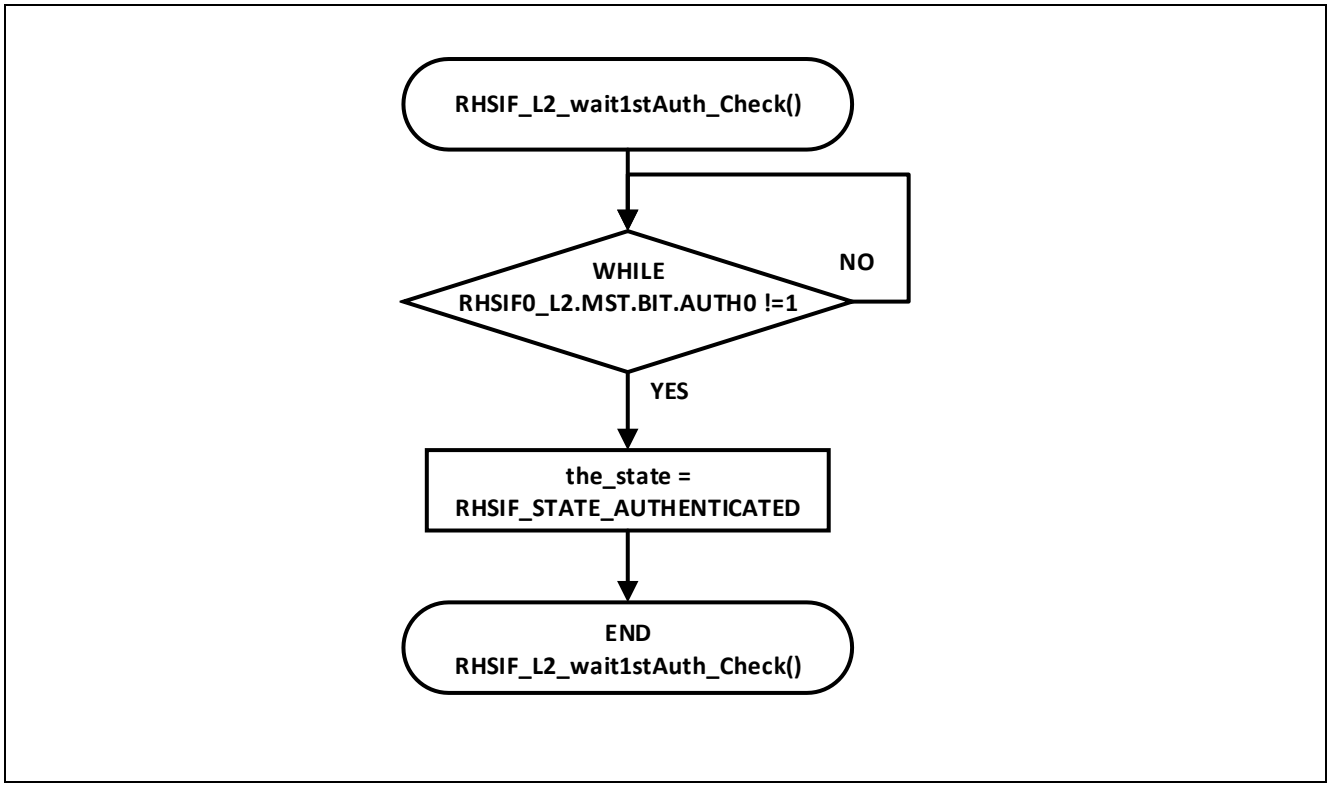

Figure 27: Flowchart "RHSIF_L2_wait1stAuth_Check" Function

7.7.4 Read32 command

A read command operation is shown in the "**RHSIF.c**" file within the function "**RHSIF_L2_read32**".

#1. The function starts with 4 different switch cases for the interface channel n ready bits. This function ensures, that the read process is only started when the interface is ready to receive new commands. So no data can get lost due to a busy interface.

#2. In the next first step the window number is checked against value which was handover together with the function call. The next steps will only be carried out when the window number is below "4"

   a. First the status register is cleared.

   b. After this the address will be written.

   c. Then the L2 Command_read32 is triggered.

#3. The WHILE-Loop checks the RDY bit in the channel n status register. When it is set to "1" the read command is issued to the link partner.

#4. At last an IF-statement checks, if the read data from the link partner is available inside the register

   a. When the statement is true, the data is read from the register and stored into the variable "*data" is pointing to. And the function returned with a "1".

   b. Otherwise a "0" is returned.

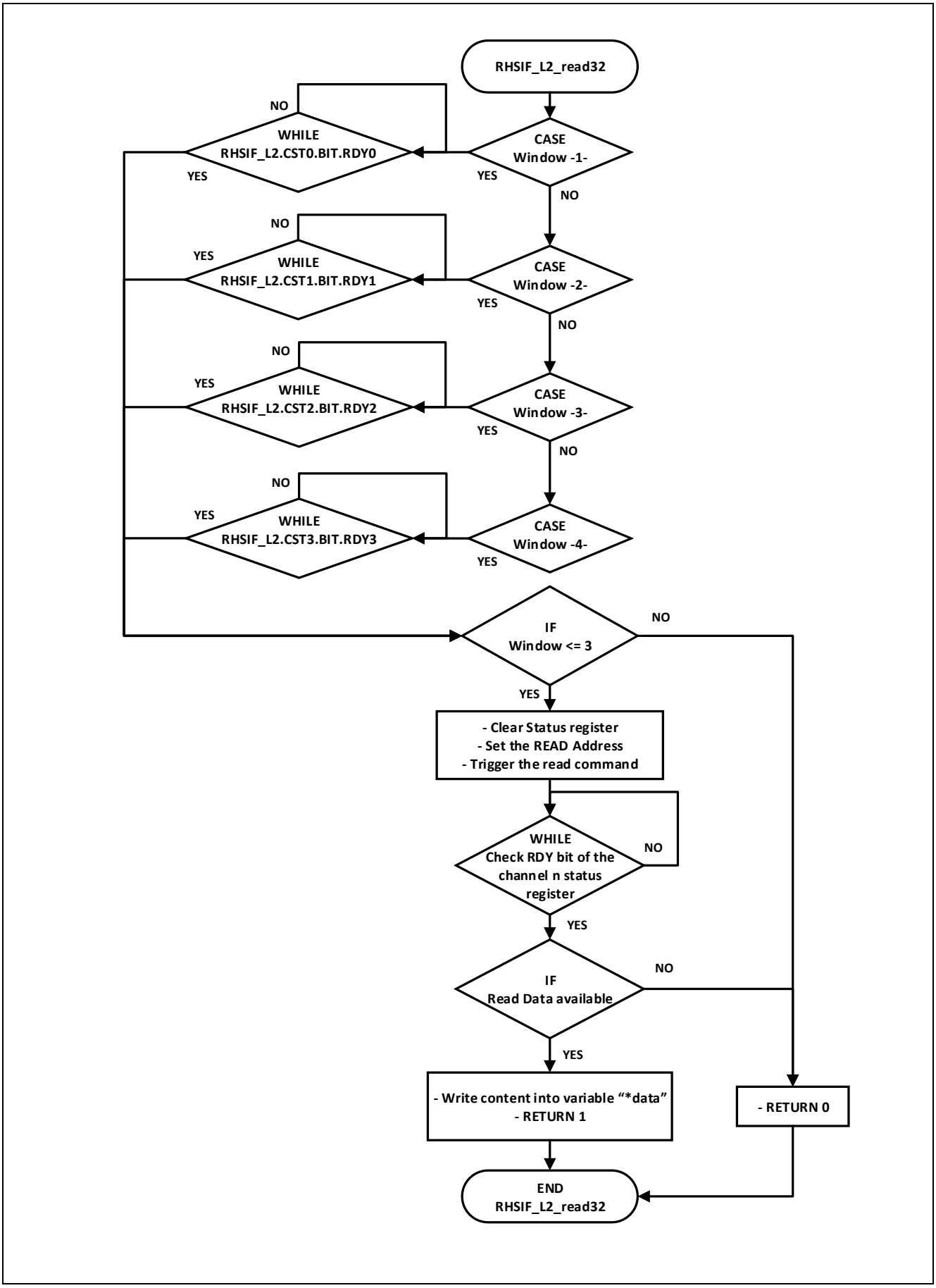

Figure 28: Flowchart "RHSIF_L2_read32" Function

7.7.5 Write32 command

A write command operation is shown in the "**RHSIF.c**" file within the function "**RHSIF_L2_write32**".

#1. The function is implemented with a SWITCH CASE command. First the four different possible cases Window 0 to Window 3 are defined. For each case the similar steps are carried out. The only difference are the dedicated register addresses. This address have to match the RHSIFnCSTn.BIT.RDYn for channel 0 or 3 depending on the chosen window number.

   a. First a WHILE-Loop is carried out to check the RHSIF_L2.CST0.BIT.RDY bit against "0". When this is false L2 is ready to transmit requests and the next steps are carried out.

   b. The channel n status register is cleared.

   c. The data which will be written is set.

   d. The write address is prepared

   e. A L2 command for write32 is issued.

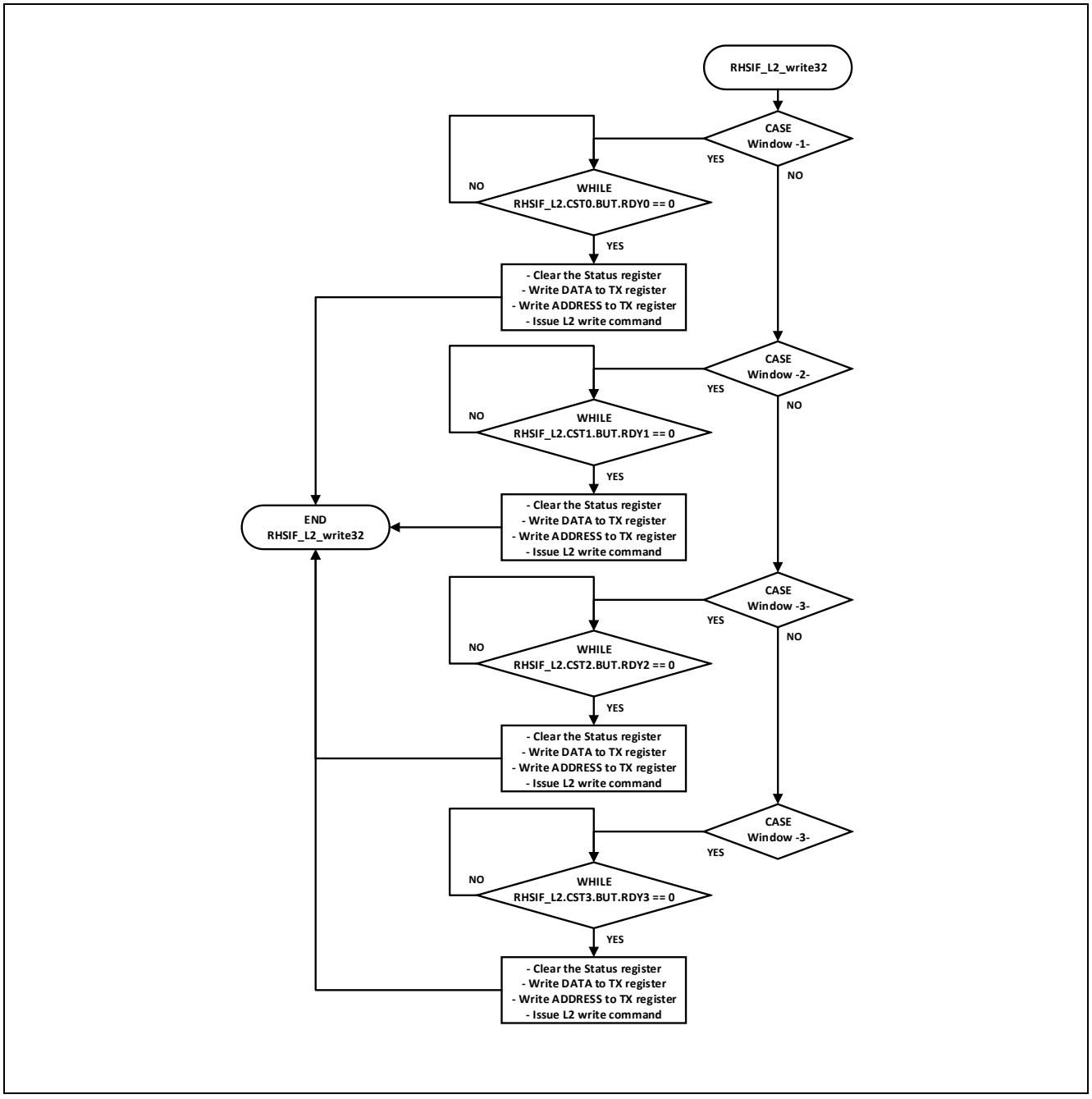

Figure 29: Flowchart "RHSIF_L2_write32" Function

7.7.6 Stream256 read command

A read command operation is shown in the "**RHSIF.c**" file within the function "**RHSIF_L2_stream256_read**".

#1. The function is implemented with a Stream receive command.

a. The data destination area address is set.

b. A L2 command for stream receive is issued.

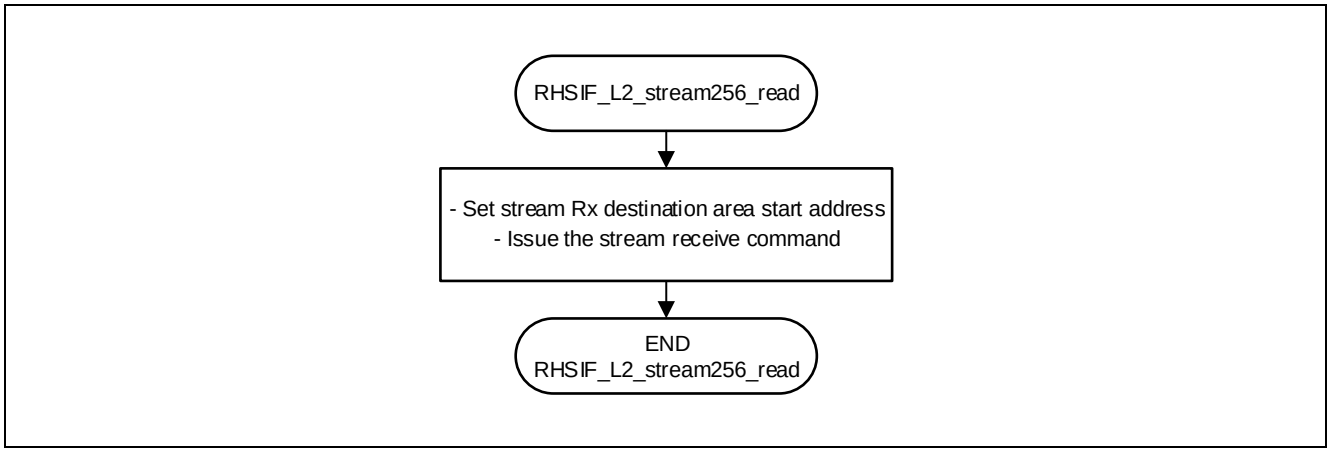

Figure 30: Flowchart "RHSIF_L2_stream256_read" Function

7.7.7 Stream256 write command

A write command operation is shown in the "**RHSIF.c**" file within the function "**RHSIF_L2_stream256_write**".

#1. The function is implemented with a Stream transmits command.

a. The channel 2 status register is cleared.

b. The stream data source address is set.

c. The channel 2 write address is prepared.

d. A L2 command for stream transmits is issued.

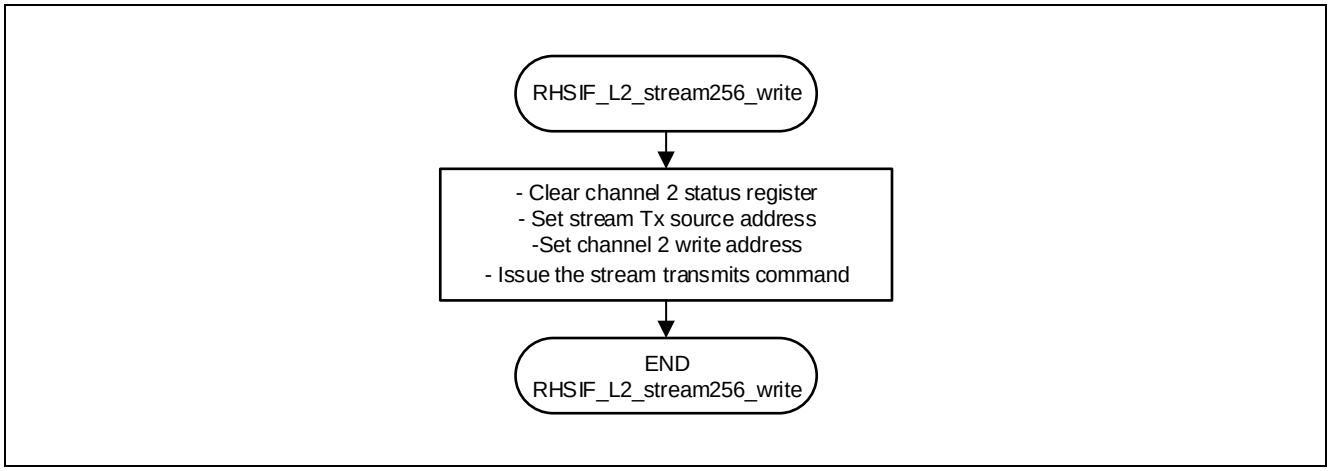

Figure 31: Flowchart "RHSIF_L2_stream256_read" Function

7.8 Transmitted and Received Data

7.8.1 Non-Stream transmitted and received data

The data will be transmitted to the respective targeted node and read back by the initiator. The data itself is stored into two different Arrays and can be read with the debugger after the transmission and reception is finished

- **L2_Data_Master[32]**
  — Contains the received data from the master device
- **L2_Data_Slave[32]**
  — Contains the received data from the slave device

The two variables contains the following data if the software runs without any errors.

- **L2_Data_Master[32]**
  — [0] = 1
  — [1] = 2
  — [2] = 4
  — [3] = 8
  — [4] = 16
  — [5] = 32
  — [6] = 64
  — [7] = 128
- **L2_Data_Slave[32]**
  — [0] = 256
  — [1] = 512
  — [2] = 1024
  — [3] = 2048
  — [4] = 4096
  — [5] = 8192
  — [6] = 16384
  — [7] = 32768

7.8.2 Stream transmitted and received data

The stream data will be transmitted from master device to slave device.

The slave device must have a receive data area of the size of data to be transmitted.

- L2_Data_Master_stream[65536]

Contains the stream transmit data to slave device.

- L2_Data_Slave_stream[65536]

Contains the stream received data from master device.

8 List of Abbreviations

Table 1: List of Abbreviations

Abbreviation	Description
MCU	Micro-Controller-Unit
RHSIF	Renesas High-Speed Serial Interface
L1	The “Datalink Layer” sub module of RHSIF
L2	The “Transport Layer” sub module of RHSIF
PEID	Processor Element ID
SPID	Serial Programmer ID
ECM	Error Correction Module
HBG	H-Bus Guard
CRG	Cluster RAM Guard
PBG	Peripheral Bus Guard
ICUM	Integrated-Cryptographic-Unit-Medium
ICLC	Interface Control Logical Channel
CTS	Clear To Send
RFP	Renesas Flash Programmer

9 Notes for the attached Demonstration Software

The developed software for this application note was created with the “Greenhills MULTI” editing and debugging environment. You can find the project data as a .zip compressed file within the RHSIF application note package files. Depending on the used device you have to edit the device files in the related register inside the project folders to get the software operating on your device.

CAUTION:

- **The programmed software is designed for demonstration purposes only.**
- **Before you implement parts of it to your own projects consider individual requirement for EMC, functional safety, quality and security purposes.**

---

10 Website and Support

Renesas Electronics Website

http://www.renesas.com/

Inquiries

http://www.renesas.com/contact/

11 Revision History

		Description	
Rev.	**Date**	**Page**	**Summary**
1.00	27.05.2022	All	Initial release version

12 General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Handling of Unused Pins

   Handle unused pins in accordance with the directions given under Handling of Unused Pins in the manual.

   — The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible. Unused pins should be handled as described under Handling of Unused Pins in the manual.

2. Processing at Power-on

   The state of the product is undefined at the moment when power is supplied.

   — The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the moment when power is supplied.
   In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the moment when power is supplied until the reset process is completed.
   In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the moment when power is supplied until the power reaches the level at which resetting has been specified.

3. Prohibition of Access to Reserved Addresses

   Access to reserved addresses is prohibited.

   — The reserved addresses are provided for the possible future expansion of functions. Do not access these addresses; the correct operation of LSI is not guaranteed if they are accessed.

4. Clock Signals

   After applying a reset, only release the reset line after the operating clock signal has become stable. When switching the clock signal during program execution, wait until the target clock signal has stabilized.

   — When the clock signal is generated with an external resonator (or from an external oscillator) during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Moreover, when switching to a clock signal produced with an external resonator (or by an external oscillator) while program execution is in progress, wait until the target clock signal is stable.

5. Differences between Products

   Before changing from one product to another, i.e. to a product with a different part number, confirm that the change will not lead to problems.

   — The characteristics of Microprocessing unit or Microcontroller unit products in the same group but having a different part number may differ in terms of the internal memory capacity, layout pattern, and other factors, which can affect the ranges of electrical characteristics, such as characteristic values, operating margins, immunity to noise, and amount of radiated noise. When changing to a product with a different part number, implement a system-evaluation test for the given product.

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 disputes 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, drawing, chart, program, algorithm, 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 otherwise misappropriate 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, copy or otherwise misappropriation of Renesas Electronics products.
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; and 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.

   Renesas Electronics products are neither intended nor 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 damages (space and 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 third parties arising from the use of any Renesas Electronics product for which the product is not intended by Renesas Electronics.
6. When using the 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 radiation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions or failure or accident arising out of the use of Renesas Electronics products beyond 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. Further, Renesas Electronics products are not subject to radiation resistance design. Please ensure to implement safety measures to guard them against the possibility of bodily injury, injury or damage caused by fire, and social damage in the event of 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 by your own responsibility as warranty for your products/system. Because the evaluation of microcomputer software alone is very difficult and not practical, please evaluate 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. Please investigate applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive carefully and sufficiently and use 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 not use Renesas Electronics products or technologies for (1) any purpose relating to the development, design, manufacture, use, stockpiling, etc., of weapons of mass destruction, such as nuclear weapons, chemical weapons, or biological weapons, or missiles (including unmanned aerial vehicles (UAVs)) for delivering such weapons, (2) any purpose relating to the development, design, manufacture, or use of conventional weapons, or (3) any other purpose of disturbing international peace and security, and you shall not sell, export, lease, transfer, or release Renesas Electronics products or technologies to any third party whether directly or indirectly with knowledge or reason to know that the third party or any other party will engage in the activities described above. When exporting, selling, transferring, etc., Renesas Electronics products or technologies, you shall comply with any applicable export control laws and regulations promulgated and administered by the governments of the countries asserting jurisdiction over the parties or transactions.
10. Please acknowledge and agree that you shall bear all the losses and damages which are incurred from the misuse or violation of the terms and conditions described in this document, including this notice, and hold Renesas Electronics harmless, if such misuse or violation results from your resale or making Renesas Electronics products available any third party.
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 majority-owned subsidiaries.

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

(Rev.3.0-1 November 2016)

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.
2801 Scott Boulevard Santa Clara, CA 95050-2549, U.S.A.
Tel: +1-408-588-6000, Fax: +1-408-588-6130

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: +44-1628-585-100, Fax: +44-1628-585-900

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, P. R. China 200333
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 II Stage, Indiranagar, Bangalore, India
Tel: +91-80-67208700, Fax: +91-80-67208777

Renesas Electronics Korea Co., Ltd.
12F., 234 Teheran-ro, Gangnam-Gu, Seoul, 135-080, Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5141

Colophon 6.0