Bluetooth® Low Energy Protocol Stack

BLE Virtual UART Application

R01AN3130EJ0111
Rev.1.11
Nov 22, 2016

Introduction

This manual describes BLE virtual UART application overview, software architecture, functions, setup procedures and implementation details.

The application runs with Renesas Bluetooth® Low Energy Protocol Stack on a Renesas RL78/G1D device as embedded configuration and provides the following functions.

- Simple AT command function to control and configure BLE connection
- Virtual UART function to send / receive characters to / from a remote device over BLE communication

Target Device

RL78/G1D

Related Documents

Document Name	Document No.
Bluetooth® Low Energy Protocol Stack	
User's Manual	R01UW0095E
API Reference Manual: Basics	R01UW0088E
Quick Start Guide	R01AN2767E
GUI Tool	R01AN2469E
Bluetooth® Specification	
Vol 6. Low Energy Controller volume	Core_v4.2

Contents

1. Overview

This manual describes BLE virtual UART application overview, software architecture, functions, setup procedures and implementation details.

The application runs with Renesas Bluetooth® Low Energy Protocol Stack on a Renesas RL78/G1D device as embedded configuration and provides the following functions.

- Simple AT command function to control and configure BLE connection
- Virtual UART function to send / receive characters to / from a remote device over BLE communication

1.1 Application Behavior

Figure 1-1 shows the application execution environment setup.

Prepare two RL78/G1D evaluation boards and write the application onto them. Then connect them respectively to PC via USB cable. A user operates the application through a terminal software.

Simple AT command can be used to establish, disconnect and configure BLE connection. After establishing a BLE connection, characters typed on the local terminal are sent to a remote device and are displayed on the remote terminal. In the contrary, characters typed on the remote terminal are sent to the local device and are displayed on the local terminal.

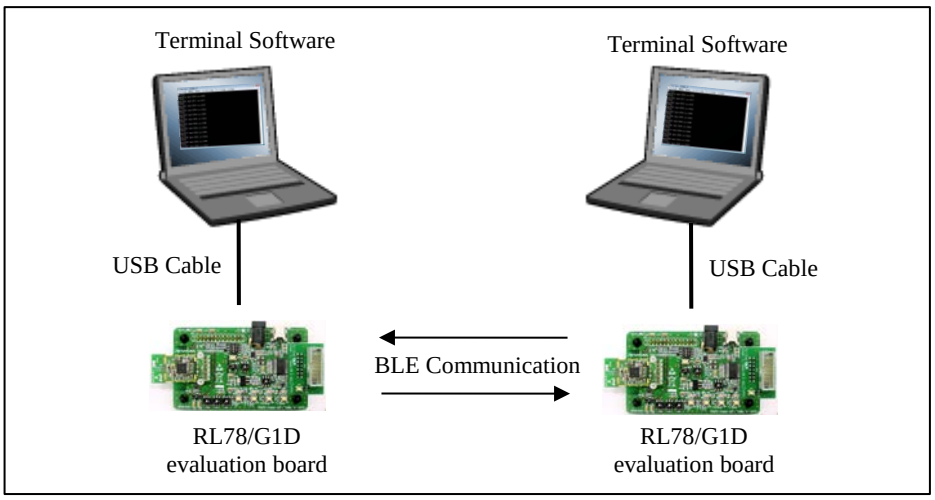


Figure 1-1: Application execution environment

2. Architecture

2.1 Software Architecture

Figure 2-1 shows software architecture of this application.

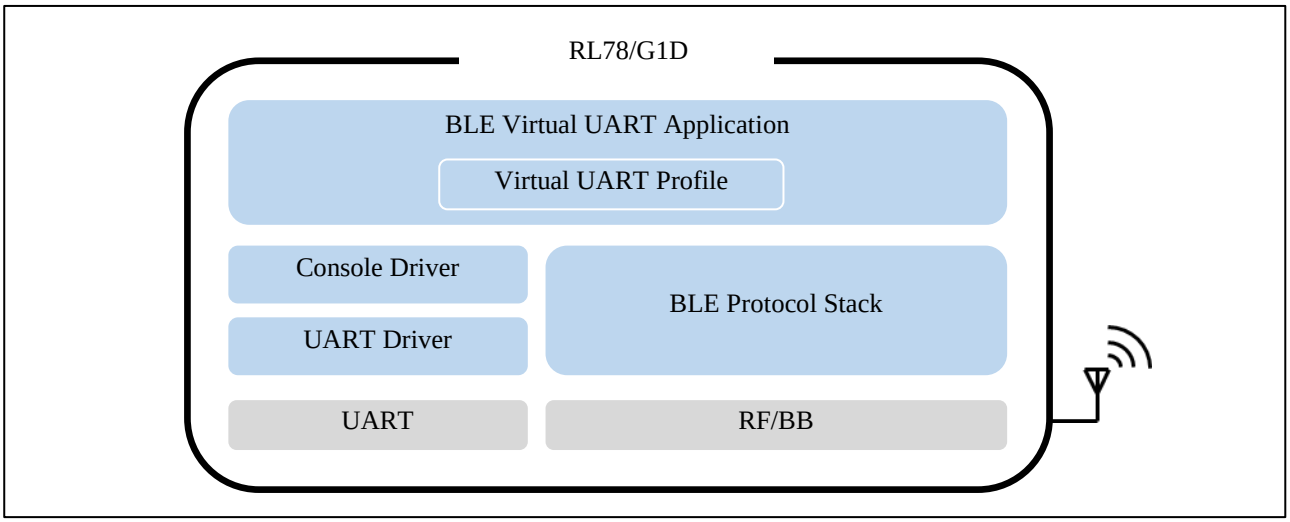


Figure 2-1: Software architecture

Table 2-1 lists software components.

Table 2-1: Software components

Component	Description
BLE Virtual UART Application	The application is used for execution of simple AT command and transmission of characters. Virtual UART profile is defined to transfer characters.
Console Driver	The driver is used to relay characters between a terminal software and BLE virtual UART application by using UART Driver functionality.
UART Driver	The device driver to control UART IP of RL78/G1D.
BLE Protocol Stack	Renesas Bluetooth® Low Energy Protocol Stack. Refer to Bluetooth® Low Energy Protocol Stack User's manual [R1UW0095E].

2.2 File Composition

The application is implemented based on BLE software which include BLE Protocol Stack. In this section, only files that have modified or added to the BLE software are listed. The modified files are marked with (M) and the added files are marked with (A).

```
RL78G1D_Virtual_UART_App/
    ├─── Project_Source/
    │    ├─── rBLE/
    │    │    └─── src/
    │    │         ├─── sample_app/
    │    │         │    ├─── r_vuart_app.c              (A)  ┐ BLE Virtual UART Application
    │    │         │    ├─── r_vuart_app.h              (A)  │
    │    │         │    ├─── r_vuart_app_param.c       (A)  ┘
    │    │         │    ├─── r_vuart_console.c         (A)  ┐ Console Driver
    │    │         │    └─── r_vuart_console.h         (A)  ┘
    │    │         └─── sample_profile/
    │    │              └─── vuart/
    │    │                   ├─── vuart.h               (A)  ┐ Virtual UART Profile
    │    │                   ├─── vuartc.c              (A)  │
    │    │                   ├─── vuartc.h              (A)  │
    │    │                   ├─── vuarts.c              (A)  │
    │    │                   └─── vuarts.h              (A)  ┘
    │    └─── renesas/
    │         ├─── src/
    │         │    ├─── arch/
    │         │    │    └─── rl78/
    │         │    │         ├─── arch_main.c          (M)  Modified to enable low power mode
    │         │    │         ├─── db_handle.h          (M)  ┐ Modified for Virtual UART Profiles
    │         │    │         ├─── ke_conf.c            (M)  │
    │         │    │         ├─── prf_config.c         (M)  │
    │         │    │         └─── prf_config.h         (M)  ┘
    │         │    ├─── driver/
    │         │    │    ├─── dataflash/
    │         │    │    │    ├─── eel_descriptor_t02.c (M)  ┐ Modified to add definitions to
    │         │    │    │    └─── eel_descriptor_t02.h (M)  ┘ access the Data Flash
    │         │    │    ├─── uart/
    │         │    │    │    └─── uart.c               (M)  Modified to enable low power mode
    │         │    └─── tools/
    │         │         └─── project/
    │         │              ├─── CS_CCRL/             (M)  ┐
    │         │              ├─── CubeSuite/           (M)  │Project files for
    │         │              ├─── e2studio/            (M)  │development environments
    │         │              └─── iar_v2/              (M)  ┘
    └─── ROM_File/
         ├─── ca78k0r/                                      ┐
         │    └─── Embedded/                                │ca78k0r generated firmware
         │         └─── RL78_G1D_CCE(VUART).hex       (A)  ┘
         ├─── ccrl/                                         ┐
         │    └─── Embedded/                                │ccrl generated firmware
         │         └─── RL78_G1D_CE(VUART).hex        (A)  ┘
         └─── iar_v2/                                       ┐
              └─── Embedded/                                │iar_v2 generated firmware
                   └─── RL78_G1D_IE(VUART).hex       (A)  ┘
```

3. Application Mode

The application have two modes as shown in Table 3-1.

A user can switch the application modes by ESC key (ASCII: 0x1B) through a terminal software.

Table 3-1: Application mode

Application Mode	Description
Simple AT Command Mode	This mode is used for execution of the simple AT command to control and configure a BLE connection. Refer to section 4 for the detail. In this mode, characters are never sent to the remote device.
Virtual UART Mode	This mode is used for transmission of characters over BLE communication. Refer to section 5 for the detail. In this mode, characters typed during the application in disconnect state are not sent to the remote device.

4. Simple AT Command Mode

In Simple AT Command Mode, a user can control and configure BLE connection by simple AT command. Table 4-1 shows the simple AT commands that the application supports.

Table 4-1: List of simple AT commands

Simple AT command	Description
AT-C	Create a connection to the address specified by AT-AP=<addr>.
AT-C=<addr>	Create a connection to the address specified by <addr>.
AT-R	When the device is in connect state, disconnect the connection and start advertising. When the device is in disconnect state, start advertising.
AT-AS=<addr>	Set <addr> as the public device address of the local device.
AT-AS?	Display the public device address of the local device.
AT-AP=<addr>	Set the address used by AT-C.
AT-AP?	Display the address used by AT-C.
AT-DS	Display the address of devices which support virtual UART profile.
AT-S?	Display the application state. Connect state or Disconnect state.
AT-CI=<con_intv>	Change Connection Interval.
AT-CI?	Display Connection interval setting value.
ATE0	Disable local echo.
ATE1	Enable local echo.

4.1 Details of Simple AT Command

4.1.1 AT-C

Description	Create a connection to the address specified by AT-AP=<addr>	
Response	OK	Success
	ERROR	Failed due to the application is connect state
	CONNECT	Connection established
Command Example	AT-C OK CONNECT	

4.1.2 AT-C=<addr>

Description	Create a connection to the address specified by <addr>	
Response	OK	Success
	ERROR	Failed due to the application is connect state
	CONNECT	Connection established
Command Example	AT-C=CBA987654321 (Set CB:A9:87:65:43:21) OK CONNECT	

4.1.3 AT-R

Description	When the application is in connect state, disconnect the connection and start advertising. When the application is in disconnect state, start advertising.	
Response	OK	Success
	DISCONNECT	Disconnected
Command Example	[Connect state] AT-R OK DISCONNECT [Disconnect state] AT-R OK	

4.1.4 AT-AS=<addr>

Description	Set <addr> as the local device public device address. The address set by this command is preserved over power cycles. The address set by this command is reflected after reset. Please reset the system for example by pushing the reset button on the board.	
Response	OK	Success
	ERROR	Failed due to the application is connect state
Command Example	AT-AS=CCCCBBBBAAAA (Set CC:CC:BB:BB:AA:AA) OK	

4.1.5 AT-AS?

Description	Display the local device public address.	
Response	OK	Success
Command Example	AT-AS? -AS: CCCCBBBBAAAA (The address is CC:CC:BB:BB:AA:AA) OK	

4.1.6 AT-AP=<addr>

Description	Set <addr> as the public device address used by AT-C. The address set by this command is preserved over power cycles.	
Response	OK	Success
	ERROR	Failed due to the application is connect state
Command Example	AT-AP=CCCCBBBBAAAA (Set CC:CC:BB:BB:AA:AA) OK	

4.1.7 AT-AP?

Description	Display the public device address used by AT-C.	
Response	OK	Success
Command Example	AT-AP? -AP: CCCCBBBBAAAA (The address is CC:CC:BB:BB:AA:AA) OK	

4.1.8 AT-DS

Description	Display the address of the device which support virtual UART profile. Whether a device is supporting virtual UART profile is confirmed by checking the advertising data includes virtual UART service UUID.	
Response	OK	Success
	ERROR	Failed due to the application is connect state
Command Example	AT-DS -DS: CBA987654321 (The address is CB:A9:87:65:43:21) -DS: CCCCBBBBAAAA (The address is CC:CC:BB:BB:AA:AA) OK	

4.1.9 AT-S?

Description	Display the local device address connect state. The state is CONNECT or DISCONNECT.	
Response	OK	Success
Command Example	[Connect state] AT-S? CONNECT OK [Disconnect state] AT-S? DISCONNECT OK	

4.1.10 AT-CI=<con_intv>

Description	Change Connection Interval. Execute this command in Disconnect state, the application retains the con_intv value internally, and the value will be used for following connection. This command cannot change Connection Interval in Connect state. Connection Interval is calculated by following calculation. Connection Interval = con_intv * 1.25[ms] Ex) When you want to set 20[ms], execute AT-CI=16 The default value of Connection Interval is 15[ms]. Refer Figure 8-3 for Connection Interval Change sequence. After establishing the connection, Slave device requests Connection Interval parameter update to the Master device. Master device can decline the request depending on the restriction that Master device have. You can check whether requested Connection Interval is accepted by executing “AT-CI?” command.	
Response	OK	Success
	ERROR	Failed due to the application is connect state
		The setting value is out of range (Range: 6~3200)
Command Example	[Disconnect state] AT-CI=20 OK AT-CI=3201 [Connect state] AT-CI=20 ERROR	

4.1.11 AT-CI?

Description	When execute this command in Connect state, display Connection Interval of the connection. When execute this command in Disconnect state, display the retained Connection Interval. To compute the actual Connection Interval, multiply the response value by 1.25[ms]. Ex) If the response is “-CI: 20”, Connection Interval is 20 * 1.25[ms] = 25[ms].	
Response	OK	Success
Command Example	AT-CI? -CI: 20 OK	

4.1.12 ATE0

Description	Disable local echo.	
Response	OK	Success
Command Example	ATE0 OK	

4.1.13 ATE1

Description	Enable local echo.	
Response	OK	Success
Command Example	ATE1 OK	

5. Virtual UART Mode

After establishing BLE connection between two devices, a user can exchange characters with the remote device. A transferable character is ASCII printable characters and new-line character.

5.1 Virtual UART Profile

Character transfer is enabled by GATT based virtual UART profile. Refer to 8.1.

The connection initiating device, which is executes AT-C command, works as GATT client and the remote device works as GATT server. Below is character transfer details.

- To send characters from the client to the server, send “Write Request” to the server. The server receives the characters, and replies “Response” to the client.
- To send characters from the server to the client, send “Indication” to the client. The client receives the characters, and replies “Confirmation” to the server.

The characters reception by the remote device can be confirmed by waiting for the "Response" or "Confirmation" reply from the remote device.

Figure 5-1 shows the character transfer sequence.

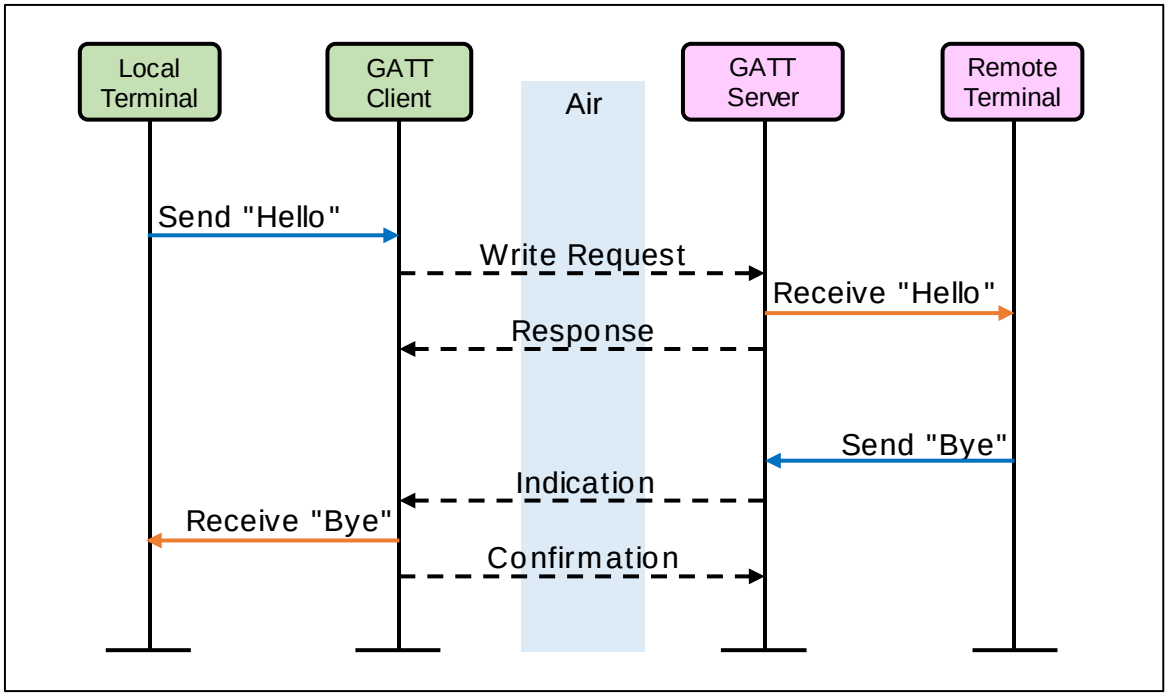


Figure 5-1: Character transfer sequence

5.2 Buffering of the Send Characters

In order to avoid the loss of send characters, the application have a send character buffer.

To send characters within the period between “Write Request” and “Response” or “Indication” and “Confirmation” is not possible. So characters typed within this period can be lost.

Virtual UART profile has a buffer and stores characters typed within this period. If there are characters in the buffer when receiving “Response” or “Confirmation” from the remote device, the profile send it soon.

Figure 5-2 shows a send character buffering sequence.

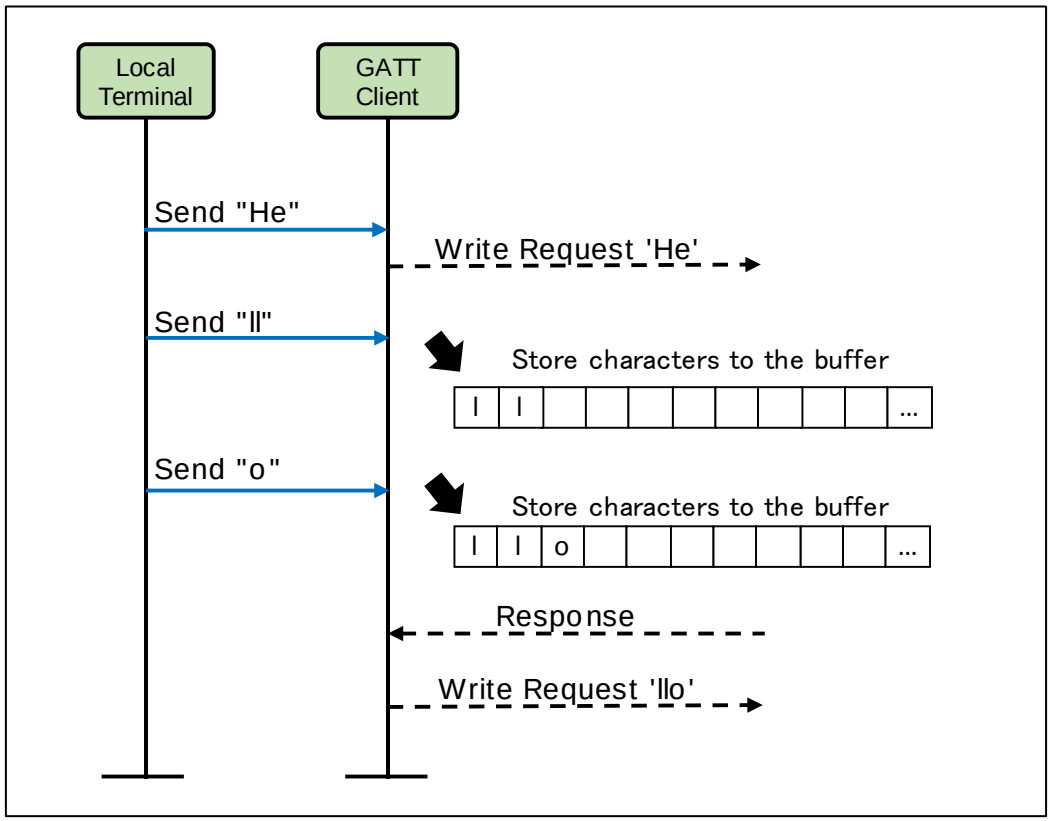


Figure 5-2: Send characters buffering sequence

5.3 Encryption of BLE Connection

To protect the BLE connection from such as eavesdropping, the encryption of the BLE connection is enabled.

The application does not hold the pairing information, it performs pairing on each connection.

6. Power Saving Function

6.1 CPU STOP Mode

When 3 seconds have elapsed after the last character transfer, CPU enters STOP mode in order to reduce the power consumption. During STOP mode, the blinking of the LED1/LED2 becomes slower or stops. If character transfer is occurred when CPU is in STOP mode, CPU returns from STOP mode soon.

6.2 Changes of Advertising Interval

When 30 seconds has elapsed after advertising started, the application sets a longer advertising interval in order to reduce the power consumption. The longer advertising interval is reset by re-enabling advertising with connection and disconnection or AT-R.

The default advertising interval is 30 milliseconds and the longer advertising interval is 3 seconds.

7. Operational Check

7.1 Environment

Below is the environment to use for application build and operation check.

- Hardware
  - Host PC
    - PC/AT™ compatible machine
    - Processor Speed : 1.6GHz or higher
    - Main Memory : 1GB or more
    - Interface : USB2.0 (Used for E1 and RL78/G1D evaluation board)
  - Device
    - RL78/G1D evaluation board [RTK0EN0001010001BZ]

- Tool
  - Renesas On-chip debug emulator E1

- Software
  - Windows®7 Service Pack1
  - e^2 studio V4.3.1.001 / RL78 Family C Compiler Package V1 (without IDE) V1.03.00
    or Renesas CS+ for CC V4.00.00 / RL78 Family C Compiler Package V1 (without IDE) V1.03.00
    or Renesas CS+ for CA, CX V3.02.00 / Renesas CA78K0R V1.72
    or IAR Embedded Workbench for Renesas RL78 V2.20.1
  - Renesas Flash Programmer v3.02.00
  - Teraterm Version 4.89
  - UART-USB conversion device driver

Note: It may be that device driver of UART-USB conversion IC “FT232RL” is requested is in the first connection with host. In this case, you can get the device driver from below link.

FTDI (Future Technology Device International) – Drivers
http://www.ftdichip.com/Drivers/D2XX.html

7.2 Build Procedure

This section describes how to build the application. Using with one of the following development environment can build the application for demonstration.

- e2 studio V4.3.1.001 / RL78 Family C Compiler Package V1 (without IDE) V1.03.00
- Renesas CS+ for CC V4.00.00 / RL78 Family C Compiler Package V1 (without IDE) V1.03.00
- Renesas CS+ for CA, CX V3.02.00 / Renesas CA78K0R V1.72
- IAR Embedded Workbench for Renesas RL78 V2.20.1

7.2.1 Common Procedure

Followings are the procedures to build the application.

1. You need BLE protocol stack and EEPROM emulation library. Get these from Renesas Web page.
    - BLE protocol stack
        - https://www.renesas.com/en-in/solutions/key-technology/connectivity/bluetooth-smart/protocol-stack.html
    - EEPROM emulation library
        - For CA78K0R
            - https://www.renesas.com/en-us/software/D3017959.html
        - For CC-RL
            - https://www.renesas.com/en-us/software/D3017960.html
        - For IARv2:
            - RENESAS_EEL_RL78_T02E_IARV2_V1.00.zip
            - RENESAS_FDL_RL78_T02E_IARV2_V1.00.zip

    NOTE: The link address can be changed without notice.
2. Unzip BLE protocol stack. The path to unzip the package should not include white spaces or multi-byte characters. This manual uses “\BLE_Software_Ver_X_XX” as the target path.
3. Install EEPROM emulation library. Refer “Quick Start Guide [R01AN2767] section 4.2 Installing EEPROM Emulation Library” for the procedures.
4. Overwrite \BLE_Software_Ver_X_XX\RL78_G1D\Project_Source with the application Project_Source directory.

7.2.2 e² studio

1. Launch e² studio.
2. Right click on “Project Explorer” and select “Import…” from the dropdown menu.
3. “Import” window is popped up and select “Existing Projects into Workspace” and click “Next >”.
4. Fill “Select root directory:” form with the project directory shown in Table 7-1 and make sure that the project you selected is displayed in “Projects:” and click “Finish”. Then the windows is closed.
5. Right click on the project just imported on “Project Explorer” and Select “Build Project” from the dropdown menu.
6. Refer Table 7-1 for the Hex file generate path.

7.2.3 CS+

1. Double click the project file shown in Table 7-1.
2. Right click on “BLE_Emb” in “Project Tree” and select “Build BLE_Emb” from the dropdown menu.
3. Refer Table 7-1 for the Hex file generate path.

7.2.4 IAR Embedded Workbench

1. Double click the workspace file shown in Table 7-1.
2. Push F7 key and start building.
3. Refer Table 7-1 for the Hex file generate path.

Table 7-1: Project file and Hex file Location

e² studio with CC-RL	
Project Directory	\BLE_Software_Ver_X_XX\RL78_G1D \Project_Source\renesas\tools\project\ e2studio\BLE_Embedded \rBLE_Emb
Firmware	\BLE_Software_Ver_X_XX \RL78_G1D\Project_Source\renesas\tools\project\ e2studio\ BLE_Embedded\rBLE_Emb\DefaultBuild\rBLE_Emb_CCRL.hex
CS+ with CC-RL	
Project File	\BLE_Software_Ver_X_XX \RL78_G1D \Project_Source\renesas\tools\project\ CS_CCRL\BLE_Embedded\rBLE_Emb.mtpj
Firmware	\BLE_Software_Ver_X_XX \RL78_G1D \Project_Source\renesas\tools\project\ CS_CCRL\BLE_Embedded\rBLE_Emb\DefaultBuild\rBLE_Emb_CCRL.hex
CS+ with CA78K0R	
Project File	\BLE_Software_Ver_X_XX \RL78_G1D \Project_Source\renesas\tools\project\ CubeSuite\BLE_Embedded\rBLE_Emb.mtpj
Firmware	\BLE_Software_Ver_X_XX \RL78_G1D \Project_Source\renesas\tools\project\ CubeSuite\BLE_Embedded\BLE_Emb\DefaultBuild\BLE_Emb.hex
IAR Embedded Workbench	
Workspace File	\BLE_Software_Ver_X_XX \RL78_G1D \Project_Source\renesas\tools\project\ iar_v2\BLE_Embedded\BLE_Embedded.eww
Firmware	\BLE_Software_Ver_X_XX \RL78_G1D \Project_Source\renesas\tools\project\ iar_v2\BLE_Embedded\rBLE_Emb_Peripheral\BLE_Emb\Debug\Exe\BLE_Emb.hex

7.3 Preparation for Execution Environment

1. Write the firmware onto two RL78/G1D evaluation boards. Refer to Bluetooth® Low Energy Protocol Stack Quick Guide [R01AN2767E] Section 5.
2. As shown in Figure 1-1, connect both RL78/G1D evaluation boards to PCs respectively.
3. Launch a terminal software on both PCs and configure them as Table 7-2.

Table 7-2: Terminal software configuration

Setting	Value
New-line Receive	LF
New-line Send	CR
Baud rate	4800 [bps]
Data length	8 [bit]
Parity bit	none
Stop bit	1 [bit]
Flow control	none

7.4 Usage Example

The following is the usage example of the application. In this example, set device addresses, establish a BLE connection, transfer characters and disconnect the connection.

Figure 7-12 shows the terminal result of the local device. Figure 7-23 shows the terminal result of the remote terminal. Figure 7-34 shows the sequence diagram of this example usage. The red numbers in figures are corresponding to the numbers in the following procedures.

1. Set local and remote device address. To set the device address use “AT-AS=<addr>” command. For example, to set 12:34:56:78:9A:BC, execute “AT-AS=123456789ABC”. To display current device address settings, use “AT-AS?”.
   If a device address is 00:00:00:00:00:00 or the local device and the remote device have the same addresses, you need to change the device address. In the following example, we assume that you set 12:34:56:78:9A:BC to the local device address, CB:A9:87:65:43:21 to the remote device address.
2. **When you change a device address by “AT-AS=<addr>” command, you need to reset the device to reflect the change by pushing RESET button (SW5) on the board.**
3. Execute “AT-AP=CBA987654321” to set the target device address for a connection.
4. Execute “AT-C” on the local terminal. This command start the connection to the device which have the address CB:A9:87:65:43:21. After the connection established, “CONNECT” response is displayed on both of the local and remote terminal.
5. Type ESC key on the local terminal to switch the application mode to Virtual UART mode.
6. Type “Hello” to the local terminal. Then “Hello” is displayed on the remote terminal.
7. Type ESC key on the remote terminal to switch the application mode to Virtual UART mode.
8. Type “Bye” on the remote terminal. Then “Bye” is displayed on the local terminal.
9. Type ESC key on the local terminal to switch the application mode to AT command mode.
10. Execute “AT-R” on the local terminal to disconnect the connection. After completing the disconnection, “DISCONNECT” response is displayed on both of the local and remote terminal.

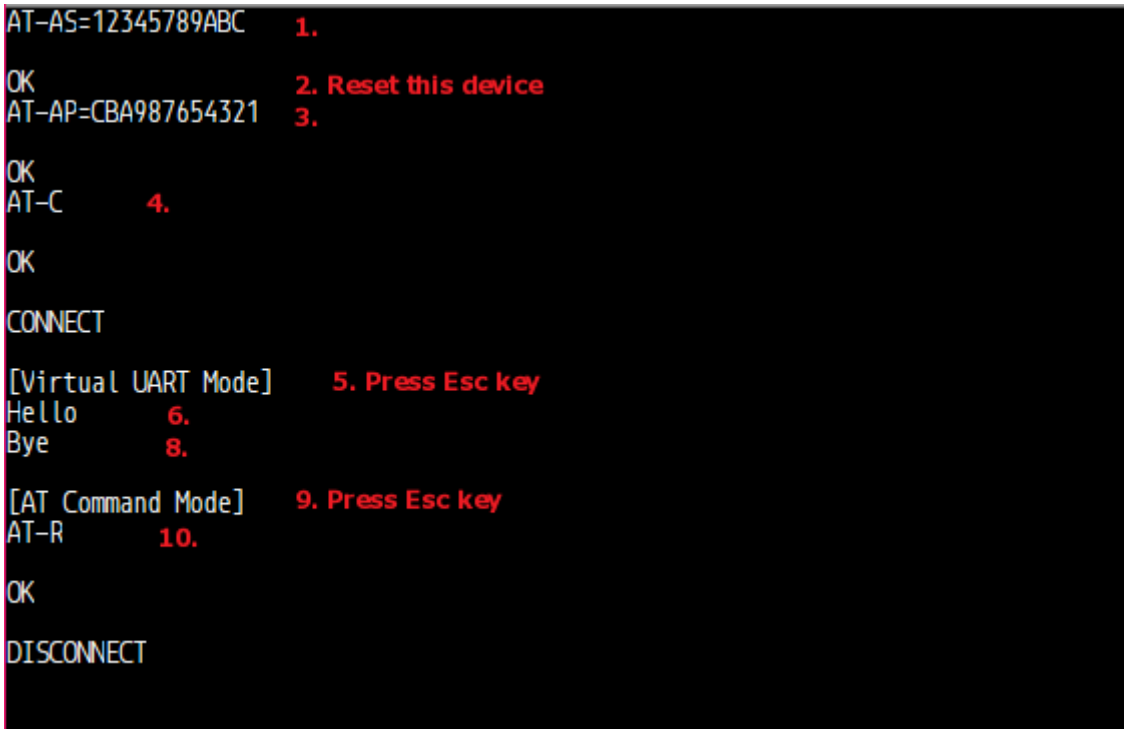


Figure 7-1: Local terminal result

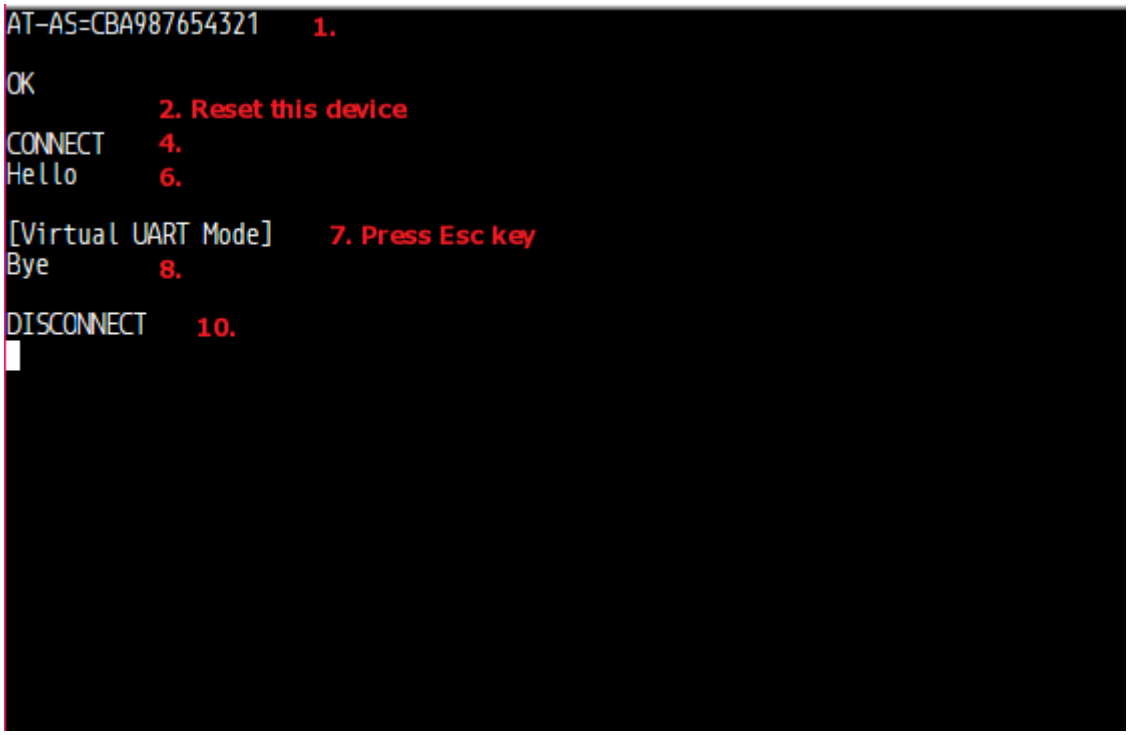


Figure 7-2: Remote terminal result

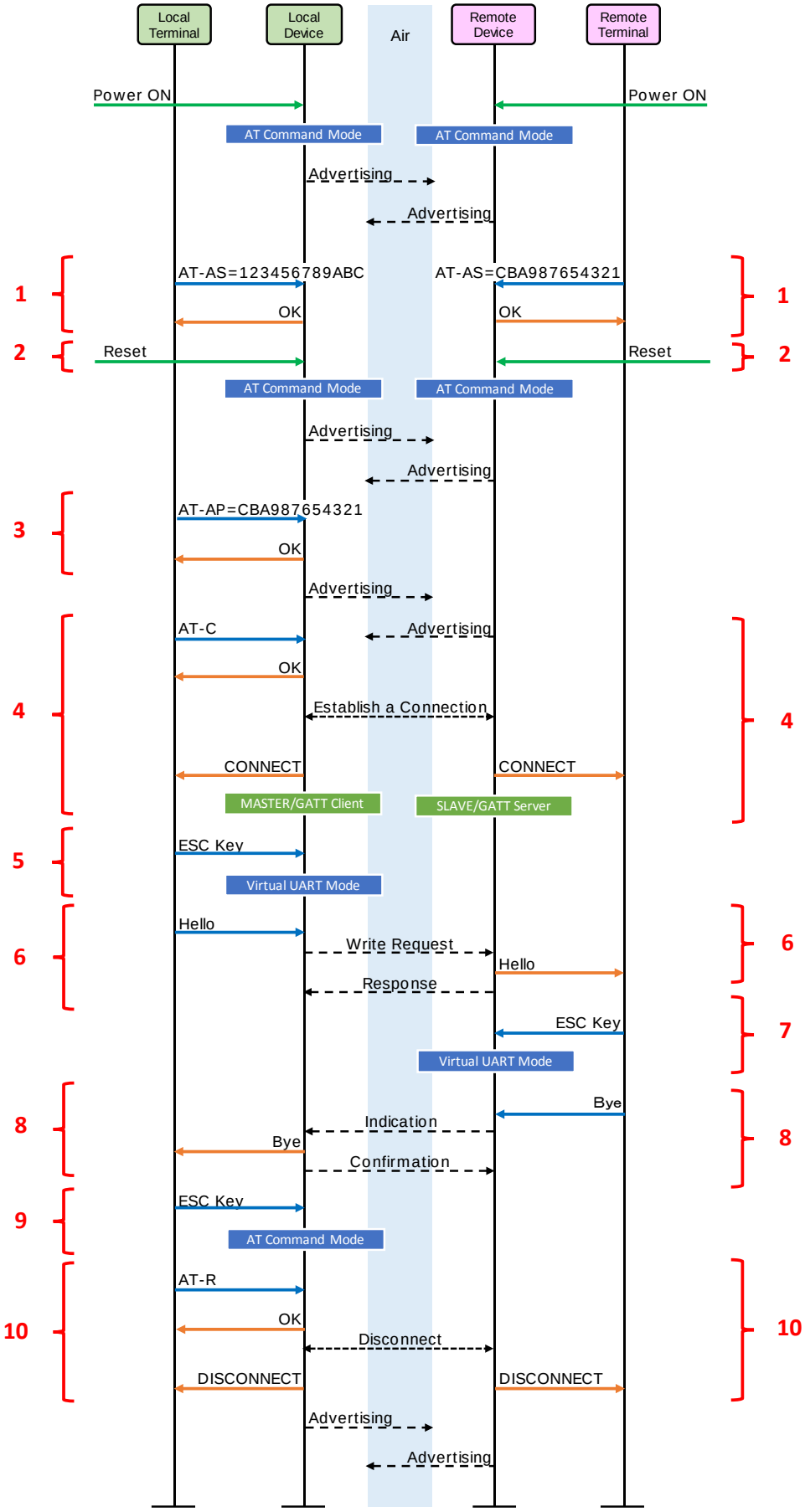


Figure 7-3: Example usage sequence

8. Implementation Details

8.1 Virtual UART Profile

Table 8-1 shows the specification of Virtual UART Profile.

Table 8-1: Virtual UART Profile specification

Attribute Handle	Attribute type and the value
VUART_HDL_SVC 0x000C	Type: Primary Service Declaration UUID: D68C0001-A21B-11E5-8CB8-0002A5D5C51B UUID for virtual UART service
VUART_HDL_INDICATION_CHAR 0x000D	Type: Characteristic Declaration UUID: D68C0002-A21B-11E5-8CB8-0002A5D5C51B Property: Indicate Used for character transfer from the server to the client
VUART_HDL_INDICATION_VAL 0x000E	Type: Indication Value By setting characters to this characteristic and send Indication, the characters are sent from the server to the client. Max 20 characters.
VUART_HDL_INDICATION_CFG 0x000F	Type: Client Characteristic Configuration Descriptor Used for Indication enable / disable of the server from the client
VUART_HDL_WRITE_CHAR 0x0010	Type: Characteristic Declaration UUID: D68C0003-A21B-11E5-8CB8-0002A5D5C51B Property: Write Used for character transfer from the client to the server.
VUART_HDL_WRITE_VAL 0x0011	Type: Write Value By writing characters to this characteristic with “Write Request”, the characters are sent from the client to the server. Max 20 characters.

Note: The hex value of attribute handle can be changed depends on profiles included in the firmware.

8.2 Advertising

Table 8-2 shows the default settings of advertising.

Table 8-2: Advertising specification

Advertising Type		Connectable undirected advertising (ADV_IND)
Advertising Interval Min		Default: 20 [ms], Low Power: 1.5 [s]
Advertising Interval Max		Default: 30 [ms], Low Power: 3 [s]
Advertising Channel Map		All Channels (37, 38, 39 ch)
Advertising Data		-
	Length of this Data	2 [bytes]
	Data Type	<<Flags>> (0x01)
	Flags	LE General Discoverable Mode BR/EDR Not Supported
	Length of this Data	8 [bytes]
	Data Type	<<Complete Local Name>> (0x09)
	Local Name	REL-BLE
	Length of this Data	17 [bytes]
	Data Type	<<Complete List of 128-bit Service Class UUIDs>> (0x07)
	UUID	D68C0001-A21B-11E5-8CB8-0002A5D5C51B
	Scan Response Data	none

8.3 Connection

Table 8-3 shows the default settings of connection.

Table 8-3: Connection specification

Scan Interval	30 [ms]
Scan Window Size	30 [ms]
Initiator Filter Policy	Ignore White List
Peer Address Type	Public Address
Peer BD Address	Specified by AT-C or AT-AP
Own Address Type	Public Address
Minimum of Connection Interval	15 [ms]
Maximum of Connection Interval	15 [ms]
Connection Latency	0 [ms]
Link Supervision Timeout	5 [s]
Minimum CE Length	0 [ms]
Maximum CE Length	50 [ms]

8.4 Pairing

Table 8-4 shows the pairing default settings.

Table 8-4: Pairing specification

Bonding	Bondable Mode
Security Mode	Unauthenticated pairing with encryption
Pairing Method	Just Works
IO capability	No Input No Output
OOB flag	OOB Data not present
Authentication Requirements	No MITM Bonding
Encryption key size	128 [bit]
Initiator key distribution	None
Responder key distribution	Encryption key

8.5 Virtual UART Function API

This section describes virtual UART function definitions and APIs.

8.5.1 Virtual UART Definitions

- Event type enumeration declaration

```
typedef enum {
  RBLE_VUART_EVENT_SERVER_ENABLE_CMP = 0x01,
  RBLE_VUART_EVENT_SERVER_WRITE_REQ,
  RBLE_VUART_EVENT_SERVER_INDICATION_CFM,
  RBLE_VUART_EVENT_CLIENT_ENABLE_CMP = 0x81,
  RBLE_VUART_EVENT_CLIENT_INDICATION,
  RBLE_VUART_EVENT_CLIENT_WRITE_RSP,
} RBLE_VUART_EVENT_TYPE;
```

- Event callback function declaration

```
typedef void (*RBLE_VUART_EVENT_HANDLER)(RBLE_VUART_EVENT *event);
```

- Event parameter structure

```
typedef struct RBLE_VUART_EVENT_t {
  RBLE_VUART_EVENT_TYPE type;     Virtual UART event type
  union Event_Vuart_Paramter_u {
    Server role enable completion event
    struct {
      RBLE_STATUS status;       Status
    } server_enable_cmp;
    Server role character receive event
    struct {
      RBLE_STATUS status;       Status
      char value[20];           Received characters
      uint16_t len;             Received characters length
    } server_write_req;
    Server role character send completion event
    struct {
      RBLE_STATUS status;       Status
    } server_indicate_cnf;
    Client role enable completion event
    struct {
      RBLE_STATUS status;       Status
    } client_enable_cmp;
    Client role characters receive event
    struct {
      RBLE_STATUS status;       Status
      char value[20];           Received characters
      uint16_t len;             Received characters length
    } client_indication;
    Client role character send completion event
    struct {
      RBLE_STATUS status;       Status
    } client_write_rsp;
  } param;
} RBLE_VUART_EVENT;
```

8.5.2 Function

(1) RBLE_VUART_Server_Enable

RBLE_STATUS RBLE_VUART_Server_Enable(uint16_t conhdl, RBLE_VUART_EVENT_HANDLER callback)		
This function enables server role of virtual UART function. The result is informed by RBLE_VUART_EVENT_SERVER_ENABLE_CMP event.		
Parameters:		
	conhdl	Connection handle
	callback	Callback for virtual UART event
Return:		
	RBLE_OK	Success
	RBLE_STATUS_ERROR	Failed due to rBLE mode is in RBLE_MODE_ACTIVE

(2) RBLE_VUART_Server_Disable

RBLE_STATUS RBLE_VUART_Server_Disable(void)		
This function disables server role of virtual UART function.		
Parameters:		
	-	-
Return:		
	RBLE_OK	Success

(3) RBLE_VUART_Server_Send_Indication

RBLE_STATUS RBLE_VUART_Server_Send_Indication(const char *chars, uint16_t len)		
This function send characters from the server to the client. The characters sent by the server are received by the client. After the reception, the client responses with Confirmation. The confirmation is informed to the server by RBLE_VUART_EVENT_SERVER_INDICATION_CFM event.		
Parameters:		
	chars	Received characters
	len	Received characters length
Return:		
	RBLE_OK	Success
	RBLE_STATUS_ERROR	Failed due to rBLE mode is in RBLE_MODE_ACTIVE

(4) RBLE_VUART_Client_Enable

`RBLE_STATUS RBLE_VUART_Client_Enable( uint16_t conhdl, RBLE_VUART_EVENT_HANDLER callback)`		
This function enables client role of virtual UART function. The result is informed by RBLE_VUART_EVENT_CLIENT_ENABLE_CMP event.		
Parameters:		
	conhdl	Connection handle
	callback	Callback for virtual UART event
Return:		
	RBLE_OK	Success
	RBLE_STATUS_ERROR	Failed due to rBLE mode is in RBLE_MODE_ACTIVE

(5) RBLE_VUART_Client_Disable

`RBLE_STATUS RBLE_VUART_Client_Disable(void)`		
This function disables client role of virtual UART function.		
Parameters:		
	-	-
Return:		
	RBLE_OK	Success

(6) RBLE_VUART_Client_Send_Chars

`RBLE_STATUS RBLE_VUART_Client_Send_Chars( const char *chars, uint16_t len)`		
This function send characters from the client to the server. The characters sent by the client are received by server. After the reception, the server responses with “Response” to the client. The response is informed to the client by RBLE_VUART_EVENT_CLIENT_WRITE_RSP event.		
Parameters:		
	chars	Received characters
	len	Received characters length
Return:		
	RBLE_OK	Success
	RBLE_STATUS_ERROR	Failed due to rBLE mode is in RBLE_MODE_ACTIVE

8.5.3 Event

This section describes the events defined by virtual UART function.

(1) RBLE_VUART_EVENT_SERVER_ENABLE_CMP

RBLE_VUART_EVENT_SERVER_ENABLE_CMP		
This event informs completion of server role enable.		
Parameters:		
	status	server role enable status

(2) RBLE_VUART_EVENT_SERVER_WRITE_REQ

RBLE_VUART_EVENT_SERVER_WRITE_REQ		
This event informs characters receive.		
Parameters:		
	status	The result of receiving characters
	value	Received characters
	len	Received characters length

(3) RBLE_VUART_EVENT_SERVER_INDICATION_CFM

RBLE_VUART_EVENT_SERVER_INDICATION_CFM		
This event informs server role send characters completion.		
Parameters:		
	status	The result of character send

(4) RBLE_VUART_EVENT_CLIENT_ENABLE_CMP

RBLE_VUART_EVENT_CLIENT_ENABLE_CMP		
This event informs client role enable completion.		
Parameters:		
	status	

(5) RBLE_VUART_EVENT_CLIENT_INDICATION

RBLE_VUART_EVENT_CLIENT_INDICATION		
This event informs receive characters.		
Parameters:		
	status	The result of character receive
	value	Received characters
	len	Received characters length

(6) RBLE_VUART_EVENT_CLIENT_WRITE_RSP

RBLE_VUART_EVENT_CLIENT_WRITE_RSP		
This event informs send character completion.		
Parameters:		
	status	The result of characters sending

8.6 Application State Change

Figure 8-1 shows the application state transition diagram. The application changes the state depends on connection and disconnection event and simple AT command execution.

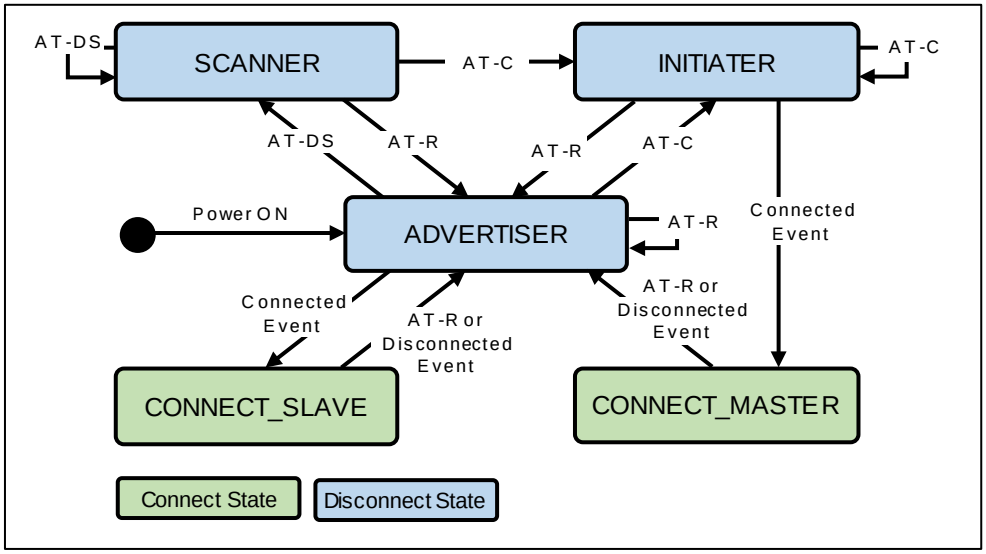


Figure 8-1: Application state diagram

Table 8-5 shows the application state list.

Table 8-5: Application state list

Application State	Description
ADVERTISER	The application is advertising.
SCANNER	The application is scanning neighbor devices by executing AT-DS command. After AT-DS has finished, the application remains in this state.
INITIATER	The application create a connection to a remote device by executing AT-C.
CONNECT_MASTER	BLE connection is established as master role. CONNECT_MASTER is GATT client.
CONNECT_SLAVE	BLE connection is established as slave role. CONNECT_SLAVE is GATT sever.

8.7 Application Detailed Sequence

This section shows the sequence of boot, connection, character transfer and disconnection. Refer to Bluetooth® Low Energy Protocol Stack API Reference: Basics [R01UW0088E].

8.7.1 Boot Sequence

Figure 8-2 shows the boot sequence.

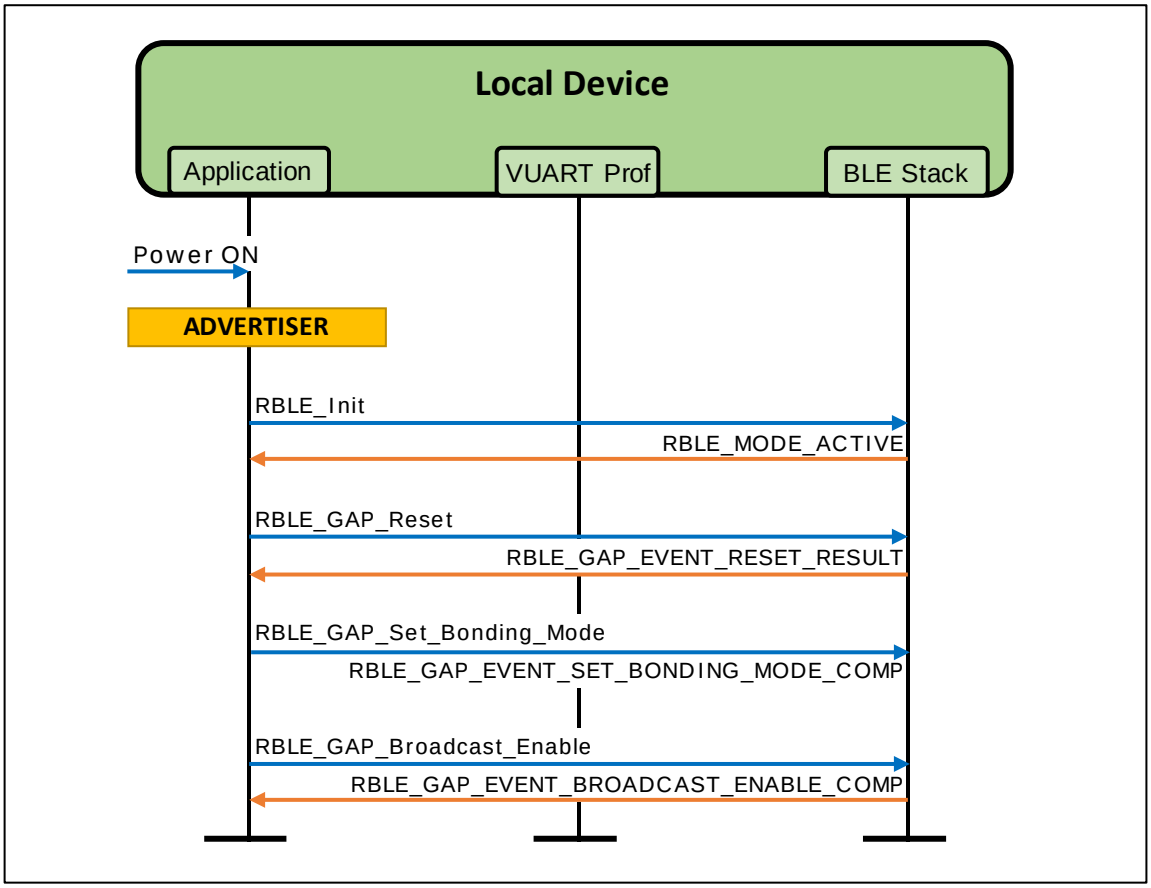


Figure 8-2: Boot Sequence

8.7.2 Connection Sequence

Figure 8-3 shows the connection sequence.

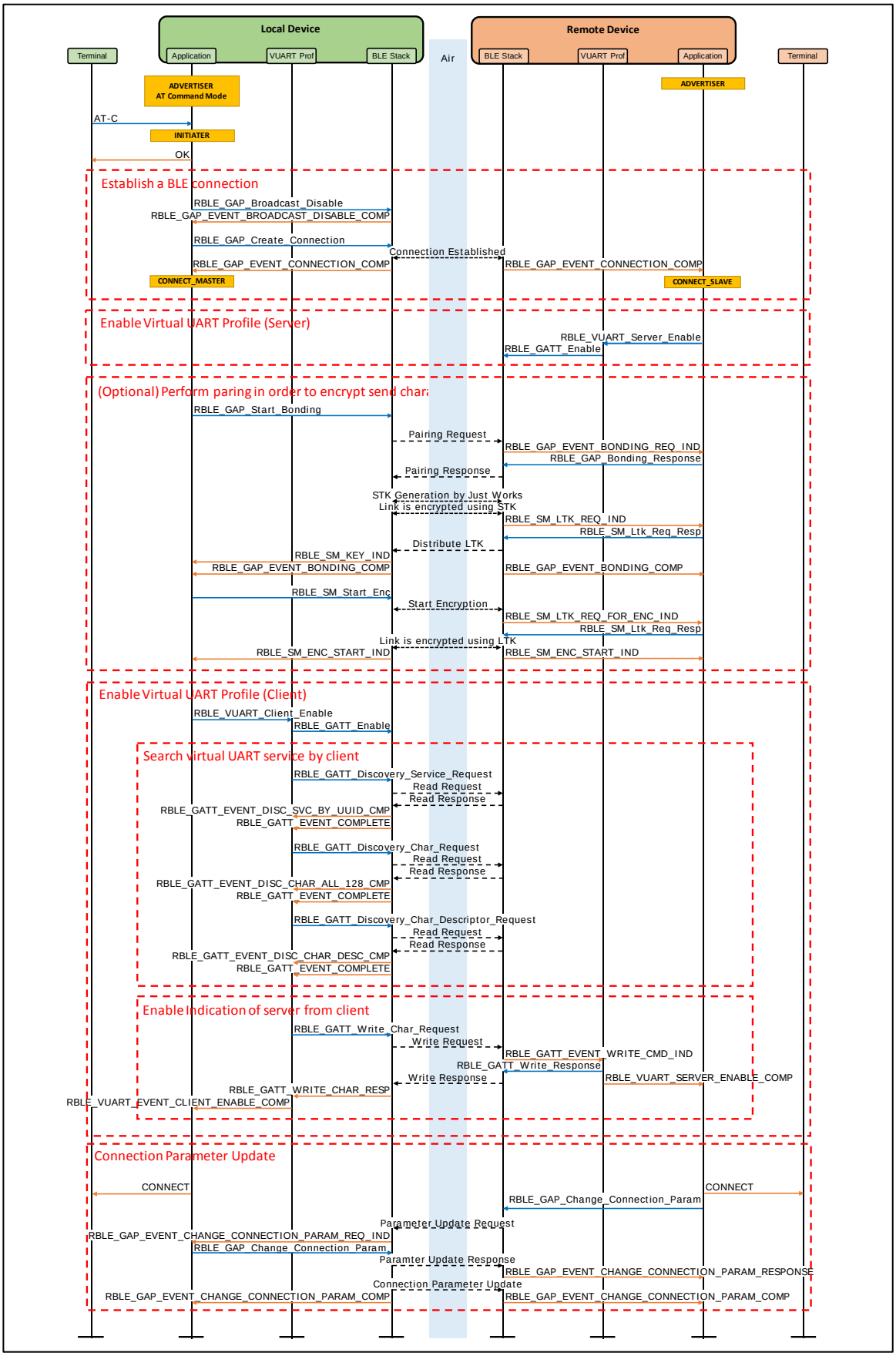


Figure 8-3: Connection sequence

8.7.3 Character Transfer Sequence

Figure 8-4 shows the character transfer sequence.

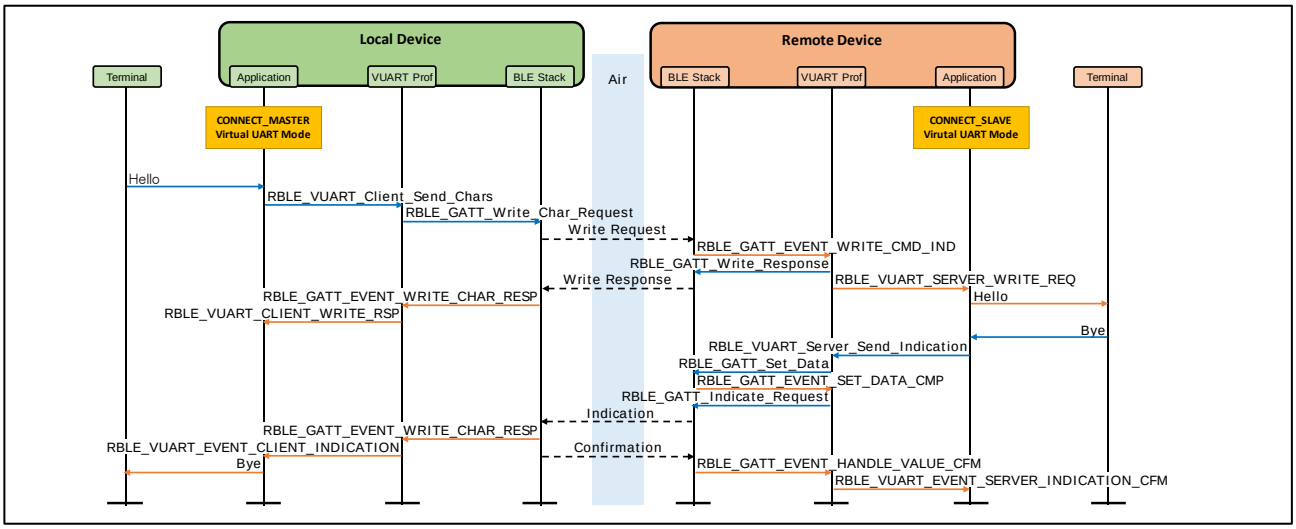


Figure 8-4: Character receive sequence

8.7.4 Disconnection Sequence

Figure 8-5 shows the disconnection sequence.

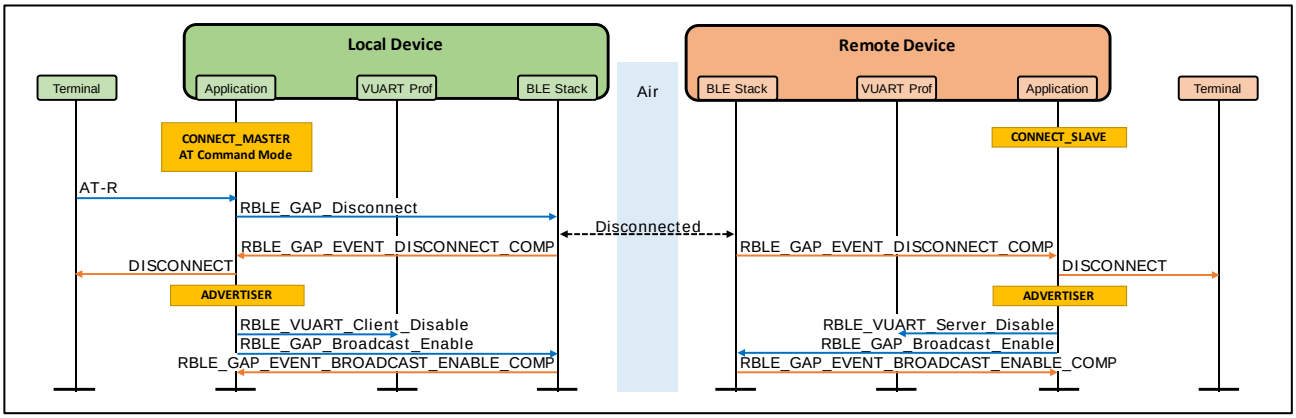


Figure 8-5: Disconnection sequence

8.8 Others

8.8.1 Caution when implementing the program to connect to the application

As described in section 8.7.2, when AT-C command is executed, the following processing are executed in order. After these steps have finished successfully the application responses with “CONNECT” message on both of the devices. If you implement the program connect to the application also follows these steps. The pairing is optional.

- Establish a BLE connection
- Perform paring in order to encrypt send characters
- Search virtual UART service by client
- Enable Indication of server from client

8.8.2 Macro Settings

Table 8-6 shows macro settings.

Table 8-6 Macro Settings

マクロ名	説明
DISABLE_LOCAL_ECHO_BY_DEFAULT	When this macro is declared, the default settings of local echo is disabled. This macro should be set in the project file from development environment setting panel.

9. Appendix

9.1 ROM size, RAM size

Table 9-1 shows the ROM size and the RAM size of the “BLE Virtual UART Application” and “Console Driver” in Figure 2-1.

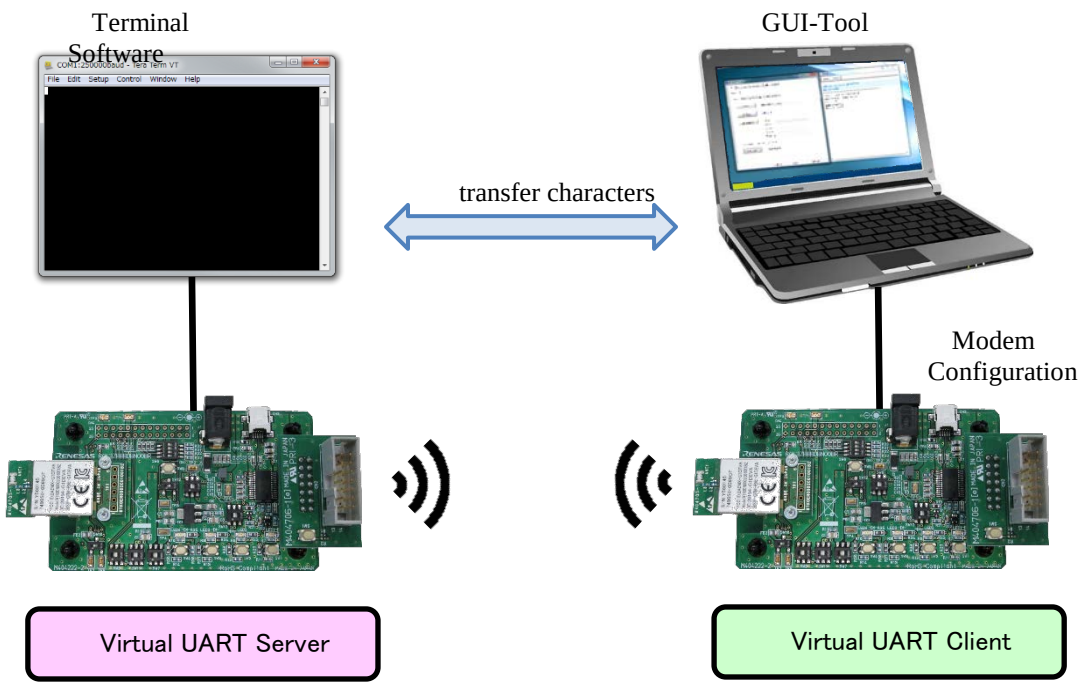
Table 9-1: ROM size, RAM size

Compiler	ROM size	RAM size
RL78 Family C Compiler Package V1 V1.03.00	5392	613
Renesas CA78K0R V1.72	7366	610
IAR Embedded Workbench for Renesas RL78 V2.20.1	5157	612

9.2 Operational Check by Using the GUI-Tool

This section describes the operation check procedure of Virtual UART by using the GUI-Tool [R01AN2469].

Figure 9-1 shows overview diagram of operational check by using the GUI-Tool. This application operates as a Virtual UART server, and the GUI-Tool operates as a Virtual UART client. It is possible to transfer characters to each other.



Note: GUI-Tool Version 1.12 (R01AN2469XX0112) or later is required in order to check the operation.

Figure 9-1: Operation check by using the GUI-Tool

Hereafter, the combination of evaluation board (Virtual UART application was written) and terminal software is mentioned as “Virtual UART Server”. And the combination of evaluation board (Modem Configuration Hex file was written) and the GUI-Tool is mentioned as “Virtual UART Client”.

9.2.1 Preparation

- Virtual UART Server

  In accordance with the procedures described in the following section, write the firmware onto RL78/G1D evaluation board and then launch a terminal software on PC.
  - 7.2 Build Procedure
  - 7.3 Preparation for Execution Environment

- Virtual UART Client

  Write a Modem configuration Hex file (any of the build environment) that is included in the package of BLE protocol stack onto RL78/G1D evaluation board, then launch the GUI-Tool.

  Notes: 1. In order to access the service on the Virtual UART Server by using the GATT APIs, it does not matter the profile type of Hex file to be written.
  2. Refer to Bluetooth Low Energy Protocol Stack GUI Tool [R01AN2469] “6. Utilization” about how to launch the GUI-Tool.

9.2.2 Operation

It is possible to transfer characters by operating Virtual UART Server and Virtual UART Client in the following procedure.

1. Discoverable Mode (Virtual UART Server)
   Push the RESET button (SW5) on RL78/G1D evaluation board that operates as Virtual UART Server.
   After the reset, the virtual UART application transitions to discoverable mode automatically, and then start the Advertising.

2. Device Discovery (Virtual UART Client)
   Search discoverable mode devices by operating the GUI-Tool.
   (1) Activate [Scanning] tab of [GAP] tab.
   (2) Select “General Discovery” in the Discovery group.
   (3) Press [Discover] button.

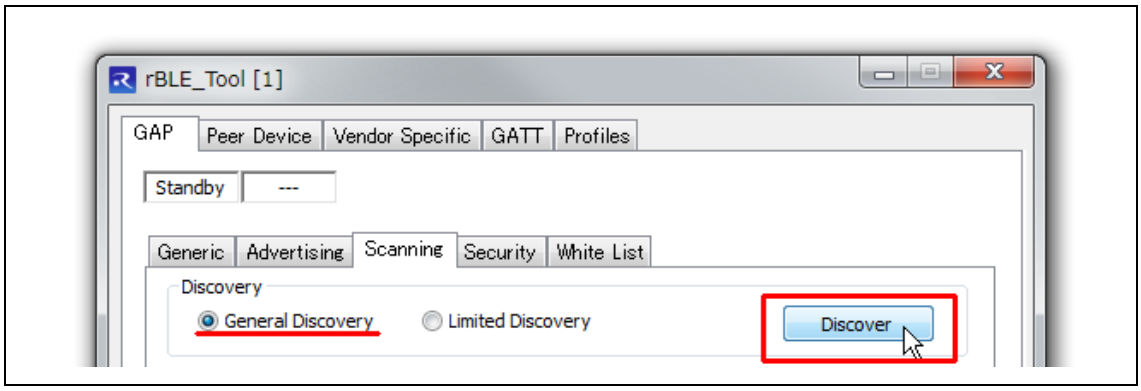


Figure 9-2: Device Discovery

(4) Discoverable mode devices will display in the list of Received Advertising data.

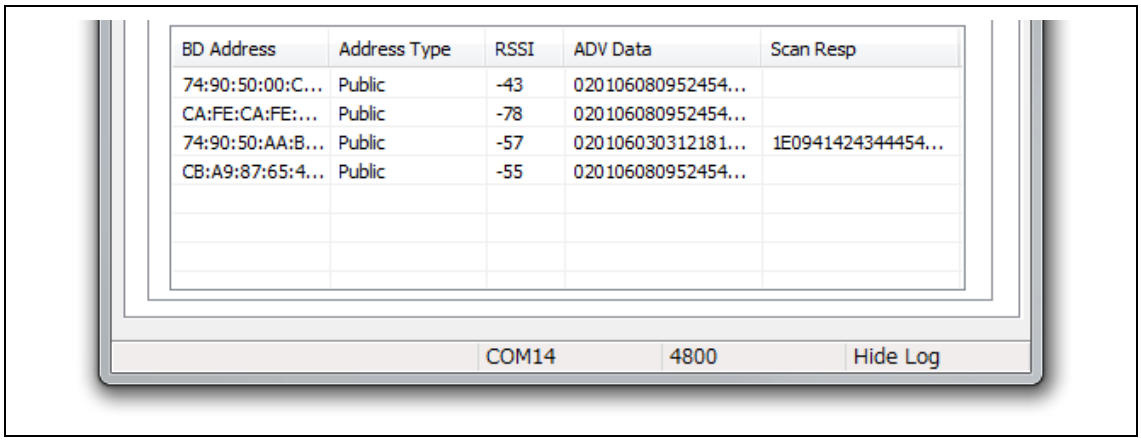


Figure 9-3: Result of Device Discovery

3. Connection (Virtual UART Client)
   Initiate connection to Virtual UART Server by operating the GUI-Tool.
   (1) In [Scanning] tab, Double-click onto the target device in the list of Received Advertising data.

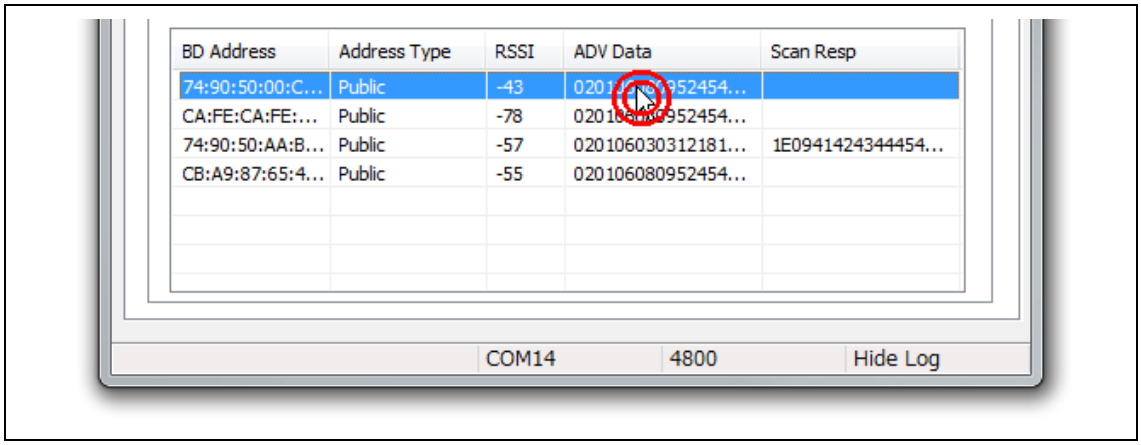


Figure 9-4: Select Device

Tips: It will be displayed the Advertising data analysis dialog by [Ctrl] key + double-clicking arbitrary row in the list of Received Advertising data.
The device which operates as Virtual UART Server contains “Renesas Virtual UART Service” to the <<Complete List of 128-bit Service UUIDs>>

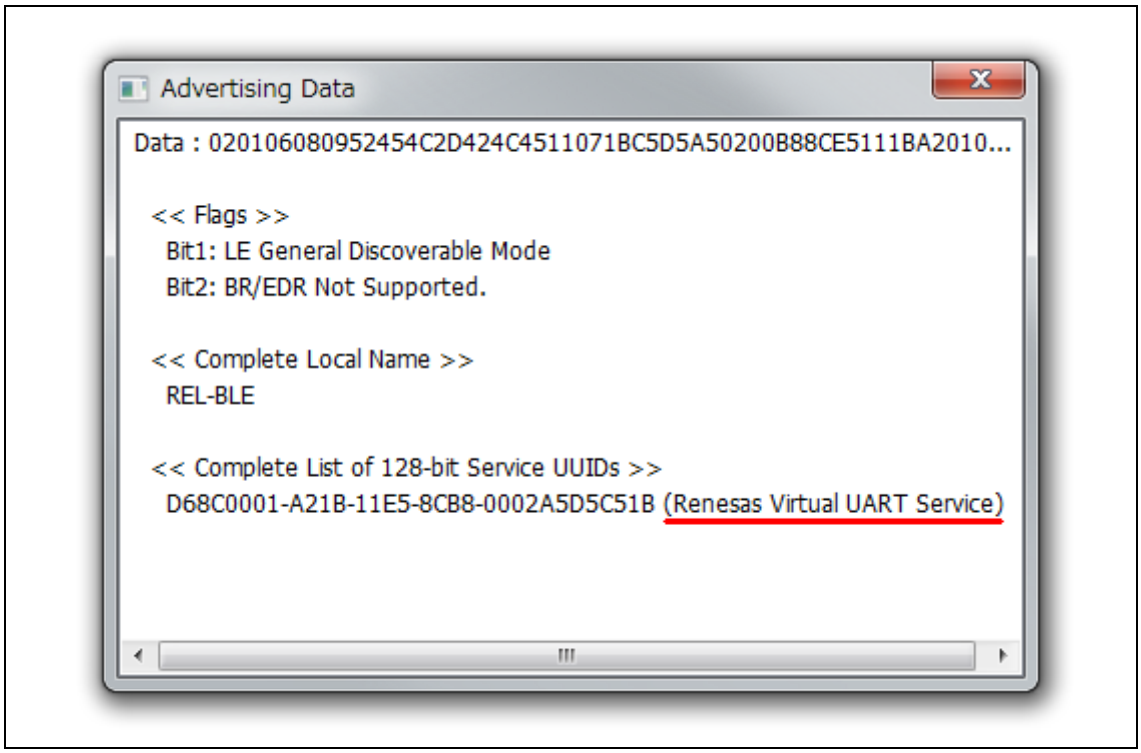


Figure 9-5: Advertising Data Analysis Dialog

(2) Activate [Connection] tab of [Peer Device] tab.
At this time, make sure that the target device address will reflect to “Peer Addr” field in top of [Peer Device] tab.

(3) Initiate a connection to Virtual UART Server by pressing [Connect] button.

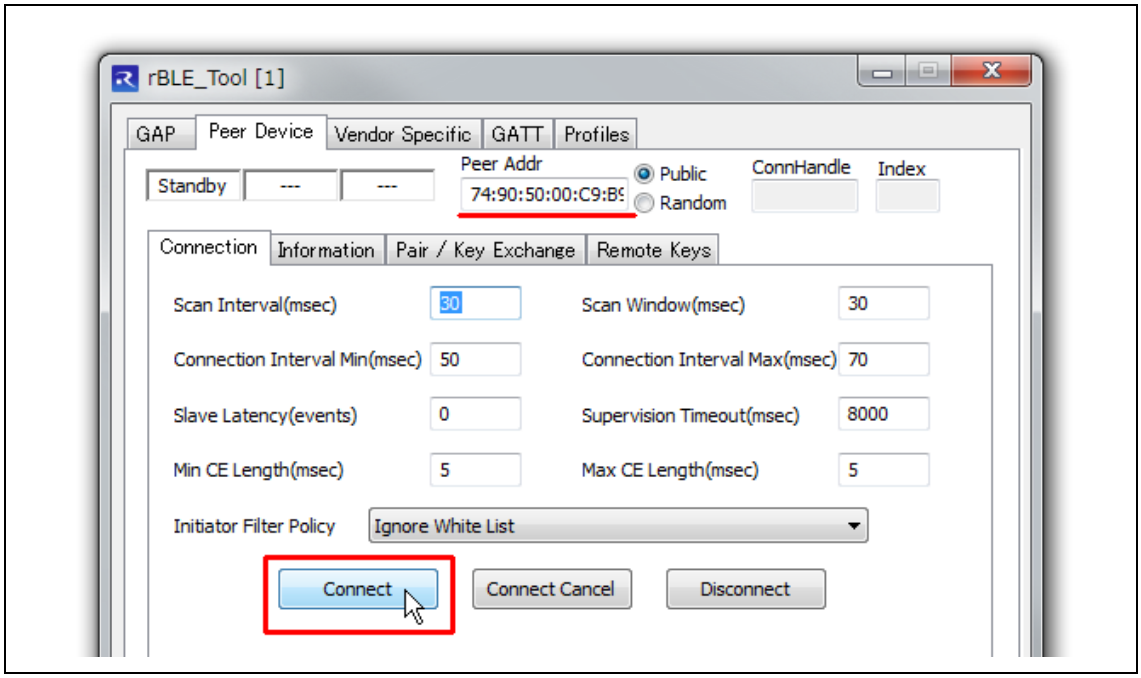


Figure 9-6: Initiate Connection

(4) When a connection is established, the State display in top of [Peer Device] tab is changed to “Connected”.

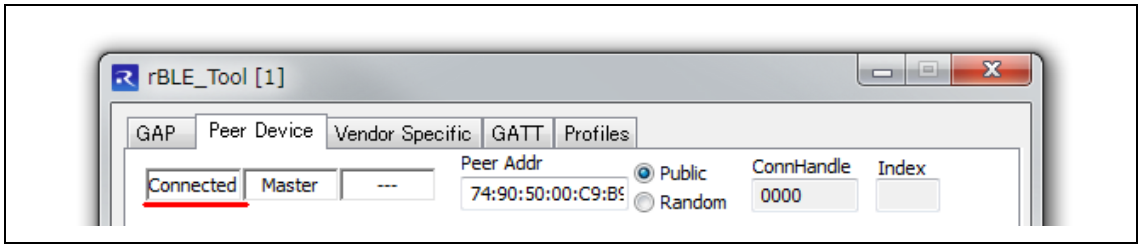


Figure 9-7: Established Connection

4. Service Discovery (Virtual UART Client)
   Discover services and characteristics on Virtual UART Server by operating the GUI-Tool.
   - Service Discovery
     Discover all services on Virtual UART Server.
     (1) Activate [Service Discovery] tab of [GATT]→[Client] tab.
     (2) Select “Discover All Primary Services” in the Discovery Type drop-down list, and press [Discover] button.

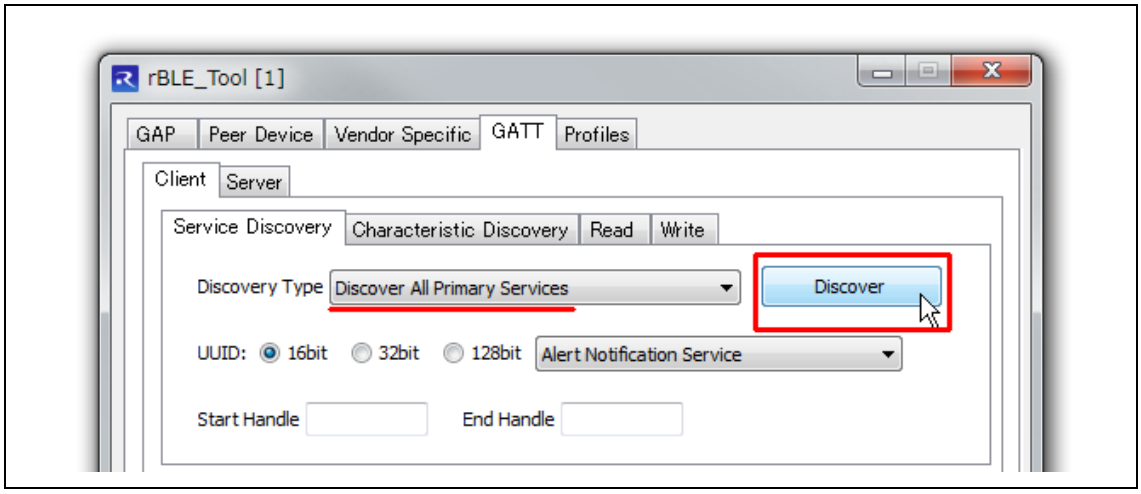


Figure 9-8: Service Discovery

(3) Acquired service information is displayed in the list of “Remote GATT Database”.

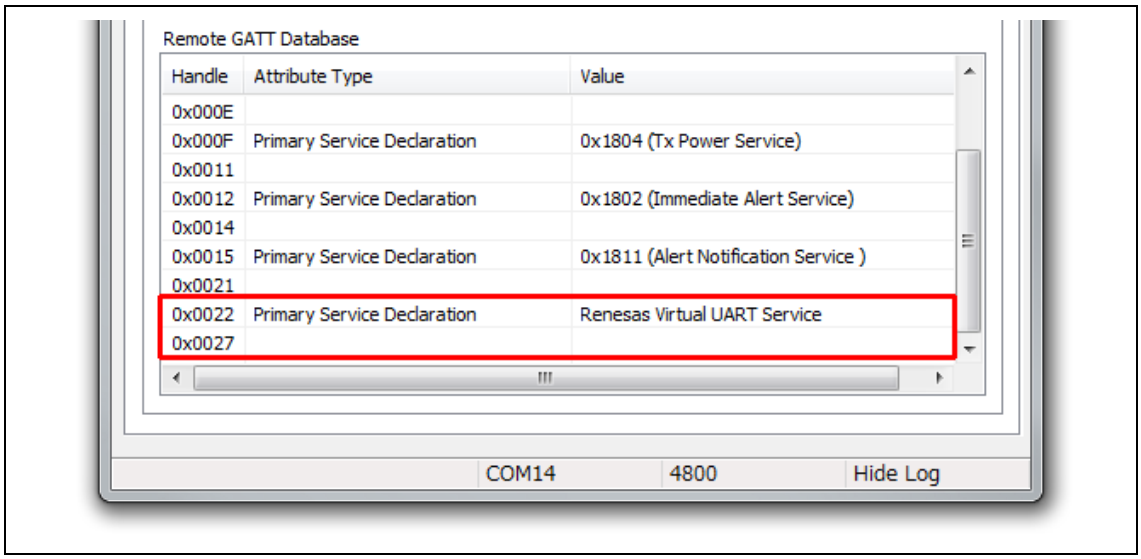


Figure 9-9: Result of Service Discovery

- Characteristic Discovery

Discover all service characteristics on Virtual UART Server.

(1) Activate [Characteristic Discovery] tab of [GATT]→[Client] tab.

(2) Select “Discover Characteristics of a Service” in the Discovery Type drop-down list, and press [Discover] button.

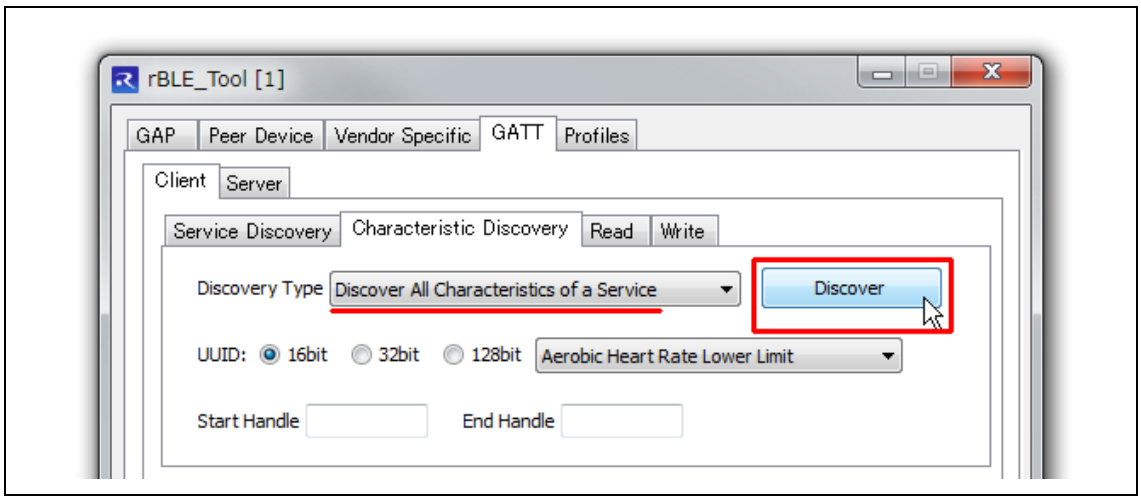


Figure 9-10: Characteristic Discovery

(3) Acquired characteristic information is displayed in the list of “Remote GATT Database”.

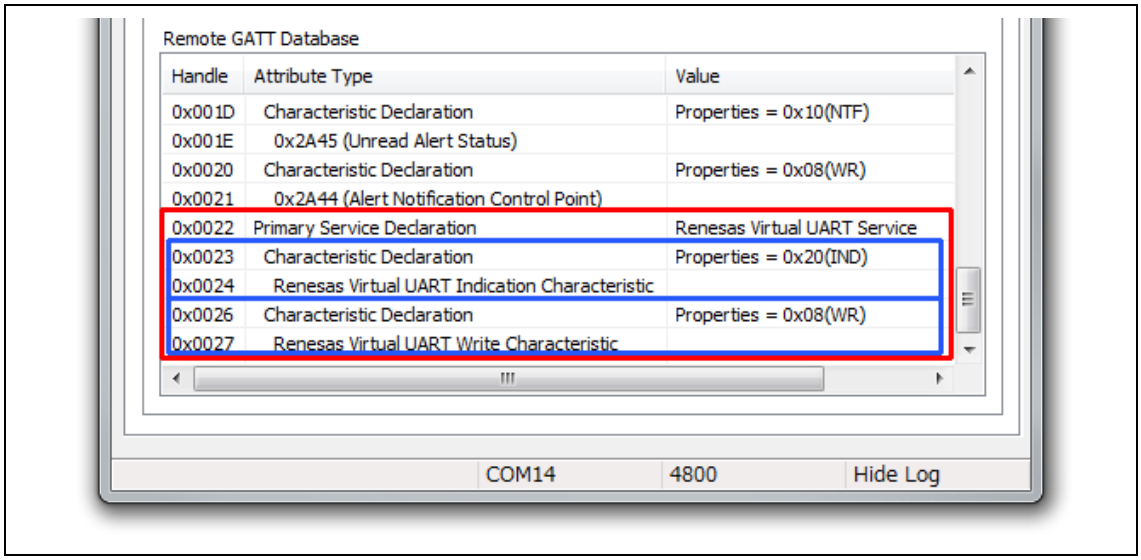


Figure 9-11: Result of Characteristic Discovery

- Characteristic Descriptor Discovery

Discover characteristic descriptors of a characteristic on Virtual UART Server.

(1) Activate [Characteristic Discovery] tab of [GATT]→[Client] tab.

(2) Select “Discover All Characteristic Descriptors” in the Discovery Type drop-down list, and press [Discover] button.

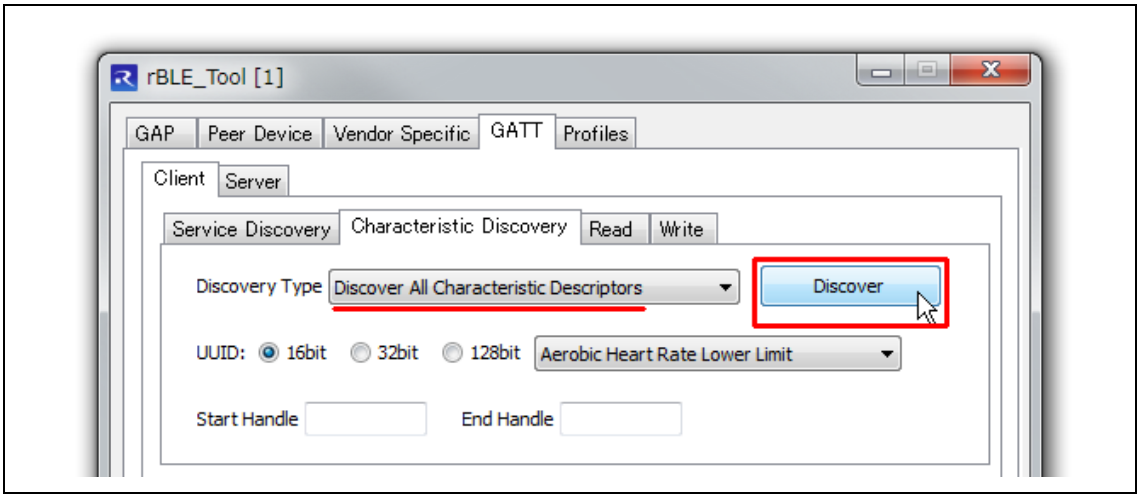


Figure 9-12: Characteristic Descriptor Discovery

(3) Acquired characteristic descriptor information is displayed in the list of “Remote GATT Database”.

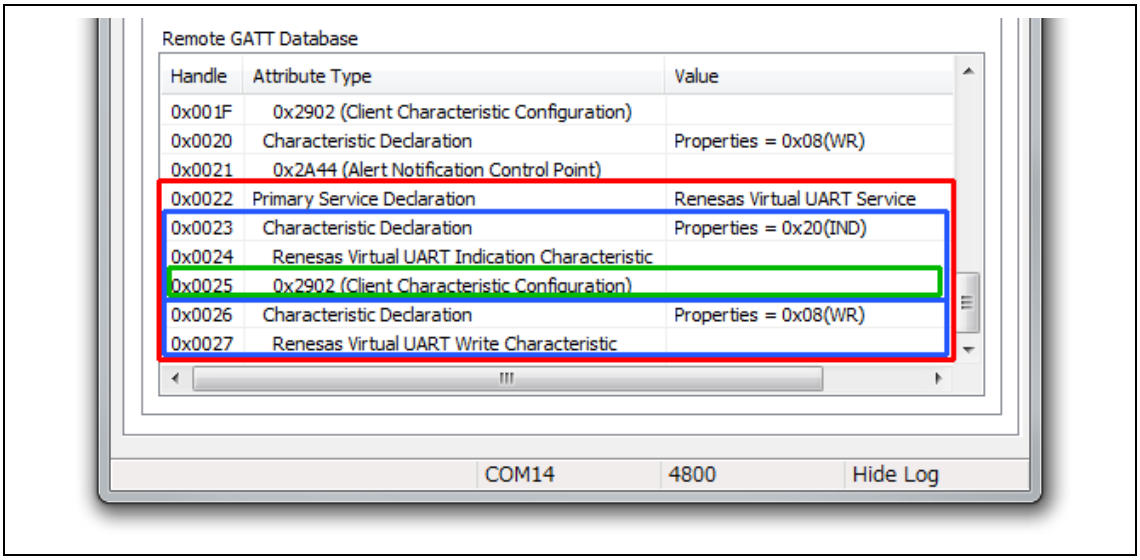


Figure 9-13: Result of Characteristic Descriptor Discovery

5. Enable Indication (Virtual UART Client)
   Enable character transfer from Virtual UART Server to Virtual UART Client (Indication) by operating the GUI-Tool.
   (1) Activate [Write] tab of [GATT]→[Client] tab.
   (2) Select “Write Characteristic Descriptors” in the Write Type drop-down list.
   (3) In the list of “Remote GATT Database”, double-click the Client Characteristic Configuration Descriptor within “Renesas Virtual UART Indication Characteristic”.
   By double-clicking, the handle value of Client Characteristic Configuration Descriptor will reflect to “Handle” field in [Write] tab.
   (4) Enter the value of “0002” (it means that “Indications enabled”) in “Write Data” field.
   (5) By pressing [Write] button, write the characteristic descriptor value to Virtual UART Server.

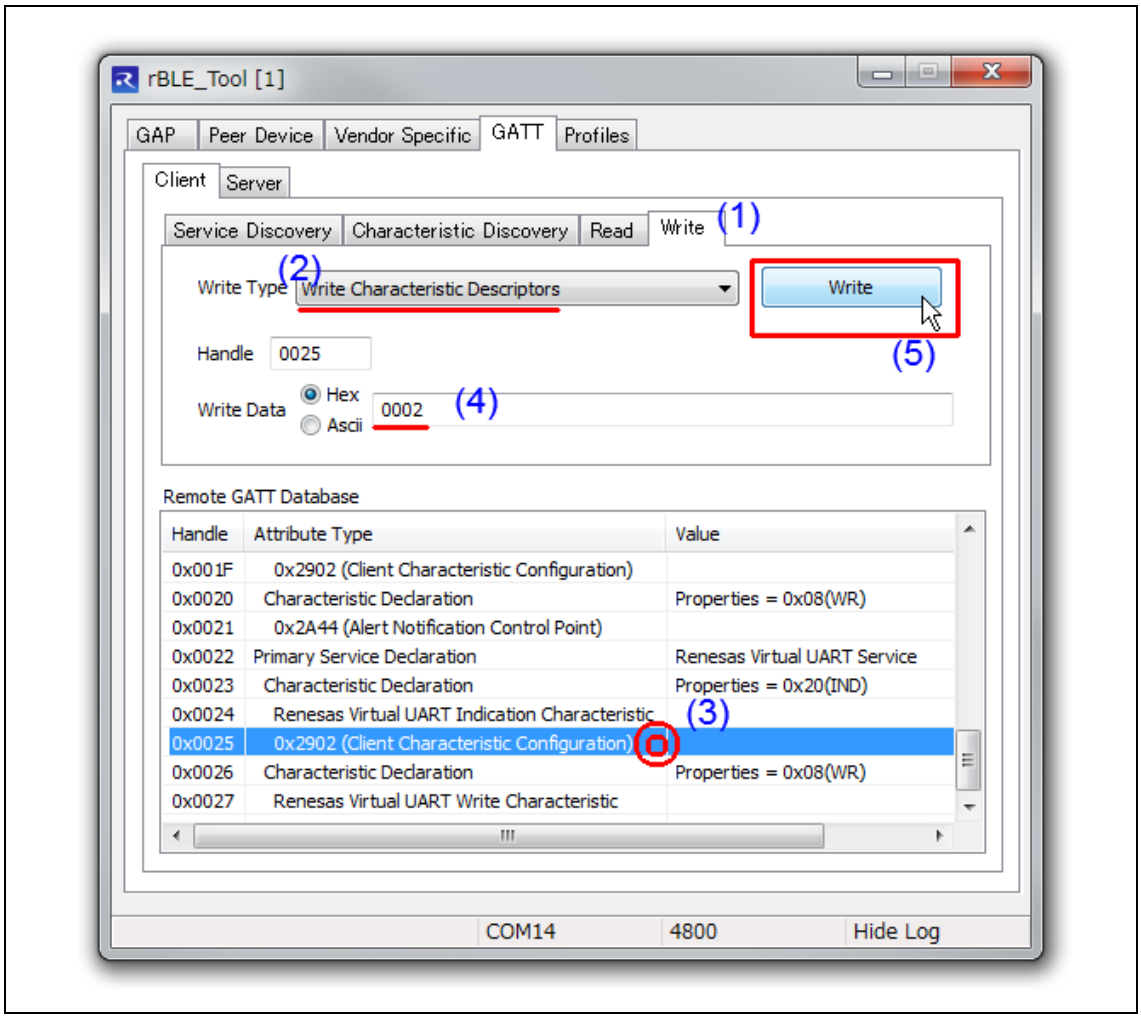


Figure 9-14: Enable Indication

(6) Console window is displayed when the response is received from Virtual UART Server.

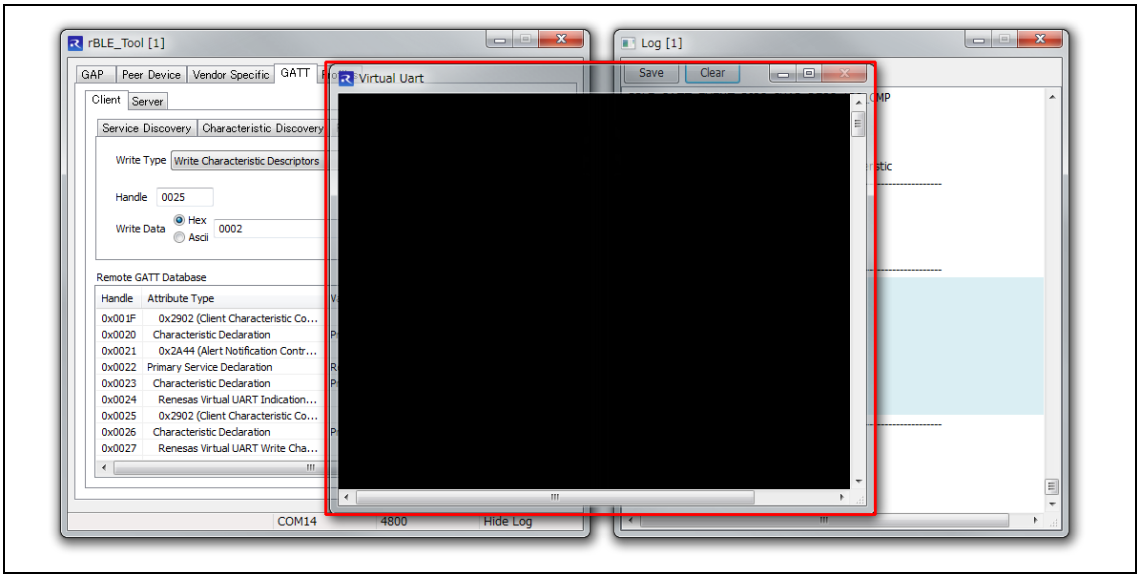


Figure 9-15: Console Window

6. Character Transfer (Virtual UART Server / Virtual UART Client)

- Character transfer from Virtual UART Server

(1) Input ESC key on the terminal software in order to switch the application mode to Virtual UART mode.
(2) Type arbitrary characters (e.g. “Hello!”) on the terminal software.
(3) Input characters are displayed by yellow characters in the console window of Virtual UART Client.

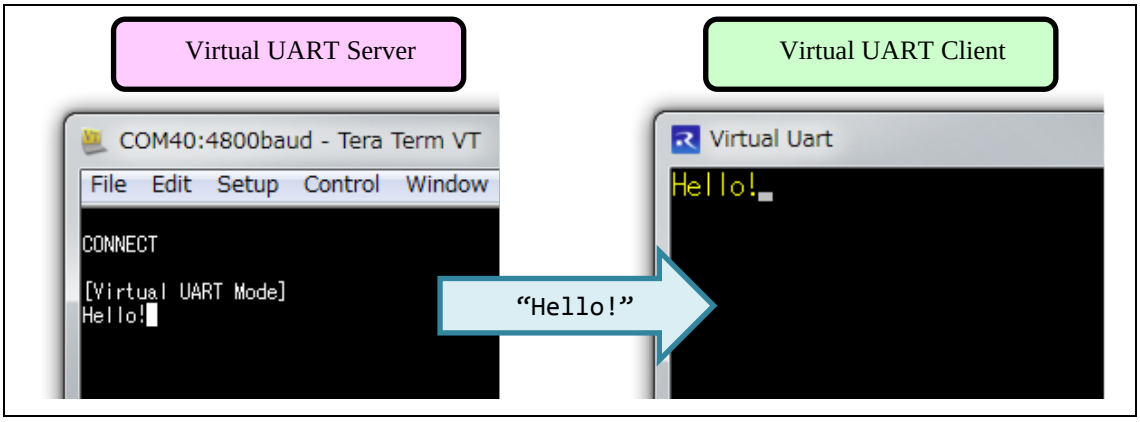


Figure 9-16: Character Transfer (Server→Client)

- Character transfer from Virtual UART Client

(1) Type arbitrary characters (e.g. “Bye”) on the console window.
(2) Input characters are displayed in the terminal software of Virtual UART Server.

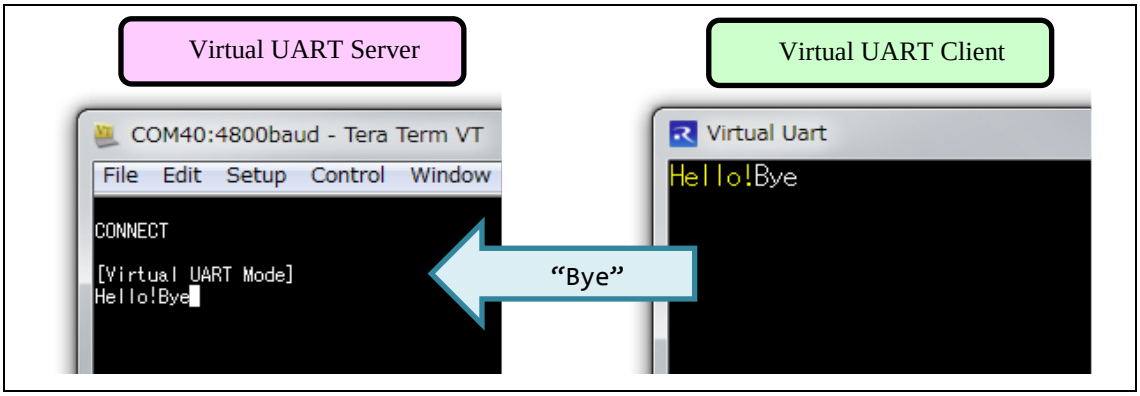


Figure 9-17: Character Transfer (Client→Server)

7. Disconnection (Virtual UART Server / Virtual UART Client)
   - Disconnect from Virtual UART Server
     (1) Input ESC key on the terminal software in order to switch the application mode to Simple AT command mode.
     (2) Execute “AT-R” on the terminal software (Disconnect the established connection).
     (3) When the connection is terminated, it will be displayed “DISCONNECT” on the terminal software.

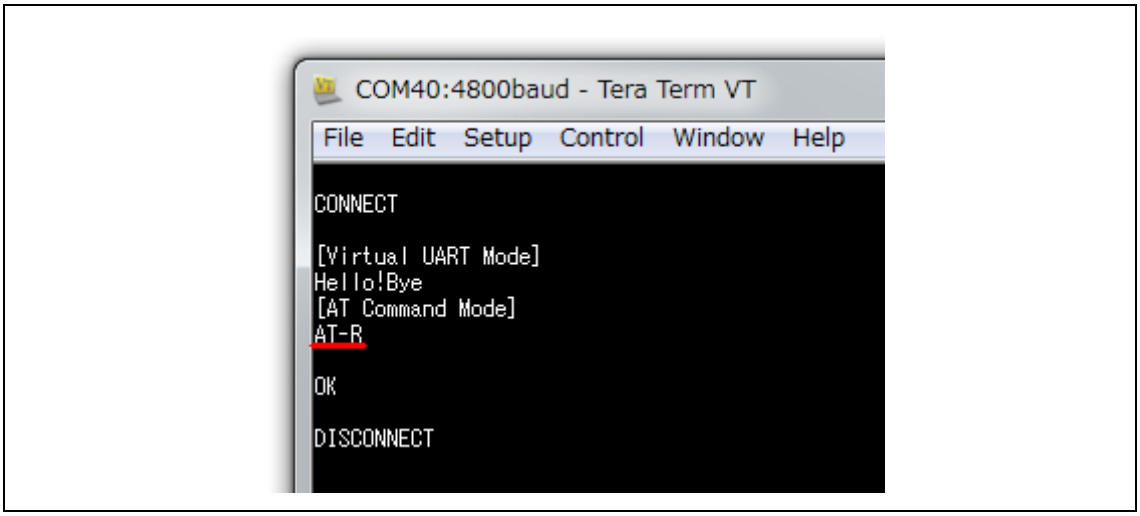


Figure 9-18: Disconnect from Virtual UART Server

   - Disconnect from Virtual UART Client
     (1) Activate [Connection] tab of [Peer Device] tab.
     (2) Disconnect the established connection by pressing [Disconnect] button.
     (3) When the connection is terminated, the State display in top of [Peer Device] tab is changed to “Standby”.

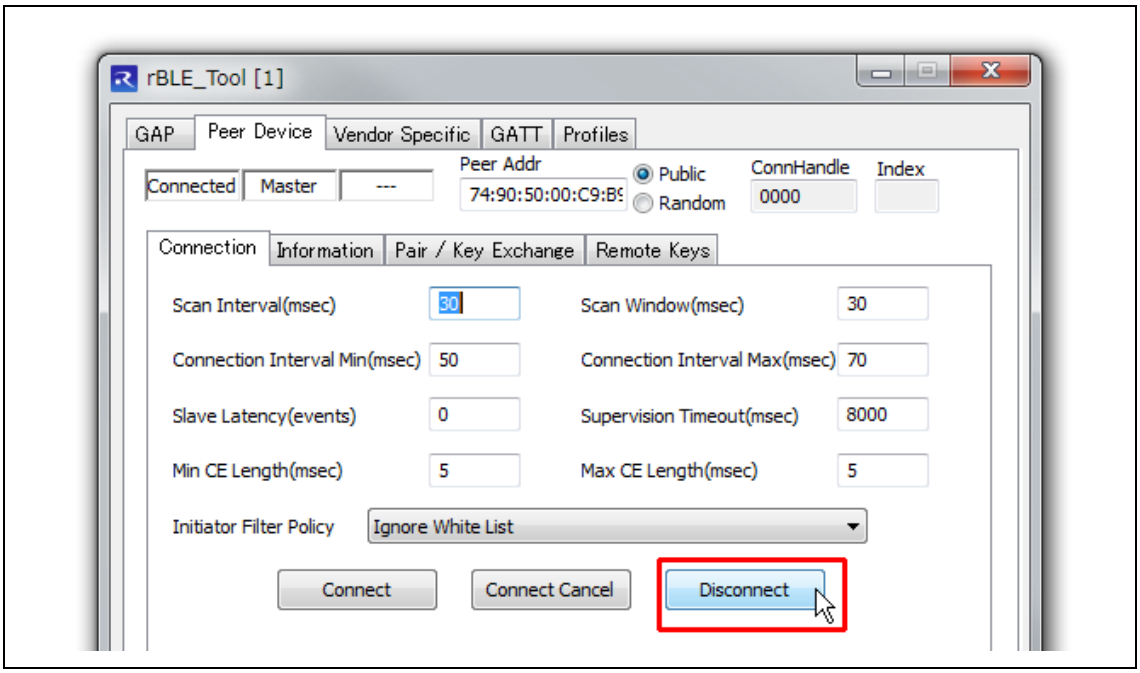


Figure 9-19: Disconnect from Virtual UART Client

Website and Support

Renesas Electronics Website
http://www.renesas.com/

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

Revision Record

Rev.	Date	Description	
		Page	Summary
1.00	Feb 24, 2016	—	Initial issue
1.10	Oct 7, 2016	5	2.2 File Composition : Add the file composition for development environments and firmware.
		7	4 Simple AT Command Mode : Add commands AT-CI=<con_intv>, AT-CI?, ATE0, ATE1.
		16	7 Operational Check : Add Environment setup and build procedure descriptions for development environments.
		19	7.4 Usage Example : Add the procedure to set a device address.
		23	8.3 Connection : Change Connection Interval Default setting value.
		32	8.7.2 Connection Sequence : Add the procedure for Connection Interval change.
		34	8.8.2 Macro Settings : Newly added.
		36	9.2 Operational Check by Using the GUI-Tool : Newly added.
1.11	Nov 22, 2016	-	Revision change associated with source code bug fix (No document update).

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.

   ¾ 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.

   ¾ 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.

   ¾ 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.

   ¾ 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.

   ¾ 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 of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
3. Renesas Electronics does not assume any liability for infringement of 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. 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 should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from such alteration, modification, copy or otherwise misappropriation of Renesas Electronics product.
5. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The recommended 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 systems; anti-disaster systems; anti-crime systems; and safety 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 (nuclear reactor control systems, military equipment etc.). You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application for which it is not intended. Renesas Electronics shall not be in any way liable 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. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages 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 its 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 be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, 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, 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 use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technology may 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 should not use Renesas Electronics products or technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. When exporting the Renesas Electronics products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, who distributes, disposes of, or otherwise places the product with a third party, to notify such third party in advance of the contents and conditions set forth in this document, Renesas Electronics assumes no responsibility for any losses incurred by you or third parties as a result of unauthorized use of Renesas Electronics products.
11. This document may not be 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, or if you have any other inquiries.

(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.

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 5.0