RZ/N2L Group

BACnet to OPC UA Gateway Sample Software

R01AN6963EJ0100
Rev.1.00
Jul.31.2023

Introduction

This document describes sample software for running Gateway, which converts the BACnet communication protocol for Building Automation (BA) to OPC UA, on the RZ/N2L.

Target Device

RZ/N2L Group

Contents

List of Abbreviations and Acronyms

In this document, the terms below are defined as follows:

Terms	Description
FSP	Flexible Software Package
RSK	Renesas Starter Kit
BA	Building Automation
BACnet	Building Automation and Control Networking
B-SS	BACnet Smart Sensor
ASHRAE	American Society of Heating, Refrigerating and Air-Conditioning Engineers
ANSI	American National Standards Institute
BIBB	BACnet Interoperability Building Blocks
API	Application Program Interface
APDU	Application Layer Protocol Data Unit
OPC UA	Open Platform Communications Unified Architecture
XML	Extensible Markup Language

Related documents

Document Type	Document Title	Document No.
Data Sheet	RZ/N2L Group Datasheet	R01DS0397EJ****
User's Manual	RZ/N2L Group User's Manual: Hardware	R01UH0955EJ****
User's Manual	Renesas Starter Kit+ for RZ/N2L User's Manual	R20UT4984EG****
Application Note	RZ/N2L Group TCP/IP lwIP Sample Program Package	R01AN6588EJ****
Application Note	RZ/N2L BACnet Sample Software	R01AN6789EJ****

1. Overview

1.1 Abstract

OPC UA, which enables interoperability of industrial applications, is becoming widely used not only in factory automation (FA) but also in various industries. OPC 30030, a companion spec to OPC UA and BACnet, the major communication protocol for Building Automation (BA), has been developed, and interoperability across industries has been started.

This document describes the sample software configuration and its usage to realize Gateway between OPC UA and BACnet on RZ/N2L, which is the RZ processor for industrial networks.

Fig. 1-1 RSK+ for RZ/N2L

The sample software described in this document is for the Gateway, an interface device that connects different protocol devices.

Therefore, as shown in Fig. 1-2, the operation test in this document uses an application on a PC as the OPC Client, and RZ/N2L BACnet Sample Software (r01an6789xx0101-rzn2l-bacnet) as the BACnet server on the other end of the network. For the convenience of explanation, the Gateway described in this document is referred to as B-GW, and the BACnet server on the other side is referred to as B-SS.

Also, the sample software includes code to generate BACnet pseudo data so that it can be evaluated with a single RZ/N2L RSK board.

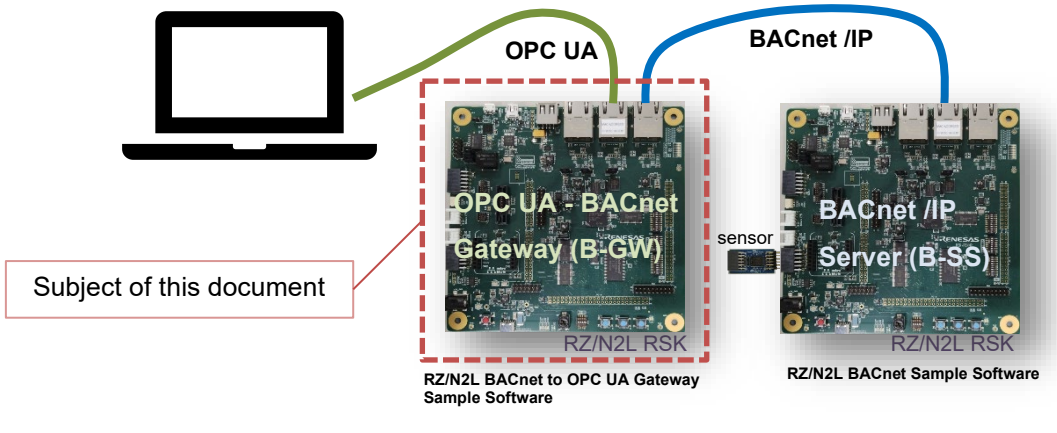


Fig. 1-2 Subject of this document and test setup

1.2 Operating Environment

1.2.1 Software Environment

The operating environment of this sample software is shown in Table 1-1

Table 1-1 Operating Environment

Category	Name	Version	Link	備考
RZ/N2L BACnet sample software	Sample Package			
IDE	e2studio	22.10.0	https://www.renesas.com/document/sws/e-studio-and-rzn2l-fsp-installer	Included with e2studio installer
Flexible Software Package	FSP	1.1.0		Included with e2studio installer
GNU Arm Embedded Toolchain	GCC Toolchain	V9.3.1.20200408 (*1)		Included with e2studio installer
OPC UA Client Tool	UaExpert	1.6.3	OPC UA Clients - Unified Automation (unified-automation.com)	
Packet analyzer	Wireshark	4.0.7	Wireshark · Download	

(*1). V10.3.1.20210824 is also installed when installing e2studio, but we recommend using V9.3.1.20200408.

1.2.2 Hardware Environment

This sample software is tested under the hardware environment of Table 1-2.

Table 1-2 Hardware Environment

Name	Type Name	Maker	Link	Note
Renesas Starter Kit+ for RZ/N2L	RTK9RZN2L0S00000BE	Renesas Electronics	www.renesas.com/rskrzn2l	RSK Board 2pcs
Air Velocity Sensor Pmod™ Board	US082-FS3000EVZ	Renesas Electronics	US082-FS3000EVZ - Air Velocity Sensor Pmod™ Board (Renesas Quick-Connect IoT) | Renesas	Renesas Quick Connect IoT

2. Hardware configuration

This section describes the hardware configuration of executing the sample software.

2.1 RSK Board Settings

When executing the sample software, configure the RSK board settings in Fig. 2-1

- The boot mode is NOR Flash ROM boot mode.
- Use SD-RAM as external memory
- Ethernet port 2 (ETH2) is not available on the RSK board with the settings.

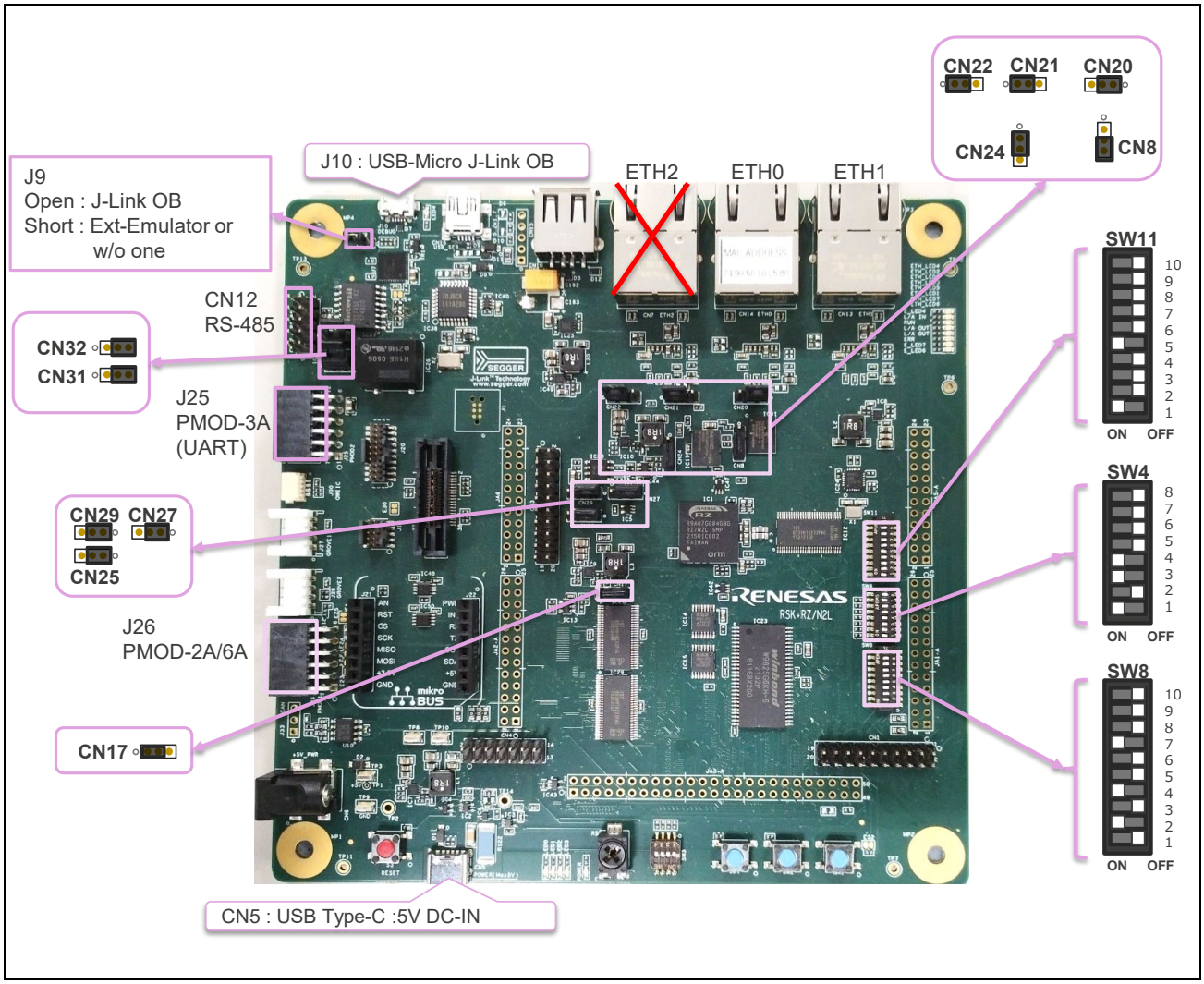


Fig. 2-1 Board Configuration

Each switch and jumper setting is shown in Table 2-1 and Table 2-2. The red text indicates differences from the RSK board settings for the BACnet sample software (r01an6789xx0101-rzn2l-bacnet).

Table 2-1 DIPSW Settings

DIPSW		Setting	Description
SW11	1	ON	Enable LED_RED2 signal
	2	OFF	

	3	OFF	
	4	OFF	Enable RS485_RX signal
	5	ON	
	6	OFF	Disable P21_5、M2_VP、CAN_RX、ADTRG、P01_7
	7	OFF	
	8	OFF	
	9	OFF	
	10	OFF	
SW4	1	ON	16bit Bus boot mode (NOR Flash ROM Boot)
	2	OFF	
	3	ON	
	4	ON	JTAG Authentication by Hash is disabled
	5	OFF	-
	6	OFF	Enables signals other the trace. (Motor, RS485, etc.) (TRACE_OPTION_SEL=H)
	7	OFF	Enables external bus. (BSC_OPTION_SW=H)
	8	OFF	Enable SW3 (general purpose DIPSW)
SW8	1	OFF	Enable LED_GREEN
	2	ON	
	3	OFF	
	4	ON	Enable LED5
	5	OFF	
	6	OFF	Enable RS485_DE
	7	ON	
	8	OFF	Disable P02_2, IRQ4, CAN_TX
	9	OFF	
	10	OFF	

Table 2-2 Jumper Settings

Jumper	Setting	Description
J9	open	When using the J-Link® OB
	short	When using the external emulator or not using the emulator
CN31	2-3short	RS485 Half Duplex
CN32	2-3short	RS485 Half Duplex
CN20	1-2short	When using 3 ports in the same PHY mode
CN21	1-2short	When using 3 ports in the same PHY mode
CN22	1-2short	When using 3 ports in the same PHY mode
CN24	1-2short	Connect 3.3V Power rail to VCC1833_3. (Using External Bus)
CN8	2-3short	Select QSPI Serial Flash (QSPI_CS)
CN29	1-2short	USB Serial (UART_USB_RX)
CN27	1-2short	HyperRAM (IC41)
CN25	1-2short	Other than the SHOST interface. (Trace, SPI, external bus)
CN17	1-2short	Use 3.3V for VCC1833_2 (disable ETH2)

2.2 Pmod™ Connection

In the sample software, Renesas Quick Connect IOT air velocity sensor (US082-FS3000EVZ) is connected to J26 of RZ/N2L RSK board and input a sensor signal for the B-SS (BACnet Smart Sensor).

To support Pmod™ Type 6A (Extended I2C), connect E2 and E3 of the RSK board (solder bridge) and cut E23 and E24 (trace cut) as shown in Fig.2-2

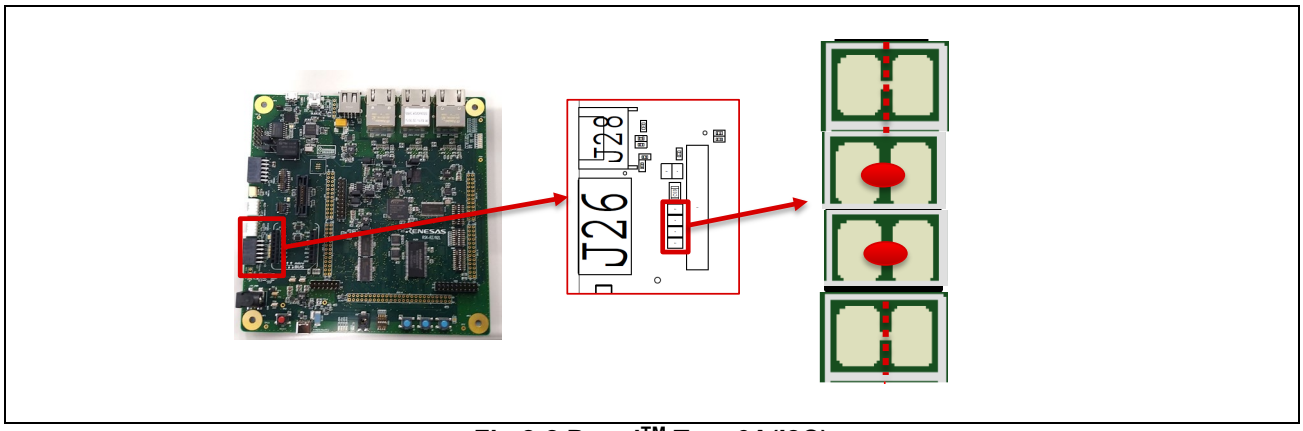


Fig.2-2 Pmod™ Type6A(I2C)

3. Sample Software

This chapter describes the structure and usage of the sample software.

3.1 Folder structure

The folder structure of the sample software is shown below. As a guide, the bolded text indicates folders containing files that users will customize with this sample software.

RZN2L_OPC_BGW_V1.0.0

```
|
├──BACNETOSS                BACnet Open Source Software
|   └──bacnet               BACnet Protocol Stack
├──OPC_UA_SERVER            Open62541 , BACnet related OPC UA files
├──rzn                      RZ
|   ├──arm                  ARM
|   |   └──CMSIS_5          CMSIS
|   ├──aws                  AWS
|   |   └──amazon-freertos  FreeRTOS
|   ├──board                Board
|   |   └──rzn2l_rsk        RZ/N2L Renesas Starter Kit
|   └──fsp                  Flexible Software Package
├──rzn_cfg                  Configuration
|   ├──aws                  AWS
|   └──fsp_cfg              FSP
|       └──bsp              Board Support Package
├──rzn_gen                  Generated files
├──script                   Linker Script
├──src                      User Thread Entry
└──user                     User files
    ├──oss                  Other OSS
    |   ├──amazon-freertos  AWS_OSS
    |   └──lwip             lwIP_OSS
    └──renesas              Renesas common files
        ├──application      User lwIP application, OPCfeature
        ├──module           User module
        └──oss_deps         lwIP OSS dependencies
```

Fig.3-1 Folder Structure

3.2 Boot Sequence

Describes the boot procedure and memory allocation.

The boot mode of the sample software is 16-bit bus NOR flash boot mode. The figure below shows the BSP tag in the Smart Configurator.

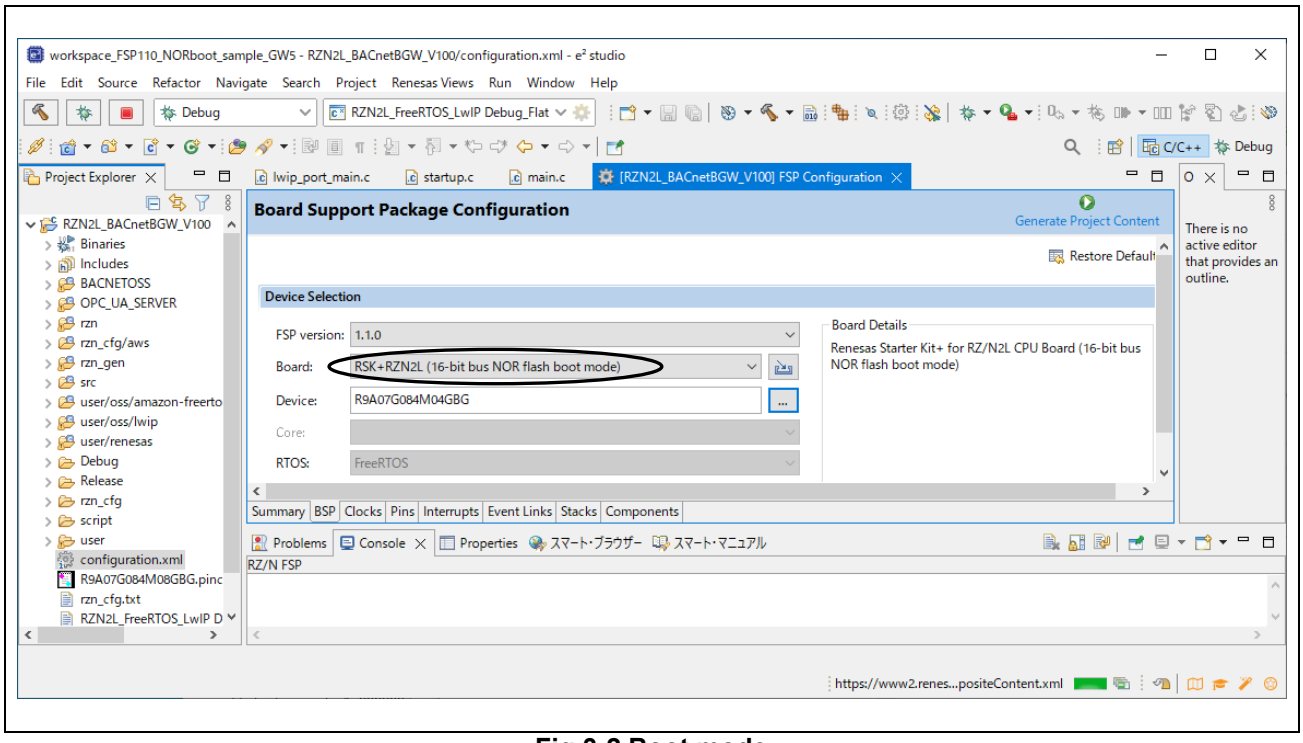


Fig.3-2 Boot mode

After downloading the program to the flash memory, the board operates independently by pressing the RESET button on the RSK board or turning the power ON without a debugger connection. You can still connect the debugger for evaluation. However, if jumper 9 (J9) of the RSK board is shorted, the debugger (J-Link OB) cannot be connected.

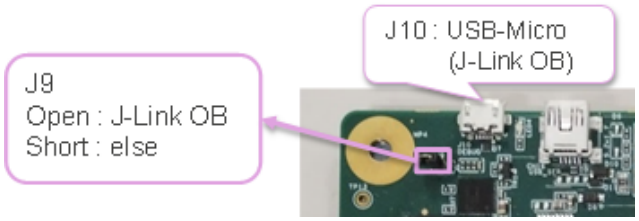


Fig.3-3 J9

This is the Smart Configurator screen showing the terminal settings (Pins tag) of the NOR flash memory device. No changes are required because they have already been configured.

When 16-bit bus NOR flash boot mode is selected, the pin settings for address buses A0-A20 are made automatically, however, if the program size is 2 MB or larger, the pin settings for A21-A25 must be made individually in BSC. In this sample project, the settings have already been made and need not to be changed.

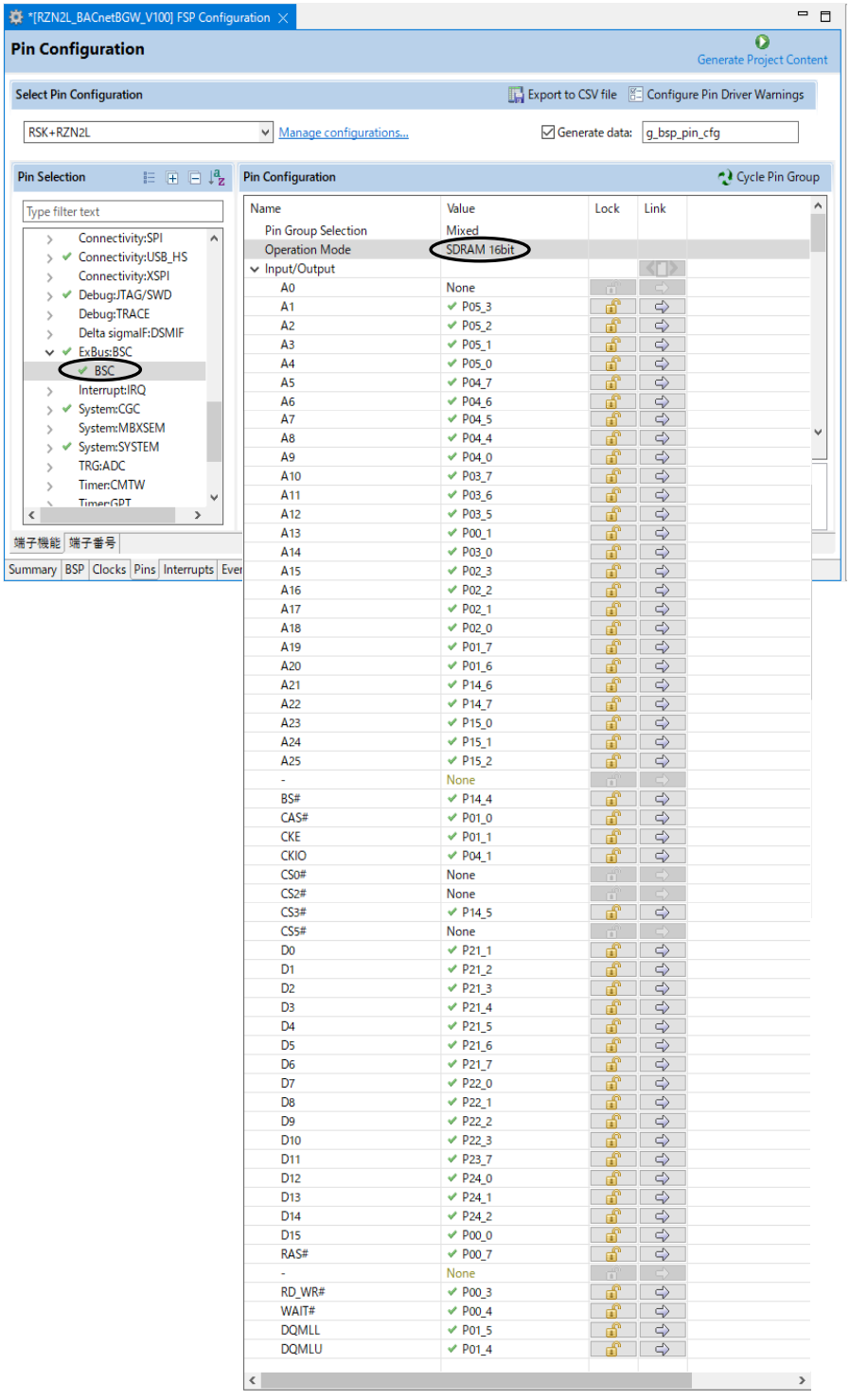


Fig.3-4 BSC Pin Configuration

The order of memory writing in the boot sequence is shown in the writing order column in Fig.3-5. For an overview of (1) through (5), see Section エラー! 参照元が見つかりません。.

Address	Memory	Content	Length		writing order	remarks
0x00000000	ATCM	intvec(64B)	0x00020000	128KB	(3)	Internal tightly coupled memory
0x00000040		Unused				
0x00000100		hal_entry,ROMdata				
0x00020000	Reserved area	-	-			
0x00100000	BTCM	Unused	0x00020000	128KB	(2)	Internal tightly coupled memory
0x00102000		Loader program(24KB)				
0x00108000		stack(60KB)				
0x00120000	Reserved area	-	-			
0x10000000	SYSTEM_RAM	Body of program and data	0x00180000	1.5MB	(4)	Cached system RAM
0x10180000	Reserved area	-	-			
0x30000000	SYSTEM_RAM_MIRROR	Unused	0x00180000	1.5MB		
0x30180000	Reserved area	-	-			
0x40000000	xSPI0_CS0_SPACE_MIRROR	Unused	0x04000000	64MB		
0x44000000	xSPI0_CS1_SPACE_MIRROR	Unused	0x04000000	64MB		
0x48000000	xSPI1_CS0_SPACE_MIRROR	Unused	0x04000000	64MB		
0x4C000000	xSPI1_CS1_SPACE_MIRROR	Unused	0x04000000	64MB		
0x50000000	CS0_SPACE_MIRROR	Unused	0x04000000	64MB		
0x54000000	CS2_SPACE_MIRROR	Unused	0x04000000	64MB		
0x58000000	CS3_SPACE_MIRROR	Unused	0x04000000	64MB		
0x5C000000	CS5_SPACE_MIRROR	Unused	0x04000000	64MB		
0x60000000	xSPI0_CS0_SPACE	Unused	0x04000000	64MB		
0x64000000	xSPI0_CS1_SPACE	Unused	0x04000000	64MB		
0x68000000	xSPI1_CS0_SPACE	Unused	0x04000000	64MB		
0x6C000000	xSPI1_CS1_SPACE	Unused	0x04000000	64MB		
0x70000000	CS0_SPACE	Parameters for the loader(76B)	0x02000000	32MB	(1)	256M bits NOR Flash
0x7000004C		Loader program(24KB)				
0x7000604C		Body of program and data				
0x72000000		Unused	0x02000000	32MB		
0x74000000	CS2_SPACE	Unused	0x04000000	64MB		
0x78000000	CS3_SPACE	Body of program and data	0x02000000	32MB	(5)	256M bits SDRAM
0x7A000000		Unused	0x02000000	32MB		
0x7C000000	CS5_SPACE	Unused	0x04000000	64MB		

Fig.3-5 Memory layout

3.3 OPC UA Stack

3.3.1 OPC UA

OPC UA was developed by the OPC Foundation as an open communication standard to realize secure and reliable data exchange for various industries including the industrial automation field. OPC Classic, the predecessor of OPC UA, was Windows-based, but OPC UA is now multi-platform and able to run on various platforms from Windows systems including cloud computing to RTOS for field devices.

This sample software implements the open source open62541 protocol stack on FreeRTOS, which is provided as sample software for RZ/N2L.

3.3.2 Information Model

To achieve interoperability among vendors and industries, OPC UA provides a unified data model called the Information Model in xml file format. It includes built-in models commonly used in OPC UA, companion models used by each industry or organization, and vendor-specific models that can be customized by each vendor. (Fig. 3-6)

In this sample software, the .xml file of the information model is converted to C language code. For details, please refer to chapter 5.1.

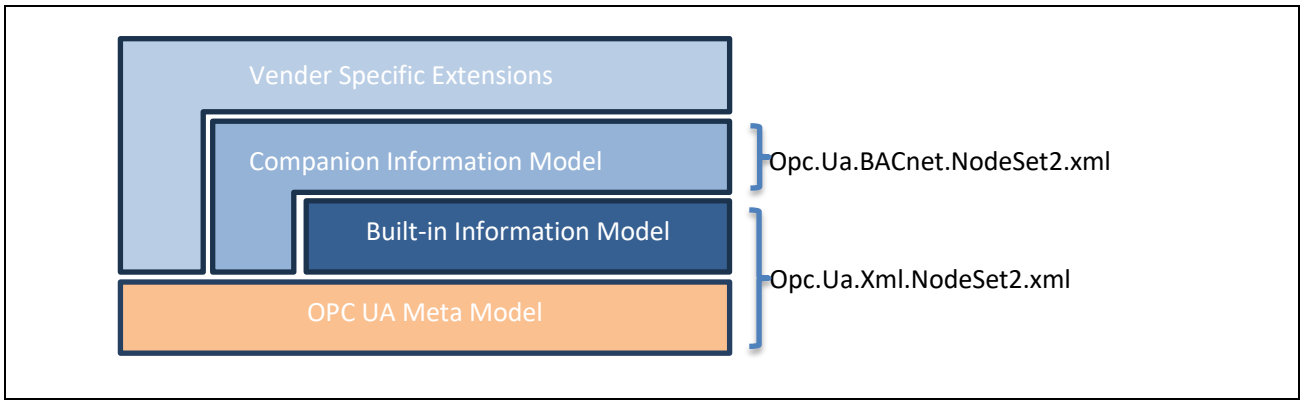


Fig. 3-6 Information Model

(1) Opc.Ua.Xml.NodeSet2.xml

It provides the "Meta Model", which is a set of rules for describing the OPC UA information model, and the "Built-in Information Model", which is the basic information model of OPC UA described by the Meta Model.

In this sample software, the following versions are applied.

The OPC UA XML version 1.05.01 (UA-Nodeset/XML/Opc.Ua.Xml.NodeSet2.xml at d1bb6a22125bd7cd986272b1ee98a18a91d76fff · OPCFoundation/UA-Nodeset · GitHub)

(2) Opc.Ua.BACnet.NodeSet2.xml

This is one of the information models defined for each industry segment according to companion specifications, and is a companion information model for BACnet, a communication protocol for building automation.

(OPC UA Companion Specification : OPC 30030: BACnet)

In this sample software, the following versions are applied.

bacnet XML version 2.0 (UA-Nodeset/BACnet/Opc.Ua.BACnet.NodeSet2.xml at d1bb6a22125bd7cd986272b1ee98a18a91d76fff · OPCFoundation/UA-Nodeset · GitHub)

3.3.3 Open62541

This sample software adopts the open source open62541 as the protocol stack for the OPC UA server. For more information on Open62541, refer to the following Link.

open62541

(1) Version

Base version of open62541 in this sample software is the following.

Base Version : v1.3.4-564-gb7e5e49f3

(commit b7e5e49f32d00490be74c2eacef892c7fbd0be60)

(2) License

The license terms for the Open62541 are MPL v2.0.

Please refer https://www.mozilla.org/en-US/MPL/2.0/ for more information and comply with the license terms and conditions.

(3) open62541 files

To run Open62541 in the environment of freeRTOS + LwIP, the method to generate open62541.c and open62541.h files using CMake as described in the following link is applied

Building open62541 — open62541 1.3.0-dirty documentation

For more information, please refer to chapter 5.1 in the Appendix.

3.3.4 Restrictions

The released version V1.0.0 of this sample software has the following restrictions.

- ✓ Not supported for security certificates.
- ✓ Not supported for NTP client. (It is possible to obtain UTC time by Time Synchronization Method.)

3.4 BACnet Stack

BACnet (Building Automation and Control Network) is the major communication protocol for Building Automation (BA) standardized in ASHRAE/ANSI Standard 135. Air conditioning, lighting, disaster prevention, access control, etc. can be integrated to control and monitor buildings.

BACnet devices are classified into different profiles according to their function and application, such as operator or controller. Major profiles include the central monitoring profile B-OWS (BACnet Operator Workstation), the controller profile B-BC (BACnet Building Controller), and the profile for various sensors B-SS (BACnet Smart Sensor). In addition, there are also B-RTR (BACnet Router) and B-GW (BACnet Gateway) profiles for relaying between different communication protocol devices as Miscellaneous profiles that can be used in combination with the above-mentioned controller profiles.

This sample software realizes a Gateway (B-GW) between BACnet and OPC UA and consists of two device profiles, B-GW and B-BC. B-GW maps properties of objects defined in BACnet to node variables defined in OPC UA according to OPC UA's Companion Specification OPC 30030: BACnet. The B-GW forward access requests from OPC UA clients to BACnet server equipment (in this case, B-SS sample software) in BACnet/IP networks, and then forwards responses from the BACnet server to the OPC UA client.

Details on how to build and start the B-SS sample software are described in the application note (R01AN6789EJ****). Refer to Section 5. BACnet I/P Communication in the application note.

3.4.1 BACnet Protocol Stack

BACnet Protocol Stack (bacnet-stack) is an open-source stack for the BACnet communication protocol. This sample software is a port of BACnet Protocol Stack to RZ/N2L.

Base Version : eb36033f (Commits on Jan 18, 2023)

GitHub - bacnet-stack/bacnet-stack: BACnet Protocol Stack library provides a BACnet application layer, network layer and media access (MAC) layer communications services.

3.4.2 License

The license terms for the BACnet Protocol Stack are GPL with exception license. The original text is transcribed below for reference. Please refer BACnet Protocol Stack download | SourceForge.net for more information and comply with the license terms and conditions.

> *This BACnet protocol stack implementation is specifically designed for the embedded BACnet appliance, using a GPL with exception license (like eCos), which means that any changes to the core code that are distributed are shared, but the BACnet library can be linked to proprietary code without the proprietary code becoming GPL. Note that some of the source files are designed as skeleton or example or template files, and are not copyrighted as GPL.*
>
> *The text of the GPL exception included in each source file is as follows:*
>
> *"As a special exception, if other files instantiate templates or use macros or inline functions from this file, or you compile this file and link it with other works to produce a work based on this file, this file does not by itself cause the resulting work to be covered by the GNU General Public License. However the source code for this file must still be made available in accordance with section (3) of the GNU General Public License."*

3.4.3 Specifications

3.4.3.1 Restrictions

This sample software supports the gateway (B-GW) device profile defined in the BACnet standard. It also includes some B-BC controller device profile features, but does not meet the requirements for standard B-BC device profiles in the BACnet specification and is not supported in this version.

B-BC will be supported in the next version or later. Please refer to Chapter 5.2 for the support status of B-BC in this version.

3.4.3.2 BACnet Revision

The protocol version and revision of the BACnet stack used in this sample software are as follows

・BACnet standard Protocol Version：1

・BACnet standard Protocol Revision：22

The BACnet standard document (ANSI/ASHRAE Standard 135-2020) indicates version 1 and revision 22.

The define value of the ported open-source stack revision is 24, but the objects added in revisions 23 and 24 are not supported and changed to 22 in this release, as shown below.

BACNETOSS\bacnet\bacdef.h

#define BACNET_PROTOCOL_REVISION 22

3.4.3.3 Service

The sequence of BACnet stack implemented in the sample software is service driven. Interoperability of BACnet devices is provided by the connection between users and providers via services (WhoIs, I-Am, ReadProperty, etc.).

There are two types of services: Unconfirmed and Confirmed. In the unconfirmed type, the provider does not return an Ack for the service requested by the user. On the other hand, confirmed type will return an Ack.

- **Users** of the sample software mean the following.

  It corresponds to a client that connects to BACnet server through the BACnet /IP protocol.

- **Providers** mean the following.

  It corresponds to a server that connects to BACnet clients through BACnet /IP protocol.

The B-GW running this sample software is a user to other providers (B-SS) in the BACnet internetwork.

However, it could also be a provider to other users in the BACnet Internetwork.

The services implemented in the sample software are as follows.

Table 3-1 Implemented Services in the sample software

BACnet service	Initiate[1]	Execute[2]
Who-Is	✓	✓
I-Am	✓	✓
Who-Has	✓	✓
I-Have	✓	✓

BACnet service	Initiate[1]	Execute[2]
ReadProperty	✓	✓
WriteProperty	✓	✓
DeviceCommunicationControl		
ReinitializeDevice		
AtomicReadFile		
AtomicWriteFile		
TimeSynchronization		
UTCTimeSynchronization		
SubscribeCOV		
ConfirmedCOVNotification		
UnconfirmedCOVNotification		
ReadPropertyMultiple		
ReadPropertyConditional		
ReadRange		
WritePropertyMultiple		
GetAlarmSummary		
GetEventInformation		
GetEnrollmentSummary		
AcknowledgeAlarm		
ConfirmedEventNotification		
UnconfirmedEventNotification		
UnconfirmedTextMessage		
ConfirmedTextMessage		
AddListElement		
RemoveListElement		
CreateObject		
DeleteObject		
UnconfirmedPrivateTransfer		
ConfirmedPrivateTransfer		
VTOpen		
VTData		
VTClose		

✓ is applicable, blank is not applicable.

1. Send a BACnet service request or notification.
2. Execute the BACnet service and send a response (if a confirmed service is requested).

The following is an overview of the implemented services

Table 3-2 Implemented services in the B-SS sample software

BACnet service	Description
Who-Is	Who-Is service is used by BACnet users to know which other BACnet devices are sharing the network. Who-Is service is a broadcasted, unconfirmed (does not require an Ack) service.
I-Am	I-Am service is intended to respond to Who-Is service requests. However, I-Am service requests are broadcast transmissions that can be sent anytime. Receipt of Who-Is service request need not be preceded.
Who-Has	Who-Has service is used by BACnet users to identify BACnet devices with specific objects. Who-Has service is a broadcasted, unconfirmed type of service.
I-Have	I-Have service is available to respond to Who-Has service requests. However, I-Have service requests can be issued at any time. Receipt of Who-Has service requests need not be preceded; I-Have service is sent broadcast and is an unconfirmed type of service.
ReadProperty	ReadProperty service is used by BACnet users to request the value of one property of one BACnet object; the BACnet provider responds with Ack and returns the result.
WriteProperty	WriteProperty service is used by BACnet users to change the value of a specified property of one of the BACnet objects. BACnet provider responds with an Ack. If you want to restrict the write access to a specified property, an error with "Error Class" PROPERTY and "Error Code" WRITE_ACCESS_DENIED is returned.

3.4.3.4 Object

A BACnet device consists of a set of objects. An object is represented by an object type and an instance number from 0 to 4194303, which is called an object ID. However, the number 4194303 means invalid and is not used.

The device itself is also an object and is defined in Device object; the object ID of the device is called the device ID. Each BACnet device is required to have a Device object.

Furthermore, objects consist of a set of properties of various data types, and a BACnet device accesses hardware to read and write these properties.

The implemented Objects of the sample software is as follows.

Table 3-3 Implemented Objects in the sample software

BACnet object type	Object ID	Implementation
Accumulator		
Analog Input	Analog Input, 0	✓
	Analog Input, 1	✓
Analog Output		
Analog Value		
Averaging		
Binary Input		
Binary Output		
Binary Value		
Calendar		

BACnet object type	Object ID	Implementation
Command		
Device	Device, 12	✓
Event Enrollment		
File		
Group		
Life Safety Point		
Life Safety Zone		
Loop		
Multi state Input		
Multi state Output		
Multi state Value		
Notification Class		
Program		
Pulse Converter		
Schedule		
Trend Log		
Access Door		
Event Log		
Load Control		
Structured View		
Trend Log Multiple		
Access Point		
Access Zone		
Access User		
Access Rights		
Access Credential		
Credential Data Input		
CharacterString Value		
DateTime Value		
Large Analog Value		
BitString Value		
OctetString Value		
Time Value		
Integer Value		
Positive Integer Value		
Date Value		
DateTime Pattern Value		

BACnet object type	Object ID	Implementation
Time Pattern Value		
Date Pattern Value		
Network Security		
Global Group		
Notification Forwarder		
Alert Enrollment		
Channel		
Lighting Output		
Network Port	NetworkPort,1	✓
Binary Lighting Output		

✓ is applicable, blank is not applicable, and “Not available” is restriction of this sample software version.

Outlines of the implemented object types are as follows.

Table 3-4 Outlines of the implemented object types

BACnet Object Type	Description
Analog Input	Analog Input object has properties that represent analog inputs from hardware.
Device	Binary Value object has properties that represent two states, ACTIVE or INACTIVE, resident in the memory of the BACnet device.
Network Port	The Network Port object has properties that represent the network configuration of the BACnet device; BACnet devices must have at least one Network Port object.

3.4.3.5 BIBB

BIBB (BACnet Interoperability Building Blocks) defines a set of services that apply to interoperating BACnet devices. "A" and "B" devices are defined, with the "A" device representing the BACnet user and the "B" device representing the BACnet provider.

BACnet standard (Annex L) defines various device profiles that describe the characteristics of each device, such as B-SS (BACnet Smart Sensor), B-BC (BACnet Building Controller), B-OWS (BACnet Operator WorkStation) and else. B-GW in this sample software have "B" characteristics.

The implemented BIBB of the sample software is as follows.

Table 3-5 Implemented BIBB of B-GW device profile

BIBB Class	BIBB	BACnet Service	Initiate[1]	Execute[2]	B-GW Standardized[3]
DataSharing	DS-RP-B	ReadProperty		✓	✓
	DS-WP-B	WriteProperty		✓	✓
Device & Network Management	DM-DDB-B	Who-Is		✓	✓
		I-Am	✓		✓
	DM-DOB-B	Who-Has		✓	✓
		I-Have	✓		✓

BIBB Class	BIBB	BACnet Service	Initiate[1]	Execute[2]	B-GW Standardized[3]
	GW-EO-B	The B device provides access to data and functionality in non-BACnet devices.			✓

✓ is applicable, blank is not applicable.

1. Sends a BACnet service request or notification. However, the B-SS does not send service requests, only notifications.
2. Execute the BACnet service and send a response (if a confirmed service is requested).
3. BIBBs which is defined as normalized for B-GW in ANNEX L of BACnet standards.

Outlines of the implemented BIBB in the B-SS sample software is as follows.

Table 3-6 Outlines of the implemented BIBB

BIBB	Description
DS-RP-B	Device B returns one property value to device A.
DS-WP-B	Device B writes value from device A to one property.
DM-DDB-B	Device B responds to the identification request from Device A.
DM-DOB-B	Device B responds to an identification request from Device A with the specified object.
DM-RD-B	Device B responds to the reinitialization request from Device A.
GW-EO-B	B devices provide access to the data and functionality of non-BACnet devices; B devices contain the data and functionality of other devices through BACnet objects and services.

3.5 Installation of Development Environment

3.5.1 e2studio

3.5.1.1 Install

Download the version listed in Table 1-1 and install it on your PC. The latest version has a downloadable installer that includes FSP, e2studio, and the GCC toolchain as a single package.

・Double-click the downloaded “setup_rznfsp_v1_1_0_e2s_v2022-10.exe”.

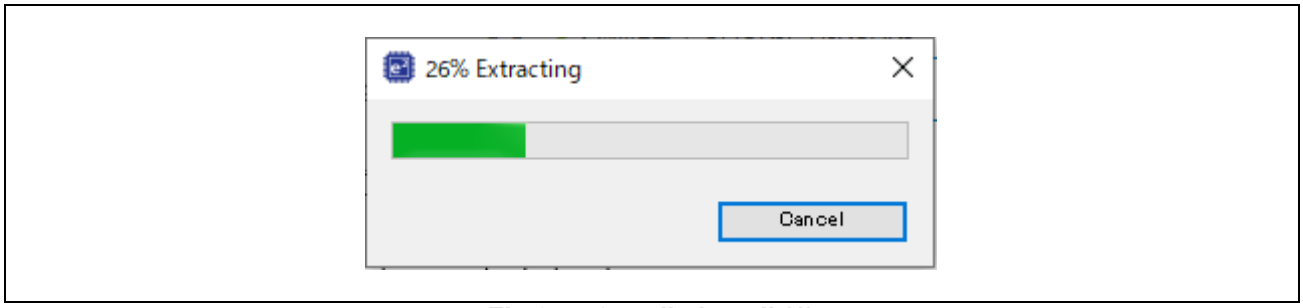


Fig.3-7 e2studio Install (1)

・Select Users

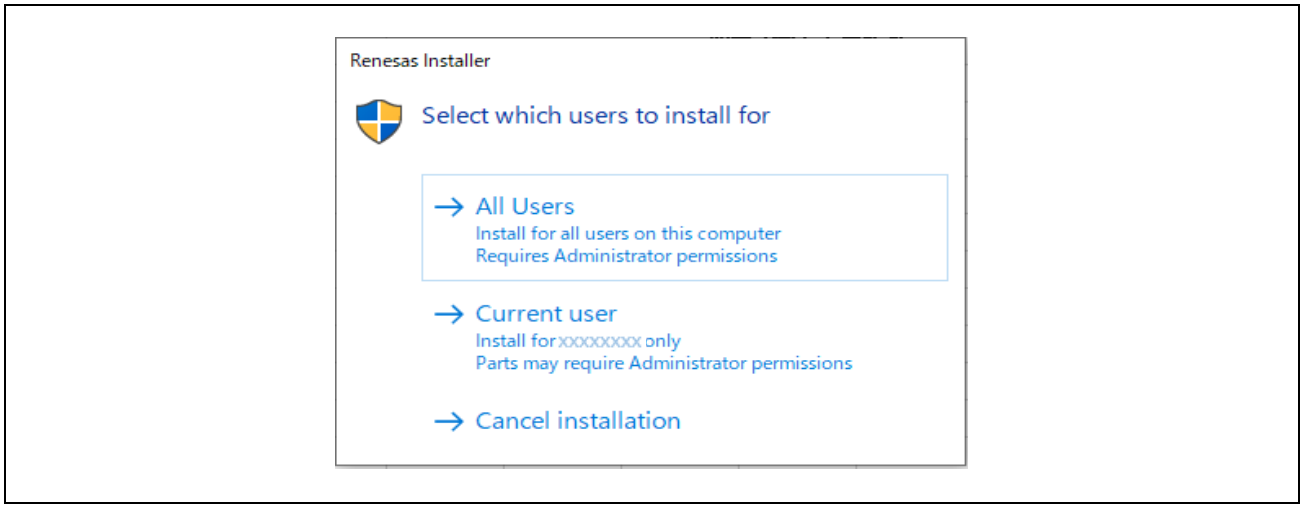


Fig.3-8 e2studio Install (2)

・Select Install Type

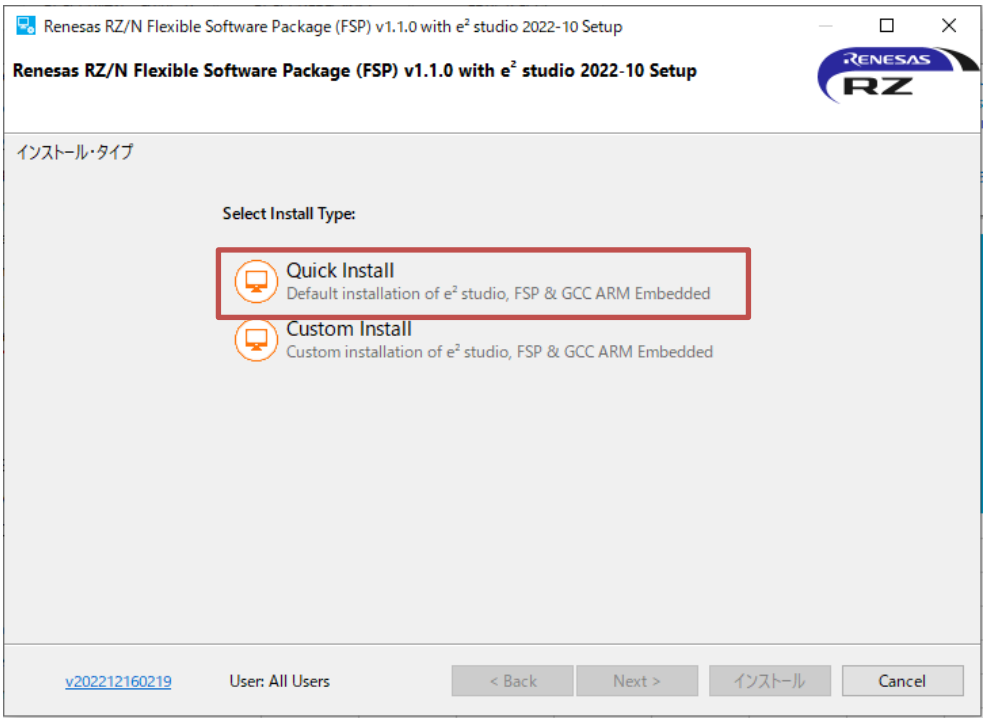


Fig.3-9 e2studio Install (3)

・Select Install folder

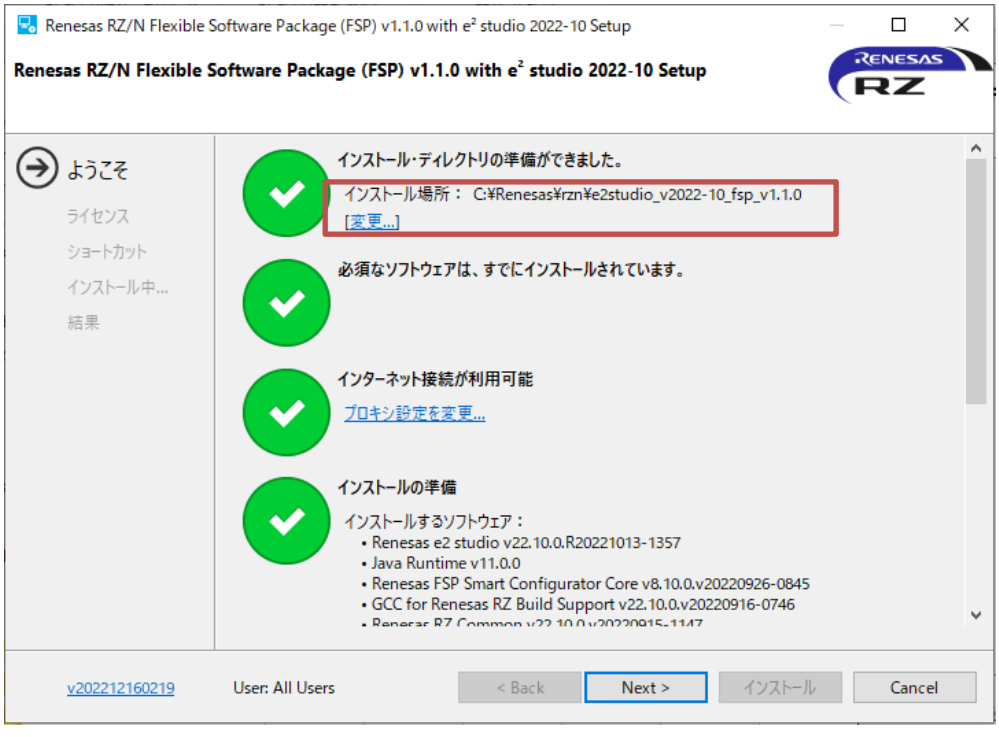


Fig.3-10 e2studio Install (4)

・Check and Click “Next”

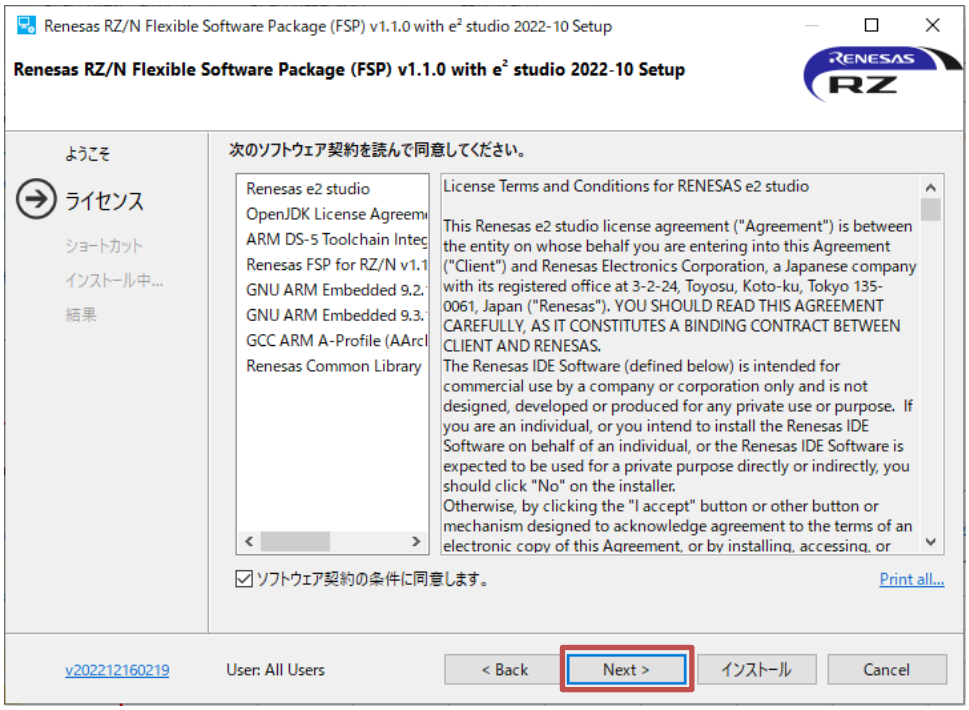


Fig.3-11 e2studio Install (5)

・Click “Install”

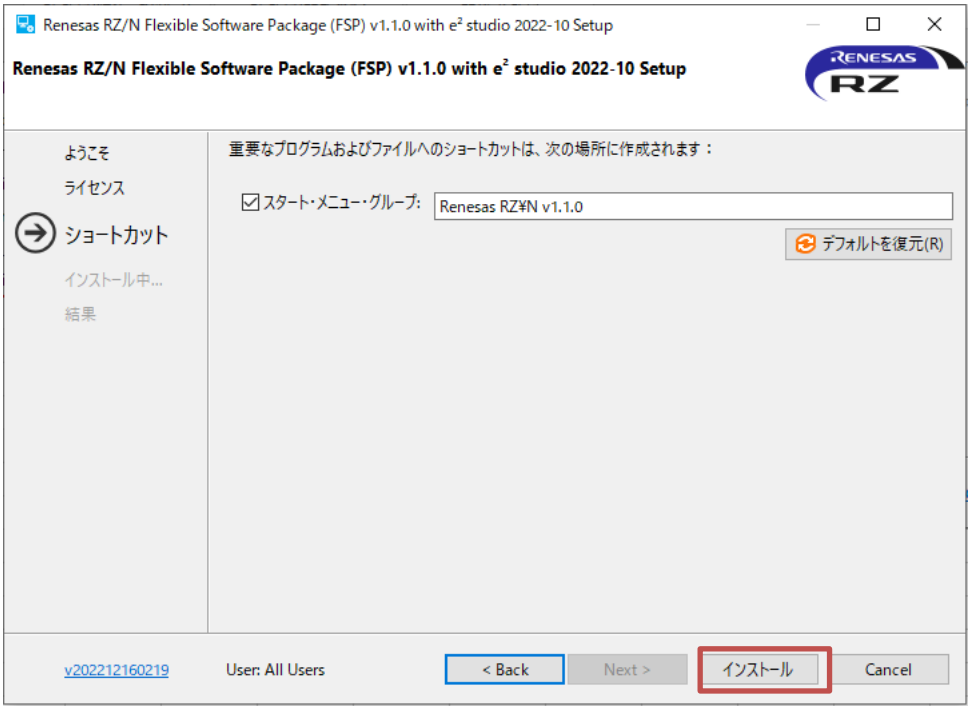


Fig.3-12 e2studio Install (6)

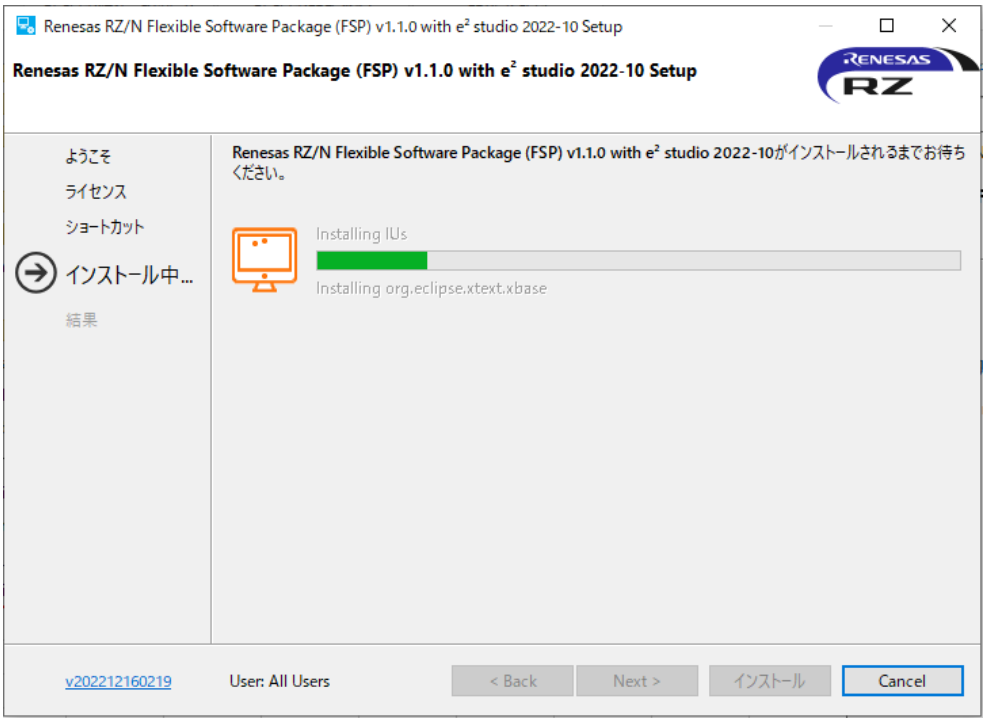


Fig.3-13 e2studio Install (7)

・Click “OK”

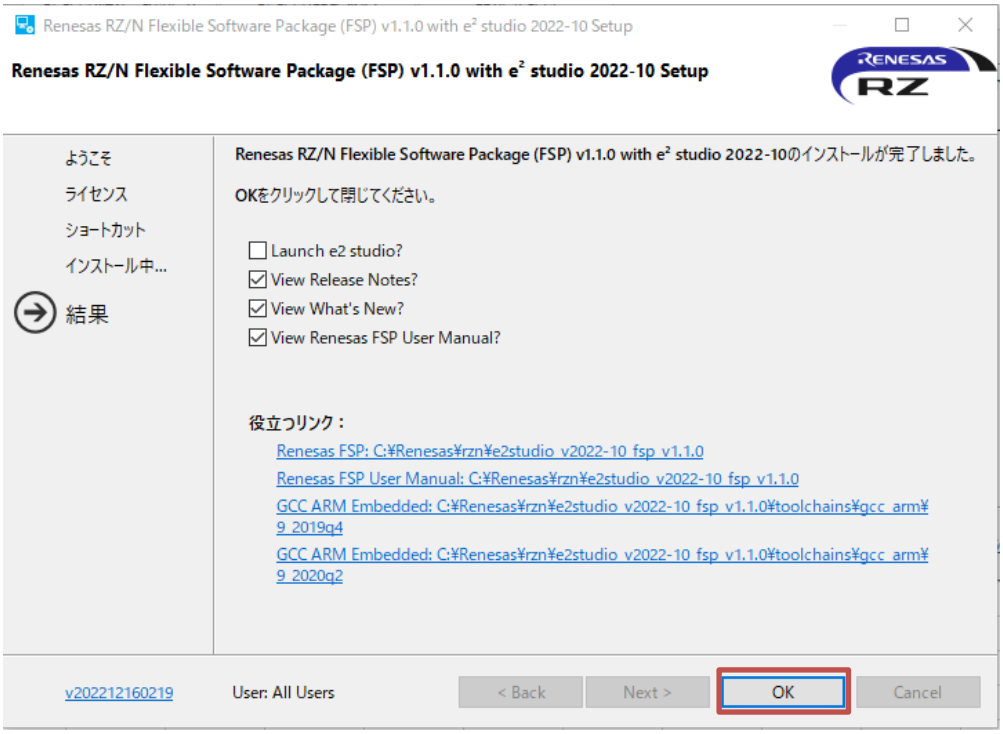


Fig.3-14 e2studio Install (8)

3.5.1.2 Project start-up

(1) Unzip package

First, unzip the archived package of this sample software (RZN2L_OPC_BGW_V***.zip) and store it in arbitrary folder. Because e2studio cannot recognize project properly if file path is too long in the folder hierarchy, place it in shorter path. Also, do not use multi-byte character, such as Japanese, in the folder path.

(2) Execute e2studio

Execute "e2studio.exe" to start e2studio in the following folder (default case) installed:

\Renesas\rzn\e2studio_v2022-10_fsp_v1.1.0\eclipse¥e2studio.exe

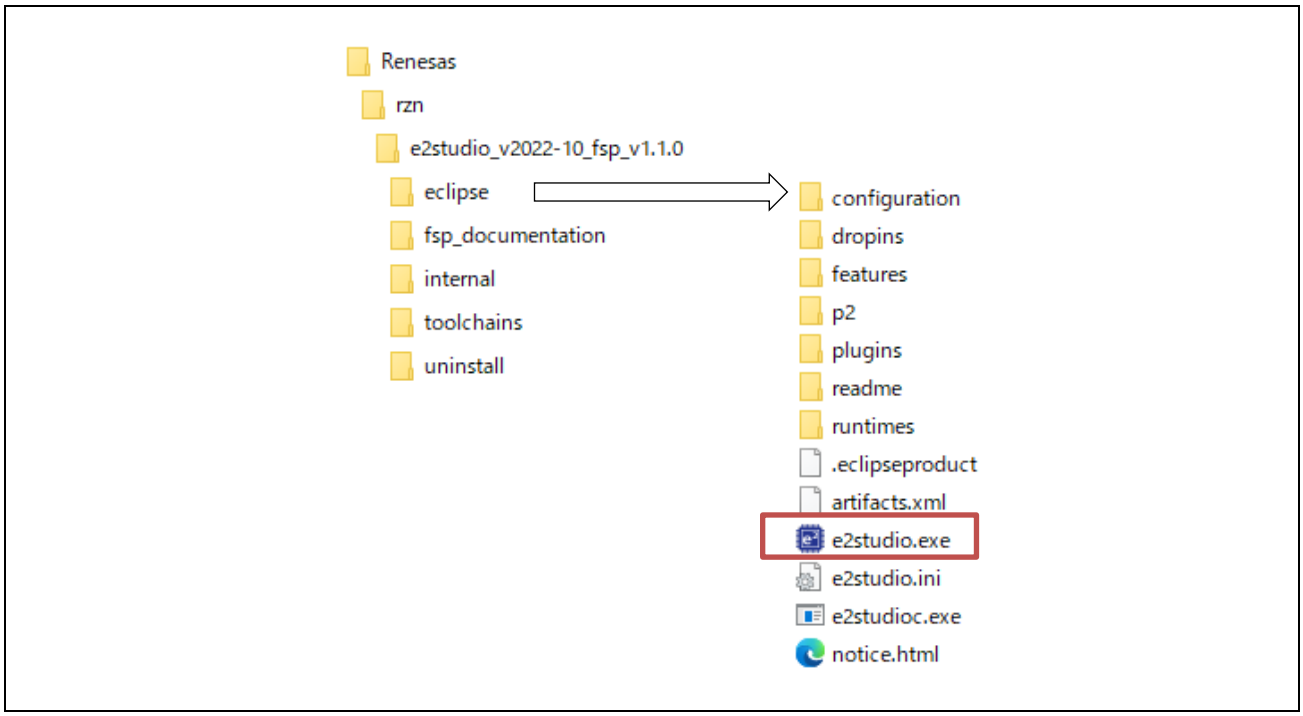


Fig.3-15 Launch project (1)

(3) Import Project

Enter any workspace directory and click “Launch”.

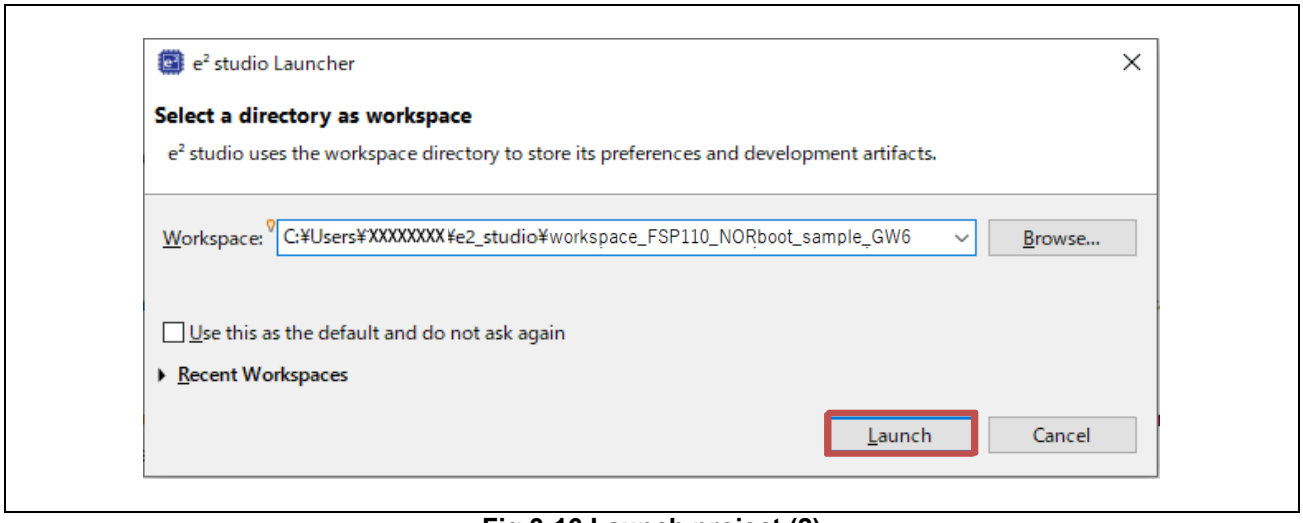


Fig.3-16 Launch project (2)

Select the toolchain “GNU ARM Embedded – 9.3.1.20200408”

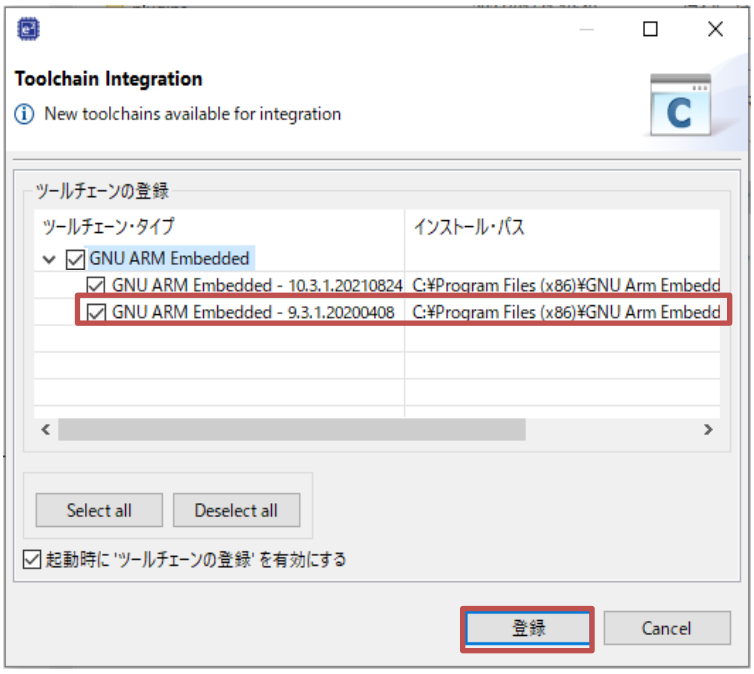


Fig.3-17 Launch project (3)

・Select “Import existing projects”

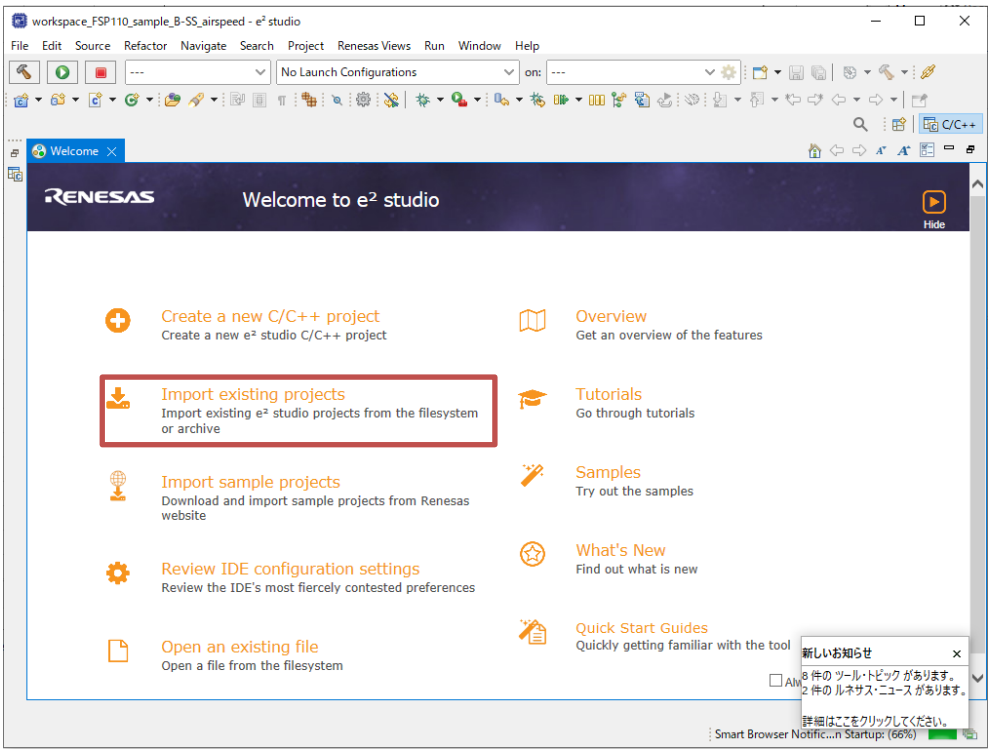


Fig.3-18 Launch project (4)

Click ”Browse” at “Select root directory” and enter the project folder to be imported.

Check the “Copy projects into workspace” checkbox to copy the import project.

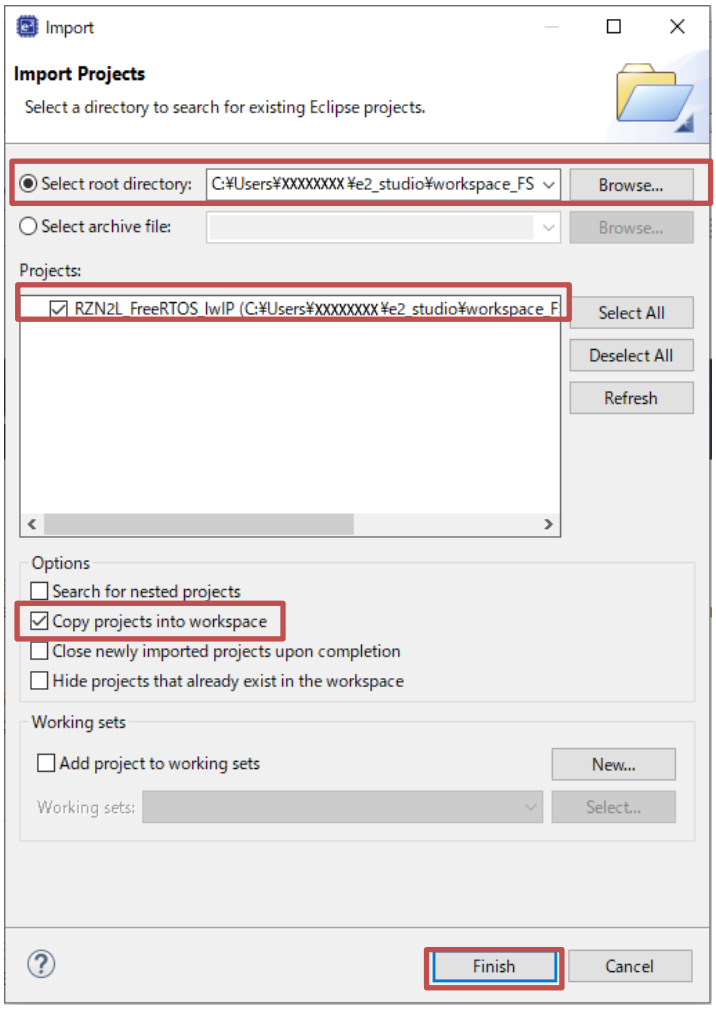


Fig.3-19 Launch project (5)

Click “Finish” in Fig.3-19 to display the following and click “Yes To All”.

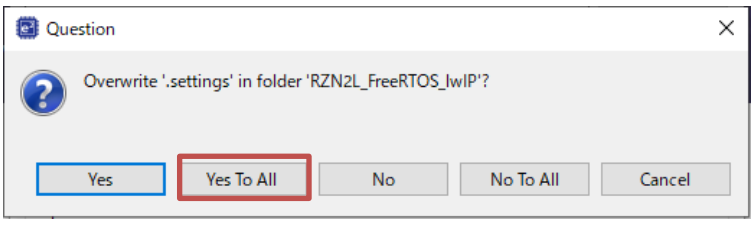


Fig.3-20 Launch project (6)

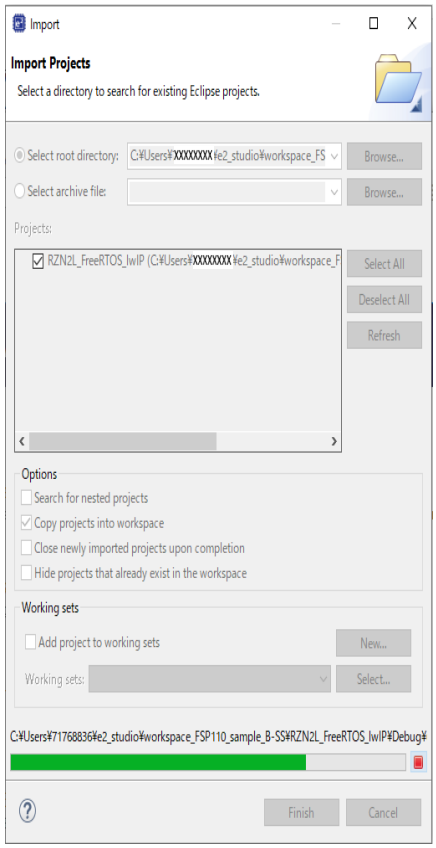


Fig.3-21 Launch project (7)

・When the project import is complete, the following will be displayed. The subsequent sections will be explained in chapter 4.3.1.

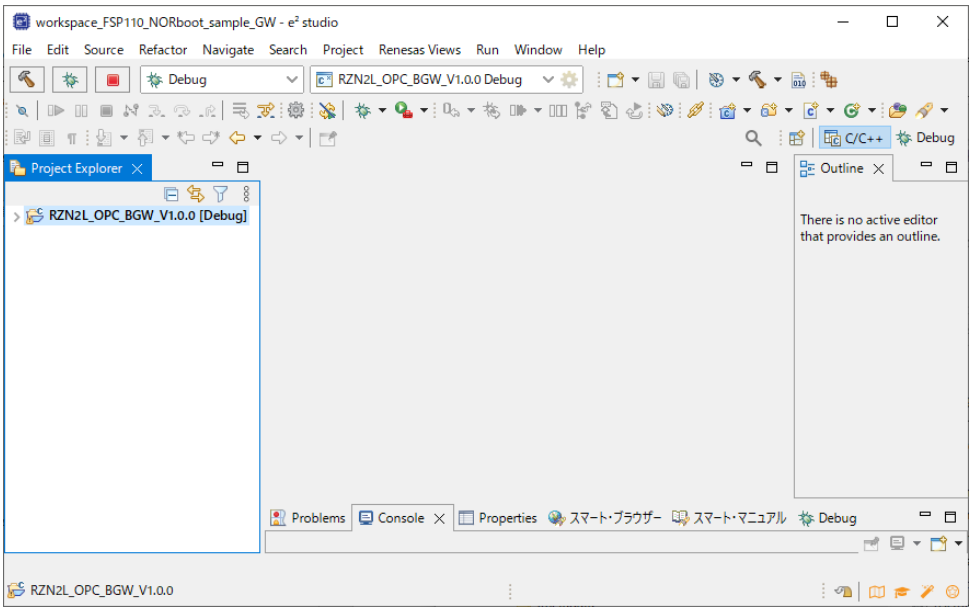


Fig.3-22 Launch project (8)

3.5.2 UaExpert

UaExpert is an OPC UA client tool. In this document, it is used to connect to the OPC UA server (B-GW) to access the object nodes of B-SS in the BACnet /IP network.

Download the version listed in Table 1-1 from the website and install it on your PC. Before downloading, you must register on the Unified Automation website and activate your account. All content is provided free of charge, but by downloading or installing the software from this web page, you automatically accept the Unified Automation Software License Agreement (SLA). For license terms for software and information, please refer to the following link.

https://www.unified-automation.com/products/sdk-overview/licenses.html#c341

Please check the above conditions of use before usage.

Fig.3-23 UaExpert

3.5.3 Wireshark

Wireshark is a free network protocol analyzer. Download and install Wireshark from the link in Table 1-1.

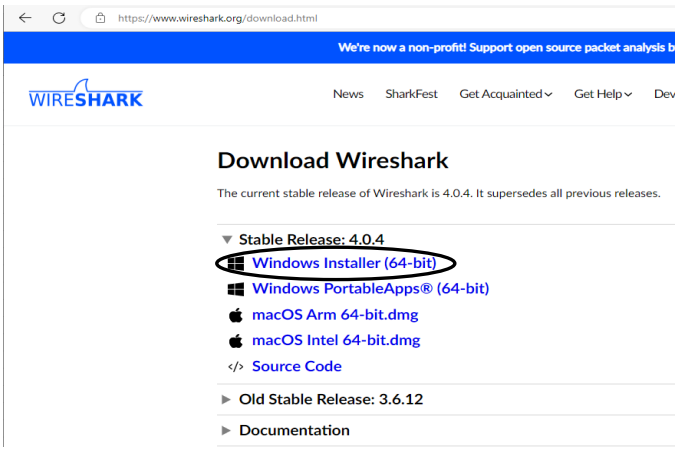


Fig.3-24 download Wireshark

4. Operation check

4.1 Connection

Fig.4-1 shows a connection diagram when running the sample software. Connect the Ethernet cable, J-Link OB debugger, and 5V DC cables to the RZ/N2L RSK board. As shown in the figure, when connecting board for B-SS, connect the air velocity sensor to the J26 connector on the board.

Note that the hardware settings (jumpers and DIP SW) for the B-GW board and the B-SS board are different. Please refer to chapter 2.1 for details. ETH2 (Ethernet2) connector on board for B-GW cannot be used. When using the debugger J-Link OB on the RSK board, open J9 and connect the USB Micro cable.

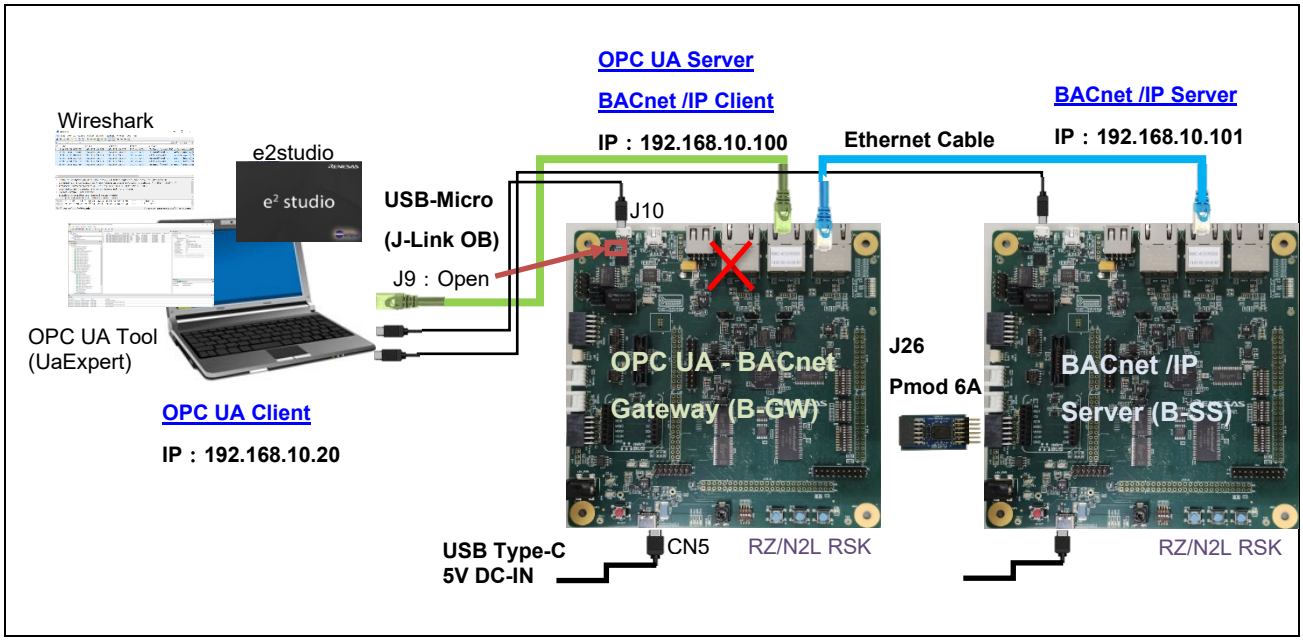


Fig.4-1 OPC UA - BACnet /IP Hardware Diagram

4.2 IP Address Settings

Set the address of the Ethernet on the PC that serves as the OPC UA Client.

Click on settings in Windows Start. Configure the IP address as follows.

Settings > Network and Internet > Change adapter options > Ethernet

＞Properties > Internet Protocol Version 4 (TCP/IPv4) > Properties

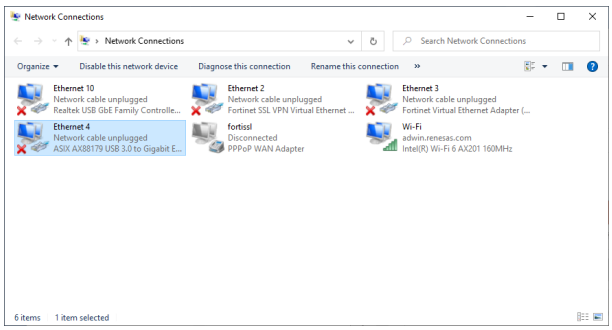


Fig.4-2 network connection

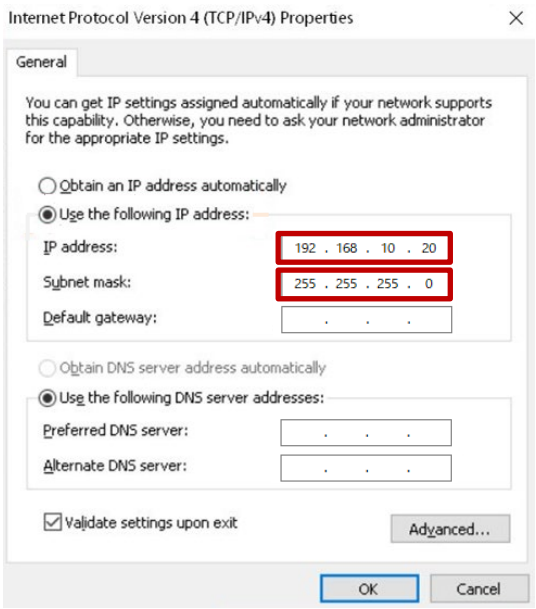


Fig.4-3 TCP/IPv4 properties

The IP address of the RSK board set in the B-GW sample software is 192.168.10.100. The IP address of the PC needs to be set to 192.168.10.XXX. In this document, 192.168.10.20 is used.

4.3 Start Project

First, import the project as described in section 3.5.1.2.

4.3.1 Build Configuration

Select the project name in the Project Explorer window, then open Properties in the Project menu.

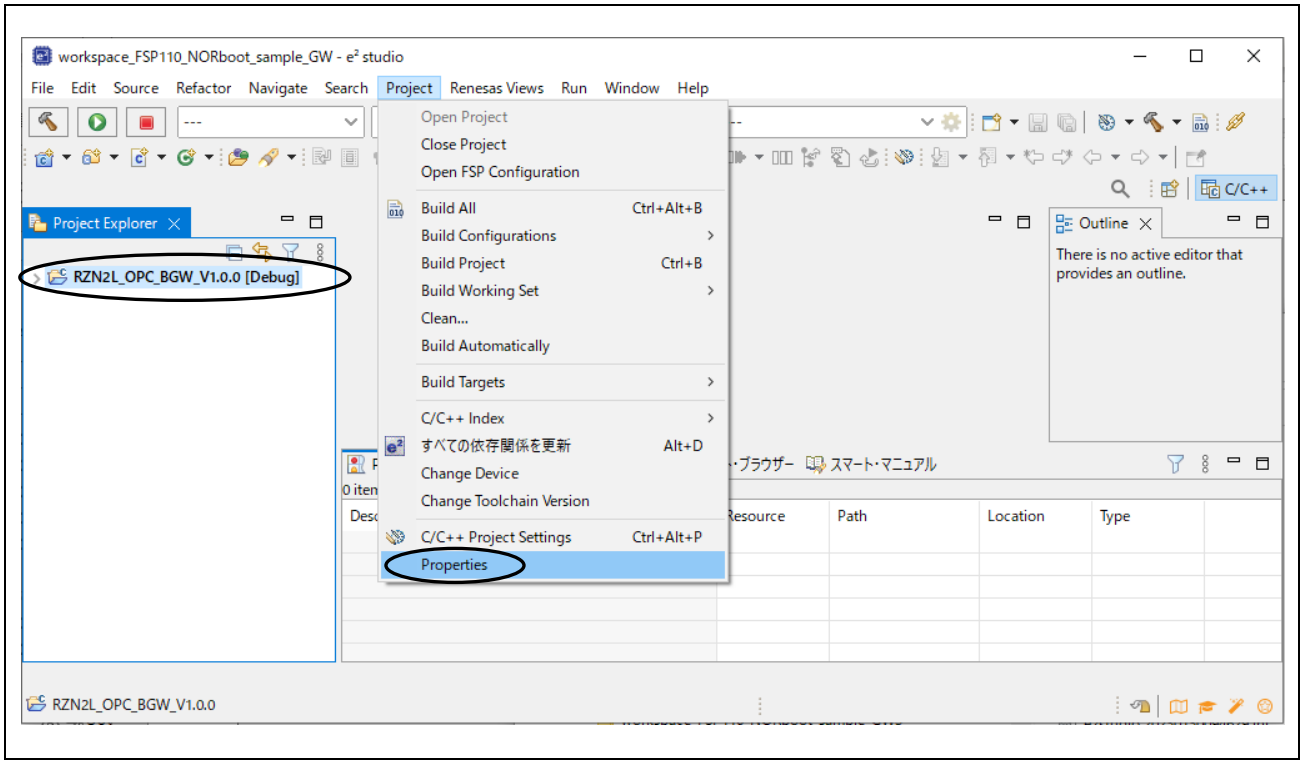


Fig.4-4 Open project properties

Select GNU C in Languages from the #Symbols tag in C/C++General > Paths and Symbols.

(1) Evaluation with a single RSK board

To evaluate the B-GW RSK board alone without connecting the B-SS RSK board, change #WITHOUT_B_SS_BOARD in Symbol to 1. Click "Edit..." to make changes

This allows to generate pseudo sensor input values inside the B-GW which are originally read from the B-SS, and read them from the PresentValue node of the AnalogInput,0 object of the B-GW, which is explained in Chapter 4.5.

(2) Evaluation with two RSK boards

To connect another RSK board for B-SS, change #WITHOUT_B_SS_BOARD in Symbol to 0 as shown in Fig. 4 5.

Click "Apply and Close" to apply the settings. Click "Yes" on the pop-up dialog.

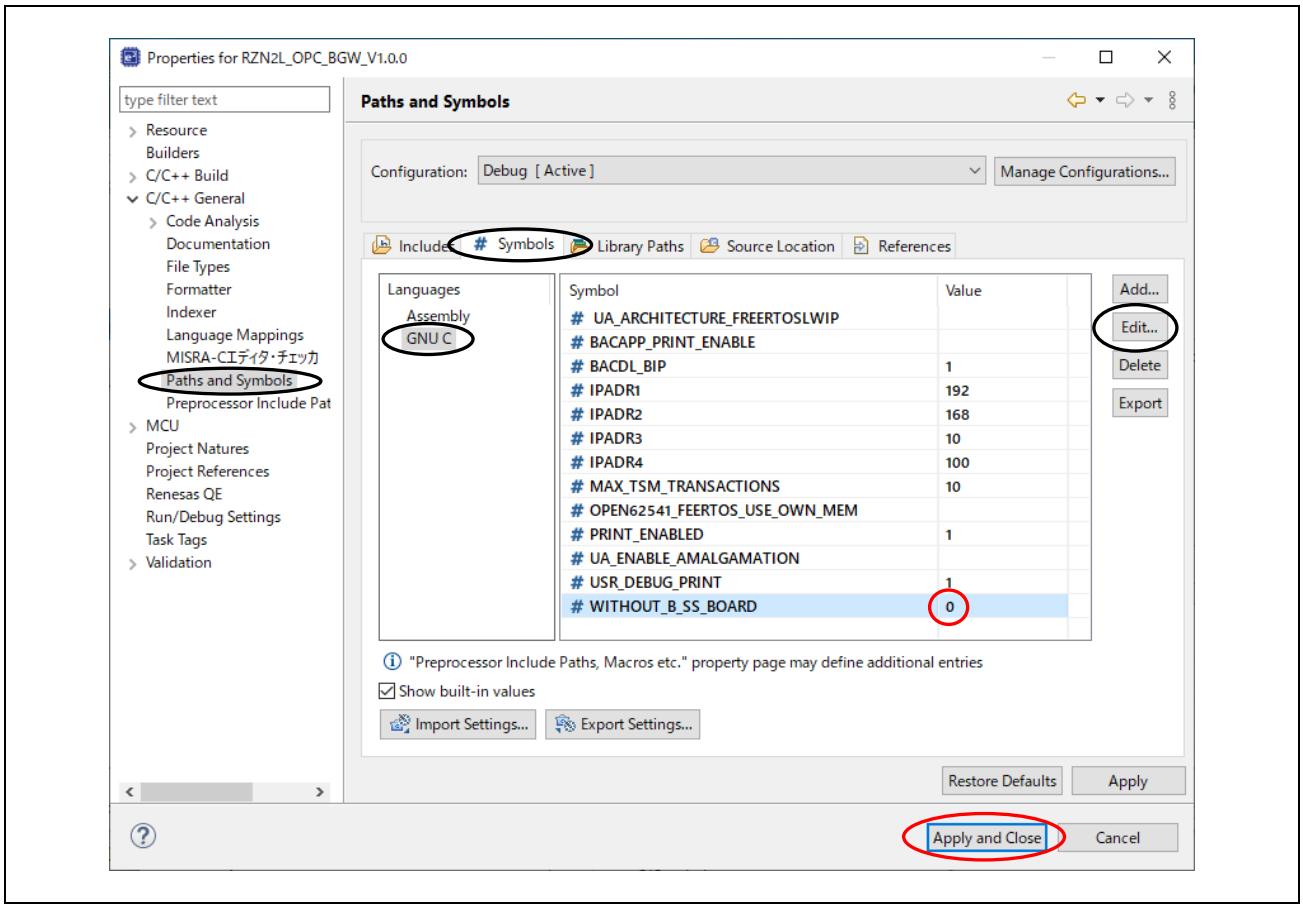


Fig.4-5 Change #WITHOUT_B_SS_BOARD

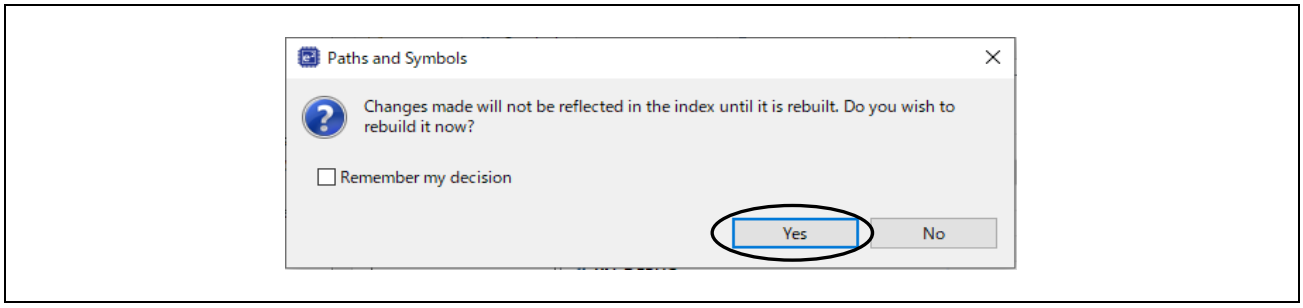


Fig.4-6 Click YES

4.3.2 Build

Select the project name in the Project Explorer window and click "Clean..." in the Project menu.

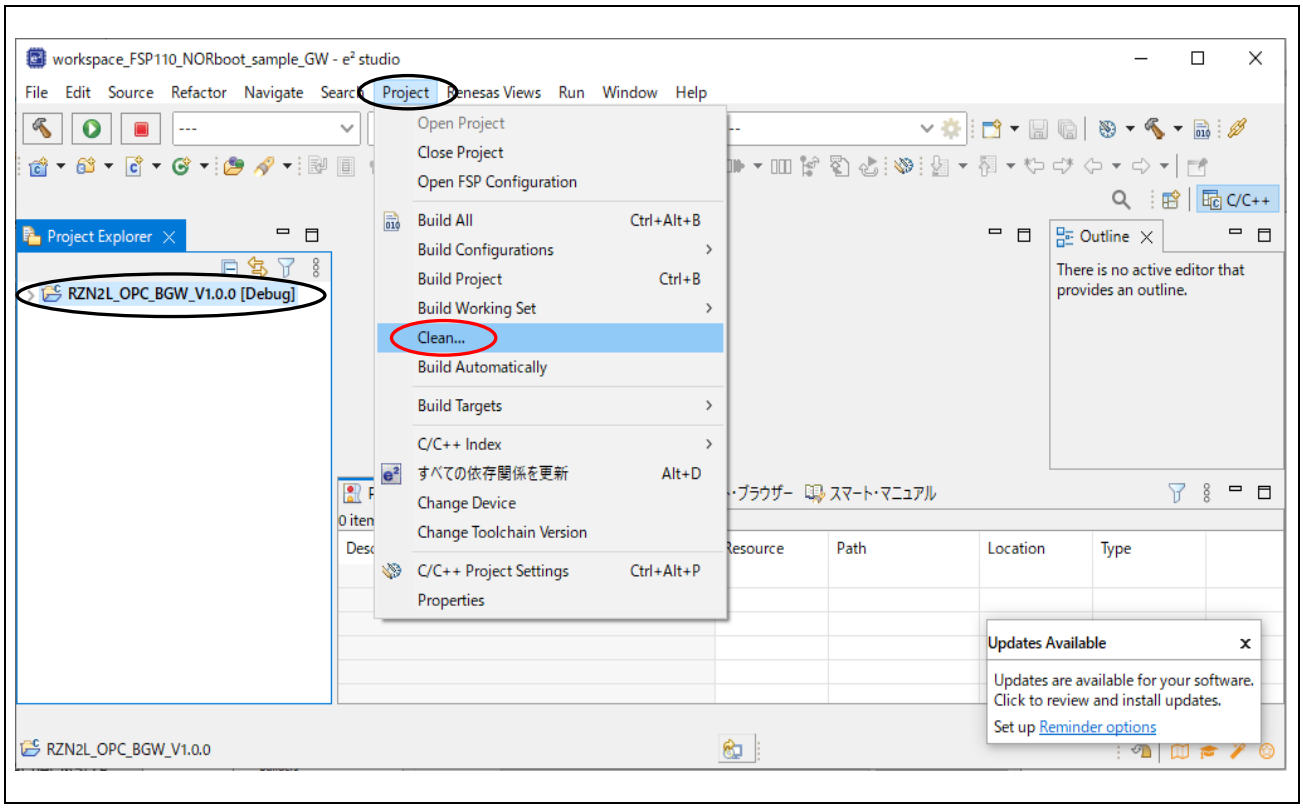


Fig.4-7 Open project Clean…

Enable the followings in the pop-up dialog and click "Clean" to start all builds.

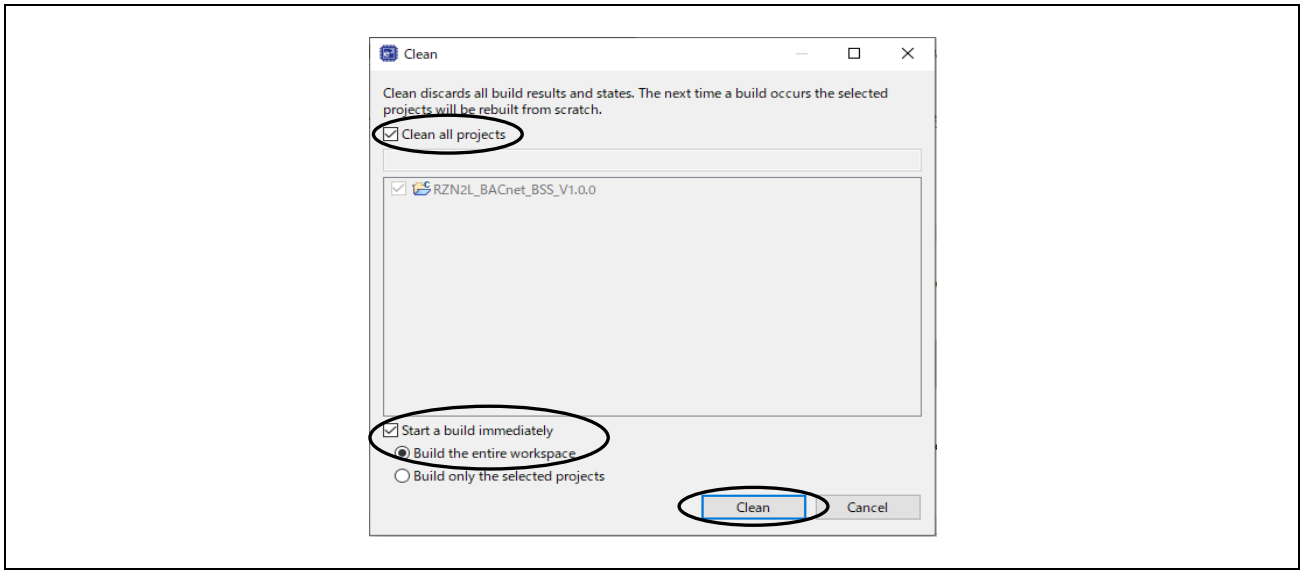


Fig.4-8 clean and rebuild

4.3.3 Debug Configurations

After confirming that the build result is 0 errors, select the project name in the Project Explorer window and click Debug Configurations... in the Run menu. Ignore the warning message that appears.

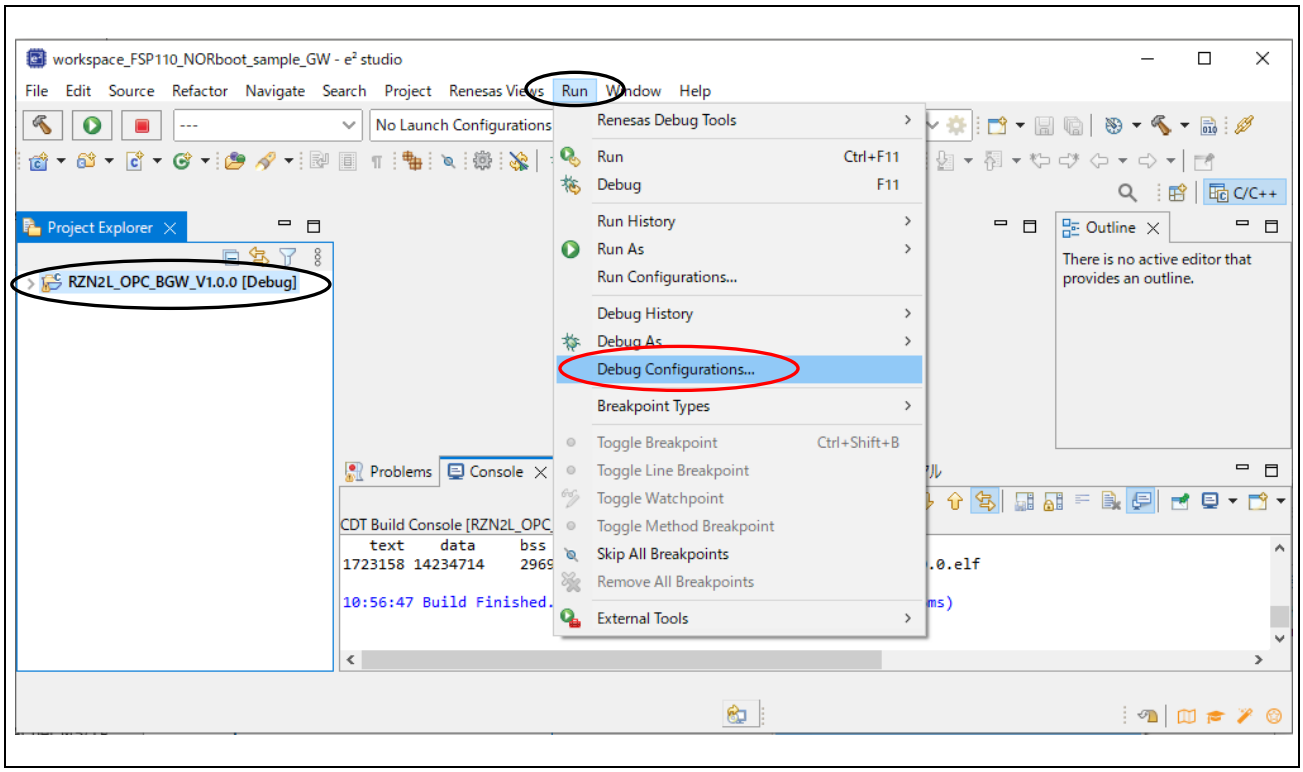


Fig.4-9 Open Debug Configurations…

Operations when starting the debugger for the first time after importing a project

Only when importing a project and launching the debugger for the first time, the following operations should be performed.

- Create RZN2L_OPC_BGW_V*** Debug[local]
- Select Target Device
- Debut Tool Settings

See the following explanation of the above.

a. Create RZN2L_OPC_BGW_V*** Debug[local]

Double click on Renesas GDB Hardware Debugging to generate RZN2L_OPC_BGW_V*** Debug[local]

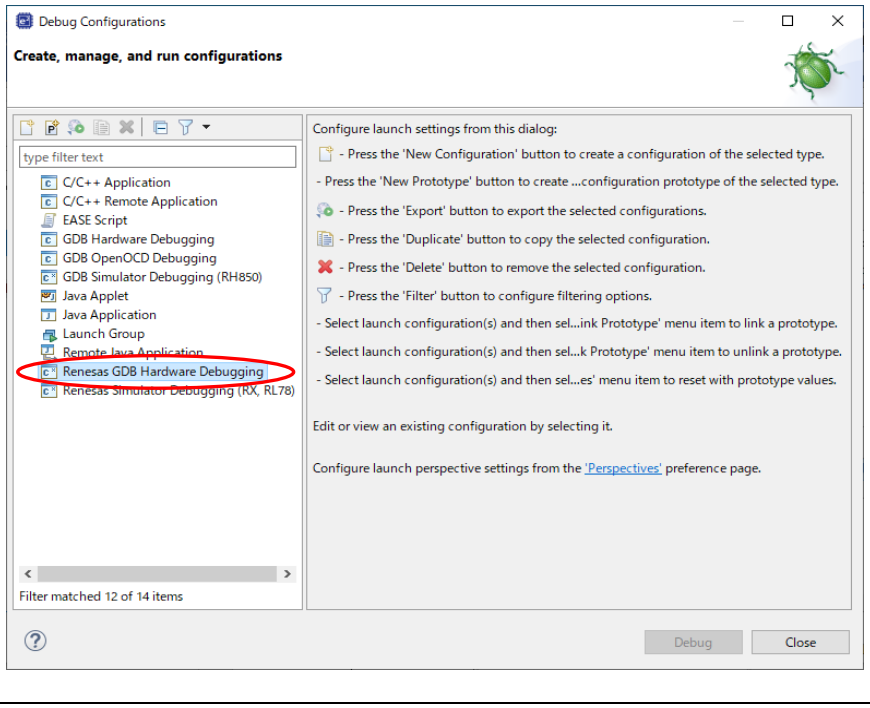


Fig.4-10 Debug Configurations(1)

b. Select Target Device

Click on the Debugger tag in the displayed dialog and select Target Device.

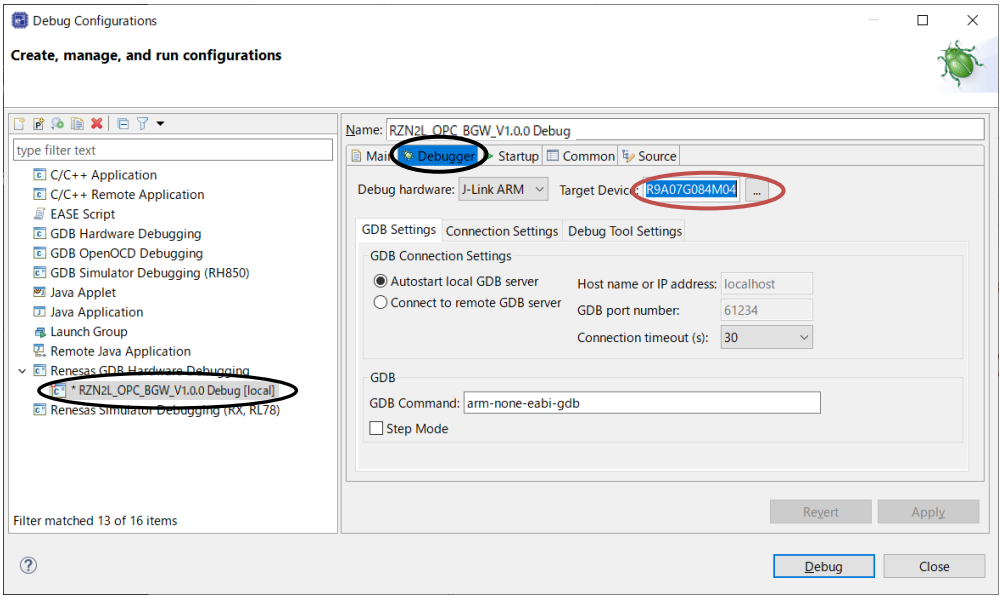


Fig.4-11 Debug Configurations(2)

Select R9A07G084M04 and click OK.

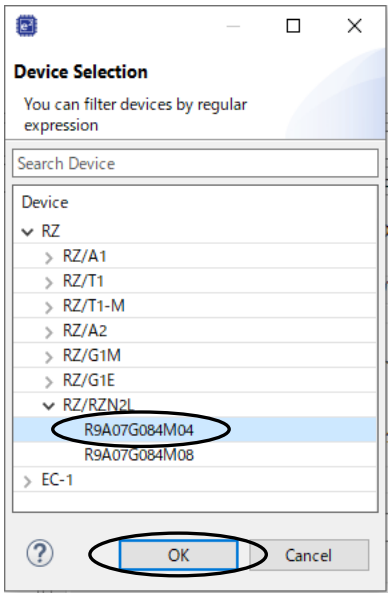


Fig.4-12 Debug Configurations(3)

c. Debug Tool Settings

Click the Debut Tool Settings tag and write 400 at Operating Frequency [MHz].

Click on "Debug" to start the download. Continue to Fig.4-16 for instructions.

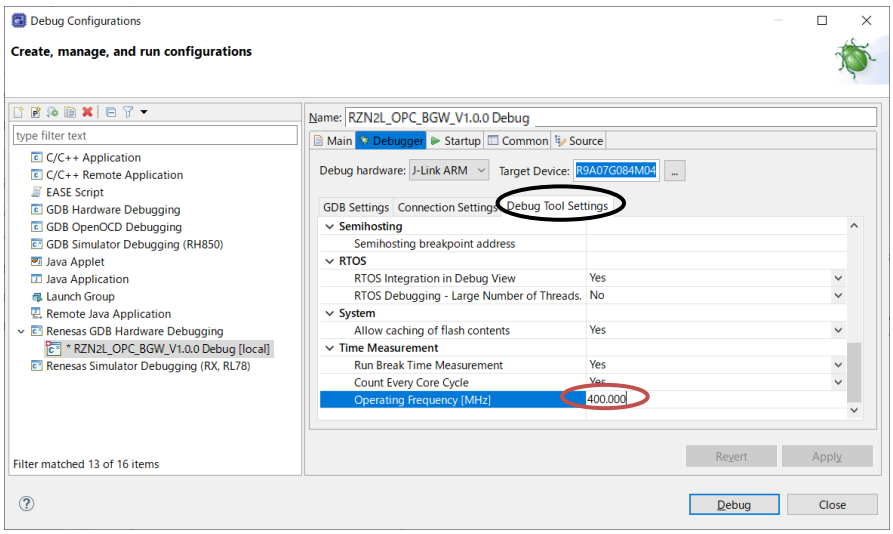


Fig.4-13 Debug Configurations(4)

4.3.4 Debug

The download procedure after completing the build is shown below.

At the second and subsequent debugger launches, click the Run menu with the project name selected in the C/C++ view. Place the cursor on "Debug As" and click on "Renesas GDB Hardware Debugging".

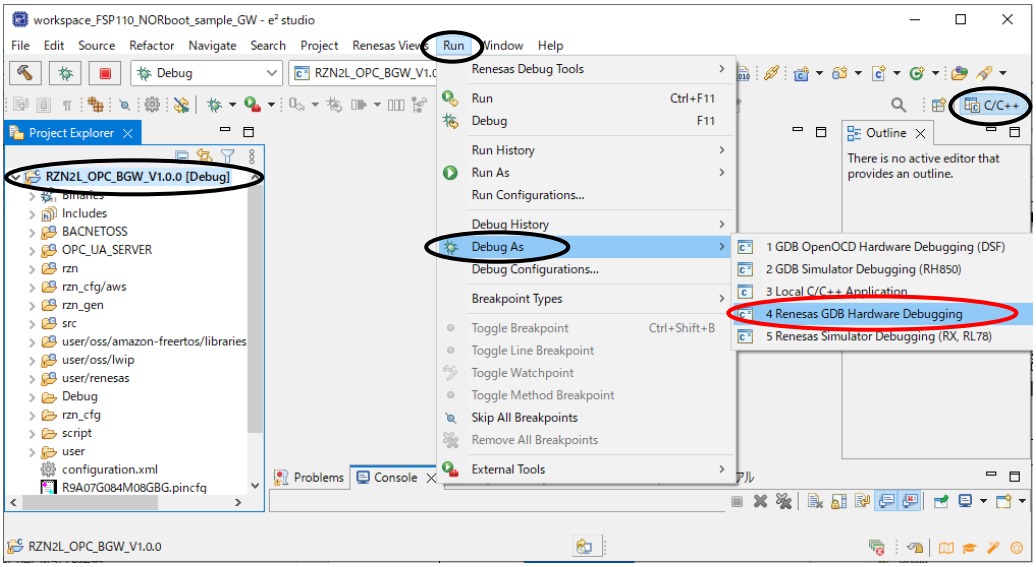


Fig.4-14 Run menu Debug As

Download the program to NOR flash memory. (It will take a few minutes.)

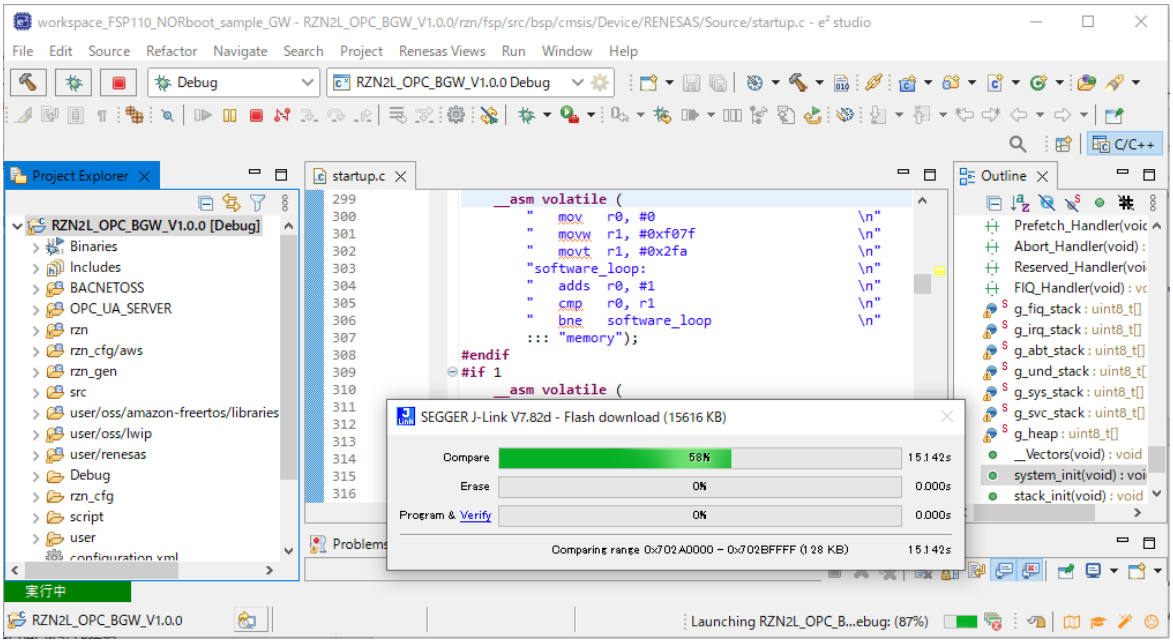


Fig.4-15 Download

Click Switch to change to debug view.

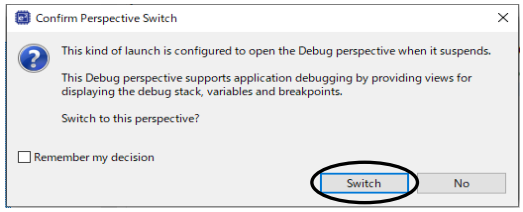


Fig.4-16 Perspective Switch

The CPU automatically extracts the loader program included in the download data to the BTCM. After extraction, it breaks in system_init() at the beginning of the initialization on the loader program.

- In case of operating the RSK board alone without using the debugger, turn off the board power supply, disconnect the debugger cable, and then turn on the board power supply again.

When using the debugger, **click the "reset" icon and then "resume"** after switching to the Debug screen.

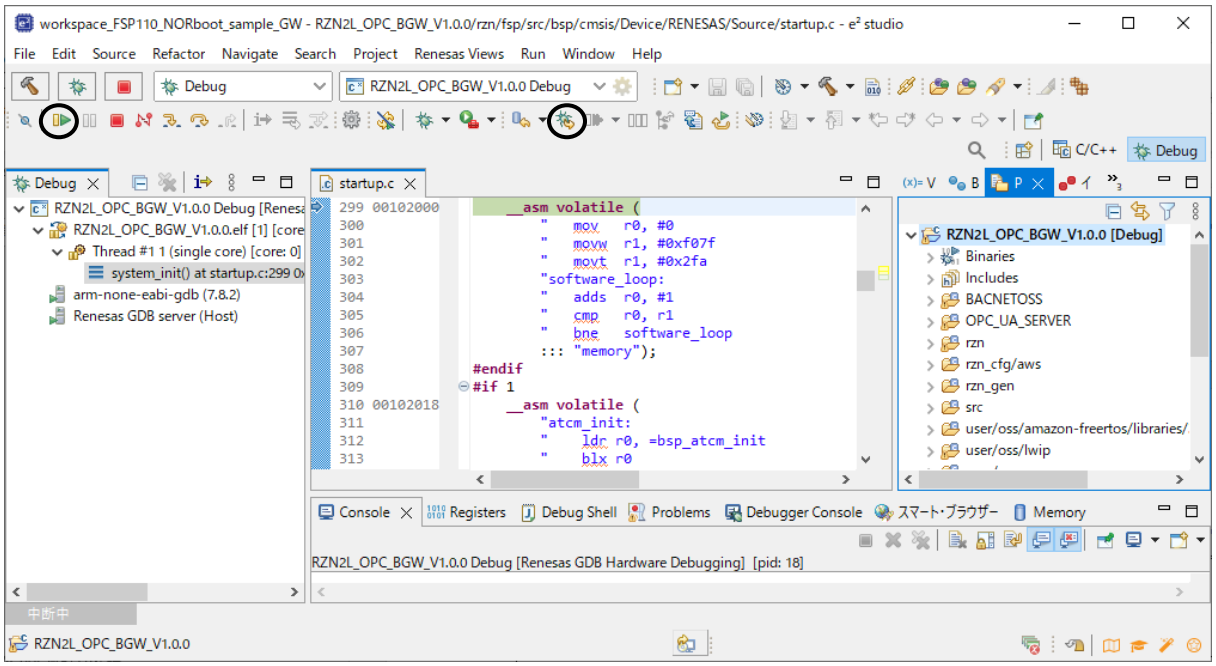


Fig.4-17 Break at system_init()

After completing initialization, the loader program stops at the beginning of main(), Then, click "resume"  to return to the running state.

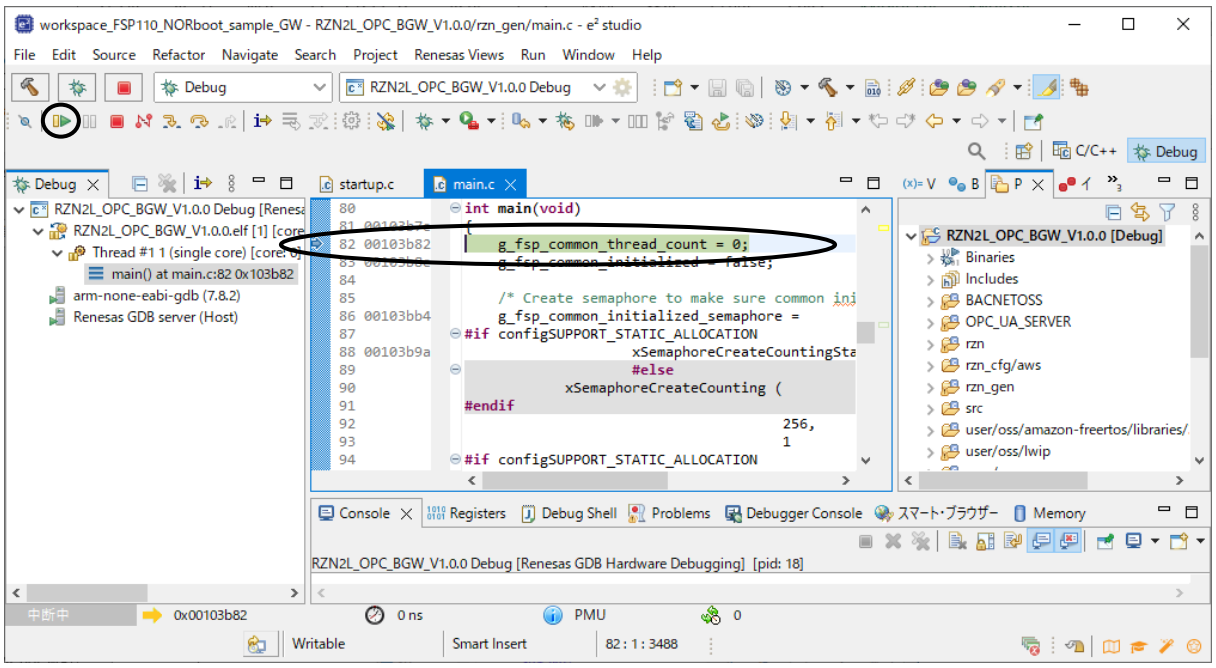


Fig.4-18 Break at main()

4.4 BACnet to OPC UA Gateway Communication

- Launch UaExpert

Open Windows Start menu and click UaExpert

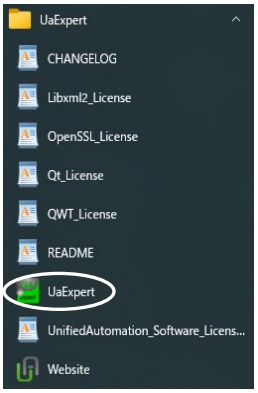


Fig.4-19 Launch UaExpert

- Add OPC UA server

Click on the tool bar in UaExpert.

Open the Advanced tab, set the Endpoint Url to "opc.tcp://192.168.10.100:4840", select Anonymous. Check "Connect Automatically" and then click OK at the end.

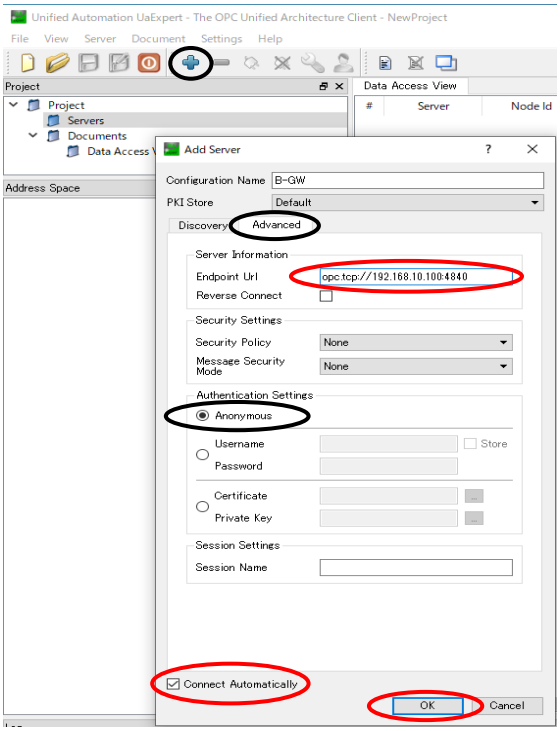


Fig.4-20 UaExpert Add server

When the OPC UA server, namely the B-GW, is connected, an indicator icon is displayed in the Project window to show that the B-GW is connected. BACnet-Client-Mapping displayed under the Object tree in the Address Space window is an object of B-GW.

An object called BACnet-Server-Mapping also appears below it, accessing the object node of the BACnet device connected to the B-GW.

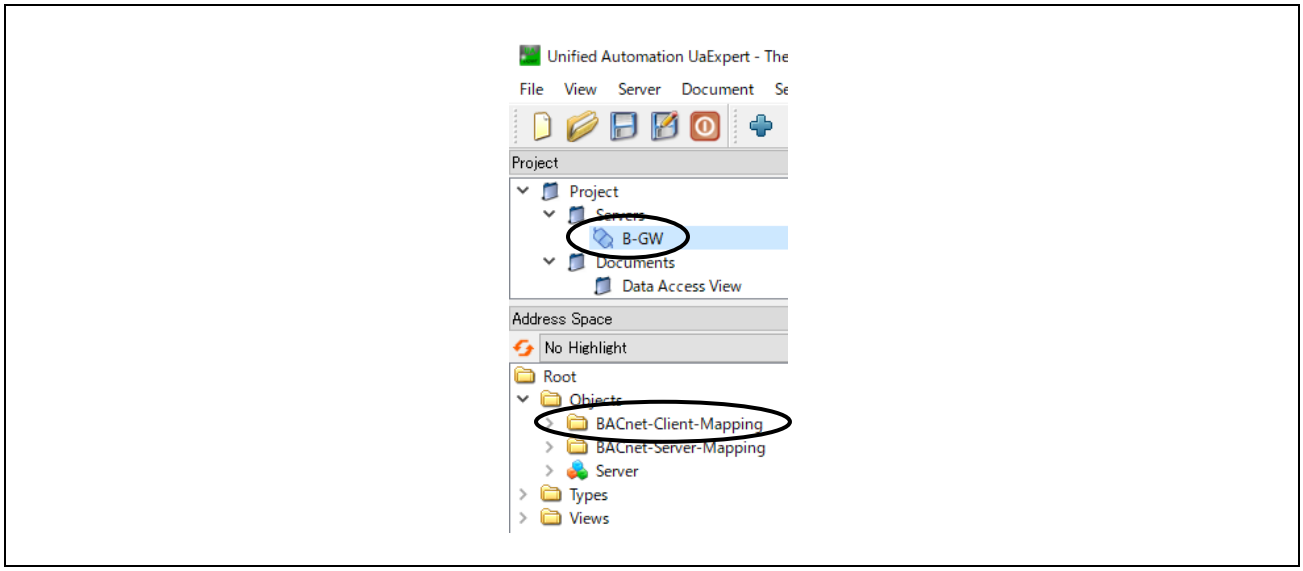


Fig.4-21 UaExpert OPC UA server connection

4.4.1 TimeSynchronization Method

The TimeSynchronization method sets the UTC time to the B-GW. The setting time is applied to the timestamp internally in the B-GW. After correcting the received UTC time to the local time, it is forwarded to the BACnet server device with local broadcast.

Select *Root>Objects>BACnet-Client-Mapping>OBJECT_INTERNETWORKTYPE>TimeSynchronization* in the Address Space window, right-click and select “Call...”.

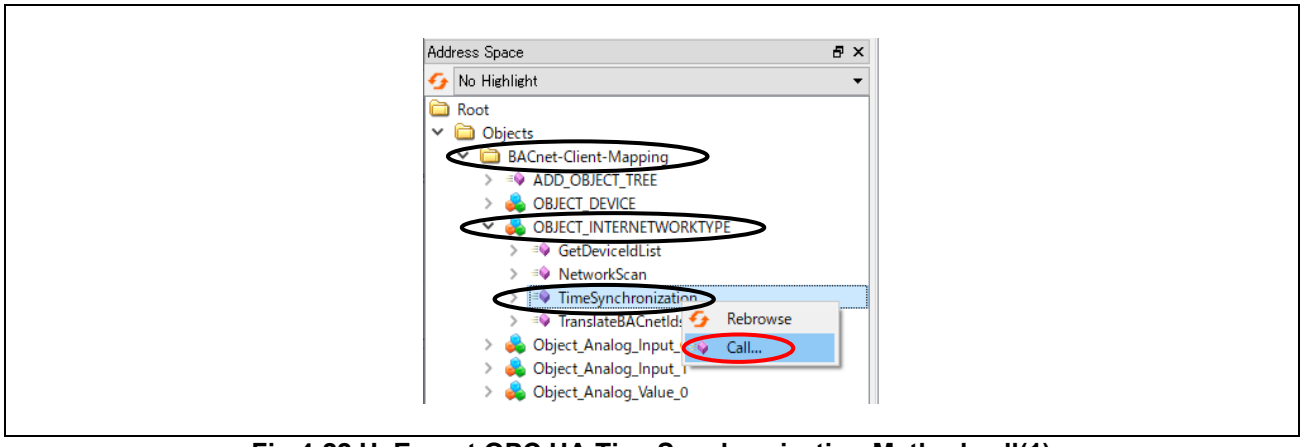


Fig.4-22 UaExpert OPC UA TimeSynchronization Method call(1)

Set the UTC time in the dialog displayed and click “Call”.

Correct the UTC time to the local time of your time zone. For example, in the case of TOKYO JAPAN, UTC time is the result of subtracting 9 hours from the local time.

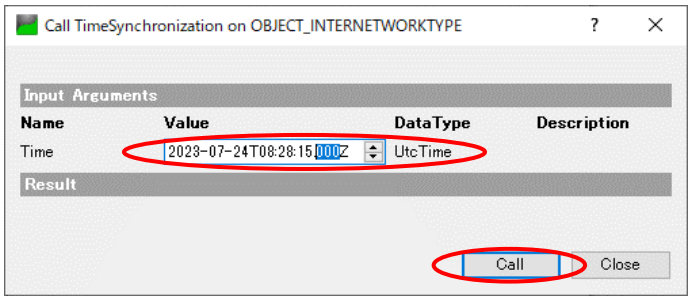


Fig.4-23 UaExpert OPC UA TimeSynchronization Method call(2)

Confirm that the method ends successfully and click “Close”.

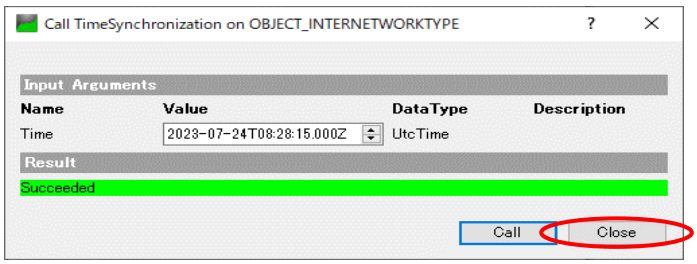


Fig.4-24 UaExpert OPC UA TimeSynchronization Method call(3)

The following wireshark log shows the above methods CallRequest and CallResResponse.

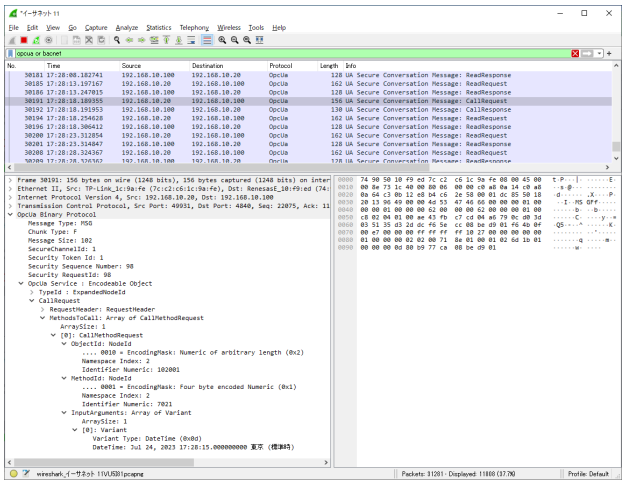


Fig.4-25 UaExpert OPC UA TimeSynchronization Method call(4)

4.4.2 NetworkScan Method

The NetworkScan method obtains IP addresses and device instance numbers for other devices connected to the network to which the B-GW is connected.

Select *Root>Objects>BACnet-Client-Mapping>OBJECT_INTERNETWORKTYPE> NetworkScan* in the Address Space window, right-click and select “Call...”.

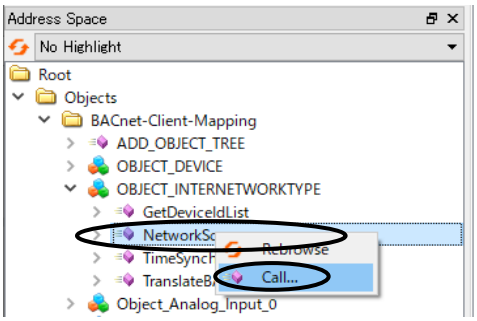


Fig.4-26 UaExpert OPC UA NetworkScan Method call(1)

Set the followings in the dialog that appears

・*Wait TimeInSeconds*：Set the I-Am response wait time from other devices in seconds.

・*ApplyRange*：Check to enable search range setting for connected devices. If disabled (unchecked), all device instances range 0~ 4194303 are searched.

・*DeviceRangeLow*：If search range is enabled, set the minimum instance number of the connected device.

・*DeviceRangeHigh*：If search range is enabled, set the maximum instance number of the connected device.

After setting the above, click ”Call”.

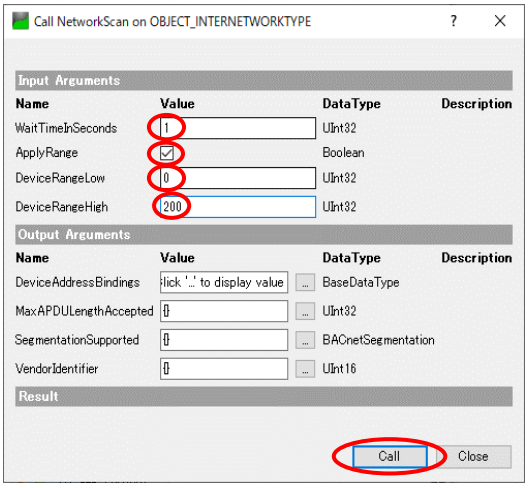


Fig.4-27 UaExpert OPC UA NetworkScan Method call(2)

Confirm the method completes successfully and click on [...] of DeviceAddressBindings in Output Arguments.
This example shows B-SS detected, connected device IP: 192.168.10.101, device instance number: 100.

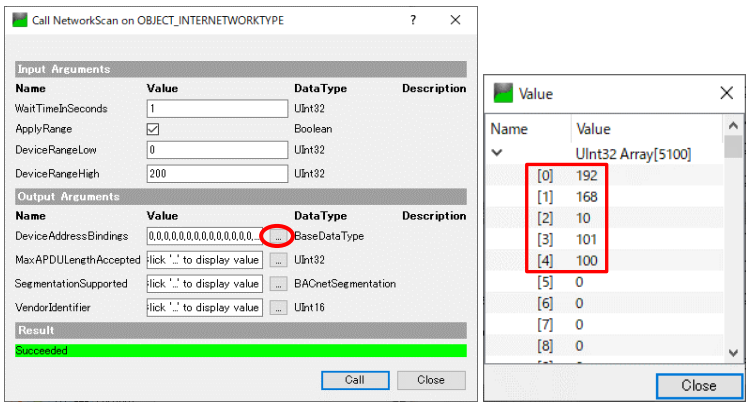


Fig.4-28 UaExpert OPC UA NetworkScan Method call(3)

The following wireshark log shows the above methods CallRequest and CallResResponse.

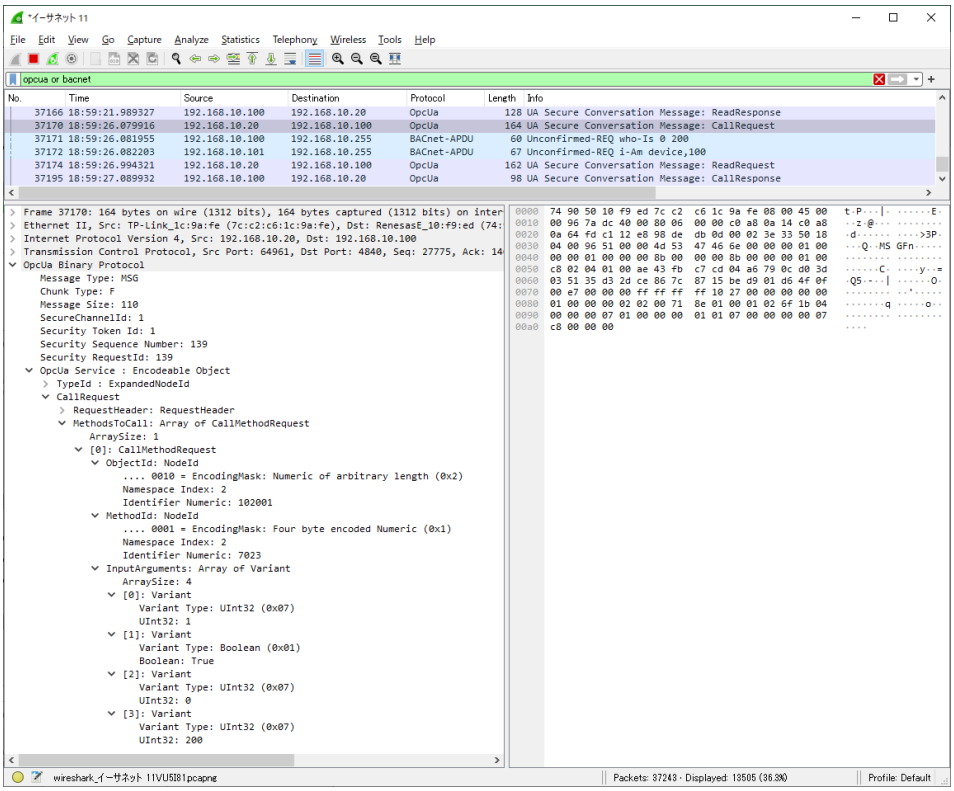


Fig.4-29 UaExpert OPC UA TimeSynchronization Method call(4)

4.4.3 Write property Method

Write property method changes the property values of the B-SS device object connected over BACnet.

Select *Root>Objects>BACnet-Client-Mapping> Write property* in the Address Space window, right-click and select “Call...”.

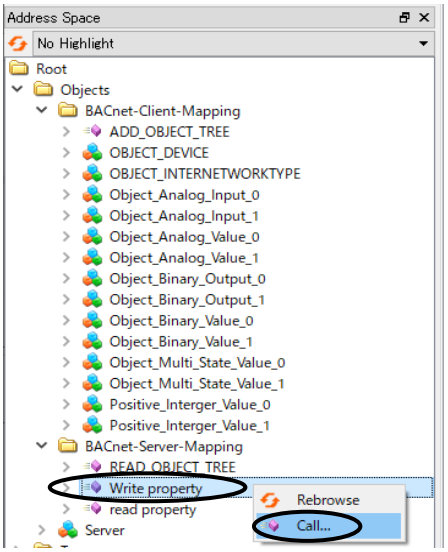


Fig.4-30 UaExpert OPC UA Write property Method call(1)

Set the followings in the dialog displayed.

Table 4-1 Write property Method Input Arguments(1)

Input Arguments	Property	Object_Type					
		AnalogInput	AnalogValue	BinaryOutput	BinaryValue	MultiStateValue	PositiveIntegerValue
DEVICE_ID		100					
OBJECT_TYPE		0	2	4	5	19	48
OBJECT_INSTANCE		0 or 1	0 or 1	0 or 1 or 2 or 3	0 or 1	0 or 1	0 or 1
PROPERTY_ID	Present_Value	85	85	85	85	85	85
PRIORITY		1~16					
TAG		4	4	9	9	2	2
OBJECT_VALUE		0.0~	0.0~	0 or 1	0 or 1	1 or 2 or 3	0~4294967295

Table 4-2 Write property Method Input Arguments(2)

Input Arguments	Property	Object_Type	
		Device	
DEVICE_ID		100	
OBJECT_TYPE		8	
OBJECT_INSTANCE		100	
PROPERTY_ID	Apdu_Timeout	11	
	Number_Of_Apdu_Retries		73
PRIORITY		1~16	
TAG		2	
OBJECT_VALUE (Recommended value)		1~ (6000)	0~ (3)

DEVICE_ID：Device instance number of the B-SS.

OBJECT_TYPE：The input is the value defined in *BACNETOSS\bacnet\bacenum.h*

OBJECT_INSTANCE：Instance number of each object.

PROPERTY_ID：The input is a value defined in *BACNETOSS\bacnet\bacenum.h*

PRIORITY：The priority for writing the same property across multiple clients, where 16 is the lowest priority and 1 is the highest priority.

TAG：Data type of the property value defined in *BACNETOSS\bacnet\bacenum.h*

OBJECT_VALUE：The set value for the property.

The following example writes B-SS device 100, AnalogOutput,0 object, Present_Value property, priority 16, data type Enumerated, and set value Active(1).

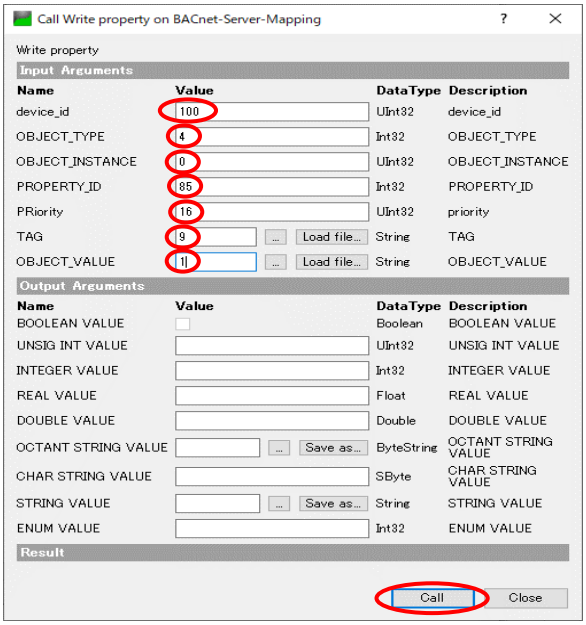


Fig.4-31 UaExpert OPC UA Write property Method call(2)

Check for successful completion and Output Arguments as follows

The readback value represents Active(1), which has the same data type Enumerated as the set value.

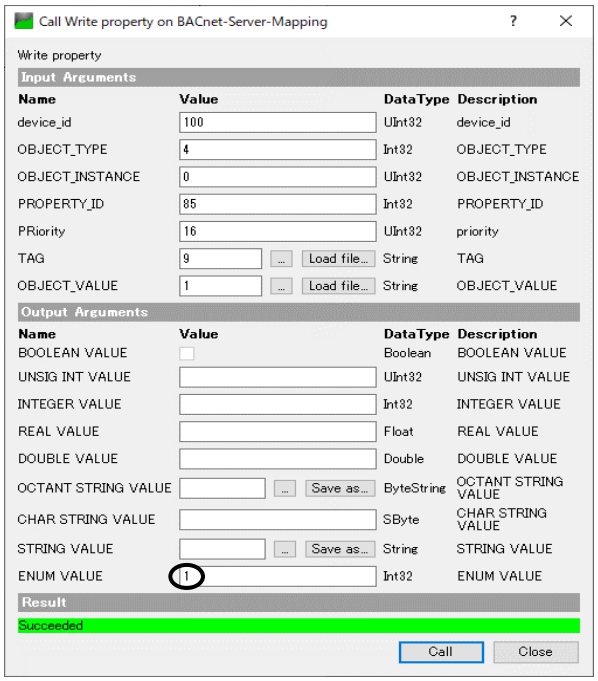


Fig.4-32 UaExpert OPC UA Write property Method call(3)

The following wireshark log shows the above methods CallRequest and CallResResponse.

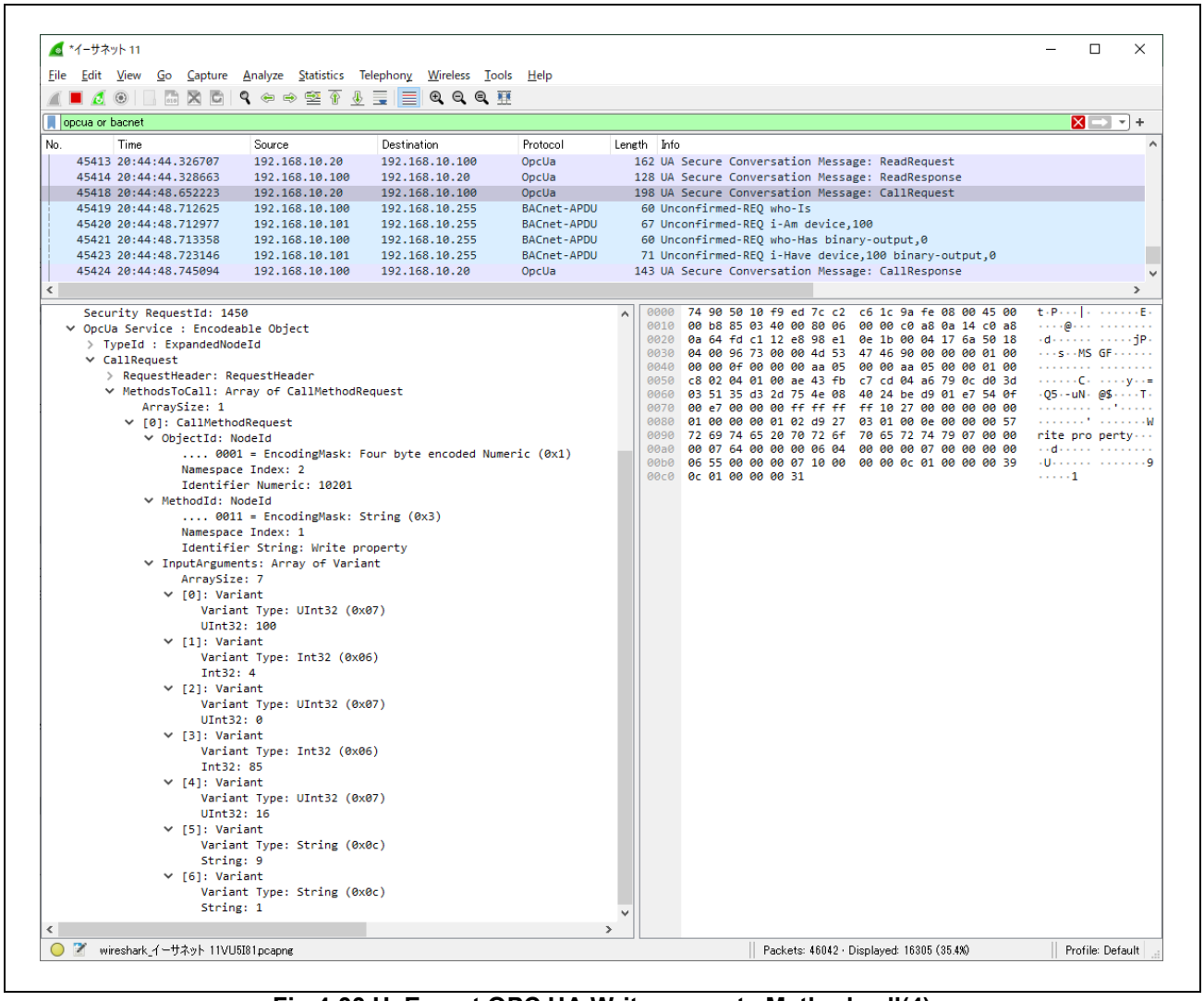


Fig.4-33 UaExpert OPC UA Write property Method call(4)

4.4.4 Read property Method

Read property method reads the property values of the B-SS device object connected over BACnet.

Select *Root>Objects>BACnet-Client-Mapping> Read property* in the Address Space window, right-click and select “Call...”.

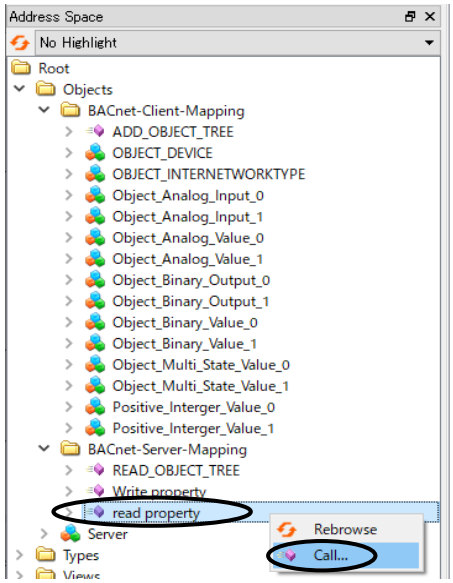


Fig.4-34 UaExpert OPC UA Read property Method call(1)

Set the followings in the dialog displayed.

Table 4-3 Reaed property Method Input Arguments(1)

Input Arguments	Property	Object_Type					
		AnalogInput	AnalogValue	BinaryOutput	BinaryValue	MultiStateValue	PositiveIntegerValue
DEVICE_ID		100					
OBJECT_TYPE		0	2	4	5	19	48
OBJECT_INSTANCE		0 or 1	0 or 1	0 or 1 or 2 or 3	0 or 1	0 or 1	0 or 1
PROPERTY_ID	Present_Value	85	85	85	85	85	85

Table 4-4 Reaed property Method Input Arguments(2)

Input Arguments	Property	Object_Type	
		Device	
DEVICE_ID		100	
OBJECT_TYPE		8	
OBJECT_INSTANCE		100	
PROPERTY_ID	Apdu_Timeout	11	
	Number_Of_Apdu_Retries		73

DEVICE_ID：Device instance number of the B-SS.

OBJECT_TYPE：The input is the value defined in *BACNETOSS\bacnet\bacenum.h*

OBJECT_INSTANCE：Instance number of each object.

PROPERTY_ID：The input is a value defined in *BACNETOSS\bacnet\bacenum.h*

The following example reads B-SS device 100, AnalogInput,0 object, and Present_Value property.

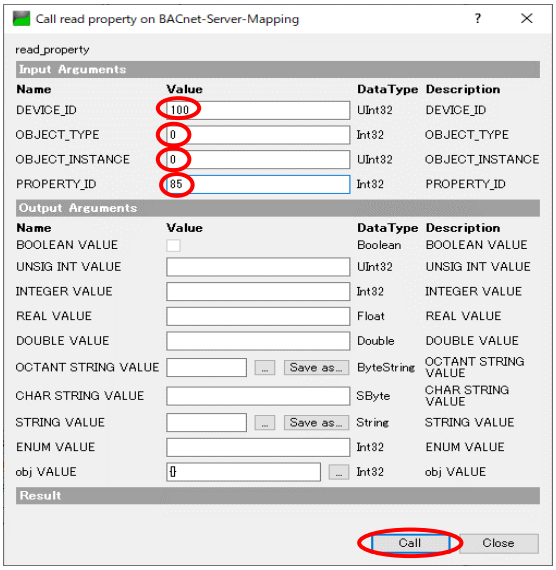


Fig.4-35 UaExpert OPC UA Read property Method call(2)

Check for successful completion and Output Arguments as follows.

The readback value represents a read value of the same data type REAL_VALUE (float) as the set value.

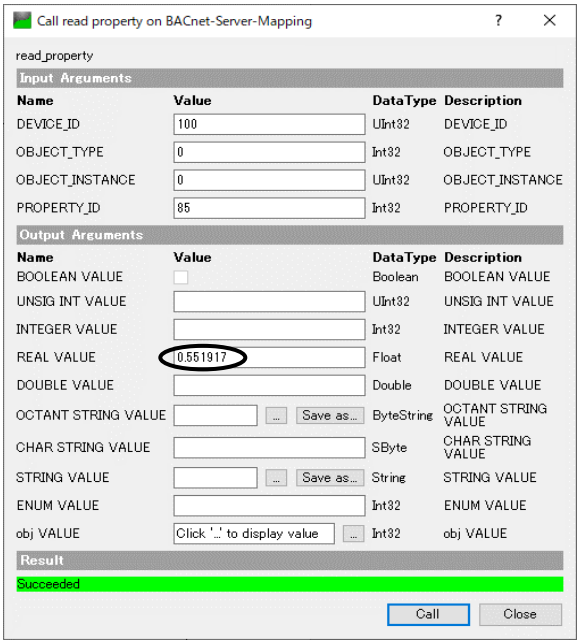


Fig.4-36 UaExpert OPC UA Read property Method call(3)

The following wireshark log shows the above methods CallRequest and CallResResponse.

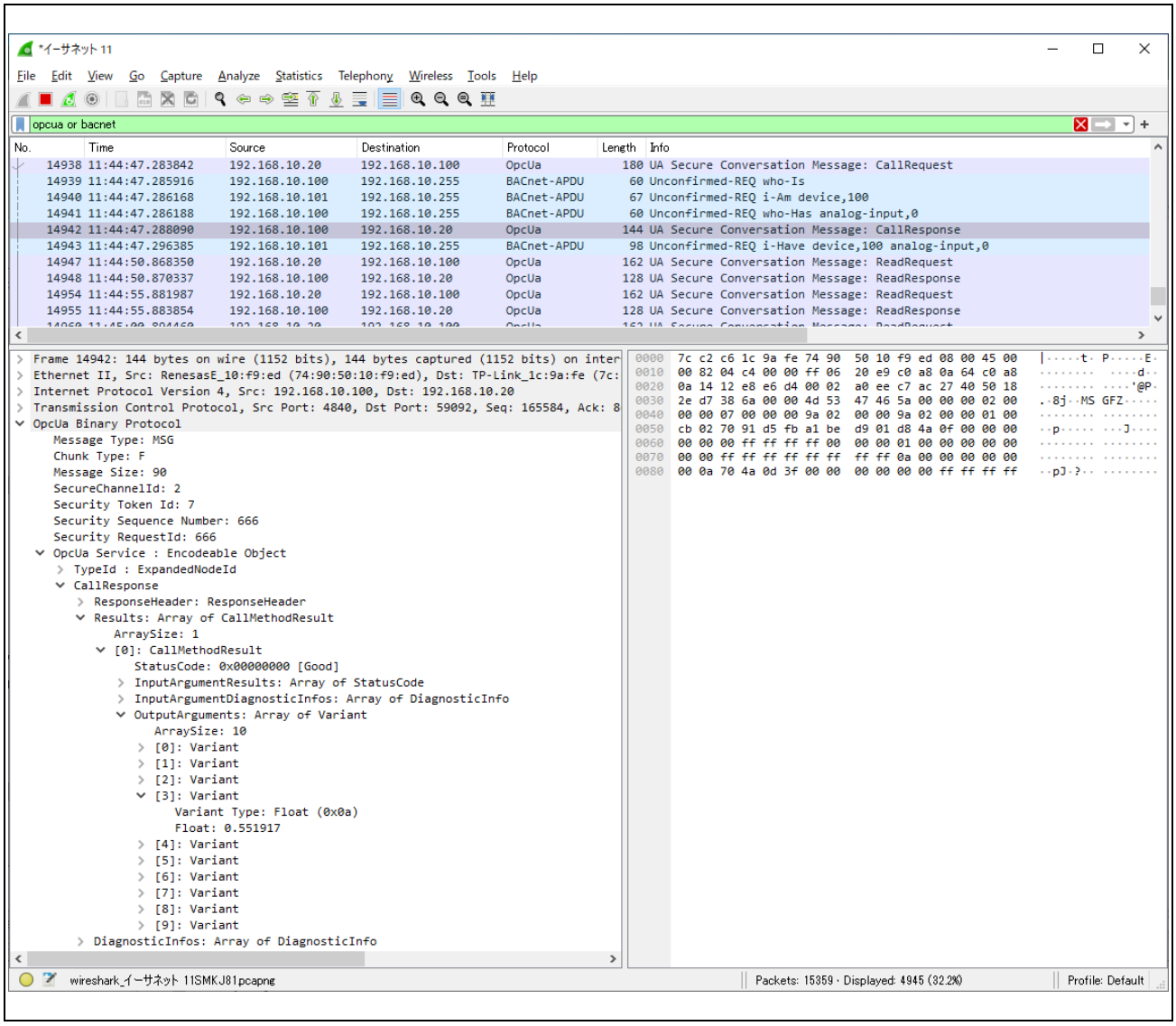


Fig.4-37 UaExpert OPC UA Read property Method call(4)

4.4.5 ADD/READ_OBJECT_TREE Method

The ADD_OBJECT_TREE method creates an object tree for the B-SS device.

Add target objects to the tree in order to read multiple objects at once (instead of reading them out one by one like the Read property method).

The object types to be read in batch in this version are as follows

AnalogInput, BinaryOutput, MultiStateValue AnalogValue, BinaryInput, PositiveIntegerValue

<Restrictions>

The following object types are not supported for batch read in this version

Device, AnalogOutput, BinaryValue

Select *Root>Objects>BACnet-Client-Mapping> ADD_OBJECT_TREE* in the Address Space window, right-click and select “Call...”.

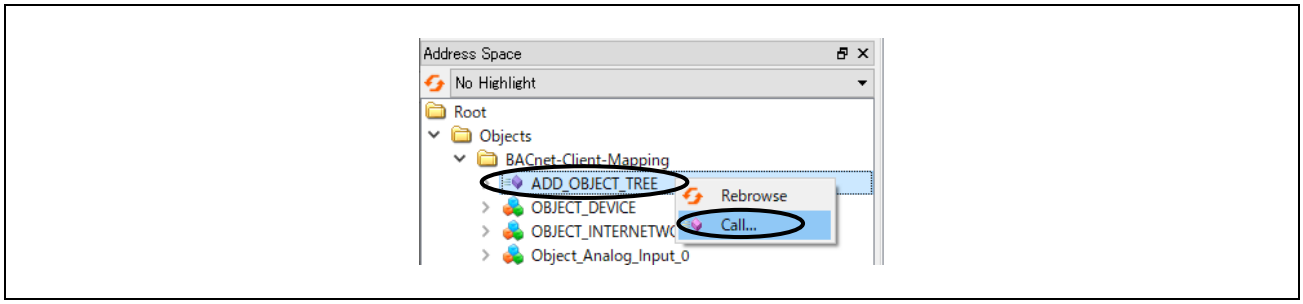


Fig.4-38 UaExpert OPC UA ADD_OBJECT_TREE Method call(1)

Set the followings in the dialog displayed, then click "Call".

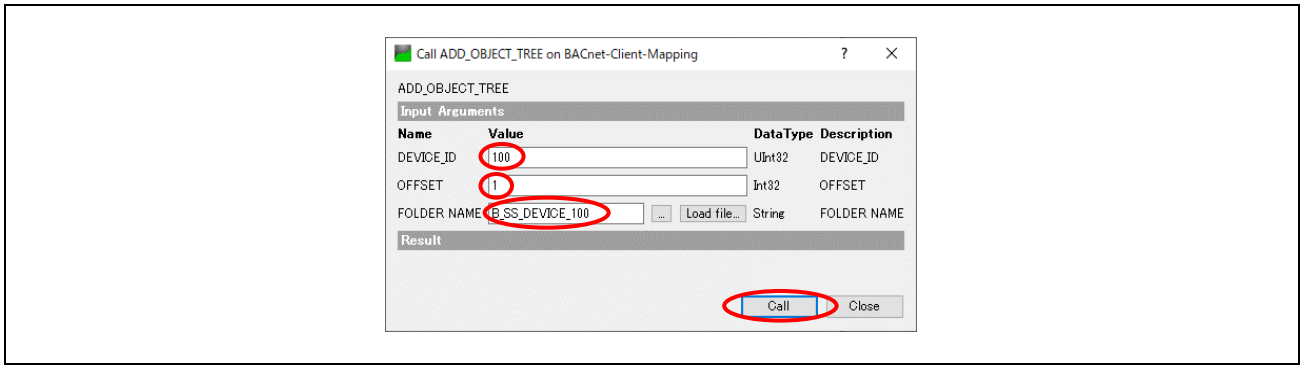


Fig.4-39 UaExpert OPC UA ADD_OBJECT_TREE Method call(2)

DEVICE_ID： Device instance number of the B-SS targeted. In the example, 100 is set.

OFFSET：A value for internal use of B-GW. 1~255 to be selected. In the example, 1 is set.

FOLDER_NAME：Name of the B-SS object tree. In the example, it is set to B_SS_DEVICE_100.

Select *Root>Objects>BACnet-Server-Mapping* in the Address Space window, right-click and select “Rebrowse”.

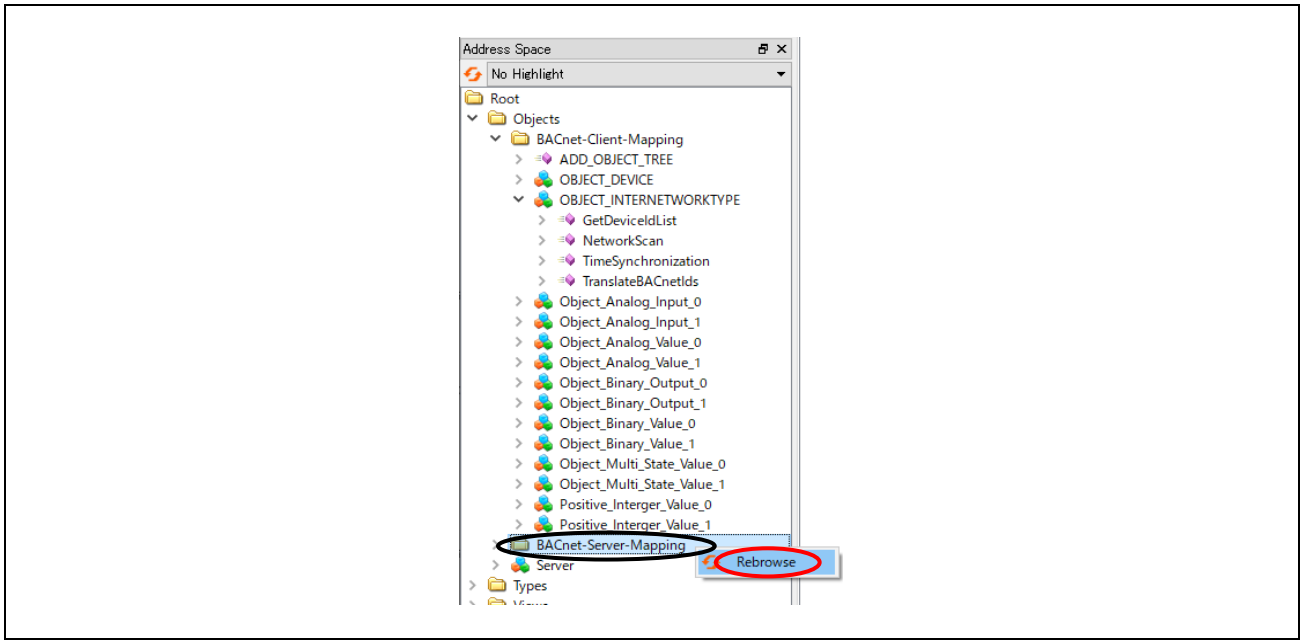


Fig.4-40 UaExpert OPC UA ADD_OBJECT_TREE Method call(3)

Expand *Root>Objects>BACnet-Server-Mapping* in the Address Space window to see the object tree that was added.

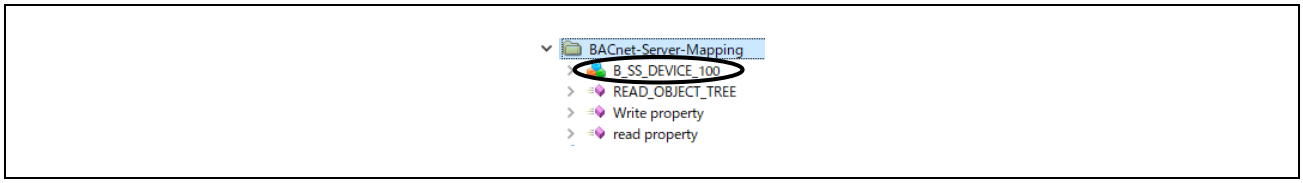


Fig.4-41 UaExpert OPC UA ADD_OBJECT_TREE Method call(4)

Expand the added object tree.

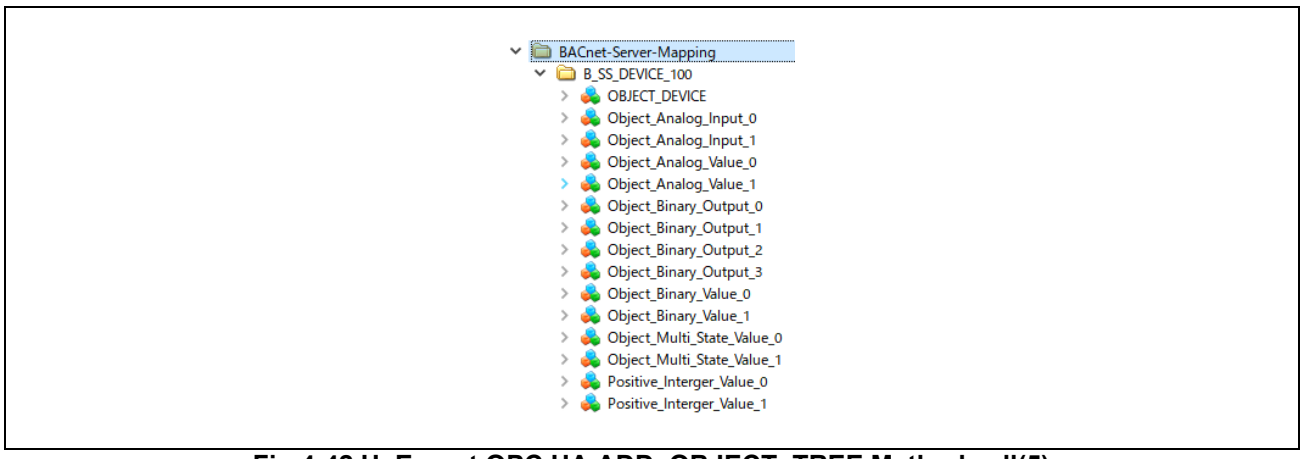


Fig.4-42 UaExpert OPC UA ADD_OBJECT_TREE Method call(5)

Expand each object in the object tree and drag and drop the Present_Value node into the Data Access View window. In the figure, only Object_Analog_Input_0 is expanded, but expand all objects and drag and drop each Present_Value. However, Device objects indicated as OBJECT_DEVICE are excluded.

The initial value of Present_Value for each object is displayed in the Value column of the Data Access View window.

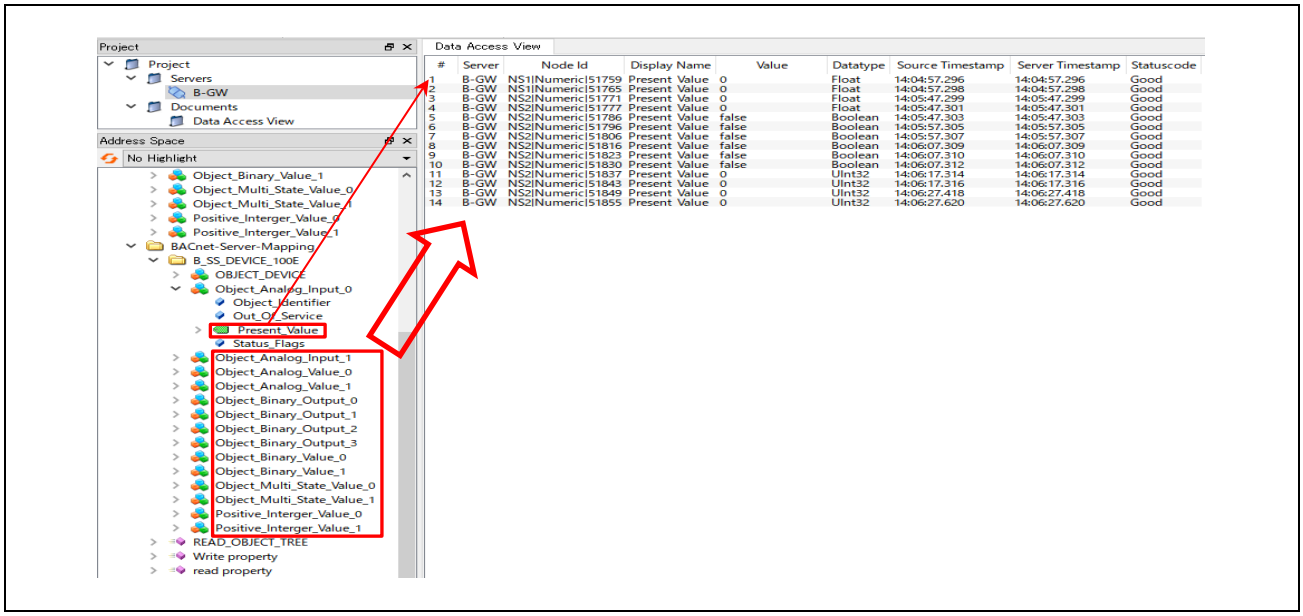


Fig.4-43 UaExpert OPC UA ADD_OBJECT_TREE Method call(6)

Call READ_OBJECT_TREE method.

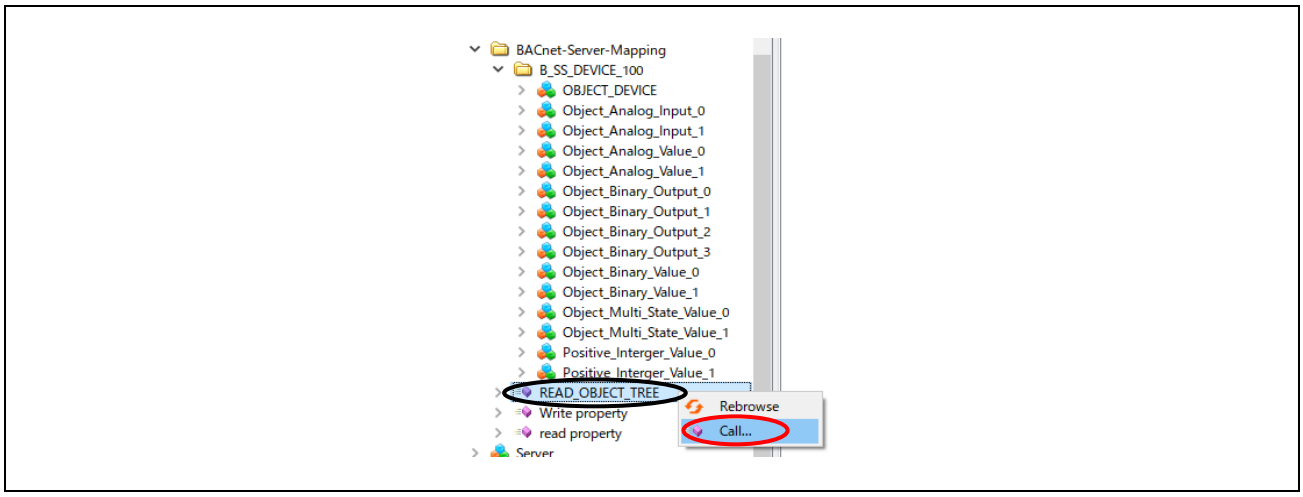


Fig.4-44 UaExpert OPC UA READ_OBJECT_TREE Method call(1)

Set 1 to the next dialog and click “Call”. The B-GW internally makes a ReadProperty service request to the B-SS, and the result of the response from the B-SS is reflected in the value in the Value column. Thus, the value is updated each time the READ_OBJECT_TREE method is called with read_object_tree=1 set. Since it does not automatically read the data in succession, call the READ_OBJECT_TREE method before reading the data.

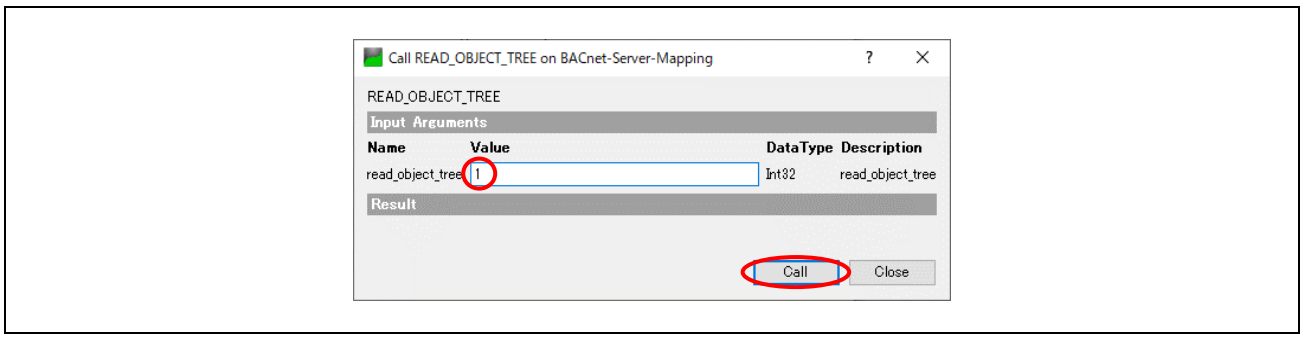
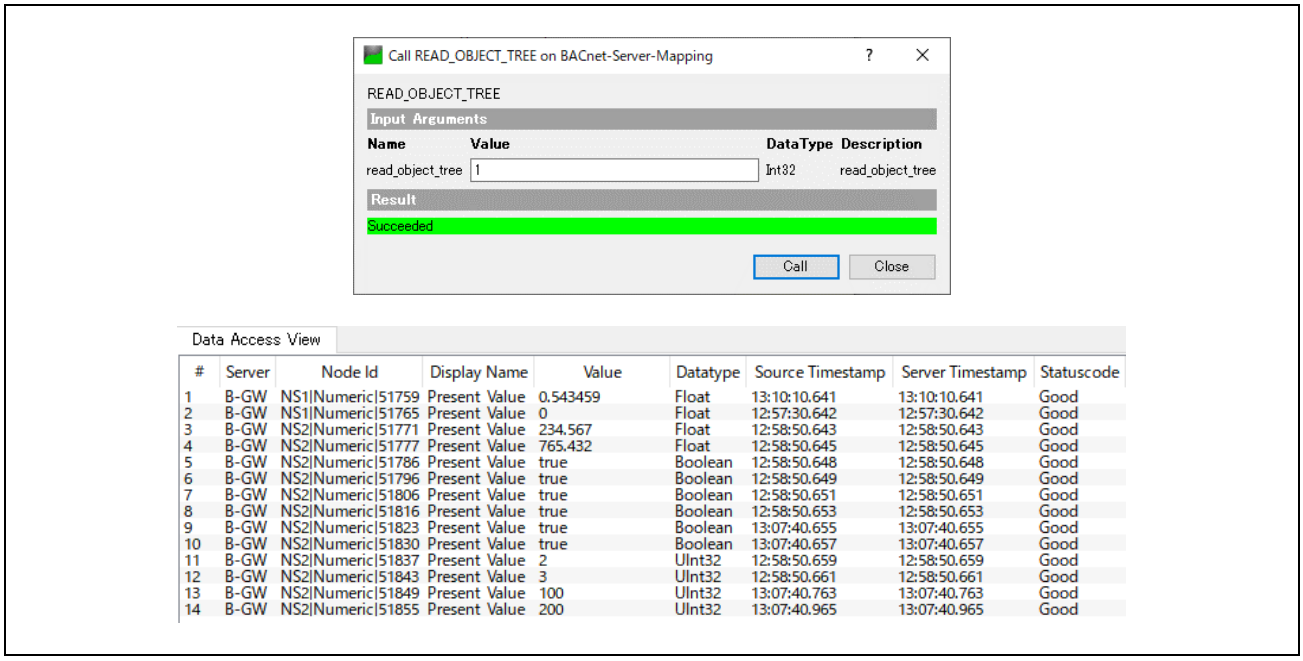


Fig.4-45 UaExpert OPC UA READ_OBJECT_TREE Method call(2)

After the B-SS object has been successfully read, check the Value column in the Data Access View window.



#	Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Statuscode
1	B-GW	NS1|Numeric|51759	Present Value	0.543459	Float	13:10:10.641	13:10:10.641	Good
2	B-GW	NS1|Numeric|51765	Present Value	0	Float	12:57:30.642	12:57:30.642	Good
3	B-GW	NS2|Numeric|51771	Present Value	234.567	Float	12:58:50.643	12:58:50.643	Good
4	B-GW	NS2|Numeric|51777	Present Value	765.432	Float	12:58:50.645	12:58:50.645	Good
5	B-GW	NS2|Numeric|51786	Present Value	true	Boolean	12:58:50.648	12:58:50.648	Good
6	B-GW	NS2|Numeric|51796	Present Value	true	Boolean	12:58:50.649	12:58:50.649	Good
7	B-GW	NS2|Numeric|51806	Present Value	true	Boolean	12:58:50.651	12:58:50.651	Good
8	B-GW	NS2|Numeric|51816	Present Value	true	Boolean	12:58:50.653	12:58:50.653	Good
9	B-GW	NS2|Numeric|51823	Present Value	true	Boolean	13:07:40.655	13:07:40.655	Good
10	B-GW	NS2|Numeric|51830	Present Value	true	Boolean	13:07:40.657	13:07:40.657	Good
11	B-GW	NS2|Numeric|51837	Present Value	2	UInt32	12:58:50.659	12:58:50.659	Good
12	B-GW	NS2|Numeric|51843	Present Value	3	UInt32	12:58:50.661	12:58:50.661	Good
13	B-GW	NS2|Numeric|51849	Present Value	100	UInt32	13:07:40.763	13:07:40.763	Good
14	B-GW	NS2|Numeric|51855	Present Value	200	UInt32	13:07:40.965	13:07:40.965	Good

Fig.4-46 UaExpert OPC UA READ_OBJECT_TREE Method call(3)

4.5 Evaluation with a Single Board

This section describes the means to check OPC UA server operation with a single board without a B-SS board connected. See chapter 4.3.1(1)for the build procedure.

After building, execute the TimeSynchronization method described in section 4.4.1 at first.

Air velocity sensor input values that are originally read from the B-SS can be pseudo-generated inside the B-GW and read from the PresentValue node of the AnalogInput,0 object of the B-GW.

As shown in the following figure, drag and drop the *Root>Objects>BACnet-Client-Mapping>Object_Analog_Input_0>Present_Value* node in the Address Space window to the Data Access View window to change the value in the Value column.

The value repeatedly changes in the range of 0.0[m/sec]~7.23[m/sec] according to the air velocity sensor specification.

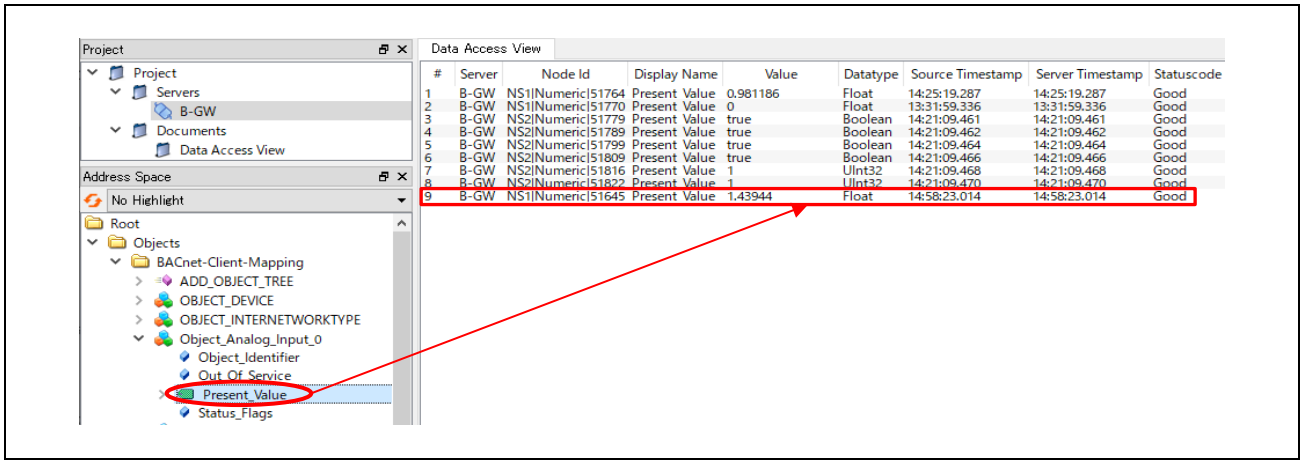


Fig.4-47 B-SS pseudo input value reading

5. Appendix

5.1 File Generation of open62541

The OPC UA stack of this sample software uses the open source open62541. To run Open62541 in a freeRTOS + LwIP environment, the following link recommends an approach to generate open62541.c and open62541.h using CMake, which is also used in this sample software.

Building open62541 — open62541 1.3.0-dirty documentation

This chapter describes the procedure for generating open62541 and BACnet information models as files for e2studio execution in a Windows 10 environment. Here Window 10 version 1903 or later (OS Build 19044.2965) is used, in which WSL2 is executable.

5.1.1 Linux environment Setup

Set up a Linux environment to run CMake. In this document, we will run CMake on a Linux (Ubuntu 18.04) environment installed using WSL2 with reference to the following linked pages.

(Reference) Manual installation steps for older versions of WSL | Microsoft Learn

1) Launch PowerShell as Administrator. Search PowerShell > right-click > Run as Administrator
2) Enter the following command to enable the Windows Subsystem for Linux.
   dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart

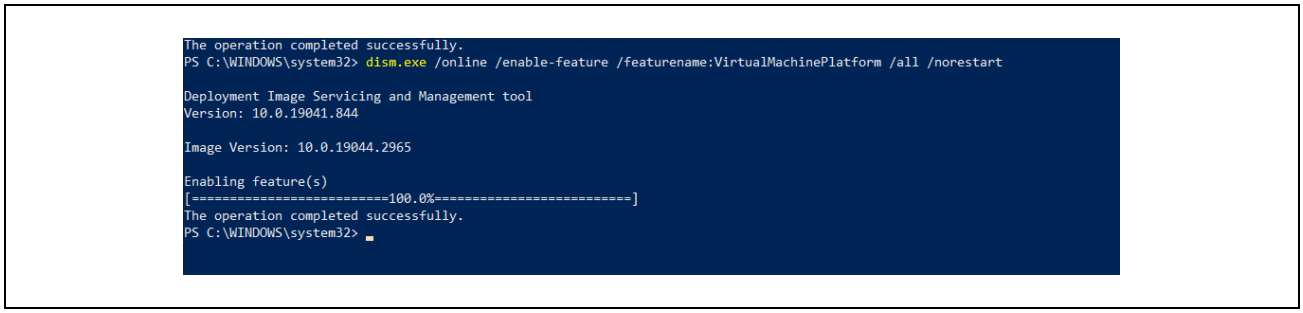


Fig.5-1 Microsoft-Windows-Subsystem-Linux

3) Enter the following command to enable the virtual machine platform feature:
   dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart

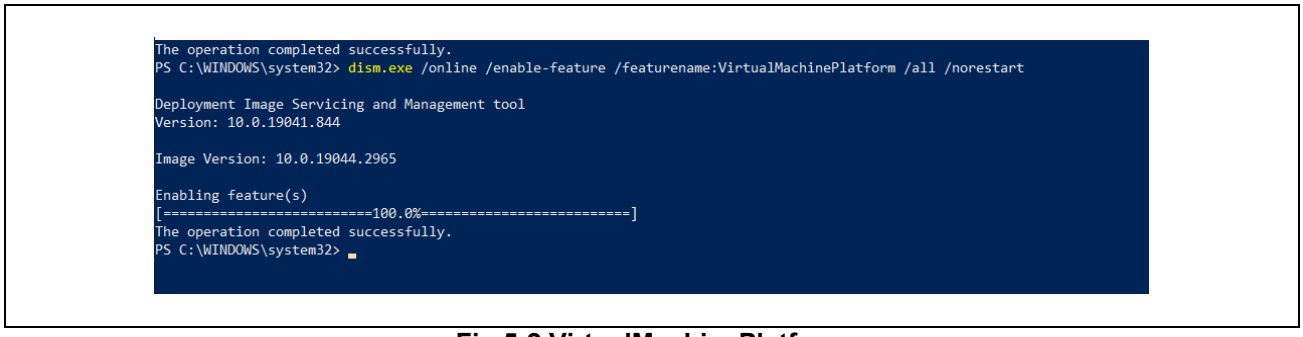


Fig.5-2 VirtualMachinePlatform

4) Restart your PC and complete the WSL installation.
5) Download and run the WSL2 Linux kernel update package for x64 machines below.
   WSL2 Linux kernel update package for x64 machines

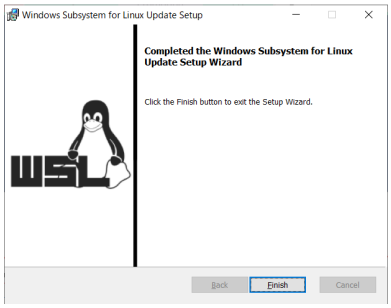


Fig.5-3 wsl_update_x64.msi

6) Run the following command to set WSL 2 as the default version.
wsl --set-default-version 2

7) Download Linux distribution. Here download Ubuntu 18.04 below.
Ubuntu 18.04

8) Go to the folder containing the downloaded file and execute the following command.
Add-AppxPackage .\app_name.appx

9) Double-click Ubuntu_1804.2019.522.0_x64.appx to install.

Fig.5-4 Ubuntu Install

10) Set the Linux username and password.
(Reference) Set up a WSL development environment | Microsoft Learn

Fig.5-5 UNIX username

5.1.2 Install CMake

11) Execute the following Linux command to update apt-get

sudo apt-get update

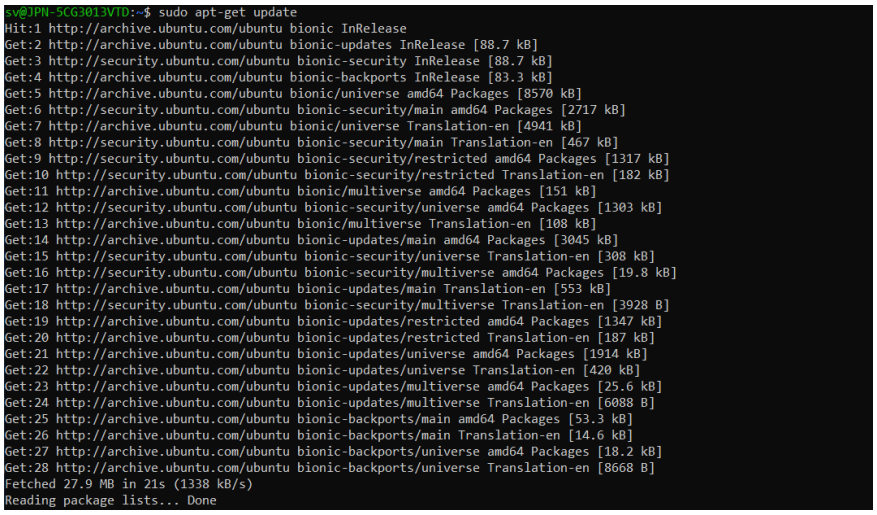


Fig.5-6 apt-get update

12) Execute the following Linux command

sudo apt-get install git build-essential gcc pkg-config cmake python

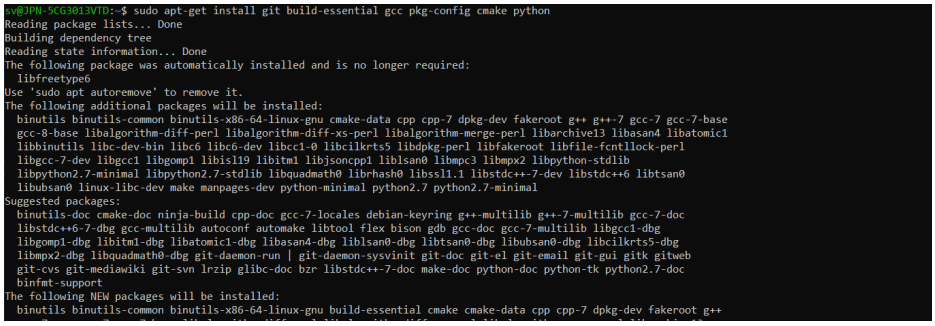


Fig.5-7 install

If the following screen appears during the process, select OK.

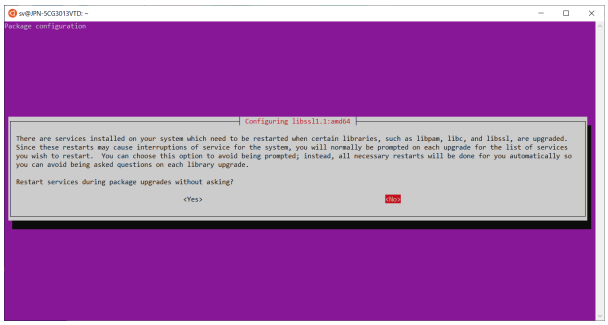


Fig.5-8 restart

13) Execute each of the following commands.

```
sudo apt-get install cmake-curses-gui          # Needed for CMAKE GUI
sudo apt-get install libmbedtls-dev            # For encryption
sudo apt-get install liburcu-dev               # For multithreading
sudo apt-get install check                     # For unit tests
sudo apt-get install python-sphinx graphviz    # For doc generation
sudo apt-get install python-sphinx-rtd-theme   # For doc's style
```

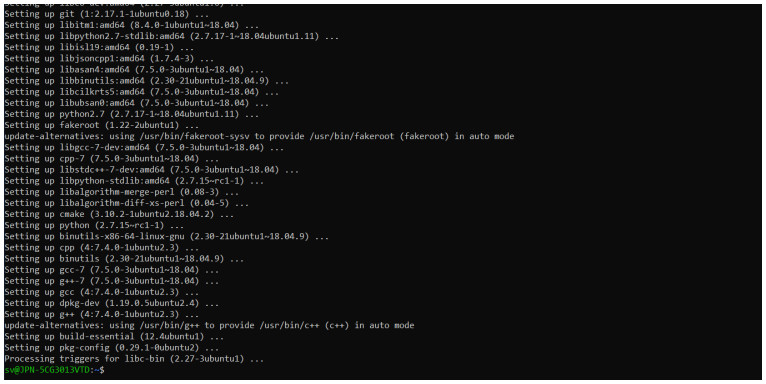


Fig.5-9 install

5.1.3 Open62541 File Generation

14) Clone open62541 to any folder

```
git clone https://github.com/open62541/open62541.git
```

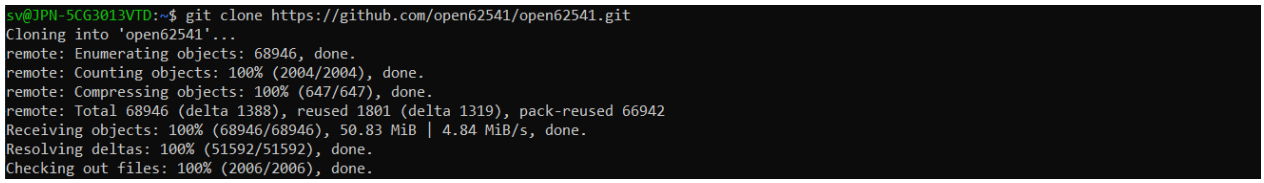


Fig.5-10 git clone

15) Go to /open62541 directory and check out the specific version (here, version v1.3.4-564-gb7e5e49f3).

```
git clone https://github.com/open62541/open62541.git
cd open62541/
git log -1
git checkout b7e5e49f32d00490be74c2eacef892c7fbd0be60
git submodule init
git submodule update
```

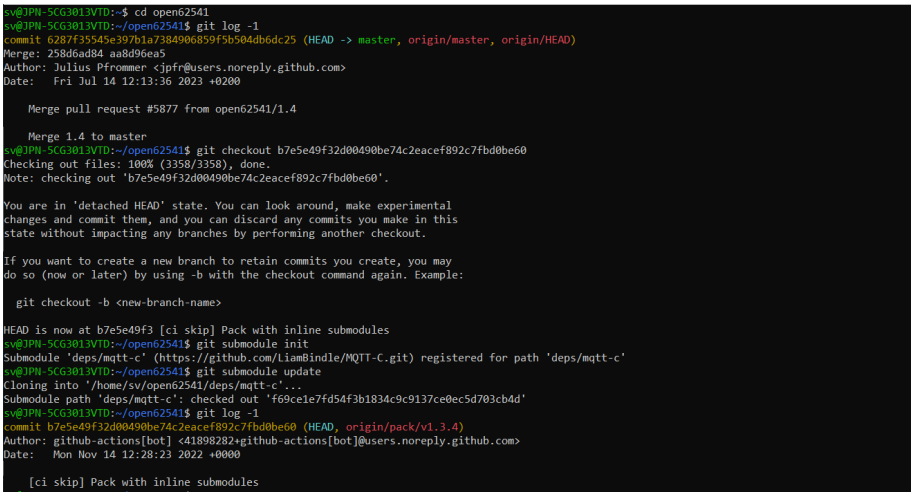


Fig.5-11 git submodule

16) Open the Linux folder from File Explorer. Confirm that CMakeLists.txt is present in */home/(username)/open62541* directory. Copy the following four patch files obtained by unzipping patch_open62541.zip attached to the sample software to this directory.

include-bacnet-xmls.patch

CMakeLists.txt.patch

Opc.Ua.NodeSet2.Reduced.xml.patch

datatypes_dataaccess.txt.patch

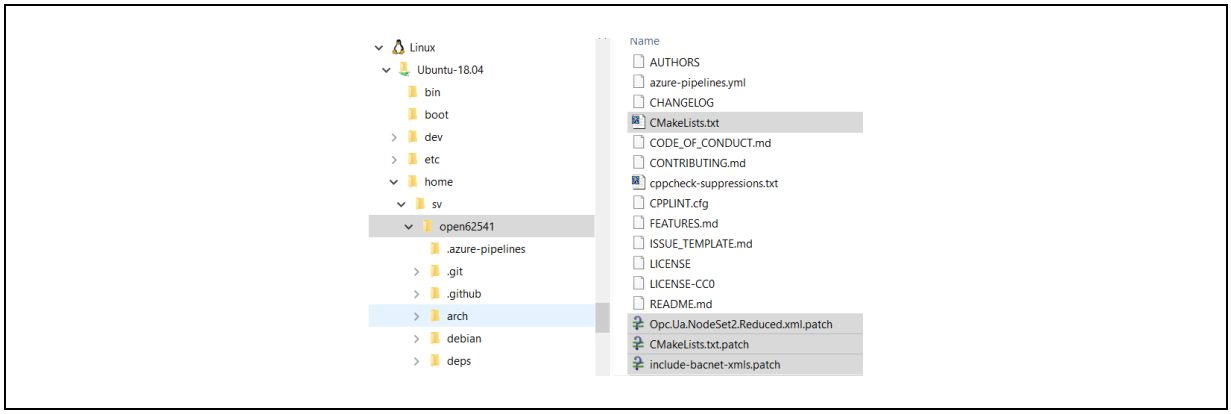


Fig.5-12 Copy patch files

17) Execute patch commands below in /open62541 directory

patch -p1 < include-bacnet-xmls.patch

patch -p1 < CMakeLists.txt.patch

patch -p1 < Opc.Ua.NodeSet2.Reduced.xml.patch

patch -p1 < datatypes_dataaccess.txt.patch

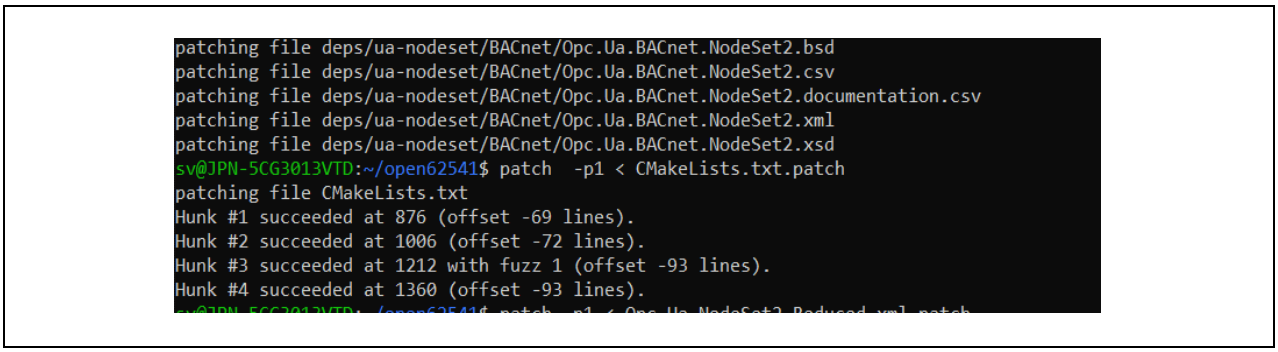


Fig.5-13 patch command

18) Compile the library according to the standard procedures of the cmake project. Create /open62541/build directory and run cmake . (Some items will be Failed, but there is no problem. (Some items will be Failed, but that is not a problem.)

mkdir build && cd build

cmake ..

Fig.5-14 cmake

19) Execute the following command to start the ccmake setting window.

ccmake ..

20) Change the settings as follows, and after executing [c] to configure, close it by [q] to quit without generating.

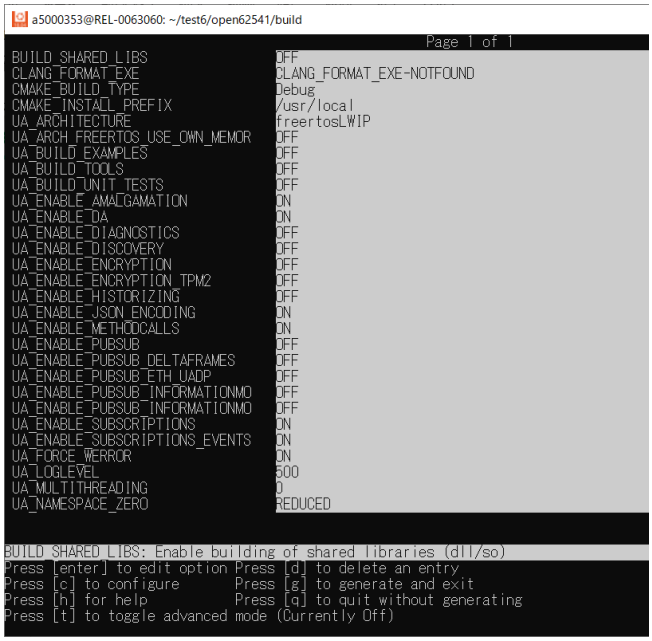


Fig.5-15 ccmake

21) Execute the following command to make in the */open62541/build* directory. The make process will finish with an error, but it does not matter.

```
make -j
```

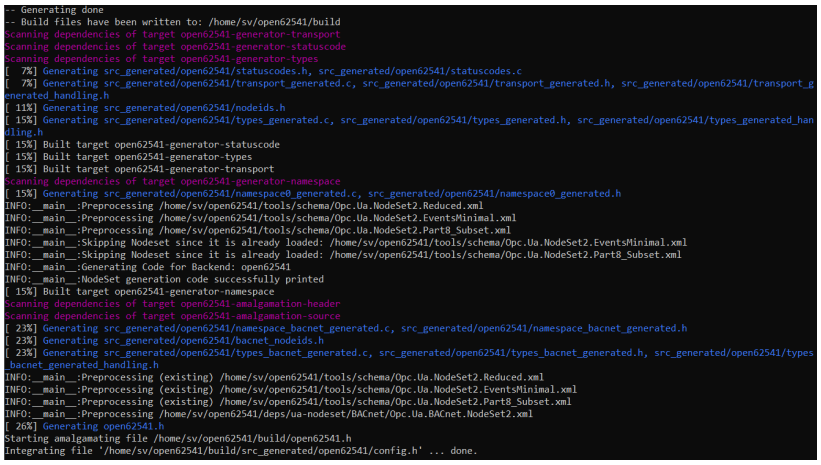


Fig.5-16 make

22) After "make -j" is completed, execute each of the following commands.

```
find ./ -type f -exec sed -i -e 's/fields\.unsigned\;/fields\.unsignedValue\;/g' {} \;
find ./ -type f -exec sed -i -e 's/fields\.boolean\;/fields\.booleanValue\;/g' {} \;
find ./ -type f -exec sed -i -e 's/fields\.enum\;/fields\.enumValue\;/g' {} \;
find ./ -type f -exec sed -i -e 's/fields\.enum)/fields\.enumValue)/g' {} \;
```

```
find ./ -type f -exec sed -i -e 's/fields\.boolean)/fields\.booleanValue)/g' {} \;
find ./ -type f -exec sed -i -e 's/fields\.unsigned)/fields\.unsignedValue)/g' {} \;
find ./ -type f -exec sed -i -e 's/fields\.signed)/fields\.signedValue)/g' {} \;
find ./ -type f -exec sed -i -e 's/enum\;/enumValue\;/g' {} \;
find ./ -type f -exec sed -i -e 's/boolean\;/booleanValue\;/g' {} \;
find ./ -type f -exec sed -i -e 's/unsigned\;/unsignedValue\;/g' {} \;
find ./ -type f -exec sed -i -e 's/signed\;/signedValue\;/g' {} \;
```

Fig.5-17 substitution command

23) Confirm that the following files are generated in /opn62541 and /src_generated/open62541 directories.

- *open62541.c*
- *open62541.h*
- *types_bacnet_generated.c*
- *types_bacnet_generated.h*
- *types_bacnet_generated_handling.h*
- *Namespace_bacnet_generated.h*
- *Namespace_bacnet_generated.c*
- *Bacnet_nodeids.h*

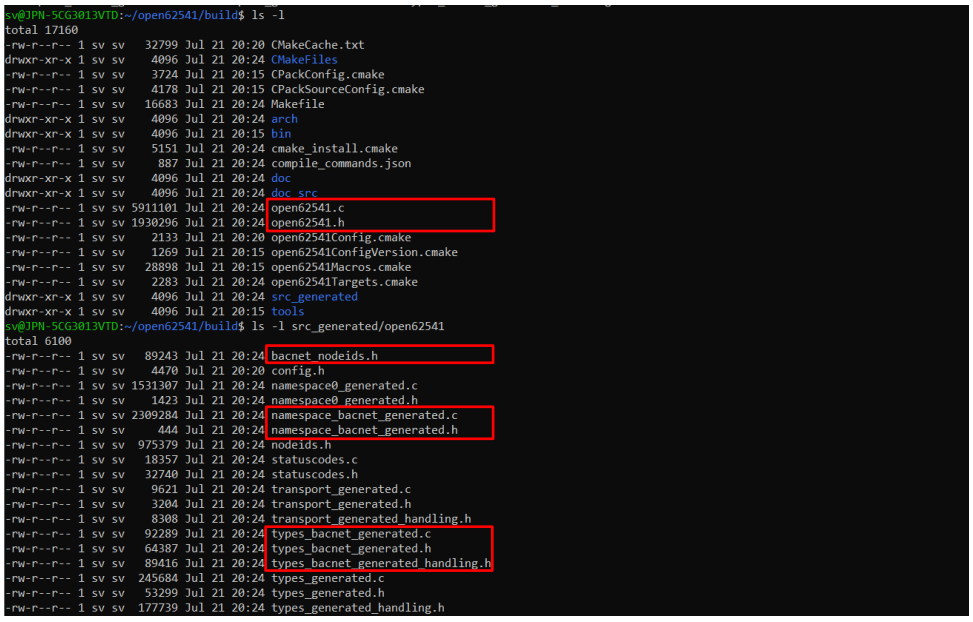


Fig.5-18 Generated Files

24) Create a folder in the IDE project and import the generated files into the project as shown in the figure. Where OPC_UA_SERVER is the file created under the e2studio project.

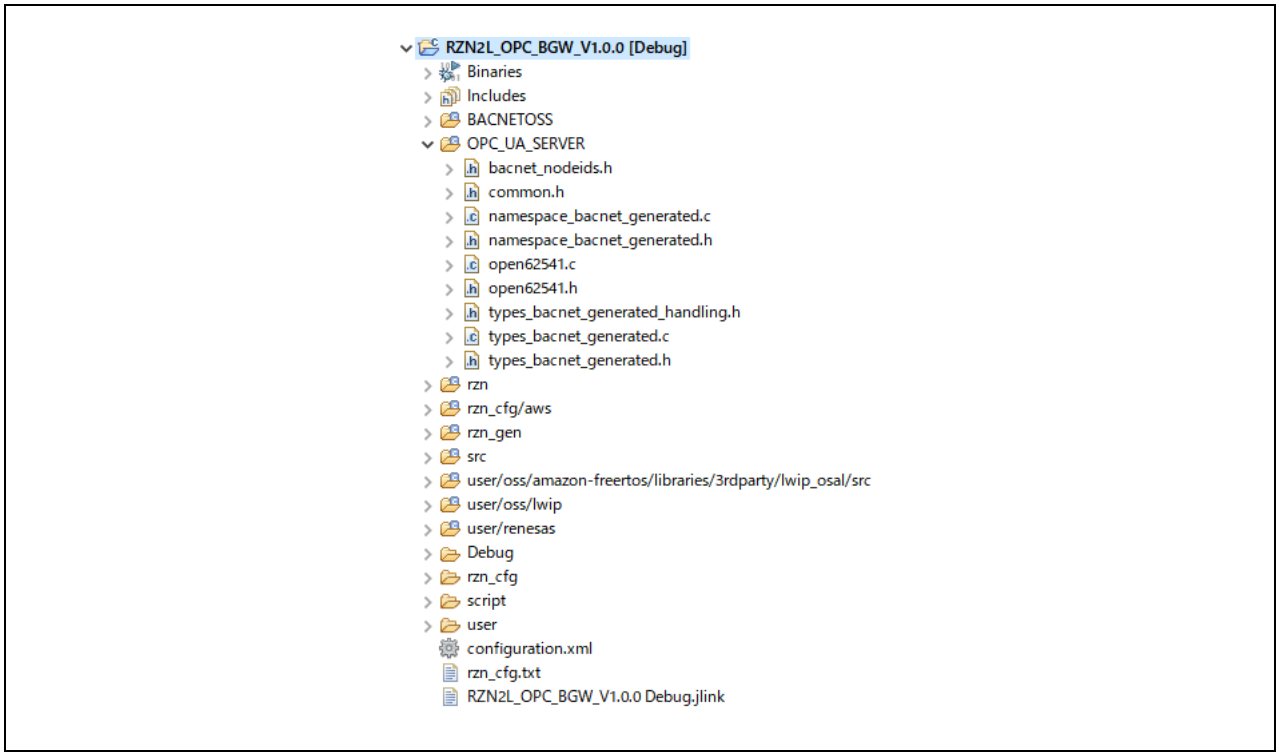


Fig.5-19 Import

5.1.4 Changes in Generated Files

Several changes have been made to the open62541 files generated by this procedure. A summary is given below.

1. Open62541.c

The open62541.c is modified during integration to *avoid namespace mismatch and merging BACnet namespace with the default application namespace* as in below Fig.5-20

include the following code snippet to merge bacnet namespace in application

```
for(size_t i = 0; i < UA_TYPES_BACNET_COUNT; ++i) {
if(UA_NodeId_equal(&UA_TYPES_BACNET[i].typeId, typeId))
        return &UA_TYPES_BACNET[i];
 }
```

Replace the following

value->value.data = booleanValue; to value->value.data = boolean;

Comment the following code snippet to avoid namespace mismatch between application namespace and bacnet namespace.

```
//if(n1->namespaceIndex != n2->namespaceIndex)
//    return (n1->namespaceIndex < n2->namespaceIndex) ? UA_ORDER_LESS : UA_ORDER_MORE;
```

The nodeset compiler uses python script the order in which the nodes are created are *not specific*.

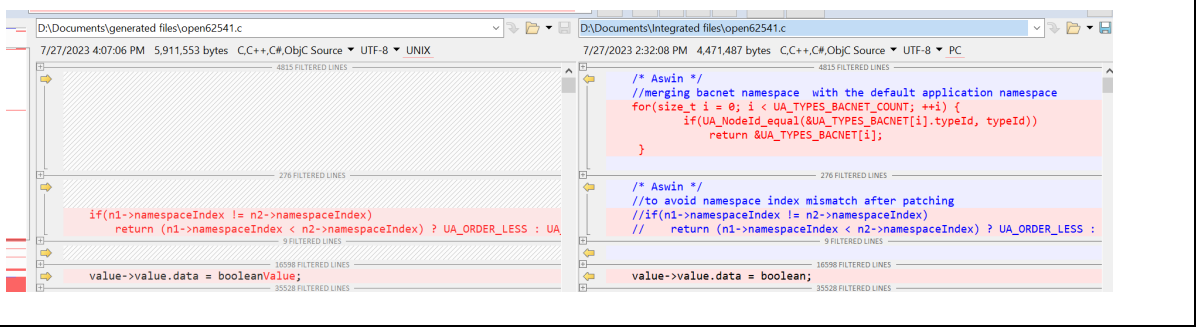


Fig.5-20 difference in open62541.c

2. Open62541.h
NO CHANGES

3. bacnet_nodeids
NO CHANGES

4. types_bacnet_generated_handling.h
NO CHANGE

5. Namespace_bacnet_generated.c
UA_TYPES->UA_TYPES_BACNET

&UA_TYPES->&UA_TYPES_BACNET

78LU->80LU

78LU is macro for optional node

80LU is macro for Mandatory node

To *enable optional node to mandatory nodes* like network scan, reinitialization ,time synchronization method and include them in Object folder In address space above change is made.

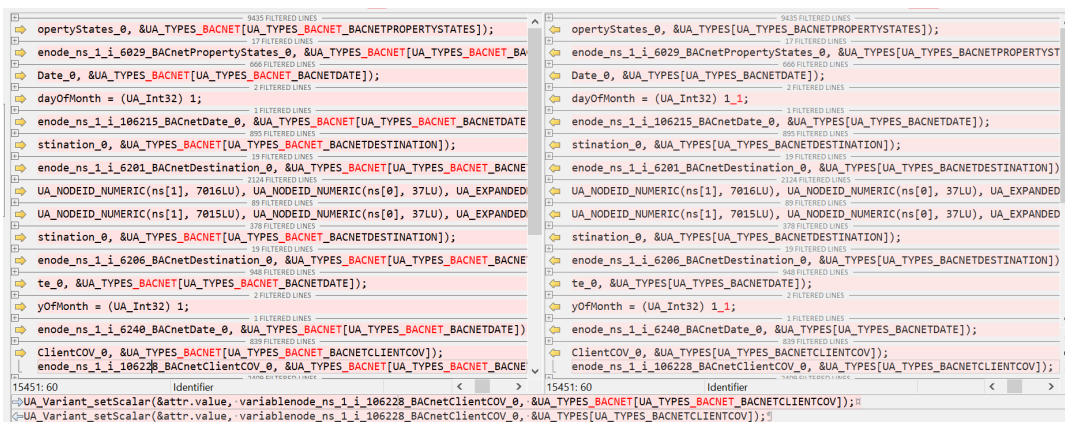


Fig.5-21 difference in namespace_bacnet_generated.c

6. Namespace_bacnet_generated.h

Remove types_bacnet_generated.h since open62541.h is included.

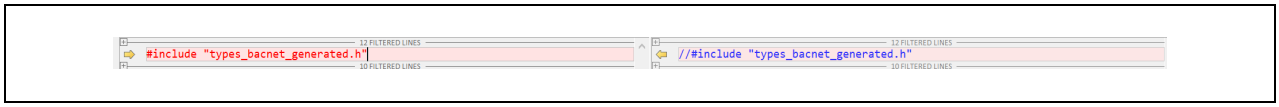


Fig.5-22 difference in namespace_bacnet_generated.h

7. **types_bacnet_generated.h**

Remove #include "types_generated.h"

Fig.5-23 difference in types_bacnet_generated.h

8. types_bacnet_generated.c

Remove the included types_generated .h and include open62541.h

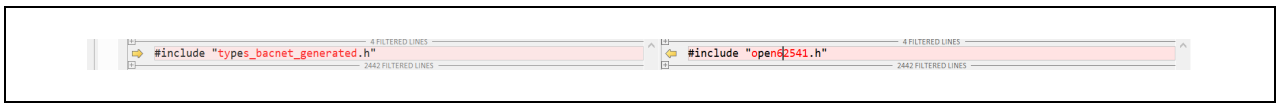


Fig.5-24 difference in types_bacnet_generated.c

5.2 B-BC Device Profile (Reference)

Indicates the support status of the B-BC device profile in this sample software version. The "Not available" in the subsequent tables does not meet the requirement in this version. They will be supported in the next version or later.

Table 5-1 BACnet Service implementation status required for B-BC

BACnet Service	Initiate[1]	Execute[2]
Who-Is	✓	✓
I-Am	✓	✓
Who-Has	✓	✓
I-Have	✓	✓
ReadProperty	✓	✓
WriteProperty	✓	✓
DeviceCommunicationControl		Not available
ReinitializeDevice		Not available
AtomicReadFile		Not available
AtomicWriteFile		Not available
TimeSynchronization		Not available
UTCTimeSynchronization		
SubscribeCOV		
ConfirmedCOVNotification		
UnconfirmedCOVNotification		
ReadPropertyMultiple	Not available	Not available
ReadPropertyConditional		
ReadRange		Not available
WritePropertyMultiple	Not available	Not available
GetAlarmSummary		
GetEventInformation		Not available
GetEnrollmentSummary		
AcknowledgeAlarm		Not available
ConfirmedEventNotification	Not available	
UnconfirmedEventNotification	Not available	
UnconfirmedTextMessage		
ConfirmedTextMessage		
AddListElement		
RemoveListElement		
CreateObject		

BACnet Service	Initiate[1]	Execute[2]
DeleteObject		
UnconfirmedPrivateTransfer		
ConfirmedPrivateTransfer		
VTOpen		
VTData		
VTClose		

✓ is applicable, blank is not applicable, and "Not available" does not meet the requirements.

1. Sends a BACnet service request or notification. However, the B-SS does not send service requests, only notifications.
2. Execute the BACnet service and send a response (if a confirmed service is requested).

Table 5-2 BACnet Object implementation status required for B-BC

BACnet Object Type	Object ID	Implementation
Accumulator		
Analog Input	Analog Input, 0	✓
	Analog Input, 1	Not available
Analog Output		
Analog Value	Analog Value, 0	Not available
	Analog Value, 1	Not available
Averaging		
Binary Input		
Binary Output	Binary Output, 0	Not available
	Binary Output, 1	Not available
Binary Value	Binary Value, 0	Not available
	Binary Value, 1	Not available
Calendar		
Command		
Device	Device, 12	Not available
Event Enrollment		
File		
Group		
Life Safety Point		
Life Safety Zone		
Loop		
Multi state Input		
Multi state Output		
Multi state Value	Multi state Value, 0	Not available
	Multi state Value, 1	Not available

BACnet Object Type	Object ID	Implementation
Notification Class		Not available
Program		
Pulse Converter		
Schedule		Not available
Trend Log		Not available
Access Door		
Event Log		
Load Control		
Structured View		
Trend Log Multiple		
Access Point		
Access Zone		
Access User		
Access Rights		
Access Credential		
Credential Data Input		
CharacterString Value		
DateTime Value		
Large Analog Value		
BitString Value		
OctetString Value		
Time Value		
Integer Value		
Positive Integer Value	Positive Integer Value, 0	Not available
	Positive Integer Value, 1	Not available
Date Value		
DateTime Pattern Value		
Time Pattern Value		
Date Pattern Value		
Network Security		
Global Group		
Notification Forwarder		
Alert Enrollment		
Channel		
Lighting Output		
Network Port		Not available

BACnet Object Type	Object ID	Implementation
Binary Lighting Output		

✓ is applicable, blank is not applicable, and "Not available" does not meet the requirements.

Table 5-3 BIBB implementation status required for B-BC

BIBB Class	BIBB	BACnet Service	Initiate[1]	Execute[2]	B-BC Standardized[3]
DataSharing	DS-RP-A,B	ReadProperty	✓	✓	✓
	DS-WP-A,B	WriteProperty	✓	✓	✓
	DS-RPM-A,B	ReadPropertyMultiple	Not available	Not available	✓
	DS-WPM-A,B	WritePropertyMultiple	Not available	Not available	✓
Alarm & Event Management	AE-N-I-B	ConfirmedEventNotification	Not available		✓
		UnconfirmedEventNotification	Not available		✓
	AE-ACK-B	AcknowledgeAlarm		Not available	✓
	AE-INFO-B	GetEventInformation		Not available	✓
Scheduling	SCHED-E-B	WriteProperty	Not available	Not available	✓
		ReadProperty		Not available	✓
Trending	T-VMT-I-B	ReadRange		Not available	✓
	T-ATR-B	ConfirmedEventNotification	Not available		✓
		UnconfirmedEventNotification	Not available		✓
		ReadRange		Not available	✓
Device & Network Management	DM-DDB-A,B	Who-Is	✓	✓	✓
		I-Am	✓	✓	✓
	DM-DOB-A,B	Who-Has	✓	✓	✓
		I-Have	✓	✓	✓
	DM-DCC-B	DeviceCommunicationControl		Not available	✓
	DM-TS-B	TimeSynchronization		Not available	✓
	DM-RD-B	ReinitializeDevice		Not available	✓
	DM-BR-B	AtomicReadFile		Not available	✓
		AtomicWriteFile		Not available	✓
		ReinitializeDevice		Not available	✓

✓ is applicable, blank is not applicable, and "Not available" does not meet the requirements.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Jul/31/2023	-	First Edition

Trademark

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. Precaution against Electrostatic Discharge (ESD)
   A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.
2. Processing at power-on
   The state of the product is undefined at the time 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 time 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 time 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 time when power is supplied until the power reaches the level at which resetting is specified.
3. Input of signal during power-off state
   Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.
4. 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 the 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.
5. Clock signals
   After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is 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. Additionally, 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.
6. Voltage application waveform at input pin
   Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).
7. Prohibition of access to reserved addresses
   Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.
8. Differences between products
   Before changing from one product to another, for example to a product with a different part number, confirm that the change will not lead to problems. The characteristics of a microprocessing unit or microcontroller unit products in the same group but having a different part number might differ in terms of 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 claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.
   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.
   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
7. No semiconductor product is absolutely secure. Notwithstanding any security measures or features that may be implemented in Renesas Electronics hardware or software products, Renesas Electronics shall have absolutely no liability arising out of any vulnerability or security breach, including but not limited to any unauthorized access to or use of a Renesas Electronics product or a system that uses a Renesas Electronics product. RENESAS ELECTRONICS DOES NOT WARRANT OR GUARANTEE THAT RENESAS ELECTRONICS PRODUCTS, OR ANY SYSTEMS CREATED USING RENESAS ELECTRONICS PRODUCTS WILL BE INVULNERABLE OR FREE FROM CORRUPTION, ATTACK, VIRUSES, INTERFERENCE, HACKING, DATA LOSS OR THEFT, OR OTHER SECURITY INTRUSION (“Vulnerability Issues”). RENESAS ELECTRONICS DISCLAIMS ANY AND ALL RESPONSIBILITY OR LIABILITY ARISING FROM OR RELATED TO ANY VULNERABILITY ISSUES. FURTHERMORE, TO THE EXTENT PERMITTED BY APPLICABLE LAW, RENESAS ELECTRONICS DISCLAIMS ANY AND ALL WARRANTIES, EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT AND ANY RELATED OR ACCOMPANYING SOFTWARE OR HARDWARE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE.
8. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
12. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
13. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
14. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.5.0-1 October 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.