R-IN32M3 Module (RY9012A0)

Firmware Update Guide

R30AN0401EJ0103
Rev.1.03
May.31.2024

Introduction

This document describes how to update the firmware for R-IN32M3 Module (RY9012A0).

Target Device

R-IN32M3 Module (RY9012A0)

Contents

List of Abbreviations and Acronyms

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

Terms	Description
API	Application Programming Interface
GOAL	Generic Open Abstraction Layer See "R-IN32M3 Module User's Manual: Software API description" (R17US0002ED****)
uGOAL	Micro Generic Open Abstraction Layer, simplified and shrunk memory usage software package of GOAL
HTTP	Hyper-Text Transfer Protocol
FoE	Ethernet over EtherCAT
SPI	Serial Peripheral Interface

Related documents

Document Type	Document Title	Document No.
Data Sheet	R-IN32M3 Module Datasheet	R19DS0109ED****
User’s Manual	R-IN32M3 Module User’s Manual: Hardware	R19UH0122ED****
User’s Manual	R-IN32M3 Module User’s Manual: Software	R17US0002ED****
Quick Start Guide	R-IN32M3 Module Application Note: Quick Start Guide	R12QS0042ED****
Application Note	R-IN32M3 Module (RY9012A0) User's Implementation Guide	R30AN0386EJ****
User’s Manual	Adaptor Board with R-IN32M3 module YCONNECT-IT-I-RJ4501	R12UZ0094EJ****
Application Note	RA6M3/RA6M4 Sample application	R30AN0398EJ****
Application Note	Management Tool Instruction Guide	R30AN0390EJ****
Application Note	Software PLC Connection Guide TwinCAT	R30AN0380EJ****

1. Overview

1.1 Abstract

This document describes how to update the firmware of R-IN32M3 module (RY9012A0) or Host MCU.

1.2 Operating Environment

This section describes the operating environment for updating the firmware of R-IN32M3 Module.

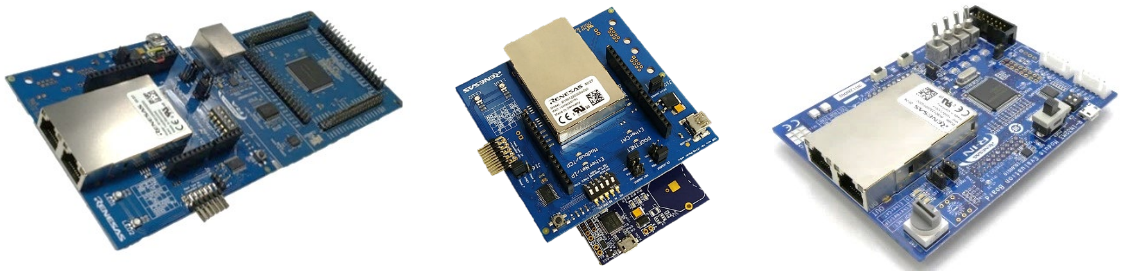
1.2.1 Hardware Environment

The firmware update function introduced in this document has been confirmed in the following hardware configurations.

(1) Combination of Adapter Board with R-IN32M3 Module and EK-RA6M3 or EK-RA6M4

(2) Combination of Adapter Board with R-IN32M3 Module and RL78/G14 (RTK5RLG140C00000BJ)

(3) RX66T CPU Card with R-IN32M3 Module



(1) R-IN32M3 Module + EK-RA6M4

(2) R-IN32M3 Module + RL78/G14 Board

(3) RX66T CPU Card with R-IN32M3 Module

Figure 1-1 R-IN32M3 Module Development Environment

Table 1-1 Hardware environments

Name	Type Name	Maker	Link	Note
Adapter Board with R-IN32M3 Module	YCONNECT-IT-I-RJ4501	Renesas Electronics Corporation	R-IN32M3-Module-Solution-Kit	
EK-RA6M3	RTK7EKA6M3S00001BU	Renesas Electronics Corporation	RA6M3 MCU Group Evaluation Board	
EK-RA6M4	RTK7EKA6M4S00001BE	Renesas Electronics Corporation	Evaluation Kit for RA6M4 MCU Group	
RL78/G14 Fast Prototyping Board	RTK5RLG140C00000BJ	Renesas Electronics Corporation	RL78/G14 Fast Prototyping Board	
RX66T CPU Card with R-IN32M3 Module	SEMB1320	SHIMAFUJI Electric Incorporated	https://www.renesas.com/SEMB1320	

In this document, it is explained by using a hardware environment in which R-IN32M3 Module Adapter board is connected to EK-RA6M4.

1.2.2 Software environment

The operating environment for updating firmware is shown inTable 1-1.

Each software is confirmed in Windows 10 (64bit) environment.

Table 1-2 Operating Environments

Category	Name	Version	Link	Remarks
R-IN32M3 module Sample package	Sample package	—	r18an0064xx01**.zip	
RA family Flexible Software Package	FSP	V5.3.0	github.com/renesas/	
Integrated development environment	e2studio	2024-04	-	Included with FSP in the installer on the above Link
GNU Arm Embedded Toolchain	GCC Toolchain	13.2.1.arm-13-7	-	Included with FSP in the installer on the above Link
Management Tool, simple software PLC	ICE	V1.5.1	-	products by port industrial automation GmbH Including with Sample package
Software PLC of EtherCAT	TwinCAT	V3.1	https://www.beckhoff.com/	products by Beckhoff Automation GmbH
RA firmware update	Python	V.3.8.3 or later	https://www.python.org/	Use Python scripts for embedding signature data
RX firmware update	Renesas Secure Flash Programmer	V2.0.2	Renesas Flash Programmer	Firmware Generation Tools Including with Sample package

1.2.3 Target Project

Sample project supporting updates.

Sample software	Update target	
	R-IN32M3 module	Host MCU
01_pnio	-	-
02_eip	-	-
03_ecat	-	-
04_pnio_largesize	-	-
05_eip_largesize	-	-
06_ecat_largesie	-	-
07_mbus_tcp_server	-	-
10_multi_protocol	-	-
11_pnio_http	-	✓
12_eip_http	-	✓
13_ecat_http	-	✓
17_fwup_bootloader	-	✓
18_fwup_spi	✓	-

Firmware program update for R-IN32M3 Module (RY9012A0)

The firmware update features by HTTP in Chapter 2.3.1 and FoE in Chapter 2.3.2 are implemented in the sample application for each hardware environment included in the sample package.

For more information about the firmware update by SPI communication, see Chapter 2.3.3.

Firmware program update for host MCU

Firmware update of the host microcontroller is executed via Ethernet (R-IN32M3 Module). To implement the firmware update function, a dedicated Bootloader must be programmed in advance.

2. R-IN32M3 Module Firmware Update

The method of updating firmware of R-IN32M3 Module is described in this chapter.

2.1 Update Methods

2.1.1 Method

There are two ways of updating the firmware for R-IN32M3 Module as below.

(1) With the Management Tool (ICE.exe)

(2) Without the Management Tool (ICE.exe)

At the way of (1), the Management tool does the whole process from unzipping an update package file, transferring, and committing it automatically. As the way of (2), three methods by HTTP, FoE and SPI are supported for transferring data at the time of firmware update. Table 2-1 shows available methods for firmware update and the default behavior for firmware update regarding commit:

Table 2-1 Firmware update method

Method	Description	Firmware Commit configuration
HTTP	Firmware upload using an HTTP-POST Request, used by the Management Tool or curl command.	by default, commit is required, thus a specific URL has to be requested after firmware update which triggers the commit
FoE	Firmware upload using an EtherCAT FoE transfer, used by the communication explorer tool for EtherCAT devices	commit is automatically done after booting of the updated firmware
SPI	Firmware upload using RPC functions by the application controller	commit is automatically done after booting of the updated firmware

2.1.2 General procedure

R-IN32M3 Module provides multiple ways to update the firmware. On the module two firmware objects are present. Those are the bootloader and the application. The bootloader is immutable by the user. It contains the basic startup code, firmware flashing routines and signature verification code for firmware update.

All shown firmware update methods have a basic procedure in common:

① The running application puts the firmware update data into the RAM of the module.

② If firmware update data was received, a reboot of the running application is performed. This step can be permitted by the application controller of host MCU.

③ The bootloader starts and detects a pending firmware update.

④ The bootloader verifies the firmware signature. If passed, the firmware is flashed - the previously running firmware is still kept.

⑤ The bootloader starts the new firmware.

⑥ If configured, commits the running firmware during bootup - else this commit process is triggered from external.

Depending on the commit execution the bootloader then starts the new firmware or the previously run firmware.

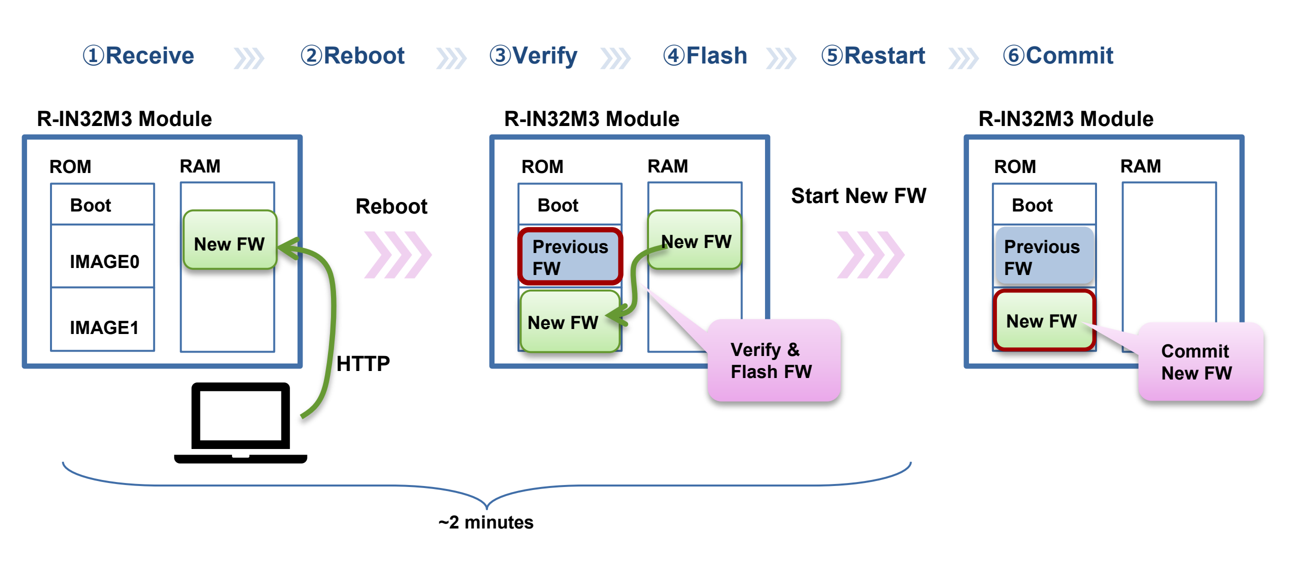


Figure 2-1 Firmware Update Procedure (HTTP)

2.1.3 Update Package

Firmware is delivered as a firmware package. This file, e.g. irj45_2.1.0.0_ci37.pfw, is basically a zip archive which contains the following files:

Unzip the firmware package, there are firmware update files in /bundle folder. (Table 2-2)

Table 2-2 Firmware update bundle content

File (/bundle)	Description
bundle.xml	bundle description
irj45_2port.dat	firmware update file for HTTP method
irj45_2port.bin	firmware update file for FoE/SPI method

(Note) Both data files are signed and must not be manipulated.

2.2 Update Procedure with Management Tool

Under control of the Management Tool, the firmware of the R-IN32M3 Module can be updated. The firmware file will be sent to R-IN32M3 Module via Ethernet connection.

For more information about the Management Tool, see the application notes (Management Tool Instruction Guide [R30AN0390EJ****]).

Without connecting to host CPU, R-IN32M3 module starts in general TCP/IP mode and the firmware can be updated.

2.2.1 Scan Device

Connect R-IN32M3 module adapter board to the EK-RA6M3 board or EK-RA6M4 board with the Arduino™ connector, and then connect PC to LAN cable and USB micro cable (Figure 2-2). By connecting USB micro B cable, the board is powered.

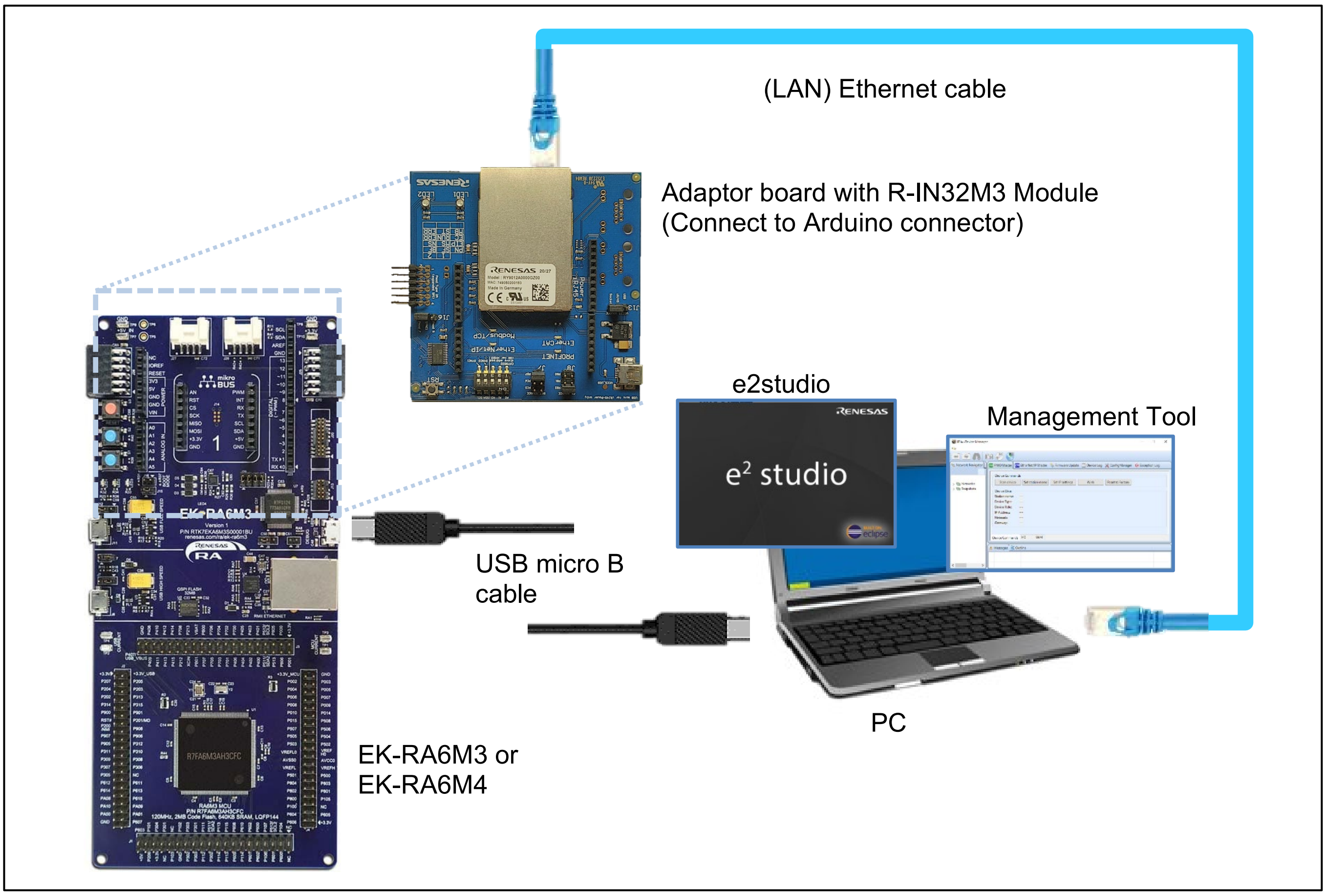
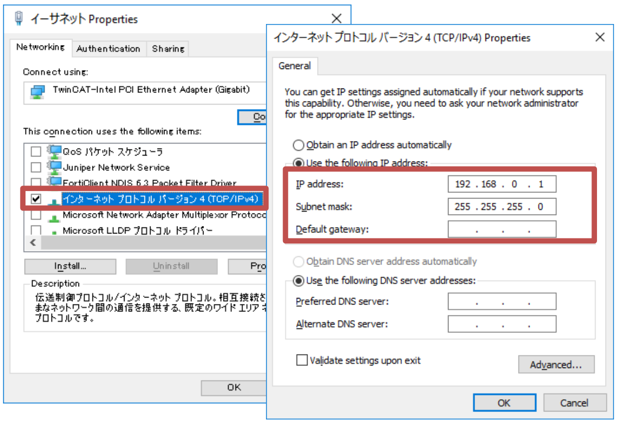


Figure 2-2 Connections

Open "Ethernet Properties" and set IP address of the PC that connects to the R-IN32M3 module.

	PC	R-IN32M3 Module (default)
IP address	192.168.0.1	192.168.0.100
Subnet mask	255.255.255.0	255.255.255.0



イーサネット Properties

Networking | Authentication | Sharing

Connect using:

TwinCAT-Intel PCI Ethernet Adapter (Gigabit)

This connection uses the following items:

- □ QoS パケット スケジューラ
- □ Juniper Network Service
- □ FortiClient NDIS 6.3 Packet Filter Driver
- ☑ インターネット プロトコル バージョン 4 (TCP/IPv4)
- □ Microsoft Network Adapter Multiplexor Protoco
- □ Microsoft LLDP プロトコル ドライバー

Description

伝送制御プロトコル/インターネット プロトコル。相互接続さ
まなネットワーク間の通信を提供する、既定のワイド エリア ネ
プロトコルです。

インターネット プロトコル バージョン 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically

● Use the following IP address:

IP address: 192 . 168 . 0 . 1

Subnet mask: 255 . 255 . 255 . 0

Default gateway: . . .

○ Obtain DNS server address automatically

● Use the following DNS server addresses:

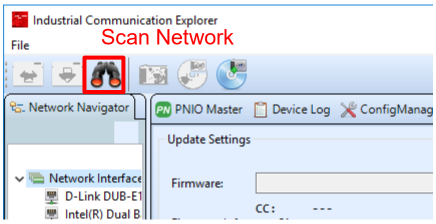
Preferred DNS server: . . .

Alternate DNS server: . . .

□ Validate settings upon exit

Figure 2-3 Setting IP address

Connect the R-IN32M3 module to the PC and press "Scan Network" to run it.



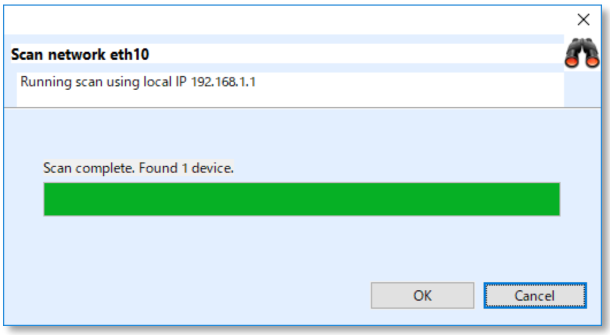
Firmware:

CC: ---

Figure 2-4 Scan Network

To communicate with R-IN32M3 module, open the "Networks" list in [Network Navigator]. Next, select the "network interfaces" that the R-IN32M3 module can connect to. Press "Scan Network" button on the toolbar.

The following dialog appears, reporting one device found:



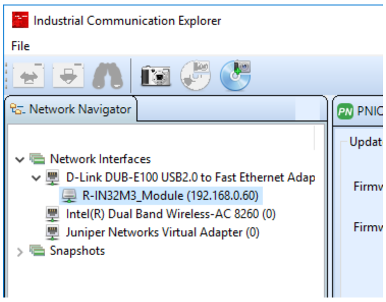
Scan network eth10

Running scan using local IP 192.168.1.1

Scan complete. Found 1 device.

Figure 2-5 Dialog of Scan Network

As a result, R-IN32M3 module appears as a new device in Network Navigator in the scanned network.



Industrial Communication Explorer

File

Network Navigator

- Network Interfaces
  - D-Link DUB-E100 USB2.0 to Fast Ethernet Adap
    - R-IN32M3_Module (192.168.0.60)
  - Intel(R) Dual Band Wireless-AC 8260 (0)
  - Juniper Networks Virtual Adapter (0)
- Snapshots

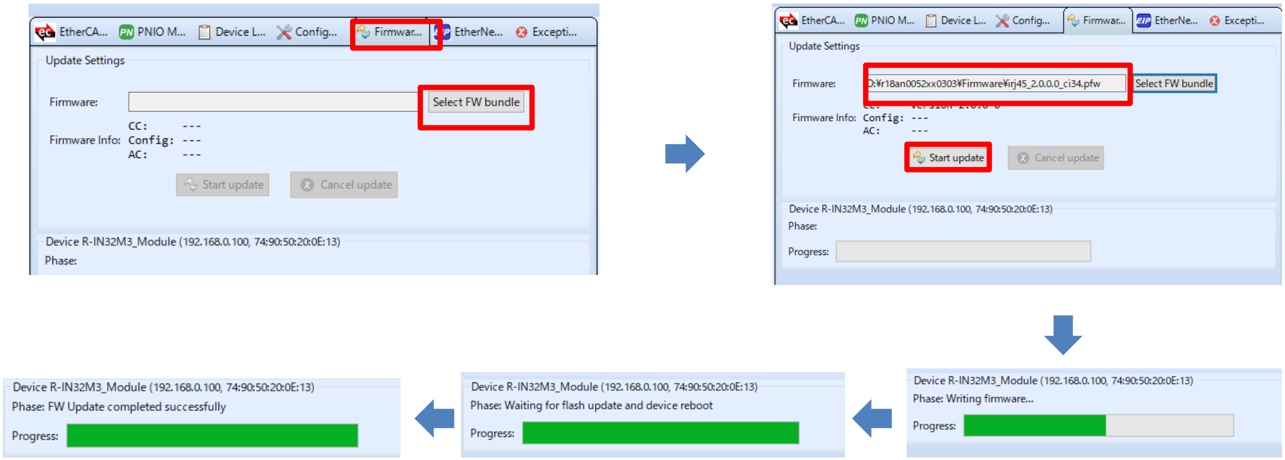
Figure 2-6 Scanned device

2.2.2 Firmware Update

Open the tab of "Firmware Update".

Specify the firmware file "Irj45_**.pfw" from "Select FW bundle".

Press "Start update", then start firmware updating. It takes about 2 minutes to complete. If Phase shows "FW Update completed successfully", it indicates that all firmware updates are complete.



Update Settings

Firmware:

Firmware Info: CC: ---
Config: ---
AC: ---

Device R-IN32M3_Module (192.168.0.100, 74:90:50:20:0E:13)
Phase:

Update Settings

Firmware: D:¥r18an0052xx0303¥Firmware¥irj45_2.0.0.0_ci34.pfw

Firmware Info: CC: Version 2.0.0 0
Config: ---
AC: ---

Device R-IN32M3_Module (192.168.0.100, 74:90:50:20:0E:13)
Phase:
Progress:

Device R-IN32M3_Module (192.168.0.100, 74:90:50:20:0E:13)
Phase: Writing firmware...
Progress:

Device R-IN32M3_Module (192.168.0.100, 74:90:50:20:0E:13)
Phase: Waiting for flash update and device reboot
Progress:

Device R-IN32M3_Module (192.168.0.100, 74:90:50:20:0E:13)
Phase: FW Update completed successfully
Progress:

Figure 2-7 Firmware Update

2.2.3 Check Firmware Version

To check the current firmware version, select [ConfigureManager] function panel and then the [Read configuration] button (Figure 2-8).

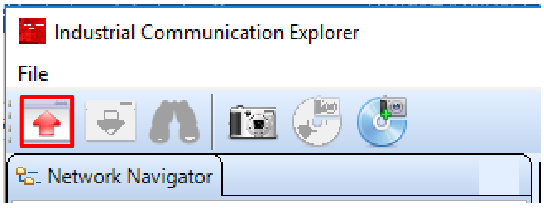
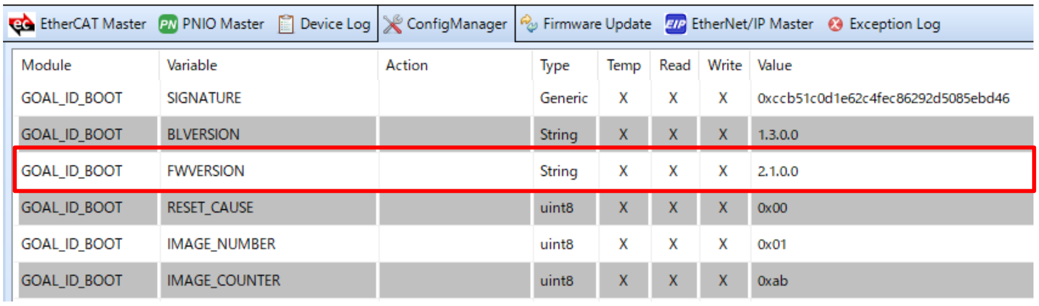


Figure 2-8 Read Configuration

The value of "FWVERSION" indicates the firmware version.



Module	Variable	Action	Type	Temp	Read	Write	Value
GOAL_ID_BOOT	SIGNATURE		Generic	X	X	X	0xccb51c0d1e62c4fec86292d5085ebd46
GOAL_ID_BOOT	BLVERSION		String	X	X	X	1.3.0.0
GOAL_ID_BOOT	FWVERSION		String	X	X	X	2.1.0.0
GOAL_ID_BOOT	RESET_CAUSE		uint8	X	X	X	0x00
GOAL_ID_BOOT	IMAGE_NUMBER		uint8	X	X	X	0x01
GOAL_ID_BOOT	IMAGE_COUNTER		uint8	X	X	X	0xab

Figure 2-9 R-IN32M3 Module Firmware version

The relevant variable parameters are shown in Table 2-3.

There are two areas in the Flash ROM to store two firmware images as shown in Figure 2-1. An image to boot-up is set by committing. The "IMAVE_NUMBER" indicates the image number to boot. It switches each time by committing during a firmware update. "IMAGE_NUMBER" represents the number of firmware image updates.

Table 2-3 Module Id = GOAL_ID_BOOT (37)

Variable Name	Variable ID	Type	Max. Size	Long description
SIGNATURE	0	GENERIC	16	Signature of booted image
BLVERSION	1	STRING	16	Bootloader Version
FWVERSION	2	STRING	16	Firmware Version
RESET_CAUSE	1000	UINT8	1	Reset cause:0, Unspecified1, Firmware Update Requested2, Watchdog3, Firmware Commit Required4, Reserved
IMAGE_NUMBER	1001	UINT8	1	Booted image number
IMAGE_COUNTER	1002	UINT8	1	Booted image age counter

2.3 Other Methods of Firmware Update

In this chapter, firmware updating methods without Management Tool are described.

2.3.1 HTTP

2.3.1.1 Outline

Provides instructions for updating firmware via HTTP protocol by use of curl command, which is standard in Windows 10.

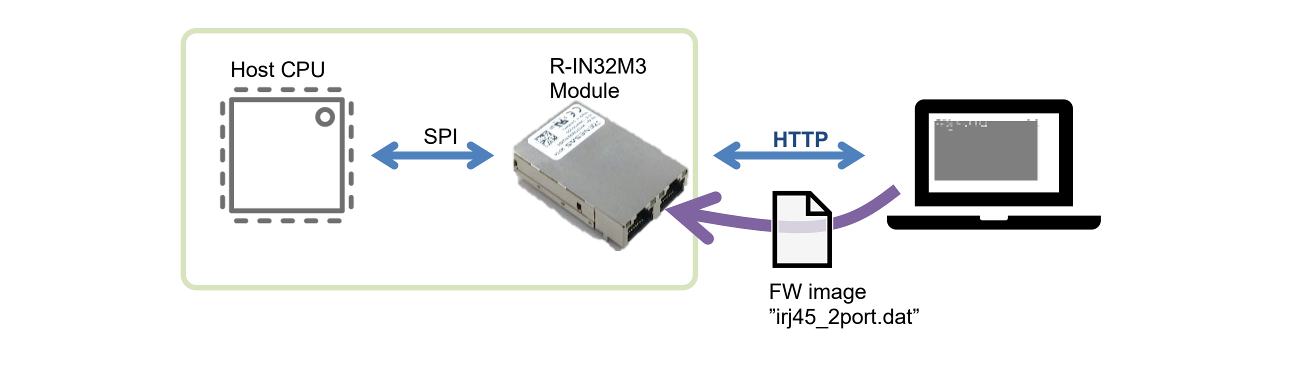


Figure 2-10 Firmware Update (HTTP)

2.3.1.2 Procedure

(1) Build the project and write it to the RA6 board.

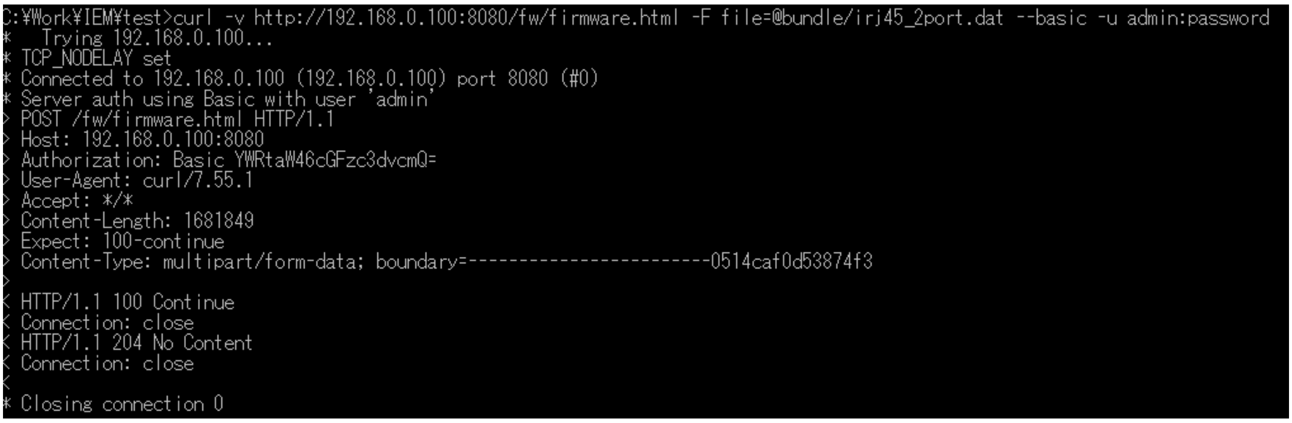
(2) Launch the command prompt and go to any folder.

In the moved folder, place the firmware update package (irj45_2.1.0.0_ci37.pfw, etc.) and zip it out.

(3) Execute the following command to write the unzipped file, " irj45_2port.dat".

(Specify the IP address as stored in the R-IN32M3 module. In the following example, 192.168.0.100. is specified. "admin" and "password" are fixed.)

curl -v http://192.168.0.100:8080/fw/firmware.html -F file=@bundle/irj45_2port.dat --basic -u admin:password



```
C:¥Work¥IEM¥test>curl -v http://192.168.0.100:8080/fw/firmware.html -F file=@bundle/irj45_2port.dat --basic -u admin:password
*   Trying 192.168.0.100...
* TCP_NODELAY set
* Connected to 192.168.0.100 (192.168.0.100) port 8080 (#0)
* Server auth using Basic with user 'admin'
> POST /fw/firmware.html HTTP/1.1
> Host: 192.168.0.100:8080
> Authorization: Basic YWRtaW46cGFzc3dvcmQ=
> User-Agent: curl/7.55.1
> Accept: */*
> Content-Length: 1681849
> Expect: 100-continue
> Content-Type: multipart/form-data; boundary=------------------------0514caf0d53874f3
>
< HTTP/1.1 100 Continue
< Connection: close
< HTTP/1.1 204 No Content
< Connection: close
<
* Closing connection 0
```

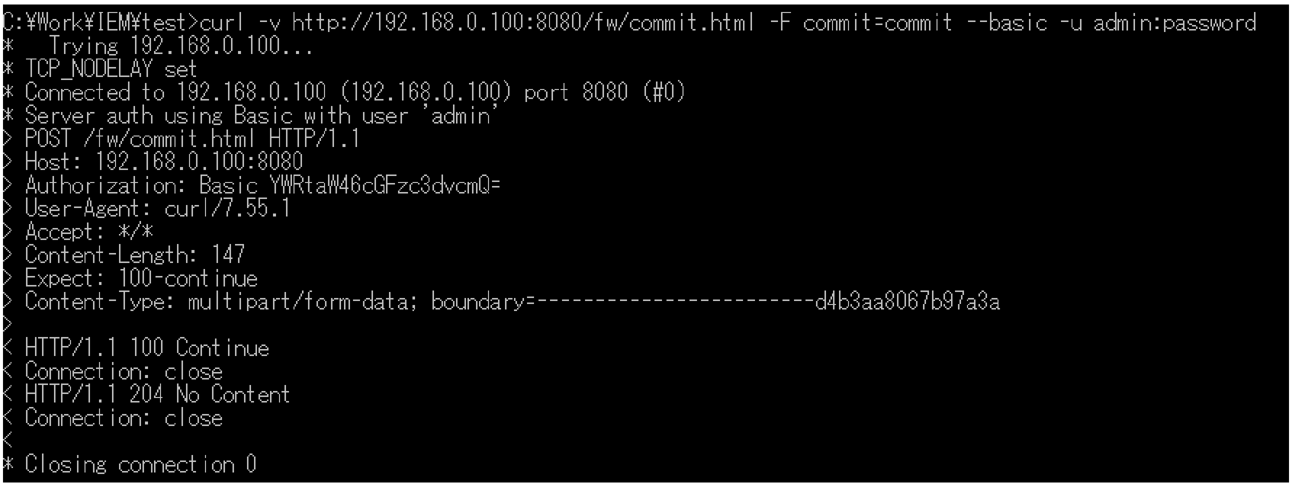
Figure 2-11 Success log

(4) Wait about 2 minutes. (until completion of ⑤ in chapter 2.1)

(Note) There are no log or indication in the command prompt during this sequence.

(5) Execute the following command to commit it（Specify the IP address as stored in the R-IN32M3 module.）

curl -v http://192.168.0.100:8080/fw/commit.html -F commit=commit --basic -u admin:password

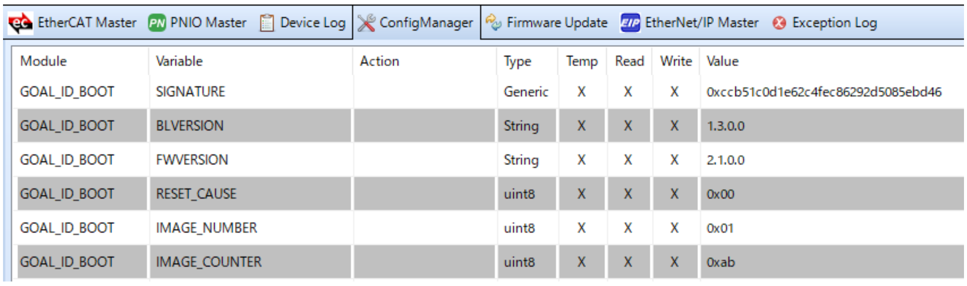


```
C:¥Work¥IEM¥test>curl -v http://192.168.0.100:8080/fw/commit.html -F commit=commit --basic -u admin:password
*   Trying 192.168.0.100...
* TCP_NODELAY set
* Connected to 192.168.0.100 (192.168.0.100) port 8080 (#0)
* Server auth using Basic with user 'admin'
> POST /fw/commit.html HTTP/1.1
> Host: 192.168.0.100:8080
> Authorization: Basic YWRtaW46cGFzc3dvcmQ=
> User-Agent: curl/7.55.1
> Accept: */*
> Content-Length: 147
> Expect: 100-continue
> Content-Type: multipart/form-data; boundary=------------------------d4b3aa8067b97a3a
>
< HTTP/1.1 100 Continue
< Connection: close
< HTTP/1.1 204 No Content
< Connection: close
<
* Closing connection 0
```

Figure 2-12 Success log

(6) Reboot the R-IN32M3 Module. It starts with the updated firmware.

The firmware version can be checked from "ConfigManager" of Management Tool (Chapter 2.2.3)



Module	Variable	Action	Type	Temp	Read	Write	Value
GOAL_ID_BOOT	SIGNATURE		Generic	X	X	X	0xccb51c0d1e62c4fec86292d5085ebd46
GOAL_ID_BOOT	BLVERSION		String	X	X	X	1.3.0.0
GOAL_ID_BOOT	FWVERSION		String	X	X	X	2.1.0.0
GOAL_ID_BOOT	RESET_CAUSE		uint8	X	X	X	0x00
GOAL_ID_BOOT	IMAGE_NUMBER		uint8	X	X	X	0x01
GOAL_ID_BOOT	IMAGE_COUNTER		uint8	X	X	X	0xab

Figure 2-13 Check Firmware Version

2.3.1.3 Multi-module update

Firmware update for multiple R-IN32M3 Modules can be performed simultaneously by connecting a switching hub between them. Figure 2-14 shows an example of network connection to update the firmware of two R-IN32M3 Modules. Send commands from PC to each IP address of R-IN32M3 Module as the procedure described in Chapter 2.3.1.2.

Simultaneous updates with daisy chain connection are not possible because there is a period of non-communication during the update procedure.

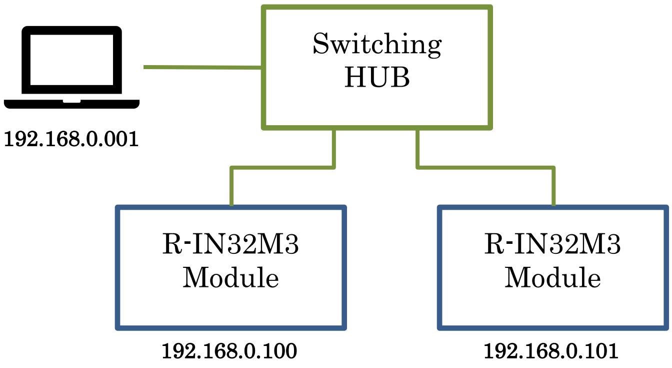


Figure 2-14 Network Connection for Multiple Module Update

2.3.1.4 Batch File (for reference))

The following is the example of batch file for updating the firmware via HTTP.

Place the .bat file in the same folder as the update file " irj45_2port.dat" and execute it with an IP address as an argument, the download starts for the target.

After sending the first command, check the ping response. When it received, send a second command to commit the updated firmware. The batch program waits ping response up to 2 minutes and exits with error if not.

<Batch example>

```
@echo off

set count=0
ping -n 1 %1 | find "ms TTL=" > NUL
if not errorlevel 1 goto upload
echo host stopped or not found
goto end

:upload

@echo on
curl -v http://%1:8080/fw/firmware.html -F file=@irj45_2port.dat --basic -u admin:password

@echo off
echo R-IN32M3_Module connectting ...
:retry
set /a count=count+1
ping -n 1 -w 1000 %1 | find "ms TTL=" > NUL
if not errorlevel 1 goto commit
if "%count%" == "120" (
  echo timeout error
  exit /b
)
goto retry

:commit
@echo on
curl -v http://%1:8080/fw/commit.html -F commit=commit --basic -u admin:password
:end
```

2.3.2 FoE

2.3.2.1 Outline

When R-IN32M3 module is running on EtherCAT, the firmware can be updated via FoE, instead of HTTP. This chapter describes the procedure for updating the firmware with EtherCAT FoE communication using TwinCAT3.

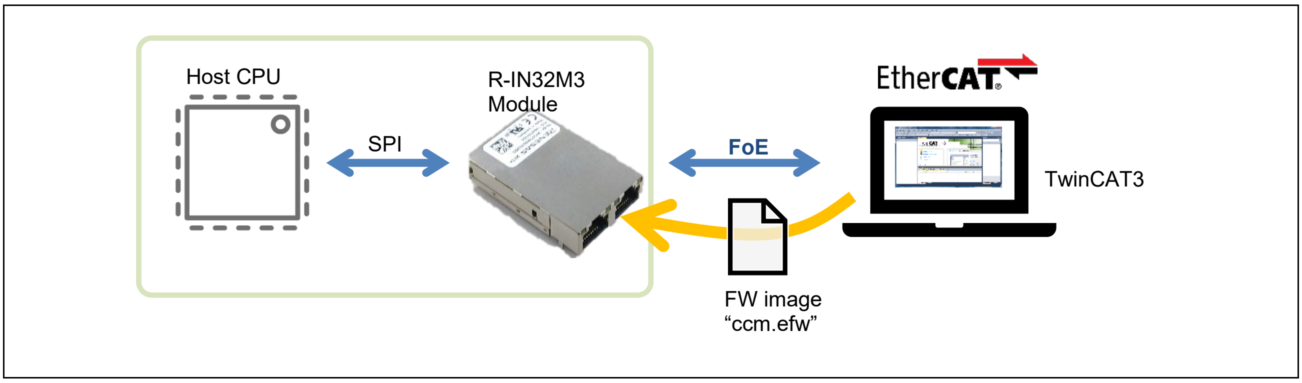


Figure 2-15 Image of Firmware Update via FoE

2.3.2.2 Preparation

(a) Check Configuration

The configuration settings of R-IN32M3 Module can be seen with Management Tool. (Chapter 2.2.3)

Table 2-4 shows the configurations associated with FoE firmware updates.

This chapter describes how to update the firmware when these settings are the default values.

Table 2-4 FoE configuration

GOAL_ID_CCM	Variable ID	Type	Default value	Description
FOE_FILENAME	9	STRING	ccm.efw	EtherCAT FoE update file name
FOE_PASSWORD	10	UINT32	0x00000000	EtherCAT FoE update password
FOE_UPDATE_REQUIRES_BOOT	11	UINT8	0x01	EtherCAT FoE update required state
FOE_FILENAME_MATCH_LEN	12	UINT8	0x00	EtherCAT FoE update file name match length

(b) Firmware update file

Unzip the update package (irj45_2.0.0.0_ci34.pfw etc.) to extract "irj45_2port.bin" for update.

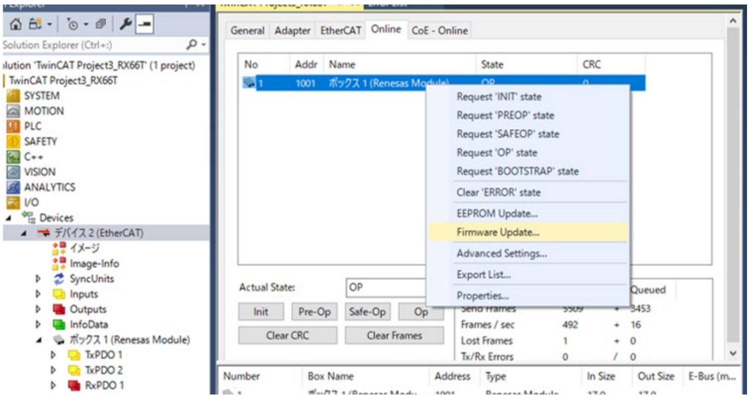
Rename "irj45_2port.bin" to the name set at "FOE_FILENAME" above. (In this description, renamed to the default value, "ccm.efw".)

2.3.2.3 Procedure

Learn how to use TwinCAT3 to update firmware with FoE.

(1) Build the project and write it to the RA6 board.

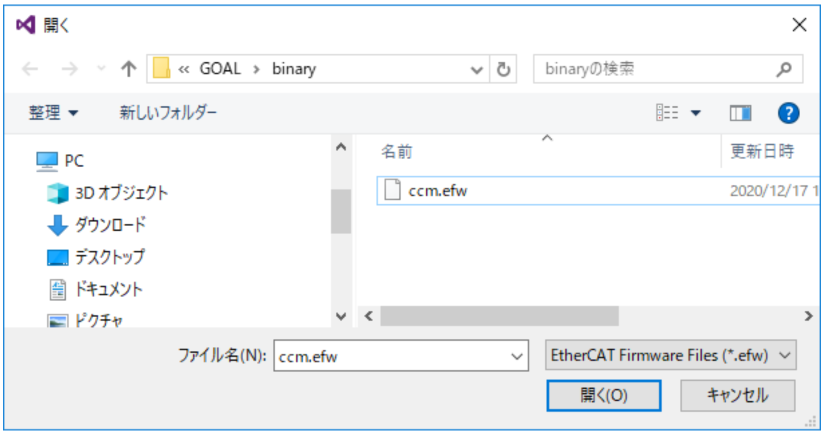
(2) Make an EtherCAT connection with TwinCAT and open online tab of [Device(EtherCAT)] on project tree. Right-click on the Renesas Module and select Firmware Update.



Actual State: OP

Figure 2-16 TwinCAT Online device nodes

(3) Press the Download button and select the firmware image file to rewrite. (In this example, "ccm.efw")



ccm.efw 2020/12/17

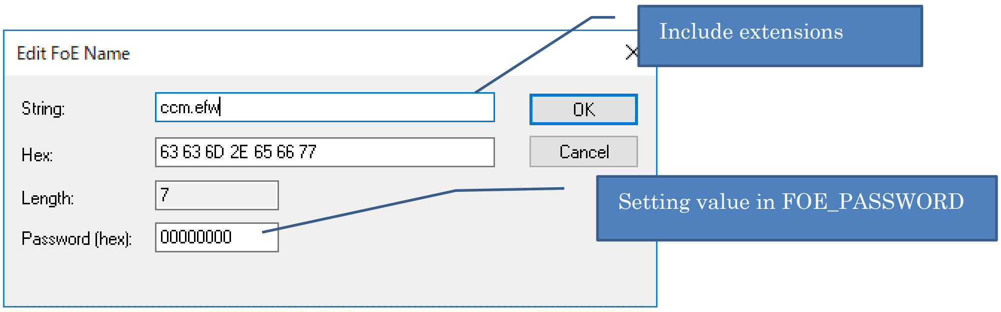
ファイル名(N): ccm.efw

EtherCAT Firmware Files (*.efw)

Figure 2-17 Select File

(4) Enter the specified parameters of FoE and press OK to start the download.

(Note) The file name displayed in "String" field does not include an extension just by selecting the file. Enter the extension manually.



String: ccm.efw

Hex: 63 63 6D 2E 65 66 77

Length: 7

Password (hex): 00000000

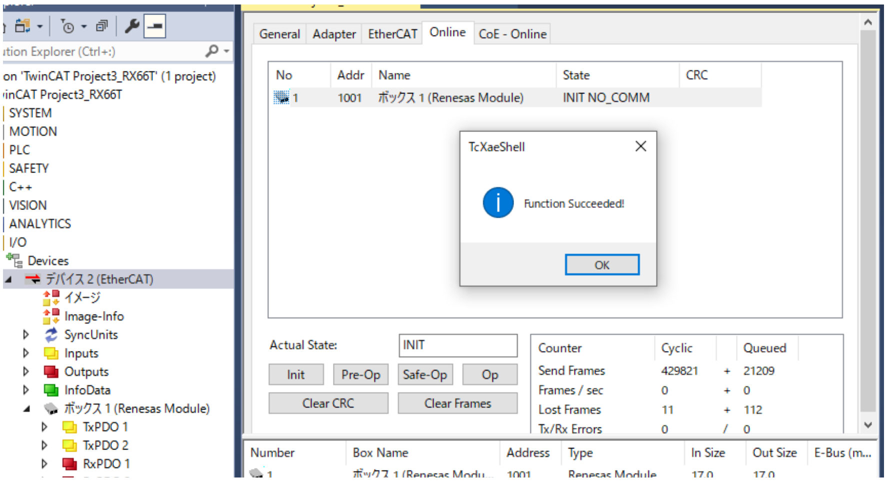
Include extensions

Setting value in FOE_PASSWORD

Figure 2-18 Edit FoE Name

(5) Waiting for update to complete

Once downloaded, status returns to INIT NO_COMM and process runs automatically.



No	Addr	Name	State	CRC
1	1001	ボックス 1 (Renesas Module)	INIT NO_COMM	

TcXaeShell

Function Succeeded!

OK

Actual State: INIT

Counter	Cyclic		Queued
Send Frames	429821	+	21209
Frames / sec	0	+	0
Lost Frames	11	+	112
Tx/Rx Errors	0	/	0

Number	Box Name	Address	Type	In Size	Out Size	E-Bus (m...
1	ボックス 1 (Renesas Modu...	1001	Renesas Module	17.0	17.0	

Figure 2-19 INIT NO_COMM

Wait about 1 minute for it to write to the Flash ROM. It then restarts and the status returns to the OP mode.

2.3.3 SPI

2.3.3.1 Overview

Describing the procedure for updating firmware of R-IN32M3 module from the host CPU via SPI interface.

This feature is implemented in the sample software for RA6 host MCU of R-IN32M3 Module. (/01_pnio, /02_eip, /03_ecat). It is enabled by setting compile option GOAL_CONFIG_FW_UPDATE_SPI=1.

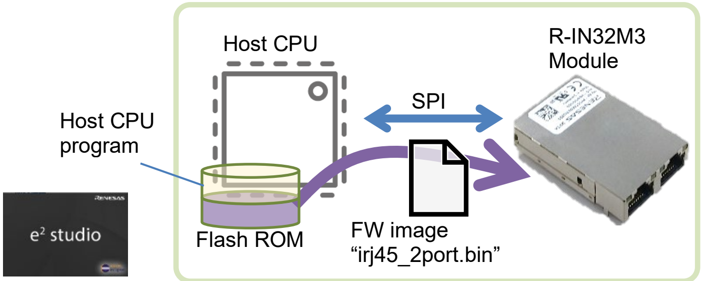


Figure 2-20 Image of Firmware Update via SPI

In this sample software, a firmware image of R-IN32M3 module is also saved in Flash ROM together with the host CPU execution program. When the host CPU is started while pressing the user switch (S1) on the EK-RA6M4 or EK-RA6M3 board, the firmware image is transferred to R-IN32M3 Module via SPI and the update process is executed.

(Note) To run this sample software, a certain ROM capacity of the host CPU is required to save the firmware image of the R-IN32M3 Module.

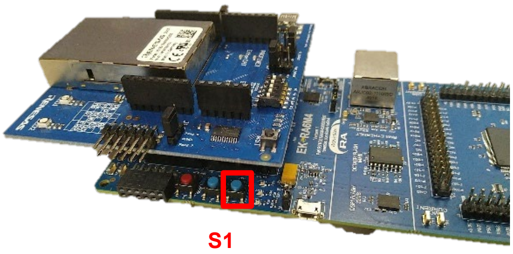


Figure 2-21 User Switch S1 on EK-RA6M4

This function is implemented in the sample software for the RA6 host MCU (/18_fwup_spi). The projects corresponding to each device are listed below.

Device	Sample software	Project Name
RA6M3	18_fwup_spi	18_fwup_spi__ra6m3_osless
RA6M4	18_fwup_spi	18_fwup_spi__ra6m4_osless

2.3.3.2 Procedure

(1) Prepare the binary file of firmware for updating.

Put " irj45_2port.bin" file, which is included in the unzipped files of firmware update package (irj45_2.1.0.0_ci37.pfw etc.), into the "binary" folder. If "binary" folder does not exist, create it under "/uGOAL" folder. (or under "/GOAL" folder with GOAL sample software.)

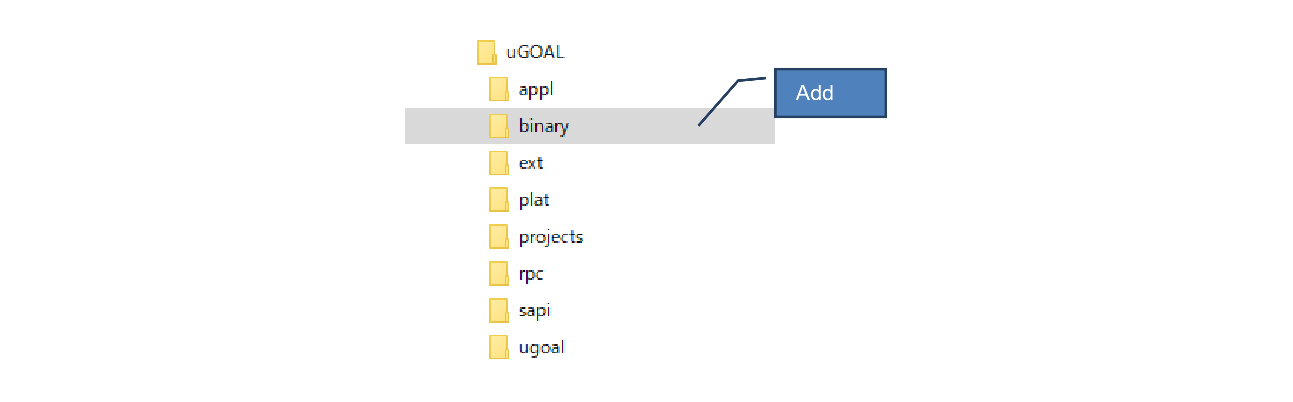
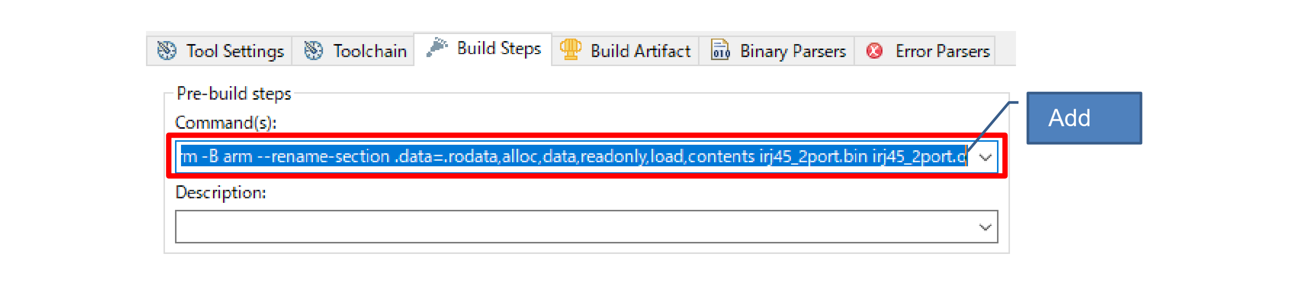


Figure 2-22 binary folder

(2) Convert the firmware binary data to an object file.

Add the following command in the Pre-build steps to convert the firmware binary data to the linkable format by objcopy. (Figure 2-23) （Already set up for /18_fwup_spi project）

cd ${GoalDirPath}/binary && arm-none-eabi-objcopy -I binary -O elf32-littlearm -B arm --rename-section .data=.rodata,alloc,data,readonly,load,contents irj45_2port.bin irj45_2port.o



Pre-build steps

Command(s):

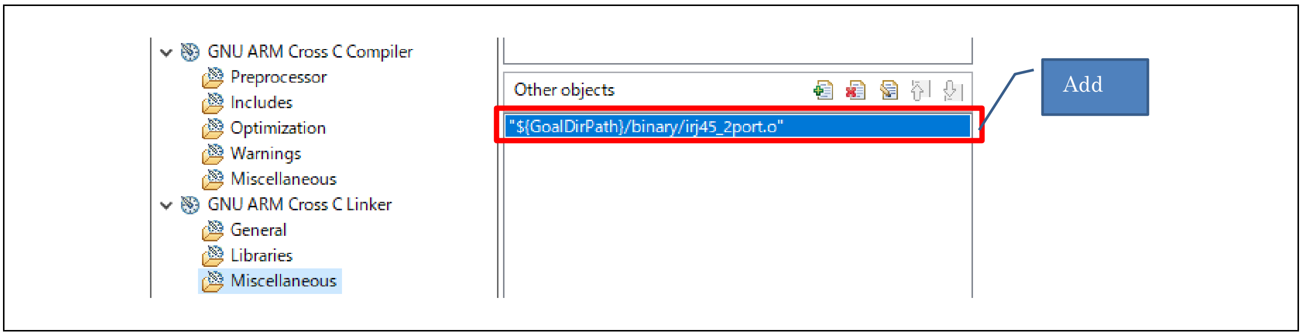
```
rm -B arm --rename-section .data=.rodata,alloc,data,readonly,load,contents irj45_2port.bin irj45_2port.o
```

Description:

Figure 2-23 Add Pre-Build steps

(3) Add the following linker setting to merge the object file created in (2). （Already set up for /18_fwup_spi project）

"${GoalDirPath}/binary/irj45_2port.o"



"${GoalDirPath}/binary/irj45_2port.o"

Figure 2-24 Add Linker setting

(4) Build the project and download it to the RA6 board.

Note that some evaluation boards take longer time.

(EK-RA6M3 takes about 10seconds. EK-RA6M4 and EK-RA4M3 board takes about 3 minutes.)

(Note) Depending on the evaluation board and program size, it might take a long time to write and verify it, then fail to start e2studio project. In that case, once close and start again to complete writing by avoiding verification time.

(5) Execute Firmware Update

While pushing on S1 switch on RA6 board, turning on the power (reset start if using a debugger) will start transferring and writing the firmware image via SPI

When data transfer starts, LED1 and LED2 flash alternately, and it stops when data transfer ends. At the same time, the operating protocol LED (PROFINET, etc.) turns off. (It takes about 3 minutes.))

When SPI data transfer is complete, R-IN32M3 module firmware is updated. the module cannot respond for about 1 minute during updating. When firmware update is complete, the host MCU is automatically restarted. Then, the update is completed.

If the debugger is running, return to top of the program. Press continue button.

2.3.3.3 Batch File (for reference)

As explained in Chapter 2.3.3.2, e2studio may fail if the program takes too long to download. This chapter shows an example batch program which directly executes J-Link and update firmware of host MCU (RA6M3/RA6M4 without e2studio for reference.

Condition

- For RA6M3 host MCU sample software (uGOAL).

- /01_pnio project

- Installed J-Link V6.3 or later in C:/Program Files (x86)/SEGGER/JLink)

How to use

(1) Build the project to download in advance.

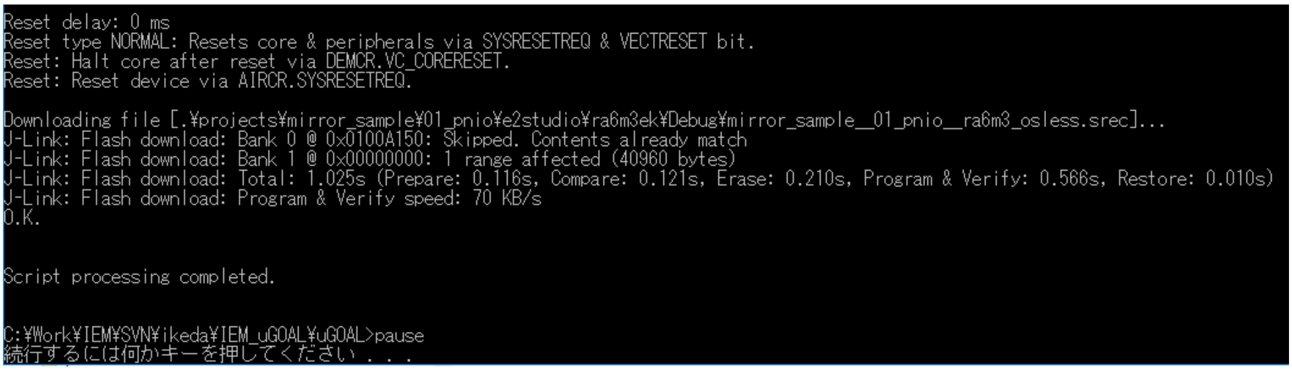
(2) Place the batch file directly under /GOAL folder.

In the example, 01_pnio project for RA6M3 is downloaded. If other projects, modify the sample batch program according to the path or device instruction of J-Link.

(3) Double-click the batch file to execute.

(4) If it succeeds, the message in Figure 2-25 is displayed.

(5) Turn the power back on, then completed.



```
Reset delay: 0 ms
Reset type NORMAL: Resets core & peripherals via SYSRESETREQ & VECTRESET bit.
Reset: Halt core after reset via DEMCR.VC_CORERESET.
Reset: Reset device via AIRCR.SYSRESETREQ.

Downloading file [.¥projects¥mirror_sample¥01_pnio¥e2studio¥ra6m3ek¥Debug¥mirror_sample__01_pnio__ra6m3_osless.srec]...
J-Link: Flash download: Bank 0 @ 0x0100A150: Skipped. Contents already match
J-Link: Flash download: Bank 1 @ 0x00000000: 1 range affected (40960 bytes)
J-Link: Flash download: Total: 1.025s (Prepare: 0.116s, Compare: 0.121s, Erase: 0.210s, Program & Verify: 0.566s, Restore: 0.010s)
J-Link: Flash download: Program & Verify speed: 70 KB/s
O.K.


Script processing completed.


C:¥Work¥IEM¥SVN¥ikeda¥IEM_uGOAL¥uGOAL>pause
続行するには何かキーを押してください . . .
```

Figure 2-25 Log of Batch file execution

<Batch example>

```
if exist .\ProgramLoader.Cmd (
    del ProgramLoader.Cmd
)
echo h>> ProgramLoader.Cmd
echo r>> ProgramLoader.Cmd
echo
loadfile .\projects\mirror_sample\01_pnio\e2studio\ra6m3ek\Debug\mirror_sample__01_pnio
__ra6m3_osless.srec,0x00000000>> ProgramLoader.Cmd
echo q>> ProgramLoader.Cmd
"C:\Program Files (x86)\SEGGER\JLink\JLink.exe" -speed 4000 -if SWD -device
R7FA6M3AH -autoconnect 1 -CommanderScript ProgramLoader.Cmd
pause
```

3. Host MCU Firmware Update

This section describes the firmware update of the host microcontroller. The firmware update of the host microcontroller is executed via Ethernet (R-IN32M3 Module).

The update method varies depending on the platform.

(Note) The sample software for RL does not support firmware updates for the host microcontroller.

3.1 RA microcontroller

This section describes how to update the firmware of the RA microcontroller.
RA Sample Application Note [R30AN0398EJ****]) and set up the development environment.

(Note) Updating the firmware of RA microcontrollers assumes that a dedicated program is written in the Bootloader area. Since the Bootloader cannot be rewritten via Ethernet, the Initial Firm [Boot Program + User Program] must be written in the Flash Writer at the time of product production.

Bootloader sample project: 17_fwup_bootloader

3.1.1 Tool

A Python script is used to embed the signature data. Please follow the steps below to build a Python environment and embed the signature data.

-1. Install Python Ver.3.8.3 or later.

-2. Run Smart configurator of Bootloader project on e2studio

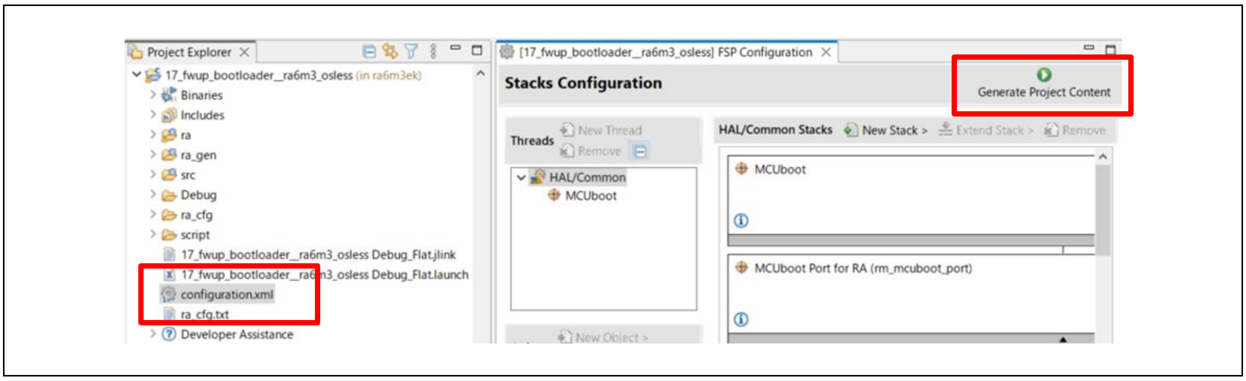


Figure 3-1 Code generation for Bootloader

-3. Select MCUboot and run [Command Prompt]

projects\17_fwup_bootloader\e2studio\<DeviceName>\ra\mcu-tools\

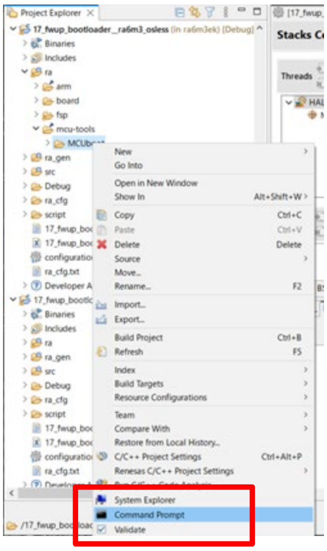


Figure 3-2 Command Prompt

-4. Update pip

Enter the following command to update pip.

```
python -m pip install --upgrade pip
```

-5. Install MCUboot dependencies

install all MCUboot dependencies by entering the following command line in the command window.

```
pip3 install --user -r scripts/requirements.txt
```

If you have more than one Python version installed, make sure the Python version is version 3.8.3 or later.

3.1.2 Preparation for firmware update

3.1.2.1 Bootloader program

-1. Build Bootloader project

The projects corresponding to each device are listed below.

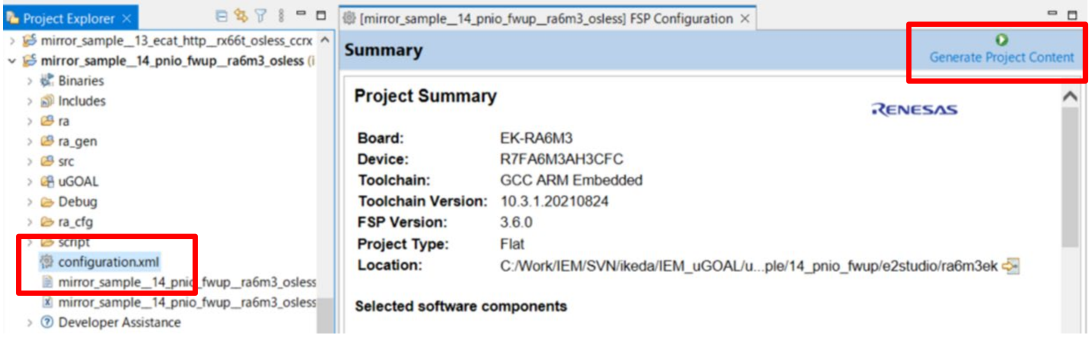
Device	Sample software	Project Name
RA6M3	17_fwup_bootloader	17_fwup_bootloader__ra6m3_osless
RA6M4	17_fwup_bootloader	17_fwup_bootloader__ra6m4_osless

-2. Build User project
Since the Bootloader does not have a write function, the main program must be programed by a flash loader or other means in the initial state (when the main program has not been programed).

The projects corresponding to each device/protocol are as follow.

Device	Sample software	Project Name
RA6M3	11_pnio_http	11_pnio_http__ra6m3_osless
	12_eip_http	12_eip_http__ra6m3_osless
	13_ecat_http	13_ecat_http__ra6m3_osless
RA6M4	11_pnio_http	11_pnio_http__ra6m4_osless
	12_eip_http	12_eip_http__ra6m4_osless
	13_ecat_http	13_ecat_http__ra6m4_osless

1) FSP code generate before Build


Project Explorer
mirror_sample__13_ecat_http__rx66t_osless_ccrx
mirror_sample__14_pnio_fwup__ra6m3_osless
Binaries
Includes
ra
ra_gen
src
uGOAL
Debug
ra_cfg
script
configuration.xml
mirror_sample__14_pnio_fwup__ra6m3_osless
mirror_sample__14_pnio_fwup__ra6m3_osless
Developer Assistance
[mirror_sample__14_pnio_fwup__ra6m3_osless] FSP Configuration
Summary
Generate Project Content
RENESAS

Project Summary

Board: EK-RA6M3
Device: R7FA6M3AH3CFC
Toolchain: GCC ARM Embedded
Toolchain Version: 10.3.1.20210824
FSP Version: 3.6.0
Project Type: Flat
Location: C:/Work/IEM/SVN/ikeda/IEM_uGOAL/u...ple/14_pnio_fwup/e2studio/ra6m3ek

Selected software components

Figure 3-3 Code Generate

.

2) Select “DebugFwup”

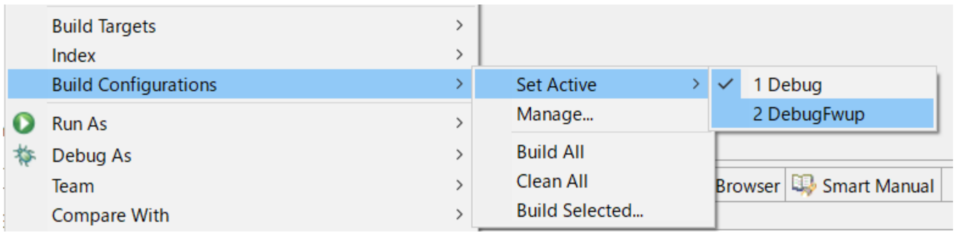
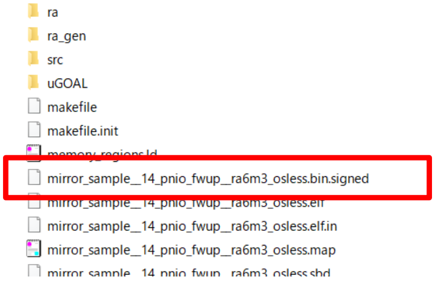


Figure 3-4 Change Build configuration

3) If the build is successful, a user program download file <project name>.bin.signed will be generated under the project folder.
ex: projects\11_pnio_http\e2studio\ra6m3ek\DebugFwup



```
ra
ra_gen
src
uGOAL
makefile
makefile.init
memory_regions.ld
mirror_sample__14_pnio_fwup__ra6m3_osless.bin.signed
mirror_sample__14_pnio_fwup__ra6m3_osless.elf
mirror_sample__14_pnio_fwup__ra6m3_osless.elf.in
mirror_sample__14_pnio_fwup__ra6m3_osless.map
mirror_sample__14_pnio_fwup__ra6m3_osless.sbd
```

Figure 3-5 bin.signed file

-3. Debug

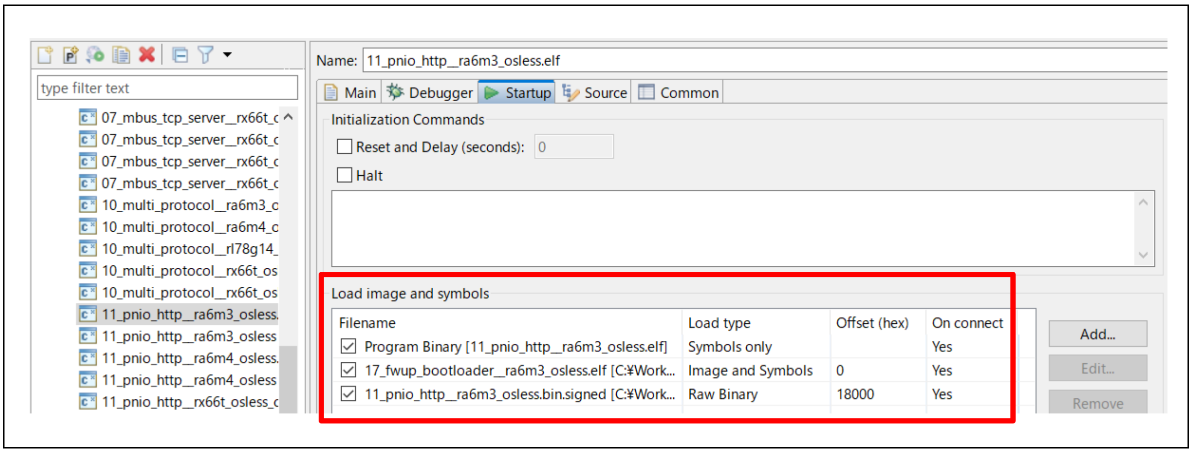
When debugging a project, the Bootloader and user program must be downloaded at the same time, set before debugging.

1) Added setting to load Bootloader in debugger settings *

* This setting has already been set in the debugger configuration file of the sample project

Select [Run] > [Debug Configurations …] from menu, and show [Debug Configuration].

- Change “Load type” to “Symbols only”
- Add Bootloader for same device type
- Set User program area “H’ 18000”


type filter text
07_mbus_tcp_server__rx66t_c
07_mbus_tcp_server__rx66t_c
07_mbus_tcp_server__rx66t_c
07_mbus_tcp_server__rx66t_c
10_multi_protocol__ra6m3_c
10_multi_protocol__ra6m4_c
10_multi_protocol__rl78g14_
10_multi_protocol__rx66t_os
10_multi_protocol__rx66t_os
11_pnio_http__ra6m3_osless.
11_pnio_http__ra6m3_osless
11_pnio_http__ra6m4_osless.
11_pnio_http__ra6m4_osless
11_pnio_http__rx66t_osless_c
Main
Debugger
Startup
Source
Common
Add...
Edit...
Remove

Name: 11_pnio_http__ra6m3_osless.elf

Initialization Commands

☐ Reset and Delay (seconds): 0

☐ Halt

Load image and symbols

Filename	Load type	Offset (hex)	On connect
☑ Program Binary [11_pnio_http__ra6m3_osless.elf]	Symbols only		Yes
☑ 17_fwup_bootloader__ra6m3_osless.elf [C:¥Work...	Image and Symbols	0	Yes
☑ 11_pnio_http__ra6m3_osless.bin.signed [C:¥Work...	Raw Binary	18000	Yes

Figure 3-6 debug configuration

2) Run Debug

Bootloader and user programs are programmed.

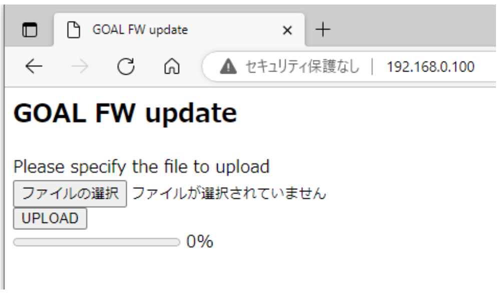
3.1.2.2 Update User program

Update the user programs over the network.

To update the user program, the Bootloader and user program described in 3.1.2.1 Bootloader program must be written beforehand.

-1. Connect with evaluation board via web browser
The IP address is set to "192.168.0.100" in sample software, connect at http://192.168.0.100.

For EtherCAT, the EtherCAT master must also be configured, see the chapter Web Server Function in the RX Sample Application Note [R30AN0399EJ****]).



GOAL FW update

セキュリティ保護なし | 192.168.0.100

GOAL FW update

Please specify the file to upload

ファイルの選択 ファイルが選択されていません

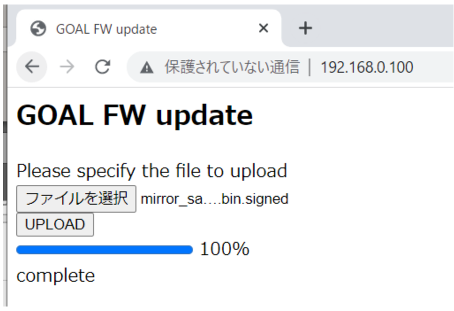
UPLOAD

0%

Figure 3-7 Web browser access

-2. Select load module
Set load module "*.bin.signed" file from [File Select].

-3. Upload
Push [UPLOAD] to start the update. The upload will take 1 minutes.



GOAL FW update

保護されていない通信 | 192.168.0.100

GOAL FW update

Please specify the file to upload

ファイルを選択 mirror_sa....bin.signed

UPLOAD

100%

complete

Figure 3-8 Update

-4. Reboot
When the upload is complete, the system is operating with the old firmware. After rebooting, the host MCU will operate with the new firmware that has been programed.

3.2 RX microcontroller

This section describes how to update the firmware of the RX microcontroller.
RX Sample Application Note [R30AN0399EJ****]) and set up the development environment.

(Note) Updating the firmware of RX microcontrollers assumes that a dedicated program is written in the Bootloader area. Since the Bootloader cannot be rewritten via Ethernet, the Initial Firm [Boot Program + User Program] must be written in the Flash Writer at the time of product production.

Bootloader sample project: 17_fwup_bootloader

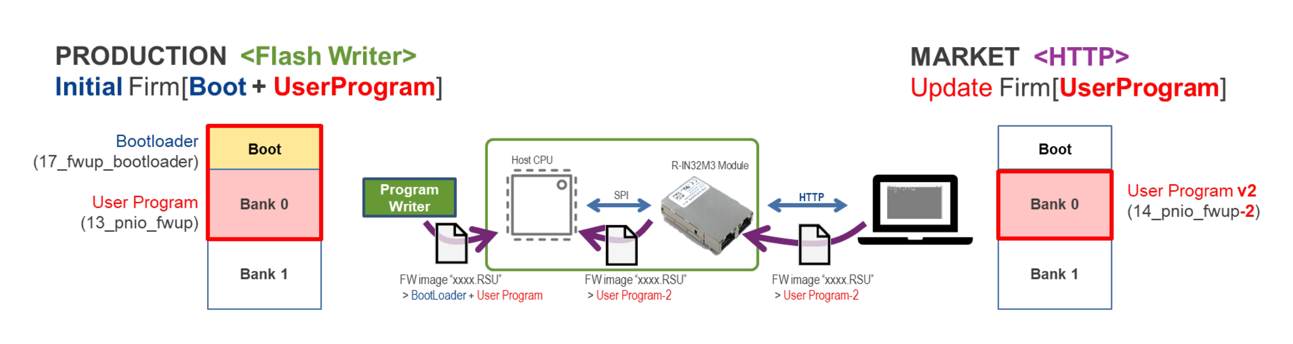


Figure 3-9 Web browser access

3.2.1 Tool

A Python script is used to embed the signature data. Please follow the steps below to build a Python environment and embed the signature data.

When updating via Ethernet, the mot file generated by building the application must be converted to a downloadable RSU file format. The Renesas Image Generator included in the sample package [r18an0064xx01**] is used as a tool to convert the file format.

(Note) The Renesas Image Generator included in the sample package is the host microcontroller (R5F566TKADFP) used in the CPU card with SEMB1320 R-IN32M3 module modified to original Renesas Image Generator.
Original Renesas Image Generator is released: Convert MOT file to RSU | (github.com/renesas)

3.2.2 Preparation for firmware update

To update the firmware of an RX microcontroller, a dedicated program must be programed in the Bootloader area. This section describes the method of generating update firmware and the update procedure.

Sample projects of update firmware are provided for each protocol. Also, GNU-C and CC-RX versions are available depending on the build environment.

Table 3-1 Sample projects for bootloaders and updating firmware

Compiler	Protocol/Function	Sample software	Project Name
GNU-C	PROFINET	11_pnio_http	11_pnio_http__rx66t_osless
	EtherNet/IP	12_eip_http	12_eip_http__rx66t_osless
	EtherCAT	13_ecat_http	13_ecat_http__rx66t_osless
	Bootloader	17_fwup_bootloader	17_fwup_bootloader__ra6m4_osless
CC-RX	PROFINET	11_pnio_http	11_pnio_http__rx66t_osless_ccrx
	EtherNet/IP	12_eip_http	12_eip_http__rx66t_osless_ccrx
	EtherCAT	13_ecat_http	13_ecat_http__rx66t_osless_ccrx
	Bootloader	17_fwup_bootloader	17_fwup_bootloader__ra6m4_osless_ccrx

3.2.2.1 Bootloader program

Bootloader and user program programming procedures.

-1. Add Tinycrypt library

To ensure the integrity of the updated firmware, a digital signature (ECDSA+SHA256) is used for verification. The following code must be added to the sample project to perform the verification.

Get it from Github | (github.com/renesas/) and add it to the lib folder under the bootloader project.

lib folder: 17_fwup_bootloader\tinycrypt\lib\include\tinycrypt

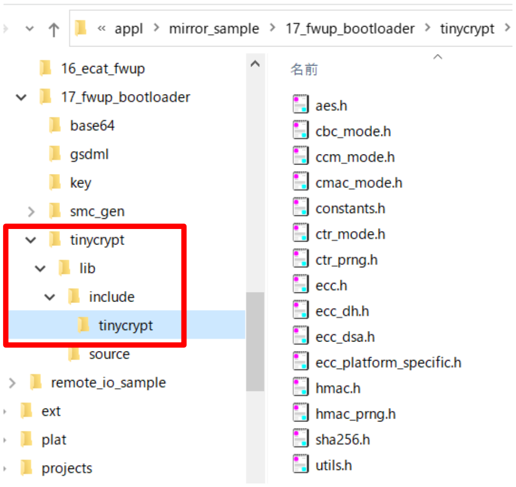


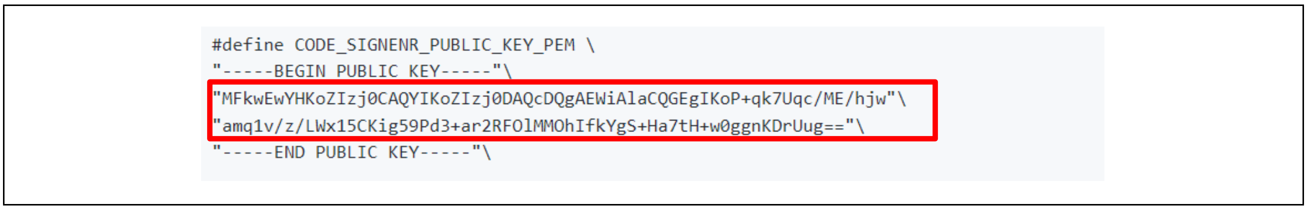
Figure 3-10 Library

-2. Public key information of key file

Set the public key information for signature verification in the sample application header file [code_signer_public_key.h].

Key creation with OpenSSL
Refer to the explanation below on how to create the file.
Key creation with OpenSSL | (github.com/renesas/)

After creation, public keys and private keys are generated in the files secp256r1.publickey and secp256r1.privatekey, respectively.



```
#define CODE_SIGNENR_PUBLIC_KEY_PEM \
"-----BEGIN PUBLIC KEY-----"\
"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEWiAlaCQGEgIKoP+qk7Uqc/ME/hjw"\
"amq1v/z/LWx15CKig59Pd3+ar2RFOlMMOhIfkYgS+Ha7tH+w0ggnKDrUug=="\
"-----END PUBLIC KEY-----"\
```

Figure 3-11 Key pair

-3. Run Build

Build the Bootloader project.

-4. Programing Bootloader

Start the debugger. To continue writing the user program, stop debugging with the stop button, etc., and proceed to the next step.

-5. Run User program

When the program download is complete, select the [Resume] button to run the program.

3.2.2.2 Generate user program for updating

This section describes the generation of the update firmware RSU file for the user program.

-1. Generate mod file

To generate updated firmware, generate a file in Motorola format (*.mot) when building the project.

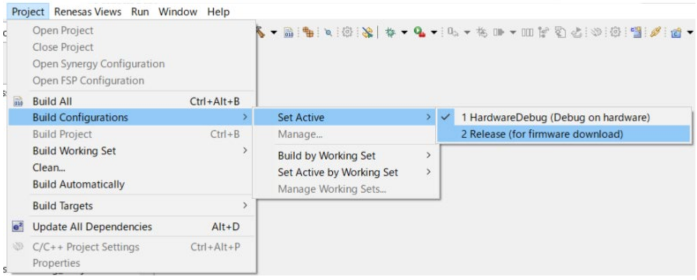
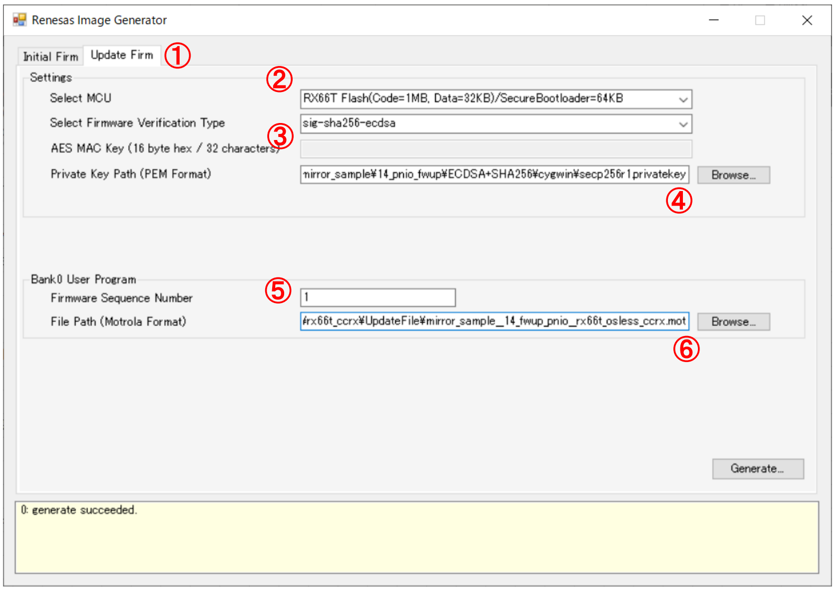


Figure 3-12 Build mod file

-2. RSU file Conversion

Convert the generated mot file to RSU file using Renesas Image Generator.



Settings

Select MCU: RX66T Flash(Code=1MB, Data=32KB)/SecureBootloader=64KB

Select Firmware Verification Type: sig-sha256-ecdsa

AES MAC Key (16 byte hex / 32 characters):

Private Key Path (PEM Format): mirror_sample¥14_pnio_fwup¥ECDSA+SHA256¥cygwin¥secp256r1.privatekey

Bank0 User Program

Firmware Sequence Number: 1

File Path (Motrola Format): ¥rx66t_ccrx¥UpdateFile¥mirror_sample__14_fwup_pnio__rx66t_osless_ccrx.mot

```
0: generate succeeded.
```

Figure 3-13 RSU conversion [Renesas Image Generator]

①	Select [Update Firm]
②	Select [RX66T Flash(Code=1MB, Data=32KB)/SecureBootloader=64KB]
③	Select [sig-sha256-ecdsa]
④	Configuration of secp256r1.privatekey create in 3.2.2.1
⑤	Set any value
⑥	Set mod file generated in -1.

After pressing the Generate button, the message "generate succeeded" will be displayed if the RSU file is successfully generated.

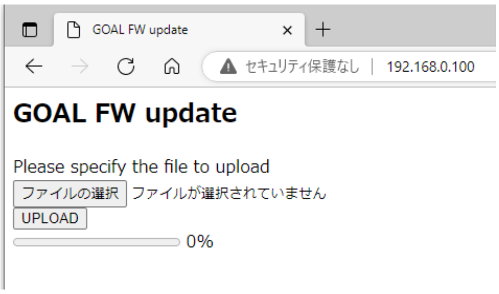
3.2.2.3 Update User program

Update the user program via the network.

To update the user program, the Bootloader and user program described in 3.2.2.1 Bootloader program must have been written beforehand.

-1. Connect with evaluation board via web browser
The IP address is set to "192.168.0.100" in sample software, connect at http://192.168.0.100.

For EtherCAT, the EtherCAT master must also be configured, see the chapter Web Server Function in the RX Sample Application Note [R30AN0399EJ****]).



GOAL FW update

Please specify the file to upload

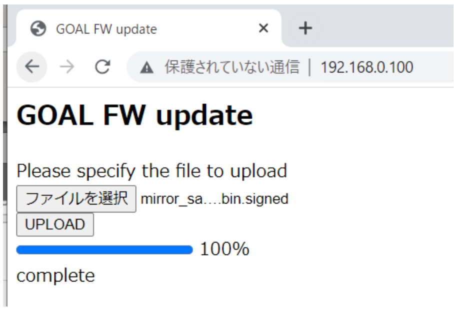
ファイルが選択されていません

0%

Figure 3-14 Web browser access

-2. Select RSU file
Set the RSU file generated by Renesas Image Generator from the [Select File].

-3. Upload
Push [UPLOAD] to start the update. The upload will take 1 minutes.



GOAL FW update

Please specify the file to upload

mirror_sa....bin.signed

100%

complete

Figure 3-15 Update

-4. Reboot
When the upload is complete, the system is operating with the old firmware. After rebooting, the host MCU will operate with the new firmware that has been programed.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Oct/15/2021	-	First Edition
1.01	May/31/2023	25	Added Host MCU firmware update
1.02	Dec/15/2023	18	Changed description of FW update procedure with EtherCAT
1.03	May/31/2024	6	Updated Table 1-2
		7, 21, 22	Added firmware update project via SPI
		27	Added project list
		32	Added project list

Trademark

- ARM and Cortex are registered trademarks of ARM Limited (or its subsidiaries) in the EU and/or elsewhere. All rights reserved.
- Ethernet is a registered trademark of Fuji Xerox Co., Ltd.
- EtherCAT® and TwinCAT® are registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- Additionally, all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners.

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

(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.4.0-1 November 2017)

Corporate Headquarters

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

Trademarks

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

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/.