RYZ014A and RX MCU

Firmware Upgrade from Host MCU for RX

要旨

本書では、RYZ014A のファームウェアをホスト MCU からアップグレードするサンプルアプリケーションについて説明します。

本サンプルアプリケーションは、ホスト MCU に RX65N を搭載した CK-RX65N ボードを使用し、Pmod コネクタに Pmod™ Expansion Board for RYZ014A を接続した構成で動作します。RYZ014A ファームウェアファイルは USB フラッシュドライブに格納し、CK-RX65N ボードの USB Full Speed コネクタに接続します。USB フラッシュドライブから RYZ014A へファームウェアを転送してアップグレードを行います。

本サンプルアプリケーションはベアメタル上で動作します。

動作確認デバイス

RYZ014A

CK-RX65N

関連ドキュメント

- Renesas LTE Cat-M1 Cellular IoT Module RYZ014A Pmod™ Expansion Board (R21QS0004)
- RYZ014 Module System Integration Guide (R19AN0074)
- RX65N グループ、RX651 グループ ユーザーズマニュアル ハードウェア編 (R01UH0590)
- Cloud Kit for RX65N Microcontroller Group CK-RX65N v1 User's Manual (R20UT5100)
- Firmware Integration Technology ユーザーズマニュアル (R01AN1833)

目次

1. 概要

製品寿命の間に少なくとも 1 回はファームウェアのアップグレードが必要になる可能性があります。RYZ014A のファームウェアをアップグレードする方法は以下の 3 種類あり、本アプリケーションノートではこの中の制御マイコン(ホスト MCU)からアップグレードする方法について説明します。

- LTE ネットワークに接続して無線でアップグレード(Firmware Upgrade Over-the-Air (FOTA))
- PC からアップグレード
- 制御マイコン(ホスト MCU)からアップグレード

ホスト MCU に RX65N を搭載した CK-RX65N ボードを使用し、Pmod コネクタに Pmod Expansion Board for RYZ014A を接続した構成で動作します。RYZ014A ファームウェアファイルは USB フラッシュドライブに格納し、CK-RX65N ボードの USB Full Speed コネクタに接続します。USB フラッシュドライブから RYZ014A へファームウェアを転送してアップグレードを行います。

ファームウェアアップグレード・サンプルアプリケーションの構成図を図 1-1 に示します。

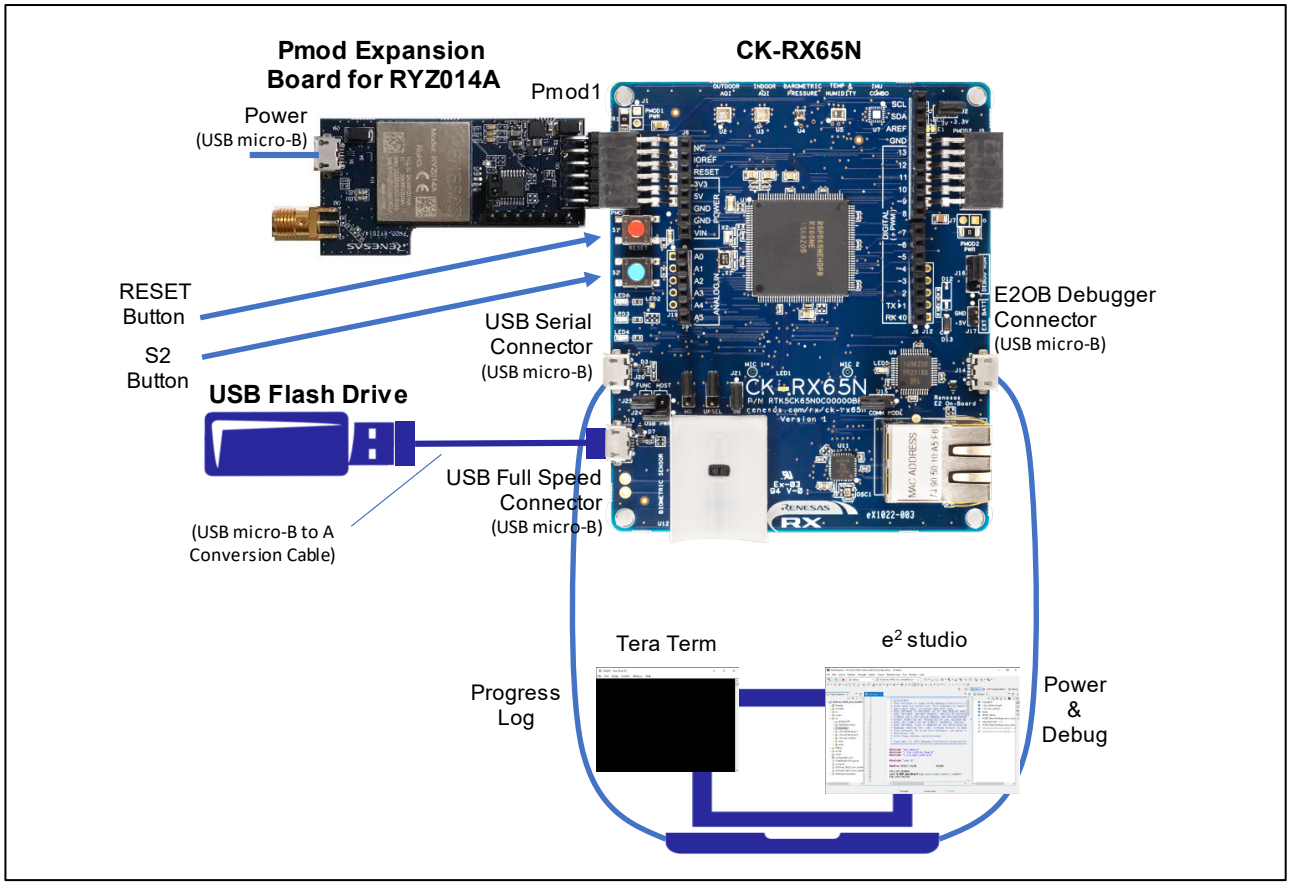

図 1-1 ファームウェアアップグレード・サンプルアプリケーション構成図

2. 動作環境

2.1 ハードウェア

サンプルアプリケーションで使用するハードウェア要件を「表 2-1 ハードウェア要件」に示します。

表 2-1 ハードウェア要件

周辺機能	型名等
Pmod Expansion Board for RYZ014A	RTKYZ014A0B00000BE
CK-RX65N	RTK5CK65N0S04000BE
USB フラッシュドライブ	10MB 以上
Windows 10 PC	---
USB ケーブル	• CK-RX65N E2OB Debugger(Power & Debug)用 (micro-B) • CK-RX65N USB Serial Connecter(Progress Log)用 (micro-B) • Pmod Expansion Board for RYZ014A(Power)用 (micro-B) • USB フラッシュドライブ用 (USB micro-B to A 変換)

2.2 ソフトウェア

サンプルアプリケーションで使用するソフトウェア要件を「表 2-2 ソフトウェア要件」に示します。

表 2-2 ソフトウェア要件

ソフトウェア	バージョン等
e^2 studio	2022-10
CC-RX コンパイラ	3.04
Renesas Flash Programmer	V3.11
Tera Term	V4.106 ※動作確認用として実行ログの表示で使用します。

2.3 マイコン周辺機能

サンプルアプリケーションで使用するマイコン周辺機能を「表 2-3 マイコン周辺機能」に示します。

表 2-3 マイコン周辺機能

周辺機能名		用途
シリアルコミュニケーションインタフェース	SCI6(UART6)	Pmod1: RYZ014A との UART 通信 ボーレート:921600 bps データ長:8 bit パリティ:none ストップビット:1 bit フロー制御: ハードウェア CTS/ソフトウェア RTS RTS 信号:P02 CTS 信号:PJ3
	SCI5(UART5)	USB Serial Connector: Tera Term で進行ログの表示 ボーレート:115200 bps データ長:8 bit パリティ:none ストップビット:1 bit フロー制御: なし
コンペアマッチタイマ W	CMTW0	RYZ014A との UART 通信タイムアウト
	CMTW1	LED 点滅周期
USB2.0 FS ホスト	USB2.0 FS Host	USB Full Speed Connector: ファームウェアファイルが格納されている USB フラッシュドライブとの通信
I/O ポート	P55	RYZ014A のリセット端子制御
	P02	RYZ014A との UART 通信で使用する RTS 信号
	PJ3	RYZ014A との UART 通信で使用する CTS 信号
	PA3	ユーザーLED 青
	P25	ユーザーLED 赤
外部端子割り込み	IRQ1	ユーザスイッチ S2 の外部端子割り込み

2.4 FIT Modules

サンプルアプリケーションで使用する FIT モジュールを「表 2-4 使用する FIT モジュール」に示します。スマート・コンフィグレータにてコード生成を行うことにより FIT モジュールを自動で入手することができます。

表 2-4 使用する FIT モジュール

機能	コンポーネント名		バージョン
BSP	Board Support Package	r_bsp	7.20
ポート	Config_PORT	Config_PORT	2.4.0
SCI	SCI Driver	r_sci_rx	4.40
	Byte-based circular buffer library	r_byteq	2.00
	GPIO Driver	r_gpio_rx	4.50
IRQ	IRQ Driver	r_irq_rx	4.10
USB	USB Basic	r_usb_basic	1.40
	USB Host Mass Storage Class	r_usb_hmsc	1.31
File System	Open Source Fat File System	r_tfat_rx	4.02
	Memory Driver Interface for Open Source FAT File System	r_tfat_driver_rx	2.20
	CMT driver	r_cmt_rx	5.20
	Generic system timer for RX MCUs using CMT module	r_sys_timer_rx	1.01
Timer	CMTW Driver	r_cmtw_rx	2.60

2.5 ディレクトリ・ファイル構成

サンプルアプリケーションのディレクトリ・ファイル構成を「表 2-5 ディレクトリ・ファイル構成」に示します。

表 2-5 ディレクトリ・ファイル構成

ディレクトリ・ファイル構成				説明
RYZ014A_FWUP_from_HostMCU¥		.cproject .project RYZ014A_FWUP_from_HostMCU HardwareDebug.launch RYZ014A_FWUP_from_HostMCU.rcpc RYZ014A_FWUP_from_HostMCU.scfg		プロジェクトファイル
	.settings¥	com.renesas.cdt.managedbuild.renesas.ccrx.prefs com.renesas.hardwaredebug.rx.e2.PerfAnalysisSettings.xml com.renesas.smc.e2studio.qe.xml com.renesas.smc.generationsetting.properties CoverageSetting.xml DebugVirtualConsoleSetting.xml Dependency_Scan_Preferences.prefs e2studio_project.prefs IORegisterSetting.xml org.eclipse.cdt.core.prefs org.eclipse.core.resources.prefs renesasPGModel.xml		e^2 studio 設定ファイル
	src¥	main.c r_lte_ryz014a_fwup.c r_lte_ryz014a_fwup.h r_lte_ryz014a_fwup_console.c r_lte_ryz014a_fwup_console.h r_lte_ryz014a_fwup_hal_data.c r_lte_ryz014a_fwup_hal_data.h r_lte_user_config.h	ファームウェアアップグレード・サンプルアプリケーション・ソースコード	
		smc_gen¥	FIT モジュールソースコード	

3. 動作確認

サンプルアプリケーションの使用方法について示します。実行環境については「図 1-1 ファームウェアアップグレード・サンプルアプリケーション構成図」を参照してください。

サンプルアプリケーションは LTE ネットワークに接続していない状態で実行してください。

3.1 RYZ014A ファームウェアの準備

RYZ014A のファームウェアファイル(注)を以下の名前に変更し USB フラッシュドライブのルートディレクトリにコピーします。

ryz014a_firmware.dup

注: ファームウェアファイルは必要に応じてルネサスからお客様に直接提供します。動作中のファームウェアのバージョンによってファイルが異なりますので、ルネサスへお問い合わせの際には動作中のファームウェアバーションをお知らせください。

3.2 CK-RX65N ジャンパ設定

CK-RX65N の USB Full Speed インタフェースは、USB フラッシュドライブを接続するため USB Host として使用します。「RX65N Group Cloud Kit for RX65N Microcontroller Group CK-RX65N v1 User's Manual」(R20UT5100)に従い、次のジャンパ設定を行ってください。

J25: ピン 1-2 ショート (USB Host)

J24: ショート (Bus-powered operation)

その他のジャンパは、「RX65N Group Cloud Kit for RX65N Microcontroller Group CK-RX65N v1 User's Manual」(R20UT5100)の「Table 2. Default Jumper Settings」に記載されている設定にしてください。

3.3 プロジェクトのインポート

e² studio にサンプルアプリケーションのプロジェクトをインポートする手順を示します。

1. e² studio を起動してワークスペースディレクトリを指定し「Launch」ボタンをクリックします。

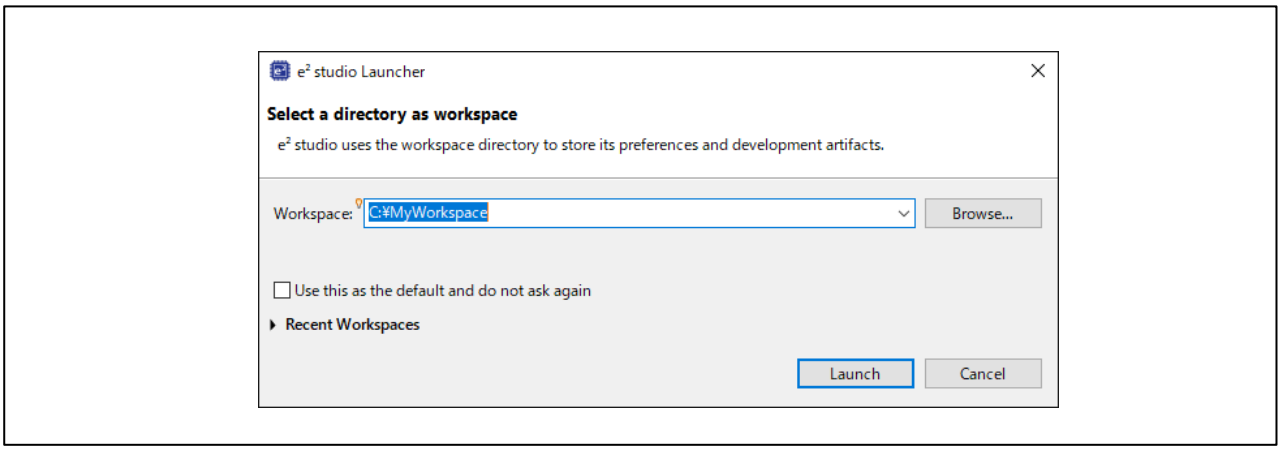

図 3-1 Workspace の選択

2. メニューバーから「File」→「Import...」を選択します。

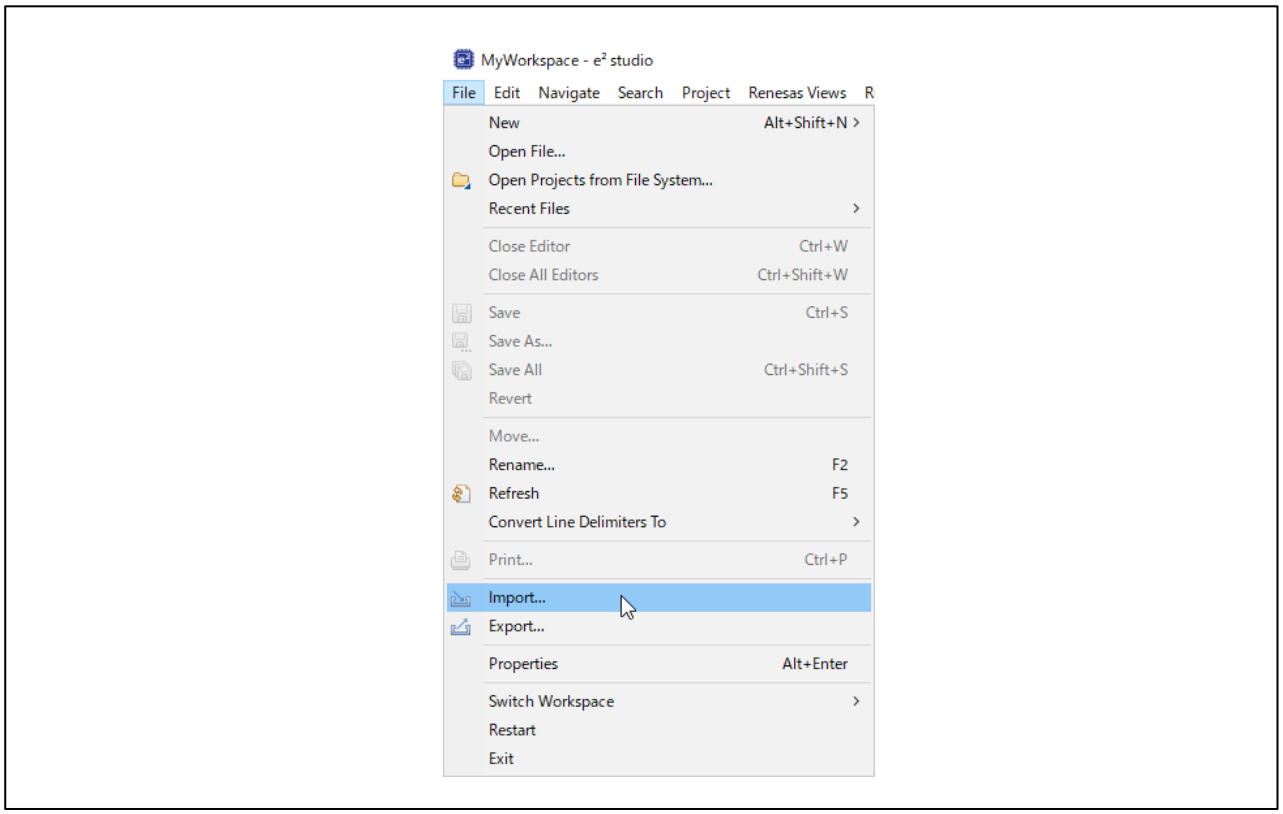

図 3-2 File メニュー

3. 「Existing Projects into Workspace」を選択し「Next」をクリックします。

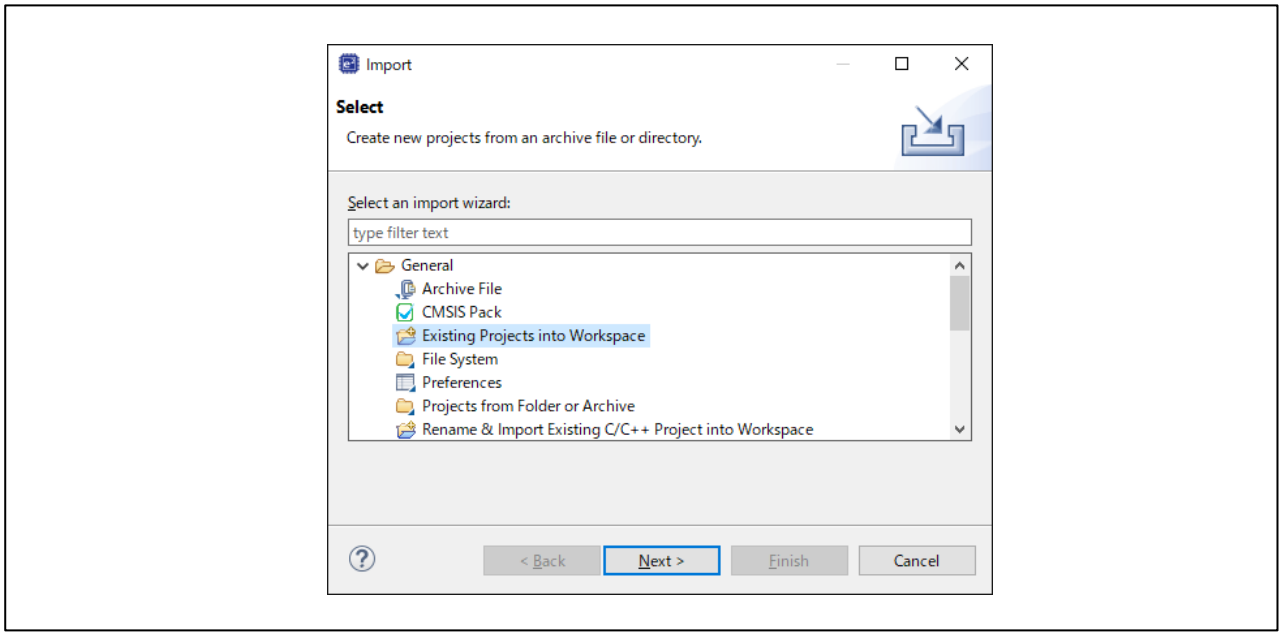

図 3-3 Import wizard の選択

4. 「Select root directory」を選択して「Browse...」をクリックし、サンプルプロジェクトのディレクトリを選択します。「Finish」ボタンを押すとプロジェクトがインポートされます。

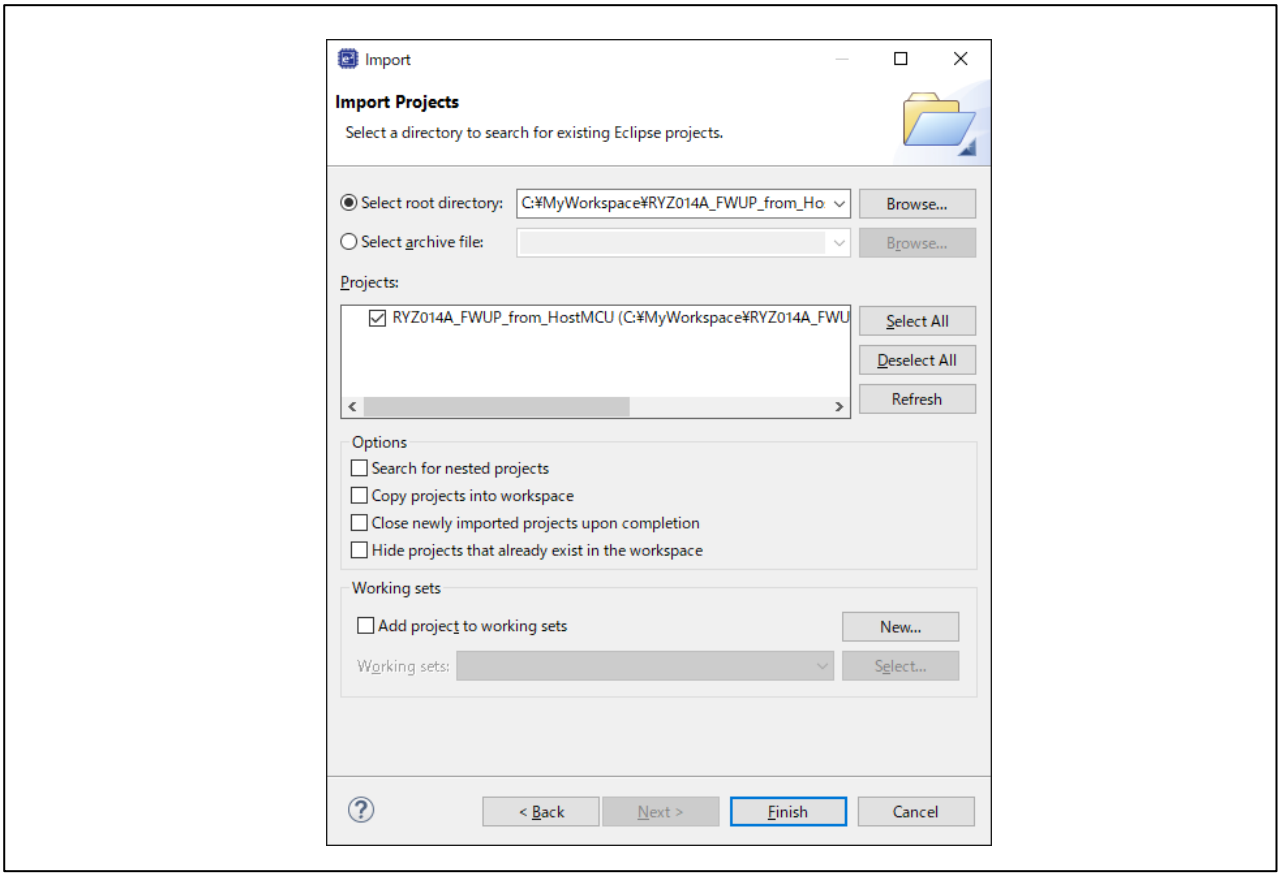

図 3-4 Project の import

3.4 ビルドとダウンロード

1. RYZ014A_FWUP_from_HostMCU.scfg をダブルクリックしスマート・コンフィグレータウインドウを開きます。

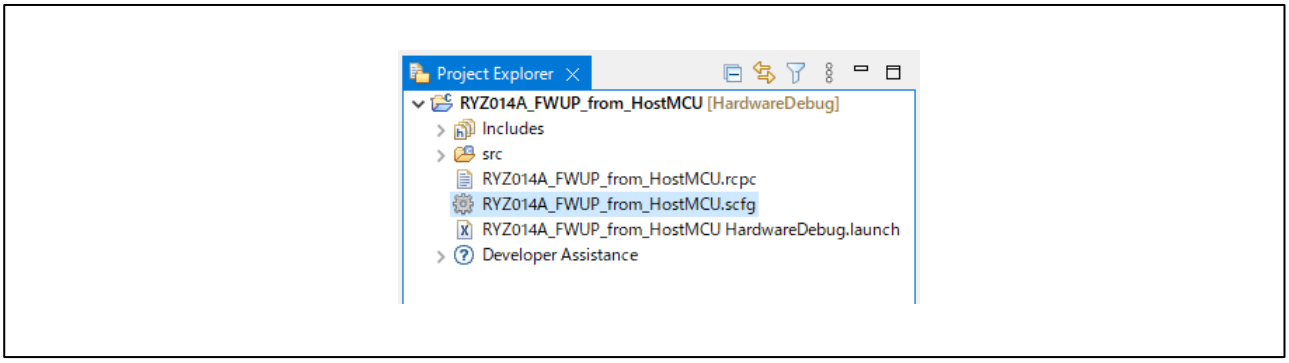

図 3-5 configuration.xml ファイル

2. スマート・コンフィグレータウインドウの右上部にある Generate Code アイコン 

をクリックします。必要なソースコードが生成されます。

3. メニューバーから「Project」→「Build Project」を選択するか、Build アイコン  をクリックしてプロジェクトをビルドします。

4. デバッグアイコン  をクリックしてプロジェクトを起動します。プロジェクトが起動するとサンプルアプリケーションが CK-RX65N にダウンロードされます。

3.5 Tera Term の起動

1. Tera Term を起動し、メニューバーから「Setup」→「Serial port」を選択します。
2. 「表 2-3 マイコン周辺機能」の SCI5(UART5)を参照して Serial port の設定を行います。

3.6 ファームウェアアップグレード実行

1. e^2 studio 右上の 

アイコンをクリックしてデバッグパースペクティブを開きます。再開アイコン  をクリックしてサンプルアプリケーションを実行します。

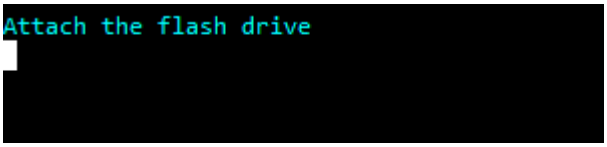

図 3-6 サンプルアプリケーション実行

2. USB フラッシュドライブを CK-RX65M の USB Full Speed コネクタに接続した USB micro-B to A 変換ケーブルへ接続します。
USB フラッシュドライブに格納されている RYZ014A ファームウェアファイルが見つかりファイルサイズが表示されます。

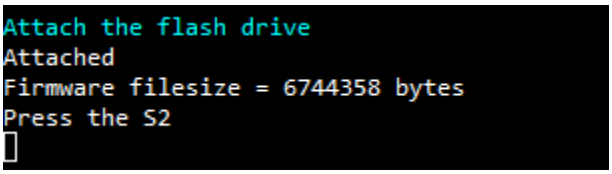

図 3-7 RYZ014A ファームウェアファイル検出

3. CK-RX65N の S2 ボタンを押すとファームウェアアップグレードが実行されます。ファームウェアアップグレードが正常に実行され「R_LTR_FWUpgrade success」が表示されます。

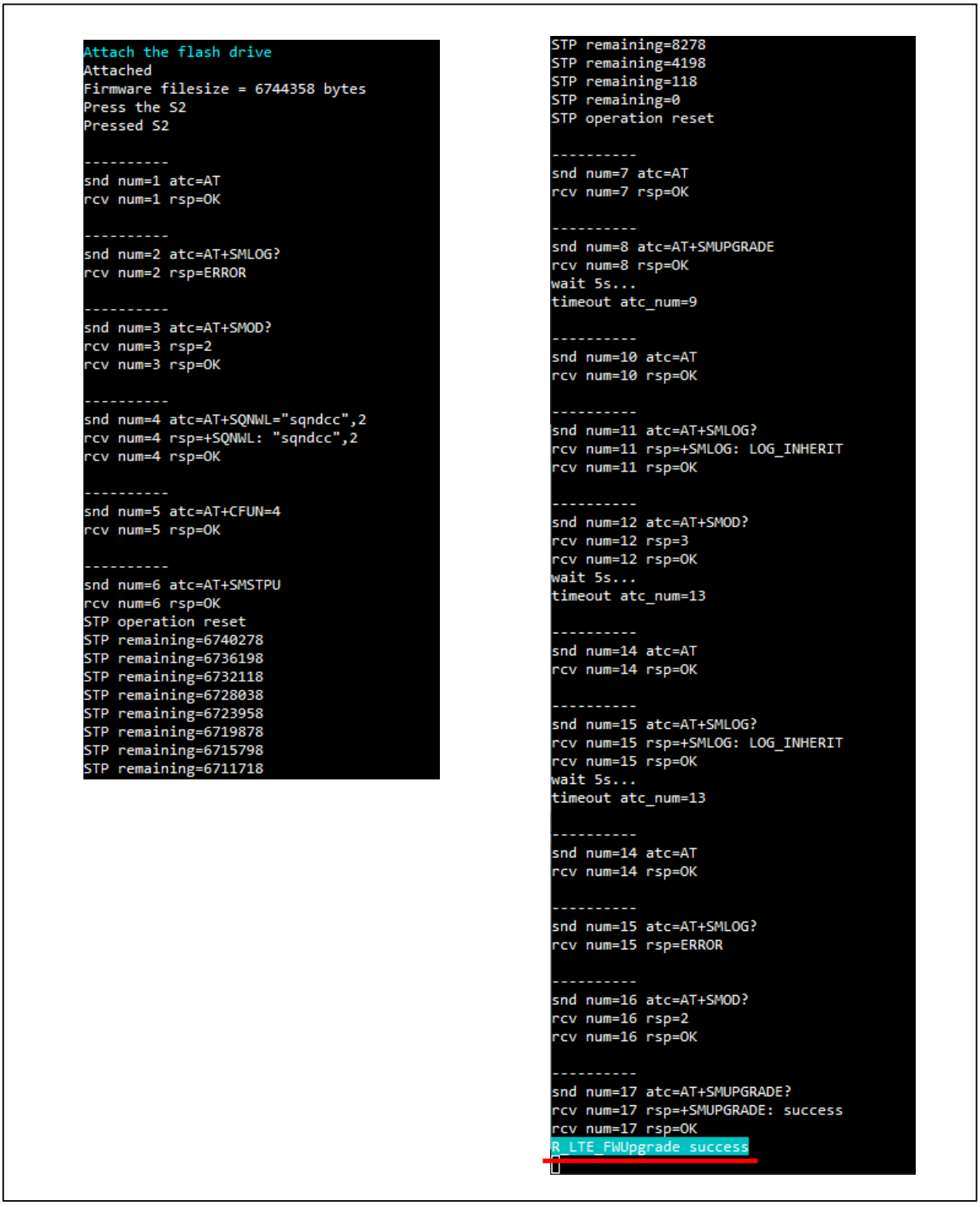

図 3-8 ファームウェアアップグレード実行

4. プログラム処理

ファームウェアアップグレードは R_LTE_FWUpgrade 関数を呼び出すことで実行されます。USB フラッシュドライブに格納された RYZ014A のファームウェアファイルが検出され、かつ S2 ボタンが押されると AT コマンド(ATC)処理を開始します。

AT コマンド処理では「4.2.1 AT コマンドフロー」で示す順序で AT コマンド送信し、レスポンスを受信します。そしてファームウェアアップグレードの実行を RYZ014A へ指示します。

Firmware data 転送処理では「4.3 Firmware data Transfer 処理」で示す Simple Transfer Protocol (STP)を使用して RYZ014A へファームウェアデータを送信します。

ファームウェアアップグレードの途中でホスト MCU や RYZ014A の電源断が発生した場合は、電源を再投入した後に R_LTE_FWUpgrade 関数を呼び出してください。ファームウェアアップグレードを再実行することで正常にアップグレードが完了します。

サンプルアプリケーションを使用したファームウェアアップグレードの実行方法は「3. 動作確認」を参照してください。

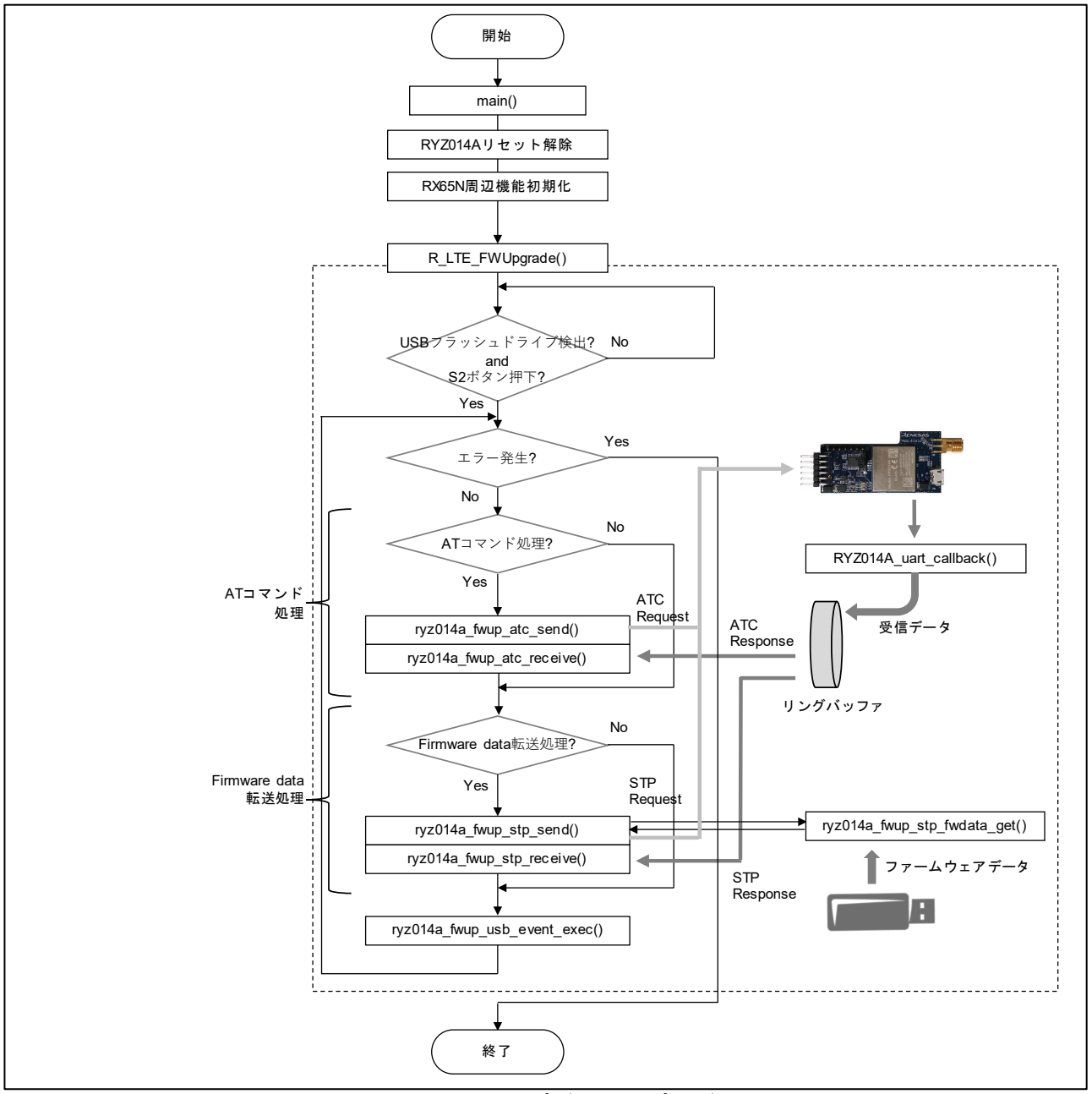

図 4-1 ファームウェアアップグレードプログラム処理概要図

4.1 API

4.1.1 ユーザーAPI

4.1.1.1 Firmware Upgrade 関数

(1) R_LTE_FWUpgrade

機能	RYZ014A のファームウェアをアップグレードします。
引数	なし
戻り値	正常終了：LTE_SUCCESS(ファームウェアアップグレード成功) エラー　：LTE_ERR_FWUPGRADE(ファームウェアアップグレード失敗) LTE_ERR_NOT_FOUND_FWFILE(ファームウェアファイルが見つかりません) ※ エラー定義マクロは r_lte_ryz014a_fwup.h を参照してください。

4.1.2 内部処理関数

4.1.2.1 AT コマンド処理関数

(1) ryz014a_fwup_atc_send

機能	RYZ014A に AT コマンドを送信します。送信する AT コマンドは「4.2.1 AT コマンドフロー」を参照してください。
引数	なし
戻り値	なし

(2) ryz014a_fwup_atc_receive

機能	AT コマンドのレスポンスを RYZ014A から受信し結果を判定します。受信するレスポンスは「4.2.1 AT コマンドフロー」を参照してください。 以下のエラーが発生した場合は同じ AT コマンドを 5 回再送信します。再送信中に正常なレスポンスを受信できない場合は RYZ014A をリセットし AT コマンド処理の先頭から再実行します。再度同様のエラーが発生した場合はファームウェアアップグレードが失敗となります。 ● 不正なレスポンスを受信した ● 5 秒間レスポンスが受信できなかった
引数	なし
戻り値	なし

4.1.2.2 Firmware Data Transfer 処理関数

(1) ryz014a_fwup_stp_send

機能	RYZ014A に STP のリクエストを送信します。プロトコルの詳細は「4.3 Firmware data Transfer 処理」を参照してください。
引数	なし
戻り値	なし

(2) ryz014a_fwup_stp_receive

機能	STP のレスポンスを RYZ014A から受信し結果を判定します。プロトコルの詳細は「4.3 Firmware data Transfer 処理」を参照してください。 以下のエラーが発生した場合は RYZ014A をリセットし AT コマンド処理の先頭から再実行します。再度同様のエラーが発生した場合はファームウェアアップグレードが失敗となります。 ● 不正なレスポンスを受信した ● 5 秒間レスポンスが受信できなかった
引数	なし
戻り値	なし

(3) ryz014a_fwup_stp_fwdata_get

機能	RYZ014A ファームウェアファイルから指定されたサイズのファームウェアデータを取得します。
引数	● ファームウェアデータ格納バッファのポインタ (出力) ● 取得するデータサイズ (入力)
戻り値	なし

4.2 AT コマンド処理

4.2.1 AT コマンドフロー

ファームウェアアップグレードで使用する AT コマンドのフローを示します。

右側の分岐ルートは、アップグレード中に RYZ014A の電源がオフになった場合のリカバリ処理となります。通常は実行されません。

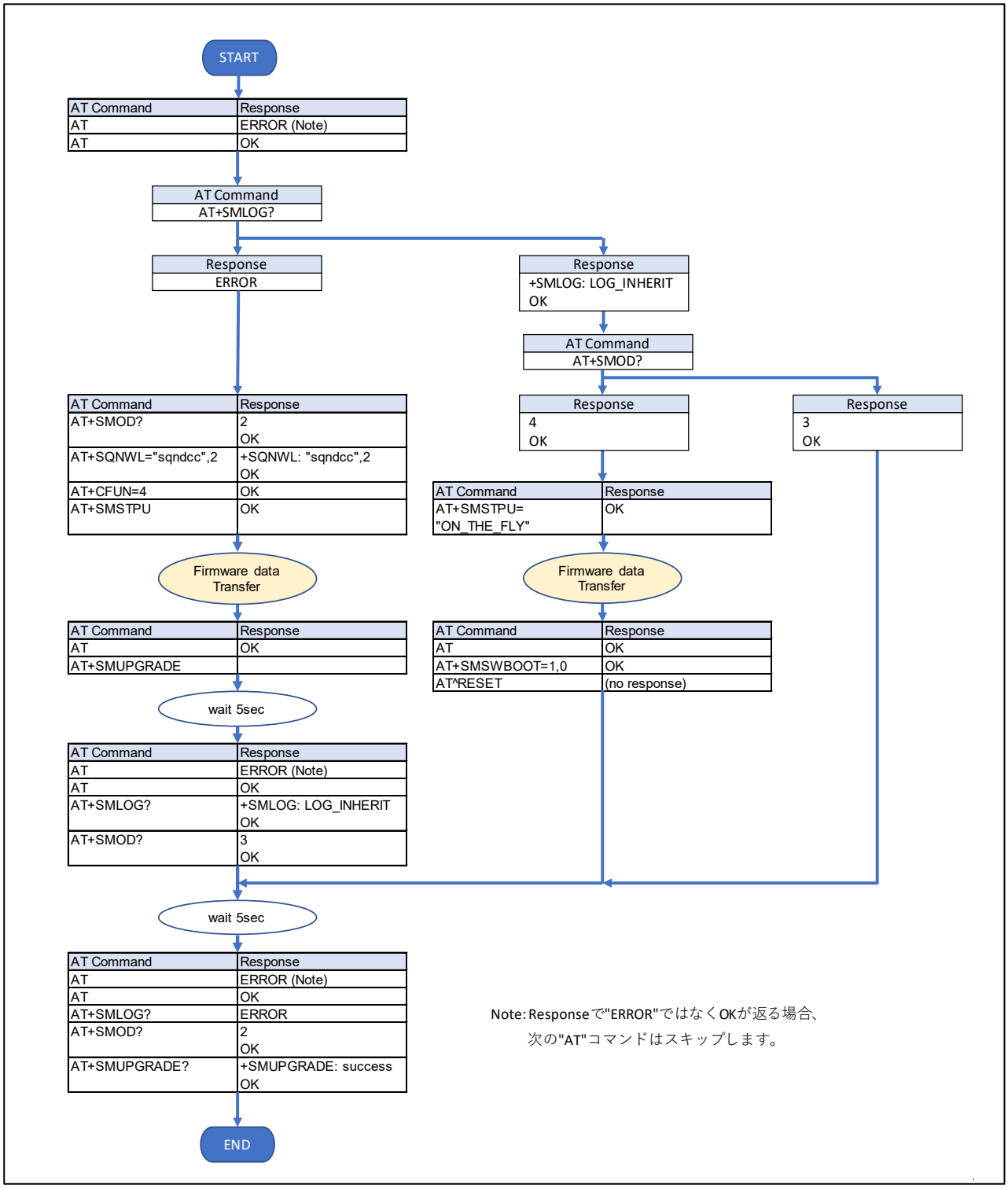

図 4-2 AT コマンドフロー

4.2.2 AT コマンド

ファームウェアアップグレードで使用する AT コマンドの概要を「表 4-1 AT コマンド」に示します。AT コマンドの詳細は「5.1 AT Commands Definition」を参照してください。

表 4-1 AT コマンド

AT コマンド名	説明
AT+SMLOG	コンソールログの出力方法を指定します。
AT+SMOD	ブートモードを返します。
AT+SQNWL	スリープモードへの移行を制御します。
AT+CFUN	Mobile Termination (MT)の機能レベルを選択します。
AT+SMSTPU	Simple Transfer Protocol (STP)転送を開始します。
AT+SMUPGRADE	ファームウェアアップグレードを実行します。
AT+SMSWBOOT	デバイスを指定したモードでブートします。

4.3 Firmware data Transfer 処理

Firmware data 転送処理では、Simple Transfer Protocol (STP)と呼ぶ独自のプロトコルを使用してファームウェアデータを RYZ014A に送信します。ホスト MCU から RYZ014A へ送信するリクエストパケット、RYZ014A からホスト MCU へ送信するレスポンスパケットで通信します。

パケットの構成を「4.3.1 パケット構成」、STP の処理内容を「4.3.2 Operation フロー」に示します。

4.3.1 パケット構成

4.3.1.1 STP packet

STP packet の構成を「表 4-2 STP パケット構成」に示します。「表 4-2 STP パケット構成」の payload を除いた 16 バイトが STP header と呼ばれます。

表 4-2 STP パケット構成

Name	Size	Description
signature	4	リクエスト : 0x66617374 ("fast") レスポンス : 0x74736166 ("tsaf")
operation	1	0x00 : reset 0x01 : open session 0x02 : transfer block command 0x03 : transfer block ※Response の operation は 0x80 と OR した値です。
session ID (sid)	1	set to 0 on reset, set to 1 once session is open.
payload length	2	payload length
transaction ID (tid)	4	transaction ID, set to 0 on reset, then incremented after each answer.
full header checksum	2	Request : full header checksum Response : 0
payload checksum	2	payload checksum or 0 if no payload.
payload	-	payload data データ構成は「4.3.1.2 Payload」を参照。

4.3.1.2 Payload

Payload の構成を示します。(1)～(4)の各 operation の状態により、使用される payload のサイズが異なります。

(1) open session operation payload (レスポンス)

open session operation のレスポンスで付加される payload 構成を「表 4-3 open session payload 構成」に示します。

表 4-3 open session payload 構成

Name	Size	Description
success	1	Indicate if session has been open with success (1) or not (0).
version	1	protocol version, always 1.
max transfer size	2	max size for STP header + payload.

(2) transfer block command operation payload (リクエスト)

transfer block command operation のリクエストで付加される payload 構成を「表 4-4 transfer block command payload 構成」に示します。

表 4-4 transfer block command payload 構成

Name	Size	Description
block size	2	block size to transfer.

(3) transfer block operation payload (リクエスト)

transfer block operation のリクエストで付加される payload 構成を「表 4-5 transfer block operation request payload 構成」に示します。

表 4-5 transfer block operation request payload 構成

Name	Size	Description
firmware data	Refer to Description	firmware data. size = max transfer size - STP header

(4) transfer block operation payload (レスポンス)

transfer block operation のレスポンスで付加される payload 構成を「表 4-6 transfer block operation response payload 構成」に示します。

表 4-6 transfer block operation response payload 構成

Name	Size	Description
residue	2	always 0.

4.3.2 Operation フロー

STP を使用したファームウェアデータの転送処理は「表 4-2 STP パケット構成」の「operation」で示される項目ごとに STP パケットの送受信を行うことで実行されます。

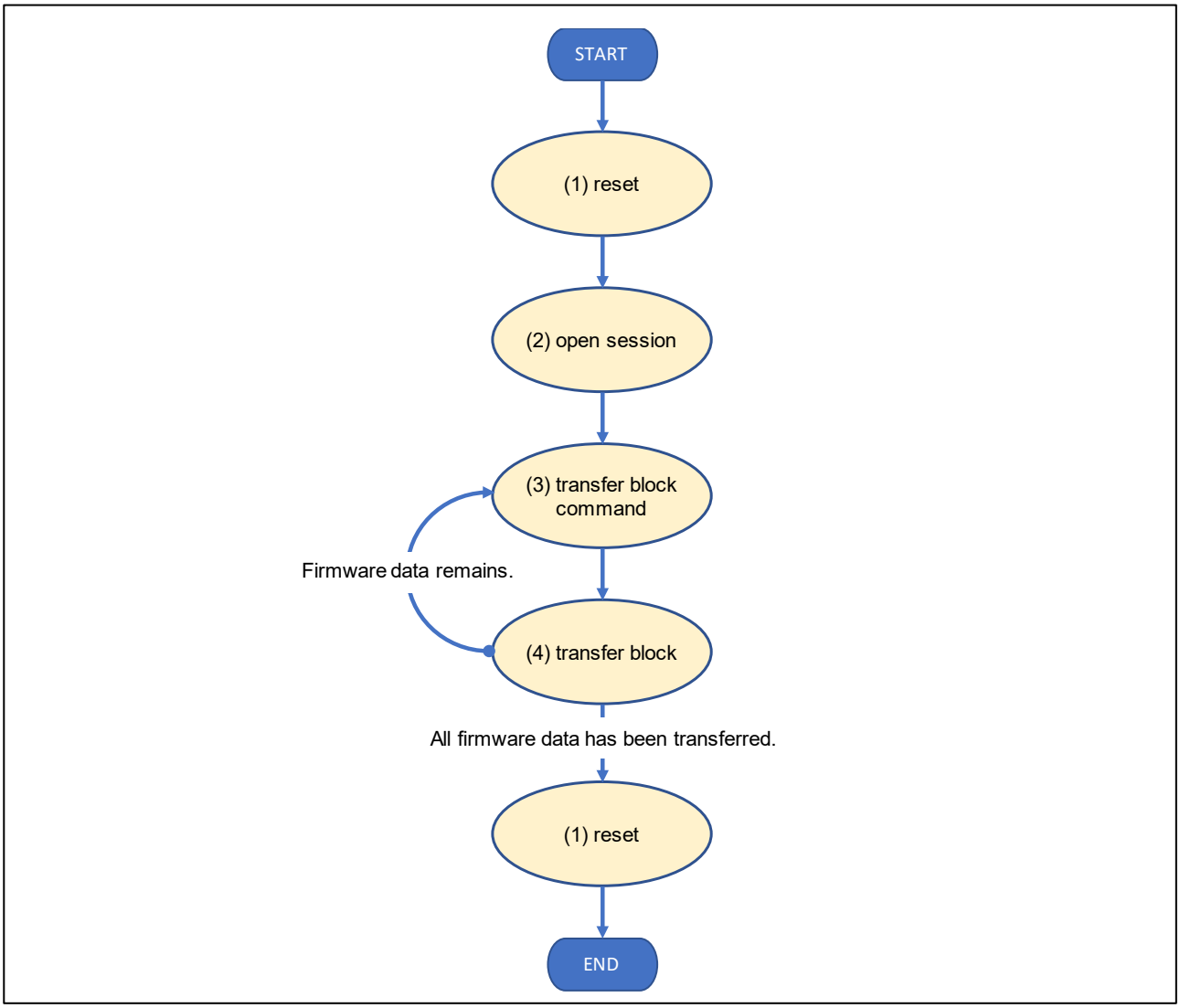

図 4-3 オペレーションフロー

(1) reset

目的	RYZ014A のセッションをリセットします。
内容	● sid と tid を 0 にセットします。 ● リクエストパケットを送信します。(payload なし) ● レスポンスパケットを受信します。(payload なし)

(2) open session

目的	RYZ014A のセッションをオープンします。
内容	● sid を 1 にセットします。 ● tid をインクリメントします。 ● リクエストパケットを送信します。(payload なし) ● レスポンスパケットを受信します。 (「open session operation payload (レスポンス)」あり)

(3) transfer block command

目的	ファームウェアデータの受信を RYZ014A に指示します。
内容	● tid をインクリメントします。 ● リクエストパケットを送信します。 (「transfer block command operation payload (リクエスト)」あり) ● レスポンスパケットを受信します。(payload なし)

(4) transfer block

目的	ファームウェアデータの受信を RYZ014A に指示します。
内容	● tid をインクリメントします。 ● リクエストパケットを送信します。 (「transfer block operation payload (リクエスト)」あり) ● レスポンスパケットを受信します。 (「transfer block operation payload (レスポンス)」あり)

5. 付録

5.1 AT Commands Definition

ファームウェアアップグレードで使用する AT コマンドについて説明します。

(1) AT+SMSWBOOT

Description:

This command forces the device to boot in mode <mode> (FFF, FFH, Updater or Recovery).

Syntax:

Command	Possible Response(s)
AT+SMSWBOOT=<mode>[,<reboot>]	OK

Parameters:

<mode>: integer 0, 1, 2 or 3. Device start-up mode at next boot

0: FFH

1: FFF

2: UPDATER

3: RECOVERY

<reboot>: integer 0 or 1. Automatic device reboot after <mode> change

0 (default): no reboot

(2) AT+SMOD

Description:

This command returns the boot mode.

Syntax:

Command	Possible Response(s)
AT+SMOD	<mode>

Parameters:

<mode>: Integer. Device start-up mode at next boot

0: FFH

1: FFF

2: UPDATER

3: RECOVERY

4: OTHER

(3) AT+SMUPGRADE

Description:

AT+SMUPGRADE parses the .dup file and directly flashes the data into the corresponding regions depending on which boot mode the module is:

- FFH: Upgrade all regions and filesystem
- RECOVERY: Upgrade all regions and filesystem
- FFF: Upgrade UPDATER and BOOTROM regions
- UPDATER: Upgrade FFF region and filesystem

When in FFF, all authorized regions are upgraded before the module reboots in UPDATER mode to finish the execution of the .dup file. Once the UPDATER mode is over, the module reboots in FFF mode. The SFU tool takes care of both steps, its use is highly recommended for any upgrade.

Syntax:

Command	Possible Response(s)
AT+SMUPGRADE	-
AT+SMUPGRADE?	Upgrade report

(4) AT+SMLOG

Description:

Available in manufacturing mode (AT+CFUN=5)

Forces console logs to be printed on the specified UART (regardless of their programmed "console" functionality).

Syntax:

Command	Possible Response(s)
AT+SMLOG=<log>	OK
AT+SMLOG?	+SMLOG=<log>
AT+SMLOG=?	+SMLOG [list of supported <log>]

Parameters:

<log>: String. Chooses the UART the console log is directed to.

- LOG_DISABLE: The logs are discarded
- LOG_INHERIT: The logs are printed on the UART configured as ‘console’ (default)
- LOG_FORCE_UART0: The logs are printed on UART 0
- LOG_FORCE_UART1: The logs are printed on UART 1
- LOG_FORCE_UART2: The logs are printed on UART 2

(5) AT+SMSTPU

Description:

This command starts a STP transfer, waiting for an FFF image containing upgrade images.

Syntax:

Command	Possible Response(s)
AT+SMSTPU[="ON_THE_FLY"]	-

Parameters:

none: Normal transfer

ON_THE_FLY: Recovery transfer

(6) AT^RESET

Description:

This command performs an hardware reset.

Syntax:

Command	Possible Response(s)
AT^RESET	Device is reset
-	+SHUTDOWN ... +SYSSTART

(7) AT+SQNWL

Description:

This command manages resources wake locks to indicate that a client application running on the Host CPU needs to secure full and immediate availability of some device resources, which implies to prevent the considered resources to enter sleep mode.

Currently, the wake-lockable system resources are:

- CPU and external interfaces (UART, GPIO)
- Device memory (RAM)

The Set command is used to set and release the wake locks based on the resource identified by the bitmask <wl_mask>. To set the wake locks and prevent the sleep mode for the specified resource(s), <wl_mask> bit(s) should be set to 1. To release the wake locks and allow sleep mode for the specified resource(s), <wl_mask> bit(s) should be set to 0.

Note: The wake locks configuration is volatile. It is lost at reboot.

The Set command entered with bitmask omitted will return the wake lock defined by the <app> client application.

The Read command returns the list of client applications using wake locks and lock status.

Caution: It is very important release wake lock as soon as possible to avoid running down the device's battery excessively. Each application

<app> setting a wake lock to 1 should explicitly reset it to 0 when the need for resource availability is not present anymore.

Syntax:

Command	Possible Response(s)
AT+SQNWL=<app>[,<wl_mask>]	+CME ERROR:<err> +SQNWL:<app>,<wl_mask> OK
AT+SQNWL?	+SQNWL:<app1>,<mask1> [<CR><LF>...<CR><LF>+SQNWL: <appN>,<maskN>[...]] OK
AT+SQNWL=?	+SQNWL:"",(0-3) OK

Parameters:

app

- Client application name, string.

wl_mask

- Bitmask as integer in range [0-3] identifying the resource to keep available. Bitis set to 1 to keep the resource available and prevent sleep mode, or reset to 0 to release.

wl_mask

Value	Description
0	Default value. No system resource locked
Bit 0 (0x01)	Keep CPU and external interfaces (UART, GPIO) active. Prevents the CPU and external interfaces to enter sleep mode.
Bit 1 (0x02)	Keep device's RAM memory active. Prevents the device's RAM memory to enter sleep mode.
Bits 0 and 1 (0x03)	Keep CPU, external interfaces (UART, GPIO) and device's RAM memory active. Prevents the

6. 改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.0	2022/12/15	-	初版発行

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。