RX72M グループ

EtherCAT CiA402 ETG.5003 サンプルプログラム

Firmware Information Technology

要旨

本アプリケーションノートは、産業イーサネット通信プロトコルの１つである EtherCAT®通信において、IO 機器やモータ制御用装置、半導体製造装置に適用されるサブデバイス機器のサンプルプログラムについて説明します。

本サンプルプログラムでは下記の仕様をサポートしています。

- CANopen over EtherCAT (CoE)
- File access over EtherCAT (FoE)
- Distributed Clocks (DC)
- CiA402 Drive Profile
- Common Device Profile(ETG.5003.1)
- Firmware update functionality(ETG.5003.2)

動作確認デバイス

- RX72M グループ

お客様の製品にてご利用される際は、お客様の環境に合わせて十分に評価してください。

また、本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

旧ドキュメントについて

本アプリケーションノートは下記の旧ドキュメントを統合したドキュメントとなっています。

- RX72M グループ 通信ボード EtherCAT スタートアップマニュアル
  （ドキュメント r01an4672jj0105-rx72m-ecat.pdf）
- RX72N グループ CPU カード EtherCAT スタートアップマニュアル
  （ドキュメント r01an6749jj0101-rx72m-cpucard-ecat.pdf）
- RX72M グループ RSK ボード EtherCAT スタートアップマニュアル
  （ドキュメント r01an4689jj0105-rx72m-ecat-rsk.pdf）
- RX72M グループ EtherCAT CiA402 サンプルプログラム Firmware Information Technology
  （ドキュメント r01an5393jj0110-rx72m-ecat-cia402.pdf）
- RX72M グループ EtherCAT ETG.5003 サンプルプログラム Firmware Information Technology
  （ドキュメント r01an5538jj0120-rx72m-ecat-etg5003.pdf）

今後、旧ドキュメントは更新されず、本アプリケーションノートが更新されます。

目次

内容

1. 概要

1.1 本アプリケーションについて

本アプリケーションノートでは、CANopen over EtherCAT (CoE)プロトコルのプロセスデータ通信やCiA402 ドライブプロファイル、File Access over EtherCAT (FoE)プロトコルを利用したファームウェア更新機能(ETG.5003.2)をサポートしたサンプルプログラムについて説明します。

半導体製造装置に適用するプロファイル ETG.5003 Semiconductor Device Profile のサポート内容については、『9. 付録 A. Semiconductor Device Profile [ETG.5003]』を参照してください。

本サンプルプログラムは、ボードサポートパッケージ（以下、BSP と略す）、EtherCAT 等の FIT モジュールと組み合わせて動作します。

サンプルプログラムには SSC は含まれておりません。EtherCAT Technology Group(ETG 協会)より SSC ツールを入手の上、SSC を使用してください。

1.2 動作確認

表 1-1 に、本サンプルプログラムで確認した動作環境を示します。

表 1-1　動作確認環境

対応 MCU	RX72M グループ　R5F572MNxDBD コードフラッシュメモリ容量：4M バイト
評価ボード	テセラ・テクノロジー社製 RX72M 搭載評価ボード (TS-RX72M-COM) （以下、COM ボードと略す）
	RX72M CPU Card with RDC-IC (RTK0EMXDE0C00000BJ) （以下、CPU カードと略す）
	Renesas Starter Kit+ for RX72M (RTK5572MNxCxxxxxBJ) （以下、RSK ボードと略す）
統合開発環境（IDE）	ルネサスエレクトロニクス製 e^2 studio 2024-01
クロスツール	C/C++ Compiler Package for RX Family V3.06.00
エミュレータ	E2 Lite
SSC tool	EtherCAT Technology Group (ETG) 提供 Slave Stack Code (SSC) Tool Version 5.13
ソフトウェア PLC	Beckhoff Automation 製 TwinCAT® 3（Beckhoff web サイトからダウンロード）

1.3 FIT モジュール構成

表 1-2 に本サンプルプログラムで使用する FIT モジュールの一覧を示します。

表 1-2　FIT モジュール構成

種類	モジュール名	FIT モジュール名	Rev.
Board Support Package	ボードサポートパッケージ(BSP)	r_bsp	7.42
Device Driver	コンペアマッチタイマ(CMT)	r_cmt_rx	5.60
Device Driver	シリアルコミュニケーションインターフェース(SCI)	r_sci_rx	4.90
Middleware	バイト型キューバッファ（BYTEQ）	r_byteq	2.10
Device Driver	EtherCAT	r_ecat_rx	1.31
Device Driver	フラッシュ	r_flash_rx	5.10

1.4　プロジェクトについて

本サンプルプログラムに含まれるプロジェクトの一覧を表 1-3 に示します。

以降の章では RX72M CPU カードのリニアモードのプロジェクト（ecat_linear_demo_cpurx72m）を例にして説明します。異なるプロジェクトを使用する場合は適宜、プロジェクト名を置き換えてお読みください。

表 1-3　プロジェクト一覧

MCU	評価ボード名	デュアルバンク機能	プロジェクト名
RX72M	通信ボード	リニアモード	ecat_linear_demo_comrx72m
		デュアルモード	ecat_dual_demo_comrx72m
	CPU カード	リニアモード	ecat_linear_demo_cpurx72m
		デュアルモード	ecat_dual_demo_cpurx72m
	RSK ボード	リニアモード	ecat_linear_demo_rskrx72m
		デュアルモード	ecat_dual_demo_rskrx72m

2.　開発環境の入手

2.1　e² studio の入手方法

以下の URL にアクセスし、e² studio をダウンロードしてください。

https://www.renesas.com/e2studio_download

なお、本アプリケーションノートは e² studio 2024-01 以降を使用することを前提としています。2024-01 よりも古い Ver.を使用した場合、e² studio の一部機能を使用できない可能性があります。ダウンロードする場合、ホームページに掲載されている最新 Ver.の e² studio を入手してください。

2.2　コンパイラパッケージの入手方法

以下の URL にアクセスし、RX ファミリ用 C/C++コンパイラパッケージをダウンロードしてください。

https://www.renesas.com/jp/ja/software-tool/cc-compiler-package-rx-family

なお、本アプリケーションノートは V3.06.00 以降を使用することを前提としています。V3.06.00 よりも古い Ver.を使用した場合、CC-RX の一部機能を使用できない可能性があります。ダウンロードする場合、ホームページに掲載されている最新 Ver.の CC-RX を入手してください。

3. プロジェクトの構築

3.1 EtherCAT スレーブスタックコードをサンプルプログラムにインポート

本プロジェクトには EtherCAT スレーブスタックコードは同梱されていません。

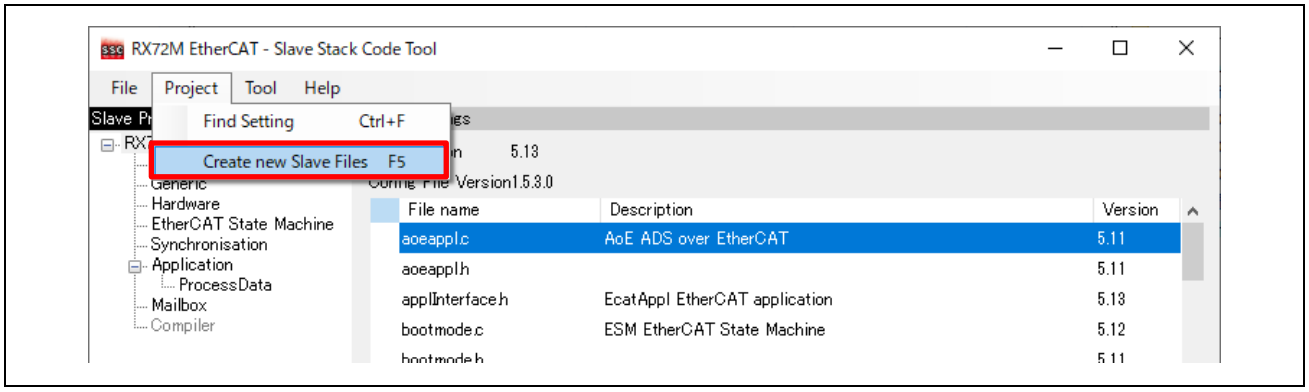
*EtherCAT スレーブスタックコードの生成には”EtherCAT Slave Stack Code(SSC) Tool”が必要です。

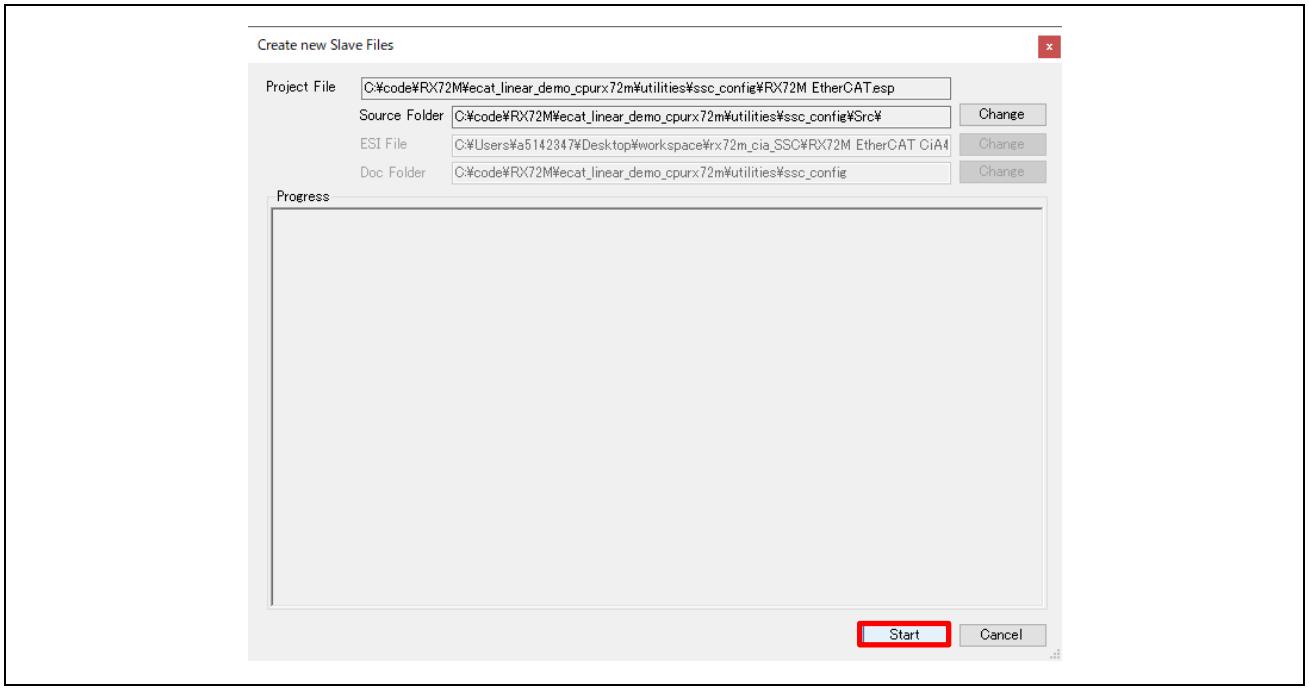
*SSC Tool は ETG 協会から入手可能です。

サンプルプログラムは"ecat_linear_demo_cpurx72m.zip"の形式で提供されますので、予め任意のフォルダに解凍してください。

(1) サンプルプログラムの SSC プロジェクトファイルをダブルクリックして SSC ツールを起動します。
ecat_linear_demo_cpurx72m¥utilities¥ssc_config¥RX72M EtherCAT.esp

(2) [Project]→[Create New Slave Files]をクリックして、[Current new Slave Files]ダイアログで[Start]をクリックします。





(3) ソースコードが生成され、成功すると”New Files created successfully”と表示されるので[OK]をクリックします。

(4) パッチコマンドをインストールしていない場合 、GNU Patch Ver2.5.9 以降が必要です。

インストール済みの場合は本手順をスキップしてください。

下記の Web サイトからパッチコマンド(Ver2.5.9)をダウンロードし、”patch.exe”をコマンドプロンプトから実行可能なパスの通ったフォルダに格納します。

http://gnuwin32.sourceforge.net/packages/patch.htm

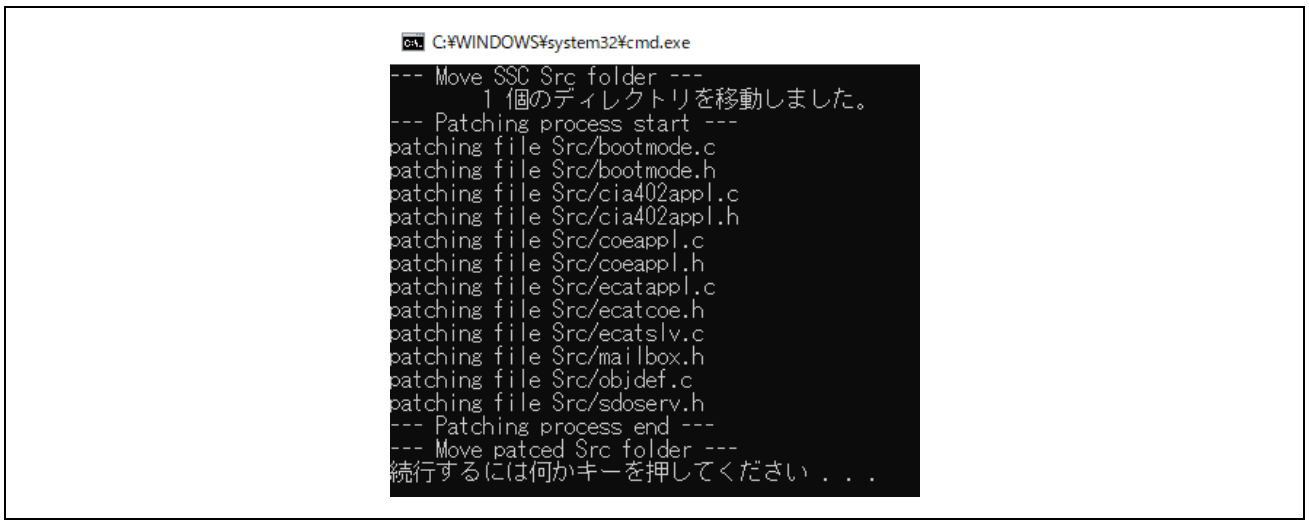
(5) apply_patch.bat ファイルをダブルクリックします。

パッチファイルは SSC ソースファイルに対する RX 向けの修正を含んでいます。

ecat_linear_demo_cpurx72m¥utilities¥rx72m¥patch¥apply_patch.bat

パッチ実行後、修正されたソースファイルは下記のフォルダに格納されます。

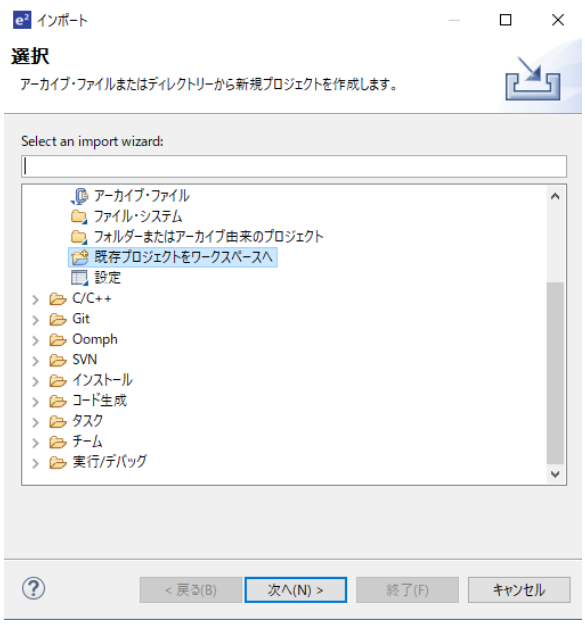
ecat_linear_demo_cpurx72m¥project¥src¥application¥ecat



3.2 プロジェクトのインポート

(1) [ファイル]→[インポート]をクリックします。

(2) [選択]ダイアログで[一般]→[既存プロジェクトをワークスペースへ]を選択し[次へ]をクリックします。



(3) [プロジェクトのインポート]ダイアログの[ルート・ディレクトリの選択]チェックボックスを選択し、[参照]をクリックします。

(4) ”ecat_linear_demo_cpurx72m”を選択して[開く]をクリックします。

(5) [プロジェクト]の”ecat_linear_demo_cpurx72m”をチェックし[次へ]をクリックすると RX72M CPU カードのリニアモードのプロジェクトがインポートされます。

3.3　EtherCAT FIT モジュールを e² studio にインポート

スマートコンフィグレータで EtherCAT FIT モジュールを使用できるようにするには、e² studio に追加する必要があります。

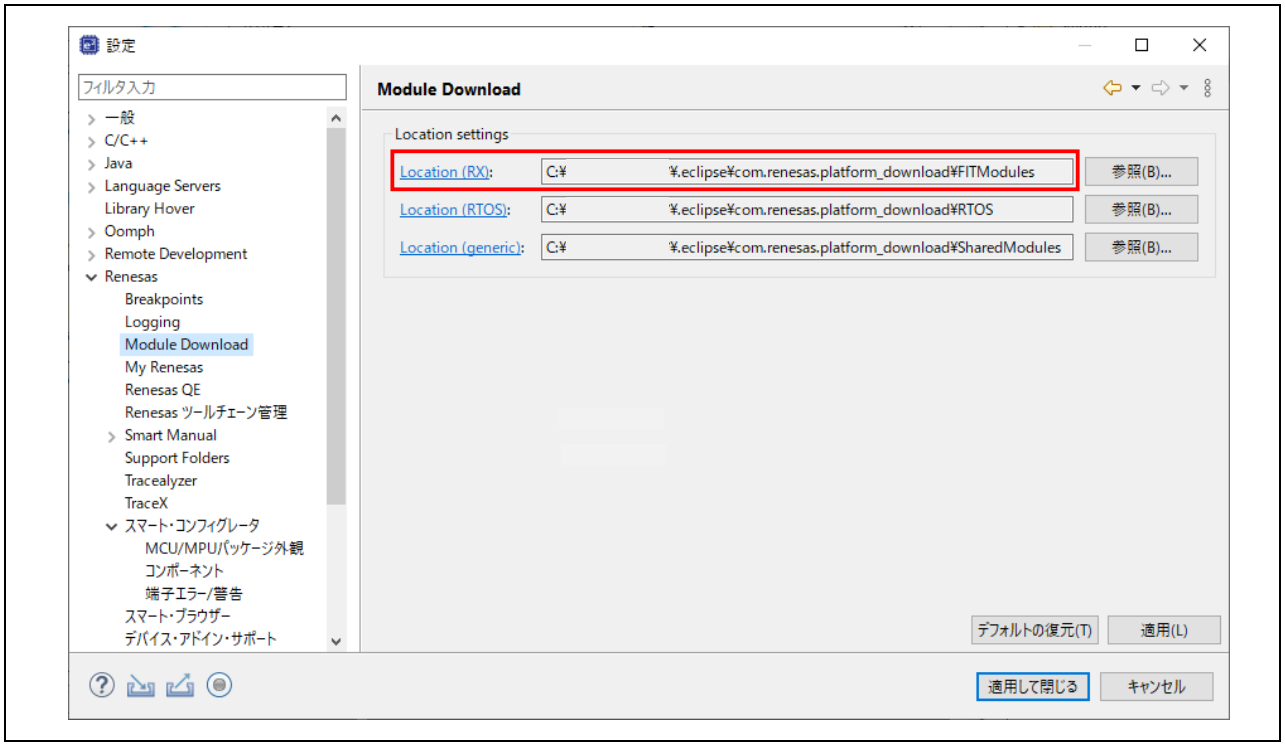
手動で追加する方法を示します。

➢ e² studio の FIT モジュールの保存先フォルダに EtherCAT FIT モジュールをコピーする。

e² studio で FIT モジュールの保存先を確認します。

1. 「ウインドウ」→「設定」→設定ウインドウが開きます。
2. 「Renesas」→「Module Download」を選択してください。

Location (RX):として表示されているパスが FIT モジュールの保存先フォルダになります。



EtherCAT FIT モジュールはサンプルプログラムの FITModules フォルダに格納されています。

※　r01an7288xx0100-rx72m-ecat¥FITModules フォルダにあるファイルを FIT モジュールの保存先フォルダにコピーしてください。

r_ecat_rx_v*N.NN*.xml

r_ecat_rx_v*N.NN*.zip

r_ecat_rx_v*N.NN*_extend.mdf

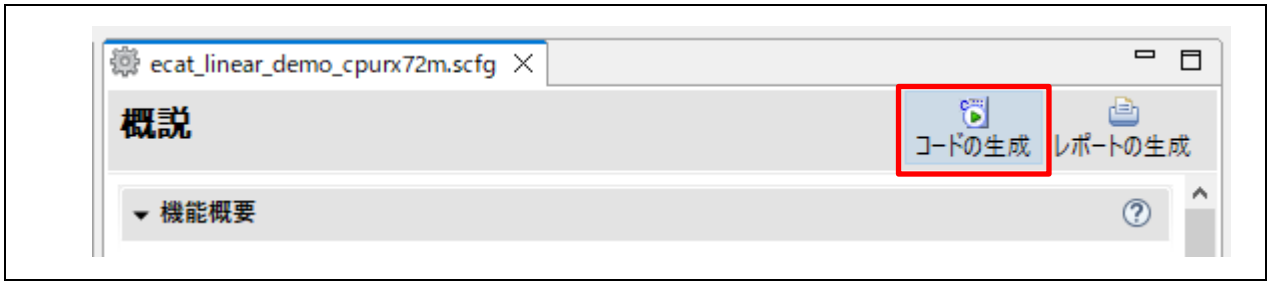
なお、*N.NN* はバージョンを表す数値になります。

4. プロジェクトのビルドとデバッグ

4.1 コード生成

(1) スマートコンフィギュレーションファイル“ecat_linear_demo_cpurx72m.scfg”を開き、[コード生成]を押してコード生成を実行してください。

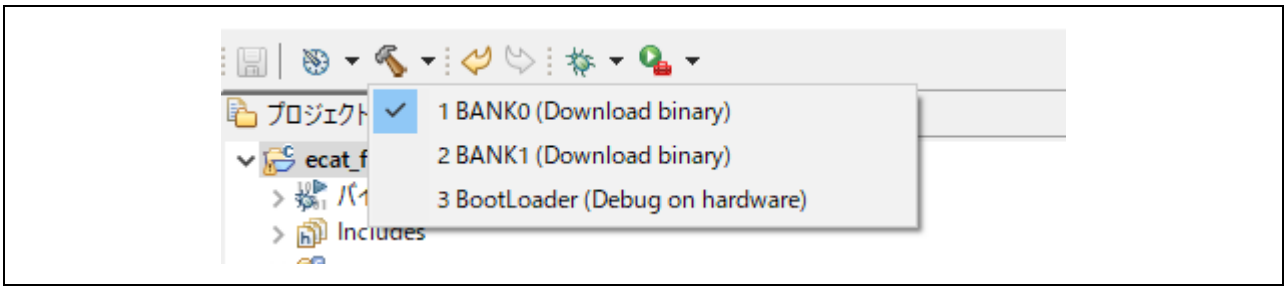
(2) ecat_linear_demo_cpurx72m¥src¥smc_gen フォルダ以下に生成されたコードが格納されます。



4.2 プロジェクトのビルド

4.2.1 リニアモードのビルド

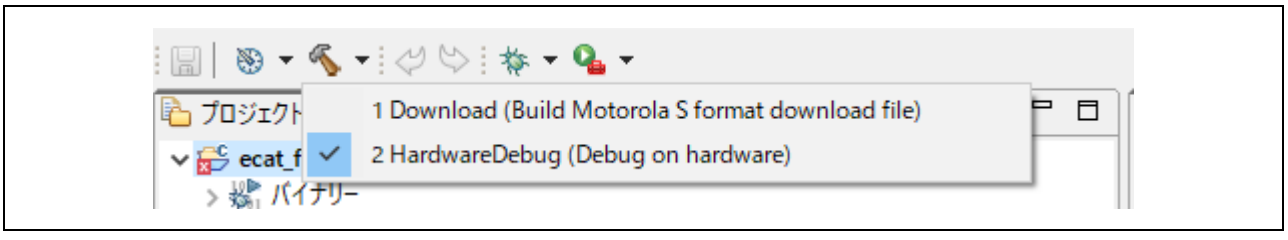
(1) ツールバーの [ビルド]ボタン（ハンマーアイコン）の横にある矢印をクリックし、ドロップダウンメニューから [BANK0]を選択します。



(2) 同様に[BANK1]と[BootLoader]も選択してビルドしてください。

4.2.2 デュアルモードのビルド

(1) ツールバーの [ビルド]ボタン（ハンマーアイコン）の横にある矢印をクリックし、ドロップダウンメニューから [HardwareDebug]を選択します。



(2) 同様に[Download]も選択してビルドしてください。

4.3　デバッグの準備

本サンプルプログラムを動作させるための、評価ボードの設定を表 4-1 に示します。

表 4-1　評価ボード設定一覧

設定項目	MCU	評価ボード名	設定内容
LAN ケーブル	RX72M	通信ボード	”ECAT IN”側に接続
		CPU カード	“CN8”側に接続
		RSK ボード	”ECAT IN”側に接続
デバッガ	RX72M	通信ボード	E2 Lite を JTAG コネクタに接続
		CPU カード	USB ケーブルを USB コネクタに接続
		RSK ボード	E2 Lite を JTAG コネクタに接続

(1)　表 4-1 に従い、LAN ケーブルを PC と評価ボードに接続してください。

(2)　表 4-1 に従い、デバッガと PC を接続してください。

(3)　“新しいハードウェアの検出”ウイザードが表示されますので、以下の手順に従って、ドライバをインストールしてください。Windows™ 7/8/8.1 の場合、管理者権限が必要です。

Windows™ 7/8/8.1：インストールが完了すると Windows タスクバーに完了通知されます。

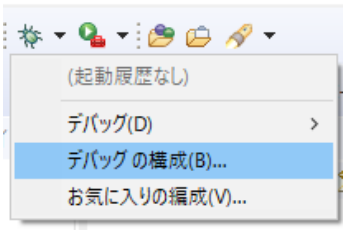
Windows™ 10：Windows タスクバーにデバイス設定ボタンが表示され自動インストールされます。

(4)　評価ボードに電源を供給してください。

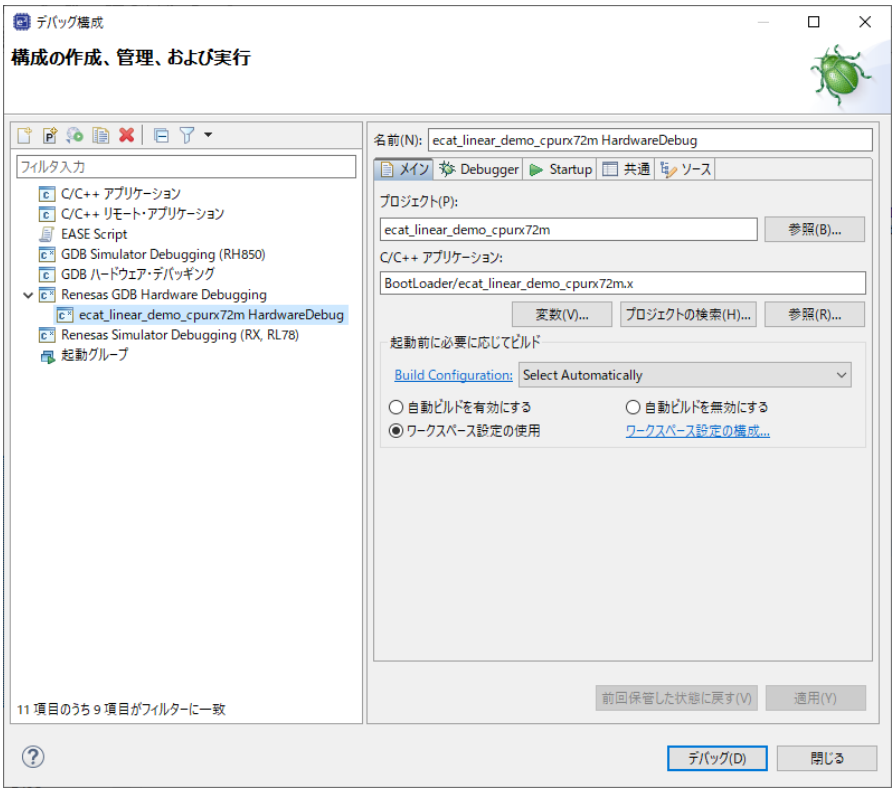
4.4　プロジェクトのデバッグ

4.4.1　リニアモードのデバッグ

(1) [デバッグ]ボタン（バグアイコン）の横にある矢印をクリックし、「デバッグ構成」を選択することでデバッグを開始できます。



(2) “ecat_linear_demo_cpurx72m HardwareDebug”をクリックしてターゲットにプログラムをダウンロードし、デバッグボタンを押して開始します。



(3) 'e2-server-gdb.exe'のファイアウォール警告が表示されることがあります。 [自宅や職場のネットワークなどのプライベートネットワーク]のチェックボックスをチェックにして、<アクセスを許可>をクリックします。

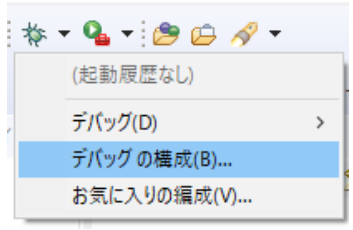
(4) ユーザーアカウント制御（UAC）ダイアログが表示されることがあります。 管理者パスワードを入力して、 [はい]をクリックします。

(5) パースペクティブ切り替えの確認ダイアログにてパースペクティブの変更を勧めるダイアログが表示される場合は「常にこの設定を使用する」チェックボックスにチェックし、[はい]をクリックします。

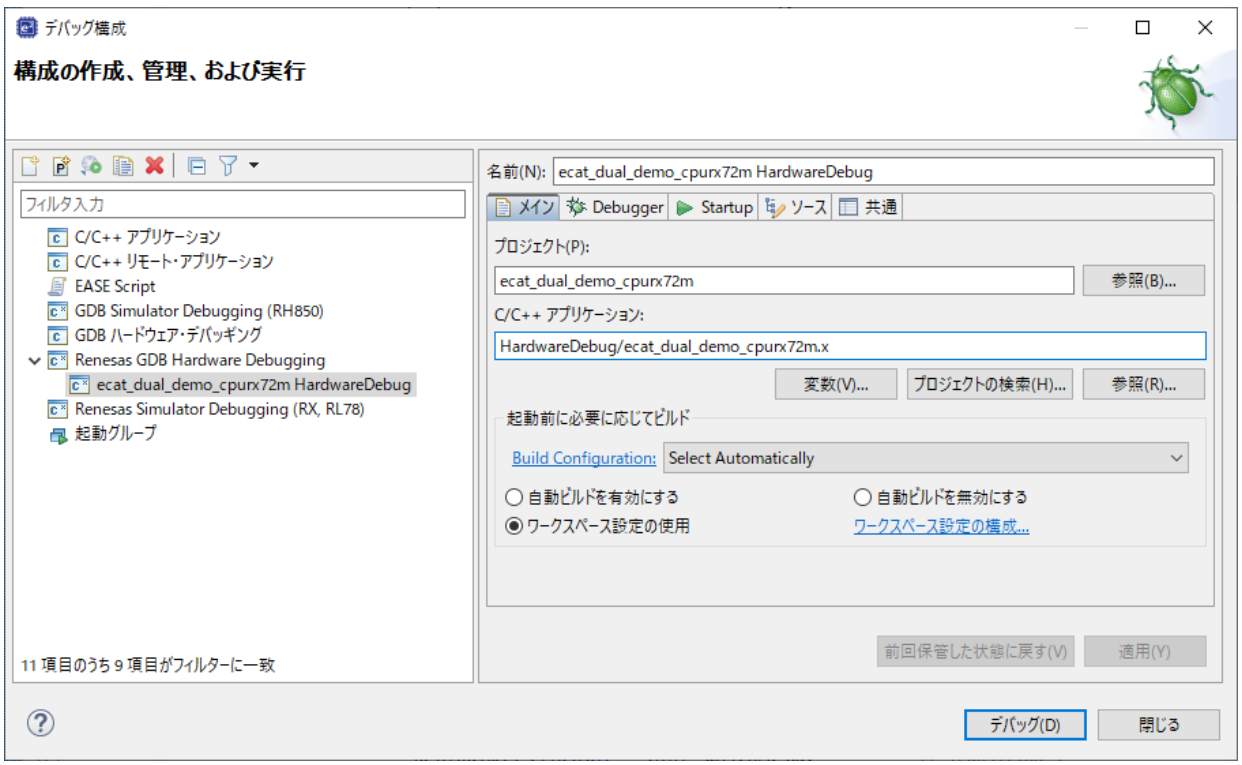
(6) コードをダウンロードしたら、<再開>ボタンをクリックして、メイン関数 main（）の最初の行までコードを実行します。 もう一度<再開>ボタンをクリックすると、残りのコードでターゲットが実行されます。

4.4.2 デュアルモードのデバッグ

(1) [デバッグ]ボタン（バグアイコン）の横にある矢印をクリックし、「デバッグ構成」を選択することでデバッグを開始できます。



(2) “ecat_dual_demo_cpurx72m HardwareDebug”をクリックしてターゲットにプログラムをダウンロードし、デバッグボタンを押して開始します。



(3) 'e2-server-gdb.exe'のファイアウォール警告が表示されることがあります。 [自宅や職場のネットワークなどのプライベートネットワーク]のチェックボックスをチェックにして、<アクセスを許可>をクリックします。

(4) ユーザーアカウント制御（UAC）ダイアログが表示されることがあります。 管理者パスワードを入力して、 [はい]をクリックします。

(5) パースペクティブ切り替えの確認ダイアログにてパースペクティブの変更を勧めるダイアログが表示される場合は「常にこの設定を使用する」チェックボックスにチェックし、[はい]をクリックします。

(6) コードをダウンロードしたら、<再開>ボタンをクリックして、メイン関数 main（）の最初の行までコードを実行します。 もう一度<再開>ボタンをクリックすると、残りのコードでターゲットが実行されます。

5. TwinCAT との接続

5.1 ESI ファイルの準備

TwinCAT を起動する前に、サンプルプログラムに含まれている下記の ESI ファイルを TwinCAT の所定の場所(¥TwinCAT¥3.x¥Config¥IO¥EtherCAT)にコピーしてください。

ecat_linear_demo_comrx72m¥utilities¥esi¥RX72M EtherCAT.xml

5.2 TwinCAT の起動

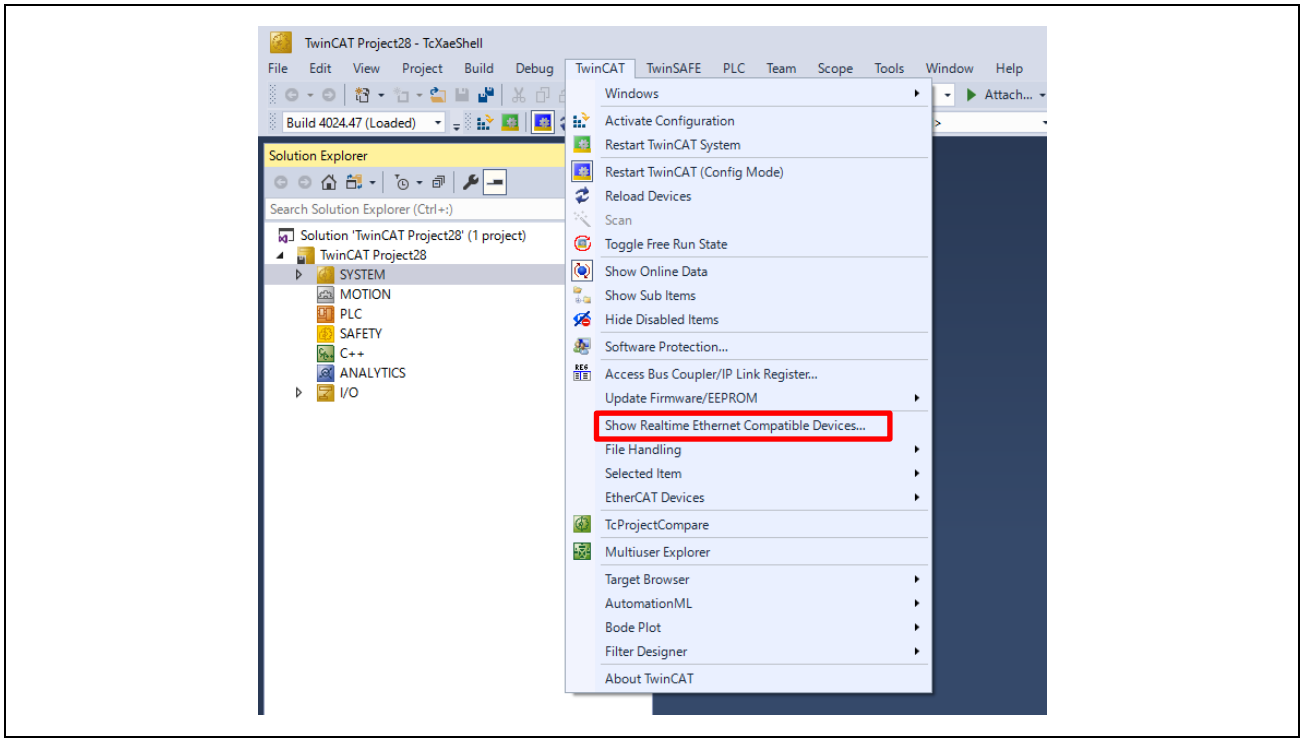
(1) スタートメニューから、[Beckhoff] → [TwinCAT3] → [TwinCAT XAE (VS20xx)]を選択します。

(2) プログラム起動後、[File] → [New] → [Project] として、TwinCAT XAE Project タイプの新規プロジェクトを作成してください。

5.3 Ether driver の追加

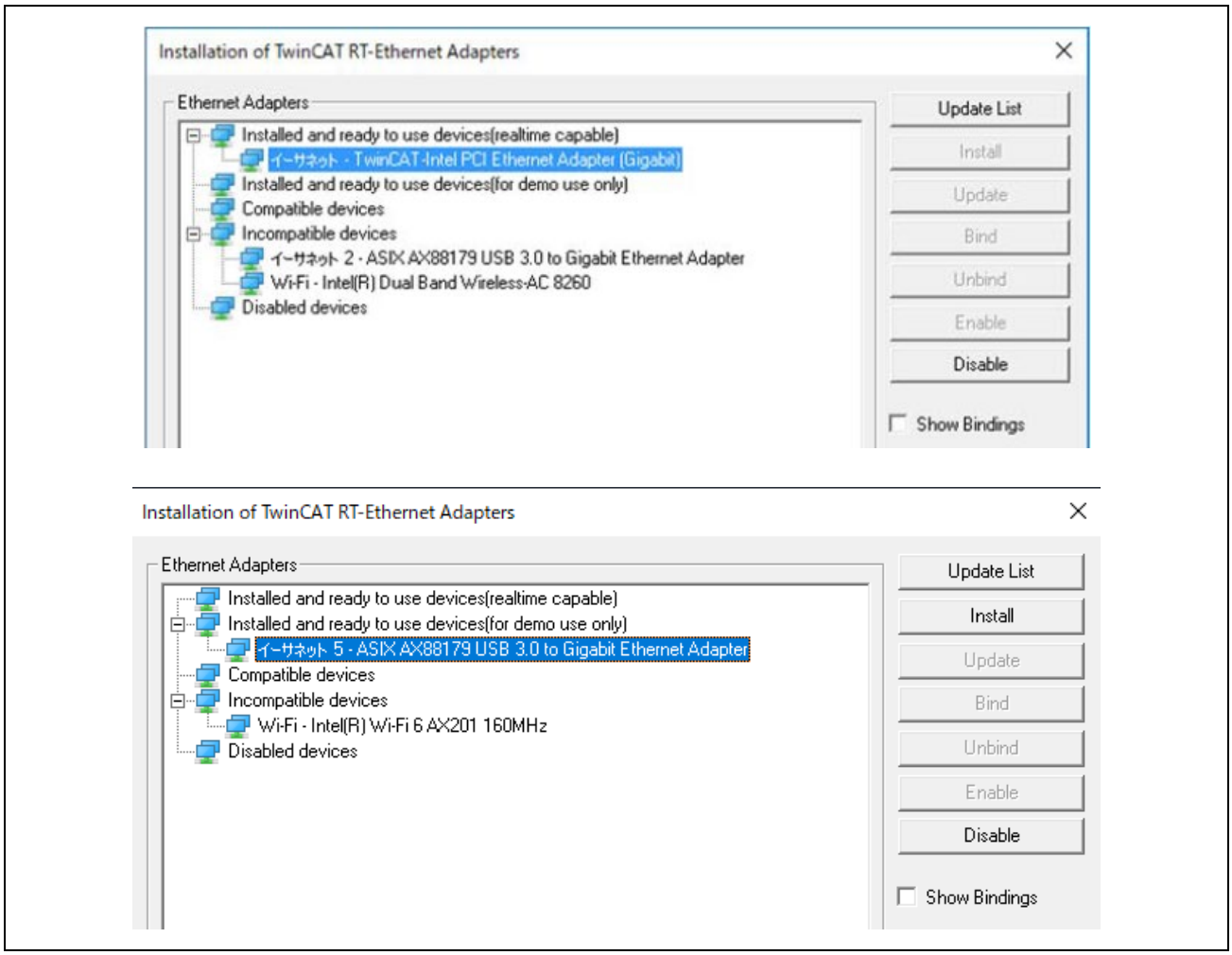
*既に本節の処理を行ったことがある場合は、本節の処理は不要です。

(1) 上のメニューバーから[TwinCAT] → [Show Realtime Ethernet Compatible Devices…]を選択してください。



(2) PC に接続されている Ethernet アダプタを選択した後に、[Install]を押してインストールしてください。

[Installed and ready to use devices(realtime capable)] または [Installed and ready to use devices(for demo use only)] にインストールしたドライバが追加されていることを確認してください。



5.4 ネットワークのスキャン

(1) システムマネージャツリーで[I/O]→[Devices]を右クリックし、[Scan]を選択します。

(2) [HINT: Not all types of devices can be found automatically]ダイアログで[OK]をクリックします。

(3) [new I/O devices found]ダイアログでスキャンを行うイーサネットアダプタのチェックボックスを選択し[OK]をクリックします。

(4) [Scan for Boxes]ダイアログで[Yes]をクリックします。

(5) ”Active Free Run”ダイアログが表示されるので[Yes]をクリックします。システムマネージャツリーで“I/O”→”Devices” の下に ”Device 1 ”→”Box 1 ” のように Box が追加されていれば正常です。

5.5　SII EEPROM の書き込み

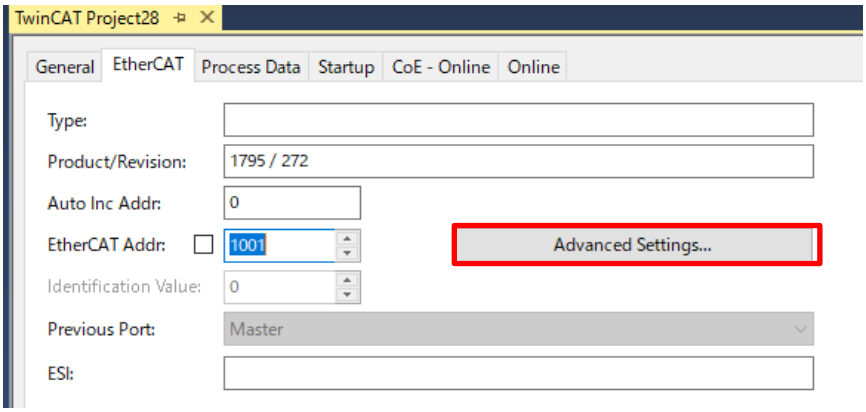
*評価ボードは出荷時に EEPROM がブランクになっていますので書き込みを必ず実施してください。

*EEPROM の書き込みを行っている場合は本節の処理は不要です。

EEPROM がブランクの場合、システムマネージャツリーには”Box1 (PFFFFFFFF RFFFFFFFF)”のように表示されます。

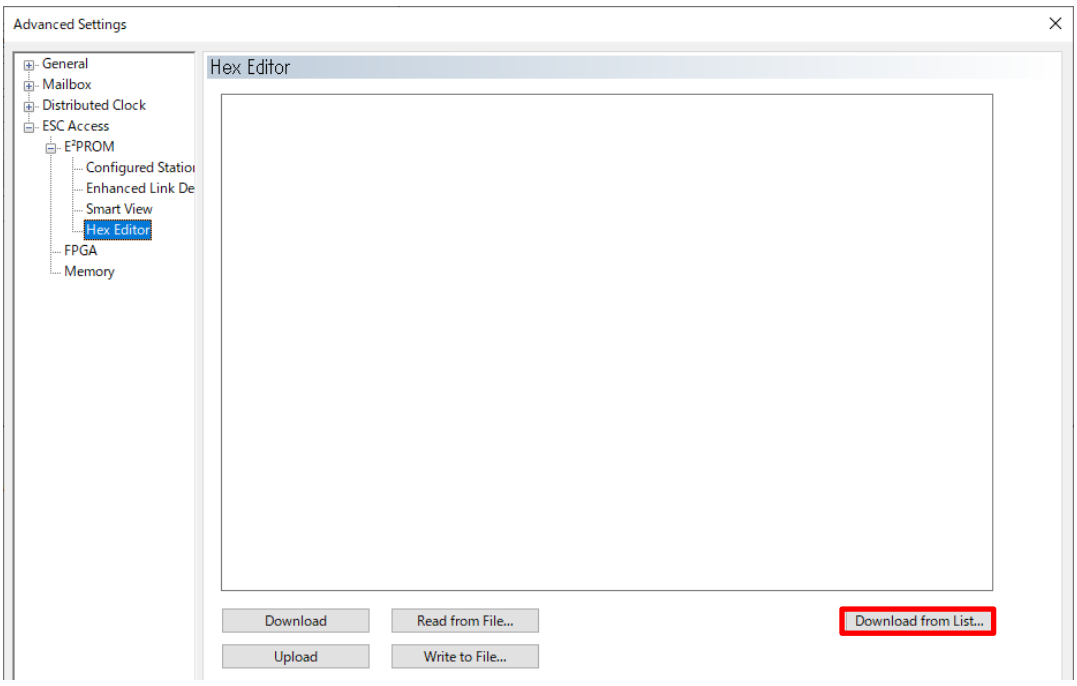
(1)　システムマネージャツリーで [Box 1]をダブルクリックすると、右側にパネルが表示されます。

(2)　[EtherCAT]タブを選択し [Advanced Settings]のボタンをクリックします。

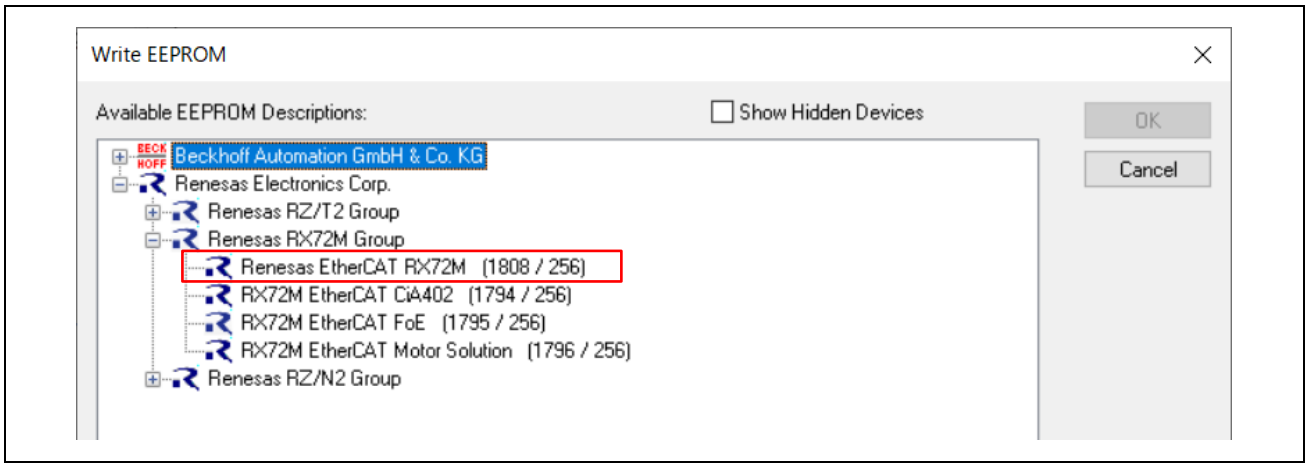


(3)　[Advanced Settings]ダイアログの左ツリーで[ESC Access] → [EEPROM] → [Hex Editor]を選択します。

(4)　[Hex Editor]ダイアログで[Download from list…]を選択します。



(5) [Write EEPROM]ダイアログで[Renesas Electronics Corp.] → [Renesas RX72M Group] → [Renesas EtherCAT RX72M] を選択し[OK]をクリックします。EEPROM に書き込まれます。



(6) 書き込み後は通信ボードを再起動し（電源再投入またはリセット）、書き換えたデータがマイコンの動作に反映されるようにしてください。

5.6 デバイスの再スキャン

(1) [I/O]の[devices]の下にある[Device x (EtherCAT)]を一旦削除してください。

(2) 再びシステムマネージャツリーで[I/O] → [Devices]を右クリックし、[Scan]を選択します。

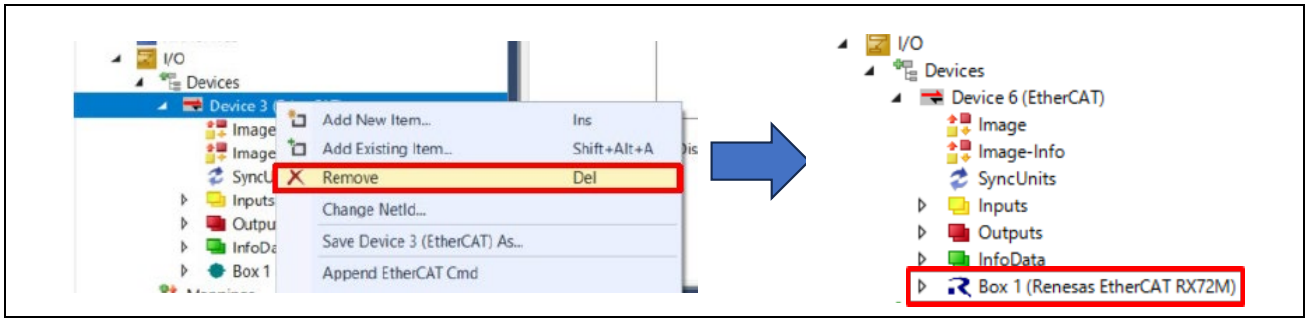
(3) [HINT: Not all types of devices can be found automatically]ダイアログで[OK]をクリックします。

(4) [new I/O devices found]ダイアログでスキャンを行うイーサネットアダプタのチェックボックスを選択し[OK]をクリックします。

(5) [Scan for Boxes]ダイアログで[Yes]をクリックします。

(6) [EtherCAT drive(s) added]ダイアログで、[NC - Configuration]を選択して[OK]を押します。

(7) [Active Free Run]ダイアログで[Yes]をクリックします。
システムマネージャツリーの”Box1 が”Box1(Renesas EtherCAT RX72M)”になっていれば正常です。

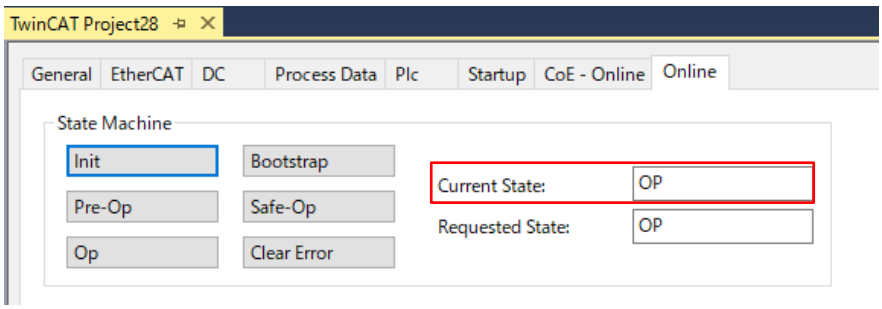


6. TwinCAT による動作確認

6.1 I/O 動作確認

(1) システムマネージャツリーで“Box 1”をダブルクリックすると、右側にパネルが表示されます。

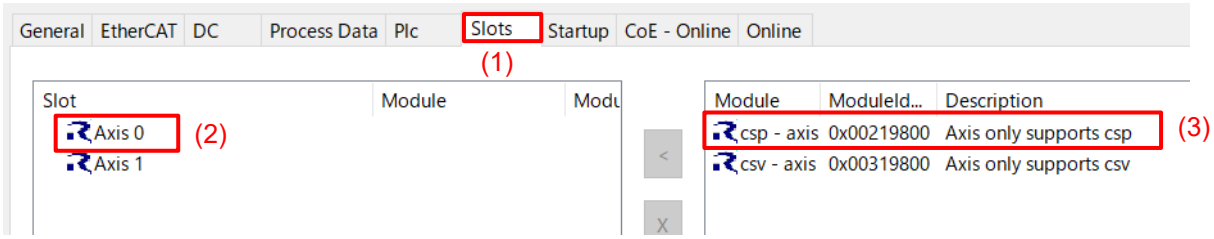
(2) “Online”タブを選択し、”Current Status”が”OP”になっていることを確認します。



(3) ”Current Status”が”ERR PREOP”となっている場合、“Slots”タブを選択して現在の Slot を確認してください。

左枠の“Slot”の”Axis 0”に Module ”csp-axis”が無ければ、右枠の”csp-axis”を選択して”Axis 0”に追加してください。

“Slot”に新しい Module を追加した場合は、上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。



(4) システムマネージャツリーで “Box 1” 左横の＋を展開します。

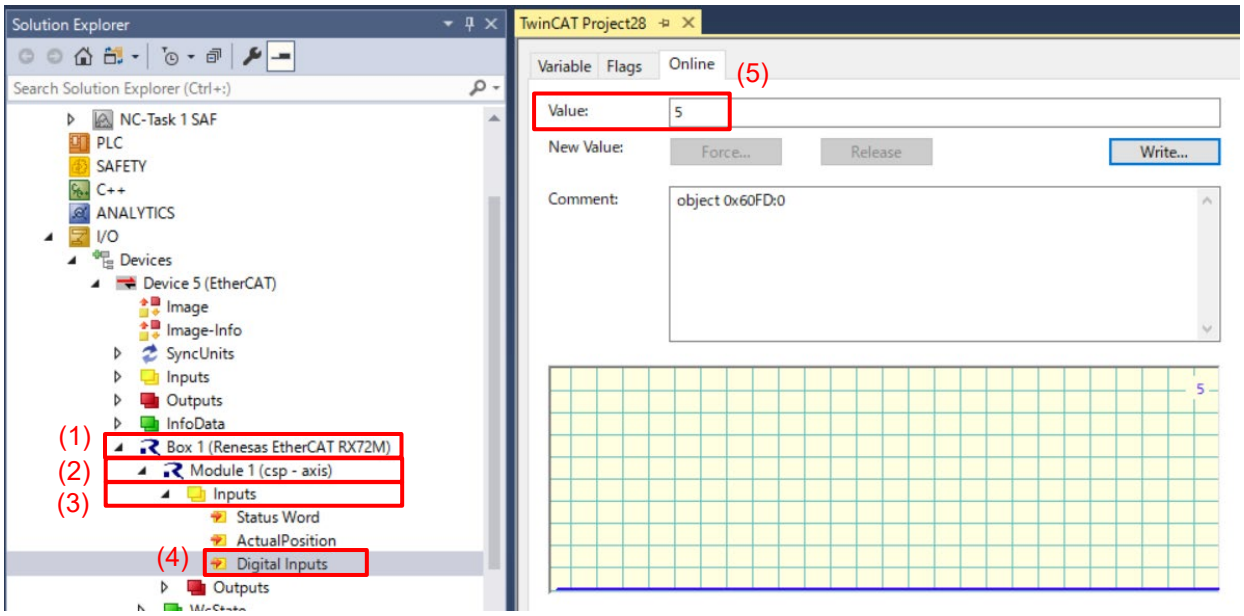
(5) “Module 1 (csp - axis)” → “Inputs” → “Digital Inputs” を選択し、右側パネルの” Online” タブを選択すると"Value"が表示されます。

”Digital Inputs”の値は評価ボード上の DIP SW またはジャンパーピン（JP）によって決定されます。

表 6-1 に本サンプルプログラムで”Digital Inputs”の値として使用される DIP SW または JP を示します。

表 6-1　”Digital Inputs”の値として使用される DIP SW またはジャンパーピン（JP）

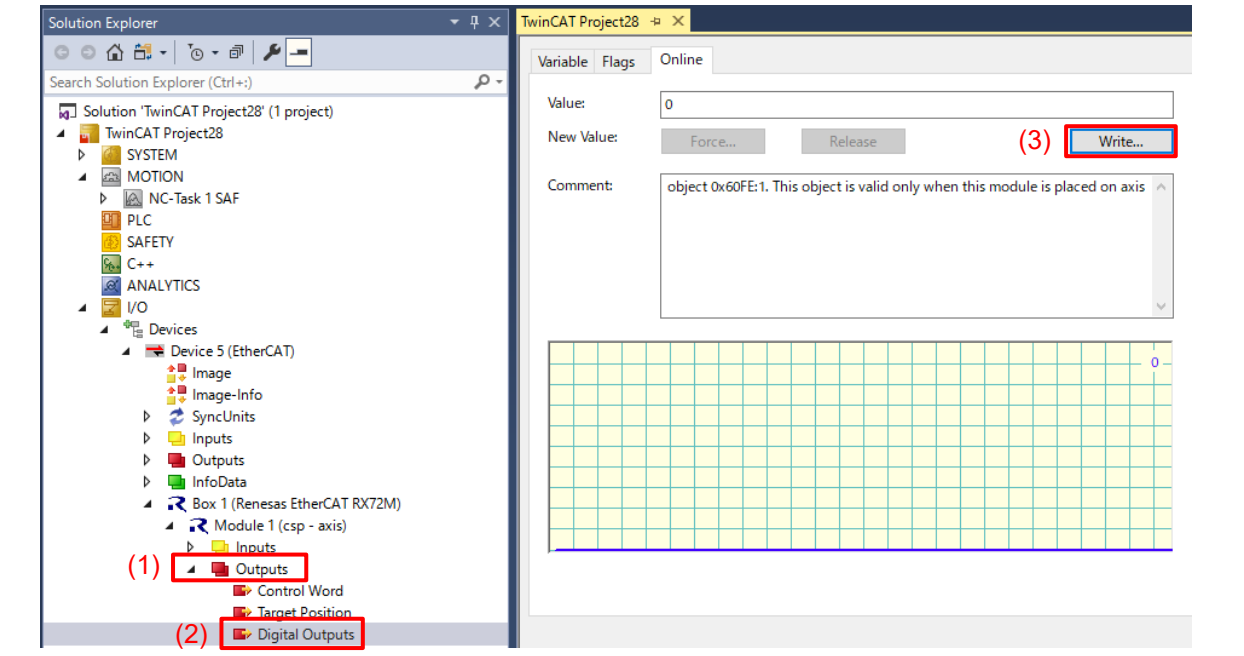
評価ボード名	使用する DIPSW または JP	数値の上下限
通信ボード	DIP SW 5	0 ~ 255
CPU カード	JP 5	0 ~ 7
RSK ボード	DIP SW 5	0 ~ 255



(6) システムマネージャツリーで“Outputs”→”Digital Outputs”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

"0"が表示されていることを確認してください。

(7) [Write]をクリックし”Set Value Dialog”ダイアログで任意の数値を入力してください。



(8) 「OK」をクリックし、"Value"の表示が入力した値になっていることを確認してください。

入力した値に応じて、評価ボード上の LED が点灯します。(bit=1 のとき LED 点灯)

表 6-2 に本プログラムで定義されている”Digital Outputs”と各評価ボードの LED との対応を示します。

表 6-2　”Digital Outputs”と各評価ボードの LED との対応

評価ボード名	”Digital Outputs”の値	点灯する LED
通信ボード	Bit 0 が 1	LED 1
	Bit 1 が 1	LED 2
	Bit 2 が 1	LED 3
	Bit 3 が 1	LED 4
CPU カード	Bit 0 が 1	LED 6
	Bit 1 が 1	LED 7
RSK ボード	Bit 0 が 1	LED 1
	Bit 1 が 1	LED 2
	Bit 2 が 1	LED 3
	Bit 3 が 1	LED 4

6.2　CiA402 Drive Profile 動作確認

6.2.1　CiA402 状態遷移

csp モードおよび csv モードの動作確認を行うためには、両モードともに、まず“Operation Enabled”の状態に遷移する必要があります。

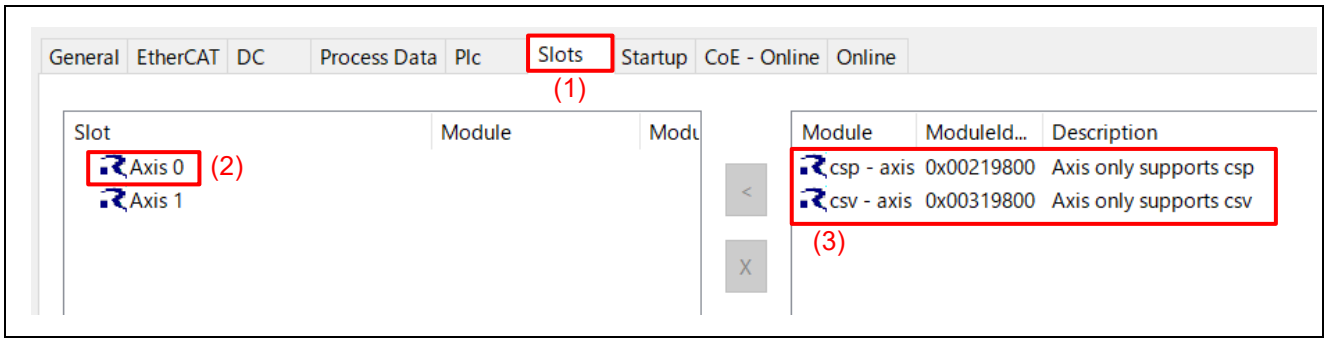
“Control Word”オブジェクトに値を設定することで状態遷移を発生させ、“Status Word”オブジェクトの値で状態を確認します。

(1)　システムマネージャツリーで“Box 1”をダブルクリックすると、右側にパネルが表示されます。

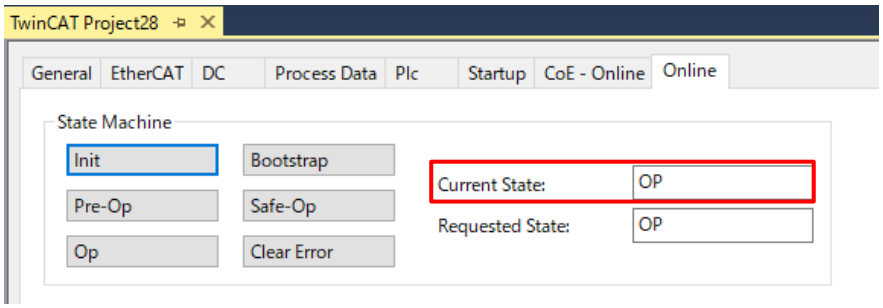
(2)　“Slots”タブを選択し、現在の Slot を確認してください。

左枠の“Slot”の”Axis 0”または”Axis 1”に“Operation Enabled”状態に遷移させたい Module を選択して追加してください。

“Slot”に Module を追加・変更した場合は、上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。



(3) “Online”タブを選択し、”Current Status”が”OP”になっていることを確認します。



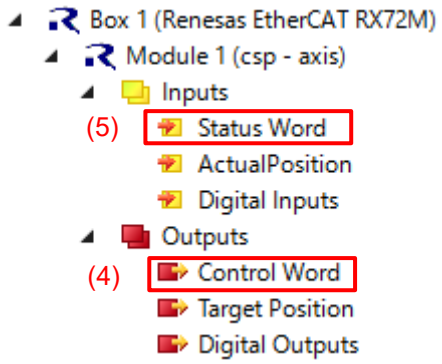
(4) システムマネージャツリーで“Outputs” → ”Control Word”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

[Write]をクリックし、値を[7]→[15]と順に設定します。

(5) システムマネージャツリーで“Inputs” → ”Status Word”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

[4663]になっていれば、”Operation Enabled”に遷移していますので次の手順に進んで下さい。

[4616]になっていれば、何等かの原因で”Fault”に遷移しています。”Control Word”をいったん、[128]に設定してから、(1)に戻ってください。



6.2.2 csp モード

(1) システムマネージャツリーで“Box 1”をダブルクリックすると、右側にパネルが表示されます。

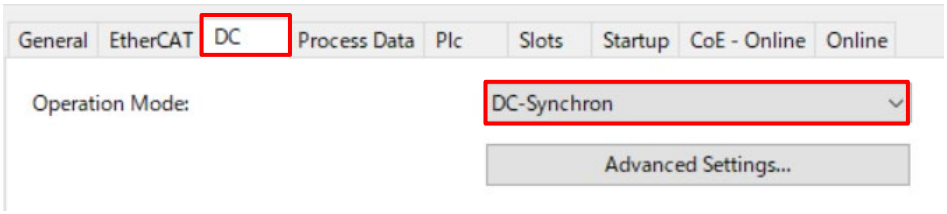
(2) “Slots”タブを選択し、左枠の“Slot”の”Axis 0”に”csp-axis”があることを確認してください。

そうでなければ、左枠の“Slot”の”Axis 0”に”csp-axis”を選択して追加・変更してください。

“Slot”に Module を追加・変更した場合は、上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。

(3) “Online”タブを選択し、”Current Status”が”OP”になっていることを確認します。

(4) “DC”タブを選択し、Operation Mode が”DC-Synchron”であることを確認してください。



(5) 6.2.1 の手順に従い、”Operation Enabled”に遷移してください。

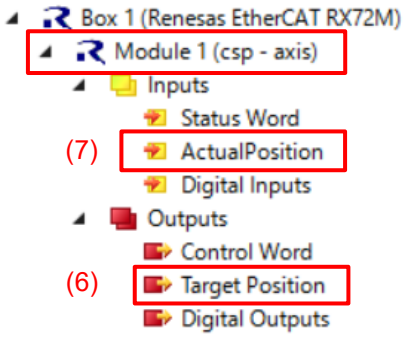
(6) システムマネージャツリーで“Outputs” → ”Target Position”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

[Write]をクリックし、値を任意の値を設定してください。

例として、ここでは[100000]を設定します。

(7) システムマネージャツリーで“Inputs→”Actual Position”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

”Target Position”に設定した[100000]までインクリメントされることを確認してください。



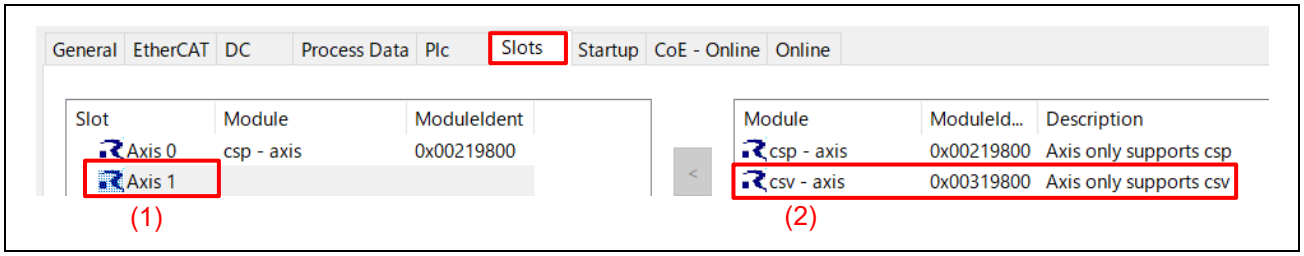
6.2.3 csv モード

(1) システムマネージャツリーで“Box 1”をダブルクリックすると、右側にパネルが表示されます。

(2) “Slots”タブを選択し、左枠の“Slot”の”Axis 1”に”csv-axis”があることを確認してください。

そうでなければ、左枠の“Slot”の”Axis 1”に”csv-axis”を選択して追加・変更してください。

“Slot”に Module を追加・変更した場合は、上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。



(3) “Online”タブを選択し、”Current Status”が”OP”になっていることを確認します。

(4) “DC”タブを選択し、Operation Mode が”DC-Synchron”であることを確認してください。

(5) 6.2.1 の手順に従い、“Operation Enabled”に遷移してください。

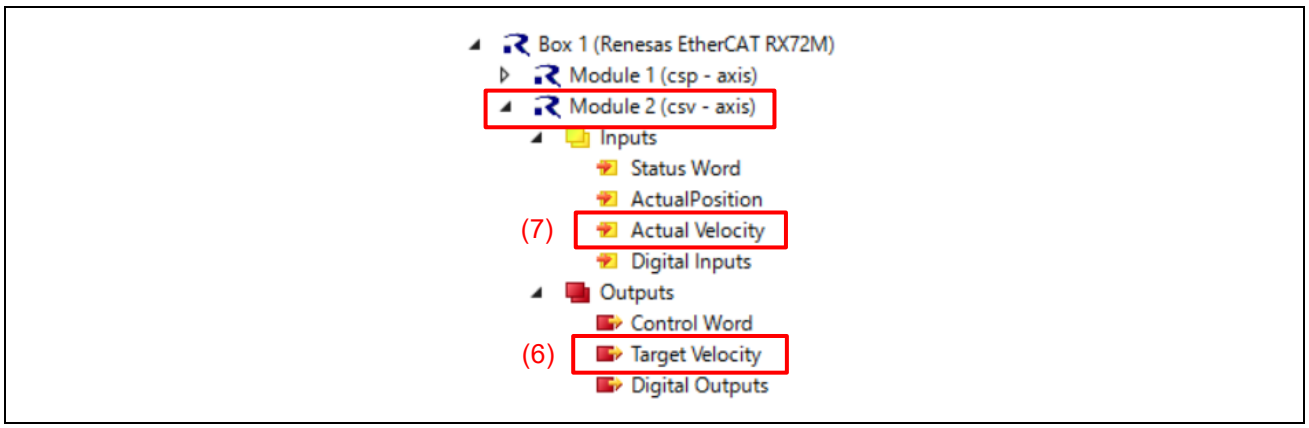
(6) システムマネージャツリーで“Outputs→”Target Velocity”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

[Write]をクリックし、値を任意の値を設定してください。

例として、ここでは[100000]を設定します。

(7) システムマネージャツリーで“Inputs→”Actual Velocity”を選択し、右側パネルの”Online”タブを選択すると"Value"が表示されます。

”Target Velocity”に設定した[100000]までインクリメントされることを確認してください。



6.3　ファームウェアアップデート動作確認

6.3.1　ファームウェア書き込み

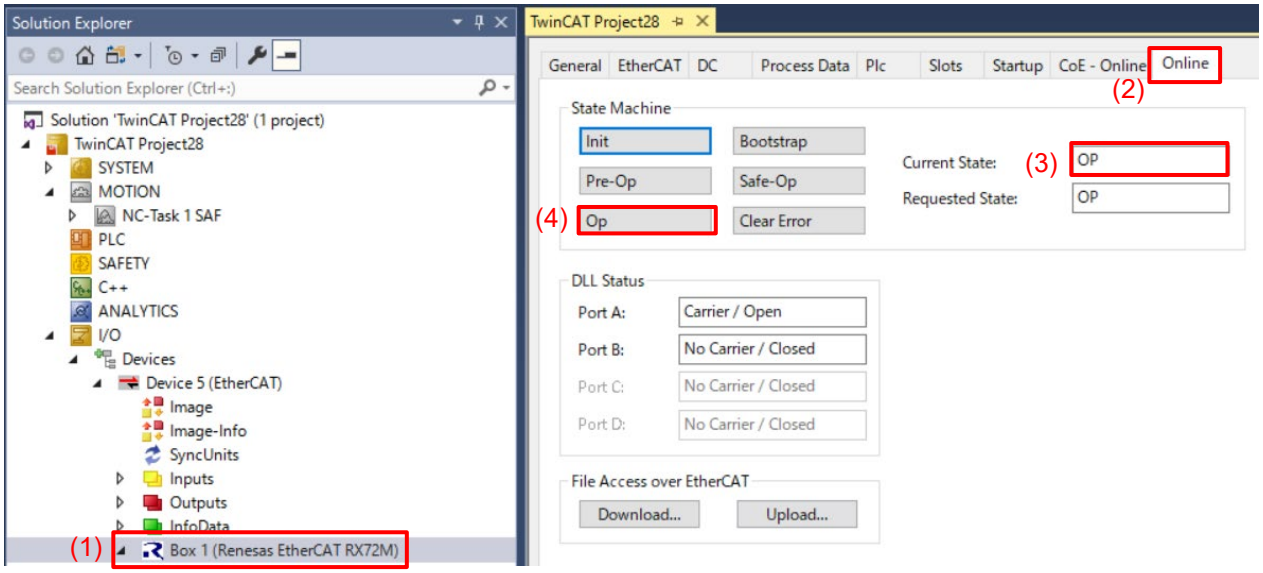
BANK0 でプログラムを実行中に BANK1 に新しいリビジョンのファームウェアを書き込む手順を説明します。

(1)　“Solution Explorer”で”Box 1（Renesas EtherCAT RX72M）”を選択後、

(2)　“Online”タブをクリックして

(3)　Current State が”OP”であることを確認してください。

(4)　“OP”でない場合は、”Op”ボタンを押して”OP”に遷移させてください。



※”Current Status”が”ERR PREOP”である場合、“Slots”タブを選択して現在の Slot を確認してください。

左枠の“Slot”の”Axis 0”に Module ”csp-axis”が無ければ、右枠の”csp-axis”を選択して”Axis 0”に追加してください。

“Slot”に新しい Module を追加した場合は、上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。

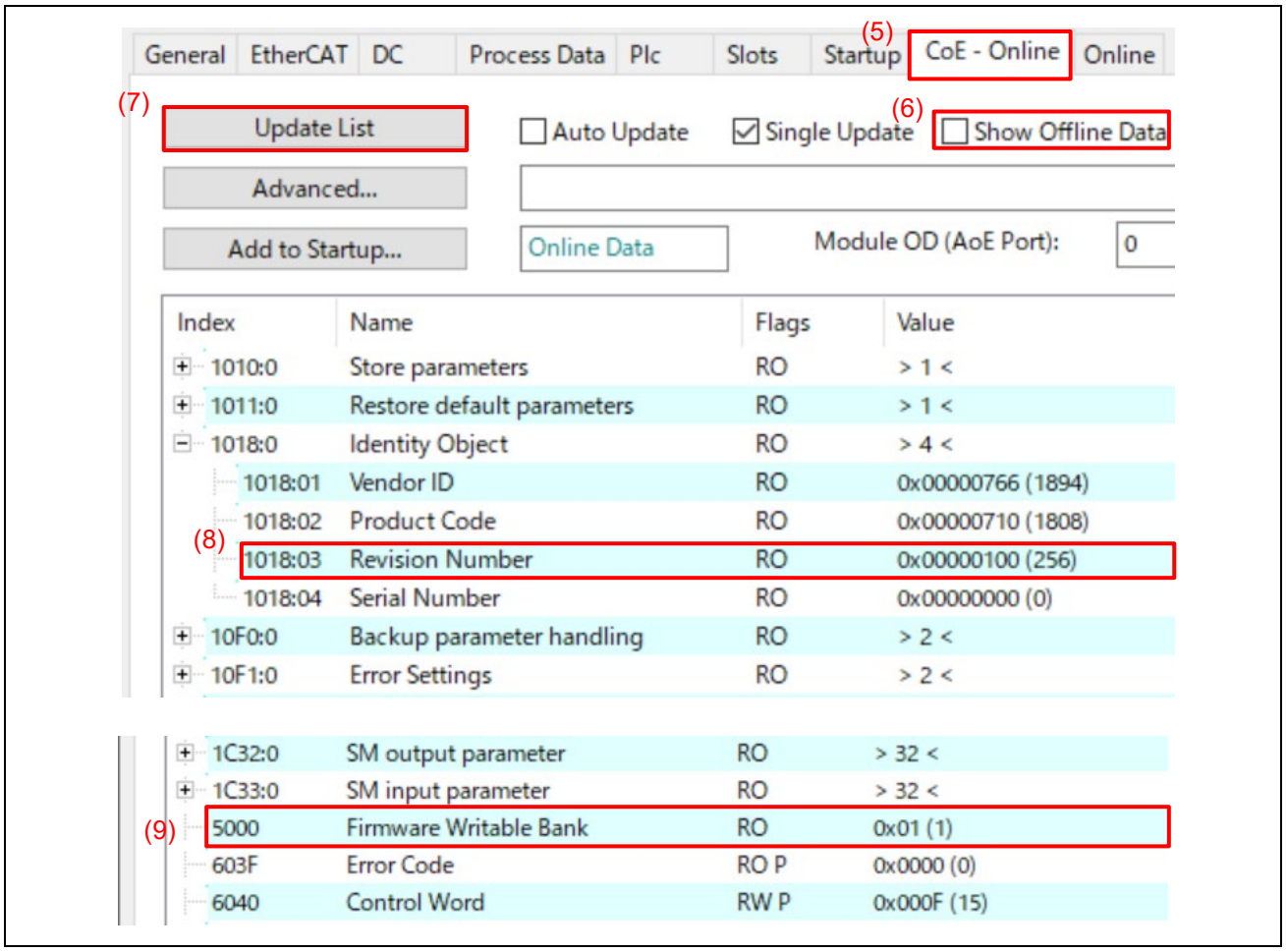
(5) CoE – Online”タブをクリックし、

(6) “Show Offline Data”のチェックが外れていることを確認してください。チェックしてある場合は、チェックを外してください。

(7) ”Update List”ボタンを押し、CoE リストを更新します。

(8) Index 1018:03 Revision の値を確認してください。例では Rev 1.00 を意味する 0x00000100(256)が表示されています。

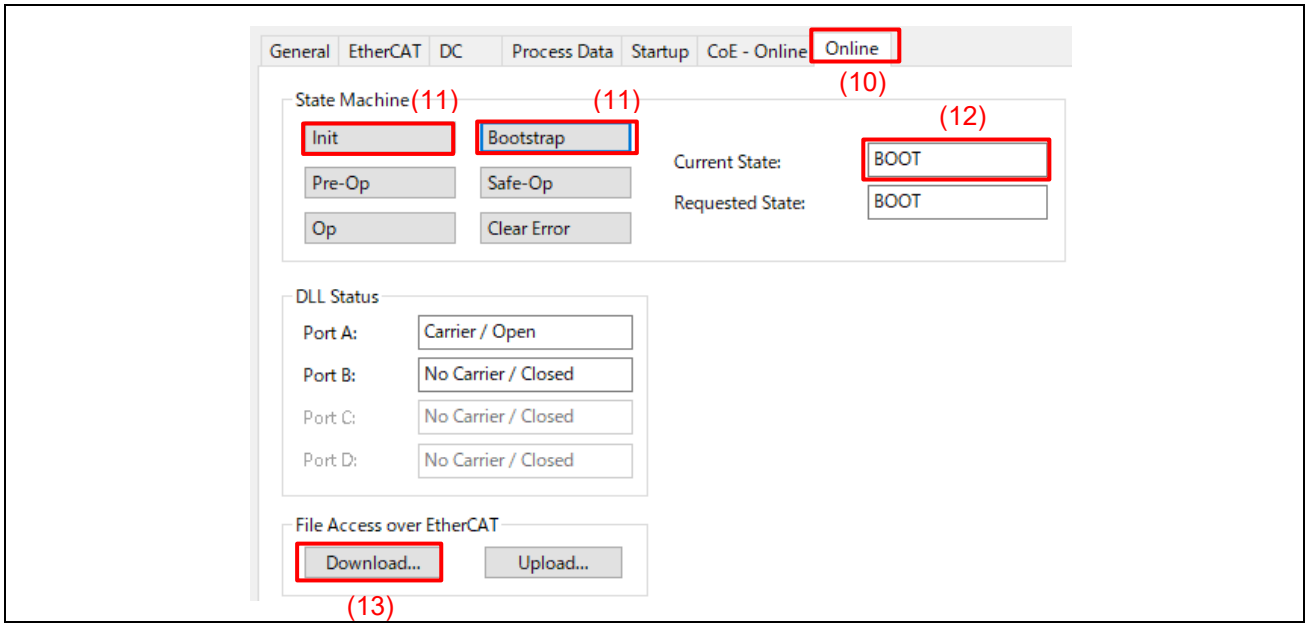
(9) Index 5000 Firmware Writable Bank の値を確認してください。例では書き込み可能バンクは BANK1 なので 0x01(1)が表示されています。



(10) “Online”タブをクリックして

(11) “Init”ボタン-> “Bootstrap”ボタンを順に押し、

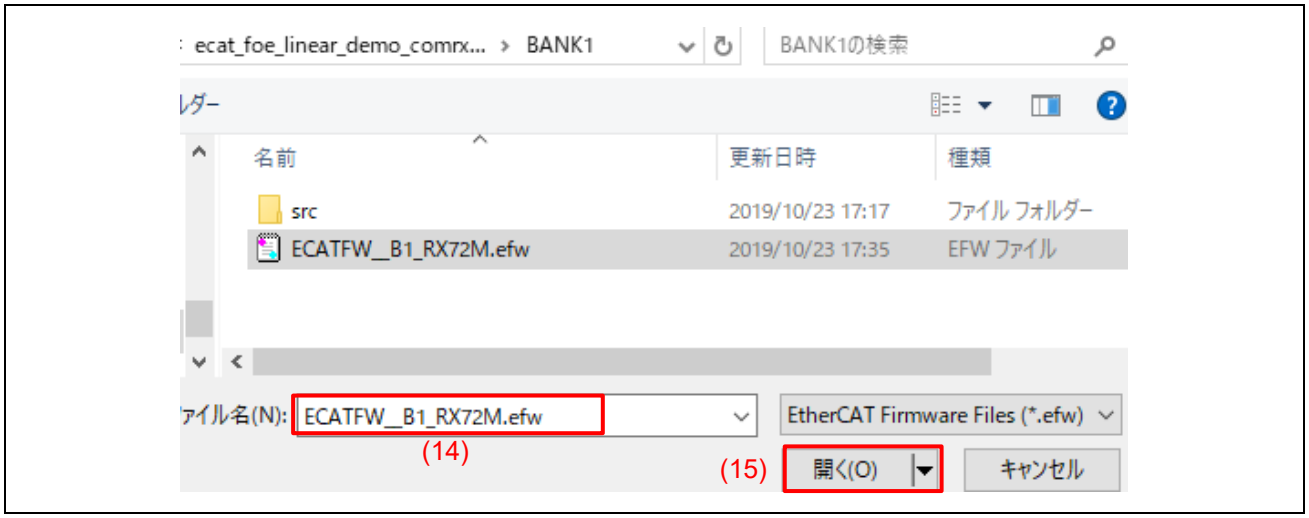
(12) Current State が”BOOT”に遷移することを確認してください。



(13) File Access over EtherCAT の”Download”ボタンを押すと、ダウンロードファイルの選択ウインドウが開きます。

(14) BANK1 用のダウンロードファイルである”ECAT__B1_RX72M.efw”を選び

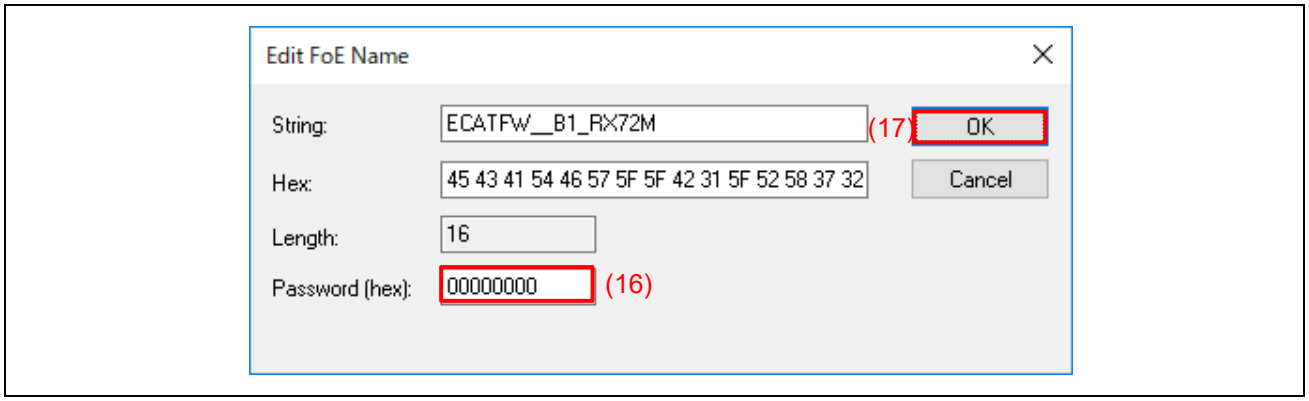
(15)「開く」を押してください。



ファイル名編集ウインドウが開きます。

(16) パスワードは”00000000”のまま

(17) “OK”を押します。



TwinCAT System Manager の画面最下部左側に”Downloading”のメッセージと共にダウンロード状況が表示されます。

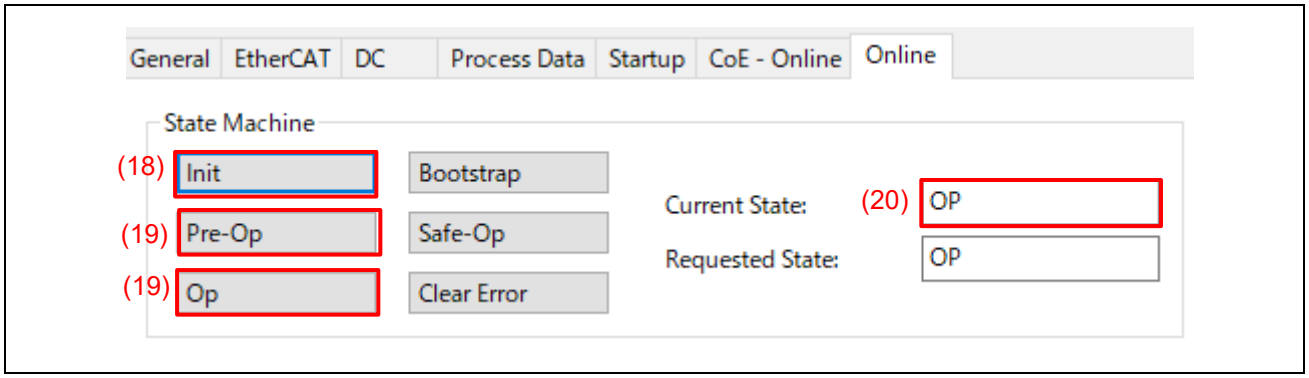
エラーメッセージが表示されず、上のウインドウが消えて”Ready”になれば、ファームウェア更新が成功しています。

“Online”タブで

(18) “Init”ボタンを押すと更新されたファームウェアで再起動します。

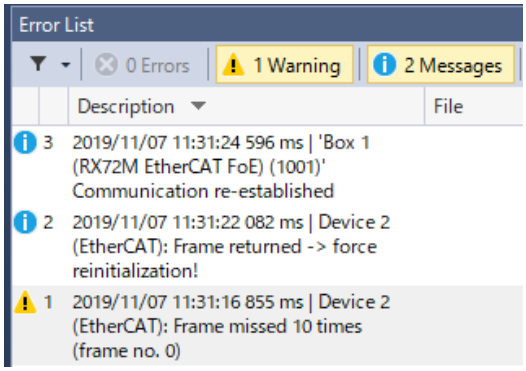
(19) “Preop”ボタン -> “Op”ボタンと押します。

(20) Current State が”OP”に遷移し、更新されたファームウェアの動作を確認することができます。



リニアモードの場合、再起動の際に EtherCAT のリンク切れが発生しないことを確認できます。

デュアルモード場合、再起動の際にソフトウェアリセットを行うため、一旦リンク切れが発生し、下図のような Warning メッセージが表示されますが問題ありません。



再び”CoE – Online”タブをクリックし、”Update List”ボタンを押し、CoE リストを更新します。

(21) Index 1018:03 Revision の値を確認してください。BANK1 のリビジョンの Rev 1.10 である 0x00000110(272)に変わったのが確認できます。

(22) Index 5000 Firmware Writable Bank の値を確認してください。リニアモードの場合、書き込み可能バンクは BANK1 から BANK0 に変わったので 0x00(0)が表示されています。

デュアルモードの場合は 0x01(1)のままになります。

6.3.2　ファームウェア読み出し

プログラムを実行している BANK 領域に格納されているファームウェアのバイナリデータを読み出すことができます。

バイナリデータはアップロードファイルとして EtherCAT マスターに保存されます。

アップロードファイルの形式を表 6-3 に示します。

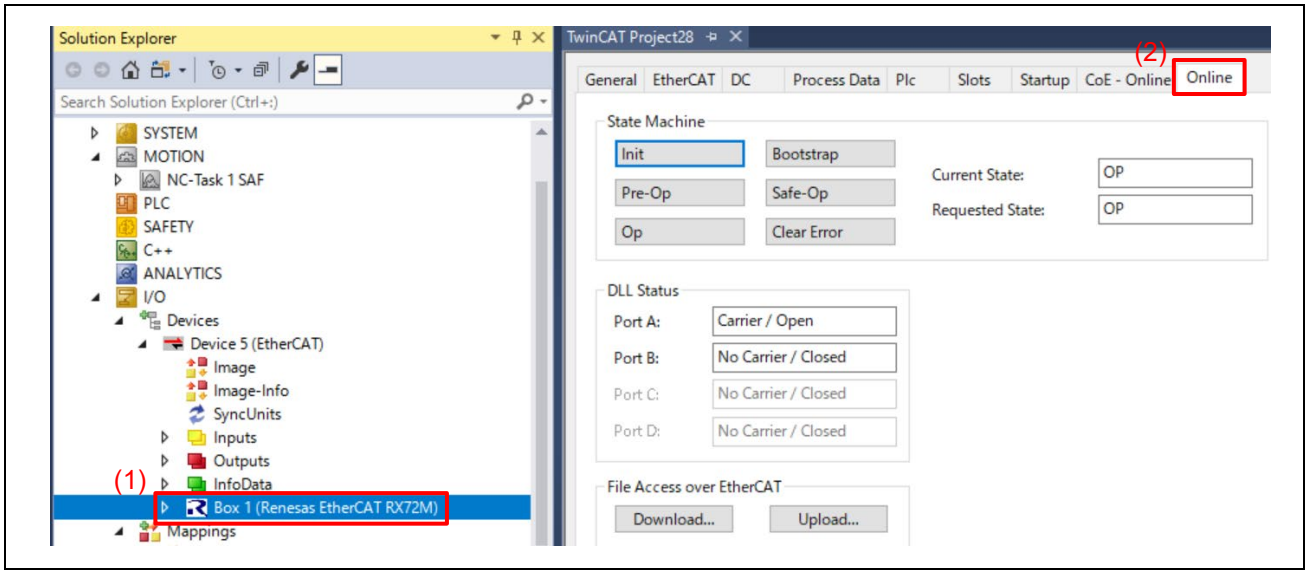
表 6-3　アップロードファイルの形式

項目	内容	
ファイル名接頭辞	”ECATFW__”	
ファイル拡張子	bin	
ファイル形式	バイナリ形式 ※ダウンロードファイルのモトローラ S 形式とは異なりますのでご注意ください。	
ファイルサイズ	リニアモード	4MB 製品：2016KB 2MB 製品：992KB
	デュアルモード	4MB 製品：2048KB 2MB 製品：1024KB
読み出し対象 BANK	リニアモード	Firmware Writable Bank=0 のとき BANK1 Firmware Writable Bank=1 のとき BANK0
	デュアルモード	常に BANK0

ファームウェアのバイナリデータを読み出す手順について説明します。

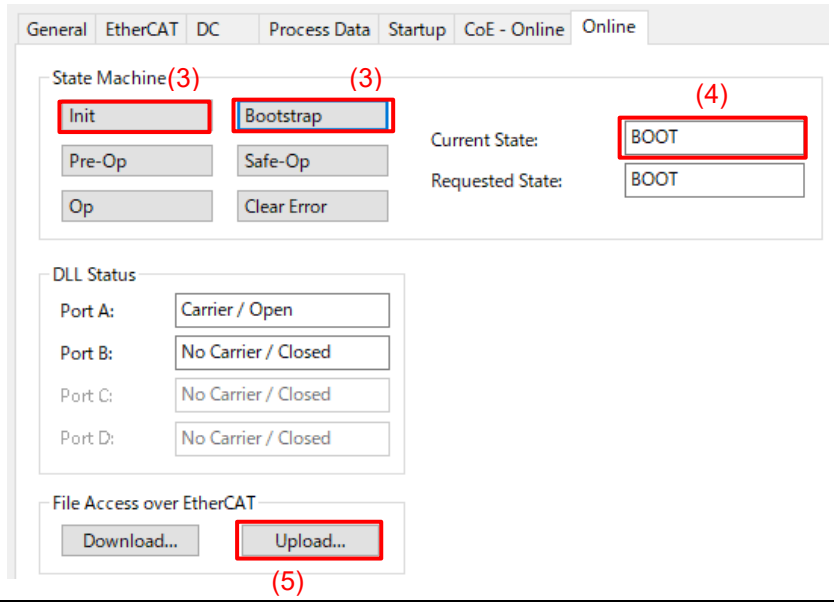
(1)　“Solution Explorer”で”Box 1（Renesas EtherCAT RX72M）”を選択後、

(2)　“Online”タブをクリックします。



(3) “Init”ボタン-> “Bootstrap”ボタンを順に押し、

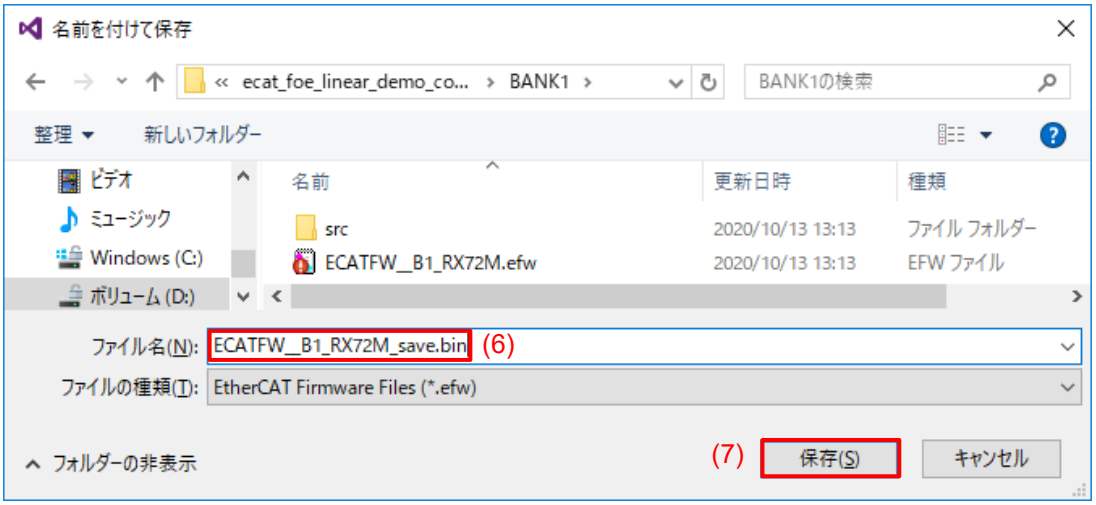
(4) Current State が”BOOT”に遷移することを確認してください。



(5) File Access over EtherCAT の”Upload”ボタンを押すと、アップロードファイルの保存ウインドウが開きます。

(6) アップロードファイル名として”ECATFW__B1_RX72M_save.bin”を入力し

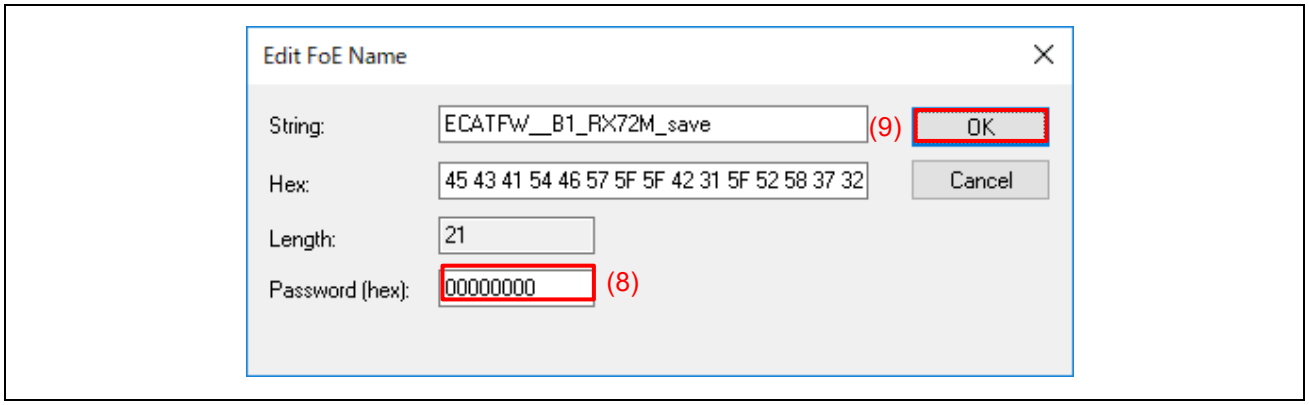
(7) 「保存」を押してください。



ファイル名編集ウインドウが開きます。

(8) パスワードは”00000000”のまま

(9) “OK”を押します。



TwinCAT System Manager の画面最下部左側に”Uploading”のメッセージと共にアップロード状況が表示されます。エラーメッセージが表示されず、上のウインドウが消えて”Ready”になれば、アップロードの成功です。

7. プロジェクトのコンフィグレーション

本章では、本サンプルプログラムを構成するためのコンフィギュレーションを説明します。

新規にプロジェクトを構築する場合に参照してください。

7.1 FIT モジュールのコンフィギュレーション

本プロジェクトでは、本サンプルプログラムを構成するために FIT モジュールのコンフィグレーション設定をデフォルトから変更しています。

スマートコンフィグレーションファイルを開き、コンポーネントタブ上で FIT モジュールのコンフィグレーションを確認・変更することができます。

FIT モジュールのコンフィグレーションの変更箇所一覧を表 7-1 に示します。

表 7-1　FIT モジュールコンフィグレーション変更一覧

FIT モジュール名	変更理由	Configuration	設定値
r_bsp	ヒープサイズの拡大	Heap size	0x8000
	User charget 関数を使用	Enable user stdio charget function	Use user charget() function
	User charput 関数を使用	Enable user stdio charput function	Use user charput() function
r_sci_rx	SCI CH6 有効化	Include software suport for channel 6	Include
	送信完了割込みの有効化	Transmit end interrupt	Enable

EtherCAT FIT モジュールの設定については、評価ボード毎に異なります。変更箇所を表 7-2 に示します。

表 7-2　EtherCAT FIT モジュールコンフィグレーション

評価ボード	変更理由	Configuration	設定値
COM ボード CPU カード	PHY リセットウェイト時間の設定	The waitng time for reset completion of PHY-LSI (us)	500
	使用する PHY LSI の設定	Use supported PHY-LSI	The KSZ8081MNX is used.
RSK ボード	PHY リセットウェイト時間の設定	The waitng time for reset completion of PHY-LSI (us)	1000
	使用する PHY LSI の設定	Use supported PHY-LSI	The KSZ8041NL is used.

Flash FIT モジュールについては、バンクモード毎に異なります。変更箇所を表 7-3 に示します。

表 7-3　Flash FIT モジュールコンフィグレーション

バンクモード	変更理由	Configuration	設定値
リニアモード	コードフラッシュの書き換えを許可する。	Enable code flash programming	Include code to program ROM area
	RAM 領域上でコードフラッシュを書き換える。	Enable code flash self-programming	Programming code flash while executing on RAM. (デフォルト)
デュアルモード	コードフラッシュの書き換えを許可する。	Enable code flash programming	Include code to program ROM area
	別バンクでコードフラッシュを書き換える。	Enable code flash self-programming	Programming code flash while executing from another segment in ROM.

バンクモードの設定について、バンクモード毎に設定する必要があります。変更内容を表 7-4 に示します。

表 7-4　バンクモードの設定

マクロ名	バンクモード	設定値
BSP_CFG_CODE_FLASH_BANK	リニアモード	1(デフォルト値)
	デュアルモード	0

バンクモードの設定については、コンポーネントタブ上で確認・変更することができません。

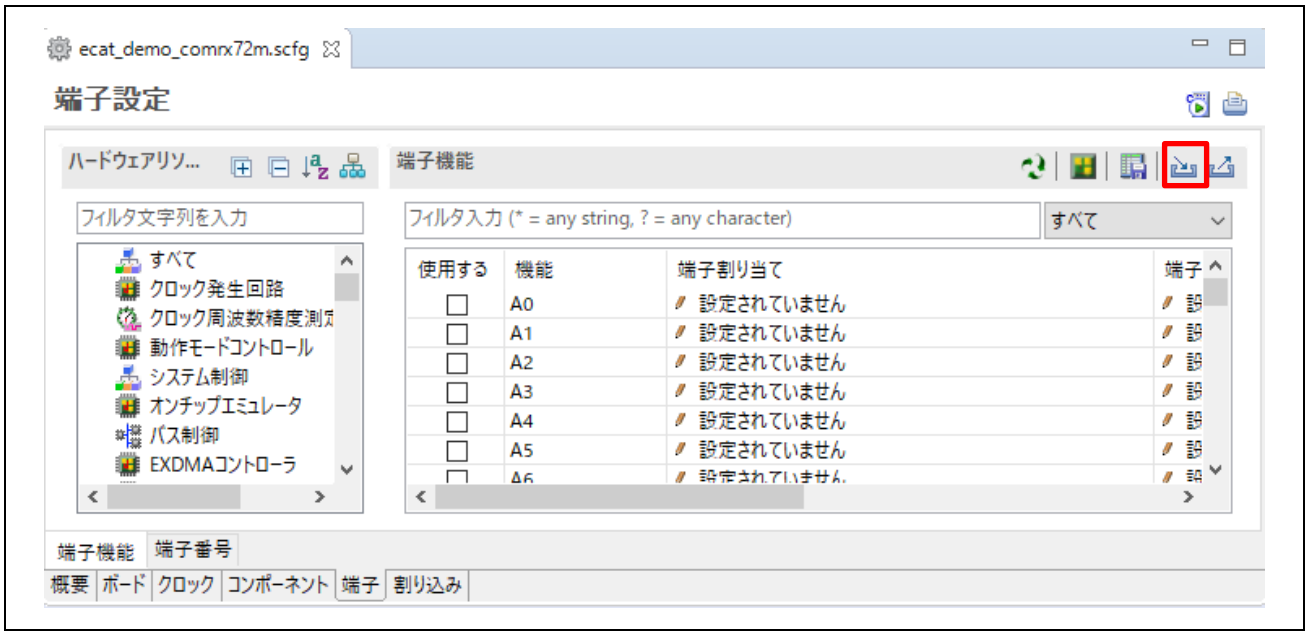
コード生成後に生成されるファイル ”src¥smc_gen¥r_config¥r_bsp_config.h” 内で確認・変更することができます。

7.2　端子設定

サンプルプログラムを動かすための端子設定をインポートすることができます。

スマートコンフィグレータの[端子]タブで[端子機能の配置をインポート]ボタンをクリックして、“ecat_linear_demo_cpurx72m_board.xml” を選択してください。

xml ファイルはサンプルプログラムに含まれています。



FIT モジュールで使用する端子をスマートコンフィグレータの[コンポーネント]タブの[リソース]で、表 7-5 に示す項目を有効にする必要があります。

表 7-5 有効化する端子一覧

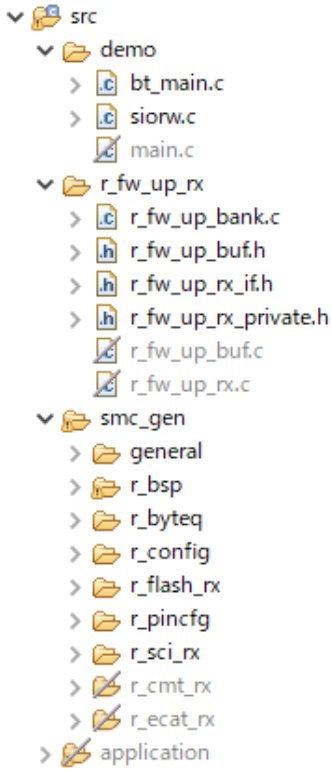
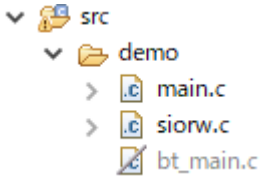
コンポーネント	有効にするリソース	有効にする端子
r_ecat_rx	ESC ESC_MII0 ESC_MII1	全端子
r_sci_rx	SCI6	RXD6/SMISO6/SSCL6 端子 TXD6/SMOSI6/SSDA6 端子

7.3　ビルド構成

7.3.1　リニアモードのビルド構成

リニアモードで使用するプロジェクトの各ビルド構成の変更箇所について説明します。

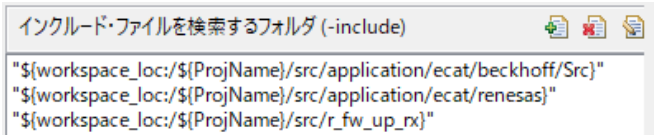
表 7-6　リニアモードのビルド構成

構成名	説明
BootLoader	ブートローダープログラムをビルドします。 ブートローダーに必要な機能以外のファイルはビルドから除外しています。 該当のファイルまたはフォルダを右クリックし「リソース構成」→「ビルドから除外」でビルドの除外が可能です。 ＜設定＞ 
BANK0	BANK0 にダウンロードする EtherCAT スレーブプログラムをビルドします。 ビルド成果物はモトローラ S 形式のファイルです。 demo/bt_main.c のみビルドから除外します。 ＜設定＞ 
BANK1	BANK1 にダウンロードする EtherCAT スレーブプログラムをビルドします。 BANK0 と BANK1 はコードフラッシュのマッピングとビルド成果物であるダウンロードファイル名が異なる以外、設定は同じです。

7.3.1.1 BootLoader

設定内容の確認は、プロジェクト→プロパティ→C/C++ビルド→設定にて行うことができます。

表 7-7　BootLoader – ツール設定タブ

項目	変更内容	説明
Compiler - ソース	「インクルード・ファイル・ディレクトリ」にインクルードパスを追加	各 FIT モジュールで設定が必要なインクルードパスを追加しています。 コード生成されたファルダについては、「FIT Configurator」を使用して各 FIT モジュールを組み込む場合に限り自動で設定されます。
		コード生成しないプログラムファイルについては、インクルードパスを手動で追加する必要があります。 本プロジェクトでは下記の 3 項目を追加しています。 
Compiler – 最適化	最適化レベルを変更 レベル 0：最適化を実施しない	ブートローダーが参照する const テーブルが最適化によりビルドされないことを防ぐため、最適化レベルを 0 に設定しています。
Linker – セクション	ROM 領域に配置される PResetPRG の開始位置を 0xFFFF8000 に変更 ＜設定＞ 0xFFFF8000 PResetPRG C_1 C_2 C C_8 C$* D* W* L P PFRAM 0xFFFFFF80 EXCEPTVECT 0xFFFFFFFC RESETVECT	BootLoader のマッピングは 0xFFF8000 から 0xFFFFFFFF としています。サイズは 32KB です。 最後尾にリセットベクタを含む 128 バイトの例外ベクタテーブルを配置します。

7.3.1.2　BANK0

設定内容の確認は、プロジェクト→プロパティ→C/C++ビルド→設定にて行うことができます。

表 7-8　BANK0 - ツール設定タブ

項目	変更内容	説明
Compiler - ソース	「インクルード・ファイル・ディレクトリ」にインクルードパスを追加	BootLoader と同様の設定です。
Compiler - ソース	プロセッサマクロ定義を追加 ＜設定＞ プリプロセッサ・マクロの定義 FLASH_APPL_BANK=0 _DISABLE_REVNO_CHECK REVISION_NUMBER=0x00000100	“FLASH_APPL_BANK=0”とすると BANK0 用のコードがビルドされます。 “_DISABLE_REVNO_CHECK”を定義するとファームウェアと EEPROM のリビジョン No が異なっていてもエラーになりません。 “REVISION_NUMBER”は 1.00 として定義しています。
Compiler – 最適化	最適化レベルを変更 レベル 1 : 一部最適化を実施する	SSC が生成するコードの一部が最適化によりビルドされないことを防ぐため、最適化レベルを 1 に設定しています。
Linker – セクション	RAM 領域に RPFRAM セクションを追加 ＜設定＞ 0x00000004 SU SI B_1 R_1 B_2 R_2 B R B_8 R_8 RPFRAM	コードフラッシュメモリを書き換えるコードを RAM 領域に追加します。
	ROM 領域に配置される PResetPRG の開始位置を 0xFFE00000 に変更 PFRAM セクションを追加 ＜設定＞ 0xFFE00000 PResetPRG C_1 C_2 C C_8 C$* D* W* L P PFRAM	BANK0 のマッピングは 0xFFE00000 から 0xFFFF7FFF としています。サイズは 2016KB です。 コードフラッシュメモリを書き換えるためのコードを ROM 領域に追加します。

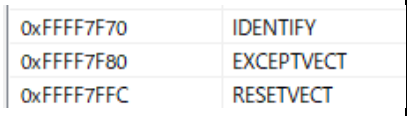
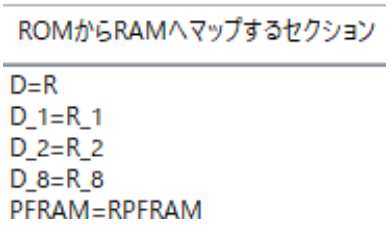
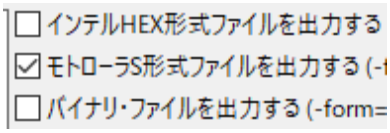
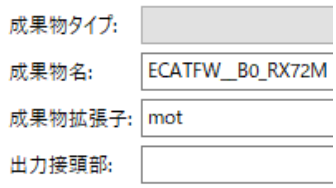
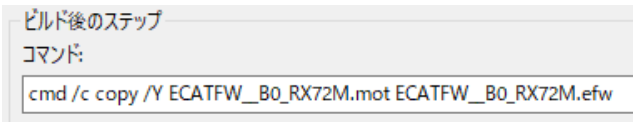
	ROM 領域に IDENTIFY セクションを追加し、開始位置を 0xFFFF7F70 とする ＜設定＞ 	IDENTIFY セクションにはファームウェアのリビジョン No を含む 16 バイトのテーブルデータが配置されます。 BANK0 の最後尾に IDENTIFY セクションとリセットベクタを含む例外ベクタテーブル(128 バイト)を配置します。
Linker – セクション – シンボルファイル	ROM から RAM へマップするセクションを追加 ＜設定＞ 	サンプルプログラムは、コードフラッシュメモリの書き換えを RAM 上で実行します。そのため ROM から RAM にマップする設定”PFRAM=RPFRAM”を追加しています。
Converter – 出力	モトローラ S 形式ファイルを出力する設定に変更 ＜設定＞ 	ダウンロードファイルはモトローラ S 形式ファイルとしています。

表 7-9　BANK0 - ツール設定タブ以外のタブ

タブ項目	変更内容	説明
ビルド成果物	出力ファイルを ECATFW__B0_RX72M.mot に変更 ＜設定＞ 	ダウンロードファイル名の接頭辞は”ECATFW__B0”としています。
ビルド・ステップ	ビルド終了後に出力ファイルのコピーを ECATFW__B0_RX72M.efw として作成 ＜設定＞ 	TwinCAT でダウンロード可能なファイルの拡張子は”efw”となっているためです。

7.3.1.3 BANK1

設定内容の確認は、プロジェクト→プロパティ→C/C++ビルド→設定にて行うことができます。

表 7-10　BANK1 - ツール設定タブ

項目	変更内容	説明
Compiler - ソース	「インクルード・ファイル・ディレクトリ」にインクルードパスを追加	BootLoader と同様の設定です。
Compiler - ソース	プロセッサマクロ定義を追加 ＜設定＞ プリプロセッサ・マクロの定義 FLASH_APPL_BANK=1 _DISABLE_REVNO_CHECK REVISION_NUMBER=0x00000110	“FLASH_APPL_BANK=1”とすると BANK1 用のコードがビルドされます。 “_DISABLE_REVNO_CHECK”を定義するとファームウェアと EEPROM のリビジョン No が異なっていてもエラーになりません。 “REVISION_NUMBER”は 1.10 として定義しています。
Compiler – 最適化	最適化レベルを変更 レベル 1：一部最適化を実施する	SSC が生成するコードの一部が最適化によりビルドされないことを防ぐため、最適化レベルを 1 に設定しています。
Linker – セクション	RAM 領域に RPFRAM セクションを追加 ＜設定＞ 0x00000004 SU SI B_1 R_1 B_2 R_2 B R B_8 R_8 RPFRAM	コードフラッシュメモリを書き換えるコードを RAM 領域に追加します。
	ROM 領域に配置される PResetPRG の開始位置を 0xFFC00000 に変更 PFRAM セクションを追加 ＜設定＞ 0xFFC00000 PResetPRG C_1 C_2 C C_8 C$* D* W* L P PFRAM	BANK1 のマッピングは 0xFFC00000 から 0xFFDF7FFF としています。サイズは 2016KB です。 コードフラッシュメモリを書き換えるためのコードを ROM 領域に追加します。

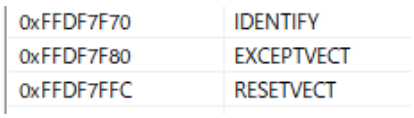
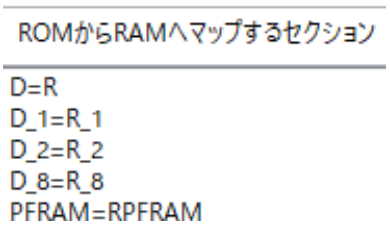
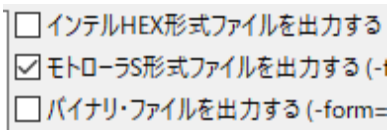
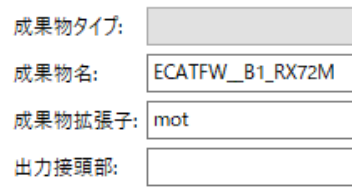
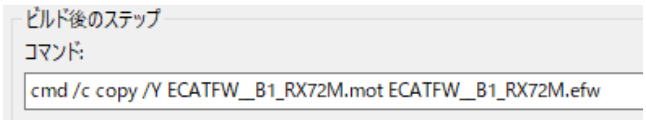
	ROM 領域に IDENTIFY セクションを追加し、開始位置を 0xFFDF7F70 とする ＜設定＞ 	IDENTIFY セクションにはファームウェアのリビジョン No を含む 16 バイトのテーブルデータが配置されます。 BANK0 の最後尾に IDENTIFY セクションとリセットベクタを含む例外ベクタテーブル(128 バイト)を配置します。
Linker – セクション – シンボルファイル	ROM から RAM へマップするセクションを追加 ＜設定＞ 	サンプルプログラムは、コードフラッシュメモリの書き換えを RAM 上で実行します。そのため ROM から RAM にマップする設定”PFRAM=RPFRAM”を追加しています。
Converter – 出力	モトローラ S 形式ファイルを出力する設定に変更 ＜設定＞ 	ダウンロードファイルはモトローラ S 形式ファイルとしています。

表 7-11　BANK1 - ツール設定タブ以外のタブ

タブ項目	変更内容	説明
ビルド成果物	出力ファイルを ECATFW__B1_RX72M.mot に変更 ＜設定＞ 	ダウンロードファイル名の接頭辞は”ECATFW__B1”としています。
ビルド・ステップ	ビルド終了後に出力ファイルのコピーを ECATFW__B1_RX72M.efw として作成 ＜設定＞ 	TwinCAT でダウンロード可能なファイルの拡張子は”efw”となっているためです。

7.3.2　デュアルモードのビルド構成

デュアルモードで使用するプロジェクトの各ビルド構成の変更箇所について説明します。

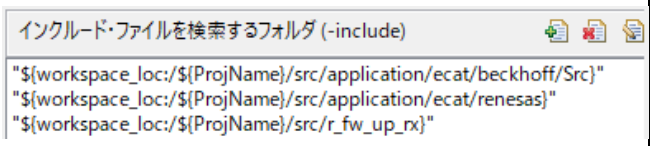
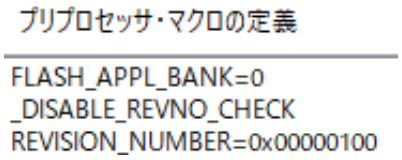
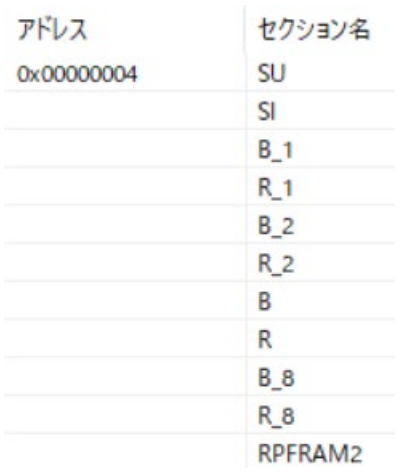
表 7-12　デュアルモードのビルド構成

構成名	説明
Hardware Debug	EtherCAT スレーブプログラムのデバッグ情報付きのロードモジュールを生成します。 ビルドから除外するファイルはありません。
Download	ダウンロードする EtherCAT スレーブプログラムをビルドします。 ビルド成果物はモトローラ S 形式ファイルです。 ビルドから除外するファイルはありません。

7.3.2.1 HardwareDebug

設定内容の確認は、プロジェクト→プロパティ→C/C++ビルド→設定にて行うことができます。

表 7-13　HardwareDebug - ツール設定タブ

項目	変更内容	説明
Compiler - ソース	「インクルード・ファイル・ディレクトリ」にインクルードパスを追加	各 FIT モジュールで設定が必要なインクルードパスを追加しています。コード生成されたファルダについては、「FIT Configurator」を使用して各 FIT モジュールを組み込む場合に限り自動で設定されます。
		コード生成しないプログラムファイルについては、インクルードパスを手動で追加する必要があります。 本プロジェクトでは下記の 3 項目を追加しています。 
Compiler - ソース	プロセッサマクロ定義を追加 ＜設定＞ 	“FLASH_APPL_BANK=0”とすると BANK0 用のコードがビルドされます。 “_DISABLE_REVNO_CHECK”を定義するとファームウェアと EEPROM のリビジョン No が異なっていてもエラーになりません。 “REVISION_NUMBER”は 1.00 として定義しています。
Compiler – 最適化	最適化レベルを変更 レベル 1：一部最適化を実施する	SSC が生成するコードの一部が最適化によりビルドされないことを防ぐため、最適化レベルを 1 に設定しています。
Linker – セクション	RAM 領域に RPFRAM2 セクションを追加 ＜設定＞ 	バンクを切り換えるコードを RAM 領域に追加します。

<table>
<tr><td></td><td>ROM 領域に配置される PResetPRG の開始位置を 0xFFE08000 に変更。

PFRAM2 セクションを追加。

＜設定＞

<table><tr><td>0xFFE08000</td><td>PResetPRG</td></tr><tr><td></td><td>C_1</td></tr><tr><td></td><td>C_2</td></tr><tr><td></td><td>C</td></tr><tr><td></td><td>C_8</td></tr><tr><td></td><td>C$*</td></tr><tr><td></td><td>D*</td></tr><tr><td></td><td>W*</td></tr><tr><td></td><td>L</td></tr><tr><td></td><td>P</td></tr><tr><td></td><td>PFRAM2</td></tr></table></td><td>BANK0 のマッピングは 0xFFE00000 から 0xFFFFFFFF ですが、PResetPRG の開始位置を 0xFFE08000 と設定しています。

BANK0 のサイズは 2048KB です。

コードフラッシュメモリを書き換えるためのコードを ROM 領域に追加しています。

本サンプルプログラムの 0xFFE00000～0xFFE08000 領域は、Store Parameters のために空けています。Store Parameters の詳細については 9.2.3 を参照してください。</td></tr>
<tr><td></td><td>ROM 領域に IDENTIFY セクションを追加し、開始位置を 0xFFFFFF70 とする。

＜設定＞

<table><tr><td>0xFFFFFF70</td><td>IDENTIFY</td></tr><tr><td>0xFFFFFF80</td><td>EXCEPTVECT</td></tr><tr><td>0xFFFFFFFC</td><td>RESETVECT</td></tr></table></td><td>IDENTIFY セクションにはファームウェアのリビジョン No を含む 16 バイトのテーブルデータが配置されます。

BANK0 の最後尾に IDENTIFY セクションとリセットベクタを含む例外ベクタテーブル(128 バイト)を配置します。</td></tr>
<tr><td>Linker

– セクション

– シンボルファイル</td><td>ROM から RAM へマップするセクションを追加する。

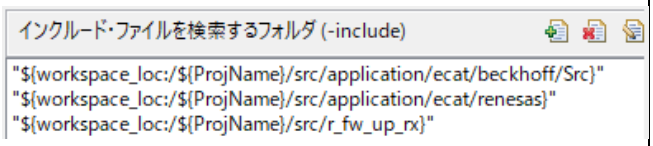
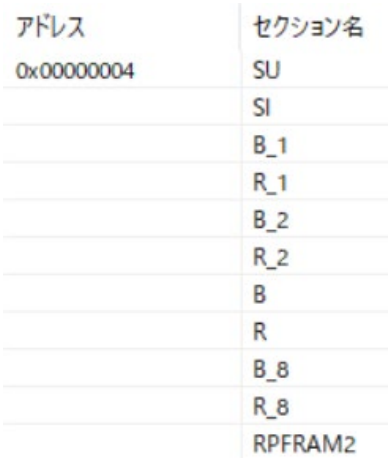
＜設定＞

<table><tr><td>ROMからRAMへマップするセクション</td></tr><tr><td>D=R
D_1=R_1
D_2=R_2
D_8=R_8
PFRAM2=RPFRAM2</td></tr></table></td><td>バンクの切り替えを行うコードは RAM 上で実行します。そのため ROM から RAM にマップする設定を追加します。</td></tr>
</table>

7.3.2.2 Download

設定内容の確認は、プロジェクト→プロパティ→C/C++ビルド→設定にて行うことができます。

表 7-14　Download - ツール設定タブ

項目	変更内容	説明
Compiler - ソース	「インクルード・ファイル・ディレクトリ」にインクルードパスを追加	各 FIT モジュールで設定が必要なインクルードパスを追加しています。コード生成されたファルダについては、「FIT Configurator」を使用して各 FIT モジュールを組み込む場合に限り自動で設定されます。 コード生成しないプログラムファイルについては、インクルードパスを手動で追加する必要があります。 本プロジェクトでは下記の 3 項目を追加しています。 
Compiler - ソース	プロセッサマクロ定義を追加 ＜設定＞ 	“FLASH_APPL_BANK=0”とすると BANK0 用のコードがビルドされます。 “_DISABLE_REVNO_CHECK”を定義するとファームウェアと EEPROM のリビジョン No が異なっていてもエラーになりません。 “REVISION_NUMBER”は 1.10 として定義しています。
Compiler – 最適化	最適化レベルを変更 レベル 1：一部最適化を実施する	SSC が生成するコードの一部が最適化によりビルドされないことを防ぐため、最適化レベルを 1 に設定しています。
Linker – セクション	RAM 領域に、RPFRAM2 セクションを追加 ＜設定＞ 	バンクを切り換えるコードを RAM 領域に追加します。

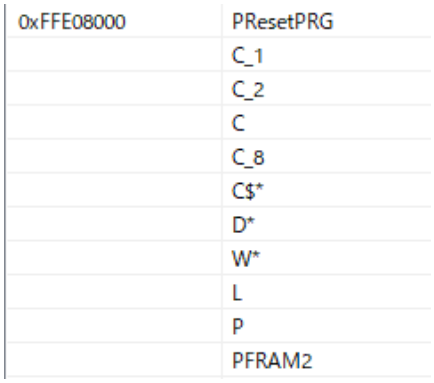
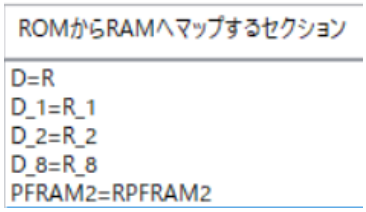
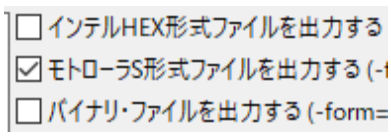
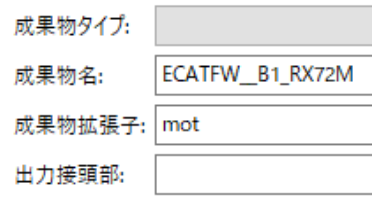
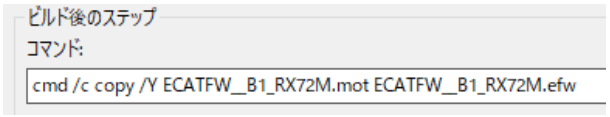
	ROM 領域に配置される PResetPRG の開始位置を 0xFFE08000 に変更。 PFRAM2 セクションを追加。 ＜設定＞ 	BANK0 のマッピングは 0xFFE00000 から 0xFFFFFFFF ですが、PResetPRG の開始位置を 0xFFE08000 と設定しています。 BANK0 のサイズは 2048KB です。 コードフラッシュメモリを書き換えるためのコードを ROM 領域に追加しています 本サンプルプログラムの 0xFFE00000～0xFFE08000 領域は、Store Parameters のために空けています。Store Parameters の詳細については 9.2.3 を参照してください。
	ROM 領域に IDENTIFY セクションを追加し、開始位置を 0xFFFFFF70 とする。 ＜設定＞ 	IDENTIFY セクションにはファームウェアのリビジョン No を含む 16 バイトのテーブルデータが配置されます。 BANK0 の最後尾に IDENTIFY セクションとリセットベクタを含む例外ベクタテーブル(128 バイト)を配置します。
Linker – セクション – シンボルファイル	ROM から RAM へマップするセクションを追加 ＜設定＞ 	バンクの切り替えを行うコードは RAM 上で実行します。そのため ROM から RAM にマップする設定を追加します。
Converter – 出力	モトローラ S 形式ファイルを出力する設定に変更 ＜設定＞ 	ダウンロードファイルはモトローラ S 形式ファイルとしています。

表 7-15　Download - その他タブ

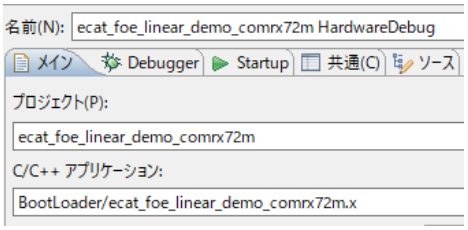
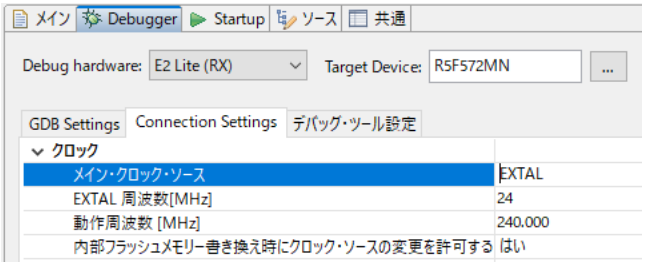
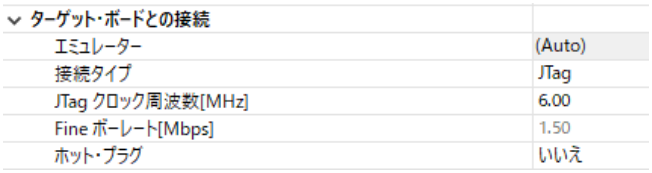
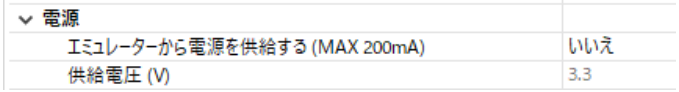
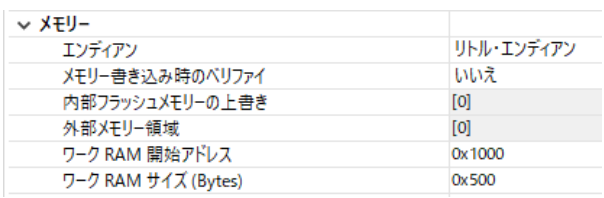
タブ項目	変更内容	説明
ビルド成果物	出力ファイルを ECATFW__B1_RX72M.mot に変更 ＜設定＞ 	ダウンロードファイル名の接頭辞は” ECATFW__B1”としています。
ビルド・ステップ	ビルド終了後に出力ファイルのコピーを ECATFW__B1_RX72M.efw として作成 ＜設定＞ 	TwinCAT でダウンロード可能なファイルの拡張子は”efw”となっているためです。

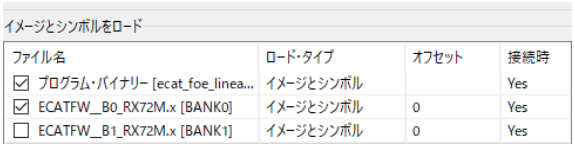
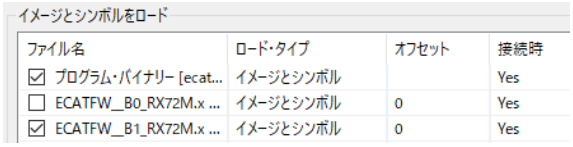
7.4　デバッグ構成

7.4.1　リニアモードのデバッグ構成

設定内容の確認は、[実行] → [デバッグ構成] → [ecat_linear_demo_cpurx72m] にて行うことができます。

表 7-16　リニアモードのデバッグ構成

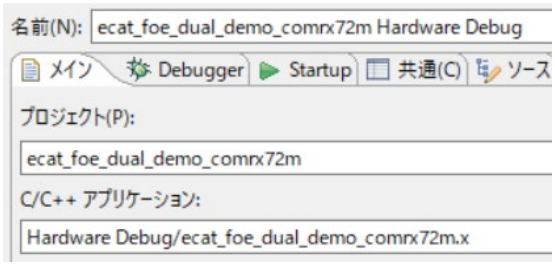
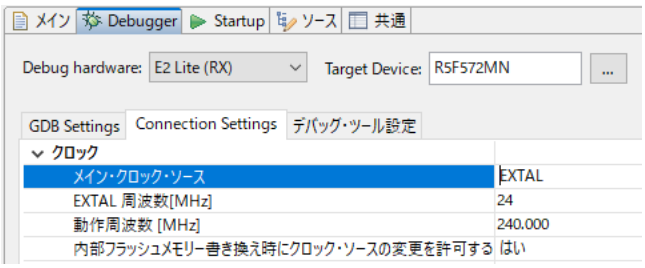
タブ項目	変更内容	説明
メイン	C/C++アプリケーションをブートローダーに変更する。 ＜設定＞ 	
Debugger – Connection Settings – クロック	メイン・クロック・ソースを EXTAL、EXTAL 周波数を 24MHz、動作周波数を 240MHz に設定する。 ＜設定＞ 	
Debugger – Connection Settings – ターゲット・ボードとの接続	接続タイプを”JTAG”に設定する。 CPU カードを使用する場合は”FINE”に設定する。 ＜設定例＞ 	
Debugger – Connection Settings – 電源	エミュレータから電源を供給しない設定にする。 ＜設定例＞ 	
Debugger – デバッグツール設定	内部フラッシュメモリーの上書きを全ブロックに変更する。 ＜設定＞ 	「内部フラッシュメモリーの上書き」の右端にあるボタンをクリックし、「すべての選択解除」→「OK」をクリックすると全ブロックを消去してから上書きする設定となり、[0]の表示となります。

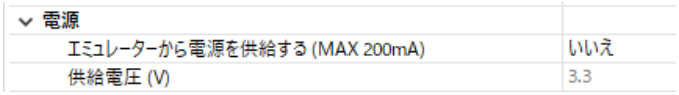
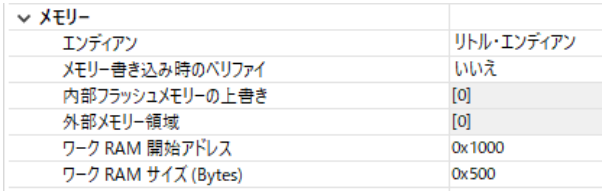
Startup	ブートローダーと一緒にコードフラッシュに書き込むイメージを設定する。 ＜設定例＞ブートローダーと BANK0 をデバッグ  イメージとシンボルをロード ファイル名 / ロード・タイプ / オフセット / 接続時 プログラム・バイナリー [ecat_foe_linea... / イメージとシンボル / / Yes ECATFW_B0_RX72M.x [BANK0] / イメージとシンボル / 0 / Yes ECATFW_B1_RX72M.x [BANK1] / イメージとシンボル / 0 / Yes ＜設定例＞ブートローダーと BANK1 をデバッグ  イメージとシンボルをロード ファイル名 / ロード・タイプ / オフセット / 接続時 プログラム・バイナリー [ecat... / イメージとシンボル / / Yes ECATFW_B0_RX72M.x ... / イメージとシンボル / 0 / Yes ECATFW_B1_RX72M.x ... / イメージとシンボル / 0 / Yes	デバッグしたいバンクのイメージとシンボルをロードする設定にします。 ダウンロードモジュールの指定は「追加」→「プロジェクトの検索」から行います。

7.4.2　デュアルモードのデバッグ構成

設定内容の確認は、[実行] → [デバッグ構成] → [ecat_dual_demo_cpurx72m]にて行うことができます。

表 7-17　デュアルモードのデバッグ構成

タブ項目	変更内容	説明
メイン	C/C++アプリケーションを設定する。 ＜設定＞  名前(N): ecat_foe_dual_demo_comrx72m Hardware Debug メイン / Debugger / Startup / 共通(C) / ソース プロジェクト(P): ecat_foe_dual_demo_comrx72m C/C++ アプリケーション: Hardware Debug/ecat_foe_dual_demo_comrx72m.x	
Debugger – Connection Settings – クロック	メイン・クロック・ソースを EXTAL、EXTAL 周波数を 24MHz、動作周波数を 240MHz に設定する。 ＜設定＞  メイン / Debugger / Startup / ソース / 共通 Debug hardware: E2 Lite (RX)　Target Device: R5F572MN GDB Settings / Connection Settings / デバッグ・ツール設定 クロック メイン・クロック・ソース: EXTAL EXTAL 周波数[MHz]: 24 動作周波数 [MHz]: 240.000 内部フラッシュメモリー書き換え時にクロック・ソースの変更を許可する: はい	

Debugger – Connection Settings – ターゲット・ボードとの接続	接続タイプを”JTAG”に設定する。 CPU カードを使用する場合は”FINE”に設定する。 ＜設定例＞  	
Debugger – Connection Settings – 電源	エミュレータから電源を供給しない設定にする。 ＜設定例＞  	
Debugger – Connection Settings – 通信モード	[CPU 動作モード] – [起動バンクを変更する] を「いいえ」に設定し、BANK0 のユーザープログラムを起動するようにする。	デバッガ起動時に BANKSWP[2:0]ビットは”111b”に設定されます。
Debugger – デバッグツール設定	内部フラッシュメモリーの上書きを全ブロックに変更する。 ＜設定＞  	「内部フラッシュメモリーの上書き」の右端にあるボタンをクリックし、「すべての選択解除」→「OK」をクリックすると全ブロックを消去してから上書きする設定となり、[0]の表示となります。

8. サンプルプログラム概要

8.1 ファイル構成

“ecat_linear_demo_cpurx72m”フォルダ以下のファイル構成を表 8-1 に示します。

表 8-1　本サンプルプログラムのファイル構成

フォルダ	サブフォルダ	ファイル	内容
project	-	.cproject	CPROJECT ファイル
		.project	PROJECT ファイル
		ecat_linear_demo_cpurx72m HardwareDebug.launch	デバッグ時の設定ファイル
		ecat_linear_demo_cpurx72m.scfg	スマートコンフィグレーションファイル
		ecat_linear_demo_cpurx72m_board.xml	端子設定保存ファイル
utilities	esi	Renesas EtherCAT RX72M.xml	ESI ファイル
	patch	apply_patch.bat	パッチを適応するバッチファイル
		RX72M.patch	本サンプルプログラム用パッチファイル
	ssc_config	RX72M EtherCAT.esp	SSC Tool プロジェクトファイル
		RX72M_EtherCAT_config.xml	SSC Tool コンフィグファイル

“project”フォルダにはプログラムファイルが格納された“src”フォルダも含まれています。

表 8-2 に“src”フォルダ以下のフォルダ構成及び各ファイル詳細を示します。

表 8-2　“src”フォルダ以下のフォルダ構成

フォルダ名	内容	
application¥ecat¥beckhoff	SSC で生成され、パッチで修正されたプログラムの格納先フォルダ	
application¥ecat¥renesas	SSC で生成されない EtherCAT 通信用プログラムを格納するフォルダ	
	ファイル名	内容
	cia402sample.c	CiA402 遷移関数定義ファイル
	cia402sample.h	CiA402 遷移関数宣言ファイル
	foesample.c	FoE プロトコル用関数定義ファイル
	foesample.h	FoE プロトコル用関数宣言とオブジェクト 0x5000 定義用ファイル
	iosample.c	I/O 動作確認用関数定義ファイル
	iosample.h	I/O 動作確認用関数宣言ファイル
	semisample.c	Store Parameters 用関数定義ファイル
	semisample.h	Store Parameters 用関数宣言と CDP 関連オブジェクト定義ファイル

フォルダ名	内容	
demo	main 関数やブートローダーを含むプログラムを格納するフォルダ	
	ファイル名	**内容**
	bt_main.c	ブートローダープログラム（リニアモードのみ使用）
	main.c	本サンプルプログラムのメイン文
	siorw.c	SCI 関連ソースプログラム
r_fw_up_rx	ファームウェアアップデート用プログラムを格納するフォルダ	
	ファイル名	**内容**
	r_fw_up_rx.c	ファームウェアアップデートソースファイル
	r_fw_up_rx_if.h	ファームウェアアップデートインタフェースファイル
	r_fw_up_rx_private.h	ファームウェアアップデートヘッダファイル
	r_fw_up_buf.c	ファームウェアデータ用バッファ処理ソースファイル
	r_fw_up_buf.h	ファームウェアデータ用バッファ処理ヘッダファイル
	r_fw_up_bank.c	バンクに関する情報を取り扱うソースファイル
smc_gen	FIT モジュール関連のプログラムを格納するフォルダ	

application¥ecat¥beckhoff フォルダに格納されるプログラムファイルの詳細については Application Note ET9300 (EtherCAT Slave Stack Code)を参照してください。

smc_gen フォルダに格納される FIT モジュール関連ファイルの詳細については、各 FIT モジュールのドキュメントを参照してください。

8.2　スレーブスタックコードの修正事項

本サンプルプログラムでは、SSC で生成された EtherCAT 通信用プログラムをパッチで修正しています。

表 8-3 に修正の概要を示します。

表 8-3　SSC 生成ソースコードの修正の概要

ファイル名	修正の概要
bootmode.c	IDENTIFY セクションに配置される定数配列を追加。 ブートローダー用関数の定義を追加。
bootmode.h	ブートローダー用関数の宣言を追加。
cia402appl.c	CiA402_Init 関数のオブジェクトディクショナリエントリ部分を修正。 CiA402 アプリケーション関数に CiA402 状態遷移関数を追加。 CiA402_LocalError 関数と CiA402_DummyMotionControl 関数定義を削除。 プロセスデータ入出力関数を PDO エントリに合わせて修正。 APPL_Application 関数に I/O と CiA402 アプリケーション関数を追加。
cia402appl.h	オブジェクトディクショナリ定義の修正。
coeappl.c	オブジェクトディクショナリ定義の修正。
coeappl.h	条件付きコンパイルを追加。
ecatappl.c	MainInit 関数と Mainloop 関数を修正。
ecatcoe.h	構造体のアライメントを 1 にするよう修正。
ecatslv.c	AL event で sync0, sync1 event を発生させないように修正。 AL_ControlInd 関数に SII のファームウェアバージョン情報の更新関数と ファームウェアアップデート後の再起動関数を追加。
mailbox.h	構造体のアライメントを 1 にするよう修正。
objdef.c	条件付きコンパイルを追加。
sdoserv.h	構造体のアライメントを 1 にするよう修正。

修正の詳細については、RX72M.patch ファイルを参照してください。

8.3　CiA402 Drive Profile 概要

CiA402 ドライブプロファイルはドライブおよびモーションコントロール用のデバイスプロファイルであり、主にサーボドライブ、正弦波インバータ、およびステッピングモーター用コントローラの機能動作を定義します。

このプロファイルでは、複数の動作モードと対応する設定パラメータがオブジェクトディクショナリとして規定されます。

また、状態ごとの内部および外部動作を規定する有限状態オートマトン（Finite State Automaton: FSA）も含まれます。

状態を変更する場合はコントロールワードオブジェクトを通じて指定することで、現在の状態を示すステータスワードオブジェクトに遷移後の結果が反映されます。

コントロールワードと各種コマンド値（速度など）は RxPDO に割り当てられ、ステータスワードと各種実際値（位置など）は TxPDO に割り当てられます。

詳細については CiA402 規格書の内容を確認してください。

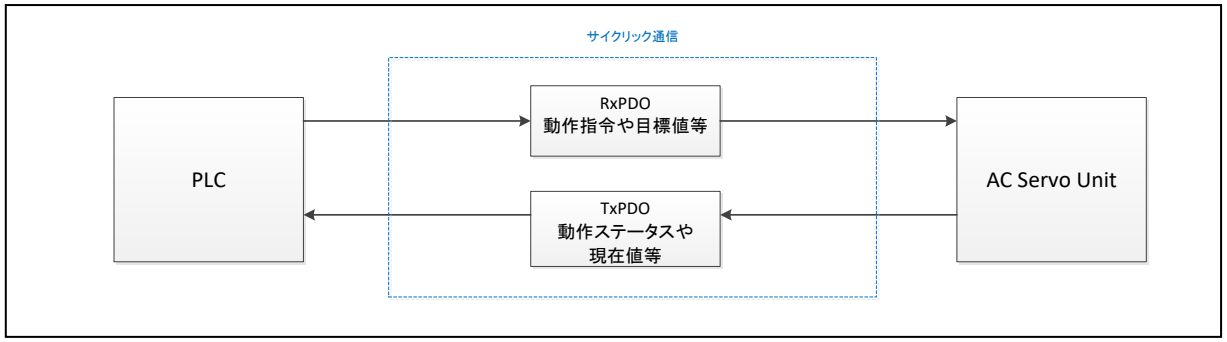


図 8-1　CiA402 通信の流れ

8.3.1　動作モード

CiA402 として規定されている動作モードのうち、本アプリケーションノートでサポートしている動作モードを表 8-4 に示します。

表 8-4 サポート動作モード一覧表

Operation Mode	Support
Profile position mode	×
Velocity mode(frequency converter)	×
Profile velocity mode	×
Profile torque mode	×
Homing mode	×
Interpolated position mode	×
Cyclic synchronous position mode	○
Cyclic synchronous velocity mode	○
Cyclic synchronous torque mode	×
Cyclic synchronous torque mode with　commutation angle	×
Manufacturer specific mode	×

8.3.2　状態遷移

CiA402 として規定されている FSA として、本アプリケーションノートでは以下をサポートします。

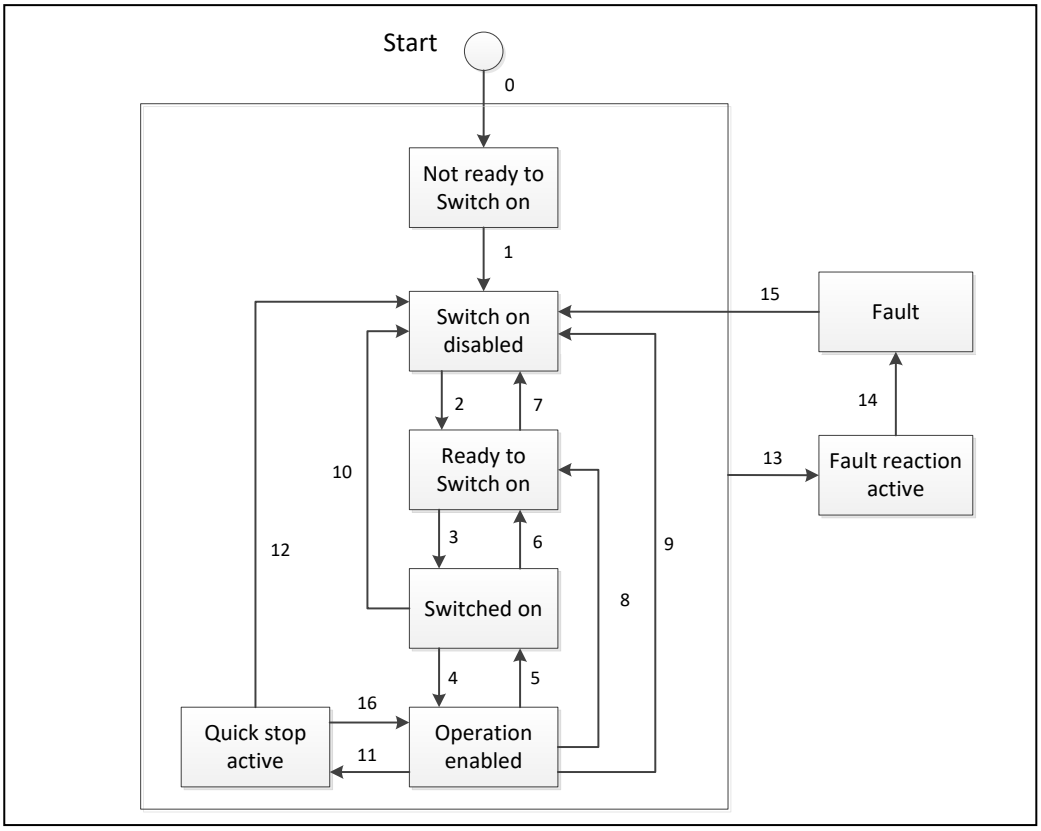


図 8-2　CiA402 状態遷移図

8.3.3　状態遷移関数

CiA402 状態遷移関数一覧を表 8-5 に示します。CiA402_StateTransition 関数と CiA402_LocalError 関数は図 8-1 に示す CiA402 FSA の各状態遷移の番号とリンクしており、状態遷移が発生した際に対応する関数が呼び出されます。

本サンプルプログラムをモータ制御等に使用する場合は、各状態遷移関数内に状態遷移が発生した際に実行する処理を記述してください。

表 8-5　CiA402 プロトコル・スタック I/F 関数一覧

関数名	概要
CiA402_StateTransition1	状態遷移 1 が発生した時に呼び出される関数。
CiA402_StateTransition2	状態遷移 2 が発生した時に呼び出される関数。
CiA402_StateTransition3	状態遷移 3 が発生した時に呼び出される関数。
CiA402_StateTransition4	状態遷移 4 が発生した時に呼び出される関数。
CiA402_StateTransition5	状態遷移 5 が発生した時に呼び出される関数。
CiA402_StateTransition6	状態遷移 6 が発生した時に呼び出される関数。
CiA402_StateTransition7	状態遷移 7 が発生した時に呼び出される関数。
CiA402_StateTransition8	状態遷移 8 が発生した時に呼び出される関数。
CiA402_StateTransition9	状態遷移 9 が発生した時に呼び出される関数。
CiA402_StateTransition10	状態遷移 10 が発生した時に呼び出される関数。
CiA402_StateTransition11	状態遷移 11 が発生した時に呼び出される関数。
CiA402_StateTransition12	状態遷移 12 が発生した時に呼び出される関数。
CiA402_LocalError	状態遷移 13 が発生した時に呼び出される関数。
CiA402_StateTransition14	状態遷移 14 が発生した時に呼び出される関数。
CiA402_StateTransition15	状態遷移 15 が発生した時に呼び出される関数。
CiA402_StateTransition16	状態遷移 16 が発生した時に呼び出される関数。
APPL_MOTOR_MotionControl_Main	“Operation enabled”状態のときに呼び出される関数。 動作モード毎に処理を記述してください。

表 8-5 に示される関数の詳細は cia402sample.c を参照してください。

8.4　ファームウェアアップデート概要

8.4.1　バンク番号について

サンプルプログラムはコードフラッシュメモリに格納され、プログラムの実行中にファームウェアの更新が可能です。

サンプルプログラムはユーザープログラムを実行するためのバンクとアップデートプログラムを書き込むためのバンクの 2 バンクをサポートします。

本アプリケーションノートにおいては、バンクまたは BANK 番号の割り当てはリニアモードとデュアルモードで共通にするため、固定とします。

デュアルモードにおいて起動バンク切り替えビット（BANKSEL.BANKSWP[2:0]）の値により番号が入れ替わることはしませんので、注意してください。

リセットベクタ(FFFF FFFCH－FFFF FFFFH)に近い方から、0→1 としています。

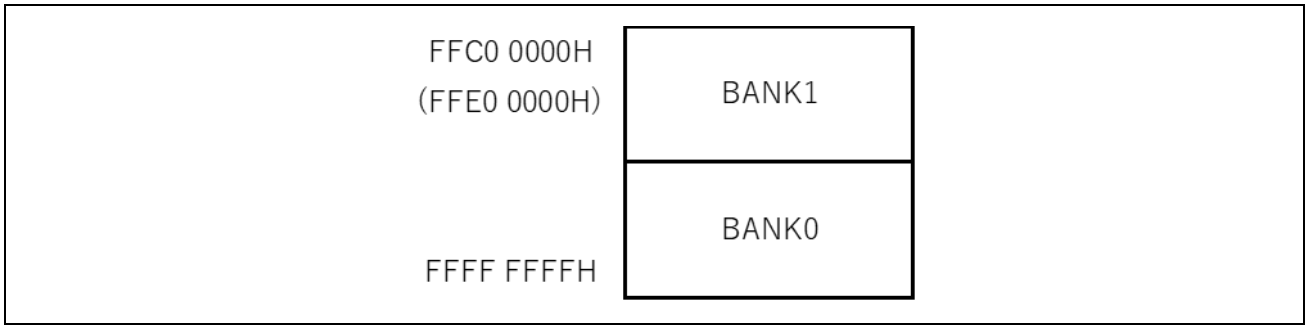


図 8-3　バンク番号

8.4.2　リニアモードとデュアルモードの比較

本アプリケーションノートではリニアモードとデュアルモードの両方をサポートしますが、モードによる機能の違いや制約があります。

それらを表にまとめますので、モードを選択する際の参考にしてください。

表 8-6　サンプルプログラムの機能比較

項目	リニアモード	デュアルモード
起動時のユーザープログラム切り替え	ブートローダープログラムによる切り替え	ハードウェアのデュアルバンク機能による切り替え
コードフラッシュブロック構成	2 つのバンクで 8K バイトと 32K バイトブロックの構成が異なる	2 つのバンクは同じ構成
Trusted Memory 対象領域	ブロック 8,9	ブロック 8,9 およびブロック 78,79
1 バンクで使用可能なユーザープログラムサイズ	搭載コードフラッシュメモリ容量の 1/2 からブートローダーに割り当てた 32KB を除いた容量	搭載コードフラッシュメモリ容量の 1/2 の全て
再ブート時の EtherCAT リンク切れ※	リンク切れ無し	リンク切れ有り （ソフトウェアリセットを使用するため）
ファームウェアアップデートの運用	ユーザーは 2 つのバンクのどちらに書き込み可能かを判断して、適切なダウンロードファイルをダウンロードする必要あり	ユーザーはバンクを意識することなく、ダウンロードファイルをダウンロード可能

※再ブート時に EtherCAT のリンク切れが発生しても再接続が可能です

8.4.3　リニアモードの動作概要

リニアモードではコードフラッシュを 3 つの領域に分けて使用します。

表 8-7　リニアモードのコードフラッシュ領域区分

名称	機能
BANK1	ユーザープログラム格納用領域：BANK1 BANK0 でユーザープログラムを実行しているときに書き換え可能です。
BANK0	ユーザープログラム格納用領域：BANK0 BANK1 でユーザープログラムを実行しているときに書き換え可能です。
Boot Loader	起動時に BANK0 と BANK1 に格納してある Rev No.を比較して新しい方の BANK のユーザープログラムを選択、分岐することでユーザープログラムを実行します。 ユーザープログラムによる書き換え対象には含まれません。

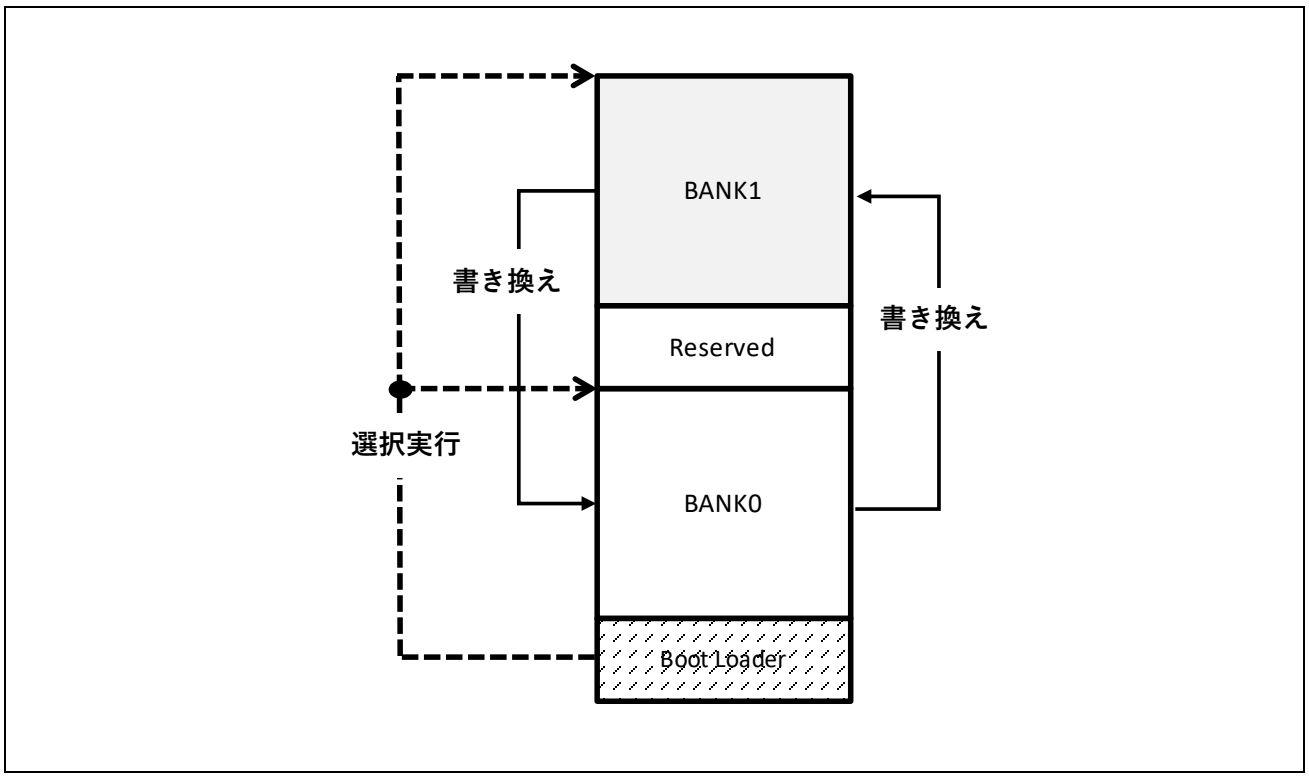


図 8-4　リニアモードのコードフラッシュ使用イメージ

各領域のアドレスと容量および各領域に属するブロック No の関係は下表のようになります。

表 8-8　各領域のアドレス範囲と容量およびブロック No (リニア 4MB 製品の場合)

名称	先頭アドレス	最後尾アドレス	容量	ブロック No.
BANK1	FFC0_0000H	FFDF_7FFFH	2016KB	133～71 (32KB)
Reserved	FFDF_8000H	FFDF_FFFFH	32KB	70 (32KB)
BANK0	FFE0_0000H	FFFF_7FFFH	2016KB	69～8(32KB) 7～4(8KB)
Boot Loader	FFFF_8000H	FFFF_FFFFH	32KB	3～0(8KB)

表 8-9　各領域のアドレス範囲と容量およびブロック No (リニア 2MB 製品の場合)

名称	先頭アドレス	最後尾アドレス	容量	ブロック No.
BANK1	FFE0_0000H	FFEF_7FFFH	992KB	69～39(32KB)
Reserved	FFEF_8000H	FFEF_FFFFH	32KB	38(32KB)
BANK0	FFF0_0000H	FFFF_7FFFH	992KB	37～8 (32KB) 7～4(8KB)
Boot Loader	FFFF_8000H	FFFF_FFFFH	32KB	3～0(8KB)

図で示すと下図のようになります。

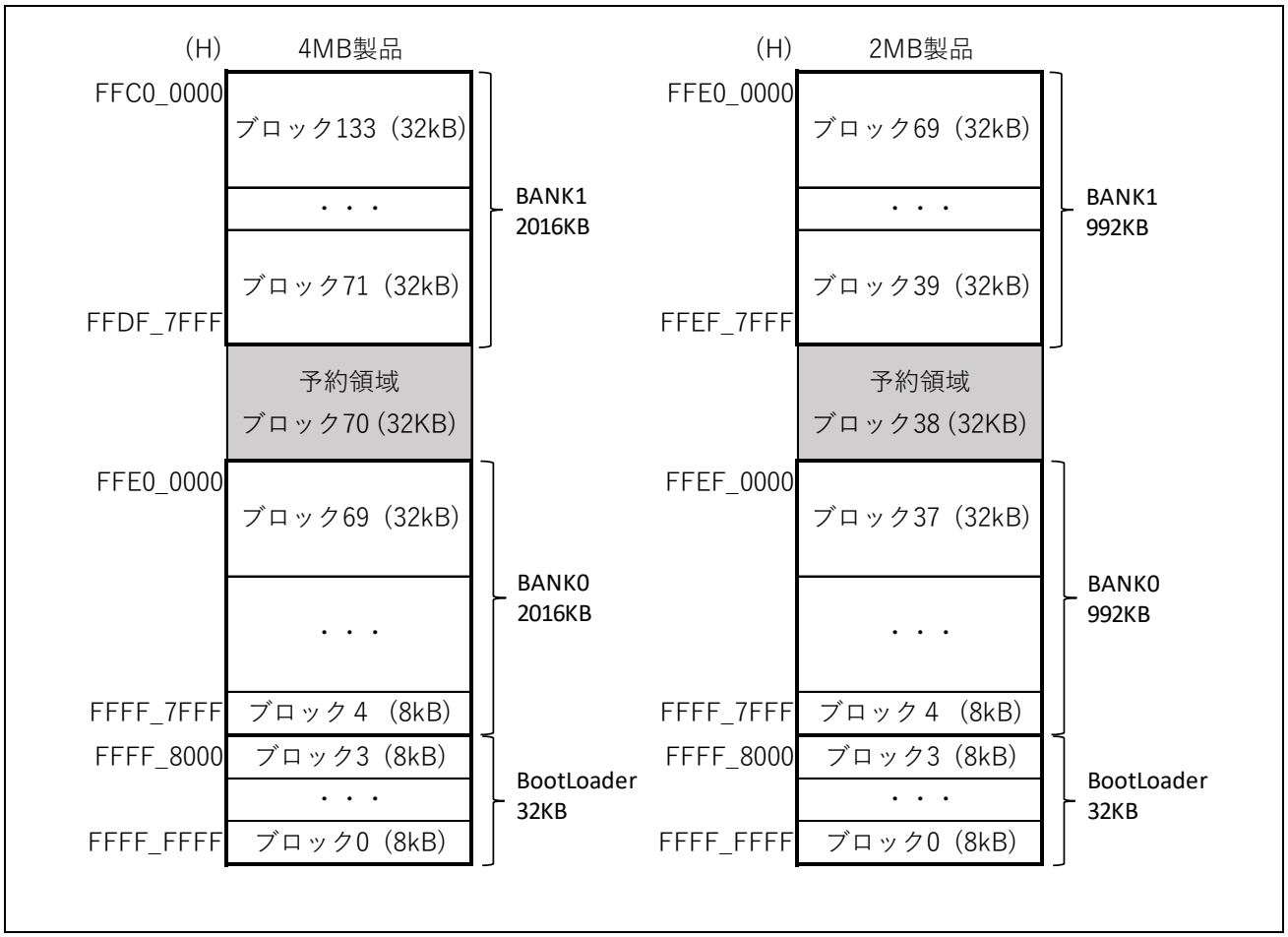


図 8-5　リニアモードにおけるブロック割り当て

8.4.4　デュアルモードの動作確認

ここでは、ユーザープログラムを Rev1.0 から Rev1.1 に書き換え、再起動するシーケンスを例にデュアルモードの動作を説明します。

(1) ブート後、コードフラッシュ領域の後半 2MB に格納された Rev1.0 のユーザープログラムが起動します。このときの起動バンク切り替えビット(BANKSEL.BANKSWP[2:0])の値は 111b とします。

(2) Rev1.0 のユーザープログラムを実行中に前半 2MB をイレーズし、Rev1.1 のユーザープログラムに書き換えます。

(3) 起動バンク切り替えビットをビット反転(BANKSEL.BANKSWP[2:0]=000b)した後、ソフトウェアリセットを実行し再起動します。

(4) 起動バンク選択機能により Rev1.1 が後半 2MB に、Rev1.0 が前半 2MB に入れ替わります。Rev1.1 のユーザープログラムが起動します。

Rev1.1 から Rev1.2 への更新も同様のシーケンスとなります。

したがってデュアルモードでのファームウェア更新は次の特徴があります。

- コードフラッシュを書き換えるのは常に前半 2MB(1MB)が対象
- 再起動後にバンクが入れ替わるので、後半 2MB(1MB)で実行するコードで書き換える

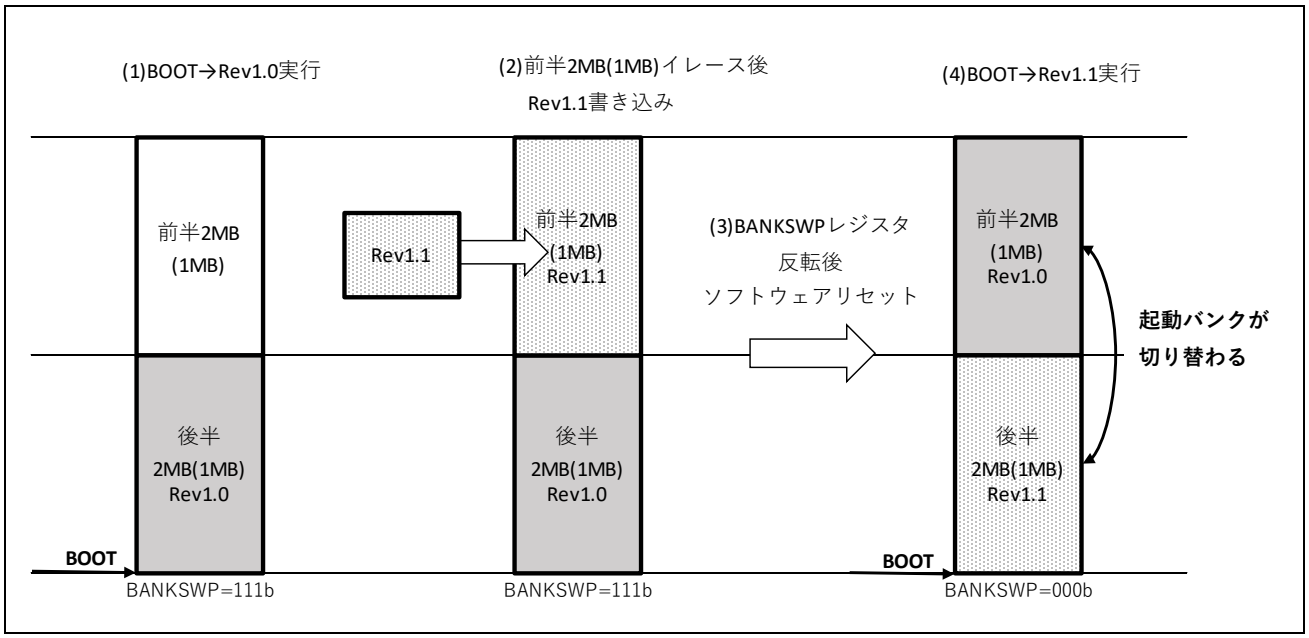


図 8-6　デュアルモードの動作シーケンス

「8.4.1 バンク番号について」で説明したように、本アプリケーションノートでは前半 2MB(1MB)を BANK1、後半 2MB(1MB)を BANK0 とします。

各領域のアドレスと容量および各領域に属するブロック No の関係は下表のようになります。

表 8-10　各領域のアドレス範囲と容量およびブロック No (デュアル 4MB 製品の場合)

名称	先頭アドレス	最後尾アドレス	容量	ブロック No.
BANK1	FFC0_0000H	FFDF_FFFFH	2048KB	139～78 (32KB) 77～70(8KB)
BANK0	FFE0_0000H	FFFF_FFFFH	2048KB	69～8(32KB) 7～0(8KB)

表 8-11　各領域のアドレス範囲と容量およびブロック No (デュアル 2MB 製品の場合)

名称	先頭アドレス	最後尾アドレス	容量	ブロック No.
BANK1	FFD0_0000H	FFDF_FFFFH	1024KB	107～78(32KB) 77～70(8KB)
BANK0	FFF0_0000H	FFFF_FFFFH	1024KB	37～8 (32KB) 7～0(8KB)

8.4.5　バンク関連

(1)　バンクのイレーズ

バンクにユーザープログラムを書き込む前にイレーズする必要があります。

リニアモードではプログラム実行中のバンクとは反対側のバンクをイレーズします。

（BANK0→BANK1、BANK1→BANK0）

一方、デュアルモードではイレーズするのは常に BANK1 になります。

(2)　書き込み可能バンクオブジェクト

どちらのバンクにファームウェアを書き込むことが出来るのかを EtherCAT マスターから確認できるように、書き込み可能なバンクを示すオブジェクトとして”Firmware Writable Bank”を追加しています。

BANK1 が書き換え可能な場合、オブジェクト値は”1”、BANK0 が書き換え可能な場合、オブジェクト値は”0”を示すものとします。

オブジェクトの詳細を表 8-11 に示します。

表 8-11　書き込み可能バンクオブジェクト

OBJECT Name	INDEX	Category	Access	Data Type	PDO Mapping
Firmware Writable Bank	0x5000	Optional	RO	UINT8	No

(3)　リブートと起動バンクの選択

新しいファームウェアをバンクに書き込み終えるとリブートして新しいファームウェア実行します。

リブートの方法と起動バンクの選択方法を表 8-12 に示します。

表 8-12　リブートとバンクの選択

項目	リニアモード	デュアルモード
リブートの方法	ユーザープログラムからブートローダーに分岐する。	ユーザープログラムでソフトウェアリセットを実行する。
起動バンクの選択方法	ブートローダーが BANK0 と BANK1 に格納してある Rev No.を比較して新しい方の BANK のユーザープログラムを選択する。	起動バンク切り替えビット (BANKSEL.BANKSWP[2:0])の値により選択される。 リブートの前に起動バンク切り替えビットの値を反転することで次のリセット時にバンクが切り替わる。

8.4.6　ダウンロード関連

(1)　ダウンロードファイルの形式

サンプルプログラムではビルド成果物として TwinCAT で転送可能なダウンロードファイルを出力します。ダウンロードファイルの形式を表 8-13 に示します。

表 8-13　ダウンロードファイルの形式

項目	内容
ファイル名接頭辞	バンク 0：”ECATFW__B0” バンク 1：”ECATFW__B1”
ファイル拡張子	efw
ファイル形式	モトローラ S 形式

(2)　ダウンロードファイルとバンクの関係

ダウンロードファイルは各バンク専用となり、他のバンクやモード向けのものは使用できません。ダウンロードファイルとバンクの関係を示します。

デュアルモードのダウンロードファイルは BANK1 向けですが、アドレス範囲が BANK0 の範囲であることに注意してください。これは書き換え後に起動バンクを入れ替えることで BANK0 として実行されるためです。

表 8-14　ダウンロードファイルとバンクの関係

バンクモード	リニアモード		デュアルモード
ダウンロードファイル	ECATFW__B1_xxx.efw	ECATFW__B0_xxx.efw	ECATFW__B1_xxx.efw
書き換え対象バンク	BANK1	BANK0	BANK1
アドレス範囲 4MB 製品	FFC0_0000H~ FFDF_7FFFH	FFE0_0000H~ FFFF_7FFFH	FFE0_0000H~ FFFF_FFFFH
アドレス範囲 2MB 製品	FFE0_0000H~ FFEF_7FFFH	FFF0_0000H~ FFFF_7FFFH	FFF0_0000H~ FFFF_FFFFH
ダウンロード前に確認すべき Firmware Writable Bank の値	1	0	1
ダウンロード時にプログラムを実行しているバンク	BANK0	BANK1	BANK0

(3)　ファイルダウンロード時のチェック項目

ダウンロードファイルのデータでファームウェアを書き換えるため、不正なファイルを使用すると動作しなくなる可能性があります。

このため、幾つかのチェック項目を設けています。チェック項目とチェック内容などを表 8-15 に示します。

表 8-15　ダウンロードファイルのチェック項目

項目	チェック内容	チェックするタイミング
ファイル名接頭辞	BANK0 の場合：”ECATFW__B0” BANK1 の場合：”ECATFW__B1”	ダウンロード開始時に接頭辞が正しいかチェック
ファイルパスワード	数字 8 桁 デフォルト値：00000000	ダウンロード開始時にパスワードが一致するかチェック
アドレス範囲	モトローラ S レコードフォーマットのアドレスフィールド	ダウンロード中はアドレスが書き換え対象バンクのアドレス範囲内かチェック
チェックサム	モトローラ S レコードフォーマットのチェックサムフィールド	ダウンロード中は受信レコードから算出したチェックサムが一致するかチェック

(4)　リニアモードでのダウンロードファイル書き込み

リニアモードではプログラムを実行しているバンクとは反対側のバンクにダウンロードファイルを書き込みます。

BANK0 のユーザープログラムを実行中に BANK0 用のダウンロードファイルを書き込むことはできません。また、同様に BANK1 のユーザープログラムを実行中に BANK1 用のダウンロードファイルを書き込むことはできません。

このため、ユーザーは事前に書き込み対象バンクがどちらかを調べ、適切なダウンロードファイルを用意する必要があります。

書き込み可能なバンクは、オブジェクト”Firmware Writable Bank”の値として EtherCAT マスターから確認できます。

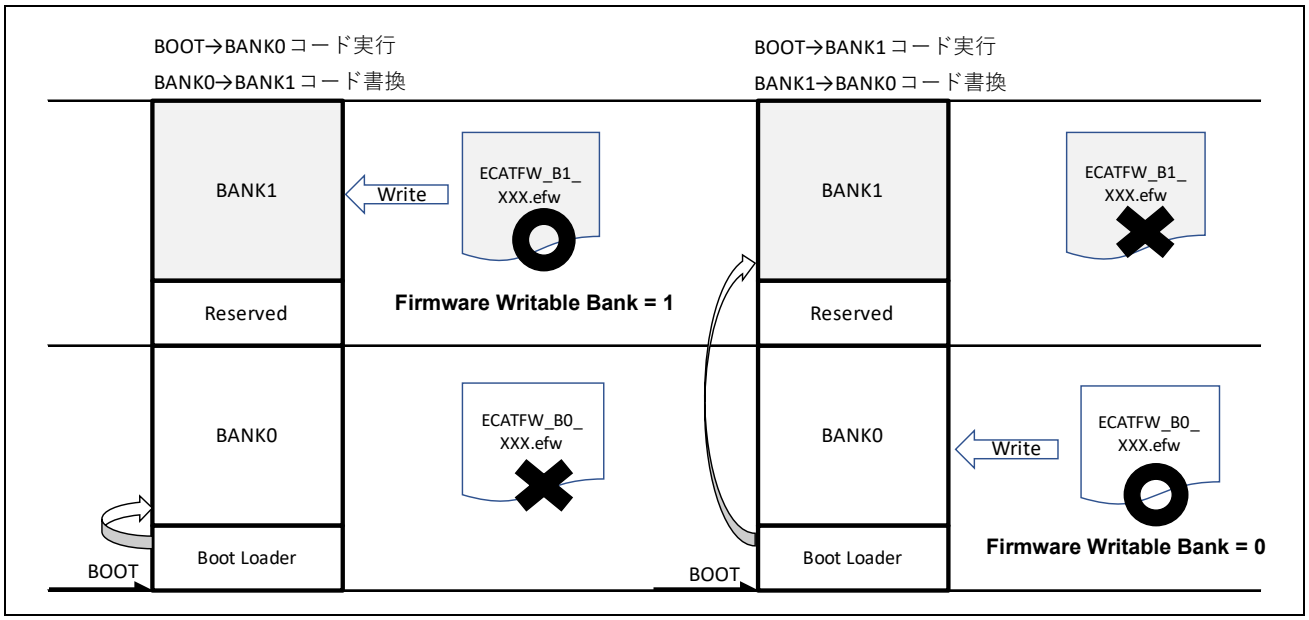


図 8-7　リニアモードにおけるバンクとダウンロードファイルの関係

(5)　デュアルモードでのダウンロードファイル書き込み

デュアルモードでモードではリブート後に起動バンクを入れ替えるため

- 書き込み対象バンクは常に BANK1 にする
- ダウンロードファイルは BANK0 にマッピングしたものを使用する

ことになります。

ダウンロードファイルのファイル形式であるモトローラ S レコードフォーマットに含まれるアドレスは BANK0 を示しているので、BANK1 に書き込む際はアドレスを-2MB または-1MB して書き込みます。

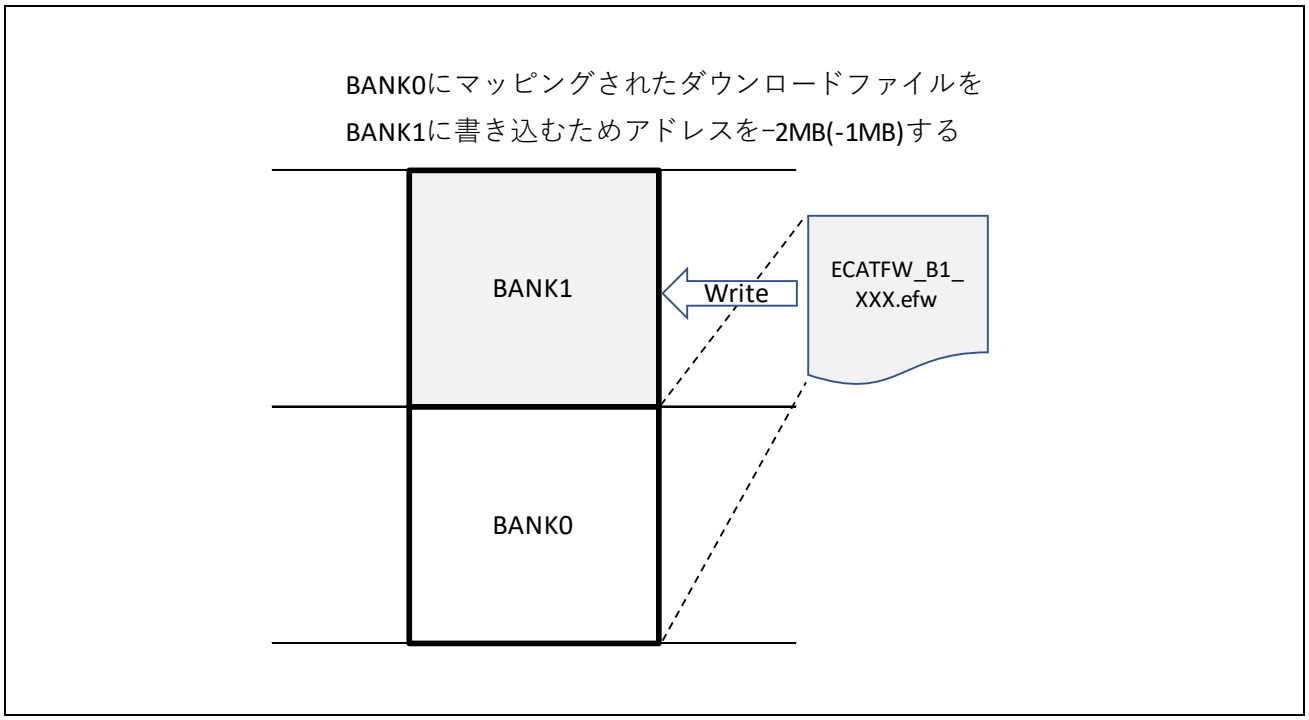


図 8-8　デュアルモードにおけるダウンロードファイルの書き込み

8.4.7　SII EEPROM

(1)　SII EEPROM 更新

サンプルプログラムにはファームウェア更新の際に SII EEPROM の Revision も更新する機能が含まれています。

SII EEPROM のアドレス 16~31 バイトには Identify オブジェクト(Index:1018)の内容が格納されます。

Identify オブジェクトの構成を表 8-16 に示します。

表 8-16　Identify オブジェクトの構成

名称	内容	Index	オフセットアドレス
Vendor ID	ecat_def.h の”VENDOR_ID”として定義されます。	1018:01	0-3 Byte
Product code	ecat_def.h の”PRODUCT_CODE”として定義されます。	1018:02	4-7 Byte
Revision	ecat_def.h またはプリプロセッサマクロの”REVESION_NUMBER”として定義されます。	1018:03	8-11 Byte
Serial number	ecat_def.h の”SERAIL_NUMBER”として定義されます。	1018:04	12-15 Byte

Identify オブジェクトの内容を IDENTIFY セクションとしてコードフラッシュの固定アドレスに割り当てています。

IDNTIFY セクションのベースアドレスを表 8-17 に示します。

表 8-17　IDENTIFY セクションのベースアドレス

モード	バンク	ベースアドレス(H)
リニアモード	BANK0	FFFF_7F80
	BANK1	FFDF_7F80
デュアルモード	BANK1	FFFF_7F80

ファームウェア更新後にフラッシュに書き込まれたファームウェア Revision の値に SII EEPROM の Revision を更新します。

(2)　ファームウェアリビジョンチェック

INIT->PREOP 遷移時に実行中のファームウェアのリビジョンと SII EEPROM の Revision が一致しているかチェックします。

不一致の場合は EtherCAT マスターにエラーコードを返します。

AL stratus code (0x0006) AL Status (ERROR INIT)

なおエラーコードの発行は、プリプロセッサマクロ”_DISABLE_REVNO_CHECK”を定義することで無効化できます。

8.4.8　ファームウェアアップデートプログラム詳細

8.4.8.1　ファイル構成

表 8-18　ファームウェアアップデートプログラムで使用するファイル

ファイル名	概要
r_fw_up_rx.c	ファームウェアアップデートソースファイル
r_fw_up_rx_if.h	ファームウェアアップデートインタフェースファイル
r_fw_up_rx_private.h	ファームウェアアップデートヘッダファイル
r_fw_up_buf.c	ファームウェアデータ用バッファ処理ソースファイル
r_fw_up_buf.h	ファームウェアデータ用バッファ処理ヘッダファイル
r_fw_up_bank.c	バンクに関する情報を取り扱う処理のソースファイル

表 8-19　ファームウェアアップデートプログラムで使用する標準インクルードファイル

ファイル名	概要
stdbool.h	論理型、および論理値に関するマクロを定義します。
stdint.h	指定した幅の整数型を宣言してマクロを定義します。
stdlib.h	記憶領域管理等の C プログラムで標準的処理を行うライブラリです。
string.h	文字列の比較、複写等を行うライブラリです。

8.4.8.2　定数一覧

表 8-20　ファームウェアアップデートプログラムで使用する定数(r_fw_up_rx_if.h)

定数名	設定値	内容
FW_UP_BANK0	(0)	BANK0 を定義
FW_UP_BANK1	(1)	BANK1 を定義
FW_UP_CFG_PART_MEMORY_SIZE	BSP_CFG_MCU_PART_MEMORY_SIZE	BSP FIT モジュールで定義された値を参照、CPU のメモリーサイズを定義する。
FW_UP_DUAL_BANK_MODE	FLASH_IN_DUAL_BANK_MODE	フラッシュ FIT モジュールで定義された値を参照、デュアルモードかどうかを示す。デュアルモードのとき(1)

表 8-21　ファームウェアアップデートプログラムで使用する定数(r_fw_up_rx_private.h)

定数名	設定値	内容
FW_UP_BINARY_BUF_SIZE	(256u)	コードフラッシュメモリ書き込み用データのバッファサイズ
FW_UP_BINARY_BUF_NUM	(2u)	コードフラッシュメモリ書き込み用データのバッファ数
FW_UP_BUF_NUM	(60u)	解析したモトローラ S レコードフォーマットデータの内容を格納する配列の数
FW_UP_BLANK_VALUE	(0xFFFFFFFFu)	コードフラッシュメモリがブランク時の読み出し値

表 8-22　ファームウェアアップデートプログラムで使用する定数(r_fw_up_buf.h)

定数名	設定値	内容
MOT_S_CHECK_SUM_FIELD	(0x02)	モトローラ S レコードフォーマットのチェックサムフィールドの文字数
ADDRESS_LENGTH_S1	(0x04)	モトローラ S レコードフォーマットのアドレスフィールドの文字数(S1 タイプ)
ADDRESS_LENGTH_S2	(0x06)	モトローラ S レコードフォーマットのアドレスフィールドの文字数(S2 タイプ)
ADDRESS_LENGTH_S3	(0x08)	モトローラ S レコードフォーマットのアドレスフィールドの文字数(S3 タイプ)
BUF_LOCK	(1)	指定したモトローラ S レコードフォーマットのバッファはロックされている。
BUF_UNLOCK	(0)	指定したモトローラ S レコードフォーマットのバッファは開放されている。

8.4.8.3 型定義一覧

```
typedef enum e_fw_up_return_t
{
    FW_UP_SUCCESS,
    FW_UP_ERR_OPENED,
    FW_UP_ERR_NOT_OPEN,
    FW_UP_ERR_NULL_PTR,
    FW_UP_ERR_INVALID_RECORD,
    FW_UP_ERR_BUF_FULL,
    FW_UP_ERR_BUF_EMPTY,
    FW_UP_ERR_INITIALIZE,
    FW_UP_ERR_ERASE,
    FW_UP_ERR_WRITE,
    FW_UP_ERR_INTERNAL,
} fw_up_return_t;

typedef struct st_fw_up_fl_data_t
{
    uint32_t src_addr;
    uint32_t dst_addr;
    uint32_t len;
    uint16_t count;
} fw_up_fl_data_t;

typedef struct st_fw_up_bank_t
{
    uint32_t low_addr;
    uint32_t high_addr;
    uint32_t start_block;
    uint32_t revno;
    uint32_t blockno;
} fw_up_bank_t;

typedef struct st_fw_up_bankinfo_t
{
    uint16_t active;
    uint16_t writable;
} fw_up_bankinfo_t;
```

図 8-9 ファームウェアアップデートプログラムで使用する型定義(r_fw_up_rx_if.h)

```
typedef enum fw_up_mot_s_cnt_t
{
    STATE_MOT_S_RECORD_MARK = 0,
    STATE_MOT_S_RECORD_TYPE,
    STATE_MOT_S_LENGTH_1,
    STATE_MOT_S_LENGTH_2,
    STATE_MOT_S_ADDRESS,
    STATE_MOT_S_DATA,
    STATE_MOT_S_CHKSUM_1,
    STATE_MOT_S_CHKSUM_2
} fw_up_mot_s_cnt_t;

typedef struct MotSBufS
{
    uint8_t addr_length;
    uint8_t data_length;
    uint8_t *paddress;
    uint8_t *pdata;
    uint8_t type;
    uint8_t act;
    struct MotSBufS *pnext;
} fw_up_mot_s_buf_t;

typedef struct WriteDataS
{
    uint32_t addr;
    uint32_t len;
    uint8_t data[FW_UP_BINARY_BUF_SIZE];
    struct WriteDataS *pnext;
    struct WriteDataS *pprev;
} fw_up_write_data_t;
```

図 8-10　ファームウェアアップデートプログラムで使用する型定義(r_fw_up_buf.h)

8.4.8.4 変数一覧

表 8-23 ファームウェアアップデートプログラムで使用する static 型変数(r_fw_up_rx.c)

型	変数名	内容	使用関数
static bool	is_opend	ファームウェアアップデート初期設定完了フラグ	fw_up_open fw_up_close write_firmware fw_up_put_data fw_up_get_data

表 8-24 ファームウェアアップデートプログラムで使用する static 型変数(r_fw_up_buf.c)

型	変数名	内容	使用関数
static fw_up_mot_s_buf_t	*papp_put_mot_s_buf	モトローラ S フォーマット解析処理で現在使用しているモトローラ S レコードデータバッファへのポインタ	fw_up_buf_init fw_up_put_mot_s
static fw_up_mot_s_buf_t	*papp_get_mot_s_buf	コードフラッシュメモリ書き込み用データ作成処理で現在使用しているモトローラ S レコードデータバッファへのポインタ	fw_up_buf_init fw_up_get_binary
static fw_up_mot_s_buf_t	mot_s_buf[FW_UP_BUF_NUM]	モトローラ S レコードフォーマットデータの内容を格納するバッファ	fw_up_buf_init fw_up_memory_init
static fw_up_write_data_t	*papp_write_buf	現在使用しているコードフラッシュメモリ書き込み用データバッファへのポインタ	fw_up_buf_init fw_up_get_binary
static fw_up_write_data_t	write_buf[FW_UP_BINARY_BUF_NUM]	コードフラッシュメモリ書き込み用データを格納するバッファ	fw_up_buf_init
static fw_up_mot_s_cnt_t	mot_s_data_state	モトローラ S レコードフォーマットデータの解析状態	fw_up_buf_init fw_up_put_mot_s
static uint32_t	write_current_address	現在のコードフラッシュメモリ書き込み先アドレス	fw_up_buf_init fw_up_get_binary
static bool	detect_terminal_flag	終端レコード検出フラグ	fw_up_buf_init fw_up_put_mot_s fw_up_get_binary

表 8-25　ファームウェアアップデートプログラムで使用するバンク情報に関する変数(r_fw_up_bank.c)

型	変数名	内容	使用関数
fw_up_bank_t	Bank[2]	BANK0、BANK1 についてアドレスの上限、下限、開始ブロックアドレス、ファームウェアリビジョン、総ブロック数を示す。	main(ブートローダー) fw_up_bank_initial fw_up_check_addr_value fw_up_bank_revno_update BL_Data BL_Reboot
fw_up_bank_t	BankInfo	プログラムを実行しているバンクと書き込み可能なバンクを示す。	main(ユーザープログラム) erase_another_bank analyze_and_write_data BL_Data

8.4.8.5　関数一覧

表 8-26　　ファームウェアアップデートプログラムで使用する関数

関数名	概要	記載ファイル
fw_up_open_flash	フラッシュ FIT モジュールの初期化	r_fw_up_rx.c
fw_up_open	ファームウェアアップデートの初期化	r_fw_up_rx.c
fw_up_close	ファームウェアアップデートの終了処理	r_fw_up_rx.c
erase_another_bank	バンクのコードフラッシュメモリ消去	r_fw_up_rx.c
analyze_and_write_data	受信データ解析とコードフラッシュメモリ書き込み処理	r_fw_up_rx.c
bank_toggle	起動バンク切り替え	r_fw_up_rx.c
fw_up_soft_reset	ソフトウェアリセット実行	r_fw_up_rx.c
write_firmware	コードフラッシュメモリ書き込み	r_fw_up_rx.c
fw_up_put_data	受信データ解析	r_fw_up_rx.c
fw_up_get_data	コードフラッシュメモリ書き込みデータ取得	r_fw_up_rx.c
fw_up_buf_init	ファームウェアアップデートで使用するバッファの初期化	r_fw_up_buf.c
fw_up_memory_init	バッファへのポインタの初期化	r_fw_up_buf.c
fw_up_put_mot_s	モトローラ S レコードフォーマットデータの解析	r_fw_up_buf.c
fw_up_get_binary	コードフラッシュメモリ書き込みデータの取得	r_fw_up_buf.c
fw_up_ascii_to_hexbyte	アスキー形式データからバイナリ形式データへの変換	r_fw_up_buf.c
fw_up_bank_initial	バンク情報の初期値設定	r_fw_up_bank.c
fw_up_check_addr_value	指定アドレスがバンク範囲内かチェックする	r_fw_up_bank.c
fw_up_bank_revno_update	バンク情報のリビジョンを現在の値に更新する	r_fw_up_bank.c

8.4.9　FoE プログラム詳細

8.4.9.1　ファイル構成

表 8-27　FoE プログラムで使用するファイル

ファイル名	概要
foesample.c	FoE サービスを含むアプリ層処理とリニアモードリブート処理のソースファイル
foesample.h	FoE サービスを含むアプリ層処理とリニアモードリブート処理のヘッダファイル
bootmode.c	ファームウェア更新およびデュアルモードリブート処理のソースファイル
bootmode.h	ファームウェア更新およびデュアルモードリブート処理のヘッダファイル

表 8-28　FoE プログラムで使用する標準インクルードファイル

ファイル名	概要
stdio.h	入出力に関する関数とマクロを定義します。
stdint.h	指定した幅の整数型を宣言してマクロを定義します。

8.4.9.2　定数一覧

表 8-29　FoE プログラムで使用する定数(foesample.h)

定数名	設定値	内容
SII_EEP_IDENTIFY_OFFSET	0x08	SII EEPROM に格納されている Identify の開始ワードアドレス
SII_EEP_VENDORID	0x00	SII EEPROM に格納されているベンダーID のオフセットワードアドレス
SII_EEP_PRODUCTCODE	0x02	SII EEPROM に格納されているプロダクトコードのオフセットワードアドレス
SII_EEP_REVESIONNO	0x04	SII EEPROM に格納されているリビジョンのオフセットワードアドレス
SII_EEP_SERIALNO	0x04	SII EEPROM に格納されているシリアルナンバーのオフセットワードアドレス

表 8-30　FoE プログラムで使用する定数(bootmode.c)

定数名	設定値	内容
BL_DATA_STATUS_IDLE	(0)	ファイルデータ受信処理ステータスが IDLE であることを示す
BL_DATA_STATUS_ERASE_START	(1)	ファイルデータ受信処理ステータスがコードフラッシュの消去開始状態であることを示す
BL_DATA_STATUS_ERASE	(2)	ファイルデータ受信処理ステータスがコードフラッシュの消去完了状態であることを示す
BL_DATA_STATUS_WRITE	(3)	ファイルデータ受信処理ステータスがコードフラッシュの書き込み中であることを示す

8.4.9.3 型定義一覧

```
typedef union
{
    UINT32      dword[4];
    UINT16      word[8];
    UINT8       byte[16];
}EEPBUFFER;
```

図 8-11 FoE プログラムで使用する型定義(foesample.h)

8.4.9.4 変数一覧

表 8-31 FoE プログラムで使用する static 型変数(bootmode.c)

型	変数名	内容	使用関数
static UINT8	DataStatus	ファイルデータ受信処理ステータス	BL_StartDownload BL_Data
static BOOL	bReboot	リブートフラグ。リブートを行うときに 1 にセットされる。	BL_SetRebootFlag BL_CheckRebootFlag

表 8-32 FoE プログラムで使用する const 型変数(bootmode.c)

型	変数名	内容	使用関数
const UINT32	tbl_identify[4]	(VENDOR_ID), (PRODUCT_CODE), (REVISION_NUMBER), (SERIAL_NUMBER)	AL_ControlInd

表 8-33 FoE プログラムで使用する const 型変数(foesample.c)

型	変数名	内容	使用関数
const UINT16	aFileDownloadHeader[5]	ダウンロードファイル接頭辞文字列 バンク 0："ECATFW__B0" バンク 1："ECATFW__B1"	FoE_Write FoE_Read
const UINT32	aFilePassword	ダウンロードファイルパスワード 数字 8 桁 初期値：0x00000000	FoE_Write FoE_Read

8.4.9.5 関数一覧

表 8-34 FoE プログラムで使用する関数

関数名	概要	記載ファイル
BL_Start	INIT→BOOT 遷移時に実行される処理	bootmode.c
BL_Stop	BOOT→INIT 遷移時に実行される処理	bootmode.c
BL_StartDownload	ファイルダウンロード開始処理	bootmode.c
BL_Data	ファイルデータ受信処理。 コードフラッシュのイレーズと書き込みを行う。	bootmode.c
BL_SetRebootFlag	リブートフラグのセット	bootmode.c
BL_CheckRebootFlag	リブートフラグのチェック	bootmode.c
BL_Reboot	デュアルモードのリブート処理	bootmode.c
FoE_Read	FoE リードリクエスト受信処理	foesample.c
FoE_ReadData	FoE ファイルデータ受信処理	foesample.c
FoE_WriteData	FoE ライトリクエスト受信処理	foesample.c
FoE_Write	FoE ファイルデータ送信処理	foesample.c
HW_Reboot	リニアモードのリブート処理	foesample.c

8.5 オブジェクトディクショナリ

本サンプルプログラムで定義しているオブジェクトディクショナリを表 8-35 に示します。

表 8-35 オブジェクトディクショナリ

Index	ObjectCode	SI	DataType	Access	Object Name
0x1nnn Communication Area					
0x1000	VAR	-	UDINT	RO	Device Name
0x1001	VAR	-	USINT	RO	Error Register
0x1008	VAR	-	STRING(32)	RO	Manufacture Device Name
0x1009	VAR	-	STRING(3)	RO	Manufacturer Hardware Version
0x100A	VAR	-	STRING(3)	RO	Manufacturer Software Version
0x100B	VAR	-	STRING(3)	RO	Manufacturer Bootloader Version
0x1010	ARRAY	0	USINT	RO	Store parameters
		1	UDINT	RW	SubIndex 00x
0x1011	ARRAY	0	USINT	RO	Restore default parameters
		1	UDINT	RW	SubIndex 00x
0x1018	RECORD	0	USINT	RO	Identity Object
		1	UDINT	RO	Vendor ID
		2	UDINT	RO	Product Code
		3	UDINT	RO	Revision Number
		4	UDINT	RO	Serial Number
0x10F0	RECORD	0	USINT	RO	Backup parameter handling
		1	UDINT	RO	Checksum
		2	BOOL	RW	Backup Parameter Changed
0x10F1	RECORD	0	USINT	RO	Error Settings
		1	UDINT	RO	Local Error Reaction
		2	UINT	RW	Sync Error Counter Limit
0x10F8	VAR	-	ULINT	RO	Timestamp Object
0x1600	RECORD	0	USINT	RO	csp/csv RxPDO Mapping
		1-5	UDINT	RO	SubIndex 00x
0x1601	RECORD	0	USINT	RO	csp RxPDO Mapping
		1-3	UDINT	RO	SubIndex 00x
0x1602	RECORD	0	USINT	RO	csv RxPDO Mapping
		1-3	UDINT	RO	SubIndex 00x
0x17FF	RECORD	0	USINT	RW	Device User RxPDO-Map
		1	UDINT	RW	SubIndex 00x
0x1A00	RECORD	0	USINT	RO	csp/csv TxPDO Mapping
		1-5	UDINT	RO	SubIndex 00x

Index	ObjectCode	SI	DataType	Access	Object Name
0x1A01	RECORD	0	USINT	RO	csp TxPDO Mapping
		1-3	UDINT	RO	SubIndex 00x
0x1A02	RECORD	0	USINT	RO	csv TxPDO Mapping
		1-4	UDINT	RO	SubIndex 00x
0x1BFF	RECORD	0	USINT	RW	Device User TxPDO-Map
		1	UDINT	RW	SubIndex 00x
0x1C00	ARRAY	0	USINT	RO	Sync Manager Communication Type
		1-4	USINT	RO	SubIndex 00x
0x1C12	ARRAY	0	USINT	RO *	RxPDO Assign
		1-2	UINT	RO *	SubIndex 00x
0x1C13	ARRAY	0	USINT	RO *	TxPDO Assign
		1-2	UINT	RO *	SubIndex 00x
0x1C32	RECORD	0	USINT	RO	SM output parameter
		1	UINT	RO *	Synchronization Type
		2	UDINT	RO	Cycle Time
		4	UINT	RO	Synchronization Types supported
		5	UDINT	RO	Minimum Cycle Time
		6	UDINT	RO	Calc and Copy Time
		8	UINT	RW	Get Cycle Time
		9	UDINT	RO	Delay Time
		10	UDINT	RW	Sync0 Cycle Time
		11	UINT	RO	SM-Event Missed
		12	UINT	RO	Cycle Time Too Small
		13	UINT	RO	Shift Time Too Short Counter
		32	BOOL	RO	Sync Error
0x1C33	RECORD	0	USINT	RO	SM input parameter
		1	UINT	RO *	Synchronization Type
		2	UDINT	RO	Cycle Time
		4	UINT	RO	Synchronization Types supported
		5	UDINT	RO	Minimum Cycle Time
		6	UDINT	RO	Calc and Copy Time
		8	UINT	RW	Get Cycle Time
		9	UDINT	RO	Delay Time
		10	UDINT	RW	Sync0 Cycle Time
		11	UINT	RO	SM-Event Missed
		12	UINT	RO	Cycle Time Too Small
		13	UINT	RO	Shift Time Too Short Counter
		32	BOOL	RO	Sync Error

Index	ObjectCode	SI	DataType	Access	Object Name
0x5nnn Manufacturer Area					
0x5000	VAR	-	USINT	RO	Firmware Writable Bank
0x6nnn CiA402 Drive Profile Area					
0x603F	VAR	-	UINT	RO	Error Code
0x6040	VAR	-	UINT	RW	Control Word
0x6041	VAR	-	UINT	RO	Status Word
0x605A	VAR	-	INT	RW	Quick stop option code
0x605B	VAR	-	INT	RW	Shutdown option code
0x605C	VAR	-	INT	RW	Disable operation option code
0x605D	VAR	-	INT	RW	Halt option code
0x605E	VAR	-	INT	RW	Fault reaction option code
0x6060	VAR	-	SINT	RW	Modes of Operation
0x6061	VAR	-	SINT	RO	Modes of operation display
0x6062	VAR	-	DINT	RO	Position demand value
0x6063	VAR	-	DINT	RO	Position actual internal value
0x6064	VAR	-	DINT	RO	Position actual value
0x6065	VAR	-	UDINT	RW	Following error window
0x6066	VAR	-	UINT	RW	Following error time out
0x6067	VAR	-	UDINT	RW	Position window
0x606C	VAR	-	DINT	RO	Velocity actual value
0x6072	VAR	-	UINT	RW	Max Torque
0x6077	VAR	-	INT	RO	Torque actual value
0x607A	VAR	-	DINT	RW	Target position
0x607B	RECORD	0	USINT	RO	Position range limit
		1	DINT	RW	Min position range limit
		2	DINT	RW	Max position range limit
0x607C	VAR	-	DINT	RW	Home Offset
0x607D	RECORD	0	USINT	RO	Software position limit
		1	DINT	RW	Min position limit
		2	DINT	RW	Max position limit
0x6091	RECORD	0	USINT	RO	Gear ratio
		1	UDINT	RW	Motor revolutions
		2	UDINT	RW	Shaft revolutions
0x6099	RECORD	0	USINT	RO	Homing speeds
		1	UDINT	RW	Speed during search for switch
		2	UDINT	RW	Speed during search for zero

Index	ObjectCode	SI	DataType	Access	Object Name
0x60B0	VAR	-	DINT	RW	Position offset
0x60B1	VAR	-	DINT	RW	Velocity offset
0x60B2	VAR	-	INT	RW	Torque offset
0x60B8	VAR	-	UINT	RW	Touch probe function
0x60B9	VAR	-	UINT	RO	Touch probe status
0x60BA	VAR	-	DINT	RO	Touch probe position 1 positive value
0x60BB	VAR	-	DINT	RO	Touch probe position 1 negative value
0x60C2	RECORD	0	USINT	RO	Interpolation time period
		1	USINT	RW	Interpolation time period value
		2	SINT	RW	Interpolation time index
0x60D0	RECORD	0	USINT	RO	Touch probe source
		1	INT	RW	Touch probe 1 source
0x60E0	VAR	-	UINT	RW	Positive torque limit value
0x60E1	VAR	-	UINT	RW	Negative torque limit value
0x60F4	VAR	-	DINT	RO	Following error actual value
0x60FD	VAR	-	UDINT	RO	Digital inputs
0x60FE	RECORD	0	USINT	RO	Digital outputs
		1	UDINT	RW	Physical outputs
		2	UDINT	RW	Bit mask
0x60FF	VAR	-	DINT	RW	Target velocity
0x6402	VAR	-	UINT	RW	Motor Type
0x6502	VAR	-	UDINT	RO	Supported drive modes
0xFnnn Common Device Profile Area					
0XF000	RECORD	0	USINT	RO	Semiconductor Device Profile
		1	UINT	RO	Index Distance
		2	UINT	RO	Maximum Number of Modules
0xF010	ARRAY	0	USINT	RO	Module Profile List
		1	UDINT	RO	SubIndex 00x
0xF020	ARRAY	0	USINT	RO	Configured Address List
		1	UDINT	RO	SubIndex 00x
0xF030	ARRAY	0	USINT	RO	Configured Module Ident List
		1	UDINT	RO	SubIndex 00x
0xF050	ARRAY	0	USINT	RO	Detected Module Ident List
		1	UDINT	RO	SubIndex 00x
0xF380	VAR	-	USINT	RO	Active Exception Status
0xF381	ARRAY	0	USINT	RO	Active Device Warming Details
		1	UDINT	RO	SubIndex 00x

Index	ObjectCode	SI	DataType	Access	Object Name
0xF382	ARRAY	0	USINT	RO	Active Manufacture Warming Details
		1	UDINT	RO	SubIndex 00x
0xF383	ARRAY	0	USINT	RO	Active Device Error Details
		1	UDINT	RO	SubIndex 00x
0xF384	ARRAY	0	USINT	RO	Active Manufacture Error Details
		1	UDINT	RO	SubIndex 00x
0xF385	RECORD	0	USINT	RO	Active Global Device Warming Details
		1	UDINT	RO	SubIndex 001
0xF386	RECORD	0	USINT	RO	Active Global Manufacture Warming Details
		1	UDINT	RO	SubIndex 001
0xF387	RECORD	0	USINT	RO	Active Global Device Error Details
		1	UDINT	RO	SubIndex 001
0xF388	RECORD	0	USINT	RO	Active Global Manufacture Error Details
		1	UDINT	RO	SubIndex 001
0xF390	VAR	-	USINT	RO	Latched Exception Status
0xF391	ARRAY	0	USINT	RO	Latched Device Warming Details
		1	UDINT	RO	SubIndex 00x
0xF392	ARRAY	0	USINT	RO	Latched Manufacture Warming Details
		1	UDINT	RO	SubIndex 00x
0xF393	ARRAY	0	USINT	RO	Latched Device Error Details
		1	UDINT	RO	SubIndex 00x
0xF394	ARRAY	0	USINT	RO	Latched Manufacture Error Details
		1	UDINT	RO	SubIndex 00x
0xF395	RECORD	0	USINT	RO	Latched Global Device Warming Details
		1	UDINT	RO	SubIndex 001
0xF396	RECORD	0	USINT	RO	Latched Global Manufacture Warming Details
		1	UDINT	RO	SubIndex 001
0xF397	RECORD	0	USINT	RO	Latched Global Device Error Details
		1	UDINT	RO	SubIndex 001
0xF398	RECORD	0	USINT	RO	Latched Global Manufacture Error Details
		1	UDINT	RO	SubIndex 001
0xF3A1	ARRAY	0	USINT	RO	Device Warming Mask
		1	UDINT	RW	SubIndex 00x
0xF3A2	ARRAY	0	USINT	RO	Manufacture Warming Mask
		1	UDINT	RW	SubIndex 00x
0xF3A3	ARRAY	0	USINT	RO	Device Error Mask
		1	UDINT	RW	SubIndex 00x

Index	ObjectCode	SI	DataType	Access	Object Name
0xF3A4	ARRAY	0	USINT	RO	Manufacture Error Mask
		1	UDINT	RW	SubIndex 00x
0xF3A5	RECORD	0	USINT	RO	Global Device Warming Mask
		1	UDINT	RW	SubIndex 001
0xF3A6	RECORD	0	USINT	RO	Global Manufacture Warming Mask
		1	UDINT	RW	SubIndex 001
0xF3A7	RECORD	0	USINT	RO	Global Device Error Mask
		1	UDINT	RW	SubIndex 001
0xF3A8	RECORD	0	USINT	RO	Global Manufacture Error Mask
		1	UDINT	RW	SubIndex 001
0xF6F0	ARRAY	0	USINT	RO	Input Latch Local Timestamp
		1	UDINT	RO	SubIndex 00x
0xF6F1	ARRAY	0	USINT	RO	Input Latch ESC Timestamp (32-bit)
		1	UDINT	RO	SubIndex 00x
0xF6F2	ARRAY	0	USINT	RO	Input Latch ESC Timestamp (64-bit)
		1	ULINT	RO	SubIndex 00x
0xF9F0	VAR	-	STRING(8)	RO	Manufacturer Serial Number
0xF9F1	ARRAY	0	USINT	RO	CDP Function Generation Number
		1	UDINT	RO	SubIndex 00x
0xF9F2	ARRAY	0	USINT	RO	SDP Function Generation Number
		1	UDINT	RO	SubIndex 00x
0xF9F3	VAR	-	STRING(7)	RO	Vendor Name
0xF9F4	ARRAY	0	USINT	RO	Semiconductor SDP Device Name
		1	STRING(8)	RO	SubIndex 00x
0xF9F5	ARRAY	0	USINT	RO	Output Identifier
		1	USINT	RW	SubIndex 00x
0xF9F6	VAR	-	UDINT	RO	Time since power on
0xF9F7	VAR	-	UDINT	RO	Total time powered
0xF9F8	VAR	-	UDINT	RO	Firmware Update Functional Generation Number
0xF9F9	ARRAY	0	USINT	RO	Module Manufacturer Hardware Version
		1	STRING (8)	RO	SubIndex 00x
0xF9FA	ARRAY	0	USINT	RO	Module Manufacturer Software Version
		1	STRING (8)	RO	SubIndex 00x
0xF9FB	ARRAY	0	USINT	RO	Module Manufacturer Serial Number
		1	STRING (8)	RO	SubIndex 00x

Index	ObjectCode	SI	DataType	Access	Object Name
0xFBF0	RECORD	0	USINT	RO	Device Reset Command
		1	ARRAY [0..5] OF BYTE	RW	Command
		2	USINT	RO	Status
		3	ARRAY [0..1] OF BYTE	RO	Response
0xFBF1	RECORD	0	USINT	RO	Exception Reset Command
		1	ARRAY [0..4] OF BYTE	RW	Command
		2	USINT	RO	Status
		3	ARRAY [0..1] OF BYTE	RO	Response
0xFBF2	RECORD	0	USINT	RO	Store Parameters Command
		1	ARRAY [0..3] OF BYTE	RW	Command
		2	USINT	RO	Status
		3	ARRAY [0..1] OF BYTE	RO	Response
0xFBF3	RECORD	0	USINT	RO	Calculate Checksum Command
		1	ARRAY [0..3] OF BYTE	RW	Command
		2	USINT	RO	Status
		3	ARRAY [0..7] OF BYTE	RO	Response
0xFBF4	RECORD	0	USINT	RO	Load Parameters Command
		1	ARRAY [0..3] OF BYTE	RW	Command
		2	USINT	RO	Status
		3	ARRAY [0..1] OF BYTE	RO	Response

* PreOP 状態でのみ RW になります。

※ 本サンプルプログラムをお客様の製品にてご利用される際は ETG 仕様書をご参照の上、各オブジェクトの設定や必要処理を別途検討・実装してください。

9. 付録 A. Semiconductor Device Profile [ETG.5003]

9.1 Common Device Profile [ETG.5003.1]

EtherCAT にて半導体デバイスを取り扱う場合、ETG5003 の仕様に規定されたデバイスプロファイルをサポートする必要があります。

ETG.5003 の構成は以下の内容となります。

1. Common Device Profile (CDP) [ETG.5003.1]
2. Firmware update functionality [ETG.5003.2]
3. Specific Device Profile (SDP) [ETG.5003.2xxx]

Common Device Profile (CDP) は、Specific Device Profile (SDP)で説明されているすべてのデバイスに適用される要件を指定します。

サンプルプログラムでは、CDP [ETG.5003.1 Ver1.1.0] Appendix A 相当のオブジェクトディクショナリ定義を提供します。CDP 関連の詳細について表 9-1 に示します。

表 9-1 本サンプルプログラムで提供する CDP 詳細

項目	詳細
CDP 関連オブジェクト定義ファイル	ヘッダファイル“semisample.h” ESI ファイル“Renesas EtherCAT RX72M.xml”
CDP 関連オブジェクト Index	0x10B0, x17FFF, 0x1BFF に加えて 0xF000 以降全て
CDP 関連オブジェクト定義を無効にしたい場合	semisample.h に定義されているマクロ“CDP_SAMPLE”を 0 にして、ESI ファイルの CDP 関連情報を削除して下さい。

サンプルプログラムでの提供はオブジェクトディクショナリ定義の枠組みのみとなります。設定や必要処理に関しては別途検討・実装してください。

Common Device Profile Ver1.1.0 については、下記の ETG.5003.1 規格書を参照ください。

また、CDP に関するご質問は、ETG 協会へお問い合わせください。

ETG5003.1 規格書：

ETG5003-1 S (R) V1.1.0

EtherCAT Semiconductor Device Profile

Part1 Common Device Profile

9.2　Semi Test Record [ETG.7000.2-Annex5003-0001]

本サンプルプログラムでは、下記の Semi Test Record ETG.7000.2-Annex5003-0001 のテスト項目に対応することを想定したプログラムを実装しています。

1. Device Reset Command (Standard reset)
2. Dynamic PDO
3. Store Parameters

9.2.1　Device Reset Command (Standard reset)

オブジェクト 0xFBF0 のサブインデックス 1 に特定の値が入力されると評価ボードが再起動します。

- 確認方法

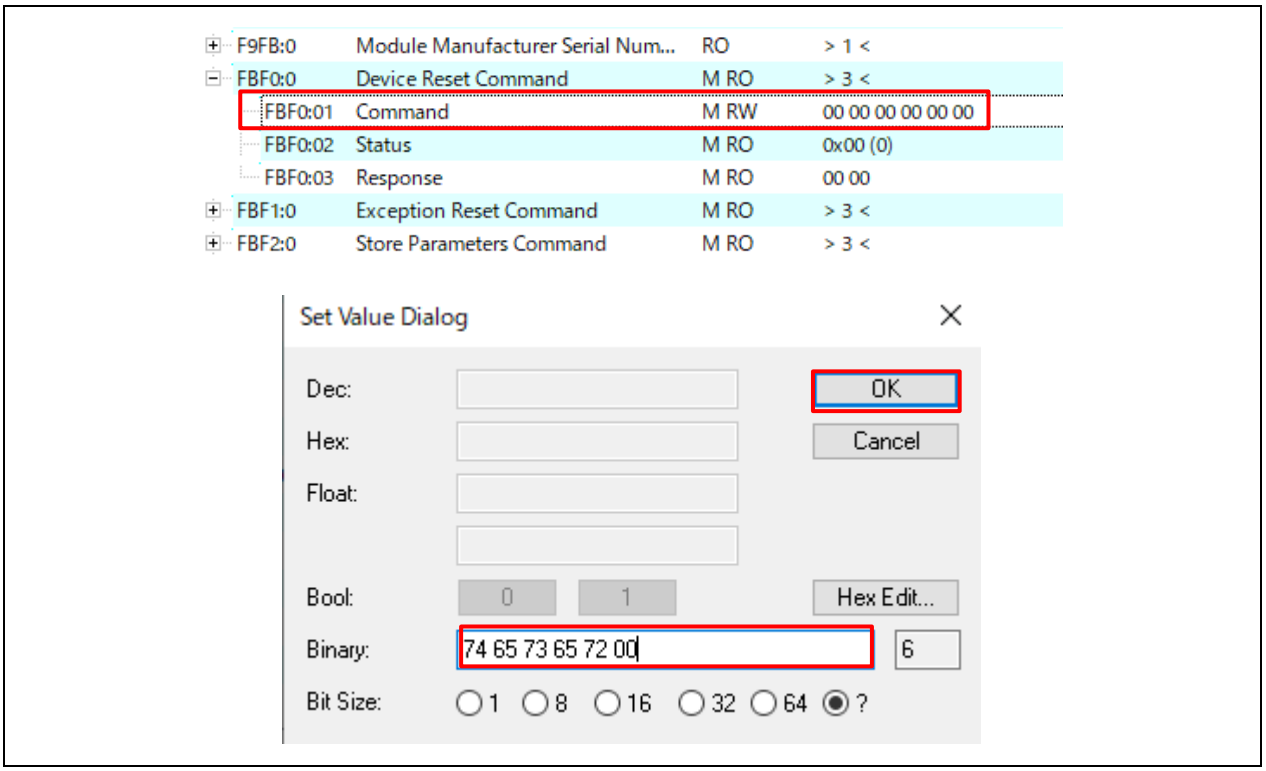
(1) 評価ボードのシリアルポートと PC のシリアルポートを接続し、PC 上で Tera Term などのターミナルソフトを起動します。ターミナルのシリアル設定では、115200bps、8 ビットデータ、パリティなし、1 ストップビット、フロー制御なしを設定します。

(2) 5.『TwinCAT との接続』の 5.6『デバイスの再スキャン』まで行い、TwinCAT と評価ボードを接続してください。

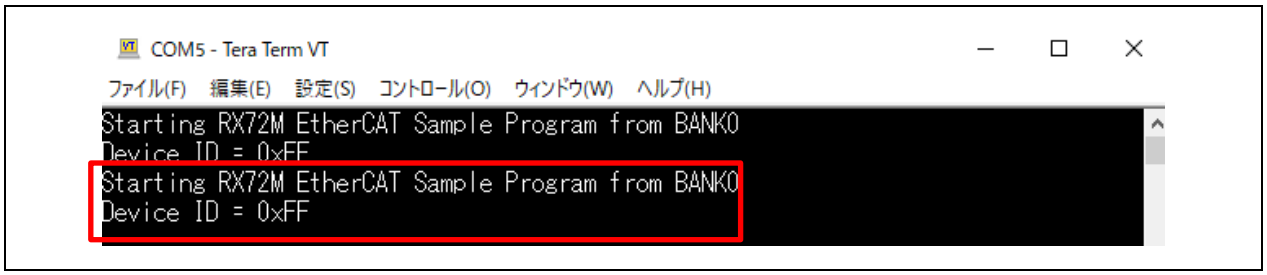
(3) “Online”タブを選択し、”Current Status”が”OP”になっていることを確認します。

(4) “CoE - Online”タブを選択し、Index FBF0:01 “Command”をダブルクリックして”binary”に

”74 65 73 65 72 00”を書き込みます。



(5) 評価ボードの起動時にシリアル通信で出力される”Starting RX72M EtherCAT Sample Program from BANKx”が Tera Term などのターミナルソフトに表示されれば、正常に再起動されています。



9.2.2 Dynamic PDO

本サンプルプログラムでは、ESI ファイルで PDO に関する設定を表 9-2 のようにしています。

表 9-2　本サンプルプログラムの PDO に関する設定

PDO 項目	設定
PdoAssign	true
PdoConfig	true

また、PDO アサイン・マッピングオブジェクトの設定を表 9-3 のようにしています。

表 9-3　PDO アサイン・マッピングオブジェクトの設定

Object Name	Index	Access
RxPDO assign	0x1C12	PreOP 状態でのみ RW
TxPDO assign	0x1C13	PreOP 状態でのみ RW
Device User RxPDO-Map	0x17FF	PreOP 状態でのみ RW
Device User TxPDO-Map	0x1BFF	PreOP 状態でのみ RW

そのため、デフォルトではアサインされていない Object 0x17FF “Device User RxPDO-Map”、0x1BFF “Device User TxPDO-Map”を EtherCAT 設定ツール上でアサイン・マッピングすることができます。

- 確認方法

(1) 5.『TwinCAT との接続』の 5.6『デバイスの再スキャン』まで行い、TwinCAT と評価ボードを接続してください。

(2) “Slots”タブを選択し、現在の Slot を確認してください。

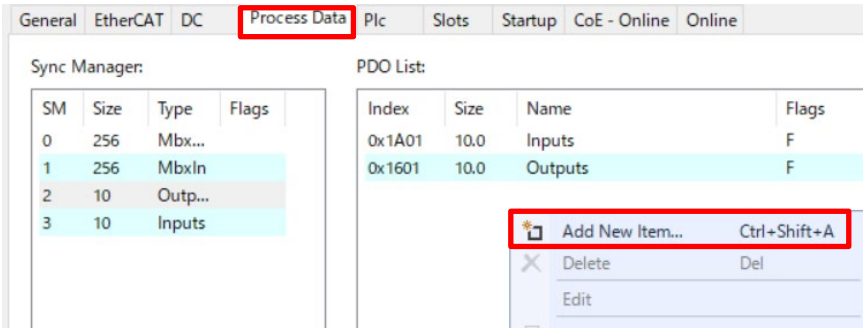
左枠の“Slot”の”Axis 0”に”csp-axis”を選択して追加・変更してください。

また、”Axis 1”は空にしてください。

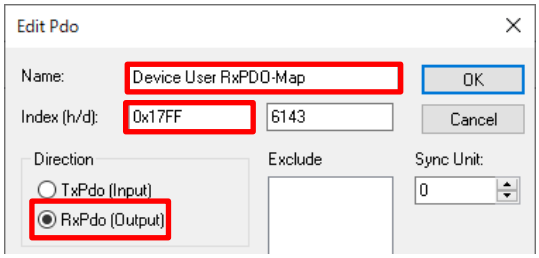
“Slot”に Module を追加・変更した場合は、上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。

(3) “Online” タブを選択し、”Current Status”が”OP”になっていることを確認します。

(4) “Process Data” タブを選択し、”PDO List”エリアで右クリックして“Add New Item…”を選択します。



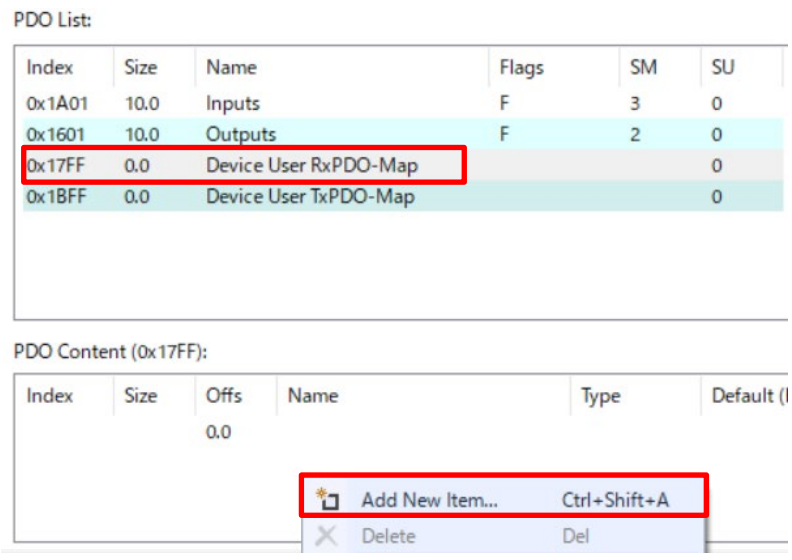
(5) “Edit Pdo” ウインドウ内で、名前を”Device User RxPDO-Map”、Index を”0x17FF”、Direction を”RxPdo”に設定して”OK”を押します。



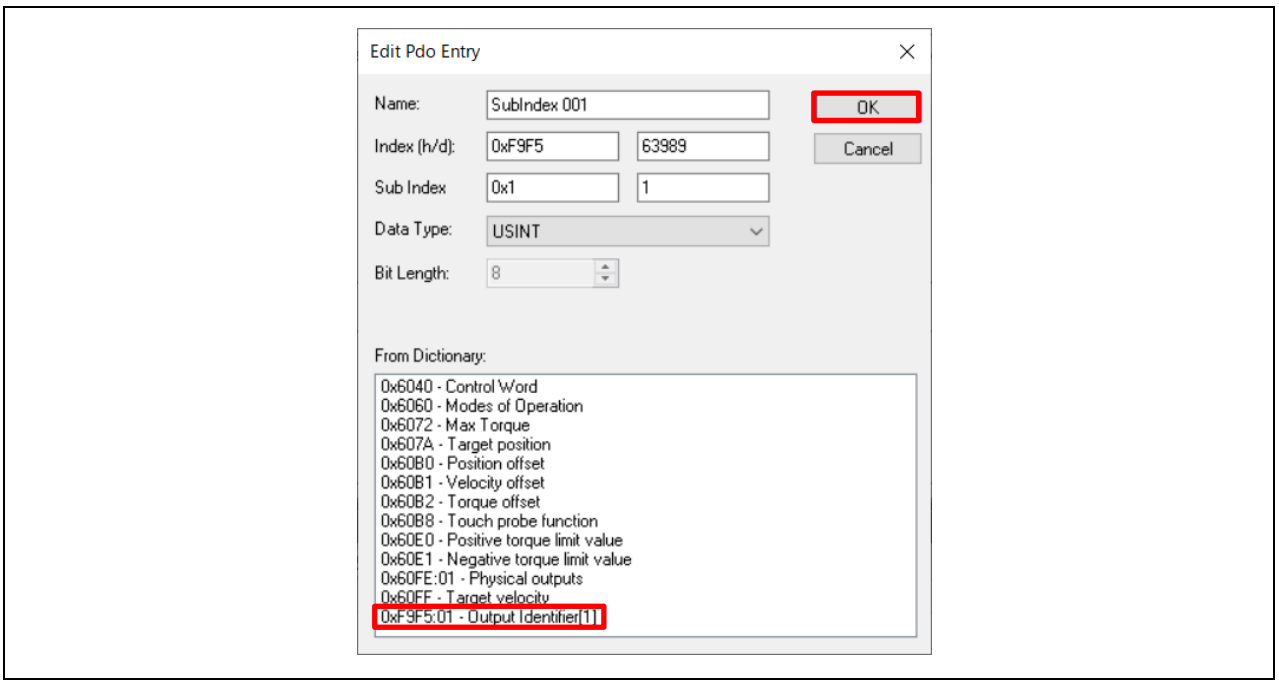
(6) 再び”PDO List”エリアで右クリックして“Add New Item…”を選択します。“Edit Pdo”ウインドウ内で名前を”Device User TxPDO-Map”、Index を”0x1BFF”、Direction を”TxPdo”に設定して”OK”を押します。

これで、”PDO List”に”Device User RxPDO-Map”と”Device User TxPDO-Map”が追加されました。

(7) ”PDO List”の”Device User RxPDO-Map”をクリックし、”PDO Content (0x17FF):”エリアで右クリックして“Add New Item…”を選択してください。



(8) “Edit Pdo Entry”ウインドウ内で、”From Dictionary”から”0xF9F5:01 – Output Identifier[1]”をクリックし、OK を押します。これで、Index 0x17FF に Index 0xF9F5 がマッピングされました。

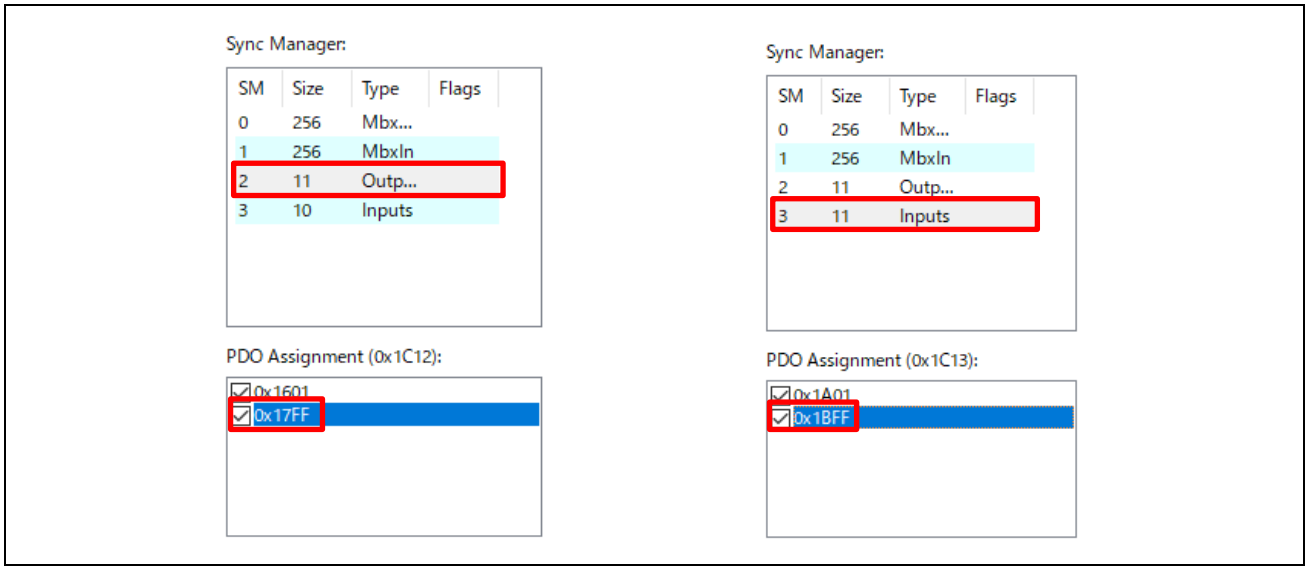


(9) 同様に、”PDO List”の”Device User TxPDO-Map”をクリックし、”PDO Content (0x1BFF):”エリアで右クリックして“Add New Item…”を選択します。

“Edit Pdo Entry”ウインドウ内で、”From Dictionary”から”0xF9F5:01 – Output Identifier[1]”をクリックし、OK を押します。これで、Index 0x1BFF に Index 0xF9F5 がマッピングされました。

(10) ”Sync Manager”エリアで”SM”列が 2 の行をクリックし、”PDO Assignment (0x1C12)”エリアで 0x17FF にチェックを入れます。

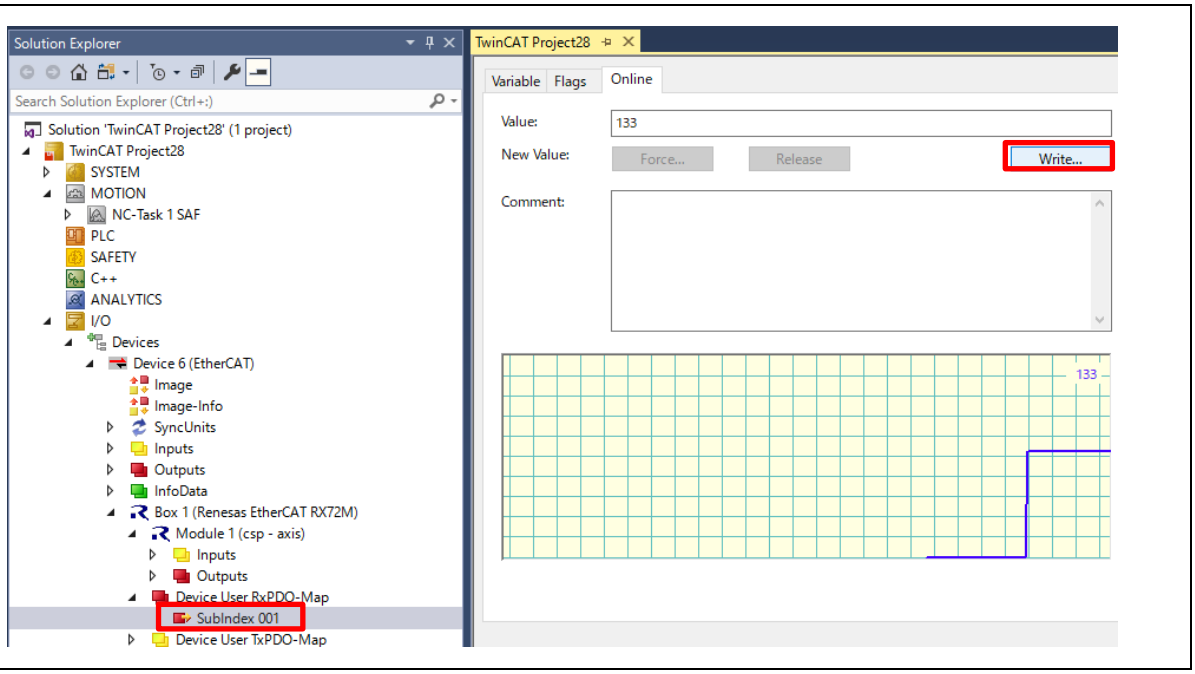
(11) 同様に、”Sync Manager”エリアで”SM”列が 3 の行をクリックし、”PDO Assignment (0x1C13)”エリアで 0x1BFF にチェックを入れます。これで、Index 0x1C12 に 0x17FF が、0x1C13 に 0x1BFF が追加でアサインされました。



(12) 上のメニューバーの[TwinCAT] → [Restart TwinCAT (Config Mode)]を押して TwinCAT の再起動を行ってください。

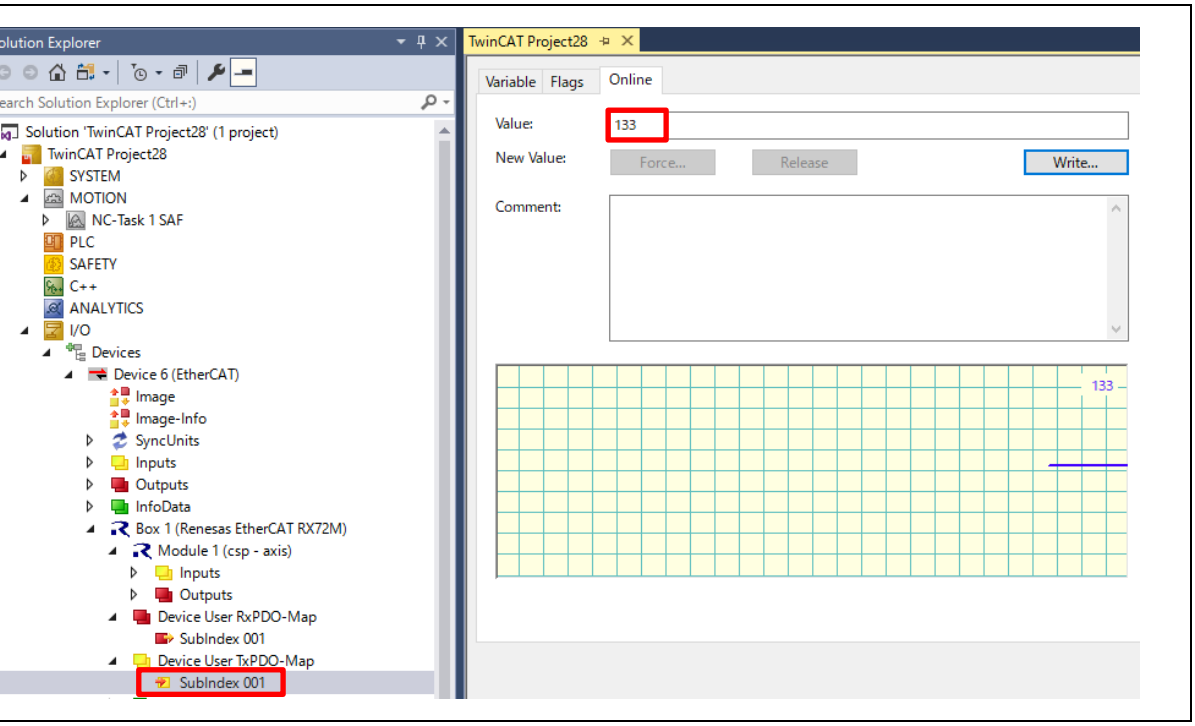
(13) [Box 1]の[Device User RxPDO-Map]を展開し、[SubIndex 001]をクリックしてください。

(14) Value に 1~255 の任意の値を書き込んでください。



(15) [Device User TxPDO-Map]を展開し、[SubIndex 001]をクリックしてください。

(16) Value の値が(13)で書き込んだ値と同じであれば、正常に設定できています。



9.2.3　Store Parameters

本サンプルプログラムでは、Backup フラグを持つオブジェクトの値が SDO 通信によって変更されると評価ボードの不揮発性メモリにその値が保存されます。

本サンプルプログラムで Backup フラグを持つオブジェクトを表 9-4 に示します。

表 9-4　Backup フラグを持つオブジェクト

Index	Name
0x10F0	Backup parameter handling
0xF3A1	Device Warming Mask
0xF3A2	Manufacture Warming Mask
0xF3A3	Device Error Mask
0xF3A4	Manufacture Error Mask
0xF3A5	Global Device Warming Mask
0xF3A6	Global Manufacture Warming Mask
0xF3A7	Global Device Error Mask
0xF3A8	Global Manufacture Error Mask

Backup フラグを持つオブジェクトの値が保存される不揮発性メモリ領域として、本サンプルプログラムでは表 9-5 に示すコードフラッシュメモリ領域を使用しています。

表 9-5　Backup オブジェクト用不揮発性メモリ領域

バンクモード	先頭アドレス	最後尾アドレス	容量	ブロック No.
リニア（4MB）	FFDF_8000H	FFDF_FFFFH	32KB	70 (32KB)
リニア（2MB）	FFEF_8000H	FFEF_FFFFH	32KB	38 (32KB)
デュアル（4MB）	FFC0_0000H	FFC0_8000H	32KB	139 (32KB)
デュアル（2MB）	FFD0_0000H	FFD0_8000H	32KB	107 (32KB)

不揮発性メモリに保存するプログラムで使用する定数を表 9-6 に示します。

表 9-6　不揮発性メモリに保存するプログラムで使用する定数 (semisample.h)

定数名	設定値	内容
BACKUP_MEMORY_START_ADDRESS	表 9-5 の先頭アドレス	Backup オブジェクト用不揮発性メモリ領域の先頭アドレス
BACKUP_BYTESIZE	FLASH_CF_MIN_PGM_SIZE	Backup オブジェクト用不揮発性メモリ領域の容量
Default_Data_Is_Not_Initialized	(0xFFFF)	Backup オブジェクト用不揮発性メモリ領域が初期化されていない。
Default_Data_Is_Initialized	(0x5555)	Backup オブジェクト用不揮発性メモリ領域が初期化されている。
NonVolatileWordOffset_0x10F0	(0x0001)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A1	(0x0008)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数

NonVolatileWordOffset_0xF3A2	(0x000c)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A3	(0x0010)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A4	(0x0014)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A5	(0x0018)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A6	(0x001c)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A7	(0x0020)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数
NonVolatileWordOffset_0xF3A8	(0x0024)	BACKUP_MEMORY_START_ADDRESS からのオフセットワード数

本サンプルプログラムでは、Application Note ET9300 (EtherCAT Slave Stack Code) の 6.2.4.1 『Backup Parameter Support』で定義されている関数を使用しています。

詳細は Application Note ET9300 (EtherCAT Slave Stack Code)と semisample.c を参照してください。

- 動作確認方法

(1) 評価ボードのシリアルポートと PC のシリアルポートを接続し、PC 上で Tera Term などのターミナルソフトを起動します。ターミナルのシリアル設定では、115200bps、8 ビットデータ、パリティなし、1 ストップビット、フロー制御なしを設定します。

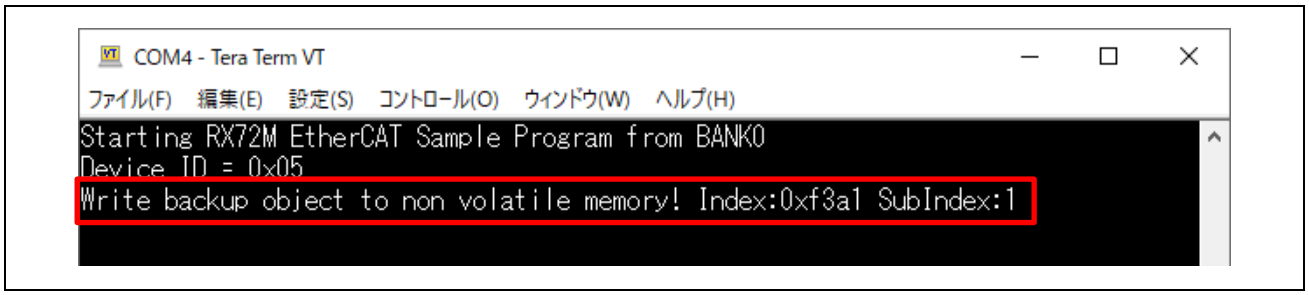
(2) 5.『TwinCAT との接続』の 5.6『デバイスの再スキャン』まで行い、TwinCAT と評価ボードを接続してください。

(3) “Online”タブを選択し、”Current Status”が”OP”になっていることを確認します。

(4) Backup オブジェクトに値を書き込みます。ここでは、Index 0xF3A1:01 に”0xAAAAAAAA”を書き込みます。

(5) 書き込みに成功すると、Tera Term などのターミナルソフトに

“Write backup object to non volatile memory! Index:0xf3a1 SubIndex:1” が表示されます。



(6) 9.2.1 『Device Reset Command』を実行して評価ボードを再起動して下さい。

(7) (4) で書き込んだ Index 0xF3A1:01 が ”0xAAAAAAAA” になっていれば正常です。Backup フラグを持つオブジェクトは不揮発性メモリに保存されており、起動時に読み出されています。

10. 付録 B. コードフラッシュメモリ容量 2MB に対応するには

本アプリケーションノートに含まれるプロジェクトはコードフラッシュメモリ容量 4MB の製品のものとなります。

コードフラッシュメモリ容量 2MB の製品に対応させるためには BSP のコンフィギュレーションファイルの変更とビルド構成のセクション設定の一部を変更する必要があります。

BSP のコンフィギュレーションファイルの変更点は下記になります。

【 src/smc_gen/r_config/r_bsp_config.h 】

BSP_CFG_MCU_PART_MEMORY_SIZE の値を 0xD に変更してください。

```
#define BSP_CFG_MCU_PART_MEMORY_SIZE        (0xD)
```

ビルド構成のセクション設定の変更点は次になります。

設定内容の変更は、プロジェクト→プロパティ→C/C++ビルド→設定にて行うことができます。

- リニアモード BANK0

表 10-1　BANK0 変更点 - ツール設定タブ

項目	変更内容	説明
Linker – セクション	ROM の開始位置を 0xFFF00000 に変更する。 ＜設定＞ 0xFFF00000 C_1 C_2 C C_8 C$* D* W* L P*	BANK0 のマッピングは 0xFFF00000 から 0xFFFF7FFF としています。サイズは 992KB です。

- リニアモード BANK1

表 10-2　BANK1 変更点 - ツール設定タブ

項目	変更内容	説明
Linker – セクション	ROM の開始位置を 0xFFE00000 に変更する。 ＜設定＞ 0xFFE00000 C_1 C_2 C C_8 C$* D* W* L P*	BANK1 のマッピングは 0xFFE00000 から 0xFFEF7FFF としています。サイズは 992KB です。

	ROM 領域に IDENTIFY セクションを追加し、開始位置を 0xFFEF7F70 とする。 ＜設定＞ 0xFFEF7F70 IDENTIFY 0xFFEF7F80 EXCEPTVECT 0xFFEF7FFC RESETVECT	

- デュアルモード HardwareDebug

表 10-3　HardwareDebug 変更点 - ツール設定タブ

項目	変更内容	説明
Linker – セクション	ROM 領域に配置される PResetPRG の開始位置を 0xFFF08000 に変更。 ＜設定＞ 0xFFF08000 PResetPRG C_1 C_2 C C_8 C$* D* W* L P PFRAM2	BANK0 のマッピングは 0xFFF00000 から 0xFFFFFFFF ですが、PResetPRG の開始位置を 0xFFF08000 と設定します。 BANK0 のサイズは 1024KB です。

- デュアルモード Download

表 10-4　Download 変更点 - ツール設定タブ

項目	変更内容	説明
Linker – セクション	ROM 領域に配置される PResetPRG の開始位置を 0xFFF08000 に変更。 ＜設定＞ 0xFFF08000 PResetPRG C_1 C_2 C C_8 C$* D* W* L P PFRAM2	BANK0 のマッピングは 0xFFF00000 から 0xFFFFFFFF ですが、PResetPRG の開始位置を 0xFFF08000 と設定します。 BANK0 のサイズは 1024KB です。

11. 付録 C. SSC Tool コンフィギュレーションファイルの使用方法

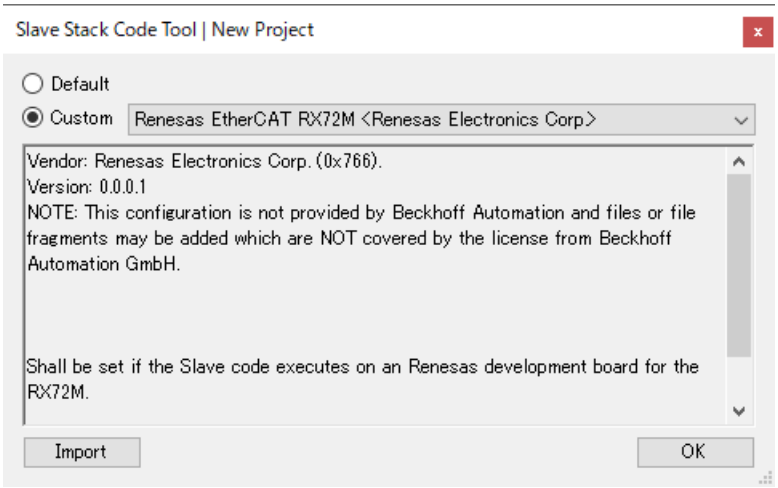
ここでは、SSC Tool コンフィギュレーションファイルを使用して SSC プロジェクトファイルを作成する方法について説明します。

(1) Windows スタートメニューから[EtherCAT Slave Stack Code] → [SSC Tool]をクリックして SSC Tool を起動します。

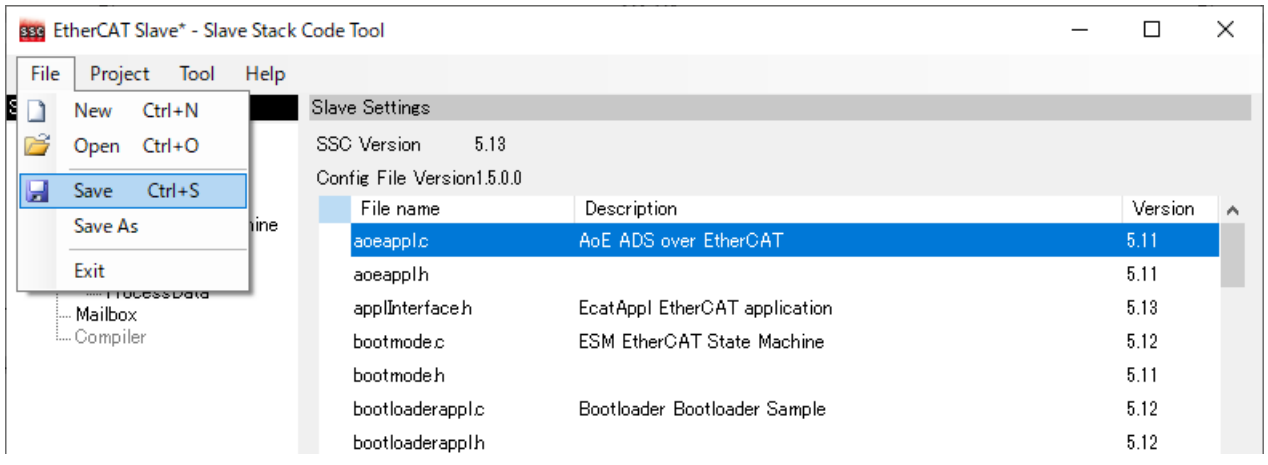
(2) [New Project]ダイアログで[Import]をクリックし、サンプルプログラムにある下記の SSC Tool コンフィギュレーションファイルを選択して[OK]をクリックします。

ecat_linear_demo_cpurx72m¥utilities¥ssc_config¥RX72M_EtherCAT_config.xml

(3) [Custom]チェックボックスをチェックしてから、リストから“Renesas EtherCAT RX72M <Renesas Electoronics Corp>”を選択し[OK]をクリックします。



(4) [File]→[Save]をクリックし、任意のプロジェクトファイル名で SSC プロジェクトファイルを保存することができます。
（デフォルトでは、“Renesas EtherCAT RX72M.esp”となります。）



12. 参考ドキュメント

ユーザーズマニュアル：

RX72M グループ ユーザーズマニュアル ハードウェア編（ドキュメント r01uh0804jj0120-rx72m.pdf）

RX72M CPU Card with RDC-IC ユーザーズマニュアル（ドキュメント r12uz0098jj0110-motor.pdf）

RX72M Renesas Starter Kit+ for RX72M　ユーザーズマニュアル

（ドキュメント REN_r20ut4391jg0100-rsk+rx72m-usermanual_MAT_20190731.pdf）

（最新版をルネサスエレクトロニクスホームページから入手してください。）

TS-RX72M-COM ユーザーズマニュアル

（最新版をテセラ・テクノロジー株式会社ホームページから入手してください。）

EtherCAT Slave Stack Code 参考資料：

Application Note ET9300 (EtherCAT Slave Stack Code)

EtherCAT 規格書：

ETG.1000.1 EtherCAT Specification – Part1 Overview

ETG.1000.2 EtherCAT Specification – Part2 Physical Layer service and protocol specification

ETG.1000.3 EtherCAT Specification – Part3 Data Link Layer service definition

ETG.1000.4 EtherCAT Specification – Part4 Data Link Layer protocol specification

ETG.1000.3 EtherCAT Specification – Part5 Application Layer service definition

ETG.1000.6 EtherCAT Specification – Part6 Application Layer protocol specification

ETG.1020 EtherCAT Protocol Enhancements

ETG.2000 EtherCAT Slave Information Specification

ETG.5003.1 EtherCAT Semiconductor Device Profile Part1 Common Device Profile

ETG.5003.2 EtherCAT Semiconductor Device Profile Part2 Firmware Update

ETG.6010 EtherCAT Implementation Directive for CiA402 Drive Profile

ETG.7000.Annex5003-0001 EtherCAT Semi Test Record

CiA402 規格書：

IEC 61800-7-201 Edition 1.0

Adjustable speed electrical power drive systems Part 7-201: Generic interface and use of profiles for power drive systems Profile type 1 specification

IEC 61800-7-301 Edition 1.0

Adjustable speed electrical power drive systems Part 7-301: Generic interface and use of profiles for power drive systems Mapping of profile type 1 to network technologies

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Mar.31.2024	—	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、フラッシュメモリー、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。お客様の機器・システムの設計において、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には、お客様の責任において行ってください。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、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、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。
5. 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。
6. 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。
7. 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。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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.4.0-1　2017.11)

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。