

RZ/T2N グループ

4 軸モータ制御サンプルプログラムスタートアップマニュアル(EtherCAT CiA402)

要旨

本書は、ルネサスエレクトロニクス製 RZ/T2N グループの MPU を搭載した RZ/T2N Evaluation Board による EtherCAT CiA402 を使用した 4 軸モータ制御サンプルプログラムのスタートアップマニュアルです。

動作確認デバイス

RZ/T2N グループ

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。

目次

1. 概要	5
1.1 4 軸モータ制御サンプルプログラム	5
1.2 ハードウェア構成	5
1.3 接続機器	6
1.3.1 RZ/T2N Evaluation Board と Inverter Board との接続	7
1.3.2 モータ/エンコーダ と Inverter Board との接続	8
1.3.3 PC との接続	8
1.3.4 ICE	8
1.3.5 電源	8
2. 動作環境	9
2.1 動作環境	9
2.2 フォルダ、ファイル構成	10
3. 関連ドキュメント	11
4. ボードの設定	12
4.1 RZ/T2N Evaluation Board の設定	12
4.1.1 Switches	12
4.1.2 Jumpers	13
4.2 Inverter Board	13
4.2.1 Jumpers	13
5. プログラム書き込み	14
5.1 EtherCAT SubDevice Stack の生成	14
5.2 プロジェクトのビルド	16
5.3 プロジェクトの書き込み	19
6. 実行方法	20
6.1 TwinCAT との接続	20
6.1.1 ESI ファイルの準備 (初回のみ)	20
6.1.2 TwinCAT ドライバのインストール (初回のみ)	20
6.1.3 デバイスのスキャン・EEPROM への書き込み	21
6.2 TwinCAT プロジェクトの実行方法	26
6.3 動作確認方法	33
6.3.1 CiA402 動作モードの選択	33
6.3.2 CiA402 State Machine 制御方法	34
6.3.2.1 Operation enable への遷移方法	34
6.3.2.2 Fault からの復帰方法	36
6.3.3 モータ動作方法	37
6.3.3.1 csp mode	37
6.3.3.2 pp mode	38
6.3.3.3 csv mode	40

6.3.3.4	hm mode.....	41
7.	ソフトウェア仕様.....	44
7.1	ソフトウェア構成図.....	44
7.2	Motor Control(RZ/T2N Cortex-R52 CPU0).....	45
7.2.1	ソフトウェア全体構成.....	45
7.2.2	ベースプロジェクトからの変更点.....	46
7.2.3	Network CPU I/F モジュール.....	50
7.2.3.1	モジュール構成.....	50
7.2.3.2	Shared Memory 割り当て.....	51
7.2.3.3	API.....	52
7.2.3.4	プライベート関数.....	58
7.2.3.5	Callback 関数.....	60
7.2.3.6	マクロ定義.....	61
7.2.3.7	列挙型.....	62
7.2.3.8	構造体・変数情報.....	63
7.2.3.9	グローバル変数情報.....	65
7.2.4	CPU0 CPU 間データ交換制御モジュール.....	66
7.2.4.1	モジュール構成.....	66
7.2.4.2	API.....	67
7.2.4.3	Callback 関数.....	67
7.2.4.4	構造体・変数情報.....	67
7.2.4.5	グローバル変数情報.....	68
7.3	EtherCAT(CiA402) (RZ/T2N Cortex-R52 CPU1).....	69
7.3.1	基本仕様.....	69
7.3.2	ソフトウェア全体構成.....	70
7.3.3	src フォルダ構成.....	71
7.3.4	hal_entry・メイン処理.....	72
7.3.5	CiA402 Drive Profile.....	75
7.3.5.1	CiA402 動作モード.....	75
7.3.5.2	Cyclic synchronous position mode.....	76
7.3.5.3	Cyclic synchronous velocity mode.....	77
7.3.5.4	Profile position mode.....	78
7.3.5.5	Homing mode.....	80
7.3.5.6	CiA402 状態遷移.....	82
7.3.5.7	CiA402 Error Code.....	83
7.3.5.8	CiA402 Drive Profile 向け関数.....	84
7.3.5.9	マクロ定義.....	97
7.3.5.10	SSC tool 生成後ファイルからの変更点.....	97
7.3.5.11	Object Dictionary.....	100
7.3.6	Motor CPU I/F.....	104
7.3.6.1	モジュール構成.....	104
7.3.6.2	Shared Memory 割り当て.....	104
7.3.6.3	API.....	105
7.3.6.4	プライベート関数.....	110
7.3.6.5	Callback 関数.....	112
7.3.6.6	マクロ定義.....	113

7.3.6.7	列挙型	113
7.3.6.8	構造体・変数情報	115
7.3.6.9	グローバル変数情報	118
7.3.7	CPU1 CPU 間データ交換制御モジュール	119
7.3.7.1	モジュール構成	119
7.3.7.2	CPU 間データ交換シーケンス	120
7.3.7.3	API	121
7.3.7.4	Callback 関数	122
7.3.7.5	マクロ定義	122
7.3.7.6	列挙型	122
7.3.7.7	構造体・変数情報	122
7.3.7.8	グローバル変数情報	123
8.	Appendix	124
8.1	統合開発環境インストール	124
8.1.1	e ² studio, FSP	124
8.2	付録 : SSC Tool 使用時の注意点	125
8.3	付録 : patch のインストール方法	127
8.3.1	Git for Windows (64bit) を使用する方法	127
8.3.2	MinGW Installation Manager を使用する方法	128
	改訂記録	130

1. 概要

1.1 4 軸モータ制御サンプルプログラム

モータ制御サンプルプログラムは RZT2N Evaluation Board に RZ/T series Inverter Board Kit(Inverter Board +BLDC)を接続する構成において、最大 4 軸のモータを制御することができます。

ソフトウェア機能を以下に示します。

- ・シリアルエンコーダを使用した永久磁石同期モータのベクトル制御
- ・EtherCAT SubDevice 通信
- ・CiA402 プロファイルを用いたモータ制御

Renesas Motor Workbench(RMW)を用いたモータ制御については、「RZ/T2N 永久磁石同期モータのエンコーダベクトル制御 - シリアルエンコーダ アプリケーションノート(R01AN8225)」を参照してください。

1.2 ハードウェア構成

図 1-1 にハードウェアの全体構成を示します。

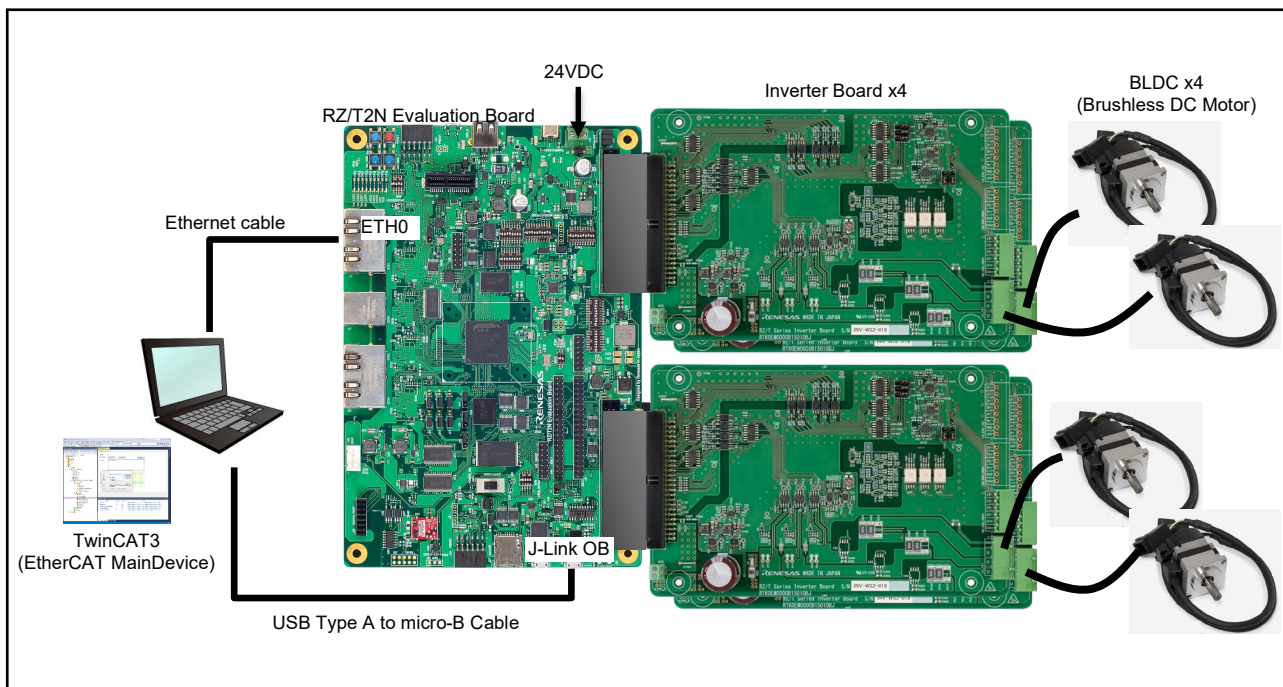


図 1-1 ハードウェア構成

1.3 接続機器

表 1.1 に接続機器一覧を示します。

表 1.1 接続機器一覧

Item	型名	備考
RZ/T2N Evaluation Board Kit	RTK9RZT2N1S1010BJ	https://www.renesas.com/rzt2n-evk
RZ/T Series Inverter Board Kit	RTK0EM0000S05010BJ (Renesas) - ボード型名: RTK0EM0000B15010BJ - モータ型名: TSM3101N2001E020 (多摩川精機社製)	下記より購入可能です https://www.renesas.com/invb-lv-rzt-b
フラットケーブル	MIL ソケットケーブル (50 ピン, ケーブル長: 10cm)	INV3/INV4 使用時に必要です
スペーサー	(1) 高さ: 10mm (2) 高さ: 7mm (3) 高さ: 30mm 以上	(1) INV1/INV2 使用時に必要です (2) INV1/INV2 使用時に必要です (3) INV3/INV4 使用時に必要です
電源	24V/8A(Max) DC, 5.08mm ピッチ 2 極端子台	安定化電源をご使用ください (4 軸/1000RPM 動作時の電流値: 1.1A)
ICE	SEGGER J-Link OB	USB Type A to micro-B ケーブルをご使用ください

1.3.1 RZ/T2N Evaluation Board と Inverter Board との接続

RZ/T2N Evaluation Board と Inverter Board (INV1 と INV2)は直接接続します。RZ/T2N Evaluation Board と Inverter Board の高さを合わせる必要があります。RZ/T2N Evaluation Board のゴム足が 10mm(1) の時、Inverter board 用に 7mm のスペーサ(2)をご使用ください。

INV3 と INV4 はフラットケーブルを使用して接続します。10cm のフラットケーブルを使用してください。また、INV3 と INV4 は INV1 と INV2 に 30mm 以上のスペーサ(3)を用いて取り付けます。

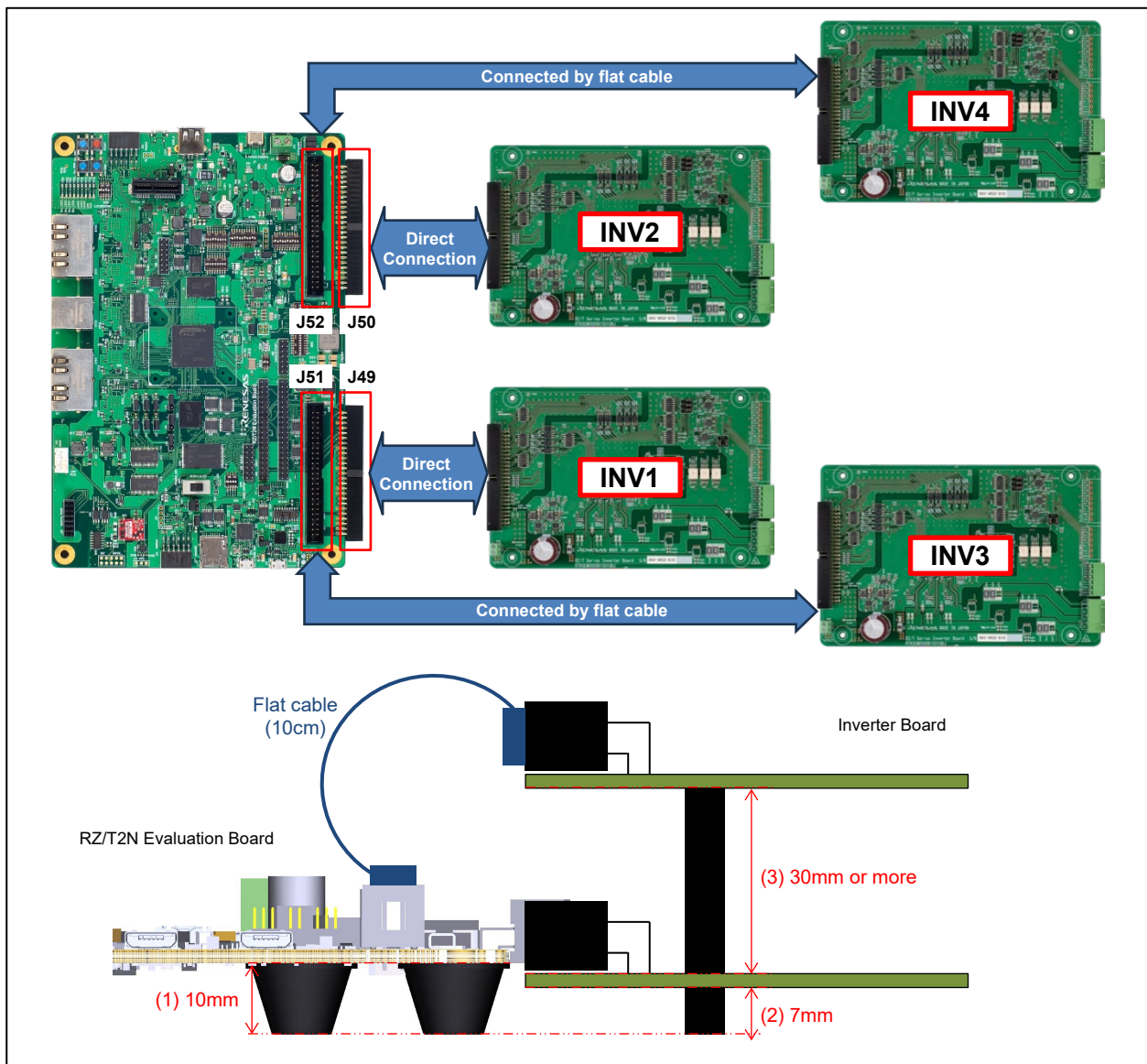


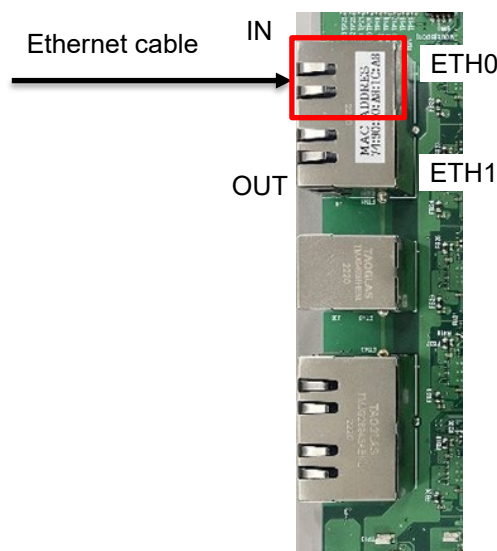
図 1-2 RZ/T2N Evaluation Board と Inverter Board との接続

1.3.2 モータ/エンコーダと Inverter Board との接続

多摩川精機社製:TSM3101N2001E020 (モータ/エンコーダ(FA-CODER))と Inverter Board との接続に関しては、「RZ/T Series Inverter Board/Kit ユーザーズマニュアル(R12UZ0155)」を参照してください。

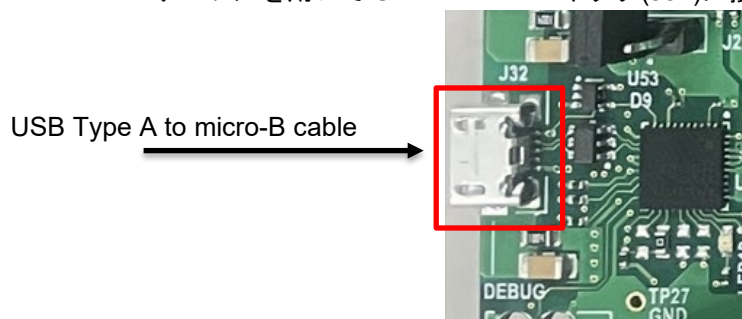
1.3.3 PC との接続

PC と RZ/T2N Evaluation Board の RJ45 コネクタの ETH0 に Ethernet ケーブルを接続します。



1.3.4 ICE

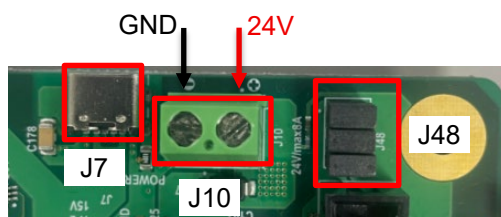
USB Type A to micro-B ケーブルを用いて USB micro-B コネクタ(J32)に接続してください。



1.3.5 電源

RZ/T2N Evaluation Board の端子台(J10)に 24V DC を印加することで、4 台のインバータボードに電源を供給することが可能です。その場合、Inverter Board に DC24V を印加しないでください。

J10 から供給可能な電流は最大 8A です。RZ/T2N Evaluation Board 上で消費する電流と Inverter board で消費する電流の合計が 8A を超える場合は、USB Type-C コネクタ(J7)より USB PD 対応 USB Type-C ケーブルを用いて 15V 電源を供給し、Inverter board には直接 24V 電源を供給してください。この時 J48 ジャンパは Open に設定してください。



2. 動作環境

2.1 動作環境

表 2.1 動作環境

項目	内容
使用ボード	RZ/T2N Evaluation Board
MPU	RZ/T2N グループ R9A07G076M48GBG: 537pinFBGA
動作周波数	CPU Core0 : 800MHz (Arm [®] Cortex [®] -R52) CPU Core1 : 800MHz (Arm [®] Cortex [®] -R52)
動作電圧	3.3V/1.8V/1.1V
動作モード	xSPI0 boot mode(x1 boot serial flash)
使用デバイス	Serial Flash ROM (64Mbytes) Macronix 製 MX25LW51245GXDI00-T I2C EEPROM (32Kbit) Microchip 製 24CW320T-I
通信プロトコル	EtherCAT [®] 
統合開発環境	RENESAS 製 e ² studio 2026-07 (26.07.0)
ICE	SEGGER 製 J-Link OB
SSC Tool	Provided by EtherCAT Technology Group (ETG) Slave Stack Code (SSC) Tool Version 5.13
Software PLC	Beckhoff Automation TwinCAT [®] 3
Flexible Software Package (FSP)	4.2.0

2.2 フォルダ、ファイル構成

```
r01an8308xx0100-rzt2n-4axis-motor-ecat-package
|- r01an8225jj0100-motor-encoder-foc-rzt2n.pdf      // Encoder Vector Control of Permanent Magnet
|                                                    Synchronous Motor - Serial Encoder (Japanese)
|- r01an8225ej0100-motor-encoder-foc-rzt2n.pdf      // Encoder Vector Control of Permanent Magnet
|                                                    Synchronous Motor - Serial Encoder (English)
|- r01an8308jj0100-rzt2n-4axis-motor-ecat-startup.pdf // This document (Japanese)
|- r01an8308ej0100-rzt2n-4axis-motor-ecat-startup.pdf // This document (English)
|
|- common
|   |- ESI
|   |   +- Renesas EtherCAT RZT2N MotorSolution.xml    // ESI file
|   |
|   |- Patch
|   |   |- apply_patch_e2studio.bat                    // Bat file
|   |   +- RZT2N_Motor_Solution.patch                  // Patch file
|   |
|   |- SSCconfig
|   |   +- RZT2N_Motor_Solution.esp                    // esp file
|   |
|- project
|   +- RZT2N_ESC_CIA402_E2S_V100                      // RZ/T2N CPU1 software
|   +- RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100             // RZ/T2N CPU0 software
```

3. 関連ドキュメント

- RTK9RZT2N1CW1010BJ ユーザーズマニュアル(R20UT5574)
- RZ/T Series Inverter Board/Kit ユーザーズマニュアル(R12UZ0155)
- RZ/T2N 永久磁石同期モータのエンコーダベクトル制御 - シリアルエンコーダ アプリケーションノート(R01AN8225)

- CiA402 規格書 :

- IEC 61800-7-201 Edition 1.0

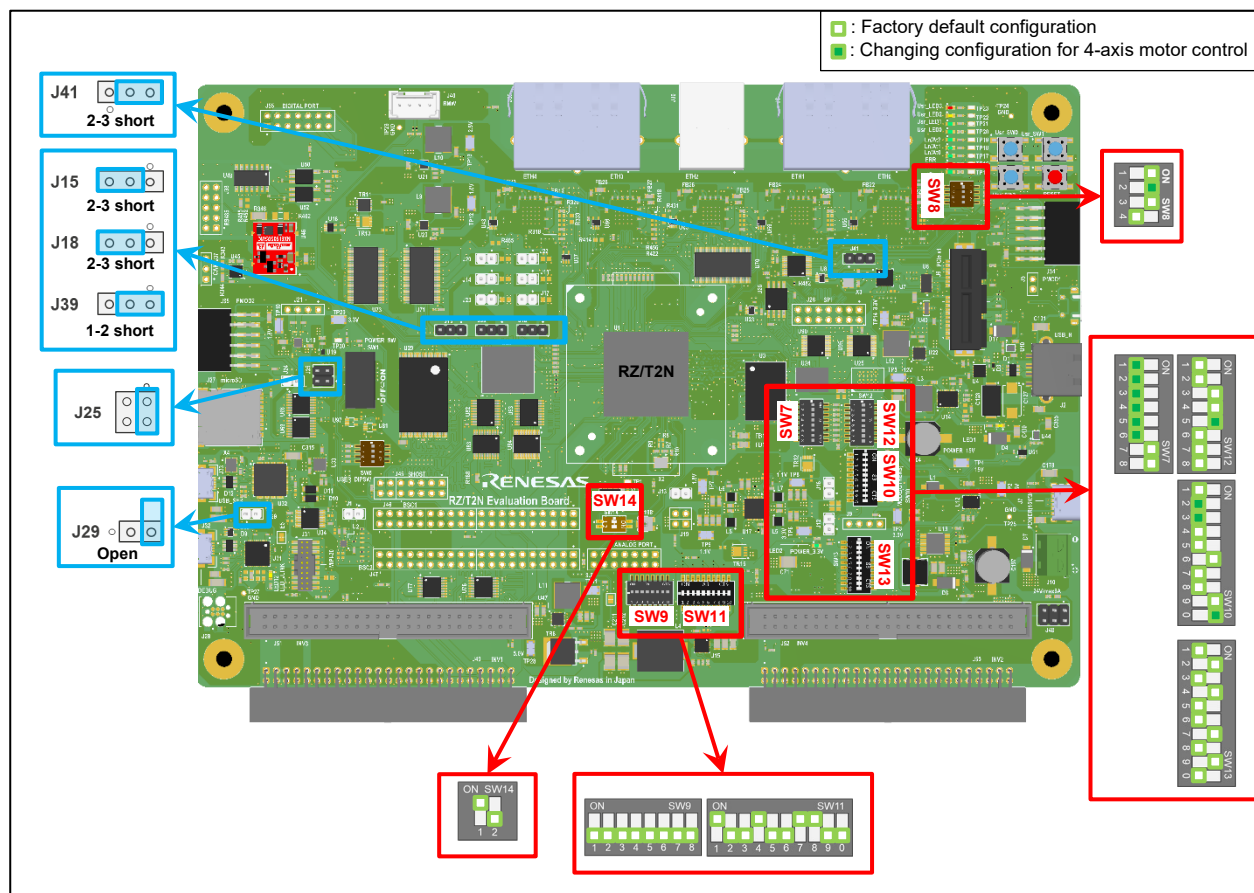
Adjustable speed electrical power drive systems Part 7-201: Generic interface and use of profiles for power drive systems Profile type 1 specification

- 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

4. ボードの設定

4.1 RZ/T2N Evaluation Board の設定



4.1.1 Switches

RZ/T2N Evaluation Board のスイッチの詳細は、「RTK9RZT2N1CW1010BJ ユーザーズマニュアル (R20UT5574)」を参照してください。

下表の赤字は出荷時の設定状態から変更する設定を示しています。

SW7

Pin	1	2	3	4	5	6	7	8
	OFF	OFF	OFF	OFF	OFF	OFF	ON	ON

SW8

Pin	1	2	3	4
	ON	ON	ON	OFF

SW9

Pin	1	2	3	4	5	6	7	8
	OFF	OFF	OFF	OFF	OFF	OFF	OFF	OFF

SW10

Pin	1	2	3	4	5	6	7	8	9	10
	OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	ON	ON

SW11

Pin	1	2	3	4	5	6	7	8	9	10
	ON	OFF	OFF	ON	OFF	OFF	ON	ON	OFF	OFF

SW12

Pin	1	2	3	4	5	6	7	8
	OFF	OFF	ON	ON	ON	OFF	OFF	OFF

SW13

Pin	1	2	3	4	5	6	7	8	9	10
	ON	OFF	ON	OFF	OFF	ON	OFF	ON	OFF	OFF

SW14

Pin	1	2
	ON	OFF

4.1.2 Jumpers

RZ/T2N Evaluation Board のジャンパの詳細は、「RTK9RZT2N1CW1010BJ ユーザーズマニュアル (R20UT5574)」を参照してください。

下表の赤文字は出荷時の設定状態からの変更する設定を示しています。

No	JP	設定
1	J15	2-3 short
2	J18	2-3 short
3	J25	1-2 short
4	J29	Open
5	J39	1-2 short
6	J41	2-3 short

4.2 Inverter Board

4.2.1 Jumpers

Inverter Board のジャンパの詳細は、「RZ/T Series Inverter Board/Kit ユーザーズマニュアル (R12UZ0155)」を参照してください。

No	JP	設定
1	JP7,8,9	1-2 short
2	JP10	2-3 short
3	JP11	2-3 short

5. プログラム書き込み

5.1 EtherCAT SubDevice Stack の生成

※ 本サンプル・プロジェクトには EtherCAT SubDevice 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 SubDevice Stack Code(SSC) Tool”が必要です。

SSC Tool は ETG 協会から入手可能です。

※ SSC Tool 起動前に“付録：SSC Tool 使用時の注意点”を確認してください。

※ 本ガイドを実行するためには、PC に patch.exe がインストールされている必要があります。

“付録：patch のインストール方法”を参照して patch.exe をインストールして下さい。

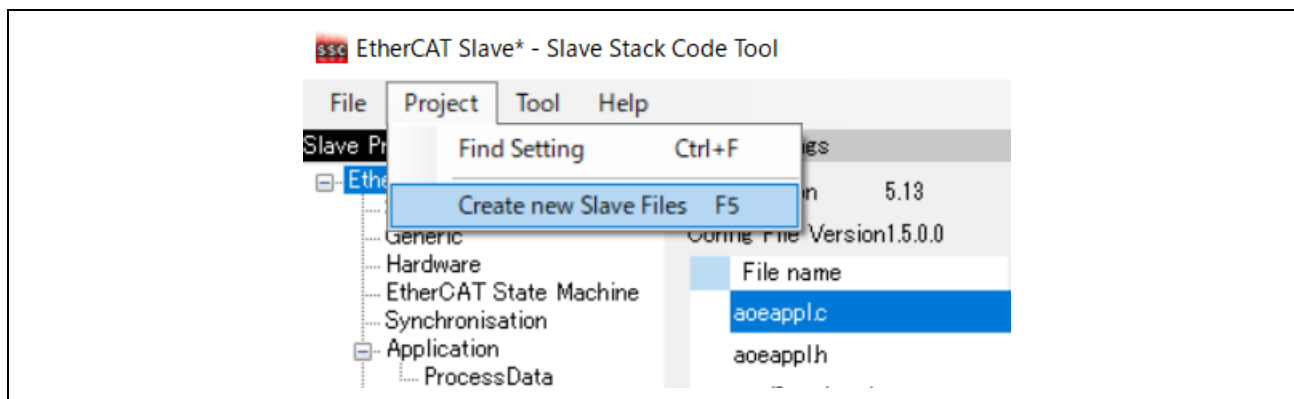
(1) 下記のフォルダ構成になるように、r01an8308xx0100-rzt2n-4axis-motor-ecat-package.zip を解凍します。

```
r01an8308xx0100-rzt2n-4axis-motor-ecat-package
├── common
└── project
    ├── RZT2N_ESC_CIA402_E2S_V100
    └── RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100
```

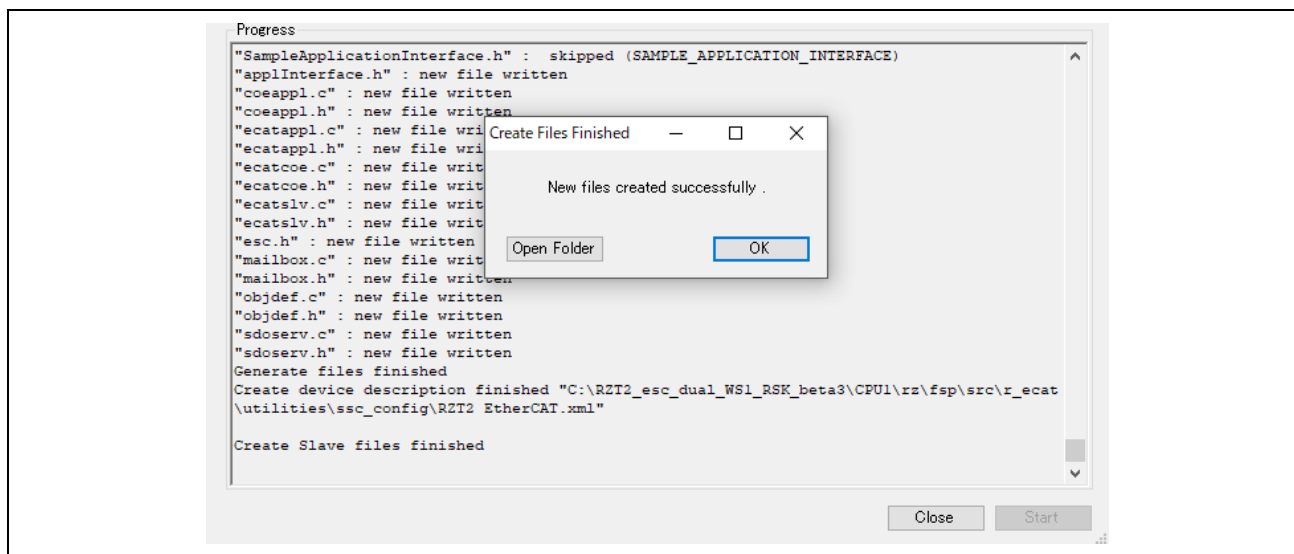
(2) “Common¥SSCConfig”フォルダに格納されている esp ファイルをダブルクリックし、SSC Tool を起動して下さい。

 RZT2N_Motor_Solution.esp

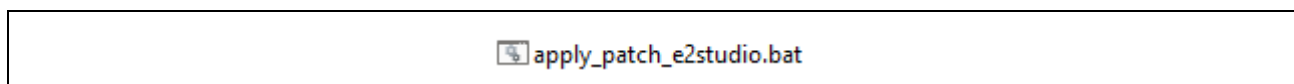
(3) [Project]→[Create New Slave Files]をクリックします。ダイアログが開いたら[Start]をクリックします。



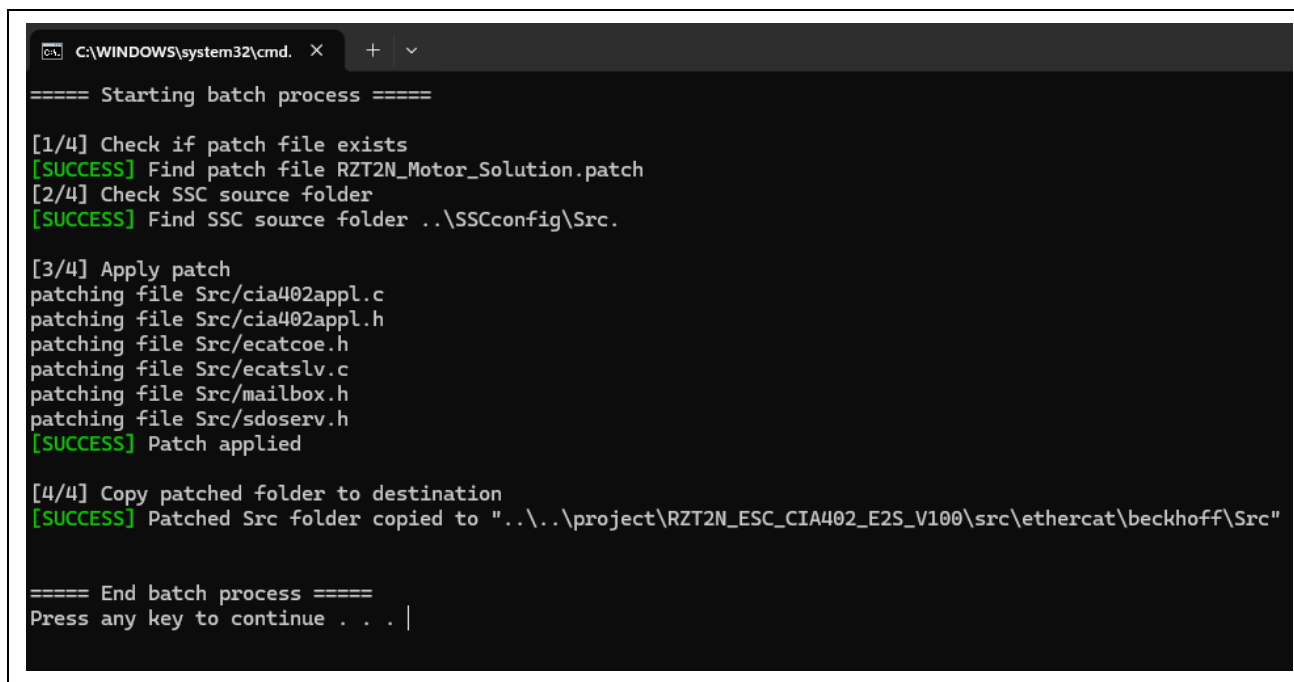
- (4) ソースコードが生成され、成功すると” New Files created successfully” と表示されるので[OK]をクリックします。



- (5) “Common¥Patch”フォルダに格納されている[apply_patch_e2studio.bat]をダブルクリックして bat ファイルを実行します。

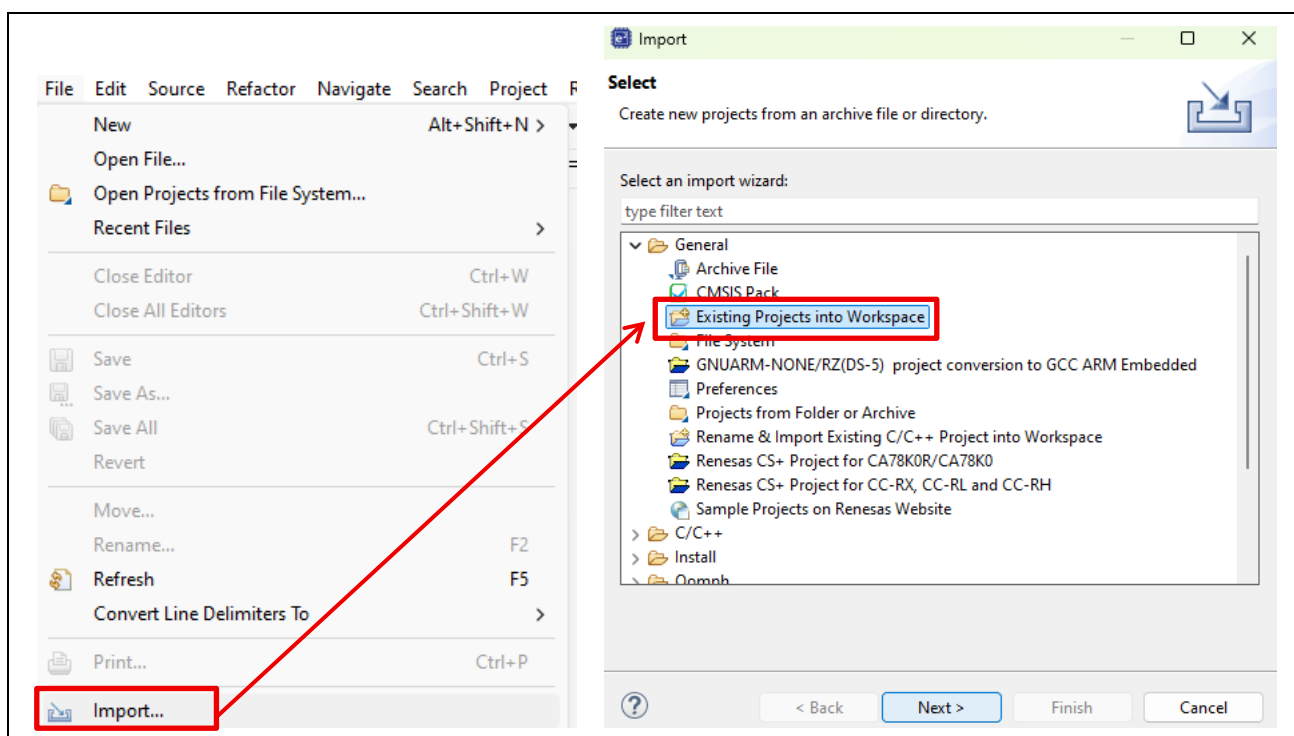


- (6) Bat ファイル実行後、project¥RZT2N_ESC_CIA402_E2S_V100¥src¥ethercat¥beckhoff フォルダに Src フォルダが移動されます。

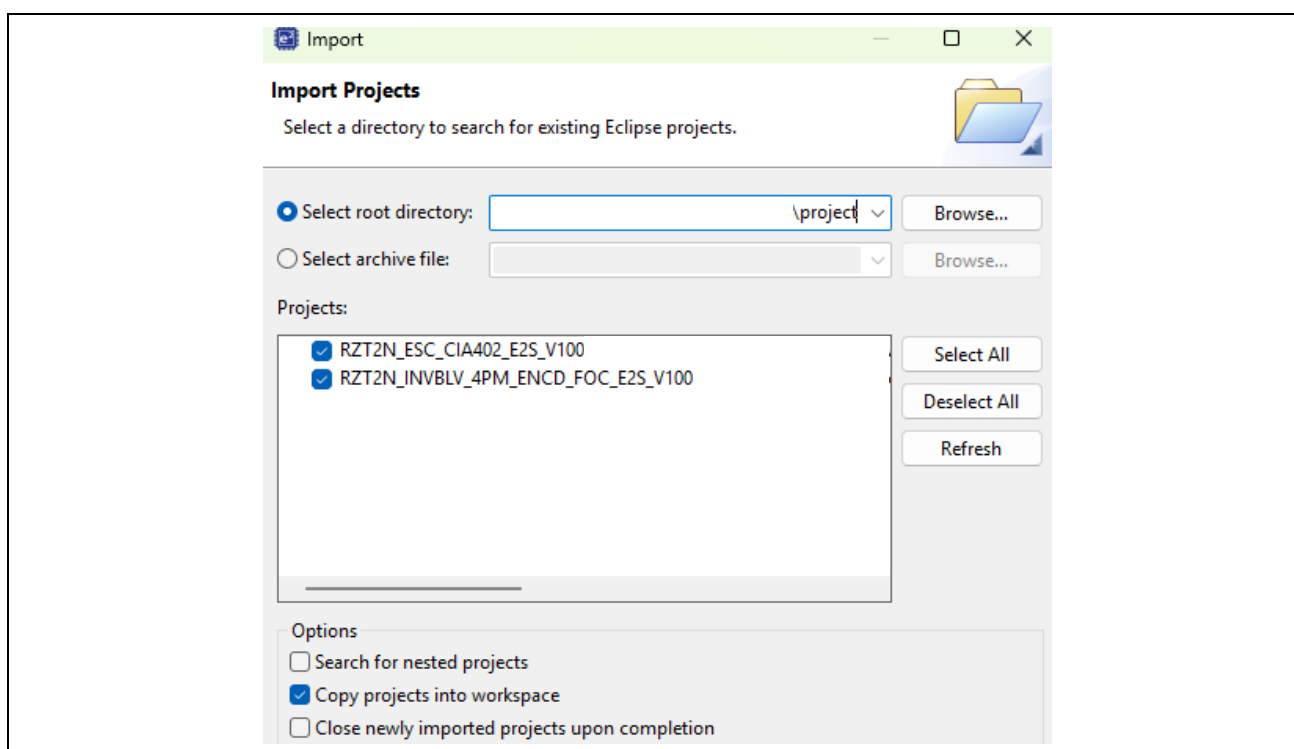


5.2 プロジェクトのビルド

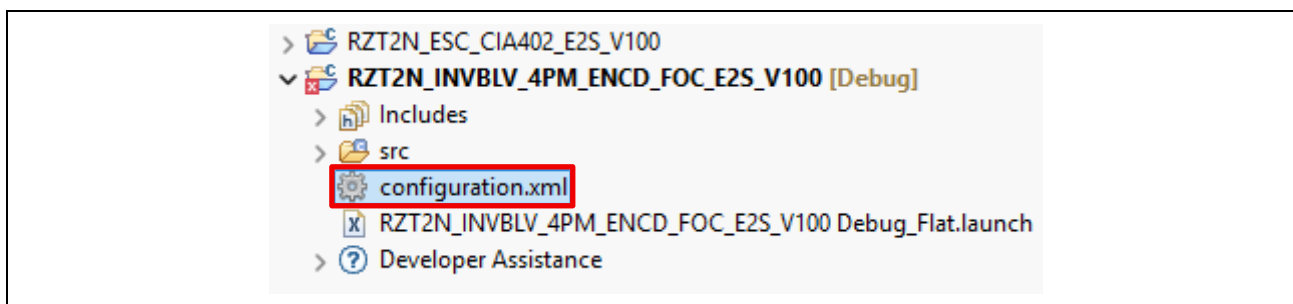
- (1) e² studio を起動し、上のメニューバーから"File">"Import">"Existing Projects into Workspace"を選択して"Next"を押します。



- (2) "Select root directory" を選択し、"project"フォルダを選択します。
- (3) "RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100"および"RZT2N_ESC_CIA402_E2S_V100"にチェックマークを付けます。Options の中で、"Copy projects into workspace"にチェックマークを付け、ワークスペース内にコピーしながらインポートします。



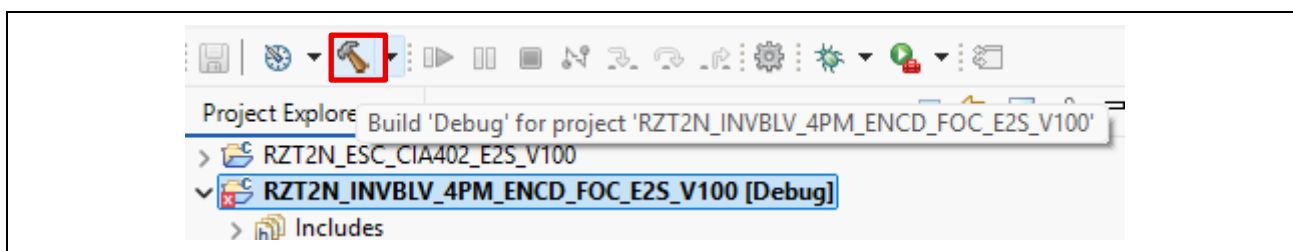
(4) RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 の configuration.xml を開きます。



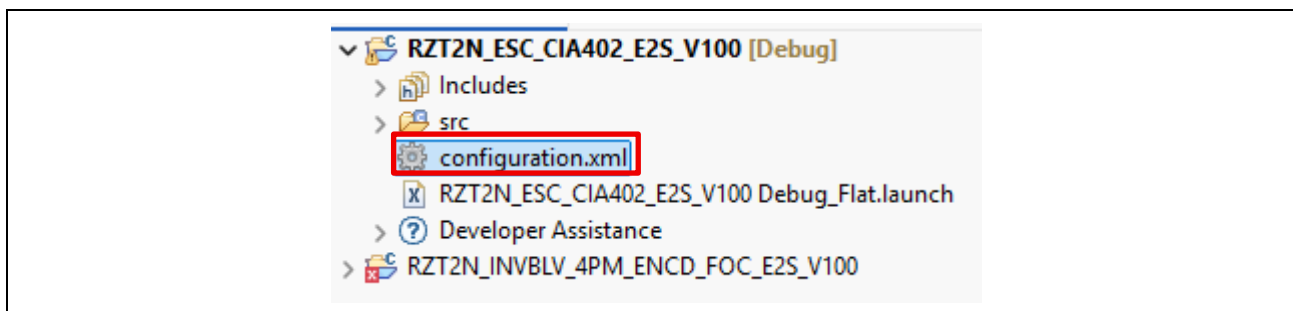
(5) コード生成を行います。コード生成が完了したら configuration.xml を閉じます。



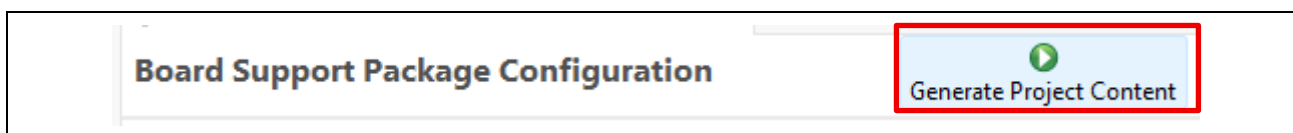
(6) RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 のビルドを実行します。



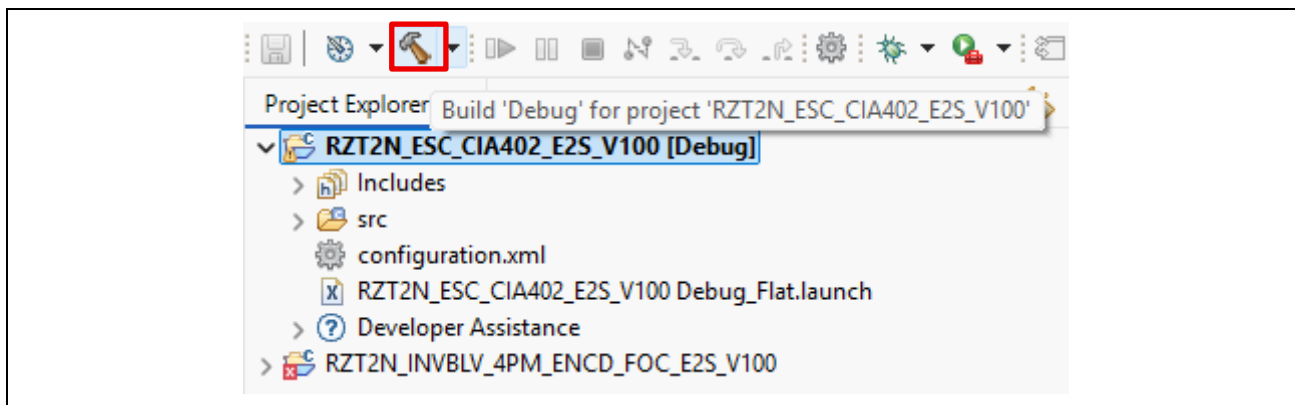
(7) RZT2N_ESC_CIA402_E2S_V100 の configuration.xml を開きます。



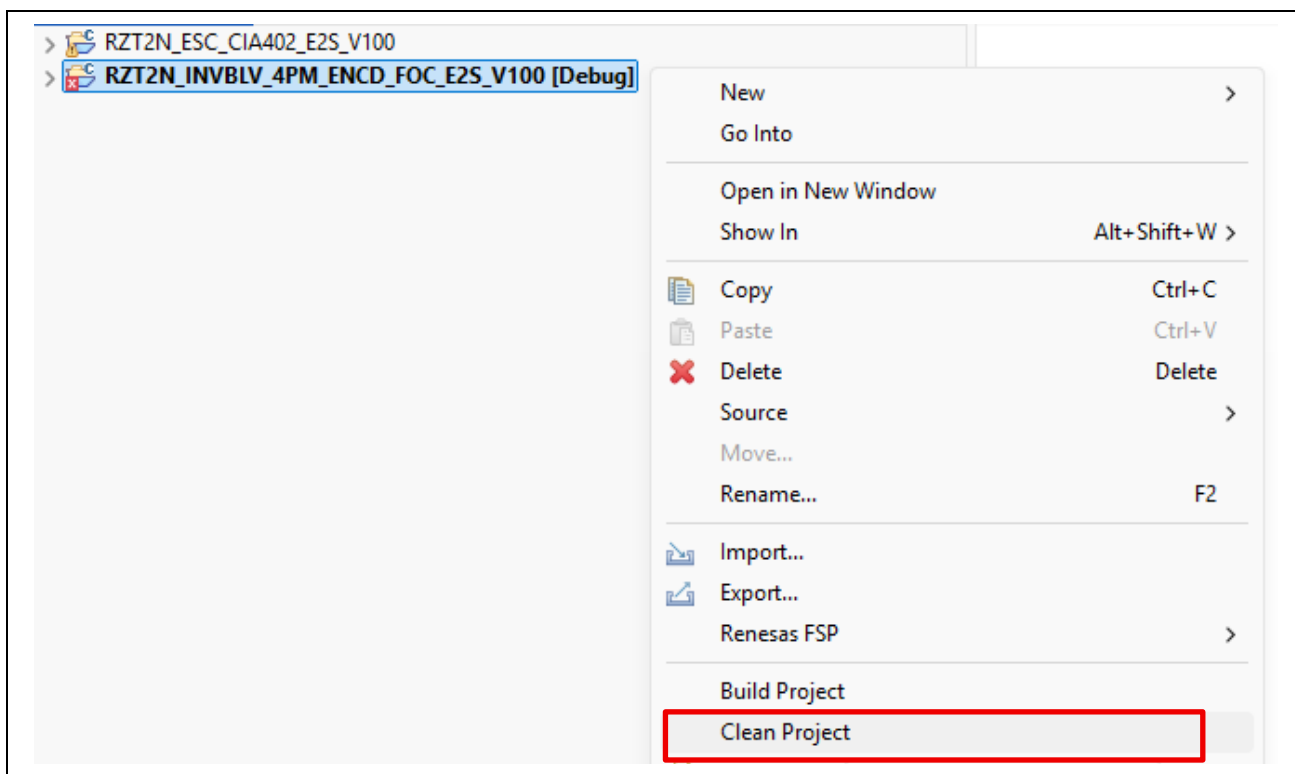
(8) コード生成を行います。コード生成が完了したら configuration.xml を閉じます。



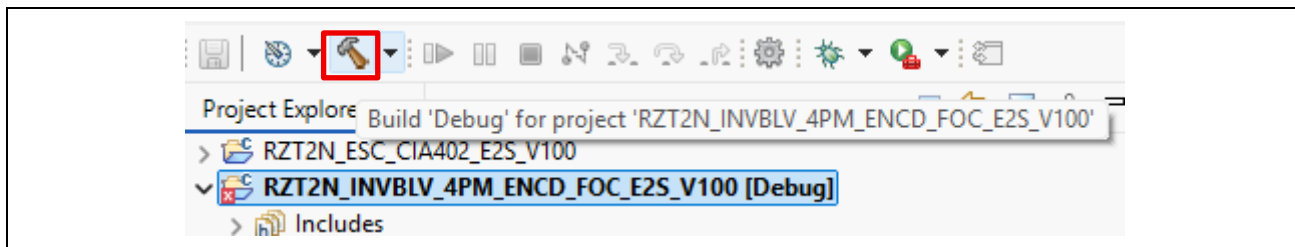
(9) RZT2N_ESC_CIA402_E2S_V100 のビルドを実行します。



(10) RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 プロジェクトでクリーンを行います。

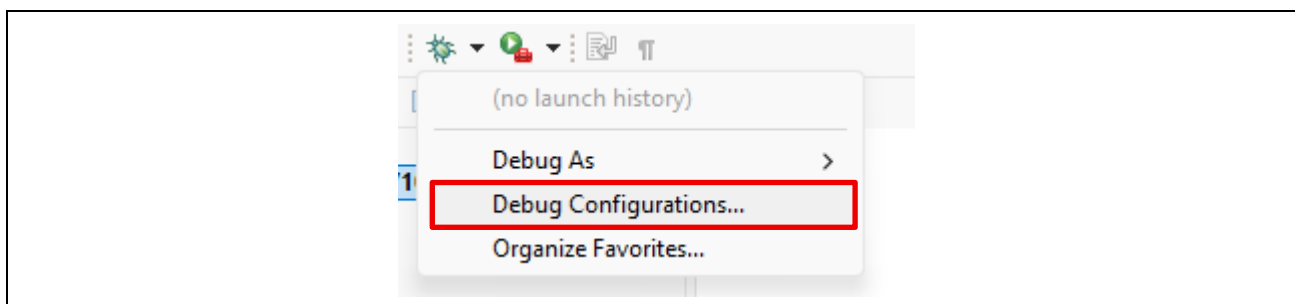


(11) クリーン完了後、再度 RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 のビルドを実行します。

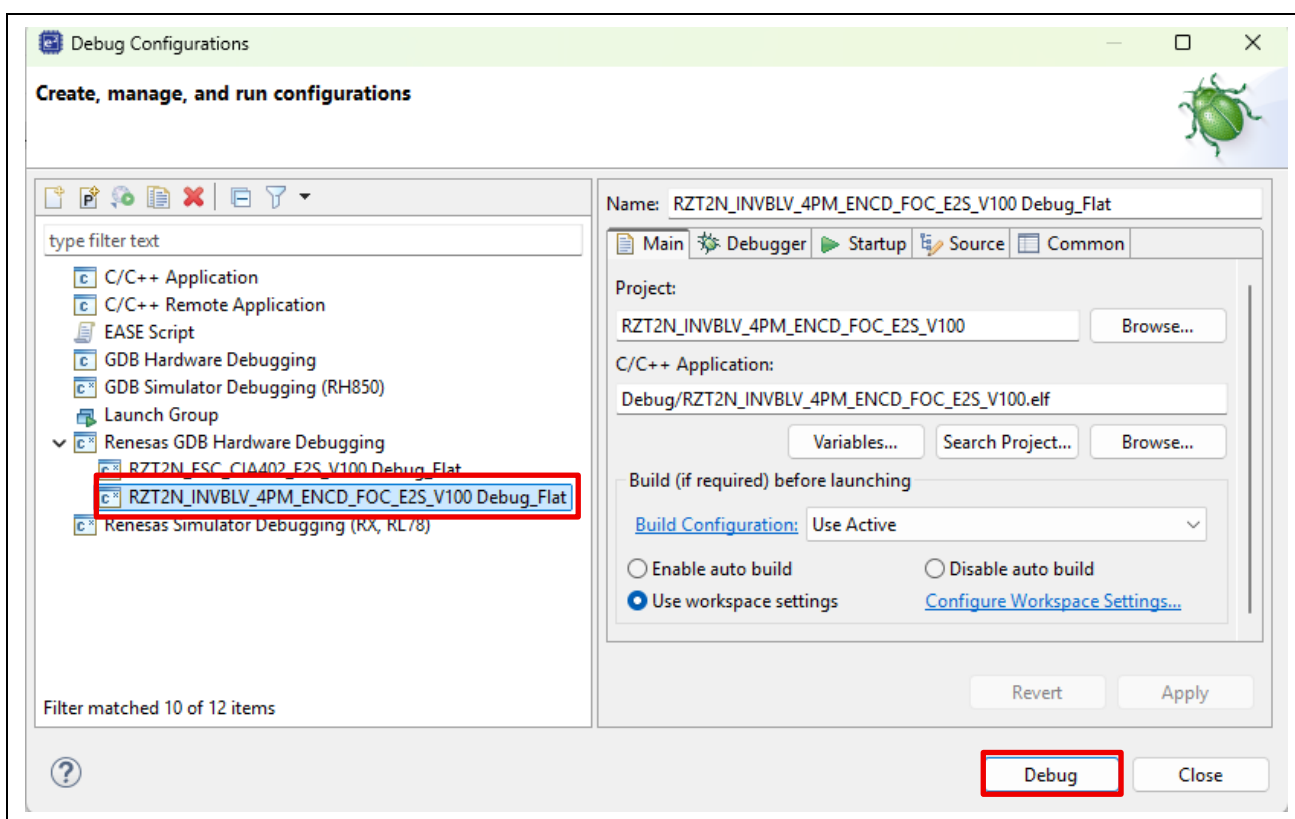


5.3 プロジェクトの書き込み

- (1) RZ/T2N EVB に 24V DC を印加し、電源を入れます。
- (2) USB type A to USB micro-B ケーブルを RZ/T2N EVB に接続した後、デバッグ構成を開きます。



- (3) 『RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 Debug_Flat』を選択してデバッグを行います。



- (4) 書き込みが完了し、デバッグが開始したらデバッグを終了し、ボード上のリセットボタンを押して再起動させます。

本サンプルはスタンドアロンでの実行が可能です。

USER LED1 (緑色 LED) が点灯した場合、CPU0・CPU1 共に正常に書き込みが完了しています。

6. 実行方法

6.1 TwinCAT との接続

6.1.1 ESI ファイルの準備（初回のみ）

- (1) TwinCAT を起動する前に、本サンプル向け ESI ファイルを特定のフォルダへコピーする必要があります。

コピー元の ESI ファイル(Common¥ESI フォルダに格納)

- “Renesas EtherCAT RZT2N MotorSolution.xml”

コピー先フォルダ（お使いの環境によっては異なる場合があります）

- C:¥TwinCAT¥3.1¥Config¥Io¥EtherCAT

6.1.2 TwinCAT ドライバのインストール（初回のみ）

- (1) TwinCAT3 を起動します。
- (2) スタートメニューから[TwinCAT3]→[Show Realtime Ethernet Compatible Devices...]を選択します。
- (3) 使用するイーサネットアダプタを選択し、インストールを行います。
- (4) 選択したイーサネットアダプタが[Installed and ready to use devices(realtime capable)]に移動すれば問題ありません。

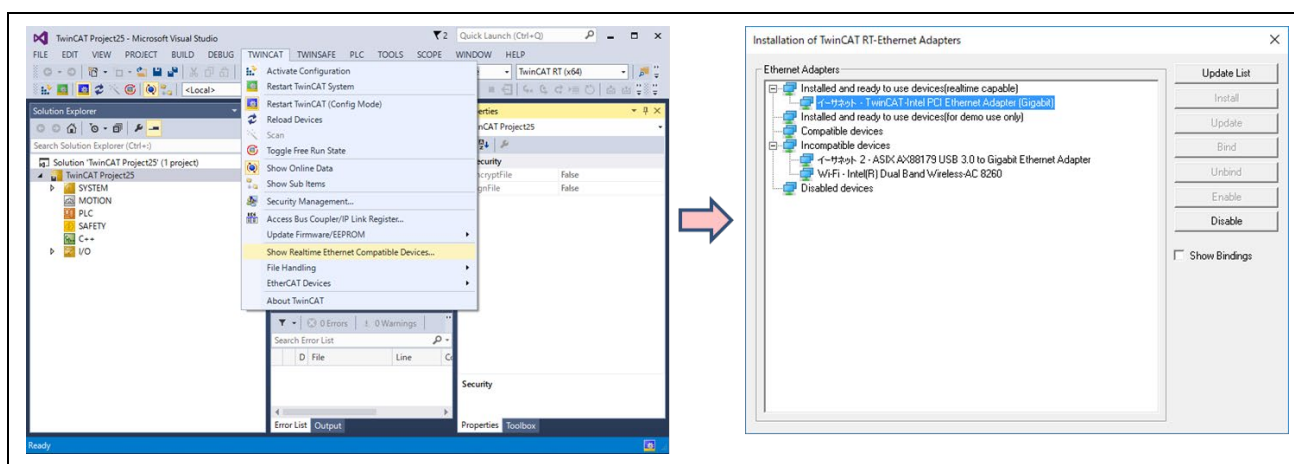


図 6-1. TwinCAT ドライバのインストール

6.1.3 デバイスのスキャン・EEPROM への書き込み

- (1) TwinCAT で新規プロジェクトを作成します。
- (2) ソリューションエクスプローラ→I/O →デバイス→「スキャン」を選択します。

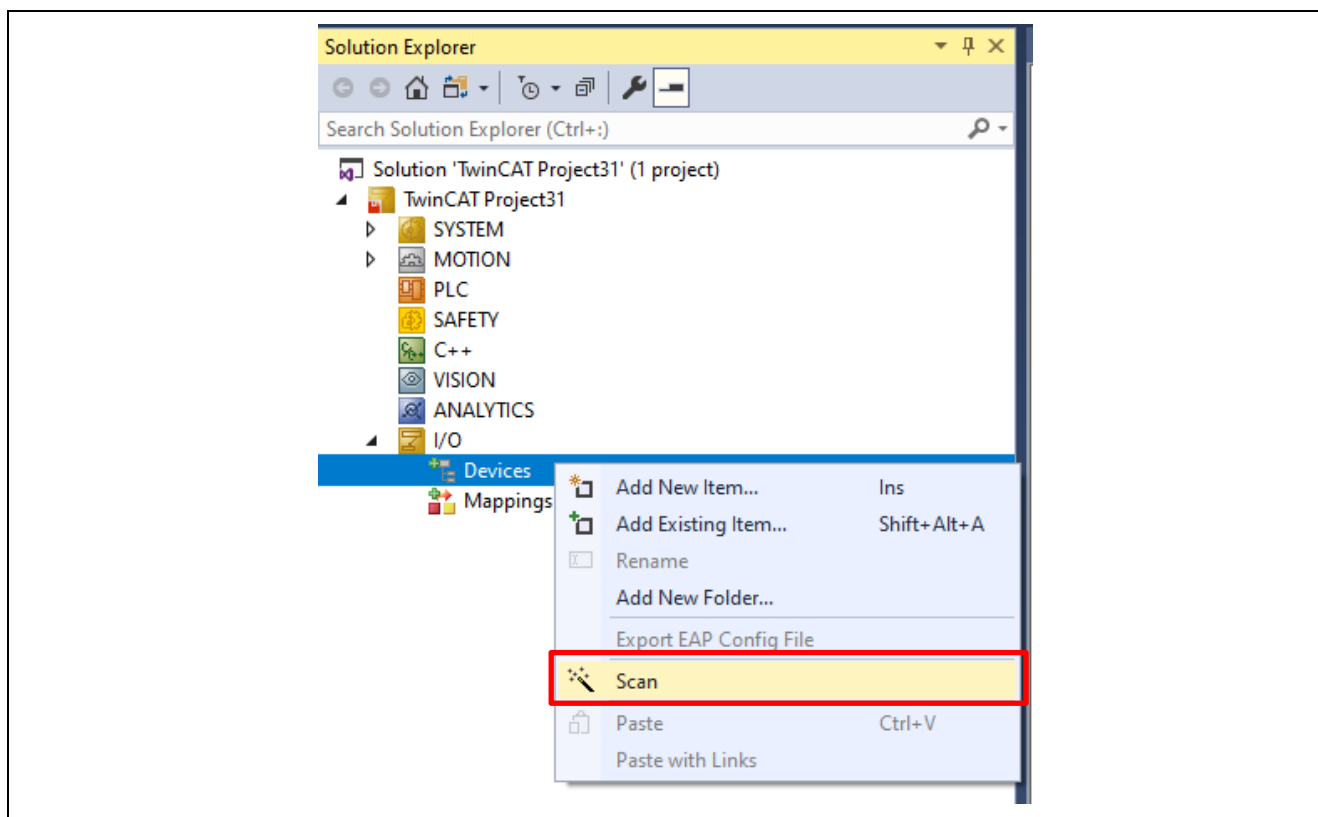


図 6-2. デバイスのスキャン

- (3) RZ/T2N が EtherCAT として認識された場合、下記のように EtherCAT ポートが表示されます。

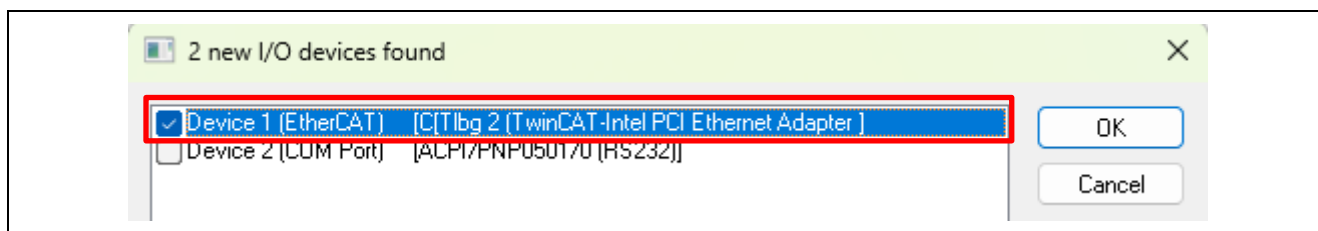


図 6-3. スキャン結果

(4) [Scan for boxes]を実行します。

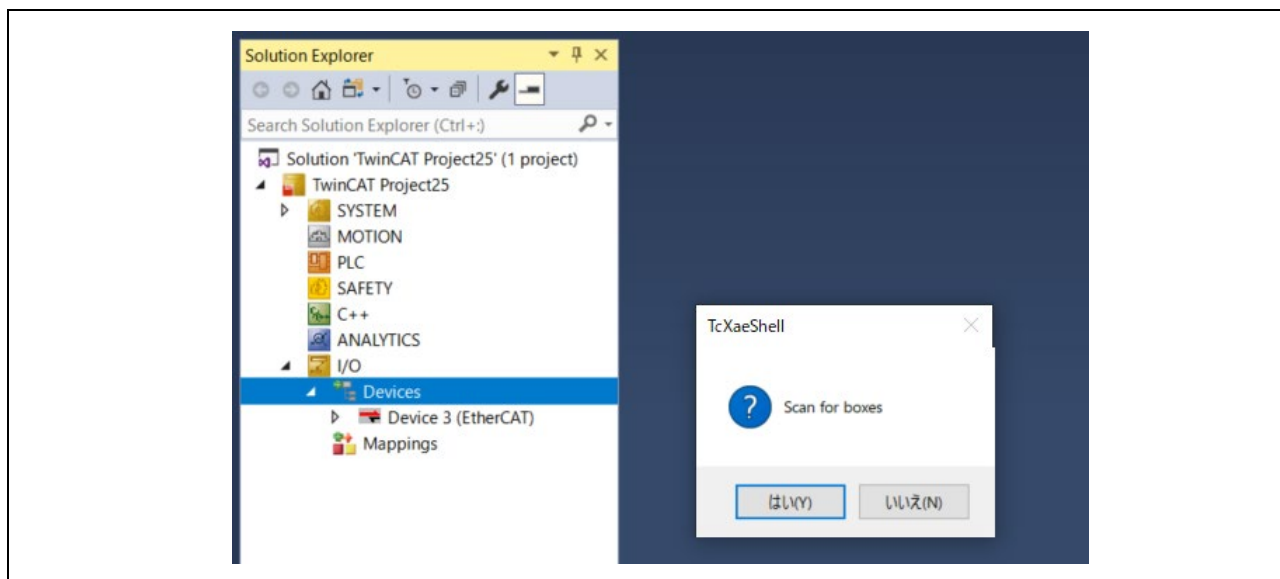


図 6-4. Scan for boxes ウィンドウ

(5) [Activate free run]は OK を押します。

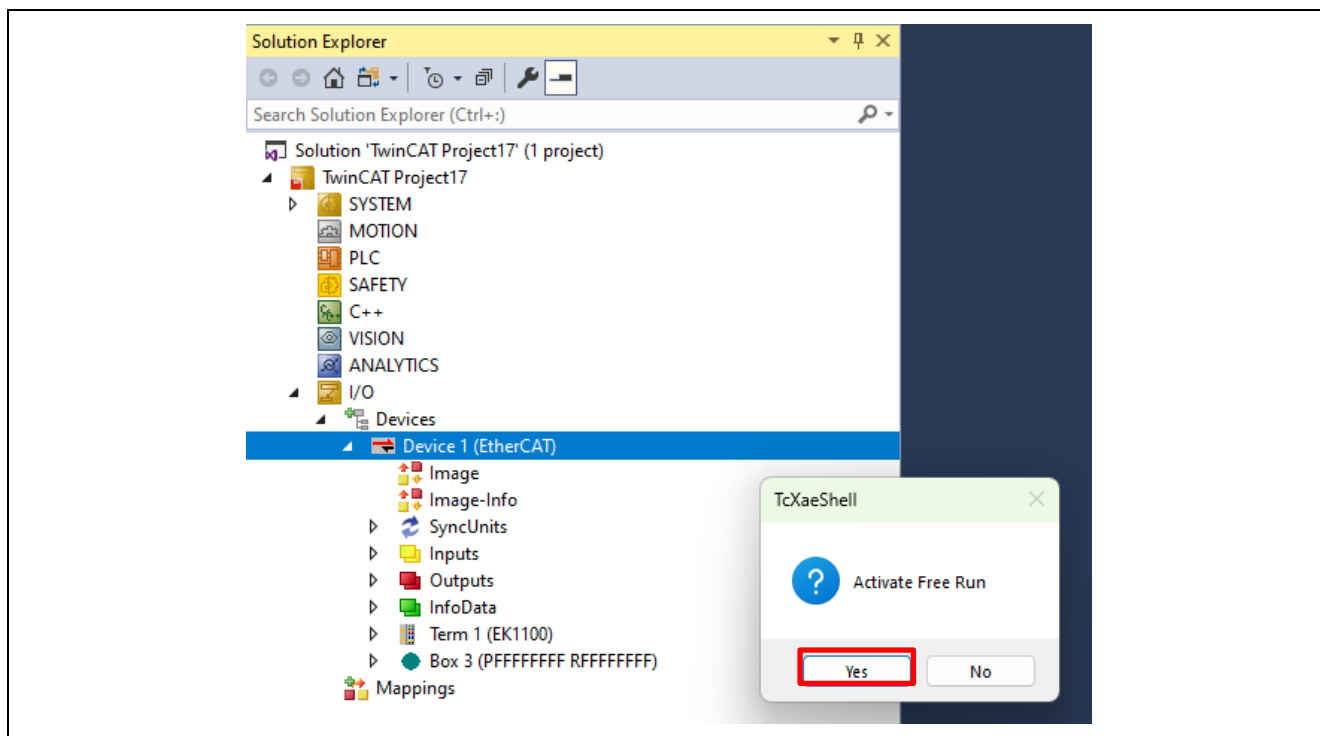


図 6-5. Active Free Run の有効化

(6) Box x (PFFFFFF..)と表示されている場合、RZ/T2N EVB の EEPROM は空の状態です。EEPROM に本サンプル用の EtherCAT SubDevice 情報を書き込む必要があります。

(7) EEPROM に書き込む手順は下記の通りです。

- [Box X] をダブルクリックしてください。設定画面が表示されます。
- [EtherCAT] tab を選択してください。
- [Advanced Setting] ボタンを実行してください。

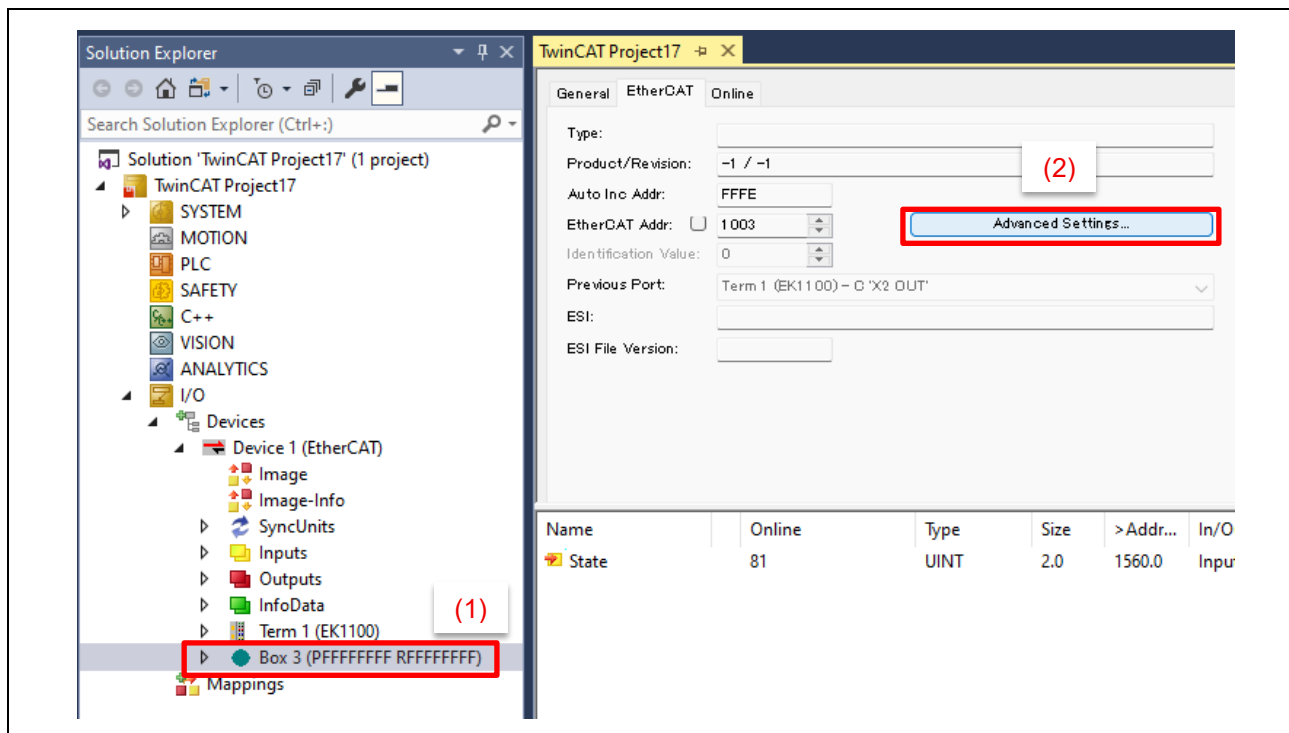


図 6-6 EEPROM 書き込み方法 1

- [ESC Access] → [EEPROM] → [Hex Editor]を選択してください
- [Download from List] を選択してください。

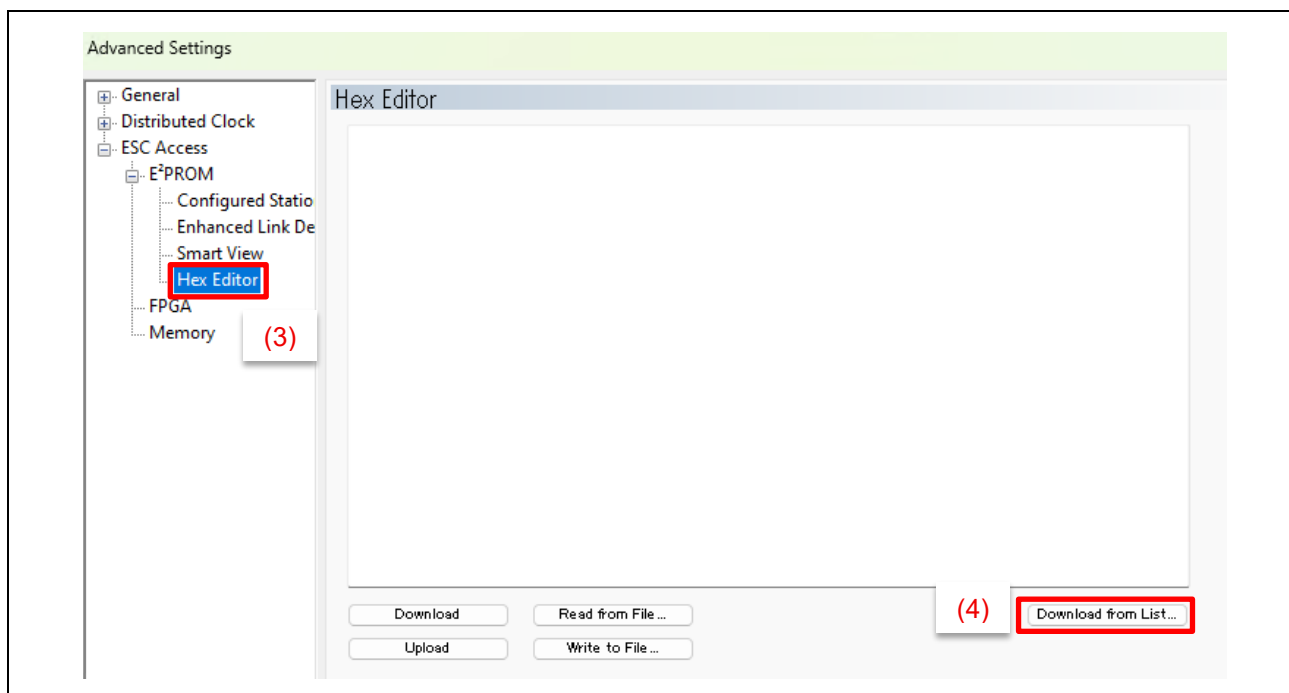


図 6-7. EEPROM 書き込み方法 2

- TwinCAT3 に登録してある ESI ファイルの一覧が現れますので、該当するファイルを選択してください。本サンプルでは、[Renesas EtherCAT RZ/T2N MotorSolution] を選択します。

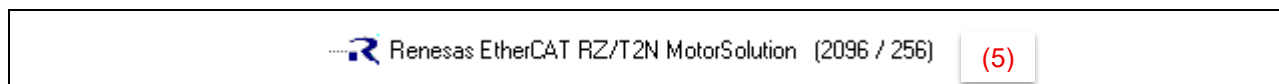


図 6-8. EEPROM 書き込み方法 3

- (8) EEPROM への書き込み完了後、ボードのリセットボタンを押して RZ/T2N を再起動し、EEPROM を再ロードします。
- (9) TwinCAT 上で[Device x (EtherCAT)]を一旦削除した後、再スキャンを行います。

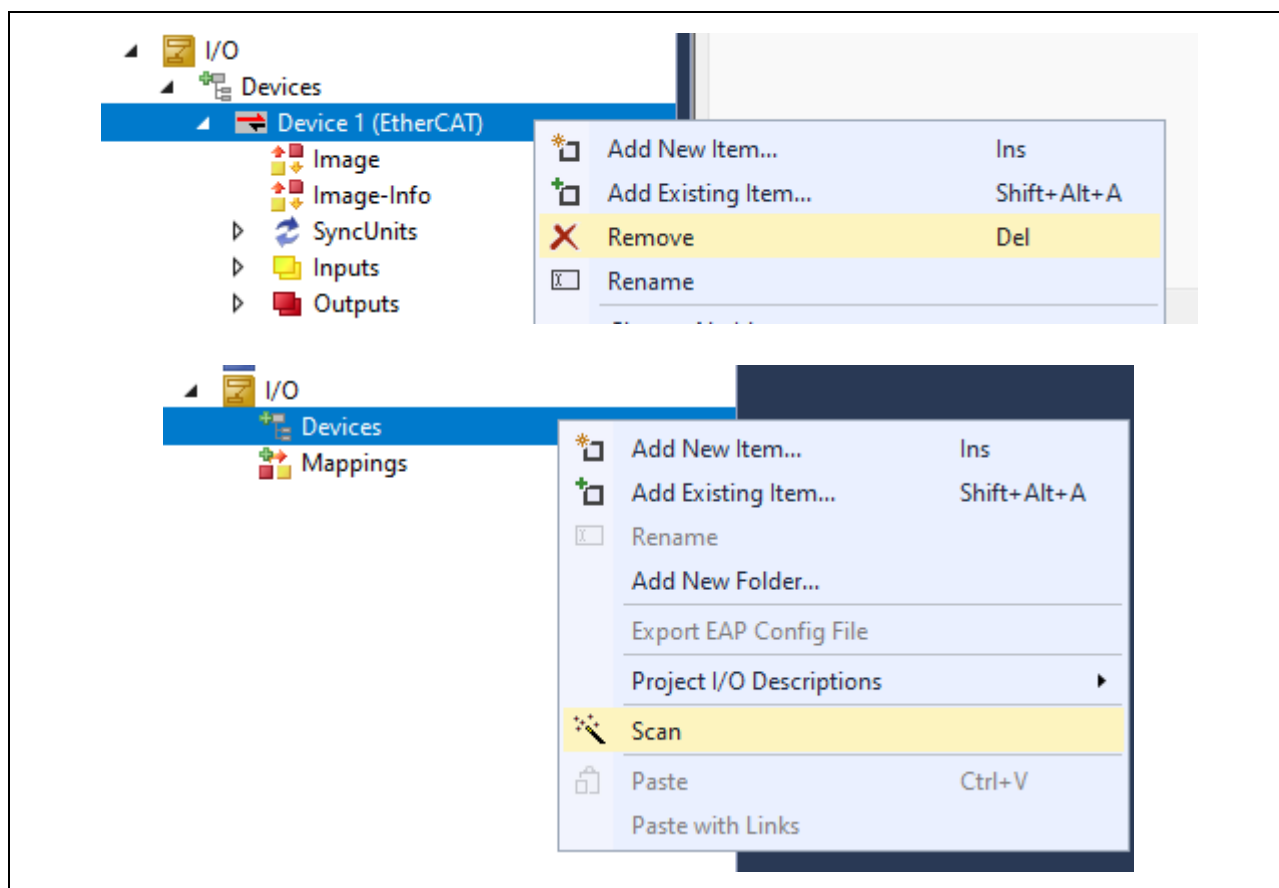


図 6-9. デバイスの再スキャン

(10) 正常にスキャンされた場合、下記の表示が出ます。

EtherCAT drive は動作確認で使用しないため、[Cancel]を選択します。

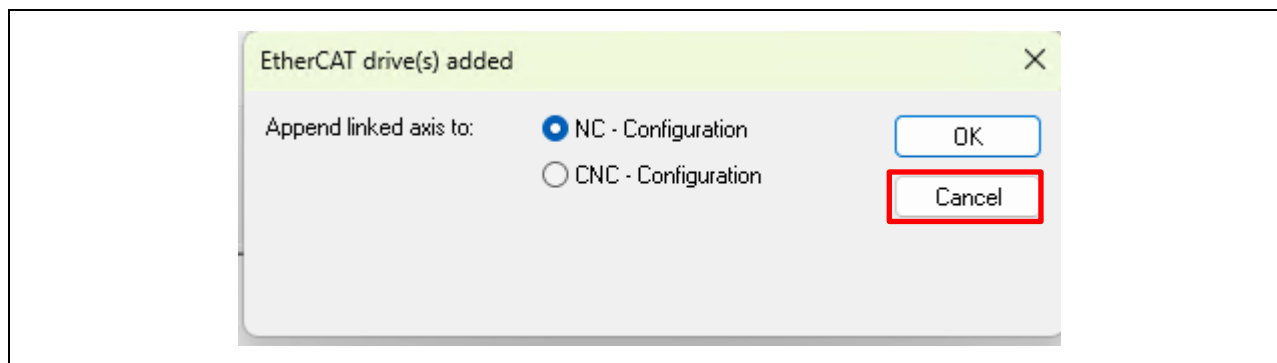


図 6-10. EtherCAT drive の選択

(11) [Reload Devices]を実行し、ネットワークを再起動します。

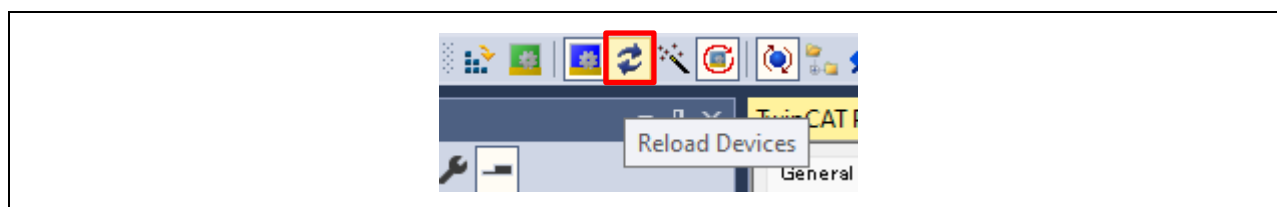


図 6-11. ネットワークの再起動

(12) ツリー上に、下記の画像のようにデバイスが表示されており、[Online]タブで Current State が“OP”になっていることを確認します。

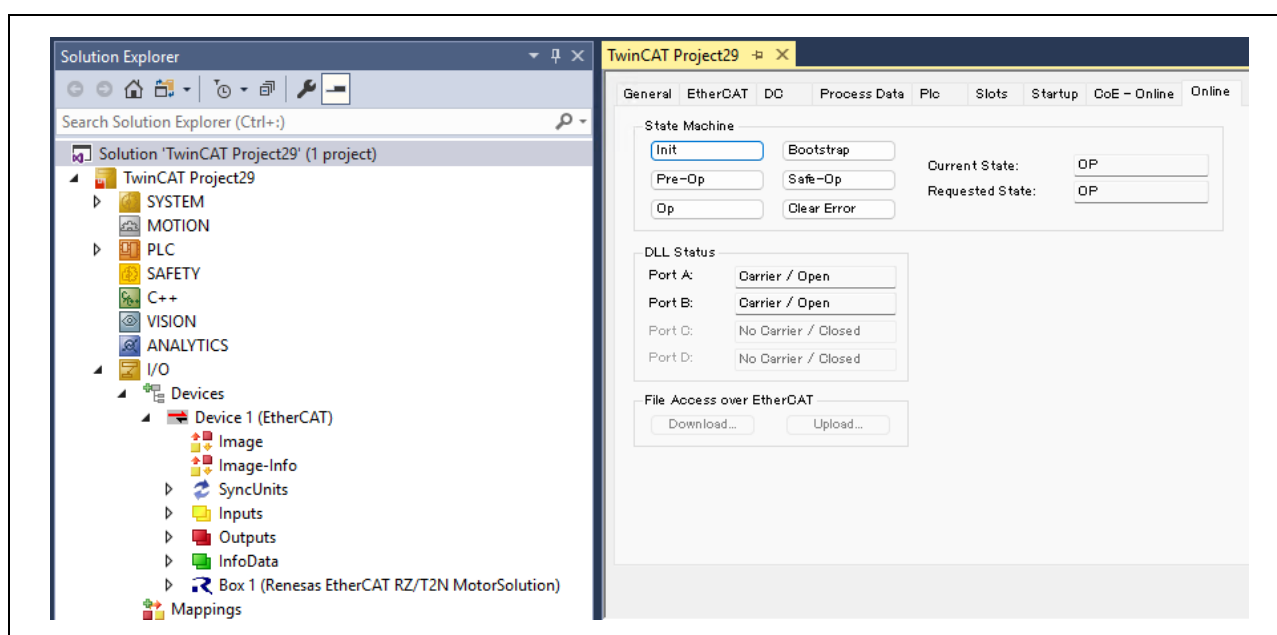


図 6-12. “OP”の確認

6.2 TwinCAT プロジェクトの実行方法

はじめに、RZ/T2N EVB に電源投入を行い、ボード上のマイコンに書き込んだプログラムを動作させてください。

以降に、TwinCAT プロジェクトを実行するまでの手順を示します。

(1) はじめに、EtherCAT 通信周期を設定します。本章では、EtherCAT 通信周期を 250us にする場合の設定例を示します。

1. TwinCAT3 のシステムマネージャツリーで、[PLC]を右クリックします。
2. [Add New Item...]を選択し、新規プロジェクトを作成します。
3. [Standard PLC Project]を選択してください。
4. 任意のプロジェクト名を記載してください。
5. [Add]をクリックしてください。

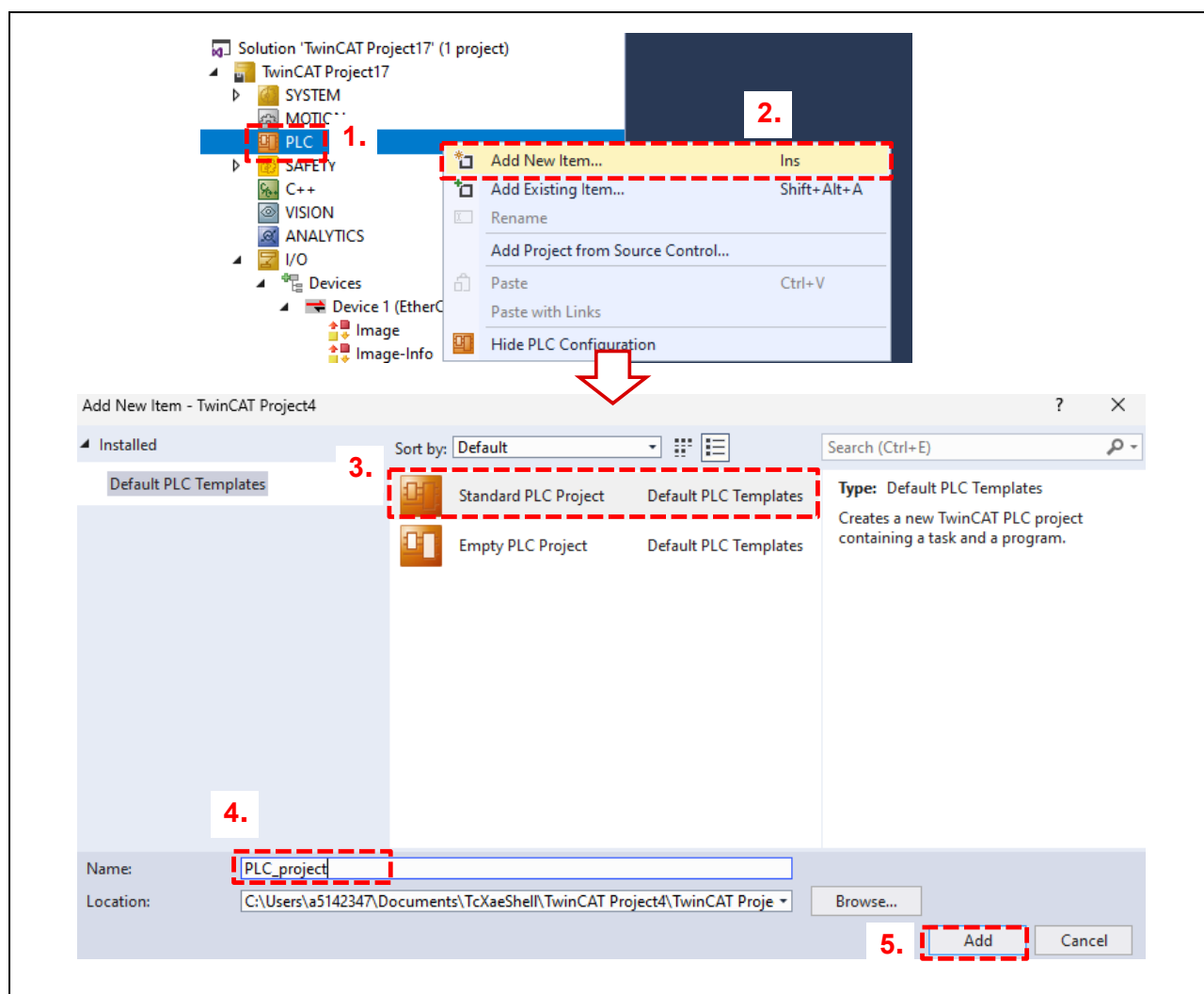


図 6-13. PLC プロジェクトの作成

6. TwinCAT3 のシステムマネージャツリーで[SYSTEM]→[Real-Time]を順にクリックし、プロパティ画面を開きます。
7. [Settings]タブをクリックします。
8. [Base Time]のプルダウンメニューから"250us"を選択します。
9. TwinCAT3 のシステムマネージャツリーで[SYSTEM]→[Tasks]→[PlcTask]を順にクリックし、プロパティ画面を開きます。
10. [Task]タブをクリックします。
11. [Cycle ticks]左側の設定メニューにおいて、スピンボタンで"1"を設定します。
12. [Cycle ticks]右側の設定メニューの値が、"0.250ms"になっていることを確認します。

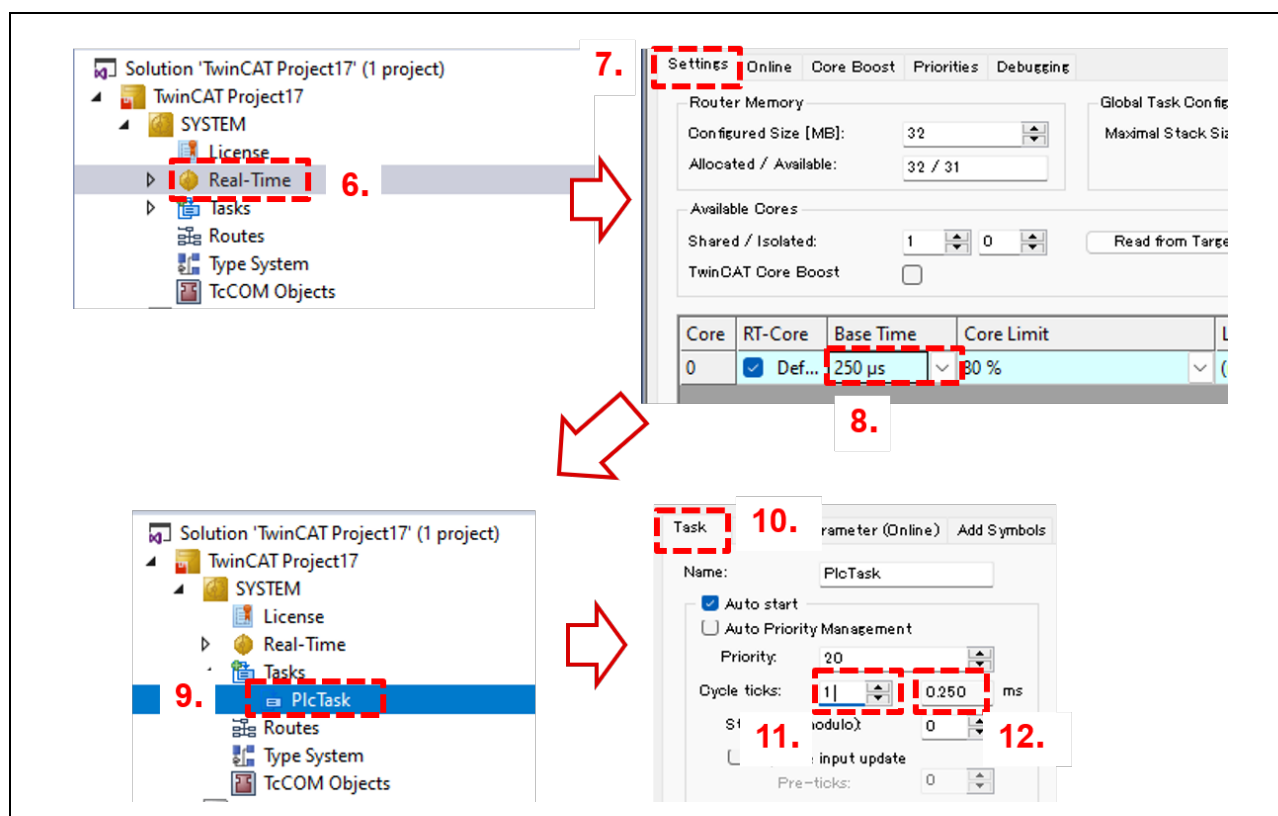


図 6-14. EtherCAT 通信周期の設定

(2) 次に、EtherCAT 同期モードを DC 同期モードに設定する手順を示します。

1. TwinCAT3 のシステムマネージャツリーで[Box X (Renesas EtherCAT RZ/T2N MotorSolution)]をクリックし、プロパティ画面を開きます。
2. [DC]タブをクリックします。
3. Operation Mode で、[DC-Synchron]を選択します。
4. [Advanced Settings...]をクリックします。
5. [Enable SYNC1]にチェックを入れます。
6. [Shift Time (us):]にて、SYNC1 割り込みの発生タイミングを設定します。
7. [OK]をクリックします。

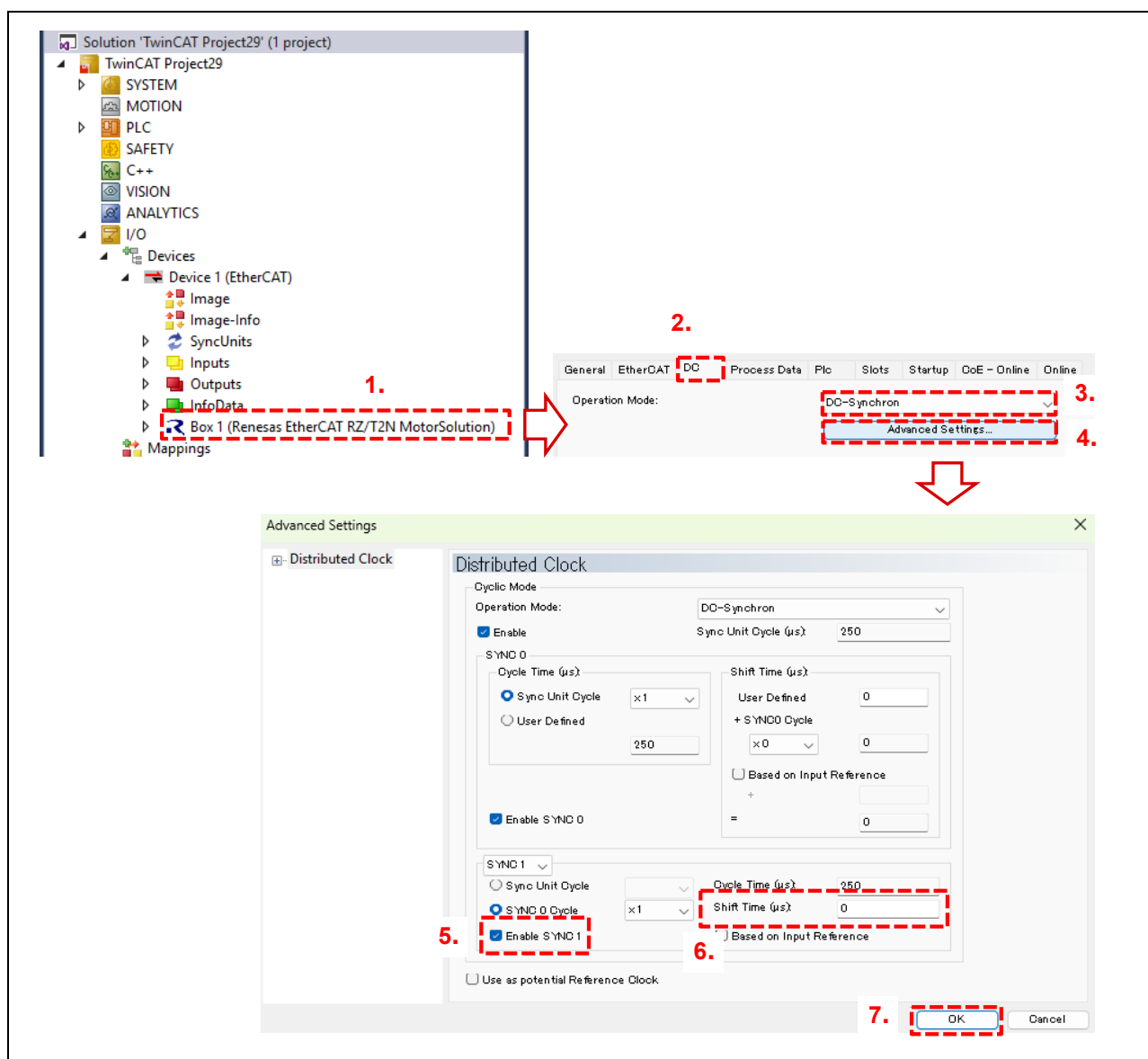


図 6-15. DC 同期モードの設定 1

8. TwinCAT3 のシステムマネージャツリーで[I/O]→[Devices]→[Device X (EtherCAT)]を順にクリックし、プロパティ画面を開きます。
9. [EtherCAT]タブをクリックします。
10. [Advanced Settings...]をクリックし、Advanced Settings ウィンドウを開きます。
11. Advanced Settings ウィンドウにおいて、[Distributed Clocks]をクリックします。
12. Percent of cycle time にて、SYNC0 割り込みの発生タイミングを設定します。
13. [OK]をクリックします。

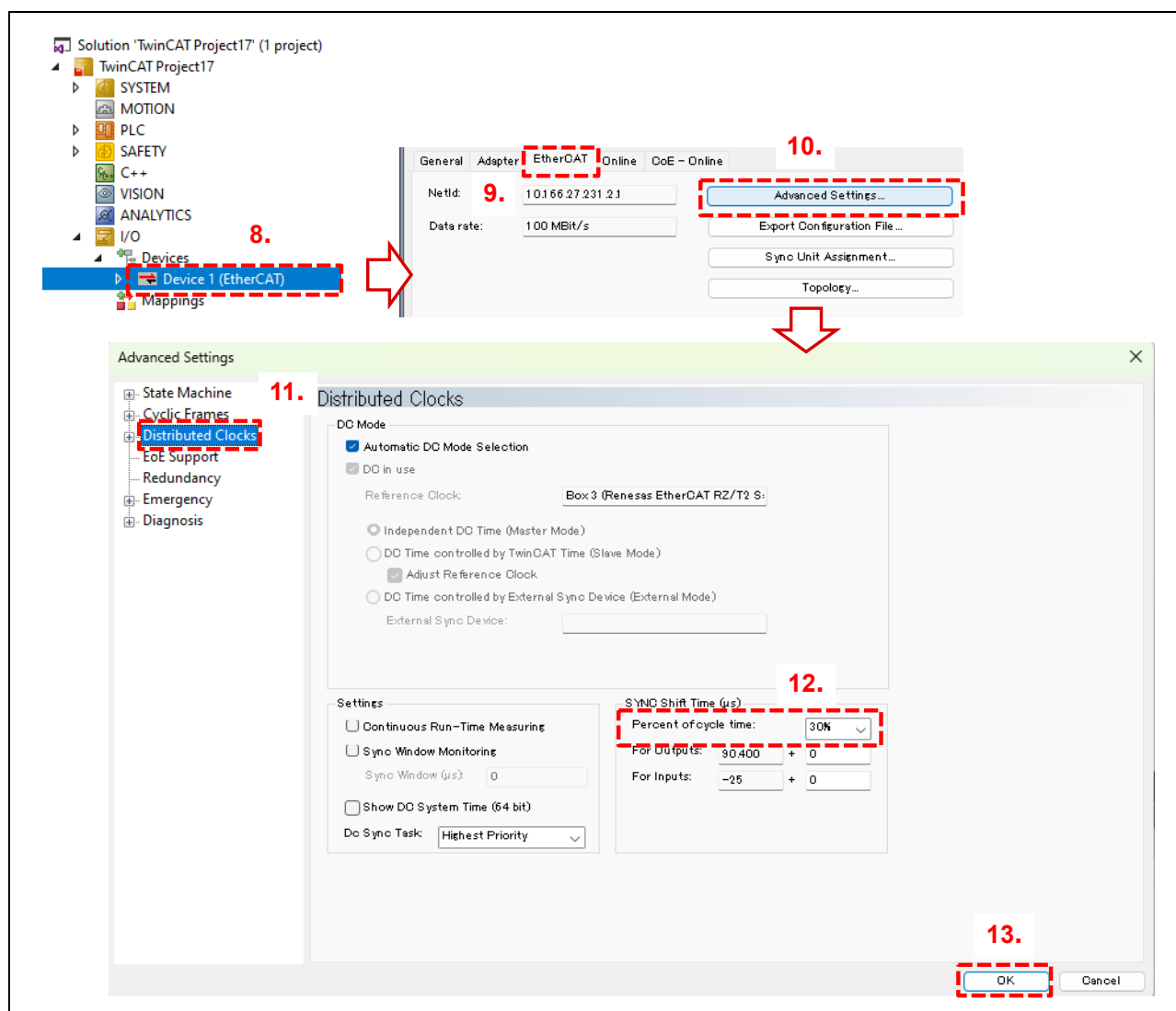


図 6-16. DC 同期モードの設定 2

(3) 次に、EtherCAT ネットワークの設定を有効化します。

1. TwinCAT3 のシステムマネージャツリーで[I/O]→[Devices]→[Device X (EtherCAT)]を順にクリックし、プロパティ画面を開きます。
2. [EtherCAT]タブをクリックします。
3. [Sync Unit Assignment...]をクリックし、Sync Unit Assignment ウィンドウを開きます。
4. Task が<default>に設定されている場合、PlcTask に変更します。
5. [Apply]をクリックした後、[OK]をクリックします。

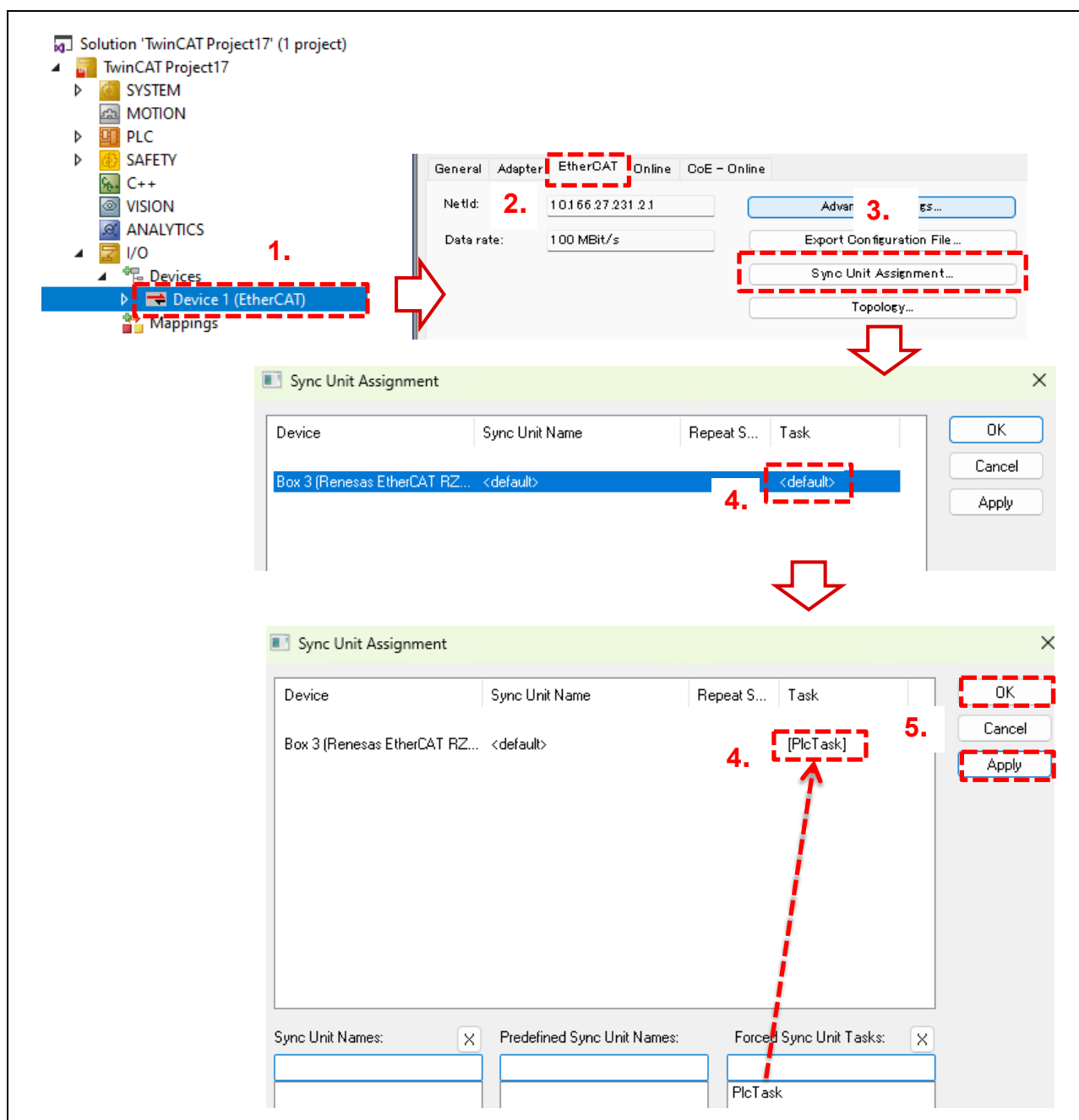


図 6-17. Sync Unit Assignment 設定

6. TwinCAT3 の[Activate Configuration]アイコンをクリックします。
7. "Activate Configuration"ウィンドウが表示されるので、[OK]をクリックします。
8. 続いて、"Restart TwinCAT System in Run Mode"ウィンドウが表示されるので、[OK]をクリックし、TwinCAT3 プロジェクトを Run Mode に切り替えます。

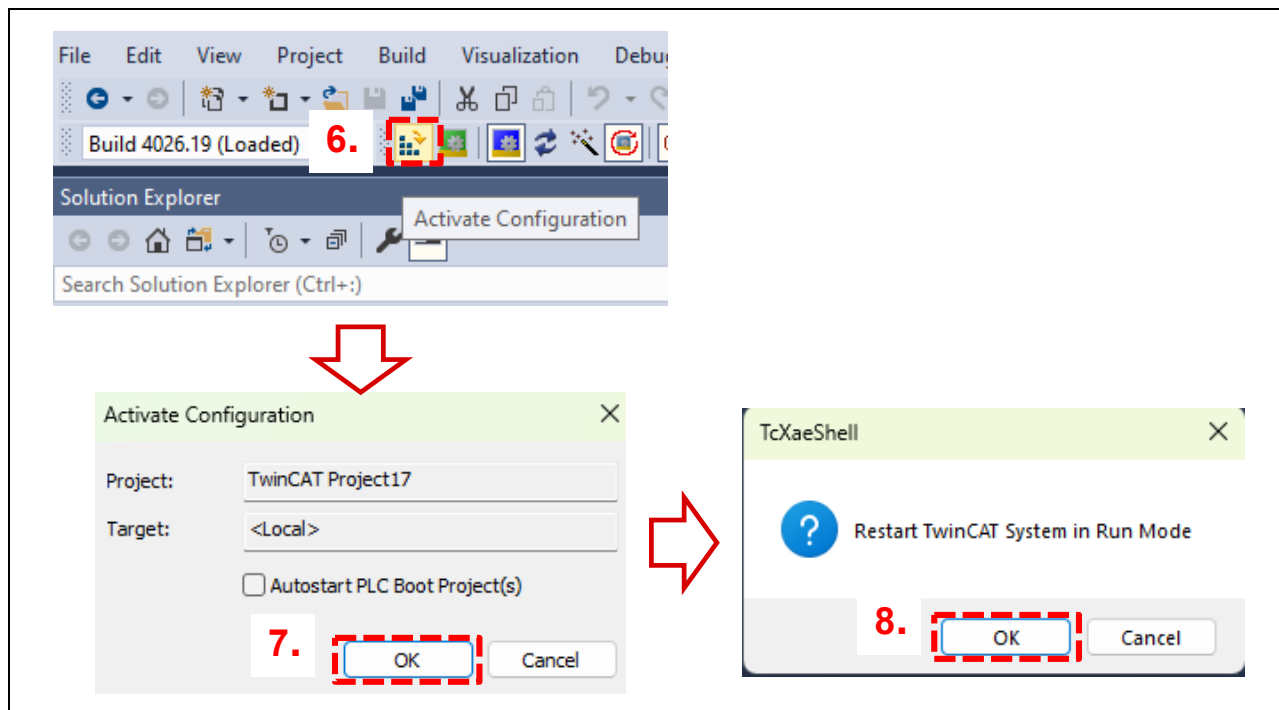


図 6-18. Run mode の開始

(4) 次に、TwinCAT3 と RZ/T2N EVB の接続状態を確認します。

1. TwinCAT3 のシステムマネージャツリーで、[Box X (Renesas EtherCAT RZ/T2N MotorSolution)] をダブルクリック、プロパティ画面を開きます。
2. [Online] タブをクリックします。
3. "Current Status" が [OP] になっていれば、TwinCAT3 と RZ/T2N EVB が接続されている状態です。

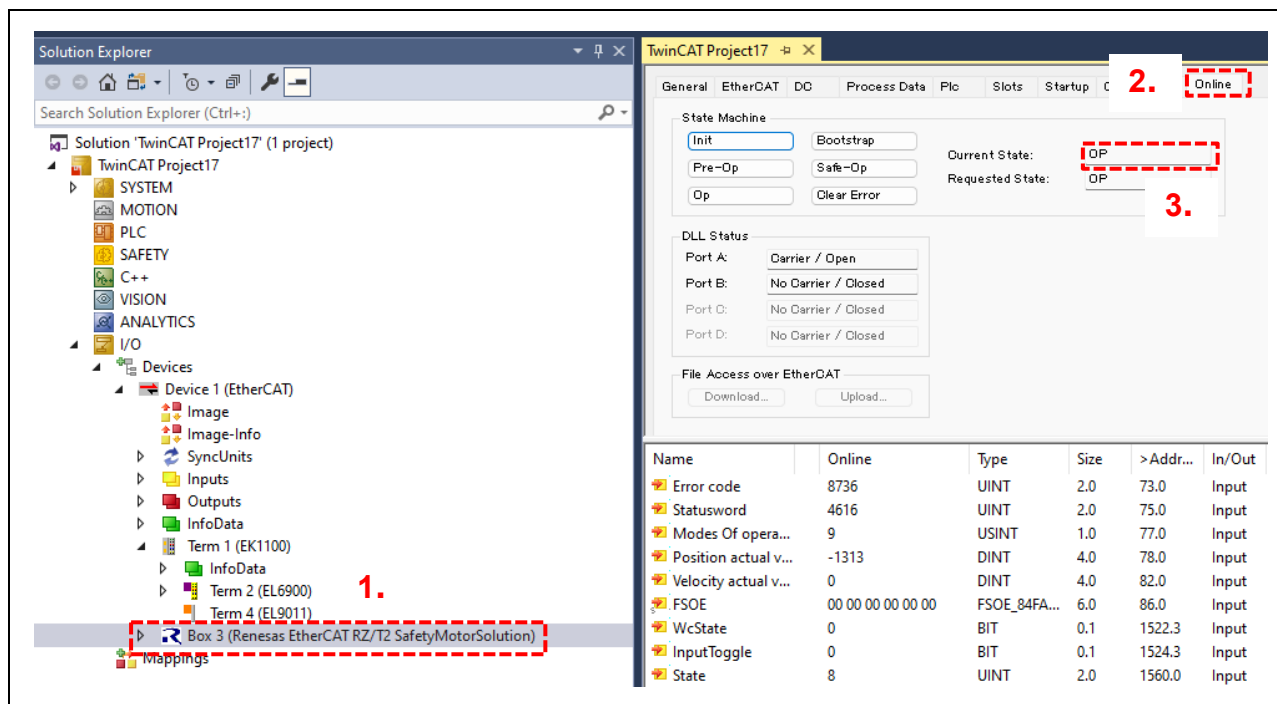


図 6-19. TwinCAT3 と RZ/T2N EVB の接続状態の確認

6.3 動作確認方法

本章では、本サンプルプログラムの動作確認手順を説明します。

6.3.1 CiA402 動作モードの選択

- (1) TwinCAT3 のシステムマネージャツリーで、[Box X (Renesas EtherCAT RZ/T2N MotorSolution)] をダブルクリック、プロパティ画面を開きます。
- (2) [Slots] タブをクリックします。
- (3) [Slots] タブの左画面の[Motor1]行をクリックします。
- (4) [Slots] タブの右画面から CiA402 動作モードをクリックして選択します。
- (5) [<] ボタンを押して CiA402 動作モードを変更します。
- (6) [Slots] タブの左画面の[Motor1]行を見て、選択した CiA402 動作モードになっていることを確認します。
- (7) [Motor2], [Motor3], [Motor4] についても同様の手順で CiA402 動作モードを変更可能です。
- (8) TwinCAT3 の[Activate Configuration]アイコンをクリックし、Run mode の再起動を行ってください。

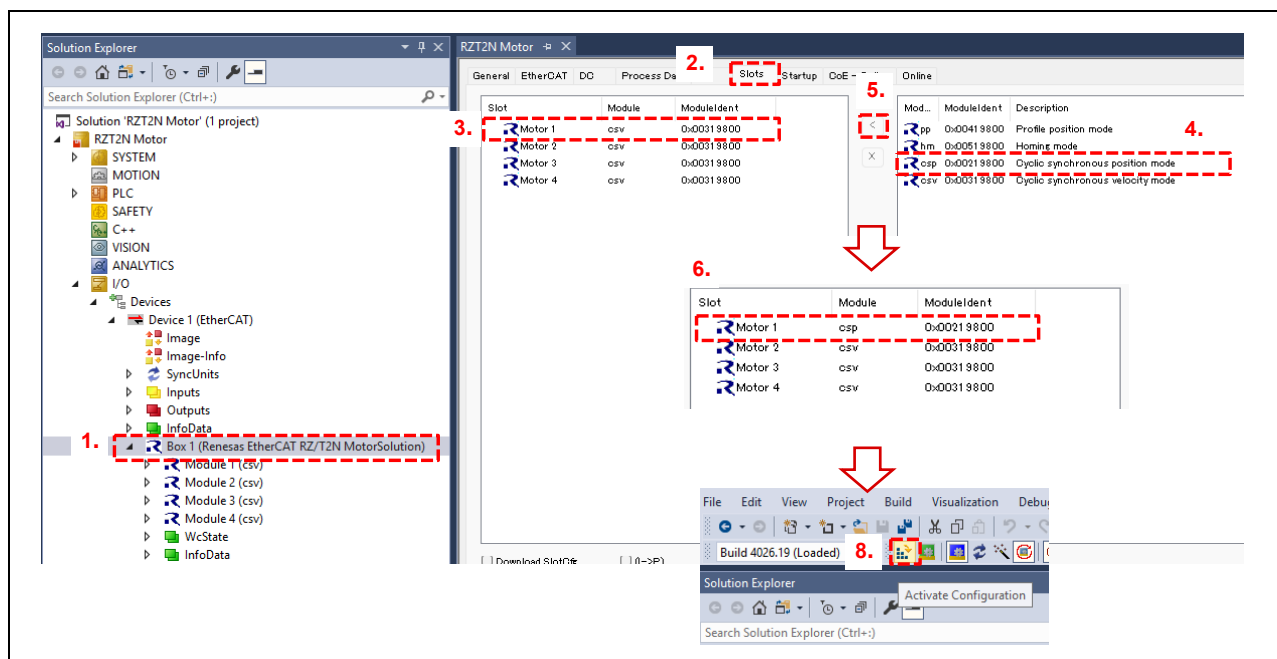


図 6-20. CiA402 動作モードの変更

6.3.2 CiA402 State Machine 制御方法

6.3.2.1 Operation enable への遷移方法

1. TwinCAT3 のシステムマネージャツリーで、[Box X (Renesas EtherCAT RZ/T2N MotorSolution)]→[Module 1]→[Inputs]→[Statusword]をダブルクリックします。
2. [Flags]タブを開きます。
3. [Statusword]の Display mode を Hex に変更します。
4. [Online]タブを開きます。
5. [Statusword]の 3bit 目が 0 であることを確認します。1 である場合は Fault 状態ですので、Fault から復帰する処理を行う必要があります。6.3.2.2 章の処理を行い、Fault から復帰してください。
6. [Module 2] ~ [Module 4]に対しても同様の手順を行うことでモータ 4 軸分の[Statusword]を確認可能です。

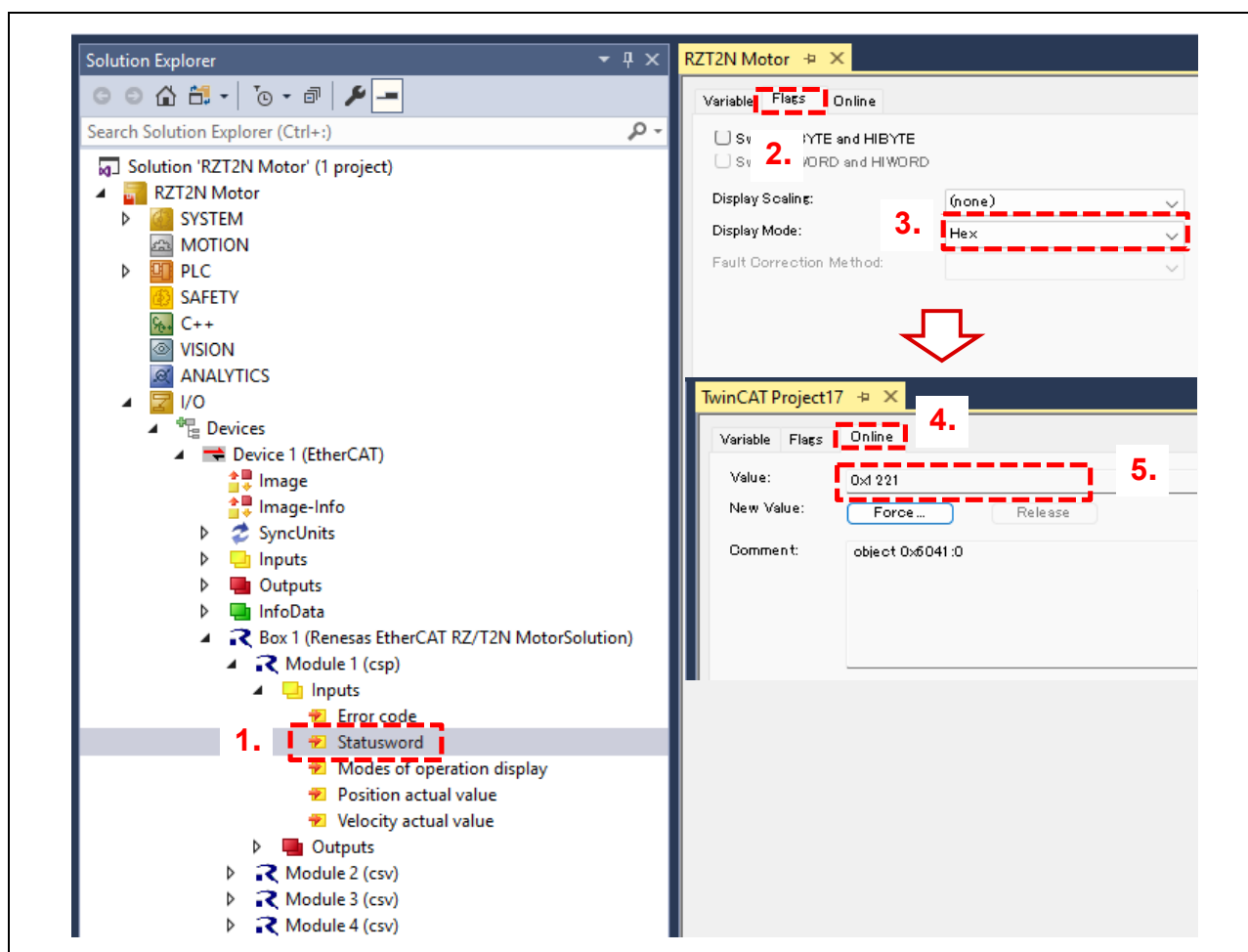


図 6-21. Operation enable への遷移方法 1

7. TwinCAT3 のシステムマネージャツリーで、[Box X (Renesas EtherCAT RZ/T2N MotorSolution)]→[Module 1]→[Outputs]→[Controlword]をクリックします。
8. [Online]タブを開きます。
9. [Write]ボタンを押します。
10. “15”を書き込みます。
11. [Statusword]の値を確認します。2bit 目が1になっていれば Operation enable への遷移が成功しています。
12. [Module 2]～[Module 4]に対しても同様の手順を行い、全てのモータを Operation enable へ遷移させます。

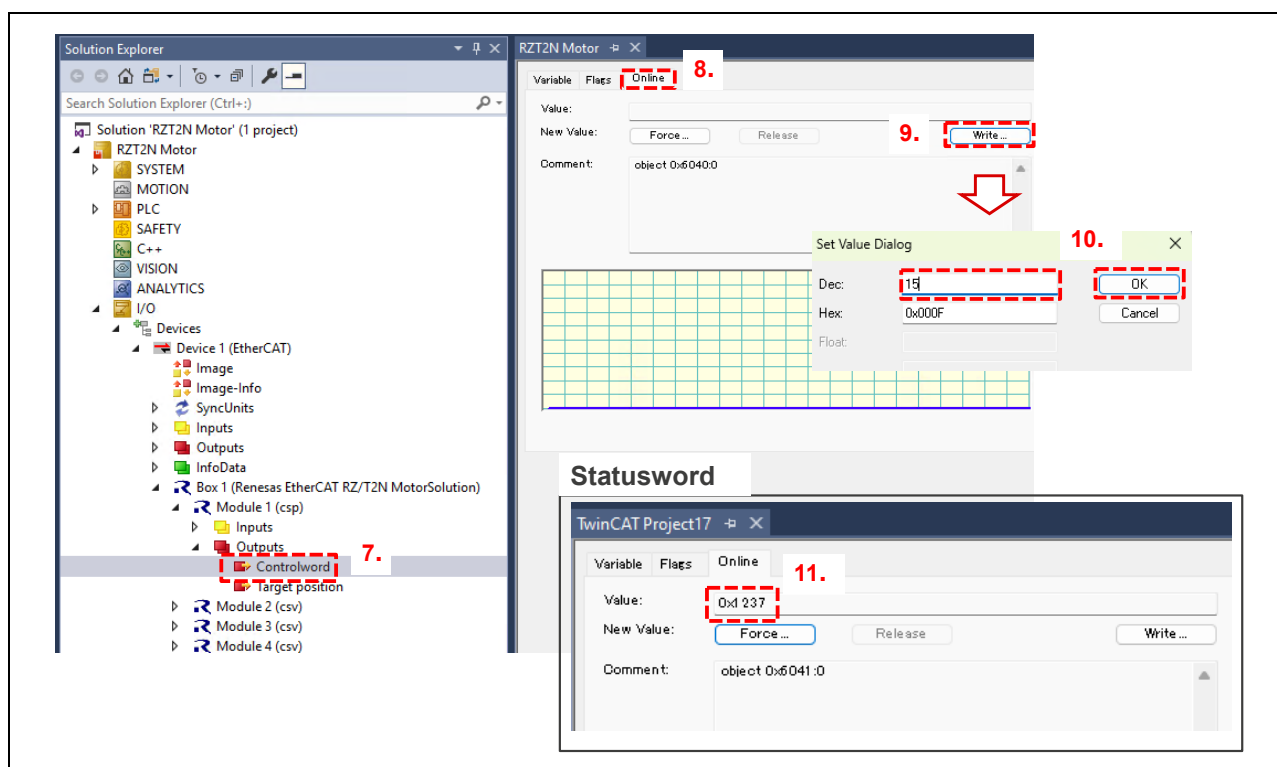


図 6-22. Operation enable への遷移方法 2

13. Operation enable から元の状態に戻したい場合、[Controlword]に“6”を書き込みます。

6.3.2.2 Fault からの復帰方法

1. [Statusword]の 3bit 目が 1 である場合は Fault 状態です。

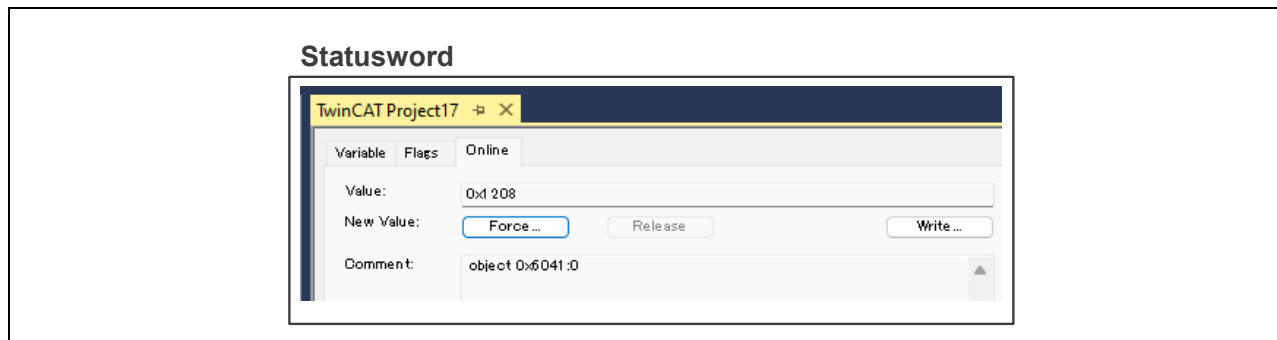


図 6-23. Fault 状態の判別方法

2. TwinCAT3 のシステムマネージャツリーで、[Box X (Renesas EtherCAT RZ/T2N MotorSolution)]→[Module 1]→[Outputs]→[Controlword]をクリックします。
3. [Online]タブを開きます。
4. [Write]ボタンを押します。
5. 128 を書き込みます。
6. [Statusword]の 3bit 目が 0 であることを確認します。これで Fault からの復帰は完了です。
7. [Module 2] ~ [Module 4]でも同様の手順で Fault から復帰可能です。

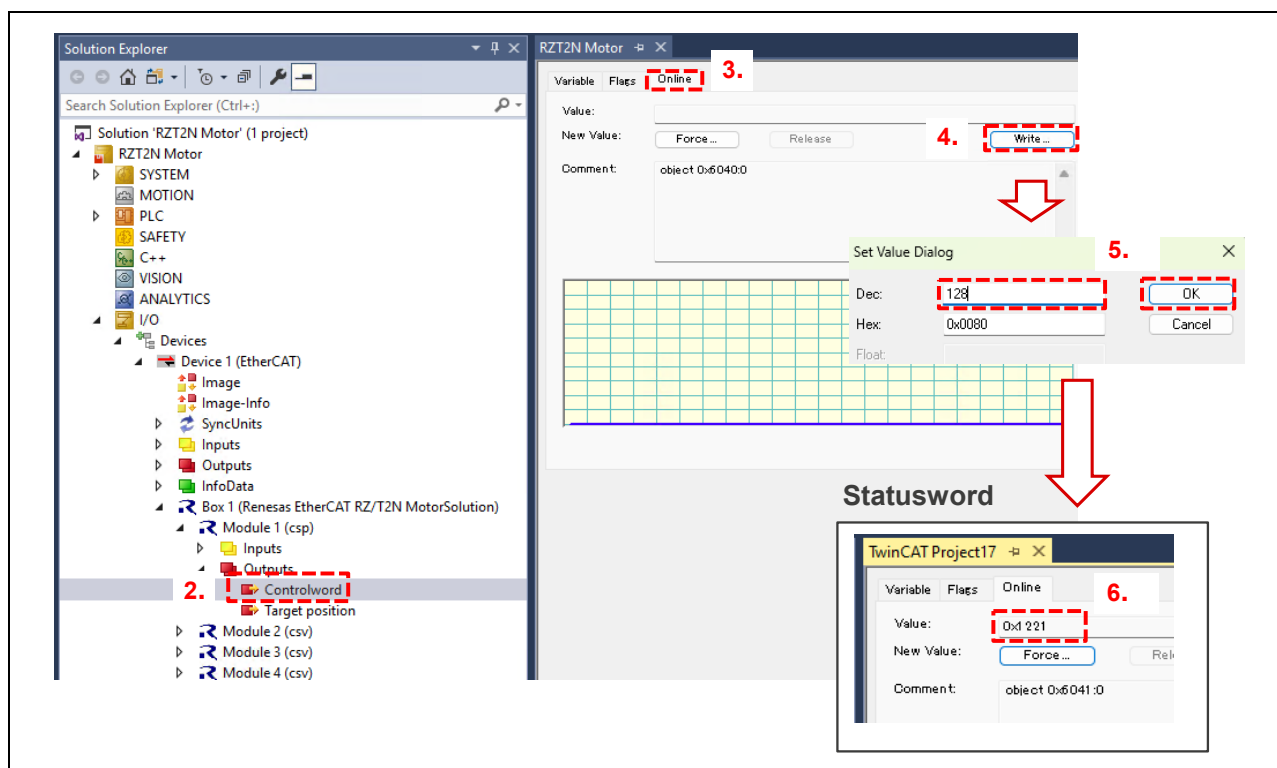


図 6-24. Fault からの復帰方法

6.3.3 モータ動作方法

6.3.3.1 csp mode

1. 6.3.1 章の手順を行い、動作させたいモータの CiA402 動作モードを csp mode にします。
2. 6.3.2 章の手順を行い、CiA402 ステートマシンを Operation enable にします。
3. TwinCAT3 のシステムマネージャツリーで、[Target position]をクリックします。
4. [Online]タブを開き、[Write]ボタンを押します。
5. 36000 を書き込み、3600.0 deg に移動するようモータへ位置指令を出します。
6. モータが回転し、3600.0 deg に移動します。
7. TwinCAT3 のシステムマネージャツリーで、[Position actual value]をクリックします。
8. [Online]タブを開き、値が 36000 になっていることを確認します。

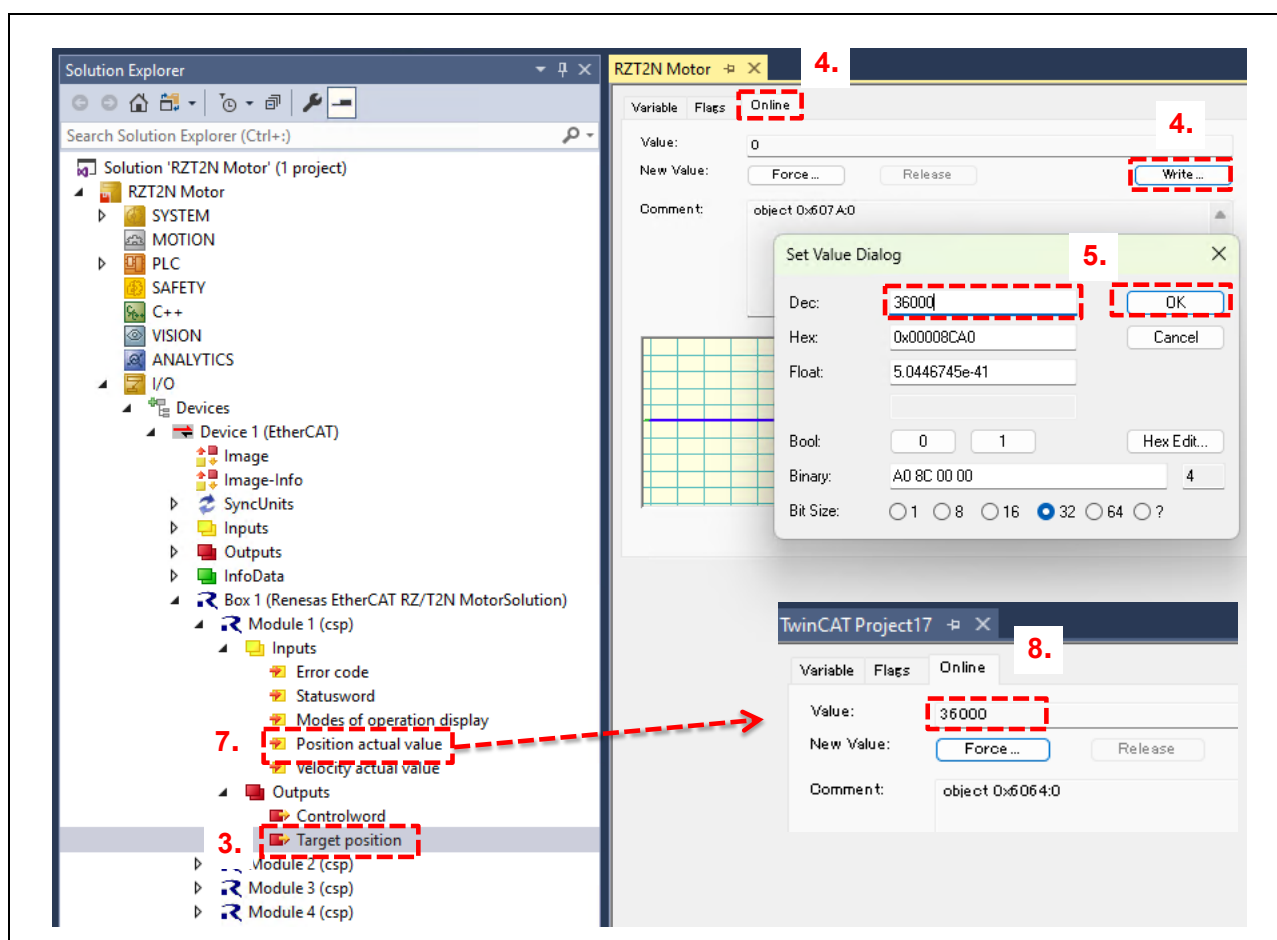


図 6-25. csp モードの動作確認

6.3.3.2 pp mode

1. 6.3.1 章の手順を行い、動作させたいモータの CiA402 動作モードを pp mode にします。
2. 6.3.2 章の手順を行い、CiA402 ステートマシンを Operation enable にします。
3. TwinCAT3 のシステムマネージャツリーで、[Profile velocity]をクリックします。
4. [Online]タブを開き、[Write]ボタンを押します。
5. 30000 を書き込み、速度プロファイルを 3000.0 rpm に設定します。
6. TwinCAT3 のシステムマネージャツリーで、[Profile acceleration]をクリックします。
7. [Online]タブを開き、[Write]ボタンを押します。
8. 150000 を書き込み、加速度プロファイルを 15000.0 rpm/s に設定します。

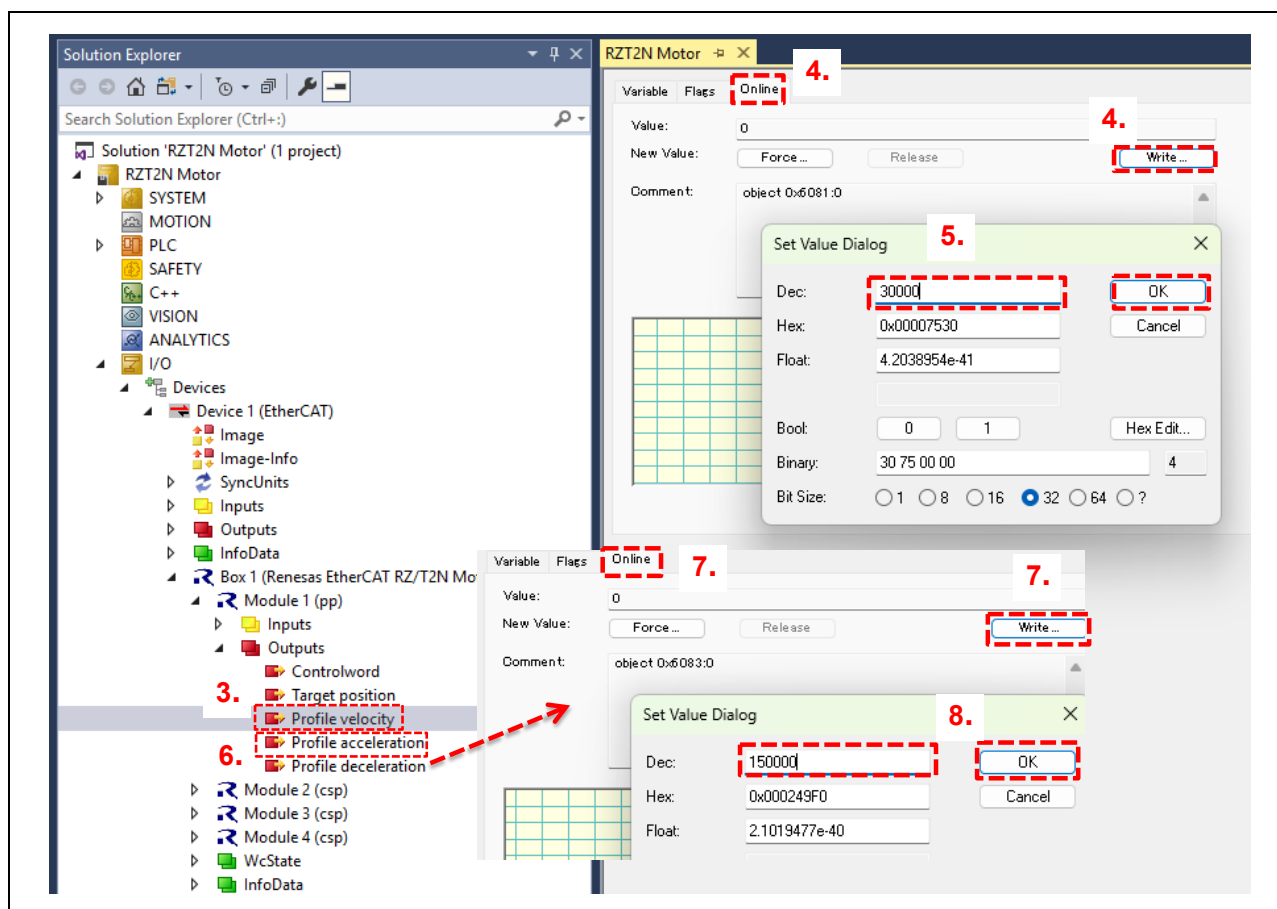


図 6-26. pp モードの動作確認 1

9. TwinCAT3 のシステムマネージャツリーで、[Target position]をクリックします。
10. [Online]タブを開き、[Write]ボタンを押します。
11. 36000 を書き込み、3600.0 deg に移動するようモータへ位置指令を出します。
12. 設定したプロファイルでモータが回転し、3600.0 deg に移動します。
13. TwinCAT3 のシステムマネージャツリーで、[Position actual value]をクリックします。
14. [Online]タブを開き、値が 36000 になっていることを確認します。

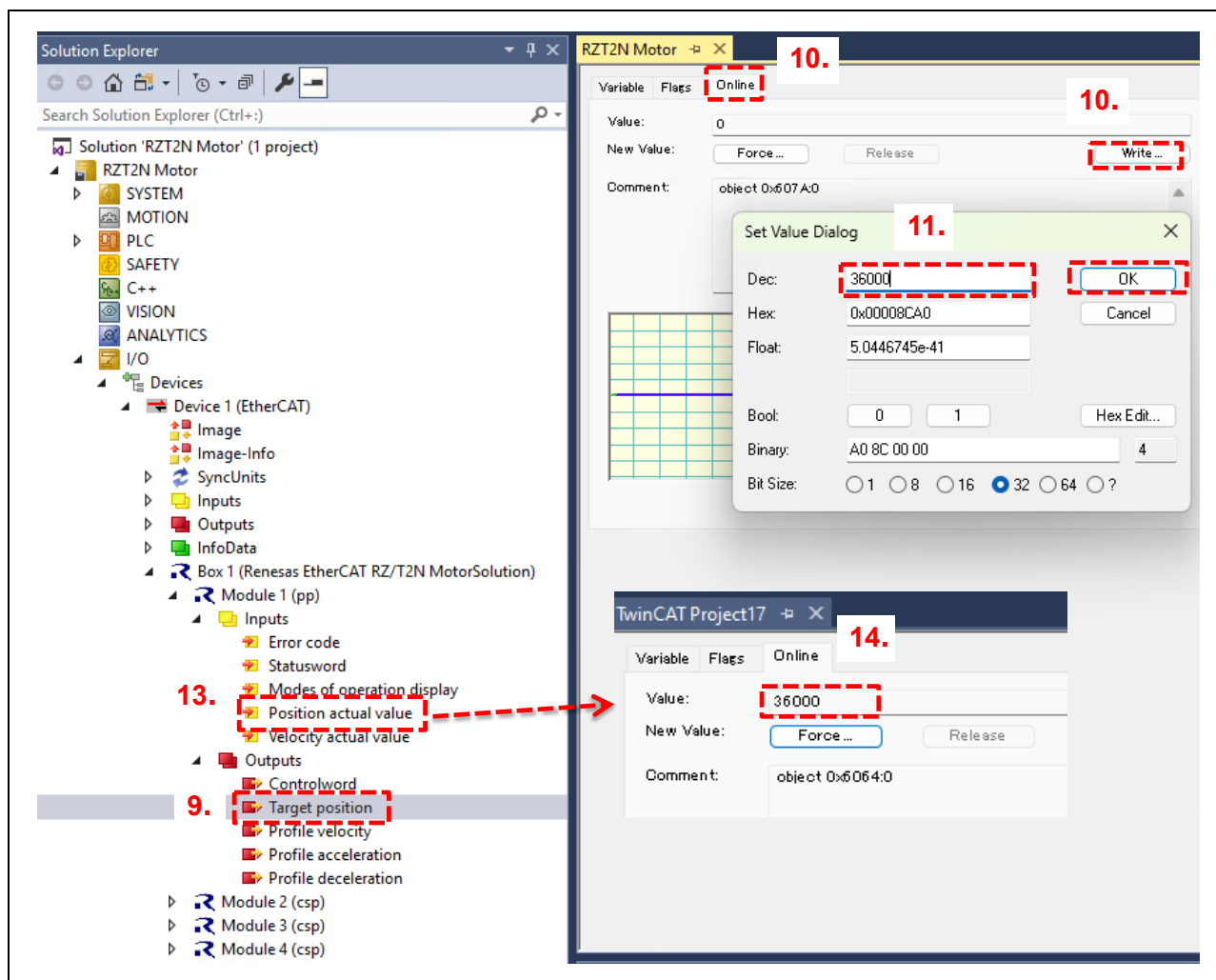


図 6-27. pp モードの動作確認 2

6.3.3.3 csv mode

1. 6.3.1 章の手順を行い、動作させたいモータの CiA402 動作モードを csv mode にします。
2. 6.3.2 章の手順を行い、CiA402 ステートマシンを Operation enable にします。
3. TwinCAT3 のシステムマネージャツリーで、[Target velocity]をクリックします。
4. [Online]タブを開き、[Write]ボタンを押します。
5. 20000 を書き込み、2000rpm で回転するようモータへ速度指令を出します。
6. 2000rpm でモータが回転します。
7. TwinCAT3 のシステムマネージャツリーで、[Velocity actual value]をクリックします。
8. [Online]タブを開き、値が 20000 付近になっていることを確認します。

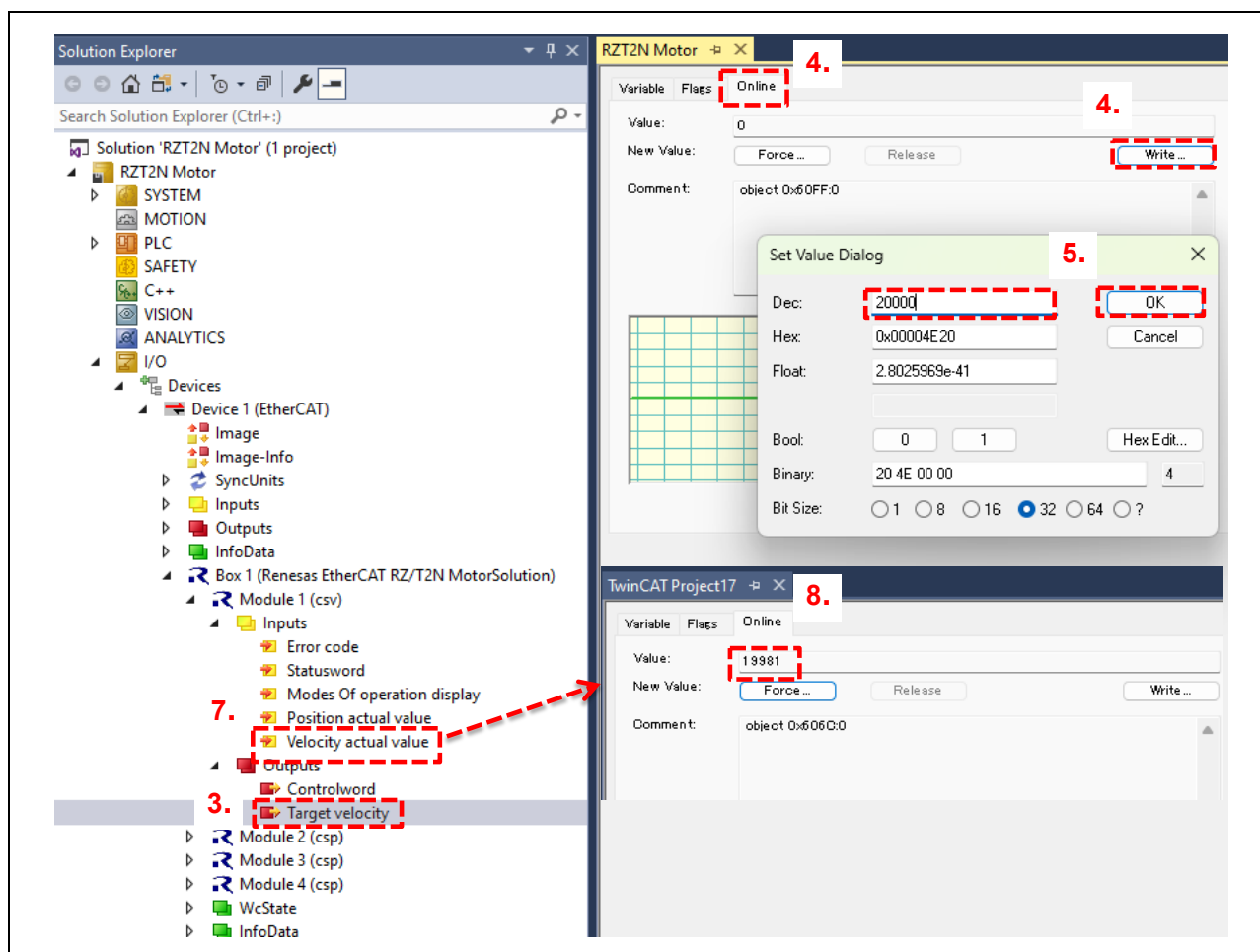


図 6-28. csv モードの動作確認

6.3.3.4 hm mode

1. 6.3.1 章の手順を行い、動作させたいモータの CiA402 動作モードを hm mode にします。
2. 6.3.2 章の手順を行い、CiA402 ステートマシンを Operation enable にします。
3. TwinCAT3 のシステムマネージャツリーで、[Position actual value]をクリックします。
4. [Online]タブを開き、現在の値を確認します。
5. TwinCAT3 のシステムマネージャツリーで、[Home offset]をクリックします。
6. “0”になっていることを確認します。

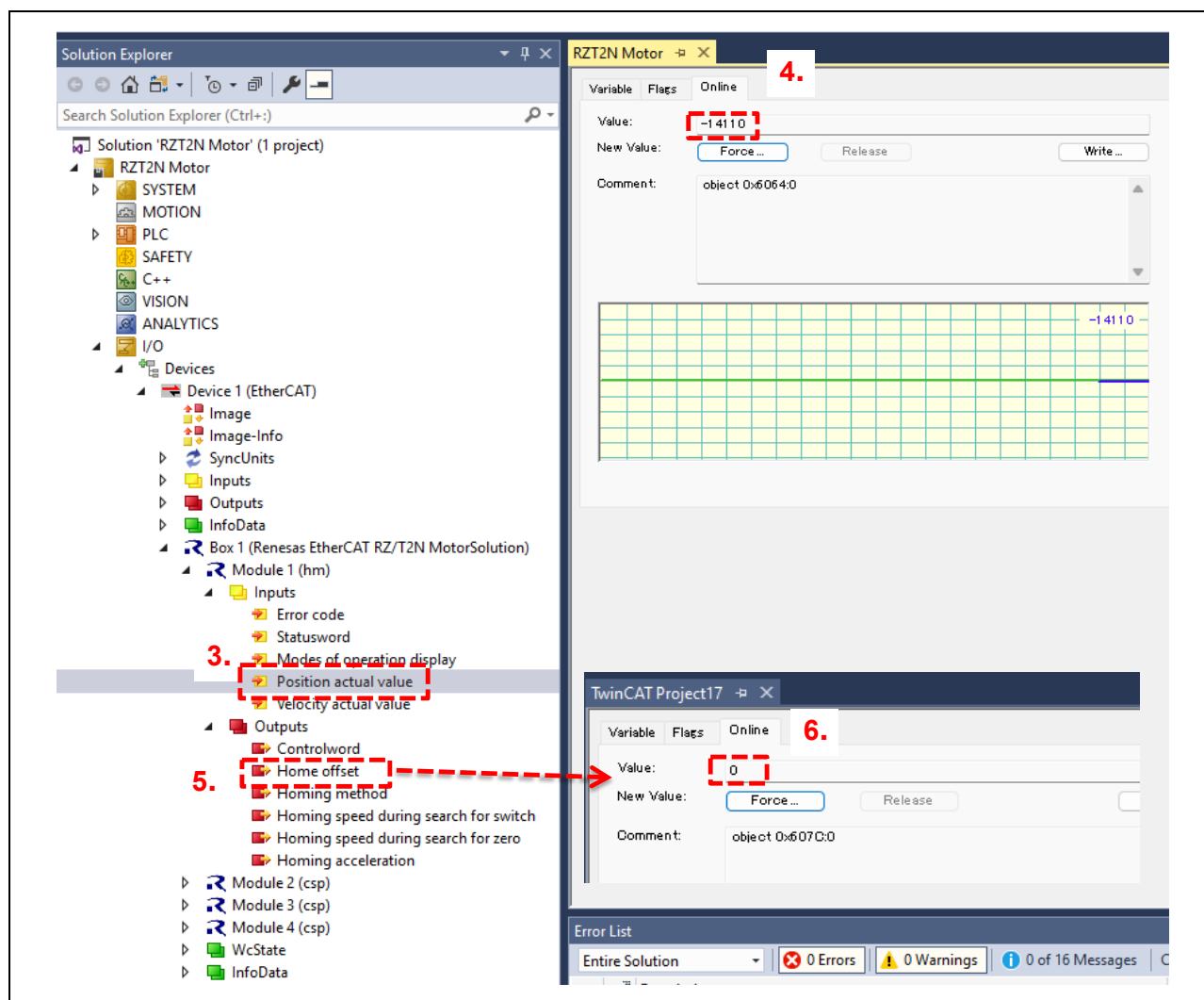


図 6-29. hm モードの動作確認 1

7. TwinCAT3 のシステムマネージャツリーで、[Home method]をクリックします。
8. [Online]タブを開き、[Write]ボタンを押します。
9. 37 を書き込み、Home Method 37 を指定します。

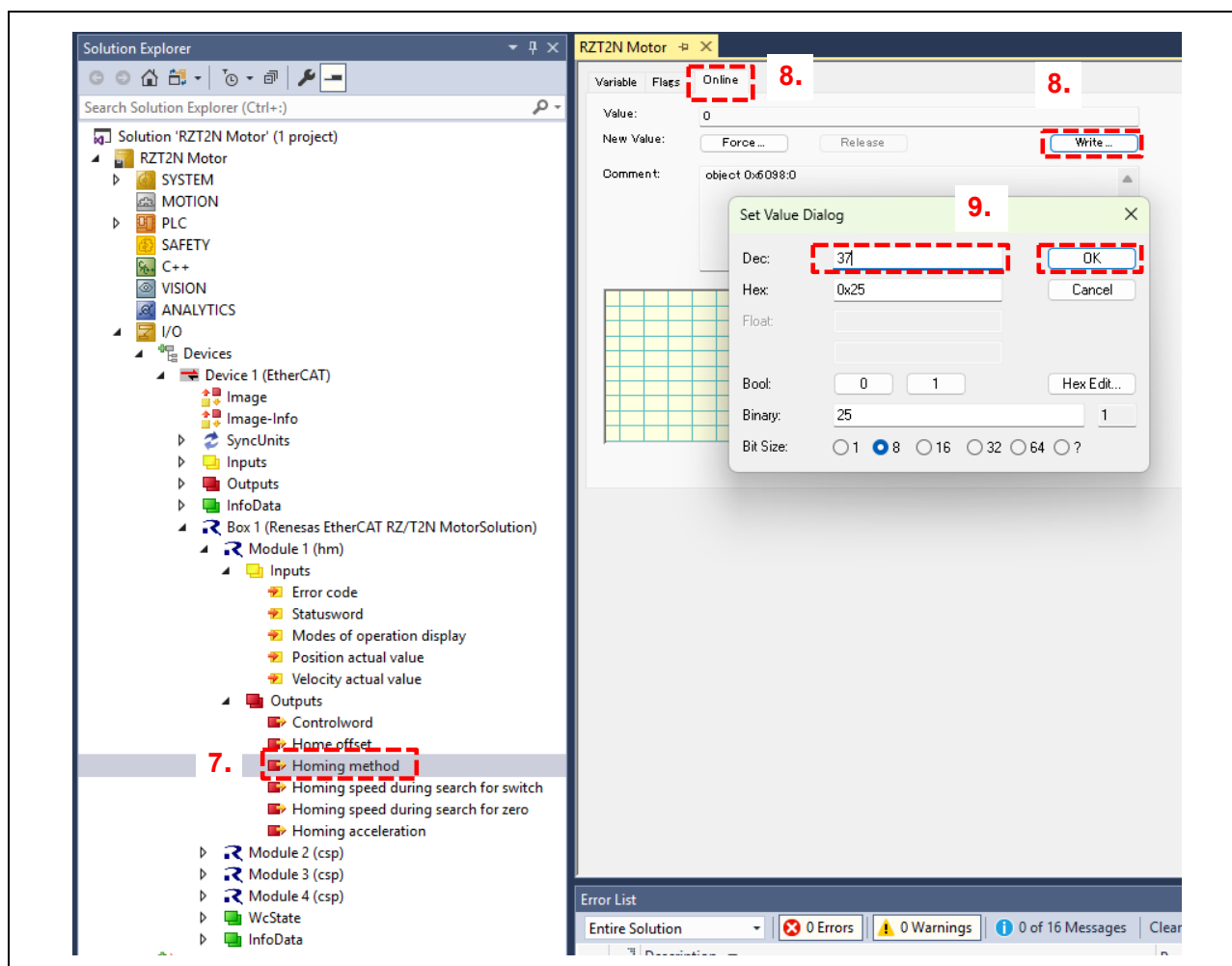


図 6-30. hm モードの動作確認 2

10. TwinCAT3 のシステムマネージャツリーで、[Controlword]をクリックします。
11. [Online]タブを開き、[Write]ボタンを押します。
12. 31 を書き込み、Home Method 37 を実行します。
13. TwinCAT3 のシステムマネージャツリーで、[Position actual value]をクリックします。
14. [Online]タブを開き、値が 0 に変更されていることを確認します。

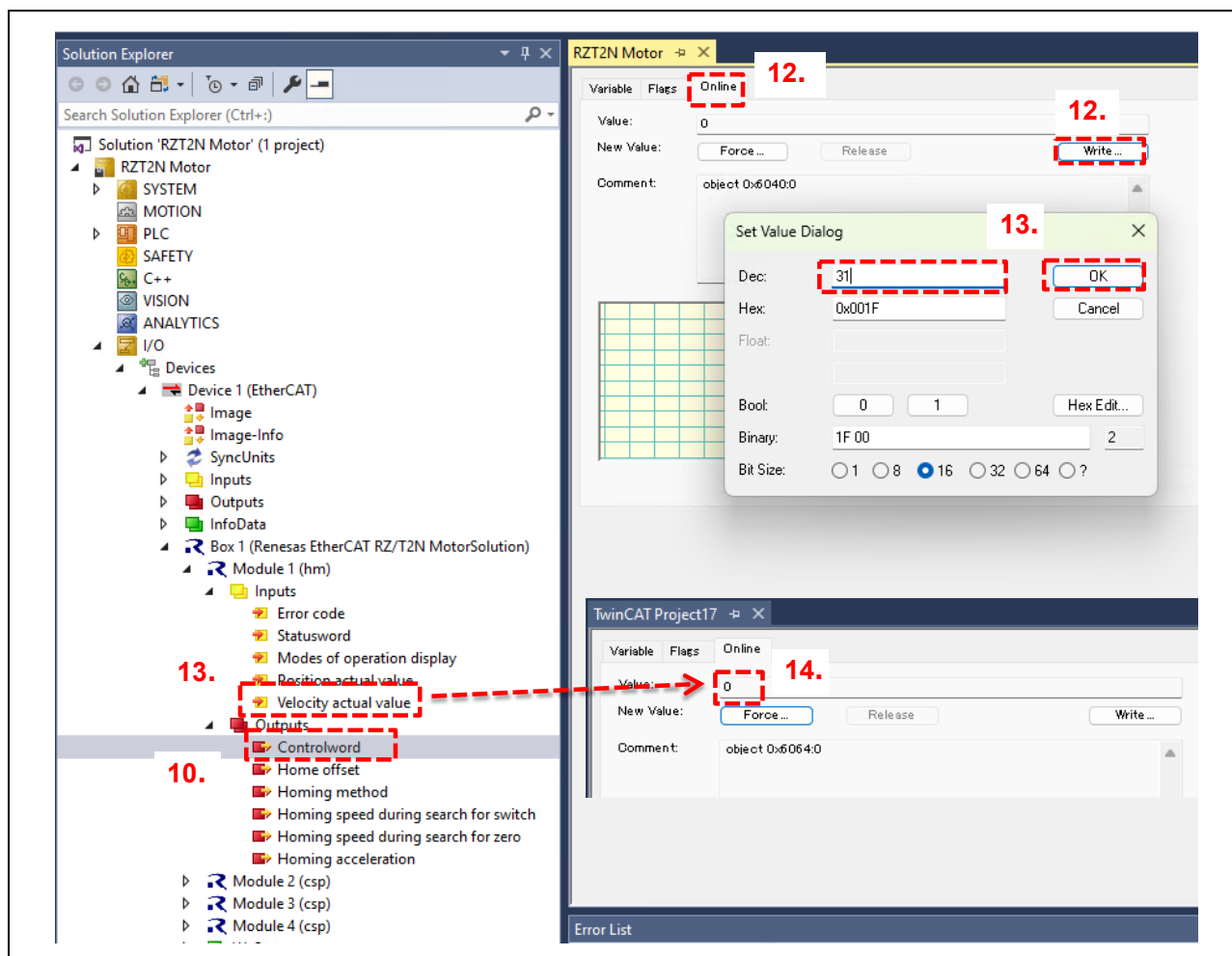


図 6-31. hm モードの動作確認 3

7. ソフトウェア仕様

7.1 ソフトウェア構成図

図 7-1 に本サンプルプログラムのソフトウェア構成図を示します。

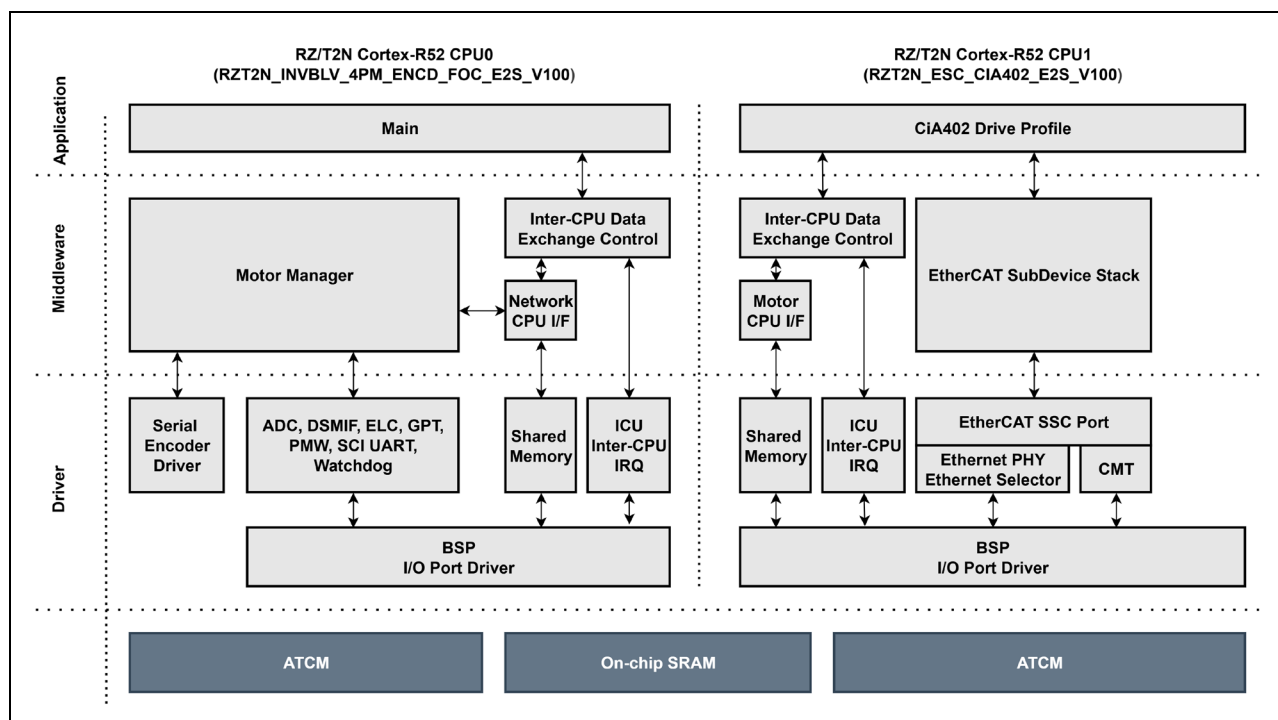


図 7-1. 本サンプルプログラムのソフトウェア構成図

7.2 Motor Control(RZ/T2N Cortex-R52 CPU0)

7.2.1 ソフトウェア全体構成

本サンプルプログラムの RZ/T2N CPU0 ソフトウェアは、『RZ/T2N 永久磁石同期モータのエンコーダベクトル制御 (シリアルエンコーダ) - 4 モータ駆動(R01AN8225)』をベースに作成しています。

ベースプロジェクトの詳細は R01AN8225 のアプリケーションノートを参照してください。

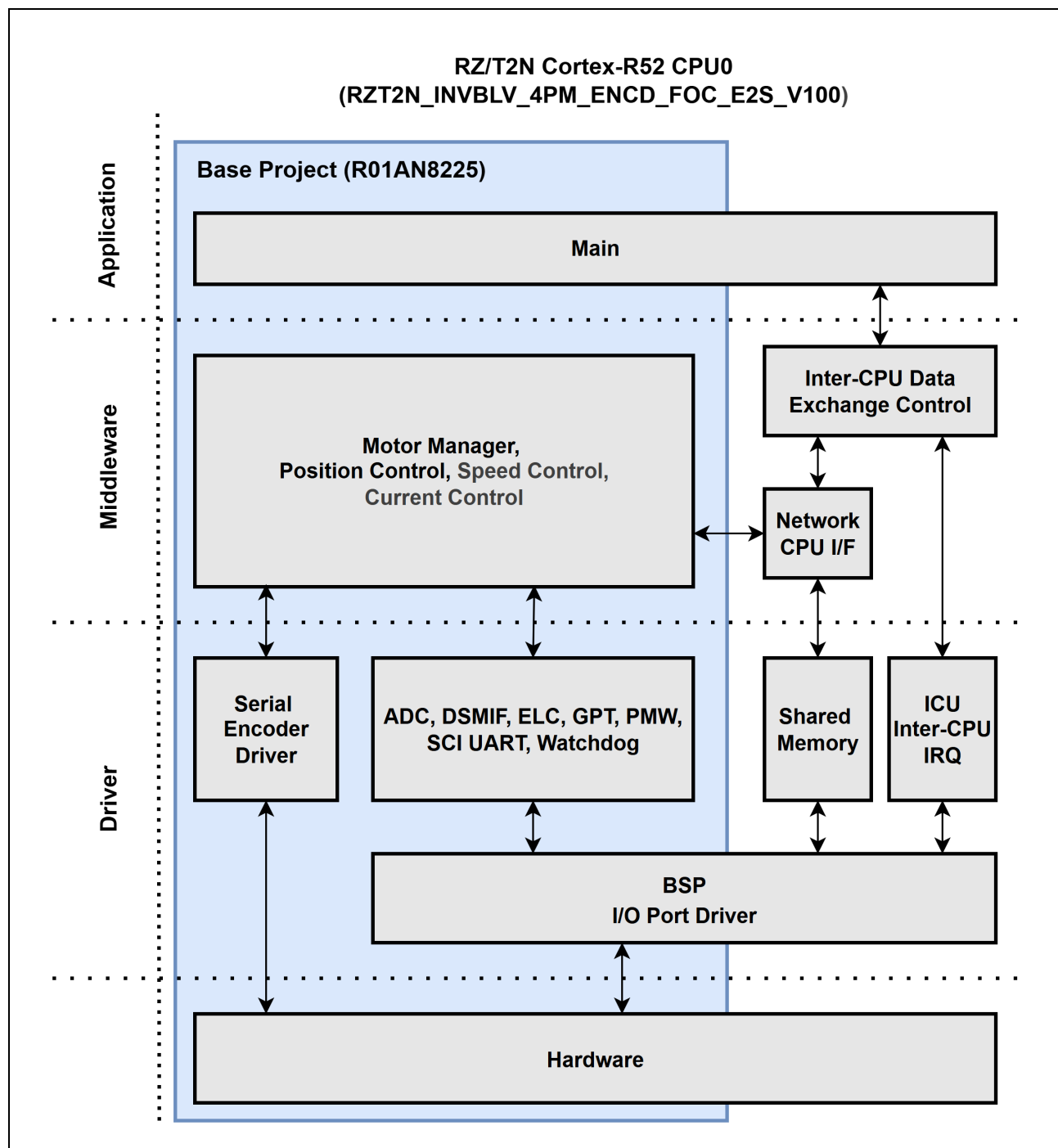


図 7-2. RZ/T2N Cortex-R52 CPU0 ソフトウェア全体構成

7.2.2 ベースプロジェクトからの変更点

ここでは、ベースプロジェクトからの変更点について記述します。

表 7-1. ベースプロジェクトからの変更点

変更のあったフォルダ・ファイル			変更内容
rz			Configuration.xml の変更によって、変更が加えられたファイルや追加されたファイル・フォルダがある
rz_gen			
rz_cfg			
src	network_cpu_if	r_network_cpu_if.c	Network CPU I/F モジュールの追加 (詳細は 7.2.3 章を参照)
		r_network_cpu_if.h	
	cpu0_exchange_ctrl	cpu0_exchange_ctrl.c cpu0_exchange_ctrl.h	CPU0 CPU 間データ交換制御モジュールの追加 (詳細は 7.2.4 章を参照)
	hal_entry.c		hal_entry 関数内の処理を変更 CPU0_ExchangeCtrl_Init()と CPU0_ExchangeCtrl_Main()の呼び出しを追加 (詳細は図 7-3 を参照)
configuration.xml			Shared Memory Driver の追加 (設定は表 7-2, 表 7-3, 表 7-4 を参照) CPU 間データ交換制御用割り込み設定の追加 (設定は表 7-5, 表 7-6 を参照) CPU1 クロックを 800MHz に設定
.cproject			“src/network_cpu_if”インクルードパスの追加 “src/cpu0_exchange_ctrl” インクルードパスの追加 RZ/T2N CPU1 オブジェクトファイルの追加

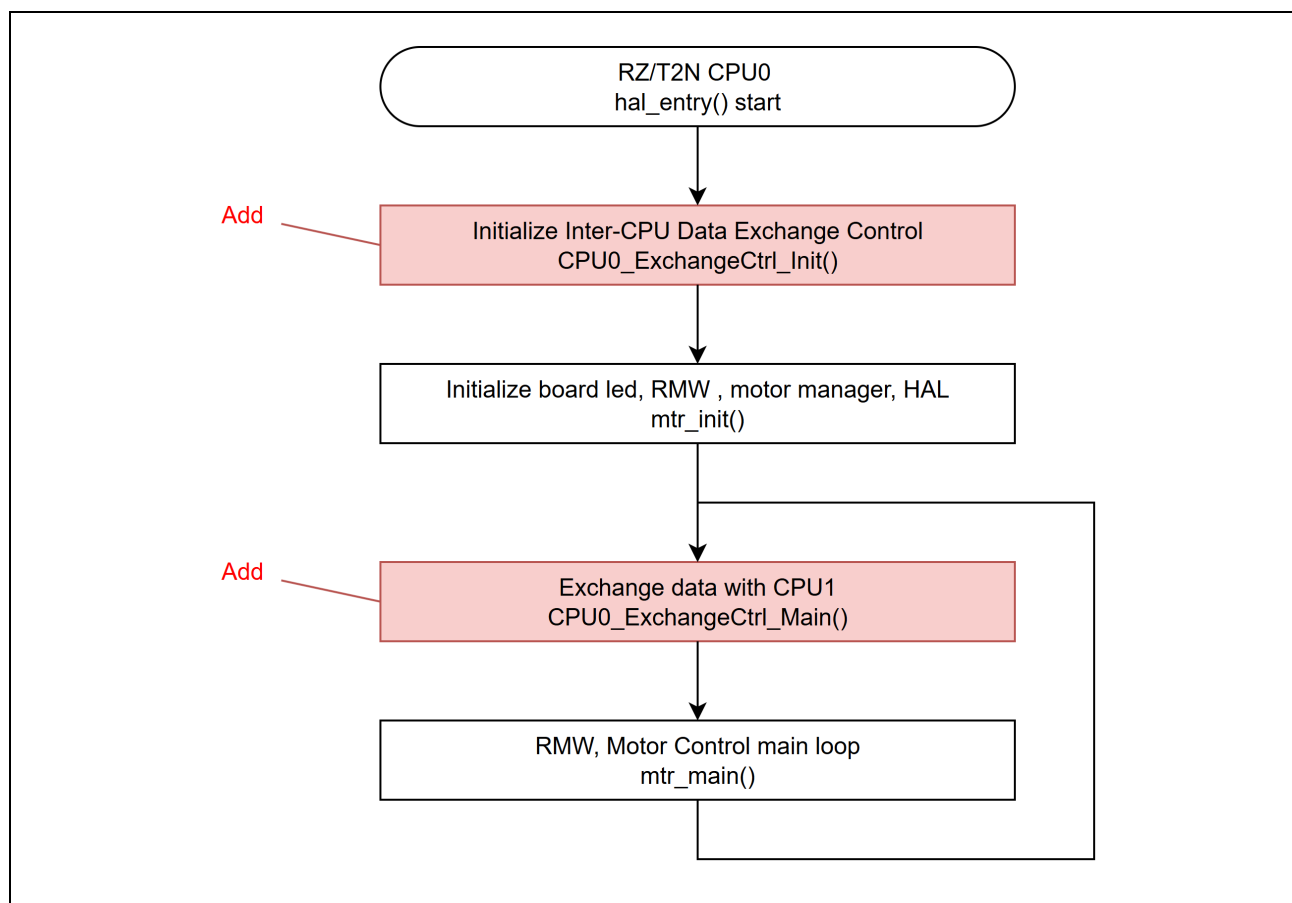


図 7-3. hal_entry 関数フローチャート

表 7-2. 追加した Shared Memory Driver N 設定 (r_shared_memory, N=0,1,2,3)

プロパティ	設定値
Common - Parameter Checking	Default (BSP)
Module - Name	g_shared_memoryN
Module - Shared Memory Address	0x101E0000 + 0x200N
Module - Shared Memory Size	0x1FF
Module - Semaphore	N
Module - Callback	shared_memory_callback

表 7-3. 追加した Shared Memory Driver N 設定 (r_icu_inter_cpu_irq, tx, X=2N)

プロパティ	設定値
Common - Parameter Checking	Default (BSP)
Common - Multiplex Interrupt	Disabled
Module - Name	g_icu_inter_cpu_irqX
Module - Channel	X
Module - Software Interrupt Priority	Disabled
Module - Callback	r_shared_memory_callback

表 7-4. 追加した Shared Memory Driver N 設定 (r_icu_inter_cpu_irq, rx, Y=2N+1)

プロパティ	設定値
Common - Parameter Checking	Default (BSP)
Common - Multiplex Interrupt	Disabled
Module - Name	g_icu_inter_cpu_irqY
Module - Channel	Y
Module - Software Interrupt Priority	Priority 6
Module - Callback	r_shared_memory_callback

表 7-5. 追加した CPU 間データ交換制御 設定 (r_icu_inter_cpu_irq, ch8)

プロパティ	設定値
Common - Parameter Checking	Default (BSP)
Common - Multiplex Interrupt	Disabled
Module - Name	g_icu_inter_cpu_irq8
Module - Channel	8
Module - Software Interrupt Priority	Disabled
Module - Callback	NULL

表 7-6. 追加した CPU 間データ交換制御 設定 (r_icu_inter_cpu_irq, ch9)

プロパティ	設定値
Common - Parameter Checking	Default (BSP)
Common - Multiplex Interrupt	Disabled
Module - Name	g_icu_inter_cpu_irq9
Module - Channel	9
Module - Software Interrupt Priority	Priority 6
Module - Callback	CPU0_ExchangeCtrl_IntCallback

7.2.3 Network CPU I/F モジュール

Network CPU I/F モジュールは、Shared Memory Driver を使用して Network CPU (RZ/T2N CPU1)とモータ制御コマンド・指令値・モータステータスの送受信及びモータマネージャ API を呼び出すモジュールです。

本サンプルプログラムではこのモジュールを 4 つ使用することで 4 軸モータに対応しています。

7.2.3.1 モジュール構成

モジュール構成図を図 7-4 に示します。

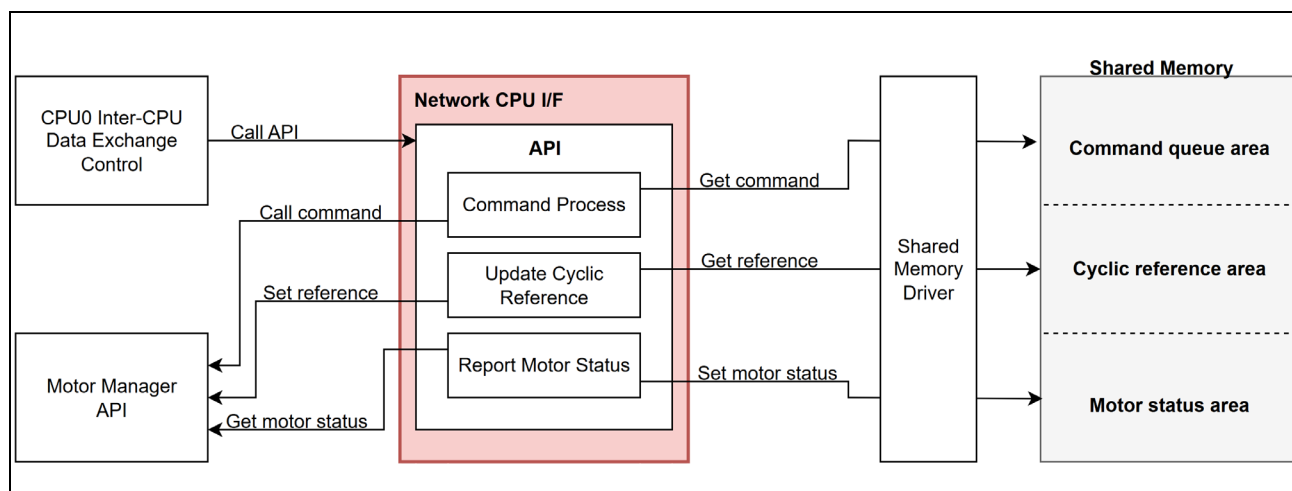


図 7-4. Network CPU I/F モジュール構成図

本モジュールでは主に 3 種類の API を提供します。

(1) Command Process

この API は、Shared Memory の Command queue area にモータ制御コマンドが格納されているか確認します。コマンドが存在する場合、該当するモータマネージャ API を呼び出して処理を実行します。

(2) Update Cyclic Reference

この API は、Shared Memory の Cyclic reference area に設定されている各種指令値が更新されているかを確認します。指令値が Network CPU によって更新されていた場合、該当するモータマネージャ API を呼び出し、新しい指令値をセットします。

(3) Report Motor Status

この API は、モータマネージャ API を呼び出して各種モータステータスを取得し、更新の有無を確認します。ステータスが更新されていた場合、Shared Memory の Motor status area に反映します。

7.2.3.2 Shared Memory 割り当て

各モジュール (N=0,1,2,3) のベースアドレスは 0x101E_0000 + 0x200N です。

表 7-7. Shared Memory 割り当て (Command queue area)

Offset address	名称	サイズ [byte]	使用用途
0x0	Command queue n (n = 0 ... 7)	16 × 8 (=128)	コマンドキュー
0x80	head	4	キューの先頭
0x84	tail	4	キューの末尾

表 7-8. Shared Memory 割り当て (Cyclic reference area)

Offset address	名称	サイズ [byte]	使用用途
0x100	f4_ref_position	4	指令位置
0x104	f4_ref_speed	4	指令速度
0x108	f4_ref_torque	4	指令トルク

表 7-9. Shared Memory 割り当て (Motor status area)

Offset address	名称	サイズ [byte]	使用用途
0x180	f4_pos_deg	4	現在位置
0x184	f4_speed	4	現在速度
0x188	u1_status	1	ステータマシン 0x00 : STOP 0x01 : RUN 0x02 : ERROR
0x189	reserved1	3	未使用領域
0x18C	u2_error_status	2	エラー状態 0x0000 : エラーなし 0x0001 : HW 過電流エラー 0x0002 : 過電圧エラー 0x0004 : 回転速度エラー 0x0080 : 低電圧エラー 0x0100 : SW の過電流エラー 0x0200 : インバータ過熱エ ラー 0xffff : 不明なエラー
0x18E	reserved2	2	未使用領域
0x190	u1_ctrl_loop_mode	1	LOOP MODE 0: d 軸電流制御 1: q 軸電流制御

			2: 速度制御 3: 位置制御
0x191	reserved3	3	未使用領域
0x194	u1_in_position	1	位置制御完了状態 0: 位置制御未完了状態 1: 位置制御完了状態
0x195	reserved4	3	未使用領域
0x198	u1_offset_cal_finished_flag	1	オフセット測定状態 0: 測定中 1: 測定完了
0x199	reserved5	3	未使用領域

7.2.3.3 API

表 7-10. Network CPU I/F API 一覧

API	説明
R_NETWORK_CPU_IF_Open	Network CPU I/F の初期化を行います。 また、Shared memory Driver の初期化を行います。 フローチャート：図 7-5
R_NETWORK_CPU_IF_Close	Network CPU I/F を終了します。 また、Shared memory Driver を終了します。 フローチャート：図 7-6
R_NETWORK_CPU_IF_CommandProcess	Shared Memory の Command queue area にモータ制御コマンドが格納されているか確認します。 コマンドが存在する場合、該当するモータマネージャ API を呼び出して処理を実行します。 フローチャート：図 7-7
R_NETWORK_CPU_IF_UpdateCyclicReference	Shared Memory の Cyclic reference area に設定されている各種指令値が更新されているかを確認します。 指令値が Network CPU によって更新されていた場合、該当するモータマネージャ API を呼び出し、新しい指令値をセットします。 フローチャート：図 7-8
R_NETWORK_CPU_IF_ReportMotorStatus	モータマネージャ API を呼び出して各種モータステータスを取得し、更新の有無を確認します。 ステータスが更新されていた場合、Shared Memory の Motor status area に反映します。 フローチャート：図 7-9

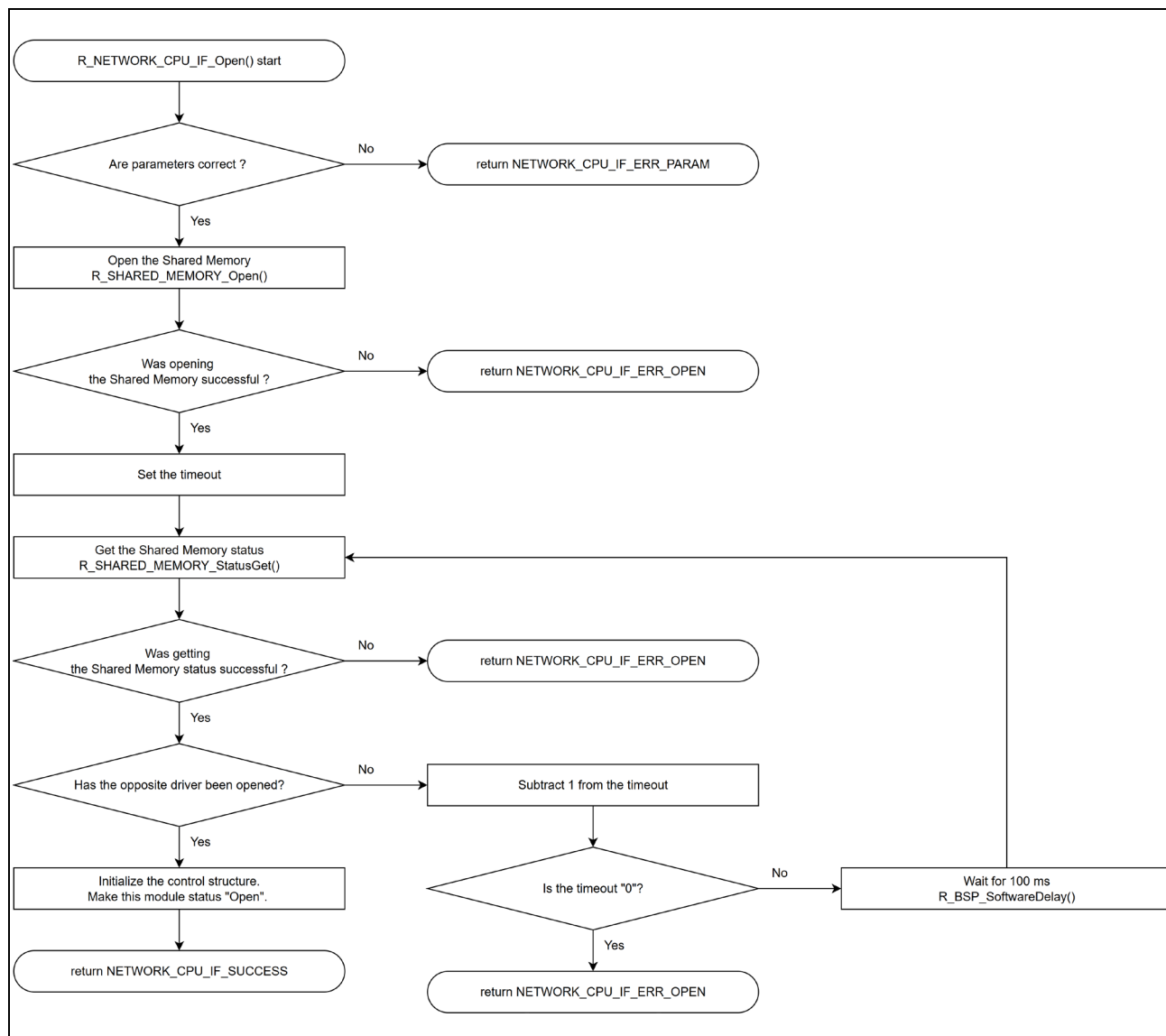


図 7-5. R_NETWORK_CPU_IF_Open 関数フローチャート

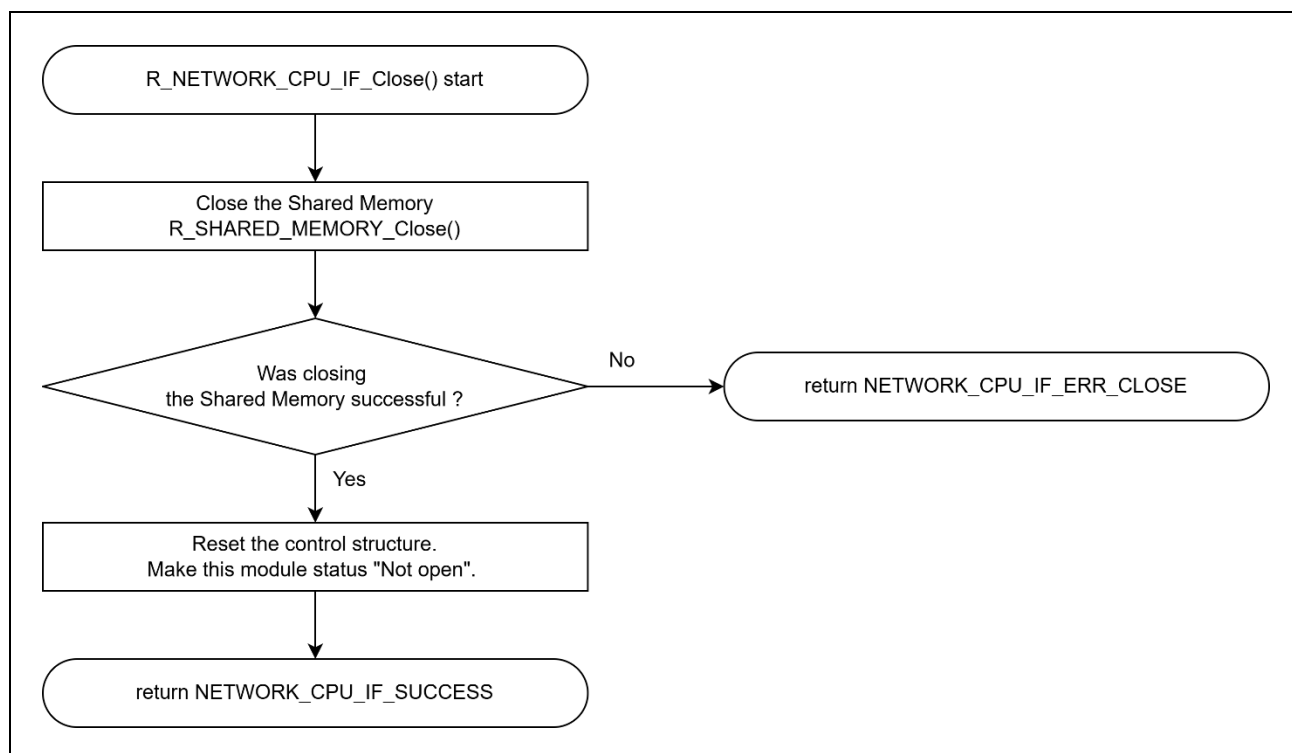


図 7-6. R_NETWORK_CPU_IF_Close 関数フローチャート

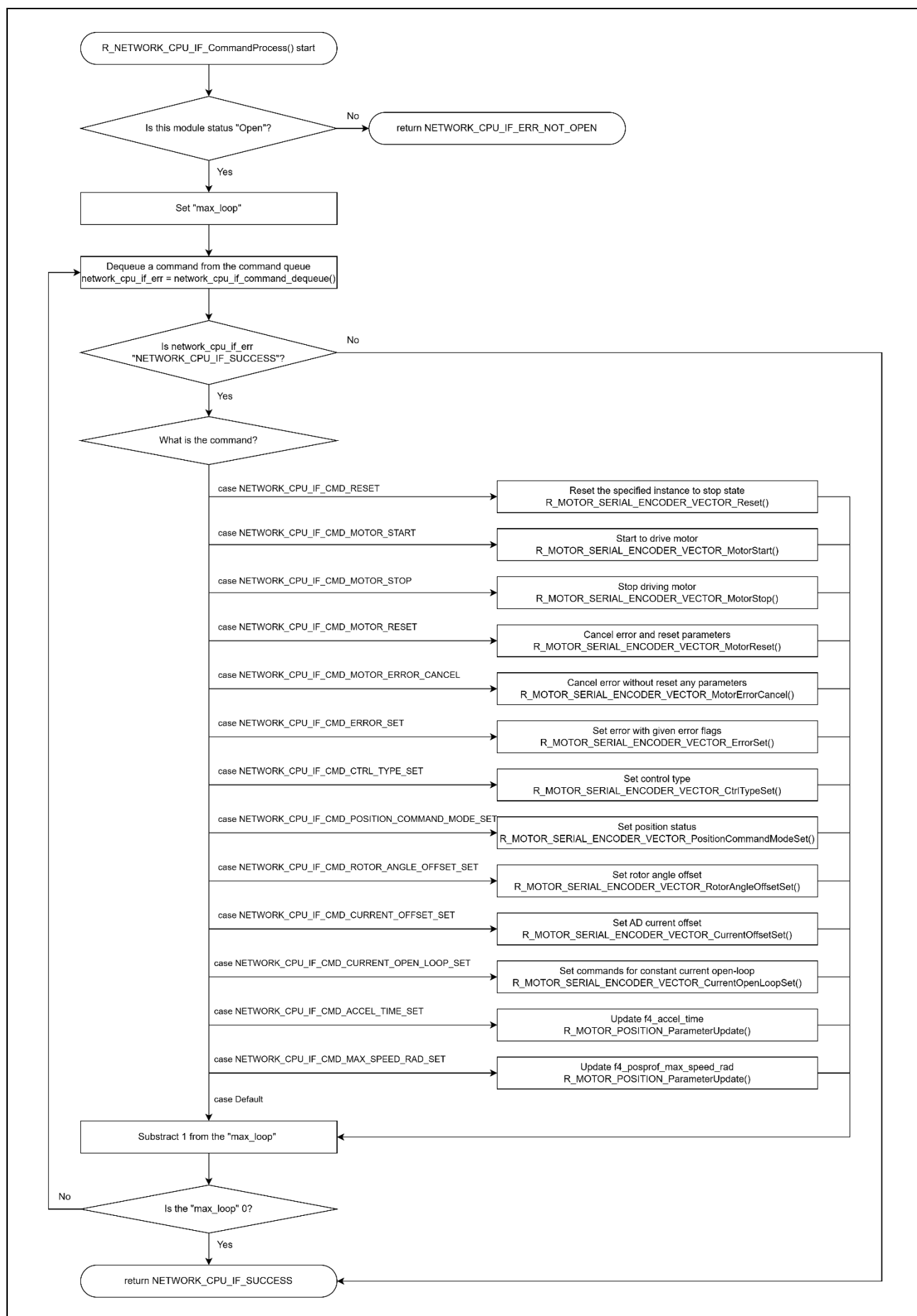


図 7-7. R_NETWORK_CPU_IF_CommandProcess 関数フローチャート

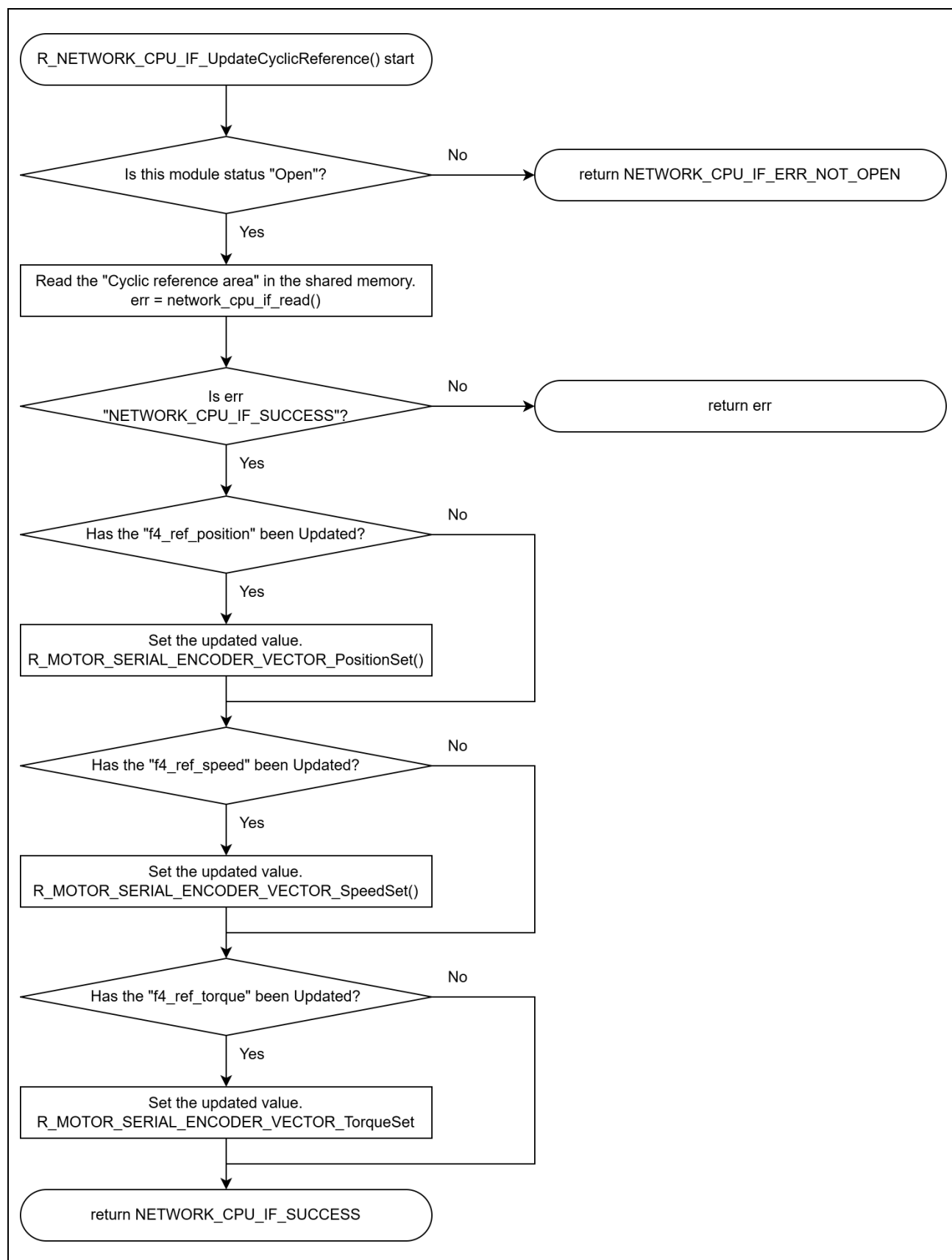


図 7-8. R_NETWORK_CPU_IF_UpdateCyclicReference 関数フローチャート

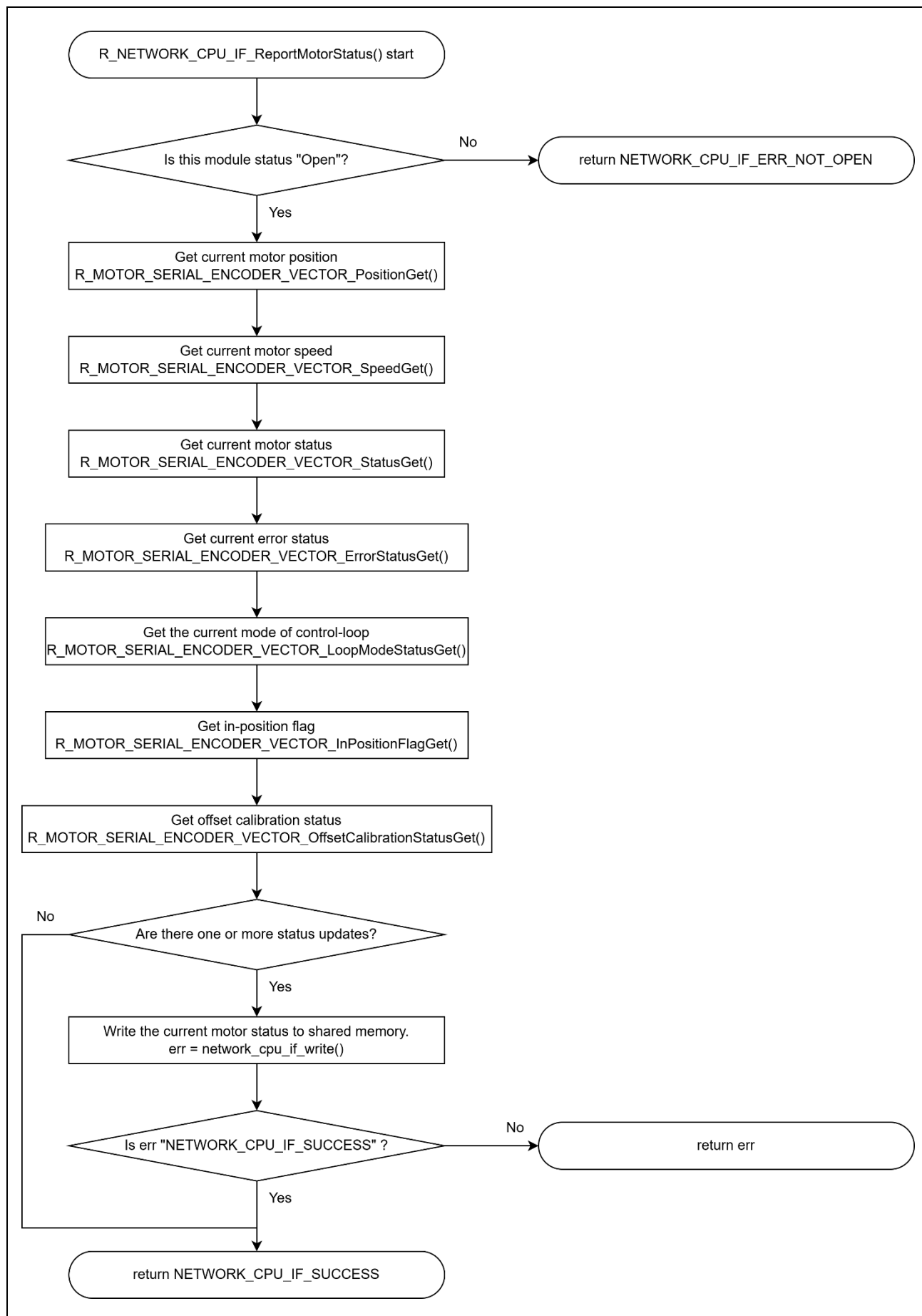


図 7-9. R_NETWORK_CPU_IF_ReportMotorStatus 関数フローチャート

7.2.3.4 プライベート関数

表 7-11. Network CPU I/F プライベート関数 一覧

プライベート関数	説明
network_cpu_if_write	Shared memory にデータを書き込みます。 フローチャート：図 7-10
network_cpu_if_read	Shared memory からデータを読み取ります。 フローチャート：図 7-11
network_cpu_if_command_dequeue	コマンドキューからコマンドを読み取ります。 フローチャート：図 7-12

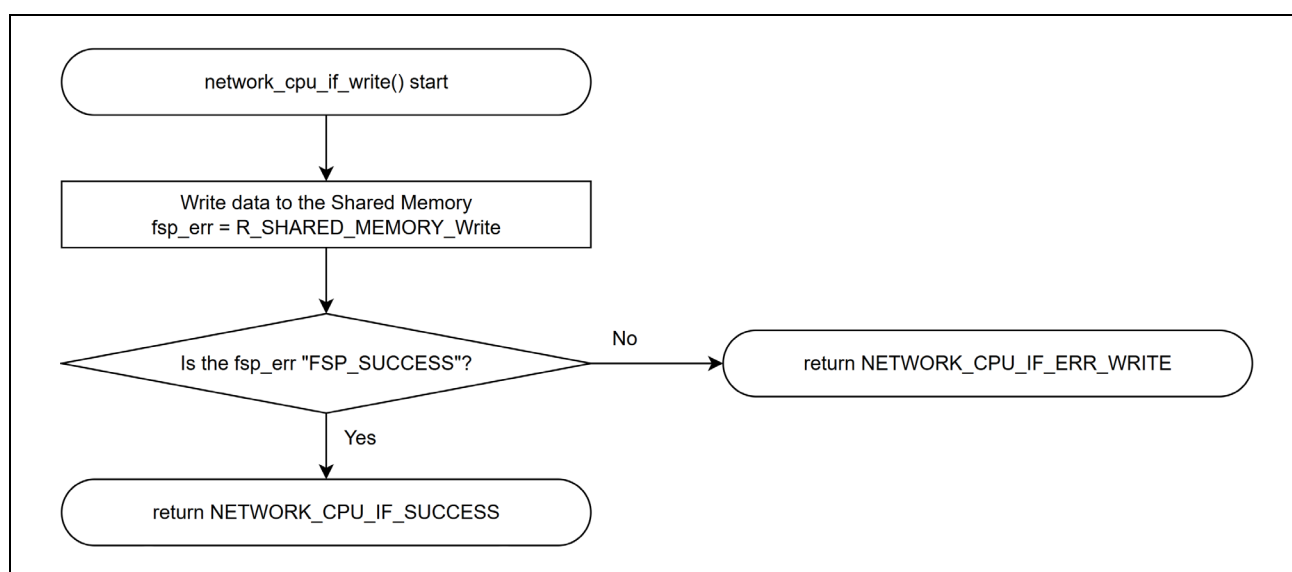


図 7-10. network_cpu_if_write 関数フローチャート

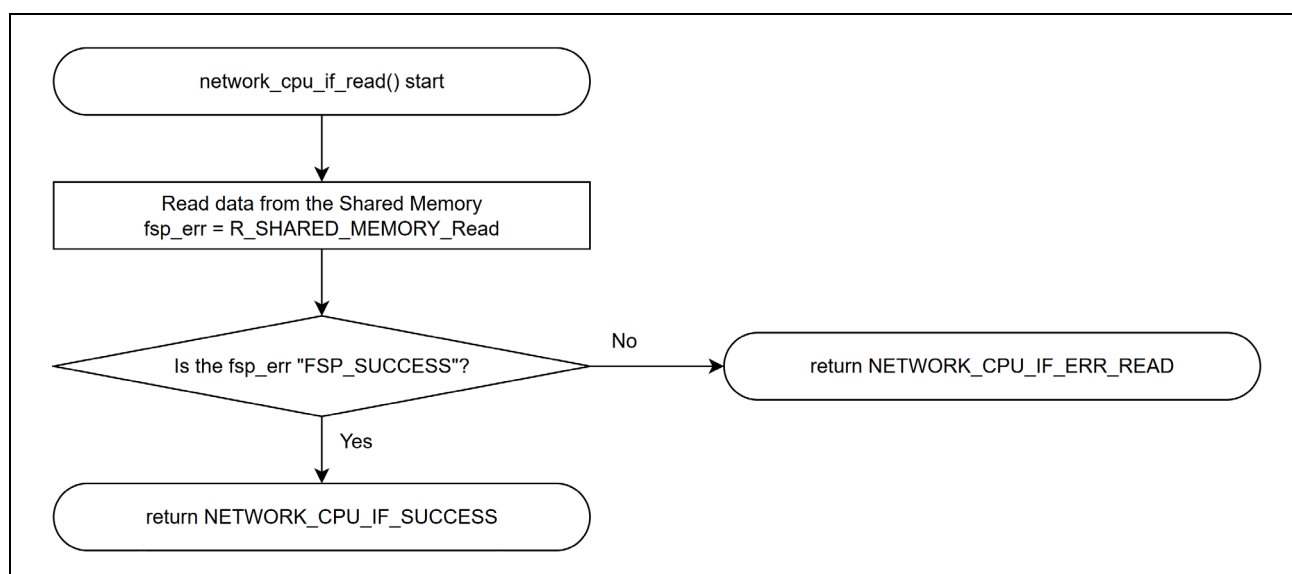


図 7-11. network_cpu_if_read 関数フローチャート

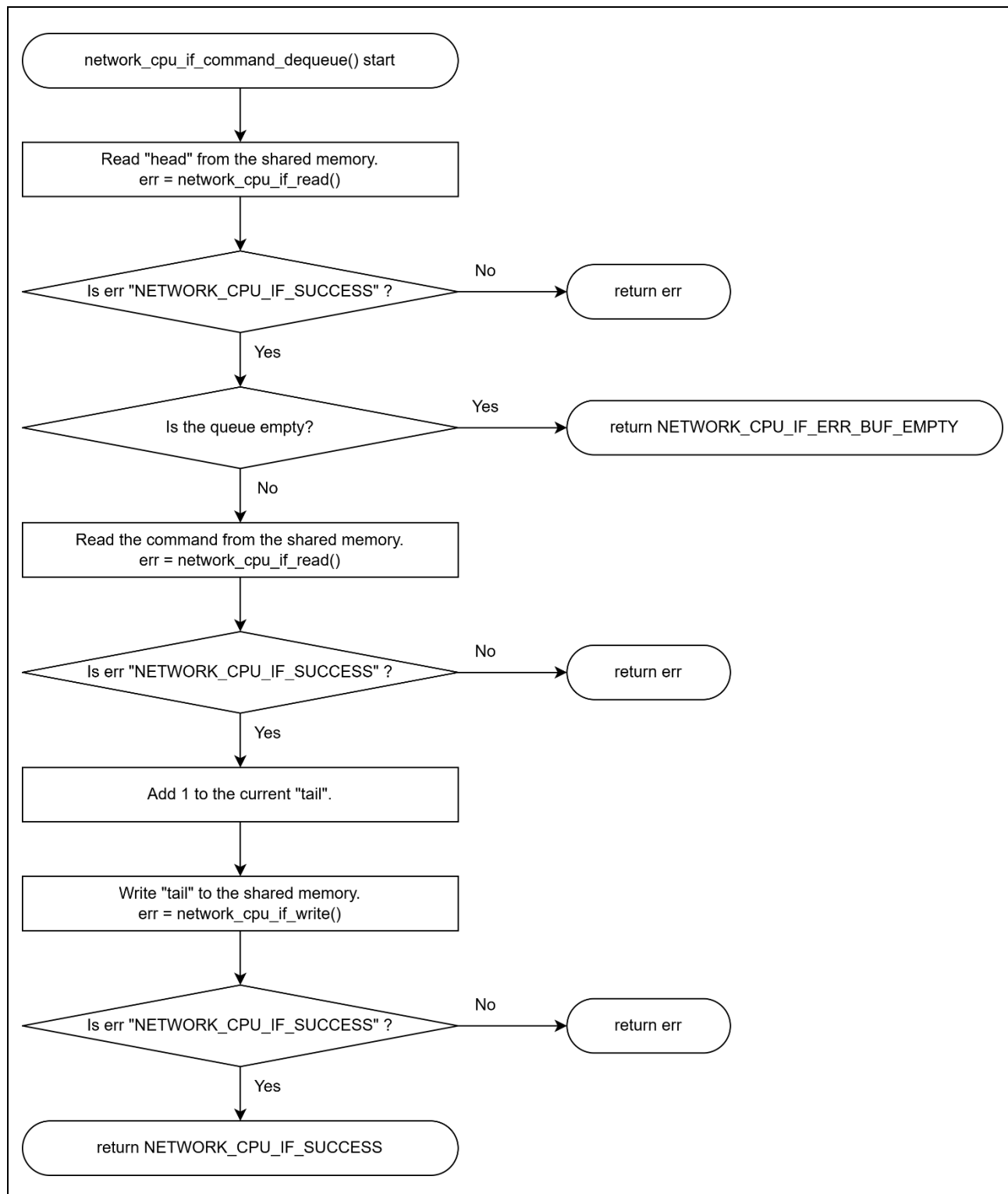


図 7-12. network_cpu_if_command_dequeue 関数フローチャート

7.2.3.5 Callback 関数

表 7-12. Network CPU I/F Callback 関数一覧 (r_network_cpu_if.c)

関数名	説明
shared_memory_callback	これは Shared Memory Driver の Callback 関数です。 RZ/T2N CPU1 側で Shared Memory Driver が Shared Memory へ書き込むと、この割り込みが発生します。 Callback 関数内で行われる処理はありません。 必要に応じて処理の追加を検討して下さい。

7.2.3.6 マクロ定義

表 7-13. Network CPU I/F マクロ定義一覧 (r_network_cpu_if.h)

マクロ名	定義値	備考
CFG_CMD_Q_NUM	8	コマンドキューの数 (2 の累乗にする必要があります)
CFG_CMD_Q_OFFSET	0x0000	Command queue area のオフセットアドレス
CFG_CYCLIC_REF_AREA_OFFSET	0x0100	Cyclic reference area のオフセットアドレス
CFG_MTR_STATUS_AREA_OFFSET	0x0180	Motor status area のオ フセットアドレス
CFG_NETWORK_CPU_IF_TIMEOUT	100	Open 時の Shared memory 準備完了待ち のタイムアウト回数
CFG_SHARED_MEMORY_INSTANCE_ADDR0	&g_shared_memory0	Shared memory ドライ バ 0~3 のインスタンス へのポインタ
CFG_SHARED_MEMORY_INSTANCE_ADDR1	&g_shared_memory1	
CFG_SHARED_MEMORY_INSTANCE_ADDR2	&g_shared_memory2	
CFG_SHARED_MEMORY_INSTANCE_ADDR3	&g_shared_memory3	
CFG_MOTOR_CTRL_INSTANCE_ADDR0	&g_st_serial_encoder_vector 0	モータコントロールイ ンスタンス 0~3 へのポ インタ
CFG_MOTOR_CTRL_INSTANCE_ADDR1	&g_st_serial_encoder_vector 1	
CFG_MOTOR_CTRL_INSTANCE_ADDR2	&g_st_serial_encoder_vector 2	
CFG_MOTOR_CTRL_INSTANCE_ADDR3	&g_st_serial_encoder_vector 3	
NETWORK_CPU_IF_CMD_DATA_MAX_BYTE	12	コマンドキューの最大 データサイズ

7.2.3.7 列挙型

表 7-14. Network CPU I/F 列挙型一覧 (r_network_cpu_if.h)

列挙型	定義	備考
network_cpu_if_err_t Network CPU I/F エラーコード	NETWORK_CPU_IF_SUCCESS	エラー無し
	NETWORK_CPU_IF_ERR_OPEN	Open エラー
	NETWORK_CPU_IF_ERR_CLOSE	Close エラー
	NETWORK_CPU_IF_ERR_PARAM	Parameter エラー
	NETWORK_CPU_IF_ERR_NOT_OPEN	Not open エラー
	NETWORK_CPU_IF_ERR_BUF_FULL	コマンドキューが満杯
	NETWORK_CPU_IF_ERR_BUF_EMPTY	コマンドキューが空
	NETWORK_CPU_IF_ERR_WRITE	書き込みエラー
	NETWORK_CPU_IF_ERR_READ	読み取りエラー
	NETWORK_CPU_IF_ERR_EXCEED_PAYLOAD	データが超過した

列挙型	定義	備考
network_cpu_if_status_t Network CPU I/F ステータス	NETWORK_CPU_IF_STATUS_NOT_OPEN	Not open 状態
	NETWORK_CPU_IF_STATUS_OPEN	Open 状態
network_cpu_if_cmd_t Network CPU I/F コマンド定義	NETWORK_CPU_IF_CMD_NONE	コマンド無し
	NETWORK_CPU_IF_CMD_RESET	Command Process API で 使用されるコマンド (参照 : 図 7-7)
	NETWORK_CPU_IF_CMD_MOTOR_START	
	NETWORK_CPU_IF_CMD_MOTOR_STOP	
	NETWORK_CPU_IF_CMD_MOTOR_RESET	
	NETWORK_CPU_IF_CMD_MOTOR_ERROR_CANCEL	
	NETWORK_CPU_IF_CMD_ERROR_SET	
	NETWORK_CPU_IF_CMD_CTRL_TYPE_SET	
	NETWORK_CPU_IF_CMD_POSITION_COMMAND_MODE_SET	
	NETWORK_CPU_IF_CMD_ROTOR_ANGLE_OFFSET_SET	
	NETWORK_CPU_IF_CMD_CURRENT_OFFSET_SET	
	NETWORK_CPU_IF_CMD_CURRENT_OPEN_LOOP_SET	
	NETWORK_CPU_IF_CMD_ACCEL_TIME_SET	
	NETWORK_CPU_IF_CMD_MAX_SPEED_RAD_SET	

7.2.3.8 構造体・変数情報

表 7-15. Network CPU I/F 構造体定義一覧 (r_network_cpu_if.h)

構造体	変数	備考
st_current_offset_cfg_t Current offset configuration 構造体	f4_iu_offset	U-phase 電流オフセット
	f4_iv_offset	V-phase 電流オフセット
	f4_iw_offset	W-phase 電流オフセット
st_current_open_loop_cfg_t Open loop current configuration 構造体	f4_open_loop_current	Open loop 電流値
	f4_ref_speed_rpm	指令速度 (RPM)
network_cpu_if_cmd_hdr_t コマンドキュー head/tail 構 造体	head	コマンドキューの head
	tail	コマンドキューの tail
network_cpu_if_cmd_q_t コマンドキュー 構造体	cmd	コマンド
	size	データサイズ
	data[NETWORK_CPU_IF_ CMD_DATA_MAX_BYTE]	データ配列
network_cpu_if_cyclic_ref_t Cyclic Reference 構造体	f4_ref_position	指令位置
	f4_ref_speed	指令速度
	f4_ref_torque	指令トルク
network_cpu_if_mtr_status_t Motor Status 構造体	f4_pos_deg	現在位置
	f4_speed	現在速度
	u1_status	ステータマシン
	reserved1	未使用領域
	u2_error_status	エラー状態
	reserved2	未使用領域
	u1_ctrl_loop_mode	LOOP MODE
	reserved3	未使用領域
	u1_in_position	位置制御完了状態
	reserved4	未使用領域
	u1_offset_cal_finished_flag	オフセット測定状態
	reserved5	未使用領域
network_cpu_if_cfg_t Network CPU I/F コンフィグ構造体	p_shared_memory_instance	Shared Memory インスタンス へのポインタ
	timeout	Open 時の Shared memory 準 備完了待ちのタイムアウト回数
	shr_mem_cmd_data_max_size	コマンドデータ最大サイズ
	shr_mem_cmd_q_num	コマンドキューの数

	shr_mem_cmd_q_mask	コマンドキューマスク
	shr_mem_cmd_q_offset	共有メモリのコマンドキュー領域のオフセット
	shr_mem_cmd_q_size	共有メモリのコマンドキュー領域のサイズ
	shr_mem_cmd_head_offset	共有メモリのコマンドキュー head 領域のオフセット
	shr_mem_cmd_head_size	共有メモリのコマンドキュー head 領域のサイズ
	shr_mem_cmd_tail_offset	共有メモリのコマンドキュー tail 領域のオフセット
	shr_mem_cmd_tail_size	共有メモリのコマンドキュー tail 領域のサイズ
	shr_mem_cyclic_ref_area_offset	共有メモリの Cyclic Reference Area のオフセット
	shr_mem_cyclic_ref_area_size	共有メモリの Cyclic Reference Area のサイズ
	shr_mem_ref_pos_offset	共有メモリの指令位置領域のオフセット
	shr_mem_ref_vel_offset	共有メモリの指令速度領域のオフセット
	shr_mem_ref_tor_offset	共有メモリの指令トルク領域のオフセット
	shr_mem_mtr_status_area_offset	共有メモリの Motor Status Area のオフセット
	shr_mem_mtr_status_area_size	共有メモリの Motor Status Area のサイズ
	shr_mem_mtr_pos_offset	共有メモリの現在位置領域のオフセット
	shr_mem_mtr_vel_offset	共有メモリの現在速度領域のオフセット
	shr_mem_mtr_status_offset	共有メモリのステートマシン領域のオフセット
	shr_mem_mtr_err_status_offset	共有メモリのエラー状態領域のオフセット
	shr_mem_mtr_ctrl_loop_mode_offset	共有メモリの LOOP MODE 領域のオフセット
	shr_mem_mtr_in_pos_offset	共有メモリの位置制御完了状態領域のオフセット
	shr_mem_mtr_cal_finished_flag_offset	共有メモリのオフセット測定状態領域のオフセット
	max_loop	CommandProcess API で使用される最大ループ回数

	p_st_serial_encoder_vector_control	Serial encoder vector 制御構造体へのポインタ
network_cpu_if_ctrl_t Network CPU I/F 制御構造体	status	Network CPU I/F のステータス
	cmd_hdr	コマンドキュー head/tail 構造体インスタンス
	cyclic_ref_buf	Cyclic Reference 領域のバッファ
	mtr_status_buf	Motor Status 領域のバッファ
	p_cfg	Network CPU I/F コンフィグ構造体へのポインタ
network_cpu_if_api_t Network CPU I/F API 構造体	Open	オープン関数
	Close	クローズ関数
	CommandProcess	コマンド処理関数
	UpdateCyclicReference	周期的指令値アップデート関数
	ReportMotorStatus	モータステータスレポート関数

7.2.3.9 グローバル変数情報

表 7-16. Network CPU I/F グローバル変数一覧 (r_network_cpu_if.c)

型名	インスタンス名	説明
network_cpu_if_ctrl_t	g_network_cpu_if_ctrl0 g_network_cpu_if_ctrl1 g_network_cpu_if_ctrl2 g_network_cpu_if_ctrl3	モータ 4 軸分の Network CPU I/F 制御構造体インスタンス。 Open 関数で初期化されます。
network_cpu_if_cfg_t	g_network_cpu_if_cfg0 g_network_cpu_if_cfg1 g_network_cpu_if_cfg2 g_network_cpu_if_cfg3	モータ 4 軸分の Network CPU I/F コンフィグ構造体インスタンス。 インスタンス生成時に各メンバが初期化されます。 初期化以外で各メンバの値を変更することはできません。
network_cpu_if_api_t	g_network_cpu_if_api	Network CPU I/F API 構造体インスタンス。 インスタンス生成時に各メンバが初期化されます。 初期化以外で各メンバの値を変更することはできません。

7.2.4 CPU0 CPU 間データ交換制御モジュール

CPU0 CPU 間データ交換制御モジュールは、CPU1 から INTCPU9 を受信したときだけ、Network CPU I/F を通じて Shared Memory とデータを交換し、完了を INTCPU8 で CPU1 へ通知します。

7.2.4.1 モジュール構成

モジュール構成図を図 7-13 に示します。

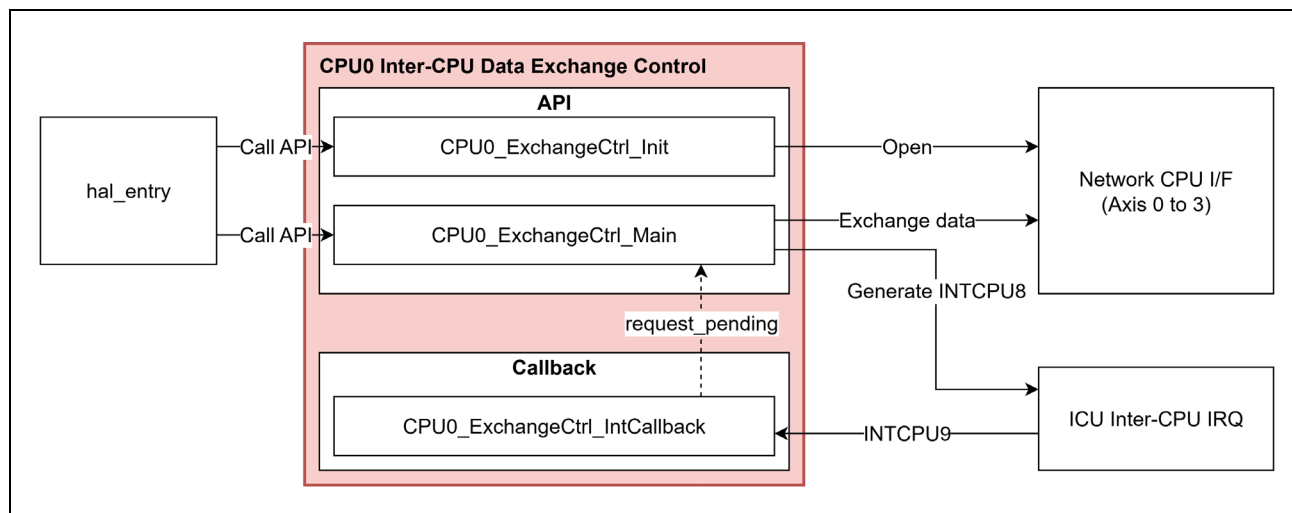


図 7-13. CPU0 CPU 間データ交換制御モジュール構成図

7.2.4.2 API

表 7-17. CPU0 CPU 間データ交換制御 API 一覧

API	説明
CPU0_ExchangeCtrl_Init	CPU0 CPU 間データ交換制御モジュールを初期化します。 4 軸分の Network CPU I/F を Open し、INTCPU9 受信用と INTCPU8 送信用の Inter CPU IRQ を Open します。
CPU0_ExchangeCtrl_Main	INTCPU9 を受信していない場合は、何もせずに戻ります。 受信している場合は、4 軸すべてについて R_NETWORK_CPU_IF_CommandProcess、R_NETWORK_CPU_IF_UpdateCyclicReference、R_NETWORK_CPU_IF_ReportMotorStatus を順に呼び出した後、INTCPU8 を発生させて CPU1 へ通知します。

7.2.4.3 Callback 関数

表 7-18. CPU0 CPU 間データ交換制御 Callback 関数一覧 (cpu0_exchange_ctrl.c)

関数名	説明
CPU0_ExchangeCtrl_IntCallback	これは INTCPU9 (r_icu_inter_cpu_irq ch9) の Callback 関数です。 request_pending を true にします。Shared Memory へはアクセスしません。

7.2.4.4 構造体・変数情報

表 7-19. CPU0 CPU 間データ交換制御 構造体定義一覧 (cpu0_exchange_ctrl.h)

構造体	変数	備考
cpu0_exchange_ctrl_cfg_t コンフィグ構造体	p_network_cpu_if_ctrl[4]	軸 0~3 の Network CPU I/F 制御構造体へのポインタ
	p_network_cpu_if_cfg[4]	軸 0~3 の Network CPU I/F コンフィグ構造体へのポインタ
cpu0_exchange_ctrl_t 制御構造体	request_pending	INTCPU9 を受信し、データ交換が必要であることを示すフラグ
	p_cfg	コンフィグ構造体へのポインタ

7.2.4.5 グローバル変数情報

表 7-20. CPU0 CPU 間データ交換制御 グローバル変数一覧 (cpu0_exchange_ctrl.c)

型名	インスタンス名	説明
cpu0_exchange_ctrl_t	g_cpu0_exchange_ctrl	CPU0 CPU 間データ交換制御の制御構造体インスタンス。 ゼロ初期化され、Init 関数で p_cfg が設定されます。
cpu0_exchange_ctrl_cfg_t	g_cpu0_exchange_ctrl_cfg	CPU0 CPU 間データ交換制御のコンフィグ構造体インスタンス。 4 軸分の Network CPU I/F 制御構造体・コンフィグ構造体インスタンスで初期化されます。

7.3 EtherCAT(CiA402) (RZ/T2N Cortex-R52 CPU1)

7.3.1 基本仕様

表 7-21. RZ/T2N Cortex-R52 CPU1 ソフトウェア基本仕様

機能	項目	説明
EtherCAT SubDevice Stack	PHY とのインターフェース	RGMII
	MDIO タイプ	GMAC
	通信ポート数	In, Out の 2 ポート
	EtherCAT LED	RUN, ERR, L/A IN, L/A OUT の 4 種類
	EtherCAT Device ID	SW6(4bit)により設定可能
	デバイスプロファイル	CiA402 Drive Profile
	メールボックス	CoE をサポート
	同期モード	DC Mode (DC-Synchronous with SM2 event), SM mode (SM-synchronous) の 2 種類をサポート
	DC Mode の Cycle Time	最小 50 μ s, 最大約 3 s
	割り込み優先度 (PDI, Sync0, Sync1)	Priority 4
	割り込み優先度(Timer)	Priority 6
CiA402 Drive Profile	Mode	Profile position mode, Homing mode, Cyclic synchronous position mode, Cyclic synchronous velocity mode, の 4 種類をサポート
	モータ軸数	4 軸
Motor CPU I/F	役割	Motor 制御 CPU との通信 I/F を提供
	通信手段	Shared Memory
	割り込み優先度 (INTCPU)	Priority 6
	モジュール数	4

7.3.2 ソフトウェア全体構成

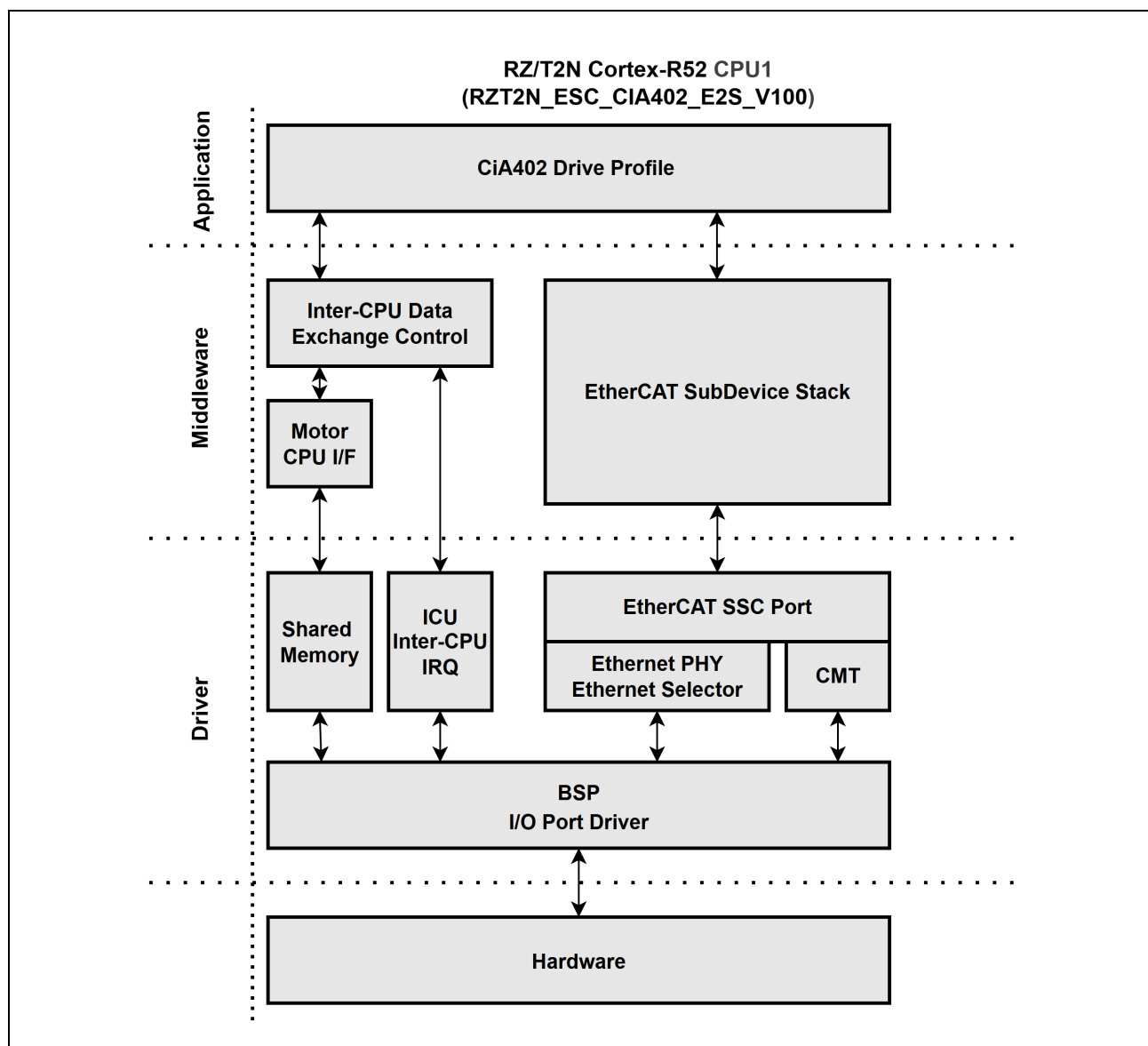


図 7-14. RZ/T2N Cortex-R52 CPU1 ソフトウェア全体構成

7.3.3 src フォルダ構成

表 7-22. RZ/T2N CPU1 src フォルダ構成

フォルダ・ファイル				説明	
src	ethercat	Beckhoff/Src	applInterface.h	EtherCAT SubDevice Stack (SSC Tool 生成コード)	
			coeappl.c		
			coeappl.h		
			ecatappl.c		
			ecatappl.h		
			ecatcoe.c		
			ecatcoe.h		
			ecatslv.c		
			ecatslv.h		
			ecat_def.h		
			esc.h		
			mailbox.c		
			mailbox.h		
			objdef.c		
			objdef.h		
			sdoserv.c		
			sdoserv.h		
			cia402appl.c		CiA402 Drive Profile (SSC Tool 生成コード)
			cia402appl.h		
		renesas	samplecia402.c	CiA402 Drive Profile (ルネサス作成コード)	
			samplecia402.h		
	cpu1_exchange_ctrl			CPU1 CPU 間データ交換制御モジュール	
	motor_cpu_if			Motor CPU I/F モジュール	
	hal_entry.c			hal_entry 関数	

7.3.4 hal_entry・メイン処理

hal_entry 関数のフローチャート図を図 7-15 に示します。

また、hal_entry 関数で呼び出される関数について表 7-23 に示します。

表 7-23. hal_entry 関数で呼び出される関数一覧

関数名	説明。
RM_ETHERCAT_SSC_PORT_Open	FSP で提供される EtherCAT SSC Port の Open 関数。 詳細は RZT Flexible Software Package Documentation を参照してください。
CPU1_ExchangeCtrl_Init	CPU1 CPU 間データ交換制御の初期化関数。 詳細は 7.3.7 章を参照してください。
ecat_ssc_init	EtherCAT SubDevice Stack Code および CiA402 Drive Profile の初期化処理を実行する関数。 図 7-16 にフローチャートを示します。 フローチャート内の各関数・変数の詳細は " Application Note ET9300 (EtherCAT SubDevice Stack Code)" を参照してください。
R_BSP_PinSet	FSP で提供される関数。LED9 を点灯します。 詳細は RZT Flexible Software Package Documentation を参照してください。
MainLoop	EtherCAT SubDevice Stack Code および CiA402 Drive Profile のメイン処理関数。 詳細は " Application Note ET9300 (EtherCAT SubDevice Stack Code)" を参照してください。

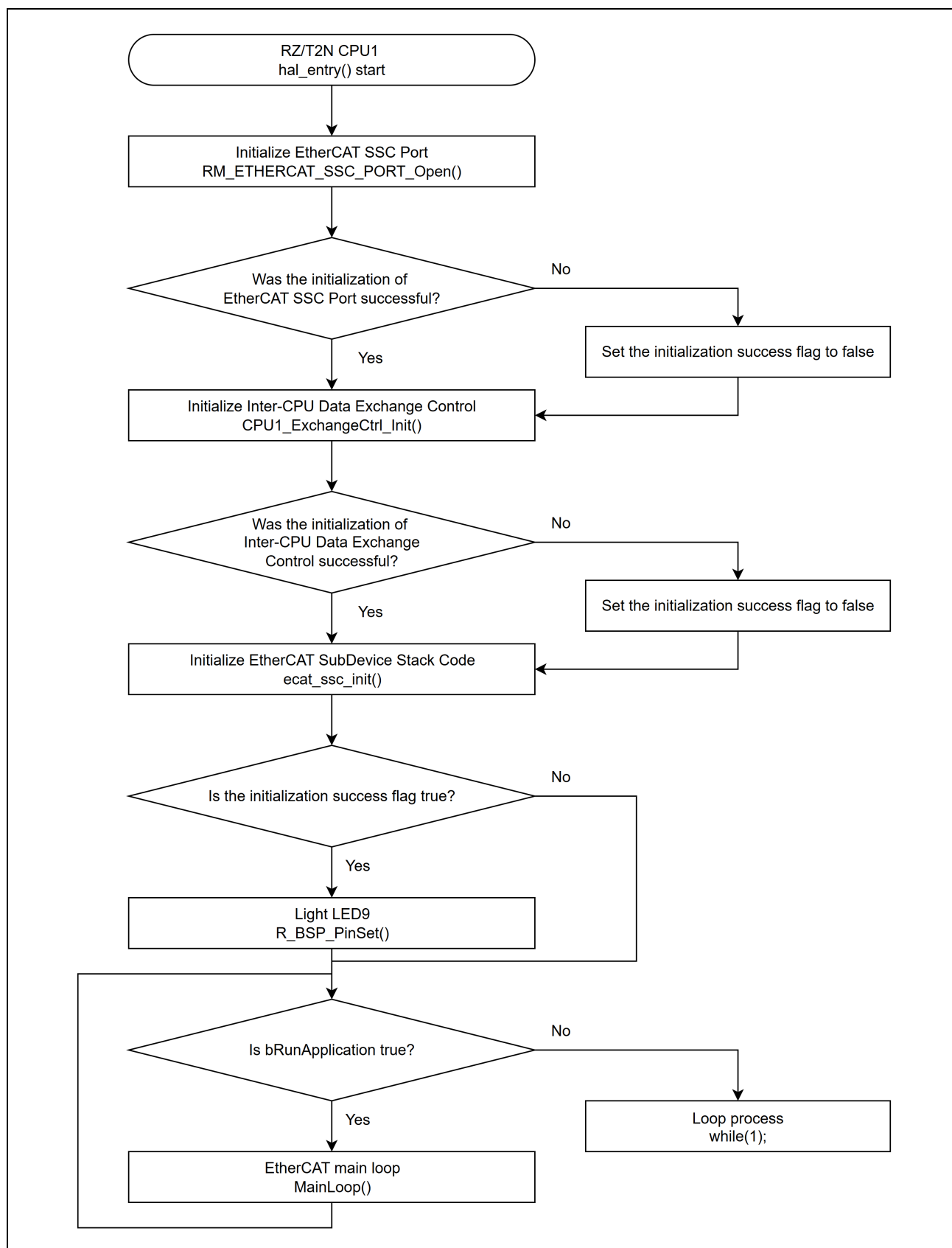


図 7-15. RZ/T2N CPU1 hal_entry 関数フローチャート

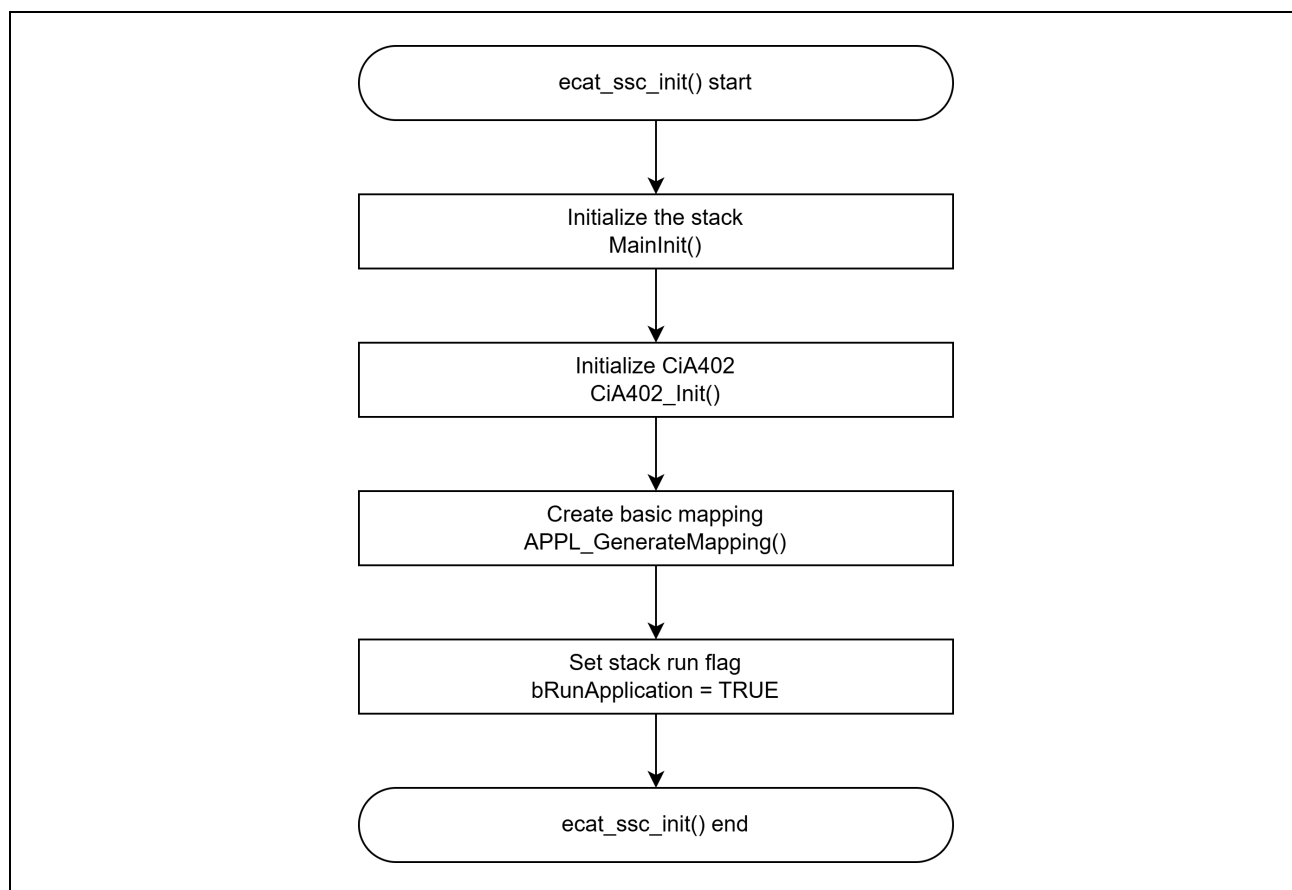


図 7-16. ecat_ssc_init 関数フローチャート

7.3.5 CiA402 Drive Profile

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査値（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

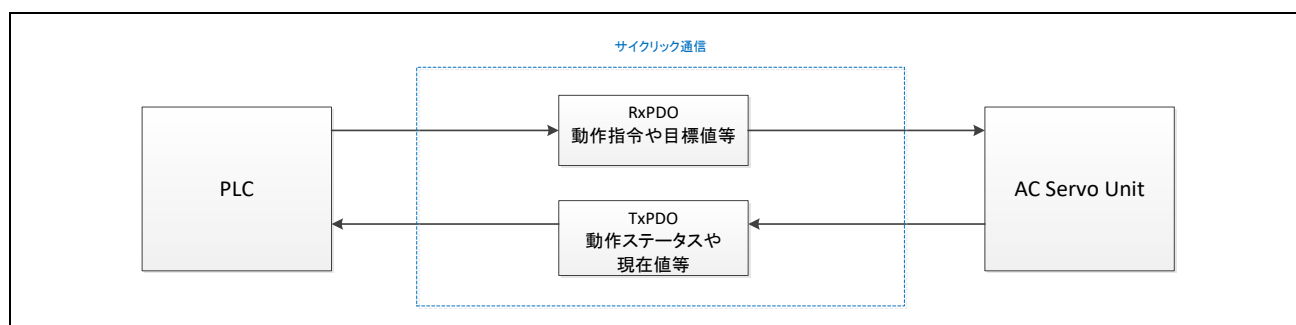


図 7-17. CiA402 Drive Profile 概要

7.3.5.1 CiA402 動作モード

CiA402 として規定されている動作モードのうち、本プログラムでは以下をサポートします。

表 7-24. 動作モード一覧

Operation Mode	Support	Refer for details
Profile position mode	Available	7.3.5.4
Velocity mode (frequency converter)	Not available	-
Profile velocity mode	Not available	-
Profile torque mode	Not available	-
Homing mode	Available	7.3.5.5
Interpolated position mode	Not available	-
Cyclic synchronous position mode	Available	7.3.5.2
Cyclic synchronous velocity mode	Available	7.3.5.3
Cyclic synchronous torque mode	Not available	-
Cyclic synchronous torque mode with commutation angle	Not available	-
Manufacturer specific mode	Not available	-

7.3.5.2 Cyclic synchronous position mode

このモードでは、周期的に指令位置を送信し、モータの位置制御を行います。

表 7-25. csp モード RxPDO 一覧

RxPDO	Index	説明
Control Word	0x6040	CiA402 制御用コマンド
Target Position	0x607A	位置指令値 [deg] (*1)

(*1) モータへの指令値は入力値の 1/10 に変換されます。例) 1234 を入力→モータへの指令値は 123.4。

表 7-26. csp モード TxPDO 一覧

TxPDO	Index	説明
Error Code	0x603F	CiA402 エラーコード
Status Word	0x6041	CiA402 状態
Modes of Operation Display	0x6061	モードの種類 (=8)
Position Actual Value	0x6064	モータ現在位置 [deg] (*2)
Velocity Actual Value	0x606C	モータ現在速度 [rpm] (*2)

(*2) 実際の値から 10 倍に変換されます。例) 5678 が出力→実際の値は 567.8。

表 7-27. csp モード モータ設定

項目	設定値
LOOP MODE	位置制御
位置制御の位置指令生成のモード	台形駆動動作

7.3.5.3 Cyclic synchronous velocity mode

このモードでは、周期的に指令速度を送信し、モータの速度制御を行います。

表 7-28. csv モード RxPDO 一覧

RxPDO	Index	説明
Control Word	0x6040	CiA402 制御用コマンド
Target Velocity	0x60FF	速度指令値 [rpm] (*1)

(*1) モータへの指令値は入力値の 1/10 に変換されます。例) 1234 を入力→モータへの指令値は 123.4

表 7-29. csv モード TxPDO 一覧

TxPDO	Index	説明
Error Code	0x603F	CiA402 エラーコード
Status Word	0x6041	CiA402 状態
Modes of Operation Display	0x6061	モードの種類 (=9)
Position Actual Value	0x6064	モータ現在位置 [deg] (*2)
Velocity Actual Value	0x606C	モータ現在速度 [rpm] (*2)

(*2) 実際の値から 10 倍に変換されます。例) 5678 が表示→モータ現在位置は 567.8deg

表 7-30. csv モード モータ設定

項目	設定値
LOOP MODE	速度制御

7.3.5.4 Profile position mode

このモードでは、速度・加速度を設定し、モータの位置制御を行います。

表 7-31. pp モード RxPDO 一覧

RxPDO	Index	説明
Control Word	0x6040	CiA402 制御用コマンド
Target Position	0x607A	位置指令値 [deg] (*1)
Profile Velocity	0x6081	プロファイル速度 [rpm] (*1)
Profile Acceleration	0x6083	プロファイル加速度[rpm/s] (*1)
Profile Deceleration	0x6084	プロファイル減速度[rpm/s] (*3)

(*1) モータへの指令値は入力値の 1/10 に変換されます。例) 1234 を入力→モータへの指令値は 123.4。

(*3) 本ソフトウェアでは未使用です。

表 7-32. pp モード TxPDO 一覧

TxPDO	Index	説明
Error Code	0x603F	CiA402 エラーコード
Status Word	0x6041	CiA402 状態
Modes of Operation Display	0x6061	モードの種類 (=1)
Position Actual Value	0x6064	モータ現在位置 [deg] (*2)
Velocity Actual Value	0x606C	モータ現在速度 [rpm] (*2)

(*2) 実際の値から 10 倍に変換されます。例) 5678 が出力→実際の値は 567.8。

表 7-33. pp モード モータ設定

項目	設定値
LOOP MODE	位置制御
位置制御の位置指令生成のモード	台形駆動動作
加減速時間 [s]	(Profile Velocity) / (Profile Acceleration)
台形速度制御の最大速度 [rpm]	(Profile Velocity) / 10

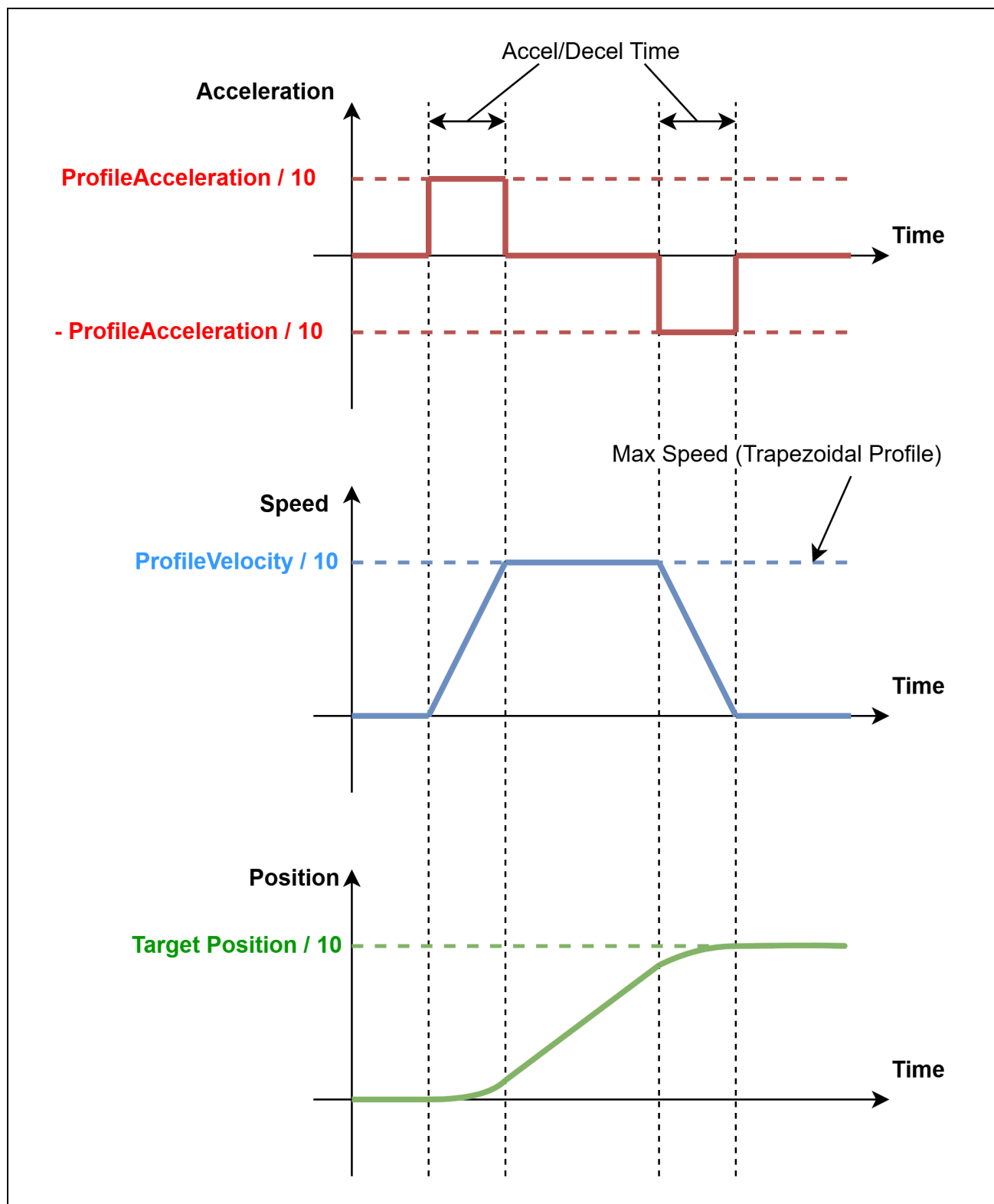


図 7-18. pp mode 台形駆動動作設定

7.3.5.5 Homing mode

このモードでは、Homing Method 37 を使用してモータの現在位置を変更します。

表 7-34. Homing モード RxPDO 一覧

RxPDO	Index	説明
Control Word	0x6040	CiA402 制御用コマンド
Home Offset	0x607C	指令現在位置 [deg] (*1)
Home Method	0x6098	Homing 方法
Homing Speed For Switch	0x6099	スイッチ用指令速度 (*3)
Homing Speed For Zero	0x6099	原点用指令速度 (*3)
Homing Acceleration	0x609A	Homing 全体の加減速 (*3)

(*1) モータへの指令値は入力値の 1/10 に変換されます。例) 1234 を入力→モータへの指令値は 123.4。

(*3) 本ソフトウェアでは未使用です。

表 7-35. Homing モード TxPDO 一覧

TxPDO	Index	説明
Error Code	0x603F	CiA402 エラーコード
Status Word	0x6041	CiA402 状態
Modes of Operation Display	0x6061	モードの種類 (=6)
Position Actual Value	0x6064	モータ現在位置 [deg] (*2)
Velocity Actual Value	0x606C	モータ現在速度 [rpm] (*2)

(*2) 実際の値から 10 倍に変換されます。例) 5678 が出力→実際の値は 567.8。

表 7-36. Homing モード モータ設定

項目	設定値
LOOP MODE	位置制御
位置制御の位置指令生成のモード	台形駆動動作

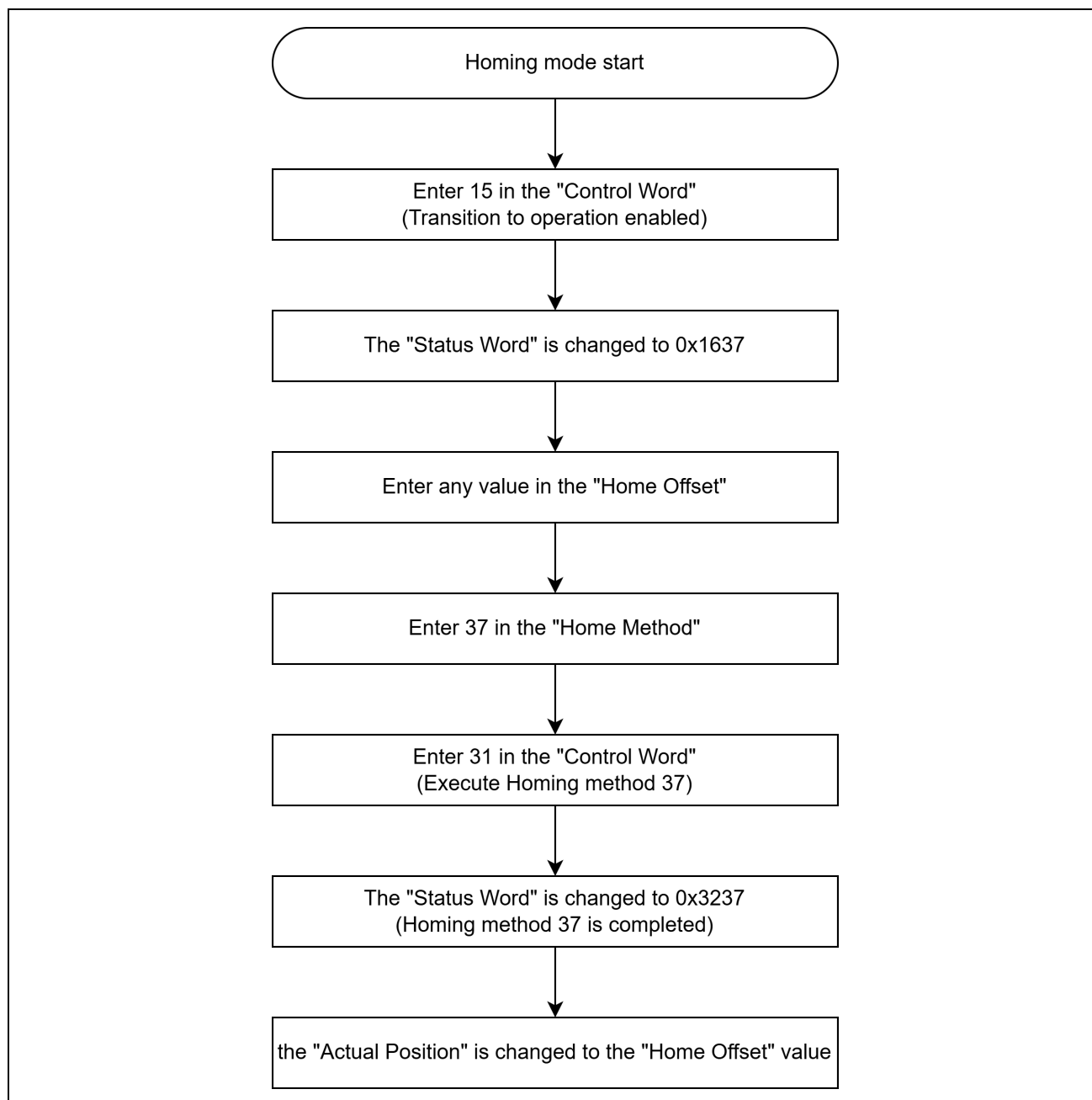


図 7-19. Homing method 37 の実行フローチャート

7.3.5.6 CiA402 状態遷移

CiA402 として規定されている FSA として、サンプルプログラムでは以下をサポートします。

表 7-37 に対応する遷移番号の発生トリガーおよび挙動を示します。

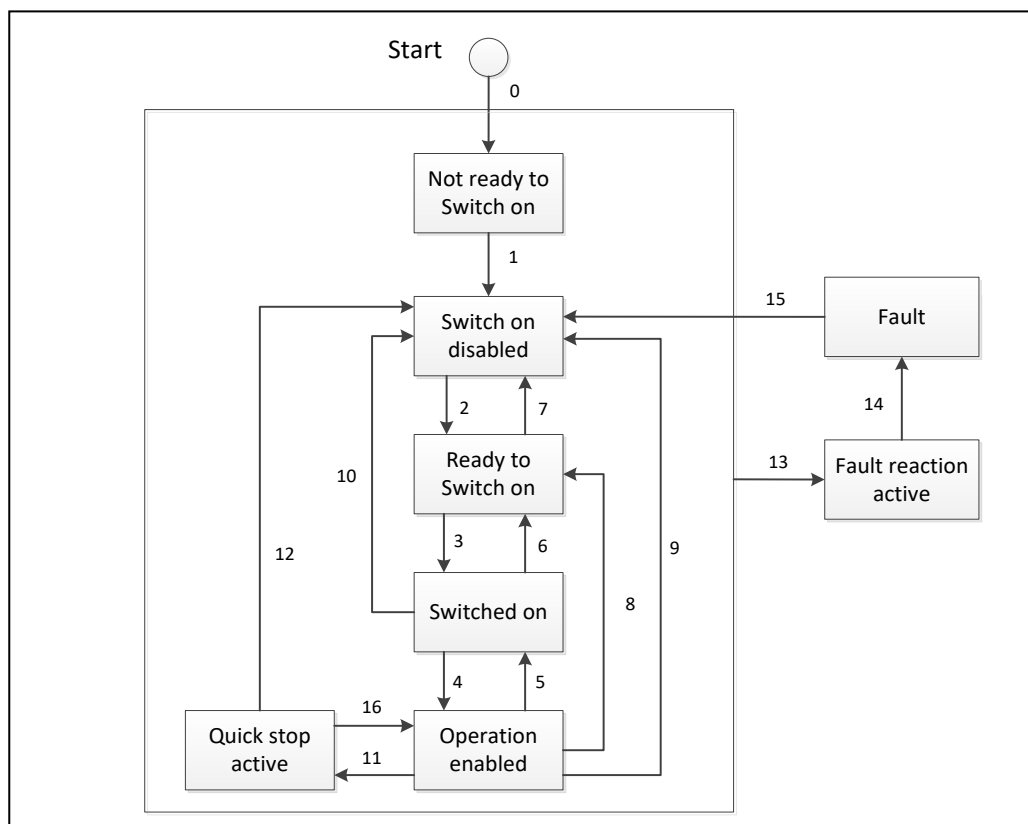


図 7-20. CiA402 状態遷移図

表 7-37. CiA402 各状態遷移

対応する 遷移番号	遷移発生トリガー	遷移時の挙動
0, 1	EtherCAT State Machine(ESM)が OP であれば自動でトリガー	何もしない。
2		図 7-21 を参照。
3	Control Word = 7 または Control Word = 15	図 7-22 を参照。
4	Control Word = 15	図 7-23 を参照。
5	Control Word = 7	図 7-24 を参照。
6	Control Word = 6	何もしない。
7	Control Word = 0	何もしない。
8	Control Word = 6	図 7-24 を参照。
9	Control Word = 0	図 7-24 を参照。
10	Control Word = 0	何もしない。
11	Control Word = 2	図 7-24 を参照。

12	Control Word = 0	何もしない。
13	A) 遷移 2 実行時、モータエラーがある場合 B) Operation enabled 時にモータエラーがある場合	図 7-25 を参照。
14	Fault reaction active のとき自動でトリガー。	何もしない。
15	Control Word = 128	図 7-26 を参照。
16	Control Word = 15	図 7-27 を参照。

7.3.5.7 CiA402 Error Code

Fault reaction active へ遷移する時、発生したエラー内容に応じて Error Code (0x603F) に値が格納されます。表 7-38 に Error Code 一覧を示します。

表 7-38. Error Code(0x603F) 一覧

エラー種類	Error Code	意味
モータエラー	0x2214	HW 過電流エラー(0x0001)が発生した。
	0x3211	過電圧エラー(0x0002)が発生した。
	0x7310	回転速度エラー(0x0004)が発生した。
	0x3221	低電圧エラー(0x0080)が発生した。
	0x2220	SW 過電流エラー(0x0100)が発生した。
	0x4310	温度超過エラー(0x0200)が発生した。
	0x7120	未定義エラー(0xFFFF)が発生した。
モータ停止要求	0xFF00	STO の要求があった。
	0xFF01	SS1-t の要求があった。
モータ動作禁止	0x2250	モータ動作が禁止されている。

7.3.5.8 CiA402 Drive Profile 向け関数

ここでは、本サンプルソフトウェアにて実装されている関数について説明します。

SSC Tool が生成する EtherCAT SubDevice Stack Code に実装されている関数については、

” Application Note ET9300 (EtherCAT SubDevice Stack Code)”を参照してください。

表 7-39. CiA402 状態遷移関数一覧 (samplecia402.c)

関数名	説明
CiA402_StateTransition1	本ソフトウェアでは処理はありません。 (*1)
CiA402_StateTransition2	フローチャートを図 7-21 に示します。 (*1)
CiA402_StateTransition3	フローチャートを図 7-22 に示します。 (*1)
CiA402_StateTransition4	フローチャートを図 7-23 に示します。 (*1)
CiA402_StateTransition5	フローチャートを図 7-24 に示します。 (*1)
CiA402_StateTransition6	本ソフトウェアでは処理はありません。 (*1)
CiA402_StateTransition7	本ソフトウェアでは処理はありません。 (*1)
CiA402_StateTransition8	フローチャートを図 7-24 に示します。 (*1)
CiA402_StateTransition9	フローチャートを図 7-24 に示します。 (*1)
CiA402_StateTransition10	本ソフトウェアでは処理はありません。 (*1)
CiA402_StateTransition11	フローチャートを図 7-24 に示します。 (*1)
CiA402_StateTransition12	本ソフトウェアでは処理はありません。 (*1)
CiA402_LocalError	フローチャートを図 7-25 に示します。 (*1)
CiA402_StateTransition14	本ソフトウェアでは処理はありません。 (*1)
CiA402_StateTransition15	フローチャートを図 7-26 に示します。 (*1)
CiA402_StateTransition16	フローチャートを図 7-27 に示します。 (*1)

(*1) CiA402FSA(図 7-20)にて規定されている状態遷移 1~16 が発生した時に呼び出されます。

表 7-40. CiA402 向け関数一覧 (samplecia402.c)

関数名	説明
APPL_MOTOR_MotionControl_Main	周期的に実行される CiA402_Application 関数から呼ばれるモータ制御関数。 フローチャートを図 7-28 に示します。
check_position_limit	現在位置が Limit を超えてないか確認する関数。 フローチャートを図 7-29 に示します。
int32_to_real32	int32 から real32(float)に変換する関数。
uint32_to_real32	uint32 から real32(float)に変換する関数。
rpm_to_rads	rpm から rad に変換する関数。
update_actual_value	モータの現在位置・速度を取得する関数。 フローチャートを図 7-30 に示します。

process_error_state	モータエラー発生時に呼び出される関数。 フローチャートを図 7-31 に示します。
process_profile_position_mode	pp モード時に呼び出される関数。 フローチャートを図 7-32 に示します。
process_homing_mode	hm モード時に呼び出される関数。 フローチャートを図 7-33 に示します。
process_cyclic_position_mode	csp モード時に呼び出される関数。 フローチャートを図 7-34 に示します。
process_cyclic_velocity_mode	csv モード時に呼び出される関数。 フローチャートを図 7-35 に示します。
APPL_GetDeviceID	EtherCAT ID を取得する関数。

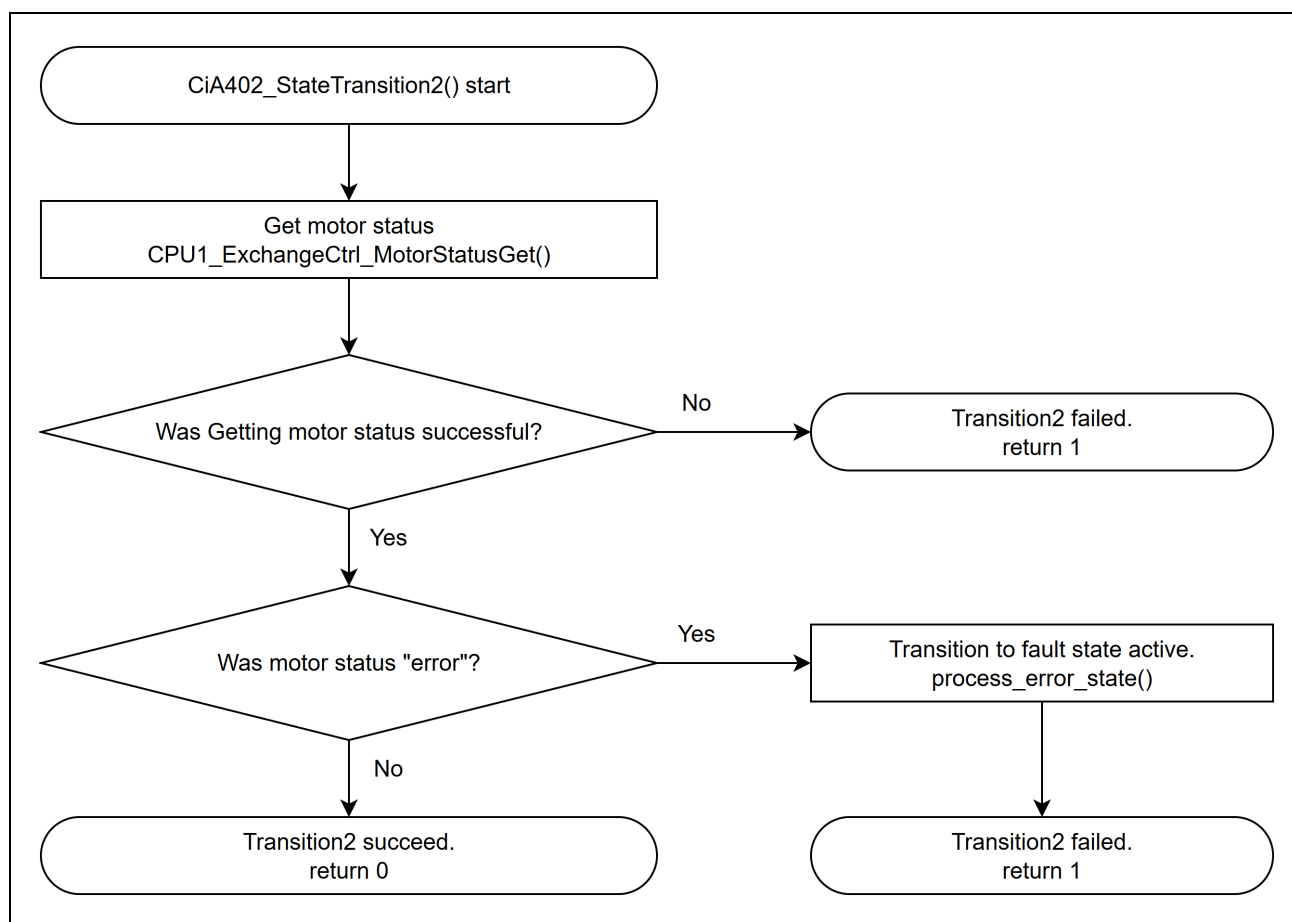


図 7-21. CiA402_StateTransition2 関数フローチャート

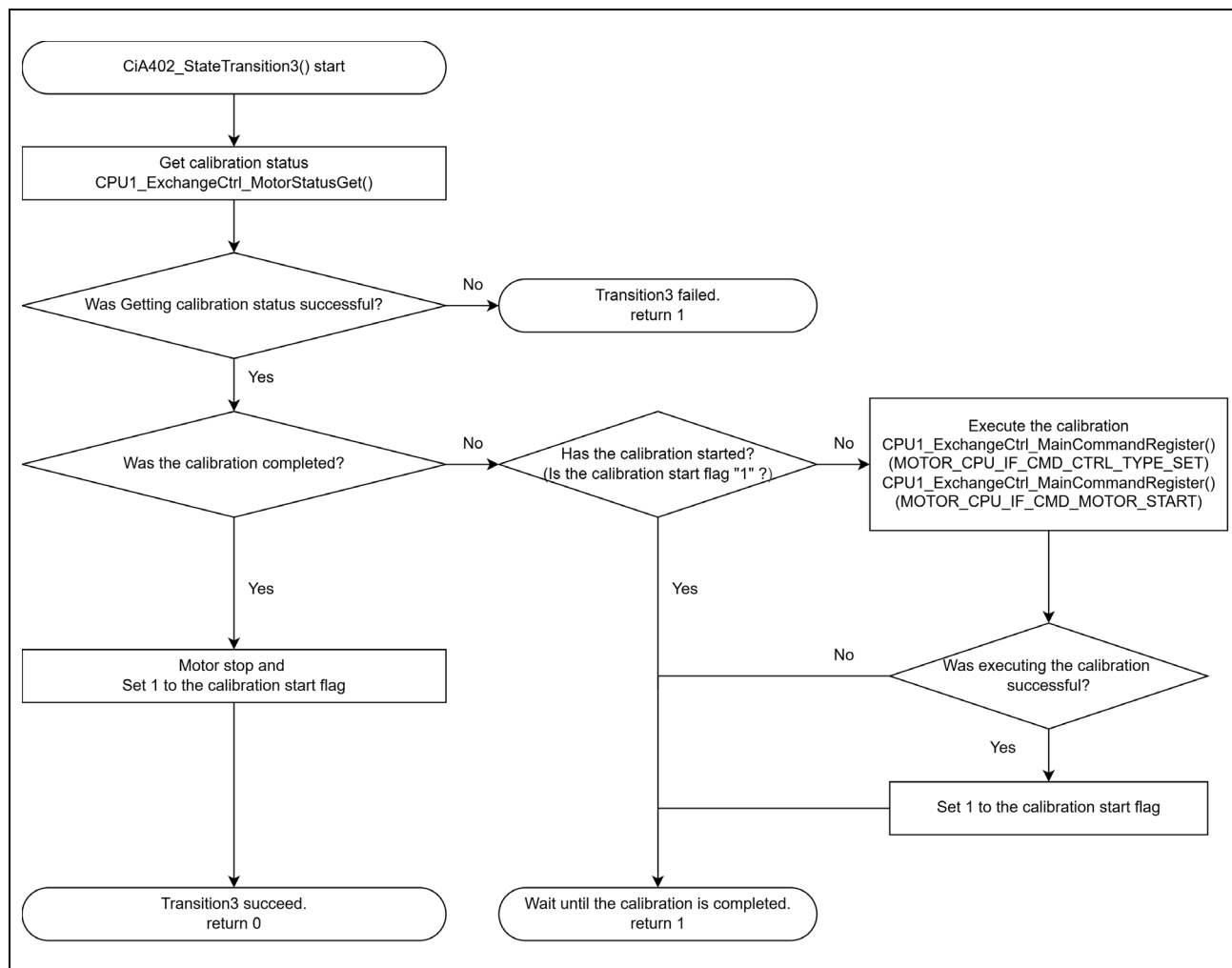


図 7-22. CiA402_StateTransition3 関数フローチャート

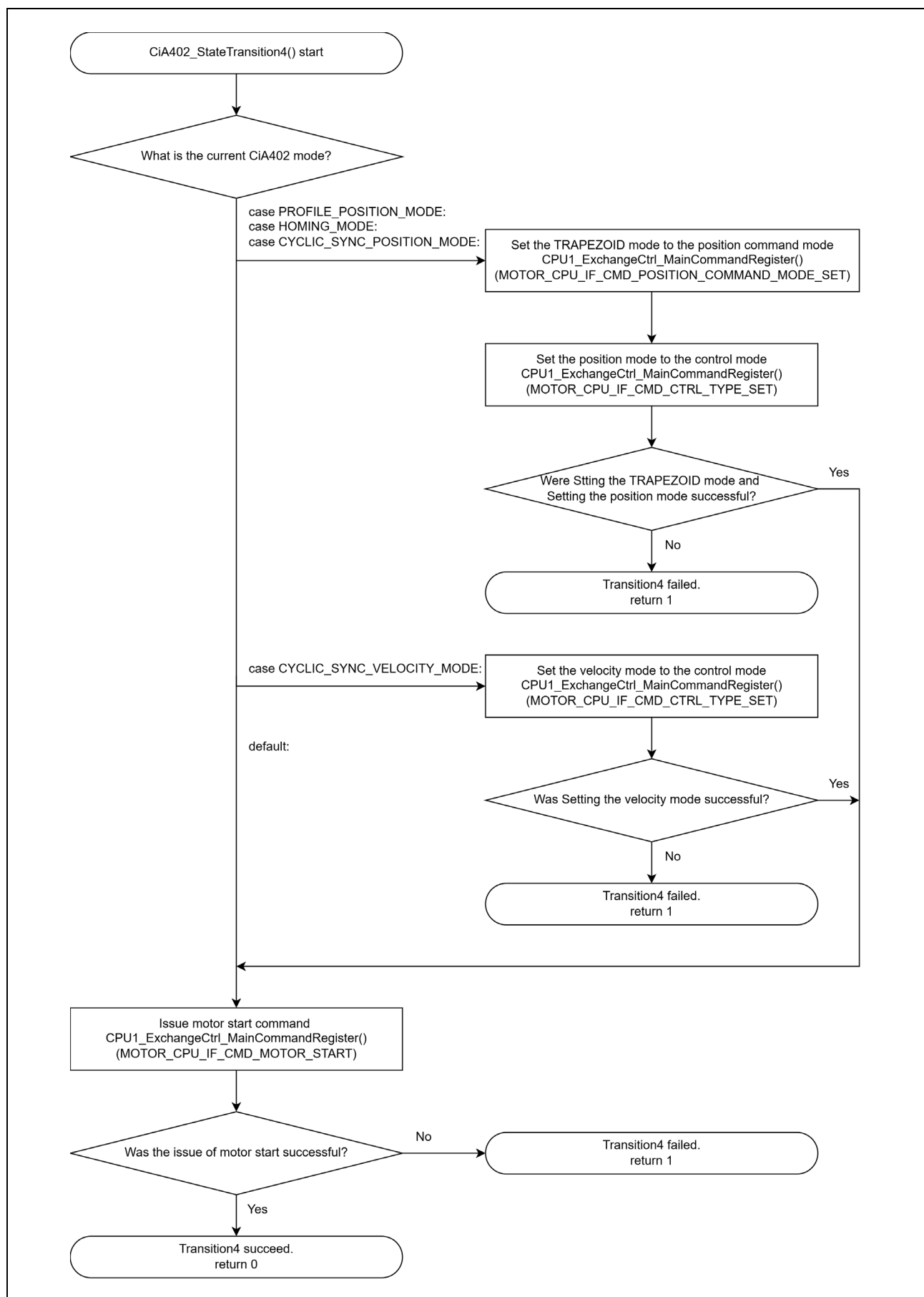


図 7-23. CiA402_StateTransition4 関数フローチャート

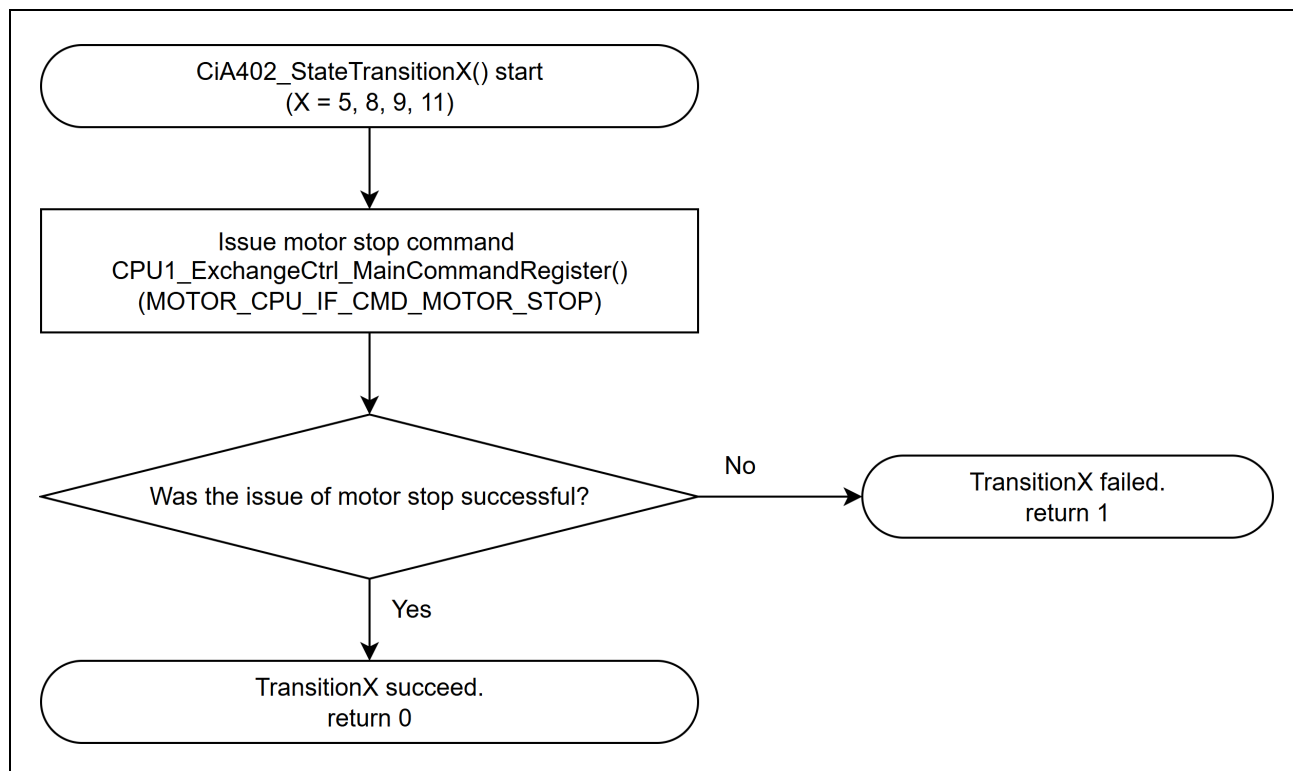


図 7-24. CiA402_StateTransition5, 8, 9, 11 関数フローチャート

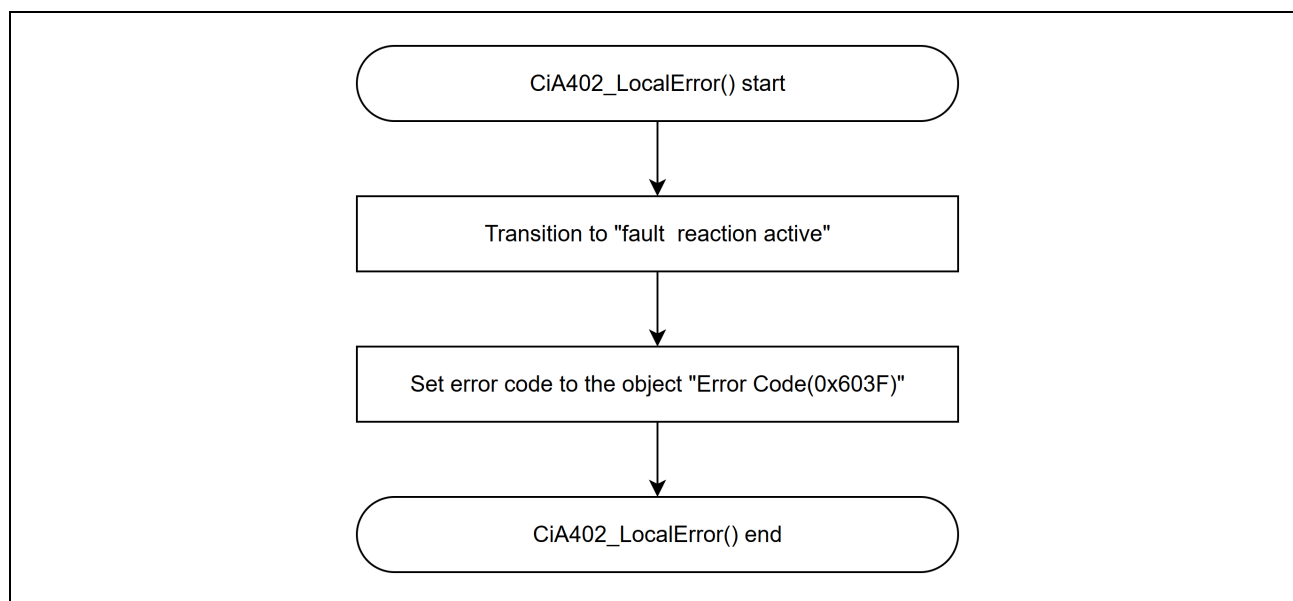


図 7-25. CiA402_LocalError 関数フローチャート

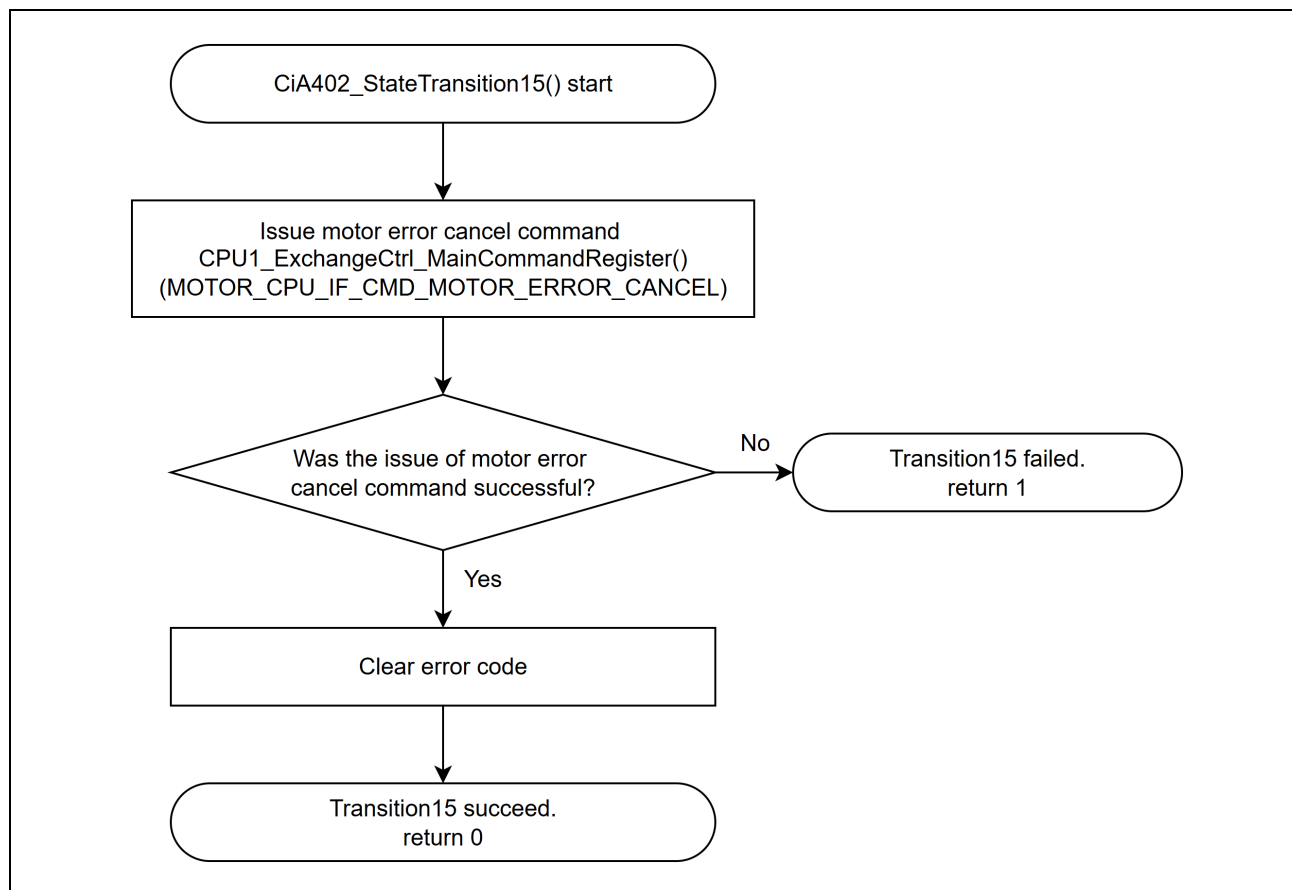


図 7-26. CiA402_StateTransition15 関数フローチャート

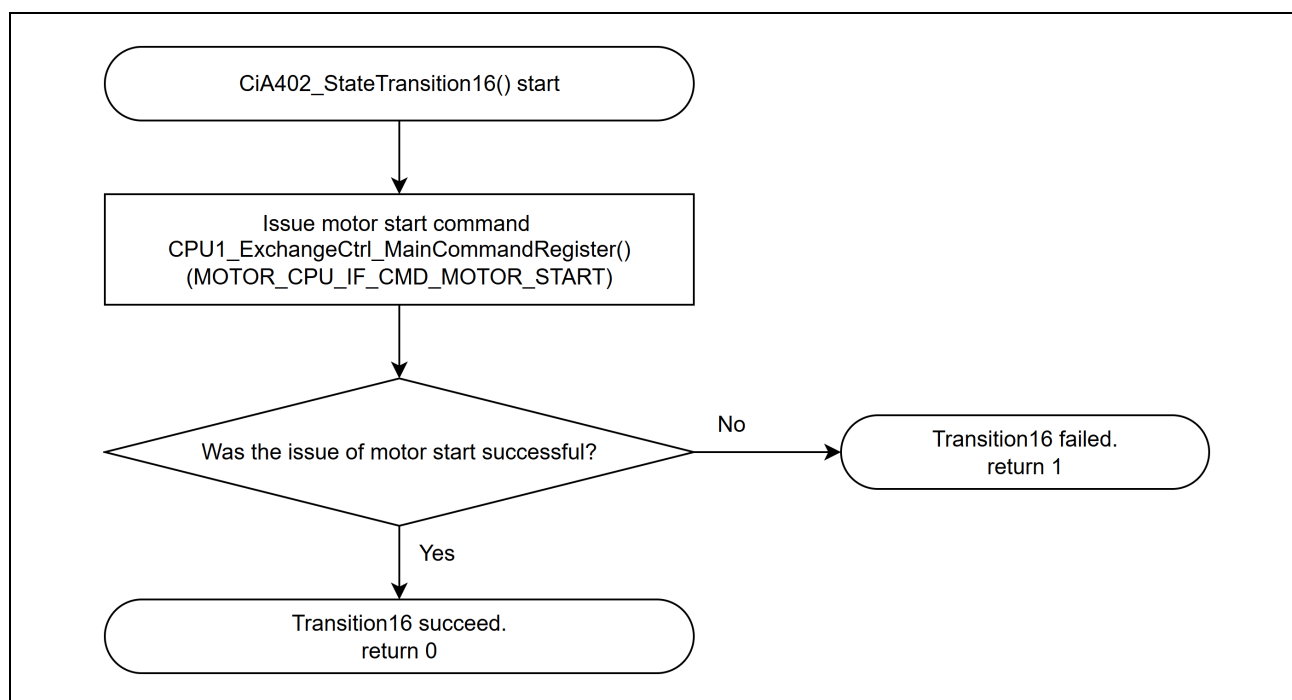


図 7-27. CiA402_StateTransition16 関数フローチャート

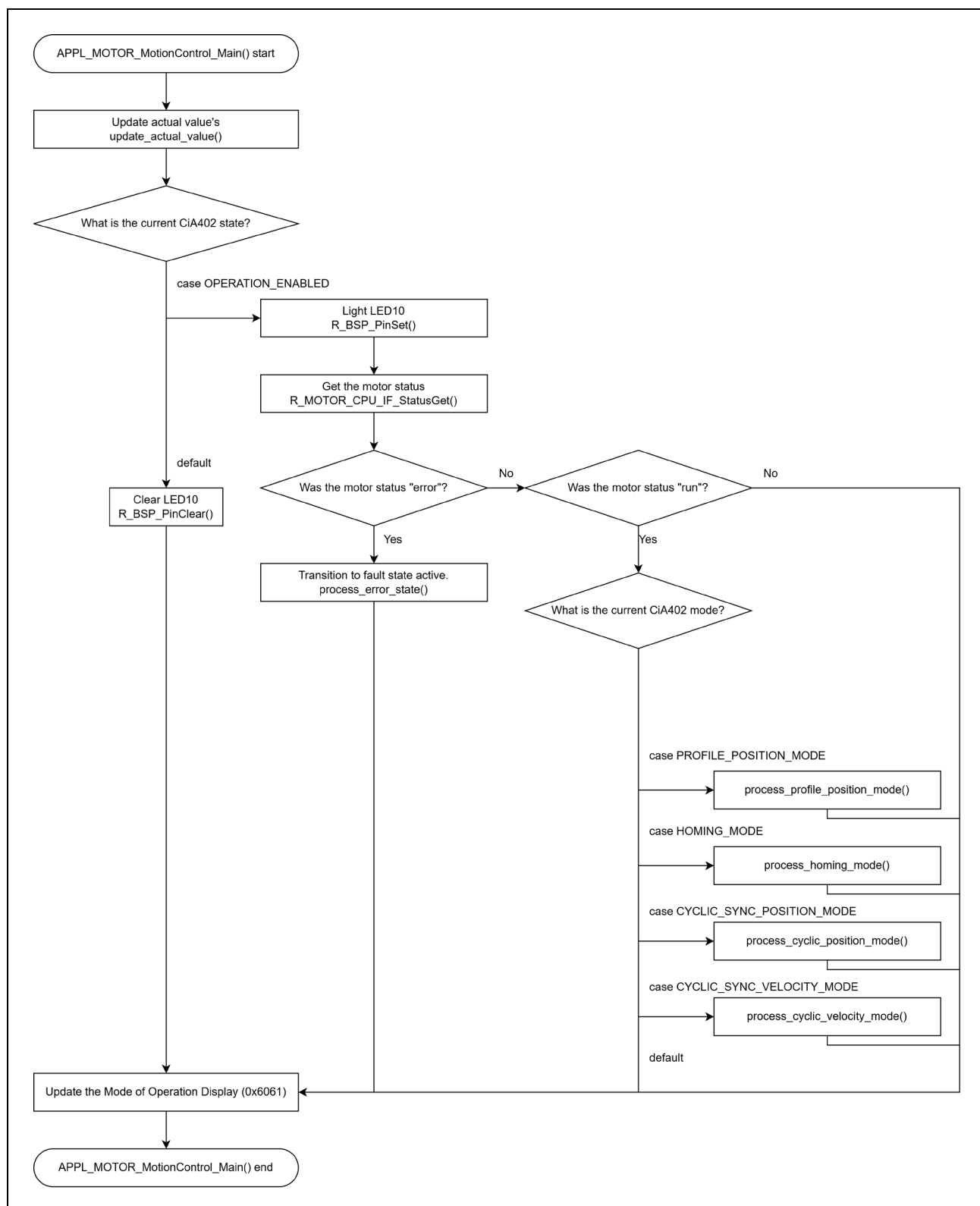


図 7-28. APPL_MOTOR_MotionControl_Main 関数フローチャート

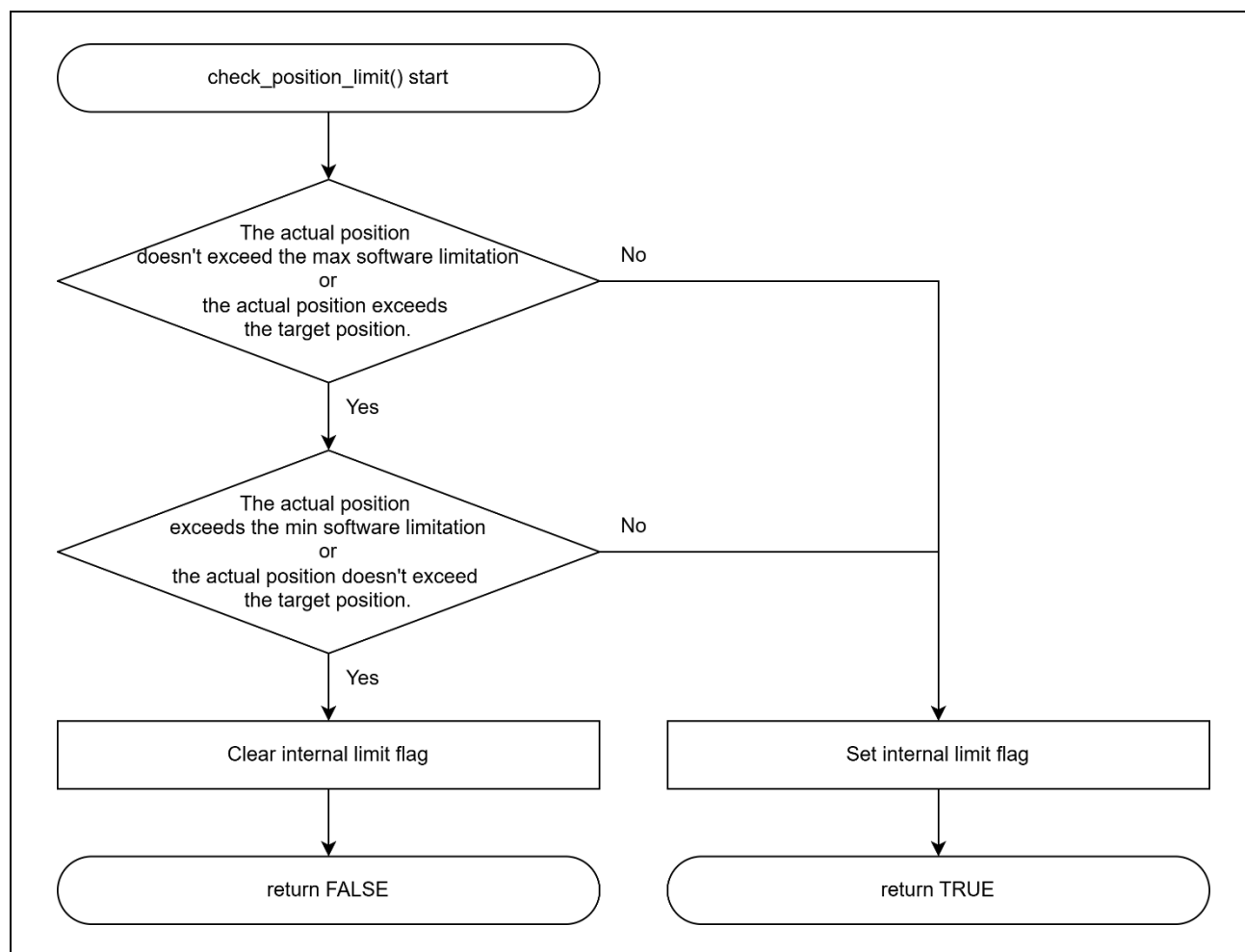


図 7-29. check_position_limit 関数フローチャート

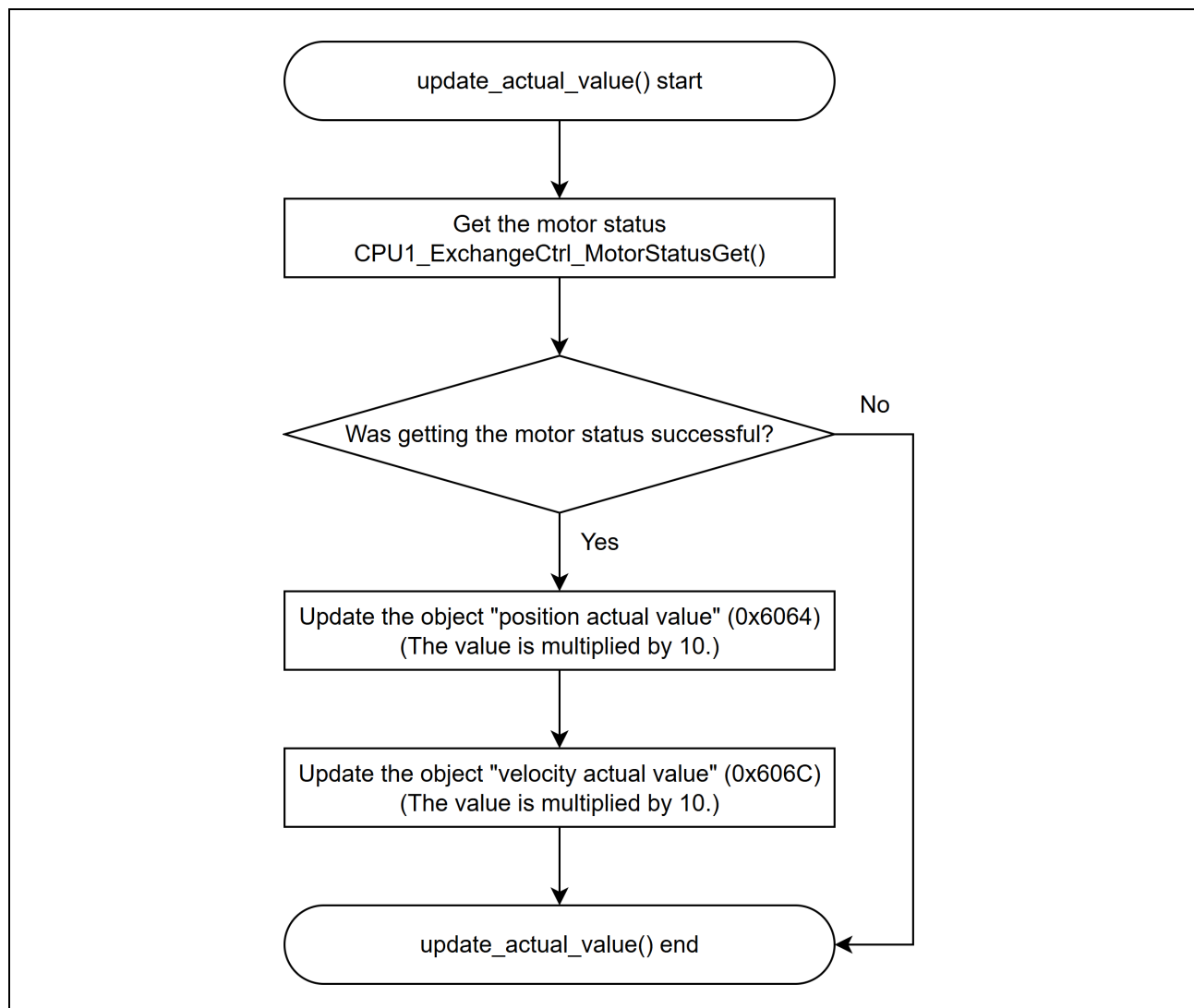


図 7-30. update_actual_value 関数フローチャート

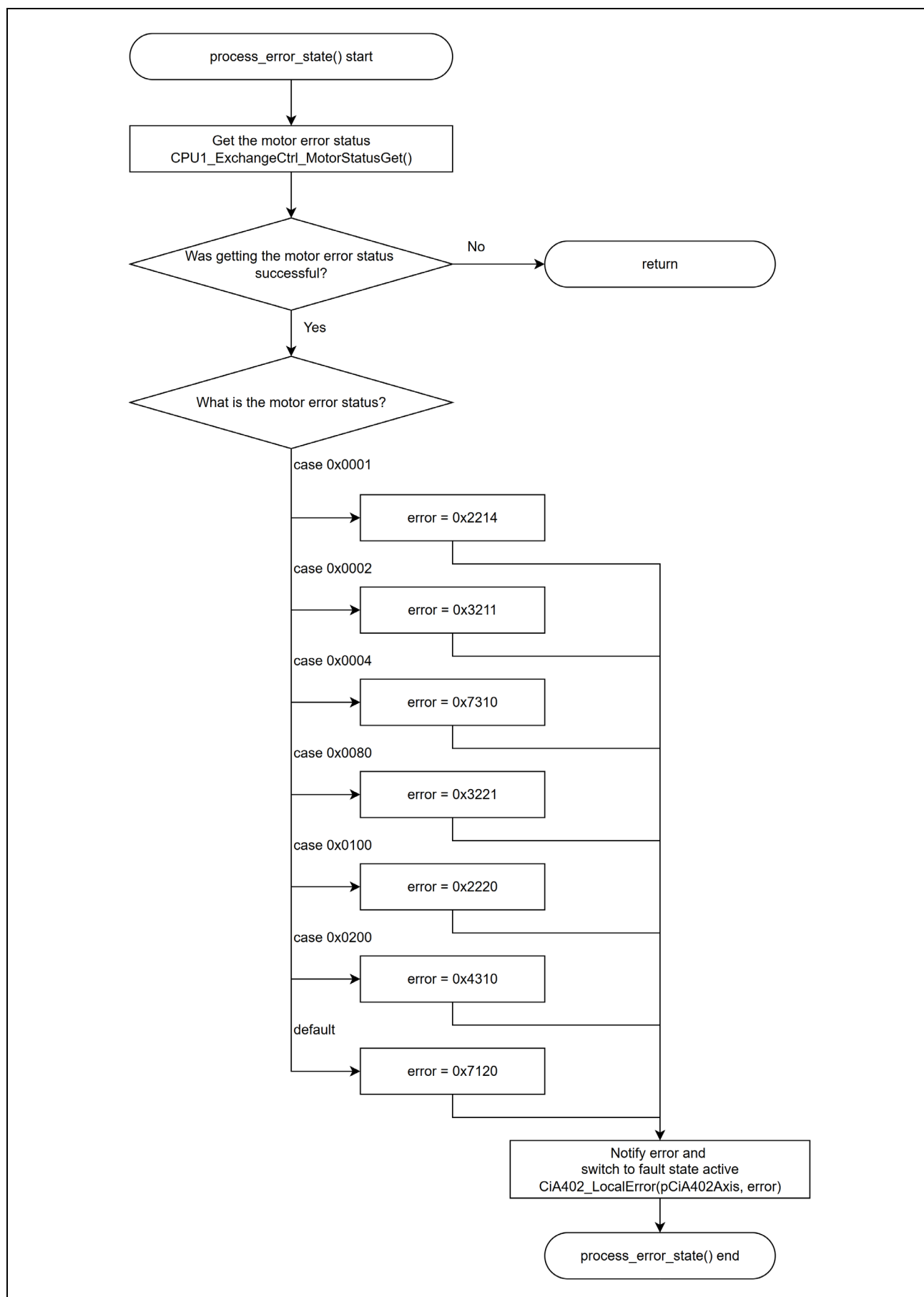


図 7-31. process_error_state 関数フローチャート

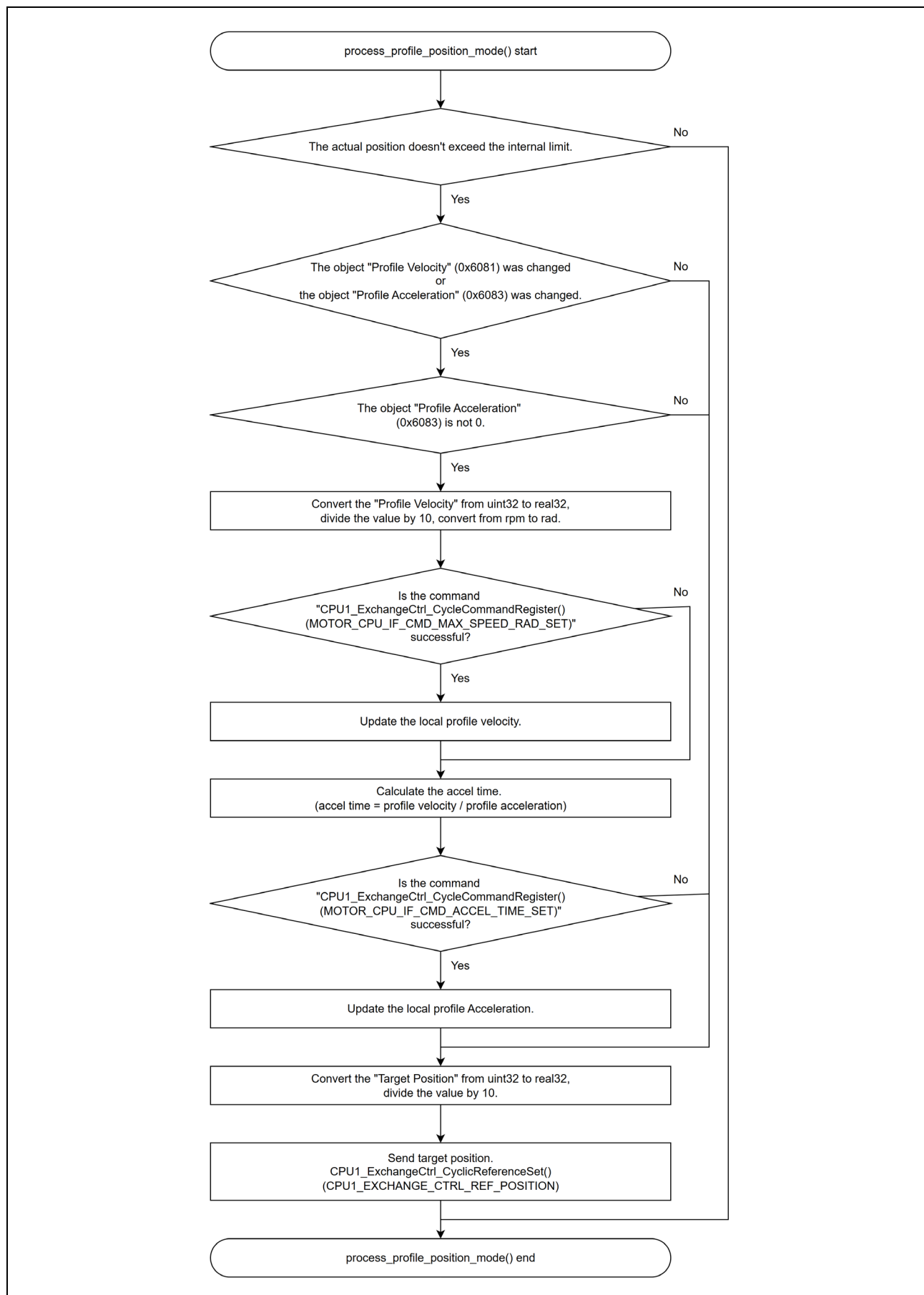


図 7-32. process_profile_position_mode 関数フローチャート

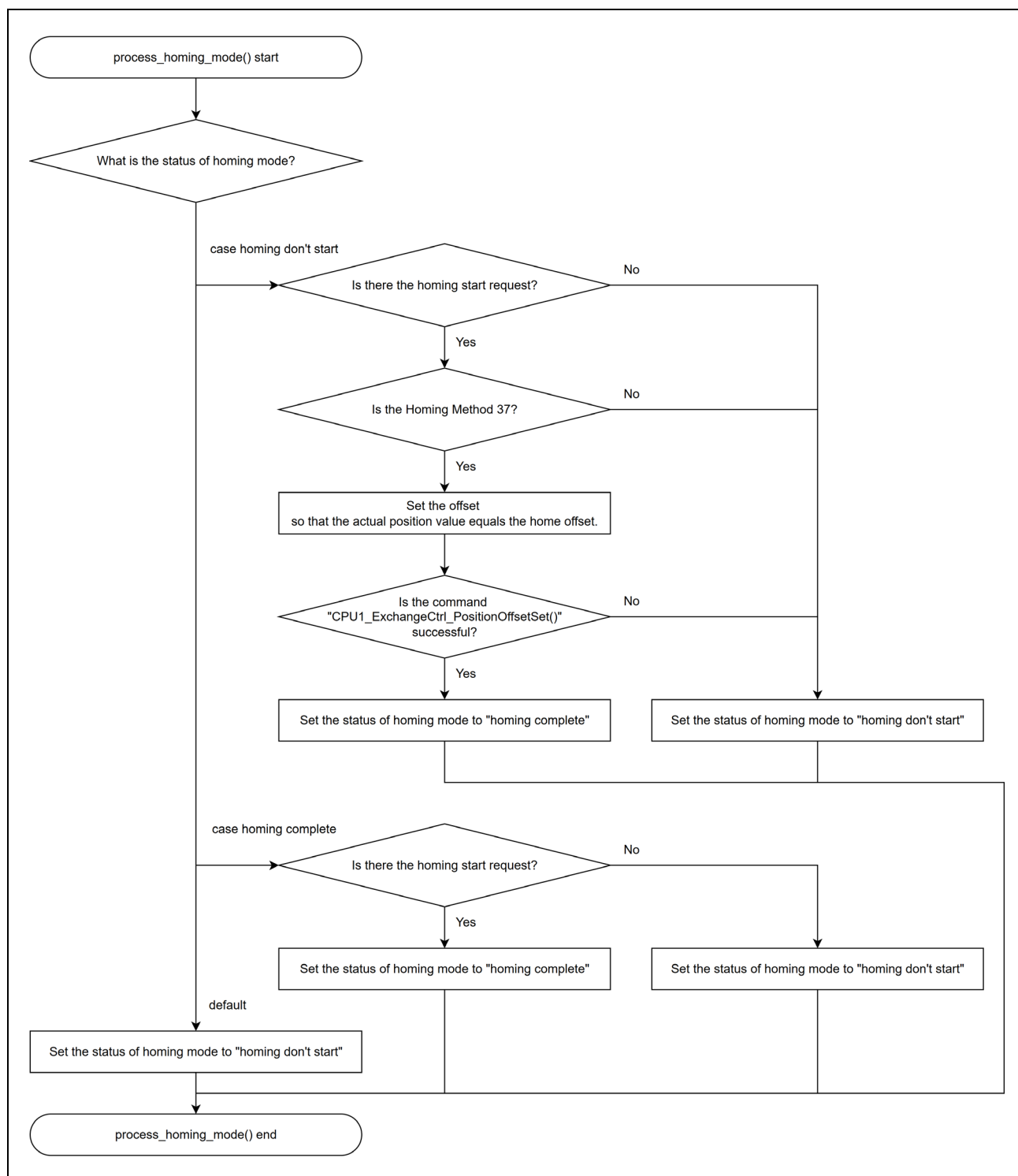


図 7-33. process_homing_mode 関数フローチャート

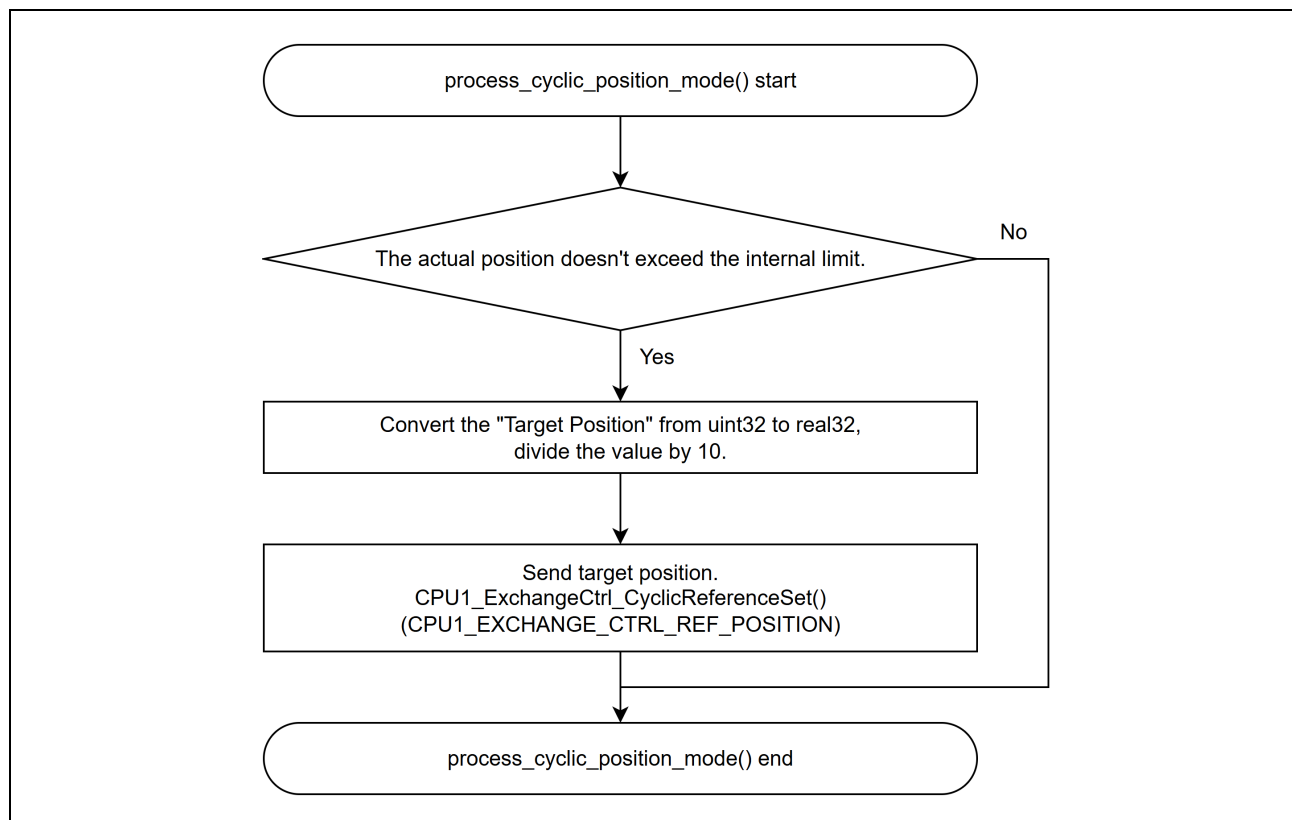


図 7-34. process_cyclic_position_mode 関数フローチャート

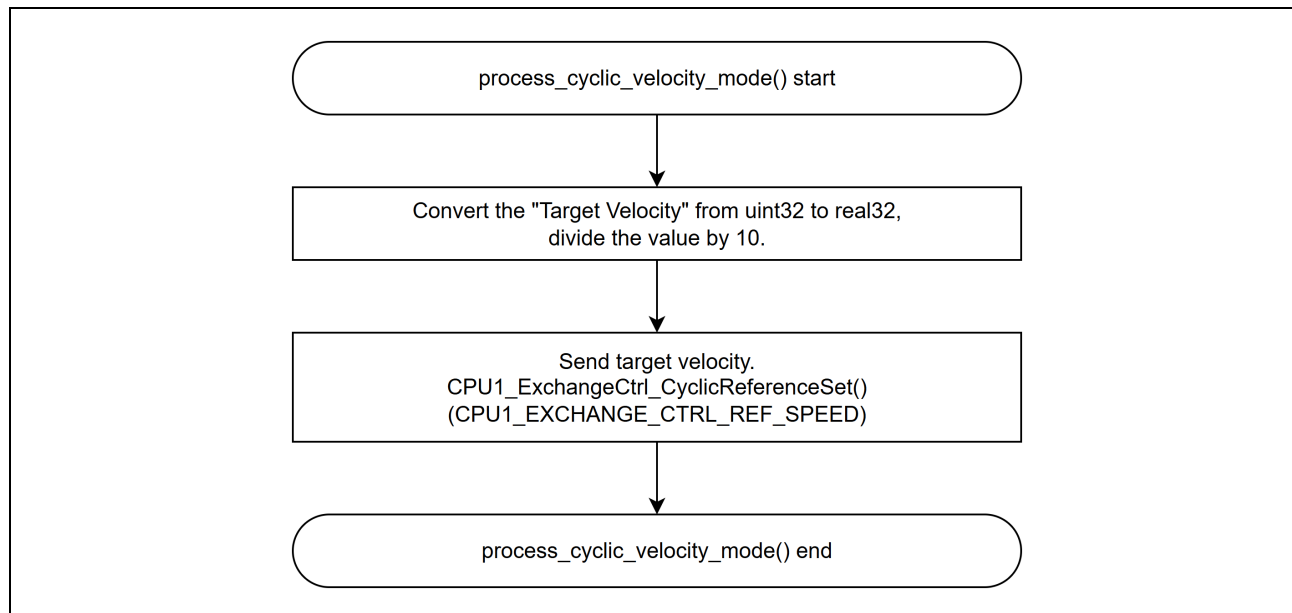


図 7-35. process_cyclic_velocity_mode 関数フローチャート

7.3.5.9 マクロ定義

ここでは、本サンプルソフトウェアにて実装されているマクロ定義について説明します。SSC Tool が生成する EtherCAT SubDevice Stack Code に実装されているマクロ定義については、

” Application Note ET9300 (EtherCAT SubDevice Stack Code)”を参照してください。

表 7-41. CiA402 Drive Profile 向けマクロ定義一覧 (samplecia402.c)

マクロ名	定義値	備考
CFG_POS_MODE	MOTOR_CPU_IF_POS_MODE_TRAPEZOID	モータ位置制御モードの設定

7.3.5.10 SSC tool 生成後ファイルからの変更点

ここでは、SSC Tool が生成する EtherCAT SubDevice Stack Code から変更した内容について述べます。変更前のソースコードの詳細は、”Application Note ET9300 (EtherCAT SubDevice Stack Code)”を参照してください。

表 7-42. SSC tool 生成後ファイルからの変更点一覧

ファイル名	変更点	詳細
cia402appl.c	インクルードパスを追加	hal_data.h、samplecia402.h、cpu1_exchange_ctrl.h を追加
	CiA402_Init 関数の処理を変更	構造体 TCiA402Axis に追加したメンバの初期化処理を追加 追加したオブジェクトの初期化処理を追加
	CiA402_StateMachine 関数の処理を変更	CiA402 状態遷移時に表 7-39 で示される各 CiA402 遷移関数が呼ばれるように変更
	CiA402_LocalError 関数および CiA402_DummyMotionControl 関数を削除	CiA402_LocalError 関数は samplecia402.c で定義する
	CiA402_Application 関数の処理を変更	CiA402 状態遷移時に表 7-39 で示される各 CiA402 遷移関数が呼ばれるように変更 DC モードだけでなく SM モードでもモータ動作が可能ないように変更 CiA402_DummyMotionControl 関数ではなく APPL_MOTOR_MotionControl_Main 関数を呼び出すように変更
	Write0xF030 関数の処理を変更	PDO Assign および PDO mapping の変更に合わせて処理を変更
	APPL_GenerateMapping 関数の処理を変更	プロセスデータサイズを適切に計算できるように処理を変更
	APPL_InputMapping 関数の処理を変更	PDO Assign および PDO mapping の変更に合わせて処理を変更

	APPL_OutputMapping 関数の処理を変更	
	APPL_Application 関数の処理を変更	EtherCAT の入力更新中 (bEcatInputUpdateRunning == TRUE) は、先頭で CPU1_ExchangeCtrl_MotorStatusRefresh 関数、末尾で CPU1_ExchangeCtrl_MotorRequestFlush 関数を呼び出すように変更
	APPL_GetDeviceID 関数を削除	APPL_GetDeviceID 関数は samplecia402.c で定義する
cia402appl.h	マクロ定義を追加	表 7-44 に追加したマクロ定義を示します。
	マクロ定義の設定値を変更	マクロ定義 MAX_AXES の設定値を 4 に変更
	オブジェクトディクショナリの定義を追加・変更	追加・変更後のオブジェクトディクショナリー一覧を表 7-45 に示します。
	構造体 TCiA402Axis にメンバを追加	追加したメンバを表 7-43 に示します。
	CiA402_LocalError 関数のプロトタイプ宣言を変更	引数に対象軸 (TCiA402Axis *pCiA402Axis) を追加
ecatappl.c	PDO_OutputMapping 関数の処理を変更	RxPDO の読出しを、HW_EscReadIsr 関数から 4 バイト単位で読み出す ecat_esc_read_pdo 関数に置き換え
ecatcoe.h	構造体のアラインメントを 1 に固定	-
ecatslv.c	StartInputHandler 関数の処理を変更	AL イベントマスクレジスタの 2 ビット目と 3 ビット目を 1 にする処理を削除
mailbox.h	構造体のアラインメントを 1 に固定	-
sdoserv.h	構造体のアラインメントを 1 に固定	-

表 7-43. 構造体 TCiA402Axis に追加したメンバー一覧 (cia402appl.h)

メンバ名	役割
u32AxisNumber	モータ軸番号を格納するメンバ
bCalibration	キャリブレーション実行フラグ
u32LocalProfVel	以前の profile velocity 値を格納するメンバ
u32LocalProfAcc	以前の profile acceleration 値を格納するメンバ

表 7-44. 追加したマクロ定義一覧 (cia402appl.h)

マクロ名	定義値	備考
STATUSWORD_HOMING_MASK	0x2400	ステータスワードのホーミング用ビットマスク
STATUSWORD_HOMING_NOT_STARTED	0x0400	ホーミング処理がまだスタートしていない
STATUSWORD_HOMING_COMPLETED	0x2000	ホーミング処理が完了した
CONTROLWORD_HOMING_START	0x0010	ホーミング処理スタートビット
HOMING_METHOD_37	0x0025	Homing Method 37
PP_MODULE_ID	0x00419800	pp モードを示す ID
HM_MODULE_ID	0x00519800	hm モードを示す ID
ENTRIES_1600	5	PDO 0x1600 のマッピング数
ENTRIES_1601	6	PDO 0x1601 のマッピング数
ENTRIES_1602	2	PDO 0x1602 のマッピング数
ENTRIES_1603	2	PDO 0x1603 のマッピング数
ENTRIES_1A00	5	PDO 0x1A00 のマッピング数
ENTRIES_1A01	5	PDO 0x1A01 のマッピング数
ENTRIES_1A02	5	PDO 0x1A02 のマッピング数
ENTRIES_1A03	5	PDO 0x1A03 のマッピング数

7.3.5.11 Object Dictionary

表 7-45. オブジェクトディクショナリー一覧

Index	ObjectCode	SI	DataType	Access	Object Name
0x1nnn Communication Area					
0x1000	VAR	-	UDINT	RO	Device type
0x1001	VAR	-	USINT	RO	Error Register
0x1008	VAR	-	STRING(43)	RO	Device Name
0x1009	VAR	-	STRING(4)	RO	Manufacturer Hardware Version
0x100A	VAR	-	STRING(4)	RO	Manufacturer Software Version
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
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	pp RxPDO
		1-5	UDINT	RO	SubIndex 00x
0x1601	RECORD	0	USINT	RO	hm RxPDO
		1-6	UDINT	RO	SubIndex 00x
0x1602	RECORD	0	USINT	RO	csp RxPDO
		1-2	UDINT	RO	SubIndex 00x
0x1603	RECORD	0	USINT	RO	csv RxPDO
		1-2	UDINT	RO	SubIndex 00x
0x1A00	RECORD	0	USINT	RO	pp TxPDO
		1-5	UDINT	RO	SubIndex 00x
0x1A01	RECORD	0	USINT	RO	hm TxPDO
		1-5	UDINT	RO	SubIndex 00x
0x1A02	RECORD	0	USINT	RO	csp TxPDO
		1-5	UDINT	RO	SubIndex 00x
0x1A03	RECORD	0	USINT	RO	csv TxPDO
		1-5	UDINT	RO	SubIndex 00x
0x1C00	ARRAY	0	USINT	RO	Sync Manager 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
0x6nnn CiA402 Drive Profile Area (ETG.6010)					
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
0x607F	VAR	-	UDINT	RW	Max profile velocity
0x6080	VAR	-	UDINT	RW	Max motor speed
0x6081	VAR	-	UDINT	RW	Profile velocity
0x6083	VAR	-	UDINT	RW	Profile acceleration
0x6084	VAR	-	UDINT	RW	Profile deceleration
0x6085	VAR	-	UDINT	RW	Quick stop deceleration
0x6091	RECORD	0	UDINT	RW	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
0x609A	VAR	-	UDINT	RW	Homing acceleration
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
0xF000	RECORD	0	USINT	RO	Modular Device Profile
		1	UINT	RO	Index distance
		2	UINT	RO	Maximum number of modules
0xF010	ARRAY	0	USINT	RO	Module List
		1-4	UDINT	RO	SubIndex 00x
0xF030	ARRAY	0	USINT	RO	Configured Module Ident List
		1-4	UDINT	RO	SubIndex 00x
0xF050	ARRAY	0	USINT	RO	Detected modules
		1-4	UDINT	RO	SubIndex 00x

* This is RW only if its status is PreOP.

7.3.6 Motor CPU I/F

Motor CPU I/F モジュールは、Shared Memory Driver を使用して Motor CPU (RZ/T2N CPU0)とモータ制御コマンドの送信・指令値の更新・モータステータスの取得を行うモジュールです。

本サンプルプログラムではこのモジュールを 4 つ使用することで 4 軸モータに対応しています。

7.3.6.1 モジュール構成

モジュール構成図を図 7-36 に示します。

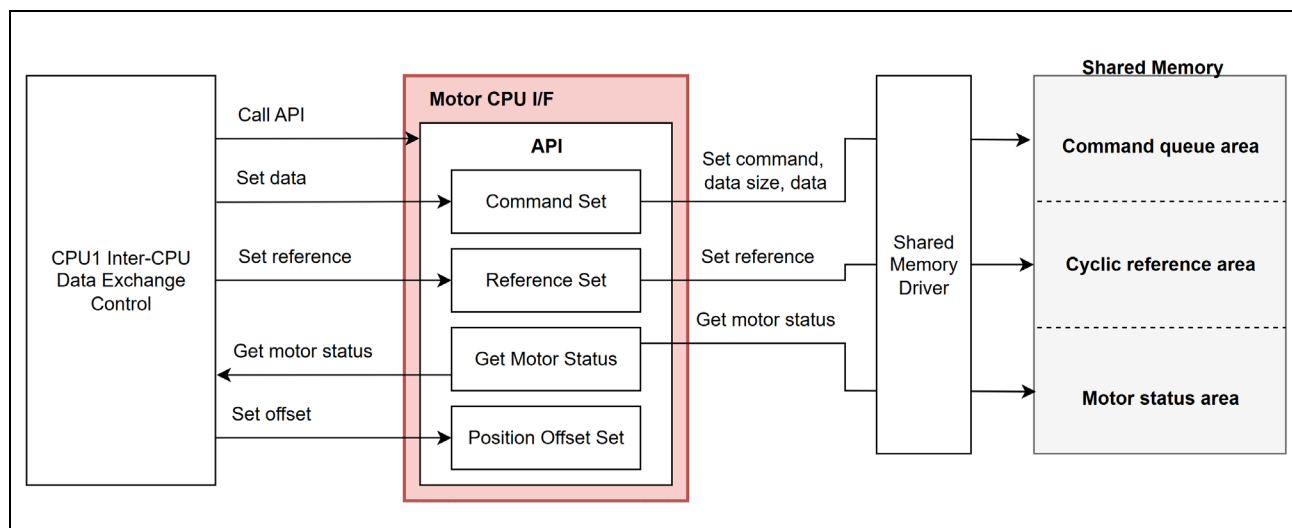


図 7-36. Motor CPU I/F モジュール構成図

本モジュールでは主に 4 種類の API を提供します。

(1) Command Set

この API は、Shared Memory の Command queue area にモータ制御コマンド・データサイズ・データを格納します。

(2) Reference Set

この API は、Shared Memory の Cyclic reference area に設定されている各種指令値を更新します。

(3) Get Motor Status

この API は、Shared Memory の Motor status area に格納されている各種モータステータスを取得します。

(4) Position Offset Set

この API は、Home 位置の設定を目的として用意されている API です。位置情報にオフセットを設定します。

7.3.6.2 Shared Memory 割り当て

7.2.3.2 章に記載されている表と同様です。

7.3.6.3 API

表 7-46. Motor CPU I/F API 一覧

API	説明
R_MOTOR_CPU_IF_Open	Motor CPU I/F の初期化を行います。 また、Shared Memory Driver の初期化を行います。 フローチャート：図 7-37
R_MOTOR_CPU_IF_Close	Motor CPU I/F を終了します。 また、Shared Memory Driver を終了します。 フローチャート：図 7-38
R_MOTOR_CPU_IF_Reset R_MOTOR_CPU_IF_MotorStart R_MOTOR_CPU_IF_MotorStop R_MOTOR_CPU_IF_MotorReset R_MOTOR_CPU_IF_MotorErrorCancel R_MOTOR_CPU_IF_ErrorSet R_MOTOR_CPU_IF_CtrlTypeSet R_MOTOR_CPU_IF_PositionCommandModeSet R_MOTOR_CPU_IF_RotorAngleOffsetSet R_MOTOR_CPU_IF_CurrentOffsetSet R_MOTOR_CPU_IF_CurrentOpenLoopSet R_MOTOR_CPU_IF_AccelTimeSet R_MOTOR_CPU_IF_PosProfMaxSpeedRadSet	Command Set API です。 Shared Memory の Command queue area にモータ制御コマンド・データサイズ・データを格納します。 フローチャート：図 7-39
R_MOTOR_CPU_IF_RefPositionSet R_MOTOR_CPU_IF_RefSpeedSet R_MOTOR_CPU_IF_RefTorqueSet	Reference Set API です。 Shared Memory の Cyclic reference area に設定されている各種指令値を更新します。 フローチャート：図 7-40
R_MOTOR_CPU_IF_PositionGet R_MOTOR_CPU_IF_SpeedGet R_MOTOR_CPU_IF_StatusGet R_MOTOR_CPU_IF_ErrorStatusGet R_MOTOR_CPU_IF_LoopModeStatusGet R_MOTOR_CPU_IF_InPositionFlagGet R_MOTOR_CPU_IF_OffsetCalibrationStatusGet	Get Motor Status API です。 Shared Memory の Motor status area に格納されている各種モータステータスを取得します。 フローチャート：図 7-41
R_MOTOR_CPU_IF_PositionOffsetSet	Position Offset Set API です。 位置情報にオフセットを設定します。 フローチャート：図 7-42

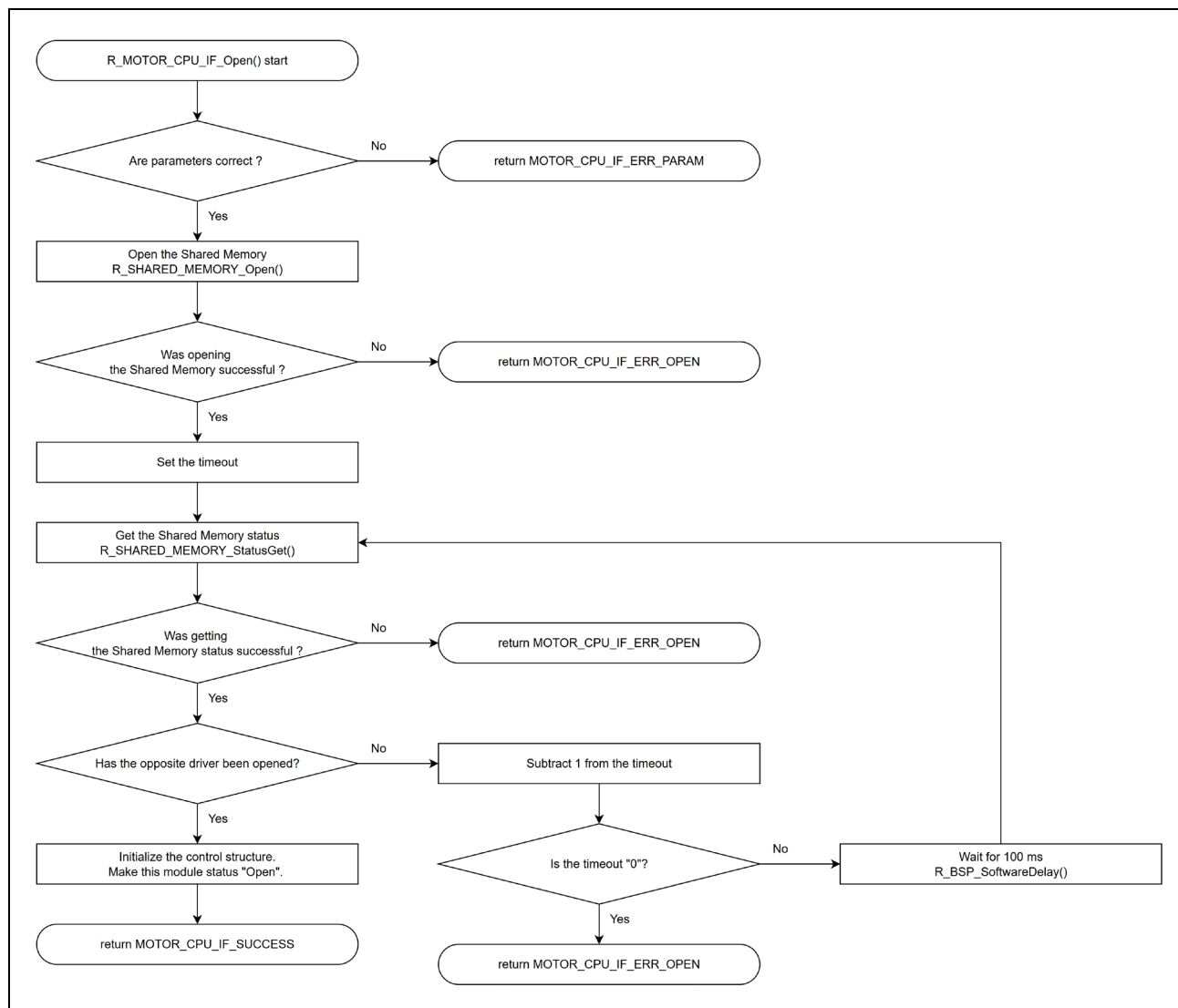


図 7-37. R_MOTOR_CPU_IF_Open 関数フローチャート

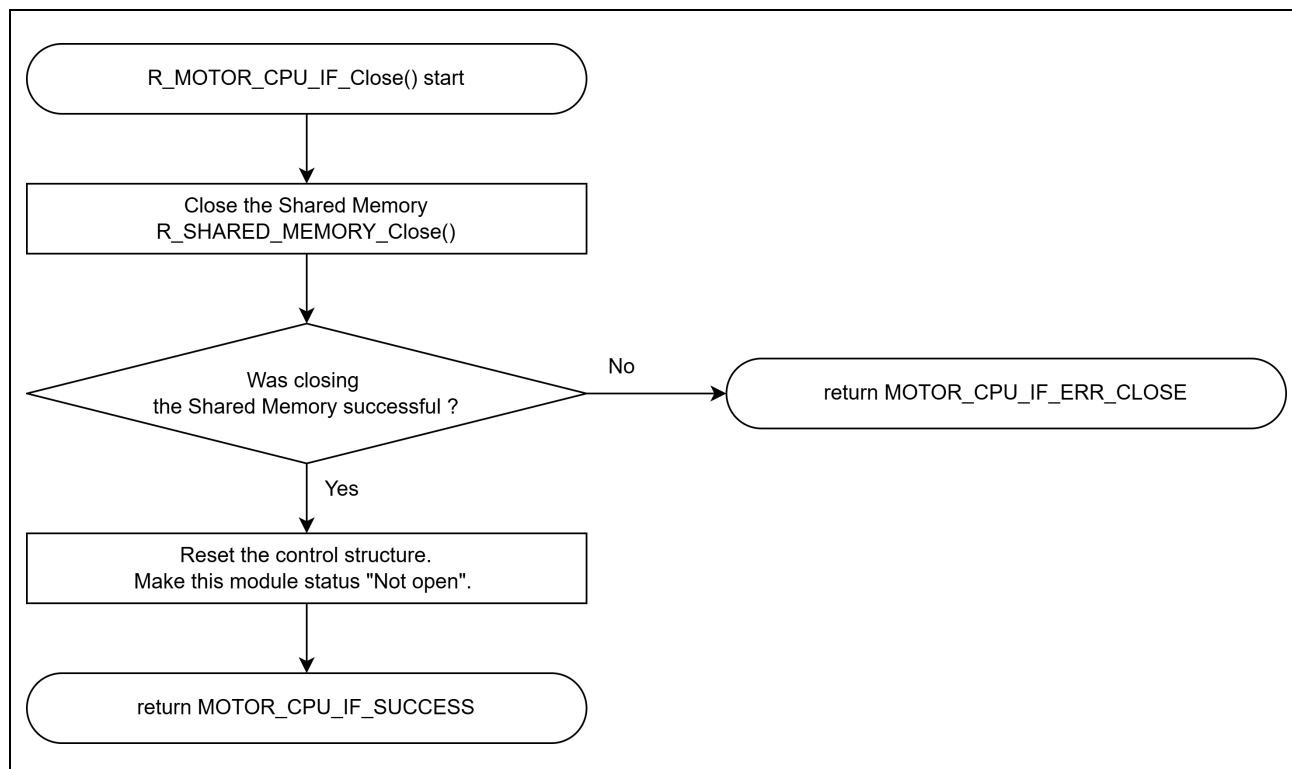


図 7-38. R_MOTOR_CPU_IF_Close 関数フローチャート

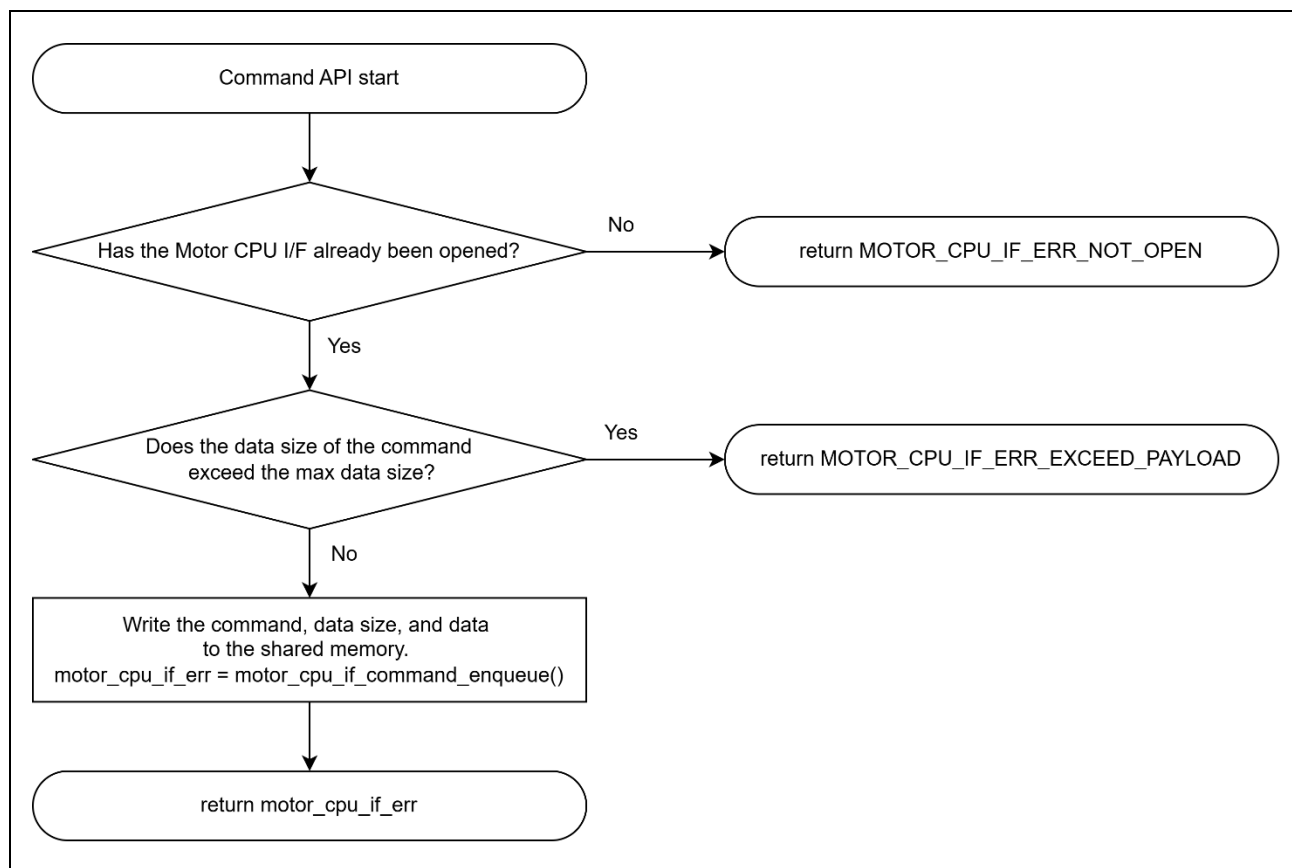


図 7-39. R_MOTOR_CPU_IF Command Set API フローチャート

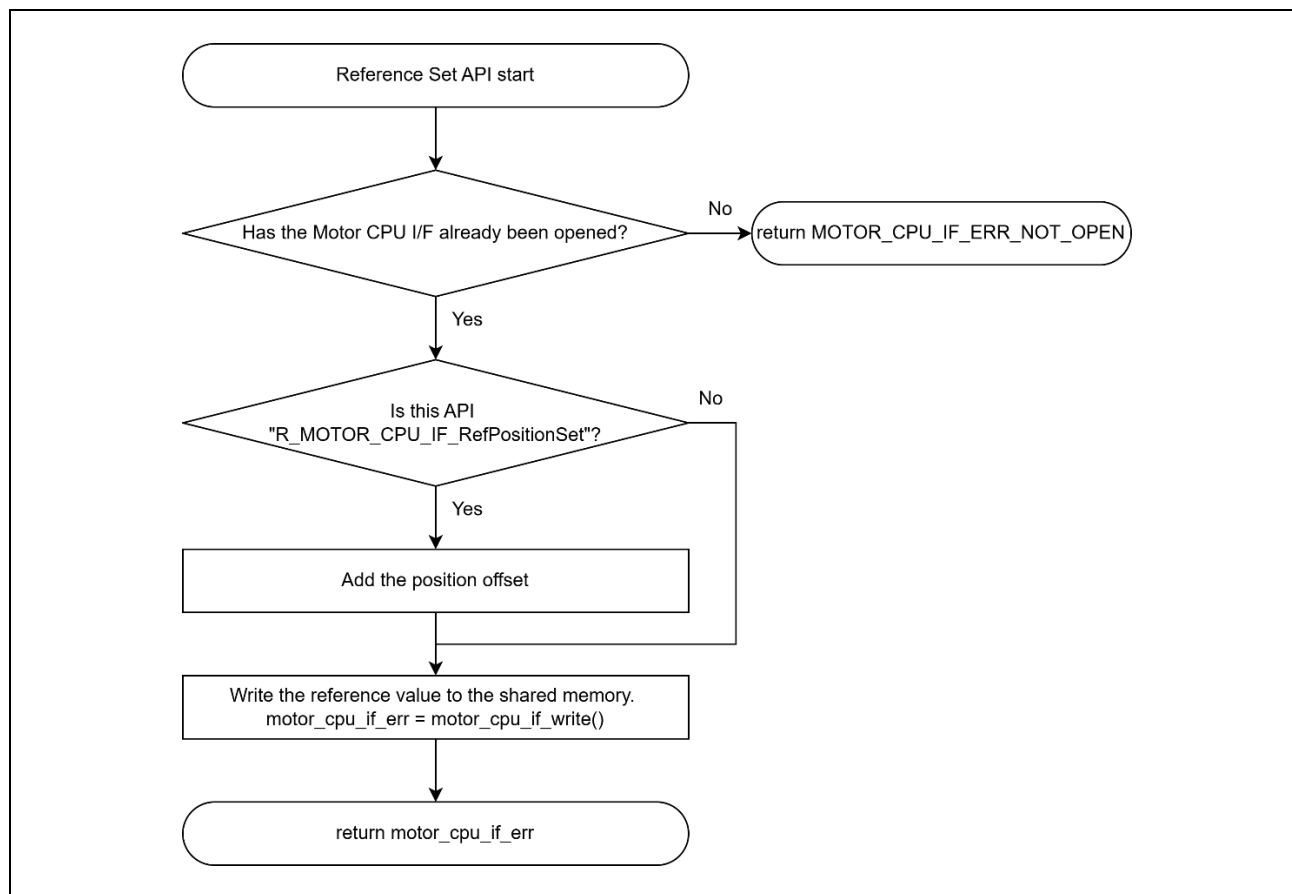


図 7-40. R_MOTOR_CPU_IF Reference Set API フローチャート

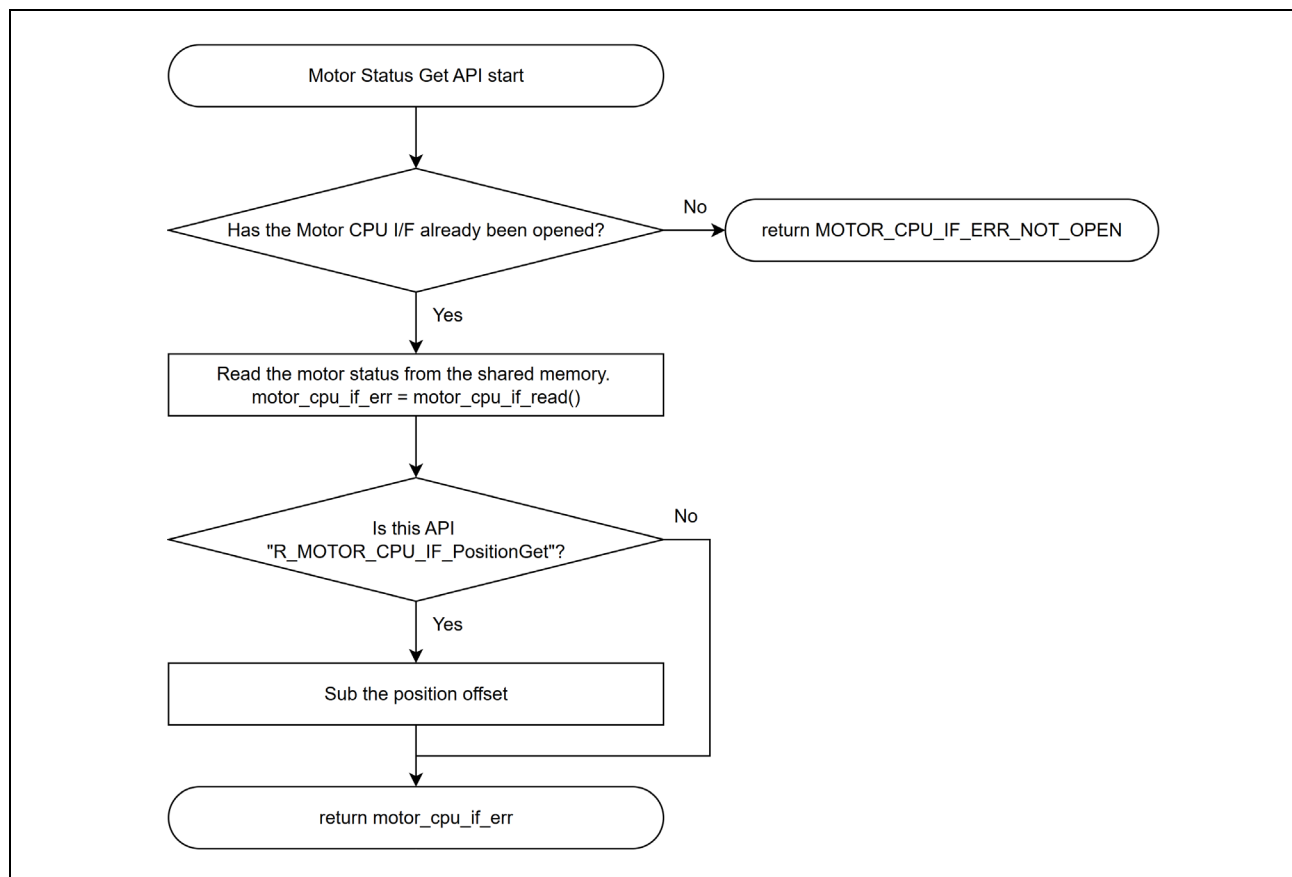


図 7-41. R_MOTOR_CPU_IF Motor Status Get API フローチャート

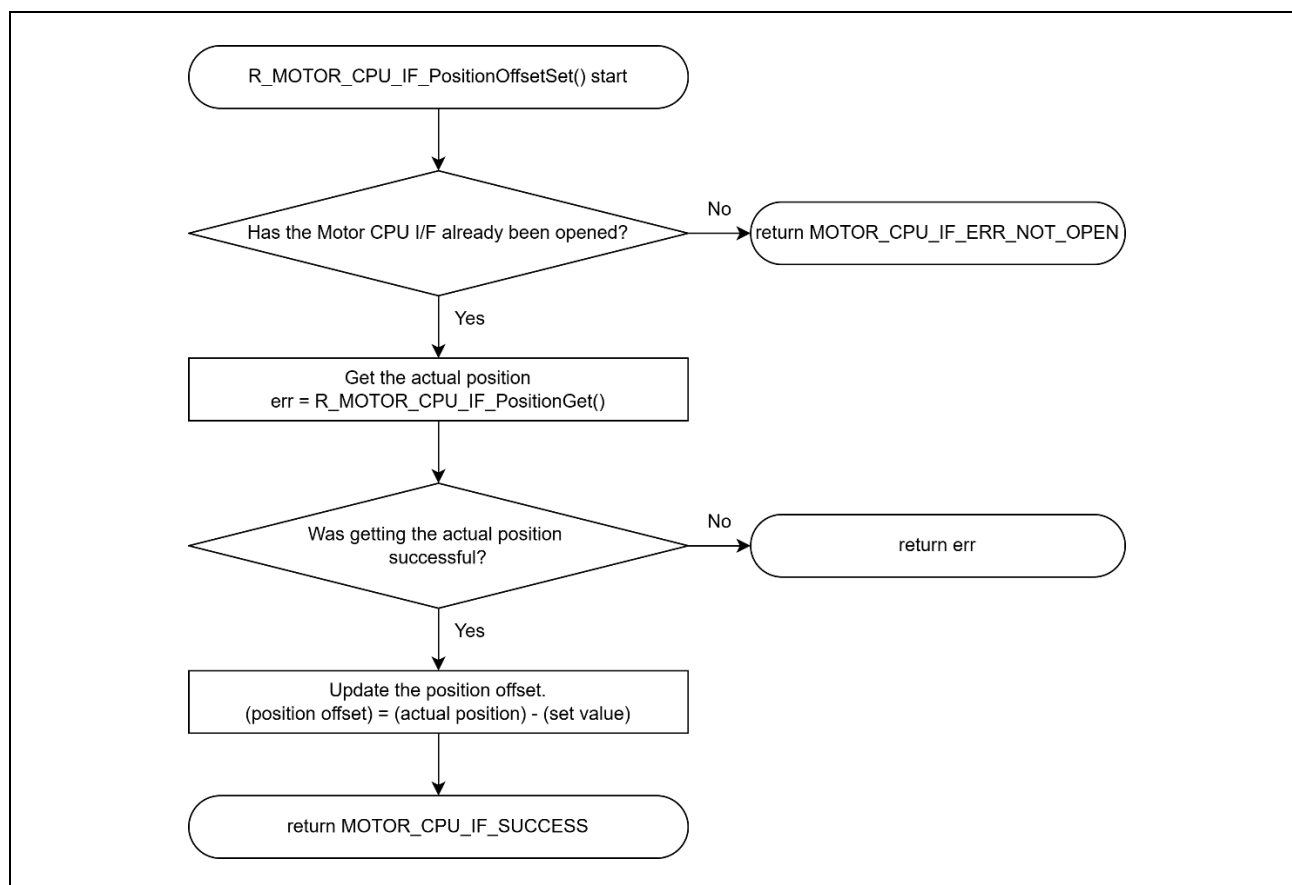


図 7-42. R_MOTOR_CPU_IF_PositionOffsetSet 関数フローチャート

7.3.6.4 プライベート関数

表 7-47. Motor CPU I/F プライベート関数 一覧

プライベート関数	説明
motor_cpu_if_write	Shared memory にデータを書き込みます。 フローチャート：図 7-43
motor_cpu_if_read	Shared memory からデータを読み取ります。 フローチャート：図 7-44
motor_cpu_if_command_enqueue	コマンドキューからコマンドを書き込みます。 フローチャート：図 7-45

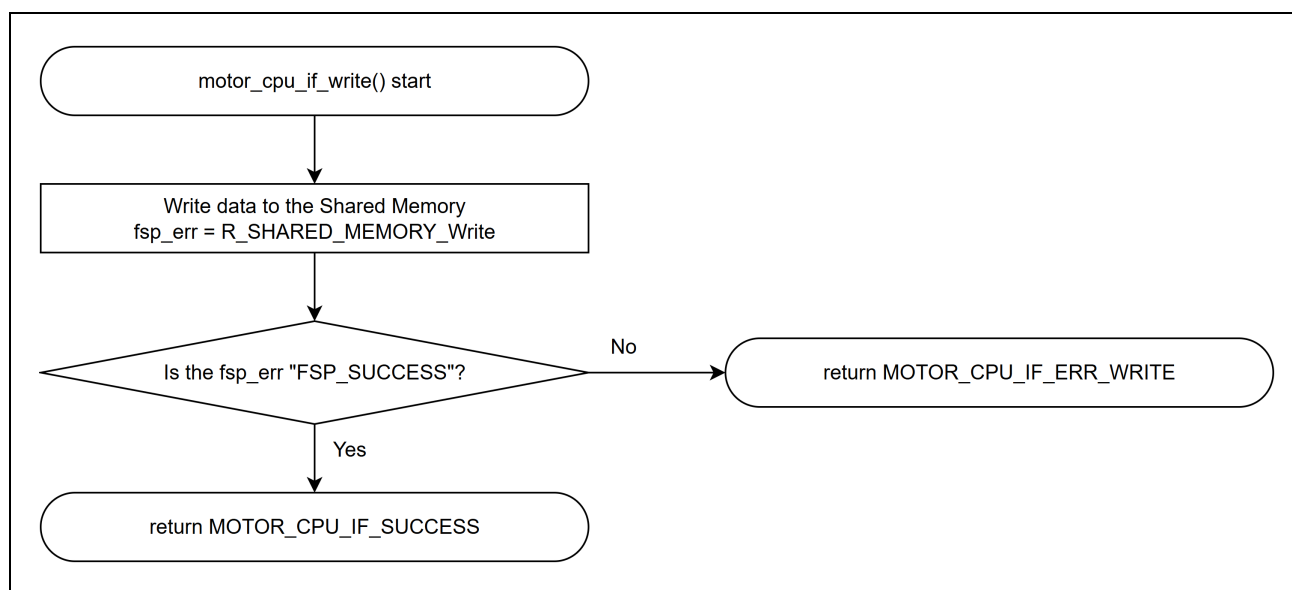


図 7-43. motor_cpu_if_write 関数フローチャート

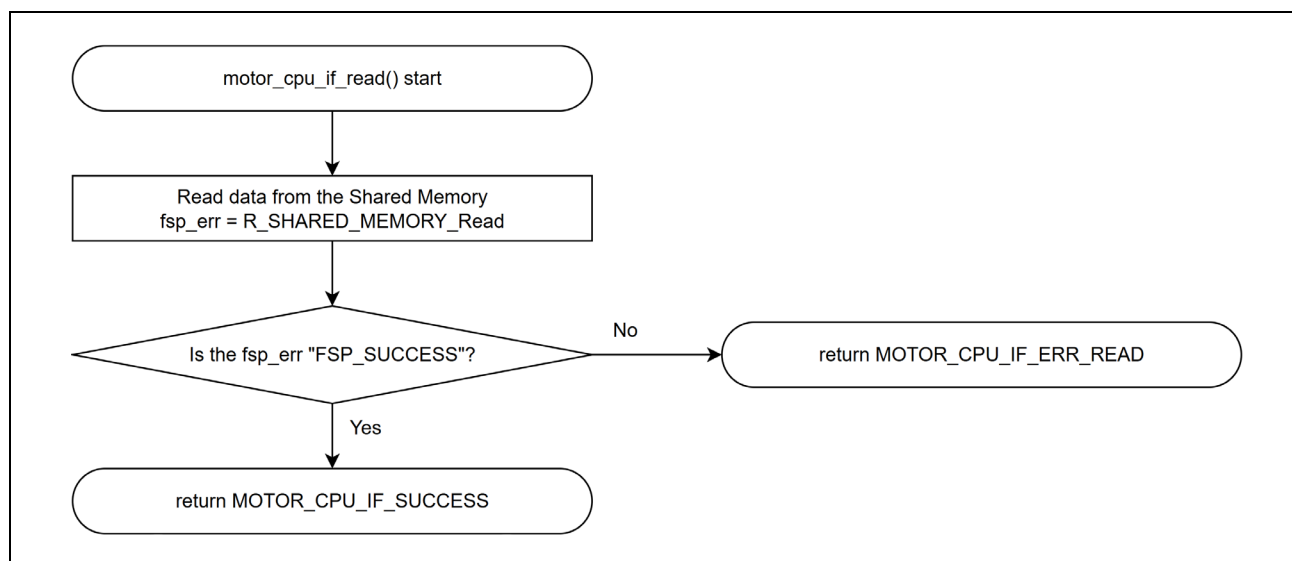


図 7-44. motor_cpu_if_read 関数フローチャート

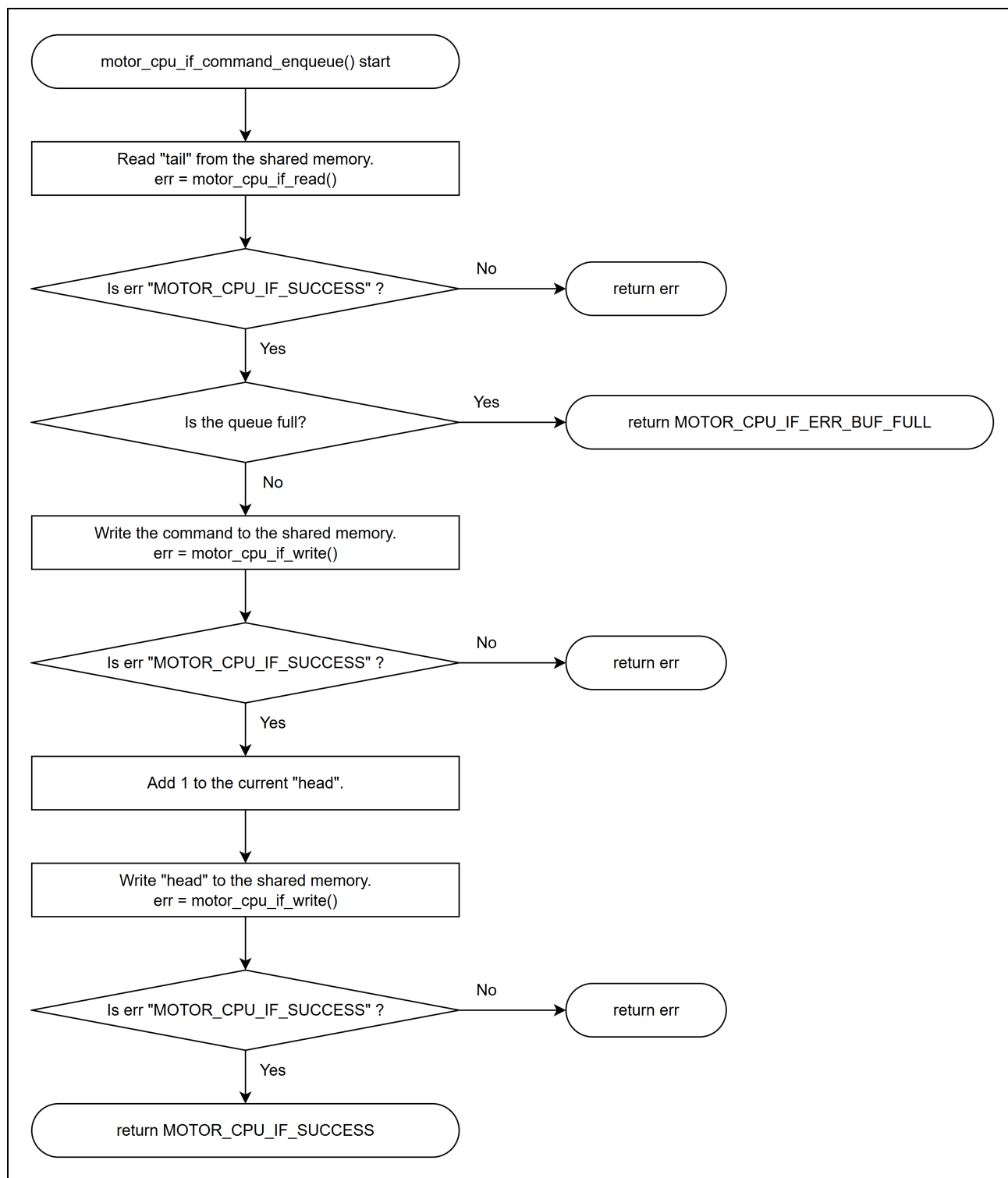


図 7-45. motor_cpu_if_command_enqueue 関数フローチャート

7.3.6.5 Callback 関数

表 7-48. Motor CPU I/F Callback 関数一覧 (r_motor_cpu_if.c)

関数名	説明
shared_memory_callback	これは Shared Memory Driver の Callback 関数です。 RZ/T2N CPU0 側で Shared Memory Driver が Shared Memory へ書き込むと、この割り込みが発生します。 Callback 関数内で行われる処理はありません。 必要に応じて処理の追加を検討して下さい。

7.3.6.6 マクロ定義

表 7-49. Motor CPU I/F マクロ定義一覧 (r_motor_cpu_if.h)

マクロ名	定義値	備考
CFG_CMD_Q_NUM	8	コマンドキューの数 (2 の累乗にする必要があります)
CFG_CMD_Q_OFFSET	0x0000	Command queue area のオフセットアドレス
CFG_CYCLIC_REF_AREA_OFFSET	0x0100	Cyclic reference area のオフセットアドレス
CFG_MTR_STATUS_AREA_OFFSET	0x0180	Motor status area のオ フセットアドレス
CFG_MOTOR_CPU_IF_TIMEOUT	100	Open 時の Shared memory 準備完了待ち のタイムアウト回数
CFG_SHARED_MEMORY_INSTANCE_ADDR0	&g_shared_memory0	Shared memory ドライ バ 0 ~ 3 のインスタン スへのポインタ
CFG_SHARED_MEMORY_INSTANCE_ADDR1	&g_shared_memory1	
CFG_SHARED_MEMORY_INSTANCE_ADDR2	&g_shared_memory2	
CFG_SHARED_MEMORY_INSTANCE_ADDR3	&g_shared_memory3	
MOTOR_CPU_IF_CMD_DATA_MAX_BYTE	12	コマンドキューの最大 データサイズ

7.3.6.7 列挙型

表 7-50. Motor CPU I/F 列挙型一覧 (r_motor_cpu_if.h)

列挙型	定義	備考
motor_cpu_if_err_t Motor CPU I/F エラーコード	MOTOR_CPU_IF_SUCCESS	エラー無し
	MOTOR_CPU_IF_ERR_OPEN	Open エラー
	MOTOR_CPU_IF_ERR_CLOSE	Close エラー
	MOTOR_CPU_IF_ERR_PARAM	Parameter エラー
	MOTOR_CPU_IF_ERR_NOT_OPEN	Not open エラー
	MOTOR_CPU_IF_ERR_BUF_FULL	コマンドキューが満杯
	MOTOR_CPU_IF_ERR_BUF_EMPTY	コマンドキューが空
	MOTOR_CPU_IF_ERR_WRITE	書き込みエラー
	MOTOR_CPU_IF_ERR_READ	読み取りエラー
	MOTOR_CPU_IF_ERR_EXCEED_PAYLOAD	データが超過した

列挙型	定義	備考
motor_cpu_if_status_t Motor CPU I/F ステータス	MOTOR_CPU_IF_STATUS_NOT_OPEN	Not open 状態
	MOTOR_CPU_IF_STATUS_OPEN	Open 状態
motor_cpu_if_cmd_t Motor CPU I/F コマンド定義	MOTOR_CPU_IF_CMD_NONE	コマンド無し
	MOTOR_CPU_IF_CMD_RESET	Command Set API で使用され るコマンド
	MOTOR_CPU_IF_CMD_MOTOR_START	
	MOTOR_CPU_IF_CMD_MOTOR_STOP	
	MOTOR_CPU_IF_CMD_MOTOR_RESET	
	MOTOR_CPU_IF_CMD_MOTOR_ERROR_CANCEL	
	MOTOR_CPU_IF_CMD_ERROR_SET	
	MOTOR_CPU_IF_CMD_CTRL_TYPE_SET	
	MOTOR_CPU_IF_CMD_POSITION_COMMAND_MODE_SET	
	MOTOR_CPU_IF_CMD_ROTOR_ANGLE_OFFSET_SET	
	MOTOR_CPU_IF_CMD_CURRENT_OFFSET_SET	
	MOTOR_CPU_IF_CMD_CURRENT_OPEN_LOOP_SET	
	MOTOR_CPU_IF_CMD_ACCEL_TIME_SET	
	MOTOR_CPU_IF_CMD_MAX_SPEED_RAD_SET	
motor_cpu_if_pos_mode_t Motor CPU I/F 位置制御 モード	MOTOR_CPU_IF_POS_MODE_STEP	ステップモード
	MOTOR_CPU_IF_POS_MODE_TRAPEZOID	台形モード
motor_cpu_if_ctrl_type_t Motor CPU I/F モータ制御 タイプ	MOTOR_CPU_IF_CTRL_TYPE_POS	位置制御
	MOTOR_CPU_IF_CTRL_TYPE_SPEED	速度制御
	MOTOR_CPU_IF_CTRL_TYPE_TORQUE	トルク制御
	MOTOR_CPU_IF_CTRL_TYPE_VOLTAGE	オープンループ 制御
	MOTOR_CPU_IF_CTRL_TYPE_CURRENT	
	MOTOR_CPU_IF_CTRL_TYPE_ADJ_CRNT_OFFSET	電流オフセット
	MOTOR_CPU_IF_CTRL_TYPE_ADJ_POS_OFFSET	位置オフセット
motor_cpu_if_motor_err_code_t Motor CPU I/F モータエ ラーコード	MOTOR_CPU_IF_MOTOR_ERROR_NONE	エラー無し
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_CURRENT_HW	HW 過電流エ ラー
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_VOLTAGE	過電圧エラー
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_SPEED	回転速度エラー
	MOTOR_CPU_IF_MOTOR_ERROR_UNDER_VOLTAGE	定電圧エラー
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_CURRENT_SW	SW の過電流エ ラー

motor_cpu_if_motor_state_t Motor CPU I/F モータステータス	MOTOR_CPU_IF_MOTOR_ERROR_OVER_TEMPERATURE	インバータ加熱エラー
	MOTOR_CPU_IF_MOTOR_ERROR_UNKNOWN	不明なエラー
	MOTOR_CPU_IF_STATEMACHINE_STATE_STOP	STOP モード
	MOTOR_CPU_IF_STATEMACHINE_STATE_RUN	RUN モード
	MOTOR_CPU_IF_STATEMACHINE_STATE_ERROR	ERROR モード

7.3.6.8 構造体・変数情報

表 7-51. Motor CPU I/F 構造体定義一覧 (r_motor_cpu_if.h)

構造体	変数	備考
st_current_offset_cfg_t Current offset configuration 構造体	f4_iu_offset	U-phase 電流オフセット
	f4_iv_offset	V-phase 電流オフセット
	f4_iw_offset	W-phase 電流オフセット
st_current_open_loop_cfg_t Open loop current configuration 構造体	f4_open_loop_current	Open loop 電流値
	f4_ref_speed_rpm	指令速度 (RPM)
motor_cpu_if_cmd_hdr_t コマンドキュー head/tail 構造体	head	コマンドキューの head
	tail	コマンドキューの tail
motor_cpu_if_cmd_q_t コマンドキュー 構造体	cmd	コマンド
	size	データサイズ
	data[MOTOR_CPU_IF_CMD_DATA_MAX_BYTE]	データ配列
motor_cpu_if_cyclic_ref_t Cyclic Reference 構造体	f4_ref_position	指令位置
	f4_ref_speed	指令速度
	f4_ref_torque	指令トルク
motor_cpu_if_mtr_status_t Motor Status 構造体	f4_pos_deg	現在位置
	f4_speed	現在速度
	u1_status	ステートマシン
	reserved1	未使用領域
	u2_error_status	エラー状態
	reserved2	未使用領域
	u1_ctrl_loop_mode	LOOP MODE
	reserved3	未使用領域
	u1_in_position	位置制御完了状態
	reserved4	未使用領域
	u1_offset_cal_finished_flag	オフセット測定状態

	reserved5	未使用領域
motor_cpu_if_cfg_t Motor CPU I/F コンフィグ構造体	p_shared_memory_instance	Shared Memory インスタンスへのポインタ
	timeout	Open 時の Shared memory 準備完了待ちのタイムアウト回数
	shr_mem_cmd_data_max_size	コマンドデータ最大サイズ
	shr_mem_cmd_q_num	コマンドキューの数
	shr_mem_cmd_q_mask	コマンドキューマスク
	shr_mem_cmd_q_offset	共有メモリのコマンドキュー領域のオフセット
	shr_mem_cmd_q_size	共有メモリのコマンドキュー領域のサイズ
	shr_mem_cmd_head_offset	共有メモリのコマンドキュー head 領域のオフセット
	shr_mem_cmd_head_size	共有メモリのコマンドキュー head 領域のサイズ
	shr_mem_cmd_tail_offset	共有メモリのコマンドキュー tail 領域のオフセット
	shr_mem_cmd_tail_size	共有メモリのコマンドキュー tail 領域のサイズ
	shr_mem_cyclic_ref_area_offset	共有メモリの Cyclic Reference Area のオフセット
	shr_mem_cyclic_ref_area_size	共有メモリの Cyclic Reference Area のサイズ
	shr_mem_ref_pos_offset	共有メモリの指令位置領域のオフセット
	shr_mem_ref_vel_offset	共有メモリの指令速度領域のオフセット
	shr_mem_ref_tor_offset	共有メモリの指令トルク領域のオフセット
	shr_mem_mtr_status_area_offset	共有メモリの Motor Status Area のオフセット
	shr_mem_mtr_status_area_size	共有メモリの Motor Status Area のサイズ
	shr_mem_mtr_pos_offset	共有メモリの現在位置領域のオフセット
	shr_mem_mtr_vel_offset	共有メモリの現在速度領域のオフセット
	shr_mem_mtr_status_offset	共有メモリのステートマシン領域のオフセット
	shr_mem_mtr_err_status_offset	共有メモリのエラー状態領域のオフセット

	shr_mem_mtr_ctrl_loop_mode_offset	共有メモリの LOOP MODE 領域のオフセット
	shr_mem_mtr_in_pos_offset	共有メモリの位置制御完了状態領域のオフセット
	shr_mem_mtr_cal_finished_flag_offset	共有メモリのオフセット測定状態領域のオフセット
motor_cpu_if_ctrl_t Motor CPU I/F 制御構造体	status	Motor CPU I/F のステータス
	cmd_hdr	コマンドキュー head/tail 構造体インスタンス
	position_offset	位置オフセット
	p_cfg	Motor CPU I/F コンフィグ構造体へのポインタ
motor_cpu_if_api_t Motor CPU I/F API 構造体	Open	オープン関数
	Close	クローズ関数
	Reset	コマンド送信関数
	MotorStart	
	MotorStop	
	MotorReset	
	MotorErrorCancel	
	ErrorSet	
	<u>CtrlTypeSet</u>	
	PositionCommandModeSet	
	RotorAngleOffsetSet	
	CurrentOffsetSet	
	CurrentOpenLoopSet	
	<u>AccelTimeSet</u>	
	PosProfMaxSpeedRadSet	
	PositionOffsetSet	位置オフセット設定関数
	RefPositionSet	参照値セット関数
	RefSpeedSet	
	RefTorqueSet	
	PositionGet	モータステータス取得関数
	SpeedGet	
	StatusGet	
	<u>ErrorStatusGet</u>	
	LoopModeStatusGet	
	InPositionFlagGet	
	OffsetCalibrationStatusGet	

7.3.6.9 グローバル変数情報

表 7-52. Motor CPU I/F グローバル変数一覧 (r_motor_cpu_if.c)

型名	インスタンス名	説明
motor_cpu_if_ctrl_t	g_motor_cpu_if_ctrl0 g_motor_cpu_if_ctrl1 g_motor_cpu_if_ctrl2 g_motor_cpu_if_ctrl3	モータ 4 軸分の Motor CPU I/F 制御構造体インスタンス。 Open 関数で初期化されます。
motor_cpu_if_cfg_t	g_motor_cpu_if_cfg0 g_motor_cpu_if_cfg1 g_motor_cpu_if_cfg2 g_motor_cpu_if_cfg3	モータ 4 軸分の Motor CPU I/F コンフィグ構造体インスタンス。 インスタンス生成時に各メンバが初期化されます。 初期化以外で各メンバの値を変更することはできません。
motor_cpu_if_api_t	g_motor_cpu_if_api	Motor CPU I/F API 構造体インスタンス。 インスタンス生成時に各メンバが初期化されます。 初期化以外で各メンバの値を変更することはできません。

7.3.7 CPU1 CPU 間データ交換制御モジュール

CPU1 CPU 間データ交換制御モジュールは、CiA402 Drive Profile からの要求と指令値をローカル領域に保持し、Shared Memory へのアクセス権がある周期だけ Motor CPU I/F を通じてデータを交換し、完了を INTCPU9 で CPU0 へ通知します。

7.3.7.1 モジュール構成

モジュール構成図を図 7-46 に示します。

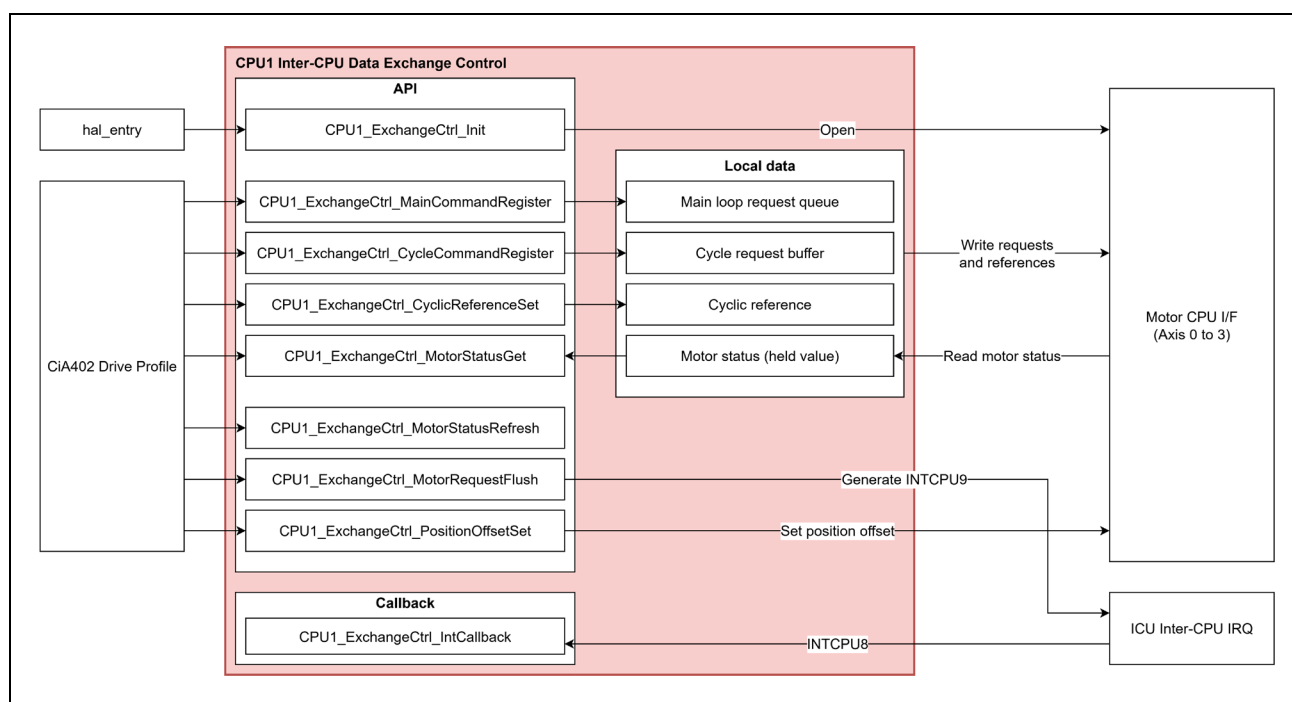


図 7-46. CPU1 CPU 間データ交換制御モジュール構成図

7.3.7.2 CPU 間データ交換シーケンス

CPU0 とのデータ交換シーケンスを図 7-47 に示します。

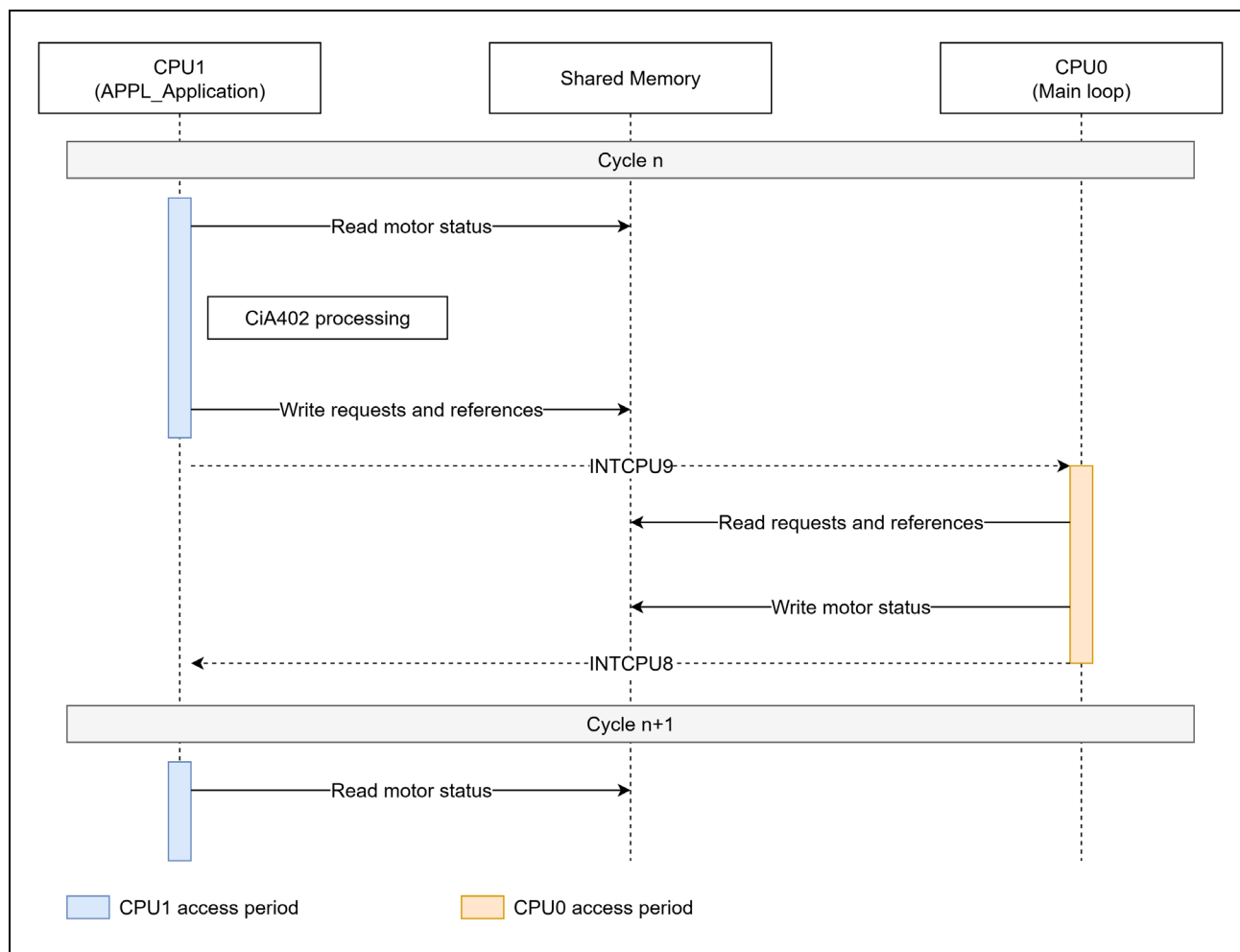


図 7-47. CPU 間データ交換シーケンス

CPU1 は、周期の開始時にアクセス権が無い場合でもアクセス権を待たずに、保持値で周期処理を行い、Shared Memory へのアクセスを次の周期に持ち越します。

7.3.7.3 API

表 7-53. CPU1 CPU 間データ交換制御 API 一覧

API	説明
CPU1_ExchangeCtrl_Init	CPU1 CPU 間データ交換制御モジュールを初期化します。 4 軸分の Motor CPU I/F を Open し、INTCPU8 受信用と INTCPU9 送信用の Inter CPU IRQ を Open します。Shared Memory へのアクセス権の初期値は CPU1 です。
CPU1_ExchangeCtrl_MotorStatusRefresh	APPL_Application 関数の先頭で呼び出します。 アクセス権がある場合は、4 軸分の Motor Status を Motor CPU I/F から取得し、保持値を更新します。アクセス権がない場合は、Shared Memory へアクセスせずに保持値を維持します。
CPU1_ExchangeCtrl_MotorRequestFlush	APPL_Application 関数の末尾で呼び出します。 アクセス権がある場合は、軸ごとにメインループ要求、周期要求、周期指令値の順に Motor CPU I/F へ書き込みます。その後、アクセス権を CPU0 へ渡し、INTCPU9 を発生させて CPU0 へ通知します。
CPU1_ExchangeCtrl_MainCommandRegister	CPU1 メインループ (CiA402 状態遷移) から呼び出します。 コマンドとデータを軸ごとのメインループ要求キューへ登録します。無効な軸番号・データサイズの場合、またはキューが満杯の場合は false を返します。
CPU1_ExchangeCtrl_CycleCommandRegister	CPU1 周期処理から呼び出します。 コマンドとデータを軸ごとの周期要求バッファへ登録します。無効な軸番号・データサイズの場合、または登録数が 4 件を超える場合は false を返します。
CPU1_ExchangeCtrl_CyclicReferenceSet	CPU1 周期処理から呼び出します。 周期指令値 (位置・速度・トルク) を軸ごとの保持領域へ設定します。無効な軸番号の場合は false を返します。
CPU1_ExchangeCtrl_MotorStatusGet	Shared Memory へアクセスせず、軸ごとの Motor Status の保持値をコピーします。未受信・更新中の場合、または無効な軸番号の場合は false を返します。
CPU1_ExchangeCtrl_PositionOffsetSet	Homing mode の処理から呼び出します。 アクセス権がある場合は、対応する R_MOTOR_CPU_IF_PositionOffsetSet を実行します。アクセス権がない場合、無効な軸番号の場合、または実行に失敗した場合は false を返します。

7.3.7.4 Callback 関数

表 7-54. CPU1 CPU 間データ交換制御 Callback 関数一覧 (cpu1_exchange_ctrl.c)

関数名	説明
CPU1_ExchangeCtrl_IntCall back	これは INTCPU8 (r_icu_inter_cpu_irq ch8) の Callback 関数です。 access_allowed を true にします。Shared Memory へはアクセスしません。

7.3.7.5 マクロ定義

表 7-55. CPU1 CPU 間データ交換制御 マクロ定義一覧 (cpu1_exchange_ctrl.h)

マクロ名	定義値	備考
CPU1_EXCHANGE_CTRL_AXIS_NUM	4	管理対象の軸数
CPU1_EXCHANGE_CTRL_MAIN_REQ_SLOT_NUM	4	1 軸当たりのメイン ループ要求キューのス ロット数 (最大 3 件を 保持)
CPU1_EXCHANGE_CTRL_MAIN_REQ_MAX	3	1 軸当たりのメイン ループ要求の最大件数
CPU1_EXCHANGE_CTRL_CYCLE_REQ_MAX	4	1 軸当たりの周期要求 の最大件数

7.3.7.6 列挙型

表 7-56. CPU1 CPU 間データ交換制御 列挙型一覧 (cpu1_exchange_ctrl.h)

列挙型	定義	備考
cpu1_exchange_ctrl_ref _type_t 周期指令種別	CPU1_EXCHANGE_CTRL_REF_POSITION	位置指令
	CPU1_EXCHANGE_CTRL_REF_SPEED	速度指令
	CPU1_EXCHANGE_CTRL_REF_TORQUE	トルク 指令

7.3.7.7 構造体・変数情報

表 7-57. CPU1 CPU 間データ交換制御 構造体定義一覧 (cpu1_exchange_ctrl.h)

構造体	変数	備考
cpu1_exchange_ctrl_main_r eq_queue_t メインループ要求キュー構 造体	motor_cpu_if_cmd_q[4]	メインループ要求のリングバッ ファ
	head	次の書込み位置 (メインループ が更新)
	tail	次の読出し位置 (周期処理が更 新)

cpu1_exchange_ctrl_cycle_req_buffer_t 周期要求バッファ構造体	motor_cpu_if_cmd_q[4]	当該周期で実行するコマンド要求
	count	登録済み件数（周期処理の終了時に 0 へ戻す）
cpu1_exchange_ctrl_cyclic_ref_t 周期指令値構造体	type	周期指令種別
	value	周期指令値
	write_pending	周期指令値が書き込み待ちであることを示すフラグ
cpu1_exchange_ctrl_status_t Motor Status 保持値構造体	sequence	更新シーケンス番号（0：未受信、奇数：更新中、0 以外の偶数：更新完了）
	status_value	Motor Status の保持値
cpu1_exchange_ctrl_cfg_t コンフィグ構造体	p_motor_cpu_if_ctrl[4]	軸 0～3 の Motor CPU I/F 制御構造体へのポインタ
	p_motor_cpu_if_cfg[4]	軸 0～3 の Motor CPU I/F コンフィグ構造体へのポインタ
cpu1_exchange_ctrl_t 制御構造体	access_allowed	CPU1 が Shared Memory へアクセスできることを示すフラグ
	main_req_queue[4]	軸ごとのメインループ要求キュー
	cycle_req_buffer[4]	軸ごとの周期要求バッファ
	cyclic_ref[4]	軸ごとの周期指令値
	motor_status[4]	軸ごとの Motor Status の保持値
	p_cfg	コンフィグ構造体へのポインタ

7.3.7.8 グローバル変数情報

表 7-58. CPU1 CPU 間データ交換制御 グローバル変数一覧（cpu1_exchange_ctrl.c）

型名	インスタンス名	説明
cpu1_exchange_ctrl_t	g_cpu1_exchange_ctrl	CPU1 CPU 間データ交換制御の制御構造体インスタンス。 ゼロ初期化され、Init 関数で p_cfg の設定と access_allowed の初期化（true）が行われます。
cpu1_exchange_ctrl_cfg_t	g_cpu1_exchange_ctrl_cfg	CPU1 CPU 間データ交換制御のコンフィグ構造体インスタンス。 4 軸分の Motor CPU I/F 制御構造体・コンフィグ構造体インスタンスで初期化されます。

8. Appendix

8.1 統合開発環境インストール

8.1.1 e² studio, FSP

下記から e2 studio と同梱のインストーラ（FSP Platform Installer）をダウンロードしてインストールしてください。

<https://github.com/renesas/rz-fsp/releases>

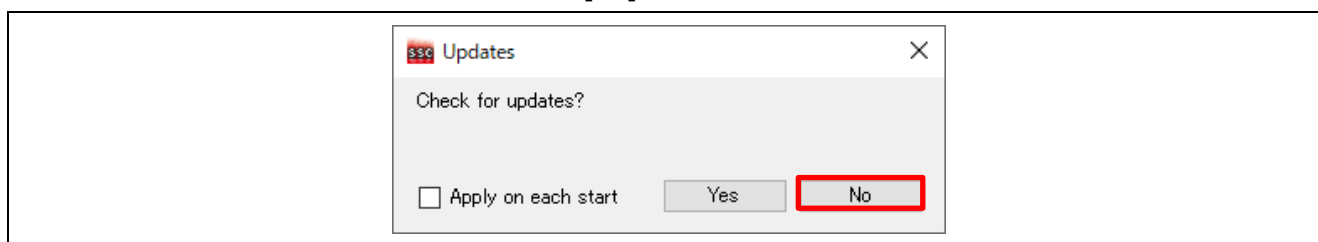
8.2 付録：SSC Tool 使用時の注意点

※ SSC Tool を起動する際の注意点

1. SSC Tool は必ず「管理者権限」で起動してください。

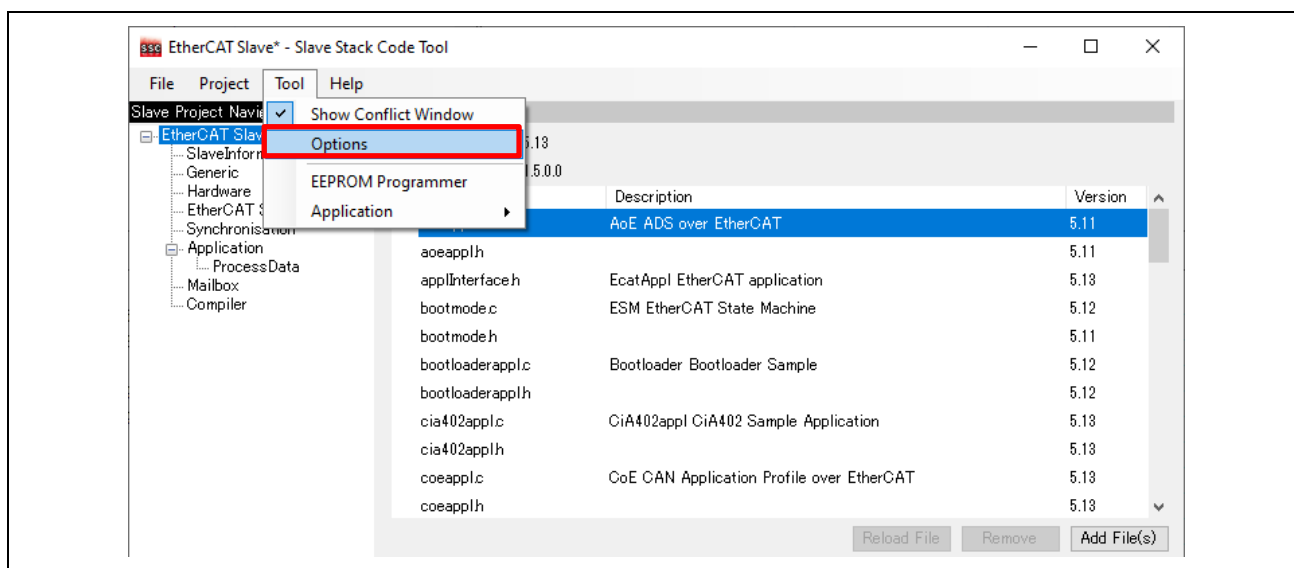
管理者権限で起動しない場合、SSC コードの生成に失敗する可能性があります。

2. 初めて SSC Tool を起動する際、以下のウィンドウが表示される場合があります。
その場合は、更新の確認を行わないように **[No]** を選択してください。



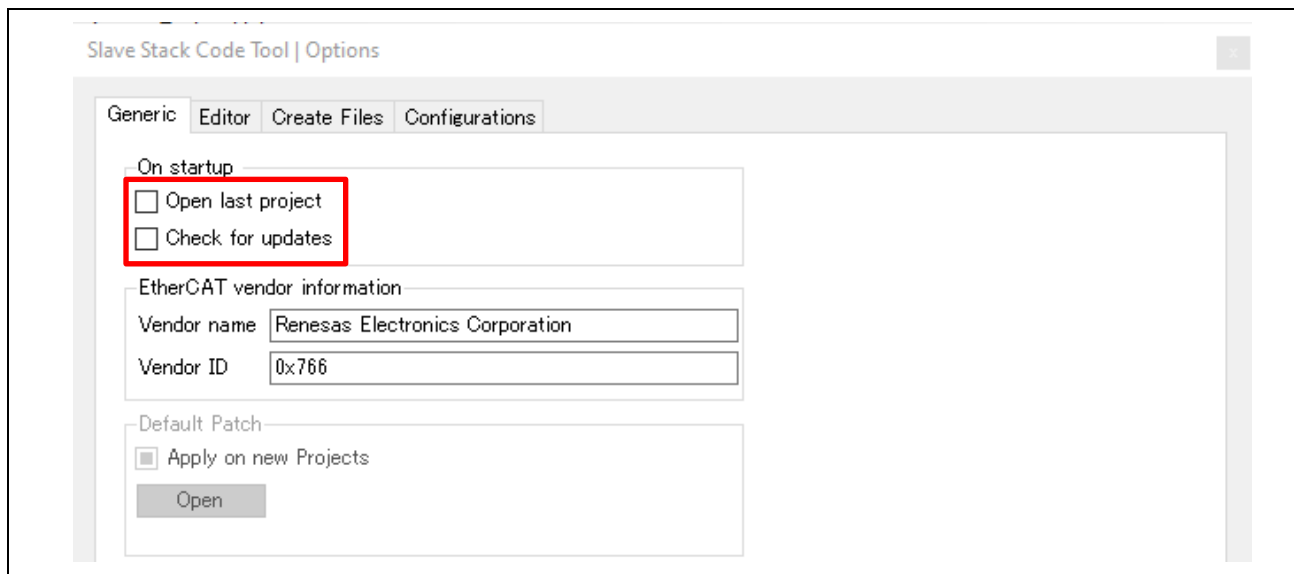
※ SSC Tool の設定について

3. メニューから **[Tool]** → **[Options]** を開き、SSC Tool の設定を確認してください。

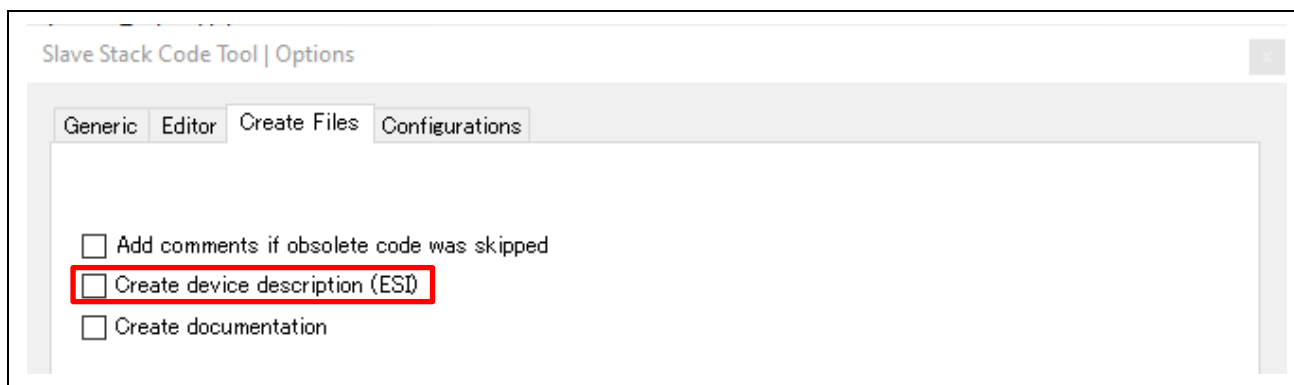


4. **[Generic]** タブ

- **[Open last project]** のチェックを外してください。
- **[Check for updates]** のチェックを外してください。

5. **[Create Files]** タブ

- **[Create device description (ESI)]** のチェックを外してください。

6. 設定を変更したら、**[OK]** をクリックして適用します。

これで、本サンプルプログラム用の **SSC Tool** のセットアップは完了です。

8.3 付録 : patch のインストール方法

patch をインストールする方法は、以下の 2 通りです。

1. Git for Windows (64bit) を使用する方法
2. MinGW Installation Manager を使用する方法

8.3.1 Git for Windows (64bit) を使用する方法

1. Git for Windows の公式サイトからインストーラ（例 : `Git-x.xx.x-64-bit.exe`）をダウンロードします。

[Git for Windows](#)

2. ダウンロードしたインストーラを実行し、セットアップ手順に従ってインストールします。特別な要件がない場合は、デフォルト設定を使用してください。

3. インストール後、`patch.exe` のパスをシステム環境変数に追加します。

- デフォルトのインストールパスは下記の通りです：
`C:¥Users¥<ユーザー名>¥AppData¥Local¥Programs¥Git¥usr¥bin`

環境変数を更新した後、PC を再起動して変更を適用します。

4. コマンドプロンプトを起動し、`where patch` と入力します。 `patch.exe` のパスが表示されれば、インストール成功です。

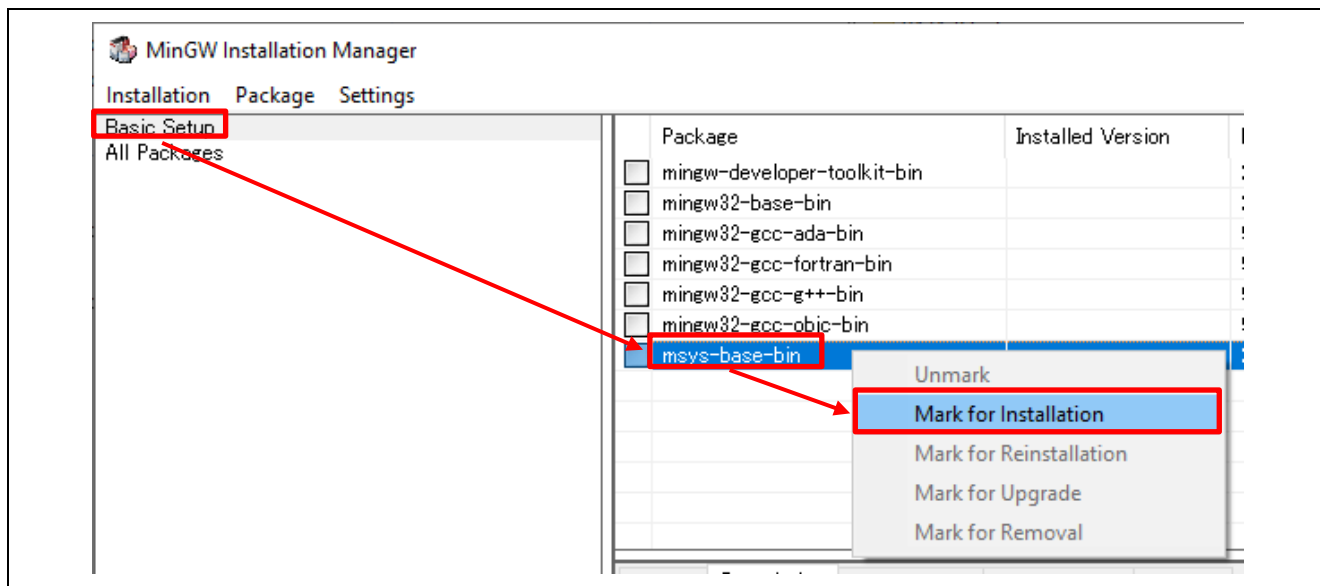
```
C:¥>where patch
C:¥Users¥          ¥AppData¥Local¥Programs¥Git¥usr¥bin¥patch.exe
```

8.3.2 MinGW Installation Manager を使用する方法

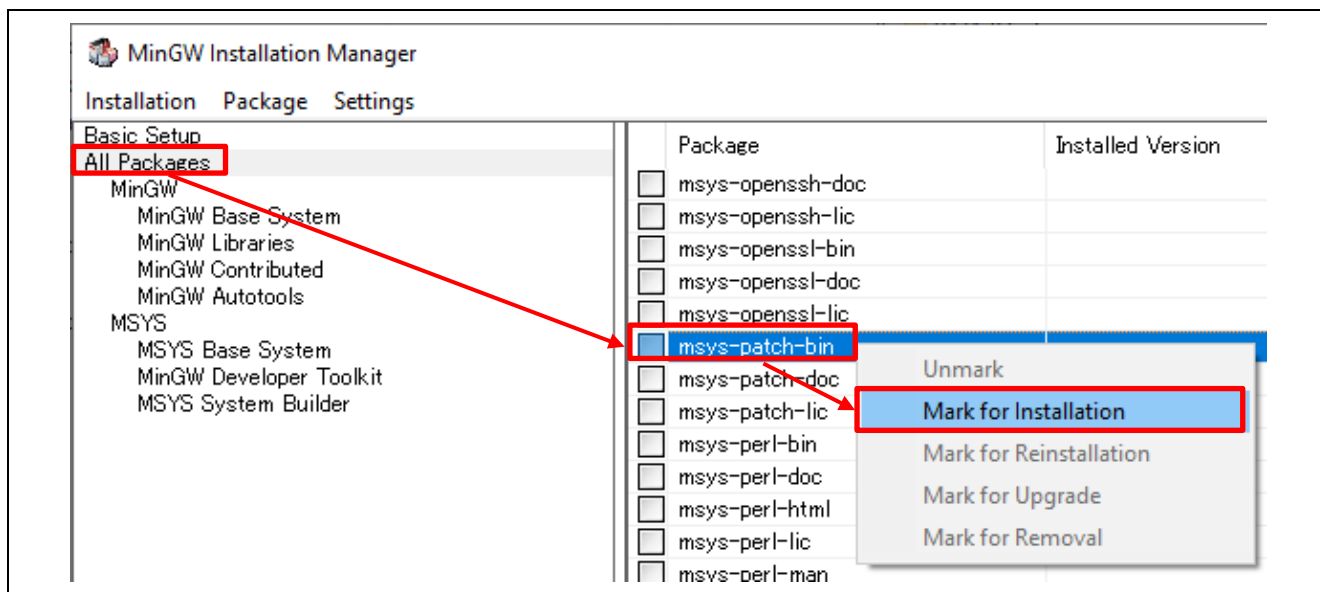
1. 以下の URL から mingw-get-setup.exe をダウンロードします：

<https://sourceforge.net/projects/mingw/>

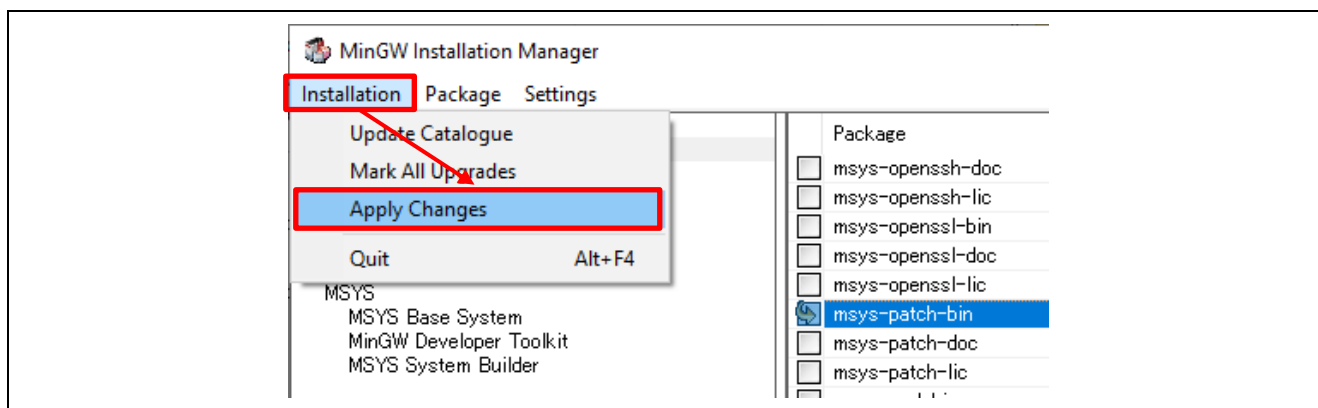
2. mingw-get-setup.exe を実行し、ダイアログに従って「Mingw-installation-manager」をインストールします。
3. インストール完了後、Mingw-installation-manager ウィンドウが表示されたら、左画面で **Basic Setup** を選択し、右画面で **msys-base-bin** を右クリックして「Mark for Installation」を選択します。



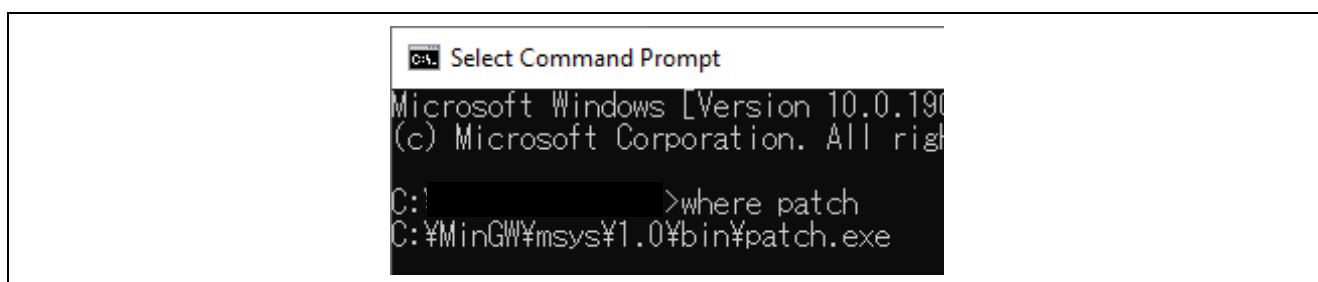
4. 次に、左画面で **All Packages** を選択し、右画面で **msys-patch-bin** を右クリックして「Mark for Installation」を選択します。



5. メニューバーの「Installation」から「Apply Changes」を選択します。



6. 「Schedule of Pending Actions」ウィンドウが表示されたら、「Apply」ボタンをクリックします。
7. 「All changes were applied successfully; you may now close this dialogue.」と表示されれば、patch.exe のインストールは成功です。
8. インストールした patch.exe のパスをシステム環境変数に追加します。
- デフォルトパスの例：
C:¥MinGW¥msys¥1.0¥bin
- 環境変数を更新した後、PC を再起動して変更を適用します。
9. コマンドプロンプトを起動し、where patch と入力します。
インストールした patch.exe のパスが表示されれば問題ありません。



改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2026.9.30	-	新規作成

製品ご使用上の注意事項

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

1. 静電気対策

CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。

2. 電源投入時の処置

電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。

3. 電源オフ時における入力信号

当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。

4. 未使用端子の処理

未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。

5. クロックについて

リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。

6. 入力端子の印加波形

入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、 V_{IL} (Max.) から V_{IH} (Min.) までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、 V_{IL} (Max.) から V_{IH} (Min.) までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。

7. リザーブアドレス（予約領域）のアクセス禁止

リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。

8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻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。

OArm および Cortex は、Arm Limited（またはその子会社）の EU またはその他の国における登録商標です。 All rights reserved.

OIEEE は、the Institute of Electrical and Electronics Engineers, Inc. の登録商標です。

OEtherCAT® および TwinCAT ®は、ドイツ Beckhoff Automation GmbH によりライセンスされた特許取得済み技術であり登録商標です。

OFA-CODER は、多摩川精機株式会社の商標です

Oその他、本資料中の製品名やサービス名は全てそれぞれの所有者に属する商標または登録商標です。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。

標準水準： コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等

高品質水準： 輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等

当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。

7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限られません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因したまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものとしします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

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

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

(Rev.5.0-1 2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

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

www.renesas.com/contact/

商標について

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