

RZ/T2N Group

4-axis Motor Control Sample Program Startup Manual(EtherCAT CiA402)

Introduction

This application note is a quick start manual for 4-axis Motor Control Sample Program using EtherCAT CiA402 on RZ/T2N Evaluation Board equipped with MPU of RZ/T2N group by Renesas Electronics Corporation.

Target Device

RZ/T2N Group

When applying the sample program covered in this application note to another microcomputer, modify the program according to the specifications for the target microcomputer and conduct an extensive evaluation of the modified program.

Contents

1. Overview	5
1.1 4-axis Motor Control Sample Program	5
1.2 Hardware configuration	5
1.3 Connection Equipment	6
1.3.1 RZ/T2N Evaluation Board and Inverter Board Connection	7
1.3.2 Motor/ Encoder and Inverter Board Connection	8
1.3.3 PC Connection	8
1.3.4 ICE	8
1.3.5 Power Supply	8
2. Operating Environment	9
2.1 Operation Environment	9
2.2 Folder and File Configuration	10
3. Related Application Notes	11
4. Board Setting	12
4.1 RZ/T2N EVB Setting	12
4.1.1 Switches	12
4.1.2 Jumpers	13
4.2 Inverter Board	13
4.2.1 Jumpers	13
5. How to Write Program	14
5.1 Generating the EtherCAT SubDevice Stack	14
5.2 Building the Project	16
5.3 Writing the Project	19
6. How to Operate Program	20
6.1 Connecting to TwinCAT	20
6.1.1 Preparing the ESI File (First Time Only)	20
6.1.2 Installing the TwinCAT Driver (First Time Only)	20
6.1.3 Scanning Devices and Writing to EEPROM	21
6.2 How to Run the TwinCAT Project	26
6.3 How to Verify Operation	33
6.3.1 Selecting CiA402 Operating Mode	33
6.3.2 Controlling CiA402 State Machine	34
6.3.2.1 How to Transition to Operation Enable	34
6.3.3 How to Recover from Fault	36
6.3.4 Motor Operation Methods	37
6.3.4.1 CSP Mode (Cyclic Synchronous Position)	37

6.3.4.2	PP Mode (Profile Position)	38
6.3.4.3	CSV Mode (Cyclic Synchronous Velocity)	40
6.3.4.4	HM Mode (Homing)	41
7.	Software Specification	44
7.1	Software Configuration	44
7.2	Motor Control (RZ/T2N Cortex-R52 CPU0)	45
7.2.1	Overall Software Structure	45
7.2.2	Changes from the Base Project	46
7.2.3	Network CPU I/F Module	50
7.2.3.1	Module Structure	50
7.2.3.2	Shared Memory Allocation	51
7.2.3.3	API	52
7.2.3.4	Private Functions	58
7.2.3.5	Callback Function	60
7.2.3.6	Macro Definitions	61
7.2.3.7	Enumerations	62
7.2.3.8	Structure and Variable Information	63
7.2.3.9	Global Variable Information	65
7.2.4	CPU0 Inter-CPU Data Exchange Control Module	66
7.2.4.1	Module Structure	66
7.2.4.2	API	66
7.2.4.3	Callback Function	66
7.2.4.4	Structure and Variable Information	67
7.2.4.5	Global Variable Information	67
7.3	EtherCAT (CiA402) (RZ/T2N Cortex-R52 CPU1)	68
7.3.1	Basic Specifications	68
7.3.2	Overall Software Structure	69
7.3.3	src Folder Structure	70
7.3.4	hal_entry and Main Processing	71
7.3.5	CiA402 Drive Profile	74
7.3.5.1	CiA402 Operating Modes	74
7.3.5.2	Cyclic Synchronous Position Mode (CSP)	75
7.3.5.3	Cyclic synchronous velocity mode (CSV)	76
7.3.5.4	Profile position mode (PP)	77
7.3.5.5	Homing Mode (HM)	79
7.3.5.6	CiA402 State Transitions	81
7.3.5.7	CiA402 Error Code	82
7.3.5.8	Functions for CiA402 Drive Profile	83
7.3.5.9	Macro Definitions	96
7.3.5.10	Changes After SSC Tool Generated Files	96

7.3.5.11 Object Dictionary	99
7.3.6 Motor CPU I/F	103
7.3.6.1 Module Structure	103
7.3.6.2 Shared Memory Allocation	103
7.3.6.3 API	104
7.3.6.4 Private Functions	109
7.3.6.5 Callback Function	111
7.3.6.6 Macro Definitions	112
7.3.6.7 Enumerations	112
7.3.6.8 Structures & Variable Information	114
7.3.6.9 Global Variables	117
7.3.7 CPU1 Inter-CPU Data Exchange Control Module	118
7.3.7.1 Module Structure	118
7.3.7.2 Inter-CPU Data Exchange Sequence	119
7.3.7.3 API	120
7.3.7.4 Callback Function	121
7.3.7.5 Macro Definitions	121
7.3.7.6 Enumerations	121
7.3.7.7 Structure and Variable Information	121
7.3.7.8 Global Variable Information	122
8. Appendix	123
8.1 Integrated development environment Installation	123
8.1.1 e ² studio, FSP	123
8.2 Appendix : A point of caution when using SSC Tool	124
8.3 Appendix : How to install patch	126
8.3.1 Via Git for Windows (64bit)	126
8.3.2 Via MinGW Installation Manager	127
Revision History	129

1. Overview

1.1 4-axis Motor Control Sample Program

4-axis Motor Control Sample Program can operate maximum 4-axis motor by connecting RZ/T2N Evaluation Board and Renesas' RZ/T series Inverter Board Kit (Inverter Board + BLDC).

The software features are as follows:

- Vector control of permanent magnet synchronous motors using a serial encoder
- EtherCAT SubDevice communication
- Motor control by CiA402 profile

For the motor control by Renesas Motor Workbench (RMW), refer to the "RZ/T2N Encoder Vector Control of Permanent Magnet Synchronous Motor - Serial Encoder (R01AN8225)".

1.2 Hardware configuration

Figure 1.1 is shown the hardware configuration.

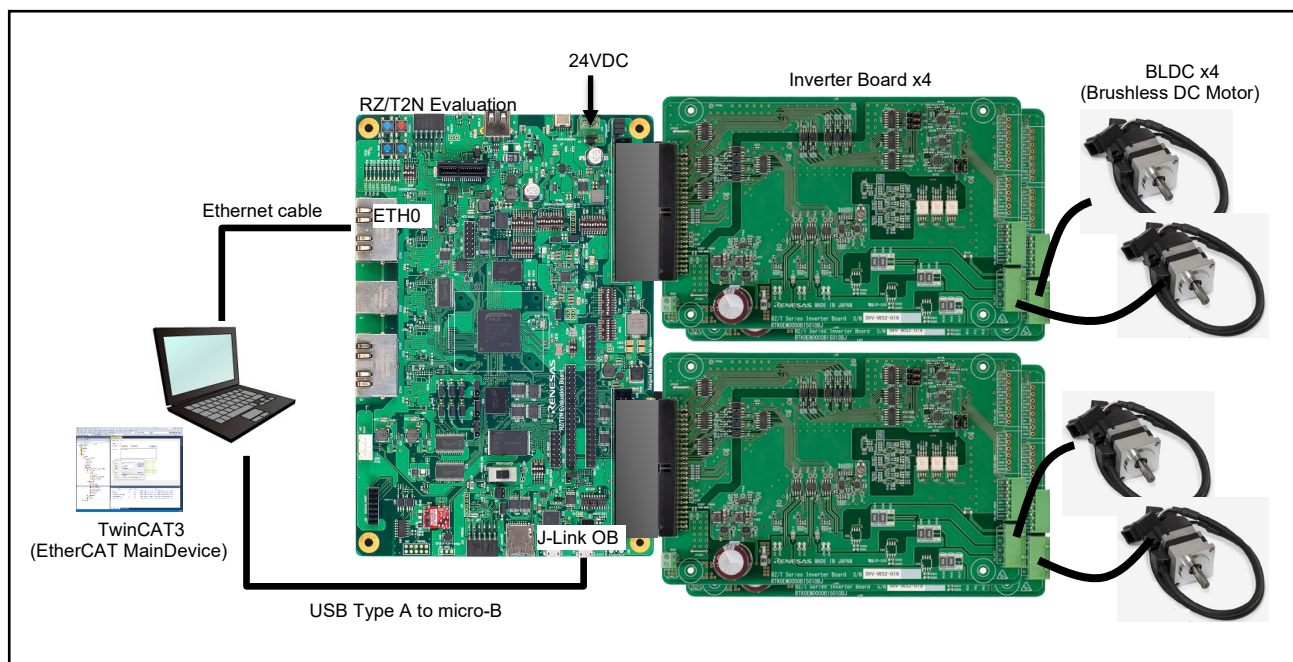


Figure 1.1 Hardware configuration

1.3 Connection Equipment

Table 1.1 is shown the connection equipment list.

Table 1.1 connection equipment list

Item	Model	Remarks
Evaluation Board Kit	RTK9RZT2N1S00010BJ	https://www.renesas.com/rzt2n-evk
RZ/T Series Inverter Board Kit	RTK0EM0000S05010BJ (Renesas) - Board P/N: RTK0EM0000B15010BJ - Motor P/N: TSM3101N2001E020 (Tamagawa-Seiki)	Available from URL below. https://www.renesas.com/invb-lv-rzt-b
Flat cable	MIL Socket Cable (50pin, cable length: 10cm)	Required if use INV3/INV4
Board spacer	(1) Height: 10mm (2) Height: 7mm (3) Height: 30mm or more	(1) Required if use INV1/INV2 (2) Required if use INV1/INV2 (3) Required if use INV3/INV4
Power supply	24V/8V(Max) DC, 5.08-mm-pitch 2-pin terminal block	Use stabilized power supply. (Current value at 4-axis/1000RPM operation is 1.1A)
ICE	SEGGER J-Link OB	Use USB type-A to micro-B cable

1.3.1 RZ/T2N Evaluation Board and Inverter Board Connection

RZ/T2N Evaluation Board and Inverter Board (INV1 and INV2) connect directly. The heights of RZ/T2N Evaluation Board and Inverter Board need to be aligned. When using 10mm rubber feet (1) with RZ/T2N Evaluation Board, use 7mm spacers (2) for Inverter Board.

INV3 and INV4 are connected using the 10 cm flat cable. Also, INV3 and INV4 are attached to INV1 and INV2 using spacers (3) of 30mm or more.

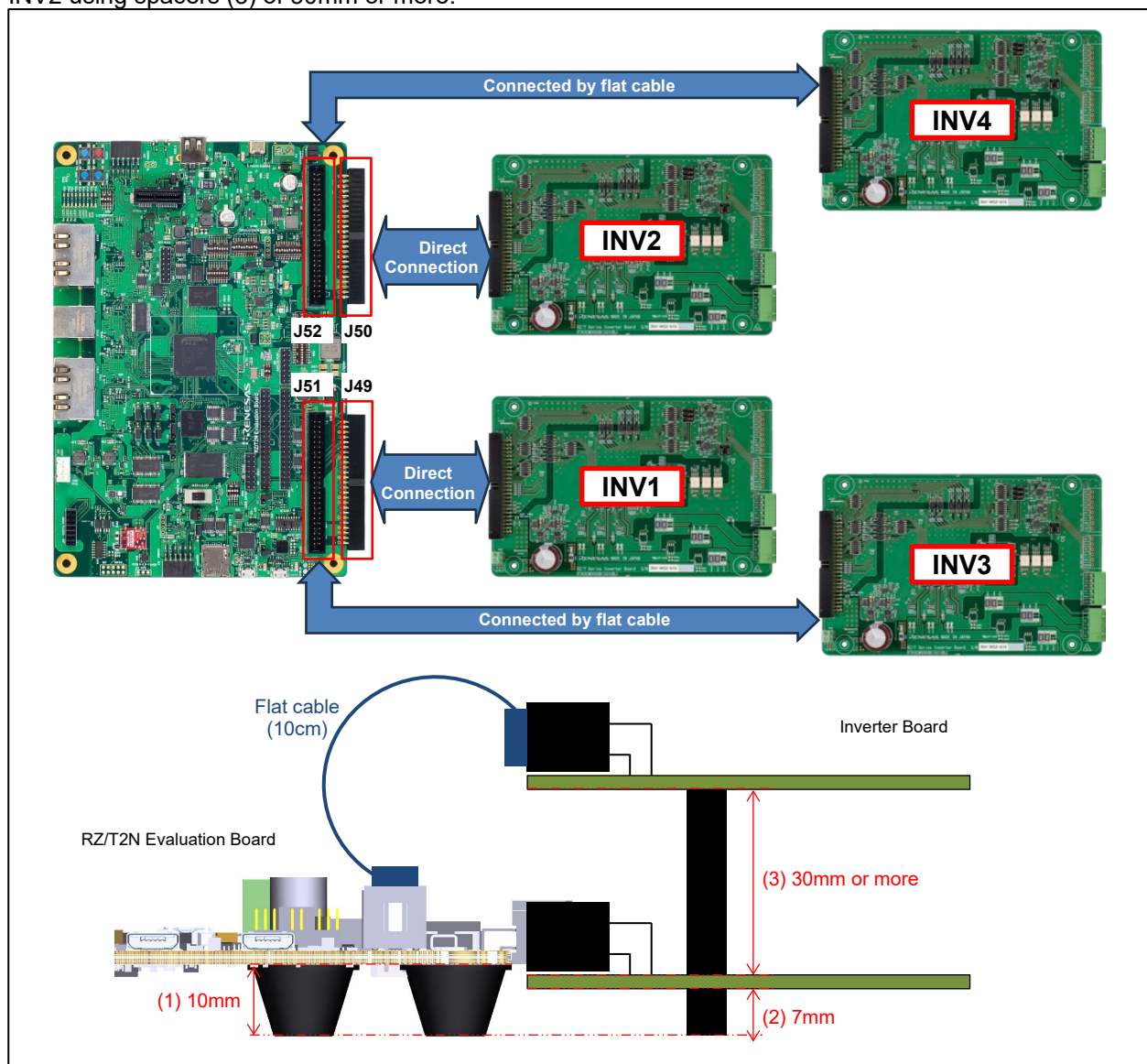


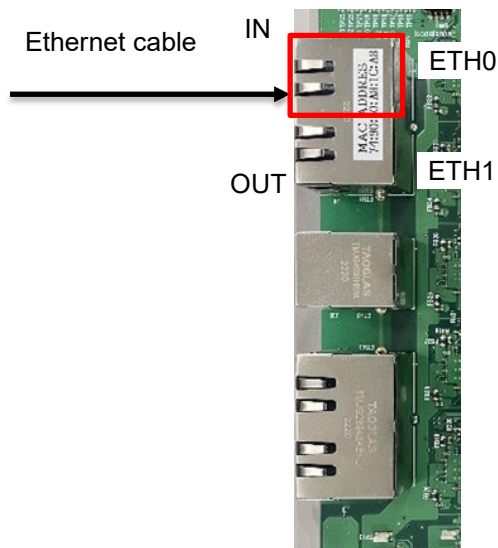
Figure 1.2 RZ/T2N Evaluation Board and Inverter Board connection

1.3.2 Motor/ Encoder and Inverter Board Connection

For Tamagawa Seiki: TSM3101N2001E020 (Motor/ Encoder (FA-CODER)) and Inverter Board Connection, please see “RZ/T Series Inverter Board/Kit User’s Manual (R12UZ0155)”.

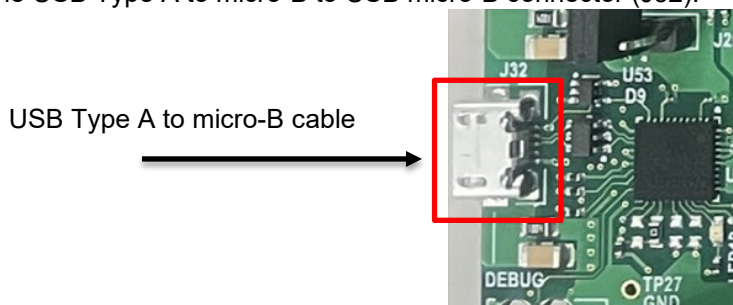
1.3.3 PC Connection

Connect the Ethernet cable to PC and the ETH0 of the RJ45 connector on RZ/T2N Evaluation board.



1.3.4 ICE

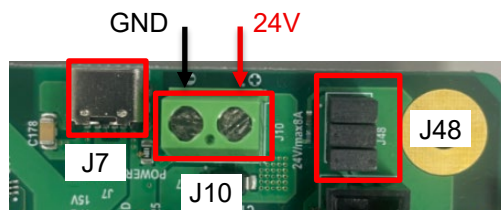
Connect the USB Type A to micro-B to USB micro-B connector (J32).



1.3.5 Power Supply

By applying 24V DC to the terminal block (J10) of the RZ/T2N Evaluation Board, it is possible to supply power to four inverter boards. In this case, do not apply 24V DC to the Inverter Board.

The maximum current that can be supplied from J10 is 8A. If the total current consumed by the RZ/T2N Evaluation Board and the Inverter Boards exceeds 8A, supply 15V power using a USB PD-compatible USB Type-C cable via the USB Type-C connector (J7), and supply 24V power directly to the Inverter Boards. In this case, set the J48 jumper to Open.



2. Operating Environment

2.1 Operation Environment

Table 2.1 Operating environment

Item	Content
Board	RZ/T2N Evaluation Board
MPU	RZ/T2N Group R9A07G076M48GBG: 537pinFBGA
Operating frequency	CPU Core0 : 800MHz (Arm® Cortex®-R52) CPU Core1 : 800MHz (Arm® Cortex®-R52)
Operating voltage	3.3V/1.8V/1.1V
Operating mode	xSPI0 boot mode(x1 boot serial flash)
Device	Serial Flash ROM (64Mbyte) Macronix MX25LW51245GXDI00-T I2C EEPROM (32Kbit) Microchip 24CW320T-I
Communication protocol	EtherCAT® 
Integrated development environment	RENESAS e² studio 2026-07 (26.07.0)
ICE	SEGGER 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 Folder and File Configuration

```
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. Related Application Notes

The application notes related to this application note are listed below for reference.

- RTK9RZT2N1CW1010BJ User's Manual (R20UT5574)
- RZ/T Series Inverter Board/Kit User's Manual (R12UZ0155)
- RZ/T2N Encoder Vector Control of Permanent Magnet Synchronous Motor - Serial Encoder (R01AN8225)

- CiA402 Documents:

- IEC 61800-7-201 Edition 1.0

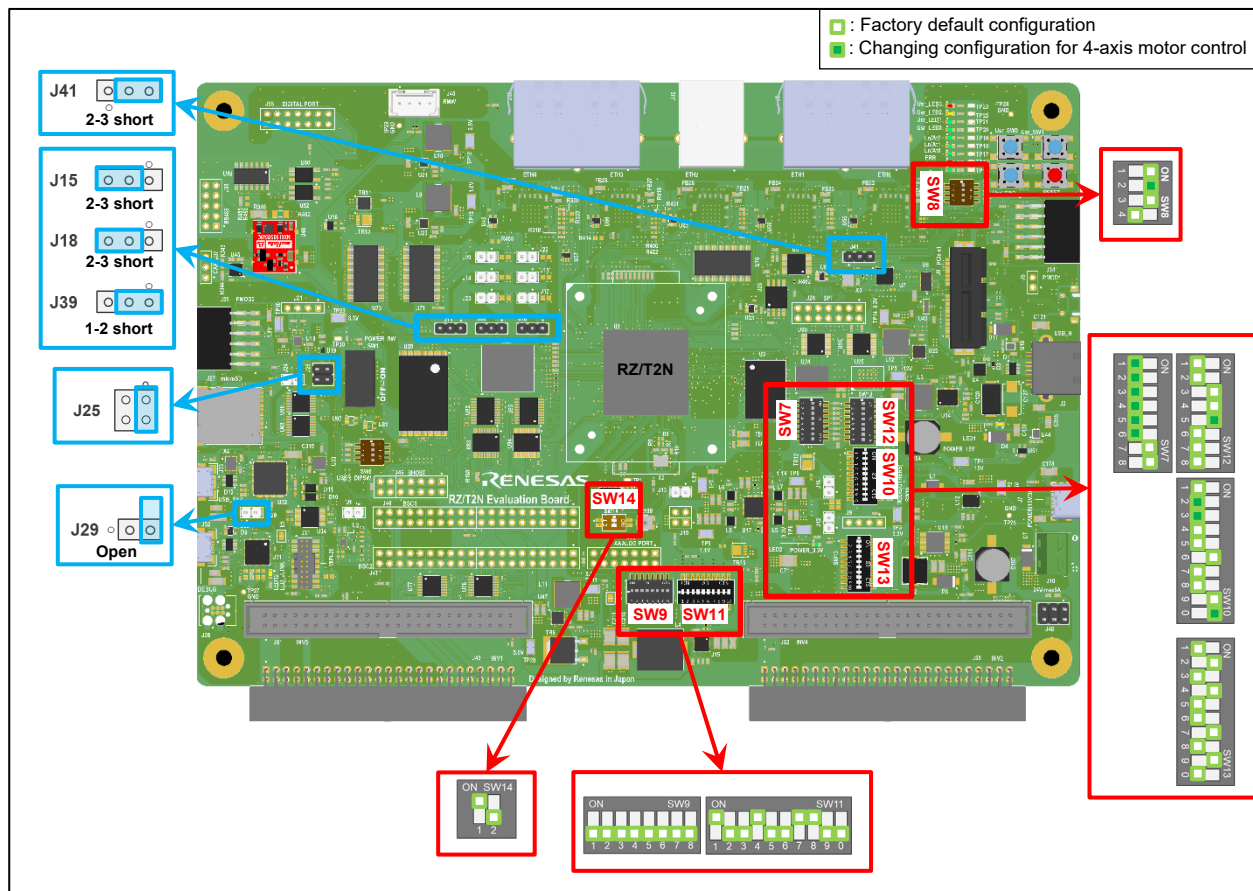
Adjustable speed electrical power drive systems Part 7-201: Generic interface and use of profiles for power drive systems Profile type 1 specification

- 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. Board Setting

4.1 RZ/T2N EVB Setting



4.1.1 Switches

For switch details of RZ/T2N Evaluation Board, refer to the “RTK9RZT2N1CW1010BJ User’s Manual (R20UT5574)”. The text in red in the tables below indicates settings that will be changed from the factory default configuration.

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

For jumper details of RZ/T2N Evaluation Board, refer to the “RTK9RZT2N1CW1010BJ User’s Manual (R20UT5574)”. The text in red in the tables below indicates settings that will be changed from the factory default configuration.

No	JP	Setting
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

For Jumper details of Inverter Board, refer to the “RZ/T Series Inverter Board/Kit User’s Manual (R12UZ0155)”.

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

5. How to Write Program

5.1 Generating the EtherCAT SubDevice Stack

Note:

This sample project does not include the EtherCAT SubDevice stack code.

To generate EtherCAT slave stack code, you need the EtherCAT SubDevice Stack Code (SSC) Tool.

The SSC Tool can be obtained from the ETG (EtherCAT Technology Group).

Before launching the SSC Tool, please check “Appendix : A point of caution when using SSC Tool”.

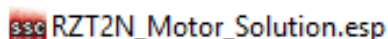
To execute this guide, patch.exe must be installed on your PC.

Refer to “Appendix : How to install patch” and install patch.exe.

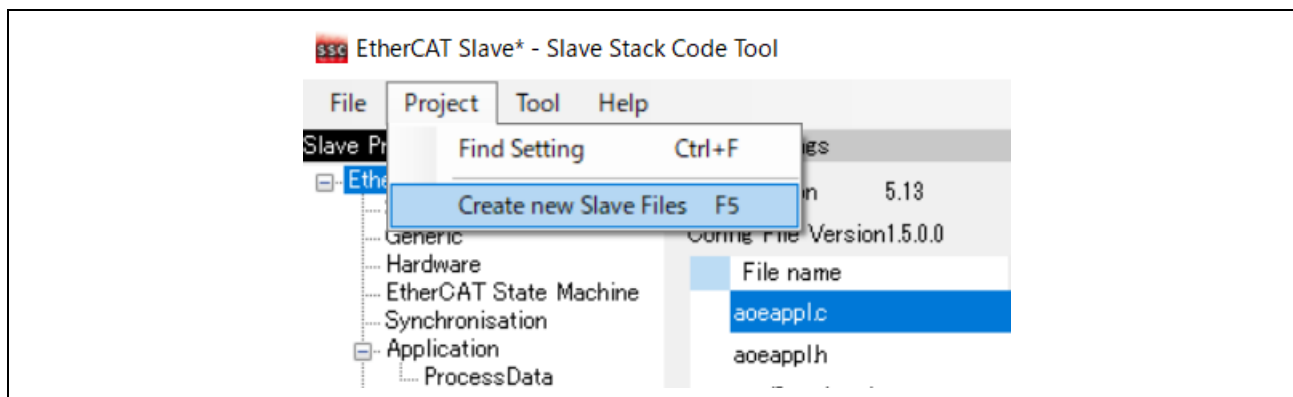
1. Extract r01an8308xx0100-rzt2n-4axis-motor-ecat-package.zip so that the folder structure looks like this:

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

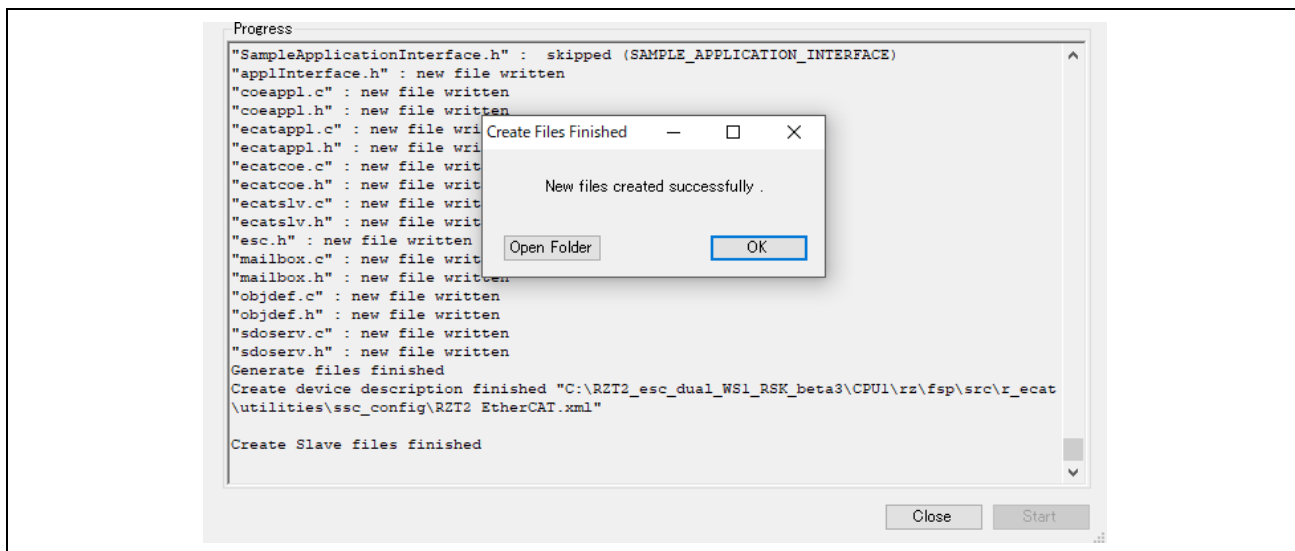
2. Double-click the .esp file located in the “Common\SSCConfig” folder to launch the SSC Tool.

The image shows a file icon for 'RZT2N_Motor_Solution.esp'. The icon is a red square with the letters 'SSC' in white, followed by the text 'RZT2N_Motor_Solution.esp' in blue.

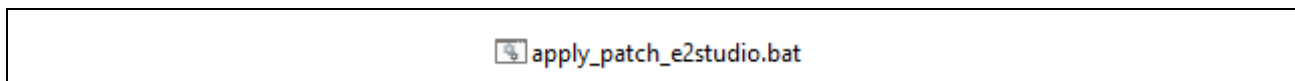
3. Click [Project] → [Create New Slave Files]. When the dialog opens, click [Start].



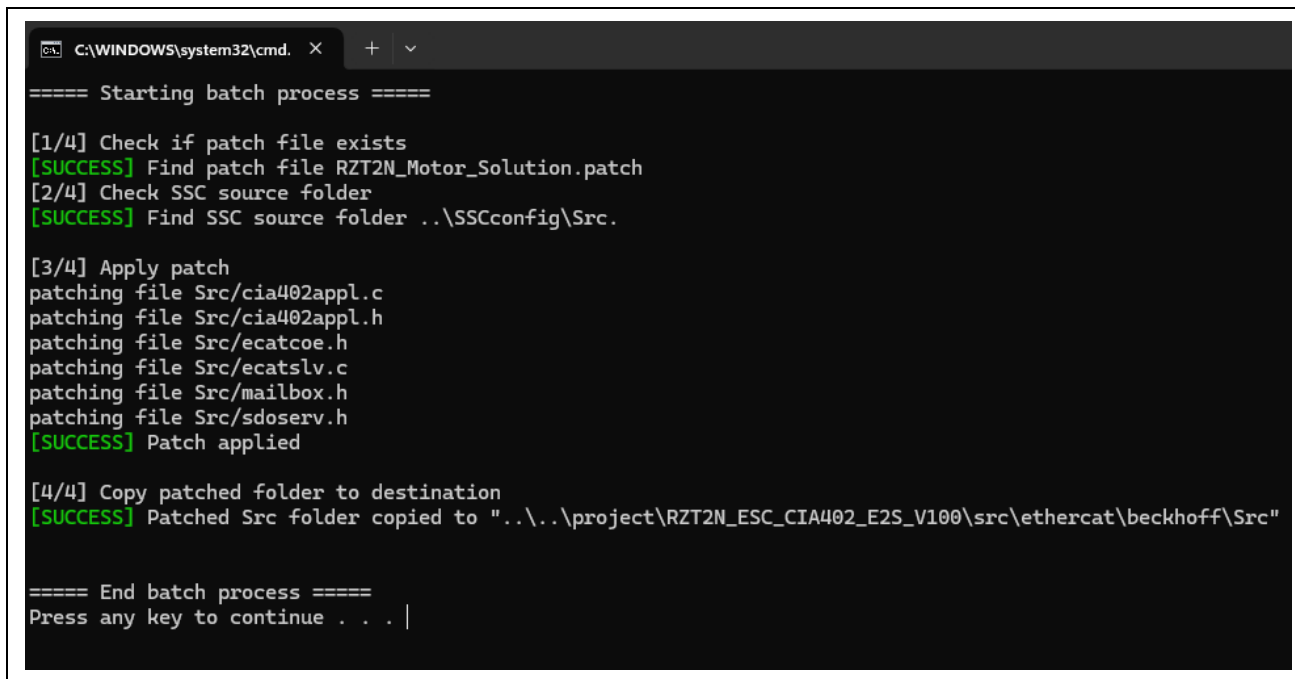
4. When the source code is successfully generated, the message “New Files created successfully” will appear. Click [OK].



5. Double-click [apply_patch_e2studio.bat] located in the “Common\Patch” folder to execute the batch file.

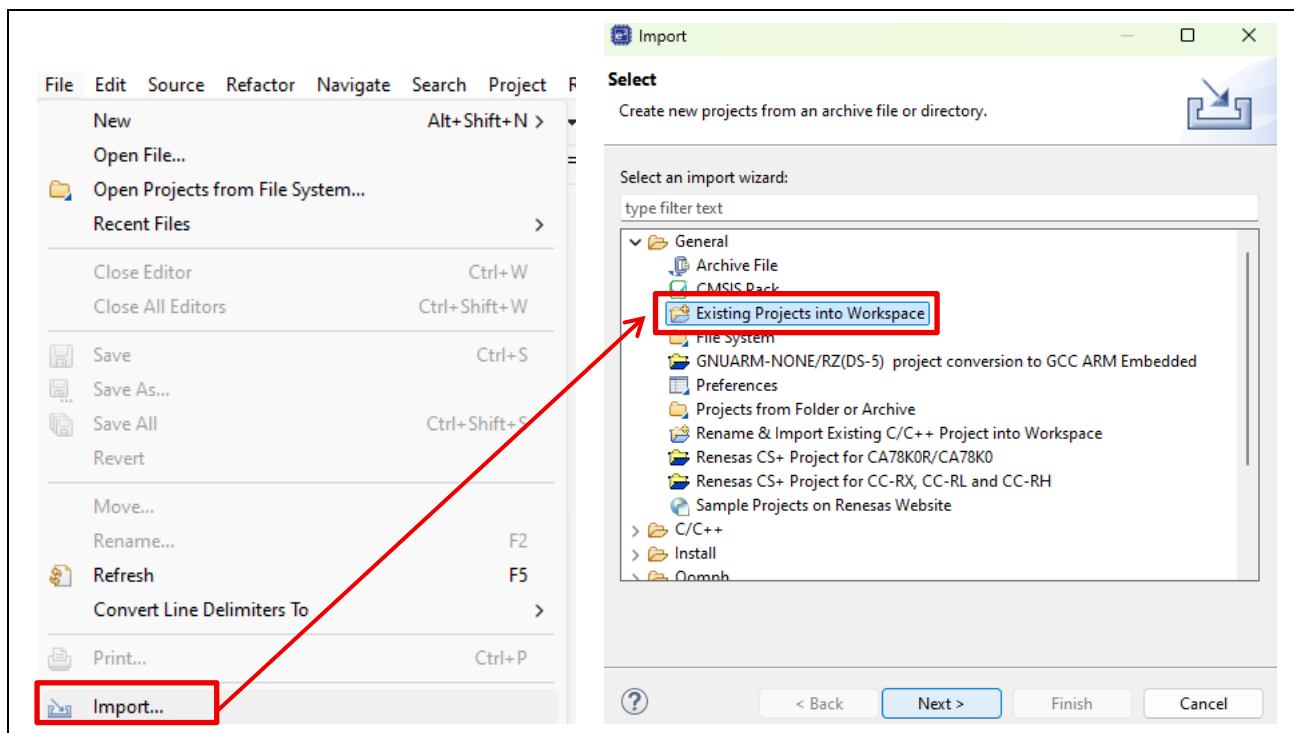


6. After executing the batch file, the Src folder will be moved to:
project\RZT2N_ESC_CIA402_E2S_V100\src\ethercat\beckhoff.

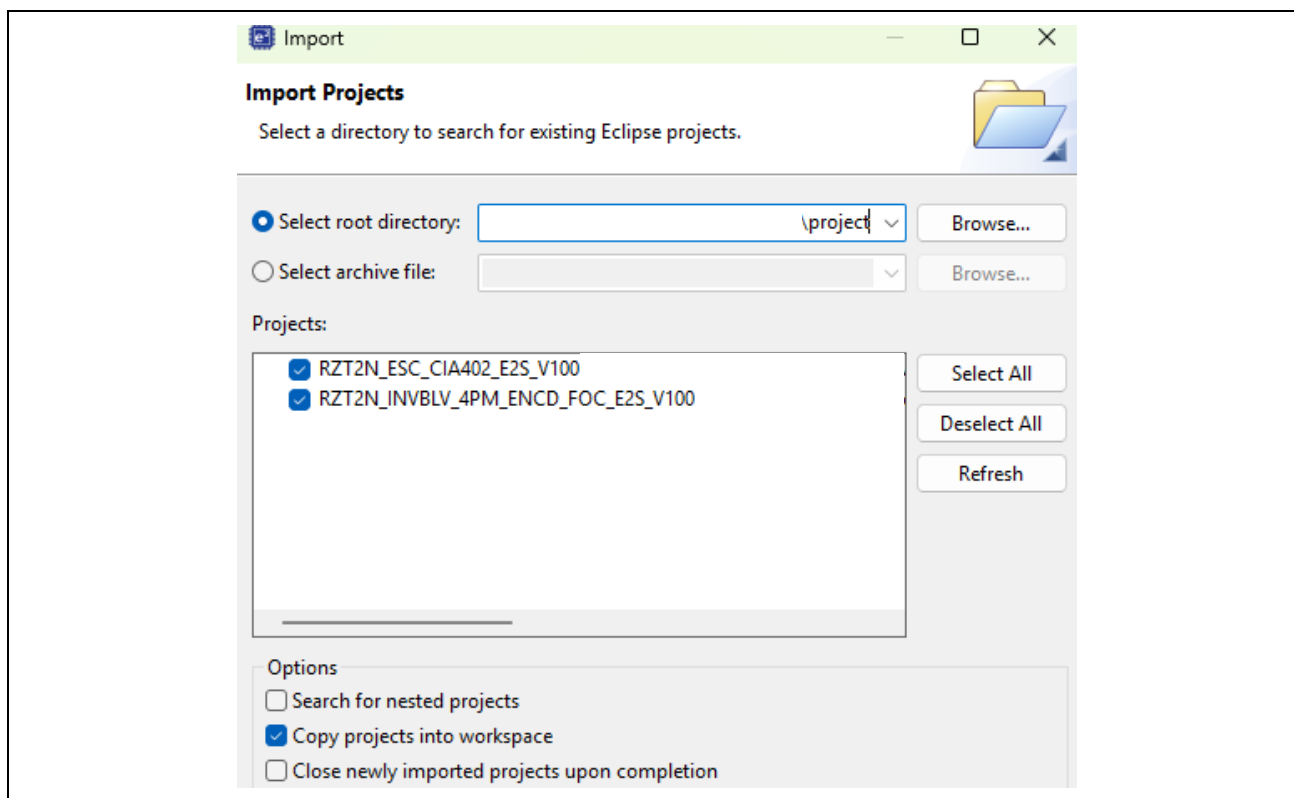


5.2 Building the Project

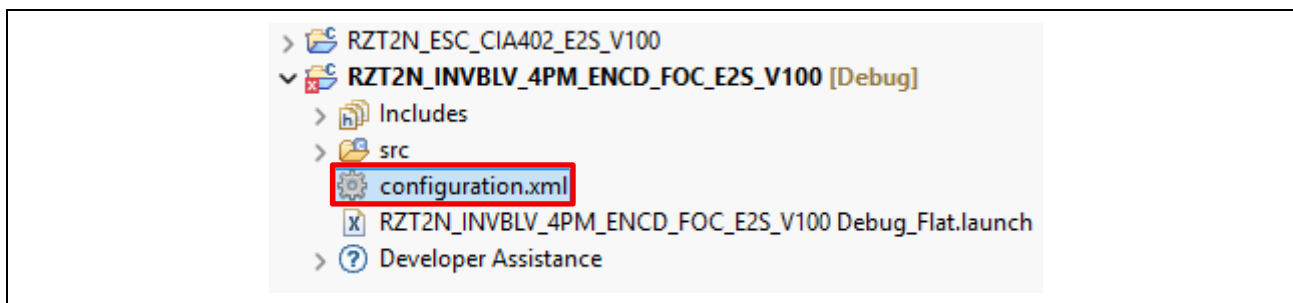
- (1) Launch e² studio, then from the top menu select: [File] > [Import] > [Existing Projects into Workspace], and click Next.



- (2) Select "Select root directory", then choose the "project" folder.
- (3) Check both : "RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100" and "RZT2N_ESC_CIA402_E2S_V100". Also, under Options, check "Copy projects into workspace" to import while copying into the workspace.



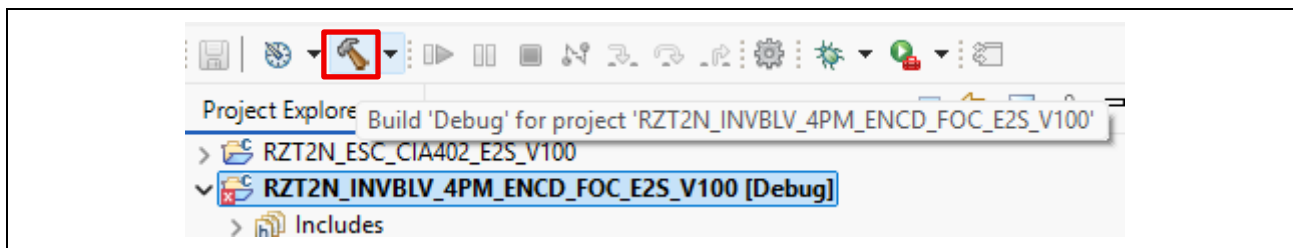
- (4) Open configuration.xml in the RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 project.



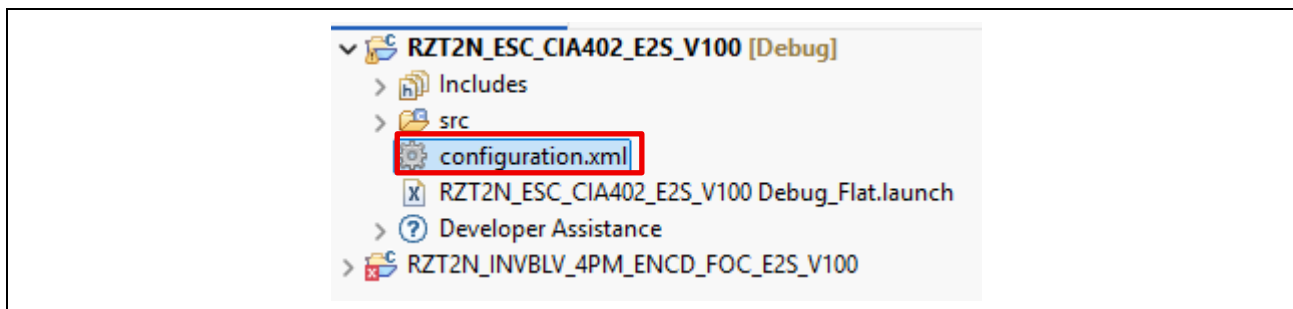
- (5) Perform code generation. Once code generation is complete, close configuration.xml.



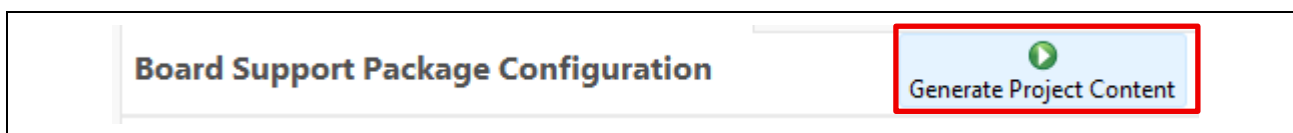
- (6) Build the RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 project.



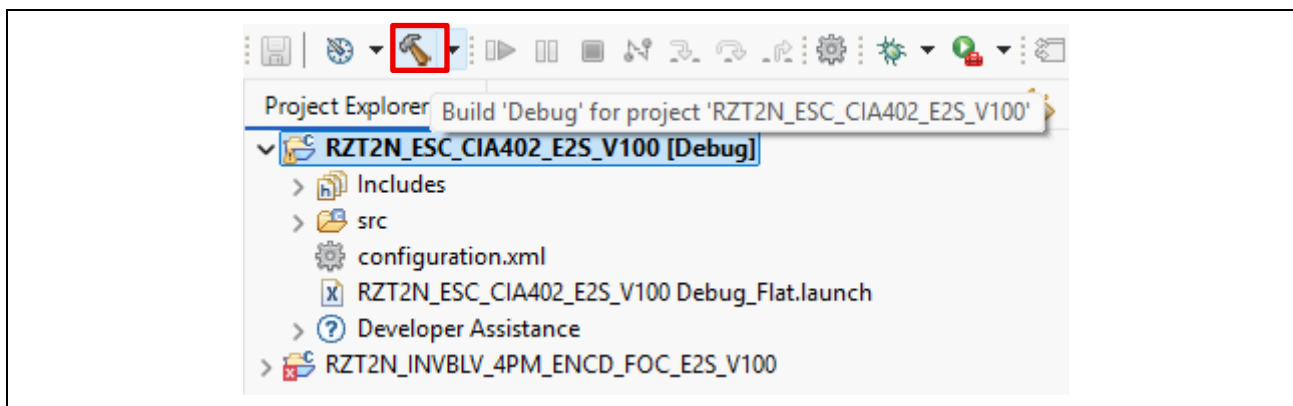
- (7) Open configuration.xml in the RZT2N_ESC_CIA402_E2S_V100 project.



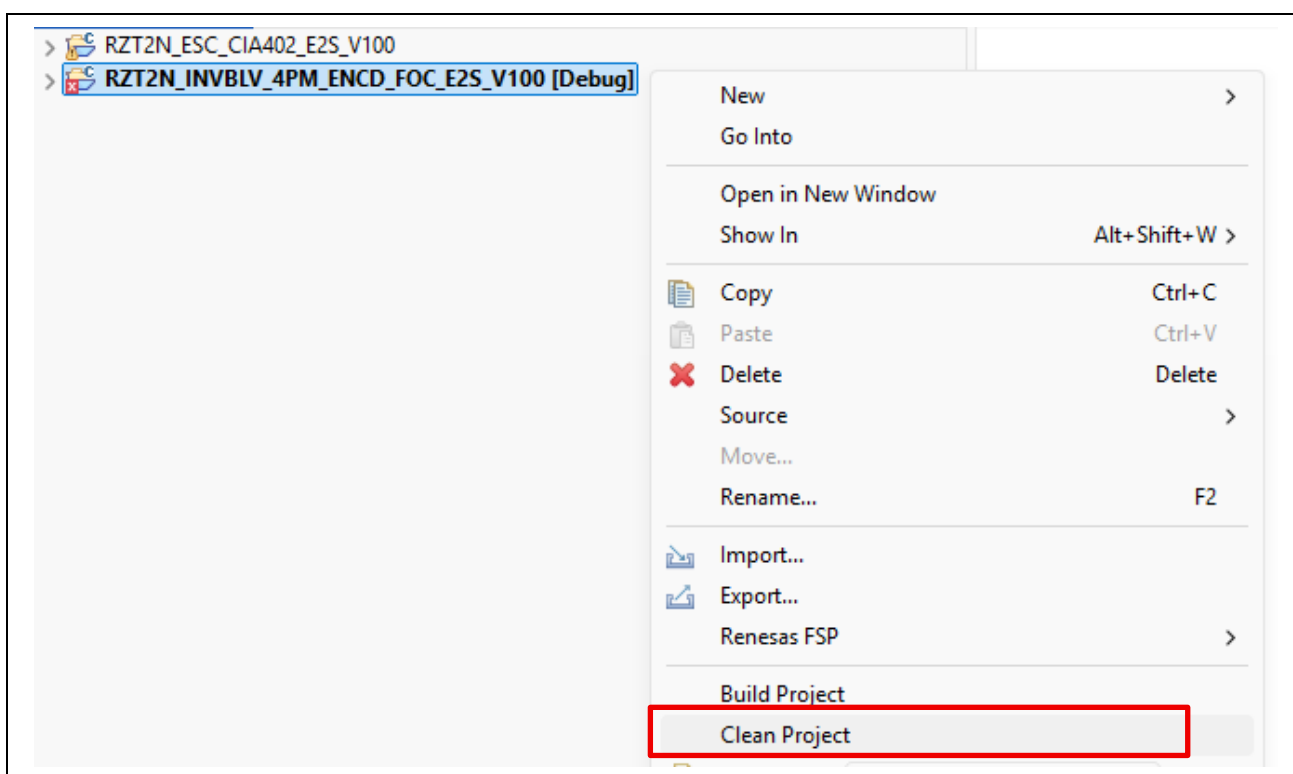
- (8) Perform code generation. Once code generation is complete, close configuration.xml.



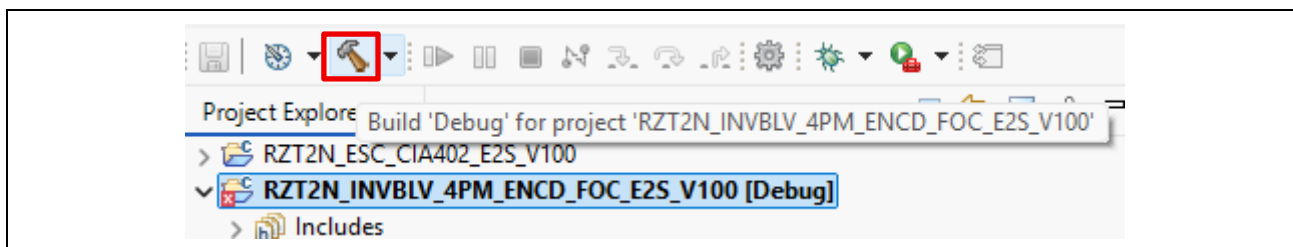
(9) Build the RZT2N_ESC_CIA402_E2S_V100 project.



(10) Clean the RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 project.

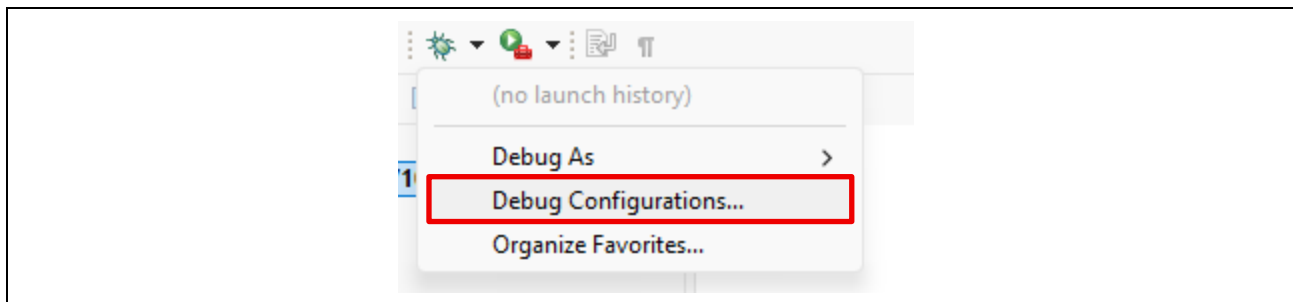


(11) After cleaning, build the RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 project again.

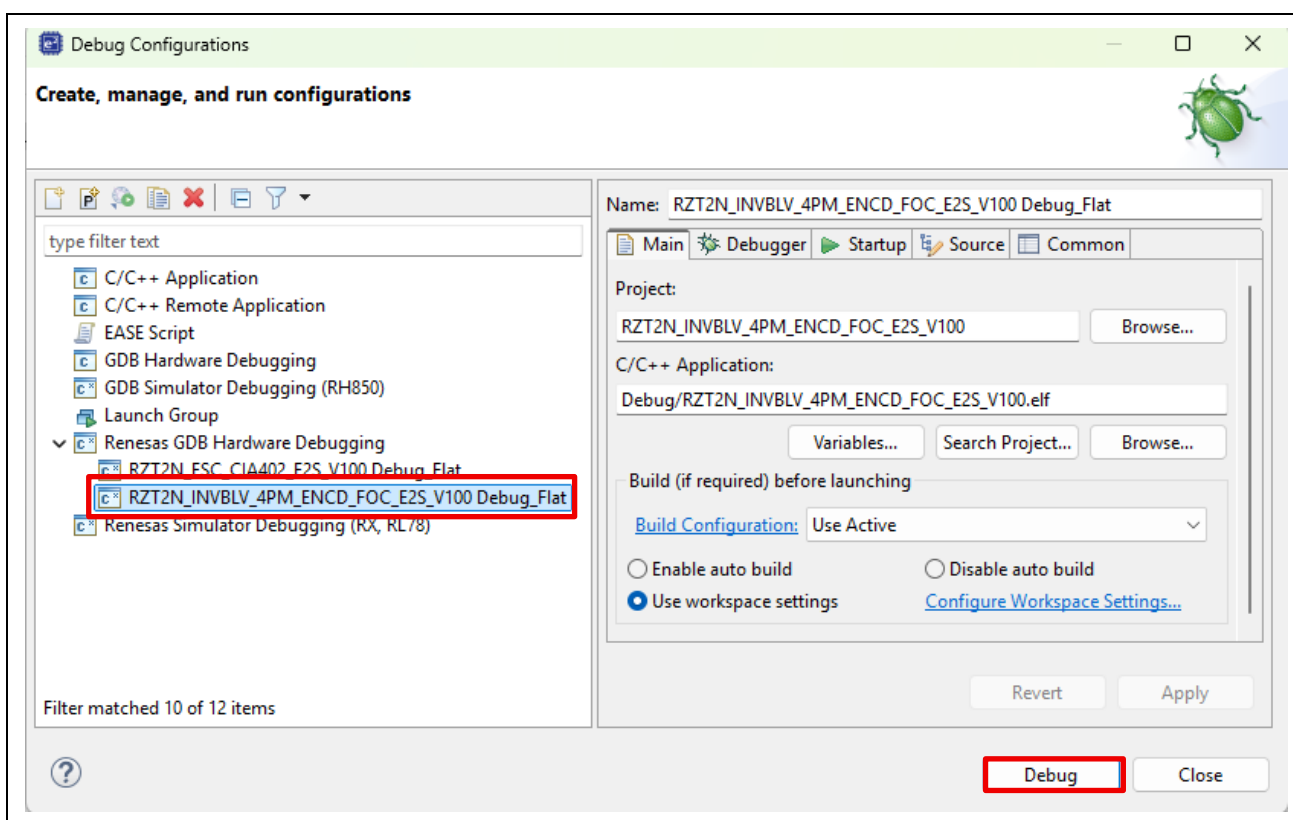


5.3 Writing the Project

- (1) Apply 24V DC to the RZ/T2N EVB and turn on the power.
- (2) Connect the J-Link debugger to the RZ/T2N, then open the Debug Configuration.



- (3) Select "RZT2N_INVBLV_4PM_ENCD_FOC_E2S_V100 Debug_Flat" and start debugging.



- (4) After the writing process completes and debugging starts, terminate the debug session and press the reset button on the board to restart.

This sample can be run standalone.

When USER LED1 (the green LED) lights up, the write has completed successfully on both CPU0 and CPU1.

6. How to Operate Program

6.1 Connecting to TwinCAT

6.1.1 Preparing the ESI File (First Time Only)

- (1) Before starting TwinCAT, you need to copy the ESI file for this sample to a specific folder.

Source ESI file (located in Common\ESI folder):

- “Renesas EtherCAT RZT2N MotorSolution.xml”

Destination folder (may vary depending on your environment):

- C:\TwinCAT\3.1\Config\Io\EtherCAT

6.1.2 Installing the TwinCAT Driver (First Time Only)

- (1) Launch TwinCAT 3.
- (2) From the Start Menu, select : [TwinCAT 3] → [Show Realtime Ethernet Compatible Devices...]
- (3) Select the Ethernet adapter you will use and install the driver.
- (4) If the selected Ethernet adapter moves to [Installed and ready to use devices (realtime capable)], the installation is successful.

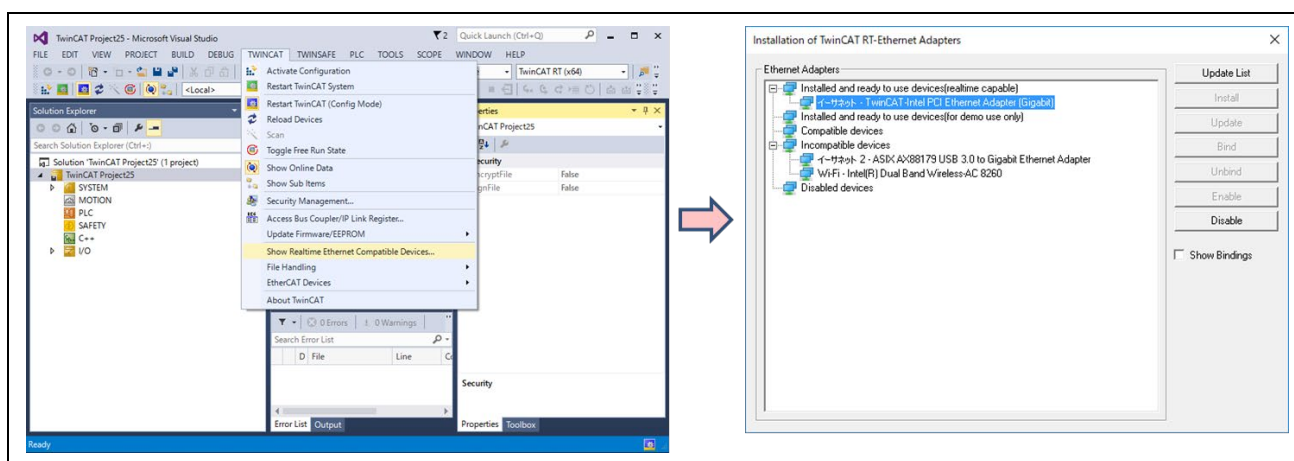


Figure 6.1 : TwinCAT Driver Installation

6.1.3 Scanning Devices and Writing to EEPROM

- (1) Create a new project in TwinCAT.
- (2) In Solution Explorer → I/O → Devices, select “Scan”.

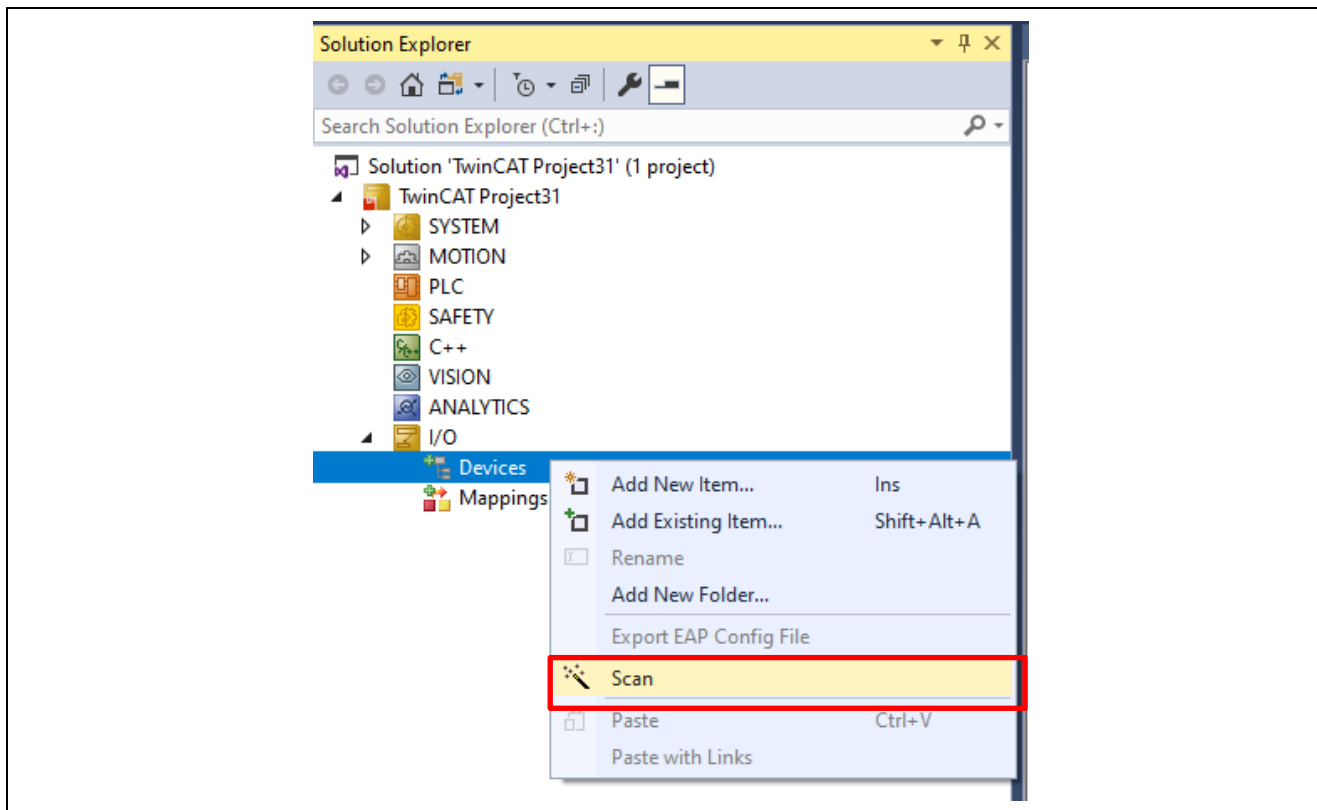


Figure 6.2. Device Scan

- (3) If the RZ/T2N is recognized as an EtherCAT device, the EtherCAT port will be displayed as shown below.

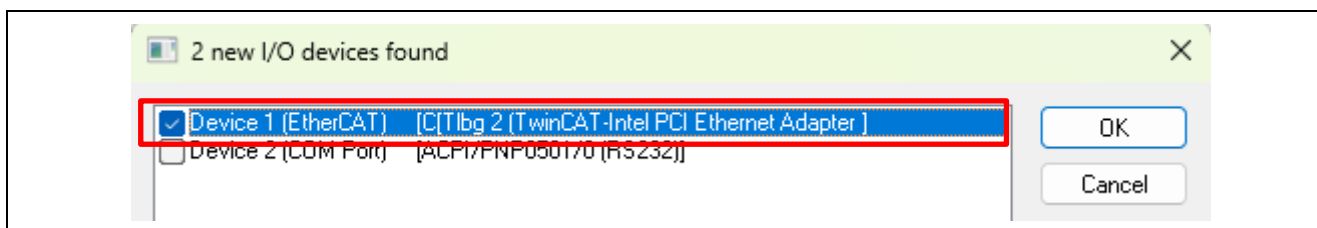


Figure 6.3. Scan Result

(4) Execute [Scan for boxes].

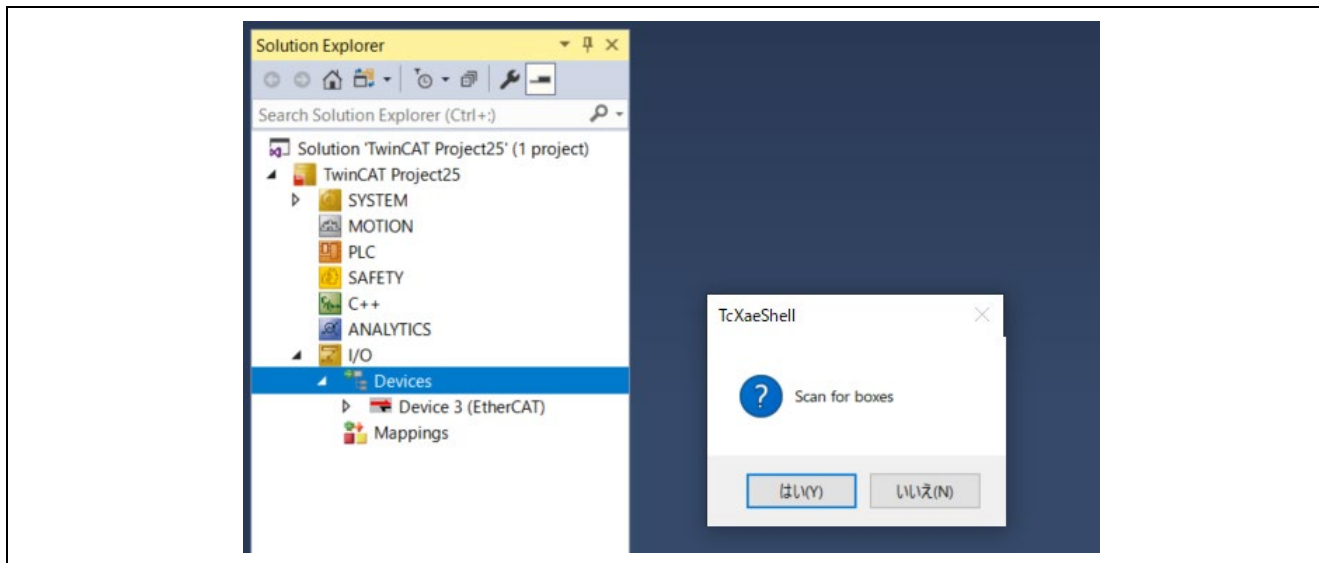


Figure 6.4. Scan for boxes Window

(5) When prompted for [Activate free run], click OK.

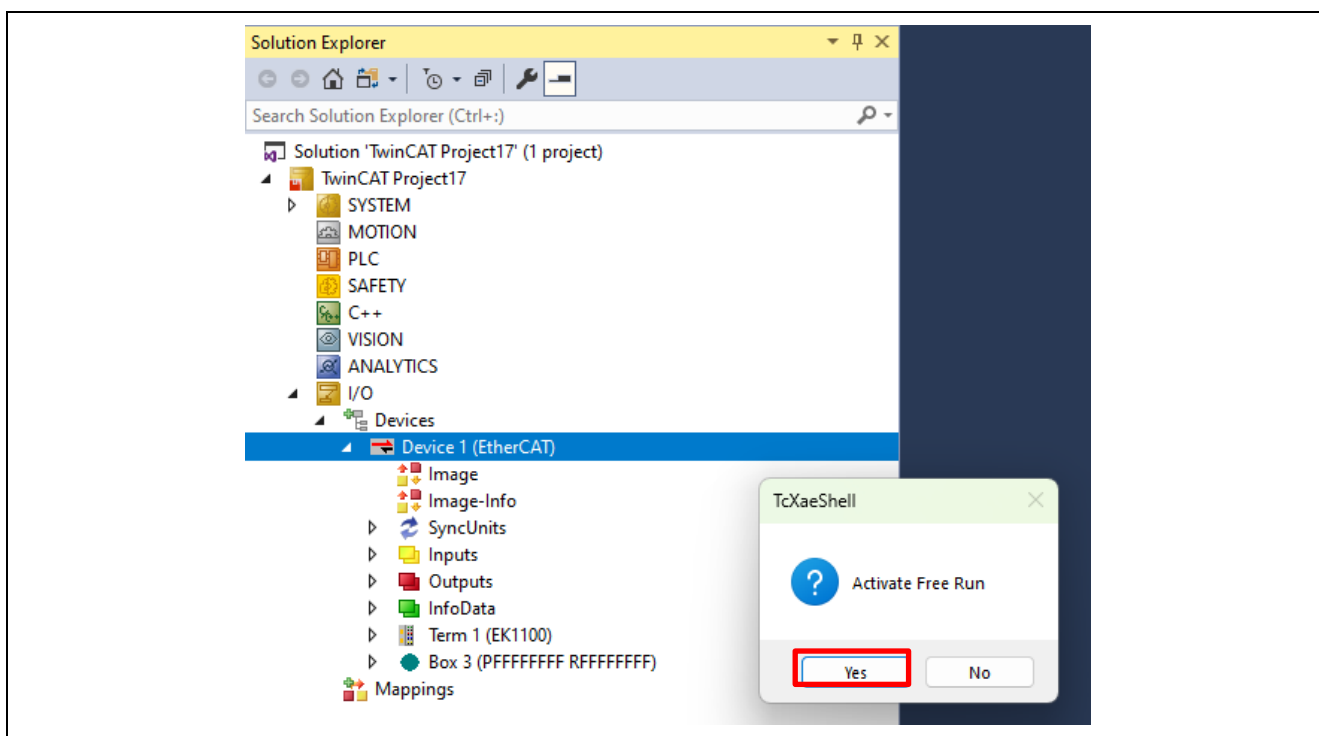


Figure 6.5. Activate Free Run

(6) If Box x (PFFFFFF..) is displayed, the RZ/T2N EVB's EEPROM is empty. You need to write the EtherCAT SubDevice information for this sample to the EEPROM.

(7) Steps to write to EEPROM:

- Double-click [Box X] to open the settings screen.
- Select the [EtherCAT] tab.
- Click [Advanced Setting].

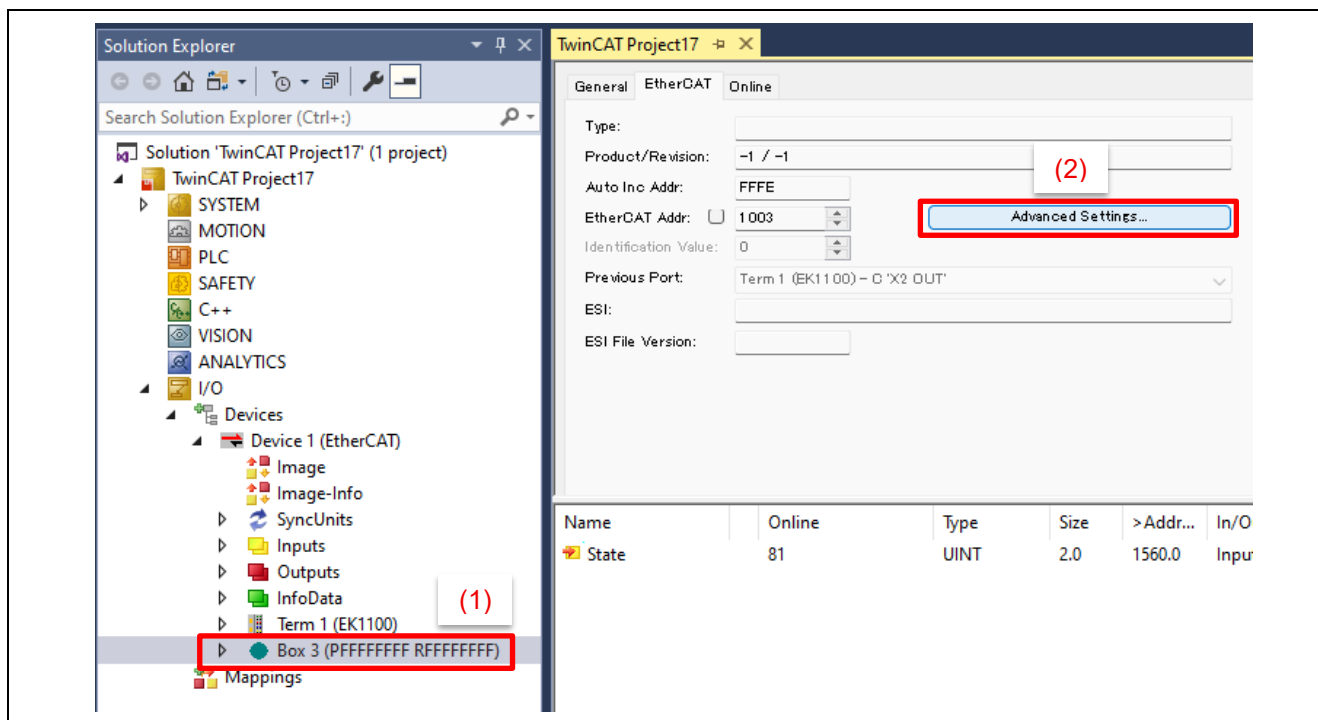


Figure 6.6. EEPROM Write Method 1

- Navigate to [ESC Access] → [EEPROM] → [Hex Editor].
- Select [Download from List].

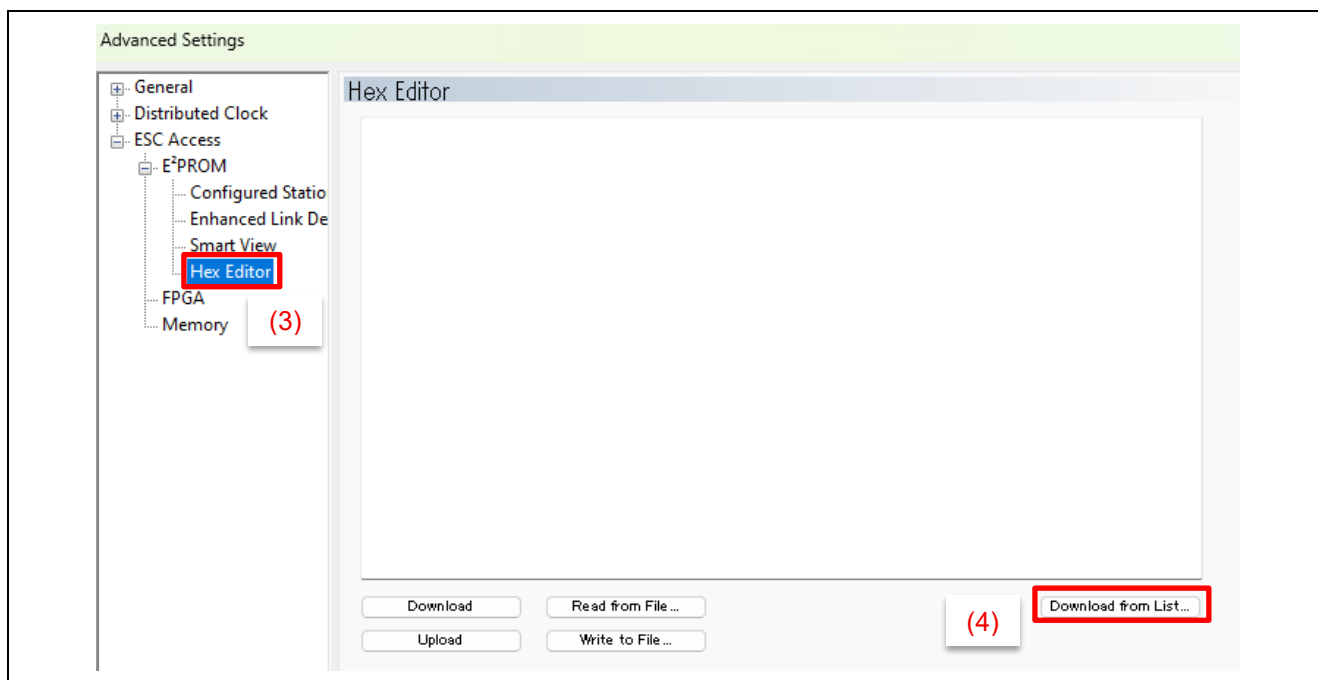


Figure 6.7. EEPROM Write Method 2

- A list of ESI files registered in TwinCAT 3 will appear. Select the appropriate file. For this sample, choose : [Renesas EtherCAT RZ/T2N MotorSolution].

Renesas EtherCAT RZ/T2N MotorSolution (2096 / 256) (5)

Figure 6.8. EEPROM Write Method 3

- (8) After writing to EEPROM, press Reset button on the board to restart the RZ/T2N and reload the EEPROM.
- (9) In TwinCAT, delete [Device x (EtherCAT)] and perform a rescan.

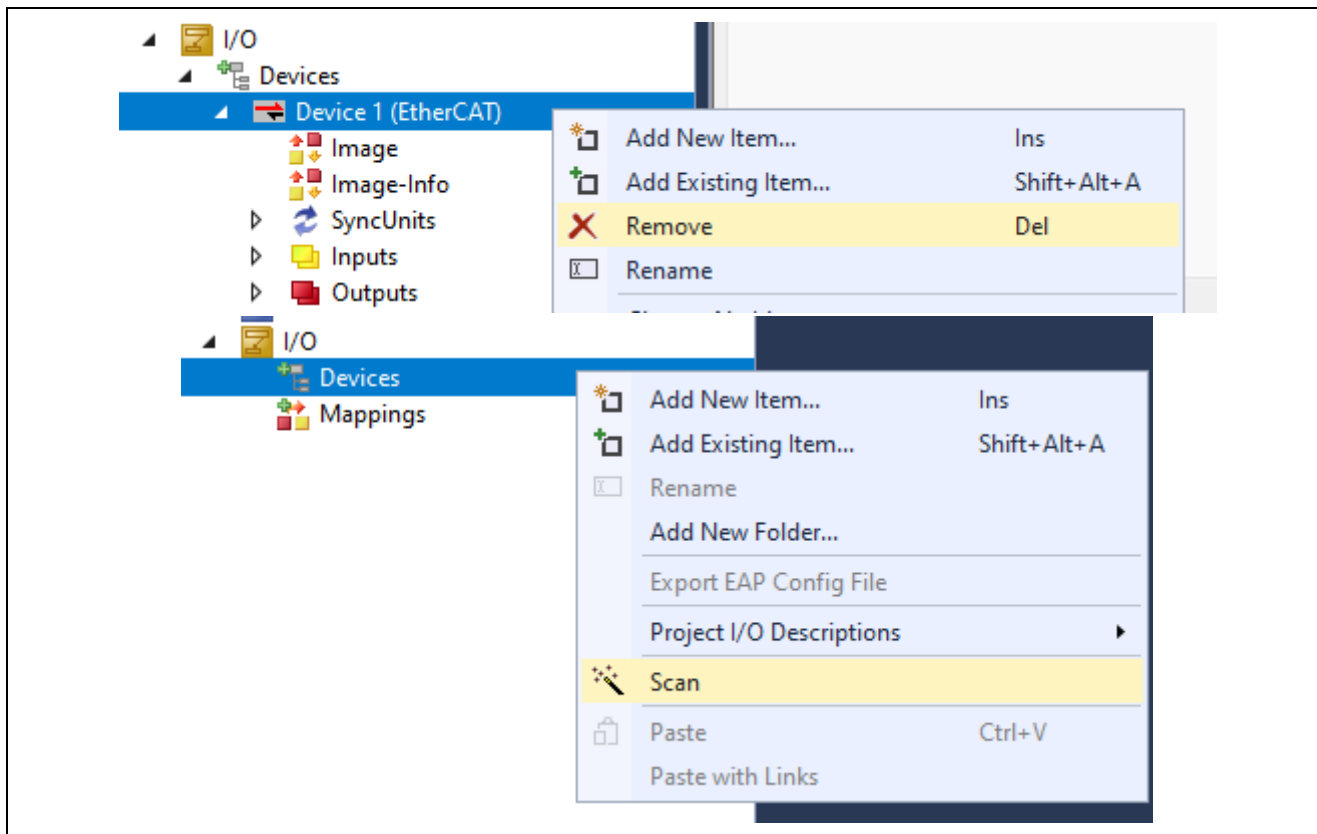


Figure 6.9. Rescanning Devices

(10) If the scan is successful, the following display will appear.

Since the EtherCAT drive is not used for operation verification, select [Cancel].

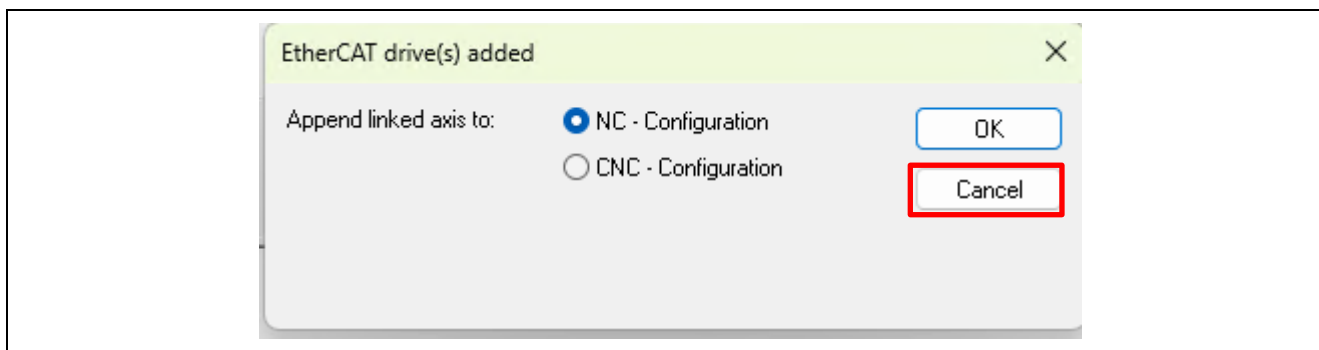


Figure 6.10. EtherCAT Drive Selection

(11) Execute [Reload Devices] to restart the network.

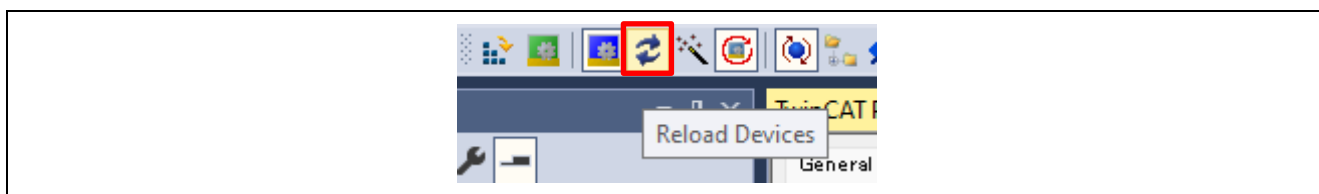


Figure 6.11. Network Restart

(12) Confirm that the device appears in the tree as shown below and that the Current State in the [Online] tab is "OP".

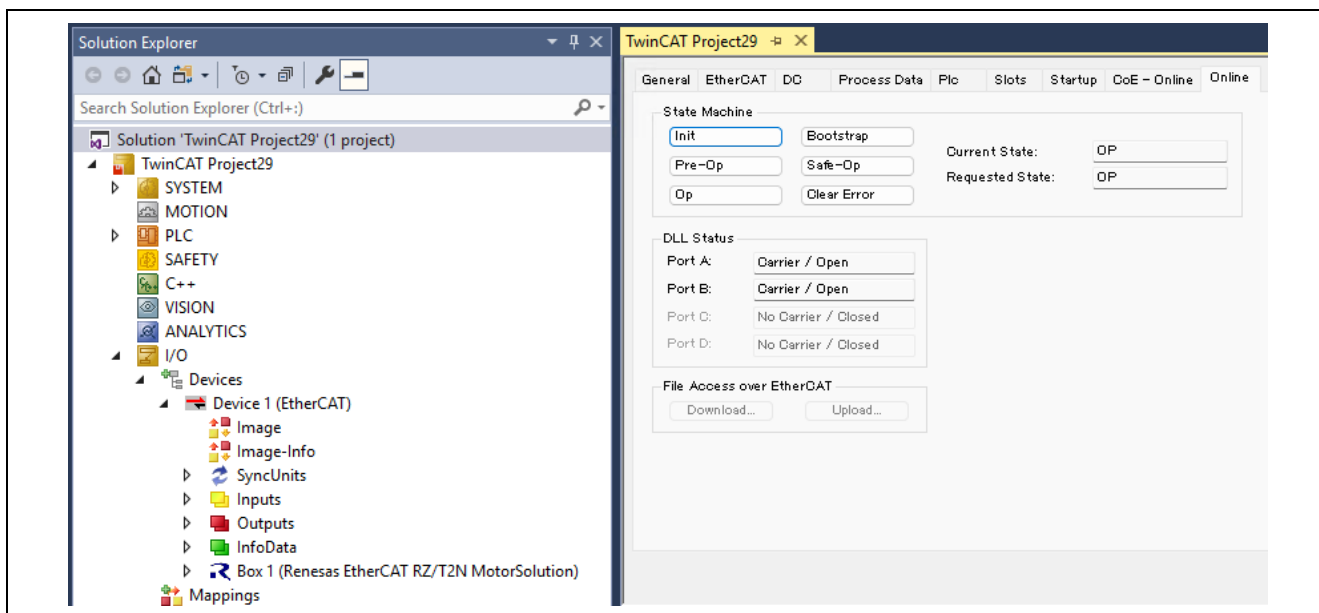


Figure 6.12. Confirming "OP" State

6.2 How to Run the TwinCAT Project

First, power on the RZ/T2N EVB and ensure that the program written to the microcontroller on the board is running.

The following steps describe how to execute the TwinCAT project.

- (1) Set the EtherCAT Communication Cycle. This example shows how to set the EtherCAT communication cycle to 250 μ s:
 1. In the TwinCAT 3 System Manager tree, right-click [PLC].
 2. Select [Add New Item...] to create a new project.
 3. Select [Standard PLC Project].
 4. Enter any project name.
 5. Click [Add].

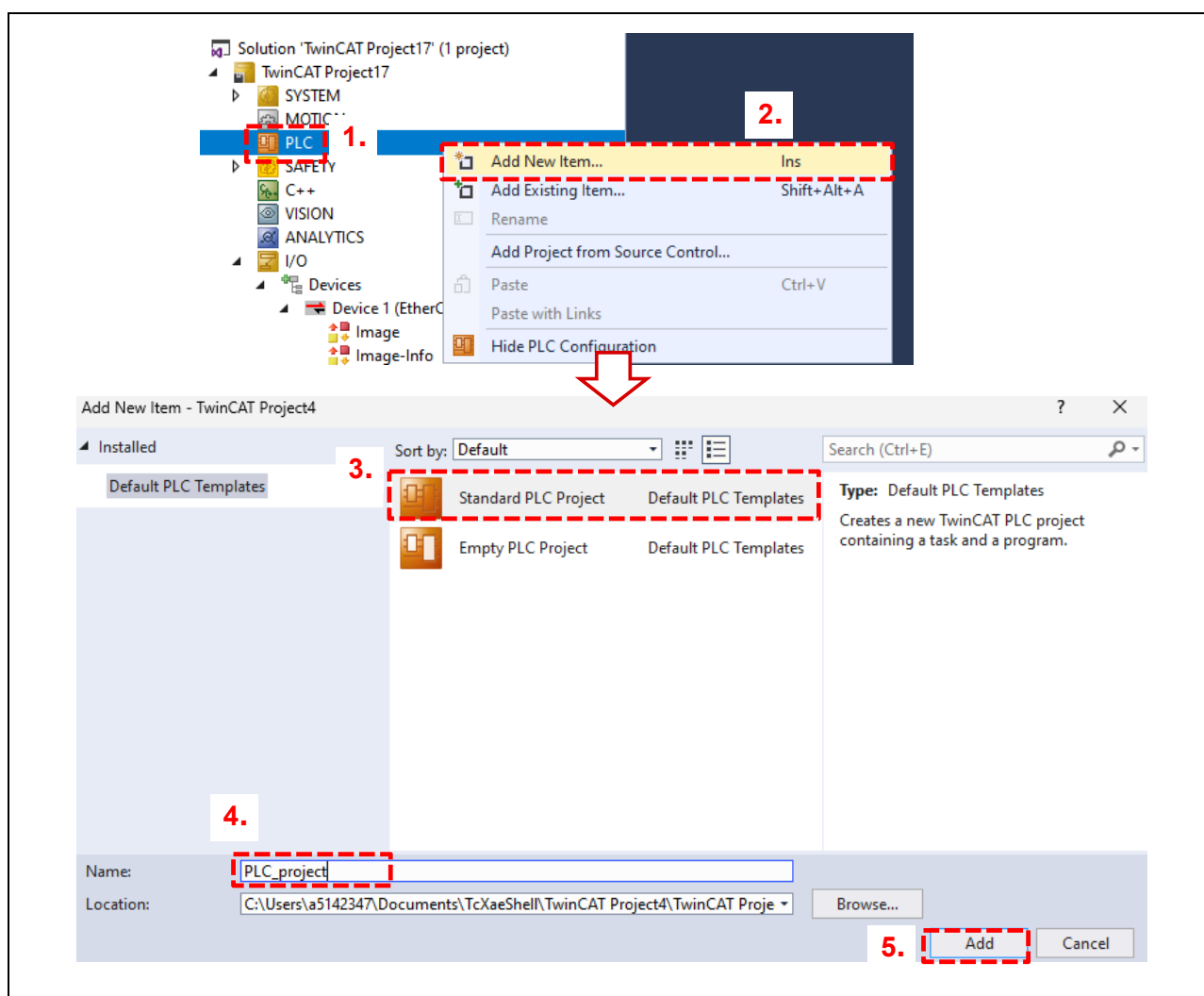


Figure 6.13 Create a PLC project

6. In the TwinCAT 3 System Manager tree, click [SYSTEM] → [Real-Time] to open the properties window.
7. Click the [Settings] tab.
8. From the [Base Time] dropdown menu, select “250 μ s”.
9. In the System Manager tree, click [SYSTEM] → [Tasks] → [PlcTask] to open the properties window.
10. Click the [Task] tab.
11. In the left-side setting menu for [Cycle ticks], set the value to “1” using the spin button.
12. Confirm that the right-side setting menu for [Cycle ticks] shows “0.250 ms”.

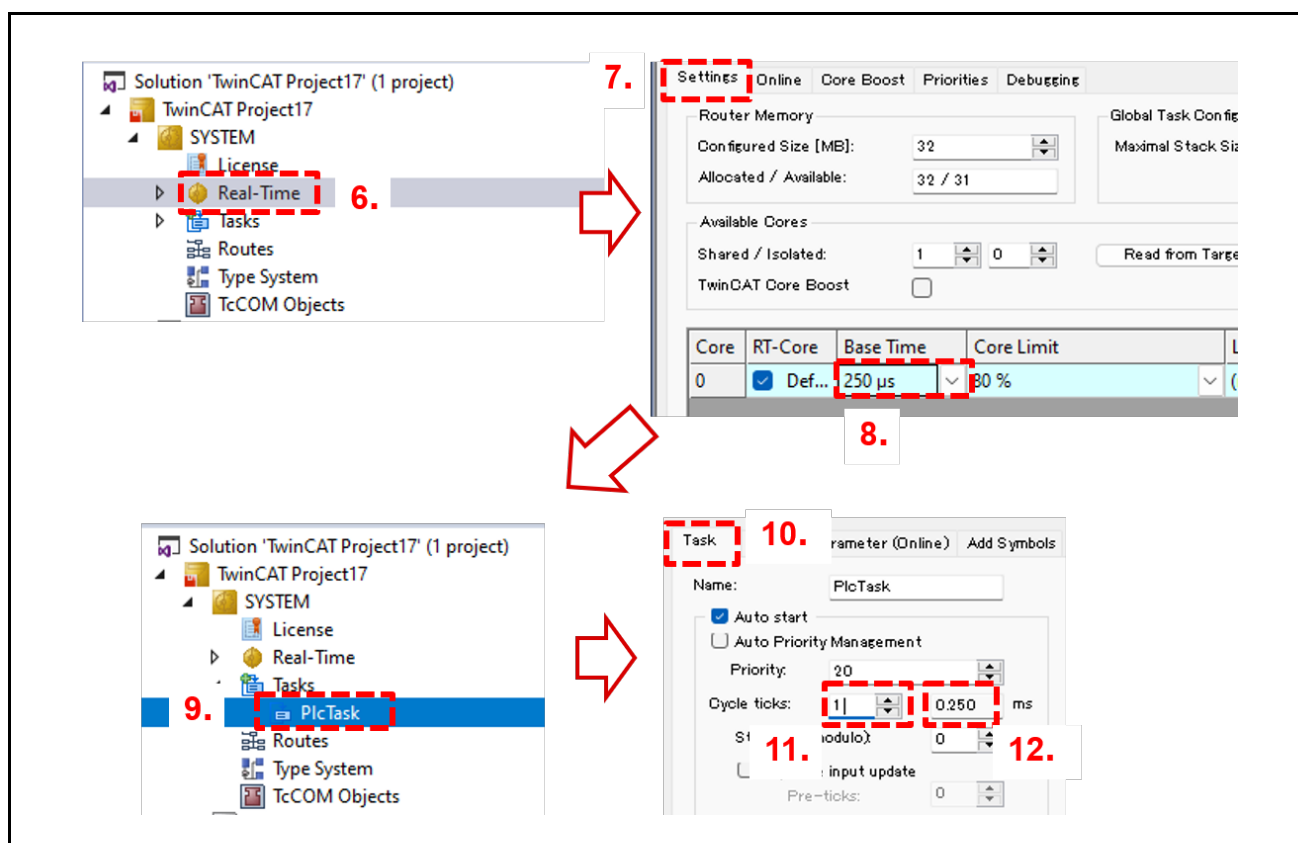


Figure 6.14. Setting EtherCAT Communication Cycle

(2) Set EtherCAT Synchronization Mode to DC Synchronization

1. In the System Manager tree, click [Box X (Renesas EtherCAT RZ/T2N MotorSolution)] to open the properties window.
2. Click the [DC] tab.
3. Under Operation Mode, select [DC-Synchron].
4. Click [Advanced Settings...].
5. Check [Enable SYNC1].
6. Set the timing for SYNC1 interrupt in [Shift Time (μs)].
7. Click [OK].

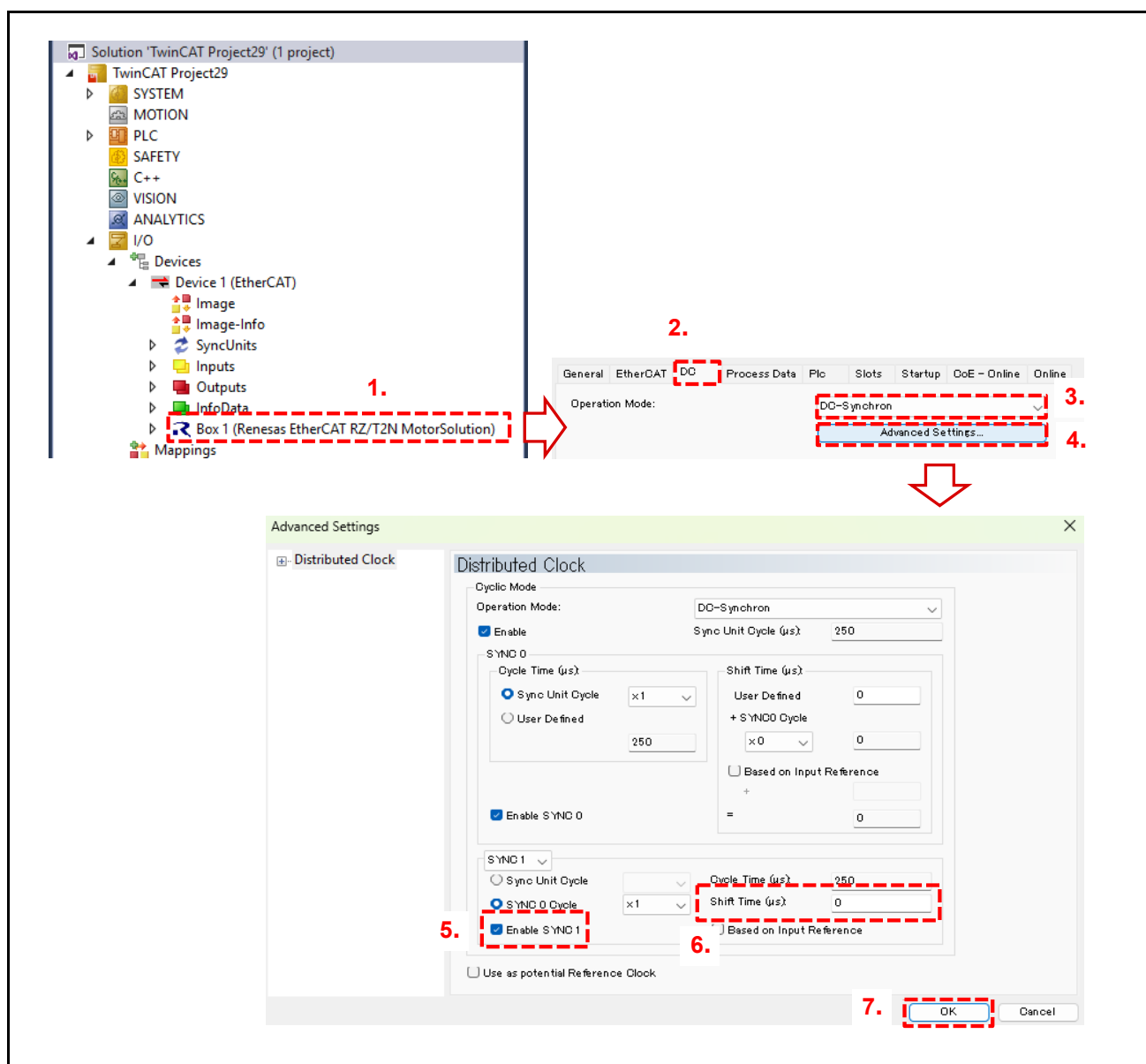


Figure 6.15. DC Synchronization Settings Part 1

8. In the System Manager tree, click [I/O] → [Devices] → [Device X (EtherCAT)] to open the properties window.
9. Click the [EtherCAT] tab.
10. Click [Advanced Settings...] to open the Advanced Settings window.
11. In the Advanced Settings window, click [Distributed Clocks].
12. Set the timing for SYNC0 interrupt under Percent of cycle time.
13. Click [OK].

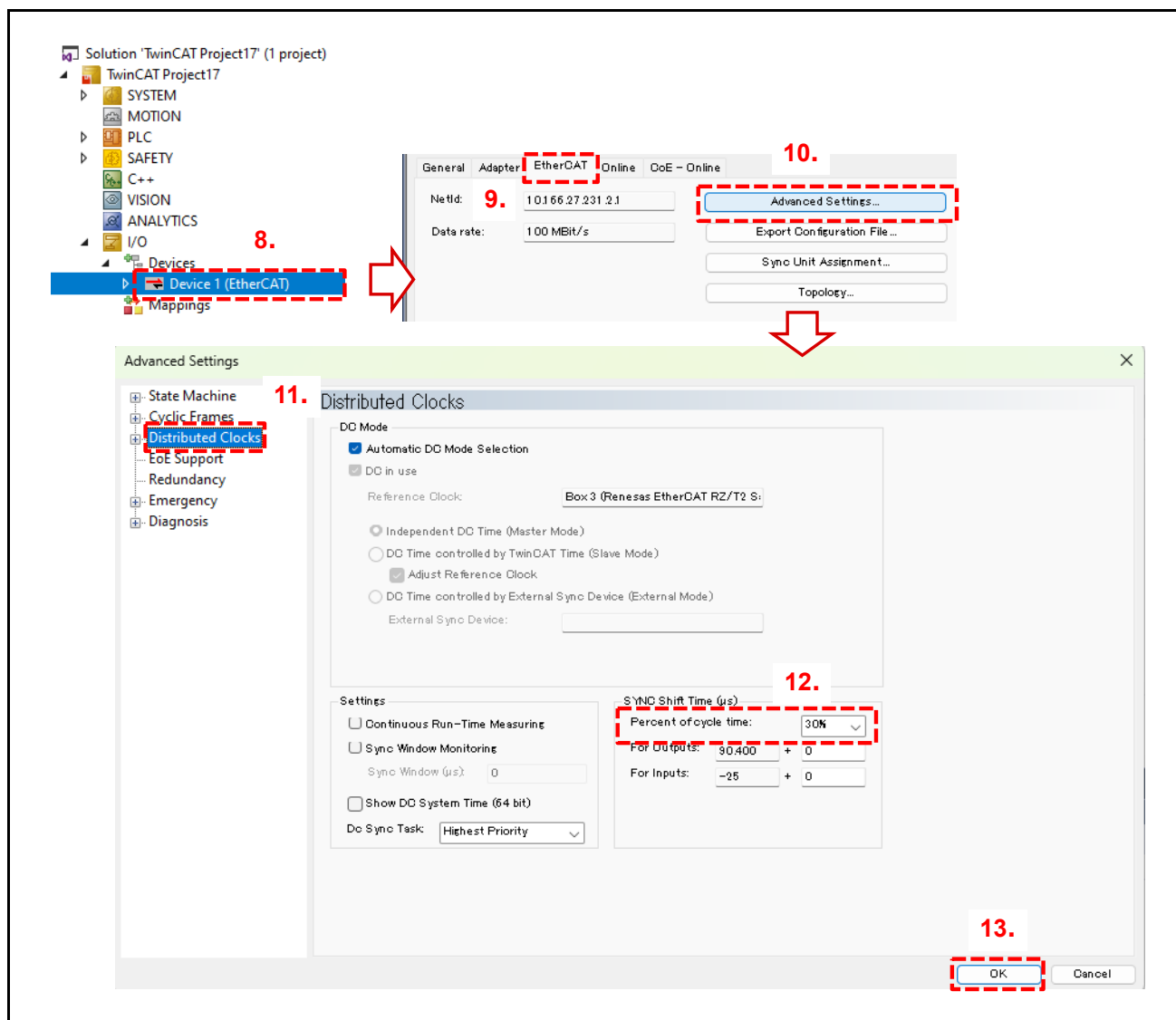


Figure 6.16. DC Synchronization Settings Part 2

(3) Enable EtherCAT Network Settings

1. In the System Manager tree, click [I/O] → [Devices] → [Device X (EtherCAT)] to open the properties window.
2. Click the [EtherCAT] tab.
3. Click [Sync Unit Assignment...] to open the Sync Unit Assignment window.
4. If Task is set to <default>, change it to PlcTask.
5. Click [Apply], then [OK].

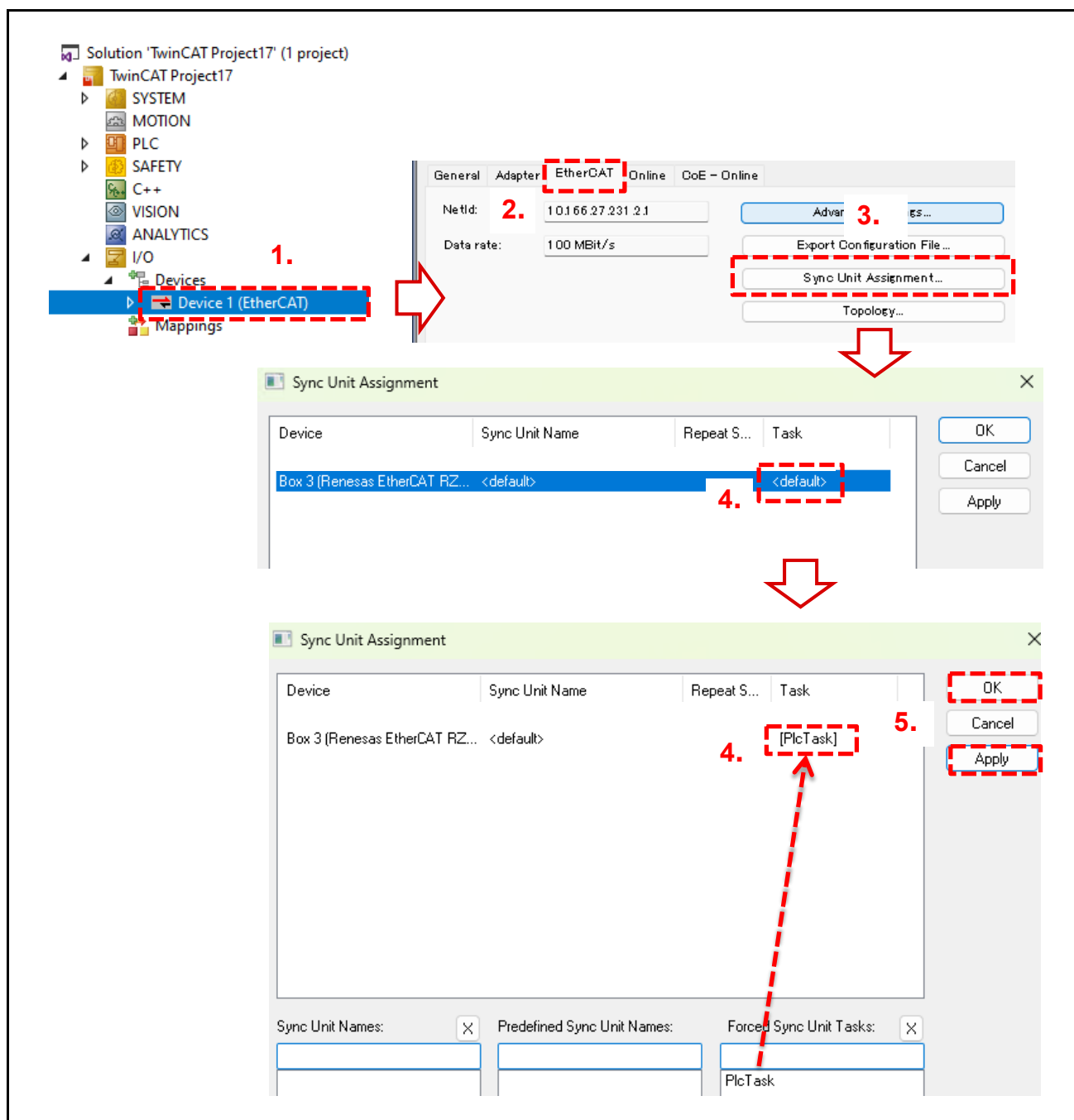


Figure 6.17. Sync Unit Assignment Settings

6. Click the [Activate Configuration] icon in TwinCAT 3.
7. When the Activate Configuration window appears, click [OK].
8. When the Restart TwinCAT System in Run Mode window appears, click [OK] to switch the TwinCAT 3 project to Run Mode.

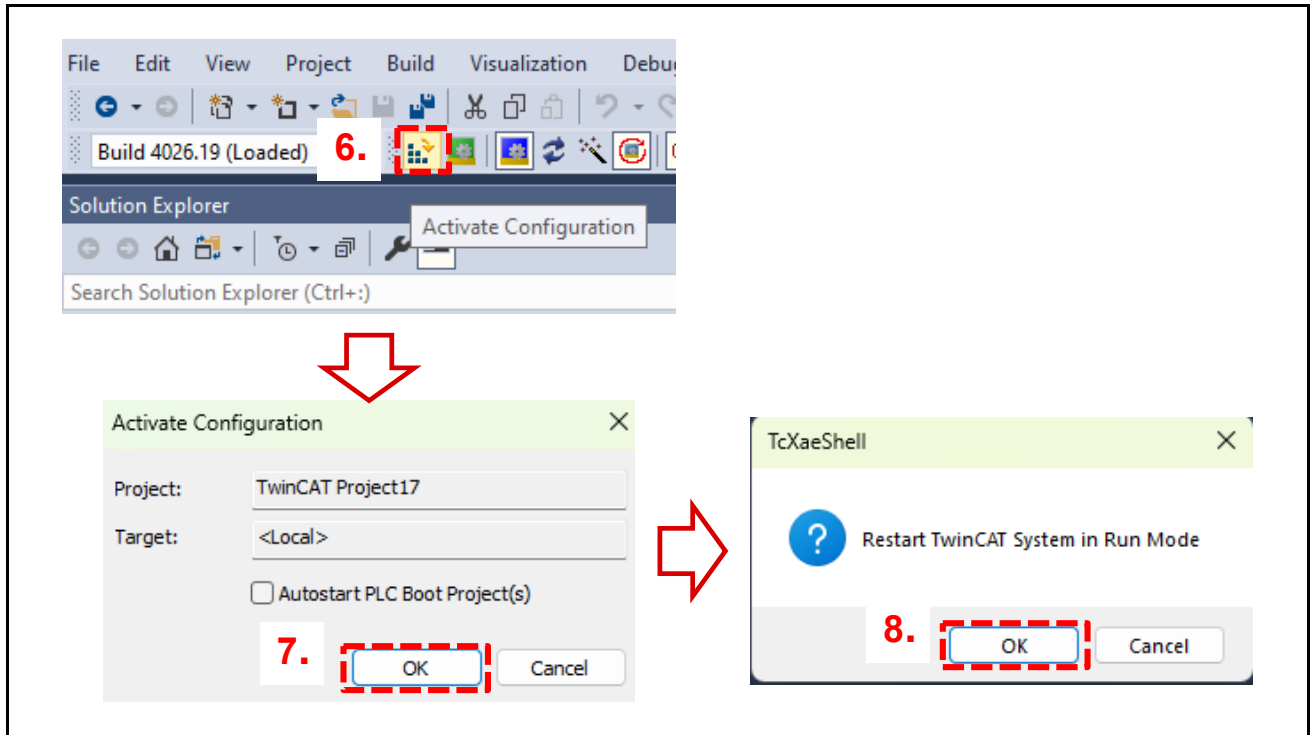


Figure 6.18. Starting Run Mode

(4) Confirm Connection Between TwinCAT 3 and RZ/T2N EVB

1. In the System Manager tree, double-click [Box X (Renesas EtherCAT RZT2N MotorSolution)] to open the properties window.
2. Click the [Online] tab.
3. If "Current Status" shows [OP], TwinCAT 3 is successfully connected to the RZ/T2N EVB.

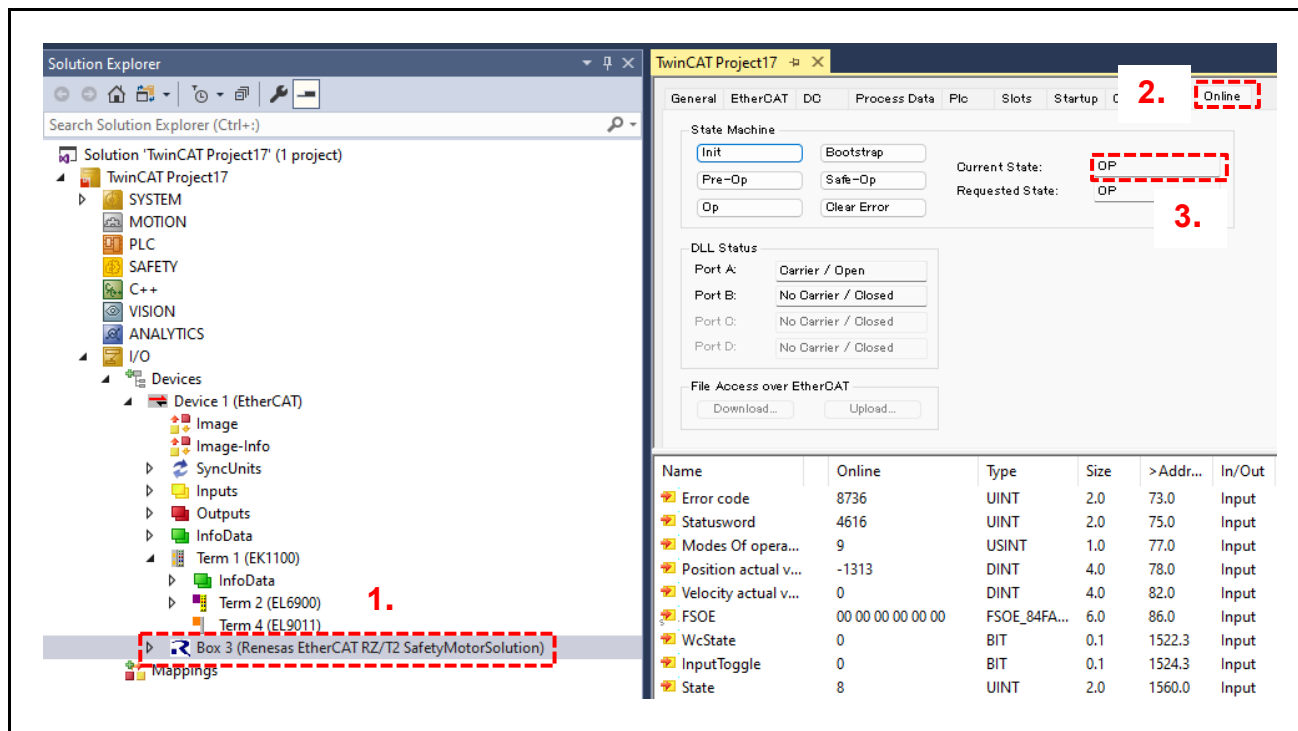


Figure 6.19. Confirming Connection Status

6.3 How to Verify Operation

This section explains the procedure for verifying the operation of the RZ/T2N EVB.

6.3.1 Selecting CiA402 Operating Mode

- (1) In TwinCAT 3 System Manager tree, double-click [Box X (Renesas EtherCAT RZT2N MotorSolution)] to open the properties window.
- (2) Click the [Slots] tab.
- (3) In the left pane of the [Slots] tab, click the [Motor1] row.
- (4) In the right pane, select the desired CiA402 operating mode.
- (5) Click the [<] button to apply the change.
- (6) Confirm that the [Motor1] row now shows the selected CiA402 operating mode.
- (7) The CiA402 operation mode can also be changed for [Motor2], [Motor3], and [Motor4] by following the same procedure.
- (8) Click [Activate Configuration] in TwinCAT 3 to restart in Run Mode.

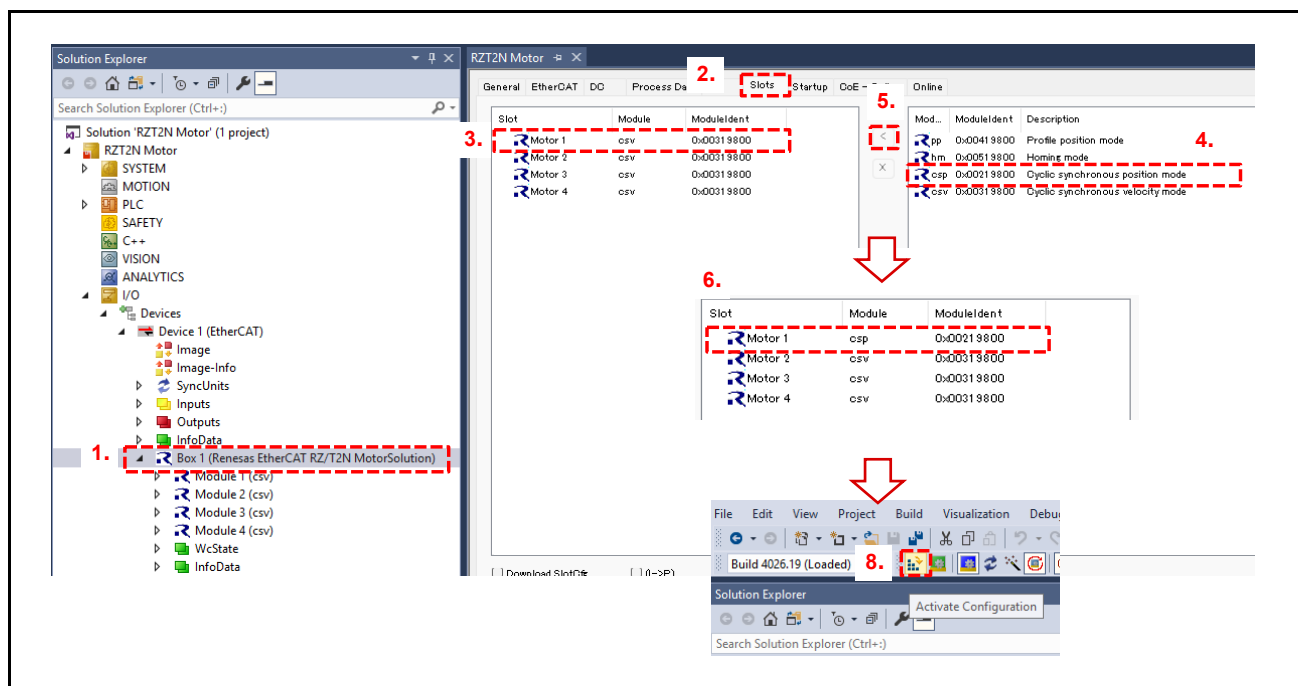


Figure 6.20. Changing CiA402 Operating Mode

6.3.2 Controlling CiA402 State Machine

6.3.2.1 How to Transition to Operation Enable

1. In TwinCAT 3 System Manager tree, navigate to:[Box X (Renesas EtherCAT RZT2N MotorSolution)] → [Module 1] → [Inputs] → [Statusword], and double-click.
2. Open the [Flags] tab.
3. Change the Display mode of [Statusword] to Hex.
4. Open the [Online] tab.
5. Confirm that bit 3 of Statusword is 0. If it is 1, the system is in Fault state and recovery is required. Perform the steps in Section 6.3.3 to recover from Fault.
6. By applying the same procedure to [Module 2] through [Module 4], you can check the [Statusword] for all 4 motor axes.

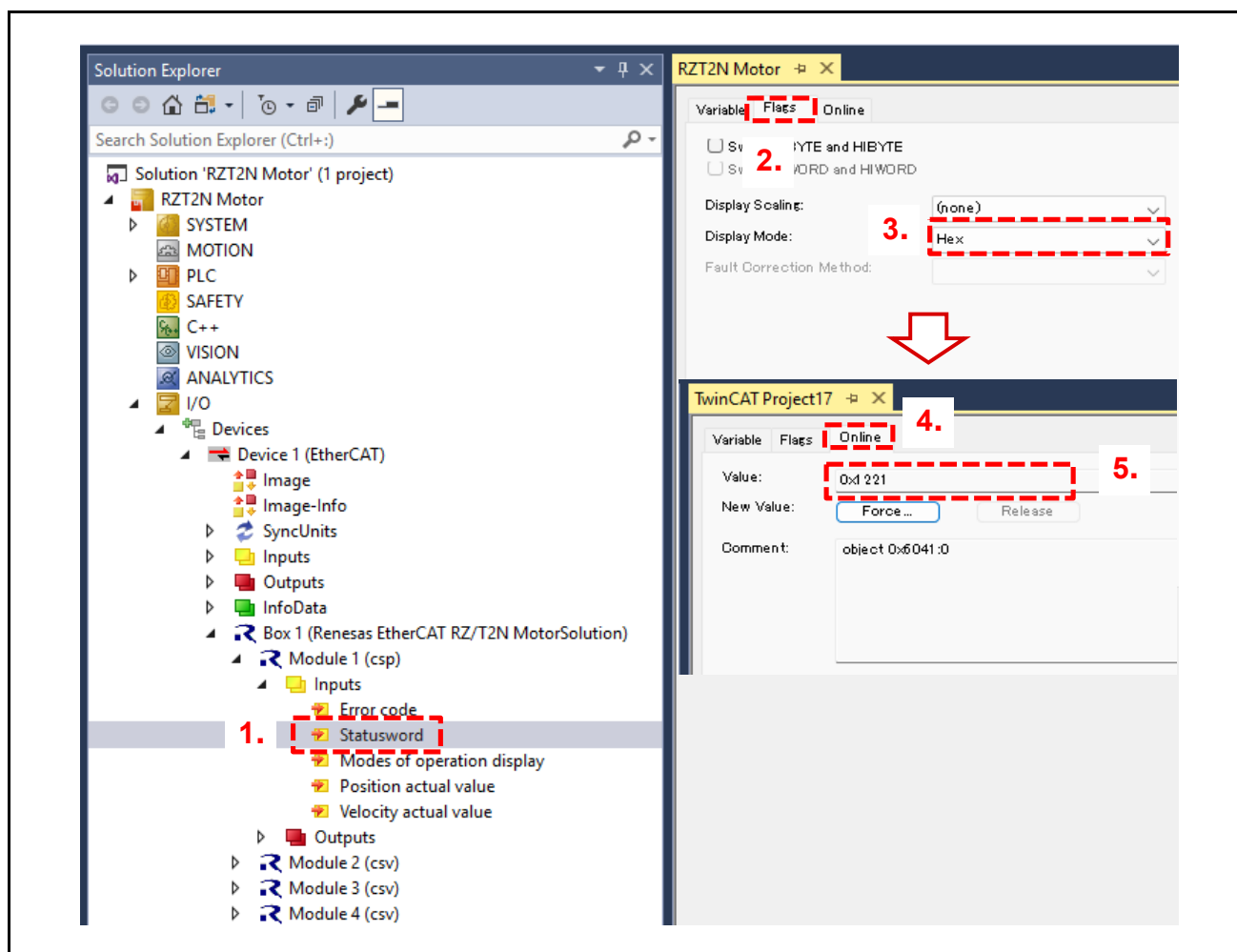


Figure 6.21. Transition to Operation Enable – Part 1

7. Navigate to [Outputs] → [Controlword], click it.
8. open the [Online] tab.
9. Click [Write].
10. Enter 15 and click OK.
11. Confirm the value of [Statusword]. If bit 2 is 1, the transition to Operation Enable was successful.
12. Apply the same procedure to [Module 2] through [Module 4] as well, transitioning all motors to Operation enable.

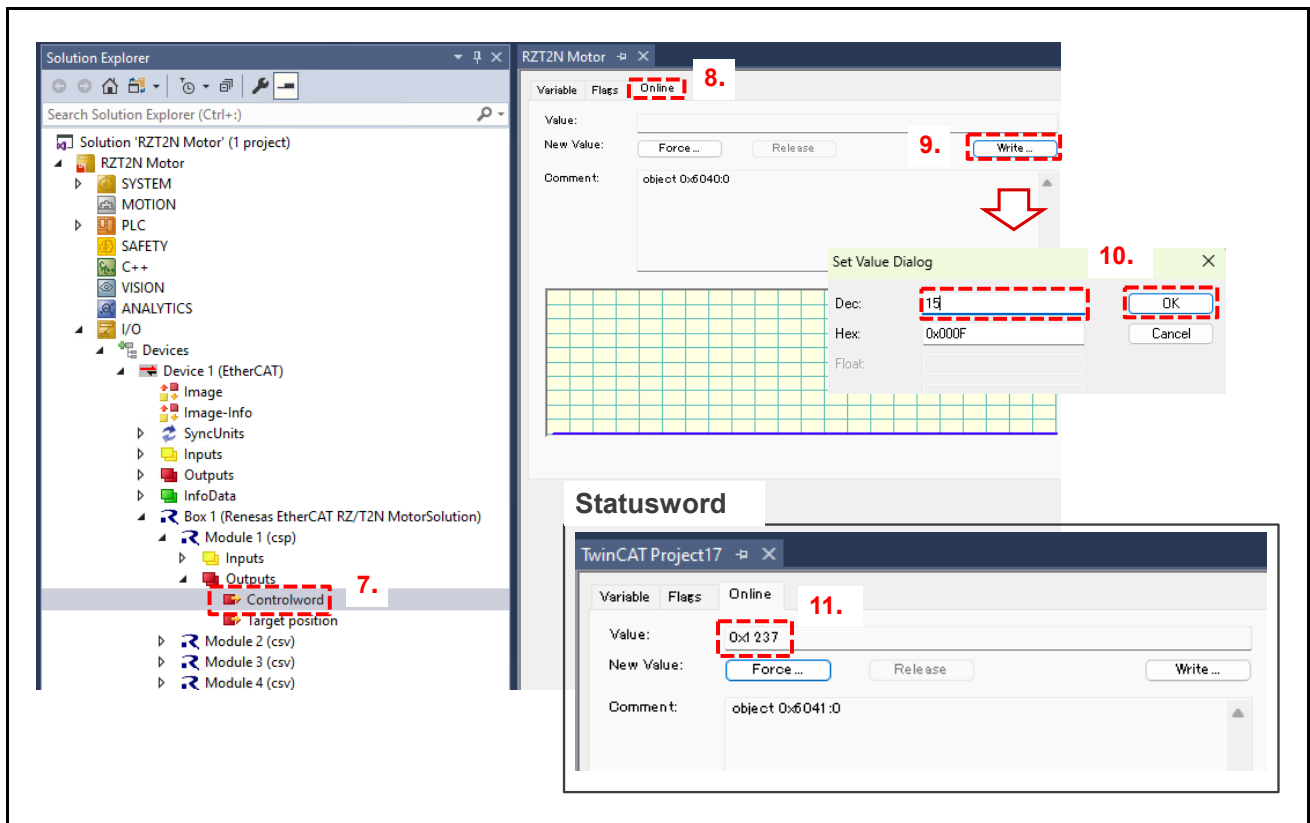


Figure 6.22. Transition to Operation Enable – Part 2

13. To return from Operation enabled to the original state, write “6” to [Controlword].

6.3.3 How to Recover from Fault

1. If bit 3 of [Statusword] is 1, the system is in Fault state.

Statusword

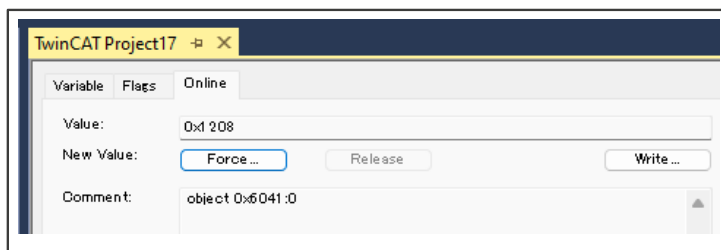


Figure 6.23. Identifying Fault State

2. In TwinCAT 3 System Manager tree, navigate to:[Box X (Renesas EtherCAT RZT2N MotorSolution)] → [Module 1] → [Outputs] → [Controlword].
3. Open the [Online] tab.
4. Click [Write].
5. Enter 128 and click OK.
6. Confirm that bit 3 of Statusword is now 0. Fault recovery is complete.
7. Fault recovery is also possible for [Module 2] through [Module 4] by following the same procedure.

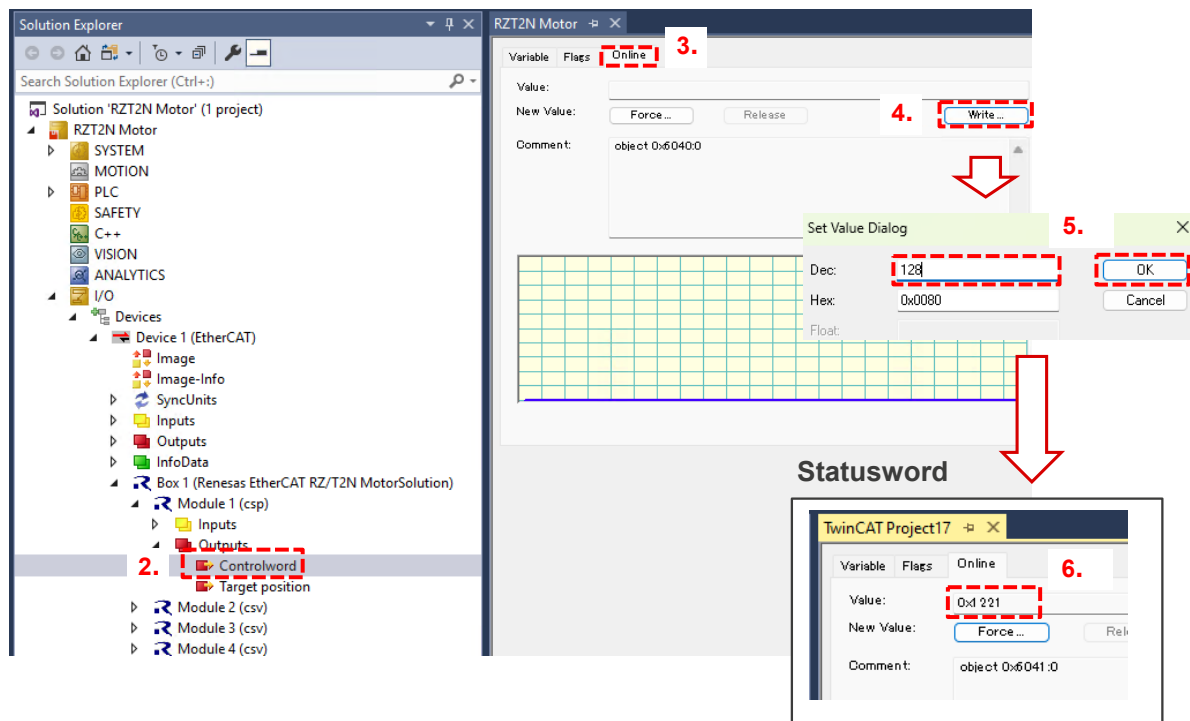


Figure 6.24. Fault Recovery

6.3.4 Motor Operation Methods

6.3.4.1 CSP Mode (Cyclic Synchronous Position)

1. Follow the steps in Section 6.3.1 to set the motor's CiA402 operating mode of the motor you want to operate to csp mode.
2. Follow the steps in Section 6.3.2 to transition the CiA402 state machine to Operation Enable.
3. In TwinCAT 3 System Manager tree, click [Target position].
4. Open the [Online] tab and click [Write].
5. Enter 36000 to issue a position command for the motor to move to 3600.0 deg.
6. The motor rotates and moves to 3600.0 deg.
7. In the System Manager tree, click [Position actual value].
8. Open the [Online] tab and confirm that the value is 36000.

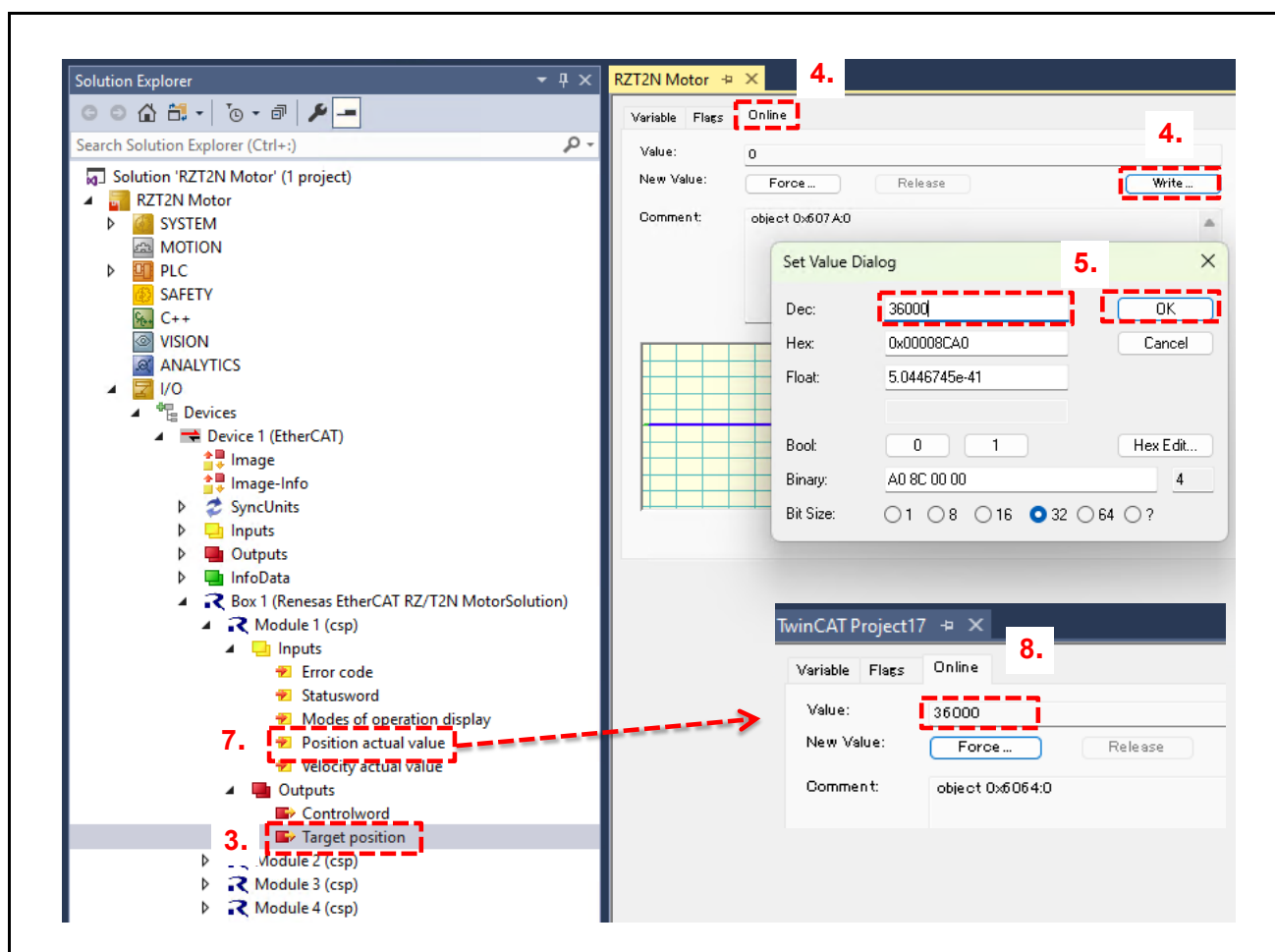


Figure 6.25. CSP Mode Operation Check

6.3.4.2 PP Mode (Profile Position)

1. Follow the steps in Section 6.3.1 to set the motor's CiA402 operating mode of the motor you want to operate to pp mode.
2. Follow the steps in Section 6.3.2 to transition the CiA402 state machine to Operation Enable.
3. In the System Manager tree, click [Profile velocity].
4. Open the [Online] tab and click [Write].
5. Enter 30000 to set the velocity profile to 3000.0 rpm.
6. In the System Manager tree, click [Profile acceleration].
7. Open the [Online] tab, click [Write].
8. Enter 150000 to set the acceleration profile to 15000.0 rpm/s.

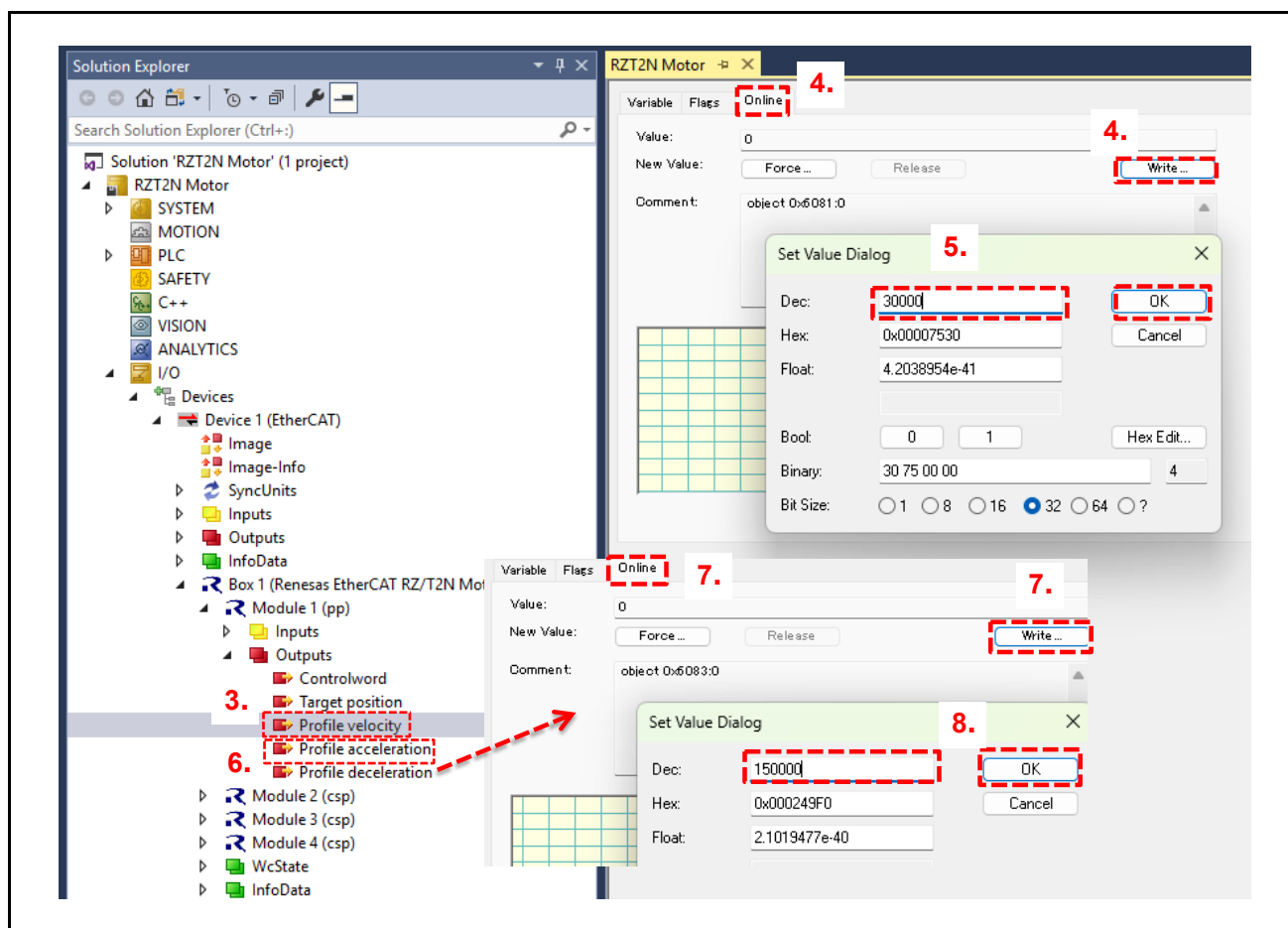


Figure 6.26. PP Mode Operation Check – Part 1

9. In the System Manager tree, click [Target position].
10. Open the [Online] tab, click [Write].
11. Enter 36000 to issue a position command for the motor to move to 3600.0°.
12. The motor rotates according to the set profile and moves to 3600.0°.
13. In the System Manager tree, click [Position actual value].
14. Open the [Online] tab and confirm that the value is 36000.

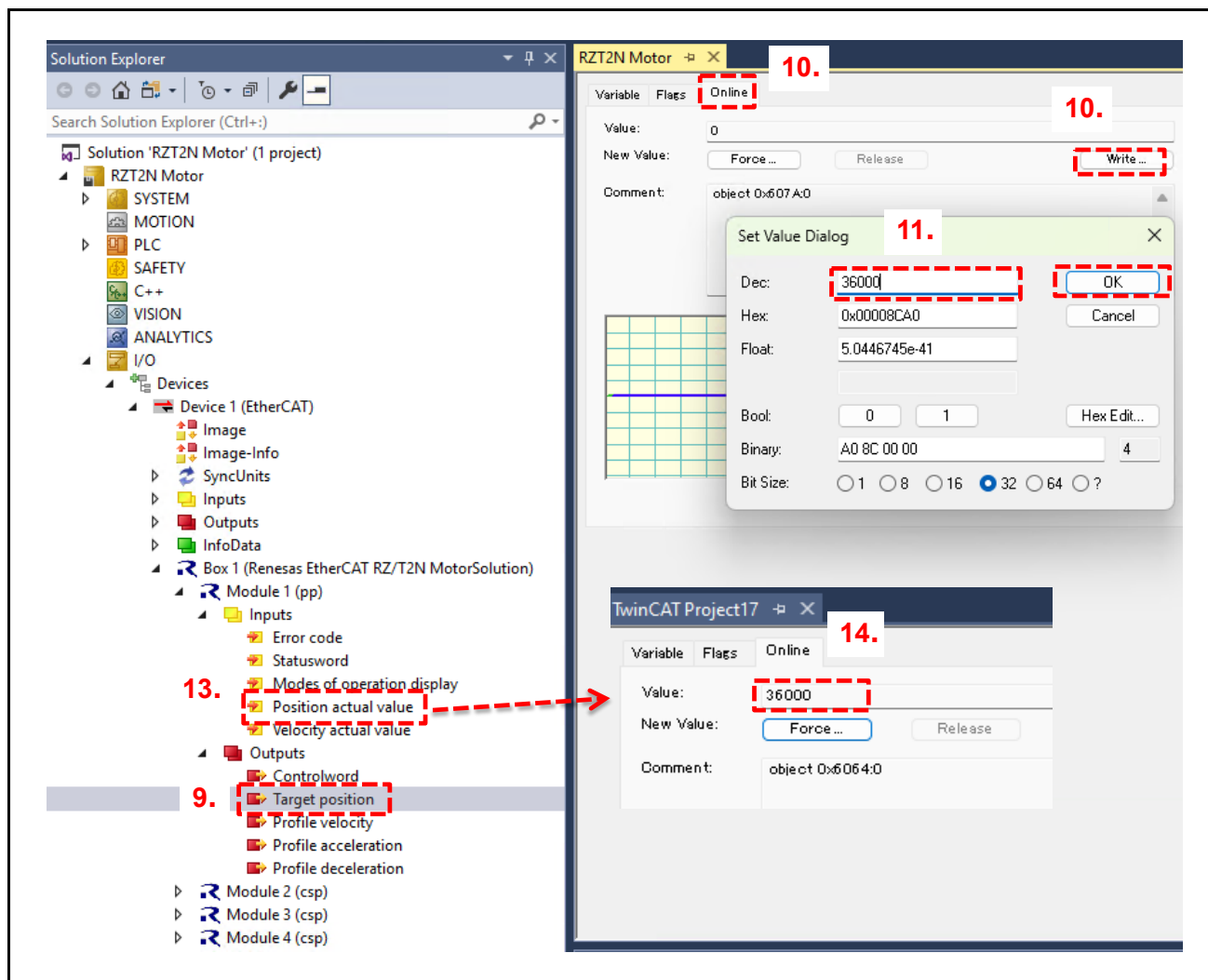


Figure 6.27. PP Mode Operation Check – Part 2

6.3.4.3 CSV Mode (Cyclic Synchronous Velocity)

1. Follow the steps in Section 6.3.1 to set the motor's CiA402 operating mode of the motor you want to operate to CSV mode.
2. Follow the steps in Section 6.3.2 to transition the CiA402 state machine to Operation Enable.
3. In the System Manager tree, click [Target velocity].
4. Open the [Online] tab and click [Write].
5. Enter 20000 to issue a speed command for the motor to rotate at 2000 rpm.
6. The motor rotates at 2000 rpm.
7. Click [Velocity actual value], open the [Online] tab, and confirm that the value is around 20000.

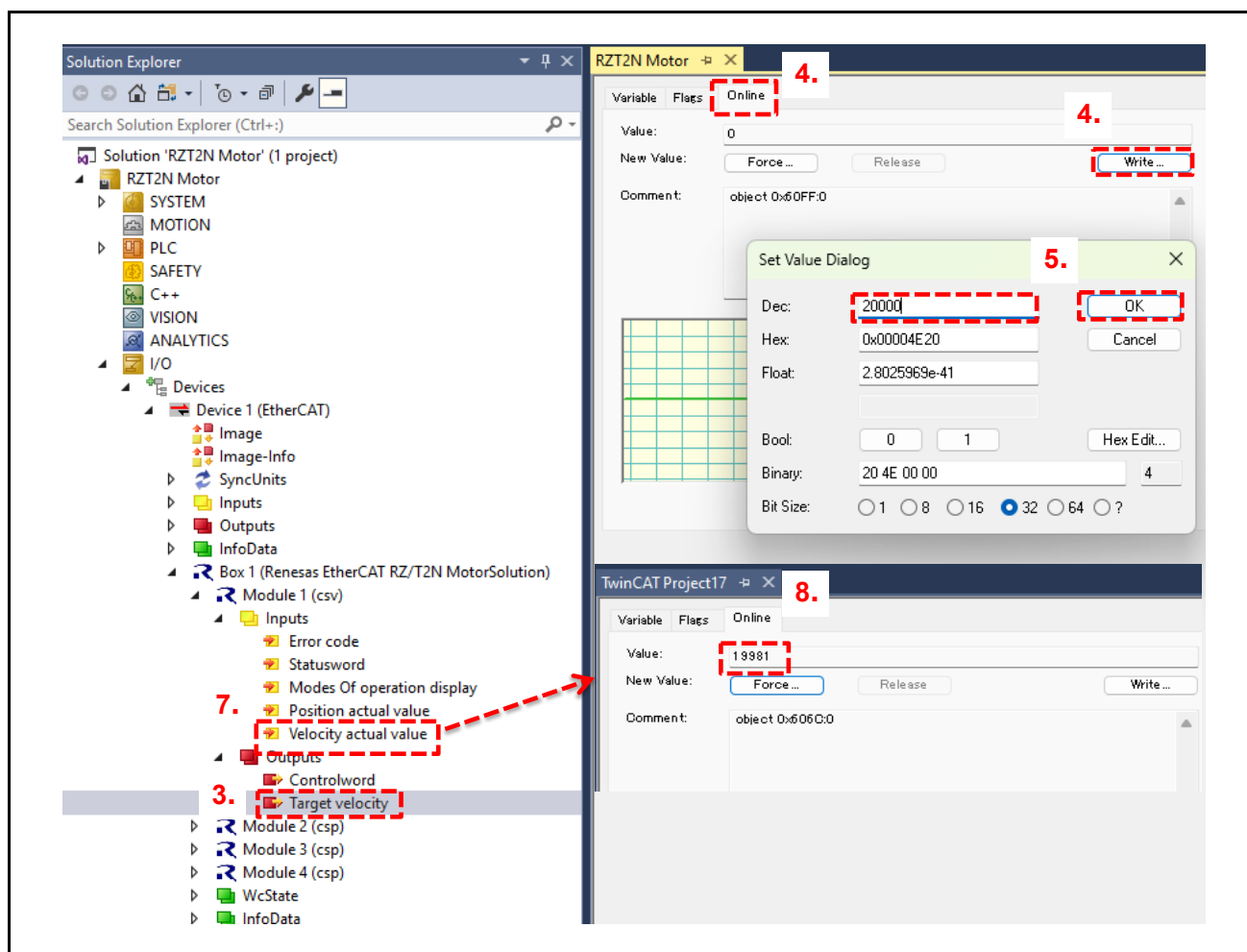


Figure 6.28. CSV Mode Operation Check

6.3.4.4 HM Mode (Homing)

1. Follow the steps in Section 6.3.1 to set the motor's CiA402 operating mode of the motor you want to operate to HM mode.
2. Follow the steps in Section 6.3.2 to transition the CiA402 state machine to Operation Enable.
3. In the System Manager tree, click [Position actual value].
4. Open the [Online] tab and check the current value.
5. In the System Manager tree, click [Home offset].
6. Confirm that it is 0.

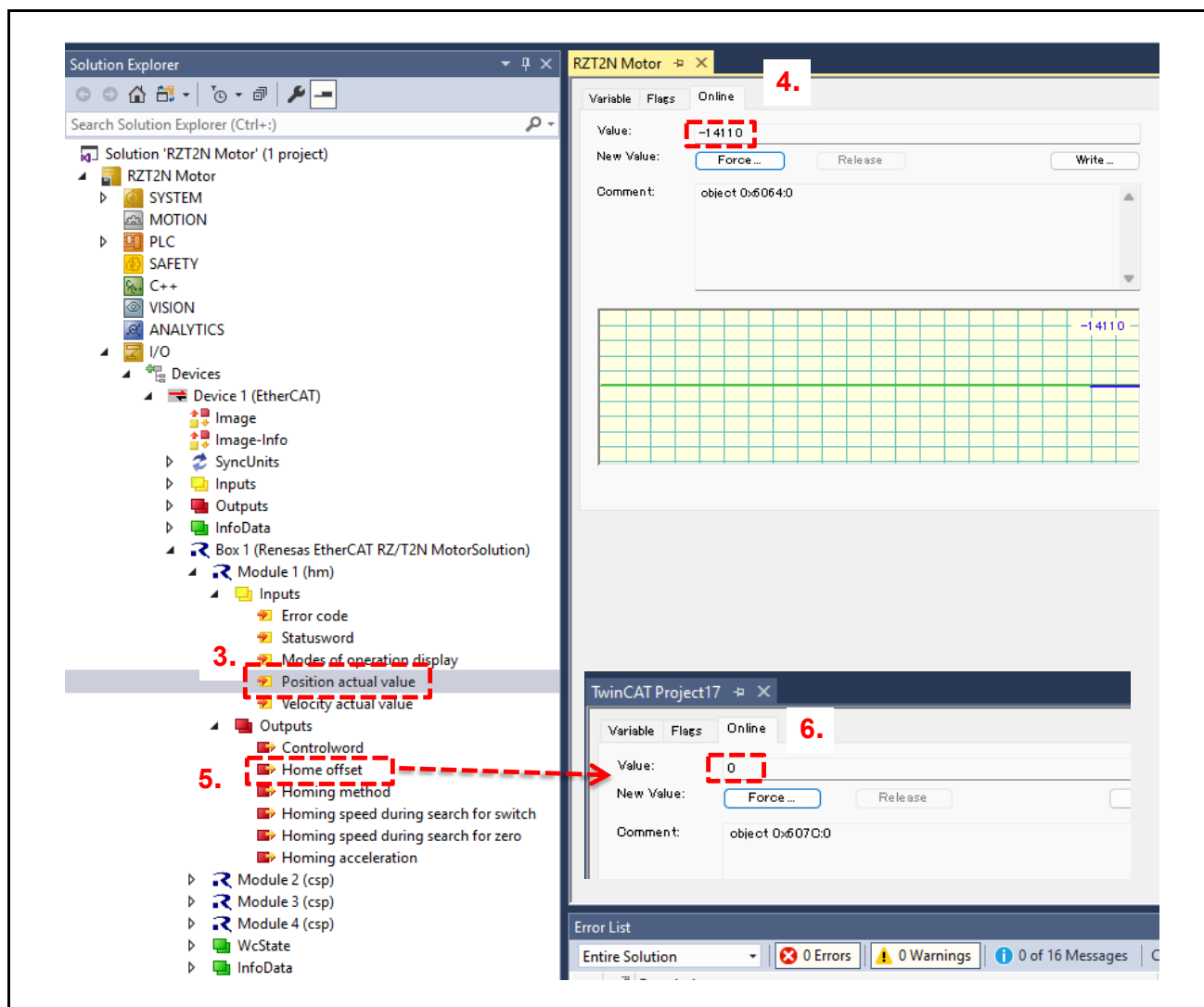


Figure 6.29. HM Mode Operation Check – Part 1

7. In the System Manager tree, click [Home method].
8. Open the [Online] tab and click [Write].
9. Enter 37 to specify Home Method 37.

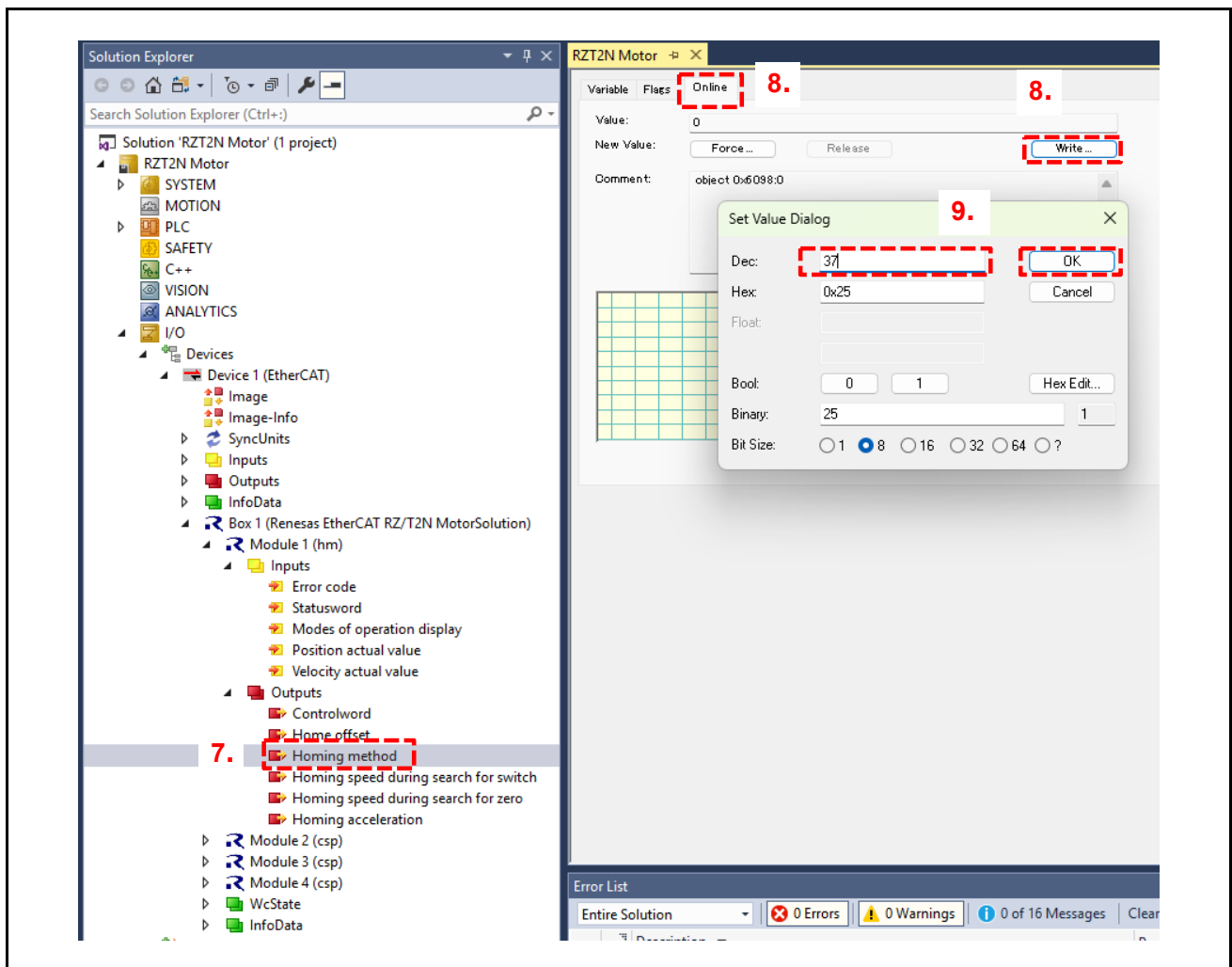


Figure 6.30. HM Mode Operation Check – Part 2

10. In the System Manager tree, click [Controlword].
11. Open the [Online] tab and click [Write].
12. Enter 31 to execute Home Method 37.
13. In the System Manager tree, click [Position actual value].
14. Open the [Online] tab and confirm that the value has changed to 0.

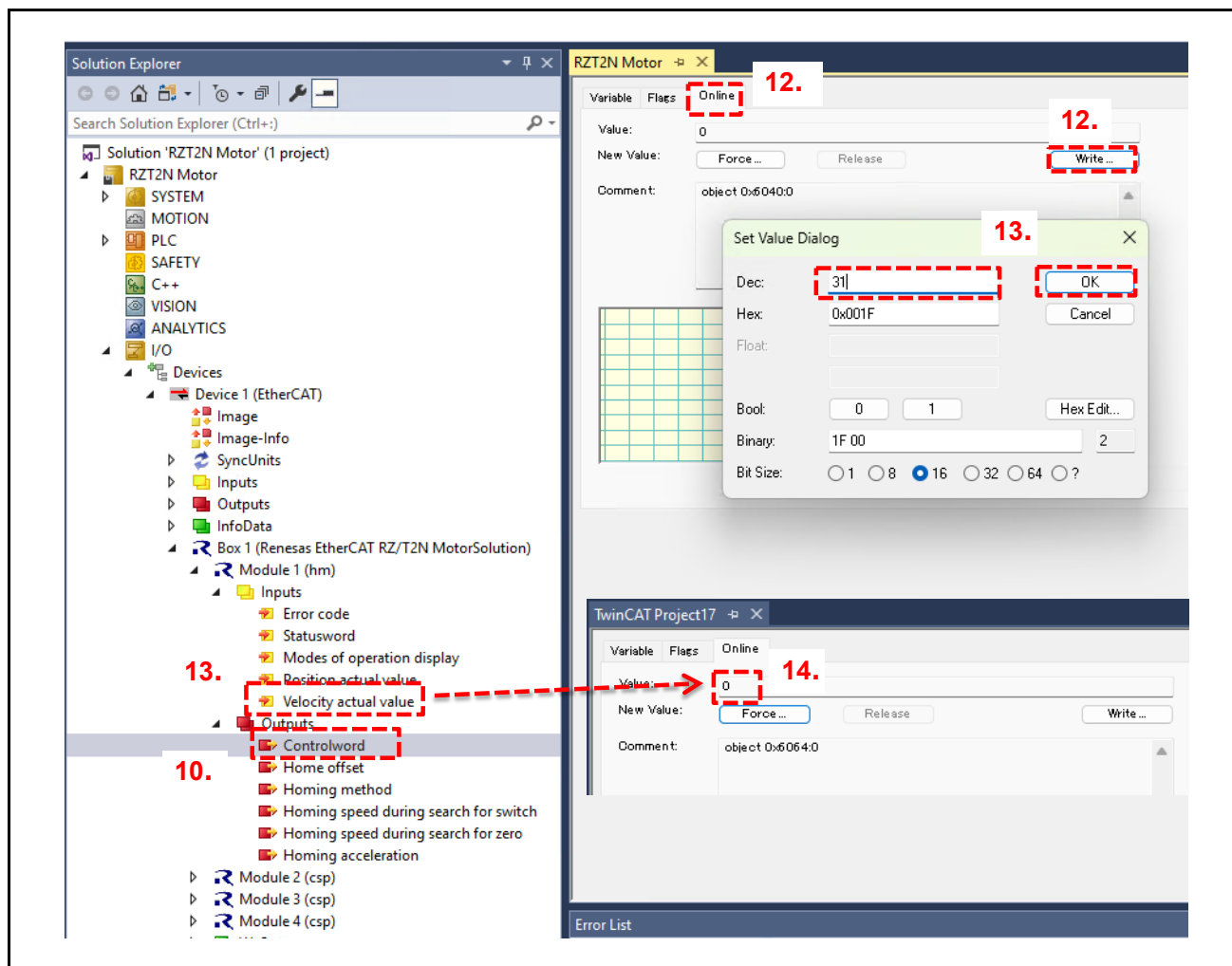


Figure 6.31. HM Mode Operation Check – Part 3

7. Software Specification

7.1 Software Configuration

Figure 7.1 shows software configuration for this sample program.

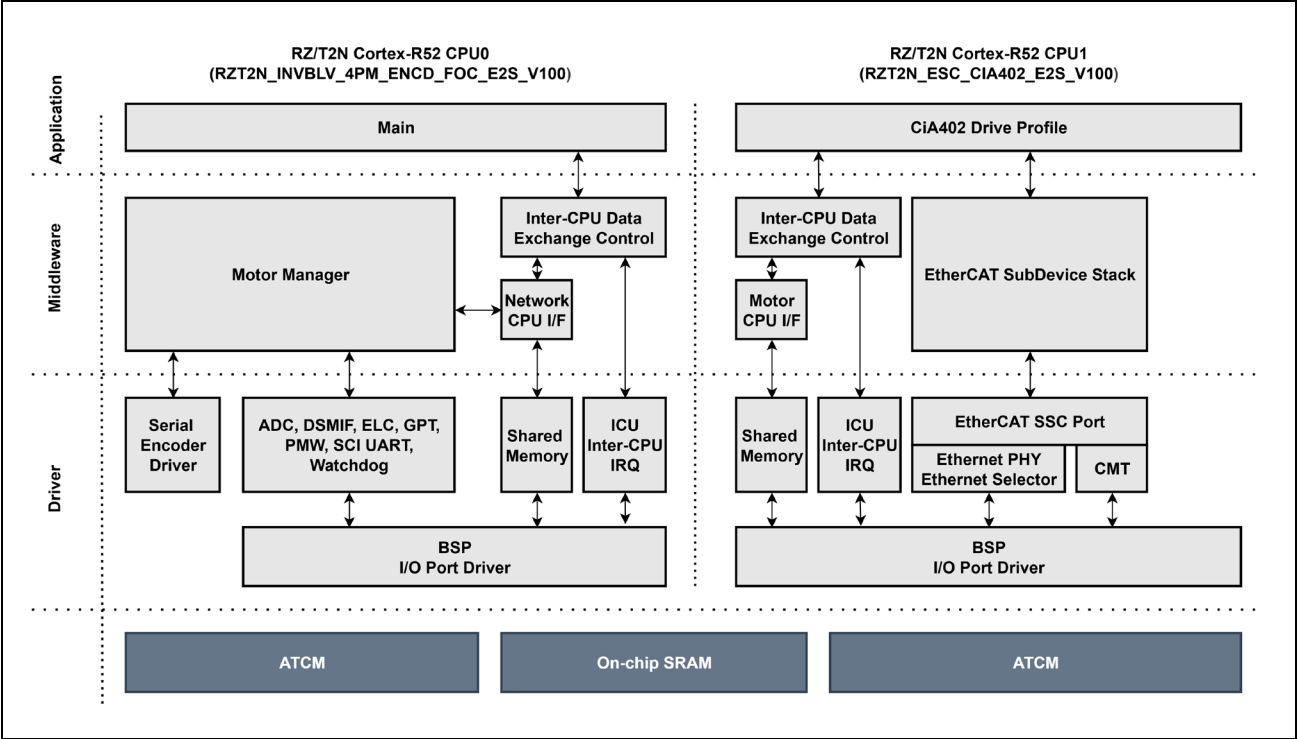


Figure 7.1. Software Configuration for this sample program

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

7.2.1 Overall Software Structure

The CPU0 software of the RZ/T2N in this sample program is based on “RZ/T2N Encoder Vector Control of Permanent Magnet Synchronous Motor (Serial Encoder) – 4 motor (R01AN8225)”.

For details of the base project, refer to the application note R01AN8225.

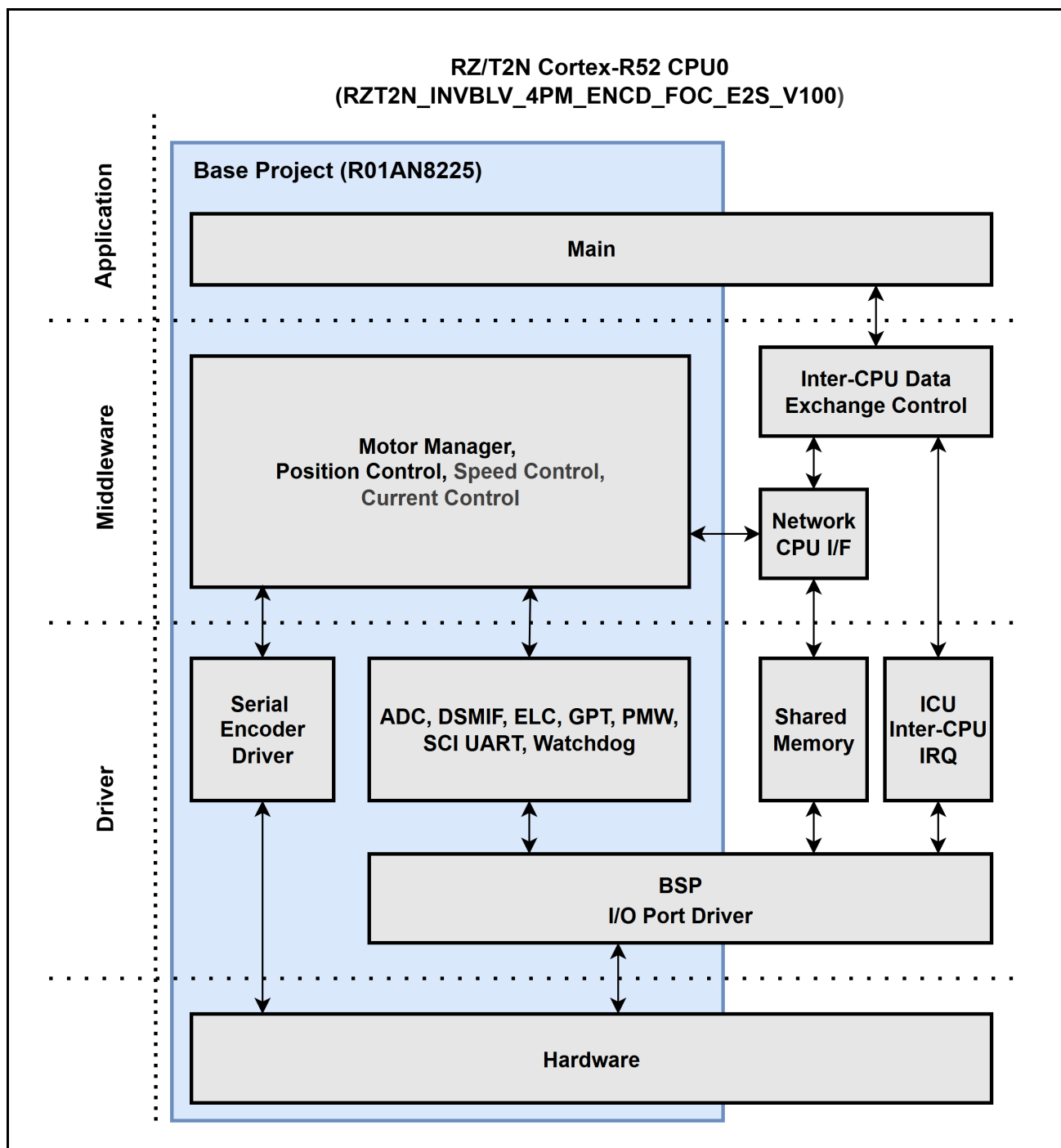


Figure 7.2. Overall Software Structure of RZ/T2N Cortex-R52 CPU0

7.2.2 Changes from the Base Project

This section describes the changes made from the base project.

Table 7.1. Changes from Base Project

Folder / File Changed			Description of Change
rz			Files and folders modified or added due to changes in Configuration.xml
rz_gen			
rz_cfg			
src	network_cpu_if	r_network_cpu_if.c	Added Network CPU I/F module (see Section 7.2.3)
		r_network_cpu_if.h	
	cpu0_exchange_ctrl	cpu0_exchange_ctrl.c	Added CPU0 Inter-CPU Data Exchange Control module (see Section 7.2.4)
		cpu0_exchange_ctrl.h	
	hal_entry.c		Modified processing inside hal_entry function Added calls to CPU0_ExchangeCtrl_Init() and CPU0_ExchangeCtrl_Main(). (see Figure Figure 7.3)
configuration.xml			Added Shared Memory Driver. (settings in Table 7.2, Table 7.3, Table 7.4) Added inter-CPU data exchange control interrupt settings. (settings in Table 7.5, Table 7.6) Set CPU1 clock to 800 MHz.
.cproject			Added include path for “src/network_cpu_if” . Added include path for “src/cpu0_exchange_ctrl”. Added RZ/T2N CPU1 object file.

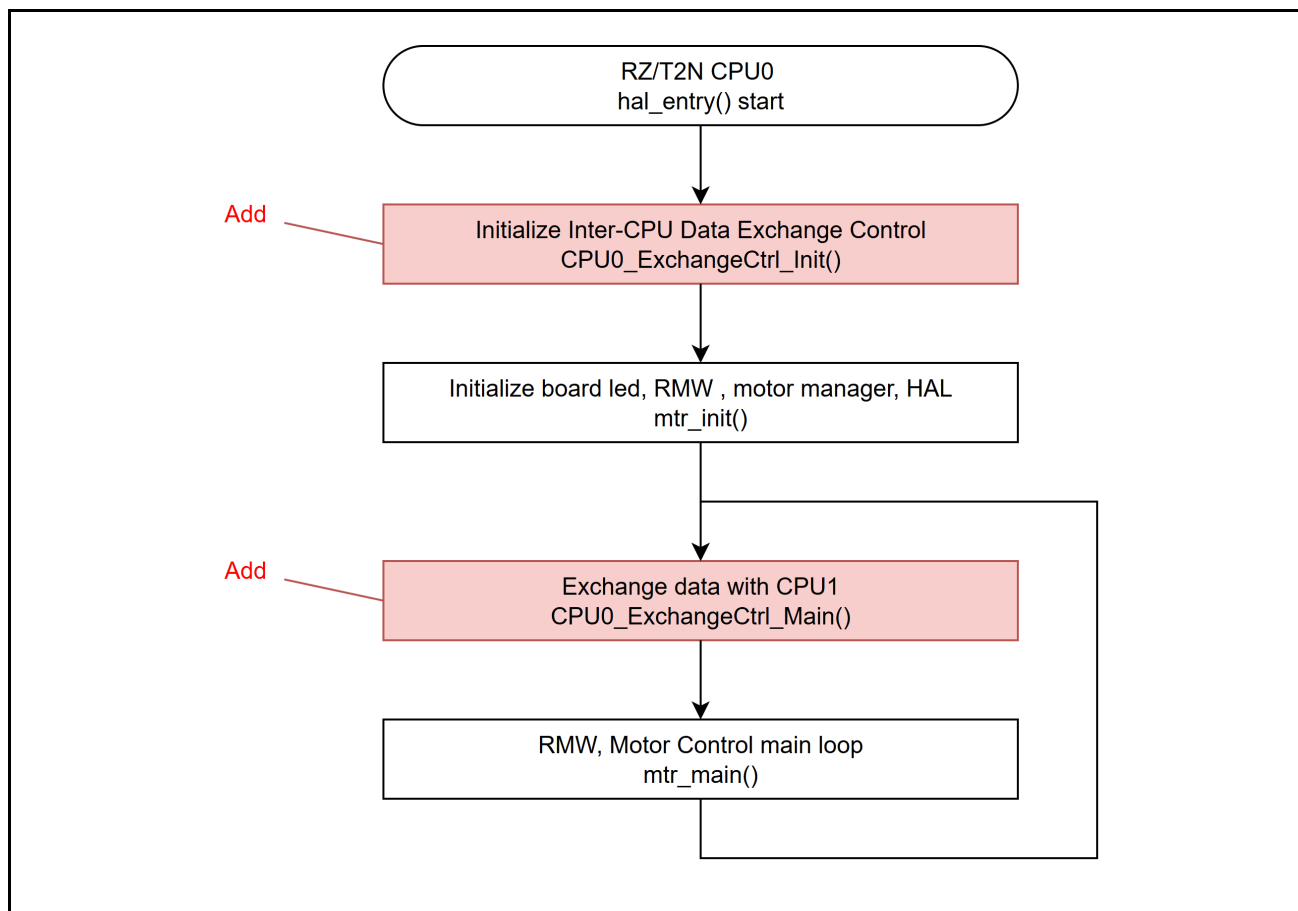
**Figure 7.3. Flowchart of hal_entry Function**

Table 7.2. Added Shared Memory Driver N Settings (r_shared_memory, N=0,1,2,3)

Property	Setting
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

Table 7.3. Added Shared Memory Driver N Settings (r_icu_inter_cpu_irq, tx, X=2N)

Property	Setting
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

Table 7.4. Added Shared Memory Driver N Settings (r_icu_inter_cpu_irq, rx, Y=2N+1)

Property	Setting
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

Table 7.5. Added Inter-CPU Data Exchange Control Settings (r_icu_inter_cpu_irq, ch8)

Property	Setting
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

Table 7.6. Added Inter-CPU Data Exchange Control Settings (r_icu_inter_cpu_irq, ch9)

Property	Setting
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 Module

The Network CPU I/F module uses the Shared Memory Driver to exchange motor control commands, reference values, and motor status between the Network CPU (RZ/T2N CPU1) and the motor control system.

It also calls the Motor Manager API to execute control operations.

This sample program supports 4-axis motors by using four instances of this module.

7.2.3.1 Module Structure

The module structure diagram is shown in Figure 7.4.

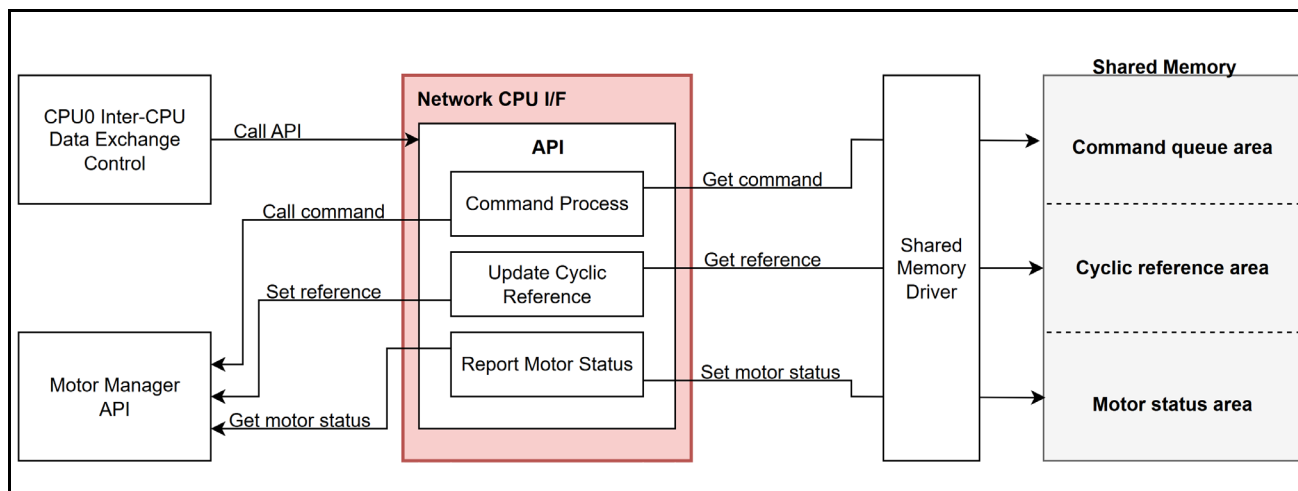


Figure 7.4. Network CPU I/F Module Structure

This module mainly provides three types of APIs:

(1) Command Process

Checks whether motor control commands are stored in the Command Queue Area of Shared Memory.

If commands exist, the corresponding Motor Manager API is called to execute the process.

(2) Update Cyclic Reference

Checks whether various reference values in the Cyclic Reference Area of Shared Memory have been updated.

If updated by the Network CPU, the corresponding Motor Manager API is called to set the new reference values.

(3) Report Motor Status

Calls the Motor Manager API to obtain various motor statuses and checks for updates.

If updated, reflects the status in the Motor Status Area of Shared Memory.

7.2.3.2 Shared Memory Allocation

The base address of each module (N = 0, 1, 2, 3) is 0x101E_0000 + 0x200N.

Table 7.7. Shared Memory Allocation (Command queue area)

Offset address	Name	Size [bytes]	Usage
0x0	Command queue n (n = 0 ... 7)	16 × 8 (=128)	Command queue
0x80	head	4	Queue head
0x84	tail	4	Queue tail

Table 7.8. Shared Memory Allocation (Cyclic reference area)

Offset address	Name	Size [bytes]	Usage
0x100	f4_ref_position	4	Reference position
0x104	f4_ref_speed	4	Reference speed
0x108	f4_ref_torque	4	Reference torque

Table 7.9. Shared Memory Allocation (Motor status area)

Offset address	Name	Size [bytes]	Usage
0x180	f4_pos_deg	4	Current position
0x184	f4_speed	4	Current speed
0x188	u1_status	1	State machine: 0x00: STOP 0x01: RUN 0x02: ERROR
0x189	reserved1	3	Unused
0x18C	u2_error_status	2	Error status: 0x0000: No error 0x0001: HW overcurrent error 0x0002: Overvoltage error 0x0004: Overspeed error 0x0080: Undervoltage error 0x0100: SW overcurrent error 0x0200: Inverter overheat error 0xFFFF: Unknown error
0x18E	reserved2	2	Unused
0x190	u1_ctrl_loop_mode	1	LOOP MODE: 0: d-axis current control 1: q-axis current control 2: Speed control 3: Position control
0x191	reserved3	3	Unused
0x194	u1_in_position	1	Position control completion: 0: Not completed 1: Completed
0x195	reserved4	3	Unused

0x198	u1_offset_cal_finished_flag	1	Offset measurement status: 0: Measuring 1: Completed
0x199	reserved5	3	Unused

7.2.3.3 API

Table 7.10. Network CPU I/F API List

API	Description
R_NETWORK_CPU_IF_Open	Initializes the Network CPU I/F and the Shared Memory Driver. Flowchart: Figure 7.5
R_NETWORK_CPU_IF_Close	Terminates the Network CPU I/F and the Shared Memory Driver. Flowchart: Figure 7.6
R_NETWORK_CPU_IF_CommandProcess	Checks if motor control commands are stored in the Command Queue Area of Shared Memory. If commands exist, calls the corresponding Motor Manager API to execute the process. Flowchart: Figure 7.7
R_NETWORK_CPU_IF_UpdateCyclicReference	Checks if reference values in the Cyclic Reference Area of Shared Memory have been updated. If updated by the Network CPU, calls the Motor Manager API to set new reference values. Flowchart: Figure 7.8
R_NETWORK_CPU_IF_ReportMotorStatus	Calls the Motor Manager API to obtain motor status and checks for updates. If updated, reflects the status in the Motor Status Area of Shared Memory. Flowchart: Figure 7.9

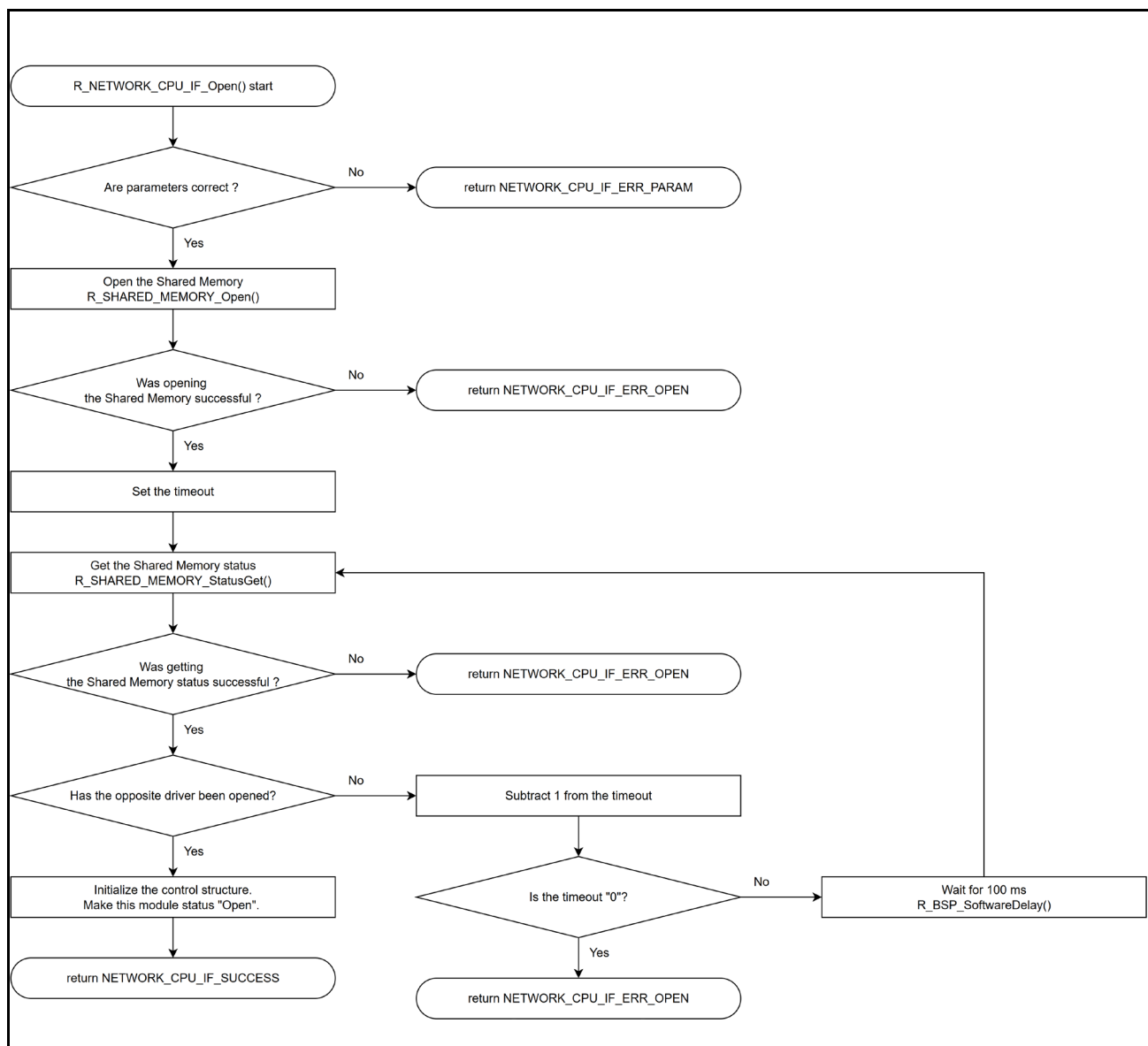


Figure 7.5. R_NETWORK_CPU_IF_Open function flowchart

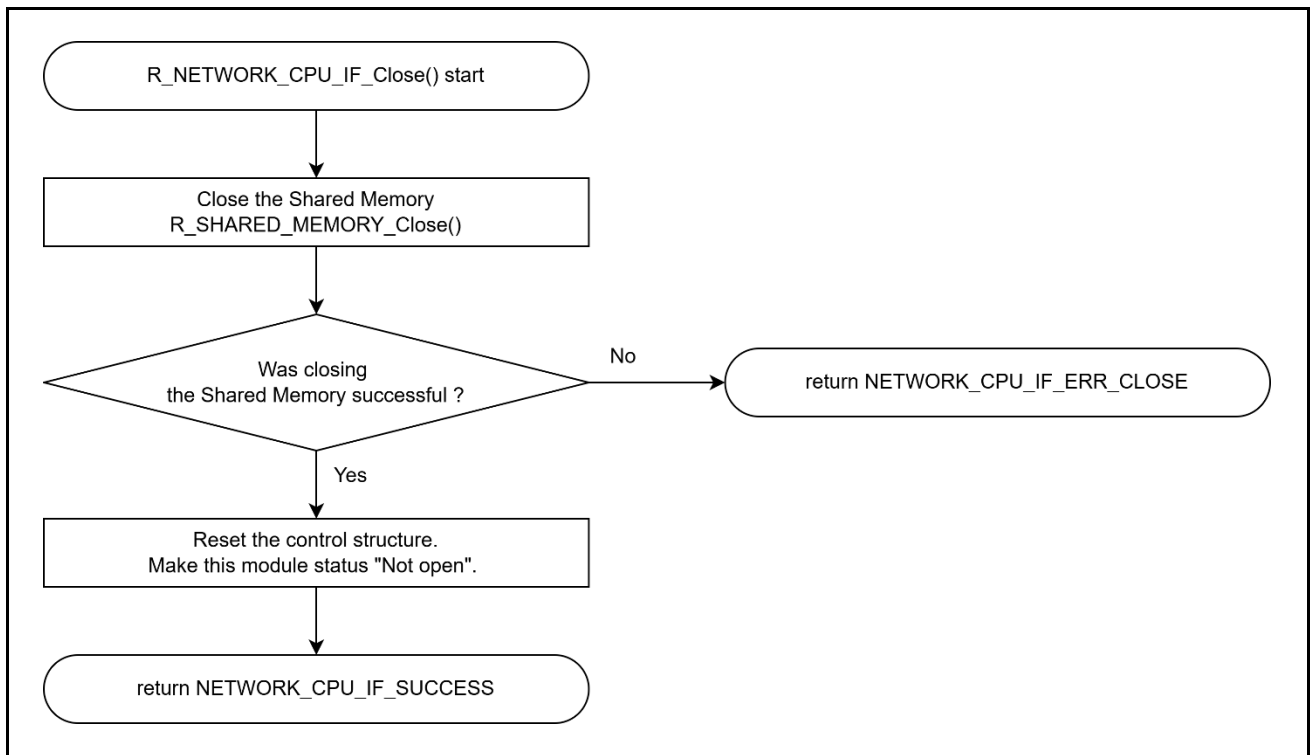


Figure 7.6. R_NETWORK_CPU_IF_Close function flowchart

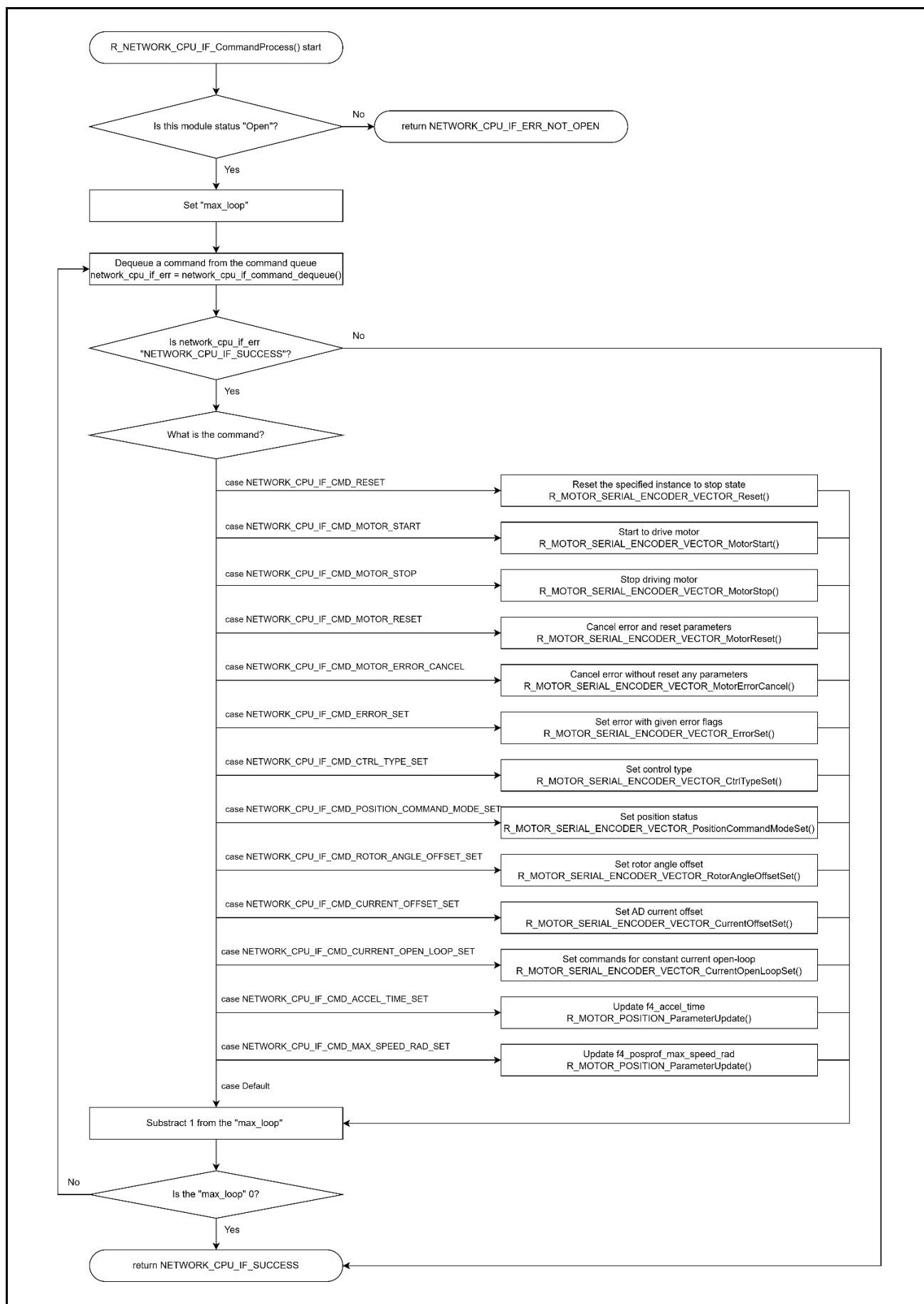


Figure 7.7. R_NETWORK_CPU_IF_CommandProcess function flowchart

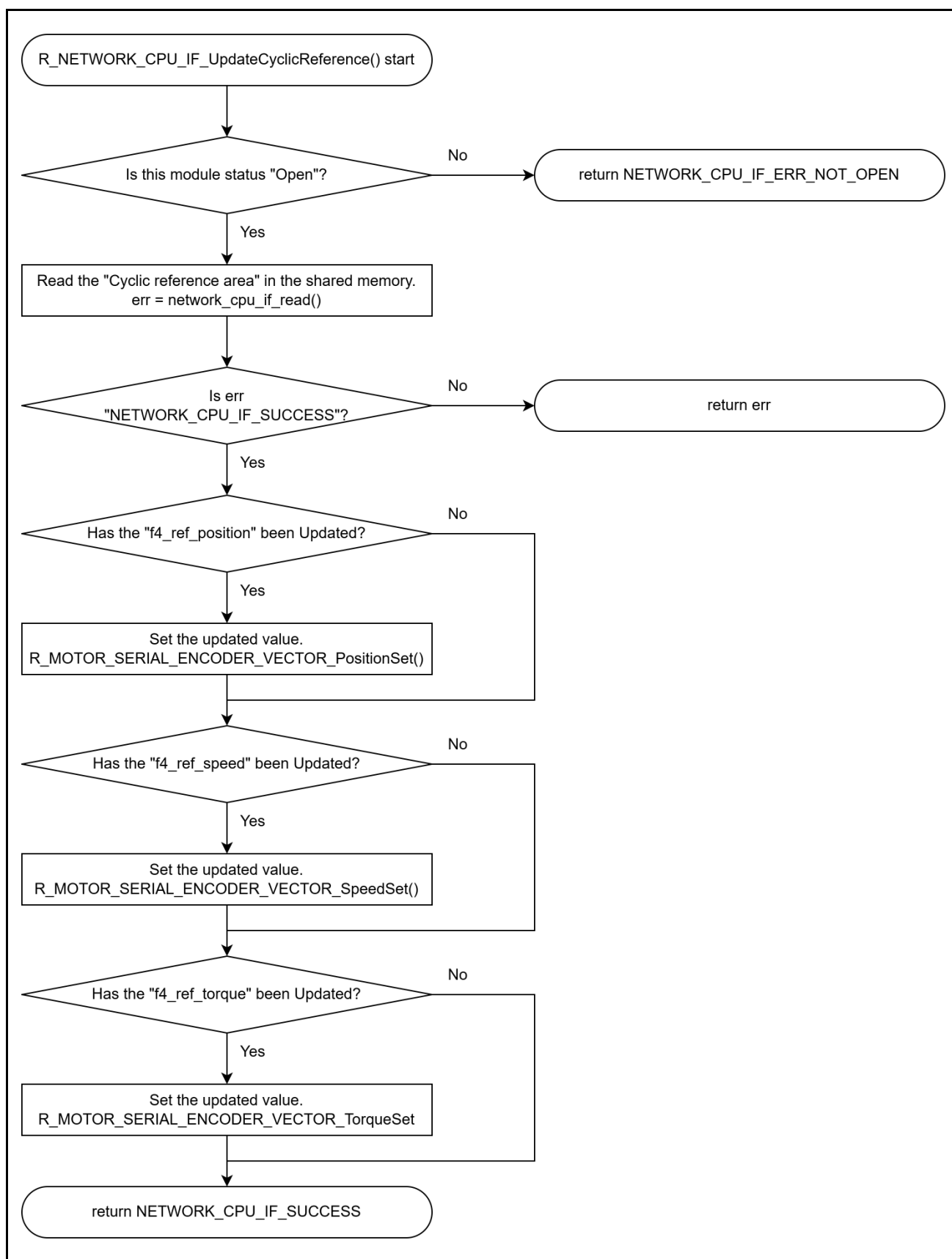


Figure 7.8. R_NETWORK_CPU_IF_UpdateCyclicReference function flowchart

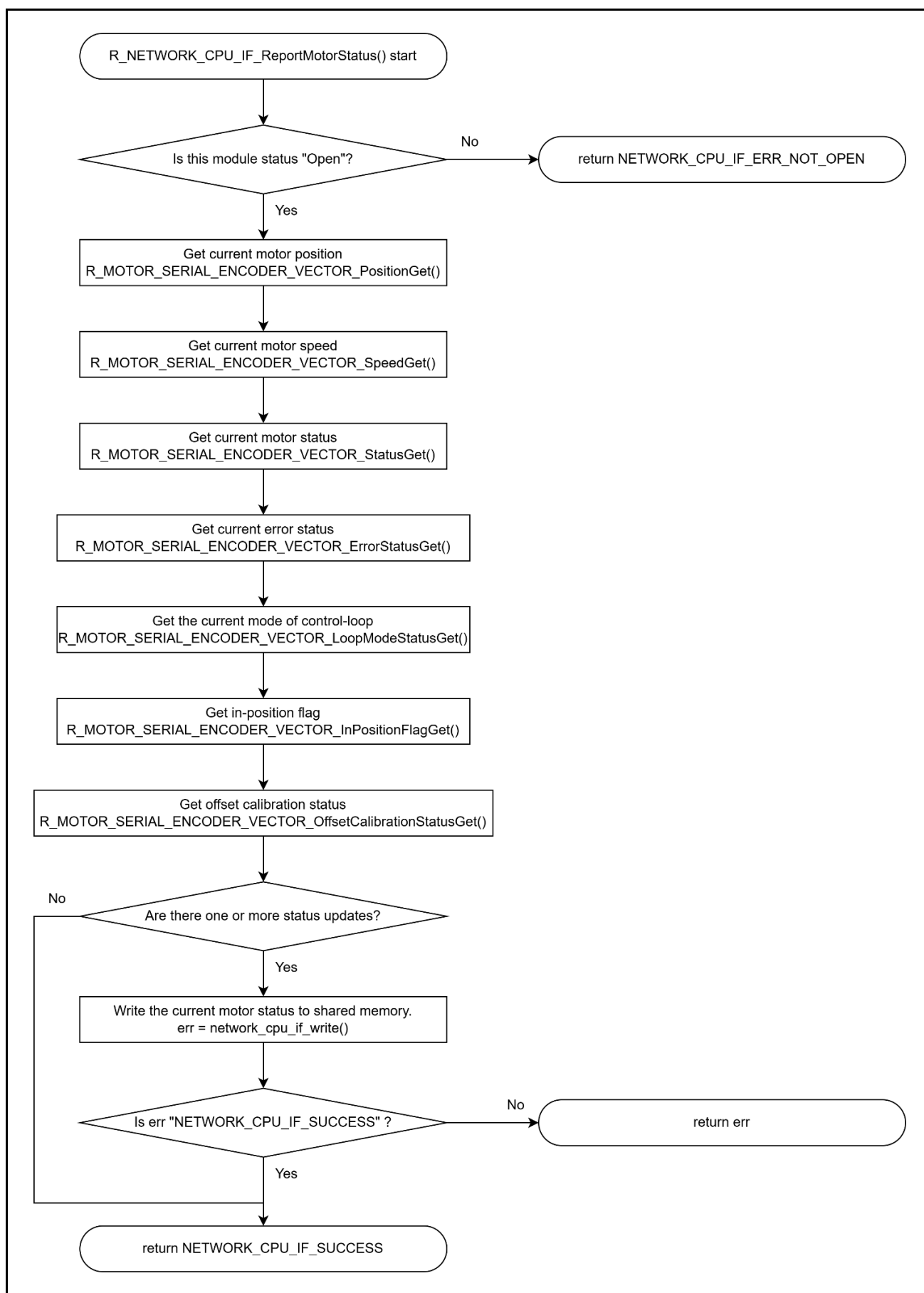


Figure 7.9. R_NETWORK_CPU_IF_ReportMotorStatus function flowchart

7.2.3.4 Private Functions

Table 7.11. Network CPU I/F Private Functions List

Private Functions	Description
network_cpu_if_write	Writes data to Shared Memory. Flowchart: Figure 7.10
network_cpu_if_read	Reads data from Shared Memory. Flowchart: Figure 7.11
network_cpu_if_command_dequeue	Reads commands from the Command Queue. Flowchart: Figure 7.12

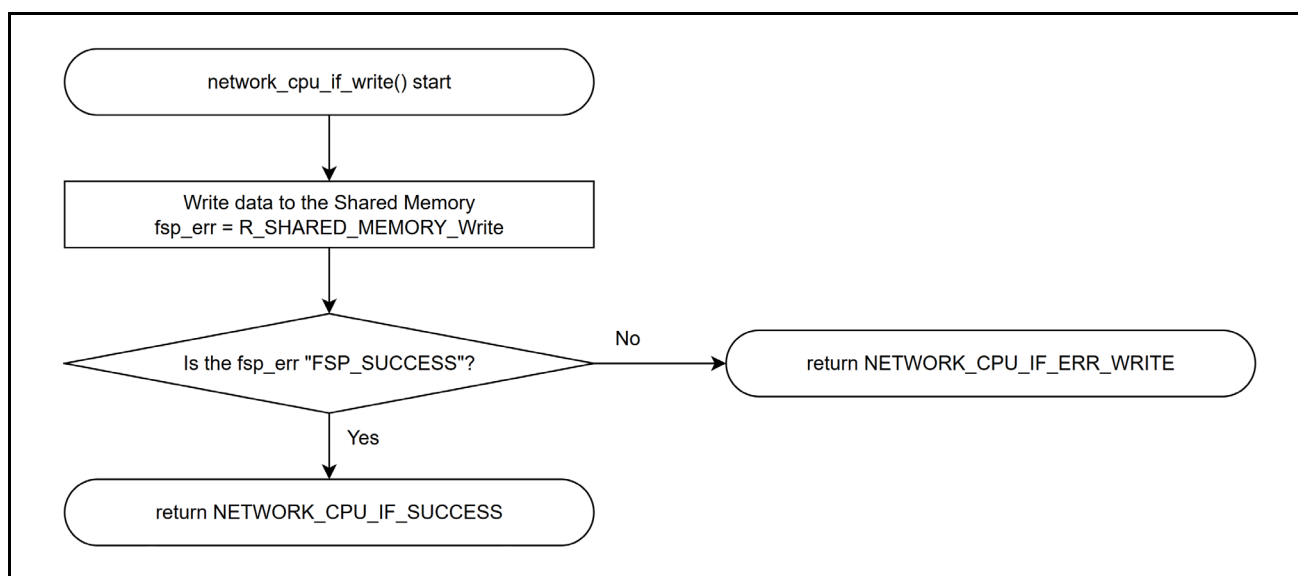


Figure 7.10. network_cpu_if_write function flowchart

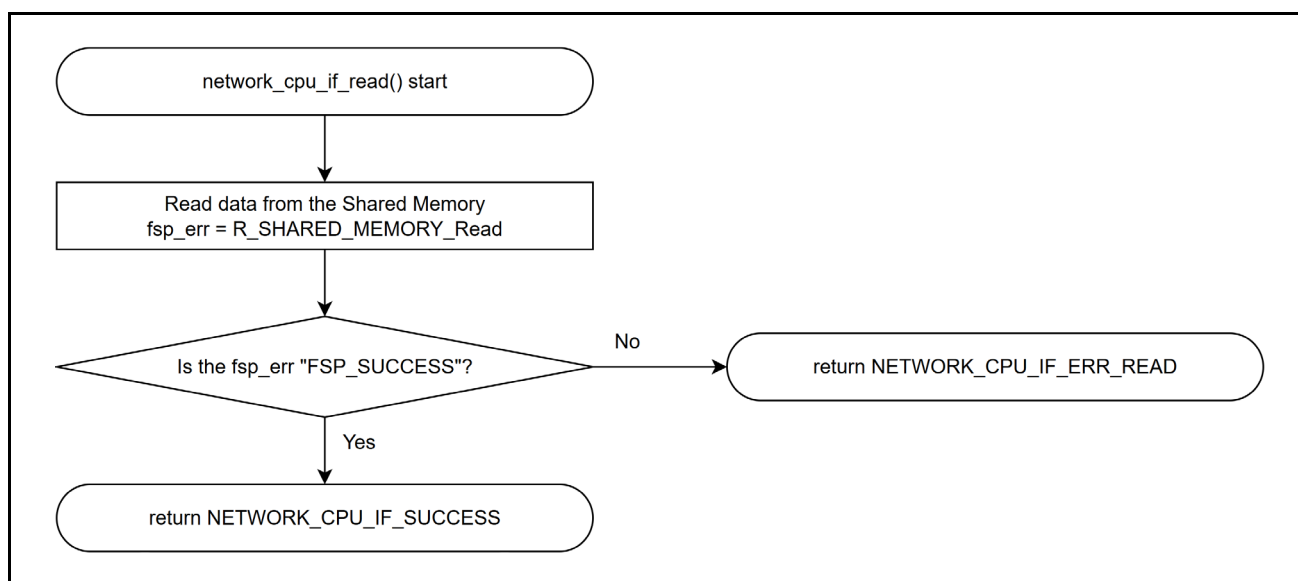
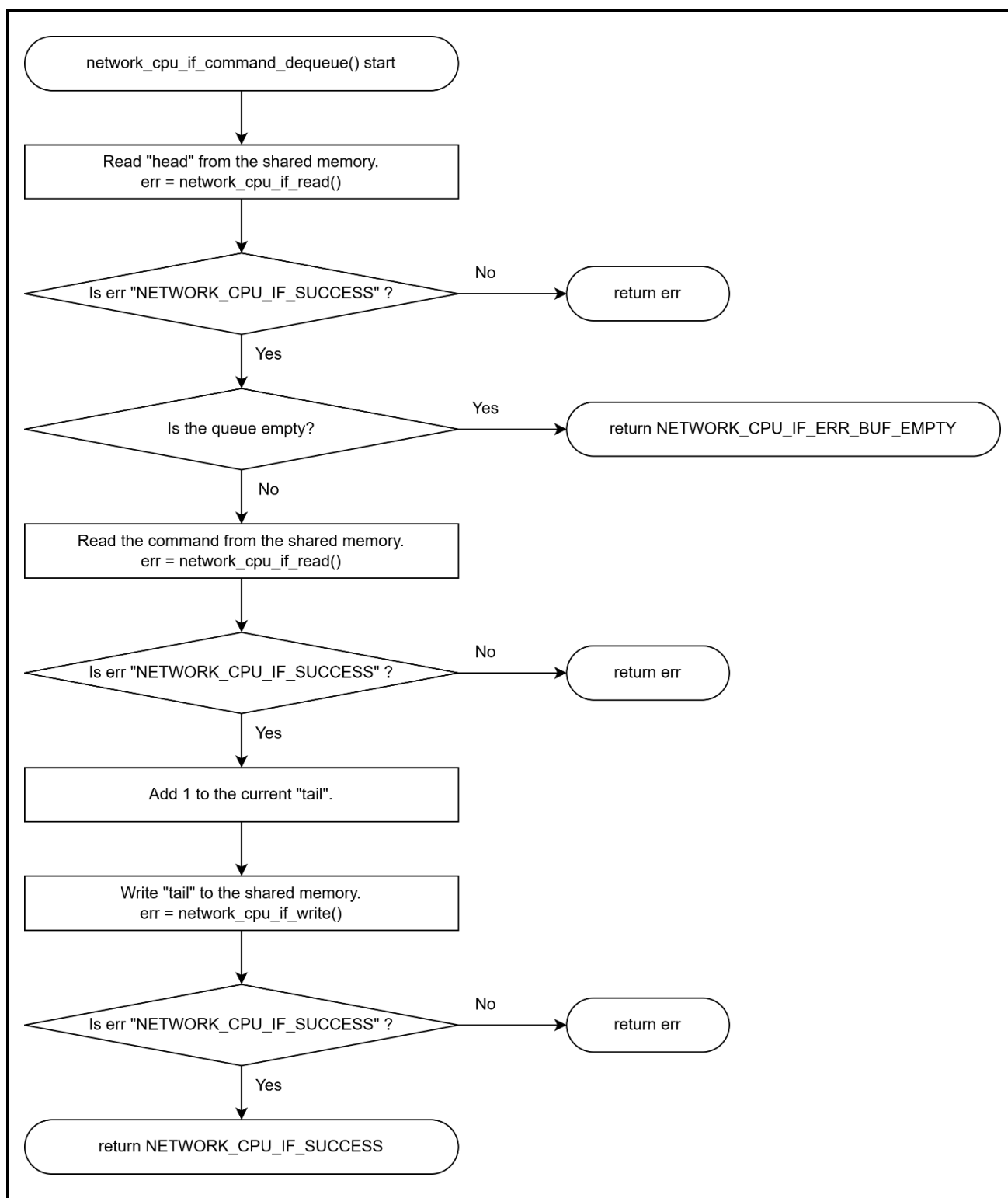


Figure 7.11. network_cpu_if_read function flowchart

**Figure 7.12. network_cpu_if_command_dequeue function flowchart**

7.2.3.5 Callback Function

Table 7.12. Network CPU I/F Callback Functions List (r_network_cpu_if.c)

Private Functions	Description
shared_memory_callback	Callback function for the Shared Memory Driver. This interrupt occurs when RZ/T2N CPU1 writes to Shared Memory. No processing is performed inside the callback function. Consider adding processing if necessary.

7.2.3.6 Macro Definitions

Table 7.13. Network CPU I/F Macro Definitions List (r_network_cpu_if.h)

Macro Name	Value	Remarks
CFG_CMD_Q_NUM	8	Number of command queues (must be a power of 2)
CFG_CMD_Q_OFFSET	0x0000	Offset address of Command Queue Area
CFG_CYCLIC_REF_AREA_OFFSET	0x0100	Offset address of Cyclic Reference Area
CFG_MTR_STATUS_AREA_OFFSET	0x0180	Offset address of Motor Status Area
CFG_NETWORK_CPU_IF_TIMEOUT	100	Number of timeout counts while waiting for Shared memory to be ready at Open
CFG_SHARED_MEMORY_INSTANCE_ADDR0	&g_shared_memory0	Pointer to the Shared memory driver 0–3 instances
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_vector0	Pointer to motor control instances 0–3
CFG_MOTOR_CTRL_INSTANCE_ADDR1	&g_st_serial_encoder_vector1	
CFG_MOTOR_CTRL_INSTANCE_ADDR2	&g_st_serial_encoder_vector2	
CFG_MOTOR_CTRL_INSTANCE_ADDR3	&g_st_serial_encoder_vector3	
NETWORK_CPU_IF_CMD_DATA_MAX_BYTE	12	Maximum data size for command queue

7.2.3.7 Enumerations

Table 7.14. Network CPU I/F Enumerations List (r_network_cpu_if.h)

Enumerations	Definitions	Remarks
network_cpu_if_err_t Network CPU I/F Error Codes	NETWORK_CPU_IF_SUCCESS	No error
	NETWORK_CPU_IF_ERR_OPEN	Open error
	NETWORK_CPU_IF_ERR_CLOSE	Close error
	NETWORK_CPU_IF_ERR_PARAM	Parameter error
	NETWORK_CPU_IF_ERR_NOT_OPEN	Not open error
	NETWORK_CPU_IF_ERR_BUF_FULL	Command queue full
	NETWORK_CPU_IF_ERR_BUF_EMPTY	Command queue empty
	NETWORK_CPU_IF_ERR_WRITE	Write error
	NETWORK_CPU_IF_ERR_READ	Read error
	NETWORK_CPU_IF_ERR_EXCEED_PAYLOAD	Data exceeded

Enumerations	Definitions	Remarks
network_cpu_if_status_t Network CPU I/F Status	NETWORK_CPU_IF_STATUS_NOT_OPEN	Not open
	NETWORK_CPU_IF_STATUS_OPEN	Open
network_cpu_if_cmd_t Network CPU I/F Command Definitions	NETWORK_CPU_IF_CMD_NONE	No command
	NETWORK_CPU_IF_CMD_RESET	Commands used in CommandProcess API (See Figure 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 Structure and Variable Information

Table 7.15. Network CPU I/F Structure Definitions (r_network_cpu_if.h)

Structure	Variable	Description
st_current_offset_cfg_t Current offset configuration structure	f4_iu_offset	U-phase current offset
	f4_iv_offset	V-phase current offset
	f4_iw_offset	W-phase current offset
st_current_open_loop_cfg_t Open loop current configuration structure	f4_open_loop_current	Open loop current value
	f4_ref_speed_rpm	Reference speed (RPM)
network_cpu_if_cmd_hdr_t command queue head/tail structure	head	Head of command queue
	tail	Tail of command queue
network_cpu_if_cmd_q_t command queue structure	cmd	Command
	size	Data size
	data[NETWORK_CPU_IF_CMD_DATA_MAX_BYTE]	Data array
network_cpu_if_cyclic_ref_t Cyclic Reference structure	f4_ref_position	Reference position
	f4_ref_speed	Reference speed
	f4_ref_torque	Reference torque
network_cpu_if_mtr_status_t Motor Status structure	f4_pos_deg	Current position
	f4_speed	Current speed
	u1_status	State machine
	reserved1	Unused area
	u2_error_status	Error status
	reserved2	Unused area
	u1_ctrl_loop_mode	LOOP MODE
	reserved3	Unused area
	u1_in_position	Position control completion status
	reserved4	Unused area
	u1_offset_cal_finished_flag	Offset measurement status
	reserved5	Unused area
network_cpu_if_cfg_t Network CPU I/F Configuration structure	p_shared_memory_instance	Pointer to Shared Memory instance
	timeout	Number of timeout counts while waiting for Shared memory to be ready at Open
	shr_mem_cmd_data_max_size	Maximum command data size
	shr_mem_cmd_q_num	Number of command queues
	shr_mem_cmd_q_mask	Command queue mask
	shr_mem_cmd_q_offset	Offset of Command Queue Area in Shared Memory

	shr_mem_cmd_q_size	Size of Command Queue Area in Shared Memory
	shr_mem_cmd_head_offset	Offset of command queue head area
	shr_mem_cmd_head_size	Size of command queue head area
	shr_mem_cmd_tail_offset	Offset of command queue tail area
	shr_mem_cmd_tail_size	Size of command queue tail area
	shr_mem_cyclic_ref_area_offset	Offset of Cyclic Reference Area
	shr_mem_cyclic_ref_area_size	Size of Cyclic Reference Area
	shr_mem_ref_pos_offset	Offset of reference position area
	shr_mem_ref_vel_offset	Offset of reference speed area
	shr_mem_ref_tor_offset	Offset of reference torque area
	shr_mem_mtr_status_area_offset	Offset of Motor Status Area
	shr_mem_mtr_status_area_size	Size of Motor Status Area
	shr_mem_mtr_pos_offset	Offset of current position area
	shr_mem_mtr_vel_offset	Offset of current speed area
	shr_mem_mtr_status_offset	Offset of state machine area
	shr_mem_mtr_err_status_offset	Offset of error status area
	shr_mem_mtr_ctrl_loop_mode_offset	Offset of LOOP MODE area
	shr_mem_mtr_in_pos_offset	Offset of position control completion area
	shr_mem_mtr_cal_finished_flag_offset	Offset of offset measurement status area
	max_loop	Maximum loop count used by CommandProcess API
	p_st_serial_encoder_vector_control	Pointer to serial encoder vector control structure
network_cpu_if_ctrl_t Network CPU I/F Control structure	status	Network CPU I/F status
	cmd_hdr	Instance of command queue head/tail structure
	cyclic_ref_buf	Buffer for Cyclic Reference area
	mtr_status_buf	Buffer for Motor Status area
	p_cfg	Pointer to Network CPU I/F configuration structure
network_cpu_if_api_t Network CPU I/F API structure	Open	Open function
	Close	Close function
	CommandProcess	Command processing function
	UpdateCyclicReference	Cyclic reference update function
	ReportMotorStatus	Motor status reporting function

7.2.3.9 Global Variable Information

Table 7.16. Network CPU I/F Global Variables (r_network_cpu_if.c)

Type	Instance Name	Description
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	Network CPU I/F control structure instance for 4 motor axes. Initialized by the Open function.
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	Network CPU I/F configuration structure instance for 4 motor axes. Each member is initialized at instance creation. Values cannot be changed after initialization.
network_cpu_if_api_t	g_network_cpu_if_api	Network CPU I/F API structure instance. Each member is initialized at instance creation. Values cannot be changed after initialization.

7.2.4 CPU0 Inter-CPU Data Exchange Control Module

The CPU0 Inter-CPU Data Exchange Control module exchanges data with Shared Memory through the Network CPU I/F only when it receives INTCPU9 from CPU1, and then notifies CPU1 of the completion using INTCPU8.

7.2.4.1 Module Structure

The module structure diagram is shown in Figure 7.13.

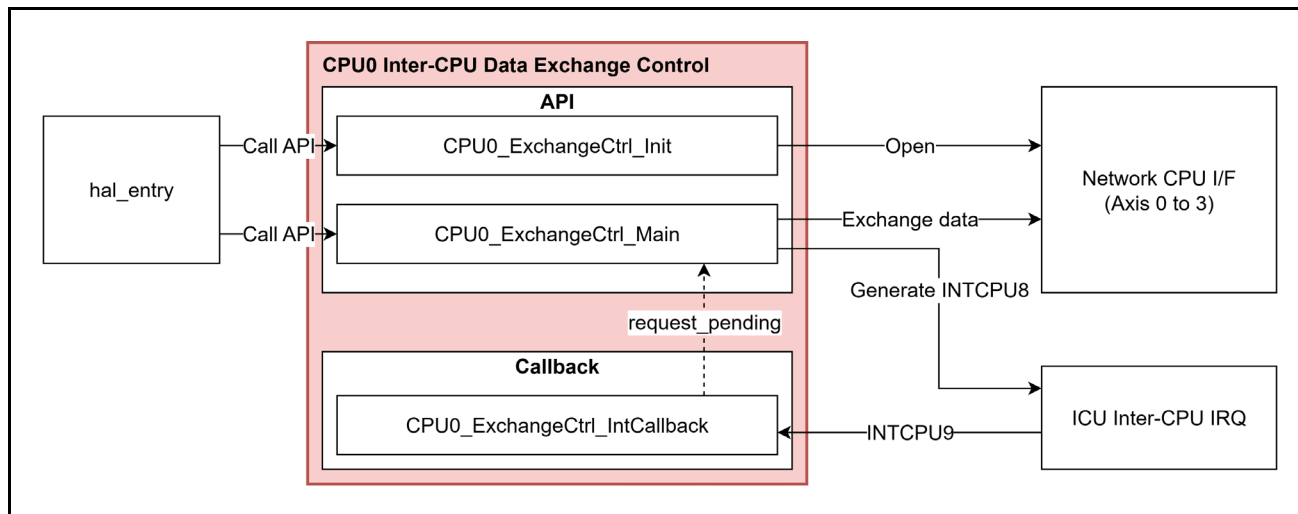


Figure 7.13. CPU0 Inter-CPU Data Exchange Control Module Structure

7.2.4.2 API

Table 7.17. CPU0 Inter-CPU Data Exchange Control API List (cpu0_exchange_ctrl.c)

API	Description
CPU0_ExchangeCtrl_Init	Initializes the CPU0 Inter-CPU Data Exchange Control module. Opens the Network CPU I/F for 4 axes, and opens the Inter CPU IRQ for INTCPU9 reception and INTCPU8 transmission.
CPU0_ExchangeCtrl_Main	If INTCPU9 has not been received, returns without doing anything. If it has been received, calls R_NETWORK_CPU_IF_CommandProcess, R_NETWORK_CPU_IF_UpdateCyclicReference, and R_NETWORK_CPU_IF_ReportMotorStatus in this order for all 4 axes, and then generates INTCPU8 to notify CPU1.

7.2.4.3 Callback Function

Table 7.18. CPU0 Inter-CPU Data Exchange Control Callback Functions List (cpu0_exchange_ctrl.c)

Function Name	Description
CPU0_ExchangeCtrl_IntCallback	Callback function of INTCPU9 (r_icu_inter_cpu_irq ch9). Sets request_pending to true. Does not access Shared Memory.

7.2.4.4 Structure and Variable Information

Table 7.19. CPU0 Inter-CPU Data Exchange Control Structure Definitions (cpu0_exchange_ctrl.h)

Structure	Variable	Description
cpu0_exchange_ctrl_cfg_t Configuration structure	p_network_cpu_if_ctrl[4]	Pointers to the Network CPU I/F control structures for axes 0 to 3
	p_network_cpu_if_cfg[4]	Pointers to the Network CPU I/F configuration structures for axes 0 to 3
cpu0_exchange_ctrl_t Control structure	request_pending	Flag indicating that INTCPU9 has been received and data exchange is required
	p_cfg	Pointer to the configuration structure

7.2.4.5 Global Variable Information

Table 7.20. CPU0 Inter-CPU Data Exchange Control Global Variables (cpu0_exchange_ctrl.c)

Type	Instance Name	Description
cpu0_exchange_ctrl_t	g_cpu0_exchange_ctrl	Control structure instance of the CPU0 Inter-CPU Data Exchange Control. Zero-initialized; p_cfg is set by the Init function.
cpu0_exchange_ctrl_cfg_t	g_cpu0_exchange_ctrl_cfg	Configuration structure instance of the CPU0 Inter-CPU Data Exchange Control. Initialized with the Network CPU I/F control and configuration structure instances for 4 axes.

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

7.3.1 Basic Specifications

Table 7.21. RZ/T2N Cortex-R52 CPU1 Basic Specifications

Function	Item	Description
EtherCAT SubDevice Stack	Interface with PHY	RGMII
	MDIO type	GMAC
	Number of Communication Ports	2 ports (In, Out)
	EtherCAT LED	4 types: RUN, ERR, L/A IN, L/A OUT
	EtherCAT Device ID	Configurable via SW6(4bit)
	Device Profile	CiA402 Drive Profile
	Mailbox	Supports CoE
	Synchronization Mode	Supports 2 types: - DC Mode (DC-Synchronous with SM2 event) - SM Mode (SM-synchronous)
	DC Mode Cycle Time	Minimum 50 μ s, maximum approximately 3 s
	Interrupt Priority (PDI, Sync0, Sync1)	Priority 4
	Interrupt Priority (Timer)	Priority 6
CiA402 Drive Profile	Mode	Supports 4 types: - Profile Position Mode - Homing Mode - Cyclic Synchronous Position Mode - Cyclic Synchronous Velocity Mode
	Number of Motor Axes	4 axis
Motor CPU I/F	Role	Provides communication interface with Motor Control CPU
	Communication Method	Shared Memory
	Interrupt Priority (INTCPU)	Priority 6
	Module number	4

7.3.2 Overall Software Structure

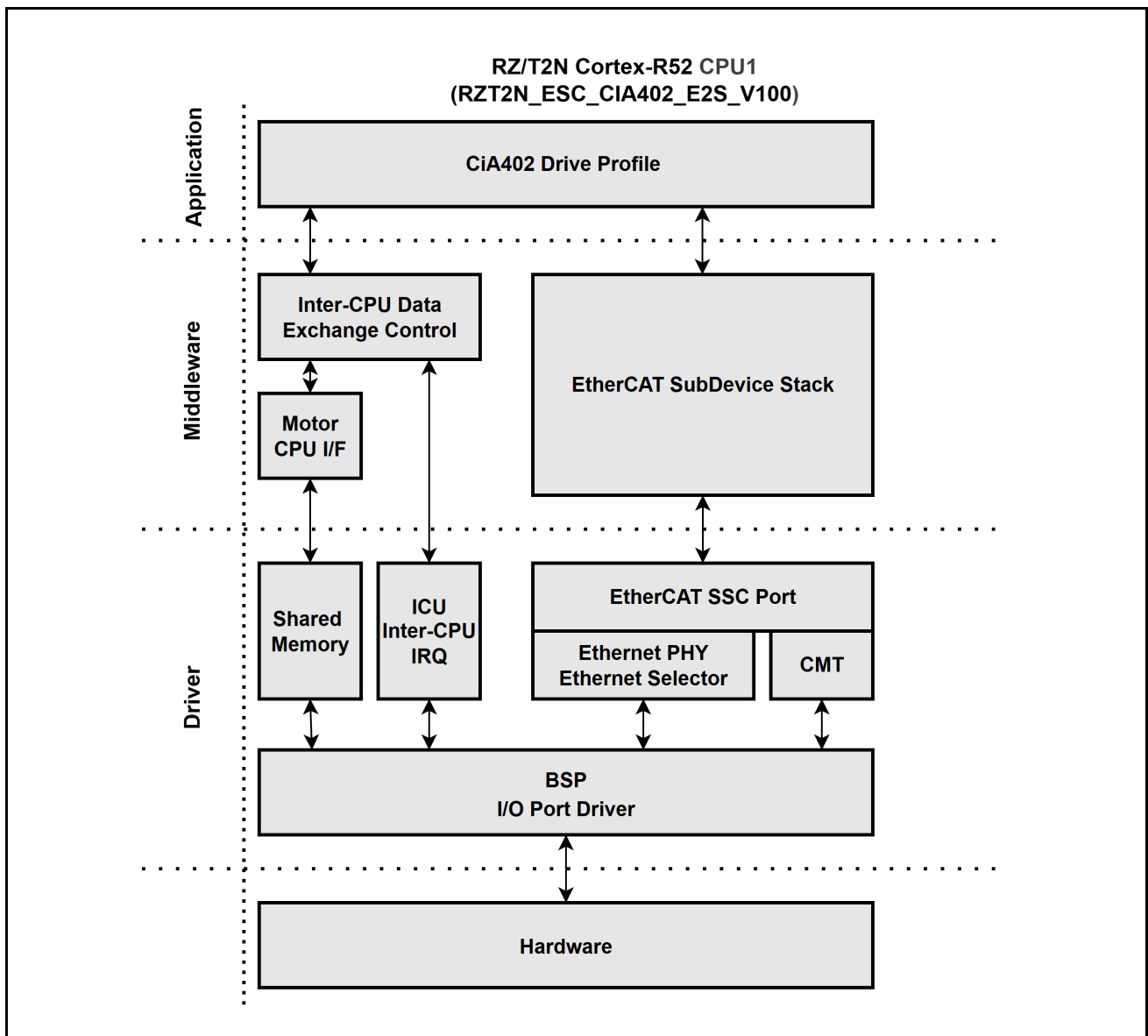


Figure 7.14. Overall Software Structure of RZ/T2N Cortex-R52 CPU1

7.3.3 src Folder Structure

Table 7.22. RZ/T2N CPU1 src Folder Structure

Folder / File				Description
src	ethercat	Beckhoff/Src	applInterface.h	EtherCAT SubDevice Stack (SSC Tool generated code)
			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 generated code)
			cia402appl.h	
		renesas	samplecia402.c	CiA402 Drive Profile (Renesas developed code)
			samplecia402.h	
	cpu1_exchange_ctrl			CPU1 Inter-CPU Data Exchange Control Module
	motor_cpu_if			Motor CPU I/F Module
	hal_entry.c			hal_entry function

7.3.4 hal_entry and Main Processing

The flowchart of the hal_entry function is shown in Figure 7.15.

Additionally, the functions called within hal_entry are listed in Table 7.23.

Table 7.23. Functions Called by hal_entry

Function Name	Description
RM_ETHERCAT_SSC_PORT_Open	Open function for EtherCAT SSC Port provided by FSP. Refer to <i>RZT Flexible Software Package Documentation</i> for details.
CPU1_ExchangeCtrl_Init	Initialization function for CPU1 Inter-CPU Data Exchange Control. See Section 7.3.7 for details.
ecat_ssc_init	Function that performs initialization of EtherCAT SubDevice Stack Code and CiA402 Drive Profile. The flowchart is shown in Figure 7.16. For details of functions and variables in the flowchart, refer to <i>Application Note ET9300 (EtherCAT SubDevice Stack Code)</i> .
R_BSP_PinSet	Function provided by FSP. Turns on LED9. Refer to <i>RZT Flexible Software Package Documentation</i> for details.
MainLoop	Main processing function for EtherCAT SubDevice Stack Code and CiA402 Drive Profile. Refer to <i>Application Note ET9300 (EtherCAT SubDevice Stack Code)</i> for details.

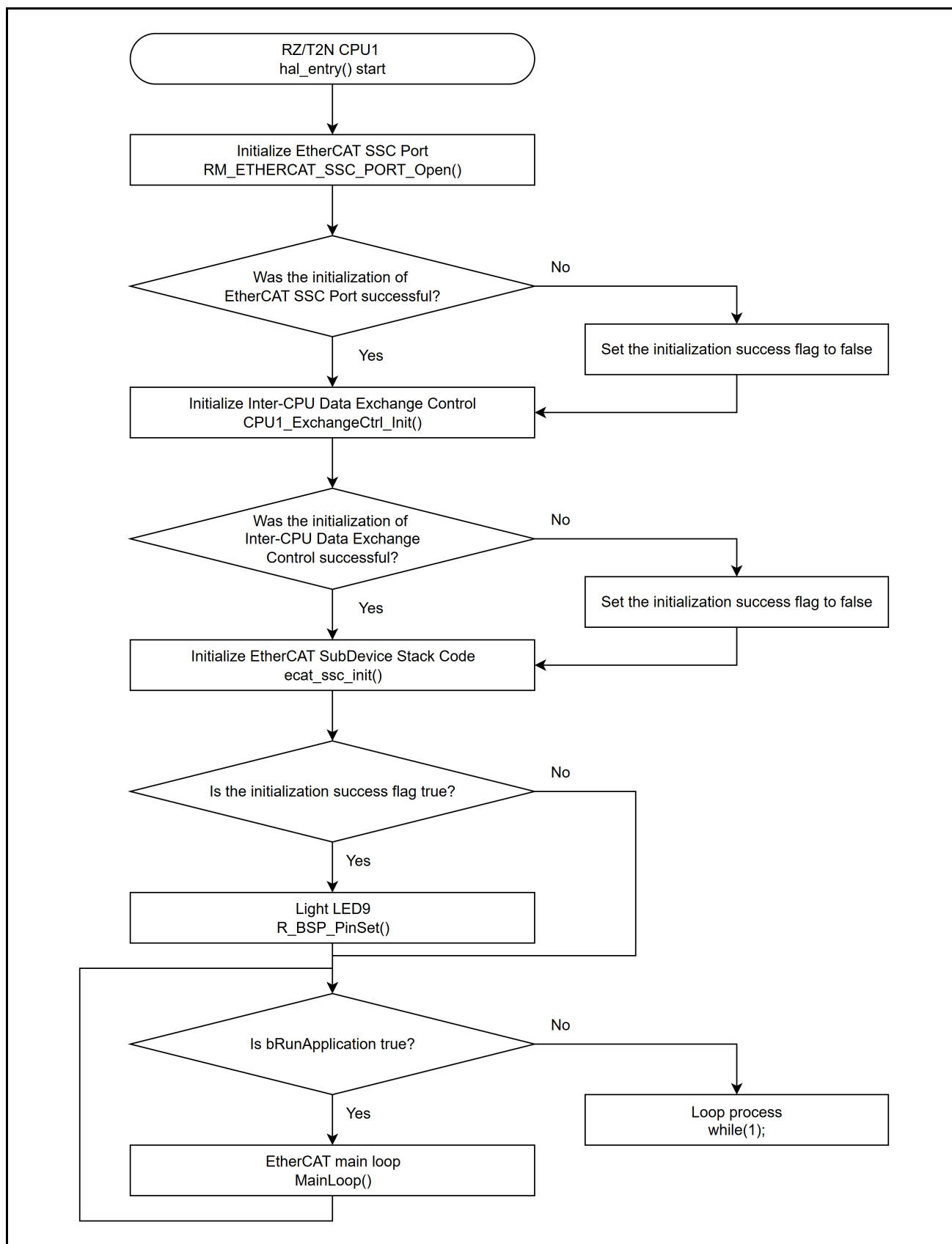
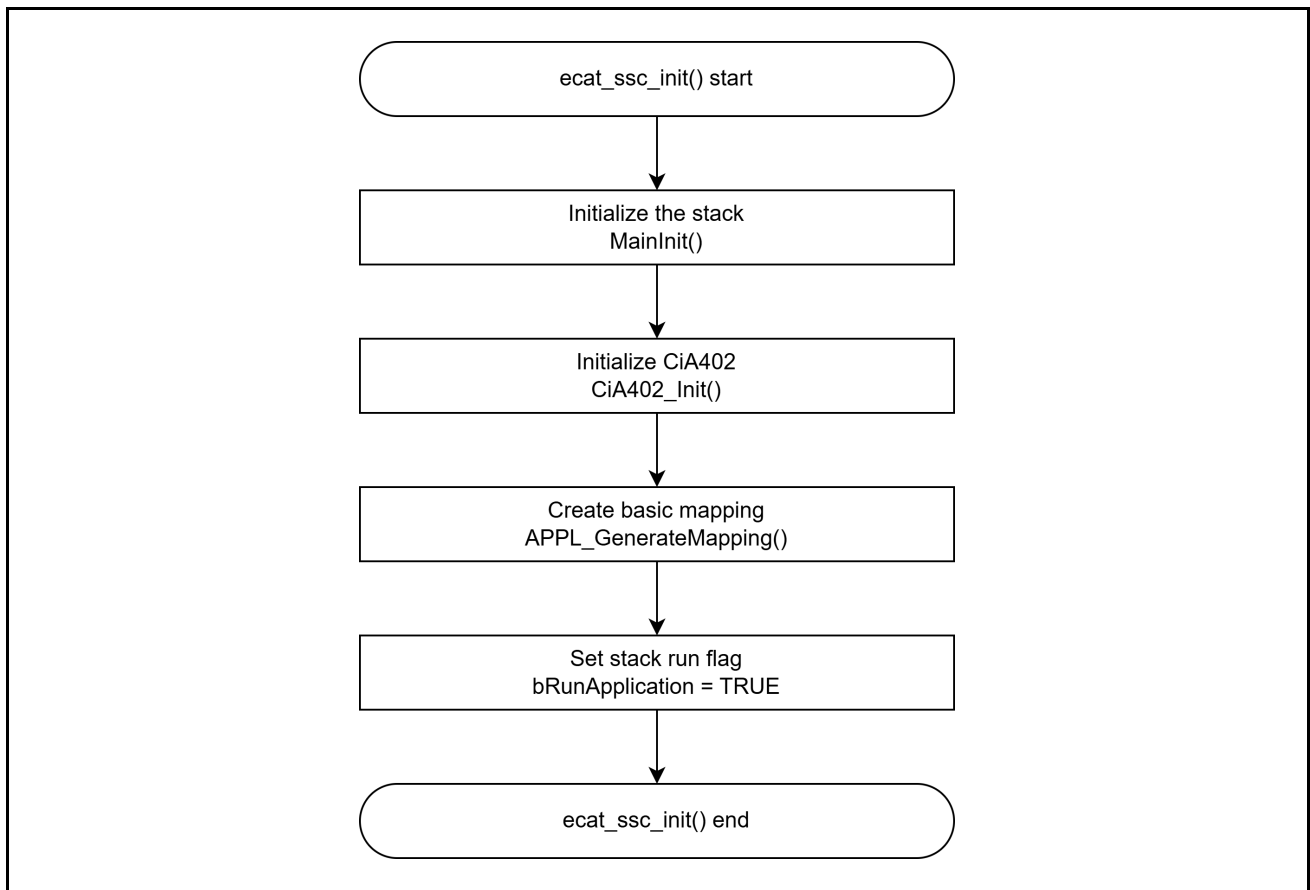


Figure 7.15. RZ/T2N CPU1 hal_entry function flowchart

**Figure 7.16. ecat_ssc_init function flowchart**

7.3.5 CiA402 Drive Profile

The CiA402 Drive Profile is a device profile for drives and motion control, primarily defining functional behavior for servo drives, sinusoidal inverters, and stepper motor controllers.

This profile specifies multiple operating modes and corresponding configuration parameters as an Object Dictionary.

It also includes a Finite State Automaton (FSA) that defines internal and external behavior for each state.

To change states, the Control Word object is used, and the result after transition is reflected in the Status Word object.

Control Word and various command values (e.g., speed) are assigned to RxPDO, while Status Word and actual values (e.g., position) are assigned to TxPDO.

For details, refer to the CiA402 specification document.

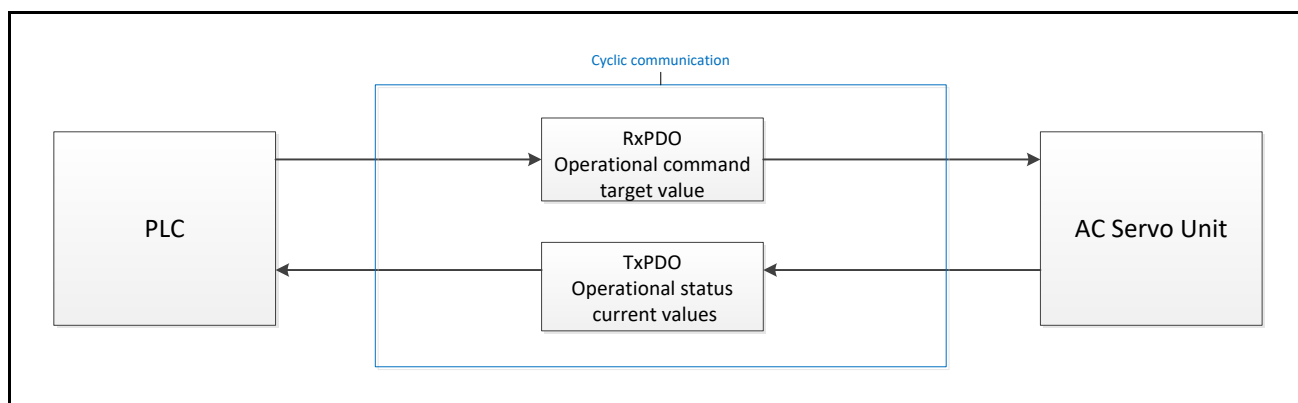


Figure 7.17. CiA402 Drive Profile Overview

7.3.5.1 CiA402 Operating Modes

Among the operating modes defined by CiA402, this program supports the following:

Table 7.24. Supported Operating Modes

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 (CSP)

In this mode, the target position is periodically transmitted for motor position control.

Table 7.25. CSP Mode RxPDO List

RxPDO	Index	Description
Control Word	0x6040	CiA402 control command
Target Position	0x607A	Position command value [deg] (*1)

(*1) The command value sent to the motor is converted to **1/10** of the input value.

Example: Input 1234 → Motor command value = 123.4.

Table 7.26. CSP Mode TxPDO List

TxPDO	Index	Description
Error Code	0x603F	CiA402 error code
Status Word	0x6041	CiA402 state
Modes of Operation Display	0x6061	Mode type (=8)
Position Actual Value	0x6064	Motor current position [deg] (*2)
Velocity Actual Value	0x606C	Motor current speed [rpm] (*2)

(*2) Actual values are converted by multiplying by **10**. Example: Output 5678 → Actual value = 567.8.

Table 7.27. CSP Mode Motor Settings

Item	Setting
LOOP MODE	Position control
Position command generation mode	Trapezoidal drive motion

7.3.5.3 Cyclic synchronous velocity mode (CSV)

In this mode, the target velocity is periodically transmitted for motor speed control.

Table 7.28. CSV Mode RxPDO List

RxPDO	Index	Description
Control Word	0x6040	CiA402 control command
Target Velocity	0x60FF	Velocity command value [rpm] (*1)

(*1) The command value sent to the motor is converted to **1/10** of the input value.

Example: Input 1234 → Motor command value = 123.4.

Table 7.29. CSV Mode TxPDO List

TxPDO	Index	Description
Error Code	0x603F	CiA402 error code
Status Word	0x6041	CiA402 state
Modes of Operation Display	0x6061	Mode type (=9)
Position Actual Value	0x6064	Motor current position [deg] (*2)
Velocity Actual Value	0x606C	Motor current speed [rpm] (*2)

(*2) Actual values are converted by multiplying by **10**. Example: Output 5678 → Actual value = 567.8.

Table 7.30. CSV Mode Motor Settings

Item	Setting
LOOP MODE	Speed control

7.3.5.4 Profile position mode (PP)

In this mode, speed and acceleration are set for motor position control.

Table 7.31. PP Mode RxPDO List

RxPDO	Index	Description
Control Word	0x6040	CiA402 control command
Target Position	0x607A	Position command value [deg] (*1)
Profile Velocity	0x6081	Profile velocity [rpm] (*1)
Profile Acceleration	0x6083	Profile acceleration [rpm/s] (*1)
Profile Deceleration	0x6084	Profile deceleration [rpm/s] (*3)

(*1) The command value sent to the motor is converted to $1/10$ of the input value.

Example: Input 1234 → Motor command value = 123.4.

(*3) Not used in this software.

Table 7.32. PP Mode TxPDO List

TxPDO	Index	Description
Error Code	0x603F	CiA402 error code
Status Word	0x6041	CiA402 state
Modes of Operation Display	0x6061	Mode type (=1)
Position Actual Value	0x6064	Motor current position [deg] (*2)
Velocity Actual Value	0x606C	Motor current speed [rpm] (*2)

(*2) Actual values are converted by multiplying by **10**. Example: Output 5678 → Actual value = 567.8.

Table 7.33. PP Mode Motor Settings

Item	Setting
LOOP MODE	Position control
Position command generation mode	Trapezoidal drive motion
Accel/Decel time [s]	(Profile Velocity) / (Profile Acceleration)
Max trapezoidal speed [rpm]	(Profile Velocity) / 10

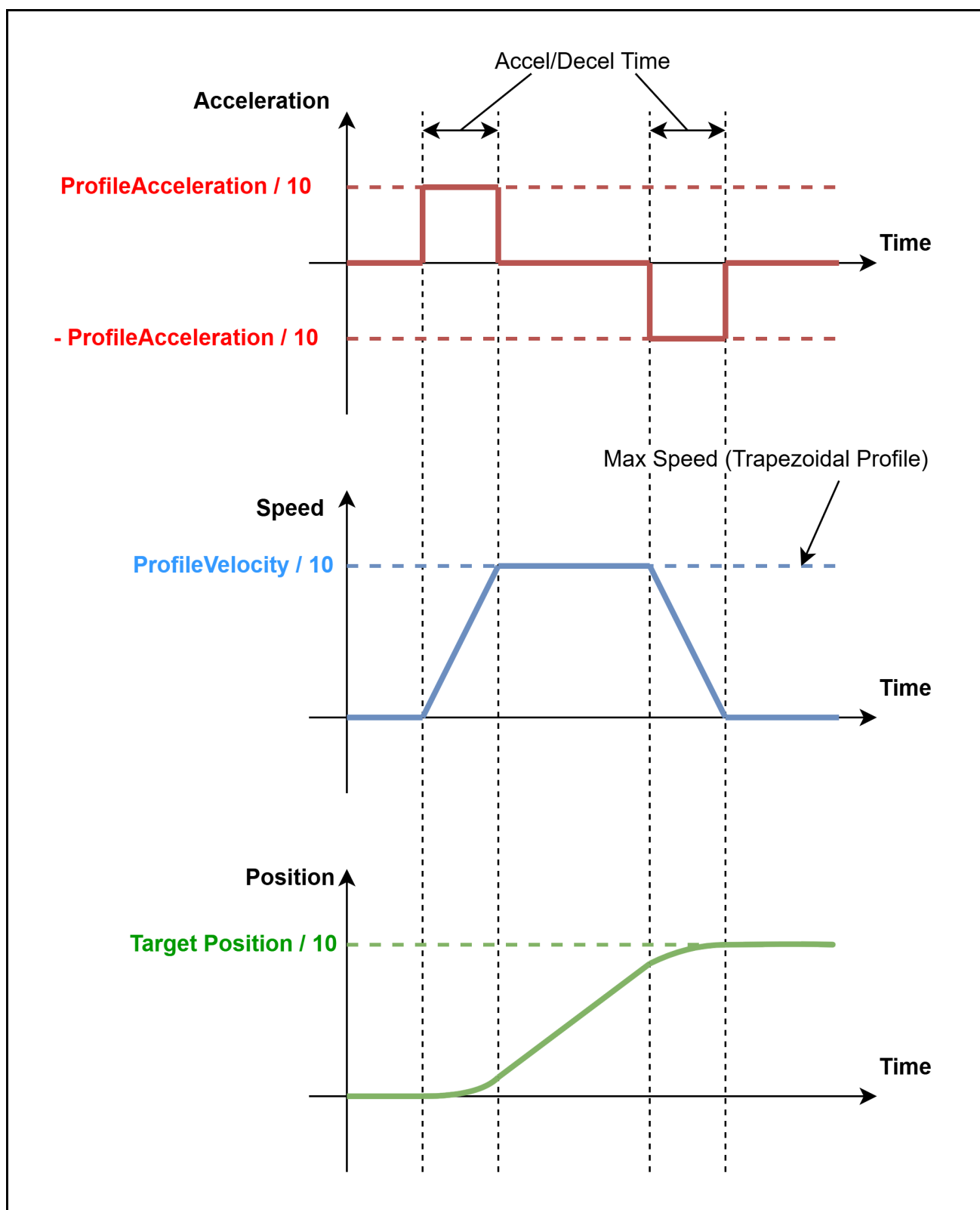


Figure 7.18. Trapezoidal Drive Motion Settings for PP Mode

7.3.5.5 Homing Mode (HM)

In this mode, Homing Method 37 is used to change the motor's current position.

Table 7.34. Homing Mode RxPDO List

RxPDO	Index	Description
Control Word	0x6040	CiA402 control command
Home Offset	0x607C	Commanded current position [deg] (*1)
Home Method	0x6098	Homing method
Homing Speed For Switch	0x6099	Switch homing speed (*3)
Homing Speed For Zero	0x6099	Zero homing speed (*3)
Homing Acceleration	0x609A	Overall homing accel/decel (*3)

(*1) The command value sent to the motor is converted to **1/10** of the input value.

Example: Input 1234 → Motor command value = 123.4.

(*3) Not used in this software.

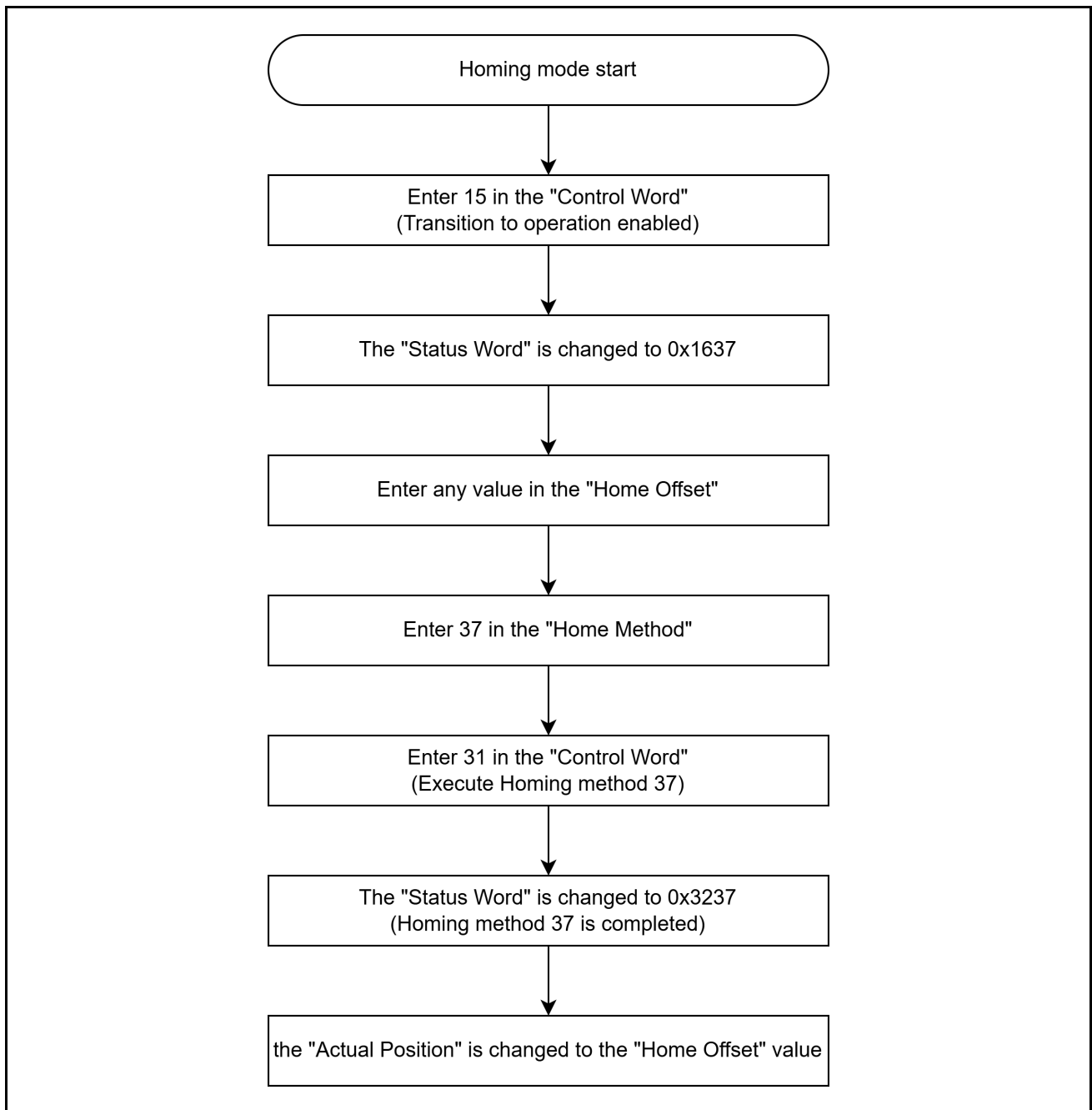
Table 7.35. Homing Mode TxPDO List

TxPDO	Index	Description
Error Code	0x603F	CiA402 error code
Status Word	0x6041	CiA402 state
Modes of Operation Display	0x6061	Mode type (=6)
Position Actual Value	0x6064	Motor current position [deg] (*2)
Velocity Actual Value	0x606C	Motor current speed [rpm] (*2)

(*2) Actual values are converted by multiplying by **10**. Example: Output 5678 → Actual value = 567.8.

Table 7.36. Homing Mode Motor Settings

Item	Setting
LOOP MODE	Position control
Position command generation mode	Trapezoidal drive motion

**Figure 7.19. Flowchart for Executing Homing Method 37**

7.3.5.6 CiA402 State Transitions

As defined by CiA402 FSA (Finite State Automaton), the sample program supports the following transitions. Table 7.37 shows the corresponding transition numbers, their triggers, and behavior.

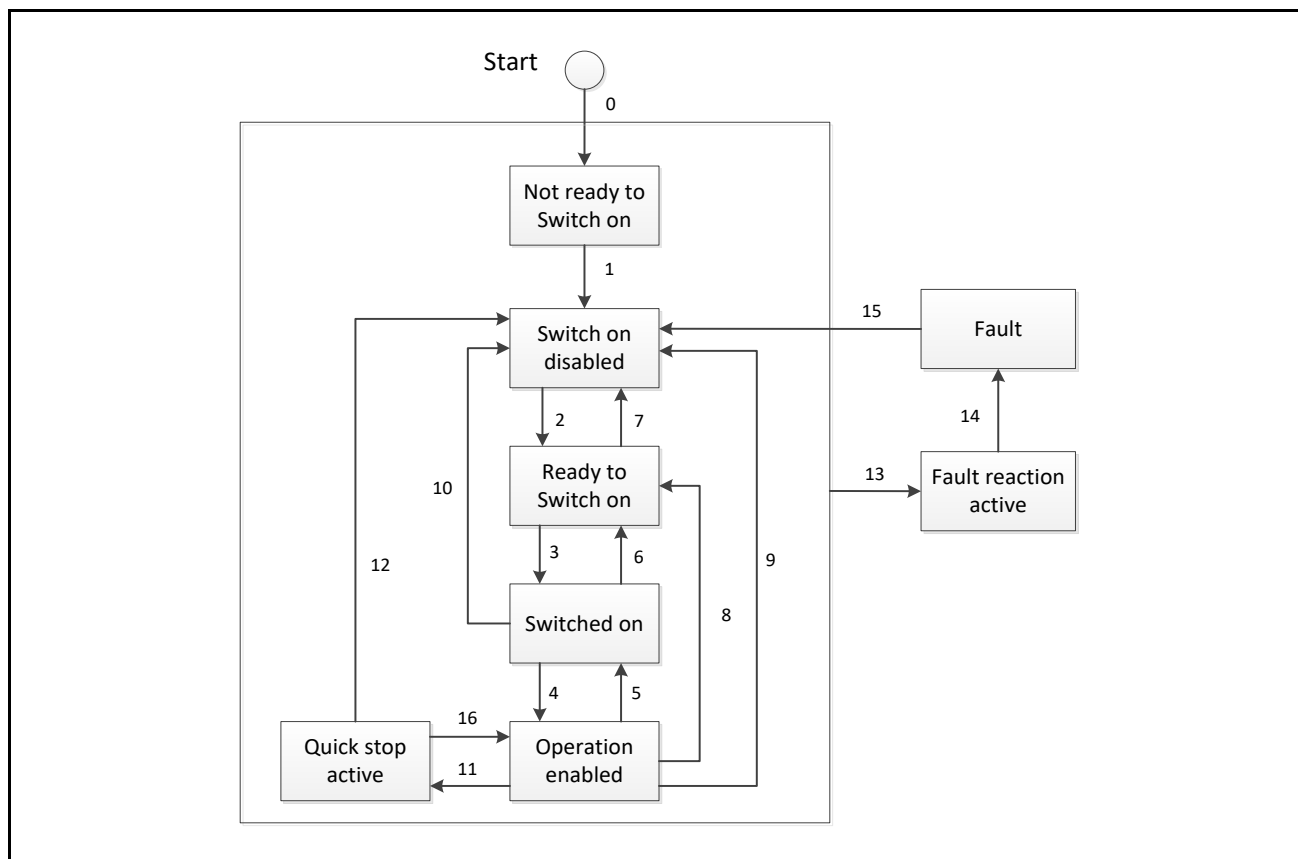


Figure 7.20. CiA402 State Transition Diagram

Table 7.37. CiA402 State Transitions

Transition No.	Trigger	Behavior
0, 1	Automatically triggered if EtherCAT State Machine (ESM) is in OP	No action
2		See Figure 7.21
3	Control Word = 7 or Control Word = 15	See Figure 7.22
4	Control Word = 15	See Figure 7.23
5	Control Word = 7	See Figure 7.24
6	Control Word = 6	No action
7	Control Word = 0	No action
8	Control Word = 6	See Figure 7.24
9	Control Word = 0	See Figure 7.24
10	Control Word = 0	No action
11	Control Word = 2	See Figure 7.24
12	Control Word = 0	No action

13	A) When executing transition 2 and a motor error occurs B) When Operation Enabled and a motor error occurs	See Figure 7.25
14	Automatically triggered when Fault Reaction Active	No action
15	Control Word = 128	See Figure 7.26
16	Control Word = 15	See Figure 7.27

7.3.5.7 CiA402 Error Code

When transitioning to Fault Reaction Active, the error code (0x603F) is set according to the error that occurred.

Table 7.38 lists the error codes.

Table 7.38. Error Code (0x603F) List

Error Type	Error Code	Description
Motor Error	0x2214	HW overcurrent error (0x0001) occurred
	0x3211	Overvoltage error (0x0002) occurred
	0x7310	Overspeed error (0x0004) occurred
	0x3221	Undervoltage error (0x0080) occurred
	0x2220	SW overcurrent error (0x0100) occurred
	0x4310	Overtemperature error (0x0200) occurred
	0x7120	Undefined error (0xFFFF) occurred
Motor Stop Request	0xFF00	STO request occurred
	0xFF01	SS1-t request occurred
Motor Operation Prohibited	0x2250	Motor operation is prohibited

7.3.5.8 Functions for CiA402 Drive Profile

This section describes the functions implemented in this sample software.

For functions implemented in the EtherCAT SubDevice Stack Code generated by SSC Tool, refer to:

“Application Note ET9300 (EtherCAT SubDevice Stack Code)”.

Table 7.39. CiA402 State Transition Functions (samplecia402.c)

Function Name	Description
CiA402_StateTransition1	No processing in this software (*1)
CiA402_StateTransition2	Flowchart shown in Figure 7.21 (*1)
CiA402_StateTransition3	Flowchart shown in Figure 7.22 (*1)
CiA402_StateTransition4	Flowchart shown in Figure 7.23 (*1)
CiA402_StateTransition5	Flowchart shown in Figure 7.24 (*1)
CiA402_StateTransition6	No processing in this software (*1)
CiA402_StateTransition7	No processing in this software (*1)
CiA402_StateTransition8	Flowchart shown in Figure 7.24 (*1)
CiA402_StateTransition9	Flowchart shown in Figure 7.24 (*1)
CiA402_StateTransition10	No processing in this software (*1)
CiA402_StateTransition11	Flowchart shown in Figure 7.24 (*1)
CiA402_StateTransition12	No processing in this software (*1)
CiA402_LocalError	Flowchart shown in Figure 7.25 (*1)
CiA402_StateTransition14	No processing in this software (*1)
CiA402_StateTransition15	Flowchart shown in Figure 7.26 (*1)
CiA402_StateTransition16	Flowchart shown in Figure 7.27 (*1)

(*1) Called when state transitions 1–16 defined in CiA402 FSA (Figure 7.20) occur.

Table 7.40. Functions for CiA402 (samplecia402.c)

Function Name	Description
APPL_MOTOR_MotionControl_Main	Motor control function called periodically from CiA402_Application function. Flowchart shown in Figure 7.28
check_position_limit	Checks whether the current position exceeds the limit. Flowchart shown in Figure 7.29
int32_to_real32	Converts int32 to real32 (float)
uint32_to_real32	Converts uint32 to real32 (float)
rpm_to_rads	Converts rpm to rad
update_actual_value	Retrieves motor's current position and speed. Flowchart shown in Figure 7.30
process_error_state	Called when a motor error occurs. Flowchart shown in Figure 7.31

process_profile_position_mode	Called in PP mode. Flowchart shown in Figure 7.32
process_homing_mode	Called in HM mode. Flowchart shown in Figure 7.33
process_cyclic_position_mode	Called in CSP mode. Flowchart shown in Figure 7.34
process_cyclic_velocity_mode	Called in CSV mode. Flowchart shown in Figure 7.35
APPL_GetDeviceID	Function to retrieve EtherCAT ID

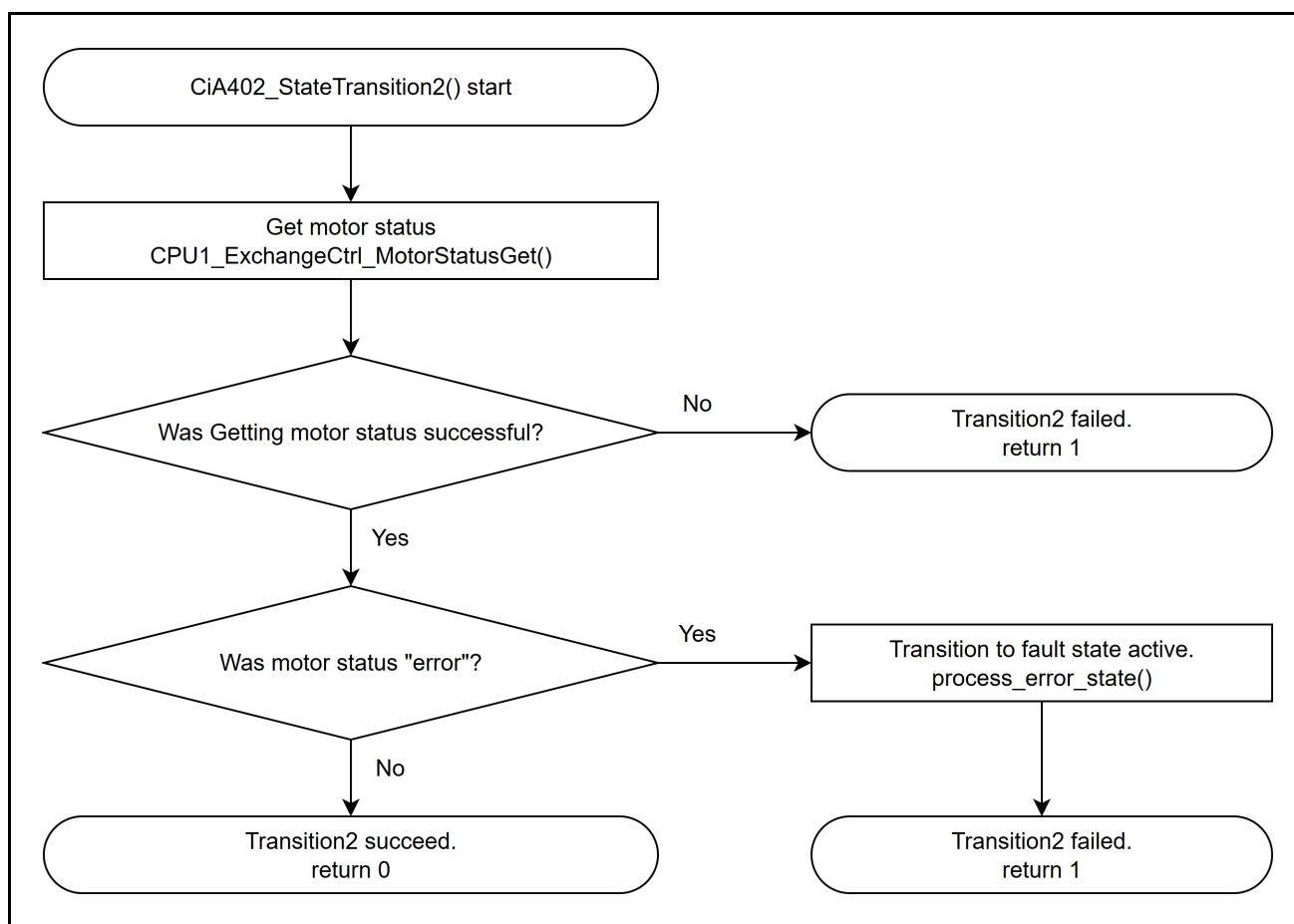


Figure 7.21. CiA402_StateTransition2 function flowchart

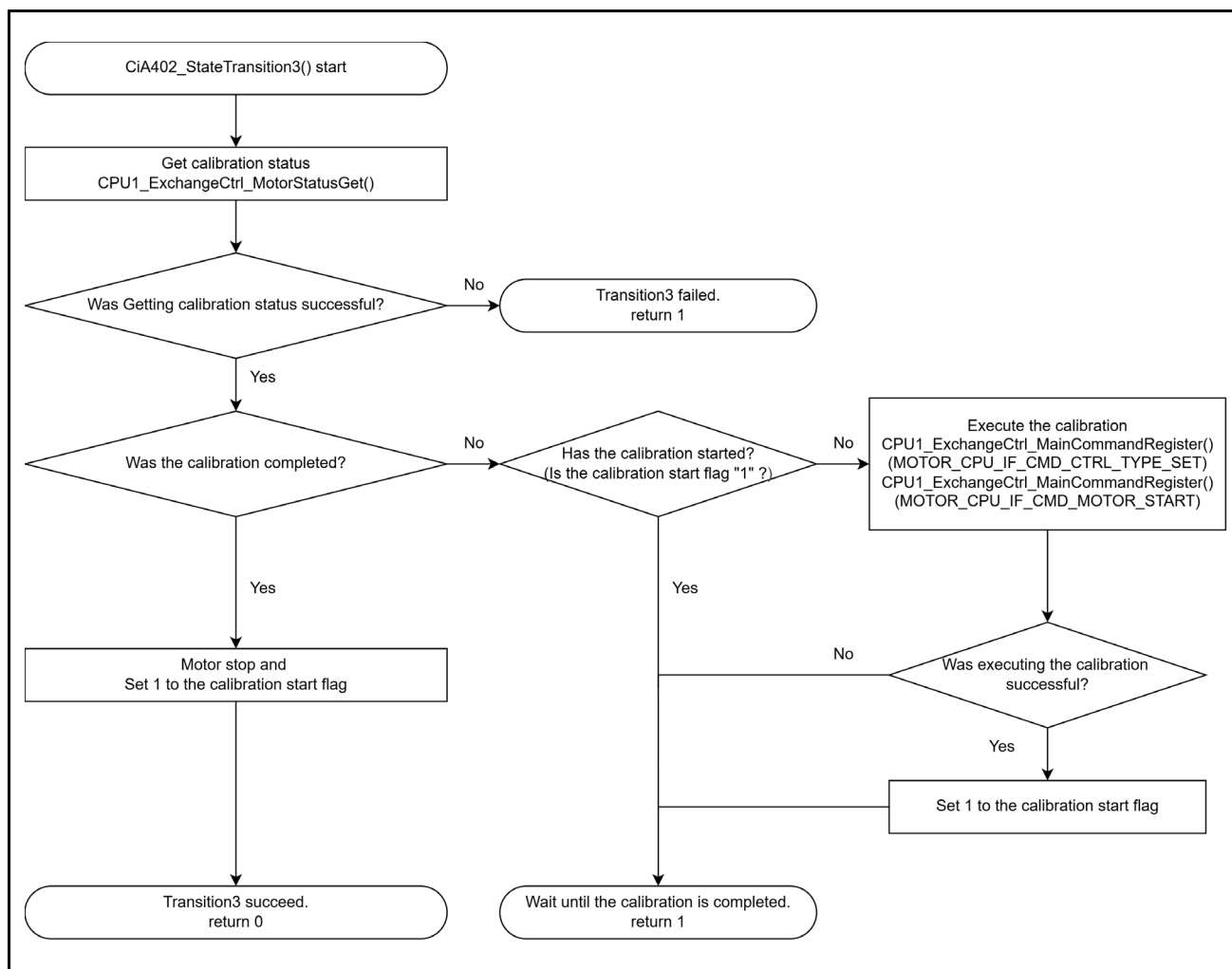


Figure 7.22. CiA402_StateTransition3 function flowchart

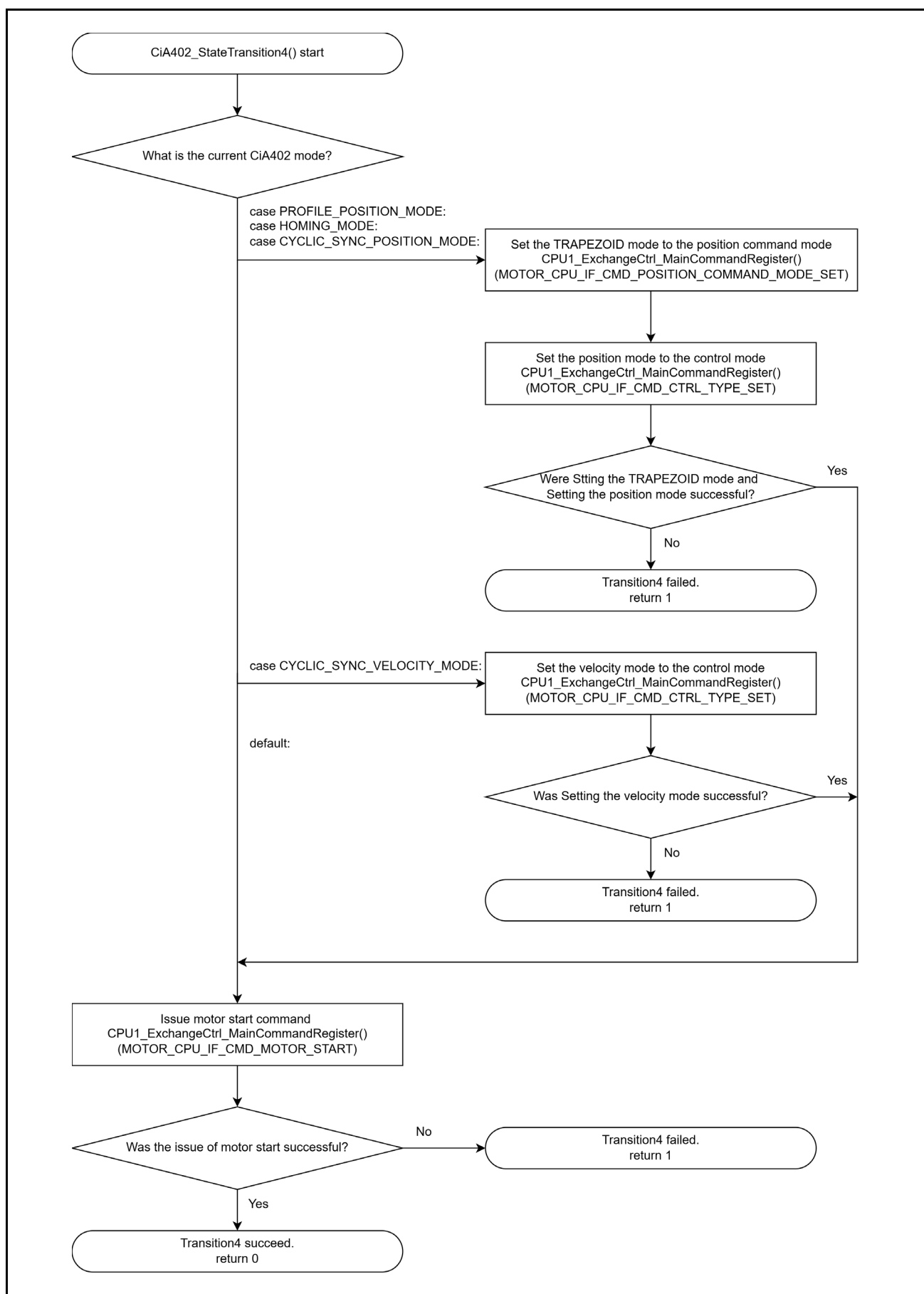
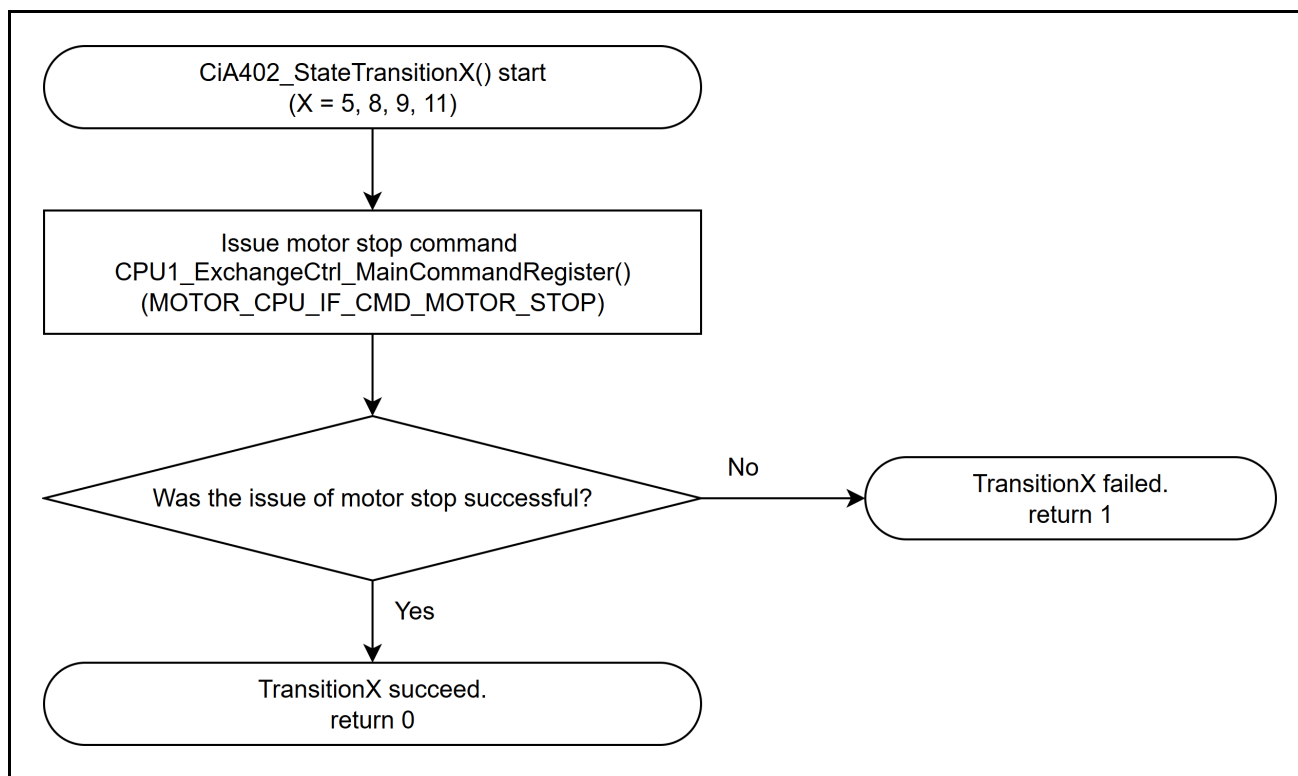
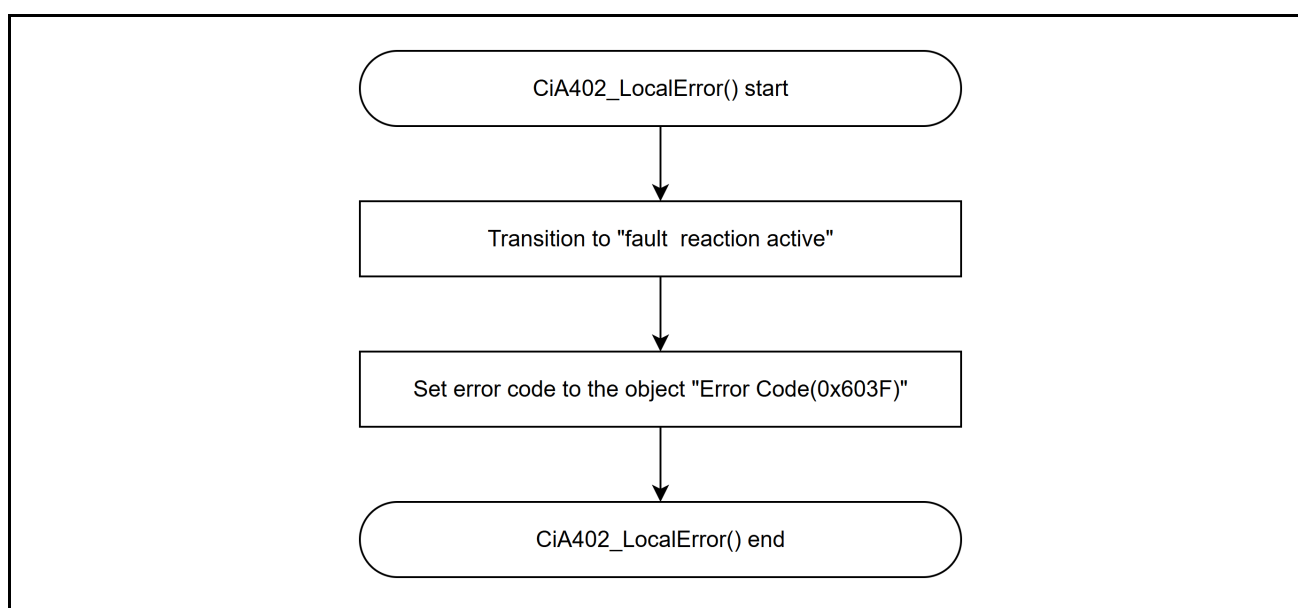
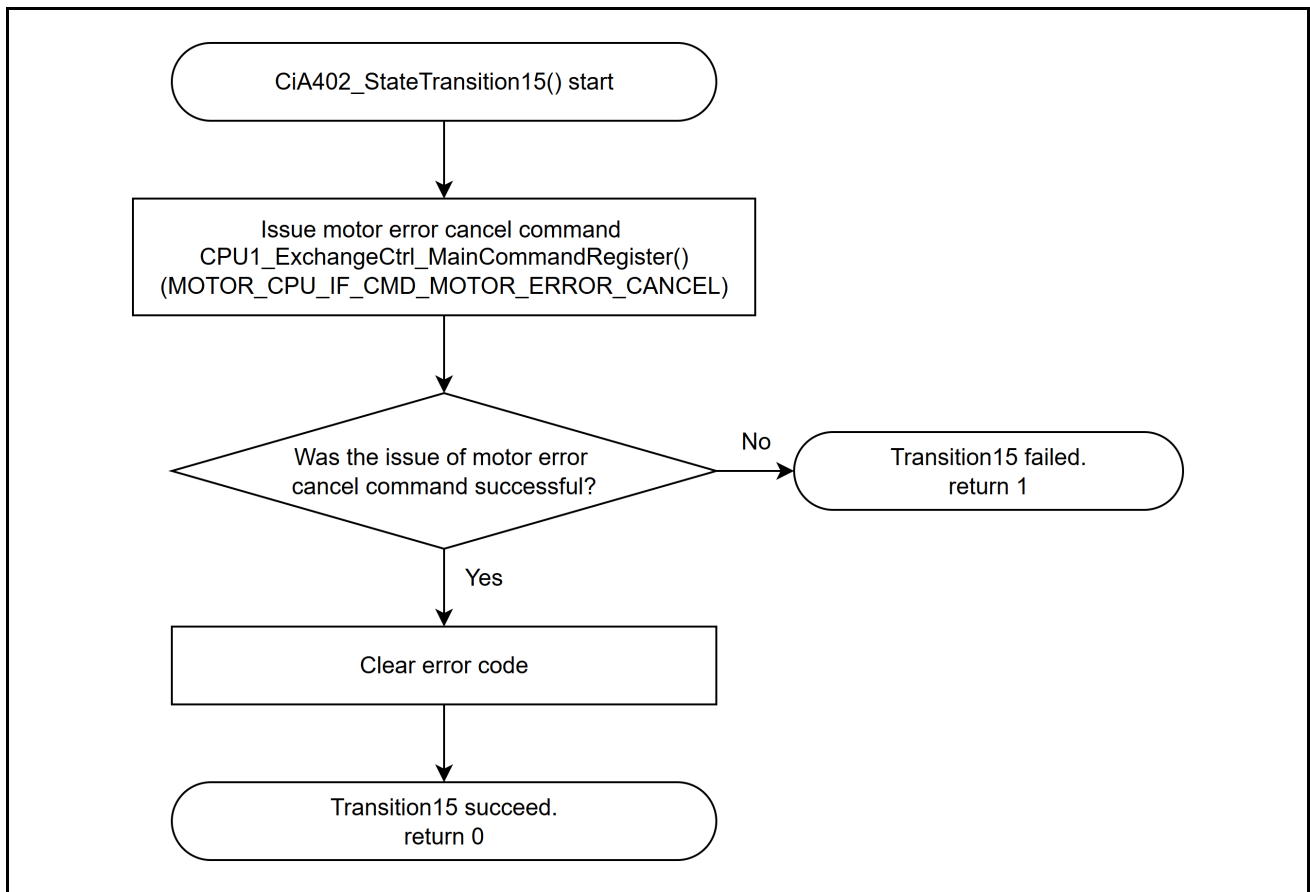
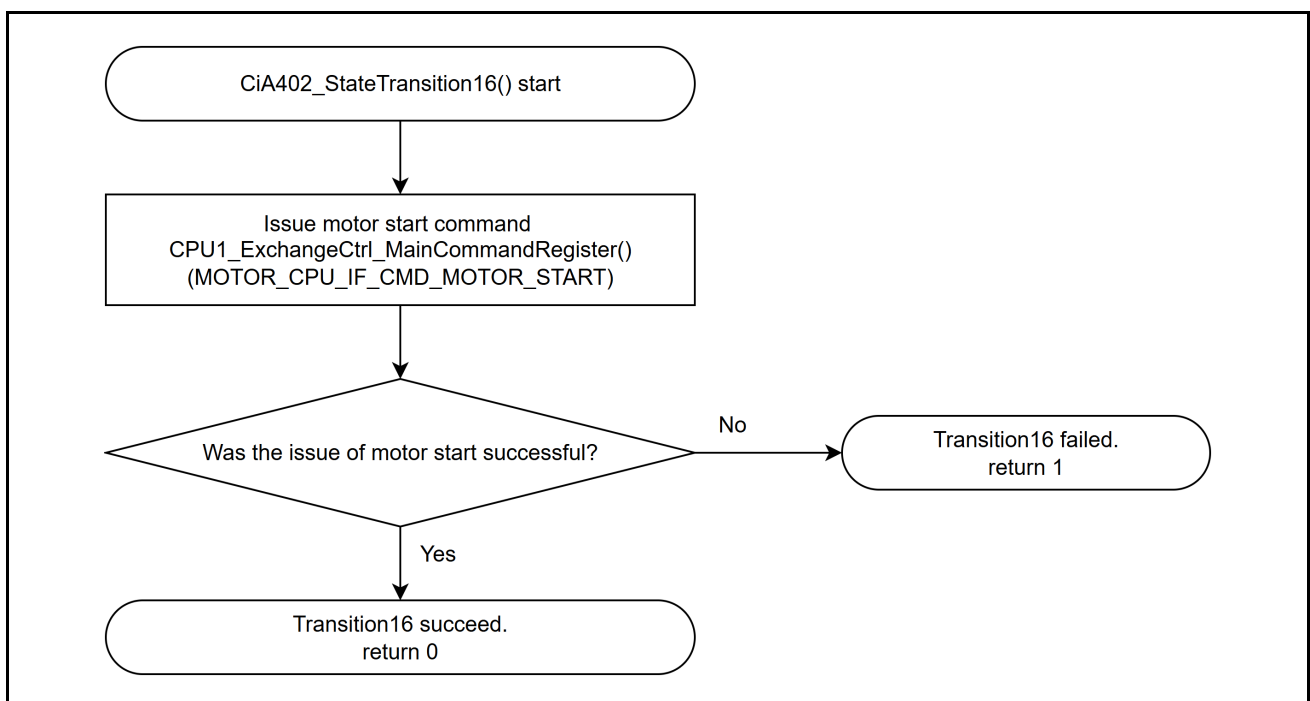


Figure 7.23. CiA402_StateTransition4 function flowchart

**Figure 7.24. CiA402_StateTransition5, 8, 9, 11 function flowcharts****Figure 7.25. CiA402_LocalError function flowchart**

**Figure 7.26. CiA402_StateTransition15 function flowchart****Figure 7.27. CiA402_StateTransition16 function flowchart**

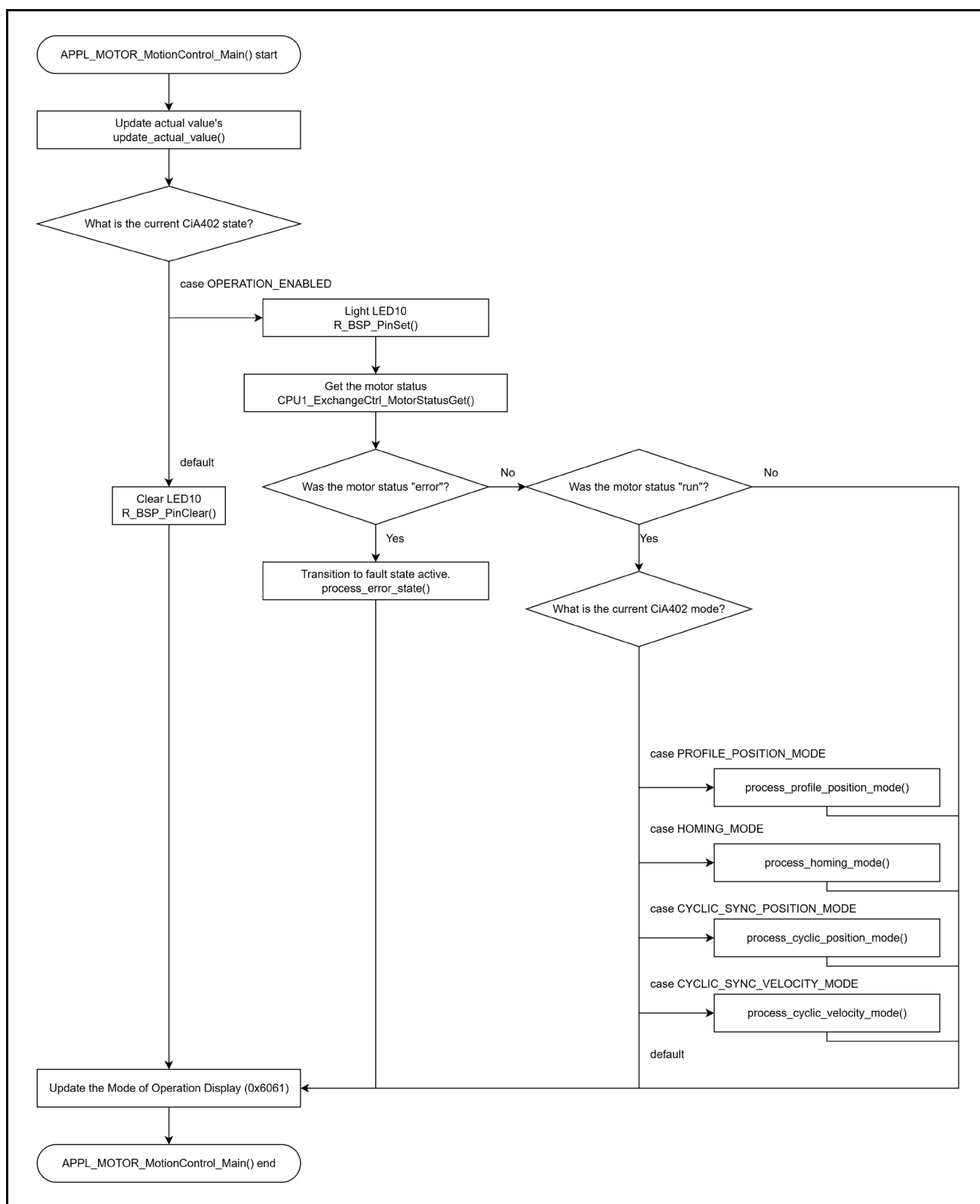
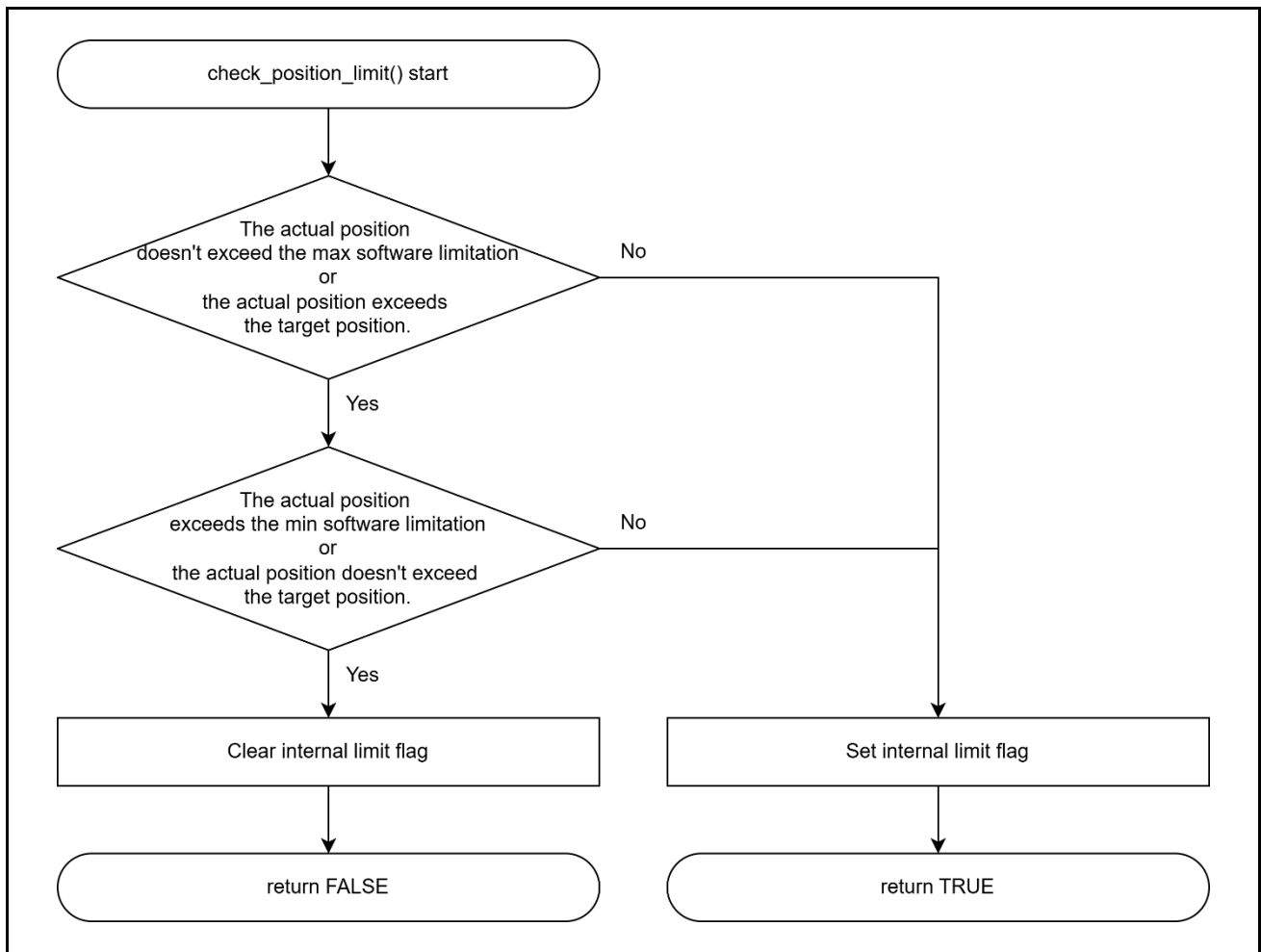
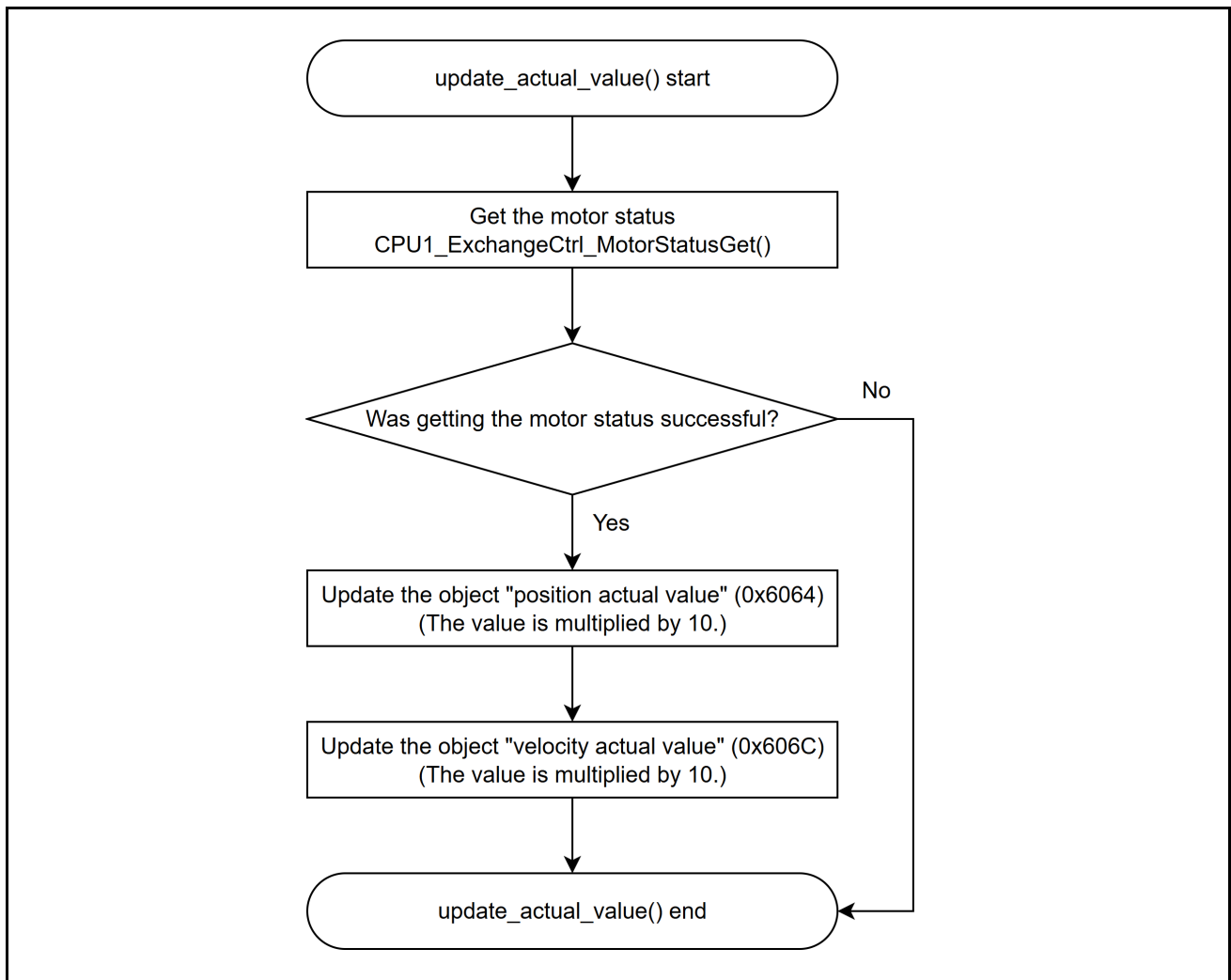


Figure 7.28. APPL_MOTOR_MotionControl_Main function flowchart

**Figure 7.29. check_position_limit function flowchart**

**Figure 7.30. update_actual_value function flowchart**

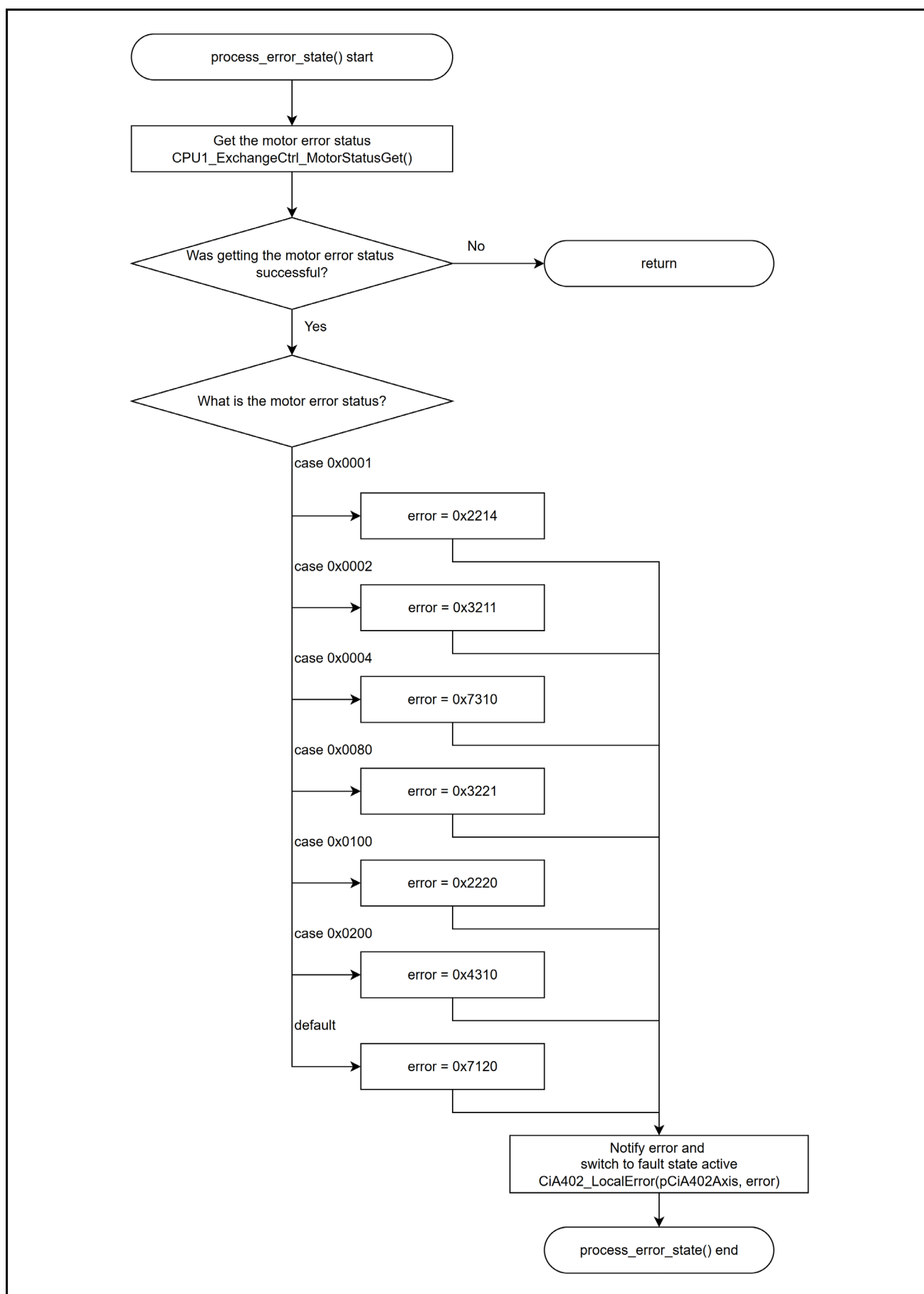


Figure 7.31. process_error_state function flowchart

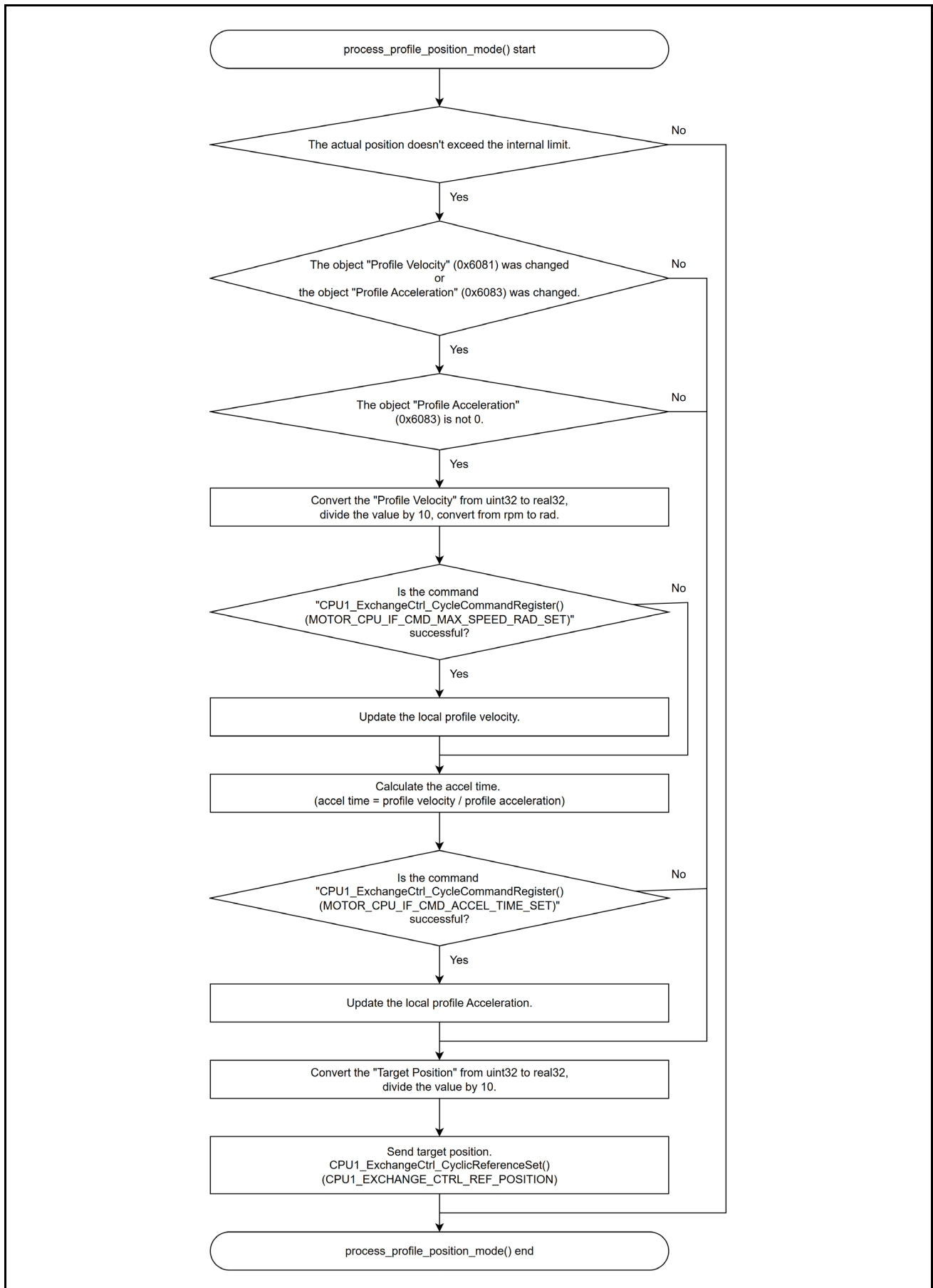


Figure 7.32. process_profile_position_mode function flowchart

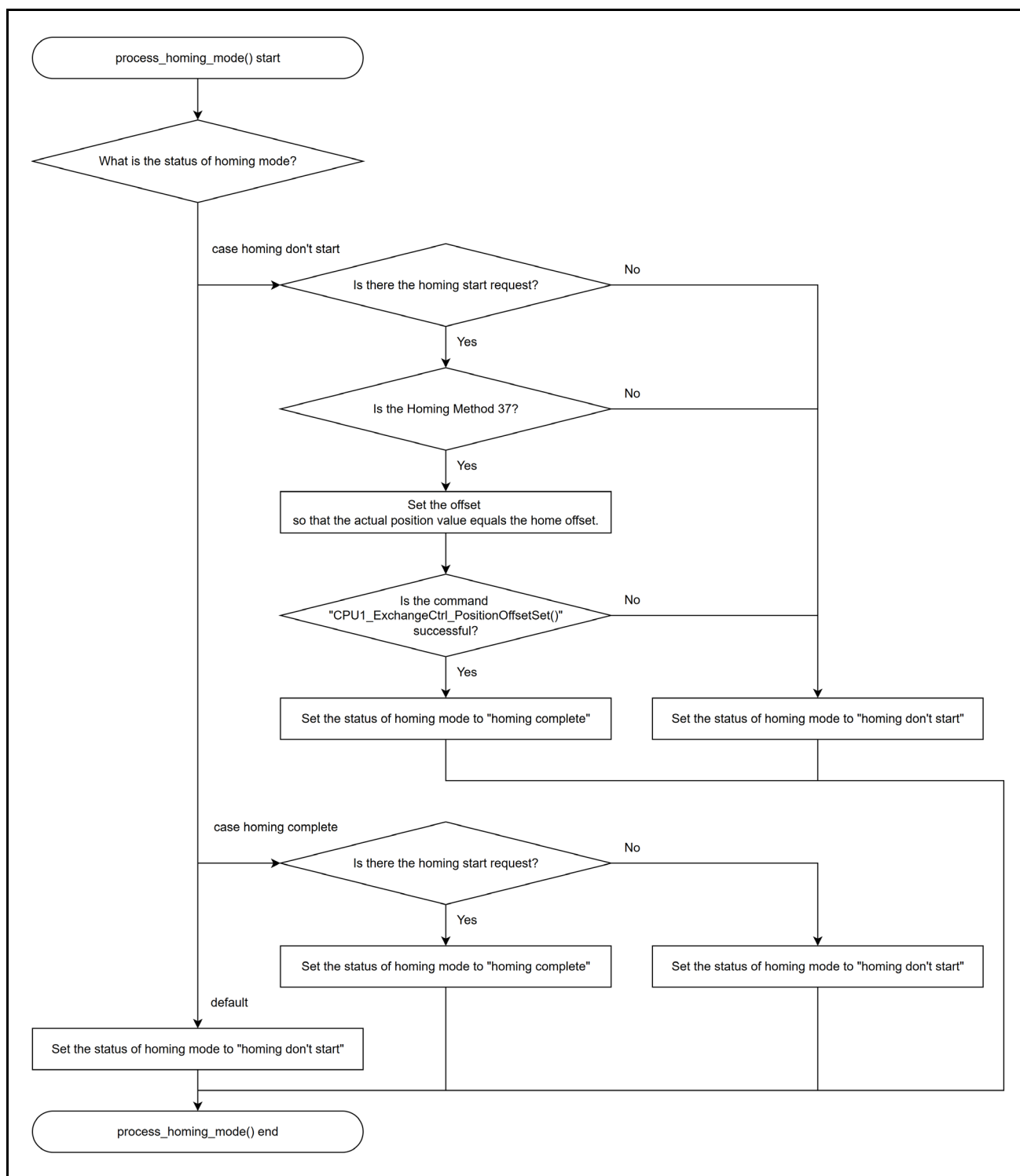
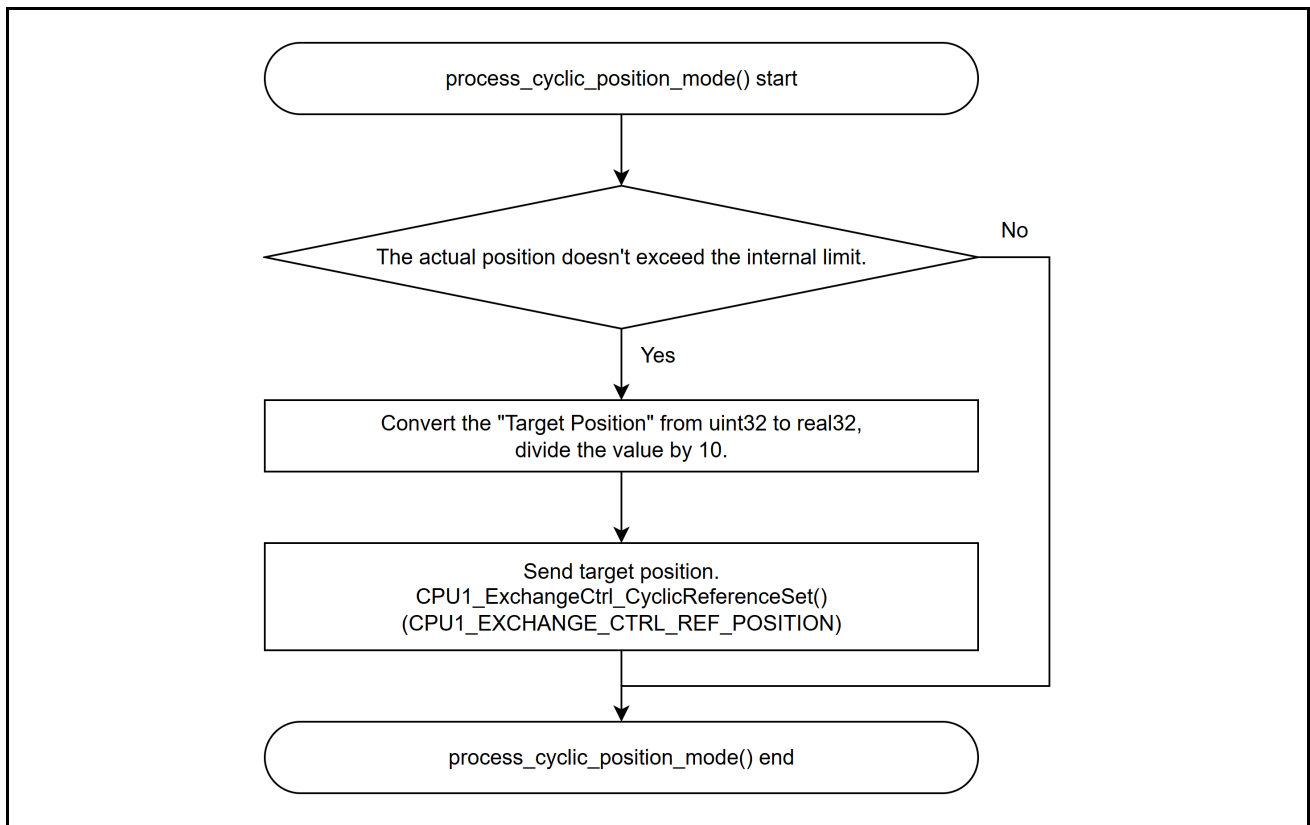
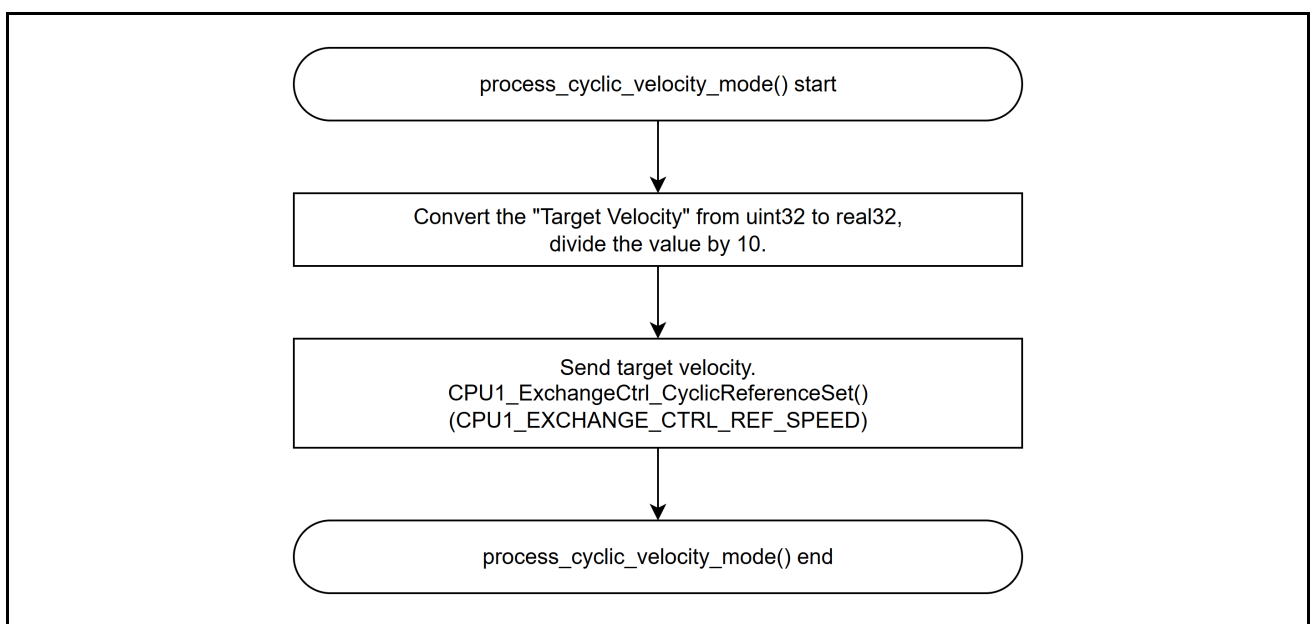


Figure 7.33. process_homing_mode function flowchart

**Figure 7.34. process_cyclic_position_mode function flowchart****Figure 7.35. process_cyclic_velocity_mode function flowchart**

7.3.5.9 Macro Definitions

This section describes the macro definitions implemented in this sample software.

For macro definitions implemented in the EtherCAT SubDevice Stack Code generated by the SSC Tool, refer to:

“Application Note ET9300 (EtherCAT SubDevice Stack Code)”.

Table 7.41. Macro Definitions for CiA402 Drive Profile (samplecia402.c)

Macro Name	Definition Value	Remarks
CFG_POS_MODE	MOTOR_CPU_IF_POS_MODE_TRAPEZOID	Motor position control mode setting

7.3.5.10 Changes After SSC Tool Generated Files

This section describes the modifications made to the EtherCAT SubDevice Stack Code generated by the SSC Tool.

For details of the original source code before modification, refer to:

“Application Note ET9300 (EtherCAT SubDevice Stack Code)”.

Table 7.42. List of Changes After SSC Tool Generation

File Name	Change	Details
cia402appl.c	Added include paths	Added hal_data.h, samplecia402.h, cpu1_exchange_ctrl.h
	Modified CiA402_Init function	Added initialization for new members in TCiA402Axis structure and initialization for added objects
	Modified CiA402_StateMachine function	Changed to call each CiA402 transition function listed in Table 7.39 during state transitions
	Deleted CiA402_LocalError and CiA402_DummyMotionControl functions	CiA402_LocalError is defined in samplecia402.c
	Modified CiA402_Application function	Changed to call CiA402 transition functions listed in Table 7.39 during state transitions. Enabled motor operation in SM mode as well as DC mode. Replaced CiA402_DummyMotionControl with APPL_MOTOR_MotionControl_Main
	Modified Write0xF030 function	Updated processing to match changes in PDO Assign and PDO mapping
	Modified APPL_GenerateMapping function	Updated to correctly calculate process data size
	Modified APPL_InputMapping function	Updated to match changes in PDO Assign and PDO mapping
	Modified APPL_OutputMapping function	

	Modified APPL_Application function	Changed to call CPU1_ExchangeCtrl_MotorStatusRefresh at the beginning and CPU1_ExchangeCtrl_MotorRequestFlush at the end while EtherCAT input update is running (bEcatInputUpdateRunning == TRUE)
	Deleted APPL_GetDeviceID function	APPL_GetDeviceID is defined in samplecia402.c
cia402appl.h	Added macro definitions	See Table 7.44
	Changed macro definition values	Changed MAX_AXES to 4
	Added/modified Object Dictionary definitions	See Table 7.45
	Added members to TCiA402Axis structure	See Table 7.43
	Changed CiA402_LocalError function prototype	Added the target axis (TCiA402Axis *pCiA402Axis) to the arguments
ecatappl.c	Modified PDO_OutputMapping function	Replaced reading of RxPDO by HW_EscReadIsr function with ecat_esc_read_pdo function, which reads in 4-byte units
ecatcoe.h	Fixed structure alignment to 1	—
ecatslv.c	Modified StartInputHandler function	Removed processing that sets bits 2 and 3 of AL Event Mask Register to 1
mailbox.h	Fixed structure alignment to 1	—
sdoserv.h	Fixed structure alignment to 1	—

Table 7.43. Added Members in TCiA402Axis Structure (cia402appl.h)

Member Name	Role
u32AxisNumber	Member that stores the motor axis number
bCalibration	Calibration execution flag
u32LocalProfVel	Stores previous profile velocity value
u32LocalProfAcc	Stores previous profile acceleration value

Table 7.44. Added Macro Definitions (cia402appl.h)

Macro Name	Definition Value	Remarks
STATUSWORD_HOMING_MASK	0x2400	Statusword homing bit mask
STATUSWORD_HOMING_NOT_STARTED	0x0400	Homing not started
STATUSWORD_HOMING_COMPLETED	0x2000	Homing completed
CONTROLWORD_HOMING_START	0x0010	Homing start bit
HOMING_METHOD_37	0x0025	Homing Method 37
PP_MODULE_ID	0x00419800	ID for PP mode
HM_MODULE_ID	0x00519800	ID for HM mode
ENTRIES_1600	5	Number of mappings for PDO 0x1600
ENTRIES_1601	6	Number of mappings for PDO 0x1601
ENTRIES_1602	2	Number of mappings for PDO 0x1602
ENTRIES_1603	2	Number of mappings for PDO 0x1603
ENTRIES_1A00	5	Number of mappings for PDO 0x1A00
ENTRIES_1A01	5	Number of mappings for PDO 0x1A01
ENTRIES_1A02	5	Number of mappings for PDO 0x1A02
ENTRIES_1A03	5	Number of mappings for PDO 0x1A03

7.3.5.11 Object Dictionary

Table 7.45. Object Dictionary List

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
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

The Motor CPU I/F module uses the Shared Memory Driver to send motor control commands, update reference values, and retrieve motor status from the Motor CPU (RZ/T2N CPU0).

This sample program supports 4-axis motors by using four instances of this module.

7.3.6.1 Module Structure

The module structure diagram is shown in Figure 7.36.

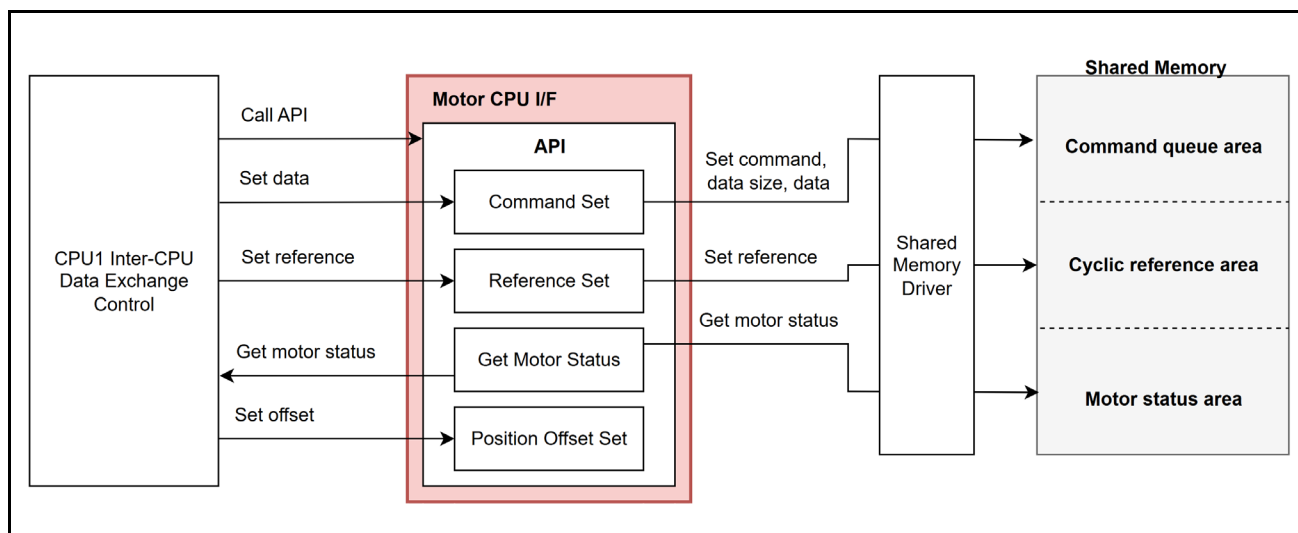


Figure 7.36. Motor CPU I/F Module Structure

This module mainly provides **four types of APIs**:

- (1) **Command Set**
Stores motor control commands, data size, and data in the **Command Queue Area** of Shared Memory.
- (2) **Reference Set**
Updates various reference values set in the **Cyclic Reference Area** of Shared Memory.
- (3) **Get Motor Status**
Retrieves various motor status values stored in the **Motor Status Area** of Shared Memory.
- (4) **Position Offset Set**
API prepared for setting the Home position. Sets an offset to position information.

7.3.6.2 Shared Memory Allocation

Same as the tables described in Section 7.2.3.2.

7.3.6.3 API

Table 7.46. Motor CPU I/F API List

API	Description
R_MOTOR_CPU_IF_Open	Initializes Motor CPU I/F and Shared Memory Driver. Flowchart: Figure 7.37
R_MOTOR_CPU_IF_Close	Terminates Motor CPU I/F and Shared Memory Driver. Flowchart: Figure 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: Stores motor control commands, data size, and data in the Command Queue Area of Shared Memory. Flowchart: Figure 7.39
R_MOTOR_CPU_IF_RefPositionSet R_MOTOR_CPU_IF_RefSpeedSet R_MOTOR_CPU_IF_RefTorqueSet	Reference Set API: Updates various reference values in the Cyclic Reference Area of Shared Memory. Flowchart: Figure 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: Retrieves various motor status values from the Motor Status Area of Shared Memory. Flowchart: Figure 7.41
R_MOTOR_CPU_IF_PositionOffsetSet	Position Offset Set API: Sets an offset to position information. Flowchart: Figure 7.42

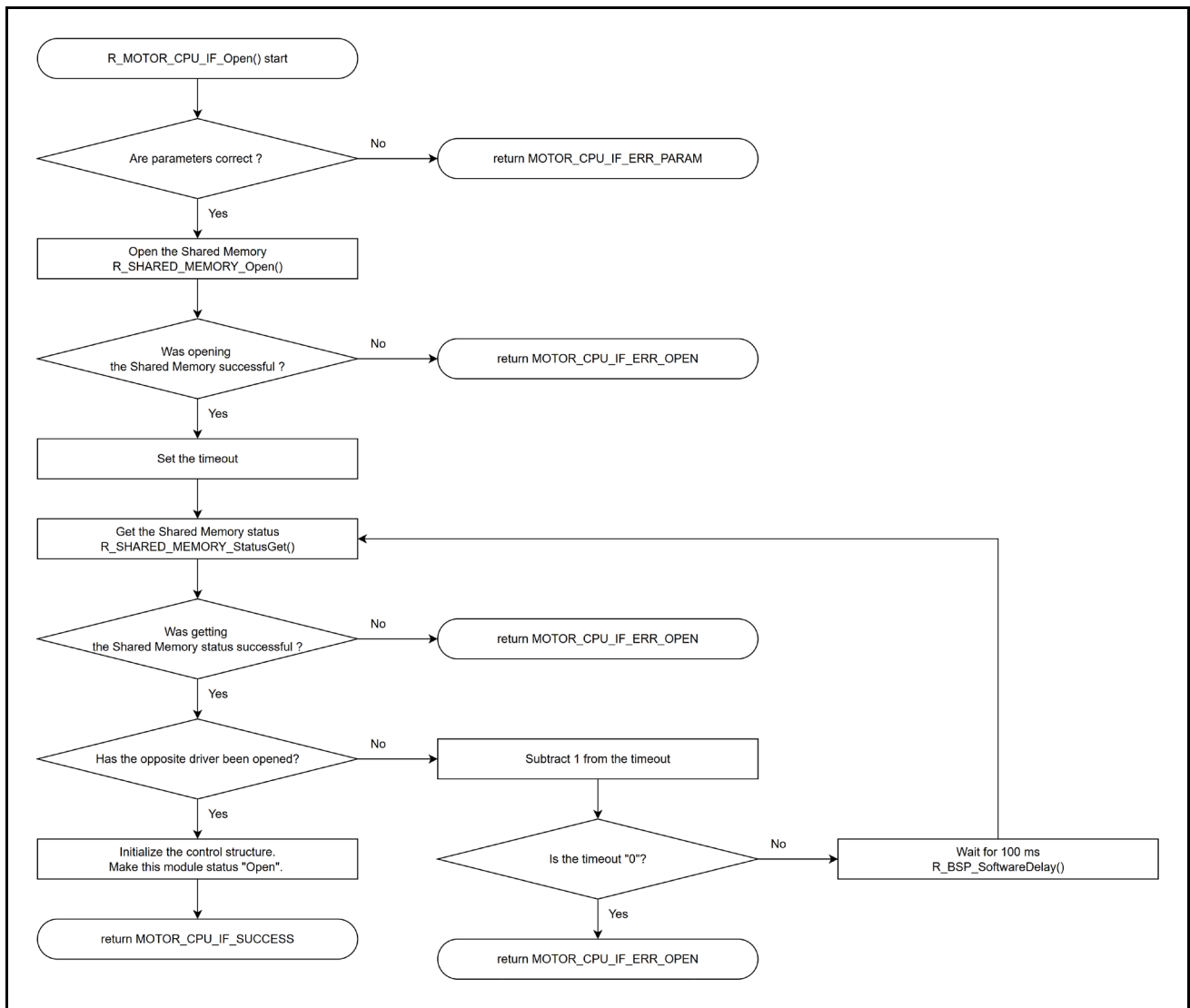


Figure 7.37. Flowchart of R_MOTOR_CPU_IF_Open Function

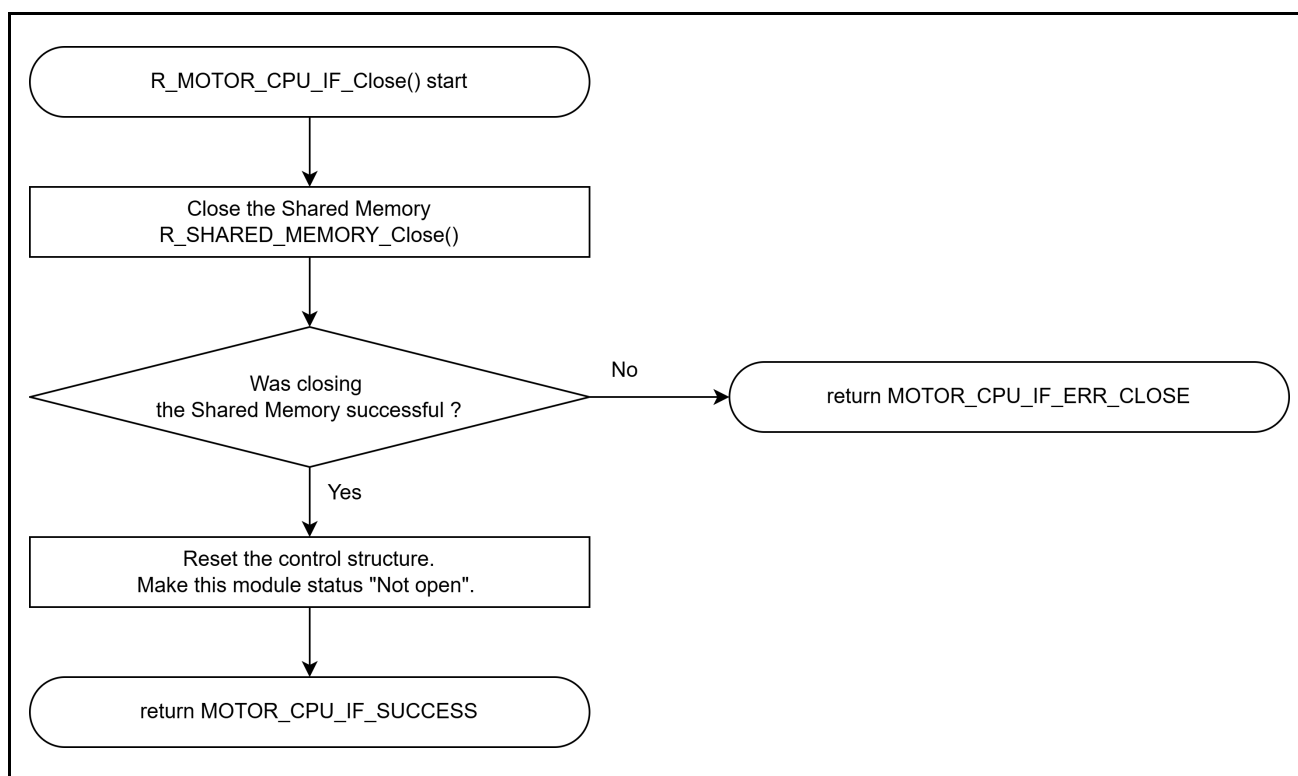


Figure 7.38. Flowchart of R_MOTOR_CPU_IF_Close Function

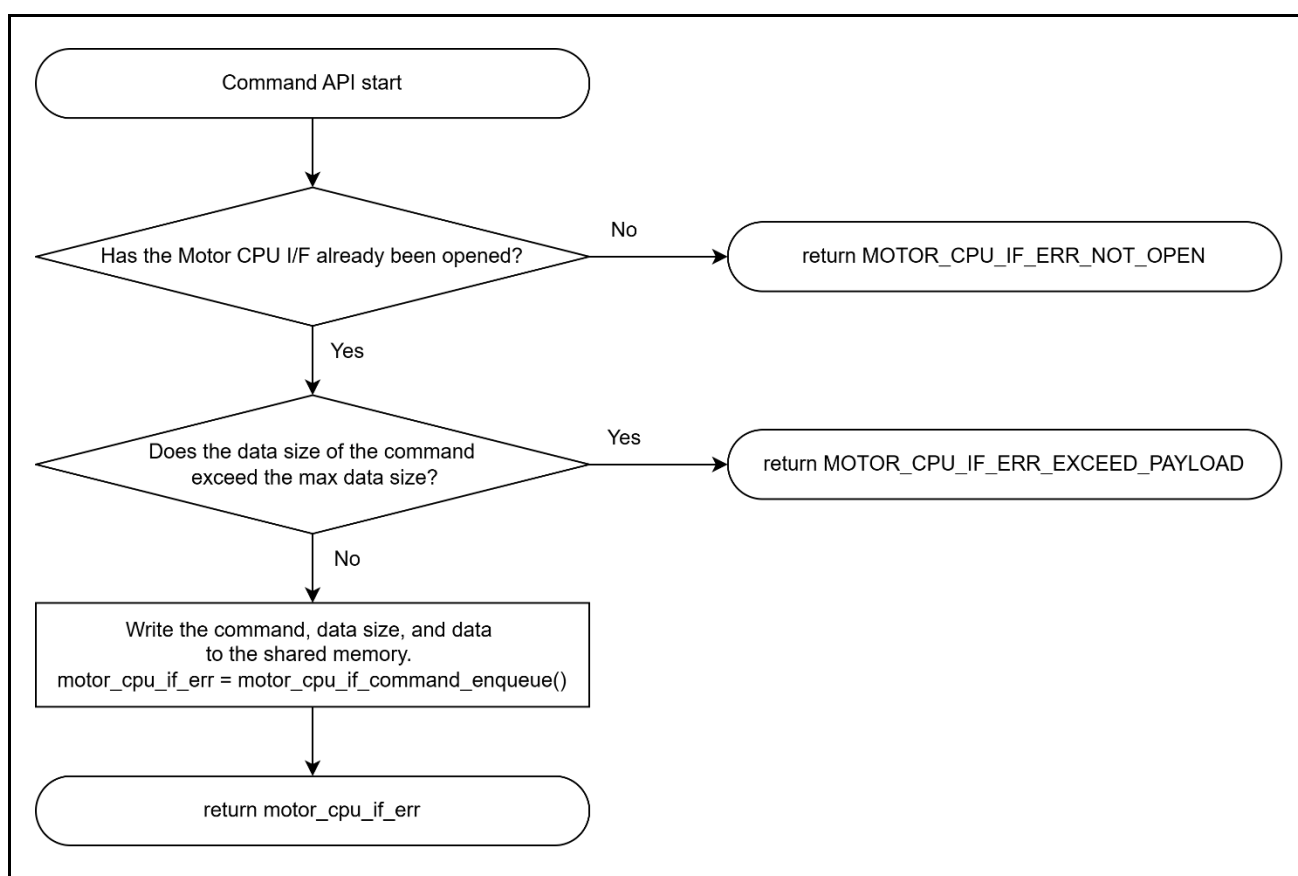


Figure 7.39. Flowchart of R_MOTOR_CPU_IF Command Set API

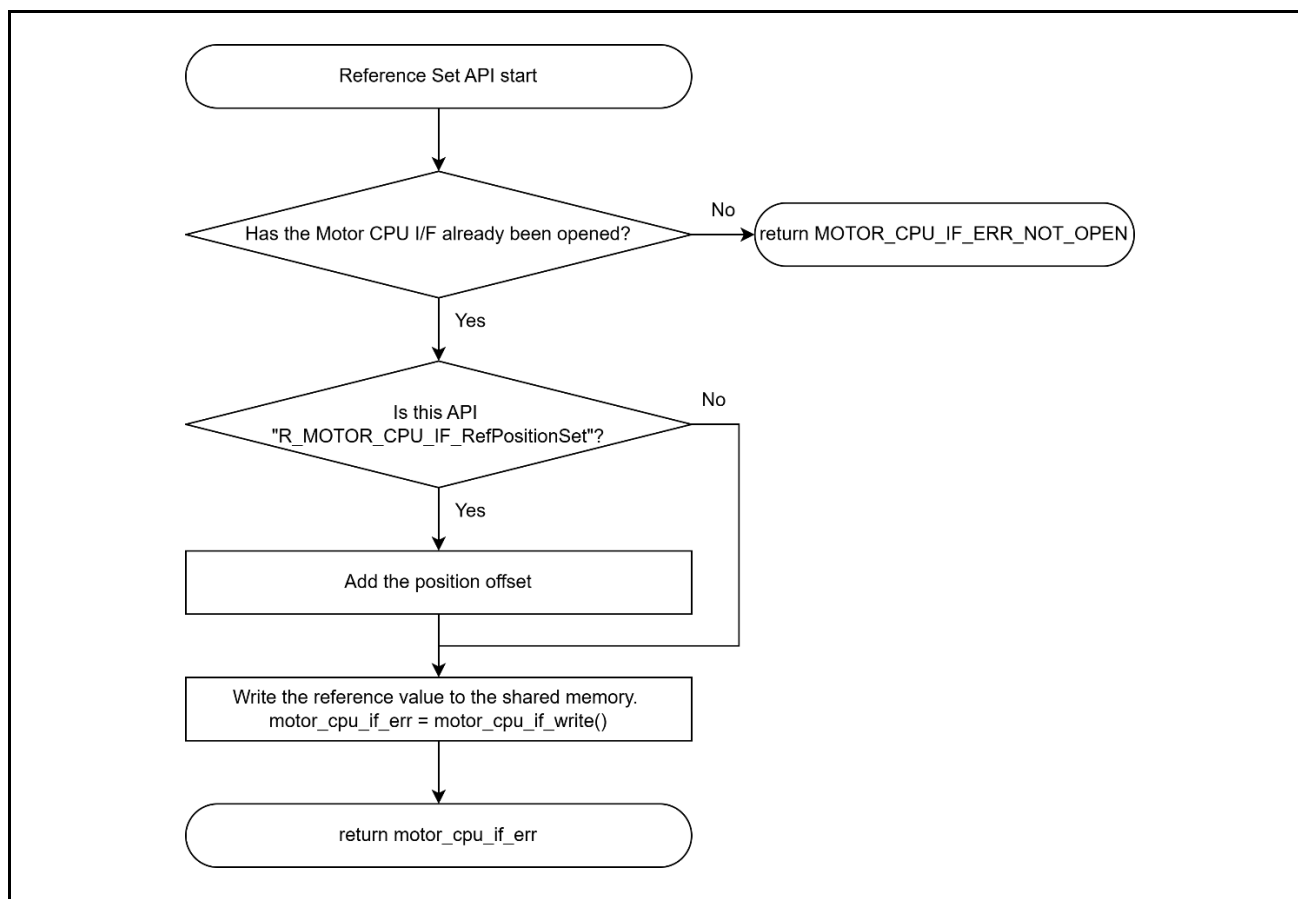


Figure 7.40. Flowchart of R_MOTOR_CPU_IF Reference Set API

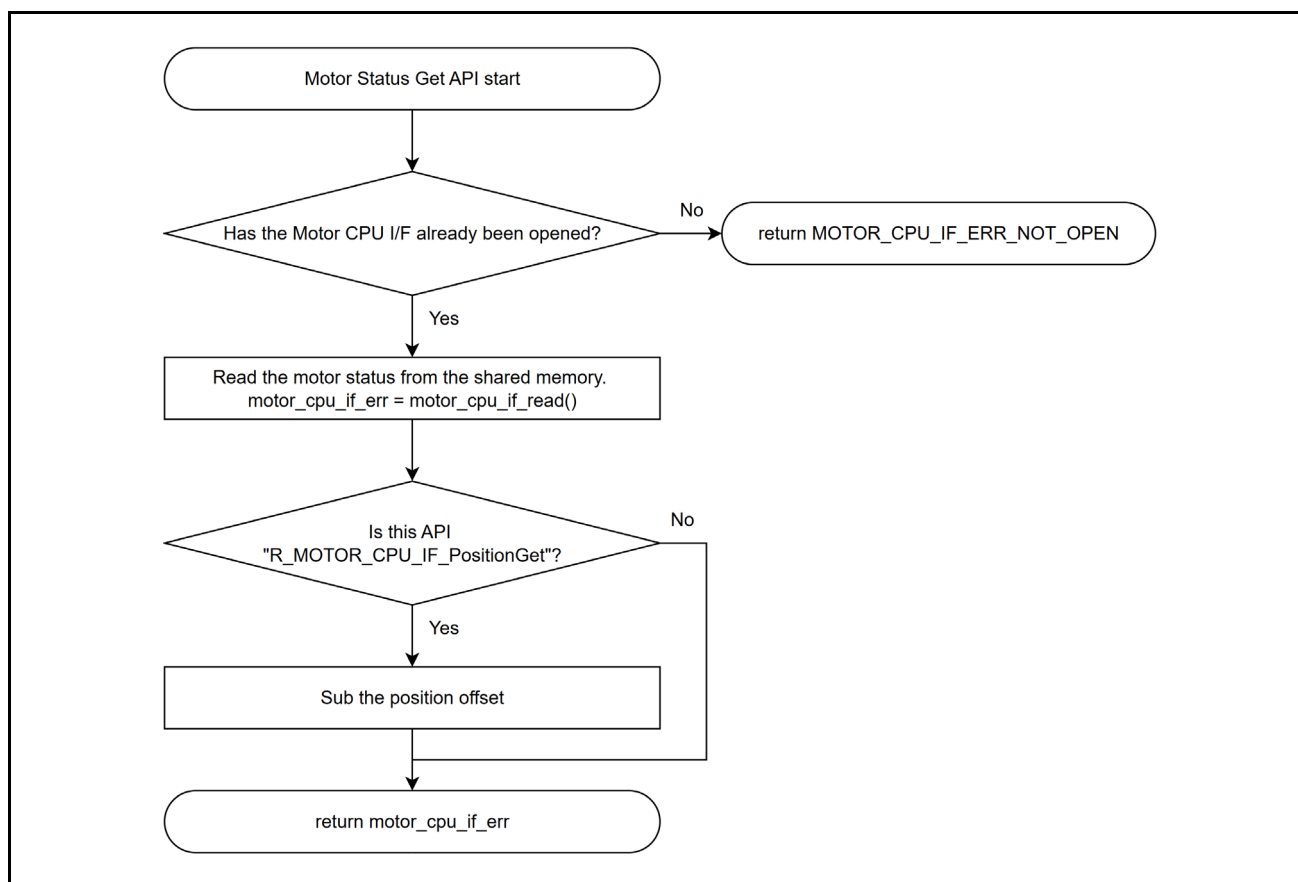
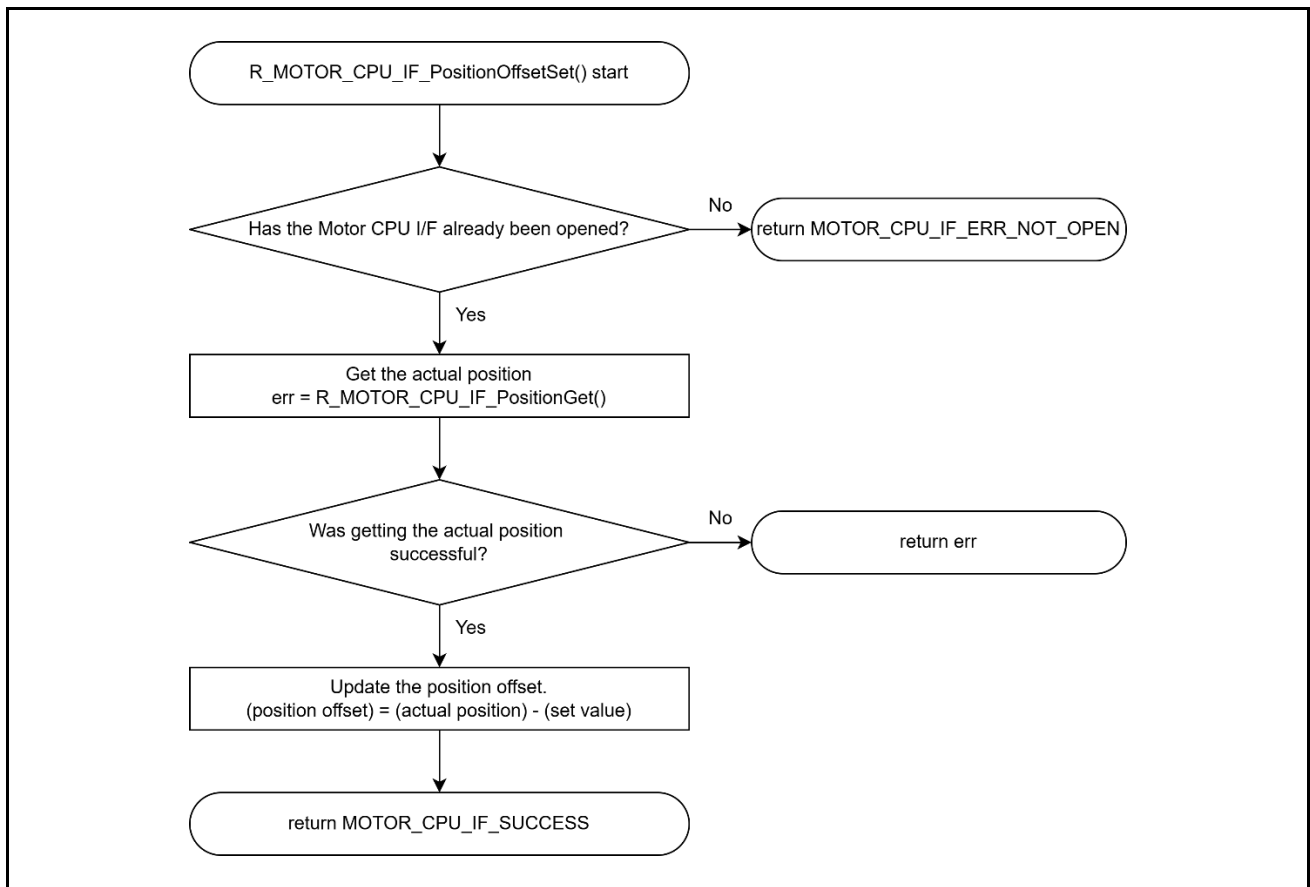


Figure 7.41. Flowchart of R_MOTOR_CPU_IF Motor Status Get API

**Figure 7.42. Flowchart of R_MOTOR_CPU_IF_PositionOffsetSet Function**

7.3.6.4 Private Functions

Table 7.47. Motor CPU I/F Private Functions

Private Function	Description
motor_cpu_if_write	Writes data to Shared Memory. Flowchart: Figure 7.43
motor_cpu_if_read	Reads data from Shared Memory. Flowchart: Figure 7.44
motor_cpu_if_command_enqueue	Writes commands to the Command Queue. Flowchart: Figure 7.45

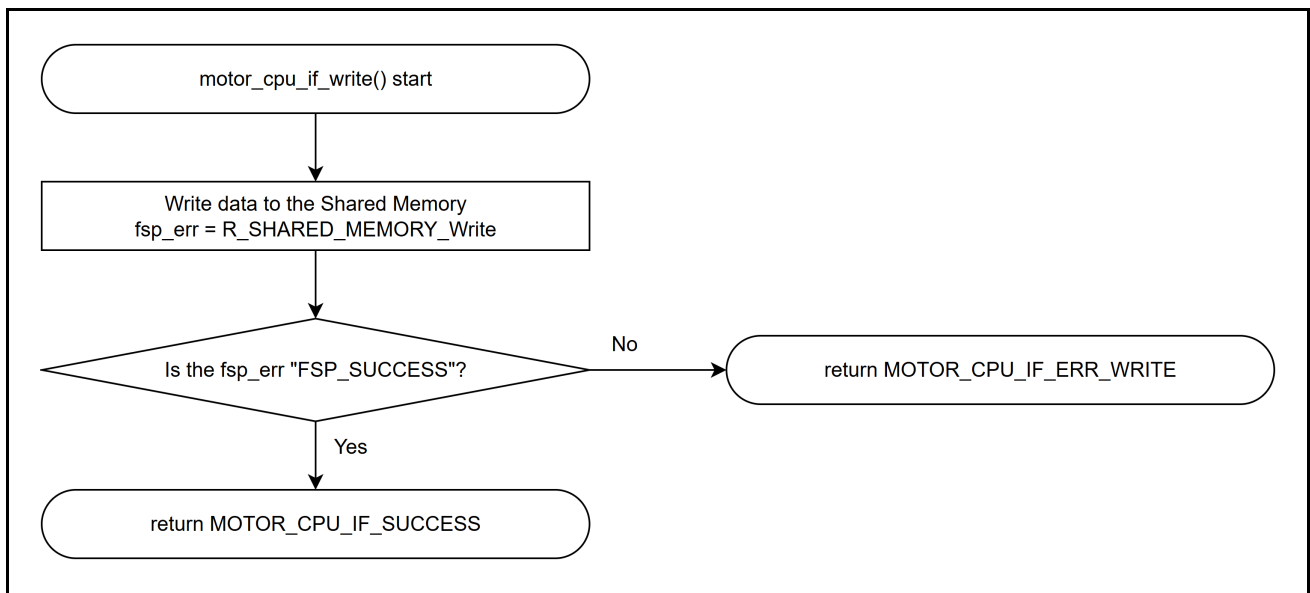


Figure 7.43. Flowchart of motor_cpu_if_write Function

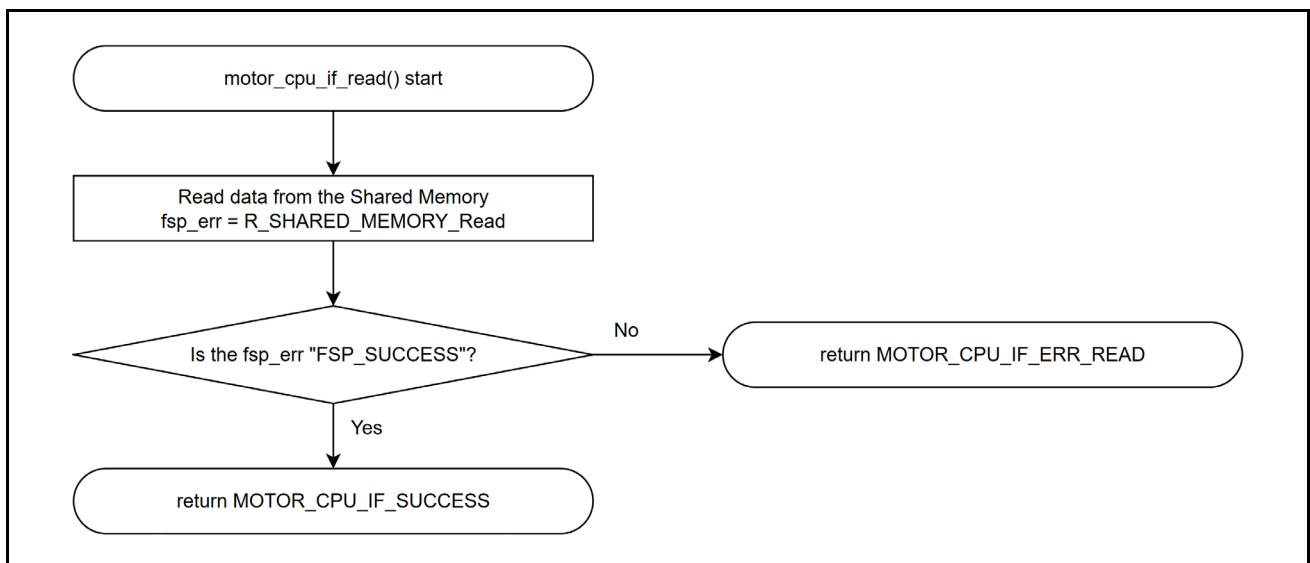
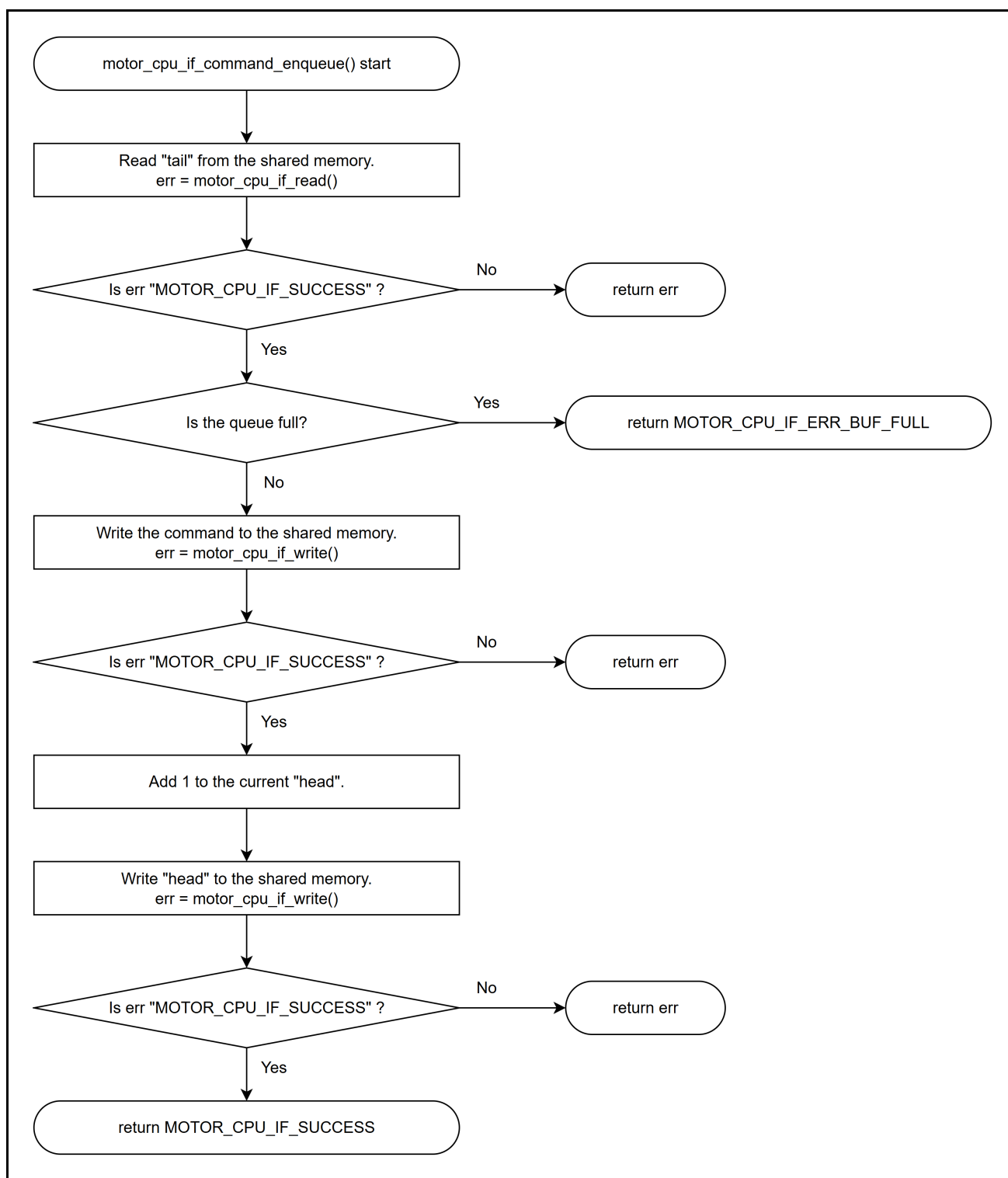


Figure 7.44. Flowchart of motor_cpu_if_read Function

**Figure 7.45. Flowchart of motor_cpu_if_command_enqueue Function**

7.3.6.5 Callback Function

Table 7.48. Motor CPU I/F Callback Functions (r_motor_cpu_if.c)

Function Name	Description
shared_memory_callback	Callback function for Shared Memory Driver. This interrupt occurs when RZ/T2N CPU0 writes to Shared Memory. No processing is performed inside the callback function. Consider adding processing if necessary.

7.3.6.6 Macro Definitions

Table 7.49. Motor CPU I/F Macro Definitions (r_motor_cpu_if.h)

Macro Name	Definition Value	Remarks
CFG_CMD_Q_NUM	8	Number of command queues (must be a power of 2)
CFG_CMD_Q_OFFSET	0x0000	Offset address of Command Queue Area
CFG_CYCLIC_REF_AREA_OFFSET	0x0100	Offset address of Cyclic Reference Area
CFG_MTR_STATUS_AREA_OFFSET	0x0180	Offset address of Motor Status Area
CFG_MOTOR_CPU_IF_TIMEOUT	100	Number of timeout counts while waiting for Shared memory to be ready at Open
CFG_SHARED_MEMORY_INSTANCE_ADDR0	&g_shared_memory0	Pointer to the Shared memory driver 0–3 instances
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	Maximum data size for the command queue

7.3.6.7 Enumerations

Table 7.50. Motor CPU I/F Enumerations (r_motor_cpu_if.h)

Enumerations	Definitions	Remarks
motor_cpu_if_err_t Motor CPU I/F Error Codes	MOTOR_CPU_IF_SUCCESS	No error
	MOTOR_CPU_IF_ERR_OPEN	Open error
	MOTOR_CPU_IF_ERR_CLOSE	Close error
	MOTOR_CPU_IF_ERR_PARAM	Parameter error
	MOTOR_CPU_IF_ERR_NOT_OPEN	Not open error
	MOTOR_CPU_IF_ERR_BUF_FULL	Command queue full
	MOTOR_CPU_IF_ERR_BUF_EMPTY	Command queue empty
	MOTOR_CPU_IF_ERR_WRITE	Write error
	MOTOR_CPU_IF_ERR_READ	Read error
	MOTOR_CPU_IF_ERR_EXCEED_PAYLOAD	Data exceeded

Enumerations	Definitions	Remarks
motor_cpu_if_status_t Motor CPU I/F Status	MOTOR_CPU_IF_STATUS_NOT_OPEN	Not open
	MOTOR_CPU_IF_STATUS_OPEN	Open
motor_cpu_if_cmd_t Motor CPU I/F Command Definitions	MOTOR_CPU_IF_CMD_NONE	No command
	MOTOR_CPU_IF_CMD_RESET	Used by Command Set API
	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 Position Control Mode	MOTOR_CPU_IF_POS_MODE_STEP	Step mode
	MOTOR_CPU_IF_POS_MODE_TRAPEZOID	Trapezoid mode
motor_cpu_if_ctrl_type_t Motor CPU I/F Motor Control Type	MOTOR_CPU_IF_CTRL_TYPE_POS	Position control
	MOTOR_CPU_IF_CTRL_TYPE_SPEED	Speed control
	MOTOR_CPU_IF_CTRL_TYPE_TORQUE	Torque control
	MOTOR_CPU_IF_CTRL_TYPE_VOLTAGE	Open-loop control
	MOTOR_CPU_IF_CTRL_TYPE_CURRENT	Current control
	MOTOR_CPU_IF_CTRL_TYPE_ADJ_CRNT_OFFSET	Adjust current offset
	MOTOR_CPU_IF_CTRL_TYPE_ADJ_POS_OFFSET	Adjust position offset
motor_cpu_if_motor_err_code_t Motor CPU I/F Motor Error Codes	MOTOR_CPU_IF_MOTOR_ERROR_NONE	No error
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_CURRENT_HW	HW overcurrent error
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_VOLTAGE	Overvoltage error
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_SPEED	Overspeed error
	MOTOR_CPU_IF_MOTOR_ERROR_UNDER_VOLTAGE	Undervoltage error

	MOTOR_CPU_IF_MOTOR_ERROR_OVER_CURRENT_SW	SW overcurrent error
	MOTOR_CPU_IF_MOTOR_ERROR_OVER_TEMPERATURE	Inverter overheat error
	MOTOR_CPU_IF_MOTOR_ERROR_UNKNOWN	Unknown error
motor_cpu_if_motor_state_t	MOTOR_CPU_IF_STATEMACHINE_STATE_STOP	STOP mode
Motor CPU I/F Motor Status	MOTOR_CPU_IF_STATEMACHINE_STATE_RUN	RUN mode
	MOTOR_CPU_IF_STATEMACHINE_STATE_ERROR	ERROR mode

7.3.6.8 Structures & Variable Information

Table 7.51. Motor CPU I/F Structure Definitions (r_motor_cpu_if.h)

Structure	Variable	Description
st_current_offset_cfg_t Current offset configuration structure	f4_iu_offset	U-phase current offset
	f4_iv_offset	V-phase current offset
	f4_iw_offset	W-phase current offset
st_current_open_loop_cfg_t Open loop current configuration structure	f4_open_loop_current	Open loop current value
	f4_ref_speed_rpm	Reference speed (RPM)
motor_cpu_if_cmd_hdr_t command queue head/tail structure	head	Head of command queue
	tail	Tail of command queue
motor_cpu_if_cmd_q_t command queue structure	cmd	Command
	size	Data size
	data[MOTOR_CPU_IF_CMD_DATA_MAX_BYTE]	Data array
motor_cpu_if_cyclic_ref_t Cyclic Reference structure	f4_ref_position	Reference position
	f4_ref_speed	Reference speed
	f4_ref_torque	Reference torque
motor_cpu_if_mtr_status_t Motor Status structure	f4_pos_deg	Current position
	f4_speed	Current speed
	u1_status	State machine
	reserved1	Unused area
	u2_error_status	Error status
	reserved2	Unused area
	u1_ctrl_loop_mode	LOOP MODE
	reserved3	Unused area
	u1_in_position	Position control completion status
	reserved4	Unused area

	u1_offset_cal_finished_flag	Offset measurement status
	reserved5	Unused area
motor_cpu_if_cfg_t Motor CPU I/F Configuration structure	p_shared_memory_instance	Pointer to Shared Memory instance
	timeout	Number of timeout counts while waiting for Shared memory to be ready at Open
	shr_mem_cmd_data_max_size	Maximum command data size
	shr_mem_cmd_q_num	Number of command queues
	shr_mem_cmd_q_mask	Command queue mask
	shr_mem_cmd_q_offset	Offset of Command Queue Area in Shared Memory
	shr_mem_cmd_q_size	Size of Command Queue Area in Shared Memory
	shr_mem_cmd_head_offset	Offset of command queue head area
	shr_mem_cmd_head_size	Size of command queue head area
	shr_mem_cmd_tail_offset	Offset of command queue tail area
	shr_mem_cmd_tail_size	Size of command queue tail area
	shr_mem_cyclic_ref_area_offset	Offset of Cyclic Reference Area
	shr_mem_cyclic_ref_area_size	Size of Cyclic Reference Area
	shr_mem_ref_pos_offset	Offset of reference position area
	shr_mem_ref_vel_offset	Offset of reference speed area
	shr_mem_ref_tor_offset	Offset of reference torque area
	shr_mem_mtr_status_area_offset	Offset of Motor Status Area
	shr_mem_mtr_status_area_size	Size of Motor Status Area
	shr_mem_mtr_pos_offset	Offset of current position area
	shr_mem_mtr_vel_offset	Offset of current speed area
	shr_mem_mtr_status_offset	Offset of state machine area
	shr_mem_mtr_err_status_offset	Offset of error status area
	shr_mem_mtr_ctrl_loop_mode_offset	Offset of LOOP MODE area
	shr_mem_mtr_in_pos_offset	Offset of position control completion area
	shr_mem_mtr_cal_finished_flag_offset	Offset of offset measurement status area
motor_cpu_if_ctrl_t Motor CPU I/F Control structure	status	Network CPU I/F status
	cmd_hdr	Instance of command queue head/tail structure
	position_offset	Position

	p_cfg	Pointer to Network CPU I/F configuration structure
motor_cpu_if_api_t Motor CPU I/F API structure	Open	Open function
	Close	Close function
	Reset	Command send function
	MotorStart	
	MotorStop	
	MotorReset	
	MotorErrorCancel	
	ErrorSet	
	<u>CtrlTypeSet</u>	
	PositionCommandModeSet	
	RotorAngleOffsetSet	
	CurrentOffsetSet	
	CurrentOpenLoopSet	
	<u>AccelTimeSet</u>	
	PosProfMaxSpeedRadSet	
	PositionOffsetSet	Position offset setting function
	RefPositionSet	Reference value set function
	RefSpeedSet	
	RefTorqueSet	
	PositionGet	Motor status get function
	SpeedGet	
	StatusGet	
	<u>ErrorStatusGet</u>	
	LoopModeStatusGet	
	InPositionFlagGet	
	OffsetCalibrationStatusGet	

7.3.6.9 Global Variables

Table 7.52. Motor CPU I/F Global Variables (r_motor_cpu_if.c)

Type	Instance Name	Description
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	Motor CPU I/F control structure instances for 4 motor axes. Initialized by the Open function.
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	Motor CPU I/F configuration structure instances for 4 motor axes. Each member is initialized at instance creation. Values cannot be changed after initialization.
motor_cpu_if_api_t	g_motor_cpu_if_api	Motor CPU I/F API structure instance. Each member is initialized at instance creation. Values cannot be changed after initialization.

7.3.7 CPU1 Inter-CPU Data Exchange Control Module

The CPU1 Inter-CPU Data Exchange Control module holds requests and reference values from the CiA402 Drive Profile in a local area, exchanges data through the Motor CPU I/F only in cycles in which CPU1 has the access right to Shared Memory, and then notifies CPU0 of the completion using INTCPU9.

7.3.7.1 Module Structure

The module structure diagram is shown in Figure 7.46.

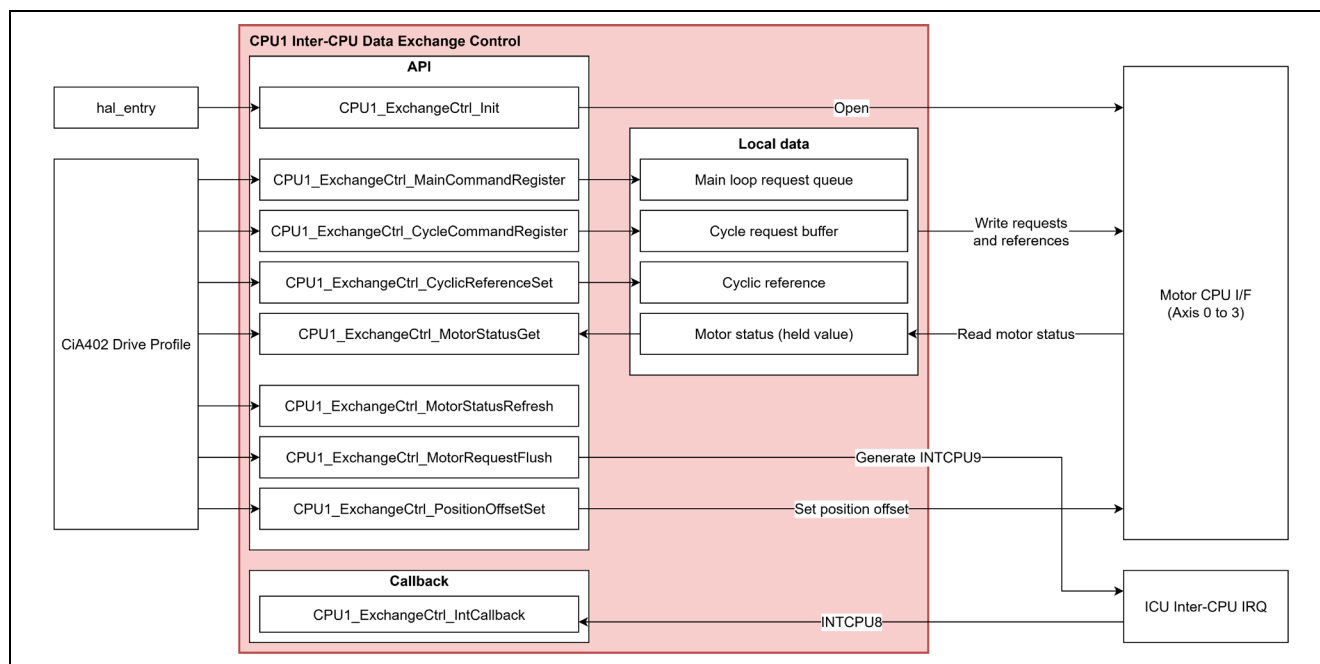


Figure 7.46. CPU1 Inter-CPU Data Exchange Control Module Structure

7.3.7.2 Inter-CPU Data Exchange Sequence

The data exchange sequence with CPU0 is shown in Figure 7.47.

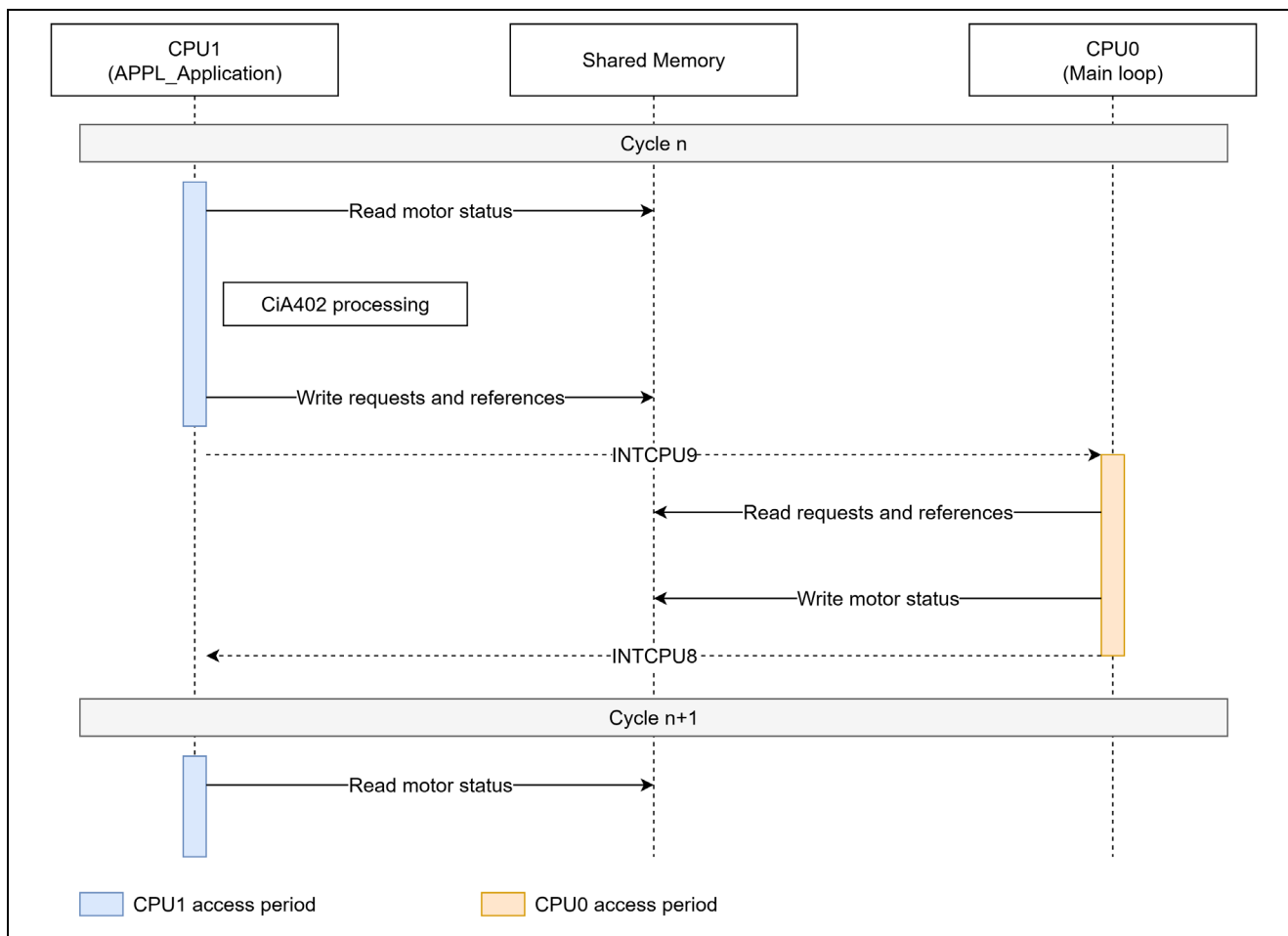


Figure 7.47. Inter-CPU Data Exchange Sequence

If CPU1 does not have the access right at the start of a cycle, CPU1 does not wait for the access right. It performs the cyclic processing with the held values and carries over the access to Shared Memory to the next cycle.

7.3.7.3 API

Table 7.53. CPU1 Inter-CPU Data Exchange Control API List (cpu1_exchange_ctrl.c)

API	Description
CPU1_ExchangeCtrl_Init	Initializes the CPU1 Inter-CPU Data Exchange Control module. Opens the Motor CPU I/F for 4 axes, and opens the Inter CPU IRQ for INTCPU8 reception and INTCPU9 transmission. The initial owner of the access right to Shared Memory is CPU1.
CPU1_ExchangeCtrl_MotorStatusRefresh	Called at the beginning of the APPL_Application function. If CPU1 has the access right, gets the Motor Status of 4 axes from the Motor CPU I/F and updates the held values. If not, keeps the held values without accessing Shared Memory.
CPU1_ExchangeCtrl_MotorRequestFlush	Called at the end of the APPL_Application function. If CPU1 has the access right, writes the main loop requests, cycle requests, and cyclic reference values to the Motor CPU I/F in this order for each axis. Then passes the access right to CPU0 and generates INTCPU9 to notify CPU0.
CPU1_ExchangeCtrl_MainCommandRegister	Called from the CPU1 main loop (CiA402 state transitions). Registers a command and data in the main loop request queue of each axis. Returns false if the axis number or data size is invalid, or if the queue is full.
CPU1_ExchangeCtrl_CycleCommandRegister	Called from the CPU1 cyclic processing. Registers a command and data in the cycle request buffer of each axis. Returns false if the axis number or data size is invalid, or if more than 4 requests are registered.
CPU1_ExchangeCtrl_CyclicReferenceSet	Called from the CPU1 cyclic processing. Sets the cyclic reference value (position, speed, or torque) in the holding area of each axis. Returns false if the axis number is invalid.
CPU1_ExchangeCtrl_MotorStatusGet	Copies the held Motor Status value of each axis without accessing Shared Memory. Returns false if the value has not been received or is being updated, or if the axis number is invalid.
CPU1_ExchangeCtrl_PositionOffsetSet	Called from the Homing mode processing. If CPU1 has the access right, executes the corresponding R_MOTOR_CPU_IF_PositionOffsetSet. Returns false if CPU1 does not have the access right, if the axis number is invalid, or if the execution fails.

7.3.7.4 Callback Function

Table 7.54. CPU1 Inter-CPU Data Exchange Control Callback Functions List (cpu1_exchange_ctrl.c)

Function Name	Description
CPU1_ExchangeCtrl_IntCallback	Callback function of INTCPU8 (r_icu_inter_cpu_irq ch8). Sets access_allowed to true. Does not access Shared Memory.

7.3.7.5 Macro Definitions

Table 7.55. CPU1 Inter-CPU Data Exchange Control Macro Definitions List (cpu1_exchange_ctrl.h)

Macro Name	Definition Value	Remarks
CPU1_EXCHANGE_CTRL_AXIS_NUM	4	Number of managed axes
CPU1_EXCHANGE_CTRL_MAIN_REQ_SLOT_NUM	4	Number of slots in the main loop request queue per axis (holds up to 3 requests)
CPU1_EXCHANGE_CTRL_MAIN_REQ_MAX	3	Maximum number of main loop requests per axis
CPU1_EXCHANGE_CTRL_CYCLE_REQ_MAX	4	Maximum number of cycle requests per axis

7.3.7.6 Enumerations

Table 7.56. CPU1 Inter-CPU Data Exchange Control Enumerations List (cpu1_exchange_ctrl.h)

Enumerations	Definitions	Remarks
cpu1_exchange_ctrl_ref_type_t Cyclic reference type	CPU1_EXCHANGE_CTRL_REF_POSITION	Position reference
	CPU1_EXCHANGE_CTRL_REF_SPEED	Speed reference
	CPU1_EXCHANGE_CTRL_REF_TORQUE	Torque reference

7.3.7.7 Structure and Variable Information

Table 7.57. CPU1 Inter-CPU Data Exchange Control Structure Definitions (cpu1_exchange_ctrl.h)

Structure	Variable	Description
cpu1_exchange_ctrl_main_req_queue_t Main loop request queue structure	motor_cpu_if_cmd_q[4]	Ring buffer of main loop requests
	head	Next write position (updated by the main loop)
	tail	Next read position (updated by the cyclic processing)
cpu1_exchange_ctrl_cycle_req_buffer_t Cycle request buffer structure	motor_cpu_if_cmd_q[4]	Command requests to be executed in the current cycle
	count	Number of registered requests (reset to 0 at the end of the cyclic processing)
cpu1_exchange_ctrl_cyclic_ref_t Cyclic reference structure	type	Cyclic reference type
	value	Cyclic reference value

	write_pending	Flag indicating that the cyclic reference value is waiting to be written
cpu1_exchange_ctrl_status_t Motor Status holding structure	sequence	Update sequence number (0: not received, odd: updating, even other than 0: update completed)
	status_value	Held Motor Status value
cpu1_exchange_ctrl_cfg_t Configuration structure	p_motor_cpu_if_ctrl[4]	Pointers to the Motor CPU I/F control structures for axes 0 to 3
	p_motor_cpu_if_cfg[4]	Pointers to the Motor CPU I/F configuration structures for axes 0 to 3
cpu1_exchange_ctrl_t Control structure	access_allowed	Flag indicating that CPU1 can access Shared Memory
	main_req_queue[4]	Main loop request queue of each axis
	cycle_req_buffer[4]	Cycle request buffer of each axis
	cyclic_ref[4]	Cyclic reference value of each axis
	motor_status[4]	Held Motor Status value of each axis
	p_cfg	Pointer to the configuration structure

7.3.7.8 Global Variable Information

Table 7.58. CPU1 Inter-CPU Data Exchange Control Global Variables (cpu1_exchange_ctrl.c)

Type	Instance Name	Description
cpu1_exchange_ctrl_t	g_cpu1_exchange_ctrl	Control structure instance of the CPU1 Inter-CPU Data Exchange Control. Zero-initialized; p_cfg is set and access_allowed is initialized (true) by the Init function.
cpu1_exchange_ctrl_cfg_t	g_cpu1_exchange_ctrl_cfg	Configuration structure instance of the CPU1 Inter-CPU Data Exchange Control. Initialized with the Motor CPU I/F control and configuration structure instances for 4 axes.

8. Appendix

8.1 Integrated development environment Installation

8.1.1 e² studio, FSP

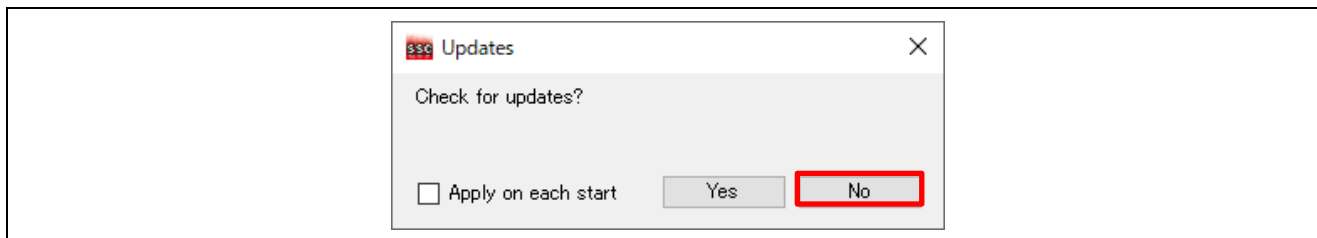
Download and install FSP v4.2.0 with the installer bundled with e2 studio (FSP Platform Installer) from the following URL.

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

8.2 Appendix : A point of caution when using SSC Tool

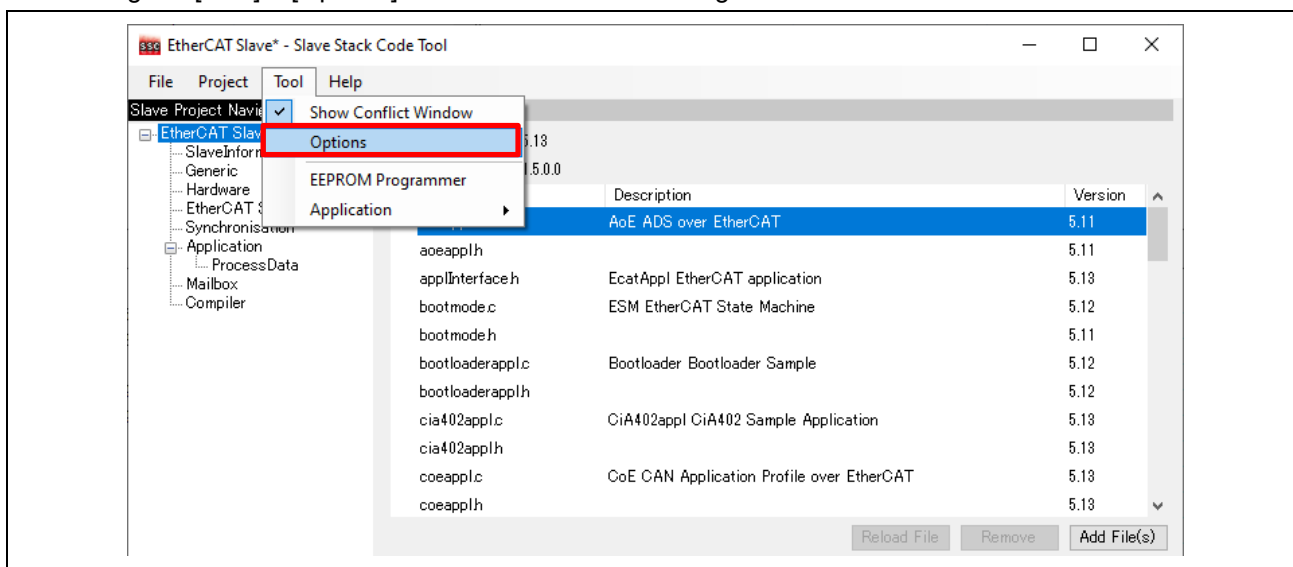
※ When opening SSC Tool

1. Please open the SSC Tool as an Administrator.
Otherwise, generating SSC code may fail.
2. When opening the SSC Tool for the first time, the following window may be displayed.
Please select [No] to not check for updates.

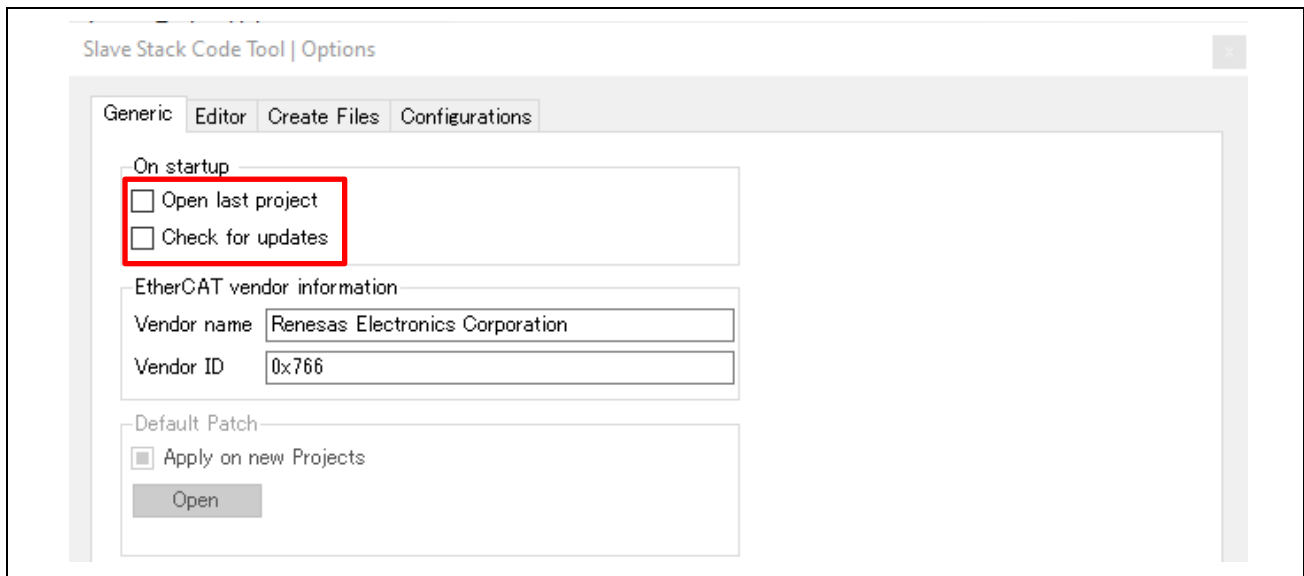


※ Regarding SSC Tool settings

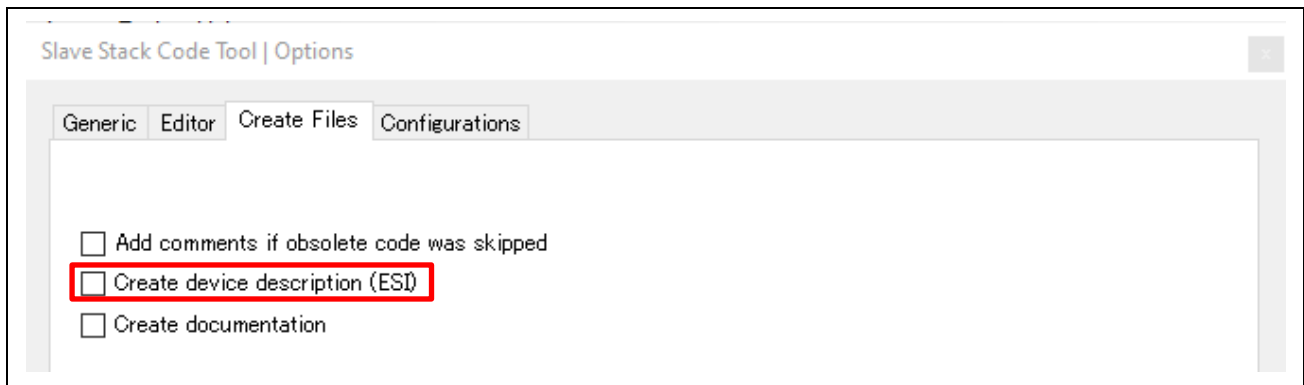
3. Please go to [Tool] > [Options] to confirm SSC Tool settings.



4. Under the [Generic] tab, uncheck [Open last project] and [Check for updates].



5. Under the [Create Files] tab, uncheck [Create device description (ESI)].



6. Click [OK] to apply the updated settings.
This concludes with the SSC Tool setup for this sample program.

8.3 Appendix : How to install patch

There are two methods as follows:

1. Via Git for Windows (64bit)
2. Via MinGW Installation Manager

8.3.1 Via Git for Windows (64bit)

This section describes how to install patch via Git for Windows.

1. Download the installer (e.g., Git-x.xx.x-64-bit.exe) from the official Git for Windows website.

[Git for Windows](#)

2. Run the downloaded installer and follow the setup instructions. Use the default settings unless you have specific requirements.
3. After installation, add the path to patch.exe to your system's environment variables.
For a default installation, the path is typically:

"C:\Users\<your-username>\AppData\Local\Programs\Git\usr\bin"

To apply the changes, restart your computer after updating the environment variables.

4. Start Command Prompt, enter "where patch".
If the path to patch.exe is displayed, the installation was successful.

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

8.3.2 Via MinGW Installation Manager

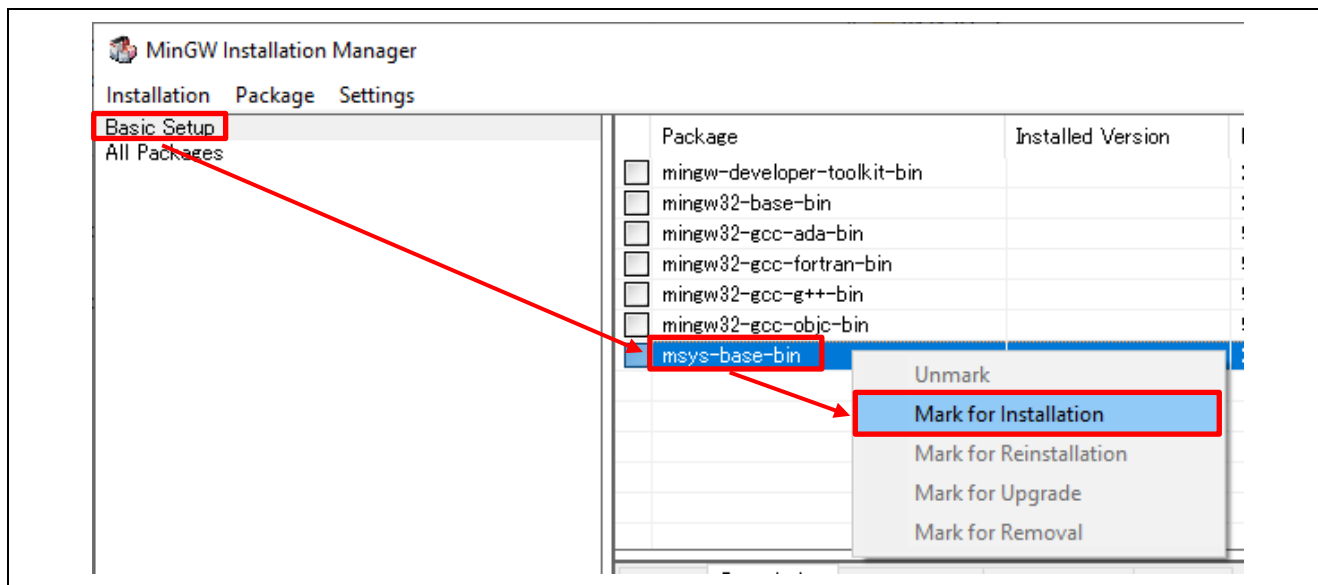
This section describes how to install patch via MinGW Installation Manager.

1. Download “mingw-get-setup.exe” from the following URL.

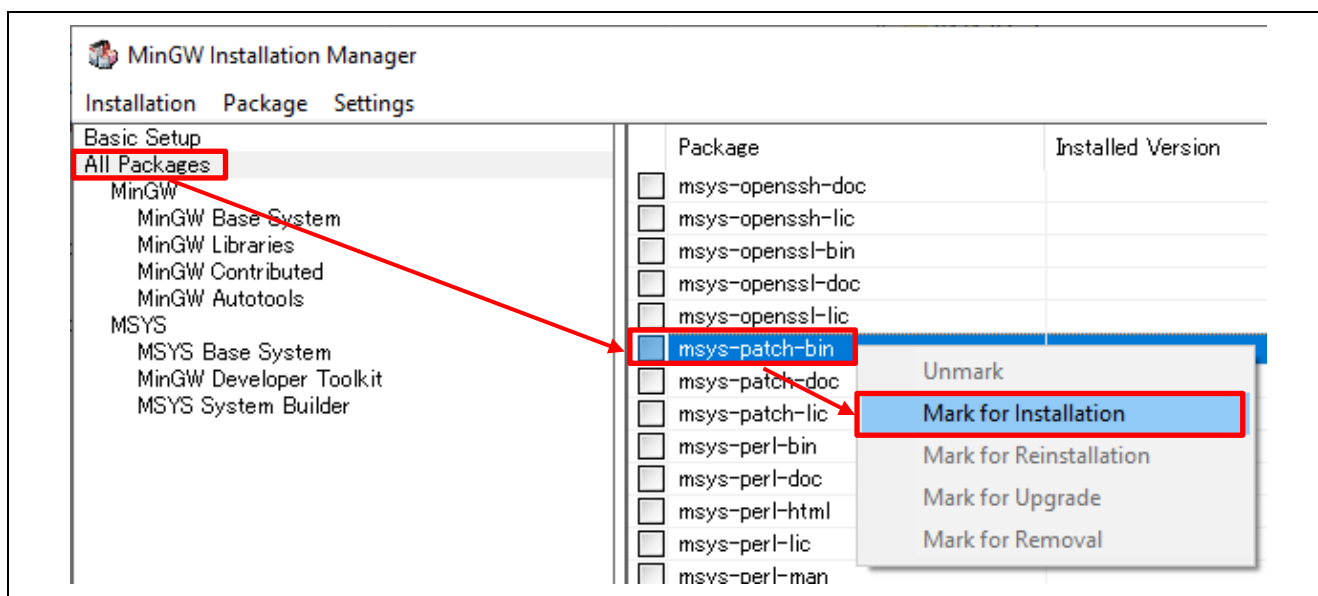
<https://sourceforge.net/projects/mingw/>

2. Execute “mingw-get-setup.exe”, install “Mingw-installation-manager” according to the dialog.

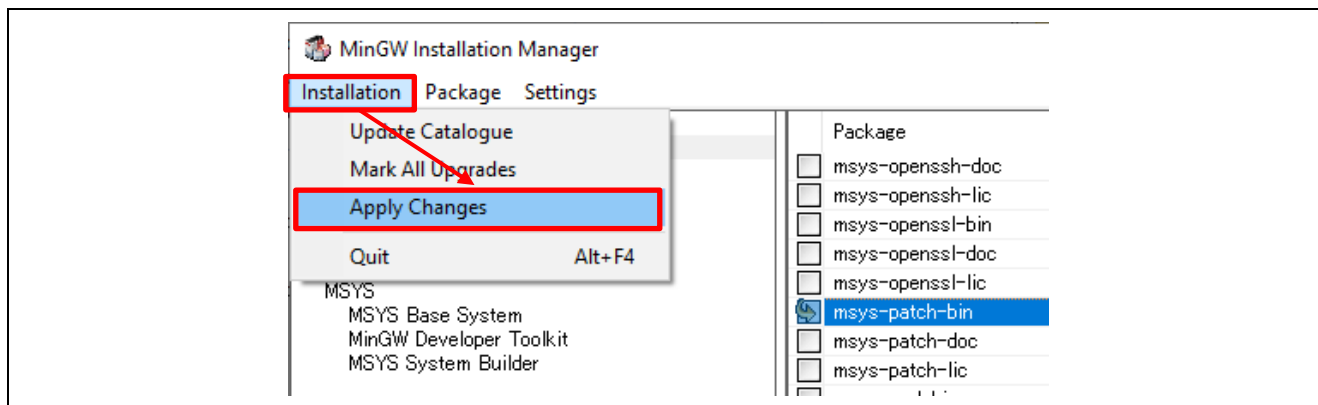
3. If It's completed and the Mingw-installation-manager window is displayed, select “Basic Setup” in the left window, right-click on “msys-base-bin” in the right window, and select “Mark for Installation”.



4. Select “All Packages” in the left window, right-click on “msys-patch-bin” in the right window, and select “Mark for Installation”.



5. Select “Apply Changes” in “Installation” in the above menubar.

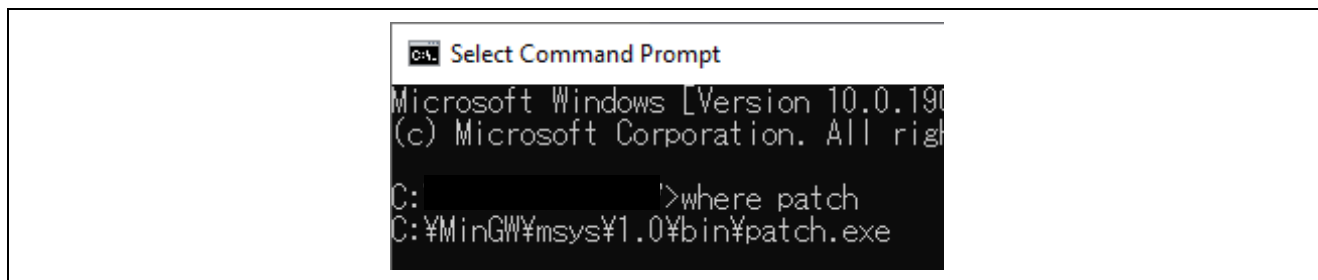


6. “Schedule of Pending Actions” window is displayed, click “Apply” button.
7. If “All changes were applied successfully; you may now clone this dialogue.” is displayed, Installing patch.exe is succeeded.
8. Add the path to the installed patch.exe to the system environment variables.
For example, add the following path in the case of default path.

“C:\MinGW\msys\1.0\bin”

After updating the system environment variables, restart your computer to apply the changes.

9. Start Command Prompt, enter “where patch”.
If the path to the installed patch.exe is displayed, there are no problem.



Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Sep.30, 2026	-	First Issued

General Precautions in the Handling of Microprocessing Unit and Microcontroller Unit Products

The following usage notes are applicable to all Microprocessing unit and Microcontroller unit products from Renesas. For detailed usage notes on the products covered by this document, refer to the relevant sections of the document as well as any technical updates that have been issued for the products.

1. Precaution against Electrostatic Discharge (ESD)

A strong electrical field, when exposed to a CMOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop the generation of static electricity as much as possible, and quickly dissipate it when it occurs. Environmental control must be adequate. When it is dry, a humidifier should be used. This is recommended to avoid using insulators that can easily build up static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work benches and floors must be grounded. The operator must also be grounded using a wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions must be taken for printed circuit boards with mounted semiconductor devices.

2. Processing at power-on

The state of the product is undefined at the time when power is supplied. The states of internal circuits in the LSI are indeterminate and the states of register settings and pins are undefined at the time when power is supplied. In a finished product where the reset signal is applied to the external reset pin, the states of pins are not guaranteed from the time when power is supplied until the reset process is completed. In a similar way, the states of pins in a product that is reset by an on-chip power-on reset function are not guaranteed from the time when power is supplied until the power reaches the level at which resetting is specified.

3. Input of signal during power-off state

Do not input signals or an I/O pull-up power supply while the device is powered off. The current injection that results from input of such a signal or I/O pull-up power supply may cause malfunction and the abnormal current that passes in the device at this time may cause degradation of internal elements. Follow the guideline for input signal during power-off state as described in your product documentation.

4. Handling of unused pins

Handle unused pins in accordance with the directions given under handling of unused pins in the manual. The input pins of CMOS products are generally in the high-impedance state. In operation with an unused pin in the open-circuit state, extra electromagnetic noise is induced in the vicinity of the LSI, an associated shoot-through current flows internally, and malfunctions occur due to the false recognition of the pin state as an input signal become possible.

5. Clock signals

After applying a reset, only release the reset line after the operating clock signal becomes stable. When switching the clock signal during program execution, wait until the target clock signal is stabilized. When the clock signal is generated with an external resonator or from an external oscillator during a reset, ensure that the reset line is only released after full stabilization of the clock signal. Additionally, when switching to a clock signal produced with an external resonator or by an external oscillator while program execution is in progress, wait until the target clock signal is stable.

6. Voltage application waveform at input pin

Waveform distortion due to input noise or a reflected wave may cause malfunction. If the input of the CMOS device stays in the area between V_{IL} (Max.) and V_{IH} (Min.) due to noise, for example, the device may malfunction. Take care to prevent chattering noise from entering the device when the input level is fixed, and also in the transition period when the input level passes through the area between V_{IL} (Max.) and V_{IH} (Min.).

7. Prohibition of access to reserved addresses

Access to reserved addresses is prohibited. The reserved addresses are provided for possible future expansion of functions. Do not access these addresses as the correct operation of the LSI is not guaranteed.

- Arm and Cortex are registered trademarks of Arm Limited (or its subsidiaries) in the EU and other countries. All rights reserved.
- IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers Inc
- EtherCAT® and TwinCAT® are registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany.
- FA-CODER is a registered trademark of Tamagawa Seiki Co., Ltd.
- Additionally all product names and service names in this document are a trademark or a registered trademark which belongs to the respective owners.

Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall be responsible for determining what licenses are required from any third parties, and obtaining such licenses for the lawful import, export, manufacture, sales, utilization, distribution or other disposal of any products incorporating Renesas Electronics products, if required.
5. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
6. Renesas Electronics products are classified according to the following two quality grades: "Standard" and "High Quality". The intended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below.

"Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

"High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user's manual or other Renesas Electronics document.

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

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

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

(Rev.5.0-1 October 2020)

Corporate Headquarters

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

Trademarks

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

Contact information

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