RX72M グループ

EtherCAT 通信を用いた永久磁石同期モータのエンコーダベクトル制御

R01AN6295JJ0110
Rev.1.10
2023/11/30

要旨

本アプリケーションノートでは、永久磁石同期モータ(以降、PMSM と表記)のエンコーダベクトル制御機能と EtherCAT 通信機能を RX72M に実装したサンプルプログラムについて説明します。本モジュールは産業イーサネット通信用 EtherCAT スレーブコントローラ（EtherCAT Slave Controller：ESC）を内蔵した RX ファミリで Beckhoff 社製 EtherCAT スレーブスタックコード(Slave Stack Code：SSC)を使用するためのインタフェースを提供します。
本モジュールには SSC は含まれておりません。EtherCAT Technology Group(ETG 協会)より SSC ツールを入手の上、コードを生成してください。
以降、本モジュールを EtherCAT FIT モジュールと称します。

対象デバイス

- RX72M グループ

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

目次ー

内容

1. 概要

1.1 本アプリケーションノートについて

本アプリケーションノートでは、永久磁石同期モータ(以降、PMSM と表記)のエンコーダベクトル制御機能と EtherCAT 通信機能を RX72M に実装したサンプルプログラムについて説明します。
サンプルプログラムは、RX72M CPU カードとインバータボードの組み合わせで動作します。

1.2 動作環境

表 1-1 動作環境

対応 MCU	RX72M グループ
評価ボード	ルネサスエレクトロニクス製 RX72M CPU カード+インバータボード※
統合開発環境（IDE）	ルネサスエレクトロニクス製 e2 studio 2023-10
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package for RX Family V3.05.00
モータ	BLDC モータ　：　BLY171D-24V-4000 (Anaheim Automation 社製) エンコーダ　：　AMT102-V (CUI DEVICES 社製)
エミュレータ	ルネサスエレクトロニクス製 e2 Lite
通信プロトコル	EtherCAT
SSC Tool	EtherCAT Technology Group (ETG) 提供 Slave Stack Code (SSC) Tool Version 5.13
ソフトウェア PLC	Beckhoff Automation 製 TwinCAT® 3 (Beckhoff web サイトからダウンロード)

※ルネサスエレクトロニクス製「Evaluation System for BLDC Motor (RTK0EMX270S00020BJ)」に同梱されています。

1.3　プロジェクトについて

サンプルプログラムはモータ制御または EtherCAT 通信、個別のプロジェクトをベースに幾つかの変更点を加えることで 2 つの機能をマージした EtherCAT 通信によるシングルチップモータ制御を実現しています。

表 1-2 ベースプロジェクトと変更点

機能/プロジェクト名 （アプリケーションノート）	変更点
モータ制御 RX72M_ESB_SPM_ENCD_FOC_E2S_V100	● EtherCAT 通信プログラムからモータ制御を行うための API 関数の追加 ● CiA402 のオブジェクトの仕様に合わせた位置や速度の単位変換
EtherCAT 通信 rx72m_com_cia402 (r01an4672jj0104-rx72m-ecat)	● CiA402 ドライブプロファイルに合わせたオブジェクトの追加 ● モータ制御を行うための API 関数の呼び出し

サンプルプログラムに含まれるプロジェクトを示します。
以降の章では RX72M CPU カード+インバータボードのプロジェクトを例にして説明します。異なるプロジェクトを使用する場合は適宜、プロジェクト名を置き換えて、お読みください。

表 1-3 プロジェクト一覧

MCU	評価ボード名	プロジェクト名
RX72M	RX72M CPUカード+インバータボード	rx72m_ecat_cia402_bldc_encd

2. システム概要

2.1 ハードウェア構成

サンプルプログラムのハードウェア構成を次に示します。

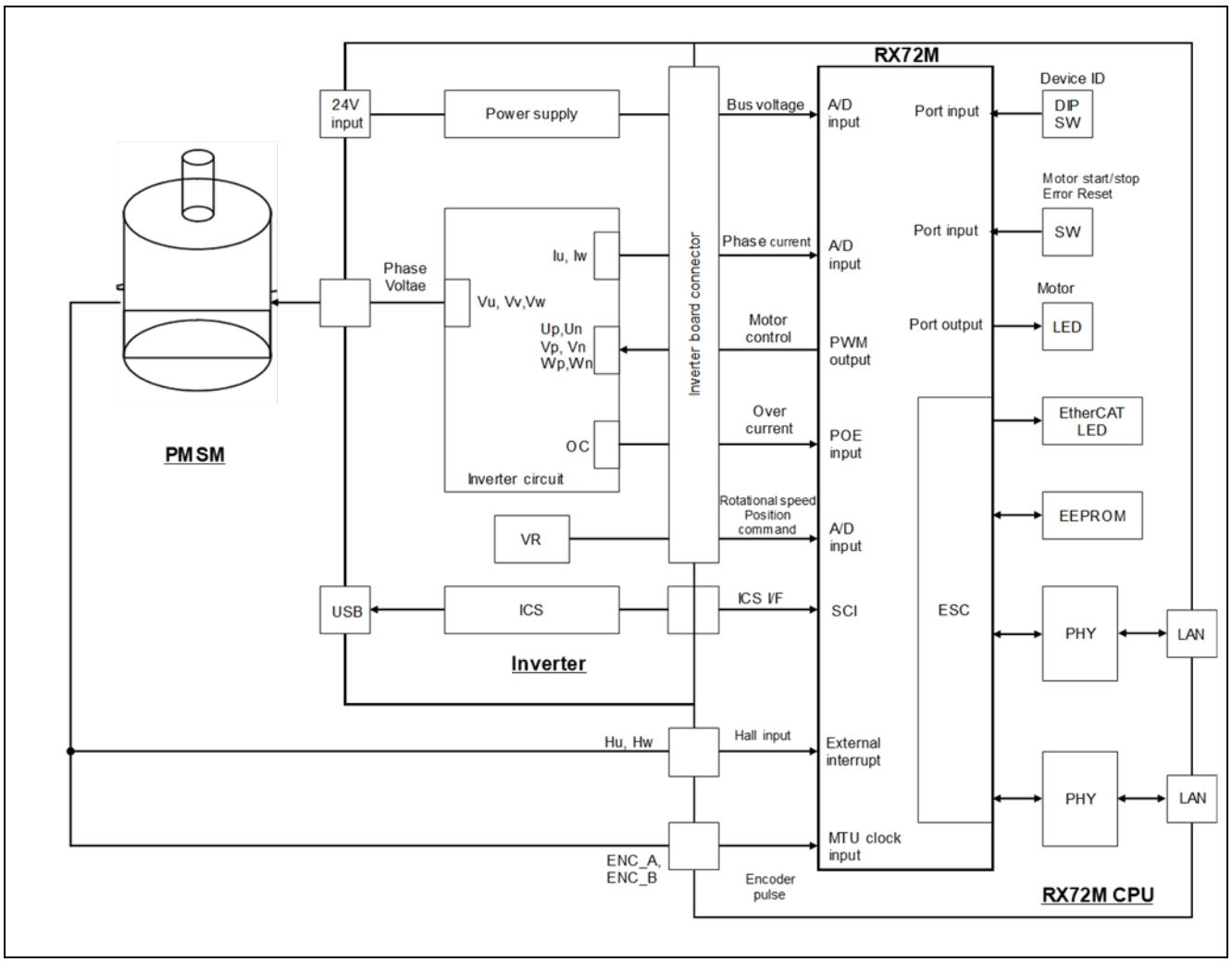


図 2-1 ハードウェア構成図

2.2　ハードウェア仕様

サンプルプログラムで使用する端子インタフェースを表 2-1 から表 2-4 に示します。

表 2-1 モータ制御関連端子インタフェース

端子名	機　能
P40 /AN000	U 相電流測定
P42 /AN002	W 相電流測定
P23/GTIOC0A	PWM 出力（Up）
P17/GTIOC0B	PWM 出力（Un）
P22/GTIOC1A	PWM 出力（Vp）
P87/GTIOC1B	PWM 出力（Vn）
P21/GTIOC2A	PWM 出力（Wp）
P86/GTIOC2B	PWM 出力（Wn）
P00/AN118	インバータ母線電圧測定
P33 / IRQ3	ホール U 相入力
P32 / IRQ2	ホール V 相入力
P31 / IRQ1	ホール W 相入力
P24 /MTCLKA	エンコーダ A 相入力
P25 /MTCLKB	エンコーダB 相入力
PC4/GTETRGC	過電流検出時の PWM 緊急停止入力
PC5	SW1 入力
PC3	SW2 入力
P01/AN119	VR 入力
P80	LED1 制御
PK2	LED2 制御
P76	LED3 制御

表 2-2 EtherCAT 通信関連端子インタフェース(1)

端子名	機能
PK6/CATLINKACT0	Link/Activity LED 制御出力
PK7/CATLINKACT1	Link/Activity LED 制御出力
PH1/CATI2CCLK	EEPROM I2C クロック出力
P15/CATLEDRUN	RUN LED (緑色 LED)制御出力
PH3/CATLEDERR	ERR LED (赤色 LED)制御出力
PL3/CAT0_RX_CLK	受信クロック入力
PM4/CAT0_ETXD2	4 ビットの送信データ出力(bit2)
PM5/CAT0_ETXD3	4 ビットの送信データ出力(bit3)
PL4/CAT0_ETXD0	4 ビットの送信データ出力(bit0)
PL5/CAT0_ETXD1	4 ビットの送信データ出力(bit1)
PK5/CAT0_ERXD3	4 ビットの受信データ入力(bit3)
PK4/CAT0_ERXD2	4 ビットの受信データ入力(bit2)
P74/CAT0_ERXD1	4 ビットの受信データ入力(bit1)
P75/CAT0_ERXD0	4 ビットの受信データ入力(bit0)
PL7/CAT0_MDIO	マネジメントデータ I/O 入出力
PN3/CAT1_RX_ER	受信エラー入力
P84/CAT1_LINKSTA	PHY-LSI からのリンクステータス入力
PQ2/CAT1_RX_DV	受信データ有効入力
PL6/CAT0_TX_EN	送信イネーブル出力
PN2/CAT1_TX_CLK	送信クロック入力
PH4/CATLEDSTER	STATE LED (2 色 LED)用の RUN LED 制御(ERR 時消灯)出力
PH5/CATLATCH0	LATCH 信号入力
PH6/CATLATCH1	LATCH 信号入力
PA4/CATIRQ	IRQ 出力
PQ7/CAT1_TX_EN	送信イネーブル出力
PC2/CAT0_RX_DV	受信データ有効入力
PM1/CAT1_ERXD1	4 ビットの受信データ入力(bit1)
PM2/CAT1_ERXD2	4 ビットの受信データ入力(bit2)
PM3/CAT1_ERXD3	4 ビットの受信データ入力(bit3)
PL2/CAT0_RX_ER	受信エラー入力
PM0/CAT1_ERXD0	4 ビットの受信データ入力(bit0)
PQ4/CAT1_RX_CLK	受信クロック入力
PJ5/CATSYNC0	SYNC0 信号出力
PA6/CATRESTOUT	PHY リセット信号出力

表 2-3 EtherCAT 通信関連端子インタフェース(2)

端子名	機能
PN1/CAT1_ETXD3	4 ビットの送信データ出力(bit3)
PQ5/CAT1_ETXD0	4 ビットの送信データ出力(bit0)
PN0/CAT1_ETXD2	4 ビットの送信データ出力(bit2)
PQ6/CAT1_ETXD1	4 ビットの送信データ出力(bit1)
P11/CATSYNC1	SYNC1 信号出力
PM6/CAT0_TX_CLK	送信クロック入力
PK0/CAT0_MDC	マネジメントデータクロック出力
P34/CAT0_LINKSTA	PHY-LSI からのリンクステータス入力
PH2/CATI2CDATA	EEPROM I2C データ入出力
PH1/CATI2CCLK	EEPROM I2C クロック入出力

表 2-4 その他端子インタフェース

端子名	機能
P72	デバイス ID DIP SW(bit3)
PC1	デバイス ID DIP SW(bit4)
PN5	デバイス ID DIP SW(bit5)

2.3　ソフトウェア構成

2.3.1　　ソフトウェア・ファイル構成

サンプルプログラムの EtherCAT 通信に関わるフォルダとファイル構成を表 2-5 に示します。
網掛けされたファイルはサンプルプログラムの機能を実現するためにベースプロジェクトから変更があったことを示し、太字はファイルが追加されたことを示します。

表 2-5 ソフトウェア・ファイル構成

フォルダ	サブフォルダ		ファイル	内容
ecat	application	renesas	cia402sample.h cia402sample.c	CiA402 アプリケーション定義
		beckhoff/ Src	SSC ソースファイル	修正用バッチファイルを適用すると格納される
	interface		**r_mtr_driver_ecat_access.h** **r_mtr_driver_ecat_access.c**	EtherCAT 通信プログラムアクセス関数定義
app	cfg		r_app_control_cfg.h	APP_CFG_USE_UI を MAIN_UI_ETHERCAT に変更
	main		r_app_main.h r_app_main.c	EtherCAT プロトコルスタック呼び出しを追加
motor_module	cfg		r_motor_module_cfg.h	制御ループを位置制御に設定

2.3.2　ソフトウェア・モジュール構成

EtherCAT 通信でモータ制御を行うモジュールは Application Layer と Middle Layer に位置します。
各 Layer でのモジュールの構成と役割について説明します。

表 2-6 ソフトウェア・モジュール構成

階層 / モジュール	関連ファイル	役割の説明
Application Layer / EtherCAT Application	cia402sample.h cia402sample.c	EtherCAT マスタからの指示でモータ制御を行う。またモータの位置情報などを EtherCAT マスタに通知する。EtherCAT インタフェースモジュールを介してモータ制御プログラムにデータを受け渡す。
Middle Layer / EtherCAT Interface module	r_mtr_dirver_ecat_acces.h r_mtr_dirver_ecat_acces.c	EtherCAT 通信プログラムが呼び出すモータ制御に関わる API を提供する。モータ制御プログラムは BLDC 向けまたはステッピングモータ向けに対応する。

EtherCAT 通信プログラムのモジュール構成を図 2-3 に示します。

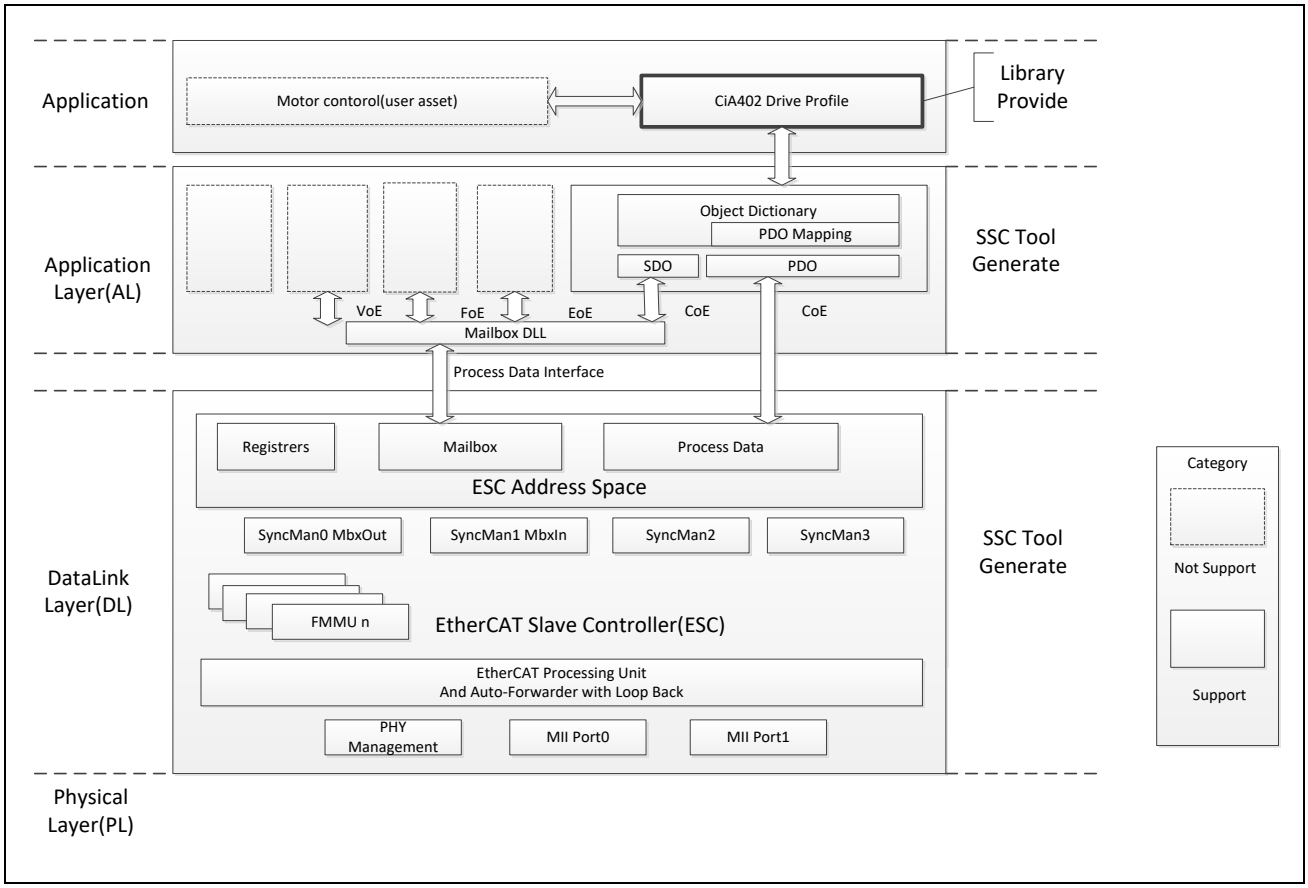


図 2-2 EtherCAT 通信プログラムモジュール構成

2.4　ソフトウェア仕様

サンプルプログラムのソフトウェアの基本仕様を下記に示します。

表 2-7 モータ制御プログラム基本仕様

項　目	内　容
制御方式	ベクトル制御
回転子磁極位置検出	インクリメンタルエンコーダ（A相、B相）、ホールセンサ（UVW相）
入力電圧	DC 24V
キャリア周波数(PWM)	20 [kHz](キャリア周期：50 [μs])
デッドタイム	2 [μs]
制御周期(電流)	50 [μs]
制御周期(速度・位置)	500 [μs]
位置指令値範囲	位置指令値の作成：速度台形波方式による位置プロファイル （入力範囲） -32768°～32767° （速度制限） CW / CCW：-4000~4000 [rpm]
速度指令範囲	CW：0[rpm] ～ 4000[rpm] CCW：0[rpm] ～ 4000[rpm]
位置分解能	0.09°（エンコーダパルス：1000 [p/r]、4逓倍時4000 [cpr]）
位置の不感帯[注1]	±1カウント
各制御系固有周波数	電流制御系：300Hz 速度制御系：30Hz 位置制御系：10Hz
コンパイラ最適化設定	最適化レベル：2 (-optimize = 2)（デフォルト設定） 最適化方法：コード・サイズ重視の最適化 (-size)（デフォルト設定）
保護停止処理	以下のいずれかの条件の時、モータ制御信号出力（6 本）を非アクティブにする 1. 各相の電流が 3.82 [A]を超過（50 [μs]毎に監視） 2. インバータ母線電圧が 28 [V]を超過（50 [μs]毎に監視） 3. インバータ母線電圧が 14 [V]未満（50 [μs]毎に監視） 4. 回転速度が 4500 [rpm]を超過（50 [μs]毎に監視） 5. ホールセンサのパターンエラー(始動時) 外部からの過電流検出信号（POE/POEG）及び出力短絡を検出した場合、PWM出力端子をハイインピーダンスにする。

【注】位置決め時のハンチング等を防ぐため、不感帯を設けています。

表 2-8 EtherCAT 通信プログラムの基本仕様

項目	内容
物理層	100BASE-TX (IEEE802.3)
ボーレート	100[Mbps] (Full duplex)
通信ポート数	2 ポート
EtherCAT LED	RUN、ERR、STAT、L/A IN、L/A OUT
ステーション ID	デバイス ID DIP SW (6bit)により設定
Explicit Device ID	対応
デバイスプロファイル	CiA402 デバイスプロファイル
Sync Manager	4
FMMU	3
通信オブジェクト	SDO (Service Data Object) PDO (Process Data Object)
同期モード	SM2 イベント同期モード DC モード
プロトコルスタックの提供形態	サンプルプログラム向けの SSC Tool プロジェクトファイルを提供。また CiA402 アプリケーションなどの追加分のパッチを提供。SSC Tool にてプロトコルスタックコードを生成後、パッチを適用することで EtherCAT 通信プログラムが出来る。

3. CiA402 ドライブプロファイル

CiA402 ドライブプロファイルはドライブおよびモーションコントロール用のデバイスプロファイルであり、主にサーボドライブ、正弦波インバータ、およびステッピングモータ用コントローラの機能動作を定義します。このプロファイルでは、複数の動作モードと対応する設定パラメータがオブジェクトディクショナリとして規定されます。また、状態ごとの内部および外部動作を規定する有限状態オートマトン（Finite State Automaton: FSA）も含まれます。状態を変更する場合はコントロールワードオブジェクトを通じて指定することで、現在の状態を示すステータスワードオブジェクトに遷移後の結果が反映されます。コントロールワードと各種コマンド値（速度など）は RxPDO に割り当てられ、ステータスワードと各種実査値（位置など）は TxPDO に割り当てられます。詳細については CiA402 規格書の内容を確認してください（リファレンス(１)）。

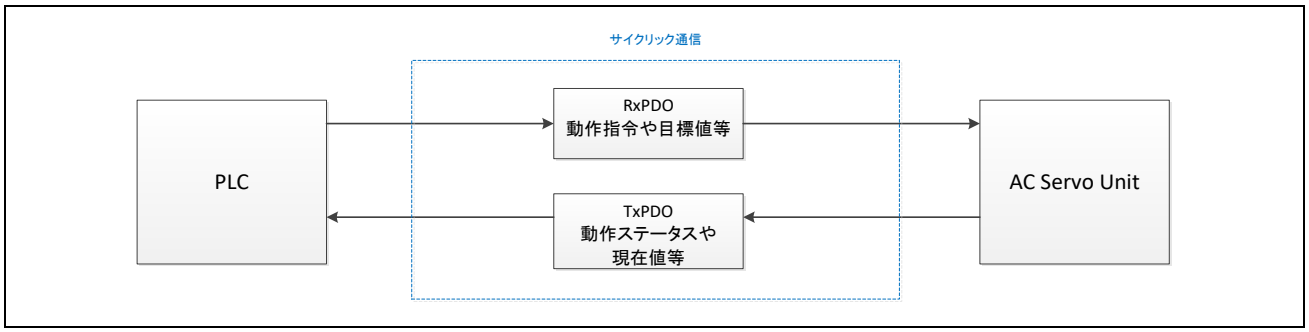


図 3-1 CiA402 通信の流れ

3.1 動作モード

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

表 3-1 サポート動作モード一覧

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

3.2　状態遷移

CiA402 として規定されている FSA として、サンプルプログラムでは以下をサポートします。
図 3-2 のうちモータにトルクがかかっている状態は"Operation enabled"になります。”Switched on”から"Operation enabled"に遷移する契機(遷移 4)でモータをアクティブにします。また、"Operation enabled"から他の状態に遷移する契機(遷移 5、8、9)ではモータを非アクティブにします。ただし、"Operation enabled"から"Quick stop active"に遷移する契機(遷移 11) 、または"Fault reaction active"に遷移する契機(遷移 13)ではトルクがかかっている状態を維持します。

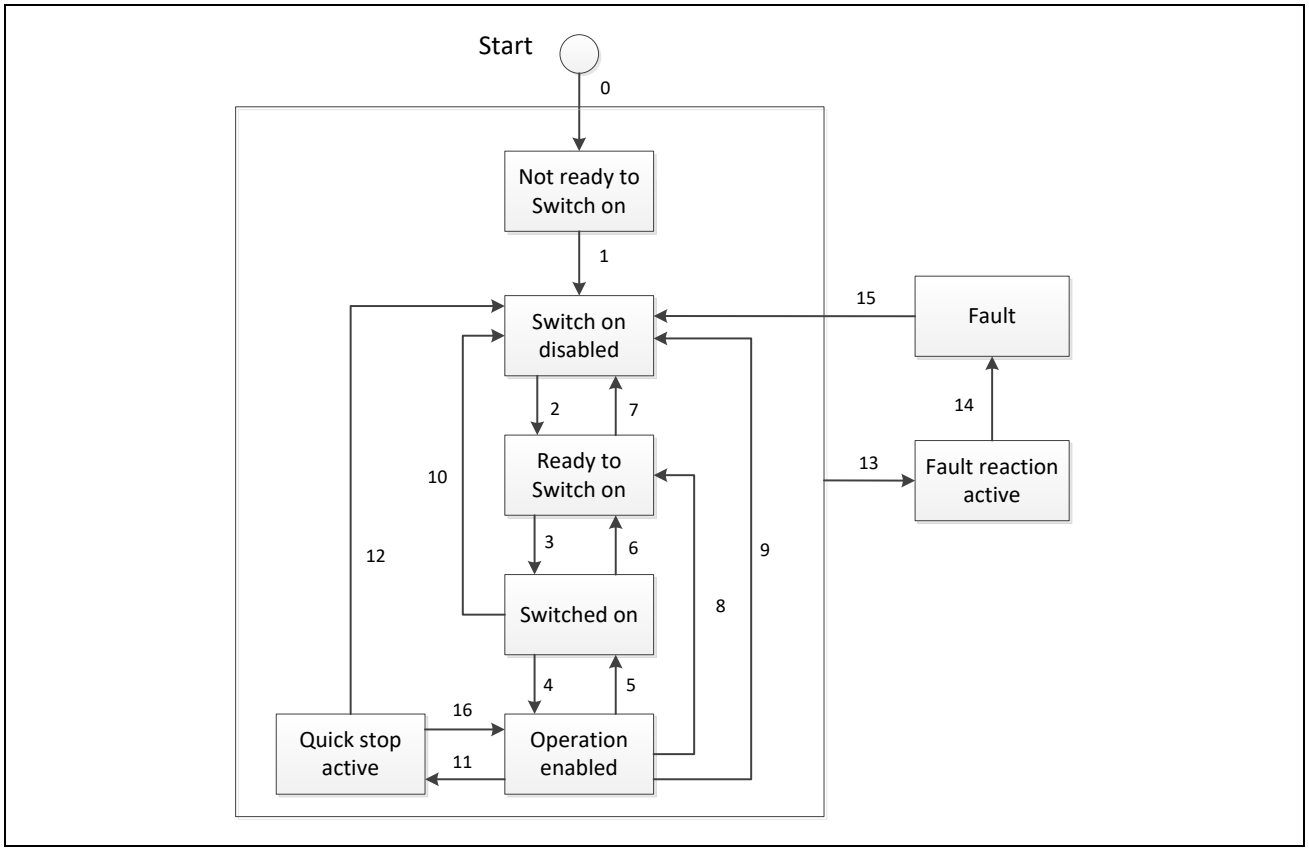


図 3-2 CiA402 状態遷移図

3.3　状態遷移関数

CiA402 状態遷移関数一覧を表 3-2 に示します。各関数は図 3-2 に示す CiA402 FSA の各状態遷移の番号とリンクしており、状態遷移が発生した際に対応する関数が呼び出されます。

表 3-2 CiA402 状態遷移関数一覧

Transition No.	function name
1	CiA402_StateTransition1
2	CiA402_StateTransition2
3	CiA402_StateTransition3
4	CiA402_StateTransition4
5	CiA402_StateTransition5
6	CiA402_StateTransition6
7	CiA402_StateTransition7
8	CiA402_StateTransition8
9	CiA402_StateTransition9
10	CiA402_StateTransition10
11	CiA402_StateTransition11
12	CiA402_StateTransition12
13	CiA402_LocalError
14	CiA402_StateTransition14
15	CiA402_StateTransition15
16	CiA402_StateTransition16

CiA402 状態遷移関数の関数仕様を示します。

CiA402_StateTransition(N)

CiA402FSA にて規定されている状態遷移(N)=1~12、14~16 が発生した際に使用します。
状態遷移が発生した際に実行する処理を記述してください。

Format

UINT16 CiA402_StateTransition(N)(TCiA402Axis *pCiA402Axis)

Parameters

TCiA402Axis *pCiA402Axis

Return Values

0：正常終了
1：状態遷移しない

Properties

cia402appl.h にプロトタイプ宣言されています。

Description

処理中に異常が発生した場合は CiA402 規格に従って、各オブジェクトに適切な値を設定して関数を終了してください。戻り値に 1 を設定した場合には状態遷移は行われません。

Example

```
TCiA402Axis *pCiA402Axis;
UINT16 retval ;

/* Transition1 */
retval = CiA402_StateTransition1 (pCiA402Axis);
```

CiA402_LocalError

CiA402 ドライブプロファイルで規定されたエラーを検出したときに使用します。本関数を実行後に
CiA402FSA にて規定されている状態遷移 13 が発生します。
エラーを検出した際に実行する処理を記述してください。

Format

void CiA402_LocalError(UINT16 ErrorCode)

Parameters

UINT16 ErrorCode : CiA402 ドライブプロファイルエラーコード

Return Values

なし

Properties

cia402appl.h にプロトタイプ宣言されています。

Description

引数として指定したエラーコードはオブジェクト 0x603F に格納され EtherCAT マスタに通知されます。

Example

```
/* Over speed error is detected */
CiA402_LocalError (ERROR_SPEED);
```

3.4　オブジェクトディクショナリ

サンプルプログラムでサポートしているオブジェクトディクショナリの一覧を以下に示します。

表 3-3 サポートオブジェクトディクショナリー覧

INDEX	Sub	OBJECT Name	Category	Access	DataType	PDO Mapping
0x603F		Error code	Optional	ro	UINT16	TxPDO
0x6040		Controlword	Mandatory(all)	rw	UINT16	RxPDO
0x6041		Statusword	Mandatory(all)	ro	UINT16	TxPDO
0x605A		Quick stop option code	Optional	rw	INT16	No
0x605B		Shutdown option code	Optional	rw	INT16	No
0x605C		Disable operation option code	Optional	rw	INT16	No
0x605E		Fault reaction option code	Optional	rw	INT16	No
0x6060		Modes of operation	Mandatory(all)	rw	INT8	No
0x6061		Modes of operation display	Mandatory(all)	ro	INT8	TxPDO
0x6064		Position actual value	Mandatory(csp,csv)	ro	INT32	TxPDO
0x6065		Following error window	Recommended(csp)	rw	UINT32	No
0x6066		Following error time out	Recommended(csp)	rw	UIN16	No
0x606C		Velocity actual value	Recommended(csv)	ro	INT32	TxPDO
0x6077		Torque actual value	Recommended(scp,csv)	ro	INT16	No
0x607A		Target position	Mandatory(csp)	rw	INT32	RxPDO
0x607B	0	Position range limit	Recommended(csp)	ro	UINT8	No
	1	Min position range limit	Recommended(csp)	rw	INT32	No
	2	Max position range limit	Recommended(csp)	rw	INT32	No
0x607C		Home Offset	Recommended(hm)	rw	INT32	RxPDO
0x607D	0	Software position limit	Recommended(csp)	ro	UINT8	No
	1	Min position limit	Recommended(csp)	rw	INT32	No
	2	Max position limit	Recommended(csp)	rw	INT32	No
0x607F		Max profile velocity	Optional	rw	UINT32	No
0x6080		Max motor speed	Optional	rw	UINT32	No
0x6081		Profile velocity	Mandatory(pp)	rw	UINT32	RxPDO
0x6083		Profile acceleration	Mandatory(pp)	rw	UINT32	RxPDO
0x6084		Profile deceleration	Optional	rw	UINT32	RxPDO
0x6085		Quick stop deceleration	Optional	rw	INT32	No
0x6098		Homing method	Mandatory(hm)	rw	INT8	RxPDO
0x6099	0	Homing speeds	Optional	ro	UINT8	No
	1	Speed during search for switch	Optional	rw	INT32	RxPDO
	2	Speed during search for zero	Optional	rw	INT32	RxPDO
0x609A		Homing acceleration	Optional	rw	UINT32	RxPDO
0x60B0		Position offset	Optional	rw	INT32	No

0x60B1		Velocity offset	Recommended(csp)	rw	INT32	No
0x60B2		Torque offset	Recommended(scp,csv)	rw	INT16	No
0x60C2	0	Interpolation time period	Recommended(csp,csv)	ro	UINT8	No
	1	Interpolation time period value	Recommended(csp,csv)	rw	UINT8	No
	2	Interpolation time index	Recommended(csp,csv)	rw	INT8	No
0x60F4		Following error actual value	Recommended(csp)	ro	INT32	No
0x60FF		Target velocity	Mandatory(csv)	rw	INT32	RxPDO
0x6402		Motor type	Optional	rw	UINT16	No
0x6502		Supported drive modes	Mandatory(all)	ro	UINT32	No

4. モータ制御パラメータ

CiA402 のオブジェクトに設定するモータ制御パラメータの設定値のサンプルプログラムにおける定義について説明します。

表 4-1 モータ制御パラメーター覧

INDEX	OBJECT Name	Support Mode	単位	Data	PDO Mapping
0x6064	Position actual value	pp,csp,csv	[deg]	INT32	TxPDO
0x606C	Velocity actual value	csv	[rpm]	INT32	TxPDO
0x607A	Target position	pp,csp	[deg]	INT32	RxPDO
0x6081	Profile velocity	pp	[deg]	UINT32	RxPDO
0x6083	Profile acceleration	pp	[rpm/s]	UINT32	RxPDO

Note：ステッピングモータの位置制御ではプロファイル制御を行わないため、Profile velocity と Profile acceleration は使用しません。

4.1 データ型

パラメータの型は INT32 または UINT32 ですが、小数点第一位を含む、固定小数点形式となっています。例えば、Target position に 178.9° を設定したい場合は「1789」を設定します。同様に Position actual value の値が「1789」であれば、178.9° であることを示します。

4.2 加速度パラメータ

モータ制御開発支援ツール「Renesas Motor Workbench」でチューニングしたゲイン等の制御パラメータはモータ制御プログラムのソースファイルに反映することで、EtherCAT による制御の際も同じ値が使用されます。

一方、RMW で使用した加速度の指令値は(Profile acceleration)CiA402 のオブジェクトに設定するモータ制御パラメータとは単位が異なるため、変換が必要となります。

RMW では加速度を定義するために、目標速度に到達するまでの加速時間[s]を使用します。
図 4-1 は目標速度 speed に到達するまでの加速時間を t1→t2 に変更することにより、加速度は acc1→acc2 に変化することを示しています。
また、そのときの加速度は速度÷加速時間の式で表すことができます。

acc1 = speed÷t1
acc2 = speed÷t2

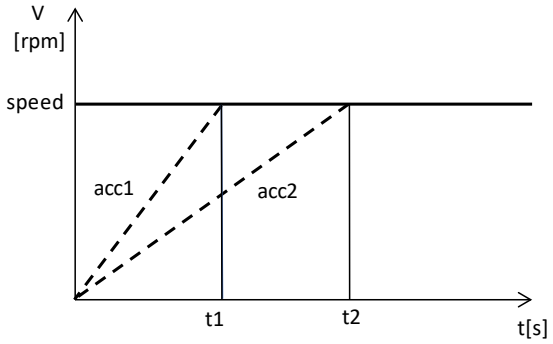


図 4-1 加速時間と加速度

変換例
・加速度指令値
R=2000[rpm]のとき T_{ACC}＝0.3[s]を A_C[rpm/s]に変換する
2000 rpm÷0.3 s = 6666.7 rpm/s
Profile acceleration = Ac × 10 = 6666.7 × 10 = 66667

5. API 関数

5.1 概要

モータ制御プログラムインタフェース用 API 関数を示します。

関数・変数	説明
R_MTR_ECAT_Open	モータ制御プログラムインタフェースの初期設定を行います。
R_MTR_ECAT_GetPositionPFStatus	エンコーダ位置制御のプロファイルステータスを取得します。
R_MTR_ECAT_SetActualPosition	現在位置[deg]を設定します。
R_MTR_ECAT_SetProfileSpeed	プロファイル速度指令値[rpm]を設定します。
R_MTR_EACAT_SetAcceleration	加速度指令値[rpm/s]を設定します。
R_MTR_ECAT_SetDeceleration	減速度指令値[rpm/s]を設定します。
p_api	モータ制御 API ポインタ※

モータ制御 API ポインタ※
モータ制御プログラムはモータ毎に異なります。このため BLDC モータとステッピングモータではモータ制御 API の名称が異なります。モータ制御 API ポインタの構造体のメンバー名はモータの種類に依存しない操作名としています。これにより EtherCAT 通信部はモータの種類を意識せずモータを制御できます。

5.2　R_MTR_ECAT_Open

モータ制御プログラムインタフェースの初期設定を行います。この関数は他の API 関数を使用する前に実行される必要があります。

Format

e_mtr_ecat_ret_t R_MTR_ ECAT_Open (st_encoder_vector_control_t *p_st_srfoc);

Parameters

st_encoder_vector_control_t *p_st_srfoc
制御対象のエンコーダベクトル制御変数へのポインタを設定します。

Return Values

MTR_ECAT_SUCCESS：初期化成功
MTR_ECAT_ERR_NOT_SUPPORT：変数ポインタが空

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

システム動作状態の初期化やモータ制御パラメータのデフォルト値の設定を行います。

Example

```
/* Setup EtherCAT motor interface */
R_MTR_ECAT_Open(&g_st_encoder_vector);
```

5.3　R_MTR_ECAT_GetPositionPFStatus

この関数は、エンコーダ位置制御のプロファイルステータスを取得します。

Format

e_mtr_ecat_ret_t R_MTR_ECAT_GetPositionPFStatus(uint8_t *u1_status)

Parameters

　　なし

Return Values

u1_status：プロファイルステータス
　MTR_POS_STEADY_STATE　　(0)：安定状態(現在位置を変更していない)
　MTR_POS_TRANSITION_STATE　(1)：遷移状態(現在位置を変更している)

MTR_ECAT_SUCCESS：取得成功
MTR_ECAT_ERR_NOT_OPEN：ドライバが Open されていないため取得失敗

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

位置制御時にモータが静止状態かどうかを調べるときに本関数を使用できます。

Example

```
  uint8_t   u1_pos_state;

/* Get position profile status */
R_MTR_ECAT_GetPositionPFStatus(&u1_pos_status);
```

5.4　R_MTR_ECAT_SetActualPosition

この関数は、現在位置[deg]を設定します。

Format

e_mtr_ecat_ret_t R_MTR_SetActualPositionUnits (float f4_actual_position)

Parameters

位置指令値[deg]

Return Values

MTR_ECAT_SUCCESS：設定成功
MTR_ECAT_ERR_NOT_OPEN：ドライバが Open されていないため設定失敗

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

位置指令値は符号付で単位は[deg]です。
本関数はモータ制御を行わずに現在位置の値のみを設定する関数です。
ホーミングモードで初期位置にオフセットを持たせたいとき等に使用できます。

Example

```
/* Set current position 180.1[deg]*/
R_MTR_SetActualPosition(180.1f);
```

5.5　R_MTR_ECAT_SetProfileSpeed

この関数は、プロファイル制御の最大速度指令値を設定します。

Format

e_mtr_ecat_ret_t R_MTR_ECAT_SetProfileSpeed(float f4_profile_speed)

Parameters

速度指令値[rpm]

Return Values

MTR_ECAT_SUCCESS：設定成功
MTR_ECAT_ERR_NOT_OPEN：ドライバが Open されていないため設定失敗

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

速度指令値は符号付で単位は[rpm]です。

Example

```
/* Set Profile Speed 2000.5 rpm */
R_MTR_SetSpeedUinits(2000.5f);
```

5.6　R_MTR_ECAT_SetAcceleration

この関数は、加速度指令値[rpm/s]を設定します。

Format

e_mtr_ecat_ret_t R_MTR_ECAT_SetAcceleration(float f4_ref_acceleration)

Parameters

加速度指令値[rpm/s]

Return Values

MTR_ECAT_SUCCESS：設定成功
MTR_ECAT_ERR_NOT_OPEN：ドライバが Open されていないため設定失敗

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

加速度指令値は符号付で単位は[rpm/s]です。

Example

```
/* Set acceleration 4678.9[rpm/s] */
R_MTR_ECAT_SetAcceleration (4678.9f);
```

5.7　R_MTR_ECAT_SetDeceleration

この関数は、減速度指令値を設定します。

Format

e_mtr_ecat_ret_t R_MTR_ECAT_SetDecceleration(float f4_ref_deceleration)

Parameters

減速度指令値[rpm/s]

Return Values

MTR_ECAT_SUCCESS：設定成功
MTR_ECAT_ERR_NOT_OPEN：ドライバが Open されていないため設定失敗

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

減速度指令値は符号付で単位は[rpm/s]です。
【注】サンプルプログラムでは減速度指令値をモータ制御に使用しておりません。

Example

```
/* Set deceleration 4678.9 [rpm/s] */
R_MTR_ECAT_SetDeceleration (4678.9f);
```

5.8　p_api

モータ制御 API の関数ポインタを格納する変数です。

Format

st_ecat_motor_api_t const * p_api;

Struct

モータ制御 API 構造体 st_ecat_motor_api_t のメンバーと格納するモータ制御 API について説明します。

メンバー名	PositionCommandModeSet
型	void (* PositionCommandModeSet) (st_encoder_vector_control_t *p_st_encoder_vector, uint8_t u1_position_command_mode);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_PositionCommandModeSet
メンバー名	CtrlTypeSet
型	void (* CtrlTypeSet) (st_encoder_vector_control_t * p_st_encoder_vector, uint8_t ctrl_type);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_CtrlTypeSet
メンバー名	MotorStart
型	void (* MotorStart)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_MotorStart
メンバー名	MotorStop
型	void (* MotorStop)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_MotorStop
メンバー名	MotorErrorCancel
型	void (* MotorErrorCancel)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_MotorReset
メンバー名	MotorReset
型	void (* MotorReset)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_MotorReset
メンバー名	PositionGet
型	float (* PositionGet)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_PositionGet
メンバー名	PositionSet
型	void (* PositionSet) (st_encoder_vector_control_t *p_st_encoder_vector, float f4_ref_position);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_PositionSet
メンバー名	SpeedGet
型	float (* SpeedGet)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_SpeedGet
メンバー名	SpeedSet
型	void (* SpeedSet) (st_encoder_vector_control_t *p_st_encoder_vector, float f4_ref_speed);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_SpeedSet
メンバー名	StatusGet
型	uint8_t (* StatusGet)(st_encoder_vector_control_t *p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_StatusGet
メンバー名	ErrorStatusGet
型	uint16_t (* ErrorStatusGet)(st_encoder_vector_control_t * p_st_encoder_vector);
モータ制御 API	R_MOTOR_ENCODER_VECTOR_ErrorStatusGet

Properties

r_mtr_driver_ecat_acces.h にプロトタイプ宣言されています。

Description

p_api に格納するモータ制御 API は r_mtr_driver_ecat_acces.c にテーブルとして定義しています。
const st_ecat_motor_api_t ecat_motor_api

Example

```
/* Motor API settings */
st_ecat_motor_api_t const * p_api;
p_api = &ecat_motor_api;
st_ecat_motor_api_t const * p_motor_api = p_api;
/* Motor Start */
(p_motor_api->MotorStart)(ecat_param_buffer.p_st_foc);
```

6. ソリューションキットでの動作確認

本章ではモータソリューションキットを用いて EtherCAT 通信でモータを制御するサンプルアプリケーションの動作について説明します。

6.1 動作環境

本マニュアルのサンプルプログラムは、下記の環境を想定しています。

表 6-1 動作環境

項目	内容
使用ボード	RX72M CPU Card with RDC-IC ルネサスエレクトロニクス製：RTK0EMXDE0C00000BJ BLDC モータ駆動用インバータボード ルネサスエレクトロニクス製：RTK0EM0000B10020BJ Evaluation System for BLDC Motor (RTK0EMX270S00020BJ)に同梱
CPU	RX CPU (RXv3)
動作電圧	24V
通信プロトコル	EtherCAT
統合開発環境	CCRX コンパイラ(V3.05.00 以降) + e2studio(2023-10 以降)
FIT Module	r_ecat_rx:1.30 以降
エミュレータ	ルネサスエレクトロニクス製 e2 Lite
SSC Tool	EtherCAT Technology Group (ETG) 提供 Slave Stack Code (SSC) Tool Version 5.13 以降
ソフトウェア PLC	Beckhoff Automation 製 TwinCAT® 3 (Beckhoff web サイトからダウンロード)

なお、SSC Tool、TwinCAT のインストールは完了しているものとします。

本マニュアルのサンプルプログラムで使用するソフトウェアコンポーネントと、そのバージョンは下記を想定しています。

表 6-2 動作確認コンポーネント

コンポーネント	バージョン	設定
Borad Support Package (r_bsp)	7.41	r_bsp
EtherCAT Slave Controller (ESC) Software (r_ecat_rx)	1.30	r_ecat_rx
イベントリンクコントローラ	1.9.1	Config_ELC
ウォッチドッグタイマ	1.11.0	Config_IWDT
コンペアマッチタイマ	2.3.0	Config_CMT0, Config_CMT2
シングルスキャンモード S12AD	2.5.0	Config_S12AD0, Config_S12AD1
ノーマルモードタイマ	1.12.0	Config_MTU0
ポート	2.4.1	Config_PORT
ポートアウトプットイネーブル	1.11.0	Config_POEG
位相計数モードタイマ	2.4.0	Config_MTU1
割り込みコントローラ	2.3.0	Config_ICU
汎用 PWM タイマ	1.5.2	Config_GPT0, Config_GPT1, Config_GPT2, Config_GPT3,

6.2　動作環境の設定、接続

電源、モータ、インバータボードを配線します。

(1)　モータの電源線とインバータボード出力部、モータのエンコーダ出力線と CPU カードの入力部を以下のように接続します。

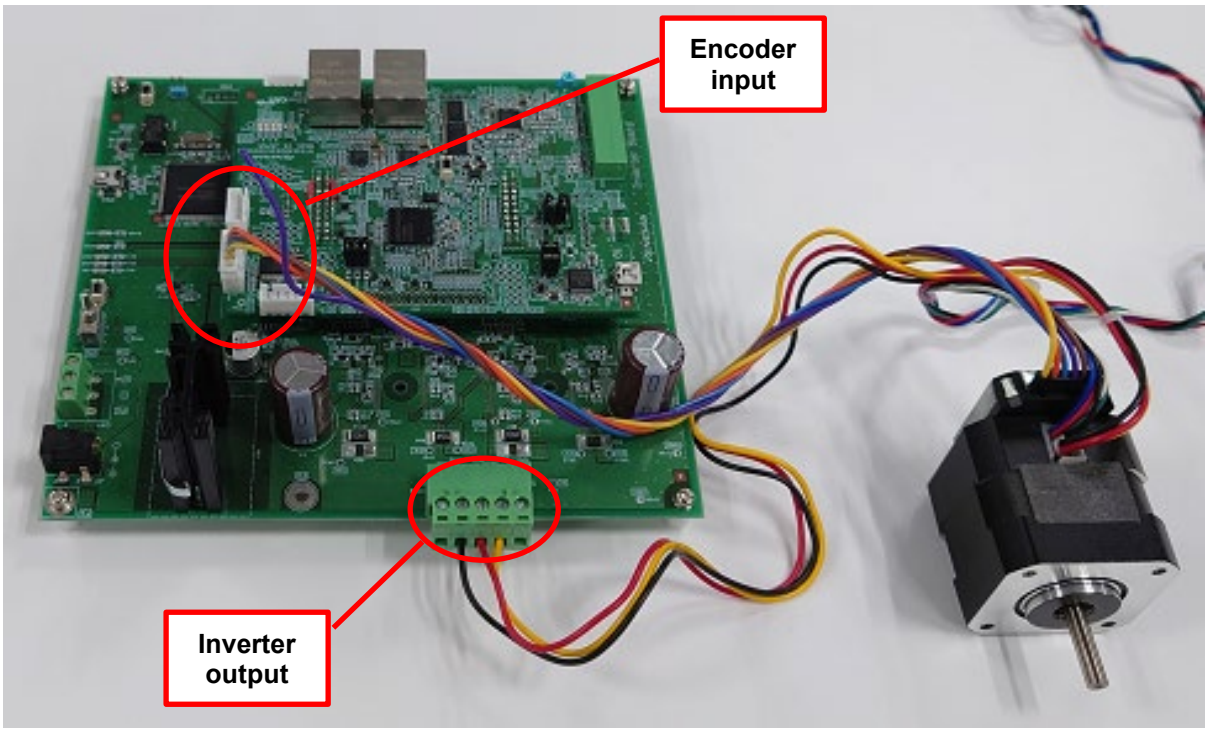


図 6-1　三相電源線の接続

(2)　インバータボードに電源を以下のように接続します。

図 6-2　電源の接続

(3)　以下のような接続構成となります。

(4)　インバータボードの詳細を以下に示します。
-- 電源投入は AC アダプタにて行ってください。
-- RMW Connector は RMW(Renesas Motor Workbench)使用時に接続してください。
-- SW は使用しません。

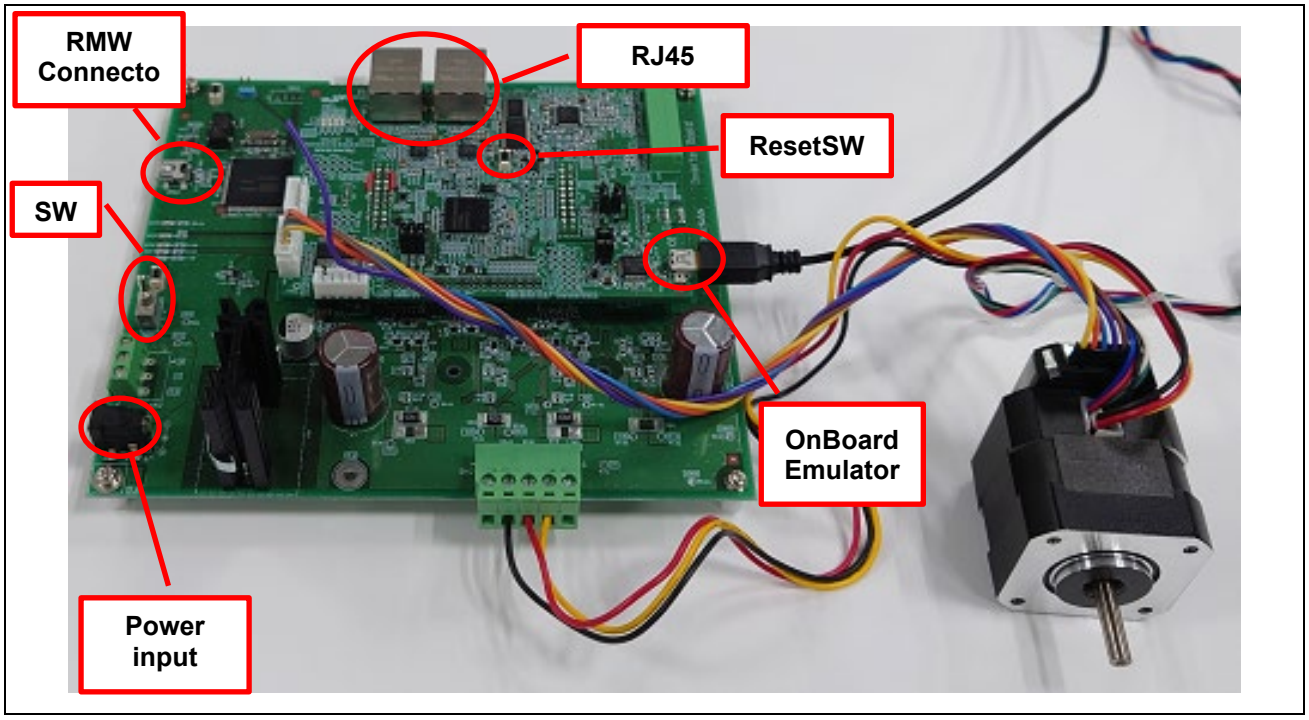


図 6-3　インバータボード接続図

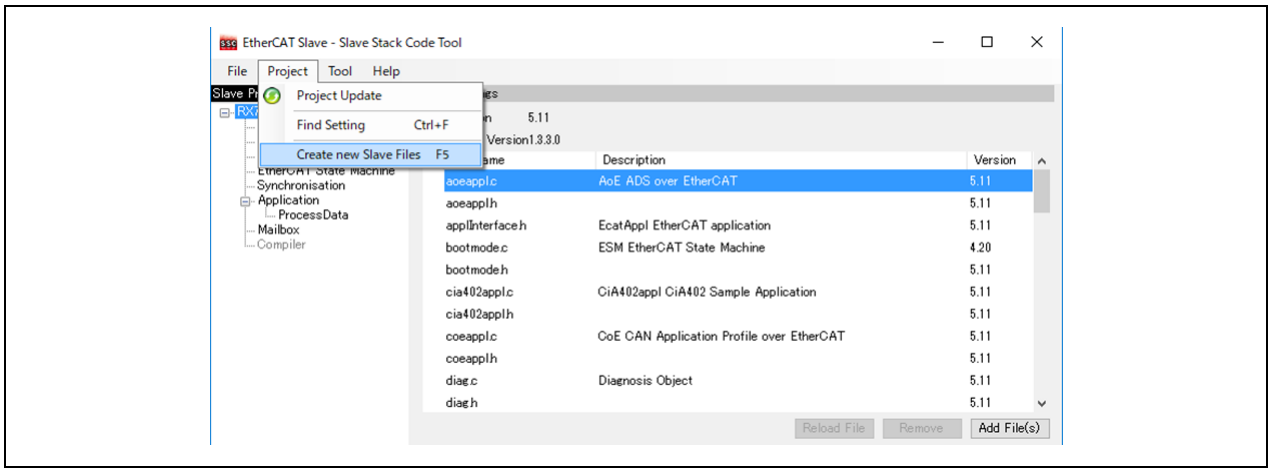
6.3　サンプルプログラムの構築

本サンプル・プロジェクトには EtherCAT スレーブスタックコードは同梱されていません。
EtherCAT スレーブスタックコードの生成には”EtherCAT Slave Stack Code(SSC) Tool”が必要です。
SSC Tool は ETG 協会から入手可能です。
サンプルプログラムは.zip 形式で提供されますので、予め任意のフォルダに解凍してください。

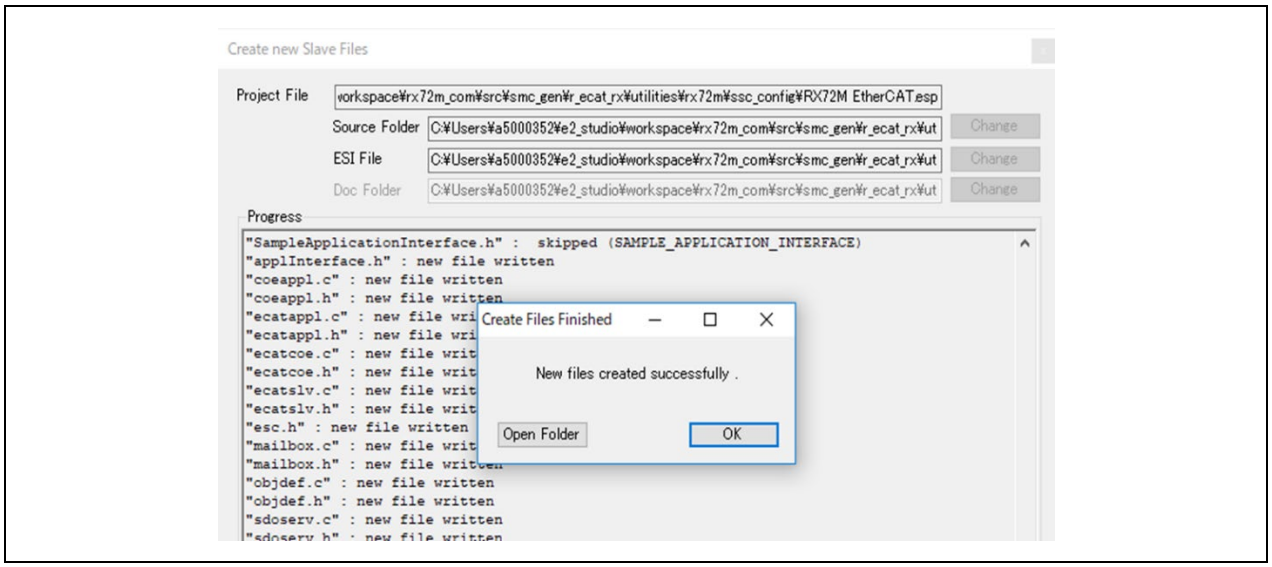
(1)　サンプルプログラムの SSC プロジェクトファイルをダブルクリックして SSC ツールを起動します。

rx72m_ecat_cia402_bldc_encd¥utilities¥ssc_config¥RX72M EtherCAT CiA402.esp

(2)　[Project]→[Create New Slave Files]をクリック[Current new Slave Files] →[Start]をクリックします。



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



(4) パッチコマンドをインストールしていない場合 、GNU Patch Ver2.5.9 以後が必要です。 インストール済みの場合は本手順をスキップしてください。

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

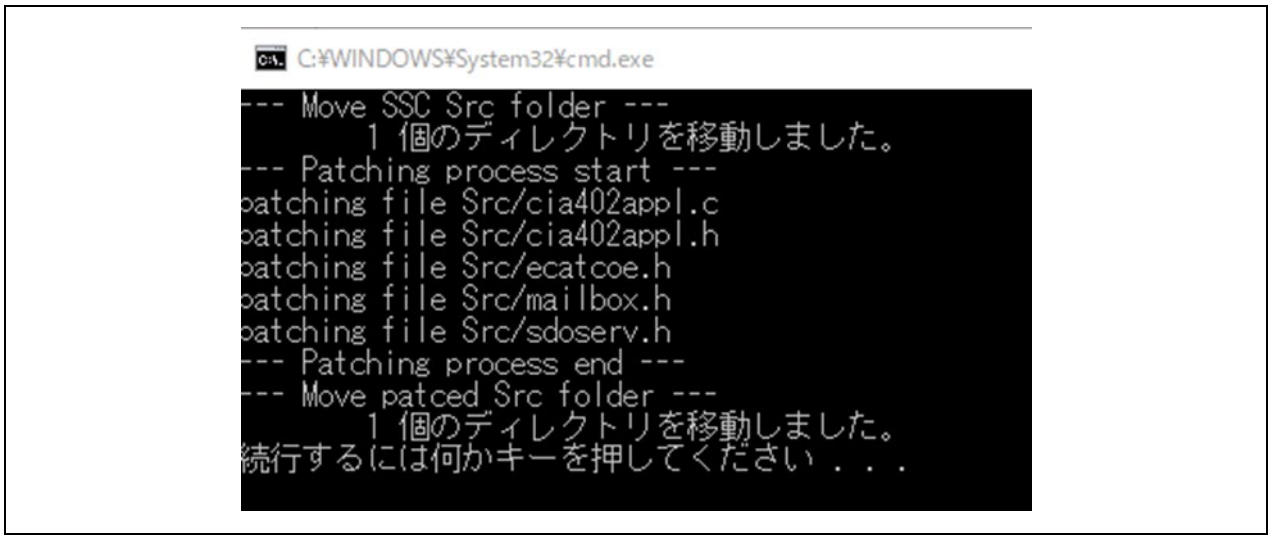
http://gnuwin32.sourceforge.net/packages/patch.htm

(5) apply_patch.bat ファイルを右クリックして[管理者として実行] ⇒ [はい]を選択します。 パッチファイルは SSC ソースファイルに対する RX 向けの修正を含んでいます。

rx72m_ecat_cia402_bldc_encd¥utilities¥batch_files¥apply_patch.bat

(6) パッチ実行後、修正されたソースファイルは下記のフォルダに格納されます。

rx72m_ecat_cia402_bldc_encd¥project¥ecat¥application¥beckhoff¥Src

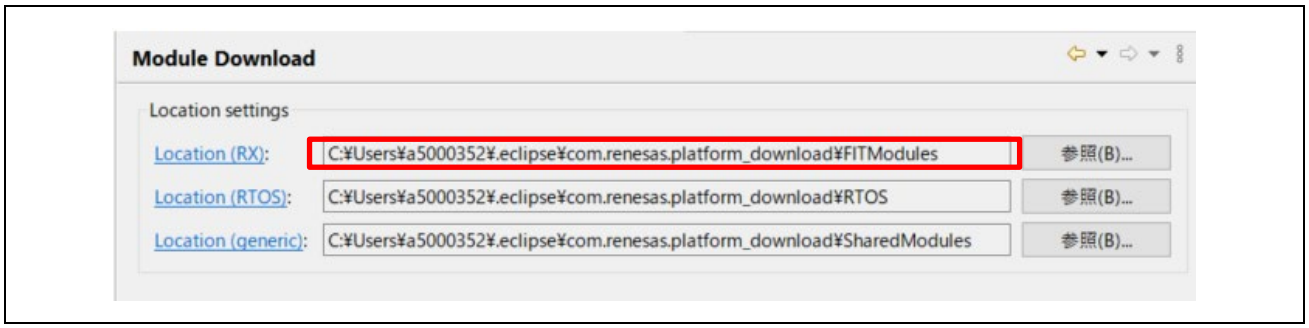


6.4　FIT モジュールの追加

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

(1)　e² studio の FIT モジュールの保存先フォルダに EtherCAT FIT モジュールをコピーします。
e² studio で FIT モジュールの保存先を確認します。
- 「ウインドウ」→「設定」→設定ウインドウが開きます。
- 「C/C++」→「Renesas」→「スマートコンフィグレータ」→「コンポーネント」を選択してください。
- 「フォルダ設定」→「Module Download」ページを開きます。

- Location(RX):として表示されているフォルダが該当フォルダです。



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

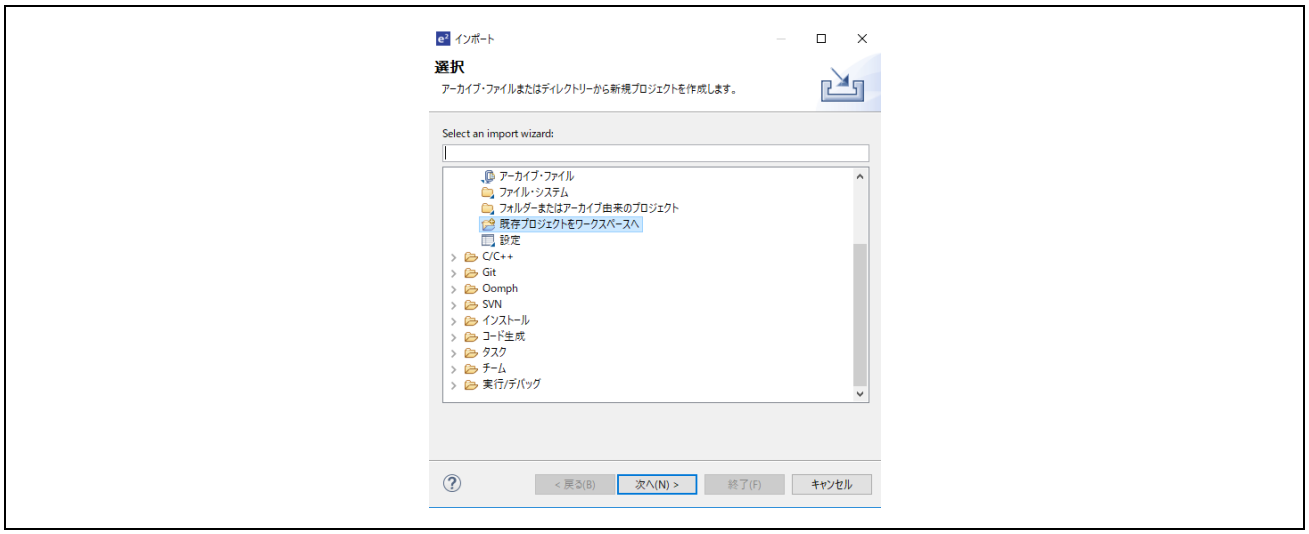
- an-r01an4881xx*NNNN*-rx-ecat¥FITModules フォルダにあるファイルを FIT モジュールの保存先フォルダにコピーしてください。
  r_ecat_rx_v*N.NN*.xml
  r_ecat_rx_v*N.NN*.zip
  r_ecat_rx_v*N.NN*_extend.mdf

なお、*NNNN* および *N.NN* はバージョンを表す数値になります。

6.5　サンプル・プロジェクトを e²studio にインポート

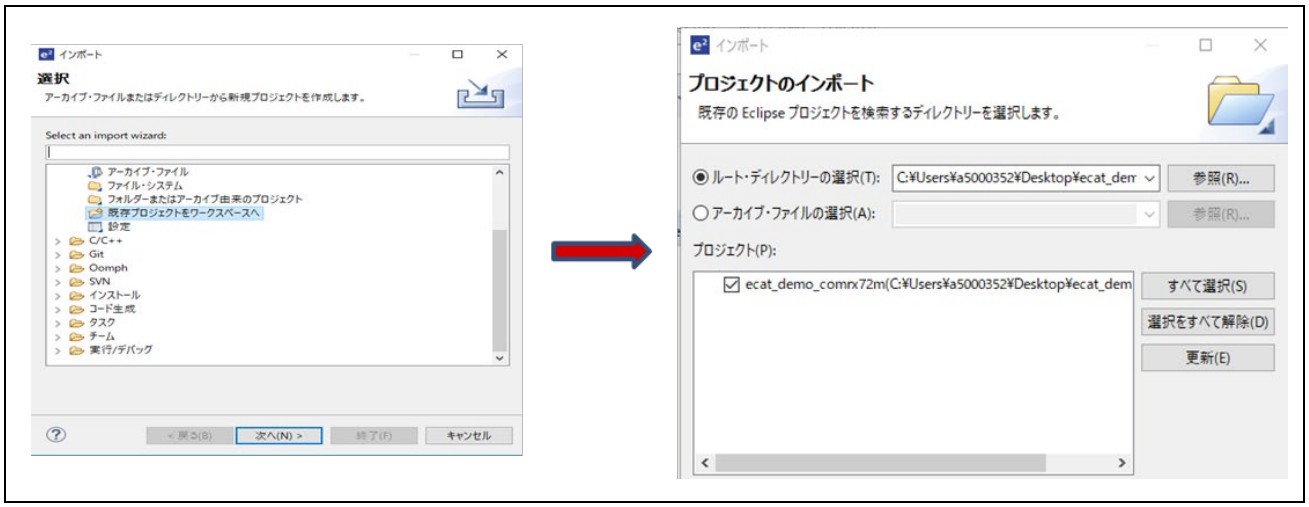
(1)　[ファイル]→[インポート]をクリックします。

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



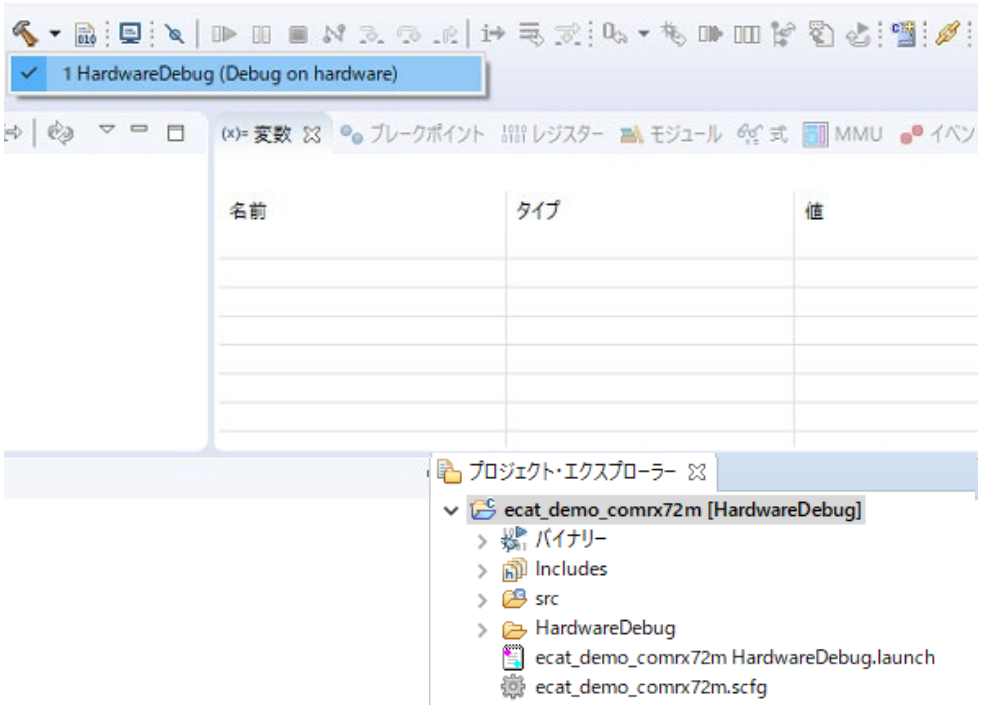
(3)　[プロジェクトのインポート]ダイアログの[ルート・ディレクトリの選択]チェックボックスを選択し、[参照]をクリックします。

(4)　通信ボード用サンプル・プロジェクトである”rx72m_ecat_cia402_bldc_encd”を選択して[開く]をクリックします。[プロジェクト]の” rx72m_ecat_cia402_bldc_encd”をチェックし[次へ]をクリックするとプロジェクトがインポートされます。

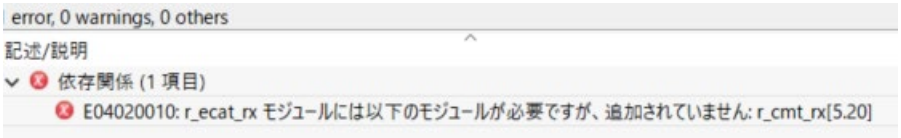


6.6　プログラミングとデバッグ

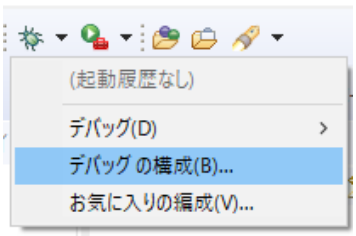
(1)　プロジェクト・エクスプローラーで“rx72m_ecat_cia402_bldc_encd”プロジェクトを左クリックし、[ビルド]ボタン（ハンマーアイコン）の横にある矢印をクリックし、ドロップダウンメニューから[Hardware Debug]を選択します。
e2studio を使用してプロジェクトをビルドします。コンソール上でビルドエラーがないことを確認してください。



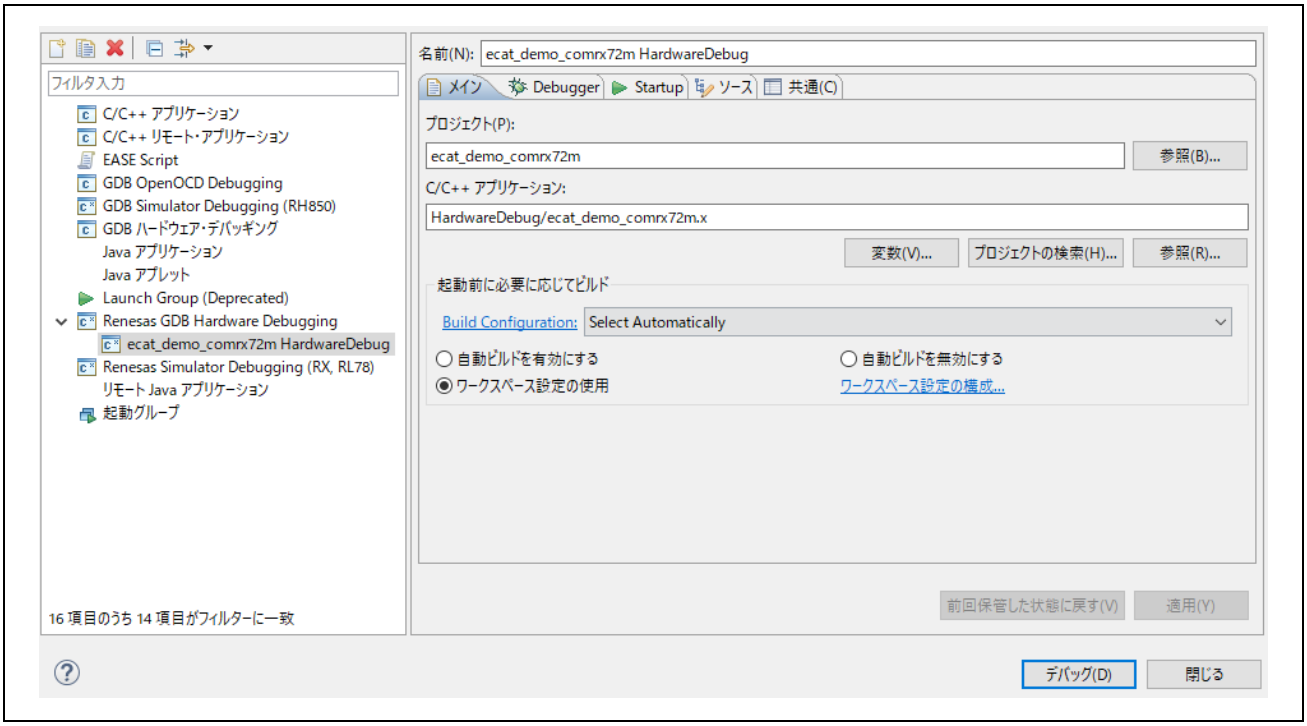
【注意】
ビルド時にコードの生成が行われます。
また、依存関係のエラーメッセージが表示されますがプログラムの構成には問題ありません。



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



(3) “rx72m_ecat_cia402_bldc_encd Hardware Debug”をクリックしてターゲットにプログラムをダウンロードし、デバッグボタンを押して開始します。



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

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

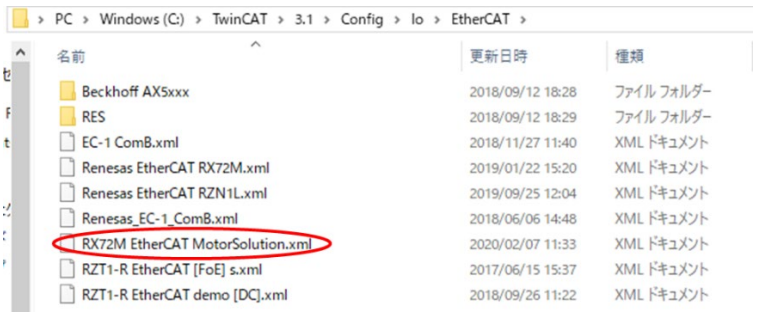
(6) パースペクティブ切り替えの確認ダイアログにてパースペクティブの変更を勧めるダイアログが表示される場合は「常にこの設定を使用する」チェックボックスにチェックし、[はい]をクリックします。

(7) E2 Lite デバッガの緑色の「ACT」LED が常に点灯します。

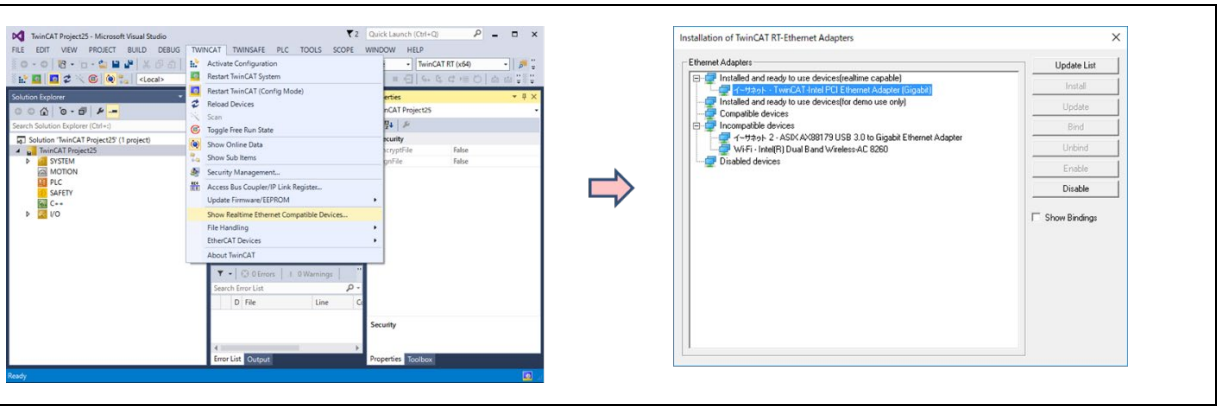
コードをダウンロードしたら、<再開>ボタンをクリックして、メイン関数 main 最初の行までコードを実行します。 もう一度<再開>ボタンをクリックすると、残りのコードでターゲットが実行されます。

6.7　TwinCAT との接続 (ESI ファイルの書き込み)

(1)　TwinCAT を開始する前に、リリースフォルダに含まれている ESI ファイルを、
“/ TwinCAT / 3.x / Config / IO / EtherCAT“ にコピーしてください
“rx72m_ecat_cia402_bldc_encd¥utilities¥esi¥ RX72M EtherCAT MotorSolution.xml”



(2)　次の手順で TwinCAT 用のドライバを追加します。（初回のみ）
スタートメニューから[TWINCAT]→[Show Realtime Ethernet Compatible Devise]を選択します。
通信ポートの中から接続している Ether ポートを選択してインストールを押します。



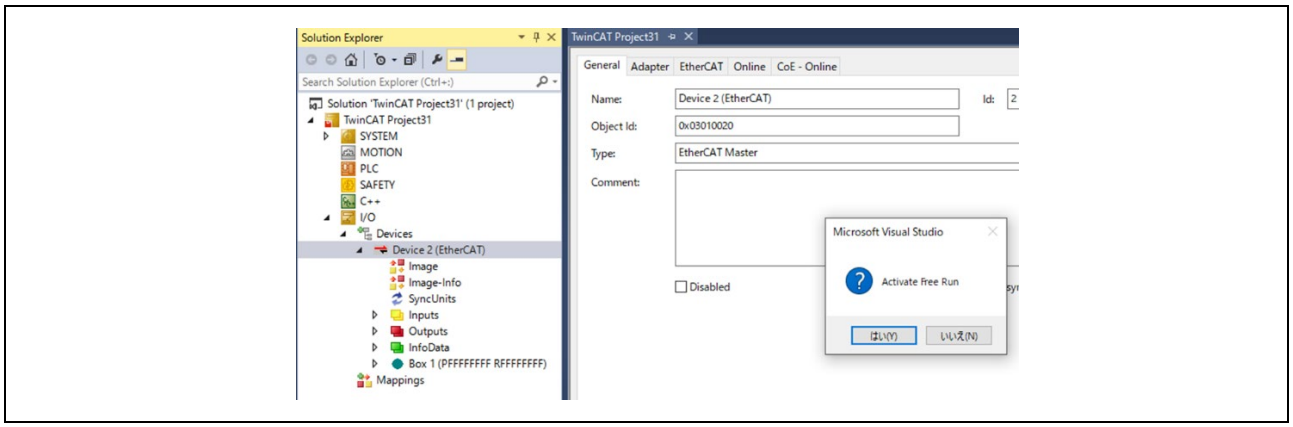
(3)　通信ポートの中から接続している Ether ポートを選択して、プロパティを表示させます。
プロパティから[TwinCAT Ethernet Protocol for All Network Adapters]のみ有効にして閉じます。

(4) 評価ボードに LAN ケーブルを接続します。EtherCAT は In/Out が決められていますので、CN2 IN に接続してください。

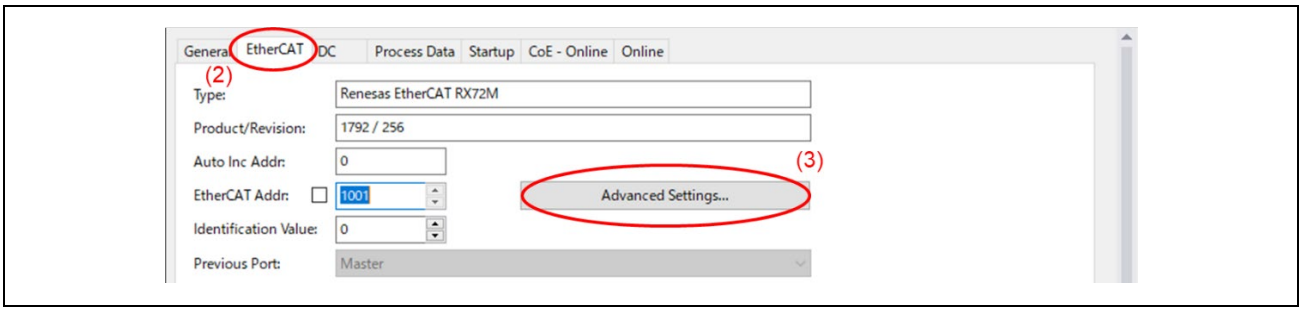
(5) スタートメニューから[Beckhoff]→[TwinCAT3]→[TwinCAT XAE（VS2013）]を選択、プログラムの起動後、[FILE]→[New]→[Project]を選択して、Templates の中の[TwinCAT XAE Project]を選択して新規プロジェクトを作成します。

(6) ソリューションエクスプローラ→I / O →デバイス→「スキャン」を選択します。

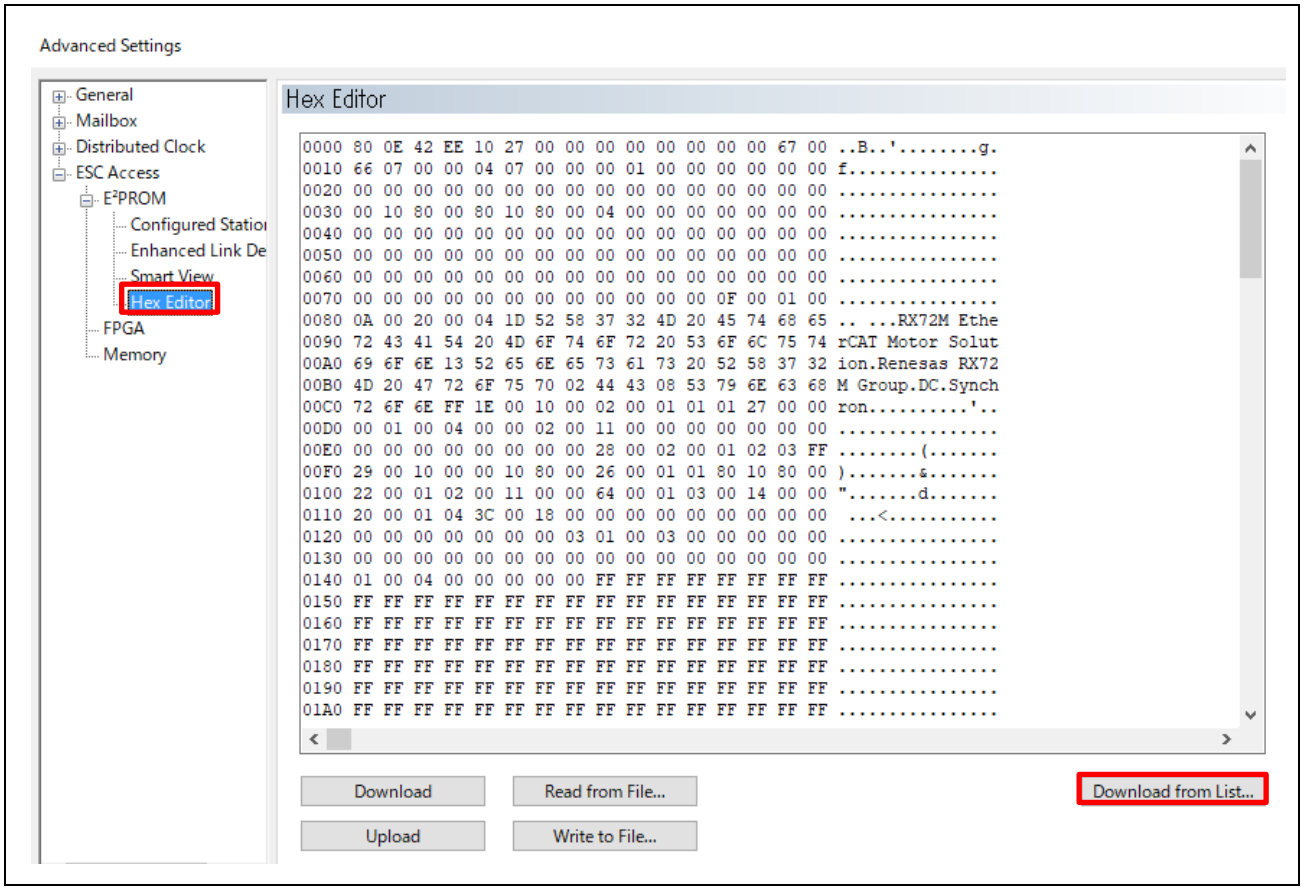
(7) [Scan for boxes]を実効すると、検出された Box1 のスレーブが Solution Explorer に現れます。
ESI ファイルを認識できていない状態では、Box 1 (PFFFFF)と等で表示されます。ESI をダウンロードする必要がありますので、[Activate free Run]はいいえとしてください。



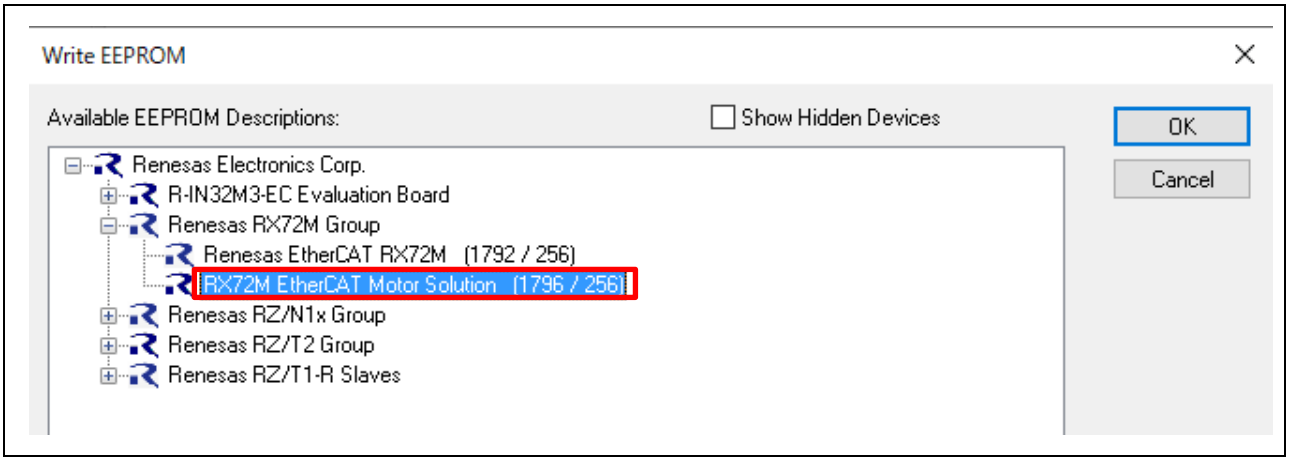
(8) 別のアプリケーションのデータがすでに EEPROM に書き込まれている場合は、データを置き換えます。EEPROM 上のデータを置換する手順は以下の通りです。
- [Box 1] をダブルクリックしてください。設定画面が表示されます。
- [EtherCAT] tab を選択してください。
- [Advanced Setting] ボタンを実行してください。



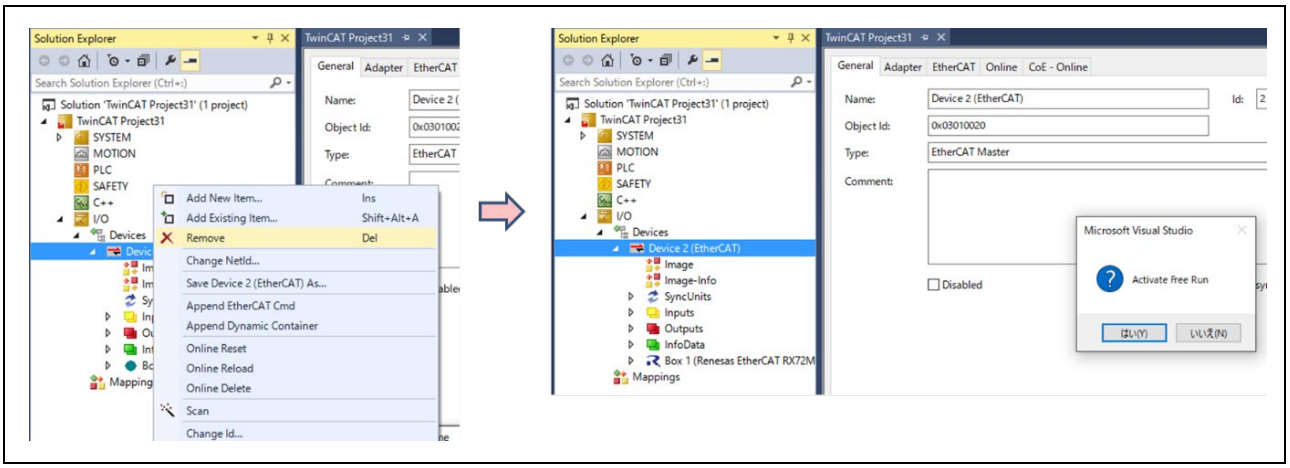
(9) [ESC Access] → [EEPROM] → [Hex Editor]を選択してください
[Download from List] を選択してください。



(10) TwinCAT3 に登録してある ESI ファイルの一覧が現れますので、該当するファイルを選択してください。モータボードの場合は、[RX72M EtherCAT MotorSolution.xml]です。I/O ボードの場合は、[Renesas EtherCAT RX72M.xml]です。



(11) DL した ESI ファイルの設定を反映させます。スレーブをリセットする必要がありますので、TwinCAT のネットワークから一旦スレーブを削除します。
スレーブをリセット後、再度スキャンをすると ESI ファイルが読み込まれていますので、Activate Free Run で実行してください。

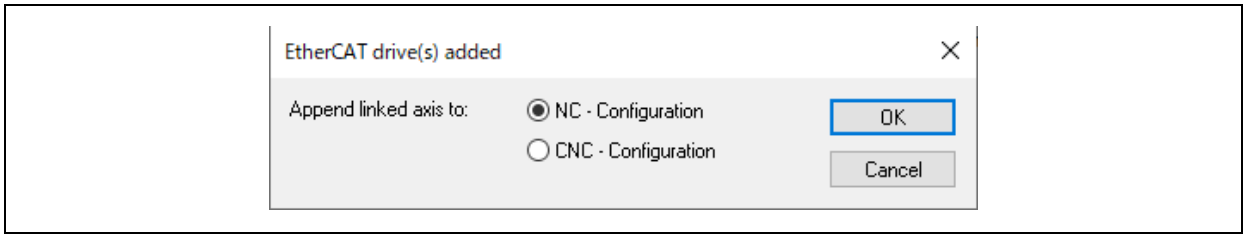


6.8　TwinCAT3 との接続確認

本章ではサンプルプログラムをインストールした評価環境を、TwinCAT3 にて接続、動作させるための手順について説明します。

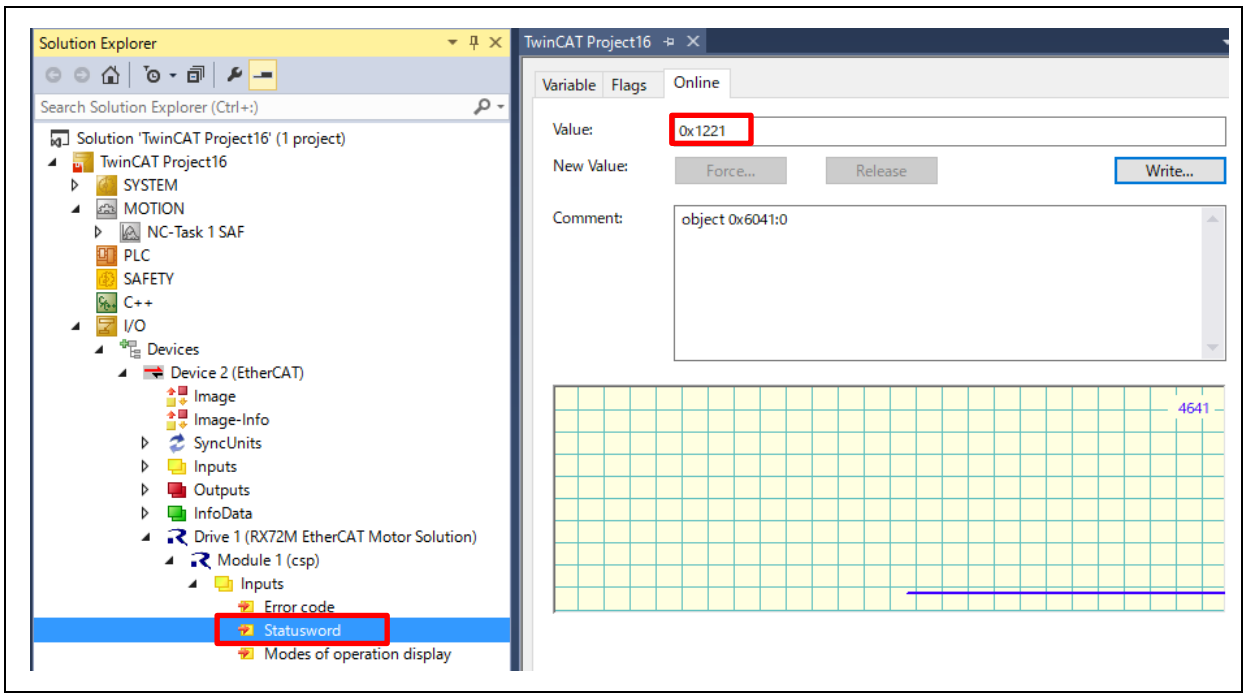
6.8.1　TwinCAT3 でのモータ動作確認

(1)　ESI ファイルの更新後、TwinCAT を再起動します。Device のスキャンを行うと、
Motor Solution 用の ESI を使用しているので、軸の Configuration 設定が現れますので、
「NC-Configuration」を選択します。

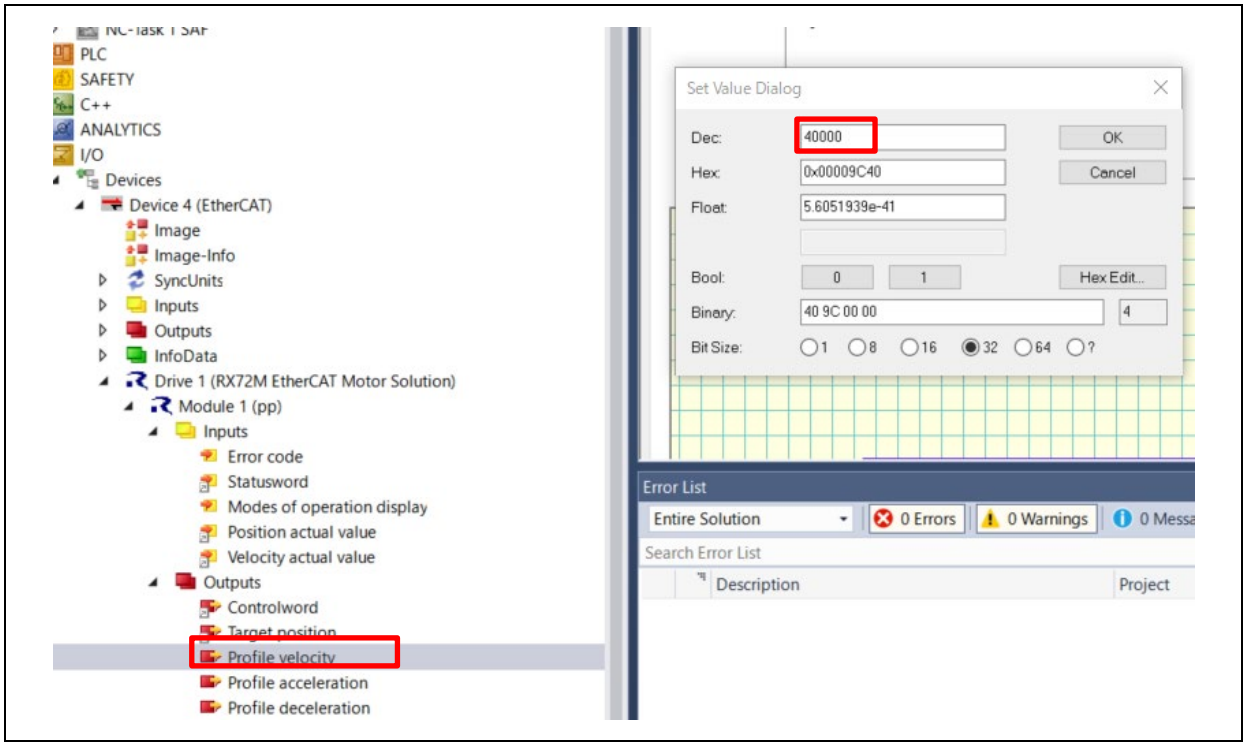


(2)　pp プロファイルの動作確認を行います。
Controlword を 128(Dec)にしてください。

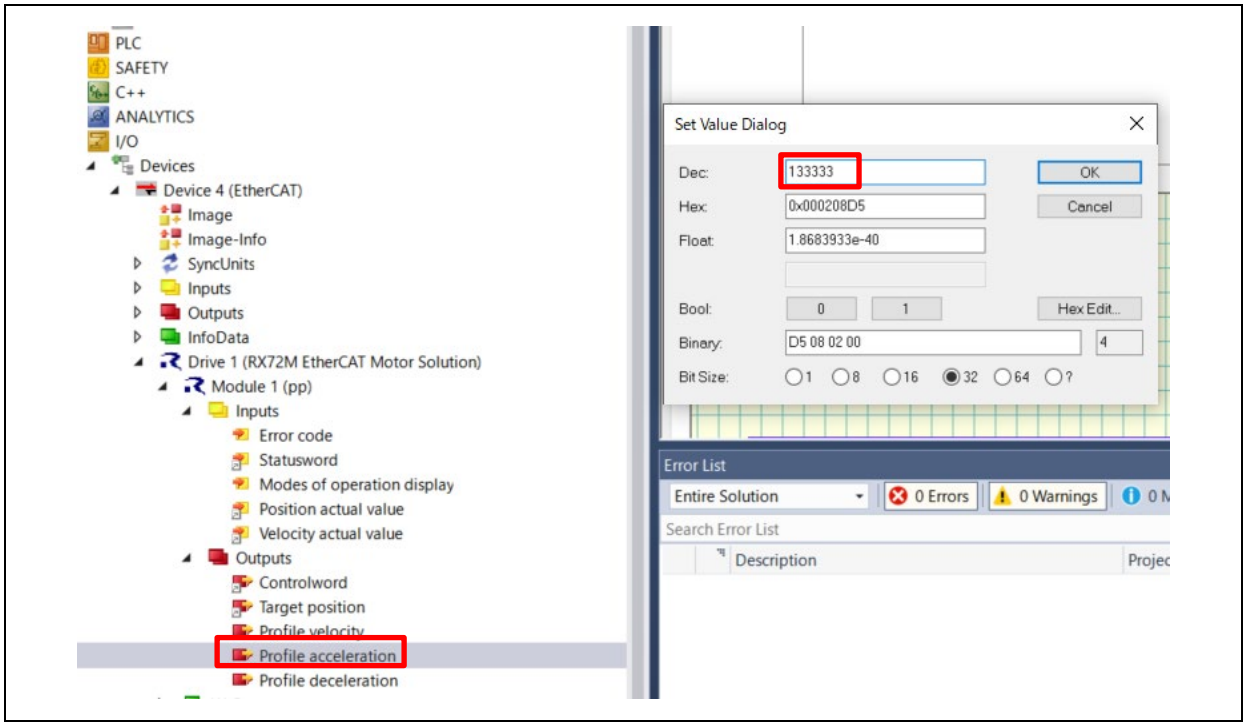
(3)　Statusword が 0xX221(Hex)であることを確認します。



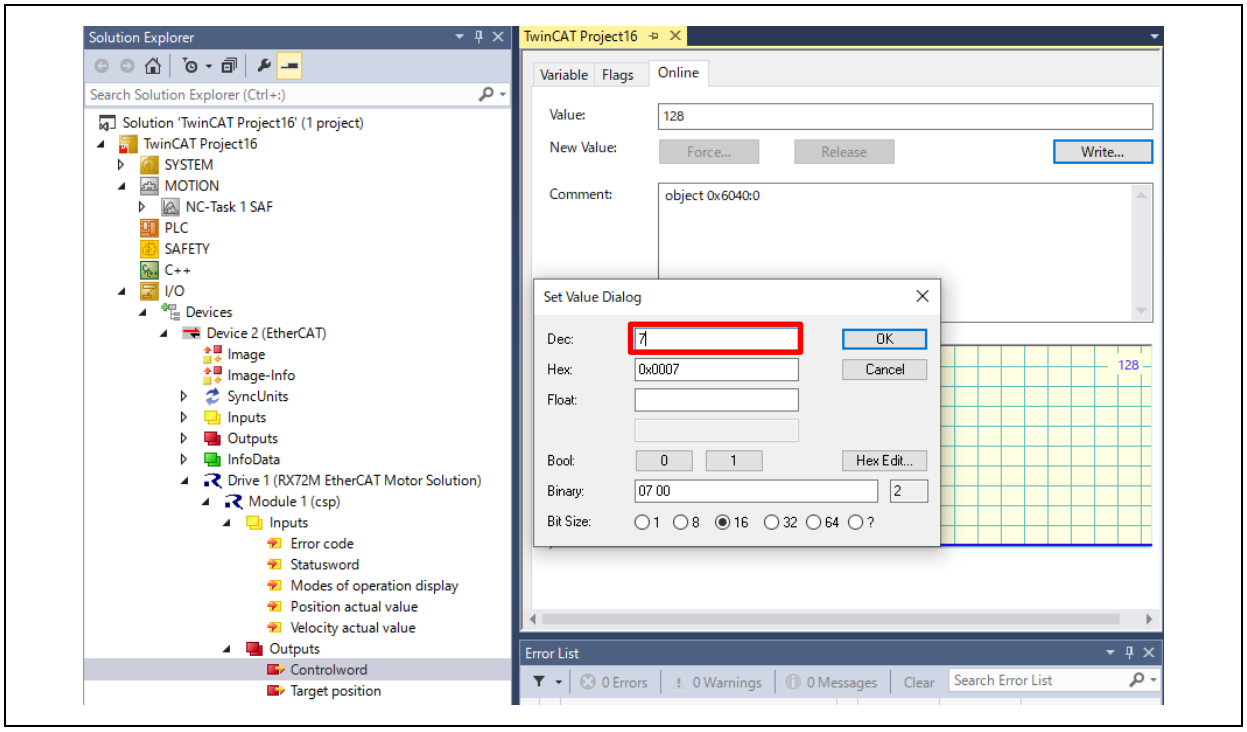
(4) Profile velocity を設定します。
Profile velocity = 40,000(Dec)を設定します。



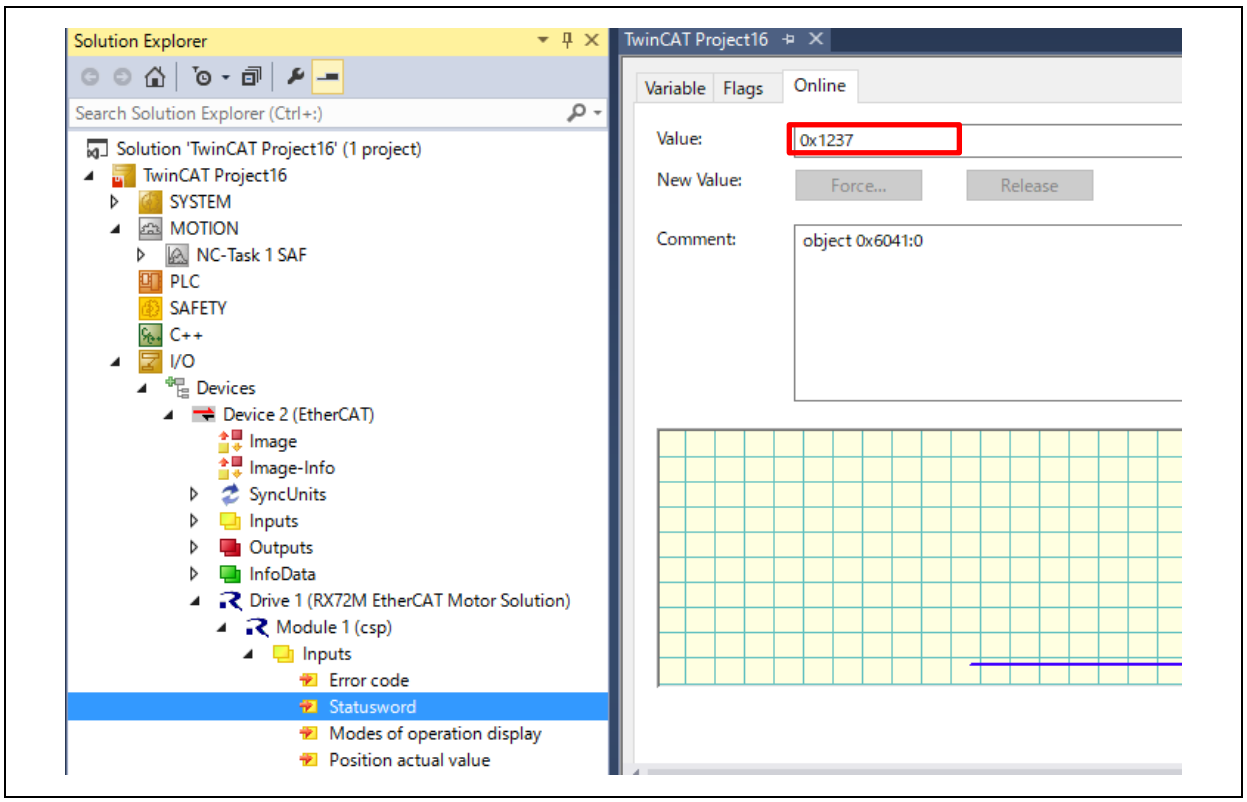
(5) Profile acceleration を設定します。
Profile accelaeration = 133333(Dec)を設定します。



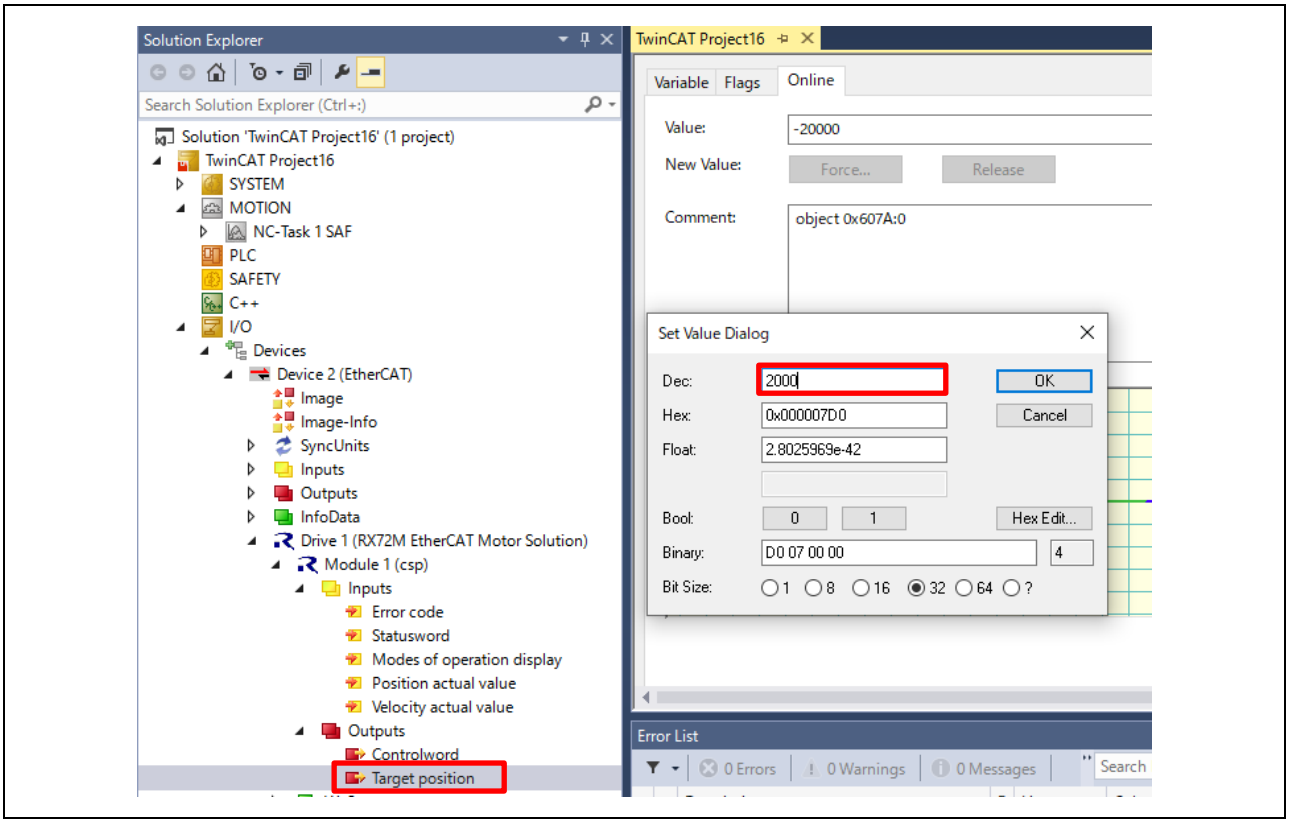
(6)　Controlword を 7 → 15 に設定します。[Controlword] → [Online] →[7],[15]



(7)　Statusword が 0x1237(Hex)であること、LED1 が点灯していることを確認します。

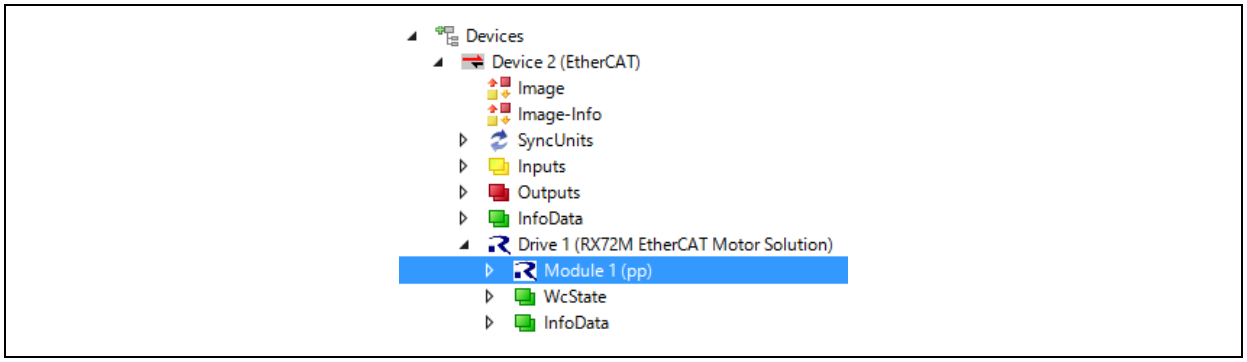


(8)　Target position に値を入力することで位置制御が可能となります。

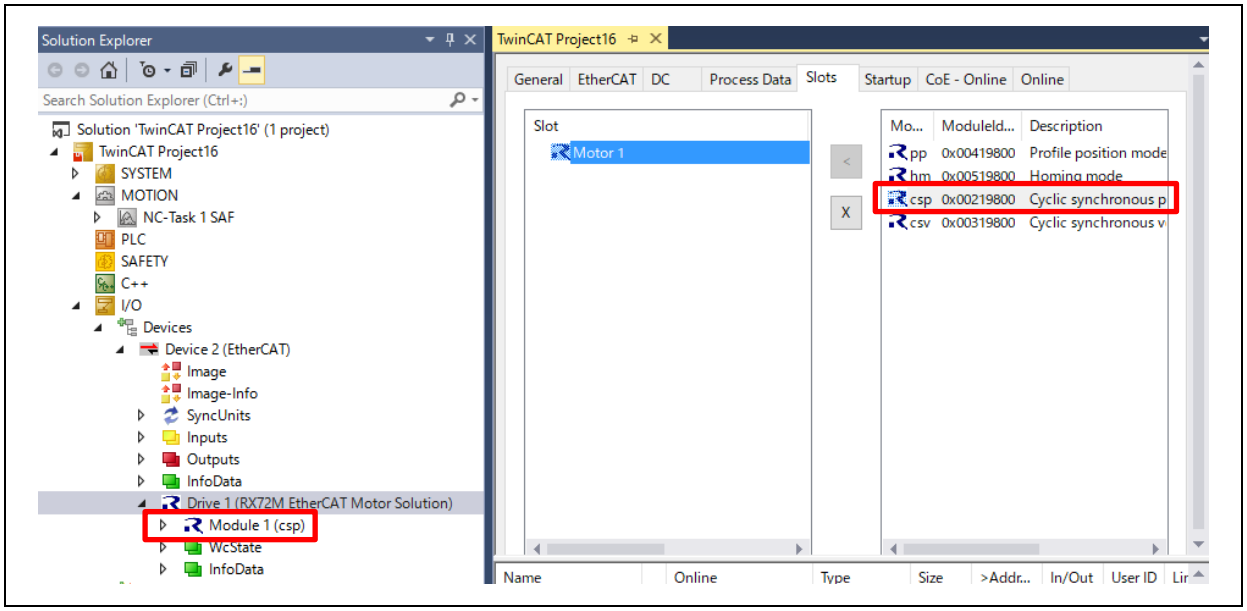


(9)　csp, (csv)プロファイルの動作確認を行います。

(10) デバイスの Slots から pp プロファイルを削除して csp, (csv)プロファイルを追加します。

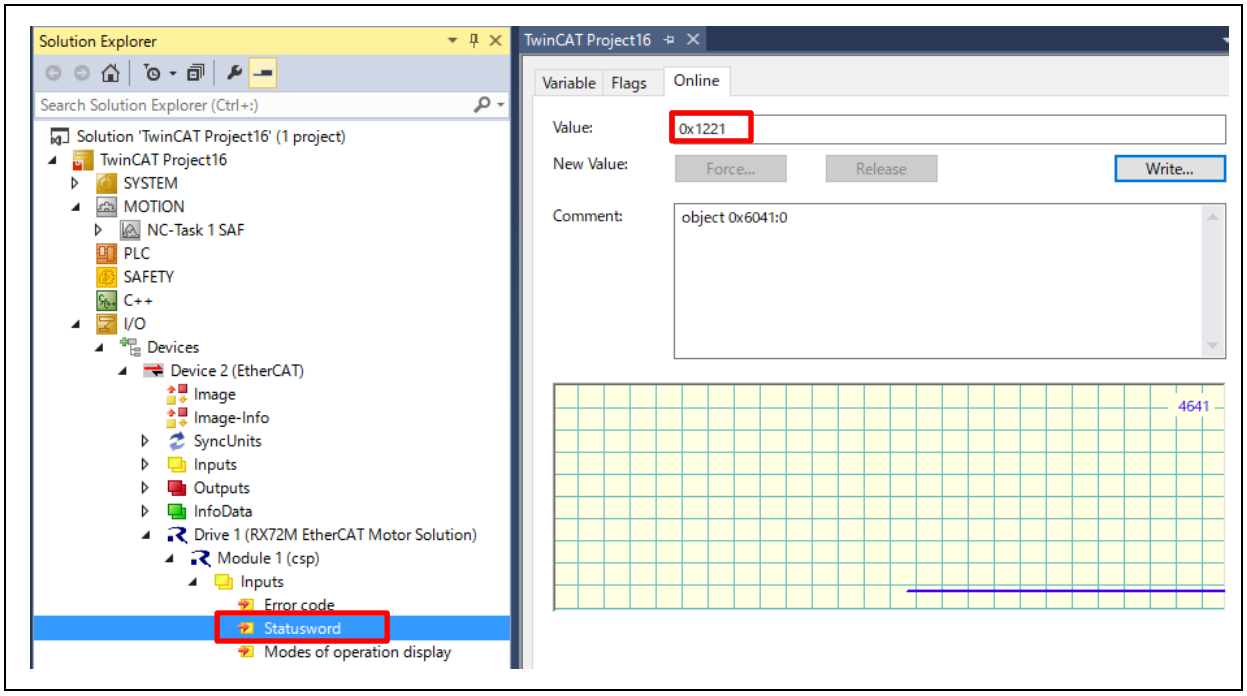


(11) Device をリロードします。

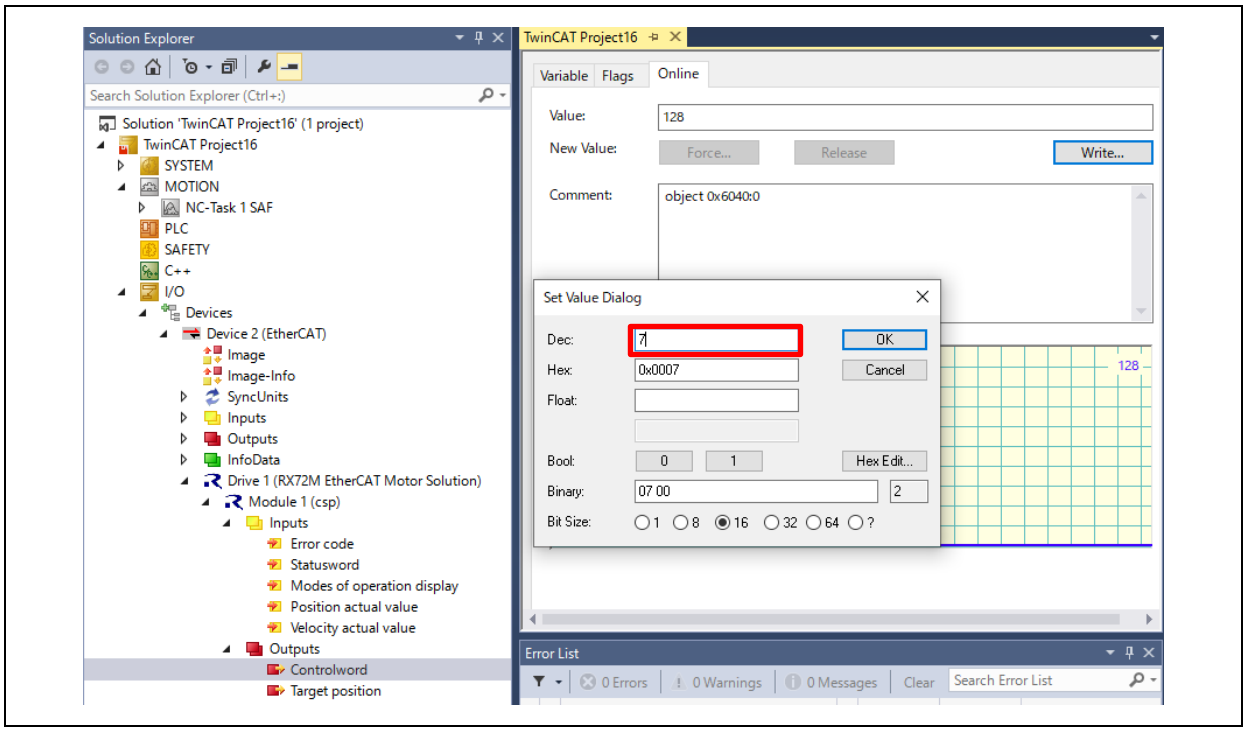


(12) Controlword を 128(Dec)にしてください。

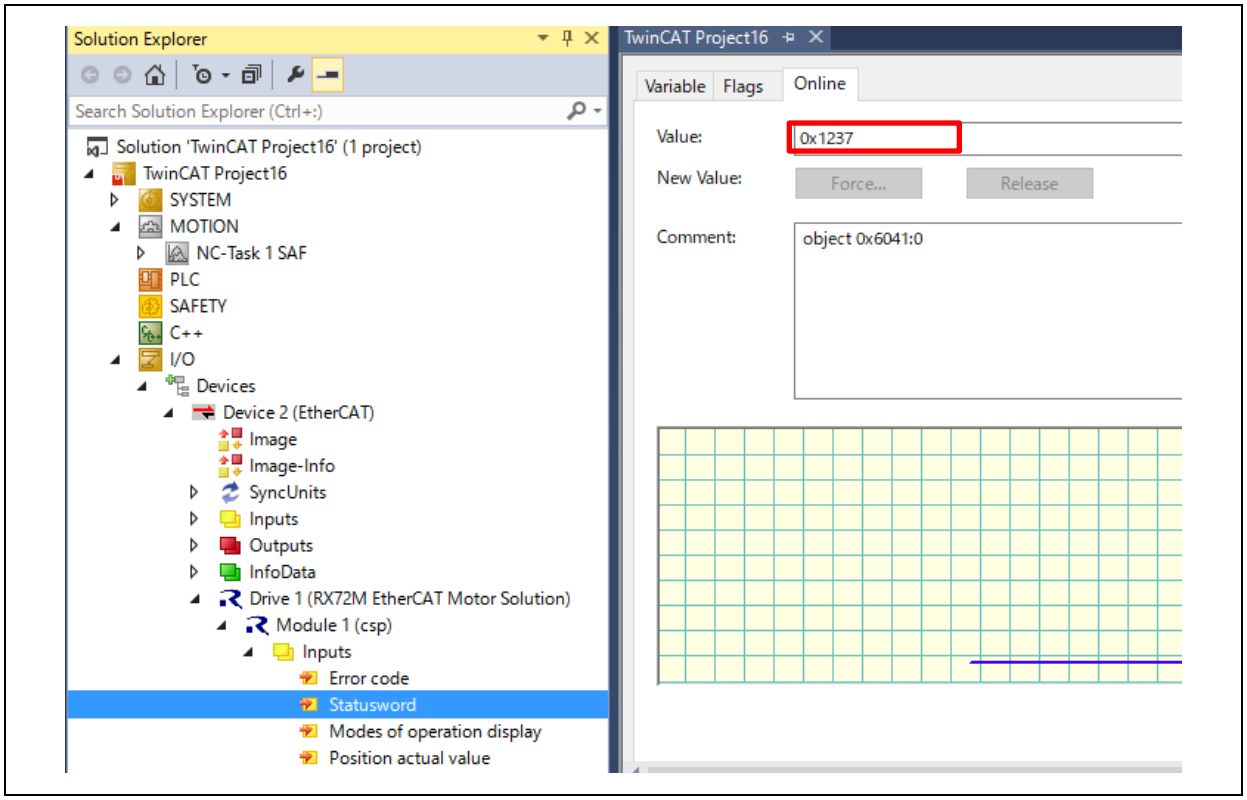
(13) Statusword が 0xX221(Hex)であることを確認します。



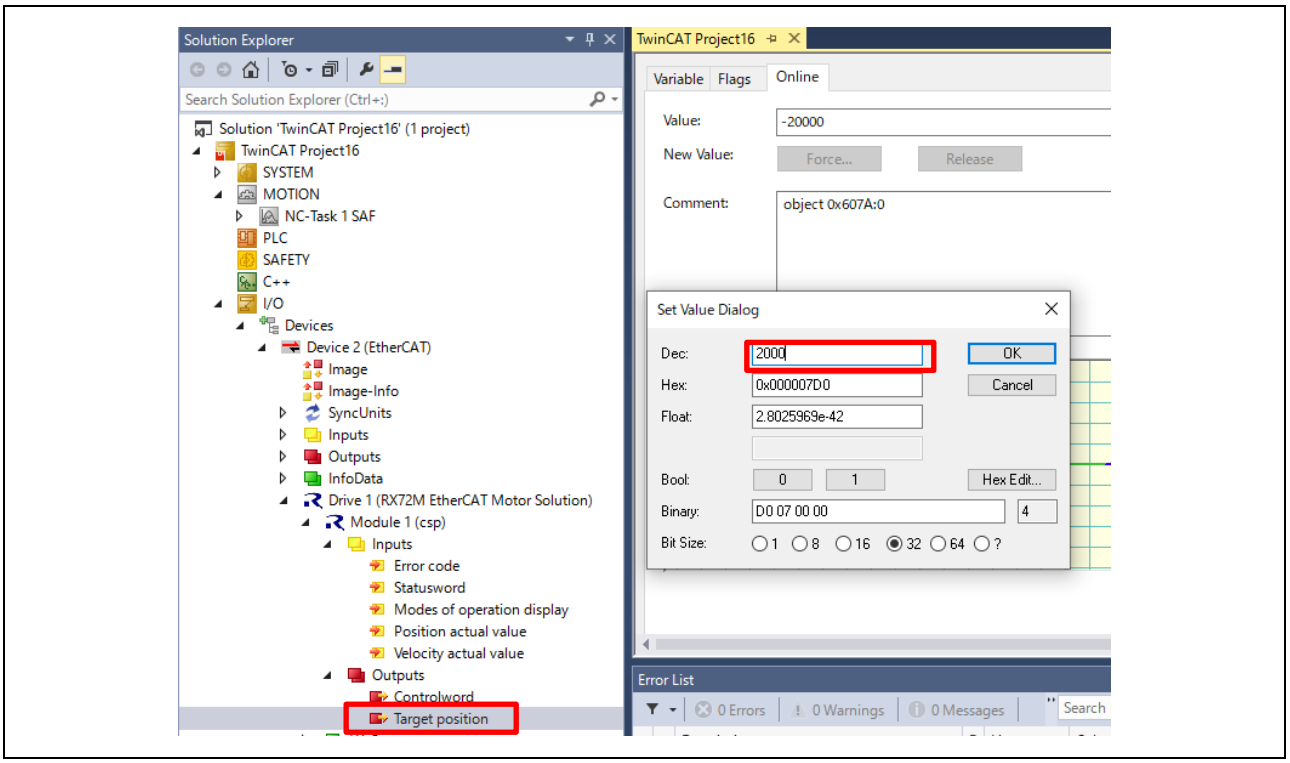
(14) Controlword を 7 → 15 に設定します。[Controlword] → [Online] →[7],[15]



(15) Statusword が 0x1237(Hex)であること、LED1 が点灯していることを確認します。



(16) Target position (velocity) に値を入力することで位置制御が可能となります。



7. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

RX72M グループ ユーザーズマニュアル ハードウェア編（ドキュメント No. R01UH0804）
Renesas Starter Kit+ for RX72M ユーザーズマニュアル（ドキュメント No. R20UT4383）
RX72M グループ 通信ボードハードウェアマニュアル（ドキュメント No. R01AN4661）
（最新版をルネサスエレクトロニクスホームページから入手してください。）

スタートアップマニュアル

RX72M グループ RSK ボード EtherCAT スタートアップマニュアル（ドキュメント No. R01AN4689）
RX72M グループ通信ボード EtherCAT スタートアップマニュアル（ドキュメント No. R01AN4672）
（最新の情報をルネサスエレクトロニクスホームページから入手してください。）

テクニカルアップデート／テクニカルニュース

（最新の情報をルネサスエレクトロニクスホームページから入手してください。）

ユーザーズマニュアル：開発環境

RX ファミリ C/C++コンパイラ、アセンブラ、最適化リンケージエディタコンパイラパッケージ
（R20UT0570）
（最新版をルネサスエレクトロニクスホームページから入手してください。）

8. APPENDIX

8.1 EtherCAT FIT モジュールについて

カスタムボードへの移植など、新規でプロジェクトを作成する場合の EtherCAT FIT モジュールの導入に関する注意点について説明します。

本サンプルプログラムで使用している EtherCAT FIT モジュール(r_ecat_rx V.1.30)はモータ制御プログラムとリソースの競合を回避するため構成に変更が加えられています。

デフォルト構成	本サンプルプログラム	設定
タイマとして FIT CMT ドライバ（r_cmt_rx）を使用	タイマとして Config_TMR2 を使用	タイマのチャネルの競合が生じないように Config_TMR2 を使用

新規でプロジェクトを作成する場合 EtherCAT FIT モジュールを追加すると通常版がデフォルト構成となります。

EtherCAT FIT モジュール(r_ecat_rx)を追加した後、コンフィグレータの追加操作が必要となります。

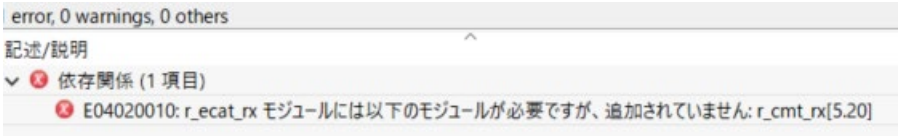
コンポーネント	追加操作	説明
CMT driver r_cmt_rx	コンポーネントから削除する	コンフィグレーションエラーが表示されるが無視して良い
コンペアマッチタイマ Config_CMT2	コンポーネントに追加する	インターバル時間　：1000us 「コンペアマッチ割り込みを許可(CMI2)」をチェックする

コード生成後は本サンプルプログラムの対象フォルダからファイルをコピーしてください。

フォルダ	ファイル	説明
src/smc_gen/Config_CMT2	Config_CMT2_user.c	EtherCAT タイマ処理を割り込みコールバック関数に追加する
src/smc_gen/r_ecat_rx	src フォルダ以下のファイル	リソースの競合を回避したモジュールソースコードに変更

また Compiler のプリプロセッサ・マクロの定義に“ECAT_CMT_CG_USE”を追加してください。

【注意】
r_cmt_rx をコンポーネントから削除することで依存関係のエラーメッセージが表示されますがプログラムの構成には問題ありません。



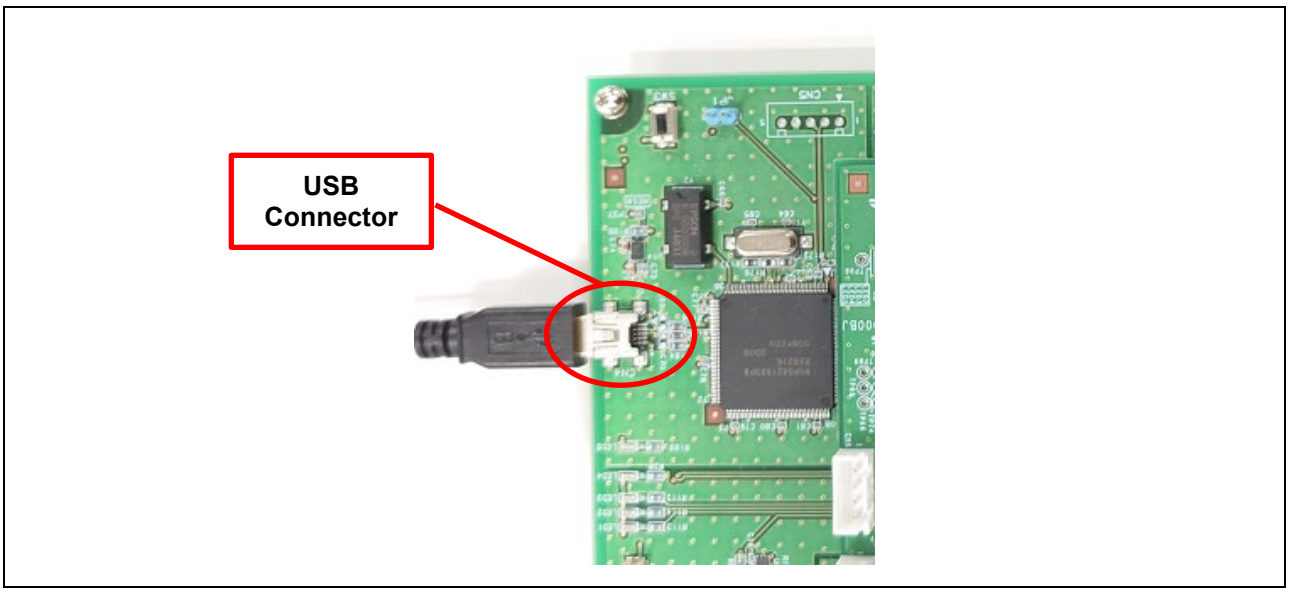
8.2　RMW の接続

本モータボードは RMW(Renesas Motor Workbench)が使用できます。

RMW(Renesas Motor Workbench)関連の手順
Renesas Motor Workbench を下記リンクからダウンロードする
https://www.renesas.com/jp/ja/solutions/proposal/motor-control.html#kits
RMW フォルダにある、RMW UM(R21UZ0004)表 2.1 に従い準備作業を実行する

※認証ファイルは RMW のリンク場所にある「認証ファイルダウンロード」から行う。

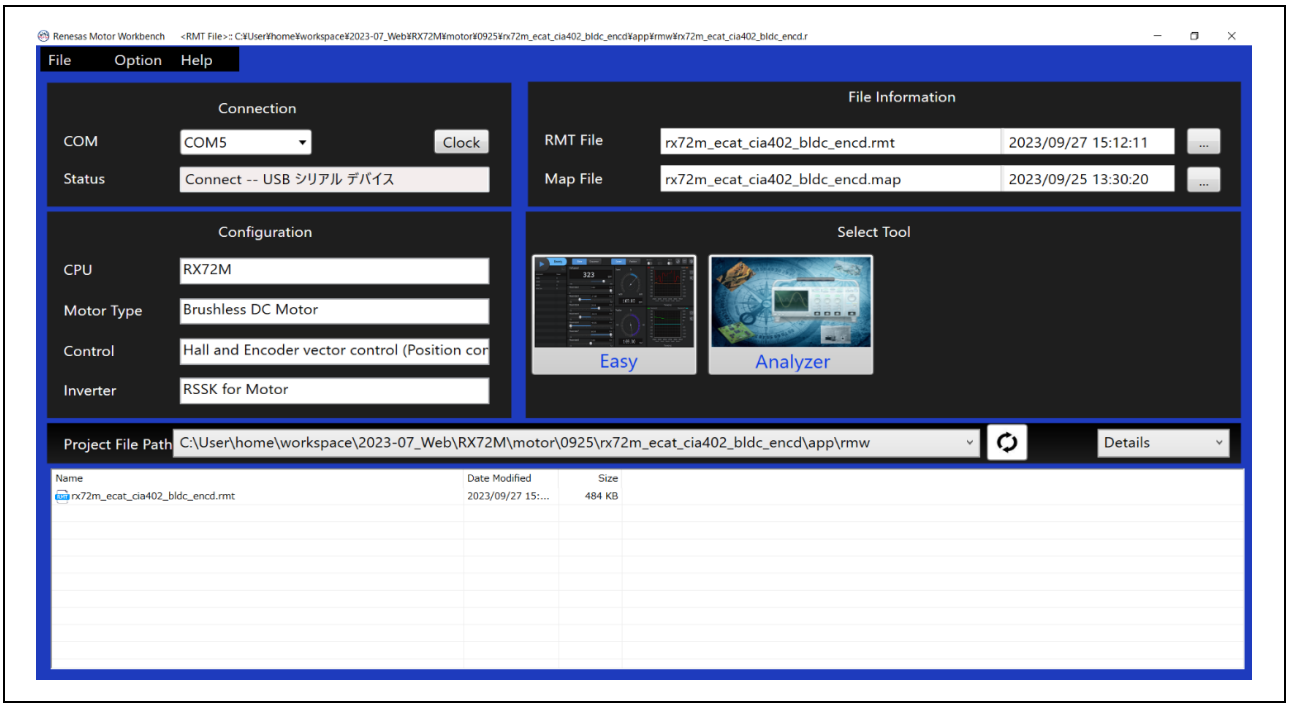
接続はインバータボードの USB1 を使用し、PC の USB ポートと接続してください



RMW を起動して次のファイルを指定します。
map ファイルは e2studio でプロジェクトをインポート後にビルドすると生成されます。

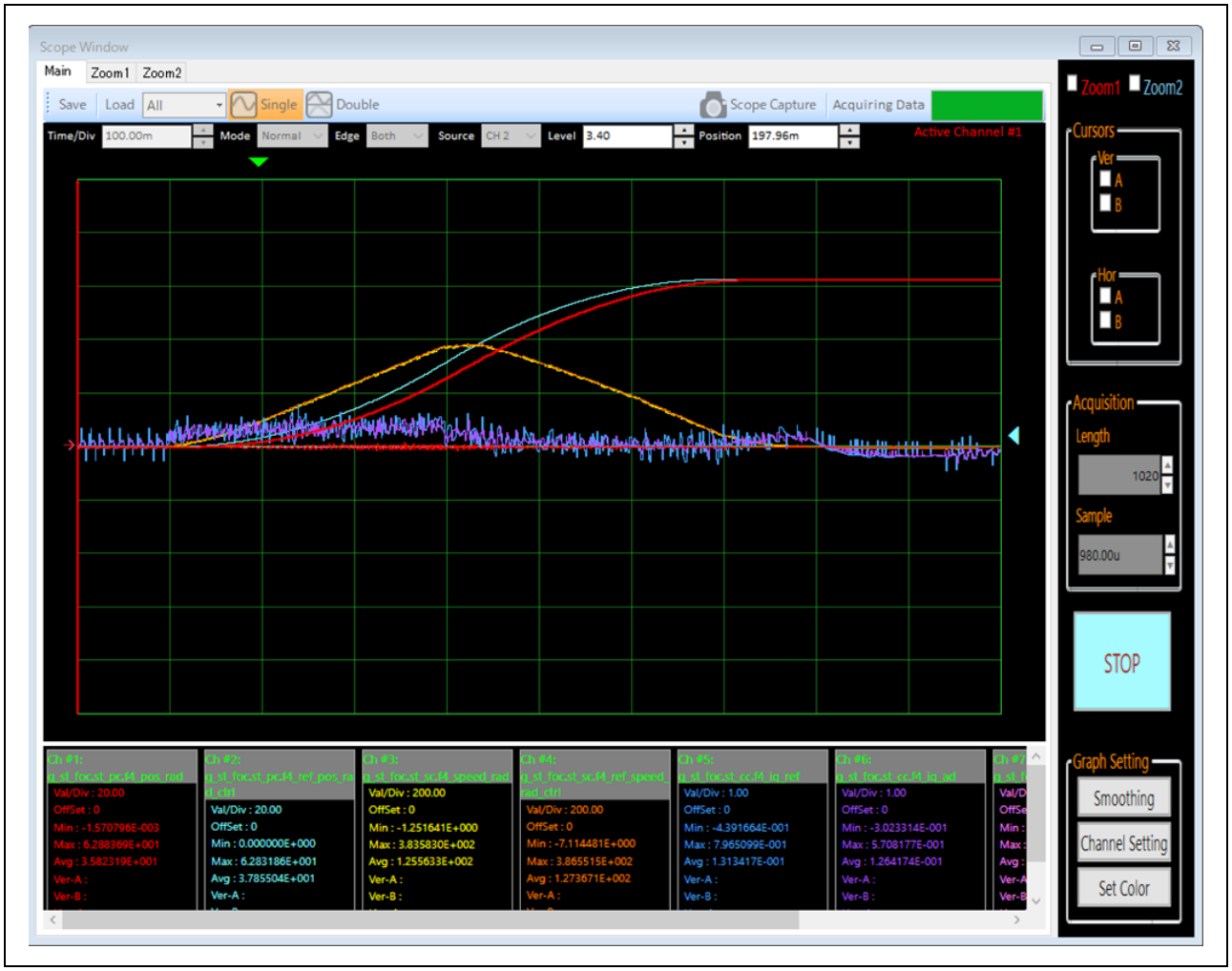
-環境ファイル
rx72m_ecat_cia402_bldc_encd¥project¥app¥rmw
¥rx72m_ecat_cia402_bldc_encd.rmt
-map ファイル
rx72m_ecat_cia402_bldc_encd¥HardwareDebug
¥rx72m_ecat_cia402_bldc_encd.map

RMW とモータ基板を接続する
Connection→COM を指定し、正しく接続すると下記の様になる。



モータ駆動波形を取得する。
「Analyzer」→「Scope Window」の「RUN」ボタンを押す

＊下記の例は Target Posistion を
「0」→「40000」に変更したときの波形。



改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2022/04/28	-	初版発行
1.10	2023/11/30	プログラム	EtherCAT FIT モジュール バージョン 1.30 に対応
			SSC 関連ファイルとプロジェクト関連ファイルを分けるため utilities フォルダと project フォルダを追加
			e2 studio 64 ビット版に対応
			SSC 5.13 に対応
			CTT 2.4.0 に対応
		4	「表 1-1 動作環境」動作環境のバージョンを変更
		5	「表 1-2 ベースプロジェクトと変更点」ベースプロジェクトのバージョンを変更
		10	「表 2-5 ソフトウェア・ファイル構成」フォルダ構成変更に伴いフォルダ名を変更。変更ファイルを追加
		10	「表 2-6 ソフトウェア・モジュール構成」役割の説明を変更
		18	「表 3-3 サポートオブジェクトディクショナリー覧」一部のオブジェクトを変更、Sub 列を追加
		22	「5-1 概要」モータ制御 API ”p_api”を追加
		29	「5-8 p_api」モータ制御 API を追加
		31	「表 6-1　動作環境」動作環境のバージョンを変更
		32	「表 6-2 動作確認コンポーネント」一部のコンポーネントを追加・削除、コンポーネントのバージョンの変更
		31, 32, 35, 36,41	「6 ソリューションキットでの動作確認」フォルダ構成変更に伴いフォルダ名を変更
		39	「6.6 プログラミングとデバッグ」ビルド時の注意点を追加
		53	「8.1 EtherCAT FIT モジュールについて」EtherCAT FIT モジュール V1.30 使用に伴い説明を変更
		55	「8-2 RMW の構成」フォルダ構成変更に伴いフォルダ名を変更。RMW 画面キャプチャを差し替え

製品ご使用上の注意事項

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

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

ご注意書き

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

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

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

(Rev.4.0-1　2017.11)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

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

www.renesas.com/contact/

商標について

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