参考資料

Design Guide: TIDEP-01032

EtherCAT® 接続型、シングルチップ、デュアル サーボ モーター ドライブのリファレンス デザイン

概要

このリファレンス デザインは、フル統合型のリアルタイム サーボ モーター ドライブ制御と産業用通信パスをサポートする、AM243x デバイスの能力を示しています。このパスは、速度に関する EtherCAT® CiA402 ターゲット コマンドを受信してから、デュアル接続のモーターに対する閉ループ FOC 速度制御の実行、実際の速度値のバック アップを EtherCAT PLC に渡すまでの範囲を網羅しています。

参照情報

TIDEP-01032	デザイン フォルダ
AM243x MCU+ SDK	ツール フォルダ
AM243x LaunchPad™	ツール フォルダ
BLDC BP	ツール フォルダ
AM243x Academy	学習用教材

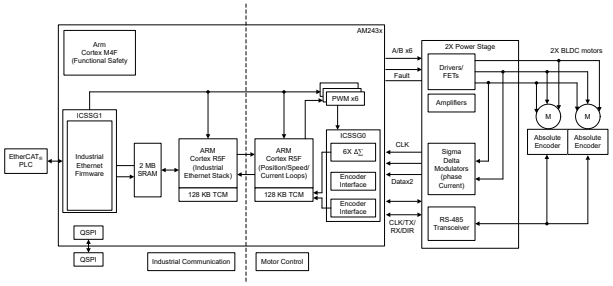
テキサス・インスツルメンツの™ E2E サポート エキスパートにお問い合わせください。

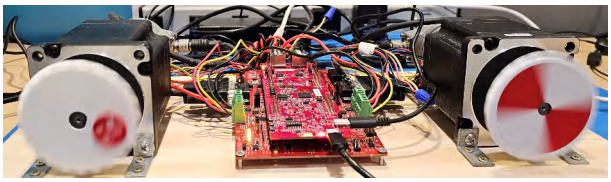
特長

- モーター速度制御用に EtherCAT CiA402 デバイス プロファイルをサポート
- シングルチップ、デュアル サーボ モーター制御
- BOOST-XL TI BoosterPack™ プラグイン モジュールの設計 - AM2x LaunchPad™ 開発キットと互換性のある 80 個のデジタルおよびアナログ I/O
- 24V、8A のモノリシック ゲートドライブ DRV8316R と複数のアンプ ブリッジを使用する、2 軸の 3 相 BLDC モーター ドライブ
- デルタ シグマ変調器 AMC1035D と、電流センス アンプ INA241A を経由する、3 相電流フィードバックの 2 軸パス (6 チャネル)
- 複数の産業用エンコーダ規格をサポートする、RS-485 ベースの 2 軸アブソリュート エンコーダ フィードバック

アプリケーション

- サーボ ドライブ向け通信モジュール
- サーボ ドライブ制御モジュール
- サーボ ドライブ位置フィードバック
- サーボ ドライブ位置センサ
- サーボ ドライブの電力段モジュール





1 システムの説明

このリファレンス デザインでは、包括的なリアルタイム サーボ モーター制御と産業用通信ルートを実現する AM243x デバイスの能力を示します。このルートは、速度に関する EtherCAT CiA402 のターゲット コマンドを受信してから始まり、接続されている 2 台のモーターに対する閉ループ FOC 速度制御の実行に進み、最後に実際の速度値を EtherCAT PLC に送信して終了します。

1.1 用語

PLC	プログラマブル ロジック コントローラ
FOC	フィールド指向制御
EtherCAT	Ethernet for Control Automation Technology の略
CiA402	ドライブとモーション制御用の EtherCAT プロファイル
RPM	回転数 / 分
EnDAT 2.2	増分エンコーダと絶対位置エンコーダ向けのデジタル双方向インターフェイス規格
ICSS	産業用通信サブシステム
PRU	プログラマブル リアルタイム ユニット
LP	LaunchPad™
SDFM	シグマ-デルタ フィルタ モジュール
SDDF	シグマ デルタ デシメーション フィルタリング
IPC	プロセッサ間通信
IEP	産業用イーサネット ペリフェラル
CMP	イベント コンパレータ
ISR	割り込みサービス ルーチン
PWM	パルス幅変調
EPWM	拡張パルス幅変調

1.2 主なシステム仕様

1. サーボ モーターの速度制御を目的として、EtherCAT CiA402 デバイス プロファイルをサポート
2. シングルチップ、デュアル サーボ モーター制御
3. 電流および速度制御用の 50kHz FOC ループ
4. 3 相デルタ シグマ変調電流フィードバックの 2 軸 (6 チャネル)
5. 2 軸 EnDat 2.2 エンコード済みの絶対位置フィードバック

2 システム概要

図 2-1 に、TIDEP-01032 システムのセットアップを示します。

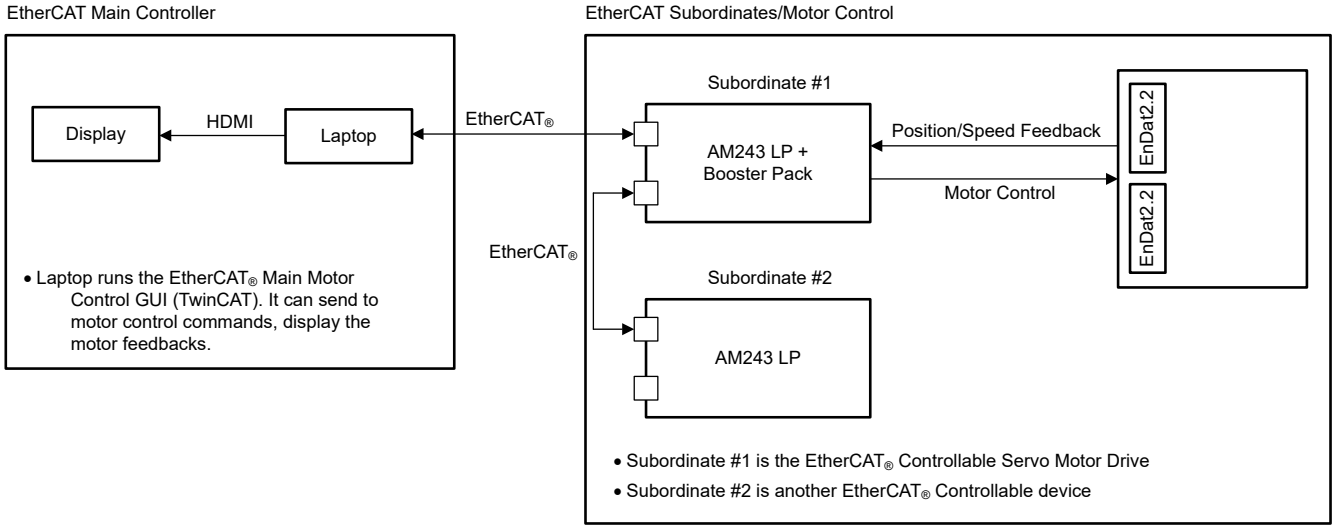


図 2-1. システムのセットアップ

2.1 ブロック図

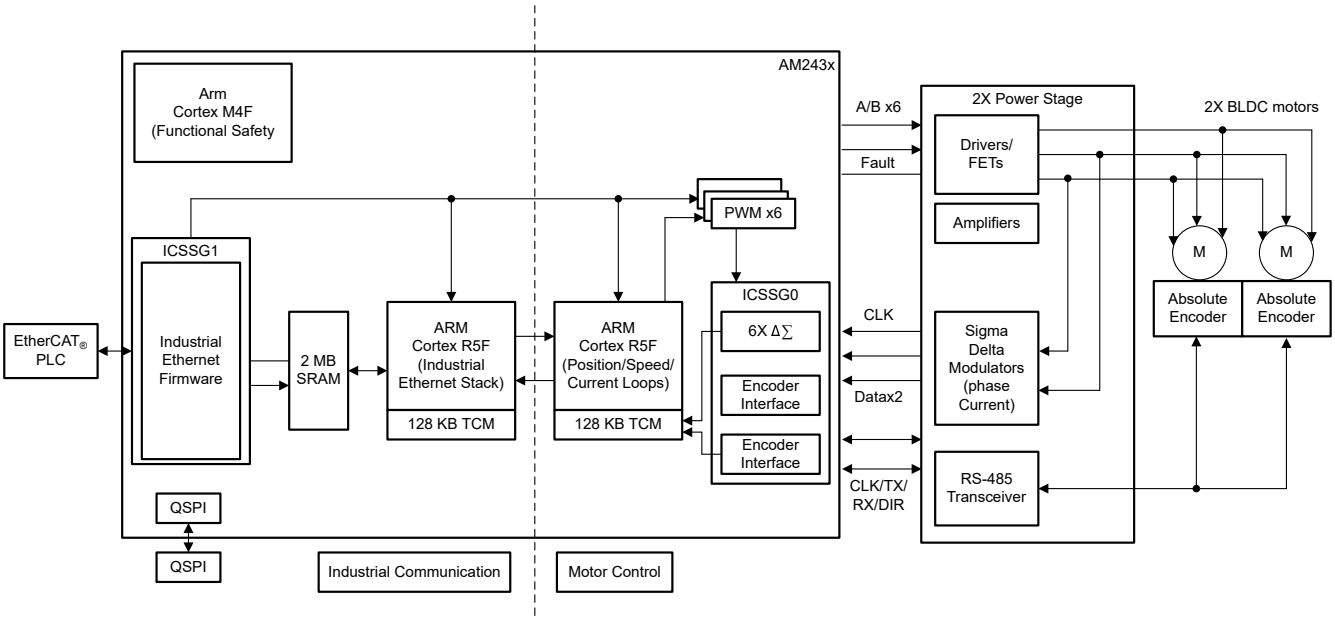


図 2-2. TIDEP-01032 のブロック図

2.2 設計の考慮事項

シングルチップ 2 軸サーボ モータードライブの実装は、以下で構成される集中型のリアルタイム パスを中心に設計されています。

- ICSSG1 - EtherCAT クライアント コントローラのファームウェア
- ICSSG0 - SDDF と EnDAT 2.2 のデコード
  - 直接接続された 2 台のモーターから位相電流をフィードバックするために PRU0 内にある RTU と PRU コアの間で負荷共有を行うデルタ シグマ フィルタリング ファームウェア

  - 直接接続された 2 台のアブソリュート エンコーダから角度、位置、速度をフィードバックするために PRU1 内の RTU コアと PRU コアの間で負荷共有を行う EnDat2.2 デコーディング ファームウェア
- R5FSS1_0 - FreeRTOS を使用して CiA402 を実装する EtherCAT クライアント スタック
- R5FSS0_0 および R5FSS0_1 – アブソリュート エンコーダを使用して、直接接続された 2 台のモーターの電流、速度、または位置の閉ループ制御が可能な 2 つの独立した閉ループ FOC
- MCU+ SDK の IPC Notify により、低レイテンシのコア間同期および通信を実現
- EPWM – 6 チャネルの拡張 PWM ペリフェラルにより 2 つの FOC ループの出力に基づいて波形を生成

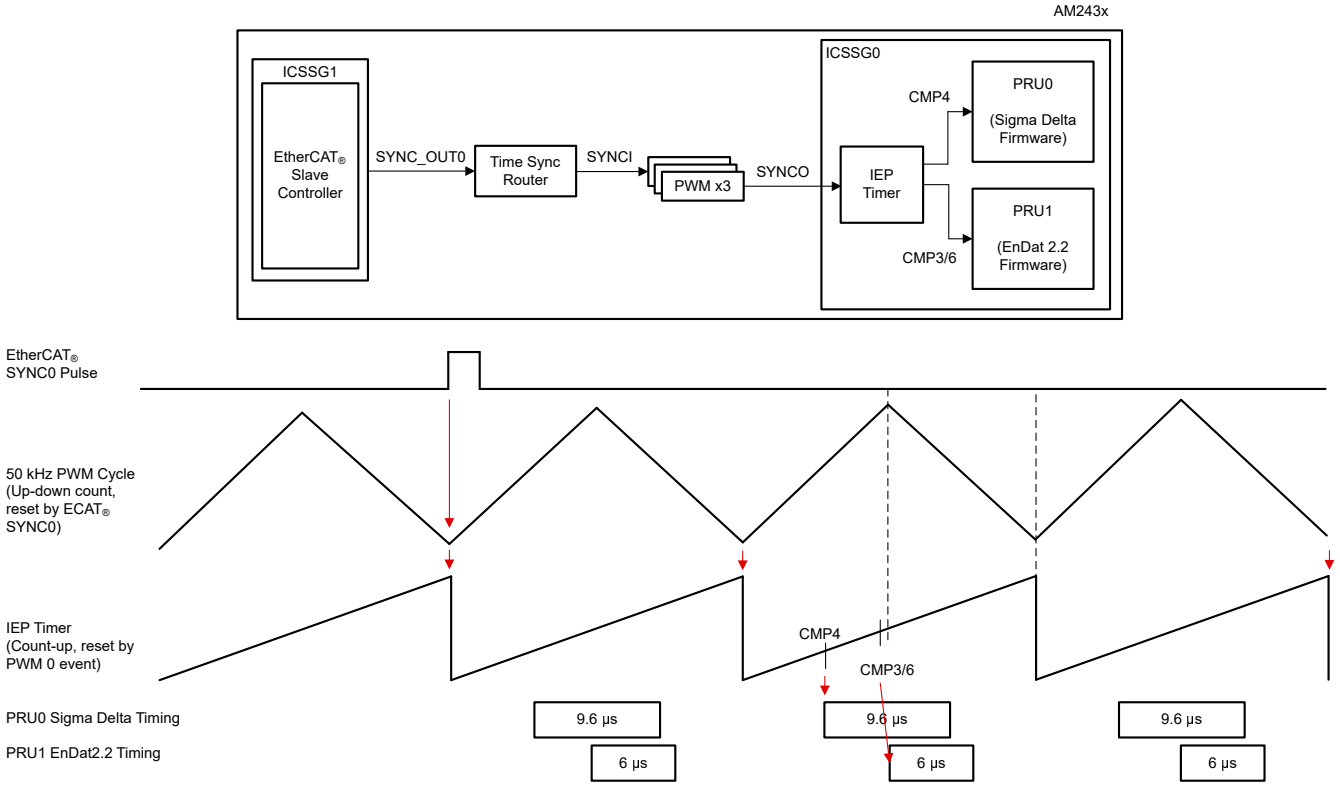


図 2-3. トリガ FOC タイミング – 50kHz

2.3 主な使用製品

2.3.1 EnDat 2.2 インターフェイス

表 2-1 に、EnDat 2.2 信号パラメータを示します。

表 2-1. EnDat 2.2 信号 (2 つの 4 線式エンコーダ)

AM243x LP (ピン番号)	BP コネクタ	BLDC BP	信号名
GPIO1_78(C16)	J8.73	VSENSOR1_SW_EN	Encoder1 イネーブル
PRG0_PRU1_GPO0(L5)	J2.11	ENCODER_CLK1	Encoder1 クロック
PRG0_PRU1_GPO2 (M2)	J7.68	ENCODER_DATA_TX_EN1	Encoder1 TX イネーブル
PRG0_PRU1_GPO1(J2)	J7.67	ENCODER_DATA_TX1	Encoder1 の評価基板 (TX)
PRG0_PRU1_GPO13(T4)	J8.71	ENCODER_DATA_RX1	Encoder1 の評価基板 (RX)
GPIO1_77(B17)	J8.74	VSENSOR2_SW_EN	Encoder2 イネーブル
PRG0_PRU1_GPO6(F5)	J7.69	ENCODER_CLK2	Encoder2 クロック
PRG0_PRU1_GPO8(F4)	J6.57	ENCODER_DATA_TX_EN2	Encoder2 TX イネーブル
PRG0_PRU1_GPO12(P2)	J8.72	ENCODER_DATA_TX2	Encoder2 の評価基板 (TX)
PRG0_PRU1_GPO11(P1)	J7.70	ENCODER_DATA_RX2	Encoder2 の評価基板 (RX)

EnDat 2.2 割り込み:

- hwiPrms.intNum = ICSSG_PRU_ENDAT_INT_NUM | ICSSG_PRU_ENDAT_INT_NUM+2;
- hwiPrms.callback = &pruEncoderIrqHandler | &pruEncoderIrqHandler2;
  - モータ制御ループ (FOC)、モータごとにコア 1 つ

EnDat 2.2 入力データ バッファ:

- R5F_0_0 TCMB 内の gEndatChInfo (.gEncChData)

ICSSG ピン MUX:

- モード (ICSSG_GPCFG0_REG[29-26]: PR1_PRU0_GP_MUX_SEL = 1h)
- ICSSG_SA_MX_REG[7] G_MUX_EN = 0

2.3.2 SDFM インターフェイス

図 2-4 にクロック ソースの分配を示し、表 2-2 に SDFM 信号を示します。

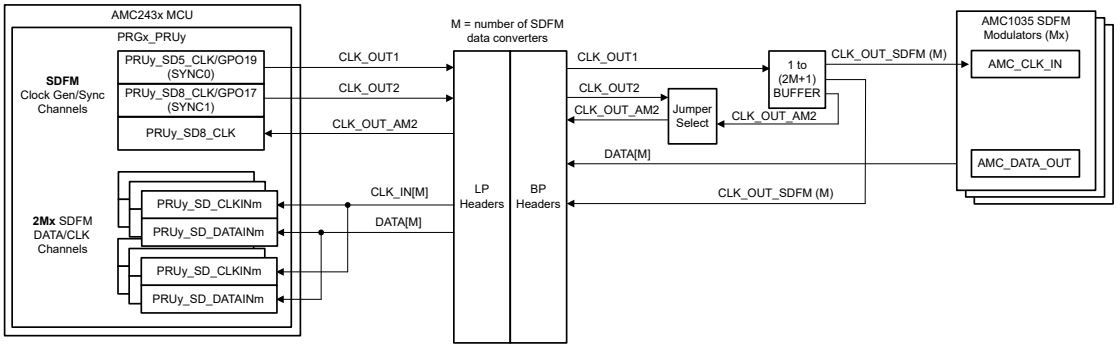


図 2-4. クロック ソースの分配

表 2-2. SDFM 信号

AM243x LP (ピン番号)	BP コネクタ	BLDC BP	信号名
PRG0_PRU0_GPI1(J4)	J4.32	SDFM 電流高 A1	SDFM 電流高 A1
PRG0_PRU0_GPI3(H1)	J2.19	SDFM 電流高 B1	SDFM 電流高 B1
PRG0_PRU0_GPI5(F2)	J2.13	SDFM 電流高 C1	SDFM 電流高 C1
PRG0_PRU0_GPI7(E2)	J5.44	SDFM 電流高 A2	SDFM 電流高 A2
PRG0_PRU0_GPI8(H5)	J2.15	SDFM 電流高 B2	SDFM 電流高 B2
PRG0_PRU0_GPO11(L1)	J2.12	SDFM 電流高 C2	SDFM 電流高 C2

SDFM クロック:

- PRG0_PRU0_GPO19(G2) → SYNC0 - SDFM CLOCK_OUT1(J5.45) → SDFM_CLOCK_SOURCE[1|2] (TP31)
  - AMC_CLKIN_A1, AMC_CLKIN_B1, AMC_CLKIN_C1, PR0_PRU0_SD0_CLK(J4.33), PR0_PRU0_SD1_CLK(J4.31), PR0_PRU0_SD2_CLK(J2.17)
  - AMC_CLKIN_A2, AMC_CLKIN_B2, AMC_CLKIN_C2, PR0_PRU0_SD3_CLK(J5.48), PR0_PRU0_SD6_CLK(J1.5), PR0_PRU0_SD7_CLK(J2.14)

SDFM 割り込み:

- hwiPrms.intNum = ICSSG_PRU_SDDF_INT_NUM | ICSSG_RTU_SDDF_INT_NUM;
- hwiPrms.callback = &pruSddfIrqHandler | &rtuSddfIrqHandler;

SDFM 入力データ バッファ:

- モーター 1 の R5F_0_0 の TCMB 内にある gSddfChSamps[0-2] (.gSddfChSampsRaw)
- モーター 2 の R5F_0_1 の TCMB 内にある gSddfChSamps[3-5] (.gSddfChSampsRaw)

2.3.3 EPWM インターフェイス

表 2-3 および 表 2-4 に、それぞれ EPWM0-2 信号モーター 1 および EPWM3-5 信号モーター 2 のデータを示します。

表 2-3. EPWM0-2 信号モーター 1

AM243x LP (ピン番号)	BP コネクタ	BLDC BP	信号名
GPIO1_64(B16)	J5.49	nPWM_EN_M1	DRV1 イネーブル
GPMC0_AD8(U18)	J4.36	DRV1 EPWM High C	DRV1 PWM High C
GPMC0_AD9(U20)	J4.35	DRV1 EPWM Low C	DRV1 PWM Low C
GPMC0_AD5(T20)	J4.38	DRV1 EPWM High B	DRV1 PWM High B
GPMC0_AD6(T18)	J4.37	DRV1 EPWM Low B	DRV1 PWM Low B
GPMC0_AD3(V21)	J4.40	DRV1 EPWM High A	DRV1 PWM High A
GPMC0_AD4(U21)	J4.39	DRV1 EPWM Low A	DRV1 PWM Low A

表 2-4. EPWM3-5 信号モーター 2

AM243x LP (ピン番号)	BP コネクタ	BLDC BP	信号名
GPIO1_65(B15)	J5.50	nPWM_EN_M2	DRV2 イネーブル
FSI_TX0_CLK (P21)	J8.79	DRV2 EPWM High C	DRV2 PWM High C
FSI_TX0_D0(Y18)	J8.80	DRV2 EPWM Low C	DRV2 PWM Low C
TEST_LED3_RED(D1)	J8.75	DRV2 EPWM High B	DRV2 PWM High B
TEST_LED4_GREEN(F3)	J8.76	DRV2 EPWM Low B	DRV2 PWM Low B
TEST_LED1_GREEN(U19)	J8.77	DRV2 EPWM High A	DRV2 PWM High A
FSI_RX0_D1(V20)	J8.78	DRV2 EPWM Low A	DRV2 PWM Low A

EPWM 設定 (**init_pwms**):

- 3 つの PWM グループを結合するように SYNCI、SYNCO のマッピングを構成します。
- 時間同期ルータ 38 から PWM0 を同期
  - CSL_REG32_WR(CSL_TIMESYNC_EVENT_INTROUTER0_CFG_BASE + ((38 × 4) + 4), (0x10000 | 29));
- 時間同期ルータ入力 29 (ICSSG1 IEP0 SYNC0) → 時間同期ルータ出力 38
  - CSL_REG32_WR(CSL_CTRL_MMR0_CFG0_BASE + CSL_MAIN_CTRL_MMR_CFG0_EPWM0_CTRL, (2 << CSL_MAIN_CTRL_MMR_CFG0_EPWM0_CTRL_SYNCIN_SEL_SHIFT));
  - TIMESYNC_INTRTR0_IN_29:PRU_ICSSG1_PR1_EDC0_SYNC0_OUT_0 (IEP0 同期イベント 0)
  - timesync_event_introuter_out_38: epwm0_sync.input2
  - TI E2E:[FAQ] AM64x：*時間同期ルータは何のためのものですか？それはどのように使うのですか？*
- 時間同期ルータ 39 から PWM3 を同期
  - CSL_REG32_WR(CSL_CTRL_MMR0_CFG0_BASE + CSL_MAIN_CTRL_MMR_CFG0_EPWM3_CTRL, (2 << CSL_MAIN_CTRL_MMR_CFG0_EPWM3_CTRL_SYNCIN_SEL_SHIFT));
- 時間同期ルータ入力 29 (ICSSG1 IEP0 SYNC0) → 時間同期ルータ出力 39
  - CSL_REG32_WR(CSL_TIMESYNC_EVENT_INTROUTER0_CFG_BASE + ((39 × 4) + 4), (0x10000 | 29));
  - TIMESYNC_INTRTR0_IN_29:PRU_ICSSG1_PR1_EDC0_SYNC0_OUT_0 (IEP0 同期イベント 0)
  - timesync_event_introuter_out_39: epwm3_sync.input2
- 時間同期ルータ 40 から PWM6 を同期
  - CSL_REG32_WR(CSL_CTRL_MMR0_CFG0_BASE + CSL_MAIN_CTRL_MMR_CFG0_EPWM6_CTRL, (2 << CSL_MAIN_CTRL_MMR_CFG0_EPWM6_CTRL_SYNCIN_SEL_SHIFT));

- 時間同期ルータ入力 29 (ICSSG1 IEP0 SYNC0) → 時間同期ルータ出力 40
  - CSL_REG32_WR(CSL_TIMESYNC_EVENT_INTROUTER0_CFG_BASE + ((40 * 4) + 4), (0x10000 | 29));
  - TIMESYNC_INTRTR0_IN_29：PRU_ICSSG1_PR1_EDC0_SYNC0_OUT_0 (IEP0 同期イベント 0)
  - timesync_event_introuter_out_40: epwm6_sync.input2
  - SW を強制的に EPWM0 に同期します。ハードウェア同期のデイジー チェーンにより他の PWM を同期します。
    - Epwm_tbTriggerSwSync(gEpwm0BaseAddr);
      - HW_WR_FIELD16(((gEpwm0BaseAddr + PWMSS_EPWM_OFFSET) + PWMSS_EPWM_TBCTL), PWMSS_EPWM_TBCTL_SWFSYNC, (uint16_t)PWMSS_EPWM_TBCTL_SWFSYNC_FORCE_SYNC);
- EPWM を 50kHz に設定：
  - appEpwmCfg.epwmOutFreq = gEpwmOutFreq;
  - App_epwmConfig(&appEpwmCfg, &epwm2PrdVal, &epwm2CmpAVal);

EPWM0 割り込み：

- hwiPrms.intNum = EPWM0_INTR;
- hwiPrms.callback = &App_epwmIntrISR;

EPWM0 出力データ：

- gEpwmPrdVal

2.3.4 ICSS-PRU IEP

IEP CMP セットアップには、次のパラメータが適用されます。

- 50kHz の EPWM サイクル時間
- 50kHz で FOC ループを更新 (EnDat ISR を使用)
- EtherCAT クライアント (ICSSG1) からの SYNC_OUT0 で EPWM クロックを同期
- PRU_ICSSG0 IEP0 周期を 6000 (300000000/50000) に設定
  - また、これは EPWM 周期でもあります。
  - PRU_ICSSG IEP0 ベースのアドレスは 0x3002E000
- シグマ デルタ エンコードされた電流フィードバック データのサンプリングをトリガするように CMP4 を設定：
  - IEP または EPWM 周期ごとに 1 つの CMP4：10us (gTestSdfmPrms.firstSampTrigTime で定義)
  - initPruSddf で設定
- EnDAT 2.2 エンコードされた位置フィードバック データのサンプリングをトリガするように CMP3 と CMP6 を設定：
  - IEP または EPWM 周期ごとに 1 つの CMP3 および CMP6：3000ns (endat_periodic_interface.cmp3 および endat_periodic_interface.cmp6 で定義)
  - endat_config_periodic_mode で設定

2.3.5 EtherCAT CiA402 速度制御

目標速度 (EtherCAT コントローラ → EtherCAT 配下→ AL243x LP)：

- キャッシュされていないオンチップ RAM 内の gCurTargetVelocity[3] (.gEtherCatCia402)
- GUI を使用して EtherCAT コントローラで設定
- EtherCAT 配下に送信
- EC_SLV_APP_CSV 経由で gCurTargetVelocity[3] に保存
- pruEncoderIrqHandler または pruEncoderIrqHandler2 で速度制御用に使用

実際の速度 (EtherCAT コントローラ ← EtherCAT 配下 ← AL243x LP) ：

- キャッシュされていないオンチップ RAM 内の gCurActualVelocity[3] (.gEtherCatCia402)
- 実際の速度については、pruEncoderIrqHandler または pruEncoderIrqHandler2 によって gCurTargetVelocity[3] に保存
- EC_SLV_APP_CSV 経由で EtherCAT コントローラに送信
- EtherCAT コントローラにより GUI に表示

3 システム設計

R5F_0_1 の初期化

モーター 1 の R5F_0_0 を初期化するには、次の手順に従います (single_chip_servo_remote_core_start)。

1. GPIO ピンの方向と初期値 を設定 (init_gpio_state)
2. EPWM ディスエーブル (enable_pwm_buffers)
3. モーター 1 の EPWM 周波数と割り込みを設定 (init_pwms)
4. EnDAT 2.2 モーター 1 の ICSSG0 PRU1 を設定 (チャネル 0、init_encoder)
   - ICSSG_SA_MX_REG レジスタで g_mux_en を 1 に設定 HW_WR_REG32((CSL_PRU_ICSSG0_PR1_CFG_SLV_BASE+0x40), (0x80))
   - レジスタ & イネーブル ICSSG EnDat PRU FW 割り込み
   - – 割り込み番号: ICSSG_PRU_ENDAT_INT_NUM
     – コールバック関数:pruEncoderIrqHandler
   - EnDat 2.2 パラメータの設定
     – gEndat_multi_ch_mask = ENDAT_MULTI_CH0 | ENDAT_MULTI_CH2;
     – gEndat_is_multi_ch = CONFIG_ENDAT0_MODE & 1;
     – gEndat_is_load_share_mode = CONFIG_ENDAT0_MODE & 2;
   - ICSSG0 PRU1 を初期化 (EnDat_pruss_init)
   - エンコーダ ドライバ API を使用してエンコーダを初期化 (endat_init)
   - エンコーダ ドライバ API を使用してエンコーダを設定 (endat_config_multi_channel_mask)
   - ICSSG 周波数に基づいて遅延を構成
   - EnDat 2.2 PRU FW を ICSSG0 PRU1 にロードして実行 (endat_pruss_load_run_fw)
   - 5 秒のタイムアウトを指定して、ファームウェアからの初期化 ack を確認 (endat_wait_initialization)
   - 2.2 エンコーダのデフォルト周波数を 16 MHz に設定 (endat_init_clock)
   - 300MHz PRU で 16MHz が動作するように伝播遅延を設定 (endat_handle_prop_delay(priv, 265))
   - EnDat 2.2 FW に周期的なトリガを設定 (endat_config_periodic_trigger)
   - EnDat 2.2 FW 周期モードのパラメータを設定 (endat_config_periodic_mode)
     – チャネル 0 の IEP0 CMP3 イベント (PWM 周期の開始から 3000ns)
     – チャネル 2 の IEP0 CMP6 イベント (PWM 周期の開始から 3000ns)
   - EnData 2.2 データの受信を開始 (endat_handle_rx)
5. モーター 1 の SDFM 用に ICSSG0 PRU0 を設定 (init_sddf)
   - IEP0 を初期化し、SYNC0 SD クロックを設定 (init_IEP0_SYNC)
   - ICSSG0 PRU0 の初期化 (initIcss)
   - ICSSG SDFM RTU FW 割り込みの登録とイネーブル
     – 割り込み番号:ICSSG_RTU_SDDF_INT_NUM
     – コールバック関数:rtuSddfIrqHandler
   - SDFM の RTU/PRU コアを初期化 (initPruSddf)
     – RTU サンプル ベース アドレス:gTestSdfmPrms.samplesBaseAddress
     – PRU サンプル ベース アドレス:gTestSdfmPrms.samplesBaseAddress+0x80
   - IEP0 を起動 (start_IEP0)
   - SW を強制的に EPWM0 に同期他の PWM は、ハードウェア同期デイジー チェーンを介して同期 (EPWM_tbTriggerSwSync)
   - EPWM 出力バッファのディスエーブル (enable_pwm_buffers)
6. FOC のパラメーターを初期化 (init_pids)
7. モーター 1 の EPWM 出力バッファをイネーブル (enable_pwm_buffers(TRUE))

R5F_0_1 の初期化

モーター 2 の R5F_0_1 を初期化するには、次の手順に従います (single_chip_servo_remote_core_start)。

1. モーター 2 の EPWM 周波数と割り込み (init_pwms) のセットアップ

2. モーター 2 の ICSSG EnDat PRU FW 割り込みを登録してイネーブル
   - 割り込み番号: ICSSG_PRU_ENDAT_INT_NUM+2
   - コールバック関数:pruEncoderIrqHandler2
3. モーター 2 の ICSSG SDFM PRU FW 割り込みイネーブル & を登録
   - 割り込み番号: ICSSG_PRU_SDDF_INT_NUM
   - コールバック関数:pruSddfIrqHandler
4. FOC のパラメーターを初期化 (init_pids)

割り込みのセットアップ

割り込みと**ハンドラ**を設定するには、次の手順に従います。

EPWM 割り込み (50kHz)、ISR - App_epwmIntrISR (モーター 1) または App_epwmIntrISR2 (モーター 2)

- EPWM 割り込みのクリア

SDFM 割り込み (50kHz)、ISR - rtuSddfIrqHandler (モーター 1) または pruSddfIrqHandler (モーター 2)

- サンプル 8192 から 16384 の範囲で、SDFM チャネルのオフセットを計算します (0–2 または 3–5)。PRECOMPUTE_LEVEL == NO_PRECOMPUTE の場合、SDFM チャネル オフセットが FOC ループで使用されます。
- サンプル 16384 で、位相 A、位相 B、位相 C の EPWM を書き込み、回転子を電気的 0 にロックし、SDFM 割り込みをディセーブル
- ソースでの割り込みをクリア

EnDAT 2.2 割り込み (50kHz)、ISR – pruEncoderIrqHandler (モーター 1)

1. ソースでの割り込みをクリア
2. サンプル 0 - 8192 の場合は何もしていない
3. サンプル 8193-16383 の場合:
   - 機械的および電気的角度オフセットの計算 (localEnDatGetSingleMulti)
4. サンプル 16384 の場合:
   - 機械的および電気的角度オフセットの平均を求めます
   - すべての位相をオフ
   - 機械的および電気的角度オフセットを保存
5. 16384 より後のサンプルの場合:
   - FOC ループを開始し、回転子のロックを解除
   - エンコーダから最新の機械的 θ と多回転位置を取得 (localEnDatGetSingleMulti)
   - 電気的な 0、4 ポールのペアから計算されたオフセットを使用
   - FOC ループを実行して空間ベクトルを計算
   - 次の CMPA 値を書き込み HW は EPWM0 を位相 C に、EPWM2 を位相 A に接続するので、cmp0 と cmp2 を交換します
   - EPWM0 は実際には EHRPWM2 を使用し、EPWM1 は EHRPWM1 を、EPWM2 は EHRPWM0 を使用
   - 詳細については、single_chip_servo_am243x-lp_r5fss0-0_nortos_ti-arm-clang の example_syscfg の EPWM 設定を参照してください。

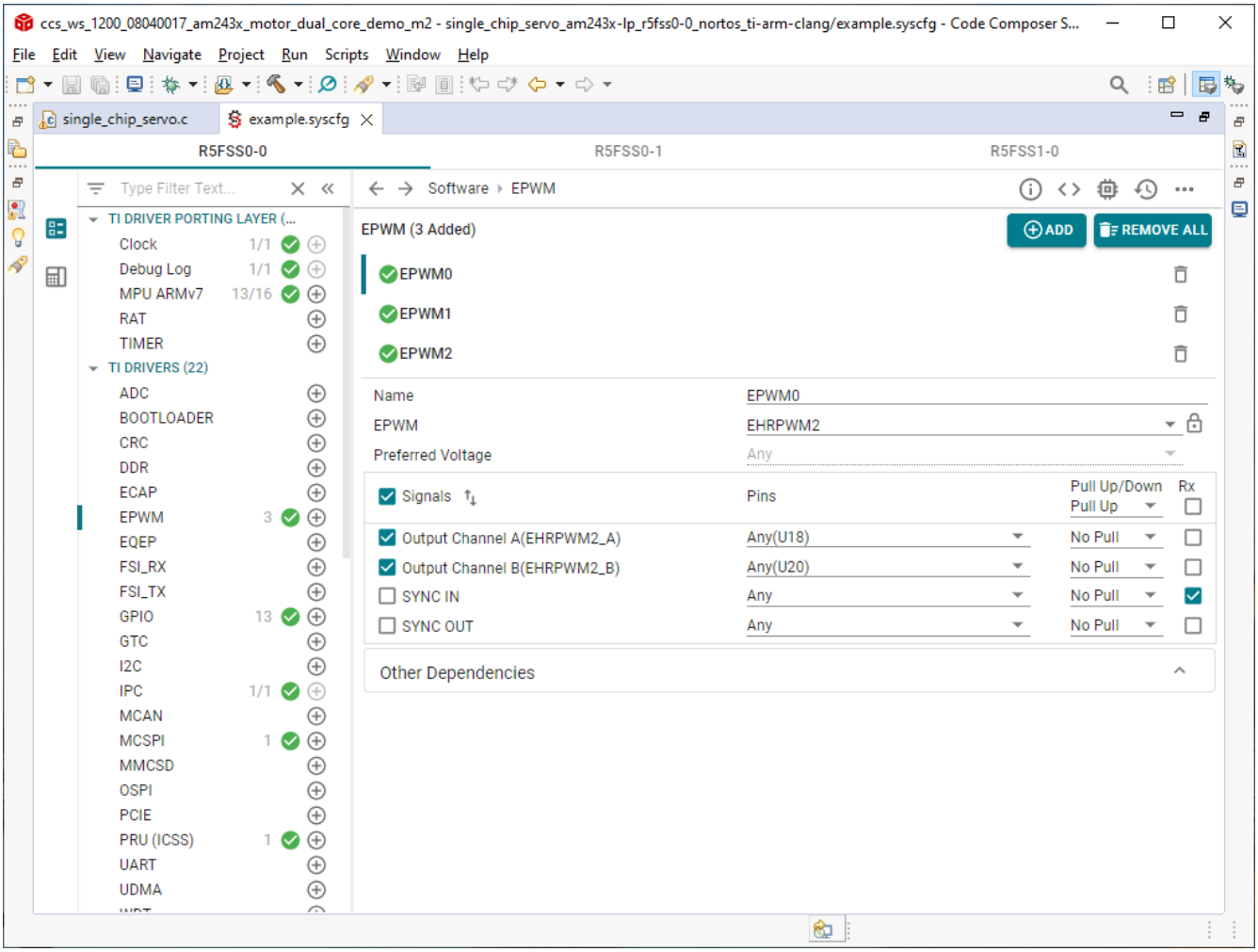


図 3-1. モーター 1 の EPWM 設定

EnDAT 2.2 割り込み (50kHz)、ISR – pruEncoderIrqHandler (モーター 2)

- ソースでの割り込みをクリア
- サンプル 0-8192 の場合は何もしない
- サンプル 8193-16383 の場合:
  - 機械的および電気的角度オフセットの計算 (localEnDatGetSingleMulti)
- サンプル 16384 の場合:
  - 機械的角度オフセットと電気的角度オフセットの平均を求めます。
  - すべての位相をオフ
  - 機械的および電気的角度オフセットを保存
- 16384 より後のサンプルの場合:
  - FOC ループを開始し、回転子のロックを解除
  - エンコーダから最新の機械的 θ と複数回転位置を取得 (localEnDatGetSingleMulti)
  - 電気的な 0、4 ポールのペアから計算されたオフセットを使用します
  - FOC ループを実行して空間ベクトルを計算
  - 次の CMPA 値を書き込みます。HW は EPWM0 を位相 C に、EPWM5 を位相 A に接続するので、cmp0 と cmp2 を交換します
  - EPWM0 は実際には EHRPWM5 を使用し、EPWM1 は EHRPWM4 を使用し、EPWM2 は EHRPWM3 を使用
  - 詳細については、single_chip_servo_am243x-lp_r5fss0-1_nortos_ti-arm-clang の example_syscfg の EPWM 設定を参照してください。

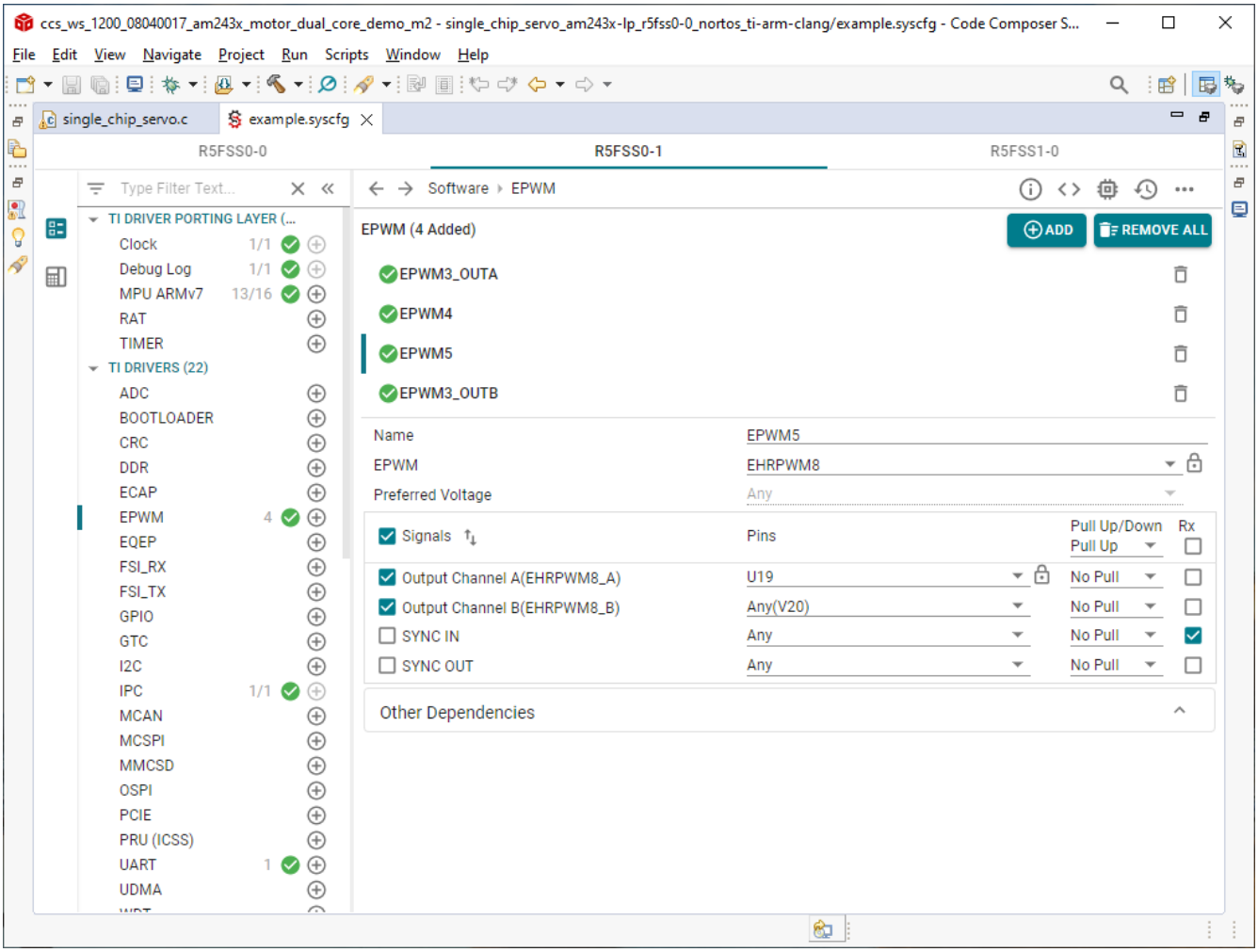


図 3-2. モーター 2 の EPWM 設定

4 ハードウェア、ソフトウェア、テスト要件、テスト結果

4.1 ハードウェア要件

このリファレンス デザインをテストするには、以下の装置が必要です。

- TwinCAT 自動化ソフトウェアがインストールされた Microsoft® Windows® パーソナルコンピュータ (PC) 1 台
- LP-AM243 評価ボード | TI.com 1 枚
- BP-AM2BLDCSERVO — AM2x ブラシレス DC (BLDC) サーボ モーター ブースタパック 1 つ
- BLY342D-48V-3200 Anaheim Automation 3 相ブラシレス DC モーター 2 台
- ケーブル付き ROQ-437 EnDat2.2 エンコーダ 2 台

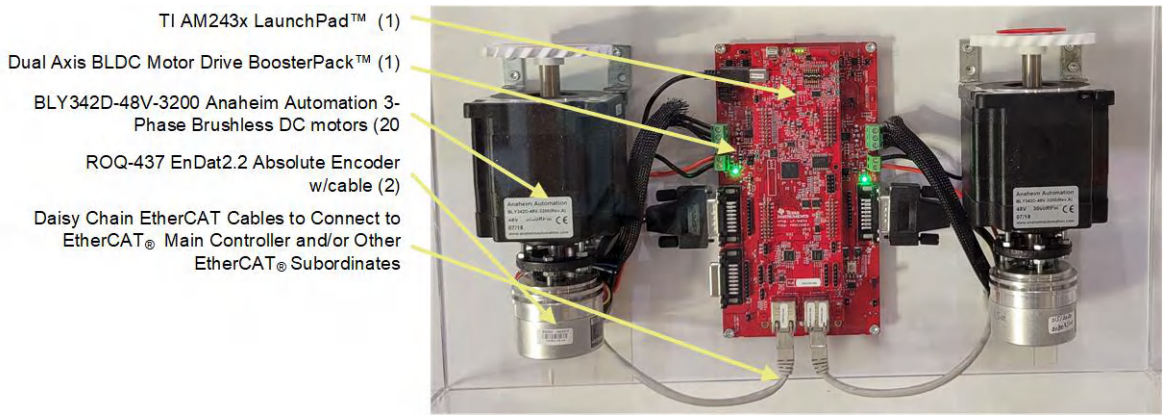


図 4-1. システム ハードウェア構成

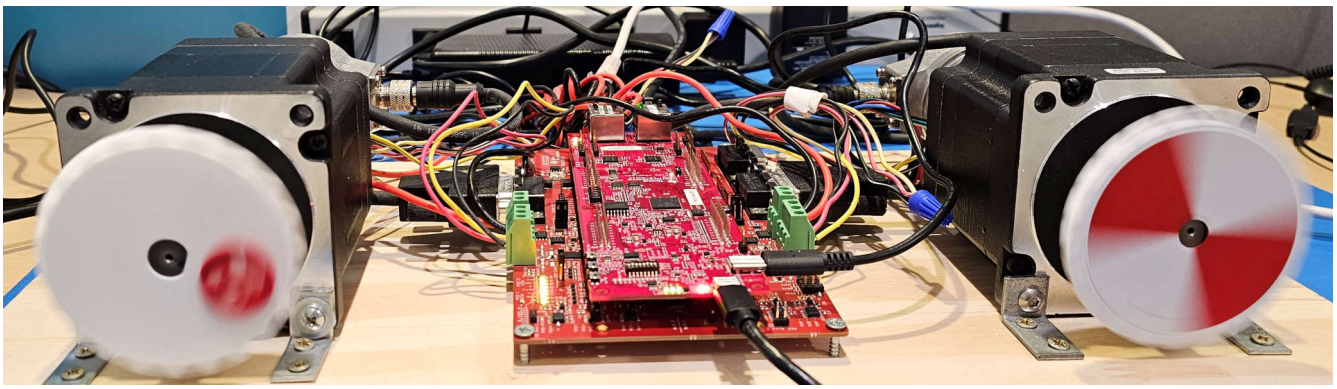


図 4-2. デュアル モーター ドライブのシステム構成

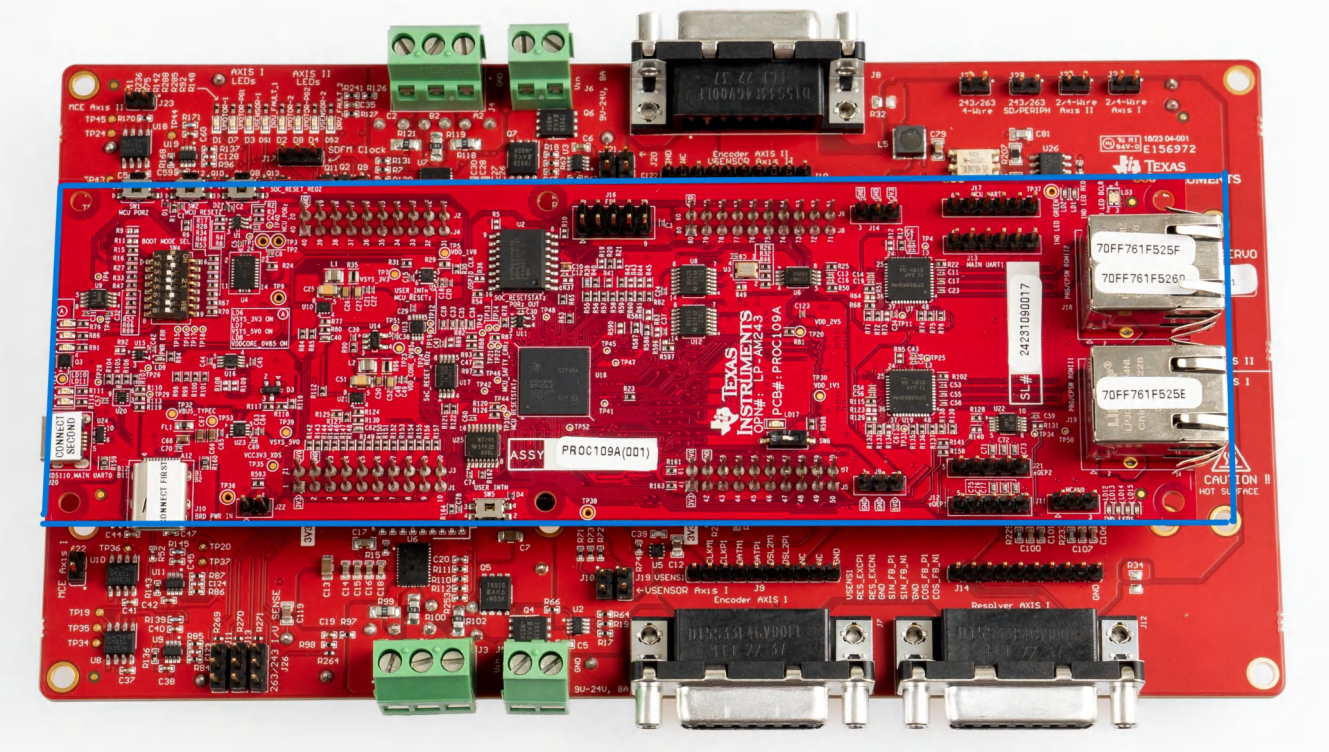


図 4-3. AM243x LP Rev A および BLDC BP E2

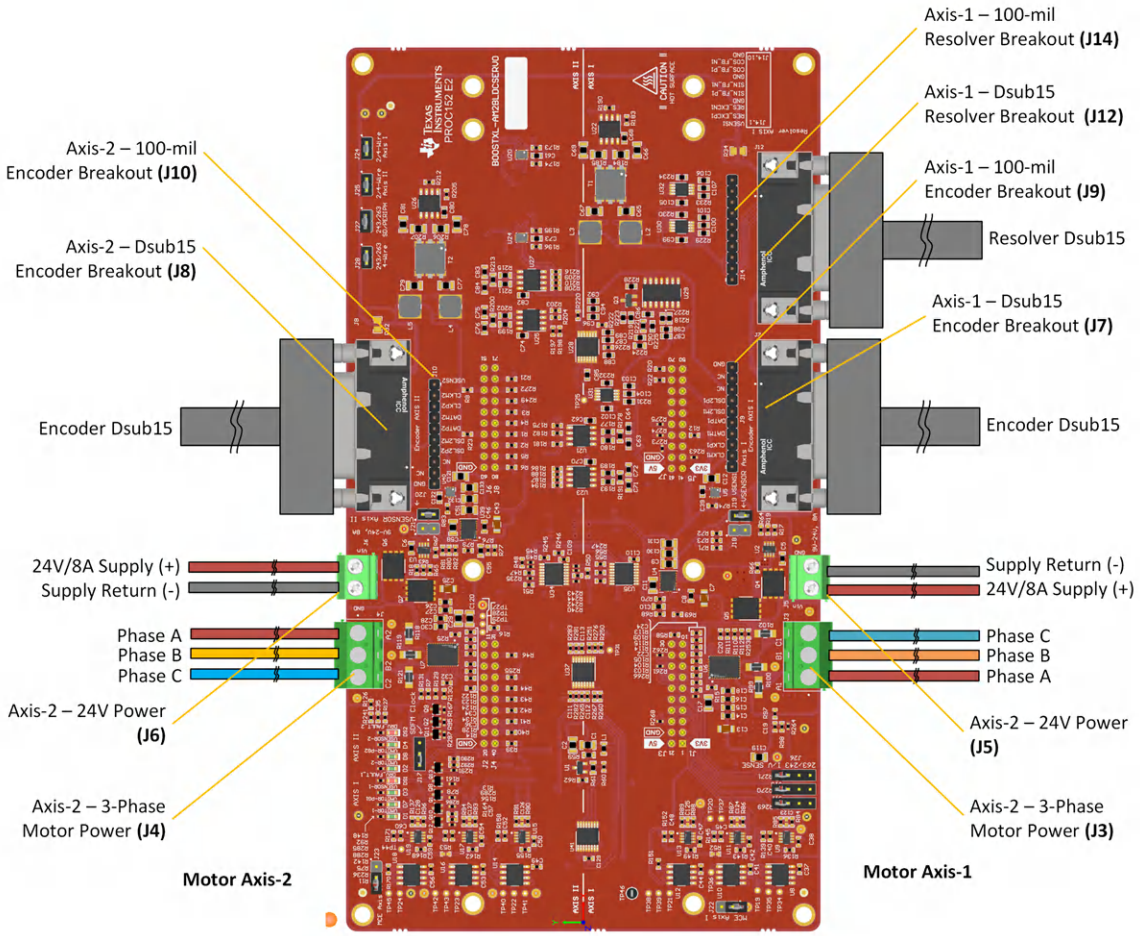
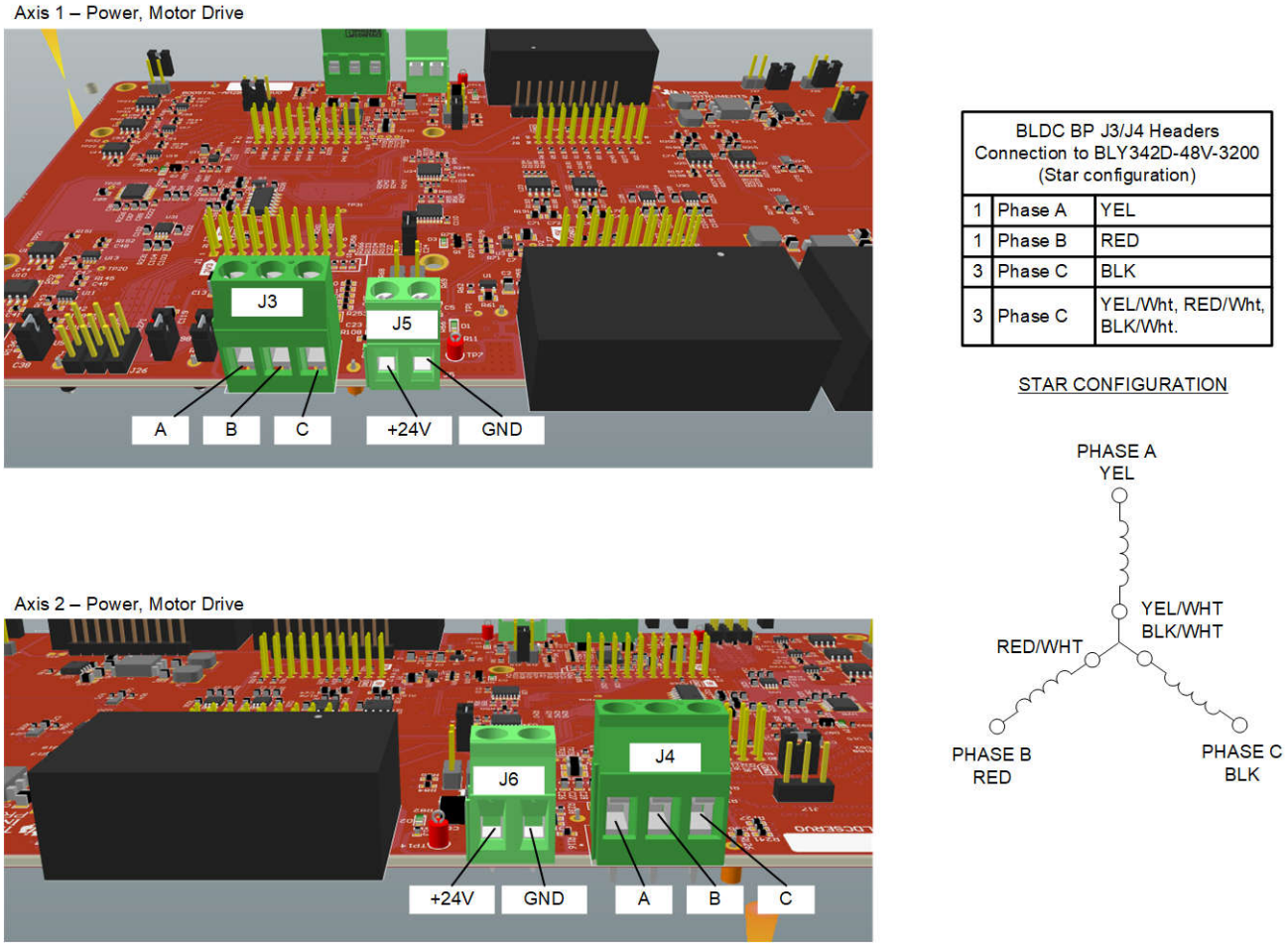


図 4-4. モーター 1 とモーター 2 の BLDC BP E2 コネクタ



BLDC BP J3/J4 Headers Connection to BLY342D-48V-3200 (Star configuration)		
1	Phase A	YEL
1	Phase B	RED
3	Phase C	BLK
3	Phase C	YEL/Wht, RED/Wht, BLK/Wht.

STAR CONFIGURATION

図 4-5. モーター 1 およびモーター 2 のスター構成

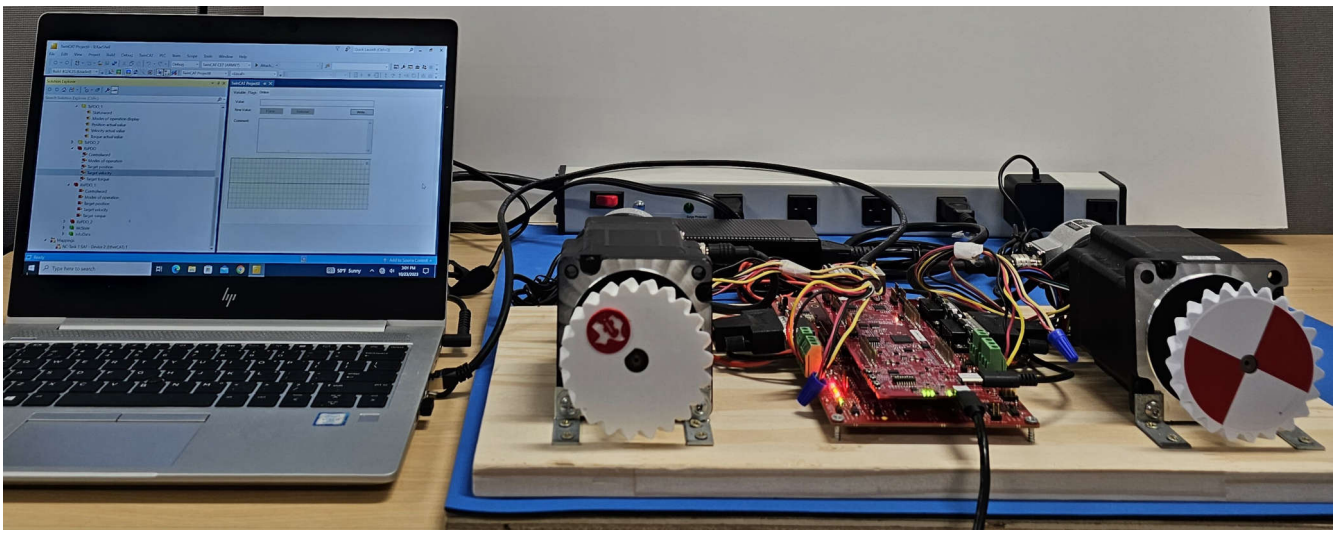
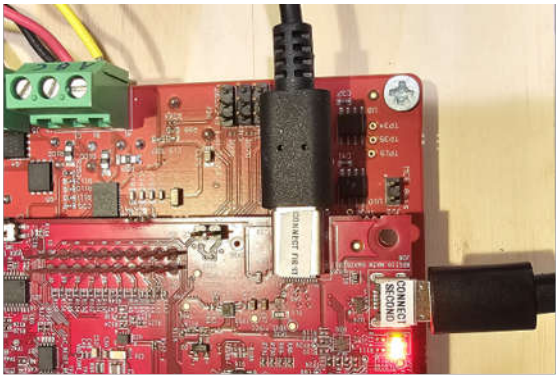


図 4-6. EtherCAT コネクテッド モーター制御のセットアップ



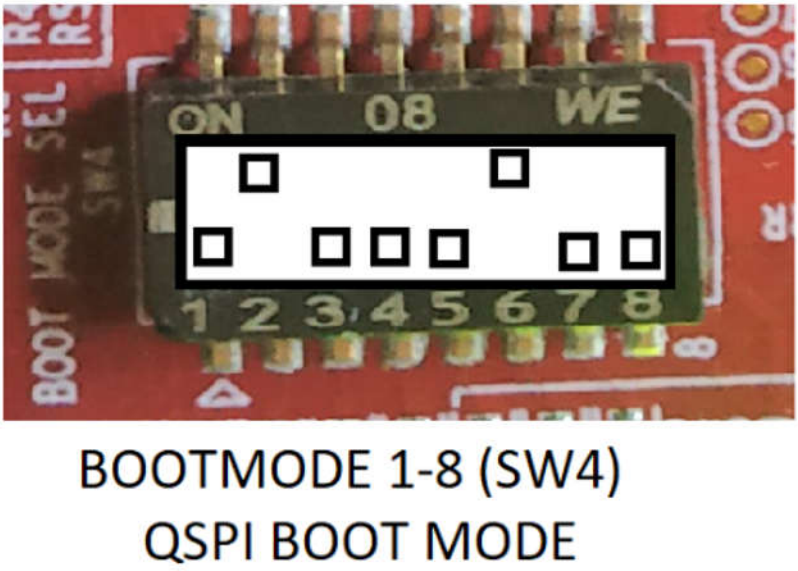


図 4-7. AM243x LP 電源、JTAG、ブート モード

4.2 ソフトウェア要件

リファレンス デザインのソース コードを確認するには、C:\ti\motor_control_sdk_am243x_09_01_00_xx にあるモーター制御 SDK 09.01.00.xx をダウンロードしてインストールします。モーター制御 SDK 09.01.00 では、tidep_01032_dual_motor_drive がリファレンス デザインのソース コードを保持しています。

C:\ti\motor_control_sdk_am243x_09_01_00_xx\examples\tidep_01032_dual_motor_drive\single_chip_servo\am243x-lp\system_freertos_nortos からシステム プロジェクトをインポートし、ビルドします。プロジェクトには、次の 3 つのサブプロジェクトがあります。

1. C:\ti\motor_control_sdk_am243x_09_01_00_xx\examples\tidep_01032_dual_motor_drive\single_chip_servo\am243x-lp\system_freertos_nortos \r5fss0-0_nortos (single_chip_servo_am243x-lp_r5fss0-1_nortos_ti-arm-clang：軸 1 のモーター ドライブ コード)
2. C:\ti\motor_control_sdk_am243x_09_01_00_xx\examples\tidep_01032_dual_motor_drive\r5fss0-1_nortos (single_chip_servo_am243x-lp_r5fss0-1_nortos_ti-arm-clang：軸 2 のモーター ドライブ コード)
3. C:\ti\motor_control_sdk_am243x_09_01_00_xx\examples\tidep_01032_dual_motor_drive\r5fss1-0_freertos (ethercat_slave_cia402_demo_am243x-lp_r5fss1-0_freertos_ti-arm-clang：EtherCAT CiA402 クライアント コード)

4.3 テスト構成

このセクションでは、テスト ソフトウェアをロードして実行する方法を説明します。

システム プロジェクトをインポートしてビルドした後、R5F_0_0、R5F_0_1、R5F_1_0 の実行可能バイナリ ファイルが CCS ワークスペース ディレクトリ（次に示す）に表示されます：
C:\ti\ccs_ws_1250_am243x_mcsdk_09.01.00.01

1. ターゲット構成ファイルを使用して、ターゲット AM243x LP に接続します。
2. モーター制御 1 – R5F_0_0 をロードして実行します。
   - R5F_0_0 が停止します。
   - single_chip_servo_am243x-LP_r5fss0-0_nortos_ti-arm-clang をロードして実行します。
   - モーター 1 は 120RPM で回転を始めることが想定されています。

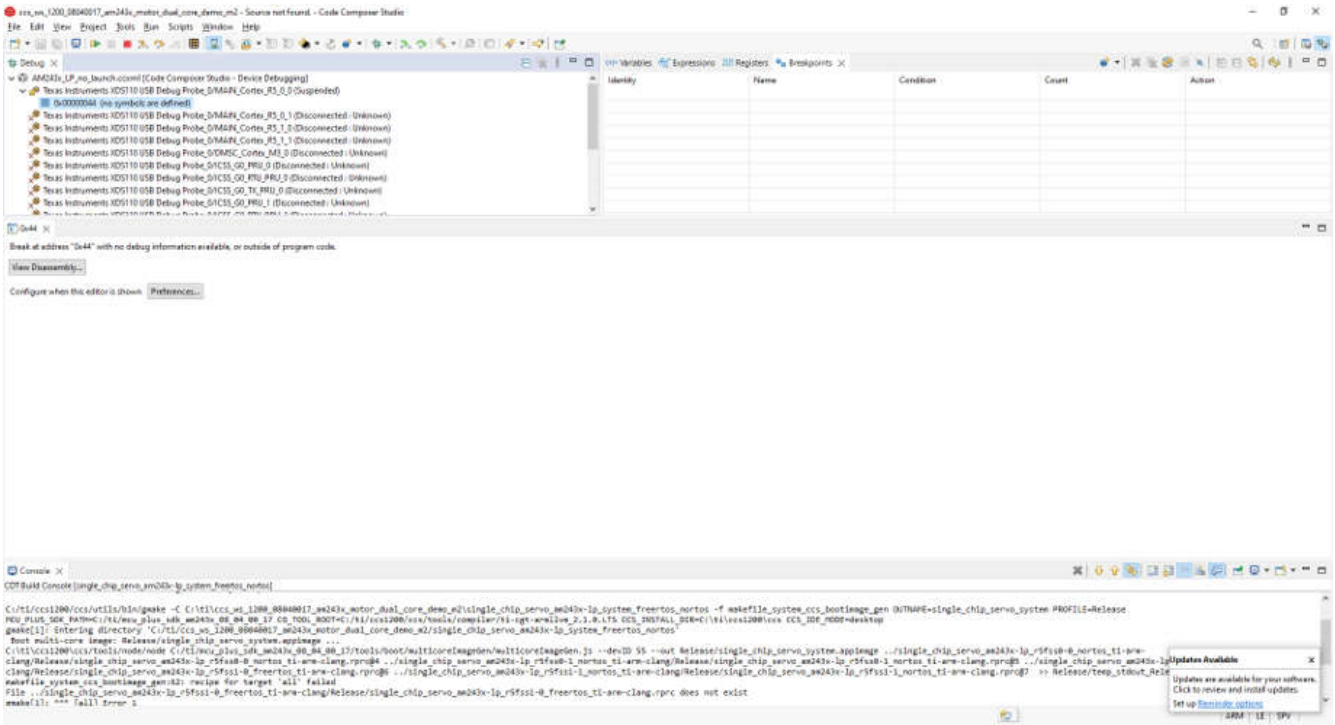


図 4-8. R5F_0_0 に接続

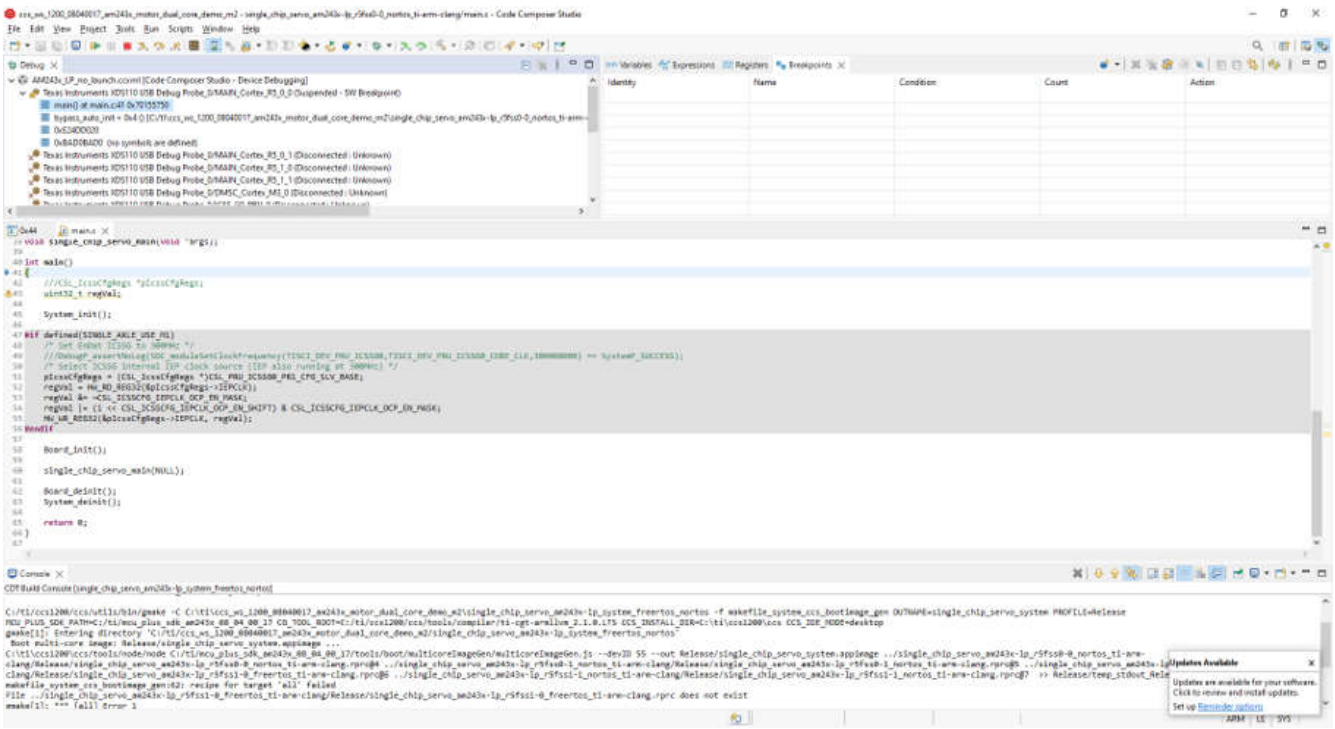


図 4-9. R5F_0_0 のロードと実行

3. モーター制御 2 – R5F_0_1 をロードして実行します。
   - R5F_0_1 が停止します。
   - single_chip_servo_am243x-LP_r5fss0-1_nortos_ti-arm-clang をロードして実行します。
   - モーター 2 は 120RPM で回転を始めることが想定されています。

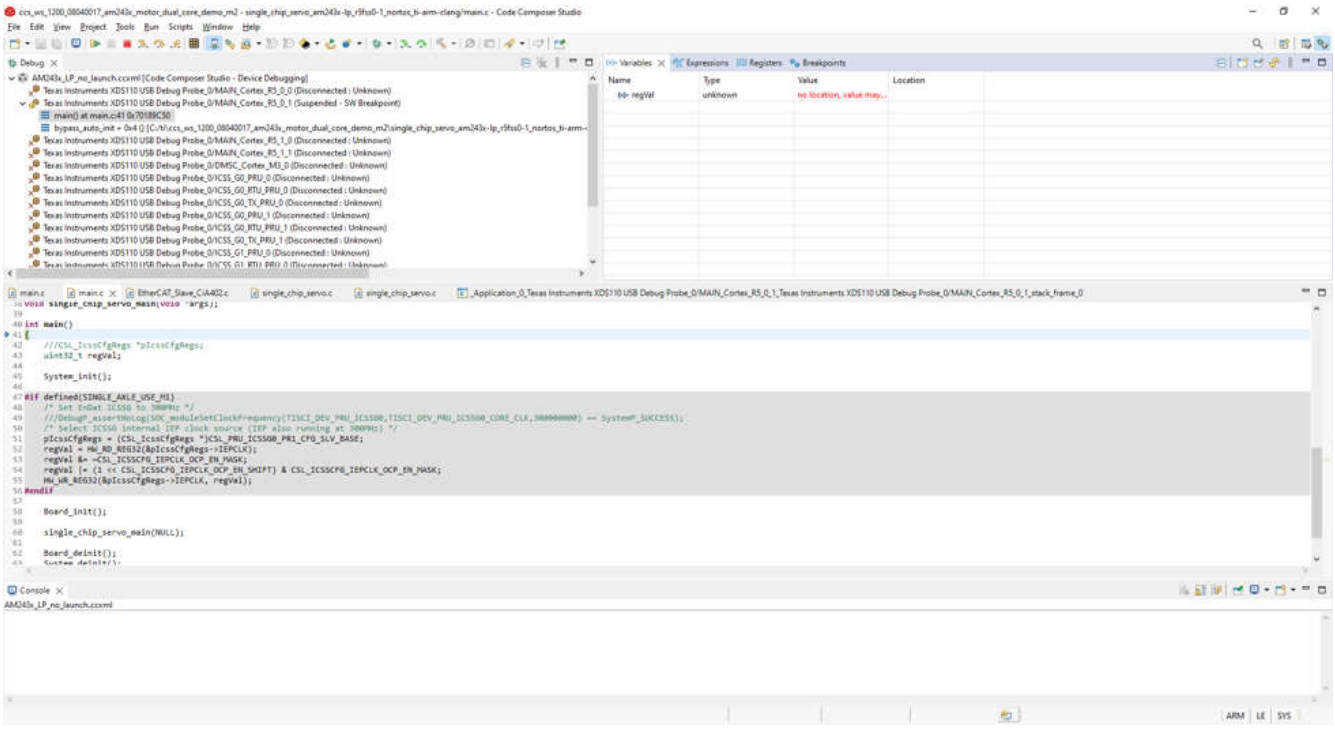


図 4-10. R5F_0_1 のロードと実行

4. EtherCat CiA402 Client – R5F_1_0 をロードして実行します。
   - R5F_1_0 が停止します。

- `ethercat_slave_cia402_demo_am243x-lp_r5fss1-0_freertos_ti-arm-clang` をロードして実行します。
- これで、EtherCat CiA402 クライアントデバイスを TwinCAT (PLC) で検出する準備が整いました。

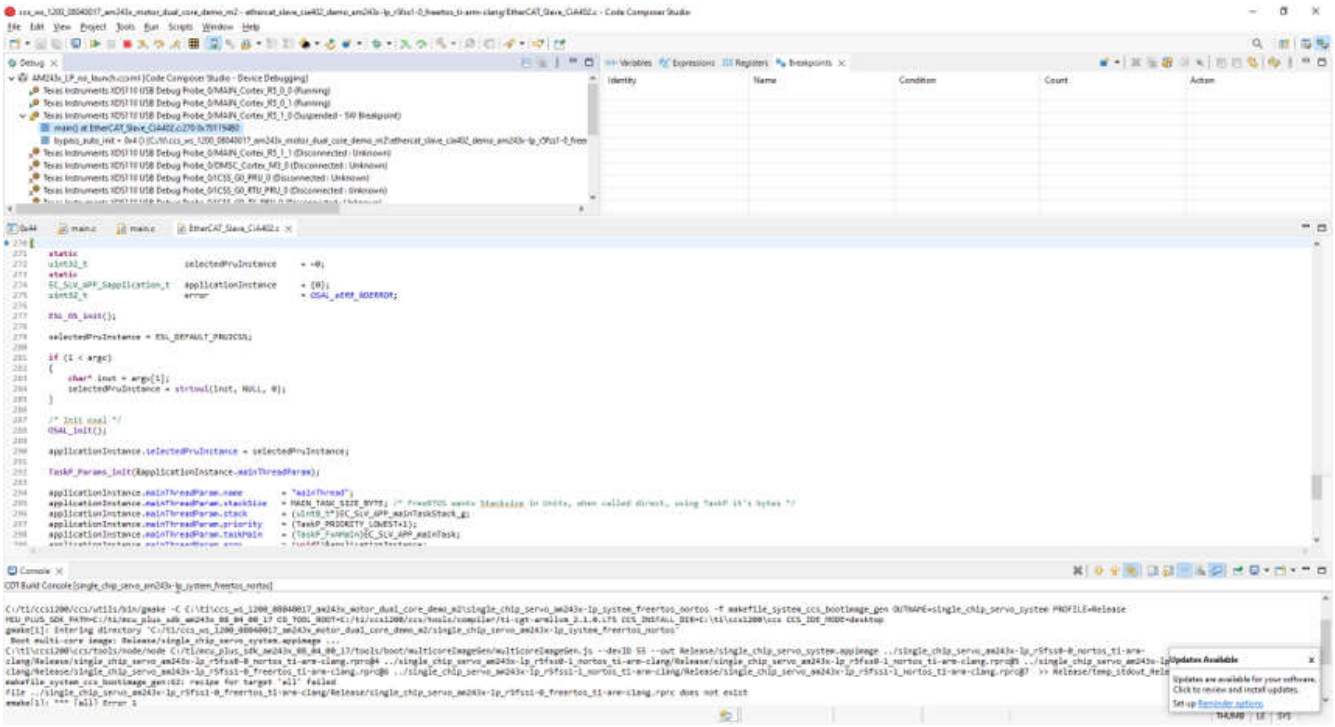


図 4-11. R5F_1_0 のロードと実行

4.4 テスト結果

このリファレンス デザインの評価は次の手順に従います。

1. Windows PC に TwinCAT をダウンロードしてインストールします
2. TwinCAT 自動化ソフトウェアを起動します
3. TwinCAT ソフトウェアの GUI の表示に従って、EtherCAT プロジェクトを作成します:

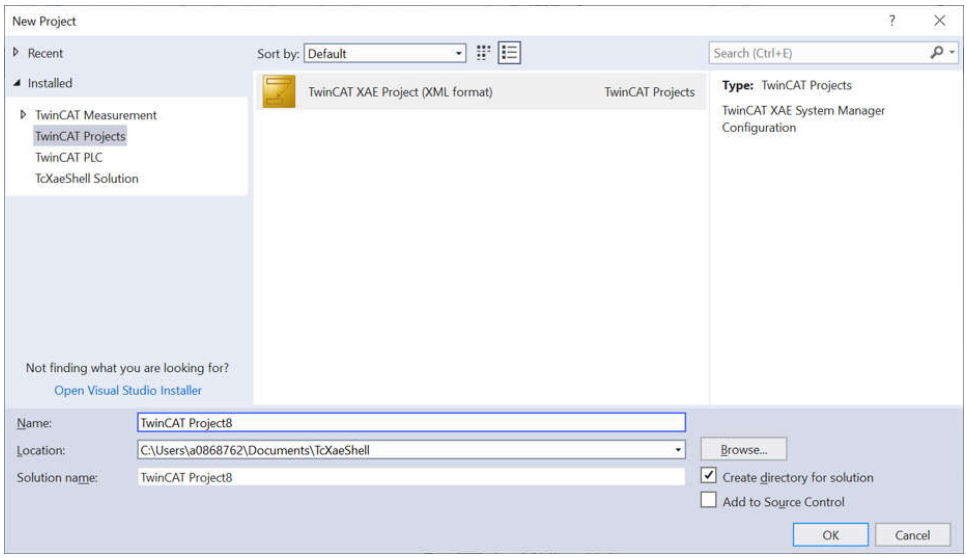


図 4-12. TwinCAT での EtherCAT® プロジェクトの作成

4. EtherCAT CiA402 – (Devices → Scan …) を右クリックして、デバイスをスキャンします。次の図を使用して、プロセスをステップ実行します。

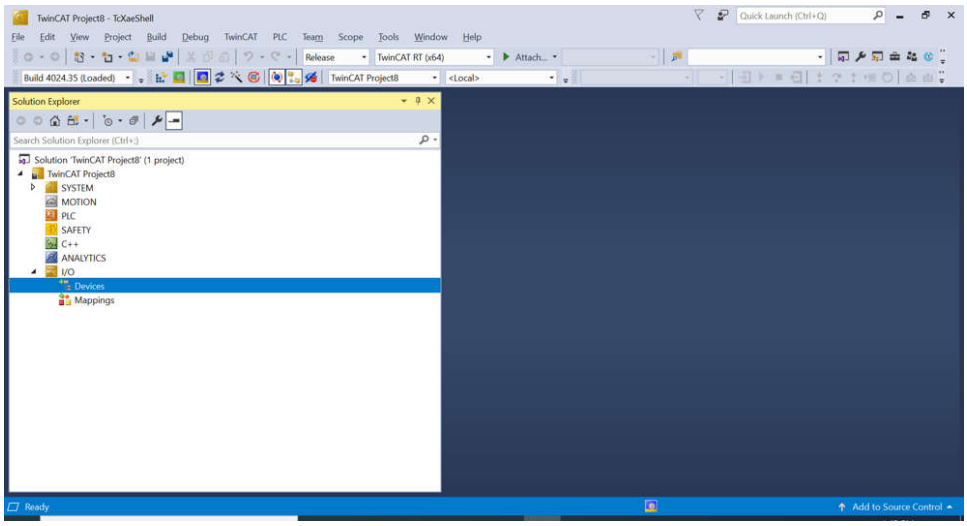


図 4-13. TwinCAT の EtherCAT® デバイスをスキャン

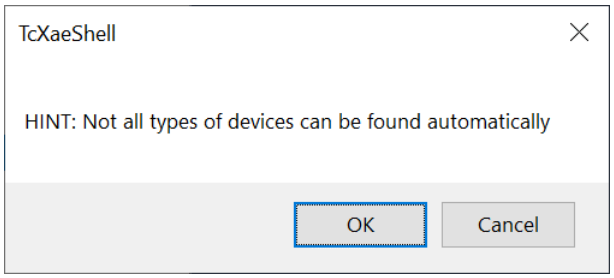


図 4-14. TwinCAT の EtherCAT® デバイスをスキャン (2)

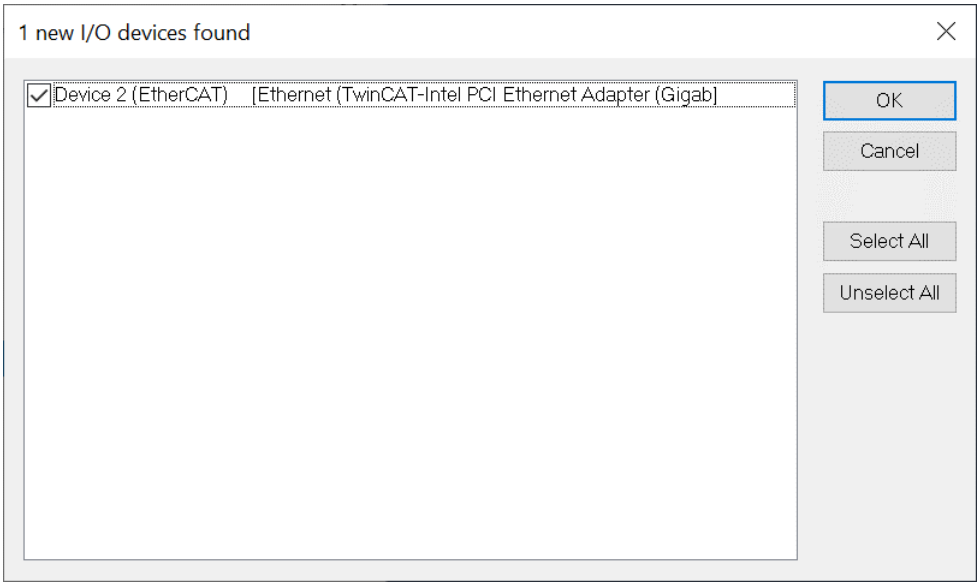


図 4-15. TwinCAT の EtherCAT® デバイスをスキャン (3)

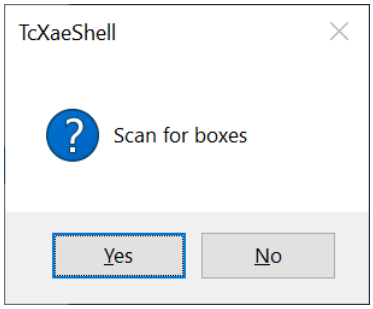


図 4-16. TwinCAT の EtherCAT® デバイスをスキャン (4)

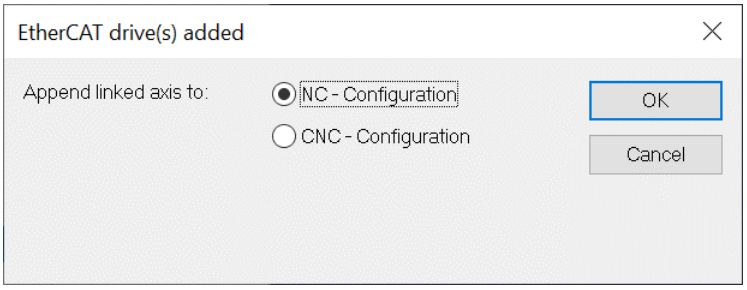


図 4-17. TwinCAT の EtherCAT® デバイスをスキャン (5)

5. EtherCAT CiA402 –デバイス 1 (AM243X.R5F 用 TI EtherCAT ツールキット CiA402) が見つかりました

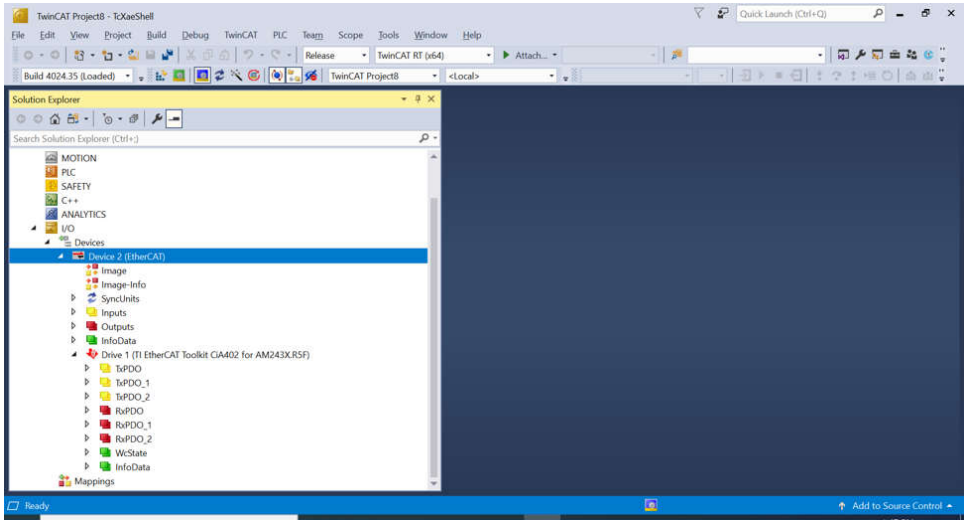


図 4-18. TwinCAT が EtherCAT® デバイスを検出します。

6. EtherCAT CiA402 – RxPDO (モーター 1) の目標速度を 240 (240RPM) に変更

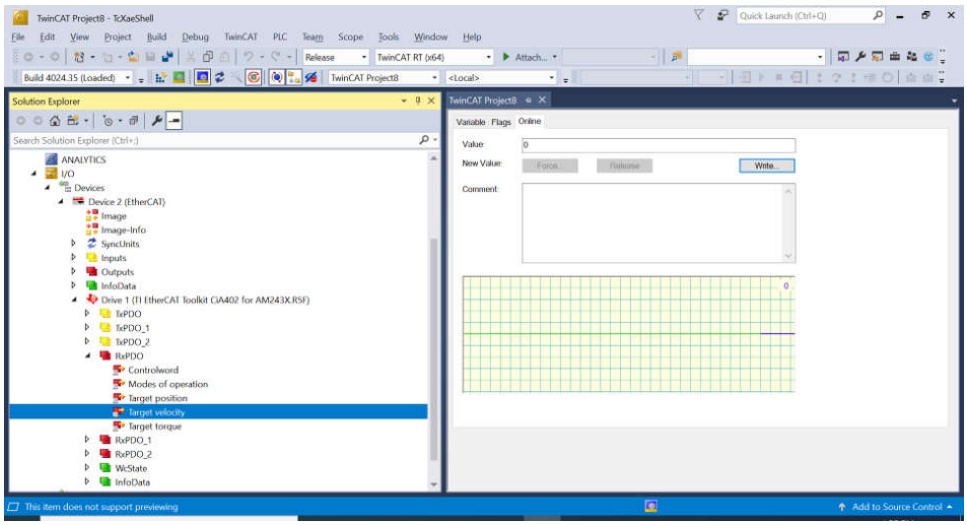


図 4-19. TwinCAT のモーター 1 の目標速度を変更

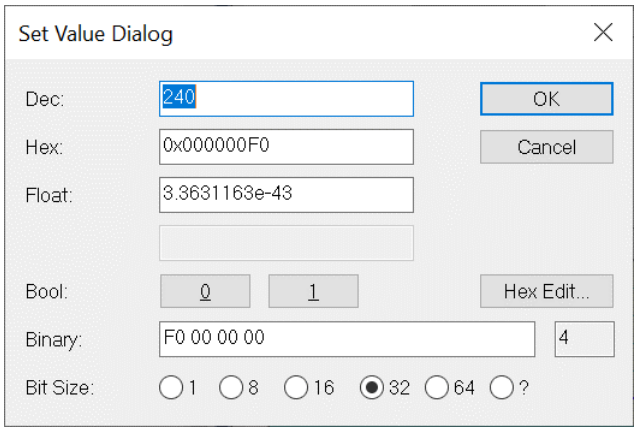


図 4-20. TwinCAT のモーター 1 の目標速度を変更 (2)

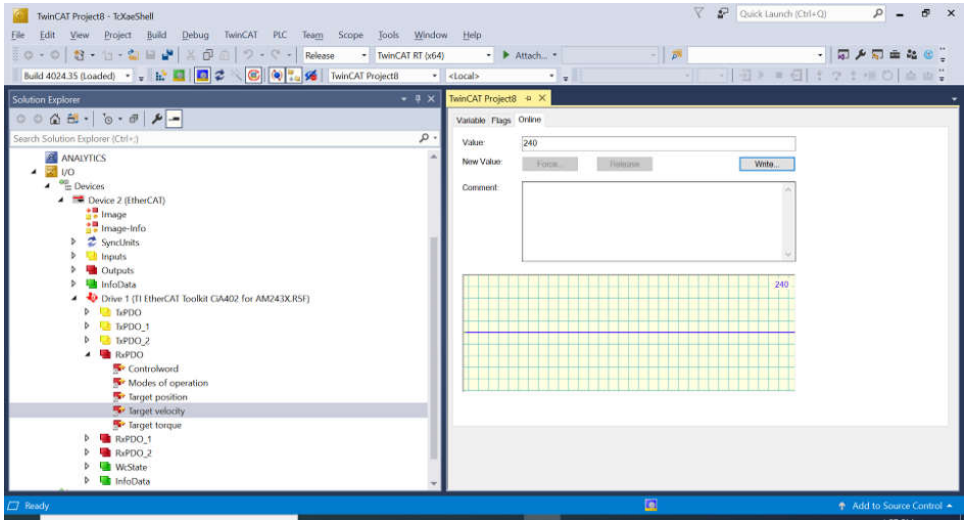


図 4-21. TwinCAT のモーター 1 の目標速度を変更 (3)

7. EtherCAT CiA402 – RxPDO (モーター 1) の動作モードを「9」(周期同期速度モード) に変更

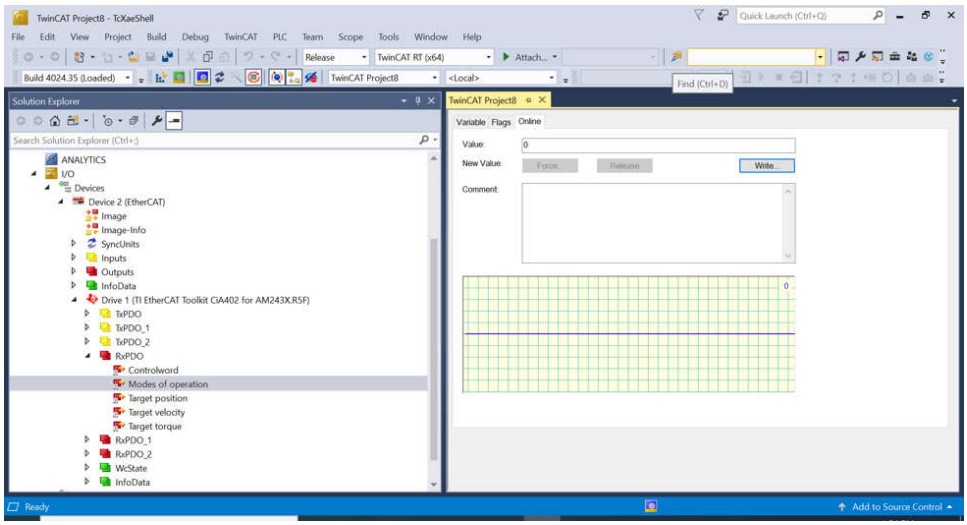


図 4-22. TwinCAT のモーター 1 の動作モードを変更

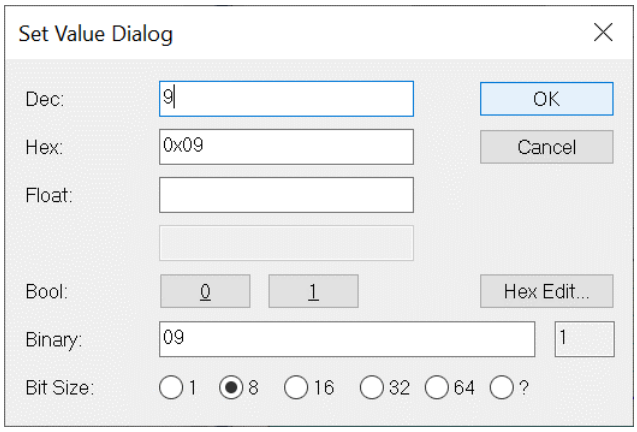


図 4-23. TwinCAT のモーター 1 の動作モードを変更 (2)

8. EtherCAT CiA402 – RxPDO (モーター 1) の制御ワードを「15」に変更 (スイッチオン | 電圧イネーブル | クイック ストップ | 動作イネーブル)

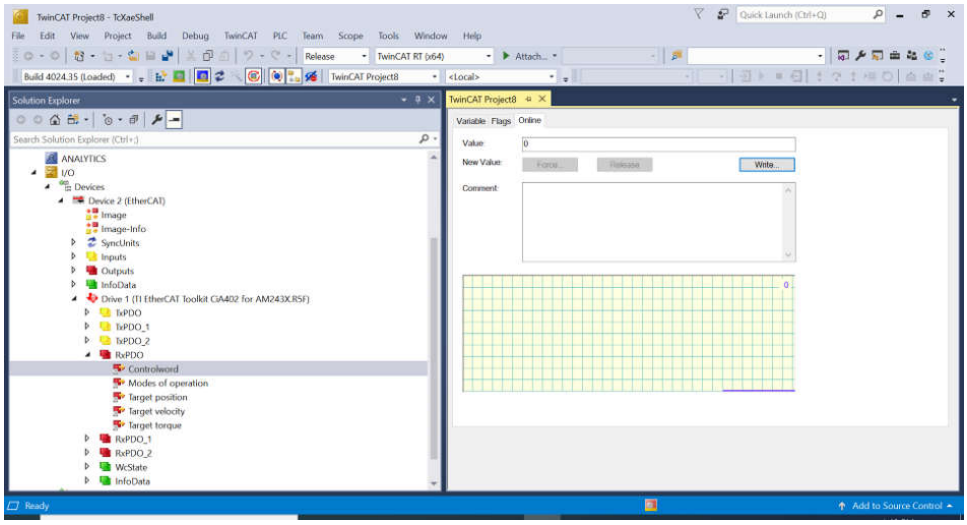


図 4-24. TwinCAT のモーター 1 の制御ワードを変更

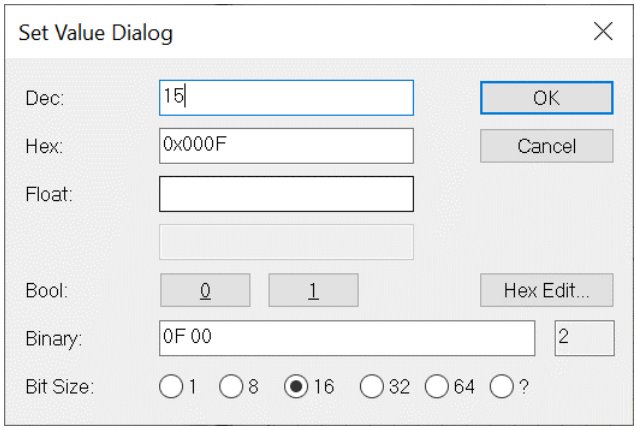


図 4-25. TwinCAT のモーター 1 の制御ワードを変更 (2)

9. この設定を変更すると、モーター 1 の速度が 120RPM から 240RPM に変更されます。
10. EtherCAT CiA402 – TxPDO (モーター 1) をチェックし、速度の実際の値が 240 (240RPM) であることを確認します

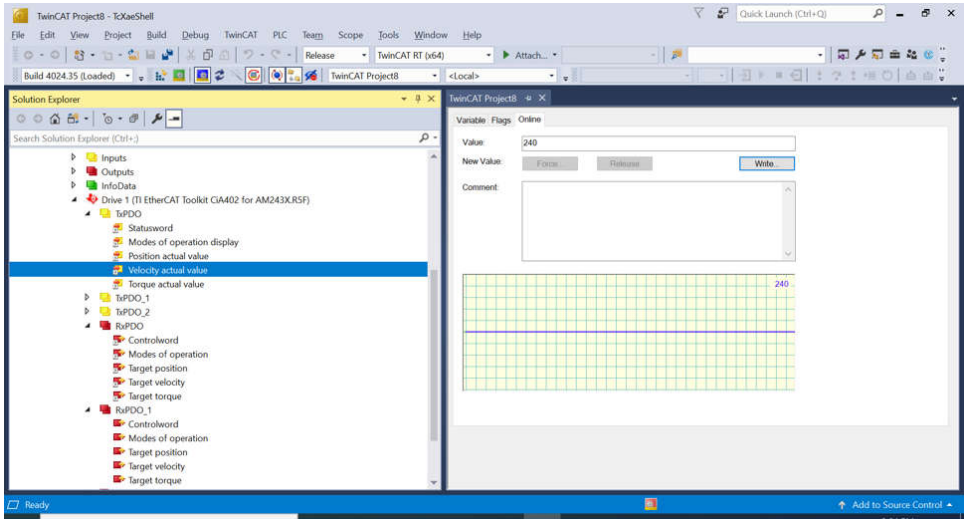


図 4-26. TwinCAT のモーター 1 の実際の速度を確認します

11. EtherCAT CiA402 – RxPDO_1 (モーター 2) の目標速度を 180 (180RPM) に変更

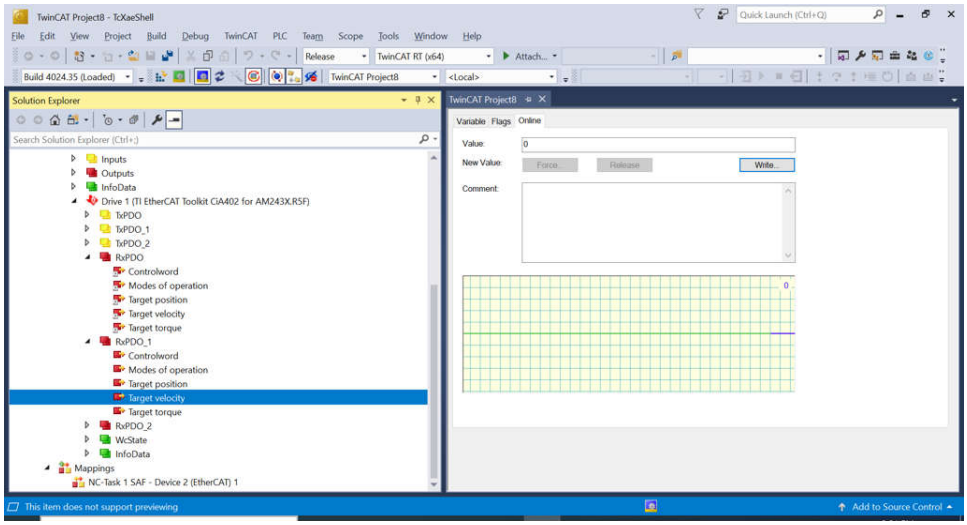


図 4-27. TwinCAT のモーター 2 の目標速度の変更

12. EtherCAT CiA402 – RxPDO_1 (モーター 2) の動作モードを「9」(周期同期速度モード) に変更

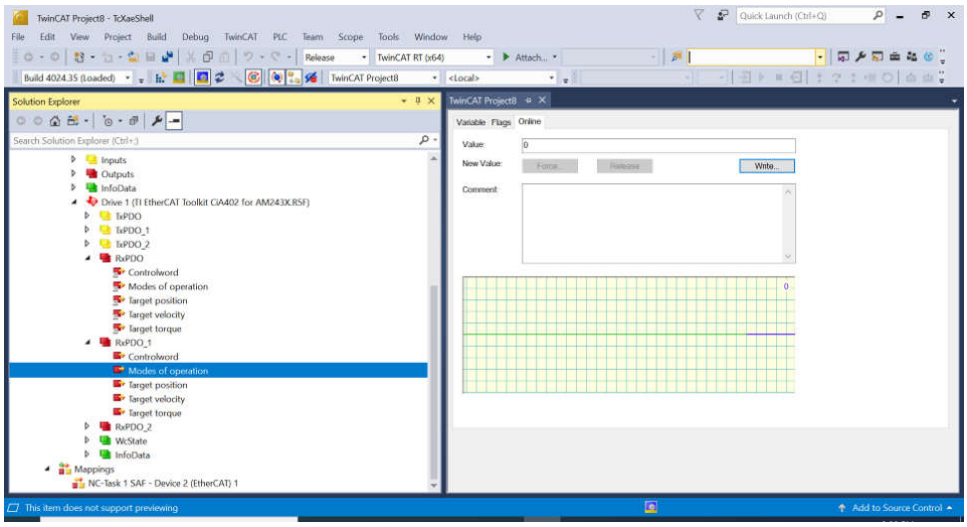


図 4-28. TwinCAT のモーター 2 の動作モードの変更

13. EtherCAT CiA402 – RxPDO_1 (モーター 2) の制御ワードを「15」に変更 (スイッチオン | 電圧イネーブル | クイックストップ | 動作イネーブル)

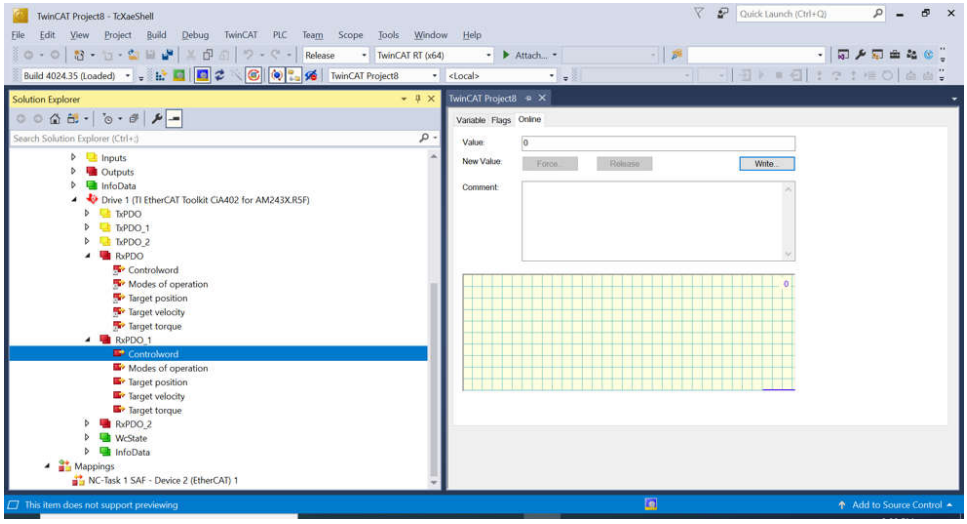


図 4-29. TwinCAT のモーター 2 の制御ワードを変更

14. 前の変更後、モーター 2 の目標速度は 180RPM です。

15. EtherCAT CiA402 – TxPDO1 (モーター 2) をチェックし、速度の実際の値が 180 (180RPM) であることを確認します

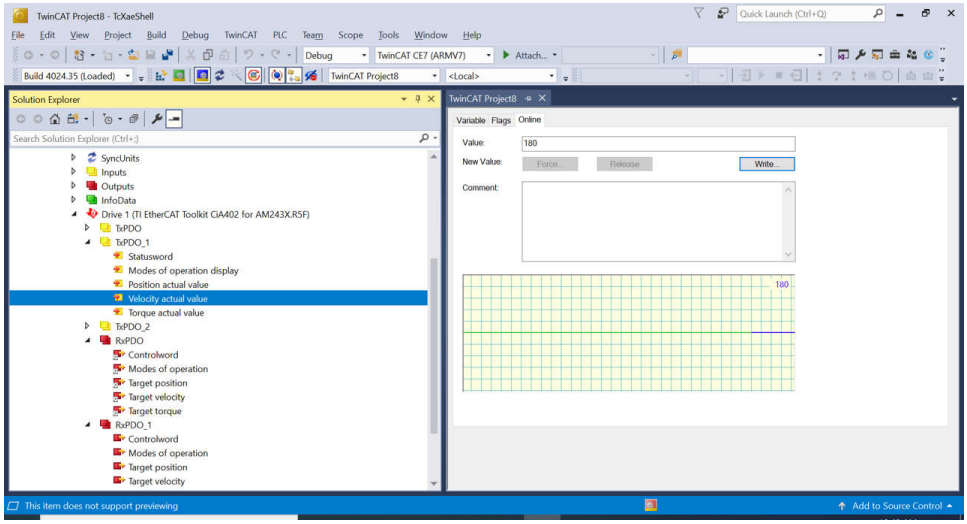


図 4-30. TwinCAT のモーター 2 の実際の速度を確認します

5 設計とドキュメントのサポート

5.1 デザイン ファイル

5.1.1 回路図

このリファレンス デザインに関連する回路図が 2 つあります。

BLDC BoosterPack の回路図をダウンロードするには、BP-AM2BLDCSERVO デザイン パッケージのデザイン ファイルを参照してください。

AM243x LaunchPad の回路図をダウンロードするには、LP-AM243 デザイン パッケージのデザイン ファイルを参照してください。

5.1.2 BOM (部品表)

BLDC BP の部品表 (BOM) をダウンロードするには、BP-AM2BLDCSERVO デザイン パッケージ ページにあるデザイン ファイルを参照してください。

AM243x LP の部品表 (BOM) をダウンロードするには、LP-AM243 デザイン パッケージのデザイン ファイルを参照してください。

5.2 ツールとソフトウェア

ツール

CCSTUDIO	Code Composer Studio™ 統合開発環境 (IDE):Windows または Linux 向けの CCS 12.5.0 バージョンをダウンロード
ARM-CGT-CLANG	Arm® コード生成ツール - コンパイラ:Windows または Linux 向け TI ARM CLANG 3.2.0.LTS をダウンロード
SYSCONFIG	SysConfig のスタンドアロン デスクトップ バージョン:Windows または Linux 向けの SysConfig 1.18.0 をダウンロード

ソフトウェア

AM243x モーター制御 SDK	モーター制御 SDK Windows インストーラ
AM243x 産業用通信 SDK	産業用通信 SDK Windows インストーラ
AM243x MCU+ SDK	MCU PLUS SDK Windows インストーラ

5.3 ドキュメントのサポート

1. テキサス・インスツルメンツ、『AM64x/AM243x プロセッサ シリコン テクニカル リファレンス・マニュアル』
2. テキサス・インスツルメンツ、『AM2x BLDC サーボ・モーター ブースタパック (BPAM2BLDCSERVO) EVM ユーザー ガイド』

5.4 サポート・リソース

テキサス・インスツルメンツ E2E™ サポート・フォーラムは、エンジニアが検証済みの回答と設計に関するヒントをエキスパートから迅速かつ直接得ることができる場所です。既存の回答を検索したり、独自の質問をしたりすることで、設計で必要な支援を迅速に得ることができます。

リンクされているコンテンツは、各寄稿者により「現状のまま」提供されるものです。これらはテキサス・インスツルメンツの仕様を構成するものではなく、必ずしもテキサス・インスツルメンツの見解を反映したものではありません。テキサス・インスツルメンツの使用条件を参照してください。

5.5 商標

6 著者について

MING WEI (MGTS) は、Sitara MCU のシニア ソフトウェア エンジニアであり、Sitara MPU/MCU 向けのプロセッサ SDK RTOS/MCU+SDK/ モーター制御 SDK と、SoC デバイスの DSP ファミリの開発とサポートに携わっています。モーター制御、リアルタイム システム、信号処理、コード最適化に関する幅広い経験と知識をこの職務に活かしています。Ming は、西安交通大学とノース テキサス大学でそれぞれ理学士号、理学修士号、およびコンピューター科学の博士号を取得しています。

重要なお知らせと免責事項

テキサス・インスツルメンツは、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、テキサス・インスツルメンツ製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した テキサス・インスツルメンツ製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている テキサス・インスツルメンツ製品を使用するアプリケーションの開発の目的でのみ、テキサス・インスツルメンツはその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。テキサス・インスツルメンツや第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、テキサス・インスツルメンツおよびその代理人を完全に補償するものとし、テキサス・インスツルメンツは一切の責任を拒否します。

テキサス・インスツルメンツの製品は、テキサス・インスツルメンツの販売条件、または ti.com やかかる テキサス・インスツルメンツ製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。テキサス・インスツルメンツがこれらのリソースを提供することは、適用されるテキサス・インスツルメンツの保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、テキサス・インスツルメンツはそれらに異議を唱え、拒否します。

郵送先住所:Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

重要なお知らせと免責事項

TI は、技術データと信頼性データ (データシートを含みます)、設計リソース (リファレンス· デザインを含みます)、アプリケーションや設計に関する各種アドバイス、Web ツール、安全性情報、その他のリソースを、欠陥が存在する可能性のある「現状のまま」提供しており、商品性および特定目的に対する適合性の黙示保証、第三者の知的財産権の非侵害保証を含むいかなる保証も、明示的または黙示的にかかわらず拒否します。

これらのリソースは、TI 製品を使用する設計の経験を積んだ開発者への提供を意図したものです。(1) お客様のアプリケーションに適した TI 製品の選定、(2) お客様のアプリケーションの設計、検証、試験、(3) お客様のアプリケーションに該当する各種規格や、その他のあらゆる安全性、セキュリティ、規制、または他の要件への確実な適合に関する責任を、お客様のみが単独で負うものとします。

上記の各種リソースは、予告なく変更される可能性があります。これらのリソースは、リソースで説明されている TI 製品を使用するアプリケーションの開発の目的でのみ、TI はその使用をお客様に許諾します。これらのリソースに関して、他の目的で複製することや掲載することは禁止されています。TI や第三者の知的財産権のライセンスが付与されている訳ではありません。お客様は、これらのリソースを自身で使用した結果発生するあらゆる申し立て、損害、費用、損失、責任について、TI およびその代理人を完全に補償するものとし、TI は一切の責任を拒否します。

TI の製品は、TI の販売条件、または ti.com やかかる TI 製品の関連資料などのいずれかを通じて提供する適用可能な条項の下で提供されています。TI がこれらのリソースを提供することは、適用される TI の保証または他の保証の放棄の拡大や変更を意味するものではありません。

お客様がいかなる追加条項または代替条項を提案した場合でも、TI はそれらに異議を唱え、拒否します。

郵送先住所：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265