エンコーダ利用ベクトル制御

RAJ306000 実装編

要旨

本アプリケーションノートは、RAJ306000 を使用して、エンコーダ付き永久磁石同期モータをベクトル制御で駆動させるサンプルプログラム、及び、開発支援ツール「In Circuit Scope」のライブラリを使用する方法について説明することを目的とする。

サンプルプログラムは、あくまで参考用途であり、弊社がこの動作を保証するものではありません。サンプルプログラムを使用する場合、適切な環境で十分な評価を行ったうえで使用して下さい。

動作確認デバイス

サンプルプログラムの動作確認は、下記のデバイスで行っている。

・RAJ306000

対象サンプルプログラム

本アプリケーションノートが対象とするサンプルプログラムを下記に示す。

・RAJ306000_ENCD_FOC_CLOSED_CSP_CA_V102 (IDE: CS+ for CA, CX)

・RAJ306000_ENCD_FOC_CLOSED_CSP_CC_V102 (IDE: CS+ for CC)

・RAJ306000_ENCD_FOC_CLOSED_E2S_CC_V102 (IDE: e^2 studio)

RAJ306000 向けエンコーダ利用ベクトル制御サンプルプログラム

(相補 PWM 方式)

参考資料

・RL78/G1F ユーザーズマニュアル ハードウェア編 (R01UH0516JJ0110)

・RAJ306000 シリーズ ユーザーズマニュアル ハードウェア編 (R18UZ0066JJ0100)

・永久磁石同期モータのエンコーダベクトル制御 アルゴリズム編 (R01AN3789JJ0101)

・In Circuit Scope 取扱説明書、「In Circuit Scope」を使用するための CS+の設定方法

(下記のサイトからダウンロードして下さい)

ダウンロードサイト: http://www.desktoplab.co.jp/download.html

目次

1. 概説

本アプリケーションノートでは、RAJ306000 を使用したエンコーダ付き永久磁石同期モータのベクトル制御サンプルプログラムを実装する方法、及び、開発支援ツール「In Circuit Scope」注1のライブラリを使用する方法について説明する。

注:

1. 開発支援ツール「In Circuit Scope」(以降、ICS)は、株式会社デスクトップラボの製品です。
   株式会社デスクトップラボ: http://www.desktoplab.co.jp/

1.1 開発環境

本アプリケーションノートが対象とするサンプルプログラムの開発環境を Table 1-1, Table 1-2 に示す。

Table 1-1 ソフトウェア開発環境

Integrated Development Environment	CS+ for CA, CX V3.02.00 [15 Mar 2016]
Compiler	CA78K0R V1.72

Integrated Development Environment	CS+ for CC V6.01.00 [01 Dec 2017]
Compiler	CC-RL V1.06.00

Integrated Development Environment	e^2 studio Version: 5.4.0.015
Compiler (Toolchain)	CC-RL V1.06.00

Table 1-2 ハードウェア開発環境

On-chip Debugging Emulator	RENESAS E1 Emulator (R0E000010KCE00)
Operation Checking Device	RAJ306000 注 2
RAJ306000 Series Evaluation Board	RTK0EML2A0D00010BJ

注:

2. MCU(RL78/G1F)とプリドライバを内蔵した SIP 製品である RAJ306000 の構成を Figure 1-1 に示す。

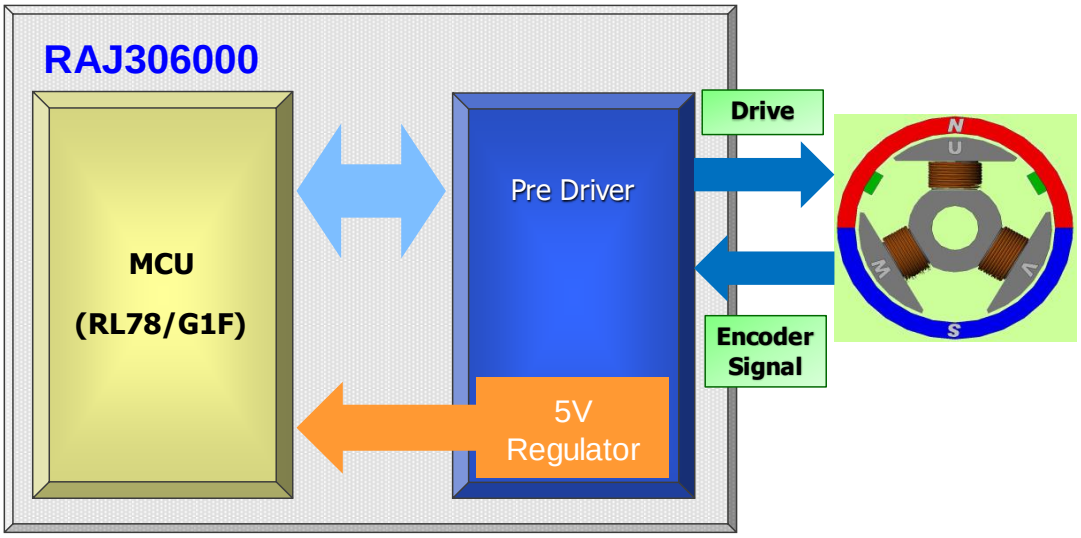

Figure 1-1 RAJ306000

2. システム概要

本システムの概要として、RAJ306000 のシステム構成を Figure 2-1 に示す。

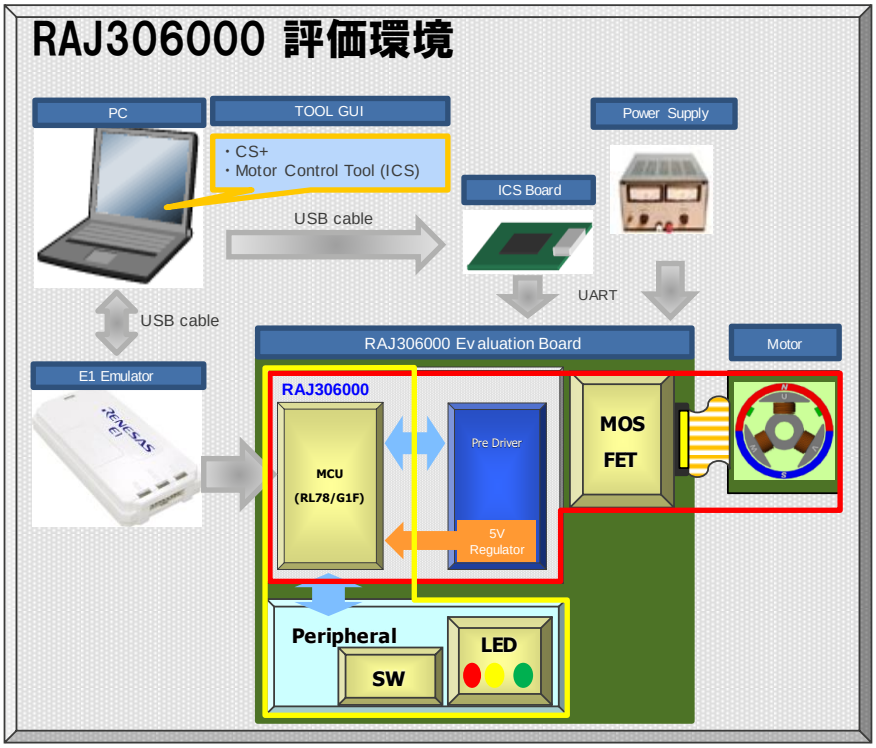

Figure 2-1 システム構成

2.1 ハードウェア構成

ハードウェア構成として、RL78/G1F とプリドライバの接続を Figure 2-2(Figure 2-1 赤線部)に、RL78/G1F と Peripheral との接続を Figure 2-3(Figure 2-1 黄線部)に示す。

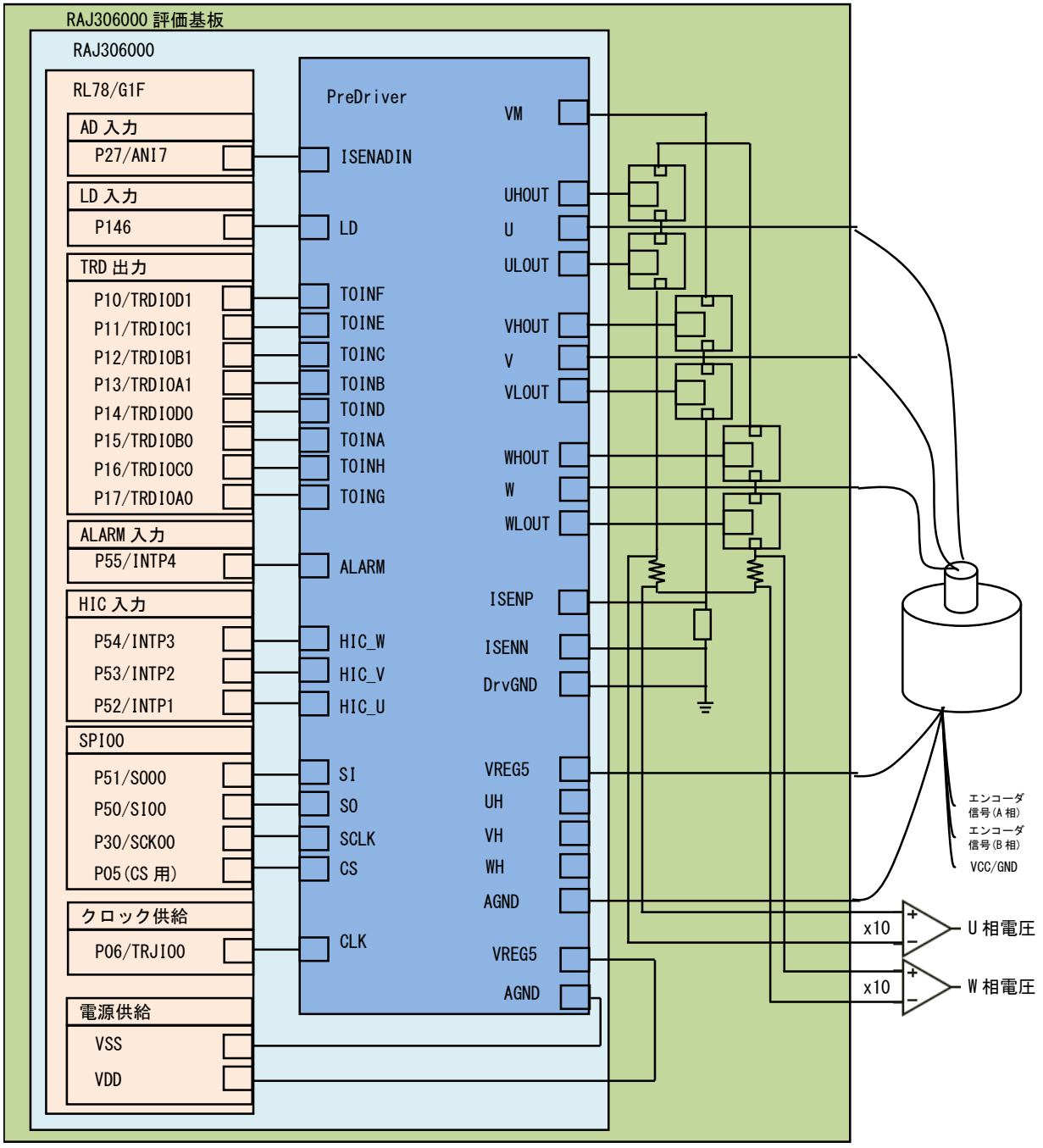

Figure 2-2 ハードウェア構成(RL78/G1F, プリドライバ)

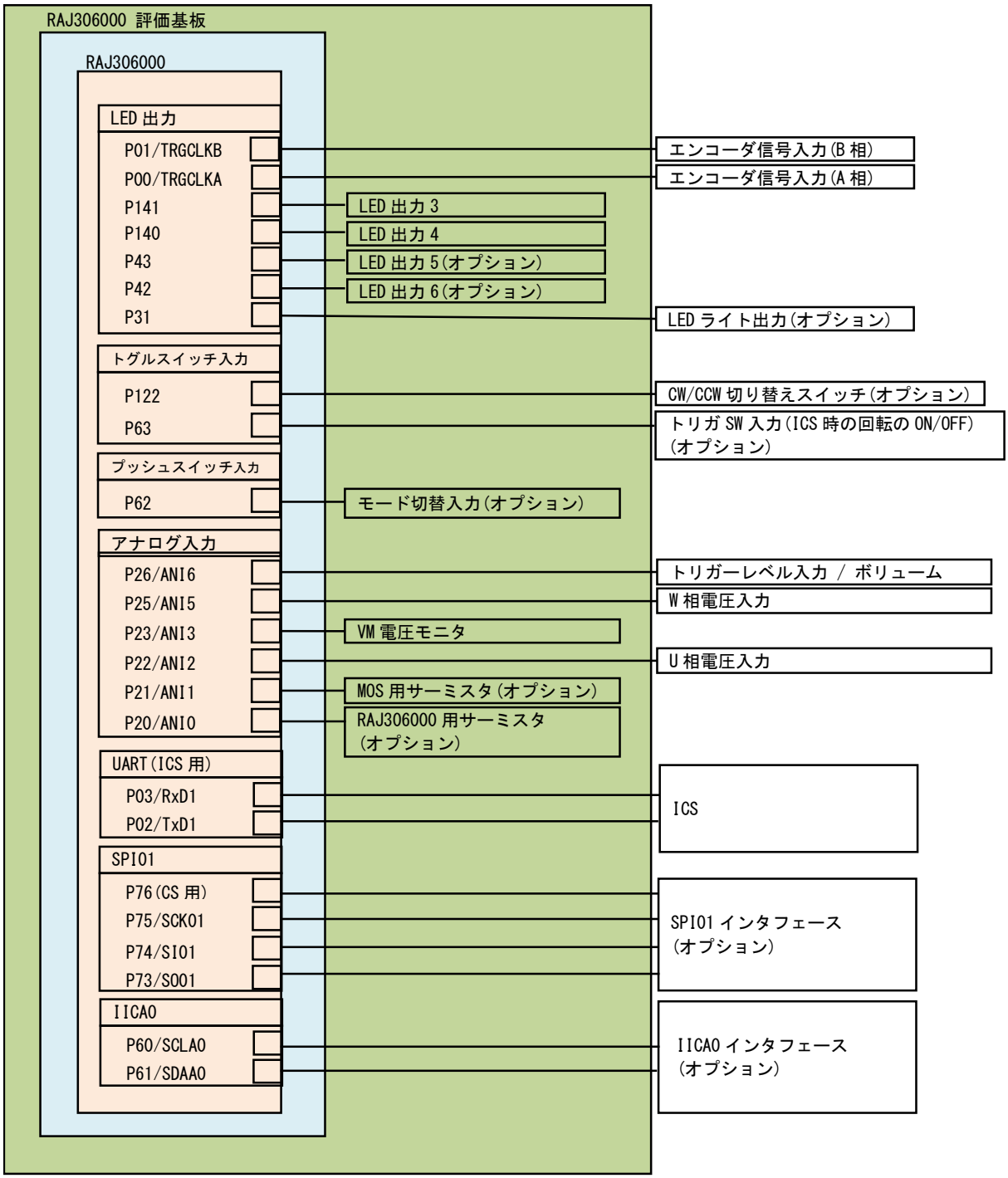

Figure 2-3 ハードウェア構成(RL78/G1F, Peripheral)

2.2 ハードウェア仕様

2.2.1 ユーザインタフェース

本システムのユーザインタフェース一覧を Table 2-1 に示す。

Table 2-1 ユーザインタフェース

項目	インタフェース部品	機能
位置	トリガレベル入力/ボリューム(VR1) or ICS	位置指令値入力(アナログ値)
回転方向	CW/CCW 切り替えスイッチ(SW1)	回転方向指令値入力(CW Only)
回転速度	トリガレベル入力/ボリューム(VR1) or ICS	回転速度指令値入力(アナログ値)
START/STOP	トリガレベル入力/ボリューム(VR1) or ICS	モータサーボ開始/停止指令
赤色 LED	LED 出力 3	・250[rpm] 以下: 消灯 ・250[rpm] 超過: 点灯
	LED 出力 4	・500[rpm] 以下: 消灯 ・500[rpm] 超過: 点灯
	LED 出力 5	・通常動作時: 消灯 ・エラー検出時: 点灯
	LED 出力 6	・通常動作時: 消灯 ・エラー検出時: 点灯

本システムの RL78/G1F 端子インタフェース一覧を Table 2-2 に示す。

Table 2-2 端子インタフェース(RL78/G1F)

端子名	機能
P27/ANI7	プリドライバ電圧測定(入力)
P10/TRDIOD1	PWM 出力(W_n)
P11/TRDIOC1	PWM 出力(V_n)
P12/TRDIOB1	PWM 出力(W_p)
P13/TRDIOA1	PWM 出力(V_p)
P14/TRDIOD0	PWM 出力(U_n)
P15/TRDIOB0	PWM 出力(U_p)
P55/INTP4	ALARM 信号入力
P51/SO00	プリドライバ制御用データ出力
P50/SI00	プリドライバ制御用データ入力
P30/SCK00	プリドライバ制御用クロック出力
P05(CS)	プリドライバ制御用チップセレクト
P06/TRJIO0	プリドライバ用システムクロック出力
VSS	グランド電位
VDD	正電源
P146, P16/TRDIOC0, P17/TRDIOA0 P54/INTP3, P53/INTP2, P52/INTP1	未使用端子
P01/TRGCLKB	エンコーダ信号入力(B 相)
P00/TRGCLKA	エンコーダ信号入力(A 相)
P141	LED 出力 3 点灯/消灯制御
P140	LED 出力 4 点灯/消灯制御
P43	LED 出力 5 点灯/消灯制御
P42	LED 出力 6 点灯/消灯制御
P122	回転方向指令値入力(CW Only)
P26/ANI6	位置指令値入力(アナログ値)
	回転速度指令値入力(アナログ値)
	モータサーボ開始/停止指令
P25/ANI5	W 相シャント抵抗電圧測定(入力)
P23/ANI3	VM 電圧測定(入力)
P22/ANI2	U 相シャント抵抗電圧測定(入力)
P03/RxD1	ICS 用 UART 入力
P02/TxD1	ICS 用 UART 出力
P31, P63, P62, P21/ANI1, P20/ANI0 P76(CS), P75/SCK01, P74/SI01, P73/SO01 P60/SCLA0, P61/SDLA0	未使用端子

本システムのプリドライバ端子インタフェース一覧を Table 2-3 に示す。

Table 2-3 端子インタフェース(プリドライバ)

端子名	機能
ISENADIN	プリドライバ電圧出力
TOINF	モータ制御信号入力(W_n)
TOINE	モータ制御信号入力(V_n)
TOINC	モータ制御信号入力(W_p)
TOINB	モータ制御信号入力(V_p)
TOIND	モータ制御信号入力(U_n)
TOINA	モータ制御信号入力(U_p)
ALARM	ALARM 信号出力
SI	SPI 用データ入力
SO	SPI 用データ出力
SCLK	SPI 用クロック入力
CS	SPI 用チップセレクト
CLK	システムクロック入力
LD, TOINH, TOING HIC_W, HIC_V, HIC_U	未使用端子
VM	Power Supply
UHOUT	U 相 High-Side Driver(Nch)駆動用出力
U	U 相位相検出用
ULOUT	U 相 Low-Side Driver(Nch)駆動用出力
VHOUT	V 相 High-Side Driver(Nch)駆動用出力
V	V 相位相検出用
VLOUT	V 相 Low-Side Driver(Nch)駆動用出力
WHOUT	W 相 High-Side Driver(Nch)駆動用出力
W	W 相位相検出用
WLOUT	W 相 Low-Side Driver(Nch)駆動用出力
ISENP	シャント抵抗プラス側接続
ISENN	シャント抵抗マイナス側接続
DrvGND	プリドライバ出力段回路 GND
VREG5	Regulator Output (5V)
AGND	プリドライバアナログ回路 GND
UH, VH, WH	未使用端子

2.2.2 周辺機能

本システムで使用する周辺機能の一覧を Table 2-4 に示す。

Table 2-4 周辺機能

周辺機能	機能
A/D コンバータ	位置指令値入力(アナログ値)
	回転速度指令値入力(アナログ値)
	電圧測定(プリドライバ電圧測定/VM 電圧測定/U 相、W 相シャント抵抗電圧測定)
	オプション: 温度測定
汎用ポート	回転方向指令値入力(CW Only)
	LED 出力点灯/消灯制御
	オプション: LED ライト出力、トグルスイッチ/プッシュスイッチ入力
タイマ・アレイ・ユニット	1[ms] インターバルタイマ
	25[us] インターバルタイマ
タイマ RJ	プリドライバ用システムクロック出力
タイマ RG	エンコーダ信号入力(位置検出)
タイマ RD	モータ制御信号出力: 相補 PWM モードを使用した PWM 出力(6 本)
外部割り込み	ALARM 信号検出
通信インタフェース	SPI00(プリドライバ制御用)
	UART1(ICS 用)
	オプション: SPI01, IICA0

(1) A/D コンバータ

位置指令値(アナログ値)、回転速度指令値入力(アナログ値)、電圧測定を「A/D コンバータ」を使用して測定する。

A/D 変換は、ソフトウェアトリガを使用して、チャネル選択モードを「セレクトモード」に、変換動作モードを「ワンショット変換モード」に設定する。

また、A/D 変換の変換時間は、1 チャネル辺り 2.375[us]で、変換入力値の最小単位を Table 2-5 に示す。

Table 2-5 A/D コンバータ

項目	A/D コンバータ 1 ビット当たりの制御値	チャネル
回転速度指令値入力 (アナログ値)	1.0[LSB] ステップ (位置範囲は、40[LSB] ～ 721[LSB])	ANI6
回転速度指令値入力 (アナログ値)	1.066[rpm] ステップ (回転速度範囲は、CW: 50[rpm] ～ 750[rpm])	
電圧測定	VM 電圧測定: 45.9[V] / 1024 = 0.045[V]	ANI3
	プリドライバ電圧[注3]測定: 48.0[V] / 1024 = 0.047[V]	ANI7
	U 相シャント抵抗電圧測定: 5.0[V] / 1024 = 0.005[V]	ANI5
	W 相シャント抵抗電圧測定: 5.0 [V] / 1024 = 0.005[V]	ANI2

注:

3. プリドライバ電圧は、プリドライバ側の ADC セレクタ用レジスタ(ADC_SEL)の設定により A/D 変換する信号を切り換えて測定することができる。 ADC_SEL に 0x00(VM 電圧検出)を設定した場合の制御値が反映される。

   詳細は、「RAJ306000 シリーズ ユーザーズマニュアル ハードウェア編 (R18UZ0066JJ0100)」を参照して下さい。

(2) タイマ・アレイ・ユニット

1[ms] インターバルタイマは、タイマ・アレイ・ユニットの「インターバルタイマ機能」を使用する。本システムでは、チャネル 0 を使用する。

25[us] インターバルタイマは、タイマ・アレイ・ユニットの「インターバルタイマ機能」を使用する。本システムでは、チャネル 2 を使用する。

また、本システムでは、チャネル 1 とチャネル 3 を使用しない。

(3) タイマ RJ

パルス出力モードを使用して、4[MHz]の矩形波を出力、プリドライバのシステムクロックとして供給する。

(4) タイマ RG

位相計数モードを使用して、エンコーダからの入力パルスをカウントする。

タイマ入力端子とエンコーダ信号入力の組み合わせを Table 2-6 に示す。

Table 2-6 タイマ入力端子とエンコーダ信号入力

端子名	エンコーダ信号
P01/TRGCLKB	B 相
P00/TRGCLKA	A 相

(5) タイマ RD

相補 PWM モードを使用して、三角波変調、短絡防止時間ありの三相波形(6 本)を出力する。

本システムでは、周期が 50[us]で、High アクティブの PWM 出力を実現する。また、ALARM 検出時(INTP4 端子に Low 入力時)は、プリドライバの出力をハイインピーダンス状態(モータ制御信号出力端子を Low 状態)にする。

タイマ出力端子とモータ制御信号出力の組み合わせを Table 2-7 に示す。

Table 2-7 タイマ出力端子とモータ制御信号出力

端子名	モータ制御信号
P10/TRDIOD1	W_n
P11/TRDIOC1	V_n
P12/TRDIOB1	W_p
P13/TRDIOA1	V_p
P14/TRDIOD0	U_n
P15/TRDIOB0	U_p

(6) 割り込み

本システムで使用する割り込みの一覧を Table 2-8 に示す。

Table 2-8 割り込み

端子名	割り込み
P55/INTP4	ALARM 信号検出
INTTM00	1[ms] インターバルタイマ
INTTM02	25[us] インターバルタイマ
INTTRD0	キャリア周波数(PWM)
INTCSI00	SPI00(プリドライバ制御用)通信完了

2.3 ソフトウェア構成

2.3.1 ファイル構成

サンプルプログラムのファイル構成を Table 2-9, Table 2-10 に示す。

Table 2-9 サンプルプログラムのファイル構成(1)

RAJ306000_ENCD_FOC_CLOSED_CSP_CA_V102 RAJ306000_ENCD_FOC_CLOSED_CSP_CC_V102 RAJ306000_ENCD_FOC_CLOSED_E2S_CC_V102		
Inc	control_parameter.h	制御特性依存部ヘッダ
	motor_parameter.h	モータ特性依存部ヘッダ
	mtr_calc_encd_foc.h	エンコーダ特性依存部ヘッダ
	mtr_common.h	共通定義用ヘッダ
	mtr_ctrl_rl78g1f.h	RL78/G1F 依存処理部ヘッダ
	mtr_ctrl_rl78g1f_t2001.h	RL78/G1F & ボード依存処理部ヘッダ
	mtr_ctrl_t2001.h	ボード依存処理部ヘッダ
	mtr_main.h	メイン関数、ユーザインタフェース制御ヘッダ
	mtr_ssns_encd_foc.h	エンコーダ利用ベクトル制御依存部ヘッダ
	r_dsp.h	演算ライブラリヘッダ
	r_stdint.h	型定義用ヘッダ
	version.h	ソフトウェアリビジョン定義用ヘッダ
ics	ICS2_CA_RL78G1F.lib	ICS 用ライブラリ(CA78K0R 用)[注4]
	ICS2_CC_RL78G1F.lib	ICS 用ライブラリ(CC-RL 用)[注5]
	ics2_RL78G1F.h	ICS 用ヘッダ
	RL78_vector.c	ICS 用割り込みハンドラ
	RL78_vector.h	ICS 用割り込みハンドラヘッダ
lib	R_dsp_rl78_CA.lib	演算ライブラリ(CA78K0R 用)[注4]
	R_dsp_rl78_CC.lib	演算ライブラリ(CC-RL 用)[注5]
src	mtr_calc_encd_foc.c	エンコーダ特性依存部
	mtr_ctrl_rl78g1f.c	RL78/G1F 依存処理部
	mtr_ctrl_rl78g1f_t2001.c	RL78/G1F & ボード依存処理部
	mtr_ctrl_t2001.c	ボード依存処理部
	mtr_interrupt.c	割り込みハンドラ、ベクトル制御
	mtr_main.c	メイン関数、ユーザインタフェース制御
	mtr_ssns_encd_foc.c	エンコーダ利用ベクトル制御依存部

注:

4. CA78K0R 用は、RAJ306000_ENCD_FOC_CLOSED_CSP_CA_V102 にのみ含まれます。
5. CC-RL 用は、RAJ306000_ENCD_FOC_CLOSED_CSP_CC_V102 と RAJ306000_ENCD_FOC_CLOSED_E2S_CC_V102 にのみ含まれます。

Table 2-10 サンプルプログラムのファイル構成(2)

RAJ306000_ENCD_FOC_CLOSED_CSP_CA_V102 RAJ306000_ENCD_FOC_CLOSED_CSP_CC_V102 RAJ306000_ENCD_FOC_CLOSED_E2S_CC_V102		
cg_src	r_cg_adc.c	RL78/G1F A/D コンバータ処理
	r_cg_adc.h	RL78/G1F A/D コンバータ処理ヘッダ
	r_cg_adc_user.c	RL78/G1F A/D コンバータ処理(ユーザ用)
	r_cg_cgc.c	RL78/G1F クロック出力処理
	r_cg_cgc.h	RL78/G1F クロック出力処理ヘッダ
	r_cg_cgc_user.c	RL78/G1F クロック出力処理(ユーザ用)
	r_cg_intp.c	RL78/G1F 割り込み機能処理
	r_cg_intp.h	RL78/G1F 割り込み機能処理ヘッダ
	r_cg_intp_user.c	RL78/G1F 割り込み機能処理(ユーザ用)
	r_cg_macrodriver.h	RL78/G1F エラー定義用ヘッダ
	r_cg_main.c	RL78/G1F メイン処理
	r_cg_main.h	RL78/G1F メイン処理ヘッダ
	r_cg_port.c	RL78/G1F 端子機能処理
	r_cg_port.h	RL78/G1F 端子機能処理ヘッダ
	r_cg_port_user.c	RL78/G1F 端子機能処理(ユーザ用)
	r_cg_predrv.c	プリドライバ処理
	r_cg_predrv.h	プリドライバ処理ヘッダ
	r_cg_predrv_prm.h	プリドライバレジスタパラメータ定義用ヘッダ
	r_cg_predrv_reg.h	プリドライバレジスタアドレス定義用ヘッダ
	r_cg_predrv_user.c	プリドライバ処理(ユーザ用)
	r_cg_sau.c	RL78/G1F シリアル・アレイ・ユニット処理
	r_cg_sau.h	RL78/G1F シリアル・アレイ・ユニット処理ヘッダ
	r_cg_sau_user.c	RL78/G1F シリアル・アレイ・ユニット処理(ユーザ用)
	r_cg_systeminit.c	RL78/G1F 初期化処理
	r_cg_tau.c	RL78/G1F タイマ・アレイ・ユニット処理
	r_cg_tau.h	RL78/G1F タイマ・アレイ・ユニット処理ヘッダ
	r_cg_tau_user.c	RL78/G1F タイマ・アレイ・ユニット処理(ユーザ用)
	r_cg_tmrd.c	RL78/G1F タイマ RD 処理
	r_cg_tmrd.h	RL78/G1F タイマ RD 処理ヘッダ
	r_cg_tmrd_user.c	RL78/G1F タイマ RD 処理(ユーザ用)
	r_cg_tmrg.c	RL78/G1F タイマ RG 処理
	r_cg_tmrg.h	RL78/G1F タイマ RG 処理ヘッダ
	r_cg_tmrg_user.c	RL78/G1F タイマ RG 処理(ユーザ用)
	r_cg_tmrj.c	RL78/G1F タイマ RJ 処理
	r_cg_tmrj.h	RL78/G1F タイマ RJ 処理ヘッダ
	r_cg_tmrj_user.c	RL78/G1F タイマ RJ 処理(ユーザ用)
	r_cg_userdefine.h	RL78/G1F ユーザ定義用ヘッダ
	r_cg_wdt.c	RL78/G1F ウォッチドッグ・タイマ処理
	r_cg_wdt.h	RL78/G1F ウォッチドッグ・タイマ処理ヘッダ
	r_cg_wdt_user.c	RL78/G1F ウォッチドッグ・タイマ処理(ユーザ用)

2.3.2 モジュール構成

サンプルプログラムの階層構造を Figure 2-4 に、ファイル構成との対応付けを Table 2-11 に示す。

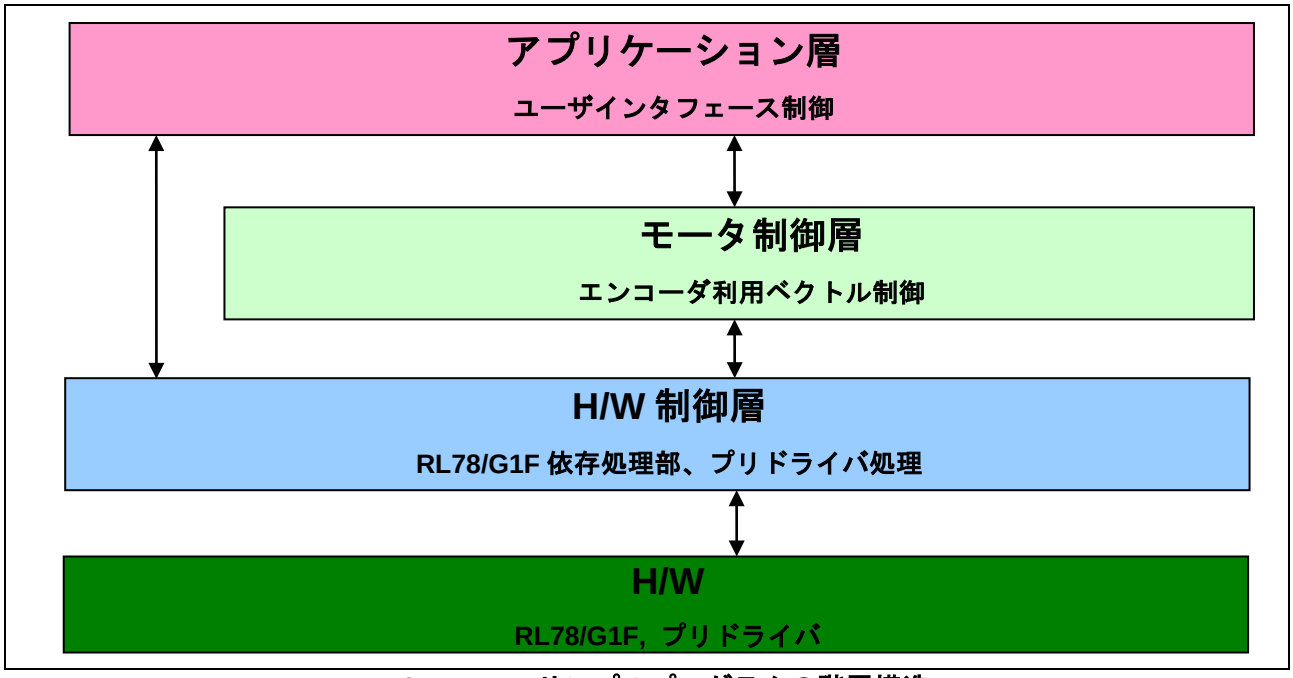

Figure 2-4 サンプルプログラムの階層構造

Table 2-11 サンプルプログラムの階層構造

アプリケーション層	mtr_main.c
モータ制御層	mtr_calc_encd_foc.c, mtr_interrupt.c, mtr_ssns_encd_foc.c
H/W 制御層	mtr_ctrl_rl78g1f.c, mtr_ctrl_rl78g1f_t2001.c, mtr_ctrl_t2001.c, r_cg_adc.c, r_cg_adc_user.c, r_cg_cgc.c, r_cg_cgc_user.c, r_cg_intp.c, r_cg_intp_user.c, r_cg_main.c, r_cg_port.c, r_cg_port_user.c, r_cg_predrv.c, r_cg_predrv_user.c, r_cg_sau.c, r_cg_sau_user.c, r_cg_systeminit.c, r_cg_tau.c, r_cg_tau_user.c, r_cg_tmrd.c, r_cg_tmrd_user.c, r_cg_tmrg.c, r_cg_tmrg_user.c, r_cg_tmrj.c, r_cg_tmrj_user.c, r_cg_wdt.c, r_cg_wdt_user.c

2.4 ソフトウェア仕様

本アプリケーションノートが対象とするソフトウェアの基本仕様を Table 2-12 に示す。

Table 2-12 ソフトウェアの基本仕様

項目	内容
制御方式	ベクトル制御
モータサーボ開始/停止	モータサーボ開始/停止指令は、VR1(ANI6 端子)のレベルによって決定する。 ICS からの入力[注6]
位置制御	位置指令値は、VR1(ANI6 端子)のレベルによって決定する。 ICS からの入力[注6]
回転方向制御	回転方向指令値(CW Only)は、SW1(P122 端子)のレベルによって決定する。
回転速度制御	回転速度指令値は、VR1(ANI6 端子)のレベルによって決定する。 ICS からの入力[注6]
回転速度範囲	CW: 50[rpm] ～ 750[rpm]
回転子磁極位置検出	エンコーダ
キャリア周波数(PWM)	20[KHz]
制御周期	1[ms]
保護停止処理	以下のエラー検出時に、モータ制御信号出力端子を Low 状態にする。 ・ALARM エラー ・回転速度異常エラー ・タイムアウトエラー

注:

6. 詳細は、「4 開発支援ツール「In Circuit Scope」」を参照して下さい。

3. 制御プログラム説明

本アプリケーションノートが対象とするサンプルプログラムについて説明する。

3.1 制御内容

3.1.1 モータサーボ開始/停止

モータのサーボ開始と停止は、VR1 と SW1、または、ICS からの入力によって制御する。

VR1 には、アナログ入力端子(ANI6 端子)が割り当てられ、その入力をメイン・ループ内で A/D 変換して、位置指令値、回転速度指令値を算出する。位置指令値が、80[LSB]以上でモータのサーボ開始、40[LSB]以下で停止と判断する。また、回転速度指令値が、100[rpm]以上でモータのサーボ開始、50[rpm]以下で停止と判断する。

SW1 には、汎用ポート(P122 端子)が割り当てられ、メイン・ループ内で、P122 端子の High/Low 状態を取得して、回転方向指令値(CW Only)とする。回転方向は、回転方向指令値で判断する。

3.1.2 位置指令値、回転方向指令値、回転速度指令値、VM 電圧

(1) 位置指令値

VR1 の出力値(アナログ値)を A/D 変換、または、ICS から入力によってモータの位置指令値を設定する。A/D 変換された VR1 の値は、下記(Table 3-1)のように、位置指令値として使用する。

Table 3-1 位置指令値の変換比

項目	変換比 (位置指令値: A/D 変換値)	チャネル
位置指令値	40[LSB] ～ 721[LSB]: 03FFH ～ 0000H	ANI6

(2) 回転方向指令値

SW1 の High/Low 状態によってモータの回転方向指令値(CW Only)を設定する。

(3) 回転速度指令値

VR1 の出力値(アナログ値)を A/D 変換、または、ICS から入力によってモータの回転速度指令値を設定する。A/D 変換された VR1 の値は、下記(Table 3-2)のように、回転速度指令値として使用する。

Table 3-2 回転速度指令値の変換比

項目	変換比 (回転速度指令値: A/D 変換値)	チャネル
回転速度指令値	50[rpm] ～ 750[rpm]: 03FFH ～ 0000H	ANI6

(4) VM 電圧

下記(Table 3-3)のように、VM 電圧を測定する。

Table 3-3 VM 電圧の変換比

項目	変換比 (VM 電圧: A/D 変換値)	チャネル
VM 電圧	0.0[V] ～ 45.9[V]: 0000H ～ 03FFH	ANI3

(5) U 相、W 相シャント抵抗電圧

下記(Table 3-4)のように、U 相，W 相シャント抵抗電圧を測定する。

Table 3-4 U 相、W 相シャント抵抗電圧の変換比

項目	変換比 (U 相、W 相シャント抵抗電圧: A/D 変換値)	チャネル
U 相シャント抵抗電圧	0.0[V] ～ 5.0[V]: 0000H ～ 03FFH	ANI5
W 相シャント抵抗電圧		ANI2

3.1.3 ベクトル制御

本アプリケーションノートが対象とするベクトル制御のブロック図を Figure 3-1 に示す。

ベクトル制御の詳細については、「永久磁石同期モータのエンコーダベクトル制御 アルゴリズム編 (R01AN3789JJ0101)」を参照して下さい。

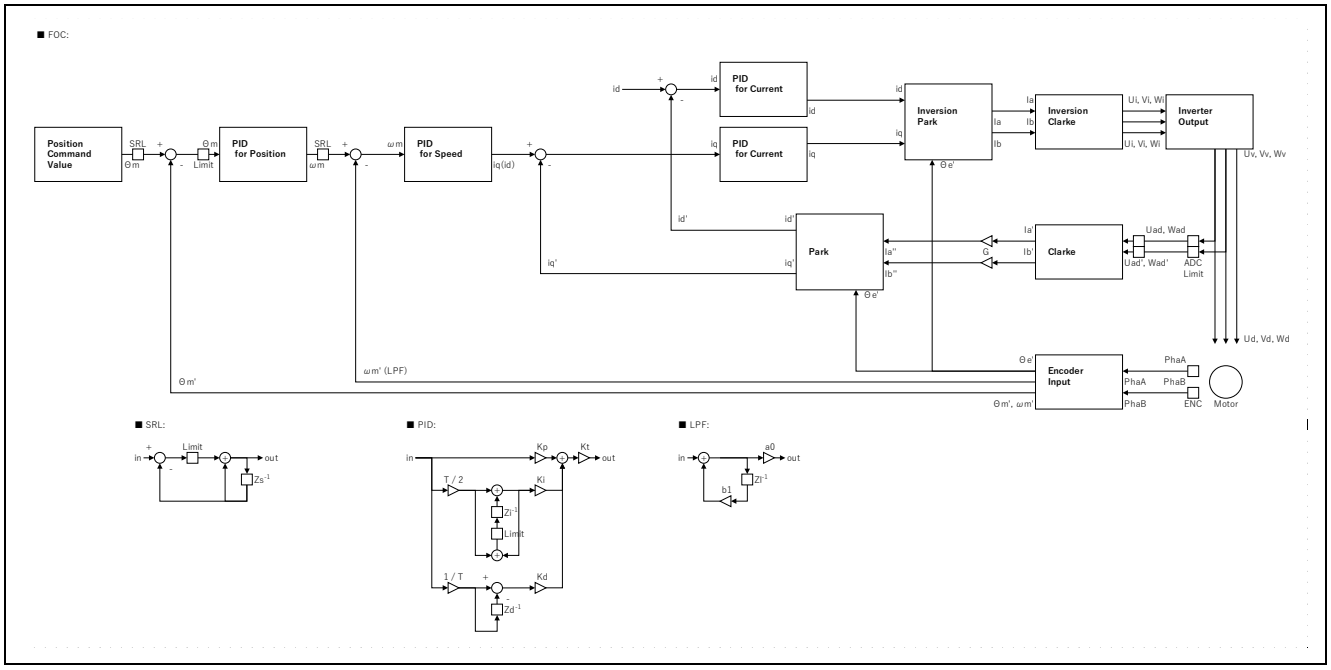

Figure 3-1 ベクトル制御

3.1.4 PWM による電圧制御

出力電圧の制御には、PWM 制御を使用する。PWM 制御とは、パルスの Duty を変化させることで平均電圧を調整していく制御方式です。PWM 制御の概念図を Figure 3-2 に示す。

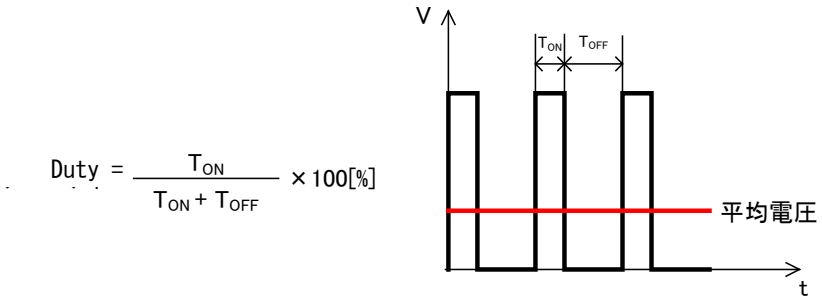

Figure 3-2 PWM 制御

ここで、変調率 m を以下のように定義する。

この変調率を、PWM Duty を決めるレジスタの設定値に反映する。

$$m = \frac{V}{E}$$

m：変調率　　V：電圧指令値　　E：VM電圧

3.1.5 状態遷移

本アプリケーションノートが対象とするサンプルプログラムの状態遷移図を Figure 3-3 に示す。

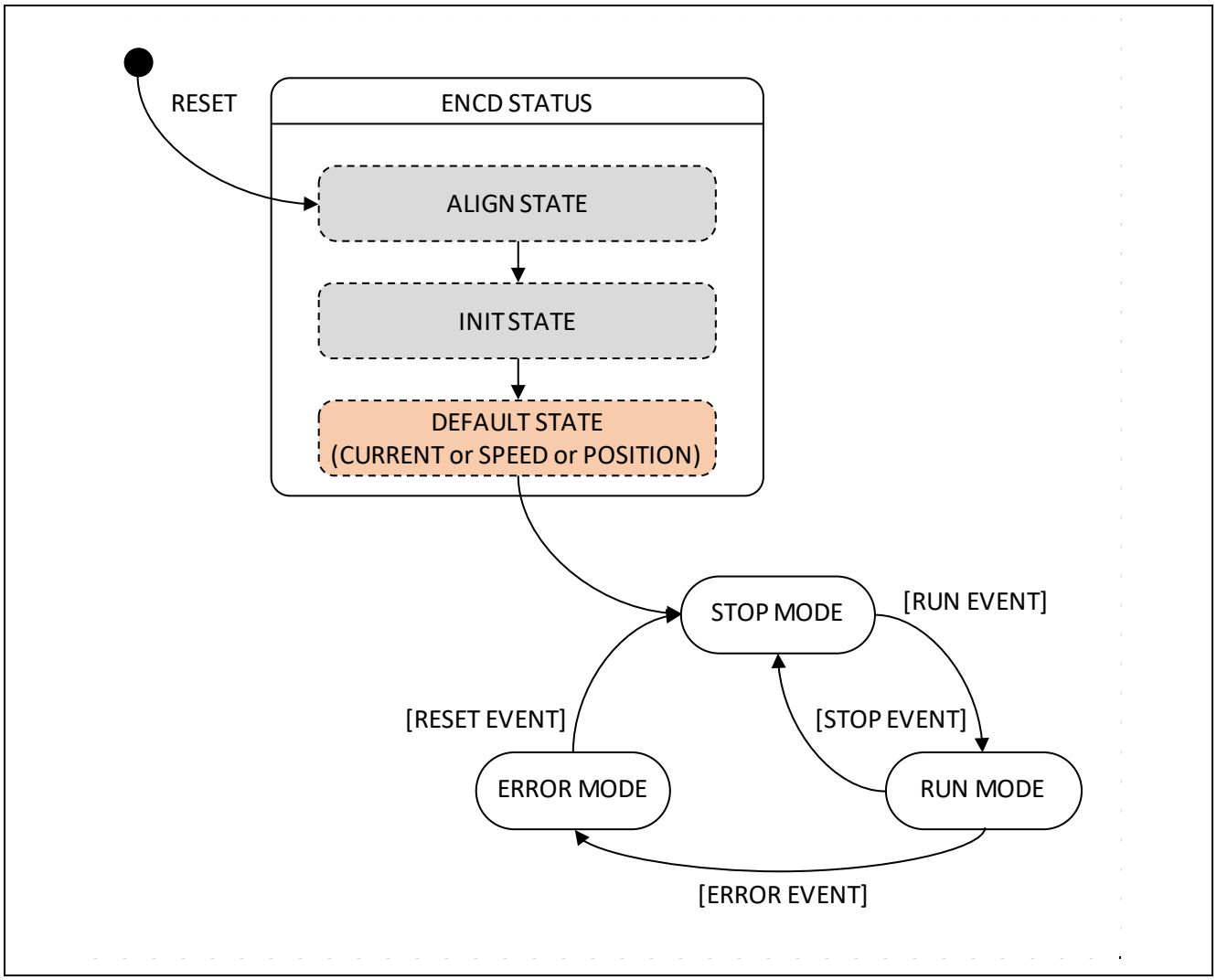

Figure 3-3 状態遷移図

3.1.6 システム保護機能

本システムは、以下のエラー状態を持ち、それぞれの場合に緊急停止機能を実現する。システム保護機能に関わる各設定値を Table 3-5 に示す。

・ALARM エラー

プリドライバからの緊急停止信号(ALARM 検出)によって、プリドライバの出力をハイインピーダンス状態(モータ制御信号出力端子を Low 状態)にして、緊急停止する。

・回転速度異常エラー

回転速度監視周期で回転速度を監視して、回転速度リミット値を超過した場合、緊急停止する。

・タイムアウトエラー

タイムアウトエラー監視周期で制御周期の間隔を監視して、タイムアウト時間を超過してもモータ制御信号出力の変更が発生していない場合、緊急停止する。

Table 3-5 システム保護機能設定値

システム保護機能		設定値
回転速度異常エラー	回転速度リミット値	1000[rpm]
	監視周期	50[us]
タイムアウトエラー	タイムアウト時間	20[ms]

3.1.7 システム保護機能(プリドライバ安全機能)

ALARM 動作設定レジスタ(ALMOPE)で、プリドライバ安全機能の有効/無効を設定可能です。

詳細は、データシートを参照して下さい。

3.2 関数仕様

サンプルプログラムの関数一覧を Table 3-6, Table 3-7 に示す。

Table 3-6 関数一覧(1)

ファイル名	関数概要	処理概要
mtr_main.c	main() 入力: なし 出力: なし	・ハードウェア初期化関数呼び出し ・ユーザインタフェース初期化関数呼び出し ・メイン関数使用変数初期化関数呼び出し ・状態遷移、及び、イベント実行関数呼び出し ・メイン関数 ⇒ メイン処理実行関数呼び出し ⇒ ウォッチドッグ・タイマクリア関数呼び出し
	ctrl_ui() 入力: なし 出力: なし	・モータステータスの変更 ・位置指令値と回転速度指令値の決定
	ics_ui() 入力: なし 出力: なし	・モータステータスの変更 ・位置指令値と回転速度指令値の決定
	ctrl_led() 入力: なし 出力: なし	LED 出力点灯/消灯制御
	ics_predrv_reg_ctrl() 入力: なし 出力: なし	ICS からのプリドライバレジスタの読み込み、書き込み制御
	mcu_sw_init() 入力: なし 出力: なし	F/W の初期化 ・F/W 変数の初期化 ・ICS の初期化 ・シーケンス処理の初期化 ・リセットイベントの実行 ・エンコーダ入力演算初期化
	software_init() 入力: なし 出力: なし	メイン関数で使用する変数の初期化
mtr_ctrl_rl78g1f.c	clear_wdt() 入力: なし 出力: なし	ウォッチドッグ・タイマクリア
	mtr_clear_oc_flag() 入力: なし 出力: なし	パルス出力強制遮断フラグクリア
	mtr_clear_trd0_imfa() 入力: なし 出力: なし	TRD0 コンペアマッチフラグクリア

mtr_ctrl_rl78g1f_t2001.c	mtr_ctrl_start() 入力: なし 出力: なし	モータ起動処理
	mtr_ctrl_stop() 入力: なし 出力: なし	モータ停止処理
	mtr_change_pattern() 入力: なし 出力: なし	モータ制御信号出力変更
	mtr_get_adc() 入力: A/D チャネル 出力: A/D 変換結果	A/D 変換実行処理
mtr_ctrl_t2001.c	get_vr1() 入力：なし 出力：A/D 変換結果(VR1)	トリガレベルの A/D 変換値取得
	led_on() 入力: LED チャネル番号 出力: なし	LED の点灯
	led_off() 入力: LED チャネル番号 出力: なし	LED の消灯
mtr_interrupt.c	mtr_alarm_interrupt() 入力: なし 出力: なし	ALARM 割り込み処理 ・モータステータス変更 ・イベント処理選択関数呼び出し ・パルス出力強制遮断フラグクリア関数呼び出し
	mtr_tau0_interrupt() 入力: なし 出力: なし	1[ms] 割り込み処理 ・エンコーダ利用ベクトル制御
	mtr_tau2_interrupt() 入力: なし 出力: なし	25[us] 割り込み処理 ・U 相、W 相シャント抵抗電圧の A/D 変換値取得
	mtr_carrier_interrupt() 入力: なし 出力: なし	キャリア周波数割り込み処理 ・モータサーボ停止待ち ・エラーチェック関数呼び出し

mtr_ssns_encd_foc.c	R_MTR_InitSequence() 入力: なし 出力: なし	シーケンス処理の初期化
	R_MTR_ExecEvent() 入力: 発生イベント 出力: なし	・ステータスの変更を行う ・発生イベントに対して、適切な処理の実行関数を呼び出し
	mtr_act_run() 入力: モータステータス 出力: モータステータス	・モータ起動時変数初期化関数呼び出し ・モータ制御開始関数呼び出し
	mtr_act_stop() 入力: モータステータス 出力: モータステータス	モータ制御終了関数呼び出し
	mtr_act_none() 入力: モータステータス 出力: モータステータス	処理はなし
	mtr_act_reset() 入力: モータステータス 出力: モータステータス	エラー状態から復帰するために必要なグローバル変数の初期化
	mtr_act_error() 入力: モータステータス 出力: モータステータス	エラー発生時のモータ制御終了関数呼び出し
	mtr_pattern_set() 入力: なし 出力: なし	モータ制御信号出力変更関数呼び出し
	mtr_start_init() 入力: なし 出力: なし	モータ起動時に必要な変数の初期化
	mtr_set_variables() 入力: なし 出力: なし	ICS から入力された値をプロテクト変数に設定
	R_MTR_IcsInput() 入力: ICS 関連変数の構造体 出力: なし	ICS から入力された値の取得
	R_MTR_SetSpeed() 入力: 回転速度指令値 出力: なし	回転速度の設定
	R_MTR_SetDir() 入力: 回転方向指令値 出力: なし	回転方向の設定
	R_MTR_GetDir() 入力: なし 出力: 回転方向情報	回転方向の取得
	R_MTR_GetStatus() 入力: なし 出力: モータステータス	モータステータスを取得
	mtr_error_check() 入力: なし 出力: なし	エラーの監視と検出

	R_MTR_InitEncoderVector() 入力: なし 出力: なし	エンコーダ利用ベクトル制御用変数の初期化
	R_MTR_SetFlagCtrlDuty() 入力: PWM Duty 制御フラグ 出力: なし	PWM Duty 制御フラグの設定
mtr_calc_encd_foc.c	mtr_calc_LPF() 入力: LPF 入力 出力: LPF 演算結果	LPF 演算処理
	mtr_calc_InitVariables() 入力: なし 出力: なし	エンコーダ利用ベクトル制御用変数の初期化
	mtr_calc_PID() 入力: PID 入力 PID パラメータ 出力: PID 演算結果	PID 演算処理
	mtr_calc_SRL() 入力: SRL 入力 遅延素子 リミット値 出力: SRL 演算結果 遅延素子	SRL 演算処理
	mtr_calc_EncoderInput() 入力: なし 出力: 位置演算結果(電気角) 位置演算結果(機械角) 回転速度演算結果(機械角)	エンコーダ入力演算処理

Table 3-7 関数一覧(2)

ファイル名	関数概要	処理概要
r_cg_adc.c	R_ADC_Create() 入力: なし 出力: なし	A/D 変換器(ADC)初期化
r_cg_adc_user.c	r_adc_interrupt() 入力: なし 出力: なし	SPI 通信 ADC モード SPI 開始判定
r_cg_cgc.c	R_CGC_Create() 入力: なし 出力: なし	クロック周波数(CGC)初期化
r_cg_intp.c	R_INTP_Create() 入力: なし 出力: なし	外部割り込み(INTP)初期化
r_cg_main.c	R_MAIN_UserInit() 入力: なし 出力: なし	プリドライバ起動処理
r_cg_port.c	R_PORT_Create() 入力: なし 出力: なし	入出力ポート初期化
r_cg_predrv.c	predriver_hw_init() 入力: なし 出力: なし	プリドライバ初期設定
	R_PREDRV_TRIM_Create() 入力: なし 出力: SPI ステータス	プリドライバトリミングデータ設定
	R_PREDRV_InitSequence() 入力: なし 出力: なし	プリドライバ初期化処理
	R_PREDRV_ ErrorRecoverySequence() 入力: ALARM ステータス 出力: なし	ALARM 復帰処理
r_cg_predrv_user.c	R_PreDrvReg_Read() 入力: read address 出力: SPI ステータス, read data	プリドライバレジスタ Read 処理
	R_PreDrvReg_Write() 入力: write address, write data 出力: SPI ステータス	プリドライバレジスタ Write 処理

r_cg_sau.c	R_SAU0_Create() 入力: なし 出力: なし	シリアル・アレイ・ユニット(SAU)初期化
	R_UART1_Create() 入力: なし 出力: なし	UART1 初期化
	R_CSI00_Create() 入力: なし 出力: なし	SPI 通信初期化(プリドライバ通信用)
	R_CSI00_Start() 入力: なし 出力: なし	SPI 通信起動(プリドライバ通信用)
	R_CSI00_ Send_Receive_SPI_mode() 入力: 送信バッファ バッファサイズ 受信バッファ SPI モード 出力: SPI ステータス	SPI 通信処理
r_cg_sau_user.c	r_csi00_interrupt() 入力: なし 出力: なし	SPI 割り込み処理(プリドライバ通信用)
r_cg_systeminit.c	hdwinit() 入力: なし 出力: なし	H/W 初期設定
r_cg_tau.c	R_TAU0_Create() 入力: なし 出力: なし	タイマ・アレイ・ユニット(TAU)初期化
	R_TAU0_Channel0_Start() 入力: なし 出力: なし	1[ms] インターバルタイマカウント開始
	R_TAU0_Channel2_Start() 入力: なし 出力: なし	25[us] インターバルタイマカウント開始
	R_TAU0_Channel2_Stop() 入力: なし 出力: なし	25[us] インターバルタイマカウント停止
r_cg_tmrd.c	R_TMRD0_Create() 入力: なし 出力: なし	タイマ RD(TRD)初期化
	R_TMRD0_Start() 入力: なし 出力: なし	PWM 出力開始
r_cg_tmrg.c	R_TMRG0_Create() 入力: なし 出力: なし	タイマ RG(TRG)初期化
	R_TMRG0_Start() 入力: なし 出力: なし	エンコーダカウント開始

r_cg_tmrj.c	R_TMRJ0_Create() 入力: なし 出力: なし	タイマ RJ(TRJ)初期化
	R_TMRJ0_Start() 入力: なし 出力: なし	プリドライバクロック供給開始
r_cg_wdt.c	R_WDT_Create() 入力: なし 出力: なし	ウォッチドッグ・タイマ初期化

3.3 変数仕様

サンプルプログラムの変数一覧を Table 3-8 に示す。

Table 3-8 変数一覧

変数名	型	内容	備考
g_s2_min_angle	Int16_t	位置指令最大値	トリガレベル A/D 変換値
g_s2_min_angle	Int16_t	位置指令最小値	トリガレベル A/D 変換値
g_s2_margin_min_angle	Int16_t	モータ停止用回位置指令最小値	トリガレベル A/D 変換値
g_s2_max_speed	int16_t	回転速度指令最大値	機械角 [rpm]
g_s2_min_speed	int16_t	回転速度指令最小値	機械角 [rpm]
g_s2_margin_min_speed	int16_t	モータ停止用回転速度指令最小値	機械角 [rpm]
g_s2_ref_speed	int16_t	ユーザ設定回転速度	機械角 [rpm]
g_u1_rot_dir	uint8_t	ユーザ設定回転方向	0: CW 1: CCW
g_u1_motor_status	uint8_t	ユーザモータステータス管理	0: 停止 1: 回転中 2: エラー
g_u1_stop_req	uint8_t	モータ停止指令フラグ	位置指令値 40[LSB]以下、 回転速度指令値 50[rpm]以下は停止判定
g_u1_pdrv_status	uint8_t	プリドライバレジスタ R/W エラーステータス	-
g_u1_err_recovery_req	uint8_t	ALARM 復帰処理要求フラグ	0: 禁止 1: 許可
g_u1_get_alarm_sts1	uint8_t	プリドライバレジスタ ALMSTS1 取得値	-
g_u1_store_alarm_sts1	uint8_t	プリドライバレジスタ ALMSTS1 保存値	-
g_u1_get_alarm_sts2	uint8_t	プリドライバレジスタ ALMSTS2 取得値	-
g_u1_store_alarm_sts2	uint8_t	プリドライバレジスタ ALMSTS2 保存値	-
g_u2_fw_revision	uint16_t	F/W リビジョン情報	F/W バージョン情報(102)
g_s2_sw_userif	int16_t	ボード UI 使用切り替えフラグ	0: 使用しない 1: 使用する
g_s2_mode_system	int16_t	システムモードフラグ	0: 停止 1: モータ駆動 2: エラー 3: リセット
g_s2_enable_write	int16_t	ICS 書き込み許可フラグ	トグル動作
ics_input	MTR_ICS_INPUT	ICS 入力用構造体	-
g_s2_ref_speed_rpm_vr1	int16_t	回転速度指令値	機械角 [rpm]
g_u1_alarm_sts1	uint8_t	プリドライバレジスタ ALMSTS1 保存値	ICS 表示用
g_u1_alarm_sts2	uint8_t	プリドライバレジスタ ALMSTS2 保存値	ICS 表示用
g_u2_cnt_wait_stop	uint16_t	モータサーボ停止待ちカウンタ	モータ停止処理後 10[ms]をカウント(ただしホール IC 信号割り込み検出時にカウントリセット)

g_u1_flg_wait_stop	uint8_t	モータサーボ停止待ちフラグ	モータ停止指令を受けてセットして、モータ停止処理後ホール IC 信号割り込みが 10[ms]の間未検出の場合クリア
g_u1_enable_write	uint8_t	ICS 入力用構造体書き込み許可フラグ	0: 禁止 1: 許可
g_s2_vdc_ad	int16_t	VM 電圧 A/D 変換値	[V]
g_s2_pdrv_ad	int16_t	プリドライバ電圧 A/D 変換値	[V]
g_u1_cnt_ics	uint8_t	ICS 関数呼び出し間隔カウンタ	-
g_u1_error_status	uint8_t	エラーステータス管理	0x01: ALARM エラー 0x04: 回転速度異常エラー 0x08: タイムアウトエラー (0x80: 未定義エラー)
g_u1_mode_system	uint8_t	ステート管理	0: ストップモード 1: ランモード 2: エラーモード
g_s2_foc_encd_status	int16_t	エンコーダ利用ベクトル制御ステータス管理	0: 初期化処理(位置調整) 1: 初期化処理(初期位置移動) 2: 電流制御 3: 回転速度制御 4: 位置制御
g_s2_foc_mech_count	int16_t	エンコーダカウント	機械角
g_s2_foc_elec_count	int16_t	エンコーダカウント	電気角
g_s2_foc_ref_theta	int16_t	位置指令値	機械角(正規化: 0 ～ 4095) [rad]
g_s2_foc_ref_omega	int16_t	回転速度指令値	機械角(Scale: Q4) [rad/s]
g_s2_foc_theta	int16_t	位置演算値	機械角(正規化: 0 ～ 4095) [rad]
g_s2_foc_theta_elec	int16_t	エンコーダ位置	電気角(正規化: 0 ～ 4095) [rad]
g_s2_foc_theta_mech	int16_t	エンコーダ位置	機械角(正規化: 0 ～ 4095) [rad]
g_s2_foc_theta_zs	int16_t	位置 SRL 遅延素子	-
g_f4_foc_theta_zi	float32_t	位置 PID 制御積分遅延素子	-
g_f4_foc_theta_zd	float32_t	位置 PID 制御微分遅延素子	-
pid_foc_theta	MTR_PID_PRM	位置制御用 PID パラメータ	-
g_s2_foc_omega	int16_t	回転速度演算値	機械角(Scale: Q4) [rad/s]
g_s2_foc_omega_mech	int16_t	エンコーダ回転速度	機械角(Scale: Q4) [rad/s]
g_s2_foc_omega_zs	int16_t	速度 SRL 遅延素子	-
g_f4_foc_omega_zi	float32_t	速度 PID 制御積分遅延素子	-
g_f4_foc_omega_zd	float32_t	速度 PID 制御微分遅延素子	-
pid_foc_omega	MTR_PID_PRM	回転速度制御用 PID パラメータ	-
g_s2_foc_id	int16_t	d 軸電流演算値	逆パーク変換入力値(Scale: Q12)
g_f4_foc_id_zi	float32_t	d 軸電流 PID 制御積分遅延素子	-
g_f4_foc_id_zd	float32_t	d 軸電流 PID 制御微分遅延素子	-
pid_foc_id	MTR_PID_PRM	d 軸電流制御用 PID パラメータ	-
g_s2_foc_iq	int16_t	q 軸電流演算値	逆パーク変換入力値(Scale: Q12)
g_f4_foc_iq_zi	float32_t	q 軸電流 PID 制御積分遅延素子	-
g_f4_foc_iq_zd	float32_t	q 軸電流 PID 制御微分遅延素子	-
pid_foc_iq	MTR_PID_PRM	q 軸電流制御用 PID パラメータ	-
g_s2_foc_ia	int16_t	α 軸電流値	逆パーク変換出力値, 逆クラーク変換入力値(Scale: Q12)
g_s2_foc_ib	int16_t	β 軸電流値	
g_s2_foc_ui	int16_t	U 相電流値	逆クラーク変換出力値(Scale: Q12)
g_s2_foc_vi	int16_t	V 相電流値	
g_s2_foc_wi	int16_t	W 相電流値	
g_f4_foc_coef_k	float32_t	A/V 変換係数	-
g_s2_foc_uv	int16_t	U 相電圧値(PWM Duty)	[%]

g_s2_foc_vv	int16_t	V 相電圧値(PWM Duty)	[%]
g_s2_foc_wv	int16_t	W 相電圧値(PWM Duty)	[%]
g_u2_foc_ud	uint16_t	U 相タイマ RD コンペアレジスタ設定値	-
g_u2_foc_vd	uint16_t	V 相タイマ RD コンペアレジスタ設定値	-
g_u2_foc_wd	uint16_t	W 相タイマ RD コンペアレジスタ設定値	-
g_s2_foc_uad_offset	int16_t	U 相シャント抵抗電圧オフセット値	シャント抵抗電圧値 A/D 変換値
g_s2_foc_wad_offset	int16_t	W 相シャント抵抗電圧オフセット値	
g_s2_foc_uad	int16_t	U 相シャント抵抗電圧値	シャント抵抗電圧値 A/D 変換値
g_s2_foc_wad	int16_t	W 相シャント抵抗電圧値	
g_s2_foc_adc_toggle	int16_t	A/D 変換トグルスイッチ	2: U 相シャント抵抗電圧測定 5: W 相シャント抵抗電圧測定
g_s2_foc_uadd	int16_t	U 相 AMP 出力電圧値	クラーク変換入力値(Scale: Q12)
g_s2_foc_wadd	int16_t	W 相 AMP 出力電圧値	
g_s2_foc_iad	int16_t	α 軸電流値	クラーク変換出力値(Scale: Q12)
g_s2_foc_ibd	int16_t	β 軸電流値	
g_s2_foc_iadd	int16_t	α 軸電流値	パーク変換入力値(Scale: Q12)
g_s2_foc_ibdd	int16_t	β 軸電流値	
g_s2_foc_idd	int16_t	d 軸電流値	パーク変換出力値(Scale: Q12)
g_s2_foc_iqd	int16_t	q 軸電流値	
g_u2_foc_flg_ctrl_duty	uint16_t	PWM Duty 制御フラグ	0: 禁止 1: 許可
g_u2_cnt_timeout	uint16_t	停止判定時間計測カウンタ	モータ制御信号出力変更関数呼び出し時にクリア
g_u1_direction	uint8_t	回転方向管理	0: CW 1: CCW
ics_input_buff	MTR_ICS_INPUT	ICS 入力変数構造体	-
g_u2_foc_encd_count	uint16_t	エンコーダカウント前回値	-
g_s2_foc_calc_count	int16_t	エンコーダカウント演算値	-
g_f4_foc_lpf_zl	float32_t	速度 LPF 遅延素子	-
g_u1_PreDriver_error	uint8_t	プリドライバシーケンスエラーステータス	・プリドライバ初期化シーケンス ・ALARM 復帰シーケンス
g_spi00_comend_flag	uint8_t	SPI 通信状態 フラグ	TURE : 通信終了 FALSE: 通信中
g_spi00_adcend_flag	uint8_t	SPI 通信 ADC End フラグ	TURE : ADC 終了 FALSE: ADC 中
g_spi00_commode	uint8_t	SPI 通信モード	-
gp_csi00_rx_address	uint8_t	SPI 通信受信データアドレス	プリドライバレジスタ値取得
g_csi00_rx_length	uint16_t	SPI 通信受信データ長	-
g_csi00_rx_count	uint16_t	SPI 通信受信カウンタ	-
gp_csi00_tx_address	uint8_t	SPI 通信送信データアドレス	プリドライバレジスタアドレス指定
g_csi00_send_length	uint16_t	SPI 通信送信データ長	-
g_csi00_tx_count	uint16_t	SPI 通信送信カウンタ	-

3.4 マクロ定義仕様

サンプルプログラムのマクロ定義一覧を Table 3-9 に示す。

Table 3-9 マクロ定義一覧

ファイル名	マクロ名	内容	備考
control_parameter.h	CP_MAX_SPEED_RPM	750	回転速度指令最大値(機械角) [rpm]
	CP_MIN_SPEED_RPM	100	回転速度指令最小値(機械角) [rpm]
mtr_main.h	ICS_UI	0	UI を ICS にセット
	BOARD_UI	1	UI をボードにセット
	M_CW	0	ユーザ回転方向設定値: CW
	M_CCW	1	ユーザ回転方向設定値: CCW
	MAX_ANGLE	721	位置指令最大値(トリガレベル A/D 変換値)
	MIN_ANGLE	80	位置指令最小値(トリガレベル A/D 変換値)
	MARGIN_ANGLE	40	モータ停止用位置指令最小値作成用定数(トリガレベル A/D 変換値)
	MARGIN_MIN_ANGLE	(MIN_ANGLE - MARGIN_ANGLE)	モータ停止用位置指令最小値(トリガレベル A/D 変換値)
	MAX_SPEED	CP_MAX_SPEED_RPM	回転速度指令最大値(機械角) [rpm]
	MIN_SPEED	CP_MIN_SPEED_RPM	回転速度指令最小値(機械角) [rpm]
	MARGIN_SPEED	50	モータ停止用回転速度指令最小値作成用定数(機械角) [rpm]
	MARGIN_MIN_SPEED	(MIN_SPEED - MARGIN_SPEED)	モータ停止用回転速度指令最小値(機械角) [rpm]
	REQ_CLR	0	モータ停止指令フラグクリア
	REQ_SET	1	モータ停止指令フラグセット
	LED_ON_1ST_SPEED	250	LED3 点灯回転数
	LED_ON_2ND_SPEED	500	LED4 点灯回転数
	REQ_ROT_CCW	0	回転方向ポート取得値: CCW
	REQ_ROT_CW	1	回転方向ポート取得値: CW
motor_parameter.h	MP_POLE_PAIRS	7	極対数補正用定数
mtr_ctrl_rl78g1f_t2001.h	MTR_PWM_TIMER_FREQ	64.0f	PWM タイマカウント周波数 [MHz]
	MTR_CARRIER_FREQ	20.0f	キャリア周波数 [KHz]
	MTR_DEADTIME	0	デットタイム [ns]

	MTR_DEADTIME_SET	((MTR_DEADTIME * MTR_PWM_TIMER_FREQ) / 1000)	デットタイム設定値
	MTR_CARRIER_SET	((((MTR_PWM_TIMER_FREQ * 1000) / MTR_CARRIER_FREQ) / 2) + MTR_DEADTIME_SET - 2)	キャリア設定値
	MTR_START_CARRIER_SET	((MTR_CARRIER_SET * 50) / 100	キャリア設定値(初期値)
	MTR_VR1_ADC_MAX	802	トリガレベル A/D 変換最大値
	MTR_THETA_CALC_COEF	(4095.0f / MTR_VR1_ADC_MAX)	位置指令値演算係数
	MTR_OMEGA_CALC_COEF	(MTR_TWOPI / 60.0f * 16.0f)	回転速度指令値演算係数
	MTR_RPM_CALC_COEF	(1221.7f / MTR_VR1_ADC_MAX)	目標回転数演算係数
	MTR_GET_ROT_DIR_REQ	P12.2	回転方向取得ポート
	MTR_PORT_LED3	P14.1	LED3 出力ポート
	MTR_PORT_LED4	P14.0	LED4 出力ポート
	MTR_PORT_LED5	P4.3	LED5 出力ポート
	MTR_PORT_LED6	P4.2	LED6 出力ポート
	MTR_LED_ON	0	Low アクティブ
	MTR_LED_OFF	1	
	MTR_VDC_SCALING	1471	VM 電圧 A/D 変換値分解能
	MTR_ADCCH_RAJ306000_TEMP	0	RAJ306000 温度測定用 A/D 変換チャネル
	MTR_ADCCH_MOS_TEMP	1	MOS 温度測定用 A/D 変換チャネル
	MTR_ADCCH_IU	2	U 相シャント抵抗電圧 A/D 変換チャネル
	MTR_ADCCH_VM	3	VM 電圧測定用 A/D 変換チャネル
	MTR_ADCCH_IW	5	W 相シャント抵抗電圧 A/D 変換チャネル
	MTR_ADCCH_VR1	6	トリガレベル A/D 変換チャネル
	MTR_ADCCH_PDRV	7	プリドライバ電圧測定用 A/D 変換チャネル
mtr_ctrl_t2001.h	MTR_LED3	3	LED パターン
	MTR_LED4	4	
	MTR_LED5	5	
	MTR_LED6	6	
mtr_ssns_encd_foc.h	MTR_TWOPI	(2 * 3.14159265f)	2π
	MTR_SPEED_LIMIT_RPM	1000	回転速度リミット値(機械角) [rpm]
	MTR_SPEED_LIMIT	((MTR_SPEED_LIMIT_RPM / 60.0f) * MTR_TWOPI * 16.0f)	回転速度リミット値(機械角) [rad/s]
	MTR_TIMEOUT_CNT	400	停止判定時間

			(カウント値 * 50[us])
	MTR_CW	0	回転方向設定値: CW
	MTR_CCW	1	回転方向設定値: CCW
	MTR_FLG_CLR	0	フラグクリア用定数
	MTR_FLG_SET	1	フラグセット用定数
	MTR_STOP_WAIT_CNT	200	モータ停止待ち時間 (カウント値 * 50[us])
	MTR_ICS_DECIMATION	4	ICS 用関数呼び出し間引き数 (カウント値 * 50[us])
	MTR_ALARM_ERROR	0x01	ALARM エラー
	MTR_OVER_SPEED_ERROR	0x04	回転速度異常度エラー
	MTR_TIMEOUT_ERROR	0x08	タイムアウトエラー
	MTR_UNKNOWN_ERROR	0x80	未定義エラー
	MTR_MODE_STOP	0x00	停止状態
	MTR_MODE_RUN	0x01	回転中
	MTR_MODE_ERROR	0x02	エラー状態
	MTR_SIZE_STATE	3	状態数
	MTR_EVENT_STOP	0x00	モータ停止イベント
	MTR_EVENT_RUN	0x01	モータ起動イベント
	MTR_EVENT_ERROR	0x02	モータエラーイベント
	MTR_EVENT_RESET	0x03	モータリセットイベント
	MTR_SIZE_EVENT	4	イベント数
mtr_calc_encd_foc.h	MTR_ENCD_CPR_MECH	1200	エンコーダカウントクロップ値
	MTR_ENCD_POLE_PAIRS	MP_POLE_PAIRS	極対数補正用定数
	MTR_ENCD_STATE_ALIGN	0	位置調整
	MTR_ENCD_STATE_INIT	1	初期位置移動
	MTR_ENCD_STATE_CURRENT	2	電流制御
	MTR_ENCD_STATE_SPEED	3	回転速度制御
	MTR_ENCD_STATE_POSITION	4	位置制御
	MTR_ENCD_STATE_DEFAULT	MTR_ENCD_STATE_POSITION	初期状態
	MTR_COEF_ENCD_THETA	3.413333333f	エンコーダカウント→位置変換係数
	MTR_COEF_ENCD_OMEGA	5.235987756f	エンコーダカウント(Δ)→回転速度変換係数
	MTR_COEF_LPF1	0.7f	速度 LPF 係数 1
	MTR_COEF_LPF2	0.3f	速度 LPF 係数 2
	MTR_COEF_AMP	20.01955034f	シャント抵抗電圧→AMP 出力電圧変換係数
	MTR_COEF_G	0.1f	静止座標電流ゲイン
	MTR_COEF_K	2216.684724f	A/V 変換係数
	MTR_CENTER_AMP	10240	AMP 出力電圧センタ値
	MTR_CENTER_ADC	512	A/D 変換センタ値(シャント抵抗電圧)
	MTR_CENTER_PWM_DUTY	50	PWM Duty センタ値
	MTR_FIXED_THETA	585	位置固定値
	MTR_FIXED_OMEGA	0	回転速度固定値

	MTR_FIXED_ID	20480	d 軸電流固定値
	MTR_FIXED_IQ	0	q 軸電流固定値
	MTR_LIMIT_ADC	1023U	A/D 変換リミット値(シャント抵抗電圧)
	MTR_LIMIT_THETA	4095U	位置リミット値
	MTR_SRL_THETA	10U	位置 SRL 値
	MTR_SRL_OMEGA	10U	速度 SRL 値
	MTR_PID_LIMIT_ZI	4000	PID 遅延素子リミット値
	MTR_PID_THETA_TH	100	位置制御用 PID パラメータ切り替え閾値
	MTR_PID_THETA_KP	100.0f	位置 PID 制御比例ゲイン
	MTR_PID_THETA_TI	0.0005f	位置 PID 制御積分ゲイン(サンプリング)
	MTR_PID_THETA_KI	0.3f	位置 PID 制御積分ゲイン
	MTR_PID_THETA_TD	1000.0f	位置 PID 制御微分ゲイン(サンプリング)
	MTR_PID_THETA_KD	0.001f	位置 PID 制御微分ゲイン
	MTR_PID_THETA_KT	0.004f	位置 PID 制御出力ゲイン
	MTR_PID_THETA_KP_H	160.0f	位置 PID 制御比例ゲイン(静止用高ゲイン設定)
	MTR_PID_THETA_KT_H	0.005f	位置 PID 制御出力ゲイン(静止用高ゲイン設定)
	MTR_PID_OMEGA_KP	1.0f	速度 PID 制御比例ゲイン
	MTR_PID_OMEGA_TI	0.0005f	速度 PID 制御積分ゲイン(サンプリング)
	MTR_PID_OMEGA_KI	0.01f	速度 PID 制御積分ゲイン
	MTR_PID_OMEGA_TD	1000.0f	速度 PID 制御微分ゲイン(サンプリング)
	MTR_PID_OMEGA_KD	0.001f	速度 PID 制御微分ゲイン
	MTR_PID_OMEGA_KT	0.075f	速度 PID 制御出力ゲイン
	MTR_PID_OMEGA_KT_H	0.1f	速度 PID 制御出力ゲイン (静止用高ゲイン設定)
	MTR_PID_ID_KP	0.5f	d 軸電流 PID 制御比例ゲイン
	MTR_PID_ID_TI	0.0005f	d 軸電流 PID 制御積分ゲイン(サンプリング)
	MTR_PID_ID_KI	0.0f	d 軸電流 PID 制御積分ゲイン
	MTR_PID_ID_TD	1000.0f	d 軸電流 PID 制御微分ゲイン(サンプリング)
	MTR_PID_ID_KD	0.0f	d 軸電流 PID 制御微分ゲイン

	MTR_PID_ID_KT	0.26f	d 軸電流 PID 制御出力ゲイン
	MTR_PID_IQ_KP	1000.0f	q 軸電流 PID 制御比例ゲイン
	MTR_PID_IQ_TI	0.0005f	q 軸電流 PID 制御積分ゲイン(サンプリング)
	MTR_PID_IQ_KI	0.1f	q 軸電流 PID 制御積分ゲイン
	MTR_PID_IQ_TD	1000.0f	q 軸電流 PID 制御微分ゲイン(サンプリング)
	MTR_PID_IQ_KD	0.0f	q 軸電流 PID 制御微分ゲイン
	MTR_PID_IQ_KT	0.0013f	q 軸電流 PID 制御出力ゲイン
version.h	FW_REVISION	102	F/W リビジョン情報

ファイル名	マクロ名	内容	備考
r_cg_userdefine.h	SPI00_CS_H	(P0 = P0 | 0x20)	SPI 通信 Chip Select 信号 = H
	SPI00_CS_L	(P0 = P0 & 0xDF)	SPI 通信 Chip Select 信号 = L
	SPI_WAIT_MODE	0x01	SPI 通信 Wait モード
	SPI_INTR_MODE	0x02	SPI 通信 Interrupt モード
	SPI_ADC_MODE	0x03	SPI 通信 ADC モード

ファイル名	マクロ名	内容	備考
r_cg_predrv.h	REG_BUFF_SIZE	2	プリドライバレジスタバッファサイズ
	SPI_CHK_MAX	100	プリドライバ SPI 通信チェック回数
	PREDRV_NORMAL	0	プリドライバシーケンスなし
	PREDRV_SPI_ERROR	1	プリドライバシーケンス SPI 通信エラー
	PREDRV_ALARM_ERROR	2	プリドライバシーケンス ALARM エラー
	PREDRV_REGRW_ERROR	4	プリドライバシーケンス レジスタ R/W エラー
	PREDRV_SPI_ACCESS_OK	0x6A	プリドライバ SPI 通信判定
	PREDRV_ALMRAW1_OK	0xEF	プリドライバ ALMRAW1 判定
	ALMSTS1_TSD_N	0x01	ALARM ステータス 1 判定
	ALMSTS1_OCP_N	0x02	
	ALMSTS1_VGB_UVP_N	0x04	
	ALMSTS1_VGB_OVP_N	0x08	
	ALMSTS1_VGT_UVP_N	0x10	
	ALMSTS1_VGT_OVP2_N	0x20	
	ALMSTS1_VGT_OVP1_N	0x40	
	ALMSTS1_VREG5_OVP_N	0x80	
	ALMSTS1_NO_ERROR	0xEF	
	ALMSTS1_VGT_UVP_MASK	0xEF	
	ALMSTS2_VM_UVP_N	0x01	ALARM ステータス 2 判定
	ALMSTS2_DI_SEL_W_CMP_N	0x20	
	ALMSTS2_DI_SEL_V_CMP_N	0x40	
	ALMSTS2_DI_SEL_U_CMP_N	0x80	
	ALMSTS2_NO_ERROR	0xFF	
	WHO_AM_I_MASK	0xFE	WHO_AM_I マスク
	INIT_PS_ALL	0x01	PS_ALL 初期値
	INIT_PS_1ST	0x3A	PS 初期値 1st
	INIT_PS_2ND	0x3B	PS 初期値 2nd
	INIT_PS_3RD	0xBB	PS 初期値 3rd
	INIT_SELSIG_U	0x03	SELSIG_U 初期値
	INIT_SELSIG_V	0x14	SELSIG_V 初期値
	INIT_SELSIG_W	0x25	SELSIG_W 初期値
	INIT_HALL_SIG	0x00	HALL_SIG 初期値
	INIT_ALMOPE1	0x10	ALMOPE1 初期値
	INIT_ALMOUT1	0x10	ALMOUT1 初期値
	INIT_CS_SET2	0x60	CS_SET2 初期値
	INIT_ERROR_WAIT	0x00	ERROR_WAIT 初期値
	INIT_CS_SET1	0x08	CS_SET1 初期値
	INIT_HAIC_TH	0x00	HAIC_TH 初期値
	INIT_LD_WAIT	0x00	LD_WAIT 初期値
	INIT_DRIVE_SET	0x01	DRIVE_SET 初期値
	INIT_IDRCNT_H	0x00	IDRCNT_H 初期値

	INIT_IDRCNT_L	0x00	IDRCNT_L 初期値
	INIT_TRCNT_P	0x00	TRCNT_P 初期値
	INIT_CPSET1	0x01	CPSET1 初期値
	INIT_CPSET2	0x02	CPSET2 初期値
	IINIT_CP_TRIM	0x00	CP_TRIM 初期値
	INIT_VREG5_TRIM	0x20	VREG5_TRIM 初期値
	INIT_CSAMP_TRIM	0x20	CSAMP_TRIM 初期値
	INIT_TRIM_PT	0x00	TRIM_PT 初期値 プロテクト設定
	INIT_TRIM_PT_UP	0x95	TRIM_PT 初期値 プロテクト解除
	INIT_TRIM_EN	0x00	TRIM_EN 初期値
	INIT_TRIM_EN_EFWD	0x01	TRIM_EN 初期値トリミングデータ有効設定値
	INIT_BGR_TRIM	0x00	BGR_TRIM 初期値
	INIT_BFAMP_TRIM	0x00	BFAMP_TRIM 初期値
	ERRRCV_PS_1ST	0x38	PS ALARM 復帰値 1st
	ERRRCV_PS_2ND	0x3A	PS ALARM 復帰値 2nd
	ERRRCV_PS_3RD	0x3B	PS ALARM 復帰値 3rd
	ERRRCV_PS_4TH	0xBB	PS ALARM 復帰値 4th
	ERRRCV_MOT_EN_CLR	0x00	DRIVE_SET ALARM 復帰値 モータ回転禁止
	ERRRCV_MOT_EN_SET	0x01	DRIVE_SET ALARM 復帰値 モータ回転許可
	ERRRCV_ALM_LATCH_CLR	0x40	DRIVE_SET ALARM 復帰値 ALARM ラッチクリア
	WAITTIME_1_MS	0x11F8	1[ms]待ち
	WAITTIME_3_MS	0x35E8	3[ms]待ち
	INIT_ICS_PS_ALL	INIT_PS_ALL	PS_ALL ICS 変数初期値
	INIT_ICS_PS	INIT_PS_3RD	PS ICS 変数初期値
	INIT_ICS_SW_RESET	0x00	SW_RESET ICS 変数初期値
	INIT_ICS_ADC_SEL	0x00	ADC_SEL ICS 変数初期値
	INIT_ICS_SELSIG_U	0x03	SELSIG_U ICS 変数初期値
	INIT_ICS_SELSIG_V	0x14	SELSIG_V ICS 変数初期値
	INIT_ICS_SELSIG_W	0x25	SELSIG_W ICS 変数初期値
	INIT_ICS_HALL_SIG	0x00	HALL_SIG ICS 変数初期値
	INIT_ICS_ALMSTS1	0xFF	ALMSTS1 ICS 変数初期値
	INIT_ICS_ALMOPE1	INIT_ALMOPE1	ALMOPE1 ICS 変数初期値

	INIT_ICS_ALMOUT1	INIT_ALMOUT1	ALMOUT1 ICS 変数初期値
	INIT_ICS_ALMSTS2	0xFF	ALMSTS2 ICS 変数初期値
	INIT_ICS_CS_SET2	INIT_CS_SET2	CS_SET2 ICS 変数初期値
	INIT_ICS_ALMOUT2	0x00	ALMOUT2 ICS 変数初期値
	INIT_ICS_ERROR_WAIT	0x00	ERROR_WAIT ICS 変数初期値
	INIT_ICS_CS_SET1	INIT_CS_SET1	CS_SET1 ICS 変数初期値
	INIT_ICS_HAIC_TH	0x00	HAIC_TH ICS 変数初期値
	INIT_ICS_PDDSTS	0xF0	PDDSTS ICS 変数初期値
	INIT_ICS_LD_WAIT	0x00	LD_WAIT ICS 変数初期値
	INIT_ICS_DRIVE_SET	INIT_DRIVE_SET	DRIVE_SET ICS 変数初期値
	INIT_ICS_DI_TIME	0x00	DI_TIME ICS 変数初期値
	INIT_ICS_IDRCNT_H	0x00	IDRCNT_H ICS 変数初期値
	INIT_ICS_IDRCNT_L	0x00	IDRCNT_L ICS 変数初期値
	INIT_ICS_TRCNT_P	0x00	TRCNT_P ICS 変数初期値
	INIT_ICS_CPSET1	0x01	CPSET1 ICS 変数初期値
	INIT_ICS_CPSET2	0x02	CPSET2 ICS 変数初期値
	INIT_ICS_CP_TRIM	INIT_CP_TRIM	CP_TRIM ICS 変数初期値
	INIT_ICS_VREG5_TRIM	INIT_VREG5_TRIM	VREG5_TRIM ICS 変数初期値
	INIT_ICS_CSAMP_TRIM	INIT_CSAMP_TRIM	CSAMP_TRIM ICS 変数初期値
	INIT_ICS_ALMRAW1	0xFF	ALMRAW1 ICS 変数初期値
	INIT_ICS_TOIN_MONI	0x00	TOIN_MONI ICS 変数初期値
	INIT_ICS_WHO_AM_I	0x6A	WHO_AM_I ICS 変数初期値
	INIT_ICS_TRIM_PT	INIT_TRIM_PT	TRIM_PT ICS 変数初期値
	INIT_ICS_TRIM_EN	INIT_TRIM_EN	TRIM_EN ICS 変数初期値
	INIT_ICS_BGR_TRIM	INIT_BGR_TRIM	BGR_TRIM ICS 変数初期値
	INIT_ICS_BFAMP_TRIM	INIT_BFAMP_TRIM	BFAMP_TRIM ICS 変数初期値
	SEQ_INIT	0	プリドライバ初期化シーケンス定義
	SEQ_CHK_SPI	1	
	SEQ_CHK_TSD_N	2	

	SEQ_SET_5VTRIM	3	
	SEQ_SET_ALMOPE1_PRM	4	
	SEQ_SET_ALMOUT1_PRM	5	
	SEQ_SET_CS_SET2_PRM	6	
	SEQ_SET_CS_SET1_PRM	7	
	SEQ_SET_SEQINIT_PRM	8	
	SEQ_SET_PS_ALL_PRM	9	
	SEQ_SET_PS_1ST_PRM	10	
	SEQ_SET_PS_2ND_PRM	11	
	SEQ_CHK_ALMRAW1	12	
	SEQ_SET_PS_3RD_PRM	13	
	SEQ_CHK_ALMSTS	14	
	SEQ_SET_MOT_EN	15	
	SEQ_END	16	
	SEQ_NUM_MAX	17	
	ERR_RCV_SEQ_INIT	0	プリドライバ ALARM 復帰シーケンス定義
	ERR_RCV_SEQ_CHK_STS	1	
	ERR_RCV_SEQ_CLR_MOT_EN	2	
	ERR_RCV_SEQ_SET_PS_1ST	3	
	ERR_RCV_SEQ_CHK_ALMSTS_1ST	4	
	ERR_RCV_SEQ_SET_ALM_LATCH_CLR	5	
	ERR_RCV_SEQ_SET_PS_2ND	6	
	ERR_RCV_SEQ_SET_PS_3RD	7	
	ERR_RCV_SEQ_CHK_ALMRAW1	8	
	ERR_RCV_SEQ_SET_PS_4TH	9	
	ERR_RCV_SEQ_CHK_ALMSTS_2ND	10	
	ERR_RCV_SEQ_SET_MOT_EN	11	
	ERR_RCV_SEQ_END	12	
	ERR_RCV_SEQ_NUM_MAX	13	

3.5 フローチャート

Figure 3-4 に全体フローチャートと初期化関数のフローチャートを示す。

また，サンプルプログラムの主要な処理のフローチャートを Figure 3-4 から Figure 3-11 に示す。

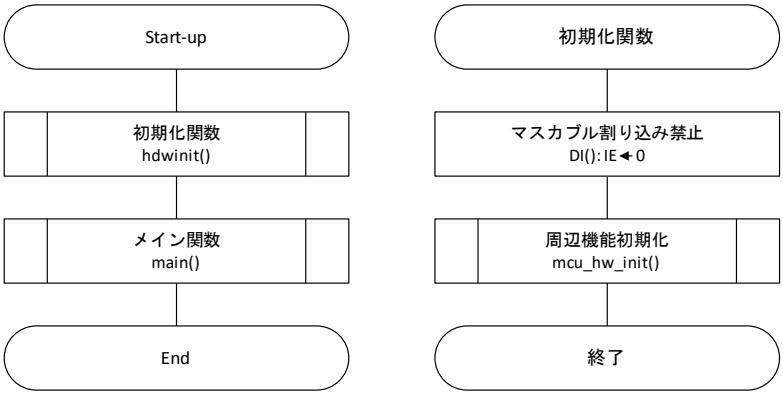

Figure 3-4 フローチャート(全体、初期化関数)

3.5.1 メイン関数

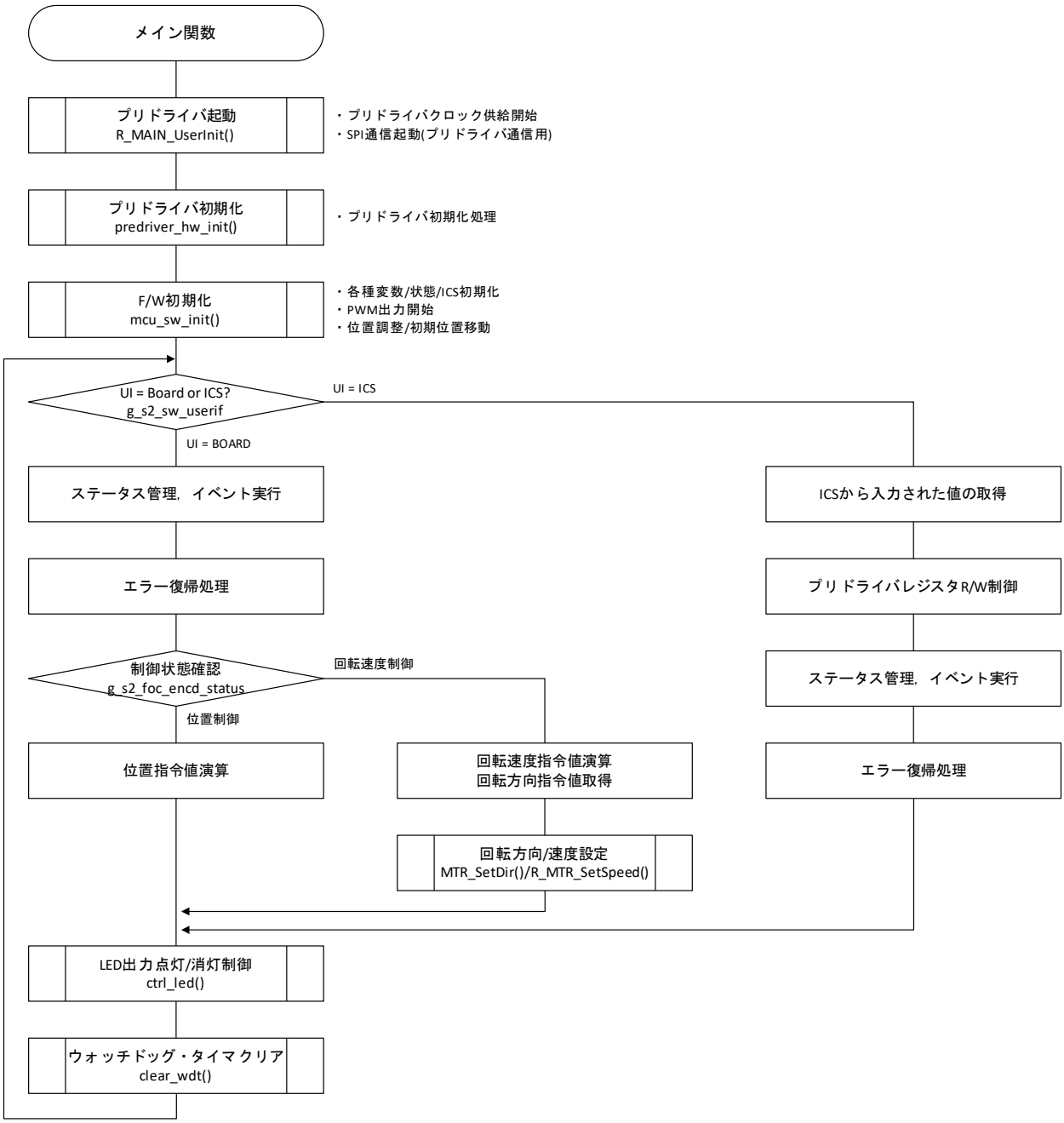

Figure 3-5 フローチャート(メイン関数)

3.5.2 プリドライバ初期化処理

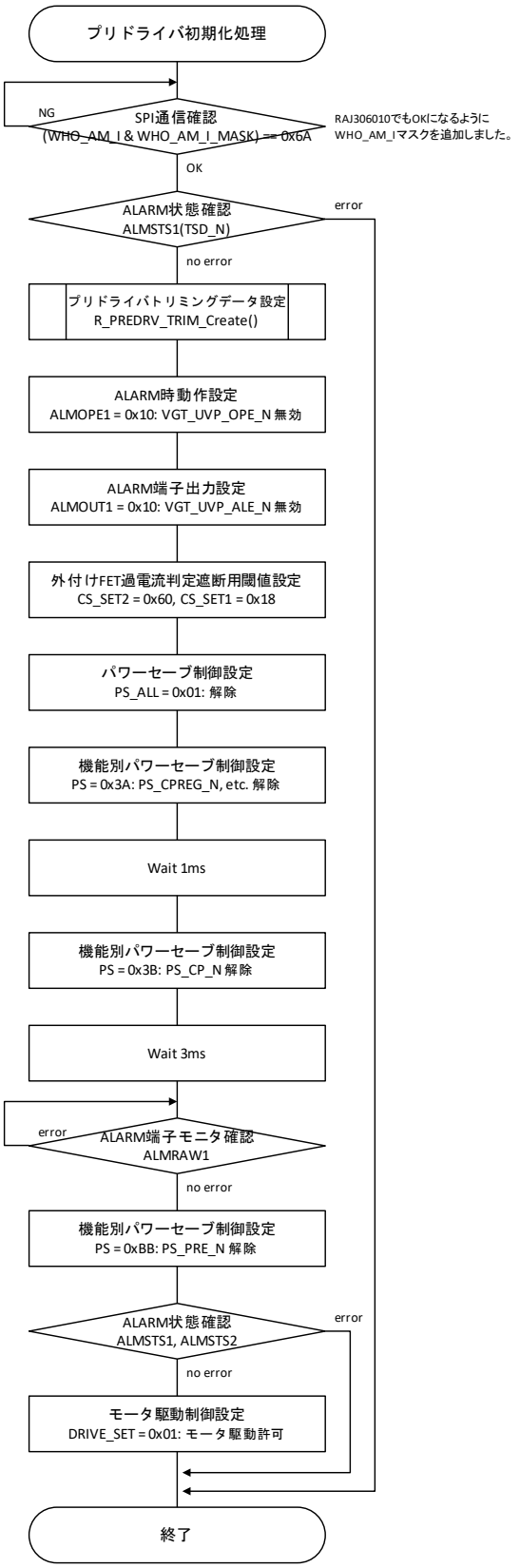

Figure 3-6 フローチャート(プリドライバ初期化処理)

3.5.3 キャリア周波数割り込み処理

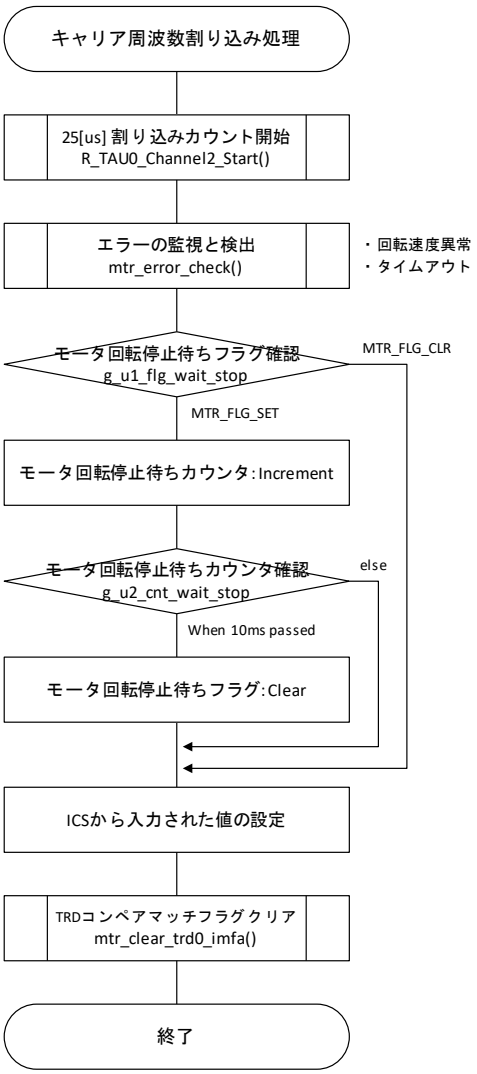

Figure 3-7 フローチャート(キャリア周波数割り込み処理)

3.5.4 1[ms] 割り込み処理

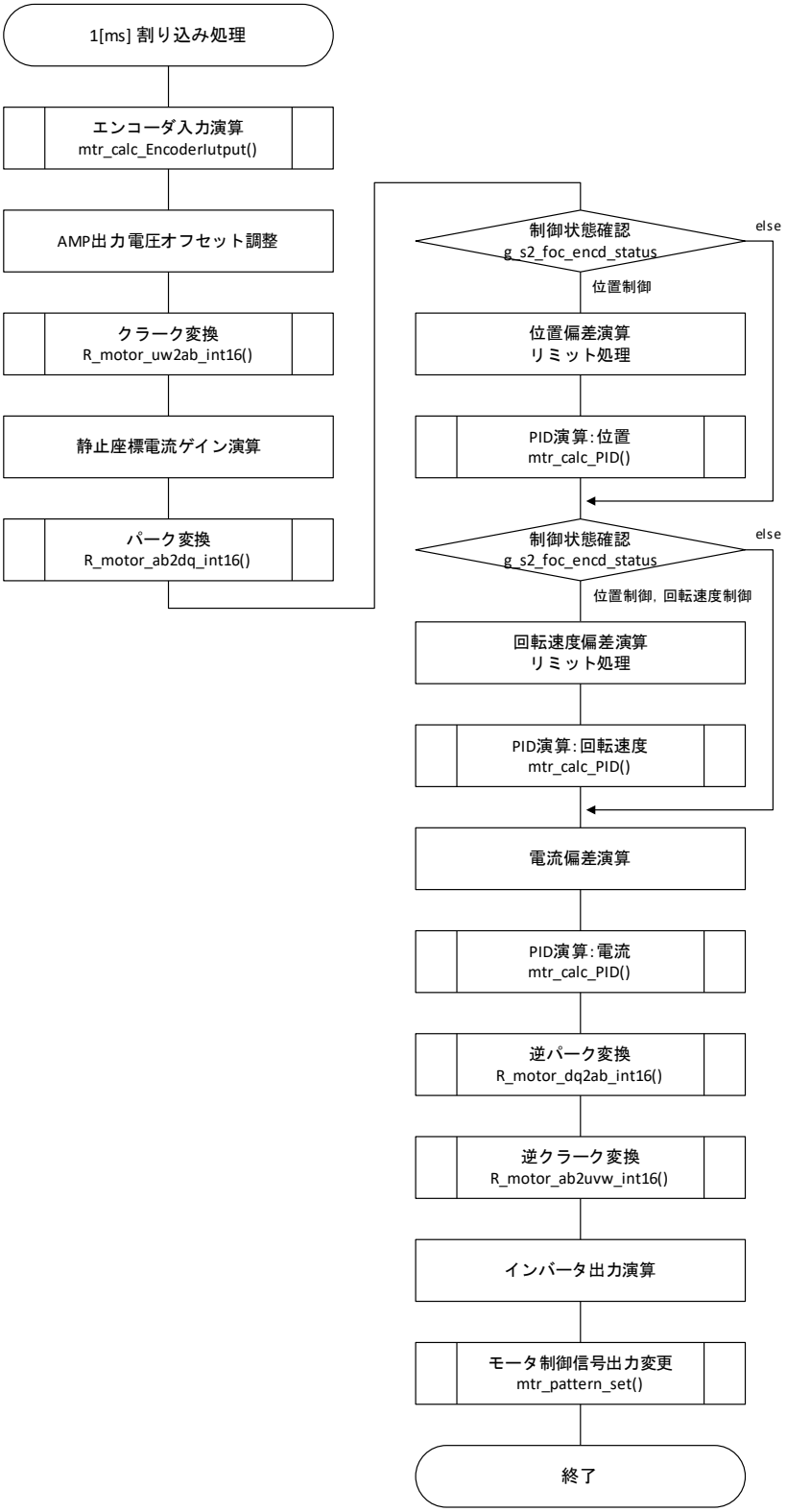

Figure 3-8 フローチャート(1[ms] 割り込み処理)

3.5.5 25[us] 割り込み処理

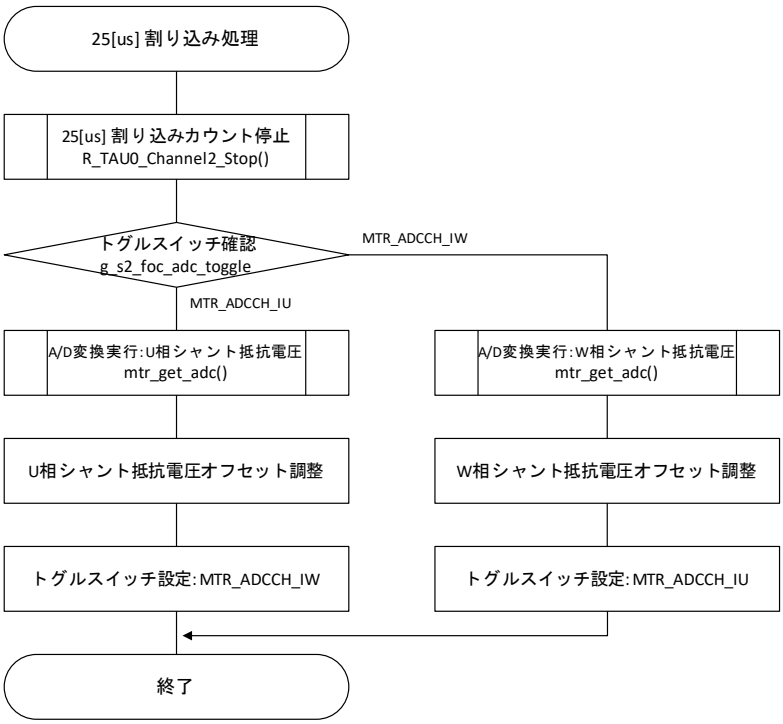

Figure 3-9 フローチャート(25[us] 割り込み処理)

3.5.6 ALARM 割り込み処理

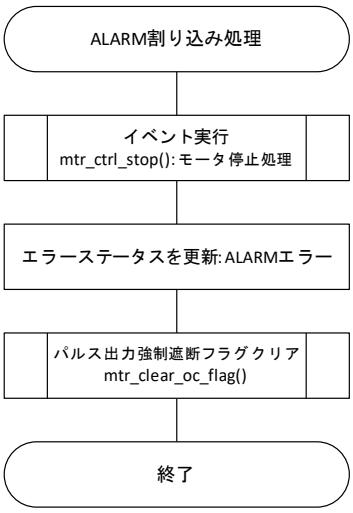

Figure 3-10 フローチャート(ALARM 割り込み処理)

3.5.7 ALARM 復帰処理

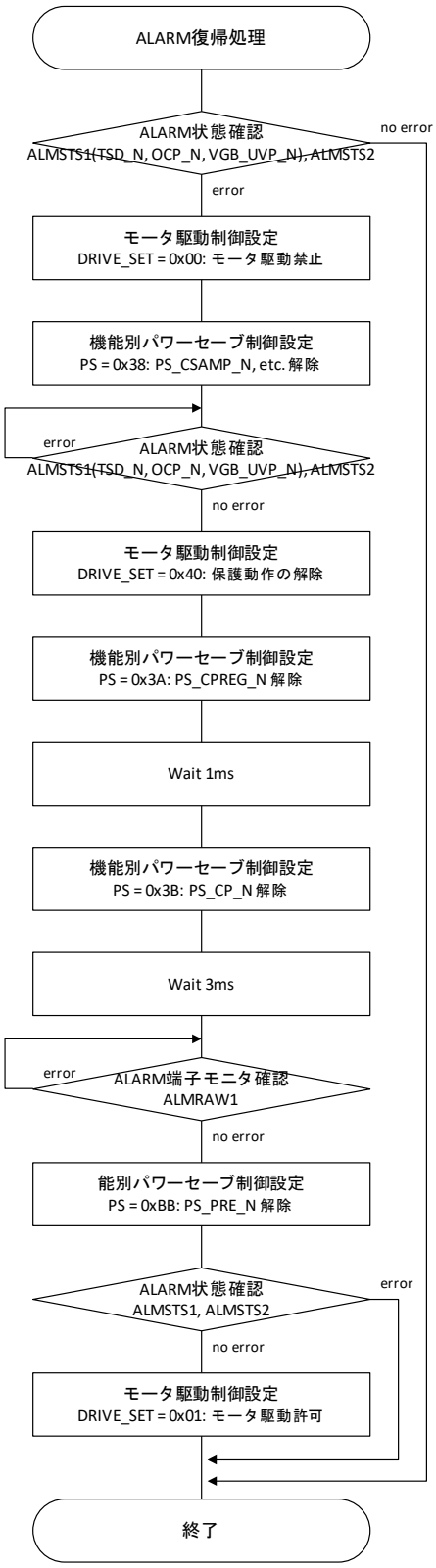

Figure 3-11 フローチャート(ALARM 復帰処理)

4. 開発支援ツール「In Circuit Scope」

4.1 概要

本アプリケーションノート対象サンプルプログラムでは、ICS によるユーザインタフェース(位置指令、回転速度指令等)が使用可能です。ICS はターゲットシステム上で実行されるプログラムのグローバル変数値をリアルタイムにパソコン上に波形表示することができるツールです。使用方法などの詳細は「In Circuit Scope 取扱説明書」を参照して下さい。

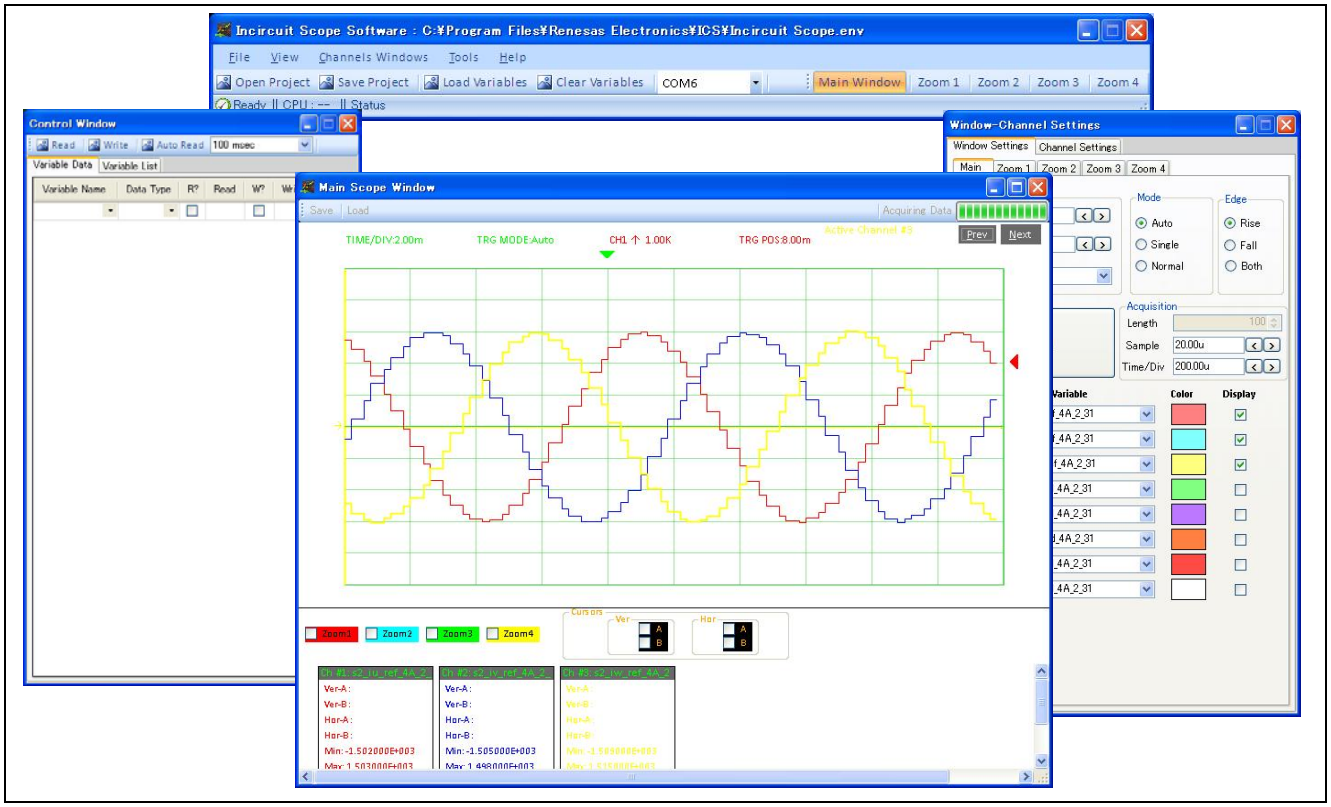

Figure 4-1 ICS 外観

4.2 ライブラリ使用方法

ICS を使用する場合、ICS に関連する関数を呼び出す必要があります。ICS 関連関数は条件コンパイル(#ifdef--#endif)で設定しています。ICS を使用する場合は下記の通り設定してください。

【ファイル名】mtr_common.h

【変更箇所】下記の宣言を加えてください。

#define ICS_USE

4.3 ICS 用変数一覧

ICS 用変数の一覧を Table 4-1, Table 4-2 に示す。Table 4-1 の変数の値は com_s2_enable_write に g_s2_enable_write と同じ値を書込んだ場合にプロテクト変数へ反映される。また、Table 4-2 の変数は com_s2_enable_write に依存しない。

Table 4-1 ICS 用変数一覧

変数名	型	内容	備考（【】:プロテクト変数名）
com_s2_enable_write	int16_t	変数書き換え許可	-
com_s2_foc_ref_theta	int16_t	位置指令値(機械角) [rad] ※ 正規化	【g_s2_foc_ref_theta】
com_s2_foc_ref_omega	int16_t	回転速度指令値(機械角) [rad/s]	【g_s2_foc_ref_omega】

Table 4-2 ICS 用変数一覧

変数名	型	内容	備考
com_s2_sw_userif	int16_t	ユーザインタフェーススイッチ 0：ICS ユーザインタフェース使用 1：ボードユーザインタフェース使用	-
com_s2_mode_system	int16_t	ステート管理 0：ストップモード 1：ランモード 3：リセット	-
com_u1_pdrvreg_ctrl	uint8_t	プリドライバレジスタ R/W コントロールフラグ	0: R/W 禁止 1: R/W 許可
com_u1_pdrvreg_ps_all_pre	uint8_t	プリドライバレジスタ PS_ALL 前回値	Read 値
com_u1_pdrvreg_ps_all_now	uint8_t	プリドライバレジスタ PS_ALL 現在値	Write 値
com_u1_pdrvreg_ps_pre	uint8_t	プリドライバレジスタ PS 前回値	Read 値
com_u1_pdrvreg_ps_now	uint8_t	プリドライバレジスタ PS 現在値	Write 値
com_u1_pdrvreg_sw_reset_pre	uint8_t	プリドライバレジスタ SW_RESET 前回値	Read 値
com_u1_pdrvreg_sw_reset_now	uint8_t	プリドライバレジスタ SW_RESET 現在値	Write 値
com_u1_pdrvreg_adc_sel_pre	uint8_t	プリドライバレジスタ ADC_SEL 前回値	Read 値
com_u1_pdrvreg_adc_sel_now	uint8_t	プリドライバレジスタ ADC_SEL 現在値	Write 値
com_u1_pdrvreg_selsig_u_pre	uint8_t	プリドライバレジスタ SELSIG_U 前回値	Read 値
com_u1_pdrvreg_selsig_u_now	uint8_t	プリドライバレジスタ SELSIG_U 現在値	Write 値
com_u1_pdrvreg_selsig_v_pre	uint8_t	プリドライバレジスタ SELSIG_V 前回値	Read 値
com_u1_pdrvreg_selsig_v_now	uint8_t	プリドライバレジスタ SELSIG_V 現在値	Write 値
com_u1_pdrvreg_selsig_w_pre	uint8_t	プリドライバレジスタ SELSIG_W 前回値	Read 値
com_u1_pdrvreg_selsig_w_now	uint8_t	プリドライバレジスタ SELSIG_W 現在値	Write 値
com_u1_pdrvreg_hall_sig_pre	uint8_t	プリドライバレジスタ HALL_SIG 前回値	Read 値
com_u1_pdrvreg_hall_sig_now	uint8_t	プリドライバレジスタ HALL_SIG 現在値	Write 値
com_u1_pdrvreg_almsts1_pre	uint8_t	プリドライバレジスタ ALMSTS1 前回値	Read 値 (ALMSTS1 Read Only)
com_u1_pdrvreg_almope1_pre	uint8_t	プリドライバレジスタ ALMOPE1 前回値	Read 値
com_u1_pdrvreg_almope1_now	uint8_t	プリドライバレジスタ ALMOPE1 現在値	Write 値
com_u1_pdrvreg_almout1_pre	uint8_t	プリドライバレジスタ ALMOUT1 前回値	Read 値
com_u1_pdrvreg_almout1_now	uint8_t	プリドライバレジスタ ALMOUT1 現在値	Write 値
com_u1_pdrvreg_almsts2_pre	uint8_t	プリドライバレジスタ ALMSTS2 前回値	Read 値 (ALMSTS2 Read Only)

com_u1_pdrvreg_cs_set2_pre	uint8_t	プリドライバレジスタ CS_SET2 前回値	Read 値
com_u1_pdrvreg_cs_set2_now	uint8_t	プリドライバレジスタ CS_SET2 現在値	Write 値
com_u1_pdrvreg_almout2_pre	uint8_t	プリドライバレジスタ ALMOUT2 前回値	Read 値
com_u1_pdrvreg_almout2_now	uint8_t	プリドライバレジスタ ALMOUT2 現在値	Write 値
com_u1_pdrvreg_error_wait_pre	uint8_t	プリドライバレジスタ ERROR_WAIT 前回値	Read 値
com_u1_pdrvreg_error_wait_now	uint8_t	プリドライバレジスタ ERROR_WAIT 現在値	Write 値
com_u1_pdrvreg_cs_set1_pre	uint8_t	プリドライバレジスタ CS_SET1 前回値	Read 値
com_u1_pdrvreg_cs_set1_now	uint8_t	プリドライバレジスタ CS_SET1 現在値	Write 値
com_u1_pdrvreg_haic_th_pre	uint8_t	プリドライバレジスタ HAIC_TH 前回値	Read 値
com_u1_pdrvreg_haic_th_now	uint8_t	プリドライバレジスタ HAIC_TH 現在値	Write 値
com_u1_pdrvreg_pddsts_pre	uint8_t	プリドライバレジスタ PDDSTS 前回値	Read 値 (PDDSTS Read Only)
com_u1_pdrvreg_ld_wait_pre	uint8_t	プリドライバレジスタ LD_WAIT 前回値	Read 値
com_u1_pdrvreg_ld_wait_now	uint8_t	プリドライバレジスタ LD_WAIT 現在値	Write 値
com_u1_pdrvreg_drive_set_pre	uint8_t	プリドライバレジスタ DRIVE_SET 前回値	Read 値
com_u1_pdrvreg_drive_set_now	uint8_t	プリドライバレジスタ DRIVE_SET 現在値	Write 値
com_u1_pdrvreg_di_time_pre	uint8_t	プリドライバレジスタ DI_TIME 前回値	Read 値
com_u1_pdrvreg_di_time_now	uint8_t	プリドライバレジスタ DI_TIME 現在値	Write 値
com_u1_pdrvreg_idrcnt_h_pre	uint8_t	プリドライバレジスタ IDRCNT_H 前回値	Read 値
com_u1_pdrvreg_idrcnt_h_now	uint8_t	プリドライバレジスタ IDRCNT_H 現在値	Write 値
com_u1_pdrvreg_idrcnt_l_pre	uint8_t	プリドライバレジスタ IDRCNT_L 前回値	Read 値
com_u1_pdrvreg_idrcnt_l_now	uint8_t	プリドライバレジスタ IDRCNT_L 現在値	Write 値
com_u1_pdrvreg_trcnt_p_pre	uint8_t	プリドライバレジスタ TRCNT_P 前回値	Read 値
com_u1_pdrvreg_trcnt_p_now	uint8_t	プリドライバレジスタ TRCNT_P 現在値	Write 値
com_u1_pdrvreg_cpset1_pre	uint8_t	プリドライバレジスタ CPSET1 前回値	Read 値
com_u1_pdrvreg_cpset1_now	uint8_t	プリドライバレジスタ CPSET1 現在値	Write 値
com_u1_pdrvreg_cpset2_pre	uint8_t	プリドライバレジスタ CPSET2 前回値	Read 値
com_u1_pdrvreg_cpset2_now	uint8_t	プリドライバレジスタ CPSET2 現在値	Write 値
com_u1_pdrvreg_cp_trim_pre	uint8_t	プリドライバレジスタ CP_TRIM 前回値	Read 値
com_u1_pdrvreg_cp_trim_now	uint8_t	プリドライバレジスタ CP_TRIM 現在値	Write 値
com_u1_pdrvreg_vreg5_trim_pre	uint8_t	プリドライバレジスタ VREG5_TRIM 前回値	Read 値
com_u1_pdrvreg_vreg5_trim_now	uint8_t	プリドライバレジスタ VREG5_TRIM 現在値	Write 値
com_u1_pdrvreg_csamp_trim_pre	uint8_t	プリドライバレジスタ CSAMP_TRIM 前回値	Read 値
com_u1_pdrvreg_csamp_trim_now	uint8_t	プリドライバレジスタ CSAMP_TRIM 現在値	Write 値
com_u1_pdrvreg_almraw1_pre	uint8_t	プリドライバレジスタ ALMRAW1 前回値	Read 値 (ALMRAW1 Read Only)
com_u1_pdrvreg_toin_moni_pre	uint8_t	プリドライバレジスタ TOIN_MONI 前回値	Read 値 (TOIN_MONI Read Only)
com_u1_pdrvreg_who_am_i_pre	uint8_t	プリドライバレジスタ WHO_AM_I 前回値	Read 値 (WHO_AM_I Read Only)
com_u1_pdrvreg_trim_pt_pre	uint8_t	プリドライバレジスタ TRIM_PT 前回値	Read 値
com_u1_pdrvreg_trim_pt_now	uint8_t	プリドライバレジスタ TRIM_PT 現在値	Write 値
com_u1_pdrvreg_trim_en_pre	uint8_t	プリドライバレジスタ TRIM_EN 前回値	Read 値
com_u1_pdrvreg_trim_en_now	uint8_t	プリドライバレジスタ TRIM_EN 現在値	Write 値

com_u1_pdrvreg_bgr_trim_pre	uint8_t	プリドライバレジスタ BGR_TRIM 前回値	Read 値
com_u1_pdrvreg_bgr_trim_now	uint8_t	プリドライバレジスタ BGR_TRIM 現在値	Write 値
com_u1_pdrvreg_bfamp_trim_pre	uint8_t	プリドライバレジスタ BFAMP_TRIM 前回値	Read 値
com_u1_pdrvreg_bfamp_trim_now	uint8_t	プリドライバレジスタ BFAMP_TRIM 現在値	Write 値

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Aug.23.18	-	新規発行
1.01	Mar.15.19	-	RAJ306000_ENCD_FOC_CLOSED_V101 誤記修正
1.02	Apr.14.20	-	RAJ306000_ENCD_FOC_CLOSED_*_*_V102 IDE: CS+ for CC, e^2 studio に対応 Table 1-1, 2-9, 10 ICS 用ライブラリを変更 Table 2-9: ics マクロ定義(WHO_AM_I_MASK)を追加 Figure 3-6, Table 3-9 誤記修正 Table 3-6, 9 Table 4-2: TRIM_PT, TRIM_EN, BGR_TRIM, BFAMP_TRIM, etc.

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。