RZ/T2M グループ

BiSS-C サンプルプログラム

要旨

本アプリケーションノートでは、RZ/T2M の Encoder I/F Configuration Library（以下 EC-Lib）を使用して、BiSS-C 仕様に準拠したエンコーダから情報を取得・表示するサンプルプログラムについて説明します。

本サンプルプログラムは、RZ/T2M Encoder I/F Configuration Data (BiSS) Ver.1.0 以降に対応しています。

プログラムの特徴を以下に示します。

- センサーモード、レジスタアクセス、連続レジスタアクセスに対応
- BiSS-C 仕様に準拠したエンコーダ (Hengstler 社製 AD36/1219AF.OCBEF, Wachendorff 社製 WDGF 58M)から、角度情報等を取得

対象デバイス

RZ/T2M

目次

1. 仕様

表 1.1 に使用する周辺機能と用途を、図 1-1 にサンプルコード実行時の動作環境を示します。

表 1.1　使用する周辺機能と用途

周辺機能	用途
BiSS-C Encoder Interface (BISS)	BiSS-C 仕様に準拠したエンコーダとの通信
割り込みコントローラ (ICU)	BiSS-C Encoder Interface 割り込み制御
汎用 PWM タイマ (GPT)	連続処理時の通信周期の生成、および、ELC に入力するイベント周期の生成
イベントリンクコントローラ (ELC)	GPT が出力するイベントと BiSS-C Encoder Interface をリンク
シリアル通信インタフェース (SCI) UART	SCI の調歩同期式 I/F を使用し、USB インタフェースによる COM ポート通信に使用 サンプルプログラムのコンソールインタフェース用

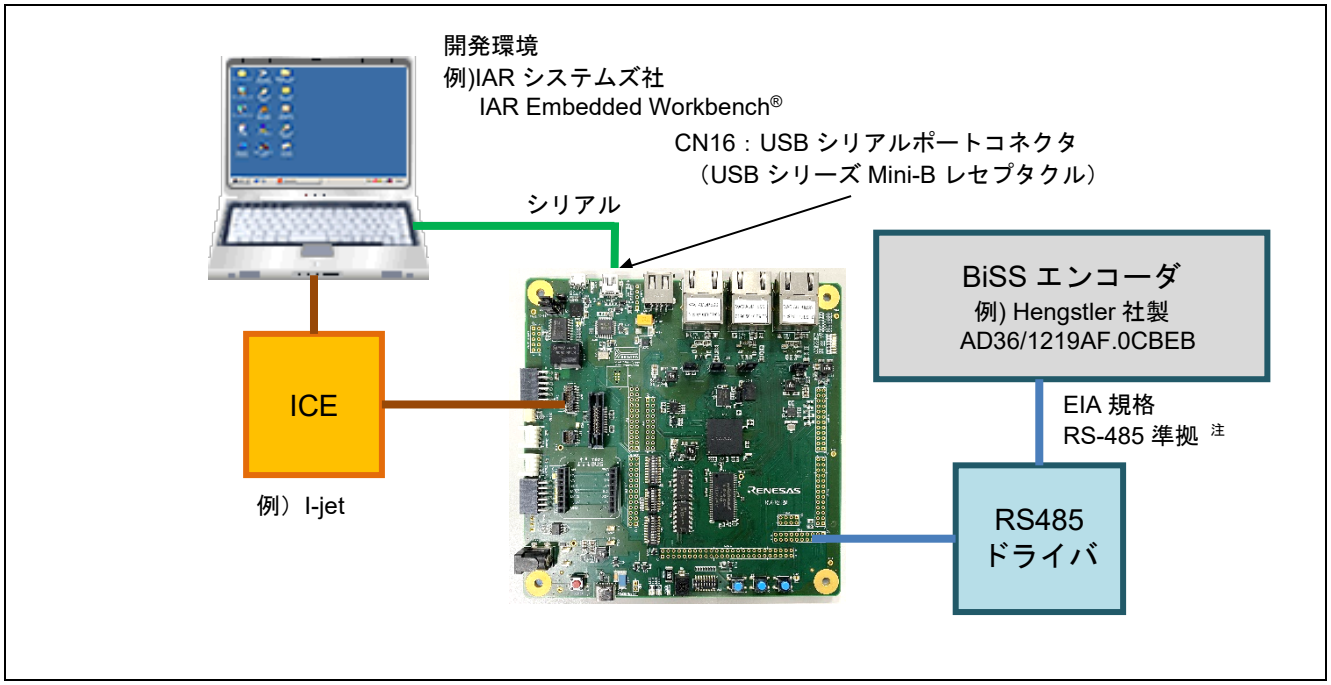


図 1-1　動作環境

【注】 送受信可能なケーブル長は、エンコーダのマニュアルを参照してください。

2. 動作環境

本アプリケーションノートのサンプルコードは、下記の環境を想定しています。

表 2.1　動作環境

項目	内容
使用マイコン	RZ/T2M グループ
動作周波数	CPUCLK = 800MHz
動作電圧	1.1V(Core) / 1.8V (PLL, etc.) / 3.3V (I/O)
統合開発環境 注	IAR システムズ製　IAR Embedded Workbench® for Arm® RENESAS 製　e² studio
使用ボード	RSK+RZT2M（RTK9RZT2M0C00000BE）
使用デバイス（ボード上で使用する機能）	なし

【注】 統合開発環境のバージョンは、RZ/T2M グループ Encoder I/F BiSS-C sample program リリースノートを参照してください。

3. 周辺機能説明

周辺機能、動作モード、レジスタについての基本的な内容は、RZ/T2M グループ・ユーザーズマニュアル ハードウェア編を参照してください。

3.1 使用端子一覧

表 3.1 に使用端子と機能を示します。

表 3.1 使用端子と機能

チャネル	端子名 (機能ピン名)	I/O ポート	入出力	内容
BISS0	SLO0 (ENCIF0)	P01_6	入力	センサ/コントロールデータ
	MA0 (ENCIF4)	P02_3	出力	クロック/コントロールデータ
BISS1	SLO1 (ENCIF5)	P17_3	入力	センサ/コントロールデータ
	MA1 (ENCIF9)	P03_3	出力	クロック/コントロールデータ

4. ソフトウェア説明

4.1 BiSS-C ドライバ機能

BiSS-C ドライバの機能は以下です。

1. BiSS-C Encoder Interface の初期設定
2. BiSS-C へのリクエスト送信、結果の受信
3. BiSS-C との通信時のエラー通知

4.2 ファイル構成

ファイル構成は、RZ/T2M グループ Encoder I/F BiSS-C sample program リリースノートを参照してください。

4.3 関数一覧

表 4.1 に関数を示します。

表 4.1 関数一覧

カテゴリ	関数名	ページ番号
BiSS API 関数	R_BISS_Open	7
	R_BISS_Close	7
	R_BISS_GetVersion	8
	R_BISS_Control	8
ユーザー定義関数	callback_get_pos	13
	callback_reg_access	13
	callback_elctimer_result	14
割り込みハンドラ	bissc0_rx_int_isr	14
	bissc1_rx_int_isr	14

4.4　API 関数仕様

4.4.1　R_BISS_Open

R_BISS_Open		
概　要	BiSS エンコーダ制御の開始	
ヘッダ	r_biss_rzt2_if.h	
宣　言	r_biss_err_t R_BISS_Open(const int32_t id, const r_biss_type_t type, r_biss_info_t* p_info);	
説　明	引数で指定された BiSS エンコーダの制御を開始します。	
引　数	id	：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。）
		R_BISS0_ID　：チャネル 0 を指定
		R_BISS1_ID　：チャネル 1 を指定
		上記以外　：設定不可
	type	エンコーダの種類を指定します。 R_BISS_TYPE_C を指定してください。
	p_info	エンコーダの情報を設定します。 エンコーダの情報を格納した構造体 r_biss_info_t のアドレスを指定してください。(構造体の詳細は「4.10.1(1)(a) r_biss_info_t」参照)
リターン値	R_BISS_SUCCESS	：正常終了
	R_BISS_ERR_INVALID_ARG	：異常終了 (引数 id, type が規定されていない値、 または p_info が NULL)
	R_BISS_ERR_ACCESS	：異常終了 (既に制御が開始されています)
注　意	本関数実行前に、必ず EC-Lib を用いて Multi-Protocol Encoder I/F のコンフィギュレーションと起動を行ってください。	

4.4.2　R_BISS_Close

R_BISS_Close		
概　要	BiSS エンコーダ制御を終了	
ヘッダ	r_biss_rzt2_if.h	
宣　言	r_biss_err_t R_BISS_Close(const int32_t id);	
説　明	指定されたチャネルの BiSS エンコーダの制御を終了します。	
引　数	id	：終了する ID を指定します。（r_biss_rzt2_dat.h で定義されています。）
		R_BISS0_ID　：チャネル 0 を指定
		R_BISS1_ID　：チャネル 1 を指定
		上記以外　：設定不可
リターン値	R_BISS_SUCCESS	：正常終了
	R_BISS_ERR_INVALID_ARG	：異常終了 （引数 id が規定されていない値）
	R_BISS_ERR_ACCESS	：異常終了（通信中または ELC 受付中です。）
注　意	エンコーダと送受信中の場合は Close できません。	

4.4.3　R_BISS_GetVersion

R_BISS_GetVersion

概　要	BiSS ドライバのバージョンを取得
ヘッダ	r_biss_rzt2_if.h
宣　言	uint32_t R_BISS_GetVersion(const r_biss_type_t type);
説　明	指定された BiSS ドライバのバージョンを取得します。
引　数	type　：ドライバの種類を指定します。 R_BISS_TYPE_CMN　：BiSS ドライバ共通部を指定 R_BISS_TYPE_C　：BiSS-C ドライバを指定
リターン値	上位 16 ビットにメジャーバージョン、下位 16 ビットにマイナーバージョンが格納されます。 例）戻り値が 0x00010002 の場合、Ver.1.2
補　足	上記以外の type が指定された場合、戻り値は 0xFFFFFFFF となります。

4.4.4　R_BISS_Control

R_BISS_Control

概　要	BiSS エンコーダの制御
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	引数 cmd を使って BiSS エンコーダを制御します。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：制御コマンド コマンド一覧は「表 4.5 BiSS-C ドライバで使用する主要な定数 (r_bissc_rzt2_if.h)」を参照してください。 p_buf　制御コマンドごとに必要な情報を格納してください。 格納する情報は、各制御コマンドの動作説明「4.4.4(1) R_BISS_TYPE_C 制御コマンド」に記載しています。
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了 （引数 id, cmd が規定されていない値、または cmd が Open している type に対応外の値） R_BISS_ERR_BUSY　：異常終了 （位置情報取得中またはレジスタアクセス中のため操作不可） R_BISS_ERR_ACCESS　：異常終了（Open されていません。）
注意	本関数実行前に、必ず R_BISS_Open を実行してください。

(1)　R_BISS_TYPE_C 制御コマンド

BiSS-C モードの、コマンド動作について記述します。

(a)　R_BISSC_CMD_POS

R_BISSC_CMD_POS	
概　要	BiSS エンコーダへの位置情報要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダの位置情報を要求します。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISSC_CMD_POS を指定します。 p_buf　：リクエスト情報（r_bissc_rzt2_if.h で定義されています。） リクエスト情報を格納した r_bissc_sensreq_t 構造体のポインタを指定します。詳細は「4.10.1(2)(a) r_bissc_sensreq_t」を参照してください。
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（p_buf が NULL） R_BISS_ERR_BUSY　：異常終了 （位置情報取得中またはレジスタアクセス中のため操作不可）

(b)　R_BISSC_CMD_REG_READ

R_BISSC_CMD_REG_READ	
概　要	BiSS エンコーダへのレジスタリード要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダのレジスタ値を読み込みます。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISSC_CMD_REG_READ を指定します。 p_buf　：リクエスト情報（r_bissc_rzt2_if.h で定義されています。） リクエスト情報を格納した r_bissc_regreq_t 構造体のポインタを指定します。詳細は「4.10.1(2)(b) r_bissc_regreq_t」を参照してください。
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（p_buf が NULL） R_BISS_ERR_BUSY　：異常終了 （位置情報取得中またはレジスタアクセス中のため操作不可）
注意	本関数実行後、データ送受信完了のコールバック（callback_get_pos）発生時には、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生するまでの間、R_BISSC_CMD_POS 要求を実行し続けてください。（「4.11.6(3) レジスタアクセス（リード・ライト）シーケンス」参照）

(c)　R_BISSC_CMD_REG_WRITE

R_BISSC_CMD_REG_WRITE	
概　要	BiSS エンコーダへのレジスタライト要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダのレジスタに指定データを書き込みます。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISSC_CMD_REG_WRITE を指定します。 p_buf　：リクエスト情報（r_bissc_rzt2_if.h で定義されています。） リクエスト情報を格納した r_bissc_regreq_t 構造体のポインタを指定します。詳細は「4.10.1(2)(b) r_bissc_regreq_t」を参照してください。
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（p_buf が NULL） R_BISS_ERR_BUSY　：異常終了 （位置情報取得中またはレジスタアクセス中のため操作不可）
注意	本関数実行後、データ送受信完了のコールバック（callback_get_pos）発生時には、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生するまでの間、R_BISSC_CMD_POS 要求を実行し続けてください。（「4.11.6(3) レジスタアクセス（リード・ライト）シーケンス」参照）

(d)　R_BISS_CMD_REG_SREAD

R_BISS_CMD_REG_SREAD	
概　要	BiSS エンコーダへの連続レジスタリード要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダのレジスタ値を連続で読み出します。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISS_CMD_REG_SREAD を指定します。 p_buf　：リクエスト情報（r_bissc_rzt2_if.h で定義されています。） リクエスト情報を格納した r_bissc_regreq_t 構造体のポインタを指定します。詳細は「4.10.1(2)(b) r_bissc_regreq_t」を参照してください。
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（p_buf が NULL） R_BISS_ERR_BUSY　：異常終了 （位置情報取得中またはレジスタアクセス中のため操作不可）
注意	本関数実行後、データ送受信完了のコールバック（callback_get_pos）発生時には、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生するまでの間、R_BISSC_CMD_POS 要求を実行し続けてください。（「4.11.6(4) 連続レジスタアクセス（リード・ライト）シーケンス」参照） 連続レジスタリード処理を中断する場合は、連続レジスタアクセス停止要求(R_BISSC_CMD_REG_SSTOP)を実行してください。

(e)　R_BISSC_CMD_REG_SWRITE

R_BISSC_CMD_REG_SWRITE	
概　要	BiSS エンコーダへの連続レジスタライト要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダのレジスタに指定データを連続で書き込みます。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISSC_CMD_REG_SWRITE を指定します。 p_buf　：リクエスト情報（r_bissc_rzt2_if.h で定義されています。） リクエスト情報を格納した r_bissc_regreq_t 構造体のポインタを指定します。詳細は「4.10.1(2)(b) r_bissc_regreq_t」を参照してください。
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（p_buf が NULL） R_BISS_ERR_BUSY　：異常終了 （位置情報取得中またはレジスタアクセス中のため操作不可）
注意	本関数実行後、データ送受信完了のコールバック（callback_get_pos）発生時には、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生するまでの間、R_BISSC_CMD_POS 要求を実行し続けてください。（「4.11.6(4) 連続レジスタアクセス（リード・ライト）シーケンス」参照） 連続レジスタリード処理を中断する場合は、連続レジスタアクセス停止要求(R_BISSC_CMD_REG_SSTOP)を実行してください。

(f)　R_BISSC_CMD_REG_SSTOP

R_BISSC_CMD_REG_SSTOP	
概　要	BiSS エンコーダへの連続レジスタアクセス停止要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダへの連続レジスタアクセスを停止します。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISSC_CMD_REG_SSTOP を指定します。 p_buf　：使用しません。(NULL を指定してください)
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（id が不正） R_BISS_ERR_ACCESS　：異常終了 （連続レジスタアクセス以外のため、操作不可）
注意	連続レジスタアクセス（リード・ライト）中のみ有効なコマンドです。 本関数実行後、データ送受信完了のコールバック（callback_get_pos）発生時には、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生するまでの間、R_BISSC_CMD_POS 要求を実行し続けてください。 連続レジスタアクセスの停止時に、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生し、停止までの間にリード・ライトしたアクセス結果情報が設定されます。（「4.11.6(6) 連続レジスタアクセス（リード・ライト）停止シーケンス」参照）

(g)　R_BISSC_CMD_REG_FSTOP

R_BISSC_CMD_REG_FSTOP	
概　要	BiSS エンコーダへのレジスタアクセス・連続レジスタアクセス停止要求
ヘッダ	r_biss_rzt2_if.h
宣　言	r_biss_err_t R_BISS_Control(const int32_t id, const r_biss_cmd_t cmd, void *const p_buf);
説　明	BiSS エンコーダへのレジスタアクセス・連続レジスタアクセスを停止します。
引　数	id　：使用する ID を指定します。（r_biss_rzt2_dat.h で定義されています。） R_BISS0_ID　：チャネル 0 を指定 R_BISS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_BISSC_CMD_REG_FSTOP を指定します。 p_buf　：使用しません。(NULL を指定してください)
リターン値	R_BISS_SUCCESS　：正常終了 R_BISS_ERR_INVALID_ARG　：異常終了（id が不正）
注意	レジスタアクセス・連続レジスタアクセスを強制停止するコマンドです。 レジスタアクセス・連続レジスタアクセスを実行時に、エンコーダからレジスタアクセス完了のステータスが返ってこない場合に使用してください。 本コマンド実行後に位置情報取得コマンド（R_BISSC_CMD_POS）を実行することで、レジスタアクセスを停止します。 タイミングによっては、本関数実行後、レジスタアクセス完了を示すコールバック（callback_reg_access）が発生する場合があります。

4.5 ユーザー定義関数仕様

4.5.1 callback_get_pos

callback_get_pos	
概　要	BiSS-C エンコーダからの位置情報取得結果を通知
ヘッダ	-
宣　言	static void callback_get_pos(void *p_result);
説　明	R_BISS_Control()関数で登録したコールバック関数です。位置情報取得結果を通知します。 本関数は割り込みハンドラのコンテキストとなります。割り込みの応答性を確保するため、速やかに return するようにしてください。関数名は例であり、自由に設定できます。
引　数	p_result　：位置情報取得結果を格納した r_bissc_sensordt_t 構造体へのポインタ。「4.10.1(2)(c) r_bissc_sensordt_t」参照。（r_bissc_rzt2_if.h で定義されています。） 本構造体は、次に R_BISSC_CMD_REG_SSTOP, R_BISSC_CMD_REG_FSTOP 以外のコマンドで R_BISS_Control()関数を実行するまで参照可能です。
リターン値	なし

4.5.2 callback_reg_access

callback_reg_access	
概　要	BiSS-C エンコーダからのレジスタリード・ライト結果を通知
ヘッダ	-
宣　言	static void callback_reg_access(void *p_result);
説　明	R_BISS_Control()関数で登録したコールバック関数です。エンコーダのレジスタアクセス要求に対する結果を通知します。 本関数は割り込みハンドラのコンテキストとなります。割り込みの応答性を確保するため、速やかに return するようにしてください。関数名は例であり、自由に設定できます。
引　数	p_result　：レジスタアクセス結果を格納した r_bissc_regdata_t 構造体へのポインタ。「4.10.1(2)(d) r_bissc_regdata_t」参照。（r_bissc_rzt2_if.h で定義されています。） 本構造体は、次にレジスタアクセスコマンド (R_BISSC_CMD_REG_READ / R_BISSC_CMD_REG_WRITE / R_BISSC_CMD_REG_SREAD / R_BISSC_CMD_REG_SWRITE)を実行するまで参照可能です。
リターン値	なし
注意	レジスタアクセス / 連続レジスタアクセス中にエラーが発生した時、レジスタアクセスの途中でも本コールバックによりエラーを通知します。 送受信結果が R_BISSC_REQ_ERR の場合は、エラーが発生したため、レジスタアクセス処理を終了します。 送受信結果が R_BISSC_REQ_ERR_WR の場合は、エラー発生によってレジスタに不正な値を書き込んだ可能性があります。再度、レジスタに正しい値を書き込んでください。（「4.11.6(5) 連続レジスタアクセス（リード・ライト）シーケンス（途中でエラー発生）」参照）

4.5.3 callback_elctimer_result

callback_elctimer_result	
概　要	BiSS-C エンコーダからの位置情報取得結果を通知 (ELC タイマモード)
ヘッダ	-
宣　言	static void callback_elctimer_result(void *p_result);
説　明	R_BISS_Control()関数で登録したコールバック関数です。ELC タイマモードで動作中に、位置情報取得結果を通知します。 本関数は割り込みハンドラのコンテキストとなります。割り込みの応答性を確保するため、速やかに return するようにしてください。関数名は例であり、自由に設定できます。
引　数	p_result　：位置情報取得結果を格納した r_bissc_sensordt_t 構造体へのポインタ。「4.10.1(2)(c) r_bissc_sensordt_t」参照。（r_bissc_rzt2_if.h で定義されています。） 本構造体は、次に R_BISSC_CMD_REG_SSTOP, R_BISSC_CMD_REG_FSTOP 以外のコマンドで R_BISS_Control()関数を実行するまで参照可能です。
リターン値	なし

4.6 割り込みハンドラ

4.6.1 bissc0_rx_int_isr

bissc0_rx_int_isr	
概　要	チャネル 0 の BiSS-C データ送受信完了割り込みハンドラ
ヘッダ	-
宣　言	void bissc0_rx_int_isr(void);
説　明	BiSS-C エンコーダからのデータ送受信 / レジスタアクセス完了 / エラー受信割り込みに対する割り込みハンドラです。割り込みレジスタの値を読み込み、状態を確認します。
引　数	なし
リターン値	なし

4.6.2 bissc1_rx_int_isr

bissc1_rx_int_isr	
概　要	チャネル 1 の BiSS-C データ送受信完了割り込みハンドラ
ヘッダ	-
宣　言	void bissc1_rx_int_isr(void);
説　明	BiSS-C エンコーダからのデータ送受信 / レジスタアクセス完了 / エラー受信割り込みに対する割り込みハンドラです。割り込みレジスタの値を読み込み、状態を確認します。
引　数	なし
リターン値	なし

4.7 使用割り込み一覧

表 4.2 に BiSS-C ドライバで使用する割り込みを示します。

表 4.2 BiSS-C ドライバで使用する割り込み

割り込み	ID	概要
ENCIF_INT0	372	チャネル 0 の通信ステータスにより割り込みが発生します。
ENCIF_INT4	376	チャネル 1 の通信ステータスにより割り込みが発生します。

4.8 定数／エラーコード一覧

表 4.3 に BiSS ドライバで使用する主要な定数 (r_biss_rzt2_if.h) を示します。表 4.4 に BiSS ドライバで使用する主要な定数 (r_biss_rzt2_dat.h)、表 4.5 に BiSS-C ドライバで使用する主要な定数 (r_bissc_rzt2_if.h)、表 4.6 に BiSS-C ドライバで使用する主要な定数 (r_bissc_rzt2_config.h)を示します。

表 4.3 BiSS ドライバで使用する主要な定数 (r_biss_rzt2_if.h)

定数名	設定値	内容	
R_BISS_TYPE_CMN	0	エンコーダまたは ドライバの種類	BiSS 共通を選択 (R_BISS_GetVersion 関数専用)
R_BISS_TYPE_C	2		BiSS-C を選択
R_BISS_SUCCESS	0	BiSS ドライバの エラーコード	正常終了
R_BISS_ERR_INVALID_ARG	-1		引数異常
R_BISS_ERR_BUSY	-2		API を実行できない状態
R_BISS_ERR_ACCESS	-3		API の実行順序エラー

表 4.4 BiSS ドライバで使用する主要な定数 (r_biss_rzt2_dat.h)

定数名	設定値	内容
R_BISS0_ID	0x01	BISS チャネル 0 の ID (R_ECL_CH_0)
R_BISS1_ID	0x02	BISS チャネル 1 の ID (R_ECL_CH_1)
R_BISS_FREQ	10000UL	BISS I/F の動作周波数 (R_ECL_FREQ_10000KHZ)

表 4.5　BiSS-C ドライバで使用する主要な定数 (r_bissc_rzt2_if.h)

定数名	設定値	内容	
R_BISSC_SLAVE_0	0	Slave ID (0～7)	
R_BISSC_SLAVE_1	1		
R_BISSC_SLAVE_2	2		
R_BISSC_SLAVE_3	3		
R_BISSC_SLAVE_4	4		
R_BISSC_SLAVE_5	5		
R_BISSC_SLAVE_6	6		
R_BISSC_SLAVE_7	7		
R_BISSC_CMD_POS	0xC0000001	コマンド	位置情報取得要求
R_BISSC_CMD_REG_READ	0xC0000002		レジスタ値リード (1 バイト)
R_BISSC_CMD_REG_WRITE	0xC0000003		レジスタ値ライト (1 バイト)
R_BISSC_CMD_REG_SREAD	0xC0000004		レジスタ値リード (指定バイト数連続)
R_BISSC_CMD_REG_SWRITE	0xC0000005		レジスタ値ライト (指定バイト数連続)
R_BISSC_CMD_REG_SSTOP	0xC0000006		連続レジスタリード・ライト停止
R_BISSC_CMD_REG_FSTOP	0xC0000007		レジスタリード・ライト、連続レジスタリード・ライト停止
R_BISSC_CMD_MAX	0xC0000007		コマンドの最大値
R_BISSC_SUCCESS	0	送受信結果	データ送受信正常終了
R_BISSC_REQ_ERR	-1		データ送受信エラー発生
R_BISSC_REQ_ERR_WR	-2		データ送受信エラー発生＆不正レジスタアクセス発生
R_BISSC_BR_10MHZ	0x13	ビットレート設定レジスタ (BR1) への設定値	
R_BISSC_BR_8_33MHZ	0x17		
R_BISSC_BR_4MHZ	0x31		
R_BISSC_BR_2_5MHZ	0x4F		
R_BISSC_BR_1MHZ	0xC7		
R_BISSC_BR_400KHZ	0x17C		
R_BISSC_BR_299_40KHZ	0x1A6		
R_BISSC_BR_200KHZ	0x1F9		
R_BISSC_BR_100KHZ	0x27C		
R_BISSC_BR_80_12KHZ	0x29B		
R_BISSC_BR_NUM	10	ビットレート設定レジスタへの設定値 定義数	
R_BISSC_ELC_DISABLE	0	ELCIN 制御ビット (ELCINE) への設定値	
R_BISSC_ELC_ENABLE	1		

表 4.6　BiSS-C ドライバで使用する主要な定数 (r_bissc_rzt2_config.h)

定数名	設定値	内容
BISSC_ISR_PRI0	11	BiSS-C I/F チャネル 0 用送受信割り込みの優先度 (0～15)
BISSC_ISR_PRI1	11	BiSS-C I/F チャネル 1 用送受信割り込みの優先度 (0～15)

4.9 固定幅整数一覧

表 4.7 にサンプルコードで使用する固定幅整数を示します。サンプルコードで使用する固定幅整数は、標準ライブラリで定義されています。

表 4.7 サンプルコードで使用する固定幅整数

シンボル	内容
int8_t	8 ビット整数、符号あり
int16_t	16 ビット整数、符号あり
int32_t	32 ビット整数、符号あり
int64_t	64 ビット整数、符号あり
uint8_t	8 ビット整数、符号なし
uint16_t	16 ビット整数、符号なし
uint32_t	32 ビット整数、符号なし
uint64_t	64 ビット整数、符号なし

4.10 構造体／共用体／列挙型一覧

4.10.1 構造体

(1) BiSS 共通

(a) r_biss_info_t

BiSS 制御部の初期化情報。

```
typedef struct
{
    uint16_t        clk_freq;       エンコーダに送信する clock 周波数設定 (80kHz～10MHz) 注1
    uint16_t        clk_re_freq;    使用しません
    r_biss_pos_crc_t crc_info;      CRC 情報 (BiSS-C 位置情報取得用)
    uint8_t         mtdata_size;    マルチターンデータサイズ設定 (0～31) 注2
    uint8_t         stdata_size;    シングルターンデータサイズ設定 (0～31) 注2
    uint8_t         aldata_size;    アラインメントデータサイズ設定 (0～31) 注2
} r_biss_info_t
```

【注】 1. 設定値の詳細は、RZ/T2M グループ BiSS インタフェース アプリケーションノートのビットレート設定レジスタ 1 (BR1) を参照してください。
2. 設定値は、使用するエンコーダの仕様を確認してください。

(b) r_biss_pos_crc_t

BiSS-C 位置情報取得時の CRC 情報

```
typedef struct
{
    uint8_t         crc_size;       CRC サイズ設定 (0～16) 注1
    uint16_t        crc_polynomial; CRC 生成多項式設定 注1
    uint16_t        crc_start_value; CRC 初期値 注1
} r_biss_pos_crc_t
```

【注】 1. 設定値は、使用するエンコーダの仕様を確認してください。
AD36/1219AF.0CBEB および WDGF 58M のデフォルト値は、ともに以下の通りです。
CRC サイズ 6 bit
CRC 生成多項式 $X^6 + X^1 + 1$
CRC データ初期値 0
設定値の詳細は、RZ/T2M グループ BiSS インタフェース アプリケーションノートのデータサイズ設定レジスタ (SIZE)の CRC データサイズ設定ビット (CRC1SIZ)、CRC 生成多項式設定レジスタ (CPOLY)、CRC 初期値設定レジスタ (CINIT)を参照してください。

(2) BiSS-C 専用

(a) r_bissc_sensreq_t

BiSS-C 準拠エンコーダから位置情報を取得する時のリクエスト情報

```
typedef struct
{
    uint8_t         slave_id;       Slave ID (0～7)
                                    Slave ID を指定します。エンコーダの接続が 1 対 1 の場合は、
                                    0 固定となります。
    uint32_t        timeout_clk;    ウォッチドッグタイマ設定値を指定
                                    設定値は、RZ/T2M グループ BiSS インタフェース アプリケー
                                    ションノートを参照してください。
    uint8_t         elc_enable;     ELC イベント入力トリガを有効にするか否かを設定
                                    (0: ELC イベント入力トリガ無効, 1: ELC イベント入力トリガ
                                    有効)
    r_biss_isr_cb_t sdresult_cb;    位置情報取得結果を通知するコールバック関数へのポインタ
                                    詳細は「4.5.1 callback_get_pos」参照。NULL を指定すると、
                                    コールバックが発生しません。
} r_bissc_sensreq_t
```

(b) r_bissc_regreq_t

BiSS-C 準拠エンコーダのレジスタにアクセスする時のリクエスト情報

```
typedef struct
{
    uint8_t         slave_id;       Slave ID (0～7)
                                    Slave ID を指定します。エンコーダの接続が 1 対 1 の場合は、
                                    0 固定となります。
    uint32_t        timeout_clk;    ウォッチドッグタイマ設定値を指定
                                    設定値は、RZ/T2M グループ BiSS インタフェース アプリケー
                                    ションノートを参照してください。
    uint8_t         regdtnum;       レジスタリード・ライト数 (1～64 Byte)
                                    レジスタリード、またはレジスタライトのバイトを設定しま
                                    す。1 バイトのレジスタリード・ライト時
                                    (R_BISSC_CMD_REG_READ, R_BISSC_CMD_REG_WRITE)
                                    には、1 を設定します。
    uint8_t         regaddress;     エンコーダのレジスタアドレスを指定します。(0～127)
                                    レジスタアドレスの詳細は、接続するエンコーダのマニュアル
                                    を参照してください。
    uint8_t         *p_regdata;     エンコーダのレジスタに書き込むデータを設定します。レジス
                                    タリード時には、リード結果が書き込まれます。
    r_biss_isr_cb_t sdresult_cb;    位置情報取得結果を通知するコールバック関数へのポインタ
                                    詳細は「4.5.1 callback_get_pos」参照。NULL を指定すると、
                                    コールバックが発生しません。
    r_biss_isr_cb_t rdresult_cb;    レジスタアクセス結果を通知するコールバック関数へのポイン
                                    タ
                                    詳細は「4.5.2 callback_reg_access」参照。NULL を指定する
                                    と、コールバックが発生しません。
} r_bissc_regreq_t
```

(c) r_bissc_sensordt_t

BiSS-C エンコーダからの位置情報受信結果ステータス

```
typedef struct
{
    r_bissc_req_err_t   result;     受信結果 注1
                                    詳細は「4.8 定数／エラーコード一覧」を参照してください。
    uint32_t            stdata;     シングルターンデータ 注2
    uint32_t            mtdata;     マルチターンデータ 注2
    bool                timeout;    タイムアウトエラー 注3
    bool                crc1_err;   CRC エラー (位置情報、エンコーダ状態) 注3
    bool                alarm;      エンコーダ Alarm 注3
    bool                warning;    エンコーダ Warning 注3
    bool                notready;   レジスタノットレディビット (true：ビジー状態) 注4
} r_bissc_sensordt_t
```

【注】 1. 受信結果は、timeout, crc1_err のどちらか一方でも true のとき、送受信エラー発生として扱われます。Alarm や Warning は、データ受信エラーとしては扱われません。Alarm や Warning の発生条件は各エンコーダの仕様に依存します。各エンコーダのマニュアルを参照してください。

2. タイムアウトエラー以外の場合は、取得した値が設定されます。
3. エラー, Alarm, Warning 発生時に true となります。
4. 本フラグが立ったまま、しばらくレジスタアクセスが完了しない場合は、リセットするか、強制停止 (R_BISSC_CMD_REG_FSTOP) 後にリトライしてください。

(d) r_bissc_regdata_t

BiSS-C エンコーダのレジスタアクセス結果

```
typedef struct
{
  r_bissc_req_err_t   result;     送受信結果 注1
                                  詳細は「4.8 定数／エラーコード一覧」を参照してください。
  uint8_t             idl;        ID-Lock データ
                                  例) エンコーダ 1 つ接続： 0b00000001
                                      エンコーダ 2 つ接続： 0b00000011
  uint8_t             combyte;    レジスタアクセス完了バイト数 注2
  bool                timeout;    タイムアウトエラー 注3
  bool                crc1_err;   CRC エラー (位置情報、エンコーダ状態) 注3
  bool                crc2_err;   CRC エラー (レジスタデータ) 注3
  bool                alarm;      エンコーダ Alarm 注3
  bool                warning;    エンコーダ Warning 注3
  bool                regw_err;   レジスタライトエラー 注3
  bool                readbit_err;  リードビットデータエラー 注3
  bool                writebit_err; ライトビットデータエラー 注3
  bool                stopbit_err;  レジスタアクセスでのストップビット情報
                                  （true: 次にアクセスを行うレジスタが存在しない、
                                    または、自動インクリメントでアドレス指定できない）
  bool                cds_err;    アフターID-Lock データエラー
                                  （true: レジスタアクセスで、IDL 8 ビットを受信後、R ビットが現
                                    れるまでの CDS 値が不正）
} r_bissc_regdata_t
```

【注】 1. 送受信結果は、timeout, crc1_err, crc2_err, regw_err, readbit_err, writebit_err, cds_err のうち、どれか 1 つでも true のとき、送受信エラー発生として扱われます。Alarm, Warning, ストップビット情報は、送受信エラーとしては扱われません。Alarm や Warning の発生条件は各エンコーダの仕様に依存します。各エンコーダのマニュアルを参照してください。

2. タイムアウトエラー以外のエラーが発生した場合は、エラー発生前までのアクセス完了バイト数が設定されます。
3. エラー, Alarm, Warning 発生時に true となります。

4.10.2 共用体

使用しません。

4.10.3 列挙型

使用しません。

4.11 サンプルプログラムの説明

4.11.1 動作概要

本サンプルプログラムは以下の処理を行います。

1) コンソールから入力したコマンドにより BiSS-C エンコーダからの位置情報読み出し行う
2) コンソールから入力したコマンドにより BiSS-C エンコーダのレジスタにリード・ライトを行う
3) ELC イベント入力トリガ機能を使用して、BiSS-C エンコーダから定期的に位置情報を取得する
4) BiSS-C エンコーダから受信したデータをコンソールに表示する

(1) システムブロック図

図 4-1 にシステムブロック図を示します。

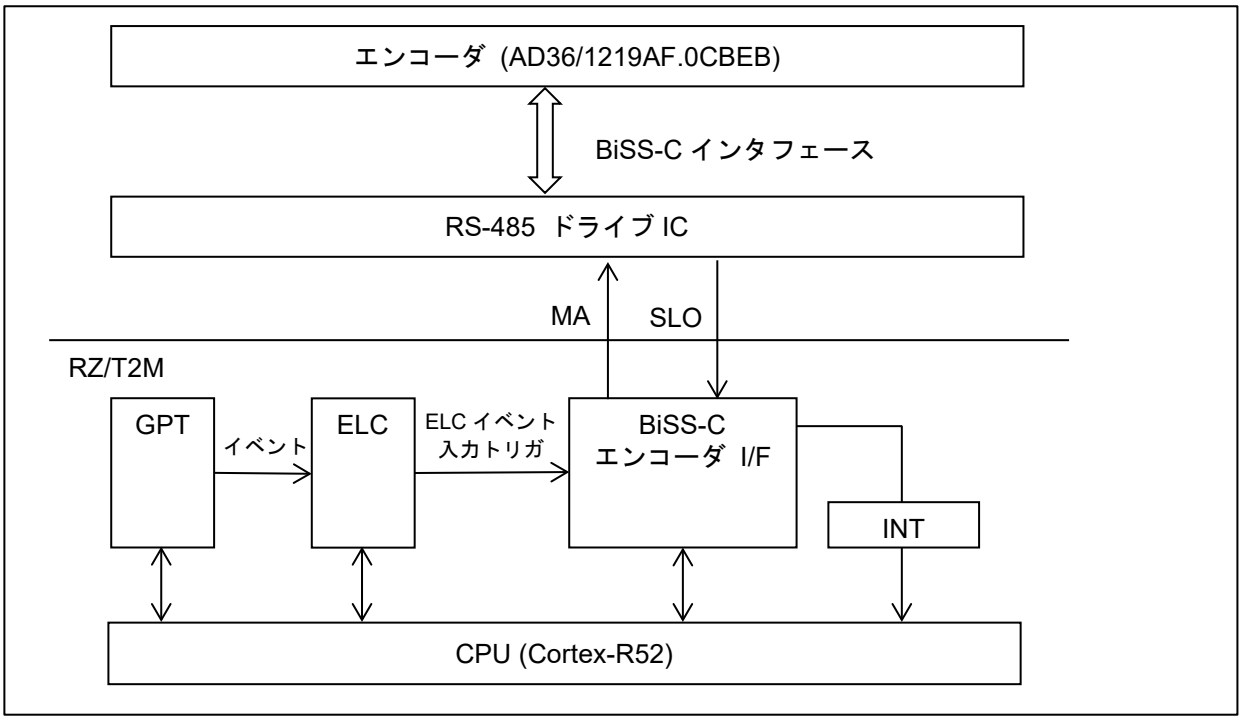


図 4-1 システムブロック図

(2)　ソフトウェア構成図

図 4-2 にソフトウェア構成図を示します。

BiSS ドライバには、BiSS ドライバ共通部、R_BISS_Open 関数で構成される開始処理部、R_BISS_Close 関数で構成される終了処理部、R_BISS_Control 関数で構成されるリクエスト送信部、コールバック関数で構成されるデータ受信部（割り込みハンドラ）があります。

サンプルプログラムには、BiSS ドライバ共通部を通して BiSS-C ドライバを制御し、リクエスト送信を行う BiSS-C ドライバ制御部、データ送受信結果の表示を行う結果表示部（コールバック）があります。

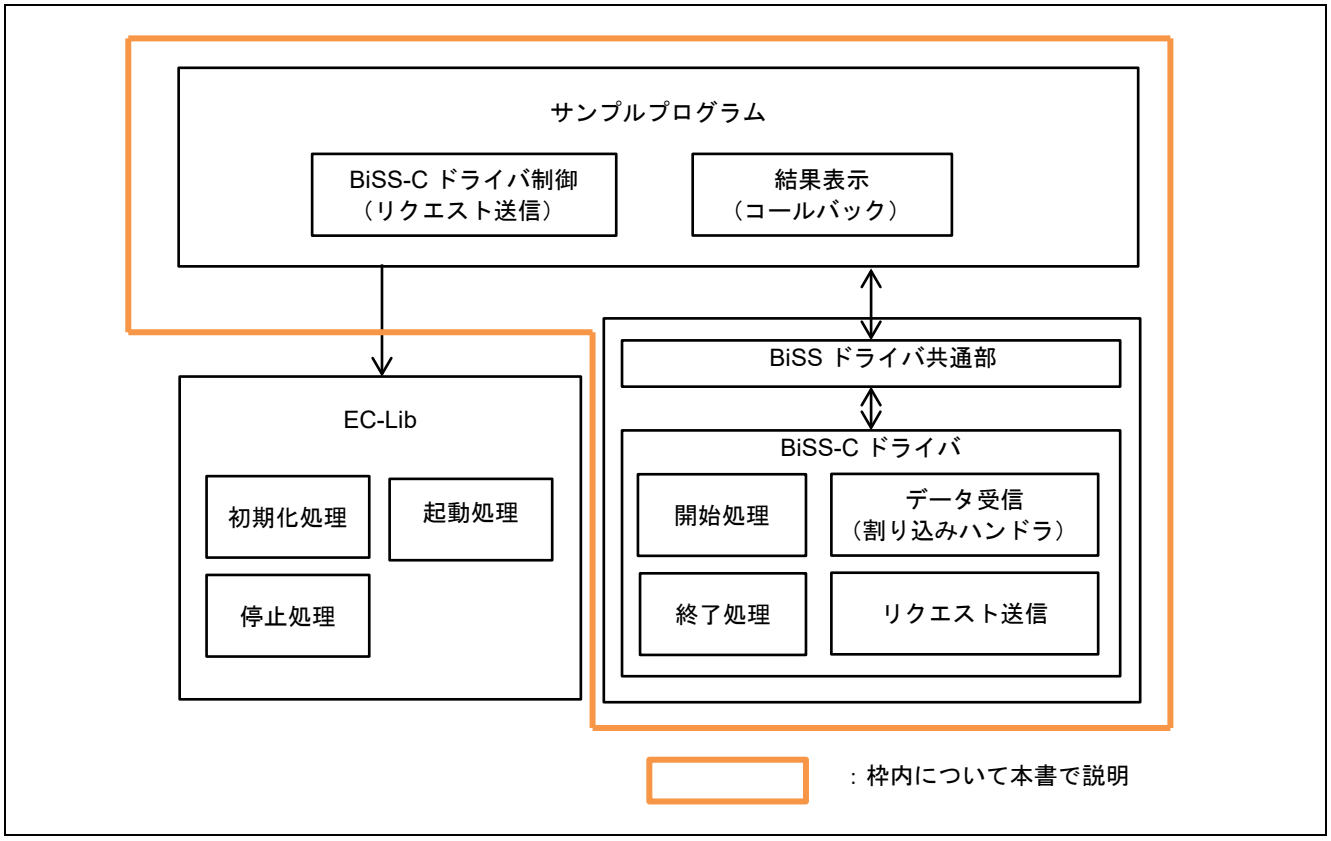


図 4-2　ソフトウェア構成図

4.11.2 サンプルプログラム関数一覧

表 4.8 にサンプルプログラム関数一覧を示します

表 4.8　サンプルプログラム関数一覧

4.11.3 サンプルプログラム関数仕様

(1) hal_entry

hal_entry		
概　要	BiSS サンプルプログラムのエントリー関数	
ヘッダ	-	
宣　言	void hal_entry(void);	
説　明	BiSS サンプルプログラムのエントリー関数です。ここから、関数 enc_main()が呼び出されます。	
引　数	なし	
リターン値	なし	

(2) enc_main

enc_main		
概　要	BiSS サンプルプログラムのメイン関数	
ヘッダ	-	
宣　言	int32_t enc_main(uint8_t ch);	
説　明	BiSS サンプルプログラムのメイン関数です。詳細は「4.11.5(1) enc_main フローチャート」を参照してください。	
引　数	ch	エンコーダチャネル番号 0：ch0 を指定, 1：ch1 を指定
リターン値	0 0 以外	：正常終了 ：異常終了（エンコーダ I/F のエラーコード）

(3) biss_req_send

biss_req_send		
概　要	BiSS-C エンコーダへのリクエスト処理関数	
ヘッダ	-	
宣　言	static int32_t biss_req_send(uint32_t arg_num, char_t *p_arg[]);	
説　明	BiSS-C エンコーダへのリクエスト処理を行います。詳細は「4.11.5(2) biss_req_send フローチャート」を参照してください。	
引　数	arg_num *p_arg[]	リクエスト情報の数 リクエスト情報
リターン値	cmd_index	コマンド番号

(4) bissc_get_pos

bissc_get_pos	
概　要	BiSS-C エンコーダから位置情報を取得する関数
ヘッダ	-
宣　言	static void bissc_get_pos(void);
説　明	BiSS-C エンコーダから位置情報を取得します。詳細は「4.11.5(3) bissc_get_pos フローチャート」を参照してください。
引　数	なし
リターン値	なし

(5) bissc_position

bissc_position		
概　要	BiSS-C エンコーダから位置情報を取得して表示する関数	
ヘッダ	-	
宣　言	static void bissc_position(char_t *p_arg[], const uint32_t arg_num);	
説　明	BiSS-C エンコーダから取得した位置情報を表示します。位置情報取得のために、内部で bissc_get_pos() 関数を呼び出しています。詳細は「4.11.5(4) bissc_position フローチャート」を参照してください。	
引　数	*p_arg[]	リクエスト情報　(使用しません)
	arg_num	リクエスト情報の数
リターン値	なし	

(6) bissc_cmd_read

bissc_cmd_read		
概　要	BiSS-C エンコーダのレジスタからデータを指定バイト数読み出す関数	
ヘッダ	-	
宣　言	static void bissc_cmd_read(char_t *p_arg[], const uint32_t arg_num);	
説　明	BiSS-C エンコーダのレジスタからデータを指定バイト数読み出します。詳細は「4.11.5(5) bissc_cmd_read フローチャート」を参照してください。	
引　数	*p_arg[]	リクエスト情報
	arg_num	リクエスト情報の数
リターン値	なし	

(7) bissc_cmd_write

bissc_cmd_write		
概　要	BiSS-C エンコーダのレジスタにデータを指定バイト数書き込む関数	
ヘッダ	-	
宣　言	static void bissc_cmd_write(char_t *p_arg[], const uint32_t arg_num);	
説　明	BiSS-C エンコーダのレジスタにデータを指定バイト数書き込みます。詳細は「4.11.5(6) bissc_cmd_write フローチャート」を参照してください。	
引　数	*p_arg[]	リクエスト情報
	arg_num	リクエスト情報の数
リターン値	なし	

(8) bissc_trans_timer

bissc_trans_timer	
概　要	BiSS-C エンコーダ I/F に ELC イベントに同期した位置情報取得を設定する関数
ヘッダ	-
宣　言	static void bissc_trans_timer(void);
説　明	BiSS-C エンコーダ I/F を、ELC イベント入力トリガに同期して位置情報を取得するように設定します。詳細は「4.11.5(7) bissc_trans_timer フローチャート」を参照してください。
引　数	なし
リターン値	なし

(9) bissc_cmd_elctimer

bissc_cmd_elctimer		
概　要	BiSS-C エンコーダから、ELC イベントに同期して位置情報を読み出す関数	
ヘッダ	-	
宣　言	static void bissc_cmd_elctimer(char_t *p_arg[], const uint32_t arg_num);	
説　明	BiSS-C エンコーダから、ELC イベント入力トリガに同期して周期的に位置情報を読み出します。BiSS-C エンコーダ I/F の設定のために、内部で bissc_trans_timer() 関数を呼び出しています。詳細は「4.11.5(8) bissc_cmd_elctimer フローチャート」を参照してください。	
引　数	*p_arg[]	リクエスト情報
	arg_num	リクエスト情報の数
リターン値	なし	

(10) bissc_cmd_elcstop

bissc_cmd_elcstop		
概　要	ELC イベントに同期した位置情報読み出しを終了する関数	
ヘッダ	-	
宣　言	static void bissc_cmd_elcstop(char_t *p_arg[], const uint32_t arg_num);	
説　明	BiSS-C エンコーダからの、ELC イベント入力トリガに同期した位置情報読み出しを終了して、取得した位置情報を表示する関数です。詳細は「4.11.5(9) bissc_cmd_elcstop フローチャート」を参照してください。	
引　数	*p_arg[]	リクエスト情報 (使用しません)
	arg_num	リクエスト情報の数
リターン値	なし	

(11) bissc_exit

bissc_exit		
概　要	BiSS-C エンコーダプログラム終了関数	
ヘッダ	-	
宣　言	static void bissc_exit(char_t *p_arg[], const uint32_t arg_num);	
説　明	BiSS-C エンコーダプログラムを終了します。詳細は「4.11.5(10) bissc_exit フローチャート」を参照してください。	
引　数	*p_arg[]	リクエスト情報 (使用しません)
	arg_num	リクエスト情報の数
リターン値	なし	

(12) reg_acc_req

reg_acc_req		
概　要	BiSS-C エンコーダレジスタアクセス関数	
ヘッダ	-	
宣　言	static void reg_acc_req(r_biss_cmd_t cmd);	
説　明	BiSS-C エンコーダのレジスタアクセスを実行します。詳細は「4.11.5(11) reg_acc_req フローチャート」を参照してください。	
引　数	cmd	コマンド
リターン値	なし	

(13) callback_get_pos

callback_get_pos	
概　要	BiSS-C エンコーダからの位置情報受信結果を通知するコールバック関数
ヘッダ	-
宣　言	static void callback_get_pos(void *p_result);
説　明	結果をメモリに保存し、取得完了フラグを立てます。詳細は「4.11.5(12) callback_get_pos フローチャート」を参照してください。
引　数	p_result　：位置情報取得結果を格納した r_bissc_sensordt_t 構造体へのポインタ。詳細は「4.5.1 callback_get_pos」を参照してください。
リターン値	なし

(14) callback_reg_access

callback_reg_access	
概　要	BiSS-C エンコーダへのレジスタアクセス結果を通知するコールバック関数
ヘッダ	-
宣　言	static void callback_reg_access(void *p_result);
説　明	結果をメモリに保存し、取得完了フラグを立てます。詳細は「4.11.5(13) callback_reg_access フローチャート」を参照してください。
引　数	p_result　：レジスタアクセス結果を格納した r_bissc_regdata_t 構造体へのポインタ。詳細は「4.5.2 callback_reg_access」を参照してください。
リターン値	なし

(15) callback_elctimer_result

callback_elctimer_result	
概　要	BiSS-C エンコーダから ELC イベント同期で位置情報受信した結果を通知するコールバック関数
ヘッダ	-
宣　言	static void callback_elctimer_result(void *p_result);
説　明	位置情報の受信結果を biss_ti_result 変数に保存します。詳細は「4.11.5(14) callback_elctimer_result フローチャート」を参照してください。
引　数	p_result　：位置情報取得結果を格納した r_bissc_sensordt_t 構造体へのポインタ。 詳細は「4.5.3 callback_elctimer_result」を参照してください。
リターン値	なし

4.11.4 サンプルプログラムの変数一覧

表 4.9 に主要な static 型変数を示します。表 4.10 に主要な static const 型変数を示します。

表 4.9　主要な static 型変数

型	変数名	内容	使用関数
bool	bissc_sensor_flg	位置情報取得完了フラグ 初期値： false	bissc_get_pos, bissc_trans_timer, reg_acc_req, callback_get_pos
bool	bissc_register_flg	レジスタアクセス完了フラグ 初期値： false	biss_req_send, reg_acc_req, callback_reg_access
r_bissc_sensreq_t	bissc_sensreq	位置情報取得時のリクエスト情報	bissc_get_pos, bissc_trans_timer
r_bissc_regreq_t	bissc_regreq	レジスタアクセス時のリクエスト情報	biss_req_send, bissc_cmd_read, bissc_cmd_write, reg_acc_req
r_bissc_sensordt_t*	p_bissc_result_sns	位置情報データ取得結果	bissc_position, callback_get_pos, callback_elctimer_result
r_bissc_regdata_t*	p_bissc_result_reg	レジスタアクセス結果	biss_req_send, callback_reg_access
bool	bissc_cmd_flg	コマンド処理実行中フラグ 初期値： false	enc_main, biss_req_send, reg_acc_req
bool	bissc_elc_flg	ELC タイマモード動作中フラグ 初期値： false	biss_req_send, bissc_cmd_elctimer, bissc_cmd_elcstop
uint8_t	reg_data[64]	レジスタリード・ライト用バッファ	biss_req_send, bissc_cmd_read, bissc_cmd_write
r_bissc_sensordt_t	biss_ti_result[10]	ELC タイマモード 位置情報格納用リングバッファ	bissc_cmd_elcstop, callback_elctimer_result
uint32_t	biss_ti_count	リングバッファ格納位置カウンタ	bissc_cmd_elctimer, bissc_cmd_elcstop, callback_elctimer_result
uint32_t	biss_ti_valid	リングバッファ有効データ数	bissc_cmd_elctimer, bissc_cmd_elcstop, callback_elctimer_result
bool	biss_ti_full	リングバッファ full フラグ	bissc_cmd_elctimer, bissc_cmd_elcstop, callback_elctimer_result

表 4.10　主要な static const 型変数

型	変数名	内容	使用関数
char_t *[]	p_cmd_tbl	コマンド文字列チェックテーブル 要素数： CMD_NUM 内容： { “pos”, “read”, “write”, “elctimer”, “elcstop”, “exit” }	biss_req_send
cmd_func_t []	cmd_func_tbl	コマンド関数アドレステーブル 要素数： CMD_NUM 内容： { &bissc_position, &bissc_cmd_read, &bissc_cmd_write, &bissc_cmd_elctimer, &bissc_cmd_elcstop, &bissc_exit }	biss_req_send

4.11.5 メイン処理のフローチャート

主要な関数のフローチャートを記載します。

(1) enc_main フローチャート

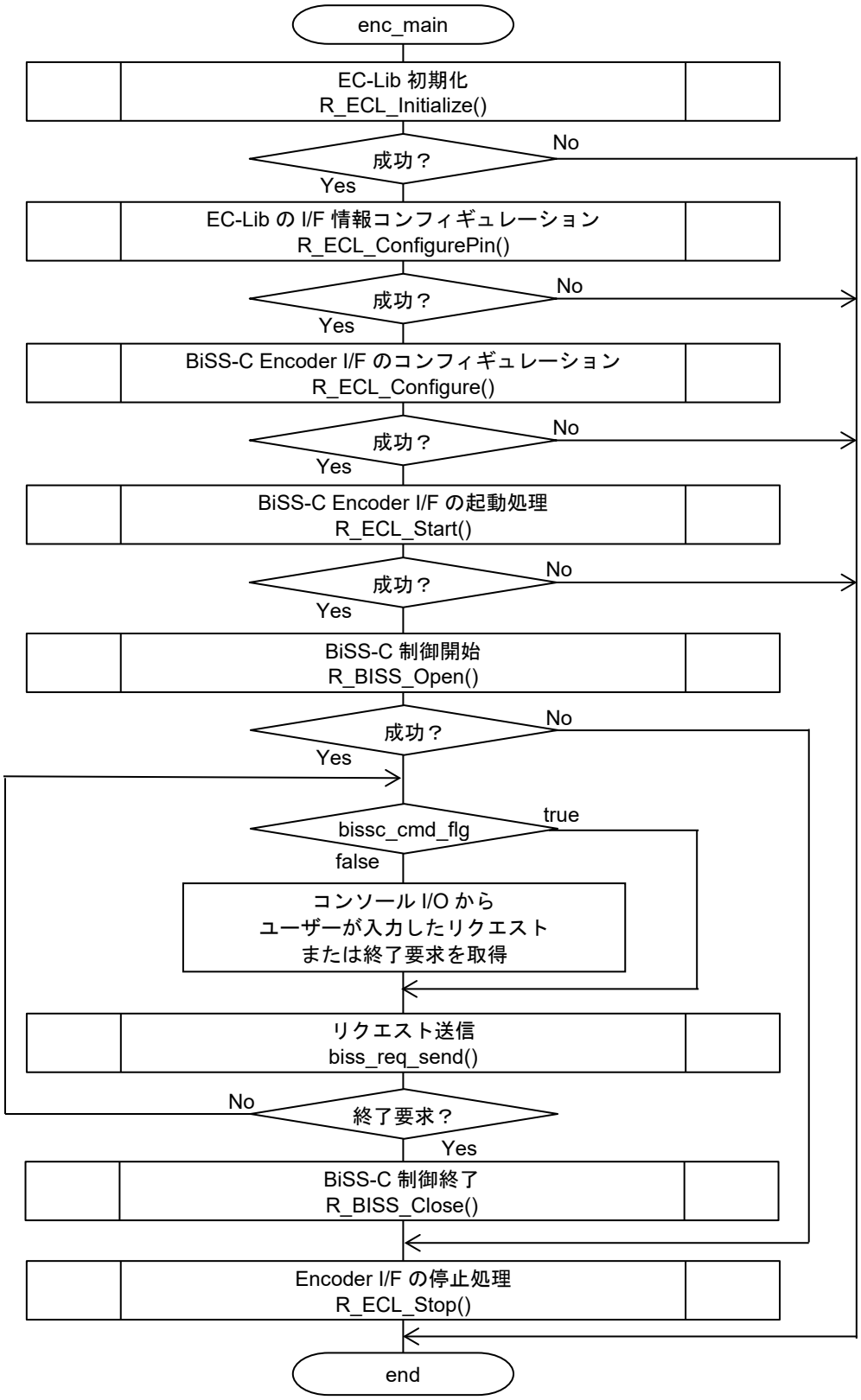
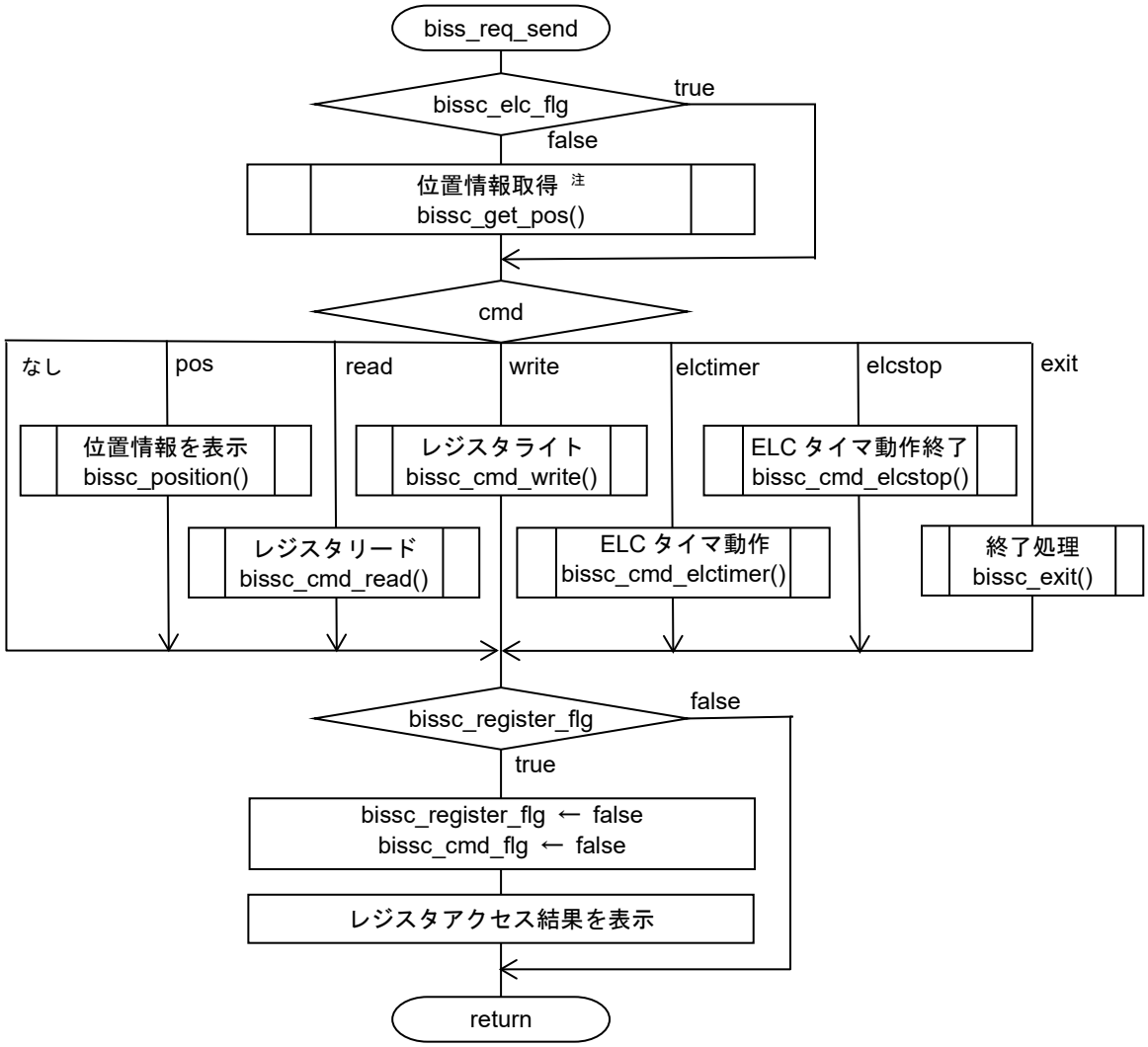


図 4-3 enc_main 関数のフローチャート

(2)　biss_req_send フローチャート



【注】　連続レジスタライト実行時、エンコーダの EEPROM アクセス時間によっては、位置情報取得前にウェイト時間が必要になります。
EEPROM アクセス時間については、使用するエンコーダの仕様を確認してください。

図 4-4　biss_req_send 関数のフローチャート

(3) bissc_get_pos フローチャート

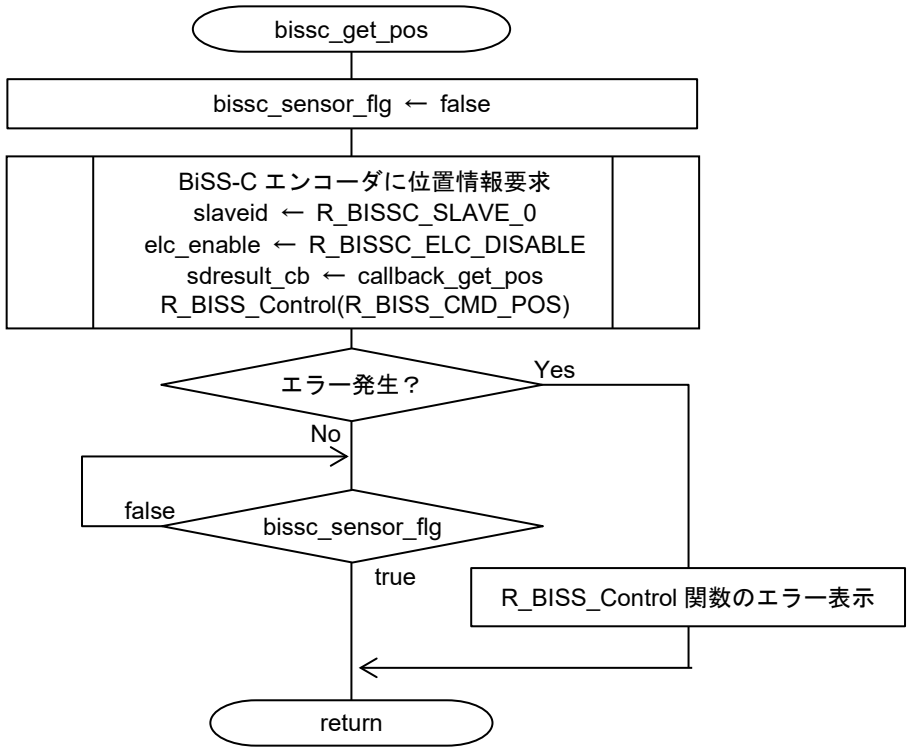


図 4-5 bissc_get_pos 関数のフローチャート

(4) bissc_position フローチャート

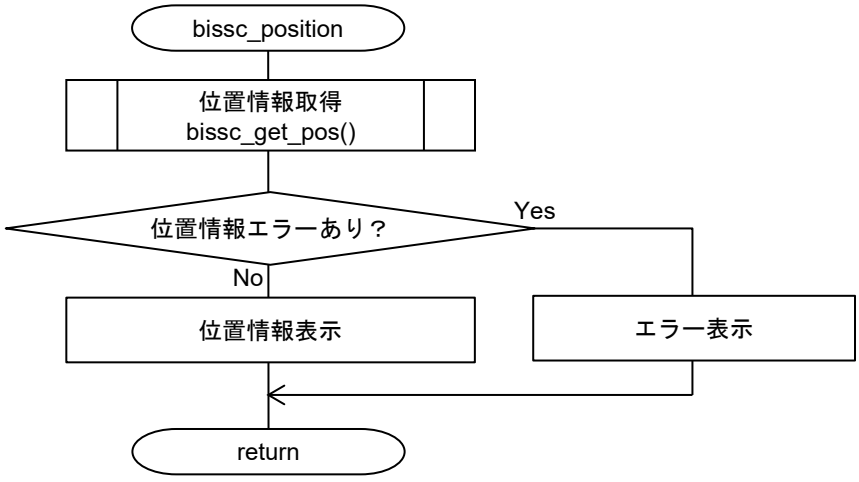


図 4-6 bissc_position 関数のフローチャート

(5) bissc_cmd_read フローチャート

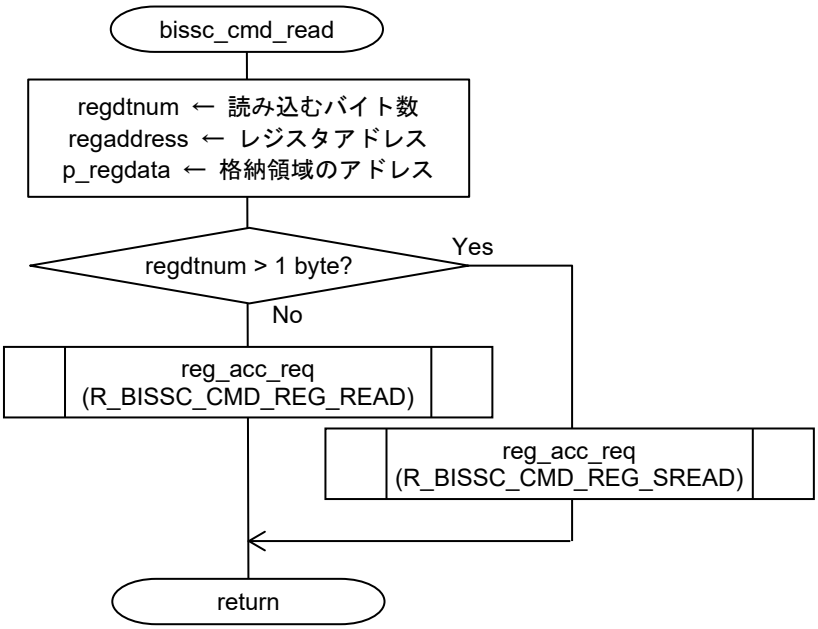


図 4-7 bissc_cmd_read 関数のフローチャート

(6) bissc_cmd_write フローチャート

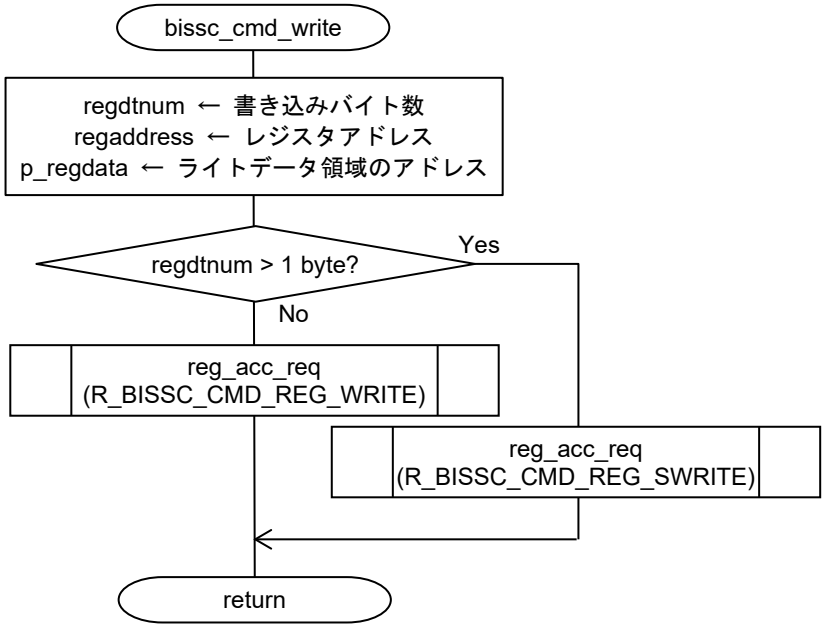


図 4-8 bissc_cmd_write 関数のフローチャート

(7) bissc_trans_timer フローチャート

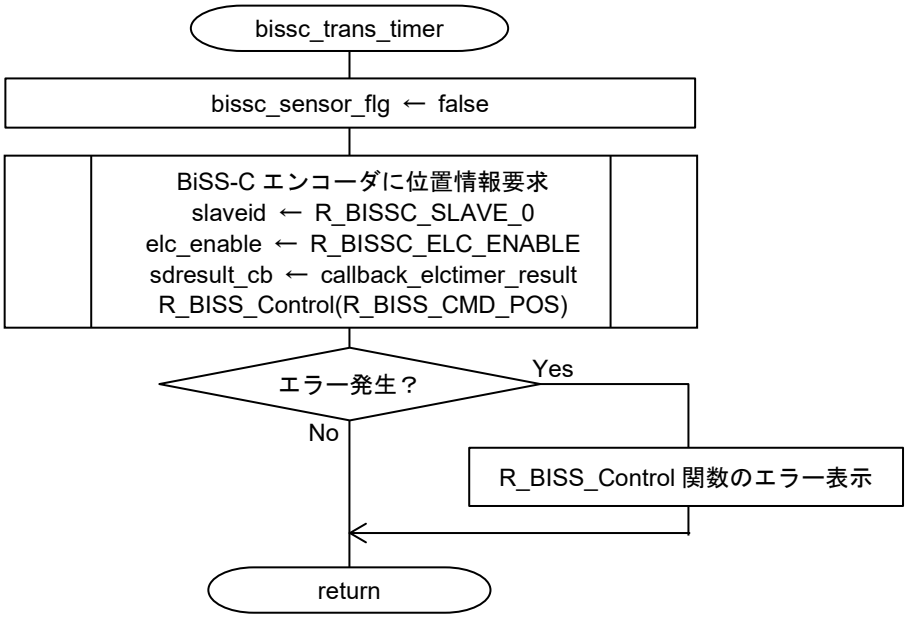


図 4-9 bissc_trans_timer 関数のフローチャート

(8) bissc_cmd_elctimer フローチャート

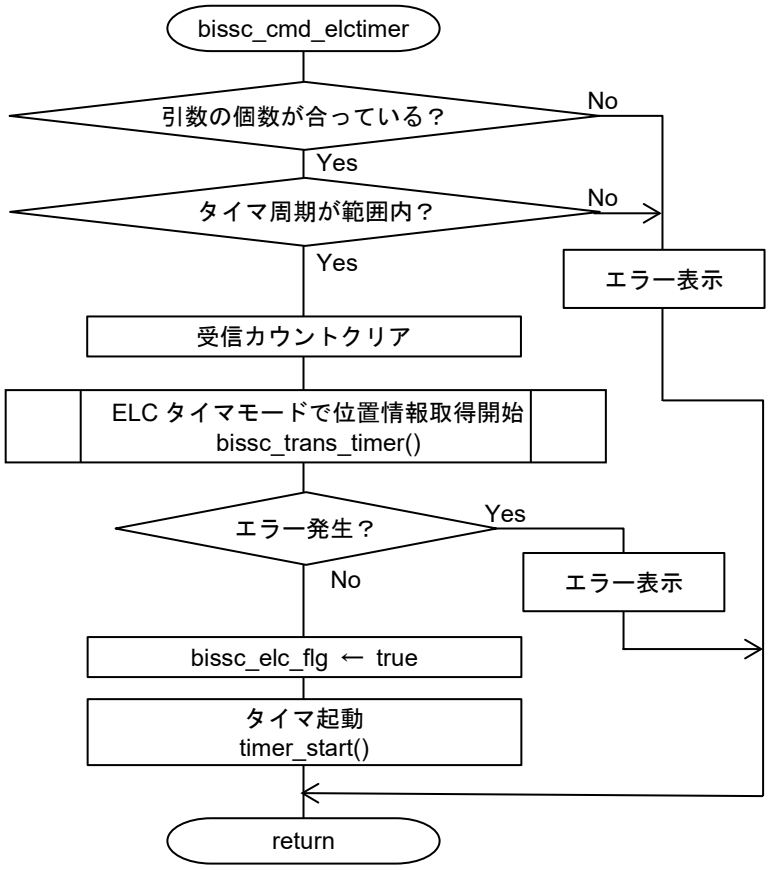


図 4-10 bissc_cmd_elctimer 関数のフローチャート

(9) bissc_cmd_elcstop フローチャート

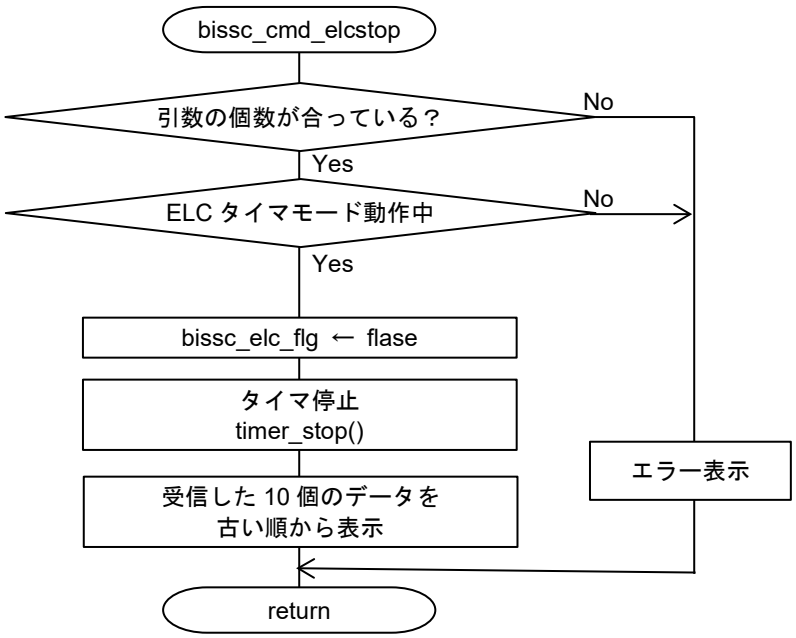


図 4-11 bissc_cmd_elcstop 関数のフローチャート

(10) bissc_exit フローチャート

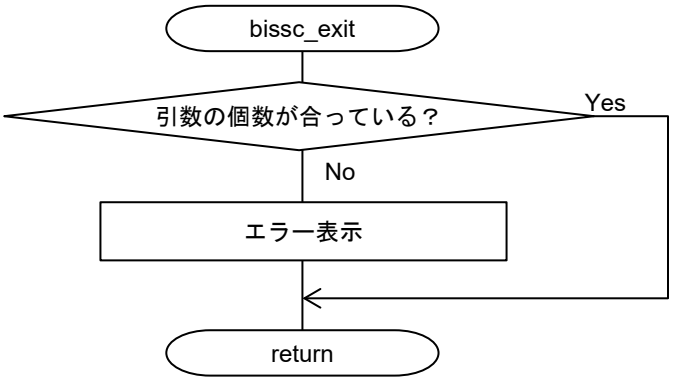


図 4-12 bissc_exit 関数のフローチャート

(11) reg_acc_req フローチャート

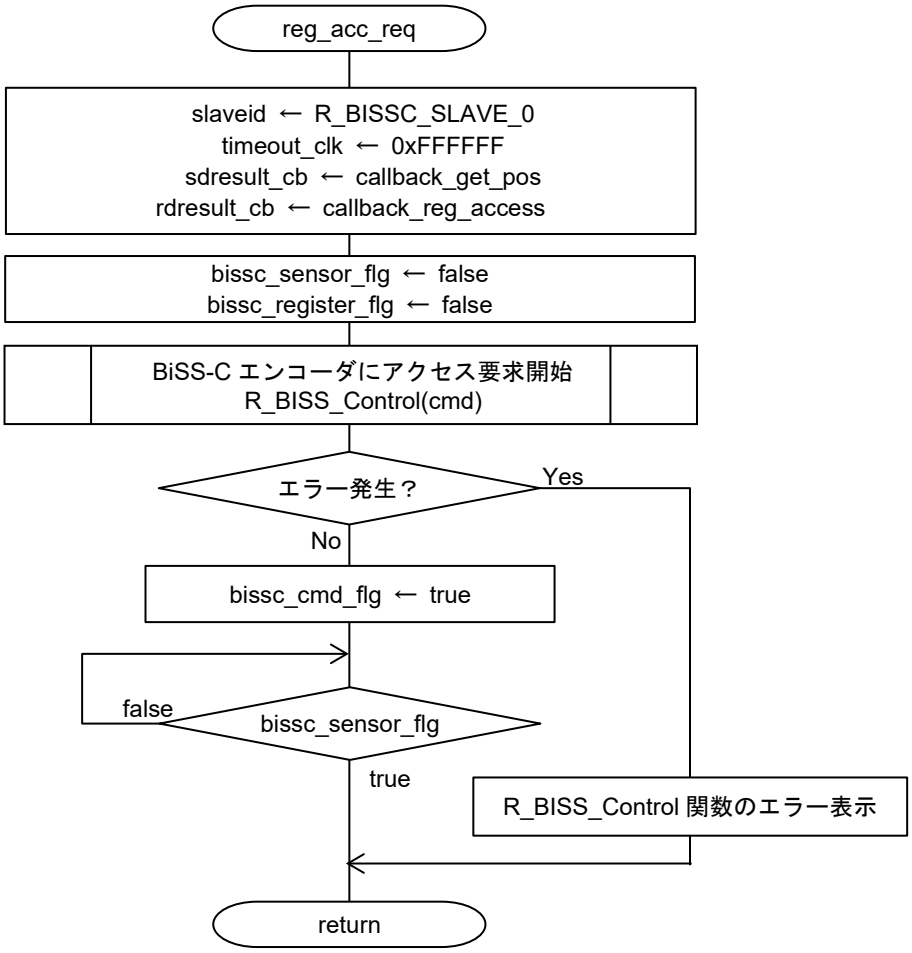


図 4-13　reg_acc_req 関数のフローチャート

(12) callback_get_pos フローチャート

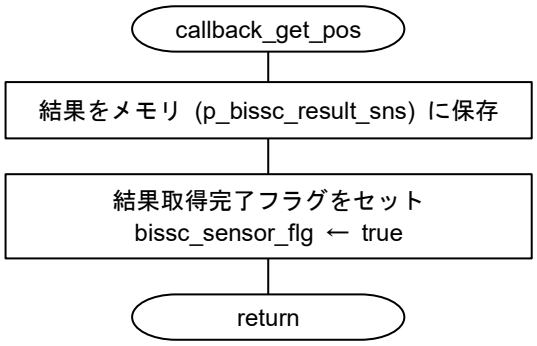


図 4-14　callback_get_pos 関数のフローチャート

(13) callback_reg_access フローチャート

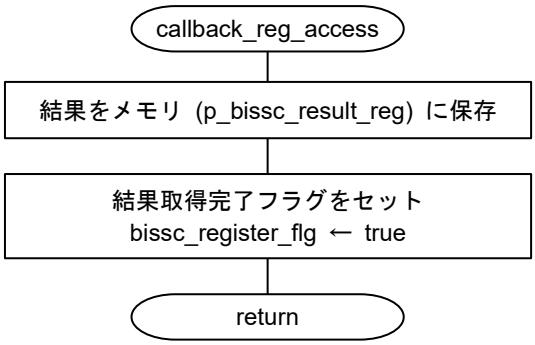


図 4-15　callback_reg_access 関数のフローチャート

(14) callback_elctimer_result フローチャート

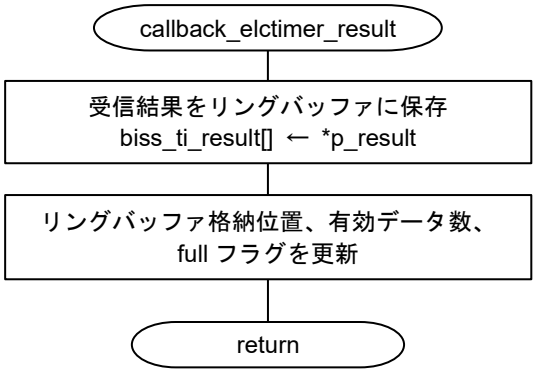


図 4-16　callback_elctimer_result 関数のフローチャート

4.11.6 動作シーケンス

(1) 開始シーケンス

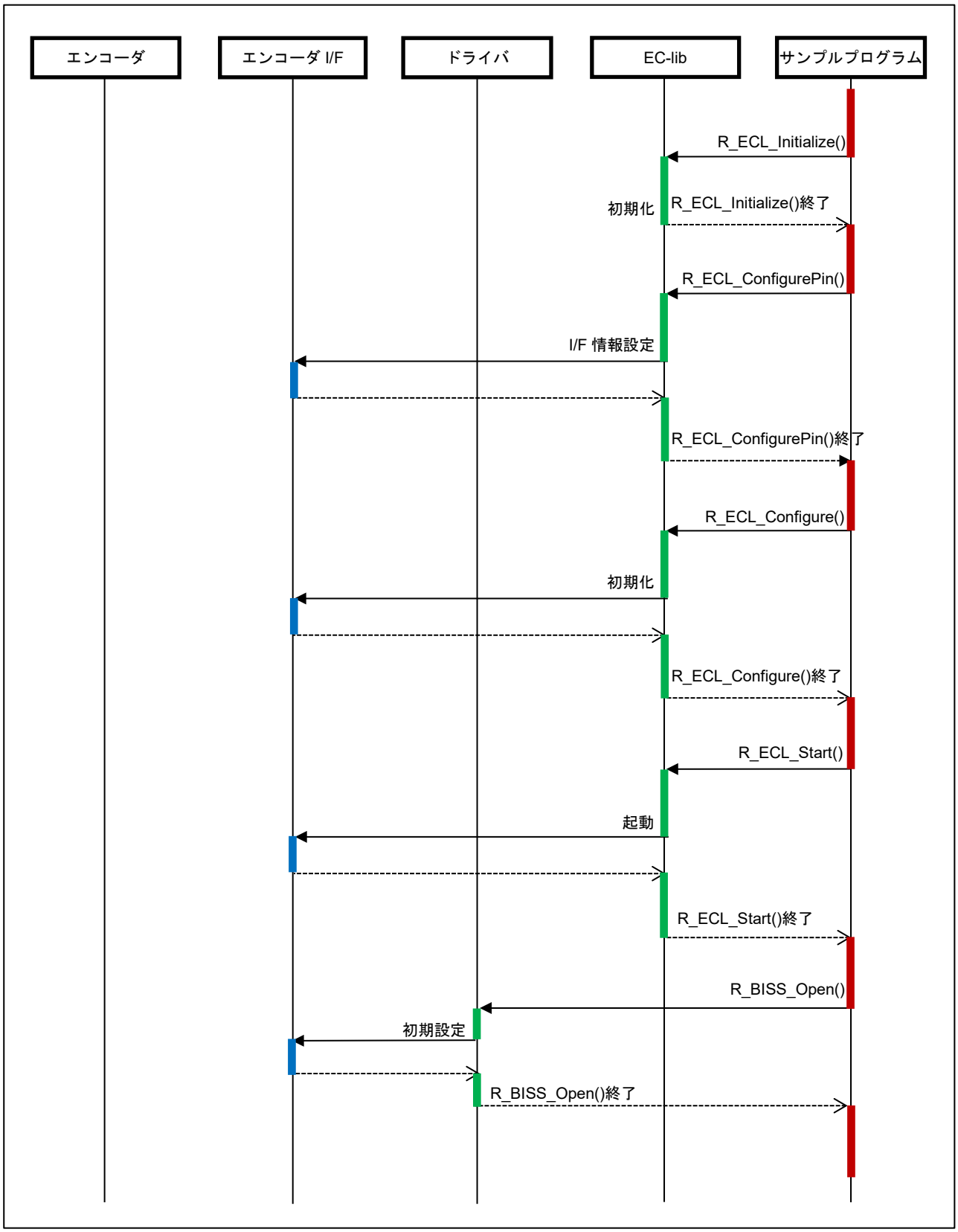


図 4-17 開始シーケンス図

(2)　センサーモード（位置情報取得）シーケンス

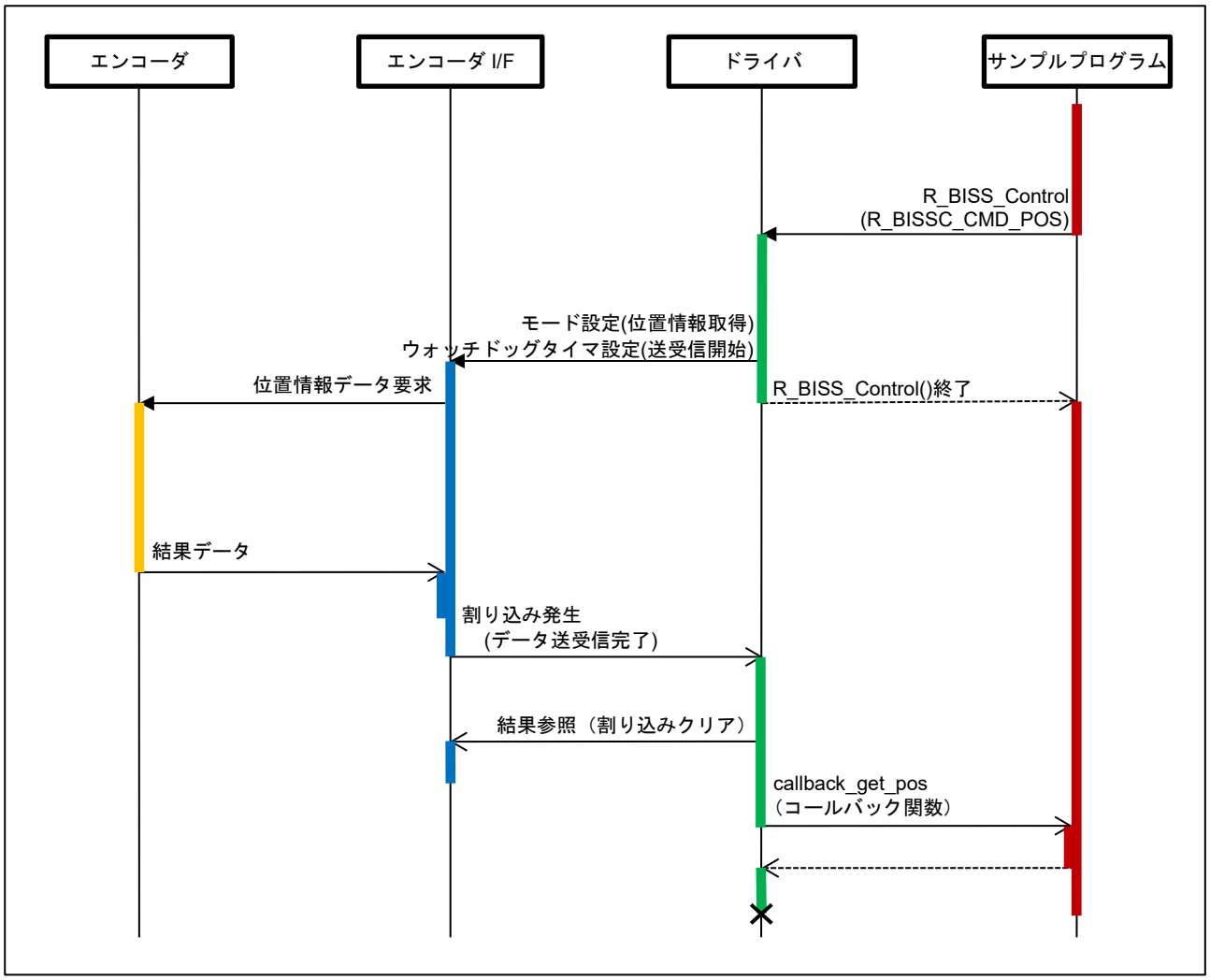


図 4-18　センサーモード（位置情報取得）シーケンス図

(3) レジスタアクセス（リード・ライト）シーケンス

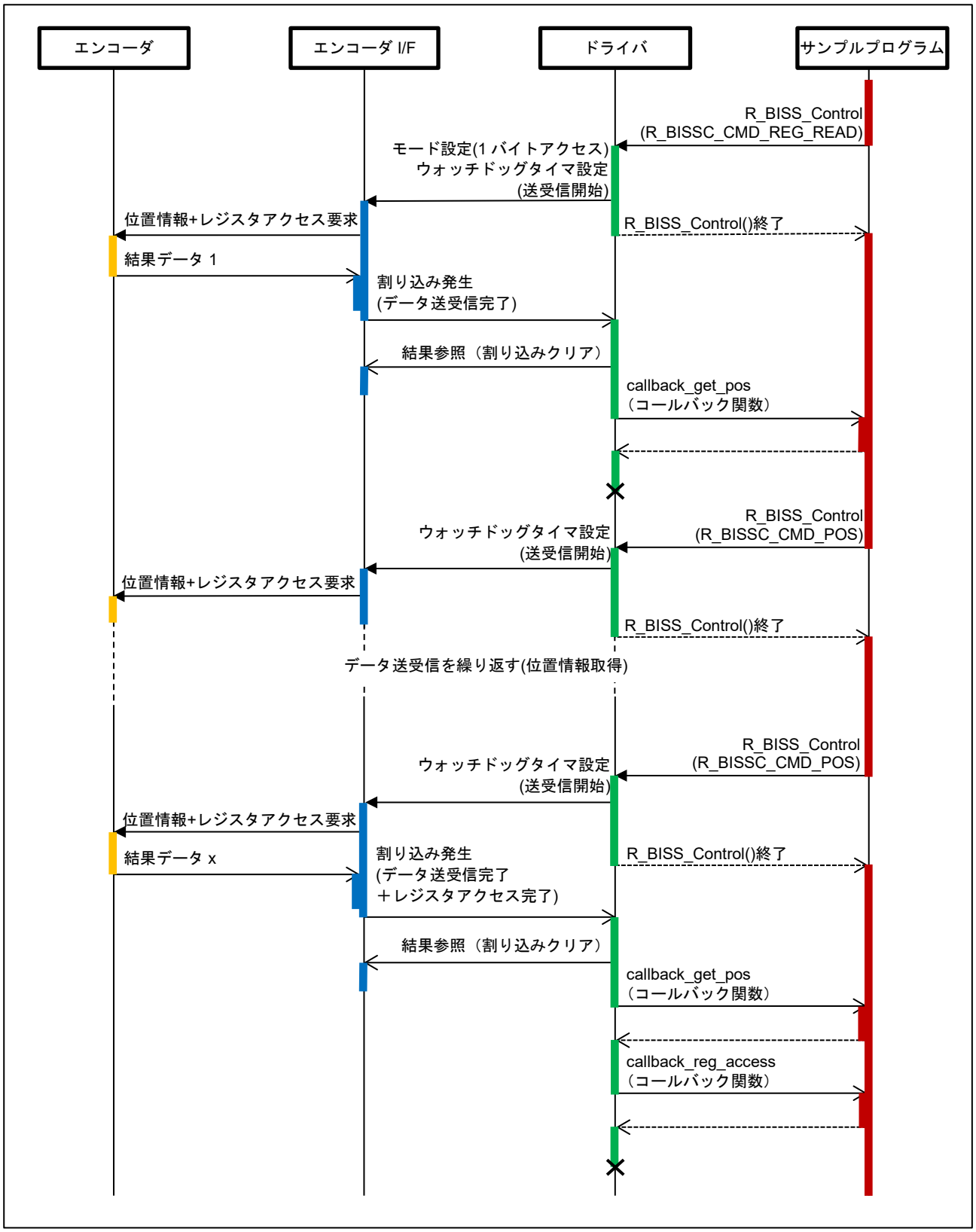


図 4-19 レジスタアクセス（リード・ライト）シーケンス図

(4)　連続レジスタアクセス（リード・ライト）シーケンス

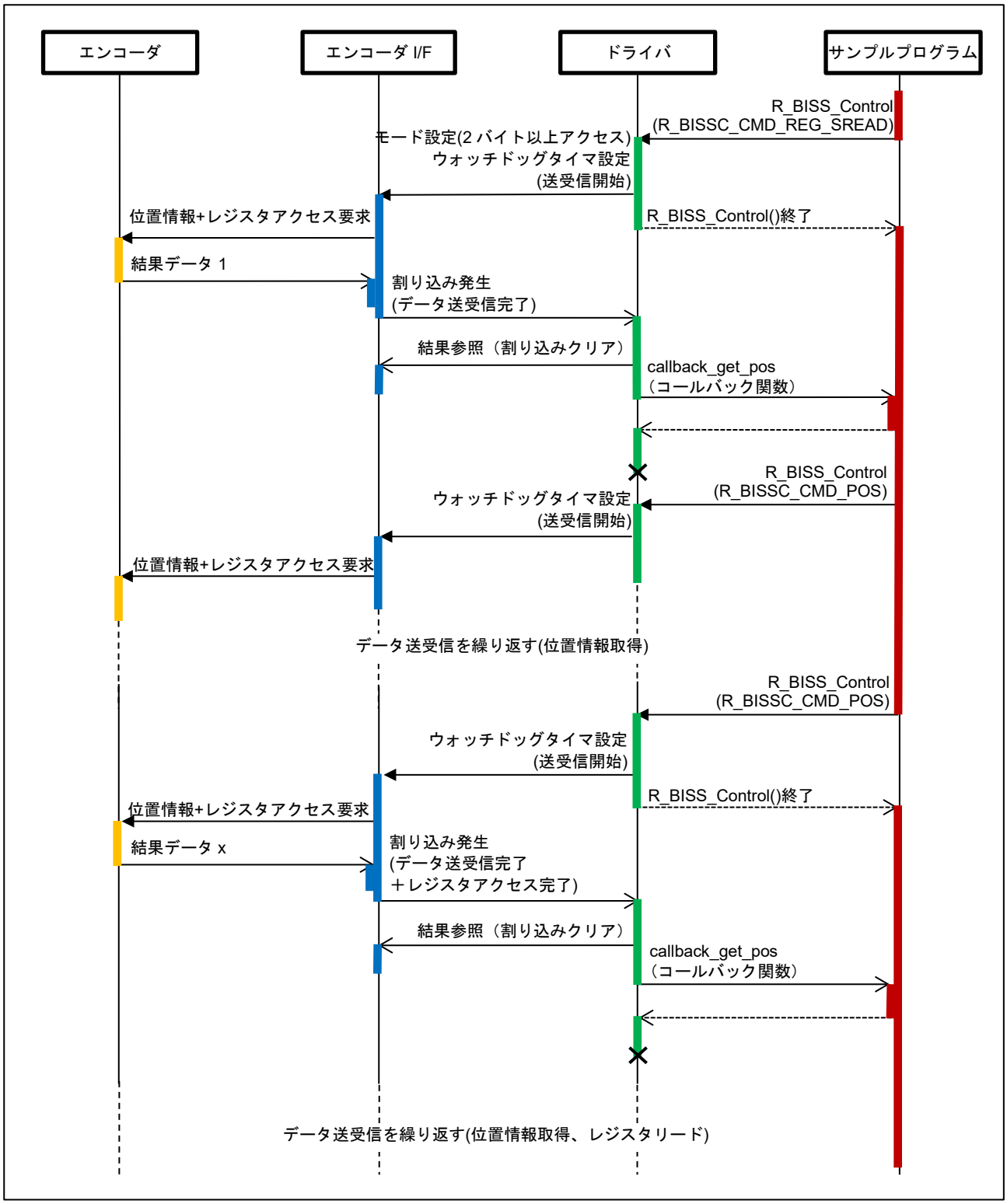


図 4-20　連続レジスタアクセス（リード・ライト）シーケンス図（1/2）

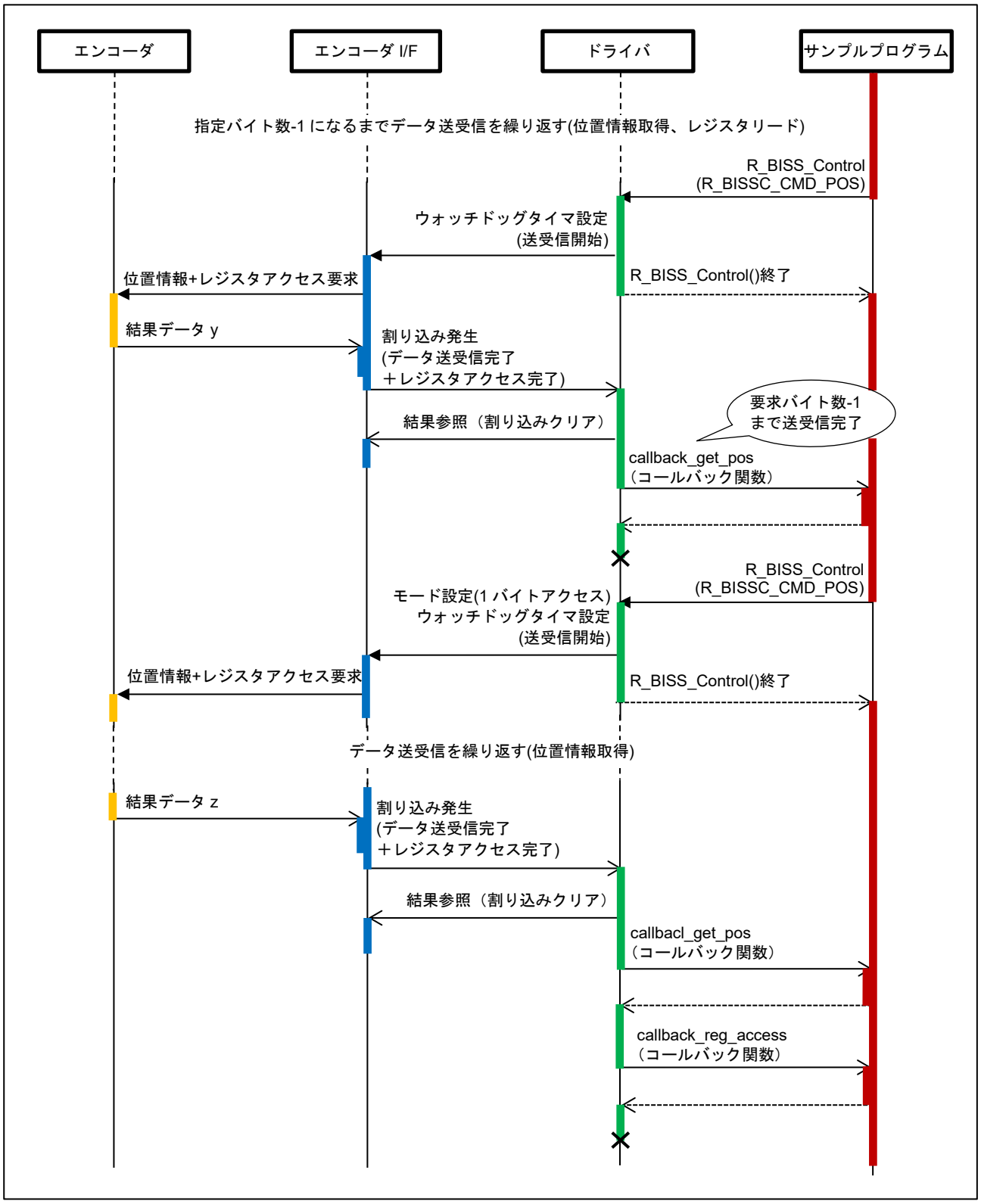


図 4-21　連続レジスタアクセス（リード・ライト）シーケンス図（2/2）

(5)　連続レジスタアクセス（リード・ライト）シーケンス（途中でエラー発生）

連続レジスタアクセスの途中で、エラーが発生した場合のシーケンス図を、図 4-22 に示します。

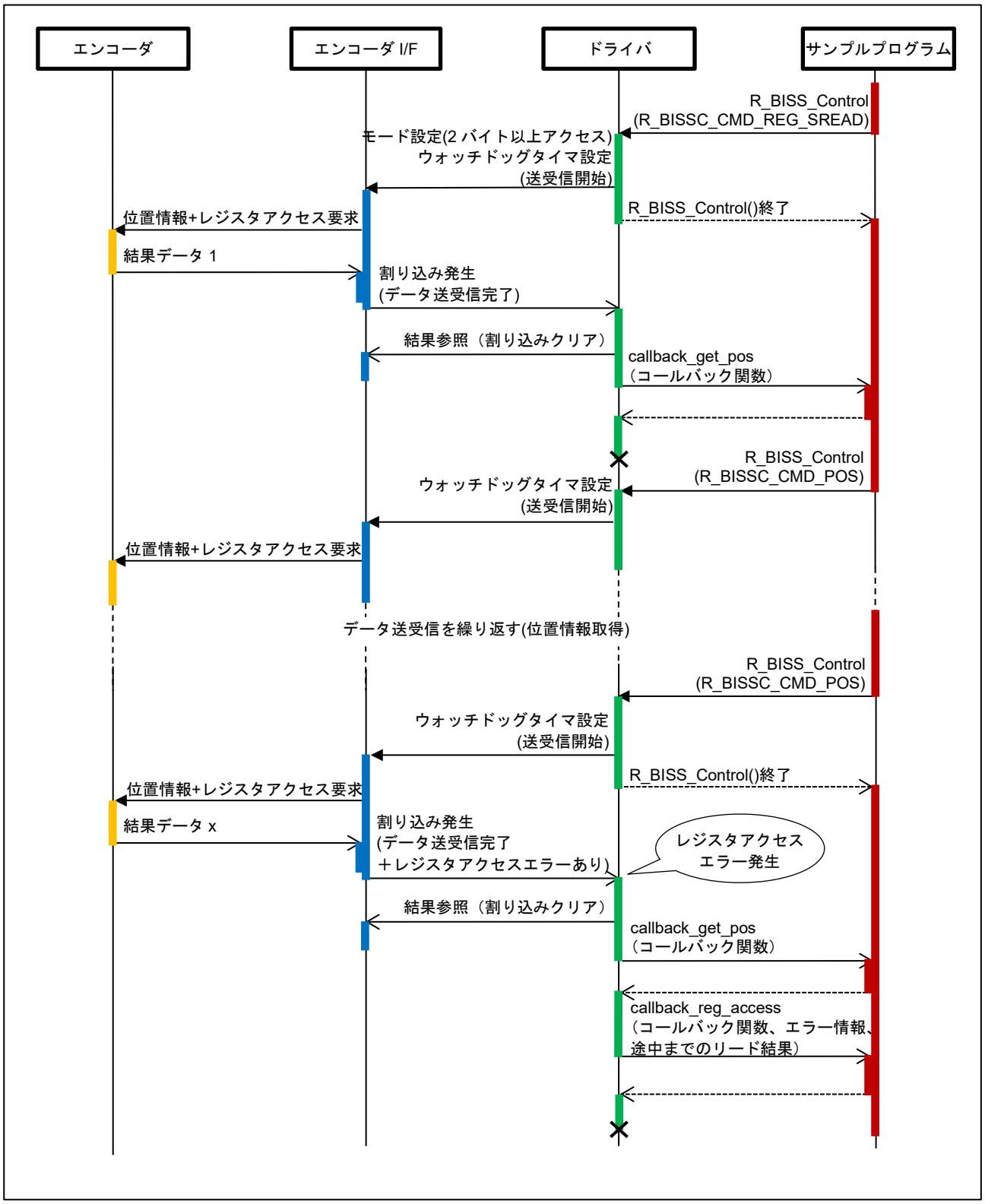


図 4-22　連続レジスタアクセス（リード・ライト）シーケンス図（途中でエラー発生）

(6)　連続レジスタアクセス（リード・ライト）停止シーケンス

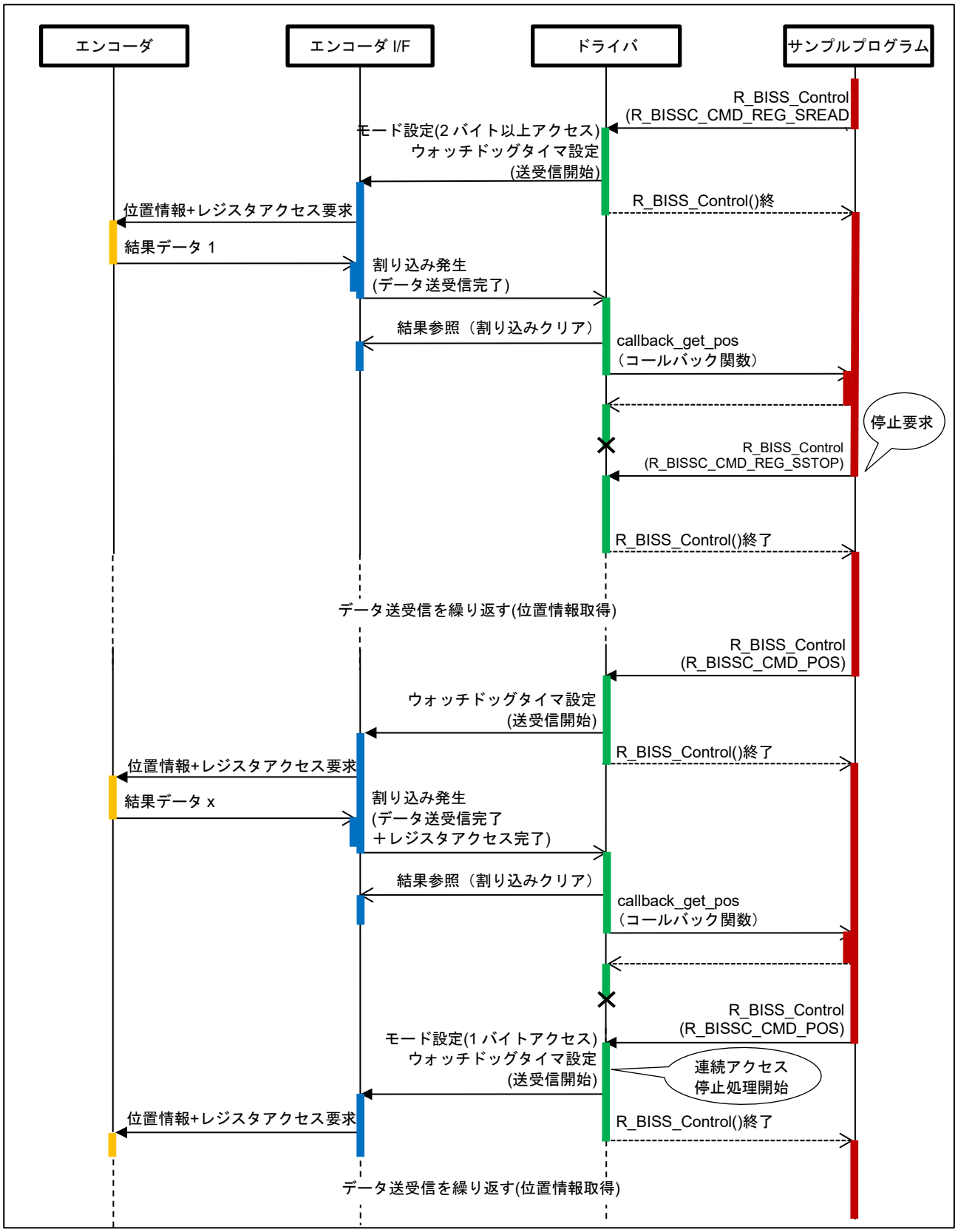


図 4-23　連続レジスタアクセス（リード・ライト）停止シーケンス図（1/2）

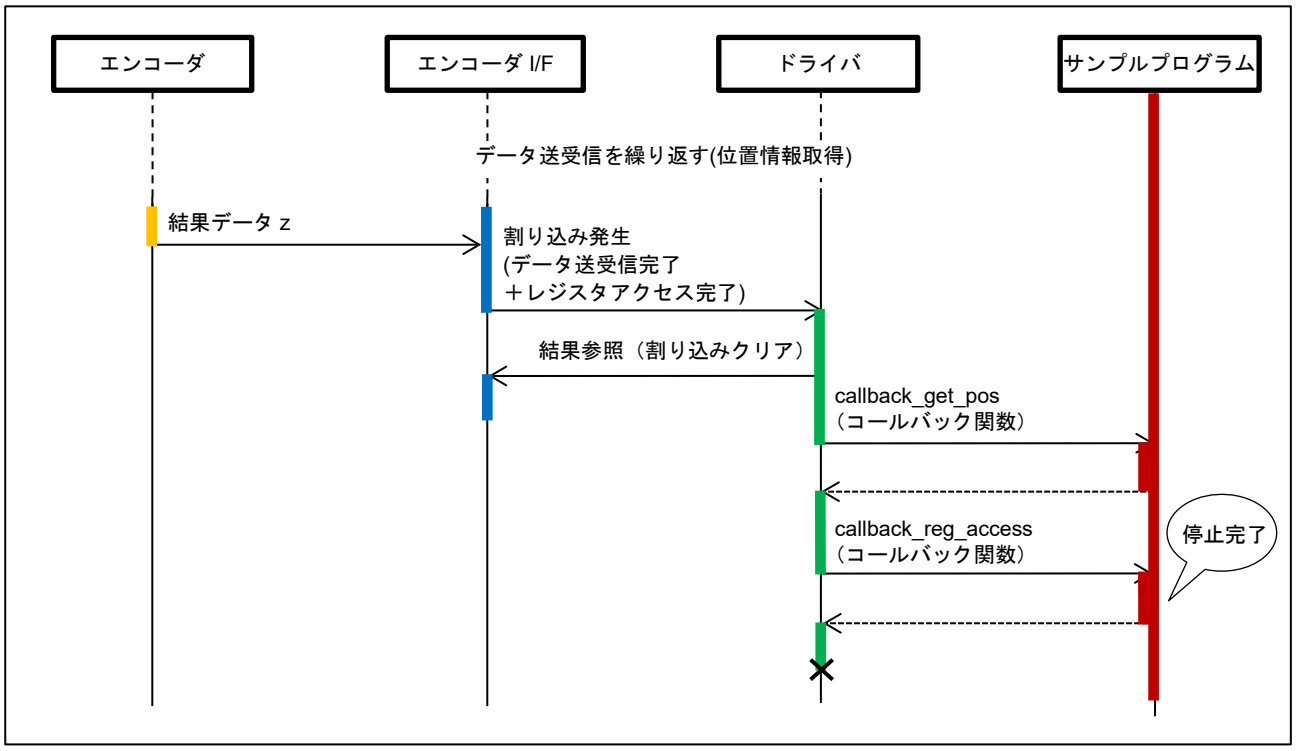


図 4-24　連続レジスタアクセス（リード・ライト）停止シーケンス図（2/2）

(7)　ELC タイマモード 位置情報取得動作のシーケンス

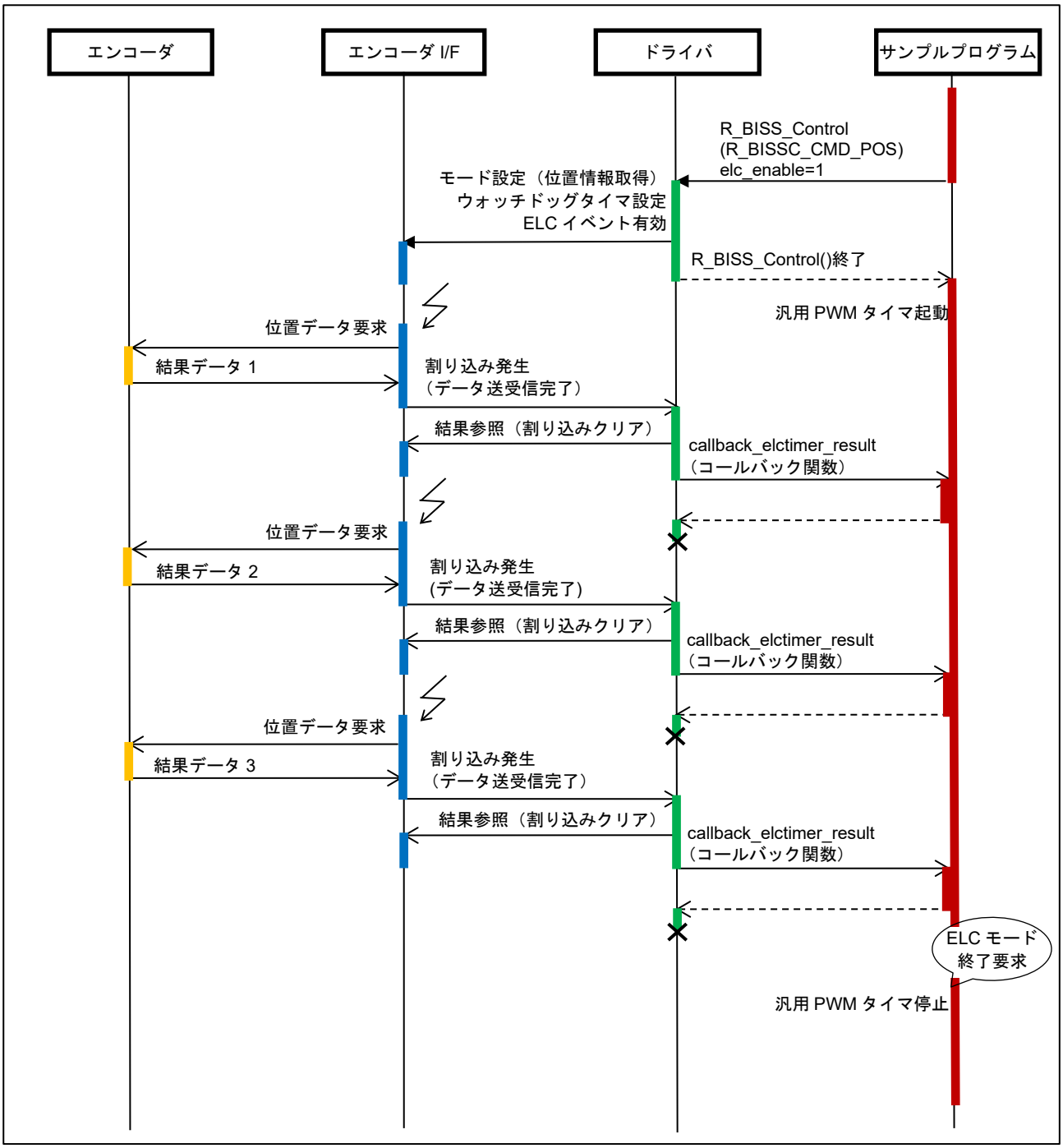


図 4-25　ELC タイマモード位置情報取得動作のシーケンス図

(8) 停止シーケンス

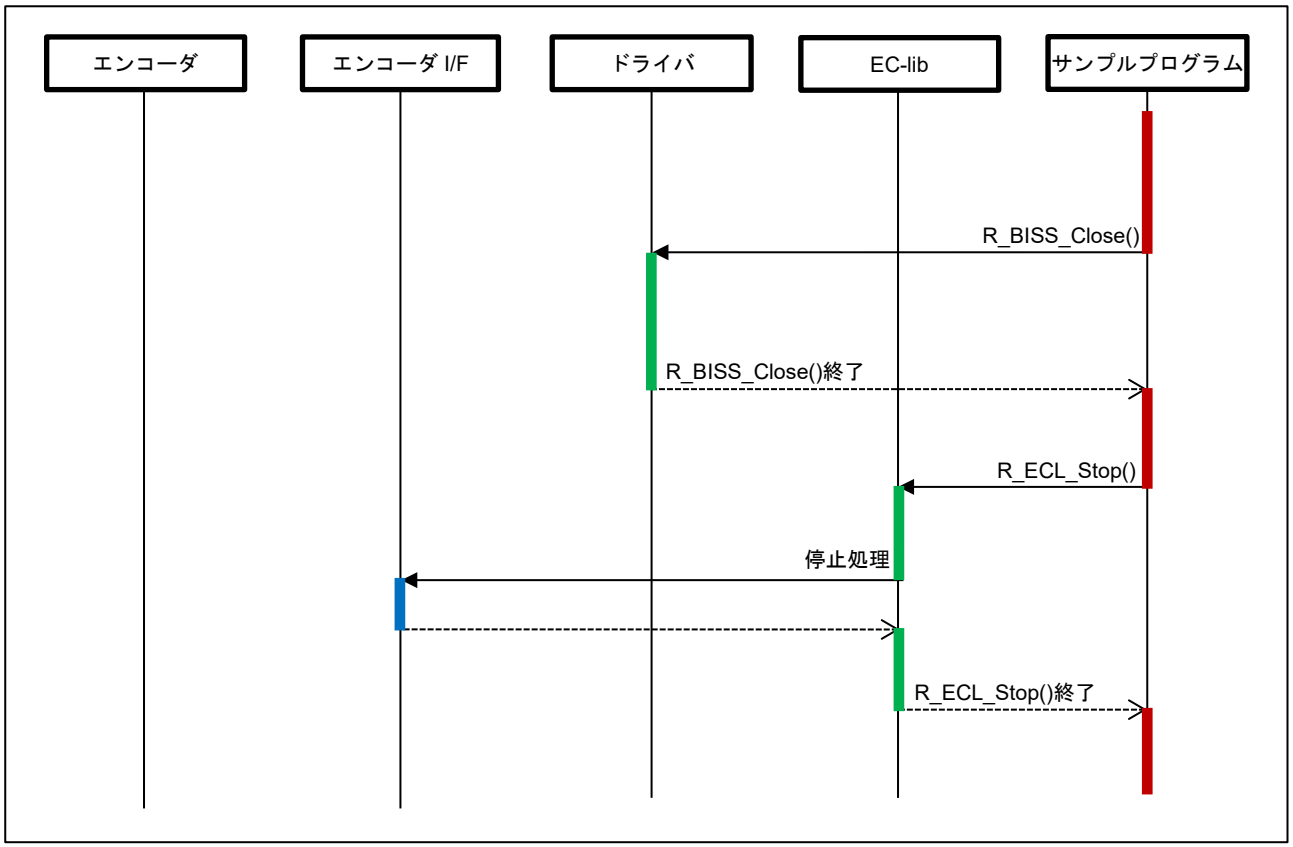


図 4-26 停止シーケンス図

4.11.7 コンソールコマンド

本サンプルプログラムは Hengstler 社製アブソリュートエンコーダ「AD36/1219AF.0CBEB」に対応しています。コンソールから入力可能なコマンドは以下となります。

表 4.11 コンソールコマンド一覧

コマンド	内容
pos	位置情報を 1 回取得します。
read *AA BB*	レジスタアドレス「*AA*」から「*BB*」バイト分、データをリードします。[注]
write *AA BB CCCCCCCC*	レジスタアドレス「*AA*」から「*BB*」バイト分、「*CCCCCCCC*」データをライトします。[注]
elctimer *VAL*	ELC イベント入力トリガ動作として、位置情報をタイマ周期で連続取得します。*VAL* は us 単位(最大 6990us)の 10 進数で指定します。連続取得を停止する場合は、「elcstop」コマンドを入力してください。
elcstop	位置情報の連続取得を停止します。
exit	プログラムを終了します。

【注】 「*AA*」、「*BB*」、「*CCCCCCCC*」は 16 進数として認識します。
例）*BB* = “24”の場合、0x24（36 バイト）要求

(1) サンプルプログラム実行

プログラムを実行すると、コマンドプロンプトが表示されます。”biss>”に続けてコマンドを入力してください。

```
BiSS sample program start
biss>
```

(2) コマンド実行例

pos コマンドを実行した例です。エンコーダからの応答に基づき、位置情報やエラー情報が表示されます。

```
biss>pos
get position
  result
    multi turn data:      2807
    single turn data:     24917
    alarm err:           0
    warning err:          0
biss>
```

5. サンプルコード

サンプルコードは、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。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
Rev.1.00	Jun.28.22	-	初版発行
Rev.2.00	Jun 20.24	4 17	使用ボードの表記を更新 固定幅整数の定義場所に関する記載を削除
Rev.3.00	Oct 10.25	1, 3, 4 4 7 - 12 49	商標の説明の記載方法を更新 使用ボード名を修正 関数仕様の説明のヘッダ欄を更新 コマンド実行例を追加
Rev.4.00	Feb 27.26	13 - 14 16 7 - 38	「4.5 ユーザー定義関数仕様」のヘッダ部分を修正 BiSS-C I/F チャネル 0, 1 用送受信割り込みの優先度を 11 に修正 ポインタ変数のプレフィクスを”p_”に変更 (例. pinfo -> p_info)

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。回路、ソフトウェアおよびこれらに関連する情報を使用する場合、お客様の責任において、お客様の機器・システムを設計ください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品または本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
6. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
10. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
11. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
12. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
13. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
14. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

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

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

(Rev.5.0-1　2020.10)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

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

www.renesas.com/contact/

商標について

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