RZ/T2L グループ

A-format 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/T2L の Encoder Interface を使用して、A-format™ 通信プロトコル仕様 Version 2.0（以下 A-format™ V2 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。

プログラムの特徴を以下に示します。

・A-format™ V2 のコマンドコードに対応
・A-format™ V2 仕様に準拠したエンコーダ(Nikon 社製 MAR-M50A)から、角度情報等を取得

動作確認デバイス

RZ/T2L

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
A-format 通信コントローラ(AFMT)	A-format V2 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)	AFMT 割り込み制御
汎用 PWM タイマ(GPT) ユニット 0 チャネル 0	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 ユニット 0 チャネル 0 が出力するイベントと AFMT をリンク
シリアル通信インターフェース(SCI) UART	SCI の調歩同期式 I/F を使用し、USB インターフェースによる 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

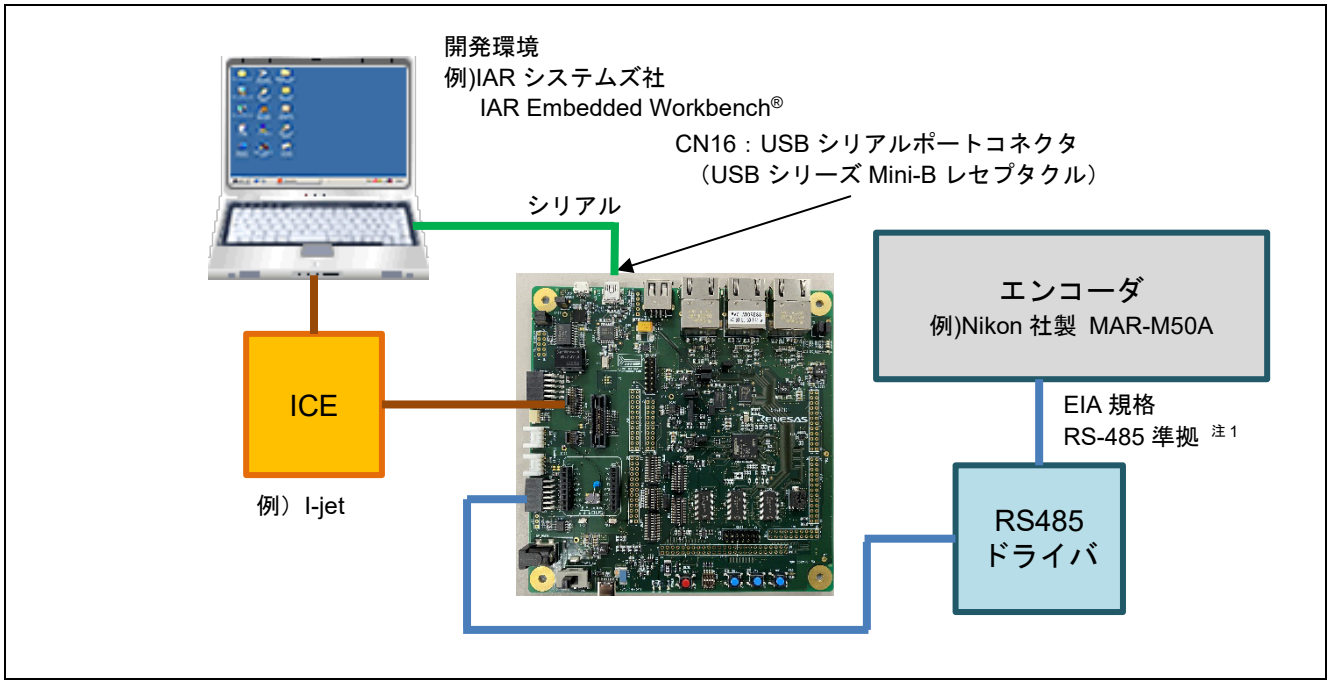

図 1-1 動作環境

【注】 1. 送受信可能なケーブル長は、エンコーダの製造元に問い合わせてください。

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/T2L 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 1	IAR システムズ製　IAR Embedded Workbench® for Arm® RENESAS 製　e² studio
使用ボード	RSK+RZT2L (RTK9RZT2L0C00000BJ)
使用デバイス（ボード上で使用する機能）	なし

【注】 1. 統合開発環境のバージョンは、RZ/T2L グループ Encoder I/F A-format 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/T2L グループ・ユーザーズマニュアル ハードウェア編に記載しています。

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	I/O ポート	入出力	内容
AFMT0	ENCIFDI0 (SDAT)	P02_2	入力	データ入力端子
	ENCIFDO0 (REQ)	P02_3	出力	データ出力端子
	ENCIFOE0 (D/R)	P01_7	出力	ドライブ/レシーブ制御端子
AFMT1	ENCIFDI1 (SDAT)	P10_1	入力	データ入力端子
	ENCIFDO1 (REQ)	P10_0	出力	データ出力端子
	ENCIFOE1 (D/R)	P09_7	出力	ドライブ/レシーブ制御端子

4. ソフトウェア説明

4.1 A-format ドライバ機能

A-format ドライバの機能は以下です。

1. 初期設定
2. コマンドコードの送信
3. 受信データの取得

4.2 ファイル構成

ファイル構成は、RZ/T2L グループ Encoder I/F A-format 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
A-format ドライバ API 関数	R_A_AS_Open	7
	R_A_AS_Close	7
	R_A_AS_GetVersion	8
	R_A_AS_Control	8
ユーザー定義関数	a_as_txerr_callback	12
	a_as_rxset_callback	12
	a_as_rxend_callback	13
	a_as_elctimer_callback	13
割り込みハンドラ	a_as0_int_isr	15
	a_as1_int_isr	15
	a_as_err_isr	15

4.4 API 関数仕様

4.4.1 R_A_AS_Open

R_A_AS_Open	
概　要	エンコーダ制御の開始
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Open(const int32_t id, r_a_as_info_t* p_info);
説　明	AFMT の初期設定を行います。 1 A-format Interface のパワーダウン設定解除 2 エンコーダインタフェースの設定 3 通信パラメータ(BRSEL レジスタ)の設定 4 割り込みイネーブルの設定
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可 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_a_as_info_t のアドレスを指定してください。
リターン値	R_A_AS_SUCCESS：正常終了 R_A_AS_ERR_INVALID_ARG：異常終了(id, p_info に指定した r_a_as_info_t 構造体のメンバ変数が規定されていない値) R_A_AS_ERR_ACCESS：異常終了（既に open されています）
注　意	本関数内でエンコーダ I/F の設定を行います。 エンコーダの電源投入後に本関数を実行した場合は、本関数の実行後に CDF8 を連続して 8 回エンコーダに送信し、ステータスフラグをクリアしてください。 コールバック関数内で、本 API 関数を実行することは禁止します。

4.4.2 R_A_AS_Close

R_A_AS_Close	
概　要	エンコーダの制御を終了
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Close(const int32_t id);
説　明	指定されたチャネルのエンコーダの制御を終了します。
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可
リターン値	R_A_AS_SUCCESS：正常終了 R_A_AS_ERR_INVALID_ARG：異常終了(id に指定した値が規定されていない値) R_A_AS_ERR_ACCESS：異常終了(リクエストを送信中です。)
注　意	コールバック関数内で、本 API 関数を実行することは禁止します。

4.4.3 R_A_AS_GetVersion

R_A_AS_GetVersion	
概　要	エンコーダ IF ドライバのバージョンを取得
ヘッダ	r_a_as_rzt2_if.h
宣　言	uint32_t R_A_AS_GetVersion(const r_a_as_type_t type);
説　明	A-format ドライバのバージョンを取得します。
引　数	type　　R_A_AS_A_FORMAT を指定してください。
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。
注　意	コールバック関数内で、本 API 関数を実行することは禁止します。

4.4.4 R_A_AS_Control

R_A_AS_Control	
概　要	エンコーダの制御
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Control(const int32_t id, const r_a_as_cmd_t cmd, void *const p_buf);
説　明	引数 cmd を使ってエンコーダを制御します。 制御コマンドの動作は 4.4.4(1) 制御コマンドを参照してください。
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：コマンド 内容は「表 4-9　R_A_AS_Control 関数の制御コマンド」を参照してください。 p_buf　：各 cmd に対応する引数
リターン値	R_A_AS_SUCCESS：正常終了 R_A_AS_ERR_INVALID_ARG：異常終了（id, cmd が規定されていない値） その他リターン値は 4.4.4(1) 制御コマンドを参照してください。
注　意	本関数実行前に、必ず R_A_AS_Open を実行してください。 コールバック関数内で、本 API 関数を実行することは禁止します。

(1) 制御コマンド

(a) R_A_AS_CMD_SET_PARAM

R_A_AS_CMD_SET_PARAM	
概　要	リクエスト情報を設定
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Control(const int32_t id, const r_a_as_cmd_t cmd, void *const p_buf);
説　明	リクエスト情報を設定します。
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_A_AS_CMD_SET_PARAM を指定します。 p_buf　：リクエスト情報 リクエスト情報を格納した r_a_as_req_t 構造体のポインタを指定します。詳細は「4.10.1(2) r_a_as_req_t」を参照してください。
リターン値	R_A_AS_SUCCESS：正常終了 R_A_AS_ERR_INVALID_ARG：異常終了（id が不正値、p_buf が NULL、p_buf に指定された r_a_as_req_t 構造体のメンバ変数が規定されていない値） R_A_AS_ERR_ACCESS：異常終了（該当チャネルが開始されていません）
注意	本制御コマンドは、リクエスト情報の設定のみを行います。 エンコーダへコマンド送信するには、以下の制御コマンドをご使用ください。 ・R_A_AS_CMD_TX_TRG ・R_A_AS_CMD_TX_ELC コールバック関数内で、本制御コマンドを実行することは禁止します。

(b) R_A_AS_CMD_ELC_DISABLE

R_A_AS_CMD_ELC_DISABLE	
概　要	ELC イベント入力トリガの無効
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Control(const int32_t id, const r_a_as_cmd_t cmd, void *const p_buf);
説　明	ELC イベント入力トリガを無効にします。
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_A_AS_CMD_ELC_DISABLE 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_A_AS_SUCCESS：ELC イベント入力トリガを無効にしました。 R_A_AS_ERR_INVALID_ARG：異常終了（id が不正値） R_A_AS_ERR_ACCESS：異常終了（ELC イベント入力トリガ動作中ではない、または、該当チャネルが開始されていません）
注　意	コールバック関数内で、本制御コマンドを実行することは禁止します。

(c) R_A_AS_CMD_TX_TRG

R_A_AS_CMD_TX_TRG	
概　要	エンコーダへのリクエスト送信を開始
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Control(const int32_t id, const r_a_as_cmd_t cmd, void *const p_buf);
説　明	エンコーダへのリクエスト送信を開始します。 通常受信時はリクエスト送信 1 回に対して、割り込み許可中の割り込み要因に対応したコールバック関数が 1 回ずつコールされます。コールバック関数の詳細は、「4.5.1 a_as_txerr_callback」～「4.5.3 a_as_rxend_callback」を参照してください。
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_A_AS_CMD_TX_TRG 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_A_AS_SUCCESS：正常終了 R_A_AS_ERR_INVALID_ARG：異常終了（id が不正値） R_A_AS_ERR_BUSY：異常終了（送信処理中） R_A_AS_ERR_ACCESS：異常終了（該当チャネルが開始されていません）
注　意	本制御コマンドは、エンコーダへのリクエスト送信の開始のみを行います。 制御コマンド R_A_AS_CMD_SET_PARAM でリクエスト情報を設定してからご使用ください。 コールバック関数内で、本制御コマンドを実行することは禁止します。

(d) R_A_AS_CMD_TX_ELC

R_A_AS_CMD_TX_ELC	
概　要	ELC イベント入力トリガによるエンコーダへのリクエスト送信を開始
ヘッダ	r_a_as_rzt2_if.h
宣　言	int32_t R_A_AS_Control(const int32_t id, const r_a_as_cmd_t cmd, void *const p_buf);
説　明	ELC イベント入力トリガを許可し、エンコーダへのリクエスト送信を開始します。 通常受信時はリクエスト送信 1 回に対して、割り込み許可中の割り込み要因に対応したコールバック関数が 1 回ずつコールされます。コールバック関数の詳細は、「4.5.1 a_as_txerr_callback」～「4.5.3 a_as_rxend_callback」を参照してください。 ELC イベント入力トリガ動作中に本制御コマンドを実行した場合、R_A_AS_ERR_BUSY が発生します。
引　数	id　：使用する ID を指定します。(r_a_as_rzt2_dat.h で定義されています。) R_A_AS0_ID　：チャネル 0 を指定 R_A_AS1_ID　：チャネル 1 を指定 上記以外　：設定不可 cmd　：R_A_AS_CMD_TX_ELC 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_A_AS_SUCCESS：正常終了 R_A_AS_ERR_INVALID_ARG：異常終了（id が不正値） R_A_AS_ERR_BUSY：異常終了（送信処理中、ELC イベント入力トリガ動作中） R_A_AS_ERR_ACCESS：異常終了（該当チャネルが開始されていません）
注　意	本制御コマンドは、エンコーダへのリクエスト送信の開始のみを行います。 制御コマンド R_A_AS_CMD_SET_PARAM でリクエスト情報を設定してからご使用ください。 コールバック関数内で、本制御コマンドを実行することは禁止します。

4.5 ユーザー定義関数仕様

4.5.1 a_as_txerr_callback

a_as_txerr_callback	
概　要	通常受信でタイムアウトエラー発生時の送受信結果を通知
ヘッダ	-
宣　言	void a_as_txerr_callback (r_a_as_result_t * p_result);
説　明	R_A_AS_Control(R_A_AS_CMD_SET_PARAM) 関数で登録したコールバック関数です。通常受信時の送受信結果を通知します。タイムアウトエラー割り込み(AFMTi_TMOUT)が発生した場合にコールされます。本関数を処理後に、a_as_rxend_callback()関数もコールされます。 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　：送受信結果 構造体 r_a_as_result_t で宣言した送受信結果が格納されている配列のポインタです。 配列の内容は表 4-2　配列の内容と送受信結果の対応を参照してください。 R_A_AS_Control(R_A_AS_CMD_SET_PARAM) 関数で指定したエンコーダアドレスに対応した送受信結果を更新します。 送受信結果の詳細は、表 4-3　送受信結果を参照してください。
リターン値	なし
注　意	ELC イベント入力トリガが有効の場合、タイマインターバル時間内に送受信結果を取得してください。 エンコーダからの応答タイミングによっては、R_A_AS_Close 関数や R_A_AS_Control (R_A_AS_CMD_ELC_DISABLE) 関数を実行した後でも、本コールバック関数が呼ばれることがあります。

4.5.2 a_as_rxset_callback

a_as_rxset_callback	
概　要	通常受信で受信データ設定完了時の送受信結果を通知
ヘッダ	-
宣　言	void a_as_rxset_callback(r_a_as_result_t * p_result);
説　明	R_A_AS_Control (R_A_AS_CMD_SET_PARAM)関数で登録したコールバック関数です。通常受信時の送受信結果を通知します。受信完了割り込み(AFMTi_EOF)が発生した場合にコールされます。本関数を処理後に、a_as_rxend_callback()関数もコールされます。 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　：送受信結果 構造体 r_a_as_result_t で宣言した送受信結果が格納されている配列のポインタです。 配列の内容は表 4-2　配列の内容と送受信結果の対応を参照してください。 R_A_AS_Control (R_A_AS_CMD_SET_PARAM) 関数で指定したエンコーダアドレスに対応した送受信結果を更新します。 送受信結果の詳細は、表 4-3　送受信結果を参照してください。
リターン値	なし
注　意	ELC イベント入力トリガが有効の場合、タイマインターバル時間内に送受信結果を取得してください。 エンコーダからの応答タイミングによっては、R_A_AS_Close 関数や、R_A_AS_Control (R_A_AS_CMD_ELC_DISABLE) 関数を実行した後でも、本コールバック関数が呼ばれることがあります。

4.5.3 a_as_rxend_callback

a_as_rxend_callback	
概　要	通常受信でデータ送受信完了時の送受信結果を通知
ヘッダ	-
宣　言	void a_as_rxend_callback(r_a_as_result_t * p_result);
説　明	R_A_AS_Control(R_A_AS_CMD_SET_PARAM) 関数で登録したコールバック関数です。 通常受信時の送受信結果を通知します。タイムアウトエラー割り込み(PERI_ERR0)や受信完了割り込み(AFMTi_EOF)が発生した場合に、a_as_txerr_callback()関数やa_as_rxset_callback()関数に続いてコールされます。 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　：送受信結果 構造体 r_a_as_result_t で宣言した送受信結果が格納されている配列のポインタです。 配列の内容は表 4-2　配列の内容と送受信結果の対応を参照してください。 R_A_AS_Control(R_A_AS_CMD_SET_PARAM) 関数で指定したエンコーダアドレスに対応した送受信結果を更新します。 送受信結果の詳細は、表 4-3　送受信結果を参照してください。
リターン値	なし
注　意	ELC イベント入力トリガが有効の場合、タイマインターバル時間内に送受信結果を取得してください。 エンコーダからの応答タイミングによっては、R_A_AS_Close 関数や、R_A_AS_Control(R_A_AS_CMD_ELC_DISABLE) 関数を実行した後でも、本コールバック関数が呼ばれることがあります。

4.5.4 a_as_elctimer_callback

a_as_elctimer_callback	
概　要	ELC イベント入力による連続受信でデータ送受信結果を通知
ヘッダ	-
宣　言	void a_as_elctimer_callback(r_a_as_result_t * p_result);
説　明	R_A_AS_Control(R_A_AS_CMD_SET_PARAM) 関数で登録したコールバック関数です。 通常受信時の送受信結果を通知します。受信完了割り込み(AFMTi_EOF)が発生するたびにコールされます。 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　：送受信結果 構造体 r_a_as_result_t で宣言した送受信結果が格納されている配列のポインタです。 配列の内容は表 4-2　配列の内容と送受信結果の対応を参照してください。 R_A_AS_Control(R_A_AS_CMD_SET_PARAM) 関数で指定したエンコーダアドレスに対応した送受信結果を更新します。 送受信結果の詳細は、表 4-3　送受信結果を参照してください。
リターン値	なし
注　意	ELC イベント入力トリガが有効の場合、タイマインターバル時間内に送受信結果を取得してください。 エンコーダからの応答タイミングによっては、R_A_AS_Close 関数や、R_A_AS_Control(R_A_AS_CMD_ELC_DISABLE) 関数を実行した後でも、本コールバック関数が呼ばれることがあります。

表 4-2 配列の内容と送受信結果の対応

配列番号	内容
p_result[0]	エンコーダ区分 ENC1 の送受信結果
p_result[1]	エンコーダ区分 ENC2 の送受信結果
p_result[2]	エンコーダ区分 ENC3 の送受信結果
p_result[3]	エンコーダ区分 ENC4 の送受信結果
p_result[4]	エンコーダ区分 ENC5 の送受信結果
p_result[5]	エンコーダ区分 ENC6 の送受信結果
p_result[6]	エンコーダ区分 ENC7 の送受信結果
p_result[7]	エンコーダ区分 ENC8 の送受信結果

表 4-3 送受信結果

割り込み要因	送受信結果(p_result のメンバ変数)		
	result	data	status
タイムアウトエラー (AFMTi_TMOUT) 注 1	コールバック関数内のみ有効	無効	コールバック関数内のみ有効
データ受信完了 (AFMTi_EOF) 注 1	コールバック関数内のみ有効	有効 注 2	コールバック関数内のみ有効

【注】 1. i = 0, 1

2. ELC イベ.ント入力トリガが無効の場合、次のリクエスト送信までデータ受信結果は有効です。
ELC イベント入力トリガが有効の場合、次の AFMTi_EOF 割り込みが発生するまでデータ受信結果は有効です。

4.6 割り込みハンドラ

4.6.1 a_as0_int_isr

a_as0_int_isr	
概　要	AFMT0_EOF 割り込みの割り込みハンドラ
ヘッダ	-
宣　言	static void a_as0_int_isr(void);
説　明	A_AS Ch0 の受信完了割り込みに対する割り込みハンドラです。
引　数	なし
リターン値	なし

4.6.2 a_as1_int_isr

a_as1_int_isr	
概　要	AFMT1_EOF 割り込みの割り込みハンドラ
ヘッダ	-
宣　言	static void a_as1_int_isr(void);
説　明	A_AS Ch1 の受信完了割り込みに対する割り込みハンドラです。
引　数	なし
リターン値	なし

4.6.3 a_as_err_isr

a_as_err_isr	
概　要	PERI_ERR0 割り込みの割り込みハンドラ
ヘッダ	-
宣　言	static void a_as_err_isr(void);
説　明	A_AS Ch0,1 のタイムアウト割り込みに対する割り込みハンドラです。
引　数	なし
リターン値	なし

4.7 使用割り込み一覧

表 4-4 に A-format ドライバで使用する割り込みを示します。

表 4-4　A-format ドライバで使用する割り込み

割り込み	ID	概要
AFMT0_EOF	267	Ch0 のデータ受信完了で割り込みが発生します。
AFMT1_EOF	268	Ch1 のデータ受信完了で割り込みが発生します。
PERI_ERR0	388	Ch0, Ch1 のタイムアウトで割り込みが発生します。

4.8 定数/エラーコード一覧

表 4-5 に定数/エラーコード定義表の一覧を示します。各定義については、それぞれの表を参照してください。

表 4-5 定数/エラーコード定義表の一覧

表番号	内容
表 4-6	A-format ドライバで使用するユーザー定義の定数(r_a_as_rzt2_config.h)
表 4-7	ドライバの種類
表 4-8	A_AS とエンコーダの接続方式
表 4-9	R_A_AS_Control 関数の制御コマンド
表 4-10	ビットレート
表 4-11	エンコーダアドレス
表 4-12	コマンド
表 4-13	エラーコード

表 4-6 A-format ドライバで使用するユーザー定義の定数(r_a_as_rzt2_config.h)

定数名	設定値	内容
R_AFMT_T2_ONE_2500KBPS	0x0000	ビットレートが 2.5 Mbps、接続方式が 1 対 1 の場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_ONE_4MBPS	0x0000	ビットレートが 4 Mbps、接続方式が 1 対 1 の場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_ONE_6670KBPS	0x0000	ビットレートが 6.67 Mbps、接続方式が 1 対 1 の場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_ONE_8MBPS	0x0000	ビットレートが 8 Mbps、接続方式が 1 対 1 の場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_BUS_2500KBPS	0x001A	ビットレートが 2.5 Mbps、接続方式がバスの場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_BUS_4MBPS	0x0010	ビットレートが 4 Mbps、接続方式がバスの場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_BUS_6670KBPS	0x0009	ビットレートが 6.67 Mbps、接続方式がバスの場合の BRSEL レジスタ TM ビット設定値[注]
R_AFMT_T2_BUS_8MBPS	0x0008	ビットレートが 8 Mbps、接続方式がバスの場合の BRSEL レジスタ TM ビット設定値[注]

【注】 サンプルプログラムでは推奨設定値を各レジスタに設定しています。

表 4-7 ドライバの種類

定数名	設定値	内容
R_A_AS_A_FORMAT	0	A-format ドライバを指定

表 4-8 A_AS とエンコーダの接続方式

定数名	設定値	内容
R_A_AS_ONE_FOR_ONE	0	1 対 1 接続
R_A_AS_BUS	1	バス接続

表 4-9 R_A_AS_Control 関数の制御コマンド

定数名	設定値	内容
R_A_AS_CMD_SET_PARAM	0xAF000000	リクエスト情報を設定
R_A_AS_CMD_ELC_DISABLE	0xAF000002	ELC イベント入力トリガを無効化
R_A_AS_CMD_TX_TRG	0xAF000003	トリガによるコマンド送信を開始
R_A_AS_CMD_TX_ELC	0xAF000005	ELC イベント入力トリガによるコマンド送信を開始

表 4-10 ビットレート

定数名	設定値	内容
R_A_AS_2500KBPS	0	2.5 Mbps
R_A_AS_4MBPS	1	4 Mbps
R_A_AS_6670KBPS	2	6.67 Mbps
R_A_AS_8MBPS	3	8 Mbps

表 4-11 エンコーダアドレス

定数名	設定値	内容
R_A_AS_ECN1	0	エンコーダ区分 ENC1 のエンコーダアドレス
R_A_AS_ECN2	1	エンコーダ区分 ENC2 のエンコーダアドレス
R_A_AS_ECN3	2	エンコーダ区分 ENC3 のエンコーダアドレス
R_A_AS_ECN4	3	エンコーダ区分 ENC4 のエンコーダアドレス
R_A_AS_ECN5	4	エンコーダ区分 ENC5 のエンコーダアドレス
R_A_AS_ECN6	5	エンコーダ区分 ENC6 のエンコーダアドレス
R_A_AS_ECN7	6	エンコーダ区分 ENC7 のエンコーダアドレス
R_A_AS_ECN8	7	エンコーダ区分 ENC8 のエンコーダアドレス

表 4-12 コマンド

定数名	設定値	内容
R_A_AS_CDF0	0	コマンドデータフレーム CDF0 の定義です。
R_A_AS_CDF1	1	コマンドデータフレーム CDF1 の定義です。
R_A_AS_CDF2	2	コマンドデータフレーム CDF2 の定義です。
R_A_AS_CDF3	3	コマンドデータフレーム CDF3 の定義です。
R_A_AS_CDF4	4	コマンドデータフレーム CDF4 の定義です。
R_A_AS_CDF5	5	コマンドデータフレーム CDF5 の定義です。
R_A_AS_CDF6	6	コマンドデータフレーム CDF6 の定義です。
R_A_AS_CDF7	7	コマンドデータフレーム CDF7 の定義です。
R_A_AS_CDF8	8	コマンドデータフレーム CDF8 の定義です。
R_A_AS_CDF9	9	コマンドデータフレーム CDF9 の定義です。
R_A_AS_CDF10	10	コマンドデータフレーム CDF10 の定義です。
R_A_AS_CDF11	11	コマンドデータフレーム CDF11 の定義です。
R_A_AS_CDF12	12	コマンドデータフレーム CDF12 の定義です。
R_A_AS_CDF13	13	コマンドデータフレーム CDF13 の定義です。
R_A_AS_CDF14	14	コマンドデータフレーム CDF14 の定義です。
R_A_AS_CDF15	15	コマンドデータフレーム CDF15 の定義です。
R_A_AS_CDF16	16	コマンドデータフレーム CDF16 の定義です。
R_A_AS_CDF17	17	コマンドデータフレーム CDF17 の定義です。
R_A_AS_CDF18	18	コマンドデータフレーム CDF18 の定義です。
R_A_AS_CDF19	19	コマンドデータフレーム CDF19 の定義です。
R_A_AS_CDF21	21	コマンドデータフレーム CDF21 の定義です。
R_A_AS_CDF22	22	コマンドデータフレーム CDF22 の定義です。
R_A_AS_CDF27	27	コマンドデータフレーム CDF27 の定義です。
R_A_AS_CDF28	28	コマンドデータフレーム CDF28 の定義です。
R_A_AS_CDF29	29	コマンドデータフレーム CDF29 の定義です。
R_A_AS_CDF30	30	コマンドデータフレーム CDF30 の定義です。

表 4-13 エラーコード

定数名	設定値	内容
R_A_AS_SUCCESS	0	正常終了
R_A_AS_ERR_INVALID_ARG	-1	引数異常
R_A_AS_ERR_BUSY	-2	API を実行できない状態
R_A_AS_ERR_ACCESS	-3	API の実行順序エラー

4.9 固定幅整数一覧

表 4-14 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 4-14 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

主要な構造体／共用体／列挙型の一覧を記載します。

4.10.1 構造体

(1) r_a_as_info_t

A_AS 制御部の初期化情報。

```
typedef struct
{
    uint8_t     connect;    接続方式
                            A_AS とエンコーダの接続方式を指定してください。指定する値は
                            「表 4-8　A_AS とエンコーダの接続方式」を参照してください。
                            ※本設定は BRSEL レジスタに反映されます。
    uint8_t     bitrate;    ビットレート
                            エンコーダとの通信におけるビットレートを指定してください。指
                            定する値は「表 4-10　ビットレート」を参照してください。
                            ※本設定は BRSEL レジスタに反映されます。
    uint16_t    ifmg;       マージン値
                            ※RZT2M グループ A-format ドライバ I/F との互換性のために設け
                            られています。設定値は、RZ/T2L では使われません。
} r_a_as_info_t
```

(2) r_a_as_req_t

エンコーダに送信するリクエスト情報。

```
typedef struct
{
```

uint8_t	encadr;	エンコーダアドレス エンコーダアドレスを指定してください。指定する値は「表 4-11　エンコーダアドレス」を参照してください。 この設定は COMMAND レジスタの XEA ビットに反映されます。
uint8_t	cmd;	コマンド エンコーダに送信するコマンドコードを指定してください。指定する値は「表 4-12　コマンド」を参照してください。 「表 4-12　コマンド」以外の値で 0x20 以上の値を指定すると、R_A_AS_ERR_INVALID_ARG が発生します。 接続方式によって使用できないコマンドがあります。
uint8_t	memadr;	メモリアドレス エンコーダのメモリアドレスを指定してください。 コマンドが以下の場合のみ設定してください。 cmd = R_A_AS_CDF13 cmd = R_A_AS_CDF14 ※ R_A_AS_CDF13 の場合、アクセス可能アドレス範囲は 0x00~0xFF までとなります。 R_A_AS_CDF14 の場合、アクセス可能アドレス範囲は 0x00~0xEF までとなります。
uint16_t	memdat;	メモリへ書き込むデータ メモリへ書き込むデータを指定してください。 コマンドが以下の場合のみ設定してください。 cmd = R_A_AS_CDF14
uint32_t	encid;	識別コード 識別コードの値は 24bit 長の値を指定してください。 コマンドが以下の場合のみ設定してください。 cmd = R_A_AS_CDF18 cmd = R_A_AS_CDF19
r_a_as_result_cb_t	cbadr_txerr	PERI_ERR0 割り込み発生時にコールされるコールバック関数のポインタ 詳細は「4.5.1 a_as_txerr_callback」を参照してください。注 1
r_a_as_result_cb_t	cbadr_rxset;	AFMTi_EOF (i = 0,1)割り込み発生時にコールされるコールバック関数のポインタ 詳細は「4.5.2 a_as_rxset_callback」を参照してください。注 1
r_a_as_result_cb_t	cbadr_rxend;	PERI_ERR0 割り込み発生時や AFMTi_EOF (i = 0,1)割り込み発生時に、cbadr_txerr()関数や cbadr_rxset() 関数に続いてコールされるコールバック関数のポインタ 詳細は「4.5.3 a_as_rxend_callback」を参照してください。注 1
bool	pre;	ELC イベント入力トリガによるデータ送受信中に、リクエスト情報を設定する場合は、true にしてください。 ELC イベント入力トリガによるデータ送受信中以外で、リクエスト情報を設定する場合は、false にしてください。 (true：ELC イベント入力トリガによるデータ送受信中にリクエスト設定)

```
} r_a_as_req_t
```

【注】 1. NULL を指定するとコールバックが発生しません。

(3) r_a_as_result_t

通常受信時の送受信結果

```
typedef struct
{
    r_a_as_req_err_t    result;     送受信結果
                                    詳細は列挙型「r_a_as_req_err_t」参照してください。
    r_a_as_data_t       data;       受信データ
                                    詳細は構造体「r_a_as_data_t」参照してください。
    r_a_as_status_t     status;     A_AS のステータス
                                    詳細は構造体「r_a_as_status_t」参照してください。
} r_a_as_result_t
```

(4) r_a_as_data_t

通常受信時の受信データ

```
typedef struct
{
    uint32_t            rxi;        ENCnRXDATA0 レジスタ値
                                    ENCnRXDATA0 レジスタの値が格納されます。
    uint32_t            rxd0;       ENCnRXDATA1 レジスタ値
                                    ENCnRXDATA1 レジスタの値が格納されます。
    uint32_t            rxd1;       ENCnRXDATA2 レジスタ値
                                    ENCnRXDATA2 レジスタの値が格納されます。
} r_a_as_data_t
```

(5) r_a_as_status_t

通常受信時の A_AS のステータス

```
typedef struct
{
    bool        iwdgerr;    IF Watchdog エラー情報。タイムアウトエラーに当たる。
    bool        dwdgerr;    DF Watchdog エラー情報。タイムアウトエラーに当たる
    bool        starterr;   スタートビットエラー情報。CA[1]の FORM ステータスに当たる。
    bool        stoperr;    ストップビットエラー情報。CA[1]の FORM ステータスに当たる。
    bool        syncerr;    シンクコードエラー情報。CA[2]の SYNC ステータスに当たる。
    bool        rxeaerr;    受信エンコーダアドレスエラー情報。CA[0]の CMD ステータスに当たる。
    bool        crcerr;     CRC エラー情報を格納。CA[3]の CRC ステータスに当たる。
    bool        rxccerr;    受信コマンドコードエラー情報。CA[0]の CMD ステータスに当たる。
    bool        mdaterr;    EEPROM データエラー情報 注1
    bool        madrerr;    EEPROM アドレスエラー情報 注1
    bool        rxdzerr;    識別コードエラー情報 注1
    bool        fd1err;     固定データエラー(1) 情報 注1
    bool        fd2err;     固定データエラー(2) 情報 注1
    bool        fd3err;     固定データエラー(3) 情報 注1
    bool        fd5err;     固定データエラー(5) 情報 注1
    bool        elcin;      ELC イベント入力情報 注1
    uint8_t     txcc;       送信コマンドコード (0 : CDF0～19,23～30、1:CDF21、2:CDF22)
    bool        rxset;      受信データ設定完了情報 (true：リード可能、false：リード不可)
    bool        timer;      タイマステータス情報 注1
    bool        txerr;      送信エラー情報 注1
    bool        rxend;      受信完了情報 (true：受信した、false：受信していない)
} r_a_as_status_t
```

【注】 1. RZ/T2M グループ A-format エンコーダドライバ I/F との互換性のために設けられています。RZ/T2L では使われません。常に false です。

4.10.2 共用体

使用しません。

4.10.3 列挙型

(1) r_a_as_req_err_t

エンコーダからの受信結果。

```
typedef enum
{
    R_A_AS_REQ_SUCCESS = 0,     データ送受信正常終了
    R_A_AS_REQ_ERR              データ送受信エラー発生
                                構造体「r_a_as_status_t」のエラー情報(timer と rxend と
                                rxset と elcin と txcc 以外)が 1 つでも true の場合、データ
                                送受信エラー発生とします。
    R_A_AS_REQ_BP_ERR           FIFO が FULL
                                バイパス受信時のみ発生します。
} r_a_as_req_err_t
```

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バス接続した 1 台から 8 台までの A-format 仕様に準拠したエンコーダ(Nikon 社製 MAR-M50A)に対応しています。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リクエスト情報をエンコーダへ送信(TXTRG レジスタへの書き込み動作による通常送受信)
2) エンコーダから受信したデータをコンソールに表示
3) AFMT の ELC イベント入力トリガ機能を使用してコマンドを送受信します。(入力イベントとして GPT のイベントをリンクしています。入力イベントの設定例は、「図 4-6 a_as_elctimer 関数のフローチャート」を参照してください。)

(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。

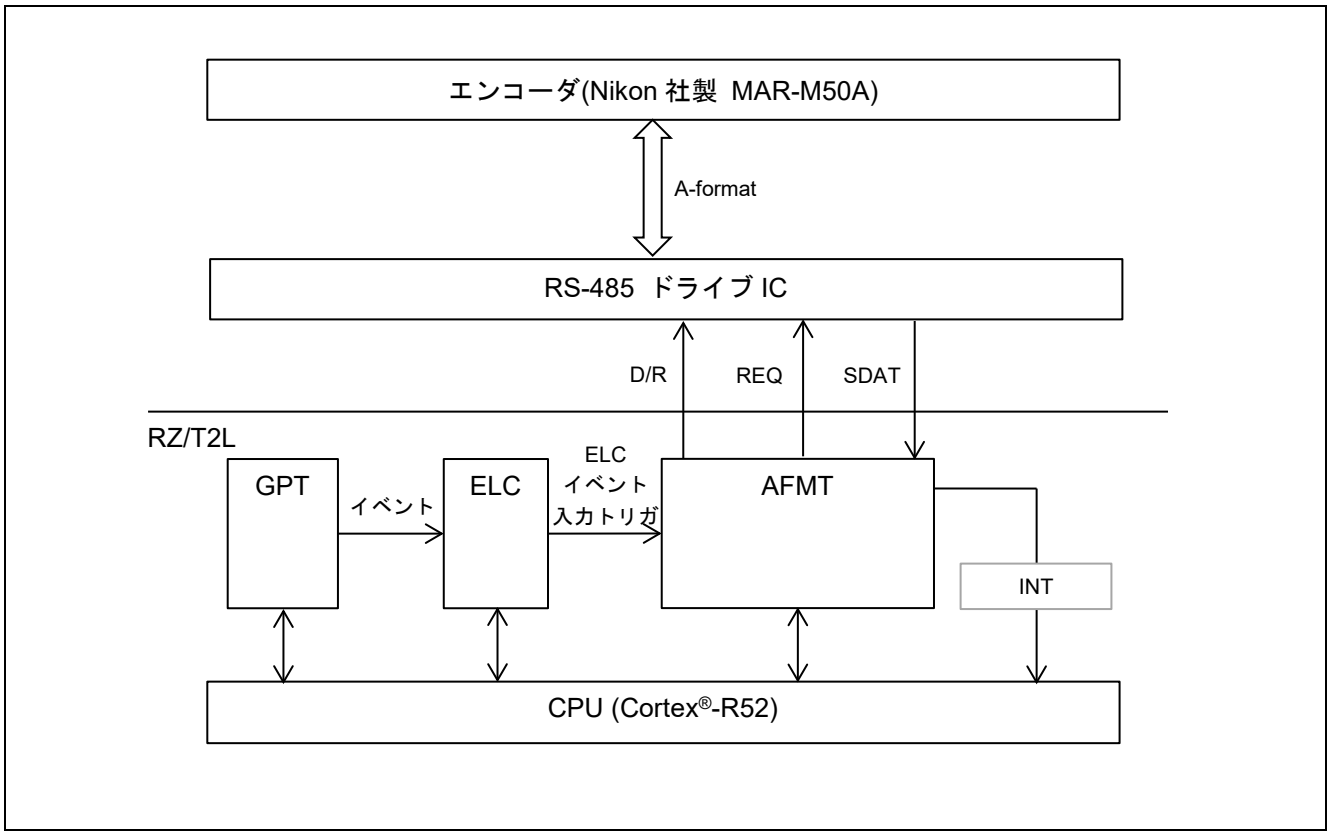

図 4-1 システムブロック図

(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。

A-format ドライバには、R_A_AS_Open 関数で構成される開始処理部、R_A_AS_Close 関数で構成される終了処理部、R_A_AS_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分（割り込みハンドラ）があります。

サンプルプログラムには、A-format ドライバを制御し、リクエスト送信を行う A-format ドライバ制御部分、データ受信結果の表示を行う結果表示部分（コールバック）があります。

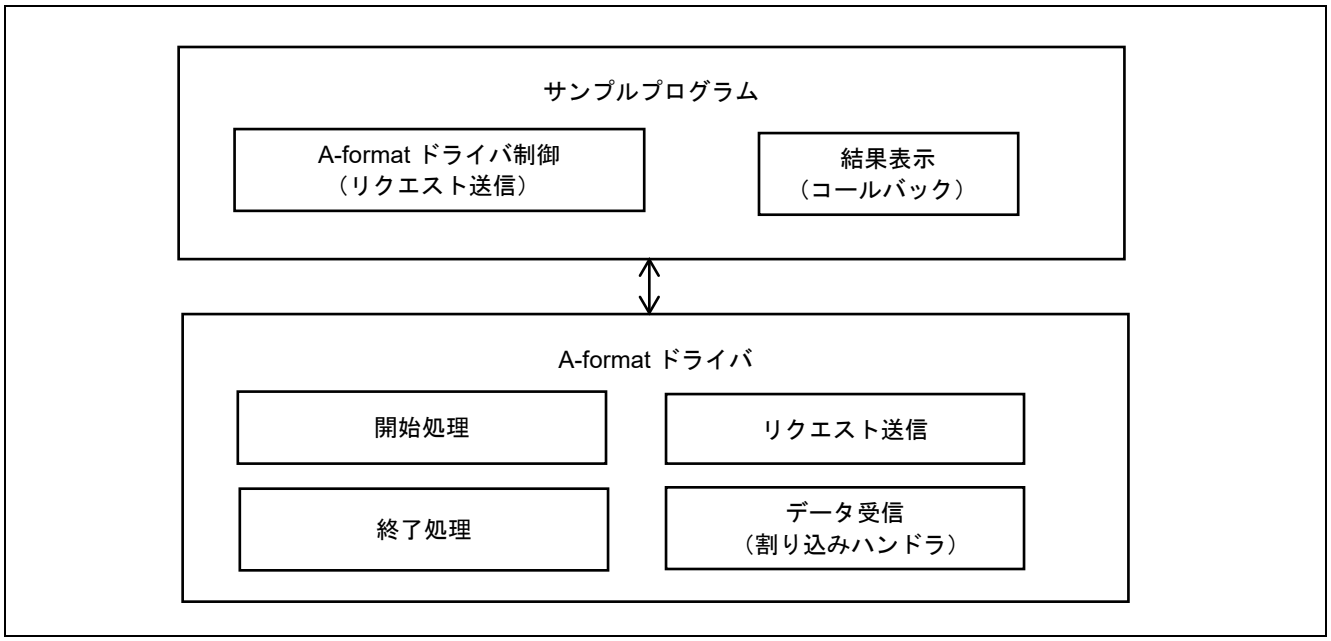

図 4-2 ソフトウェア構成図

4.11.2 サンプルプログラム関数一覧

表 4-15 に主要なサンプルプログラム関数一覧を示します

表 4-15　主要なサンプルプログラム関数一覧

関数名	ページ番号
hal_entry	28
enc_main	28
a_as_cmd_control	28
a_as_enc_init	29
a_as_req	29
a_as_elctimer	29
a_as_elcstop	30
a_as_exit	30
a_as_txerr_callback	30
a_as_rxset_callback	31
a_as_rxend_callback	31
a_as_elctimer_callback	31

4.11.3 サンプルプログラム関数仕様

(1) hal_entry

hal_entry		
概　要	A-format 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	A-format 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	A-format 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	A-format サンプルプログラムのメイン関数です。詳細は「4.11.6(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：正常終了 0 以外：異常終了（エンコーダ IF のエラーコード）	

(3) a_as_cmd_control

a_as_cmd_control		
概　要	A-format ドライバ制御関数	
ヘッダ	-	
宣　言	static void a_as_cmd_control(int32_t id);	
説　明	以下の処理を行います。 エンコーダ制御の開始 コンソールコマンドの入力処理 エンコーダ制御の終了	
引　数	id	エンコーダ ID R_A_AS0_ID: ch0 のエンコーダ ID を指定 R_A_AS1_ID: ch1 のエンコーダ ID を指定
リターン値	なし	

(4) a_as_enc_init

a_as_enc_init		
概　要	エンコーダの初期化関数	
ヘッダ	-	
宣　言	static int32_t a_as_enc_init(int32_t id);	
説　明	エンコーダを初期化します。コマンドデータフレーム CDF8 を連続して 8 回送信し、ステータスフラグをクリアします。	
引　数	id	エンコーダ ID R_A_AS0_ID: ch0 のエンコーダ ID を指定 R_A_AS1_ID: ch1 のエンコーダ ID を指定
リターン値	0：正常終了 0 以外：異常終了（エンコーダ IF のエラーコード）	

(5) a_as_req

a_as_req		
概　要	req コンソールコマンド関数	
ヘッダ	-	
宣　言	static void a_as_req(uint32_t arg_num, char_t *p_arg[]);	
説　明	コンソールコマンド req が入力された場合に実行される関数です。詳細は「4.11.6(3) a_as_req フローチャート」と「4.11.8 コンソールコマンド」を参照してください。	
引　数	arg_num	コンソールから入力された文字列の数
	*p_arg[]	コンソールから入力された文字列の先頭アドレス。
リターン値	なし	

(6) a_as_elctimer

a_as_elctimer		
概　要	elctimer コンソールコマンド関数	
ヘッダ	-	
宣　言	static void a_as_elctimer(uint32_t arg_num, char_t *p_arg[]);	
説　明	コンソールコマンド elctimer が入力された場合に実行される関数です。詳細は「4.11.6(4) a_as_elctimer フローチャート」と「4.11.8 コンソールコマンド」を参照してください。	
引　数	arg_num	コンソールから入力された文字列の数
	*p_arg[]	コンソールから入力された文字列の先頭アドレス。
リターン値	なし	

(7) a_as_elcstop

a_as_elcstop		
概　要	elcstop コンソールコマンド関数	
ヘッダ	-	
宣　言	static void a_as_elcstop(uint32_t arg_num, char_t *p_arg[]);	
説　明	コンソールコマンド elcstop が入力された場合に実行される関数です。詳細は「4.11.6(5) a_as_elcstop フローチャート」と「4.11.8 コンソールコマンド」を参照してください。	
引　数	arg_num	コンソールから入力された文字列の数
	*p_arg[]	コンソールから入力された文字列の先頭アドレス。
リターン値	なし	

(8) a_as_exit

a_as_exit		
概　要	exit コンソールコマンド関数	
ヘッダ	-	
宣　言	static void a_as_exit(uint32_t arg_num, char_t *p_arg[]);	
説　明	コンソールコマンド exit が入力された場合に実行される関数です。詳細は「4.11.8 コンソールコマンド」を参照してください。	
引　数	arg_num	コンソールから入力された文字列の数
	*p_arg[]	コンソールから入力された文字列の先頭アドレス。
リターン値	なし	

(9) a_as_txerr_callback

a_as_txerr_callback		
概　要	コンソールコマンド req で送信エラー発生時のコールバック関数	
ヘッダ	-	
宣　言	static void a_as_txerr_callback(r_a_as_result_t *p_result);	
説　明	コンソールコマンド req が入力された場合に実行されるコールバック関数です。通常受信の A-format リクエストの送受信結果を a_as_result 変数に保存し、タイムアウトエラーが発生したことを通知します。詳細は「4.11.6(6) a_as_txerr_callback フローチャート」を参照してください。	
引　数	*p_result	リクエストの送受信結果が格納されている RAM の先頭アドレス
リターン値	なし	

(10) a_as_rxset_callback

a_as_rxset_callback		
概　要	コンソールコマンド req で受信データ設定完了時のコールバック関数	
ヘッダ	-	
宣　言	static void a_as_rxset_callback(r_a_as_result_t *p_result);	
説　明	コンソールコマンド req が入力された場合に実行されるコールバック関数です。通常受信の A-format リクエストの送受信結果を a_as_result 変数に保存し、受信データの設定が完了したことを通知します。詳細は「4.11.6(7) a_as_rxset_callback フローチャート」を参照してください。	
引　数	*p_result	リクエストの送受信結果が格納されている RAM の先頭アドレス
リターン値	なし	

(11) a_as_rxend_callback

a_as_rxend_callback		
概　要	コンソールコマンド req でデータ送受信完了時のコールバック関数	
ヘッダ	-	
宣　言	static void a_as_rxend_callback(r_a_as_result_t *p_result);	
説　明	コンソールコマンド req が入力された場合に実行されるコールバック関数です。通常受信の A-format リクエストに対するデータ送受信が完了したことを通知します。詳細は「4.11.6(8) a_as_rxend_callback フローチャート」を参照してください。	
引　数	*p_result	リクエストの送受信結果が格納されている RAM の先頭アドレス
リターン値	なし	

(12) a_as_elctimer_callback

a_as_elctimer_callback		
概　要	elctimer コンソールコマンドのコールバック関数	
ヘッダ	-	
宣　言	static void a_as_elctimer_callback(r_a_as_result_t * p_result);	
説　明	コンソールコマンド elctimer が入力された場合に実行されるコールバック関数です。リクエストの送受信結果を a_as_ti_result 変数と、a_as_ti_data 変数に保存します。詳細は「4.11.6(9) a_as_elctimer_callback フローチャート」を参照してください。	
引　数	*p_result	リクエストの送受信結果が格納されている RAM の先頭アドレス
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 4-16 に主要な static 型変数を示します。

表 4-16　主要な static 型変数

型	変数名	内容
bool	a_as_flg	送受信完了フラグ (true：送受信完了、false：送受信中)
r_a_as_result_t	a_as_result[A_AS_ENC_NUM]	データ取得結果を格納します。
bool	a_as_elc_flg	ELC イベント入力トリガフラグ (true：ELC イベント入力トリガ動作中、false：ELC イベント入力トリガ動作停止)
bool	elc_trans_flg	ELC イベント入力トリガ動作中のデータ送受信フラグ (true：送受信中、false：送受信完了)
r_a_as_req_t	a_as_req_elc	ELC イベント入力トリガ動作中のリクエスト情報を格納します。

4.11.5 サンプルプログラムの定数一覧

表 4-17 にサンプルプログラムで使用する主要な定数を示します。

表 4-17　主要な定数

定数名	設定値	内容
A_AS_ENC_NUM	8	エンコーダの接続数

4.11.6 メイン処理のフローチャート

以下に主要な処理を行うものについてフローチャートを記載します。

(1) enc_main フローチャート

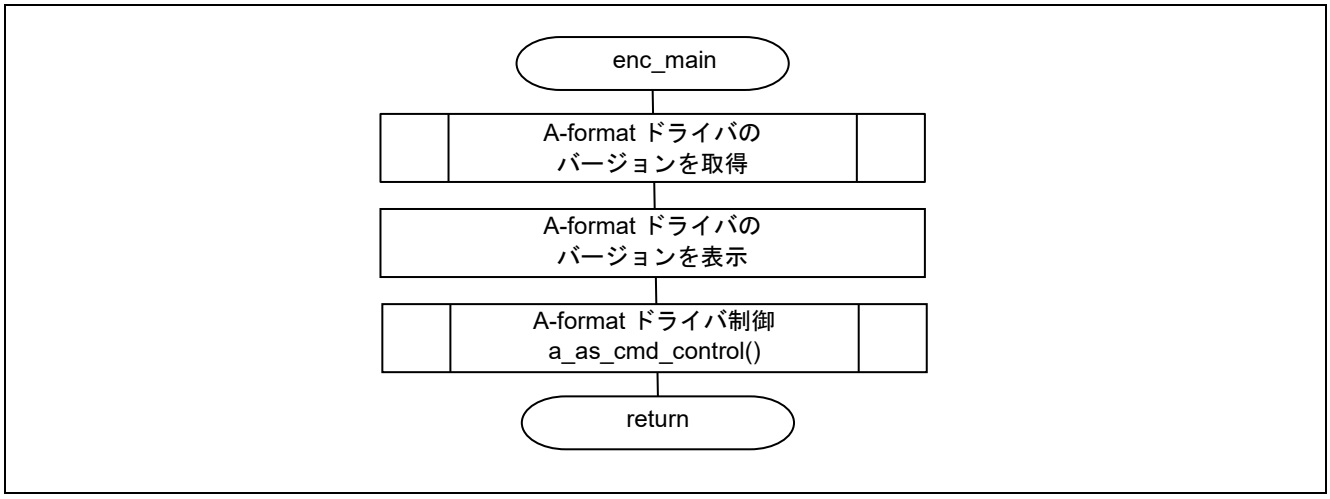

図 4-3 enc_main 関数のフローチャート

(2) a_as_cmd_control フローチャート

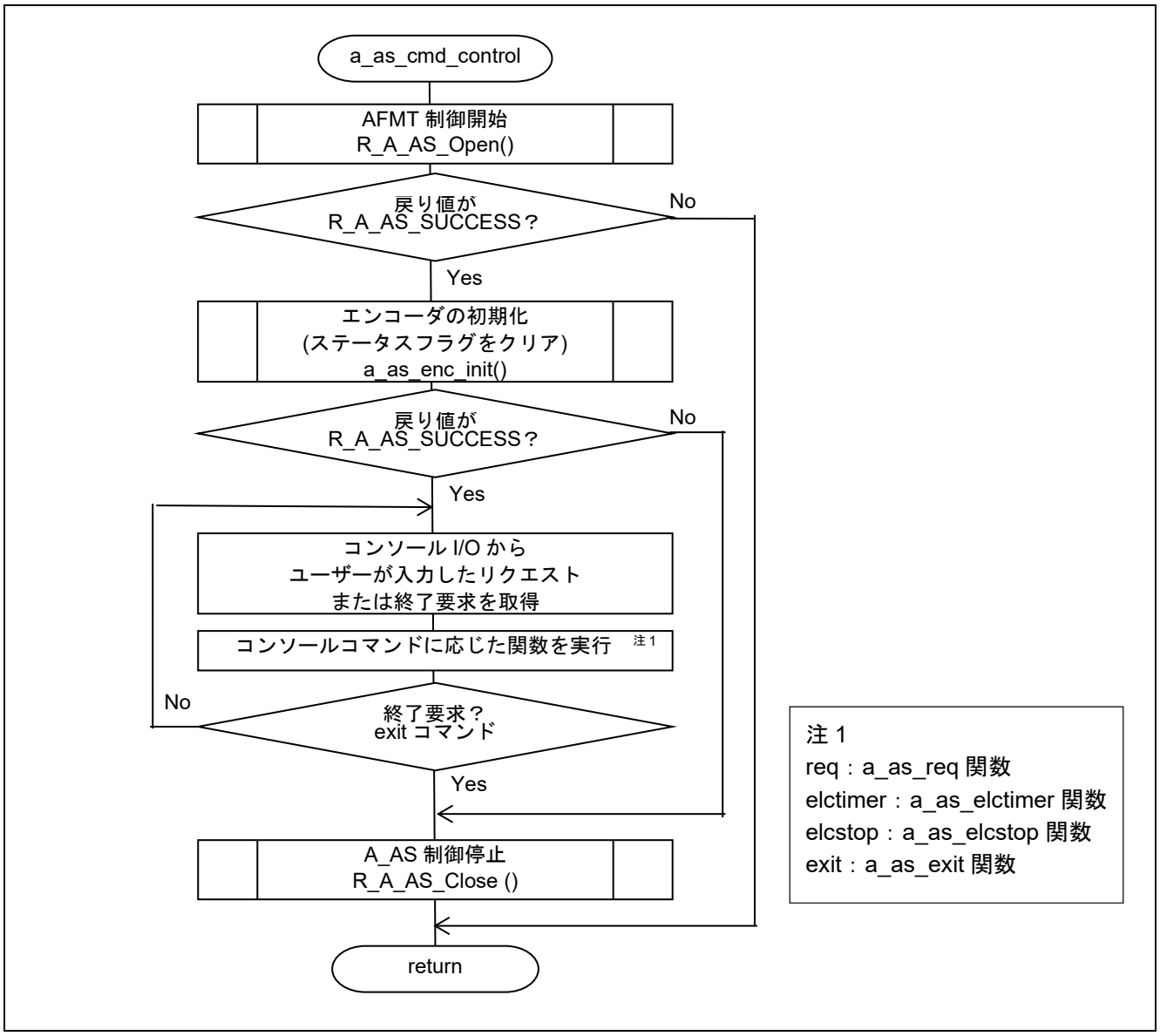

図 4-4 a_as_cmd_control 関数のフローチャート

(3) a_as_req フローチャート

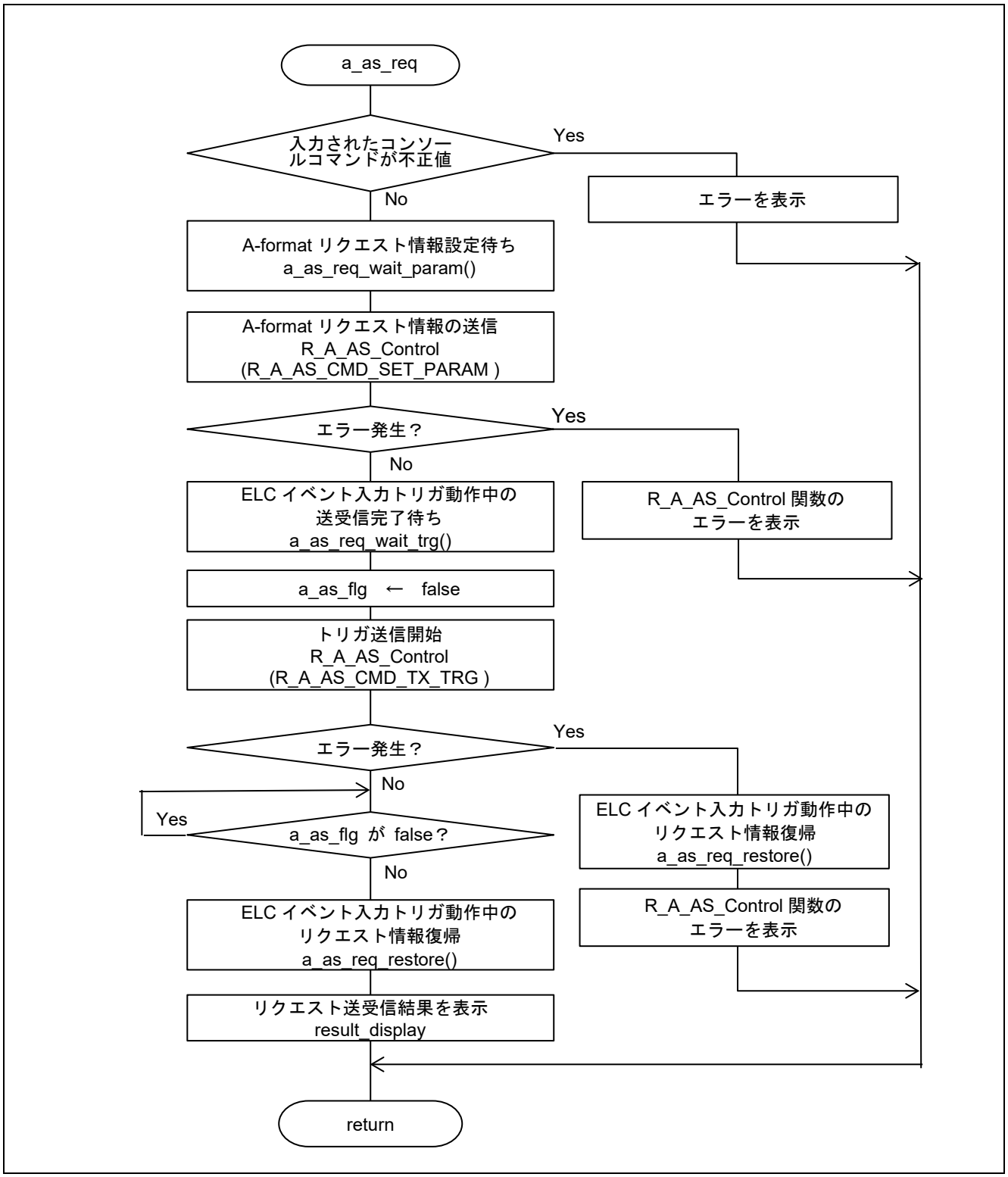

図 4-5 a_as_req 関数のフローチャート

(4) a_as_elctimer フローチャート

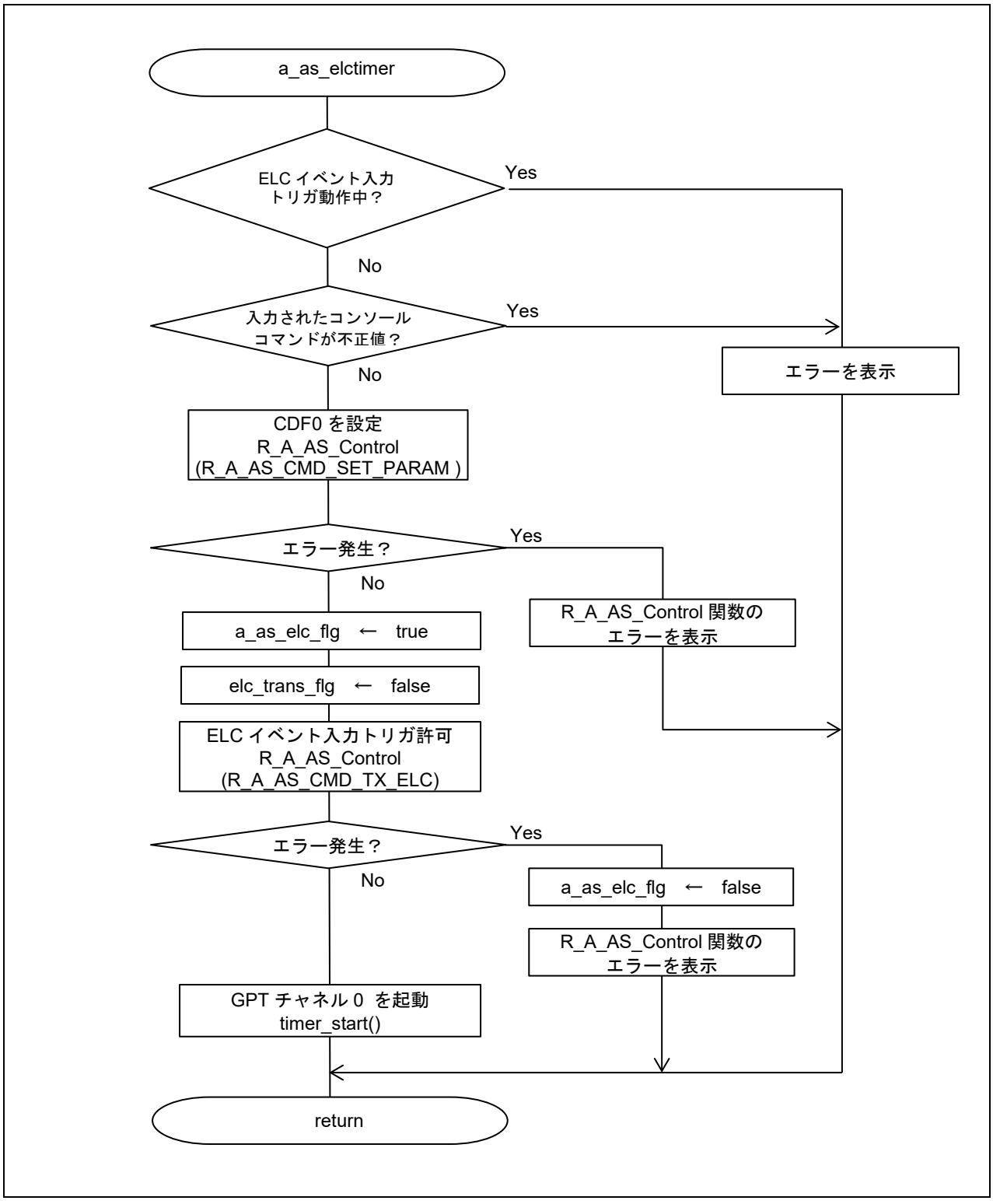

図 4-6 a_as_elctimer 関数のフローチャート

(5) a_as_elcstop フローチャート

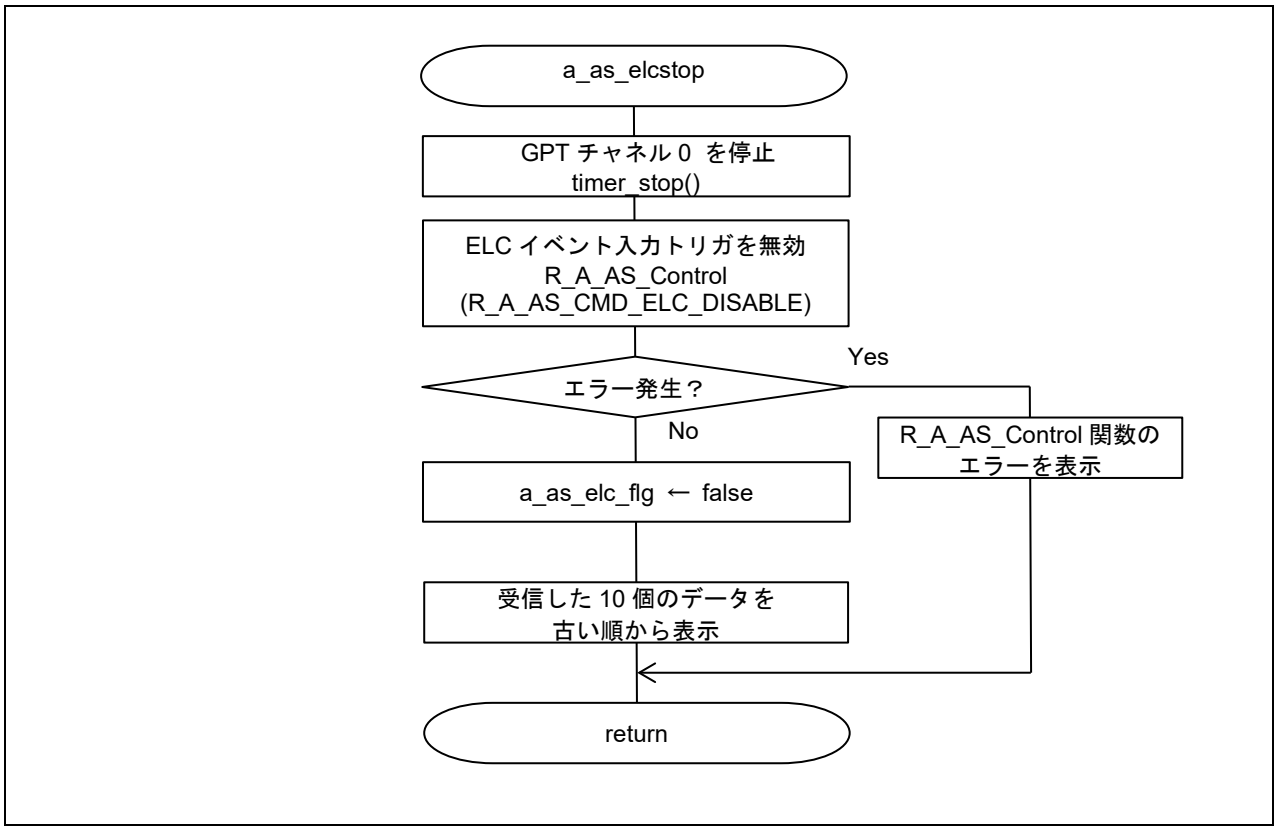

図 4-7 a_as_elcstop 関数のフローチャート

(6) a_as_txerr_callback フローチャート

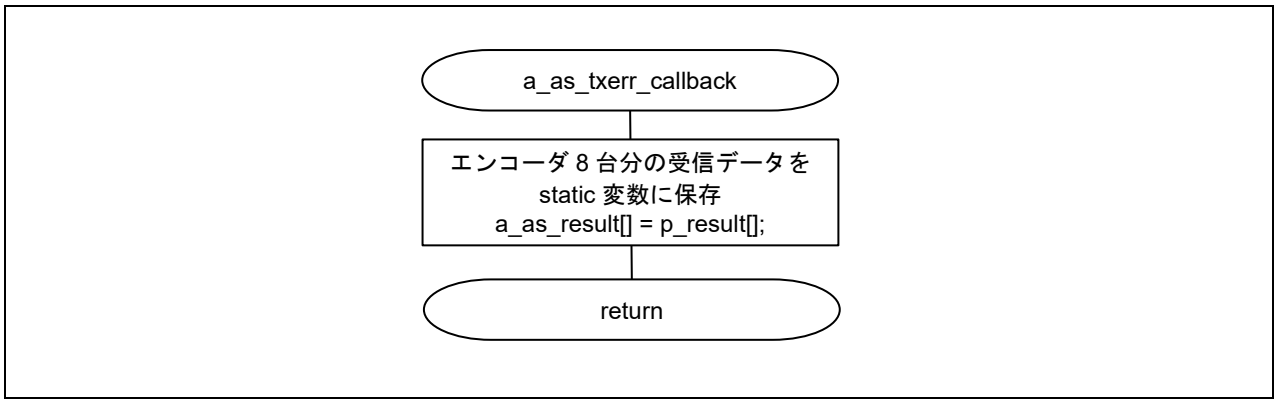

図 4-8 a_as_txerr_callback 関数のフローチャート

(7) a_as_rxset_callback フローチャート

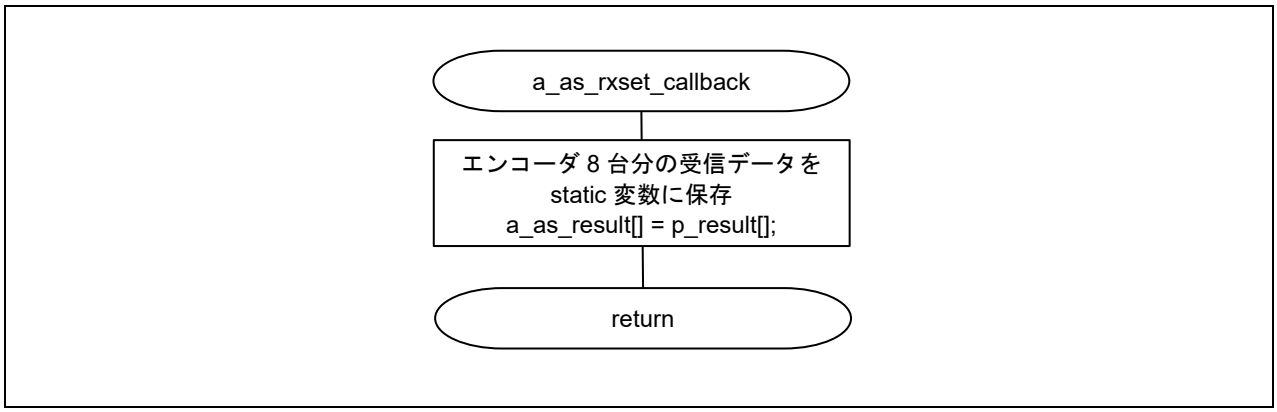

図 4-9 a_as_rxset_callback 関数のフローチャート

(8) a_as_rxend_callback フローチャート

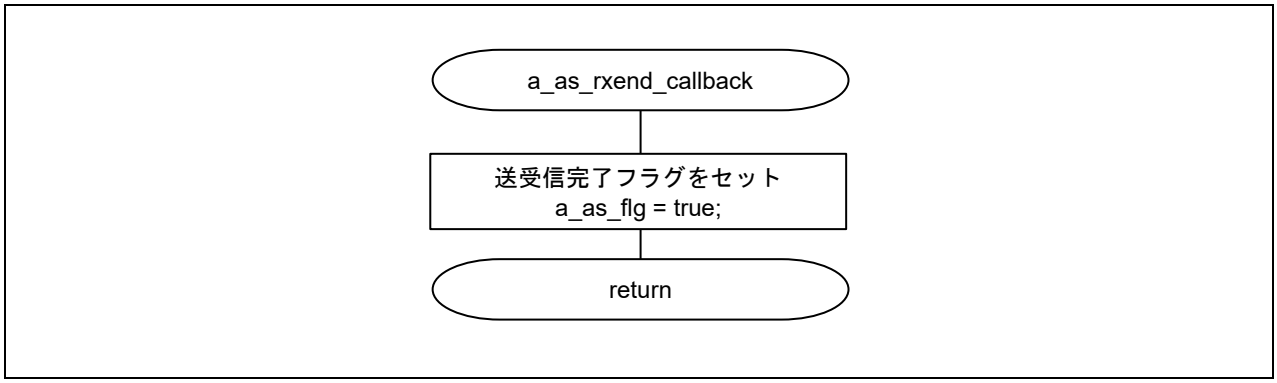

図 4-10 a_as_rxend_callback 関数のフローチャート

(9) a_as_elctimer_callback フローチャート

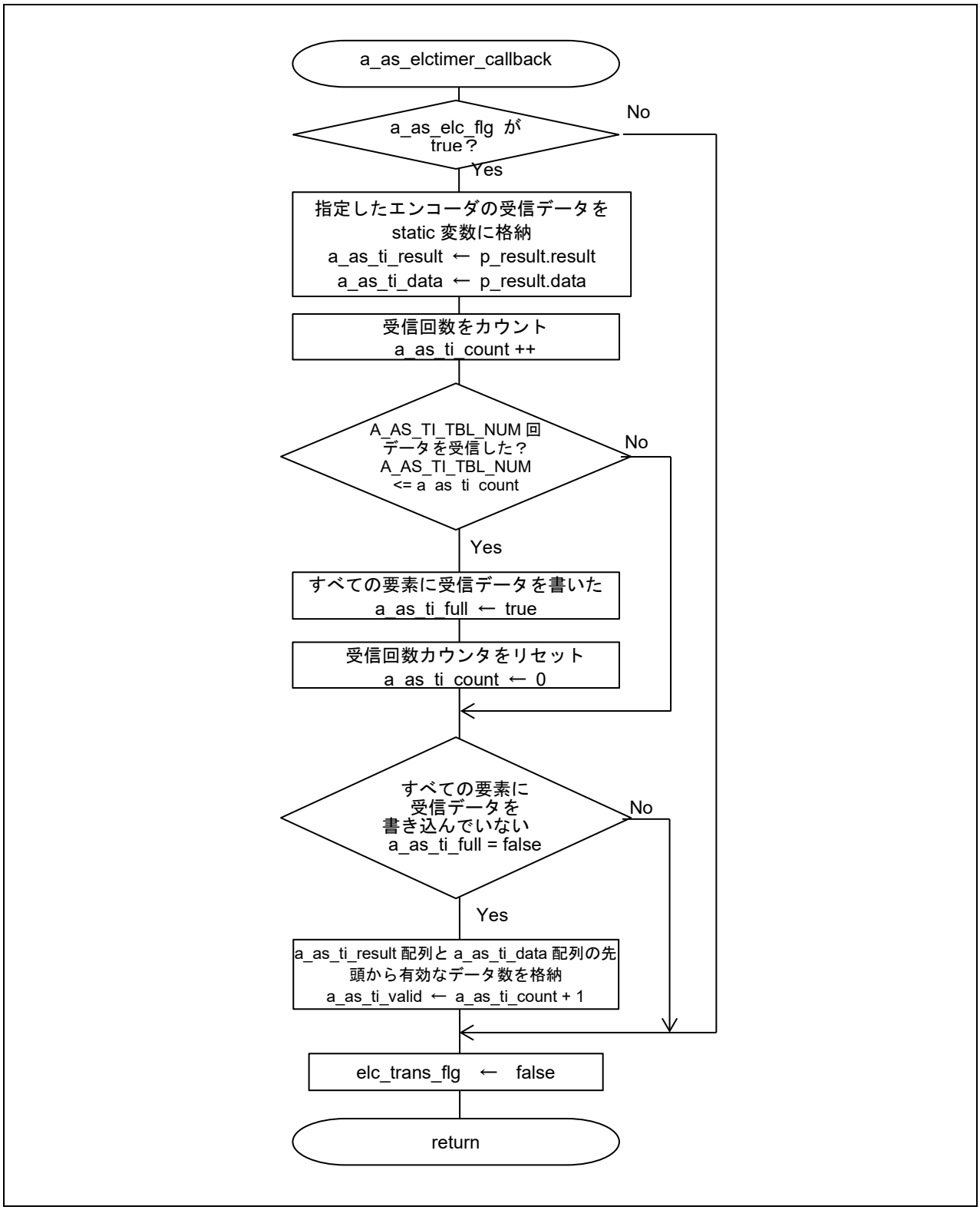

図 4-11 a_as_elctimer_callback 関数のフローチャート

4.11.7 動作シーケンス

(1) 開始シーケンス

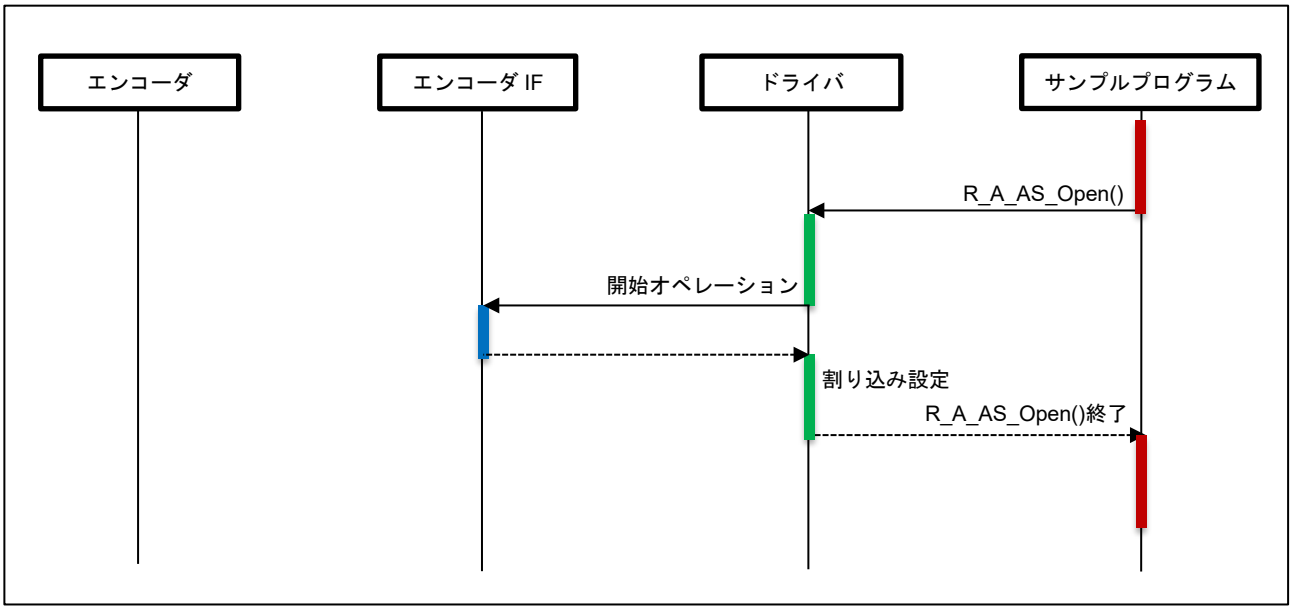

図 4-12 開始シーケンス図

(2) リクエスト送信とデータ受信のシーケンス

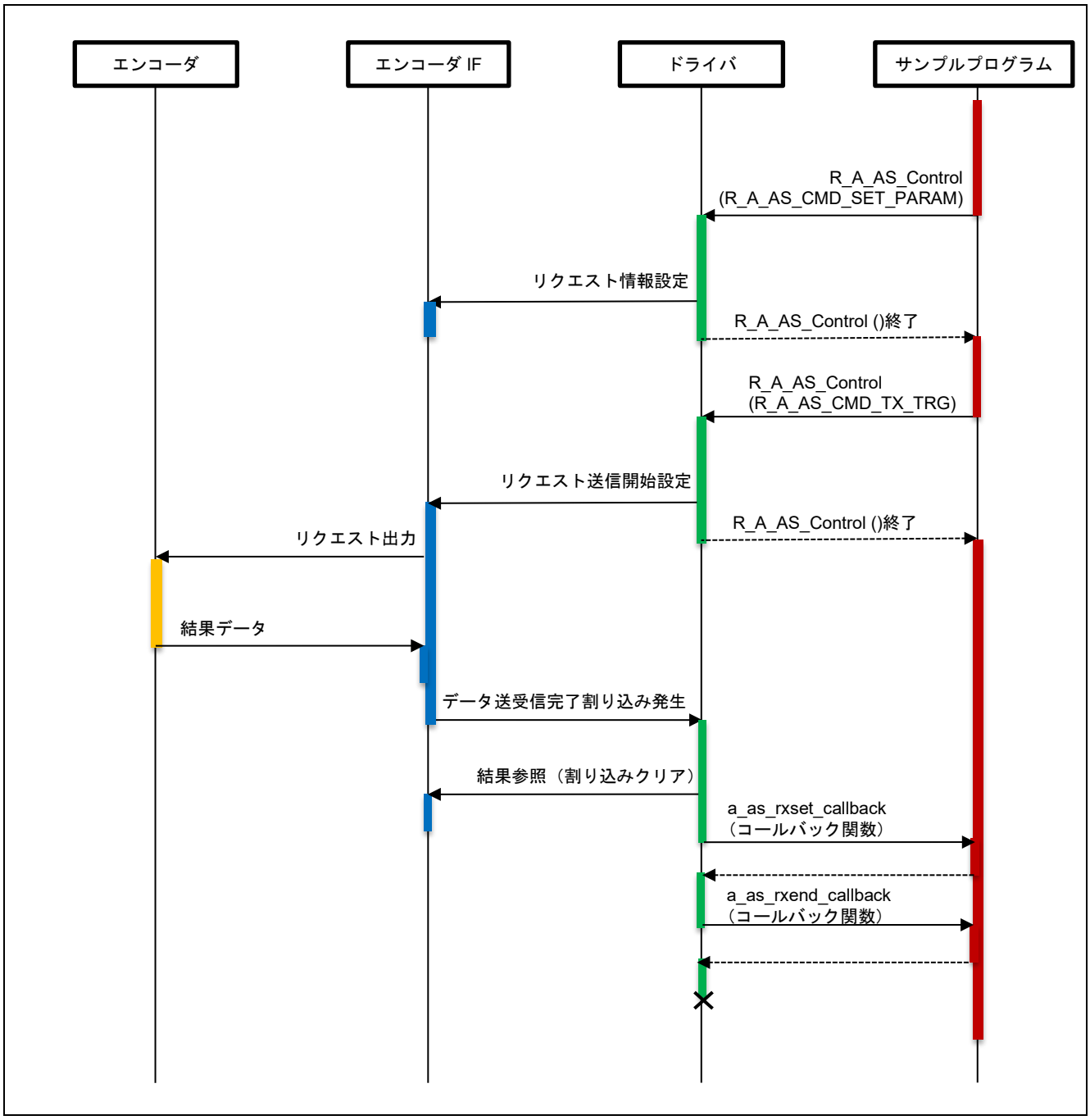

図 4-13 リクエスト送信とデータ受信のシーケンス図

(3) ELC イベント入力トリガ動作のシーケンス

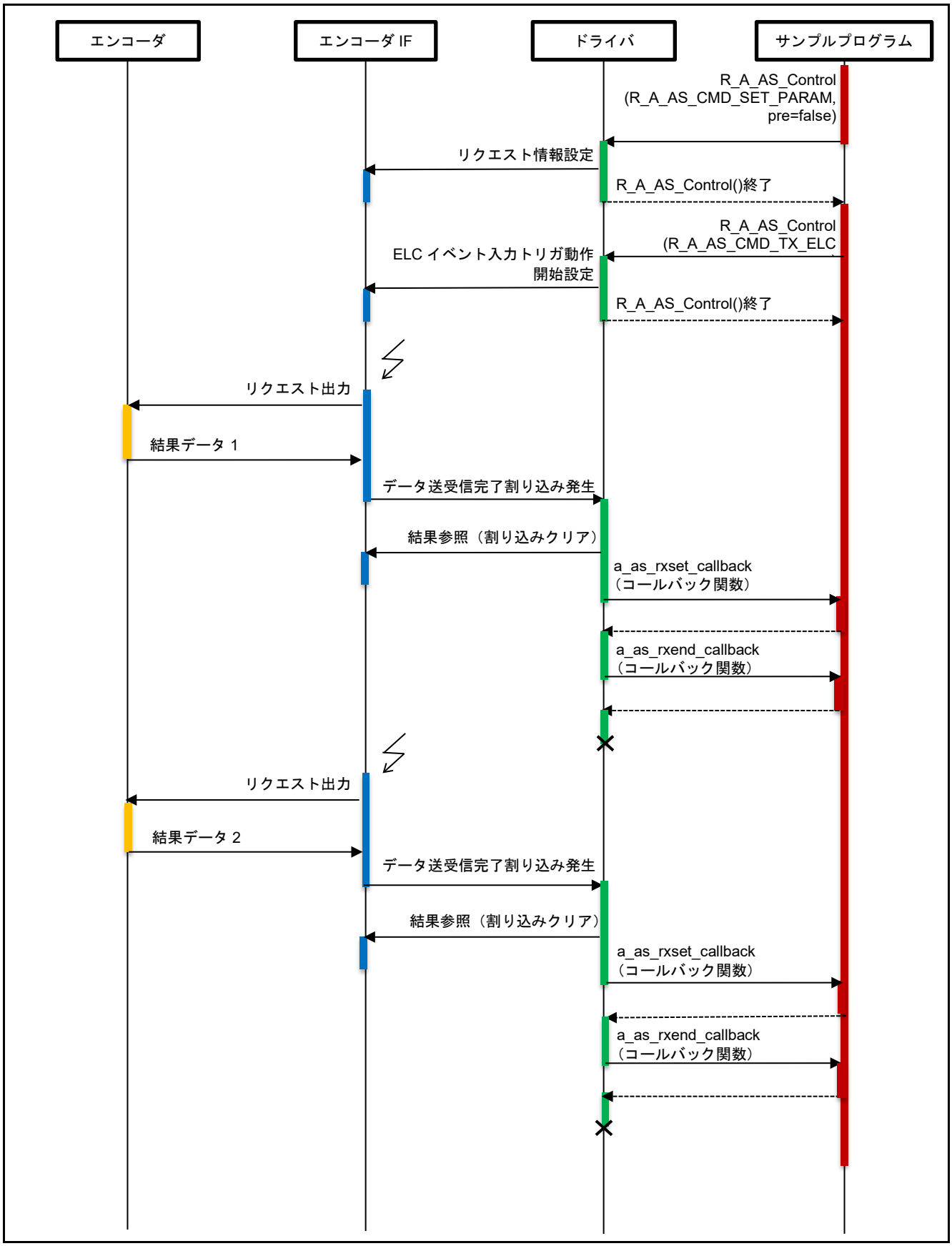

図 4-14 ELC イベントトリガ動作のシーケンス図 (1/2)

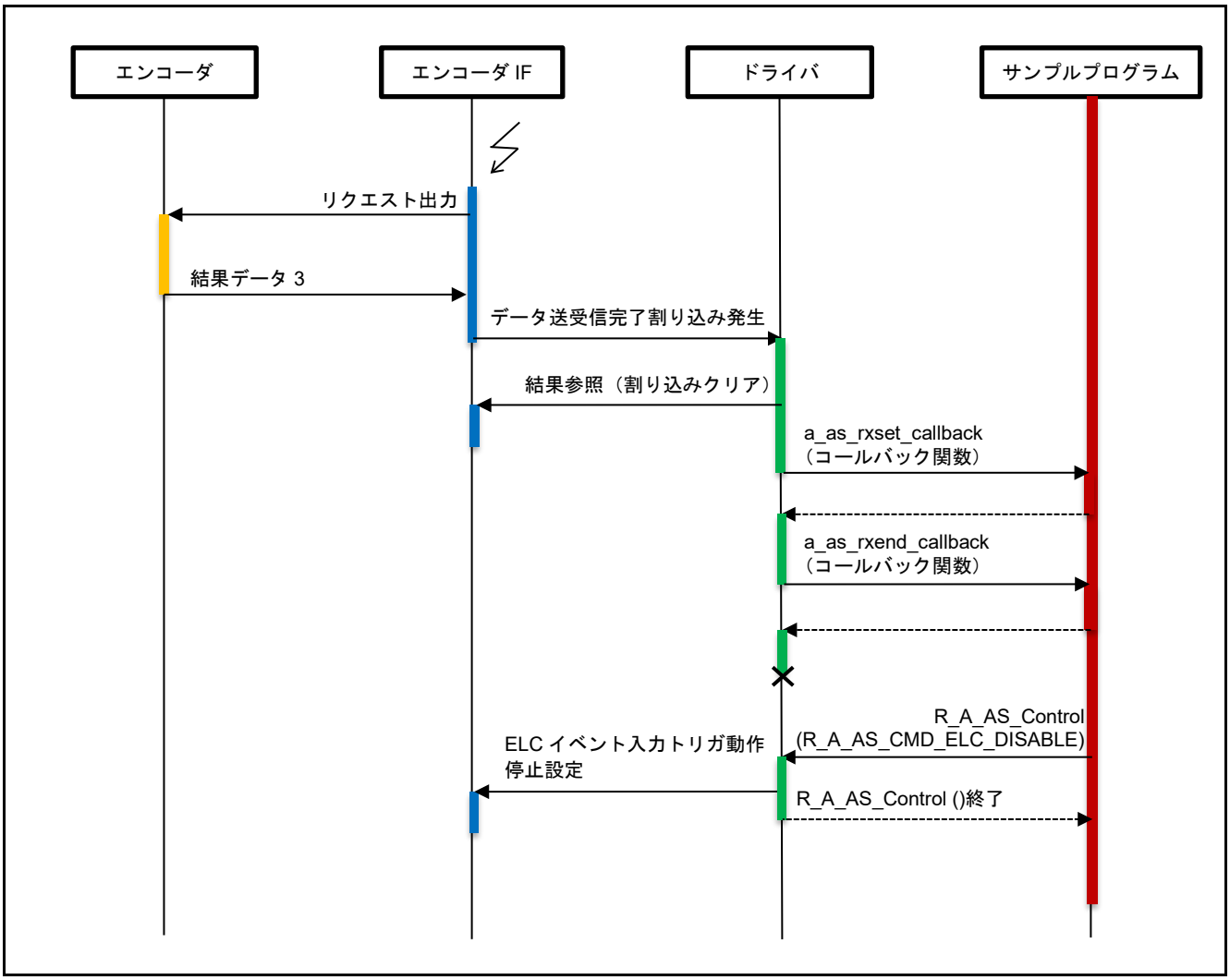

図 4-15 ELC イベントトリガ動作のシーケンス図 (2/2)

(4) 停止シーケンス

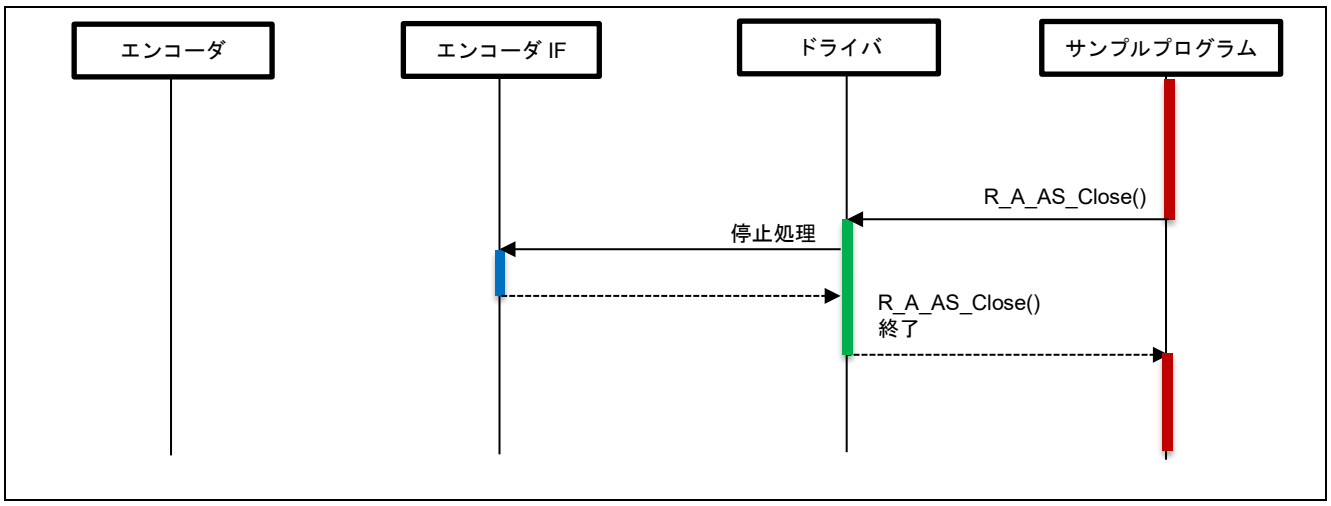

図 4-16 停止シーケンス図

4.11.8 コンソールコマンド

本サンプルプログラムは A-format 仕様に準拠したエンコーダ(MAR-M50A)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 4-18 コンソールコマンド一覧

コマンド	内容
req *ea cmd param1 param2*	エンコーダに対し *cmd* に入力したコマンドフレームを送信します。 *ea*：エンコーダ区分番号(10 進数) *cmd*：コマンドデータフレーム *param1*、*param2*：*cmd* によって入力値が異なります。詳細は「表 4-19 req コマンドのパラメータ対応表」を参照してください。
elctimer *ea val*	ELC イベント入力トリガ動作で、指定したエンコーダに対しコマンドデータフレーム CDF0 をビットレート 2.5 Mbps で送信します。 *ea*：エンコーダ区分番号(10 進数) *val*：トリガ動作のタイミング （単位は us です。最大 6990us まで設定できます。） ELC イベント入力トリガ動作を停止させる場合はコンソールコマンド「elcstop」を実行してください。
elcstop	ELC イベント入力トリガ動作を停止させます。
exit	プログラム終了

表 4-19 req コマンドのパラメータ対応表

cmd	param1	param2
CDF0~CDF12	なし	なし
CDF13	EEPROM のアドレス(16 進数) 0x00~0xFF の範囲で指定してください。	なし
	使用例（エンコーダ区分番号 2 番の EEPROM の 0 番地のデータを読み込む） req 2 CDF13 00	
CDF14	EEPROM のアドレス(16 進数) 0x00~0xEF の範囲の値を入力してください。	EEPROM へ書き込むデータ(16 進数) 0x0000~0xFFFF の範囲の値を入力してください。
	使用例（エンコーダ区分番号 2 番の EEPROM の 0 番地に 0x1234 を書き込む） req 2 CDF14 00 1234	
CDF15~17	なし	なし
CDF18, CDF19	識別コード(16 進数) 0x00 0000~0xFF FFFF の範囲の値を入力してください。	なし
	使用例（エンコーダ区分番号 2 番の識別コードに 0x12 3456 を書き込む） req 2 CDF18 123456	
CDF21, CDF27, CDF29	なし	なし

【注】 CDF11、CDF17、CDF19 は入力できますが、本サンプルプログラムの接続方式がバス接続のため使用できません。
複数伝送コマンド(CDF4～7, CDF22, CDF28, CDF30)および、CDF20 コマンドには対応していません。

(1) サンプルプログラム実行

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

```
A-format sample program start
R_A_AS_GetVersion = 4.0


a_as >
```

(2) コマンド実行例

接続しているエンコーダ ENC1 に対して、コマンドフレーム CDF0 を送信するためのコンソール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、エンコーダアドレスやステータス情報とともに、角度データが表示されます。

```
a_as >req 1 CDF0
req command
------------------------------
ENC1
R_A_AS_REQ_SUCCESS
EA : 0
ES : 0
CC : 0
ABS 40bit [39:32] : 0x00000005
ABS 40bit  [31:0] : 0x00560E11
```

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
0.80	Oct.04.22	-	初版発行
1.00	Mar.31.23	4	使用ボードの型番を更新
2.00	May 17.24	4 20 38 41 46	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 elctimer コマンドで R_A_AS_Control に設定するコマンドを、CDF4 から CDF0 に変更 フローチャート内の、格納するデータの説明を変更 表 4-18 の、elctimer コマンドの仕様を変更 表 4-19 に、複数伝送コマンドに対応していない旨の注記を追加
3.00	Sep 26.25	1, 3, 4 7 - 13 23 27, 31 45	商標の説明の記載方法を更新 関数仕様の説明のヘッダ欄を更新 構造体 r_a_as_status_t の説明を簡潔に変更 a_as_elctimer_callback をユーザ定義関数の説明に追加 コマンド実行例を追加
4.00	Apr 10.26	7 - 14, 29 - 31 12, 13	ポインタ変数のプレフィクスを”p_”に変更 「4.5 ユーザー定義関数仕様」のヘッダ部分を修正

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。