

RL78/G23

外部フラッシュ・メモリを利用した簡易 SPI (CSI) 通信ファームウェア・アップデート

要旨

本アプリケーションノートでは、外部フラッシュ・メモリを利用したファームウェア・アップデートの概要を説明します。

RL78/G23 は外部フラッシュ・メモリと簡易 SPI (CSI) 通信を行い、ファームウェア更新用のデータを取得します。Renesas Flash Driver (RFD) を使用してコード・フラッシュ・メモリに取得したデータを書き込み、ファームウェアを更新します。

動作確認デバイス

RL78/G23

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

目次

1. 概要	4
1.1 仕様概要	4
1.1.1 Renesas Flash Driver RL78 Type01 概要	4
1.1.2 コード・フラッシュ・メモリについて	5
1.1.3 セルフ・プログラミング	6
1.1.4 フラッシュ書き換え	7
1.1.5 フラッシュ・シールド・ウインドウ	8
1.1.6 通信仕様	8
1.1.7 Renesas Flash Driver RL78 Type01 の取得方法	8
1.2 動作概要	9
2. 動作確認条件	11
3. ハードウェア説明	12
3.1 ハードウェア構成例	12
3.2 使用端子一覧	13
4. ソフトウェア設定	14
4.1 オプション・バイトの設定一覧	14
4.2 スタートアップ・ルーチンの設定	15
4.2.1 スタック領域用セクション (.stack_bss) の定義	15
4.2.2 書き換え用プログラムの RAM 領域への配置	16
4.3 オンチップ・デバッグ・セキュリティ ID	17
4.4 サンプル・プログラム使用リソース	17
4.4.1 ROM 領域セクション一覧	17
4.4.2 RAM 領域セクション一覧	17
4.5 定数一覧	18
4.6 データ型定義	19
4.7 変数一覧	19
4.8 関数一覧	20
4.9 関数仕様	21
4.10 フローチャート	26
4.10.1 メイン処理	26
4.10.2 RFD RL78 Type01 初期化処理	28
4.10.3 コード・フラッシュ・メモリ ブロック制御処理	29
4.10.4 コード・フラッシュ・メモリ 書き込み・ベリファイ処理	30
4.10.5 コード・フラッシュ・メモリ ブロック消去処理	31
4.10.6 コード・フラッシュ・メモリ 書き込み処理	32
4.10.7 コード・フラッシュ・メモリ ベリファイ処理	33
4.10.8 コード・フラッシュ・メモリ シーケンサ制御終了処理	34
4.10.9 CSI11 送信完了割り込み時の処理	36
4.10.10 CSI11 受信完了割り込み時の処理	37
4.10.11 外部フラッシュ・メモリ データ取得処理	38
4.10.12 IICA0 送信完了割り込み時の処理	39
4.10.13 IICA0 送信エラー発生時の処理	40

4.10.14LCD モジュール 初期化.....	41
4.10.15LCD モジュール 表示消去処理.....	42
4.10.16LCD モジュール 文字列送信処理	43
4.10.17LCD モジュール コマンド送信処理.....	44
4.10.18LCD モジュール データ送信処理	45
4.10.19LCD モジュール 通信終了フラグ設定	46
4.10.20LCD モジュール 通信終了待ち処理.....	47
4.10.21外部割り込み (INTP0) 処理	48
 5. サンプルコード	 49
 6. 参考ドキュメント	 49
 改訂記録.....	 50

1. 概要

1.1 仕様概要

本アプリケーションノートのサンプル・プログラムは、始めに、LCD モジュールに現在のバージョン情報を表示します。その後、スイッチが押される (INTP0 割り込みが発生する) と、「フラッシュ・アクセス中」を示す LED を点灯し、セルフ・プログラミング・モードに移行します。続けて外部フラッシュ・メモリからファームウェア更新用データを取得し、コード・フラッシュ・メモリを書き換えます。書き換えが完了すると LED を消灯し、リセットを発生させます。再起動すると、LCD モジュールに書き換え後のバージョン情報を表示します。

表 1-1 使用する周辺機能と用途

周辺機能	用途
シリアル・アレイ・ユニット CSI11	簡易 SPI (CSI) でデータの通信を行う
シリアル・インタフェース IICA0	LCDモジュールとのI2C通信
外部割り込み	スイッチ入力

1.1.1 Renesas Flash Driver RL78 Type01 概要

Renesas Flash Driver (RFD) RL78 Type01 は、RL78/G23 のフラッシュ・メモリ内のデータを書き換えるためのソフトウェアです。

RFD RL78 Type01 の API 関数をユーザ・プログラムから呼び出すことにより、コード・フラッシュ・メモリ、またはデータ・フラッシュ・メモリの内容を書き換えることができます。

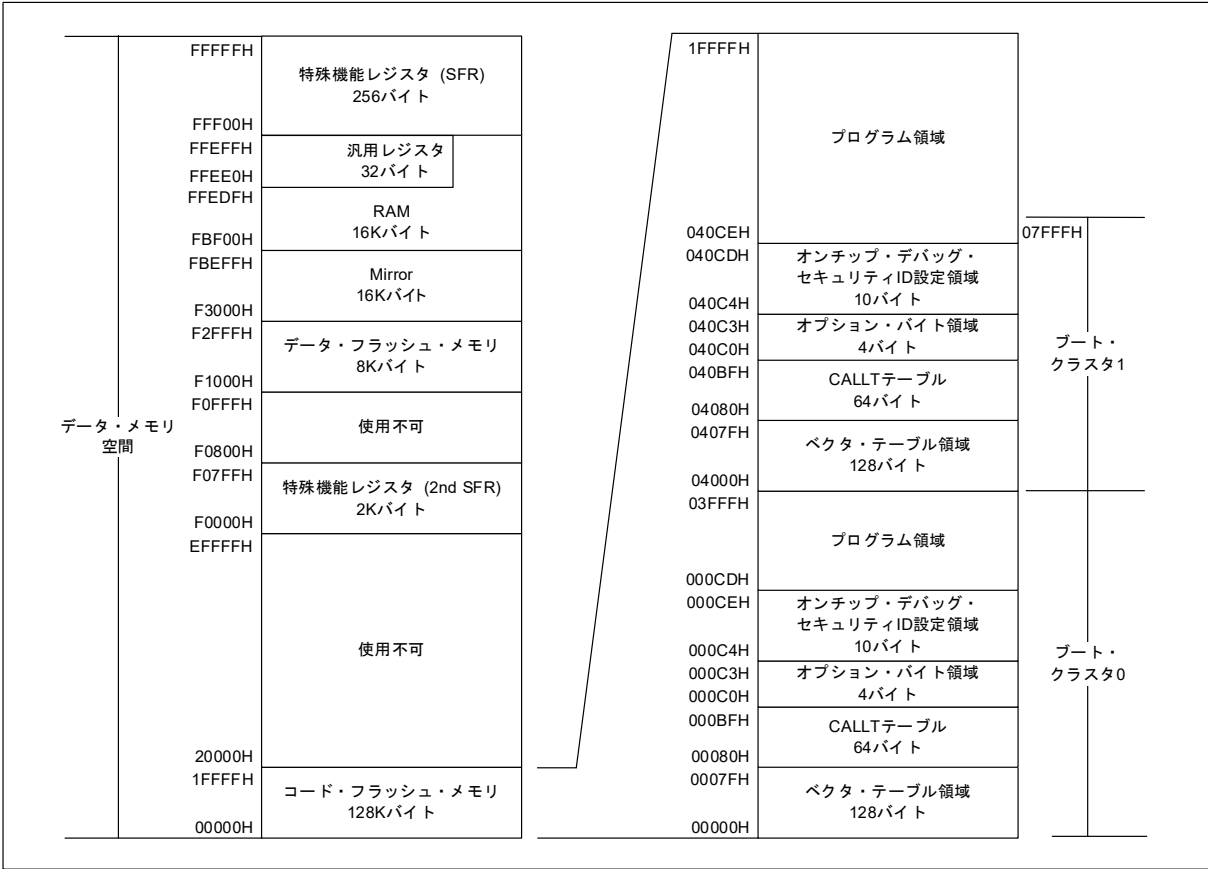
セルフ・プログラミングを行うためにはセルフ・プログラミング環境を設定する必要があります。セルフ・プログラミング中は、高速オンチップ・オシレータを動作させておく必要があります。高速オンチップ・オシレータを停止させている場合は、高速オンチップ・オシレータ・クロック動作 (HIOSTOP = 0) させ、5 μ s 経過後にセルフ・プログラミングを実行してください。また、中速オンチップ・オシレータは停止 (MIOEN = 0) させ、メイン・オンチップ・オシレータ・クロック (fOCO) は高速オンチップ・オシレータを選択 (MCM1 = 0) してください。フラッシュ・メモリの書き換え中は、フラッシュ動作モード選択レジスタ (FLMODE レジスタ) は変更しないでください。

また、コード・フラッシュ・プログラミング・モード中に実行する ROM (コード・フラッシュ・メモリ) 上のユーザ・プログラム、および参照データは、事前に RAM へコピーして、RAM 上で実行、参照してください。

1.1.2 コード・フラッシュ・メモリについて

RL78/G23 (R7F100GLG) のコード・フラッシュ・メモリの構成を以下に記載します。

図 1-1 メモリ構成



注意 ブート・スワップ機能を使用する際には、ブート・クラスタ 0 のオプション・バイト領域 (000C0H – 000C3H) は、ブート・クラスタ 1 のオプション・バイト領域 (040C0H – 040C3H) と切り替わります。そのため、ブート・スワップ機能を使用する際には、040C0H – 040C3H に、000C0H – 000C3H と同じ値を設定してください。

RL78/G23 のコード・フラッシュ・メモリの特徴を以下に記載します。

表 1-2 コード・フラッシュ・メモリの特徴

項目	内容
消去の最小単位	1 ブロック (2048byte)
書き込みの最小単位	1 ワード (4byte)
ベリファイの最小単位	1 byte
セキュリティ機能	ブロック消去、書き込み、ブート・クラスタ0の書き換え禁止設定が可能 (出荷時は全て許可)
	フラッシュ・シールド・ウインドウにより、指定したウインドウ範囲内 または範囲外の書き込みおよび消去をセルフ・プログラミング時のみ禁 止にすることが可能
	フラッシュ・セルフ・プログラミング・コード (Renesas Flash Driver RL78 Type01) によりセキュリティ設定変更可能

注意 セルフ・プログラミング時はブロック消去禁止および書き込み禁止のセキュリティ設定は無効となり
ます。セルフ・プログラミング時のブロック消去禁止および書き込み禁止に設定したい場合は、フラッ
シュ・シールド・ウインドウ機能を使用してください。

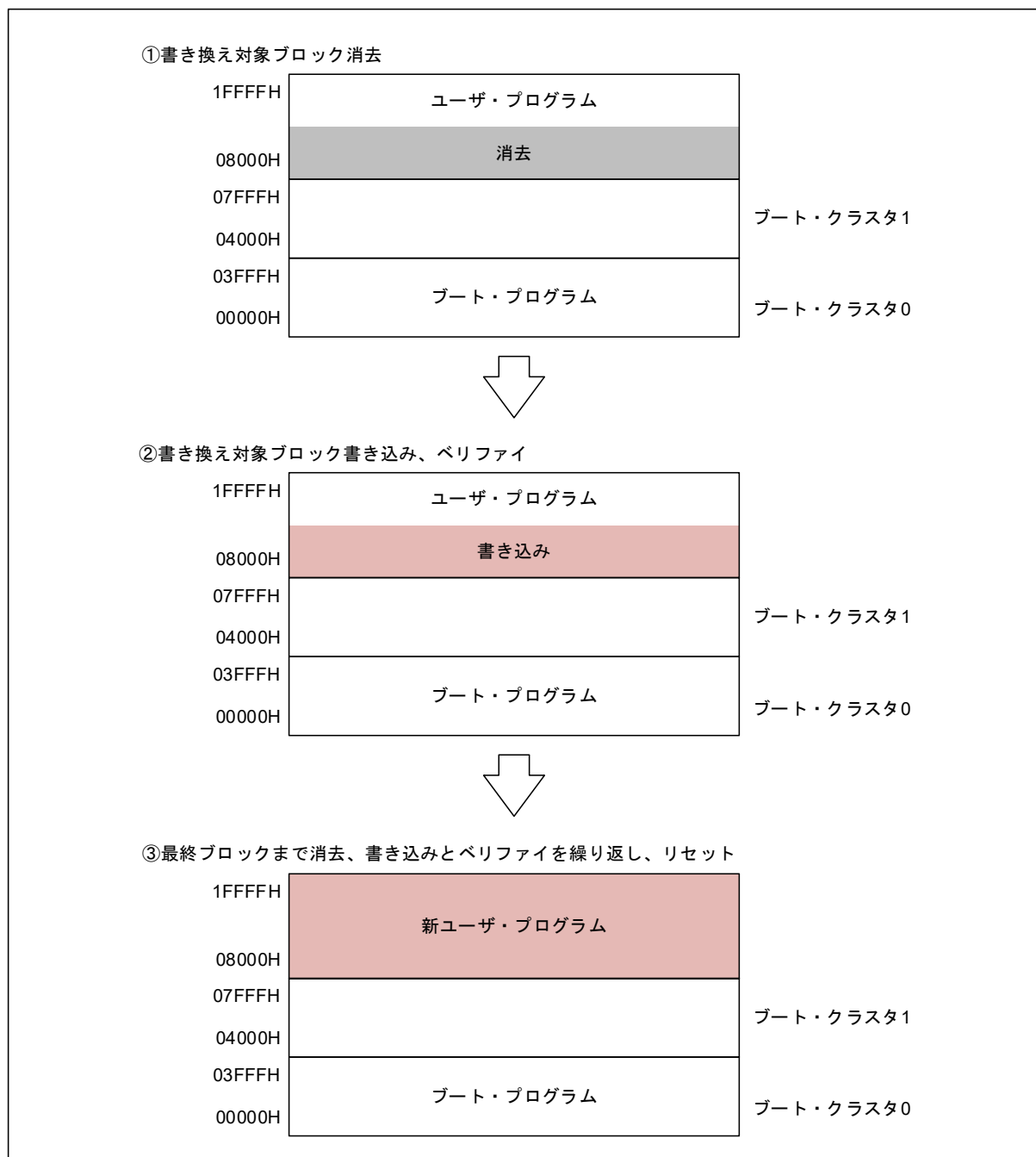
1.1.3 セルフ・プログラミング

RL78/G23 には、セルフ・プログラミングを行うための Renesas Flash Driver (RFD) が用意されています。
書き換えプログラムから RFD の API 関数を呼び出すことでセルフ・プログラミングを行います。

RL78/G23 のセルフ・プログラミングは、フラッシュ・メモリ・シーケンサを使用してフラッシュの書き
換え制御を行います。シーケンサの制御中はコード・フラッシュ・メモリを参照できません。そのため、
シーケンサ制御中にユーザ・プログラムを動作させる必要がある場合、コード・フラッシュ・メモリの消去
や書き込み、セキュリティ・フラグの設定等を行う時に、RFD の一部のセグメントや書き換えプログラムを
RAM に配置して制御を行う必要があります。

1.1.4 フラッシュ書き換え

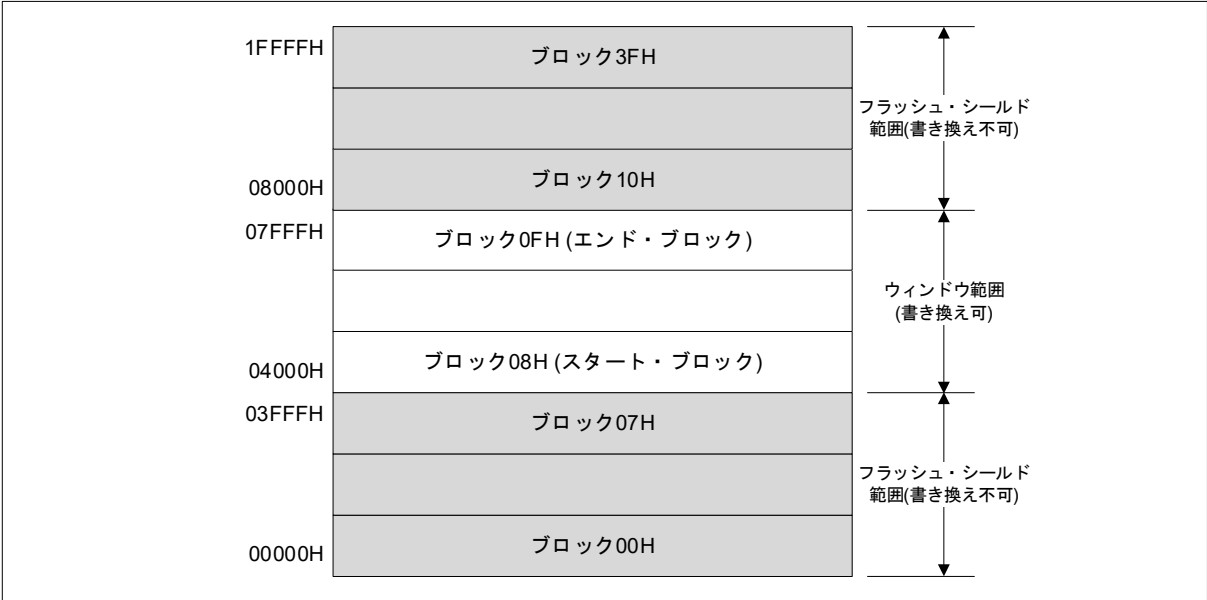
セルフ・プログラミングでのプログラムの書き換え動作イメージを以下に記載します。セルフ・プログラミングを行うプログラムは、ブート・クラスタ 0 に配置しています。



1.1.5 フラッシュ・シールド・ウインドウ

フラッシュ・シールド・ウインドウはセルフ・プログラミング時のセキュリティ機能の一つで、指定したウインドウ範囲内または範囲外の書き込み、及び消去をセルフ・プログラミング時のみ禁止に設定する機能です。

以下に、スタート・ブロックが 08H、エンド・ブロックが 0FH の場合のイメージ図を記載します。



1.1.6 通信仕様

本アプリケーションノートのサンプル・プログラムは、外部フラッシュ・メモリからファームウェア更新用データを取得し、セルフ・プログラミングを行います。異常終了の場合には応答を返さず、LCD モジュールに"ERROR!"と表示して、以降の処理は行いません。以下に 簡易 SPI (CSI) 通信設定を記載します。

表 1-3 簡易 SPI (CSI) 通信設定

転送モード	シングル転送モード
データ・ビット長[bit]	8
データ転送方向	MSB ファースト
データ送受信タイミング	タイプ1
転送レート	153600 bps

1.1.7 Renesas Flash Driver RL78 Type01 の取得方法

コンパイルを実行する前に、最新版の Renesas Flash Driver RL78 Type01 をダウンロードし、RFD_RL78_TYPE1 フォルダ内にファイルをコピーしてください。

Renesas Flash Driver RL78 Type01 は、下記 URL から取得することができます。

<https://www.renesas.com/jp/ja/document/scd/renesas-flash-driver-rl78-type-01-rl78g23>

1.2 動作概要

本アプリケーションノートでは、外部フラッシュ・メモリを利用したファームウェア・アップデートの方法を説明します。

LCD モジュールに現在のバージョン情報を表示します。その後、スイッチが押されると、「フラッシュ・アクセス中」を示す LED を点灯し、セルフ・プログラミング・モードに移行します。続けて外部フラッシュ・メモリからファームウェア更新用データを取得し、コード・フラッシュ・メモリを書き換えます。書き換えが完了すると、LED を消灯して LCD モジュールにバージョン情報を表示します。

- (1) ポートの初期設定を行います。
 - ・ P53 を出力ポートに設定 (初期値ハイ・レベルで LED1 消灯状態)
- (2) シリアル・アレイ・ユニット CSI11 の初期設定を行います。
 - ・ CSI11 を使用 (P30 : SCK11、P50 : SI11、P51 : SO11 に設定)
 - ・ CSI11 動作クロックを fCLK に設定
 - ・ 転送モードをシングル転送モードに設定
 - ・ データ・ビット長を 8 ビットに設定
 - ・ データ転送方向を MSB に設定
 - ・ データ送受信タイミングをタイプ 1 に設定
 - ・ ボーレートを 153600 bps に設定
 - ・ INTCSI11 割り込みを許可
- (3) シリアル・インタフェース IICA の初期設定を行います。
 - ・ IICA0 を使用 (P60 : SCLA0、P61 : SDAA0 に設定)
 - ・ IICA0 動作クロックを fCLK/2 に設定
 - ・ 自局アドレスを 10H に設定
 - ・ 動作モードを標準に設定
 - ・ 転送クロックを 80000 bps に設定
 - ・ INTIICA0 割り込みを許可
- (4) 外部割り込みの初期設定を行います。
 - ・ INTP0 の有効エッジを立ち下がリエッジに設定
- (5) 外部割り込みを許可します。
- (6) CSI11 の動作を開始します。
- (7) LCD モジュールの初期設定を行い、LCD モジュールに定数 LCD_STRING で設定された文字列を表示します。
- (8) スイッチの入力待ちを行います。
- (9) スイッチの入力を確認した場合、P53 をロウ・レベル出力にし、「フラッシュ・アクセス中」を示す LED1 を点灯し、セルフ・プログラミングの初期設定を行います。

- (10) 書き込み先アドレスを 08000H (コード・フラッシュ・メモリのプログラム領域)に設定します。
- (11) 書き込み先アドレスから、書き換え対象ブロックを算出します。(ブロック番号 : 010H)
- (12) r_CF_EraseBlock 関数を呼び出し、書き換え対象ブロックを消去します。
- (13) 外部フラッシュ・メモリより書き込みデータ (256 バイト) を取得します。
- (14) r_CF_WriteData 関数を呼び出し、書き込み先アドレスに取得したデータを書き込みます。
- (15) r_CF_VerifyData 関数を呼び出し、書き込まれたデータと受信データの対象ブロックのベリファイを行います。
- (16) 書き込み先アドレスを書き込みサイズ分 (256 バイト) 加算します。
- (17) 1 ブロック (2048 バイト) 分の書き込みが完了するまで (10) ~ (13) を繰り返します。
- (18) P53 をハイ・レベル出力にし、「フラッシュ・アクセス中」を示す LED1 を消灯後、R_RFD_ForceReset 関数を呼び出して内部リセットを発生させます。
- (19) 再起動後、デバイスの初期設定を行ったのち、書き換え後の定数 LCD_STRING に設定された文字列が LCD モジュールに表示されます。

注意 フラッシュ・セルフ・プログラミングを正常終了することができなかった場合 (処理中にエラーが発生した場合) は LCD モジュールに「ERROR!」と表示し、以降の処理は行いません。

2. 動作確認条件

本アプリケーションノートのサンプルコードは下記の条件で動作を確認しています。

表 2-1 動作確認条件

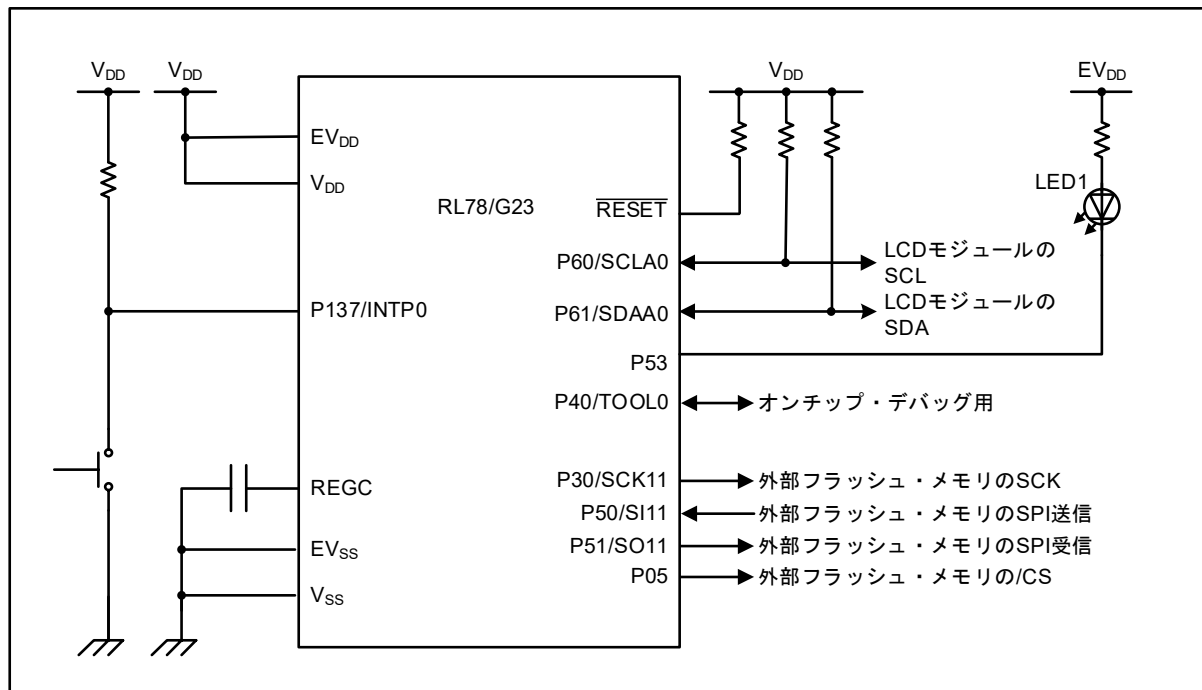
周辺機能	用途
使用マイコン	RL78/G23 (R7F100GLG)
動作周波数	高速オンチップ・オシレータ・クロック (fIH) : 32MHz
動作電圧	3.3V (3.1V~3.5V で動作可能) LVD 検出電圧: リセット・モード 立ち上がり時 TYP. 1.90V (1.84 V ~ 1.95 V) 立ち下がり時 TYP. 1.86V (1.80 V ~ 1.91 V)
統合開発環境 (CS+)	ルネサス エレクトロニクス製 CS+ for CC V8.06.00
C コンパイラ (CS+)	ルネサス エレクトロニクス製 CC-RL V1.10.00
統合開発環境 (e2studio)	ルネサス エレクトロニクス製 e2studio V2021-07
C コンパイラ (e2studio)	ルネサス エレクトロニクス製 CC-RL V1.10.00
統合開発環境 (IAR)	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 V4.21.2
C コンパイラ (IAR)	IAR Systems 製 IAR C/C++ Compiler for Renesas RL78 V4.21.2.2420
使用ボード	RL78/G23-64p Fast Prototyping Board, RTK7RLG230CLG000BJ

3. ハードウェア説明

3.1 ハードウェア構成例

図 3-1 に本アプリケーションノートで使用するハードウェア構成例を示します。

図 3-1 ハードウェア構成



- 注意 1. この回路イメージは接続の概要を示す為に簡略化しています。実際に回路を作成される場合は、端子処理などを適切に行い、電気的特性を満たすように設計してください (入力専用ポートは個別に抵抗を介して V_{DD} 又は V_{SS} に接続して下さい)。
- 注意 2. EV_{SS} で始まる名前の端子がある場合には V_{SS} に、 EV_{DD} で始まる名前の端子がある場合には V_{DD} にそれぞれ接続してください。
- 注意 3. V_{DD} は LVD0 にて設定したリセット解除電圧 (V_{LVD0}) 以上にしてください。

3.2 使用端子一覧

表 3-1 にサンプル・プログラムで使用する端子と機能を示します。

表 3-1 使用端子一覧

端子名	入出力	内容
P30/SCK11	出力	CSI シリアル・クロック入力端子
P50/SI11	入力	CSI シリアル・データ受信用端子
P51/SO11	出力	CSI シリアル・データ送信用端子
P05	出力	外部フラッシュ・メモリの CS 用端子
P53	出力	LED1(フラッシュ・アクセス中を示す LED)の点灯/消灯
P137/INTP0	入力	書き込み開始
P60/SCLA0、P61/SDAA0	入出力	LCD モジュールとの I2C 通信

注意 本アプリケーションノートは、使用端子のみを端子処理しています。実際に回路を作成される場合は、端子処理などを適切に行い、電気的特性を満たすように設計してください。

4. ソフトウェア設定

4.1 オプション・バイトの設定一覧

表 4-1 にサンプル・プログラムで使用する、オプション・バイトの設定を示します。

表 4-1 オプション・バイト設定

アドレス	設定値	内容
000C0H/040C0H	11101111B	ウォッチドッグ・タイマ動作停止 (リセット解除後、カウント開始)
000C1H/040C1H	11111110B	LVD 検出電圧：リセット・モード 立ち上がり時 TYP. 1.90 V (1.84 V ~ 1.95 V) 立ち下がり時 TYP. 1.86 V (1.80 V ~ 1.91 V)
000C2H/040C2H	11101000B	HS モード、 高速オンチップ・オシレータ・クロック：32MHz
000C3H/040C3H	10000101B	オンチップ・デバッグ許可

4.2 スタートアップ・ルーチンの設定

4.2.1 スタック領域用セクション (.stack_bss) の定義

サンプル・プログラムでは、ファームウェア更新用データをローカル変数に保存します。ローカル変数はスタック領域に配置されることから、cstart.asm を修正して任意のスタック領域の確保およびスタック領域の初期化を行います。

```

; $IF (__RENESAS_VERSION__ < 0x01010000)
; -----
;   stack area
; -----
; !!! [CAUTION] !!!
; Set up stack size suitable for a project.
.section .stack_bss, BSS
_stackend:
    .DS      0x200
_stacktop:
; $ENDIF

; -----
; setting the stack pointer
; -----

; $IF (__RENESAS_VERSION__ >= 0x01010000)
;   MOVW    SP, #LOWW(__STACK_ADDR_START)
; $ELSE    ; for CC-RL V1.00
;   MOVW    SP, #LOWW(_stacktop)
; $ENDIF

; -----
; initializing stack area
; -----

; $IF (__RENESAS_VERSION__ >= 0x01010000)
;   MOVW    AX, #LOWW(__STACK_ADDR_END)
; $ELSE    ; for CC-RL V1.00
;   MOVW    AX, #LOWW(_stackend)
; $ENDIF
CALL    !!_stkinit

```

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

先頭行に ';' を追加しコメントアウト

4.2.2 書き換え用プログラムの RAM 領域への配置

ファームウェアの書き換えに使用するプログラムの RAM 領域へ配置します。

ファームウェアの書き換えに使用するプログラムは表 4-2 に記載されるセクションに配置されています。

表 4-2 セクション情報

セクション名	配置先セクション名	配置内容
RFD_CMN_f	RFD_CMN_fR	共通フラッシュ制御 API 関数のプログラム・セクション
RFD_CF_f	RFD_CF_fR	コード・フラッシュ・メモリ制御 API 関数のプログラム・セクション
RFD_EX_f	RFD_EX_fR	エクストラ領域制御 API 関数のプログラム・セクション
SMP_CMN_f	SMP_CMN_fR	共通フラッシュ制御 サンプル関数のプログラム・セクション
SMP_CF_f	SMP_CF_fR	コード・フラッシュ・メモリ制御サンプル関数のプログラム・セクション

上記セクションを RAM 領域に配置するために cstart.asm に処理を追加する必要があります。

cstart.asm 内の下記記述の後に処理を追加します。

```

;-----
; ROM data copy
;-----

```

追記する内容は以下の通りです。

```

; copy .text to RAM (セクション名)
MOV     C,#HIGHW(STARTOF(セクション名))
MOVW    HL,#LOWW(STARTOF(セクション名))
MOVW    DE,#LOWW(STARTOF(配置先セクション名))
BR      $.L12_TEXT
.Lm1_TEXT:
MOV     A,C
MOV     ES,A
MOV     A,ES:[HL]
MOV     [DE],A
INCW    DE
INCW    HL
CLRW    AX
CMPW    AX,HL
SKNZ
INC
.Lm2_TEXT:
MOVW    AX,HL
CMPW    AX,#LOWW(STARTOF(セクション名) + SIZEOF(セクション名))
BNZ     $.L11_TEXT

```

注意 1. **セクション名**には配置対象となるセクション名を 1 つ記載してください。

注意 2. 配置が必要なセクションの数だけ上記の記述を追加してください。

注意 3. **m** は任意の数値を設定してください。セクションごとに異なる数値を設定してください。

4.3 オンチップ・デバッグ・セキュリティ ID

RL78/G23 は、第三者からメモリの内容を読み取られないようにするために、フラッシュ・メモリの 000C4H-000CDH にオンチップ・デバッグ・セキュリティ ID 設定領域を用意しています。

セルフ・プログラミング時にブート・スワップを使用する場合は、000C4H-000CDH と 040C4H-040CDH が切り替わるので、040C4H-040CDH にも 000C4H-000CDH と同じ値を設定する必要があります。

4.4 サンプル・プログラム使用リソース

4.4.1 ROM 領域セクション一覧

表 4-3 にサンプル・プログラムで使用する ROM 領域のセクション一覧を示します。

表 4-3 ROM 領域セクション一覧

セクション名	配置内容
RFD_CMN_f	共通フラッシュ制御 API 関数のプログラム・セクション
RFD_CF_f	コード・フラッシュ・メモリ制御 API 関数のプログラム・セクション
RFD_EX_f	エクストラ領域制御 API 関数のプログラム・セクション
RFD_DF_f	データ・フラッシュ・メモリ制御 API 関数のプログラム・セクション
SMP_CMN_f	共通フラッシュ・メモリ制御 サンプル関数のプログラム・セクション
SMP_CF_f	コード・フラッシュ・メモリ制御サンプル関数のプログラム・セクション

4.4.2 RAM 領域セクション一覧

表 4-4 にサンプル・プログラムで使用する RAM 領域セクションの一覧を示します。

表 4-4 RAM 領域セクション一覧

セクション名	配置内容
RFD_DATA_nR	RFD RL78 Type01 のデータ・セクション
RFD_CMN_fR	共通フラッシュ・メモリ制御 API 関数のプログラム・セクション
RFD_CF_fR	コード・フラッシュ・メモリ制御 API 関数のプログラム・セクション
RFD_EX_fR	エクストラ領域制御 API 関数のプログラム・セクション
SMP_CMN_fR	共通フラッシュ・メモリ制御 サンプル関数のプログラム・セクション
SMP_CF_fR	コード・フラッシュ・メモリ制御サンプル関数のプログラム・セクション

4.5 定数一覧

表 4-5 にサンプル・プログラムで使用する定数を示します。

表 4-5 定数

定数名	設定値	内容
STR_VERSION	"V1.00"	ファームウェア バージョン情報
LED_ON	00H	LED ON
LED_OFF	01H	LED OFF
WAITCOUNT_32M	8000	HS 32MHz 動作時の 5ms カウント値
WRITE_START_ADDRESS	00080000H	コード・フラッシュ・メモリ先頭アドレス (ブート・クラスタ 1 を除く)
FLASH_READ_START_ADDRESS	00000000H	外部フラッシュ・メモリ読み出し先頭アドレス
WRITE_DATA_SIZE	0100H	コード・フラッシュ・メモリ書き込みサイズ (256 バイト)
CF_BLOCK_SIZE	0800H	コード・フラッシュ・メモリブロックサイズ (2048 バイト)
CPU_FREQUENCY	32	CPU 動作周波数
VALUE_U08_MASK1_FSQ_STATUS_ERR_CFDSEQUENCER	10H	フラッシュ・メモリ・シーケンサ実行結果のエラー・ステータス・マスク値
VALUE_U08_MASK1_FSQ_STATUS_ERR_EXTRA_SEQUENCER	20H	フラッシュ・メモリ・シーケンサ実行結果のエラー・ステータス・マスク値 bit5: エクストラ領域シーケンサ・エラー
VALUE_U08_MASK1_FSQ_STATUS_ERR_ERASE	01H	フラッシュ・メモリ・シーケンサ実行結果のエラー・ステータス・マスク値 bit0: 消去コマンド・エラー
VALUE_U08_MASK1_FSQ_STATUS_ERR_WRITE	02H	フラッシュ・メモリ・シーケンサ実行結果のエラー・ステータス・マスク値 bit1: 書き込みコマンド・エラー
VALUE_U08_MASK1_FSQ_STATUS_ERR_BLANKCHECK	08H	フラッシュ・メモリ・シーケンサ実行結果のエラー・ステータス・マスク値 bit3: ブランク・チェック・コマンド・エラー
VALUE_U32_DF_BASE_ADDR	000F1000H	DataFlash 開始アドレス
VALUE_U08_SHIFT_ADDR_TO_BLOCK_CF	11	CodeFlash のブロック番号算出時に行うビットシフト用定数
VALUE_U08_SHIFT_ADDR_TO_BLOCK_DF	8	DataFlash のブロック番号算出時に行うビットシフト用定数
VALUE_U01_MASK0_1BIT	0	0 固定値
VALUE_U01_MASK1_1BIT	1	1 固定値
VALUE_U08_MASK0_8BIT	00H	00H 固定値
VALUE_U08_MASK1_8BIT	FFH	FFH 固定値

4.6 データ型定義

表 4-6 にサンプル・プログラムで使用する列挙型の定義を示します。

表 4-6 enum e_ret (列挙変数名: e_ret_t)

Symbol Name	値	内容
ENUM_RET_STS_OK	00H	ステータス正常
ENUM_RET_ERR_CFDI_SEQUENCER	10H	コード/データ・フラッシュ・メモリシーケンサ・エラー
ENUM_RET_ERR_EXTRA_SEQUENCER	11H	エクストラ領域シーケンサ・エラー
ENUM_RET_ERR_ERASE	12H	消去エラー
ENUM_RET_ERR_WRITE	13H	書き込みエラー
ENUM_RET_ERR_BLANKCHECK	14H	ブランク・エラー
ENUM_RET_ERR_CHECK_WRITE_DATA	15H	書き込みデータのリード値比較エラー
ENUM_RET_ERR_MODE_MISMATCHED	16H	モード不一致エラー
ENUM_RET_ERR_PARAMETER	17H	パラメータ・エラー
ENUM_RET_ERR_CONFIGURATION	18H	デバイス構成・エラー

4.7 変数一覧

表 4-7 にサンプル・プログラムで使用するグローバル変数を示します。

表 4-7 グローバル変数

型	変数名	内容	使用関数
uint8_t	g_csi11_end	CSI11 通信完了フラグ	r_Config_CSI11_callback_receiveend、 r_Config_CSI11_callback_sendend
uint8_t	f_update_start	アップデート開始フラグ	main、r_Config_INTC_intp0_interrupt

4.8 関数一覧

表 4-8 にサンプル・プログラムで使用する関数を示します。

表 4-8 関数

関数名	概要
r_rfd_initialize	RFD RL78 Type01 初期化处理
r_CF_BlockControlSequence	コード・フラッシュ・メモリ ブロック制御処理
r_CF_WriteVerifySequence	コード・フラッシュ・メモリ 書き込み・ベリファイ処理
r_CF_EraseBlock	コード・フラッシュ・メモリ ブロック消去処理
r_CF_WriteData	コード・フラッシュ・メモリ 書き込み処理
r_CF_VerifyData	コード・フラッシュ・メモリ ベリファイ処理
r_CheckCFDFSequencerEnd	コード・フラッシュ・メモリ シーケンス終了処理
r_Config_CSI11_callback_sendend	CSI11の送信完了割り込み時のコールバック処理
r_Config_CSI11_callback_receiveend	CSI11の受信完了割り込み時のコールバック処理
r_CSI11_GetData	外部フラッシュ・メモリ データ取得処理
r_Config_IICA0_callback_master_sendend	IICA0の送信完了割り込み時のコールバック処理
r_Config_IICA0_callback_master_error	IICA0の送信エラー割り込み時のコールバック処理
r_LCM_init	LCDモジュール 初期化处理
r_LCM_clear	LCDモジュール 表示消去処理クリアディスプレイ
r_LCM_send_string	LCDモジュール 文字列送信処理
r_LCM_send_command	LCDモジュール コマンド送信処理
r_LCM_send_data	LCDモジュール データ送信処理
r_LCM_turn_sendend_on	LCDモジュール 通信終了フラグ設定
r_LCM_wait_sendend	LCDモジュール 通信終了待ち処理
r_Config_INTC_intp0_interrupt	外部割り込み処理
R_RFD_Init※	RFD RL78 Type01の初期化处理
R_RFD_SetFlashMemoryMode※	フラッシュ・メモリ制御モード変更処理
R_RFD_CheckCFDFSeqEndStep1※	コード／データ・フラッシュ領域シーケンサの動作終了確認処理
R_RFD_CheckCFDFSeqEndStep2※	フラッシュ・メモリ・シーケンサ制御レジスタのクリアにより、コマンド動作が終了したかの確認処理
R_RFD_ClearSeqRegister※	コード／データ・フラッシュ領域シーケンサ、エクストラ領域シーケンサ制御を行うレジスタのクリア処理
R_RFD_GetSeqErrorStatus※	コード／データ・フラッシュ領域シーケンサ・コマンド、または、エクストラ領域シーケンサ・コマンドにより、発生したエラー情報の取得処理
R_RFD_ForceReset※	CPUの内部リセット要求
R_RFD_EraseCodeFlashReq※	コード・フラッシュ消去処理
R_RFD_WriteCodeFlashReq※	コード・フラッシュ書き込み処理

※ フラッシュ・セルフ・プログラミング・コードで定義されている API 関数です。

関数の詳細は "RL78 ファミリ Renesas Flash Driver RL78 Type01 ユーザーズマニュアル" を参照してください

4.9 関数仕様

サンプルコードの関数仕様を示します。

r_rfd_initialize	
概 要	RFD RL78 Type01 初期化处理
ヘッダ	r_rfd_common_api.h、r_rfd_code_flash_api.h、r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_rfd_initialize(void);
説 明	RFD RL78 Type01 の初期化处理を行います。
引 数	なし
リターン値	ENUM_RET_STS_OK: 正常終了 ENUM_RET_ERR_CONFIGURATION: クロック構成エラー ENUM_RET_ERR_PARAMETER: 周波数設定エラー

r_CF_BlockControlSequence	
概 要	コード・フラッシュ・メモリ ブロック制御処理
ヘッダ	r_rfd_common_api.h、r_rfd_code_flash_api.h、r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_CF_BlockControlSequence(uint32_t write_start_addr, int32_t flash_read_addr);
説 明	コード・フラッシュ・メモリに対してデータ消去・書き込み・ベリファイ処理を行います。制御単位は引数で指定されたアドレスから 1 ブロック分となります。
引 数	uint32_t write_start_addr: 消去・書き込み・ベリファイの開始アドレス int32_t flash_read_addr: 外部フラッシュ・メモリ読み出し先頭アドレス
リターン値	ENUM_RET_STS_OK: 正常終了 ENUM_RET_ERR_MODE_MISMATCHED: モード不一致エラー ENUM_RET_ERR_ERASE: 消去エラー

r_CF_WriteVerifySequence	
概 要	コード・フラッシュ・メモリ 書き込み・ベリファイ処理
ヘッダ	r_rfd_common_api.h、r_rfd_code_flash_api.h、r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_CF_WriteVerifySequence(uint32_t write_start_addr, uint16_t write_data_length, uint8_t __near * write_data);
説 明	コード・フラッシュ・メモリに対して書き込み・ベリファイ処理を行います。
引 数	uint32_t write_start_addr: 書き込み開始アドレス uint16_t write_data_length: 書き込みサイズ uint8_t __near * write_data: 書き込みデータ
リターン値	ENUM_RET_STS_OK: 正常終了 ENUM_RET_ERR_MODE_MISMATCHED: モード不一致エラー ENUM_RET_ERR_ERASE: 消去エラー

r_CF_EraseBlock	
概 要	コード・フラッシュ・メモリ ブロック消去処理
ヘッダ	r_rfd_common_api.h、r_rfd_code_flash_api.h、r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_CF_EraseBlock(uint32_t i_u32_start_addr); コード・フラッシュ・メモリに対してデータ消去を行います。
説 明	引数で指定されたアドレスが含まれるブロックを対象とし、ブロック単位で消去処理が実施されます。
引 数	uint32_t i_u32_start_addr: 消去アドレス ENUM_RET_STS_OK: 正常終了
リターン値	ENUM_RET_ERR_MODE_MISMATCHED: モード不一致エラー ENUM_RET_ERR_ERASE: 消去エラー

r_CF_WriteData	
概 要	コード・フラッシュ・メモリ 書き込み処理
ヘッダ	r_rfd_common_api.h、r_rfd_code_flash_api.h、r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_CF_WriteData(uint32_t start_addr, uint16_t write_data_length, uint8_t __near * write_data);
説 明	コード・フラッシュ・メモリに対して書き込みを行います。 uint32_t start_addr: 書き込み開始アドレス
引 数	uint16_t write_data_length: 書き込みサイズ uint8_t __near * write_data: 書き込みデータ ENUM_RET_STS_OK: 正常終了
リターン値	ENUM_RET_ERR_MODE_MISMATCHED: モード不一致エラー ENUM_RET_ERR_WRITE: 書き込みエラー

r_CF_VerifyData	
概 要	コード・フラッシュ・メモリ ベリファイ処理
ヘッダ	r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_CF_VerifyData(uint32_t start_addr, uint16_t data_length, uint8_t __near * write_data);
説 明	コード・フラッシュ・メモリに書き込まれたデータに対してベリファイ処理を行います。 uint32_t start_addr: ベリファイ開始アドレス
引 数	uint16_t data_length: データサイズ uint8_t __near * write_data: 比較データ ENUM_RET_STS_OK: 正常終了 (一致)
リターン値	ENUM_RET_ERR_CHECK_WRITE_DATA: 書き込みデータのリード値比較エラー (不一致)

r_CheckCFDFSequencerEnd

概 要	コード・フラッシュ・メモリ シーケンス終了処理
ヘッダ	r_rfd_common_api.h、r_cg_userdefine.h
宣 言	R_RFD_FAR_FUNC e_ret_t r_CheckCFDFSequencerEnd(void);
説 明	コード・フラッシュ・メモリシーケンサの動作終了を確認します。
引 数	なし
リターン値	ENUM_RET_STS_OK: 正常終了 ENUM_RET_ERR_CFDF_SEQUENCER: コード/データ・フラッシュ・メモリシーケンサ・エラー ENUM_RET_ERR_ERASE: 消去エラー ENUM_RET_ERR_WRITE: 書き込みエラー ENUM_RET_ERR_BLANKCHECK: ブランク・エラー

r_Config_CSI11_callback_sendend

概 要	CSI11 送信完了割り込み時のコールバック関数
ヘッダ	E2PROM_driver.h
宣 言	static void r_Config_CSI11_callback_sendend(void)
説 明	CSI11 の送信完了割り込み時に呼ばれるコールバック関数です。
引 数	なし
リターン値	なし

r_Config_CSI11_callback_receiveend

概 要	CSI11 受信完了割り込み時のコールバック関数
ヘッダ	E2PROM_driver.h
宣 言	static void r_Config_CSI11_callback_receiveend(void)
説 明	CSI11 の受信完了割り込み時に呼ばれるコールバック関数です。
引 数	なし
リターン値	なし

r_CSI11_GetData

概 要	外部フラッシュ・メモリ データ取得処理
ヘッダ	Config_CSI11.h、FlashMemory_driver.h
宣 言	void r_CSI11_GetData(uint32_t addr, uint16_t rx_len, uint8_t * rx_buf)
説 明	外部フラッシュ・メモリよりファームウェア・アップデート用データを取得します。 uint32_t addr: 外部フラッシュ・メモリ 取得開始アドレス
引 数	uint8_t * rx_buf: 取得データ格納先バッファのポインタ uint16_t rx_len: 取得サイズ
リターン値	なし

r_Config_IICA0_callback_master_sendend

概 要	IICA0送信完了割り込み時の処理
ヘッダ	r_cg_macrodriver.h、Config_IICA0.h、LCM_driver.h
宣 言	static void r_Config_IICA0_callback_master_receiveend(void);
説 明	IICA0 の送信完了割り込み時に呼ばれるコールバック関数です。
引 数	なし
リターン値	なし

r_Config_IICA0_callback_master_error

概 要	IICA0送信エラー発生時処理
ヘッダ	r_cg_macrodriver.h、Config_IICA0.h、LCM_driver.h
宣 言	static void r_Config_IICA0_callback_master_error(MD_STATUS flag);
説 明	IICA0 の送信エラー割り込み時に呼ばれるコールバック関数です。
引 数	MD_STATUS flag：エラータイプ
リターン値	なし

r_LCM_init

概 要	LCDモジュール初期化
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	void r_LCM_init(void);
説 明	LCD モジュールを初期化します。
引 数	なし
リターン値	なし

r_LCM_clear

概 要	LCDモジュール 表示消去処理
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	void r_LCM_clear(void);
説 明	LCD モジュールに表示消去処理のコマンドを送信します。
引 数	なし
リターン値	なし

r_LCM_send_string

概 要	LCDモジュール 文字列送信処理
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	void r_LCM_send_string(uint8_t * const str, lcm_position_t pos);
説 明	LCD モジュールに str で渡された文字列を表示します。 表示させるラインは pos で指定します。 uint8_t * const str：表示させる文字列
引 数	lcm_position_t pos：LCM_POSITION_TOP で上段に表示 LCM_POSITION_BOTTOM で下段に表示
リターン値	なし

r_LCM_send_command

概 要	LCDモジュール コマンド送信処理
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	void r_LCM_send_command(uint8_t command);
説 明	LCD モジュールに command で渡されたコマンドを送信します。
引 数	uint8_t command：LCD モジュールへ送信するコマンド
リターン値	なし

r_LCM_send_data

概 要	LCDモジュール データ送信処理
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	void r_LCM_send_data(uint8_t data);
説 明	LCD モジュールに data で渡されたデータを送信します。
引 数	uint8_t data : LCD モジュールへ送信するデータ
リターン値	なし

r_LCM_turn_sendend_on

概 要	LCDモジュール 通信終了フラグ設定
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	void r_LCM_turn_sendend_on(void);
説 明	g_LCM_is_sendend に LCD モジュールとの IIC 通信終了フラグを設定します。
引 数	なし
リターン値	なし

r_LCM_wait_sendend

概 要	LCDモジュール 通信終了待ち処理
ヘッダ	LCM_driver.h、Config_IICA0.h
宣 言	static void r_LCM_wait_sendend(void);
説 明	LCD モジュールとの IIC 通信が終了するまで待ち、コマンド実行ウエイト時間 (5ms) だけウエイトを実行します。
引 数	なし
リターン値	なし

r_Config_INTC_intp0_interrupt

概 要	外部割込み処理
ヘッダ	r_cg_macrodriver.h、r_cg_userdefine.h、Config_INTC.h
宣 言	static void __near r_Config_INTC_intp0_interrupt (void)
説 明	スイッチ押下によって、アップデート開始フラグに 1 を設定します。
引 数	なし
リターン値	なし

4.10 フローチャート

4.10.1 メイン処理

図 4-1、図 4-2 にメイン処理のフローチャートを示します。

図 4-1 メイン処理 (1/2)

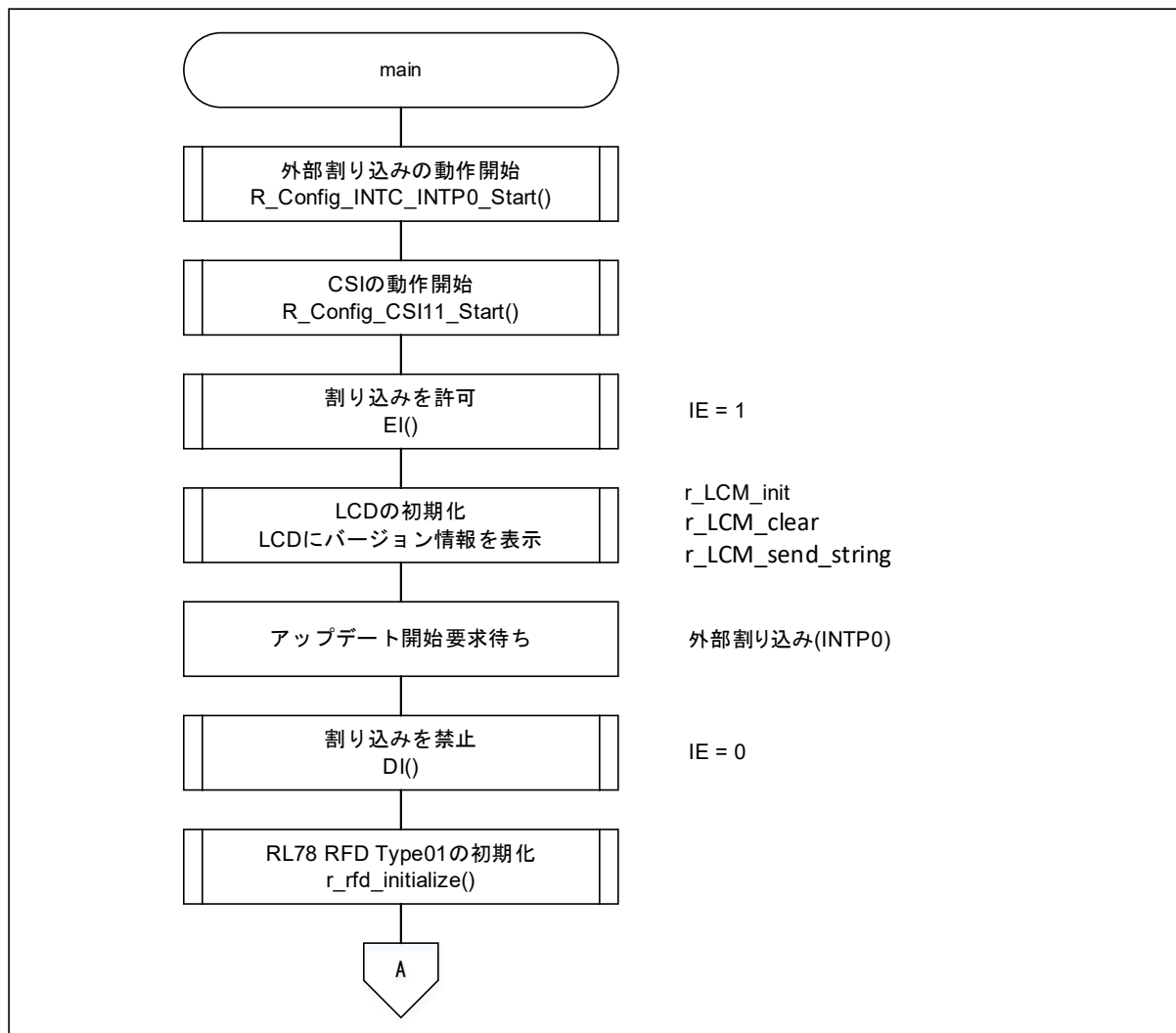
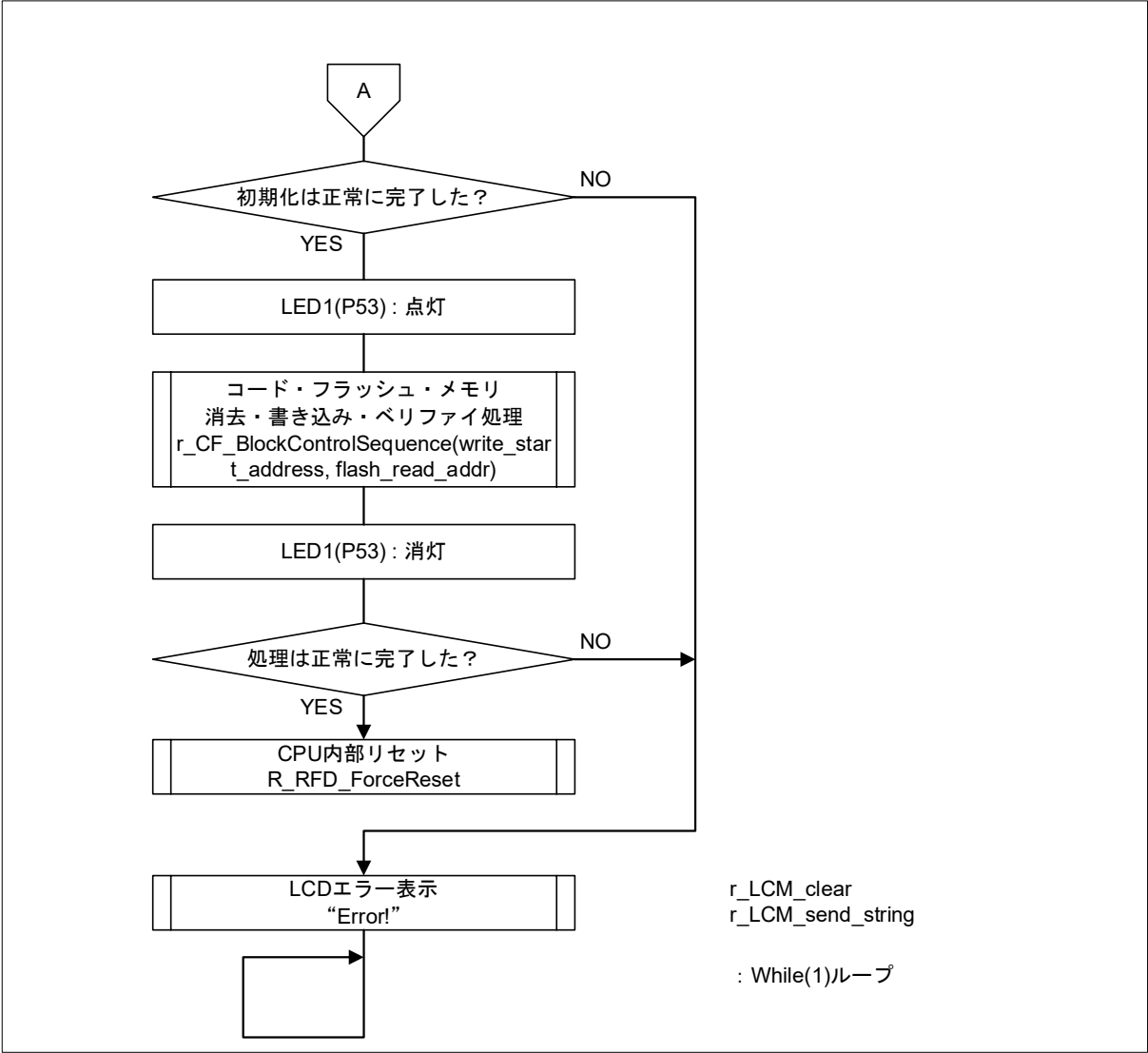


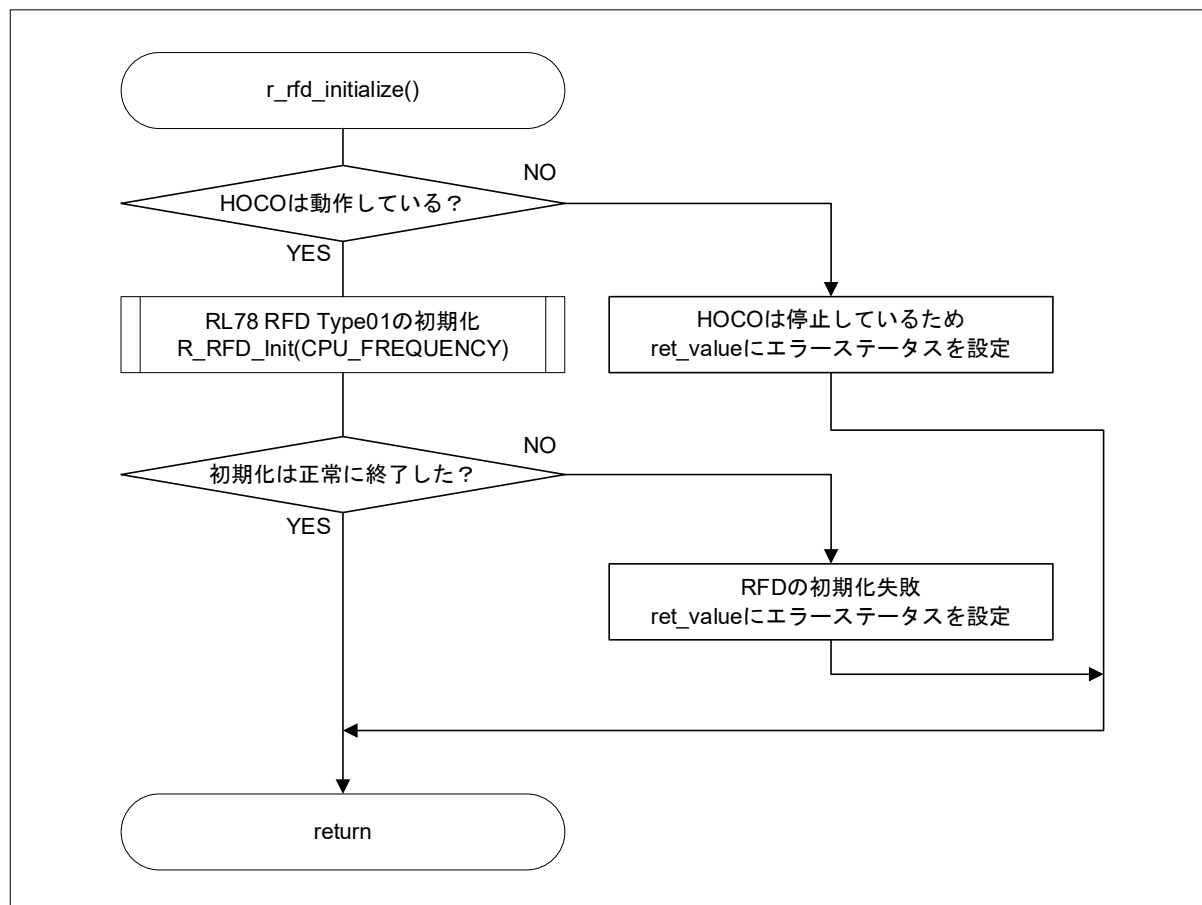
図 4-2 メイン処理 (2/2)



4.10.2 RFD RL78 Type01 初期化処理

図 4-3 に RFD RL78 Type01 初期化処理のフローチャートを示します。

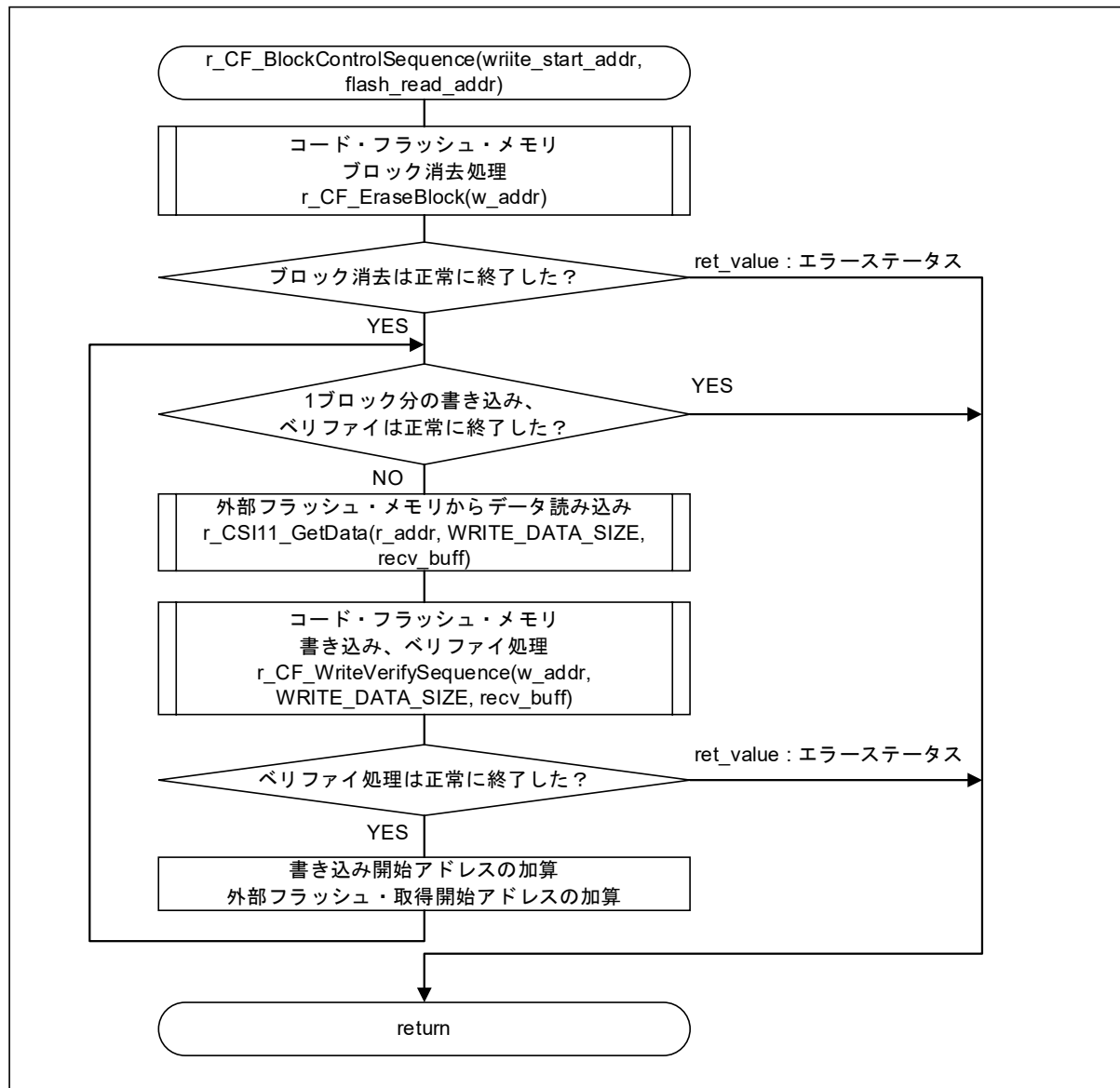
図 4-3 RFD RL78 Type01 初期化処理



4.10.3 コード・フラッシュ・メモリ ブロック制御処理

図 4-4 にコード・フラッシュ・メモリ ブロック制御処理のフローチャートを示します。

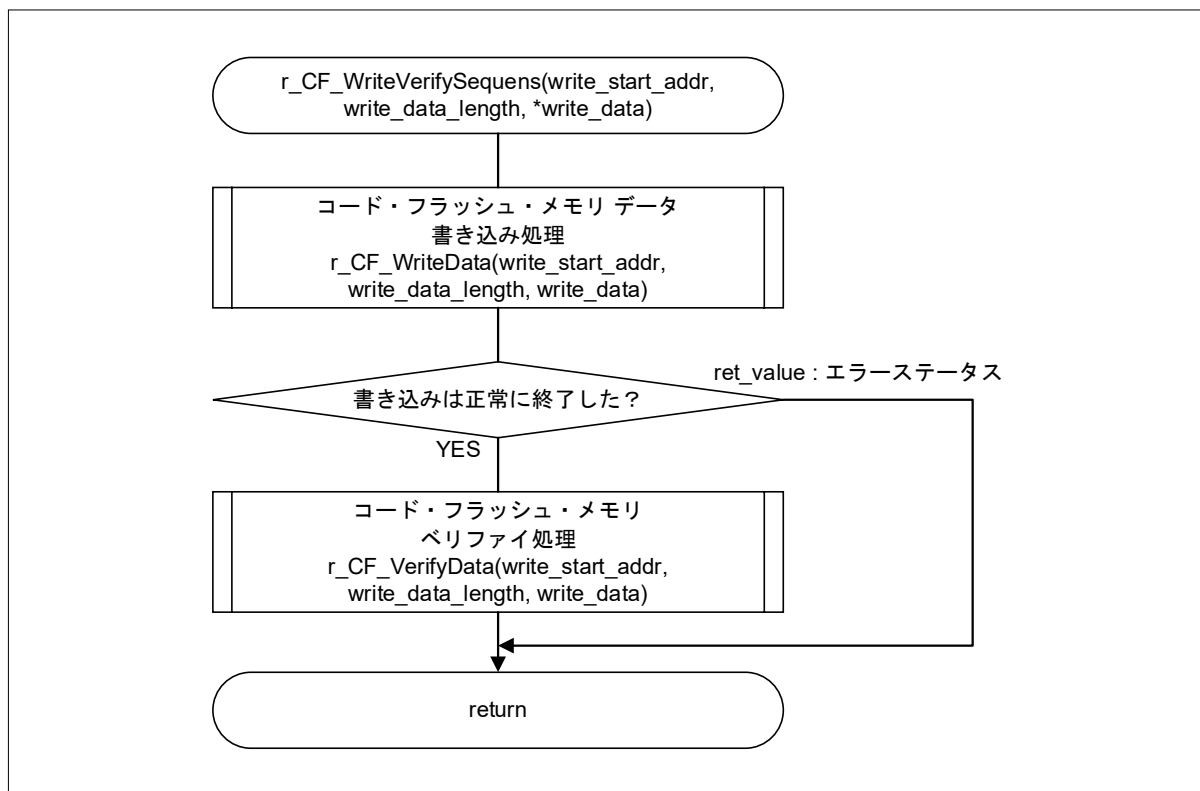
図 4-4 コード・フラッシュ・メモリ ブロック制御処理



4.10.4 コード・フラッシュ・メモリ 書き込み・ベリファイ処理

図 4-5 にコード・フラッシュ・メモリ 書き込み・ベリファイ処理のフローチャートを示します。

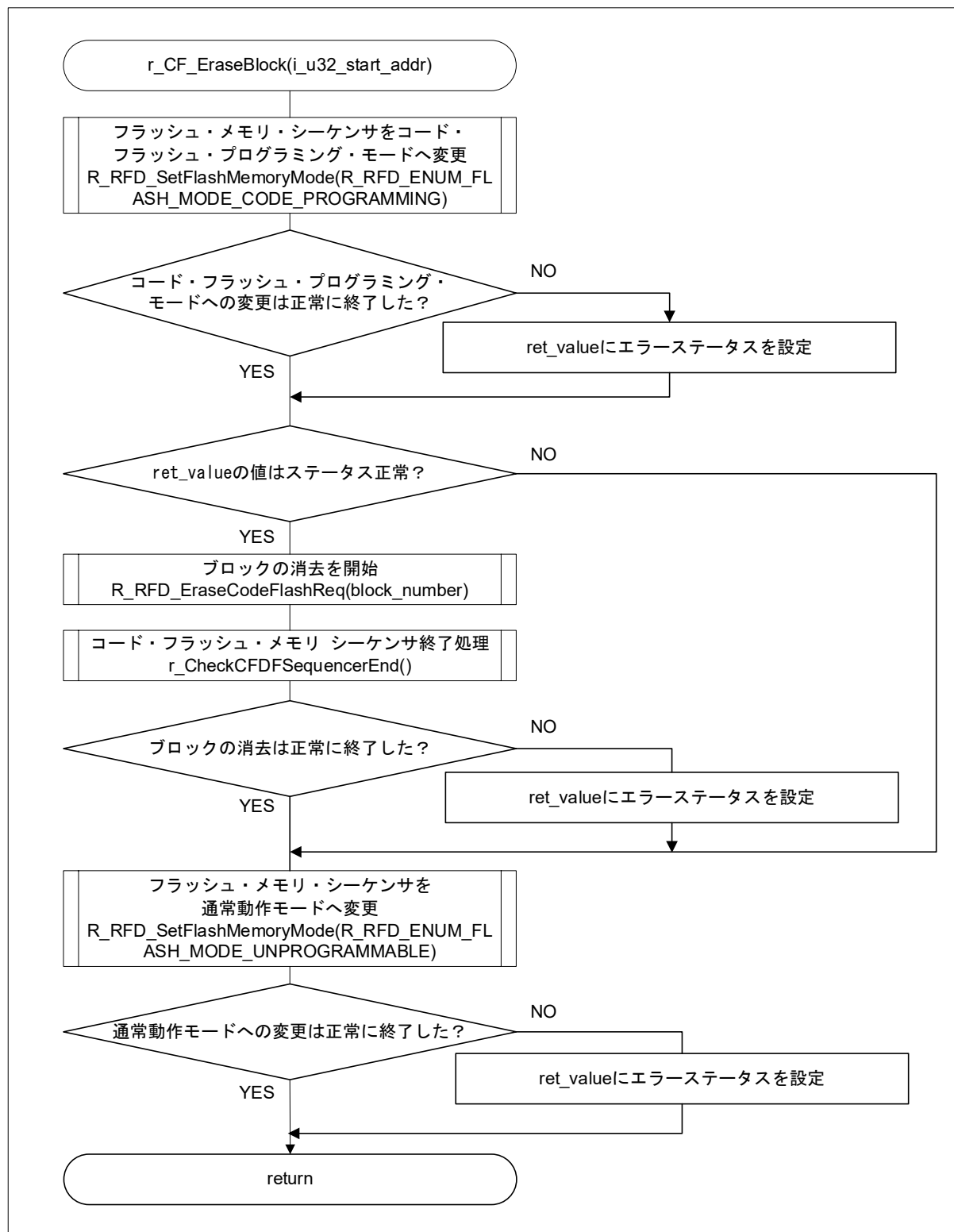
図 4-5 コード・フラッシュ・メモリ 書き込み・ベリファイ処理



4.10.5 コード・フラッシュ・メモリ ブロック消去処理

図 4-6 にコード・フラッシュ・メモリ ブロック消去処理のフローチャートを示します。

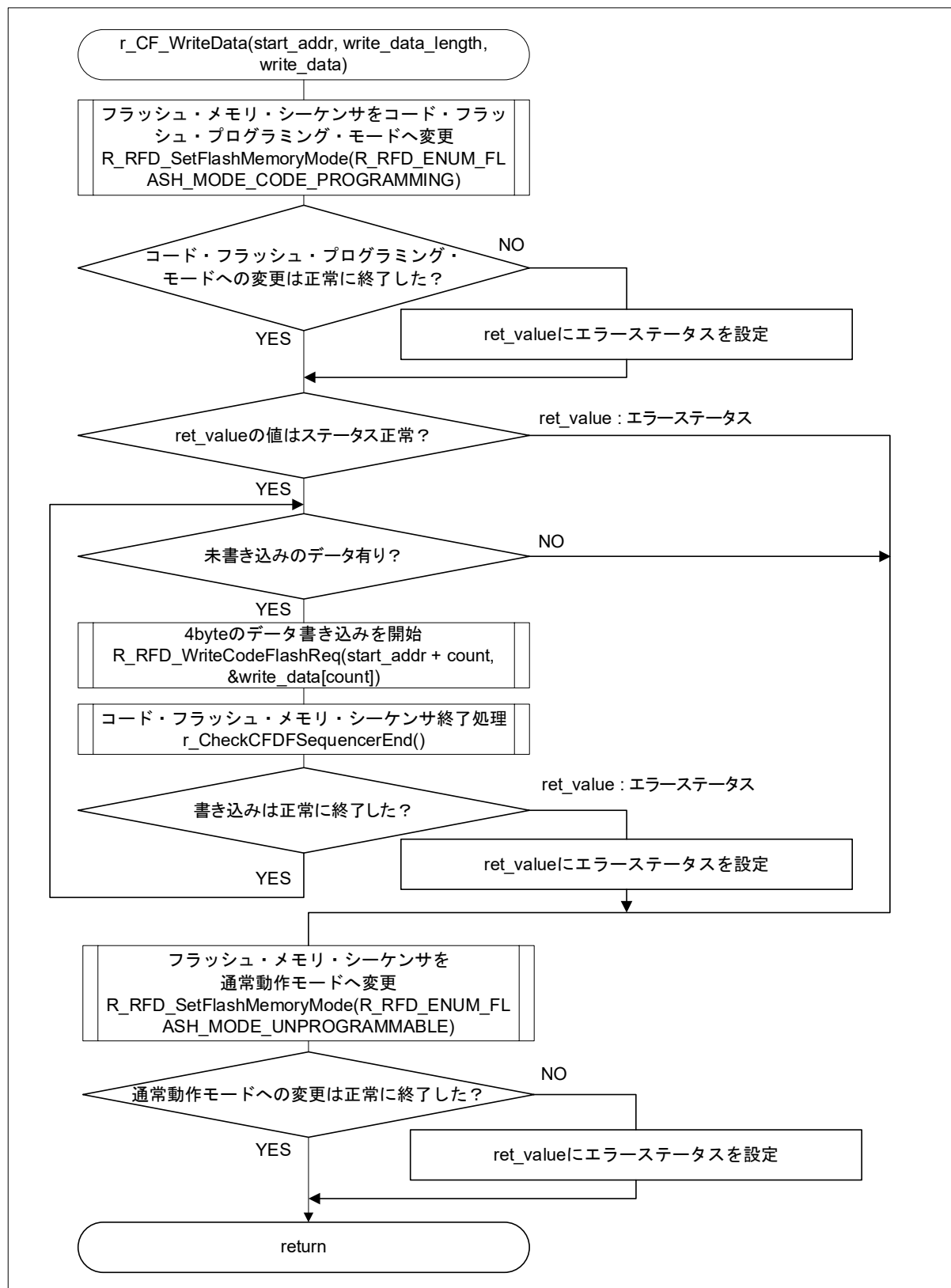
図 4-6 コード・フラッシュ・メモリ ブロック消去処理



4.10.6 コード・フラッシュ・メモリ 書き込み処理

図 4-7 にコード・フラッシュ・メモリ 書き込み処理のフローチャートを示します。

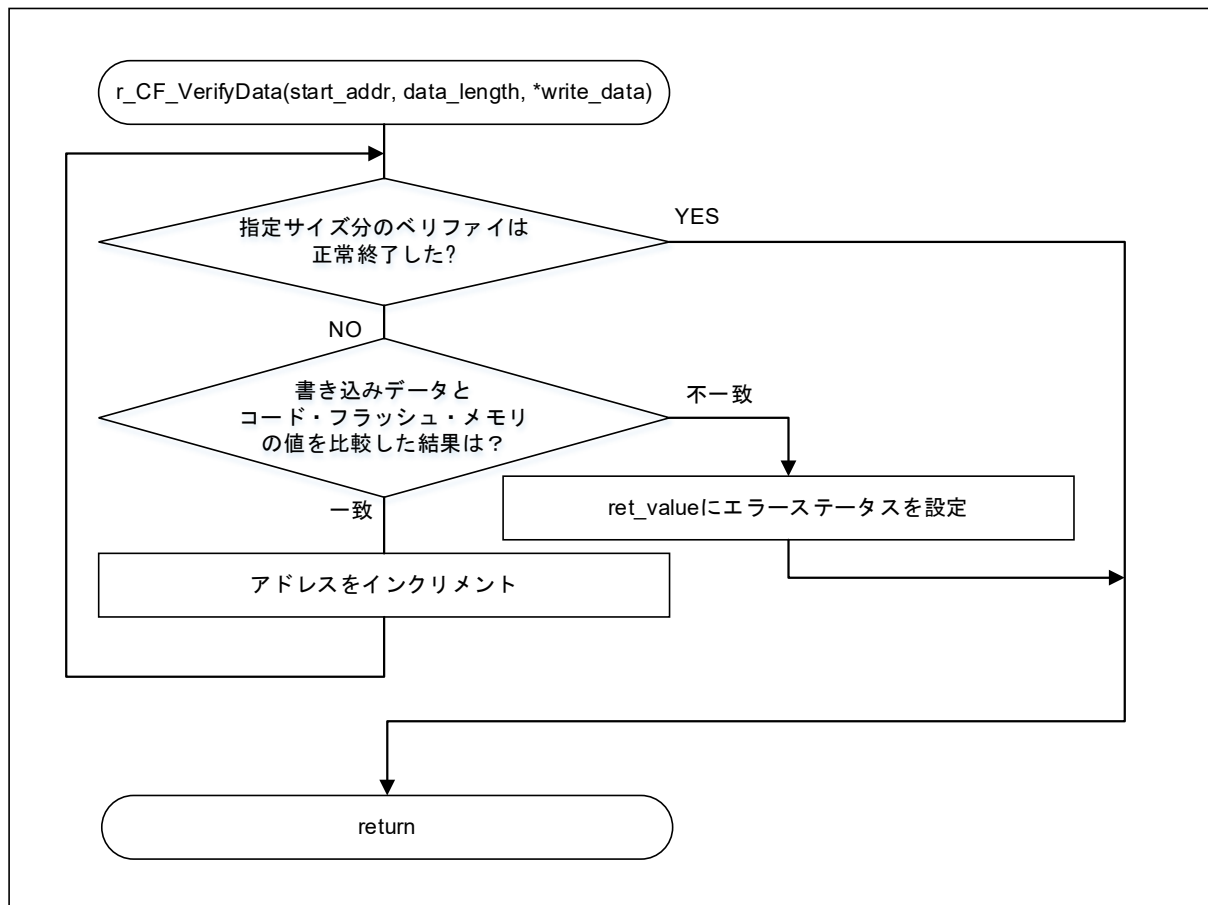
図 4-7 コード・フラッシュ・メモリ 書き込み処理



4.10.7 コード・フラッシュ・メモリ ベリファイ処理

図 4-8 にコード・フラッシュ・メモリ ベリファイ処理のフローチャートを示します。

図 4-8 コード・フラッシュ・メモリ ベリファイ処理



4.10.8 コード・フラッシュ・メモリ シーケンサ制御終了処理

図 4-9、図 4-10 にコード・フラッシュ・メモリ シーケンサ制御終了処理のフローチャートを示します。

図 4-9 コード・フラッシュ・メモリ シーケンサ制御終了処理 (1/2)

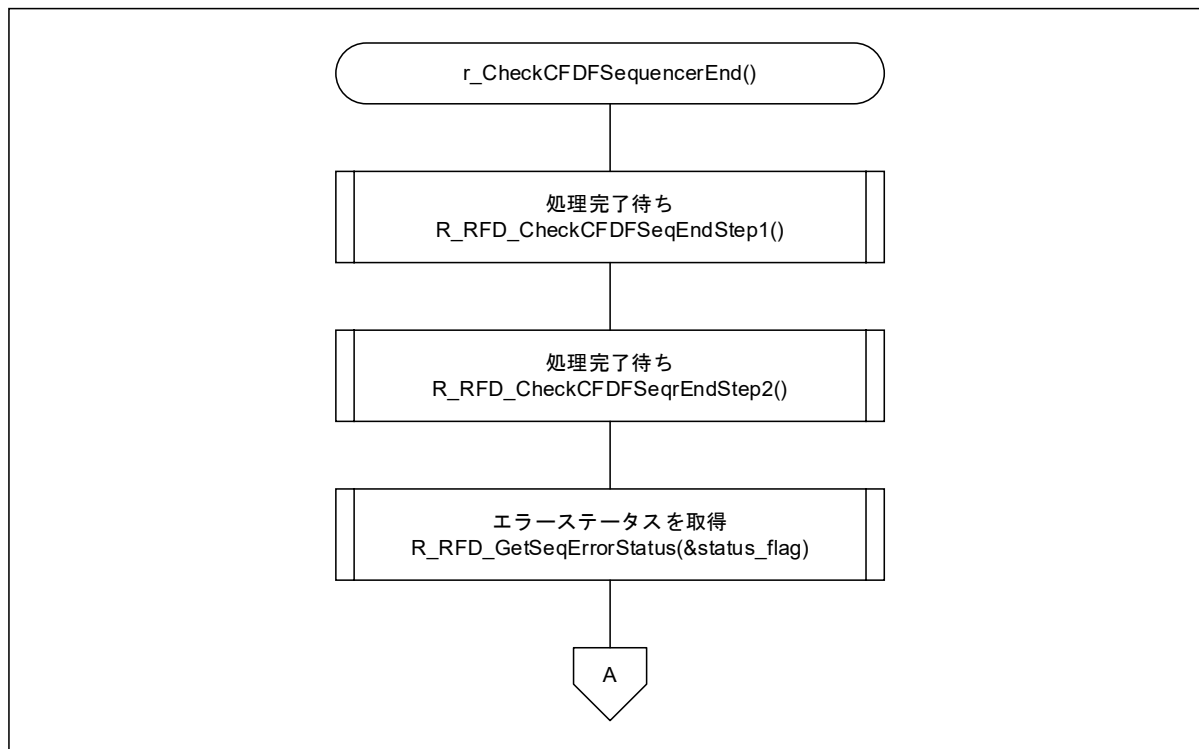
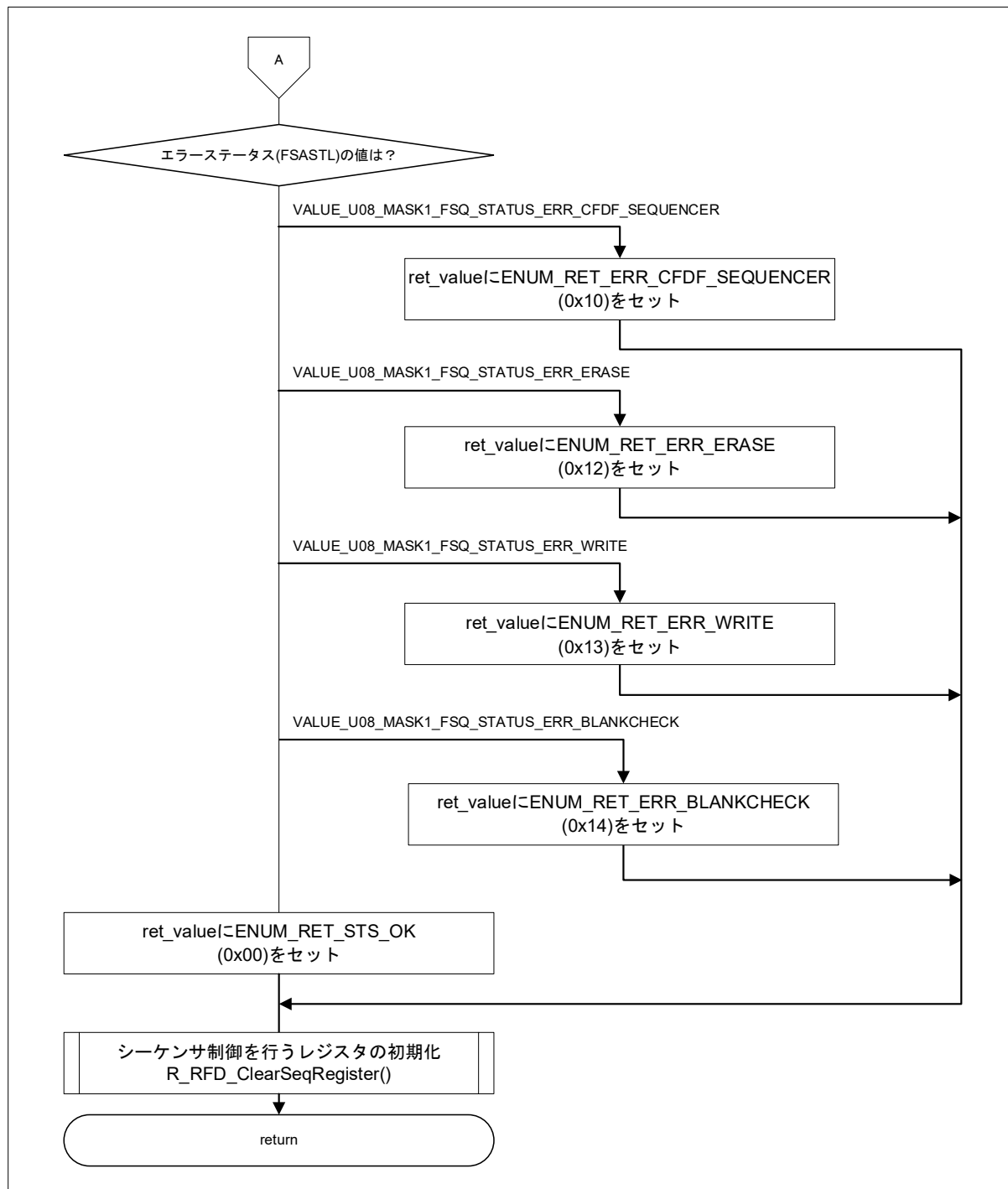


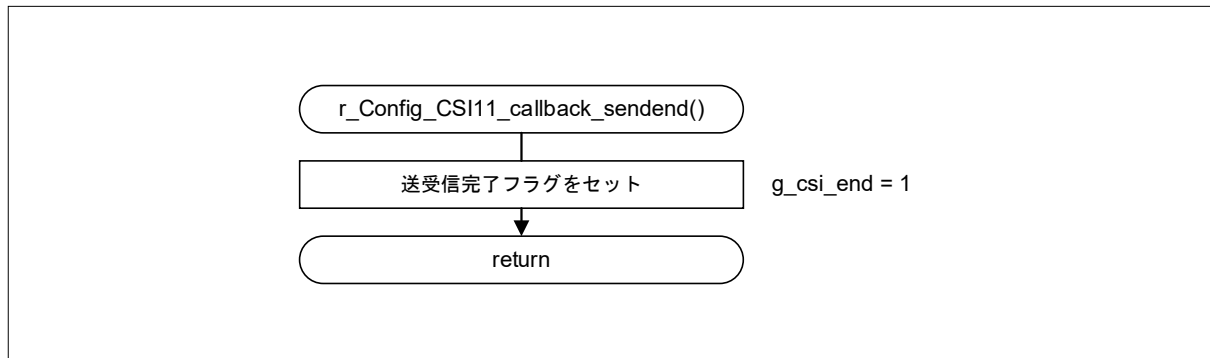
図 4-10 コード・フラッシュ・メモリ シーケンサ制御終了処理 (2/2)



4.10.9 CSI11 送信完了割り込み時の処理

図 4-11 に CSI11 送信完了割り込み時の処理のフローチャートを示します。

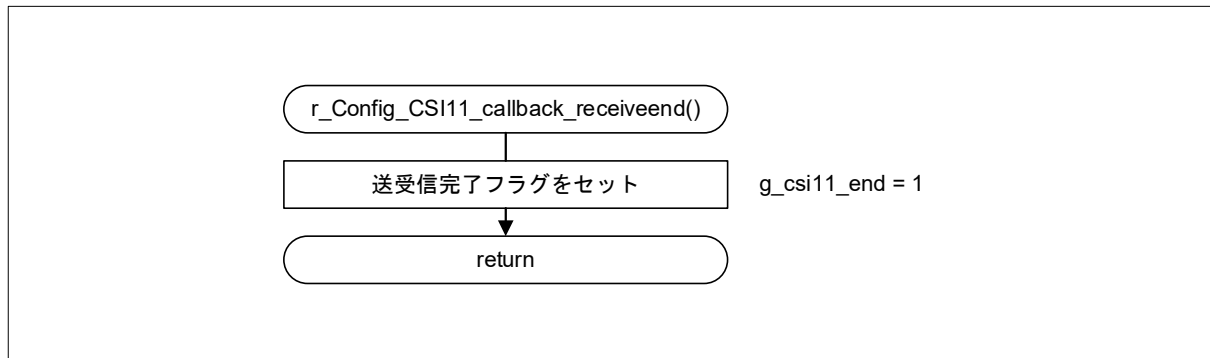
図 4-11 CSI11 送信完了割り込み時の処理



4.10.10 CSI11 受信完了割り込み時の処理

図 4-12 に CSI11 受信完了割り込み時の処理のフローチャートを示します。

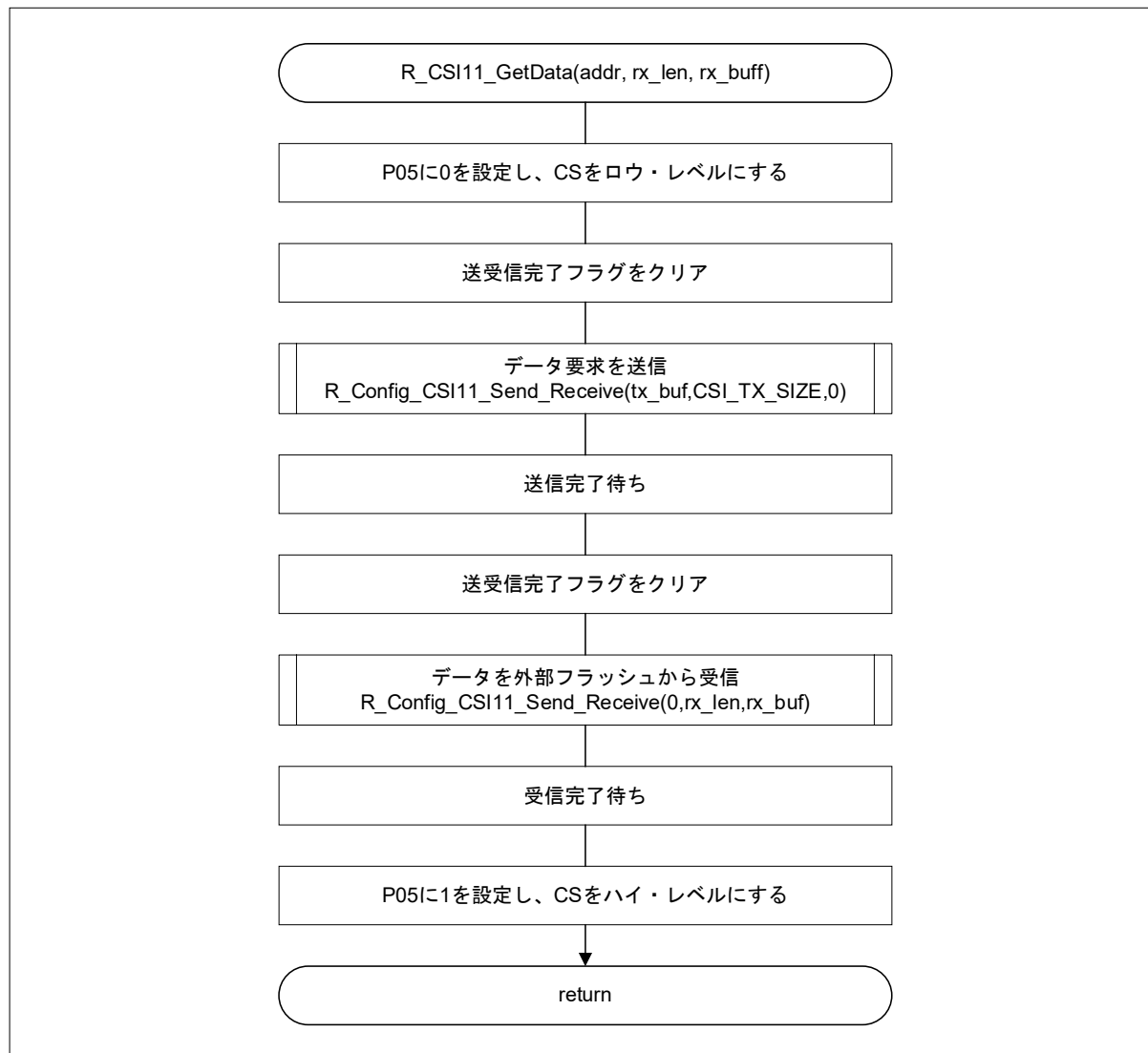
図 4-12 CSI11 受信完了割り込み時の処理



4.10.11 外部フラッシュ・メモリ データ取得処理

図 4-13 に外部フラッシュ・メモリ データ取得処理のフローチャートを示します。

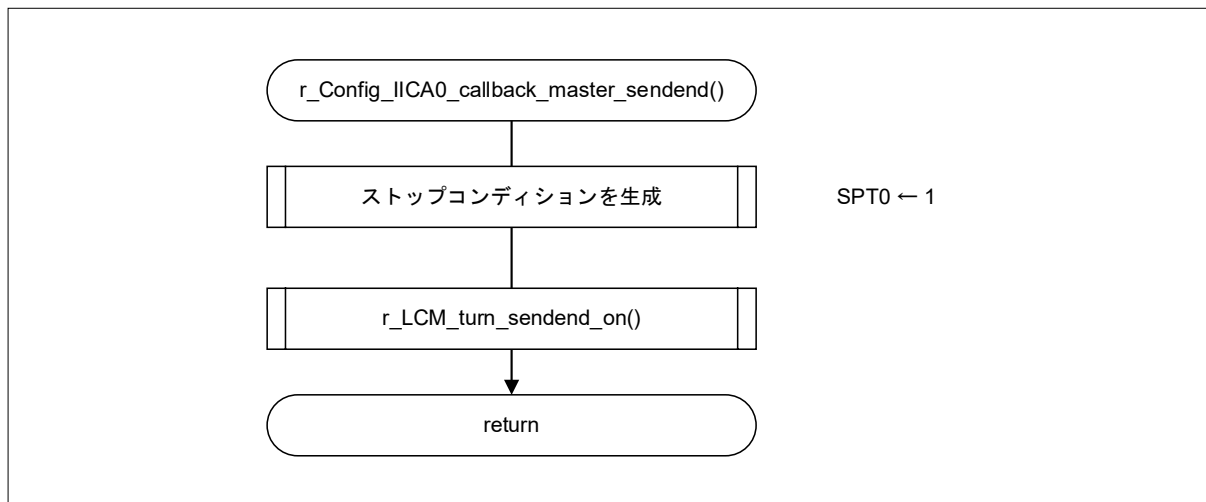
図 4-13 外部フラッシュ・メモリ データ取得処理



4.10.12 IICA0 送信完了割り込み時の処理

図 4-14 に IICA0 送信完了割り込み時の処理のフローチャートを示します。

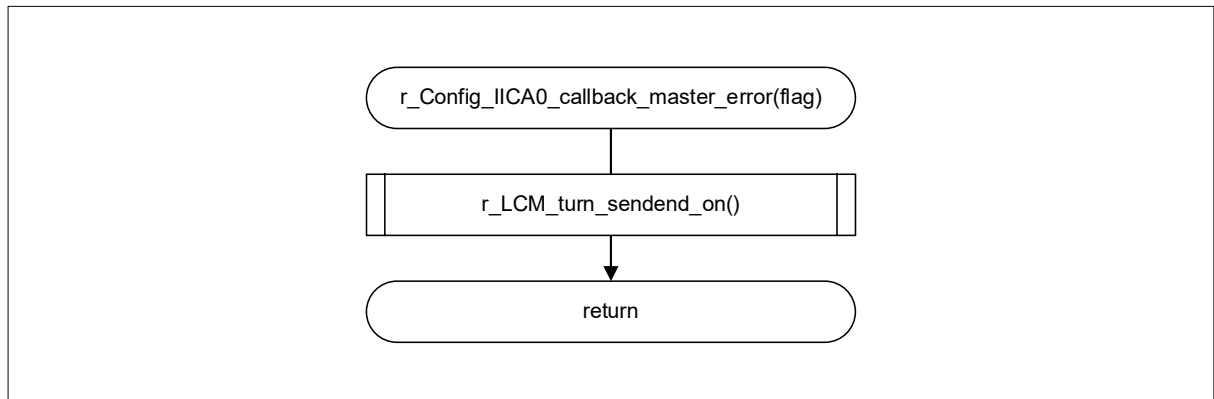
図 4-14 IICA0 送信完了割り込み時の処理



4.10.13 IICA0 送信エラー発生時の処理

図 4-15 に IICA0 送信エラー発生時の処理のフローチャートを示します。

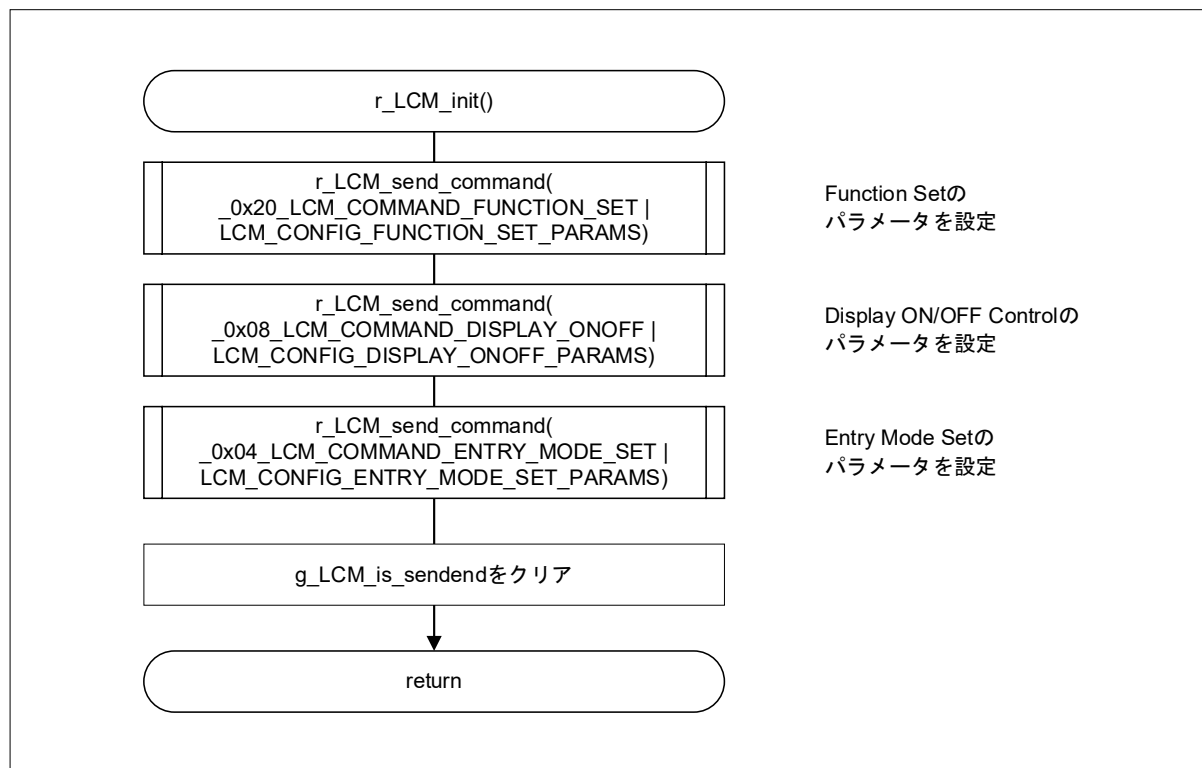
図 4-15 IICA0 送信エラー発生時の処理



4.10.14 LCD モジュール 初期化

図 4-16 に LCD モジュール 初期化のフローチャートを示します。

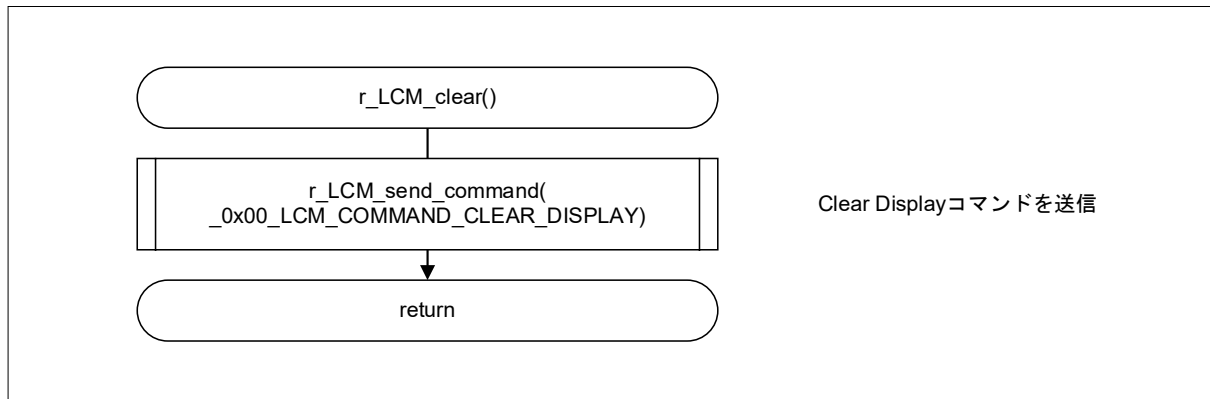
図 4-16 LCD モジュール 初期化



4.10.15 LCD モジュール 表示消去処理

図 4-17 に LCD モジュール 表示消去処理のフローチャートを示します。

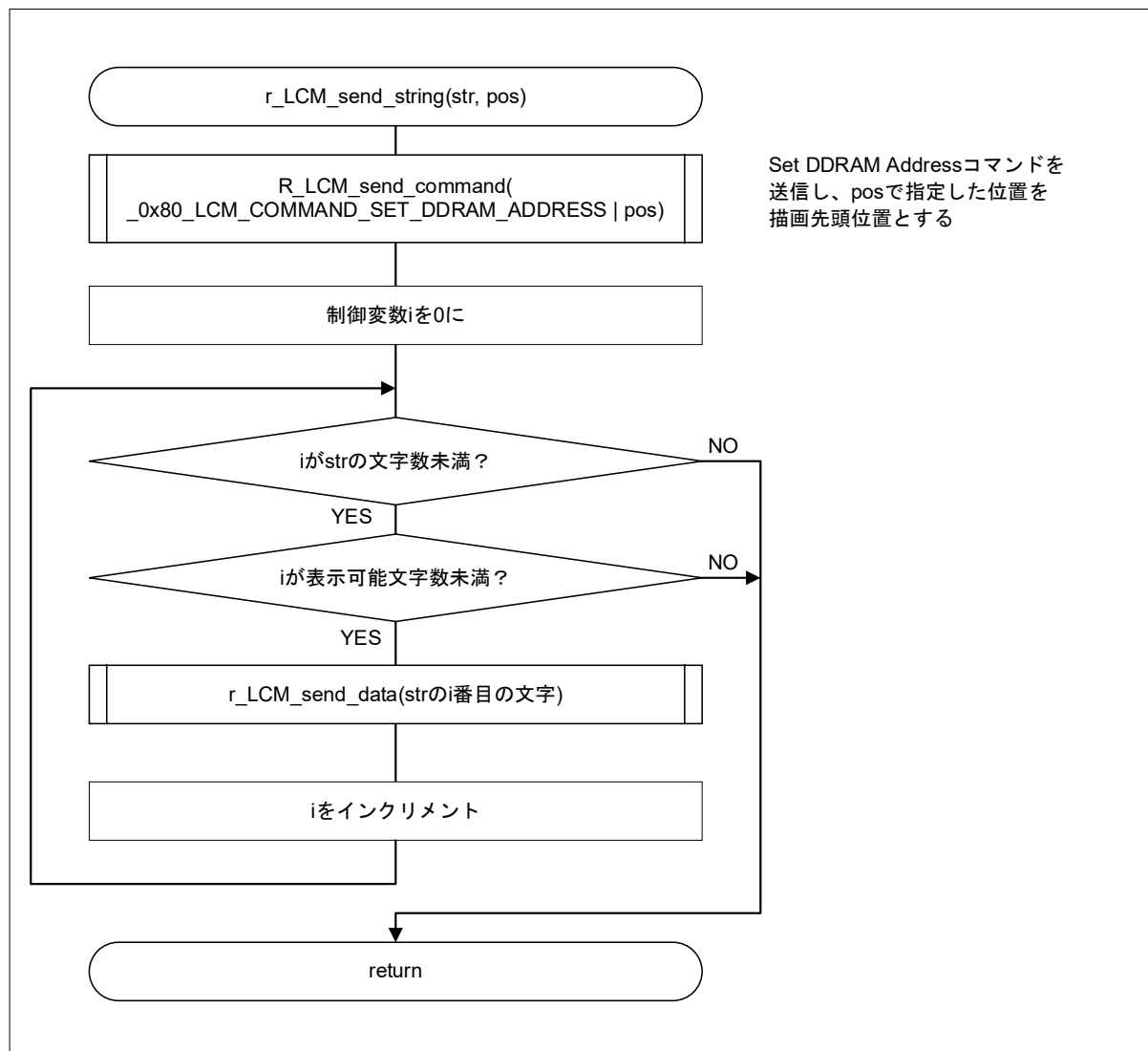
図 4-17 LCD モジュール 表示消去処理



4.10.16 LCD モジュール 文字列送信処理

図 4-18 に LCD モジュール 文字列送信処理のフローチャートを示します。

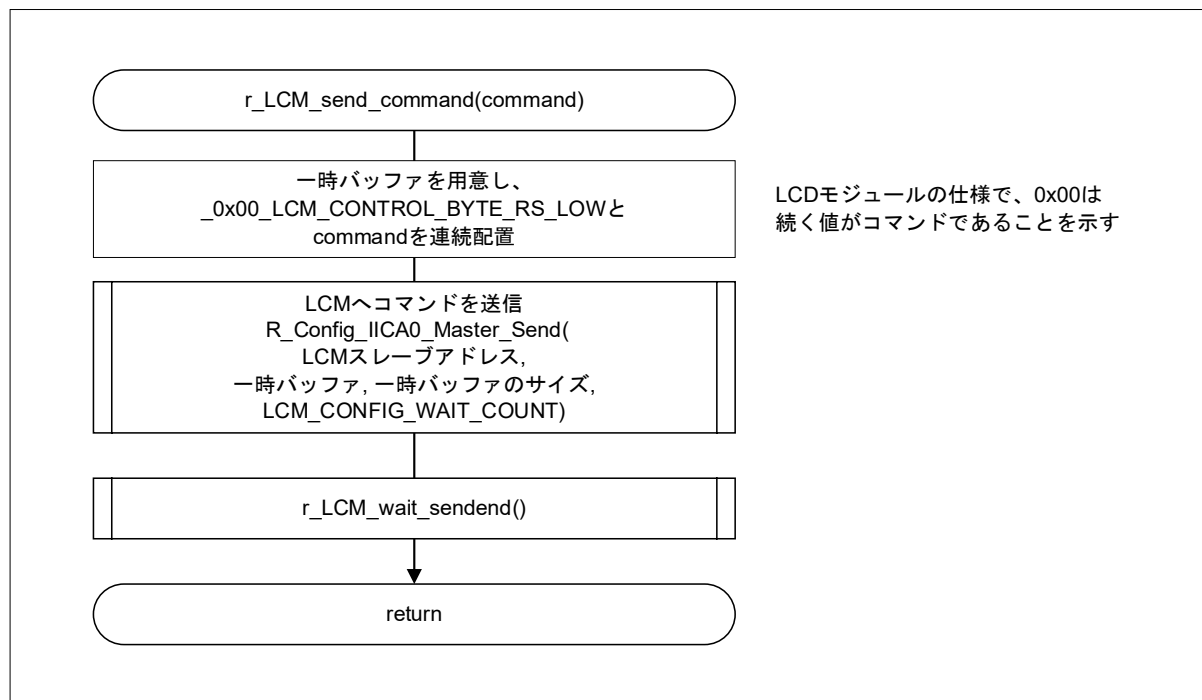
図 4-18 LCD モジュール 文字列送信処理



4.10.17 LCD モジュール コマンド送信処理

図 4-19 に LCD モジュール コマンド送信処理のフローチャートを示します。

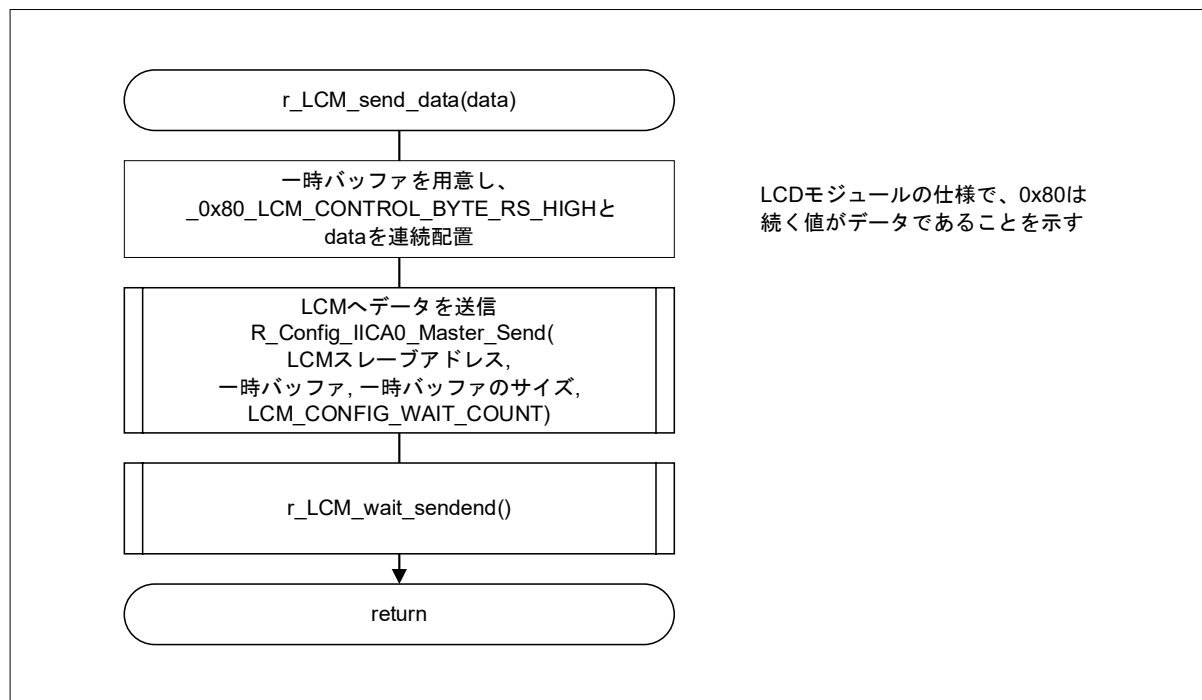
図 4-19 LCD モジュール コマンド送信処理



4.10.18 LCD モジュール データ送信処理

図 4-20 に LCD モジュール データ送信処理のフローチャートを示します。

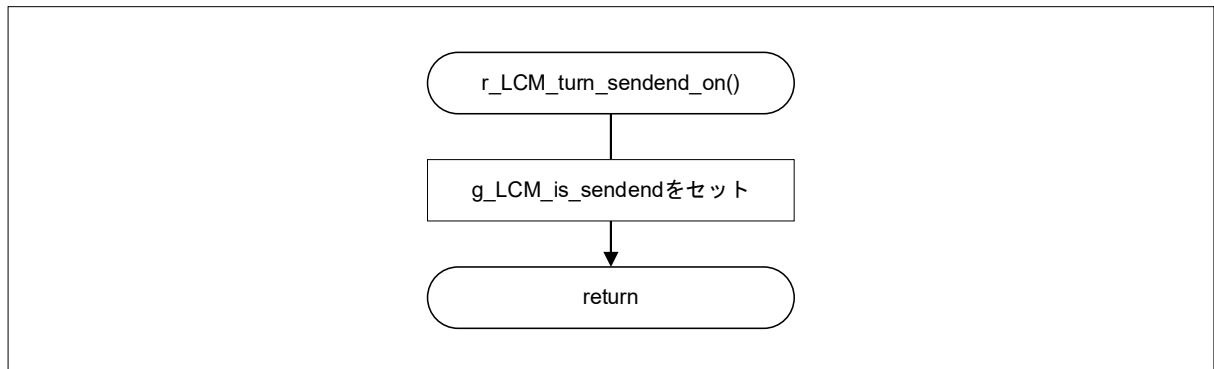
図 4-20 LCD モジュール データ送信処理



4.10.19 LCD モジュール 通信終了フラグ設定

図 4-21 に LCD モジュール 通信終了フラグ設定のフローチャートを示します。

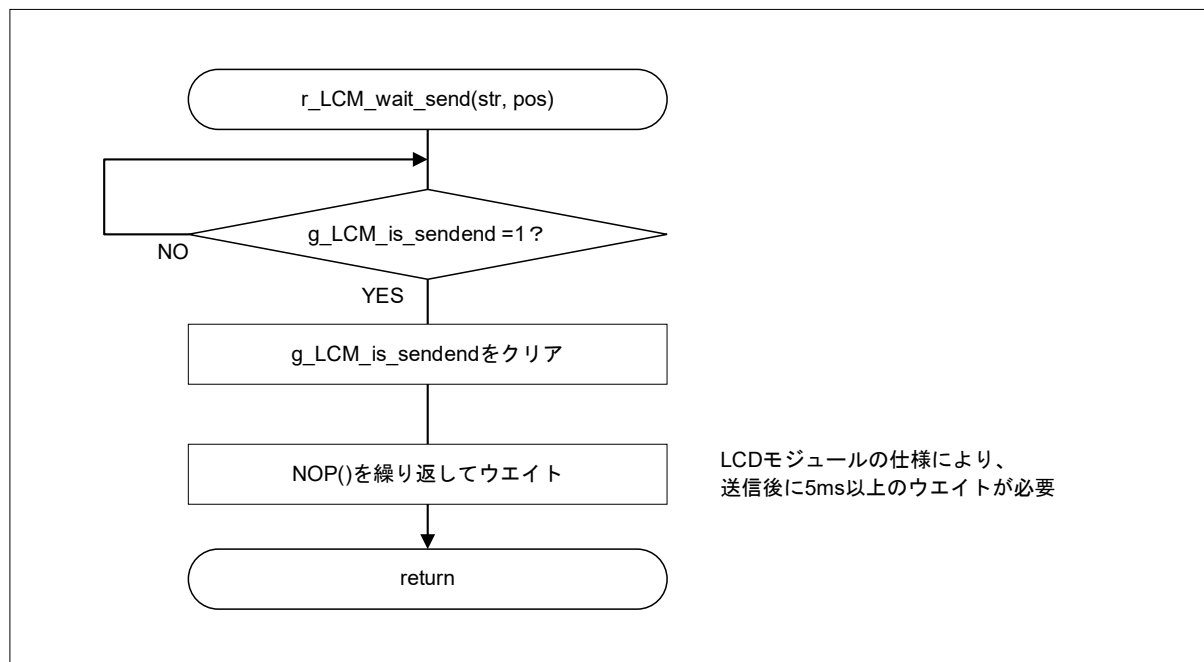
図 4-21 LCD モジュール 通信終了フラグ設定



4.10.20 LCD モジュール 通信終了待ち処理

図 4-22 に LCD モジュール 通信終了待ち処理のフローチャートを示します。

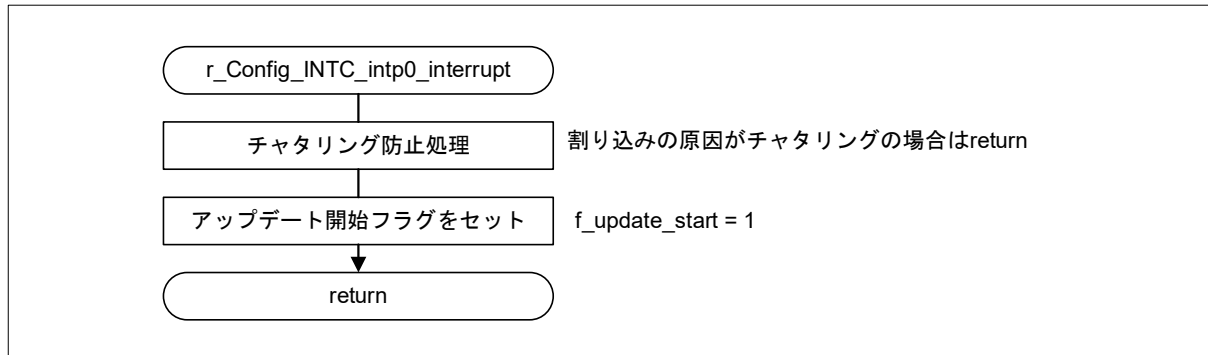
図 4-22 LCD モジュール 通信終了待ち処理



4.10.21 外部割り込み (INTP0) 処理

図 4-23 に外部割り込み (INTP0) 処理のフローチャートを示します。

図 4-23 外部割り込み (INTP0) 処理



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。

6. 参考ドキュメント

RL78/G23 ユーザーズマニュアルハードウェア編 (R01UH0896J)

RL78 ファミリユーザーズマニュアルソフトウェア編 (R01US0015J)

(最新版をルネサスエレクトロニクスホームページから入手してください。)

テクニカルアップデート

(最新の情報をルネサスエレクトロニクスホームページから入手してください。)

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。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2022.3.18	—	初版発行

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、入力信号や入出力ブルアップ電源を入れないでください。入力信号や入出力ブ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、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、これに限られません。）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た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、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

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。

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/