User's Manual

16

RL78 ファミリ

フラッシュ・セルフ・プログラミング・ライブラリ Type01

ユーザーズマニュアル別冊
「RL78/I1C(512KB)専用バンクプログラミング」

16 ビット・シングルチップ・マイクロコントローラ

対象デバイス：

RL78 / I1C(512KB)

ルネサス エレクトロニクス
www.renesas.com

Rev.1.00　2021.02

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。

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。

このマニュアルの使い方

1. 目的と対象者

本ユーザーズマニュアル別冊は、フラッシュ・セルフ・プログラミング・ライブラリ Type01 のフラッシュ関数を使用した RL78/I1C(512KB)専用の機能であるバンクプログラミングの実現方法を理解していただくことを目的としています。また、この機能を使用したアプリケーション・システムを設計するユーザを対象としています。

フラッシュ関数の使用方法については、RL78 ファミリ フラッシュ・セルフ・プログラミング・ライブラリ Type01 のユーザーズマニュアルに記載されています。デバイスの機能については、RL78/I1C(512KB)デバイスのユーザーズマニュアルに記載されていますのであわせてお読みください。

最新版はルネサス エレクトロニクスのホームページに掲載されています。

資料名	資料番号
RL78 ファミリ フラッシュ・セルフ・プログラミング・ライブラリ Type01 ユーザーズマニュアル	注 1
RL78/I1C (512 KB) ユーザーズマニュアル ハードウェア編	R01UH0889JJ

注1. 各リージョン対象のユーザーズマニュアルをご確認ください。

対象デバイス：RL78/I1C(512KB) R5F10NML, R5F10NPL

2. 数や記号の表記

注 ：本文中につけた注の説明
注意 ：気をつけて読んでいただきたい内容
数の表記 ： 2進数 …××××、または××××B
10進数 …××××
16進数 …××××H、または0x××××

3. 略語および略称の説明

略語／略称	フルスペル	備考
FSL	Flash Self-programming Library	フラッシュ・セルフ・プログラミング・ライブラリ
SCIM	Status check internal mode	ステータス・チェック・インターナル・モード
SCUM	Status check user mode	ステータス・チェック・ユーザ・モード

目次

第1章 概 説

1.1 概 要

RL78/I1C(512KB)では、FSL Type01 のフラッシュ関数と、表 1-1に示す RL78/I1C(512KB)専用の機能を組み合わせて使用することで、ユーザプログラムを実行しながら、コード・フラッシュ・メモリ上のプログラムの更新を行った後に、256KB の 2 つのバンクをスワップすることができます。本 RL78 ファミリ FSL Type01 ユーザーズマニュアル別冊では、これらを総称してバンクプログラミングと呼びます。

対象デバイス：RL78/I1C(512KB) R5F10NML, R5F10NPL
表 1-1の機能は、対象デバイス以外では使用できません。

表 1-1 RL78/I1C(512KB)専用の機能

機能名	必要な操作
バンクモード切り替え機能	デバイスのレジスタを操作。
バンクプログラミング機能	FSL Type01 のフラッシュ関数を使用。
バンクスワップ機能	FSL Type01 のフラッシュ関数を使用。

FSL Type01 のフラッシュ関数の使用方法については、FSL Type01 のユーザーズマニュアルに記載されています。また、デバイスの機能については、RL78/I1C(512KB)デバイスのユーザーズマニュアル ハードウェア編に記載されていますのであわせてお読みください。

第2章　バンクプログラミング

バンクプログラミングでは、RL78/I1C(512KB)専用機能のバンクモード切り替え機能、バンクプログラミング機能およびバンクスワップ機能を使用します。この章では、各機能の説明、および FSL Type01 のフラッシュ関数を使用した実行手順について説明します。

2.1　RL78/I1C(512KB)のユースケース

RL78/I1C(512KB)では、バンクプログラミングを使用する場合と、使用しない場合のユースケースがあり、それぞれコード・フラッシュ・メモリの書き換え方法や使用可能な機能が一部異なります。
図 2-1に、RL78/I1C(512KB)のユースケース図を示します。

・バンクプログラミングを使用する場合
コード・フラッシュ・メモリを 256KB の 2 つのバンク領域として使用してください。
バンクモード切り替え機能、バンクプログラミング機能、バンクスワップ機能が使用できます。

・バンクプログラミングを使用しない場合
コード・フラッシュ・メモリを 512KB の 1 つの領域として使用してください。
バンクモード切り替え機能、バンクプログラミング機能およびバンクスワップ機能は使用できません。
この場合、従来のセルフ・プログラミングを使用してコード・フラッシュ・メモリの書き換えが可能ですので、本ユーザーズマニュアル別冊の参照は不要です。

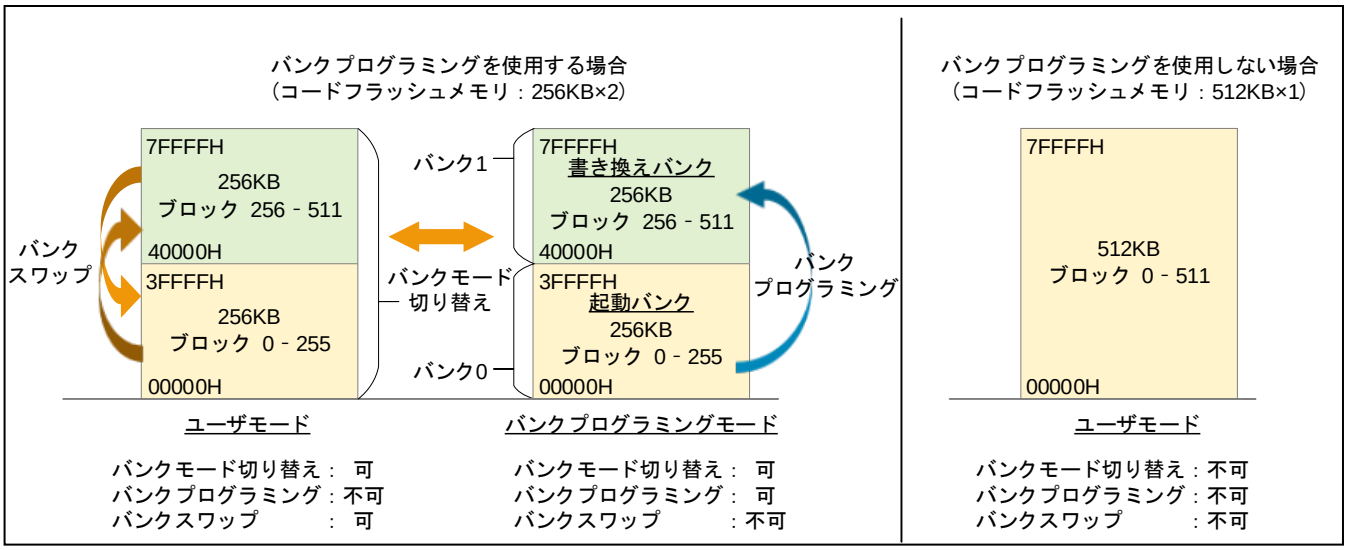

図 2-1 RL78/I1C(512KB)のユースケース図

注意 1. バンクモード切り替え機能、バンクプログラミング機能、バンクスワップ機能は、RL78/I1C(512KB)以外の製品では使用することができません。

2. RL78/I1C(512KB)では、ブート・スワップ機能は使用することができません。

2.2　バンクモード切り替え機能

ユーザモードとバンクプログラミングモードを切り替える機能で、ユーザがデバイスのフラッシュ動作モード選択レジスタ（FLMODE）を操作して切り替える必要があります。バンクモードの切り替え方法については、デバイスのユーザーズマニュアルをご参照ください。リセット解除後は、常にユーザモードで起動します。

図 2-2に、バンクモード切り替え機能の概念図を示します。

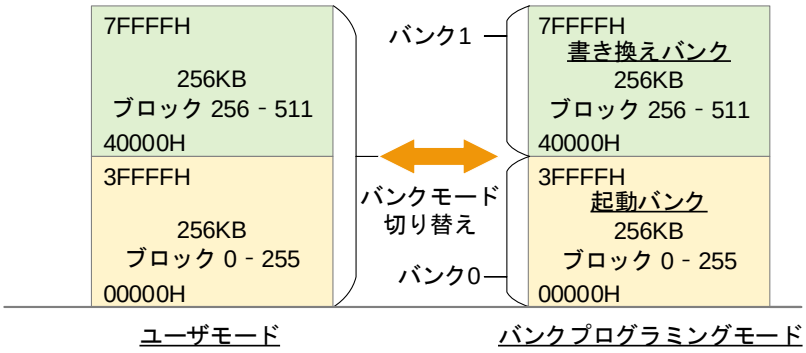

図 2-2 バンクモード切り替え機能の概念図

注意　バンクプログラミングを使用しない場合は、バンクモードをユーザモードからバンクプログラミングモードに切り替えないでください。

2.3　バンクプログラミング機能

FSL Type01 のフラッシュ関数を使用し、コード・フラッシュ・メモリの起動バンク（ROM）上でユーザプログラムを実行しながら、書き換えバンクを書き換える機能です。バンクプログラミング機能は、バンクモード切り替え機能を使用し、バンクプログラミングモードに切り替えて実行する必要があります。図 2-3に、バンクプログラミング機能の概念図を示します。

起動バンクと書き換えバンクの固定領域には同一のプログラム（通信処理や書き換え処理、およびバンクプログラミング中に実行する処理を含む）を配置します。バンクプログラミングでは、起動バンクの固定領域内の書き換え処理を実行して、書き換えバンクのプログラム領域を新プログラムへ書き換えます。

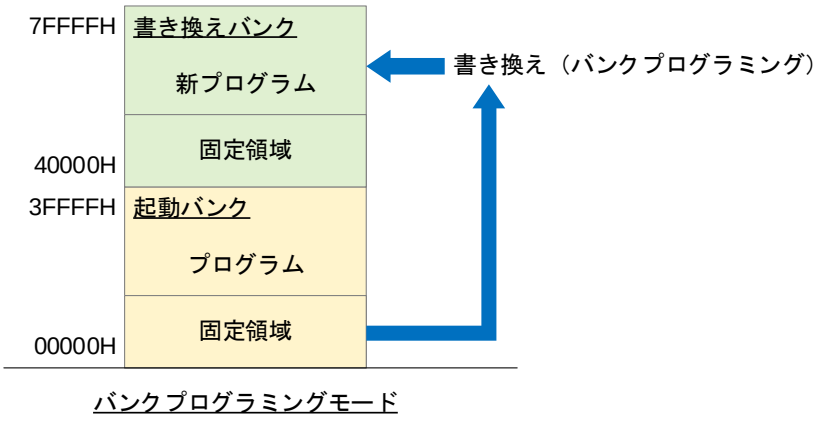

図 2-3 バンクプログラミング機能の概念図

注意 1.　バンクプログラミング機能は、ユーザモードで使用することができません。
2.　バンクプログラミング機能使用後は、ユーザモードに切り替えてください。

2.4　バンクスワップ機能

FSL Type01 のフラッシュ関数を使用してコード・フラッシュ・メモリの「バンク 0」と「バンク 1」のアドレスを切り替える機能です。バンクスワップ機能は、ユーザプログラムでバンクプログラミングを使用する場合でのみユーザモードで使用することができます。図 2-4に、バンクスワップ機能の概念図を示します。

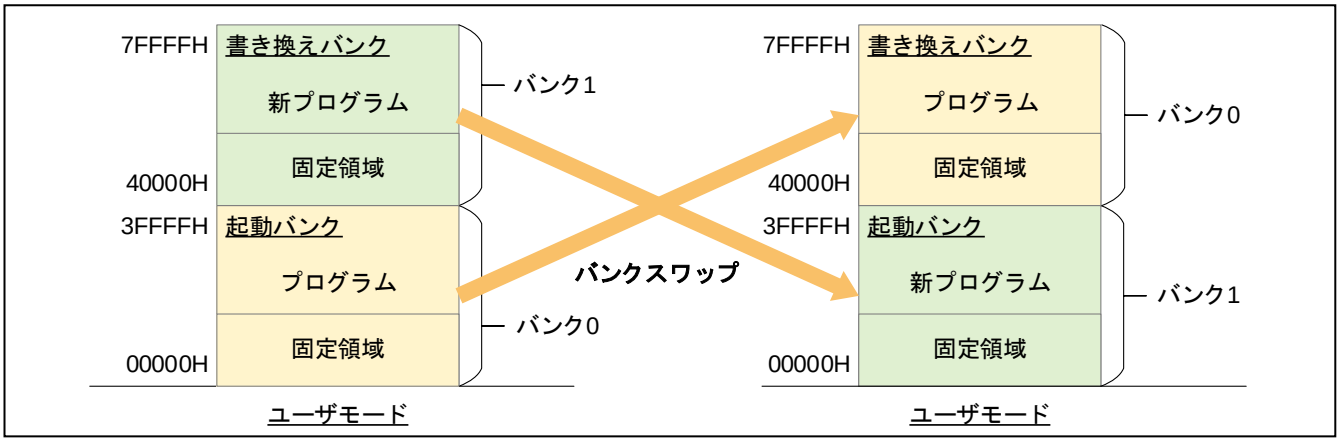

図 2-4 バンクスワップ機能の概念図

注意　バンクプログラミングを使用しない場合は、バンクスワップ機能を使用しないでください。

表 2-1に、バンクスワップ機能を実現する FSL Type01 のフラッシュ関数一覧を示します。
それぞれバンクスワップが実行されるタイミングや内部処理が異なります。

表 2-1 バンクスワップ機能を実現するFSL Type01のフラッシュ関数一覧

関数名	動作説明
FSL_InvertBootFlag	ブート・フラグの現在値を反転します。リセット後に「バンク0」と「バンク1」のアドレスが切り替わり、ブート・フラグの設定に沿ったバンクから起動します。
FSL_SwapBootCluster	バンクスワップを実行し、バンクスワップ後の領域のリセット・ベクタに登録されているアドレスへ移動します。（ブート・フラグの反転は行いません。）
FSL_SwapActiveBootCluster	ブート・フラグの現在値を反転し、バンクスワップを実行します。

2.5　バンクプログラミング実行手順

(1)　バンクプログラミング実行手順

書き換えバンクへのバンクプログラミングを行う際に、ステータス・チェックのモードを選択することが可能です。図 2-5に、ステータス・チェック・インターナル・モードのフロー例、図 2-6に、ステータス・チェック・ユーザ・モードのフロー例を示します。ステータス・チェック・モードの詳細については、FSL Type01 のユーザーズマニュアルをご参照ください。

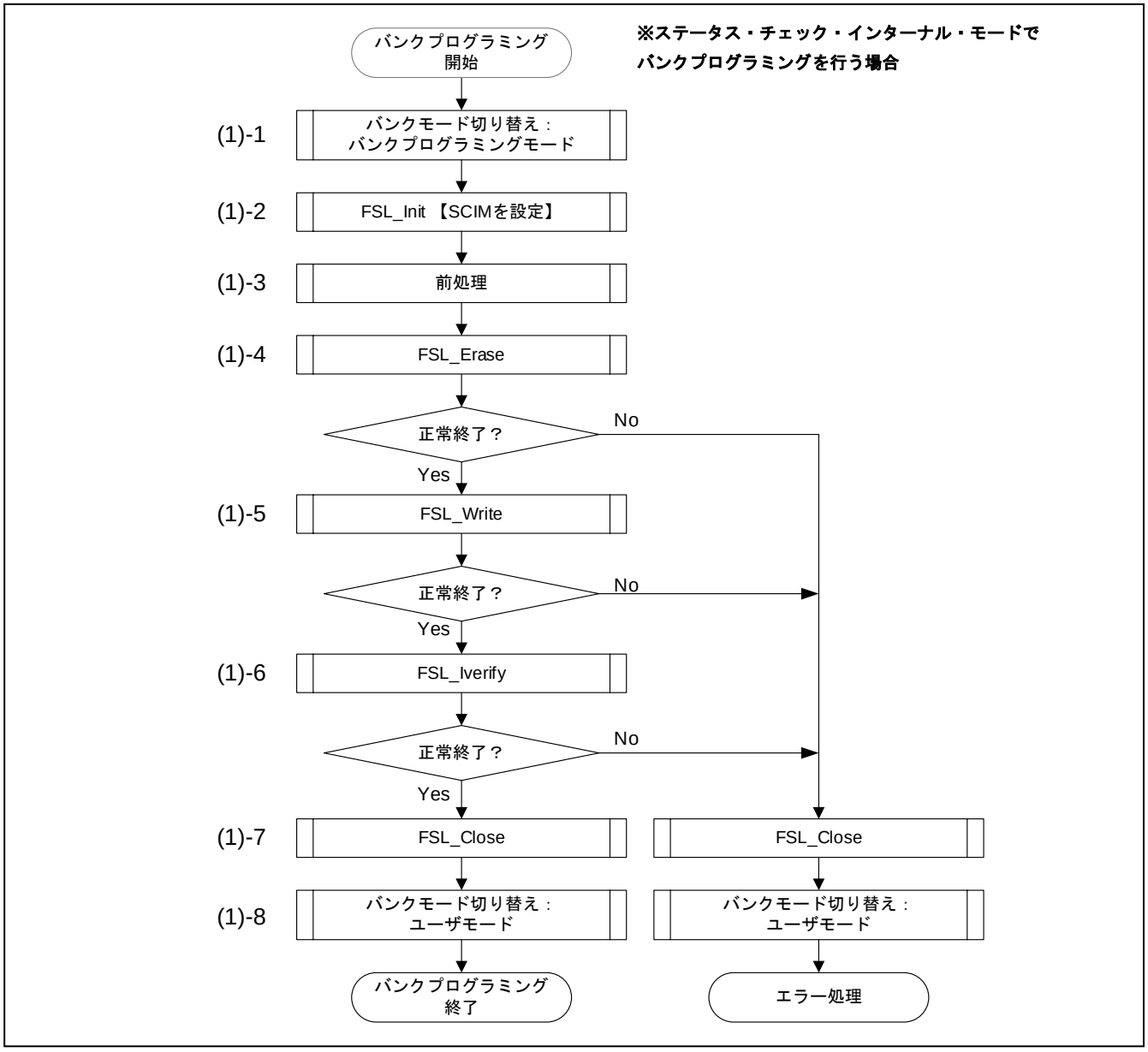

図 2-5 ステータス・チェック・インターナル・モードのフロー例

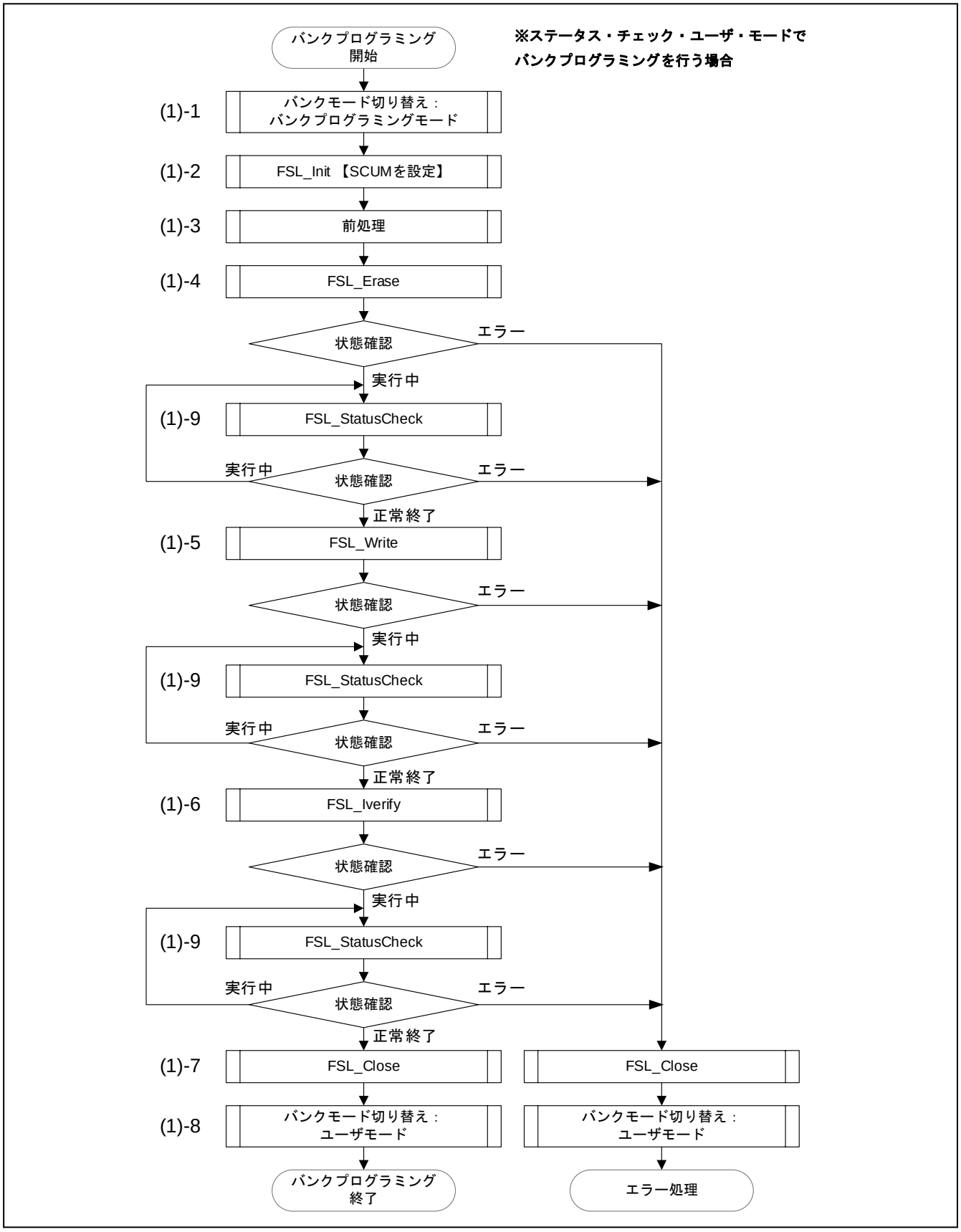

図 2-6 ステータス・チェック・ユーザ・モードのフロー例

図 2-5 ステータス・チェック・インターナル・モードのフロー例と、図 2-6 ステータス・チェック・ユーザ・モードのフロー例について説明します。

(1)-1　バンクモードの切り替え
バンクプログラミングモードに設定します。

(1)-2　フラッシュ・セルフ・プログラミングの初期化
FSL_Init 関数の実行
・ステータス・チェック・モードの設定をします。
図 2-5では、ステータス・チェック・インターナル・モードを設定。
図 2-6では、ステータス・チェック・ユーザ・モードを設定。

(1)-3　前処理
- フラッシュ環境の開始(FSL_Open 関数の実行)
- フラッシュ関数の準備処理(FSL_PrepareFunctions 関数の実行)

(1)-4　書き換えバンクの新プログラム用の領域を消去
FSL_Erase 関数を実行し、書き換えバンクの新プログラム用の領域のブロックを全て消去します。

(1)-5　書き換えバンクの新プログラム用の領域へ新しいプログラムを書き込み
FSL_Write 関数を実行し、新しいプログラム（通信プログラムなどで受信したプログラム）を書き換えバンクの新プログラム用の領域に書き込みます。

(1)-6　書き換えバンクの新プログラム用の領域のベリファイ
FSL_IVerify 関数を実行し、新しいプログラムを書き込んだ領域をベリファイします。

(1)-7　終了処理
FSL_Close 関数を実行し、フラッシュ・セルフ・プログラミングの終了処理をします。

(1)-8　バンクモードの切り替え
ユーザモードに設定します。

(1)-9　ステータス・チェック
ステータス・チェック・ユーザ・モードを使用する場合は、コード・フラッシュ・メモリの制御が終了するまで、ステータス・チェックを行う必要があります。
図 2-5では、関数から戻った時点でコード・フラッシュ・メモリの制御が終了し、各関数の処理も終了しているため、ステータス・チェックは不要です。

2.6 バンクスワップ実行手順

バンクスワップ機能を実現するフラッシュ関数は、FSL_InvertBootFlag 関数、FSL_SwapBootCluster 関数、および FSL_SwapActiveBootCluster 関数が用意されています。

FSL_InvertBootFlag 関数は実行後、リセット発生によりバンクスワップが実行されます。FSL_SwapBootCluster 関数、および FSL_SwapActiveBootCluster 関数は、関数実行中にバンクスワップが実行されます。また、これらのバンクスワップ機能を実現するフラッシュ関数は、バンクモードをユーザモード、かつ、ステータス・チェック・インターナル・モードで実行する必要があります。

以降に、それぞれの関数のバンクスワップ実行手順と詳細動作を説明します。

(1) FSL_InvertBootFlag 関数を使用したバンクスワップ実行手順

図 2-7に、FSL_InvertBootFlag関数を実行後、リセット発生によりバンクスワップが実行される例を示します。

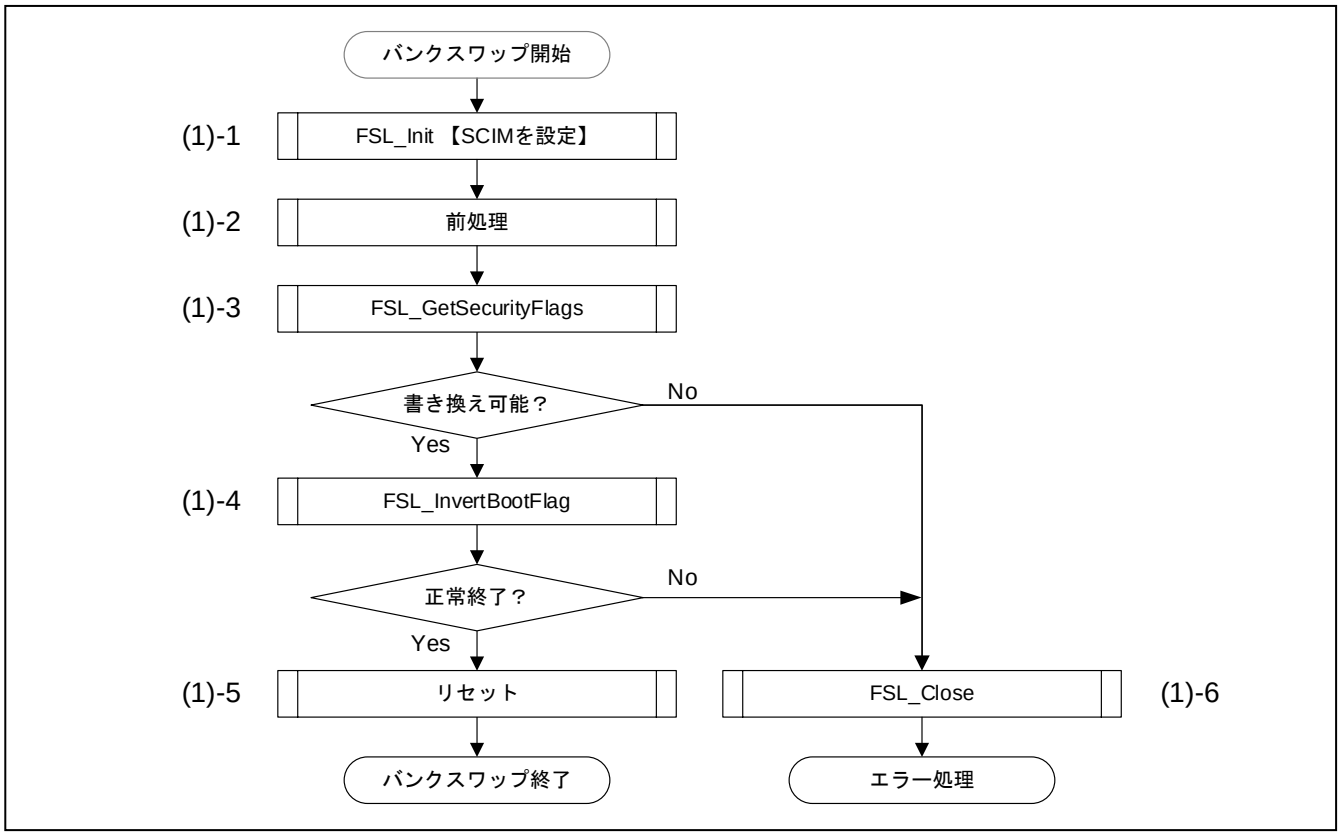

図 2-7 FSL_InvertBootFlag 関数を使用したバンクスワップのフロー例

図 2-7 FSL_InvertBootFlag 関数を使用したバンクスワップのフロー例について説明します。

(1)-1　フラッシュ・セルフ・プログラミングの初期化
FSL_Init 関数の実行
・ステータス・チェック・インターナル・モードに設定します。

(1)-2　前処理
- フラッシュ環境の開始(FSL_Open 関数の実行)
- フラッシュ関数の準備処理(FSL_PrepareFunctions 関数の実行)
- フラッシュ関数(拡張関数)の準備処理(FSL_PrepareExtFunctions 関数の実行)

(1)-3　セキュリティ・ビットの確認（推奨）
FSL_GetSecurityFlags 関数の実行により、セキュリティ・フラグ情報を取得し、セキュリティ・フラグ情報のビット 1(ブート・クラスタ 0 の書き換え禁止フラグ)が 1（許可）になっていることを確認します。
備考 : ビット 1(ブート・クラスタ 0 の書き換え禁止フラグ)が 0（禁止）になっている場合、(1)-4 の FSL_InvertBootFlag 関数の実行時にプロテクト・エラーが返ります。

(1)-4　ブート・フラグの設定
FSL_InvertBootFlag 関数を実行することにより、ブート・フラグの切り替えを行います。

(1)-5　イベントの発生
リセットを発生させることにより、起動バンクと書き換えバンクが入れ替わります。リセット後は、リセット前と異なるバンクから起動します。

(1)-6　終了処理
FSL_Close 関数を実行し、フラッシュ・セルフ・プログラミングの終了処理をします。

(2)　FSL_SwapBootCluster 関数を使用したバンクスワップ実行手順

FSL_SwapBootCluster関数は、関数の実行中にバンクスワップを実行し、バンクスワップ後の領域のリセット・ベクタに登録されているアドレスへ移動します（ブート・フラグの反転は行われません）。
図 2-8に、FSL_SwapBootCluster関数を使用したバンクスワップのフロー例を示します。

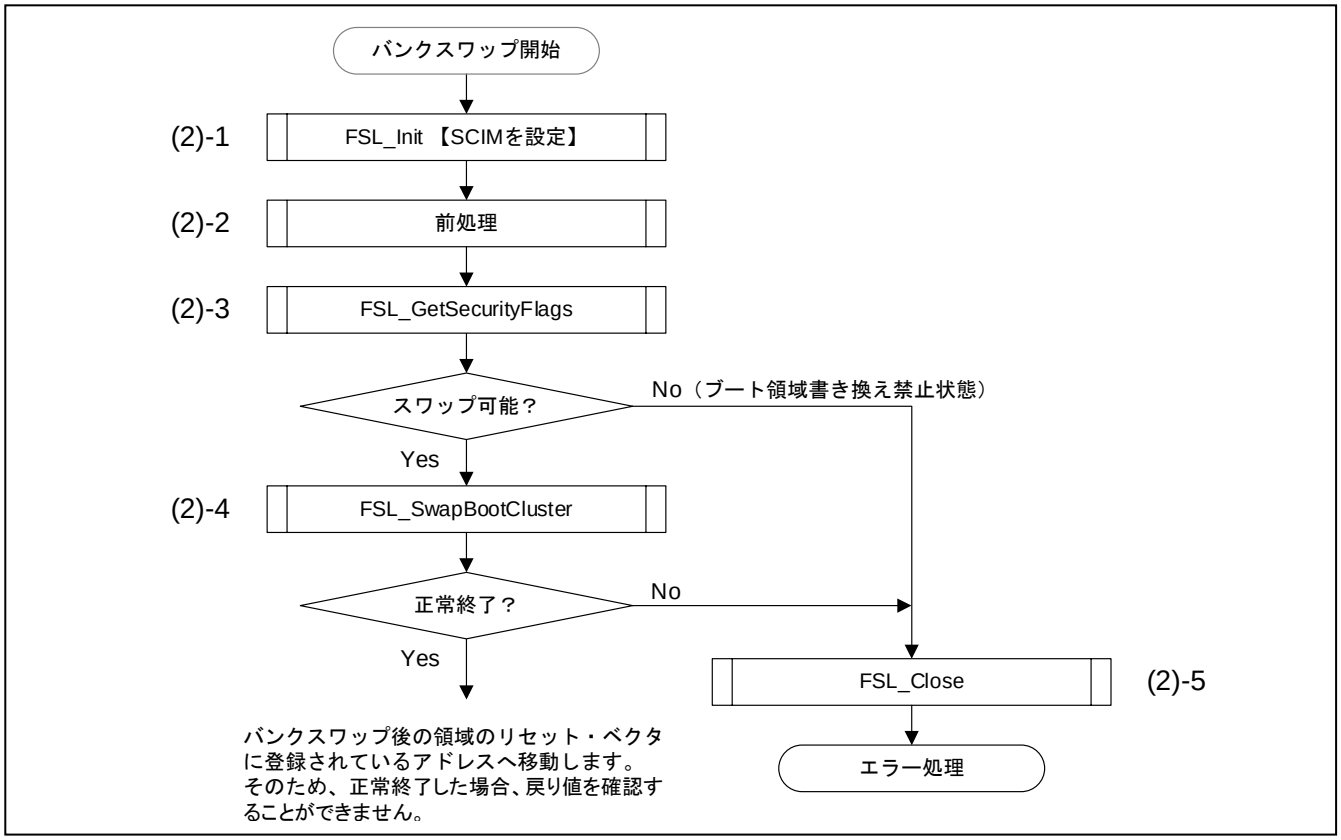

図 2-8　FSL_SwapBootCluster関数を使用したバンクスワップのフロー例

図 2-8　FSL_SwapBootCluster関数を使用したバンクスワップのフロー例について説明します。

(2)-1　フラッシュ・セルフ・プログラミングの初期化

FSL_Init 関数の実行

・ステータス・チェック・インターナル・モードに設定します。

(2)-2　前処理

- フラッシュ環境の開始(FSL_Open 関数の実行)
- フラッシュ関数の準備処理(FSL_PrepareFunctions 関数の実行)
- フラッシュ関数(拡張関数)の準備処理(FSL_PrepareExtFunctions 関数の実行)

(2)-3　セキュリティ・ビットの確認（推奨）

FSL_GetSecurityFlags 関数の実行により、セキュリティ・フラグ情報を取得し、セキュリティ・フラグ情報のビット 1(ブート・クラスタ 0 の書き換え禁止フラグ)が 1（許可）になっていることを確認します。

備考 : ビット 1(ブート・クラスタ 0 の書き換え禁止フラグ)が 0（禁止）になっている場合、(2)-4 の FSL_SwapBootCluster 関数の実行時にプロテクト・エラーが返ります。

(2)-4　バンクスワップの実行

FSL_SwapBootCluster 関数を実行することにより、起動バンクと書き換えバンクの入れ替えを行い、バンクスワップ後の領域のリセット・ベクタに登録されているアドレスへ移動します（ブート・フラグは反転されません）。

(2)-5　終了処理

FSL_Close 関数を実行し、フラッシュ・セルフ・プログラミングの終了処理をします。

注意　1. この関数が正常に実行された場合は、バンクスワップ後の領域のリセット・ベクタに登録されているアドレスに移動するため、この関数以降の処理は実行されません。

2. この関数ではブート・フラグの反転は行いません。リセットを実行した場合、起動バンクはブート・フラグの設定に沿った状態となります。

3. FSL_ChangeInterruptTable 関数で割り込み先を変更した状態で実行すると、リセット・ベクタに登録されているアドレスへ移動した後も、割り込みは FSL_ChangeInterruptTable 関数で変更された領域へ入る状態に維持されます。割り込み先を復元した状態でリセット・ベクタに登録されているアドレスに移動したい場合、FSL_SwapBootCluster 関数を実行する前に必ず FSL_RestoreInterruptTable 関数を実行し、割り込み先を復元してください。

4. 本関数は ROM 上から実行する事は出来ません。本関数を使用する場合は、FSL_RCD セクションを RAM 上に配置して実行してください。

5. スワップ実行後、リセットが実行された場合にもスワップ後の状態を維持したい場合は、(2)-4 FSL_SwapBootCluster 関数を実行する前に FSL_InvertBootFlag 関数を実行してください。

(3)　FSL_SwapActiveBootCluster関数を使用したバンクスワップ実行手順

FSL_SwapActiveBootCluster関数は、関数の実行中にブート・フラグの現在値を反転し、バンクスワップを実行します。
図 2-9に、FSL_SwapActiveBootCluster関数を使用したバンクスワップのフロー例を示します。

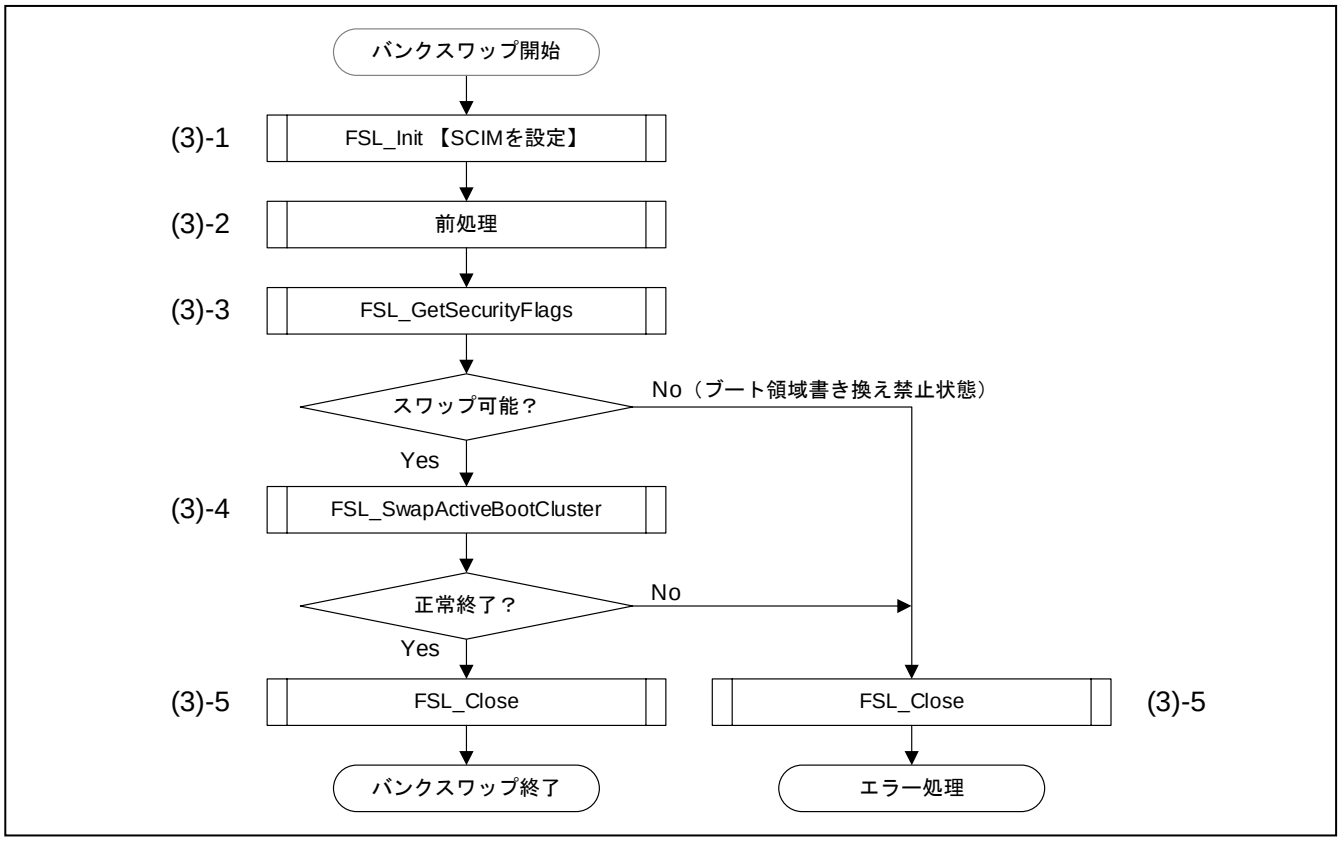

図 2-9　FSL_SwapActiveBootCluster関数を使用したバンクスワップのフロー例

図 2-9　FSL_SwapActiveBootCluster関数を使用したバンクスワップのフロー例について説明します。

(3)-1　フラッシュ・セルフ・プログラミングの初期化

FSL_Init 関数の実行

・ステータス・チェック・インターナル・モードに設定します。

(3)-2　前処理

- フラッシュ環境の開始(FSL_Open 関数の実行)
- フラッシュ関数の準備処理(FSL_PrepareFunctions 関数の実行)
- フラッシュ関数(拡張関数)の準備処理(FSL_PrepareExtFunctions 関数の実行)

(3)-3　セキュリティ・ビットの確認（推奨）

FSL_GetSecurityFlags 関数の実行により、セキュリティ・フラグ情報を取得し、セキュリティ・フラグ情報のビット 1(ブート・クラスタ 0 の書き換え禁止フラグ)が 1（許可）になっていることを確認します。

備考：ビット 1(ブート・クラスタ 0 の書き換え禁止フラグ)が 0（禁止）になっている場合、(3)-4 の FSL_SwapActiveBootCluster 関数の実行時にプロテクト・エラーが返ります。

(3)-4　バンクスワップの実行

FSL_SwapActiveBootCluster 関数を実行することにより、ブート・フラグの現在値を反転し、起動バンクと書き換えバンクの入れ替えを行います。

(3)-5　終了処理

FSL_Close 関数を実行し、フラッシュ・セルフ・プログラミングの終了処理をします。

注意　1. 本関数は ROM 上から実行する事は出来ません。本関数を使用する場合は、FSL_RCD セクションを RAM 上に配置して実行してください。

2. 本関数実行後は ROM 上の割り込みベクタも変更されます。実行前から実行後を通して ROM 上の割り込み処理を使用する場合は、ROM 上の割り込みベクタが動作中に切り替わる事に配慮して使用してください。

3. リセット等を行わない状態でバンクの入れ替えを行い、実行したアドレスに戻るため、FSL_SwapActiveBootCluster 関数を呼び出すユーザ処理を RAM 上に配置する必要があります。

2.7　バンクプログラミングを使用する場合の実行可能関数

バンクプログラミングを使用する場合、バンクモード設定および、ステータス・チェックのモード設定によって実行可能なフラッシュ関数が異なります。表 2-2 にバンクプログラミングを使用する場合の実行可能関数一覧を示します。バンクプログラミングモードで書き換えバンクの新プログラム用の領域を書き換える場合は、フラッシュ関数を ROM に配置して実行することを前提としています。それに伴い拡張関数も ROM に配置されるため、ユーザモードで拡張関数を実行する場合は、ユーザ処理を ROM に配置できるステータス・チェック・インターナル・モード(SCIM)のみが設定可能です。各フラッシュ関数の詳細については、FSL Type01 のユーザーズマニュアルをご参照ください。

表 2-2 バンクプログラミングを使用する場合の実行可能関数一覧

◎：バンクプログラミング　○：使用可能　－：使用しない

フラッシュ関数名	実行モード		
	ユーザモード	バンクプログラミングモード	
	SCIM 注3	SCIM	SCUM
FSL_Init	○	○	○
FSL_Open	○	○	○
FSL_Close	○	○	○
FSL_PrepareFunctions	○	○	○
FSL_PrepareExtFunctions	○	○	○
FSL_ChangeInterruptTable	○	－	－
FSL_RestoreInterruptTable	○	－	－
FSL_BlankCheck	－注2	◎	◎
FSL_Erase	－注2	◎	◎
FSL_IVerify	－注2	◎	◎
FSL_Write	－注2	◎	◎
FSL_GetSecurityFlags	○	○	○
FSL_GetBootFlag	○	○	○
FSL_GetSwapState	○	○	○
FSL_GetBlockEndAddr	○	○	○
FSL_GetFlashShieldWindow	○	○	○
FSL_SwapBootCluster	○注1	－	－
FSL_SwapActiveBootCluster	○注1	－	－
FSL_InvertBootFlag	○	－	－
FSL_SetBlockEraseProtectFlag	○	－	－
FSL_SetWriteProtectFlag	○	－	－
FSL_SetBootClusterProtectFlag	○	－	－
FSL_SetFlashShieldWindow	○	－	－
FSL_StatusCheck	－	－	○
FSL_StandBy	－	－	○
FSL_WakeUp	－	－	○
FSL_ForceReset	○	○	○
FSL_GetVersionString	○	○	○

注 1. FSL_RCD（セグメント／セクション）をRAM上に配置する必要があります。
注 2. 書き換えバンクは、バンクプログラミングモードで書き換えます。
注 3. 処理中に割り込みを使用する場合は、割り込み先を RAM に変更する必要があります。

2.8 バンクプログラミングに関する注意事項

- ブート・クラスタ 0 書き換え禁止フラグが 0（禁止）になっている場合、バンクスワップは実行できません。
- バンクスワップを使用する際には、アドレス「400C0H-400C3H」には「000C0H-000C3H」と同じ値を設定してください。
- バンクプログラミングは、VDD≧2.7V で実施してください。また、LP または LV モードでのバンクプログラミングは禁止です。

他にも注意事項がありますので、あわせてデバイスのユーザーズマニュアルをご参照ください。

RL78ファミリ フラッシュ・セルフ・プログラミング・ライブラリ Type01
ユーザーズマニュアル別冊
「RL78/I1C(512KB)専用バンクプログラミング」

発行年月日　2021年 2月 2日　　Rev.1.00
発行　　ルネサス エレクトロニクス株式会社
〒135-0061　東京都江東区豊洲3-2-24（豊洲フォレシア）

RL78 ファミリ

ルネサス エレクトロニクス株式会社

R20UT4871JJ0100