RX100シリーズ

R01AN1938JJ0110
Rev.1.10
2020.08.20

Renesas Starter Kit+ for RX63N を使用した RX100 シリーズ用フラッシュプログラマ(SCI インタフェース)

要旨

本アプリケーションノートでは、Renesas Starter Kit+ for RX63N(以下 RSK+RX63N と略します)を使用したRX100シリーズ用のフラッシュプログラマを実現する方法について説明します。

書き換え対象は RX100 シリーズです。RX100 シリーズのユーザ領域書き換えには、ブートモード(SCI インタフェース)を使用します。

対象デバイス

・RX100 シリーズ

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. 仕様

フラッシュプログラマは、RSK+RX63N で動作し、ターゲットの RX100 シリーズマイコンをブートモード(SCI インタフェース)で起動します。その後、調歩同期式シリアルで通信することで RX100 シリーズのユーザ領域を書き換えます。

表 1.1に使用する周辺機能と用途を、図 1.1にフラッシュプログラマの使用例を示します。

調歩同期式のシリアル通信には、シリアルコミュニケーションインタフェースのチャネル 0(SCI0)を使用します。

通信データフォーマットおよび出力形式は以下のとおりです。

スタートビット：1 ビット
転送データ　　：8 ビット
パリティビット：なし
ストップビット：1 ビット
ビットレート　：19,200bps(動作周波数選択コマンドの応答まで)
　　：1 Mbps(プログラム/イレーズステータス遷移コマンド以降)
出力形式　　：CMOS 出力

表1.1　使用する周辺機能と用途

周辺機能	用途
SCI0	調歩同期式シリアル送受信
CMT0	待ち時間用タイマ
I/O ポート	ブートモード制御、LCD 出力

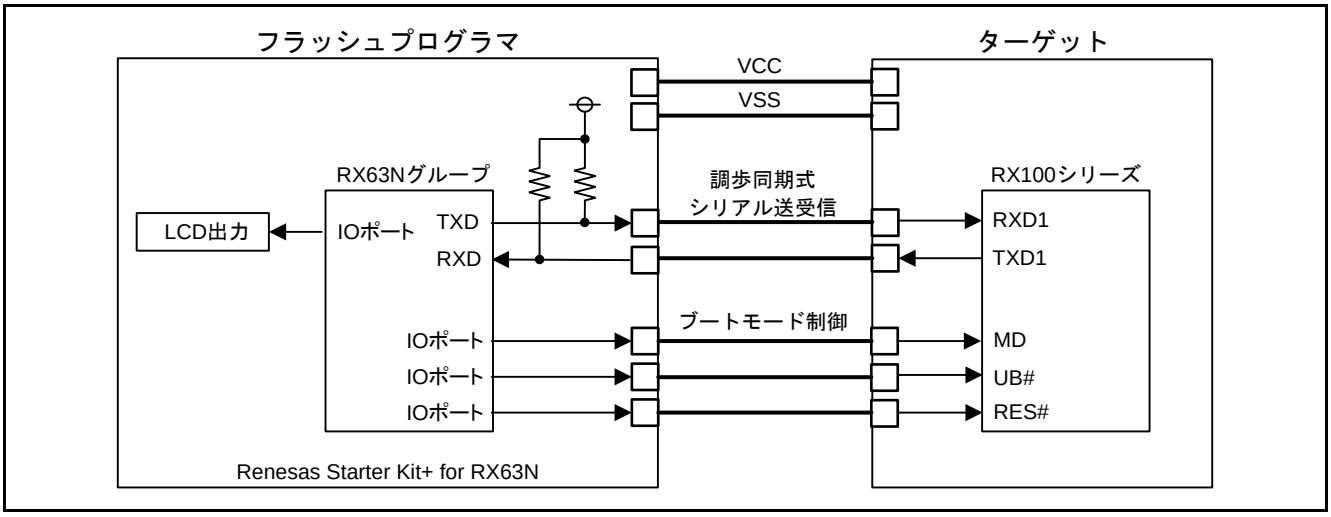


図1.1　フラッシュプログラマの使用例

1.1　RSK+RX63N のユーザ領域メモリ配置

RSK+RX63N のユーザ領域には、フラッシュプログラマのプログラムとターゲットマイコンのユーザ領域に書き込むデータを配置します。図 1.2にRSK+RX63N のユーザ領域メモリ配置を示します。

RSK+RX63N のユーザ領域への書き込みについては「5.1 RSK+RX63N への書き込み方法」を参照してください。

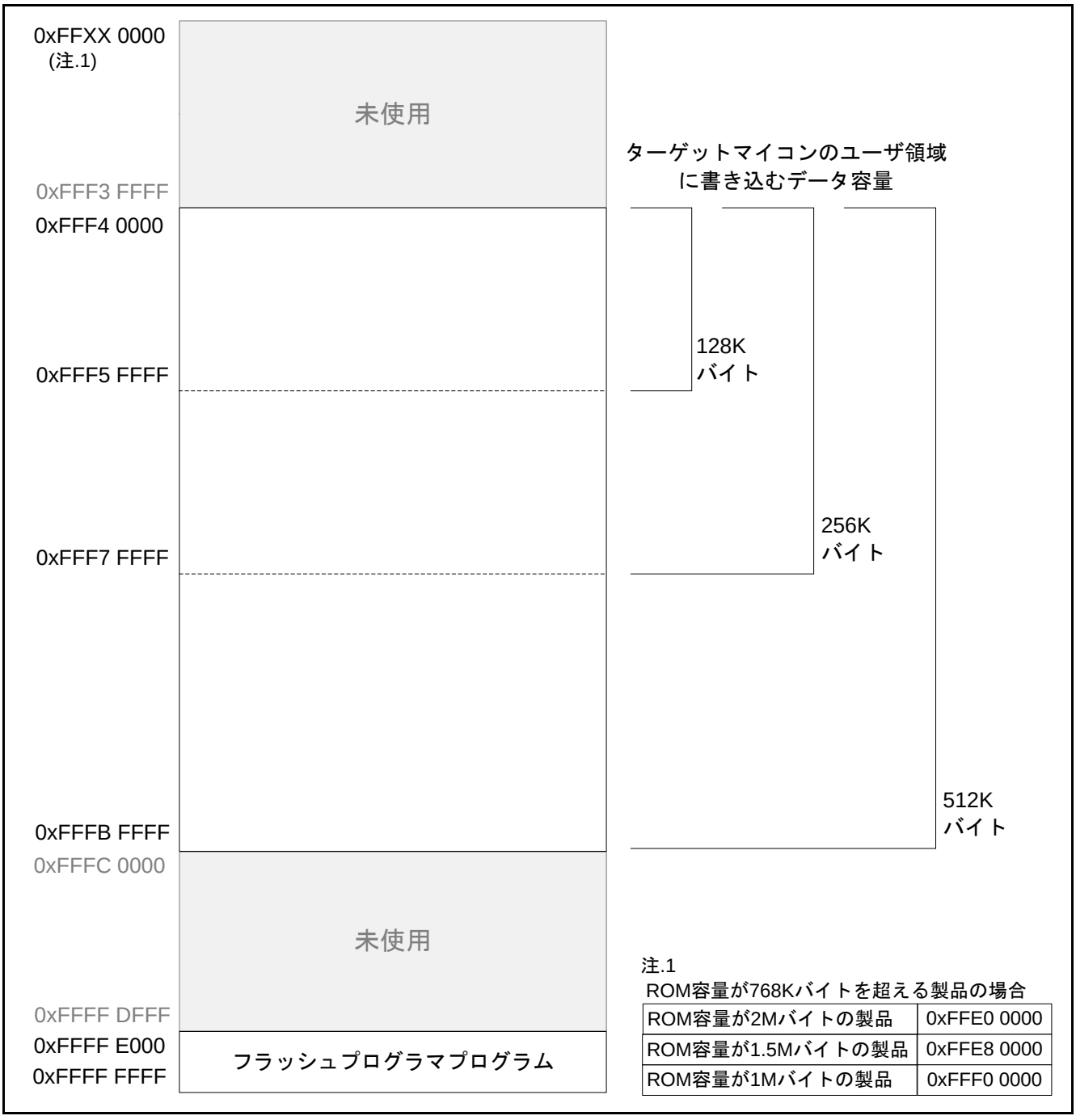


図1.2　RSK+RX63N のユーザ領域メモリ配置

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

項目	内容
使用マイコン	R5F563NBDDFC (RX63N グループ)
動作周波数	メインクロック: 12MHz PLL: 192MHz (メインクロック 1 分周 16 逓倍) システムクロック(ICLK): 96MHz (PLL 2 分周) 周辺モジュールクロック B (PCLKB): 48MHz (PLL 4 分周)
動作電圧	3.3V
統合開発環境	ルネサスエレクトロニクス製 e² studio 2020-04
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package for RX Family V.3.02.00
	コンパイルオプション (統合開発環境のデフォルト設定を使用しています)
iodefine.h のバージョン	Version 1.6A
エンディアン	リトルエンディアン
動作モード	シングルチップモード
プロセッサモード	スーパバイザモード
サンプルコードのバージョン	Version 1.10
使用ボード	Renesas Starter Kit+ for RX63N (製品型名: R0K50563NC000BE)

注1.　元のプロジェクトで指定するツールチェーン(C コンパイラ）と同一のバージョンがインポートする先にない場合は、ツールチェーンが選択されない状態になり、エラーが発生します。プロジェクトの設定画面でツールチェーンの選択状態を確認してください。

選択方法は、FAQ 3000404 を参照してください。

FAQ 3000404 :インポートしたプロジェクトをビルドすると「PATH でプログラム "make" が見つかりません」エラーになる(e² studio)

3. 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。併せて参照してください。

RX63N グループ、RX631 グループ　初期設定例 Rev.1.10(R01AN1245JJ0110_RX63N)
RX63N High-performance Embedded Workshop 用ルネサススタータキットのサンプルコード
(R01AN1395JG0100_RX63N)

上記アプリケーションノートの初期設定関数、デバック LCD 出力関数を、本アプリケーションノートのサンプルコードで使用しています。Rev は本アプリケーションノート作成時点のものです。

最新版がある場合、最新版に差し替えて使用してください。最新版はルネサスエレクトロニクスホームページで確認および入手してください。

4.　ハードウェア説明

4.1　ハードウェア構成例

図 4.1に接続例を示します。

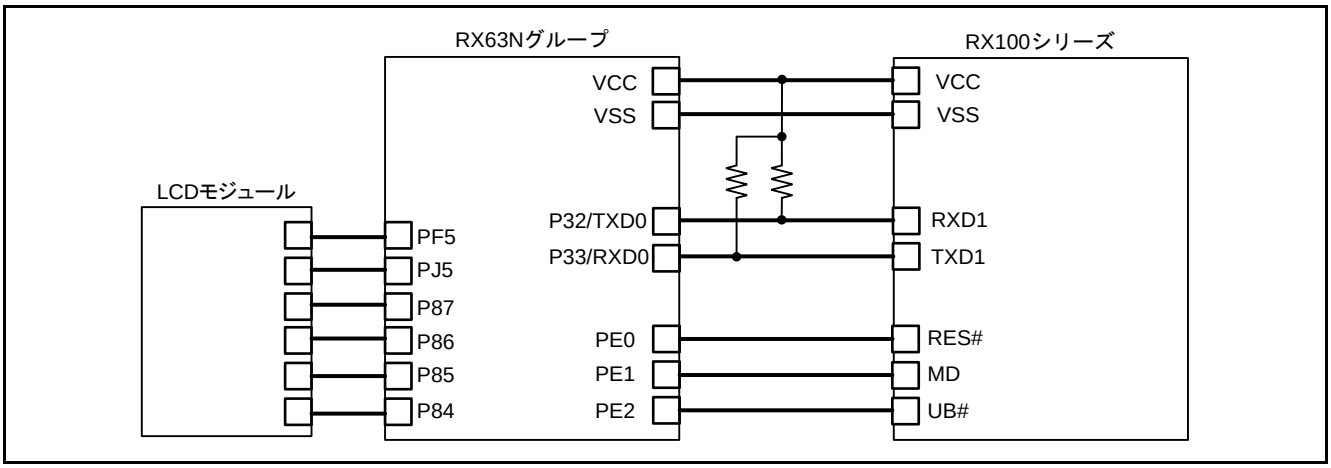


図4.1　接続例

4.2　使用端子一覧

表 4.1に使用端子と機能を示します。

表4.1　使用端子と機能

端子名	入出力	内容
P87	出力	Debug LCD データ 7 出力
P86	出力	Debug LCD データ 6 / バックライト出力
P85	出力	Debug LCD データ 5 / Y ドライブ出力
P84	出力	Debug LCD データ 4 / X ドライブ出力
PF5	出力	Debug LCD Enable 出力
PJ5	出力	Debug LCD Register select 出力
P33/RXD0	入力	SCI0 の受信データ入力端子
P32/TXD0	出力	SCI0 の送信データ出力端子
PE0	出力	RES#端子の制御
PE1	出力	MD 端子の制御
PE2	出力	UB#端子の制御

5.　ソフトウェア説明

5.1　RSK+RX63N への書き込み方法

RSK+RX63N のユーザ領域に書き込むデータは、以下の 2 つです。
・ターゲット MCU のユーザ領域に書き込むユーザプログラム
・フラッシュプログラマのプログラム

このアプリケーションノートでは、ルネサスフラッシュ開発ツールキット(以下 FDT と略します)を使用した場合を例に説明します。

FDT の S レコード形式、または 16 進数ファイルのエディタ機能を使用して RSK+RX63N のユーザ領域に書き込むデータを結合します。また、FDT を使用して RSK+RX63N のユーザ領域に結合したデータを書き込みます。
FDT の使い方は、ルネサスフラッシュ開発ツールキットのユーザーズマニュアル (R20UT0508JJ)を参照してください。

図 5.1にRSK+RX63N への書き込み方法の流れを示します。

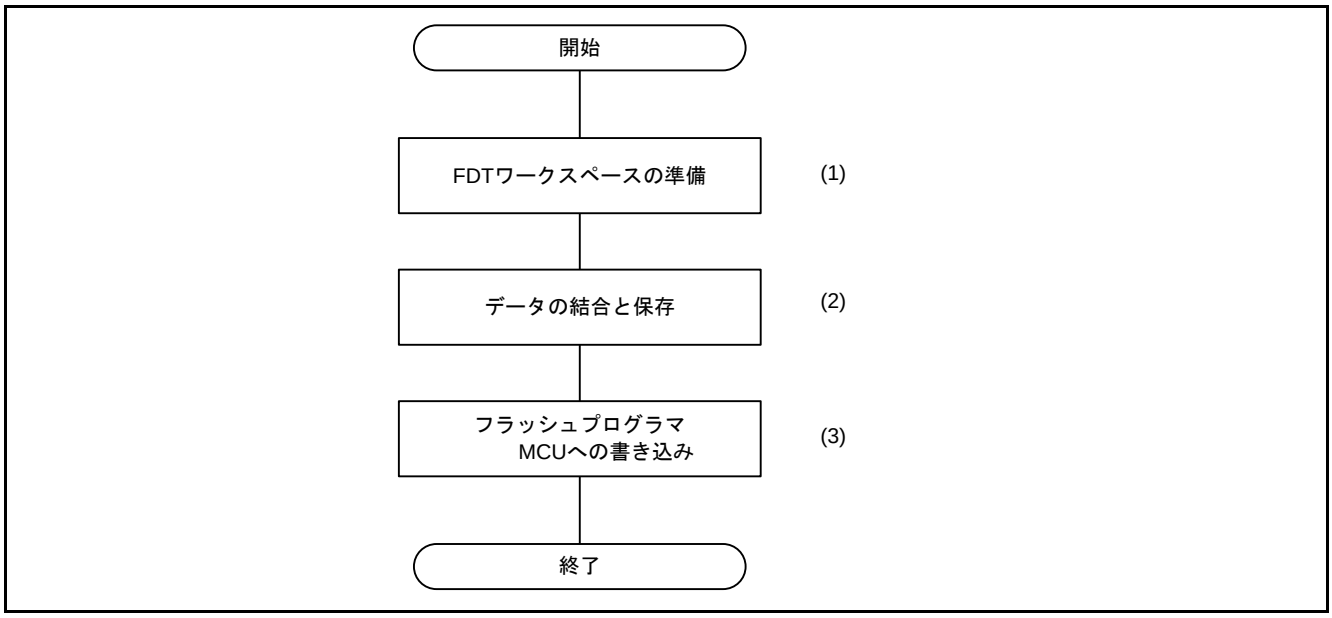


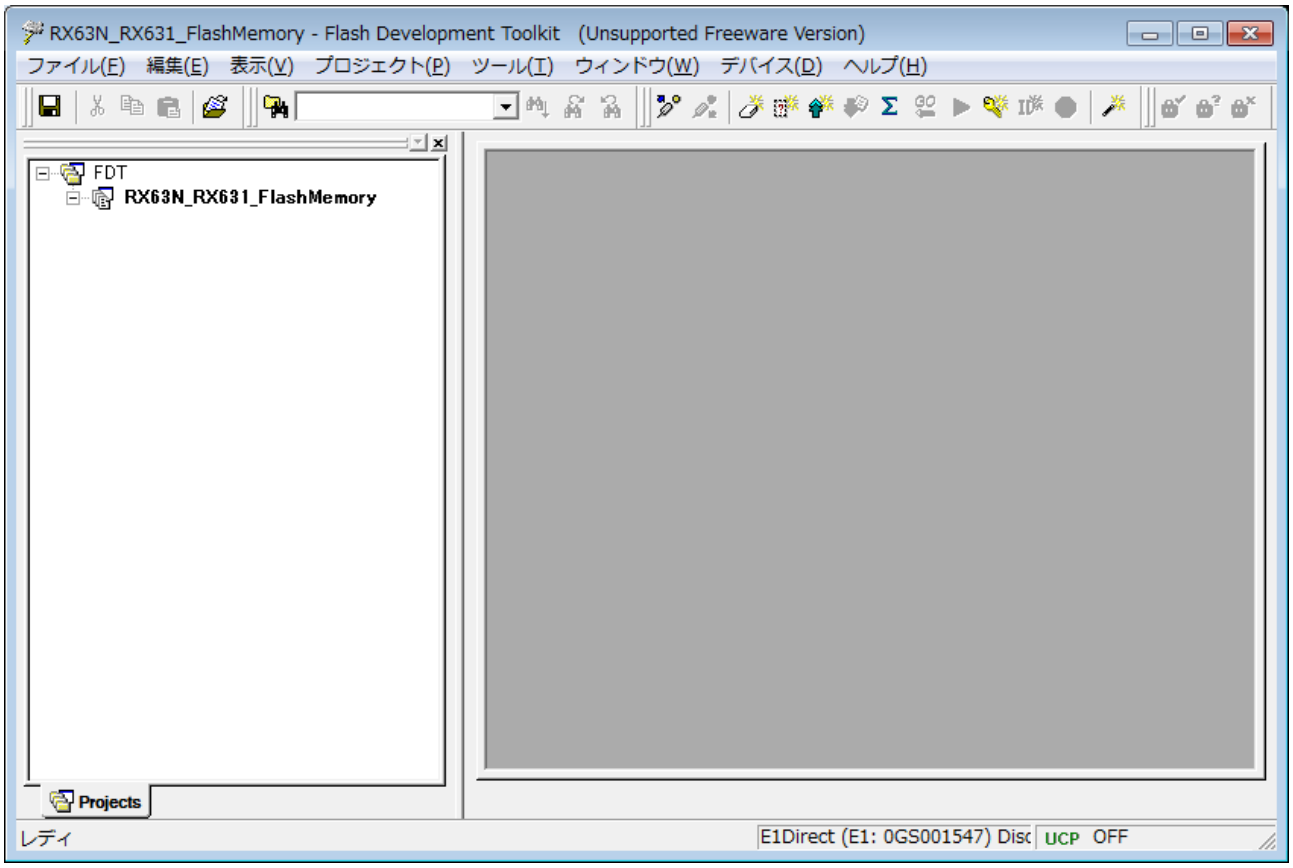
図5.1　RSK+RX63N への書き込み方法の流れ

(1)　「5.1.2 FDT ワークスペースの準備」を参照してください。
(2)　「5.1.3 データの結合と保存」を参照してください。
(3)　「5.1.4 RSK+RX63N のユーザ領域書き込み」を参照してください。

5.1.2　FDT ワークスペースの準備

FDT を使用するためにワークスペースとプロジェクトを作成します。ターゲットデバイスはフラッシュプログラマに使用する MCU を指定してください。

例では「ワークスペース名(W)」を“FDT”、「プロジェクト名(P)」を“RX63N_RX631_FlashMemory”としています。



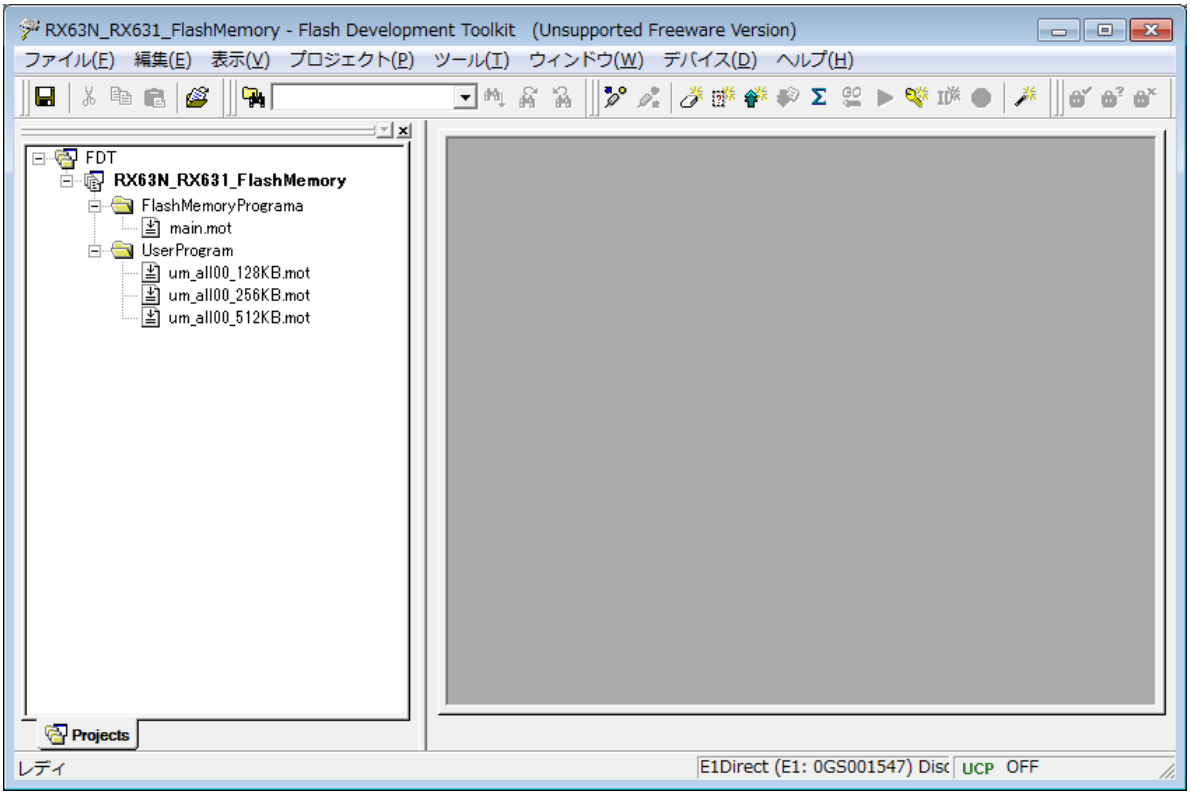
5.1.3　データの結合と保存

データの結合と保存は以下の(1)から(7)を行います。

(1)　FDT のプロジェクトに結合するデータファイルを追加します。

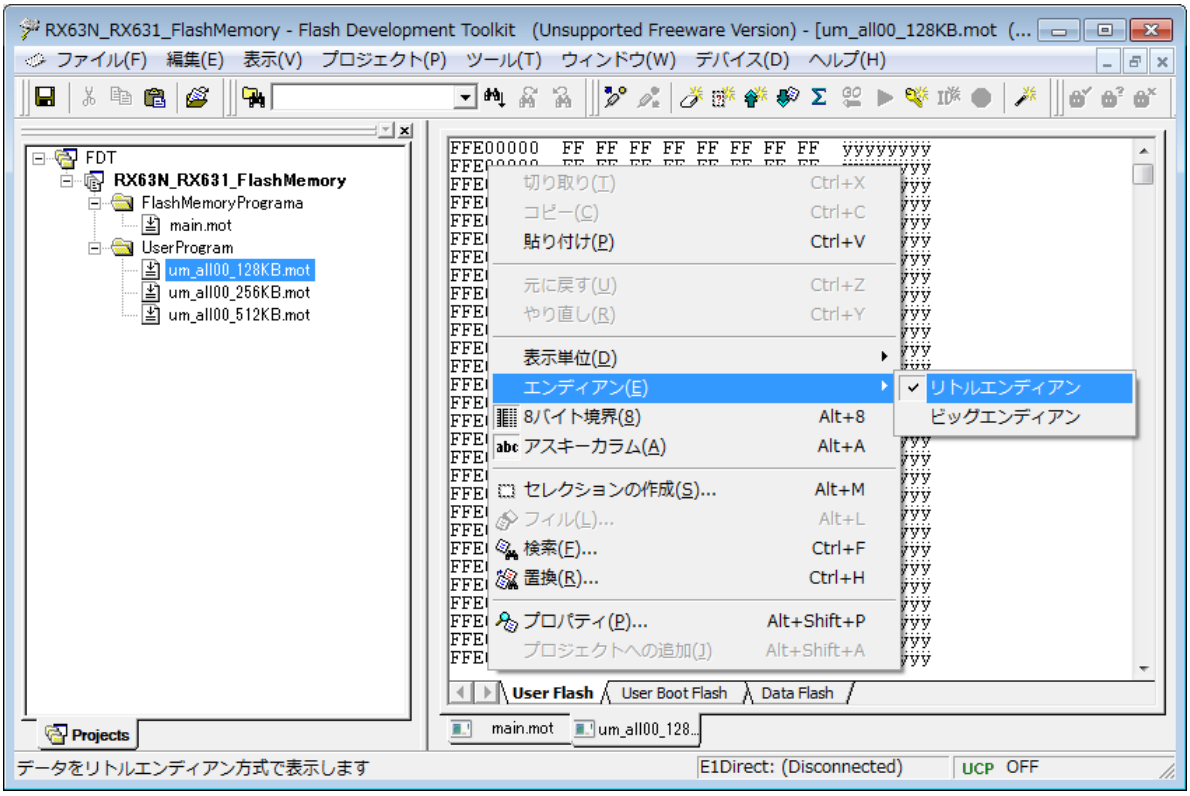
例では“RX63N_RX631_FlashMemory”プロジェクトに“FlashMemoryPrograma”フォルダと“UserProgram”フォルダを追加して、“FlashMemoryPrograma”フォルダにはフラッシュプログラマのプログラム“main.mot”ファイルを、“UserProgram”フォルダにはターゲット MCU のユーザ領域に書き込むユーザプログラムの容量ごとに以下のデータファイルを追加しています。

・ユーザプログラムの容量が 128K バイトは“um_all00_128KB.mot”ファイル
・ユーザプログラムの容量が 256K バイトは“um_all00_256KB.mot”ファイル
・ユーザプログラムの容量が 512K バイトは“um_all00_512KB.mot”ファイル



(2)　結合するデータファイルを 16 進数エディタウィンドウに開き、エンディアンを指定します。

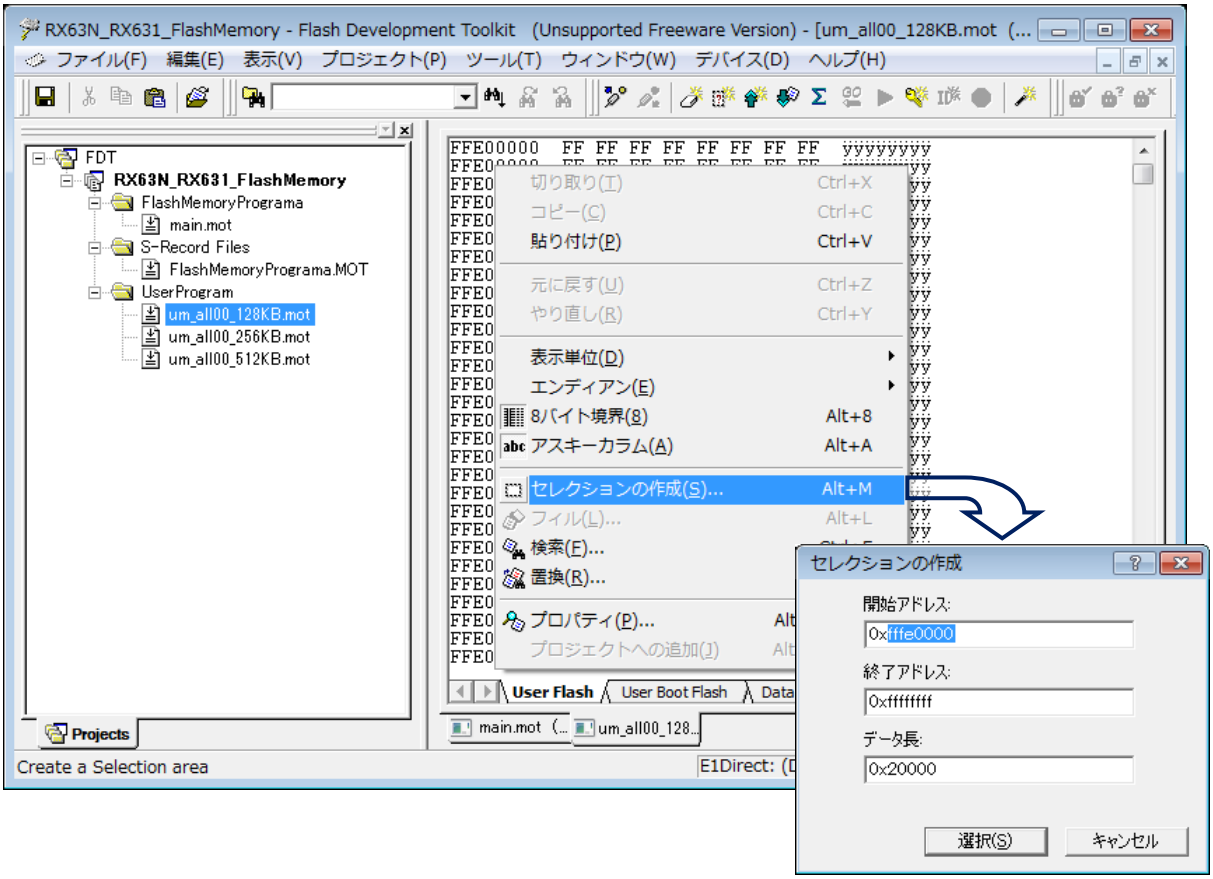
例では“main.mot”ファイルと“um_all00_128KB.mot”ファイルを 16 進数エディタウィンドウに開いています。また、エンディアンは 2 つのファイル共、リトルエンディアンを選択しています。



(3)　結合するターゲット MCU のユーザ領域に書き込むユーザプログラムデータを選択します。

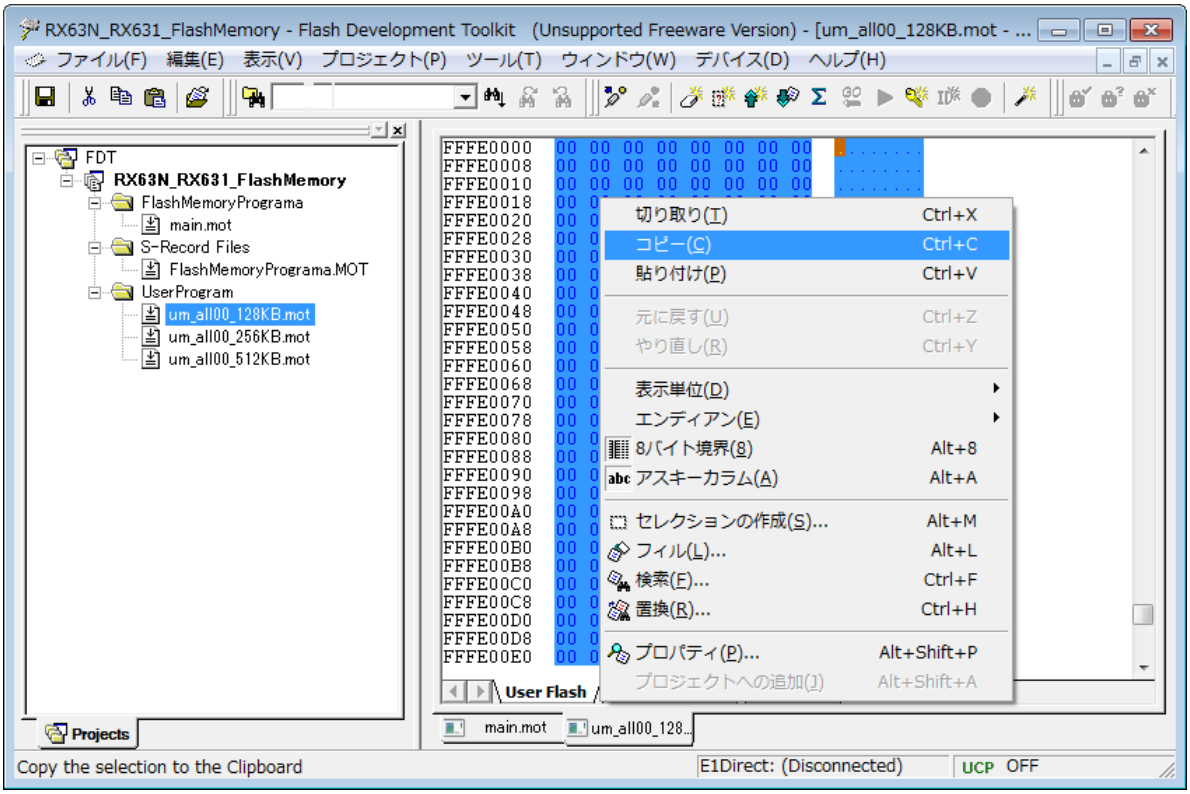
選択する範囲は以下のとおりです。
・ユーザプログラムの容量が 128K バイトの場合、0xFFFE 0000 番地から 0xFFFF FFFF 番地
・ユーザプログラムの容量が 256K バイトの場合、0xFFFC 0000 番地から 0xFFFF FFFF 番地
・ユーザプログラムの容量が 512K バイトの場合、0xFFF8 0000 番地から 0xFFFF FFFF 番地

例では“um_all00_128KB.mot”ファイルの 0xFFFE 0000 番地から 0xFFFF FFFF 番地を選択します。



(4)　反転表示しているターゲット MCU のユーザ領域に書き込むユーザプログラムデータを Windows のクリップボードにコピーします。

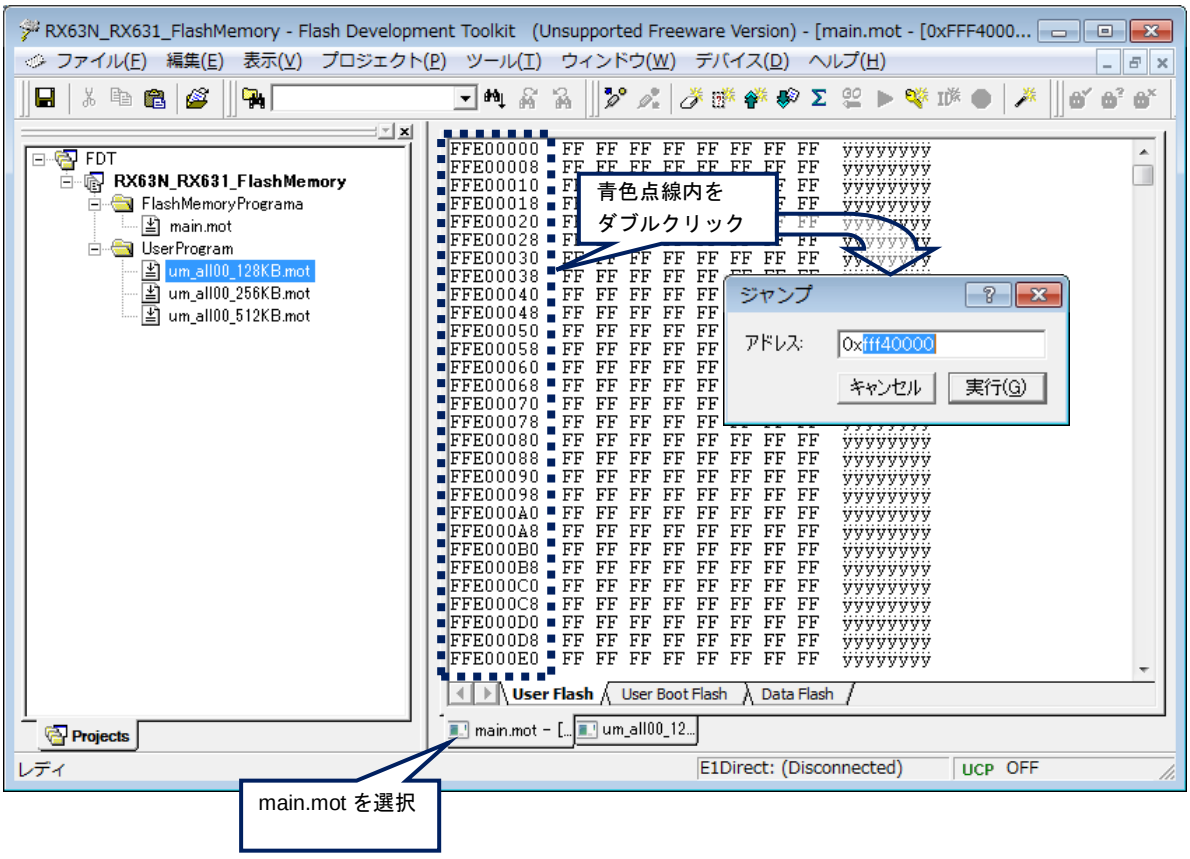
例では“um_all00_128KB.mot”ファイルの 0xFFFE 0000 番地から 0xFFFF FFFF 番地を Windows のクリップボードにコピーします。



(5)　RSK+RX63N のユーザ領域に書き込むデータを結合して作成します。

16 進数エディタウィンドウに“main.mot”ファイルを選択し、前項(4)で Windows のクリップボードにコピーしたデータを 0xFF40 0000 番地以降に貼り付けます。

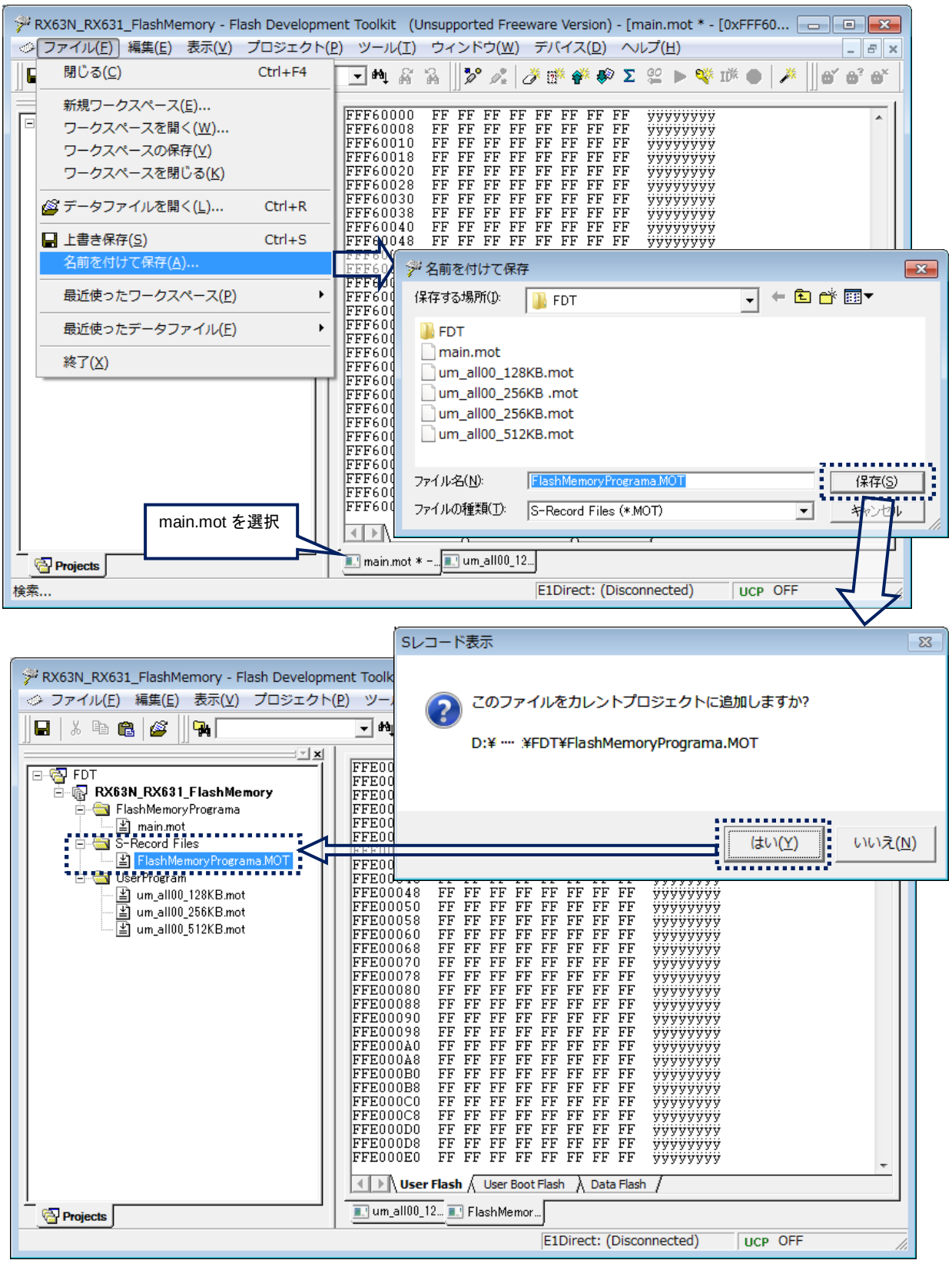
例は“main.mot”ファイルの貼り付け先の開始アドレスを 0xFF40 0000 番地に指定しています。その後、クリップボードのデータを貼り付けてください。



(6)　RSK+RX63N のユーザ領域に書き込むデータを保存します。

16 進数エディタウィンドウに“main.mot”ファイルを選択し、前項(5)で作成したデータに名前を付けてファイルに保存し、プロジェクトに追加します。

例では“S-Record Files”フォルダに“FlashMemoryPrograma.MOT”ファイルを保存します。

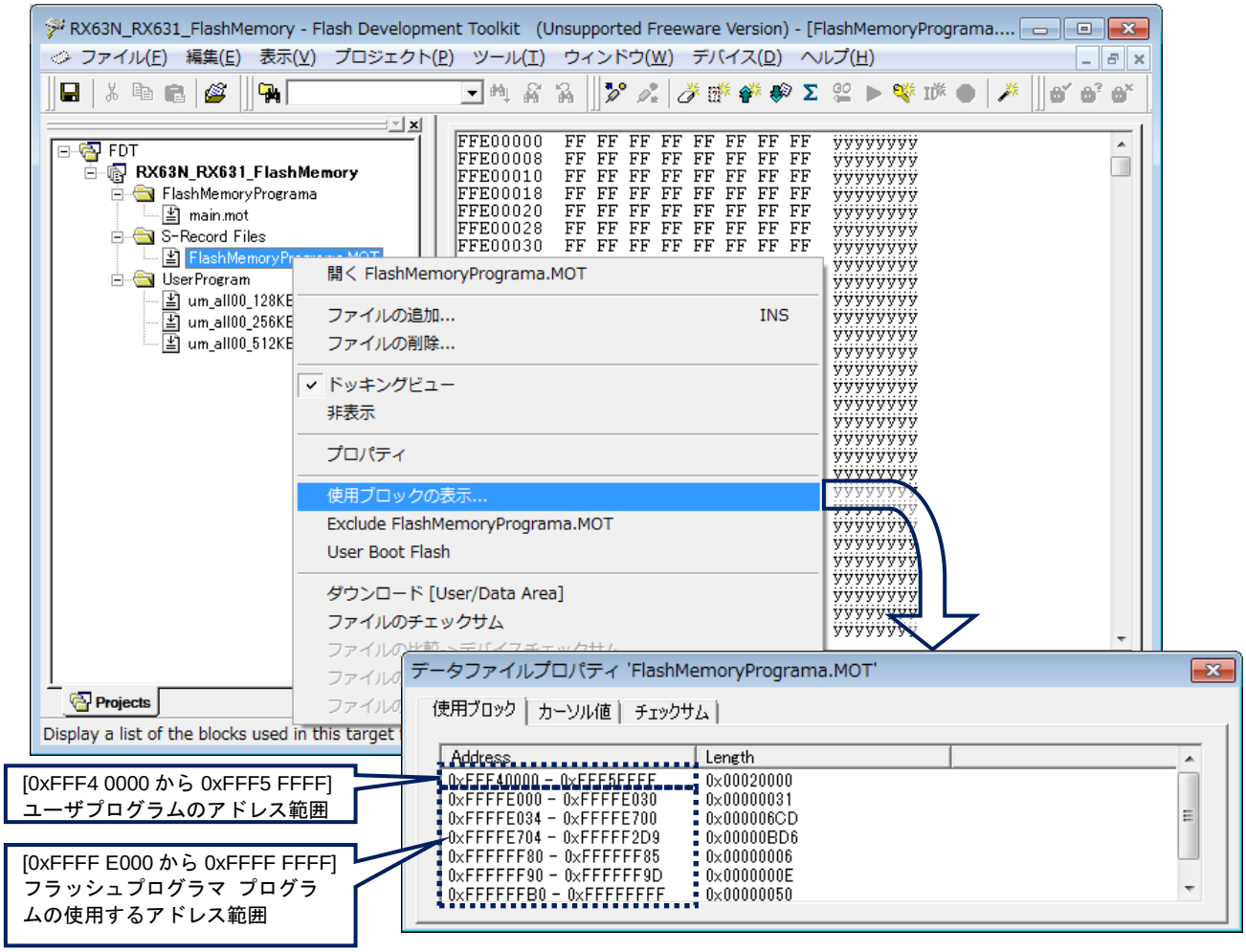


(7)　RSK+RX63N のユーザ領域に書き込むデータを確認します。

前項(6)で作成したデータファイルの結合したデータの配置が正しいか確認します。
ワークスペースウィンドウで RSK+RX63N のユーザ領域に書き込むデータファイルを選択し、使用ブロックのアドレス範囲を確認します。

確認する使用ブロックのアドレス範囲は以下のとおりです。
・ユーザプログラムの容量が 128K バイトの場合、0xFFF4 0000 番地から 0xFFF5 FFFF 番地
・ユーザプログラムの容量が 256K バイトの場合、0xFFF4 0000 番地から 0xFFF7 FFFF 番地
・ユーザプログラムの容量が 512K バイトの場合、0xFFF4 0000 番地から 0xFFFB FFFF 番地
・フラッシュプログラマのプログラムは、0xFFFF E000 番地から 0xFFFF FFFF 番地

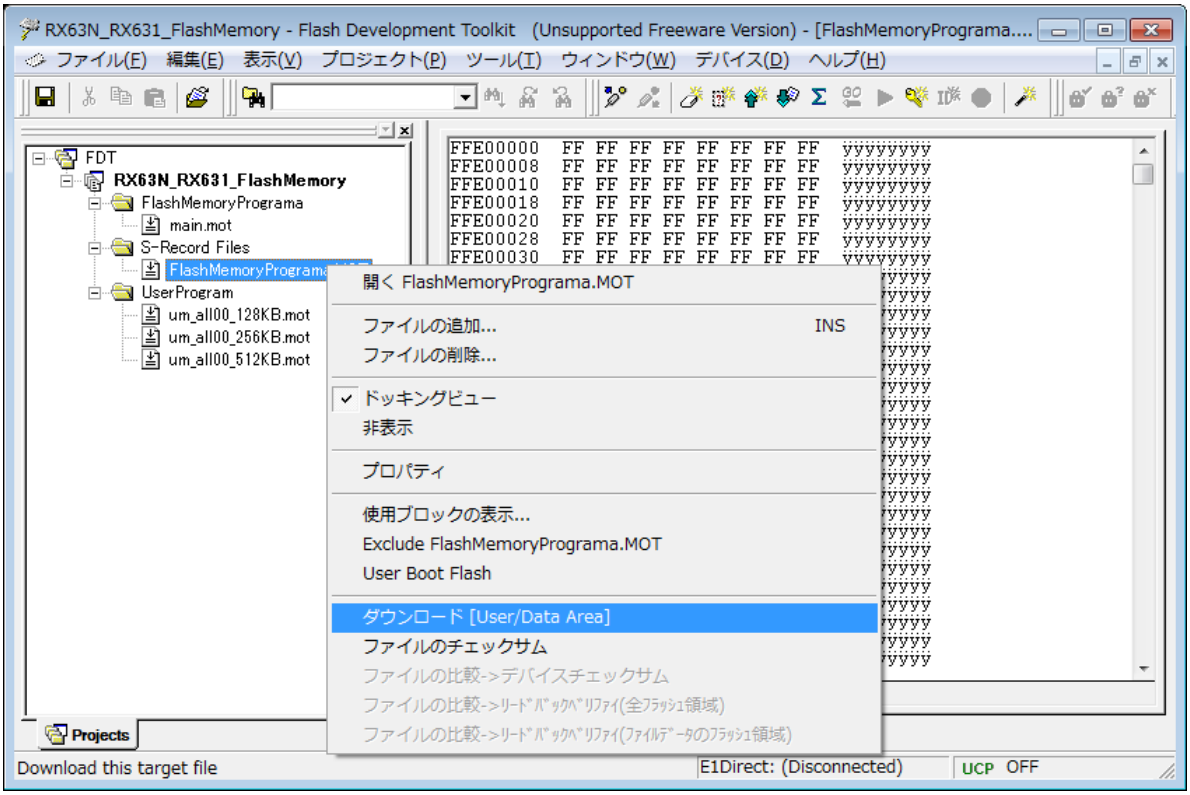
例ではユーザプログラムの容量が 128K バイトの場合の使用ブロック　アドレス範囲であることを確認しています。



5.1.4　RSK+RX63N のユーザ領域書き込み

ワークスペースウィンドウで RSK+RX63N のユーザ領域に書き込むデータファイルを選択してダウンロードします。

例では“S-Record Files”フォルダの“FlashMemoryPrograma.MOT”ファイルをダウンロードします。



5.2　動作概要

ターゲット MCU をブートモード(SCI インタフェース)で起動してビットレート自動調整を行い通信ビットレート 19200bps で接続します。

接続後、サポートデバイス問い合わせコマンド、デバイス選択コマンド、ブロック情報問い合わせコマンドを送信してターゲット MCU の情報を取得し、動作周波数選択コマンドを送信して通信ビットレートを 1Mbps に変更します。

プログラム/イレーズ遷移コマンドを送信してターゲット MCU の ID コードプロテクト状態を判断しブートモード ID コードプロテクトの処置を実行します。

取得したターゲット MCU の情報よりターゲット MCU のユーザ領域を消去後、書き込みます。書き込み完了後、ターゲット MCU に書き込んだ領域を読み出して書き込んだデータとベリファイチェックします。

図 5.2にフラッシュプログラマの状態遷移を示します。

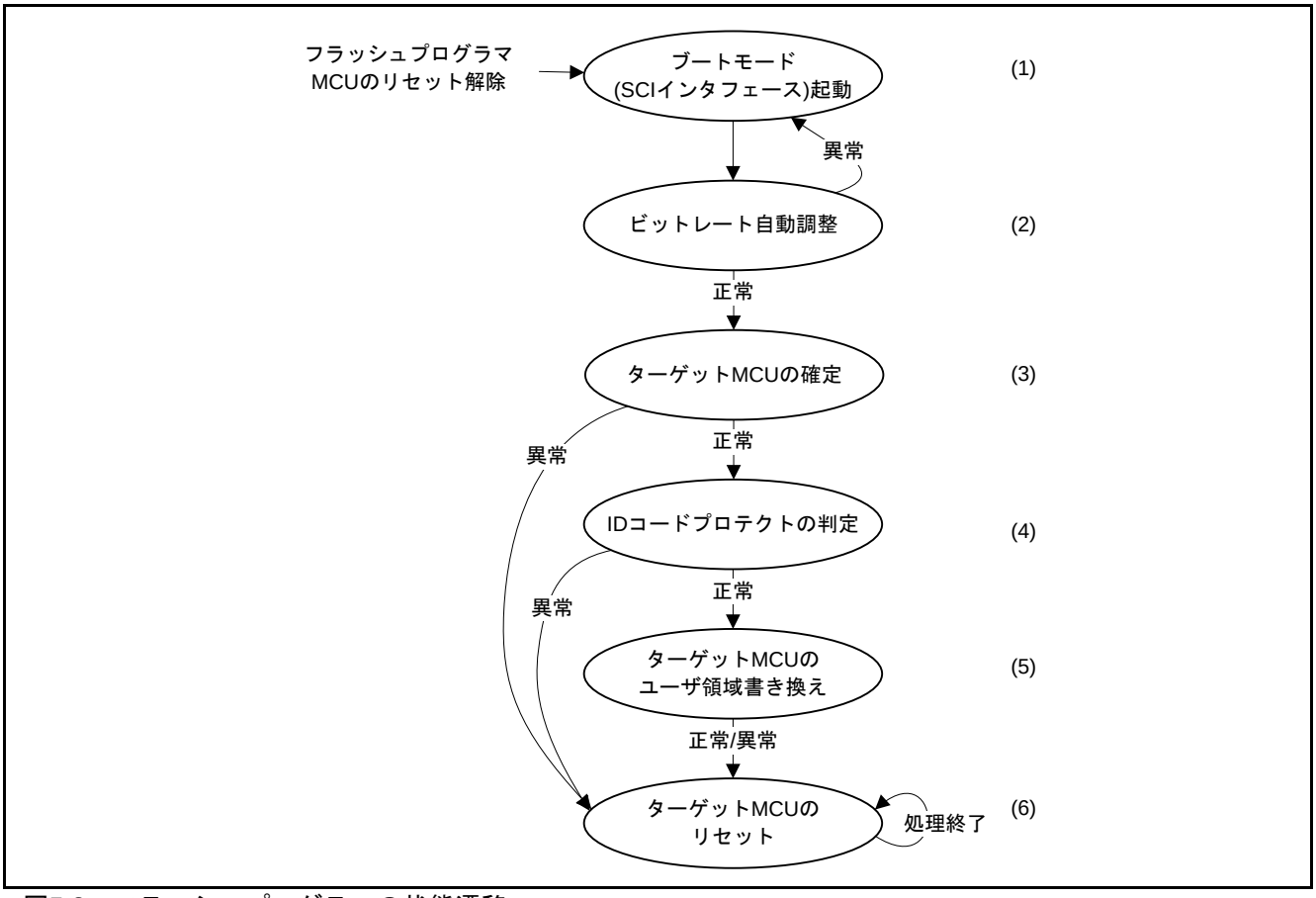


図5.2　フラッシュプログラマの状態遷移

(1)　詳細は「5.2.1 ブートモード(SCI インタフェース)起動」を参照してください。
(2)　詳細は「5.2.2 ビットレート自動調整」を参照してください。
(3)　詳細は「5.2.3 ターゲット MCU の確定」を参照してください。
(4)　詳細は「5.2.4 ID コードプロテクトの判定」を参照してください。
(5)　詳細は「5.2.5 ターゲット MCU のユーザ領域書き換え」を参照してください。
(6)　詳細は「5.2.6 ターゲット MCU のリセット」を参照してください。

5.2.1　ブートモード(SCI インタフェース)起動

(1) ターゲット MCU の RES#端子を“L”にします。
(2) ターゲット MCU の MD 端子を“L”にします。
(3) ターゲット MCU の UB#端子を“H”にします。
(4) 3ms のウェイト後、ターゲット MCU の RES#端子を“H”にします。

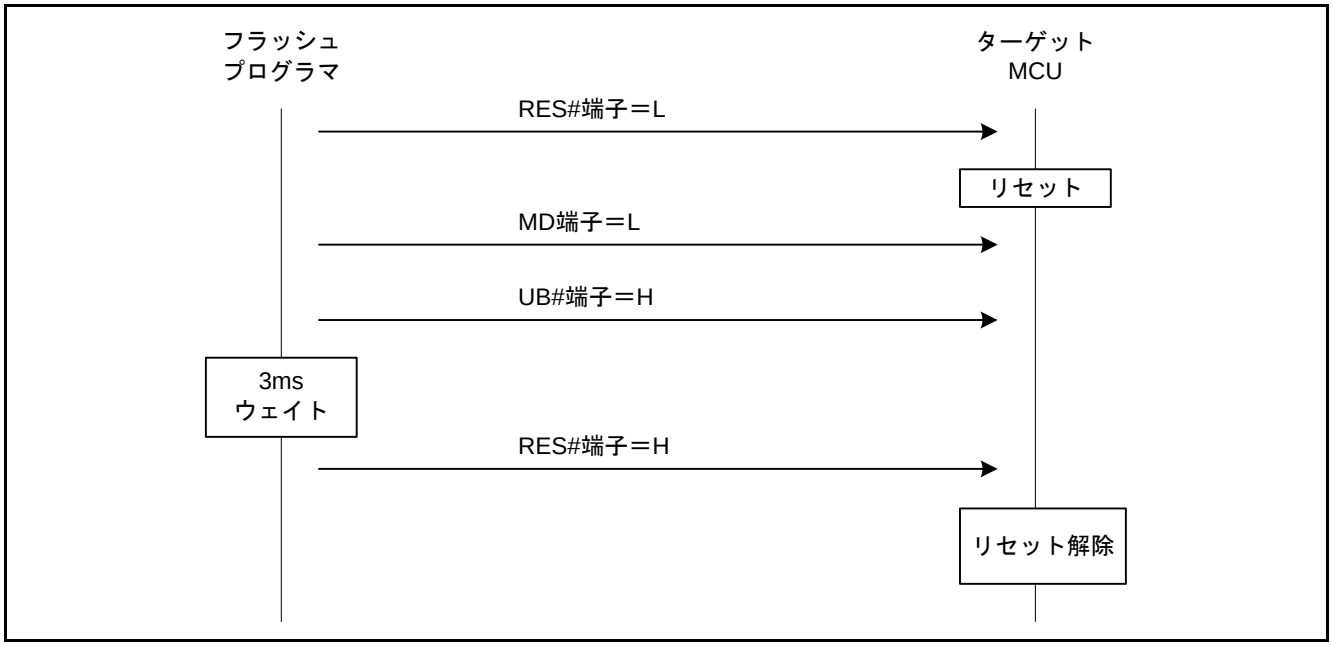


図5.3　ブートモード(SCI インタフェース)起動の手順

5.2.2　ビットレート自動調整

フラッシュプログラマは、ターゲット MCU をブートモード(SCI インタフェース)で起動して、400ms 経過した後にビットレートを 19,200bps に調整するために“00h”を 30 回送信します。

フラッシュプログラマが“00h”を受信した場合、ターゲット MCU に“55h”を送信してください。“00h”を受信できなかった場合、ターゲット MCU をブートモードで再起動し、再度ビットレート自動調整を実施します。

フラッシュプログラマは“55h”を送信後に“E6h”を受信することでビットレートの自動調整を完了します。“55h”を送信後に“FFh”を受信した場合、ターゲット MCU をブートモードで再起動し、再度ビットレート自動調整を実施します。

図 5.4にビットレート自動調整の手順を示します。

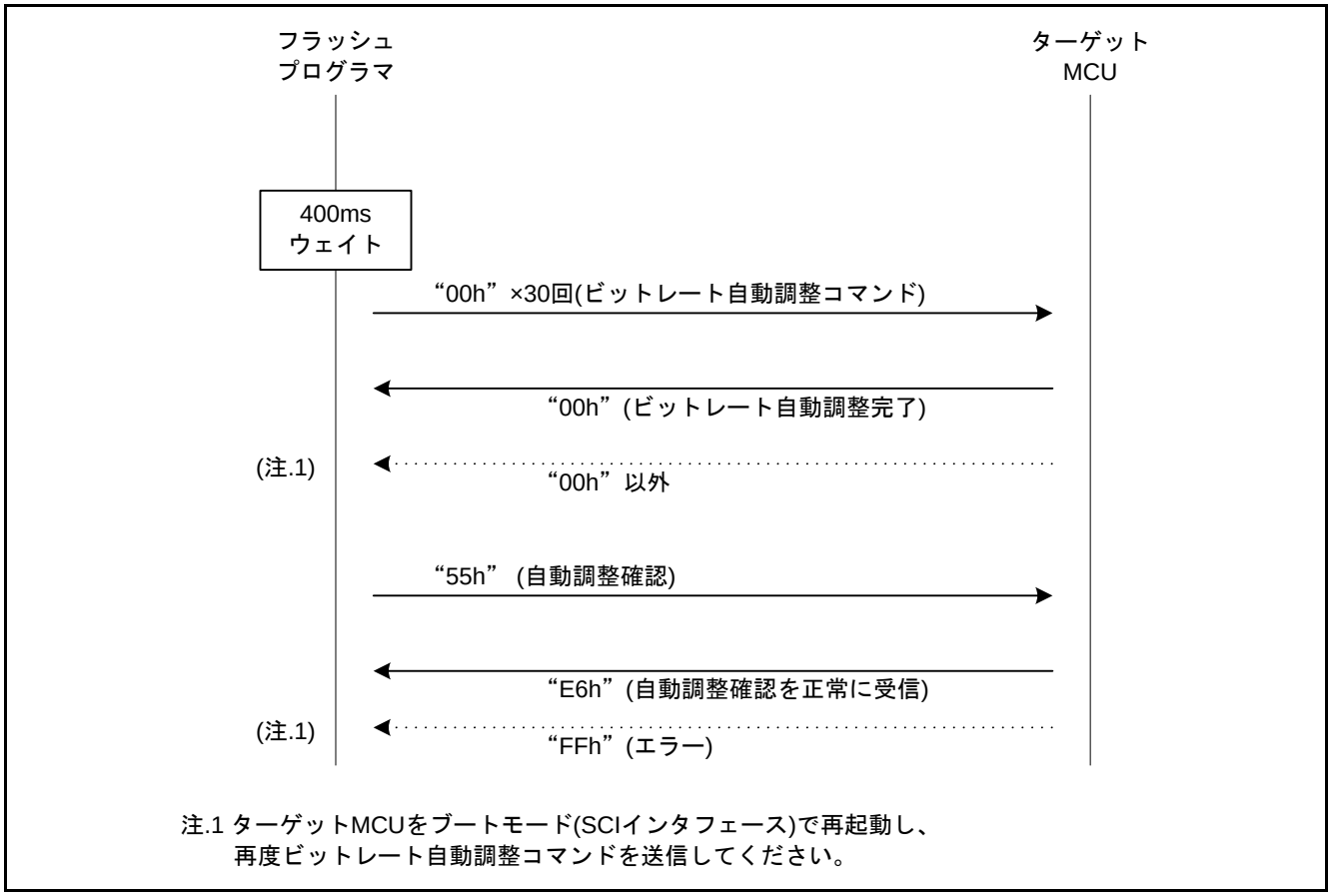


図5.4　ビットレート自動調整の手順

5.2.3　ターゲット MCU の確定

ターゲット MCU の確定は以下の(1)から(4)の処理を行います。

(1)　サポートデバイス問い合わせコマンドを送信してユーザ領域に書き込むデータのエンディアンを選択する識別コードを記憶します。

フラッシュプログラマはサポート問い合わせコマンドのレスポンス(30h から始まるデータ)を受信することでユーザ領域に書き込むデータのエンディアンを選択する識別コードを記憶します。レスポンス(30h から始まるデータ)以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.5に識別コードの記憶手順を示します。

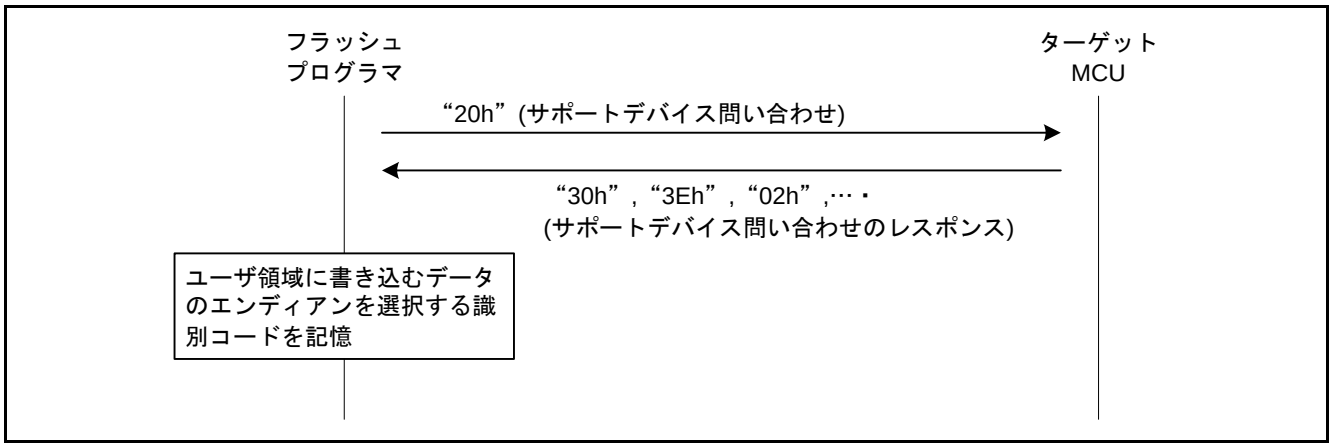


図5.5　識別コードの記憶手順

(2)　デバイス選択コマンドを送信して、ユーザ領域に書き込むデータのエンディアンを選択します。

フラッシュプログラマはユーザ領域に書き込むデータのエンディアンを指定するため、デバイス選択コマンド(10h)を送信します。このとき、指定する識別コードは、サポートデバイス問い合わせコマンドで記憶した識別コードの内、フラッシュプログラマのエンディアンに合わせた識別コードを用います。
フラッシュプログラマはデバイス選択コマンドを送信後にレスポンス(46h)を受信することでエンディアンの選択を完了します。デバイス選択コマンドを送信後にレスポンス(46h)以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.6にエンディアンの選択手順を示します。

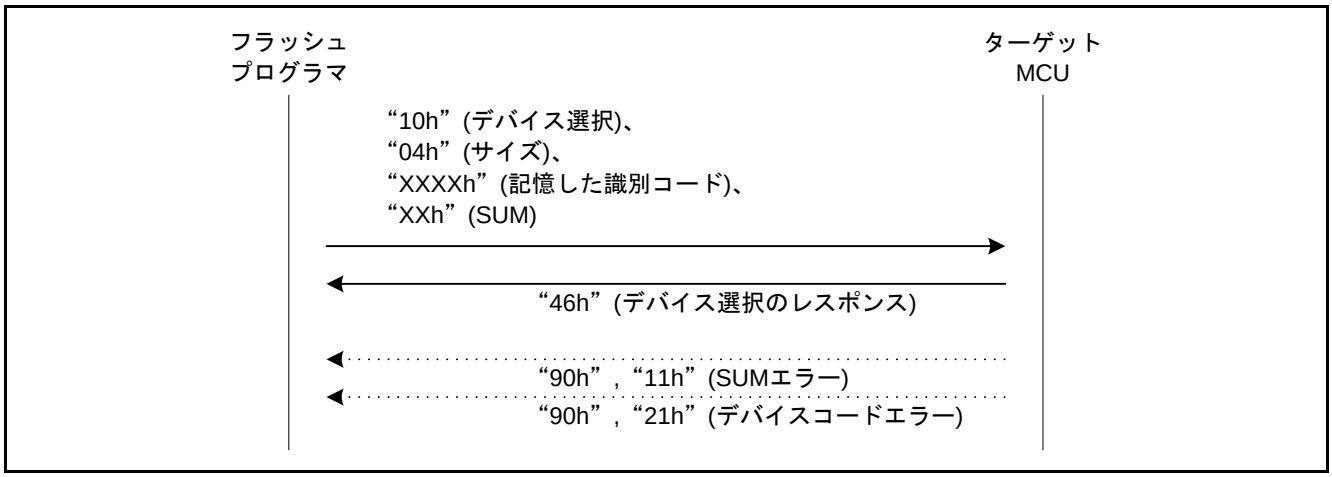


図5.6　エンディアンの選択手順

(3)　ブロック情報問い合わせコマンドを送信してターゲット MCU のブロック情報を記憶します。

フラッシュプログラマはブロック情報問い合わせコマンドのレスポンス(36h から始まるデータ)を受信することでターゲット MCU のブロック情報をブロック情報格納バッファに記憶します。ブロック情報問い合わせコマンドのレスポンス(36h から始まるデータ)以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.7にブロック情報の記憶手順を示します。

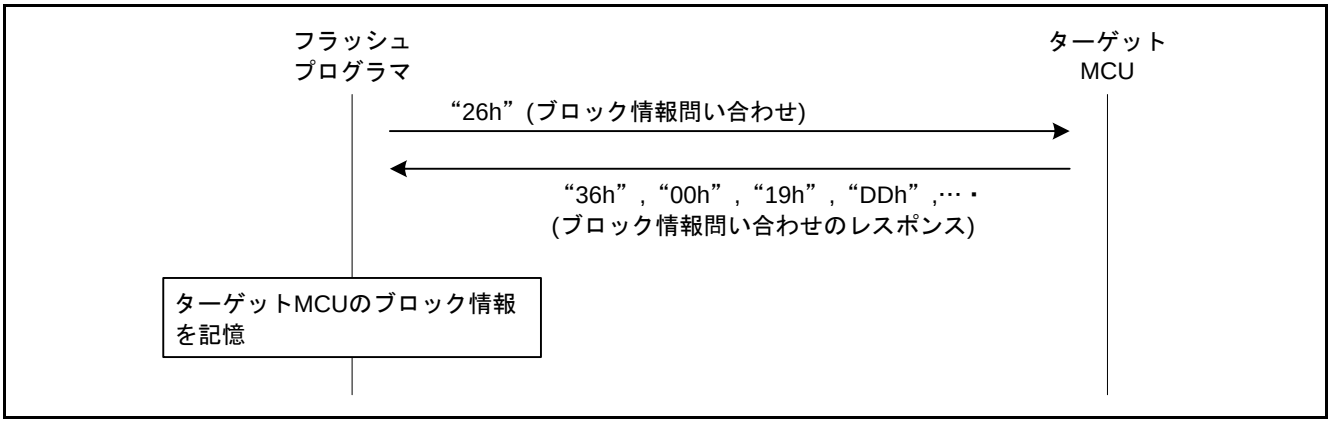


図5.7　ブロック情報の記憶手順

(4)　動作周波数選択コマンドを送信して、ターゲット MCU との通信ビットレートを 1Mbps に変更します。

フラッシュプログラマはビットレートを 1Mbps に変更するため、動作周波数選択コマンド(3Fh)を送信します。フラッシュプログラマは動作周波数選択コマンドを送信後に ACK(06h)を受信すると動作周波数選択コマンド送信時のビットレートで 1 ビット期間ウェイトし、ビットレートを 1Mbps に変更します。その後、変更後のビットレートで通信確認データ(06h)を送信し、通信確認データのレスポンス(06h)を受信することでビットレートの変更を完了します。
フラッシュプログラマは動作周波数選択コマンドを送信後にレスポンス(06h)以外を受信した場合や、通信確認データ(06h)を送信後に、通信確認データのレスポンス(06h) 以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.8にビットレートの変更手順を示します。

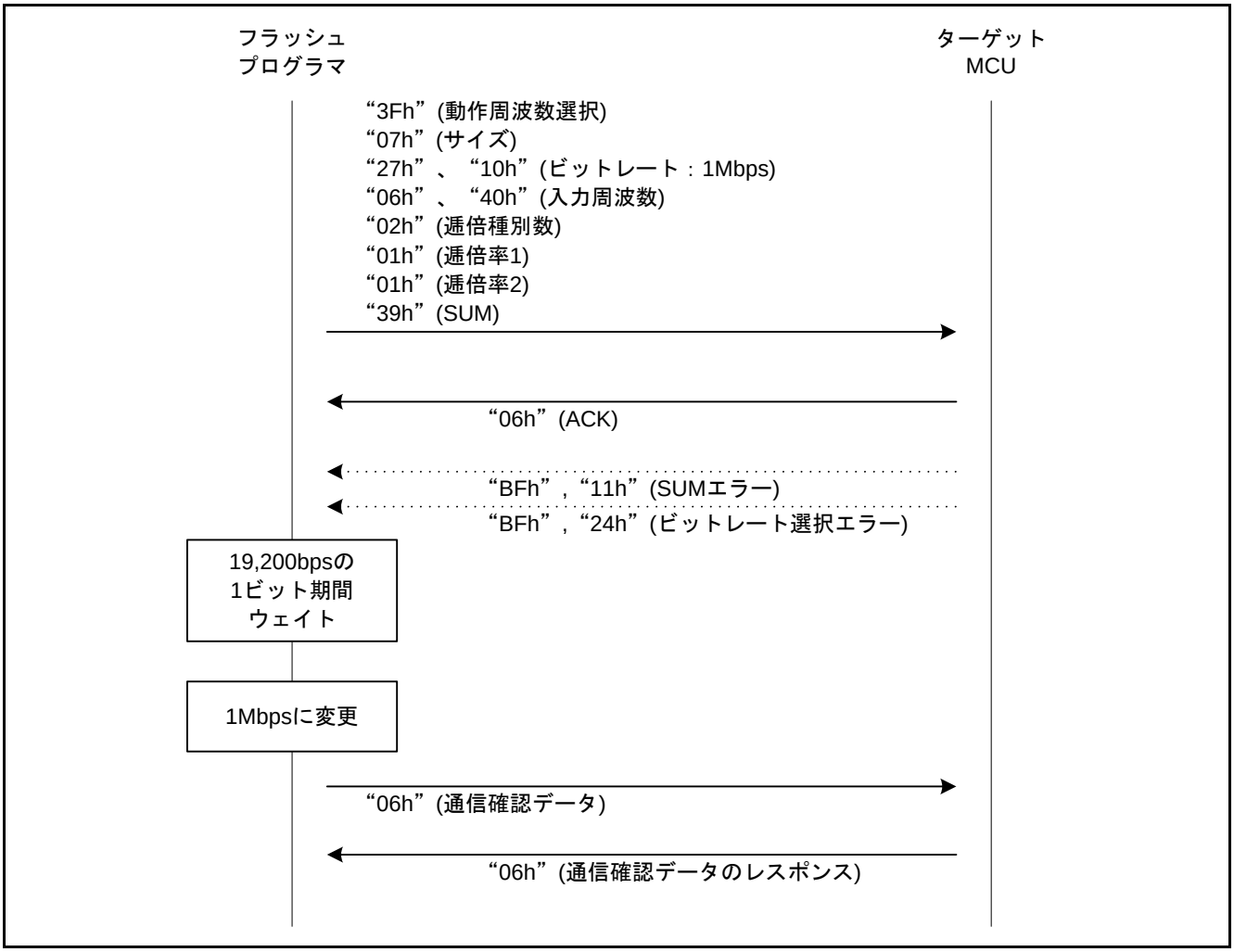


図5.8　ビットレートの変更手順

5.2.4　ID コードプロテクトの判定

ID コードプロテクトの判定は以下の(1)、(2)の処理を行います。

(1) プログラム/イレーズステータス遷移コマンドを送信してターゲット MCU の ID コードプロテクト状態を判定し記録します。

フラッシュプログラマはターゲット MCU の ID コードプロテクト状態を判定するため、プログラム/イレーズステータス遷移コマンド(40h)を送信します。
フラッシュプログラマはプログラム/イレーズステータス遷移コマンドを送信後に受信したレスポンスを判定して ID コードプロテクト状態判定バッファに記録します。
判定するレスポンスとID コードプロテクト状態判定バッファへの記録値を表 5.1に示します。

表5.1　ID コードプロテクト状態判定バッファへの記録値

レスポンス	ID コードプロテクト状態判定バッファへの記録値
“06h”	“00h”
“16h”	“01h”
“56h”	“02h”

フラッシュプログラマはプログラム/イレーズステータス遷移コマンドを送信後に受信したレスポンス値が表 5.1にない値の場合、ターゲット MCU をリセットして中断します。

図 5.9にプログラム/イレーズステータス遷移コマンドの ID コードプロテクト状態判定の手順を示します。

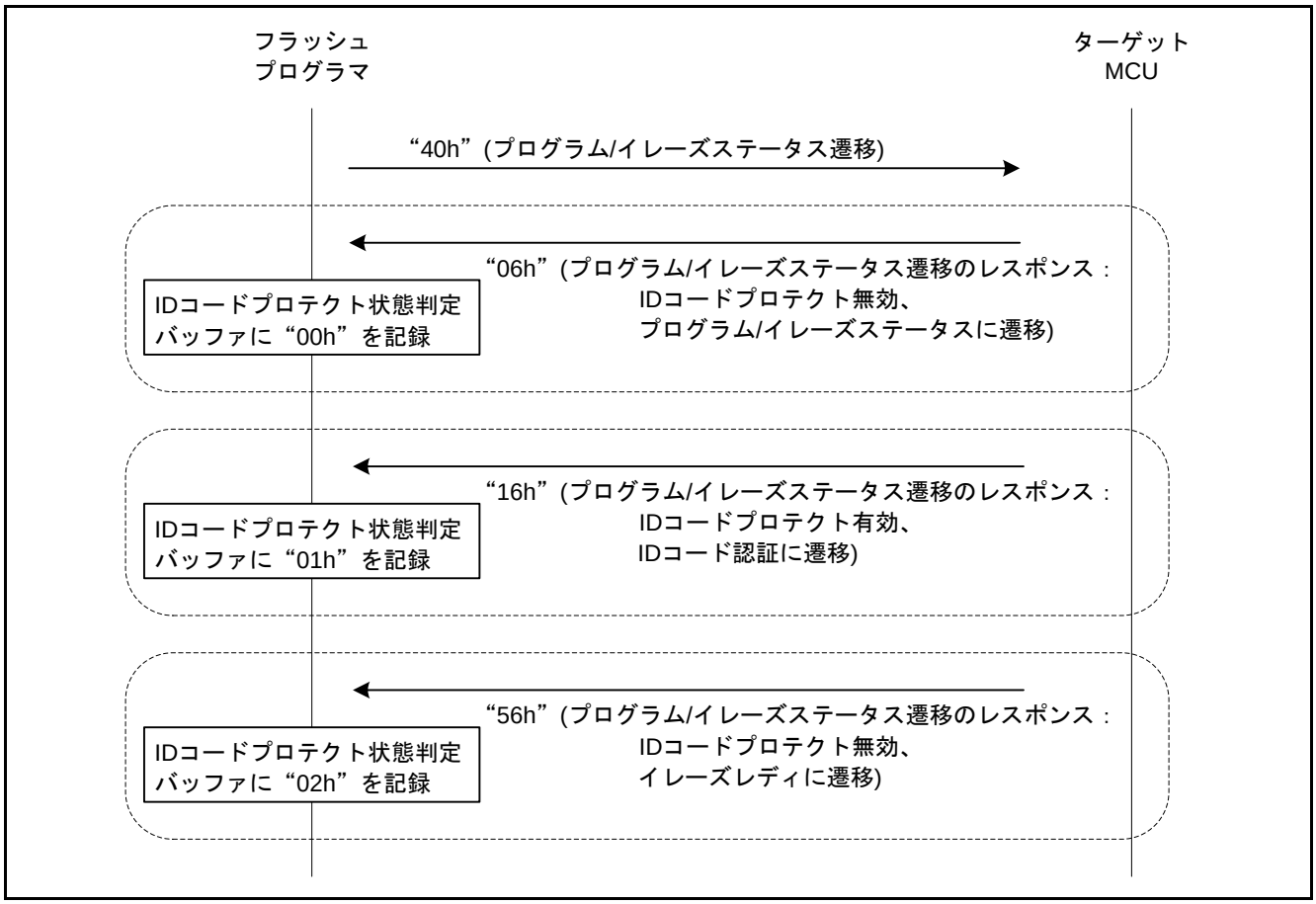


図5.9　プログラム/イレーズステータス遷移コマンドの ID コードプロテクト状態判定の手順

(2) ID コードチェックコマンドを送信してターゲット MCU の ID コードプロテクト状態を判定し記録します。

この処理は ID コードプロテクト状態判定バッファの記録値が“01h”の場合に実施します。
フラッシュプログラマはターゲット MCU の ID コードプロテクト状態を判定するため、ID コードチェックコマンド(60h)を送信します。このとき、指定する制御コード、および ID コード 1 から ID コード 15 はターゲット MCU のユーザ領域に書き込むデータを読み出して用います。
フラッシュプログラマは ID コードチェックコマンドを送信後に受信したレスポンスを判定して ID コードプロテクト状態判定バッファに記録します。
判定するレスポンスとID コードプロテクト状態判定バッファへの記録値を表 5.1に示します。

表5.2　ID コードプロテクト状態判定バッファへの記録値

レスポンス	ID コードプロテクト状態判定バッファへの記録値
“06h”	“00h”
“56h”	“02h”

フラッシュプログラマは ID コードチェックコマンドを送信後に受信したレスポンス値が表 5.2にない値の場合、ターゲット MCU をリセットして中断します。

図 5.10にID コードチェックコマンドの ID コードプロテクト状態判定の手順を示します。

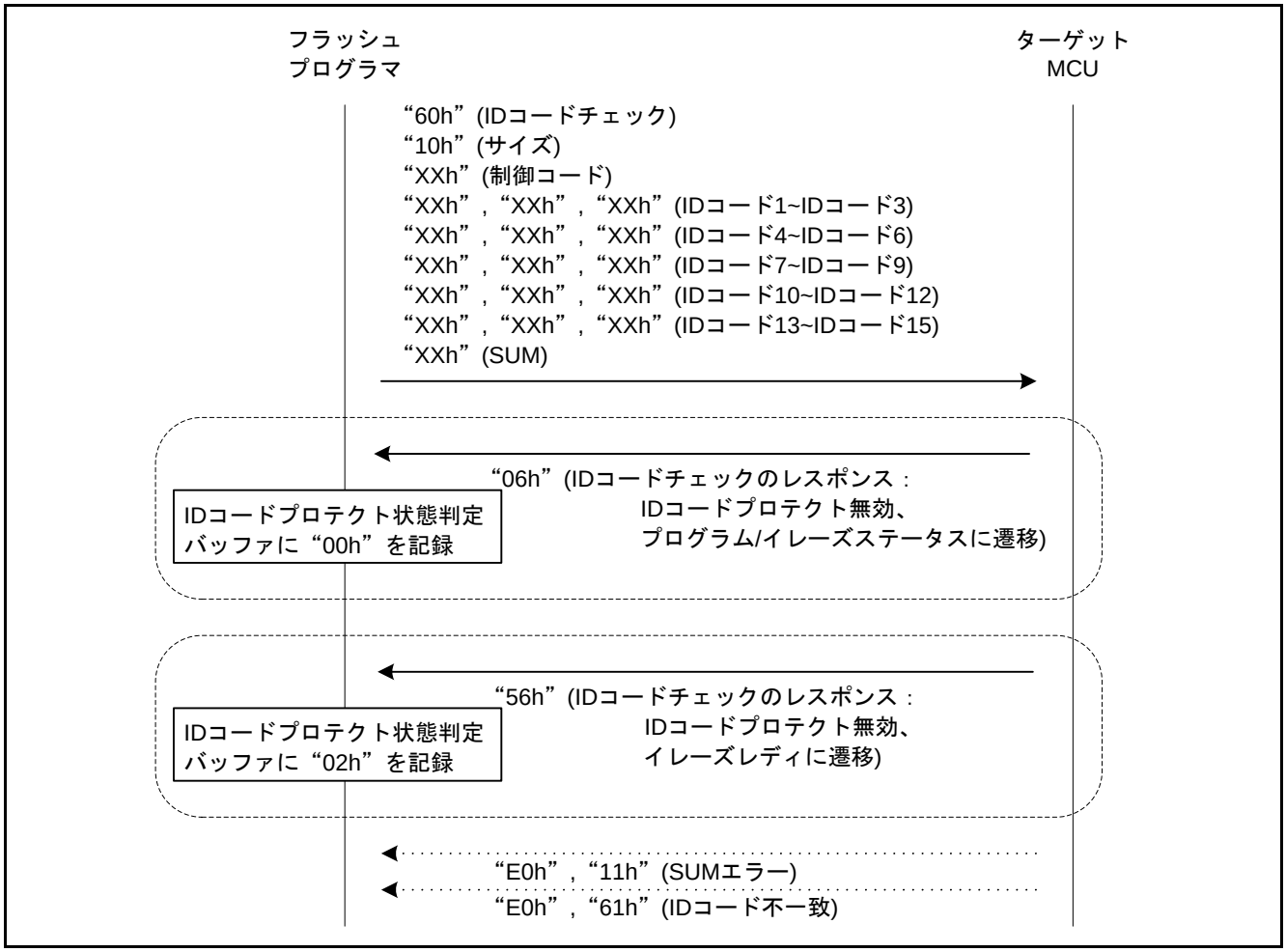


図5.10　ID コードチェックコマンドの ID コードプロテクト状態判定の手順

5.2.5　ターゲット MCU のユーザ領域書き換え

ターゲット MCU のユーザ領域書き換えは以下の(1)から(4)の処理を行います。

(1) ターゲット MCU のユーザ領域にユーザプログラムを書き込むため、ターゲット MCU のフラッシュメモリを消去します。

フラッシュプログラマは先ずイレーズ準備コマンド(48h)を送信します。イレーズ準備コマンドを送信後にイレーズ準備のレスポンス(06h)を受信することでイレーズ準備を完了します。イレーズ準備コマンドを送信後にイレーズ準備のレスポンス(06h)以外を受信した場合、ターゲット MCU をリセットして中断します。
次にブロックイレーズコマンド(59h)の送信を消去するブロック数回、繰り返します。消去するブロック数は ID コードプロテクト状態判定バッファへの記録値から、以下のとおり求めます。
・記録値が“02h”の場合、ブロック情報格納バッファ 3 とブロック情報格納バッファ 6 を加算した値
・記録値が“02h”以外の場合、ブロック情報格納バッファ 3 の値
消去するブロック数回、ブロックイレーズコマンドを送信すると、ブロックイレーズを終了するブロックイレーズコマンド(59h 04h FFh FFh FFh FFh A7h)を送信します。ブロックイレーズコマンドを送信後にブロックイレーズのレスポンス(06h)を受信すると、1 回のブロックイレーズを完了します。ブロックイレーズコマンドを送信後にブロックイレーズのレスポンス(06h)以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.11にターゲット MCU のフラッシュメモリ消去の手順を示します。

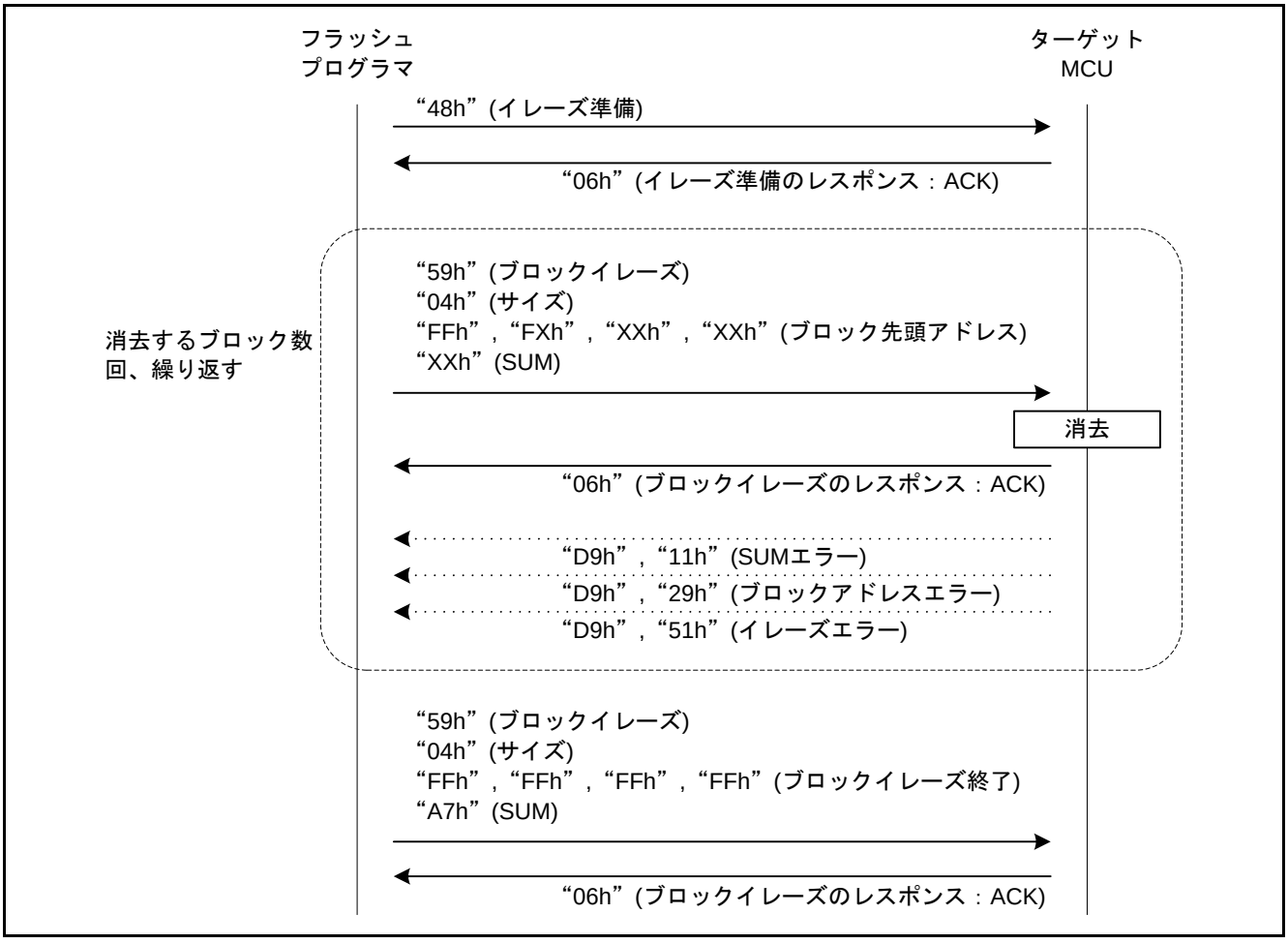


図5.11　ターゲット MCU のフラッシュメモリ消去の手順

(2)　ターゲット MCU がフラッシュメモリの消去を正常に終了したこと確認します。

フラッシュプログラマはブートモードステータス問い合わせコマンド(4Fh)送信します。
ブートモードステータス問い合わせコマンドを送信後にブートモードステータス問い合わせコマンドのレスポンス(5Fh から始まるデータ)を受信することでターゲット MCU がフラッシュメモリの消去を正常に終了したこと確認します。
ブートモードステータス問い合わせのレスポンス(5Fh から始まるデータ)以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.12にターゲット MCU の書き込み準備完了の手順を示します。

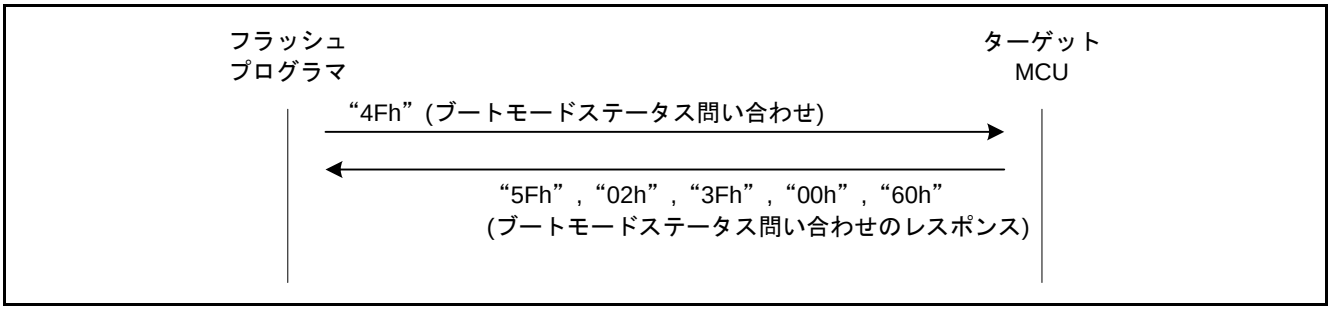


図5.12　ターゲット MCU の書き込み準備完了の手順

(3)　ターゲット MCU のユーザ領域にユーザプログラムを書き込みます。

フラッシュプログラマは先ずユーザ/データ領域プログラム準備コマンド(43h)を送信します。ユーザ/データ領域プログラム準備コマンドを送信後にユーザ/データ領域プログラム準備のレスポンス(06h)を受信することでプログラム準備を完了します。ユーザ/データ領域プログラム準備コマンドを送信後にユーザ/データ領域プログラム準備のレスポンス(06h)以外を受信した場合、ターゲット MCU をリセットして中断します。
次にプログラムコマンド(50h)をプログラムアドレスは 256 バイトでアライメントされたアドレス、プログラムデータは 256 バイト単位に指定してターゲット MCU のユーザ領域に書き込むユーザプログラムの容量分、送信します。

プログラムアドレス(ターゲット MCU の書き込み先)の範囲は以下のとおりです。
・ユーザプログラムの容量が 128K バイトの場合、0xFFFE 0000 番地から 0xFFFF FFFF 番地
・ユーザプログラムの容量が 256K バイトの場合、0xFFFC 0000 番地から 0xFFFF FFFF 番地
・ユーザプログラムの容量が 512K バイトの場合、0xFFF8 0000 番地から 0xFFFF FFFF 番地

プログラムデータ(RSK+RX63N 上の書き込み元データ)の範囲は以下のとおりです。
・ユーザプログラムの容量が 128K バイトの場合、0xFFF4 0000 番地から 0xFFF5 FFFF 番地
・ユーザプログラムの容量が 256K バイトの場合、0xFFF4 0000 番地から 0xFFF7 FFFF 番地
・ユーザプログラムの容量が 512K バイトの場合、0xFFF4 0000 番地から 0xFFFB FFFF 番地

ターゲット MCU のユーザ領域に書き込むユーザプログラムの容量分、プログラムコマンドを送信すると、プログラムを終了するプログラムコマンド(50h FFh FFh FFh FFh B4h)を送信します。プログラムコマンドを送信後にプログラムのレスポンス(06h)を受信すると、1 回のプログラムを完了します。プログラムコマンドを送信後にプログラムのレスポンス(06h)以外を受信した場合、ターゲット MCU をリセットして中断します。

図 5.13にユーザ領域をプログラムする手順を示します。

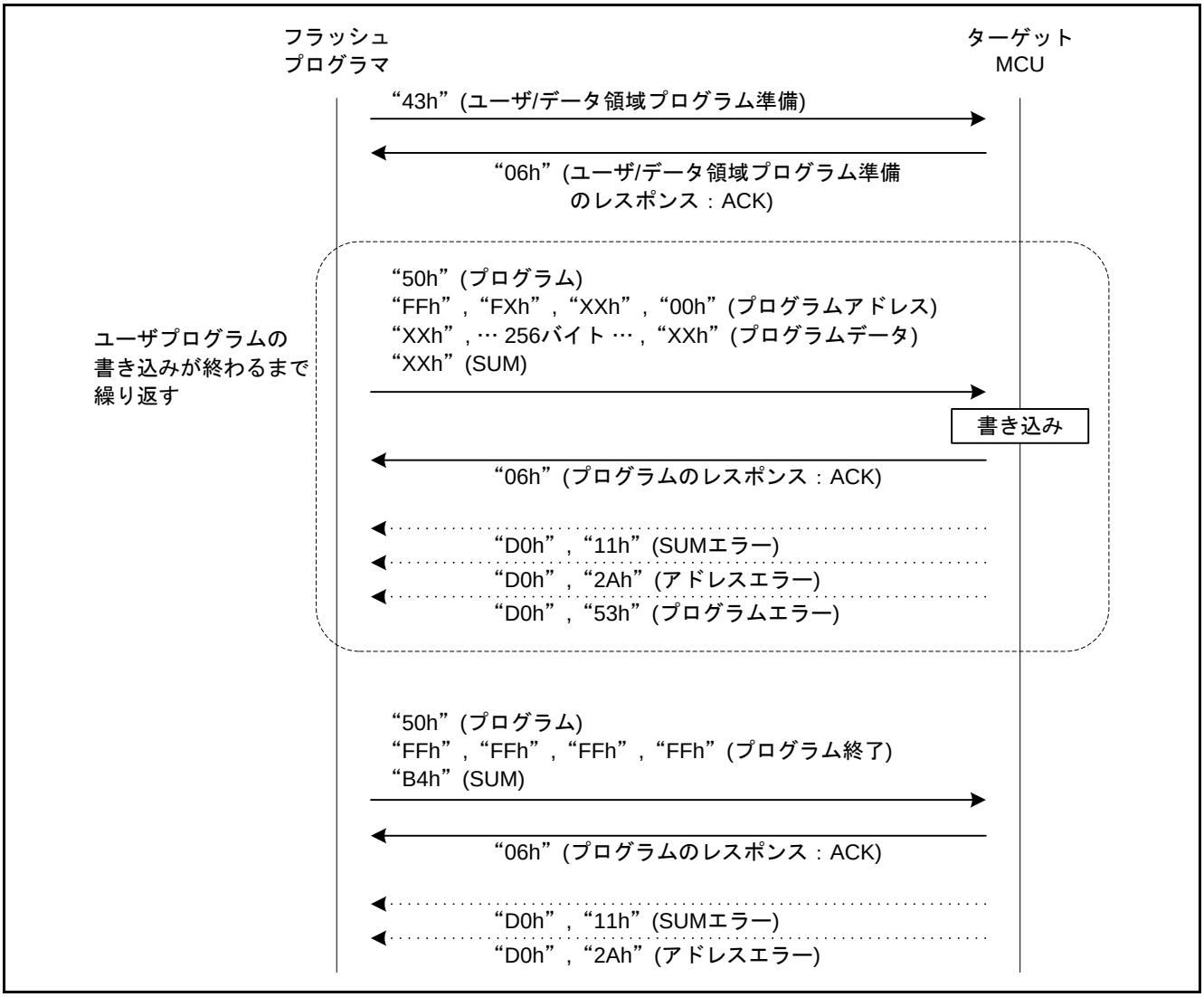


図5.13　ユーザ領域をプログラムする手順

(4)　ターゲット MCU のユーザ領域に正しく書き込めたことを確認します。

フラッシュプログラマはターゲット MCU のユーザ領域に書き込んだデータを確認するため、ターゲット MCU のユーザ領域にあるデータを読み出して書き込んだ値と比較します。
フラッシュプログラマはメモリリードコマンド(52h)を読み出しアドレスは256バイトでアライメントされたアドレスを指定してターゲット MCU のユーザ領域に書き込むユーザプログラムの容量分、送信します。

読み出しアドレスの範囲は以下のとおりです。
・ユーザプログラムの容量が 128K バイトの場合、0xFFFE 0000 番地から 0xFFFF FFFF 番地
・ユーザプログラムの容量が 256K バイトの場合、0xFFFC 0000 番地から 0xFFFF FFFF 番地
・ユーザプログラムの容量が 512K バイトの場合、0xFFF8 0000 番地から 0xFFFF FFFF 番地

メモリリードコマンドを送信後にメモリリードのレスポンス(52h から始まるデータ)を受信すると、RSK+RX63N のユーザ領域の書き込み元データと比較します。比較結果が一致しなかった場合や、メモリリードコマンドを送信後にメモリリードのレスポンス(52h から始まるデータ)以外を受信した場合、ターゲット MCU をリセットして中断します。

書き込み元データの範囲は以下のとおりです。
・ユーザプログラムの容量が 128K バイトの場合、0xFFF4 0000 番地から 0xFFF5 FFFF 番地
・ユーザプログラムの容量が 256K バイトの場合、0xFFF4 0000 番地から 0xFFF7 FFFF 番地
・ユーザプログラムの容量が 512K バイトの場合、0xFFF4 0000 番地から 0xFFFB FFFF 番地

図 5.14にユーザ領域のデータを確認する手順を示します。

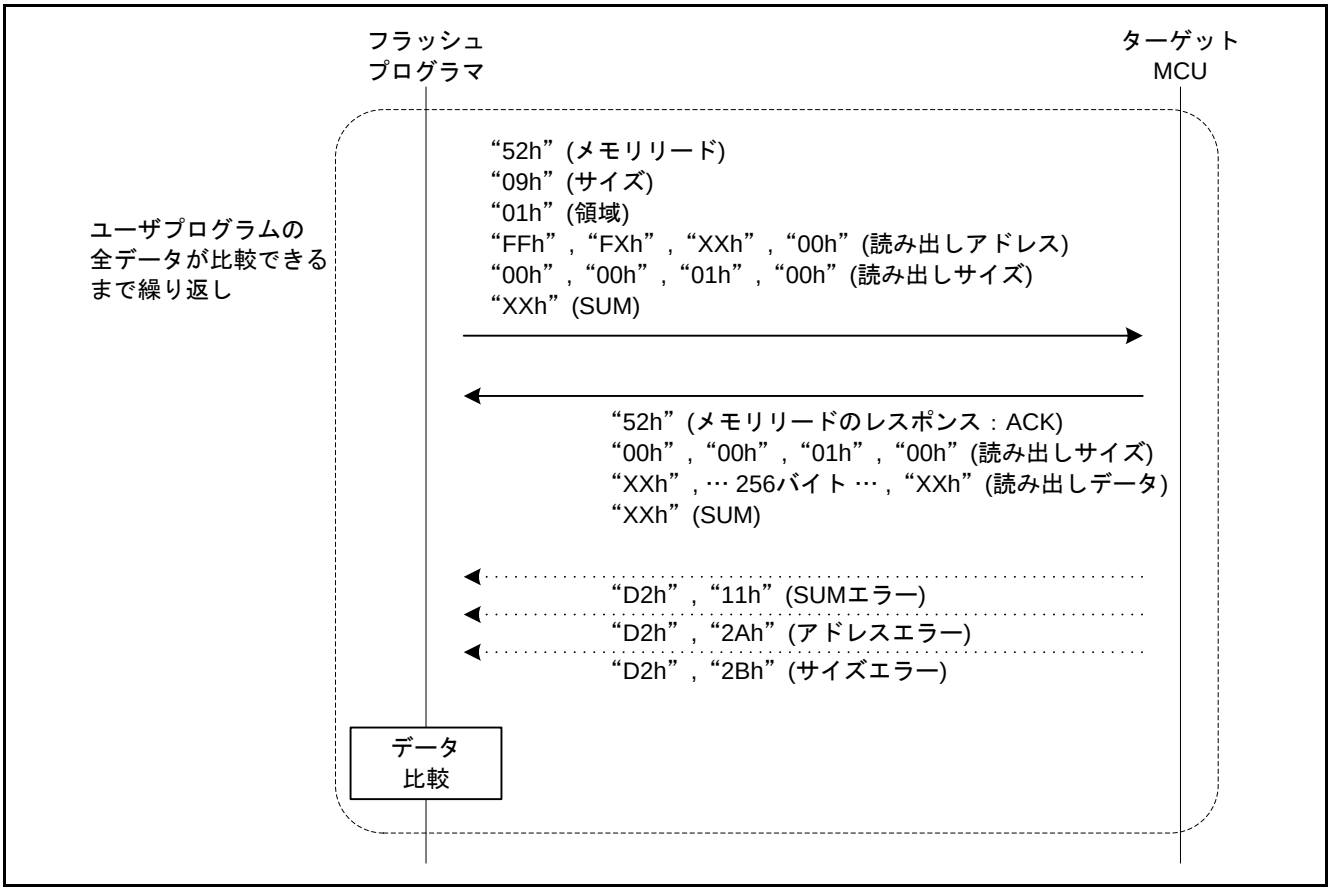


図5.14　ユーザ領域のデータを確認する手順

5.2.6　ターゲット MCU のリセット

(1)　ターゲット MCU の MD 端子を“H”にします。
(2)　ターゲット MCU の RES#端子を“L”にします。
(3)　3ms のウェイト後、ターゲット MCU の RES#端子を“H”にします。
(4)　無限ループします。

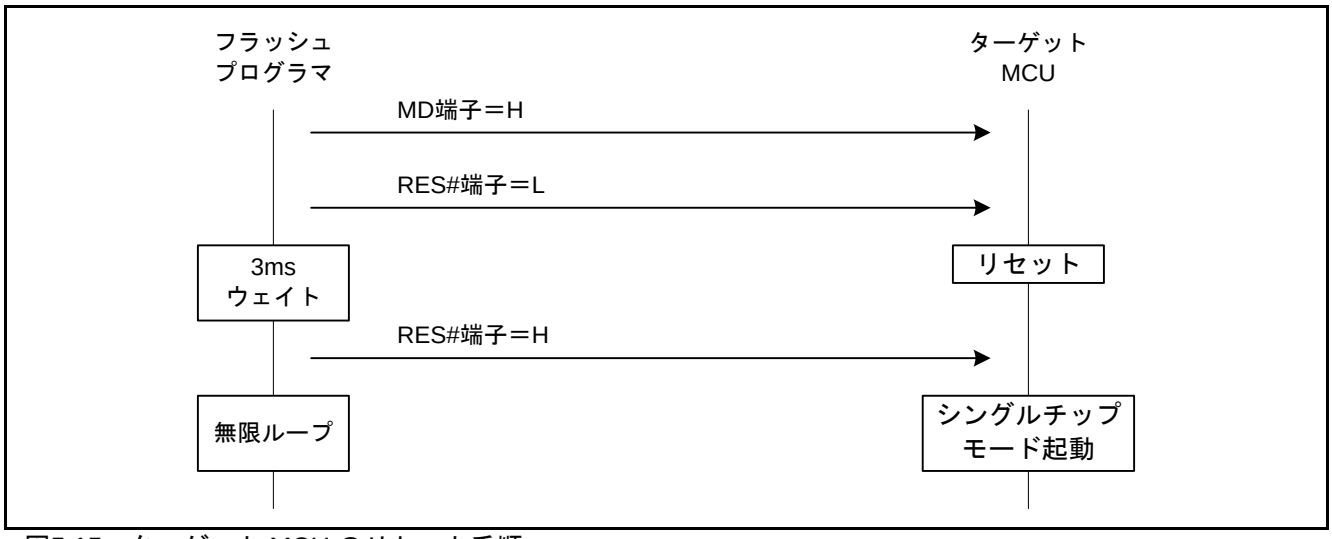


図5.15　ターゲット MCU のリセット手順

5.3　ファイル構成

表 5.3にサンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表5.3　サンプルコードで使用するファイル

ファイル名	概要
main.c	メイン処理、コマンド送信、レスポンス受信処理
cmt_wait.c	CMT を使用した時間待ち処理
cmt_wait.h	cmt_wait.c のヘッダファイル

表5.4　標準インクルードファイル

参照元	内容
stdint.h	指定した幅の整数型を宣言してマクロを定義します。
stdbool.h	論理型、および論理値に関するマクロを定義します。
machine.h	RX ファミリ用 組み込み関数の形式を定義します。
string.h	文字列の比較、複写等を行うライブラリです。

表5.5　参照する関連アプリケーションノートの関数と設定値(RX63N グループ、RX631 グループ　初期設定例)

ファイル名	関数	設定値
r_init_stop_module.c	R_INIT_StopModule()	-
r_init_stop_module.h	-	DMT/DTC、EXDMAC の停止指定
r_init_non_existent_port.c	R_INIT_NonExistentPort()	-
r_init_non_existent_port.h	-	176 ピン版の指定
r_init_clock.c	R_INIT_Clock()	-
r_init_clock.h	-	システムクロックを PLL とし、サブクロックを使用しない指定

表5.6　参照する関連アプリケーションノートの関数と設定値(RX63N High-performance Embedded Workshop 用ルネサススタータキットのサンプルコード)

ファイル名	関数	設定値
lcd.c	Init_LCD() Display_LCD()	-
lcd.h	-	-
rskrx63ndef.h	-	-

5.4　オプション設定メモリ

表 5.7にサンプルコードで使用するオプション設定メモリの状態を示します。必要に応じて、お客様のシステムに最適な値を設定してください。

表5.7　サンプルコードで使用するオプション設定メモリ

シンボル	アドレス	設定値	内容
OFS0	FFFF FF8Fh～FFFF FF8Ch	FFFF FFFFh	リセット後、IWDT は停止 リセット後、WDT は停止
OFS1	FFFF FF8Bh～FFFF FF88h	FFFF FFFFh	リセット後、電圧監視 0 リセット無効 リセット後、HOCO 発振が無効
MDES	FFFF FF83h～FFFF FF80h	FFFF FFFFh	リトルエンディアン

5.5　定数一覧

表 5.8、表 5.9、表 5.10、表 5.11、表 5.12、表 5.13にサンプルコードで使用する定数を示します。

表5.8　サンプルコードで使用する定数

定数名	設定値	内容
ROMVOL_128KB	(128 * 1024)	ターゲット MCU のユーザ領域容量(128K バイト)選択値
ROMVOL_256KB	(256 * 1024)	ターゲット MCU のユーザ領域容量(256K バイト)選択値
ROMVOL_512KB	(512 * 1024)	ターゲット MCU のユーザ領域容量(512K バイト)選択値
TARGET_ROMVOL	ROMVOL_128KB	ターゲット MCU のユーザ領域容量 (128K バイト選択)
TARGET_DATA_ADD	0xFFF40000	ターゲット MCU のユーザ領域に書き込むデータを格納する先頭アドレス
READING_HEAD_ADD	WRITING_HEAD_ADD	ターゲット MCU の読み出し開始アドレス (書き込み開始アドレスと同一)
MDES_ADD	0xFFFFFF80	MDES 判定アドレス
WRITING_TIME	(TARGET_ROMVOL / 256)	ターゲット MCU の書き込み回数 (256 バイト単位に書き込む回数)
READING_TIME	WRITING_TIME	ターゲット MCU の読み出し回数 (書き込み回数と同一)
RES_BUF_SIZE	(262)	受信データ格納バッファサイズ
OK	(0)	真値
NG	(1)	偽値
ERRLOOP_ON	(1)	受信時にエラー検出した場合、エラー処置(無限ループ)する選択値
ERRLOOP_OFF	(0)	受信時にエラー検出した場合、エラー処置(無限ループ)しない選択値
INTERVAL_ON	(1)	送信時に間隔を設ける選択値
INTERVAL_OFF	(0)	送信時に間隔を設けない選択値
RES_ACK_NORMAL	(0x06)	通常 ACK 判定値
RES_ACK_ID	(0x16)	ID コードプロテクト有効 ACK 判定値
RES_ACK_BERS_EXSPC	(0x46)	ブロックイレーズ拡張仕様 ACK 判定値
RES_ACK_MERSMD	(0x56)	イレーズレディ処置 ACK 判定値
ARRAY_SIZE_OF(a)	(sizeof(a) / sizeof(a[0]))	コマンド送信データのバイト数取得マクロ関数
WT_BASE_US	(1000000L)	1μs 単位の待ち時間計算値
WT_BASE_MS	(1000L)	1ms 単位の待ち時間計算値
WT_CMT_CLOCK	(48L * WT_BASE_US)	CMT カウントソース周波数 (PCLKB:48MHz)
WT_CMT_DIVIDE	(512L)	CMT カウントソースの分周比
WAIT_52US	((52. * (WT_CMT_CLOCK / WT_CMT_DIVIDE)) / WT_BASE_US +0.5)	CMT による待ち時間 (52μs)
WAIT_1MS	((　1. * (WT_CMT_CLOCK / WT_CMT_DIVIDE)) / WT_BASE_MS +0.5)	CMT による待ち時間 (1ms)

定数名	設定値	内容
WAIT_3MS	((　3. * (WT_CMT_CLOCK / WT_CMT_DIVIDE)) / WT_BASE_MS +0.5)	CMT による待ち時間 (3ms)
WAIT_100MS	((100. * (WT_CMT_CLOCK / WT_CMT_DIVIDE)) / WT_BASE_MS +0.5)	CMT による待ち時間 (100ms)
WAIT_400MS	((400. * (WT_CMT_CLOCK / WT_CMT_DIVIDE)) / WT_BASE_MS +0.5)	CMT による待ち時間 (400ms)

表5.9　サンプルコードで使用する定数(TARGET_ROMVOL に ROMVOL_128KB を選択した場合)

定数名	設定値	内容
TARGET_ID1_ADD	0xFFF5FFA0	ターゲット MCU に書き込む 制御コードと ID コード 1 から ID コード 3 の参照アドレス
TARGET_ID2_ADD	0xFFF5FFA4	ターゲット MCU に書き込む ID コード 4 から ID コード 7 の参照アドレス
TARGET_ID3_ADD	0xFFF5FFA8	ターゲット MCU に書き込む ID コード 8 から ID コード 11 の参照アドレス
TARGET_ID4_ADD	0xFFF5FFAC	ターゲット MCU に書き込む ID コード 12 から ID コード 15 の参照アドレス
WRITING_HEAD_ADD	0xFFFE0000	ターゲット MCU の書き込み開始アドレス

表5.10　サンプルコードで使用する定数(TARGET_ROMVOL に ROMVOL_256B を選択した場合)

定数名	設定値	内容
TARGET_ID1_ADD	0xFFF7FFA0	ターゲット MCU に書き込む 制御コードと ID コード 1 から ID コード 3 の参照アドレス
TARGET_ID2_ADD	0xFFF7FFA4	ターゲット MCU に書き込む ID コード 4 から ID コード 7 の参照アドレス
TARGET_ID3_ADD	0xFFF7FFA8	ターゲット MCU に書き込む ID コード 8 から ID コード 11 の参照アドレス
TARGET_ID4_ADD	0xFFF7FFAC	ターゲット MCU に書き込む ID コード 12 から ID コード 15 の参照アドレス
WRITING_HEAD_ADD	0xFFFC0000	ターゲット MCU の書き込み開始アドレス

表5.11　サンプルコードで使用する定数(TARGET_ROMVOL に ROMVOL_512B を選択した場合)

定数名	設定値	内容
TARGET_ID1_ADD	0xFFFBFFA0	ターゲット MCU に書き込む 制御コードと ID コード 1 から ID コード 3 の参照アドレス
TARGET_ID2_ADD	0xFFFBFFA4	ターゲット MCU に書き込む ID コード 4 から ID コード 7 の参照アドレス
TARGET_ID3_ADD	0xFFFBFFA8	ターゲット MCU に書き込む ID コード 8 から ID コード 11 の参照アドレス
TARGET_ID4_ADD	0xFFFBFFAC	ターゲット MCU に書き込む ID コード 12 から ID コード 15 の参照アドレス
WRITING_HEAD_ADD	0xFFF80000	ターゲット MCU の書き込み開始アドレス

表5.12　サンプルコードで使用する定数(ブートモードエントリに使用する定義)

定数名	設定値	内容
BTMD_PMR	(PORTE.PMR.BYTE)	ターゲット MCU の UB#端子、MD 端子、RES#端子への出力端子割り付け(ポートモードレジスタ)
BTMD_PODR	(PORTE.PODR.BYTE)	ターゲット MCU の UB#端子、MD 端子、RES#端子への出力端子割り付け(ポート出力データレジスタ
BTMD_PDR	(PORTE.PDR.BYTE)	ターゲット MCU の UB#端子、MD 端子、RES#端子への出力端子割り付け(ポート方向レジスタ)
UB_PIN	(PORTE.PODR.BIT.B2)	ターゲット MCU の UB#端子への出力割り当て
MD_PIN	(PORTE.PODR.BIT.B1)	ターゲット MCU の MD 端子への出力割り当て
RES_PIN	(PORTE.PODR.BIT.B0)	ターゲット MCU の RES#端子への出力割り当て
BTMD_PDR_INIT	(0x07)	ターゲット MCU の UB#端子、MD 端子、RES#端子出力初期設定値
BTMD_PODR_INIT	(0x04)	ターゲット MCU の UB#端子 High レベル出力初期設定値

表5.13　サンプルコードで使用する定数(調歩同期式シリアルの定義)

定数名	設定値	内容
SCIn	SCI0	SCI チャネル: SCI0
MSTP_SCIn	MSTP(SCI0)	SCI0 モジュールストップ設定ビット
IR_SCIn_RXIn	IR(SCI0,RXI0)	SCI0.RXI0 割り込みステータスフラグ
IR_SCIn_TXIn	IR(SCI0,TXI0)	SCI0.TXI0 割り込みステータスフラグ
RXDn_PDR	(PORT3.PDR.BIT.B3)	SCI0.RXI0 端子方向制御ビット
RXDn_PMR	(PORT3.PMR.BIT.B3)	SCI0.RXI0 端子モード制御ビット
RXDnPFS	P33PFS	SCI0.RXI0 端子機能制御レジスタ
RXDnPFS_SELECT	(0x0B)	RXD0 端子機能選択ビット設定値
TXDn_PODR	(PORT3.PODR.BIT.B2)	SCI0.TXI0 端子出力データ格納ビット
TXDn_PDR	(PORT3.PDR.BIT.B2)	SCI0.TXI0 端子方向制御ビット
TXDn_PMR	(PORT3.PMR.BIT.B2)	SCI0.TXI0 端子モード制御ビット
TXDnPFS	P32PFS	SCI0.TXI0 端子機能制御レジスタ
TXDnPFS_SELECT	(0x0B)	TXD0 端子機能選択ビット設定値
SSR_ERROR_FLAGS	(0x38)	SCI.SSR レジスタのエラーフラグのビットパターン
BRR_SET(bps)	(WT_CMT_CLOCK/(32*(0.5)*(bps))-1+0.5)	SCI.BRR レジスタの設定値計算マクロ関数

5.6　構造体/共用体一覧

図 5.16にサンプルコードで使用する構造体/共用体を示します。

```
typedef struct BOOT_CMD_s
{
    uint32_t    TrnSize;        /* expected value of the transmit size of command */
    uint32_t    RecSize;        /* expected value of the receive size of response */
    uint8_t     ACKRes;         /* ACK value of response */
    uint8_t     *Command;       /* boot command sequence data pointer */
} BOOT_CMD_t;
```

図5.16　サンプルコードで使用する構造体/共用体

5.7　変数一覧

表 5.14にグローバル変数を、表 5.15にstatic 型変数を示します。

表5.14　グローバル変数

型	変数名	内容	使用関数
volatile uint8_t	CMT_InterruptFlag	待ち時間有効フラグ	CMT_WaitSet CMT_Wait Excep_CMT0_CMI0 ReceiveResponse

表5.15　static 型変数

型	変数名	内容	使用関数
uint8_t	ResponseBuffer[RES_BUF_SIZE]	受信データ格納バッファ	main ReceiveResponse
uint8_t	TransferMode	送信モードフラグ	main TransferCommand
uint8_t	ReceiveMode	受信モードフラグ	main ReceiveResponse
uint8_t	IDProtectMode	ID コードプロテクト状態判定バッファ	main
uint32_t	BufferIndex	受信データ格納バッファインデックス	ReceiveResponse
uint32_t	DeviceCode	デバイスコード格納バッファ	main
uint32_t	BlockInfoData[6]	ブロック情報格納バッファ	main
uint8_t	CMD_BitRateAdjustment_1st[]	ビットレート自動調整コマンドデータ	-
uint8_t	CMD_BitRateAdjustment_2nd[]	ビットレート自動調整確認コマンドデータ	-
uint8_t	CMD_EnquiryDevice[]	サポートデバイス問い合わせコマンドデータ	-
uint8_t	CMD_SelectDevice[]	デバイス選択コマンドデータ	-
uint8_t	CMD_BlockInfo[]	ブロック情報問い合わせコマンドデータ	-
uint8_t	CMD_OperatingFreqSel_1st[]	動作周波数選択コマンドデータ	-
uint8_t	CMD_OperatingFreqSel_2nd[]	動作周波数選択確認コマンドデータ	-
uint8_t	CMD_PEstatusTransition[]	プログラム/ イレーズステータス遷移コマンドデータ	-
uint8_t	CMD_IDCodeCheck[]	ID コードチェックコマンドデータ	-
uint8_t	CMD_ErasePreparation[]	イレーズ準備コマンドデータ	-
uint8_t	CMD_BlockErase[]	ブロックイレーズ(拡張仕様)コマンドデータ	-
uint8_t	CMD_BootModeStatusInquiry[]	ブートモードステータス問い合わせコマンドデータ	-

型	変数名	内容	使用関数
uint8_t	CMD_ProgramPreparation[]	ユーザ/データ領域プログラム準備コマンドデータ	-
uint8_t	CMD_Program[]	プログラムコマンドデータ	-
uint8_t	CMD_ProgramTermination[]	プログラム終了コマンドデータ	-
uint8_t	CMD_MemoryRead[]	メモリリードコマンドデータ	-
BOOT_CMD_t	BitRateAdjustment_1st	ビットレート自動調整コマンド構造体	main
BOOT_CMD_t	BitRateAdjustment_2nd	ビットレート自動調整確認コマンド構造体	main
BOOT_CMD_t	EnquiryDevice	サポートデバイス問い合わせコマンド構造体	main
BOOT_CMD_t	SelectDevice	デバイス選択コマンド構造体	main
BOOT_CMD_t	BlockInfo	ブロック情報問い合わせコマンド構造体	main
BOOT_CMD_t	OperatingFreqSel_1st	動作周波数選択コマンド構造体	main
BOOT_CMD_t	OperatingFreqSel_2nd	動作周波数選択確認コマンド構造体	main
BOOT_CMD_t	PEstatusTransition	プログラム/ イレーズステータス遷移コマンド構造体	main
BOOT_CMD_t	IDCodeCheck	ID コードチェックコマンド構造体	main
BOOT_CMD_t	ErasePreparation	イレーズ準備コマンド構造体	main
BOOT_CMD_t	BlockErase	ブロックイレーズ(拡張仕様)コマンド構造体	main
BOOT_CMD_t	BootModeStatusInquiry	ブートモードステータス問い合わせコマンド構造体	main
BOOT_CMD_t	ProgramPreparation	ユーザ/データ領域プログラム準備コマンド構造体	main
BOOT_CMD_t	Program	プログラムコマンド構造体	main
BOOT_CMD_t	ProgramTermination	プログラム終了コマンド構造体	main
BOOT_CMD_t	MemoryRead	メモリリードコマンド構造体	main

5.8　関数一覧

表 5.16に関数を示します。

表5.16　関数

関数名	概要
main	メイン処理、通信プロトコル制御
peripheral_init	周辺機能初期設定
CMT_WaitInit	CMT による待ち時間用タイマ初期設定
CMT_WaitSet	CMT による待ち時間設定
CMT_Wait	CMT による時間待ち処理
Excep_CMT0_CMI0	CMT0,CMI0 割り込み処理
SCI_Init	SCI 初期設定
SCI_change	SCI ビットレート変更処理
CalcSumData	“SUM”データ計算処理
BootModeEntry	ターゲット MCU のブートモード起動処理
BootModeRelease	ターゲット MCU のリセット処理
TransferCommand	コマンド送信処理
ReceiveResponse	レスポンス受信処理
U4memcpy	符号なし 4 バイトデータの複写

5.9　関数仕様

サンプルコードの関数仕様を示します。

main	
概　要	メイン処理
ヘッダ	lcd.h, cmt_wait.h
宣　言	void main(void)
説　明	初期設定後、ターゲット MCU をブートモード(SCI インタフェース)起動して、ユーザ領域を書き換えます。
引　数	なし
リターン値	なし

peripheral_init	
概　要	周辺機能初期設定
ヘッダ	lcd.h, cmt_wait.h
宣　言	void peripheral_init(void)
説　明	使用する周辺機能の初期設定を行います。
引　数	なし
リターン値	なし

CMT_WaitInit	
概　要	CMT による待ち時間用タイマ初期設定
ヘッダ	cmt_wait.h
宣　言	void CMT_WaitInit(void)
説　明	待ち時間用タイマ(CMT0)の初期設定を行います。
引　数	なし
リターン値	なし

CMT_WaitSet	
概　要	CMT による待ち時間設定
ヘッダ	cmt_wait.h
宣　言	void CMT_WaitSet(uint16_t cnt)
説　明	引数で指定した時間(μs)を CMCOR レジスタに設定し、CMCNT レジスタをカウント開始します。
引　数	uint16_t cnt　　待ち時間
リターン値	なし
備　考	待ち時間の最小単位は 1 / (PCLKB[MHz] / 512) ≒ 10.67us

CMT_Wait	
概　要	CMT による時間待ち処理
ヘッダ	cmt_wait.h
宣　言	void CMT_Wait(uint16_t cnt)
説　明	引数で指定した時間(μs)待ちます。
引　数	uint16_t cnt　　待ち時間
リターン値	なし
備　考	待ち時間の最小単位は 1 / (PCLKB[MHz] / 512) ≒ 10.67us

Excep_CMT0_CMI0	
概　要	CMT0,CMI0 割り込み処理
ヘッダ	cmt_wait.h
宣　言	void Excep_CMT0_CMI0(void)
説　明	CMT0.CMCNT と CMT0.CMCOR のコンペアマッチ割り込み処理を行います。
引　数	なし
リターン値	なし

SCI_Init	
概　要	SCI 初期設定
ヘッダ	なし
宣　言	void SCI_Init(void)
説　明	SCI の初期設定を行います。
引　数	なし
リターン値	なし

SCI_change	
概　要	SCI ビットレート変更処理
ヘッダ	なし
宣　言	void SCI_change(void)
説　明	SCI のビットレートを 19200bps から 1Mbps に変更します。
引　数	なし
リターン値	なし

CalcSumData	
概　要	“SUM”データ計算処理
ヘッダ	なし
宣　言	uint8_t CalcSumData(uint8_t *pData, uint32_t Length)
説　明	ブート通信プロトコルの“SUM”データを計算します。
引　数	uint8_t *pData　　“SUM”対象データアドレス
	uint32_t Length　　“SUM”対象データ数
リターン値	“SUM”データ

BootModeEntry	
概　要	ターゲット MCU のブートモード起動処理
ヘッダ	なし
宣　言	void BootModeEntry(void)
説　明	ターゲット MCU をブートモード(SCI インタフェース)で起動するように、MD 端子、UB#端子、RES#端子を制御します。
引　数	なし
リターン値	なし

BootModeRelease		
概　要	ターゲット MCU のリセット処理	
ヘッダ	なし	
宣　言	void BootModeRelease(uint8_t mode)	
説　明	ターゲット MCU をリセットします。	
引　数	uint8_t mode	Debug LCD ライン 2 の出力パターン選択
リターン値	なし	

TransferCommand		
概　要	コマンド送信処理	
ヘッダ	なし	
宣　言	void TransferCommand(BOOT_CMD_t *pCmd)	
説　明	引数で指定されたコマンド構造体のコマンドデータを送信します。	
引　数	BOOT_CMD_t *pCmd	送信するコマンド構造体のアドレス
リターン値	なし	
備　考	TransferMode 変数が INTERVAL_ON の場合、CMT_Wait(WAIT_1MS)を呼び出す。	

ReceiveResponse	
概　要	レスポンス受信処理
ヘッダ	なし
宣　言	uint8_t　ReceiveResponse(BOOT_CMD_t *pCmd)
説　明	引数のコマンド構造体内にあるレスポンスサイズ期待値のバイト数分受信します。
引　数	BOOT_CMD_t *pCmd　　受信するコマンド構造体のアドレス
リターン値	OK：正常に受信完了 NG：タイムアウト(1s)、エラーレスポンスを受信
備　考	ReceiveMode 変数が ERRLOOP_ON かつリターン値が NG の場合、 BootModeRelease(NG)関数を呼び出してこの関数から復帰しない。

U4memcpy			
概　要	符号なし 4 バイトデータの複写		
ヘッダ	なし		
宣　言	void　*U4memcpy(void *pS1, const void *pS2)		
説　明	複写元の記憶域の内容を、4 バイト分、複写先の記憶域に複写します。 データ配置がリトルエンディアンの場合、複写先の 4 バイトデータをバイトリバースします。		
引　数	void　*pS1	複写先の記憶域のアドレス	
	const void *pS2	複写元の記憶域のアドレス	
リターン値	pS1 の値		

5.10　フローチャート

5.10.1　メイン処理、通信プロトコル制御

図 5.17から図 5.30にメイン処理、通信プロトコル制御のフローチャートを示します。

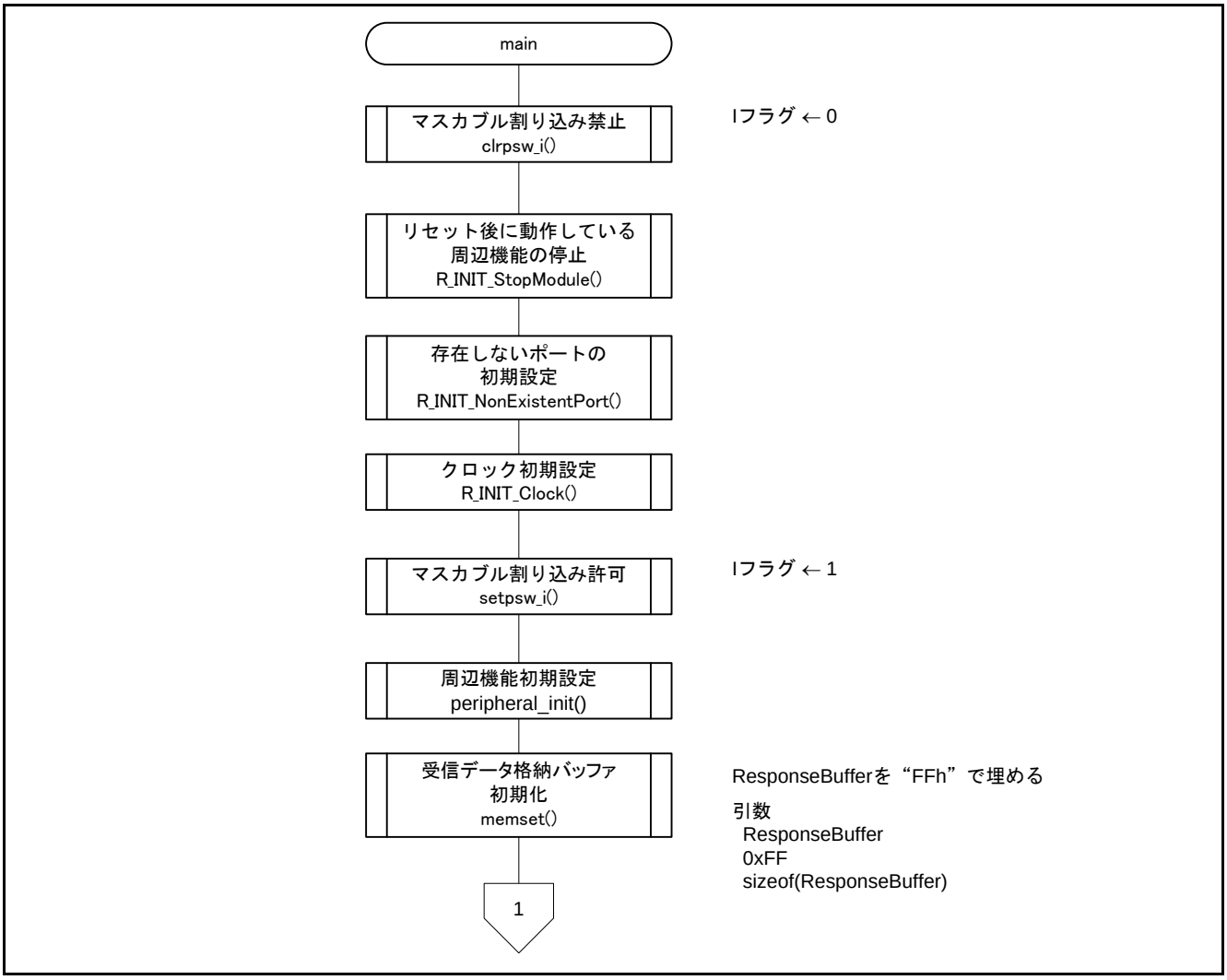


図5.17　メイン処理、通信プロトコル制御

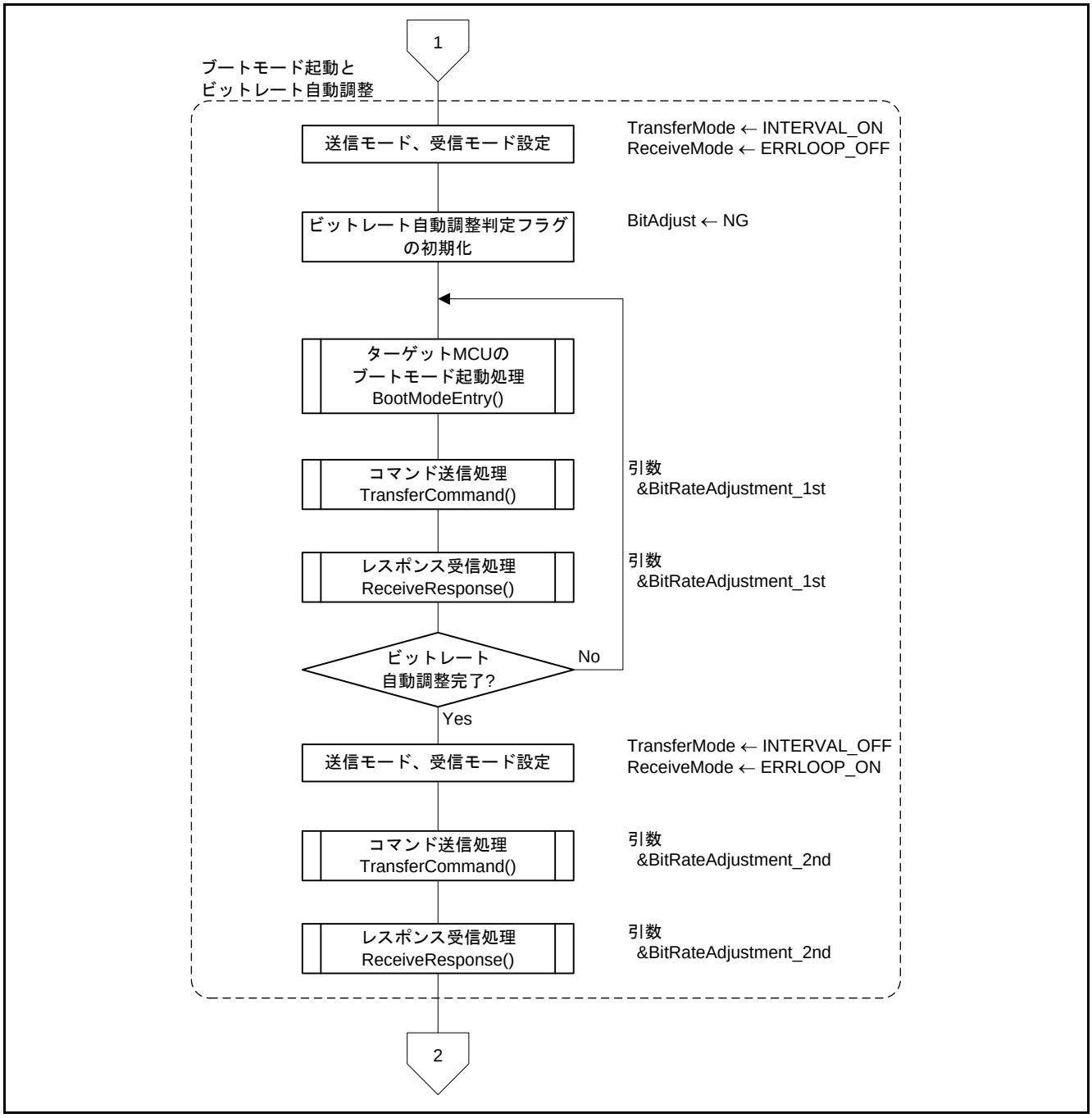


図5.18　メイン処理、通信プロトコル制御

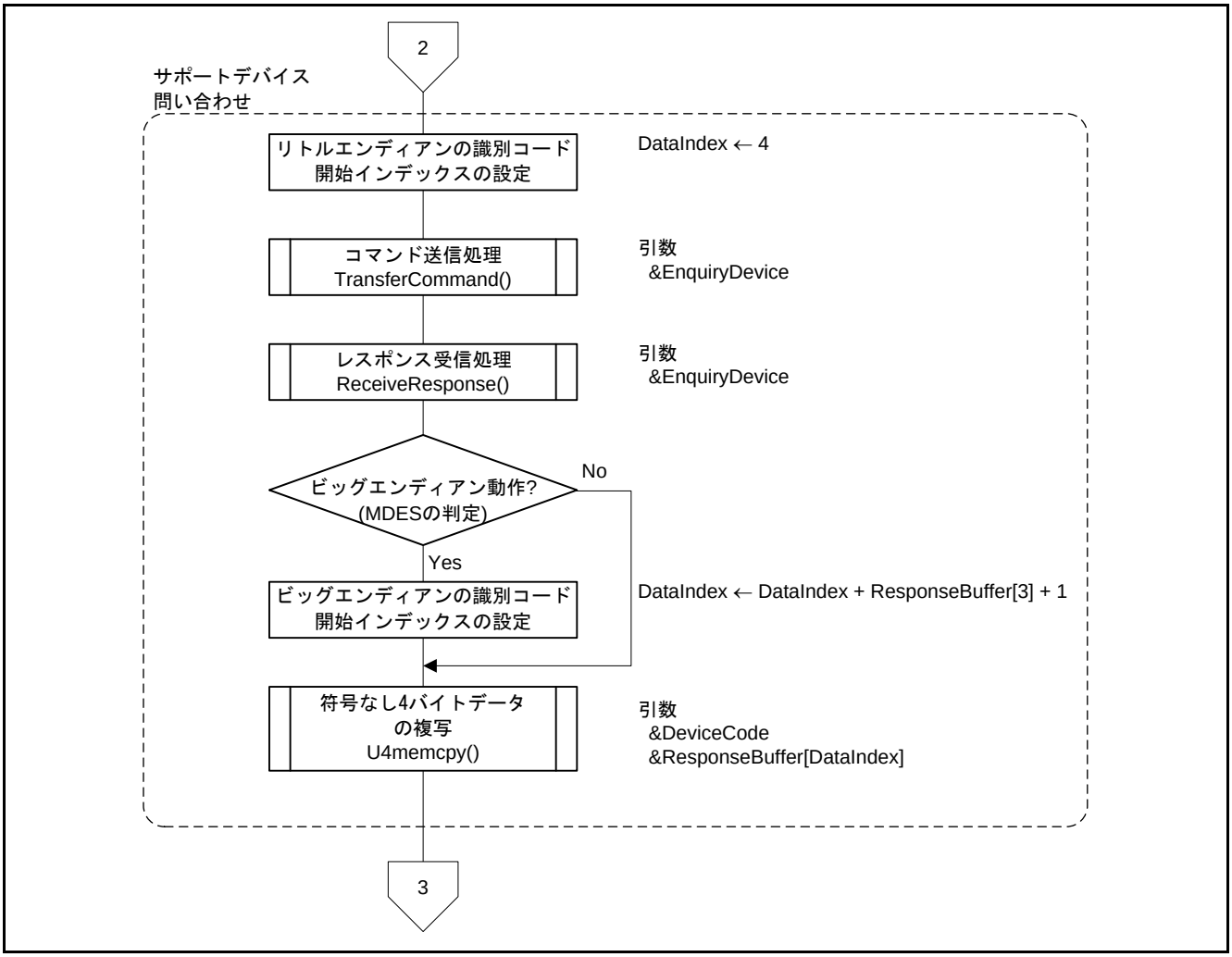


図5.19　メイン処理、通信プロトコル制御

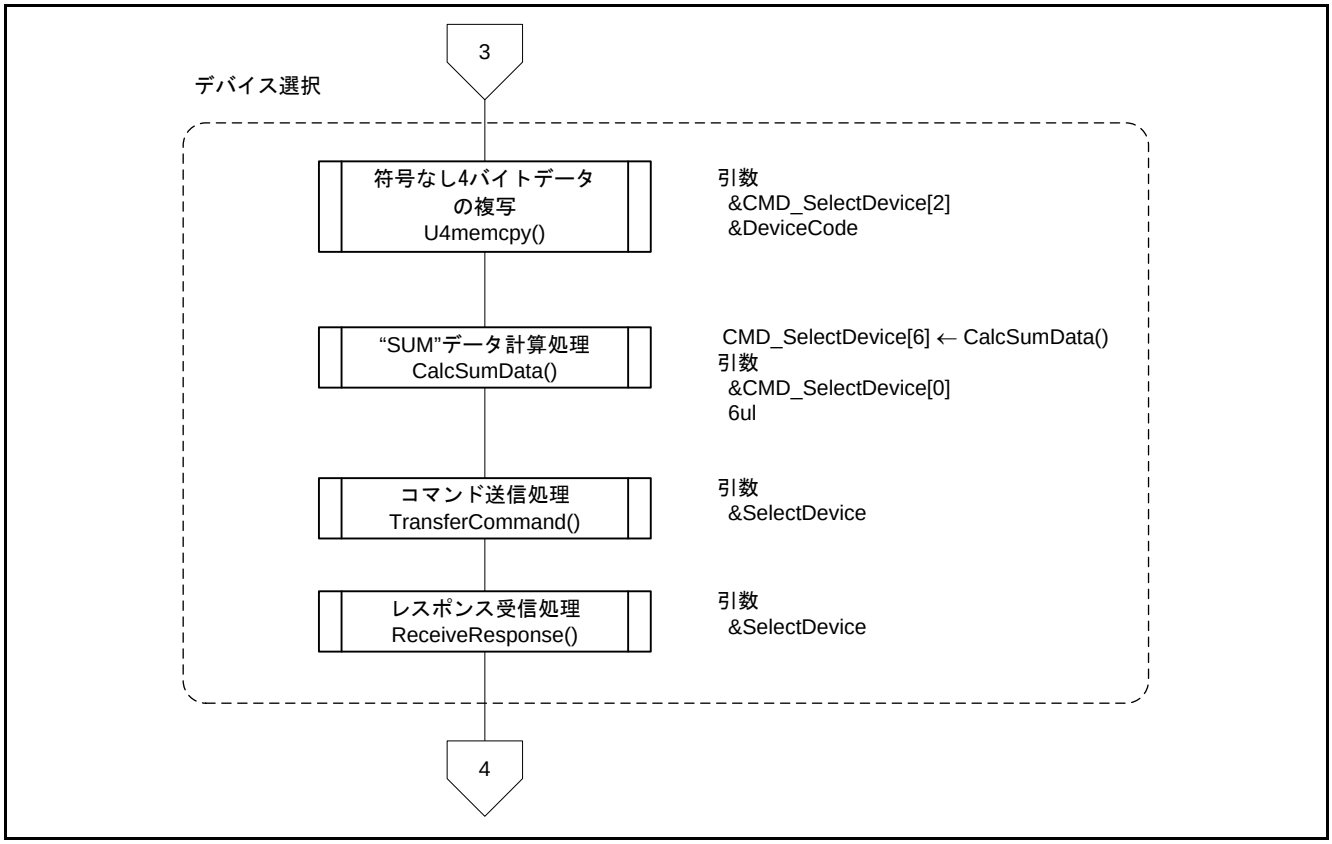


図5.20　メイン処理、通信プロトコル制御

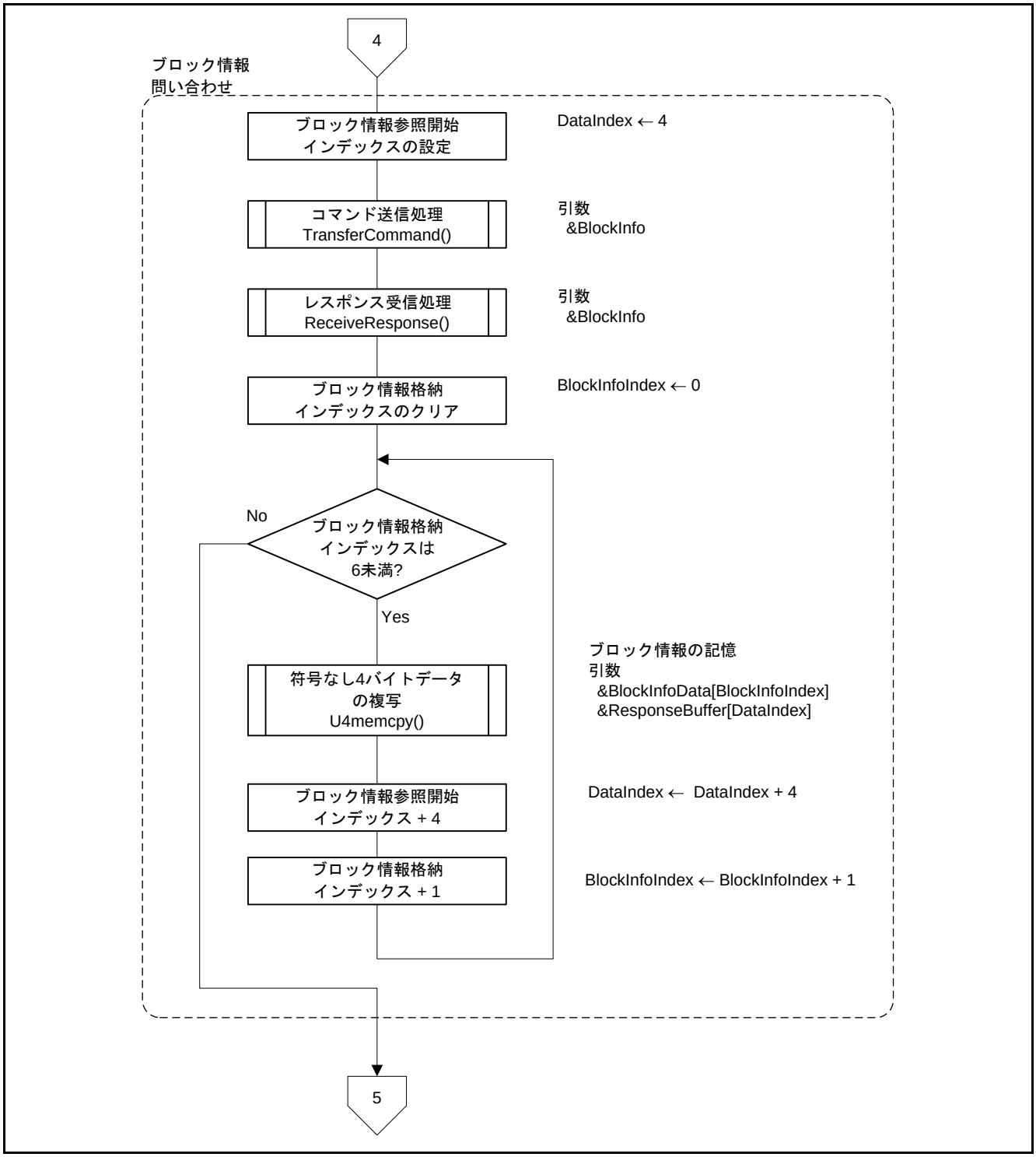


図5.21　メイン処理、通信プロトコル制御

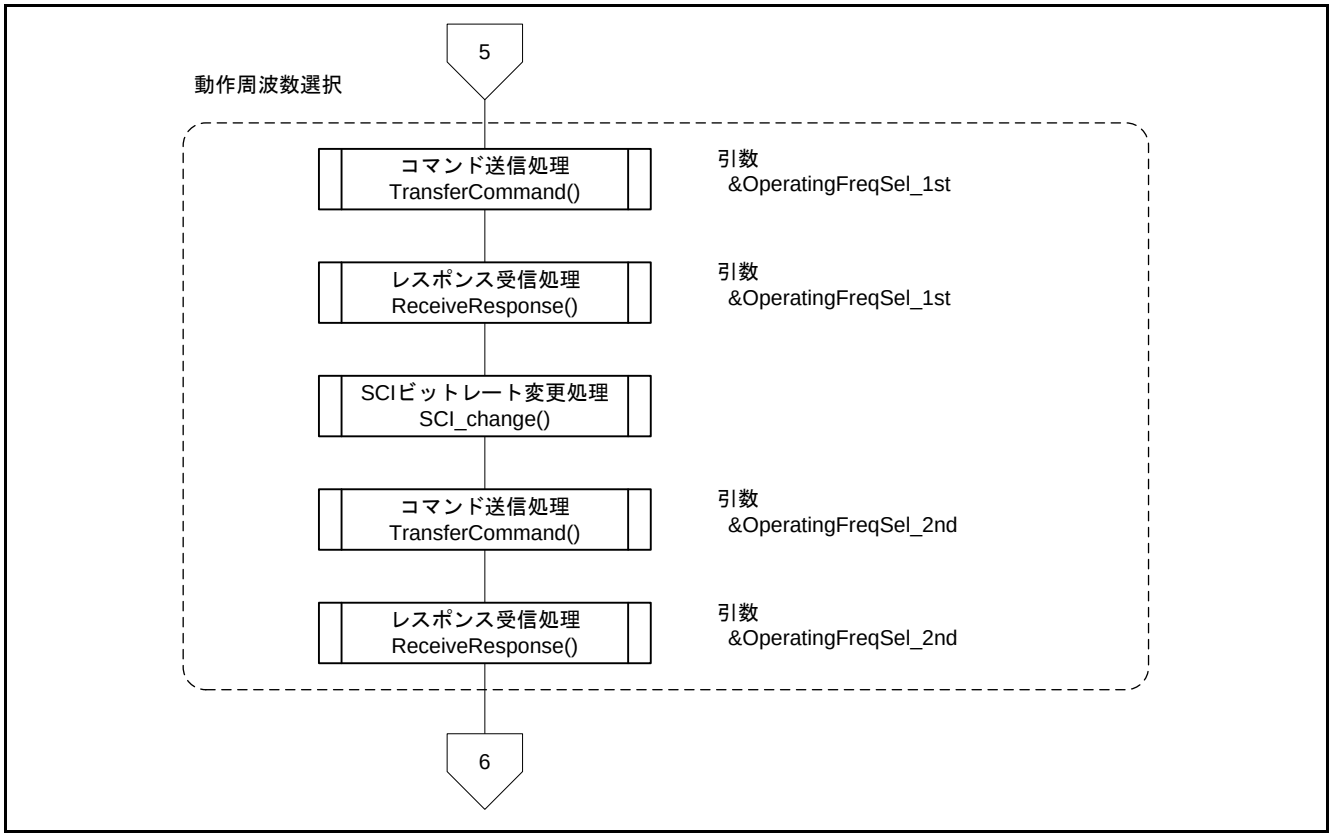
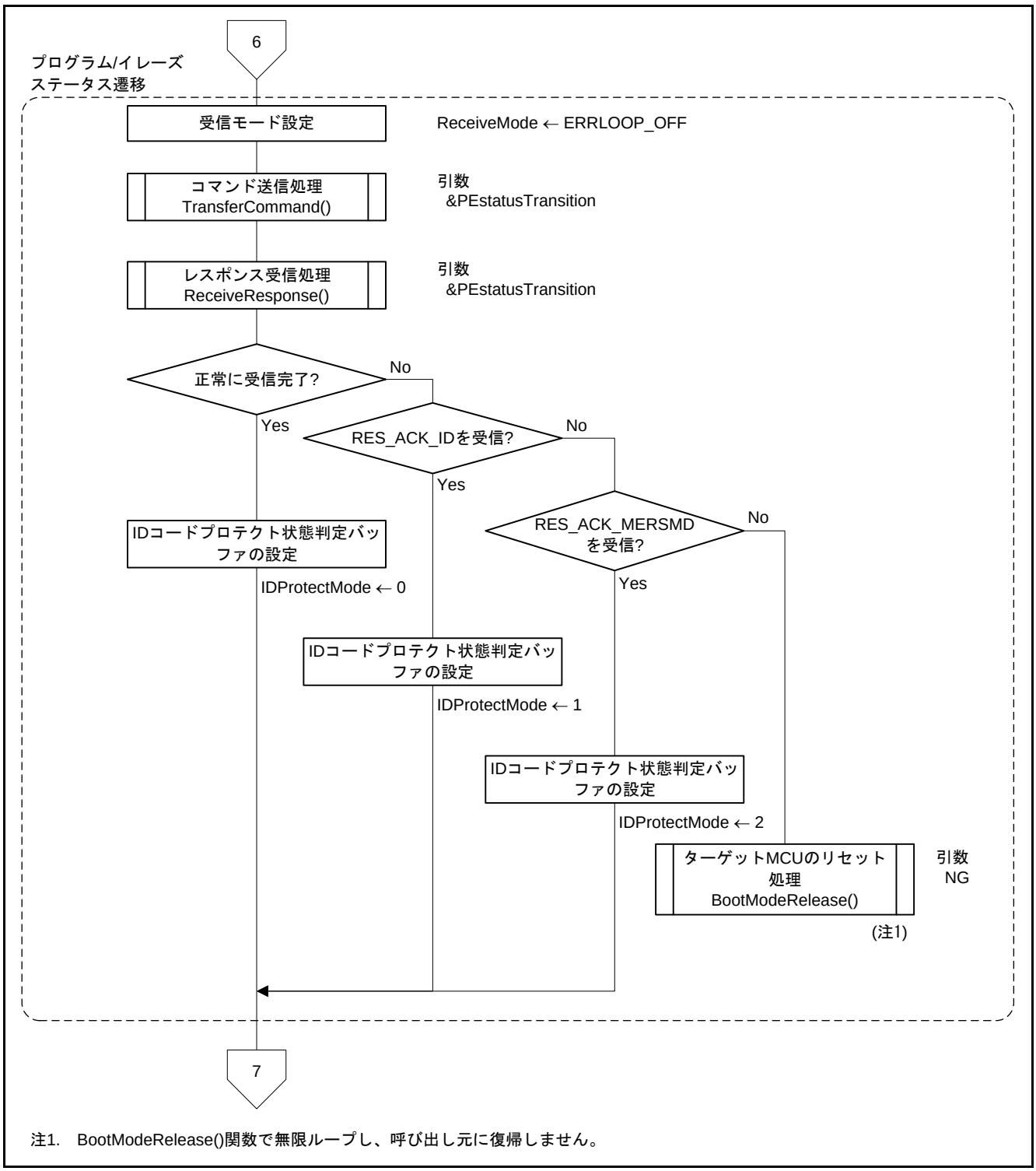
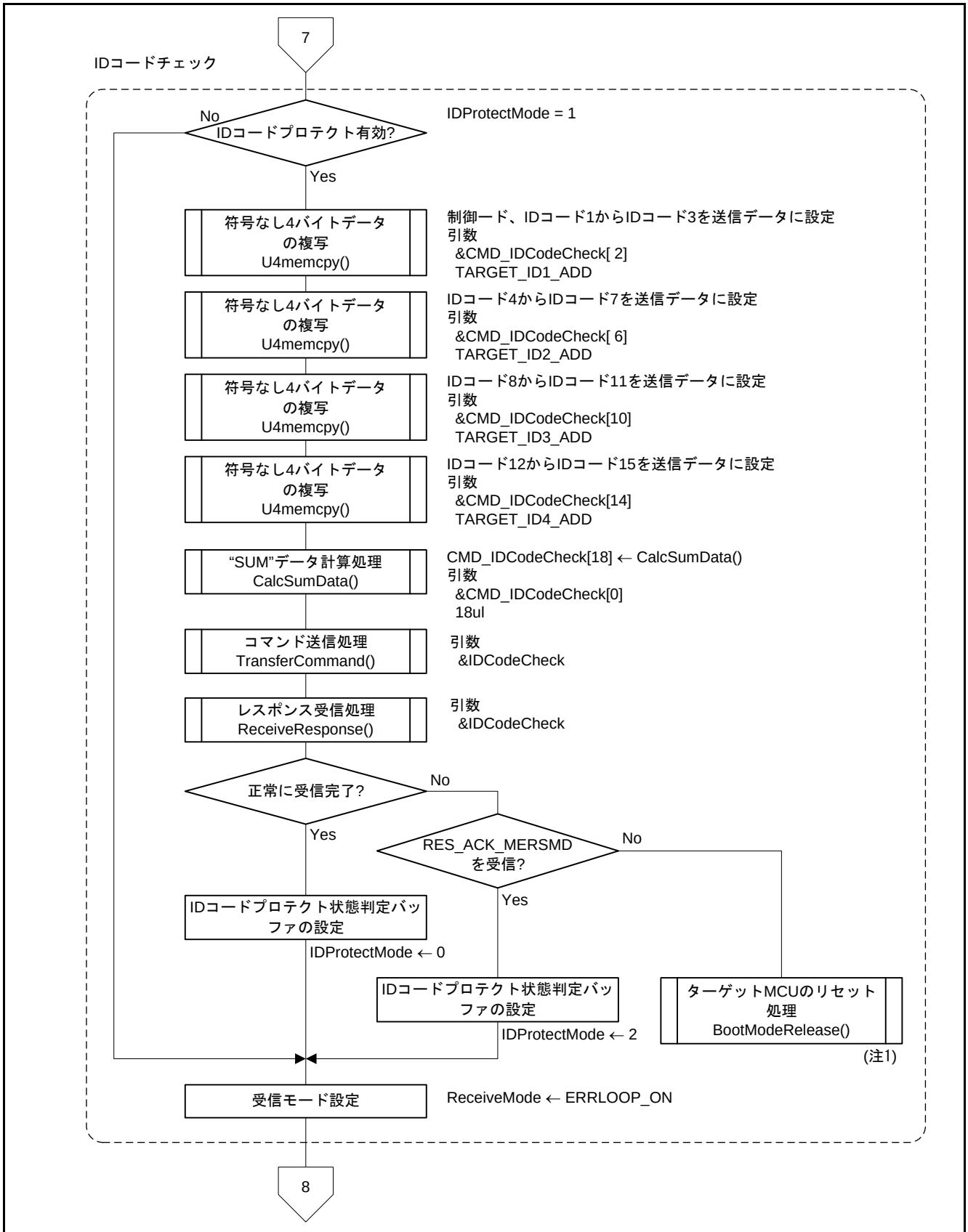


図5.22　メイン処理、通信プロトコル制御



注1. BootModeRelease()関数で無限ループし、呼び出し元に復帰しません。

図5.23 メイン処理、通信プロトコル制御



注1. BootModeRelease()関数で無限ループし、呼び出し元に復帰しません。

図5.24 メイン処理、通信プロトコル制御

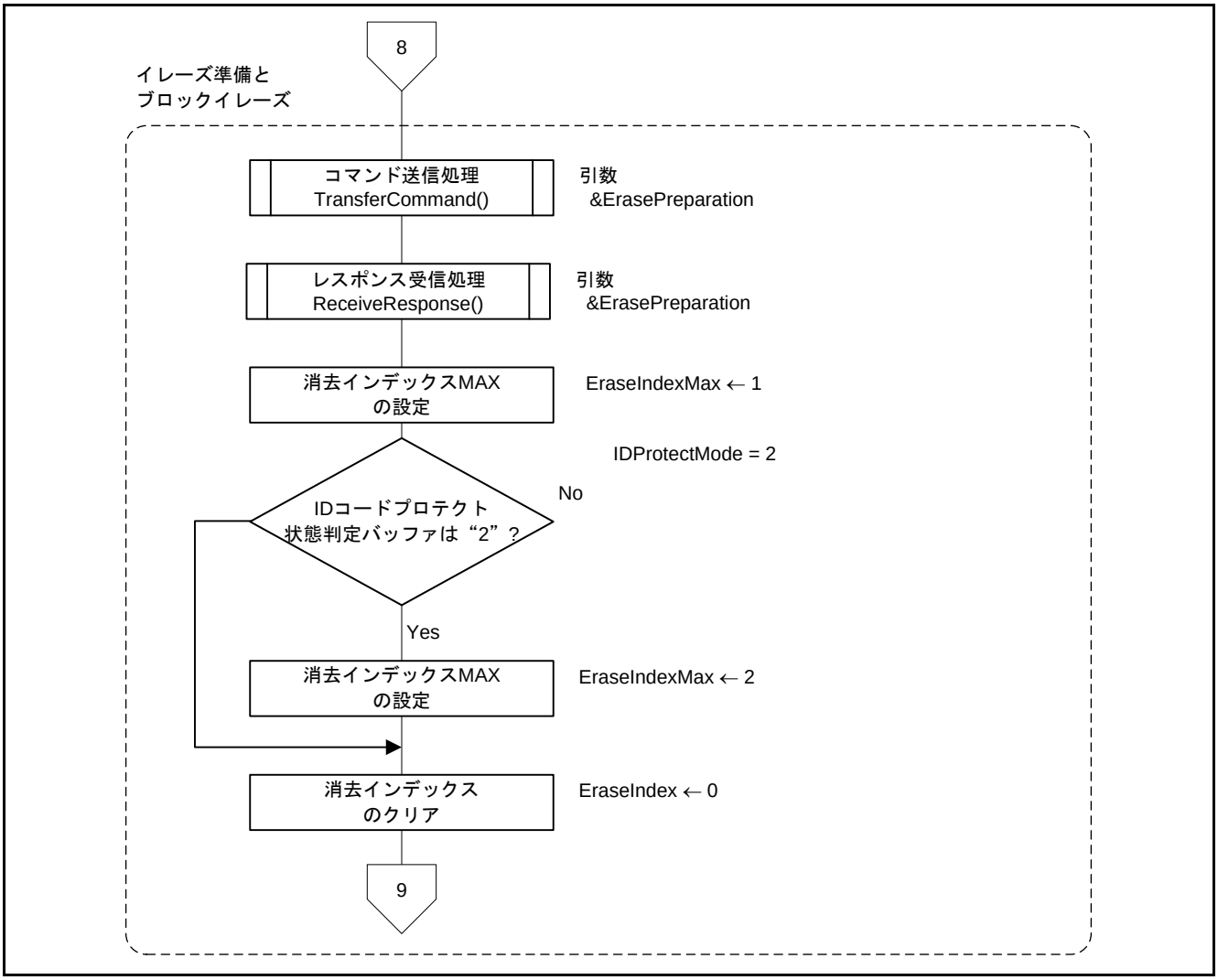


図5.25　メイン処理、通信プロトコル制御

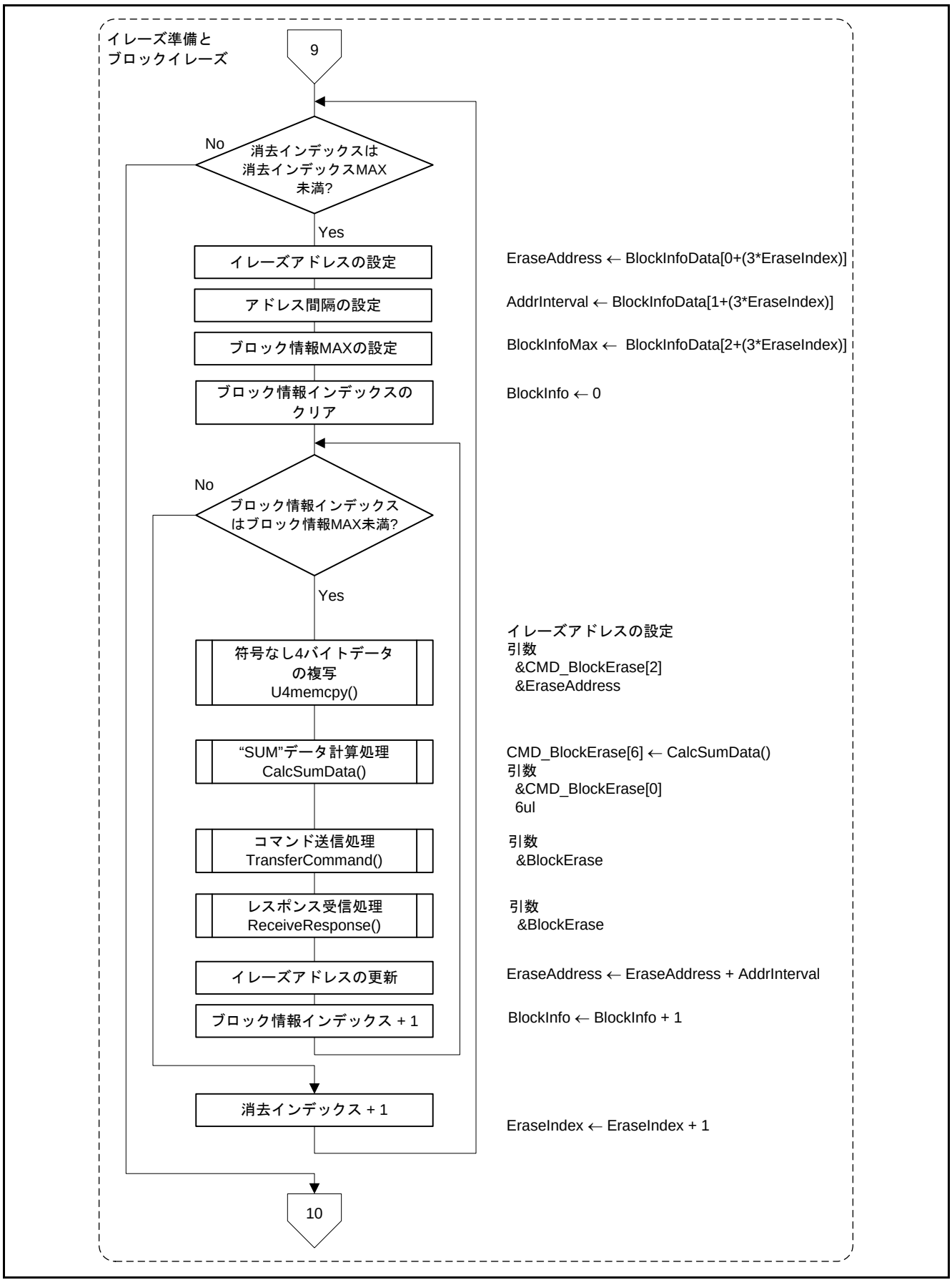


図5.26　メイン処理、通信プロトコル制御

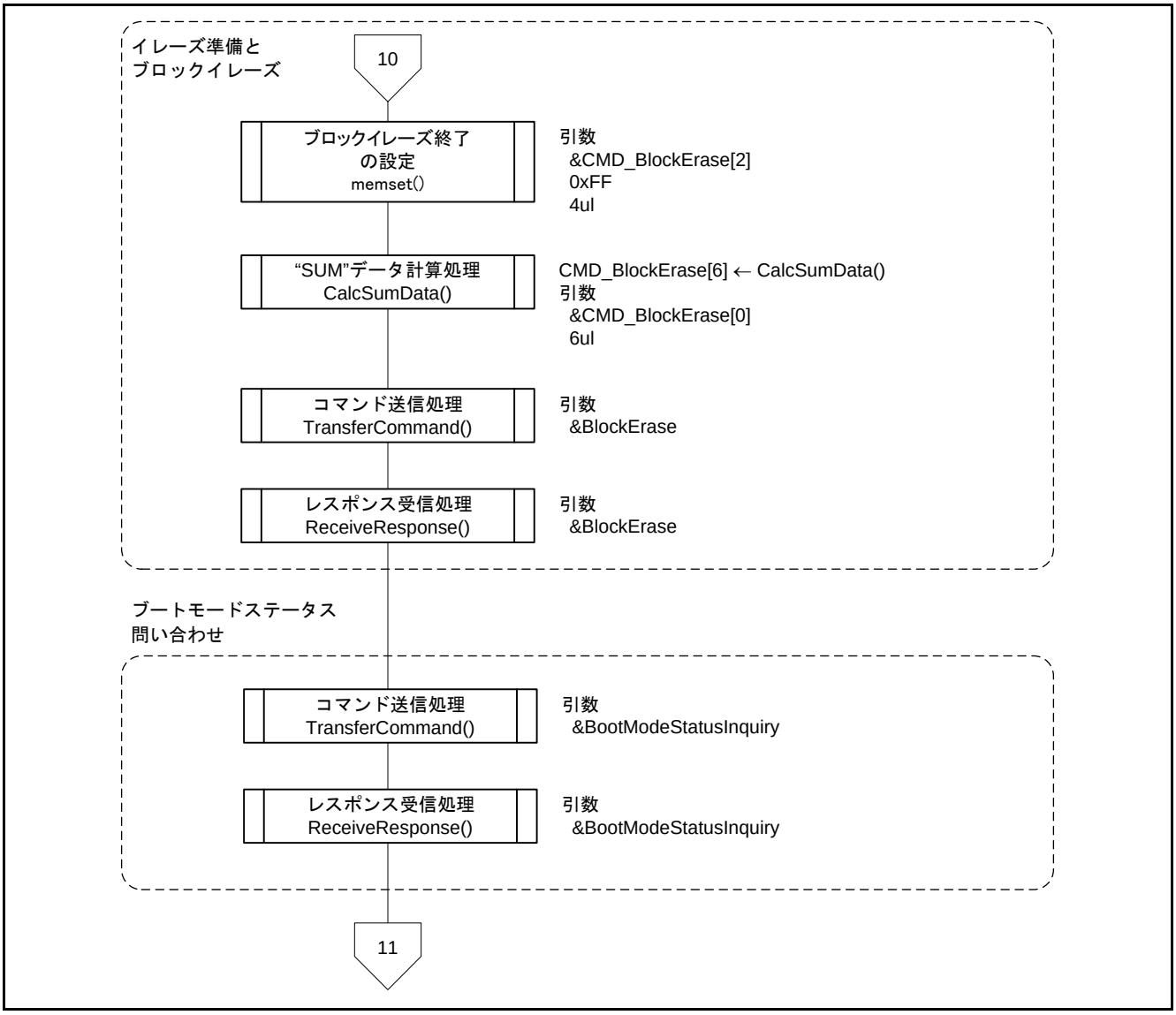


図5.27　メイン処理、通信プロトコル制御

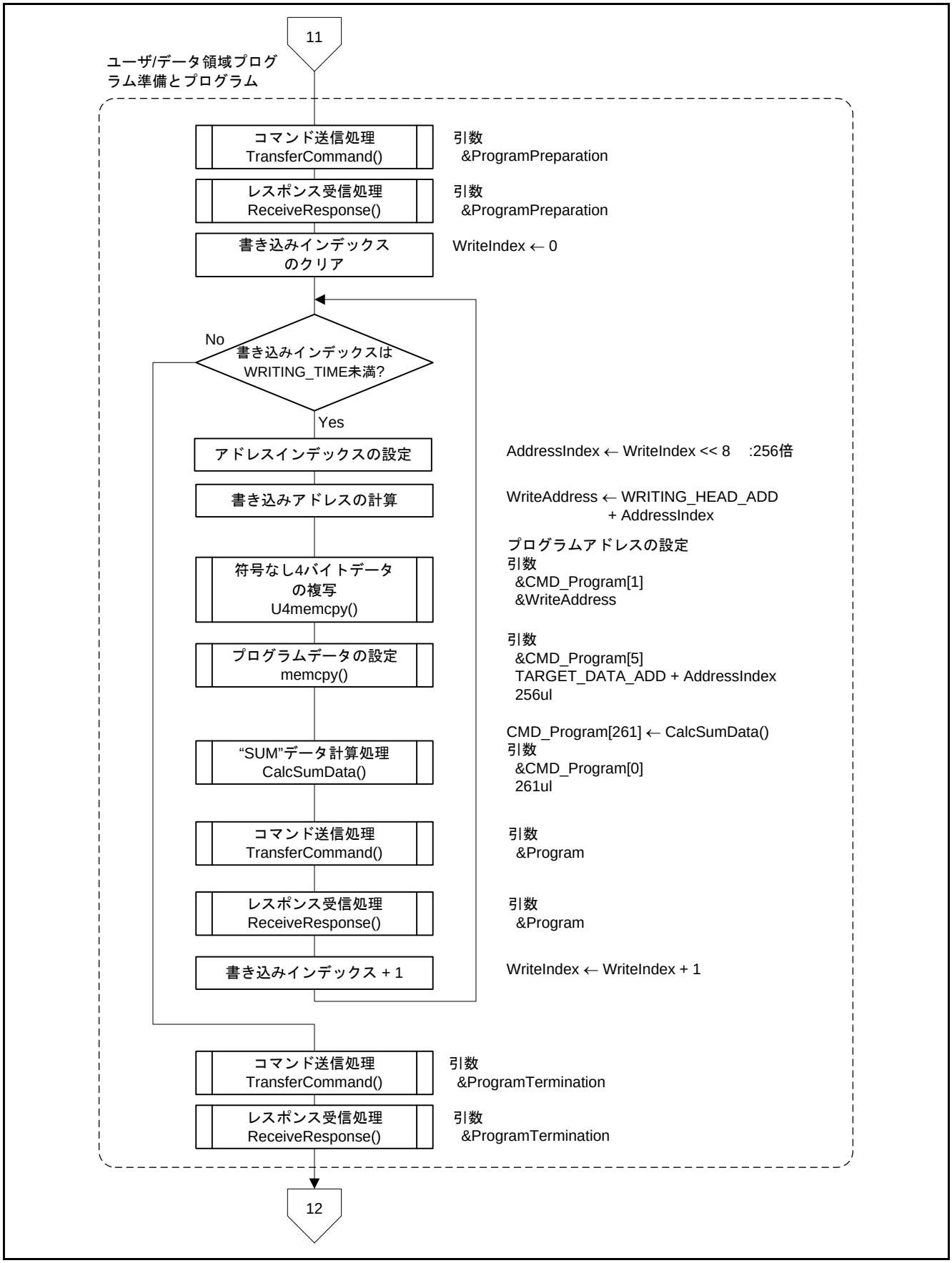
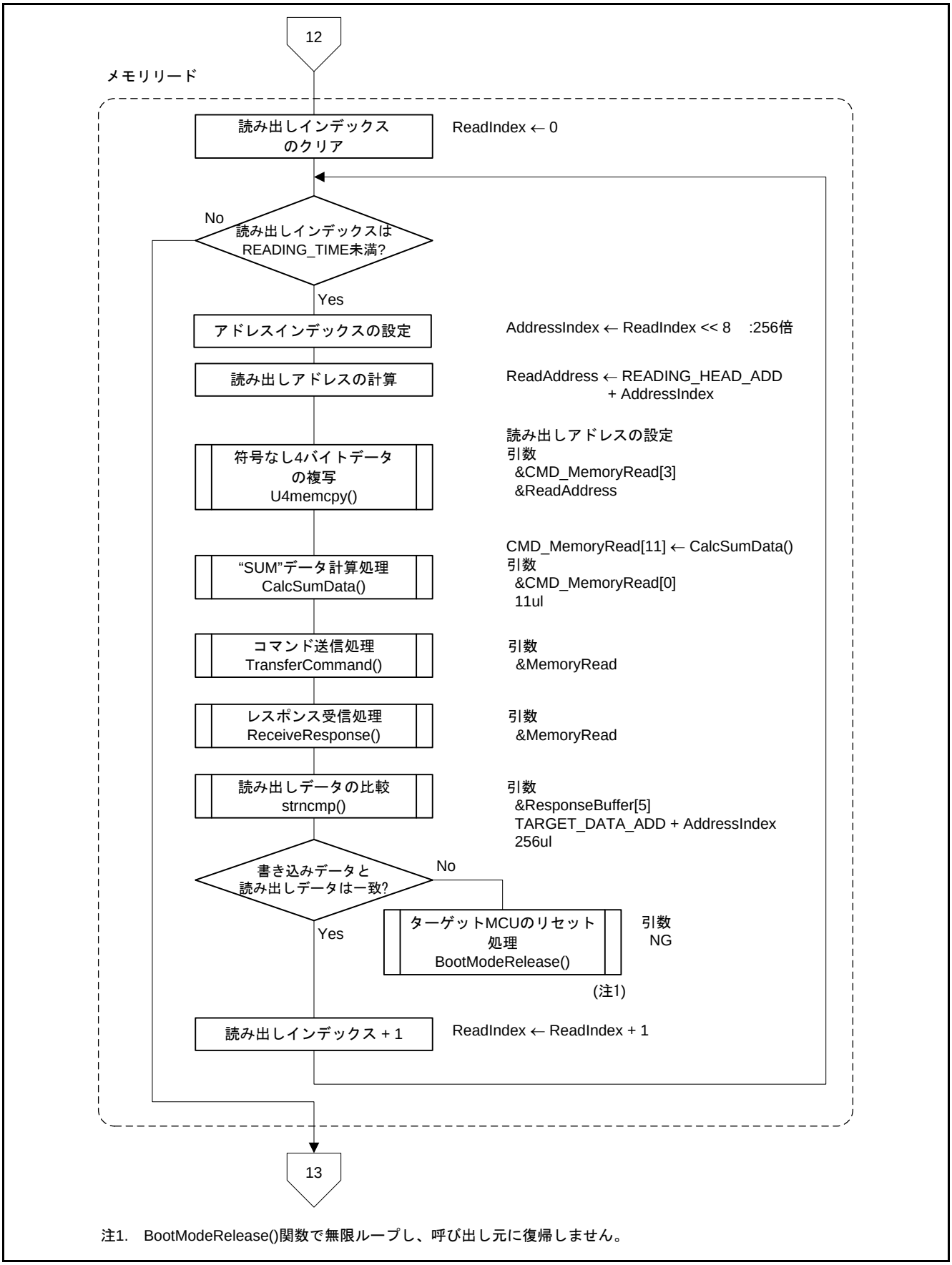
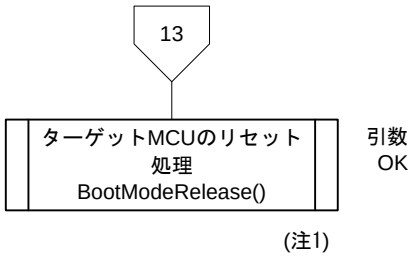


図5.28　メイン処理、通信プロトコル制御



注1. BootModeRelease()関数で無限ループし、呼び出し元に復帰しません。

図5.29 メイン処理、通信プロトコル制御



注1.　BootModeRelease()関数で無限ループし、呼び出し元に復帰しません。

図5.30　メイン処理、通信プロトコル制御

5.10.2　周辺機能初期設定

図 5.31に周辺機能初期設定のフローチャートを示します。

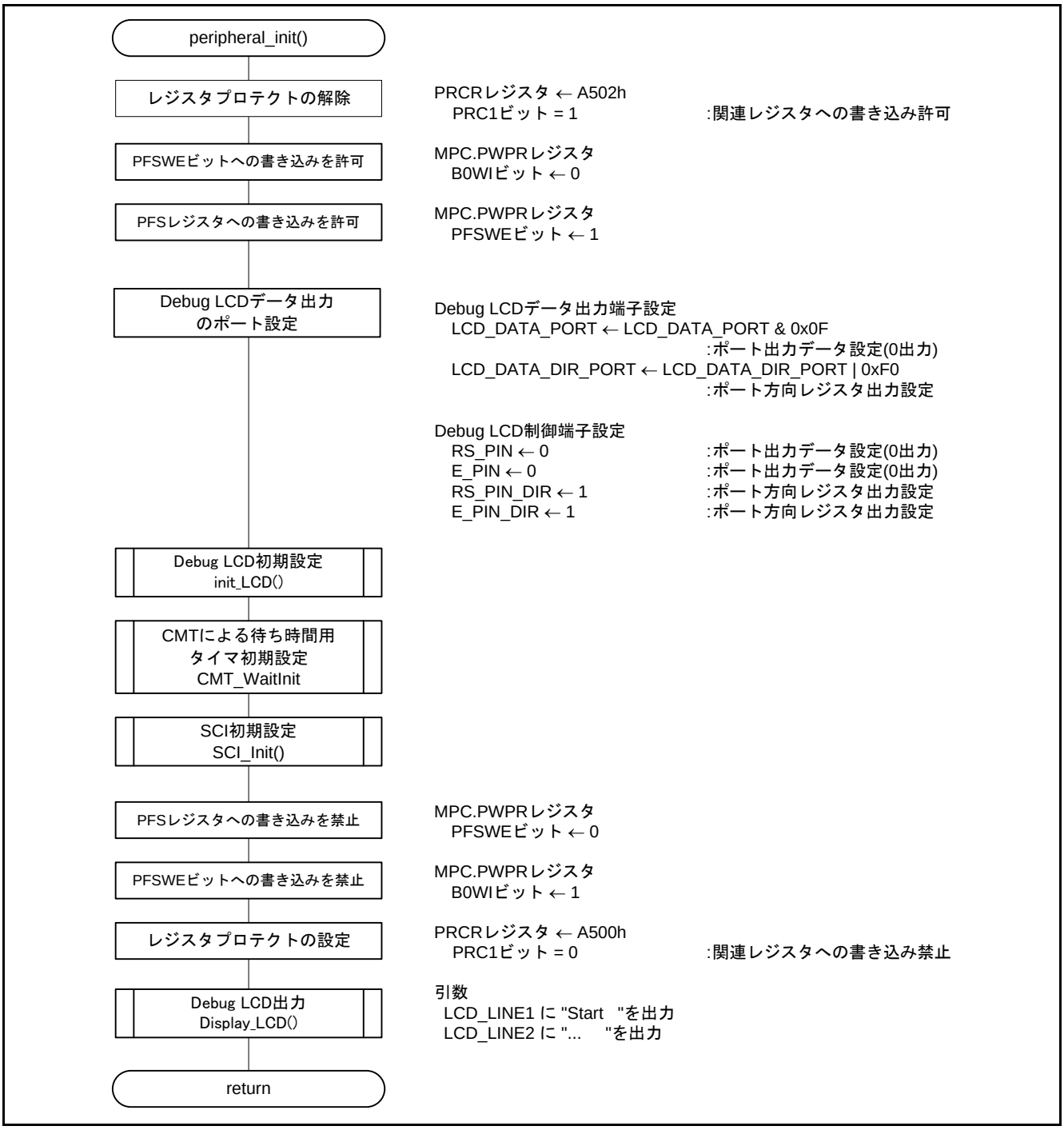


図5.31　周辺機能初期設定

5.10.3　CMT による待ち時間用タイマ初期設定

図 5.32にCMT による待ち時間用タイマ初期設定のフローチャートを示します。

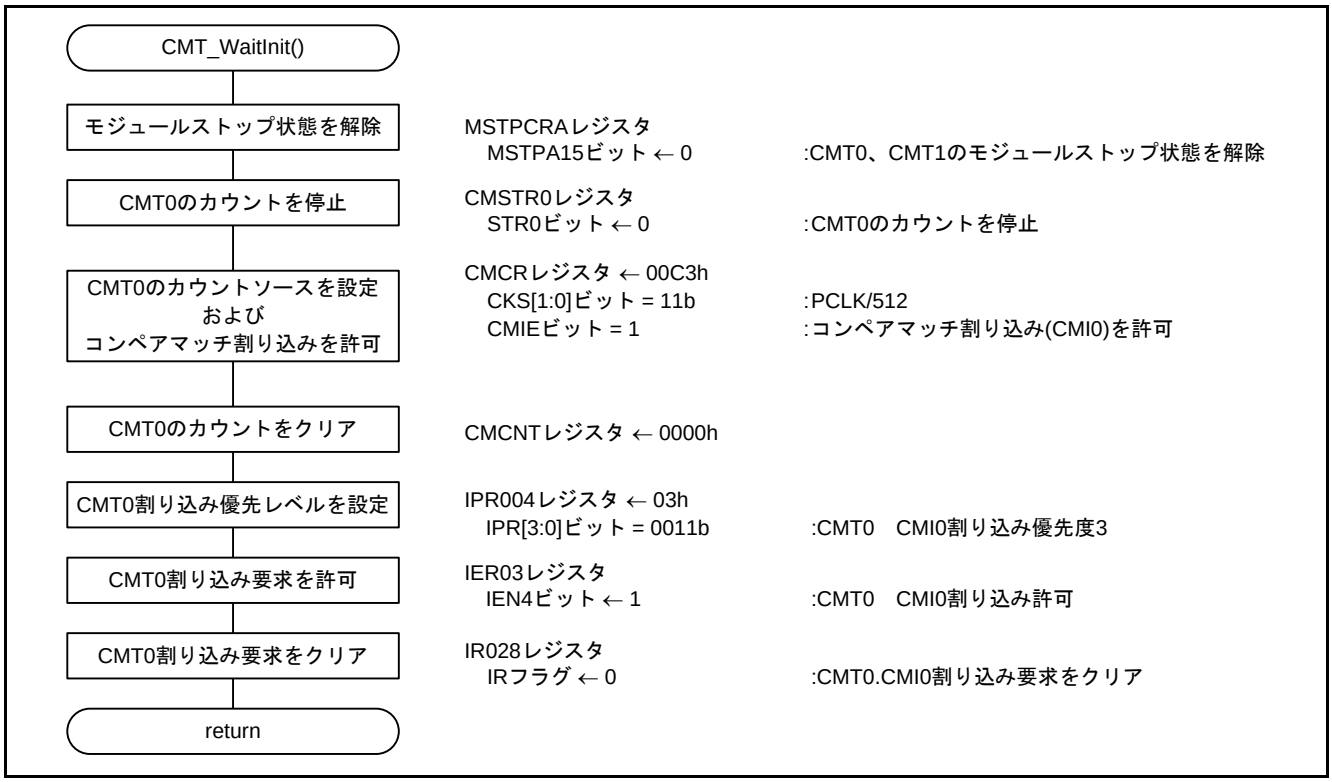


図5.32　CMT による待ち時間用タイマ初期設定

5.10.4　CMT による待ち時間設定

図 5.33にCMT による待ち時間設定のフローチャートを示します。

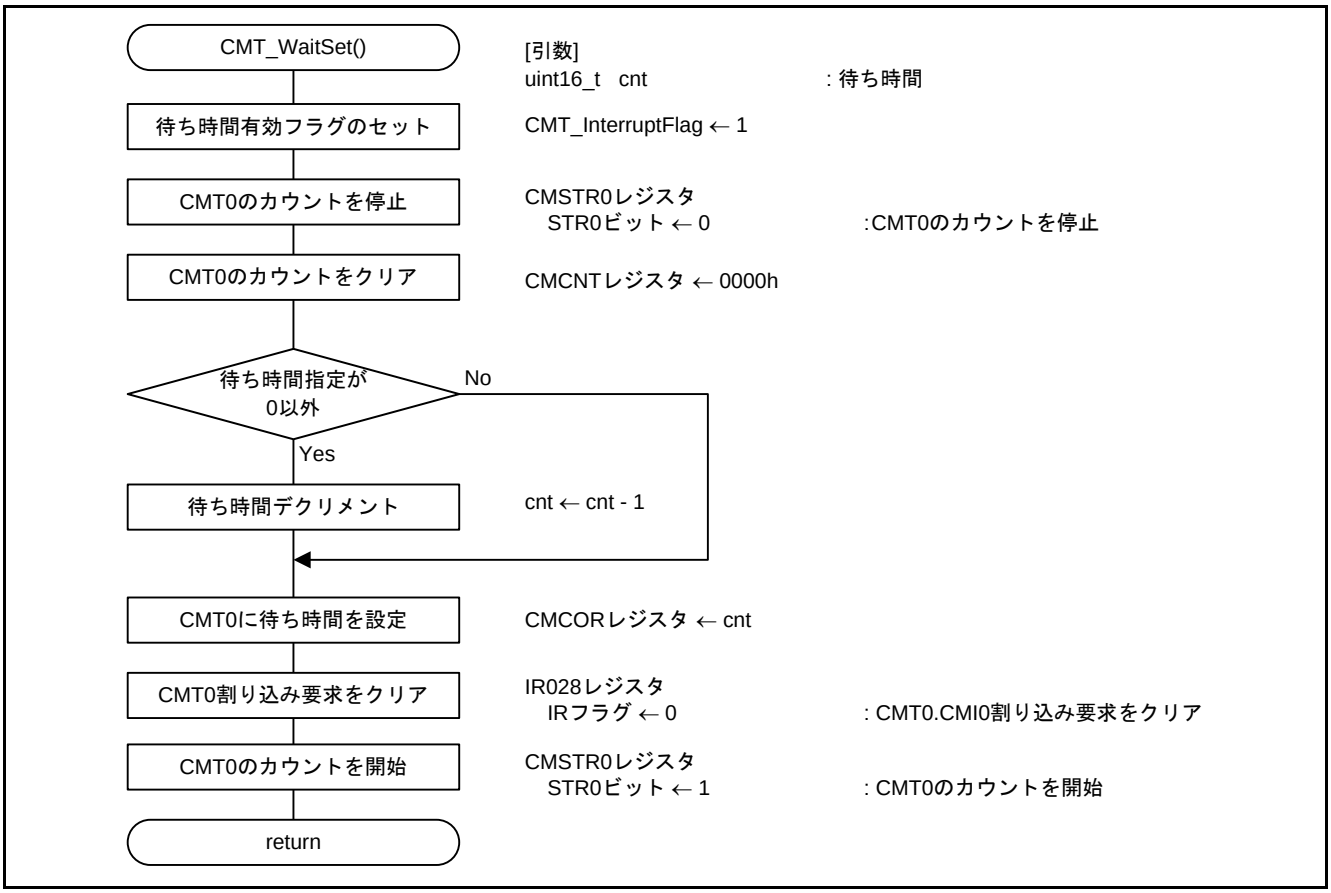


図5.33　CMT による待ち時間設定

5.10.5　CMT による時間待ち処理

図 5.34にCMT による時間待ち処理のフローチャートを示します。

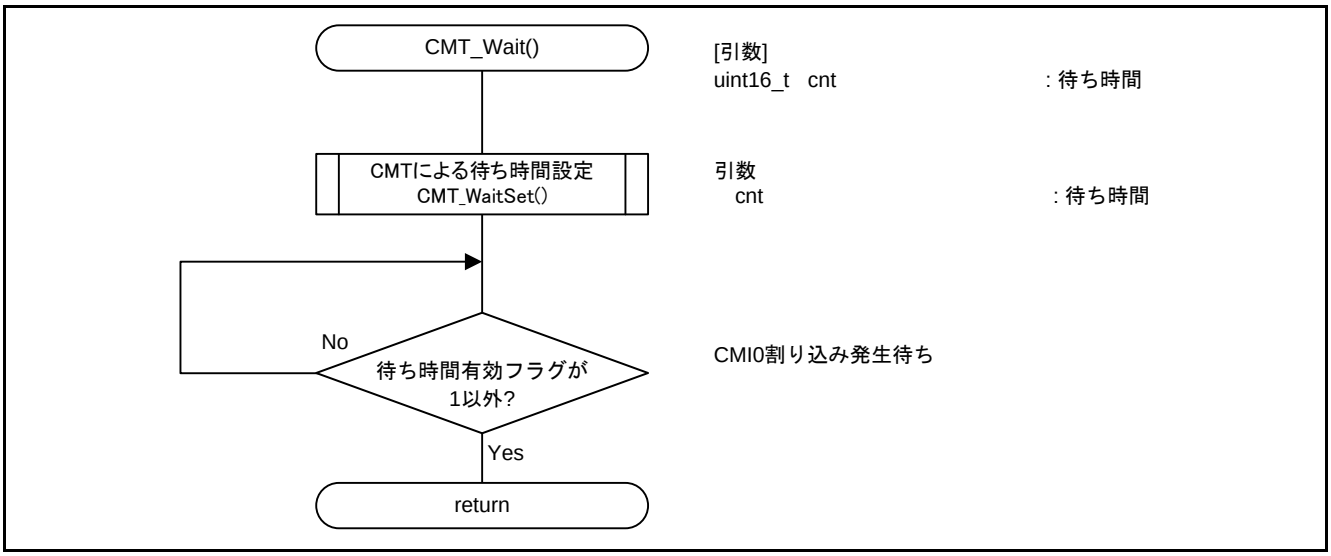


図5.34　CMT による時間待ち処理

5.10.6　CMT0,CMI0 割り込み処理

図 5.35にCMT0,CMI0 割り込み処理のフローチャートを示します。

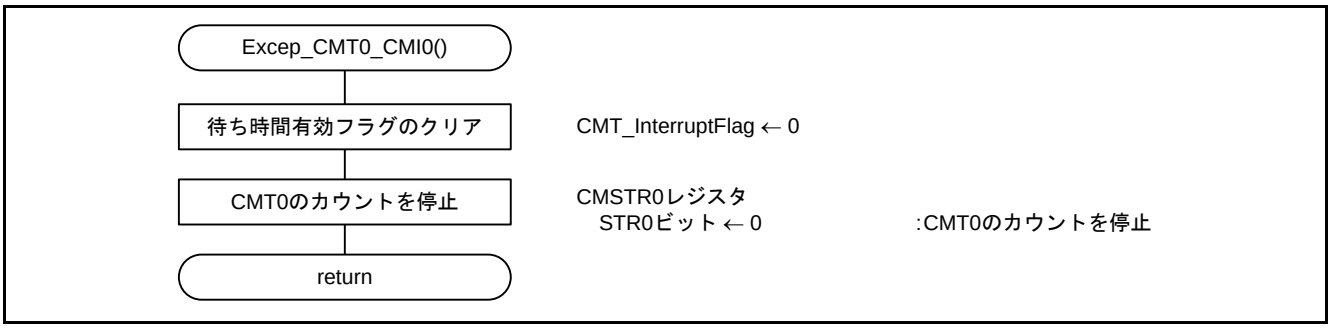


図5.35　CMT0,CMI0 割り込み処理

5.10.7　SCI 初期設定

図 5.36にSCI 初期設定のフローチャートを示します。

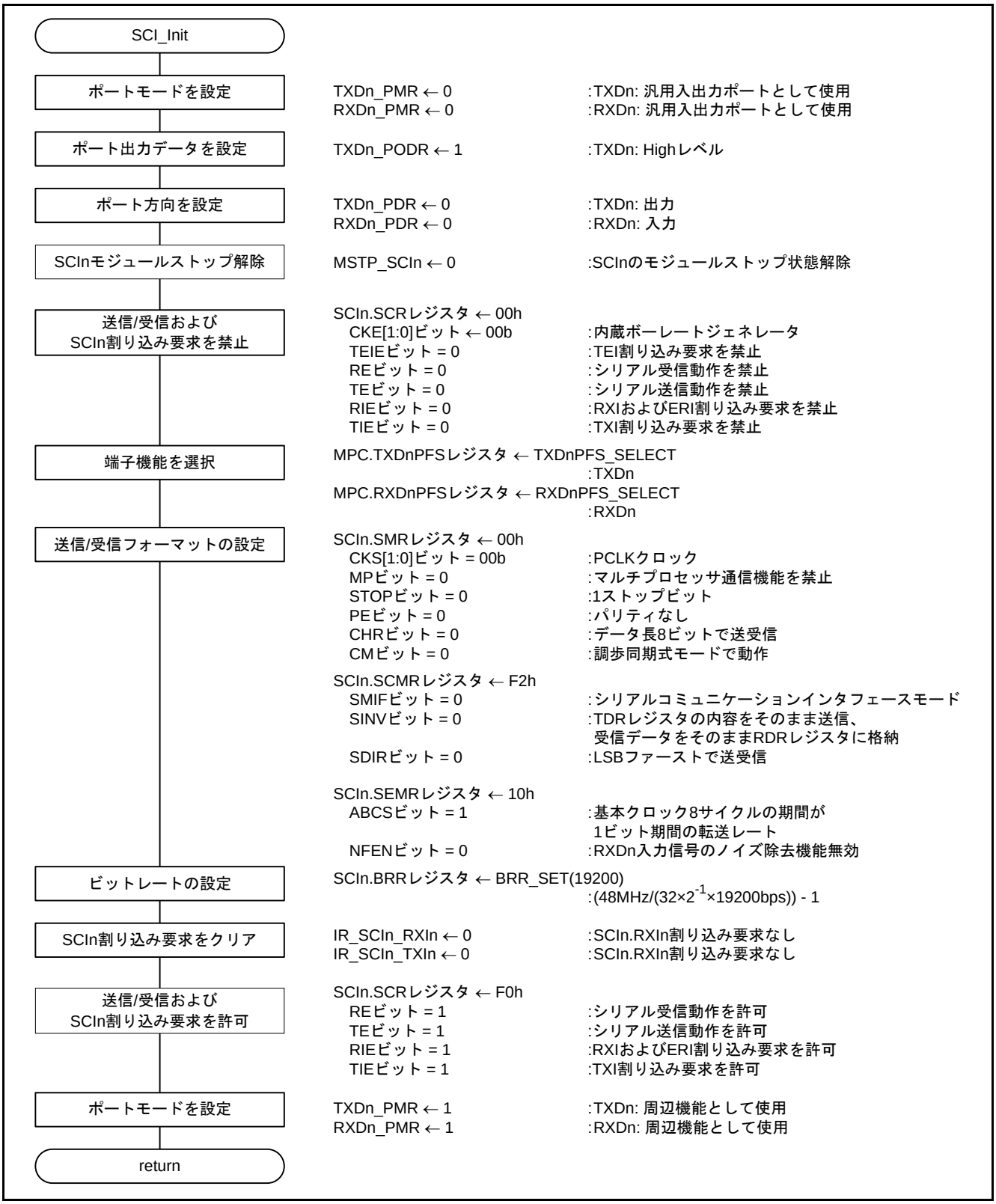


図5.36　SCI 初期設定

5.10.8　SCI ビットレート変更処理

図 5.37にSCI ビットレート変更処理のフローチャートを示します。

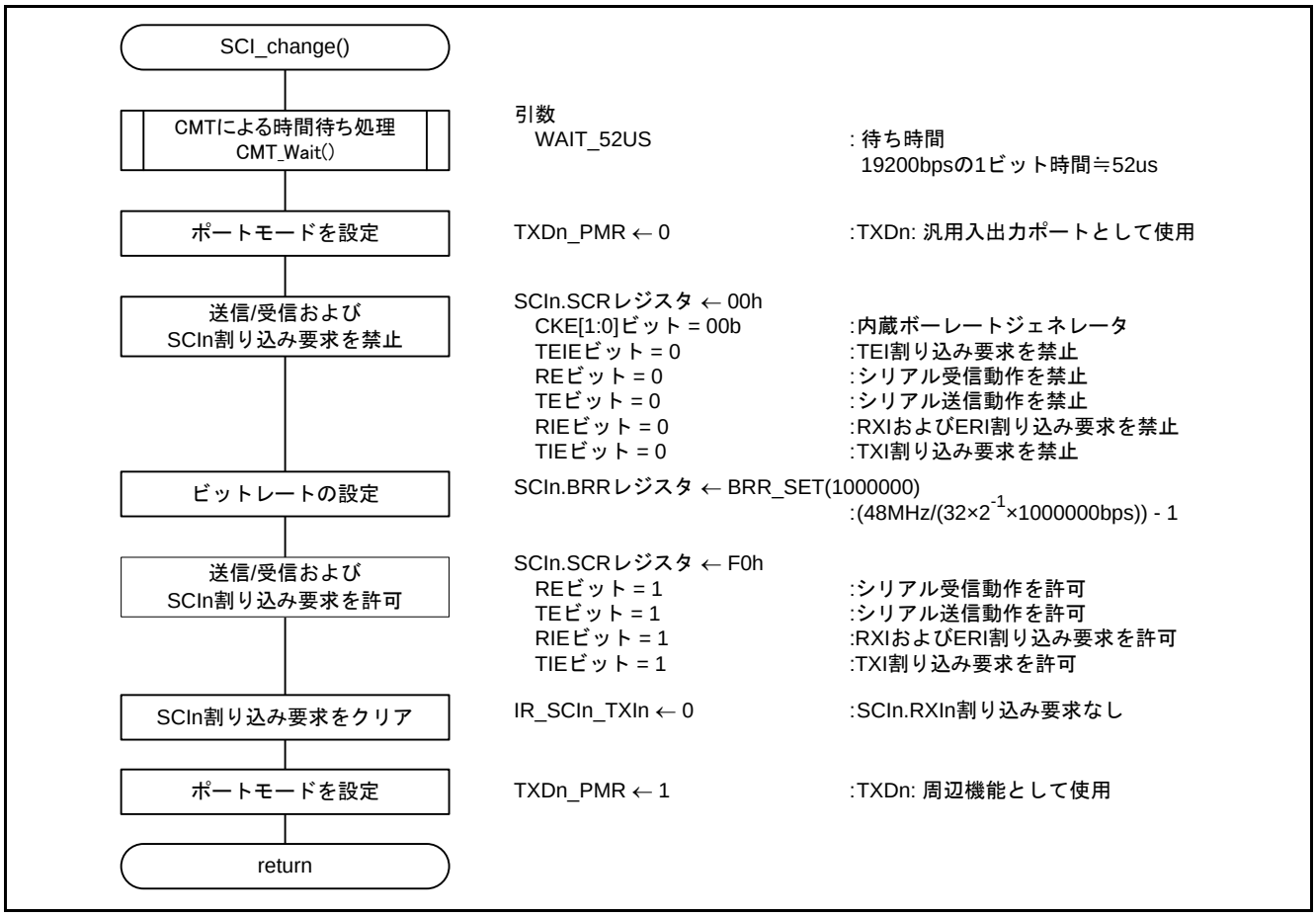


図5.37　SCI ビットレート変更処理

5.10.9　“SUM”データ計算処理

図 5.38に“SUM”データ計算処理のフローチャートを示します。

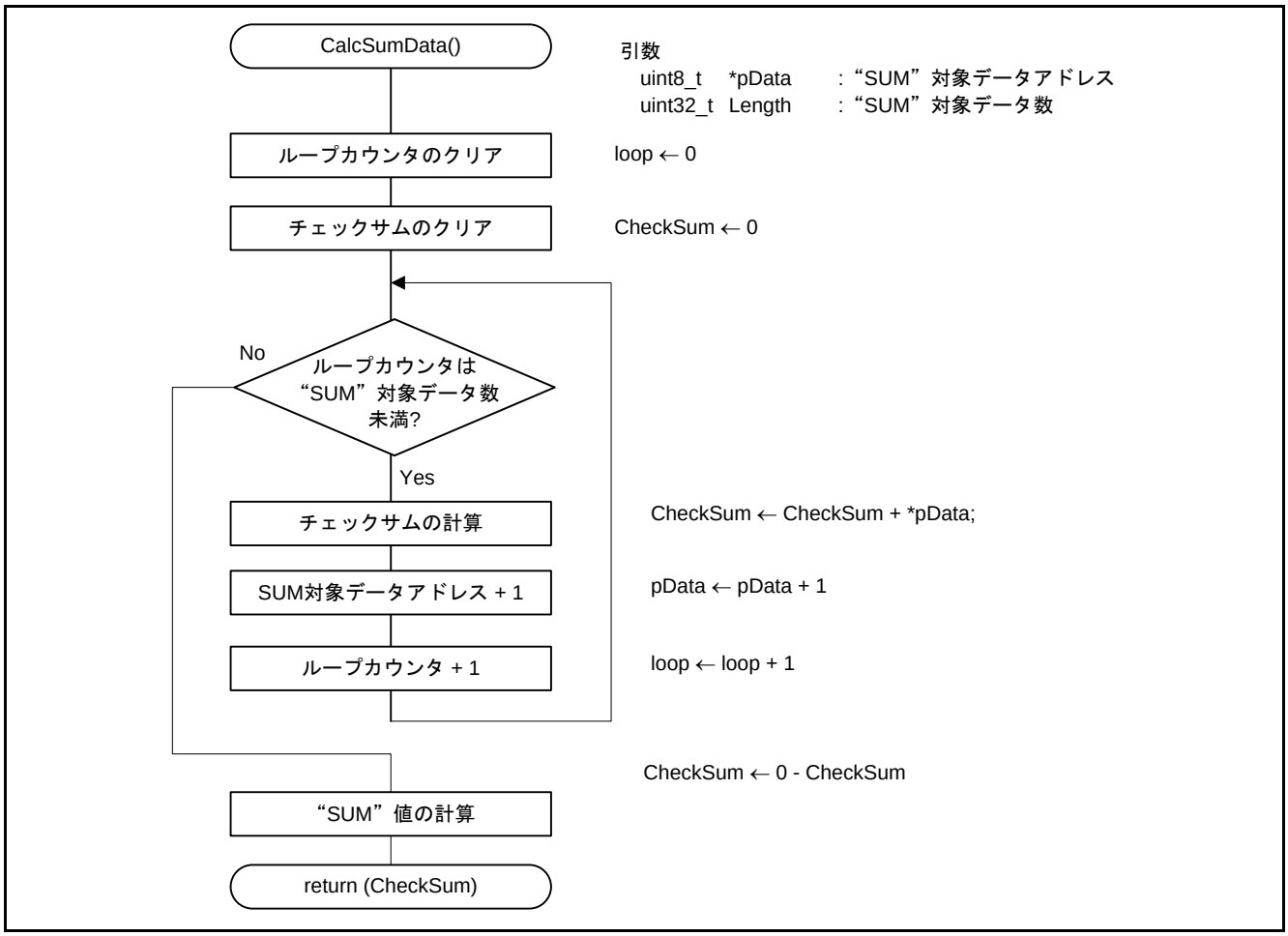


図5.38　“SUM”データ計算処理

5.10.10　ターゲット MCU のブートモード起動処理

図 5.39にターゲット MCU のブートモード起動処理のフローチャートを示します。

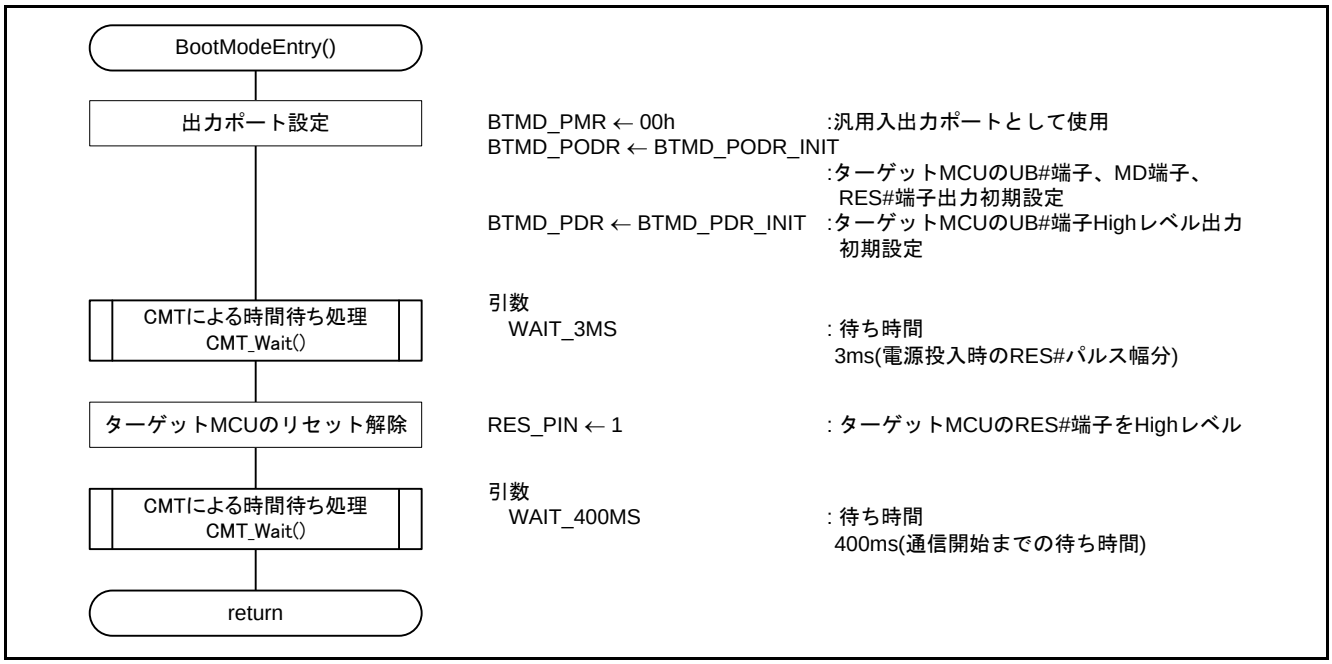


図5.39　ターゲット MCU のブートモード起動処理

5.10.11　ターゲット MCU のリセット処理

図 5.40にターゲット MCU のリセット処理のフローチャートを示します。

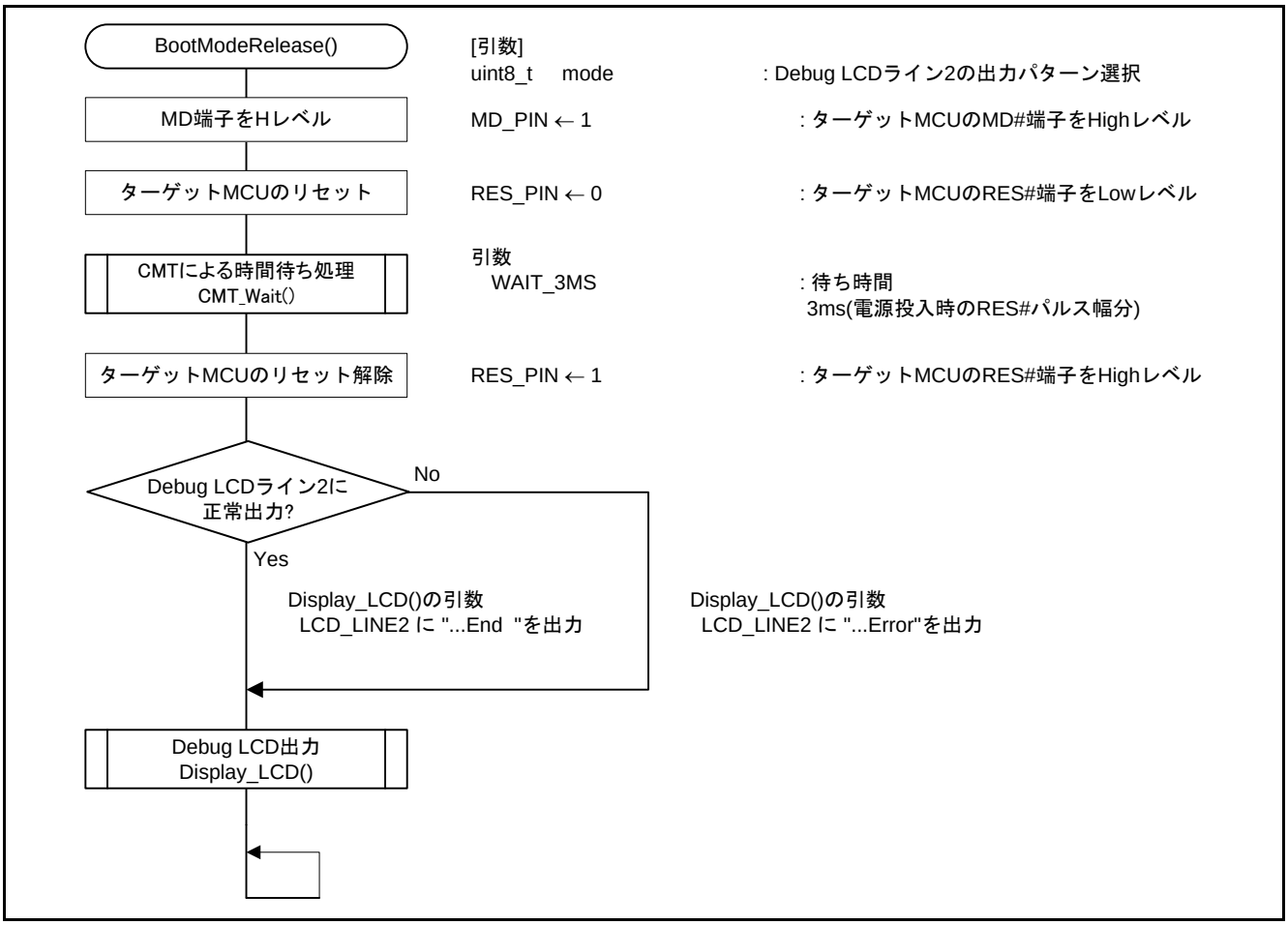


図5.40　ターゲット MCU のリセット処理

5.10.12　コマンド送信処理

図 5.41にコマンド送信処理のフローチャートを示します。

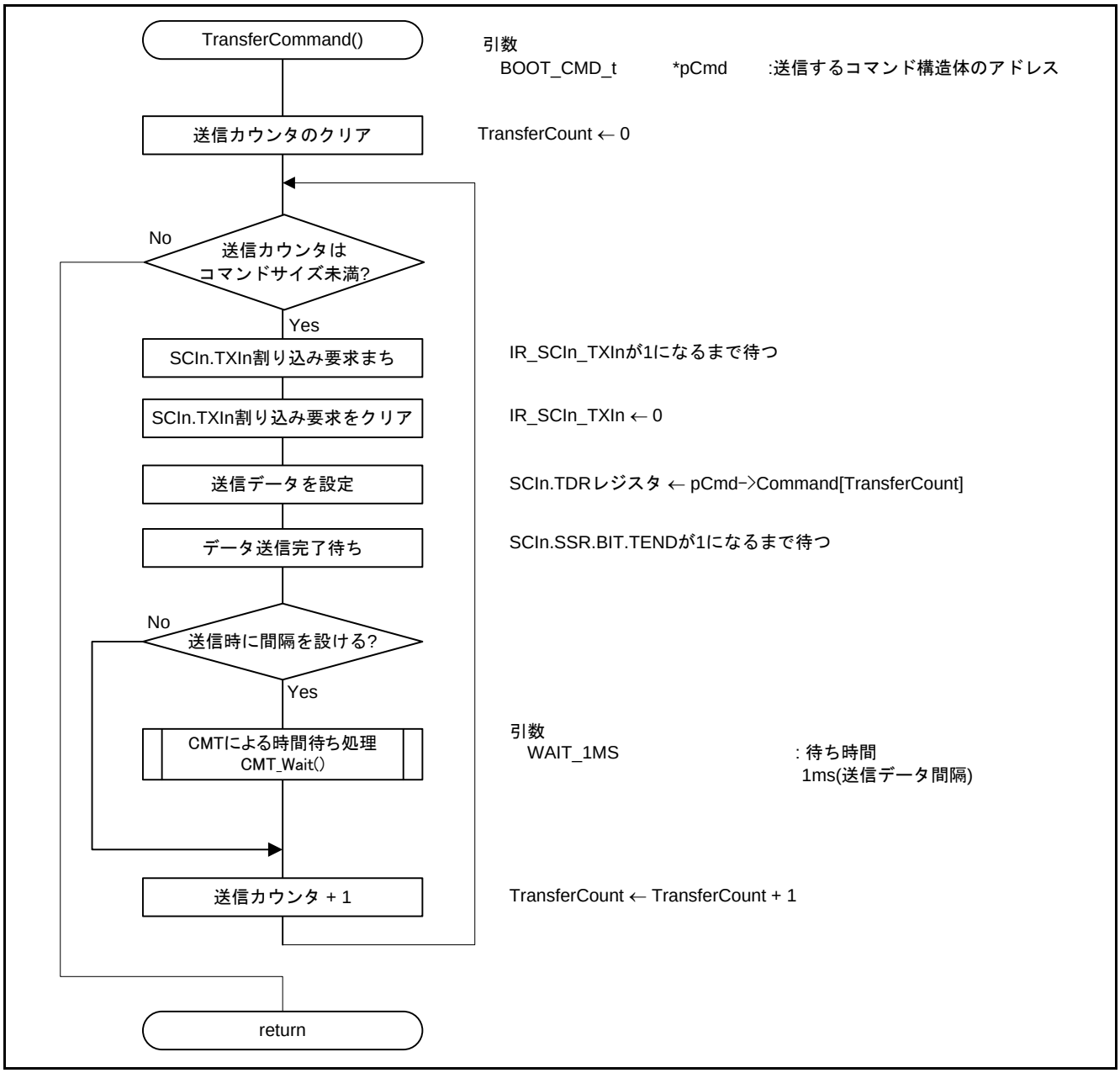


図5.41　コマンド送信処理

5.10.13　レスポンス受信処理

図 5.42から図 5.45にレスポンス受信処理のフローチャートを示します。

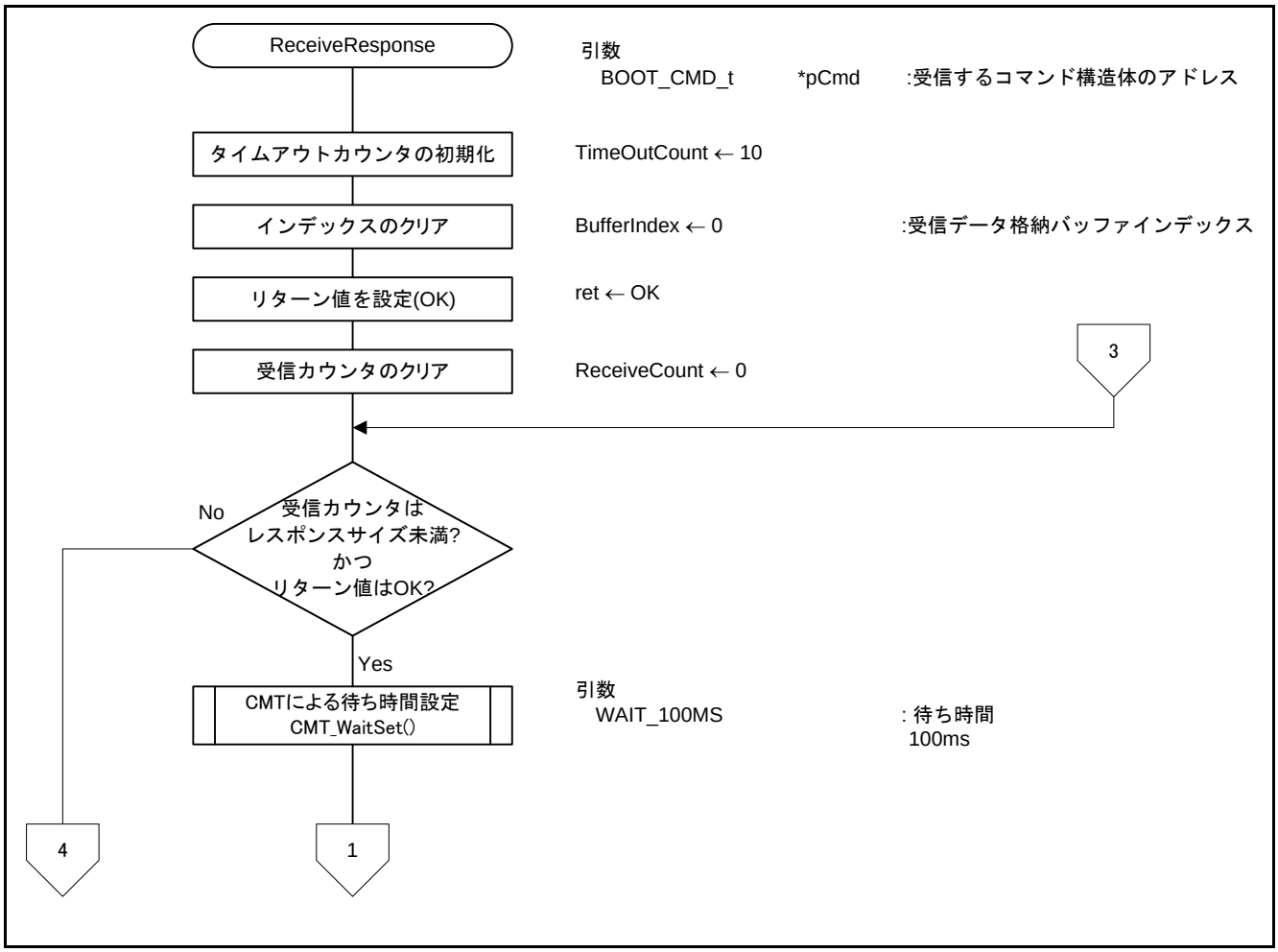
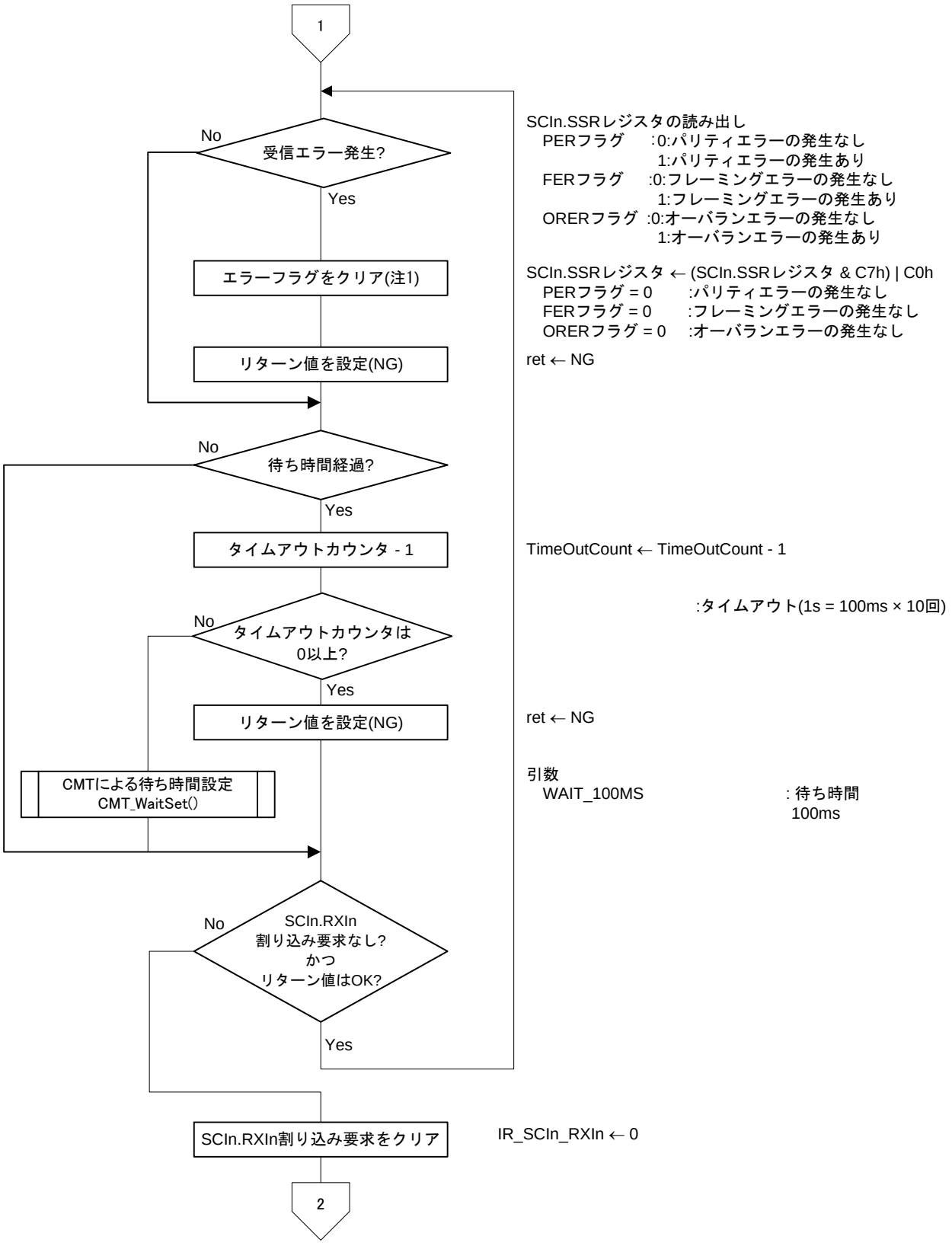


図5.42　レスポンス受信処理



注1. REビット、RIEビット、PERフラグ、FERフラグ、ORERフラグに値を書き込んだ後、書き込んだ値が読めることを確認してください。

図5.43 レスポンス受信処理

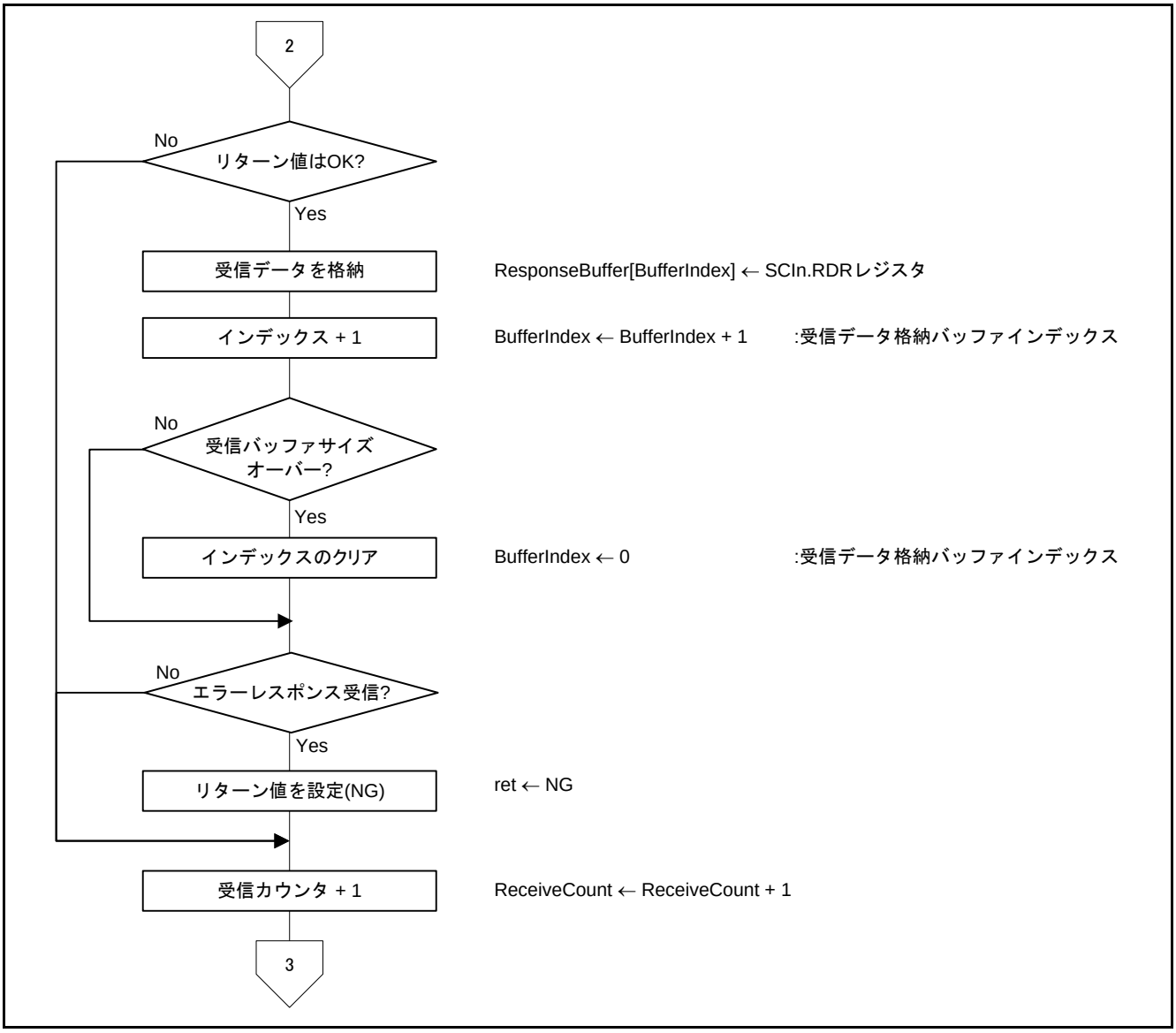
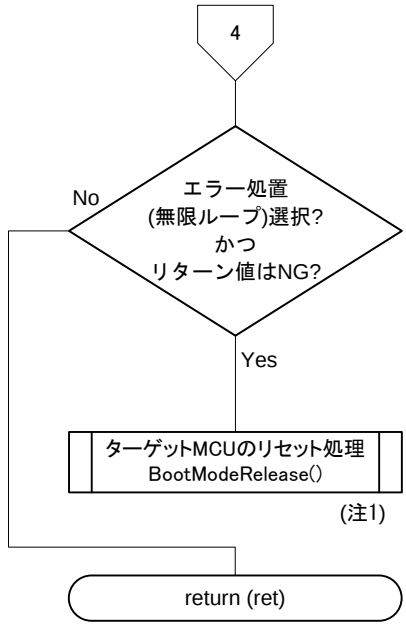


図5.44　レスポンス受信処理



引数
NG : LCD_LINE2 に "...Error"を出力

注1. BootModeRelease()関数で無限ループし、呼び出し元に復帰しません。

図5.45 レスポンス受信処理

5.10.14　符号なし 4 バイトデータの複写

図 5.46に符号なし 4 バイトデータの複写のフローチャートを示します。

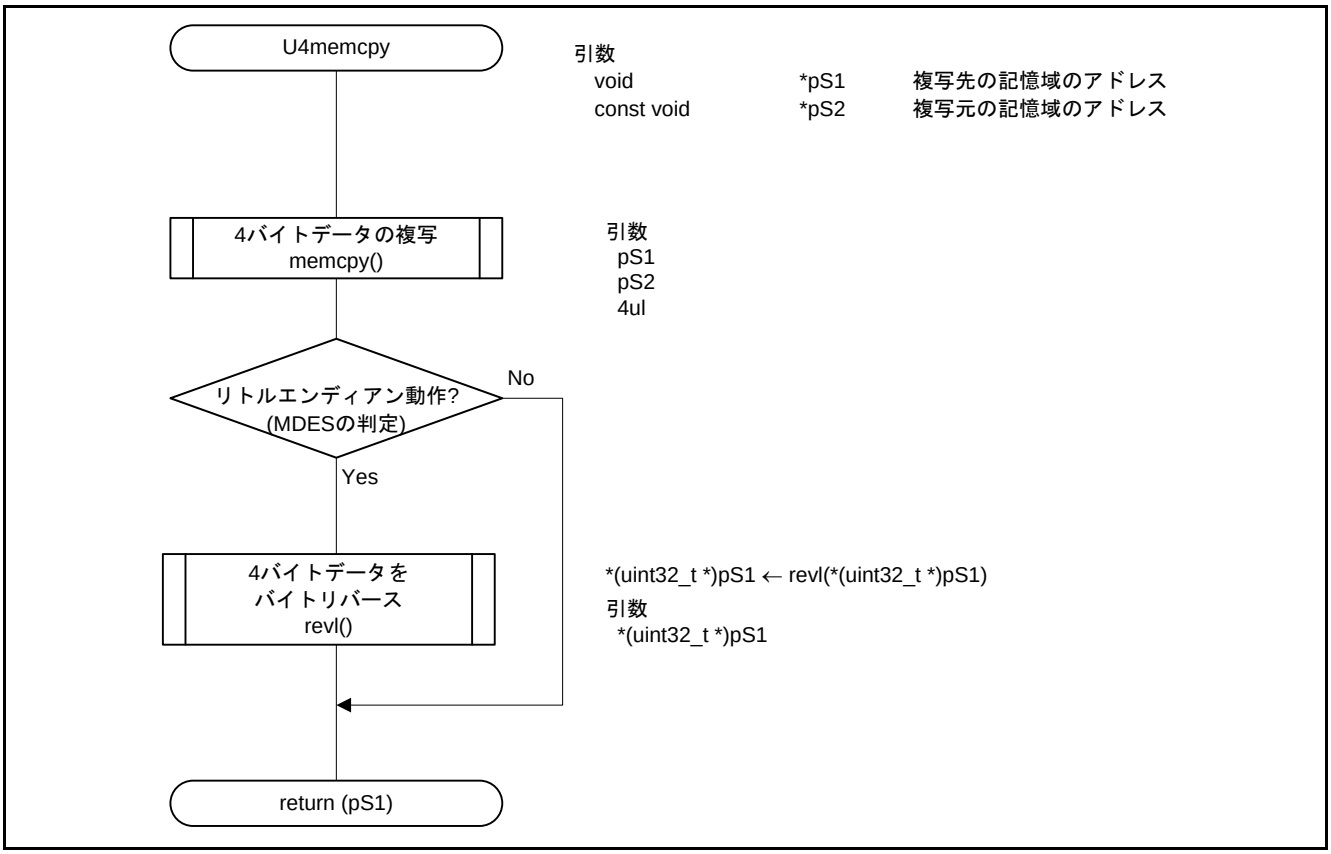


図5.46　符号なし 4 バイトデータの複写

6.　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。

7.　参考ドキュメント

ユーザーズマニュアル：ハードウェア
RX110 グループ ユーザーズマニュアル ハードウェア編 Rev.1.00 (R01UH0421JJ)
RX111 グループ ユーザーズマニュアル ハードウェア編 Rev.1.00 (R01UH0365JJ)
RX63N グループ、RX631 グループ ユーザーズマニュアル ハードウェア編 Rev.1.70 (R01UH0041JJ)
（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／テクニカルニュース
（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。）

ユーザーズマニュアル：開発環境
RX ファミリ C/C++コンパイラパッケージ V.1.01 ユーザーズマニュアル Rev.1.00 (R20UT0570JJ)
（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。）

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com

お問合せ先
http://japan.renesas.com/contact/

改訂記録	RX100シリーズ　Renesas Starter Kit+ for RX63N を使用した RX100 シリーズ用フラッシュプログラマ(SCI インタフェース)

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2014.04.01	—	初版発行
1.10	2020.08.20	—	toolchain　version の更新

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について

型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

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

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

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

(Rev.4.0-1　2017.11)

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

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

www.renesas.com/contact/

商標について