RZ/A2Mグループ

シリアルフラッシュメモリからのブート例

要旨

本アプリケーションノートは、RZ/A2Mの SPI マルチ I/O バスコントローラ（以下、SPIBSC とします）を使用して、ブートモード 3（シリアルフラッシュブート 3.3V 品）によってシリアルフラッシュメモリからブートを行う例について説明します。

動作確認デバイス

RZ/A2M

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

1.1 シリアルフラッシュメモリからのブート

RZ/A2Mは、ブートモード 3 の場合、SPI マルチ I/O バス空間に配置されたシリアルフラッシュメモリからブートします（以下、シリアルフラッシュブートとします）。図1.1にシリアルフラッシュブートの動作イメージを示します。

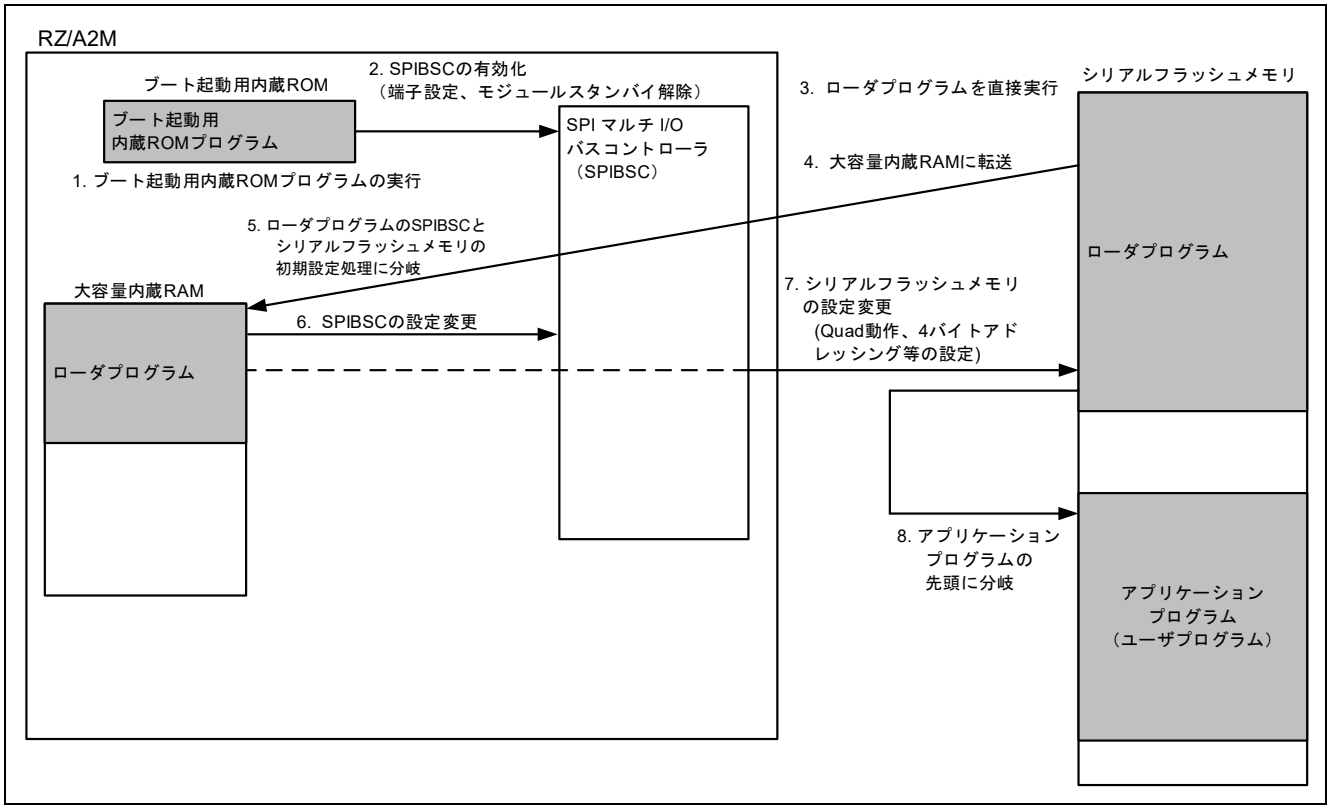


図1.1　シリアルフラッシュブートの動作イメージ

シリアルフラッシュブートの動作イメージについて説明します。

1　RZ/A2Mは、シリアルフラッシュブートで起動した場合、パワーオンリセット解除後にブート起動用内蔵 ROM プログラムを実行します。
2　ブート起動用内蔵 ROM プログラムは、SPIBSC を外部アドレス空間リードモードに設定し、SPI マルチ I/O バス空間に配置されたプログラムを直接実行できる状態にします。
3　シリアルフラッシュメモリに格納されたローダプログラムを直接実行します。
4　ローダプログラムのセクション初期化処理により、ローダプログラムをシリアルフラッシュメモリから大容量内蔵 RAM に転送します。
5　大容量内蔵 RAM に転送したローダプログラムの SPIBSC とシリアルフラッシュメモリの初期設定処理に分岐します。
6　ローダプログラムにより、SPIBSC の設定を変更します。
7　ローダプログラムにより、シリアルフラッシュメモリの設定を変更します。
8　アプリケーションプログラムの先頭アドレスに分岐します。

ブート起動用内蔵 ROM プログラムは、一般的なシリアルフラッシュメモリに共通でアクセスできる設定を行っているため、お客様が使用するシリアルフラッシュメモリに最適な設定を行う必要があります。このため、本アプリケーションノートでは、ブート起動用内蔵 ROM プログラムより分岐する SPI マルチ I/O バス空間の先頭番地（H'2000_0000）にローダプログラムを配置し、ローダプログラムによりお客様が使用するシリアルフラッシュメモリに最適な設定を行った後、お客様が作成するアプリケーションプログラム（ユーザプログラム）に分岐する方法を説明します。

1.2　使用する周辺機能

本サンプルコードでは、SPIBSC の設定とともに、クロックパルス発振器、割り込みコントローラ、汎用入出力ポート、メモリ管理ユニット、1 次キャッシュ（L1 キャッシュ）、および 2 次キャッシュ（L2 キャッシュ）の初期設定を行います。

本アプリケーションノートでは、SPI マルチ I/O バスコントローラを SPIBSC、クロックパルス発振器を CPG、割り込みコントローラを INTC、OS タイマを OSTM、FIFO 内蔵シリアルコミュニケーションインタフェースを SCIFA、汎用入出力ポートを GPIO、低消費電力モードを STB、メモリ管理ユニットを MMU とします。

表1.1に使用する周辺機能と用途を、図1.2にサンプルコード実行時の動作環境を示します。

表1.1　使用する周辺機能と用途

周辺機能	用途
SPI マルチ I/O バスコントローラ（SPIBSC）	外部アドレス空間リードモードに設定し、CPU が SPI マルチ I/O バス空間に接続されたシリアルフラッシュメモリから、直接リードするための信号を生成。
クロックパルス発振器（CPG）	RZ/A2Mの動作周波数の生成
割り込みコントローラ（INTC）	OSTM チャネル 0、OSTM チャネル 2 および SCIFA チャネル 4 の割り込み制御に使用
OS タイマ（OSTM）	OSTM チャネル 0 およびチャネル 2 を使用 • OSTM チャネル 0 LED 点灯および消灯の周期を生成 • OSTM チャネル 2 OS Abstration Layer による時間管理に使用
FIFO 内蔵シリアルコミュニケーション インタフェース（SCIFA）	SCIFA チャネル 4 を用いて、ホスト PC との通信用として使用
汎用入出力ポート（GPIO）	SCIFA チャネル 4 の兼用端子の切り替えに使用、LED の点灯および消灯のための端子制御に使用
低消費電力モード（STB）	RZ/A2Mの周辺 IO のモジュールスタンバイを解除するために使用、保持用内蔵 RAM をライト許可するために使用
メモリ管理ユニット（MMU）、L1 キャッシュ、L2 キャッシュ	RZ/A2Mの外部アドレス空間において、L1 キャッシュの有効領域の指定やメモリタイプの指定などの変換テーブルを生成。L1 キャッシュおよび L2 キャッシュを有効に設定

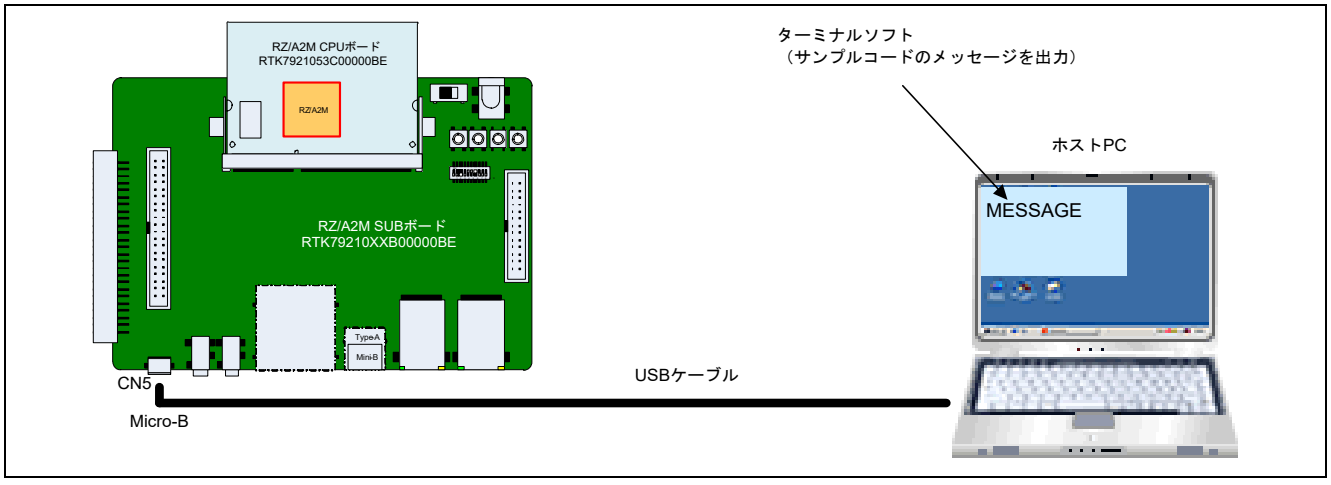


図1.2　動作環境

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（1/2）

項目	内容
使用 MCU	RZ/A2M
動作周波数（注）	CPU クロック（Iφ）：528MHz 画像処理クロック（Gφ）：264MHz 内部バスクロック（Bφ）：132MHz 周辺クロック 1（P1φ）：66MHz 周辺クロック 0（P0φ）：33MHz QSPI0_SPCLK：66MHz CKIO：132MHz
動作電圧	電源電圧（I/O）：3.3V 電源電圧（1.8/3.3V 切替 I/O（PVcc_SPI））：3.3V 電源電圧（内部）：1.2V
統合開発環境	e2 studio Version 2020-07 (20.7.0)
C コンパイラ	GNU Arm Embedded Toolchain 6-2017-q2-update
	コンパイラオプション（ディレクトリパスの追加は除く） Release コンフィグレーション： -mcpu=cortex-a9 -march=armv7-a -marm -mlittle-endian -mfloat-abi=hard -mfpu=neon -mno-unaligned-access -Os -ffunction-sections -fdata-sections -Wunused -Wuninitialized -Wall -Wextra -Wmissing-declarations -Wconversion -Wpointer-arith -Wpadded -Wshadow -Wlogical-op -Waggregate-return -Wfloat-equal -Wnull-dereference -Wmaybe-uninitialized -Wstack-usage=100 -fabi-version=0 Hardware Debug コンフィグレーション： -mcpu=cortex-a9 -march=armv7-a -marm -mlittle-endian -mfloat-abi=hard -mfpu=neon -mno-unaligned-access -Og -ffunction-sections -fdata-sections -Wunused -Wuninitialized -Wall -Wextra -Wmissing-declarations -Wconversion -Wpointer-arith -Wpadded -Wshadow -Wlogical-op -Waggregate-return -Wfloat-equal -Wnull-dereference -Wmaybe-uninitialized -g3 -Wstack-usage=100 -fabi-version=0

【注】 クロックモード 1（EXTAL 端子からの 24MHz のクロック入力）で使用時の動作周波数です。

表2.2　動作確認条件（2/2）

項目	内容
動作モード	ブートモード 3（シリアルフラッシュブート 3.3V 品）
ターミナルソフトの通信設定	• 通信速度：115200bps • データ長：8 ビット • パリティ：なし • ストップビット長：1 ビット • フロー制御：なし
使用ボード	RZ/A2M CPUボード　RTK7921053C00000BE RZ/A2M SUBボード　RTK79210XXB00000BE
使用デバイス （ボード上で使用する機能）	• シリアルフラッシュメモリ（SPI マルチ I/O バス空間に接続） メーカ名：Macronix社、型名：MX25L51245GXD • RL78/G1C（USB 通信とシリアル通信を変換し、ホスト PC との通信に使用） • LED1

3.　関連アプリケーションノート

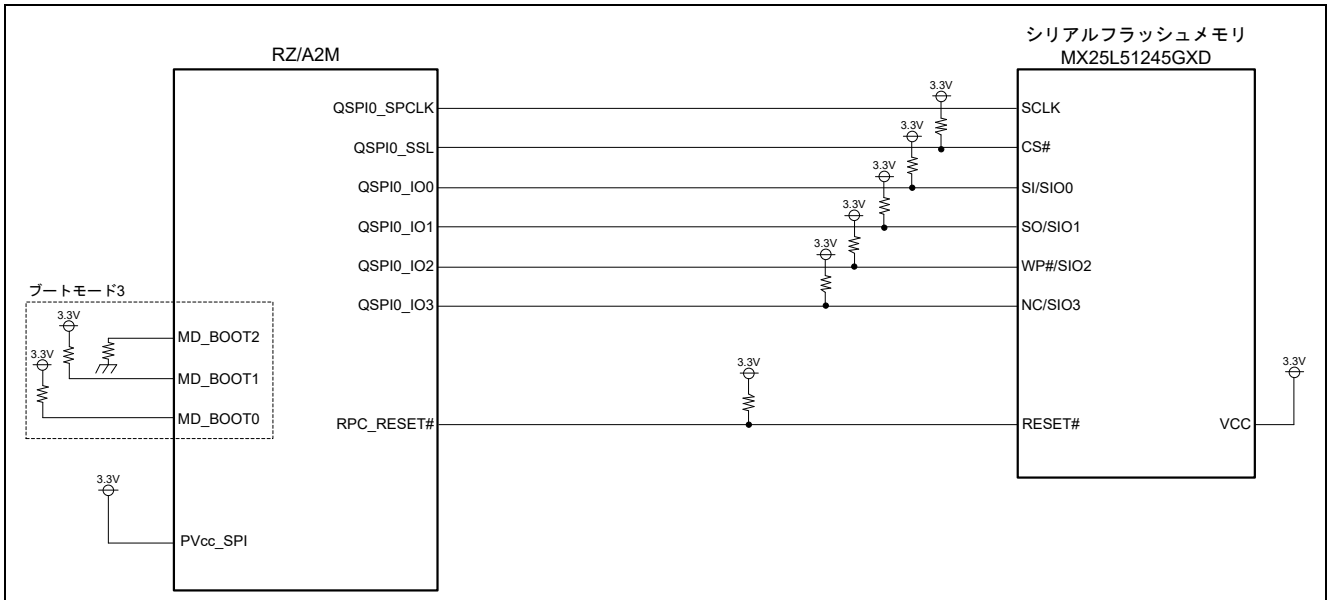
本アプリケーションノートに関連するアプリケーションノートを以下に示します。併せて参照してください。

- RZ/A2M グループ 初期設定例（R01AN4321JJ）

4.　ハードウェア説明

4.1　ハードウェア構成例

本アプリケーションノートで紹介するシリアルフラッシュブート例は、ブートモード 3 を使用して、SPI マルチ I/O バス空間に接続されたシリアルフラッシュメモリに格納されたプログラムにより処理を行います。図4.1にブートモード3でシリアルフラッシュメモリからブートする場合の接続例を示します。



【注】　#は負論理（またはアクティブロー）を示す記号です。

図4.1　ブートモード 3 でシリアルフラッシュメモリからブートする場合の接続例

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
MD_BOOT2	入力	ブートモードの選択（ブートモード 3 に設定） MD_BOOT2："L"、MD_BOOT1："H"、MD_BOOT0："H"
MD_BOOT1	入力	
MD_BOOT0	入力	
QSPI0_SPCLK	出力	シリアルフラッシュメモリのクロック
QSPI0_SSL	出力	シリアルフラッシュメモリのスレーブセレクト
QSPI0_IO0	入出力	シリアルフラッシュメモリのデータ 0
QSPI0_IO1	入出力	シリアルフラッシュメモリのデータ 1
QSPI0_IO2	入出力	シリアルフラッシュメモリのデータ 2
QSPI0_IO3	入出力	シリアルフラッシュメモリのデータ 3
RPC_RESET#	出力	シリアルフラッシュメモリのリセット
P6_0	出力	LED の点灯および消灯
RxD4(P9_1)	入力	シリアル受信データ信号
TxD4(P9_0)	出力	シリアル送信データ信号

【注】 #は負論理（またはアクティブロー）を示す記号です。

5. ソフトウェア説明

5.1 動作概要

ここでは、本アプリケーションノートのサンプルコードの動作概要について説明します。

5.1.1 シリアルフラッシュブートに関する用語

表5.1に本アプリケーションノートで説明するシリアルフラッシュブート動作に関する用語を示します。

表5.1 シリアルフラッシュブート動作に関する用語

用語	説明
ブート起動用内蔵 ROM プログラム	ブート起動用内蔵 ROM プログラムは、ブートモード 3（シリアルフラッシュブート 3.3V 品）で起動した場合に、SPI マルチ I/O バス空間に接続されたシリアルフラッシュメモリに格納されているプログラムを直接実行するための設定を行うプログラムです。 RZ/A2Mはブート起動用内蔵 ROM プログラムの実行完了後、SPI マルチ I/O バス空間の先頭アドレスである H'2000_0000 番地に分岐します。なお、ブート起動用内蔵 ROM プログラムでは、一般的なシリアルフラッシュメモリに共通でアクセスできる設定を行っています。 RZ/A2Mの内蔵 ROM に格納されているプログラムのため、お客様が作成する必要はありません。
ローダプログラム	ローダプログラムは、ブート起動用内蔵 ROM プログラムの処理完了後に実行するプログラムです。 ローダプログラムは、お客様が使用するシリアルフラッシュメモリに合わせて、SPIBSC およびシリアルフラッシュメモリのレジスタ設定処理を行い、アプリケーションプログラムの先頭アドレスへ分岐する処理を行います。 ローダプログラムは、本アプリケーションノートを参考に、使用するシリアルフラッシュメモリの仕様に合わせてお客様が作成してください。なお、サンプルコードでは、Macronix社製シリアルフラッシュメモリ（MX25L51245GXD）を使用する場合に最適な設定を行っています。
アプリケーションプログラム （ユーザプログラム）	アプリケーションプログラムは、お客様がシステムに合わせて作成するプログラムです。

5.1.2　サンプルコード全体の動作概要

サンプルコードは、ブート起動用内蔵 ROM プログラムの処理完了後に実行するローダプログラムとアプリケーションプログラムで構成されています。

1　ローダプログラム（プロジェクト名：rza2m_sflash_boot_loader_gcc）
ローダプログラムは、使用するシリアルフラッシュメモリ（Macronix社製シリアルフラッシュメモリ（MX25L51245GXD））に最適な設定を行います。ローダプログラムはブート起動用内蔵 ROM プログラムより分岐する SPI マルチ I/O バス空間の先頭番地（H'2000_0000）に配置し、ブート起動用内蔵 ROM プログラムから実行できるようにしています。ローダプログラム実行後、アプリケーションプログラムの先頭番地に分岐します。
アプリケーションプログラムの先頭アドレスは、「linker_script.ld」のリンカスクリプトのシンボル定義"__application_base_address"により指定しています。

2　アプリケーションプログラム（プロジェクト名：rza2m_sflash_boot_sample_osless_gcc）
アプリケーションプログラムは、ローダプログラムにてシリアルフラッシュメモリに最適な設定後に実行するプログラムです。アプリケーションプログラムの"VECTOR_TABLE"のセクションが、"__application_base_address"で指定したアドレスと一致するように配置アドレスを変更してください。
サンプルコードでは、アプリケーションプログラムを H'2001_0000 番地に配置しています。

図5.1に本アプリケーションノートのサンプルコードの動作概要を示します。

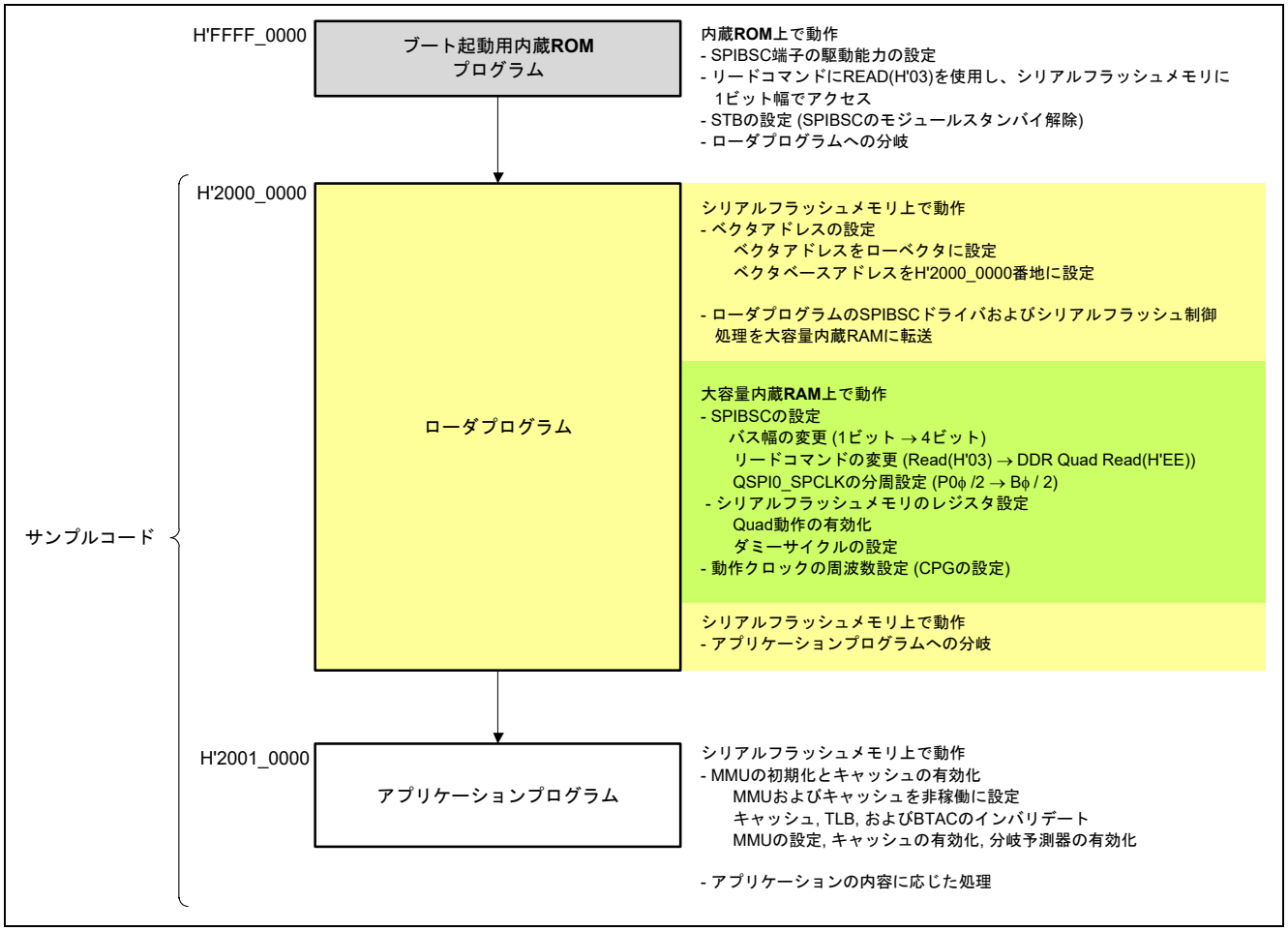


図5.1　サンプルコードの動作概要

5.1.3　ローダプログラムの動作概要

ローダプログラムは、ブート起動用内蔵 ROM プログラムの処理完了後に実行されるプログラムです。ローダプログラムは、ブート起動用内蔵 ROM プログラムより分岐する SPI マルチ I/O バス空間の先頭番地（H'2000_0000）に配置してください。

ブート起動用内蔵 ROM プログラムは、SPIBSC を外部アドレス空間リードモードに設定します。設定により、RZ/A2Mは SPI マルチ I/O バス空間へのリードを SPI 通信に変換し、接続されたシリアルフラッシュメモリに対して直接リードが可能となり、SPI マルチ I/O バス空間に配置されたプログラムを直接実行することが可能な状態となります。SPI 通信への変換に使用するシリアルフラッシュメモリへのコマンドの設定は、一般的なシリアルフラッシュメモリに共通でアクセスできる設定にしているため、ローダプログラムにて、お客様が使用するシリアルフラッシュメモリに最適な設定を行う必要があります。

シリアルフラッシュメモリの最適な設定は、SPIBSC モジュール内のレジスタの設定（以下、SPIBSC 設定とします）、およびシリアルフラッシュメモリのレジスタ設定（以下、シリアルフラッシュメモリ設定とします）を行う必要があります。

サンプルコードのローダプログラムでは、データバス幅を 4 ビットにし、使用するリードコマンドに合わせて転送ビットレートを最適化し、4 バイトアドレスを出力するために、SPIBSC のレジスタを設定します。また、シリアルフラッシュメモリのダミーサイクル数、Quad 動作の有効化、4 バイトアドレッシングへの変更を行うために、Macronix社製シリアルフラッシュメモリ（MX25L51245GXD）のレジスタを設定し、MX25L51245GXDにアクセスするための最適な設定を行っています。

ローダプログラムの SPIBSC 設定およびシリアルフラッシュメモリ設定を行う処理は、SPI マルチ I/O バス空間に配置されたプログラムで設定することはできないため、大容量内蔵 RAM 上で実行する必要があり、サンプルコードでは SPIBSC ドライバの処理やシリアルフラッシュメモリの制御処理を大容量内蔵 RAM に転送して実行しています。

ブート起動用内蔵 ROM プログラムおよびローダプログラム実行後の設定については、表5.2～表5.4を参照してください。

表5.2～表5.4に、ブート起動用内蔵 ROM プログラムおよびローダプログラムの設定内容を示します。

ローダプログラムにて、表5.2～表5.4に示す設定を行った後、アプリケーションプログラムの先頭番地に分岐します。サンプルコードでは、アプリケーションプログラムを H'2001_0000 番地に配置しています。

表5.2　ブート起動用内蔵 ROM プログラムおよびローダプログラムの設定内容（1/3）

	項目	ブート起動用内蔵 ROM プログラム実行後	ローダプログラム実行後
SPIBSC 設定	遅延設定		
	次アクセス遅延設定： SSLDR.SPNDL[2:0]	B'000 (1 QSPIn_SPCLK)	B'000 (1 QSPIn_SPCLK)
	QSPIn_SSL ネゲート遅延設定： SSLDR.SLNDL[2:0]	B'000 (1 QSPIn_SPCLK)	B'000 (1 QSPIn_SPCLK)
	クロック遅延設定： SSLDR.SCKDL[2:0]	B'000 (1.5 QSPIn_SPCLK)	B'000 (1.5 QSPIn_SPCLK)
	シリアルクロック： (QSPI0_SPCLK)	P0φ/ 2=16.5 [MHz] (注 1)	Bφ/ 2=66 [MHz] (注 1)
	QSPIn_SSL 出力アイドル値固定： QSPIn_IO3 の設定 QSPIn_IO2 の設定 QSPIn_IO1 の設定 QSPIn_IO0 の設定	QSPIn_SSL ネゲート期間の出力値を、前回転送の最終ビットに設定 CMNCR.MOIIO3[1:0]=B'10 CMNCR.MOIIO2[1:0]=B'10 CMNCR.MOIIO1[1:0]=B'10 CMNCR.MOIIO0[1:0]=B'10	QSPIn_SSL ネゲート期間の出力値を、Hi-Z に設定 CMNCR.MOIIO3[1:0]=B'11 CMNCR.MOIIO2[1:0]=B'11 CMNCR.MOIIO1[1:0]=B'11 CMNCR.MOIIO0[1:0]=B'11
	1 ビット幅時の端子出力値の設定: QSPIn_IO3 の設定 QSPIn_IO2 の設定 QSPIn_IO0 の設定	1 ビット幅時の端子の出力値を、前回転送の最終ビットに設定 CMNCR.IO3FV[1:0]=B'10 CMNCR.IO2FV[1:0]=B'10 CMNCR.IO0FV[1:0]=B'10	1 ビット幅時の端子の出力値を、Hi-Z に設定 CMNCR.IO3FV[1:0]=B'11 CMNCR.IO2FV[1:0]=B'11 CMNCR.IO0FV[1:0]=B'11 (注 2)
	データバス幅： CMNCR.BSZ[1:0]	シリアルフラッシュ×1 B'00	シリアルフラッシュ×1 B'00
	リードキャッシュ：　DRCR.RBE	1 (有効)	1 (有効)
	QSPIn_SSL ネゲート設定： DRCR.SSLE	転送終了毎にQSPIn_SSL ネゲート 0	アドレスが連続な限りQSPIn_SSL アサート保持し、前回転送のアドレスから不連続の場合は、QSPIn_SSL ネゲート 1
	リードデータバースト長： DRCR.RBURST[4:0]	4 データ長 (32 バイト) B'00011	4 データ長 (32 バイト) B'00011
	データリードビット幅： DRENR.DRDB[1:0]	1 ビット B'00	4 ビット B'10
	リードコマンド： DRCMR.CMD[7:0]	Read H'03	Quad I/O DT Read (4B address) H'EE
	コマンドイネーブル： DRENR.CDE	出力する 1	出力する 1
	コマンドビット幅： DRENR.CDB[1:0]	1 ビット B'00	1 ビット B'00
	オプショナルコマンドイネーブル： DRENR.OCDE	出力しない 0	出力しない 0

【注】 1　表5.4の「動作クロックの設定」および「SPIBSC クロックの選択」の項目を参照してください。
2　1 ビット幅以外のデータリード転送を行う場合には、CMNCR の IO0FV は B'11（QSPIn_IO0 の出力値を Hi-Z）に設定する必要があります。

表5.3　ブート起動用内蔵 ROM プログラムおよびローダプログラムの設定内容（2/3）

	項目	ブート起動用内蔵 ROM プログラム実行後	ローダ プログラム実行後
SPIBSC 設定	アドレスイネーブル： DRENR.ADE[3:0]	Address[23:0]を出力 B'0111	Address[31:0]を出力 B'1111
	アドレスビット幅： DRENR.ADB[1:0]	1 ビット B'00	4 ビット B'10
	オプションデータイネーブル： DRENR.OPDE[3:0]	出力しない B'0000	OPD3 を出力（注） B'1000
	オプションデータビット幅： DRENR.OPDB[1:0]	－	4 ビット B'10
	オプションデータ： DROPR.OPD3[7:0] DROPR.OPD2[7:0] DROPR.OPD1[7:0] DROPR.OPD0[7:0]	 － － － －	 H'00 － － －
	ダミーサイクルイネーブル： DRENR.DME	挿入しない 0	挿入する 1
	ダミーサイクル数： DRDMCR.DMCYC[4:0]	－	7 サイクル B'00110
	拡張アドレス： DREAR.EAC[2:0] DREAR.EAV[7:0]	外部アドレス[24:0]が有効 32MB の空間に直接アクセス可能 B'000 H'00	外部アドレス [27:0]が有効 256MB の空間に直接アクセス可能 B'011 H'00
	転送フォーマット： DRDRENR.HYPE[2:0] DRDRENR.ADDRE DRDRENR.OPDRE DRDRENR.DRDRE PHYOFFSET1.DDRTMG[1:0] PHYOFFSET2.OCTTMG[2:0]	アドレス、オプションデータ、データは SDR 転送、SPI フラッシュモード B'000 0 0 0 B'11 (SDR) B'100 (シリアルフラッシュ)	アドレス、オプションデータ、データは DDR 転送、SPI フラッシュモード B'000 1 1 1 B'10 (DDR) B'100 (シリアルフラッシュ)
	Octal-SPI フラッシュメモリ代替アライメント： PHYCNT.ALT_ALIGN PHYCNT.OCTA[1:0]	Octal-SPI フラッシュメモリ接続時の代替アライメントをサポートしない 0 B'00	Octal-SPI フラッシュメモリ接続時の代替アライメントをサポートしない 0 B'00
	Octal-SPI フラッシュメモリプロトコルモード： PHYCNT.OCT	Octal-SPI フラッシュメモリプロトコルモードを使用しない 0	Octal-SPI フラッシュメモリプロトコルモードを使用しない 0
	外部データストローブ： PHYCNT.EXDS	外部データストローブ信号を使用しない 0	外部データストローブ信号を使用しない 0
	デバイス選択： PHYCNT.PHYMEM[1:0]	SDR モードのシリアルフラッシュ B'00	DDR モードのシリアルフラッシュ B'01
	ハイスピード応答モード： PHYCNT.HS	ハイスピード応答モードを使用しない 0	ハイスピード応答モードを使用しない 0

【注】 MX25L51245GXDは、アドレスサイクルに続く Performance enhance indicator のサイクル期間に、ビット 7～4 とビット 3～0 をトグルさせるデータ（例えば、H'A5、H'5A、H'F0、H'0F など）が入力されると、Performance Enhance Mode に遷移します。RZ/A2Mの外部アドレス空間リードモードは Performance Enhance Mode のデータ転送に対応していませんので、サンプルコードでは QuadIO DT Read コマンド発行時に、OPD3 から H'00 を出力するように設定し、MX25L51245GXDが Performance Enhance Mode に遷移しないようにしています。

表5.4　ブート起動用内蔵 ROM プログラムおよびローダプログラムの設定内容（3/3）

	項目	ブート起動用内蔵 ROM プログラム実行後	ローダ プログラム実行後
端子の設定	端子電圧 PPOC.POCSEL0 PPOC.POC0	SPIBSC の端子電圧が 3.3V で動作する設定 1 1	SPIBSC の端子電圧が 3.3V で動作する設定 1 1
	駆動能力 PSPIBSC[31:0]	 H'0555_5555 (SPIBSC 関連端子の駆動能力が 8mA)	 H'0555_5555 (SPIBSC 関連端子の駆動能力が 8mA)
シリアルフラッシュメモリ設定	Status Register	変更なし (注 1)	Quad 動作 Enable QE=1
	Configuration Register	変更なし (注 1)	DC[1:0] = B'10 ODS[2:0] = B'110
その他	動作クロックの設定 クロックモード 1 で EXTAL から 24MHz 入力時	Iφ=132[MHz] Gφ=264[MHz] Bφ=132[MHz] P1φ=66[MHz] P0φ=33[MHz]	Iφ=528[MHz] Gφ=264[MHz] Bφ=132[MHz] P1φ=66[MHz] P0φ=33[MHz]
	SPIBSC クロックの選択 SCLKSEL.SPICR[1:0]	P0φ を選択 B'00	Bφ を選択 B'10
	CPU の例外処理ベクタのアドレス	ハイベクタ (H'FFFF_0000～)	ローベクタ (H'0000_0000～)
	CP15 ベクタベースアドレスレジスタ (VBAR)	－	H'2000_0000

【注】 1. RZ/A2Mのブートモード 3（シリアルフラッシュブート 3.3V 品）では、シリアルフラッシュメモリにリードコマンド（オペコード：H'03、アドレスビット：24 ビット、ダミーサイクル：出力しない）を発行するように SPIBSC のレジスタを設定します。このため、シリアルフラッシュメモリのレジスタ設定値が、シリアルフラッシュブート実行時に上記のリードコマンドを正常に受信できない設定となっている場合は、正常にブートできない可能性があります。

2. シリアルフラッシュメモリを接続してデータ転送を行う場合に、入力データを取り込むためのタイミング調整を行う必要があり、ローダプログラムで実施しています。タイミング調整時には、PHYCNT、PHYADJ1、PHYADJ2 のレジスタを設定し、これらのレジスタはブート起動用内蔵 ROM プログラムとローダプログラム実行後で値が変わります。タイミング調整の詳細は、「5.10.8 SPIBSCをDDR転送で使用時のタイミング調整」を参照してください。

5.1.4　アプリケーションプログラム

(1)　アプリケーションプログラムの動作

リセット解除後に、ブート起動用内蔵 ROM プログラム、ローダプログラムの順にプログラムが実行され、H'2001_0000 番地に配置されているアプリケーションプログラムのスタートアップ処理に分岐します。

スタートアップ処理では、スタックポインタ、MMU、FPU の設定を行い、セクションの初期化を行い、resetprg 関数に分岐します。

resetprg 関数では、RTC と USB 未使用チャネルの初期化処理を実行後、L1 キャッシュおよび L2 キャッシュの有効化と INTC の初期化を行い、割り込み処理高速化のために VBAR に大容量内蔵 RAM のアドレスを設定し、IRQ 割り込みおよび FIQ 割り込みを許可にし、main 関数をコールしています。

main 関数では、CPG、OSTM チャネル 0、SCIFA チャネル 4、GPIO の初期設定処理を行います。これら初期化処理により、シリアルインタフェースに接続されたホスト PC 上のターミナルに文字列（起動メッセージ）を出力し、OSTM のチャネル 0 をインターバルタイマモードに設定してタイマを起動します。500ms の周期で OSTM チャネル 0 の割り込みを発生させ、CPU ボード上の LED を割り込み処理により 500ms ごとに点灯および消灯を繰り返す処理を行います。

アプリケーションプログラムで実施している初期設定の詳細は、「RZ/A2M グループ 初期設定例」のアプリケーションノートを参照してください。

(2)　アプリケーションプログラム作成時の注意事項

アプリケーションプログラムは、ローダプログラムから分岐するアドレスに配置してください。なお、アプリケーションプログラムは、ローダプログラムとは異なるシリアルフラッシュメモリのセクタに配置してください。

RZ/A2M CPUボードに搭載されているMacronix社製シリアルフラッシュメモリ（MX25L51245GXD）のセクタサイズは 4KB です。サンプルコードでは、アプリケーションプログラムをセクタ 16 の H'2001_0000 番地に配置しています。

図5.2にサンプルコードのプログラム配置を示します。

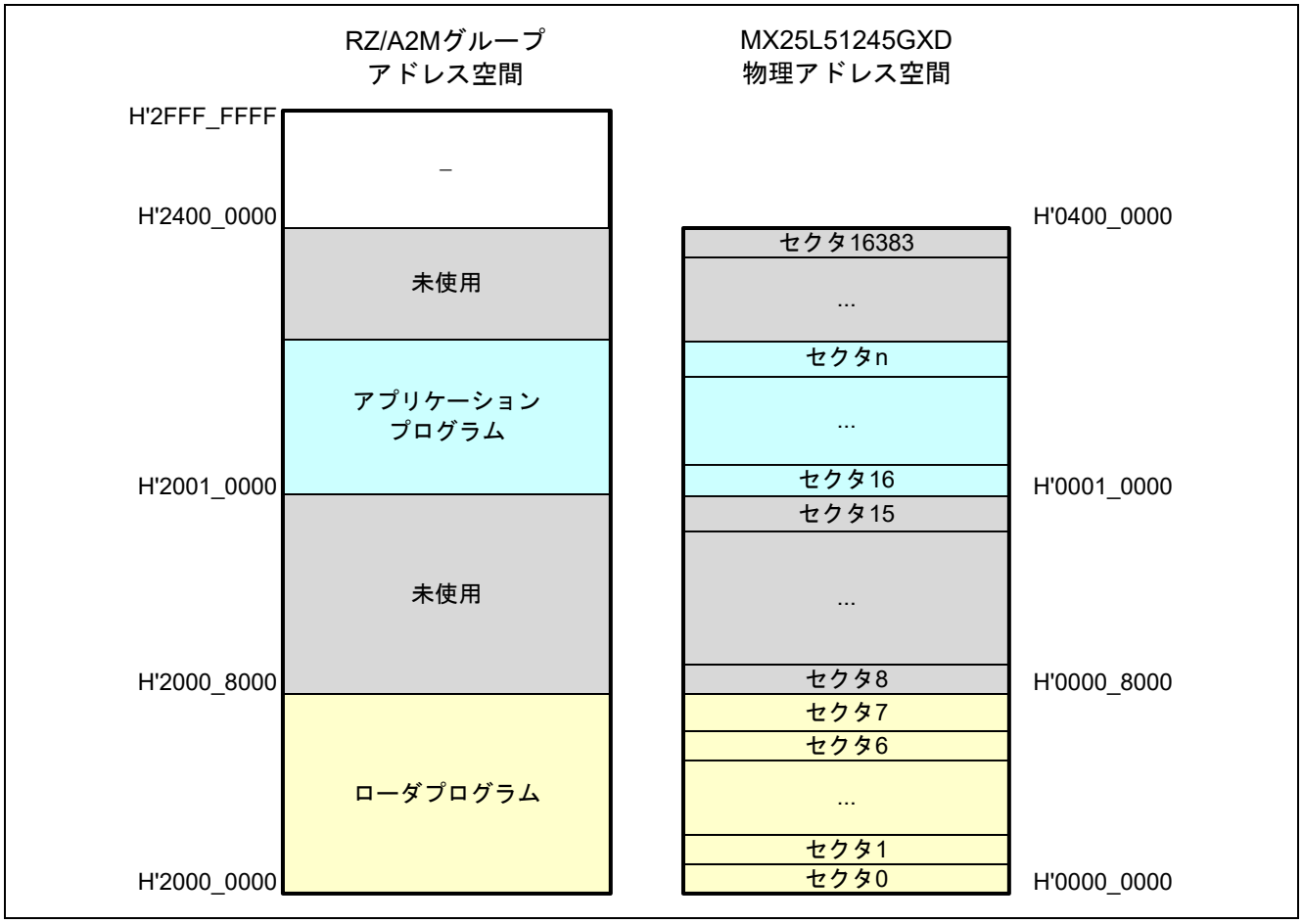


図5.2　サンプルコードのプログラム配置

アプリケーションプログラムの先頭アドレスは、以下の内容を変更することにより、アドレス配置の変更が可能です。

- ローダプログラムのプロジェクト
  アプリケーションプログラムの先頭アドレスへの分岐は、ローダプログラム（reset_handler.asm）にて行っています。「linker_script.ld」のリンカスクリプトのシンボル定義"__application_base_address"により、分岐先を指定してください。
- アプリケーションプログラムのプロジェクト
  アプリケーションプログラムの"VECTOR_TABLE"のセクションが、"__application_base_address"で指定したアドレスと一致するように配置アドレスを変更してください。

5.2　サンプルコード実行時の周辺機能の設定およびメモリ配置

5.2.1　周辺機能の設定

表5.5にサンプルコード実行時の周辺機能の設定内容を示します。

表5.5　周辺機能の設定内容

モジュール	設定内容
CPG	CPU クロック：PLL 回路の周波数に対して×1/2 倍に設定 内部バスクロック：PLL 回路の周波数に対して×1/8 倍に設定 周辺クロック 1（P1φ）：PLL 回路の周波数に対して×1/16 倍に設定 クロックモード 1（分周器 1：×1/2 倍、PLL 回路：×88 倍）で、入力クロックが 24MHz の場合に以下の周波数となるように設定 • CPU クロック（Iφ）：528MHz • 画像処理クロック（Gφ）：264MHz • 内部バスクロック（Bφ）：132MHz • 周辺クロック 1（P1φ）：66MHz • 周辺クロック 0（P0φ）：33MHz • QSPI0_SPCLK：66MHz（Bφ選択時） • CKIO クロック：132MHz（Bφ選択時）
SPIBSC	外部アドレス空間リードモードに設定し、CPU が SPI マルチ I/O バス空間に接続されたシリアルフラッシュメモリから、直接リードするための信号を生成するための設定
STB	保持用内蔵 RAM へのライト許可および周辺機能へのクロック供給 STBCR3, STBCR4, STBCR8 で OSTM0, OSTM2, SCIFA4, SPIBSC にクロックを供給
GPIO	PORT6、PORT9 の兼用端子機能を設定 • P6_0：LED の点灯および消灯 • P9_1：RxD4、P9_0：TxD4
OSTM	チャネル 0、チャネル 2 をインターバルタイマモードに設定 • チャネル 0 P1φ=66MHz の時に 500ms ごとに割り込み要求を発生するように、タイマカウントを設定。LED 点灯および消灯処理で使用。 • チャネル 2 P1φ=66MHz の時に 1ms ごとに割り込み要求を発生するように、タイマカウントを設定。OS Abstraction による時間管理に使用。
INTC	INTC の初期設定および OSTM チャネル 0 割り込み（割り込み ID が 88）ハンドラの登録、OSTM チャネル 2 割り込み（割り込み ID が 90）ハンドラの登録、SCIFA チャネル 4 割り込み（割り込み ID が 322, 323）ハンドラの登録と実行
SCIFA	チャネル 4 を調歩同期式モードに設定 • データ長：8 ビット • ストップビット長：1 ビット • パリティ：なし • データ転送方向：LSB ファースト転送 P1φ=66MHz の時に、クロックソースを分周なし、ボーレートジェネレータは倍速モード、ビットレートの 8 倍の基本クロックで動作するように設定。ビットレートが 115200bps となるように、ビットレート値に 71 を設定 （誤差：-0.53%）

5.2.2　メモリマップ

図5.3にRZ/A2Mグループのアドレス空間とRZ/A2M CPUボードのメモリマップを示します。

サンプルコードでは、ROM 領域を使用するコードおよびデータを SPI マルチ I/O バス空間に接続したシリアルフラッシュメモリに配置し、RAM 領域を使用するコードおよびデータを大容量内蔵 RAM に配置するようにしています。

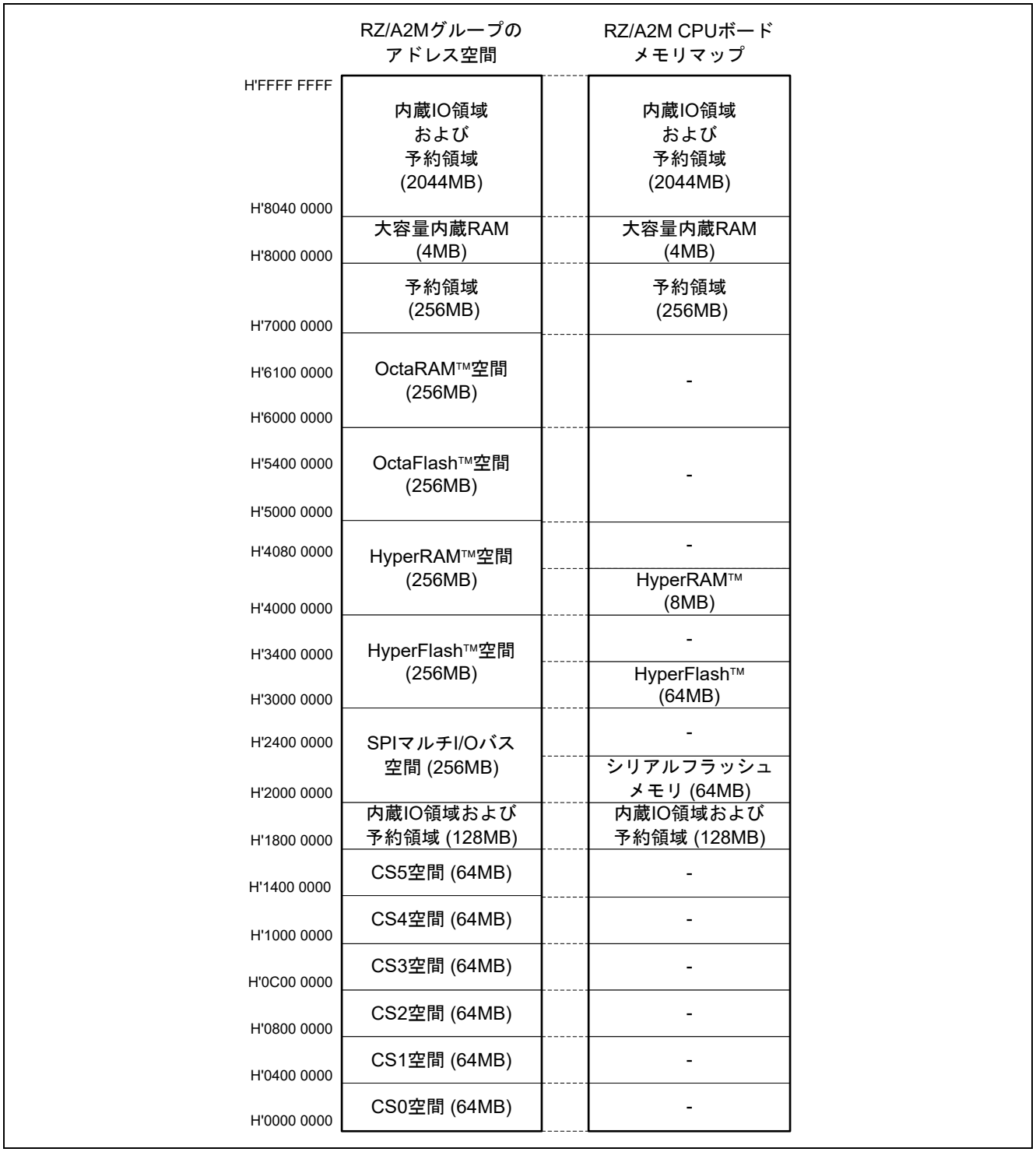


図5.3　メモリマップ

5.2.3　サンプルコードのセクション配置

表5.6にローダプログラムで使用するセクション名とオブジェクト名一覧を示します。

アプリケーションプログラムで使用するセクション配置については、「RZ/A2M グループ 初期設定例」のアプリケーションノートをご参照ください。

表5.6　ローダプログラムで使用するセクション名とオブジェクト名一覧

出力セクション名	入力セクション名 入力オブジェクト名	内容	ロード領域	実行領域
LOAD_MODULE1	VECTOR_TABLE	例外処理ベクタテーブル	FLASH	FLASH
LOAD_MODULE2	*/r_cpg/*.o (.text .rodata)	CPG の設定処理	FLASH	LRAM
	*/rza_io_regrw.o (.text .rodata)	IO レジスタアクセス処理		
	/r_spibsc/.o (.text .rodata)	SPIBSC の設定処理 (キャリブレーション用の定数データを除く)		
	*/hwsetup.o (.text .rodata)	HardwareSetup の設定処理		
	* (.data .data.*)	デフォルトの初期値ありデータ領域		
LOAD_MODULE3	RESET_HANDLER	リセット処理	FLASH	FLASH
	INIT_SECTION */sections.o	セクション初期化処理		
	* (.text .text.*)	デフォルトのコード領域		
	* (.rodata .rodata.*)	デフォルトの定数データ領域		
.data.memclk_setup	*/r_memclk_setup.o (.text .rodata .data)	メモリクロックの設定処理	FLASH	LRAM
	*/r_spibsc_setup.o (.text .rodata .data)	SPIBSC 用メモリクロックの設定処理		
	*/r_*_memclk_setup.o (.text .rodata .data)	各ドライバ用メモリクロックの設定処理		
.bss.memclk_setup	*/r_memclk_setup.o (.bss COMMON)	メモリクロックの設定処理の初期値なしデータ領域	–	LRAM
	*/r_spibsc_setup.o (.bss COMMON)	SPIBSC 用メモリクロックの設定処理の初期値なしデータ領域		
	*/r_*_memclk_setup.o (.bss COMMON)	各ドライバ用メモリクロックの設定処理の初期値なしデータ領域		
.stack	なし	SVC モードのスタック領域	–	LRAM
.bss	* (.bss .bss.*) * (COMMON)	デフォルトの初期値なしデータ領域	–	LRAM
.heap	なし	ヒープ領域	–	LRAM

【注】　表中のロード領域および実行領域において、FLASH はシリアルフラッシュメモリの領域を、LRAM は大容量内蔵 RAM の領域を表します。

5.3　使用割り込み一覧

ローダプログラムでは割り込みは使用していません。

アプリケーションプログラムで使用する割り込みについては、「RZ/A2M グループ 初期設定例」のアプリケーションノートをご参照ください。

5.4　データ型一覧

表5.7にサンプルコードで使用するデータ型一覧を示します。

表5.7　サンプルコードで使用するデータ型一覧

シンボル	内容
char_t	8 ビット文字
bool_t	論理型。値は true（1）、false（0）
int_t	高速な整数、符号あり、本サンプルコードでは 32 ビット整数。
int8_t	8 ビット整数、符号あり（標準ライブラリ stdint.h にて定義）
int16_t	16 ビット整数、符号あり（標準ライブラリ stdint.h にて定義）
int32_t	32 ビット整数、符号あり（標準ライブラリ stdint.h にて定義）
int64_t	64 ビット整数、符号あり（標準ライブラリ stdint.h にて定義）
uint8_t	8 ビット整数、符号なし（標準ライブラリ stdint.h にて定義）
uint16_t	16 ビット整数、符号なし（標準ライブラリ stdint.h にて定義）
uint32_t	32 ビット整数、符号なし（標準ライブラリ stdint.h にて定義）
uint64_t	64 ビット整数、符号なし（標準ライブラリ stdint.h にて定義）
float32_t	32 ビット浮動小数
float64_t	64 ビット浮動小数
float128_t	128 ビット浮動小数

5.5　定数一覧

表5.8および表5.9にローダプログラムで使用する定数を示します。

アプリケーションプログラムで使用する定数については、「RZ/A2M グループ 初期設定例」のアプリケーションノートをご参照ください。

表5.8　ローダプログラムで使用する定数（1/2）

定数名	設定値	内容
SPIBSC_SUCCESS	(0)	正常終了
SPIBSC_ERR_INVALID	(-1)	え終了
SPIBSC_PORT_VOLTAGE_3_3V	(0)	SPIBSC が使用する専用端子の動作電圧を 3.3V に設定
SPIBSC_PORT_VOLTAGE_1_8V	(1)	SPIBSC が使用する専用端子の動作電圧を 1.8V に設定
SPIBSC_FLASH_SPI	(0)	フラッシュメモリの種別をシリアルフラッシュメモリに設定
SPIBSC_MODE_MANUAL	(0)	SPIBSC 動作モードを手動モードに設定
SPIBSC_MODE_XIP	(1)	SPIBSC 動作モードを外部アドレス空間リードモードに設定
SPIBSC_CMNCR_BSZ_SINGLE	(0)	接続しているシリアルフラッシュメモリの個数を 1 個に設定
SPIBSC_CMNCR_BSZ_DUAL	(1)	接続しているシリアルフラッシュメモリの個数を 2 個に設定 (未サポート)
SPIBSC_QE_DISABLE	(0)	シリアルフラッシュメモリ（MX25L51245GXD）の Status Register の QE ビットを 0 に設定。
SPIBSC_QE_ENABLE	(1)	シリアルフラッシュメモリ（MX25L51245GXD）の Status Register の QE ビットを 1 に設定。
SPIBSC_1BIT_WIDTH	(0)	コマンド、オプショナルコマンド、アドレス、オプションデータ、転送データのビット幅を 1 ビットに設定
SPIBSC_4BIT_WIDTH	(2)	コマンド、オプショナルコマンド、アドレス、オプションデータ、転送データのビット幅を 4 ビットに設定
SPIBSC_OUTPUT_DISABLE	(0)	コマンド、オプショナルコマンド、アドレス、オプションデータ、ダミーサイクルを出力しない設定
SPIBSC_OUTPUT_ENABLE	(1)	コマンド、オプショナルコマンド、ダミーサイクルを出力する設定
SPIBSC_OUTPUT_ADDR_24	(0x07)	24 ビットのアドレスを出力
SPIBSC_OUTPUT_ADDR_32	(0x0f)	32 ビットのアドレスを出力
SPIBSC_OUTPUT_OPD_3	(0x08)	オプションデータ OPD3 を出力
SPIBSC_OUTPUT_OPD_32	(0x0c)	オプションデータ OPD3,OPD2 を出力
SPIBSC_OUTPUT_OPD_321	(0x0e)	オプションデータ OPD3,OPD2,OPD1 を出力
SPIBSC_OUTPUT_OPD_3210	(0x0f)	オプションデータ OPD3,OPD2,OPD1,OPD0 を出力

表5.9　ローダプログラムで使用する定数（2/2）

定数名	設定値	内容
SPIBSC_DUMMY_02CYC	(1)	ダミーサイクル数を 2 に設定
SPIBSC_DUMMY_03CYC	(2)	ダミーサイクル数を 3 に設定
SPIBSC_DUMMY_04CYC	(3)	ダミーサイクル数を 4 に設定
SPIBSC_DUMMY_05CYC	(4)	ダミーサイクル数を 5 に設定
SPIBSC_DUMMY_06CYC	(5)	ダミーサイクル数を 6 に設定
SPIBSC_DUMMY_07CYC	(6)	ダミーサイクル数を 7 に設定
SPIBSC_DUMMY_08CYC	(7)	ダミーサイクル数を 8 に設定
SPIBSC_DUMMY_09CYC	(8)	ダミーサイクル数を 9 に設定
SPIBSC_DUMMY_10CYC	(9)	ダミーサイクル数を 10 に設定
SPIBSC_DUMMY_11CYC	(10)	ダミーサイクル数を 11 に設定
SPIBSC_DUMMY_12CYC	(11)	ダミーサイクル数を 12 に設定
SPIBSC_DUMMY_13CYC	(12)	ダミーサイクル数を 13 に設定
SPIBSC_DUMMY_14CYC	(13)	ダミーサイクル数を 14 に設定
SPIBSC_DUMMY_15CYC	(14)	ダミーサイクル数を 15 に設定
SPIBSC_DUMMY_16CYC	(15)	ダミーサイクル数を 16 に設定
SPIBSC_DUMMY_17CYC	(16)	ダミーサイクル数を 17 に設定
SPIBSC_DUMMY_18CYC	(17)	ダミーサイクル数を 18 に設定
SPIBSC_DUMMY_19CYC	(18)	ダミーサイクル数を 19 に設定
SPIBSC_DUMMY_20CYC	(19)	ダミーサイクル数を 20 に設定
SPIBSC_DDR_TRANSFER	(1)	アドレス、オプションデータ、データの転送を DDR 転送に設定
SIPBSC_SDR_TRANSFER	(0)	アドレス、オプションデータ、データの転送を SDR 転送に設定
SPIBSC_NO_DATA	(0x00)	手動モード時にデータをリード/ライトしない設定
SPIBSC_READ_DATA	(0x01)	手動モード時にデータをリードする設定
SPIBSC_WRITE_DATA	(0x02)	手動モード時にデータをライトする設定
SPIBSC_TEST_PATTERN_EXPECTED_VALUE	(0xAA00FF55)	タイミング調整 OK 判定用に使用する期待値
SPIBSC_QSPI_IO_OUTPUT_0	(0x00)	QSPIn_IO 端子に出力する値を 0 にする
SPIBSC_QSPI_IO_OUTPUT_1	(0x01)	QSPIn_IO 端子に出力する値を 1 にする
SPIBSC_QSPI_IO_OUTPUT_PREVIOUS	(0x02)	QSPIn_IO 端子に出力する値を前回転送の最終ビットにする
SPIBSC_QSPI_IO_OUTPUT_HI_Z	(0x03)	QSPIn_IO 端子に出力する値を Hi-Z にする

5.6　構造体/共用体一覧

表5.10～表5.21にローダプログラムで使用する構造体を示します。

表5.10　SPIBSC レジスタ設定構造体（st_spibsc_config_t）

メンバ名	内容
uint8_t flash_type	接続するフラッシュメモリの種別を指定します。 SPIBSC_FLASH_SPI：SPI FLASH
uint8_t flash_num	シリアルフラッシュ接続数を指定します。 SPIBSC_CMNCR_BSZ_SINGLE： 1 個接続 SPIBSC_CMNCR_BSZ_DUAL： 2 個接続（未サポート）
uint8_t flash_port_voltage	SPIBSC の専用端子の電圧設定を指定します。 SPIBSC_PORT_VOLTAGE_3_3V：3.3V SPIBSC_PORT_VOLTAGE_1_8V：1.8V

表5.11　SPIBSC 外部アドレス空間リードモード設定構造体（st_spibsc_xip_config_t）（1/5）

メンバ名	内容
uint8_t command_name[20]	リードコマンドを識別する文字列 • 本メンバはレジスタ設定に影響しません。
uint8_t cmd	リードコマンド • SPI マルチ I/O バス空間へのリードを SPI 通信に変換する時にシリアルフラッシュメモリに出力するリードコマンドを設定します。 • 本メンバに設定した値をデータリードコマンド設定レジスタ（DRCMR）の CMD[7:0]に設定します。
uint8_t cmd_width	リードコマンドビット幅 • リードコマンド発行時のビット幅を設定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の CDB[1:0]に設定します。
uint8_t cmd_output_enable	リードコマンドイネーブル • リードコマンドを発行するかどうかを選択します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：発行しない SPIBSC_OUTPUT_ENABLE：発行する • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の CDE に設定します。
uint8_t ocmd	オプショナルコマンド • SPI マルチ I/O バス空間へのリードを SPI 通信に変換する時にシリアルフラッシュメモリに出力するオプショナルコマンドを設定します。 • 本メンバに設定した値をデータリードコマンド設定レジスタ（DRCMR）の OCMD[7:0]に設定します。
uint8_t ocmd_width	オプショナルコマンドビット幅 • オプショナルコマンド発行時のビット幅を設定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の OCDB[1:0]に設定します。
uint8_t ocmd_output_enable	オプショナルコマンドイネーブル • オプショナルコマンドを発行するかどうかを選択します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：発行しない SPIBSC_OUTPUT_ENABLE：発行する • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の OCDE に設定します。

表5.12　SPIBSC 外部アドレス空間リードモード設定構造体（st_spibsc_xip_config_t）（2/5）

メンバ名	内容
uint8_t addr_width	アドレスビット幅 • SPI マルチ I/O バス空間へのリードを SPI 通信に変換する時にシリアルフラッシュメモリに出力するアドレスのビット幅を設定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の ADB[1:0]に設定します。
uint8_t addr_output_enable	アドレスイネーブル • SPI マルチ I/O バス空間へのリードを SPI 通信に変換する時にシリアルフラッシュメモリに出力するアドレスを設定します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：出力しない SPIBSC_OUTPUT_ADDR_24：24 ビットのアドレスを出力 SPIBSC_OUTPUT_ADDR_32：32 ビットのアドレスを出力 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の ADE[3:0]に設定します。
uint8_t addr_ddr_enable	アドレス DDR イネーブル • 外部アドレス空間リードモード時に出力するアドレスの SDR/DDR 転送を選択します。 • 設定可能な値： SPIBSC_SDR_TRANSFER：SDR 転送 SPIBSC_DDR_TRANSFER：DDR 転送 • 本メンバに設定した値をデータリード DDR イネーブルレジスタ（DRDRENR）の ADDRE に設定します。
uint8_t reserve1	リザーブデータ • サンプルコードでは、本メンバを参照しません。
uint8_t reserve2	リザーブデータ • サンプルコードでは、本メンバを参照しません。
uint8_t opdata_width	オプションデータビット幅 • オプションデータ発行時のビット幅を設定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の OPDB[1:0]に設定します。
uint8_t opdata_output_enable	オプションデータイネーブル • オプションデータを発行するかどうかを選択します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：出力しない SPIBSC_OUTPUT_OPD_3：OPD3 を出力 SPIBSC_OUTPUT_OPD_32：OPD3,OPD2 を出力 SPIBSC_OUTPUT_OPD_321：OPD3,OPD2,OPD1 を出力 SPIBSC_OUTPUT_OPD_3210：OPD3,OPD2,OPD1,OPD0 を出力 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の OPDE[3:0]に設定します。

表5.13　SPIBSC 外部アドレス空間リードモード設定構造体（st_spibsc_xip_config_t）（3/5）

メンバ名	内容
uint8_t opdata_ddr_enable	オプションデータ DDR イネーブル • 外部アドレス空間リードモード時に出力するオプションデータの SDR/DDR 転送を選択します。 • 設定可能な値： SPIBSC_SDR_TRANSFER：SDR 転送 SPIBSC_DDR_TRANSFER：DDR 転送 • 本メンバに設定した値をデータリード DDR イネーブルレジスタ（DRDRENR）の OPDRE に設定します。
uint8_t opd3 uint8_t opd2 uint8_t opd1 uint8_t opd0	オプションデータ • SPI マルチ I/O バス空間へのリードを SPI 通信に変換する時にシリアルフラッシュメモリに出力するオプションデータを設定します。 • 本メンバに設定した値をデータリードオプション設定レジスタ（DROPR）の OPD3[7:0]、OPD2[7:0]、OPD1[7:0]、OPD0[7:0]に設定します。
uint8_t reserve3	リザーブデータ サンプルコードでは、本メンバを参照しません。
uint8_t dummy_cycle_enable	ダミーサイクルイネーブル • ダミーサイクルを挿入するかを選択します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：挿入しない SPIBSC_OUTPUT_ENABLE：挿入する • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の DME に設定します。
uint8_t dummy_cycle_count	ダミーサイクル数 • 挿入するダミーサイクル数を設定します。 • 設定可能な値： SPIBSC_DUMMY_02CYC～SPIBSC_DUMMY_20CYC • 本メンバに設定した値をデータリードダミーサイクル設定レジスタ（DRDMCR）の DMCYC[4:0]に設定します。
uint8_t data_width	データリードビット幅 • SPI マルチ I/O バス空間へのリードを SPI 通信に変換する時のシリアルフラッシュメモリのデータリードビット幅を設定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値をデータリードイネーブル設定レジスタ（DRENR）の DRDB[1:0]に設定します。
uint8_t data_ddr_enable	転送データ DDR イネーブル • 外部アドレス空間リードモード時に転送するデータの SDR/DDR 転送を選択します。 • 設定可能な値： SPIBSC_SDR_TRANSFER：SDR 転送 SPIBSC_DDR_TRANSFER：DDR 転送 • 本メンバに設定した値をデータリード DDR イネーブルレジスタ（DRDRENR）の DRDRE に設定します。

表5.14　SPIBSC 外部アドレス空間リードモード設定構造体（st_spibsc_xip_config_t）（4/5）

メンバ名	内容
uint8_t cmncr_moiio3	SSL ネゲート時の QSPIn_IO3 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO3[1:0]に設定します。
uint8_t cmncr_moiio2	SSL ネゲート時の QSPIn_IO2 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO2[1:0]に設定します。
uint8_t cmncr_moiio1	SSL ネゲート時の QSPIn_IO1 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO1[1:0]に設定します。
uint8_t cmncr_moiio0	SSL ネゲート時の QSPIn_IO0 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO0[1:0]に設定します。

表5.15　SPIBSC 外部アドレス空間リードモード設定構造体（st_spibsc_xip_config_t）（5/5）

メンバ名	内容
uint8_t cmncr_io3fv	1 ビット幅時の QSPIn_IO3 端子状態 • 転送を 1 ビット幅で行っている間の値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の IO3FV[1:0]に設定します。
uint8_t cmncr_io2fv	1 ビット幅時の QSPIn_IO2 端子状態 • 転送を 1 ビット幅で行っている間の値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の IO2FV[1:0]に設定します。
uint8_t cmncr_io0fv	1 ビット幅入力時の QSPIn_IO0 端子状態（注） • 入力を 1 ビット幅で行っている間の値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の IO0FV[1:0]に設定します。

【注】 1 ビット幅以外（data_width に"SPIBSC_1BIT_WIDTH"以外を設定）のデータリード転送を行う場合には、cmncr_io0fv には"SPIBSC_QSPI_IO_OUTPUT_HI_Z"を設定する必要があります。

表5.16　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(1/6)

メンバ名	内容
uint8_t command_name[20]	SPI コマンドを識別する文字列 • 本メンバはレジスタ設定に影響しません。
uint8_t cmd	コマンド • 手動モード時に出力するコマンドを設定します。 • 本メンバに設定した値を手動モードコマンド設定レジスタ（SMCMR）の CMD[7:0]に設定します。
uint8_t cmd_width	コマンドビット幅 • コマンド発行時のビット幅を設定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の CDB[1:0]に設定します。
uint8_t cmd_output_enable	コマンドイネーブル • コマンドを発行するかどうかを選択します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：発行しない SPIBSC_OUTPUT_ENABLE：発行する • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の CDE に設定します。
uint8_t ocmd	オプショナルコマンド • 手動モード時に出力するオプショナルコマンドを設定します。 • 本メンバに設定した値を手動モードコマンド設定レジスタ（SMCMR）の OCMD[7:0]に設定します。
uint8_t ocmd_width	オプショナルコマンドビット幅 • 手動モード時のオプショナルコマンドビット幅を指定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の OCDB[1:0]に設定します。
uint8_t ocmd_enable	オプショナルコマンドイネーブル • 手動モード時にオプショナルコマンドを出力するかを設定します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：出力しない SPIBSC_OUTPUT_ENABLE：出力する • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の OCDE に設定します。

表5.17　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(2/6)

メンバ名	内容
uint8_t addr_width	アドレスビット幅 • 手動モード時のアドレスビット幅を指定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の ADB[1:0]に設定します。
uint8_t addr_output_enable	アドレスイネーブル • 手動モード時にアドレスを出力するかを設定します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：出力しない SPIBSC_OUTPUT_ADDR_24：ADR[23:0]を出力 SPIBSC_OUTPUT_ADDR_32：ADR[31:0]を出力 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の ADE[3:0]に設定します。
uint8_t addr_sdr_ddr	アドレス DDR イネーブル • 手動モード時に出力するアドレスの SDR/DDR 転送を選択します。 • 設定可能な値： SPIBSC_SDR_TRANSFER：SDR 転送 SPIBSC_DDR_TRANSFER：DDR 転送 • 本メンバに設定した値を手動モード DDR イネーブルレジスタ（SMDRENR）の ADDRE に設定します。
uint8_t opdata_width	オプションデータビット幅 • 手動モード時のオプションデータビット幅を指定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の OPDB[1:0]に設定します。
uint8_t opdata_output_enable	オプションデータイネーブル • 手動モード時にオプションデータを出力するかを設定します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE　：出力しない SPIBSC_OUTPUT_OPD_3　：OPD3 を出力 SPIBSC_OUTPUT_OPD_32　：OPD3,OPD2 を出力 SPIBSC_OUTPUT_OPD_321　：OPD3,OPD2,OPD1 を出力 SPIBSC_OUTPUT_OPD_3210　：OPD3,OPD2,OPD1,OPD0 を出力 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の OPDE[3:0]に設定します。

表5.18　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(3/6)

メンバ名	内容
uint8_t opdata_ddr_enable	オプションデータ DDR イネーブル • 手動モード時に出力するオプションデータの SDR/DDR 転送を選択します。 • 設定可能な値： SPIBSC_SDR_TRANSFER：SDR 転送 SPIBSC_DDR_TRANSFER：DDR 転送 • 本メンバに設定した値を手動モード DDR イネーブルレジスタ（SMDRENR）の OPDRE に設定します。
uint8_t opd3 uint8_t opd2 uint8_t opd1 uint8_t opd0	オプションデータ • 手動モード時に出力するオプションデータを設定します。 • 本メンバに設定した値を手動モードオプション設定レジスタ（SMOPR）の OPD3[7:0]、OPD2[7:0]、OPD1[7:0]、OPD0[7:0]に設定します。
uint8_t reserve3	リザーブデータ サンプルコードでは、本メンバを参照しません。
uint8_t dummy_cycle_output_enable	ダミーサイクルイネーブル • 手動モード時にダミーサイクル挿入するかどうかを設定します。 • 設定可能な値： SPIBSC_OUTPUT_DISABLE：挿入しない SPIBSC_OUTPUT_ENABLE：挿入する • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の DME に設定します。
uint8_t dummy_cycle_count	ダミーサイクル数 • 挿入するダミーサイクル数を設定します。 • 設定可能な値： SPIBSC_DUMMY_02CYC～SPIBSC_DUMMY_20CYC • 本メンバに設定した値を手動モードダミーサイクル設定レジスタ（SMDMCR）の DMCYC[4:0]に設定します。
uint8_t transfer_data_width	転送データビット幅 • 手動モード時の転送データビット幅を指定します。 • 設定可能な値： SPIBSC_1BIT_WIDTH：1 ビット幅 SPIBSC_4BIT_WIDTH：4 ビット幅 • 本メンバに設定した値を手動モードイネーブル設定レジスタ（SMENR）の SPIDB[1:0]に設定します。
uint8_t transfer_data_sdr_ddr	転送データ DDR イネーブル • 手動モード時に転送するデータの SDR/DDR 転送を選択します。 • 設定可能な値： SPIBSC_SDR_TRANSFER：SDR 転送 SPIBSC_DDR_TRANSFER：DDR 転送 • 本メンバに設定した値を手動モード DDR イネーブルレジスタ（SMDRENR）の SPIDRE に設定します。

表5.19　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(4/6)

メンバ名	内容
uint8_t reserve1	リザーブデータ サンプルコードでは、本メンバを参照しません。
uint8_t reserve2	リザーブデータ サンプルコードでは、本メンバを参照しません。
uint8_t cmncr_moiio3	SSL ネゲート時の QSPIn_IO3 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO3[1:0]に設定します。
uint8_t cmncr_moiio2	SSL ネゲート時の QSPIn_IO2 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO2[1:0]に設定します。
uint8_t cmncr_moiio1	SSL ネゲート時の QSPIn_IO1 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO1[1:0]に設定します。

表5.20　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(5/6)

メンバ名	内容
uint8_t cmncr_moiio0	SSL ネゲート時の QSPIn_IO0 端子状態 • データ転送完了後のネゲート期間に出力する値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の MOIIO0[1:0]に設定します。
uint8_t cmncr_io3fv	1 ビット幅時の QSPIn_IO3 端子状態 • 転送を 1 ビット幅で行っている間の値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の IO3FV[1:0]に設定します。
uint8_t cmncr_io2fv	1 ビット幅時の QSPIn_IO2 端子状態 • 転送を 1 ビット幅で行っている間の値を選択します。 • 設定可能な値： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の IO2FV[1:0]に設定します。

表5.21　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(6/6)

メンバ名	内容
uint8_t cmncr_io0fv	1 ビット幅入力時の QSPIn_IO0 端子状態 • 入力を 1 ビット幅で行っている間の値を選択します。 • 設定可能な値（注）： SPIBSC_QSPI_IO_OUTPUT_0：出力値は 0 SPIBSC_QSPI_IO_OUTPUT_1：出力値は 1 SPIBSC_QSPI_IO_OUTPUT_PREVIOUS：前回転送の最終ビット（前回が Hi-Z のときは Hi-Z） SPIBSC_QSPI_IO_OUTPUT_HI_Z：出力値は Hi-Z • 本メンバに設定した値を共通コントロールレジスタ（CMNCR）の IO0FV[1:0]に設定します。

【注】 1 ビット幅以外（data_width に"SPIBSC_1BIT_WIDTH"以外を設定）のデータリード転送を行う場合には、cmncr_io0fv には"SPIBSC_QSPI_IO_OUTPUT_HI_Z"を設定する必要があります。

5.7　変数一覧

表5.22にローダプログラムで使用する変数一覧を示します。

表5.22　ローダプログラムで使用する変数一覧

変数名	説明	備考
st_spibsc_xip_config_t gs_read_table[0]	外部アドレス空間リードモード用 DDR リードコマンド用の設定テーブルデータ	表6.8参照
st_spibsc_xip_config_t gs_read_table[1]	外部アドレス空間リードモード用 SDR リードコマンド用の設定テーブルデータ	表6.9参照
st_spibsc_manual_mode_command_config_t gs_command_table[0]	手動モード用 READ STATUS コマンド設定テーブルデータ	表6.10参照
st_spibsc_manual_mode_command_config_t gs_command_table[1]	手動モード用 READ CONFIGURATION コマンド設定テーブルデータ	表6.11参照
st_spibsc_manual_mode_command_config_t gs_command_table[2]	手動モード用 WRITE ENABLE コマンド設定テーブルデータ	表6.12参照
st_spibsc_manual_mode_command_config_t gs_command_table[3]	手動モード用 WRITE STATUS コマンド設定テーブルデータ	表6.13参照

5.8　関数一覧

サンプルコードは、周辺機能を使用するためのインタフェース関数（API 関数）、ユーザシステムの用途に合わせてユーザで準備が必要なユーザ定義関数（API 関数からコールされる関数）、サンプルコードを動作させるために必要なサンプル関数から構成されています。

サンプルコードのローダプログラムの関数について、表5.23にサンプル関数一覧を、表5.24にAPI関数一覧を、表5.25にユーザ定義関数一覧を示します。

表5.23　サンプル関数一覧

関数名	概要
reset_handler	リセットハンドラ処理（アセンブラ関数）
INITSCT	ローダプログラムのセクション初期化（アセンブラ関数）
R_SC_HardwareSetup	ブートに使用するハードウェアの初期設定
r_memclk_setup	メモリクロックの設定処理

表5.24　API 関数一覧

関数名	概要
R_SPIBSC_Setup	SPIBSC とシリアルフラッシュメモリの初期設定
R_SPIBSC_Init	SPIBSC 初期設定
R_SPIBSC_ChangeMode	SPIBSC 動作モード設定
R_SPIBSC_SPICMDIssue	シリアルフラッシュメモリへの SPI コマンド発行（手動モード用）
R_SPIBSC_XipStopAccess	シリアルフラッシュメモリへのアクセス停止
R_SPIBSC_FlushReadCache	SPIBSC リードキャッシュのクリア
R_SPIBSC_AdjustPhy	SPIBSC のデータ取り込みタイミングのキャリブレーション
R_CPG_InitialiseHwIf	CPG の初期化処理

表5.25　ユーザ定義関数一覧

関数名	概要
Userdef_SPIBSC_SFLASH_SetMode	シリアルフラッシュメモリのレジスタの設定
Userdef_SPIBSC_SFLASH_ReadStatus	シリアルフラッシュメモリのステータスレジスタのリード
Userdef_SPIBSC_SFLASH_ReadConfig	シリアルフラッシュメモリのコンフィグレーションレジスタのリード
Userdef_SPIBSC_SFLASH_WriteStatus	シリアルフラッシュメモリのステータスレジスタおよびコンフィグレーションレジスタのライト
Userdef_SPIBSC_SFLASH_WriteEnable	シリアルフラッシュメモリのライト許可
Userdef_SPIBSC_SFLASH_WaitReady	シリアルフラッシュメモリのライト完了待ち
Userdef_PreHardwareSetup	SPIBSC 初期化前に必要なハードウェア初期化処理
Userdef_PostHardwareSetup	SPIBSC 初期化後に必要なハードウェア初期化処理

5.9　関数仕様

ローダプログラムの関数仕様を示します。

reset_handler	
概　要	ローダプログラムのリセットハンドラ
宣　言	reset_handler
説　明	ローダプログラムのエントリ関数です。
引　数	なし
リターン値	なし

INITSCT		
概　要	ローダプログラムのセクション初期化	
宣　言	void INITSCT(void)	
説　明	RAM 領域に転送する必要のある初期値ありデータ（RAM 領域で実行する必要のあるコードおよび定数データを含む）を ROM 領域から転送し、RAM 領域の初期値なしデータの初期化を行います。	
引　数	p_dtbl	: 初期値ありデータのセクション情報が格納された領域へのポインタ
	p_btbl	: 初期値なしデータのセクション情報が格納された領域へのポインタ
リターン値	なし	

R_SC_HardwareSetup	
概　要	ブートに使用するハードウェアの初期設定
宣　言	void R_SC_HardwareSetup(void)
説　明	使用するシリアルフラッシュメモリに最適な設定を行い、SPIBSC を外部アドレス空間リードモードに設定してシリアルフラッシュメモリへのアクセスを行います。 サンプルコードでは、R_SPIBSC_Setup 関数をコールして、シリアルフラッシュメモリMX25L51245GXDのレジスタ設定とMX25L51245GXDの仕様に合わせた SPIBSC の初期設定を行います。
引　数	なし
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

r_memclk_setup	
概　要	メモリクロックの設定処理
宣　言	void r_memclk_setup (void)
説　明	R_SC_HardwareSetup 関数の実行前に、メモリクロックの設定を行います。 サンプルコードでは、シリアルフラッシュメモリに SDR 転送でアクセスするために、SDR モードのタイミング調整を行い、R_SC_HardwareSetup 関数を大容量内蔵 RAM に転送する処理を高速に実行するため、SPICLK を P1φに切り替えます。
引　数	なし
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

R_SPIBSC_Setup	
概　要	SPIBSC とシリアルフラッシュメモリの初期設定
宣　言	void R_SPIBSC_Setup (uint32_t ddrsdr)
説　明	使用するシリアルフラッシュメモリに最適な設定を行い、SPIBSC を外部アドレス空間リードモードに設定してシリアルフラッシュメモリへのアクセスを行います。 サンプルコードでは、以下の設定を行っています。 • リードコマンドの変更 : H'03→H'EE • シリアルフラッシュメモリのレジスタの設定 ステータスレジスタ : QE ビットに 1 を設定 コンフィグレーションレジスタ : DC[1:0]および ODS[2:0]ビットの設定 （DC[1:0]ビットの設定値はリードコマンドに依存して異なります。「表6.5 MX25L51245GXDの最大動作周波数に対して必要なダミーサイクル数の一覧」を参照してください。） • QSPIn_SPCLK 動作周波数を変更 : P1φ/2→Bφ/2 • DDR 転送でアクセスする場合に、R_SPIBSC_AdjustPhy 関数をコールして DDR モードのタイミング調整処理を実施
引　数	ddrsdr : DDR モード/SDR モードの指定 HWSETUP_SPIBSC_USE_DDR : DDR モード HWSETUP_SPIBSC_USE_SDR : SDR モード
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

R_SPIBSC_Init

概　要	SPIBSC 初期設定
宣　言	e_spibsc_err_t R_SPIBSC_Init(const spibsc_config_t *p_spibsc_config_tbl)
説　明	引数 spibsc_config_tbl により、SPIBSC 関連レジスタの初期設定を行います。本関数終了時は、外部アドレス空間リードモードの設定になっています。 • SPIBSC 関連端子の駆動能力の設定 • SPIBSC のモジュールスタンバイの解除 • SPIBSC のクロックの設定 • SPIBSC 関連レジスタの設定
引　数	const spibsc_config_t *p_spibsc_config_tbl　: SPIBSC の初期設定テーブルへのポインタ
リターン値	SPIBSC_SUCCESS　: 正常終了
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

R_SPIBSC_ChangeMode

概　要	SPIBSC 動作モード設定	
宣　言	void R_SPIBSC_ChangeMode(uint8_t mode, uint8_t sdr_ddr, uint8_t table_no)	
説　明	引数 mode で指定された動作モードにより、引数 sdr_ddr および table_no で指定された転送フォーマットでシリアルフラッシュメモリにアクセスするように設定します。	
引　数	uint8_t mode	: 動作モード SPIBSC_MODE_MANUAL : 手動モード SPIBSC_MODE_XIP : 外部アドレス空間リードモード
	uint8_t sdr_ddr	: 転送フォーマット SPIBSC_DDR_TRANSFER : DDR 転送 SPIBSC_SDR_TRANSFER : SDR 転送
	uint8_t table_no	: 外部アドレス空間リードモード用のコマンド設定テーブル番号 0 :コマンド設定テーブル 0 を選択（DDR リードコマンド） 1 :コマンド設定テーブル 1 を選択（SDR リードコマンド）
リターン値	なし	
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。	

R_SPIBSC_SPICMDIssue

概　要	シリアルフラッシュメモリへの SPI コマンド発行（手動モード用）
宣　言	spibsc_err_t R_SPIBSC_SPICMDIssue(uint8_t table_no, uint32_t addr, uint8_t *write_buff, uint32_t write_size, uint8_t *read_buff, uint32_t read_size)
説　明	引数 table_no で指定した SPI コマンド発行用のコンフィグレーションテーブルを使用して、SPI コマンドを発行します。 table_no の内容にしたがって、ライトコマンド発行時は、引数 addr で指定されたアドレスから引数 write_size で指定されたバイト数分、引数*write_buff に格納されたデータをライトします。リードコマンド発行時は、引数 addr で指定されたアドレスから引数 read_size で指定されたバイト数分のデータをリードし、引数*read_buff で指定された領域に格納します。
引　数	uint8_t table_no　：使用するコマンドに対する設定情報が格納されたテーブル uint32_t addr　：アドレス uint8_t * write_buff　：ライトバッファのポインタ uint32_t write_size　：ライトするバイト数 uint8_t * read_buff　：リードバッファのポインタ uint32_t read_size　：リードするバイト数
リターン値	SPIBSC_SUCCESS　：正常終了
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

R_SPIBSC_XipStopAccess

概　要	シリアルフラッシュメモリへのアクセス停止
宣　言	void R_SPIBSC_XipStopAccess(void)
説　明	外部アドレス空間リードモードで QSPIn_SSL をネゲートし、シリアルフラッシュメモリへのアクセスを停止します。 本関数は、QSPIn_SSL がネゲートされるのを待ってからリターンします。 SPIBSC 関連レジスタの設定時に、シリアルフラッシュメモリへのアクセスが発生することがないように本関数をコールします。
引　数	なし
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

R_SPIBSC_FlushReadCache

概　要	SPIBSC リードキャッシュのクリア
宣　言	void R_SPIBSC_FlushReadCache(void)
説　明	SPIBSC のリードキャッシュをクリアします。
引　数	なし
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

R_SPIBSC_AdjustPhy	
概　要	SPIBSC の DDR 転送のデータ取り込みタイミングのキャリブレーション
宣　言	e_spibsc_err_t R_SPIBSC_AdjustPhy(void)
説　明	シリアルフラッシュメモリからデータを取り込むタイミングのキャリブレーションを行います。 シリアルフラッシュメモリに格納された 4 バイトのデータが正しくリードできるように入力データの取り込みタイミングの調整を行います。4 バイトのデータは、g_spibsc_test_pattern のシンボルで定義されたアドレスに格納しています。 本関数は、シリアルフラッシュメモリに DDR 転送でアクセスする前に必ずコールしてください。
引　数	なし
リターン値	SPIBSC_SUCCESS　：正常終了 SPIBSC_ERR_INVALID：異常終了
注意事項	• 本関数は、SPIBSC にシリアルフラッシュメモリ以外のデバイスが接続された場合にはコールしないでください。 • 本関数で使用している g_spibsc_test_pattern の定数データ領域は、シリアルフラッシュメモリに配置する必要があります。 • 本関数は、シリアルフラッシュメモリに配置して実行することができません。本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。 • 本関数は、手動モードに設定している状態で使用してください。

R_CPG_InitialiseHwIf	
概　要	CPG の初期化処理
宣　言	int_t R_CPG_InitialiseHwIf(void)
説　明	r_cpg_drv_sc_cfg.h の CPG コンフィグレーションデータを使用して、CPG レジスタ（FRQCR、CKIOSEL、SCLKSEL）の設定処理を行います。 本サンプルコードでは、「表5.4　ブート起動用内蔵ROMプログラムおよびローダプログラムの設定内容（3/3）」の「動作クロックの設定」および「SPIBSC クロックの選択」に記載されている動作周波数となるように CPG の設定を行います。
引　数	なし　　なし
リターン値	DRV_SUCCESS　　：正常終了 DRV_ERROR　：r_cpg_drv_sc_cfg.h の CPG コンフィグレーションデータが不正
注意事項	本関数は、大容量内蔵 RAM に配置する必要があります。

Userdef_SPIBSC_SFLASH_SetMode	
概　要	シリアルフラッシュメモリのレジスタの設定
宣　言	void Userdef_SPIBSC_SFLASH_SetMode(uint8_t mode, uint8_t sdr_ddr)
説　明	使用するシリアルフラッシュメモリの仕様に合わせて、引数 mode で指定されたビット幅のアクセスで、引数 sdr_ddr で指定された転送フォーマットでアクセスできるように、シリアルフラッシュメモリのレジスタを設定する処理を実装してください。 サンプルコードでは、MX25L51245GXDのレジスタを設定し、4 ビット幅で DDR 転送でアクセスする設定を行っています。
引　数	uint8_t mode　：動作モード SPIBSC_QE_DISABLE: single モード（ビット幅 1 ビット） SPIBSC_QE_ENABLE: quad モード（ビット幅 4 ビット） uint8_t sdr_ddr　：転送フォーマット SPIBSC_DDR_TRANSFER : DDR 転送 SPIBSC_SDR_TRANSFER : SDR 転送
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_SPIBSC_SFLASH_ReadStatus	
概　要	シリアルフラッシュメモリのステータスレジスタのリード
宣　言	void Userdef_SPIBSC_SFLASH_ReadStatus(uint8_t *p_status)
説　明	使用するシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのステータスレジスタをリードして、引数*p_status にリードしたデータを格納する処理を実装してください。 サンプルコードでは、MX25L51245GXDの Status Register をリードする処理を行っています。
引　数	uint8_t *p_status　：ステータスレジスタからリードした値
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_SPIBSC_SFLASH_ReadConfig	
概　要	シリアルフラッシュメモリのコンフィグレーションレジスタのリード
宣　言	void Userdef_SPIBSC_SFLASH_ReadConfig(uint8_t *p_config)
説　明	使用するシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのコンフィグレーションレジスタをリードして、引数*p_config にリードしたデータを格納する処理を実装してください。 サンプルコードでは、MX25L51245GXDの Configuration Register をリードする処理を行っています。
引　数	uint8_t *p_config　：コンフィグレーションレジスタからリードした値
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_SPIBSC_SFLASH_WriteStatus	
概　要	シリアルフラッシュメモリのステータスレジスタおよびコンフィグレーションレジスタのライト
宣　言	void Userdef_SPIBSC_SFLASH_WriteStatus(uint8_t *p_status, uint8_t *p_config)
説　明	使用するシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのステータスレジスタおよびコンフィグレーションレジスタに、引数*p_status および引数*p_config で指定した値を設定する処理を実装してください。 サンプルコードでは、MX25L51245GXDの Status Register および Configuration Register にライトする処理を行っています。
引　数	uint8_t *p_status　　: status レジスタへの設定値 uint8_t *p_config　　: config レジスタへの設定値
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_SPIBSC_SFLASH_WriteEnable	
概　要	シリアルフラッシュメモリのライト許可
宣　言	void Userdef_SPIBSC_SFLASH_WriteEnable(void)
説　明	使用するシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリへのライトを許可する処理を実装しください。 サンプルコードでは、MX25L51245GXDに"Write Enable(WREN)"コマンドを発行する処理を行っています。
引　数	なし
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_SPIBSC_SFLASH_WaitReady	
概　要	シリアルフラッシュメモリのライト完了待ち
宣　言	void Userdef_SPIBSC_SFLASH_WaitReady(void)
説　明	使用するシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのライト完了を待つ処理を実装してください。 サンプルコードでは、MX25L51245GXDに"Read Status Register(RDSR)"コマンドを発行し、Status Register の内容を参照することでライト完了を待つ処理を行っています。
引　数	なし
リターン値	なし
注意事項	本関数は、シリアルフラッシュメモリに配置して実行することができません。 本関数は、シリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_PreHardwareSetup	
概　要	SPIBSC 初期化前に必要なハードウェア初期化処理
宣　言	void Userdef_PretHardwareSetup (void)
説　明	SPIBSC の初期化前に実行する必要がある、ハードウェア初期化処理を記述するためのユーザ定義可能関数です。 本サンプルコードでは、何も処理を行いません。
引　数	なし
リターン値	なし
注意事項	本関数にシリアルフラッシュメモリで実行することができない処理を記述する場合、本関数はシリアルフラッシュメモリ以外の領域に配置する必要があります。

Userdef_PostHardwareSetup	
概　要	SPIBSC 初期化後に必要なハードウェア初期化処理
宣　言	void Userdef_PostHardwareSetup (void)
説　明	SPIBSC の初期化後に実行する必要がある、ハードウェア初期化処理を記述するためのユーザ定義可能関数です。R_SC_HardwareSetup 関数の終端でコールされます。 ローダプログラムでは、R_CPG_InitialiseHwIf 関数をコールして、EXTAL から 24[MHz]が入力される前提で、以下のクロック設定を行います。 • Iϕ = 528[MHz], Gϕ = 264[MHz], Bϕ = 132[MHz], P1ϕ = 66[MHz], P0ϕ = 33[MHz], QSPI0_SPCLK = 66[MHz]
引　数	なし
リターン値	なし
注意事項	本関数にシリアルフラッシュメモリで実行することができない処理を記述する場合、本関数はシリアルフラッシュメモリ以外の領域に配置する必要があります。

5.10 ローダプログラムのフローチャート

5.10.1 ローダプログラム（全体）

図5.4にローダプログラム（全体）のフローチャートを示します。

【注】 シリアルフラッシュメモリのレジスタ設定時に、SPIBSC を外部アドレス空間リードモードから手動モードに変更し、レジスタ設定後に手動モードから外部アドレス空間リードモードに戻す処理を行います。

図5.4　ローダプログラム（全体）のフローチャート

5.10.2 メモリクロックの設定処理

SPIBSC 初期設定およびシリアルフラッシュメモリのレジスタ設定を含むハードウェア設定処理のためのセクション初期化を高速化するために、シリアルフラッシュメモリに供給するクロック（QSPIn_SPCLK）の設定を行います。

メモリクロックの設定処理は、SPI マルチ I/O バス空間に配置されたプログラムでは実行できないため、大容量内蔵 RAM に展開し、大容量内蔵 RAM で実行します。

図5.5にメモリクロックの設定処理のフローチャートを示します。

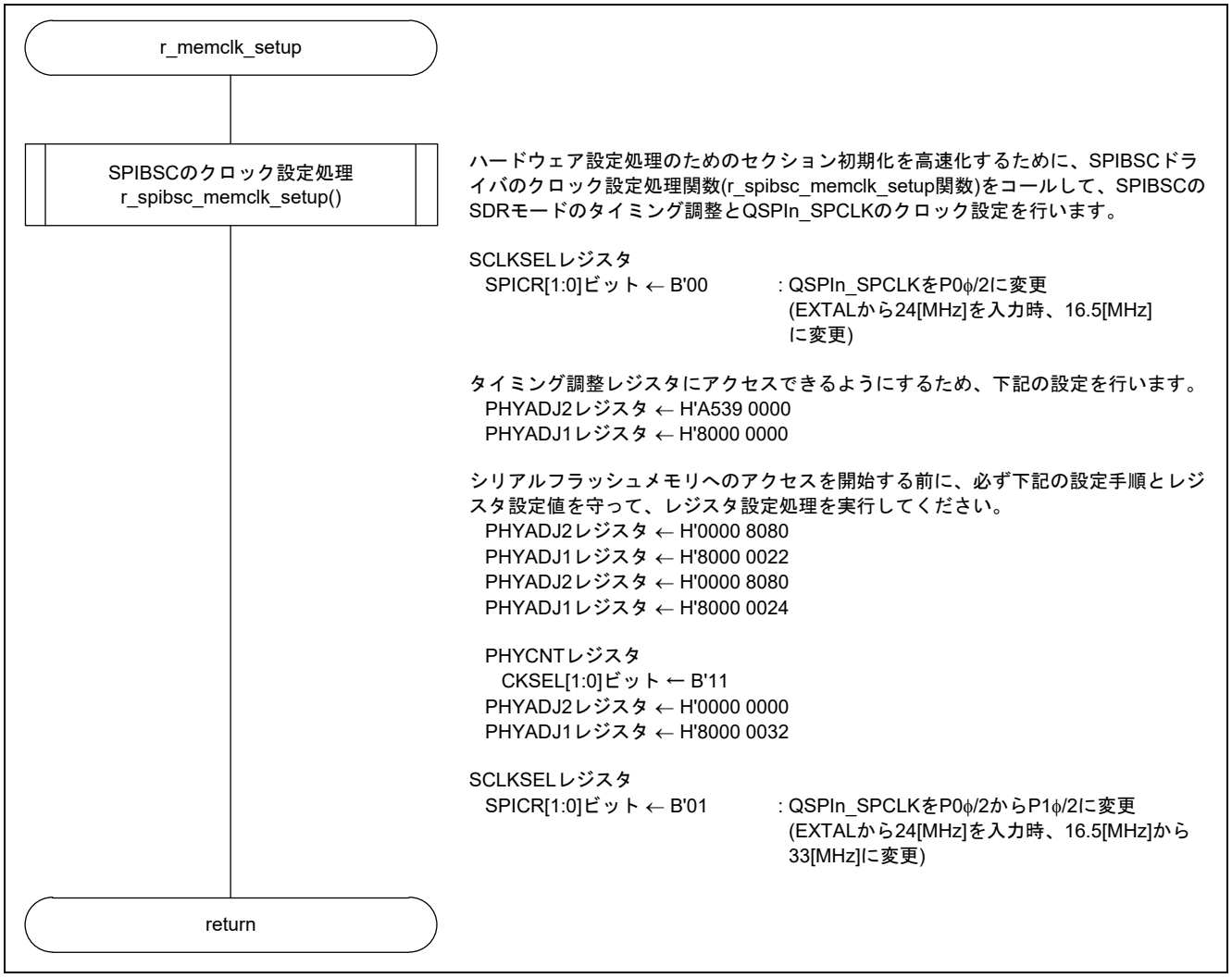


図5.5　メモリクロックの設定処理のフローチャート

5.10.3 ブートに使用するハードウェアの初期設定

ハードウェアの初期設定を行うための関数です。ローダプログラムでは高速にシリアルフラッシュメモリにアクセスができるように、R_SPIBSC_Setup 関数をコールして、SPIBSC およびシリアルフラッシュメモリの設定を行います。

SPIBSC のレジスタおよびシリアルフラッシュメモリのレジスタを設定する処理は、SPI マルチ I/O バス空間に配置されたプログラムでは実行できないため、大容量内蔵 RAM に展開し、大容量内蔵 RAM で実行します。

図5.6にハードウェア初期設定処理のフローチャートを示します。

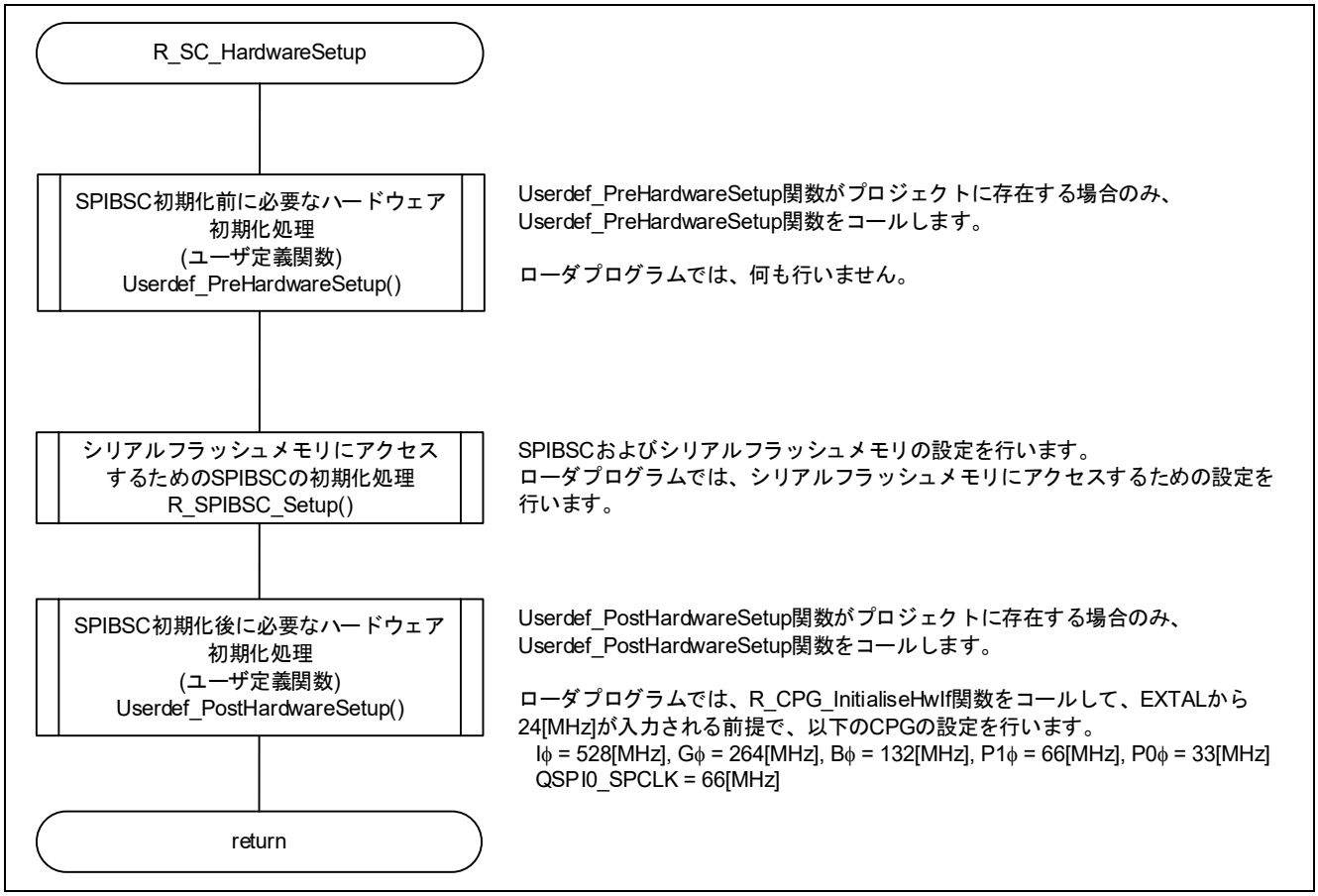


図5.6　ハードウェア初期設定処理のフローチャート

5.10.4 SPIBSC とシリアルフラッシュメモリの初期設定

ローダプログラムでは、高速にシリアルフラッシュメモリにアクセスできるように、シリアルフラッシュメモリのレジスタ（Status Register の QE ビット、Configuration Register の DC[1:0]ビット、PBE ビットおよび ODS[2:0]ビット）を設定し、バス幅を 1 ビットから 4 ビットに変更します。シリアルフラッシュメモリのレジスタ設定後、SPIBSC を外部アドレス空間リードモードで使用する場合にシリアルフラッシュメモリに発行するリードコマンドを"Quad I/O DT Read (4B address)"(H'EE)に設定し、QSPIn_SPCLK の動作周波数を Bφ/2 に変更します。

ローダプログラムの処理は、SPIBSC のレジスタを変更するため、SPI マルチ I/O バス空間に配置されたプログラムでは実行できないため、大容量内蔵 RAM に展開し、大容量内蔵 RAM 上で実行します。

図5.7にローダプログラムのフローチャートを、図5.8～図5.20に使用する関数のフローチャートを示します。

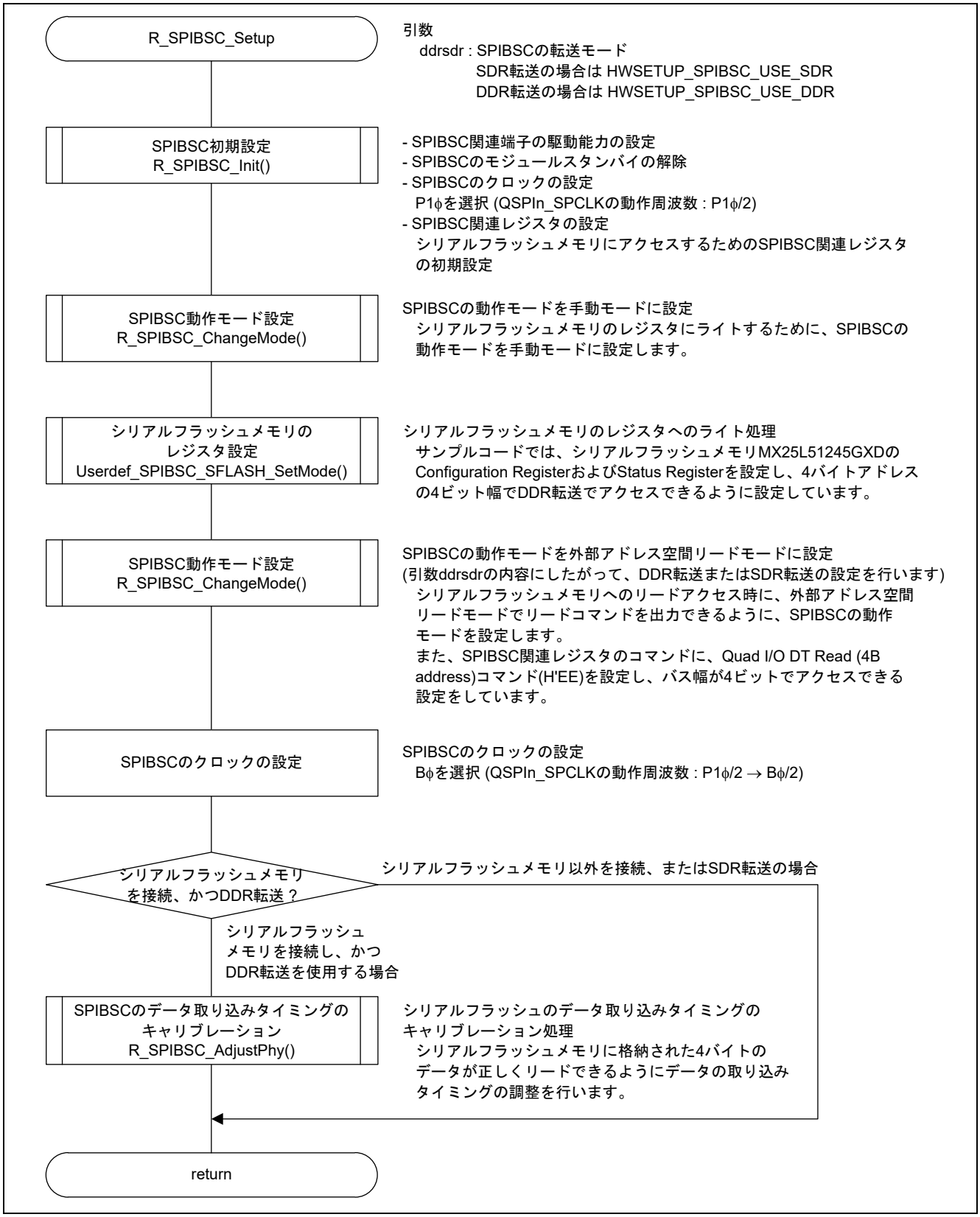


図5.7　SPIBSC とシリアルフラッシュメモリの初期設定のフローチャート

5.10.5 SPIBSC 初期設定

図5.8および図5.9に SPIBSC 初期設定フローを示します。

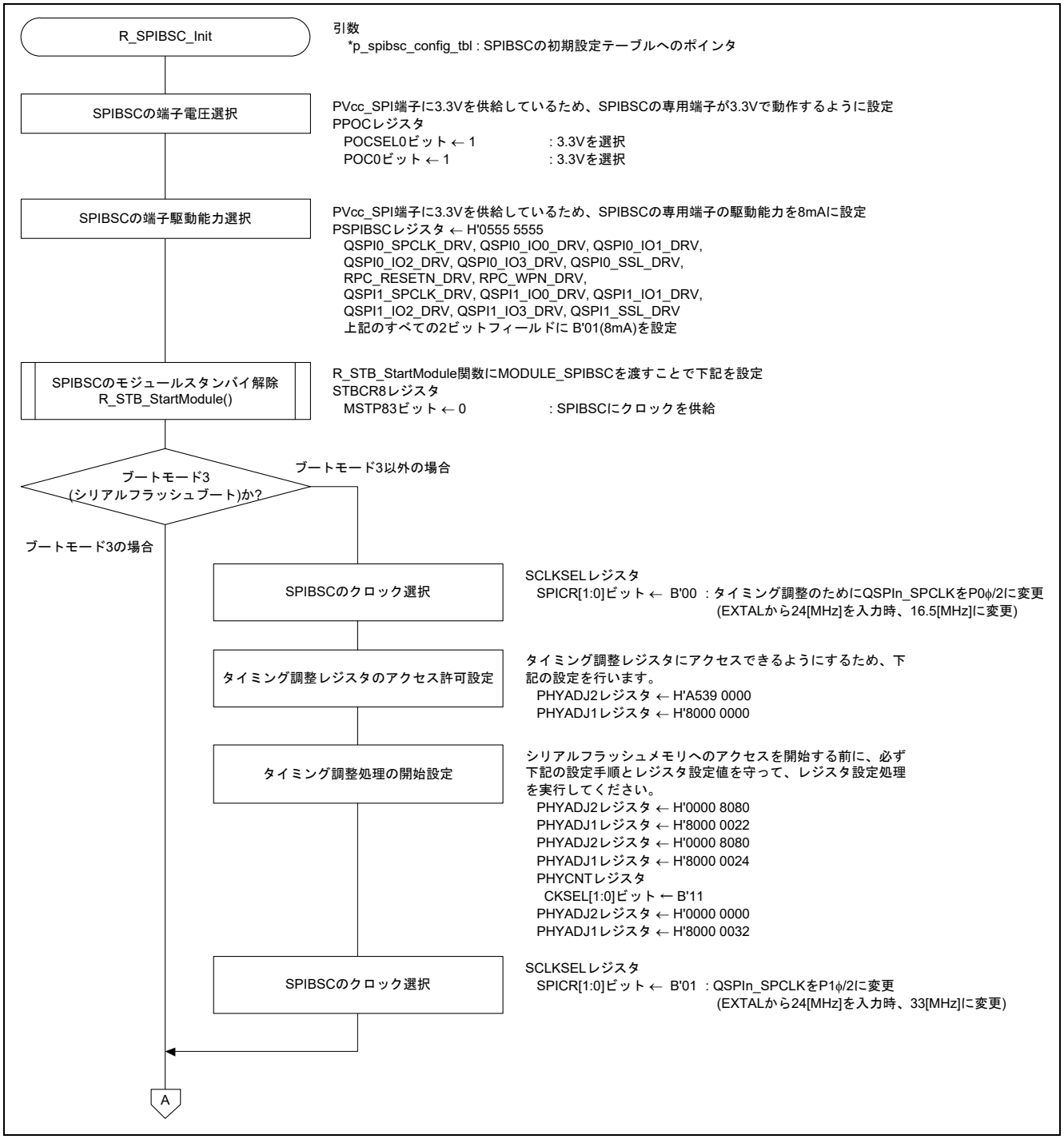


図5.8　SPIBSC 初期設定のフローチャート（1/2）

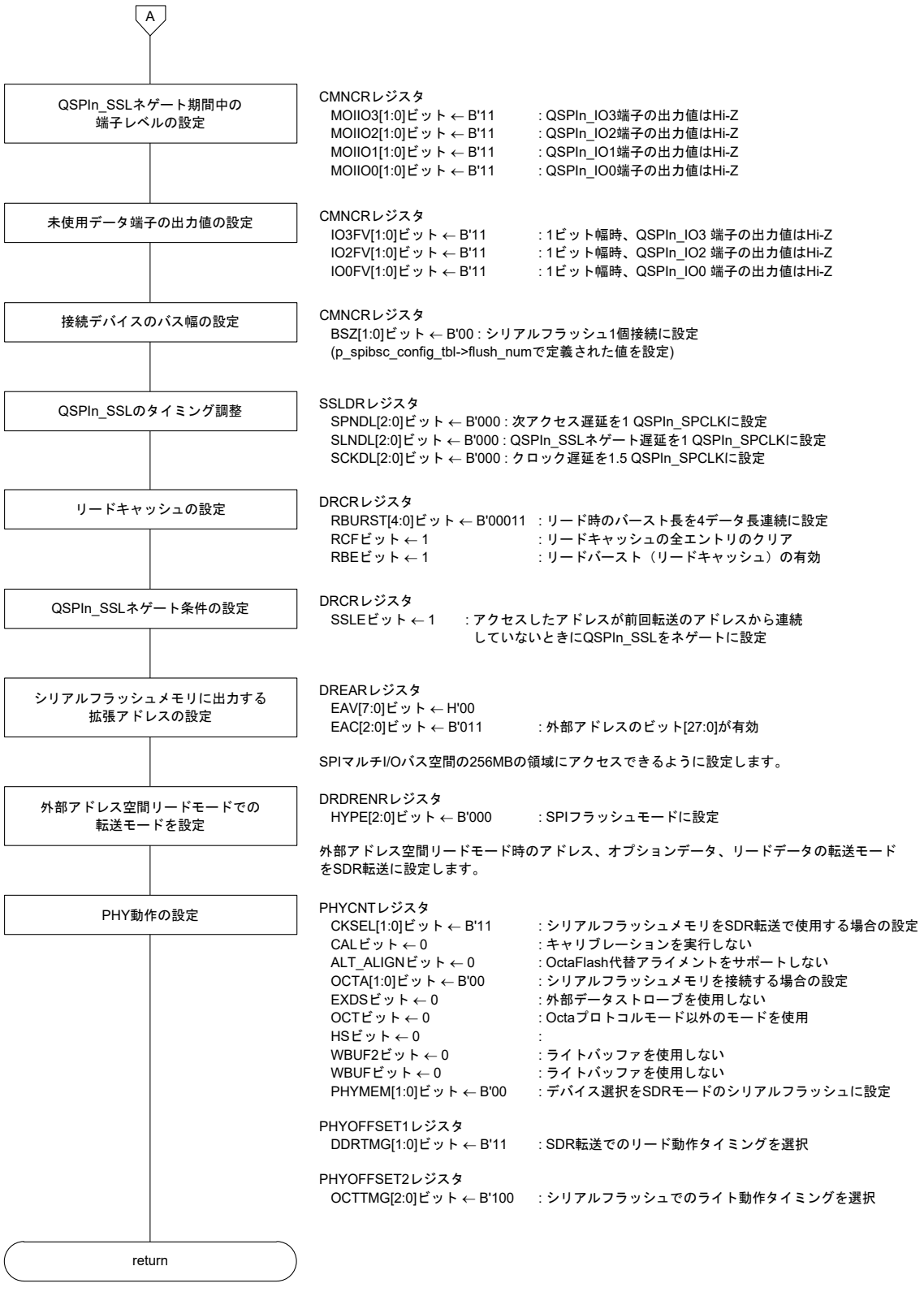


図5.9　SPIBSC 初期設定のフローチャート（2/2）

5.10.6 SPIBSC 動作モード設定

図5.10～図5.12に SPIBSC 動作モード設定を示します。

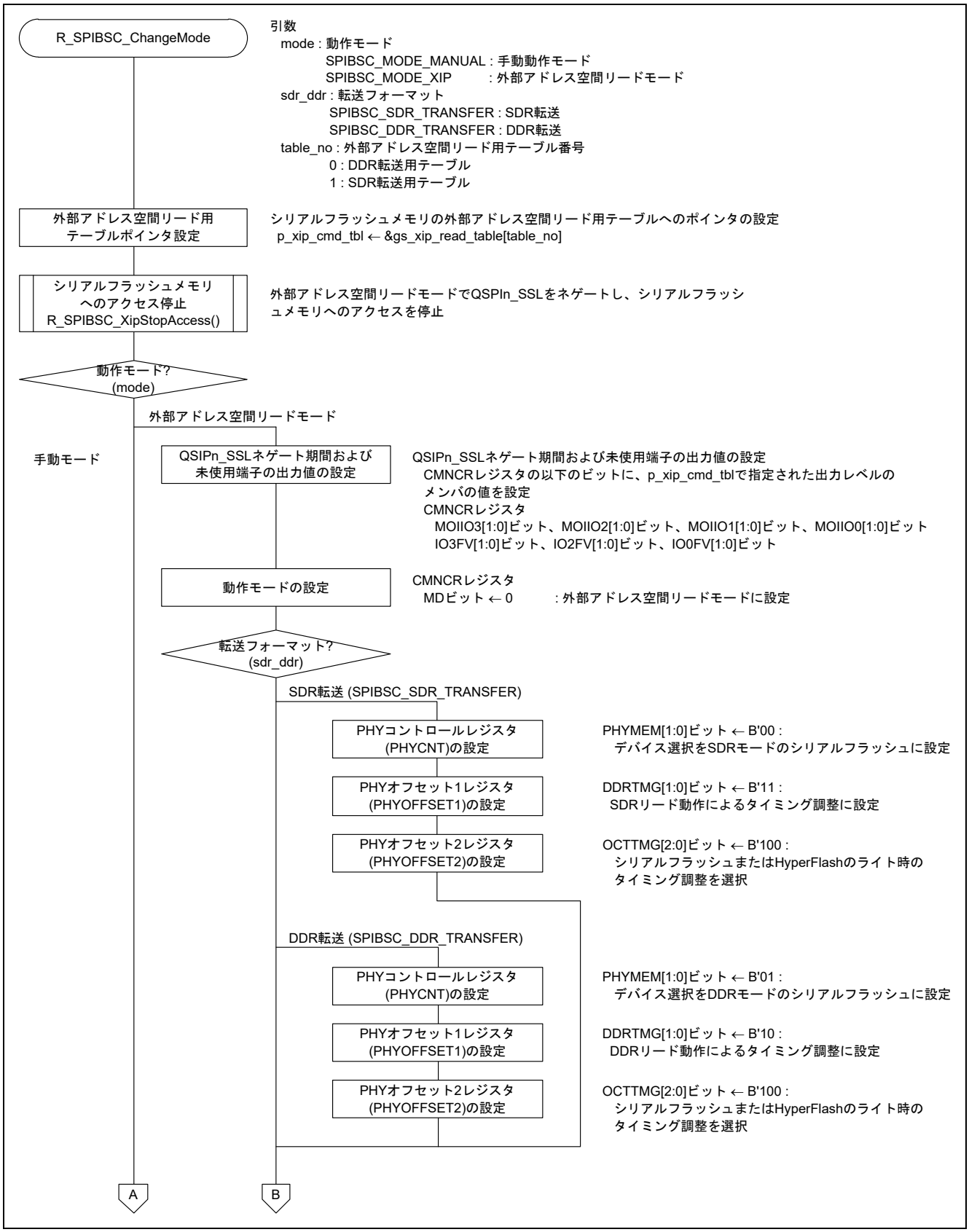


図5.10　SPIBSC 動作モード設定のフローチャート（1/3）

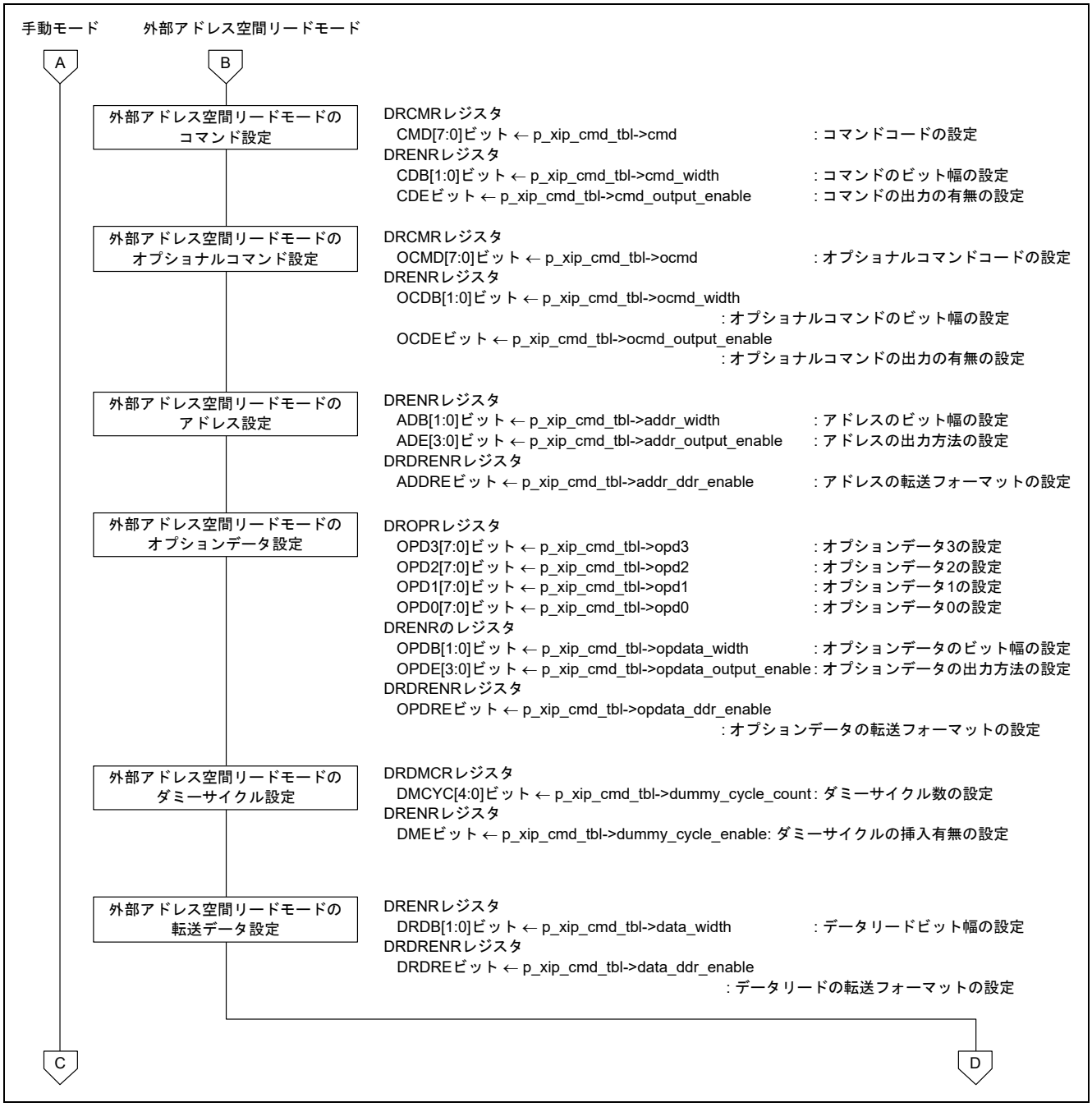


図5.11　SPIBSC 動作モード設定のフローチャート（2/3）

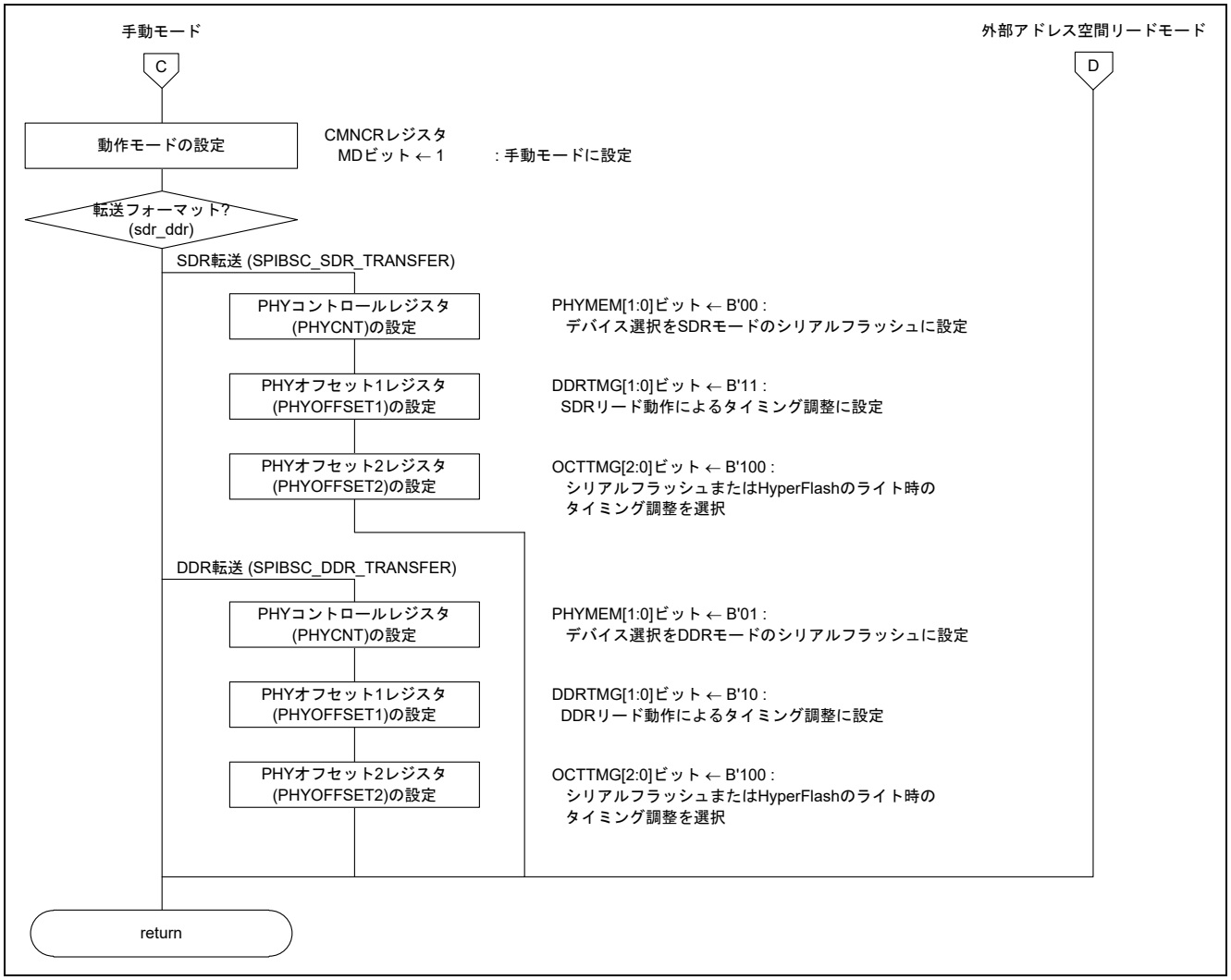


図5.12　SPIBSC 動作モード設定のフローチャート（3/3）

5.10.7 シリアルフラッシュメモリへの SPI コマンド発行

図5.13～図5.16にシリアルフラッシュメモリへの SPI コマンド発行のフローチャートを示します。本関数は手動モード時に使用してください。

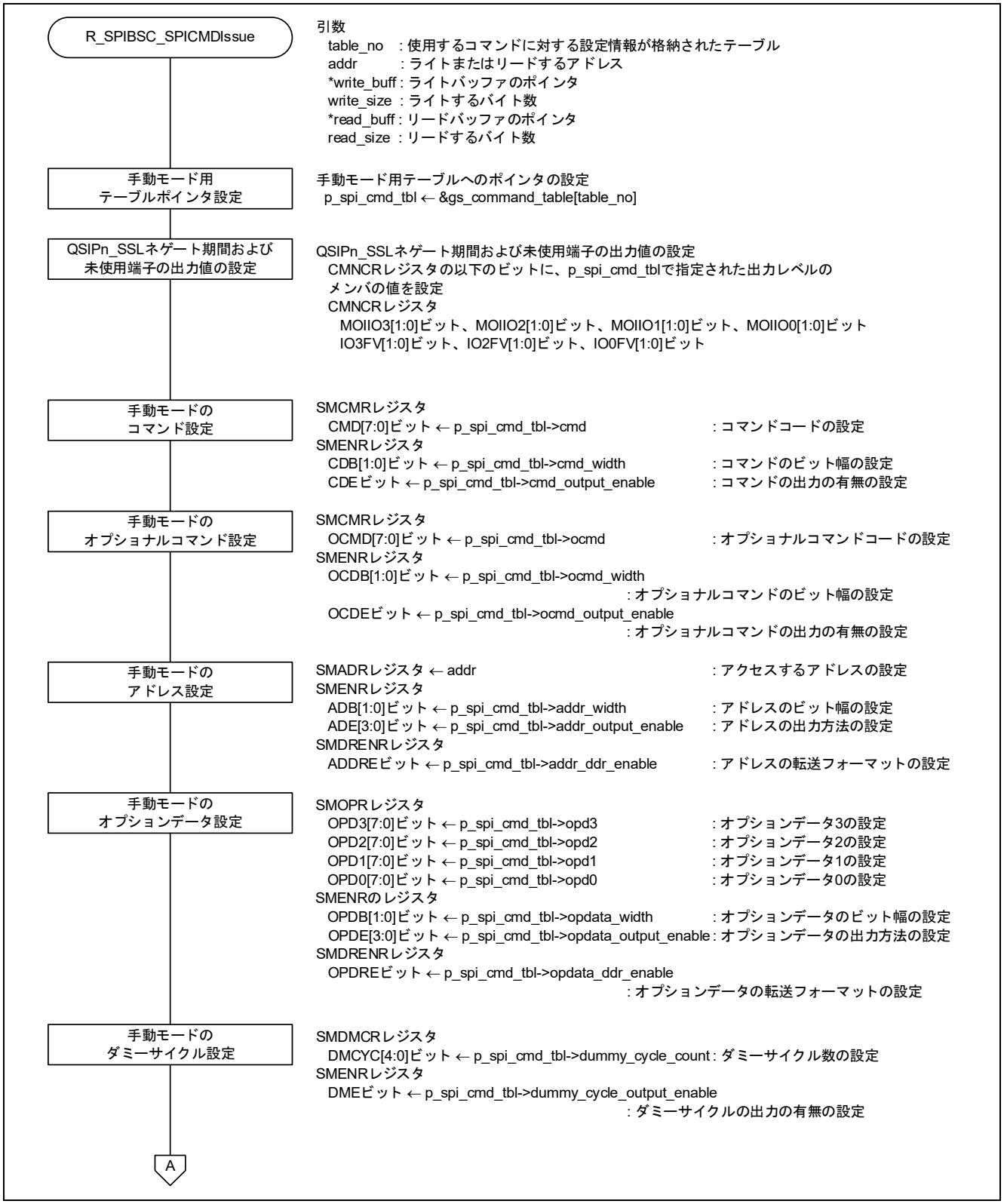


図5.13　シリアルフラッシュメモリへの SPI コマンド発行のフローチャート（1/4）

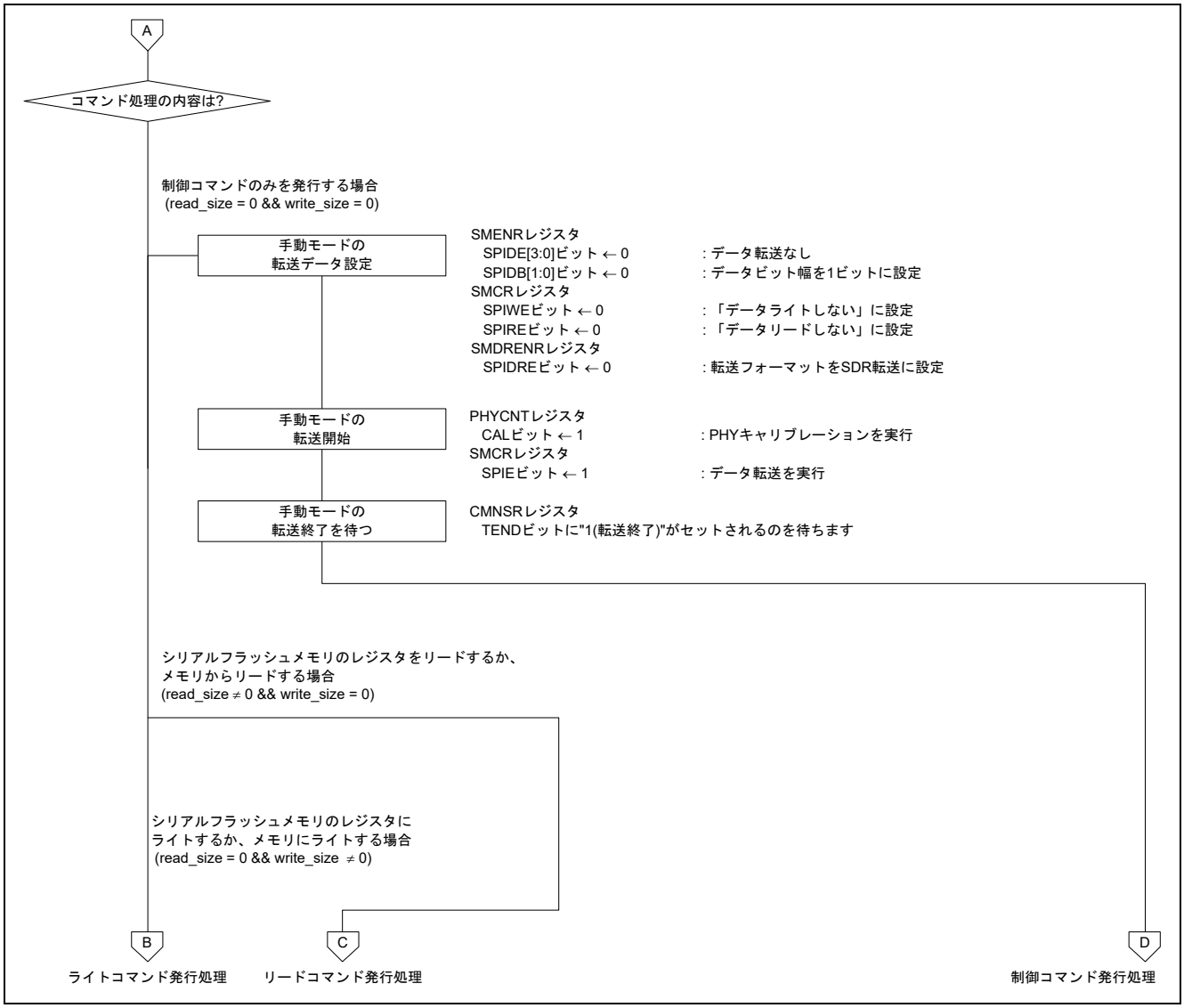


図5.14　シリアルフラッシュメモリへの SPI コマンド発行のフローチャート（2/4）

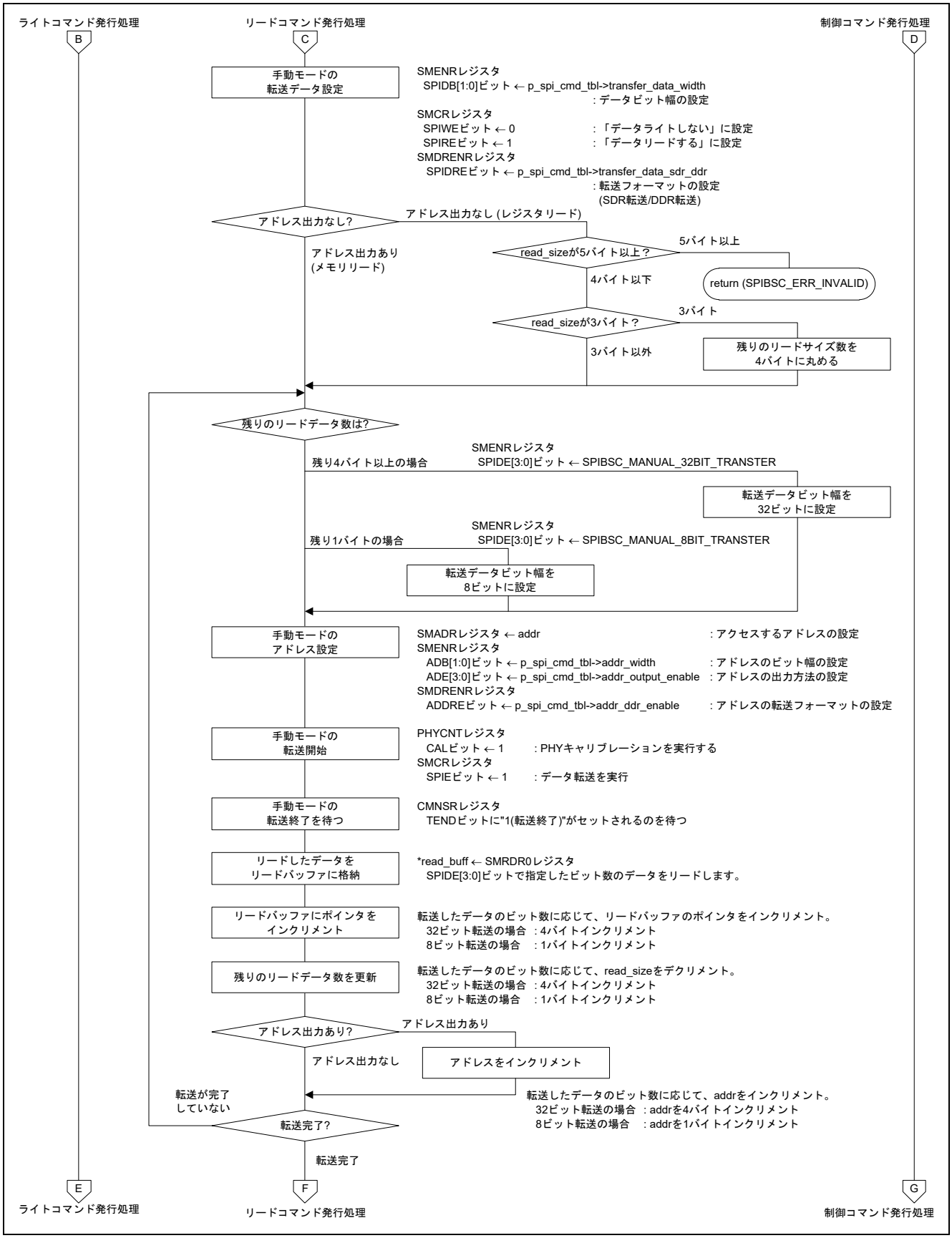


図5.15　シリアルフラッシュメモリへの SPI コマンド発行のフローチャート（3/4）

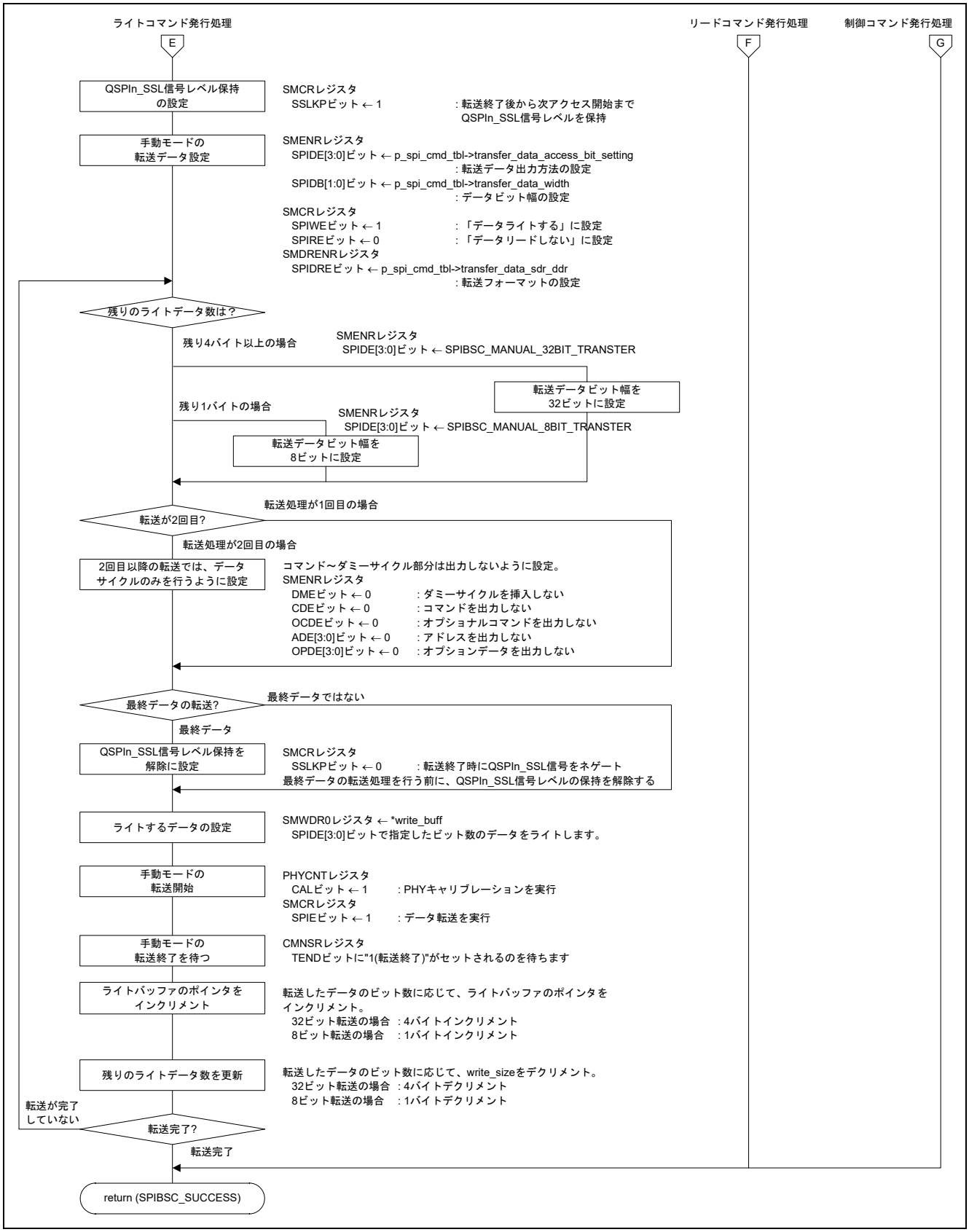


図5.16　シリアルフラッシュメモリへの SPI コマンド発行のフローチャート（4/4）

5.10.8 SPIBSC を DDR 転送で使用時のタイミング調整

シリアルフラッシュメモリに DDR 転送でアクセスする場合には、入力データの取り込みタイミングを調整するためのキャリブレーション処理を行う必要があります。

このキャリブレーション処理を実現するために、シリアルフラッシュメモリに 4 バイトの固定データ「0xAA00 FF55」を格納しておきます。図5.17にタイミング調整で使用する固定データのメモリ配置を示します。

キャリブレーション処理では、タイミング調整レジスタ（PHYADJ1、PHYADJ2、PHYCNT の CKSEL[1:0] ビット）の設定値を変更して取り込みタイミングを調整し、この固定データが正しくリードできる最適な値をタイミング調整レジスタに設定します。

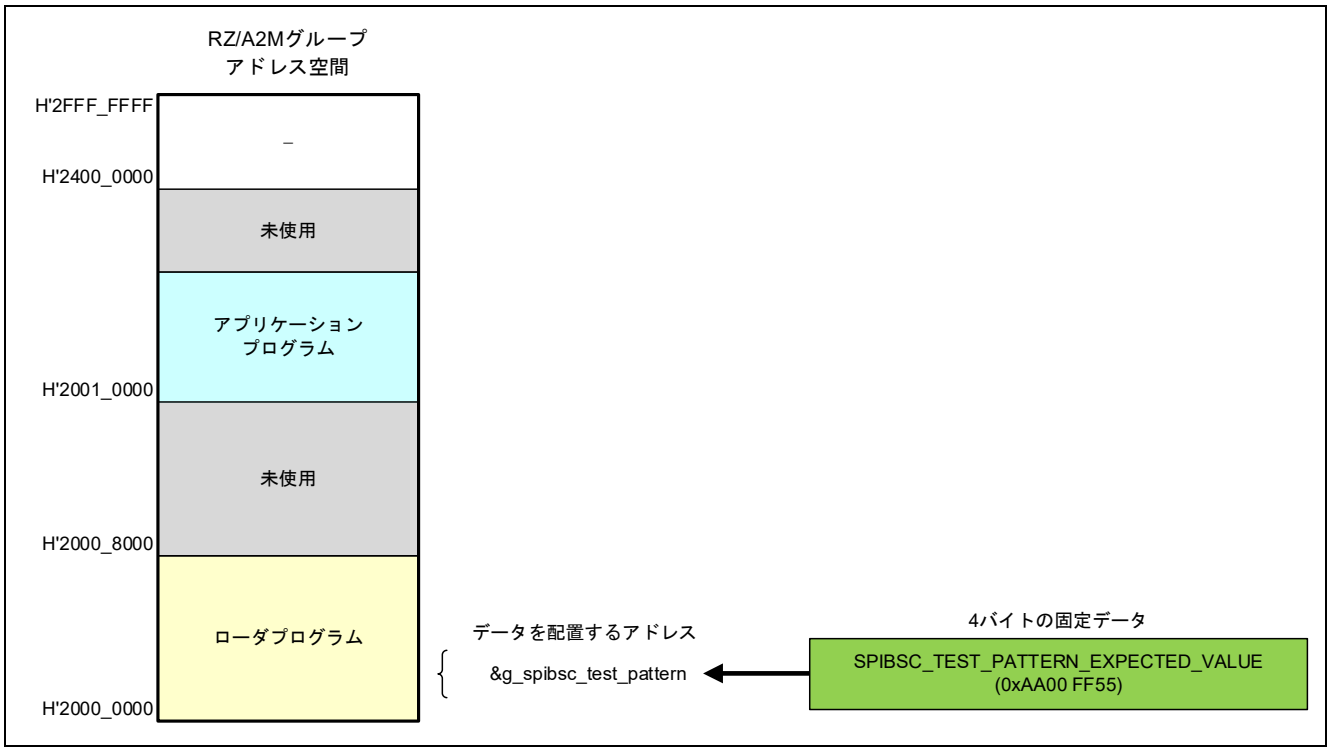


図5.17　タイミング調整で使用する固定データのメモリ配置

図5.18～図5.20にシリアルフラッシュメモリ接続時のタイミング調整のフローチャートを示します。本関数は、手動モードに設定している状態で使用してください。

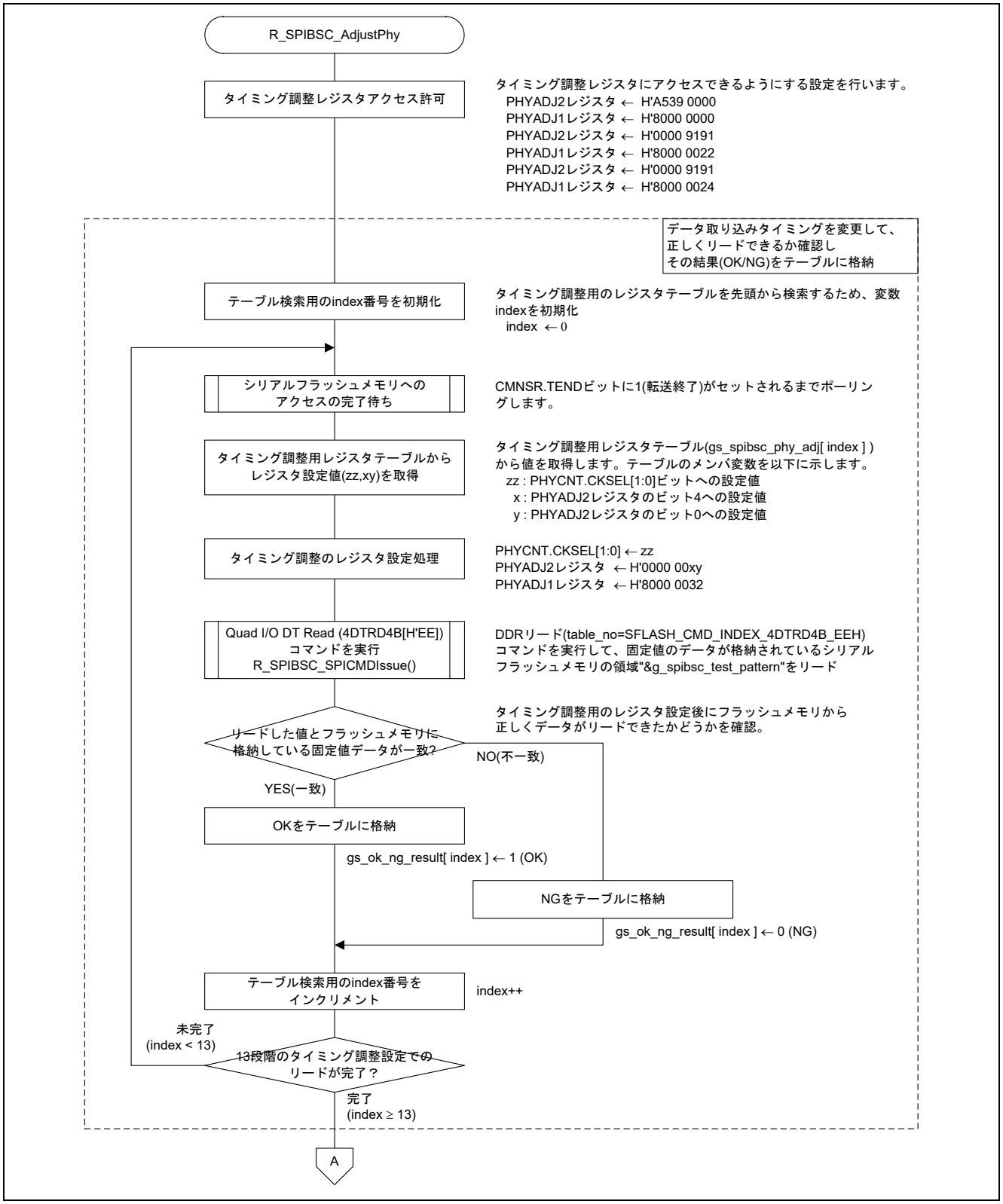


図5.18　シリアルフラッシュメモリ接続時のタイミング調整のフローチャート（1/3）

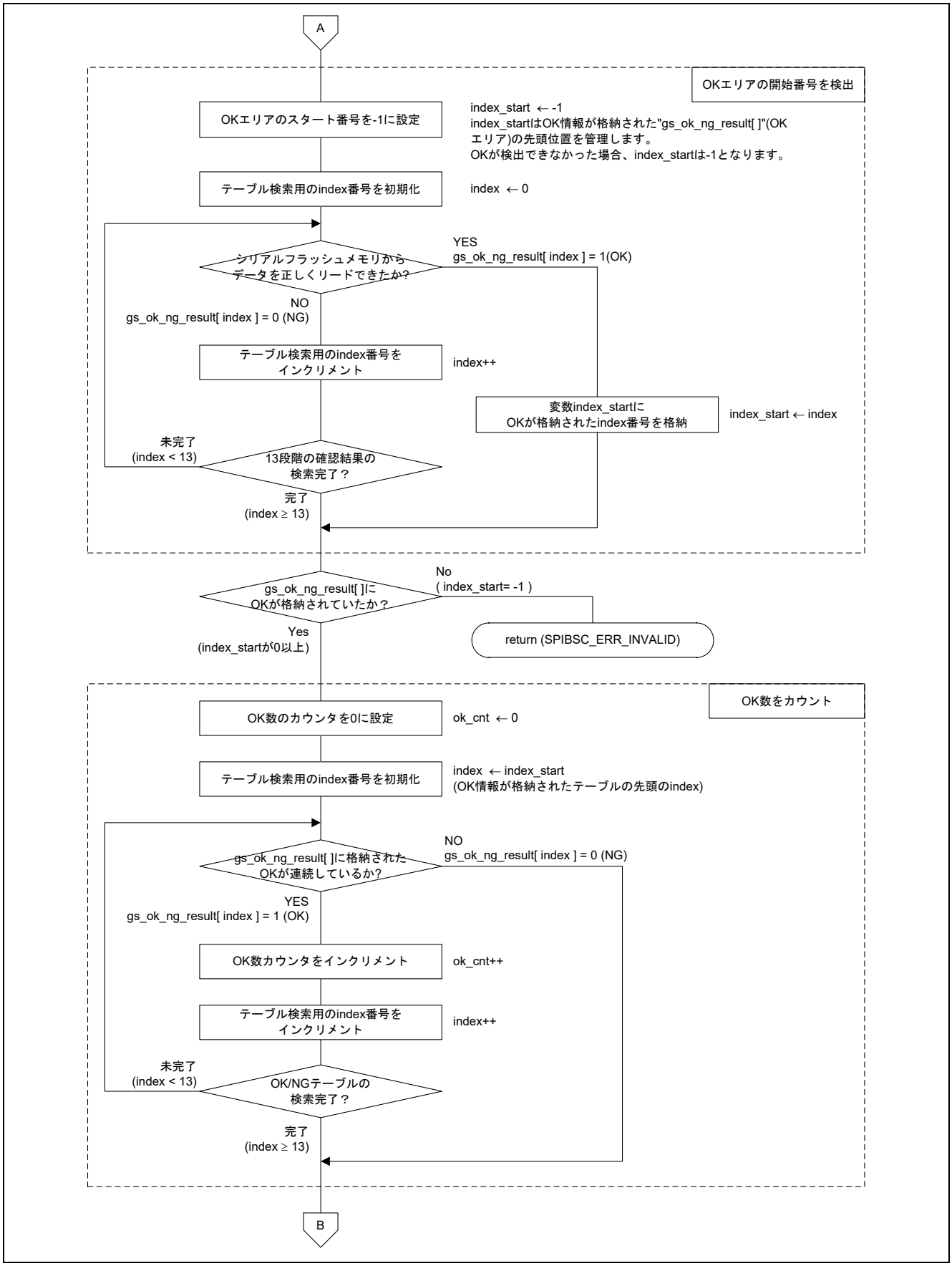


図5.19　シリアルフラッシュメモリ接続時のタイミング調整のフローチャート（2/3）

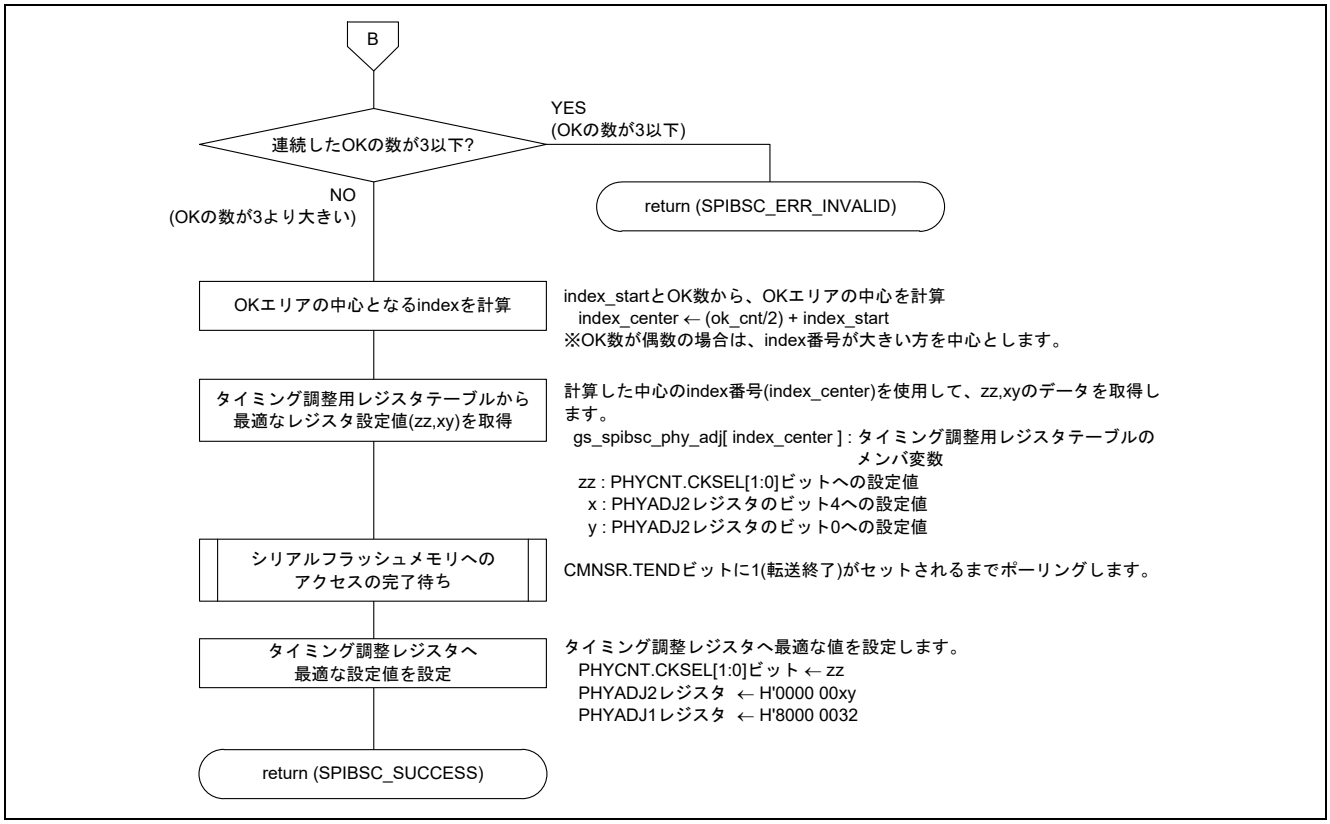


図5.20　シリアルフラッシュメモリ接続時のタイミング調整のフローチャート（3/3）

6. 応用例

6.1 サンプルコードを初期状態で使用する場合の動作

サンプルコードの初期状態では、Macronix社製シリアルフラッシュメモリ（MX25L51245GXD）に対して表6.1に示す設定内容でアクセスしています。

表6.1 サンプルコードのアクセス設定

項目	設定内容
シリアルフラッシュメモリ	• Macronix社製シリアルフラッシュメモリ • 型名：MX25L51245GXD • 使用するリードコマンド：H'EE（4DTRD4B） バス幅 4 ビット、DDR 転送 最大動作周波数 66MHz 時に、必要なダミーサイクル数 8
SPIBSC	• 接続するシリアルフラッシュメモリ：1 個 • データバス幅：4 ビット • アドレスバイト数：4 バイト （アドレス指定時に発行するバイト数） • リード時の転送フォーマット：DDR 転送

図6.1に、DDR転送のリード動作（サンプルコード初期状態）を示します。表6.2にサンプルコード初期状態でのレジスタ設定内容を示します。

コマンドサイクルでは SDR 転送を行い、MX25L51245GXDはクロックの立ち上がりエッジで入力データをサンプリングします。SPIBSC はクロックの立ち下がりエッジを基準にデータ出力を開始し、立ち上がりエッジではデータをホールドします。MX25L51245GXDが SPIBSC からの出力データの MSB を、最初の QSPIn_SPCLK の立ち上がりエッジでサンプリングします。

アドレスサイクルおよびデータサイクルは、DDR 転送を行います。アドレスサイクルでは、SPIBSC はコマンドサイクル終了後のクロックの立ち下がりエッジを基準にアドレス出力を開始し、以降はクロックの立ち上がりおよび立ち下がりの両方のエッジを基準にアドレスを出力します。MX25L51245GXDは、立ち上がりエッジでアドレスサイクルの入力サンプリングを開始し、以降は両方のエッジを基準にサンプリングします。

また、MX25L51245GXDからのデータ出力は、ダミーサイクルの最後のクロックの立ち下がりエッジでデータ出力が開始され、以降はクロックの立ち上がりおよび立ち下がりの両方のエッジを基準にデータが出力されます。SPIBSC は、データサイクルの最初のクロックの立ち上がりエッジで入力データをサンプリングし、以降は両方のエッジを基準にサンプリングします。

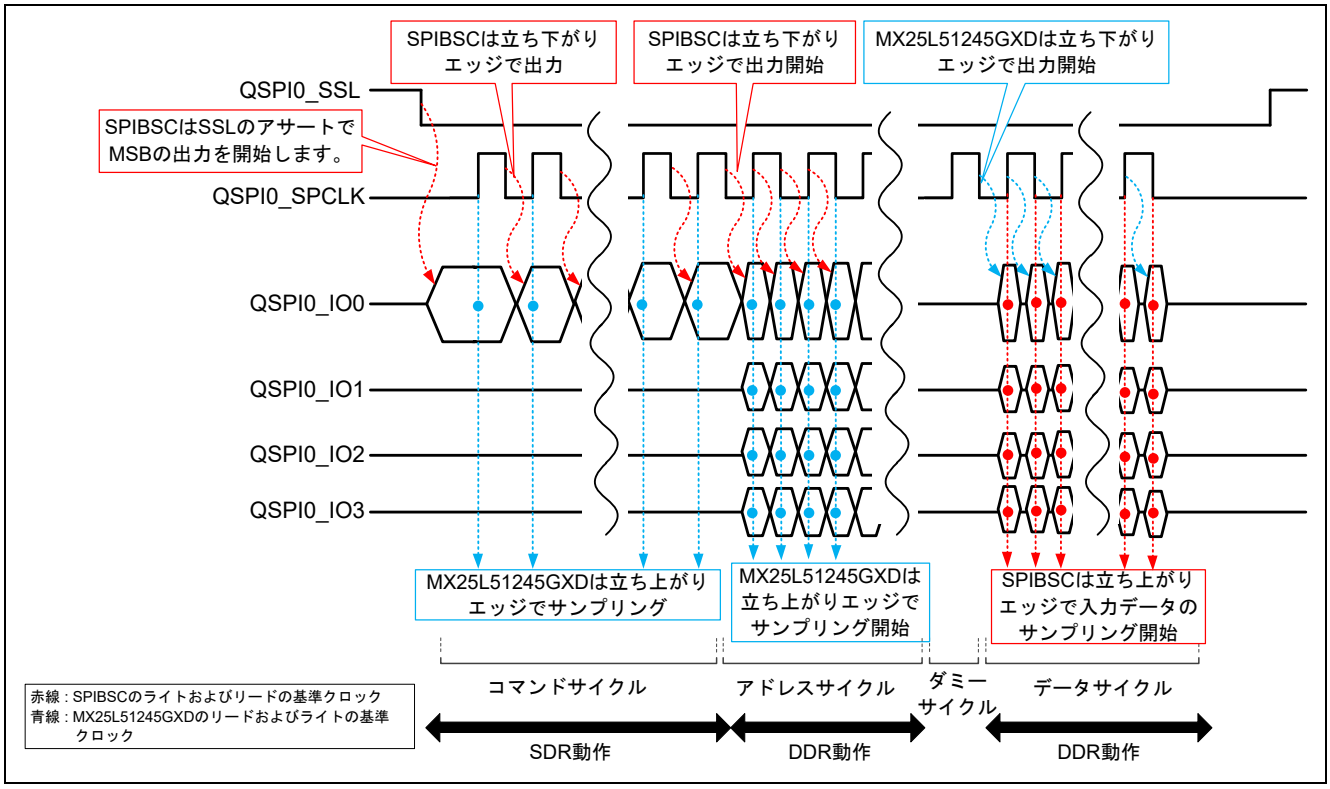


図6.1　DDR 転送のリード動作（サンプルコード初期状態）

サンプルコードでは、初期状態で SPIBSC およびシリアルフラッシュメモリのレジスタに、表6.2に示す内容を設定し、DDR 転送のリード動作を行います。

表6.2　サンプルコード初期状態でのレジスタ設定内容

設定項目	設定内容 （サンプルコードの初期状態）
リードコマンド設定	DRCMR.CMD[7:0] = 0xEE DRENR.CDB[1:0] = SPIBSC_1BIT_WIDTH DRENR.CDE = SPIBSC_OUTPUT_ENABLE
アドレス設定	DRENR.ADB[1:0] = SPIBSC_4BIT_WIDTH DRENR.ADE[3:0] = SPIBSC_OUTPUT_ADDR_32 DRDRENR.ADDRE = SPIBSC_DDR_TRANSFER
オプションデータ設定	DRENR.OPDB[1:0] = SPIBSC_4BIT_WIDTH DRENR.OPDE[3:0] = SPIBSC_OUTPUT_OPD_3 DRDRENR.OPDRE = SPIBSC_DDR_TRANSFER DROPR.OPD3[7:0] = 0x00
ダミーサイクル設定	DRENR.DME = SPIBSC_OUTPUT_ENABLE DRDMCR.DMCYC[4:0] = SPIBSC_DUMMY_07CYC
転送データ設定	DRENR.DRDB[1:0] = SPIBSC_4BIT_WIDTH DRDRENR.DRDRE = SPIBSC_DDR_TRANSFER
ステータスレジスタ設定	QE ビット = 1
コンフィグレーション レジスタ設定	DC[1:0]ビット = b'10 (8 サイクル) PBE ビット = 0 (Disable) ODS[2:0]ビット = b'110

サンプルコードは、初期化時に実行される R_SC_HardwareSetup 関数内の処理で、R_SPIBSC_ChangeMode 関数で手動モードに切り替え後、シリアルフラッシュメモリのレジスタ設定を行うためのユーザ定義関数 Userdef_SPIBSC_SFLASH_SetMode をコールします。Userdef_SPIBSC_SFLASH_SetMode 関数では、使用するリードコマンドの仕様に合わせて、シリアルフラッシュメモリのレジスタ（Status Register、Configuration Register）の設定処理を行います。

表6.3にMX25L51245GXDのStatus Registerを、表6.4にMX25L51245GXDのConfiguration Registerを示します。Userdef_SPIBSC_SFLASH_SetMode 関数により、表中に「　　」で示したビットを設定し、その他のビットは変更しません。

表6.3　MX25L51245GXDの Status Register

ビット	ビット名	属性（注）	説明
7	SRWD	NV	Status register write protect 1 = Status register write disabled 0 = Status register write enabled
6	QE	NV	Quad enable 1 = Quad enable 0 = Not Quad enable
5,4,3,2	BP3,BP2,BP1,BP0	NV	Level of protected block
1	WEL	V	Write enable latch 1 = Write enable 0 = Not write enable
0	WIP	V	Write in progress bit 1 = Write operation 0 = Not in write operation

【注】 属性の"NV"は"Non-volatile bit"を、"V"は"Volatile bit"を意味します。

表6.4　MX25L51245GXDの Configuration Register

ビット	ビット名	属性（注 1）	説明
7,6	DC1, DC0	V	Dummy cycle 1, Dummy cycle 0 DC[1:0] = B'10（注 2）
5	4BYTE	V	0 = 3-byte address mode 1 = 4-byte address mode
4	PBE	V	Preamble bit enable 0 = Disable 1 = Enable
3	TB	OTP	Top/bottom selected 0 = Top area protect 1 = Bottom area protect
2,1,0	ODS2, ODS1, ODS0	V	Output driver strength (ODS2 = 1, ODS1 = 1, ODS0 = 0)（注 3）

【注】 1. 属性の"NV"は"Non-volatile bit"を、"OTP"は"One-time prgrammable bit"を意味します。
2. 表6.5に示すように、使用するリードコマンドによってダミーサイクル数は異なります。サンプルコードでは、リードコマンドに H'EE を使用しており、最適なダミーサイクル数となるように、DC[1:0]ビットに B'10 を設定しています。
3. サンプルコードでは、ODS2 = 1, ODS1 = 1, ODS0 = 0 を設定し、RZ/A2Mと接続時にMX25L51245GXDのデータホールド時間（tCHDX）およびデータ出力遅延時間（tCLQV）の AC タイミング特性が最適となるようにしています（MX25L51245GXDに接続されるデバイスの負荷容量が 15pF 以下で、MX25L51245GXDの電源（VCC）に 3.0～3.6V を供給する場合に可能な設定です）。

表6.5にMX25L51245GXDの最大動作周波数に対して必要なダミーサイクル数の一覧を示します。リードコマンドと動作周波数によって、必要なダミーサイクル数が異なります。サンプルコードでは、リードコマンドに H'EE コマンドを使用し、QSPIn_SPCLK が 66MHz で使用しているため、ダミーサイクル数が 8 サイクルの DC[1:0] = B'10 が最適な設定です。

使用するリードコマンドを変更する場合には、リードコマンドと QSPIn_SPCLK の周波数に合わせて、ダミーサイクル数を設定してください。

表6.5　MX25L51245GXDの最大動作周波数に対して必要なダミーサイクル数の一覧

リードコマンド	Configuration Register の DC[1:0]ビット			
	B'00	B'01	B'10	B'11
FAST READ（H'0B） FAST READ4B（H'0C）	8 サイクル /133MHz	6 サイクル /133MHz	8 サイクル /133MHz	10 サイクル /66MHz
FASTDTRD（H'0D） FRDTRD4B（H'0E）	8 サイクル /66MHz	6 サイクル /66MHz	8 サイクル /66MHz	10 サイクル /83MHz
4READ（H'EB） 4READ4B（H'EC）	6 サイクル /84MHz	**4 サイクル /70MHz（注 2）**	8 サイクル /104MHz	10 サイクル /133MHz
4DTRD（H'ED） 4DTRD4B（H'EE）	6 サイクル /52MHz	4 サイクル /42MHz	**8 サイクル /66MHz（注 1）**	10 サイクル /100MHz

【注】 1. サンプルコードのデフォルト状態では、外部アドレス空間リードモードのリードコマンドとして 4DTRD4B（H'EE）を使用しています。66MHz 動作時に最小のダミーサイクル数となるように、サンプルコードでは、DC[1:0]ビットに B'10 を設定しています。

2. サンプルコードで、外部アドレス空間リードモードで SDR 転送のリードアクセスを行う場合には、4READ4B（H'EC）を使用し、66MHz 動作時に最小のダミーサイクル数となるように、DC[1:0]ビットに B'01 を設定します。

6.2　シリアルフラッシュメモリを変更しない場合のサンプルコード変更方法

6.2.1　SDR 転送のリードコマンドに変更する方法

サンプルコードでは、外部アドレス空間リードモードで使用するリードコマンドとして、SDR 転送のリードコマンドを使用することが可能です。hwsetup.c に定義されているマクロ定義 SPIBSC_PRV_DDR_SETTING の定義を無効化することにより、SDR 転送のリードコマンドを使用して、シリアルフラッシュメモリへの直接アクセスを行います。表6.6にSDR転送のリードコマンドに変更した場合のレジスタ設定内容を示します。

表6.6　SDR 転送のリードコマンドに変更した場合のレジスタ設定内容

設定項目	設定内容
リードコマンド設定	DRCMR.CMD[7:0] = 0xEC DRENR.CDB[1:0] = SPIBSC_1BIT_WIDTH DRENR.CDE = SPIBSC_OUTPUT_ENABLE
アドレス設定	DRENR.ADB[1:0] = SPIBSC_4BIT_WIDTH DRENR.ADE[3:0] = SPIBSC_OUTPUT_ADDR_32 DRDRENR.ADDRE = SPIBSC_SDR_TRANSFER
オプションデータ設定	DRENR.OPDB[1:0] = SPIBSC_4BIT_WIDTH DRENR.OPDE[3:0] = SPIBSC_OUTPUT_OPD_3 DRDRENR.OPDRE = SPIBSC_SDR_TRANSFER DROPR.OPD3[7:0] = 0x00
ダミーサイクル設定	DRENR.DME = SPIBSC_OUTPUT_ENABLE DRDMCR.DMCYC[4:0] = SPIBSC_DUMMY_02CYC
転送データ設定	DRENR.DRDB[1:0] = SPIBSC_4BIT_WIDTH DRDRENR.DRDRE = SPIBSC_SDR_TRANSFER
ステータスレジスタ設定	QE ビット = 1
コンフィグレーション レジスタ設定	DC[1:0]ビット = b'01 (4 サイクル) PBE ビット = 0 (Disable) ODS[2:0]ビット = b'110

図6.2にSDR転送のリード動作を示します。リード動作では、SPIBSC からのコマンド出力、アドレス出力、およびダミーサイクルの出力に続いて、データサイクルが開始し、MX25L51245GXDからデータが出力されます。

MX25L51245GXDは、SDR 転送時に、クロックの立ち上がりエッジで入力データをサンプリングします。SPIBSC はクロックの立ち下がりエッジを基準にデータ出力を開始し、立ち上がりエッジではデータをホールドします。

また、MX25L51245GXDは、SDR 転送時に、クロックの立ち下がりエッジを基準としてデータの出力を開始します。データサイクルでは、ダミーサイクルの最終クロックの立ち下がりエッジを基準に、次のクロックの立ち下がりエッジまでデータを出力します。SPIBSC では、クロックの立ち上がりエッジで入力データをサンプリングしています。

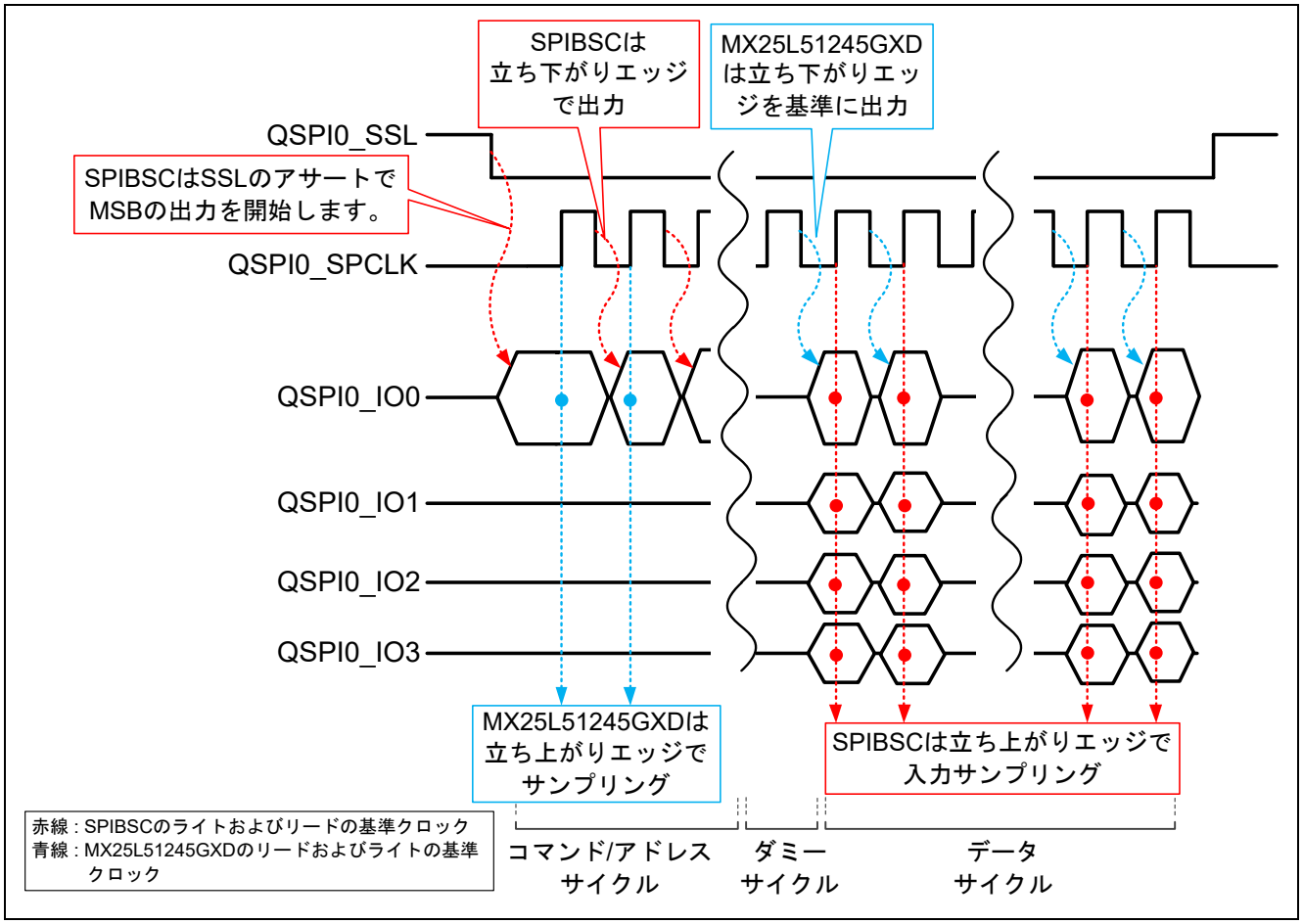


図6.2　SDR 転送のリード動作

6.3　シリアルフラッシュメモリを変更する場合のサンプルコード変更方法

シリアルフラッシュメモリを変更する場合、使用するシリアルフラッシュメモリの仕様に合わせてサンプルコードを変更する必要があります。

表6.7にサンプルコードの変更のポイントを示します。

表6.7　サンプルコードの変更のポイント

変更のポイント	内容	関連する見出し番号
リードコマンド発行時の出力信号	使用するシリアルフラッシュメモリのリードコマンドの仕様に合わせて、外部アドレス空間リードモードのリードコマンド発行時にシリアルフラッシュメモリに出力する信号を変更します。	6.3.1
シリアルフラッシュメモリのレジスタの設定	使用するシリアルフラッシュメモリに合わせて、SPIBSC を外部アドレス空間リードモードで使用する場合に必要なシリアルフラッシュメモリのレジスタの設定を行います。	6.3.2
シリアルフラッシュメモリのライト完了待ち	使用するシリアルフラッシュメモリに合わせて、シリアルフラッシュメモリのライト完了を待ちます。	6.3.3
シリアルフラッシュメモリのステータスレジスタのリード	使用するシリアルフラッシュメモリに合わせて、シリアルフラッシュメモリのステータスレジスタをリードします。	6.3.4
シリアルフラッシュメモリのコンフィグレーションレジスタのリード	使用するシリアルフラッシュメモリに合わせて、シリアルフラッシュメモリのコンフィグレーションレジスタをリードします。	6.3.5
シリアルフラッシュメモリのライト許可	使用するシリアルフラッシュメモリに合わせて、シリアルフラッシュメモリのレジスタを設定するために、ライト許可に設定します（注）。	6.3.6
シリアルフラッシュメモリのステータス/コンフィグレーションレジスタのライト	使用するシリアルフラッシュメモリに合わせて、シリアルフラッシュメモリのステータス/コンフィグレーションレジスタにライトします。	6.3.7

【注】　シリアルフラッシュメモリによっては、シリアルフラッシュメモリのレジスタにライトするために、ライト許可が必要な場合があります。

表6.7に示した設定は、SPIBSC とシリアルフラッシュメモリの初期設定処理（R_SPIBSC_Setup 関数）を実行することにより行われます。サンプルコードのユーザ定義関数の処理を、使用するシリアルフラッシュメモリに合わせて変更することによって対応することができます。図6.3にSPIBSCとシリアルフラッシュメモリの設定処理のモジュール階層図を示し、それぞれの処理について6.3.1～6.3.7にサンプルコードで実施している処理の内容を示します。

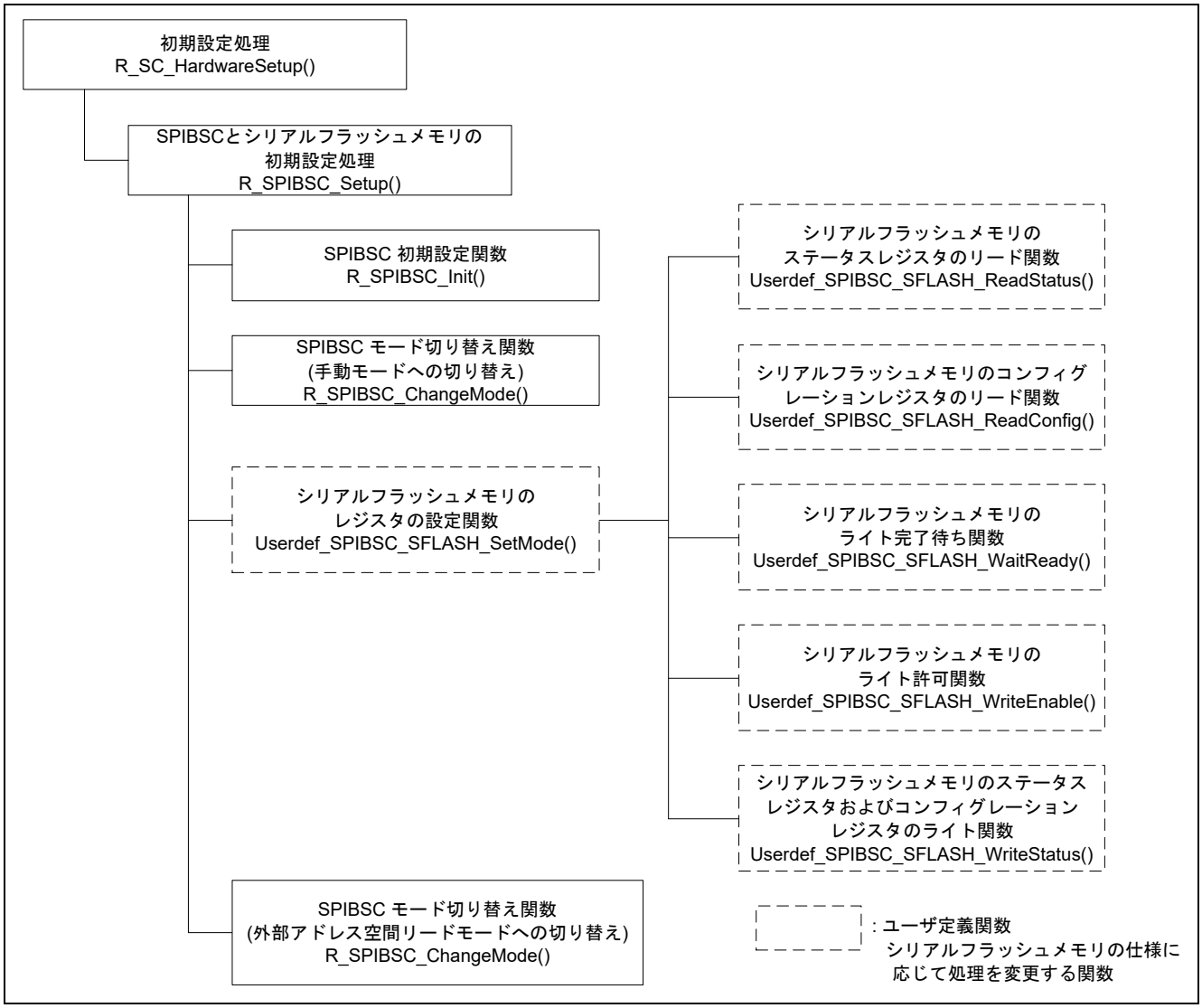


図6.3　SPIBSC とシリアルフラッシュメモリの設定処理のモジュール階層図

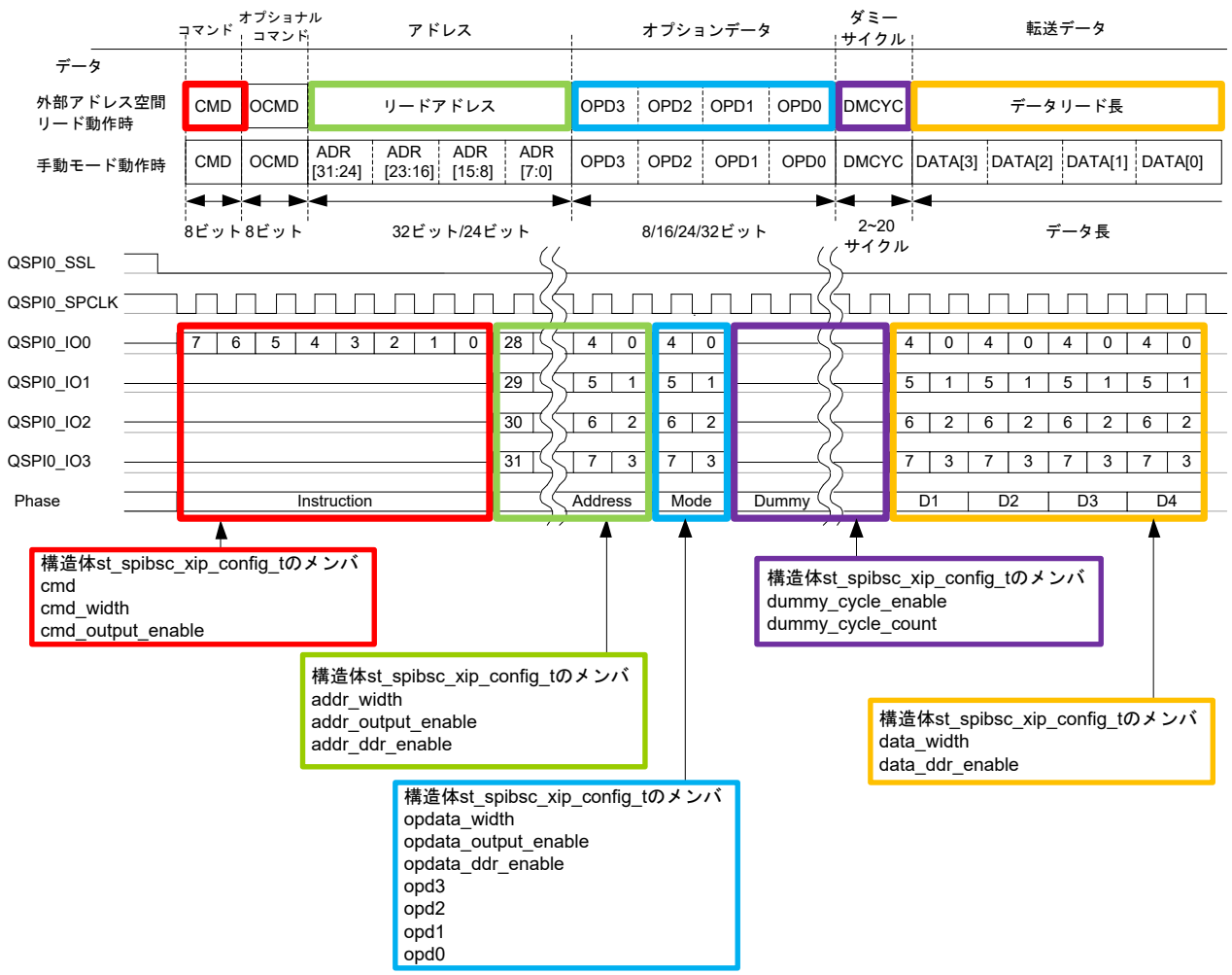
6.3.1　リードコマンド発行時の出力信号

外部アドレス空間リードモードでは、SPI マルチ I/O バス空間へのリードアクセスは、リードコマンド発行時に SPI 通信に変換した信号をシリアルフラッシュメモリに出力することにより、リード動作を開始します。使用するシリアルフラッシュメモリを変更する場合は、シリアルフラッシュメモリのリードコマンド仕様に合わせて、リードコマンド発行時の出力信号を変更する必要があります。

SPIBSC は、SPIBSC レジスタを設定することにより、外部アドレス空間リードモード時にシリアルフラッシュメモリに出力するリードコマンドの信号を変更することが可能です。

サンプルコードでは、SPIBSC 外部アドレス空間リードモードのリードコマンド設定テーブル(表6.8、表6.9）の内容を変更することにより、SPIBSC レジスタの設定値を変更し、リードコマンド発行時の出力信号を変更することが可能です。リードコマンド設定テーブルで指定した SPIBSC レジスタの設定は、SPIBSC 動作モード設定関数（R_SPIBSC_ChangeMode）を実行することにより設定しています。なお、リードコマンドの設定内容に関連するシリアルフラッシュメモリのレジスタ設定（ダミーサイクル数、ビット幅など）は、シリアルフラッシュメモリのレジスタ設定関数（Userdef_SPIBSC_SFLASH_SetMode）により設定しています。使用するシリアルフラッシュメモリの仕様に合わせて、Userdef_SPIBSC_SFLASH_SetMode 関数でのシリアルフラッシュメモリへのレジスタ設定処理の実装についても、合わせて変更してください。

図6.4にSPIBSCレジスタと外部アドレス空間リード時にシリアルフラッシュメモリに出力される波形の関係を示します。サンプルコードの内容を参照して、使用するシリアルフラッシュメモリのリードコマンドに合わせて、外部アドレス空間リードモードのリードコマンド設定テーブル（表6.8、表6.9）の設定を変更してください。



サンプルコードでは、Macronix社製シリアルフラッシュメモリ（MX25L51245GXD）を使用時に、
各構造体のメンバを以下のように設定しています。

■DDRリードコマンド(EEh)使用時：gs_xip_read_table[0]

cmd	: H'EE
cmd_width	: SPIBSC_1BIT_WIDTH
cmd_output_enable	: SPIBSC_OUTPUT_ENABLE
addr_width	: SPIBSC_4BIT_WIDTH
addr_output_enable	: SPIBSC_OUTPUT_ADDR 32
addr_ddr_enable	: SPIBSC_DDR_TRANSFER
opdata_width	: SPIBSC_4BIT_WIDTH
opdata_output_enable	: SPIBSC_OUTPUT_OPD_3
opdata_ddr_enable	: SPIBSC_DDR_TRANSFER
opd3	: H'00
opd2	: H'00 (OPD2は出力しない。)
opd1	: H'00 (OPD1は出力しない。)
opd0	: H'00 (OPD0は出力しない。)
dummy_cycle_enable	: SPIBSC_OUTPUT_ENABLE
dummy_cycle_count	: SPIBSC_DUMMY_7CYC
data_width	: SPIBSC_4BIT_WIDTH
data_ddr_enable	: SPIBSC_DDR_TRANSFER

■SDRリードコマンド(ECh)使用時：gs_xip_read_table[1]

cmd	: H'EC
cmd_width	: SPIBSC_1BIT_WIDTH
cmd_output_enable	: SPIBSC_OUTPUT_ENABLE
addr_width	: SPIBSC_4BIT_WIDTH
addr_output_enable	: SPIBSC_OUTPUT_ADDR 32
addr_ddr_enable	: SPIBSC_SDR_TRANSFER
opdata_width	: SPIBSC_4BIT_WIDTH
opdata_output_enable	: SPIBSC_OUTPUT_OPD_3
opdata_ddr_enable	: SPIBSC_SDR_TRANSFER
opd3	: H'00
opd2	: H'00 (OPD2は出力しない。)
opd1	: H'00 (OPD1は出力しない。)
opd0	: H'00 (OPD0は出力しない。)
dummy_cycle_enable	: SPIBSC_OUTPUT_ENABLE
dummy_cycle_count	: SPIBSC_DUMMY_2CYC
data_width	: SPIBSC_4BIT_WIDTH
data_ddr_enable	: SPIBSC_SDR_TRANSFER

【注】　図中の Mode サイクルは、MX25L51245GXDでは Performance Enhance Indicator のサイクルを示します。

図6.4　SPIBSC レジスタと外部アドレス空間リード時にシリアルフラッシュメモリに出力される波形の関係

表6.8　外部アドレス空間リードモード用のコマンド設定テーブル gs_xip_read_table[0]：4BIT_DDR_READ

メンバ名	説明	設定値
uint8_t command_name[20]	コマンド識別の文字列	"4BIT_DDR_READ"
uint8_t cmd	コマンドのコード	**0xEE**
uint8_t cmd_width	コマンドのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t cmd_output_enable	コマンドの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t ocmd	オプショナルコマンドのコード	0x00
uint8_t ocmd_width	オプショナルコマンドのビット幅	SPIBSC_1BIT_WIDTH
uint8_t ocmd_output_enable	オプショナルコマンドの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t addr_width	アドレスのビット幅	**SPIBSC_4BIT_WIDTH**
uint8_t addr_output_enable	アドレスの出力設定	**SPIBSC_OUTPUT_ADDR_32**
uint8_t addr_ddr_enable	アドレスの転送方式	**SPIBSC_DDR_TRANSFER**
uint8_t opdata_width	オプションデータのビット幅	**SPIBSC_4BIT_WIDTH**
uint8_t opdata_output_enable	オプションデータの出力設定	**SPIBSC_OUTPUT_OPD_3（注）**
uint8_t opdata_ddr_enable	オプションデータの転送方式	**SPIBSC_DDR_TRANSFER（注）**
uint8_t opd3	Option Data3(8bit)の設定(1 番目に出力)	**0x00（注）**
uint8_t opd2	Option Data2(8bit)の設定(2 番目に出力)	0x00
uint8_t opd1	Option Data1(8bit)の設定(3 番目に出力)	0x00
uint8_t opd0	Option Data0(8bit)の設定(4 番目に出力)	0x00
uint8_t dummy_cycle_enable	ダミーサイクルの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t dummy_cycle_count	ダミーサイクル数	**SPIBSC_DUMMY_07CYC**
uint8_t data_width	転送データのビット幅	**SPIBSC_4BIT_WIDTH**
uint8_t data_ddr_enable	転送データの転送方式	**SPIBSC_DDR_TRANSFER**

【注】 MX25L51245GXDは、アドレスサイクルに続く Performance enhance indicator のサイクル期間（図6.5 の P[7:0]の部分）に、ビット 7～4 とビット 3～0 をトグルさせるデータ（例えば、H'A5、H'5A、H'F0、H'0F など）が入力されると、Performance Enhance Mode に遷移します。RZ/A2Mの外部アドレス空間リードモードは、Performance Enhance Mode のデータ転送に対応していませんので、サンプルコードでは、OPD3 から H'00 が出力されるように設定し、MX25L51245GXDが Performance Enhance Mode に遷移しないようにしています。

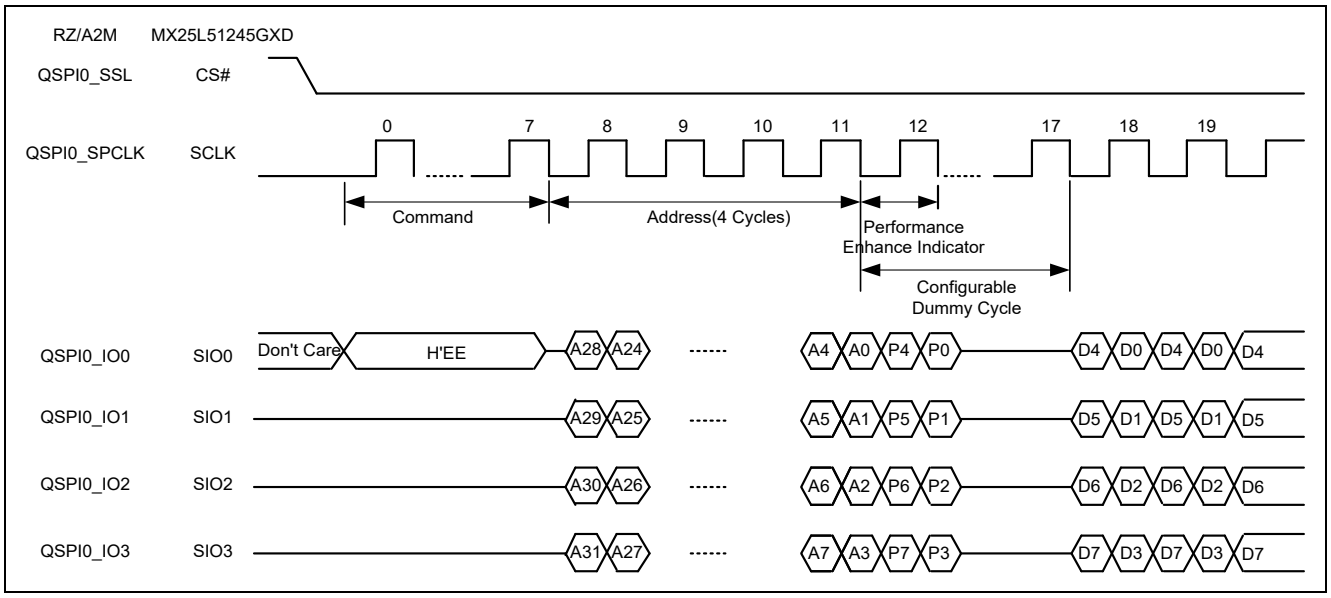


図6.5　EEH リードコマンドの波形フォーマット（参考）

表6.9　外部アドレス空間リードモード用のコマンド設定テーブル gs_xip_read_table[1]：4BIT_SDR_READ

メンバ名	説明	設定値
uint8_t command_name[20]	コマンド識別の文字列	"4BIT_SDR_READ"
uint8_t cmd	コマンドのコード	**0xEC**
uint8_t cmd_width	コマンドのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t cmd_output_enable	コマンドの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t ocmd	オプショナルコマンドのコード	0x00
uint8_t ocmd_width	オプショナルコマンドのビット幅	SPIBSC_1BIT_WIDTH
uint8_t ocmd_output_enable	オプショナルコマンドの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t addr_width	アドレスのビット幅	**SPIBSC_4BIT_WIDTH**
uint8_t addr_output_enable	アドレスの出力設定	**SPIBSC_OUTPUT_ADDR_32**
uint8_t addr_ddr_enable	アドレスの転送方式	**SPIBSC_SDR_TRANSFER**
uint8_t opdata_width	オプションデータのビット幅	**SPIBSC_4BIT_WIDTH**
uint8_t opdata_output_enable	オプションデータの出力設定	**SPIBSC_OUTPUT_OPD_3（注）**
uint8_t opdata_ddr_enable	オプションデータの転送方式	**SPIBSC_SDR_TRANSFER（注）**
uint8_t opd3	Option Data3(8bit)の設定(1 番目に出力)	**0x00（注）**
uint8_t opd2	Option Data2(8bit)の設定(2 番目に出力)	0x00
uint8_t opd1	Option Data1(8bit)の設定(3 番目に出力)	0x00
uint8_t opd0	Option Data0(8bit)の設定(4 番目に出力)	0x00
uint8_t dummy_cycle_enable	ダミーサイクルの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t dummy_cycle_count	ダミーサイクル数	**SPIBSC_DUMMY_02CYC**
uint8_t data_width	転送データのビット幅	**SPIBSC_4BIT_WIDTH**
uint8_t data_ddr_enable	転送データの転送方式	**SPIBSC_SDR_TRANSFER**

【注】 MX25L51245GXDは、アドレスサイクルに続く Performance enhance indicator のサイクル期間（図6.6 の P[7:0]の部分）に、ビット 7～4 とビット 3～0 をトグルさせるデータ（例えば、H'A5、H'5A、H'F0、H'0F など）が入力されると、Performance Enhance Mode に遷移します。RZ/A2Mの外部アドレス空間リードモードは、Performance Enhance Mode のデータ転送に対応していませんので、サンプルコードでは、OPD3 から H'00 が出力されるように設定し、MX25L51245GXDが Performance Enhance Mode に遷移しないようにしています。

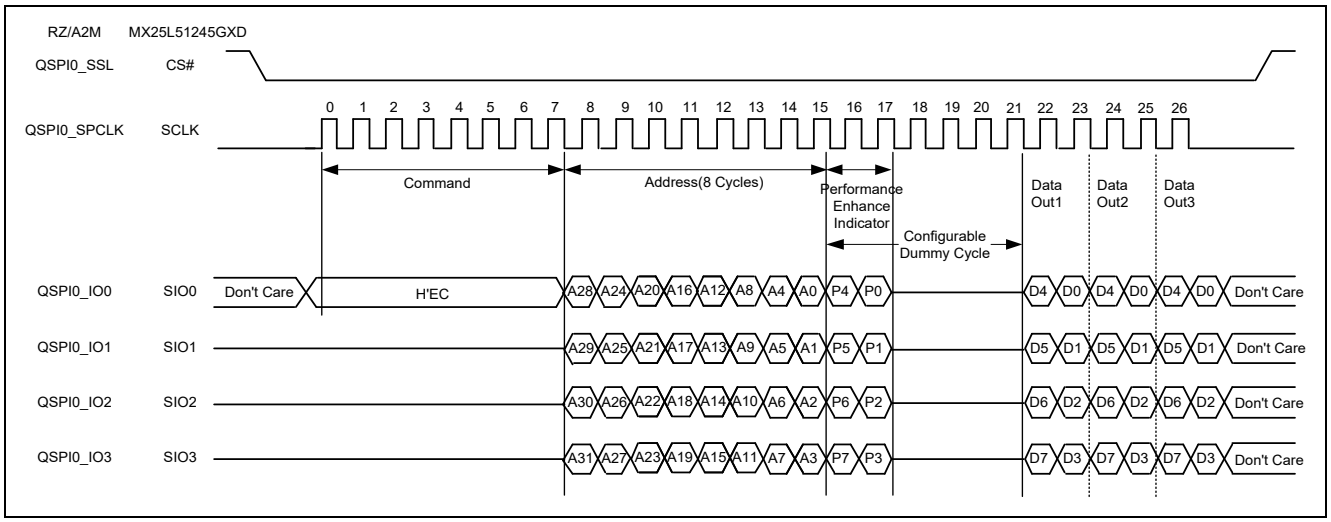


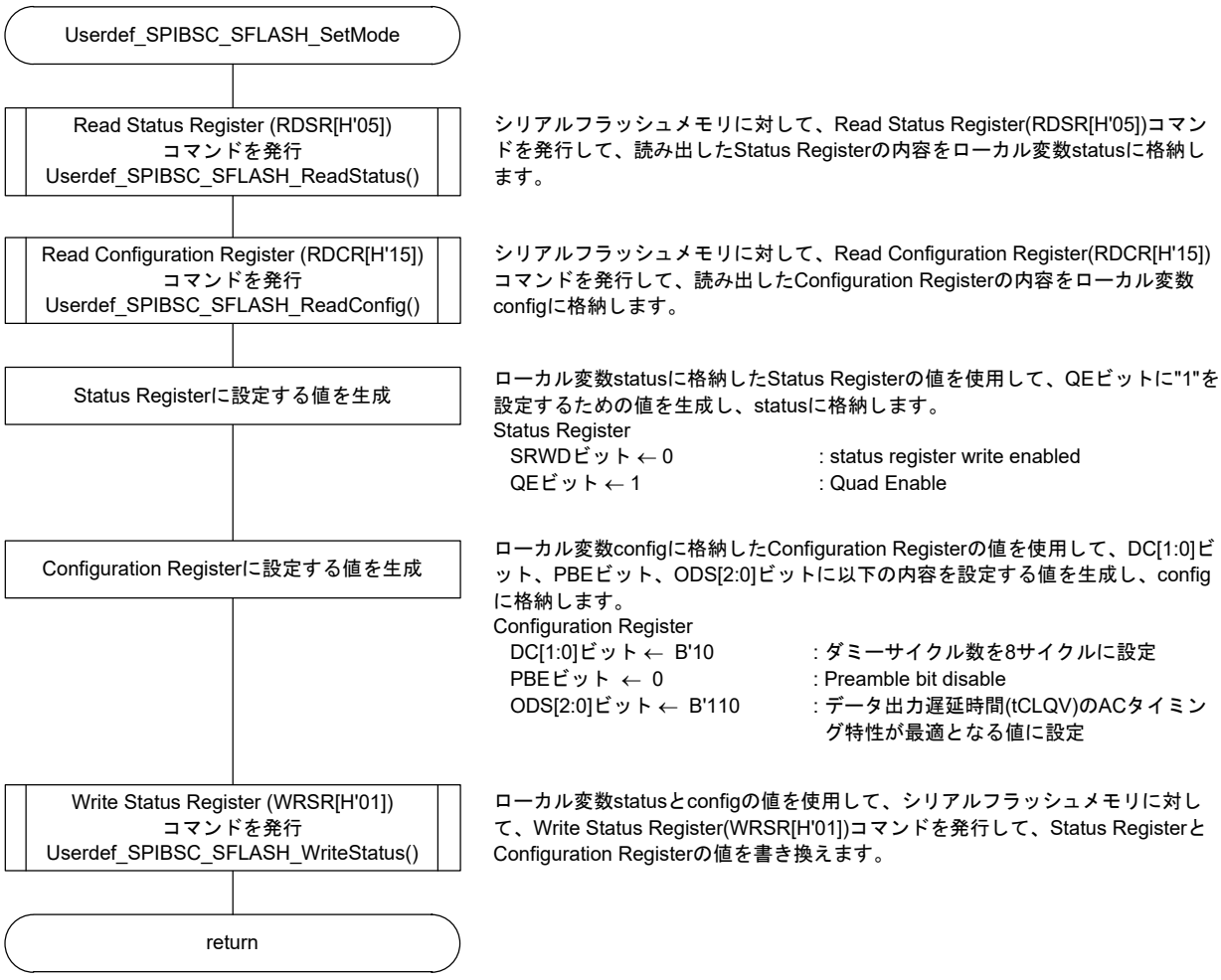
図6.6　ECH リードコマンドの波形フォーマット（参考）

6.3.2　シリアルフラッシュメモリのレジスタの設定

サンプルコードでは、使用するリードコマンド仕様に適したシリアルフラッシュメモリMX25L51245GXDのレジスタ設定（Status Register の QE ビット、Configuration Register の DC[1:0]ビット、PBE ビット、ODS[2:0]ビット）を行う処理を、ユーザ定義関数 Userdef_SPIBSC_SFLASH_SetMode 関数で実現しています。

サンプルコードでは、リードコマンドとして Quad リードコマンド（H'EE または H'EC）を使用しているため、ビット幅が 4 ビットになるように Status Register の QE（Quad Enable）ビットを 1 に設定しています。また、使用するリードコマンドと動作周波数に応じて、挿入されるダミーサイクル数が最小になるように、Configuration Register の DC[1:0]ビットの値を設定しています（詳細は表6.5 MX25L51245GXDの最大動作周波数に対して必要なダミーサイクル数の一覧を参照）。さらに、RZ/A2MとMX25L51245GXDの接続に必要なデータ出力遅延時間（tCLQV）となるように、Configuration Register の ODS[2:0]ビットに B'110 を設定しています。ご使用のシリアルフラッシュメモリのリードコマンド仕様に適した、シリアルフラッシュメモリの制御レジスタの設定値となるように、Userdef_SPIBSC_SFLASH_SetMode 関数の実装を変更してください。

図6.7に、サンプルコードのUserdef_SPIBSC_SFLASH_SetMode関数のフローを示します。



【注】　シリアルフラッシュメモリのアクセスを 1 ビット幅から 4 ビット幅に変更し、リードコマンドのダミーサイクル数を最小にするために、Status Register と Configuration Register に設定を行います。これらのレジスタの設定は、SPIBSC を外部アドレス空間リードモードから手動モードに切り替えて、WRSR コマンドをシリアルフラッシュメモリに発行する必要があります。

図6.7　Userdef_SPIBSC_SFLASH_SetMode 関数のフロー

6.3.3　シリアルフラッシュメモリのライト完了待ち

シリアルフラッシュメモリのレジスタ（Status Register および Configuration Register）やメモリにライトした場合、シリアルフラッシュメモリはビジー状態に遷移します。次のシリアルフラッシュメモリへのアクセス処理を行うためには、ライトしたデータが反映されるまでウェイトする必要があります。

サンプルコードでは、このウェイト処理を Userdef_SPIBSC_SFLASH_WaitReady 関数で実現しています。

ご使用のシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのライト完了までウェイトするように、Userdef_SPIBSC_SFLASH_WaitReady 関数を実装してください。

サンプルコードでは、Status Register の WIP ビットをリードし、ライトが完了するまでウェイトする処理を行います。

図6.8に、サンプルコードのUserdef_SPIBSC_SFLASH_WaitReady関数のフローを示します。

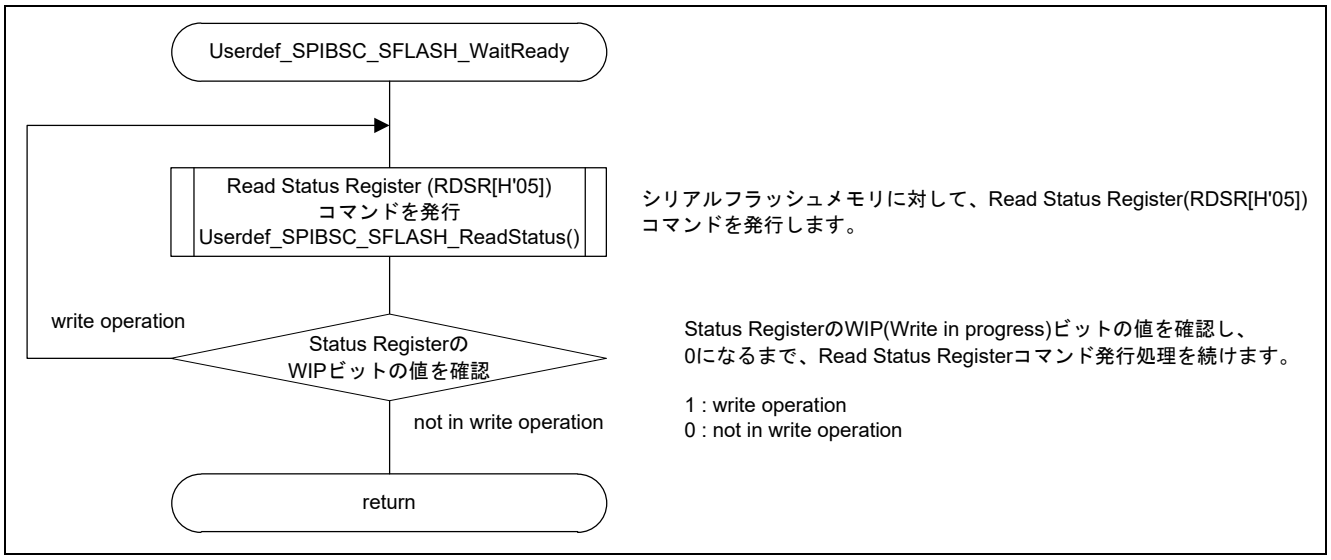


図6.8　Userdef_SPIBSC_SFLASH_WaitReady 関数のフロー

6.3.4　シリアルフラッシュメモリのステータスレジスタのリード

サンプルコードでは、シリアルフラッシュメモリの Status Register のリード処理を Userdef_SPIBSC_SFLASH_ReadStatus 関数で実現しています。

ご使用のシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのステータスレジスタをリードするように、Userdef_SPIBSC_SFLASH_ReadStatus 関数を実装してください。

図6.9に、サンプルコードのUserdef_SPIBSC_SFLASH_ReadStatus関数のフローを示します。

図6.9　Userdef_SPIBSC_SFLASH_ReadStatus 関数のフロー

表6.10　手動モード用のコマンド設定テーブル gs_command_table[0]：READ STATUS コマンド

メンバ名	説明	設定値
uint8_t command_name[20]	コマンド識別の文字列	"READ STATUS"
uint8_t cmd	コマンドのコード	**0x05**
uint8_t cmd_width	コマンドのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t cmd_output_enable	コマンドの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t ocmd	オプショナルコマンドのコード	0x00
uint8_t ocmd_width	オプショナルコマンドのビット幅	SPIBSC_1BIT_WIDTH
uint8_t ocmd_enable	オプショナルコマンドの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t addr_width	アドレスのビット幅	SPIBSC_1BIT_WIDTH
uint8_t addr_output_enable	アドレスの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t addr_sdr_ddr	アドレスの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opdata_width	オプションデータのビット幅	SPIBSC_1BIT_WIDTH
uint8_t opdata_output_enable	オプションデータの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t opdata_ddr_enable	オプションデータの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opd3	Option Data3(8bit)の設定(1 番目に出力)	0x00
uint8_t opd2	Option Data2(8bit)の設定(2 番目に出力)	0x00
uint8_t opd1	Option Data1(8bit)の設定(3 番目に出力)	0x00
uint8_t opd0	Option Data0(8bit)の設定(4 番目に出力)	0x00
uint8_t dummy_cycle_output_enable	ダミーサイクルの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t dummy_cycle_count	ダミーサイクル数	0x00
uint8_t transfer_data_width	転送データのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t transfer_data_sdr_ddr	転送データの転送方式	**SPIBSC_SDR_TRANSFER**

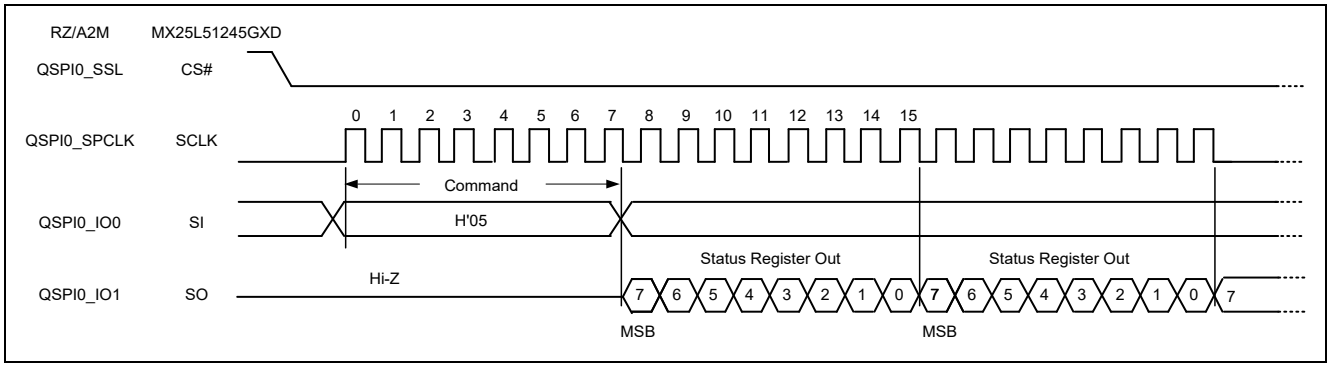


図6.10　READ STATUS コマンドの波形フォーマット（参考）

6.3.5　シリアルフラッシュメモリのコンフィグレーションレジスタのリード

サンプルコードでは、シリアルフラッシュメモリの Configuration Register のリード処理を Userdef_SPIBSC_SFLASH_ReadConfig 関数で実現しています。

ご使用のシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのコンフィグレーションレジスタをリードするように、Userdef_SPIBSC_SFLASH_ReadConfig 関数を実装してください。

図6.11に、サンプルコードのUserdef_SPIBSC_SFLASH_ReadConfig関数のフローを示します。

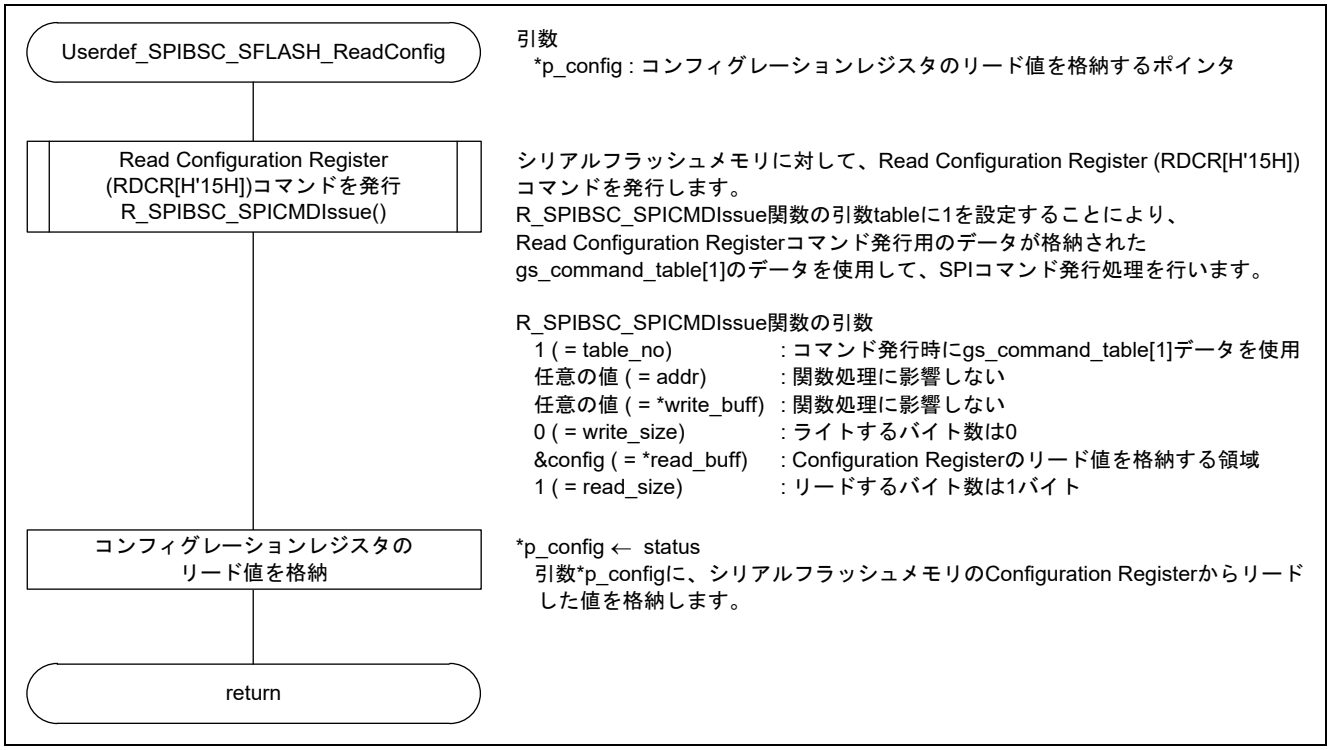


図6.11　Userdef_SPIBSC_SFLASH_ReadConfig 関数のフロー

表6.11　手動モード用のコマンド設定テーブル gs_command_table[1]：READ CONFIGURATION コマンド

メンバ名	説明	設定値
uint8_t command_name[20]	コマンド識別の文字列	"READ CONFIGURATION"
uint8_t cmd	コマンドのコード	**0x15**
uint8_t cmd_width	コマンドのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t cmd_output_enable	コマンドの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t ocmd	オプショナルコマンドのコード	0x00
uint8_t ocmd_width	オプショナルコマンドのビット幅	SPIBSC_1BIT_WIDTH
uint8_t ocmd_enable	オプショナルコマンドの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t addr_width	アドレスのビット幅	SPIBSC_1BIT_WIDTH
uint8_t addr_output_enable	アドレスの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t addr_sdr_ddr	アドレスの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opdata_width	オプションデータのビット幅	SPIBSC_1BIT_WIDTH
uint8_t opdata_output_enable	オプションデータの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t opdata_ddr_enable	オプションデータの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opd3	Option Data3(8bit)の設定(1 番目に出力)	0x00
uint8_t opd2	Option Data2(8bit)の設定(2 番目に出力)	0x00
uint8_t opd1	Option Data1(8bit)の設定(3 番目に出力)	0x00
uint8_t opd0	Option Data0(8bit)の設定(4 番目に出力)	0x00
uint8_t dummy_cycle_output_enable	ダミーサイクルの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t dummy_cycle_count	ダミーサイクル数	0x00
uint8_t transfer_data_width	転送データのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t transfer_data_sdr_ddr	転送データの転送方式	**SPIBSC_SDR_TRANSFER**

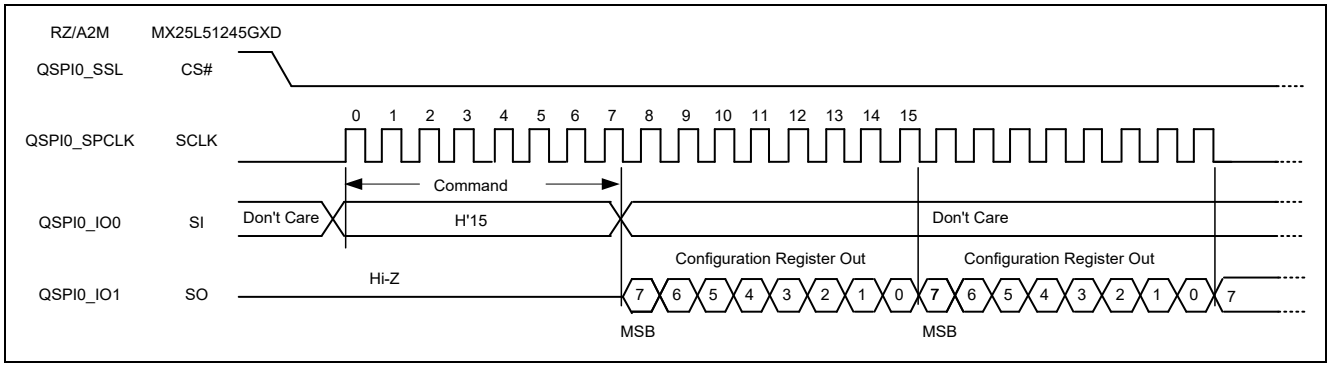


図6.12　READ CONFIGURATION コマンドの波形フォーマット（参考）

6.3.6　シリアルフラッシュメモリのライト許可

シリアルフラッシュメモリのレジスタ（Status Register および Configuration Register）にライトするためには、事前にシリアルフラッシュメモリのライトを許可にすることが必要なシリアルフラッシュメモリがあります。サンプルコードでは、この処理を Userdef_SPIBSC_SFLASH_WriteEnable 関数で実現しています。

ご使用のシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのライト許可が行えるように、Userdef_SPIBSC_SFLASH_WriteEnable 関数を実装してください。サンプルコードでは、Write Enable コマンド（WREN [H'06]）を発行することで、ライト許可（Status Register の WEL ビットを"1"に設定）に変更する処理を行います。

図6.13に、サンプルコードのUserdef_SPIBSC_SFLASH_WriteEnable関数のフローを示します。

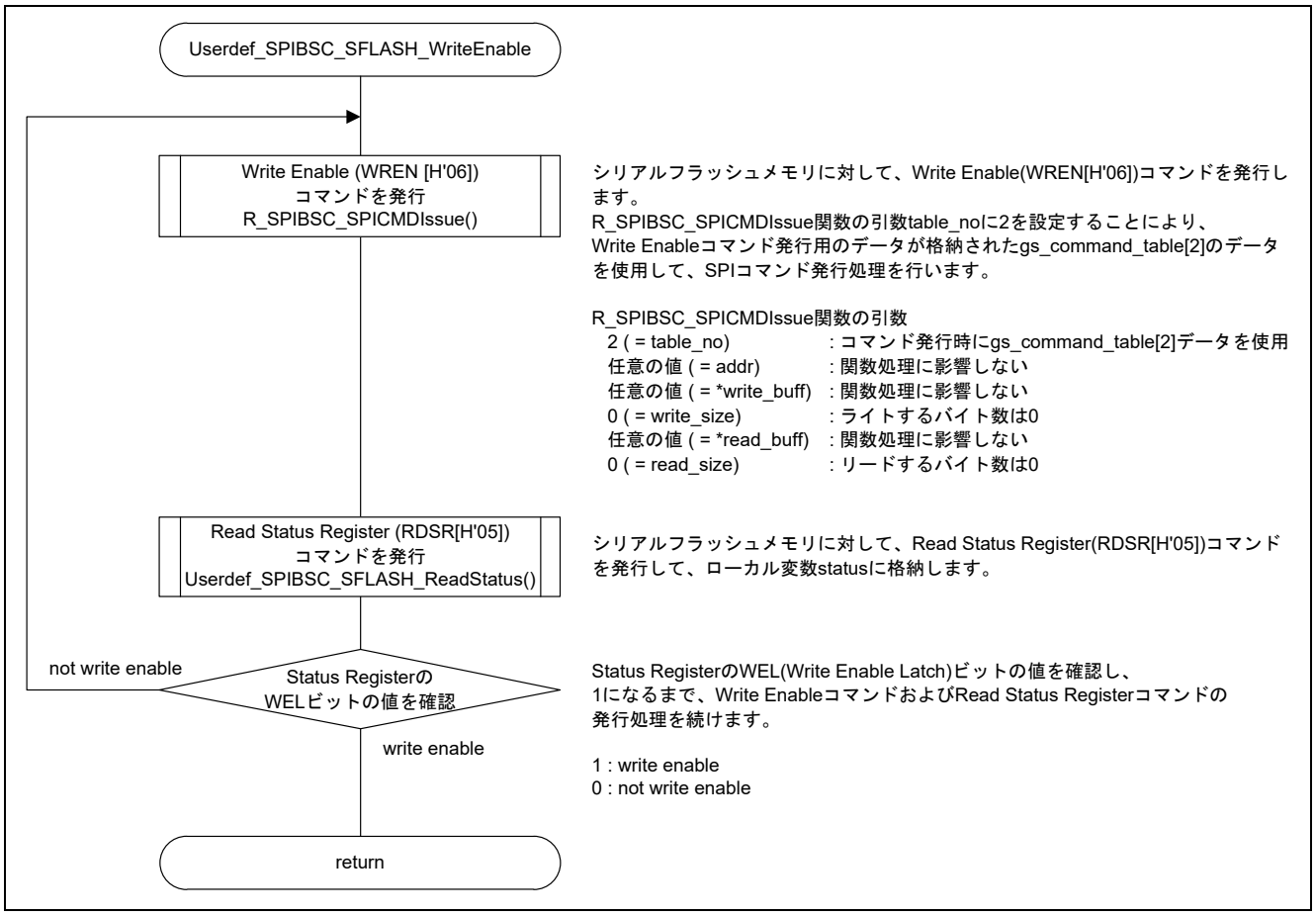


図6.13　Userdef_SPIBSC_SFLASH_WriteEnable 関数のフロー

表6.12　手動モード用のコマンド設定テーブル gs_command_table[2]：WRITE ENABLE コマンド

メンバ名	説明	設定値
uint8_t command_name[20]	コマンド識別の文字列	"WRITE ENABLE"
uint8_t cmd	コマンドのコード	**0x06**
uint8_t cmd_width	コマンドのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t cmd_output_enable	コマンドの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t ocmd	オプショナルコマンドのコード	0x00
uint8_t ocmd_width	オプショナルコマンドのビット幅	SPIBSC_1BIT_WIDTH
uint8_t ocmd_enable	オプショナルコマンドの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t addr_width	アドレスのビット幅	SPIBSC_1BIT_WIDTH
uint8_t addr_output_enable	アドレスの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t addr_sdr_ddr	アドレスの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opdata_width	オプションデータのビット幅	SPIBSC_1BIT_WIDTH
uint8_t opdata_output_enable	オプションデータの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t opdata_ddr_enable	オプションデータの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opd3	Option Data3(8bit)の設定(1 番目に出力)	0x00
uint8_t opd2	Option Data2(8bit)の設定(2 番目に出力)	0x00
uint8_t opd1	Option Data1(8bit)の設定(3 番目に出力)	0x00
uint8_t opd0	Option Data0(8bit)の設定(4 番目に出力)	0x00
uint8_t dummy_cycle_output_enable	ダミーサイクルの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t dummy_cycle_count	ダミーサイクル数	0x00
uint8_t transfer_data_width	転送データのビット幅	SPIBSC_1BIT_WIDTH
uint8_t transfer_data_sdr_ddr	転送データの転送方式	SPIBSC_SDR_TRANSFER

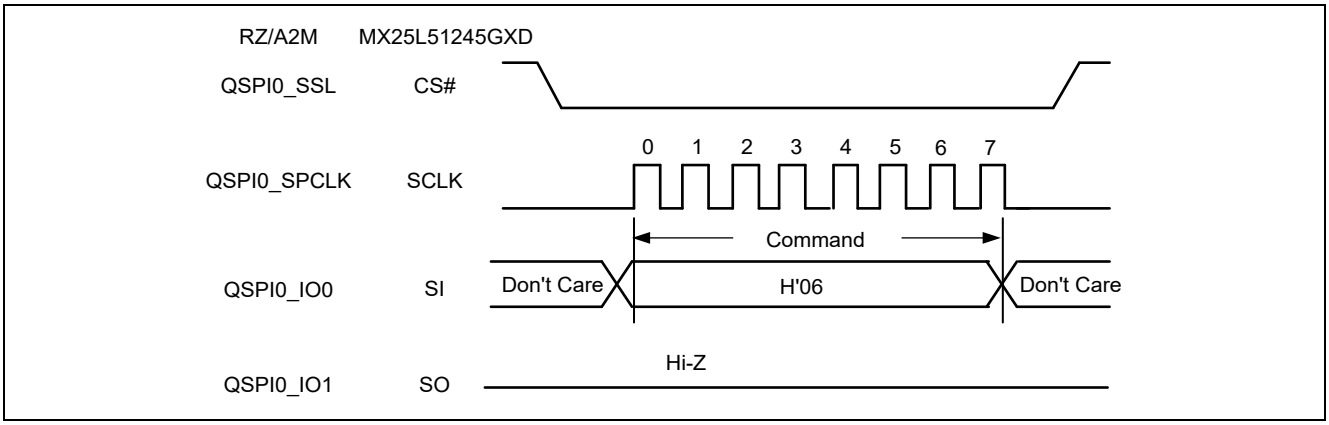


図6.14　WRITE ENABLE コマンドの波形フォーマット（参考）

6.3.7　シリアルフラッシュメモリのステータス/コンフィグレーションレジスタのライト

サンプルコードでは、シリアルフラッシュメモリにあるステータスレジスタの QE ビットと、コンフィグレーションレジスタの DC[1:0]ビットへのライト処理を Userdef_SPIBSC_SFLASH_WriteStatus 関数で実現しています。

ご使用のシリアルフラッシュメモリの仕様に合わせて、シリアルフラッシュメモリのステータスレジスタおよびコンフィグレーションレジスタへの設定値をライトするように、
Userdef_SPIBSC_SFLASH_WriteStatus 関数を実装してください。

図6.15に、サンプルコードのUserdef_SPIBSC_SFLASH_WriteStatus関数のフローを示します。

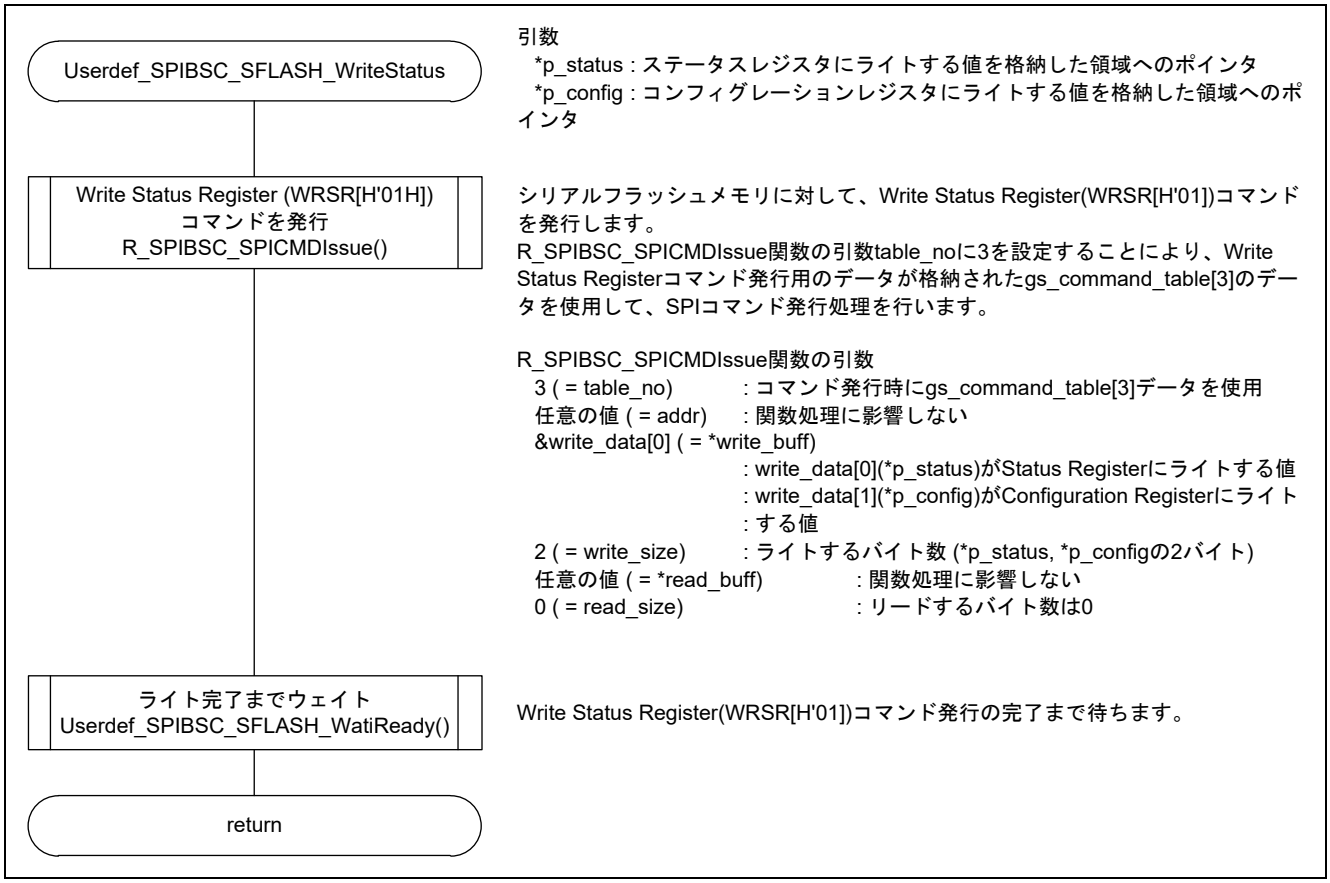


図6.15　Userdef_SPIBSC_SFLASH_WriteStatus 関数のフロー

表6.13　手動モード用のコマンド設定テーブル gs_command_table[3]：WRITE STATUS コマンド

メンバ名	説明	設定値
uint8_t command_name[20]	コマンド識別の文字列	"WRITE STATUS"
uint8_t cmd	コマンドのコード	**0x01**
uint8_t cmd_width	コマンドのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t cmd_output_enable	コマンドの出力有無	**SPIBSC_OUTPUT_ENABLE**
uint8_t ocmd	オプショナルコマンドのコード	0x00
uint8_t ocmd_width	オプショナルコマンドのビット幅	SPIBSC_1BIT_WIDTH
uint8_t ocmd_enable	オプショナルコマンドの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t addr_width	アドレスのビット幅	SPIBSC_1BIT_WIDTH
uint8_t addr_output_enable	アドレスの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t addr_sdr_ddr	アドレスの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opdata_width	オプションデータのビット幅	SPIBSC_1BIT_WIDTH
uint8_t opdata_output_enable	オプションデータの出力設定	SPIBSC_OUTPUT_DISABLE
uint8_t opdata_ddr_enable	オプションデータの転送方式	SPIBSC_SDR_TRANSFER
uint8_t opd3	Option Data3(8bit)の設定(1 番目に出力)	0x00
uint8_t opd2	Option Data2(8bit)の設定(2 番目に出力)	0x00
uint8_t opd1	Option Data1(8bit)の設定(3 番目に出力)	0x00
uint8_t opd0	Option Data0(8bit)の設定(4 番目に出力)	0x00
uint8_t dummy_cycle_output_enable	ダミーサイクルの出力有無	SPIBSC_OUTPUT_DISABLE
uint8_t dummy_cycle_count	ダミーサイクル数	0x00
uint8_t transfer_data_width	転送データのビット幅	**SPIBSC_1BIT_WIDTH**
uint8_t transfer_data_sdr_ddr	転送データの転送方式	**SPIBSC_SDR_TRANSFER**

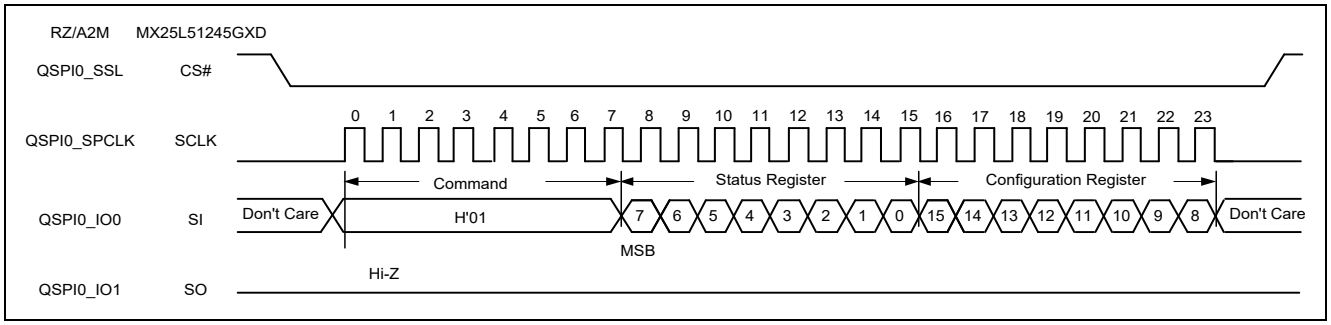


図6.16　WRITE STATUS コマンドの波形フォーマット（参考）

7. サンプルコードの注意事項

7.1 外部アドレス空間リードモードでアクセス可能な領域

外部アドレス空間リードモードでアクセス可能な領域は、SPI マルチ I/O バス空間に割り当てられた 256MB の領域（H'2000_0000 番地～H'2FFF_FFFF 番地）です。SPIBSC はこの領域へのアクセスをシリアルフラッシュメモリの H'0000_0000 番地～H'0FFF_FFFF 番地に変換してアクセスを行います。サンプルコードでは、シリアルフラッシュメモリの H'0000_0000 番地～H'0FFF_FFFF 番地の 256MB の領域に対してアクセスが可能です。RZ/A2M では SPIBSC のデータリード拡張アドレス設定レジスタ（DREAR）を制御して、SPI マルチ I/O バス空間に割り付けるシリアルフラッシュメモリのアドレスを変更することにより、256MB を超える領域についてもアクセスが可能ですが、DREAR を制御する前の領域にはアクセスができなくなるため、サンプルコードでは 256MB を超える領域へのアクセスには対応せず、シリアルフラッシュメモリの H'0000_0000 番地～H'0FFF_FFFF 番地のみアクセス可能な仕様としています。

なお、手動モードでは、4 バイトアドレスを使用したアクセスに対応しており、シリアルフラッシュメモリの 4G バイトの領域にアクセスが可能です。

8. 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。

9. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

RZ/A2Mグループ ユーザーズマニュアル ハードウェア編
（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。）

RTK7921053C00000BE（RZ/A2M CPUボード）ユーザーズマニュアル
（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。）

RTK79210XXB00000BE（RZ/A2M SUBボード）ユーザーズマニュアル
（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。）

Arm Architecture Reference Manual ARMv7-A and ARMv7-R edition Issue C
（最新版を Arm ホームページから入手してください。）

Arm Cortex™-A9 Technical Reference Manual Revision: r4p1
（最新版を Arm ホームページから入手してください。）

Arm Generic Interrupt Controller Architecture Specification - Architecture version2.0
（最新版を Arm ホームページから入手してください。）

Arm CoreLink™ Level 2 Cache Controller L2C-310 Technical Reference Manual Revision: r3p3
（最新版を Arm 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：統合開発

統合開発環境 e^2 studio のユーザーズマニュアルは、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。
（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。）

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
Rev.1.00	Dec.18.18	ー	初版発行
Rev.1.10	Mar.30.20	ー	CKIO が出力されるようにサンプルコードの内容を修正
		P6	表1.1　使用する周辺機能と用途 OSTM チャネル 2 を追加
		P7	表2.1　動作確認条件（1/2） コンパイルオプション"-mthumb-interwork"を削除
		P19	表5.5　周辺機能の設定内容 OSTM チャネル 2 を追加
		P21	表5.6　ローダプログラムで使用するセクション名とオブジェクト名一覧 r_memclk_setup 関数、r_spibsc_setup 関数を追加したため、これらの関数のセクションの記載を追加
		P38 P41 P48 P49	r_memclk_setup 関数を追加したため、関数一覧、関数仕様およびフローチャートに追加 • 表5.23　サンプル関数一覧サンプル関数一覧 • 5.9　関数仕様 • 図5.4　ローダプログラム（全体）のフローチャート • 5.10.2　メモリクロックの設定処理
		P38 P41 P51	R_SPIBSC_Setup 関数を追加したため、関数一覧、関数仕様およびフローチャートに追加 • 表5.24　API関数一覧 • 5.9　関数仕様 • 5.10.4　SPIBSCとシリアルフラッシュメモリの初期設定
		P39 P47	Userdef_PreHardwareSetup 関数および Userdef_PostHardwareSetup 関数を追加したため、関数一覧および関数仕様に追加 • 表5.25　ユーザ定義関数一覧 • 5.9　関数仕様
		P50	5.10.3　ブートに使用するハードウェアの初期設定 R_SPIBSC_Setup 関数、Userdef_PreHardwareSetup 関数および Userdef_PostHardwareSetup 関数を追加したため、R_SC_HardwareSetup 関数の仕様を変更
		P53 P54	R_SPIBSC_Init 関数において、SDR モードのタイミング調整処理の順序を変更したため、フローチャートの記載を変更 • 図5.8　SPIBSC初期設定のフローチャート（1/2） • 図5.9　SPIBSC初期設定のフローチャート（2/2）
		P73	図6.3　SPIBSCとシリアルフラッシュメモリの設定処理のモジュール階層図 モジュール階層図に、追加した R_SPIBSC_Setup 関数を記載

Rev.	発行日	改訂内容	
		ページ	ポイント
Rev.1.10	Mar.30.20	P14 P30 P36	CMNCR の IO0FV ビットの設定に関する注意事項を追加 • 表5.2　ブート起動用内蔵ROMプログラムおよびローダプログラムの設定内容（1/3） • 表5.15　SPIBSC外部アドレス空間リードモード設定構造体（st_spibsc_xip_config_t）（5/5） • 表5.21　SPIBSC 手動モード設定構造体（st_spibsc_manual_mode_command_config_t）(6/6)
Rev.1.20	Oct.12.20	P43	R_SPIBSC_XipStopAccess 関数において、QSPIn_SSL をネゲートした後の処理に、ネゲートされたことを確認する処理を追加

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

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。