

SH7268/SH7269 グループ

E10A-USB フラッシュメモリダウンロード機能の応用例 (シリアルフラッシュメモリへのダウンロード)

R01AN0883JJ0100

Rev. 1.00

2011.11.30

要旨

E10A-USB エミュレータにはフラッシュメモリにロードモジュールをダウンロードする機能があります。本機能はフラッシュメモリにアクセスするために、ダウンロードプログラム（以下、FMTOOL と称します）を使用します。

本アプリケーションノートは、この FMTOOL を応用したシリアルフラッシュメモリへのダウンロード方法について説明します。

対象デバイス

SH7268/SH7269 グループ

以下、総称して「SH7269」として説明します。

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	4
2. 動作確認条件	5
3. 関連アプリケーションノート	5
4. 周辺機能説明	6
5. ハードウェア説明	7
5.1 使用端子一覧	7
5.2 参考回路	8
6. ソフトウェア説明	9
6.1 動作概要	9
6.1.1 バッチファイル	9
6.1.2 イレーズモジュール	9
6.1.3 ライトモジュール	10
6.2 ファイル構成	11
6.3 定数一覧	12
6.4 構造体/共用体一覧	13
6.5 変数一覧	14
6.6 関数一覧	14
6.7 関数仕様	15
6.8 フローチャート	20
6.8.1 イレーズモジュール	20
6.8.2 ライトモジュール	21
6.9 注意事項	22
6.9.1 ロードモジュールへのダミーデータ追加	22
6.9.2 ロードモジュール間のセクタ共有禁止	23
7. 応用例	24
7.1 ユーザプログラムダウンロード手順	24
7.1.1 ダウンロード環境の準備	24
7.1.2 バッチファイルの登録	24
7.1.3 Configurationダイアログボックスの設定	25
7.1.4 ダウンロードモジュールの追加	26
7.1.5 ユーザプログラムのダウンロード	26
7.2 シリアルフラッシュブートへの応用	27
7.2.1 セクション配置	27

7.2.2	ダミーデータの追加	27
7.2.3	ロードモジュールのダウンロード方法	27
7.3	FMTOOLのカスタマイズ	28
7.3.1	サンプルコードに対応するデバイス仕様	28
7.3.2	カスタマイズの内容	28
7.4	FMTOOLのデバッグ手順	29
7.4.1	デバッグコードの実装	29
7.4.2	デバッグ環境の準備	29
7.4.3	FMTOOLのダウンロード	29
7.4.4	FMTOOLのデバッグ	29
8.	サンプルコード	30
9.	参考ドキュメント	30

1. 仕様

SPI マルチ I/O バス空間に配置したロードモジュールをシリアルフラッシュメモリにダウンロードします。ダウンロードにはシリアルフラッシュメモリに対応した FMTOOL を使用します。FMTOOL は、SPI マルチ I/O バスコントローラを使用して、マルチ I/O (データバス幅=4 ビット) に対応したシリアルフラッシュメモリをアクセスします。

表 1.1に使用する周辺機能と用途を、図 1.1にFMTOOLを使用したダウンロード処理の流れを示します。

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	シリアルフラッシュメモリへのダウンロード
H-UDI	E10A-USB エミュレータの接続

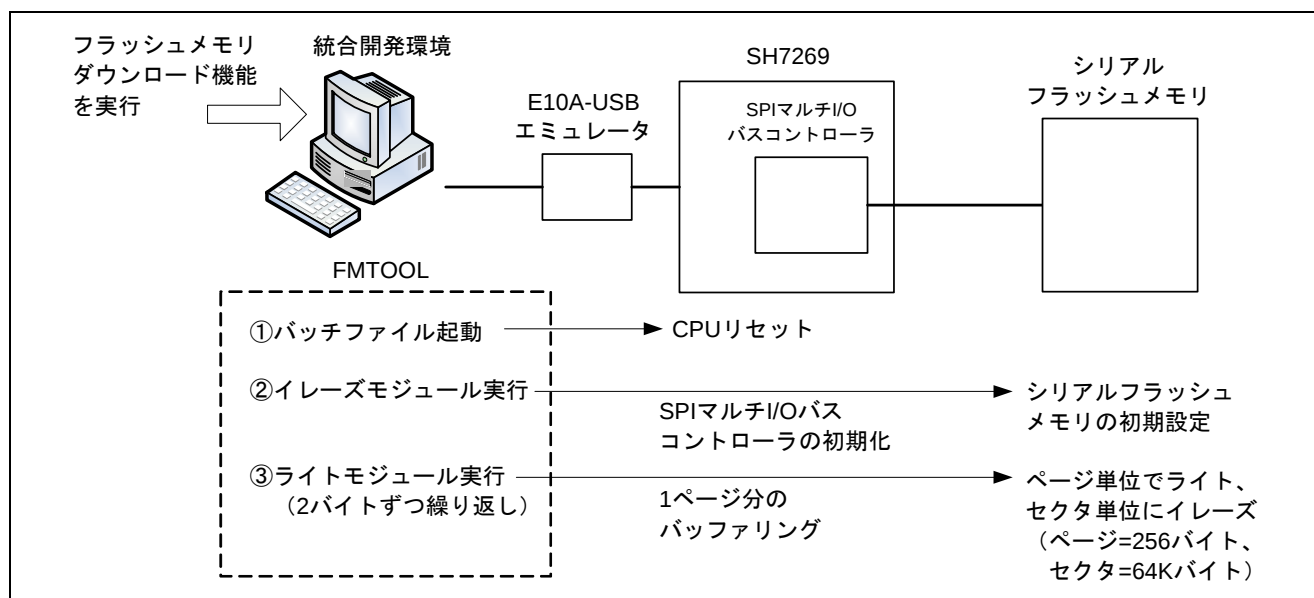


図 1.1 FMTOOL を使用したダウンロード処理の流れ

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	SH7269
使用デバイス	マルチ I/O バス対応シリアルフラッシュメモリ メーカー : Spansion 社 型名 : S25FL032P0XMF101
動作周波数	CPU 内部クロック (I ϕ) : 266.67MHz 内部クロック (B ϕ) : 133.33MHz 周辺クロック 1 (P1 ϕ) : 66.67MHz 周辺クロック 0 (P0 ϕ) : 33.33MHz
動作電圧	電源電圧 (I/O) : 3.3V 電源電圧 (内部) : 1.25V
統合開発環境	ルネサス エレクトロニクス製 High-performance Embedded Workshop Ver.4.07.00
C コンパイラ	ルネサス エレクトロニクス製 SuperH RISC engine ファミリ C/C++コンパイラパッケージ Ver.9.03 Release02 コンパイラオプション -cpu=sh2afpu -fpu=single -object="\$(CONFIGDIR)\\$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 -nologo (統合開発環境のデフォルト設定を使用しています)
使用ボード	R0K572690C000BR

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。

- SH7268/SH7269 グループ SPI マルチ I/O バスコントローラを使用したシリアルフラッシュメモリからのブート例(R01AN0663JJ)
- SH7268/SH7269 グループ SPI マルチ I/O バスコントローラ シリアルフラッシュメモリ接続例 (R01AN0671JJ)
- E10A-USB エミュレータ用 フラッシュメモリ ダウンロードプログラム アプリケーションノート (R01AN0957JJ)

4. 周辺機能説明

SPI マルチ I/O バスコントローラについて補足します。基本的な内容はハードウェアマニュアルに記載しています。

SPI マルチ I/O バスコントローラには、SPI 動作モードと外部アドレス空間リードモードがあります。シリアルフラッシュメモリに書き込んだプログラムを直接フェッチする場合には、外部アドレス空間リードモードを使用しますが、シリアルフラッシュメモリをイレーズまたはライトする場合は SPI 動作モードを使用する必要があります。

SPI 動作モードの設定手順については、アプリケーションノート「SH7268/SH7269 グループ SPI マルチ I/O バスコントローラ シリアルフラッシュメモリ接続例(R01AN0671JJ)」を参照してください。

5. ハードウェア説明

5.1 使用端子一覧

表 5.1に使用端子と機能を示します。

表 5.1 使用端子と機能

端子名	入出力	内容
SPBCLK	出力	シリアルフラッシュメモリへのクロック出力
SPBSSL	出力	シリアルフラッシュメモリへのデバイス選択信号出力
SPBIO0_0	入出力	シリアルフラッシュメモリへのデータ入出力 (ビット 0)
SPBIO1_0	入出力	シリアルフラッシュメモリへのデータ入出力 (ビット 1)
SPBIO2_0	入出力	シリアルフラッシュメモリへのデータ入出力 (ビット 2)
SPBIO3_0	入出力	シリアルフラッシュメモリへのデータ入出力 (ビット 3)
MD_BOOT0	入力	ブートモードの選択 (ビット 0)
MD_BOOT1	入力	ブートモードの選択 (ビット 1)
MD_BOOT2	入力	ブートモードの選択 (ビット 2)
AUDCK	出力	E10A-USB エミュレータ (38pin) へのクロック出力
AUDATA0	出力	E10A-USB エミュレータ (38pin) へのアドレス出力 (ビット 0)
AUDATA1	出力	E10A-USB エミュレータ (38pin) へのアドレス出力 (ビット 1)
AUDATA2	出力	E10A-USB エミュレータ (38pin) へのアドレス出力 (ビット 2)
AUDATA3	出力	E10A-USB エミュレータ (38pin) へのアドレス出力 (ビット 3)
AUDSYNC#	出力	E10A-USB エミュレータ (38pin) への同期信号出力
TCK	入力	E10A-USB エミュレータからのクロック入力
TMS	入力	E10A-USB エミュレータからのモード選択
TRST#	入力	E10A-USB エミュレータからのリセット入力
TDI	入力	E10A-USB エミュレータからのデータ入力
TDO	出力	E10A-USB エミュレータへのデータ出力
ASEBRKAK#/ASEBRK#	入出力	ブレーク要求と応答
RES#	入力	システムリセット信号
ASEMD#	入力	ASE モードの選択

【注】 #は負論理 (またはアクティブロー) を示す記号です。

5.2 参考回路

図 5.1に接続例を示します。

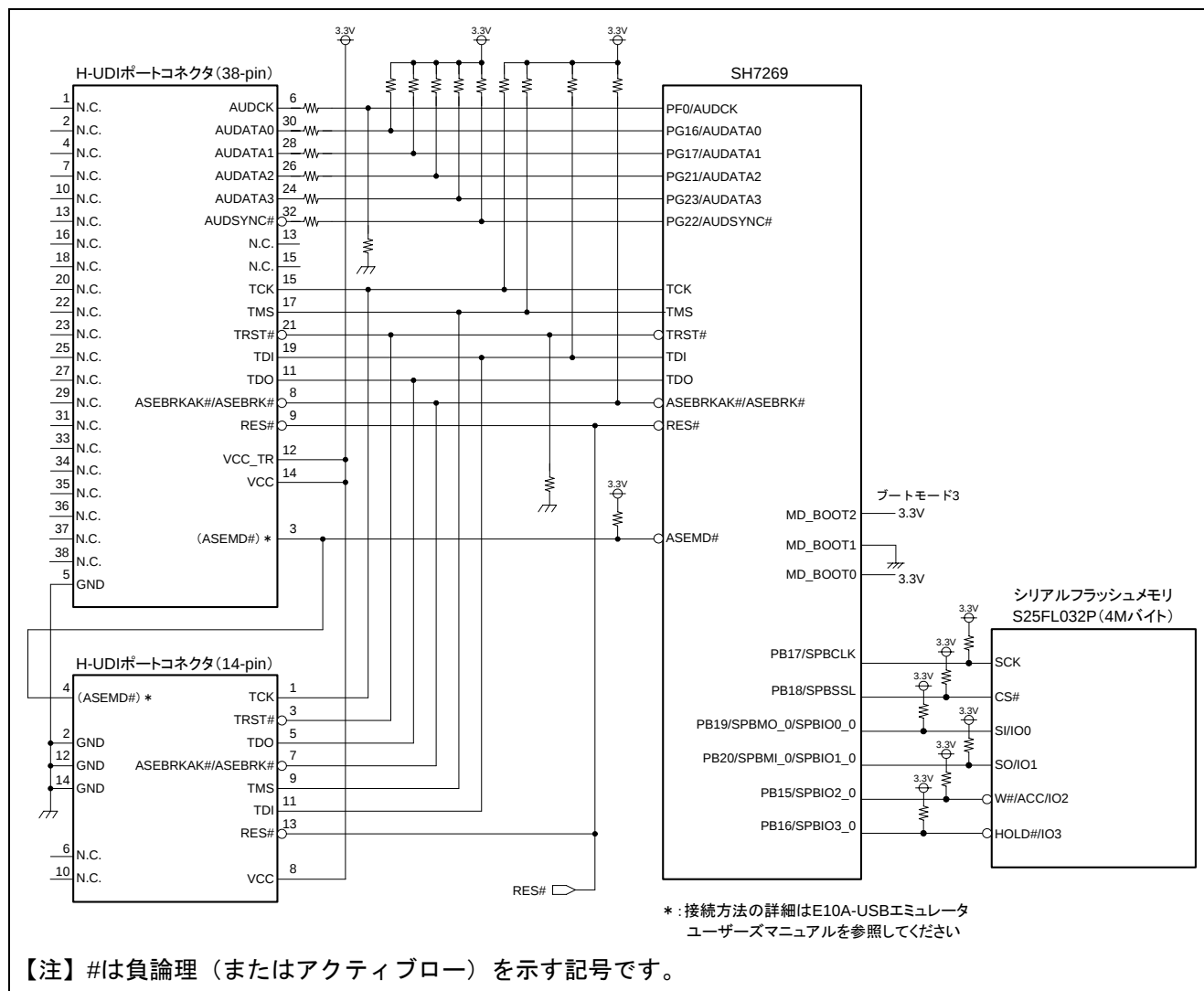


図 5.1 接続例

6. ソフトウェア説明

6.1 動作概要

FMTOOL は、イレーズモジュールとライトモジュールの 2 つのプログラムから構成されます。E10A-USB エミュレータは、この 2 つのプログラムを使用して、フラッシュメモリにプログラムデータを書き込みます。イレーズモジュールおよびライトモジュールの詳細仕様は、E10A-USB エミュレータ ユーザーズマニュアル「6.22 フラッシュメモリへのダウンロード機能」を参照してください。

6.1.1 バッチファイル

FMTOOL を実行する前にリセットコマンドを実行して SH7269 を初期化します。リセットコマンドについては統合開発環境のマニュアルを参照してください。

6.1.2 イレーズモジュール

図 6.1 にイレーズモジュールの概要を示します。イレーズモジュールは高速内蔵RAM上で実行します。イレーズモジュールには、通常、フラッシュメモリへのチップイレーズ処理を割り当てますが、サンプルコードでは初期設定処理を割り当てています。初期設定処理では、SPIマルチI/Oバスコントローラの初期化と、シリアルフラッシュメモリのモード設定およびプロテクト解除を行います。

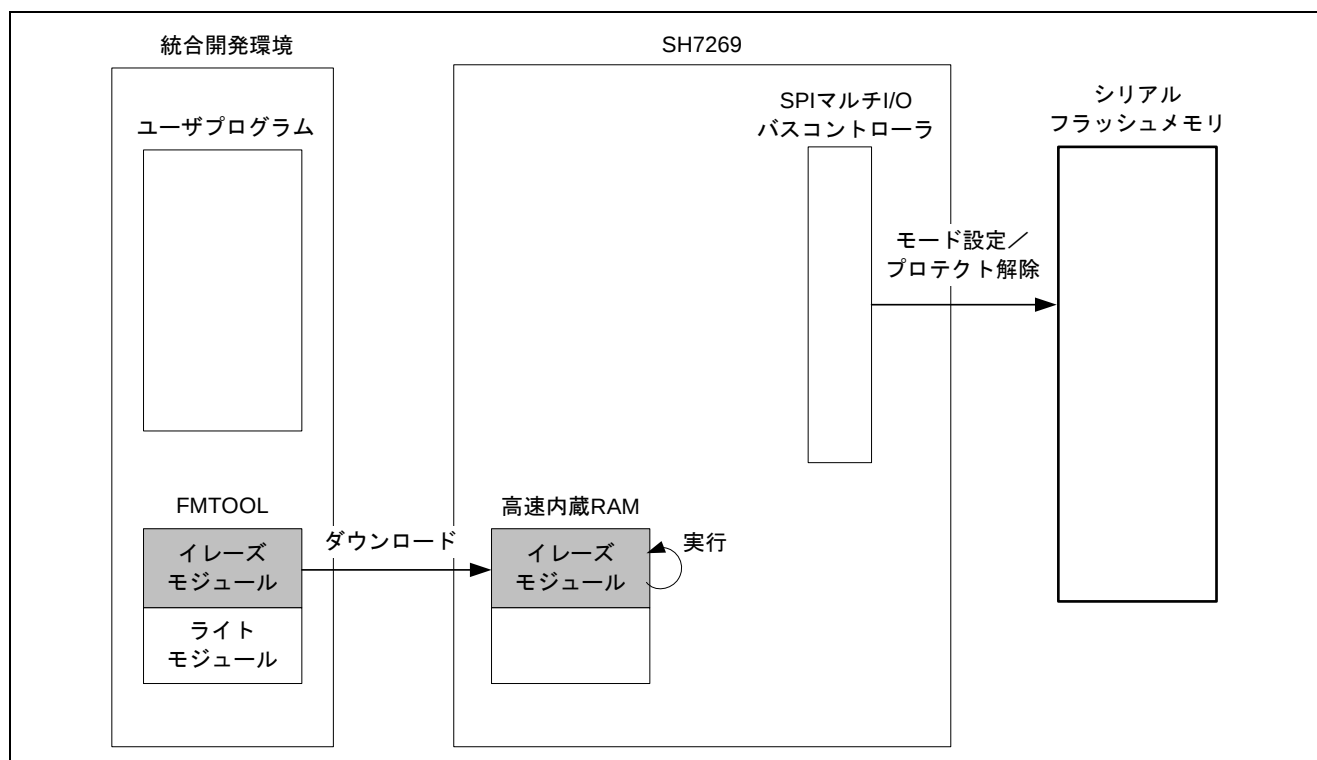


図 6.1 イレーズモジュールの概要

6.1.3 ライトモジュール

図 6.2にライトモジュールの概要を示します。ライトモジュールは高速内蔵RAM上で実行します。ライトモジュールは、ユーザプログラムのワードデータを引数として受け取ります。ライトモジュールはこれをバッファリングして、ページの切り換えタイミングでシリアルフラッシュメモリにライトします。また引数で指定された書き込み先アドレスが未消去セクタのものであれば、セクタイレーズを行ってから書き込みます。

なお、SPI マルチ I/O バス空間の先頭番地 (H'1800 0000 番地) がシリアルフラッシュメモリの先頭番地 (H'0000 0000 番地) に対応するようにライトモジュール内でアドレス変換を行っています。

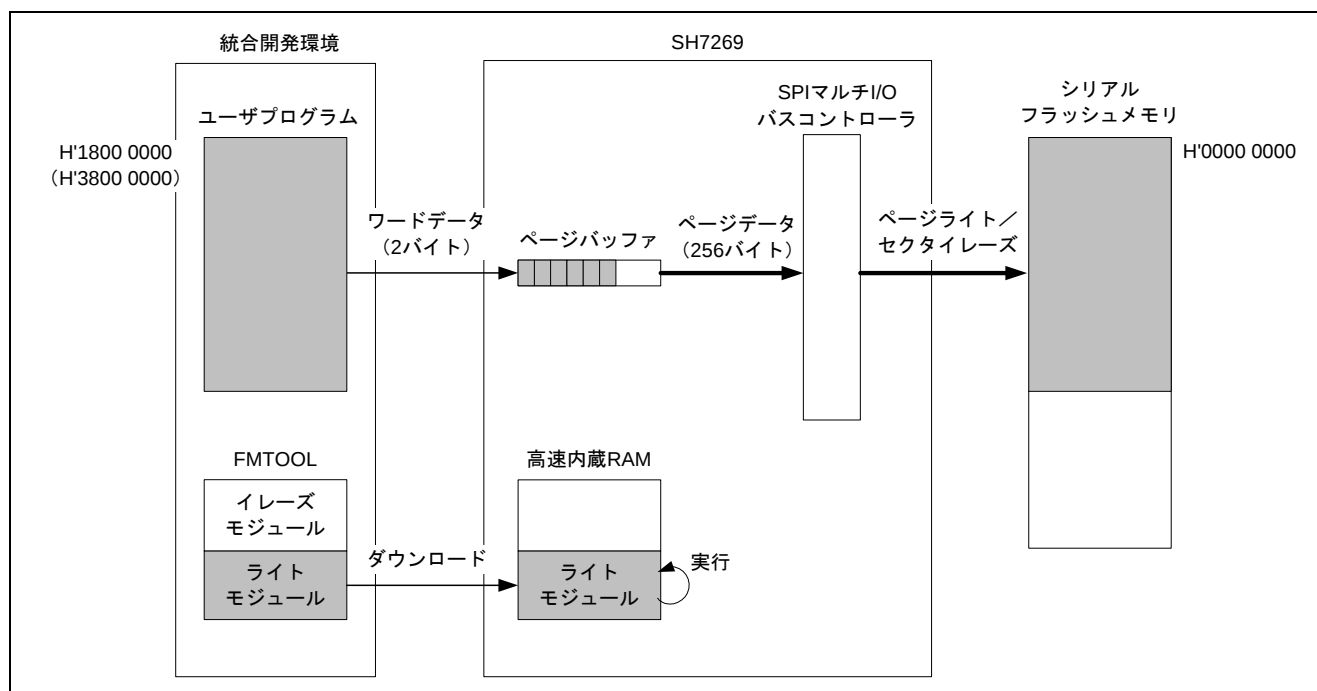


図 6.2 ライトモジュールの概要

6.2 ファイル構成

表 6.1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 6.1 ファイル構成

ファイル名	概要	備考
fmttool_entry.src	FMTOOL のエントリモジュール	イレーズモジュールと ライトモジュールのエントリ
fmttool_main.c	FMTOOL のメインモジュール	
fmttool_cpg.c	CPG の初期化処理	
fmttool_cpg.h	fmttool_cpg.c の I/F 定義	
r_sf_spibsc.c	シリアルフラッシュメモリ操作処理	マルチ I/O 対応版
r_sf_spibsc.h	r_sf_spibsc.c の I/F 定義	
io_spibsc.c	SPI マルチ I/O バスコントローラの 制御処理	
io_spibsc.h	io_spibsc.c の I/F 定義	
sh7269_spibsc_fmttool.hdc	バッチファイル	統合開発環境に登録
fmttool_debug.c	FMTOOL 用デバッグコード	デバッグ時のみ使用

6.3 定数一覧

表 6.2にサンプルコードで使用する定数を示します。

表 6.2 サンプルコードで使用する定数

定数名	設定値	内容
SFLASH_DUAL	0	シリアルフラッシュメモリの2個接続はしない
SPI_QUAD	1	マルチ I/O バス対応のシリアルフラッシュメモリ
SF_PAGE_SIZE	256	ページサイズ (256 バイト)
PAGE_SIZE	SF_PAGE_SIZE	"
SF_SECTOR_SIZE	(64*1024)	セクタサイズ (64K バイト)
SECTOR_SIZE	SF_SECTOR_SIZE	"
SR_Init	0x000000F0	ステータスレジスタの初期値
DEFAULT_VALUE	0xFFFFFFFF	FMTOOL で使用する管理データの初期値
SFLASH_ADDRESS_MASK	0xFC000000	SPI マルチ I/O バス空間のアドレスをシリアルフラッシュメモリのアドレスに変換するためのマスク設定値
TYPE_BYTE	0x4220	ライトモジュールの R5 パラメータ (データアクセスサイズは、バイトサイズ)
TYPE_WORD	0x5720	ライトモジュールの R5 パラメータ (データアクセスサイズは、ワードサイズ)
TYPE_LONG	0x4C20	ライトモジュールの R5 パラメータ (データアクセスサイズは、ロングサイズ)
ENABLE_DEBUG_MODE	1	FMTOOL 用デバッグコードは有効

6.4 構造体/共用体一覧

図 6.3にサンプルコードで使用する構造体/共用体を示します。

```

/* ==== SPIマルチI/Oバスコントローラの転送制御を行うための構造体 ==== */
typedef struct{
    /* ---- SPIモードイネーブル設定レジスタ (SMENR) の設定値 ---- */
    uint32_t      cdb      :2;    /* コマンドビット幅 */
    uint32_t      ocdb     :2;    /* オプショナルコマンドビット幅*/
    uint32_t      adb      :2;    /* アドレスビット幅*/
    uint32_t      opdb     :2;    /* オプションデータビット幅 */
    uint32_t      spidb    :2;    /* 転送データビット幅 */
    uint32_t      cde      :1;    /* コマンドイネーブル */
    uint32_t      ocde     :1;    /* オプショナルコマンドイネーブル */
    uint32_t      ade      :4;    /* アドレスイネーブル */
    uint32_t      opde     :4;    /* オプションデータイネーブル */
    uint32_t      spide    :4;    /* 転送データイネーブル */

    /* ---- SPIモードコントロールレジスタ (SMCR) の設定値 ---- */
    uint32_t      sslkp    :1;    /* SPBSSL信号レベル保持 */
    uint32_t      spire    :1;    /* データリードイネーブル */
    uint32_t      spiwe    :1;    /* データライトイネーブル */
    uint32_t      :5;

    /* ---- SPIモードコマンド設定レジスタ (SMCMR) の設定値 ---- */
    uint8_t       cmd;          /* コマンド */
    uint8_t       ocmd;        /* オプショナルコマンド */

    /* ---- SPIモードアドレス設定レジスタ (SMADR) の設定値 ---- */
    uint32_t      addr;

    /* ---- SPIモードオプション設定レジスタ (SMOPR) の設定値 ---- */
    uint8_t       opd[4];      /* オプションデータ0~3 */

    /* ---- SPIモードリードデータレジスタ (SMRDR0, SMRDR1) の設定値 ---- */
    uint32_t      smrdr[2];

    /* ---- SPIモードライトデータレジスタ (SMWDR0, SMWDR1) の設定値 ---- */
    uint32_t      smwdr[2];

} st_spibsc_spimd_reg_t;

```

図 6.3 サンプルコードで使用する構造体/共用体

6.5 変数一覧

表 6.3にグローバル変数を、表 6.4にstatic型変数を示します。

表 6.3 グローバル変数

Type	Variable Name	Contents	Function Used
st_spibsc_spimd_reg_t	g_spibsc_spimd_reg	SPI マルチ I/O バスコントローラの設定 データ	R_SF_SPIBSC_EraseChip R_SF_SPIBSC_EraseSector R_SF_SPIBSC_ByteProgram R_SF_SPIBSC_ByteRead read_data_quad read_status read_config write_enable write_status

表 6.4 static 型変数

Type	Variable Name	Contents	Function Used
uint32_t	fmttool_pre_erase_sctno	イレーズ済みセクタの管理情報	fmttool_init, fmttool_write
uint32_t	fmttool_cur_page	バッファリングしているページの 先頭アドレス	fmttool_init, fmttool_write
uint16_t	fmttool_page_buf[PAGE_SIZE / sizeof(uint16_t)]	ページバッファ	fmttool_write

6.6 関数一覧

表 6.5に関数を示します。

表 6.5 関数

関数名	概要
_ERASE_ENTRY	イレーズモジュールのエントリ処理
_WRITE_ENTRY	ライトモジュールのエントリ処理
fmttool_init	イレーズモジュールのメイン処理（初期化处理）
fmttool_write	ライトモジュールのメイン処理（イレーズ／ライト処理）
R_SF_SPIBSC_GetBsz	シリアルフラッシュメモリ操作関数（データバス幅の検出）
R_SF_SPIBSC_SetBsz	“ （データバス幅の設定）
R_SF_SPIBSC_AllocateExspace	“ （外部アドレス空間リードモードの設定）
R_SF_SPIBSC_Init	“ （SPI マルチ I/O バスコントローラの初期設定および シリアルフラッシュメモリのモード設定）
R_SF_SPIBSC_CtrlProtect	“ （プロテクト制御）
R_SF_SPIBSC_EraseChip	“ （チップイレーズ処理）
R_SF_SPIBSC_EraseSector	“ （セクタイレーズ処理）
R_SF_SPIBSC_ByteProgram	“ （ライト処理）
R_SF_SPIBSC_ByteRead	“ （リード処理）※SPI 動作モードを使用したリード処理
fmttool_debug	FMTOOL 用デバッグ関数

6.7 関数仕様

サンプルコードの関数仕様を示します。

_ERASE_ENTRY

概要	イレーズモジュールのエントリ処理
ヘッダ	なし
宣言	<code>_ERASE_ENTRY:</code>
説明	イレーズモジュールのエントリ部で、H'FFF8 2000 番地に配置します。本モジュールは E10A-USB フラッシュメモリダウンロード機能によって起動されます。 本モジュールはスタックポインタの設定後、 <code>fmttool_init</code> 関数を実行します。
引数	R4 レジスタ : アクセスサイズ (バイト= H'4220、ワード= H'5720、ロング= H'4C00)
リターン値	なし
備考	アセンブリ言語で記述されています。

_WRITE_ENTRY

概要	ライトモジュールのエントリ処理
ヘッダ	なし
宣言	<code>_WRITE_ENTRY:</code>
説明	ライトモジュールのエントリ部で、H'FFF8 2100 番地に配置します。本モジュールは E10A-USB フラッシュメモリダウンロード機能によって起動されます。 本モジュールはスタックポインタの設定後、 <code>fmttool_write</code> 関数を実行します。
引数	R4 レジスタ : ライトデータが配置されているアドレス R5 レジスタ : アクセスサイズ (バイト= H'4220、ワード= H'5720、ロング= H'4C00) R6 レジスタ : ライトデータ R7 レジスタ : ベリファイフラグ (0=ベリファイなし、1=ベリファイあり)
リターン値	R0 レジスタ=0 の場合 : 正常終了 R0 レジスタ=1 の場合 : エラー終了
備考	アセンブリ言語で記述されています。

fmttool_init

概要	イレーズモジュールのメイン処理 (初期化処理)
ヘッダ	なし
宣言	<code>void fmttool_init(void);</code>
説明	SPI マルチ I/O バスコントローラとシリアルフラッシュメモリの初期化を行います。 本関数は FMTOOL のエントリポイント (<code>_ERASE_ENTRY</code>) から実行します。
引数	なし
リターン値	なし
備考	

fmtool_write

概要	ライトモジュールのメイン処理（イレーズ／ライト処理）		
ヘッダ	なし		
宣言	int32_t fmtool_write(uint32_t addr, int32_t access_size, uint32_t write_data, int32_t v_flag);		
説明	シリアルフラッシュメモリのイレーズおよびライト処理を行います。イレーズはセクタ単位、ライトはページ単位でシリアルフラッシュメモリをアクセスします。本関数は FMTOOL のエントリポイント（_WRITE_ENTRY）から実行します。		
引数	第一引数 : addr	:	ライトデータが配置されているアドレス
	第二引数 : size	:	アクセスサイズ (バイト= H'4220、ワード= H'5720、ロング= H'4C00)
	第三引数 : write_data	:	ライトデータ
	第四引数 : v_flag	:	ベリファイフラグ (0=ベリファイなし、1=ベリファイあり) ※未使用
リターン値	0 の場合 : 正常終了 負の場合 : エラー終了		
備考	アクセスサイズはワードサイズのみ対応します。		

R_SF_SPIBSC_GetBsz

概要	シリアルフラッシュメモリ操作関数（データバス幅の検出）		
ヘッダ	"r_sf_spibsc.h"		
宣言	int32_t R_SF_SPIBSC_GetBsz(void);		
説明	SPI マルチ I/O バスコントローラに設定したデータバス幅（接続デバイス数）を返します。		
引数	なし		
リターン値	1 の場合 : データバス幅 4 ビット（デバイス×1） 2 の場合 : データバス幅 8 ビット（デバイス×2）		
備考			

R_SF_SPIBSC_SetBsz

概要	シリアルフラッシュメモリ操作関数（データバス幅の設定）		
ヘッダ	"r_sf_spibsc.h"		
宣言	void R_SF_SPIBSC_SetBsz(int32_t bsz);		
説明	SPI マルチ I/O バスコントローラに、データバス幅（接続デバイス数）を設定します。		
引数	第一引数 : bsz	:	データバス幅（接続デバイス数）
リターン値	なし		
備考			

R_SF_SPIBSC_AllocateExspace

概要	シリアルフラッシュメモリ操作関数（外部アドレス空間リードモードの設定）
ヘッダ	"r_sf_spibsc.h"
宣言	void R_SF_SPIBSC_AllocateExspace(void);
説明	SPI マルチ I/O バスコントローラに、外部アドレス空間リードモードを設定します。
引数	なし
リターン値	なし
備考	

R_SF_SPIBSC_Init

概要	シリアルフラッシュメモリ操作関数（SPI マルチ I/O バスコントローラの初期設定およびシリアルフラッシュメモリのモード設定）
ヘッダ	"r_sf_spibsc.h"
宣言	void R_SF_SPIBSC_Init(void);
説明	SPI マルチ I/O バスコントローラの基本部分の初期設定を行います。またシリアルフラッシュメモリを Quad 動作モードに設定します。
引数	なし
リターン値	なし
備考	

R_SF_SPIBSC_CtrlProtect

概要	シリアルフラッシュメモリ操作関数（プロテクト制御）
ヘッダ	"r_sf_spibsc.h"
宣言	void R_SF_SPIBSC_CtrlProtect(en_sf_req_t req);
説明	シリアルフラッシュメモリのプロテクトを解除、またはセットします。
引数	第一引数 : req : プロテクトリクエスト (SF_REQ_PROTECT : プロテクトセット、 SF_REQ_UNPROTECT : プロテクト解除)
リターン値	なし
備考	

R_SF_SPIBSC_EraseChip

概要	シリアルフラッシュメモリ操作関数（チップイレーズ処理）
ヘッダ	"r_sf_spibsc.h"
宣言	void R_SF_SPIBSC_EraseChip(void);
説明	シリアルフラッシュメモリをチップイレーズします。
引数	なし
リターン値	なし
備考	

R_SF_SPIBSC_EraseSector

概要	シリアルフラッシュメモリ操作関数（セクタイレース処理）	
ヘッダ	"r_sf_spibsc.h"	
宣言	void R_SF_SPIBSC_EraseSector(int32_t sector_no);	
説明	シリアルフラッシュメモリをセクタイレースします。	
引数	第一引数 : sector_no	: イレースするセクタ番号
リターン値	なし	
備考		

R_SF_SPIBSC_ByteProgram

概要	シリアルフラッシュメモリ操作関数（ライト処理）	
ヘッダ	"r_sf_spibsc.h"	
宣言	void R_SF_SPIBSC_ByteProgram(uint32_t addr, uint8_t * buf, int32_t size);	
説明	引数で指定されたデータをシリアルフラッシュメモリにライトします。SPI_QUAD マクロで 1 を指定した場合は、Quad 対応のページプログラムコマンド（H'32）を使用し、0 を指定した場合は、Single 対応のページプログラムコマンド（H'02）を使用します。	
引数	第一引数 : addr	: ライトアドレス (シリアルフラッシュメモリのアドレス)
	第二引数 : buf	: ライトデータ（バッファの先頭アドレス）
	第三引数 : size	: データバイト数
リターン値	なし	
備考		

R_SF_SPIBSC_ByteRead

概要	シリアルフラッシュメモリ操作関数（リード処理）	
ヘッダ	"r_sf_spibsc.h"	
宣言	void R_SF_SPIBSC_ByteRead(uint32_t addr, uint8_t * buf, int32_t size);	
説明	引数で指定されたシリアルフラッシュメモリの領域をリードしてバッファに格納します。SPI_QUAD マクロで 1 を指定した場合は、Quad 対応のリードコマンド（H'6B）を使用し、0 を指定した場合は、Single 対応のリードコマンド（H'0B）を使用します。	
引数	第一引数 : addr	: リードアドレス (シリアルフラッシュメモリのアドレス)
	第二引数 : buf	: リードバッファの先頭アドレス
	第三引数 : size	: データバイト数
リターン値	なし	
備考		

fmtool_debug

概要	FMTOOL 用デバッグ関数
ヘッダ	なし
宣言	void fmtool_debug(void);
説明	FMTOOL をデバッグするためにイレーズモジュールとライトモジュールを実行します。本関数は統合開発環境を使用して起動します。レジスタウィンドウで PC に "fmtool_debug"、SP に "H'FFF90000" を設定してください。 本関数は、最終ページを除いた先頭 2 セクタ (H'1800 0000 番地～H'1801 FFFF 番地) に連番を書き込みます。ただし最終ページはダミーデータ領域ですので実際にはライトされません。シリアルフラッシュメモリへの書き込み後に外部アドレス空間リードモードを設定しています。そのため、メモリウィンドウを開いて書き込み値を確認することができます。なお、外部アドレス空間リードモードを設定するまでは SPI マルチ I/O バス空間にはアクセスできません。したがって、メモリウィンドウも外部アドレス空間リードモードを設定した後に開いてください。
引数	なし
リターン値	なし
備考	イレーズモジュールおよびライトモジュールがスタックポインタの値を書き換えるため本関数は両モジュールのエントリ部分ではなくメイン処理を実行しています。

6.8 フローチャート

サンプルコードで使用する主な関数の処理の流れを示します。シリアルフラッシュメモリ操作関数についてはアプリケーションノート「SH7268/SH7269 グループ SPI マルチ I/O バスコントローラ シリアルフラッシュメモリ接続例(R01AN0671JJ)」を参照してください。

6.8.1 イレーズモジュール

図 6.4にイレーズモジュールのフローチャートを示します。

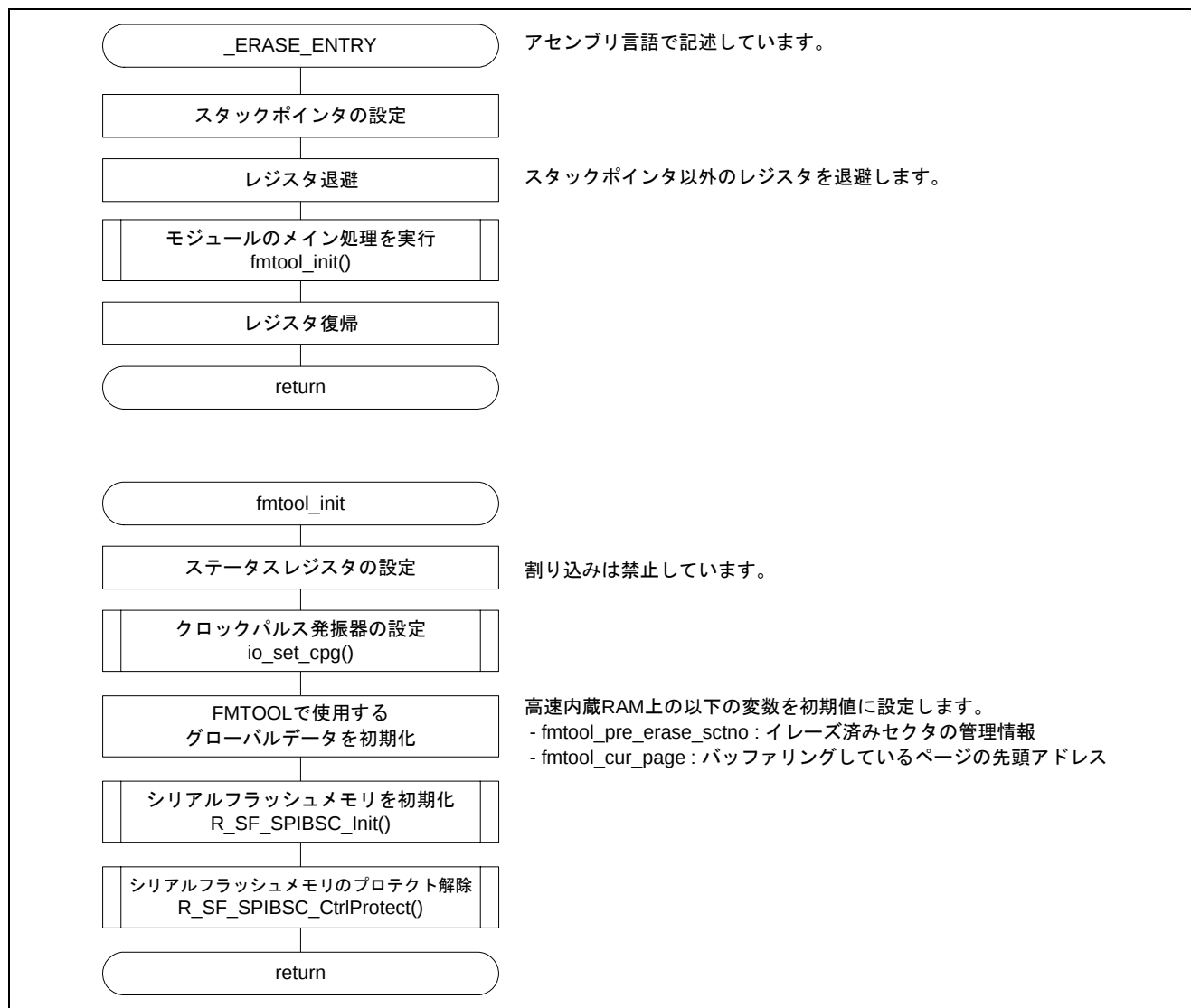


図 6.4 イレーズモジュール

6.8.2 ライトモジュール

図 6.5にライトモジュールのフローチャートを示します。

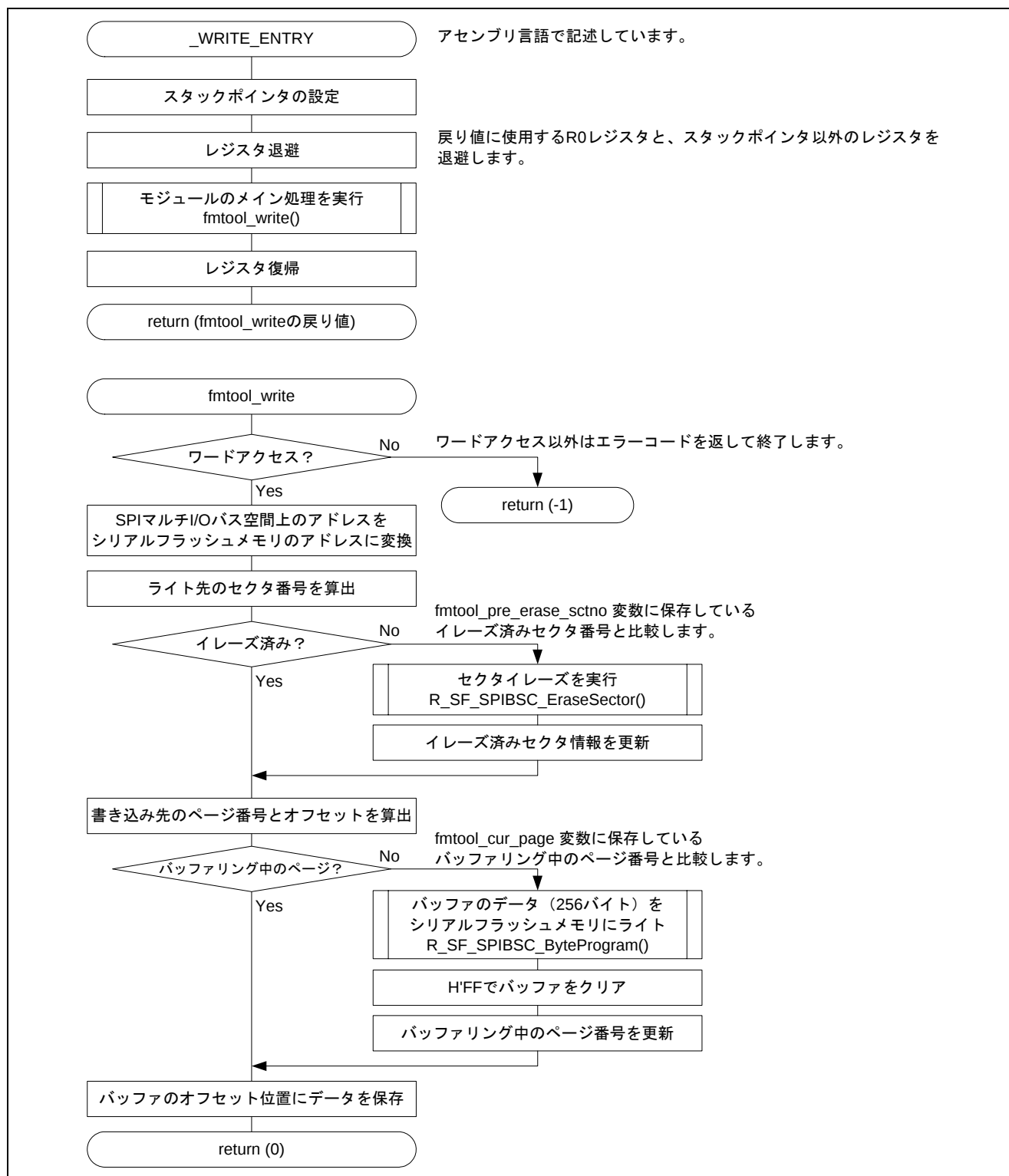


図 6.5 ライトモジュール

6.9 注意事項

6.9.1 ロードモジュールへのダミーデータ追加

シリアルフラッシュメモリへの書き込み速度を向上させるため、FMTOOL は書き込みデータをバッファリングしてページ単位に書き込みます。シリアルフラッシュメモリへの書き込みは、バッファリング中のページと異なるページのアドレスが指定されたタイミングで行われるため、最後の 1 ページ分のデータは、バッファに残ったままシリアルフラッシュメモリに書き込まれない可能性があります。有効データをバッファに残さないために、ロードモジュールの最後の 1 ページ分のデータにはダミーデータを配置してください。

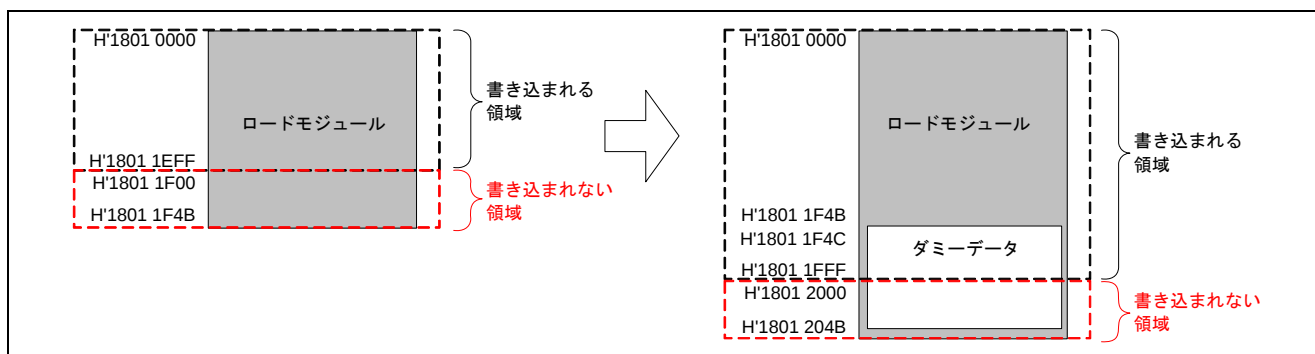


図 6.6 バッファに残って書き込まれない領域

図 6.7にダミーデータの追加例を示します。用意したダミーセクション (CDUMMY_MODULE_END) で 256 バイトの定数データを定義し、ROM領域の最後に配置します。

dummy.c

```
#define SF_PAGE_SIZE 256

#pragma section DUMMY_MODULE_END
const char dummy_area[SF_PAGE_SIZE] = { 0 };
#pragma section
```

ROM領域の
最後に配置する

The 'セクション設定' (Section Setting) dialog box shows a list of sections. The 'CDUMMY_MODULE_END' section is highlighted with a red box. The list includes sections like DAPPINFO, DVECTTBL, DINTTBL, PResetPRG, PIntPRG, P, C, C\$BSEC, C\$DSEC, D, PCACHE, CDUMMY_MODULE_END, RINTTBL, B, R, RPCACHE, and S. The address range 0x18010000 to 0xFFFF8000 is shown on the left.

図 6.7 ダミーデータの追加例

6.9.2 ロードモジュール間のセクタ共有禁止

図 6.8にロードモジュール間でセクタを共有した場合の動作を示します。FMTOOLでダウンロードするユーザプログラムを複数のロードモジュールで構成することも可能ですが、ロードモジュール間でセクタを共有することはできません。セクタを共有した場合、先にダウンロードしたロードモジュールのデータが誤って消去されます。

なお、ここでのロードモジュールの領域には、6.9.1で説明したダミーデータの領域も含まれます。

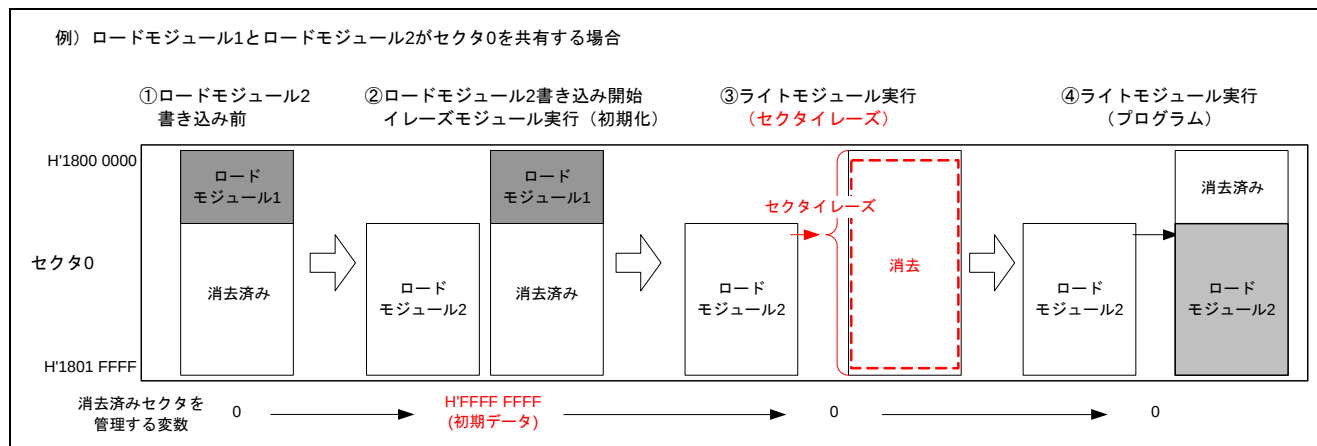


図 6.8 ロードモジュール間でセクタを共有した場合の動作

7. 応用例

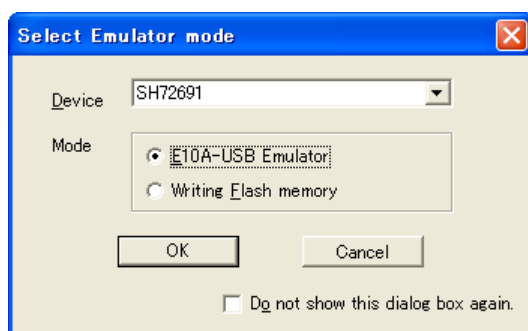
7.1 ユーザプログラムダウンロード手順

作成した FMTOOL (sh7269_spibsc_fmtool.mot) を使用してユーザプログラムをシリアルフラッシュメモリにダウンロードする手順を以下に示します。

7.1.1 ダウンロード環境の準備

- (1) PC に接続した E10A-USB エミュレータと、ユーザシステムを接続します。
- (2) High-performance Embedded Workshop を起動し、ユーザプログラムのワークスペースを開きます。
- (3) 図 7.1に示すCPU selectダイアログボックスが表示されます。

ご使用の CPU をドロップダウンリストボックスより選択し、OK ボタンを押してください。



【注】 SH72691 の場合

図 7.1 CPU select ダイアログボックス

- (4) Connectingダイアログボックスが表示され、エミュレータの接続を開始します。

図 7.2に示すRESET信号入力要求メッセージのダイアログボックスが表示されます。

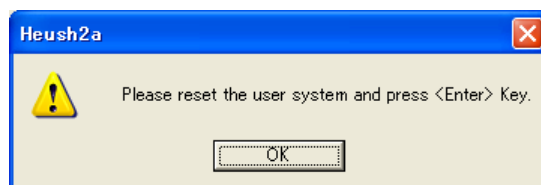


図 7.2 RESET 信号入力要求メッセージのダイアログボックス

- (5) ユーザシステムの電源を入れます。

ユーザシステムからRESET信号を入力し、OKボタンをクリックします。

High-performance Embedded Workshop のOutputウィンドウに "Connected" と表示されたら、E10A-USBエミュレータの起動は完了です。

7.1.2 バッチファイルの登録

- (1) メニューの[デバッグ]→[デバッグの設定]を選択します。
- (2) 図 7.3に示すデバッグの設定ウィンドウが開きます。
- (3) "コマンドバッチファイル実行タイミング" のプルダウンメニューから、"Before download modules"を選択します。
- (4) "コマンドバッチファイル実行順序" の追加ボタンをクリックし、バッチファイルを追加します。
- (5) OK ボタンをクリックして登録は完了です。

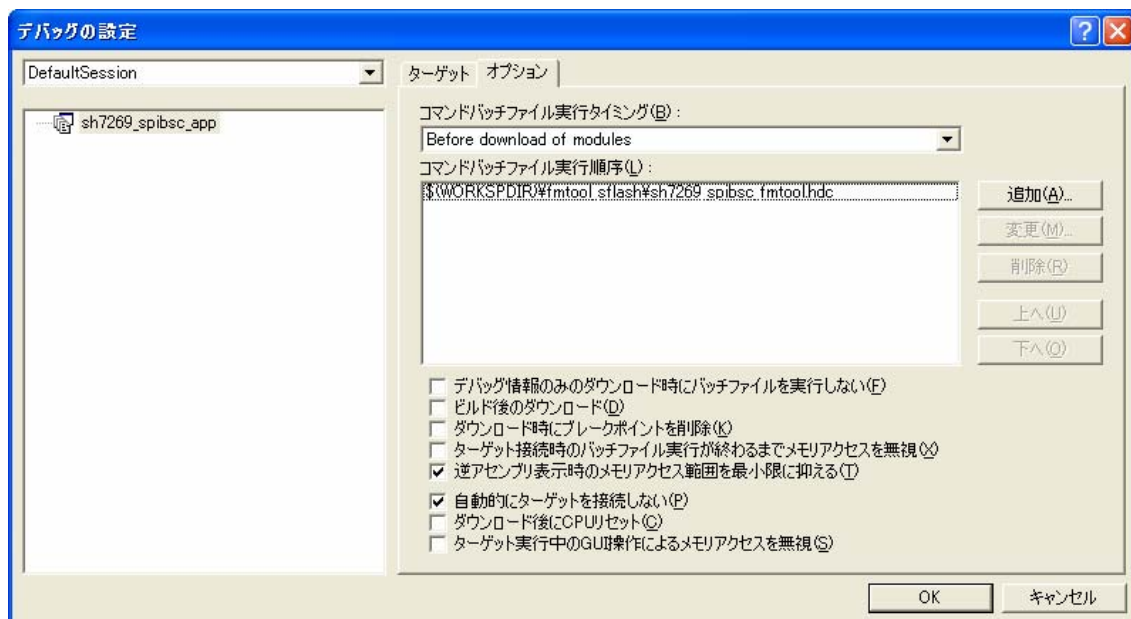


図 7.3 デバッグの設定ウィンドウ

7.1.3 Configuration ダイアログボックスの設定

図 7.4に示すConfigurationダイアログボックス（Loading flash memoryページ）で、E10A-USBエミュレータを使用して外部フラッシュメモリにユーザプログラムをダウンロードするための設定を行います。

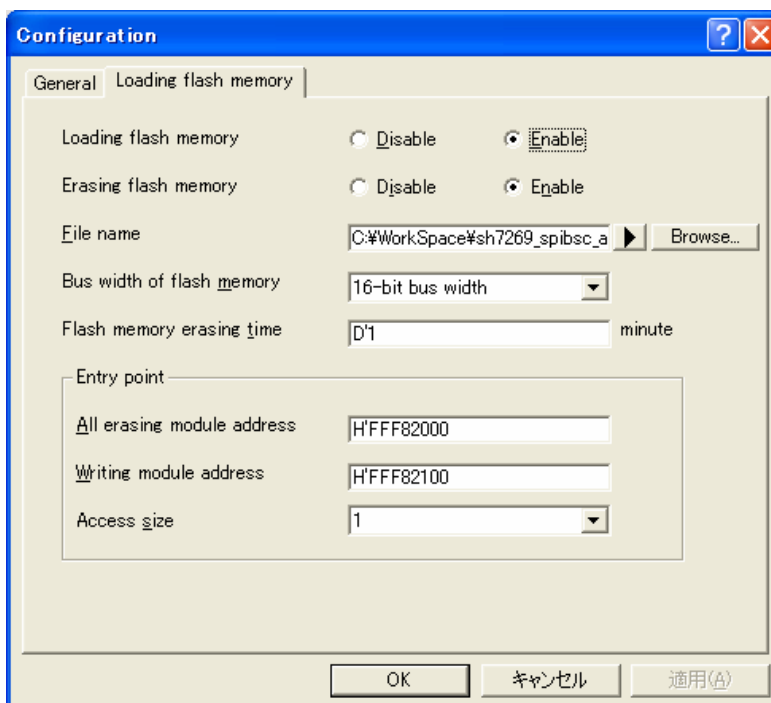


図 7.4 Configuration ダイアログボックス（Loading flash memory ページ）

各項目の設定を表 7.1に示します。設定が完了したらOKボタンをクリックして完了です。

表 7.1 Configuration ダイアログボックスの設定値

項目	設定値
Loading flash memory	Enable
Erasing flash memory	Enable
File Name	sh7269_spibsc_fmtool.mot
Bus width of flash memory	16-bit bus width
All erasing module address	イレーズモジュールの先頭アドレスを指定
Writing module address	ライトモジュールの先頭アドレスを指定

7.1.4 ダウンロードモジュールの追加

デバッグメニューからデバッグの設定ウィンドウを開き、追加ボタンをクリックしてください。図 7.5に示すダウンロードモジュールウィンドウが表示されますのでユーザプログラム（シリアルフラッシュメモリにロードしたいプログラム）をダウンロードモジュールに追加します。

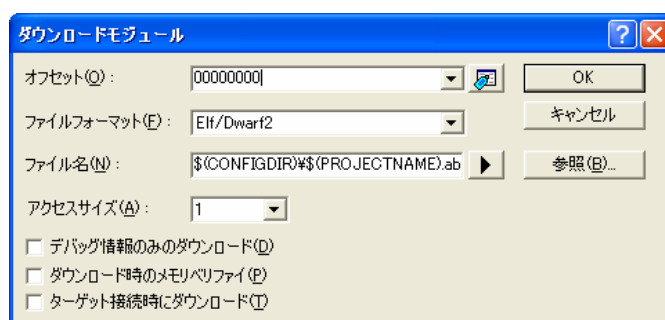


図 7.5 ダウンロードモジュールウィンドウ

7.1.5 ユーザプログラムのダウンロード

図 7.6に示すダウンロード機能にて、ユーザプログラムをダウンロードします。

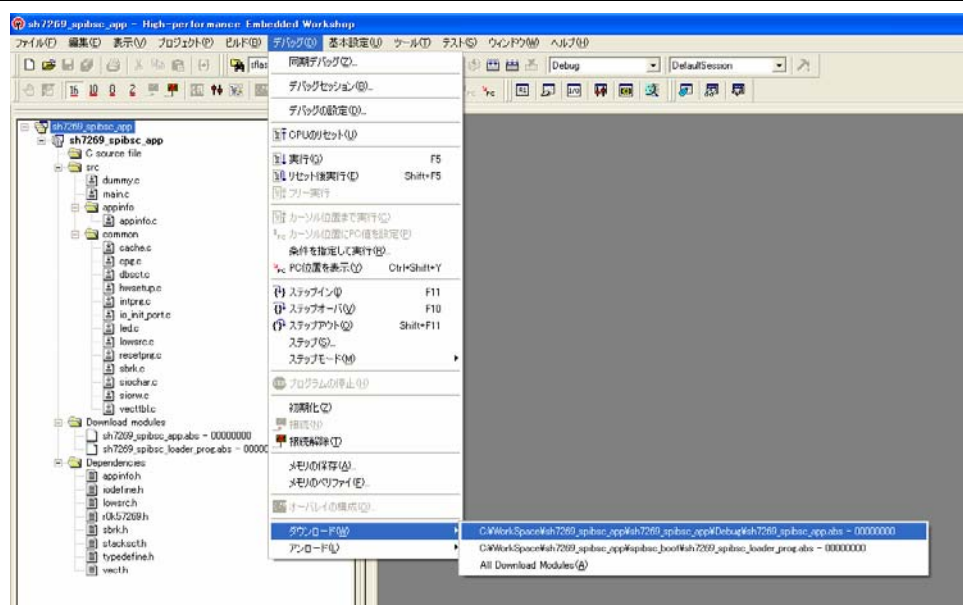


図 7.6 ユーザプログラムのダウンロード

7.2 シリアルフラッシュブートへの応用

シリアルフラッシュメモリからブートする機能を、ここではシリアルフラッシュブートと称します。シリアルフラッシュブートの詳細は、アプリケーションノート「SH7268/SH7269 グループ SPI マルチ I/O バスコントローラを使用したシリアルフラッシュメモリからのブート例(R01AN0663JJ)」に記載していますのでこちらを参照してください。

ここでは、上記アプリケーションノート(R01AN0663JJ)のフラッシュ書き込みツールであるダウンローダをFMTOOL に置き換える際の変更点について説明します。

7.2.1 セクション配置

図 7.7にFMTOOLを使用した場合のセクション配置を示します。ローダプログラムとアプリケーションプログラムは、以下の点に注意して配置してください。

- SPI マルチ I/O バス空間に、セクションを配置する
- 異なるロードモジュールで 1 つのセクタを共有しない（アプリケーションプログラムは H'1800 2000 番地ではなく H'1801 0000 番地に配置するなど）
- ローダプログラムは、最適化リンケージエディタのオプション（ROM から RAM へマップするセクション）を使って H'FFF8 0000 番地にマッピングする

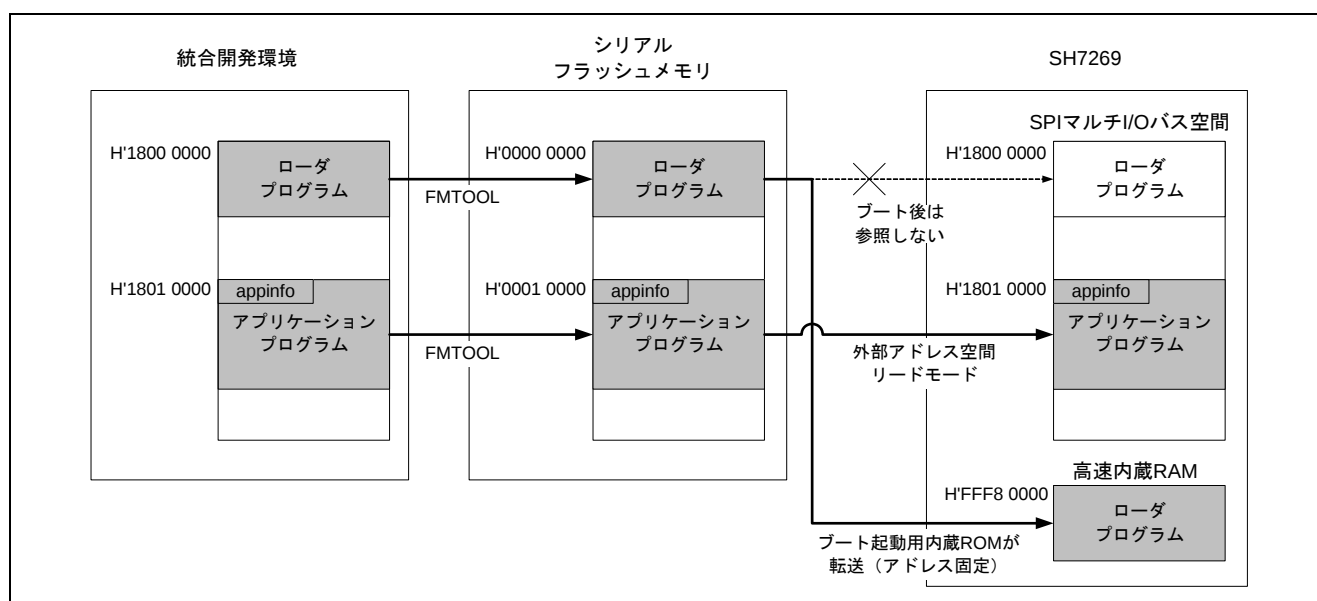


図 7.7 FMTOOL を使用した場合のセクション配置

7.2.2 ダミーデータの追加

「6.9.1 ロードモジュールへのダミーデータ追加」に示したとおり、ローダプログラムおよびアプリケーションプログラムにダミーデータを必ず追加してください。

7.2.3 ロードモジュールのダウンロード方法

ロードモジュールをダウンロードする際の統合開発環境の操作方法も変わります。「7.1 ユーザプログラムダウンロード手順」に示した手順でダウンロードしてください。

7.3 FMTOOL のカスタマイズ

サンプルコードはシリアルフラッシュメモリのデバイス仕様に依存するため、デバイス変更時はプログラムのカスタマイズが必要な場合があります。

7.3.1 サンプルコードに対応するデバイス仕様

表 7.2と表 7.3に、使用デバイスの詳細仕様およびサンプルコードで使用するコマンドを示します。

表 7.2 使用デバイスの詳細仕様

項目	内容
メーカー	Spansion Inc.
型名	S25FL032P0XMF101
容量	4M バイト
インタフェース	SPI マルチ I/O バス (Single/Dual/Quad mode)
アクセスタイム	104MHz (Single mode) 、80MHz (Dual/Quad mode)
セクタ構造	ユニフォーム型
セクタサイズ	64K バイト
ページサイズ	256 バイト

表 7.3 サンプルコードで使用するコマンド

項目	内容
イレーズコマンド	H'D8 (64KB セクタイレーズ)
プログラムコマンド	H'32 (Quad ページプログラミング) 、 H'02 (Single ページプログラミング)

7.3.2 カスタマイズの内容

以下にカスタマイズが必要なケースと、その変更内容を示します。

表 7.4 カスタマイズが必要なケースとカスタマイズ内容

項目	カスタマイズ内容
Quad モードが使用できない場合 (Single モードで動作させる場合)	SPI_QUAD マクロの設定値を 0 に変更してください。
セクタサイズが異なる場合 (64K バイトのセクタイレーズに対応していない場合)	セクタ構造がユニフォーム型の場合は、SF_SECTOR_SIZE マクロの設定値を新しいセクタサイズに変更し、 R_SF_SPIBSC_EraseChip 関数で使用するセクタイレーズコマンドを新しいセクタサイズに対応したコマンドに変更してください。セクタ構造がトップ型またはボトム型の場合は、 fmtreeol_write 関数で行っているセクタ番号の判別アルゴリズムも変更する必要があります。
デバイスの初期設定手順が異なる場合 表 7.3のコマンドが使用できない場合 電氣的特性が異なる場合	シリアルフラッシュメモリの操作関数および SPI マルチ I/O バスコントローラの制御関数のカスタマイズが必要です。詳細はサンプルコードを参照してください。

【注】 FMTOOLはフラッシュメモリの仕様に依存するため、表 7.4が全てのチェック項目を網羅しているわけではありません。使用されるデバイスのデータシートをご確認の上、デバイスの仕様に合わせて FMTOOLを修正してください。

7.4 FMTOOL のデバッグ手順

FMTOOL は統合開発環境から起動するため、通常のデバッグ手順は使用できません。ここでは FMTOOL のデバッグ手順を説明します。

7.4.1 デバッグコードの実装

ENABLE_DEBUG_MODE マクロを 1 に設定した状態でビルドを行ってください。FMTOOL のロードモジュールにデバッグ関数 (fmttool_debug) が組み込まれます。デバッグ関数の仕様については、6.7 関数仕様を参照してください。

7.4.2 デバッグ環境の準備

「7.1.1 ダウンロード環境の準備」と同じ手順で、E10A-USB エミュレータとユーザシステムを接続します。

7.4.3 FMTOOL のダウンロード

(1) CPU をリセットします。

メニューの[デバッグ]→[CPU のリセット]を選択してください。

(2) FMTOOL を高速内蔵 RAM 上にダウンロードします。

「7.1.5 ユーザプログラムのダウンロード」と同様の手順で、sh7269_spibsc_fmttool.abs を高速内蔵 RAM 上にダウンロードします。デバッグ情報もダウンロードするために、S フォーマットファイル (.mot) ではなく、ELF-DWARF ファイル (.abs) をダウンロードしてください。

7.4.4 FMTOOL のデバッグ

(1) プログラムカウンタをデバッグプログラムの先頭に設定します。

レジスタウィンドウの PC をダブルクリックして、"fmttool_debug" を入力してください。

(2) スタックポインタを設定します。

レジスタウィンドウの SP をダブルクリックして、"H'FFF90000" を入力してください。



図 7.8 PC と SP の初期化

(3) デバッグコードを実行します。

メニューから[デバッグ]→[実行]を選択するとプログラムが実行されます。必要に応じてステップ実行やブレークポイントを利用してデバッグを行ってください。

(4) 動作結果を確認します。

デバッグコードは、最終ページを除いた先頭 2 セクタ (H'1800 0000 番地～H'1801 FFFF 番地) に連番を書き込みます。ただし最終ページはダミーデータ領域ですので実際にはライトされません。シリアルフラッシュメモリへの書き込み後に外部アドレス空間リードモードを設定しているため、メモリウィンドウを開いて書き込み値を確認することができます。ただし、外部アドレス空間リードモードを設定するまでは SPI マルチ I/O バス空間にはアクセスできません。したがって、メモリウィンドウも外部アドレス空間リードモードを設定した後に開いてください。

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

SH7268/SH7269 グループ ユーザーズマニュアル ハードウェア編 Rev.1.00

(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。)

C コンパイラマニュアル

SuperH RISC engine ファミリ C/C++コンパイラパッケージ V.9.04

C コンパイラユーザーズマニュアル Rev.1.01

(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。)

SuperH ファミリ用 E10A-USB エミュレータ ユーザーズマニュアル Rev.9.00

(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。)

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/inquiry>

改訂記録	SH7268/SH7269 グループ E10A-USB フラッシュメモリダウンロード機能の応用例（シリアルフラッシュメモリへのダウンロード）
------	--

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2011.11.30	—	初版発行

すべての商標および登録商標は、それぞれの所有者に帰属します。

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理

【注意】未使用端子は、本文の「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。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

2. 電源投入時の処置

【注意】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. リザーブアドレスのアクセス禁止

【注意】リザーブアドレスのアクセスを禁止します。

アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。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。

4. クロックについて

【注意】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。

5. 製品間の相違について

【注意】型名の異なる製品に変更する場合は、事前に問題ないことをご確認下さい。

同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。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。

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 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お客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続きを行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、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に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したのですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。
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
高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）
特定水準： 航空機器、航空宇宙機器、海中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障および事故につきましては、当社は、一切その責任を負いません。
9. 当社は、当社製品の品質および信頼性の向上に努めておりますが、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、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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。お客様がかかる法令を遵守しないことにより生じた損害に関し、当社は、一切その責任を負いません。
11. 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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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その総株主の議決権の過半数を直接または間接に保有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。



ルネサスエレクトロニクス株式会社

■営業お問合せ窓口

<http://www.renesas.com>

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社 〒100-0004 千代田区大手町2-6-2（日本ビル）

(03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口： <http://japan.renesas.com/inquiry>