お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

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ものですが、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、海底中継機器、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て、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。

SH7262/SH7264 グループ

ルネサスシリアルペリフェラルインタフェース

シリアルフラッシュメモリ接続例

要旨

本アプリケーションノートは、SH7262/SH7264 のルネサスシリアルペリフェラルインタフェース（RSPI）を使用したシリアルフラッシュメモリの接続例について説明します。

動作確認デバイス

SH7262/SH7264

以下、総称して「SH7264」として説明します。

目次

1. はじめに

1.1 仕様

- 2M バイト（64K バイト×32 セクタ、256 バイト/ページ）のシリアルフラッシュメモリを SH7264 と接続します。
- シリアルフラッシュメモリへのアクセスには、RSPI のチャネル 0 を使用します。

1.2 使用機能

- ルネサスシリアルペリフェラルインタフェース（RSPI）
- 汎用入出力ポート

1.3 適用条件

マイコン	SH7262/SH7264
動作周波数	内部クロック：144 MHz バスクロック：72 MHz 周辺クロック：36 MHz
統合開発環境	ルネサステクノロジ製 High-performance Embedded Workshop Ver.4.04.01
C コンパイラ	ルネサステクノロジ製 SuperH RISC engine ファミリ C/C++コンパイラパッケージ Ver.9.02 Release00
コンパイルオプション	High-performance Embedded Workshop でのデフォルト設定 (-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)

1.4 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。合わせて参照してください。

- SH7262/SH7264 グループ　初期設定例
- SH7262/SH7264 グループ　シリアルフラッシュメモリからのブート例

2. 応用例の説明

本応用例では、SH7264（マスタ）と SPI 互換のシリアルフラッシュメモリ（スレーブ）を接続して、ルネサスシリアルペリフェラルインタフェース（RSPI）を使用したリード／ライトアクセスを行います。この章では、端子接続例と参考プログラムフローを説明します。

2.1 RSPI の動作概要

SH7264 の RSPI は、MOSI（Master Out Slave In）端子および MISO（Master In Slave Out）端子、SSL（Slave Select）端子、RSPCK（SPI Clock）端子を使用して、SPI 動作で全二重同期式のシリアル通信が可能です。RSPI は、マスタ／スレーブの選択、シリアル転送クロックの極性と位相の変更（SPI モード変更）、転送ビット長の変更（8、16、32 ビット）が可能なため、多様な SPI 互換デバイスを接続することができます。

RSPI ではチャネル 0 とチャネル 1 の両方を使用できますが、本応用例ではチャネル 0 を使用しています。

2.2 シリアルフラッシュメモリの端子接続例

表 1に本応用例で使用する SPI 互換シリアルフラッシュメモリ（ATMEL 社製 AT26DF161A）の仕様を示します。

表1　本応用例で使用するシリアルフラッシュメモリの仕様

項目	仕様
SPI モード	SPI モード 0 およびモード 3 に対応可能
クロック周波数	70MHz(max)
容量	2M バイト
セクタサイズ	64K バイト
ページサイズ	256 バイト
イレースサイズ	Chip Erase／64Kbyte／32Kbyte／4Kbyte
プログラムサイズ	Byte／Page Program（1～256 バイト）、Sequential Program（連続書込み）
プロテクト単位	セクタ単位

図 1にシリアルフラッシュメモリ接続回路例を示します。SH7264 の端子機能については、表 2のマルチプレクス出力端子に従い設定してください。

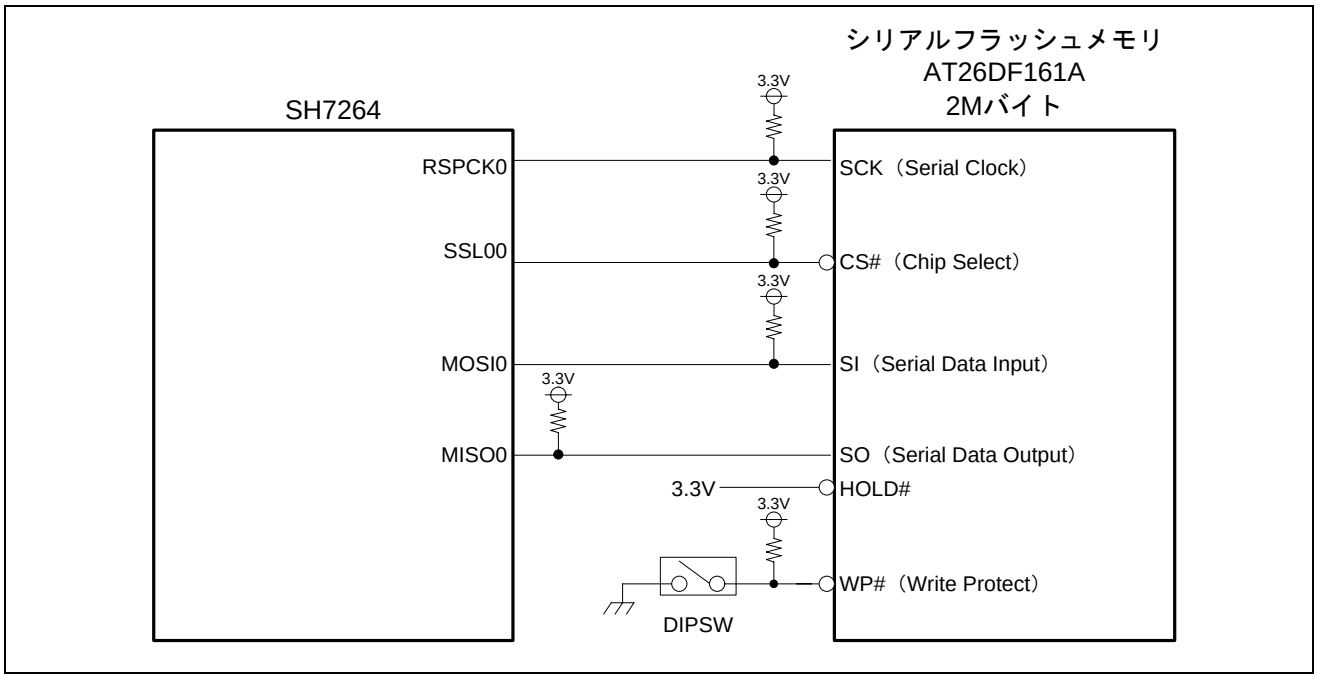


図1　シリアルフラッシュメモリ接続回路例

【注】　制御信号端子の外付け抵抗によるプルアップ/プルダウン処理について
制御信号に対するプルアップ/プルダウン処理は、マイコンの端子状態がハイインピーダンスの場合でも、外部デバイスが誤動作しないように信号線のレベルを決定します。SSL00 端子については外付け抵抗でプルアップ処理を行い、H レベルにしています。RSPCK0 端子と MOSI0 端子はプルアップまたはプルダウン処理をおこなってください。また MISO0 端子は入力のためプルアップまたはプルダウン処理により中間電位になることを防ぎます。

表2　マルチプレクス出力

周辺機能	使用端子名	SH7264 ポートコントロールレジスタ		SH7264 マルチプレクス端子名
		レジスタ名	MD ビット設定値	
RSPI	MISO0	PFCR3	PF12MD[2:0] = B'011	PF12/BS#/MISO0/TIOC3D/SPDIF_OUT
	MOSI0	PFCR2	PF11MD[2:0] = B'011	PF11/A25/SSIDATA3/MOSI0/TIOC3C/SPDIF_IN
	SSL00	PFCR2	PF10MD[2:0] = B'011	PF10/A24/SSIWS3/SSL00/TIOC3B/FCE#
	RSPCK0	PFCR2	PF9MD[2:0] = B'011	PF9/A23/SSISCK3/RSPCK0/TIOC3A/FRB

【注】　SH7264 のマルチプレクス端子について
MISO0、MOSI0、SSL00、RSPCK0 端子はマルチプレクス端子であり、初期状態は汎用入出力ポートになっています。そのためシリアルフラッシュメモリへアクセスする前に、汎用入出力ポートのコントロールレジスタによって RSPI 端子機能に設定する必要があります。

2.3 インタフェースタイミング例

SH7264 とシリアルフラッシュメモリ間のインタフェースタイミング例を示します。スレーブとなるシリアルフラッシュメモリのタイミング条件に合わせて RSPI 設定およびクロック周波数設定を行います。

図 2にデータ転送タイミング例を示します。本応用例で使用するシリアルフラッシュメモリは、クロックの立ち上がりでデータサンプルを行い、立ち下がりでデータ変化する仕様のため、コマンドレジスタ（SPCMD）の CPOL ビットと CPHA ビットにはともに 1 を設定します。本設定により、アイドル時の RSPCK は 1 に設定され、RSPI のデータ変化タイミングを奇数エッジ（ここでは立ち下がりエッジ）に設定することができます。表 3と表 4に示すタイミング条件を満たすように RSPI を設定してください。

本応用例ではビットレートを 18Mbps、データレジスタ（SPDR）のアクセス幅を 8 ビットに設定しています。これらの設定を最適化することで高速アクセスが可能になります。ただし、図 2の転送方式ではセットアップ時間が不足する可能性がありますので、その場合はデータ変化からデータサンプルまでの期間を 1 サイクル分に拡張する転送方式を利用してください。

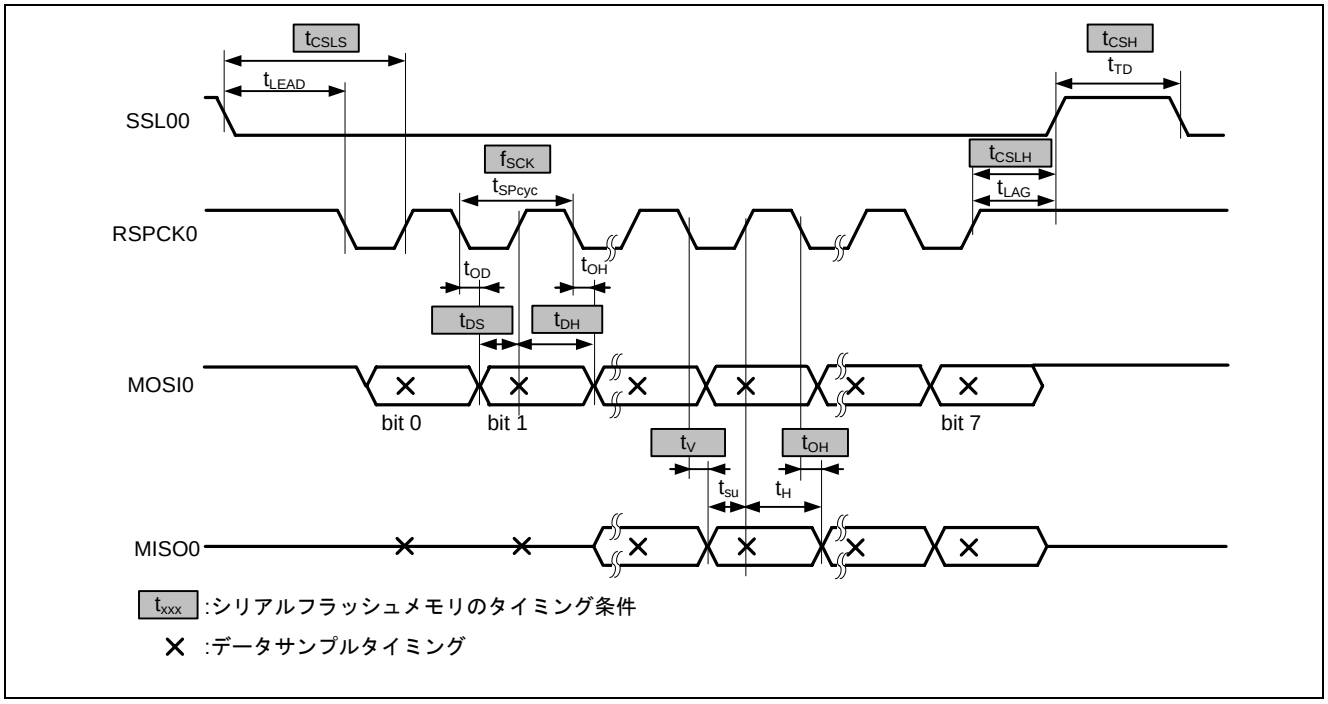


図2　データ転送タイミング例（CPOL=1, CPHA=1 の場合）

表3　データ転送におけるシリアルフラッシュメモリのタイミング条件

シンボル	項目	説明	関連レジスタ
t_{CSLS}	チップセレクト'L'セットアップ時間	SSL のアサートから RSPCK の立ち上がりでスレーブがデータサンプルするまでに必要な時間です。 以下の式を満たす設定を行います。 t_{LEAD}(=RSPCK 遅延) + 1/2 × t_{SPcyc} > t_{CSLS} (min)	SPCKD レジスタ SPCMD レジスタ
t_{CSH}	チップセレクト'H'時間	SSL のネゲート期間として必要な時間です。 以下の式を満たす設定を行います。 t_{TD}(=2 × Bφ+次アクセス遅延) > t_{CSH} (min)	SPND レジスタ SPCMD レジスタ
f_{SCK}	シリアルクロック周波数	スレーブが対応可能な最大動作周波数です。 以下の式を満たす設定を行います。 f_{SCK}(max) > 1/ t_{SPcyc}	SPBR レジスタ SPCMD レジスタ
t_{CSLH}	チップセレクト'L'ホールド時間	最後の RSPCK の立ち上がりから SSL のネゲートまでに必要なホールド時間です。 以下の式を満たす設定を行います。 t_{LAG}(=SSL ネゲート遅延) > t_{CSLH} (min)	SSLND レジスタ SPCMD レジスタ
t_{DS}	データ入力セットアップ時間	マスタのデータ出力からデータサンプルまでに必要な時間です。 以下の式を満たす設定を行います。 1/2 × t_{SPcyc} - t_{OD}(max) > t_{DS} (min)	
t_{DH}	データ入力ホールド時間	データサンプルからマスタのデータ出力の停止までに必要な時間です。 以下の式を満たす設定を行います。 t_{OH}(min) + 1/2 × t_{SPcyc} ＞ t_{DH} (min)	

表4　データ転送における SH7264 のタイミング条件

シンボル	項目	説明	関連レジスタ
t_{SU}	データ入力セットアップ時間	スレーブのデータ出力からデータサンプルまでに必要な時間です。 以下の式を満たす設定を行います。 1/2 × t_{SPcyc} – t_V(max) ＞ t_{SU}(min)	
t_H	データ入力ホールド時間	データサンプルからスレーブのデータ出力の停止までに必要な時間です。 以下の式を満たす設定を行います。 t_{OH}(min) + 1/2 × t_{SPcyc}＞ t_H(min)	

2.4 参考プログラムの動作

2.4.1 RSPI の初期設定例

図 3および図 4に本参考プログラムにおける RSPI 初期設定フローを示します。本設定によりマスタモードでの SPI 動作が可能となります。

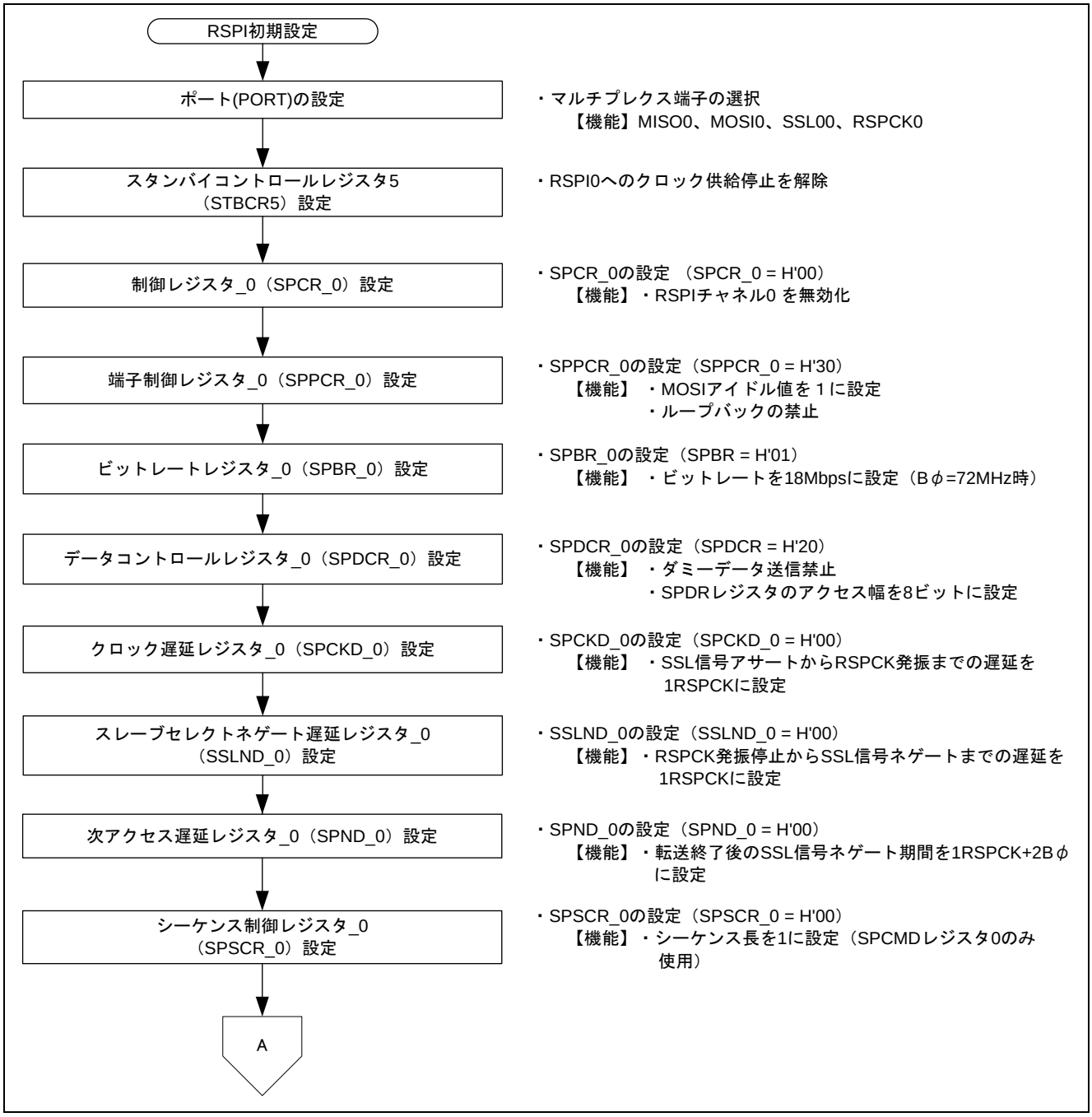


図3 参考プログラムの RSPI 初期設定フロー（1）

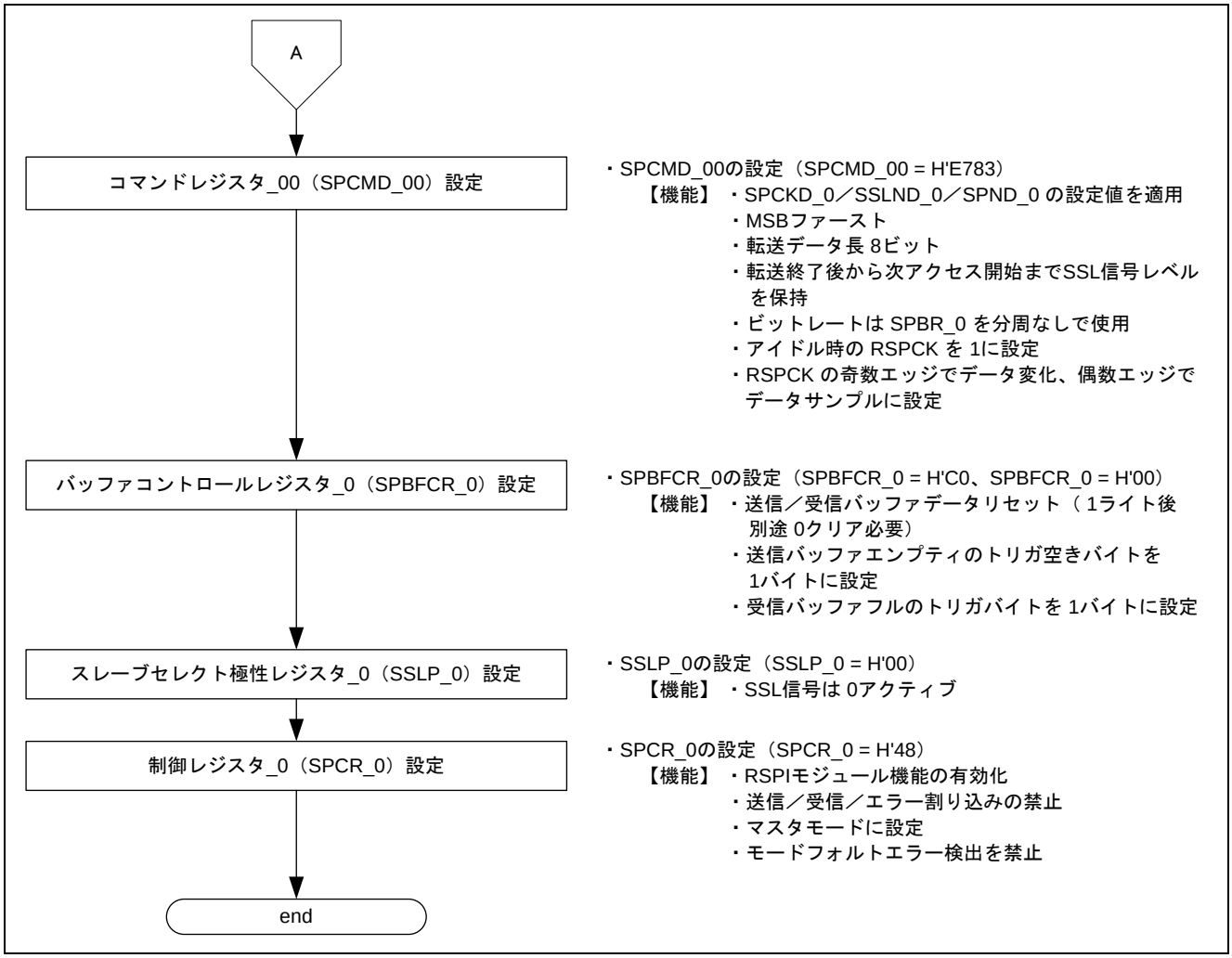


図4　参考プログラムの RSPI 初期設定フロー（2）

2.4.2 コマンド転送例

シリアルフラッシュメモリのアクセスは全てコマンド形式で行います。以下に、主なコマンドおよびコマンドシーケンス例、参考プログラムでの処理フローを示します。

なお、本応用例では ATMEL 社製の AT26DF161A のコマンドを参考にしています。コマンドの詳細は使用するデバイスのデータシートを参照してください。

A. 主なコマンド

表 5に AT26DF161A の主なコマンドを示します。

表5　AT26DF161A の主なコマンド

コマンド名	オペコード	アドレスバイト数	ダミーバイト数	データバイト数	機能
Read Array	H’0B	3	1	1 以上※1	データのリード
Read Array（低周波数向け）	H’03	3	0	1 以上※1	データのリード（クロック周波数が低い場合の高速リード）
Write Enable	H’06	0	0	0	プログラム／イレースコマンドの許可
Write Disable	H’04	0	0	0	プログラム／イレースコマンドの禁止
Block Erase（64Kbytes）	H’D8	3	0	0	セクタ（64KB）単位のイレース
Chip Erase	H’C7	0	0	0	全領域のイレース
Byte/Page Program	H’02	3	0	1 以上※2	データのライト
Read Status Register	H’05	0	0	1 以上	ステータスリード
Write Status Register	H’01	0	0	1	ステータスライト

【注】 ※1 指定アドレスからインクリメントされた領域をリードします。（最終番地を超えた場合は 0 番地に戻ります。）

※2 指定アドレスと同一ページ内で、インクリメントされた領域にライトします。（ページの最終番地を超えた場合はページの先頭に戻ります。）

B. コマンドシーケンス例

図 5に Read Array（低周波数向け）コマンドのシーケンス例を示します。

Read Array（低周波数向け）コマンドは、SSL 信号のアサート後、オペコード（H'03）に続けてアドレス（3 バイト）をマスタから転送します。その後、RSPCK の立ち下がり毎にスレーブから Read データが転送されます。

設定したアクセス幅の転送を繰り返すことでコマンドシーケンスを実現できますが、SSL 信号のレベルに注意が必要です。コマンドの先頭で SSL 信号をアサートしてから、コマンドの最終バイトの転送完了まで SSL 信号をネゲートしてはいけません。参考プログラムでは SPCMD レジスタの SSLKP ビットを 1 にセットして SSL 信号を保持しています。SSL のネゲートは全データの転送完了後に SPCR レジスタの SPE ビットを 0 クリアすることで行っています。

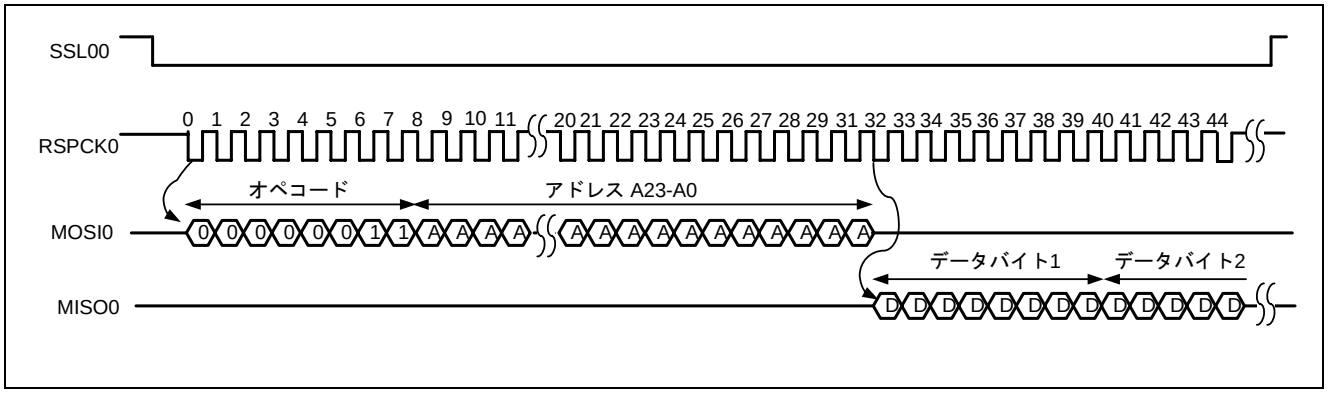
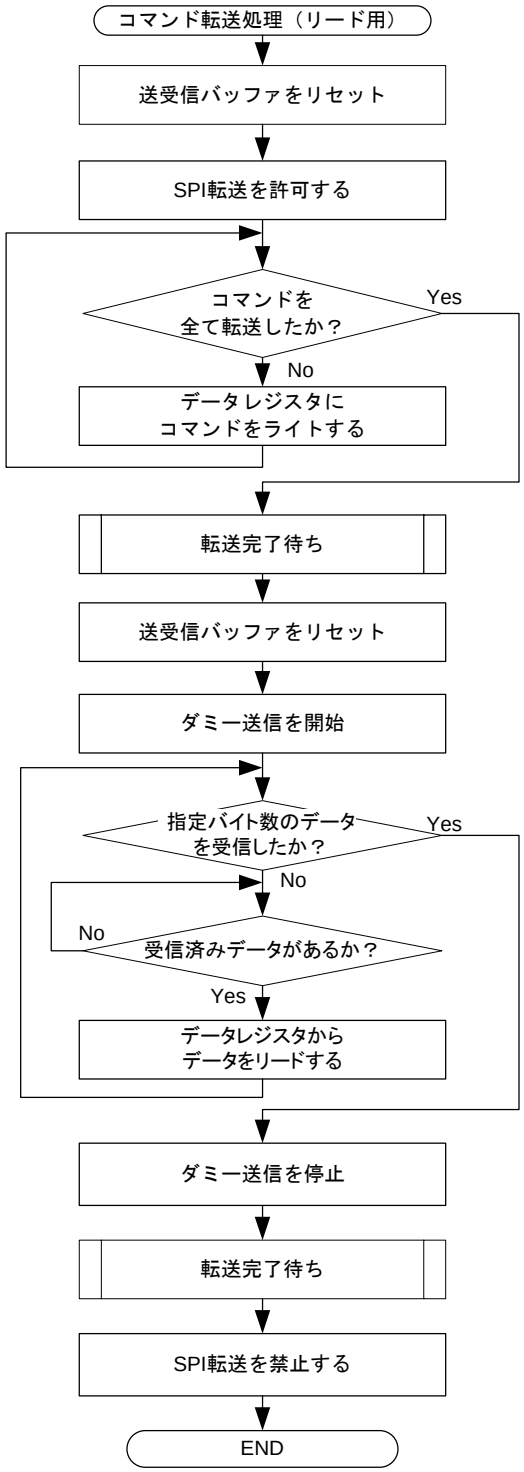


図5　リードコマンドシーケンス（H'03 オペコード）

C. 参考プログラムでのコマンド転送例

コマンドにはマスタ出力とスレーブ出力の両方を使用するリード用のコマンドと、マスタ出力のみを使用するライト用のコマンドがあります。図 6にリード用のコマンド転送処理フローを示します。Read Array（低周波数向け）コマンドはこの処理フローに従います。また図 7にライト用のコマンド転送処理フローを示します。



・SPBFCRレジスタの設定（SPBFCR = 0xC0、SPBFCR = 0x00 ）
【機能】・送信／受信バッファのデータリセット（1ライト後、別途0クリアが必要）

・SPCRレジスタの設定（SPEビット = 1 ）
【機能】・RSPIの機能を有効化
転送機能を有効にします。

・コマンドシーケンスのオペコードとアドレスなどのマスタ出力データを転送します。Read Array（低周波数向け）コマンドの場合は、オペコード（0x03）およびアドレス3バイトです。

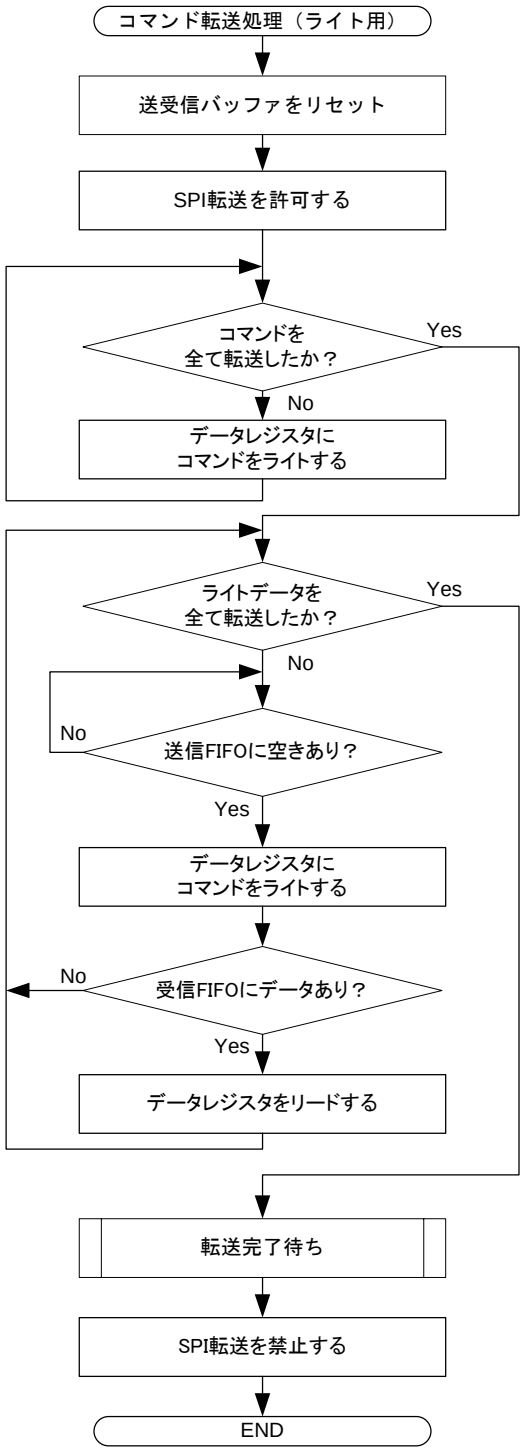
・送信FIFOのオーバフローを防ぐため、コマンドは 8バイト以下にする必要があります。

・SPDCRレジスタの設定（TXDMYビット = 1）
【機能】・ダミー送信を許可
（データレジスタにデータをライトしなくても自動的にダミーデータが送信されます。）

・SPEビットをクリアする前に転送が完了していることを確認します。

・SPCRレジスタの設定（SPEビット = 0 ）
【機能】・RSPIの機能を無効化
本設定によりSSL信号がネゲートされます。なおRSPIの制御ビットは初期化されないため、SPEビットに 1 を再設定すれば同じ転送モードでRSPIを起動できます。

図6　参考プログラムのリード用コマンド転送フロー



・SPBFCRレジスタの設定（SPBFCR = 0xC0、SPBFCR = 0x00 ）
【機能】・送信／受信バッファのデータリセット（1ライト後、別途0クリアが必要）

・SPCRレジスタの設定（SPEビット = 1 ）
【機能】・RSPIの機能を有効化
転送機能を有効にします。なお、SPEビット = 1 の状態での書き換えを禁止するレジスタがあります。注意してください。コマンドレジスタ（SPCMD）はTENDビット = 1 の状態であれば SPEビットの値に制約はありません。

・コマンドシーケンスのオペコードとアドレスなどのマスタ出力データを転送します。

・送信FIFOのオーバフローを防ぐため、コマンドは 8バイト以下にする必要があります。

・マスタ動作時に受信FIFOがオーバフローすると転送が停止するため、受信データをダミーリードします。

・SPCRレジスタの設定（SPEビット = 0 ）
【機能】・RSPIの機能を無効化
転送機能を無効にします。この設定によりSSL信号がネゲートされます。

図7　参考プログラムのライト用コマンド転送フロー

2.4.3 メイン関数

図 8に参考プログラムのメイン関数フローを示します。参考プログラムは、シリアルフラッシュメモリの全領域にライトした後、リードした値と等しいかをチェックします。

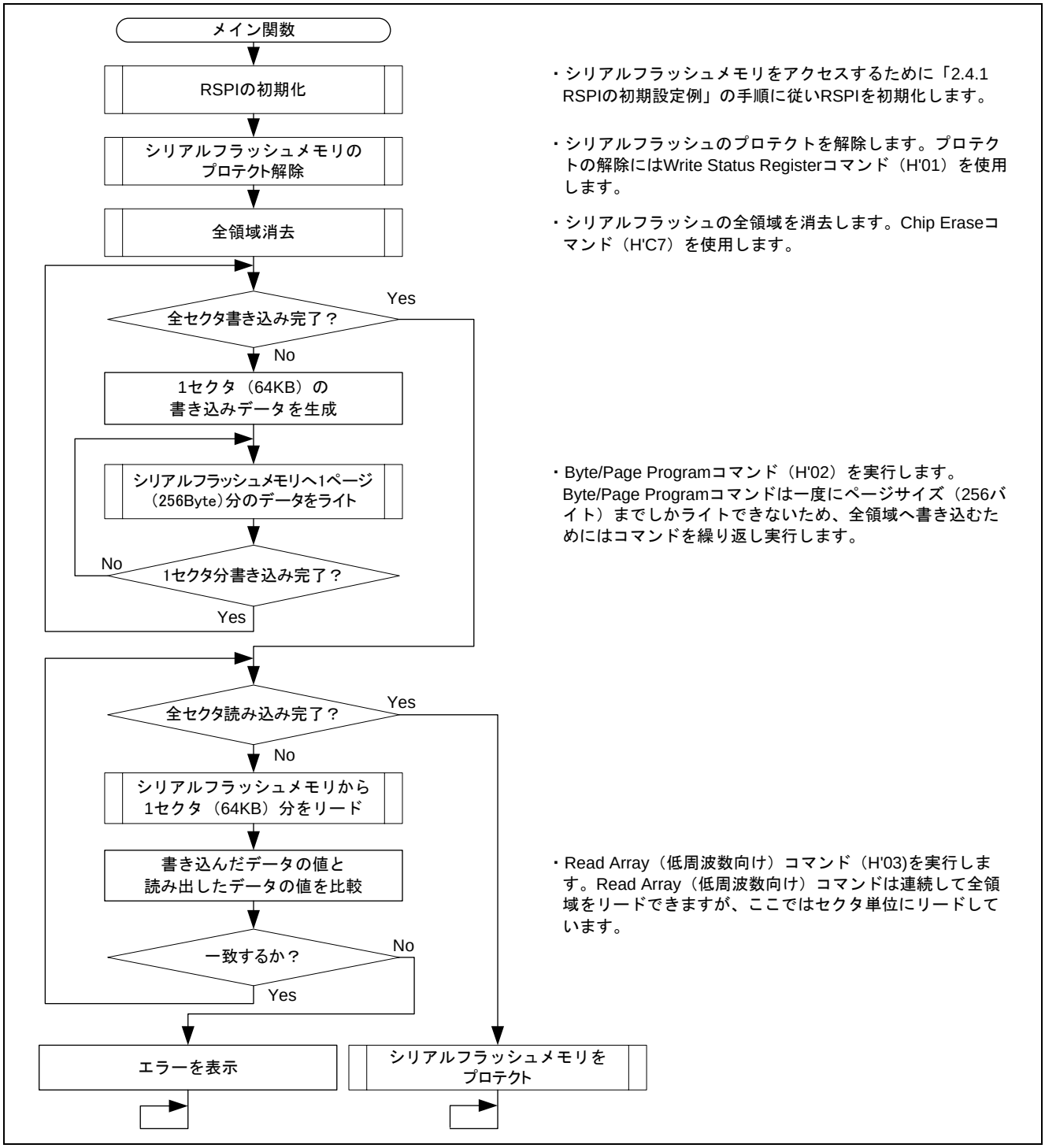


図8　参考プログラムのメイン関数フロー

3. 参考プログラムリスト

3.1 サンプルプログラムリスト"main.c"（1）

```
1     /*""FILE COMMENT""*********** Technical reference data ************************
2     *
3     *      System Name : SH7264 Sample Program
4     *      File Name   : main.c
5     *      Abstract    : ルネサスシリアルペリフェラルインタフェース シリアルフラッシュメモリ接続例
6     *      Version     : 1.00.00
7     *      Device      : SH7262/SH7264
8     *      Tool-Chain  : High-performance Embedded Workshop (Version 4.04.01)
9     *                  : C/C++ compiler package for the SuperH RISC engine family
10    *                  :                          (Ver.9.02 Release00).
11    *      OS          : None
12    *      H/W Platform: M3A-HS64G50(CPU board)
13    *      Disclaimer  :
14    *                  <注意事項>
15    *                  本サンプルプログラムはすべて参考資料であり、
16    *                  その動作を保証するものではありません。
17    *                  本サンプルプログラムはお客様のソフトウエア開発時の
18    *                  技術参考資料としてご利用ください。
19    *
20    *      The information described here may contain technical inaccuracies or
21    *      typographical errors. Renesas Technology Corporation and Renesas Solutions
22    *      assume no responsibility for any damage, liability, or other loss rising
23    *      from these inaccuracies or errors.
24    *
25    *      Copyright (C) 2009 Renesas Technology Corp. All Rights Reserved
26    *      AND Renesas Solutions Corp. All Rights Reserved
27    *
28    *      History     : Feb.23,2009 Ver.1.00.00
29    *""FILE COMMENT END""*************************************************************/
30    #include <stdio.h>
31    #include "serial_flash.h"
32
33    /* ==== マクロ定義 ==== */
34    #define TOP_ADDRESS    0              /* シリアルフラッシュメモリの先頭アドレス */
35
36    /* ==== 関数プロトタイプ宣言 ==== */
37    void main(void);
38
39    /* ==== 変数定義 ==== */
40    #pragma section DEBUG_128K_BYTES
41    static unsigned char data[SF_SECTOR_SIZE];
42    static unsigned char rbuf[SF_SECTOR_SIZE];
43    #pragma section
44
```

3.2 サンプルプログラムリスト"main.c"（2）

```
/*""FUNC COMMENT""*****************************************************************
 * ID         :
 * Outline    : シリアルフラッシュメモリアクセス メイン処理
 *-----------------------------------------------------------------------------
 * Include    :
 *-----------------------------------------------------------------------------
 * Declaration : void main(void);
 *-----------------------------------------------------------------------------
 * Function   : シリアルフラッシュメモリへのイレース、プログラム、リード処理を
 *            : 行います。RSPI チャネル 0 を初期化後、全領域をイレースした後、先頭から
 *            : 全領域にデータを書き込みます。結果は読み出して確認します。
 *-----------------------------------------------------------------------------
 * Argument   : void
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
void main(void)
{
  int i, j;
  static unsigned long addr;

  /* ==== RSPI の初期化 ==== */
  sf_init_serial_flash();

  /* ==== シリアルフラッシュメモリのプロテクト解除 ==== */
  sf_protect_ctrl( SF_REQ_UNPROTECT );

  /* ==== チップイレース（2MB、完了までに約 10 秒かかります） ==== */
  sf_chip_erase();

  /* ==== データライト（2MB、完了までに約 10 秒かかります） ==== */
  addr = TOP_ADDRESS;
  for(i = 0; i < SF_NUM_OF_SECTOR; i++){
    /* ---- データ初期化（64KB） ---- */
    for(j = 0; j < SF_SECTOR_SIZE; j++){
        data[j] = (i + j) % 100;
    }
    /* ---- セクタサイズ（64KB）をライト ---- */
    for(j = 0; j < ( SF_SECTOR_SIZE / SF_PAGE_SIZE ); j++){
        /* ---- ページサイズ（256Byte）をライト ---- */
        sf_byte_program( addr, data+(j*SF_PAGE_SIZE), SF_PAGE_SIZE );
        addr += SF_PAGE_SIZE;             /* 書き込み先アドレス更新 */
    }
  }
```

3.3 サンプルプログラムリスト"main.c"（3）

```
89      /* ==== データリード（2MB） ==== */
90      addr = TOP_ADDRESS;
91      for(i = 0; i < SF_NUM_OF_SECTOR; i++){
92        /* ---- セクタサイズ（64KB）をリード ---- */
93        sf_byte_read( addr, rbuf, SF_SECTOR_SIZE );
94        addr += SF_SECTOR_SIZE;                    /* 読み込み先アドレス更新 */
95
96        /* ---- ベリファイチェック ---- */
97        for(j = 0; j < SF_SECTOR_SIZE; j++){
98            data[j] = (i + j) % 100;               /* 書き込んだデータを再生 */
99            if( data[j] != rbuf[j] ){
100               puts("Error: verify error¥n");
101               fflush(stdout);
102               while(1);
103           }
104       }
105     }
106     /* ==== シリアルフラッシュメモリのプロテクト ==== */
107     sf_protect_ctrl( SF_REQ_PROTECT );
108
109     while(1){
110         /* loop */
111      }
112   }
113
114   /* End of File */
115
```

3.4 サンプルプログラムリスト"serial_flash.c"（1）

```
1     /*""FILE COMMENT""*********** Technical reference data ************************
2     *
3     *      System Name : SH7264 Sample Program
4     *      File Name   : serial_flash.c
5     *      Abstract    : ルネサスシリアルペリフェラルインタフェース シリアルフラッシュメモリ接続例
6     *      Version     : 1.00.00
7     *      Device      : SH7262/SH7264
8     *      Tool-Chain  : High-performance Embedded Workshop (Version 4.04.01)
9     *                  : C/C++ compiler package for the SuperH RISC engine family
10    *                  :                          (Ver.9.02 Release00).
11    *      OS          : None
12    *      H/W Platform: M3A-HS64G50(CPU board)
13    *      Disclaimer  :
14    *                  <注意事項>
15    *                  本サンプルプログラムはすべて参考資料であり、
16    *                  その動作を保証するものではありません。
17    *                  本サンプルプログラムはお客様のソフトウエア開発時の
18    *                  技術参考資料としてご利用ください。
19    *
20    *      The information described here may contain technical inaccuracies or
21    *      typographical errors. Renesas Technology Corporation and Renesas Solutions
22    *      assume no responsibility for any damage, liability, or other loss rising
23    *      from these inaccuracies or errors.
24    *
25    *      Copyright (C) 2009 Renesas Technology Corp. All Rights Reserved
26    *      AND Renesas Solutions Corp. All Rights Reserved
27    *
28    *      History     : Feb.23,2009 Ver.1.00.00
29    *""FILE COMMENT END""*************************************************************/
30    #include <stdio.h>
31    #include <machine.h>
32    #include "iodefine.h"
33    #include "serial_flash.h"
34
35    /* ==== マクロ定義 ==== */
36    #define SFLASHCMD_CHIP_ERASE      0xc7
37    #define SFLASHCMD_SECTOR_ERASE    0xd8
38    #define SFLASHCMD_BYTE_PROGRAM    0x02
39    #define SFLASHCMD_BYTE_READ       0x0B
40    #define SFLASHCMD_BYTE_READ_LOW   0x03
41    #define SFLASHCMD_WRITE_ENABLE    0x06
42    #define SFLASHCMD_WRITE_DISABLE   0x04
43    #define SFLASHCMD_READ_STATUS     0x05
44    #define SFLASHCMD_WRITE_STATUS    0x01
45    #define UNPROTECT_WR_STATUS       0x00
      #define PROTECT_WR_STATUS         0x3C
```

3.5 サンプルプログラムリスト"serial_flash.c"（2）

```
46
47    /* ==== 関数プロトタイプ宣言 ==== */
48    /*** Local function ***/
49    static void write_enable(void);
50    static void write_disable(void);
51    static void busy_wait(void);
52    static unsigned char read_status(void);
53    static void write_status(unsigned char status);
54    static void io_init_rspi(void);
55    static void io_cmd_exe(unsigned char *ope, int ope_sz, unsigned char *data, int data_sz);
56    static void io_cmd_exe_rdmode(unsigned char *ope, int ope_sz, unsigned char *rd, int rd_sz);
57    static void io_wait_tx_end(void);
58
59    /* ==== 変数定義 ==== */
60
61    /*""FUNC COMMENT""****************************************************************
62     * ID         :
63     * Outline    : シリアルフラッシュメモリの初期化
64     *-------------------------------------------------------------------------------
65     * Include    :
66     *-------------------------------------------------------------------------------
67     * Declaration : void sf_init_serial_flash(void);
68     *-------------------------------------------------------------------------------
69     * Function   : シリアルフラッシュメモリにアクセスするための初期化を行います。
70     *            : ルネサスシリアルペリフェラルインタフェース(RSPI)のチャネル 0 を初期化
71     *            : します。
72     *-------------------------------------------------------------------------------
73     * Argument   : void
74     *-------------------------------------------------------------------------------
75     * Return Value: void
76     *""FUNC COMMENT END""************************************************************/
77    void sf_init_serial_flash(void)
78    {
79      /* ==== RSPI0 の初期化 ==== */
80      io_init_rspi();
81    }
```

3.6 サンプルプログラムリスト"serial_flash.c"（3）

```
82    /*""FUNC COMMENT""*****************************************************************
83     * ID         :
84     * Outline    : プロテクト操作
85     *-------------------------------------------------------------------------------
86     * Include    :
87     *-------------------------------------------------------------------------------
88     * Declaration : void sf_protect_ctrl(enum sf_req req);
89     *-------------------------------------------------------------------------------
90     * Function   : シリアルフラッシュメモリのプロテクト設定または解除を行います。
91     *            : 設定内容は引数 req で指定します。プロテクトの初期値や解除方法は
92     *            : シリアルフラッシュメモリの仕様によって異なります。
93     *-------------------------------------------------------------------------------
94     * Argument   : enum sf_req req ; I : SF_REQ_UNPROTECT -> 全セクタプロテクト解除
95     *            :                       SF_REQ_PROTECT   -> 全セクタプロテクト
96     *-------------------------------------------------------------------------------
97     * Return Value: void
98     *""FUNC COMMENT END""*************************************************************/
99    void sf_protect_ctrl(enum sf_req req)
100   {
101     if( req == SF_REQ_UNPROTECT ){
102       write_status( UNPROTECT_WR_STATUS);       /* 全領域プロテクト解除 */
103     }
104     else{
105       write_status( PROTECT_WR_STATUS );        /* 全領域プロテクト */
106     }
107   }
108   /*""FUNC COMMENT""*****************************************************************
109    * ID         :
110    * Outline    : チップイレース
111    *-------------------------------------------------------------------------------
112    * Include    :
113    *-------------------------------------------------------------------------------
114    * Declaration : void sf_chip_erase(void);
115    *-------------------------------------------------------------------------------
116    * Function   : シリアルフラッシュメモリの全ビットをイレースします。
117    *            : イレースまたはプログラムする前にはライトイネーブルコマンドを
118    *            : 発行する必要があります。またイレースまたはプログラム後は
119    *            : シリアルフラッシュメモリのステータスを確認しビジー状態が解除
120    *            : されたことを確認してください。
121    *-------------------------------------------------------------------------------
122    * Argument   : void
123    *-------------------------------------------------------------------------------
124    * Return Value: void
125    *""FUNC COMMENT END""*************************************************************/
126   void sf_chip_erase(void)
127   {
128     unsigned char cmd[1];
129     cmd[0] = SFLASHCMD_CHIP_ERASE;
130
131     write_enable();
132     io_cmd_exe(cmd, 1, NULL, 0);
133     busy_wait();
134   }
```

3.7 サンプルプログラムリスト"serial_flash.c"（4）

```
135  /*""FUNC COMMENT""*****************************************************************
136   * ID         :
137   * Outline    : セクタイレース
138   *-------------------------------------------------------------------------------
139   * Include    :
140   *-------------------------------------------------------------------------------
141   * Declaration : void sf_sector_erase(int sector_no);
142   *-------------------------------------------------------------------------------
143   * Function   : シリアルフラッシュメモリの指定セクタをイレースします。
144   *            : イレースまたはプログラムする前にはライトイネーブルコマンドを
145   *            : 発行する必要があります。またイレースまたはプログラム後は
146   *            : シリアルフラッシュメモリのステータスを確認しビジー状態が解除
147   *            : されたことを確認してください。
148   *-------------------------------------------------------------------------------
149   * Argument   : int sector_no ; I : セクタ番号
150   *-------------------------------------------------------------------------------
151   * Return Value: void
152   *""FUNC COMMENT END""*************************************************************/
153  void sf_sector_erase(int sector_no)
154  {
155    unsigned char cmd[4];
156    unsigned long addr = sector_no * SF_SECTOR_SIZE;
157
158    cmd[0] = SFLASHCMD_SECTOR_ERASE;
159    cmd[1] = (addr >> 16) & 0xff;
160    cmd[2] = (addr >>  8) & 0xff;
161    cmd[3] =  addr        & 0xff;
162
163    write_enable();
164    io_cmd_exe(cmd, 4, NULL, 0);
165    busy_wait();
166  }
```

3.8 サンプルプログラムリスト"serial_flash.c"（5）

```
167  /*""FUNC COMMENT""*****************************************************************
168   * ID         :
169   * Outline    : データプログラム
170   *-------------------------------------------------------------------------------
171   * Include    :
172   *-------------------------------------------------------------------------------
173   * Declaration : void sf_byte_program(unsigned long addr, unsigned char *buf, int size);
174   *-------------------------------------------------------------------------------
175   * Function   : シリアルフラッシュメモリに指定データをプログラムします。
176   *            : イレースまたはプログラムする前にはライトイネーブルコマンドを
177   *            : 発行する必要があります。またイレースまたはプログラム後は
178   *            : シリアルフラッシュメモリのステータスを確認しビジー状態が解除
179   *            : されたことを確認してください。
180   *            : 最大ライトデータサイズはデバイスによって制限されます。
181   *-------------------------------------------------------------------------------
182   * Argument   : unsigned long addr ; I : ライトするシリアルフラッシュメモリのアドレス
183   *            : unsigned char *buf ; I : ライトデータを格納するバッファのアドレス
184   *            : int size           ; I : ライトするバイト数
185   *-------------------------------------------------------------------------------
186   * Return Value: void
187   *""FUNC COMMENT END""*************************************************************/
188  void sf_byte_program(unsigned long addr, unsigned char *buf, int size)
189  {
190    unsigned char cmd[4];
191
192    cmd[0] = SFLASHCMD_BYTE_PROGRAM;
193    cmd[1] = (unsigned char)((addr >> 16) & 0xff);
194    cmd[2] = (unsigned char)((addr >>  8) & 0xff);
195    cmd[3] = (unsigned char)( addr        & 0xff);
196    write_enable();
197    io_cmd_exe(cmd, 4, buf, size);
198    busy_wait();
199  }
```

3.9 サンプルプログラムリスト"serial_flash.c"（6）

```
/*""FUNC COMMENT""*****************************************************************
 * ID         :
 * Outline    : データリード
 *-----------------------------------------------------------------------------
 * Include    :
 *-----------------------------------------------------------------------------
 * Declaration : void sf_byte_read(unsigned long addr, unsigned char *buf, int size);
 *-----------------------------------------------------------------------------
 * Function   : シリアルフラッシュメモリを指定バイト数だけリードします。
 *-----------------------------------------------------------------------------
 * Argument   : unsigned long addr ; I : リードするシリアルフラッシュメモリのアドレス
 *            : unsigned char *buf ; I : リードデータを格納するバッファのアドレス
 *            : int size           ; I : リードするバイト数
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
void sf_byte_read(unsigned long addr, unsigned char *buf, int size)
{
  unsigned char cmd[4];

  cmd[0] = SFLASHCMD_BYTE_READ_LOW;
  cmd[1] = (unsigned char)((addr >> 16) & 0xff);
  cmd[2] = (unsigned char)((addr >>  8) & 0xff);
  cmd[3] = (unsigned char)( addr        & 0xff);
  io_cmd_exe_rdmode(cmd, 4, buf, size);
}

/*""FUNC COMMENT""*****************************************************************
 * ID         :
 * Outline    : 書き込み許可
 *-----------------------------------------------------------------------------
 * Include    :
 *-----------------------------------------------------------------------------
 * Declaration : static void write_enable(void);
 *-----------------------------------------------------------------------------
 * Function   : ライトイネーブルコマンドを発行して、シリアルフラッシュメモリへの
 *            : イレースまたはプログラム動作を許可します。
 *-----------------------------------------------------------------------------
 * Argument   : void
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
static void write_enable(void)
{
  unsigned char cmd[1];
  cmd[0] = SFLASHCMD_WRITE_ENABLE;
  io_cmd_exe(cmd, 1, NULL, 0);
}
```

3.10 サンプルプログラムリスト"serial_flash.c"（7）

```
248  /*""FUNC COMMENT""*****************************************************************
249   * ID         :
250   * Outline    : 書き込み禁止
251   *-------------------------------------------------------------------------------
252   * Include    :
253   *-------------------------------------------------------------------------------
254   * Declaration : static void write_disable(void);
255   *-------------------------------------------------------------------------------
256   * Function   : ライトディスエーブルコマンドを発行して、シリアルフラッシュメモリへの
257   *            : イレースまたはプログラム動作を禁止します。
258   *-------------------------------------------------------------------------------
259   * Argument   : void
260   *-------------------------------------------------------------------------------
261   * Return Value: void
262   *""FUNC COMMENT END""*************************************************************/
263  static void write_disable(void)
264  {
265    unsigned char cmd[1];
266    cmd[0] = SFLASHCMD_WRITE_DISABLE;
267    io_cmd_exe(cmd, 1, NULL, 0);
268  }
269
270  /*""FUNC COMMENT""*****************************************************************
271   * ID         :
272   * Outline    : ビジー待ち
273   *-------------------------------------------------------------------------------
274   * Include    :
275   *-------------------------------------------------------------------------------
276   * Declaration : static void busy_wait(void);
277   *-------------------------------------------------------------------------------
278   * Function   : シリアルフラッシュメモリのステータスがビジー状態の場合は内部で
279   *            : ループします。
280   *-------------------------------------------------------------------------------
281   * Argument   : void
282   *-------------------------------------------------------------------------------
283   * Return Value: void
284   *""FUNC COMMENT END""*************************************************************/
285  static void busy_wait(void)
286  {
287    while ((read_status() & 0x01) != 0) {   /* RDY/BSY */
288      /* serial flash is busy */
289    }
290  }
```

3.11 サンプルプログラムリスト"serial_flash.c"（8）

```
291  /*""FUNC COMMENT""*****************************************************************
292   * ID         :
293   * Outline    : ステータスリード
294   *-------------------------------------------------------------------------------
295   * Include    :
296   *-------------------------------------------------------------------------------
297   * Declaration : static unsigned char read_status(void);
298   *-------------------------------------------------------------------------------
299   * Function   : シリアルフラッシュメモリのステータスをリードします。
300   *-------------------------------------------------------------------------------
301   * Argument   : void
302   *-------------------------------------------------------------------------------
303   * Return Value: ステータスレジスタの値
304   *""FUNC COMMENT END""*************************************************************/
305  static unsigned char read_status(void)
306  {
307    unsigned char buf;
308    unsigned char cmd[1];
309
310    cmd[0] = SFLASHCMD_READ_STATUS;
311    io_cmd_exe_rdmode(cmd, 1, &buf, 1);
312    return buf;
313  }
314
315  /*""FUNC COMMENT""*****************************************************************
316   * ID         :
317   * Outline    : ステータスライト
318   *-------------------------------------------------------------------------------
319   * Include    :
320   *-------------------------------------------------------------------------------
321   * Declaration : static void write_status(unsigned char status);
322   *-------------------------------------------------------------------------------
323   * Function   : シリアルフラッシュメモリのステータスをライトします。
324   *-------------------------------------------------------------------------------
325   * Argument   : unsigned char status ; I : status register value
326   *-------------------------------------------------------------------------------
327   * Return Value: void
328   *""FUNC COMMENT END""*************************************************************/
329  static void write_status(unsigned char status)
330  {
331    unsigned char cmd[2];
332
333    cmd[0] = SFLASHCMD_WRITE_STATUS;
334    cmd[1] = status;
335
336    write_enable();
337    io_cmd_exe(cmd, 2, NULL, 0);
338    busy_wait();
339  }
340
```

3.12 サンプルプログラムリスト"serial_flash.c"（9）

```
/*""FUNC COMMENT""*****************************************************************
 * ID         :
 * Outline    : RSPI の初期化
 *-----------------------------------------------------------------------------
 * Include    :
 *-----------------------------------------------------------------------------
 * Declaration : static void io_init_rspi(void);
 *-----------------------------------------------------------------------------
 * Function   : ルネサスシリアルペリフェラルインタフェースのチャネル 0 を初期化します。
 *            : マスタモードに設定し、シリアルフラッシュメモリの仕様に合わせた
 *            : 転送設定を行います。
 *-----------------------------------------------------------------------------
 * Argument   : void
 *-----------------------------------------------------------------------------
 * Return Value: void
 *""FUNC COMMENT END""*************************************************************/
static void io_init_rspi(void)
{
  /* ==== PORT ==== */
  PORT.PFCR3.BIT.PF12MD = 3;  /* PF12:MISO0 */
  PORT.PFCR2.BIT.PF11MD = 3;  /* PF11:MOSI0 */
  PORT.PFCR2.BIT.PF10MD = 3;  /* PF10:SSL00 */
  PORT.PFCR2.BIT.PF9MD  = 3;  /* PF9:RSPCK0 */

  /* ==== CPG ==== */
  CPG.STBCR5.BIT.MSTP51 = 0;  /* RSPI0 active */

  /* ==== RSPI ==== */
  RSPI0.SPCR.BYTE   =  0x00;  /* RSPI チャネル 0 を動作禁止 */
  RSPI0.SPPCR.BYTE  =  0x30;  /* MOSI アイドル固定値 = 1 */
  RSPI0.SPBR.BYTE   = 0x01;   /* ベースのビットレートを 18MHz に設定(BΦ=72MHz) */
  RSPI0.SPDCR.BYTE  =  0x20;  /* ダミーデータ送信禁止 */
                              /* SPDR レジスタのアクセス幅：8 ビット */
  RSPI0.SPCKD.BYTE  =  0x00;  /* RSPCK 遅延：1 RSPCK */
  RSPI0.SSLND.BYTE  =  0x00;  /* SSL ネゲート遅延：1 RSPCK */
  RSPI0.SPND.BYTE   =  0x00;  /* 次アクセス遅延：1 RSPCK + 2 BΦ */
  RSPI0.SPSCR.BYTE  =  0x00;  /* シーケンス長：1 (SPCMD0 のみ使用) */
  RSPI0.SPCMD0.WORD = 0xE783; /* MSB ファースト */
                              /* データ長：8bit */
                              /* 転送終了後も SSL 信号レベルを保持する */
                              /* ビットレート：ベースビットレートの分周なし */
                              /* アイドル時の RSPCK：1 */
                              /* 奇数エッジでデータ変化、偶数エッジでデータサンプル */
  RSPI0.SPBFCR.BYTE =  0xC0;  /* 送受信バッファのデータリセット許可 */
  RSPI0.SPBFCR.BYTE =  0x00;  /* 送受信バッファのデータリセット禁止 */
                              /* 送信バッファのトリガ：1 バイト以上の空き */
                              /* 受信バッファのトリガ：1 バイト以上の受信 */
  RSPI0.SSLP.BYTE   =  0x00;  /* SSLP = b'0 SSL signal 0-active */
  RSPI0.SPCR.BYTE   =  0x48;  /* マスタモード */
                              /* 割り込み禁止 */
                              /* RSPI チャネル 0 の動作許可 */
}
```

3.13 サンプルプログラムリスト"serial_flash.c"（10）

```
393  /*""FUNC COMMENT""*****************************************************************
394   * ID         :
395   * Outline    : コマンド実行(リードデータなし)
396   *-------------------------------------------------------------------------------
397   * Include    :
398   *-------------------------------------------------------------------------------
399   * Declaration : static void io_cmd_exe(unsigned char *ope, int ope_sz,
400   *             :                   unsigned char *data,int data_sz)
401   *-------------------------------------------------------------------------------
402   * Function    :指定されたコマンドを実行します。
403   *             : 引数 ope を送信した後、引数 data を送信します。受信データは破棄します。
404   *             : ope_sz は 0～8 のいずれかの値を設定してください。
405   *             : data_sz は 0～256 のいずれかの値を設定してください。
406   *-------------------------------------------------------------------------------
407   * Argument    : unsigned char *ope ; I : 送信するオペコード部とアドレス部の先頭アドレス
408   *             : int ope_sz         ; I : オペコード部とアドレス部のバイト数
409   *             : unsigned char *data; I : 送信するデータ部の先頭アドレス
410   *             : int data_sz        ; I : データ部のバイト数
411   *-------------------------------------------------------------------------------
412   * Return Value: void
413   *""FUNC COMMENT END""*************************************************************/
414  static void io_cmd_exe(unsigned char *ope, int ope_sz, unsigned char *data, int data_sz)
415  {
416    unsigned char tmp;
417
418    /* ==== バッファリセット ==== */
419    RSPI0.SPBFCR.BYTE  = 0xC0u;
420    RSPI0.SPBFCR.BYTE  = 0x00u;
421
422    /* ---- SPI 転送許可 ---- */
423    RSPI0.SPCR.BIT.SPE = 1;
424
425    /* ==== MOSI(コマンド、アドレス、ライトデータ) ==== */
426    while(ope_sz--){
427      RSPI0.SPDR.BYTE = *ope++;     /* コマンドは 8 バイト以下とする */
428    }
429    while(data_sz--){
430      while( RSPI0.SPSR.BIT.SPTEF == 0 ){
431          /* wait */
432      }
433      RSPI0.SPDR.BYTE = *data++;
434      if( RSPI0.SPSR.BIT.SPRF == 1 ){
435          tmp = RSPI0.SPDR.BYTE;     /* オーバフロー防止のためのダミーリード */
436      }
437    }
438    io_wait_tx_end();               /* 完了待ち */
439
440    /* ---- SPI 転送終了（SSL ネゲート） ---- */
441    RSPI0.SPCR.BIT.SPE = 0;
442  }
```

3.14 サンプルプログラムリスト"serial_flash.c"（11）

```
443  /*""FUNC COMMENT""*****************************************************************
444   * ID         :
445   * Outline    : コマンド実行(リードデータあり)
446   *-------------------------------------------------------------------------------
447   * Include    :
448   *-------------------------------------------------------------------------------
449   * Declaration : static void io_cmd_exe_rdmode(unsigned char *ope, int ope_sz,
450   *             :                               unsigned char *rd,  int rd_sz)
451   *-------------------------------------------------------------------------------
452   * Function   : 指定されたコマンドを実行します。
453   *            : 引数 ope を送信した後、引数 rd にデータを受信します。
454   *            : ope_sz は 0～8 のいずれかの値を設定してください。
455   *            : rd_sz は 0 以上 の値を設定することが可能です。
456   *-------------------------------------------------------------------------------
457   * Argument   : unsigned char *ope ; I : 送信するオペコード部とアドレス部の先頭アドレス
458   *            : int ope_sz         ; I : オペコード部とアドレス部のバイト数
459   *            : unsigned char *rd  ; I : 受信データを格納するバッファアドレス
460   *            : int rd_sz          ; I : データ部のバイト数
461   *-------------------------------------------------------------------------------
462   * Return Value: void
463   *""FUNC COMMENT END""*************************************************************/
464  static void io_cmd_exe_rdmode(unsigned char *ope, int ope_sz, unsigned char *rd, int rd_sz)
465  {
466    /* ==== バッファリセット ==== */
467    RSPI0.SPBFCR.BYTE  = 0xC0u;
468    RSPI0.SPBFCR.BYTE  = 0x00u;
469
470    /* ---- SPI 転送許可 ---- */
471    RSPI0.SPCR.BIT.SPE = 1;
472
473    /* ---- MOSI(コマンド、アドレス、ダミー) ---- */
474    while(ope_sz--){
475      RSPI0.SPDR.BYTE = *ope++;      /* コマンドは 8 バイト以下とする */
476    }
477    io_wait_tx_end();                /* 完了待ち */
478
479    /* ---- MISO(リードデータ) ---- */
480    RSPI0.SPBFCR.BYTE  = 0xC0u;      /* バッファリセット */
481    RSPI0.SPBFCR.BYTE  = 0x00u;
482
483    RSPI0.SPDCR.BIT.TXDMY = 1;       /* ダミー送信 許可 */
484    while(rd_sz--){
485      while( RSPI0.SPSR.BIT.SPRF == 0){
486          /* wait */
487      }
488      *rd++ = RSPI0.SPDR.BYTE;
489    }
490    RSPI0.SPDCR.BIT.TXDMY = 0;       /* ダミー送信 禁止 */
491    io_wait_tx_end();                /* 完了待ち */
492
493    /* ---- SPI 転送終了（SSL ネゲート） ---- */
494    RSPI0.SPCR.BIT.SPE = 0;
495  }
```

3.15 サンプルプログラムリスト"serial_flash.c"（12）

```
496   /*""FUNC COMMENT""*****************************************************************
497    * ID         :
498    * Outline    : 送信完了待ち
499    *-------------------------------------------------------------------------------
500    * Include    :
501    *-------------------------------------------------------------------------------
502    * Declaration : static void io_wait_tx_end(void);
503    *-------------------------------------------------------------------------------
504    * Function   : 送信完了が確認できるまで内部でループします。
505    *-------------------------------------------------------------------------------
506    * Argument   : void
507    *-------------------------------------------------------------------------------
508    * Return Value: void
509    *""FUNC COMMENT END""*************************************************************/
510   static void io_wait_tx_end(void)
511   {
512     while(RSPI0.SPSR.BIT.TEND == 0){
513       /* wait */
514     }
515   }
516
517   /* End of File */
```

3.16 サンプルプログラムリスト"serial_flash.h"（1）

```
1   /*""FILE COMMENT""*********** Technical reference data ************************
2   *
3   *      System Name : SH7264 Sample Program
4   *      File Name   : serial_flash.h
5   *      Abstract    : ルネサスシリアルペリフェラルインタフェース シリアルフラッシュメモリ接続例
6   *      Version     : 1.00.00
7   *      Device      : SH7262/SH7264
8   *      Tool-Chain  : High-performance Embedded Workshop (Version 4.04.01)
9   *                  : C/C++ compiler package for the SuperH RISC engine family
10  *                  :                          (Ver.9.02 Release00).
11  *      OS          : None
12  *      H/W Platform: M3A-HS64G50(CPU board)
13  *      Disclaimer  :
14  *                  <注意事項>
15  *                  本サンプルプログラムはすべて参考資料であり、
16  *                  その動作を保証するものではありません。
17  *                  本サンプルプログラムはお客様のソフトウエア開発時の
18  *                  技術参考資料としてご利用ください。
19  *
20  *      The information described here may contain technical inaccuracies or
21  *      typographical errors. Renesas Technology Corporation and Renesas Solutions
22  *      assume no responsibility for any damage, liability, or other loss rising
23  *      from these inaccuracies or errors.
24  *
25  *      Copyright (C) 2009 Renesas Technology Corp. All Rights Reserved
26  *      AND Renesas Solutions Corp. All Rights Reserved
27  *
28  *      History     : Feb.23,2009 Ver.1.00.00
29  *""FILE COMMENT END""*************************************************************/
30  #ifndef _SERIAL_FLASH_H_
31  #define _SERIAL_FLASH_H_
32
33  /* ==== マクロ定義 ==== */
34  #define SF_PAGE_SIZE       256         /* シリアルフラッシュメモリのページサイズ */
35  #define SF_SECTOR_SIZE     0x10000     /* セクタサイズ = 64KB   */
36  #define SF_NUM_OF_SECTOR   32          /* セクタ数 32   */
37  enum sf_req{
38    SF_REQ_PROTECT = 0,                  /* プロテクト要求 */
39    SF_REQ_UNPROTECT                     /* プロテクト解除要求 */
40  };
41  /* ==== 関数プロトタイプ宣言 ==== */
42  void sf_init_serial_flash(void);
43  void sf_protect_ctrl(enum sf_req req);
44  void sf_chip_erase(void);
45  void sf_sector_erase(int sector_no);
46  void sf_byte_program(unsigned long addr, unsigned char *buf, int size);
47  void sf_byte_read(unsigned long addr, unsigned char *buf, int size);
48
49  #endif /* _SERIAL_FLASH_H_ */
50  /* End of File */
51
```

4. 参考ドキュメント

- ソフトウェアマニュアル
  SH-2A/SH-2A-FPU ソフトウェアマニュアル　Rev.3.00
  （最新版をルネサス テクノロジホームページから入手してください。）
- ハードウェアマニュアル
  SH7262 グループ、SH7264 グループ ハードウェアマニュアル　Rev.1.00
  （最新版をルネサス テクノロジ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

ルネサステクノロジホームページ
http://japan.renesas.com/

お問合せ先
http://japan.renesas.com/inquiry
csc@renesas.com

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2009.03.10		初版発行

本資料ご利用に際しての留意事項

1. 本資料は、お客様に用途に応じた適切な弊社製品をご購入いただくための参考資料であり、本資料中に記載の技術情報について弊社または第三者の知的財産権その他の権利の実施、使用を許諾または保証するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例など全ての情報の使用に起因する損害、第三者の知的財産権その他の権利に対する侵害に関し、弊社は責任を負いません。
3. 本資料に記載の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、あるいはその他軍事用途の目的で使用しないでください。また、輸出に際しては、「外国為替および外国貿易法」その他輸出関連法令を遵守し、それらの定めるところにより必要な手続を行ってください。
4. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例などの全ての情報は本資料発行時点のものであり、弊社は本資料に記載した製品または仕様等を予告なしに変更することがあります。弊社の半導体製品のご購入およびご使用に当たりましては、事前に弊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、弊社ホームページ（http://www.renesas.com）などを通じて公開される情報に常にご注意ください。
5. 本資料に記載した情報は、正確を期すため慎重に制作したものですが、万一本資料の記述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、弊社はその責任を負いません。
6. 本資料に記載の製品データ、図、表などに示す技術的な内容、プログラム、アルゴリズムその他応用回路例などの情報を流用する場合は、流用する情報を単独で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。弊社は、適用可否に対する責任は負いません。
7. 本資料に記載された製品は、各種安全装置や運輸・交通用、医療用、燃焼制御用、航空宇宙用、原子力、海底中継用の機器・システムなど、その故障や誤動作が直接人命を脅かしあるいは人体に危害を及ぼすおそれのあるような機器・システムや特に高度な品質・信頼性が要求される機器・システムでの使用を意図して設計、製造されたものではありません（弊社が自動車用と指定する製品を自動車に使用する場合を除きます）。これらの用途に利用されることをご検討の際には、必ず事前に弊社営業窓口へご照会ください。なお、上記用途に使用されたことにより発生した損害等について弊社はその責任を負いかねますのでご了承願います。
8. 第7項にかかわらず、本資料に記載された製品は、下記の用途には使用しないでください。これらの用途に使用されたことにより発生した損害等につきましては、弊社は一切の責任を負いません。
   1）生命維持装置。
   2）人体に埋め込み使用するもの。
   3）治療行為（患部切り出し、薬剤投与等）を行うもの。
   4）その他、直接人命に影響を与えるもの。
9. 本資料に記載された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、実装条件およびその他諸条件につきましては、弊社保証範囲内でご使用ください。弊社保証値を越えて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、弊社はその責任を負いません。
10. 弊社は製品の品質および信頼性の向上に努めておりますが、特に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。弊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、誤動作防止設計などの安全設計（含むハードウェアおよびソフトウェア）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。
11. 本資料に記載の製品は、これを搭載した製品から剥がれた場合、幼児が口に入れて誤飲する等の事故の危険性があります。お客様の製品への実装後に容易に本製品が剥がれることがなきよう、お客様の責任において十分な安全設計をお願いします。お客様の製品から剥がれた場合の事故につきましては、弊社はその責任を負いません。
12. 本資料の全部または一部を弊社の文書による事前の承諾なしに転載または複製することを固くお断りいたします。
13. 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弊社営業窓口までご照会ください。

D039444