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の高速リードライト例

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高速リードライト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 に接続します。
- 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。
- 大容量データに高速でアクセスするために、アクセス手順を最適化しています。

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　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
- 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。大容量データを想定して高速でアクセスします。

この章では、端子接続例と高速アクセス手順の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／64K バイト／32K バイト／4K バイト
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。

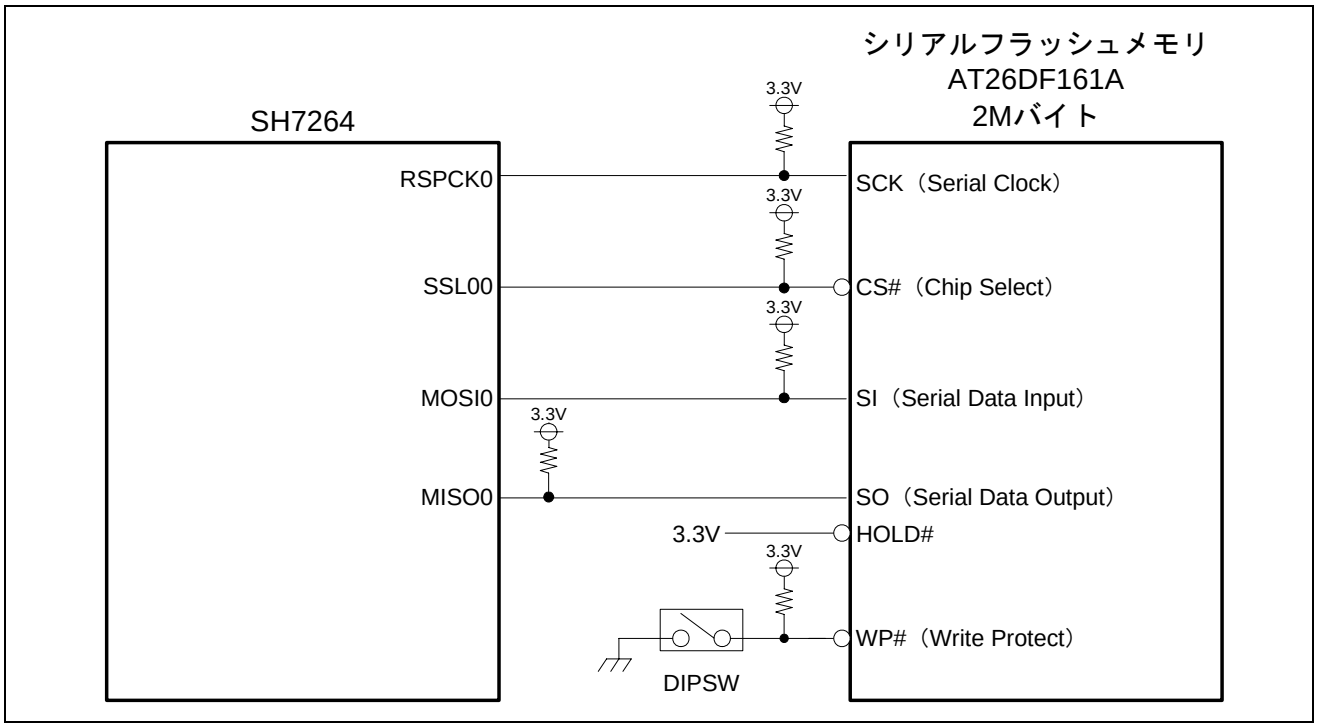


図1　シリアルフラッシュメモリ接続回路例

【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　マルチプレクス端子

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 高速アクセス時のインタフェースタイミング例

高速アクセス時のインタフェースタイミング例を示します。ここでは、標準的な SPI 制御手順によるインタフェースタイミングを示した後、リードライトを高速に行うための手法を説明します。

2.3.1 標準的な SPI 制御手順によるインタフェースタイミング

図 2に標準的なSPI制御手順におけるデータ転送タイミング例を示します。ここでは、本応用例で使用するシリアルフラッシュメモリの仕様に合わせ、マスタ／スレーブとも立ち下がりエッジでデータ変化を行い、1/2 サイクル後の立ち上がりエッジでデータサンプリングを行います。この制御手順では全二重通信が可能です。

標準的な SPI 制御手順の詳細は、アプリケーションノート「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」を参照してください。

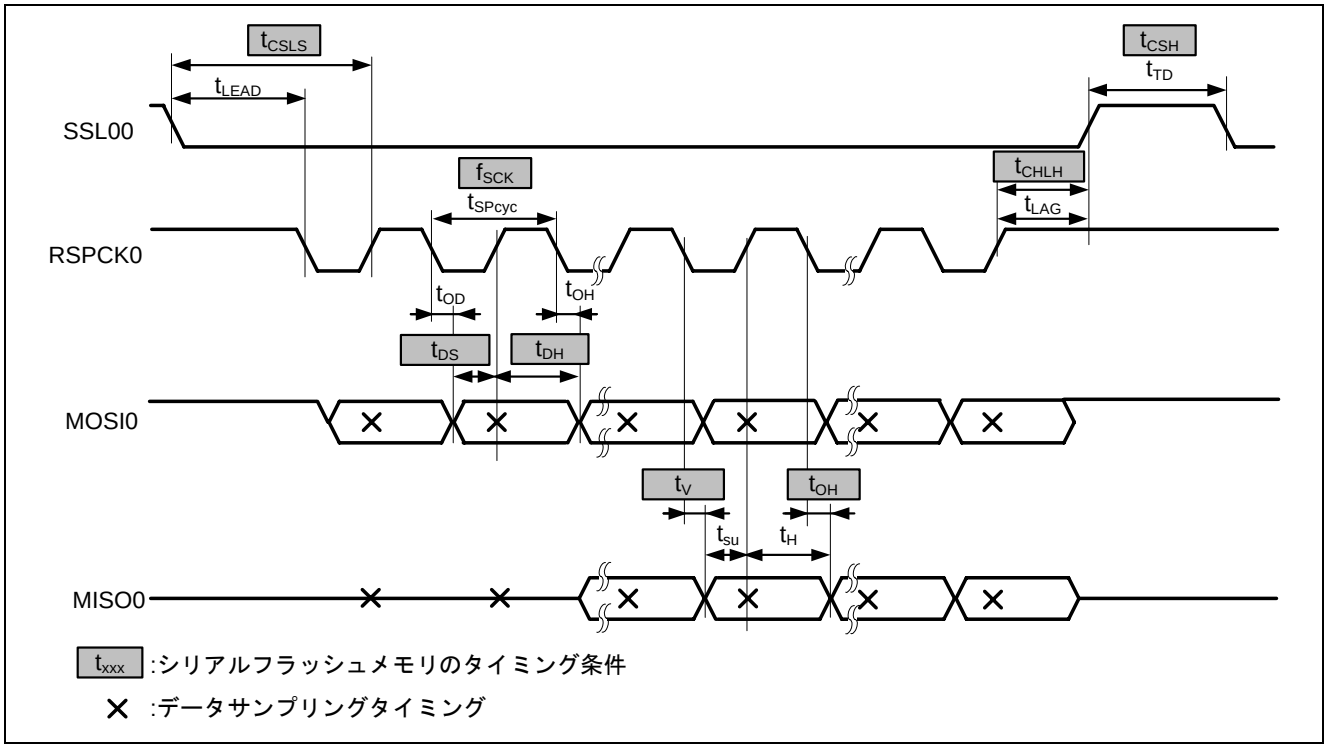


図2 標準的な SPI 制御手順での転送タイミング例（CPOL=1, CPHA=1 の場合）

2.3.2 セットアップ期間とアクセス幅の拡張

ここでは、高速アクセス時の RSPI の設定値とインタフェースタイミングについて説明します。

高速アクセス時の RSPI 設定の特徴は、セットアップ期間を 1 サイクル分に拡張して RSPCK を 36MHz にした点と、データリード時のデータレジスタ（SPDR）のアクセス幅をロングワード長（32 ビット）にした点です。制御がやや複雑になりますが効率的な転送が可能です。

(1) セットアップ期間の拡張

2.3.1節の標準的な制御方法を使用するとセットアップ期間がRSPCKの 1/2 サイクル未満となります。SH7264 のデータ入力セットアップ時間t_{SU} は、15ns（min）ですが、RSPCKを最速の 36MHz（Bφ：72MHz時）に設定した場合は 1/2 サイクル期間が約 13ns（min）となりタイミング条件が満たされません。そのため、セットアップ期間を拡張することによりRSPCK=36MHzに対応します。

Read Array コマンドの場合を例に、セットアップ期間を拡張する方法を以下に説明します。

図 3にRead Arrayコマンド（H'0Bオペコード）のコマンドシーケンスを示します。前半部はSH7264 がコマンドやアドレスを出力するMOSI転送期間、後半部はシリアルフラッシュメモリがデータを出力するMISO転送期間です。セットアップ期間を拡張するために、前半部と後半部のそれぞれにおいてSPCMDレジスタのCPOLビットとCPHAビットの設定を変更します。表 3にCPOLビットとCPHAビットの説明を示します。

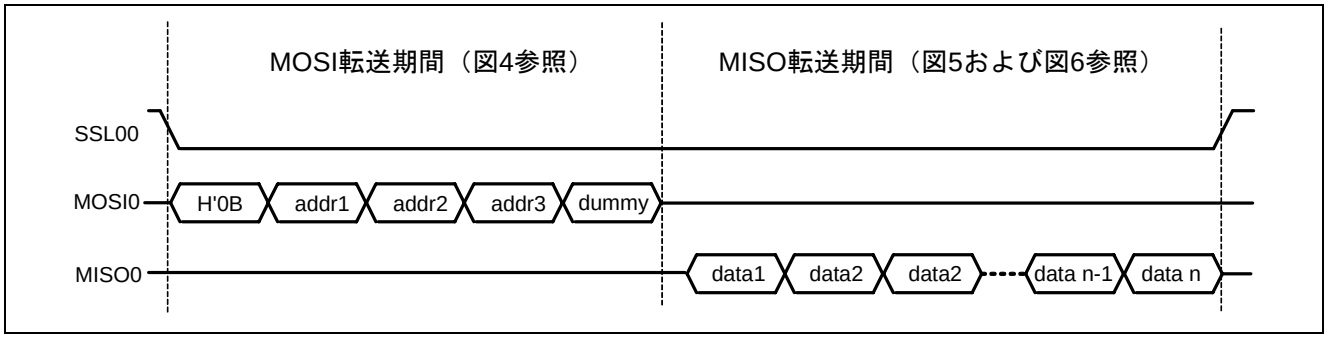


図3　セットアップ期間拡張時のコマンドシーケンス（Read Array コマンド）

表3　CPOL ビットと CPHA ビットの説明

レジスタ名	ビット	ビット名	R/W	説明
コマンドレジスタ（SPCMD）	1	CPOL	R/W	RSPCK 極性設定 マスタモード／スレーブモードの RSPCK 極性を設定するためのビットです。本モジュール間のデータ通信を行う場合、モジュール間で同一の RSPCK 極性を設定する必要があります。 0：アイドル時の RSPCK が 0 1：アイドル時の RSPCK が 1
	0	CPHA	R/W	RSPCK 位相設定 マスタモード／スレーブモードの RSPCK 位相を設定するためのビットです。本モジュール間のデータ通信を行う場合、モジュール間で同一の RSPCK 位相を設定する必要があります。 0：奇数エッジでデータサンプリング、偶数エッジでデータ変化 1：奇数エッジでデータ変化、偶数エッジでデータサンプリング

まず前半部の MOSI 転送期間について説明します。

セットアップ期間を拡張するために、マスタの出力タイミングからスレーブのサンプリングタイミングまでの期間が RSPCK の 1 サイクル分になる設定を行います。本応用例で使用するシリアルフラッシュメモリは、立ち上がりエッジでデータをサンプリングするため、SH7264 はその 1 サイクル前の立ち上がりエッジでデータを出力しておく必要があります。

立ち上がりエッジでデータを出力する設定は（CPOL=1、CPHA=0）または（CPOL=0、CPHA=1）ですがここでは以下に示す理由により（CPOL=1、CPHA=0）を設定します。

CPHA=1 の設定は、SSLアサート時ではなく最初のRSPCKのエッジ（CPOL=0 の場合は立ち上がりエッジ）で、マスタは 1 ビット目のデータを出力します。また、スレーブのデータサンプリングも 1 つ目の立ち上がりエッジで行われます。そのため（CPOL=0、CPHA=1）を設定した場合は、マスタが最初のデータを出力するタイミングでスレーブがデータをサンプリングすることになり、セットアップ条件を満たせません。（CPOL=1、CPHA=0）を設定した場合は、SSLアサート時に、マスタが 1 ビット目のデータを出力します。またスレーブがデータをサンプリングする最初のRSPCKの立ち上がりエッジまでには 1 サイクル以上の時間があるため、セットアップ条件を満たすことができます。2 ビット目以降は、RSPCKの立ち上がりエッジでマスタがデータ出力、次の立ち上がりエッジでスレーブがデータをサンプリングするのでタイミング条件を満たすことができます。図 4に（CPOL=1、CPHA=0）を設定した場合のMOSI転送タイミングを示します。

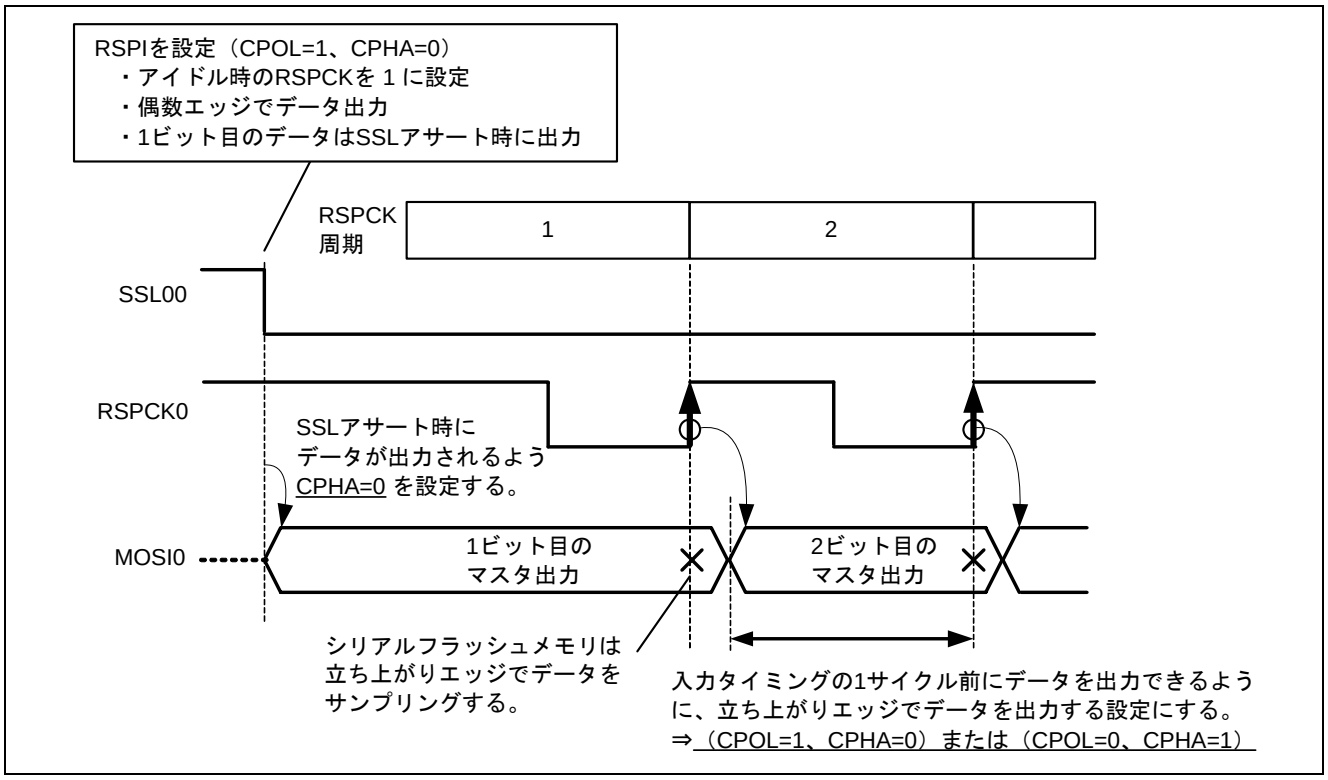


図4　MOSI 転送タイミング

次に、後半部の MISO 転送期間について説明します。

MISO 転送期間ではマスタがデータのサンプリング側になります。スレーブがデータを出力してから 1 サイクル後に、マスタがデータをサンプリングする設定を行います。本応用例で使用するシリアルフラッシュメモリは立ち下がりエッジでデータを出力するため、SH7264 はその 1 サイクル後の立ち下がりエッジでデータをサンプリングする必要があります。

立ち下がりエッジでデータをサンプリングする設定は（CPOL=1、CPHA=0）または（CPOL=0、CPHA=1）です。MOSI 転送期間で既に（CPOL=1、CPHA=0）を設定していますが、以下に示す理由により（CPOL=0、CPHA=1）に設定を変更します。

図 5に（CPOL=1、CPHA=0）のまま設定を変更しない場合のタイミングを示します。スレーブがデータを出力するRSPCKの立ち上がりエッジでマスタがデータをサンプリングするためタイミング条件を満たすことができません。

図 6に（CPOL=0、CPHA=1）に設定を変更した場合のタイミングを示します。RSPIの設定変更時にRSPCKが立ち下がるため、設定変更のタイミングでスレーブからデータが出力されます。その 1 サイクル後のRSPCKの立ち下がりエッジでマスタがデータをサンプリングするため、タイミング条件を満たすことができます。

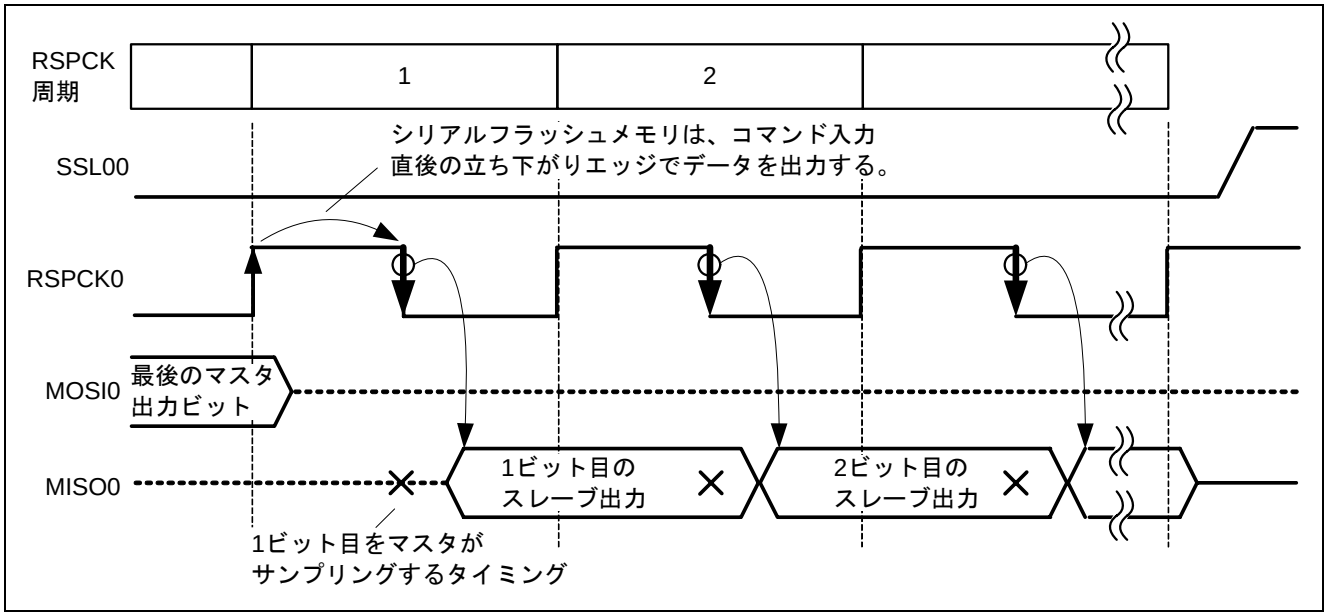


図5　MISO 転送タイミング（CPOL ビットおよび CPHA ビットを変更しない場合）

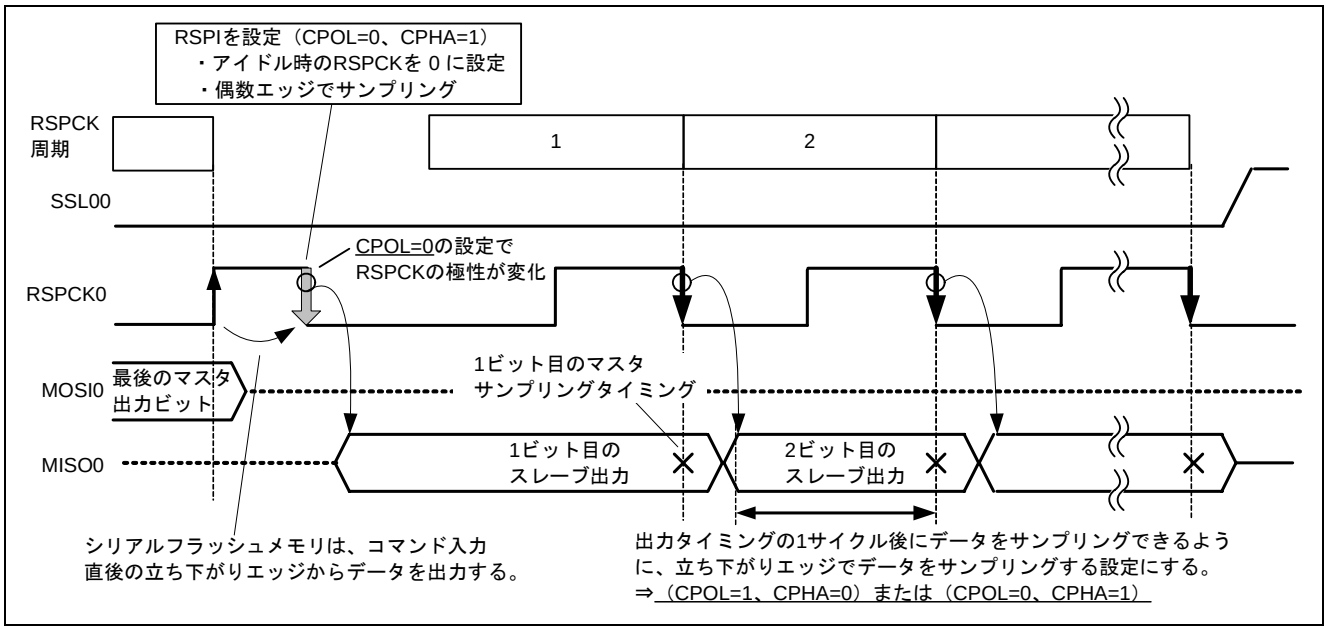


図6　MISO 転送タイミング（CPOL ビットおよび CPHA ビットを変更した場合）

図 7にセットアップ期間拡張時のインタフェースタイミングを示します。表 4および 表 5にシリアルフラッシュメモリおよびSH7264 のタイミング条件を示します。これらの条件を満たすように設定を行ってください。

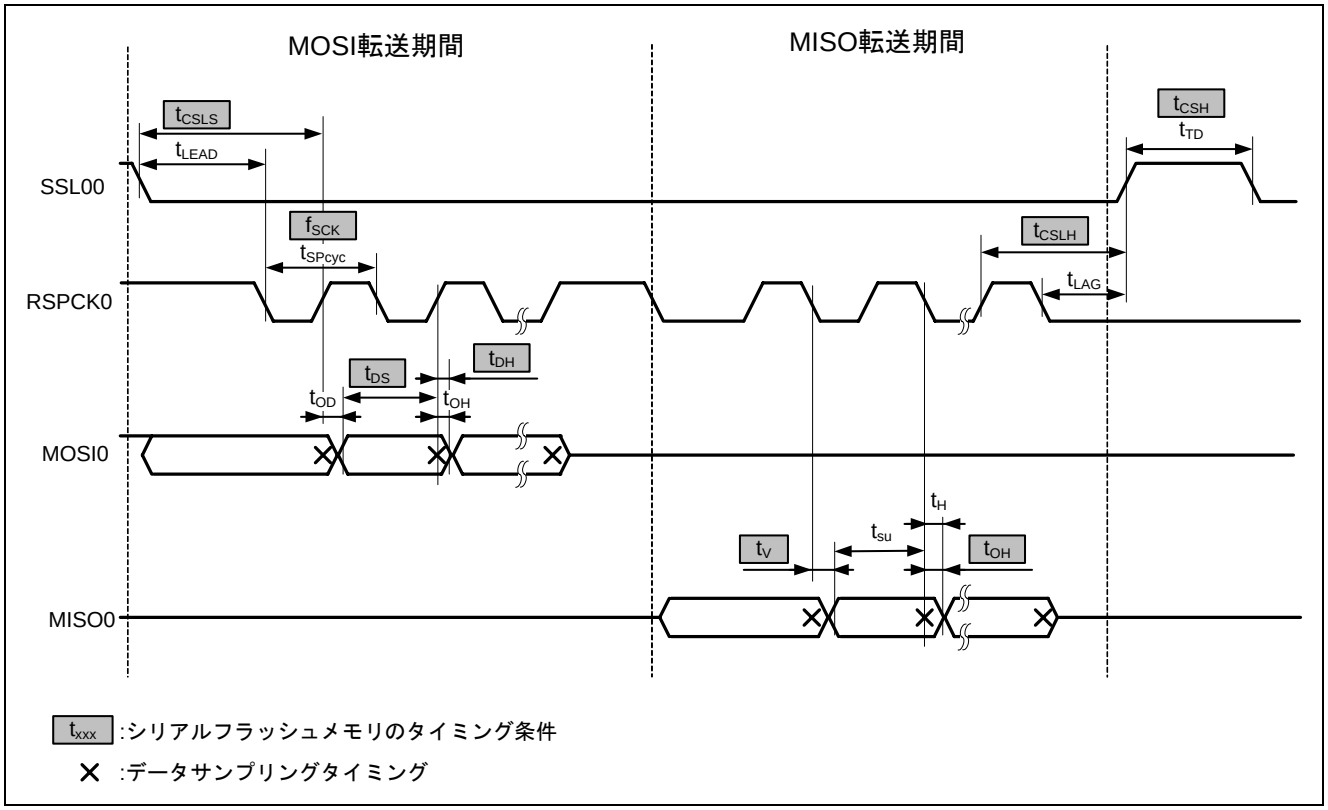


図7　セットアップ期間拡張時のインタフェースタイミング

表4　セットアップ期間拡張時の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をサンプリング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 SPBR 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)＋1/2 t_{SPcyc} > t_{CSLH}(min)	SSLND レジスタ SPBR 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リング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_{SPcyc} − t_{OD}(max) > t_{DS}(min)	SPBR レジスタ SPCMD レジスタ
t_{DH}	データ入力ホールド時間	マスタがデータ出力を保持しなければならない時間です。 以下の式を満たす設定を行います。 t_{OH}(min) > t_{DH}(min)	

表5　セットアップ期間拡張時の 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リング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_{SPcyc} − t_{V}(max) > t_{SU}(min)	SPBR レジスタ SPCMD レジスタ
t_{H}	データ入力ホールド時間	スレーブがデータ出力を保持しなければならない時間です。 以下の式を満たす設定を行います。 t_{OH}(min) > t_{H}(min)	

(2) アクセス幅の拡張

データレジスタ（SPDR）のアクセス幅をロングワード長にすると、転送前後の各種ウェイト期間（RSPCK 遅延、SSL ネゲート遅延、次アクセス遅延）の挿入回数が減るため、効率的なデータ転送が可能です。

Read Arrayコマンド（H'0Bオペコード）の場合、マスタ出力部（コマンド／アドレス／ダミーデータ）は、バイト数が 5 バイトです。そのため、マスタ出力部の転送はバイト長のアクセス幅で行い、スレーブ出力部（データ）の転送はロングワード長のアクセス幅で行います。図 8にアクセス幅を拡張したコマンドシーケンス例を示します。

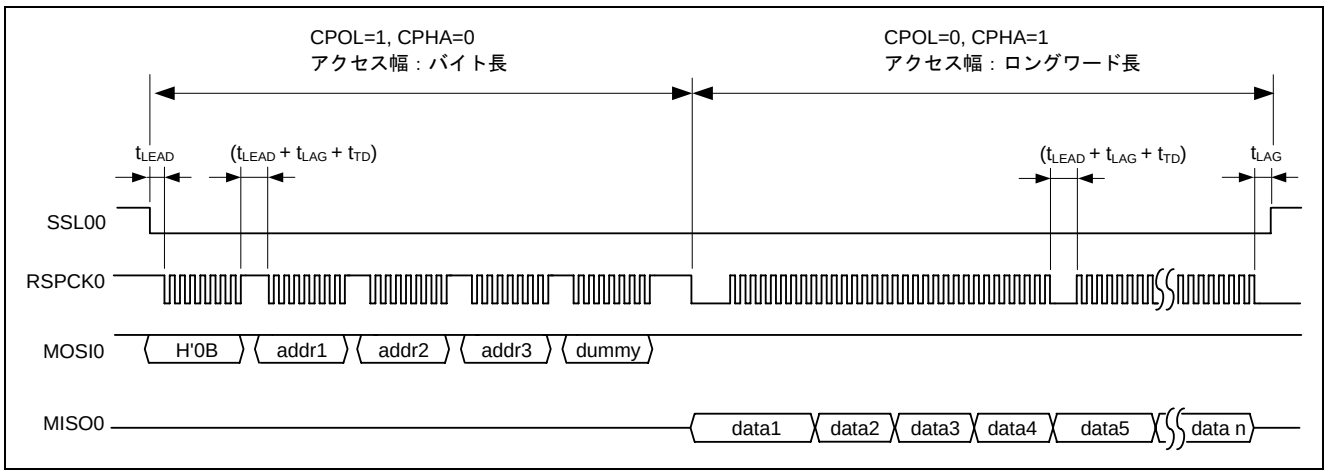


図8　アクセス幅をロングワード長にしたコマンドシーケンス（H'0B オペコード）

2.4 参考プログラムの動作

2.4.1 RSPI の初期設定例

図 9および図 10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。

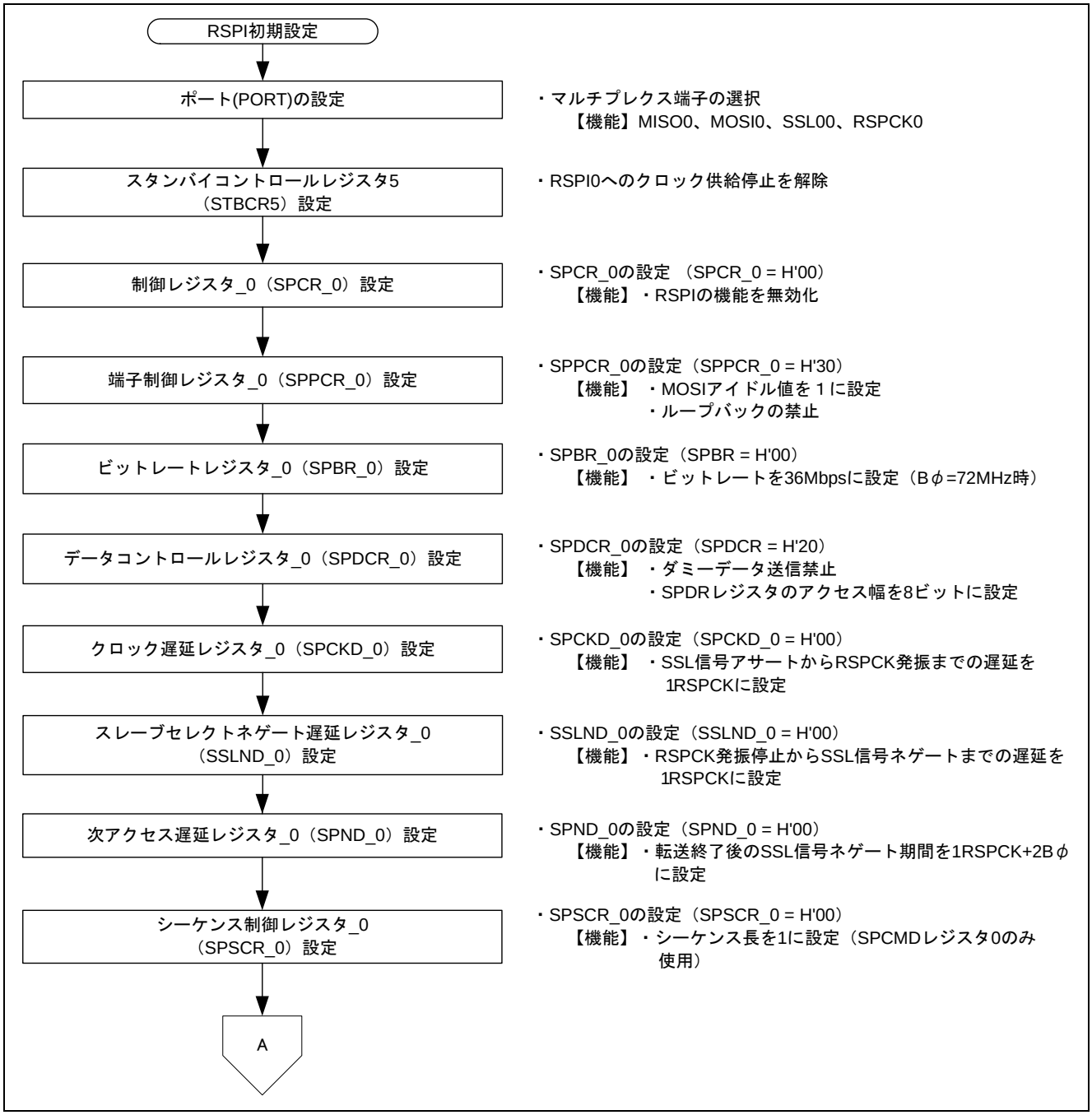


図9 参考プログラムの RSPI 初期設定フロー（1）

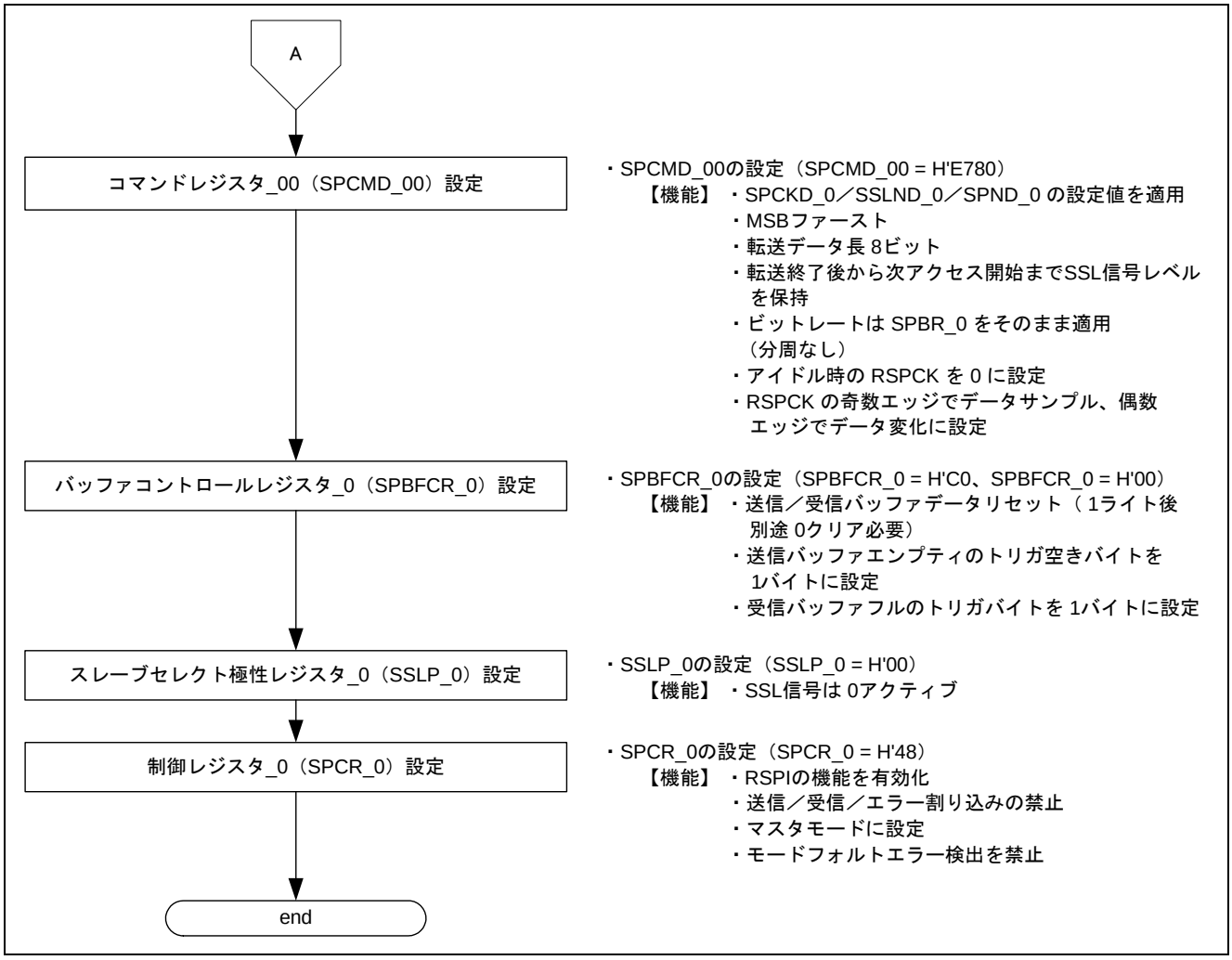


図10　参考プログラムの RSPI 初期設定フロー（2）

2.4.2 コマンド転送フロー例

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 11、図 12および図 13にリード用のコマンド転送処理フローを示します。データリード時のアクセス幅はロングワード長（32 ビット）に設定しています。またメモリへのデータ格納にはDMA転送を使用します。

図 14はライト用のコマンド転送処理フローを示します。ライト処理は、コマンド転送処理の時間よりも、ライト後のビジー解除待ち時間が長いため、ここではアクセス幅をバイト長に設定しています。

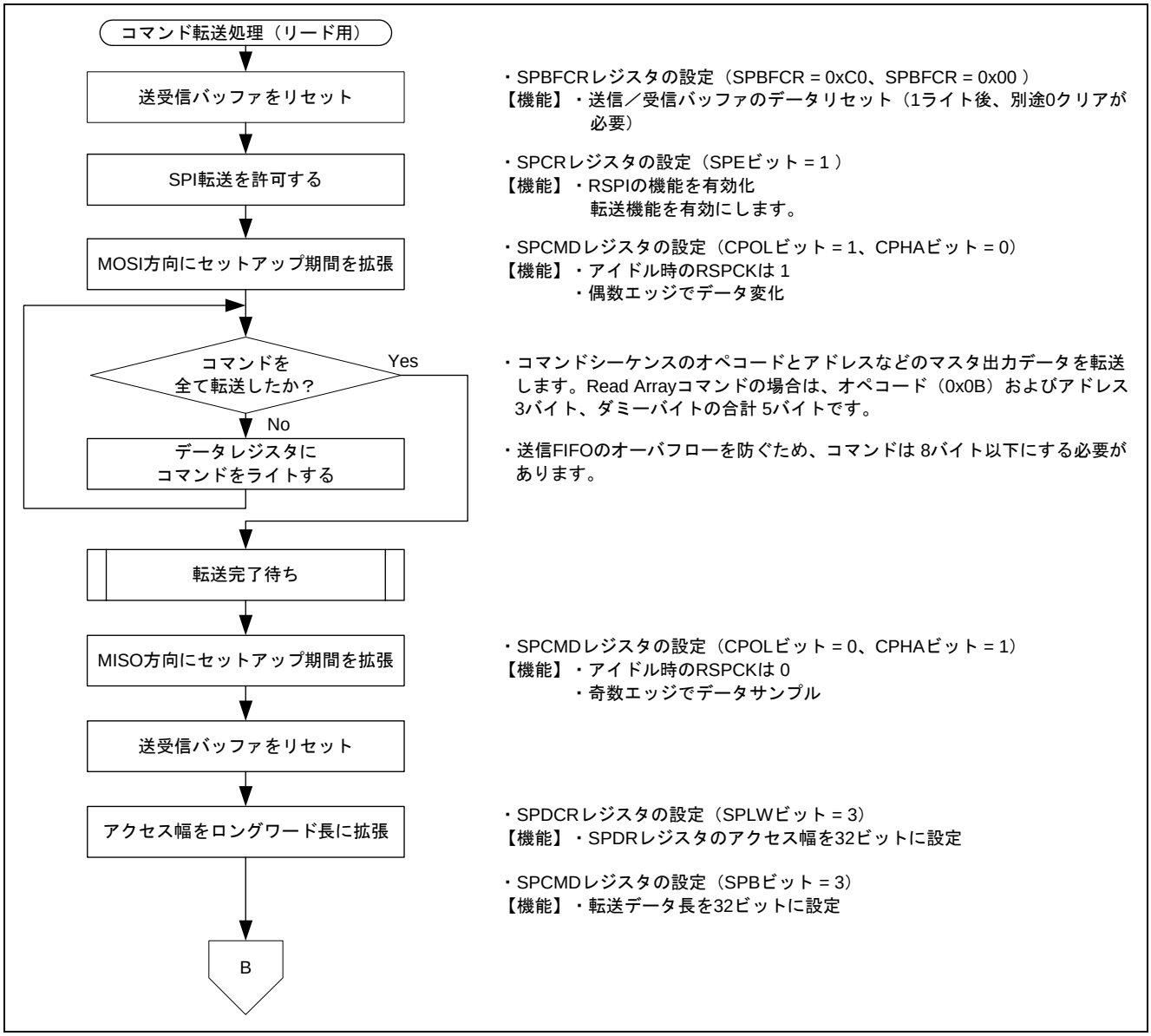


図11　参考プログラムのリード用コマンド転送フロー（1）

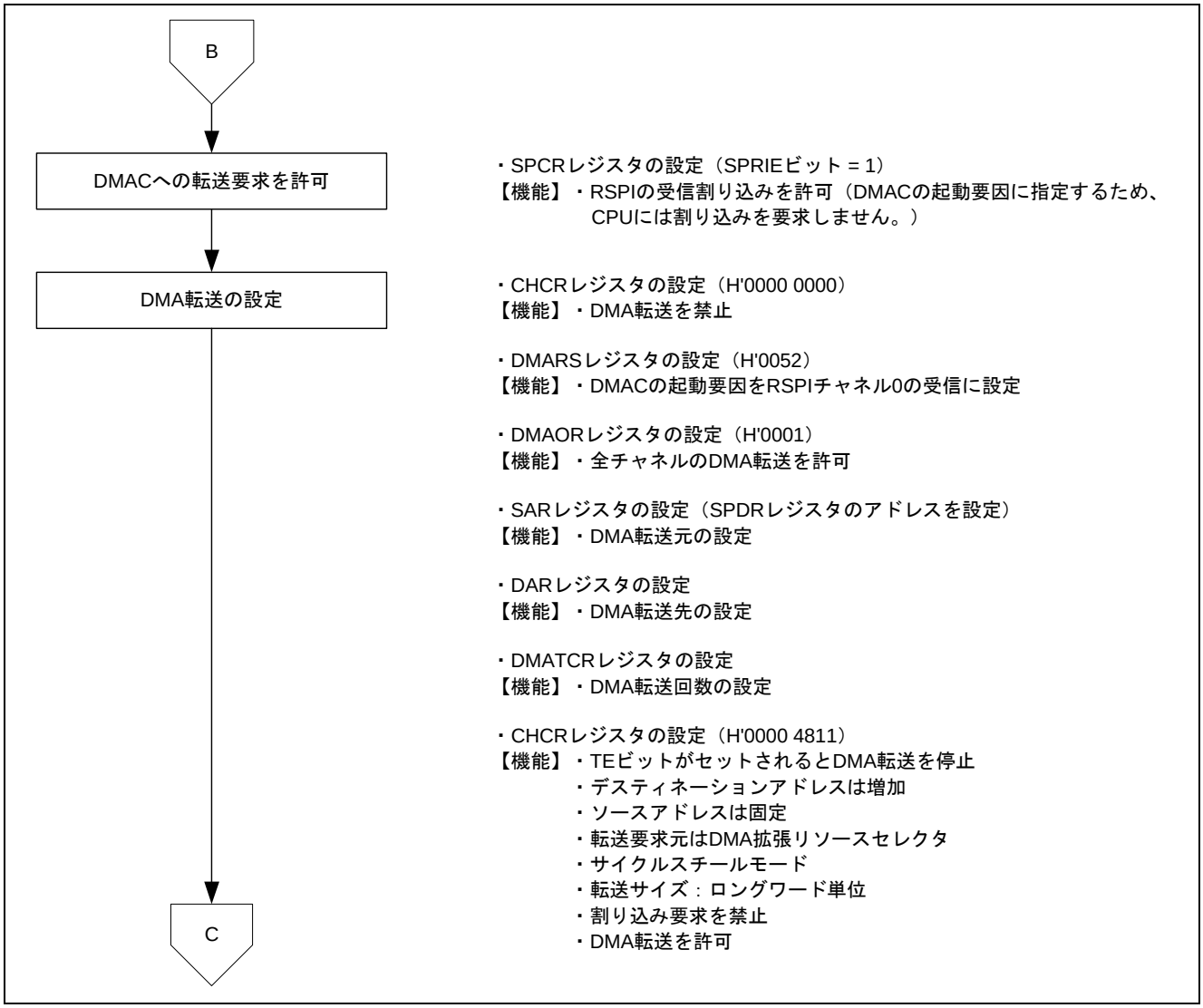
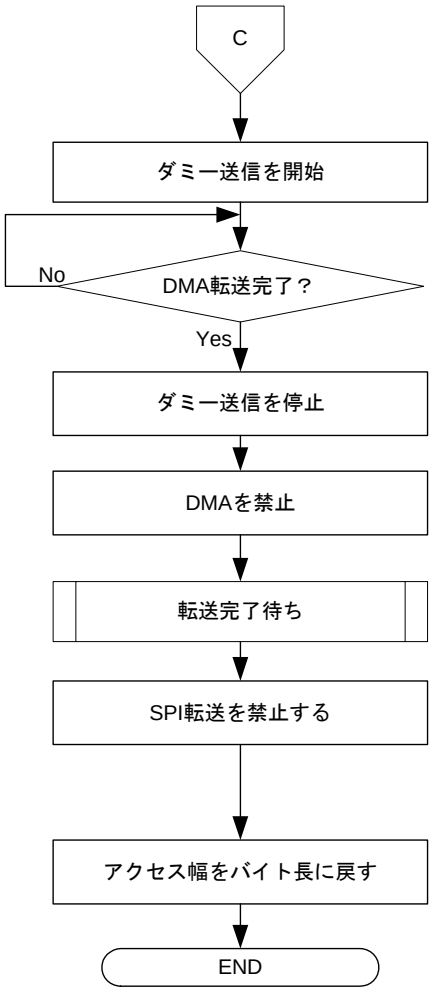


図12　参考プログラムのリード用コマンド転送フロー（2）



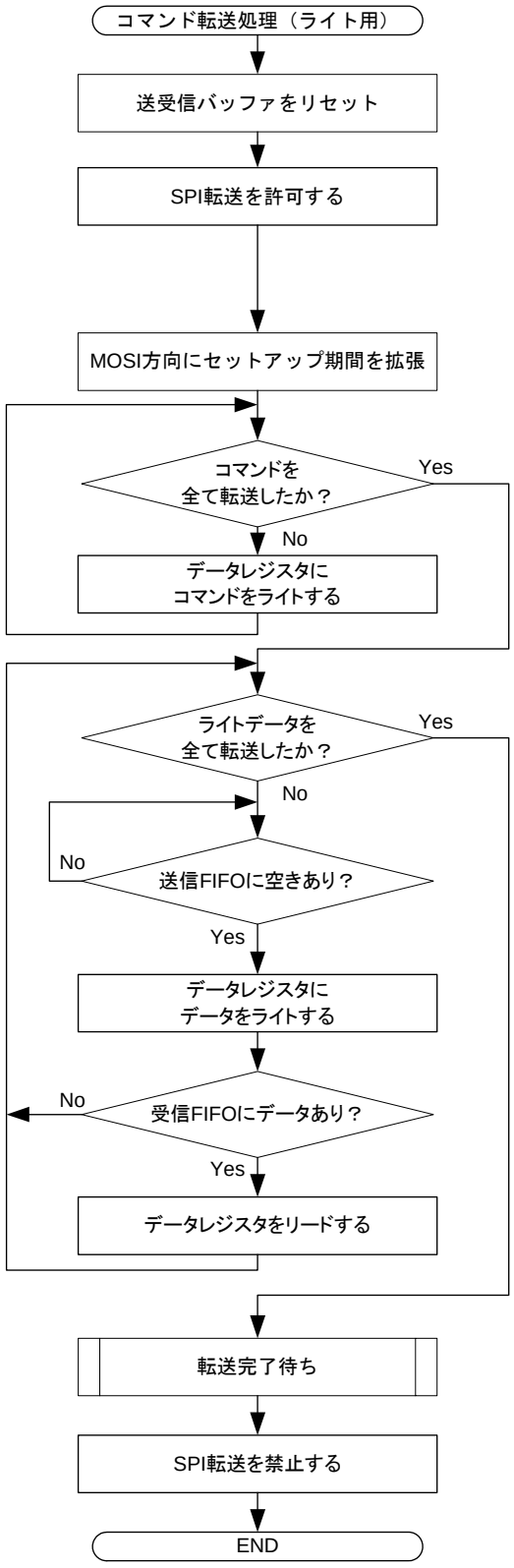
・SPDCRレジスタの設定（TXDMYビット = 1）
【機能】・ダミー送信を許可
（データレジスタにデータをライトしなくても自動的にダミーデータが送信されます。）

・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。

・SPDCRレジスタの設定（SPLWビット = 1）
【機能】・SPDRレジスタのアクセス幅を8ビットに設定
・SPCMDレジスタの設定（SPBビット = 7）
【機能】・転送データ長を8ビットに設定

図13　参考プログラムのリード用コマンド転送フロー（3）



・SPBFCRレジスタの設定（SPBFCR = H'C0、SPBFCR = H'00 ）
【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。

・SPCMDレジスタの設定（CPOLビット = 1, CPHA = 0）
【機能】・アイドル時のRSPCKは 1
・偶数エッジでデータ変化

・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がオーバフローしないようコマンドは8バイト以下にしてください。

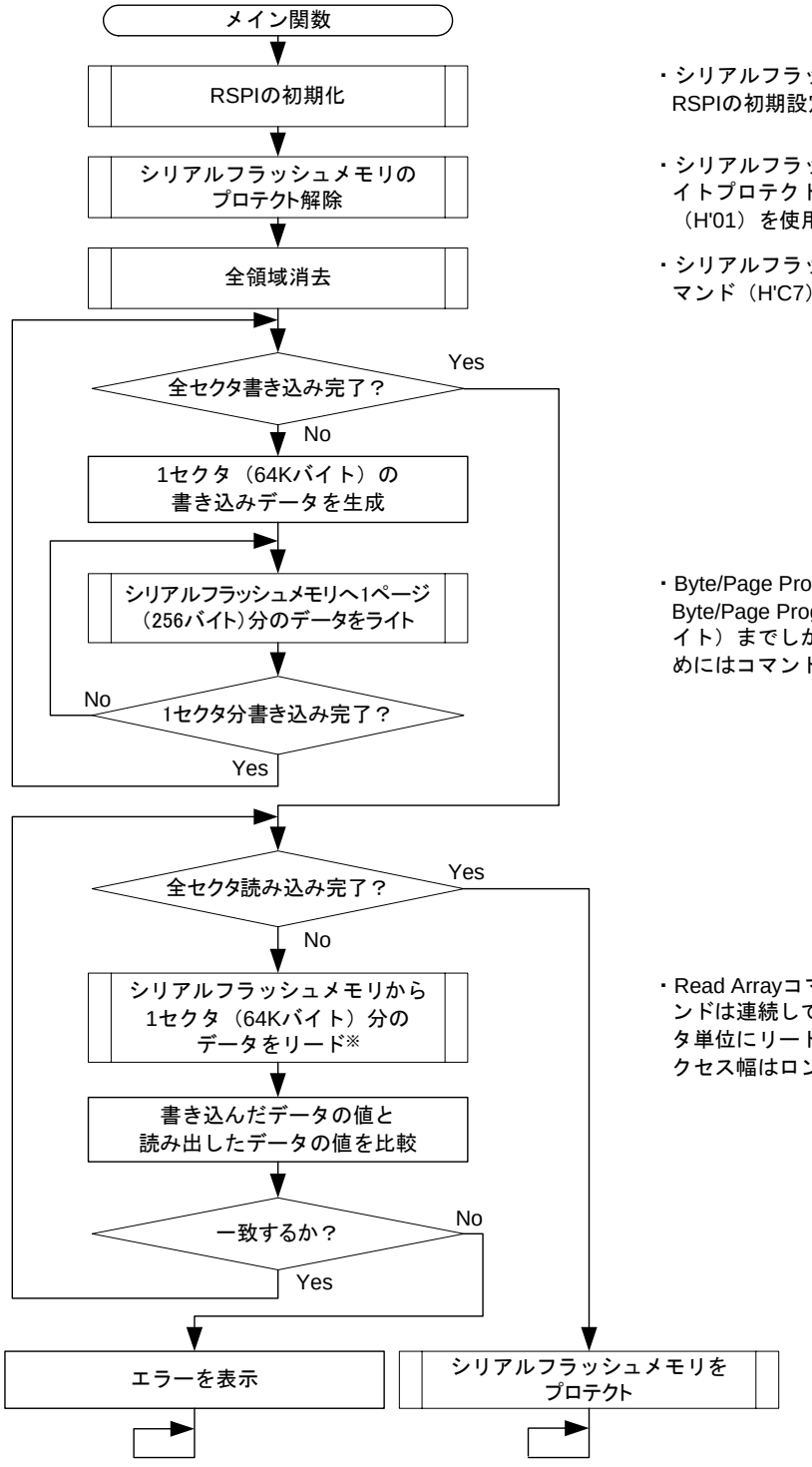
・マスタ動作時に受信FIFOがオーバフローすると転送が停止するため、ダミーリードしてください。

・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14　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 15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。



・シリアルフラッシュメモリをアクセスするために「2.4.1 RSPIの初期設定例」の手順に従いRSPIを初期化します。

・シリアルフラッシュのライトプロテクトを解除します。ライトプロテクトの解除にはWrite Status Registerコマンド（H'01）を使用します。

・シリアルフラッシュの全領域を消去します。Chip Eraseコマンド（H'C7）を使用します。

・Byte/Page Programコマンド（H'02）を実行します。Byte/Page Programコマンドは一度にページサイズ（256バイト）までしかライトできないため、全領域へ書き込むためにはコマンドを繰り返し実行します。

・Read Arrayコマンド（H'0B)を実行します。Read Arrayコマンドは連続して全領域をリードできますが、ここではセクタ単位にリードしています。データレジスタ（SPDR）のアクセス幅はロングワード長に設定します。

【注】※ DMA転送を使用する場合はキャッシュコヒーレンシの問題に注意してください。
本参考プログラムでは転送先のバッファ領域をキャッシュ無効空間に配置しています。

図15 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     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Technology Corp. and is only
5     *   intended for use with Renesas products. No other uses are authorized.
6     *
7     *   This software is owned by Renesas Technology Corp. and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ************************************************************************
28    *   Copyright (C) 2009. Renesas Technology Corp., All Rights Reserved.
29    *""FILE COMMENT""*********** Technical reference data *********************
30    *   System Name : SH7264 Sample Program
31    *   File Name   : main.c
32    *   Abstract    : ルネサスシリアルペリフェラルインタフェース
33    *               : シリアルフラッシュメモリの高速リードライト
34    *   Version     : 1.00.00
35    *   Device      : SH7262/SH7264
36    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.04.01).
37    *               : C/C++ compiler package for the SuperH RISC engine family
38    *               :                          (Ver.9.02 Release00).
39    *   OS          : None
40    *   H/W Platform: M3A-HS64G50(CPU board)
41    *   Description :
42    ************************************************************************
43    *   History     : Apr.21,2009 Ver.1.00.00
44    *""FILE COMMENT END""*****************************************************/
```

3.2 サンプルプログラムリスト"main.c"（2）

```
45    #include <stdio.h>
46    #include "serial_flash.h"
47
48    /* ==== マクロ定義 ==== */
49    #define TOP_ADDRESS    0              /* シリアルフラッシュメモリの先頭アドレス */
50
51    /* ==== 関数プロトタイプ宣言 ==== */
52    void main(void);
53
54    /* ==== 変数定義 ==== */
55    #pragma section LARGE_ONCHIP_RAM
56    static unsigned char data[SF_SECTOR_SIZE];
57    static unsigned long rbuf[SF_SECTOR_SIZE/sizeof(long)];
58    #pragma section
59
60    /*""FUNC COMMENT""*****************************************************************
61     * ID          :
62     * Outline     : シリアルフラッシュメモリアクセス メイン処理
63     *-------------------------------------------------------------------------------
64     * Include     : "serial_flash.h"
65     *-------------------------------------------------------------------------------
66     * Declaration  : void main(void);
67     *-------------------------------------------------------------------------------
68     * Description  : 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
69     *              : 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
70     *              : 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。
71     *-------------------------------------------------------------------------------
72     * Argument    : void
73     *-------------------------------------------------------------------------------
74     * Return Value : void
75     *-------------------------------------------------------------------------------
76     * Note        : None
77     *""FUNC COMMENT END""*************************************************************/
78    void main(void)
79    {
80      int i, j;
81      unsigned char *p;
82      static unsigned long addr;
83
84      /* ==== RSPI の初期化 ==== */
85      sf_init_serial_flash();
86
87      /* ==== シリアルフラッシュメモリのプロテクト解除 ==== */
88      sf_protect_ctrl( SF_REQ_UNPROTECT );
89
90      /* ==== チップイレース（2MB、完了までに約 10 秒かかります） ==== */
91      sf_chip_erase();
92
```

3.3 サンプルプログラムリスト"main.c"（3）

```
93      /* ==== データライト（2MB、完了までに約 10 秒かかります） ==== */
94      addr = TOP_ADDRESS;
95      for(i = 0; i < SF_NUM_OF_SECTOR; i++){
96        /* ---- データ初期化（64KB） ---- */
97        for(j = 0; j < SF_SECTOR_SIZE; j++){
98            data[j] = (i + j) % 100;
99        }
100       /* ---- セクタサイズ（64KB）をライト ---- */
101       for(j = 0; j < ( SF_SECTOR_SIZE / SF_PAGE_SIZE ); j++){
102           /* ---- ページサイズ（256B）をライト ---- */
103           sf_byte_program( addr, data+(j*SF_PAGE_SIZE), SF_PAGE_SIZE );
104           addr += SF_PAGE_SIZE;                  /* 書き込み先アドレス更新 */
105       }
106     }
107     /* ==== データリード（2MB） ==== */
108     addr = TOP_ADDRESS;
109     for(i = 0; i < SF_NUM_OF_SECTOR; i++){
110
111       /* ---- セクタサイズ（64KB）をリード ---- */
112       sf_byte_read_long( addr, rbuf, SF_SECTOR_SIZE );
113       addr += SF_SECTOR_SIZE;                    /* 読み込み先アドレス更新 */
114
115       /* ---- ベリファイチェック ---- */
116       p = (unsigned char *)rbuf;
117       for(j = 0; j < SF_SECTOR_SIZE; j++){
118           data[j] = (i + j) % 100;               /* 書き込んだデータを再生 */
119           if( data[j] != *(p+j) ){
120               puts("Error: verify error¥n");
121               fflush(stdout);
122               while(1);
123           }
124       }
125     }
126     /* ==== シリアルフラッシュメモリのプロテクト ==== */
127     sf_protect_ctrl( SF_REQ_PROTECT );
128
129     while(1){
130         /* loop */
131      }
132   }
133
134   /* End of File */
135
136
```

3.4 サンプルプログラムリスト"serial_flash.c"（1）

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Technology Corp. and is only
5     *   intended for use with Renesas products. No other uses are authorized.
6     *
7     *   This software is owned by Renesas Technology Corp. and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ************************************************************************
28    *   Copyright (C) 2009. Renesas Technology Corp., All Rights Reserved.
29    *""FILE COMMENT""*********** Technical reference data *********************
30    *   System Name : SH7264 Sample Program
31    *   File Name   : serial_flash.c
32    *   Abstract    : ルネサスシリアルペリフェラルインタフェース
33    *               : シリアルフラッシュメモリの高速リードライト
34    *   Version     : 1.00.00
35    *   Device      : SH7262/SH7264
36    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.04.01).
37    *               : C/C++ compiler package for the SuperH RISC engine family
38    *               :                          (Ver.9.02 Release00).
39    *   OS          : None
40    *   H/W Platform: M3A-HS64G50(CPU board)
41    *   Description :
42    ************************************************************************
43    *   History     : Mar.09,2009 Ver.1.00.00
44    *""FILE COMMENT END""*****************************************************/
45    #include <stdio.h>
46    #include <machine.h>
47    #include "iodefine.h"
48    #include "serial_flash.h"
```

3.5 サンプルプログラムリスト"serial_flash.c"（2）

```
49    /* ==== マクロ定義 ==== */
50    #define SFLASHCMD_CHIP_ERASE       0xc7
51    #define SFLASHCMD_SECTOR_ERASE     0xd8
52    #define SFLASHCMD_BYTE_PROGRAM     0x02
53    #define SFLASHCMD_BYTE_READ        0x0B
54    #define SFLASHCMD_BYTE_READ_LOW    0x03
55    #define SFLASHCMD_WRITE_ENABLE     0x06
56    #define SFLASHCMD_WRITE_DISABLE    0x04
57    #define SFLASHCMD_READ_STATUS      0x05
58    #define SFLASHCMD_WRITE_STATUS     0x01
59    #define UNPROTECT_WR_STATUS        0x00
60    #define PROTECT_WR_STATUS          0x3C
61
62    #define SF_USE_DMAC           /* sf_byte_read_long 関数を DMA 転送で実装する場合に定義する */
63
64    /* ==== 関数プロトタイプ宣言 ==== */
65    /*** Local function ***/
66    static void write_enable(void);
67    static void write_disable(void);
68    static void busy_wait(void);
69    static unsigned char read_status(void);
70    static void write_status(unsigned char status);
71    static void io_init_rspi(void);
72    static void io_cmd_exe(unsigned char *ope, int ope_sz, unsigned char *data, int data_sz);
73    static void io_cmd_exe_rdmode(unsigned char *ope, int ope_sz, unsigned char *rd, int rd_sz);
74    static void io_cmd_exe_rdmode_cpu_l(unsigned char *ope, int ope_sz, unsigned long *rd, int rd_sz);
75    static void io_cmd_exe_rdmode_dma_l(unsigned char *ope, int ope_sz, unsigned long *rd, int rd_sz);
76    static void io_wait_tx_end(void);
77
78    /*""FUNC COMMENT""*****************************************************************
79     * ID          :
80     * Outline     : シリアルフラッシュメモリの初期化
81     *-------------------------------------------------------------------------------
82     * Include     :
83     *-------------------------------------------------------------------------------
84     * Declaration  : void sf_init_serial_flash(void);
85     *-------------------------------------------------------------------------------
86     * Description  : 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。
87     *              : 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
88     *              : します。
89     *-------------------------------------------------------------------------------
90     * Argument    : void
91     *-------------------------------------------------------------------------------
92     * Return Value : void
93     *-------------------------------------------------------------------------------
94     * Note        : None
95     *""FUNC COMMENT END""*************************************************************/
96    void sf_init_serial_flash(void)
97    {
98      /* ==== RSPI0 の初期化 ==== */
99      io_init_rspi();
100   }
```

3.6 サンプルプログラムリスト"serial_flash.c"（3）

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : プロテクト操作
 *-----------------------------------------------------------------------------
 * Include     : "serial_flash.h"
 *-----------------------------------------------------------------------------
 * Declaration  : void sf_init_serial_flash(void);
 *-----------------------------------------------------------------------------
 * Description  : 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。
 *              : 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
 *              : シリアルフラッシュメモリの仕様によって異なります。
 *-----------------------------------------------------------------------------
 * Argument    : enum sf_req req ; I : SF_REQ_UNPROTECT -> 全セクタプロテクト解除
 *             :                     SF_REQ_PROTECT   -> 全セクタプロテクト
 *-----------------------------------------------------------------------------
 * Return Value : void
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
void sf_protect_ctrl(enum sf_req req)
{
  if( req == SF_REQ_UNPROTECT ){
    write_status( UNPROTECT_WR_STATUS);       /* 全領域プロテクト解除 */
  }
  else{
    write_status( PROTECT_WR_STATUS );        /* 全領域プロテクト */
  }
}
```

3.7 サンプルプログラムリスト"serial_flash.c"（4）

```
129   /*""FUNC COMMENT""*****************************************************************
130    * ID          :
131    * Outline     : チップイレース
132    *-------------------------------------------------------------------------------
133    * Include     :
134    *-------------------------------------------------------------------------------
135    * Declaration  : void sf_chip_erase(void);
136    *-------------------------------------------------------------------------------
137    * Description  : 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。
138    *              : 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
139    *              : 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
140    *              : 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
141    *              : されたことを確認してください。
142    *-------------------------------------------------------------------------------
143    * Argument    : void
144    *-------------------------------------------------------------------------------
145    * Return Value : void
146    *-------------------------------------------------------------------------------
147    * Note        : None
148    *""FUNC COMMENT END""*************************************************************/
149   void sf_chip_erase(void)
150   {
151     unsigned char cmd[1];
152     cmd[0] = SFLASHCMD_CHIP_ERASE;
153
154     write_enable();
155     io_cmd_exe(cmd, 1, NULL, 0);
156     busy_wait();
157   }
```

3.8 サンプルプログラムリスト"serial_flash.c"（5）

```
158  /*""FUNC COMMENT""*****************************************************************
159   * ID          :
160   * Outline     : セクタイレース
161   *-------------------------------------------------------------------------------
162   * Include     :
163   *-------------------------------------------------------------------------------
164   * Declaration  : void sf_sector_erase(void);
165   *-------------------------------------------------------------------------------
166   * Description  : 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。
167   *              : 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
168   *              : 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
169   *              : 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
170   *              : されたことを確認してください。
171   *-------------------------------------------------------------------------------
172   * Argument    : int sector_no ; I : セクタ番号
173   *-------------------------------------------------------------------------------
174   * Return Value : void
175   *-------------------------------------------------------------------------------
176   * Note        : None
177   *""FUNC COMMENT END""*************************************************************/
178  void sf_sector_erase(int sector_no)
179  {
180    unsigned char cmd[4];
181    unsigned long addr = sector_no * SF_SECTOR_SIZE;
182
183    cmd[0] = SFLASHCMD_SECTOR_ERASE;
184    cmd[1] = (addr >> 16) & 0xff;
185    cmd[2] = (addr >>  8) & 0xff;
186    cmd[3] =  addr       & 0xff;
187
188    write_enable();
189    io_cmd_exe(cmd, 4, NULL, 0);
190    busy_wait();
191  }
```

3.9 サンプルプログラムリスト"serial_flash.c"（6）

```
192   /*""FUNC COMMENT""*****************************************************************
193    * ID          :
194    * Outline     : データプログラム
195    *-------------------------------------------------------------------------------
196    * Include     :
197    *-------------------------------------------------------------------------------
198    * Declaration  : void sf_byte_program(unsigned long addr, unsigned char *buf, int size);
199    *-------------------------------------------------------------------------------
200    * Description  : 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。
201    *              : 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
202    *              : 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
203    *              : 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
204    *              : されたことを確認してください。
205    *              : 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。
206    *-------------------------------------------------------------------------------
207    * 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
208    *             : unsigned char *buf ; I : ライトデータを格納するバッファのアドレス
209    *             : int size         ; I : ライトするバイト数
210    *-------------------------------------------------------------------------------
211    * Return Value : void
212    *-------------------------------------------------------------------------------
213    * Note       : None
214    *""FUNC COMMENT END""*************************************************************/
215   void sf_byte_program(unsigned long addr, unsigned char *buf, int size)
216   {
217     unsigned char cmd[4];
218
219     cmd[0] = SFLASHCMD_BYTE_PROGRAM;
220     cmd[1] = (unsigned char)((addr >> 16) & 0xff);
221     cmd[2] = (unsigned char)((addr >>  8) & 0xff);
222     cmd[3] = (unsigned char)( addr       & 0xff);
223     write_enable();
224     io_cmd_exe(cmd, 4, buf, size);
225     busy_wait();
226   }
```

3.10 サンプルプログラムリスト"serial_flash.c"（7）

```
227  /*""FUNC COMMENT""*****************************************************************
228   * ID          :
229   * Outline     : データリード（バイト転送版）
230   *-------------------------------------------------------------------------------
231   * Include     :
232   *-------------------------------------------------------------------------------
233   * Declaration  : void sf_byte_read(unsigned long addr, unsigned char *buf, int size);
234   *-------------------------------------------------------------------------------
235   * Description  : 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。
236   *-------------------------------------------------------------------------------
237   * 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
238   *             : unsigned char *buf ; I : リードデータを格納するバッファのアドレス
239   *             : int size          ; I : リードするバイト数
240   *-------------------------------------------------------------------------------
241   * Return Value : void
242   *-------------------------------------------------------------------------------
243   * Note        : None
244   *""FUNC COMMENT END""*************************************************************/
245  void sf_byte_read(unsigned long addr, unsigned char *buf, int size)
246  {
247    unsigned char cmd[5];
248
249    cmd[0] = SFLASHCMD_BYTE_READ;
250    cmd[1] = (unsigned char)((addr >> 16) & 0xff);
251    cmd[2] = (unsigned char)((addr >>  8) & 0xff);
252    cmd[3] = (unsigned char)( addr       & 0xff);
253    cmd[4] = 0x00;
254    io_cmd_exe_rdmode(cmd, 5, buf, size);
255  }
```

3.11 サンプルプログラムリスト"serial_flash.c"（8）

```
256   /*""FUNC COMMENT""*****************************************************************
257    * ID          :
258    * Outline     : データリード（ロングワード転送版）
259    *-------------------------------------------------------------------------------
260    * Include     :
261    *-------------------------------------------------------------------------------
262    * Declaration  : void sf_byte_read_long(unsigned long addr, unsigned long *buf, int size);
263    *-------------------------------------------------------------------------------
264    * Description  : シリアルフラッシュメモリを指定バイト数だけロングワード幅でリードします。
265    *-------------------------------------------------------------------------------
266    * 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
267    *             : unsigned long *buf ; I : リードデータを格納するバッファのアドレス
268    *             : int size         ; I : リードするバイト数
269    *-------------------------------------------------------------------------------
270    * Return Value : void
271    *-------------------------------------------------------------------------------
272    * Note        : None
273    *""FUNC COMMENT END""*************************************************************/
274   void sf_byte_read_long(unsigned long addr, unsigned long *buf, int size)
275   {
276     unsigned char cmd[5];
277
278     cmd[0] = SFLASHCMD_BYTE_READ;
279     cmd[1] = (unsigned char)((addr >> 16) & 0xff);
280     cmd[2] = (unsigned char)((addr >>  8) & 0xff);
281     cmd[3] = (unsigned char)( addr       & 0xff);
282     cmd[4]   = 0x00;
283   #ifdef SF_USE_DMAC
284     io_cmd_exe_rdmode_dma_l(cmd, 5, buf, size);
285   #else
286     io_cmd_exe_rdmode_cpu_l(cmd, 5, buf, size);
287   #endif
288   }
```

3.12 サンプルプログラムリスト"serial_flash.c"（9）

```
289   /*""FUNC COMMENT""*****************************************************************
290    * ID          :
291    * Outline     : 書き込み許可
292    *-------------------------------------------------------------------------------
293    * Include     :
294    *-------------------------------------------------------------------------------
295    * Declaration  : static void write_enable(void);
296    *-------------------------------------------------------------------------------
297    * Description  : 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
298    *              : イレースまたはプログラム動作を許可します。
299    *-------------------------------------------------------------------------------
300    * Argument    : void
301    *-------------------------------------------------------------------------------
302    * Return Value : void
303    *-------------------------------------------------------------------------------
304    * Note        : None
305    *""FUNC COMMENT END""*************************************************************/
306   static void write_enable(void)
307   {
308     unsigned char cmd[1];
309     cmd[0] = SFLASHCMD_WRITE_ENABLE;
310     io_cmd_exe(cmd, 1, NULL, 0);
311   }
312   /*""FUNC COMMENT""*****************************************************************
313    * ID          :
314    * Outline     : 書き込み禁止
315    *-------------------------------------------------------------------------------
316    * Include     :
317    *-------------------------------------------------------------------------------
318    * Declaration  : static void write_disable(void);
319    *-------------------------------------------------------------------------------
320    * Description  : 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
321    *              : イレースまたはプログラム動作を禁止します。
322    *-------------------------------------------------------------------------------
323    * Argument    : void
324    *-------------------------------------------------------------------------------
325    * Return Value : void
326    *-------------------------------------------------------------------------------
327    * Note        : None
328    *""FUNC COMMENT END""*************************************************************/
329   static void write_disable(void)
330   {
331     unsigned char cmd[1];
332     cmd[0] = SFLASHCMD_WRITE_DISABLE;
333     io_cmd_exe(cmd, 1, NULL, 0);
334   }
```

3.13 サンプルプログラムリスト"serial_flash.c"（10）

```
335   /*""FUNC COMMENT""*****************************************************************
336    * ID          :
337    * Outline     : ビジー待ち
338    *-------------------------------------------------------------------------------
339    * Include     :
340    *-------------------------------------------------------------------------------
341    * Declaration  : static void busy_wait(void);
342    *-------------------------------------------------------------------------------
343    * Description  : 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
344    *              : ループします。
345    *-------------------------------------------------------------------------------
346    * Argument    : void
347    *-------------------------------------------------------------------------------
348    * Return Value : void
349    *-------------------------------------------------------------------------------
350    * Note         : None
351    *""FUNC COMMENT END""*************************************************************/
352   static void busy_wait(void)
353   {
354     while ((read_status() & 0x01) != 0) {   /* RDY/BSY */
355       /* serial flash is busy */
356     }
357   }
358   /*""FUNC COMMENT""*****************************************************************
359    * ID          :
360    * Outline     : ステータスリード
361    *-------------------------------------------------------------------------------
362    * Include     :
363    *-------------------------------------------------------------------------------
364    * Declaration  : static unsigned char read_status(void);
365    *-------------------------------------------------------------------------------
366    * Description  : 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。
367    *-------------------------------------------------------------------------------
368    * Argument    : void
369    *-------------------------------------------------------------------------------
370    * Return Value : ステータスレジスタの値
371    *-------------------------------------------------------------------------------
372    * Note        : None
373    *""FUNC COMMENT END""*************************************************************/
374   static unsigned char read_status(void)
375   {
376     unsigned char buf;
377     unsigned char cmd[1];
378
379     cmd[0] = SFLASHCMD_READ_STATUS;
380     io_cmd_exe_rdmode(cmd, 1, &buf, 1);
381     return buf;
382   }
```

3.14 サンプルプログラムリスト"serial_flash.c"（11）

```
383   /*""FUNC COMMENT""*****************************************************************
384    * ID          :
385    * Outline     : ステータスライト
386    *-------------------------------------------------------------------------------
387    * Include     :
388    *-------------------------------------------------------------------------------
389    * Declaration  : static void write_status(unsigned char status);
390    *-------------------------------------------------------------------------------
391    * Description  : 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。
392    *-------------------------------------------------------------------------------
393    * Argument    : unsigned char status ; I : status register value
394    *-------------------------------------------------------------------------------
395    * Return Value : void
396    *-------------------------------------------------------------------------------
397    * Note        : None
398    *""FUNC COMMENT END""*************************************************************/
399   static void write_status(unsigned char status)
400   {
401     unsigned char cmd[2];
402
403     cmd[0] = SFLASHCMD_WRITE_STATUS;
404     cmd[1] = status;
405
406     write_enable();
407     io_cmd_exe(cmd, 2, NULL, 0);
408     busy_wait();
409   }
410   /*""FUNC COMMENT""*****************************************************************
411    * ID          :
412    * Outline     : RSPI の初期化
413    *-------------------------------------------------------------------------------
414    * Include     :
415    *-------------------------------------------------------------------------------
416    * Declaration  : static void io_init_rspi(void);
417    *-------------------------------------------------------------------------------
418    * Description  : 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。
419    *              : 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
420    *              : 転送設定を行います。
421    *-------------------------------------------------------------------------------
422    * Argument    : void
423    *-------------------------------------------------------------------------------
424    * Return Value : void
425    *-------------------------------------------------------------------------------
426    * Note        : None
427    *""FUNC COMMENT END""*************************************************************/
```

3.15 サンプルプログラムリスト"serial_flash.c"（12）

```
428   static void io_init_rspi(void)
429   {
430     /* ==== PORT ==== */
431     PORT.PFCR3.BIT.PF12MD = 3;  /* PF12:MISO0 */
432     PORT.PFCR2.BIT.PF11MD = 3;  /* PF11:MOSI0 */
433     PORT.PFCR2.BIT.PF10MD = 3;  /* PF10:SSL00 */
434     PORT.PFCR2.BIT.PF9MD  = 3;  /* PF9:RSPCK0 */
435
436     /* ==== CPG ==== */
437     CPG.STBCR5.BIT.MSTP51 = 0;  /* RSPI0 active */
438
439     /* ==== RSPI ==== */
440     RSPI0.SPCR.BYTE   =  0x00;  /* RSPI チャネル 0 を動作禁止 */
441     RSPI0.SPPCR.BYTE  =  0x30;  /* MOSI アイドル固定値 = 1 */
442     RSPI0.SPBR.BYTE   = 0x00;   /* ベースのビットレートを 36MHz に設定(BΦ=72MHz) */
443     RSPI0.SPDCR.BYTE  =  0x20;  /* ダミーデータ送信禁止 */
444                                 /* SPDR レジスタのアクセス幅：8 ビット */
445     RSPI0.SPCKD.BYTE  =  0x00;  /* RSPCK 遅延：1 RSPCK */
446     RSPI0.SSLND.BYTE  =  0x00;  /* SSL ネゲート遅延：1 RSPCK */
447     RSPI0.SPND.BYTE   =  0x00;  /* 次アクセス遅延：1 RSPCK + 2 BΦ */
448     RSPI0.SPSCR.BYTE  =  0x00;  /* シーケンス長：1 (SPCMD0 のみ使用) */
449     RSPI0.SPCMD0.WORD = 0xE780; /* MSB ファースト */
450                                 /* データ長：8bit */
451                                 /* 転送終了後も SSL 信号レベルを保持する */
452                                 /* ビットレート：ベースビットレートの分周なし */
453                                 /* アイドル時の RSPCK：0 */
454                                 /* 奇数エッジでデータサンプル、偶数エッジでデータ変化 */
455     RSPI0.SPBFCR.BYTE =  0xC0;  /* 送受信バッファのデータリセット許可 */
456     RSPI0.SPBFCR.BYTE =  0x00;  /* 送受信バッファのデータリセット禁止 */
457                                 /* 送信バッファのトリガ：1 バイト以上の空き */
458                                 /* 受信バッファのトリガ：1 バイト以上の受信 */
459     RSPI0.SSLP.BYTE   =  0x00;  /* SSLP = b'0 SSL signal 0-active */
460     RSPI0.SPCR.BYTE   =  0x48;  /* マスタモード */
461                                 /* 割り込み禁止 */
462                                 /* RSPI チャネル 0 の動作許可 */
463   }
```

3.16　サンプルプログラムリスト"serial_flash.c"（13）

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : コマンド実行(データリードなし)
 *-----------------------------------------------------------------------------
 * Include     :
 *-----------------------------------------------------------------------------
 * Declaration  : static void io_cmd_exe(unsigned char *ope, int ope_sz,
 *              :                        unsigned char *data,int data_sz)
 *-----------------------------------------------------------------------------
 * Description  : 指定されたコマンドを実行します。
 *              : 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。
 *              : ope_sz は 0～8 のいずれかの値を設定してください。
 *              : data_sz は 0～256 のいずれかの値を設定してください。
 *-----------------------------------------------------------------------------
 * 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
 *             : int ope_sz         ; I : オペコード部とアドレス部のバイト数
 *             : unsigned char *data; I : 送信するデータ部の先頭アドレス
 *             : int data_sz        ; I : データ部のバイト数
 *-----------------------------------------------------------------------------
 * Return Value : void
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
```

3.17 サンプルプログラムリスト"serial_flash.c"（14）

```
487   static void io_cmd_exe(unsigned char *ope, int ope_sz, unsigned char *data, int data_sz)
488   {
489     unsigned char tmp;
490
491     /* ==== バッファリセット ==== */
492     RSPI0.SPBFCR.BYTE  = 0xC0u;
493     RSPI0.SPBFCR.BYTE  = 0x00u;
494
495     /* ---- SPI 転送許可 ---- */
496     RSPI0.SPCR.BIT.SPE = 1;
497
498     /* ==== MOSI(コマンド、アドレス、ライトデータ) ==== */
499     RSPI0.SPCMD0.BIT.CPOL= 1;       /* アイドル時の RSPCK は 1 */
500     RSPI0.SPCMD0.BIT.CPHA= 0;       /* 偶数エッジ（立ち上がり）でデータ変化 */
501
502     while(ope_sz--){
503       RSPI0.SPDR.BYTE = *ope++;     /* コマンドは 8 バイト未満とする */
504     }
505     while(data_sz--){
506       while( RSPI0.SPSR.BIT.SPTEF == 0 ){
507           /* wait */
508       }
509       RSPI0.SPDR.BYTE = *data++;
510       if( RSPI0.SPSR.BIT.SPRF == 1 ){
511           tmp = RSPI0.SPDR.BYTE;     /* オーバフロー防止のためのダミーリード */
512       }
513     }
514     io_wait_tx_end();                /* 完了待ち */
515
516     /* ---- SPI 転送終了（SSL ネゲート） ---- */
517     RSPI0.SPCR.BIT.SPE = 0;
518   }
```

3.18 サンプルプログラムリスト"serial_flash.c"（15）

```
519   /*""FUNC COMMENT""*****************************************************************
520    * ID          :
521    * Outline     : コマンド実行(リードデータあり、バイト転送)
522    *-------------------------------------------------------------------------------
523    * Include     :
524    *-------------------------------------------------------------------------------
525    * Declaration  : static void io_cmd_exe_rdmode(unsigned char *ope, int ope_sz,
526    *              :                               unsigned char *rd,  int rd_sz)
527    *-------------------------------------------------------------------------------
528    * Description  : 指定されたコマンドを実行します。
529    *              : 引数 ope を送信した後、引数 rd にデータを受信します。
530    *              : 転送は全てバイトアクセス幅で行います。
531    *              : ope_sz は 0～8 のいずれかの値を設定してください。
532    *              : rd_sz は 0 以上の値を設定することが可能です。
533    *-------------------------------------------------------------------------------
534    * 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
535    *              : int ope_sz         ; I : オペコード部とアドレス部のバイト数
536    *              : unsigned char *rd  ; I : 受信データを格納するバッファアドレス
537    *              : int rd_sz          ; I : データ部のバイト数
538    *-------------------------------------------------------------------------------
539    * Return Value : void
540    *-------------------------------------------------------------------------------
541    * Note        : None
542    *""FUNC COMMENT END""*************************************************************/
```

3.19 サンプルプログラムリスト"serial_flash.c"（16）

```
543   static void io_cmd_exe_rdmode(unsigned char *ope, int ope_sz, unsigned char *rd, int rd_sz)
544   {
545     /* ==== バッファリセット ==== */
546     RSPI0.SPBFCR.BYTE  = 0xC0u;
547     RSPI0.SPBFCR.BYTE  = 0x00u;
548
549     /* ---- SPI 転送許可 ---- */
550     RSPI0.SPCR.BIT.SPE = 1;
551
552     /* ---- MOSI(コマンド、アドレス、ダミー) ---- */
553     RSPI0.SPCMD0.BIT.CPOL= 1;        /* アイドル時の RSPCK は 1 */
554     RSPI0.SPCMD0.BIT.CPHA= 0;        /* 偶数エッジ（立ち上がり）でデータ変化 */
555
556     while(ope_sz--){
557       RSPI0.SPDR.BYTE = *ope++;  /* コマンドは 8 バイト以下とする */
558     }
559     io_wait_tx_end();                /* 完了待ち */
560
561     /* ---- MISO(リードデータ) ---- */
562     RSPI0.SPCMD0.BIT.CPOL= 0;        /* アイドル時の RSPCK は 0 */
563     RSPI0.SPCMD0.BIT.CPHA= 1;        /* 偶数エッジ（立ち下がり）でデータサンプル */
564
565     RSPI0.SPBFCR.BYTE  = 0xC0u;      /* バッファリセット */
566     RSPI0.SPBFCR.BYTE  = 0x00u;
567
568     RSPI0.SPDCR.BIT.TXDMY = 1;       /* ダミー送信 許可 */
569     while(rd_sz--){
570       while( RSPI0.SPSR.BIT.SPRF == 0){
571           /* wait */
572       }
573       *rd++ = RSPI0.SPDR.BYTE;
574     }
575     RSPI0.SPDCR.BIT.TXDMY = 0;       /* ダミー送信 禁止 */
576     io_wait_tx_end();                /* 完了待ち */
577
578     /* ---- SPI 転送終了（SSL ネゲート） ---- */
579     RSPI0.SPCR.BIT.SPE = 0;
580   }
```

3.20 サンプルプログラムリスト"serial_flash.c"（17）

```
581   /*""FUNC COMMENT""*****************************************************************
582    * ID         :
583    * Outline     : コマンド実行(リードデータあり、ロングワード転送)
584    *-------------------------------------------------------------------------------
585    * Include     :
586    *-------------------------------------------------------------------------------
587    * Declaration  : static void io_cmd_exe_rdmode_cpu_l(unsigned char *ope, int ope_sz,
588    *              :                                     unsigned long *rd,  int rd_sz);
589    *-------------------------------------------------------------------------------
590    * Description  : 指定されたコマンドを実行します。
591    *              : 引数 ope を送信した後、引数 rd にデータを受信します。
592    *              : リードデータの転送はロングワードアクセス幅で行います。
593    *              : ope_sz は 0～8 のいずれかの値を設定してください。
594    *              : rd_sz は 0 以上の値を指定することが可能ですが、4 の倍数を設定してください。
595    *-------------------------------------------------------------------------------
596    * Argument    : unsigned long *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
597    *              : int ope_sz        ; I : オペコード部とアドレス部のバイト数
598    *              : unsigned long *rd  ; I : 受信データを格納するバッファアドレス
599    *              : int rd_sz         ; I : データ部のバイト数
600    *-------------------------------------------------------------------------------
601    * Return Value : void
602    *-------------------------------------------------------------------------------
603    * Note         : None
604    *""FUNC COMMENT END""*************************************************************/
```

3.21 サンプルプログラムリスト"serial_flash.c"（18）

```
605   static void io_cmd_exe_rdmode_cpu_l(unsigned char *ope, int ope_sz, unsigned long *rd, int rd_sz)
606   {
607     /* ==== バッファリセット ==== */
608     RSPI0.SPBFCR.BYTE  = 0xC0u;
609     RSPI0.SPBFCR.BYTE  = 0x00u;
610
611     /* ---- SPI 転送許可 ---- */
612     RSPI0.SPCR.BIT.SPE = 1;
613
614     /* ---- MOSI(コマンド、アドレス、ダミー) ---- */
615     RSPI0.SPCMD0.BIT.CPOL= 1;           /* アイドル時の RSPCK は 1 */
616     RSPI0.SPCMD0.BIT.CPHA= 0;           /* 偶数エッジ（立ち上がり）でデータ変化 */
617
618     while(ope_sz--){
619       RSPI0.SPDR.BYTE = *ope++;         /* コマンドは 8 バイト以下とする */
620     }
621     io_wait_tx_end();                   /* 完了待ち */
622
623     /* ---- MISO(リードデータ) ---- */
624     RSPI0.SPCMD0.BIT.CPOL= 0;           /* アイドル時の RSPCK は 0 */
625     RSPI0.SPCMD0.BIT.CPHA= 1;           /* 偶数エッジ（立ち下がり）でデータサンプル */
626
627     RSPI0.SPBFCR.BYTE  = 0xC0u;         /* バッファリセット */
628     RSPI0.SPBFCR.BYTE  = 0x00u;
629
630     RSPI0.SPDCR.BIT.SPLW = 3;           /* SPDR のアクセス幅 32 ビット */
631     RSPI0.SPCMD0.BIT.SPB = 3;           /* 転送データ長 32 ビット */
632
633     RSPI0.SPDCR.BIT.TXDMY = 1;          /* ダミー送信 許可 */
634     rd_sz >>= 2;                        /* 転送回数をロングワードで算出 */
635     while( rd_sz-- ){
636       while( RSPI0.SPSR.BIT.SPRF == 0){
637           /* wait */
638       }
639       *rd++ = RSPI0.SPDR.LONG;
640     }
641     RSPI0.SPDCR.BIT.TXDMY = 0;          /* ダミー送信 禁止 */
642     io_wait_tx_end();                   /* 完了待ち */
643
644     /* ==== SPI 設定をデフォルトに戻す ==== */
645     RSPI0.SPCR.BIT.SPE = 0;
646     RSPI0.SPDCR.BIT.SPLW = 1;           /* SPDR のアクセス幅 8 ビット */
647     RSPI0.SPCMD0.BIT.SPB = 7;           /* 転送データ長 8 ビット */
648   }
```

3.22 サンプルプログラムリスト"serial_flash.c"（19）

```
649     #ifdef SF_USE_DMAC
650   /*""FUNC COMMENT""****************************************************************
651    * ID          :
652    * Outline     : コマンド実行(リードデータあり、ロングワード転送＋DMA)
653    *-------------------------------------------------------------------------------
654    * Include     :
655    *-------------------------------------------------------------------------------
656    * Declaration  : static void io_cmd_exe_rdmode_dma_l(unsigned char *ope, int ope_sz,
657    *              :                                     unsigned long *rd,  int rd_sz);
658    *-------------------------------------------------------------------------------
659    * Description  : 指定されたコマンドを実行します。
660    *              : 引数 ope を送信した後、引数 rd にデータを受信します。
661    *              : リードデータの転送はロングワードアクセス幅で DMA 転送を用いて行います。
662    *              : ope_sz は 0～8 のいずれかの値を設定してください。
663    *              : rd_sz は 0 以上の値を指定することが可能ですが、4 の倍数を設定してください。
664    *-------------------------------------------------------------------------------
665    * Argument    : unsigned long *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
666    *              : int ope_sz         ; I : オペコード部とアドレス部のバイト数
667    *              : unsigned long *rd  ; I : 受信データを格納するバッファアドレス
668    *              : int rd_sz          ; I : データ部のバイト数
669    *-------------------------------------------------------------------------------
670    * Return Value : void
671    *-------------------------------------------------------------------------------
672    * Note        : None
673    *""FUNC COMMENT END""************************************************************/
674   static void io_cmd_exe_rdmode_dma_l(unsigned char *ope, int ope_sz, unsigned long *rd, int rd_sz)
675   {
676
677     /* ==== バッファリセット ==== */
678     RSPI0.SPBFCR.BYTE  = 0xC0u;
679     RSPI0.SPBFCR.BYTE  = 0x00u;
680
681     /* ---- SPI 転送許可 ---- */
682     RSPI0.SPCR.BIT.SPE = 1;
683
684     /* ---- MOSI(コマンド、アドレス、ダミー) ---- */
685     RSPI0.SPCMD0.BIT.CPOL= 1;       /* アイドル時の RSPCK は 1 */
686     RSPI0.SPCMD0.BIT.CPHA= 0;       /* 偶数エッジ（立ち上がり）でデータ変化 */
687
688     while(ope_sz--){
689       RSPI0.SPDR.BYTE = *ope++;     /* コマンドは 8 バイト以下とする */
690     }
691     io_wait_tx_end();               /* 完了待ち */
692
```

3.23 サンプルプログラムリスト"serial_flash.c"（20）

```
693     /* ==== MISO(リードデータ) ==== */
694     RSPI0.SPCMD0.BIT.CPOL= 0;               /* アイドル時の RSPCK は 0 */
695     RSPI0.SPCMD0.BIT.CPHA= 1;               /* 偶数エッジ（立ち下がり）でデータサンプル */
696
697     RSPI0.SPBFCR.BYTE  = 0xC0u;             /* バッファリセット */
698     RSPI0.SPBFCR.BYTE  = 0x00u;
699
700     RSPI0.SPDCR.BIT.SPLW = 3;               /* SPDR のアクセス幅 32 ビット */
701     RSPI0.SPCMD0.BIT.SPB = 3;               /* 転送データ長 32 ビット */
702
703     /* ---- DMA 転送許可---- */
704     RSPI0.SPCR.BIT.SPRIE =  1;              /* 割込み許可（DMA 用） */
705     DMAC.CHCR0.LONG      =  0x00000000;     /* DMA 禁止 */
706     DMAC.DMARS0.WORD =   0x0052u;           /* RSPI0 Rx */
707     DMAC.DMAOR.WORD      =  0x0001u;        /* 全体 DMA 許可 */
708     DMAC.SAR0.LONG       =  (unsigned long)&(RSPI0.SPDR.BYTE);
709     DMAC.DAR0.LONG       =  (unsigned long)rd;         /* 転送先アドレス */
710     DMAC.DMATCR0.LONG    =  rd_sz >> 2;                /* 転送サイズ*/
711     DMAC.CHCR0.LONG      =  (unsigned long)0x00004811; /* トランスファサイズ【ロング】、DMA 許可 */
712
713     /* ---- データ受信 ---- */
714     RSPI0.SPDCR.BIT.TXDMY = 1;              /* ダミー送信 許可 */
715     while(DMAC.CHCR0.BIT.TE == 0){          /* DMA 転送完了を待つ */
716       /* wait */
717     }
718     RSPI0.SPDCR.BIT.TXDMY = 0;              /* ダミー送信 禁止 */
719     DMAC.CHCR0.LONG      =  0x00000000ul;   /* DMA0 禁止 */
720     io_wait_tx_end();                       /* 完了待ち */
721
722     /* ==== SPI 設定をデフォルトに戻す ==== */
723     RSPI0.SPCR.BIT.SPE = 0;
724     RSPI0.SPDCR.BIT.SPLW = 1;               /* SPDR のアクセス幅 8 ビット */
725     RSPI0.SPCMD0.BIT.SPB = 7;               /* 転送データ長 8 ビット */
726   }
727   #endif  /* SF_USE_DMAC */
```

3.24 サンプルプログラムリスト"serial_flash.c"（21）

```
728   /*""FUNC COMMENT""*****************************************************************
729    * ID          :
730    * Outline     : 送信完了待ち
731    *-------------------------------------------------------------------------------
732    * Include     :
733    *-------------------------------------------------------------------------------
734    * Declaration  : static void io_wait_tx_end(void);
735    *-------------------------------------------------------------------------------
736    * Description  : 送信完了が確認できるまで内部でループします。
737    *-------------------------------------------------------------------------------
738    * Argument    : void
739    *-------------------------------------------------------------------------------
740    * Return Value : void
741    *-------------------------------------------------------------------------------
742    * Note        : None
743    *""FUNC COMMENT END""*************************************************************/
744   static void io_wait_tx_end(void)
745   {
746     while(RSPI0.SPSR.BIT.TEND == 0){
747       /* wait */
748     }
749   }
750
751   /* End of File */
```

3.25 サンプルプログラムリスト"serial_flash.h"（1）

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Technology Corp. and is only
5     *   intended for use with Renesas products. No other uses are authorized.
6     *
7     *   This software is owned by Renesas Technology Corp. and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   TECHNOLOGY CORP. NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THE THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ***********************************************************************
28    *   Copyright (C) 2009. Renesas Technology Corp., All Rights Reserved.
29    *""FILE COMMENT""*********** Technical reference data *************************
30    *   System Name : SH7264 Sample Program
31    *   File Name   : serial_flash.h
32    *   Abstract    : ルネサスシリアルペリフェラルインタフェース
33    *               : シリアルフラッシュメモリの高速リードライト
34    *   Version     : 1.00.00
35    *   Device      : SH7262/SH7264
36    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.04.01).
37    *               : C/C++ compiler package for the SuperH RISC engine family
38    *               :                          (Ver.9.02 Release00).
39    *   OS          : None
40    *   H/W Platform: M3A-HS64G50(CPU board)
41    *   Description :
42    ***********************************************************************
43    *   History     : Mar.09,2009 Ver.1.00.00
44    *""FILE COMMENT END""*************************************************************/
```

3.26 サンプルプログラムリスト"serial_flash.h"（2）

```
45    #ifndef _SERIAL_FLASH_H_
46    #define _SERIAL_FLASH_H_
47
48    /* ==== マクロ定義 ==== */
49    #define SF_PAGE_SIZE       256         /* シリアルフラッシュメモリのページサイズ */
50    #define SF_SECTOR_SIZE     0x10000     /* セクタサイズ = 64KB   */
51    #define SF_NUM_OF_SECTOR   32          /* セクタ数 32   */
52    enum sf_req{
53      SF_REQ_PROTECT = 0,                  /* プロテクト要求 */
54      SF_REQ_UNPROTECT                     /* プロテクト解除要求 */
55    };
56    /* ==== 関数プロトタイプ宣言 ==== */
57    void sf_init_serial_flash(void);
58    void sf_protect_ctrl(enum sf_req req);
59    void sf_chip_erase(void);
60    void sf_sector_erase(int sector_no);
61    void sf_byte_program(unsigned long addr, unsigned char *buf, int size);
62    void sf_byte_read(unsigned long addr, unsigned char *buf, int size);
63    void sf_byte_read_long(unsigned long addr, unsigned long *buf, int size);
64
65    /* ==== 変数定義 ==== */
66
67    #endif /* _SERIAL_FLASH_H_ */
68    /* End of File */
```

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	2009.06.19	ー	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