RX62N グループ

RSPI を使ったクロック同期式シングルマスタ制御ソフトウェア

R01AN0323JJ0104
Rev.1.04
2012.04.05

要旨

本アプリケーションノートでは、RX62N 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（以下、RSPI）のクロック同期式（３線式）シリアル通信を使用したクロック同期式シングルマスタ制御方法とサンプルコードの使用方法を説明します。

ポート制御による SPI スレーブデバイスセレクト制御を付加することにより、SPI モード・シングルマスタ制御が可能です。

本サンプルコードは、マイコン固有のシングルマスタ基本制御方法を実現したものです。本サンプルコードを使用して、スレーブデバイスを制御するためのソフトウェアを作成してください。

なお、スレーブデバイス制御のためのソフトウェア例を用意していますので、入手してください。

対象デバイス

対応 MCU　　RX62N グループ

動作確認に使用したデバイス　　ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様に合わせて変更し、十分評価してください。

目次

1. 仕様

RX62N グループの RSPI のクロック同期式（３線式）シリアル通信を使用し、クロック同期式制御を行います。ポート制御による SPI スレーブデバイスセレクト制御を付加することにより、SPI モード・シングルマスタ制御が可能です。

表 1-1に使用する周辺機器と用途を、図 1-1に使用例を示します。

以下に、機能概略を示します。

- マスタデバイスを RX62N とし、RSPI を使ったクロック同期式シングルマスタ用ブロック型デバイスドライバです。
- MCU 内蔵のクロック同期式（３線式）シリアル通信機能を使用します。また、ユーザ設定した 1 チャネルの使用が可能です。複数チャネルの使用は、できません。
- 本サンプルコードは、チップセレクト制御をサポートしていません。SPI デバイスを制御する場合、別途、デバイスセレクト制御を組み込む必要があります。
- ビッグエンディアン／リトルエンディアンでの動作が可能です。
- MSB ファーストフォーマットで転送します。
- CPU 転送のみをサポートしています。DMAC/EXDMAC/DTC 転送をサポートしていません。
- 割り込みによる転送起動をサポートしていません。
- 受信方法として、通常受信モードと高速受信モードの選択が可能です。
- 高速受信モード選択時は、スーパバイザモードでの動作が可能です。ユーザモードでは動作しません。
  通常受信モード選択時は、スーパバイザモード／ユーザモードでの動作が可能です。
- 高速受信モード選択時は、NMI 割り込みを禁止させてください。

表 1-1　使用する周辺機器と用途

周辺機器	用途
RSPI	**クロック同期式（３線式）シリアル** **1ch（必須）**
Port	SPI **スレーブデバイスセレクト制御信号用** **使用デバイス数分のポートが必要（必須）** **ただし、本サンプルコードでは、扱いません。**

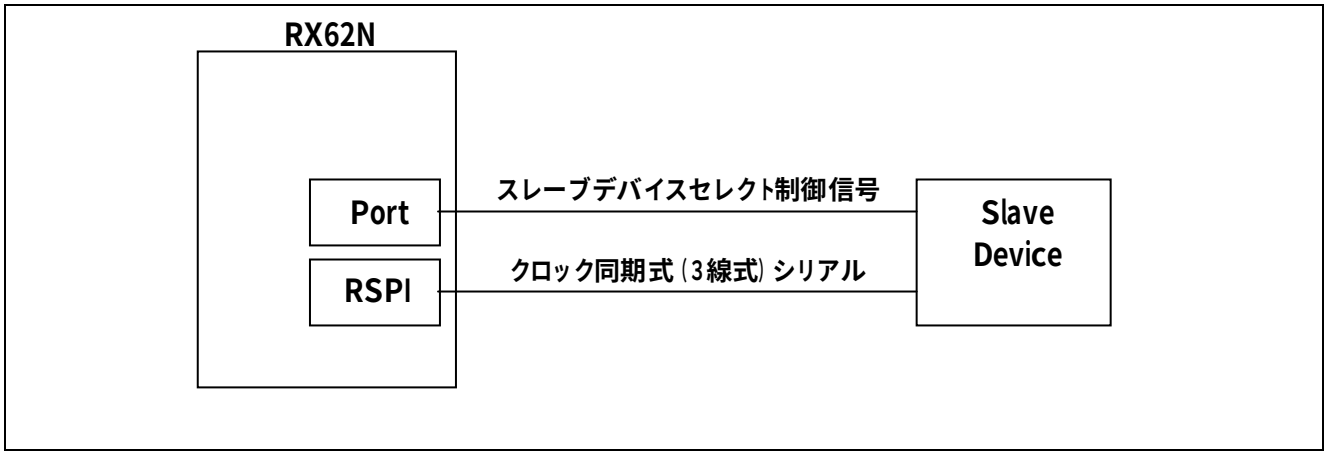


図 1-1　使用例

2. 動作確認条件

本アプリケーションノートのサンプルコードは、以下の動作条件で動作を確認しています。

表 2-1　動作確認条件

項目	内容
使用マイコン	RX62N グループ（プログラム ROM 512KB RAM 96KB）
使用メモリ	ルネサス エレクトロニクス製 R1EX25xxx シリーズ SPI Serial EEPROM
動作周波数	ICLK：96MHz、PCLK：48MHz
動作電圧	3.3V
総合開発環境	ルネサス エレクトロニクス製 High-performance embedded Workshop Version 4.09.00.007
C コンパイラ	ルネサス エレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ （ツールチェーン 1.0.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル"2"、最適化方法"サイズ優先"
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.2.04
使用ソフトウェア	Renesas R1EX25xxx シリーズ Serial EEPROM 制御ソフトウェア (R01AN0565JJ)　Ver.2.01
使用ボード	Renesas Starter Kit for RX62N

表 2-2　動作確認条件

項目	内容
使用マイコン	RX62N グループ（プログラム ROM 512KB RAM 96KB）
使用メモリ	Micron Technology 社製 M25P シリーズ Serial Flash memory 64Mbit
動作周波数	ICLK：96MHz、PCLK：48MHz
動作電圧	3.3V
総合開発環境	ルネサス エレクトロニクス製 High-performance embedded Workshop Version 4.09.00.007
C コンパイラ	ルネサス エレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ （ツールチェーン 1.0.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル"2"、最適化方法"サイズ優先"
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.2.04
使用ソフトウェア	Micron Technology 社製 M25P シリーズ Serial Flash memory 制御ソフトウェア(R01AN0566JJ)　Ver.2.01
使用ボード	Renesas Starter Kit for RX62N

表 2-3　動作確認条件

項目	内容
使用マイコン	RX62N グループ（プログラム ROM 512KB RAM 96KB）
使用メモリ	Micron Technology 社製 M45PE シリーズ Serial Flash memory 1Mbit
動作周波数	ICLK：96MHz、PCLK：48MHz
動作電圧	3.3V
総合開発環境	ルネサス エレクトロニクス製 High-performance embedded Workshop Version 4.09.00.007
Cコンパイラ	ルネサス エレクトロニクス製 RX ファミリ用 C/C++コンパイラケッケージ （ツールチェーン 1.0.1.0）
	コンパイルオプション 総合開発環境のデフォルト設定（※１）を使用しています。 ※１：最適化レベル"2"、最適化方法"サイズ優先"
エンディアン	ビッグエンディアン／リトルエンディアン
サンプルコードのバージョン	Ver.2.04
使用ソフトウェア	Micron Technology 社製 M45PE シリーズ Serial Flash memory 制御ソフトウェア(R01AN0567JJ)　Ver.2.01
使用ボード	Renesas Starter Kit for RX62N

3.　関連アプリケーションノート

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

- Renesas R1EX25xxx シリーズ Serial EEPROM 制御ソフトウェア(R01AN0565JJ)
- Micron Technology 社製 M25P シリーズ Serial Flash memory 制御ソフトウェア(R01AN0566JJ)
- Micron Technology 社製 M45PE シリーズ Serial Flash memory 制御ソフトウェア(R01AN0567JJ)

4.　周辺機能説明

RSPI は、SPI 動作（4 線式）とクロック同期式動作（3 線式）が可能です。

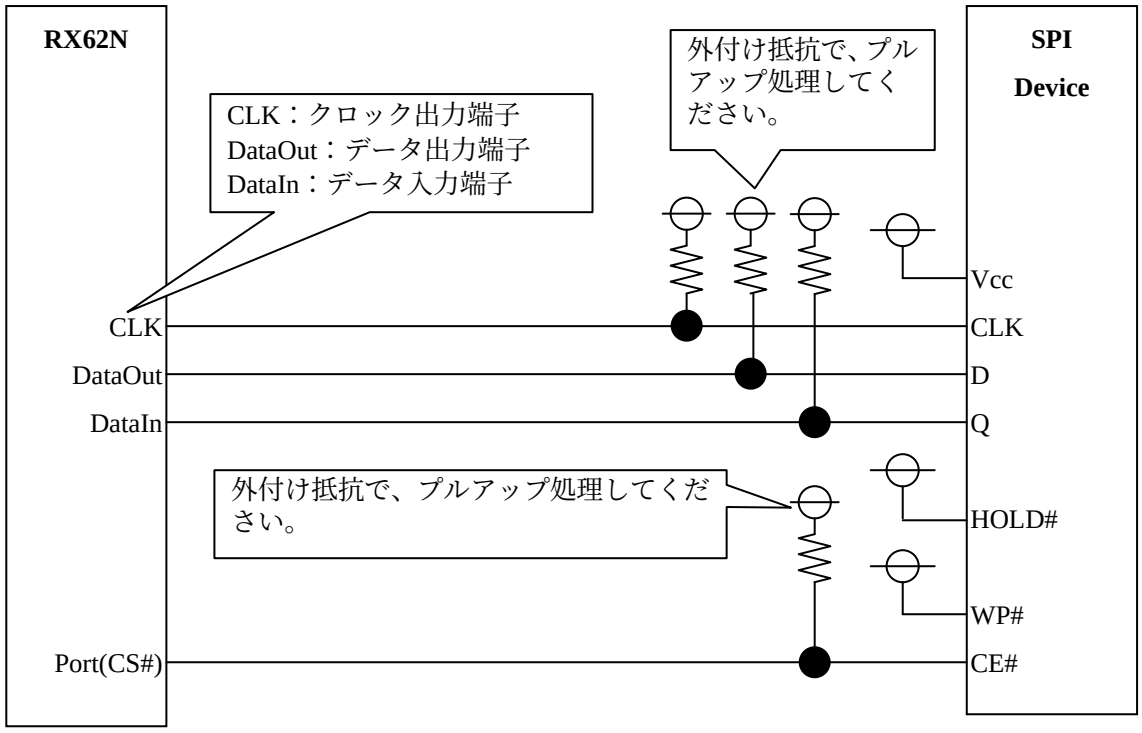
本アプリケーションノートでは、クロック同期式動作（3 線式）を使用します。本サンプルコードでは、SPI デバイスを制御する場合、SPI スレーブデバイスセレクト端子用として、Port を割り当てます。

RSPI での 4 線式制御時の SSL 端子を、3 線式制御時のポート制御にて、CE#端子に割り当てることが可能です。

5. ハードウェア説明

5.1 ハードウェア構成例

図 5-1に接続図を示します。なお、高速で動作させた場合を想定し、各信号ラインの回路的マッチングを取るためのダンピング抵抗やコンデンサの付加を検討してください。



- MCU 上のシリアル I/O に使用される端子名は、MCU に依存します。
- 本アプリケーションノートでは、サンプルコードの記述に合わせて、CLK 端子、DataIn 端子、DataOut 端子、Port(CS#)端子として、表します。

図 5-1　RX62N RSPI と SPI スレーブデバイスの接続例

5.2 使用端子一覧

表 5-1に、使用端子と機能を示します。

表 5-1　使用端子と機能

端子名	入出力	内容
RSPCK （図 5-1の CLK）	出力	クロック出力
MOSI （図 5-1の DataOut）	出力	マスタデータ出力
MISO （図 5-1の DataIn）	入力	マスタデータ入力
Port （図 5-1の Port(CS#)）	出力	スレーブデバイスセレクト出力 ただし、本サンプルコードでは、扱いません。

6. ソフトウェア説明

6.1 動作概要

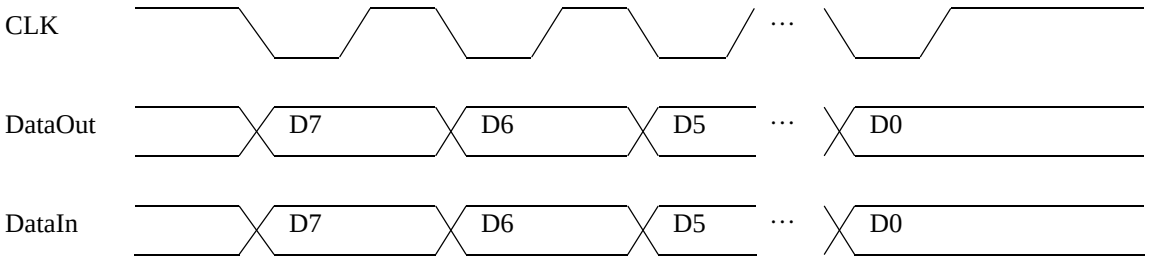
RSPI のクロック同期式（3線式）シリアル通信機能を使って、クロック同期式シングルマスタ制御を実現します。

本サンプルコードでは、以下の制御を行っています。

- データの送信／受信を、クロック同期式モード（内部クロック使用）で、制御する。

6.1.1 クロック同期式モードで発生させるタイミング

SPI スレーブデバイス制御のため、図 6-1に示す SPI モード 3（CPOL=1、CPHA=1）のタイミングを発生します。



・MCU-> SPI スレーブデバイスの送信時：転送クロックの立ち下がりで送信データ出力開始

・SPI スレーブデバイス->MCU の受信時：転送クロックの立ち上がりで受信データの入力取り込み

・MSB ファーストでの転送

転送を行っていないときの CLK 端子のレベルは、”H”です。

図 6-1 クロック同期式モード　タイミング設定

使用可能なシリアルクロック周波数は、MCU および SPI スレーブデバイスのデータシートで、確認してください。

6.1.2 SPIスレーブデバイスのCE#端子制御

本サンプルコードでは、SPI スレーブデバイスの CE#端子を制御しません。SPI デバイスを制御する場合、別途、SPI スレーブデバイスの CE#端子の制御を追加してください。

制御方法としては、MCU の Port に接続し、MCU 汎用ポート出力で、制御させることを推奨します。

また、SPI デバイスの CE#（MCU の Port(CS#)）信号の立ち下がりから、SPI デバイスの CLK（MCU の CLK）信号の立ち下がりまでの時間（SPI デバイスの CE#セットアップ時間）を設けてください。

同様に、SPI デバイスの CLK（MCU の CLK）信号の立ち上がりから、SPI デバイスの CE#（MCU の Port(CS#)）信号の立ち上がりまでの時間（SPI デバイスの CE#ホールド時間）を設けてください。

SPI デバイスのデータシートを確認して、システムに応じたソフトウェア・ウェイト時間を設定してください。

RSPI での 4 線式制御時の SSL 端子を、3 線式制御時のポート制御にて、CE#端子に割り当てることが可能です。

6.2　ソフトウェア制御概要

6.2.1　ソフトウェア構成

本サンプルコードは、マイコン固有のシングルマスタ基本制御方法を実現したものです。

本サンプルコードでは、SPI スレーブデバイスの CE#端子制御無しの SPI モード 3（CPOL=1、CPHA=1）を使った制御を実現しています。

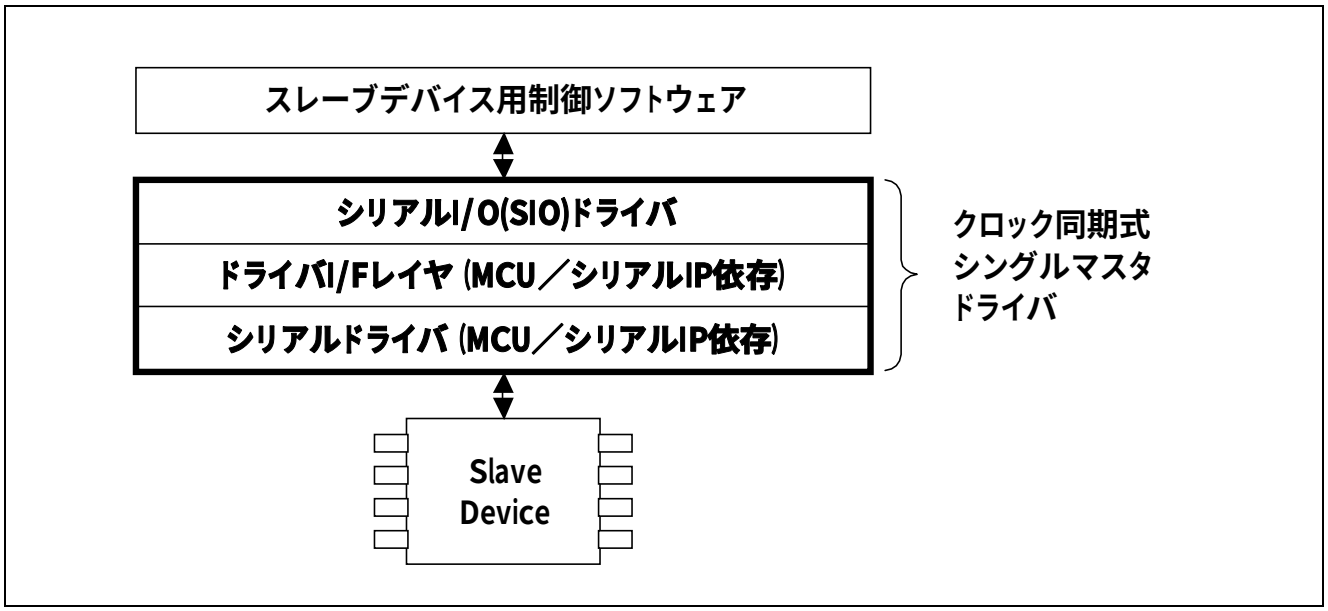


図 6-2　ソフトウェア構成

6.8 状態遷移図、および6.9 関数仕様に示す関数を参照して、スレーブデバイスへのアクセスを実現してください。

具体的な使用例として、3 関連アプリケーションノートに記載済のアプリケーションノートを参考にしてください。

6.2.2　データバッファと送信／受信データの関係

本サンプルコードは、ブロック型デバイスドライバであり、送信／受信データポインタを引数として設定します。RAM 上のデータバッファのデータ並びと送信／受信順番の関係は、以下のとおりで、エンディアンや使用するシリアル通信機能に関係なく、送信データバッファの並びの順に送信し、また、受信の順に受信データバッファに書き込みます。

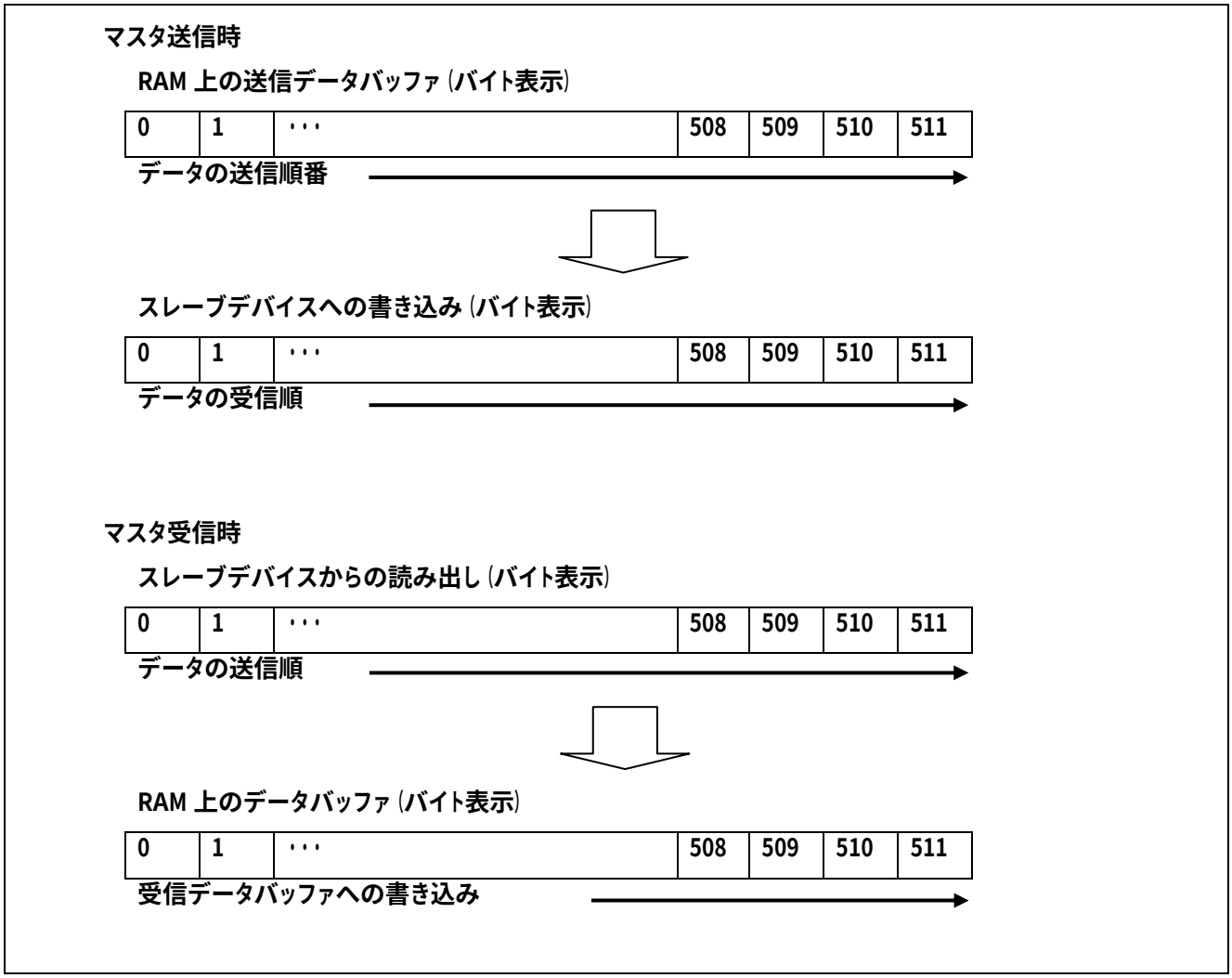


図 6-3　データバッファと送信／受信データの関係

6.3　必要メモリサイズ

表 6-1に必要とするメモリサイズを示します。

表 6-1のメモリサイズは、7.2.2 R_SIO_rspi.h (1) 使用する動作モードの定義　で SIO_OPTION_4 を選択した場合の値です。選択する定義により、メモリサイズは異なります。

最大使用ユーザスタックサイズは、Serial EEPROM 制御ソフトウェアを使用した場合の値であり、Serial EEPROM 制御ソフトウェアのスタックサイズも含みます。

表 6-1　必要メモリサイズ

使用メモリ	サイズ	備考
ROM	1,432 バイト（リトルエンディアン）	R_SIO_rspi_rx.c
RAM	0 バイト（リトルエンディアン）	R_SIO_rspi_rx.c
最大使用ユーザスタック	204 バイト	
最大使用割り込みスタック	－	

必要メモリサイズは、C コンパイラのバージョンやコンパイルオプションにより異なります。
エンディアンにより、上記のメモリサイズは、異なります。

6.4　ファイル構成

表 6-2に、サンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成するファイルを除きます。

表 6-2　ファイル構成

¥an_r01an0323jj_rx62n <DIR>			**サンプルコードのフォルダ**
r01an0323jj0104_rx62n.pdf			**アプリケーションノート**
¥ r01an0323jj_rx62n_src <DIR>			**プログラム格納用フォルダ**
	¥com <DIR>		**共通関数格納用フォルダ**
	（注１）	mtl_com.c	**共通関数の各種定義**
		mtl_com.h.common	**共通ヘッダファイル**
		mtl_com.h.RX600	**共通関数のヘッダファイル**
		mtl_endi.c	**共通ファイル（エンディアン設定関連）**
		mtl_mem.c	**共通ファイル（標準ライブラリ関数）**
		mtl_os.c　mtl_os.h	**共通ファイル（標準ライブラリ関数）**
		mtl_str.c	**共通ファイル（標準ライブラリ関数）**
		mtl_tim.c　mtl_tim.h	**共通ファイル（ループタイマ関連）**
		mtl_tim.h.sample	**ループタイマの設定値サンプル**
	¥r_sio_rspi_rx <DIR>		RSPI **用クロック同期式シングルマスタ制御ソフトウェアのフォルダ**
		R_SIO.h	**ヘッダファイル**
		R_SIO_rspi.h.rx62x	I/F **モジュール共通定義**
		R_SIO_rspi_rx.c	I/F **モジュール**

注１． com **フォルダに含まれるファイルは、スレーブデバイス用制御ソフトウェアも使用するものです。最新のものを使用してください。**

6.5　定数一覧

6.5.1　リターン値

表 6-3に、サンプルコードで使用するリターン値を示します。

表 6-3　リターン値

定数名	設定値	内容
SIO_OK	(error_t)(0)	Successful operation
SIO_ERR_PARAM	(error_t)(-1)	Parameter error
SIO_ERR_HARD	(error_t)(-2)	Hardware error
SIO_ERR_OTHER	(error_t)(-7)	Other error

6.5.2　各種定義

表 6-4に、サンプルコードで使用する各種定義した値を示します。

表 6-4　各種定義値

定数名	設定値	内容
SIO_LOG_ERR	(uint8_t)0x01	Log type : Error
SIO_TRUE	(uint8_t)0x01	Flag "ON"
SIO_FALSE	(uint8_t)0x00	Flag "OFF"
SIO_HI	(uint8_t)0x01	Port "H"
SIO_LOW	(uint8_t)0x00	Port "L"
SIO_OUT	(uint8_t)0x01	Port output setting
SIO_IN	(uint8_t)0x00	Port input setting
SIO_TX_WAIT	(uint16_t)50000	SIO transmission completion waiting time 50000* 1us =　50m
SIO_RX_WAIT	(uint16_t)50000	SIO reception completion waiting time 50000* 1us =　50ms
SIO_DMA_TX_WAIT	(uint16_t)50000	DMA transmission completion waiting time 50000* 1us =　50ms
SIO_DMA_RX_WAIT	(uint16_t)50000	DMA reception completion waiting time 50000* 1us =　50ms
SIO_T_SIO_WAIT	(uint16_t)MTL_T_1US	SIO transmission&reception completion waiting polling time
SIO_T_DMA_WAIT	(uint16_t)MTL_T_1US	DMA transmission&reception completion waiting polling time
SIO_T_BRR_WAIT	(uint16_t)MTL_T_10US	BRR setting wait time

6.5.3　各種定義

表 6-5に、サンプルコードで使用する各種定義した値を示します。

表 6-5　各種定義値

定数名	設定値	内容
SIO_TRAN_SIZE	(uint8_t)0x04	4 バイト（変更禁止）

6.6　構造体／共用体一覧

以下に、サンプルコードで使用する構造体を示します。

```
/* uint32_t <-> uint8_t conversion */
typedef union {
  uint32_t   ul;
  uint8_t uc[4];
} SIO_EXCHG_LONG;                              /* total 4byte              */


/* uint16_t <-> uint8_t conversion */
typedef union {
  uint16_t   us;
  uint8_t uc[2];
} SIO_EXCHG_SHORT;                           /* total 2byte                */
```

6.7　関数一覧

表 6-6に関数一覧を示します。

表 6-6　関数

関数名	説明
R_SIO_Init_Driver()	**ドライバ初期化処理**
R_SIO_Disable()	**シリアル** I/O **禁止設定処理**
R_SIO_Enable()	**シリアル** I/O **許可設定処理**
R_SIO_Open_Port()	**シリアル** I/O **開放設定処理**
R_SIO_Tx_Data()	**シリアル** I/O **データ送信処理**
R_SIO_Rx_Data()	**シリアル** I/O **データ受信処理**

RSPI 制御の高速化のために、SPDR レジスタを 32 ビットアクセスします。送信／受信データ格納バッファポインタを指定する場合、処理の高速化のため、開始アドレスを 4 バイト境界に合わせることを推奨します。

6.8　状態遷移図

図 6-4　**状態遷移図**に、状態遷移図を示します。

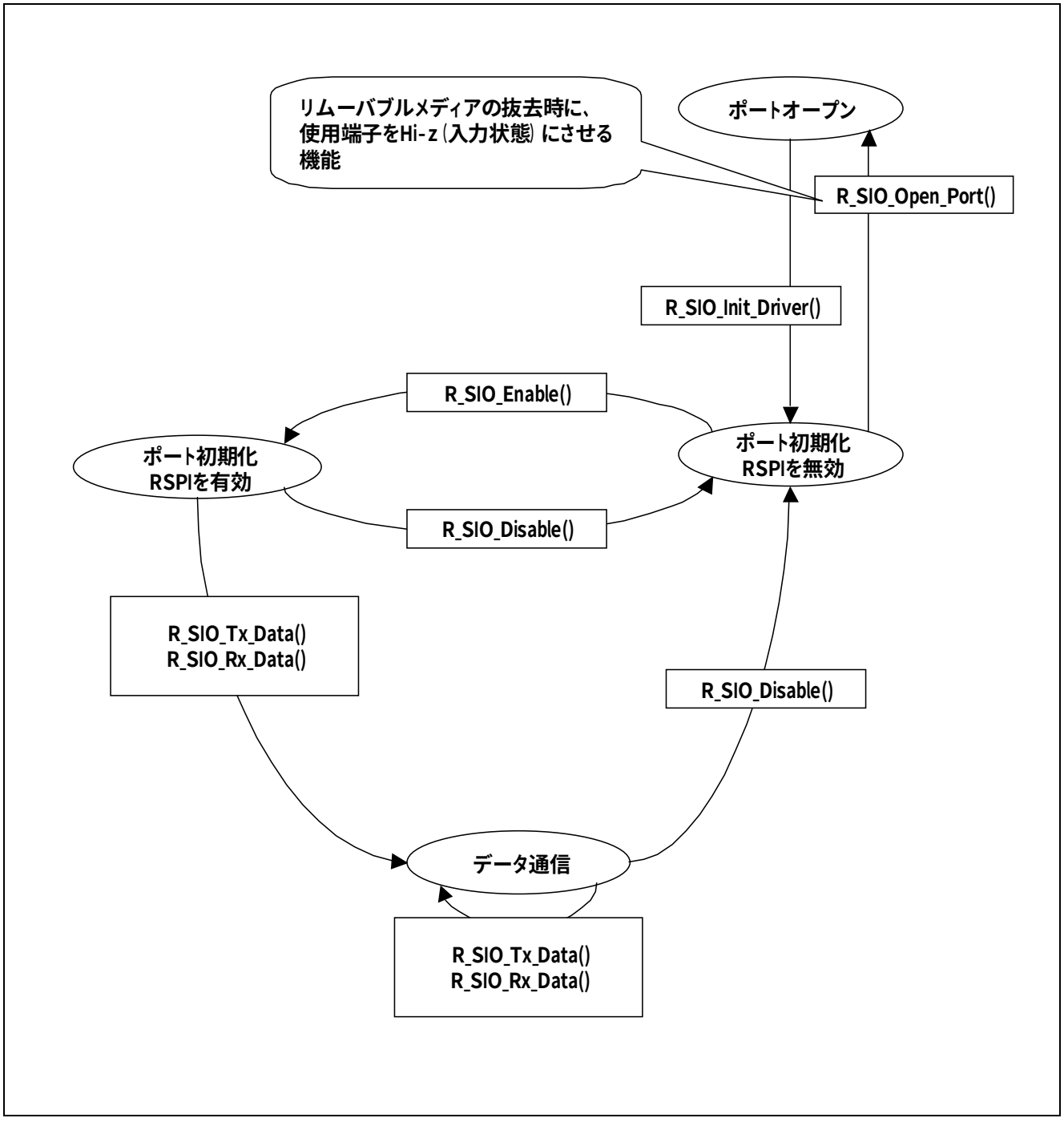


図 6-4　**状態遷移図**

6.9　関数仕様

6.9.1　ドライバ初期化処理

R_SIO_Init_Driver	
概要	**ドライバ初期化処理**
ヘッダ	R_SIO.h, R_SIO_rspi.h, mtl_com.h
宣言	error_t R_SIO_Init_Driver(void)
説明	**・ドライバの初期化を行います。シリアル** I/O **機能を無効化し、端子をポートに設定します。** **・システム起動時に一度だけ呼び出してください。** **・本関数コール前に、スレーブデバイスセレクト制御信号を**"H"**出力にしてください。**
引数	**なし**
リターン値	SIO_OK　　;　Successful operation
備考	**前の状態を考慮し、以下の処理を行います。** ・R_SIO_Disable()**をコールします。**

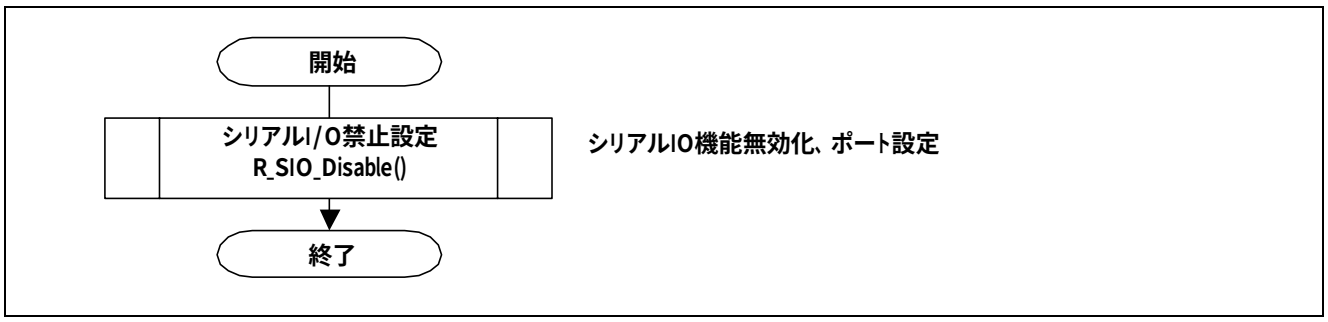


図 6-5　**ドライバ初期化処理概要**

6.9.2　シリアルI/O禁止設定処理

R_SIO_Disable	
概要	シリアル I/O 禁止設定処理
ヘッダ	R_SIO.h, R_SIO_rspi.h, mtl_com.h
宣言	error_t R_SIO_Disable(void)
説明	・シリアル I/O 機能を無効化し、端子をポートに設定します。 シリアル I/O を無効化します。 シリアル I/O で使用する端子をポート設定にします。 ・本関数コール前に、スレーブデバイスセレクト制御信号を”H”出力にしてください。
引数	なし
リターン値	SIO_OK　　;　Successful operation
備考	・RSPI 関連のレジスタへ書き込みを行うため、使用する RSPI のモジュールストップ状態を一時的に解除します。RSPI 関連のレジスタを設定後、モジュールストップ状態へ遷移します。 ・使用しない場合、本関数をコールし、シリアル I/O 機能を無効化することができます。

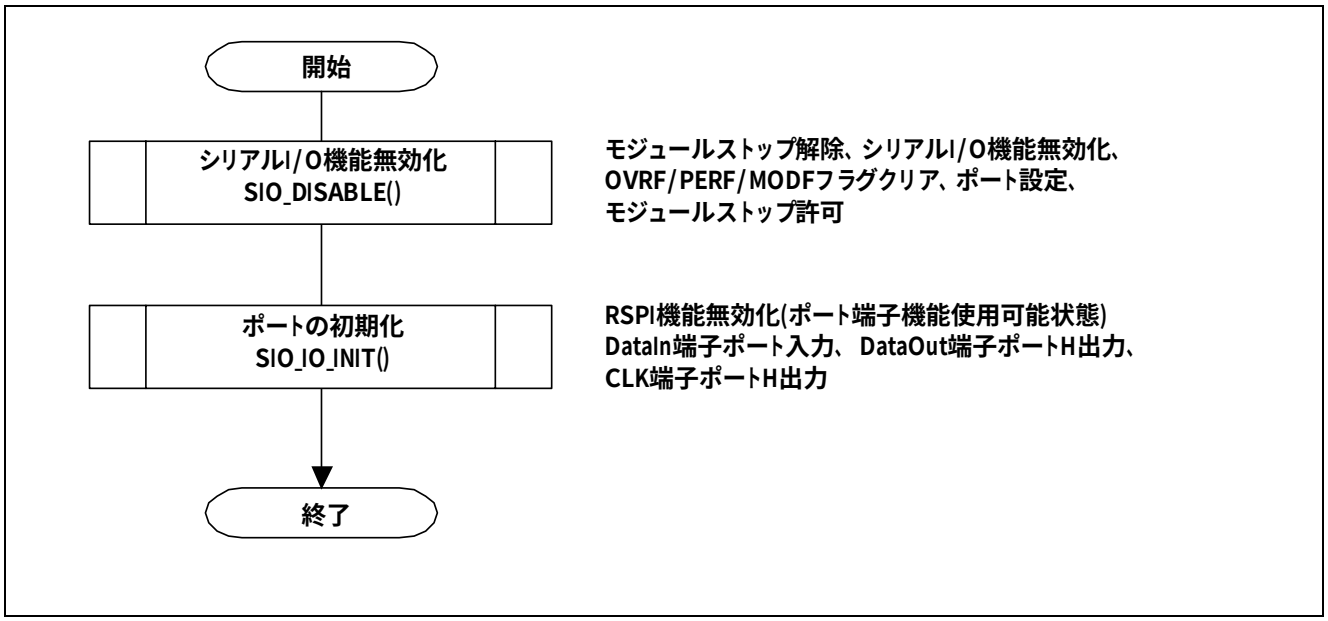


図 6-6　シリアル I/O 禁止設定処理概要

6.9.3　シリアルI/O許可設定処理

R_SIO_Enable				
概要	シリアル I/O 許可設定処理			
ヘッダ	R_SIO.h, R_SIO_rspi.h, mtl_com.h			
宣言	error_t R_SIO_Enable(uint8_t BrgData)			
説明	・シリアル I/O 機能を有効化、ビットレートを設定します。 シリアル I/O で使用する端子をポート設定にします。 シリアル I/O を有効化し、ビットレートを設定します。 ・R_SIO_Disable()コール後に、本関数をコールしてください。 ・シリアル I/O データ送信処理とシリアル I/O データ受信処理実行前に、本関数をコールしてください。 ・ビットレートを変更したい場合、本関数を使用してください。事前に、シリアル I/O 禁止設定処理を実行してください。			
引数	uint8_t	BrgData	;	ビットレート設定値
リターン値	SIO_OK		;	Successful operation
備考	・使用するシリアル I/O をモジュールストップ解除状態に設定します。 ・ソフトウェア・ウェイト（10μs）は、ビットレートの設定待ち時間です。			

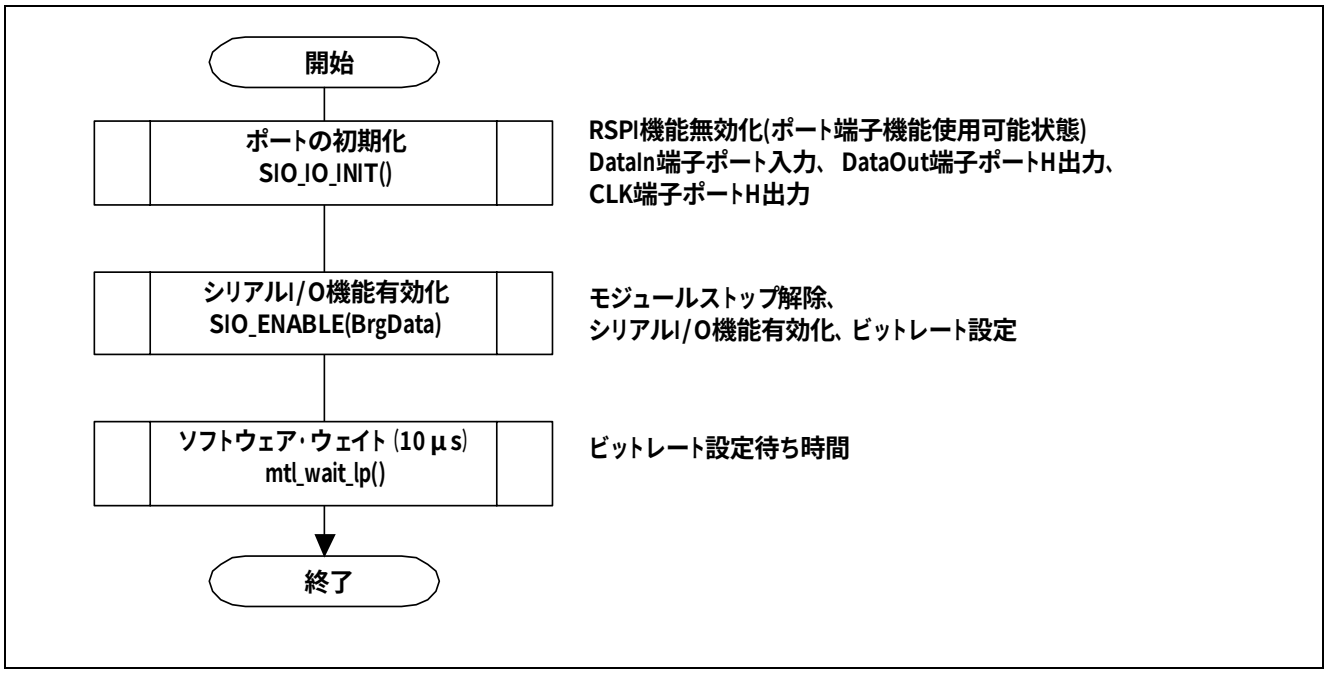


図 6-7　シリアル I/O 許可設定処理概要

6.9.4　シリアルI/O開放設定処理

R_SIO_Open_Port	
概要	SIO port(DataOut,DataIn,CLK)開放設定処理
ヘッダ	R_SIO.h, R_SIO_rspi.h, mtl_com.h
宣言	error_t R_SIO_Open_Port(void)
説明	・シリアル I/O に使用する端子をオープン（入力状態）にします。 ・本関数コール前に、スレーブデバイスセレクト制御信号を”H”出力にしてください。
引数	なし
リターン値	SIO_OK　　;　Successful operation
備考	・リムーバブルメディアの挿抜目的で、用意した関数です。リムーバブルメディアの挿入前、およびリムーバブルメディアの抜去前に、本関数を使用してください。リムーバブルメディアの抜去前には、シリアル I/O 禁止設定処理を実行してください。

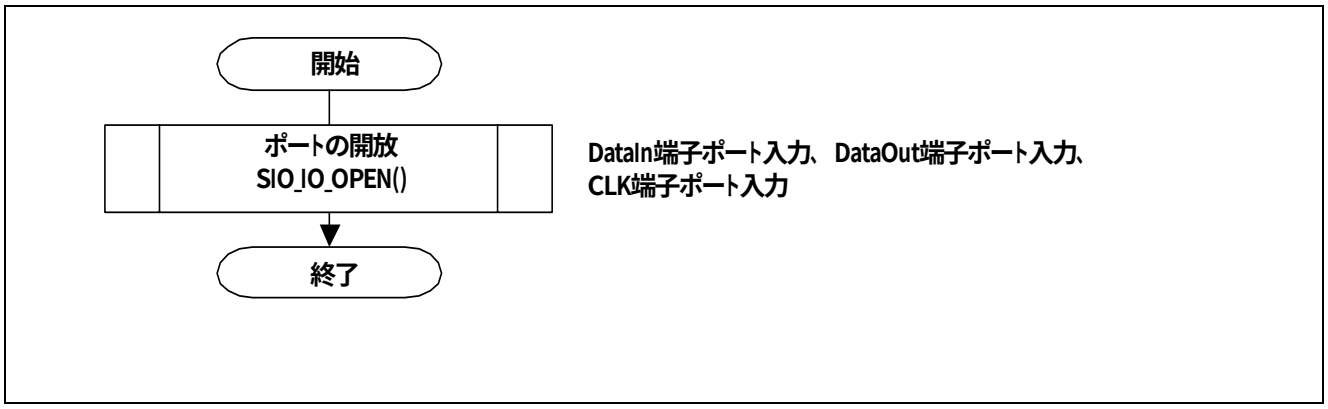


図 6-8　シリアル I/O 開放設定処理概要

6.9.5　シリアルI/Oデータ送信処理

R_SIO_Tx_Data

概要	**シリアル** I/O **データ送信処理**			
ヘッダ	R_SIO.h, R_SIO_rspi.h, mtl_com.h			
宣言	error_t R_SIO_Tx_Data(uint16_t TxCnt, uint8_t FAR* pData)			
説明	・pData **のデータを指定バイト数分送信します。** ・**本関数コール前に、シリアル** I/O **許可設定処理を実行してください。** ・**本関数の実行の結果、異常終了であれば、シリアル** I/O **禁止設定処理を実行してください。**			
引数	uint16_t	TxCnt	;	**送信バイト数**
	uint8_t FAR*	pData	;	**送信データ格納バッファポインタ**
リターン値	SIO_OK		;	Successful operation
	SIO_ERR_HARD		;	Hardware error
備考	・**半二重での送信処理の場合に使用してください。** ・**ハードウェアマニュアル記載の初期化フローチャートにしたがって、以下を実行します。（インライン関数** SIO_TX_ENABLE()**）** ①SPCR2 **設定（**RSPI **アイドル割り込み要求許可設定）** ②SPCMD **設定** ③IR **フラグクリア** ④**ポートファンクションレジスタ（**PFxSPI**）設定（**RSPI **端子の有効化）** ⑤SPCR **設定（送信許可設定）** ⑥SPCR **をリード** ・**送信完了後、上記送信許可設定の逆の処理を行うことで、シリアル通信を停止します。（インライン関数** SIO_TX_DISBLE()**）** ①SPCR **設定（送受信停止設定）** ②SPCR **をリード** ③**ポートファンクションレジスタ（**PFxSPI**）設定（**RSPI **端子の無効化）** ④SPCR2 **設定（**RSPI **アイドル割り込み要求禁止設定）** ・**送信バッファエンプティ** IR **と** RSPI **アイドル** IR **の両方を用いて、データ送信を完了確認します。** ・**継続使用しない場合、シリアル** I/O **禁止設定処理を実行することを推奨します。**			

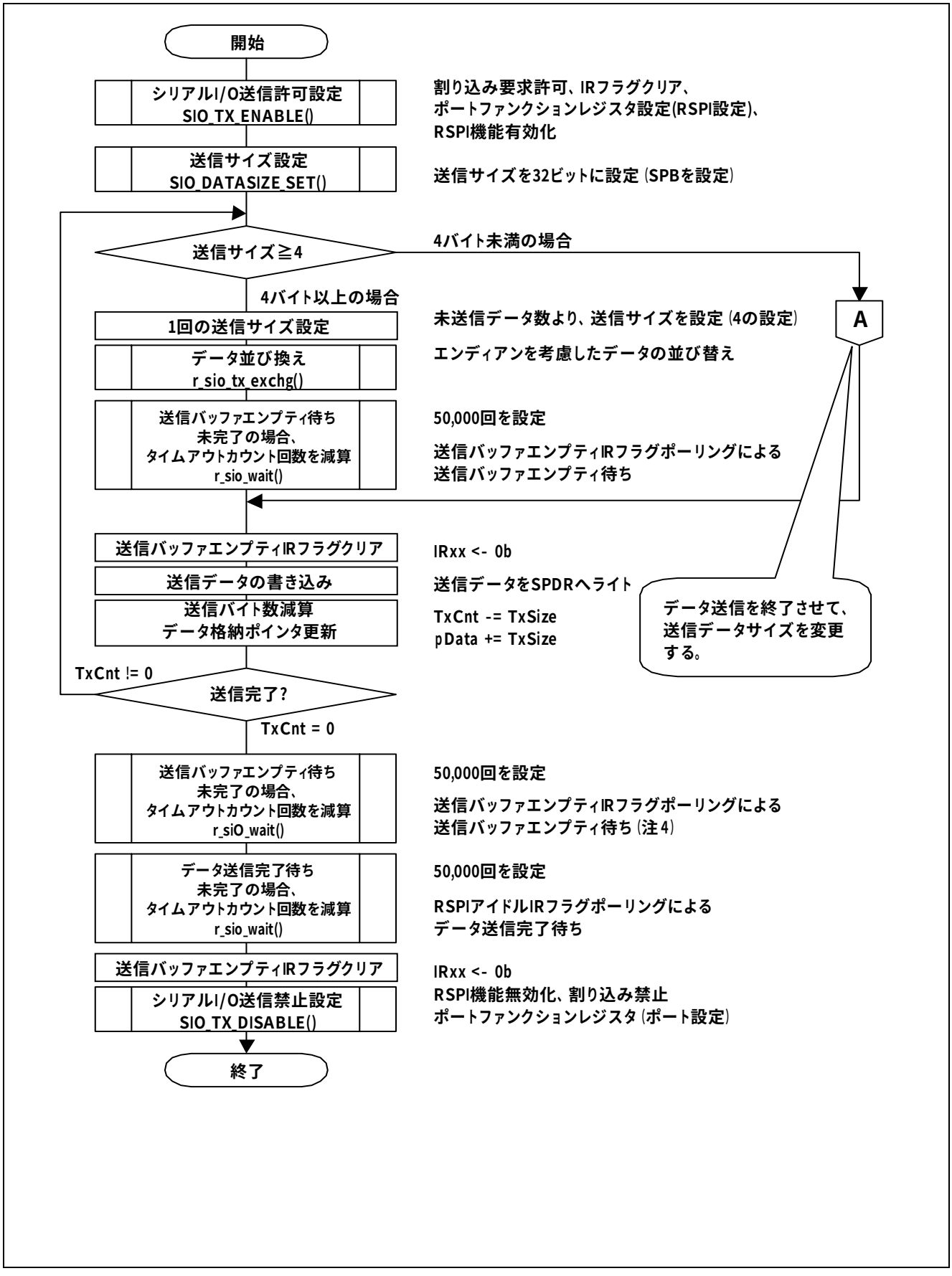


図 6-9　シリアル I/O データ送信処理概要 1

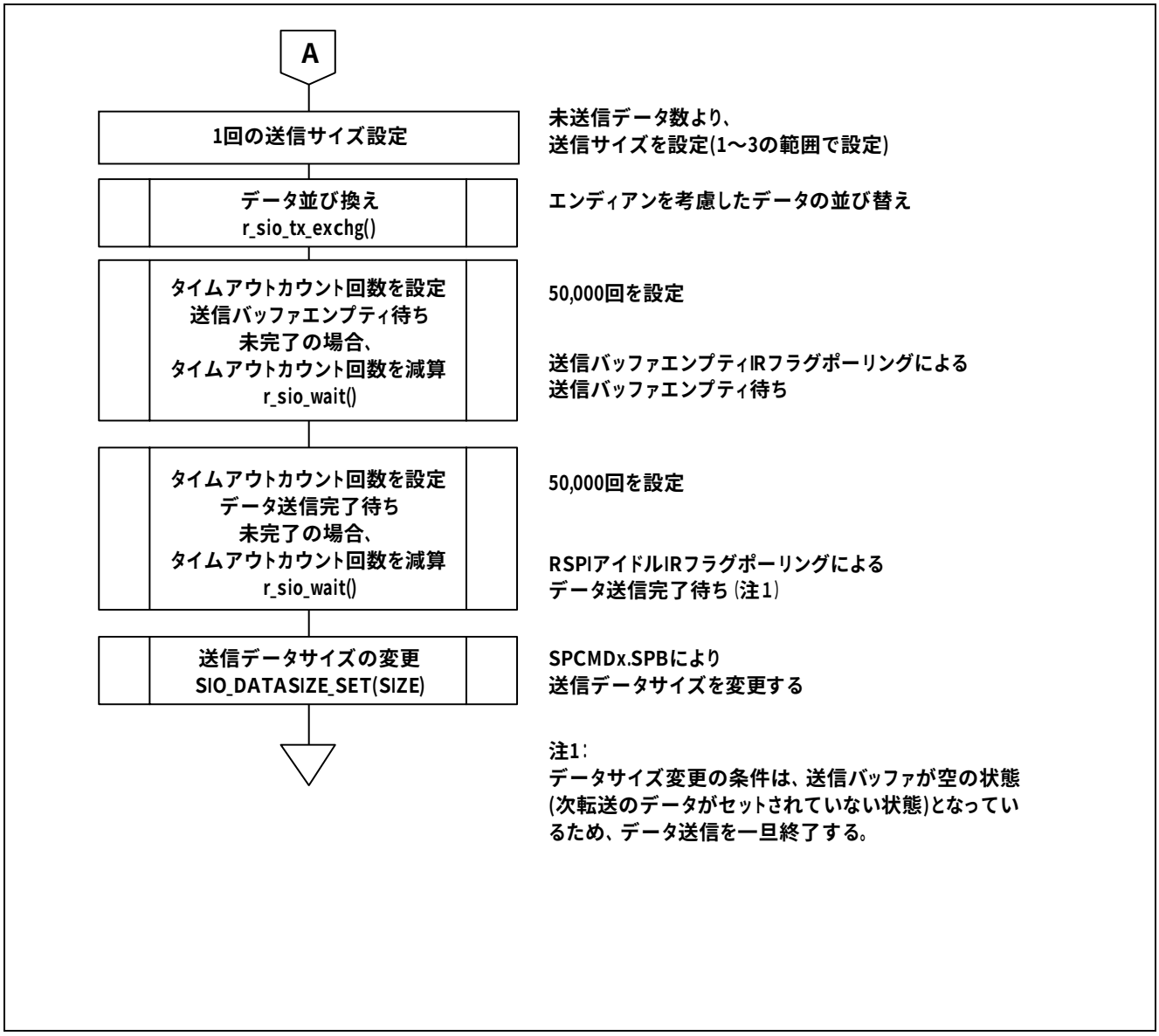


図 6-10　**シリアル** I/O **データ送信処理概要** 2

6.9.6　シリアルI/Oデータ受信処理

R_SIO_Rx_Data

概要	**シリアル** I/O **データ受信処理**
ヘッダ	R_SIO.h, R_SIO_rspi.h, mtl_com.h
宣言	error_t R_SIO_Rx_Data(uint16_t RxCnt, uint8_t FAR* pData)
説明	**・指定バイト数分でデータを受信し、**pData **に格納します。** **・本関数コール前に、シリアル** I/O **許可設定処理を実行してください。** **・本関数の実行の結果、異常終了であれば、シリアル** I/O **禁止設定処理を実行してください。** **・「通常受信」と「高速受信」を選択することができます。選択方法は**7.2.2 R_SIO_rspi.h (1) **使用する動作モードの定義　を参照してください。** **・図** 6-11　**シリアル** I/O **データ受信処理概要** 1**（通常受信）、図** 6-12　**シリアル** I/O **データ受信処理概要** 2**（通常受信）に「通常受信」の概要を示します。** **・図** 6-13　**シリアル** I/O **データ受信処理概要** 3**（高速受信）、図** 6-14　**シリアル** I/O **データ受信処理概要** 4**（高速受信）、図** 6-15　**シリアル** I/O **データ受信処理概要** 5**（高速受信）に「高速受信」の概要を示します。**
引数	uint16_t　RxCnt　;　**受信バイト数** uint8_t FAR*　pData　;　**受信データ格納バッファポインタ**
リターン値	SIO_OK　;　Successful operation SIO_ERR_HARD　;　Hardware error
備考	**・半二重での受信処理の場合に使用してください。** **・ハードウェアマニュアル記載の初期化フローチャートにしたがって、以下を実行します。（インライン関数** SIO_TRX_ENABLE()**）** **①**SPCMD **設定** **②**IR **フラグクリア** **③ポートファンクションレジスタ（**PFxSPI**）設定**(RSPI **端子の有効化**) **④**SPCR **設定（送受信許可設定）** **⑤**SPCR **をリード** **・受信完了後、上記受信許可設定の逆の処理を行うことで、シリアル通信を停止します。（インライン関数** SIO_TRX_DISBLE()**）** **①**SPCR **設定（送受信停止設定）** **②**SPCR **をリード** **③ポートファンクションレジスタ（**PFxSPI**）設定（**RSPI **端子の無効化）** **・継続使用しない場合、シリアル** I/O **禁止設定処理を実行することを推奨します。** **・「高速受信」の場合、以下の処理が追加されます。** **①連続受信中のオーバランエラー（注１）の発生を防ぐため、次のダミーデータ書き込み直前から、前の受信データが取り出されるまでの期間は、割り込みを禁止します。プロセッサ割り込み優先レベル**(IPL[3:0])**を最高位に設定することで、割り込み禁止状態を実現しています。** **②連続受信** 3 **回目以降のダミーデータ書き込みは、データ受信が完了した後に行います。これにより、連続受信中でも他の割り込みを受け付けることができます。** **注１：他アプリケーションでの** DMAC/EXDMAC/DTC **転送と本受信で使用するバスが競合した場合や優先度の高い** NMI **割り込みが発生した場合、オーバランエラーが発生することがあります。**

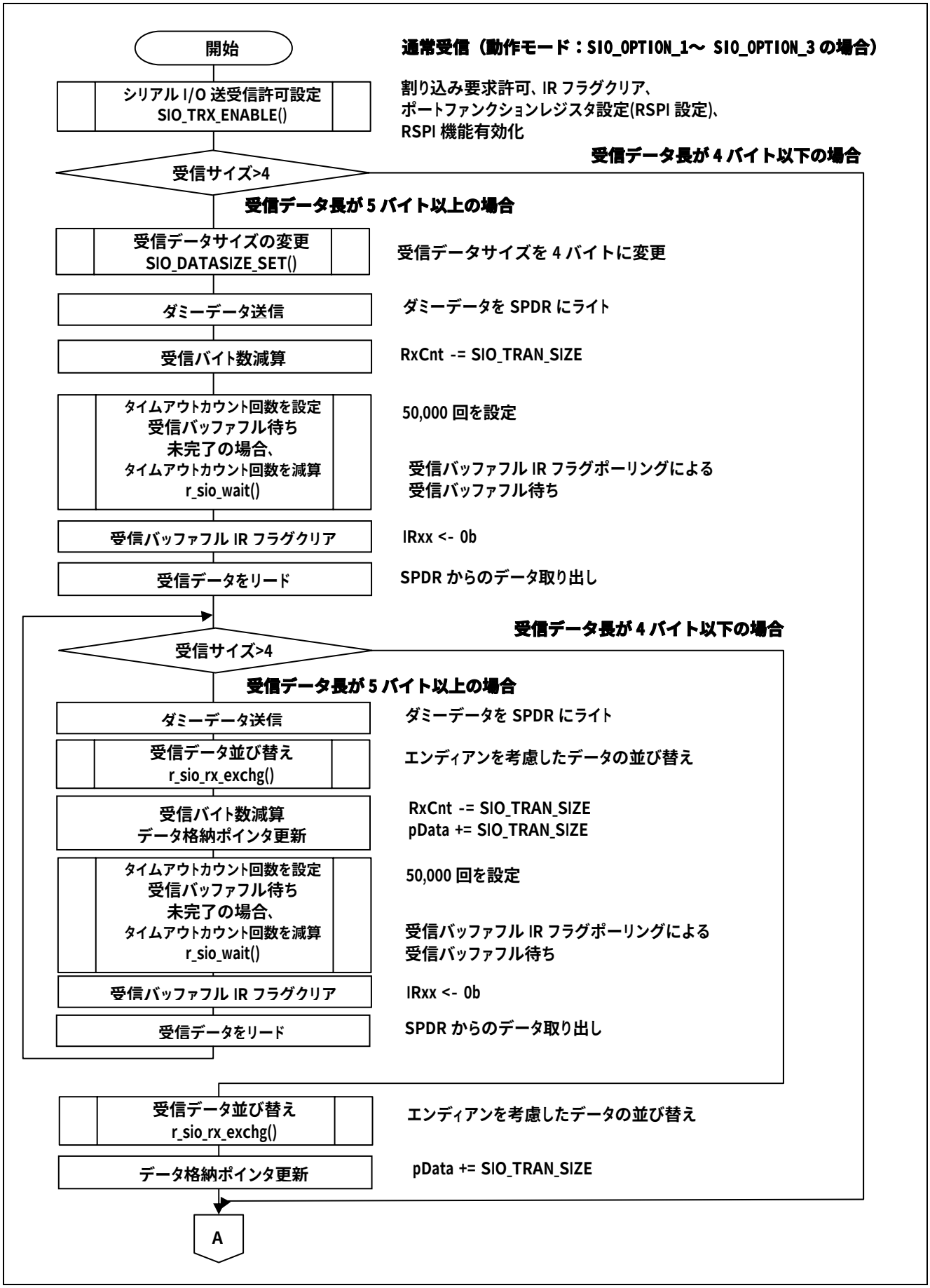


図 6-11　シリアル I/O データ受信処理概要 1（通常受信）

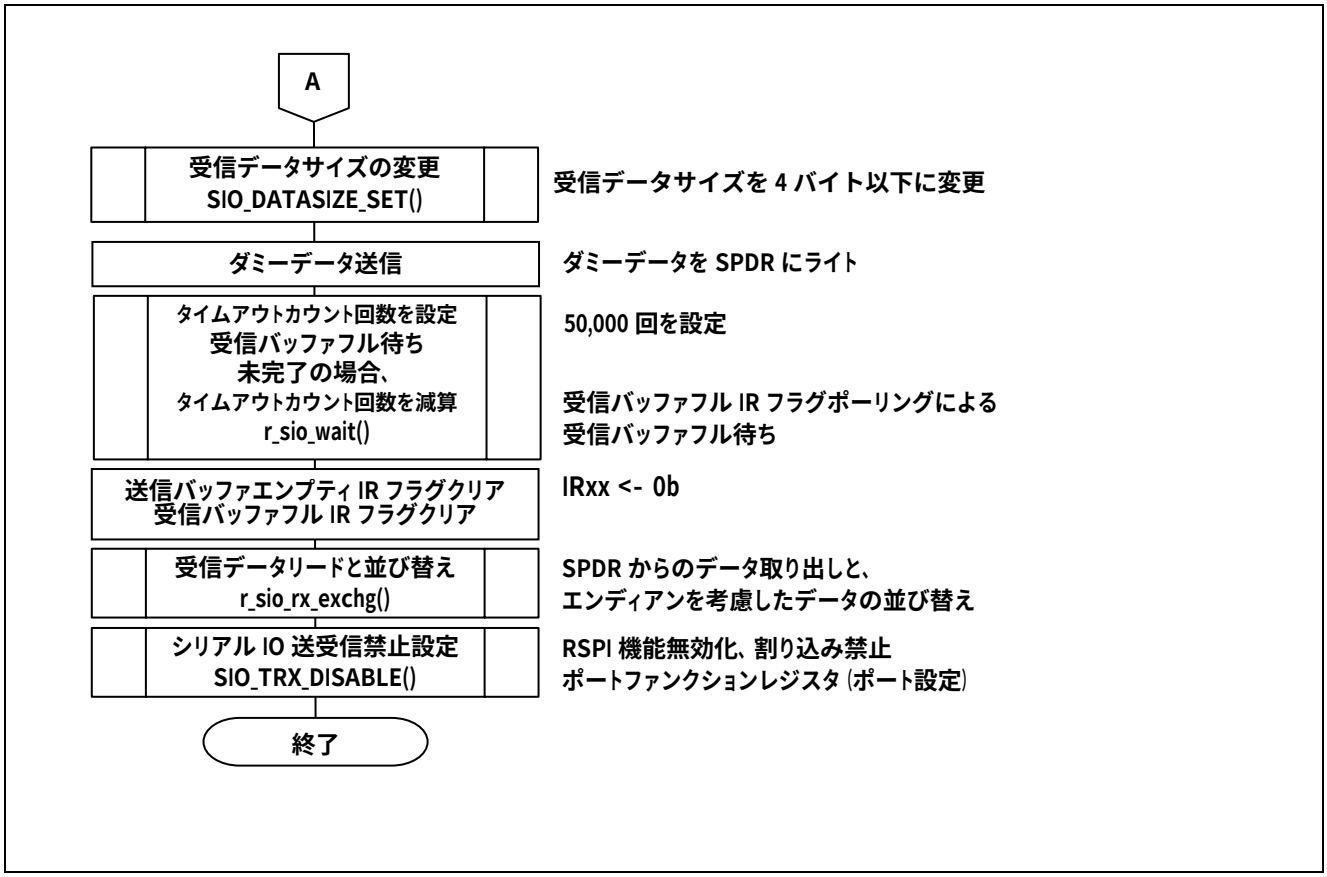


図 6-12　**シリアル** I/O **データ受信処理概要** 2**（通常受信）**

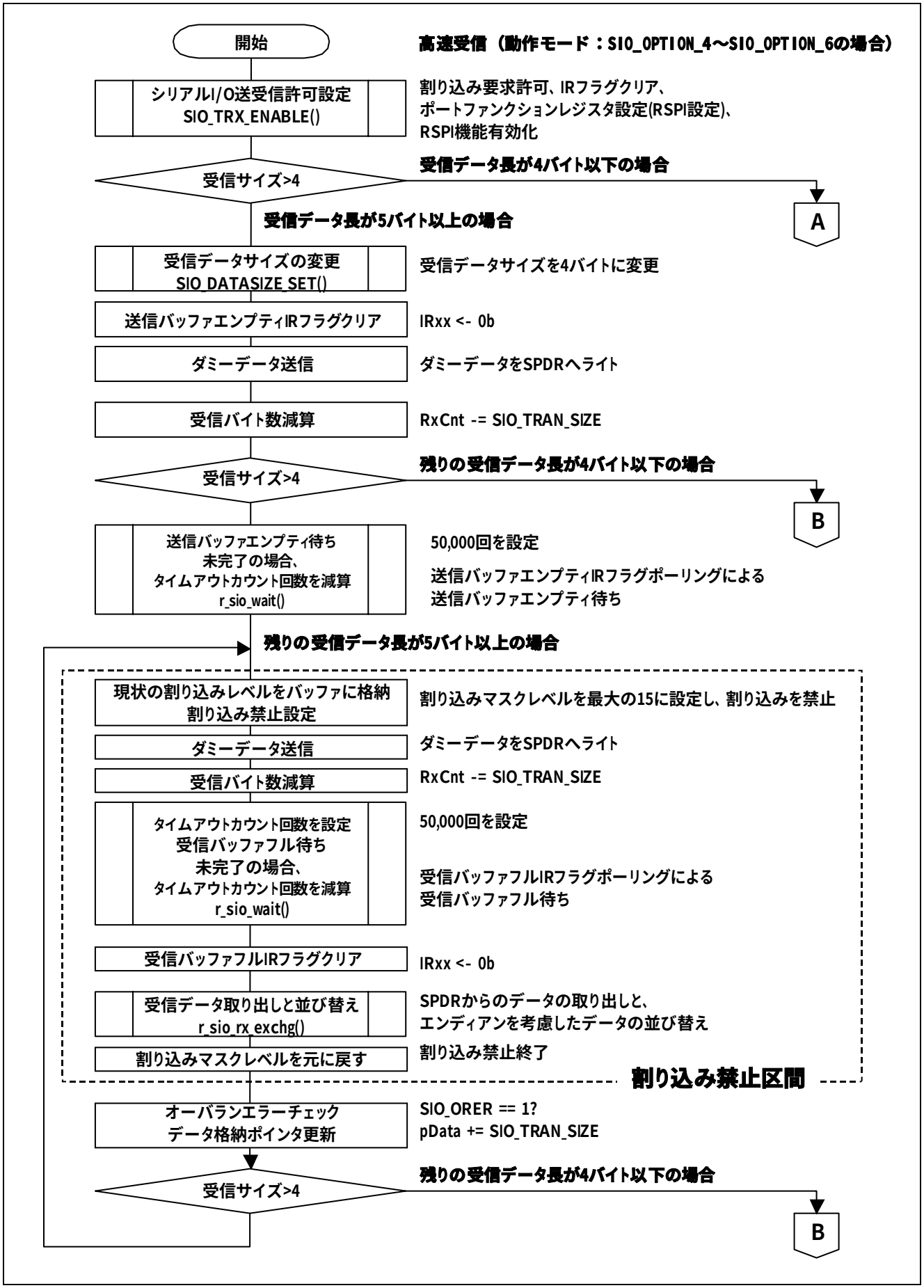


図 6-13　シリアル I/O データ受信処理概要 3（高速受信）

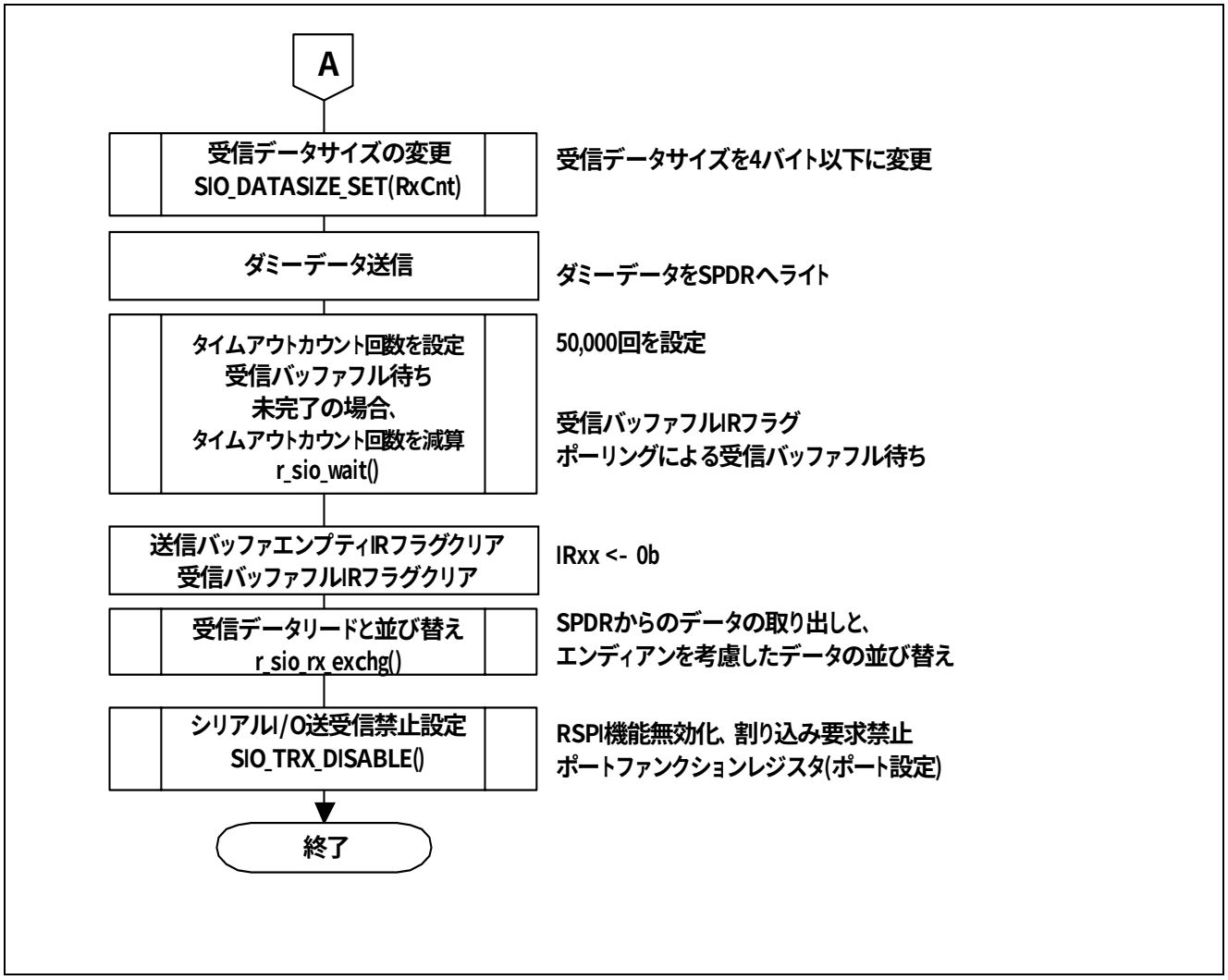


図 6-14　シリアル I/O データ受信処理概要 4（高速受信）

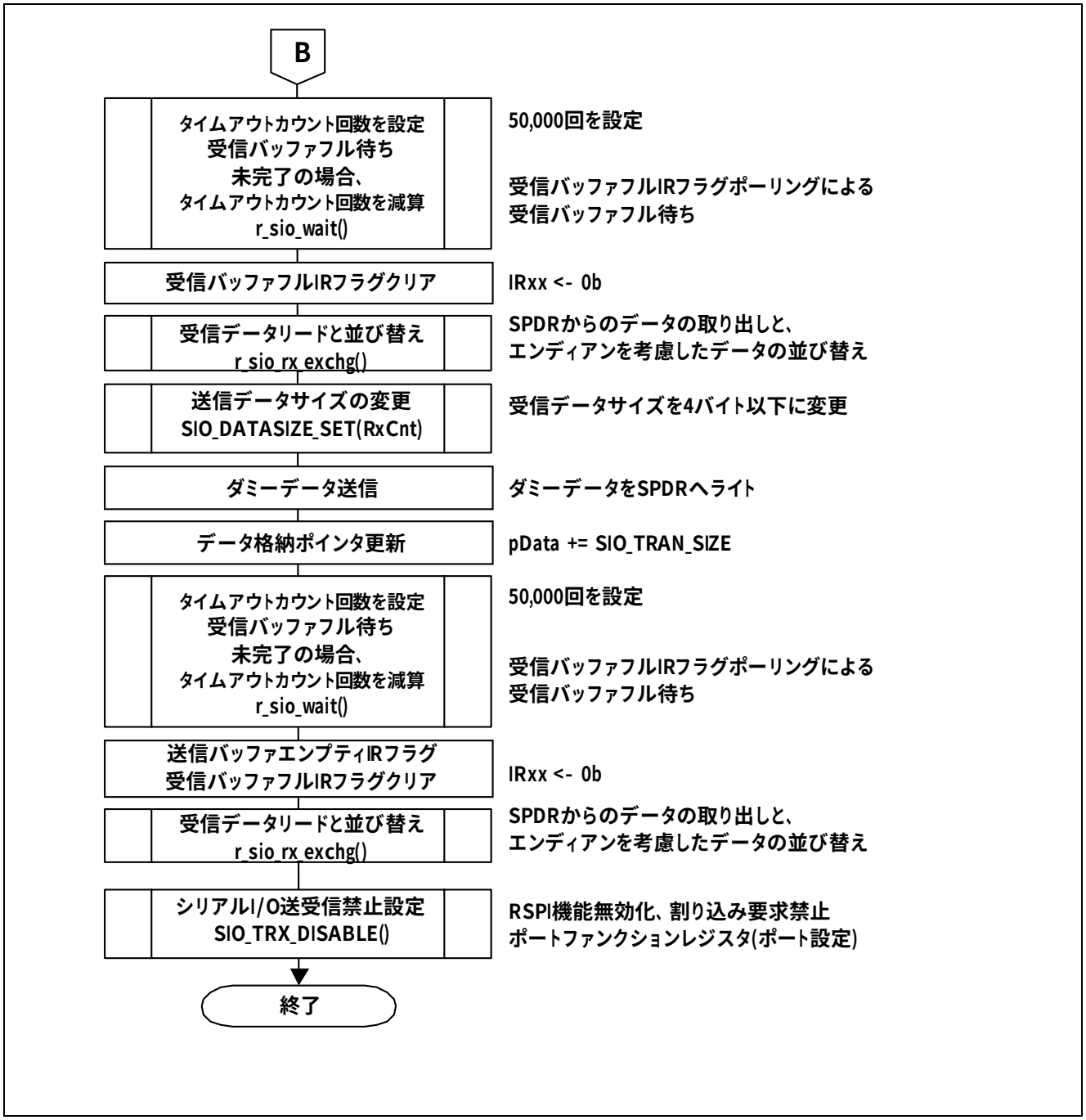


図 6-15　**シリアル** I/O **データ受信処理概要** 5**（高速受信）**

6.10　インライン関数仕様

以下に、本サンプルコードで使用するインライン関数を示します。

6.10.1　SIO_IO_INIT()

(1)　**目的**

当該端子の RSPI 機能を無効化し、入力端子をポート入力状態、出力端子をポート出力状態にします。

(2)　**機能**

当該端子の RPSI 機能を無効化して、DataIn 端子をポート入力状態、DataOut 端子と CLK 端子をポート出力状態にします。

以下の処理を実現します。必要に応じて、処理を見直してください。

①RSPI を制御するポートファンクションレジスタ(PFxSPI)により、各端子の RPSI 機能を無効にする。

・PFxSPI. <- 00h：RSPI 機能無効化(ポート許可設定)

②DataIn 端子をポート入力に設定する。

SIO_DATAI_INIT()を参照してください。

③DataOut 端子をポート”H”出力に設定する。

SIO_DATAO_INIT()を参照してください。

④CLK 端子をポート”H”出力に設定する。

SIO_CLK_INIT()を参照してください。

(3)　**備考**

本インライン関数により、端子を周辺機能からポート機能に変更します。本関数を呼び出す前に、他の周辺機能を使用していないか確認してから実行するようにしてください。

6.10.2　SIO_IO_OPEN()

(1) **目的**

入力端子と出力端子をポート入力状態にします。

(2) **機能**

DataIn 端子と DataOut 端子と CLK 端子入力端子をポート入力状態にします。

以下の処理を実現します。必要に応じて、処理を見直してください。

①DataIn 端子をポート入力に設定する。

SIO_DATAI_INIT()を参照してください。

②DataOut 端子をポート入力に設定する。

SIO_DATAO_OPEN()を参照してください。

③CLK 端子をポート入力に設定する。

SIO_CLK_OPEN()を参照してください。

(3) **備考**

リムーバブルメディアの挿入前および抜去前での、全端子の Hi-z 化をするために使用してください。

SIO_IO_INIT()を実行後に、本関数を実行してください。

6.10.3　SIO_DATAI_INIT()

(1) **目的**

DataIn 端子をポート入力状態にします。

(2) **機能**

以下の処理を実現します。必要に応じて、処理を見直してください。

①入力バッファコントロールレジスタ(ICR)を使って、DataIn 端子の入力バッファ機能を有効にする。

・DataIn 端子 ICR <- 1b：入力バッファ有効

②プルアップ抵抗コントロールレジスタ(PCR)により、DataIn 端子の入力プルアップ抵抗を無効にする。（※１）

・DataIn 端子 PCR <- 0b：入力プルアップ無効（※２)

③データディレクションレジスタ(DDR)を使って、DataIn 端子をポート入力に設定する。

・DataIn 端子 DDR <- 0b：入力ポート

(3) **備考**

※１：プルアップ抵抗コントロールレジスタ(PCR)の設定が可能な端子は、Port9,PortA～PortE 及び PortG です。PortA、PortC もしくは PortE を使用する場合は、SIO_PCR_DATAI を設定してください。

※２：必要に応じて設定してください。

6.10.4　SIO_DATAO_INIT()

(1) **目的**

DataOut 端子をポート”H”出力にします。

(2) **機能**

以下の処理を実現します。必要に応じて、処理を見直してください。

①オープンドレインコントロールレジスタ(ODR)を使って、DataOut 端子の出力形態を CMOS 出力に設定する。（※１）

・DataOut 端子 ODR <- 0b：CMOS 出力（※３)

②入力バッファコントロールレジスタ(ICR)を使って、DataOut 端子の入力バッファ機能を無効にする。（※２）

・DataOut 端子 ICR <- 0b：入力バッファ無効

③データレジスタ(DR)を使って、DataOut 端子を H 出力にする。

・DataOut 端子 DR <- 1b：H 出力

④データディレクションレジスタ(DDR)とデータレジスタ(DR)を使って、DataOut 端子をポート出力に設定する。

・DataOut 端子 DDR <- 1b：出力ポート

・DataOut 端子 DR <- 1b：H 出力

(3) **備考**

※１：オープンドレインコントロールレジスタ(ODR) の設定が可能な端子は、Port0～Port3 及び PortC です。Port2 もしくは PortC を使用する場合は、SIO_ODR_DATAO 及び SIO_ODR_CLK を設定してください。

※２：端子を出力端子として使用する場合、入力バッファコントロールレジスタ(ICR)の設定で入力バッファ機能を有効にすると、出力データが端子状態として取り込まれます。したがって、出力として使用する端子は、ICR の設定で入力バッファ機能を無効にする必要があります。

※３：必要に応じて設定してください。

6.10.5　SIO_DATAO_OPEN()

(1) **目的**

DataOut 端子をポート入力状態にします。

(2) **機能**

以下の処理を実現します。必要に応じて、処理を見直してください。

①データディレクションレジスタ(DDR)を使って、DataOut 端子をポート入力に設定する。

・DataOut 端子 DDR <- 0b：入力ポート（入力バッファ無効）

(3) **備考**

なし

6.10.6　SIO_CLK_INIT()

(1) **目的**

CLK 端子をポート”H”出力にします。

(2) **機能**

以下の処理を実現します。必要に応じて、処理を見直してください。

①オープンドレインコントロールレジスタ(ODR)を使って、CLK 端子の出力形態を CMOS 出力に設定する。（※１）

・CLK 端子 ODR <- 0b：CMOS 出力（※３)

②入力バッファコントロールレジスタ(ICR)を使って、CLK 端子の入力バッファ機能を無効にする。（※２）

・CLK 端子 ICR <- 0b：入力バッファ無効

③データレジスタ(DR)を使って、CLK 端子を H 出力にする。

・CLK 端子 DR <- 1b：H 出力

④データディレクションレジスタ(DDR)とデータレジスタ(DR)を使って、CLK 端子をポート出力に設定する。

・CLK 端子 DDR <- 1b：出力ポート

・CLK 端子 DR <- 1b：H 出力

(3) **備考**

※１：オープンドレインコントロールレジスタ(ODR) の設定が可能な端子は、Port0～Port3 及び PortC です。Port2 もしくは PortC を使用する場合は、SIO_ODR_DATAO 及び SIO_ODR_CLK を設定してください。

※２：端子を出力端子として使用する場合、入力バッファコントロールレジスタ(ICR)の設定で入力バッファ機能を有効にすると、出力データが端子状態として取り込まれます。したがって、出力として使用する端子は、ICR の設定で入力バッファ機能を無効にする必要があります。

※３：必要に応じて設定してください。

6.10.7　SIO_CLK_OPEN()

(1) **目的**

CLK 端子をポート入力状態にします。

(2) **機能**

以下の処理を実現します。必要に応じて、処理を見直してください。

①データディレクションレジスタ(DDR)を使って、CLK 端子をポート入力に設定する。

・CLK 端子 DDR <- 0b：入力ポート（入力バッファ無効）

(3) **備考**

なし

6.10.8　SIO_ENABLE()

(1) **目的**

シリアル I/O を初期化し、機能を有効化します。ただし、送信許可／受信許可／送受信許可にするまでの共通処理を実行します。また、ビットレートを設定します。

(2) **機能**

ハードウェアマニュアにしたがって、シリアル I/O を初期化します。必要に応じて、処理を見直してください。

RX62N の場合、以下の処理を行います。

①モジュールストップコントロールレジスタ(MSTPCRB)を使って、モジュールストップ解除状態に設定する。

・MSTPCRB　MSTPBxx <- 0b：モジュールストップ状態解除、RSPI レジスタへのリード/ライト可能化

・MSTPCRB　MSTPBxx をリード

②送信設定と送受信設定の有効化手順において、共通の処理を行います。

送信設定と送受信設定の共通部分の以下を設定します。

・SPPCR <- 30h：通常モード、CMOS 出力、MOSI アイドル固定値 1

・SPBR ビットレート設定

・SPDCR <- 20h：設定 1-1、SSL0-3 出力（※１）、受信バッファを読み出す、ロングワードアクセス

・SPCKD <- 00h：SPCKD 遅延値設定(初期値)（※２）

・SSLND <- 00h：SSL ネゲート遅延値(初期値)（※２）

・SPND <- 00h：次アクセス遅延値(初期値)（※２）

・SPCR2 <- 00h：パリティ機能無効、アイドル割り込み要求禁止

・SPSR OVRF/MODF/PERF クリア

SIO_SPSR_CLEAR()を参照してください。

・SPCR <- 09h：3 線式、マスタモード、送信割り込み要求禁止、RSPI 機能無効、受信割り込み要求禁止

・SPSCR <- 00h：シーケンス長 SPCMD0 のみ使用

(3) **備考**

ビットレート設定後、待ちを必要とするシリアル I/O の場合、インライン関数終了後、待ち処理を入れてください。

SIO_DISABLE()と対となるものです。本関数を実行した場合、SIO_DISABLE()を実行して、処理を終了してください。

本関数実行前には、SIO_DISABLE()/SIO_TX_DISABLE()/SIO_TRX_DISABLE()（SPCR 制御による通信動作停止）のいずれかを実行し、通信動作を停止させてください。

※１：3 線式制御のため、SSL 機能を使用していません。SSL 端子は他機能に割り当てることが可能であるため、SSL0 以外の SSL 端子を入出力に設定し、他機能に割り当てています。

※２：本サンプルコードでは使用しません。

6.10.9　SIO_DISABLE()

(1) **目的**

シリアル I/O 機能を無効化します。

(2) **機能**

シリアル I/O を無効化します。送信設定／送受信設定の無効化手順において、共通の処理を行います。必要に応じて、処理を見直してください。

RX62N の場合、以下の処理を行います。

①RSPI 関連のレジスタ設定を行うため、モジュールストップコントロールレジスタ(MSTPCRB)を使って、モジュールストップ解除状態にする（※１）

・MSTPCRB　MSTPBxx <- 0b：モジュールストップ解除、RSPI レジスタへのリード/ライト可能化

・MSTPCRB　MSTPBxx をリード

②RSPI 機能を無効化

・SPCR <- 09h：3 線式、マスタモード、送信割り込み要求禁止、RSPI 機能無効、受信割り込み要求禁止

③SPSR OVRF/MODF/PERF　クリア

SIO_SPSR_CLEAR()を参照してください。

④モジュールストップコントロールレジスタ(MSTPCRB)を使って、モジュールストップ状態に設定

・MSTPCRB　MSTPBxx <- 1b：モジュールストップ許可、RSPI レジスタへのリード/ライト不可 (RSPI レジスタの状態は保持)

・MSTPCRB　MSTPBxx をリード

(3) **備考**

SIO_ENABLE()と対となるものです。SIO_ENABLE()を実行した場合、本関数を実行して、処理を終了してください。

※１：RX62N の場合、モジュールストップ状態に設定されたモジュールのレジスタは、読み出し、書き込みともにできません。本インライン関数では SPCR により RSPI 機能の無効化を行うため、一度モジュールストップを解除し、SPCR を設定後、モジュールストップを有効にする仕様としています。尚、モジュールストップ状態では、レジスタの値は保持されます。

6.10.10　SIO_DATASIZE_SET()

(1) **目的**

SPCMD0-7 の SPB[3:0]を設定します。

(2) **機能**

データ長（8/16/24/32 ビット）を設定します。

(3) **備考**

なし

6.10.11　SIO_TX_ENABLE()

(1)　**目的**

シリアル I/O を送信許可にします。

(2)　**機能**

ハードウェアマニュアにしたがって、シリアル I/O を送信許可設定します。端子をポート機能からシリアル I/O 機能への切り替え後、送信許可設定します。必要に応じて、処理を見直してください。

本処理では、SIO_ENABLE()の後の初期化手順から、送信設定専用の初期化処理を行います。

RX62N の場合、以下の処理を行います。

①SPCR2 設定（RSPI アイドル割り込み要求許可設定）

・SPCR2 <- 04h：パリティ機能無効、アイドル割り込み要求許可（送信完了検出目的）

②SPCMD 設定

・SPCMD0 <- 0203h：CPHA=1、CPOL=1、ベースのビットレート、SSL0（※１）、転送終了時 SSL 信号ネゲート（※２）、32 ビット、MSB ファースト、（※３）

③IR フラグをクリア

SIO_IR_CLEAR()を参照してください。

④使用端子を RSPI 機能に設定

ハードウェアマニュアルのポートファンクションレジスタの設定方法（※４）に従い、以下の順番でレジスタの設定を行う。

1) PFxSPI(x=G、H)の RSPI 端子選択ビット(RSPIS)により、RSPI で使用する端子を設定する。

・PFxSPI.RSPIS <- 0b or 1b：RSPI で使用する端子を選択

2) PFxSPI(x=G、H)の RSPI 出力許可端子を有効にする。

・PFxSPI |= 0Eh　：RSPCK 端子有効、MOSI 端子有効、MISO 端子有効

⑤SPCR 設定（送信許可設定）

SPCR.TXMD, SPTIE, SPE をセットし、送信を許可する。

・SPCR <- 6Bh：3 線式、送信動作のみ、マスタモード、送信割り込み要求許可、RSPI 機能有効

⑥SPCR をリード

(3)　**備考**

SIO_TX_DISABLE()と対となるものです。本関数を実行した後は、SIO_TX_DISABLE()を実行して、処理を終了してください。

※１：3 線式制御のため、SSL 機能を使用しない。SSL 端子は他機能に割り当てることが可能であるため、SSL0 以外の SSL 端子を入出力に設定し、他機能に割り当てています。

※２：SSL 機能未使用のため、設定は無視されます。

※３：SPCMD0 b15-b13 は、SPCKD,SSLND,SPND 未使用のため、0b を設定してください。

※４：RSPI のポートファンクションレジスタ(PFxSPI)は入出力先を変更する端子選択ビットと、端子機能を有効にする許可ビットの両方が存在します。そのため、端子選択ビットで端子の入出力先を設定した後に、許可ビットで端子機能を有効にする設定にしています。

6.10.12　SIO_TX_DISABLE()

(1)　**目的**

シリアル I/O の送信機能を停止します。

(2)　**機能**

SIO_TX_ENABLE()の処理の逆手順により、送信機能を停止します。送信停止設定処理後、端子をシリアル I/O 機能からポート機能へ切り替えます。必要に応じて、処理を見直してください。

RX62N の場合、以下の処理を行います。

①SPCR 設定（送受信停止設定）

SPCR.TXMD, SPTIE, SPE, SPRIE をクリアし、送受信を停止する。

・SPCR <-09h　　：マスタモード、送信割り込み要求禁止、RSPI 機能無効、受信割り込み要求禁止

②SPCR をリード

③使用端子をポート機能に設定

ポートファンクションレジスタ(PFxSPI)の RSPI 出力許可端子を無効化する。

・PFxSPI <- 00h：RSPCK 端子無効、MOSI 端子無効、MISO 端子無効

④SPCR2 設定（RSPI アイドル割り込み要求禁止設定）

・SPCR2 <- 00h　：パリティ機能無効、アイドル割り込み要求禁止

(3)　**備考**

SIO_TX_ENABLE()と対となるものです。SIO_TX_ENABLE()を実行した後は、本関数を実行して、処理を終了してください。

6.10.13　SIO_TRX_ENABLE()

(1)　**目的**

シリアル I/O を送受信許可にします。

(2)　**機能**

ハードウェアマニュアにしたがって、シリアル I/O を送信許可設定します。端子をポート機能からシリアル I/O 機能への切り替え後、送受信許可設定します。必要に応じて、処理を見直してください。

本処理では、SIO_ENABLE()の後の初期化手順から、送信設定専用の初期化処理を行います。

RX62N の場合、以下の処理を行います。

①SPCMD 設定

・SPCMD0 <- 0203h：CPHA=1、CPOL=1、ベースのビットレート、SSL0（※１）、転送終了時 SSL 信号ネゲート（※２）、32 ビット、MSB ファースト、（※３）

②IR フラグをクリア

SIO_IR_CLEAR()を参照してください。

③使用端子を RSPI 機能に設定

ハードウェアマニュアルのポートファンクションレジスタ(PFxSPI)の設定方法（※４）に従い、以下の順番でレジスタの設定を行う。

1) PFxSPI(x=G、H)の RSPI 端子選択ビット(RSPIS)により、RSPI で使用する端子を設定する。

・PFxSPI.RSPIS <- 0b or 1b：RSPI で使用する端子を選択

2) PFxSPI(x=G、H)の RSPI 出力許可端子を有効にする。

・PFxSPI |= 0Eh　：RSPCK 端子有効、MOSI 端子有効、MISO 端子有効

④SPCR 設定（送受信許可設定）

SPCR.SPTIE, SPE, SPRIE をセットし、送受信を許可する。

・SPCR <- E9h：3 線式、全二重同期式、マスタモード、送信/受信割り込み要求許可、RSPI 機能有効

⑤SPCR をリード

(3)　**備考**

SIO_TRX_DISABLE()と対となるものです。本関数を実行した後は、SIO_TRX_DISABLE()を実行して、処理を終了してください。

※１：3 線式制御のため、SSL 機能を使用しない。SSL 端子は他機能に割り当てることが可能であるため、SSL0 以外の SSL 端子を入出力に設定し、他機能に割り当てています。

※２：SSL 機能未使用のため、設定は無視されます。

※３：SPCMD0 b15-b13 は、SPCKD,SSLND,SPND 未使用のため、0b を設定してください。

※４：RSPI のポートファンクションレジスタ(PFxSPI)は入出力先を変更する端子選択ビットと、端子機能を有効にする許可ビットの両方が存在します。そのため、端子選択ビットで端子の入出力先を設定した後に、許可ビットで端子機能を有効にする設定にしています。

6.10.14　SIO_TRX_DISABLE()

(1)　**目的**

シリアル I/O の送受信機能を停止します。

(2)　**機能**

SIO_TRX_ENABLE()の処理の逆手順により、送受信機能を停止します。送受信停止設定処理後、端子をシリアル I/O 機能からポート機能へ切り替えます。必要に応じて、処理を見直してください。

RX62N の場合、以下の処理を行います。

①SPCR 設定（送受信停止設定）

SPCR.TXMD, SPTIE, SPE, SPRIE をクリアし、送受信を停止する。

・SPCR <-09h　　：マスタモード、送信割り込み要求禁止、RSPI 機能無効、受信割り込み要求禁止

②SPCR をリード

③使用端子をポート機能に設定

ポートファンクションレジスタ(PFxSPI)の RSPI 出力許可端子を無効化する。

・PFxSPI <- 00h：RSPCK 端子無効、MOSI 端子無効、MISO 端子無効

(3)　**備考**

SIO_TRX_ENABLE()と対となるものです。SIO_TRX_ENABLE()を実行した後は、本関数を実行して、処理を終了してください。

6.10.15　SIO_SPSR_CLEAR()

(1)　**目的**

SPSR のエラーフラグをクリアします。

(2)　**機能**

OVRF フラグ、MODF フラグ、PERF フラグをクリアします。

①フラグ＝1 であれば、0 クリアする。

②フラグをリードし、0 であることを確認する。

(3)　**備考**

なし

6.10.16　SIO_IR_CLEAR()

(1)　**目的**

IR フラグをクリアします。

(2)　**機能**

以下の手順でフラグのクリアを行います。必要に応じて、処理を見直してください。

RX62N の場合、以下の処理を行います。

①IR フラグをクリア

(3)　**備考**

なし

7. 応用例

シリアル I/O 制御部分の設定例を示します。

使用する上での設定例を以下に示します。

設定箇所は、各ファイル中の「/** SET **/」というコメントの部分です。

7.1　mtl_com.h（共通ヘッダファイル）

共通で使用される共通関数のヘッダです。

mtl_com.h.XXX（mtl_com.h.common を除く）は、MCU 毎に作成したものです。どれか一つを mtl_com.h にリネームして使用してください。対象 MCU のものが無い場合には、参照して、mtl_com.h を作成してください。

(1)　OSのヘッダファイル定義

本サンプルコードは、OS のシステムコールの設定は使用していません。

下記の例は、OS を使用しない場合の例です。

本サンプルコードでは、使用しない設定にしてください。他のソフトウェアに依存します。

```
/* システムコールを使用するため、                                          */
/* プロトタイプ宣言のある OS のヘッダファイルをインクルードしてください。          */
/* OS を使用しない場合は、下記デファインとインクルードをコメントにしてください。    */
//#define MTL_OS_USE                    /* Use OS                        */
//#include <RTOS.h>                     /* OS header file                */
//#include "mtl_os.h"
```

(2)　共通アクセス領域を定義したヘッダファイル定義

MCU の機能レジスタの定義がされているヘッダファイルをインクルードします。

主にデバイスドライバがポート制御等に使用するため、インクルードする必要があります。

MCU に合わせて、ヘッダファイルをインクルードしてください。

下記の例は、RX62N のヘッダファイルをインクルードする場合の例です。

本サンプルコード使用時は、インクルードしてください。

```
/* MCU の SFR 領域のデファイン値を使用しているため、                        */
/* I/O 周りのデファイン定義のあるヘッダファイルをインクルードしてください。      */
#include "iodefine.h"                       /* definition of MCU SFR     */
```

(3)　ループタイマの定義

ソフトウェア・ループタイマを使用する場合、以下のヘッダをインクルードします。

主にデバイスドライバが待ち時間を確保するために、使用します。

ソフトウェア・ループタイマを使用しない場合は、下記インクルードをコメントにしてください

下記の例は、ソフトウェア・ループタイマを使用する場合の例です。

本サンプルコード使用時は、インクルードしてください。

```
/* ループタイマを使用しない場合は、下記インクルードをコメントにしてください。    */
#include "mtl_tim.h"
```

(4) エンディアンタイプ定義

リトルエンディアン／ビッグエンディアンのどちらかの指定が可能です。

下記は、ビッグエンディアンに設定した場合の例です。

```
/* MCU が、(1)SuperH でリトルエンディアン設定、(2)M16C  の場合、定義を有効にしてください。*/
/* その他の MCU を指定する時はリトルエンディアンの定義をコメントにしてください。          */
//#define MTL_MCU_LITTLE                    /* Little Endian                  */
```

(5) エンディアン処理の高速化の定義

mtl_end.c の処理の高速化指定が可能です。M16C を使用する場合、高速になります。

RX ファミリの場合は、コメントアウトし、定義しないでください。

```
/* M16C を使用する場合、定義を有効にしてください。                                  */
/* mtl_endi.c の処理の高速化が可能です。                                          */
//#define MTL_ENDI_HISPEED             /* Uses the high-speed function.    */
```

(6) 使用する標準ライブラリのタイプの定義

使用する標準ライブラリのタイプを定義してください。

下記に示す処理をコンパイラ添付のライブラリで使用する場合は、下記デファイン定義をコメントにしてください。

下記の例は、コンパイラ添付のライブラリを使用する場合例です。

```
/* 使用する標準ライブラリのタイプを指定してください。                               */
/* 下記に示す処理をコンパイラ添付のライブラリで使用する場合は、                     */
/* 下記デファイン定義をコメントにしてください。                                     */
/* memcmp() / memmove() / memcpy() / memset() / strcat() / strcmp() / strcpy() /
strlen()              */
//#define MTL_USER_LIB                    /* use optimized library          */
```

(7) アクセスするRAM領域の定義

使用する RAM 領域を定義してください。

標準関数や一部の処理に効率の良い処理を適用します。

RX62N 場合は、MTL_MEM_NEAR を定義してください。

```
/* 使用する処理群がアクセスする RAM 領域を定義してください。                       */
/* 標準関数や一部の処理に効率の良い処理を適用します。                               */
//#define MTL_MEM_FAR          /* Supports Far RAM area of M16C/60          */
#define MTL_MEM_NEAR           /* Supports Near RAM area.       (Others)    */
```

7.1.2　mtl_tim.h

mtl_com.h にて、ループタイマを定義した場合に、インクルードされます。

使用する MCU、クロック、コンパイルオプション等に依存します。

また、命令キャッシュを有効にしているシステムでは、キャッシュ内に、ループタイマ処理が格納されている場合を想定して、設定してください。

使用環境に応じて、測定し直してください。

```
/* タイマのカウンタ値を定義してください。                                    */
/* MCU 及びクロック、ウェイトに応じて設定してください。                      */
#if 1
/* Setting for 12MHz no wait Ix8 = 96MHz(Compile Option "-optimize=2",com.V406R00)*/
#define MTL_T_1US                          10     /* loop Number of   1us      */
#define MTL_T_2US                          20     /* loop Number of   2us      */
#define MTL_T_4US                          40     /* loop Number of   4us      */
#define MTL_T_5US                          50     /* loop Number of   5us      */
#define MTL_T_10US                        100     /* loop Number of  10us      */
#define MTL_T_20US                        200     /* loop Number of  20us      */
#define MTL_T_30US                        300     /* loop Number of  30us      */
#define MTL_T_50US                        500     /* loop Number of  50us      */
#define MTL_T_100US                      1000     /* loop Number of 100us      */
#define MTL_T_200US                      2000     /* loop Number of 200us      */
#define MTL_T_300US                      3000     /* loop Number of 300us      */
#define MTL_T_400US     ( MTL_T_200US  *  2 )     /* loop Number of 400us      */
#define MTL_T_1MS                       10000     /* loop Number of   1ms      */
#endif
```

上記は、未測定のため、妥当な値が設定されていませんので、評価してください。

7.2　クロック同期式シングルマスタ制御ソフトウェアの設定

設定箇所は、各ファイル中の「/** SET **/」というコメントの部分です。

7.2.1　R_SIO.h

(1)　BRR設定後のウェイトの定義

RSPI の SPBR 設定後、転送 1bit 期間をソフトウェア・ウェイトにより待ちます。ウェイト時間を設定してください。

初期値として、10μs を設定しています。

マルチメディアカードを使用する場合、100kHz 通信を想定し、10μs を設定してください。

```
#define SIO_T_BRR_WAIT        (uint16_t)MTL_T_10US /* BRR setting wait time */
```

7.2.2　R_SIO_rspi.h

RSPI 用の定義ファイルです。

R_SIO_rspi.h.XXX は、各 MCU に作成したものです。どれか一つを R_SIO_rspi.h にリネームして使用してください。対象 MCU のものが無い場合には、参照して、R_SIO_rspi.h を作成してください。

(1)　使用する動作モードの定義

使用する MCU のリソースの設定が可能です。必要な定義を 1 つ選んで設定してください。下記は、SIO_OPTION_4 を選択した場合の例です。**表 7-1**　動作モードに機能を示します。

```
/*------------------------------------------------------------------------*/
/*  Define the combination of the MCU's resources.                        */
/*------------------------------------------------------------------------*/
//#define SIO_OPTION_1     /*    */ /* SI/O                               */
//#define SIO_OPTION_2     /*    */ /* SI/O + CRC                         */
//#define SIO_OPTION_3     /*    */ /* SI/O + S/W CRC                     */
#define SIO_OPTION_4       /*    */ /* SI/O           + High Speed Read   */
//#define SIO_OPTION_5     /*    */ /* SI/O + CRC     + High Speed Read   */
//#define SIO_OPTION_6     /*    */ /* SI/O + S/W CRC + High Speed Read   */
```

表 7-1　動作モード

#define 定義	動作モード			
	SI/O （RSPI）	CRC 演算 （MCU 内蔵演算器）	CRC 演算 （ソフトウェア処理）	受信モード
SIO_OPTION_1	○	－	－	通常受信
SIO_OPTION_2	○	○	－	通常受信
SIO_OPTION_3	○	－	○	通常受信
SIO_OPTION_4	○	－	－	高速受信
SIO_OPTION_5	○	○	－	高速受信
SIO_OPTION_6	○	－	○	高速受信

SIO_OPTION_1～SIO_OPTION_3 を選択した場合、「通常受信」を行います。「通常受信」は、データ受信（full）を確認し、データ出力を行った後、次データの受信を行うことから、オーバランエラーの発生しない、安全な受信を行うことができます。連続受信中のデータエンディアンの変換処理を、次のダミーデータ書き込み後に行い、極力データ受信中にソフトウェア処理が行われる仕様としています。

SIO_OPTION_4～SIO_OPTION_6 を選択した場合、「高速受信」を行うことができます。「高速受信」は、データ受信中に次のデータ受信を行うためのダミーデータを書き込むことで、受信データ取り出し後、すぐに次のデータ受信を行うことができます。ただし、連続受信中は割り込み禁止区間が発生します。この区間では、他アプリケーションでの DMAC/EXDMAC/DTC 転送と本受信で使用するバスが競合した場合や優先度の高い NMI 割り込みが発生した場合、受信データの取り出しが間に合わないためにオーバランエラーが発生する可能性があります。これを回避したい場合には、SIO_OPTION_1～SIO_OPTION_3 のいずれかを選択してください。

MSB ファーストでの CRC-CCITT 演算処理のために MCU 内蔵 CRC 演算器を使う場合、SIO_OPTION_2 または SIO_OPTION_5 を指定してください。

MSBファーストでのCRC-CCITTソフトウェア演算処理を行う場合、SIO_OPTION_3またはSIO_OPTION_6を指定してください。

シリアル EEPROM、シリアルフラッシュメモリを制御する場合は、CRC-CCITT 演算処理を使用しないため、コメントアウトしてください。

(2) 使用するCRC演算処理の定義

使用する CRC 演算処理を指定してください。

シリアル EEPROM、シリアルフラッシュメモリを制御する場合は、CRC-CCITT 演算処理を使用しないため、コメントアウトしてください。

マルチメディアカードを制御する場合、定義してください。

```
/*-----------------------------------------------------------------------*/
/*  Define the CRC calculation.                                          */
/*-----------------------------------------------------------------------*/
#define SIO_CRCCCITT_USED              /* CRC-CCITT used                 */
#define SIO_CRC7_USED                  /* CRC7 used                      */
```

(3) 使用するRSPIチャネルの定義

使用する RSPI チャネル番号を指定してください。

```
/*-----------------------------------------------------------------------*/
/*  Define the RSPI channel.                                             */
/*-----------------------------------------------------------------------*/
#define SIO_RSPI_CHANNEL   0              /* RSPI Channel Select          */
```

(4) 使用する端子の定義

使用するシリアル端子の定義を示します。**表** 7-2　使用する端子の定義一覧を参考に使用する端子番号を指定してください。

```
/*------------------------------------------------------------------------*/
/*  Define the control port.                                              */
/*------------------------------------------------------------------------*/
/* Set to use port numbers and bit numbers */
#define SIO_DATAI_PORTNO    A               /* SIO DataIn Port No.          */
#define SIO_DATAI_BITNO     7               /* SIO DataIn Bit No.           */
#define SIO_CLK_PORTNO      A               /* SIO CLK Port No.             */
#define SIO_CLK_BITNO       5               /* SIO CLK Bit  No.             */
#define SIO_DATAO_PORTNO    A               /* SIO DataOut Port No.         */
#define SIO_DATAO_BITNO     6               /* SIO DataOut Bit No.          */
#define SIO_PFC_SELECT      G               /* RSPI PFC SELECT.( Set 'G'or'H' )*/
```

表 7-2　**使用する端子の定義一覧**

#define **定義**	**設定値**
SIO_DATAI_PORTNO	DataIn 端子のポート番号
SIO_DATAI_BITNO	DataIn 端子のビット番号
SIO_CLK_PORTNO	CLK 端子のポート番号
SIO_CLK_BITNO	CLK 端子のビット番号
SIO_DATAO_PORTNO	DataOut 端子のポート番号
SIO_DATAO_BITNO	DataOut 端子のビット番号
SIO_PFC_SELECT	ポートファンクションレジスタ設定”x”：PFxSPI（x = G or H）

(5) 使用するポートファンクションレジスタ（PFxSPI）のRSPISビットの定義

使用するシリアル端子に合わせて、ポートファンクションレジスタの RSPIS ビット（RSPI 端子選択ビット）の設定を行います。

```
/*------------------------------------------------------------------------*/
/*  Define the control RSPIS Bit (RSPI Pin Select) of Port Function Control Register.*/
/*------------------------------------------------------------------------*/
#define SIO_RSPI_SELECT     (uint8_t)(1)/* RSPI pin select set.(Set '0'or'1')*/
```

(6) マスクレベルの定義

高速受信動作の際に、割り込み禁止処理を入れています。割り込みを禁止させるために、優先順位の最も高いマスクレベルを指定してください。MCU に依存します。

RX62N の場合、15 を設定してください。

```
/*------------------------------------------------------------------------*/
/*  Define the interrupt mask level.                                      */
/*  Set interrupt mask level due to protect an overrun error by interrupt.  */
/*------------------------------------------------------------------------*/
#define SIO_INT_MASK_LEVEL     (uint8_t)(15)      /* Interrupt Mask Level    */
```

(7) ソフトタイマの定義

本サンプルコードのみで使用するソフトタイマを設定してください。

設定値は、0.1us 以上となる値を設定してください。

```
/*------------------------------------------------------------------------*/
/*  Define the wait time for timeout.                                     */
/*  Time out is occurred after 50000 times loop process of wait time.     */
/*------------------------------------------------------------------------*/
#define SIO_T_RSPI_WAIT    (uint16_t)(1)  /* 0.1us wait When CPU clock = 96MHz*/
```

(8) プルアップ抵抗コントロールレジスタ（PCR）の定義

インライン関数 SIO_DATAI_INIT()では、PCR を定義することができます。使用する場合はコメントを外してください。ただし、PCR の設定が可能な端子は、Port9,PortA～PortE 及び PortG です。

```
/*------------------ DataIn control ------------------*/
#pragma inline(SIO_DATAI_INIT)
static void SIO_DATAI_INIT(void)          /* DataIn Initial Setting         */
{
  SIO_ICR_DATAI    = 1;                   /* DataIn Input Buffer : Enable   */
  SIO_PCR_DATAI    = 0;     /** PA7/PC7/PE7 **//* DataIn Input Pull-up: off  */
  SIO_DDR_DATAI    = SIO_IN;              /* DataIn      Input              */
}
```

(9) オープンドレインコントロールレジスタ（ODR）の定義

インライン関数 SIO_DATAO_INIT()と SIO_DATAO_CLK()では、ODR を定義することができます。使用する場合はコメントを外してください。ただし、ODR の設定が可能な端子は、Port0～Port3 及び PortC です。

```
/*------------------ DataOut control -------------------*/
#pragma inline(SIO_DATAO_INIT)
static void SIO_DATAO_INIT(void)                /* DataOut Initial Setting */
{
/* SIO_ODR_DATAO  = 0; */  /** P26/PC6 **/      /* Open Drain Control: CMOS*/



/*------------------- CLK control --------------------*/
#pragma inline(SIO_CLK_INIT)
static void SIO_CLK_INIT(void)                  /* CLK Initial Setting     */
{
/* SIO_ODR_CLK    = 0; */  /** P27/PC5 **/      /* Open Drain Control: CMOS*/
```

8. 使用上の注意事項

8.1 組み込み時の注意事項

本サンプルコードを組み込む場合は、R_SIO.h、R_SIO_rspi.h（R_SIO_rspi.h.XXX をリネーム）をインクルードしてください。

8.2 不必要な関数について

使用されない関数は、ROM を不必要に消費するため、コメントアウト等の処理にて、組み込まれないことを推奨します。

8.3 他MCUを使用する場合

他 MCU を使用する場合、容易に対応が可能です。

準備するファイルは、

- R_SIO_rspi.h.XXX に相当する I/O モジュール共通定義
- mtl_com.h.XXX に相当するヘッダ定義

です。添付のものを参考に、作成してください。

8.4 CRC演算器のモジュールストップ設定について（オプション）

CRC 演算処理を使用した関数する場合、初期化処理にて、モジュールストップ解除を行いますが、モジュールストップ設定を行いません。モジュールストップ設定が必要な場合は、ユーザプログラムで制御してください。

8.5 入力バッファコントロールレジスタ（PORTn.ICR）に設定について

本サンプルコードでは、RSPI 以外の周辺モジュールの設定を行っていないため、当該端子に割り当てられている周辺モジュールの設定にて、入力機能を無効にした状態で、本サンプルコードを使用してください。

入力機能を無効にしていない状態で、PORTn.ICR レジスタの設定を変更する場合、端子の状態によっては内部にエッジが発生し、意図しない動作をすることがあります。

8.6 コンパイルオプションについて

最適化レベル「2」設定、かつ、最適化方法「サイズ優先」設定での動作は、確認済です。

最適化レベル「2」指定、かつ、最適化方法「スピード優先」設定での動作は、未確認です。

8.7 他アプリケーションでのDMAC/EXDMAC/DTC転送使用時の注意事項について

使用する動作モードで SIO_OPTION_4～SIO_OPTION_6 を選択し、かつ、他アプリケーションでの DMAC/EXDMAC/DTC 転送使用時に、本サンプルコードで使用するバスが競合した場合や優先度の高い NMI 割り込みが発生した場合、受信データの取り出しが間に合わないためにオーバランエラーが発生する可能性があります。

これを回避するには、使用する動作モードの定義で SIO_OPTION_1～SIO_OPTION_3 のいずれかを選択してください。

ホームページとサポート窓口

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

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

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2011.10.19	—	初版発行
1.01	2011.12.14	3	表 2-1 の C コンパイラ「最適化オプション 0 指定以外は、」を削除した。
		3	表 2-1 のサンプルコードのバージョンを Ver2.00→Ver2.02 に変更した。
		3	表 2-1 の評価に使用したソフトウェア　を修正した。
		9	6.3 必要メモリサイズ　を修正した。
		10	6.4 ファイル構成　アプリケーションノート番号を修正した。
		11	6.5.3 各種定義　ヘッダファイル名を削除した。
		14	6.8.1 ドライバ初期化処理　の備考を修正した。
		15	6.8.2 シリアル I/O 禁止設定処理　の備考を修正した。図 6-5 を修正した。
		16	6.8.3 シリアル I/O 許可設定処理　の備考を修正した。図 6-6 を修正した。
		18	6.8.5 シリアル I/O データ送信処理　の備考を修正した。
		19	図 6-8　を修正した。
		20	図 6-9　を修正した。
		21	6.8.6 シリアル I/O データ受信処理　の備考を修正した。
		22	図 6-10　を修正した。「送信データサイズの変更」を「受信データサイズの変更」に、「受信データの並び替え」を「受信データ取り出しと並び替え」に変更した。
		23	図 6-11　を修正した。「送信データサイズの変更」を「受信データサイズの変更」に、「送信バッファエンプティ」を「受信バッファフル」に、「受信データの並び替え」を「受信データ取り出しと並び替え」に変更した。
		24	図 6-12　を修正した。「受信データの並び替え」を「受信データ取り出しと並び替え」に変更した。
		25	6.8.7 送信データエンディアン変換処理　関数宣言の「STATIC」を削除した。説明を変更した。
		26	6.8.8 受信データエンディアン変換処理　関数宣言の「STATIC」を削除した。説明を変更した。
		28	6.9.3 マクロ関数　SIO_DATAI_INIT()　(2) 機能 ①　において、「入力バッファ機能を無効にする」を「入力バッファ機能を有効にする」に修正した。
		33	6.9.12 マクロ関数　SIO_TX_DISABLE()　(2) 機能 ②　において、「SPCR をリード」を追加した。
		34	6.9.14 マクロ関数　SIO_TRX_ENABLE()　(2) 機能 ②　において、「IR フラグをクリア」を追加した。

		34	6.9.14 **マクロ関数**　SIO_TRX_ENABLE()　(3) **備考**　**において、「**SIO_TX_DISABLE()**」を「**SIO_TRX_DISABLE()**」に修正した。**
		35	6.9.14 **マクロ関数**　SIO_TRX_DISABLE()　(2) **機能 ②　において、「**SPCR **をリード」を追加した。**
		38	7.1 mtl_com.h　(1)OS **のヘッダファイル定義　を変更した**
		40	7.1.2 mtl_tim.h　**タイマカウンタを変更した**
		41	7.2.1 R_SIO.h　**を追加した。**
		46	8.4 **コンパイルオプションについて　を削除した** **これにともない以降の項番を繰り上げた。**
1.02	2011.12.19	3	**表** 2-1 **のコンパイルオプションにおいて、デフォルト設定の詳細を追加した。**
		3	**表** 2-1 **のサンプルコードのバージョンを** Ver2.02→Ver2.03 **に変更した。**
		10	6.4 **ファイル構成　アプリケーションノート番号を修正した。**
		46	8.6 **コンパイルオプションについて　を追加した。**
1.03	2012.3.31	2	1. **仕様　内容を変更した。**
		3	**表** 2-1 **のサンプルコードのバージョンを** Ver2.03→Ver2.04 **に変更した。**
		3	**表** 2-2**、表** 2-3 **を追加**
		5	5.1 **ハードウェア構成と** 5.2 **使用端子一覧 を入れ替えた。**
		6	6.1 **動作概要　の一部の内容を** 6.2.2. **データバッファと送信／受信データの関係　に移動させた。**
		7	6.2.1 **ソフトウェア構成　に内容を追加した。**
		8	6.2.2 **データバッファと送信／受信データの関係　を追加した。**
		8	6.2.3 －6.2.6 **を削除した。**
		9	6.3 **必要メモリサイズ　内容を追加とメモリサイズを更新した。**
		10	6.4 **ファイル構成　アプリケーションノート番号を修正した。**
		11	6.5.1 **リターン値　元は、戻り値であった。**
		11	**表** 6-4 **各種定義値　内容欄内の用語を変更**
		13	**表** 6-6 **関数**　R_SIO_Tx_Exchg(),R_SIO_Rx_Exchg()**を削除した。**
		14	6.8 **状態遷移図**　6.10 **から** 6.8 **に移動した。**
		15	6.9.1 **ドライバ初期化処理　備考を変更した。**
		16	6.9.2 **シリアル** I/O **禁止設定処理　備考と処理概要を変更した。**
		17	6.9.3 **シリアル** I/O **許可設定処理　備考と処理概要を変更した。**
		18	6.9.4 **シリアル** I/O **開放設定処理　処理概要を変更した。**
		19-21	6.9.5 **シリアル** I/O **データ送信処理　備考と処理概要を変更した。**
		22-27	6.9.6 **シリアル** I/O **データ受信処理　説明欄に、「通常受信」と「高速受信」を追加した。備考と処理概要を変更した。**
		27	6.9.7 **送信データエンディアン変換処理、**6.8.8. **受信データエンディアン変換処理　を削除した。**
		28-37	6.10 **インライン関数　元は、マクロ関数であった。** **インライン関数をマクロ関数に変更した。**
		28	6.10.1 SIO_IO_INIT() (2)**機能を変更した。**
		29	6.10.2 SIO_IO_OPEN() (2)**機能を変更した。**
		29	6.10.3 SIO_DATAI_INIT() (2)**機能を変更した。**
		30	6.10.4 SIO_DATAO_INIT() (2)**機能と**(3)**備考を変更した。**

		30	6.10.5 SIO_DATAO_OPEN() (2)機能を変更した。
		31	6.10.6 SIO_CLK_INIT() (2)機能と(3)備考を変更した。
		31	6.10.7 SIO_CLK_OPEN() (2)機能を変更した。
		32	6.10.8 SIO_ENABLE() (2)機能と(3)備考を変更した。
		33	6.10.9 SIO_DISABLE() (2)機能を変更した。
		34	6.10.11 SIO_TX_ENABLE() (2)機能と(3)備考を変更した。
		35	6.10.12 SIO_TX_DISABLE() (2)機能を変更した。
		36	6.10.13 SIO_TRX_ENABLE() (2)機能と(3)備考を変更した。
		37	6.10.14 SIO_TRX_DISABLE() (2)機能を変更した。
		42	7.2.1 R_SIO.h (1) 内容を変更した。
		42-43	7.2.2 (1)使用する動作モードの定義　を変更した。
		43	7.2.2 (2)使用する CRC 演算処理の定義　を変更した。
		43	7.2.2 (3)使用する RSPI チャネルの定義　を変更した。
		44	7.2.2 (4)使用する端子の定義　を変更した。
		44	7.2.2 (5)使用するポートファンクションレジスタ（PFxSPI）の RSPIS ビットの定義　を追加した。
		45	7.2.2 (7)ソフトタイマの定義　を変更した。
		45	7.2.2 (8)プルアップ抵抗コントロールレジスタ（PCR）の定義を追加した。プログラムに合わせ、/* SET */を削除した。
		45	7.2.2 (9)オープンドレインコントロールレジスタ（ODR）の定義　を追加した。プログラムに合わせ、/* SET */を削除した。
		46	8.7 他アプリケーションでの DMAC/EXDMAC/DTC 転送使用時の注意事項について　を追加した。
1.04	2012.4.5	1	要旨　内容を変更した。
		7	6.2.1 ソフトウェア概要　内容を変更した。
		10	6.4 ファイル構成　アプリケーションノート番号を修正した。

製品ご使用上の注意事項

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

1. 未使用端子の処理
   【注意】未使用端子は、本文の「未使用端子の処理」に従って処理してください。
   CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。
2. 電源投入時の処置
   【注意】電源投入時は，製品の状態は不定です。
   電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。
   外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。
   同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. リザーブアドレス（予約領域）のアクセス禁止
   【注意】リザーブアドレス（予約領域）のアクセスを禁止します。
   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
4. クロックについて
   【注意】リセット時は、クロックが安定した後、リセットを解除してください。
   プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。
   リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
5. 製品間の相違について
   【注意】型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。
   同じグループのマイコンでも型名が違うと、内部 ROM、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ輻射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載され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。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

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

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

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

Colophon 1.1