RX220グループ

RSPI を用いた通信設定例

R01AN1807JJ0100
Rev.1.00
2013.10.01

要旨

本アプリケーションノートでは、RX220グループのシリアルペリフェラルインタフェース(以下、RSPI)を使用して、SPI 動作(4 線式)で全二重同期式のシリアル通信を行う方法について説明します。

本アプリケーションノートのサンプルコードは、1 つのワークスペースにマスタとスレーブのプロジェクトを登録しています。High-performance Embedded Workshop のアクティブプロジェクトでマスタかスレーブを選択してください。

対象デバイス

・RX220グループ　100 ピン版　ROM 容量：　64KB～256KB
・RX220グループ　64 ピン版　ROM 容量：　32KB～256KB
・RX220グループ　48 ピン版　ROM 容量：　32KB～256KB

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

目次

1. 仕様

RSPI の SPI 動作(4 線式)を使用して、1 つのマスタと 2 つのスレーブ(スレーブ 0 とスレーブ 1)で全二重同期式シリアル通信を行います。

マスタは、スレーブ 0 に対して 3 バイトデータを送受信します。3 バイトデータの送受信が完了すると、LED0 を点灯します。次にスレーブ 1 に対して 3 バイトデータを送受信します。3 バイトデータの送受信が完了すると LED1 を点灯します。送受信中にエラーが発生すると、送受信動作を終了し、LED2 を点灯します。

スレーブ 0 と 1 は、マスタに対して 3 バイトデータを送受信します。3 バイトデータの送受信が完了すると LED1 を点灯します。送受信中にエラーが発生すると、送受信動作を終了し、LED2 を点灯します。

- RSPI モード　　: SPI 動作(4 線式)
- 通信動作モード　　: 全二重同期式シリアル通信
- 転送速度　　: 6.25kbps
- 転送ビット長　　: 8 ビット
- パリティ　　: なし
- シーケンス長　　: 1 シーケンス
- フレーム数　　: 1 フレーム

表 1.1に使用する周辺機能と用途を、図 1.1に使用例を示します。

表1.1　使用する周辺機能と用途

周辺機能	用途
RSPI	全二重同期式シリアル通信
I/O ポート	LED 点灯

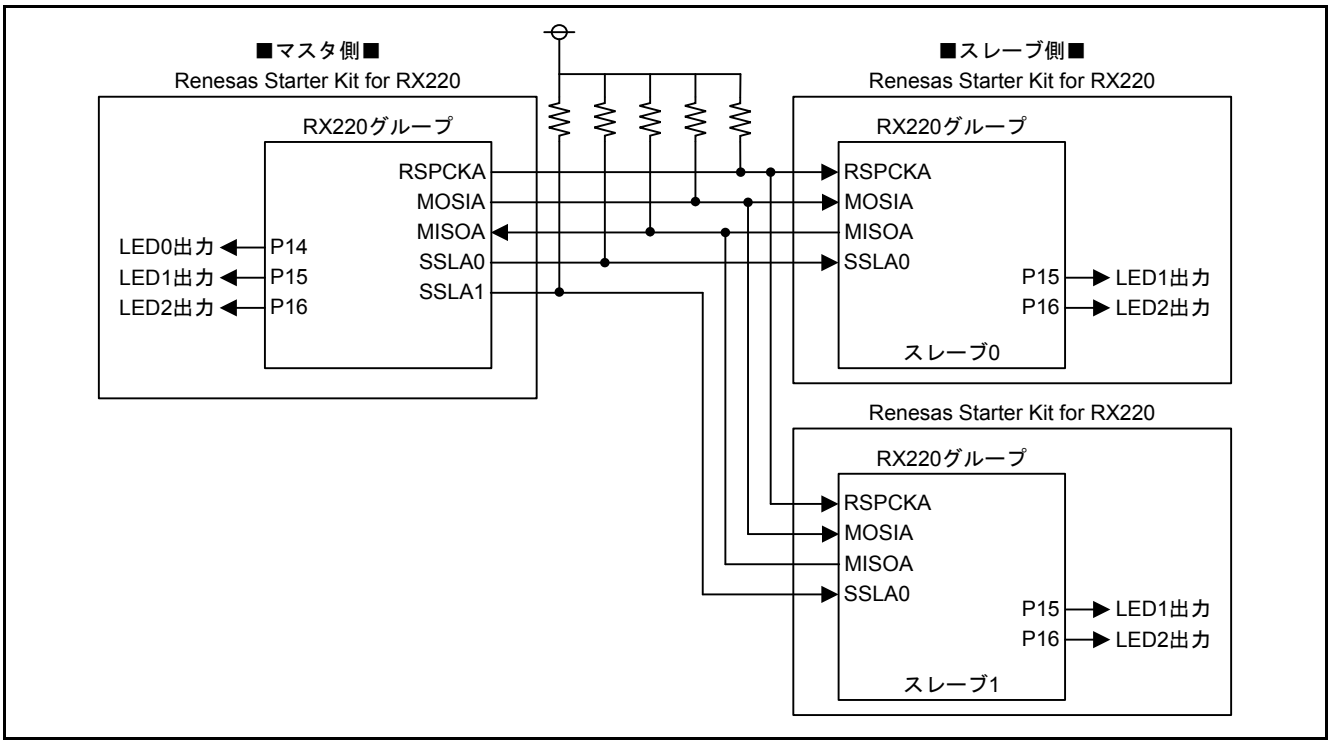


図1.1　使用例

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、下記の条件で動作を確認しています。

表2.1　動作確認条件

項目	内容
使用マイコン	R5F52206BDFP (RX220 グループ)
動作周波数	• メインクロック: 20MHz • システムクロック(ICLK): 20MHz (メインクロック 1 分周) • 周辺モジュールクロック B (PCLKB): 20MHz (メインクロック 1 分周)
動作電圧	5.0V
統合開発環境	ルネサスエレクトロニクス製 High-performance Embedded Workshop Version 4.09.01
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package for RX Family V.1.02 Release 01
	コンパイルオプション -cpu=rx200 -output=obj="$(CONFIGDIR)¥$(FILELEAF).obj" -debug -nologo (統合開発環境のデフォルト設定を使用しています)
iodefine.h のバージョン	Version 1.0A
エンディアン	リトルエンディアン
動作モード	シングルチップモード
プロセッサモード	スーパバイザモード
サンプルコードのバージョン	Version 1.00
使用ボード	Renesas Starter Kit for RX220 (製品型名: R0K5RX220C000BE)

3. 関連アプリケーションノート

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

- RX220グループ　初期設定例　Rev.1.00 (R01AN1494JJ0100_RX220)

上記アプリケーションノートの初期設定関数を、本アプリケーションノートのサンプルコードで使用しています。Rev は本アプリケーションノート作成時点のものです。
最新版がある場合、最新版に差し替えて使用してください。最新版はルネサスエレクトロニクスホームページで確認および入手してください。

4. ハードウェア説明(マスタ)

4.1 使用端子一覧

表 4.1に使用端子と機能を示します。

使用端子は 100 ピン版の製品を想定しています。100 ピン版未満の製品を使用する場合は、使用する製品に合わせて端子を選択してください。

表4.1 使用端子と機能

端子名	入出力	内容
P14	出力	LED0 出力(スレーブ 0 への RSPI 送受信完了)
P15	出力	LED1 出力(スレーブ 1 への RSPI 送受信完了)
P16	出力	LED2 出力(RSPI 送受信エラー)
PA0/SSLA1	出力	スレーブ 1 セレクト信号出力
PA4/SSLA0	出力	スレーブ 0 セレクト信号出力
PC5/RSPCKA	出力	クロック出力端子
PC6/MOSIA	出力	データ出力端子
PC7/MISOA	入力	データ入力端子

5.　ソフトウェア説明(マスタ)

リセット解除後、ユーザ I/F 関数(RSPI 初期設定)を呼び出し、RSPI の初期化を行います。

初期化後、スレーブ 0 を指定してユーザ I/F 関数(RSPI 送受信開始)を呼び出し、送受信動作を許可します。3 バイトの送受信を完了したとき、RSPI の送受信動作を禁止して、コールバック関数(スレーブ 0 への RSPI 送受信完了)を呼び出します。コールバック関数(スレーブ 0 への RSPI 送受信完了)で、LED0 を点灯します。

スレーブ 0 との送受信完了後、スレーブ 1 に対して同様に送受信を行います。送受信完了後、コールバック関数(スレーブ 1 への RSPI 送受信完了)で、LED1 を点灯します。

送受信エラーが発生した場合、RSPI の送受信動作を禁止して、コールバック関数(RSPI 送受信エラー)を呼び出します。コールバック関数(RSPI 送受信エラー)で、LED2 を点灯します。

以下に使用する周辺機能の設定を、図 5.1にソフトウェア構成を示します。

＜RSPI＞

- RSPI モード　　: SPI 動作(4 線式)
- 通信動作モード　　: 全二重同期式シリアル通信
- 転送速度　　: 6.25kbps
- クロックソース　　: PCLKB (20MHz)
- 転送ビット長　　: 8 ビット
- パリティ　　: なし
- シーケンス長　　: 1 シーケンス
- フレーム数　　: 1 フレーム
- エラー検出　　: オーバランエラー
- 割り込み要因　　: RSPI エラー割り込み(SPEI)を許可
  : RSPI 受信割り込み(SPRI)を許可
  : RSPI 送信割り込み(SPTI)を許可
  : RSPI アイドル割り込み(SPII)を許可

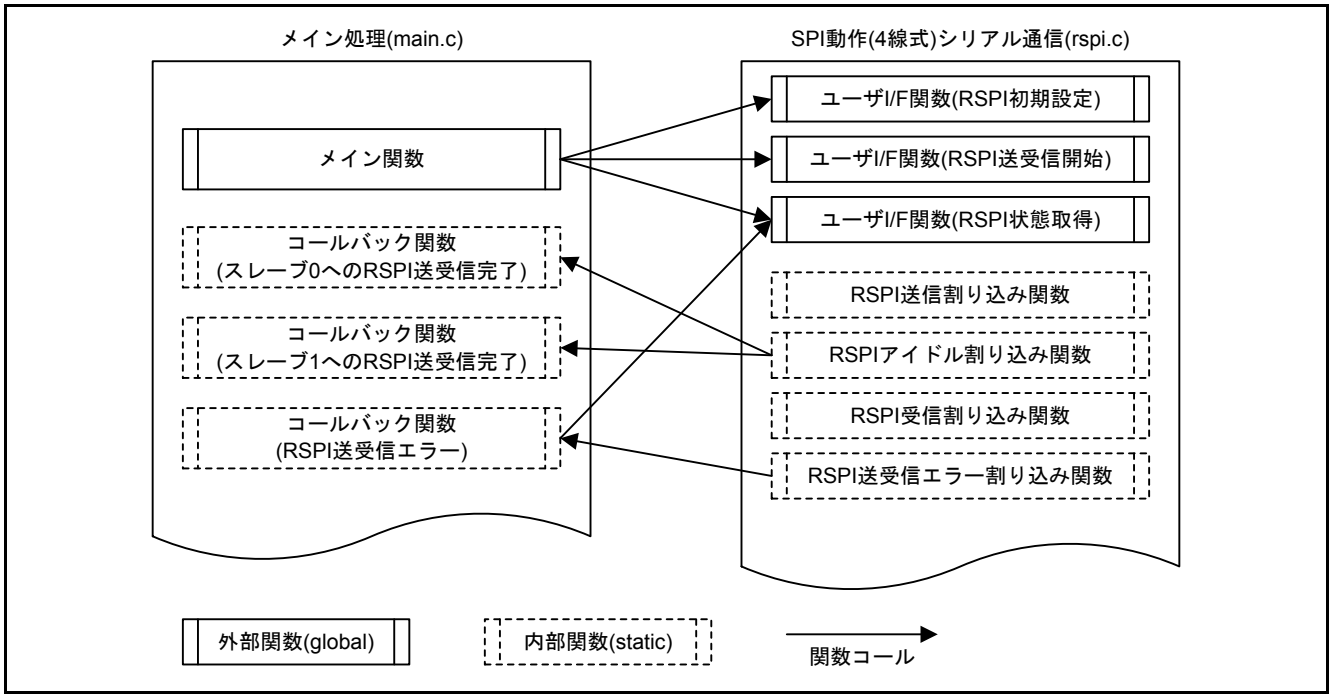


図5.1　ソフトウェア構成

5.1 動作概要

図 5.2、図 5.3にSPI 動作(4 線式)シリアル通信のタイミング図を、以下に図中の番号の動作および処理を示します。

(1) 初期設定
ユーザ I/F 関数(RSPI 初期設定)で RSPI の初期設定を行います。

(2) スレーブ 0 との送受信開始
ユーザ I/F 関数(RSPI 送受信開始)の引数にスレーブ 0 を選択して呼び出します。
ユーザ I/F 関数(RSPI 送受信開始)では、最初に SPSR.IDLNF ビットを確認します。“1”(RSPI が転送状態)の場合、RSPI_BUSY (RSPI 送受信中)を返します。“0”(RSPI がアイドル状態)の場合、送受信ビジーフラグを“1”、SPCMD0.SSLA ビットを“000b”(SSL0)にします。SPCR.SPEIE ビットを“1”(RSPI エラー割り込み要求の発生を許可)、SPCR.SPTIE ビットを“1”(RSPI 送信割り込み要求の発生を許可)、SPCR.SPE ビットを“1”(RSPI 機能は有効)、SPCR.SPRIE ビット“1”(RSPI 受信割り込みの要求の発生を許可)にします。RSPI エラー、RSPI 受信、RSPI 送信割り込みの IEN ビットを“1”にして送受信を開始します。

(3) スレーブ 0 にデータ送信
RSPI 送信割り込み処理で、SPDR レジスタにスレーブ 0 用送信バッファの値を書きます。最終データを書いたら、SPTIE ビットを“0”(RSPI 送信割り込みの発生を禁止)、SPCR2.SPIIE ビットを“1”(RSPI アイドル割り込み要求の発生を許可)にします。

(4) スレーブ 0 からデータ受信
RSPI 受信割り込み処理で、スレーブ 0 用受信バッファに SPDR レジスタの値を書きます。最終データを受信したら、SPRIE ビットを“0”(RSPI 受信割り込みの要求の発生を禁止)、SPEIE ビットを“0”(RSPI エラー割り込みの発生を禁止)にします。

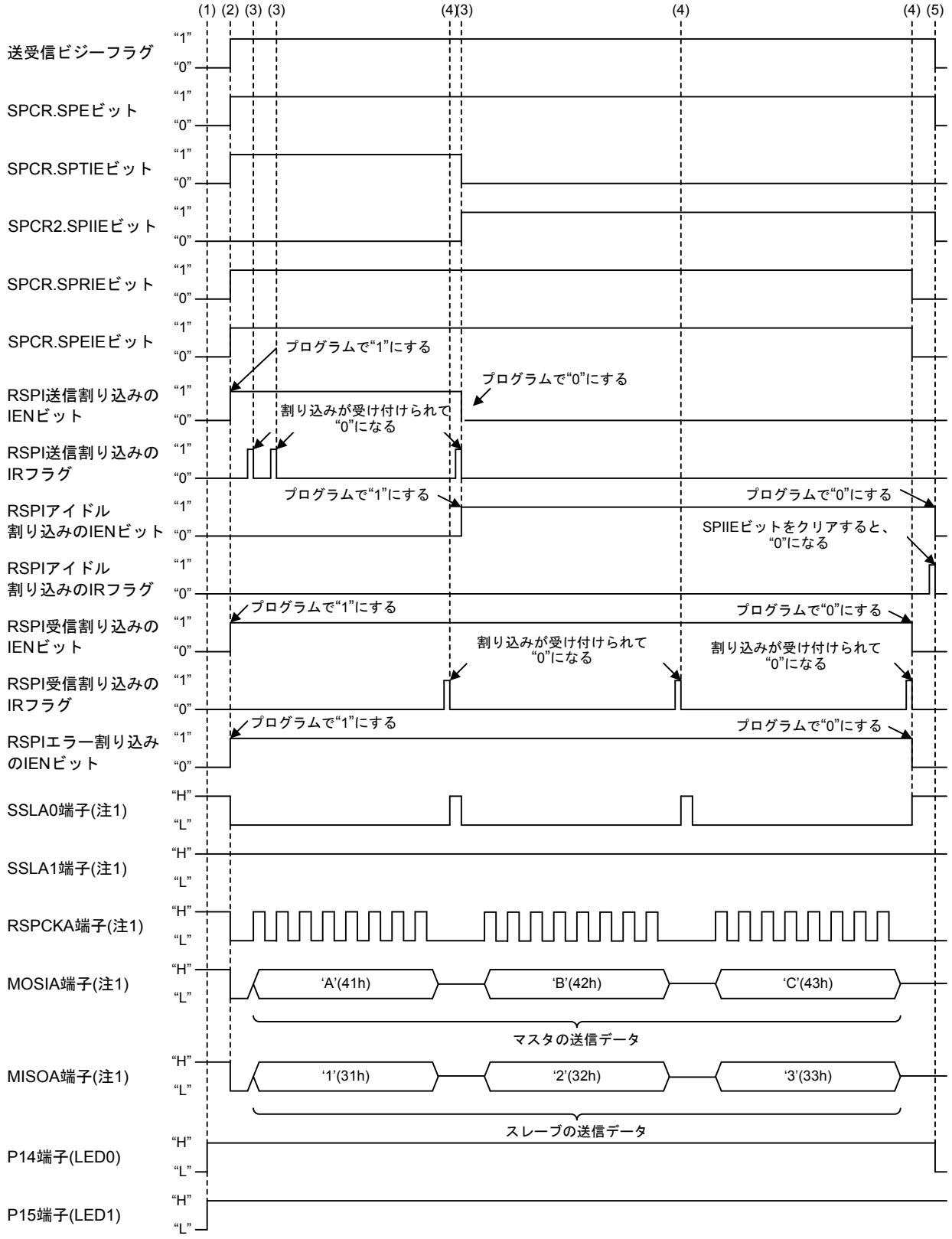
(5) スレーブ 0 との送受信完了
最終データの送受信が完了すると、RSPI アイドル割り込み要求が発生します。RSPI アイドル割り込み処理で、SPE ビットを“0”(RSPI 機能は無効)、SPIIE ビットを“0”(RSPI アイドル割り込み要求の発生を禁止)、送受信ビジーフラグを“0”にして、コールバック関数(スレーブ 0 への RSPI 送受信完了)を呼び出し、LED0 を点灯します。

(6) スレーブ 1 との送受信開始
SPCMD0.SSLA ビットに“001b”(SSL1)を設定し、(2)と同様にして送受信を開始します。

(7) スレーブ 1 にデータ送信
RSPI 送信割り込み処理で、SPDR レジスタにスレーブ 1 用送信バッファの値を書き、(3)と同様にしてデータを送信します。

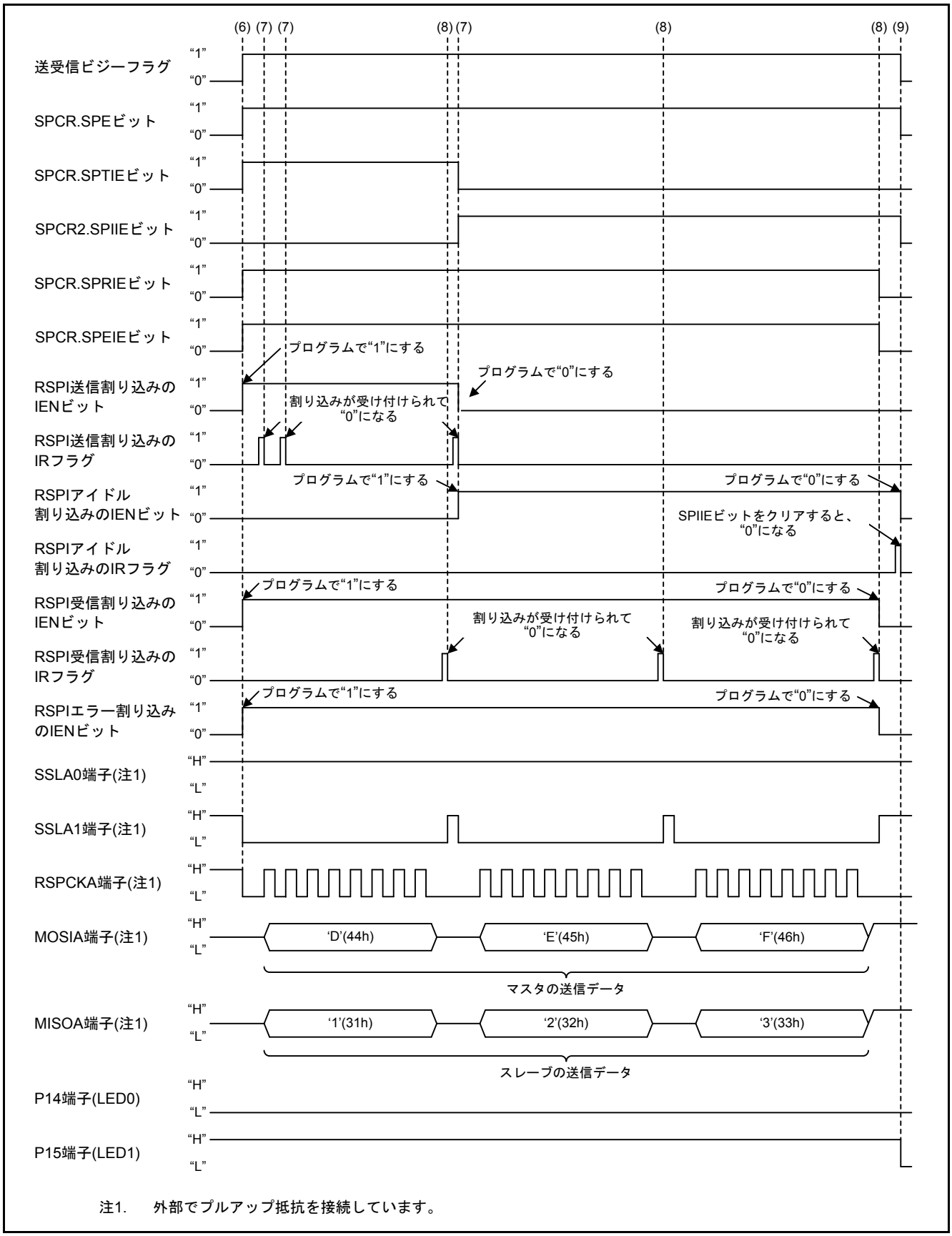
(8) スレーブ 1 からデータ受信
RSPI 受信割り込み処理で、スレーブ 1 用受信バッファに SPDR レジスタの値を書き、(4)と同様にしてデータを受信します。

(9) スレーブ 1 との送受信完了
(5)と同様にして送受信を完了します。コールバック関数(スレーブ 1 への RSPI 送受信完了)を呼び出し、LED1 を点灯します。



注1. 外部でプルアップ抵抗を接続しています。

図5.2 SPI 動作(4 線式)シリアル通信のタイミング図 (スレーブ 0 への送受信)



注1. 外部でプルアップ抵抗を接続しています。

図5.3 SPI 動作(4 線式)シリアル通信のタイミング図 (スレーブ 1 への送受信)

5.2 ファイル構成

表 5.1にサンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表5.1 サンプルコードで使用するファイル

ファイル名	概要	備考
main.c	メイン処理	
r_init_stop_module.c	リセット後に動作している周辺機能の停止	
r_init_stop_module.h	r_init_stop_module.c のヘッダファイル	
r_init_non_existent_port.c	存在しないポートの初期設定	
r_init_non_existent_port.h	r_init_non_existent_port.c のヘッダファイル	
r_init_clock.c	クロック初期設定	
r_init_clock.h	r_init_clock.c のヘッダファイル	
rspi.c	SPI 動作(4 線式)シリアル通信	
rspi.h	rspi.c のヘッダファイル	

5.3 オプション設定メモリ

表 5.2にサンプルコードで使用するオプション設定メモリの状態を示します。必要に応じて、お客様のシステムに最適な値を設定してください。

表5.2 サンプルコードで使用するオプション設定メモリ

シンボル	アドレス	設定値	内容
OFS0	FFFF FF8Fh～FFFF FF8Ch	FFFF FFFFh	リセット後、IWDT は停止
OFS1	FFFF FF8Bh～FFFF FF88h	FFFF FFFFh	リセット後、電圧監視 0 リセット無効 リセット後、HOCO 発振が無効
MDES	FFFF FF83h～FFFF FF80h	FFFF FFFFh	リトルエンディアン

5.4 定数一覧

表 5.3～表 5.5にサンプルコードで使用する定数を示します。

表5.3 サンプルコードで使用する定数(main.c)

定数名	設定値	内容
LED0_REG_PODR	PORT1.PODR.BIT.B4	LED0 出力データ格納ビット
LED0_REG_PDR	PORT1.PDR.BIT.B4	LED0 方向制御ビット
LED0_REG_PMR	PORT1.PMR.BIT.B4	LED0 端子モード制御ビット
LED1_REG_PODR	PORT1.PODR.BIT.B5	LED1 出力データ格納ビット
LED1_REG_PDR	PORT1.PDR.BIT.B5	LED1 方向制御ビット
LED1_REG_PMR	PORT1.PMR.BIT.B5	LED1 端子モード制御ビット
LED2_REG_PODR	PORT1.PODR.BIT.B6	LED2 出力データ格納ビット
LED2_REG_PDR	PORT1.PDR.BIT.B6	LED2 方向制御ビット
LED2_REG_PMR	PORT1.PMR.BIT.B6	LED2 端子モード制御ビット
LED_ON	0	LED 出力データ: 点灯
LED_OFF	1	LED 出力データ: 消灯
TR_SIZE	3	送受信サイズ
BUF_SIZE	TR_SIZE	バッファサイズ

表5.4 サンプルコードで使用する定数(rspi.c)

定数名	設定値	内容
SPSR_ERROR_FLAGS	0Dh	RSPI.SPSR レジスタのエラーフラグのビットパターン
B_RSPI_BUSY	state.bit.b_rspi_busy	送受信ビジーフラグ 0: 送受信レディ 1: 送受信ビジー
B_RX_ORER	state.bit.b_rx_orer	オーバランエラーフラグ 0: オーバランエラーなし 1: オーバランエラーあり

表5.5 サンプルコードで使用する定数(rspi.h)

定数名	設定値	内容
RSPI_OK	00h	RSPI_PreTrans 関数のリターン値: RSPI 送受信開始
RSPI_NOT_IDLE	01h	RSPI_PreTrans 関数のリターン値: RSPI 送受信中
RSPI_NG	02h	RSPI_PreTrans 関数のリターン値: 引数エラー
RSPI_SSL0	0000h	RSPI コマンドレジスタ 0～7 の SSL 信号アサート設定ビットの設定値 (SSL0 端子を選択)
RSPI_SSL1	0010h	RSPI コマンドレジスタ 0～7 の SSL 信号アサート設定ビットの設定値 (SSL1 端子を選択)

5.5 構造体/共用体一覧

図 5.4にサンプルコードで使用する構造体/共用体を示します。

```
#pragma bit_order     left          /* ビットフィールドの並び順指定: 上位ビット側からメンバを割り付ける*/
#pragma unpack                      /* 構造体メンバの境界調整数指定: メンバの型でアライメントする       */
typedef   union
{
    uint8_t    byte;
    struct
    {
        uint8_t    b_rspi_busy :1;    /*  送受信ビジーフラグ          0:送受信レディ 1:送受信ビジー          */
        uint8_t    b_rx_orer    :1;    /*  オーバランエラーフラグ     0:エラーなし   1:オーバランエラー      */
        uint8_t                 :6;    /*  使用しない                                                    */
    } bit;
} rspi_state_t;
#pragma packoption                  /* 構造体メンバの境界調整数指定の終了 */
#pragma bit_order                   /* ビットフィールドの並び順指定の終了 */
```

図5.4　サンプルコードで使用する構造体/共用体

5.6 変数一覧

表 5.6にstatic 型変数を示します。

表5.6　static 型変数

型	変数名	内容	使用関数
static uint8_t	tx_buf_0[]	スレーブ 0 用の送信バッファ	main
static uint8_t	rx_buf_0[BUF_SIZE]	スレーブ 0 用の受信バッファ	main
static uint8_t	tx_buf_1[]	スレーブ 1 用の送信バッファ	main
static uint8_t	rx_buf_1[BUF_SIZE]	スレーブ 1 用の受信バッファ	main
static const uint8_t *	pbuf_tx	送信バッファへのポインタ	RSPI_PreTrans rspi_spti_isr
static uint8_t	tx_cnt	送信カウンタ	
static uint8_t *	pbuf_rx	受信バッファへのポインタ	RSPI_PreTrans rspi_spri_isr
static uint8_t	rx_cnt	受信カウンタ	
static rspi_state_t	state	RSPI 状態	RSPI_PreTrans RSPI_GetState rspi_spii_isr rspi_spei_isr

5.7　関数一覧

表 5.7にサンプルコードで使用する関数を示します。

表5.7　サンプルコードで使用する関数

関数名	概要
main	メイン処理
port_init	ポート初期設定
R_INIT_StopModule	リセット後に動作している周辺機能の停止
R_INIT_NonExistentPort	存在しないポートの初期設定
R_INIT_Clock	クロック初期設定
peripheral_init	周辺機能初期設定
cb_rspi_slave0_end	コールバック関数(スレーブ 0 への RSPI 送受信完了)
cb_rspi_slave1_end	コールバック関数(スレーブ 1 への RSPI 送受信完了)
cb_rspi_rx_error	コールバック関数(RSPI 送受信エラー)
RSPI_Init	ユーザ I/F 関数(RSPI 初期設定)
RSPI_PreTrans	ユーザ I/F 関数(RSPI 送受信開始)
RSPI_GetState	ユーザ I/F 関数(RSPI 状態取得)
rspi_spti_isr	RSPI 送信割り込み
rspi_spii_isr	RSPI アイドル割り込み
rspi_spri_isr	RSPI 受信割り込み
rspi_spei_isr	RSPI エラー割り込み
Excep_RSPI0_SPEI0	RSPI0.SPEI0 割り込み処理
Excep_RSPI0_SPRI0	RSPI0.SPRI0 割り込み処理
Excep_RSPI0_SPTI0	RSPI0.SPTI0 割り込み処理
Excep_RSPI0_SPII0	RSPI0.SPII0 割り込み処理

5.8　関数仕様

サンプルコードの関数仕様を示します。

main	
概　要	メイン処理
ヘッダ	なし
宣　言	void main(void)
説　明	初期設定後、スレーブ 0 へ RSPI 送受信を開始します。スレーブ 0 への送受信が完了した後、スレーブ 1 へ RSPI 送受信を開始します。
引　数	なし
リターン値	なし

port_init	
概　要	ポート初期設定
ヘッダ	なし
宣　言	static void port_init(void)
説　明	ポートの初期設定を行います。
引　数	なし
リターン値	なし

R_INIT_StopModule	
概　要	リセット後に動作している周辺機能の停止
ヘッダ	r_init_stop_module.h
宣　言	void R_INIT_StopModule(void)
説　明	モジュールストップ状態へ遷移する設定を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、モジュールストップ状態への遷移は行っていません。 本関数の詳細は、アプリケーションノート「RX220グループ 初期設定例 Rev.1.00」を参照してください。

R_INIT_NonExistentPort	
概　要	存在しないポートの初期設定
ヘッダ	r_init_non_existent_port.h
宣　言	void R_INIT_NonExistentPort(void)
説　明	100 ピン未満の製品に対して、存在しないポートの端子に対応するポート方向レジスタの初期設定を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、100 ピン版(PIN_SIZE=100)に設定しています。 本関数をコールした後に、存在しないポートを含む PDR、PODR レジスタへバイト単位で書き込む場合、存在しないポートの方向制御ビットには“1”、ポート出力データ格納ビットには“0”を設定してください。 本関数の詳細は、アプリケーションノート「RX220グループ 初期設定例 Rev.1.00」を参照してください。

R_INIT_Clock	
概　要	クロック初期設定
ヘッダ	r_init_clock.h
宣　言	void R_INIT_Clock(void)
説　明	クロックの初期設定を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、システムクロックを PLL とし、サブクロックを使用しない処理を選択しています。 本関数の詳細は、アプリケーションノート「RX220グループ 初期設定例 Rev.1.00」を参照してください。

peripheral_init	
概　要	周辺機能初期設定
ヘッダ	なし
宣　言	static void peripheral_init(void)
説　明	使用する周辺機能の初期設定を行います。
引　数	なし
リターン値	なし

cb_rspi_slave0_end	
概　要	コールバック関数(スレーブ 0 への RSPI 送受信完了)
ヘッダ	なし
宣　言	static void cb_rspi_slave0_end(void)
説　明	スレーブ 0 への RSPI 送受信完了時に呼び出されます。
引　数	なし
リターン値	なし

cb_rspi_slave1_end	
概　要	コールバック関数(スレーブ 1 への RSPI 送受信完了)
ヘッダ	なし
宣　言	static void cb_rspi_slave1_end(void)
説　明	スレーブ 1 への RSPI 送受信完了時に呼び出されます
引　数	なし
リターン値	なし

cb_rspi_rx_error	
概　要	コールバック関数(RSPI 送受信エラー)
ヘッダ	なし
宣　言	static void cb_rspi_rx_error(void)
説　明	RSPI 送受信エラー発生時に呼び出されます。
引　数	なし
リターン値	なし
備　考	サンプルコードではエラー処理を行っていません。必要に応じてプログラムを追加してください。

RSPI_Init	
概　要	ユーザ I/F 関数(RSPI 初期設定)
ヘッダ	rspi.h
宣　言	void RSPI_Init(void)
説　明	RSPI の初期設定を行います。
引　数	なし
リターン値	なし

RSPI_PreTrans

概　要	ユーザ I/F 関数(RSPI 送受信開始)		
ヘッダ	rspi.h		
宣　言	uint8_t RSPI_PreTrans(uint16_t ssl, const uint8_t * pbuf_t, uint8_t * pbuf_r, uint8_t num, CallBackFunc pcb_end, CallBackFunc pcb_rx_error)		
説　明	RSPI がアイドル状態であるか確認します。引数で指定された SSL 端子を設定します。RSPI 機能を有効、RSPI 送信割り込み、RSPI 受信割り込み、RSPI エラー割り込みを許可にして、RSPI 送受信を開始します。		
引　数	uint16_t	ssl	: SSL 端子選択
	const uint8_t *	pbuf_t	: 送信データ格納バッファへのポインタ
	uint8_t *	pbuf_r	: 受信データ格納バッファへのポインタ
	uint8_t	num	: 送受信バイト数
	CallBackFunc	pcb_end	: コールバック関数ポインタ(送受信完了)
	CallBackFunc	pcb_rx_error	: コールバック関数ポインタ(送受信エラー)
リターン値	RSPI_NG		: 引数エラー(送受信バイト数が 0)
	RSPI_NOT_IDLE		: RSPI 送受信中
	RSPI_OK		: RSPI 送受信開始

RSPI_GetState

概　要	ユーザ I/F 関数(RSPI 状態取得)	
ヘッダ	rspi.h	
宣　言	rspi_state_t RSPI_GetState(void)	
説　明	RSPI 状態を返します。	
引　数	なし	
リターン値	rspi_state_t.bit.b_rspi_busy	: 送受信ビジーフラグ 0:送受信レディ 1:送受信ビジー
	rspi_state_t.bit.b_rx_orer	: オーバランエラーフラグ 0:エラーなし 1:オーバランエラー

rspi_spti_isr

概　要	RSPI 送信割り込み
ヘッダ	なし
宣　言	static void rspi_spti_isr(void)
説　明	RSPI0.SPTI0 割り込み処理関数で呼び出されます。送信データを SPDR レジスタに書き込みます。最終データを送信すると、RSPI 送信割り込み要求の発生を禁止し、RSPI アイドル割り込み要求の発生を許可します。
引　数	なし
リターン値	なし

rspi_spii_isr	
概　要	RSPI アイドル割り込み
ヘッダ	なし
宣　言	static void rspi_spii_isr(void)
説　明	RSPI0.SPII0 割り込み処理関数で呼び出されます。RSPI 機能を無効にして、コールバック関数(スレーブ 0 への RSPI 送受信完了)かコールバック関数(スレーブ 1 への RSPI 送受信完了)を呼び出します。
引　数	なし
リターン値	なし

rspi_spri_isr	
概　要	RSPI 受信割り込み
ヘッダ	なし
宣　言	static void rspi_spri_isr(void)
説　明	RSPI0.SPRI0 割り込み処理関数から呼び出されます。SPDR レジスタから受信データを読み出します。最終データを受信すると、RSPI 受信割り込み要求の発生を禁止します。
引　数	なし
リターン値	なし

rspi_spei_isr	
概　要	RSPI エラー割り込み
ヘッダ	なし
宣　言	static void rspi_spei_isr(void)
説　明	RSPI0.SPEI0 割り込み処理関数から呼び出されます。RSPI 機能を無効にして、コールバック関数(RSPI 送受信エラー)を呼び出します。
引　数	なし
リターン値	なし

Excep_RSPI0_SPEI0	
概　要	RSPI0.SPEI0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPEI0(void)
説　明	RSPI エラー割り込み処理を行います。
引　数	なし
リターン値	なし

Excep_RSPI0_SPRI0	
概　要	RSPI0.SPRI0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPRI0(void)
説　明	RSPI 受信割り込み処理を行います。
引　数	なし
リターン値	なし

Excep_RSPI0_SPTI0	
概　要	RSPI0.SPTI0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPTI0(void)
説　明	RSPI 送信割り込み処理を行います。
引　数	なし
リターン値	なし

Excep_RSPI0_SPII0	
概　要	RSPI0.SPII0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPII0(void)
説　明	RSPI アイドル割り込み処理を行います。
引　数	なし
リターン値	なし

5.9 フローチャート

5.9.1 メイン処理

図 5.5にメイン処理のフローチャートを示します。

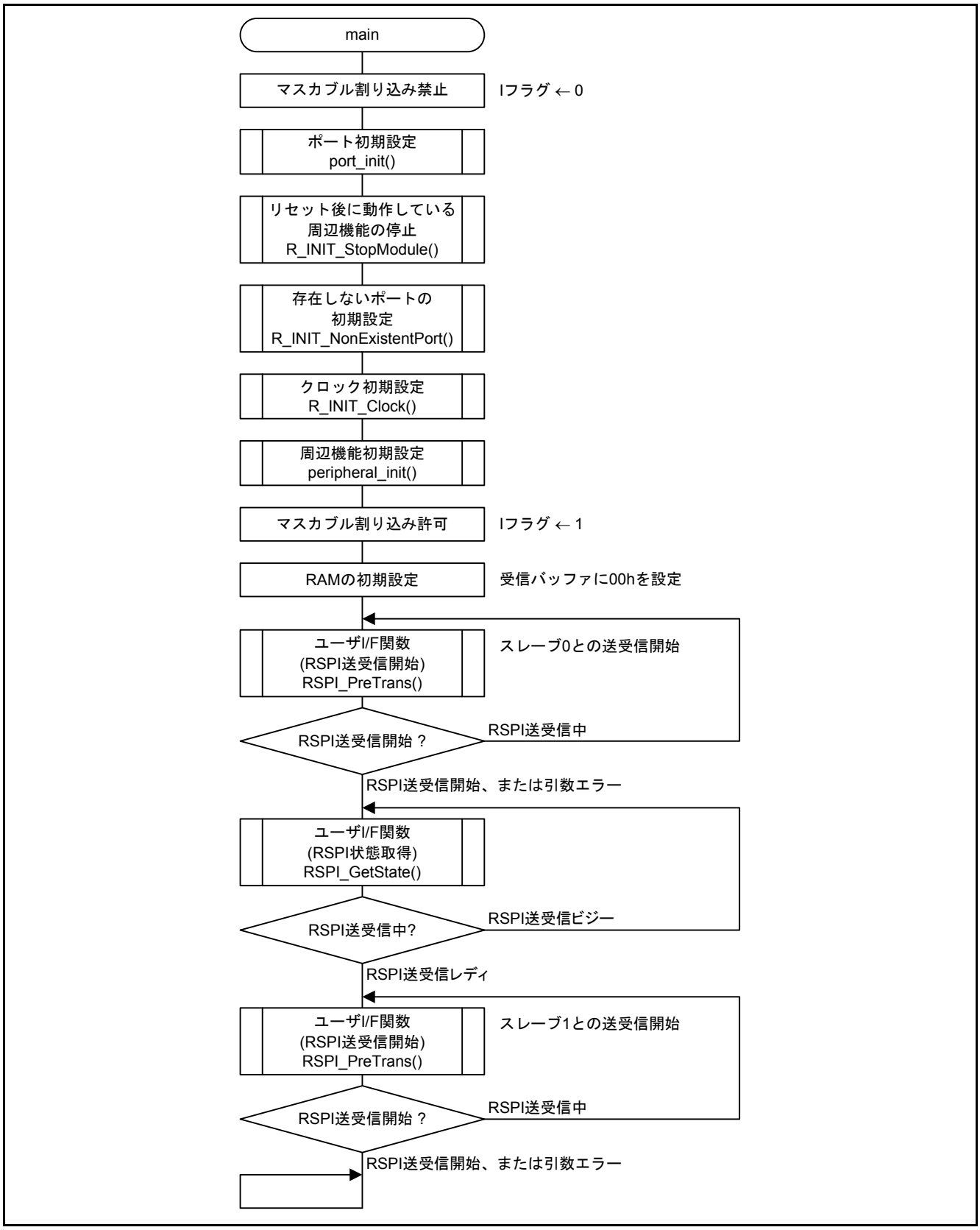


図5.5 メイン処理

5.9.2 ポート初期設定

図 5.6にポート初期設定のフローチャートを示します。

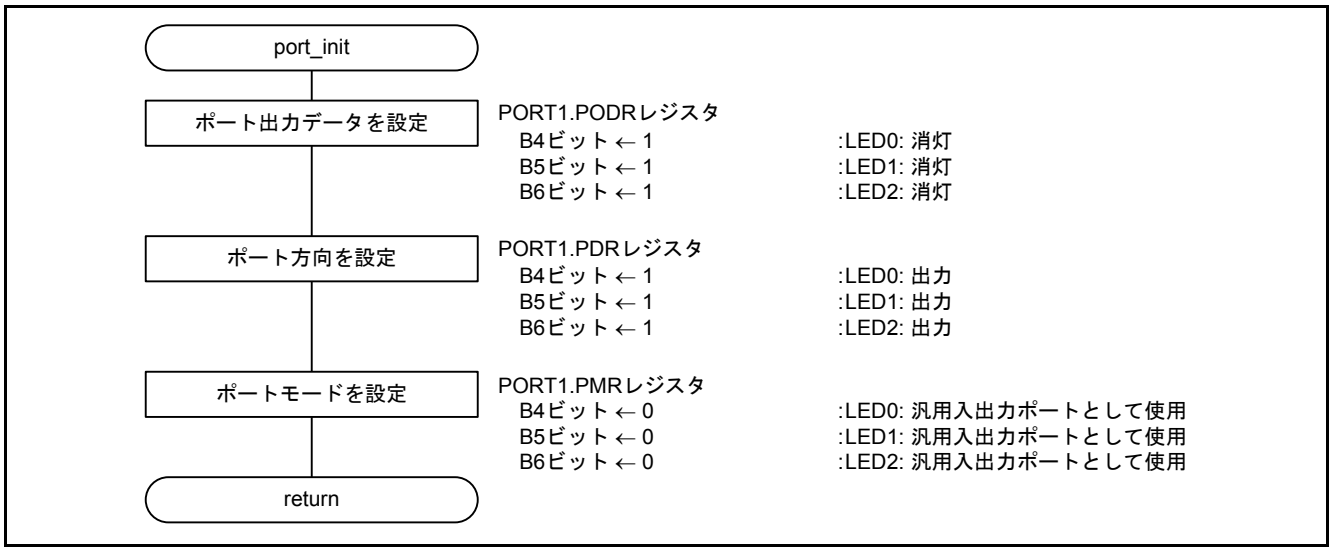


図5.6 ポート初期設定

5.9.3 周辺機能初期設定

図 5.7に周辺機能初期設定のフローチャートを示します。

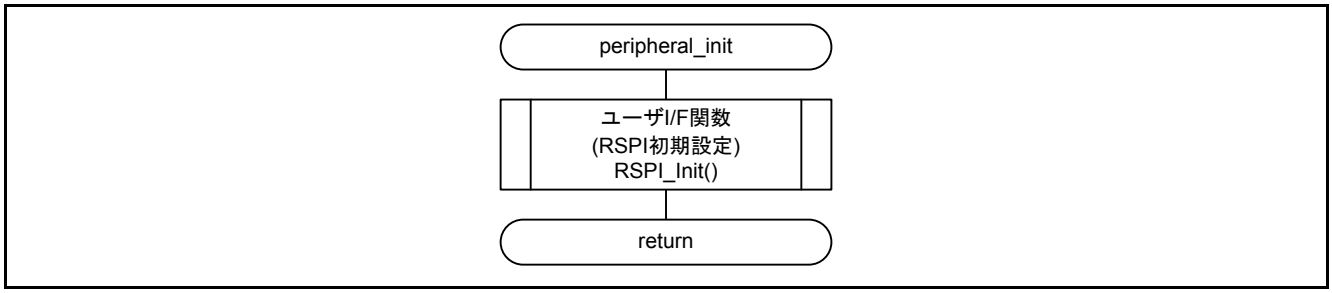


図5.7 周辺機能初期設定

5.9.4 コールバック関数(スレーブ 0 への RSPI 送受信完了)

図 5.8にコールバック関数(スレーブ 0 への RSPI 送受信完了)のフローチャートを示します。

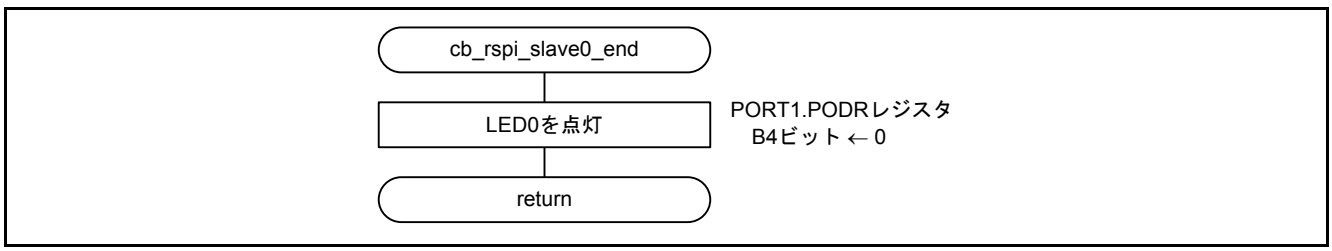


図5.8 コールバック関数(スレーブ 0 への RSPI 送受信完了)

5.9.5 コールバック関数(スレーブ 1 への RSPI 送受信完了)

図 5.9にコールバック関数(スレーブ 1 への RSPI 送受信完了)のフローチャートを示します。

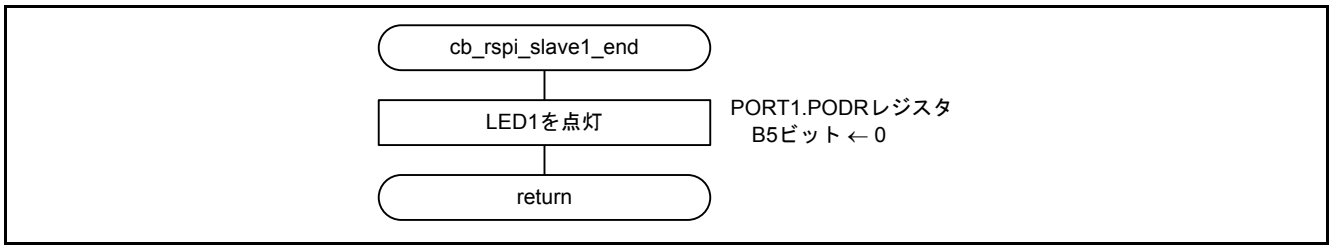
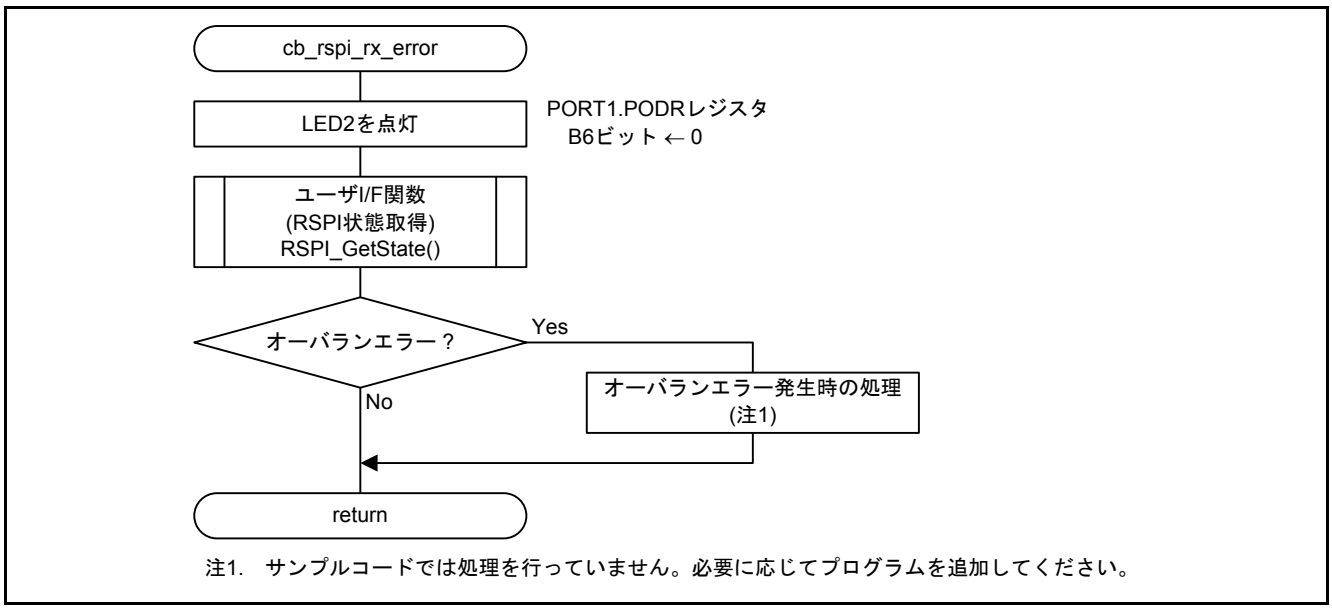


図5.9 コールバック関数(スレーブ 1 への RSPI 送受信完了)

5.9.6 コールバック関数(RSPI 送受信エラー)

図 5.10にコールバック関数(RSPI 送受信エラー)のフローチャートを示します。



注1. サンプルコードでは処理を行っていません。必要に応じてプログラムを追加してください。

図5.10 コールバック関数(RSPI 送受信エラー)

5.9.7　ユーザ I/F 関数(RSPI 初期設定)

図 5.11、図 5.12にユーザ I/F 関数(RSPI 初期設定)のフローチャートを示します。

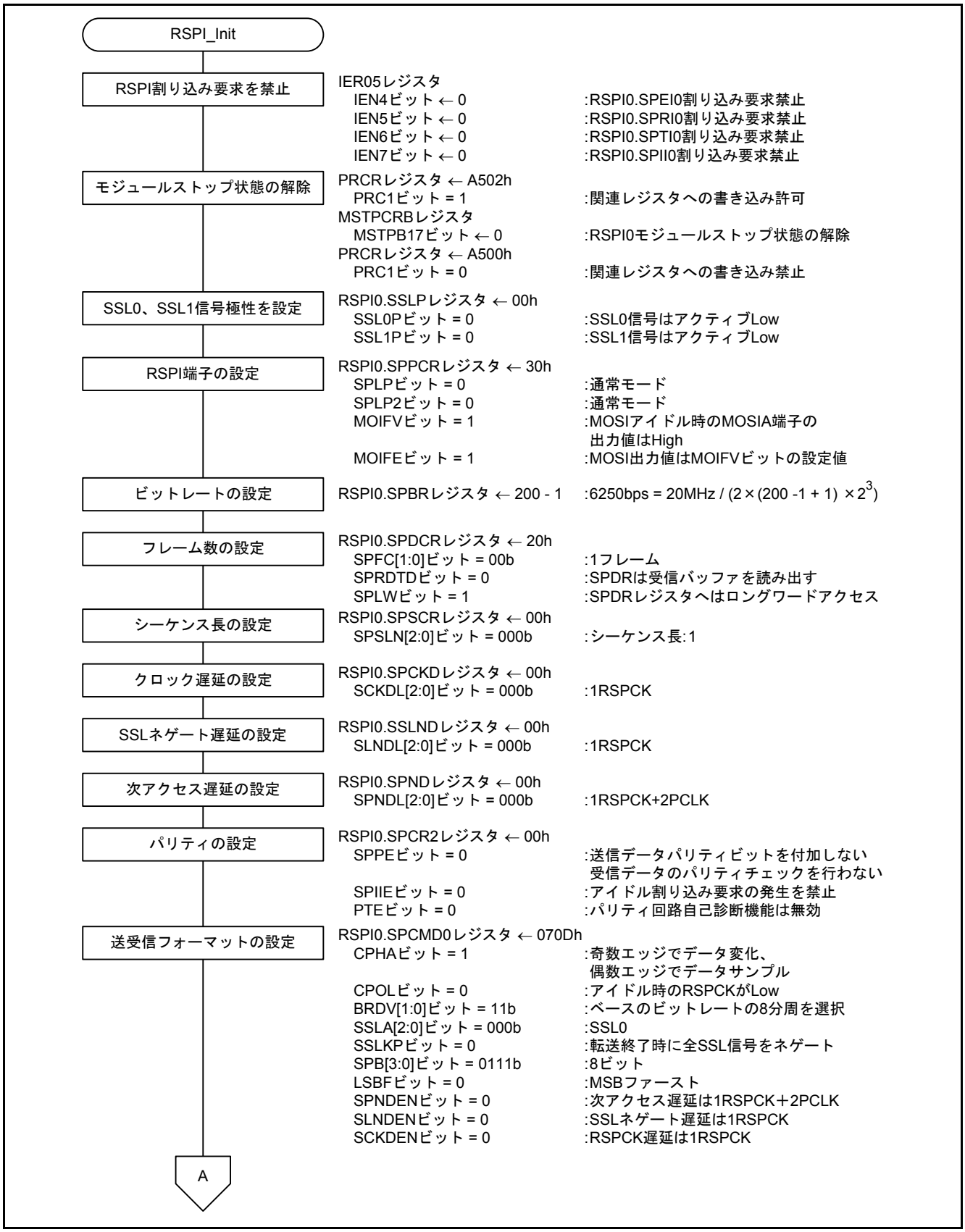


図5.11　ユーザ I/F 関数(RSPI 初期設定) (1/2)

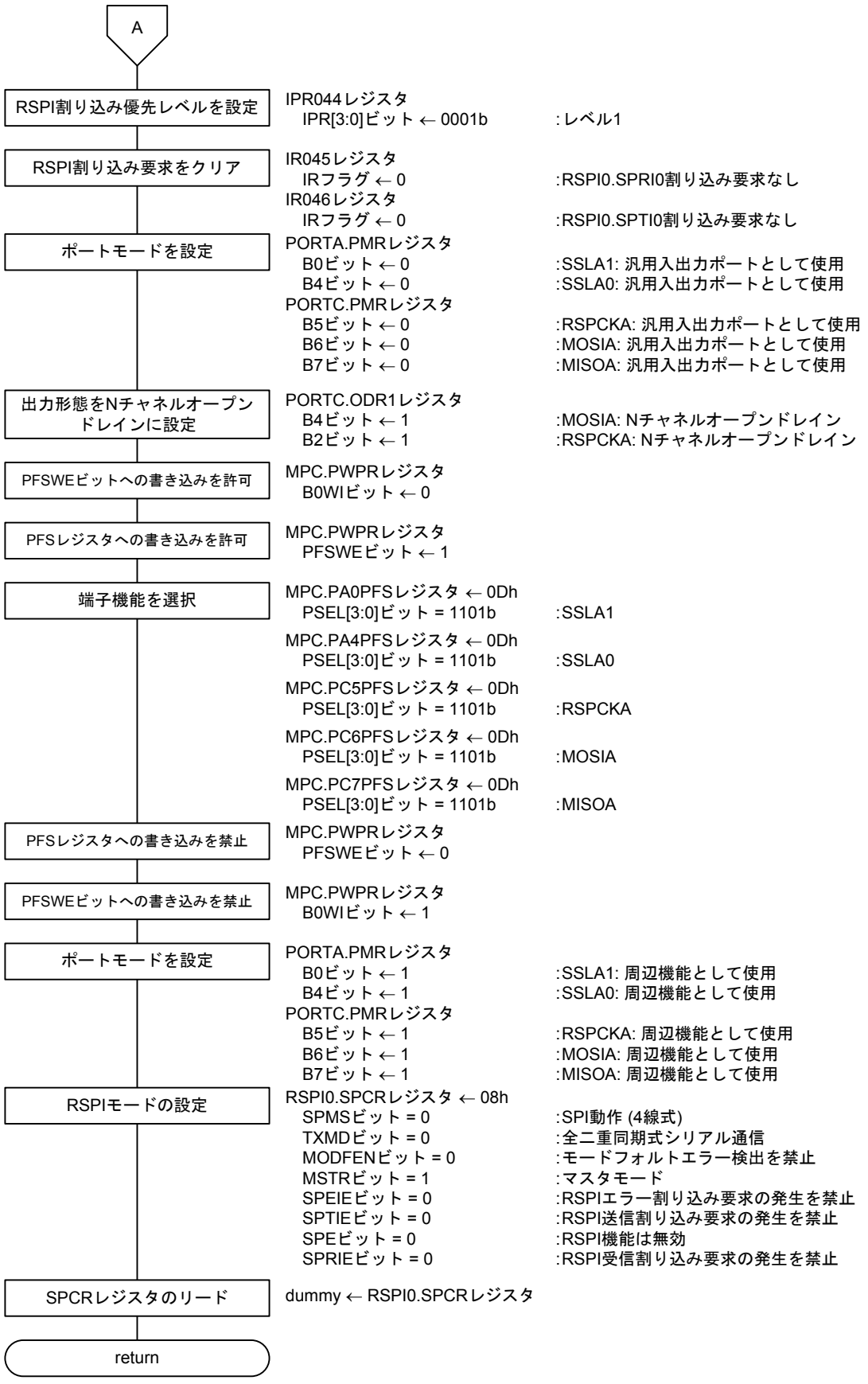


図5.12 ユーザ I/F 関数(RSPI 初期設定) (2/2)

5.9.8 ユーザ I/F 関数(RSPI 送受信開始)

図 5.13、図 5.14にユーザ I/F 関数(RSPI 送受信開始)のフローチャートを示します。

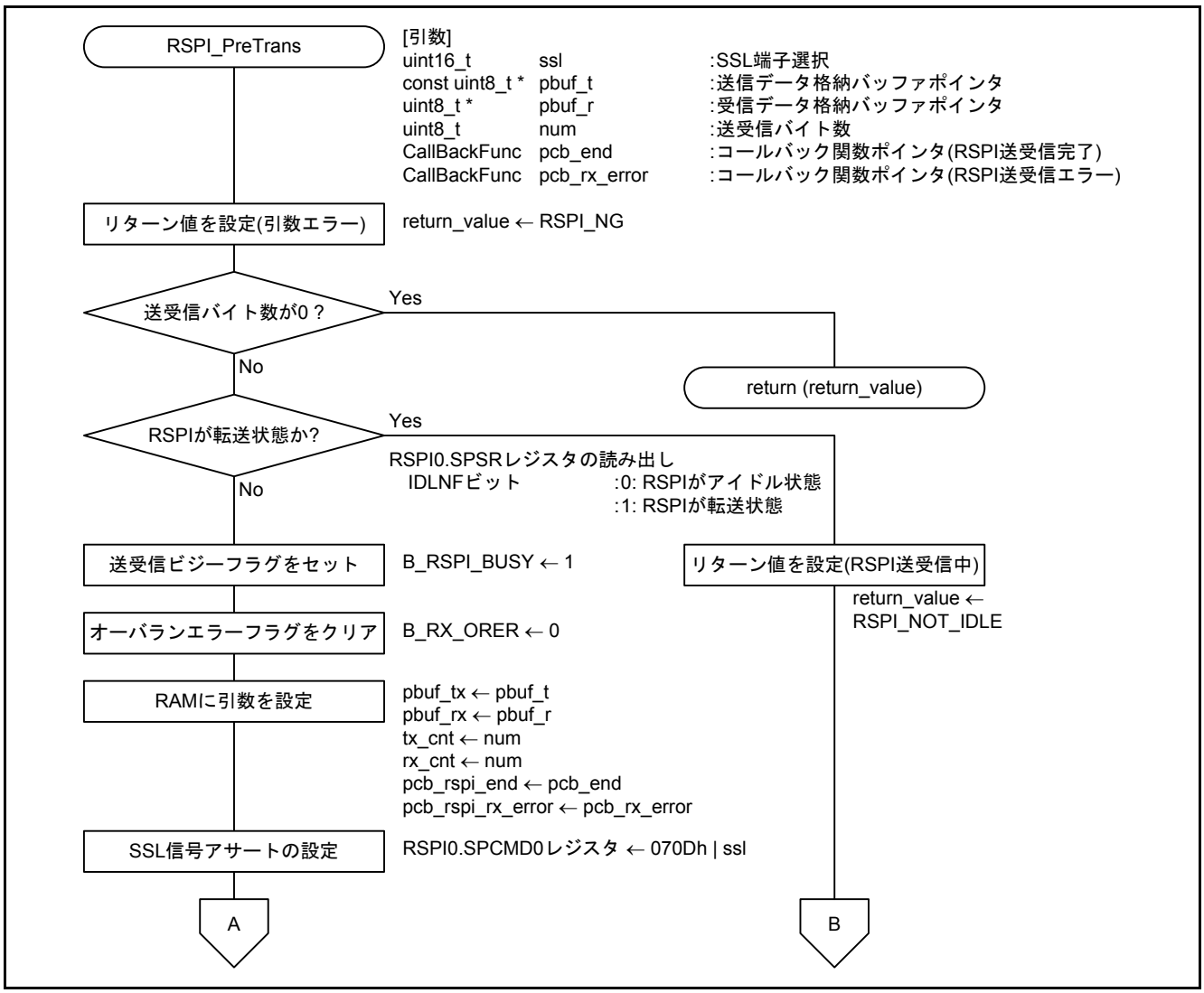
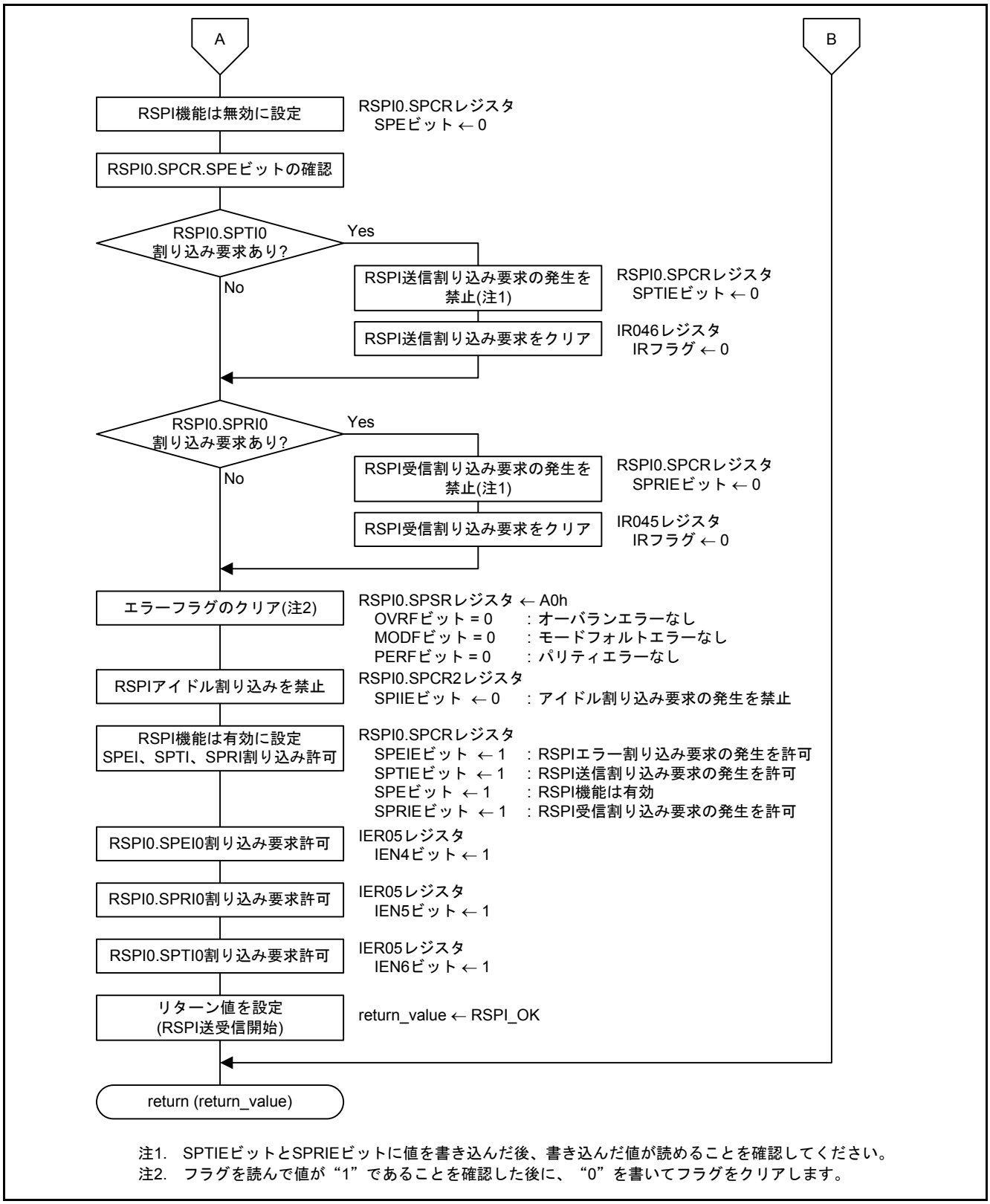


図5.13 ユーザ I/F 関数(RSPI 送受信開始) (1/2)



注1.　SPTIEビットとSPRIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。
注2.　フラグを読んで値が“1”であることを確認した後に、“0”を書いてフラグをクリアします。

図5.14　ユーザ I/F 関数(RSPI 送受信開始) (2/2)

5.9.9　ユーザ I/F 関数(RSPI 状態取得)

図 5.15にユーザ I/F 関数(RSPI 状態取得)のフローチャートを示します。

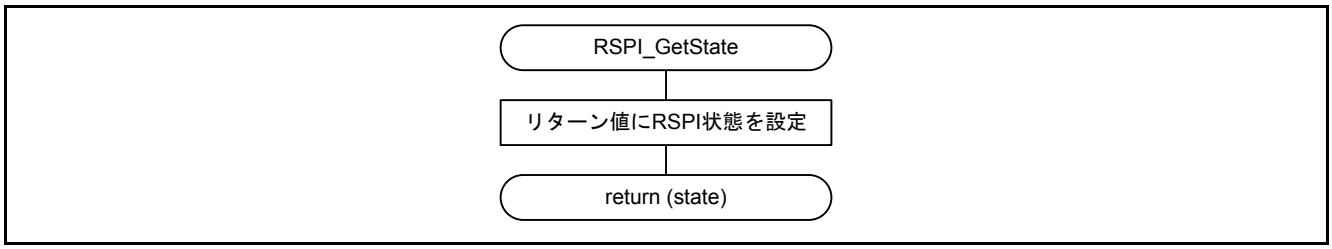
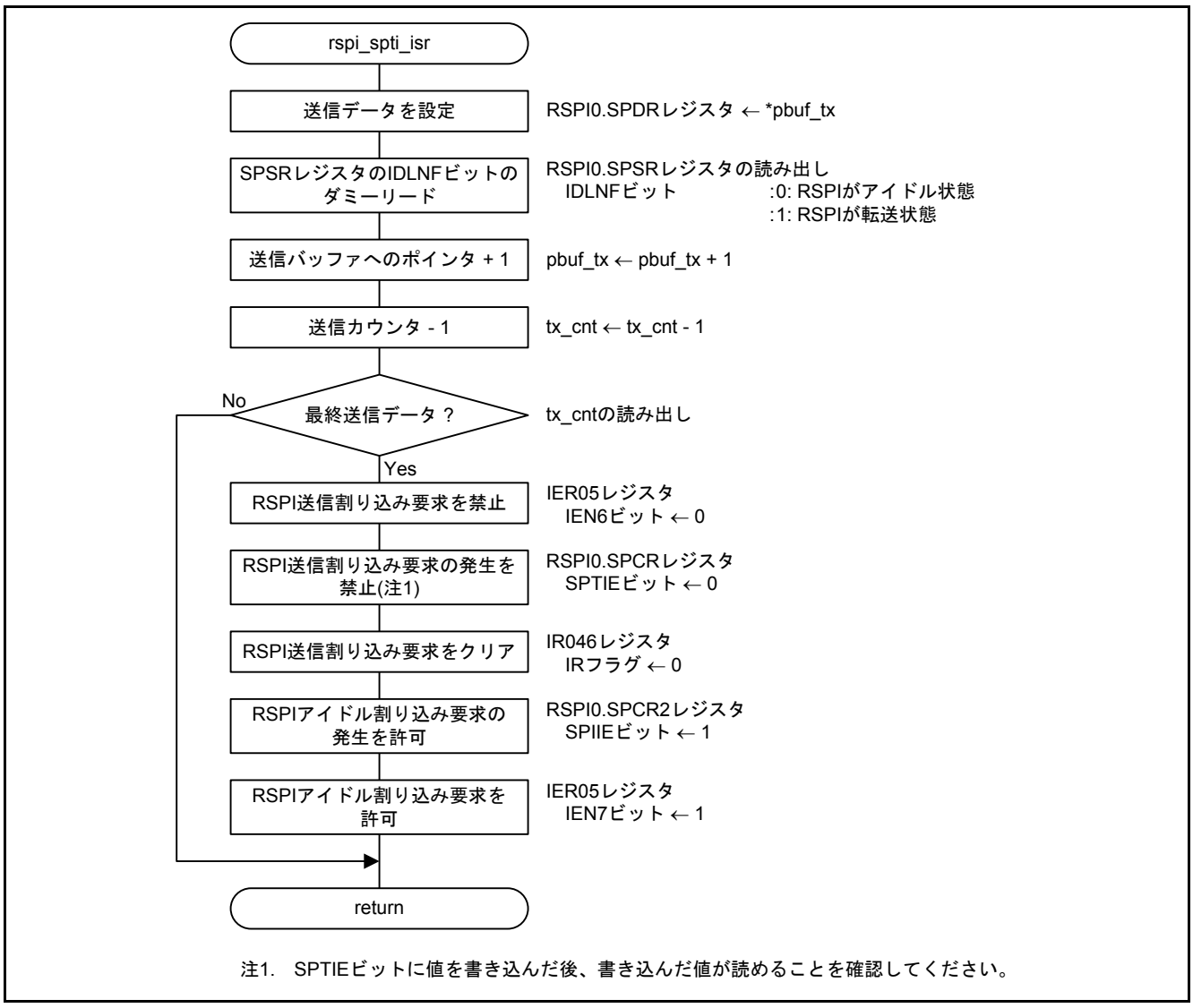


図5.15　ユーザ I/F 関数(RSPI 状態取得)

5.9.10　RSPI 送信割り込み

図 5.16にRSPI 送信割り込みのフローチャートを示します。

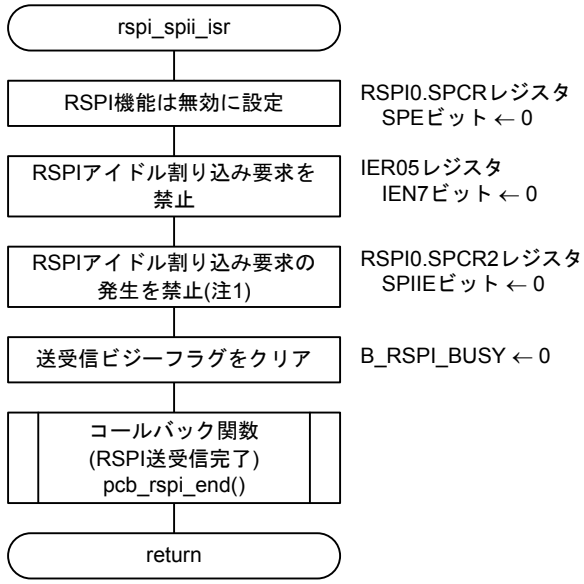


注1.　SPTIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。

図5.16　RSPI 送信割り込み

5.9.11 RSPI アイドル割り込み

図 5.17にRSPI アイドル割り込みのフローチャートを示します。

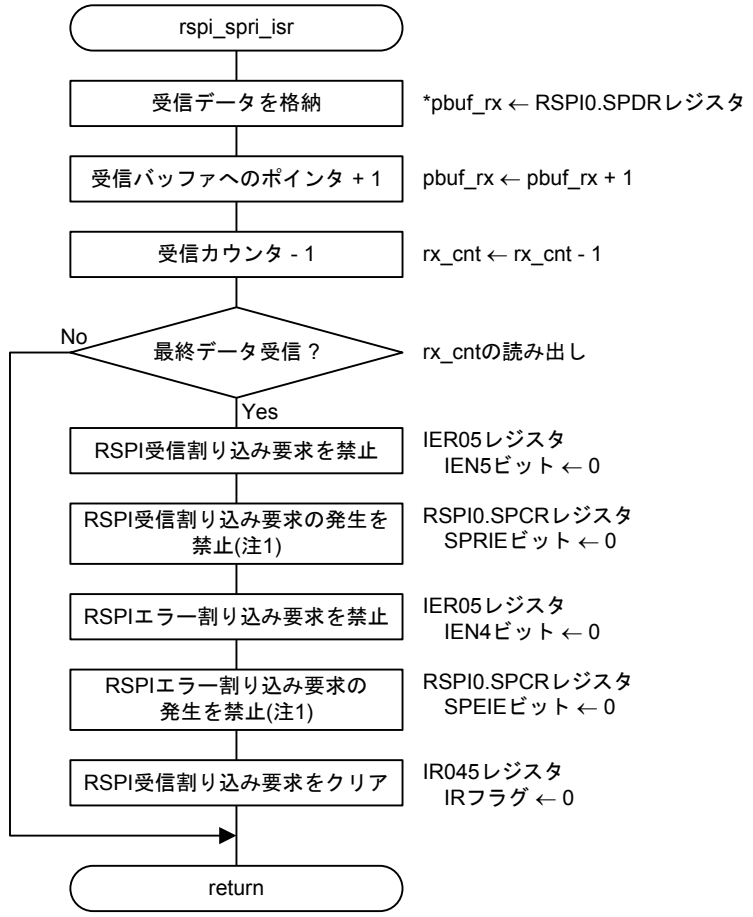


注1. SPIIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。

図5.17 RSPI アイドル割り込み

5.9.12 RSPI 受信割り込み

図 5.18にRSPI 受信割り込みのフローチャートを示します。

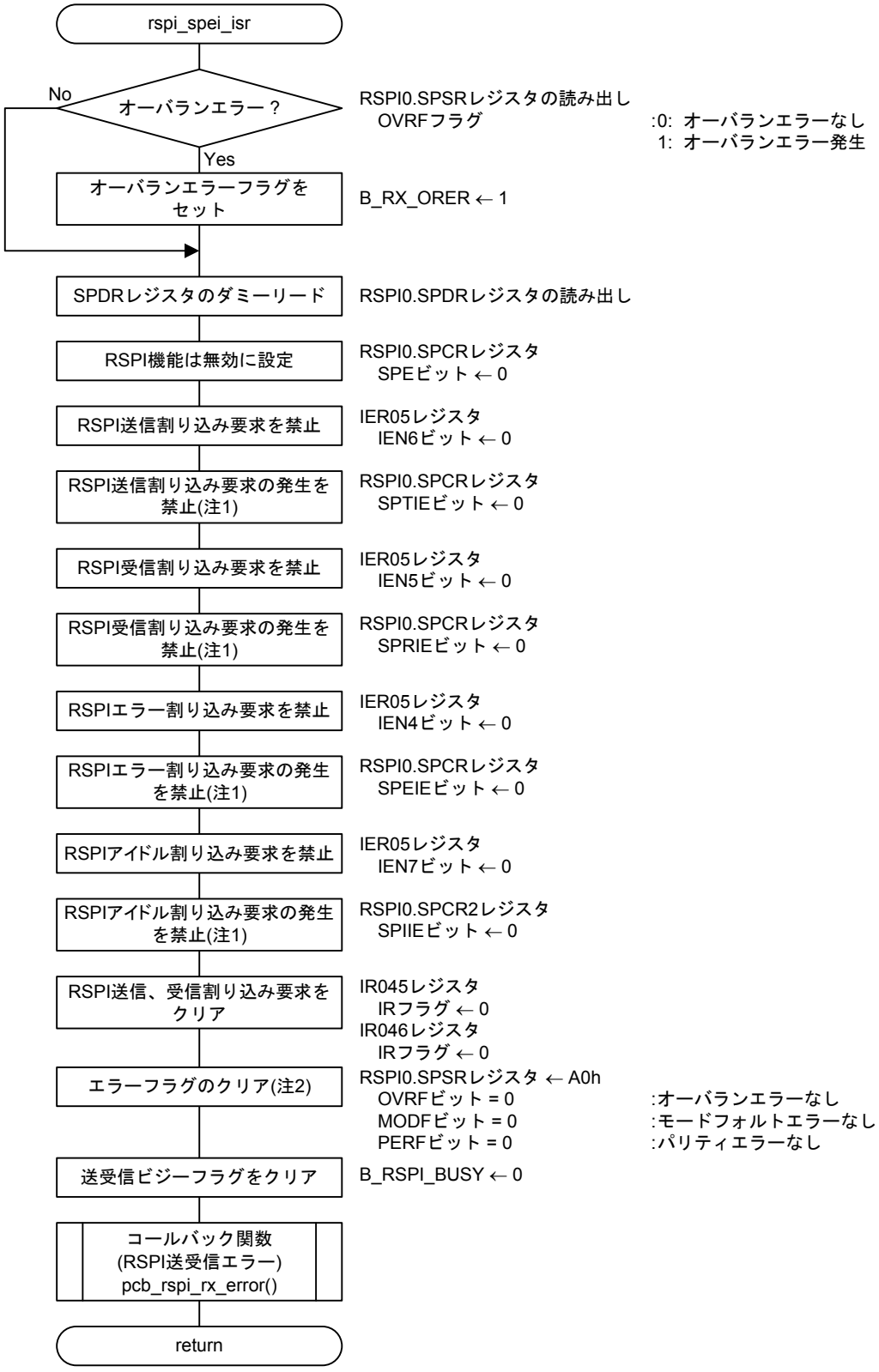


注1. SPRIEビット、SPEIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。

図5.18 RSPI 受信割り込み

5.9.13 RSPI エラー割り込み

図 5.19にRSPI エラー割り込みのフローチャートを示します。



注1. SPTIEビット、SPRIEビット、SPEIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。
注2. フラグを読んで値が“1”であることを確認した後に、“0”を書いてフラグをクリアします。

図5.19 RSPI エラー割り込み

5.9.14 RSPI0.SPEI0 割り込み処理

図 5.20にRSPI0.SPEI0 割り込み処理のフローチャートを示します。

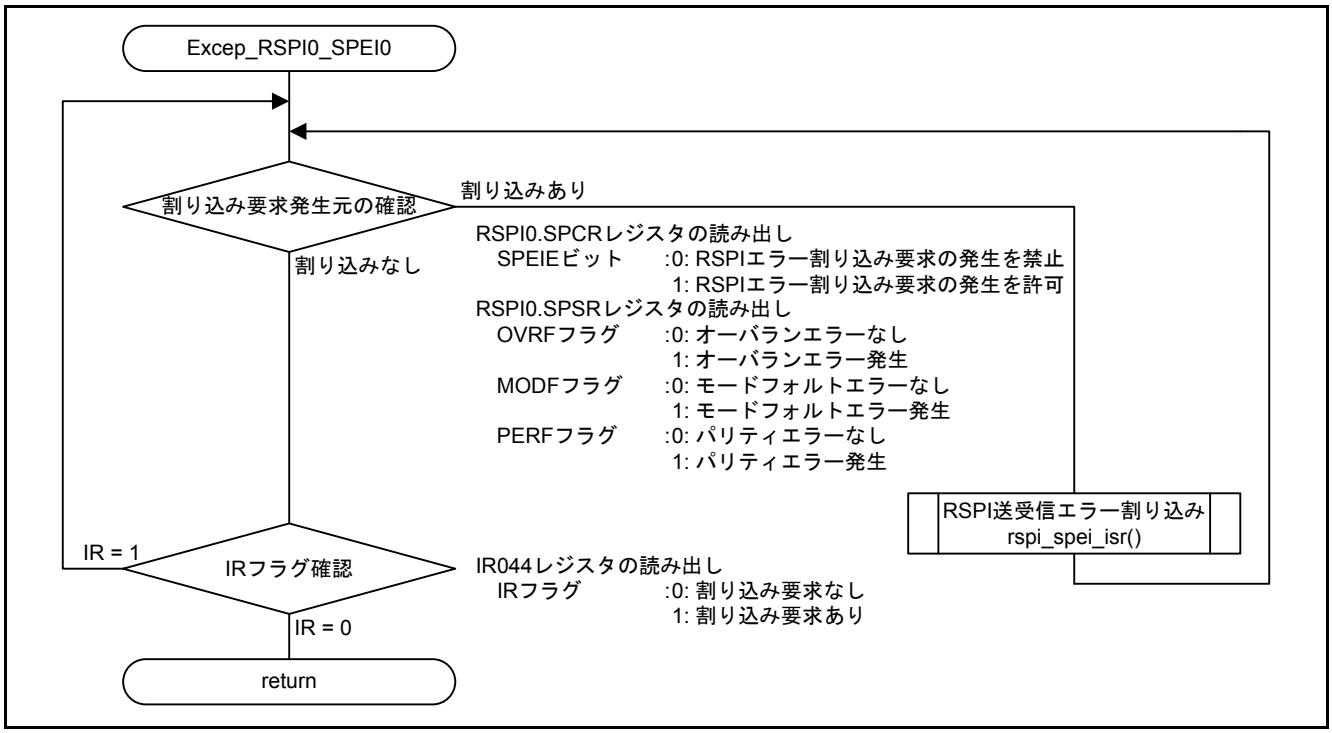


図5.20 RSPI0.SPEI0 割り込み処理

5.9.15 RSPI0.SPRI0 割り込み処理

図 5.21にRSPI0.SPRI0 割り込み処理のフローチャートを示します。

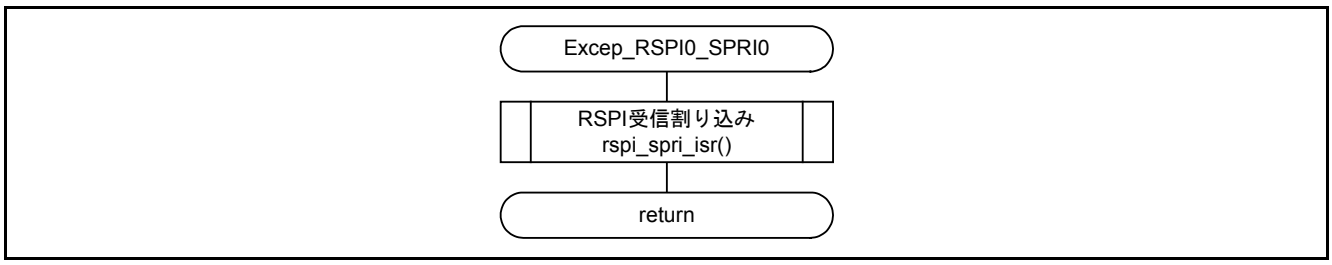


図5.21 RSPI0.SPRI0 割り込み処理

5.9.16 RSPI0.SPTI0 割り込み処理

図 5.22にRSPI0.SPTI0 割り込み処理のフローチャートを示します。

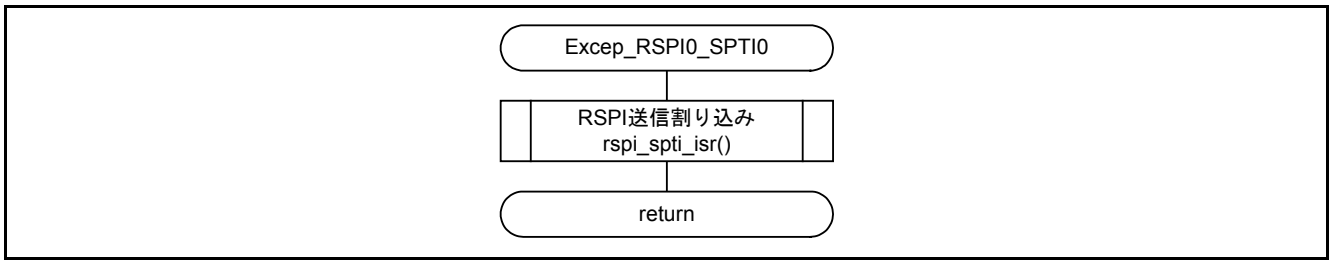


図5.22 RSPI0.SPTI0 割り込み処理

5.9.17 RSPI0.SPII0 割り込み処理

図 5.23にRSPI0.SPII0 割り込み処理のフローチャートを示します。

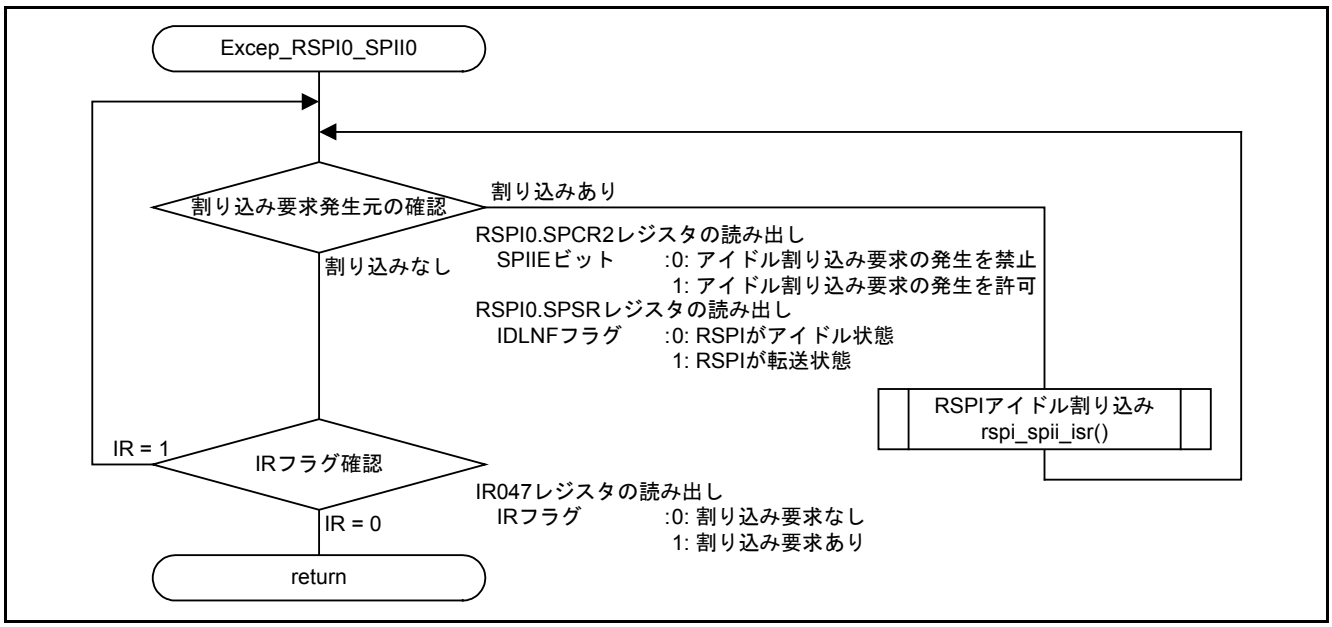


図5.23 RSPI0.SPII0 割り込み処理

6. ハードウェア説明(スレーブ)

6.1 使用端子一覧

表 6.1に使用端子と機能を示します。

使用端子は、100 ピン版の製品を想定しています。100 ピン版未満の製品を使用する場合は、使用する製品に合わせて端子を選択してください。

表6.1 使用端子と機能

端子名	入出力	内容
P15	出力	LED1 出力(マスタとの RSPI 送受信完了)
P16	出力	LED2 出力(RSPI 送受信エラー)
PA4/SSLA0	入力	スレーブセレクト信号入力
PC5/RSPCKA	入力	クロック入力端子
PC6/MOSIA	入力	データ入力端子
PC7/MISOA	出力	データ出力端子

7.　ソフトウェア説明(スレーブ)

リセット解除後、ユーザ I/F 関数(RSPI 初期設定)を呼び出し、RSPI の初期化を行います。

初期化後、ユーザ I/F 関数(RSPI 送受信開始)を呼び出し、送受信動作を許可します。3 バイトの送受信を完了したとき、RSPI の送受信動作を禁止して、コールバック関数(マスタとの RSPI 送受信完了)を呼び出します。コールバック関数(マスタとの RSPI 送受信完了)で、LED1 を点灯します。

送受信エラーが発生した場合、RSPI の送受信動作を禁止して、コールバック関数(RSPI 送受信エラー)を呼び出します。コールバック関数(RSPI 送受信エラー)で、LED2 を点灯します。

以下に使用する周辺機能の設定を、図 7.1にソフトウェア構成を示します。

＜RSPI＞

- RSPI モード　　: SPI 動作(4 線式)
- 通信動作モード　　: 全二重同期式シリアル通信
- クロックソース　　: PCLKB (20MHz)
- 転送ビット長　　: 8 ビット
- パリティ　　: なし
- シーケンス長　　: 1 シーケンス
- フレーム数　　: 1 フレーム
- エラー検出　　: オーバランエラー
  : モードフォルトエラー
- 割り込み要因　　: RSPI エラー割り込み(SPEI)を許可
  : RSPI 受信割り込み(SPRI)を許可
  : RSPI 送信割り込み(SPTI)を許可
  : RSPI アイドル割り込み(SPII)を禁止

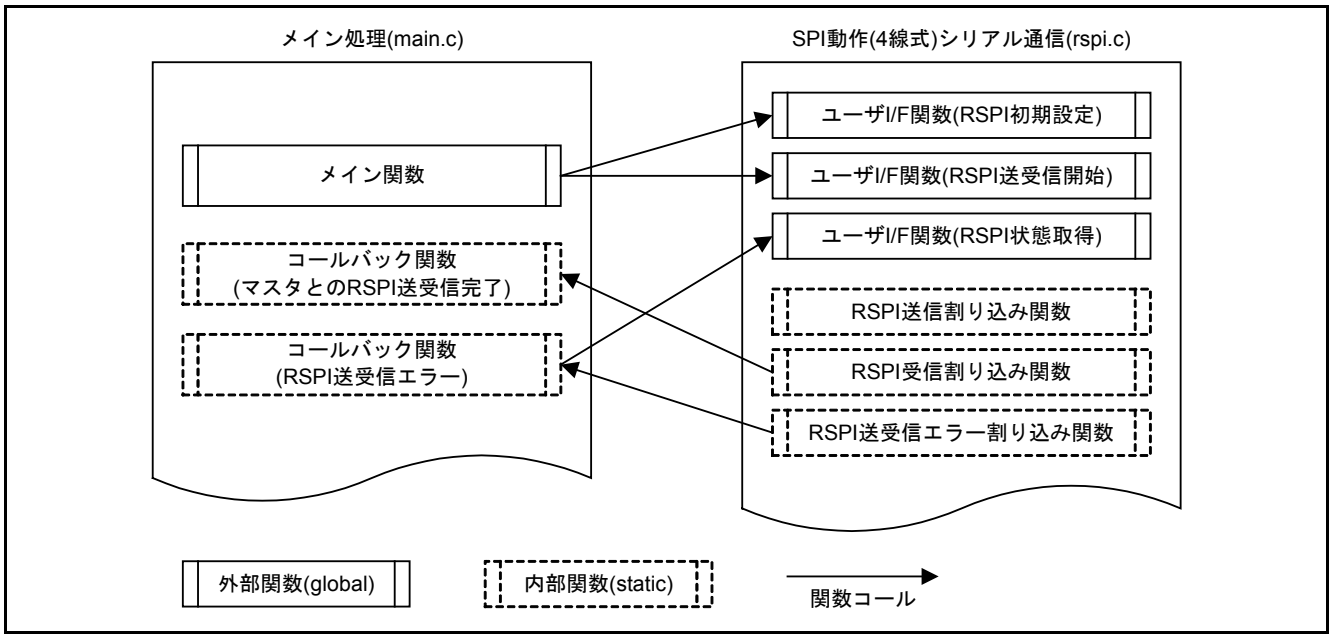


図7.1　ソフトウェア構成

7.1 動作概要

図 7.2にSPI 動作(4 線式)シリアル通信のタイミング図を、以下に図中の番号の動作および処理を示します。

(1) 初期設定
ユーザ I/F 関数(RSPI 初期設定)で RSPI の初期設定を行います。

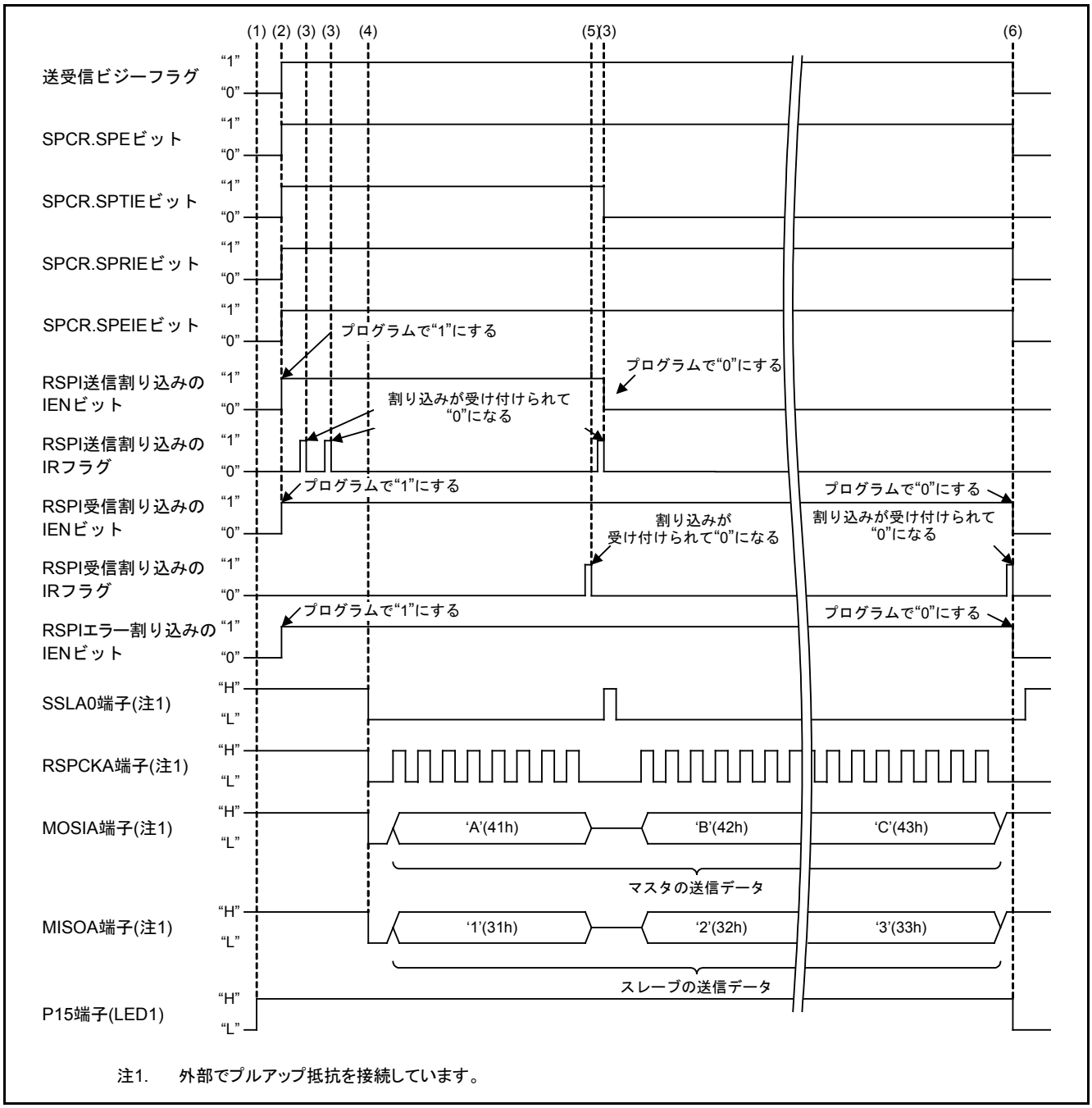
(2) マスタとの送受信開始
ユーザ I/F 関数(RSPI 送受信開始)では、最初に SPSR.IDLNF ビットを確認します。“1”(RSPI が転送状態)の場合、RSPI_BUSY (RSPI 送受信中)を返します。“0”(RSPI がアイドル状態)の場合、送受信ビジーフラグを“1”にします。SPCR.SPEIE ビットを“1”(RSPI エラー割り込み要求の発生を許可)、SPCR.SPTIE ビットを“1”(RSPI 送信割り込み要求の発生を許可)、SPCR.SPE ビットを“1”(RSPI 機能は有効)、SPCR.SPRIE ビット“1”(RSPI 受信割り込みの要求の発生を許可)にします。RSPI エラー、RSPI 受信、RSPI 送信割り込みの IEN ビットを“1”にします。その後、マスタからの SSLA0 入力信号と RSPCKA エッジの入力を待ちます。

(3) 送信データの設定
RSPI 送信割り込み処理で、SPDR レジスタに送信バッファの値を書きます。最終データを書いたら、SPTIE ビットを“0”(RSPI 送信割り込みの発生を禁止)にします。

(4) マスタとの送受信
マスタから SSLA0 入力信号と RSPCKA エッジ入力が確認されると、データ送受信を行います。

(5) マスタからデータ受信
RSPI 受信割り込み処理で、受信バッファに SPDR レジスタの値を書きます。

(6) マスタとの送受信完了
最終データを受信したら、SPE ビットを“0”(RSPI 機能は無効)、SPRIE ビットを“0”(RSPI 受信割り込みの要求の発生を禁止)、SPEIE ビットを“0”(RSPI エラー割り込みの発生を禁止)にします。送受信ビジーフラグを“0”にして、コールバック関数(マスタとの RSPI 送受信完了)を呼び出し、LED1 を点灯します。



注1. 外部でプルアップ抵抗を接続しています。

図7.2 SPI 動作(4 線式)シリアル通信のタイミング図

7.2 ファイル構成

表 7.1にサンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表7.1 サンプルコードで使用するファイル

ファイル名	概要	備考
main.c	メイン処理	
r_init_stop_module.c	リセット後に動作している周辺機能の停止	
r_init_stop_module.h	r_init_stop_module.c のヘッダファイル	
r_init_non_existent_port.c	存在しないポートの初期設定	
r_init_non_existent_port.h	r_init_non_existent_port.c のヘッダファイル	
r_init_clock.c	クロック初期設定	
r_init_clock.h	r_init_clock.c のヘッダファイル	
rspi.c	SPI 動作(4 線式)シリアル通信	
rspi.h	rspi.c のヘッダファイル	

7.3 オプション設定メモリ

表 7.2にサンプルコードで使用するオプション設定メモリの状態を示します。必要に応じて、お客様のシステムに最適な値を設定してください。

表7.2 サンプルコードで使用するオプション設定メモリ

シンボル	アドレス	設定値	内容
OFS0	FFFF FF8Fh～FFFF FF8Ch	FFFF FFFFh	リセット後、IWDT は停止
OFS1	FFFF FF8Bh～FFFF FF88h	FFFF FFFFh	リセット後、電圧監視 0 リセット無効 リセット後、HOCO 発振が無効
MDES	FFFF FF83h～FFFF FF80h	FFFF FFFFh	リトルエンディアン

7.4 定数一覧

表 7.3～表 7.5にサンプルコードで使用する定数(main.c)を示します。

表7.3 サンプルコードで使用する定数(main.c)

定数名	設定値	内容
LED1_REG_PODR	PORT1.PODR.BIT.B5	LED1 出力データ格納ビット
LED1_REG_PDR	PORT1.PDR.BIT.B5	LED1 方向制御ビット
LED1_REG_PMR	PORT1.PMR.BIT.B5	LED1 端子モード制御ビット
LED2_REG_PODR	PORT1.PODR.BIT.B6	LED2 出力データ格納ビット
LED2_REG_PDR	PORT1.PDR.BIT.B6	LED2 方向制御ビット
LED2_REG_PMR	PORT1.PMR.BIT.B6	LED2 端子モード制御ビット
LED_ON	0	LED 出力データ: 点灯
LED_OFF	1	LED 出力データ: 消灯
TR_SIZE	3	送受信サイズ
BUF_SIZE	TR_SIZE	バッファサイズ

表7.4　サンプルコードで使用する定数(rspi.c)

定数名	設定値	内容
SPSR_ERROR_FLAGS	0Dh	RSPI.SPSR レジスタのエラーフラグのビットパターン
B_RSPI_BUSY	state.bit.b_rspi_busy	送受信ビジーフラグ 0: 送受信レディ 1: 送受信ビジー
B_RX_ORER	state.bit.b_rx_orer	オーバランエラーフラグ 0: オーバランエラーなし 1: オーバランエラーあり
B_RX_MODF	state.bit.b_rx_modf	モードフォルトエラーフラグ 0: モードフォルトエラーなし 1: モードフォルトエラーあり

表7.5　サンプルコードで使用する定数(rspi.h)

定数名	設定値	内容
RSPI_OK	00h	RSPI_PreTrans 関数のリターン値: RSPI 送受信開始
RSPI_NOT_IDLE	01h	RSPI_PreTrans 関数のリターン値: RSPI 送受信中
RSPI_NG	02h	RSPI_PreTrans 関数のリターン値: 引数エラー

7.5 構造体/共用体一覧

図 7.3にサンプルコードで使用する構造体/共用体を示します。

```
#pragma bit_order     left            /* ビットフィールドの並び順指定: 上位ビット側からメンバを割り付ける*/
#pragma unpack                        /* 構造体メンバの境界調整数指定: メンバの型でアライメントする       */
typedef   union
{
    uint8_t    byte;
    struct
    {
        uint8_t    b_rspi_busy :1;    /*  送受信ビジーフラグ              0:送受信レディ 1:送受信ビジー          */
        uint8_t    b_rx_orer   :1;    /*  オーバランエラーフラグ          0:エラーなし   1:オーバランエラー      */
        uint8_t    b_rx_modf   :1;    /*  モードフォルトエラーフラグ      0:エラーなし   1:モードフォルトエラー  */
        uint8_t                :5;    /*  使用しない                                                             */
    } bit;
} rspi_state_t;
#pragma packoption                    /* 構造体メンバの境界調整数指定の終了 */
#pragma bit_order                     /* ビットフィールドの並び順指定の終了 */
```

図7.3　サンプルコードで使用する構造体/共用体

7.6 変数一覧

表 7.6にstatic 型変数を示します。

表7.6　static 型変数

型	変数名	内容	使用関数
static uint8_t	tx_buf[]	送信バッファ	main
static uint8_t	rx_buf[BUF_SIZE]	受信バッファ	main
static const uint8_t *	pbuf_tx	送信バッファへのポインタ	RSPI_PreTrans rspi_spti_isr
static uint8_t	tx_cnt	送信カウンタ	
static uint8_t *	pbuf_rx	受信バッファへのポインタ	RSPI_PreTrans rspi_spri_isr
static uint8_t	rx_cnt	受信カウンタ	
static rspi_state_t	state	RSPI 状態	RSPI_PreTrans RSPI_GetState rspi_spri_isr rspi_spei_isr

7.7　関数一覧

表 7.7にサンプルコードで使用する関数を示します。

表7.7　サンプルコードで使用する関数

関数名	概要
main	メイン処理
port_init	ポート初期設定
R_INIT_StopModule	リセット後に動作している周辺機能の停止
R_INIT_NonExistentPort	存在しないポートの初期設定
R_INIT_Clock	クロック初期設定
peripheral_init	周辺機能初期設定
cb_rspi_end	コールバック関数(マスタとの RSPI 送受信完了)
cb_rspi_rx_error	コールバック関数(RSPI 送受信エラー)
RSPI_Init	ユーザ I/F 関数(RSPI 初期設定)
RSPI_PreTrans	ユーザ I/F 関数(RSPI 送受信開始)
RSPI_GetState	ユーザ I/F 関数(RSPI 状態取得)
rspi_spti_isr	RSPI 送信割り込み
rspi_spri_isr	RSPI 受信割り込み
rspi_spei_isr	RSPI エラー割り込み
Excep_RSPI0_SPEI0	RSPI0.SPEI0 割り込み処理
Excep_RSPI0_SPRI0	RSPI0.SPRI0 割り込み処理
Excep_RSPI0_SPTI0	RSPI0.SPTI0 割り込み処理

7.8　関数仕様

サンプルコードの関数仕様を示します。

main	
概　要	メイン処理
ヘッダ	なし
宣　言	void main(void)
説　明	初期設定後、マスタからの SSLA0 入力信号待ち状態になります。SSLA0 入力信号が入力され、RSPCKA エッジを検出すると RSPI 送受信を開始します。
引　数	なし
リターン値	なし

port_init	
概　要	ポート初期設定
ヘッダ	なし
宣　言	static void port_init(void)
説　明	ポートの初期設定を行います。
引　数	なし
リターン値	なし

R_INIT_StopModule	
概　要	リセット後に動作している周辺機能の停止
ヘッダ	r_init_stop_module.h
宣　言	void R_INIT_StopModule(void)
説　明	モジュールストップ状態へ遷移する設定を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、モジュールストップ状態への遷移は行っていません。 本関数の詳細は、アプリケーションノート「RX220グループ 初期設定例 Rev.1.00」を参照してください。

R_INIT_NonExistentPort	
概　要	存在しないポートの初期設定
ヘッダ	r_init_non_existent_port.h
宣　言	void R_INIT_NonExistentPort(void)
説　明	100 ピン未満の製品に対して、存在しないポートの端子に対応するポート方向レジスタの初期設定を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、100 ピン版(PIN_SIZE=100)に設定しています。 本関数をコールした後に、存在しないポートを含む PDR、PODR レジスタへバイト単位で書き込む場合、存在しないポートの方向制御ビットには“1”、ポート出力データ格納ビットには“0”を設定してください。 本関数の詳細は、アプリケーションノート「RX220グループ 初期設定例 Rev.1.00」を参照してください。

R_INIT_Clock	
概　要	クロック初期設定
ヘッダ	r_init_clock.h
宣　言	void R_INIT_Clock(void)
説　明	クロックの初期設定を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、システムクロックを PLL とし、サブクロックを使用しない処理を選択しています。 本関数の詳細は、アプリケーションノート「RX220グループ 初期設定例 Rev.1.00」を参照してください。

peripheral_init	
概　要	周辺機能初期設定
ヘッダ	なし
宣　言	static void peripheral_init(void)
説　明	使用する周辺機能の初期設定を行います。
引　数	なし
リターン値	なし

cb_rspi_end

概　要	コールバック関数(マスタとの RSPI 送受信完了)
ヘッダ	なし
宣　言	static void cb_rspi_end(void)
説　明	マスタとの RSPI 送受信完了時に呼び出されます。
引　数	なし
リターン値	なし

cb_rspi_rx_error

概　要	コールバック関数(RSPI 送受信エラー)
ヘッダ	なし
宣　言	static void cb_rspi_rx_error(void)
説　明	RSPI 送受信エラー発生時に呼び出されます。
引　数	なし
リターン値	なし
備　考	サンプルコードではエラー処理を行っていません。必要に応じてプログラムを追加してください。

RSPI_Init

概　要	ユーザ I/F 関数(RSPI 初期設定)
ヘッダ	rspi.h
宣　言	void RSPI_Init(void)
説　明	RSPI の初期設定を行います。
引　数	なし
リターン値	なし

RSPI_PreTrans

概　要	ユーザ I/F 関数(RSPI 送受信開始)		
ヘッダ	rspi.h		
宣　言	uint8_t RSPI_PreTrans(const uint8_t * pbuf_t, uint8_t * pbuf_r, uint8_t num, CallBackFunc pcb_end, CallBackFunc pcb_rx_error)		
説　明	RSPI がアイドル状態であるか確認します。RSPI 機能を有効、RSPI 送信割り込み、RSPI 受信割り込み、RSPI エラー割り込みを許可にして、マスタからの SSLA0 入力信号と RSPCKA エッジの入力を待ちます。		
引　数	const uint8_t *	pbuf_t	: 送信データ格納バッファへのポインタ
	uint8_t *	pbuf_r	: 受信データ格納バッファへのポインタ
	uint8_t	num	: 送受信バイト数
	CallBackFunc	pcb_end	: コールバック関数ポインタ(送受信完了)
	CallBackFunc	pcb_rx_error	: コールバック関数ポインタ(送受信エラー)
リターン値	RSPI_NG		: 引数エラー(送受信バイト数が 0)
	RSPI_NOT_IDLE		: RSPI 送受信中
	RSPI_OK		: RSPI 送受信開始

RSPI_GetState	
概　要	ユーザ I/F 関数(RSPI 状態取得)
ヘッダ	rspi.h
宣　言	rspi_state_t RSPI_GetState(void)
説　明	RSPI 状態を返します。
引　数	なし
リターン値	rspi_state_t.bit.b_rspi_busy　：送受信ビジーフラグ 0:送受信レディ 1:送受信ビジー rspi_state_t.bit.b_rx_orer　：オーバランエラーフラグ 0:エラーなし 1:オーバランエラー rspi_state_t.bit.b_rx_modf　：モードフォルトエラーフラグ 0:エラーなし 1:モードフォルトエラー

rspi_spti_isr	
概　要	RSPI 送信割り込み
ヘッダ	なし
宣　言	static void rspi_spti_isr(void)
説　明	RSPI0.SPTI0 割り込み処理関数で呼び出されます。送信データを書き込みます。最終データを送信すると、RSPI 送信割り込み要求の発生を禁止します。
引　数	なし
リターン値	なし

rspi_spri_isr	
概　要	RSPI 受信割り込み
ヘッダ	なし
宣　言	static void rspi_spri_isr(void)
説　明	RSPI0.SPRI0 割り込み処理関数から呼び出されます。受信データを格納します。最終データを受信すると、RSPI 機能を無効にして、コールバック関数(マスタとの RSPI 送受信完了)を呼び出します。
引　数	なし
リターン値	なし

rspi_spei_isr	
概　要	RSPI エラー割り込み
ヘッダ	なし
宣　言	static void rspi_spei_isr(void)
説　明	RSPI0.SPEI0 割り込み処理関数から呼び出されます。RSPI 機能を無効にして、コールバック関数(RSPI 送受信エラー)を呼び出します。
引　数	なし
リターン値	なし

Excep_RSPI0_SPEI0	
概　要	RSPI0.SPEI0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPEI0(void)
説　明	RSPI エラー割り込み処理を行います。
引　数	なし
リターン値	なし

Excep_RSPI0_SPRI0	
概　要	RSPI0.SPRI0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPRI0(void)
説　明	RSPI 受信割り込み処理を行います。
引　数	なし
リターン値	なし

Excep_RSPI0_SPTI0	
概　要	RSPI0.SPTI0 割り込み処理
ヘッダ	なし
宣　言	static void Excep_RSPI0_SPTI0(void)
説　明	RSPI 送信割り込み処理を行います。
引　数	なし
リターン値	なし

7.9　フローチャート

7.9.1　メイン処理

図 7.4にメイン処理のフローチャートを示します。

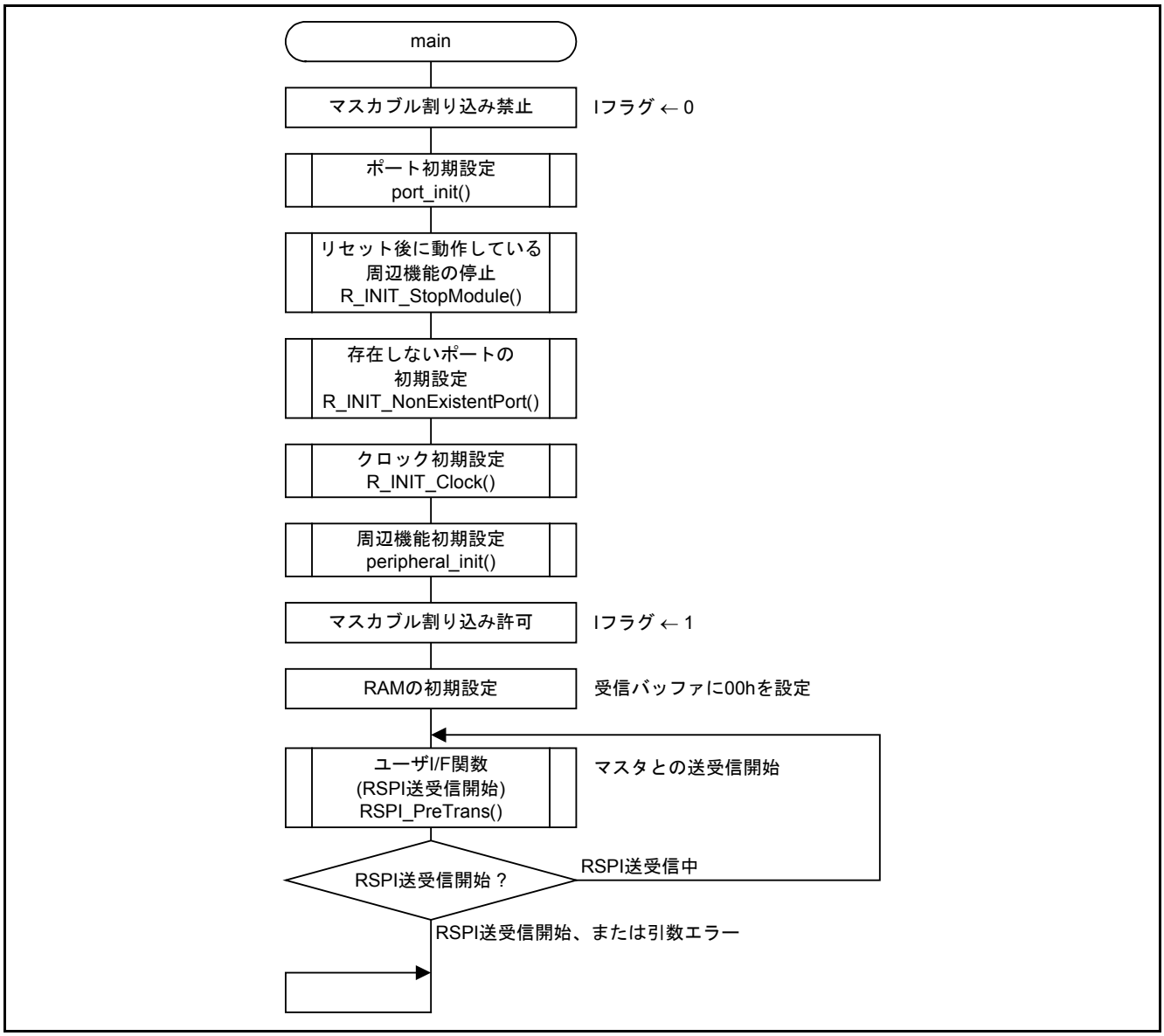


図7.4　メイン処理

7.9.2 ポート初期設定

図 7.5にポート初期設定のフローチャートを示します。

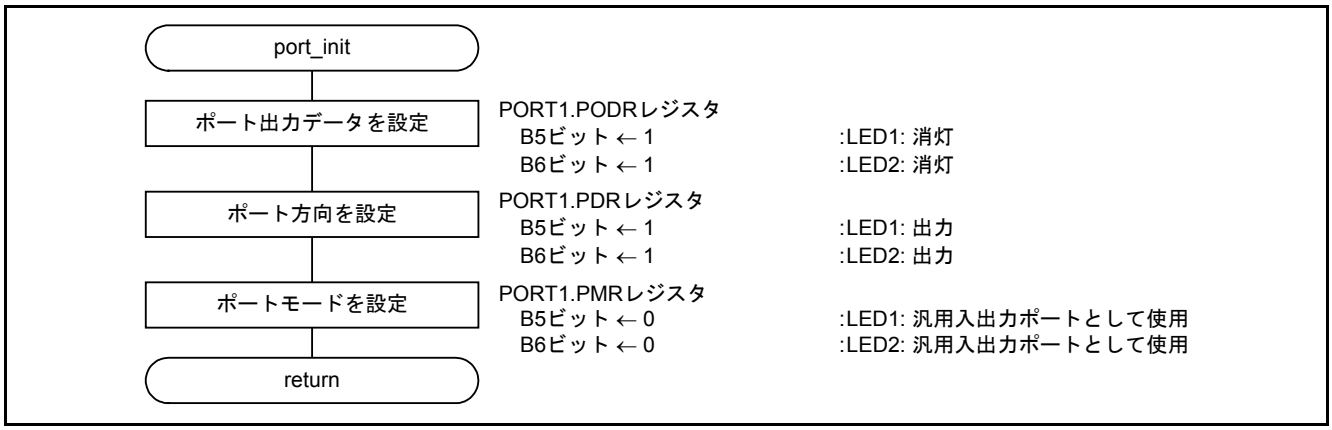


図7.5 ポート初期設定

7.9.3 周辺機能初期設定

図 7.6に周辺機能初期設定のフローチャートを示します。

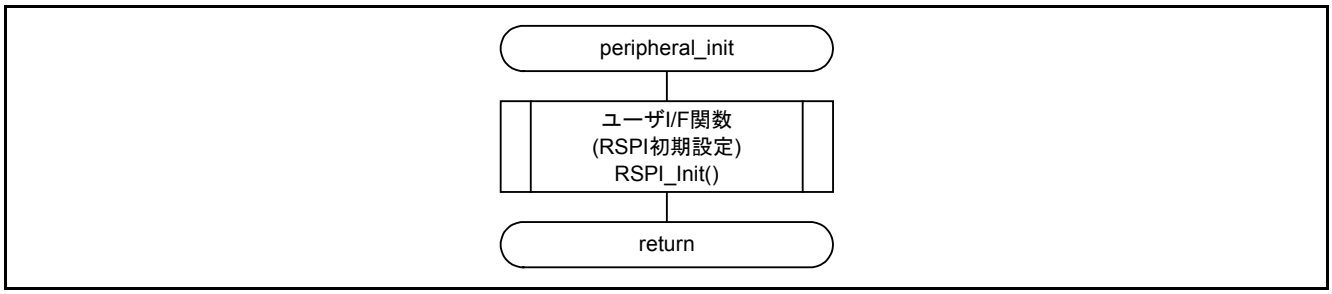


図7.6 周辺機能初期設定

7.9.4 コールバック関数(マスタとの RSPI 送受信完了)

図 7.7にコールバック関数(マスタとの RSPI 送受信完了)のフローチャートを示します。

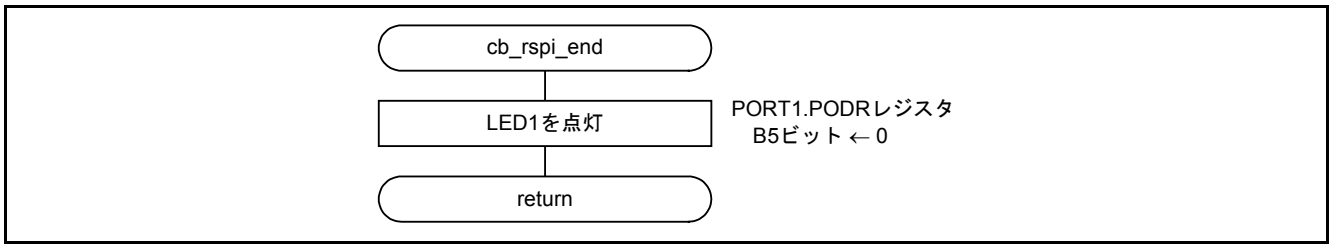
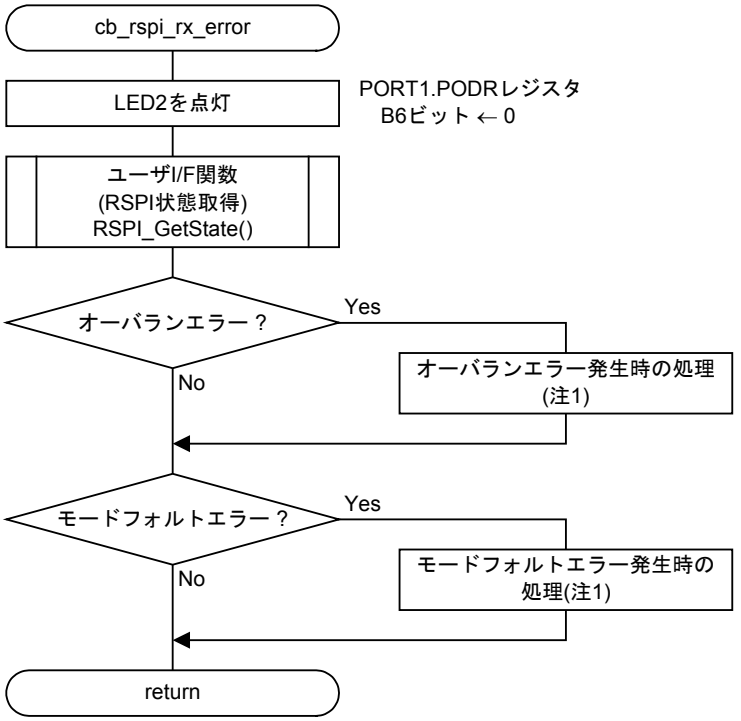


図7.7 コールバック関数(マスタとの RSPI 送受信完了)

7.9.5 コールバック関数(RSPI 送受信エラー)

図 7.8にコールバック関数(RSPI 送受信エラー)のフローチャートを示します。



注1. サンプルコードでは処理を行っていません。必要に応じてプログラムを追加してください。

図7.8 コールバック関数(RSPI 送受信エラー)

7.9.6　ユーザ I/F 関数(RSPI 初期設定)

図 7.9、図 7.10にユーザ I/F 関数(RSPI 初期設定)のフローチャートを示します。

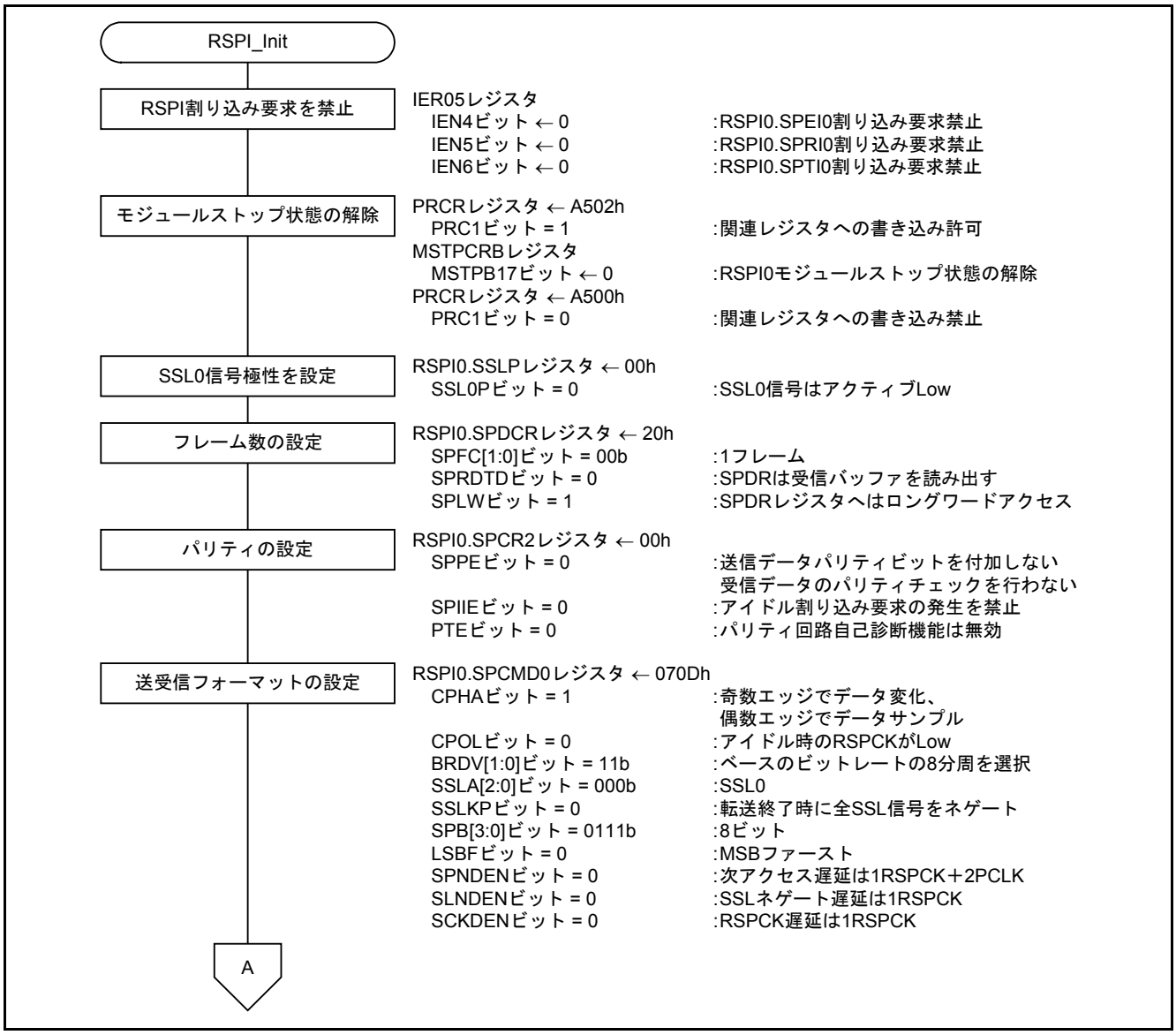


図7.9　ユーザ I/F 関数(RSPI 初期設定) (1/2)

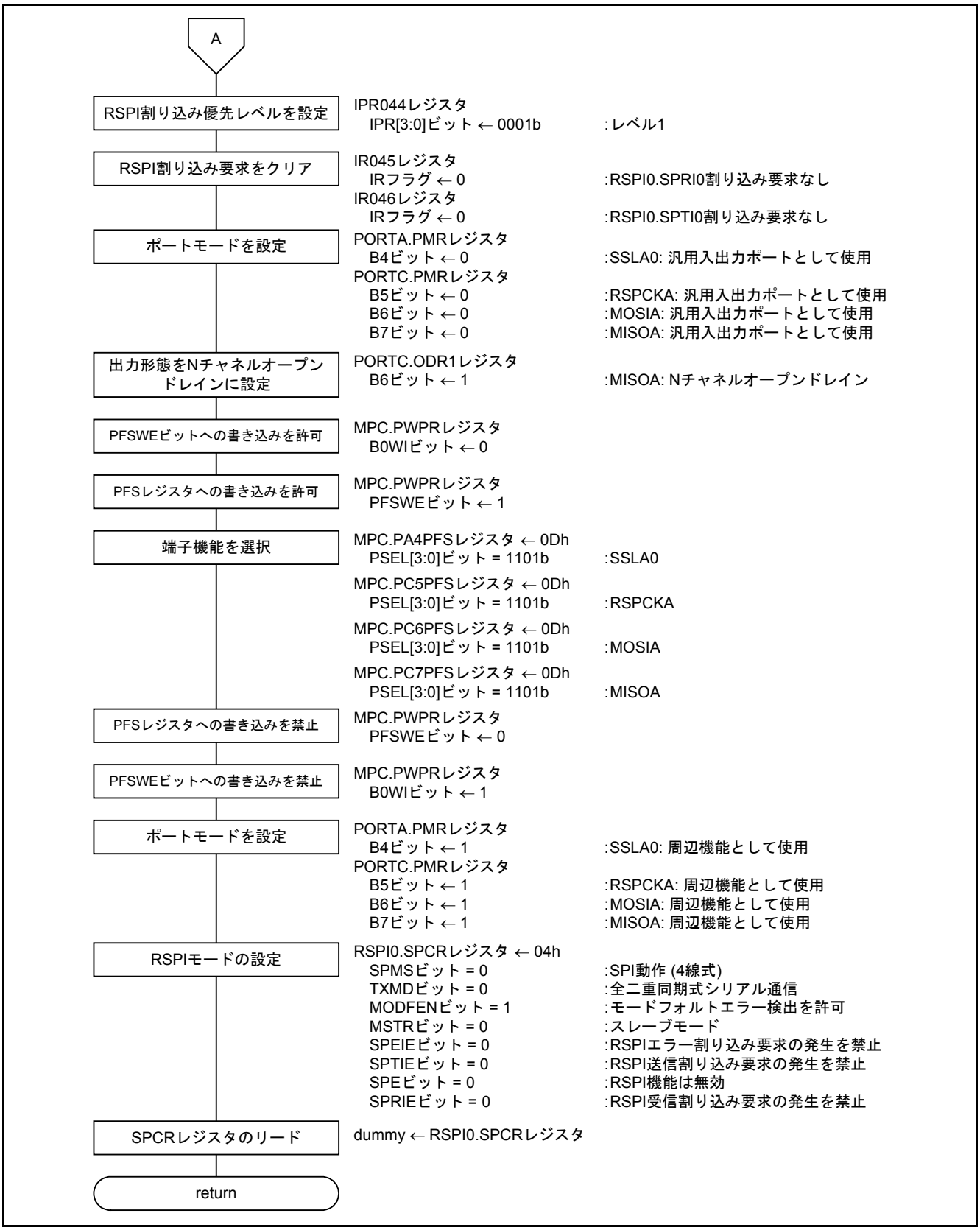


図7.10　ユーザ I/F 関数(RSPI 初期設定) (2/2)

7.9.7 ユーザ I/F 関数(RSPI 送受信開始)

図 7.11、図 7.12にユーザ I/F 関数(RSPI 送受信開始)のフローチャートを示します。

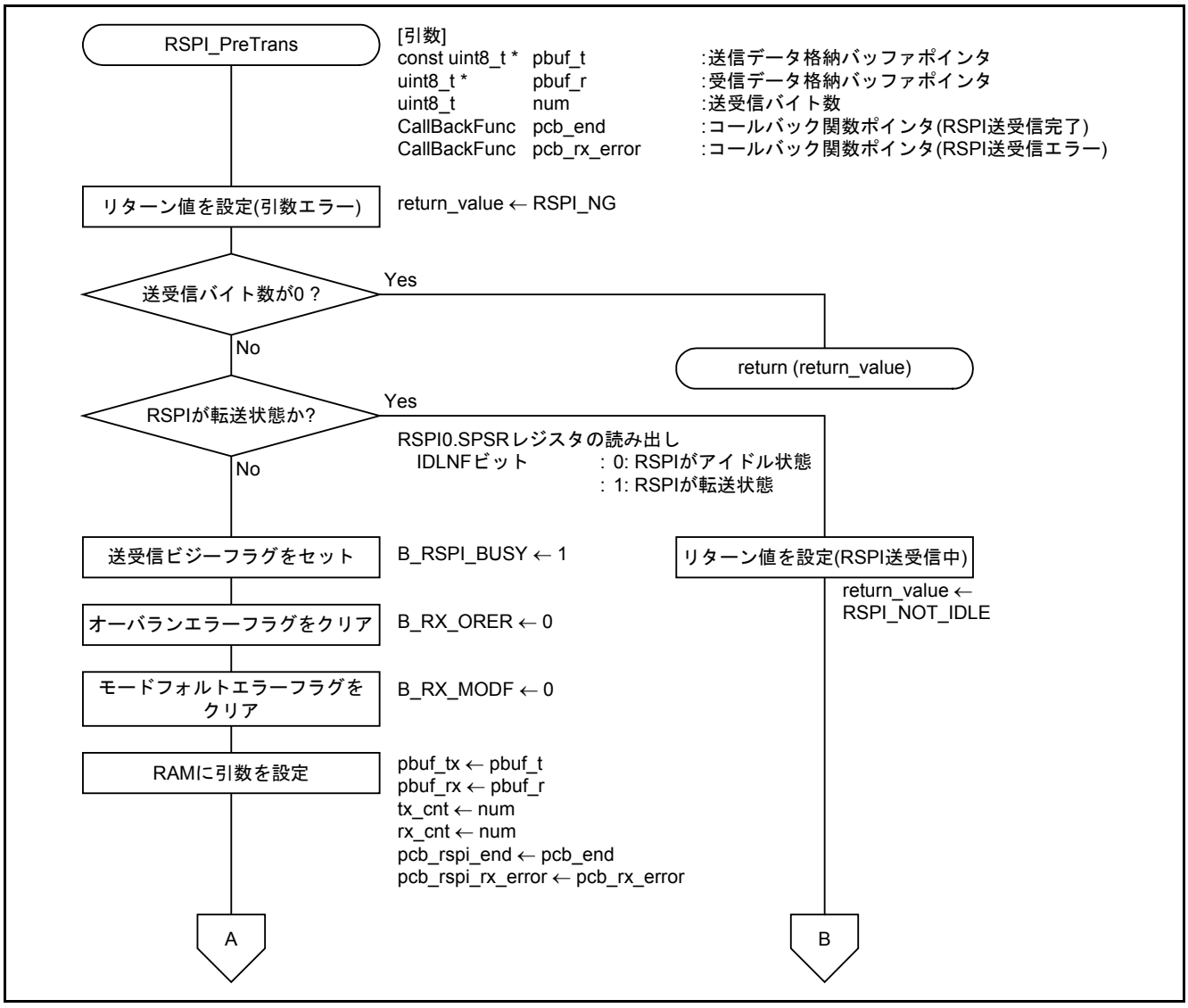
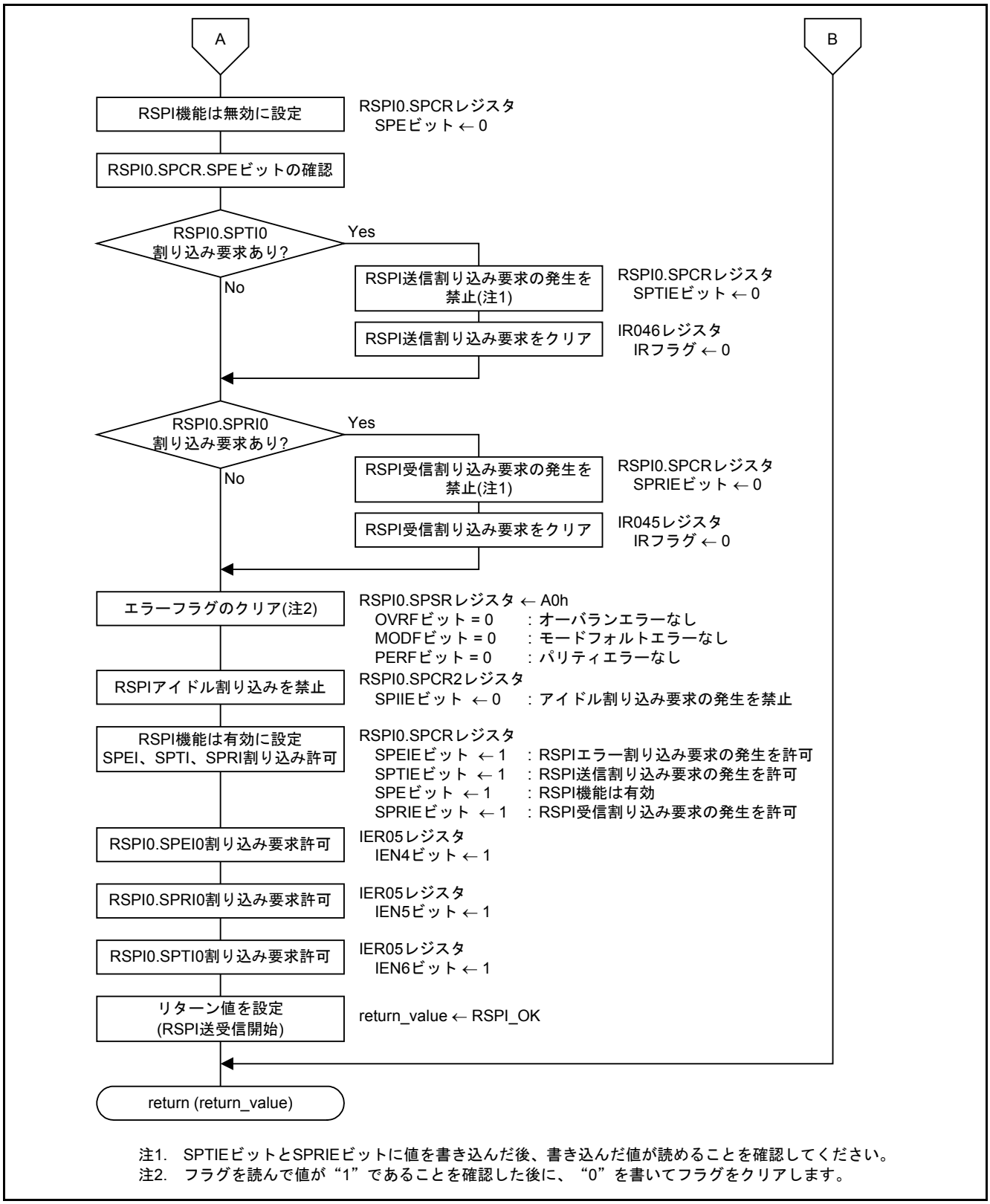


図7.11 ユーザ I/F 関数(RSPI 送受信開始) (1/2)



注1.　SPTIEビットとSPRIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。
注2.　フラグを読んで値が“1”であることを確認した後に、“0”を書いてフラグをクリアします。

図7.12　ユーザ I/F 関数(RSPI 送受信開始) (2/2)

7.9.8　ユーザ I/F 関数(RSPI 状態取得)

図 7.13にユーザ I/F 関数(RSPI 状態取得)のフローチャートを示します。

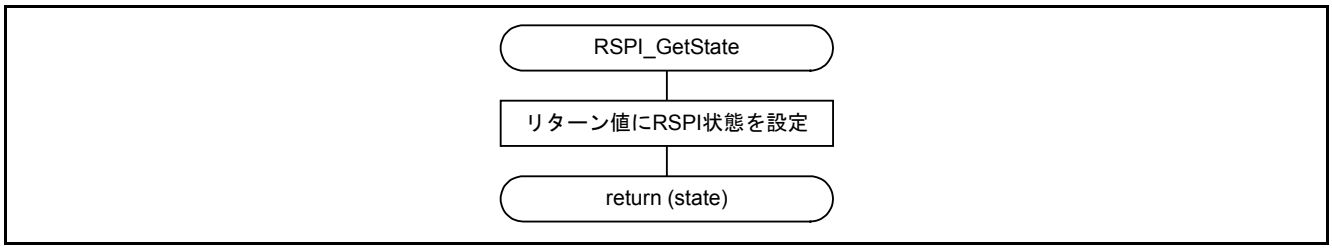
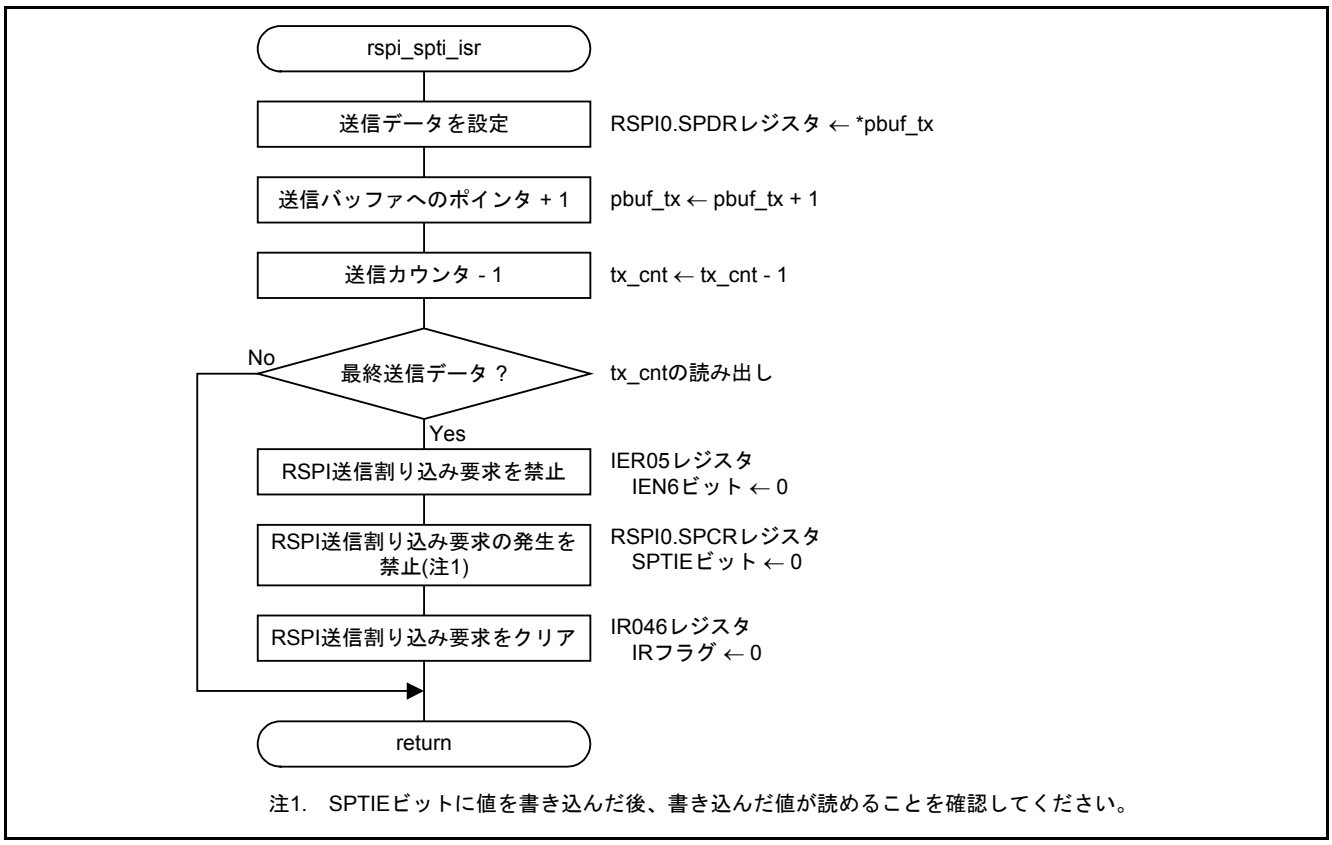


図7.13　ユーザ I/F 関数(RSPI 状態取得)

7.9.9　RSPI 送信割り込み

図 7.14にRSPI 送信割り込みのフローチャートを示します。

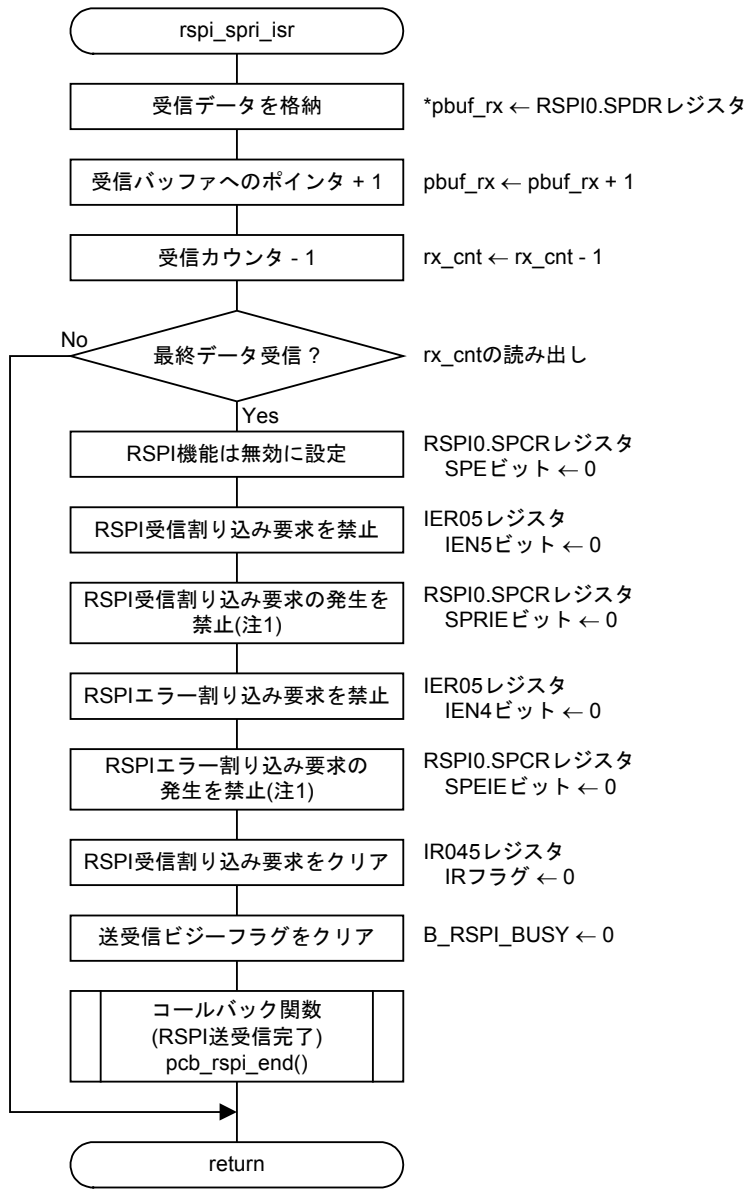


注1.　SPTIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。

図7.14　RSPI 送信割り込み

7.9.10 RSPI 受信割り込み

図 7.15にRSPI 受信割り込みのフローチャートを示します。

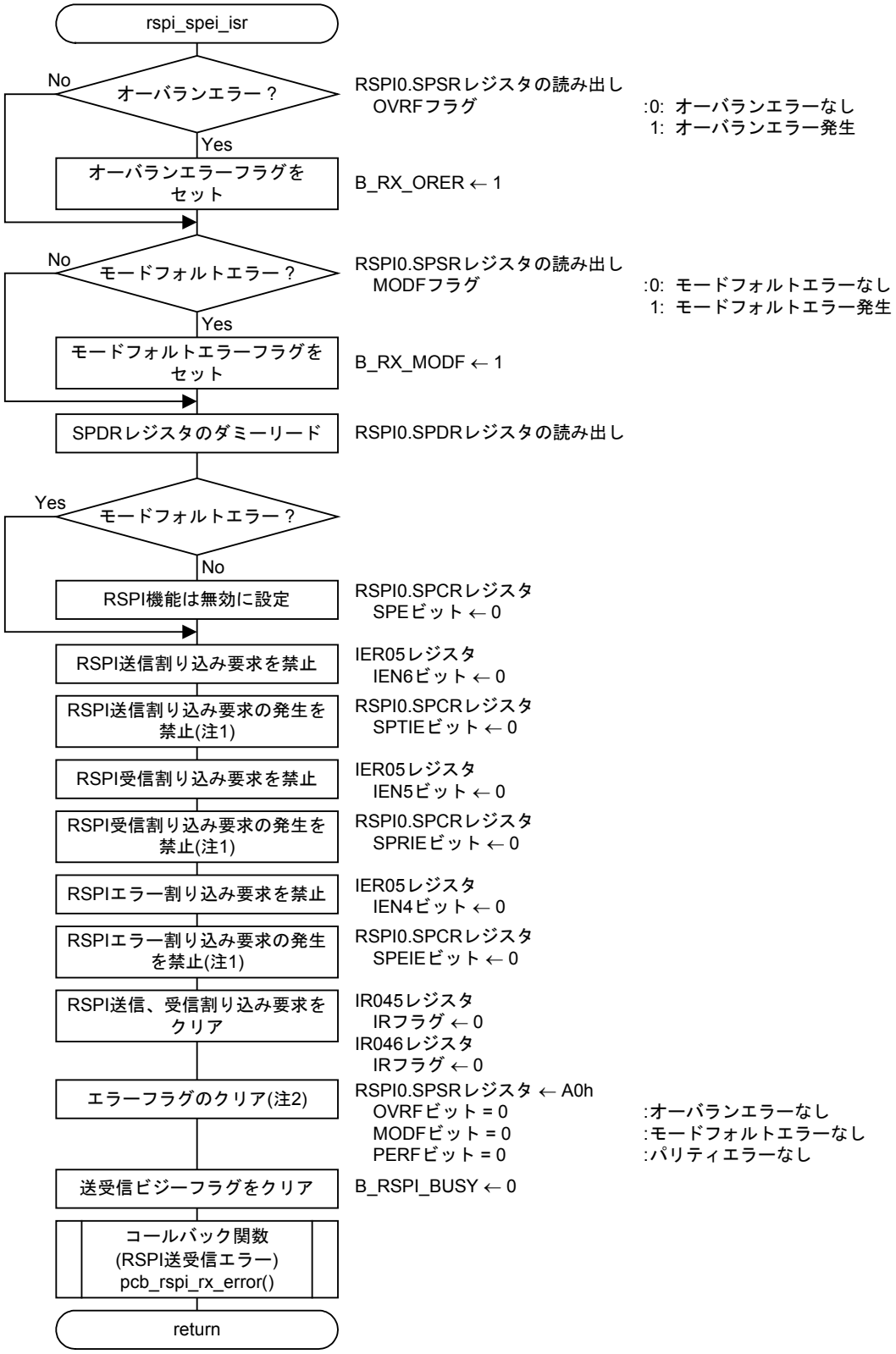


注1. SPRIEビット、SPEIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。

図7.15 RSPI 受信割り込み

7.9.11 RSPI エラー割り込み

図 7.16にRSPI エラー割り込みのフローチャートを示します。



注1. SPTIEビット、SPRIEビット、SPEIEビットに値を書き込んだ後、書き込んだ値が読めることを確認してください。
注2. フラグを読んで値が“1”であることを確認した後に、“0”を書いてフラグをクリアします。

図7.16 RSPI エラー割り込み

7.9.12 RSPI0.SPEI0 割り込み処理

図 7.17にRSPI0.SPEI0 割り込み処理のフローチャートを示します。

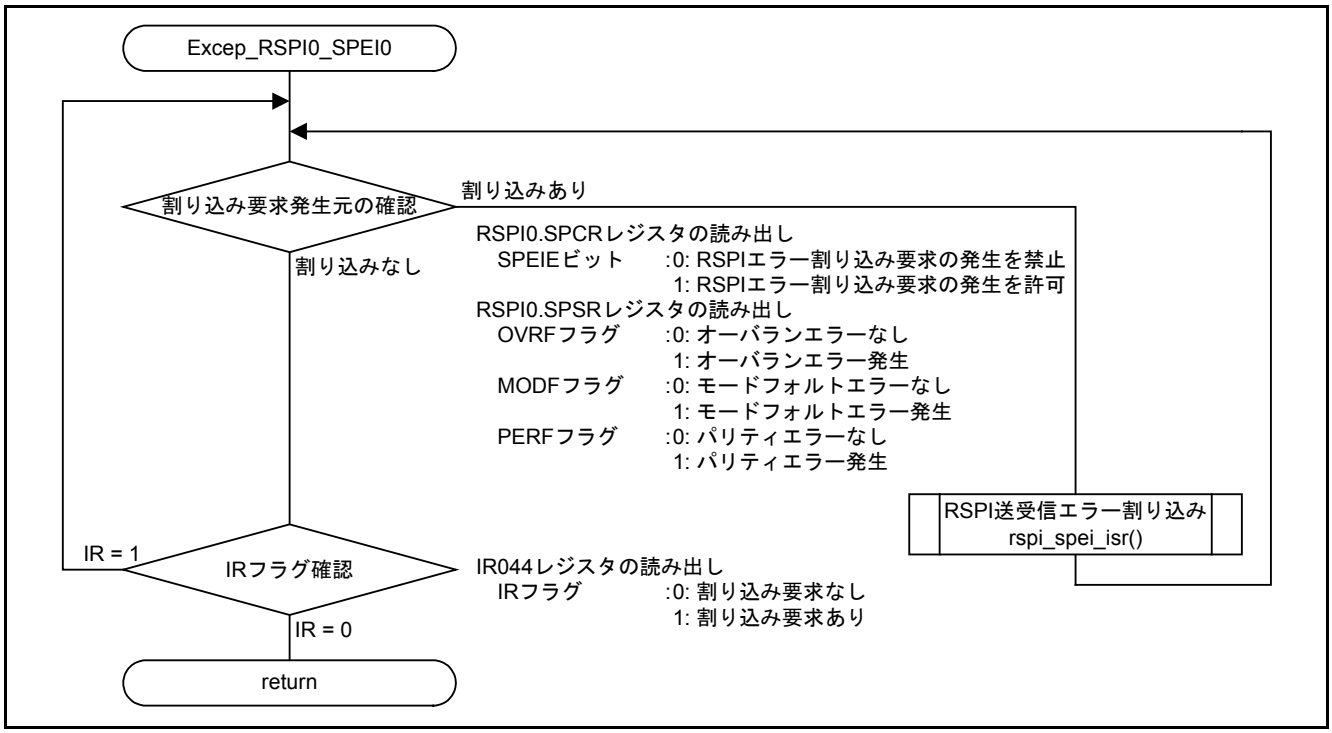


図7.17 RSPI0.SPEI0 割り込み処理

7.9.13 RSPI0.SPRI0 割り込み処理

図 7.18にRSPI0.SPRI0 割り込み処理のフローチャートを示します。

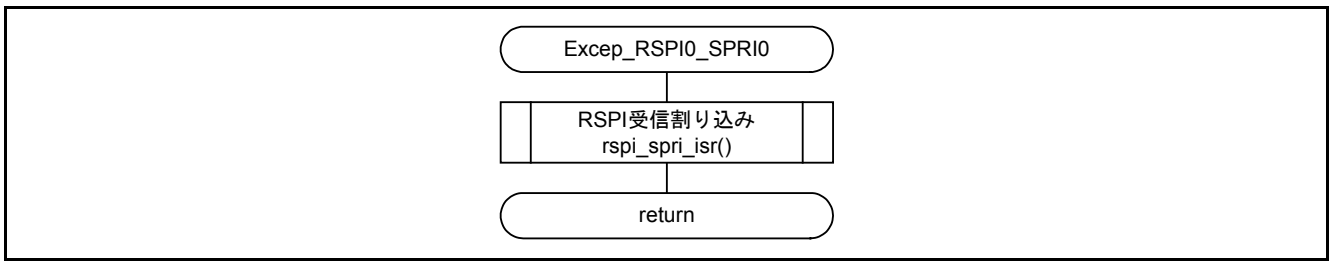


図7.18 RSPI0.SPRI0 割り込み処理

7.9.14 RSPI0.SPTI0 割り込み処理

図 7.19にRSPI0.SPTI0 割り込み処理のフローチャートを示します。

図7.19 RSPI0.SPTI0 割り込み処理

8. サンプルコード

サンプルコードは、ルネサス エレクトロニクスホームページから入手してください。

9. 参考ドキュメント

ユーザーズマニュアル：ハードウェア
RX220グループ ユーザーズマニュアル ハードウェア編 Rev.1.00 (R01UH0292JJ)
(最新版をルネサス エレクトロニクスホームページから入手してください。)

テクニカルアップデート／テクニカルニュース
(最新の情報をルネサス エレクトロニクスホームページから入手してください。)

ユーザーズマニュアル：開発環境
RX ファミリ C/C++コンパイラパッケージ V.1.01 ユーザーズマニュアル Rev.1.00 (R20UT0570JJ)
(最新版をルネサス エレクトロニクスホームページから入手してください。)

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/contact/

改訂記録	RX220グループ　アプリケーションノート RSPI を用いた通信設定例

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2013.10.01	—	初版発行

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. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して、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。
2. 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。
3. 本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害に関し、当社は、何らの責任を負うものではありません。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。
4. 当社製品を改造、改変、複製等しないでください。かかる改造、改変、複製等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、
   各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準： コンピュータ、OA機器、通信機器、計測機器、AV機器、
   家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通用信号機器、
   防災・防犯装置、各種安全装置等
   当社製品は、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（原子力制御システム、軍事機器等）に使用されることを意図しておらず、使用することはできません。たとえ、意図し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損害が生じても、当社は一切その責任を負いません。なお、ご不明点がある場合は、当社営業にお問い合わせください。
6. 当社製品をご使用の際は、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他の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。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は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、社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制するRoHS指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。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。
9. 本資料に記載されている当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。また、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に使用しないでください。当社製品または技術を輸出する場合は、「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。
10. お客様の転売等により、本ご注意書き記載の諸条件に抵触して当社製品が使用され、その使用から損害が生じた場合、当社は何らの責任も負わず、お客様にてご負担して頂きますのでご了承ください。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。

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）

■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/