

RX210 グループ

RSPI を用いた SPI 通信

要旨

本サンプルコードでは、SPI 通信の初期設定から送信、別チャンネルで送信データを受信する方法について説明します。

対象デバイス

- RX210

内容

1.	仕様	3
2.	動作確認条件	3
3.	ハードウェア説明.....	4
3.1	使用端子一覧.....	4
3.2	接続信号	5
4.	ソフトウェア説明.....	6
4.1	動作概要	6
4.2	ファイル構成.....	10
4.3	オプション設定メモリ.....	11
4.4	定数一覧	11
4.5	変数一覧	12
4.6	関数一覧	13
4.7	関数仕様	14
4.8	作成する関数のフローチャート.....	16
4.8.1	初期設定.....	16
4.8.2	メイン処理.....	17
4.8.3	RSPI0 転送完了処理	18
4.8.4	RSPI1 転送完了処理	19
4.8.5	RSPI0 エラー検出処理	21
4.8.6	RSPI1 エラー検出処理	エラー! ブックマークが定義されていません。
5.	PDG の設定	22
5.1	SYSTEM 設定	24
5.2	RSPI0 設定	25
5.3	RSPI1 設定	26
5.4	SYSTEM の端子設定	27
5.5	I/O 設定	28
5.6	ソースの生成.....	30
5.7	CS+への登録.....	31
6.	CS+のプロジェクトに PDG のソースファイルを登録する際の設定.....	33
7.	動作確認方法	36
7.1	ウォッチ式の登録.....	36
7.2	実行	38
8.	参考ドキュメント.....	39

1. 仕様

RSPI0（マスタモード）から SCI5（簡易 SPI スレーブモード）へ SPI（4 線式）通信でデータを送信します。SCI5 は受信したデータを RSPI0 に送り返します。RSPI0 が SCI5 へ送信したデータと SCI5 が RSPI0 へ送り返したデータが一致した場合、LED(D3)が点灯します。

2. 動作確認条件

本サンプルコードは、表 2.1 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	R5F5210BBDFP（RX210 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）
ボード電源電圧	5V
マイコン動作電圧	5V
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
統合開発環境	ルネサスエレクトロニクス製品 CS+ for CC-RL V5.00.00
エミュレータ	ルネサスエレクトロニクス製 E1 エミュレータ
使用ボード	北斗電子製評価ボード HSBRX210-100B (R5F5210BBDFP)

3. ハードウェア説明

3.1 使用端子一覧

表 3.1 に使用端子と機能を示します。

表 3.1 使用端子と機能

端子名	入出力	内容
PA7	入力	MISOA (マスタデータ入力)
PA6	出力	MOSIA (マスタデータ出力)
PA5	出力	RSPCKA (マスタクロック出力)
PA4	出力	SSLA0 (マスタチップセレクト出力)
PC4	入力	SCK5 (スレーブクロック入力)
PC3	入力	SMOSI5 (スレーブデータ入力)
PC2	出力	SMISO5 (スレーブデータ出力)
PC0	入力	SS5# (スレーブチップセレクト入力)
PH2	出力	LED (D3)

3.2 接続信号

本サンプルコードでは、使用ボードで表 3.2 に示す接続を追加してください。

表 3.2 接続信号一覧

	RSPIO	SCI5
接続 1	J2 の 1pin (PA7/MISOA)	J1 の 11pin (PC2/SMIS05)
接続 2	J2 の 2pin (PA6/MOSIA)	J1 の 12pin (PC3/SMOSI5)
接続 3	J2 の 3pin (PA5/RSPCKA)	J1 の 13pin (PC4/SCK5)
接続 4	J2 の 4pin (PA4/SSLA0)	J1 の 9pin (PC0/SS5#)

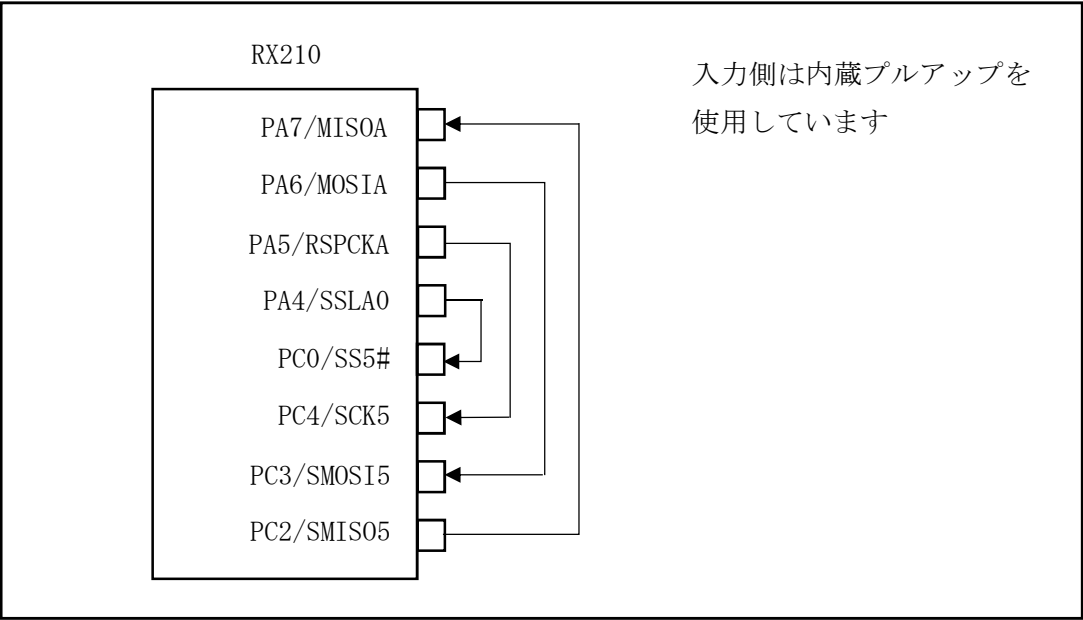


図 3.1 信号接続図

4. ソフトウェア説明

4.1 動作概要

図 4.1 に RSPIO → SCI5 の動作タイミング図を、以下に図中の番号の動作および処理を示します。使用関数も合わせて参照ください。

※RSPIO、SCI5 共に送信および受信機能を選択していますので、送信と受信を同時に実行します。

(1) 初期設定

RSPIO から SCI5 へデータを送信するために、以下の初期設定を行います。

●RSPIO

- ・SPI 動作（4 線式）マスタモード
- ・送信および受信（全二重同期式シリアル通信）
- ・バッファへのアクセスサイズはロングワード
- ・パリティビットなし
- ・ベースビットレートは 100kbps
- ・MOSI アイドル値を” L” 固定
- ・SSL 端子制御は SSL0 のみ使用し、アクティブ値は” L”
- ・コマンド数が 1 で転送フレーム数が 4

[コマンド]

- ・データ長は 8 ビット
- ・MSB ファースト
- ・奇数エッジでサンプル、偶数エッジでデータ変化
- ・RSPCK はアイドル時” L”
- ・SSL アサート信号は SSL0
- ・転送終了時に全 SSL 信号をネゲート
- ・SSL アサート開始から RSPCK 発振までの期間を 1RSPCK
- ・最終 RSPCK エッジ送出から SSL 信号ネゲートまでの期間を 1RSPCK
- ・転送終了後の SSL 信号の非アクティブ期間を 1RSPCK+2PCLK

●SCI5

- ・簡易 SPI モード
- ・送信および受信
- ・スレーブモード
- ・MSB ファースト
- ・転送クロックは外部クロック
- ・クロック遅れあり

(2) SCI5（スレーブ）の送受信開始

送受信の開始設定をします。このとき、送信データはダミーデータ（00h、00h、00h、00h）を使用しています。

(3) RSPI0（マスタ）の送受信開始

SSLA0 端子から” L” を出力し、RSPCKA 端子のクロックに同期させ、データ（11h、22h、33h、44h）を送信します。このとき、受信データはダミーバッファへ格納しています。

※SCI5（スレーブ）は SS5#端子に” L” が入力されることで SCK5 端子に入力されるクロックに同期し、送受信します。

(4) RSPI0 の送受信完了および SCI5 の送受信完了

1 コマンド 4 フレームの送受信が完了すると、転送完了割り込みが発生します。

(5) 1 コマンド 4 フレームの終了

SSLA0 端子から” H” を出力します。

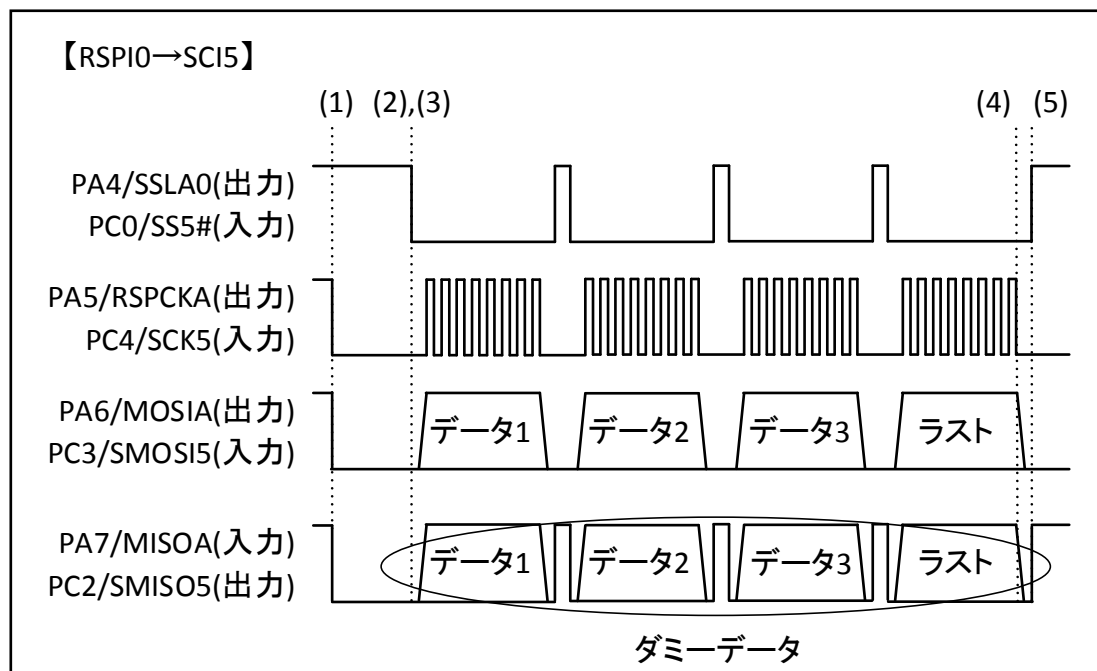


図 4.1 RSPIO → SCI5 の動作タイミング

図 4.2 に SCI5 → RSPiO の動作タイミング図を、以下に図中の番号の動作および処理を示します。使用関数も合わせて参照ください。

- (1) SCI5（スレーブ）の送受信開始
送受信の開始設定をします。このとき、送信データは RSPiO → SCI5 で受信したデータ（11h、22h、33h、44h）を使用します。
- (2) RSPiO（マスタ）の送受信開始
SSLA0 端子から” L” を出力し、RSPCKA 端子のクロックに同期させ、ダミーデータ（00h、00h、00h、00h）を送信します。
※SCI5（スレーブ）は SS5#端子に” L” が入力されることで SCK5 端子に入力されるクロックに同期し、送受信します。
- (3) RSPiO の送受信完了および SCI5 の送受信完了
1 コマンド 4 フレームの送受信が完了すると、転送完了割り込みが発生します。
※RSPiO で SCI5 へ送信したデータと同じデータ（11h、22h、33h、44h）が受信できれば、送受信が正しくできていると判断し、LED(D3)が点灯します。
- (4) 1 コマンド 4 フレームの終了
SSLA0 端子から” H” を出力します。

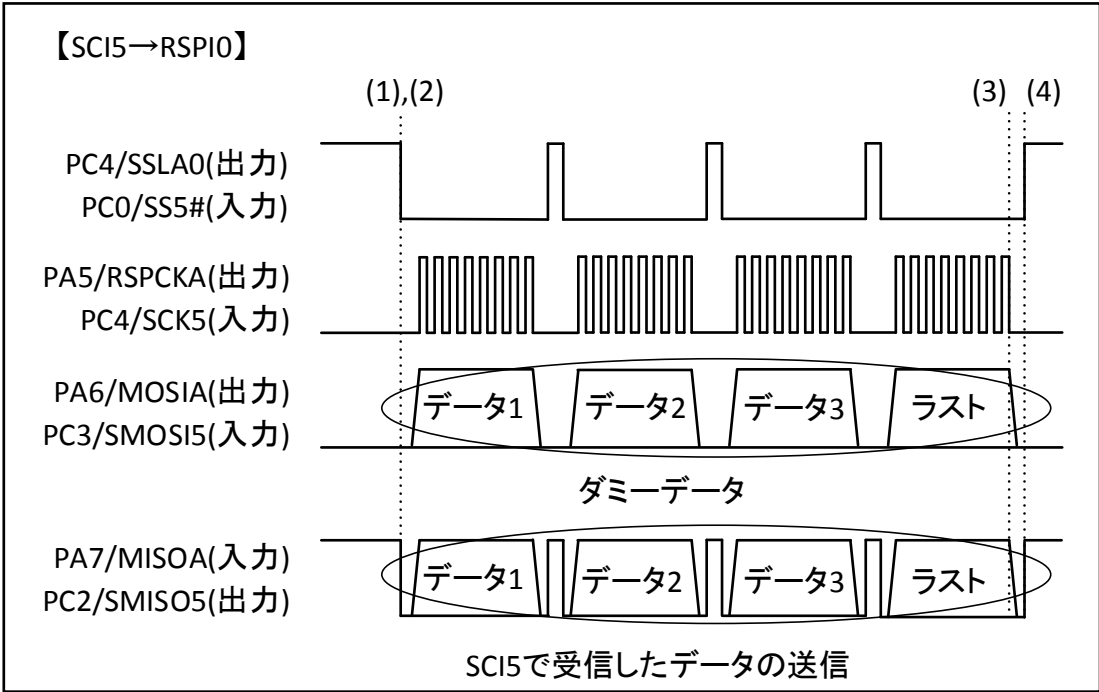


図 4.2 SCI5 → RSPiO の動作タイミング

4.2 ファイル構成

本サンプルコードを作成するにあたり、編集したファイルを表 4.1 に示します。統合開発環境で自動生成されて編集していないファイル、および 5. PDG の設定で生成されるファイルに関しましては割愛します。

表 4.1 ファイル名一覧

ファイル名	概要	備考
RSPI_RX210.c	メインファイル <ul style="list-style-type: none"> • SCI5 送信/受信処理 • RSPi0 送信/受信処理 • LED(D3)制御 • オプション設定メモリ 	
hwsetup.c	初期設定 <ul style="list-style-type: none"> • 存在しない端子の処理 • クロックの設定 • ポートの設定 • RSPi0 の設定 • SCI5 の設定 	
resetprg.c	リセット例外処理	HardwareSetup(); のコメントアウトを解除しました

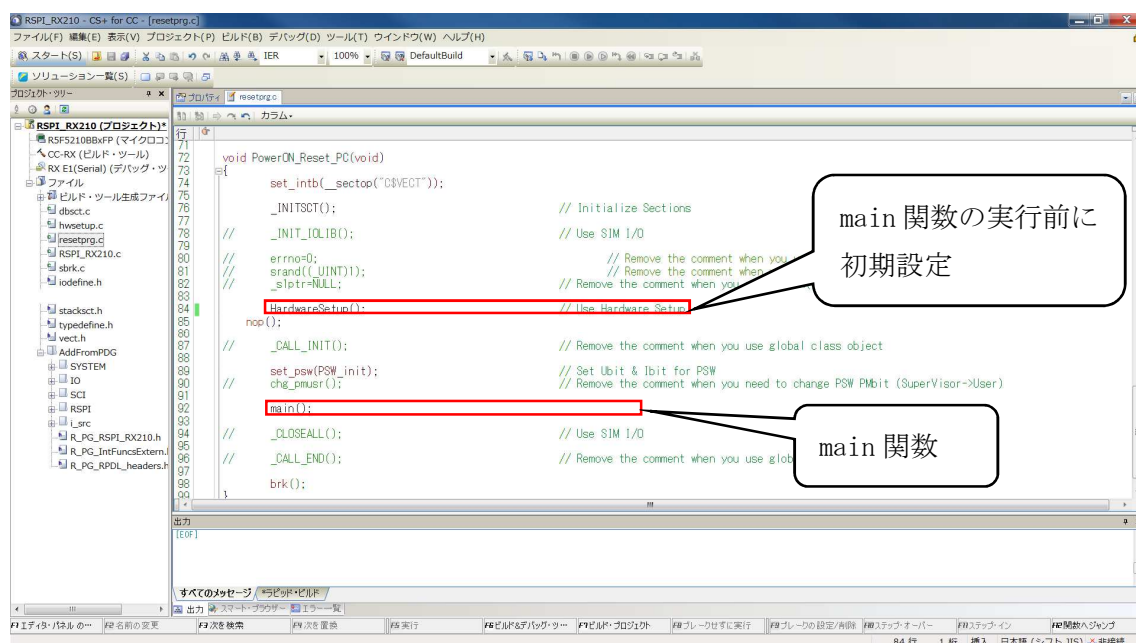


図 4.3 resetprg.c

4.3 オプション設定メモリ

表 4.2 に本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。

表 4.2 オプション設定メモリー一覧

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 リセット後、WDT 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 H0C0(高速オンチップオシレー タ)発振が無効
MDES	FFFF FF83h～FFFF FF80h	FFFF FFFFh	リトルエンディアン

OFS0 と OFS1 はメインファイルの最後尾に記載しています。

MDES については vecttbl.c ファイル(プロジェクト作成時に自動生成されるファイル)に定義されています。

4.4 定数一覧

表 4.3 に本サンプルコードで使用する定数、表 4.4 に const 型定数を示します。

表 4.3 サンプルコードで使用する定数

定数名	設定値	内容
BUFF_SIZE	4	通信バッファサイズ

表 4.4 サンプルコードで使用する const 型定数

型	変数名	内容	使用関数
const unsigned long	send_buf[BUFF_SIZE]	RSPI0 の送信データ	main Spi0IntFunc

4.5 変数一覧

表 4.5 に本サンプルコードで使用する変数を示します。

表 4.5 サンプルコードで使用する変数

型	変数名	内容	使用関数
unsigned long	CH0_read_buf[BUFF_SIZE]	RSPI0 の受信バッファ	Spi0IntFunc Sci5ReFunc
unsigned long	dummy_rcv[BUFF_SIZE]	ダミー用受信バッファ	main
unsigned long	dummy_trs[BUFF_SIZE]	ダミー用送信バッファ	Sci5ReFunc
unsigned char	SCI5_read_buf[BUFF_SIZE]	SCI5 の受信バッファ	main Sci5ReFunc
unsigned char	SCI5_dummy_rcv[BUFF_SIZE]	ダミー用受信バッファ	main Sci5ReFunc
unsigned char	SCI5_dummy_trs[BUFF_SIZE]	ダミー用送信バッファ	main
bool	rspi0_receive_flag	RSPI0 用受信フラグ (受信データ使用時 true)	main Spi0IntFunc Sci5ReFunc
bool	sci5_receive_flag	SCI5 用受信フラグ (受信時 true)	main Sci5ReFunc
bool	over_run0	RSPI0 用オーバーランエ ラーフラグ格納	Spi0ErIntFunc
bool	mode_fault0	RSPI0 用モードフォー ルトエラーフラグ格納	Spi0ErIntFunc
bool	parity_error0	RSPI0 用パリティエラ ーフラグ格納	Spi0ErIntFunc
bool	over_run_sci5	SCI5 用オーバーランエ ラーフラグ格納	Sci5ErFunc

4.6 関数一覧

表 4.6 に関数一覧を掲載します。本サンプルコードで新規作成、もしくは編集した関数のみ記載しています。PDG の設定は 5. PDG の設定を参照ください。サンプルコードで使用している PDG で生成された関数に関しましては、「RX210 グループ Peripheral Driver Generator リファレンスマニュアル」を参照ください。

表 4.6 関数一覧

関数名	概要
main	メイン処理
Spi0IntFunc	RSPI0 転送完了関数
Sci5TrFunc	SCI5 送信完了関数
Sci5ReFunc	SCI5 受信完了関数
Sci5ErFunc	SCI5 エラー検出関数

4.7 関数仕様

本サンプルコードで作成、もしくは編集した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	SCI5 の送受信開始 RSPI0 の送受信開始
引数	なし
リターン値	なし
Spi0IntFunc	
概要	RSPI0 転送完了処理
ヘッダ	なし
宣言	void Spi0IntFunc(void)
説明	エラー確認 ・受信データを使用時 送信データと受信データの比較 LED(D3)制御 ・その他 処理なし
引数	なし
リターン値	なし
Sci5TrFunc	
概要	SCI5 送信完了処理
ヘッダ	なし
宣言	void Sci5TrFunc(void)
説明	・送信時 受信時に設定 SCI5 の受信データを送信データとして送受信開始 RSPI0 の送受信開始 ・その他 処理なし
引数	なし

リターン値 なし

Sci5ReFunc

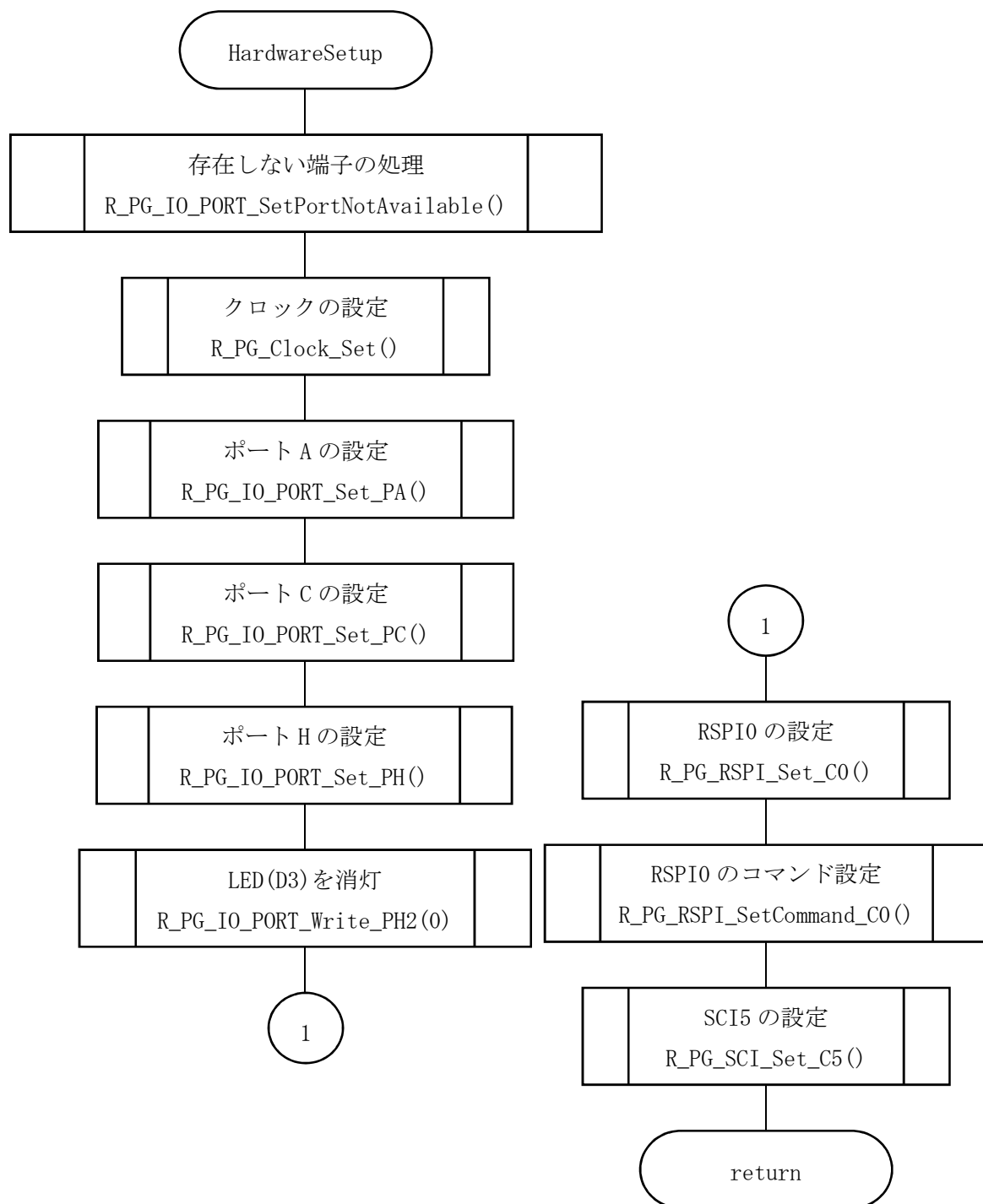
概要	SCI5 受信完了処理
ヘッダ	なし
宣言	void Sci5ReFunc(void)
説明	<ul style="list-style-type: none">・受信時送信時に設定・その他SCI5 の送受信を停止
引数	なし
リターン値	なし

Sci5ErFunc

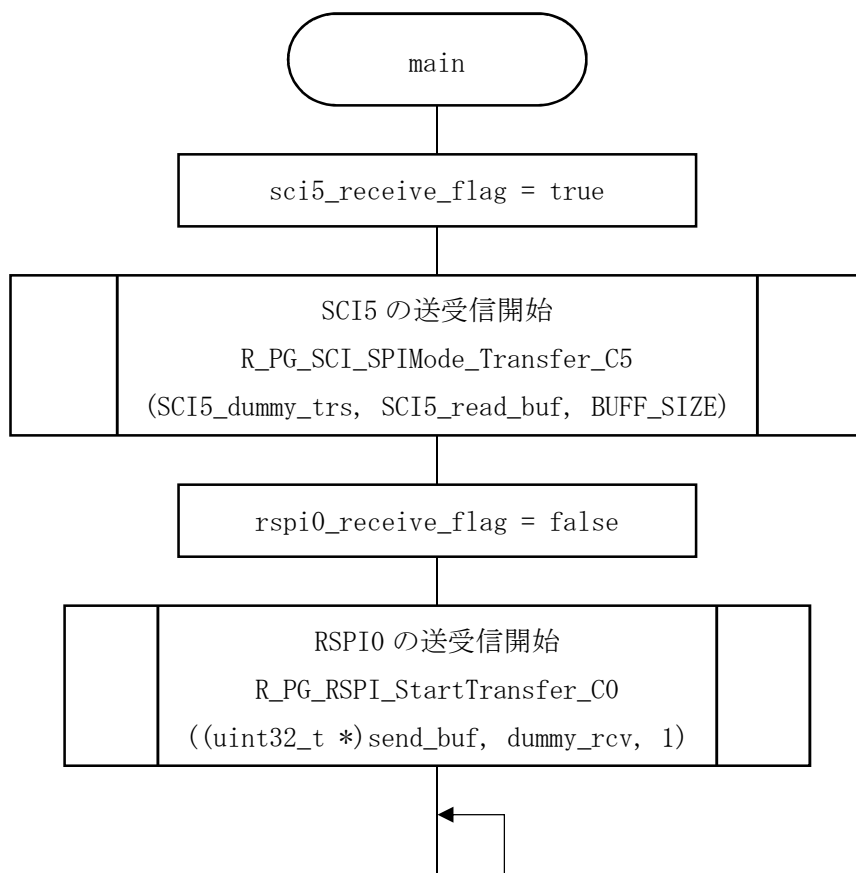
概要	SCI5 エラー検出処理
ヘッダ	なし
宣言	void Sci5ErFunc (void)
説明	エラーフラグの取得
引数	なし
リターン値	なし

4.8 作成する関数のフローチャート

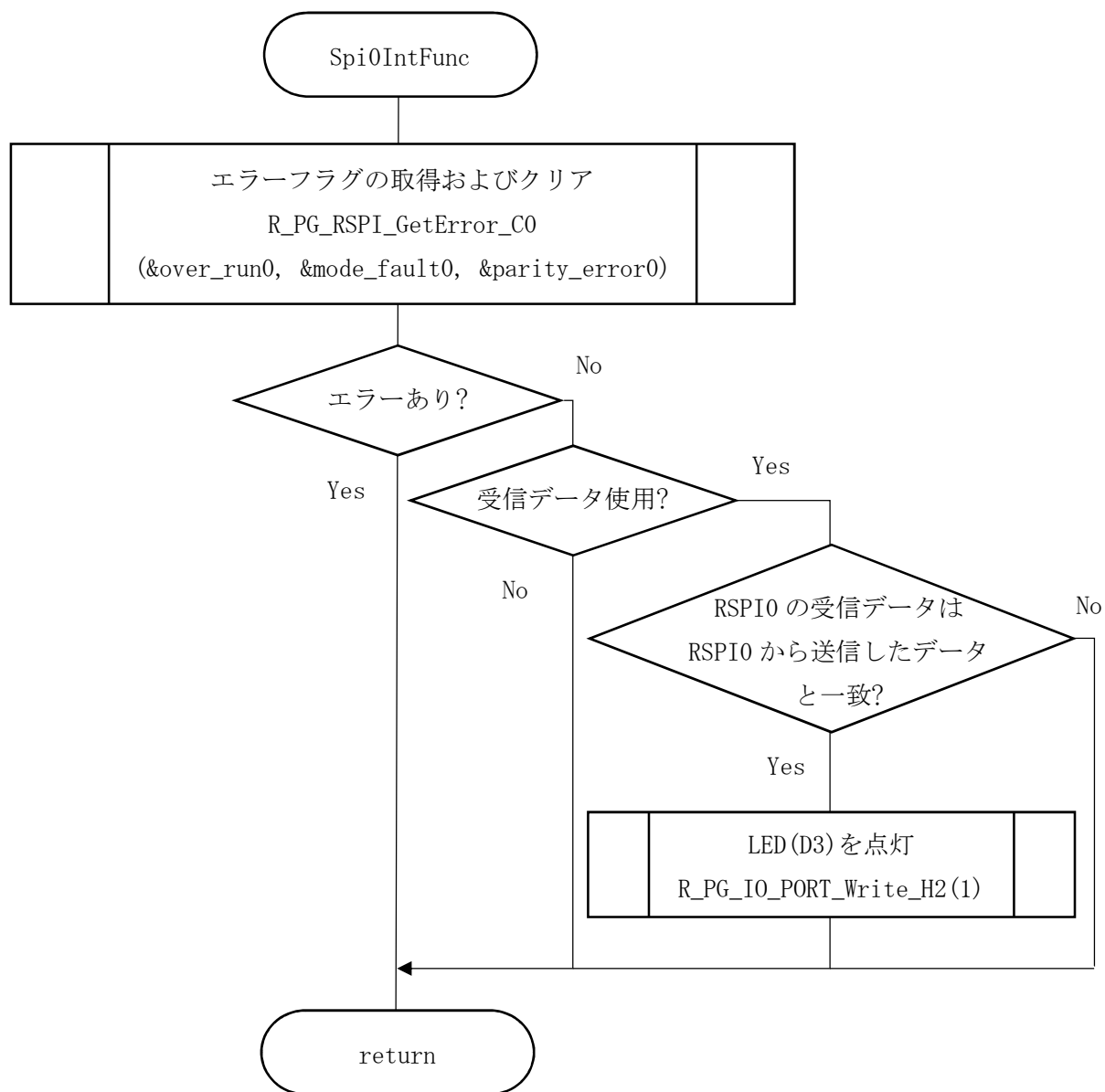
4.8.1 初期設定



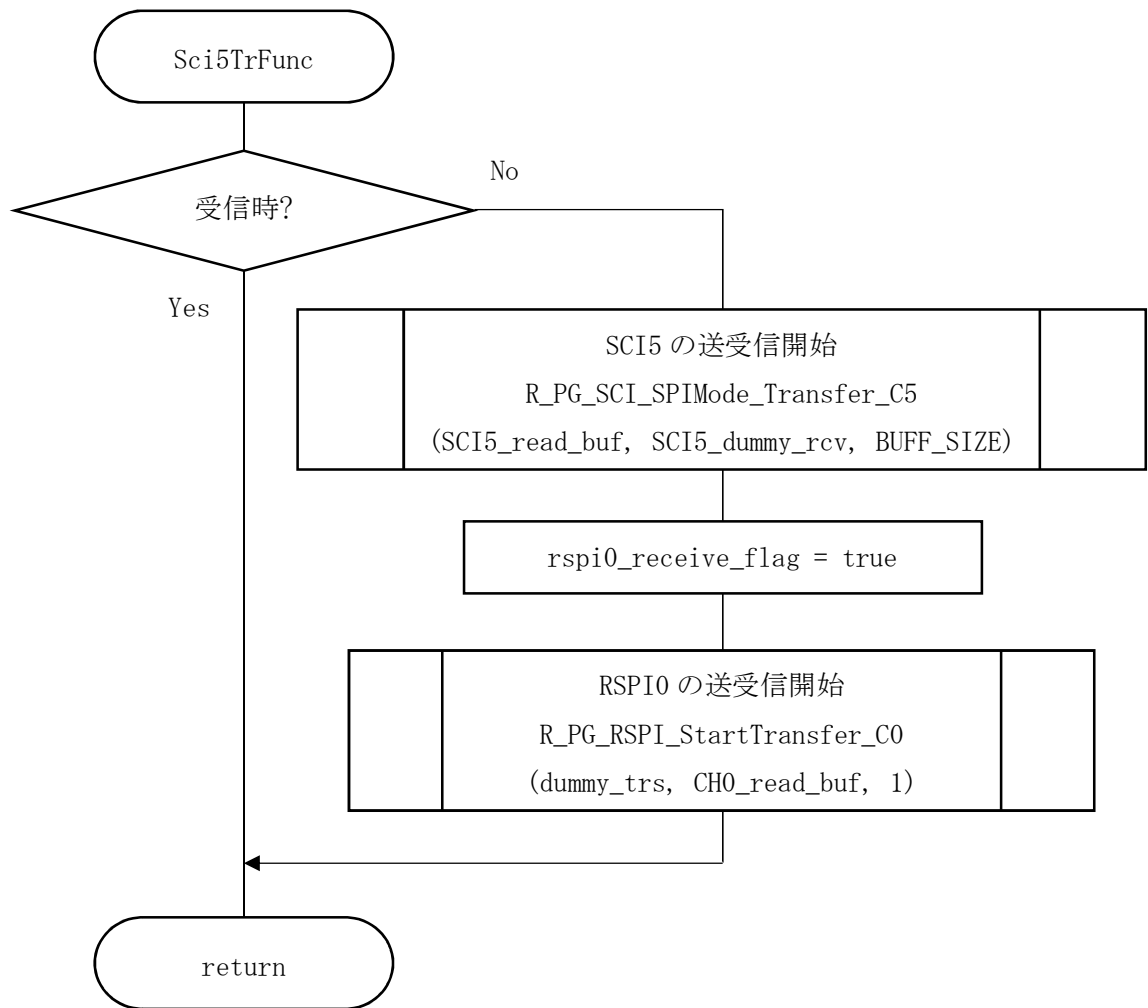
4.8.2 メイン処理



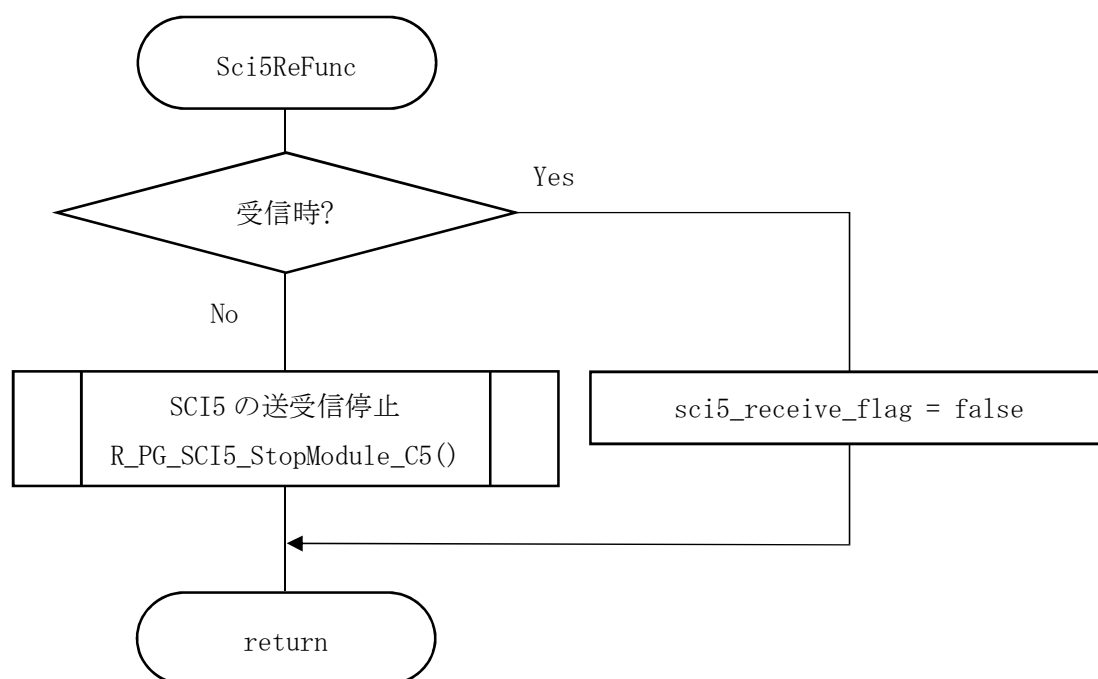
4.8.3 RSPI0 転送完了処理



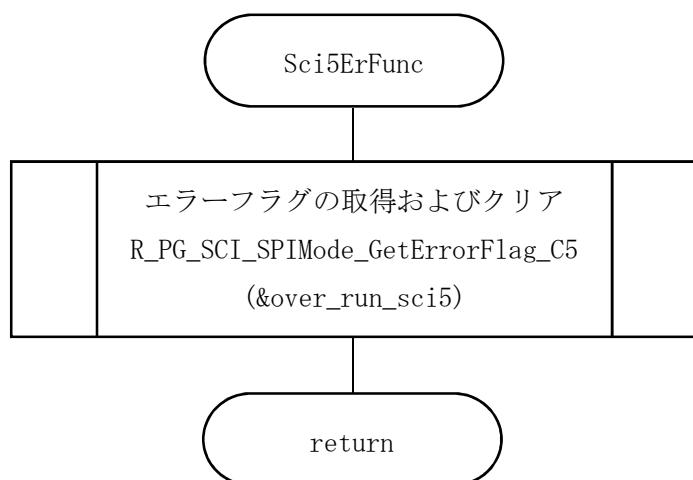
4.8.4 SCI5 送信完了処理



4.8.5 SCI5 受信完了処理



4.8.6 SCI5 エラー検出処理




5. PDG の設定

本サンプルコードにおける PDG の設定を以下に説明します。本設定において生成されるソースファイルの詳細は”RX210 グループ Peripheral Driver Generator リファレンスマニュアル”を参照ください。

Peripheral Driver Generator 2 を起動します。




メニューバーのファイル→プロジェクトの新規作成 をクリックすると、以下のウィンドウが表示されます。プロジェクト名、マイコンのグループ、型を入力し、「OK」をクリックすると、プロジェクトが作成されます。



新規作成

プロジェクト名:
RSPI_RX210

ディレクト 
c:\renesas\PDG2_proj 参照...

デバイス選択

シリーズ: RX200

グループ: RX210

型: R5F5210BBxFP

パッケージ: PLQP0100KB-A

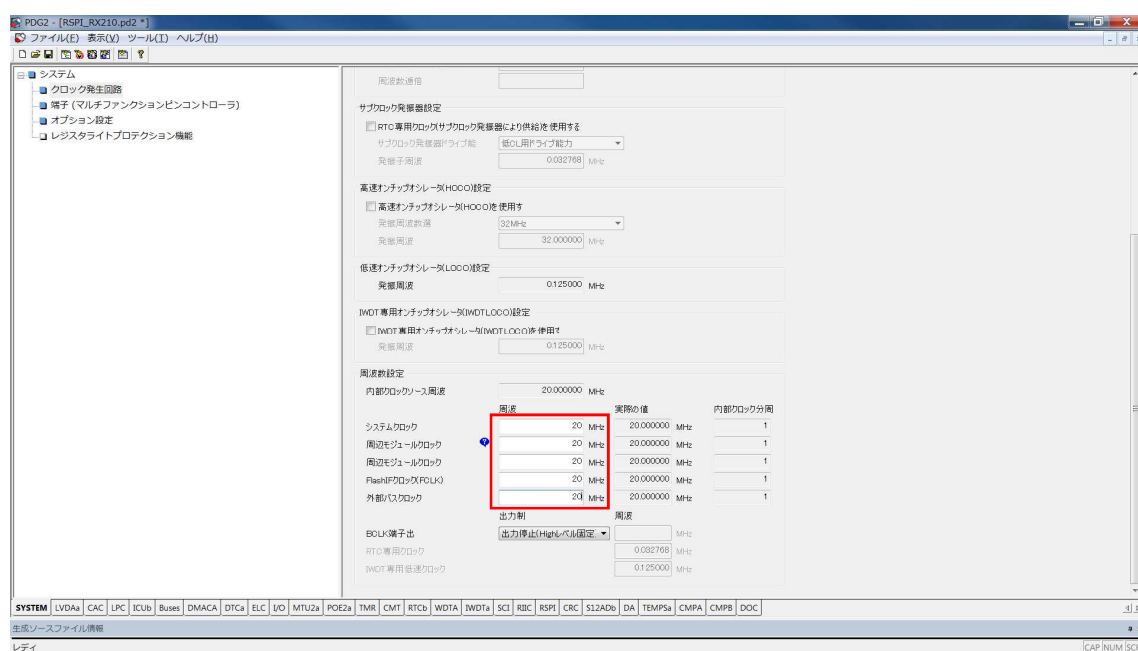
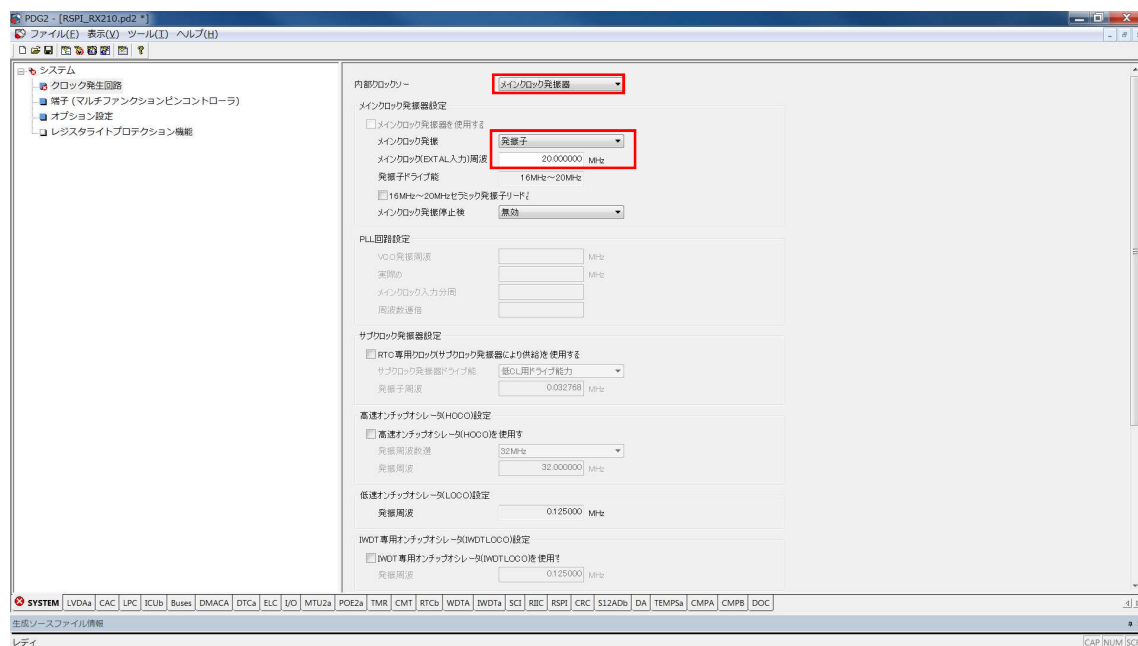
ROM容 1M バイト

RAM容 96K バイト

クリック OK キャンセル

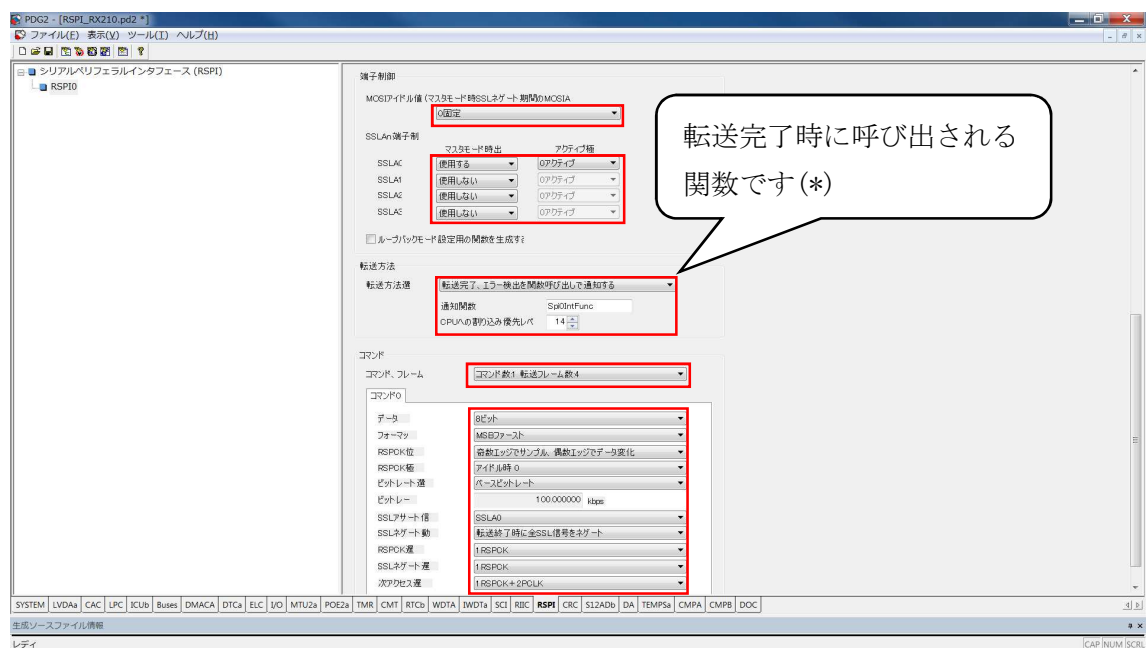
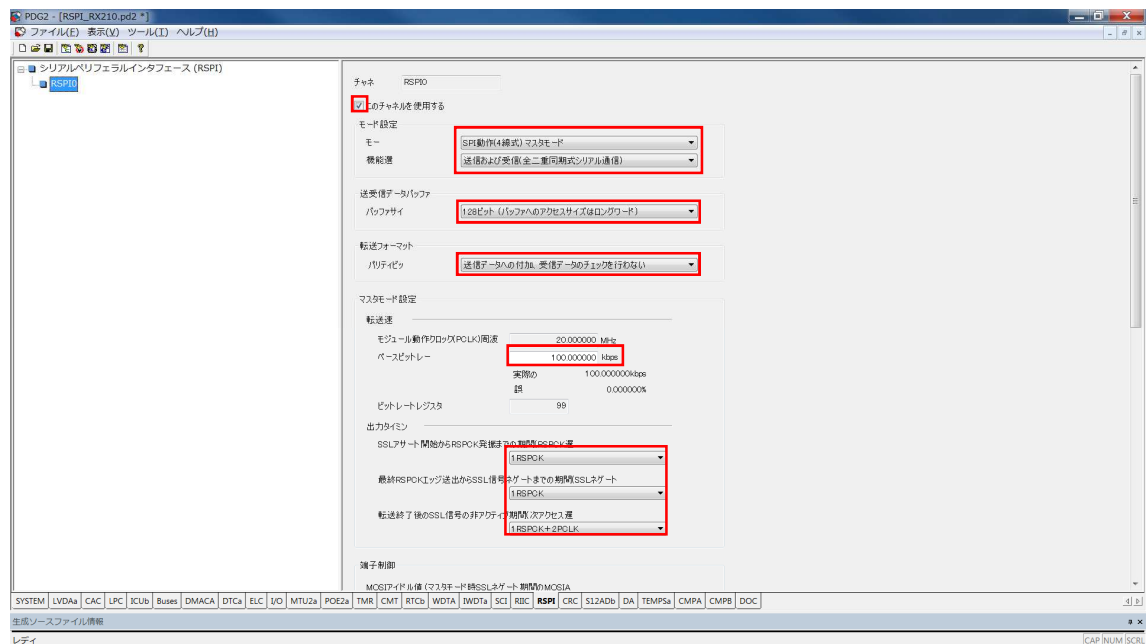
5.1 SYSTEM 設定

システムタブのクロック発生回路の設定を以下に示します。



5.2 RSP10 設定

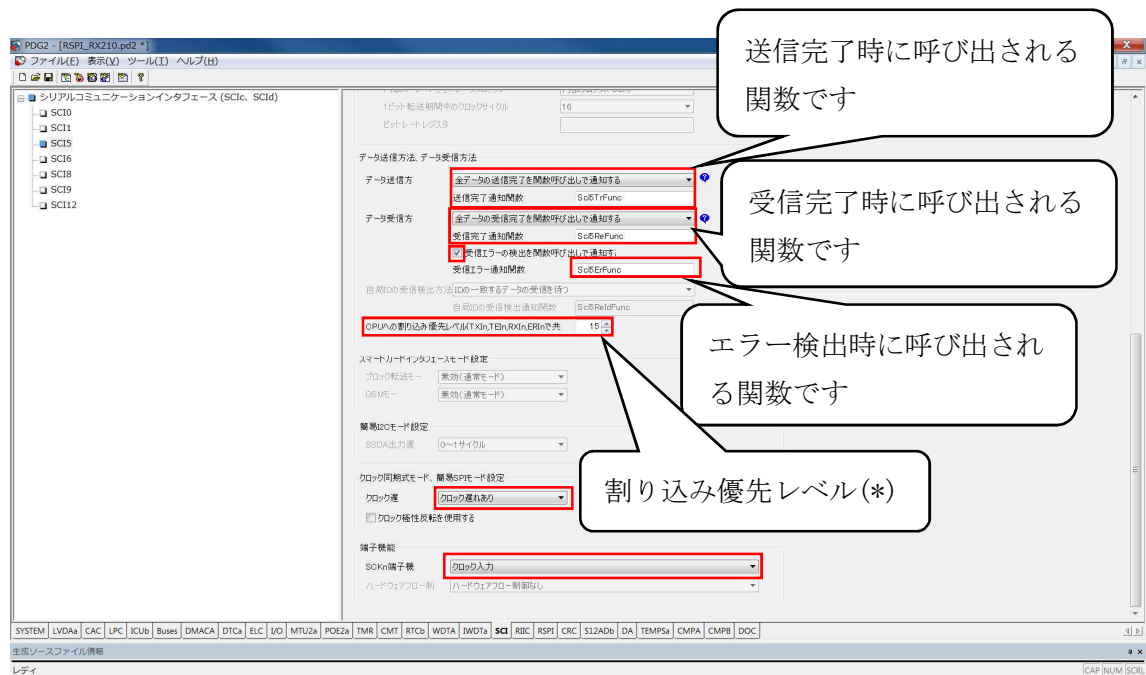
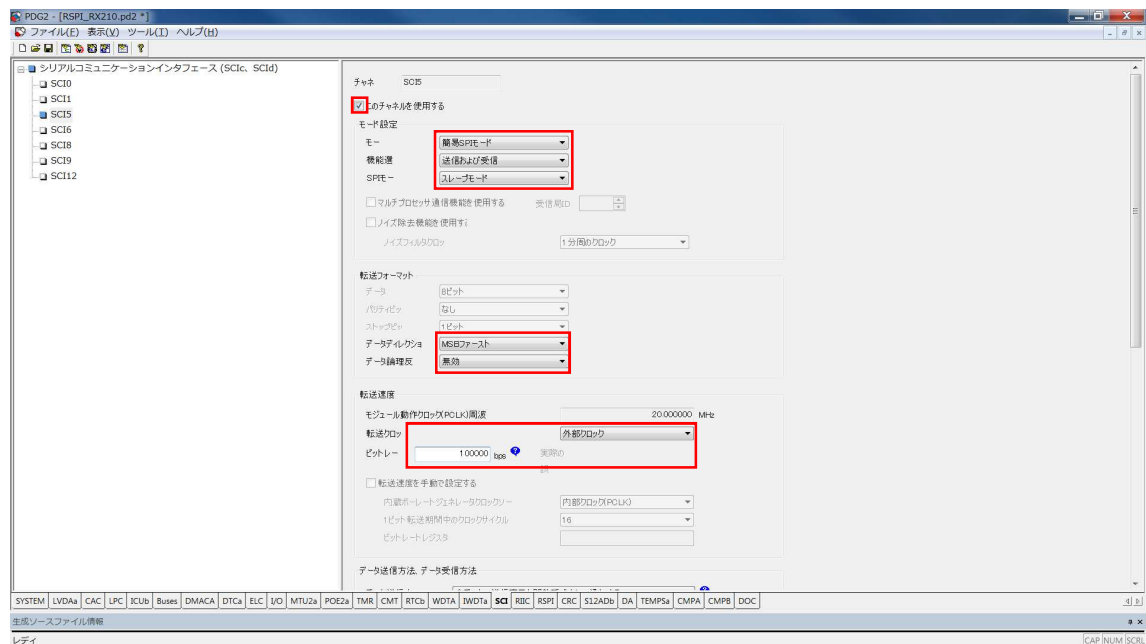
RSP10 の設定を以下に示します。



(*): マスタが送受信を開始する時には、スレーブは送受信の準備ができていないといけな
いため、マスタの優先レベル < スレーブの優先レベルとしています。

5.3 SCI5 設定

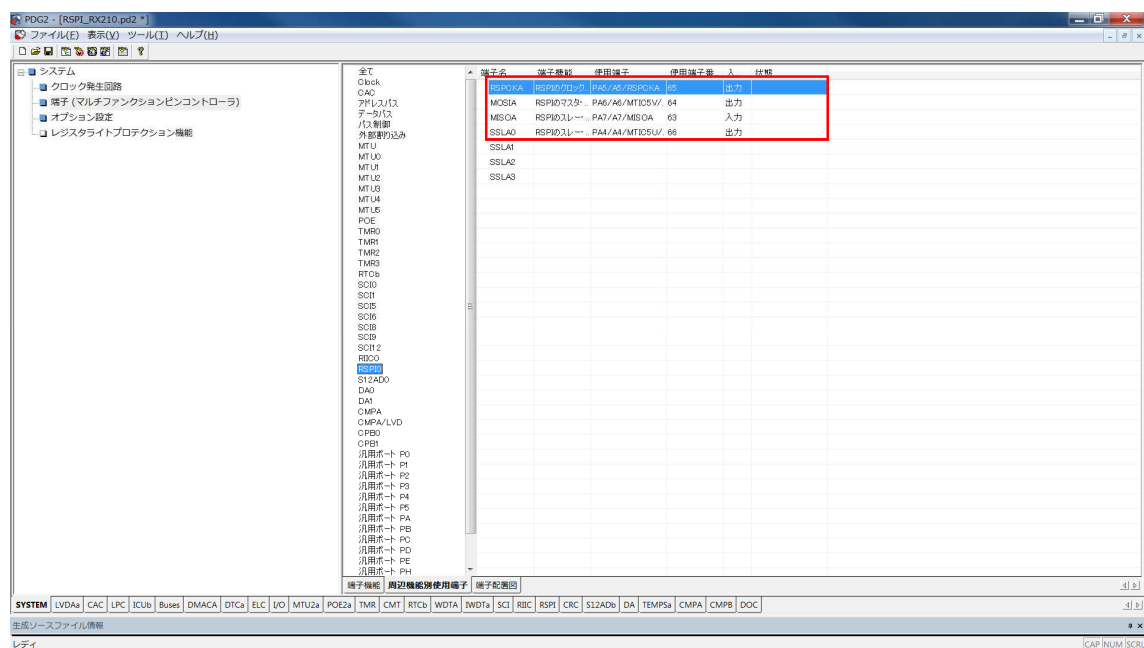
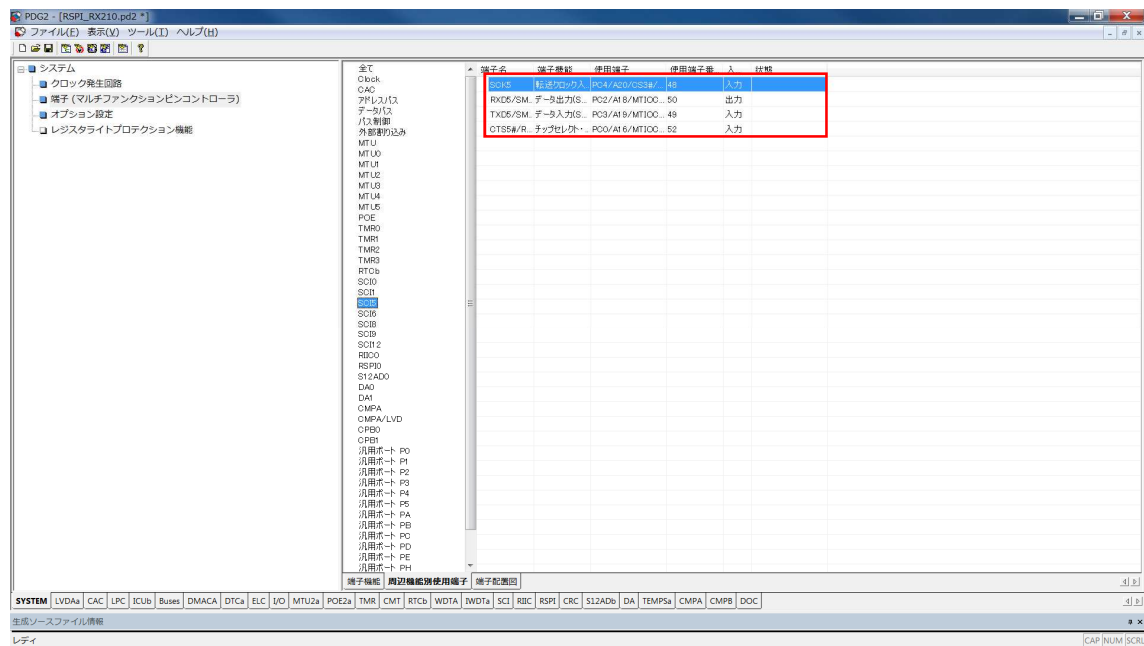
SCI5 の設定を以下に示します。



(*): マスタが送受信を開始する時には、スレーブは送受信の準備ができていないといけな
いため、マスタの優先レベル < スレーブの優先レベルとしています。

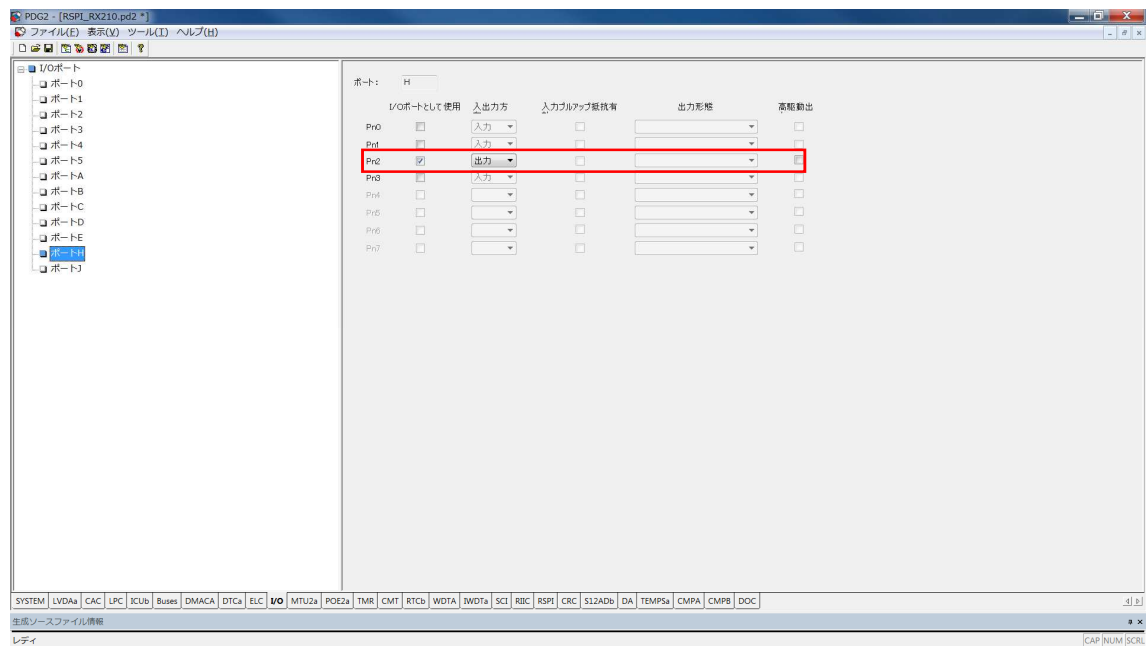
5.4 SYSTEM の端子設定

SYSTEM の端子設定を確認します。

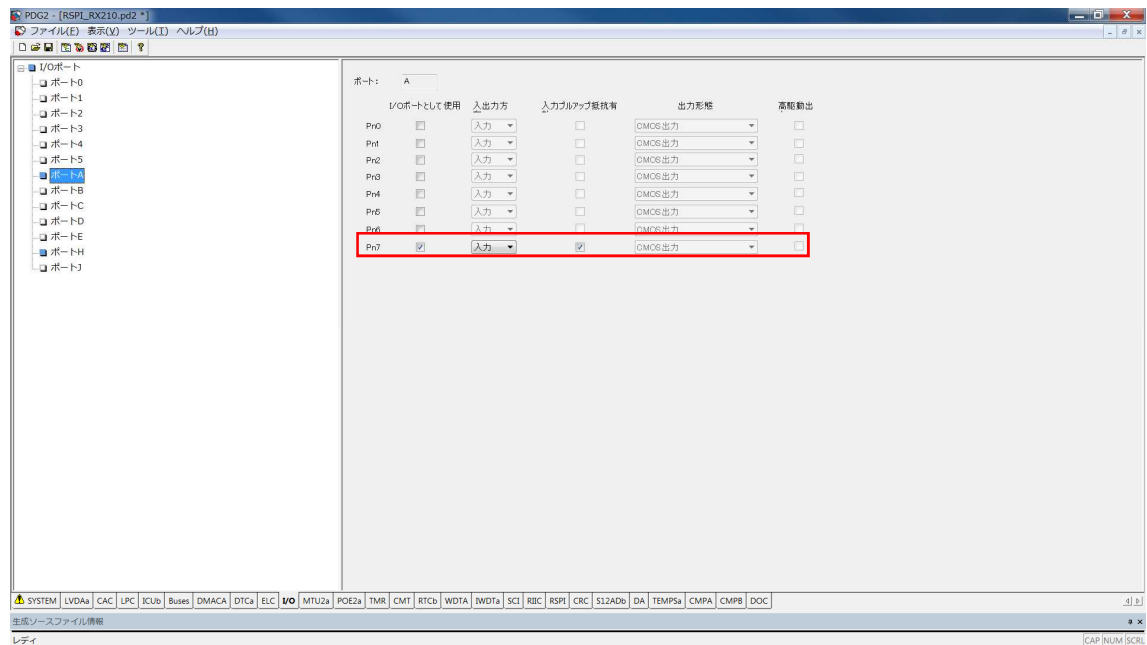


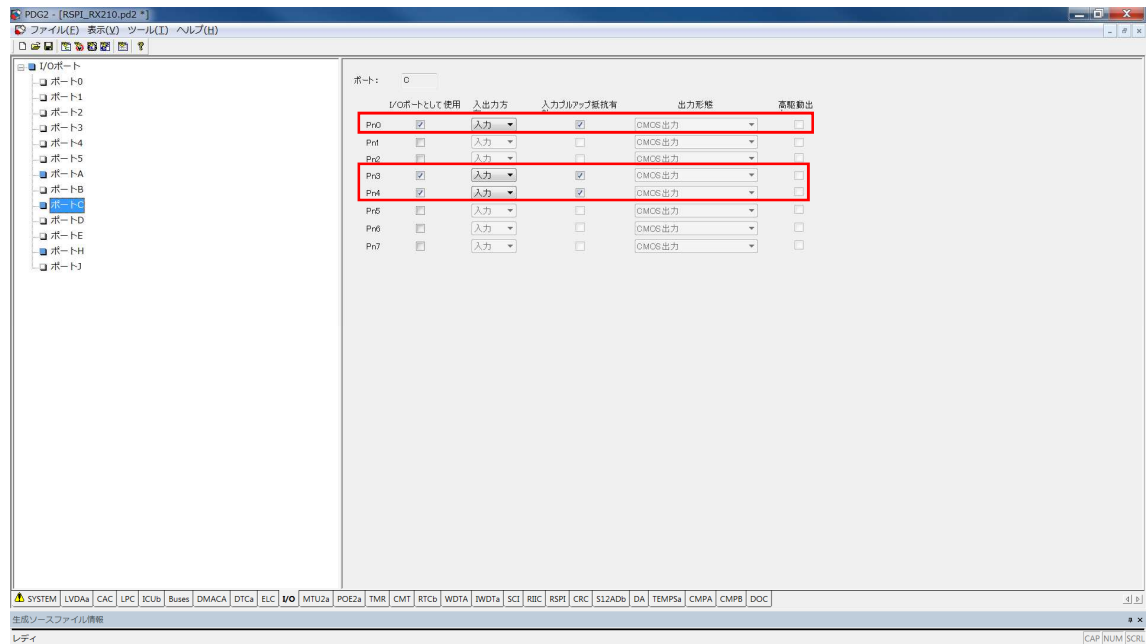
5.5 I/O 設定

I/O の設定を以下に示します。LED (D3) で使用する PH2 を出力にします。

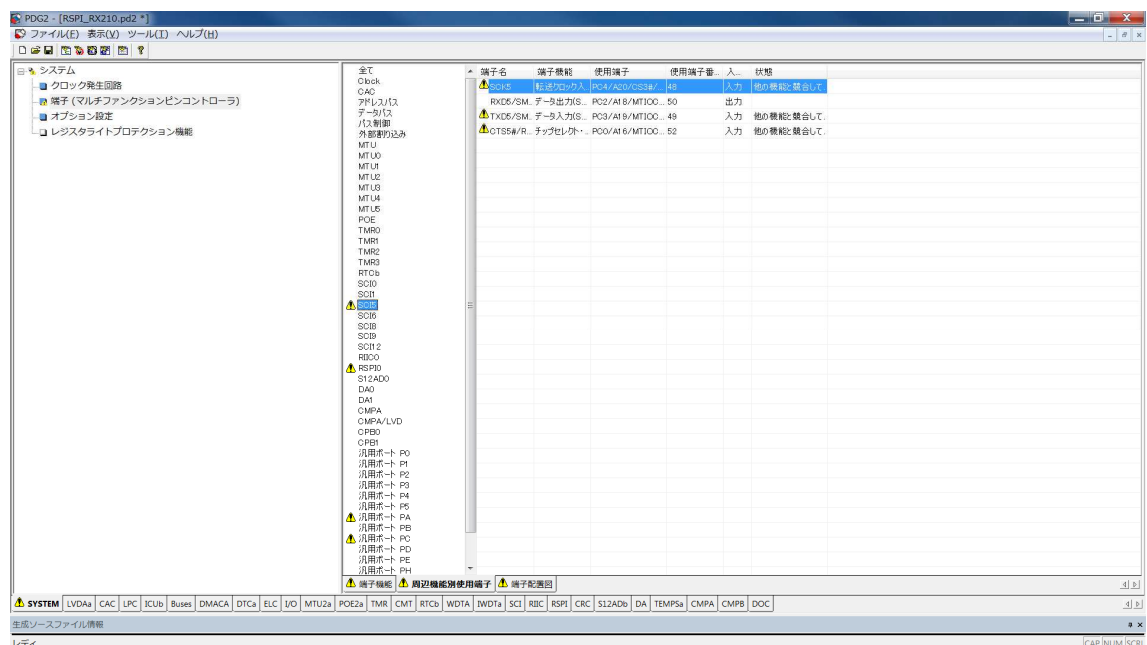


RSPI0、SCI5 の入力端子に内蔵プルアップを設定します。



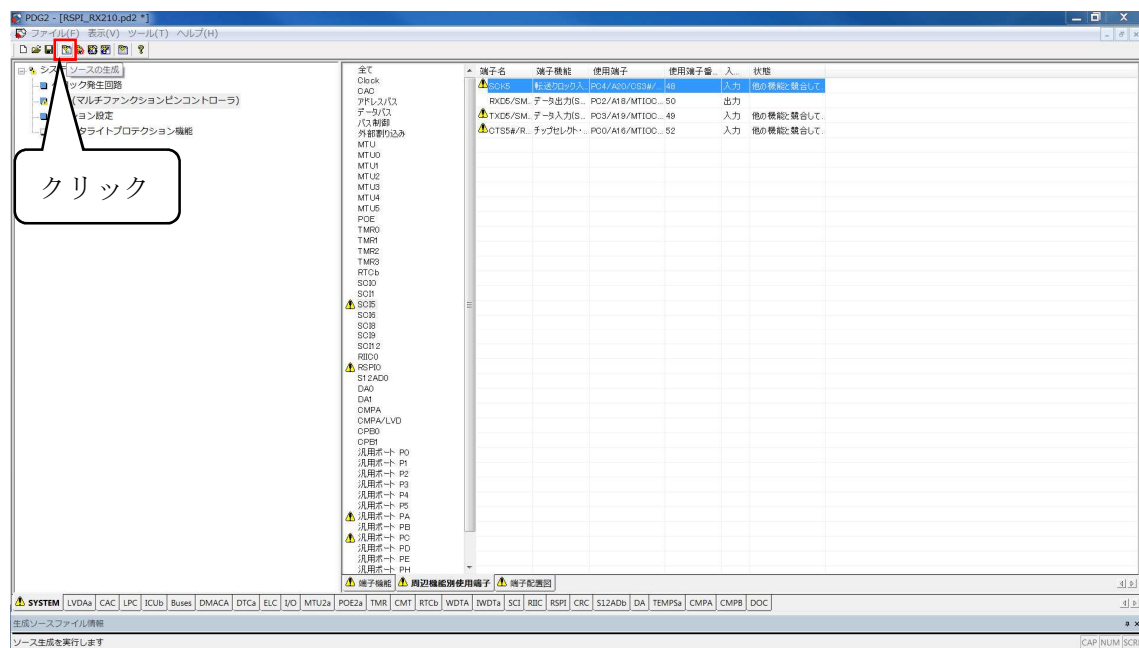


RSPI0、SCI5 の入力端子に内蔵プルアップを設定すると、SYSTEM タブの端子で以下のように汎用入力ポートと機能が競合している旨の注意が表示されましたが、ハードウェアマニュアルで各対象ポートのブロック図を参照し、内蔵プルアップが有効で問題なさそうでした（動作確認済みです）。

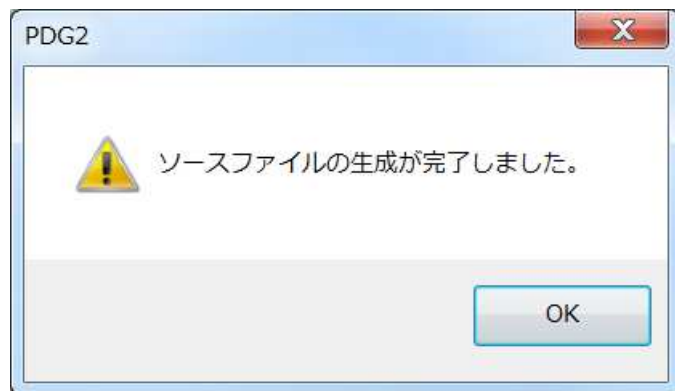


5.6 ソースの生成

以下の GUI をクリックすると、

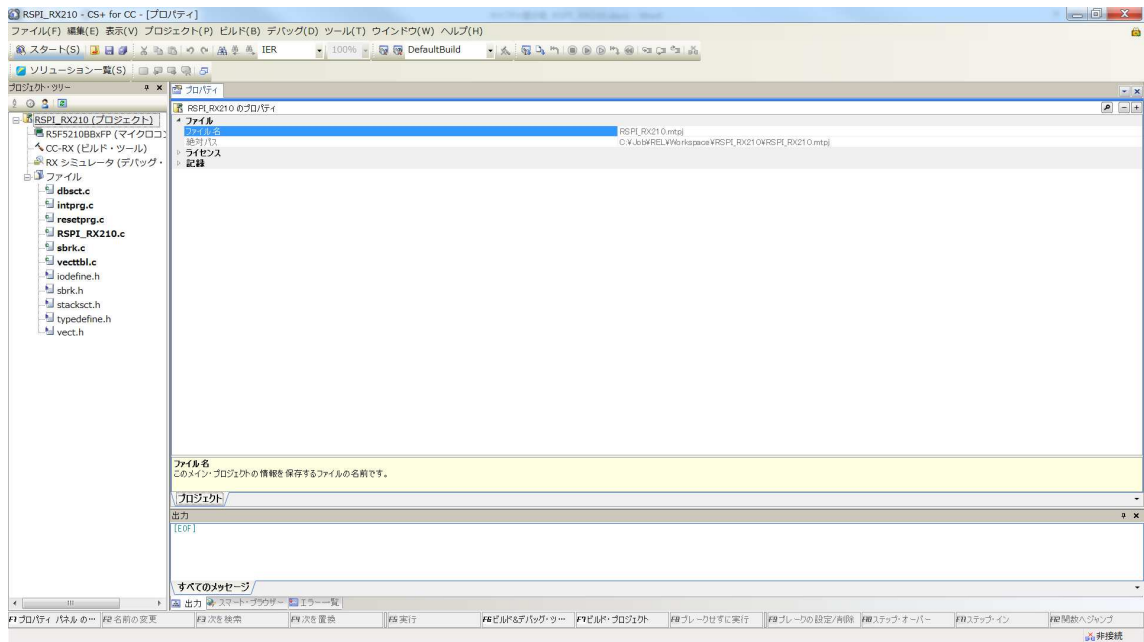


ソースファイルが生成されます。

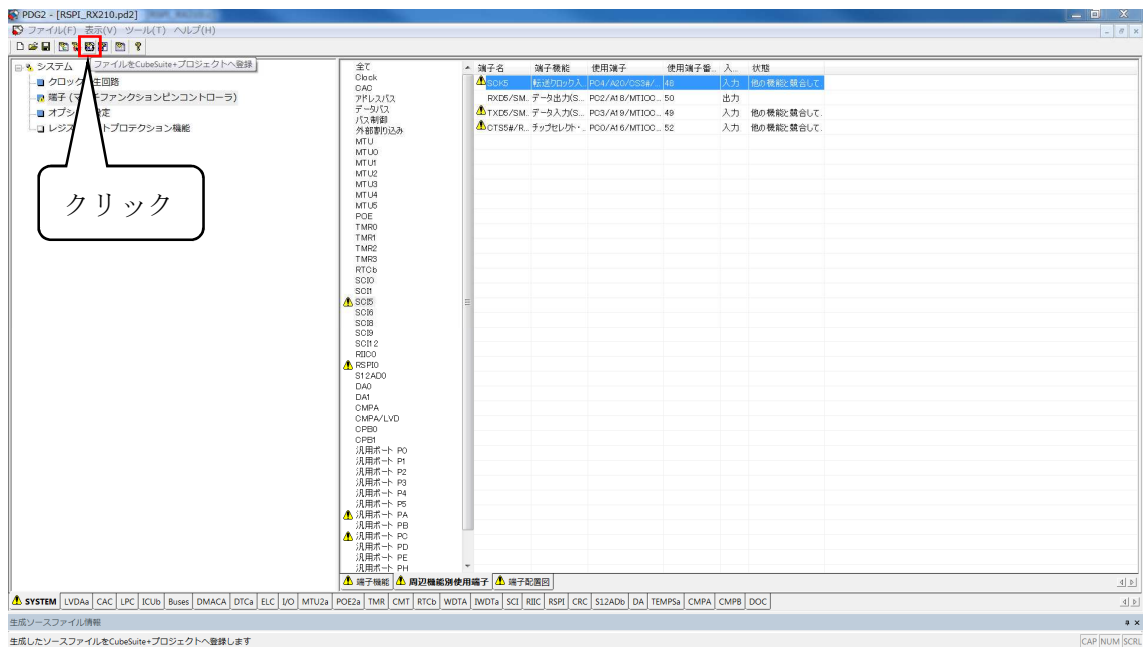


5.7 CS+への登録

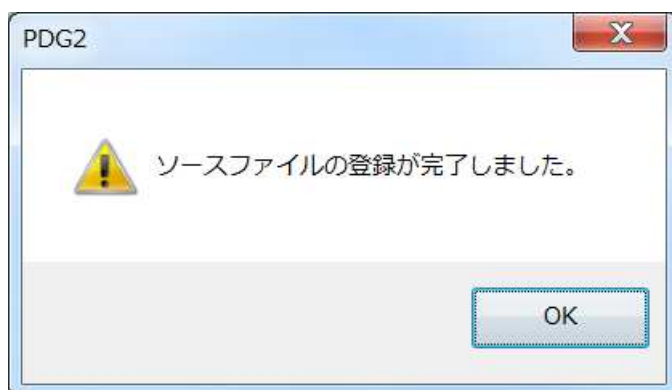
対象の CS+プロジェクトを開きます。



以下の GUI をクリックします

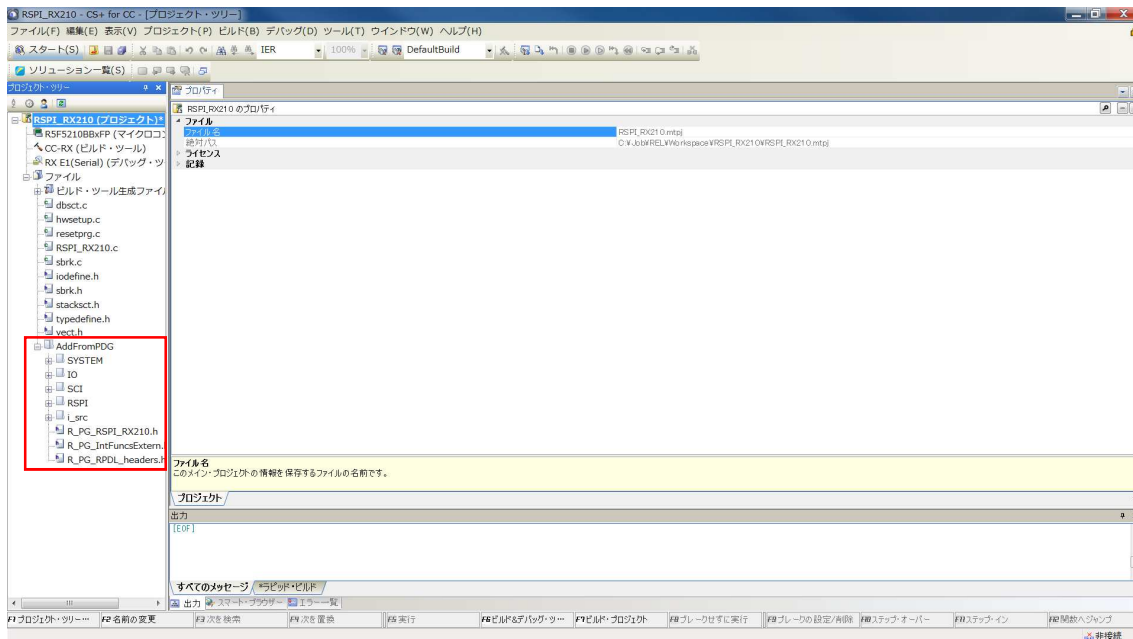


ソースファイルの登録が完了しました。



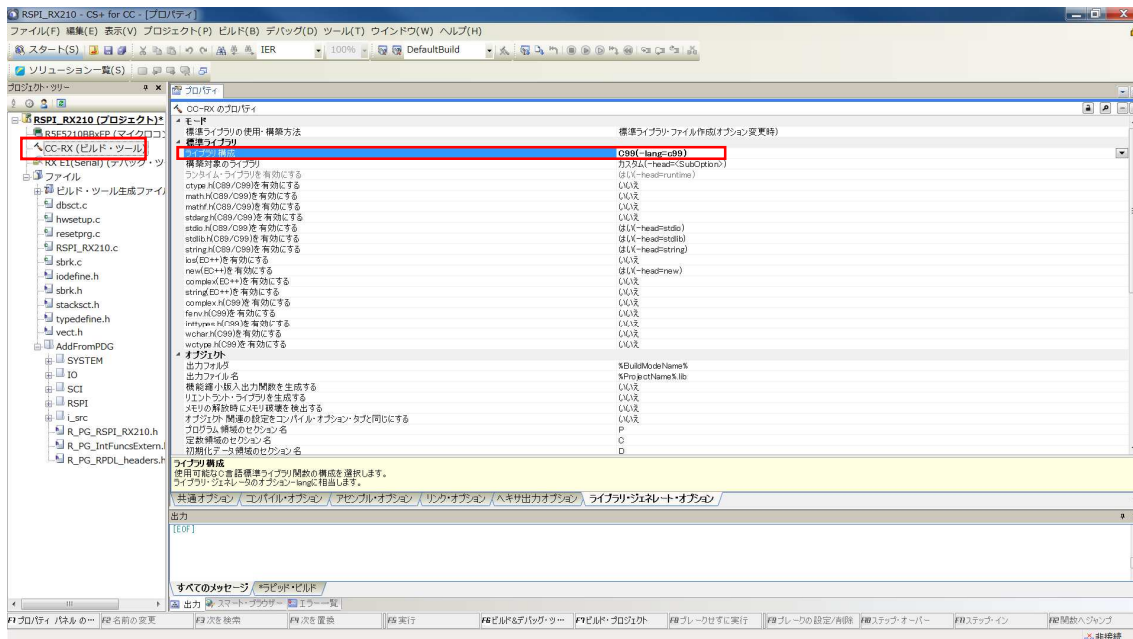
6. CS+のプロジェクトに PDG のソースファイルを登録する際の設定

CS+のプロジェクトに PDG で生成されたソースファイルを登録すると、プロジェクトのファイルに AddFromPDG フォルダが追加されます。

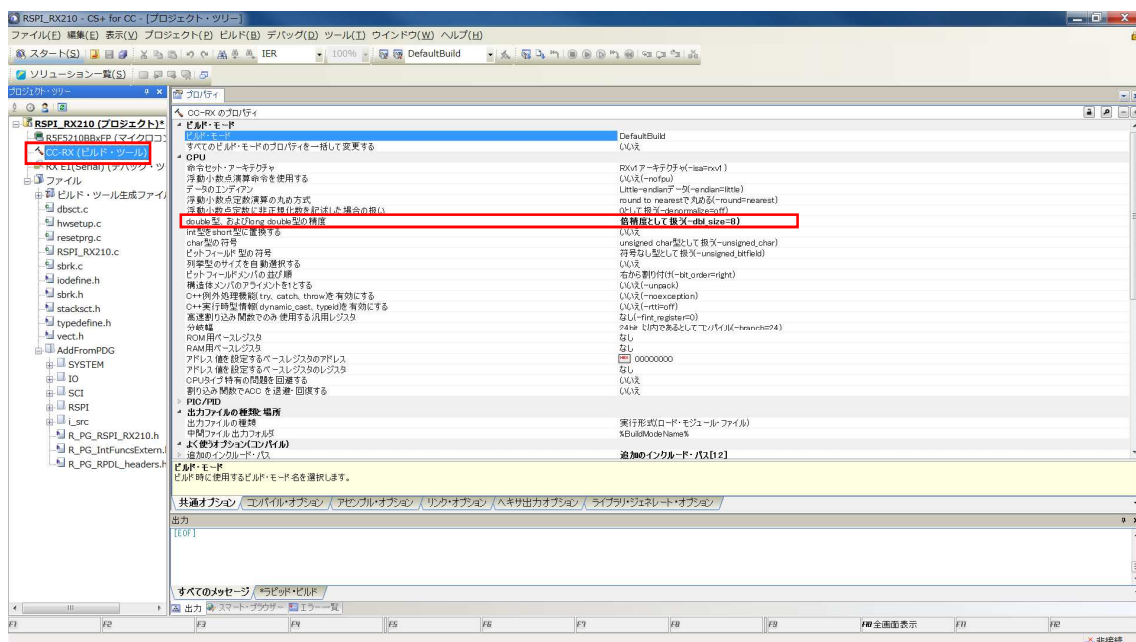


そのままビルドをすると、エラーおよび警告が発生します。解消する設定を以下に示します。

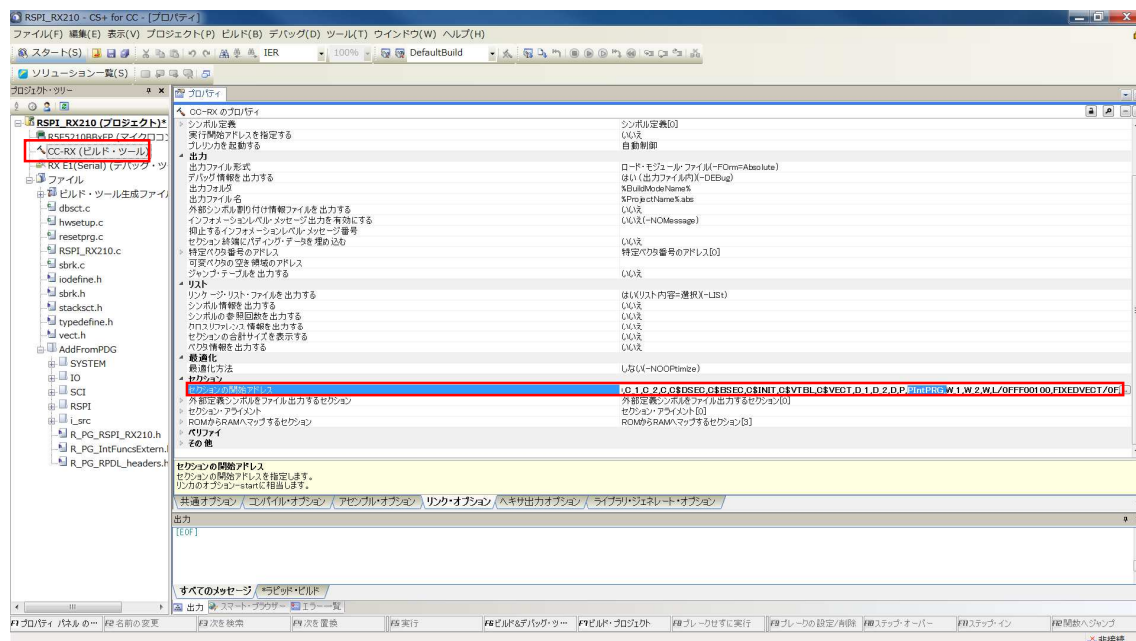
PDG で生成されるソースファイルは bool 変数を使用しています。対応させるため、ビルド・ツールを右クリック→プロパティを表示し、ライブラリ・ジェネレート・オプションタブにある「ライブラリ構成」を”C99(-lang=c99)”に設定します。



PDG で生成されるソースファイルは double 型、および long double 型の精度を倍精度として扱っているため、ビルド・ツールを右クリック→プロパティを表示し、共通オプションタブにある「double 型、および long double 型の精度」を” 倍精度として扱う(-dbl_size=8)” に設定します。



PDG で生成されるソースファイルを登録すると PIntPRG セクションを使用しないため、CS+プロジェクトを生成した際にデフォルトで設定されている PIntPRG セクションを削除します。ビルド・ツールを右クリック->プロパティを表示し、リンクオプションタブにある「セクションの開始アドレス」から” PIntPRG ” を削除します。

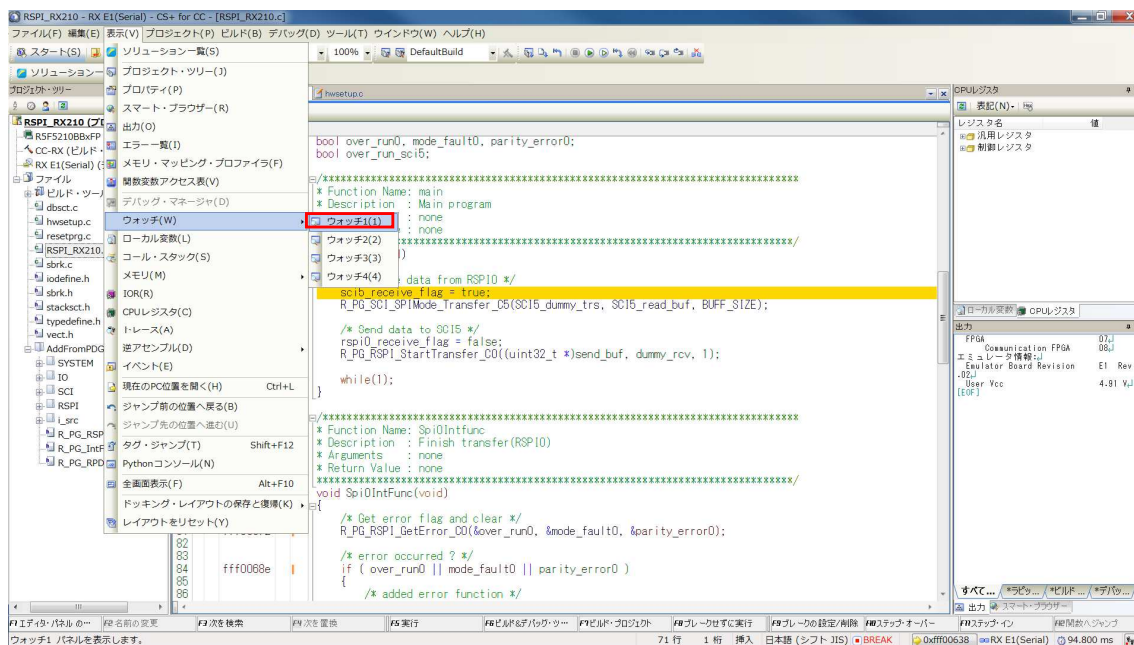


7. 動作確認方法

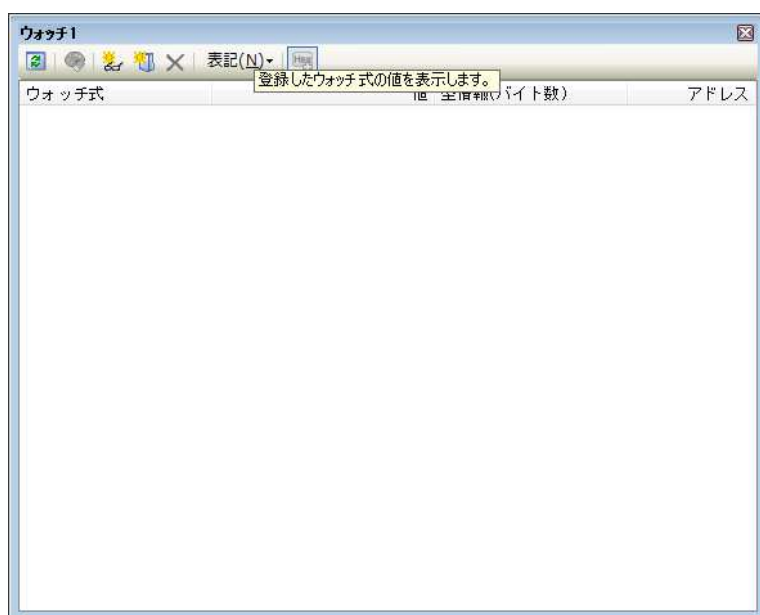
7.1 ウォッチ式の登録

RSPIO から SCI5 にデータが送信され、SCI5 の受信データを RSPIO に送り返されているかを確認する 1 つの方法として、CS+に搭載されている機能のウォッチ式を使用しました（エミュレータを使用しない場合は、LED(D3)の点灯でのみ動作を確認）。ウォッチ式に登録することで対象の変数に格納されているデータを確認することができます。使用方法を以下に示します。

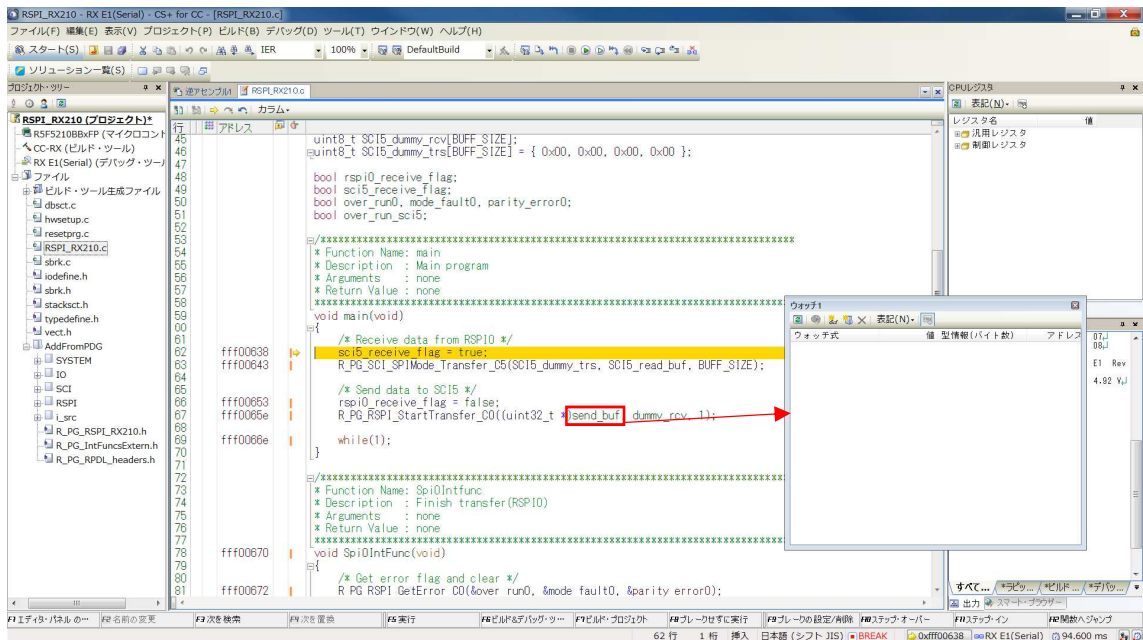
表示タブからウォッチ、そしてウォッチ 1 を選択します。



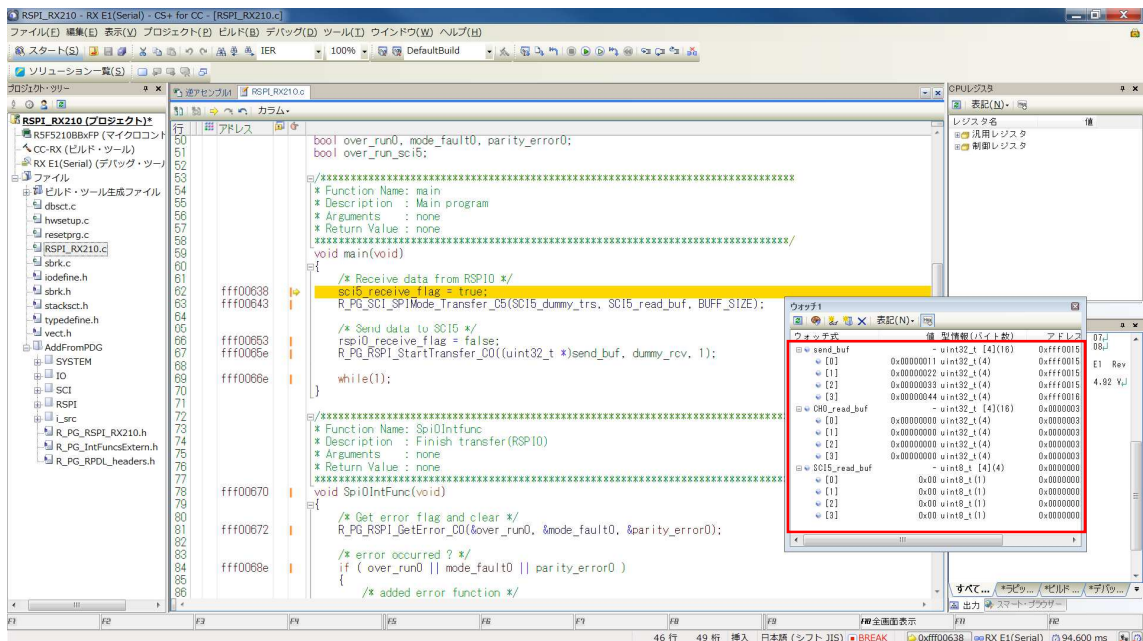
ウォッチ(ウォッチ 1)パネルが表示されました。



ウォッチ1が表示されたら main 関数の中の“send_buf”をウォッチパネルにドラッグ&ドロップします。



“CH0_read_buf”と“SCI5_read_buf”も同様にドラッグ&ドロップします。

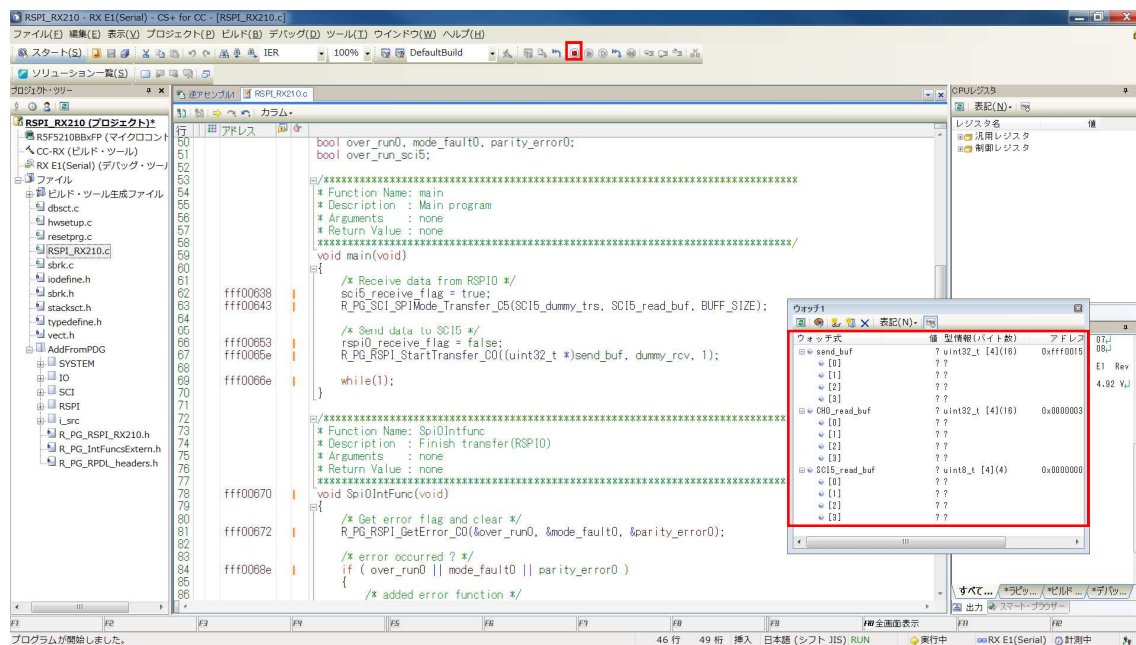


上の図のようにウォッチパネルには“send_buf”にのみ 00000011h, 00000022h, 00000033h, 00000044h が表示され、“CH0_read_buf”と“SCI5_read_buf”は全て 00000000h (00h) であることが確認できます。ボード側は、LED (D3) は消灯となっています。この状態で実行前の準備は完了です。

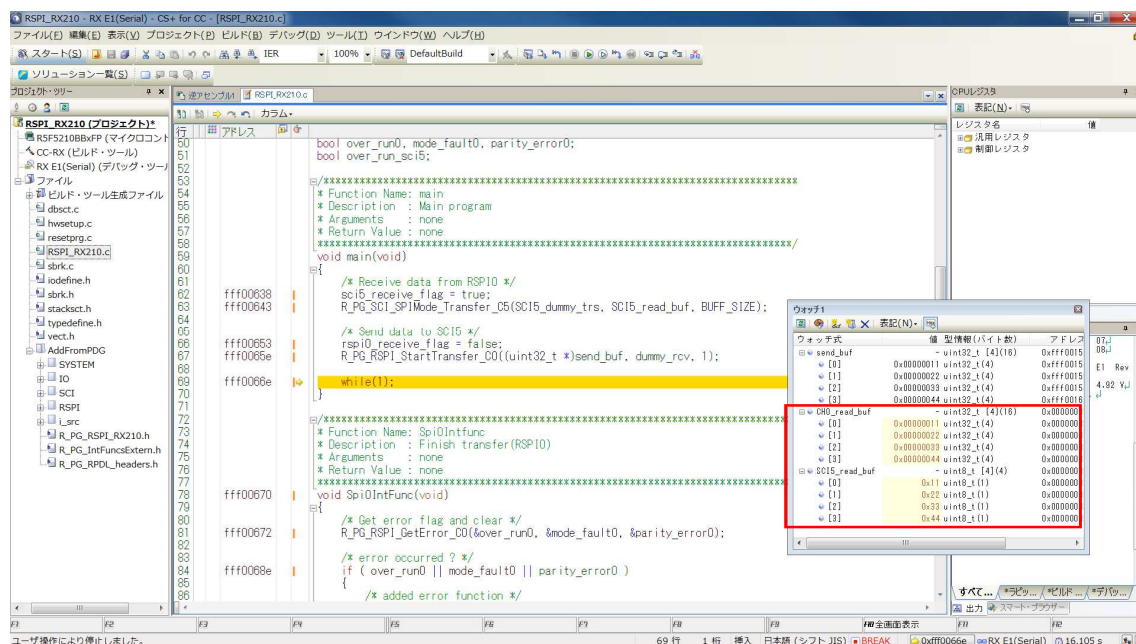
7.2 実行

実行ボタンをクリックします。

実行中はウォッチパネルのウォッチ式の値は全て“?”マーク表示です。なお、LED1は点灯しました。ここで、停止をクリックします。



プログラムを停止すると、ウォッチパネルで“CH0_read_buf”と“SCI5_read_buf”は共に 00000011h(11h), 00000022h(22h), 00000033h(33h), 00000044h(44h)が表示されています。これで SCI5 は RSPiO から 00000011h(11h), 00000022h(22h), 00000033h(33h), 00000044h(44h)を受信できており、RSPiO は SCI5 から送り返された 00000011h(11h), 00000022h(22h), 00000033h(33h), 00000044h(44h)を受信できたことを確認できました。



8. 参考ドキュメント

RX210 グループ ユーザーズマニュアルハードウェア編

RX210 グループ Peripheral Driver Generator リファレンスマニュアル

以上