RX63N 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

・RX63N

内容

1. 仕様

RSPI0（マスタモード）から RSPI1（スレーブモード）へ SPI（4 線式）通信でデータを送信します。RSPI1 は受信したデータを RSPI0 に送り返します。RSPI0 が RSPI1 へ送信したデータと RSPI1 が RSPI0 へ送り返したデータが一致した場合、LED1 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	R5F563NFDDFP（RX63N グループ）
動作周波数	・メインクロック：12MHz ・PLL：192MHz（メインクロック 1 分周 16 逓倍） ・システムクロック(ICLK)：96MHz（PLL 2 分周） ・周辺モジュールクロック B(PCLKB)：48MHz（PLL 4 分周）
ボード電源電圧	5V
マイコン動作電圧	3.3V
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 HSBRX63NP(R5F563NFDDFP)

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
PA5	出力	RSPCKA（マスタクロック出力）
PA6	出力	MOSIA（マスタデータ出力）
PA7	入力	MISOA（マスタデータ入力）
PC4	出力	SSLA0（マスタチップセレクト出力）
PE1	入力	RSPCKB（スレーブクロック入力）
PE2	入力	MOSIB（スレーブデータ入力）
PE3	出力	MISOB（スレーブデータ出力）
PE4	入力	SSLB0（スレーブチップセレクト入力）
PD6	出力	LED1

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

	RSPI0	RSPI1
接続 1	J1 の 27pin (PA5/RSPCKA)	J1 の 38pin (PE1/RSPCKB)
接続 2	J1 の 24pin (PA6/MOSIA)	J1 の 37pin (PE2/MOSIB)
接続 3	J1 の 25pin (PA7/MISOA)	J1 の 36pin (PE3/MISOB)
接続 4	J1 の 10pin (PC4/SSLA0)	J1 の 35pin (PE4/SSLB0)

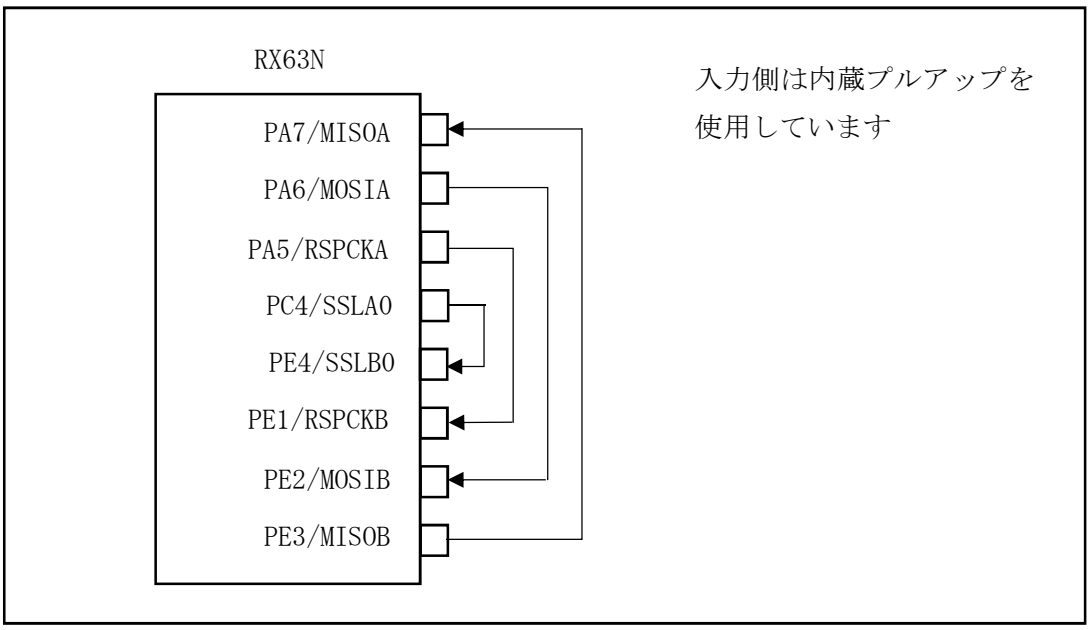


図 3.1　信号接続図

4. ソフトウェア説明

4.1 動作概要

図 4.1 に RSPI0 → RSPI1 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。

※RSPI0、RSPI1 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

RSPI0 から RSPI1 へデータを送信するために、以下の初期設定を行います。

●RSPI0

・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

●RSPI1

・SPI 動作（4 線式）スレーブモード

・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

・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”

(2) RSPI1（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。

※RSPI1（スレーブ）は SSLB0 端子に”L”が入力されることで RSPCKB 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 RSPI1 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。

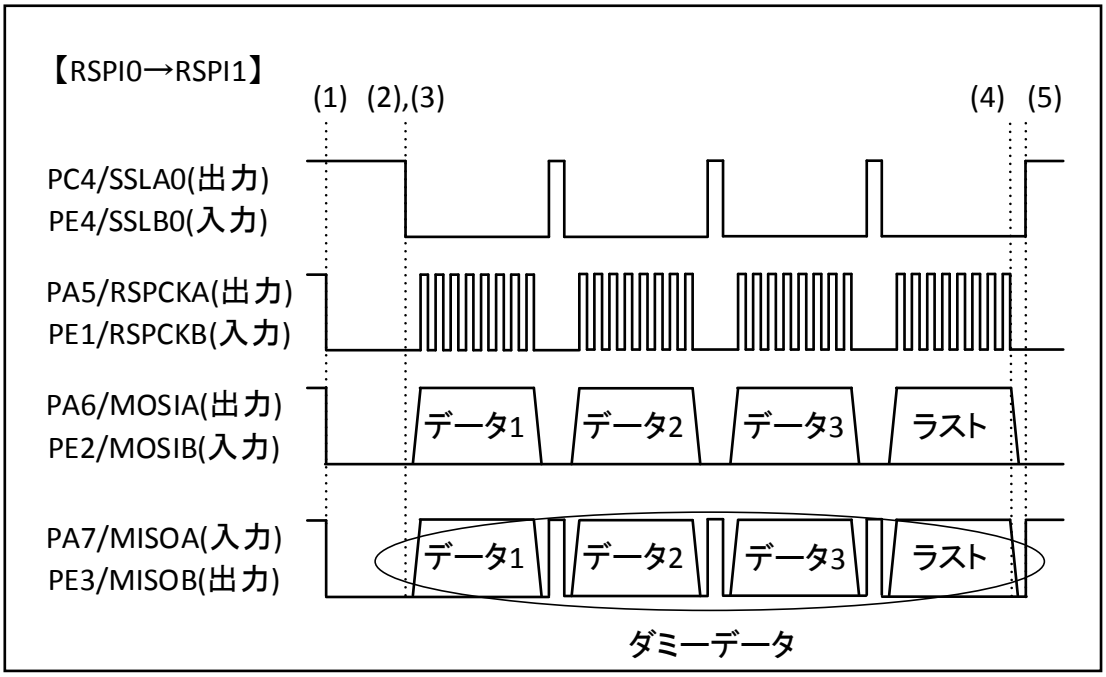


図 4.1　RSPI0 → RSPI1 の動作タイミング

図 4.2 に RSPI1 → RSPI0 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) RSPI1（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 RSPI0 → RSPI1 で受信したデータ（11h、22h、33h、44h）を使用します。

(2) 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、ダミーデータ（00h、00h、00h、00h）を送信します。

※RSPI1（スレーブ）は SSLB0 端子に”L”が入力されることで RSPCKB 端子に入力されるクロックに同期し、送受信します。

(3) RSPI0 の送受信完了および RSPI1 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。

※RSPI0 で RSPI1 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、LED1 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。

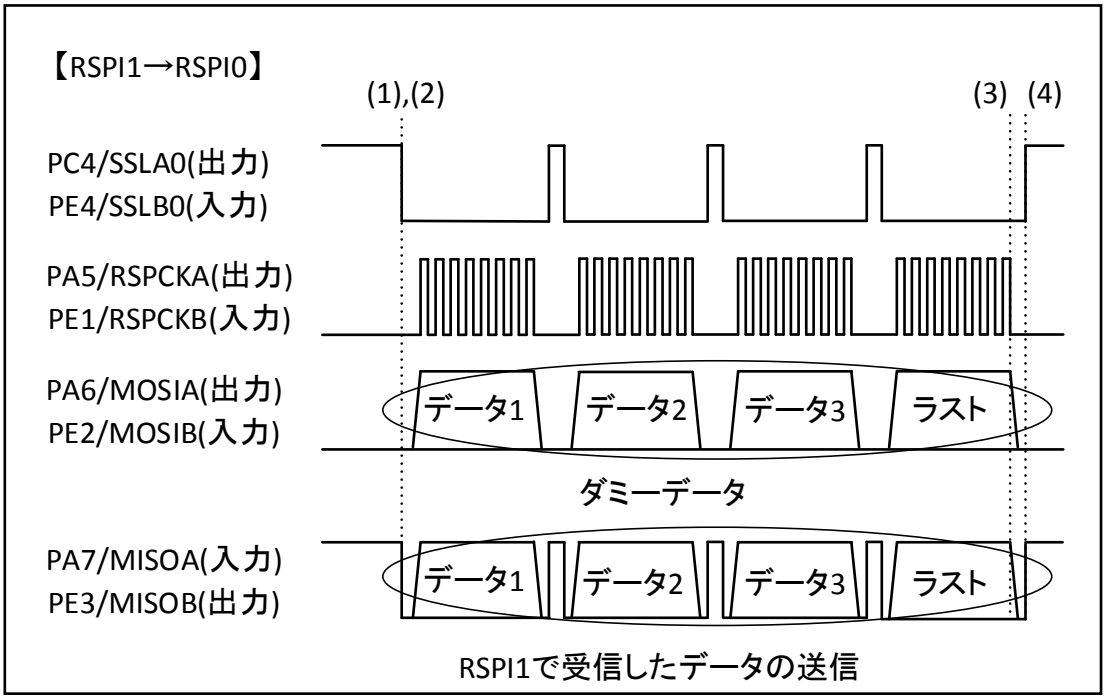


図 4.2　RSPI1 → RSPI0 の動作タイミング

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_RX63N.c	メインファイル ・RSPI1 送信/受信処理 ・RSPI0 送信/受信処理 ・LED1 制御 ・オプション設定メモリ	
hwsetup.c	初期設定 ・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 ・RSPI1 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

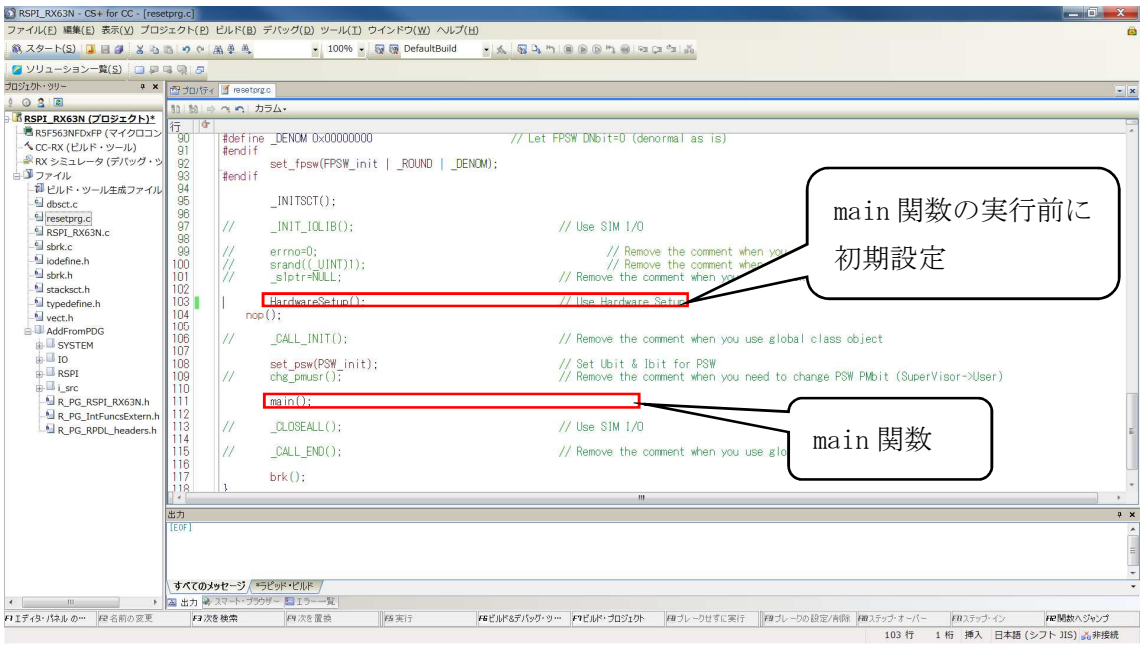


図 4.3　resetprg.c

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 HOCO(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 Spi1IntFunc
unsigned long	CH1_read_buf[BUFF_SIZE]	RSPI1 の受信バッファ	main Spi1IntFunc
unsigned long	dummy_rcv[BUFF_SIZE]	ダミー用受信バッファ	main Spi1IntFunc
unsigned long	dummy_trs[BUFF_SIZE]	ダミー用送信バッファ	main Spi1IntFunc
bool	rspi0_receive_flag	RSPI0 用受信フラグ（受信データ使用時 true）	main Spi0IntFunc Spi1IntFunc
bool	rspi1_receive_flag	RSPI1 用受信フラグ（受信データ使用時 true）	main Spi1IntFunc
bool	over_run0	RSPI0 用オーバ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_run1	RSPI1 用オーバランエラーフラグ格納	Spi1ErIntFunc
bool	mode_fault1	RSPI1 用モードフォールトエラーフラグ格納	Spi1ErIntFunc
bool	parity_error1	RSPI1 用パリティエラーフラグ格納	Spi1ErIntFunc

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、「RX63N グループ、RX631 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
Spi1IntFunc	RSPI1 転送完了関数
Spi0ErIntFunc	RSPI0 エラー検出関数
Spi1ErIntFunc	RSPI1 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	RSPI1 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	・受信データを使用時 送信データと受信データの比較 LED1 制御 ・その他 処理なし
引数	なし
リターン値	なし

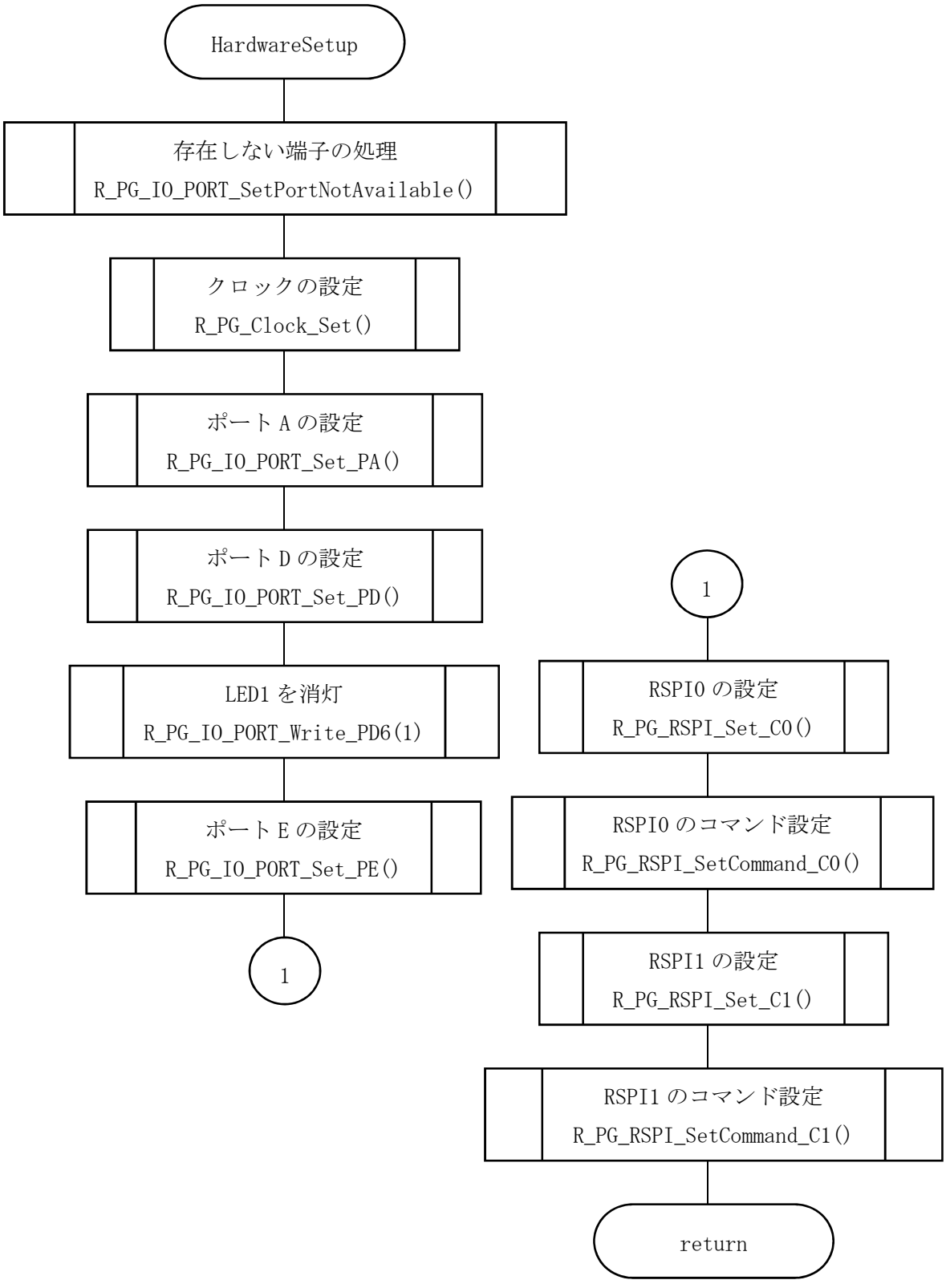
Spi1IntFunc	
概要	RSPI1 転送完了処理
ヘッダ	なし
宣言	void Spi1IntFunc(void)
説明	・受信データを使用時 RSPI1 の受信データを送信データとして送受信開始 RSPI0 の送受信開始 ・その他 処理なし
引数	なし
リターン値	なし

Spi0ErIntFunc	
概要	RSPI0 エラー検出処理
ヘッダ	なし
宣言	void Spi0ErIntFunc (void)
説明	エラーフラグの取得およびクリア
引数	なし
リターン値	なし

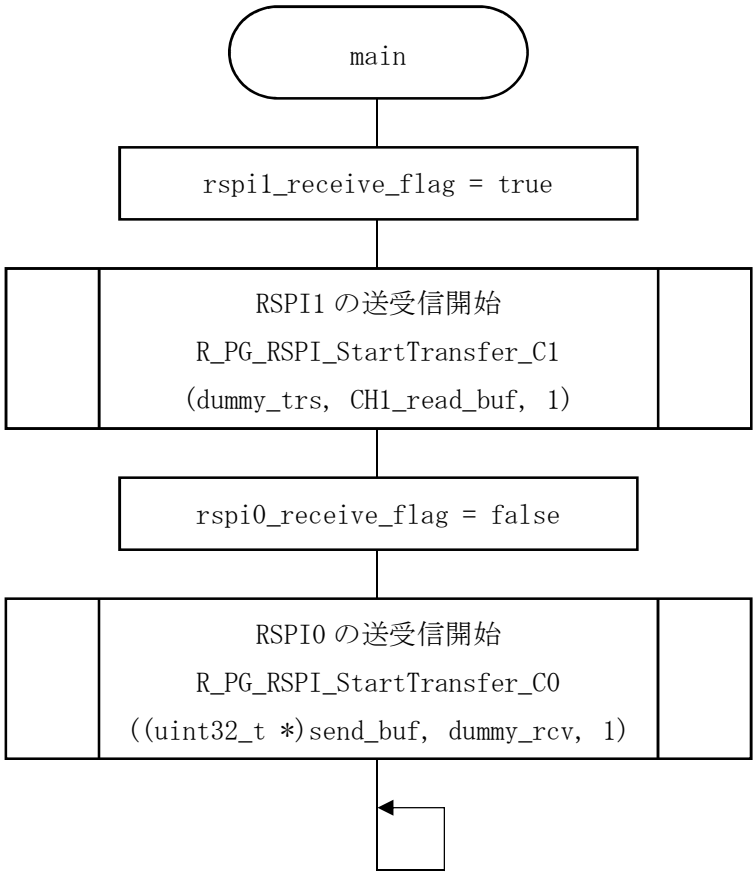
Spi1ErIntFunc	
概要	RSPI1 エラー検出処理
ヘッダ	なし
宣言	void Spi1ErIntFunc (void)
説明	エラーフラグの取得およびクリア
引数	なし
リターン値	なし

4.8　　作成する関数のフローチャート

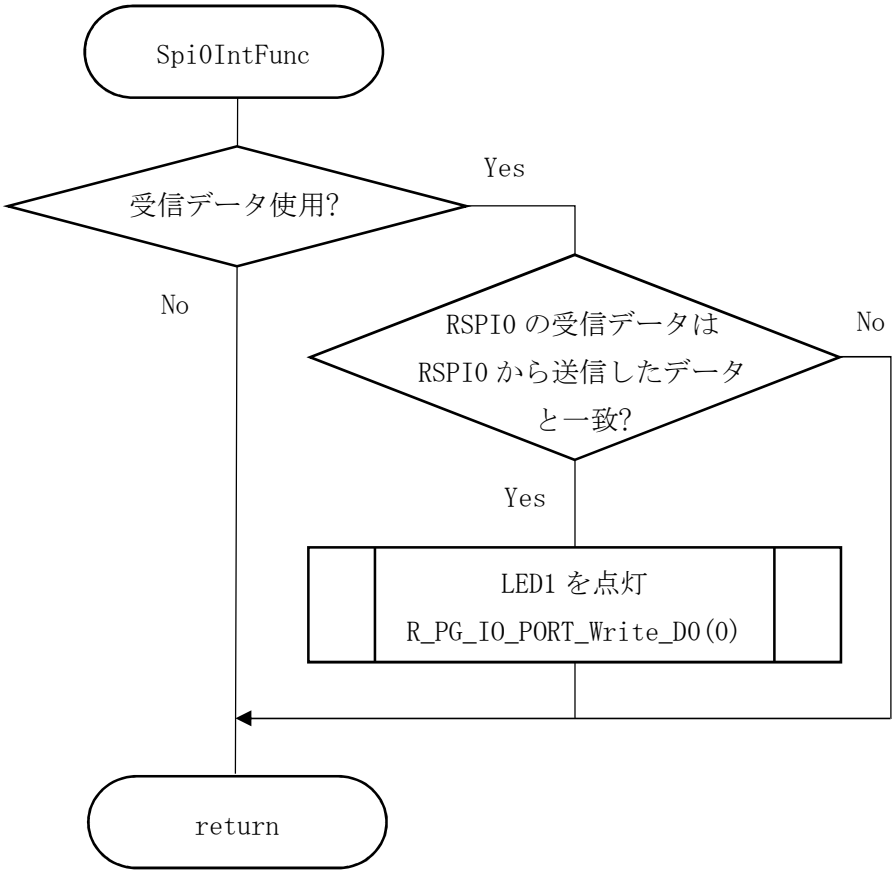
4.8.1　　初期設定



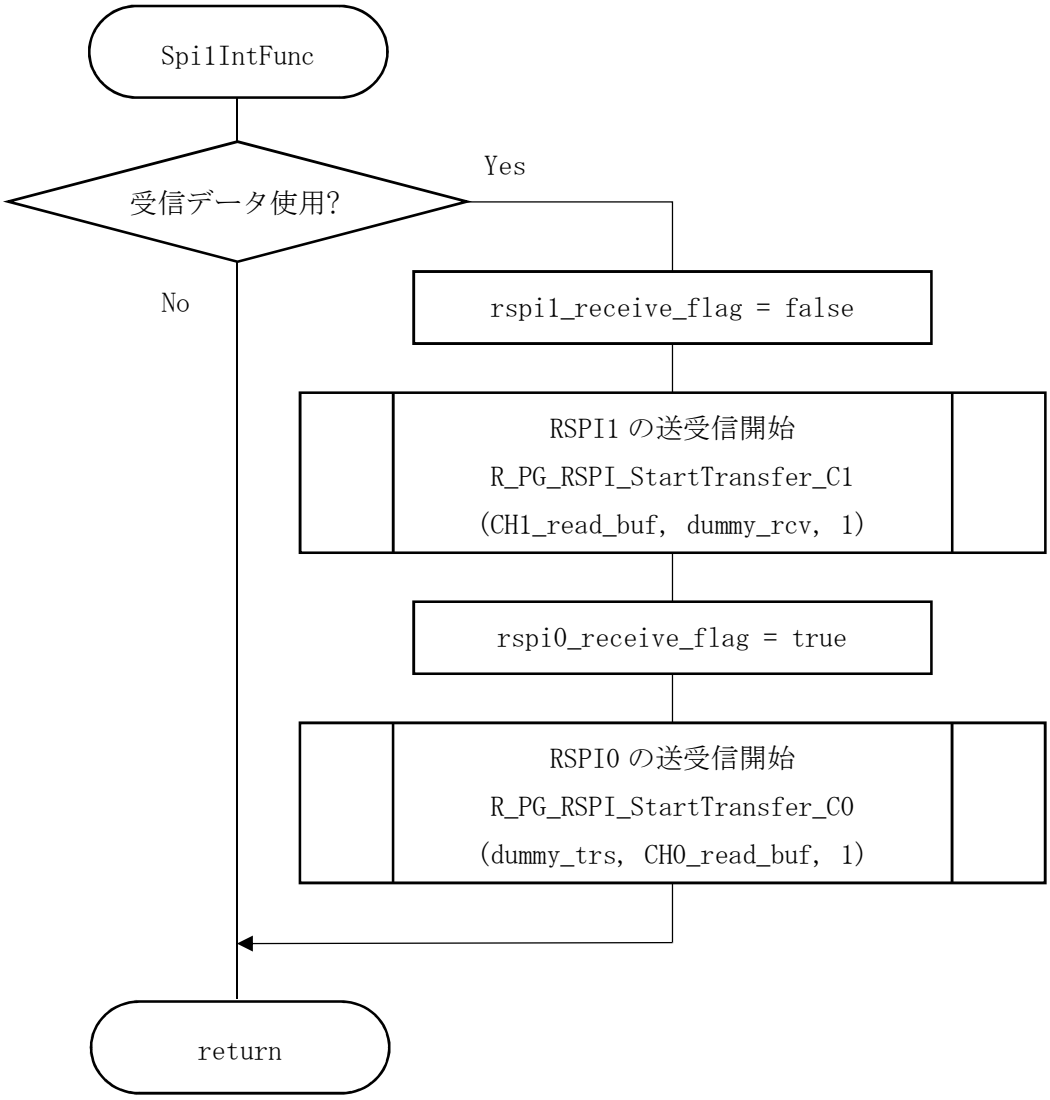
4.8.2　　メイン処理


main
rspi1_receive_flag = true
RSPI1 の送受信開始
R_PG_RSPI_StartTransfer_C1
(dummy_trs, CH1_read_buf, 1)
rspi0_receive_flag = false
RSPI0 の送受信開始
R_PG_RSPI_StartTransfer_C0
((uint32_t *)send_buf, dummy_rcv, 1)

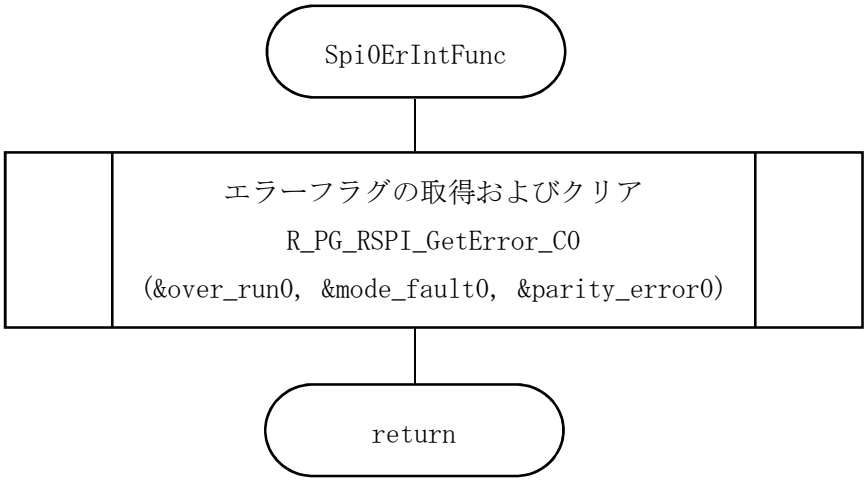
4.8.3　　RSPI0 転送完了処理



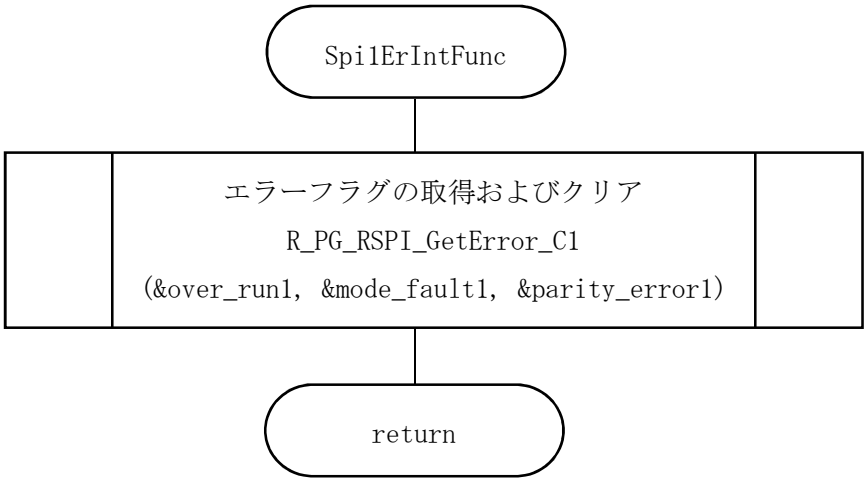
4.8.4　　RSPI1 転送完了処理



4.8.5　　RSPI0 エラー検出処理



4.8.6　　RSPI1 エラー検出処理

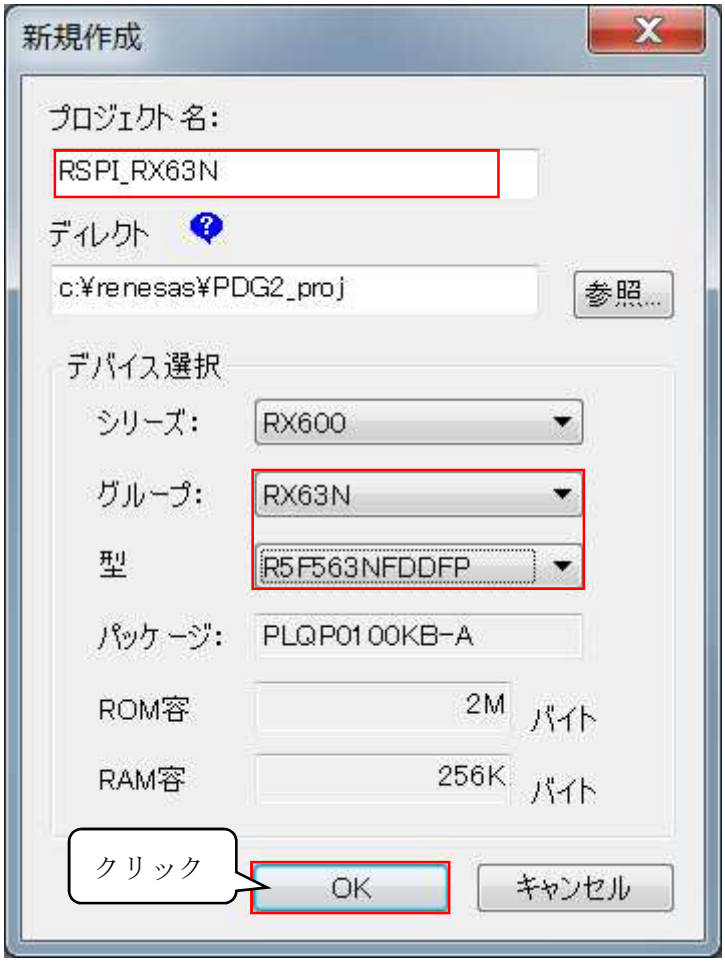


5. PDG の設定

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”RX63N グループ、RX631 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。

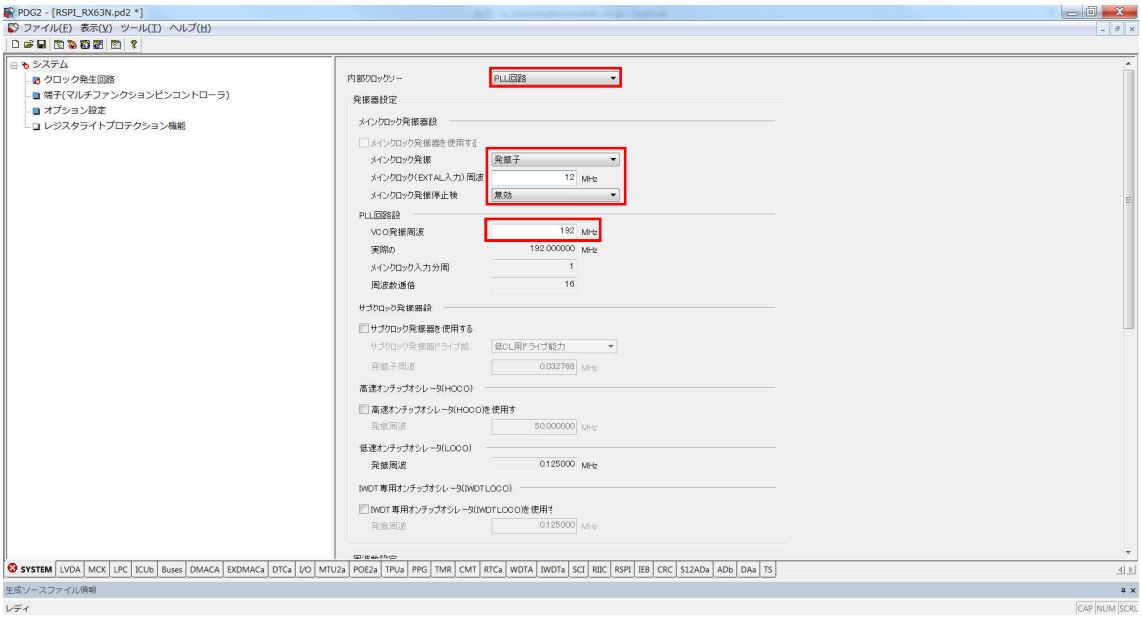
Peripheral Driver Generator 2 を起動します。

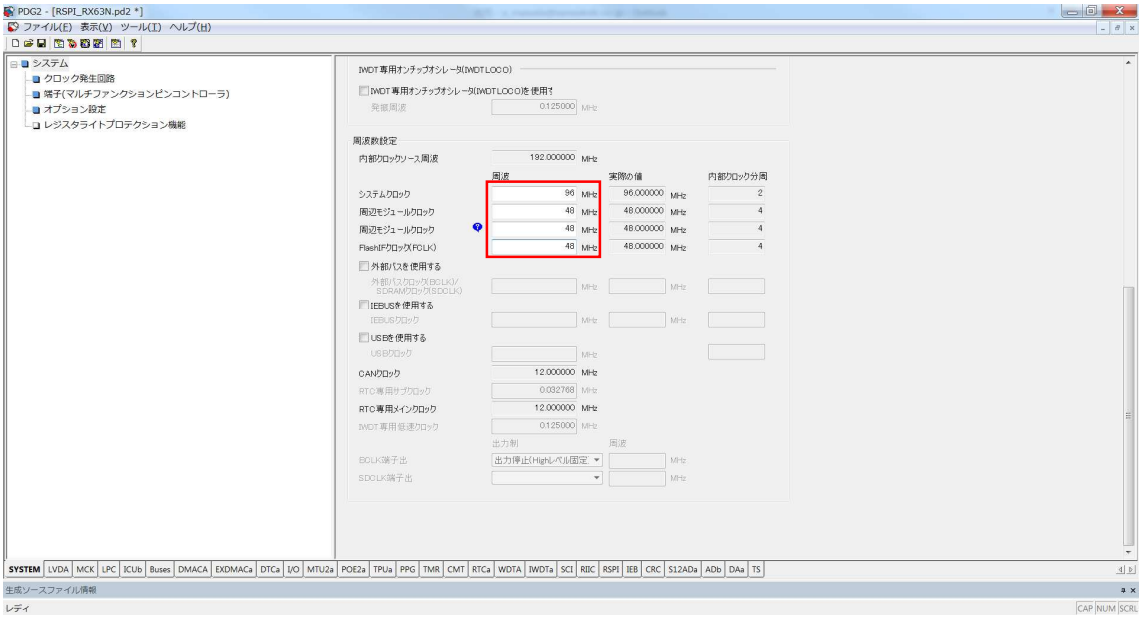
メニューバーのファイル->プロジェクトの新規作成 をクリックすると、以下のウィンドウが表示されます。プロジェクト名、マイコンのグループ、型を入力し、「OK」をクリックすると、プロジェクトが作成されます。



5.1　　SYSTEM 設定

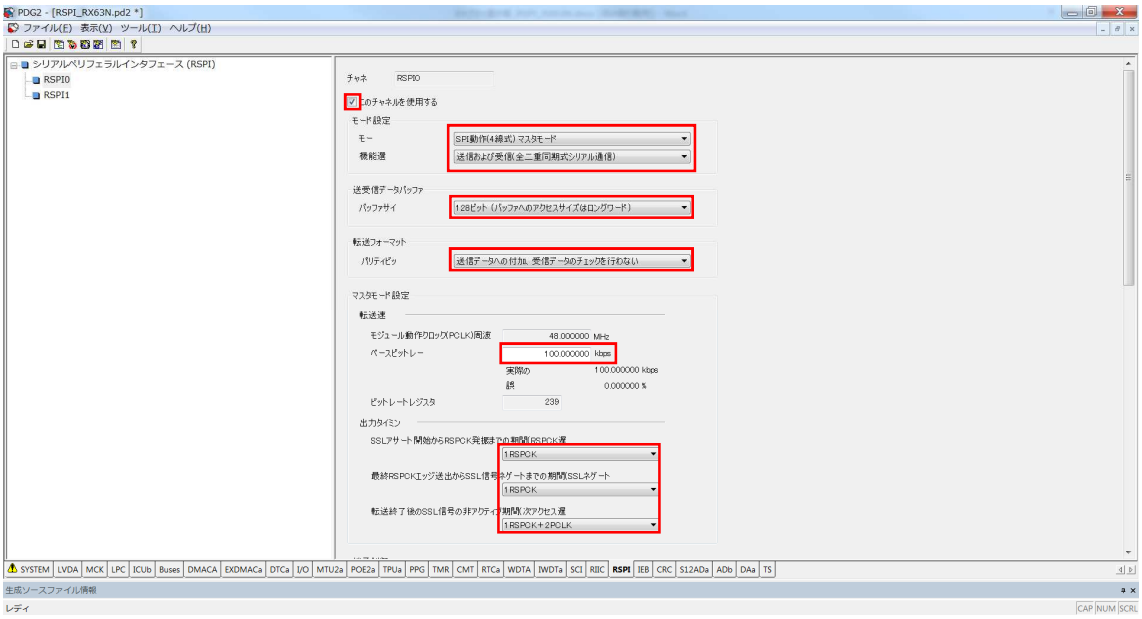
システムタブのクロック発生回路の設定を以下に示します。

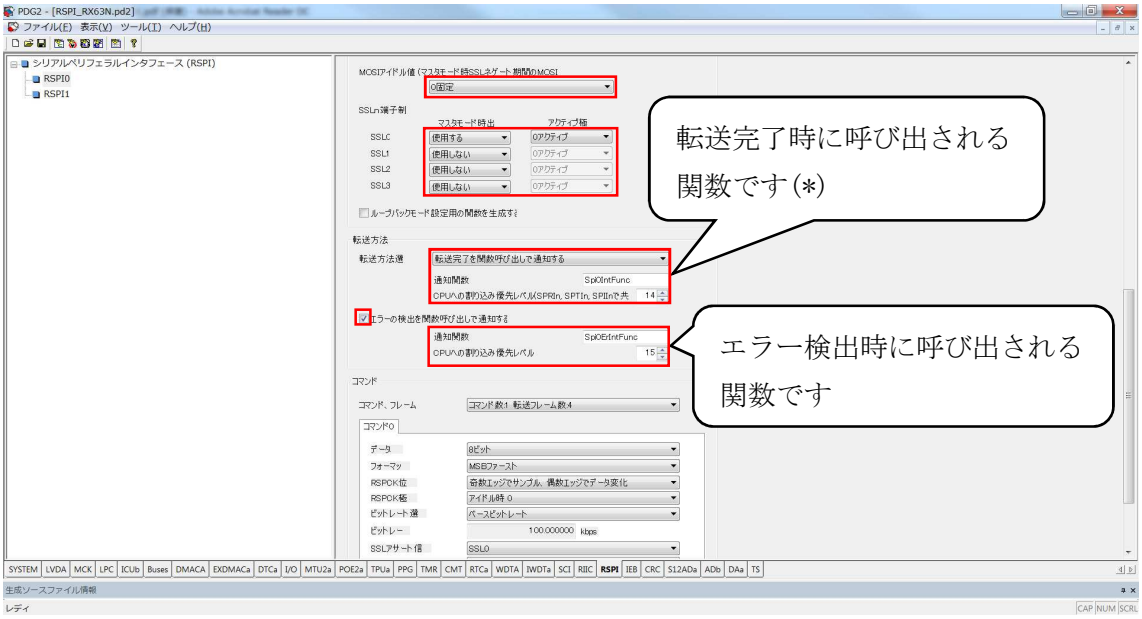




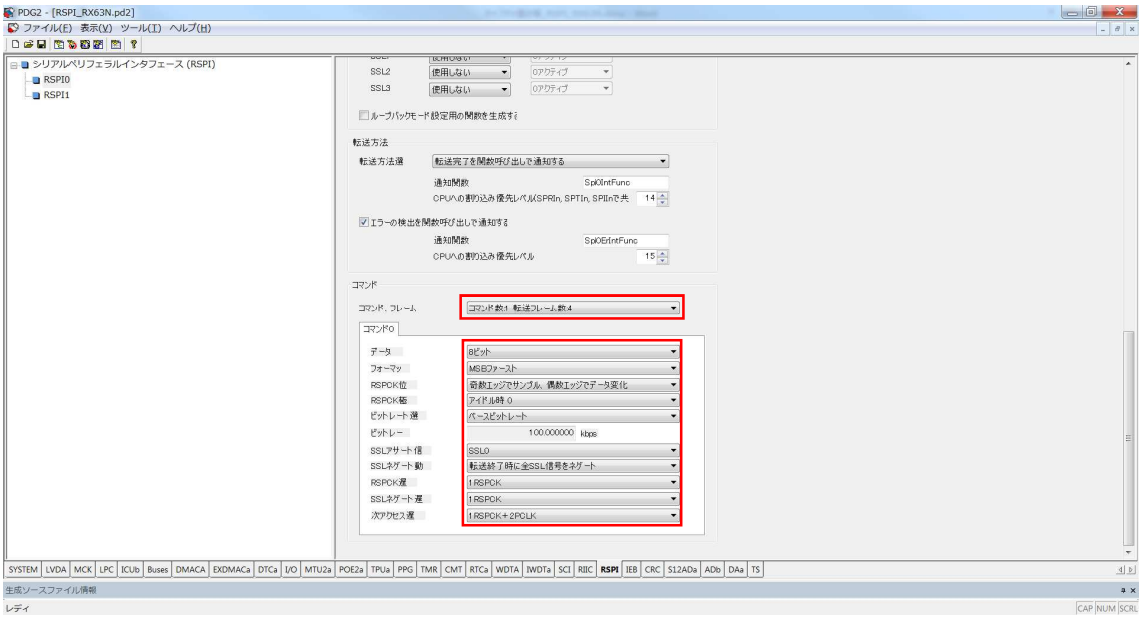
5.2　　RSPI0 設定

RSPI0 の設定を以下に示します。



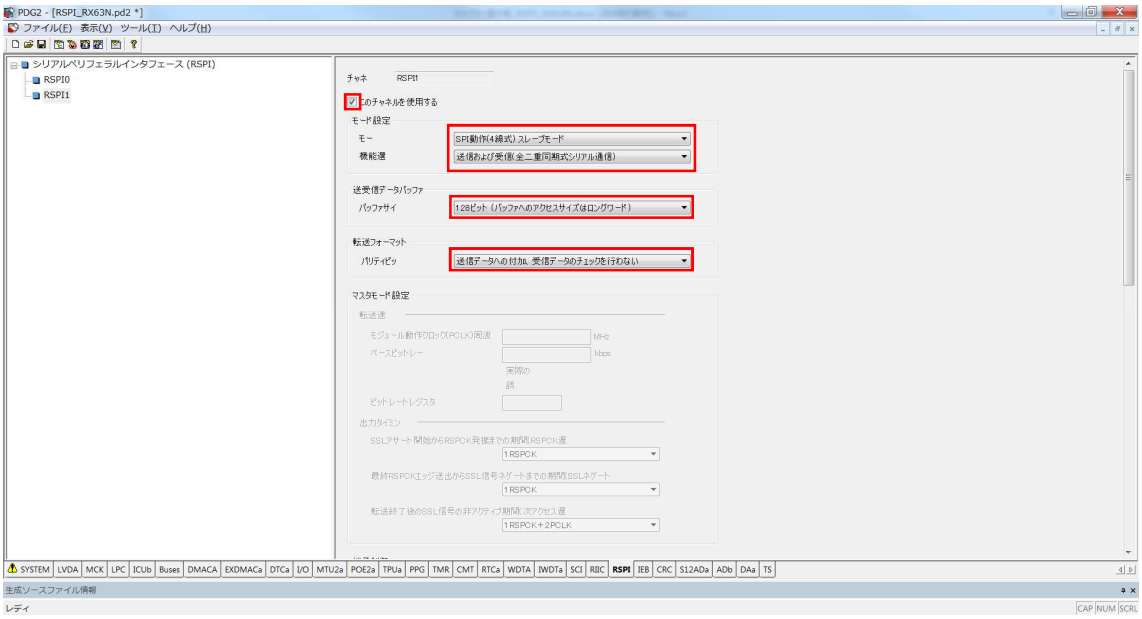


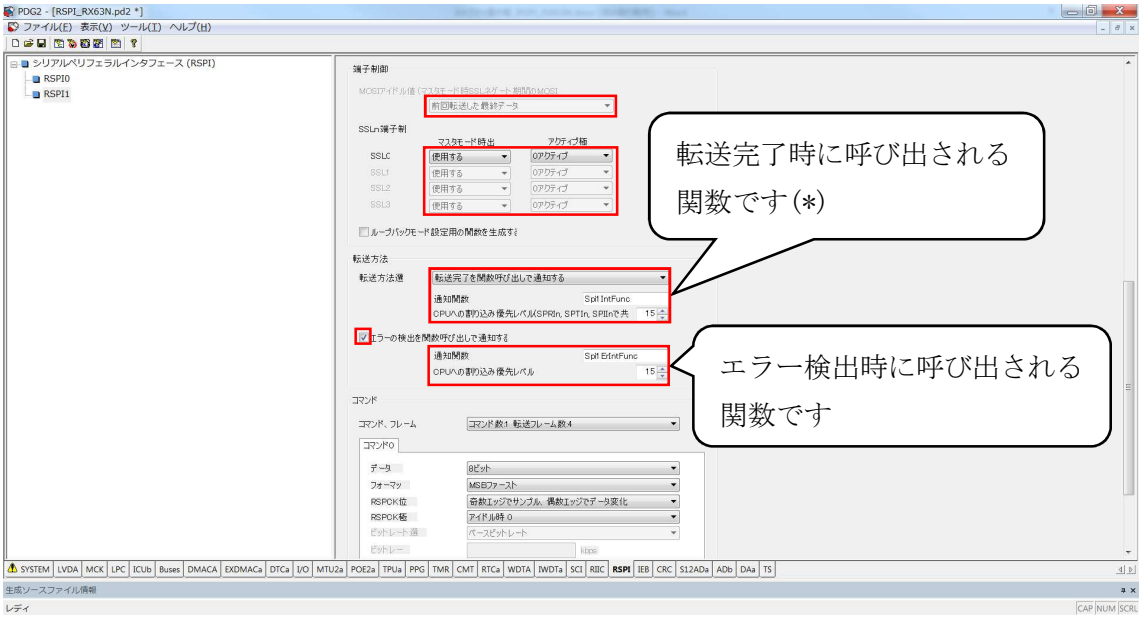
(*):マスタが送受信を開始する時には、スレーブは送受信の準備ができていないといけないため、マスタの優先レベル ＜ スレーブの優先レベルとしています。


PDG2 - [RSPI_RX63N.pd2]
ファイル(F) 表示(V) ツール(T) ヘルプ(H)
シリアルペリフェラルインタフェース (RSPI)
RSPI0
RSPI1
SSL2
SSL3
使用しない
0アクティブ
ループバックモード設定用の関数を生成する
転送方法
転送方法選
転送完了を関数呼び出しで通知する
通知関数
RspiIntFunc
CPUへの割り込み優先レベル(SPRIn, SPTIn, SPIInで共
14
エラーの検出を関数呼び出しで通知する
RspiErIntFunc
15
コマンド
コマンド, フレーム
コマンド数:1 転送フレーム数:4
コマンド0
データ
8ビット
フォーマット
MSBファースト
RSPCK位
奇数エッジでサンプル、偶数エッジでデータ変化
RSPCK極
アイドル時 0
ビットレート選
ベースビットレート
ビットレート
100.000000 kbps
SSLアサート信
SSL0
SSLネゲート動
転送終了時に全SSL信号をネゲート
RSPCK遅
1 RSPCK
SSLネゲート遅
1 RSPCK
次アクセス遅
1 RSPCK+2PCLK
SYSTEM LVDA MCK LPC ICUb Buses DMACA EXDMACa DTCa I/O MTU2a POE2a TPUa PPG TMR CMT RTCa WDTA IWDTa SCI RIIC RSPI IEB CRC S12ADa ADb DAa TS
生成ソースファイル情報
レディ
CAP NUM SCRL

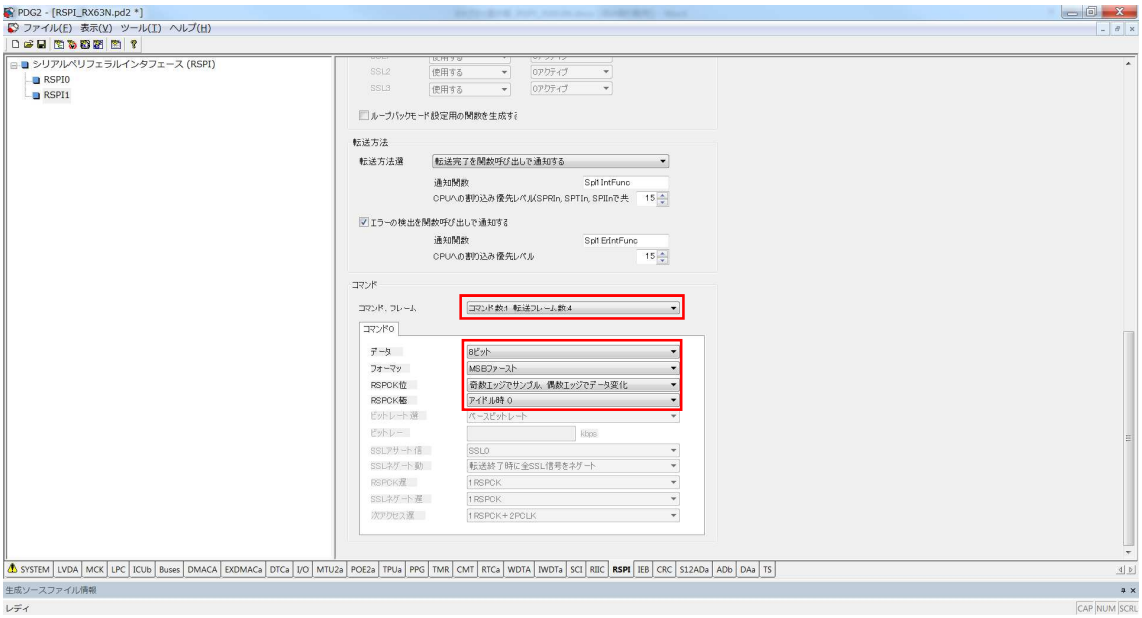
5.3　　RSPI1 設定

RSPI1 の設定を以下に示します。



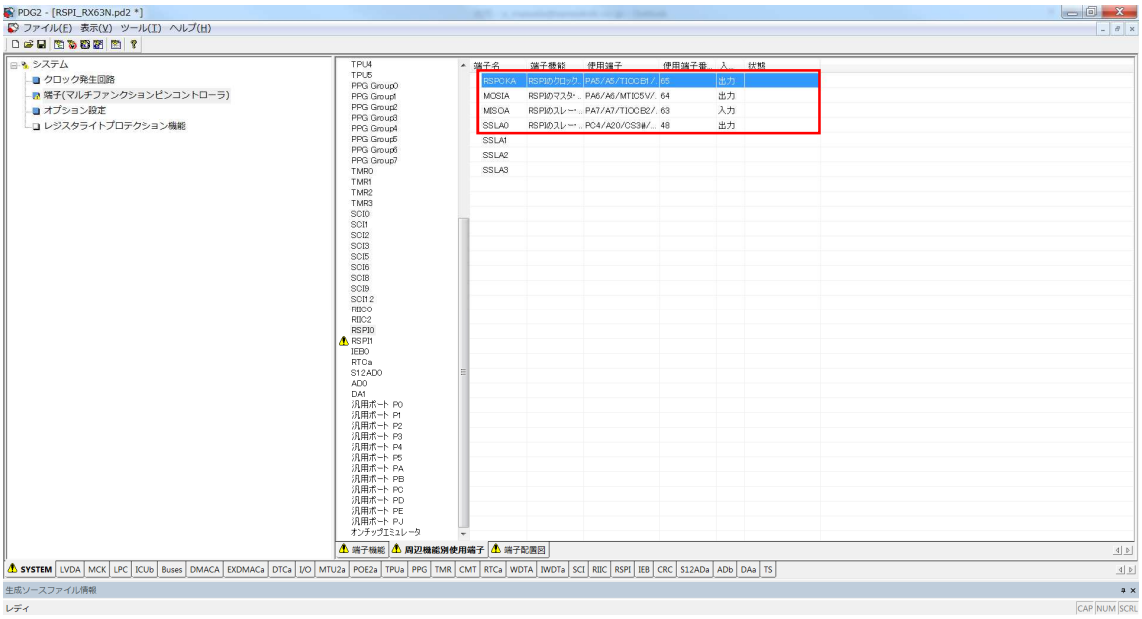


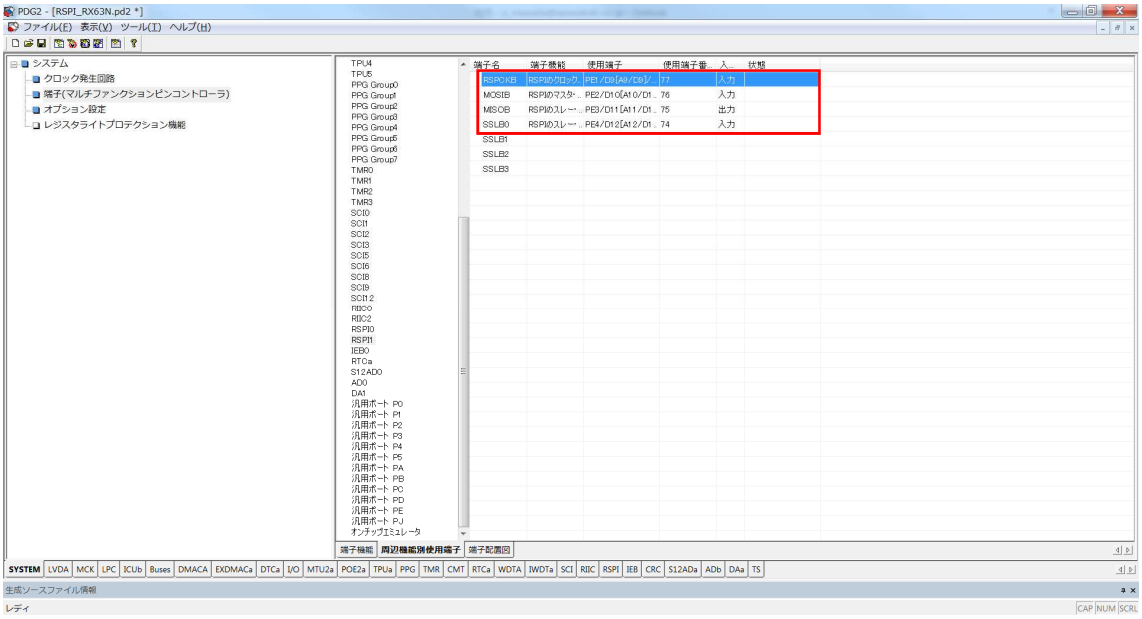
(*):マスタが送受信を開始する時には、スレーブは送受信の準備ができていないといけないため、マスタの優先レベル ＜ スレーブの優先レベルとしています。


PDG2 - [RSPI_RX63N.pd2 *]
ファイル(F) 表示(V) ツール(T) ヘルプ(H)
シリアルペリフェラルインタフェース (RSPI)
RSPI0
RSPI1
SSL2
使用する
0アクティブ
SSL3
使用する
0アクティブ
ループバックモード設定用の関数を生成する
転送方法
転送方法選
転送完了を関数呼び出しで通知する
通知関数
Spi1 IntFunc
CPUへの割り込み優先レベル(SPRIn, SPTIn, SPIInで共
15
エラーの検出を関数呼び出しで通知する
通知関数
Spi1 ErIntFunc
CPUへの割り込み優先レベル
15
コマンド
コマンド, フレーム
コマンド数:1 転送フレーム数:4
コマンド0
データ
8ビット
フォーマッ
MSBファースト
RSPCK位
奇数エッジでサンプル、偶数エッジでデータ変化
RSPCK極
アイドル時 0
ビットレート選
ベースビットレート
ビットレー
kbps
SSLアサート信
SSL0
SSLネゲート動
転送終了時に全SSL信号をネゲート
RSPCK遅
1 RSPCK
SSLネゲート遅
1 RSPCK
次アクセス遅
1 RSPCK + 2PCLK
SYSTEM LVDA MCK LPC ICUb Buses DMACA EXDMACa DTCa I/O MTU2a POE2a TPUa PPG TMR CMT RTCa WDTA IWDTa SCI RIIC RSPI IEB CRC S12ADa ADb DAa TS
生成ソースファイル情報
レディ
CAP NUM SCRL

5.4　　SYSTEM の端子設定

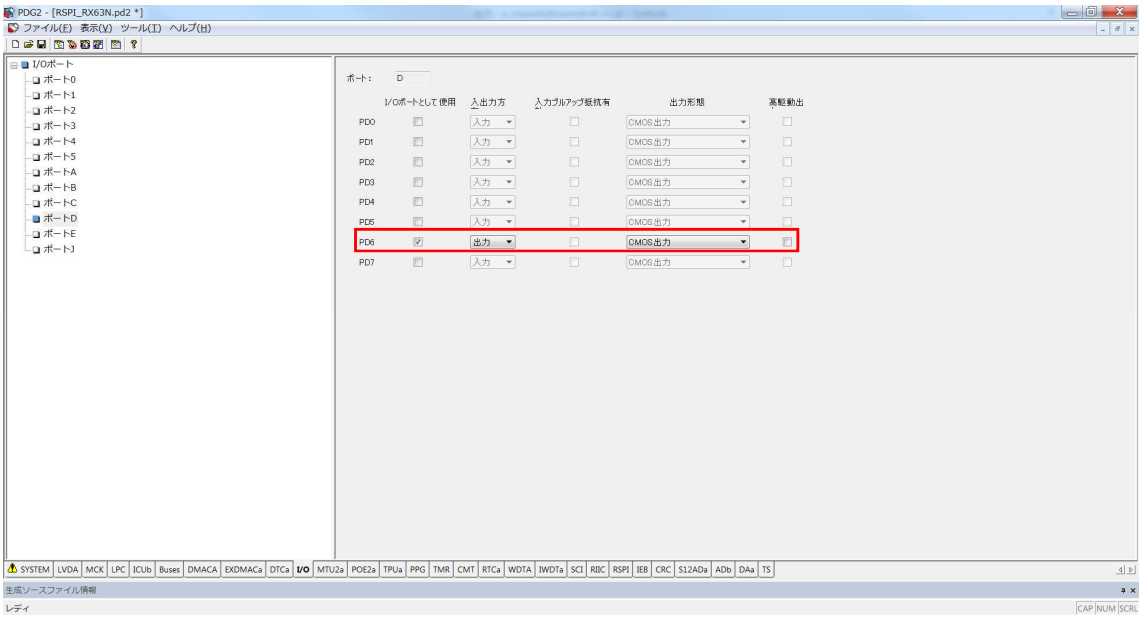
SYSTEM の端子設定を確認します。



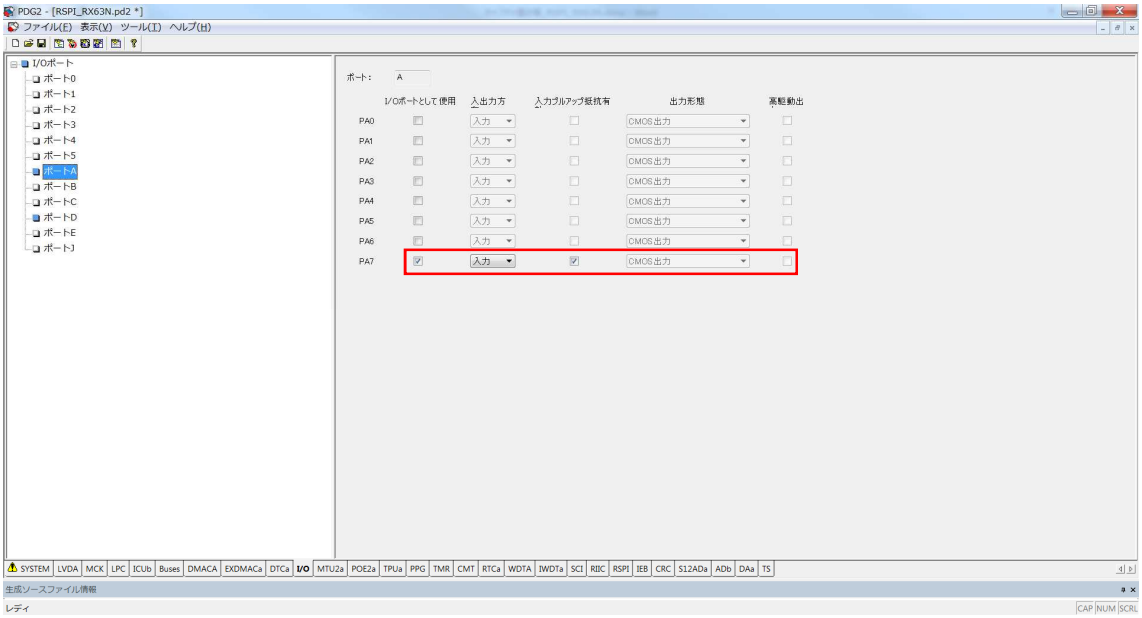


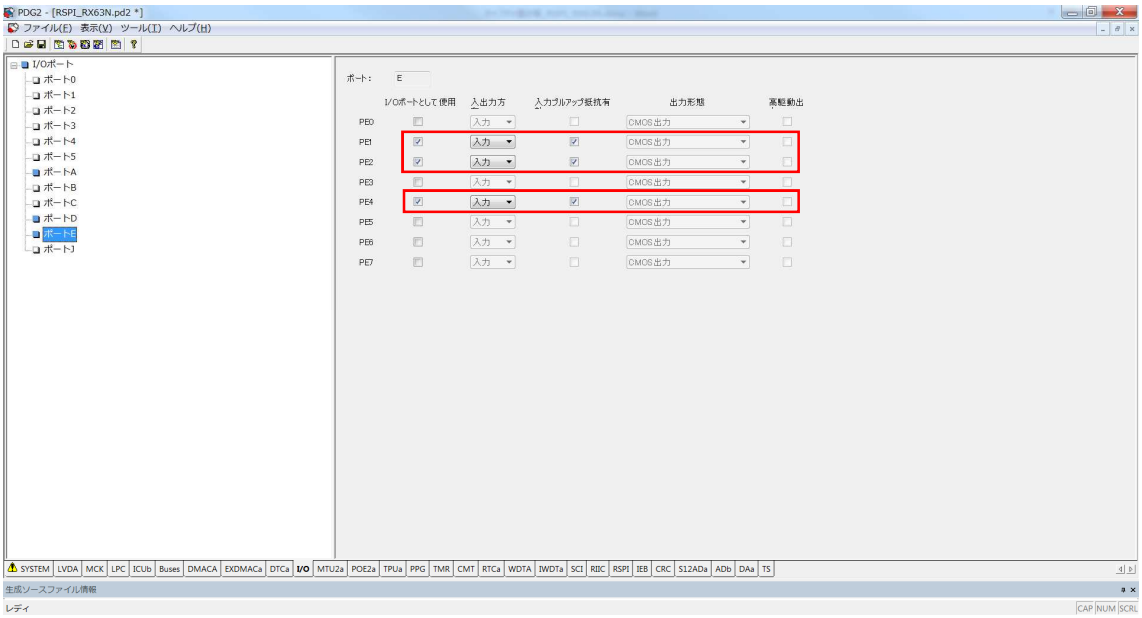
5.5　　I/O 設定

I/O の設定を以下に示します。LED1 で使用する PD6 を出力にします。

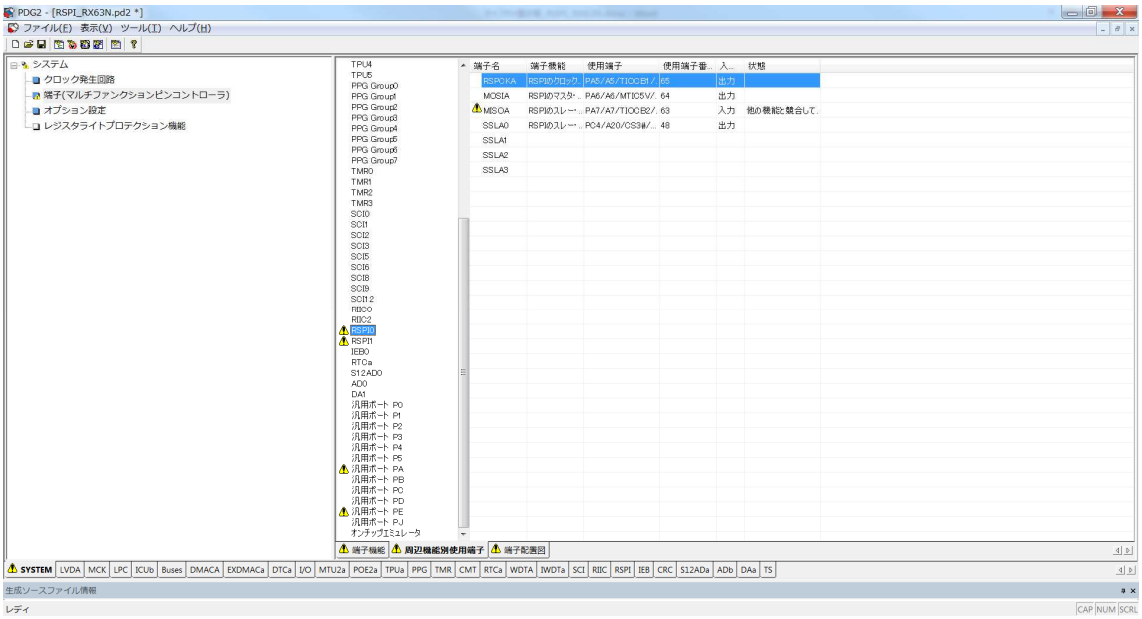


RSPI0、1 の入力端子に内蔵プルアップを設定します。



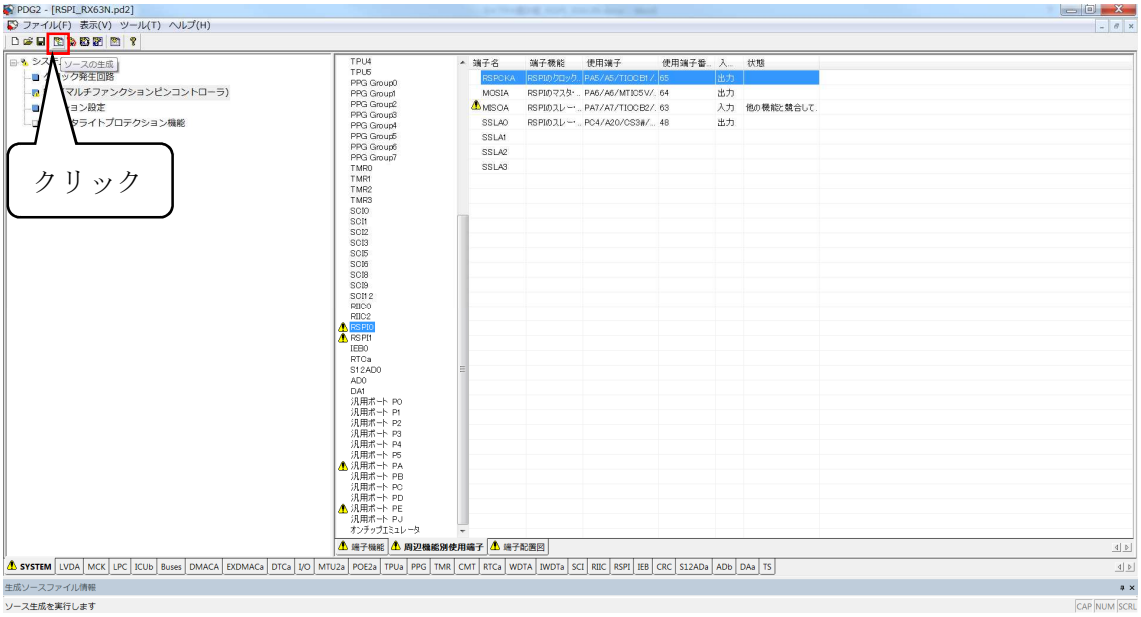


RSPI0、1 の入力端子に内蔵プルアップを設定すると、SYSTEM タブの端子で以下のように汎用入力ポートと機能が競合している旨の注意が表示されましたが、ハードウェアマニュアルで各対象ポートのブロック図を参照し、内蔵プルアップが有効で問題なさそうでした（動作確認済みです）。

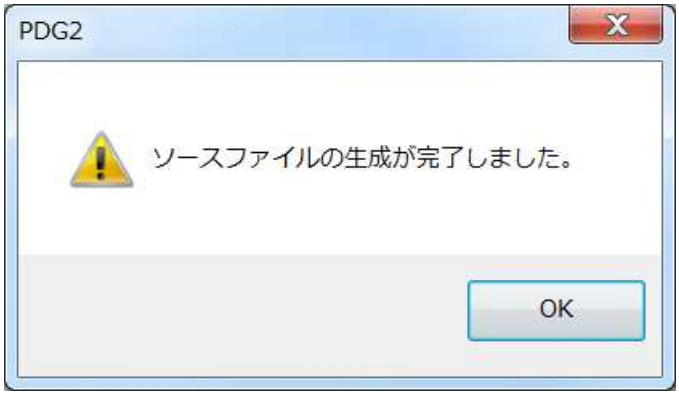


5.6　　ソースの生成

以下の GUI をクリックすると、

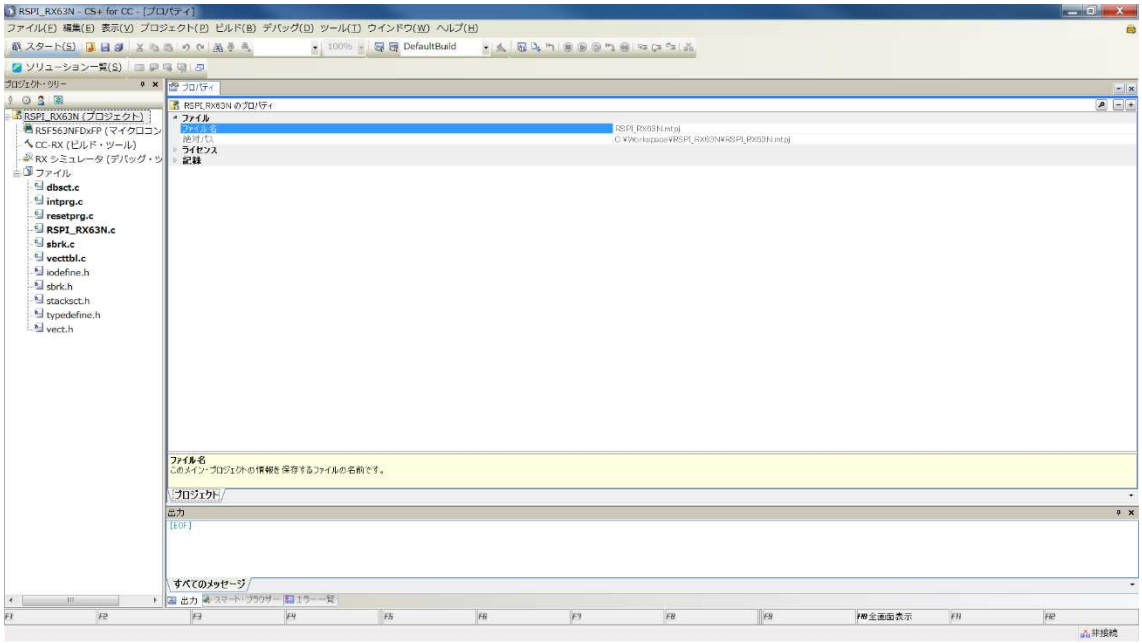


ソースファイルが生成されます。

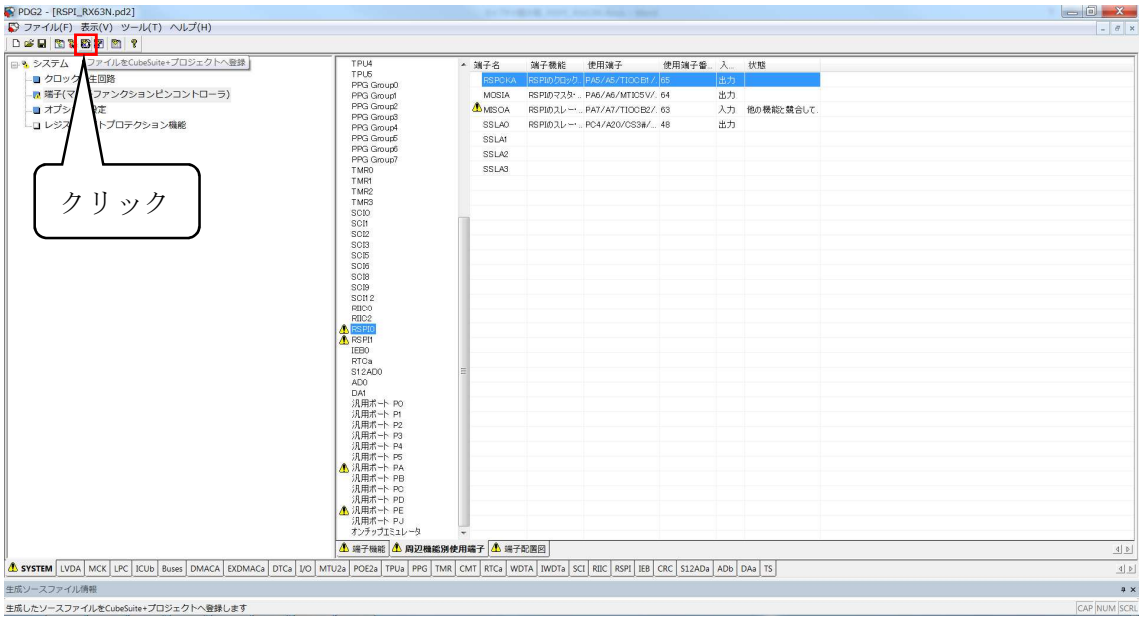


5.7　　CS+への登録

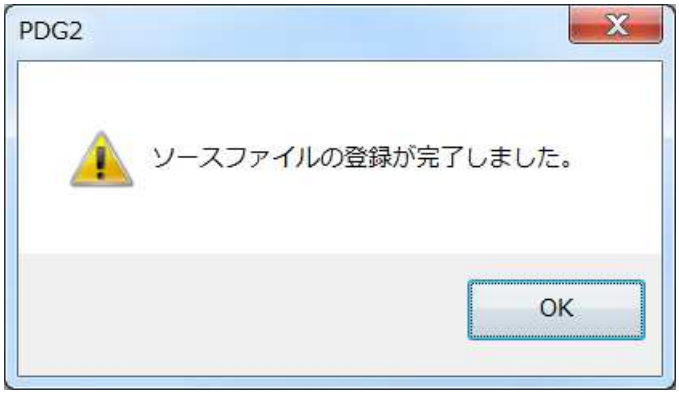
対象の CS+プロジェクトを開きます。



以下の GUI をクリックします

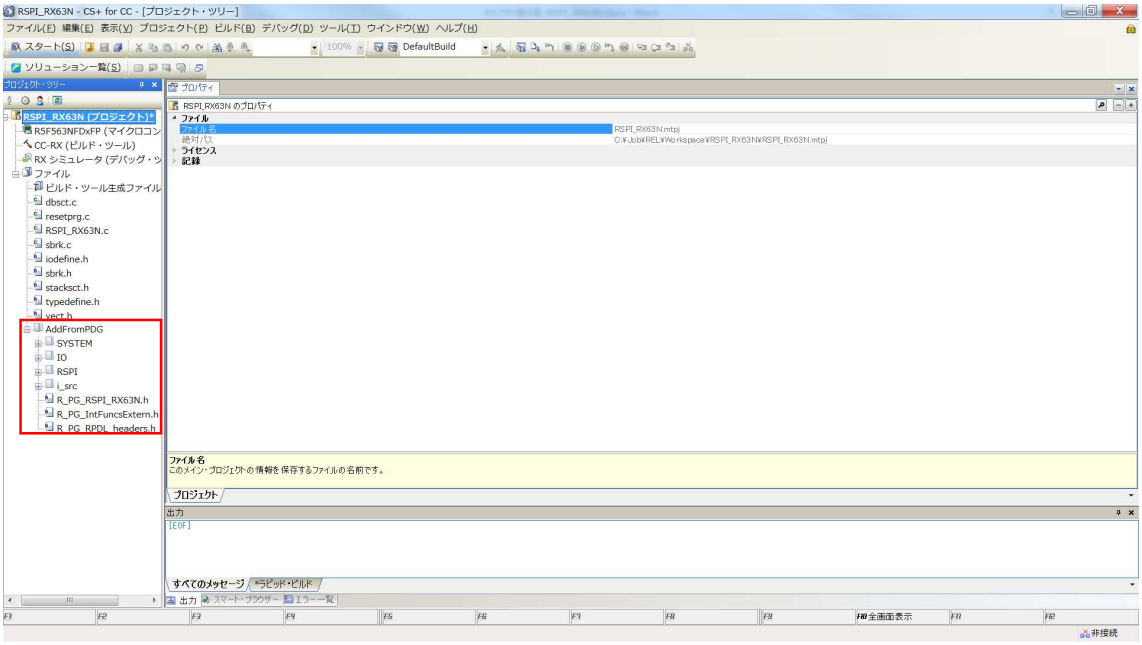


ソースファイルの登録が完了しました。



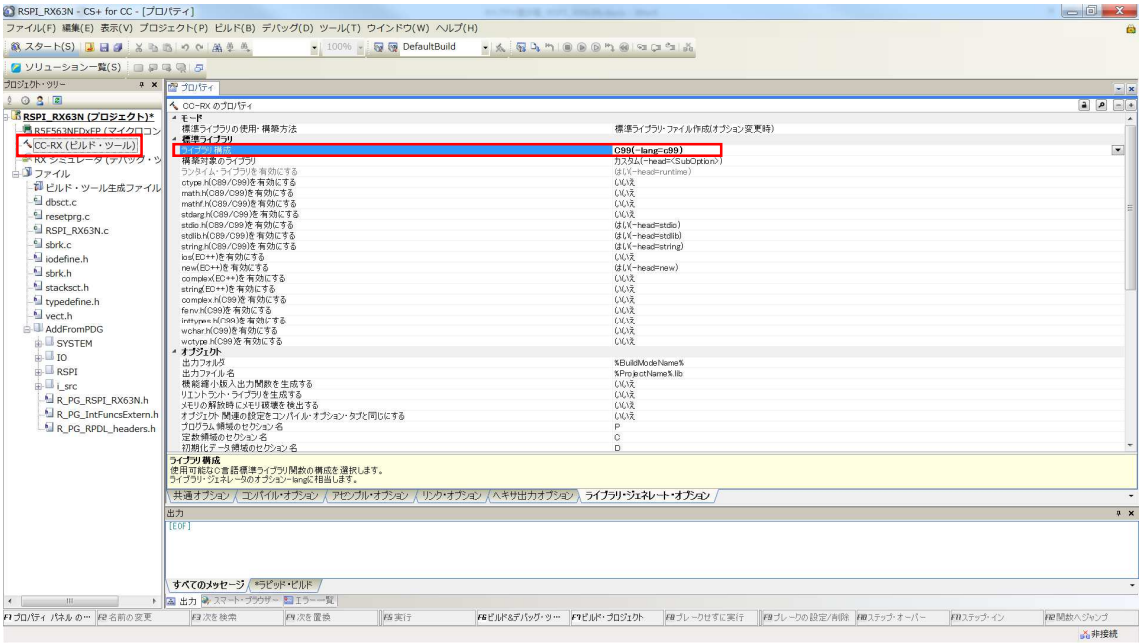
6.　CS+のプロジェクトに PDG のソースファイルを登録する際の設定

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。

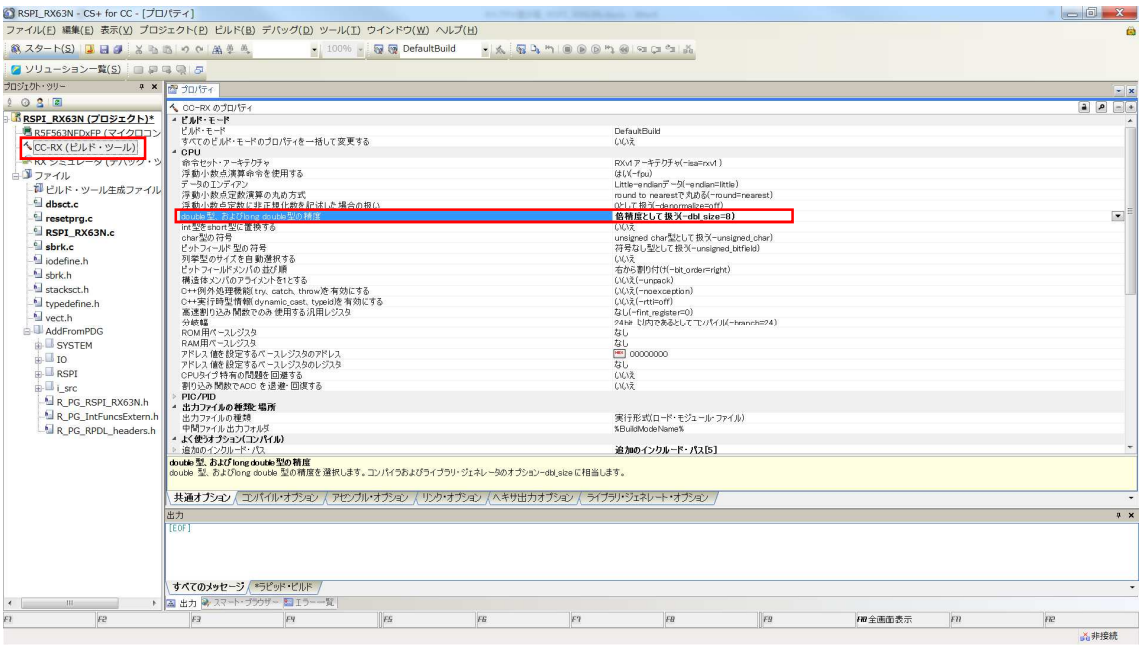


そのままビルドをすると、エラーおよび警告が発生します。解消する設定を以下に示します。

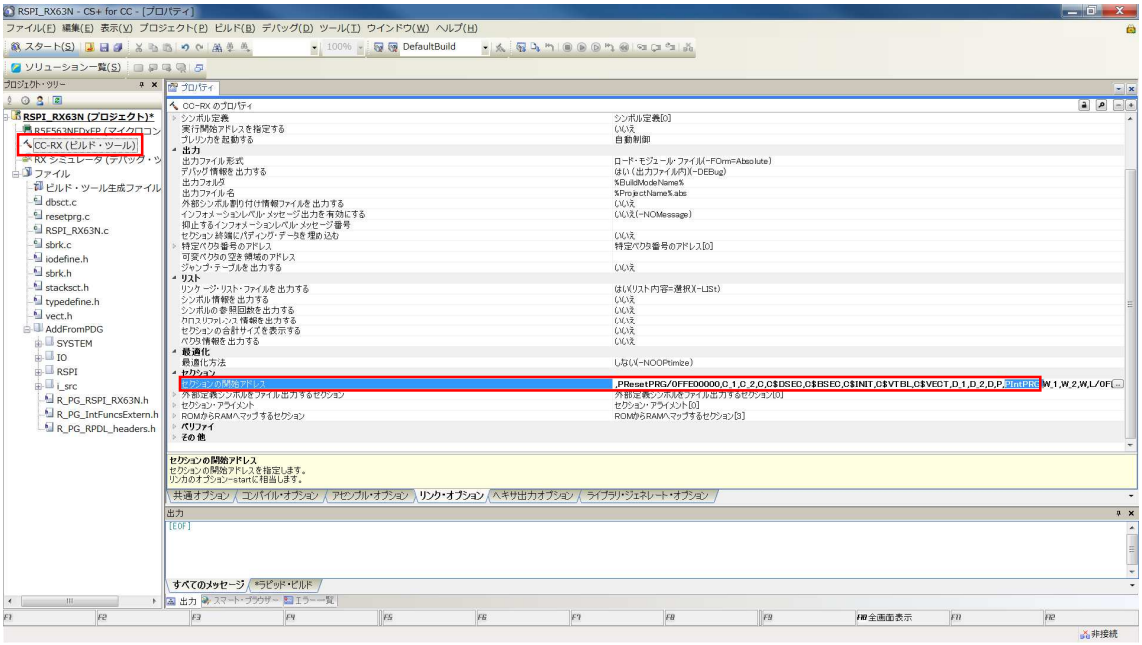
PDG で生成されるソースファイルは bool 変数を使用しています。対応させるため、ビルド・ツールを右クリック->プロパティを表示し、ライブラリ・ジェネレート・オプションタブにある「ライブラリ構成」を”C99(-lang=c99)”に設定します。



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。

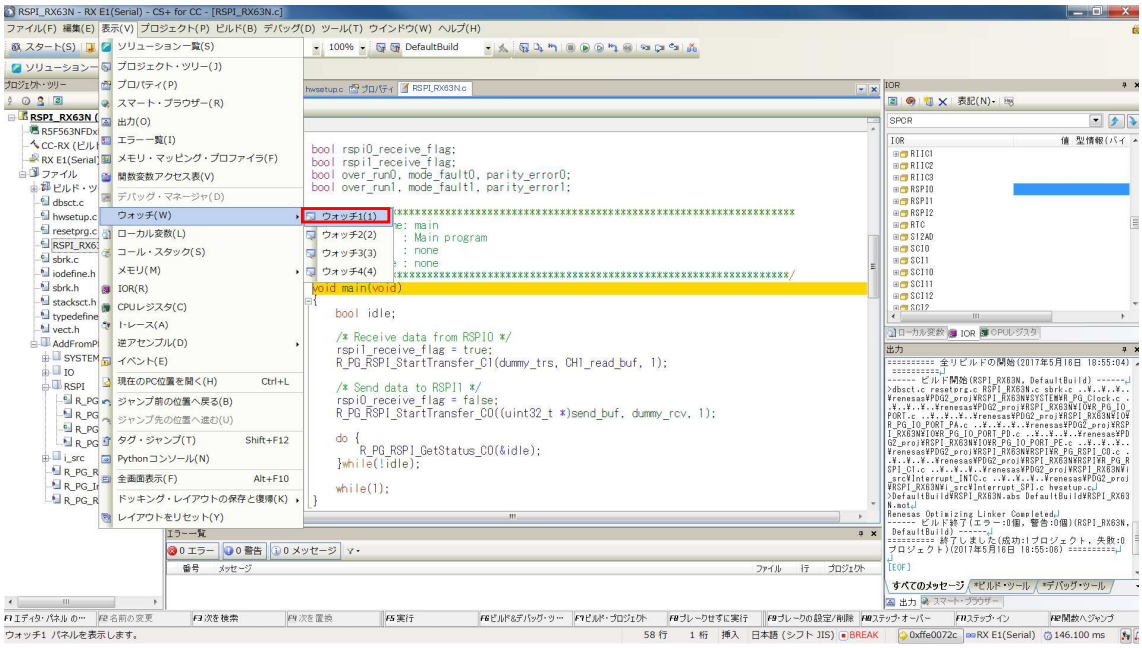


7.　動作確認方法

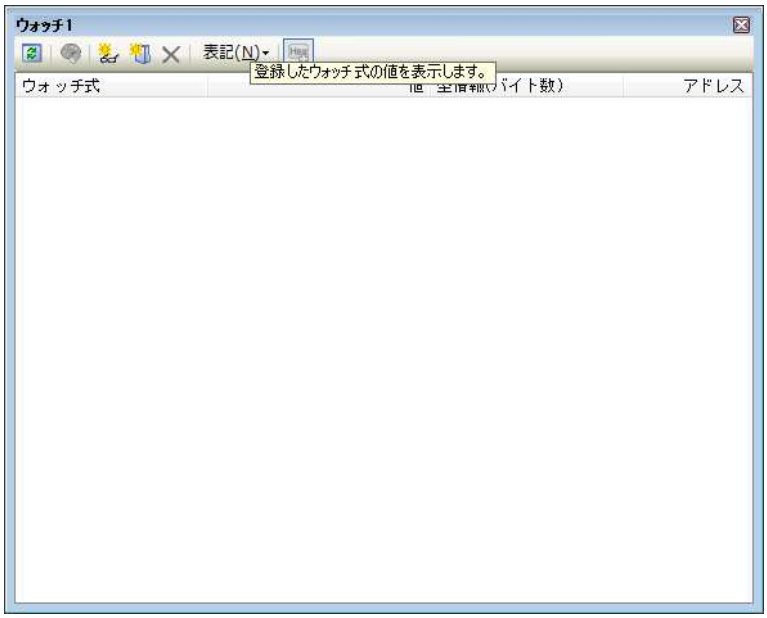
7.1　　ウォッチ式の登録

RSPI0 から RSPI1 にデータが送信され、RSPI1 の受信データを RSPI0 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、LED1 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。

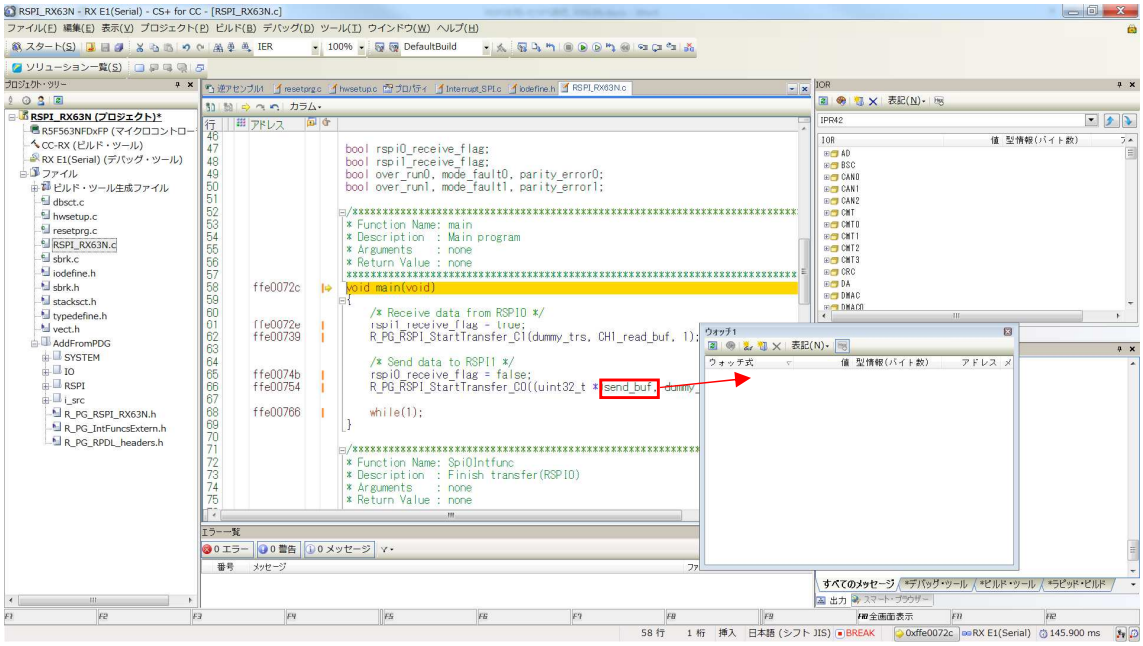
表示タブからウォッチ、そしてウォッチ 1 を選択します。



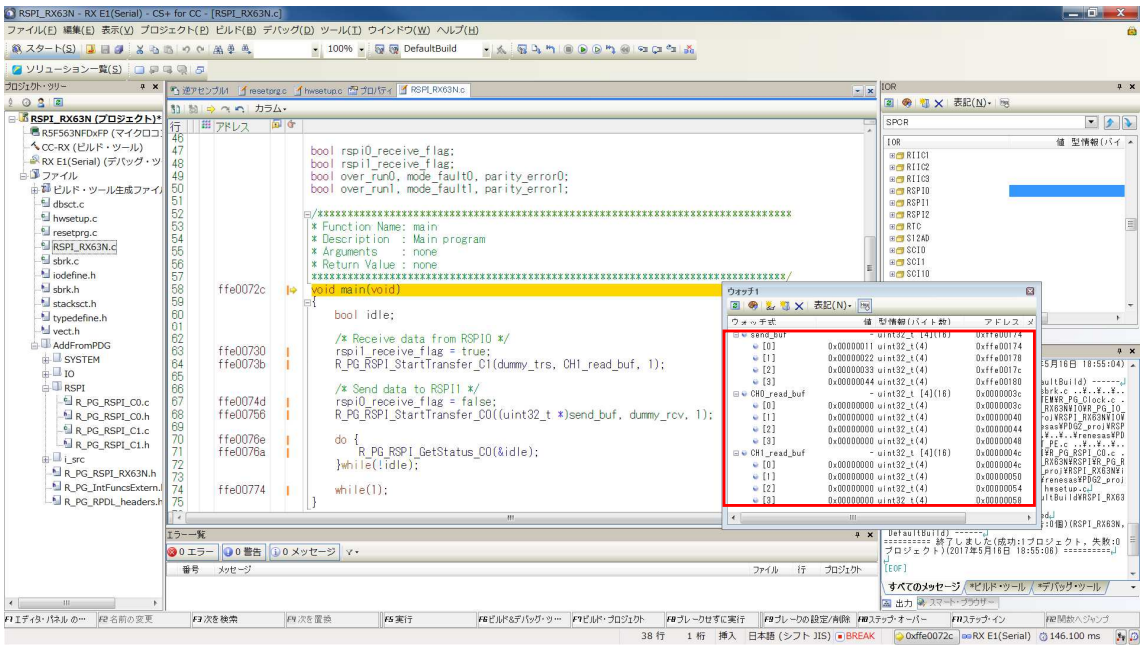
ウォッチ(ウォッチ 1)パネルが表示されました。



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“CH1_read_buf”も同様にドラッグ&ドロップします。

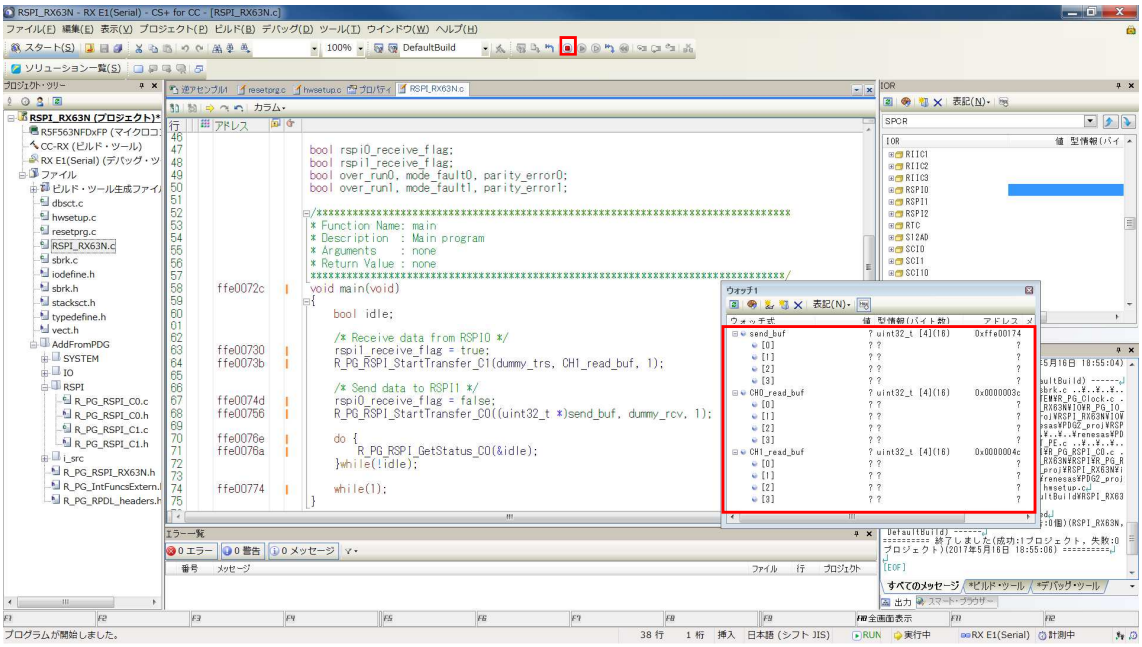


上の図のようにウォッチパネルには“send_buf”にのみ 00000011h, 00000022h, 00000033h, 00000044h が表示され、“CH0_read_buf”と“CH1_read_buf”は全て 00000000h であることが確認できます。ボード側は、LED1 は消灯となっています。この状態で実行前の準備は完了です。

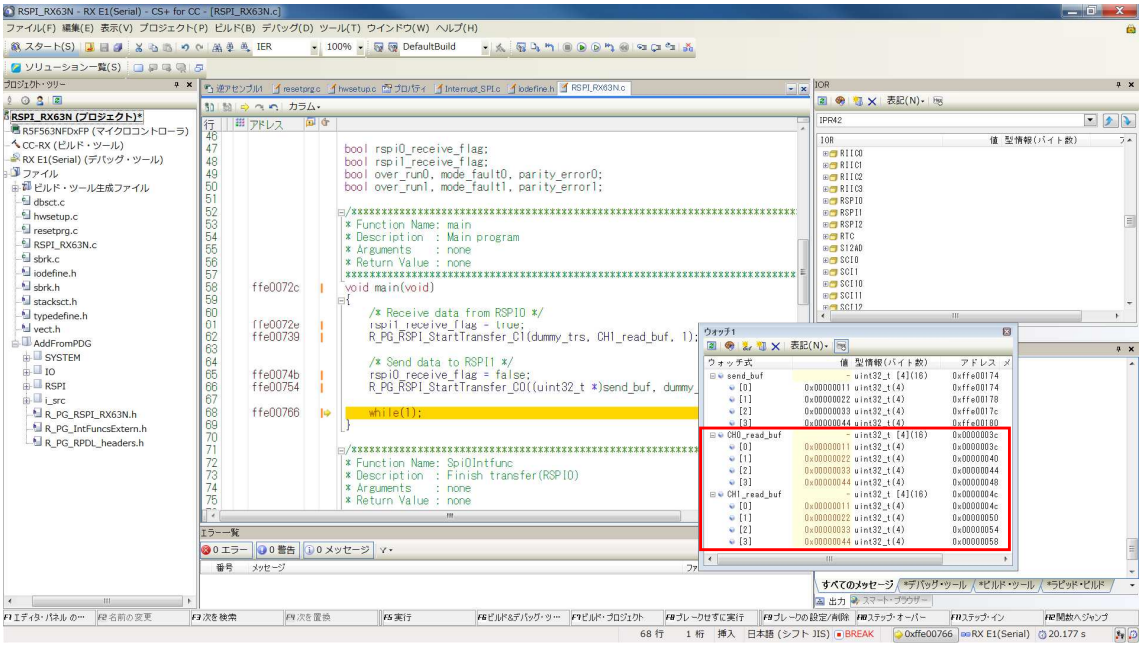
7.2　　実行

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“CH1_read_buf”は共に 00000011h, 00000022h, 00000033h, 00000044h が表示されています。これで RSPI1 は RSPI0 から 00000011h, 00000022h, 00000033h, 00000044h を受信できており、RSPI0 は RSPI1 から送り返された 00000011h, 00000022h, 00000033h, 00000044h が受信できたことが確認できました。



8. 参考ドキュメント

RX63N グループ、RX631 グループ ユーザーズマニュアルハードウェア編

RX63N グループ、RX631 グループ Peripheral Driver Generator リファレンスマニュアル

以上