

RX210 グループ

DMAC を用いた調歩同期式通信

要旨

本サンプルコードでは、DMA コントローラ (DMAC) を用いた調歩同期式通信の方法について説明します。

対象デバイス

- RX210

内容

1.	仕様	4
2.	動作確認条件	5
3.	ハードウェア説明.....	6
3.1	使用端子一覧.....	6
3.2	接続信号	6
4.	ソフトウェア説明.....	7
4.1	動作概要	7
4.1.1	送信動作.....	7
4.1.2	受信動作.....	9
4.2	ファイル構成.....	11
4.3	オプション設定メモリ.....	12
4.4	定数一覧	12
4.5	変数一覧	13
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.....	17
4.8.1	初期設定.....	17
4.8.2	メイン処理.....	18
4.8.3	TDR0 へ送信データ転送終了処理	18
4.8.4	SCI0 受信完了処理	19
4.8.5	TDR2 へ送信データ転送終了処理	20
4.8.6	SCI2 受信完了処理	21
4.8.7	SCI0 受信エラー処理	21
4.8.8	SCI5 受信エラー処理	22
5.	PDG の設定	23
5.1	SYSTEM 設定	25
5.2	SCI0 設定	26
5.3	SCI5 設定	28
5.4	DMAC の設定	29
5.5	SYSTEM の端子設定	33
5.6	I/O 設定	34
5.7	ソースの生成.....	35
5.8	CS+への登録.....	36
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.....	38

7.	動作確認方法	41
7.1	ウォッチ式の登録.....	41
7.2	実行	43
8.	参考ドキュメント.....	49

1. 仕様

SCI0 から SCI5 へ調歩同期式通信を用い、送信データエンプティ割り込み要求(TXI0)でDMAC0を起動し、データを送信します。SCI5は受信データフル割り込み要求(RXI5)でDMAC3を起動し、受信したデータをRAMへ格納します。次にTXI5でDMAC2を起動し、SCI0から受信したデータを送り返します。SCI0はRXI0でDMAC1を起動し、受信したデータをRAMへ格納します。SCI0がSCI5へ送信したデータとSCI5がSCI0へ送り返したデータが一致した場合、LED(D3)が点灯します。図 1.1 にDMACを用いた調歩同期式データ通信の概要を示します。

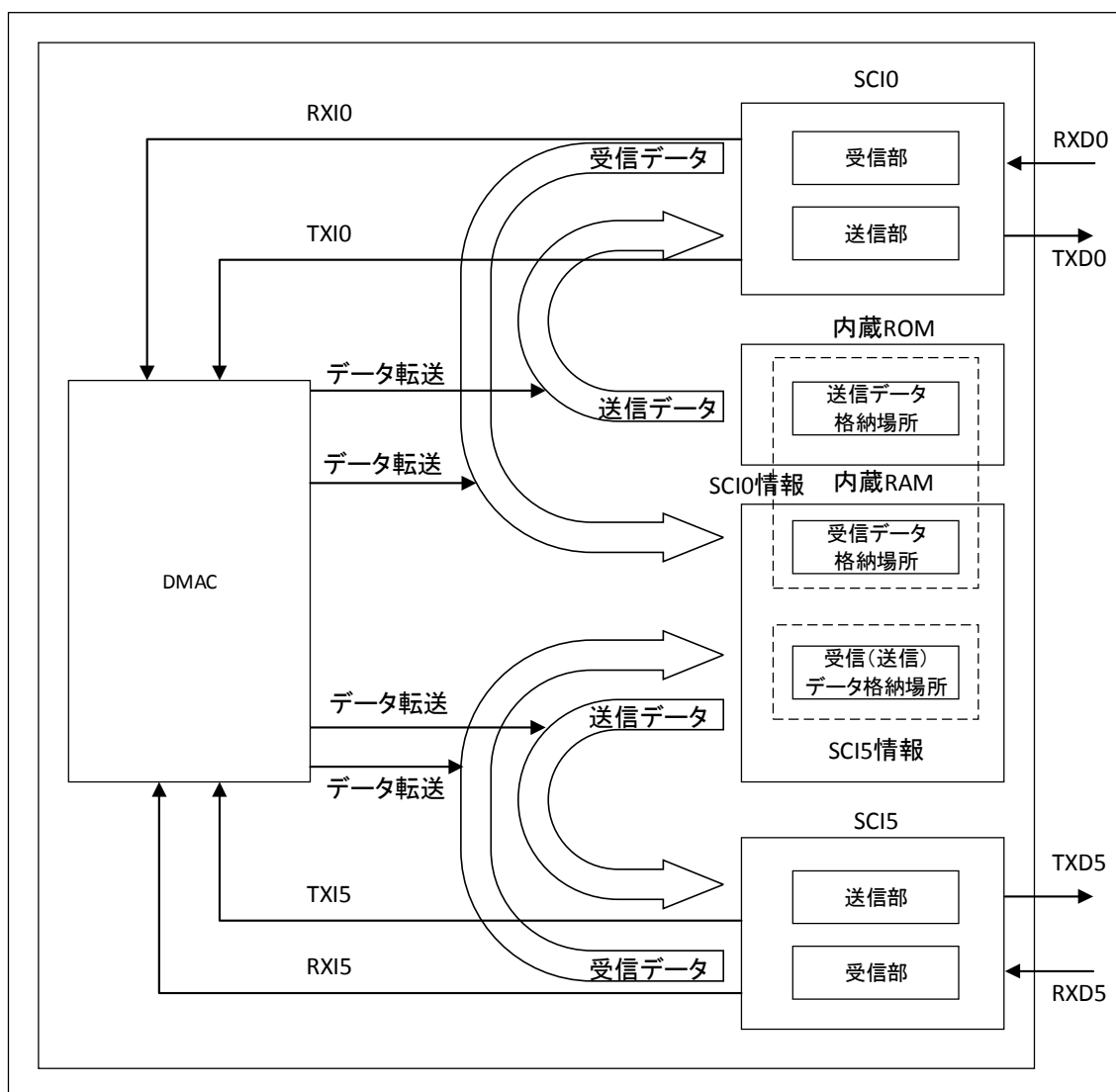


図 1.1 DMAC を用いた調歩同期式データ通信の概要

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 (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
P20	出力	TXD0
P21	入力	RXD0
PA4	出力	TXD5
PA2	入力	RXD5
PH2	出力	D3(LED)

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

	送信	受信
接続 1	J1 の 34pin (P20/TXD0)	J2 の 6pin (PA2/RXD5)
接続 2	J2 の 4pin (PA4/TXD5)	J1 の 33pin (P21/RXD0)

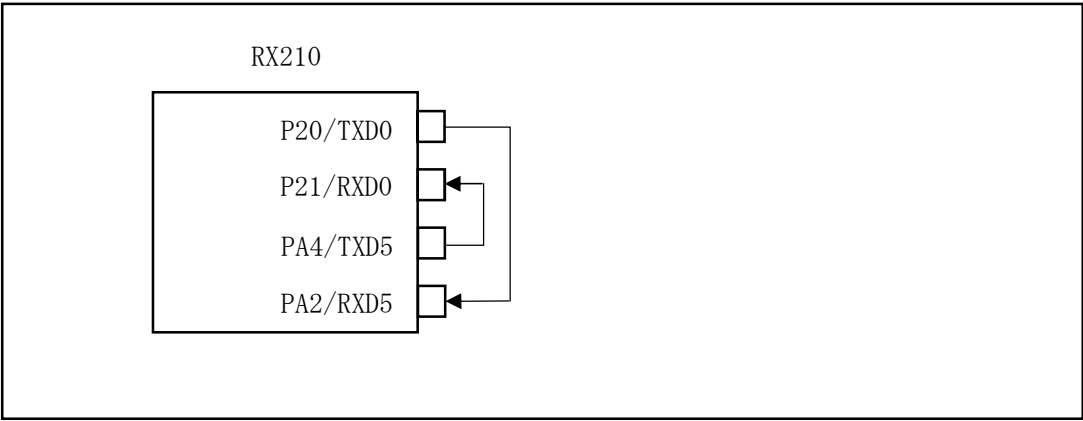


図 3.1 信号接続図

4. ソフトウェア説明

4.1 動作概要

4.1.1 送信動作

図 4.1 に SCI0 の送信動作タイミング図を示した上でそれぞれ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) 初期設定

SCI0 から SCI5 に送信するために、以下のように初期設定を行います。

DMAC0(TXI0)の設定概要は次の通りです。詳細は「5.4 DMAC の設定」を参照ください。

DMAC0 は(TXI0)、DMAC2 は(TXI5)を設定しています。

- ・ ノーマル転送モード
- ・ 転送回数 13 回
- ・ 転送データのビット長は 1 バイト
- ・ 転送元アドレスは内蔵 RAM でアドレス更新モードはインクリメント
- ・ 転送先アドレスは SCI0 トランスミットデータレジスタでアドレス更新モードは固定
- ・ SCI0 送信データエンプティ割り込みで起動

SCI0 の設定概要は次の通りです。詳細は「5.2 SCI0 設定」を参照ください。

- ・ 調歩同期式モード
- ・ データ長を 8 ビットとし、ストップビットは 1 ビット、LSB ファーストとする
- ・ ボーレートは 57600bps とする

(2) 送信動作及び TXI 割り込み要求を許可

TE ビットを 1 にすることによりシリアル送信動作が、TIE ビットを 1 にすることにより TXI 割り込み要求が許可されます。TXI 割り込み要求により DMAC が起動し、データ 1 が TDR に転送されます。

(3) データ 1 を TSR に転送

TDR からデータ 1 を TSR に転送します。データは直ちに TXD 端子より送信されます。同時に TXI 割り込み要求により DMAC が起動し、データ 2 が TDR に転送されます。

(4) データラストを TDR に転送

DMAC が起動し、データラストを TDR に転送します。指定回数の転送が完了したため、CPU へ割り込みが発生し、DMAC 割り込み処理 (*1) を実行します。

(*1) : PDG の DMAC 項目にある割り込み通知関数 (割り込み要因は転送終了)

(5) データラストを TSR に転送

TDR にあるデータラストを TSR に転送します。データは直ちに TXD0 端子より送信されます。

(6) データラストの送信完了

TXD0 端子から送信が完了します。

※本サンプルコードでは使用していませんが、TEI を使用する場合は、このタイミングで SCI の送信完了を確認できます。

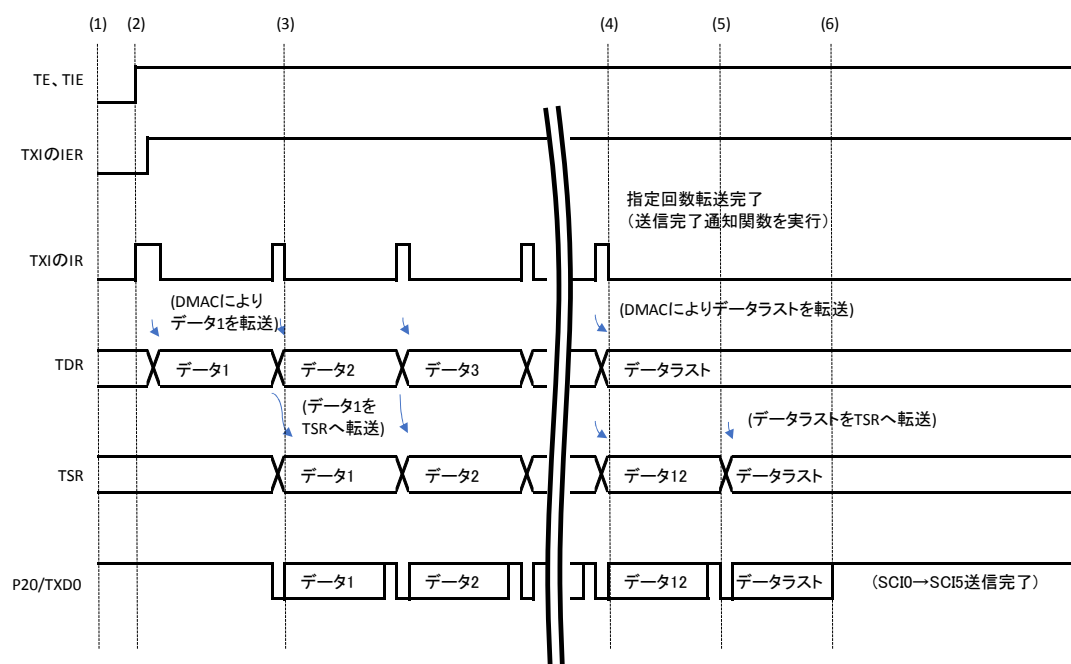


図 4.1 送信動作タイミング図

4.1.2 受信動作

図 4.2 に SCI5 の受信動作タイミング図を示した上でそれぞれ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) 初期設定

SCI0 からのデータを SCI5 が受信するために、以下のように初期設定を行います。

DMAC(RXI5)の設定概要は次の通りです。詳細は「5.4 DMAC の設定」を参照ください。

DMAC1 は(RXI0)、DMAC3 は(RXI5)を設定しています。

- ・ ノーマル転送モード
- ・ 転送回数 13 回
- ・ 転送データのビット長は 1 バイト
- ・ 転送元アドレス更新モードは SCI5 レシーブデータレジスタで固定
- ・ 転送先アドレス更新モードは内蔵 RAM でインクリメント
- ・ SCI5 受信データフル割り込みで起動

SCI5 の設定概要は次の通りです。詳細は「5.3 SCI5 設定」を参照ください。

- ・ 調歩同期式モード
- ・ データ長を 8 ビットとし、ストップビットは 1 ビット、LSB ファーストとする
- ・ ボーレートは 57600bps とする

(2) 受信動作及び RXI 割り込み要求を許可

RIE ビットを 1 にすることにより RXI 割り込み要求が、RE ビットを 1 にすることによりシリアル受信動作が許可されます。

(3) データ 1 受信完了

データ 1 の受信が完了すると、RXI 割り込み要求により DMAC が起動され、受信されたデータ 1 は RDR から受信バッファ (RAM) へ転送されます。

(4) データラスト受信完了

データラストの受信が完了すると、RXI 割り込み要求により DMAC が起動され、受信されたデータは RDR から受信バッファ (RAM) へ転送されます。指定回数の転送が完了したため、CPU へ割り込みが発生し、DMAC 割り込み処理 (*2) を実行します。

(*2): PDG の DMAC 項目にある割り込み通知関数 (割り込み要因は転送終了) 受信完了通知関数はこのタイミングで呼ばれます。

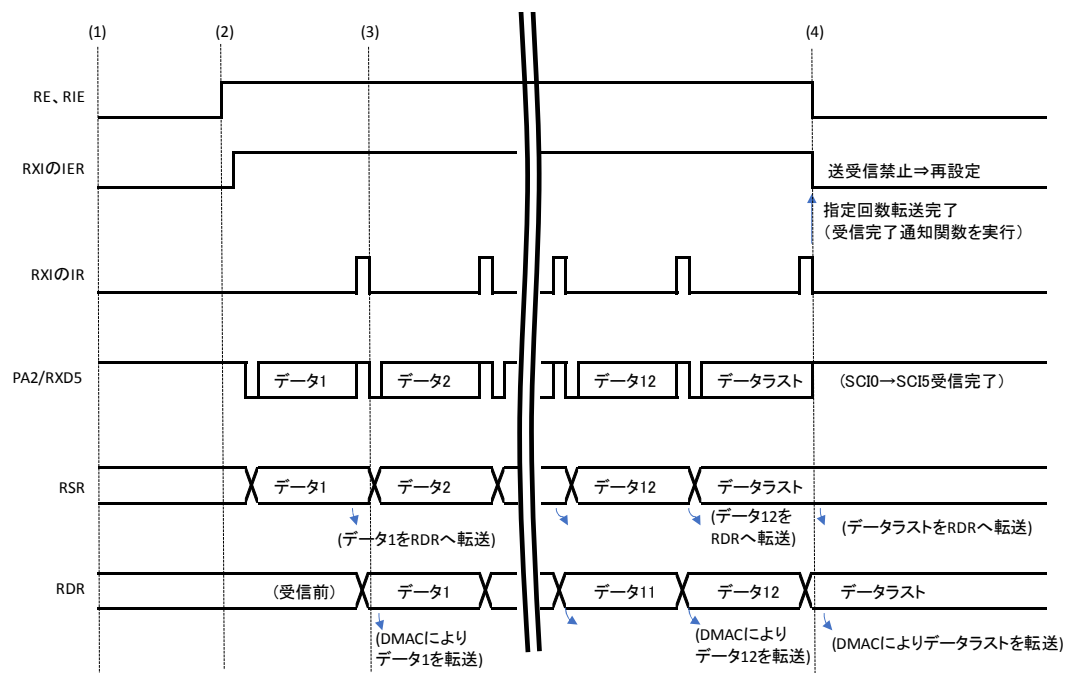


図 4.2 受信動作タイミング図

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
DMAC_RX210.c	メインファイル <ul style="list-style-type: none"> • SCIO 送信/受信処理 • SCIO5 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 • DMAC の設定 • SCIO の設定 • SCIO5 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

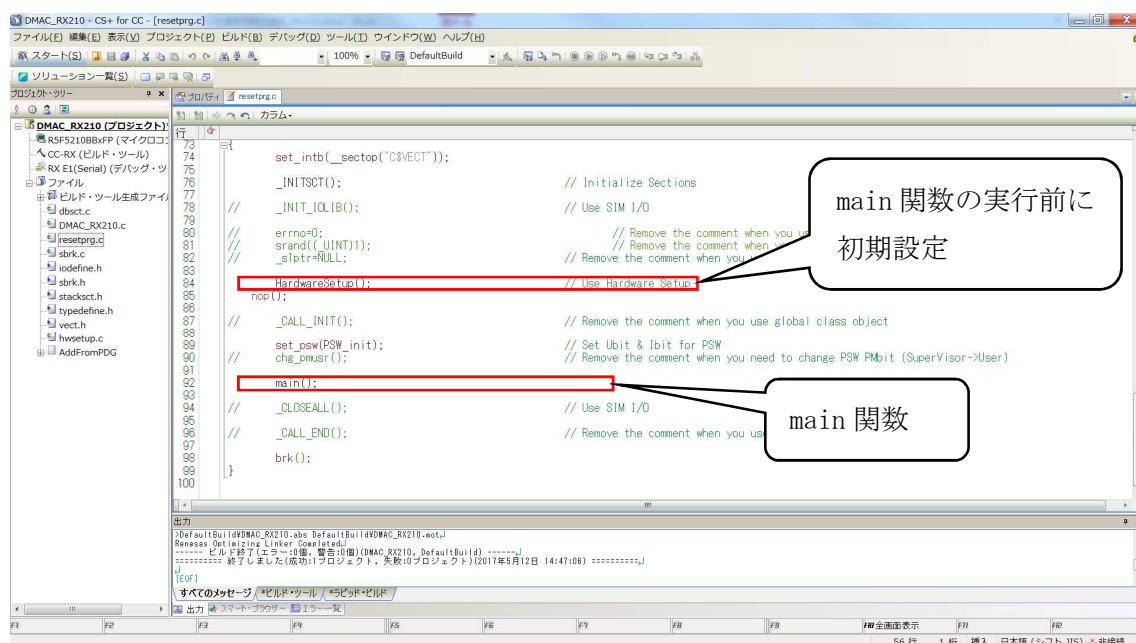


図 4.3 resetprg.c

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	13	通信バッファサイズ
VECT_SIZE	256	ベクタテーブルサイズ
DEST_SIZE	16	転送情報保存テーブルサイズ

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 char	send_buf[BUFF_SIZE]	SCI0 の送信データ	main Dmac1IntFunc

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 char	CH0_read_buf[BUFF_SIZE]	SCI0 の受信バッファ	Dmac1IntFunc Dmac3IntFunc
unsigned char	CH5_read_buf[BUFF_SIZE]	SCI5 の受信バッファ	main Dmac3IntFunc

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する 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
Dmac0IntFunc	DMAC0 割り込み通知関数 ※DMAC0 の転送完了(最後の送信データを TDR0 へ転送し、SCI0 送信データエンプティとなるタイミング)
Dmac1IntFunc	DMAC1 割り込み通知関数 ※SCI0 受信完了
Dmac2IntFunc	DMAC2 割り込み通知関数 ※DMAC2 の転送完了(最後の送信データを TDR5 へ転送し、SCI5 送信データエンプティとなるタイミング)
Dmac3IntFunc	DMAC3 割り込み通知関数 ※SCI5 受信完了
Sci0ErFunc	SCI0 受信エラー関数
Sci5ErFunc	SCI5 受信エラー関数

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 の受信開始 SCI0 の送信開始
引数	なし
リターン値	なし

Dmac0IntFunc

概要	TDR0 へ送信データ転送終了処理
ヘッダ	なし
宣言	void Dmac0IntFunc(void)
説明	SCI0 の送信完了ではないため、シリアル送信の終了待ち SCI0 のシリアル送受信停止 DMAC0 の停止
引数	なし
リターン値	なし

Dmac1IntFunc

概要	SCI0 受信完了処理
ヘッダ	なし
宣言	void Dmac1IntFunc(void)
説明	SCI0 のシリアル送受信停止 DMAC1 の停止 送信データと受信データの比較 D3(LED)制御
引数	なし
リターン値	なし

Dmac2IntFunc

概要	TDR5 へ送信データ転送終了処理
ヘッダ	なし
宣言	void Dmac2IntFunc(void)
説明	SCI5 の送信完了ではないため、シリアル送信の終了待ち SCI5 のシリアル送受信停止 DMAC2 の停止
引数	なし
リターン値	なし

Dmac3IntFunc

概要	SCI5 受信完了処理
ヘッダ	なし
宣言	void Dmac3IntFunc(void)
説明	SCI5 のシリアル送受信停止 DMAC3 の停止 SCI0 の受信開始 SCI5 の受信データを送信開始
引数	なし
リターン値	なし

Sci0ErFunc

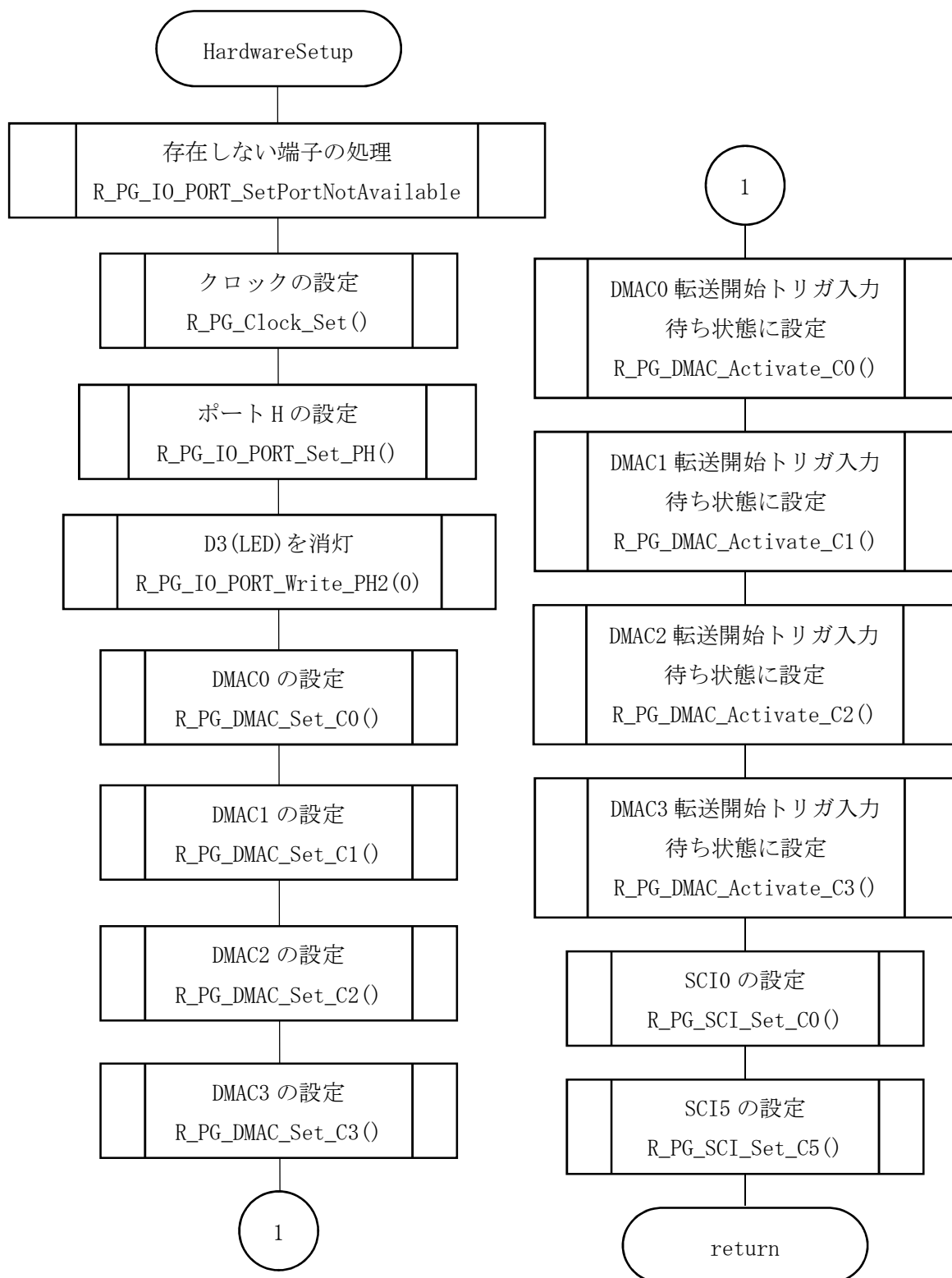
概要	SCI0 受信エラー処理
ヘッダ	なし
宣言	void Sci0ErFunc(void)
説明	処理なし
引数	なし
リターン値	なし

Sci5ErFunc

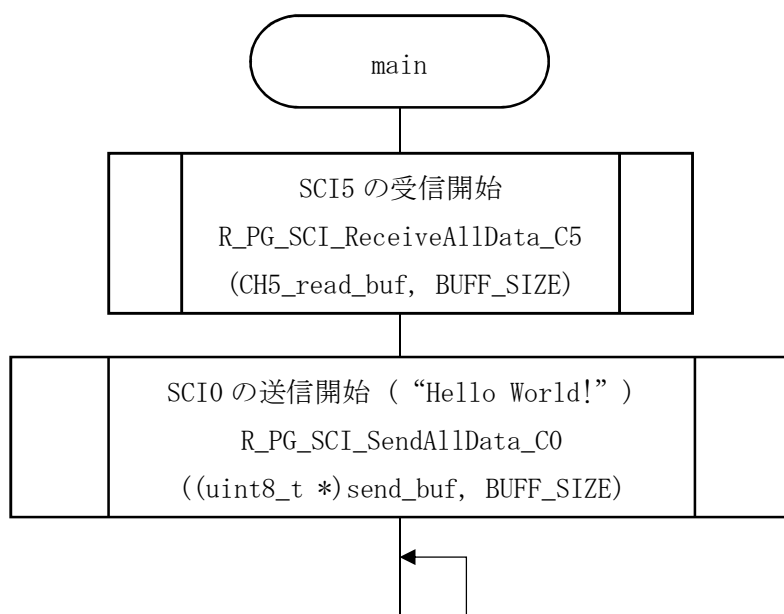
概要	SCI5 受信エラー処理
ヘッダ	なし
宣言	<code>void Sci5ErFunc(void)</code>
説明	処理なし
引数	なし
リターン値	なし

4.8 作成する関数のフローチャート

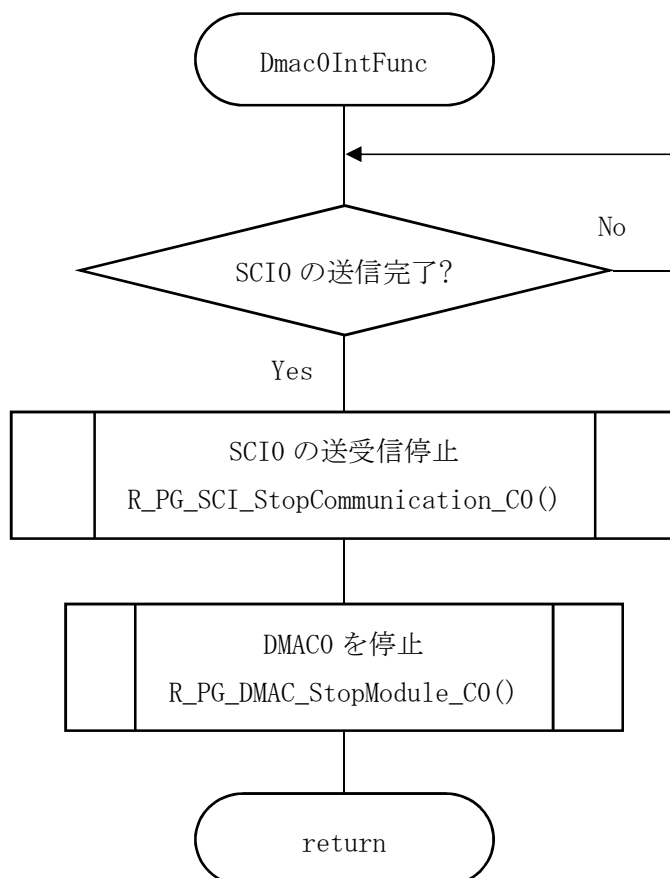
4.8.1 初期設定



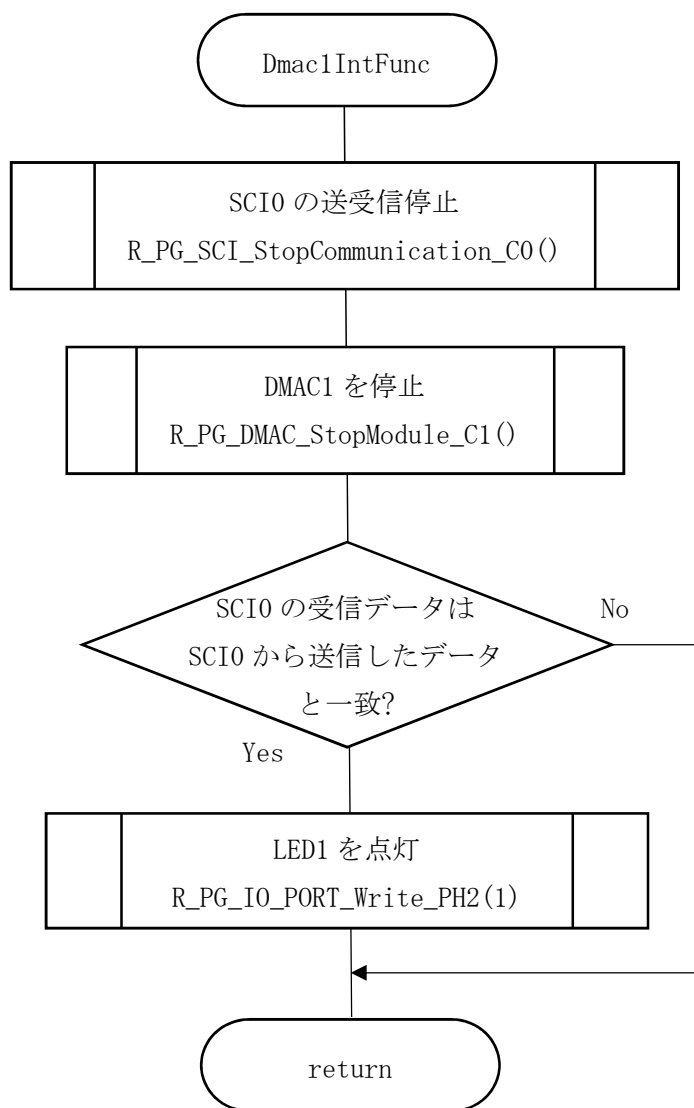
4.8.2 メイン処理



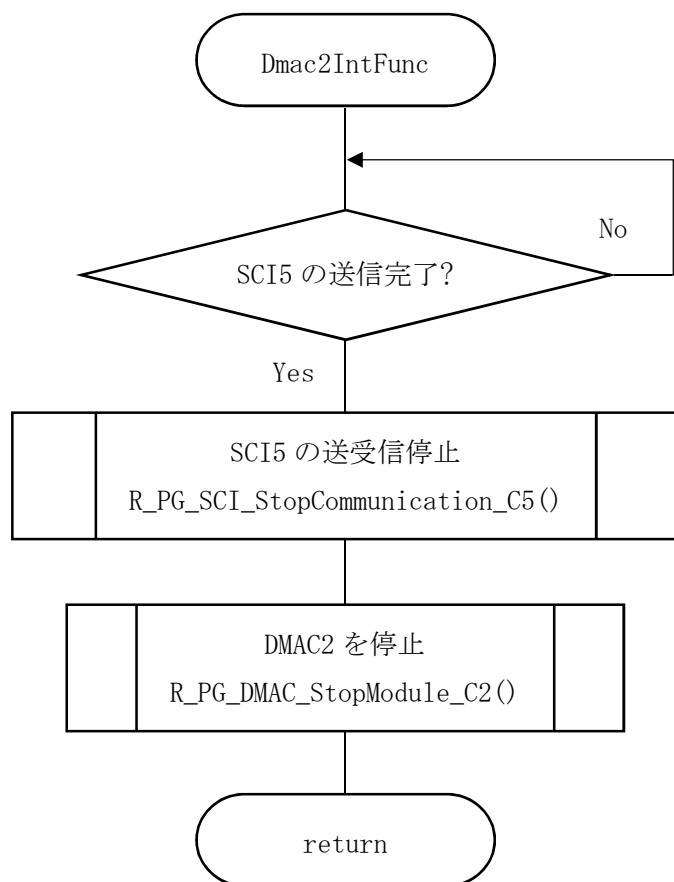
4.8.3 TDR0 へ送信データ転送終了処理



4.8.4 SCI0 受信完了処理



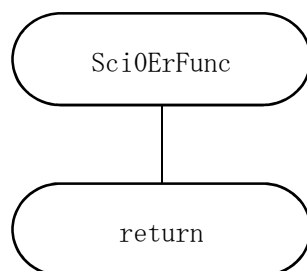
4.8.5 TDR2 へ送信データ転送終了処理



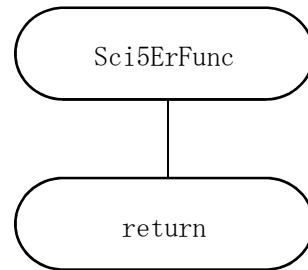
4.8.6 SCI2 受信完了処理



4.8.7 SCI0 受信エラー処理



4.8.8 SCI5 受信エラー処理



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:
DMAC_RX210

ディレクトリ: ?
c:\renesas\PDG2_proj 参照...

デバイス選択

シリーズ: RX200

グループ: RX210

型名: R5F5210BBxFP

パッケージ: PLQP0100KB-A

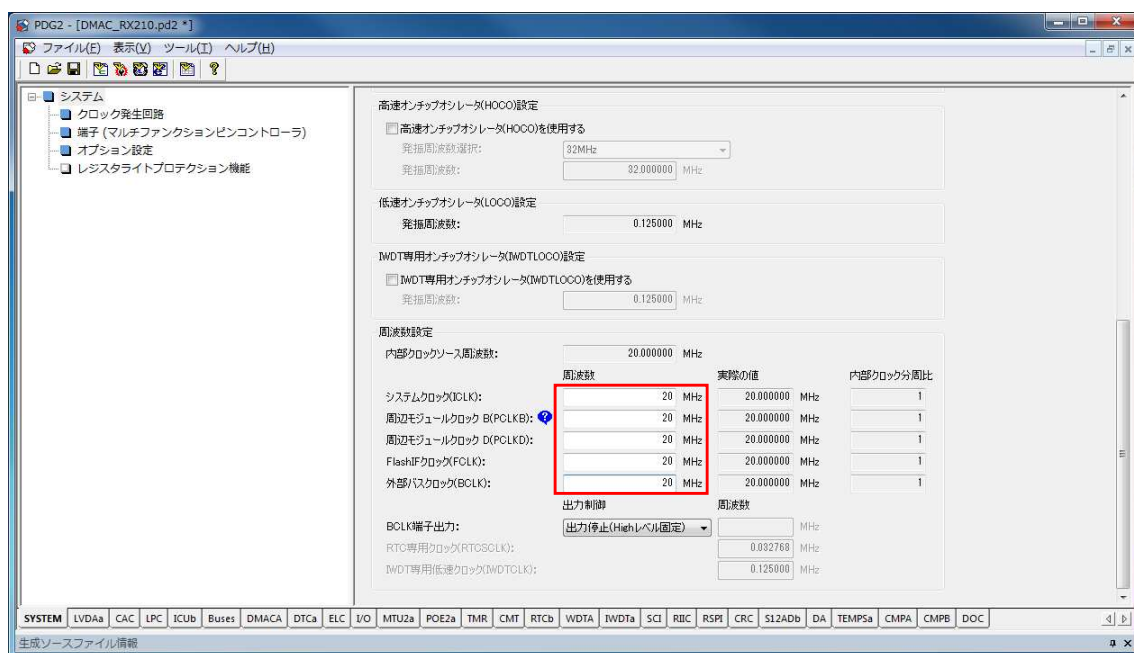
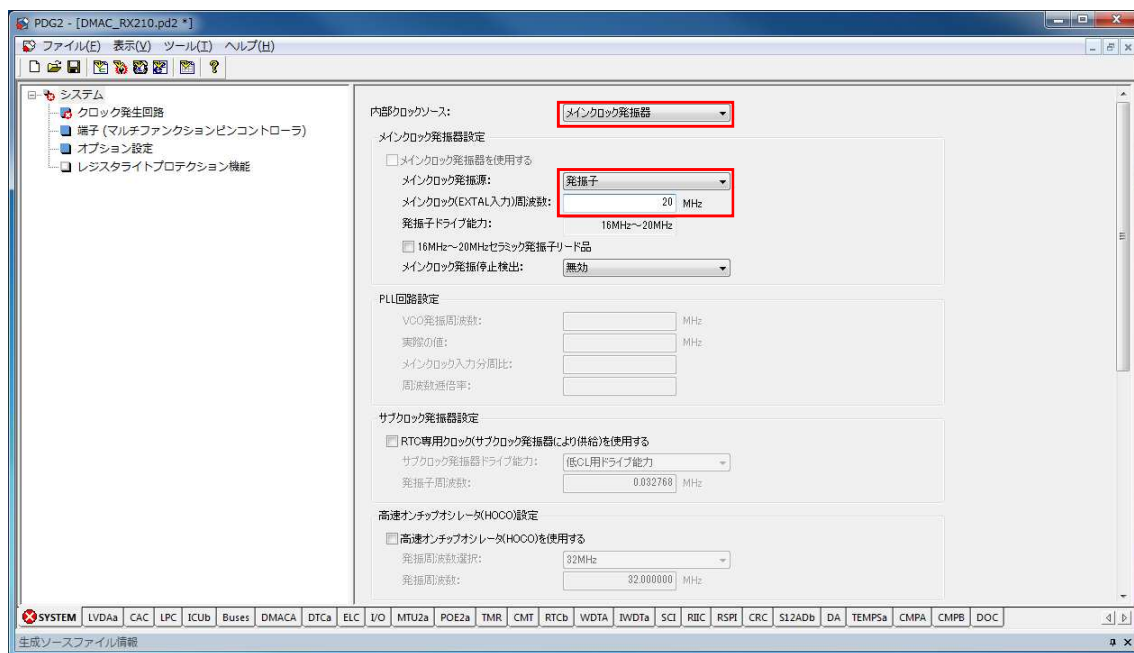
ROM容量: 1M バイト

RAM容量: 96K バイト

クリック OK キャンセル

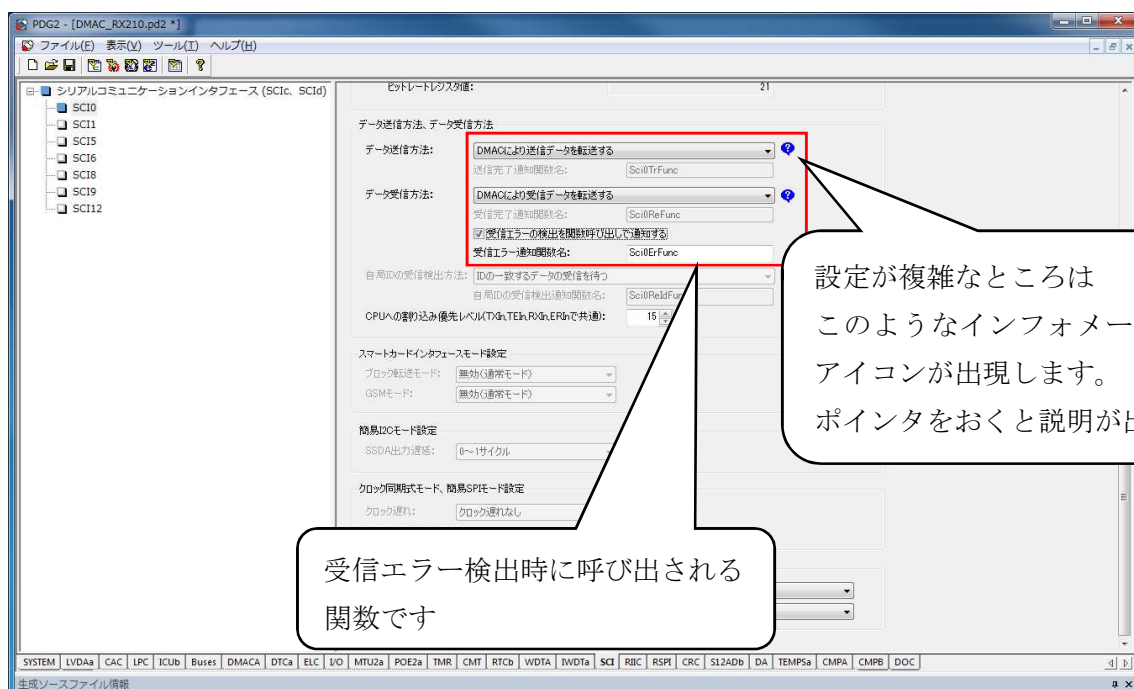
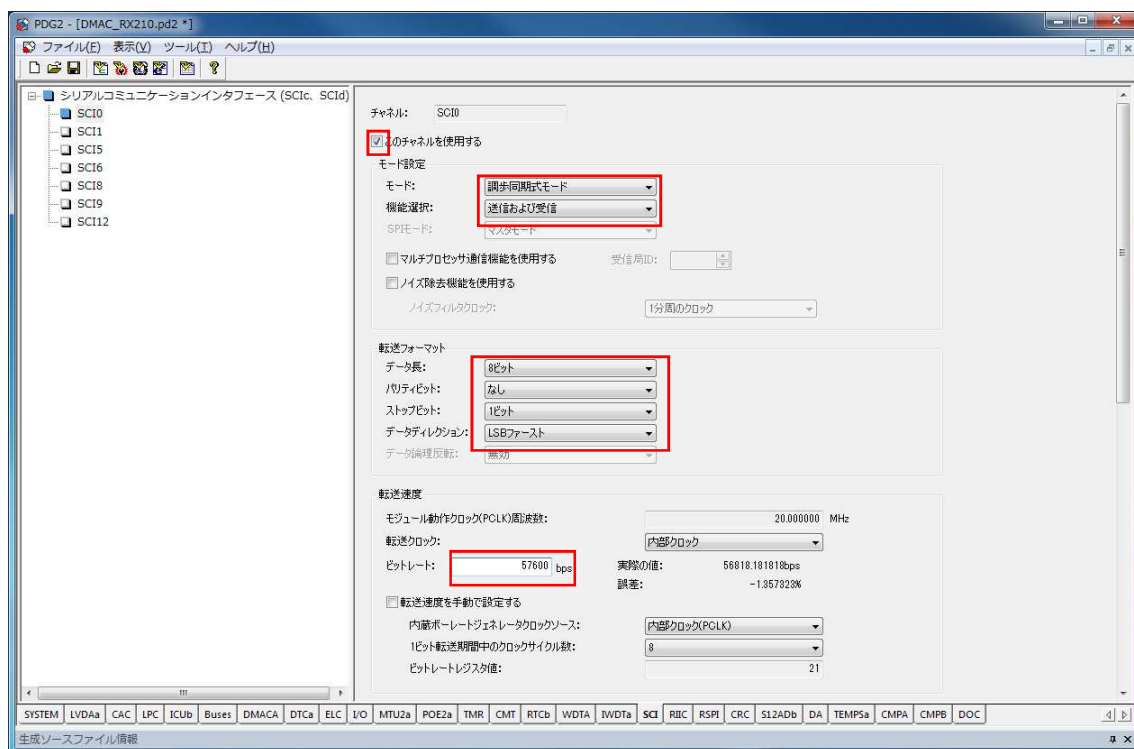
5.1 SYSTEM 設定

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 SCI0 設定

SCI0 の設定を以下に示します。



「DMAC により送信データを転送する」のインフォメーションアイコンをポイントしたときは以下が表示されます。

後程 DMAC の設定をする際に重要です。設定におけるポイントを赤枠で囲みます。

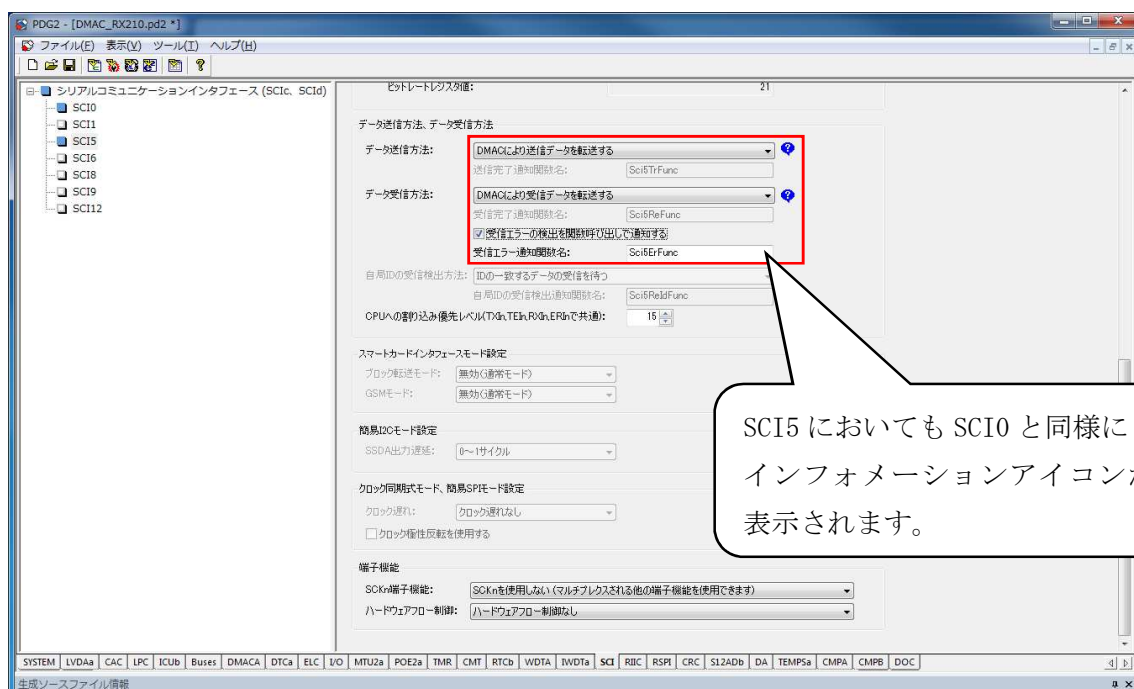
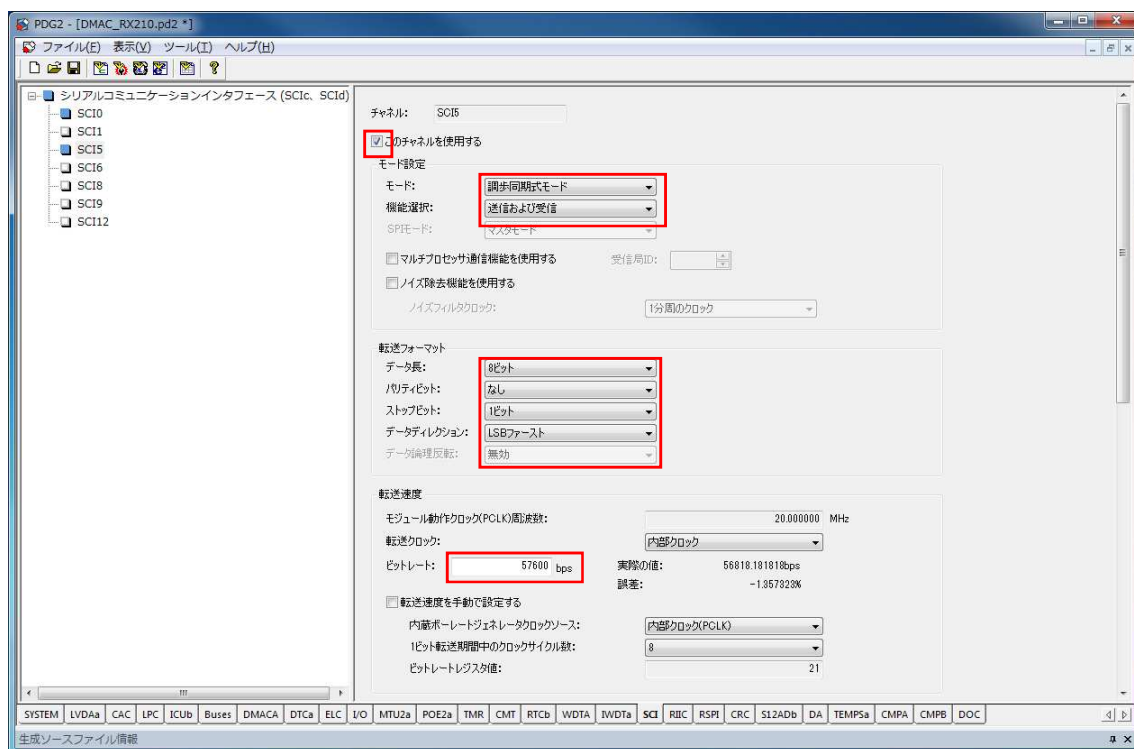
DMACにより送信するデータを転送する場合、送信データエンプティ割り込み(TXIn)をトリガとしたDMAC転送を設定し、トランスミットデータレジスタのアドレスをデータ転送先に指定してください。関数 R_PG_SCI_SendAllData_Cn を呼ぶと、DMAC転送を開始しデータを送信します。転送終了時はDMAC割り込みの通知関数内から R_PG_SCI_StopCommunication を呼び出してください。

「DMAC により受信データを転送する」のインフォメーションアイコンをポイントしたときは以下が表示されます。

DMACにより受信したデータを転送する場合、受信データフル割り込み(RXIn)をトリガとしたDMAC転送を設定し、レシーブデータレジスタのアドレスをデータ転送元に指定してください。関数 R_PG_SCI_ReceiveAllData_Cn を呼ぶと、受信を開始しDMAC転送を行います。転送終了時はDMAC割り込みの通知関数内から R_PG_SCI_StopCommunication を呼び出してください。

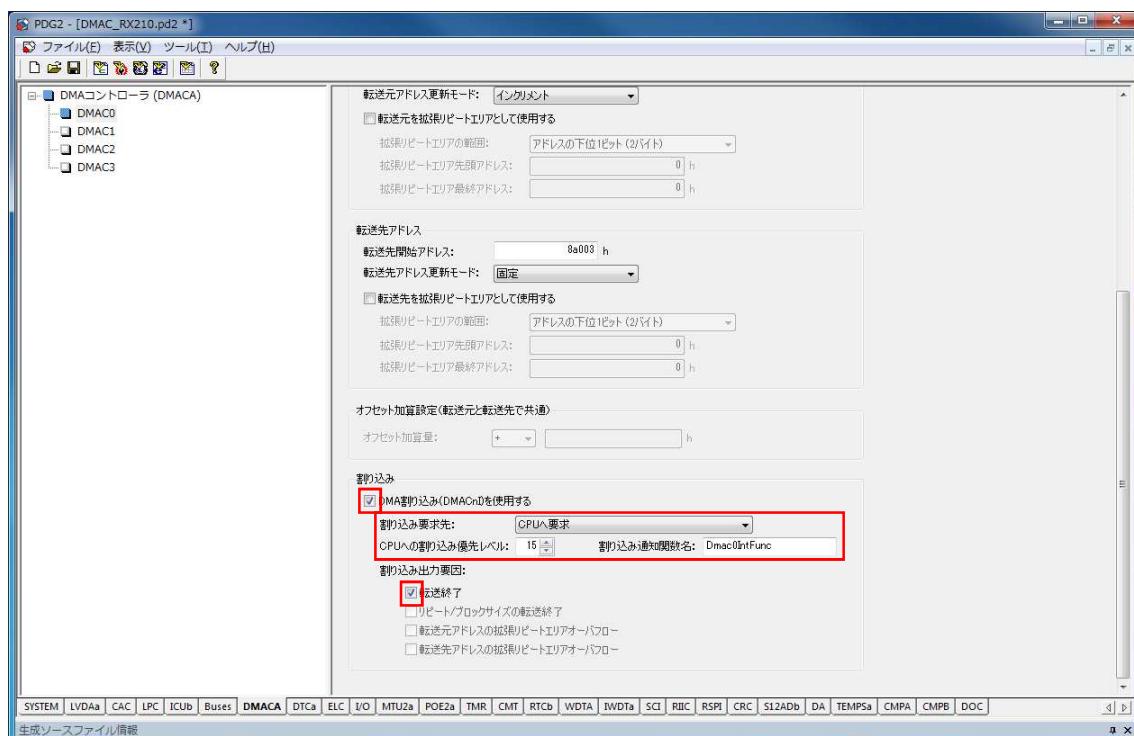
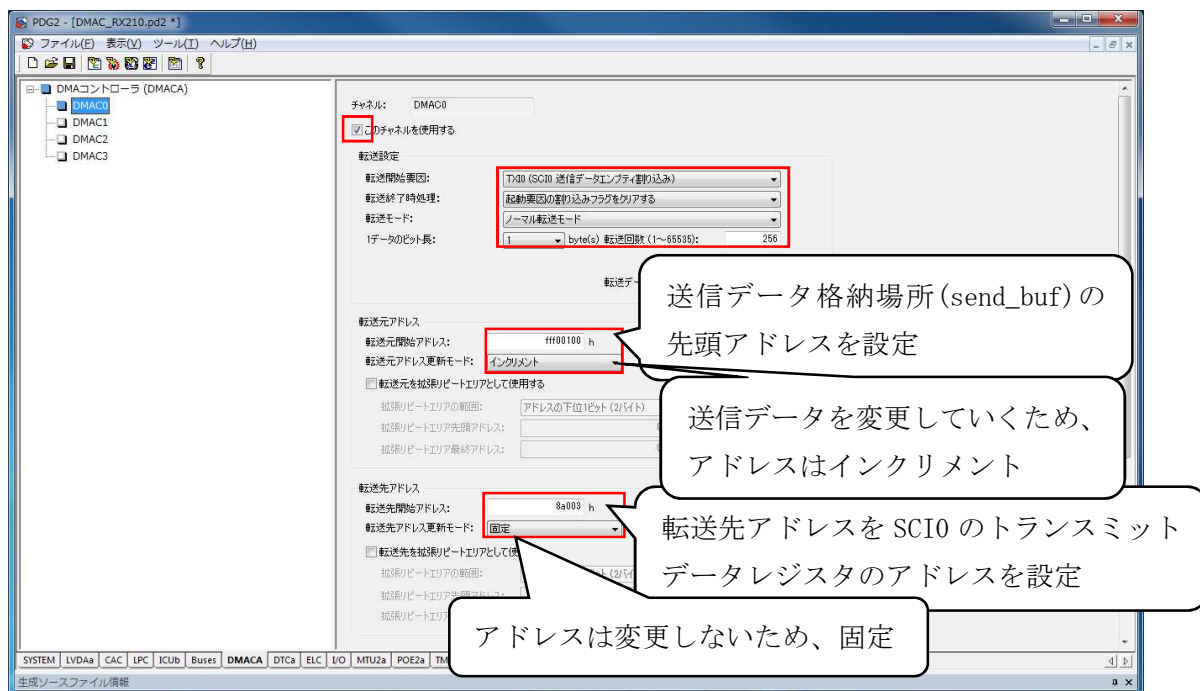
5.3 SCI5 設定

SCI5 の設定を以下に示します。(SCI0 と同様です)

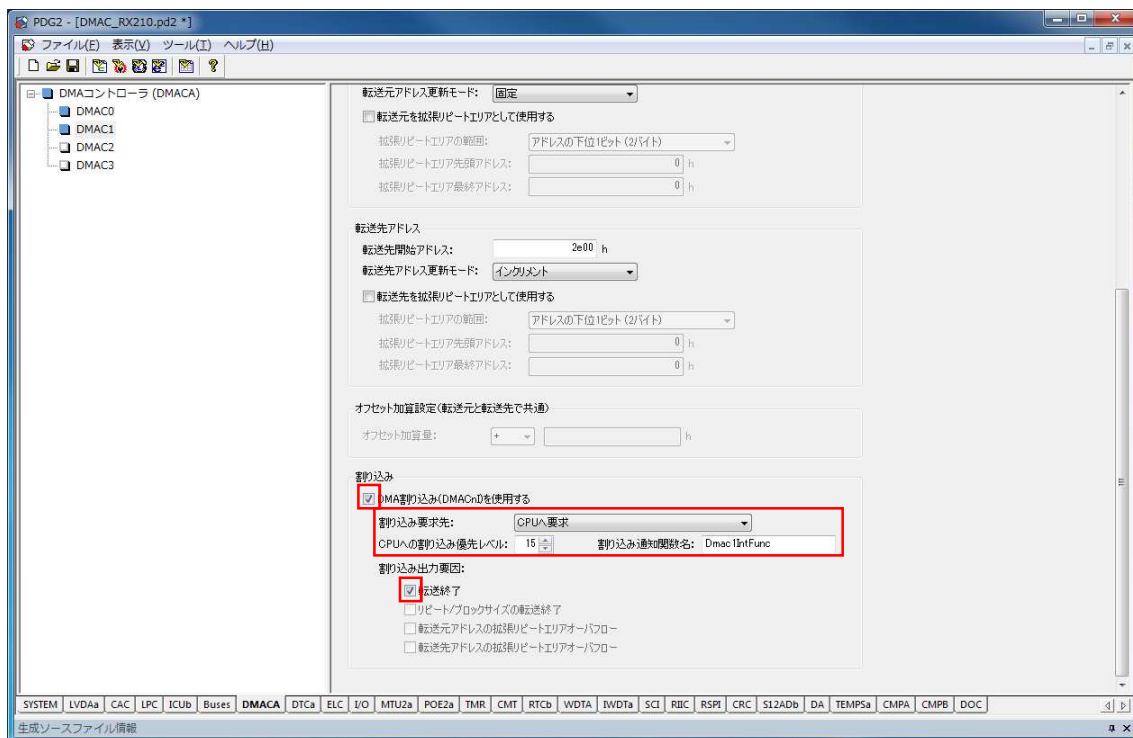
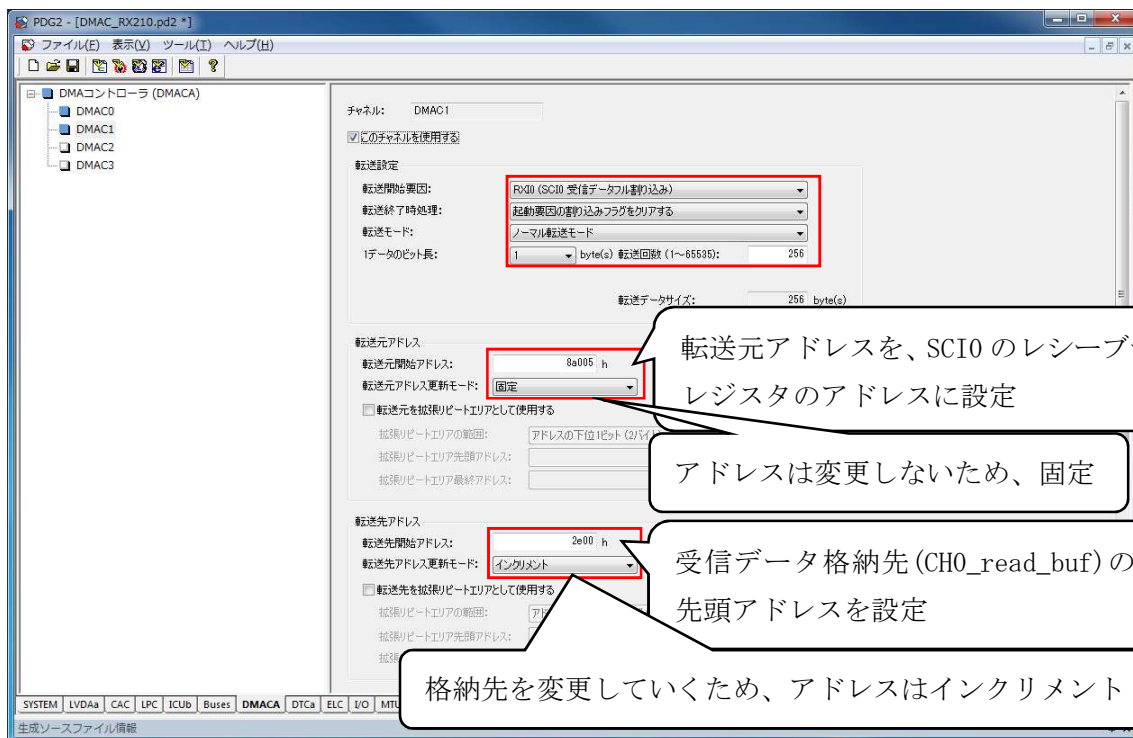


5.4 DMAC の設定

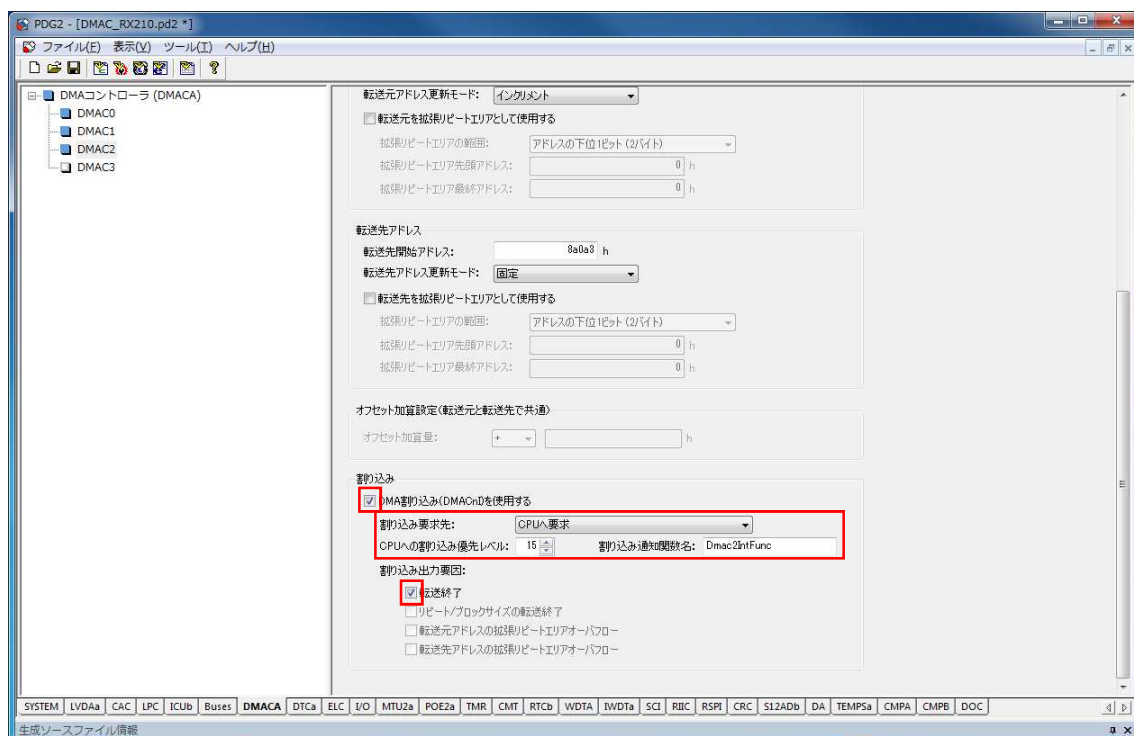
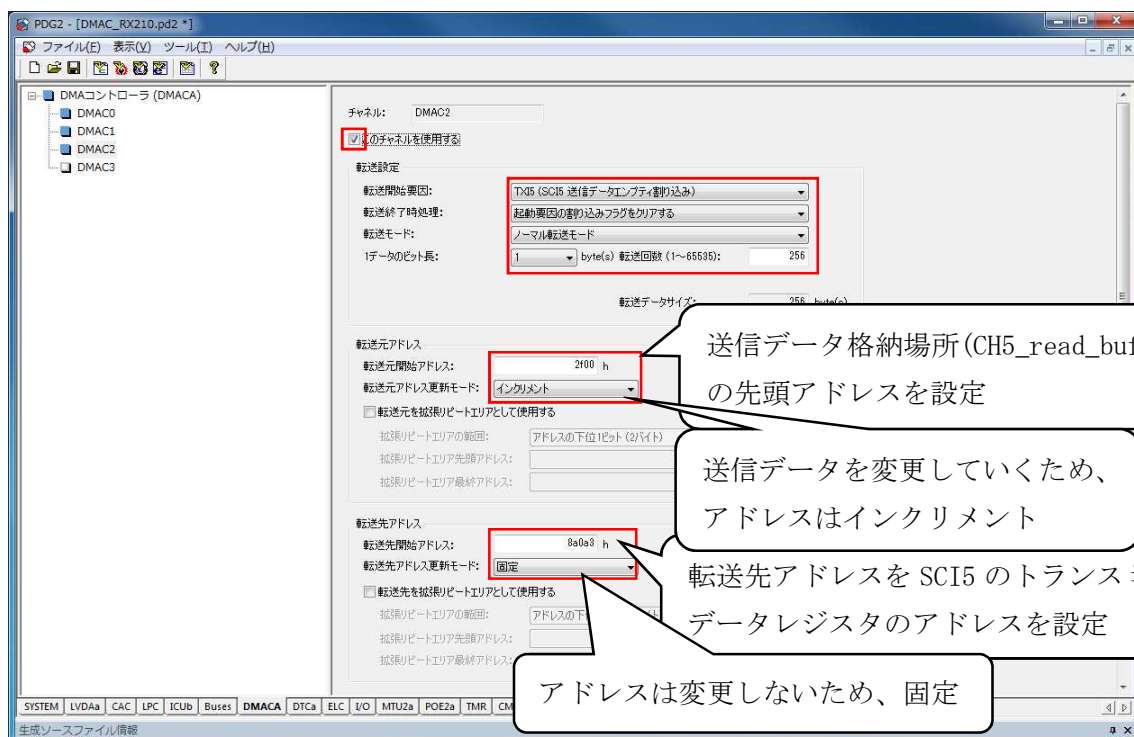
DMAC0 の設定を以下に示します。先程のインフォメーションアイコンの説明に従います。



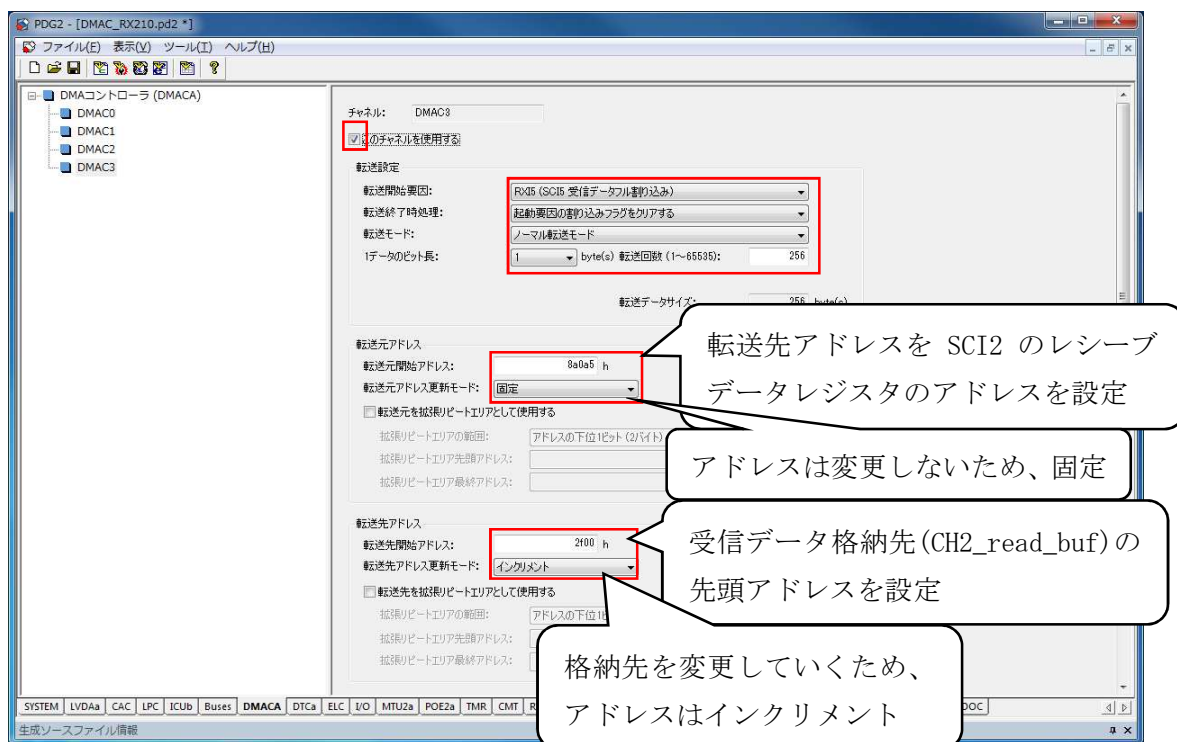
DMAC1 の設定を以下に示します。先程のインフォメーションアイコンの説明に従います。



DMAC2 の設定を以下に示します。先程のインフォメーションアイコンの説明に従います。

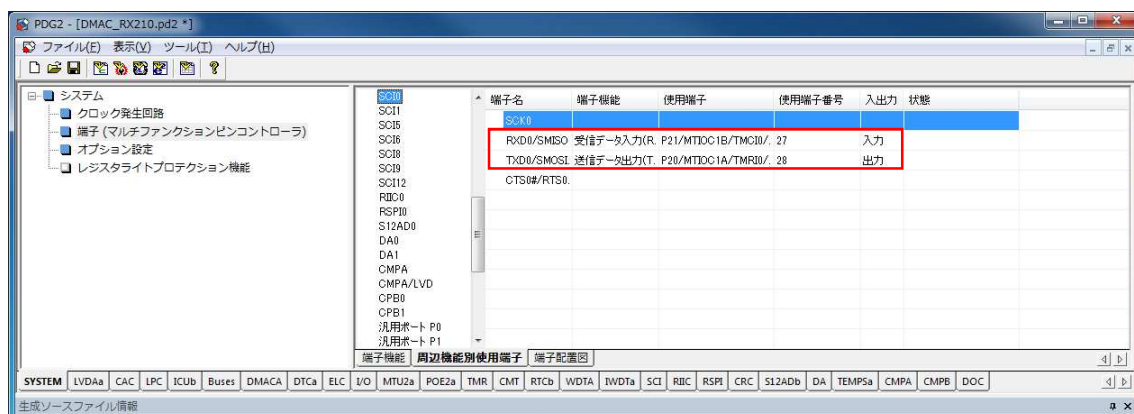


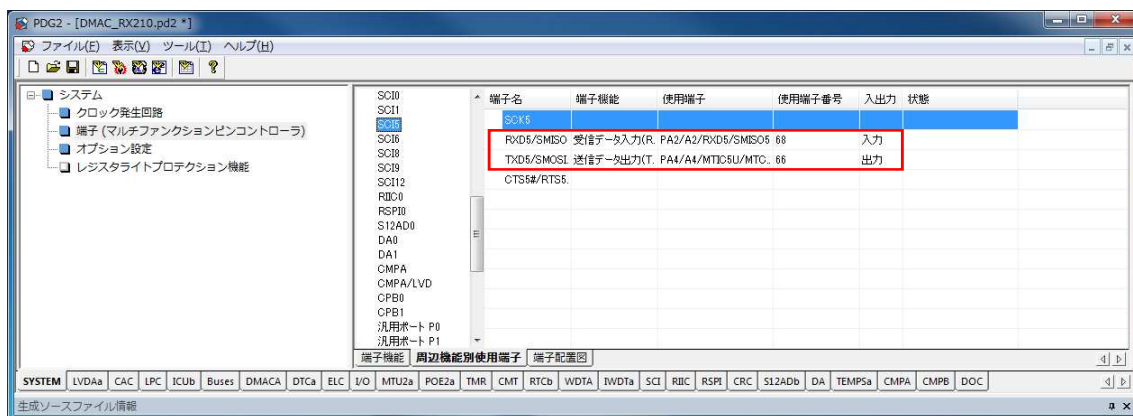
DMAC3 の設定を以下に示します。先程のインフォメーションアイコンの説明に従います。



5.5 SYSTEM の端子設定

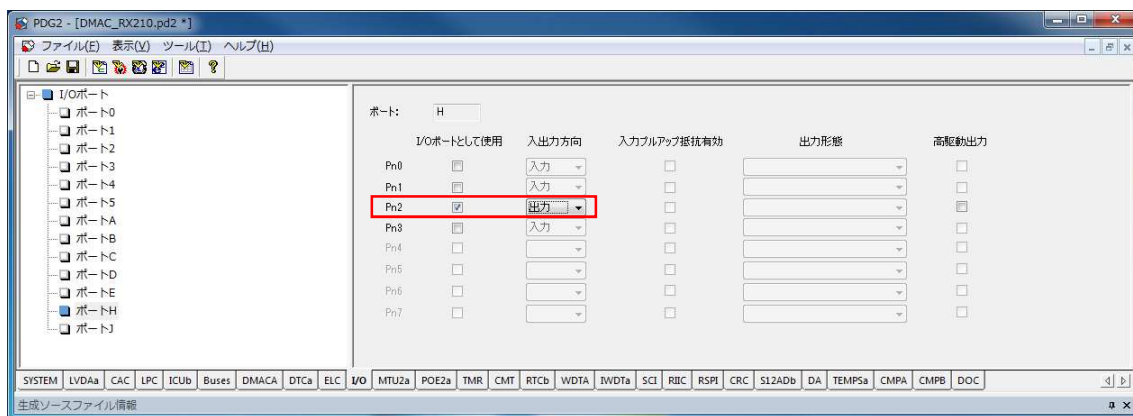
SYSTEM の端子設定を確認します。





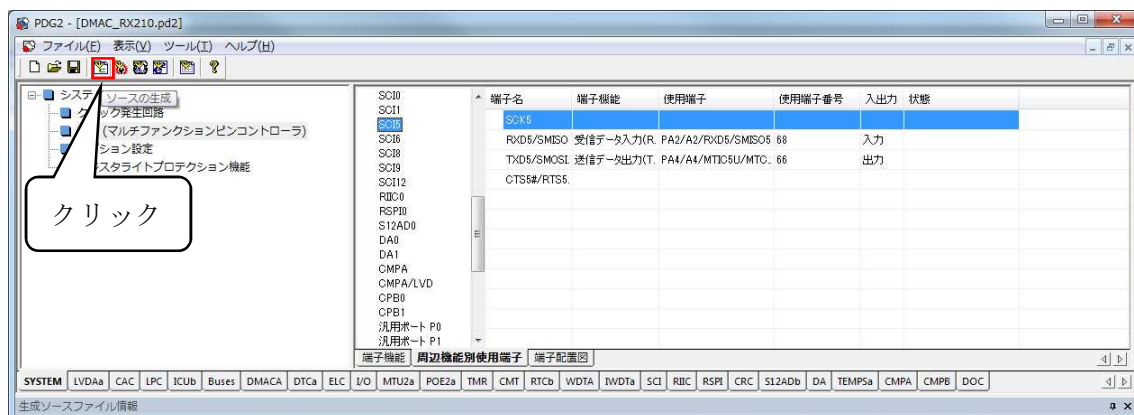
5.6 I/O 設定

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

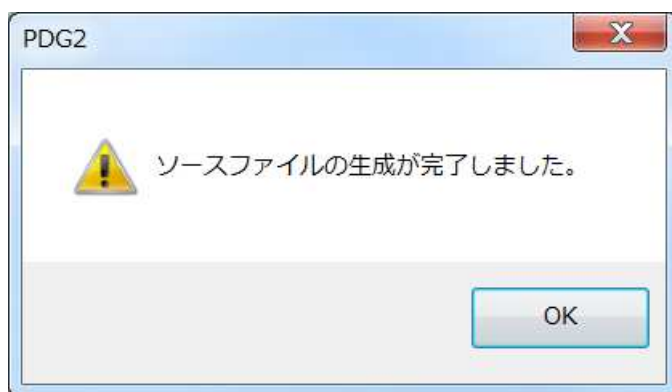


5.7 ソースの生成

以下の GUI をクリックすると、

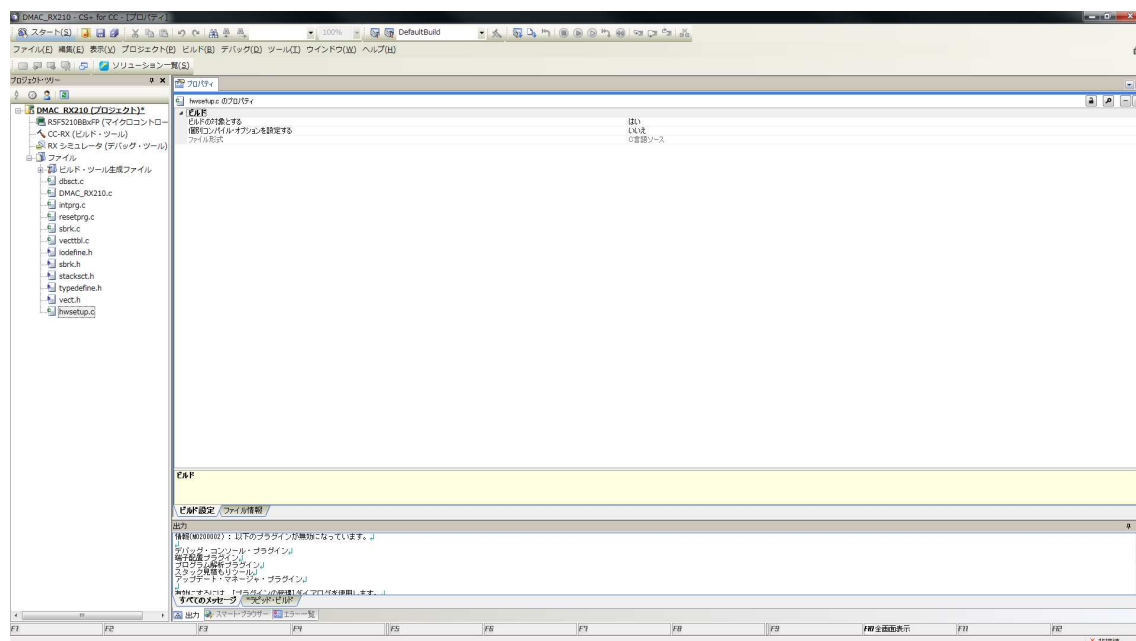


ソースファイルが生成されます。

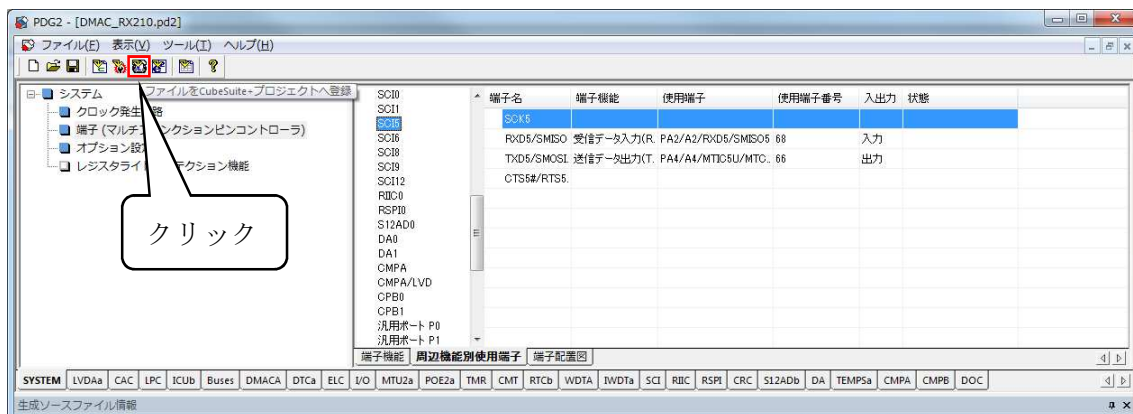


5.8 CS+への登録

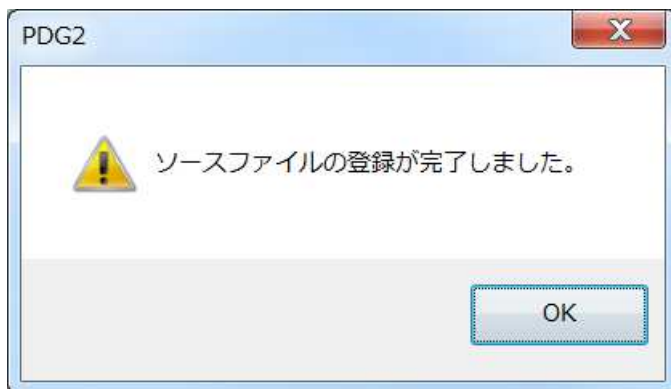
対象の CS+プロジェクトを開きます。



以下の GUI をクリックします

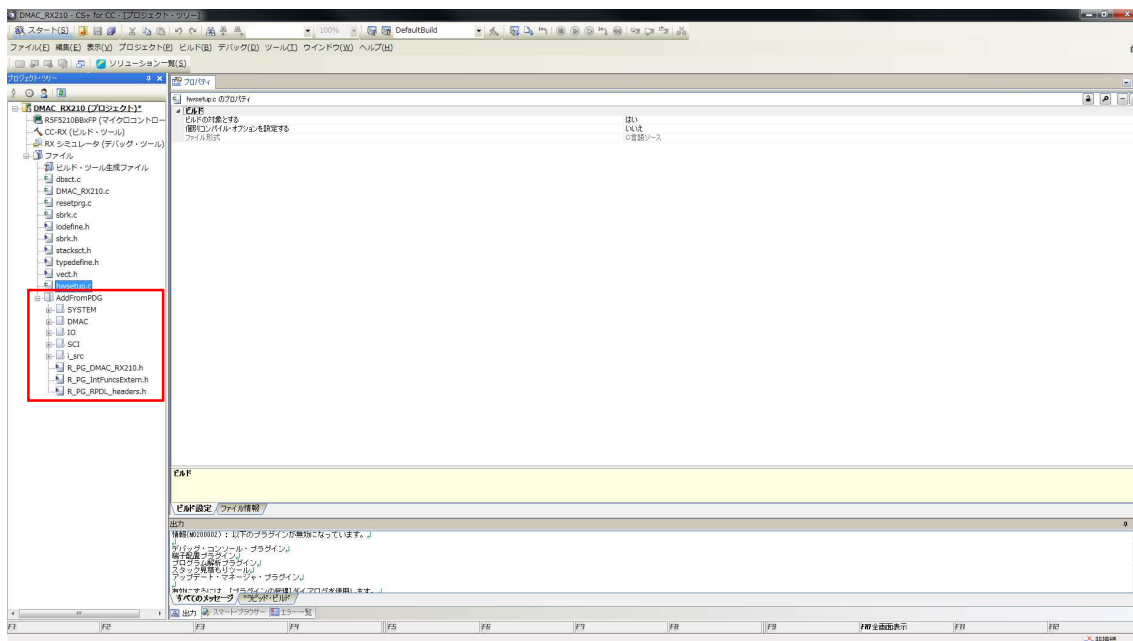


ソースファイルの登録が完了しました。



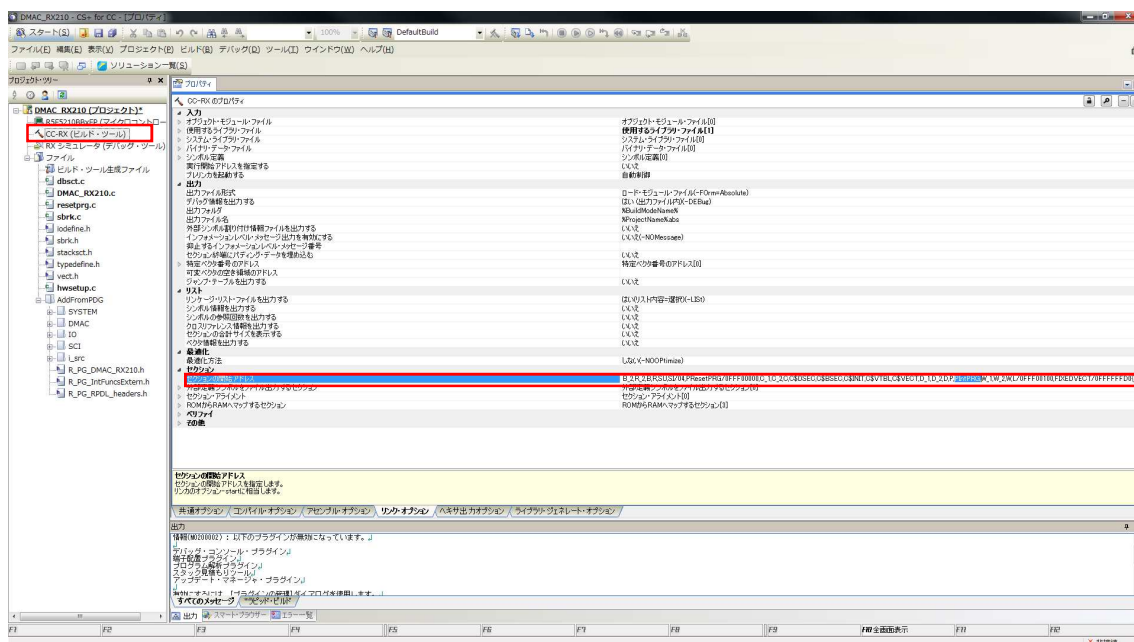
6. CS+のプロジェクトに PDG のソースファイルを登録する際の設定

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。

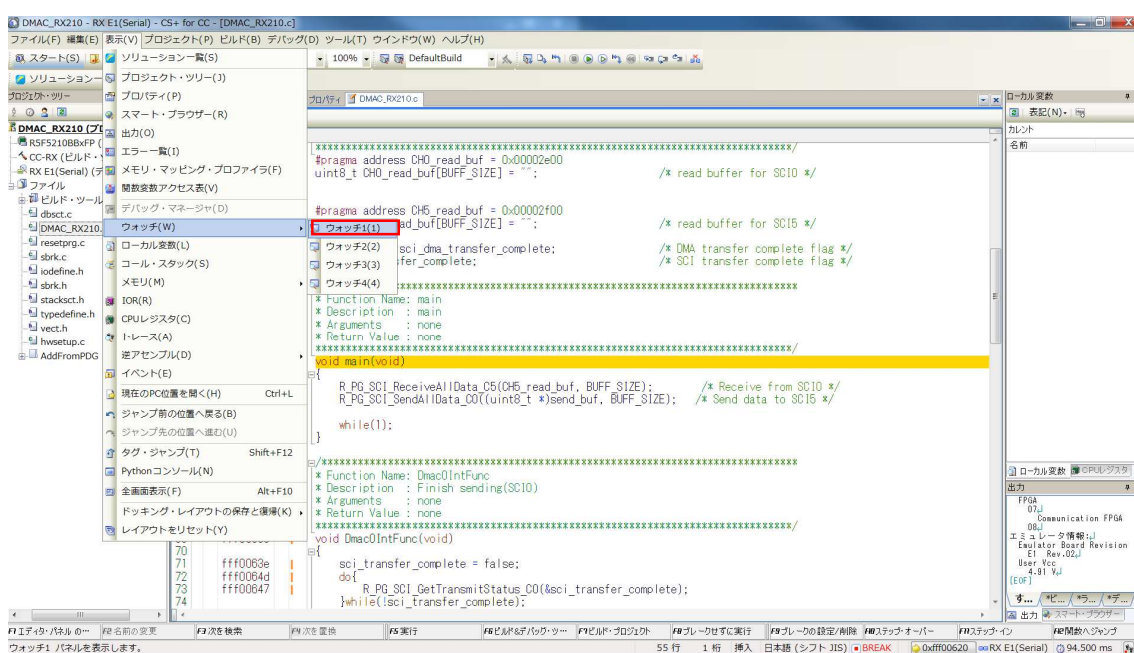


7. 動作確認方法

7.1 ウォッチ式の登録

DMAC を用いて SCI0 から SCI5 にデータが送信され、SCI5 の受信データが SCI0 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。

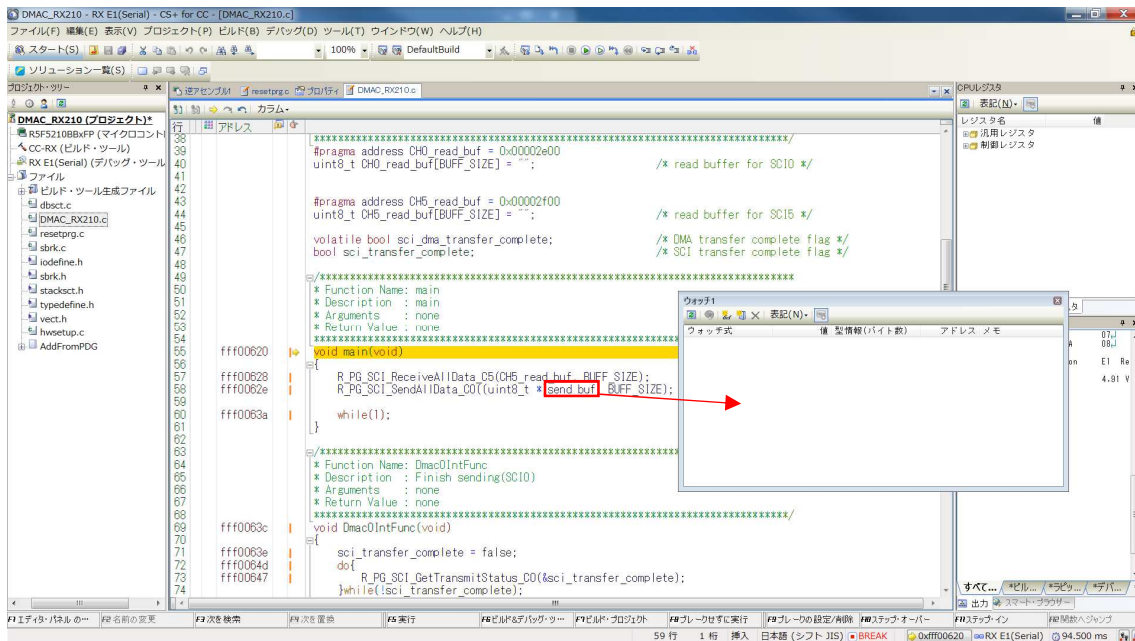
表示タブからウォッチ、そしてウォッチ 1 を選択します。



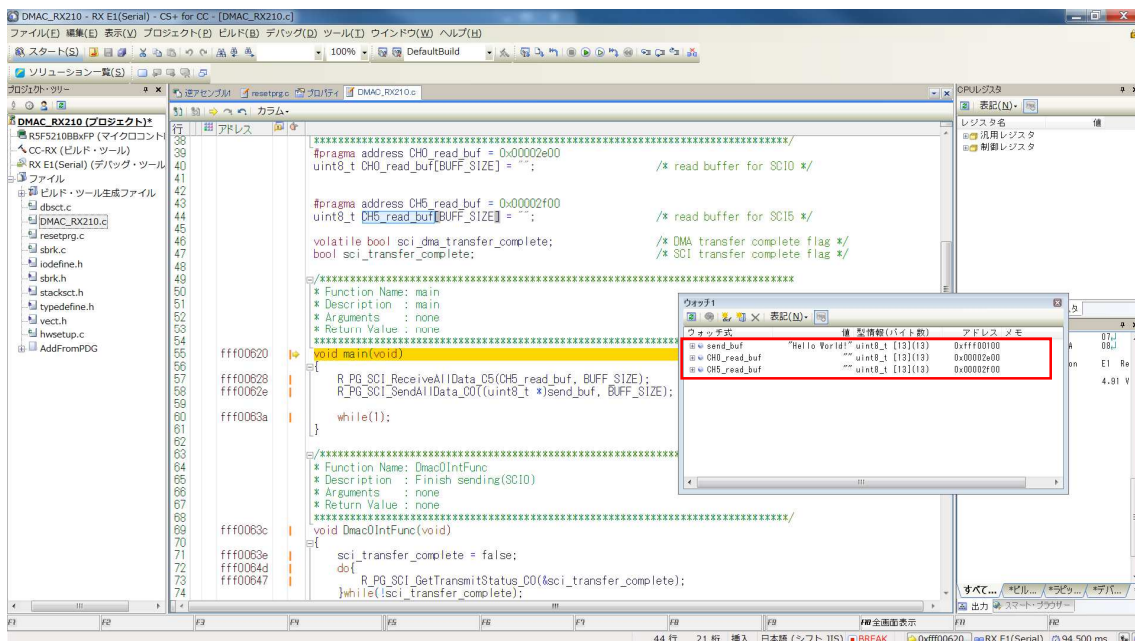
ウォッチ(ウォッチ 1)パネルが表示されました。



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 “CH5_read_buf” も同様にドラッグ&ドロップします。



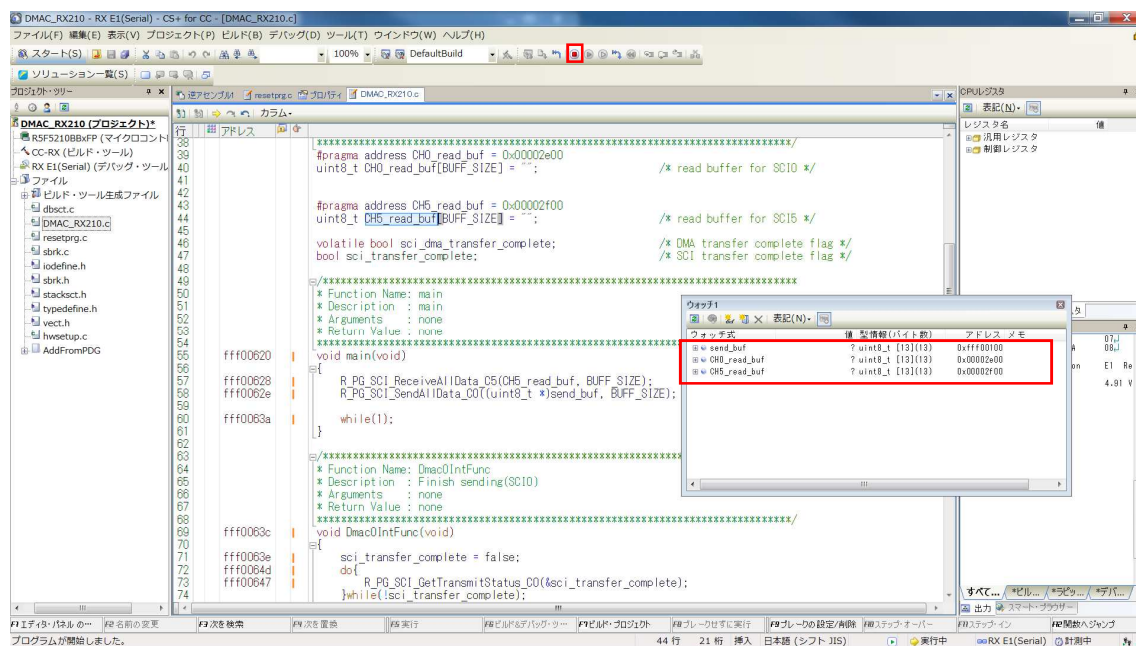
上の図のようにウォッチパネルには “send_buf” にのみ “Hello World!” が表示され、“CH0_read_buf” と “CH5_read_buf” は空欄であることが確認できます。ボード側は、D3(LED)は消灯となっています。この状態で実行前の準備は完了です。

7.2 実行

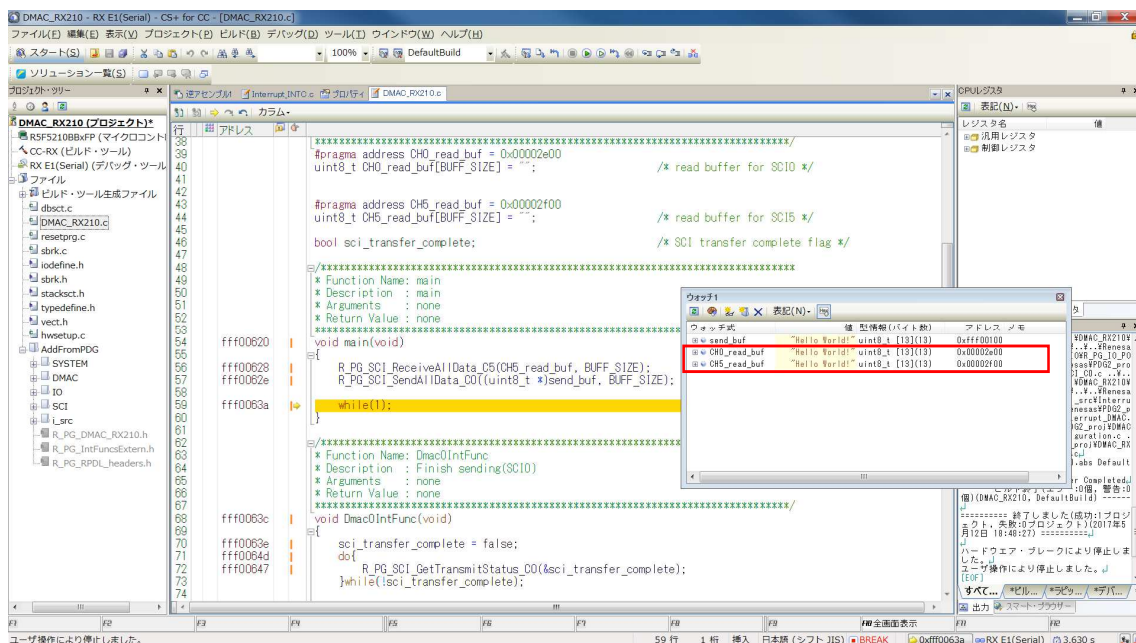
実行ボタンをクリックします。

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、D3 (LED) は点灯しました。

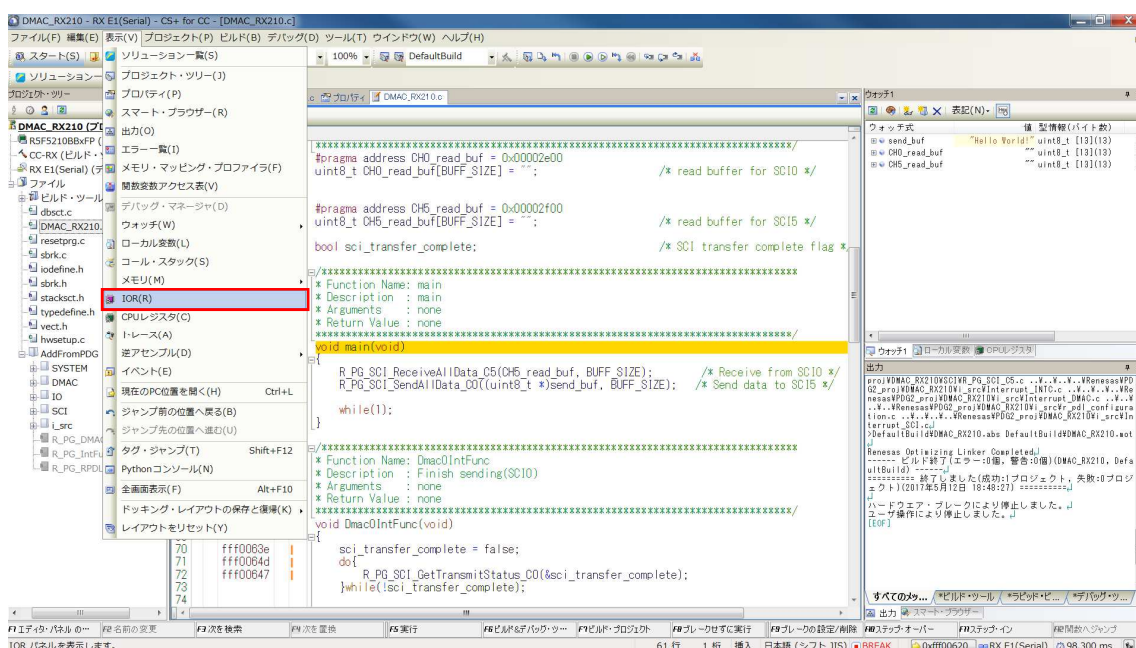
ここで、停止をクリックします。



プログラムを停止すると、ウォッチパネルで“CH0_read_buf”と“CH5_read_buf”は共に“Hello World!”が表示されています。これで SCI5 は SCI0 から“Hello World!”を受信できており、SCI0 は SCI5 から送り返された“Hello World!”を受信できたことが確認できました。



なお、DMAC 関連のレジスタにも変化が見られるため、確認します。表示タブから IOR を選択します。実行前と実行後の IOR を拡大します。



実行前の DMAC0 は次の通りです。

IOR	値	型情報(バイト数)	アドレス
DMAC0			
DMAC0.DMSAR	0xffff00100	IOR(4)	0x00082000
DMAC0.DMDAR	0x0008a003	IOR(4)	0x00082004
DMAC0.DMCRA	0x0000000d	IOR(4)	
DMAC0.DMCRB	0x0000	IOR(2)	0x0008200c
DMAC0.DMTMD	0x0001	IOR(2)	0x00082010
DMAC0.DMT...	0x0	IOR(2ビット)	0x00082010...
DMAC0.DMT...	0x0	IOR(2ビット)	0x00082010...
DMAC0.DMT...	0x0	IOR(2ビット)	0x00082010.8
DMAC0.DMT...	0x1	IOR(2ビット)	0x00082010.8

実行前の DMAC1 は次の通りです。

IOR	値	型情報(バイト数)	アドレス
DMAC1			
DMAC1.DMSAR	0x0008a005	IOR(4)	
DMAC1.DMDAR	0x00002e00	IOR(4)	0x00082044
DMAC1.DMCRA	0x0000000d	IOR(4)	
DMAC1.DMCRB	0x0000	IOR(2)	0x0008204c
DMAC1.DMTMD	0x0001	IOR(2)	0x00082050
DMAC1.DMT...	0x0	IOR(2ビット)	0x00082050...
DMAC1.DMT...	0x0	IOR(2ビット)	0x00082050...
DMAC1.DMT...	0x0	IOR(2ビット)	0x00082050.8
DMAC1.DMT...	0x1	IOR(2ビット)	0x00082050.8

実行前の DMAC2 は次の通りです。

IOR	値	型情報(バイト数)	アドレス
DMAC2			
DMAC2.DMSAR	0x00002f00	IOR (4)	0x00082080
DMAC2.DMDAR	0x0008a0a3	IOR (4)	0x00082084
DMAC2.DMCRA	0x0000000d	IOR (4)	
DMAC2.DMCRB	0x0000	IOR (2)	0x0008208c
DMAC2.DMTMD	0x0001	IOR (2)	0x00082090
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090...
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090...
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090.8
DMAC2.DMT...	0x1	IOR (2ビット)	0x00082090.8

実行前の DMAC3 は次の通りです。

IOR	値	型情報(バイト数)	アドレス
DMAC3			
DMAC3.DMSAR	0x0008a0a5	IOR (4)	
DMAC3.DMDAR	0x00002f00	IOR (4)	0x000820c4
DMAC3.DMCRA	0x0000000d	IOR (4)	
DMAC3.DMCRB	0x0000	IOR (2)	0x000820cc
DMAC3.DMTMD	0x0001	IOR (2)	0x000820d0
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0...
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0...
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0.8
DMAC3.DMT...	0x1	IOR (2ビット)	0x000820d0.8

実行後の DMAC0 は次の通りです。

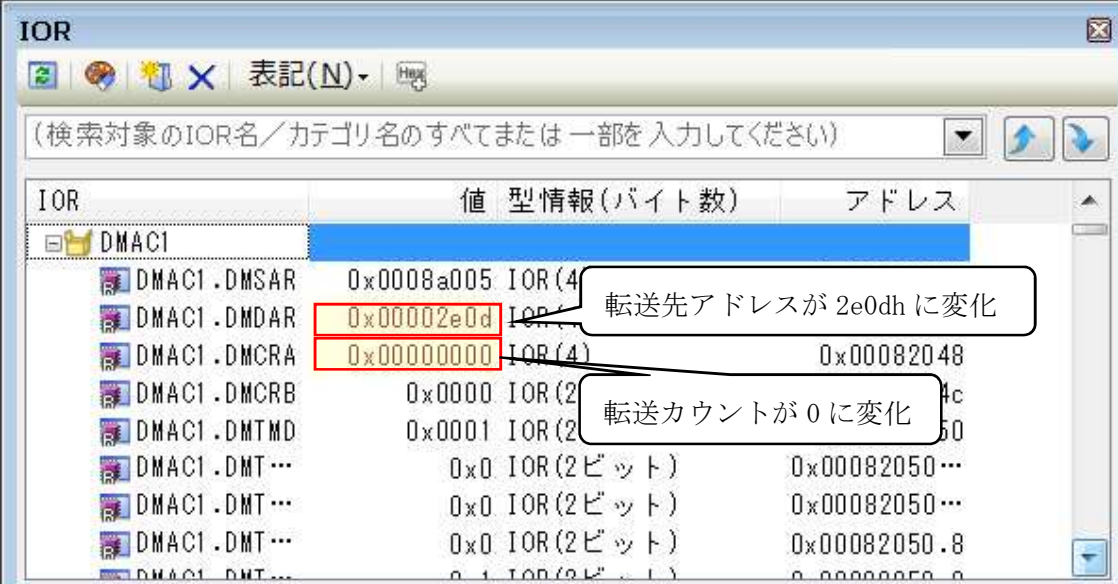
転送元アドレスにおいては 13 バイト分のデータ転送によりインクリメントされ、
fff00100h から fff0010dh になりました。転送カウンタはデクリメントされるため、dh(13
回)から 0 になりました。



IOR	値	型情報(バイト数)	アドレス
DMAC0			
DMAC0.DMSAR	0xffff0010d	IOR (4)	
DMAC0.DMDAR	0x0008a003	IOR (4)	0x00082004
DMAC0.DMCRA	0x00000000	IOR (4)	0x00082008
DMAC0.DMCRB	0x0000	IOR (2)	
DMAC0.DMTMD	0x0001	IOR (2)	
DMAC0.DMT...	0x0	IOR (2ビット)	0x00082010...
DMAC0.DMT...	0x0	IOR (2ビット)	0x00082010...
DMAC0.DMT...	0x0	IOR (2ビット)	0x00082010.8

実行後の DMAC1 は次の通りです。

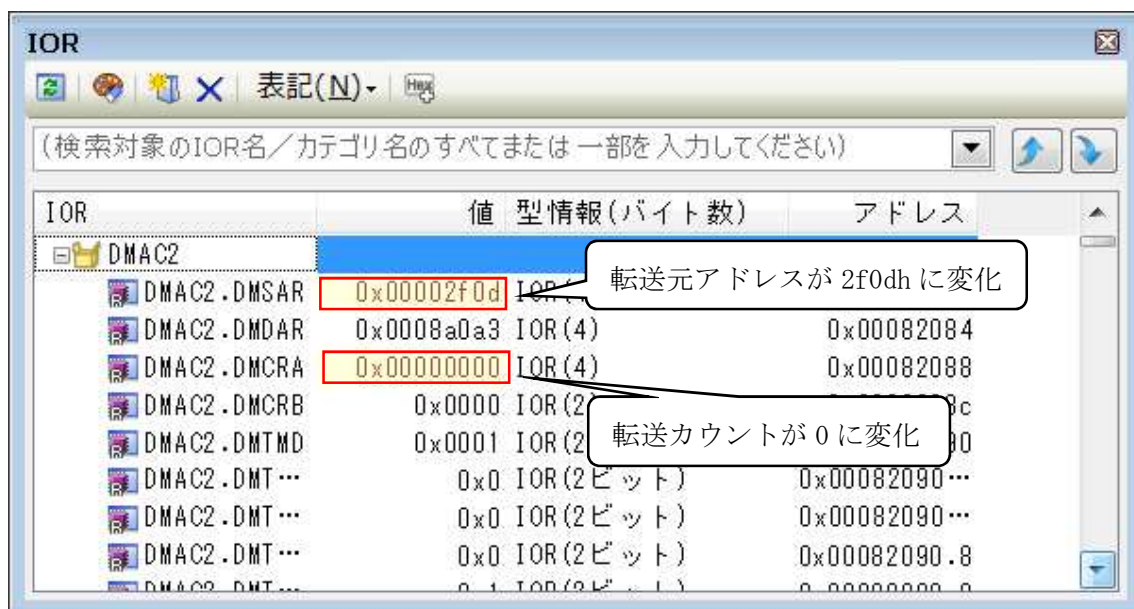
転送先アドレスにおいては 13 バイト分のデータ転送によりインクリメントされ、2e00h から 2e0dh になりました。転送カウンタはデクリメントされるため、dh(13 回)から 0 になりました。



IOR	値	型情報(バイト数)	アドレス
DMAC1			
DMAC1.DMSAR	0x0008a005	IOR (4)	
DMAC1.DMDAR	0x00002e0d	IOR (4)	
DMAC1.DMCRA	0x00000000	IOR (4)	0x00082048
DMAC1.DMCRB	0x0000	IOR (2)	
DMAC1.DMTMD	0x0001	IOR (2)	
DMAC1.DMT...	0x0	IOR (2ビット)	0x00082050...
DMAC1.DMT...	0x0	IOR (2ビット)	0x00082050...
DMAC1.DMT...	0x0	IOR (2ビット)	0x00082050.8

実行後の DMAC2 は次の通りです。

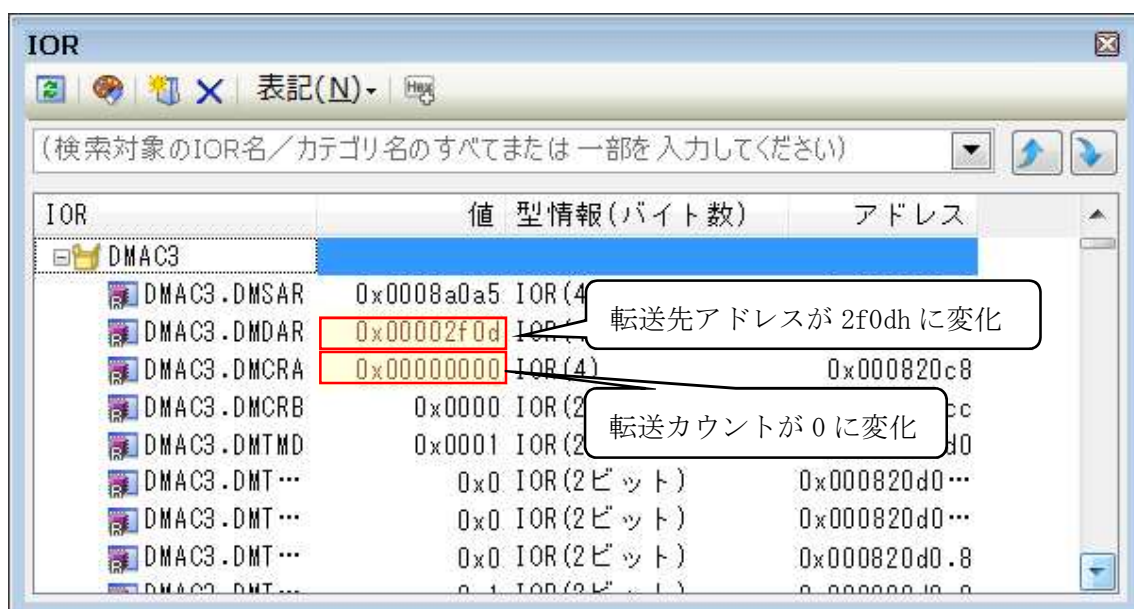
転送元アドレスにおいては 13 バイト分のデータ転送によりインクリメントされ、2f00h から 2f0dh になりました。転送カウンタはデクリメントされるため、dh(13 回)から 0 になりました。



IOR	値	型情報(バイト数)	アドレス
DMAC2.DMSAR	0x00002f0d	IOR (4)	
DMAC2.DMDAR	0x0008a0a3	IOR (4)	0x00082084
DMAC2.DMCRA	0x00000000	IOR (4)	0x00082088
DMAC2.DMCRB	0x0000	IOR (2)	0x0008208c
DMAC2.DMTMD	0x0001	IOR (2)	0x00082090
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090...
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090...
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090.8
DMAC2.DMT...	0x0	IOR (2ビット)	0x00082090.8

実行後の DMAC3 は次の通りです。

転送先アドレスにおいては 13 バイト分のデータ転送によりインクリメントされ、2f00h から 2f0dh になりました。転送カウンタはデクリメントされるため、dh(13 回)から 0 になりました。



IOR	値	型情報(バイト数)	アドレス
DMAC3.DMSAR	0x0008a0a5	IOR (4)	
DMAC3.DMDAR	0x00002f0d	IOR (4)	
DMAC3.DMCRA	0x00000000	IOR (4)	0x000820c8
DMAC3.DMCRB	0x0000	IOR (2)	0x000820cc
DMAC3.DMTMD	0x0001	IOR (2)	0x000820d0
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0...
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0...
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0.8
DMAC3.DMT...	0x0	IOR (2ビット)	0x000820d0.8

以上により、DMAC は正常に機能していることが確認できました。

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