

【注意事項】

R20TS0591JS0100

Rev.1.00

2020.06.16

e² studio Smart Configurator プラグイン,

RX スマート・コンフィグレータ

概要

タイトルに記載している製品の使用上の注意事項を連絡します。

1. データトランスファコントローラ (DTC) のコンポーネントを使用しベクタベースアドレスを設定する場合の注意事項
2. SCI/SCIF 調歩同期式モードのコンポーネントを使用しビットレートの設定を行う場合の注意事項
3. S12AD のコンポーネントで AN007 または AN107 をアナログ入力端子として使用する場合の注意事項

1. データトランスファコントローラ (DTC) のコンポーネントを使用しベクタベースアドレスの設定を行う場合の注意事項

1.1 該当製品

- e² studio V6.2.0 (Smart Configurator プラグイン V1.3.0) 以降
- RX スマート・コンフィグレータ V1.3.0 以降

1.2 該当デバイス

- RX ファミリ :
RX230、RX231 グループ(RAM : 32K バイト製品のみ)
RX651、RX65N グループ(RAM : 640K バイト製品のみ)

1.3 内容

データトランスファコントローラ(DTC)のコンポーネントでベクタベースアドレスの設定を行う際、ベクタベースアドレスのアドレス範囲判定に誤りがあるため、エラー表示が正しく行われません。

- RX230、RX231 グループ(RAM : 32K バイト製品)の場合
正しいアドレス範囲 : 0x00000000~0x00007C00
誤ったアドレス範囲 : 0x00000000~0x0000FC00 (図 1.1 参照)
- RX651、RX65N グループ(RAM : 640K バイト製品)の場合
正しいアドレス範囲 : 0x00000000~0x0003FC00 および 0x00800000~0x0085FC00
誤ったアドレス範囲 : 0x00000000~0x0003FC00 のみ (図 1.2 参照)

基本設定
DTC0

転送情報リードスキップ
許可しない

アドレスモード
フルアドレスモード(32ビット)

DTC ベクタベースアドレス
0x0000FC00

正しいアドレス入力範囲は
'0x00000000 ~ 0x00007C00' ですが、
'0x00007C01~0x0000FC00' の値を設定しても
エラーが表示されません。

図 1.1 DTC ベクタベースアドレスを誤ったアドレス範囲で判定 (RX230、RX231 グループ)

基本設定
DTC0

転送情報リードスキップ
許可しない

アドレスモード
フルアドレスモード(32ビット)

DTC ベクタベースアドレス
0x0085EC00

'0x00800000 ~ 0x0085FC00' の値を設定すると
エラーが表示されます。

図 1.2 DTC ベクタベースアドレスを誤ったアドレス範囲で判定 (RX651、RX65N グループ)

1.4 回避策

- RX230、RX231 グループ(RAM : 32K バイト製品)の場合
DTC ベクタベースアドレスの値を手動で、0x00007C00 以下の値に修正してください。
- RX651、RX65N グループ(RAM : 640K バイト製品)の場合
入力したアドレスの値が '0x00800000 ~ 0x0085FC00' の範囲内であり、かつ 1K バイト単位以内の場合、DTC ベクタベースアドレスのテキストボックス横に表示されるエラーマークを無視してください。

1.5 恒久対策

以下のバージョンで改修予定です。(2020 年 7 月公開予定)

- e² studio 2020-07
- RX スマート・コンフィグレータ V2.6.0

2. SCI/SCIF 調歩同期式モードのコンポーネントを使用しビットレートの設定を行う場合の注意事項

2.1 該当製品

- e² studio V6.0.0 (Smart Configurator プラグイン V1.2.0) 以降
- RX スマート・コンフィグレータ V1.2.0 以降

2.2 該当デバイス

- RX ファミリ :
RX651、RX65N グループ

2.3 内容

SCI/SCIF 調歩同期式モードのコンポーネントにおいて、ビットレートの設定をテキストボックス入力で行う場合、入力値が設定範囲内であっても設定可能最小値の 8 倍より小さい場合、ビットレートを設定するコードが誤って生成されます。

発生例：SCI チャンネル 0 を使用し、ビットレートのテキストボックスに 500bps を入力した場合

入力値 500bps は、設定範囲内(114.441~7500000.0)ですが、設定可能最小値の 8 倍 ($8 \times 114.441 = 915.528$) より小さい値のため、ビットレートを設定するコードが誤って生成されます。

以下に、GUI 設定(図 2.1)と誤った生成コード(図 2.2)を記します。

転送速度設定	
転送クロック	内部クロック
基本クロック	1ビット期間の16サイクル
ビットレート	500
☐ ビットレートモジュレーション機能有効	
SCK0端子機能	SCKを使用しない
ノイズフィルタ設定	
☐ ノイズ除去機能を使用する	
ノイズフィルタクロック	1分周のクロック
	60000000 (Hz)

ビットレートのテキストボックス内をダブルクリックすると、入力範囲が出力コンソールに表示され（この場合、114.441 ~ 915.528 のいずれかの入力値）、誤ったビットレートのコードが生成されます。

図 2.1 SCI/SCIF 調歩同期式モードのビットレート GUI 設定

```

/*****
* Function Name: R_Config_SCI0_Create
* Description  : This function initializes the SCI0 channel
* Arguments    : None
* Return Value : None
*****/

void R_Config_SCI0_Create(void)
{
    /* Cancel SCI stop state */
    MSTP(SCI0) = 0U;
    .....

    /* Set control registers */
    SCI0.SMR.BYTE = _00_SCI_CLOCK_PCLK | _00_SCI_MULTI_PROCESSOR_DISABLE |
    _00_SCI_STOP_1 | _00_SCI_PARITY_DISABLE | _00_SCI_DATA_LENGTH_8 |
    _00_SCI_ASYNCHRONOUS_OR_I2C_MODE;
    .....

    SCI0.SEMR.BYTE = _03_SCI_CLOCK_PCLK_64 | _00_SCI_16_BASE_CLOCK |
    _00_SCI_NOISE_FILTER_DISABLE | _00_SCI_BAUDRATE_SINGLE |
    _00_SCI_LOW_LEVEL_START_BIT;

    /* Set bit rate */
    SCI0.BRR = 0x2EU;
    .....

    R_Config_SCI0_Create_UserInit();
}

```

誤った CKS 値
正しくは '03_SCI_CLOCK_PCLK_64'

誤った ABCS 値
正しくは '10_SCI_8_BASE_CLOCK'

誤った BRR 値
正しくは 'SCI0.BRR = 0x74U'

図 2.2 SCI/SCIF 調歩同期式モードのビットレートの生成コード

2.4 回避策

RX64M プロジェクト用 SCI/SCIF 調歩同期式モードのコンポーネントの生成コードを参照して、以下の手順に従ってください。

- (1) Smart Configurator 用 RX64M プロジェクトを作成し、SCI/SCIF 調歩同期式モードのコンポーネントを追加してください。リソースには、SCI0~7 または SCI12 のいずれかを設定してください。
- (2) 「転送速度設定」グループにある設定項目を Smart Configurator 用 RX651、RX65N プロジェクト^{*1}と同様に設定してください。

転送速度設定	
転送クロック	内部クロック
基本クロック	1ビット期間の16サイクル
ビットレート	500 (bps)
☐ ビットレートモジュレーション機能有効	
SCK0端子機能	SCKを使用しない

- (3) コード生成後、RX651、RX65N プロジェクトの初期化 API にある、SMR.CKS および SEMR.ABCS のマクロ値と、BRR レジスタ値をコピーし、それぞれ現在の値（図 2.2 参照）と置き換えてください。

*1 : RX651、RX65N、および RX64M プロジェクト用のクロックページ内 PCLK 周波数設定値が同じであることを確認してください。RX651、RX65N は SCI0~9 および SCI12 で PLCKB を使用、SCI10 および SCI11 で PCLKA を使用。RX64M は全チャンネルで PCLKB を使用。

2.5 恒久対策

以下のバージョンで改修予定です。(2020 年 7 月公開予定)

- e² studio 2020-07
- RX スマート・コンフィグレータ V2.6.0

3. S12AD コンポーネントで AN007 または AN107 をアナログ入力端子として使用する 場合の注意事項

3.1 該当製品

- e² studio V7.2.0 (Smart Configurator プラグイン V1.5.0) 以降
- RX スマート・コンフィグレータ V1.5.0 以降

3.2 該当デバイス

- RX ファミリ :
RX66T、RX72T グループ

3.3 内容

AN007 または AN107 をアナログ入力端子として使用する場合、PxDEN ビット*1は 0 にクリアされるべきですが正しいコードが生成されません。そのため、アナログ入力端子として使用することができません。

*1 : AN007 は x=000~002、AN107 は x=100~102

以下に、AN007 をアナログ入力端子として使用している場合の発生例を記載します。

なお、AN107 をご使用の場合は、以降 “AN007” を “AN107” に読み替えてください。

■ 発生例 1

・使用条件

項目		設定内容
AN001 および AN002	端子設定	アナログ入力端子
	アナログ入力パス	コンボボックスにある上位 4 つの選択肢のうちいずれか
アンプ入力		シングルエンド

図 3.1 発生例 1 の使用条件の GUI 設定

・生成コード

```

/*****
* Function Name: R_Config_S12AD0_Create
* Description  : This function initializes the S12AD0 channel
* Arguments    : None
* Return Value : None
*****/

void R_Config_S12AD0_Create(void)
{
    /* Cancel S12AD0 module stop state */
    MSTP(S12AD) = 0U;
    .....

    S12AD.ADANSA0.WORD = _0002_AD_ANx01_USED | _0004_AD_ANx02_USED |
_0080_AD_ANx07_USED;
    S12AD.ADADS0.WORD = _0080_AD_ANx07_USED;
    /* Set AN001 amplifier */
    S12AD.ADPGADC0.BIT.P001DEN = 0U;
    S12AD.ADPGACR.BIT.P001CR = _0001_AD_PATH_ANx_NONE_NONE;

    /* Set AN002 amplifier */
    S12AD.ADPGADC0.BIT.P002DEN = 0U;
    S12AD.ADPGACR.BIT.P002CR = _0001_AD_PATH_ANx_NONE_NONE;
    S12AD.ADCER.WORD = _0000_AD_AUTO_CLEARING_DISABLE |
_0000_AD_SELFTDIAGST_DISABLE | _0000_AD_RIGHT_ALIGNMENT;
    S12AD.ADELCCR.BYTE = _02_ALL_SCAN_COMPLETION;
    .....

    R_Config_S12AD0_Create_UserInit();
}

```

ここに P000DEN ビットをクリアするコード
'S12AD.ADPGADC0.BIT.P000DEN = 0U' が
ない

図 3.2 発生例 1 の生成コード

■ 発生例 2

・ 使用条件

項目		設定内容
AN001	端子設定	アナログ入力端子
	アナログ入力パス	コンボボックスにある上位 4 つの選択肢のうちいずれか
アンプ入力		シングルエンド

アナログ入力チャネル設定

☐ AN000 ☒ AN001 ☐ AN002 ☐ AN003 ☒ AN007

プログラマブルゲインアンプ設定

アンプ入力選択 (AN000~AN0) シングルエンド

AN000アナログ入力パスの選択 A/D_AN000 | CMPC00_なし | CMPC01_なし

アンプゲイン選択 x 4.000

AN001アナログ入力パスの選択 A/D_AN001 | CMPC10_なし | CMPC11_なし

アンプゲイン選択 x 4.000

AN002アナログ入力パスの選択 A/D_AN002 | CMPC20_なし | CMPC21_なし

アンプゲイン選択 x 4.000

図 3.3 発生例 2 の使用条件の GUI 設定

・ 生成コード

```

/*****
 * Function Name: R_Config_S12AD0_Create
 * Description  : This function initializes the S12AD0 channel
 * Arguments    : None
 * Return Value : None
 *****/

void R_Config_S12AD0_Create(void)
{
    /* Cancel S12AD0 module stop state */
    MSTP(S12AD) = 0U;
    .....

    S12AD.ADANSA0.WORD = _0002_AD_ANx01_USED | _0080_AD_ANx07_USED;
    S12AD.ADADS0.WORD = _0080_AD_ANx07_ADD_USED;

    /* Set AN001 amplifier */
    S12AD.ADPGADCR0.BIT.P001DEN = 0U;
    S12AD.ADPGACR.BIT.P001CR = _0001_...

    /* Set compare control register */
    S12AD.ADCMPCR.WORD = _0000_AD_WINDOWB_DISABLE | _0000_AD_WINDOWA_DISABLE |
    _0000_AD_WINDOWFUNCTION_DISABLE;
    .....

    R_Config_S12AD0_Create_UserInit();
}

```

図 3.4 発生例 2 の生成コード

■ 発生例 3

・ 使用条件

項目		設定内容
AN002	端子設定	アナログ入力端子
	アナログ入力パス	コンボボックスにある上位 4 つの選択肢のうちいずれか
アンプ入力		シングルエンド

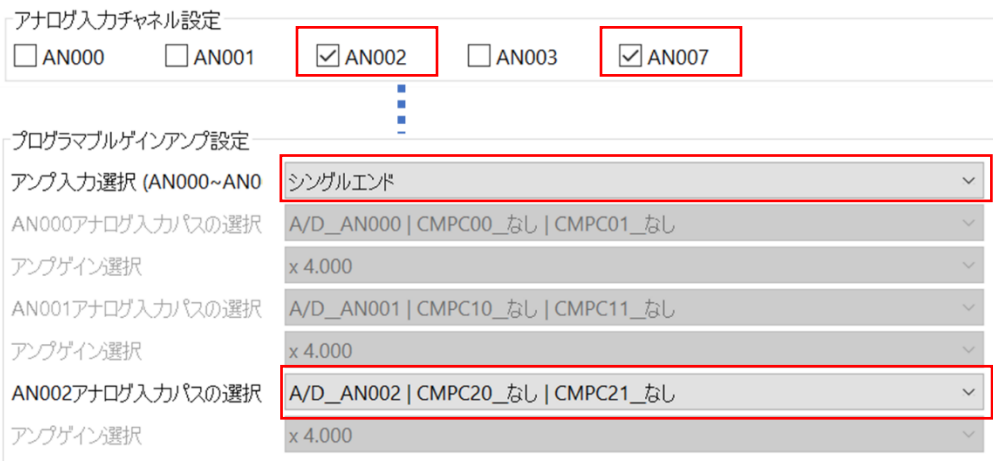


図 3.5 発生例 3 の使用条件の GUI 設定

・ 生成コード

```
/* *****
 * Function Name: R_Config_S12AD0_Create
 * Description  : This function initializes the S12AD0 channel
 * Arguments    : None
 * Return Value : None
 * ***** */

void R_Config_S12AD0_Create(void)
{
    /* Cancel S12AD0 module stop state */
    MSTP(S12AD) = 0U;
    .....

    S12AD.ADANSA0.WORD = _0004_AD_ANx02_USED | _0080_AD_ANx07_USED;
    S12AD.ADADS0.WORD = _0080_AD_ANx07_USED;

    /* Set AN002 amplifier */
    S12AD.ADPGADCR0.BIT.P002DEN = 0U;
    S12AD.ADPGACR.BIT.P002CR = _0001_AD_...
    S12AD.ADCER.WORD = _0000_AD_AUTO_CL...
    _0000_AD_SELFTESTDIAGST_DISABLE | _0000_AD_...
    S12AD.ADELCCR.BYTE = _02_ALL_SCAN_COMPLETION;
    S12AD.ADCSR.WORD |= _1000_AD_SCAN_END_INTERRUPT_ENABLE;
    .....

    R_Config_S12AD0_Create_UserInit();
}
```

図 3.6 発生例 3 の生成コード

3.4 回避策

AN007 または AN107 をアナログ入力端子として使用する場合、PxDEN ビット^{*1}をクリアするためのコード 'S12AD.ADPGADCR0.BIT.PxDEN = 0U' を生成ファイルに手動で追加してください。

*1 : AN007 は x=000~002、AN107 は x=100~102

- ・ ソースファイル : "<コンフィグレーション名>.c"
- ・ 関数 : "void R_<コンフィグレーション名>_Create(void)"

<コンフィグレーション名>は設定する S12AD のコンポーネントにより異なります。

注意: 生成コードは、再度コード生成を行うと修正前の状態に戻りますので、ソースファイルの修正はコード生成を行う度に実施してください。

以下に<コンフィグレーション名>が Config_S12AD0（初期値）の場合の修正例を記します。

■ 3.3 項の発生例 1 の修正例

```

/*****
* Function Name: R_Config_S12AD0_Create
* Description   : This function initializes the S12AD0 channel
* Arguments     : None
* Return Value  : None
*****/

void R_Config_S12AD0_Create(void)
{
    /* Cancel S12AD0 module stop state */
    MSTP(S12AD) = 0U;
    .....

    S12AD.ADANSA0.WORD = _0002_AD_ANx01_USED | _0004_AD_ANx02_USED |
_0080_AD_ANx07_USED;
    S12AD.ADADS0.WORD = _0080_AD_ANx07_ADD_USED;

    S12AD.ADPGADCR0.BIT.P000DEN = 0U;

    /* Set AN001 amplifier */
    S12AD.ADPGADCR0.BIT.P001DEN = 0U;
    S12AD.ADPGACR.BIT.P001CR = _0001_AD_PATH_ANx_NONE_NONE;

    /* Set AN002 amplifier */
    S12AD.ADPGADCR0.BIT.P002DEN = 0U;
    S12AD.ADPGACR.BIT.P002CR = _0001_AD_PATH_ANx_NONE_NONE;
    S12AD.ADCER.WORD = _0000_AD_AUTO_CLEARING_DISABLE |
_0000_AD_SELFTESTDIAGST_DISABLE | _0000_AD_RIGHT_ALIGNMENT;
    S12AD.ADELCCR.BYTE = _02_ALL_SCAN_COMPLETION;
    .....

    R_Config_S12AD0_Create_UserInit();
}

```

図 3.7 3.3 項の発生例 1 の修正コード

■ 3.3 項の発生例 2 の修正例

```

/*****
* Function Name: R_Config_S12AD0_Create
* Description  : This function initializes the S12AD0 channel
* Arguments    : None
* Return Value : None
*****/

void R_Config_S12AD0_Create(void)
{
    /* Cancel S12AD0 module stop state */
    MSTP(S12AD) = 0U;
    .....

    S12AD.ADANSA0.WORD = _0002_AD_ANx01_USED | _0080_AD_ANx07_USED;
    S12AD.ADADS0.WORD = _0080_AD_ANx07_ADD_USED;

    S12AD.ADPGADCR0.BIT.P000DEN = 0U;

    /* Set AN001 amplifier */
    S12AD.ADPGADCR0.BIT.P001DEN = 0U;
    S12AD.ADPGACR.BIT.P001CR = _0001_AD_PATH_ANx_NONE_NONE;

    S12AD.ADPGADCR0.BIT.P002DEN = 0U;

    /* Set compare control register */
    S12AD.ADCMPCR.WORD = _0000_AD_WINDOWB_DISABLE | _0000_AD_WINDOWA_DISABLE |
    _0000_AD_WINDOWFUNCTION_DISABLE;
    .....

    R_Config_S12AD0_Create_UserInit();
}

```

P000DEN ビットをクリアするコードを追加

P002DEN ビットをクリアするコードを追加

図 3.8 3.3 項の発生例 2 の修正コード

■ 3.3 項の発生例 3 の修正例

```

/*****
* Function Name: R_Config_S12AD0_Create
* Description   : This function initializes the S12AD0 channel
* Arguments     : None
* Return Value  : None
*****/

void R_Config_S12AD0_Create(void)
{
    /* Cancel S12AD0 module stop state */
    MSTP(S12AD) = 0U;
    .....

    S12AD.ADANSA0.WORD = _0004_AD_ANx02_USED | _0080_AD_ANx07_USED;
    S12AD.ADADS0.WORD = _0080_AD_ANx07_ADD_USED;

    S12AD.ADPGADCR0.BIT.P000DEN = 0U;
    S12AD.ADPGADCR0.BIT.P001DEN = 0U;

    /* Set AN002 amplifier */
    S12AD.ADPGADCR0.BIT.P002DEN = 0U;
    S12AD.ADPGACR.BIT.P002CR = _0001_AD_PATH_ANx_NONE_NONE;
    S12AD.ADCER.WORD = _0000_AD_AUTO_CLEARING_DISABLE |
_0000_AD_SELFTDIAGST_DISABLE | _0000_AD_RIGHT_ALIGNMENT;
    S12AD.ADELCCR.BYTE = _02_ALL_SCAN_COMPLETION;
    S12AD.ADCSR.WORD |= _1000_AD_SCAN_END_INTERRUPT_ENABLE;
    .....

    R_Config_S12AD0_Create_UserInit();
}

```

P000DENP000DEN ビットをクリアするコードを追加

図 3.9 3.3 項の発生例 3 の修正コード

3.5 恒久対策

以下のバージョンで改修予定です。(2020 年 7 月公開予定)

- e² studio 2020-07
- RX スマート・コンフィグレータ V2.6.0

以上

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Jun.16.20	-	新規発行

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。

過去のニュース内容は発行当時の情報をもとにしており、現時点では変更された情報や無効な情報が含まれている場合があります。

ニュース本文中の URL を予告なしに変更または中止することがありますので、あらかじめご承知ください。

本社所在地

〒135-0061 東京都江東区豊洲 3-2-24（豊洲フォレシア）

www.renesas.com

お問合せ窓口

弊社の製品や技術、ドキュメントの最新情報、最寄の営業お問合せ窓口に関する情報などは、弊社ウェブサイトをご覧ください。

www.renesas.com/contact/

商標について

ルネサスおよびルネサスロゴはルネサス エレクトロニクス株式会社の商標です。すべての商標および登録商標は、それぞれの所有者に帰属します。

© 2020 Renesas Electronics Corporation. All rights reserved.

TS Colophon 4.1