【注意事項】
RH850 スマート・コンフィグレータ

R20TS0407JJ0100
Rev.1.00
2019.03.16 号

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. クロックを生成しない設定時にビルドエラーが発生する注意事項
2. RAM サイズの表示ミスの注意事項
3. 割り込み優先度の表示ミスの注意事項

1. クロックを生成しない設定時にビルドエラーが発生する注意事項

1.1 該当製品

RH850 スマート・コンフィグレータ V1.0.0

1.2 該当 MCU

RH850 ファミリ : RH850/F1KM グループ

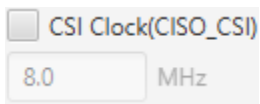
1.3 内容

1.2 項の該当 MCU で「クロックを出力する設定」を行った場合は正しいコードが生成されますが、以下のいずれかのクロックにおいて「クロックを出力しない設定」を行った場合、マクロ値が正しく生成されないためビルド時にビルドエラーが発生します。

- TAUJ クロック(C_AWO_TAUJ)
- ADCA0 クロック(C_AWO_ADCA)
- RTCA クロック(C_AWO_RTCA)
- RLIN クロック(C_ISO_LIN)
- RS-CANFD クロック(C_ISO_CAN)
- CSI クロック(C_ISO_CSI)

「クロックを出力する設定」

「クロックを出力しない設定」



■ 発生例

以下に、ビルドエラーの発生例を記します。

r_cg_cgc.c の関数 void R_CGC_Create(void) に以下のコードが生成されて、赤文字の箇所でビルドエラーが発生します。

- TAUJ クロック(C_AWO_TAUJ)の場合

```
void R_CGC_Create(void)
{
    ...
    /* TAUJ0 clock domain setting */
    WPROTR.PROTCMD0 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ATAUJS_CTL = _;
    CLKCTL.CKSC_ATAUJS_CTL = (uint32_t) ~_;
    CLKCTL.CKSC_ATAUJS_CTL = _;
    while (CLKCTL.CKSC_ATAUJS_ACT != _)
   {
        NOP();
    }
    ...
}
```

- ADCA0 クロック(C_AWO_ADCA)の場合

```
void R_CGC_Create(void)
{
    ...
    /* ADCA0 clock domain setting */
    WPROTR.PROTCMD0 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_AADCAS_CTL = _;
    CLKCTL.CKSC_AADCAS_CTL = (uint32_t) ~_;
    CLKCTL.CKSC_AADCAS_CTL = _;
    while (CLKCTL.CKSC_AADCAS_ACT != _)
    {
        NOP();
    }
    ...
}
```

- RTCA クロック(C_AWO_RTCA)の場合

```
void R_CGC_Create(void)
{
    ...
    /* RTCA0 clock domain setting */
    WPROTR.PROTCMD0 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ARTCAS_CTL = _;
    CLKCTL.CKSC_ARTCAS_CTL = (uint32_t) ~_;
    CLKCTL.CKSC_ARTCAS_CTL = _;
    while (CLKCTL.CKSC_ARTCAS_ACT != _)
    {
        NOP();
    }
    ...
}
```

- RLIN クロック(C_ISO_LIN)の場合

```
void R_CGC_Create(void)
{
    ...
    /* RLIN clock domain setting */
    WPROTR.PROTCMD1 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ILINS_CTL = _;
    CLKCTL.CKSC_ILINS_CTL = (uint32_t) ~_;
    CLKCTL.CKSC_ILINS_CTL = _;
    while (CLKCTL.CKSC_ILINS_ACT != _)
    {
        NOP();
    }
    ...
}
```

- RS-CANFD クロック(C_ISO_CAN)の場合

```
void R_CGC_Create(void)
{
    ...
    /* RS-CANn clock domains setting */
    WPROTR.PROTCMD1 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ICANS_CTL = _;
    CLKCTL.CKSC_ICANS_CTL = (uint32_t) ~_;
    CLKCTL.CKSC_ICANS_CTL = _;
    while (CLKCTL.CKSC_ICANS_ACT != _)
    {
        NOP();
    }
    ...
}
```

- CSI クロック(C_ISO_CSI)の場合

```
void R_CGC_Create(void)
{
    ...
    /* CSI clock domain setting */
    WPROTR.PROTCMD1 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ICSIS_CTL = _;
    CLKCTL.CKSC_ICSIS_CTL = (uint32_t) ~_;
    CLKCTL.CKSC_ICSIS_CTL = _;
    while (CLKCTL.CKSC_ICSIS_ACT != _)
    {
        NOP();
    }
    ...
}
```

1.4 回避策

r_cg_cgc.c の関数 void R_CGC_Create(void) の生成コードを手動で修正してください(注)。

以下に修正例を記します。赤文字の部分が修正内容です。

注：コード生成を実行するたびに修正が必要です。

- TAUJ クロック(C_AWO_TAUJ)の場合

```
void R_CGC_Create(void)
{
    ...
    /* TAUJ0 clock domain setting */
    WPROTR.PROTCMD0 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ATAUJS_CTL = _CGC_TAUJ_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ATAUJS_CTL = (uint32_t) ~_CGC_TAUJ_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ATAUJS_CTL = _CGC_TAUJ_CLK_SOURCE_DISABLE;
    while (CLKCTL.CKSC_ATAUJS_ACT != _CGC_TAUJ_CLK_SOURCE_DISABLE)
    {
        NOP();
    }
    ...
}
```

- ADCA0 クロック(C_AWO_ADCA)の場合

```
void R_CGC_Create(void)
{
    ...
    /* ADCA0 clock domain setting */
    WPROTR.PROTCMD0 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_AADCAS_CTL = _CGC_ADCA0_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_AADCAS_CTL = (uint32_t) ~_CGC_ADCA0_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_AADCAS_CTL = _CGC_ADCA0_CLK_SOURCE_DISABLE;
    while (CLKCTL.CKSC_AADCAS_ACT != _CGC_ADCA0_CLK_SOURCE_DISABLE)
    {
        NOP();
    }
    ...
}
```

- RTCA クロック(C_AWO_RTCA)の場合

```
void R_CGC_Create(void)
{
    ...
    /* RTCA0 clock domain setting */
    WPROTR.PROTCMD0 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ARTCAS_CTL = _CGC_RTCA_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ARTCAS_CTL = (uint32_t) ~_CGC_RTCA_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ARTCAS_CTL = _CGC_RTCA_CLK_SOURCE_DISABLE;
    while (CLKCTL.CKSC_ARTCAS_ACT != _CGC_RTCA_CLK_SOURCE_DISABLE)
    {
        NOP();
    }
    ...
}
```

- RLIN クロック(C_ISO_LIN)の場合

```
void R_CGC_Create(void)
{
    ...
    /* RLIN clock domain setting */
    WPROTR.PROTCMD1 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ILINS_CTL = _CGC_RLIN_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ILINS_CTL = (uint32_t) ~_CGC_RLIN_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ILINS_CTL = _CGC_RLIN_CLK_SOURCE_DISABLE;
    while (CLKCTL.CKSC_ILINS_ACT != _CGC_RLIN_CLK_SOURCE_DISABLE)
    {
        NOP();
    }
    ...
}
```

- RS-CANFD クロック(C_ISO_CAN)の場合

```
void R_CGC_Create(void)
{
    ...
    /* RS-CANn clock domains setting */
    WPROTR.PROTCMD1 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ICANS_CTL = _CGC_RSCAN_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ICANS_CTL = (uint32_t) ~_CGC_RSCAN_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ICANS_CTL = _CGC_RSCAN_CLK_SOURCE_DISABLE;
    while (CLKCTL.CKSC_ICANS_ACT != _CGC_RSCAN_CLK_SOURCE_DISABLE)
    {
        NOP();
    }
    ...
}
```

- CSI クロック(C_ISO_CSI)の場合

```
void R_CGC_Create(void)
{
    ...
    /* CSI clock domain setting */
    WPROTR.PROTCMD1 = _WRITE_PROTECT_COMMAND;
    CLKCTL.CKSC_ICSIS_CTL = _CGC_CSI_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ICSIS_CTL = (uint32_t) ~_CGC_CSI_CLK_SOURCE_DISABLE;
    CLKCTL.CKSC_ICSIS_CTL = _CGC_CSI_CLK_SOURCE_DISABLE;
    while (CLKCTL.CKSC_ICSIS_ACT != _CGC_CSI_CLK_SOURCE_DISABLE)
    {
        NOP();
    }
    ...
}
```

1.5 恒久対策

次期バージョンで改修予定です。

2. RAM サイズの表示ミスの注意事項

2.1 該当製品

RH850 スマート・コンフィグレータ V1.0.0

2.2 該当 MCU

RH850 ファミリ : RH850/F1KM グループ
R7F701685, R7F701686, R7F701688, R7F701689, R7F701691, R7F701692,
R7F701694, R7F701695

2.3 内容

下記の表示において、該当 MCU の RAM サイズを正しく表示せず、常に 96KB と表示します。

[概要] – [現在の設定状態]

- 生成したレポート・ファイルを表示

```
================================================================================↓
プロジェクト: R7F701692↓
日付: 2019-02-13↓
================================================================================↓
↓
1 ボード設定↓
  ボード: null↓
↓
  デバイス: R7F701692 (ROM size: 512 KB , RAM size: 96 KB , Pin count: 64)↓
↓
```

2.4 回避策

「RAM size」の値を以下に読み替えてください。

- R7F701685, R7F701688, R7F701691, R7F701694 の場合 : 64KB
- R7F701686, R7F701689, R7F701692, R7F701695 の場合 : 32KB

2.5 恒久対策

次期バージョンで改修予定です。

3. 割り込み優先度の表示ミスの注意事項

3.1 該当製品

RH850 スマート・コンフィグレータ V1.0.0

3.2 該当 MCU

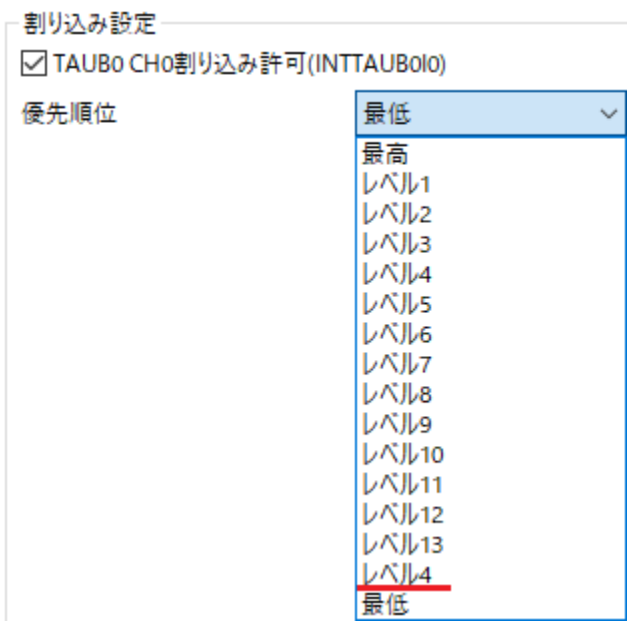
RH850 ファミリ : RH850/F1KM グループ

3.3 内容

以下の周辺機能の“割り込み設定”画面において、割り込み設定の優先順位の「レベル 14」が正しく表示されず「レベル 4」と表示します。なお、生成されるコードは正しい設定（レベル 14）です。

- クロック分周
- 入力インターバルタイマ
- 入力期間カウント検出
- 入力位置検出
- 入力パルスインターバル判定
- 入力パルスインターバル測定
- 入力信号幅判定
- 入力信号幅測定
- インターバルタイマ
- PWM 出力
- 三角波 PWM 出力

■ 誤った表示の画面例



3.4　回避策

「レベル 13」の下に表示される「レベル 4」を「レベル 14」に読み替えてください。

3.5　恒久対策

次期バージョンで改修予定です。

以上

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Mar.16.19	-	新規発行

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/