[Notes]
RH850 Smart Configurator

R20TS0407EJ0100
Rev.1.00
Mar. 16, 2019

Outline

When using the product in the title, note the following points.

1. Build error occurs when setting not to generate clocks
2. RAM size display error

1. Build Error Occurs When Setting Not to Generate Clocks

1.1 Applicable Products

RH850 Smart Configurator V1.0.0

1.2 Applicable MCUs

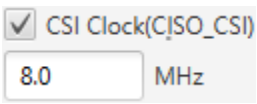
RH850 family: RH850/F1KM group

1.3 Details

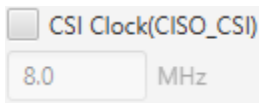
When the clocks are output for the applicable MCUs listed in section 1.2, the code is generated correctly. However, if you specify not to output any of the following clocks, a build error occurs because a macro value is not generated correctly.

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

"Output Clock"



"Not Output Clock"



■ Examples

The examples of the build errors are described below.

The following code is generated for a function void R_CGC_Create(void) in the r_cg_cgc.c file. The build error occurs at the points indicated in red.

- In the case of TAUJ clock (C_AWO_TAUJ)

```
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();
    }
    ...
}
```

- In the case of ADCA0 clock (C_AWO_ADCA)

```
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();
    }
    ...
}
```

- In the case of RTCA clock (C_AWO_RTCA)

```
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();
    }
    ...
}
```

- In the case of RLIN clock (C_ISO_LIN)

```
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();
    }
    ...
}
```

- In the case of RS-CANFD clock (C_ISO_CAN)

```
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();
    }
    ...
}
```

- In the case of CSI clock (C_ISO_CSI)

```
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 Workaround

Modify the generated code of the function void R_CGC_Create(void) in the r_cg_cgc.c file manually (Note).

Examples of the modification are described below. Corrections are indicated in red.

Note: Code needs to be modified every time you generate the code.

- In the case of TAUJ clock (C_AWO_TAUJ)

```
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();
    }
    ...
}
```

- In the case of ADCA0 clock (C_AWO_ADCA)

```
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();
    }
    ...
}
```

- In the case of RTCA clock (C_AWO_RTCA)

```
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();
    }
    ...
}
```

- In the case of RLIN clock (C_ISO_LIN)

```
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();
    }
    ...
}
```

- In the case of RS-CANFD clock (C_ISO_CAN)

```
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();
    }
    ...
}
```

- In the case of CSI clock (C_ISO_CSI)

```
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 Schedule for Fixing the Problem

This problem will be fixed in the next version.

2. RAM Size Display Error

2.1 Applicable Products

RH850 Smart Configurator V1.0.0

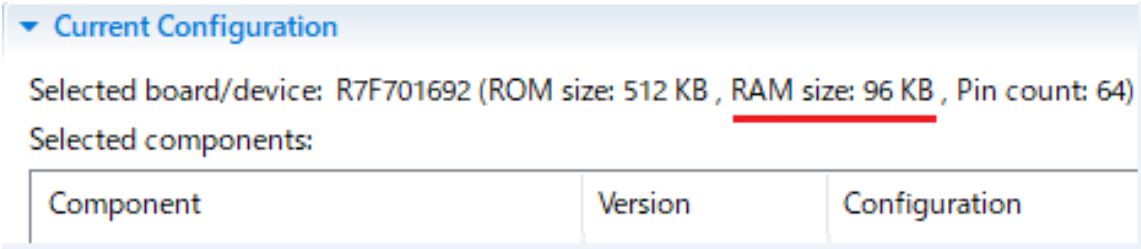
2.2 Applicable MCUs

RH850 family: RH850/F1KM group
R7F701685, R7F701686, R7F701688, R7F701689, R7F701691, R7F701692, R7F701694, R7F701695

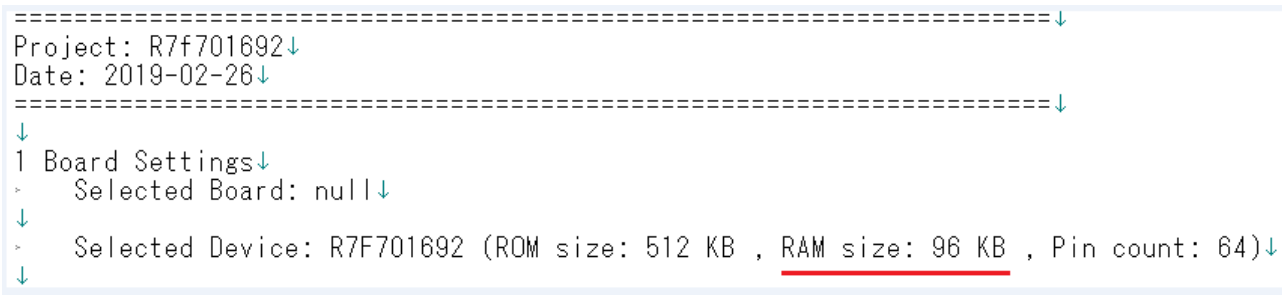
2.3 Details

In the following display, the RAM size of the applicable MCU is not displayed correctly and is always displayed as 96 KB.

[Overview] – [Current Configuration]



- Display of generated report file



2.4 Workaround

Interpret the value of "RAM size" as follows:

- In the case of R7F701685, R7F701688, R7F701691, R7F701694: 64 KB
- In the case of R7F701686, R7F701689, R7F701692, R7F701695: 32KB

2.5 Schedule for Fixing the Problem

This problem will be fixed in the next version.

Revision History

Rev.	Date	Description	
		Page	Summary
1.00	Mar.16.19	-	First edition issued

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Trademarks

Contact information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:
www.renesas.com/contact/.

TS Colophon 4.0