RX ファミリ

ユニーク ID リードモジュール Firmware Integration Technology

要旨

本アプリケーションノートは、Firmware Integration Technology (FIT)を使用したユニーク ID リードモジュール(以下、UID)について説明します。本モジュールはエクストラ領域に格納されている 32 バイトのユニーク ID を読み出し、指定の領域へ格納します。以降、本モジュールをユニーク ID リード (UID)FIT モジュールと称します。

対象デバイス

- RX110 グループ
- RX111 グループ
- RX113 グループ
- RX130 グループ
- RX13T グループ

本アプリケーションノートを他のマイコンへ適用する場合、そのマイコンの仕様にあわせて変更し、十分評価してください。

対象コンパイラ

- Renesas Electronics C/C++ Compiler Package for RX Family
- GCC for Renesas RX
- IAR C/C++ Compiler for Renesas RX

各コンパイラの動作確認内容については 4.1 動作確認環境を参照してください。

関連ドキュメント

ボードサポートパッケージモジュール Firmware Integration Technology (R01AN1685)

目次

1. 概要

1.1 ユニーク ID リード FIT モジュールとは

本モジュールは API として、プロジェクトに組み込んで使用します。本モジュールの組み込み方については、「2.9 FIT モジュールの追加方法」を参照してください。

1.2 ユニーク ID FIT モジュールの概要

ユニーク ID リード (UID) FIT モジュールは、エクストラ領域に格納されている 32 バイトのユニーク ID を読み出し、指定の領域に格納する API 関数を提供します。

本モジュールは、フラッシュメモリのセルフプログラミングの機能を利用し、エクストラ領域からユニーク ID を読み出します。ユニーク ID の読み出しは ROM P/E モードに移行して行います。ROM P/E モード中は、ROM の値を読み出せなくなるため、RAM にプログラムを転送して RAM でプログラムを実行します。ユニーク ID を読み出し後は、ROM に戻ります。

1.2.1 ユニーク ID の読み出し手順

ユニーク ID の読み出しは以下の手順で行います。

(1) R_UID_Open() 関数のコール
  (1-1) ROM から RAM へプログラムを転送

(2) R_UID_Read() 関数のコール
  (2-1) RAM にジャンプ
  (2-2) P/E モードに移行　　→ ROM の読み出し不可
  (2-3) ユニーク ID リードコマンドによるユニーク ID の読み出し
  (2-4) リードモードに移行　　→ ROM の読み出し可能
  (2-5) ROM にジャンプ

1.2.2　プログラムの配置セクション

本モジュールは、RAM で実行するプログラムを配置するセクションを用意しています。また、RAM に転送するプログラムを PFRAM セクションに配置する設定にしています。

表 1.1 にプログラムソースの配置セクションを示します。

表 1.1　プログラムソースの配置セクション

ソースファイル	関数	配置セクション
r_uid_rx.c	R_UID_Open R_UID_Read R_UID_GetVersion uid_codecopy	P セクション
r_uid_ram.c	uid_read_ram uid_pe_mode_enter uid_pe_mode_exit uid_read uid_write_fpmcr uid_enable_dataflashaccess uid_delay uid_delay_us	PFRAM セクション

1.3　API の概要

表 1.2 に本モジュールに含まれる API 関数を示します。

表 1.2　API 関数一覧

関数	関数説明
R_UID_Open()	ROM から RAM にユニーク ID を読み出すためのプログラムを転送し、本モジュールが使用できる状態にします。
R_UID_Read()	エクストラ領域からユニーク ID を読み出します。
R_UID_GetVersion()	本モジュールのバージョン番号を返します。

1.4　処理例

図 1.1 に本モジュールの処理例を示します。

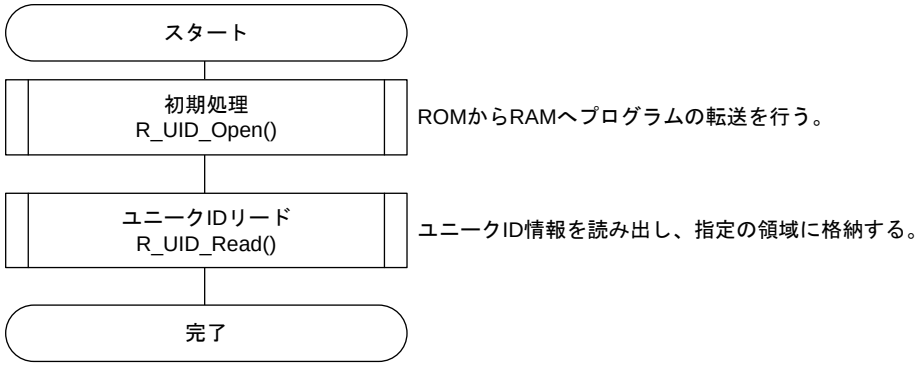


図 1.1　ユニーク ID リード (UID) FIT モジュールの処理例

1.5　制限事項

本モジュールには、以下の制限事項があります。

- 読み出したユニーク ID の格納先には ROM、E2 データフラッシュの領域を指定できません。

2. API 情報

本 FIT モジュールは、下記の条件で動作を確認しています。

2.1 ハードウェアの要求

ご使用になる MCU が以下の機能をサポートしている必要があります。

- ユニーク ID リード
- FlashIF クロック(FCLK)が 1MHz 以上
- FlashIF クロック(FCLK)が 4MHz 未満の場合、FCLK に設定可能な周波数は 1MHz、2MHz、3MHz (1MHz から 4MHz 未満において小数点のある周波数は使用不可)

2.2 ソフトウェアの要求

このドライバは以下の FIT モジュールに依存しています。

- ボードサポートパッケージ (r_bsp) v5.00 以上

2.3 サポートされているツールチェーン

本 FIT モジュールは「4.1 動作確認環境」に示すツールチェーンで動作確認を行っています。

2.4 ヘッダファイル

すべての API 呼び出しとそれをサポートするインタフェース定義は r_uid_rx_if.h に記載しています。

2.5 整数型

このドライバは ANSI C99 を使用しています。これらの型は stdint.h で定義されています。

2.6 コンパイル時の設定

本モジュールにはコンフィギュレーションオプションの設定はありません。
(r_uid_rx_config.h　<コンフィギュレーションヘッダファイル> はありません。)

2.7　コードサイズ

本モジュールの ROM サイズ、RAM サイズ、最大使用スタックサイズを下表に示します。

下表の値は下記条件で確認しています。

モジュールリビジョン: r_uid_rx rev1.11

コンパイラバージョン: Renesas Electronics C/C++ Compiler Package for RX Family V3.01.00
(統合開発環境のデフォルト設定に”-lang = c99”オプションを追加)
GCC for Renesas RX 4.8.4.201801
(統合開発環境のデフォルト設定に” -std=gnu99”オプションを追加)
IAR C/C++ Compiler for Renesas RX version 4.10.1
(統合開発環境のデフォルト設定)

コンフィグレーションオプション: デフォルト設定

ROM、RAM およびスタックのコードサイズ

デバイス	分類	使用メモリ		
		Renesas Compiler	GCC	IAR Compiler
RX110	ROM	**5613 バイト**	**8152 バイト**	**4065 バイト**
	RAM	**3294 バイト**	**3484 バイト**	**1982 バイト**
	スタック (※1)	**196 バイト**	**-**	**136 バイト**
RX111	ROM	**5846 バイト**	**8564 バイト**	**4417 バイト**
	RAM	**3354 バイト**	**3576 バイト**	**2043 バイト**
	スタック (※1)	**196 バイト**	**-**	**136 バイト**
RX113	ROM	**5881 バイト**	**8529 バイト**	**4469 バイト**
	RAM	**3438 バイト**	**3660 バイト**	**2127 バイト**
	スタック (※1)	**196 バイト**	**-**	**136 バイト**
RX130	ROM	**5852 バイト**	**8424 バイト**	**4465 バイト**
	RAM	**3414 バイト**	**3636 バイト**	**2103 バイト**
	スタック (※1)	**196 バイト**	**-**	**136 バイト**

注 1　割り込み関数の最大使用スタックサイズを含みます。

注 2　BSP を含んだサイズです。

2.8　戻り値

API 関数の戻り値を示します。この列挙型は、API 関数のプロトタイプ宣言とともに r_uid_rx_if.h で記載されています。

```
typedef enum
{
    UID_SUCCESS = 0,          /* R_UID_Open 関数が正常に終了した場合                          */
                              /* R_UID_Read 関数の実行でユニーク ID を正常に読み出せた場合  */
    UID_ERR_UNINITIALIZED,    /* R_UID_Open 関数の実行前に R_UID_Read 関数を実行した場合  */
    UID_ERR_LOCK_FUNC,        /* R_UID_Open 関数または R_UID_Read 関数実行中に、
                                 R_UID_Open 関数または R_UID_Read 関数が実行された場合    */
    UID_ERR_FAILURE,          /* R_UID_Open 関数を 2 回以上実行した場合                       */
                              /* R_UID_Read 関数の実行でユニーク ID の読み出しに失敗した場合*/
} uid_err_t;
```

2.9　FIT モジュールの追加方法

本モジュールは、使用するプロジェクトごとに追加する必要があります。ルネサスでは、スマート・コンフィグレータを使用した(1)、(3)、(5)の追加方法を推奨しています。ただし、スマート・コンフィグレータは、一部の RX デバイスのみサポートしています。サポートされていない RX デバイスについては(2)、(4)の方法を使用してください。

(1) e^2 studio 上でスマート・コンフィグレータを使用して FIT モジュールを追加する場合
e^2 studio のスマート・コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX スマート・コンフィグレータ ユーザーガイド: e^2 studio 編 (R20AN0451)」を参照してください。

(2) e^2 studio 上で FIT コンフィグレータを使用して FIT モジュールを追加する場合
e^2 studio の FIT コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加することができます。詳細は、アプリケーションノート「RX ファミリ e^2 studio に組み込む方法 Firmware Integration Technology (R01AN1723)」を参照してください。

(3) CS+上でスマート・コンフィグレータを使用して FIT モジュールを追加する場合
CS+上で、スタンドアロン版スマート・コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX スマート・コンフィグレータ ユーザーガイド: CS+編 (R20AN0470)」を参照してください。

(4) CS+上で FIT モジュールを追加する場合
CS+上で、手動でユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826)」を参照してください。

(5) IAREW 上でスマート・コンフィグレータを使用して FIT モジュールを追加する場合
スタンドアロン版スマート・コンフィグレータを使用して、自動的にユーザプロジェクトに FIT モジュールを追加します。詳細は、アプリケーションノート「RX スマート・コンフィグレータ ユーザーガイド: IAREW 編 (R20AN0535)」を参照してください。

2.10　for 文、while 文、do while 文について

本モジュールでは、レジスタの反映待ち処理等で for 文、while 文、do while 文（ループ処理）を使用しています。これらループ処理には、「WAIT_LOOP」をキーワードとしたコメントを記述しています。そのため、ループ処理にユーザがフェイルセーフの処理を組み込む場合は、「WAIT_LOOP」で該当の処理を検索できます。

以下に記述例を示します。

```
while 文の例：
/* WAIT_LOOP */
while(0 == SYSTEM.OSCOVFSR.BIT.PLOVF)
{
    /* The delay period needed is to make sure that the PLL has stabilized. */
}

for 文の例：
/* Initialize reference counters to 0. */
/* WAIT_LOOP */
for (i = 0; i < BSP_REG_PROTECT_TOTAL_ITEMS; i++)
{
    g_protect_counters[i] = 0;
}

do while 文の例：
/* Reset completion waiting */
do
{
    reg = phy_read(ether_channel, PHY_REG_CONTROL);
    count++;
} while ((reg & PHY_CONTROL_RESET) && (count < ETHER_CFG_PHY_DELAY_RESET)); /* WAIT_LOOP */
```

2.11　プロジェクトの設定

ユニーク ID の読み出しは ROM P/E モードに移行して行います。ROM P/E モード中は ROM 上でプログラムを実行できません。ユニーク ID を読み出すためには、本モジュールの一部プログラムを RAM に転送し、RAM で実行する必要があります。

コンパイラによって設定が異なりますので、使用するコンパイラに応じて設定してください。

2.11.1　Renesas Electronics C/C++ Compiler Package for RX Family

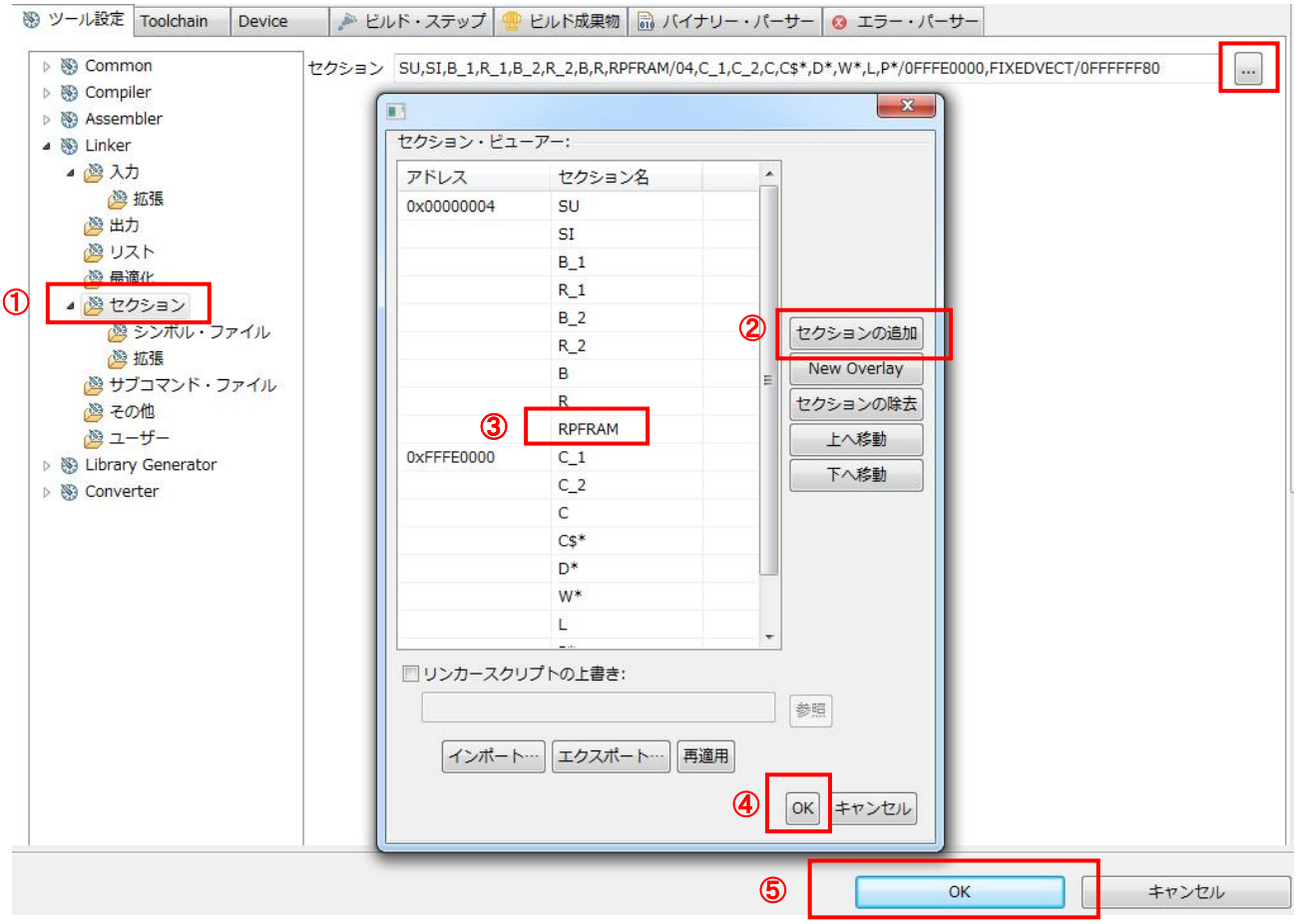
e^2 studio のリンカの設定で、以下の設定をしてください。

- セクションの追加
- ROM から RAM へマップするセクションを指定

セクションの追加

RAM に RPFRAM セクションを追加

① 「セクション」をクリック
② 「セクションの追加」をクリック
③ 「RPFRAM」を入力
④ 「OK」をクリック
⑤ 「OK」をクリック



ROM から RAM へマップするセクションを指定

“PFRAM=RPFRAM”を追加

①「シンボル・ファイル」をクリック

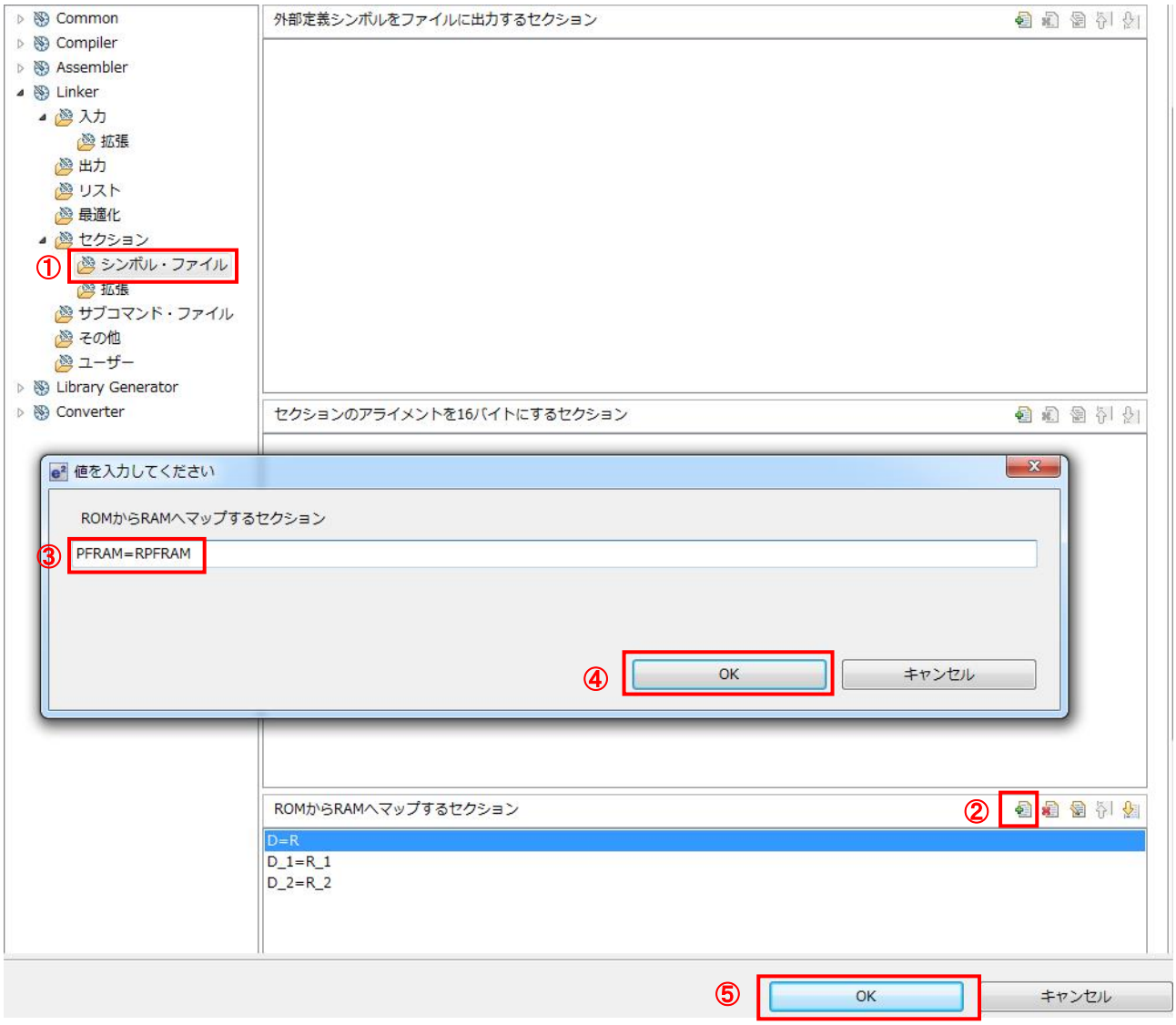
②「追加」をクリック

③“PFRAM=RPFRAM”を入力

④「OK」をクリック

⑤「OK」をクリック

RAM に転送するプログラムは、“R_BSP_ATTRIB_SECTION_CHANGE(P,FRAM)”を設定しており、PFRAM セクションに配置されます。この設定を行うと、RPFRAM セクション(RAM)に PFRAM セクション(ROM)のアドレスをマッピングすることができます。



2.11.2　GCC for Renesas RX

linker_script.ld ファイルを編集して、セクション及びシンボルの追加をしてください。

セクション及びシンボルの追加

下記 3 つのコードを追加

(a)

```
            . += _edata - _data;
```

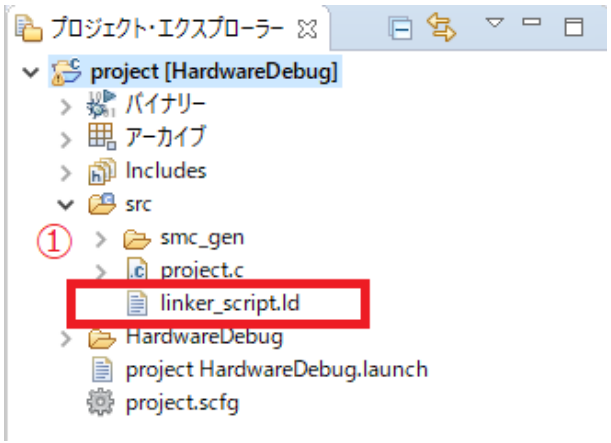
(b)

```
      .pfram ALIGN(4):
      {
            _PFRAM_start = .;
            . += _RPFRAM_end - _RPFRAM_start;
            _PFRAM_end = .;
      } > ROM
```

(c)

```
      .rpfram ALIGN(4): AT(_PFRAM_start)
      {
            _RPFRAM_start = .;
            *(PFRAM)
            . = ALIGN(4);
            _RPFRAM_end = .;
      } > RAM
```

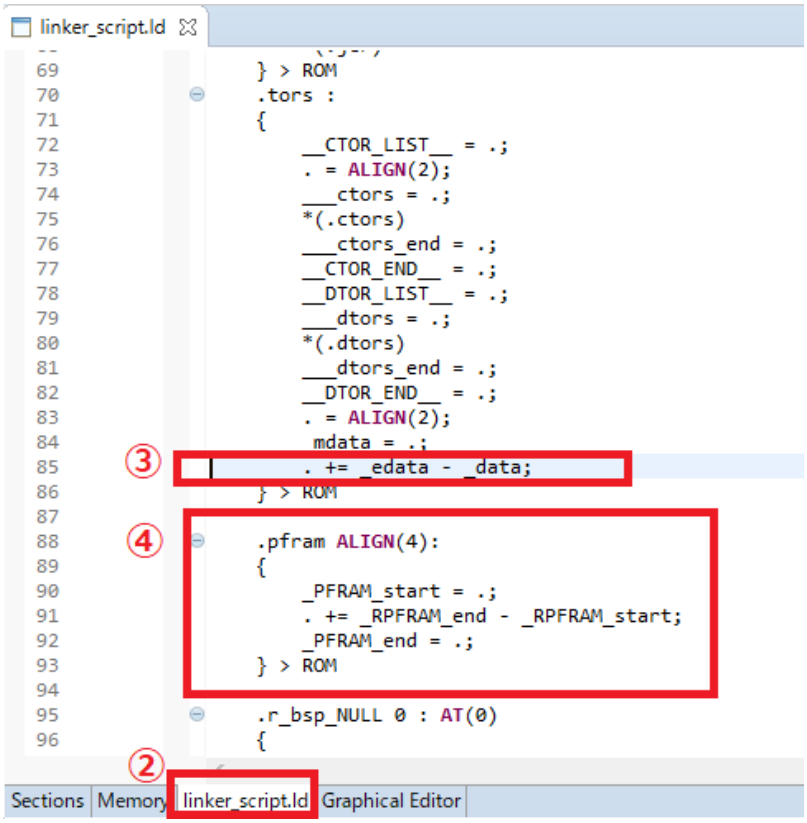
①プロジェクトエクスプローラーから「linker_script.ld」を開く



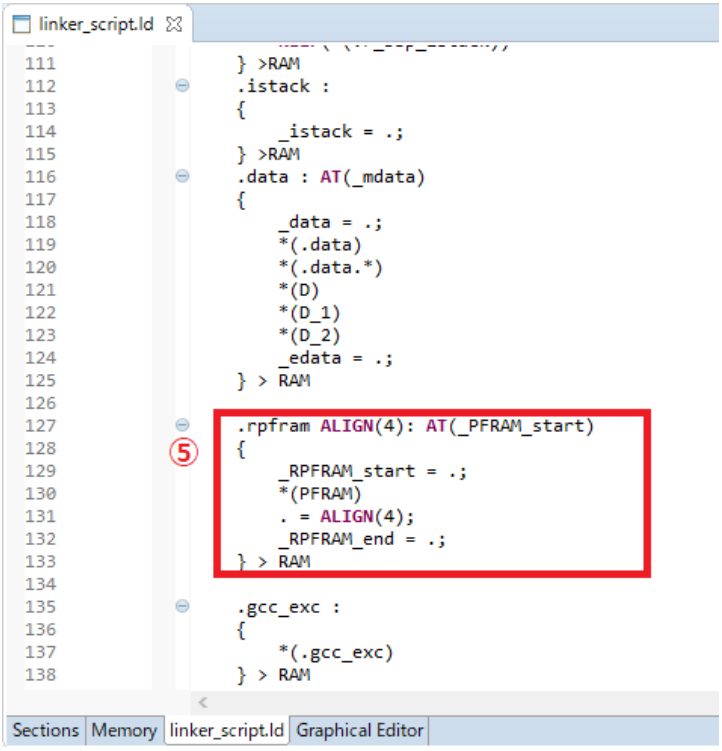
②「linker_script.ld」をクリック

③ (a)のコードを「_mata = . ;」の下に入力

④ (b)のコードを「.tors セクション」の下に入力



⑤ (c)のコードを「.data セクション」の下に入力



2.11.3　IAR C/C++ Compiler for Renesas RX

icf ファイルを編集して、セクション設定を追加してください。

編集する icf ファイルはプロジェクトのターゲットデバイスにより異なりますので、

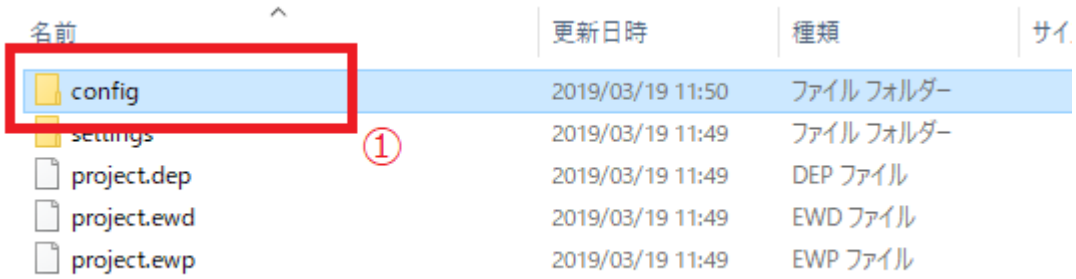
使用するデバイスの型名上 8 桁を確認して編集してください。

例として RX113(R5F51138ADFP)では「lnkr5f51138.icf」を編集します。

以下に RX113(R5F51138ADFP)での編集例を説明します。

セクション設定の追加

①プロジェクトフォルダに「config」フォルダを作成



② IAR C/C++ Compiler for Renesas RX(以下「EWRX」と記載)をインストールしたフォルダの「¥rx¥config」から「lnkr5f51138.icf」をプロジェクトフォルダの「config」フォルダにコピー

インストール時のデフォルトは「C:¥Program Files (x86)¥IAR Systems¥Embedded Workbench 8.1」です

③コピーした「lnkr5f51138.icf」ファイルを開き「initialize manually」に「section PFRAM」を追加

④「define block ISTACK…」の下に下記のコードを追加

```
define block PFRAM with alignment = 4 {section PFRAM};
define block PFRAM_init with alignment = 4 {section PFRAM_init};
```

⑤「place in ROM_region32」に「block PFRAM_init」を追加

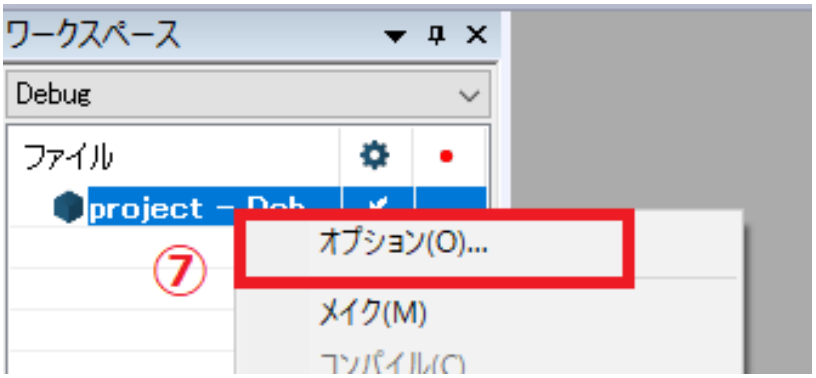
⑥「place in RAM_region32」の「ro section D_2」の下に「block PFRAM」を追加

lnkr5f51138.icf - メモ帳

ファイル(F)　編集(E)　書式(O)　表示(V)　ヘルプ(H)

```
define region DATA_FLASH    = mem:[from 0x00100000 to 0x00101FFF];

③ initialize manually { rw section .textrw,section PFRAM};
initialize by copy { rw, ro section D, ro section D_1, ro section D_2 };
initialize by copy with packing = none { section __DLIB_PERTHREAD };
do not initialize  { section .*.noinit };

define block HEAP     with alignment = 4, size = _HEAP_SIZE { };
define block USTACK   with alignment = 4, size = _USTACK_SIZE { };
define block ISTACK   with alignment = 4, size = _ISTACK_SIZE { };

④ define block PFRAM with alignment = 4 {section PFRAM};
define block PFRAM_init with alignment = 4 {section PFRAM_init};

define block STACKS with fixed order { block USTACK,
                                       block ISTACK };

place at address mem:0xFFFFFFFC      { ro section .resetvect };
place at address mem:0xFFFFFF80      { ro section .exceptvect };

"ROM16":place in ROM_region16        { ro section .code16*,
                                       ro section .data16*};
"RAM16":place in RAM_region16        { rw section .data16*,
                                       rw section __DLIB_PERTHREAD};
"ROM24":place in ROM_region24        { ro section .code24*,
                                       ro section .data24* };
"RAM24":place in RAM_region24        { rw section .data24* };
"ROM32":place in ROM_region32        { ro,
⑤                                      block PFRAM_init};
"RAM32":place in RAM_region32        { rw,
                                       ro section D,
                                       ro section D_1,
                                       ro section D_2,
                                  ⑥    block PFRAM,
                                       block HEAP};
"STACKS":place at end of RAM_region32 { block STACKS };
```

⑦EWRX から使用するプロジェクトを開き、ワークスペースのプロジェクトを右クリックし
オプションを開く

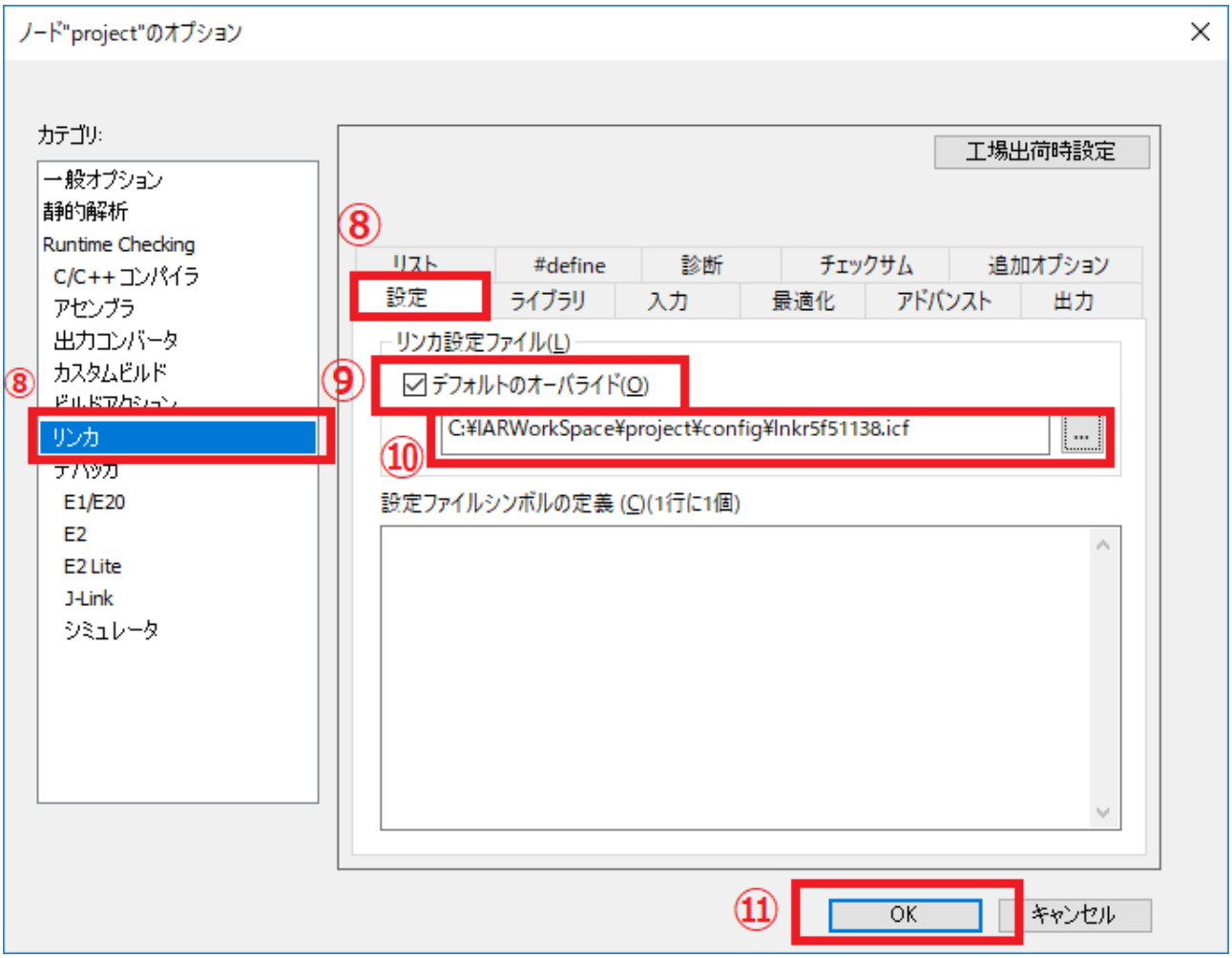


⑧「カテゴリ：リンカ」を選択し、設定をクリック

⑨「デフォルトのオーバライド」にチェックを入れる

⑩③で編集したファイルを参照先に設定する

⑪ 「OK」をクリック



2.12 API 関数使用時の注意事項

ユニーク ID の読み出しは ROM P/E モードに移行して行います。ROM P/E モード中は ROM の値を読み出せません。ベクタテーブルはデフォルトで ROM に配置されているため、ユニーク ID の読み出し(ROM P/E モード)中に割り込み要求が発生すると、割り込み処理を実行できずに暴走します。この問題を回避するには、R_UID_Read()関数実行前に割り込み要求を禁止に設定するか、ベクタテーブルと割り込み処理のプログラムを RAM に配置し、合わせて割り込みテーブルレジスタ(INTB)の値を変更しておく必要があります。また、ノンマスカブル割り込み要求を発生させないでください。

2.13 オンチップデバッグ時の注意事項

R_UID_Read 関数を実行中は ROM の値を読み出せません。

3. API 関数

3.1 R_UID_Open ()

この関数は、ROM から RAM にユニーク ID を読み出すためのプログラムを転送し、本モジュールが使用できる状態にします。この関数は他の API 関数を使用する前に実行される必要があります。

Format

uid_err_t　R_UID_Open(void)

Parameters

なし

Return Values

UID_SUCCESS　　*/* R_UID_Open 関数が正常に終了した場合 */*

UID_ERR_LOCK_FUNC　　*/* R_UID_Open 関数または R_UID_Read 関数が実行中に、
R_UID_Open 関数または R_UID_Read 関数が実行された場合 */*

UID_ERR_FAILURE　　*/* R_UID_Open 関数を 2 回以上実行した場合 */*

Properties

r_uid_rx_if.h にプロトタイプ宣言されています。

Description

ユニーク ID を読み出すための準備を行います。ROM から RAM へユニーク ID を読み出すためのプログラムを転送します。

Example

```
uid_err_t   err;

/* Initialization of the API. */
err = R_UID_Open();

/* Check error */
if ( UID_SUCCESS  !=  err)
{
   . . .
}
```

Special Notes:

なし

3.2　R_UID_Read ()

エクストラ領域からユニーク ID を読み出す関数です。

Format

uid_err_t　R_UID_Read(uint8_t *pdest_addr)

Parameters

uint8_t*　pdest_addr

ユニーク ID を格納するポインタを設定してください。

ポインタで指定した領域に 32 バイトのユニーク ID のデータが格納されます。

ポインタで指定した領域のサイズは 32 バイト以上を確保してください。

Return Values

UID_SUCCESS	*/* R_UID_Read 関数の実行でユニーク ID を正常に読み出せた場合 */*
UID_ERR_UNINITIALIZED	*/* R_UID_Open 関数を実行する前に R_UID_Read 関数を実行した場合 */*
UID_ERR_LOCK_FUNC	*/* R_UID_Open 関数または R_UID_Read 関数が実行中に、 R_UID_Open 関数または R_UID_Read 関数が実行された場合 */*
UID_ERR_FAILURE	*/* R_UID_Read 関数の実行でユニーク ID の読み出しに失敗した場合 */*

Properties

r_uid_rx_if.h にプロトタイプ宣言されています。

Description

エクストラ領域からユニーク ID を読み出し、指定した領域に格納します。

Example

```
uid_err_t   err;
uint8_t   unique_id[32];       /* Variable that stores the unique ID read */

/* Unique ID read */
err = R_UID_Read(unique_id);

/* Check error */
if ( UID_SUCCESS  !=  err)
{
   . . .
}
```

Special Notes:

本関数を実行前に、割り込み要求を禁止に設定するか、ベクタテーブルと割り込み処理のプログラムを RAM に配置し、合わせて割り込みテーブルレジスタ(INTB)の値を変更してください。また、ノンマスカブル割り込み要求を発生させないでください。

3.3　R_UID_GetVersion ()

この関数は、API のバージョンを返す関数です。

Format

uint32_t　　R_UID_GetVersion(void)

Parameters

なし

Return Values

バージョン番号

Properties

r_uid_rx_if.h にプロトタイプ宣言されています。

Description

本 API のバージョン番号を返します。

Example

```
uint32_t   version;

version = R_UID_GetVersion();
```

Special Notes:

なし

4. 付録

4.1 動作確認環境

本 FIT モジュールの動作確認環境を以下に示します。

表 4.1 動作確認環境 (Rev. 1.13)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e² studio V7.4.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V3.01.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.13
使用ボード	RX13T CPU カード（型名：RTK0EMXA10C00000BJ）

表 4.2 動作確認環境 (Rev. 1.11)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e² studio V7.3.0 IAR Embedded Workbench for Renesas RX 4.10.1
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V3.01.00 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
	GCC for Renesas RX 4.8.4.201801 コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -std=gnu99
	IAR C/C++ Compiler for Renesas RX version 4.10.1 コンパイルオプション：統合開発環境のデフォルト設定
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.11
使用ボード	Renesas Starter Kit for RX113（型名：R0K505113S900BE）

表 4.3 動作確認環境 (Rev.1.10)

項目	内容
統合開発環境	ルネサスエレクトロニクス製 e² studio V6.1.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler for RX Family V2.07.00
	コンパイルオプション：統合開発環境のデフォルト設定に以下のオプションを追加 -lang = c99
エンディアン	ビッグエンディアン/リトルエンディアン
モジュールのリビジョン	Rev1.10
使用ボード	Renesas Starter Kit for RX130（型名：RTK5005130S00000BE）

4.2　トラブルシューティング

(1) Q：本 FIT モジュールをプロジェクトに追加しましたが、ビルド実行すると「Could not open source file "platform.h"」エラーが発生します。

A：FIT モジュールがプロジェクトに正しく追加されていない可能性があります。プロジェクトへの追加方法をご確認ください。

- CS+を使用している場合
  アプリケーションノート RX ファミリ CS+に組み込む方法 Firmware Integration Technology (R01AN1826)」
- e^2 studio を使用している場合
  アプリケーションノート RX ファミリ e^2 studio に組み込む方法 Firmware Integration Technology (R01AN1723)」

また、本 FIT モジュールを使用する場合、ボードサポートパッケージ FIT モジュール(BSP モジュール)もプロジェクトに追加する必要があります。BSP モジュールの追加方法は、アプリケーションノート「ボードサポートパッケージモジュール(R01AN1685)」を参照してください。

(2) Q：本 FIT モジュールをプロジェクトに追加しましたが、ビルド実行すると「This MCU is not supported by the current r_uid_rx module.」エラーが発生します。

A：追加した FIT モジュールがユーザプロジェクトのターゲットデバイスに対応していない可能性があります。追加した FIT モジュールの対象デバイスを確認してください。

5. 参考ドキュメント

ユーザーズマニュアル：ハードウェア
（最新版をルネサス エレクトロニクスホームページから入手してください。）

テクニカルアップデート／テクニカルニュース
（最新の情報をルネサス エレクトロニクスホームページから入手してください。）

ユーザーズマニュアル：開発環境
RX ファミリ CC-RX コンパイラ ユーザーズマニュアル（R20UT3248）
（最新版をルネサス エレクトロニクスホームページから入手してください。）

テクニカルアップデートの対応について

本モジュールは以下のテクニカルアップデートの内容を反映しています。

対応しているテクニカルアップデートはありません。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Dec.01.14	—	初版発行
1.10	Dec.06.17	1	対象デバイス：　RX130 を追加
		6	2.2 ソフトウェアの要求：　RX130 サポートの r_bsp の v3.60 を更新
		6	2.3 サポートされているツールチェイン：　RX130 サポートのツールチェインの v.2.07.00 を追記
1.11	May.20.19	—	以下のコンパイラに対応 ・GCC for Renesas RX ・IAR C/C++ Compiler for Renesas RX
		1	「対象コンパイラ」の章を追加
		3	1.2「ユニーク ID FIT モジュールの概要」章を更新
		6	1.5「制限事項」章を追加
		8	2.7「コードサイズ」章を追加
		11	2.10「for 文、while 文、do while 文について」章を追加
		12-16	2.11「プロジェクトの設定」章に「GCC for Renesas RX」「IAR C/C++ Compiler for Renesas RX」を追加
		21	「R_UID_GetVersion()」の章を更新
		22	4.1 「動作確認環境」章 に 「表 4.1 動作確認環境(Rev.1.11)」を追加
		プログラム	R_UID_GetVersion 関数のインライン展開を削除
1.12	Jul.05.19	14-15	2.11.2「GCC for Renesas RX」章の内容を変更
1.13	Jul.31.19	1	対象デバイス：　RX13T を追加
		8	2.7「コードサイズ」章の内容を変更
		22	4.1 「動作確認環境」章 に 「表 4.1 動作確認環境(Rev.1.13)」を追加
1.14	Jun.10.20	—	API 関数のコメントを Doxygen スタイルに変更
		1	関連ドキュメントの、R01AN1833 を削除
		10	2.9「FIT モジュールの追加方法」を更新
		20-23	API 説明ページの、「Reentrant」項目を削除

製品ご使用上の注意事項

ここでは、マイコン製品全体に適用する「使用上の注意事項」について説明します。個別の使用上の注意事項については、本ドキュメントおよびテクニカルアップデートを参照してください。

1. 静電気対策
   CMOS 製品の取り扱いの際は静電気防止を心がけてください。CMOS 製品は強い静電気によってゲート絶縁破壊を生じることがあります。運搬や保存の際には、当社が出荷梱包に使用している導電性のトレーやマガジンケース、導電性の緩衝材、金属ケースなどを利用し、組み立て工程にはアースを施してください。プラスチック板上に放置したり、端子を触ったりしないでください。また、CMOS 製品を実装したボードについても同様の扱いをしてください。
2. 電源投入時の処置
   電源投入時は、製品の状態は不定です。電源投入時には、LSI の内部回路の状態は不確定であり、レジスタの設定や各端子の状態は不定です。外部リセット端子でリセットする製品の場合、電源投入からリセットが有効になるまでの期間、端子の状態は保証できません。同様に、内蔵パワーオンリセット機能を使用してリセットする製品の場合、電源投入からリセットのかかる一定電圧に達するまでの期間、端子の状態は保証できません。
3. 電源オフ時における入力信号
   当該製品の電源がオフ状態のときに、入力信号や入出力プルアップ電源を入れないでください。入力信号や入出力プルアップ電源からの電流注入により、誤動作を引き起こしたり、異常電流が流れ内部素子を劣化させたりする場合があります。資料中に「電源オフ時における入力信号」についての記載のある製品は、その内容を守ってください。
4. 未使用端子の処理
   未使用端子は、「未使用端子の処理」に従って処理してください。CMOS 製品の入力端子のインピーダンスは、一般に、ハイインピーダンスとなっています。未使用端子を開放状態で動作させると、誘導現象により、LSI 周辺のノイズが印加され、LSI 内部で貫通電流が流れたり、入力信号と認識されて誤動作を起こす恐れがあります。
5. クロックについて
   リセット時は、クロックが安定した後、リセットを解除してください。プログラム実行中のクロック切り替え時は、切り替え先クロックが安定した後に切り替えてください。リセット時、外部発振子（または外部発振回路）を用いたクロックで動作を開始するシステムでは、クロックが十分安定した後、リセットを解除してください。また、プログラムの途中で外部発振子（または外部発振回路）を用いたクロックに切り替える場合は、切り替え先のクロックが十分安定してから切り替えてください。
6. 入力端子の印加波形
   入力ノイズや反射波による波形歪みは誤動作の原因になりますので注意してください。CMOS 製品の入力がノイズなどに起因して、V_{IL}（Max.）から V_{IH}（Min.）までの領域にとどまるような場合は、誤動作を引き起こす恐れがあります。入力レベルが固定の場合はもちろん、V_{IL}（Max.）から V_{IH}（Min.）までの領域を通過する遷移期間中にチャタリングノイズなどが入らないように使用してください。
7. リザーブアドレス（予約領域）のアクセス禁止
   リザーブアドレス（予約領域）のアクセスを禁止します。アドレス領域には、将来の拡張機能用に割り付けられている リザーブアドレス（予約領域）があります。これらのアドレスをアクセスしたときの動作については、保証できませんので、アクセスしないようにしてください。
8. 製品間の相違について
   型名の異なる製品に変更する場合は、製品型名ごとにシステム評価試験を実施してください。同じグループのマイコンでも型名が違うと、フラッシュメモリ、レイアウトパターンの相違などにより、電気的特性の範囲で、特性値、動作マージン、ノイズ耐量、ノイズ幅射量などが異なる場合があります。型名が違う製品に変更する場合は、個々の製品ごとにシステム評価試験を実施してください。

ご注意書き

1. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器・システムの設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因して生じた損害（お客様または第三者いずれに生じた損害も含みます。以下同じです。）に関し、当社は、一切その責任を負いません。
2. 当社製品、本資料に記載された製品データ、図、表、プログラム、アルゴリズム、応用回路例等の情報の使用に起因して発生した第三者の特許権、著作権その他の知的財産権に対する侵害またはこれらに関する紛争について、当社は、何らの保証を行うものではなく、また責任を負うものではありません。
3. 当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
4. 当社製品を、全部または一部を問わず、改造、改変、複製、リバースエンジニアリング、その他、不適切に使用しないでください。かかる改造、改変、複製、リバースエンジニアリング等により生じた損害に関し、当社は、一切その責任を負いません。
5. 当社は、当社製品の品質水準を「標準水準」および「高品質水準」に分類しており、各品質水準は、以下に示す用途に製品が使用されることを意図しております。
   標準水準：　コンピュータ、OA 機器、通信機器、計測機器、AV 機器、家電、工作機械、パーソナル機器、産業用ロボット等
   高品質水準：輸送機器（自動車、電車、船舶等）、交通制御（信号）、大規模通信機器、金融端末基幹システム、各種安全制御装置等
   当社製品は、データシート等により高信頼性、Harsh environment 向け製品と定義しているものを除き、直接生命・身体に危害を及ぼす可能性のある機器・システム（生命維持装置、人体に埋め込み使用するもの等）、もしくは多大な物的損害を発生させるおそれのある機器・システム（宇宙機器と、海底中継器、原子力制御システム、航空機制御システム、プラント基幹システム、軍事機器等）に使用されることを意図しておらず、これらの用途に使用することは想定していません。たとえ、当社が想定していない用途に当社製品を使用したことにより損害が生じても、当社は一切その責任を負いません。
6. 当社製品をご使用の際は、最新の製品情報（データシート、ユーザーズマニュアル、アプリケーションノート、信頼性ハンドブックに記載の「半導体デバイスの使用上の一般的な注意事項」等）をご確認の上、当社が指定する最大定格、動作電源電圧範囲、放熱特性、実装条件その他指定条件の範囲内でご使用ください。指定条件の範囲を超えて当社製品をご使用された場合の故障、誤動作の不具合および事故につきましては、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質および信頼性の向上に努めていますが、半導体製品はある確率で故障が発生したり、使用条件によっては誤動作したりする場合があります。また、当社製品は、データシート等において高信頼性、Harsh environment 向け製品と定義しているものを除き、耐放射線設計を行っておりません。仮に当社製品の故障または誤動作が生じた場合であっても、人身事故、火災事故その他社会的損害等を生じさせないよう、お客様の責任において、冗長設計、延焼対策設計、誤動作防止設計等の安全設計およびエージング処理等、お客様の機器・システムとしての出荷保証を行ってください。特に、マイコンソフトウェアは、単独での検証は困難なため、お客様の機器・システムとしての安全検証をお客様の責任で行ってください。
8. 当社製品の環境適合性等の詳細につきましては、製品個別に必ず当社営業窓口までお問合せください。ご使用に際しては、特定の物質の含有・使用を規制する RoHS 指令等、適用される環境関連法令を十分調査のうえ、かかる法令に適合するようご使用ください。かかる法令を遵守しないことにより生じた損害に関して、当社は、一切その責任を負いません。
9. 当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器・システムに使用することはできません。当社製品および技術を輸出、販売または移転等する場合は、「外国為替及び外国貿易法」その他日本国および適用される外国の輸出管理関連法規を遵守し、それらの定めるところに従い必要な手続きを行ってください。
10. お客様が当社製品を第三者に転売等される場合には、事前に当該第三者に対して、本ご注意書き記載の諸条件を通知する責任を負うものといたします。
11. 本資料の全部または一部を当社の文書による事前の承諾を得ることなく転載または複製することを禁じます。
12. 本資料に記載されている内容または当社製品についてご不明な点がございましたら、当社の営業担当者までお問合せください。

注 1. 本資料において使用されている「当社」とは、ルネサス エレクトロニクス株式会社およびルネサス エレクトロニクス株式会社が直接的、間接的に支配する会社をいいます。

注 2. 本資料において使用されている「当社製品」とは、注 1 において定義された当社の開発、製造製品をいいます。

(Rev.4.0-1　2017.11)

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。