RX ファミリ

例外ベクタテーブルと選択型割り込みの使い方

要旨

本アプリケーションノートでは、RX ファミリ MCU の例外ベクタテーブルと選択割り込みの使い方について説明します。

例外ベクタテーブルでは、例外ベクタテーブルの配置アドレスを変更する方法を説明します。選択型割り込みでは、割り込みベクタ番号に割り込み要因を割り当てて選択型割り込みを発生させる方法を説明します。

また、特に記載のない箇所は RX65N グループについて説明します。他のマイコンを使用する場合は、「6 他 RX ファミリにサンプルコードをポーティングする方法」および各マイコンのユーザーズマニュアルハードウェア編を確認してください。

対象デバイス

- 例外ベクタテーブルと選択型割り込みを搭載する RX ファミリ MCU

目次

1.　例外ベクタテーブルと選択型割り込み

1.1　例外ベクタテーブル

例外ベクタテーブルは例外テーブルレジスタ(EXTB)の値を先頭アドレスとする 124 バイトの領域に、各例外事象のベクタを配置します。EXTB に任意のアドレスを設定することで、例外ベクタテーブルの配置アドレスを変えることができます。

本アプリケーションノートでは、EXTB の値を 0xFFFFFF80 から 0x0000FF80 に変更しています。図 1.1 に RX65N グループ、RX651 グループの例外ベクタテーブルを示します。

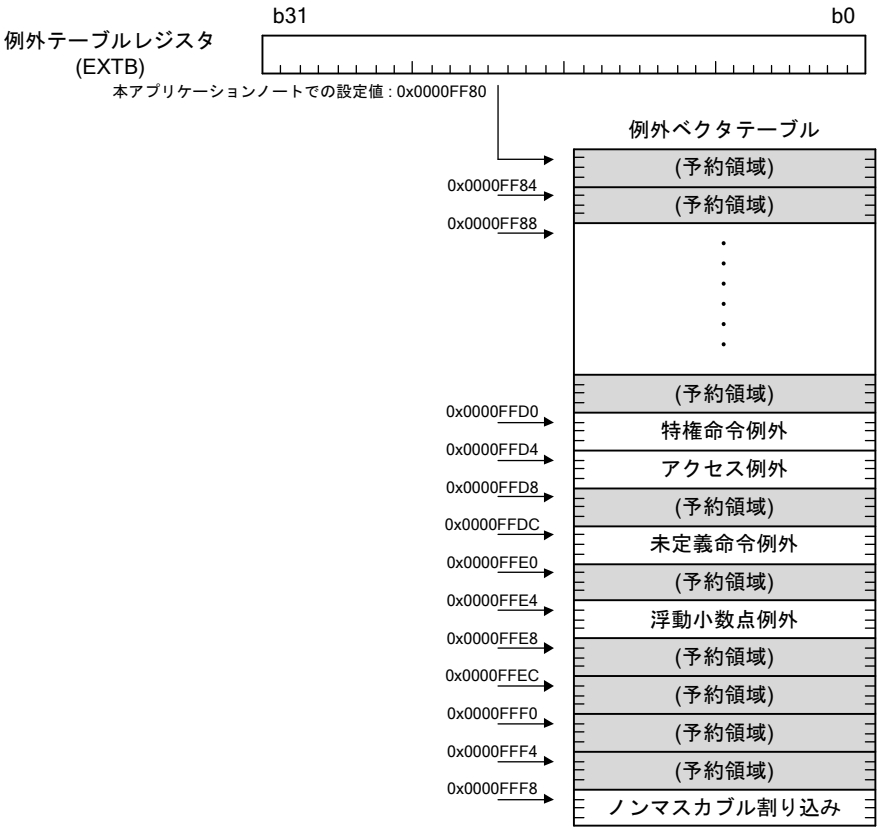


図 1.1　RX65N グループ、RX651 グループの例外ベクタテーブル

1.2　選択型割り込み

選択型割り込みは、割り込みベクタ番号 128～255 に複数の周辺モジュールの割り込み要因から任意の 1 つを選択して割り当てることができます。またそれらは周辺モジュールの動作クロックにより、選択型割り込み B と選択型割り込み A に分類されます。

選択型割り込みでは、SLIBXRn レジスタ、SLIBRn レジスタ、SLIARn レジスタに割り込み要因番号を設定することで選択型割り込み要因の割り当てを行います。

SLIBXRn レジスタは 128 番から 143 番、SLIBRn レジスタは 144 番から 207 番までの割り込みベクタ番号に、選択型割り込み B に分類された割り込み要因を割り当てるためのレジスタです。

SLIARn レジスタは 208 番から 255 番までの割り込みベクタ番号に、選択型割り込み A に分類された割り込み要因を割り当てるためのレジスタです。

本アプリケーションノートでは、SLIAR208 レジスタ(割り込みベクタ番号 208 を選択)に TGIA0(割り込み要因番号 1)を割り当てています。

図 1.2 に RX65N グループ、RX651 グループの選択型割り込み要因の割り当てを示します。

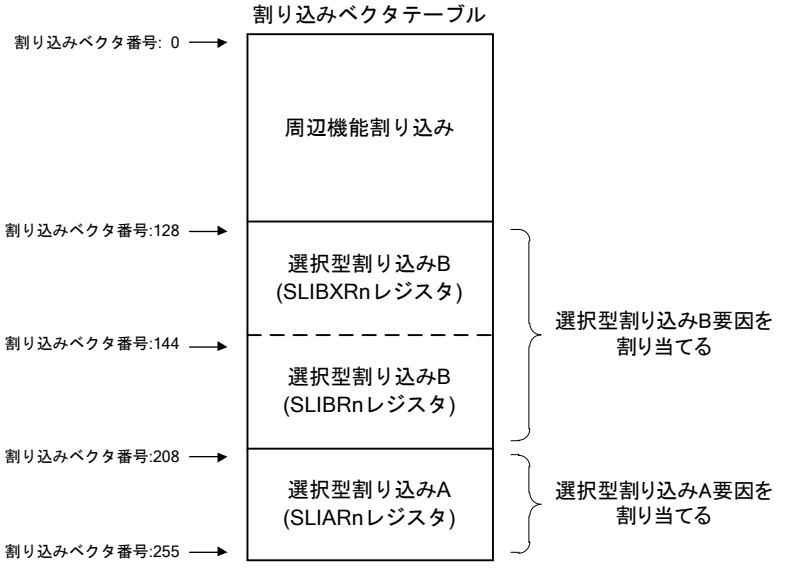


図 1.2　RX65N グループ、RX651 グループの選択型割り込み要因の割り当て

2. 動作確認条件

本アプリケーションノートのサンプルコードは、下記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	R5F565NEDDFC (RX65N グループ)
動作周波数	メインクロック:24MHz PLL:240MHz(メインクロック 1 分周 10 逓倍) システムクロック(ICLK):120MHz(PLL2 分周) 周辺モジュールクロック A(PCLKA):120MHz(PLL2 分周)
動作電圧	3.3V
統合開発環境	ルネサスエレクトロニクス製 e² studio Version:7.3.0
C コンパイラ	ルネサスエレクトロニクス製 C/C++ Compiler Package for RX Family V3.01.00
	コンパイルオプション rom オプションを使用しています -rom=RAM_EXFUNC=RAM_EXFUNC_COPY - rom=RAM_EXVECT=RAM_EXVECT_COPY
iodefine.h のバージョン	2.0a
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	例外ベクタテーブルの移動 ユーザモード 選択型割り込み スーパバイザモード
サンプルコードのバージョン	Version1.10
使用ボード	Renesas Starter Kit+ for RX65N-2MB(製品型名: RTK50565N2SxxxxxBE)

3. 関連アプリケーションノート

本アプリケーションノートに関連するアプリケーションノートを以下に示します。併せて参照してください。

- RX65N、RX651 グループ　初期設定例　(R01AN3034)

4.　ハードウェア説明

4.1　使用端子一覧

表 4.1 に使用端子と機能を、図 4.1 に接続図を示します。

表 4.1　使用端子と機能

端子名	入出力	内容
P73	出力	LED0 出力(例外処理:特権命令例外)
PG5	出力	LED3 出力(選択型割り込み:インプットキャプチャ/コンペアマッチ割り込み)

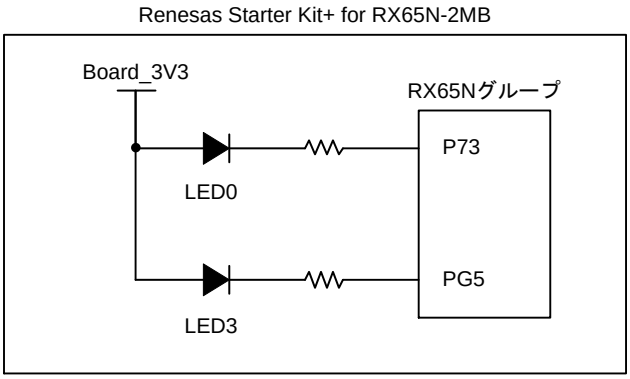


図 4.1　接続図

5.　ソフトウェア説明

5.1　例外ベクタテーブルの動作概要

本アプリケーションノートのサンプルでは、一つのプログラムで例外ベクタテーブルの移動処理と選択型割り込み処理を紹介しています。上記はファイル”r_int_config.h”内で定義されている定数 SEL_INT で切り替えることができます。

例外ベクタテーブルの移動処理を動作させる場合、定数 SEL_INT に EXCEP_HANDL を設定してください。なお、サンプルコードのデフォルト設定は EXCEP_HANDL です。

5.1.1　例外ベクタテーブルの移動処理

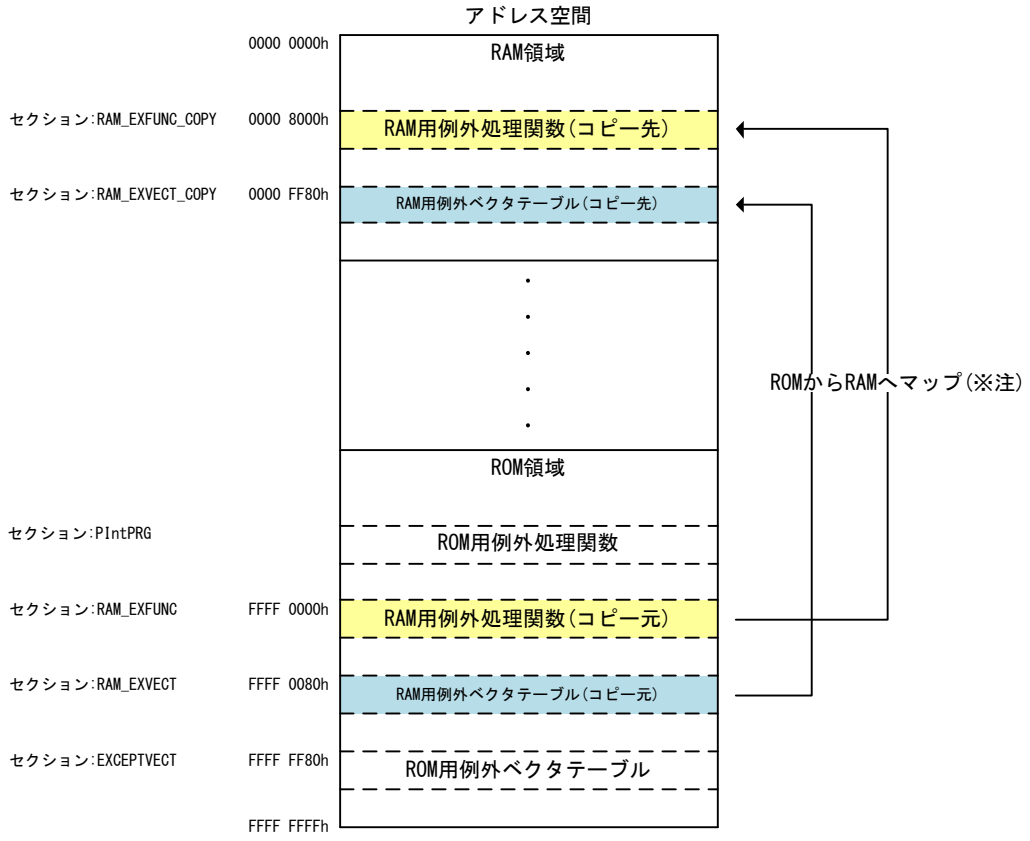
例外ベクタテーブルの移動処理では、RAM 上で例外処理を実行するために以下の処理をしています。

(1)　例外処理関数を RAM 領域(0x00008000)に移動します。
(2)　例外ベクタテーブルの内容を RAM 領域(0x0000FF80)に移動します。
(3)　EXTB の値を ROM 領域(デフォルト値:0xFFFFFF80)から RAM 領域(0x0000FF80)に変更します。
(4)　プロセッサモードをスーパバイザモードからユーザモードに切り換えて、特権命令を実行し割り込み(特権命令例外)を発生させます(注)。
(5)　割り込み処理で LED0 を点灯します。

注. プロセッサ割り込み優先レベルを 2 に設定しますが、特権命令はスーパバイザモードでのみ実行可能なため、値は変わりません。

本機能を使うことでフラッシュ書き換え中などの ROM にアクセスできない時でも RAM 上で例外処理を実行することができます。

図 5.1 に本アプリケーションノートにおけるアドレス空間を示します。



※注 RAM用例外ベクタテーブルとRAM用例外処理関数のROMからRAMへの移動方法については「5.1.2 例外ベクタテーブルをRAM領域に移動する方法」を参照してください。

図 5.1　アドレス空間

5.1.2　例外ベクタテーブルを RAM 領域に移動する方法

RAM 用の例外ベクタテーブルと例外処理関数を用意し、例外ベクタテーブルの配置アドレスを ROM 領域(デフォルト値:0xFFFFFF80)から RAM 領域(0x0000FF80)に変更する方法を説明します。

1)　セクションの設定

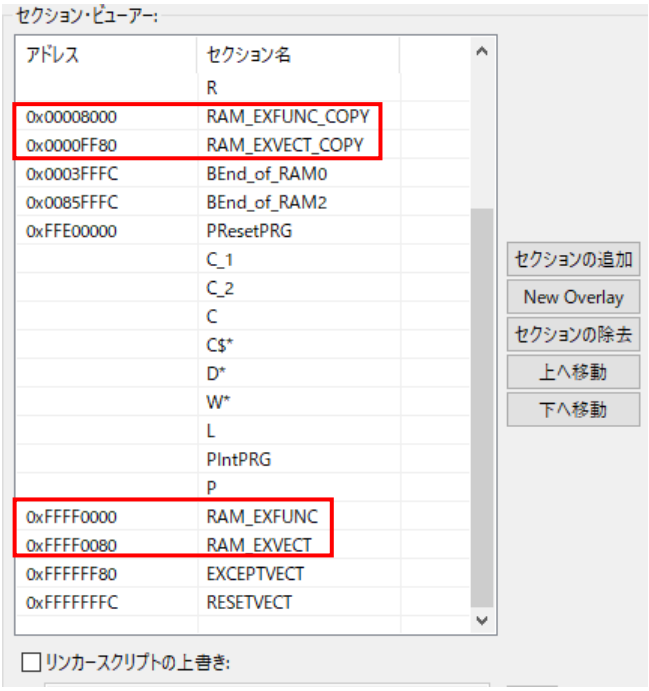
1-1) プロジェクトを右クリックし「プロパティ」を選択

1-2)「プロパティ」から「C/C++ビルド ＞ 設定 ＞Linker＞ セクション」を選択

1-3) 右上に表示される「…」をクリック

1-4) RAM 領域と ROM 領域に例外ベクタテーブルと例外処理関数のセクションを設定

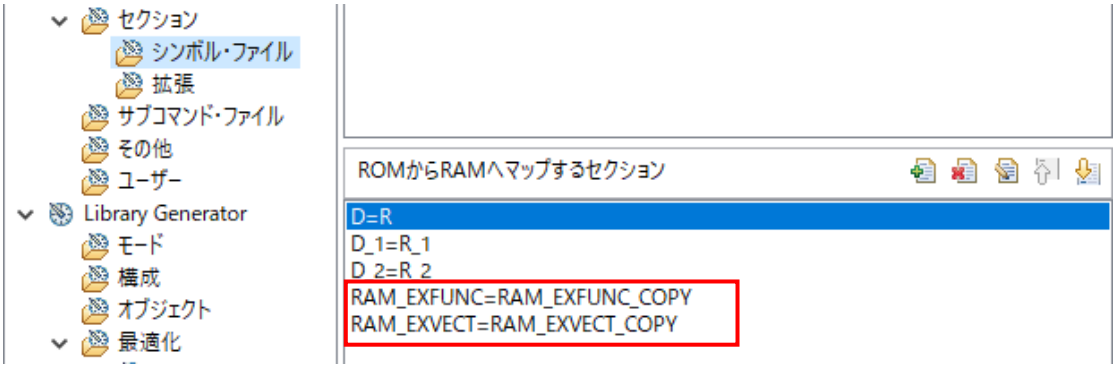
1-5)「OK」をクリック



1-6)「プロパティ」から「C/C++ビルド ＞ 設定 ＞Linker＞ シンボル・ファイル」を選択

1-7) rom オプションを使用し、ROM セクション内の定義シンボルを RAM セクション内のアドレスにリロケーション

1-8)「Apply and Close」をクリック



2)　ROM 領域に RAM 用例外ベクタテーブルと RAM 用例外処理関数を配置

2-1) ram_except_vector_table.c で#pragma section C RAM_EXVECT を宣言

RAM_EXVECT セクションに RAM 用例外ベクタテーブルを配置します。

2-2) ram_except_handlers.c で#pragma section P RAM_EXFUNC を宣言

RAM_EXFUNC セクションに RAM 用例外処理関数を配置します。

3)　RAM 領域セクションの初期化

3-1) 初期化対象のセクションを dbsct.c のセクション初期化用テーブル(DTBL)へ追加

初期化用テーブル(DTBL)に記述されているセクションは、初期化用ルーチン(_INITSCT)を呼び出すことで RAM 領域セクションが初期化されます。本アプリケーションノートでは、この処理を利用し、2)で宣言したセクションを初期化します。この初期化では、RAM_EXFUNC を RAM_EXFUNC_COPY にコピーし、RAM_EXVECT を RAM_EXVECT_COPY にコピーします。下記に RX65N グループ、RX651 グループの初期化用テーブル(DTBL)を示します。

```
#pragma section C C$DSEC
extern const struct {
    _UBYTE *rom_s;        /* Start address of the initialized data section in ROM */
    _UBYTE *rom_e;        /* End address of the initialized data section in ROM   */
    _UBYTE *ram_s;        /* Start address of the initialized data section in RAM */
}   _DTBL[] = {
    { __sectop("D"), __secend("D"), __sectop("R") },
    { __sectop("D_2"), __secend("D_2"), __sectop("R_2") },
    { __sectop("D_1"), __secend("D_1"), __sectop("R_1") },
    { __sectop("RAM_EXFUNC"), __secend("RAM_EXFUNC"), __sectop("RAM_EXFUNC_COPY") },
    { __sectop("RAM_EXVECT"), __secend("RAM_EXVECT"), __sectop("RAM_EXVECT_COPY") }
};
```

4)　例外テーブルレジスタ(EXTB)に先頭アドレス(0x0000FF80)を設定

5.2　選択型割り込みの動作概要

本アプリケーションノートのサンプルでは、一つのプログラムで例外ベクタテーブルの移動処理と選択型割り込み処理を紹介しています。上記はファイル”r_int_config.h”内で定義されている定数 SEL_INT で切り替えることができます。

選択型割り込み処理を動作させる場合、定数 SEL_INT に SOFTWARE_CONFIG_INT を設定してください。なお、サンプルコードのデフォルト設定は EXCEP_HANDL です。

5.2.1　選択型割り込み処理

選択型割り込み処理では、選択型割り込みを発生させるために以下の処理をしています。

(1)　初期設定
ポート、クロック、周辺機能の初期設定を行います。

(2)　周辺機能(選択型割り込み)の初期設定
IER1A.IEN0 ビットを“0”(割り込み要求を禁止)、SLIAR208 レジスタ(割り込みベクタ番号 208)を“01h”(TGIA0(TGRA のインプットキャプチャ/コンペアマッチ)を割り当て)にします。その後、SLIPRCR.WPCR ビットを“1”(SLIARn レジスタへの書き込み禁止)、 IR208.IR フラグを“0”にします。

(3)　MTU0.TCNT カウンタのカウント動作開始
IER1A.IEN0 ビットを“1”(割り込み要求を許可)、TSTRA.CST0 ビットを“1”(MTU0.TCNT カウンタのカウント動作を開始)にします。

(4)　コンペアマッチ割り込み
MTU0.TCNT カウンタと MTU0.TGRA レジスタの値が一致したときに、TGIA0 割り込みの IR208.IR フラグが“1”になり、コンペアマッチ割り込み要求が発生します。MTU0.TCNT カウンタと MTU0.TGRA レジスタの値が一致すると MTU0.TCNT カウンタは“0000h”になり、再びカウントアップを開始します。割り込みが受け付けられると IR208.IR フラグは“0”になります。250ms の間隔でコンペアマッチ割り込みが発生するごとに LED3 の点灯/消灯を切り換えます。

(5)　MTU0.TCNT カウンタのカウント動作停止
20 回目の割り込みが発生したとき、IER1A.IEN0 ビットを“0”(割り込み要求を禁止)、TSTRA.CST0 ビットを“0”(MTU0.TCNT カウンタのカウント動作を停止)にします。

表 5.1 に使用する周辺機能と用途を、図 5.2 に選択型割り込みのタイミング図を示します。

表 5.1　使用する周辺機能と用途

周辺機能	用途
MTU	コンペアマッチ
I/O ポート	LED 点灯

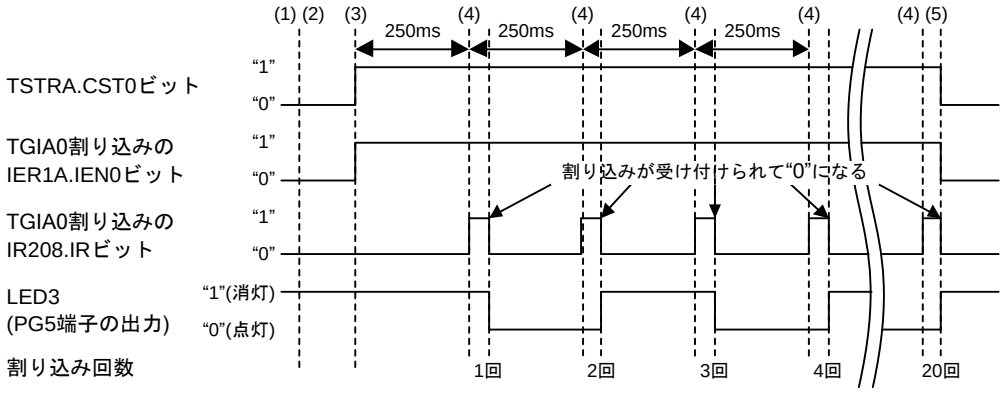


図 5.2　選択型割り込みのタイミング図

5.3　ファイル構成

表 5.2 にサンプルコードで使用するファイルを示します。なお、統合開発環境で自動生成されるファイルは除きます。

表 5.2　サンプルコードで使用するファイル

ファイル名	概要	備考
main.c	メイン処理	
r_init_stop_module.c	リセット後に動作している周辺機能の停止	
r_init_stop_module.h	r_init_stop_module.c のヘッダファイル	
r_init_port_initialize.c	存在しないポートの初期設定	
r_init_port_initialize.h	r_init_port_initialize.c のヘッダファイル	
r_init_clock.c	クロック初期設定	
r_init_clock.h	r_init_clock.c のヘッダファイル	
r_int_config.c	例外ベクタテーブルの移動と選択型割り込みの初期設定	
r_int_config.h	r_int_config.c のヘッダファイル	
r_ram_except_handlers.c	RAM 用例外処理関数	
r_ram_except_handlers.h	RAM 用例外処理関数のヘッダファイル	
r_ram_except_vector_table.c	RAM 用例外ベクタテーブル	

5.4　オプション設定メモリ

表 5.3 に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。必要に応じて、お客様のシステムに最適な値を設定してください。

表 5.3　サンプルコードで使用するオプション設定メモリ

シンボル	アドレス	設定値	内容
OFS0	FE7F 5D04h～FE7F 5D07h	FFFF FFFFh	リセット後、IWDT は停止 リセット後、WDT は停止
OFS1	FE7F 5D08h～FE7F 5D0Bh	FFFF FFFFh	リセット後、電圧監視 0 リセット無効 リセット後、HOCO 発振が無効
MDE	FE7F 5D00h～FE7F 5D03h	FFFF FFFFh	リトルエンディアン

5.5　定数一覧

表 5.4 にサンプルコードで使用する定数を示します。

表 5.4　サンプルコードで使用する定数

定数名	設定値	内容
EXCEP_HANDL	0	例外処理
SOFTWARE_CONFIG_INT	1	選択型割り込み
SEL_INT	EXCEP_HANDL	割り込みの選択 EXCEP_HANDL　：例外処理 SOFTWARE_CONFIG_INT　：選択型割り込み
OUTPUT_HIGH	1	High 出力
OUTPUT_LOW	0	Low 出力
LED_ON	OUTPUT_LOW	LED 点灯時の出力データ OUTPUT_HIGH　: High 出力 OUTPUT_LOW　: Low 出力
LED_OFF	OUTPUT_HIGH	LED 消灯時の出力データ OUTPUT_HIGH　: High 出力 OUTPUT_LOW　: Low 出力
LED0_PODR	PORT7.PODR.BIT.B3	LED0 と接続されているポートのポート出力データレジスタ(PODR)
LED0_PDR	PORT7.PDR.BIT.B3	LED0 と接続されているポートのポート方向レジスタ(PDR)
LED3_PODR	PORTG.PODR.BIT.B5	LED3 と接続されているポートのポート出力データレジスタ(PODR)
LED3_PDR	PORTG.PDR.BIT.B5	LED3 と接続されているポートのポート方向レジスタ(PDR)
MTU_PCLK_HZ	120000000	マルチファンクションタイマパルスユニット(MTU)の動作周波数

5.6　変数一覧

表 5.5 に static 型変数を示します。

表 5.5　static 型変数

型	変数名	内容	使用関数
static uint8_t	gs_int_cnt	割り込みカウンタ	mtu_init peria_inta208

5.7　関数一覧

表 5.6 に関数を示します。

表 5.6　関数

関数名	概要
main	メイン処理
R_INIT_StopModule	リセット後に動作している周辺機能の停止
R_INIT_Port_Initialize	存在しないポートの初期設定
R_INIT_Clock	クロック初期設定
R_INT_Config	例外ベクタテーブルの移動と選択型割り込みの設定
port_init	ポート初期設定
exception_handling_main	例外ベクタテーブルの移動
mtu_init	MTU 初期設定
software_config_int_main	選択型割り込みの設定
mtu_start	MTU カウントアップ開始
mtu_stop	MTU カウントアップ停止
peria_inta208	選択型割り込み処理(割り込みベクタ番号:208)
ram_excep_super_visor_inst	特権命令例外処理
ram_dummy	ダミー処理

5.8　関数仕様

サンプルコードの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	初期設定後、特権命令実行または MTU のカウントアップを開始します。
引　数	なし
リターン値	なし

R_INIT_StopModule	
概　要	リセット後に動作している周辺機能の停止
ヘッダ	r_init_stop_module.h
宣　言	void R_INIT_StopModule (void)
説　明	モジュールストップ状態へ遷移する設定を行います。
引　数	なし
リターン値	なし
備考	サンプルコードでは、モジュールストップ状態への遷移は行っていません。 本関数の詳細は、アプリケーションノート「RX65N グループ、RX651 グループ 初期設定例」を参照してください。

R_INIT_ Port_Initialize	
概　要	存在しないポートの初期設定
ヘッダ	r_init_port_initialize.h
宣　言	void R_INIT_Port_Initialize (void)
説　明	存在しないポートの端子に対応するポート方向レジスタの初期設定を行います。
引　数	なし
リターン値	なし
備考	サンプルコードでは、176 ピン版(PIN_SIZE=176)に設定しています。 本関数をコールした後に、存在しないポートを含む PDR、PODR レジスタへバイト単位で書き込む場合、存在しないポートの方向制御ビットには“1”、ポート出力データ格納ビットには“0”を設定してください。 本関数の詳細は、アプリケーションノート「RX65N グループ、RX651 グループ 初期設定例」を参照してください。

R_INIT_Clock	
概　要	クロック初期設定
ヘッダ	r_init_clock.h
宣　言	void R_INIT_Clock (void)
説　明	クロックの初期設定、ROM ウェイトサイクルの設定を行います。
引　数	なし
リターン値	なし
備考	サンプルコードでは、システムクロックと PLL、ROM ウェイトサイクルを 2 ウェイトとし、サブクロックを使用しない処理を選択しています。 本関数で呼んでいる set_ad_conversion_time 関数は、PSW.I ビットが“0”のとき、かつ ADCSR.ADST ビットが“0”のときに呼び出される必要があります。そのため、本関数を呼び出す前に、PSW.I ビットを“0”(割り込み禁止)、ADCSR.ADST ビットを“0”にしてください。 本関数の詳細は、アプリケーションノート「RX65N グループ、RX651 グループ 初期設定例」を参照してください。

R_INT_Config	
概　要	例外ベクタテーブルの移動と選択型割り込みの設定
ヘッダ	r_int_config.h
宣　言	void R_INT_Config (void)
説　明	例外ベクタテーブルの移動または選択型割り込みの設定を行います。
引　数	なし
リターン値	なし

port_init	
概　要	ポート初期設定
ヘッダ	r_int_config.h
宣　言	static void port_init (void)
説　明	LED0、LED3 点滅用のポートの初期設定を行います。
引　数	なし
リターン値	なし

exception_handling_main	
概　要	例外ベクタテーブルの移動
ヘッダ	r_int_config.h
宣　言	static void exception_handling_main (void)
説　明	例外ベクタテーブルの移動を行います。
引　数	なし
リターン値	なし

mtu_init	
概　要	MTU 初期設定
ヘッダ	r_int_config.h
宣　言	static void mtu_init (void)
説　明	MTU の初期設定を行います。
引　数	なし
リターン値	なし

software_config_int_main	
概　要	選択型割り込みの設定
ヘッダ	r_int_config.h
宣　言	static void software_config_int_main (void)
説　明	選択型割り込みの設定を行います。
引　数	なし
リターン値	なし

mtu_start	
概　要	MTU カウントアップ開始
ヘッダ	r_int_config.h
宣　言	static void mtu_start (void)
説　明	MTU0.TCNT カウンタをクリア、TGIA0 割り込み要求を許可し、MTU0.TCNT カウンタのカウントアップを開始します。
引　数	なし
リターン値	なし

mtu_stop	
概　要	MTU カウントアップ停止
ヘッダ	r_int_config.h
宣　言	static void mtu_stop (void)
説　明	MTU0.TCNT カウンタのカウントアップを停止し、TGIA0 割り込み要求を禁止します。
引　数	なし
リターン値	なし

peria_inta208	
概　要	選択型割り込み処理(割り込みベクタ番号:208)
ヘッダ	r_int_config.h
宣　言	void peria_inta208 (void)
説　明	LED3 の点灯/消灯を切り換えます。20 回目の割り込みが発生すると MTU カウントアップ停止関数を呼び出します。
引　数	なし
リターン値	なし

ram_excep_super_visor_inst	
概　要	特権命令例外処理
ヘッダ	r_ram_except_handlers.h
宣　言	ram_excep_super_visor_inst (void)
説　明	特権命令例外処理を行います。LED0 を点灯後、無限ループします。
引　数	なし
リターン値	なし

ram_dummy	
概　要	ダミー処理
ヘッダ	r_ram_except_handlers.h
宣　言	ram_dummy (void)
説　明	この関数では何もしていません。
引　数	なし
リターン値	なし

5.9　フローチャート

5.9.1　メイン処理

図 5.3 にメイン処理のフローチャートを示します。

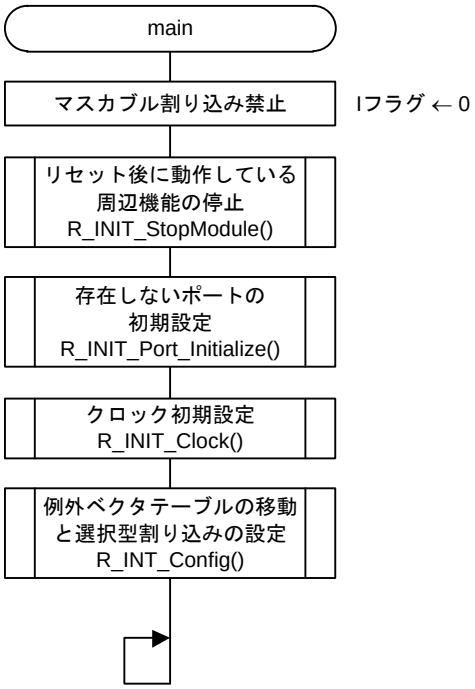


図 5.3　メイン処理

5.9.2　例外ベクタテーブルの移動と選択型割り込みの設定

図 5.4 に例外ベクタテーブルの移動と選択型割り込みの設定のフローチャートを示します。

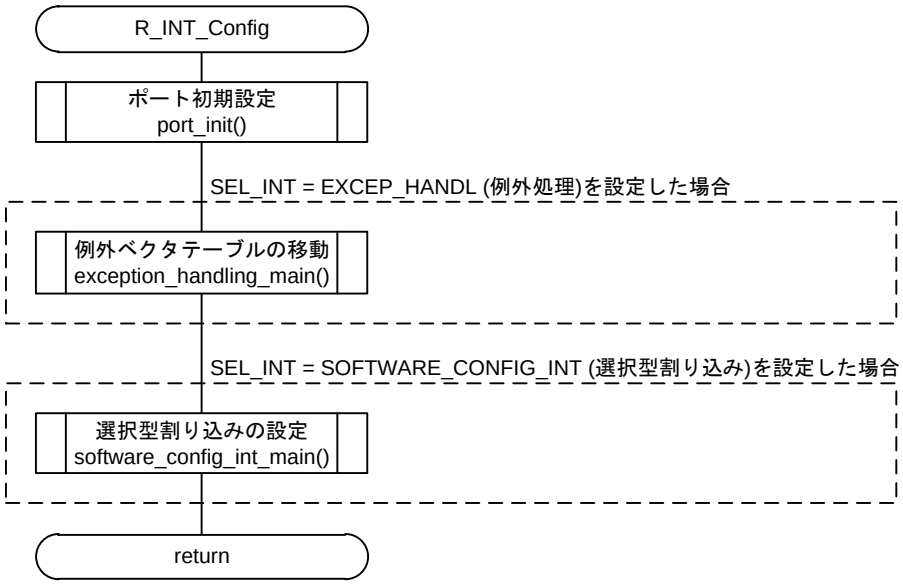


図 5.4　例外ベクタテーブルの移動と選択型割り込みの設定

5.9.3　ポート初期設定

図 5.5 にポート初期設定のフローチャートを示します。

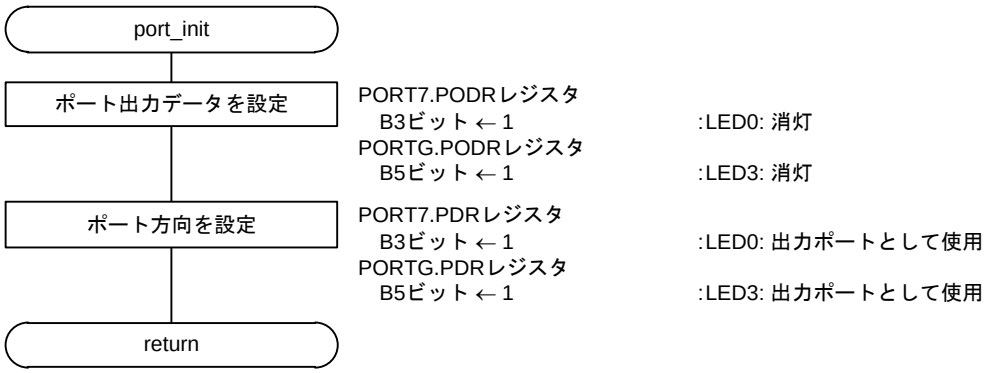
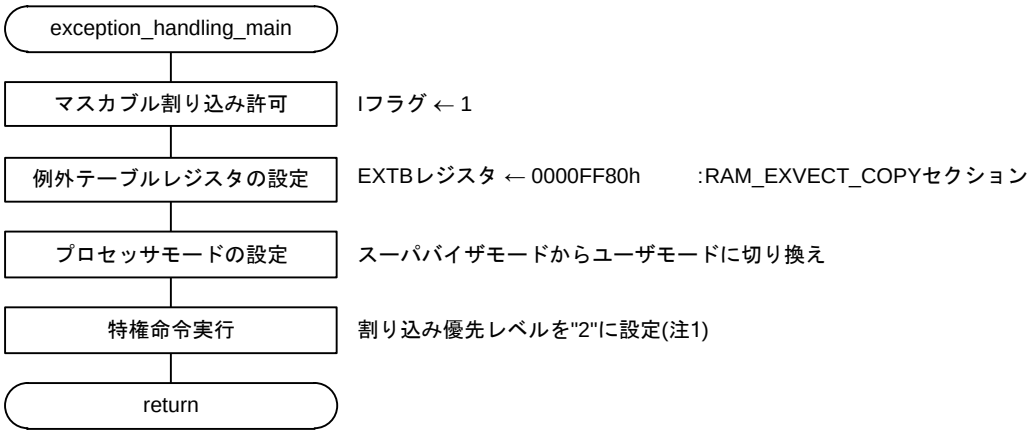


図 5.5　ポート初期設定

5.9.4　例外ベクタテーブルの移動

図 5.6 に例外ベクタテーブルの移動のフローチャートを示します。



注1.　ユーザモードなので特権命令例外が発生します。

図 5.6　例外ベクタテーブルの移動

5.9.5　MTU 初期設定

図 5.7 に MTU 初期のフローチャートを示します。

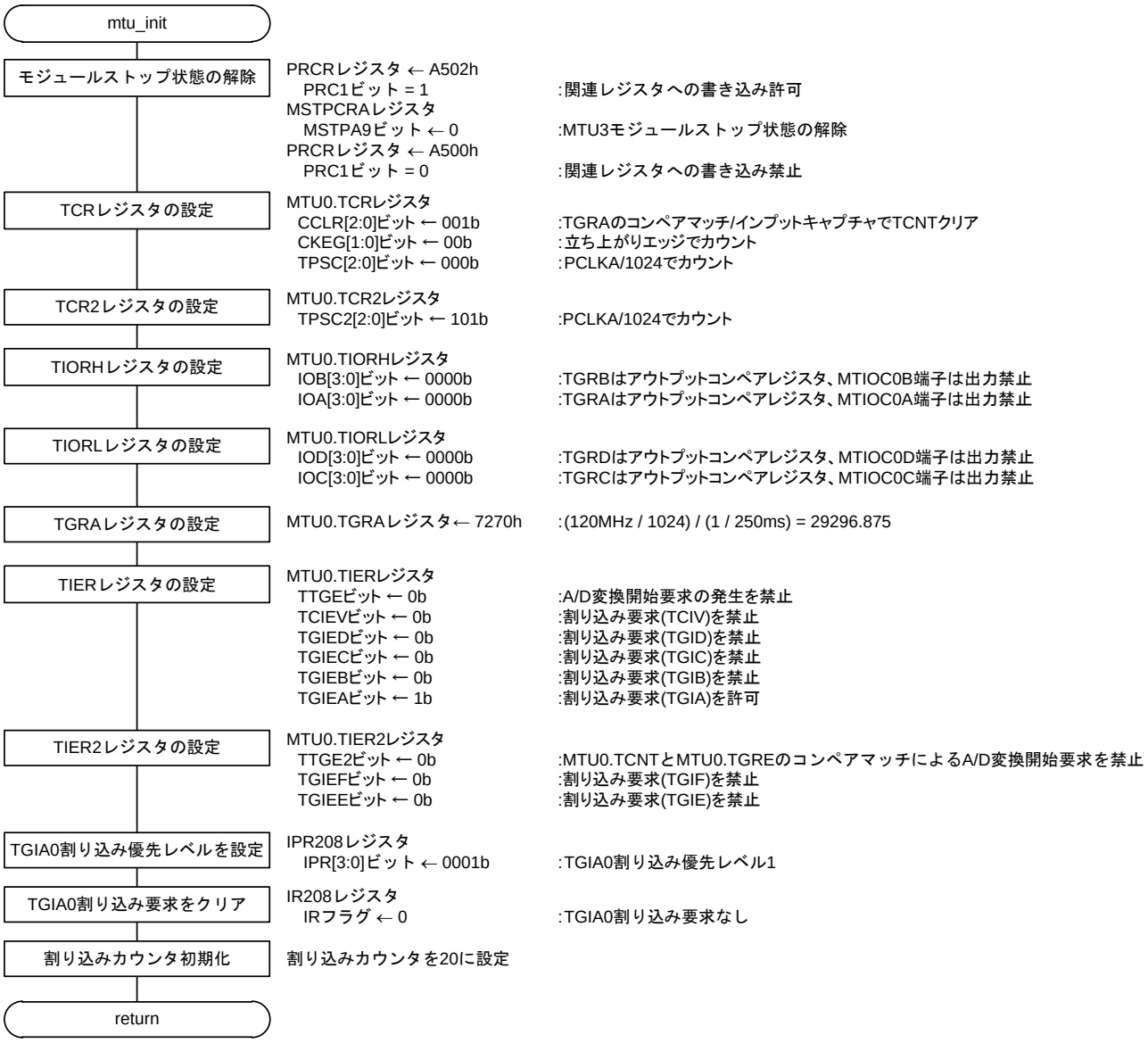
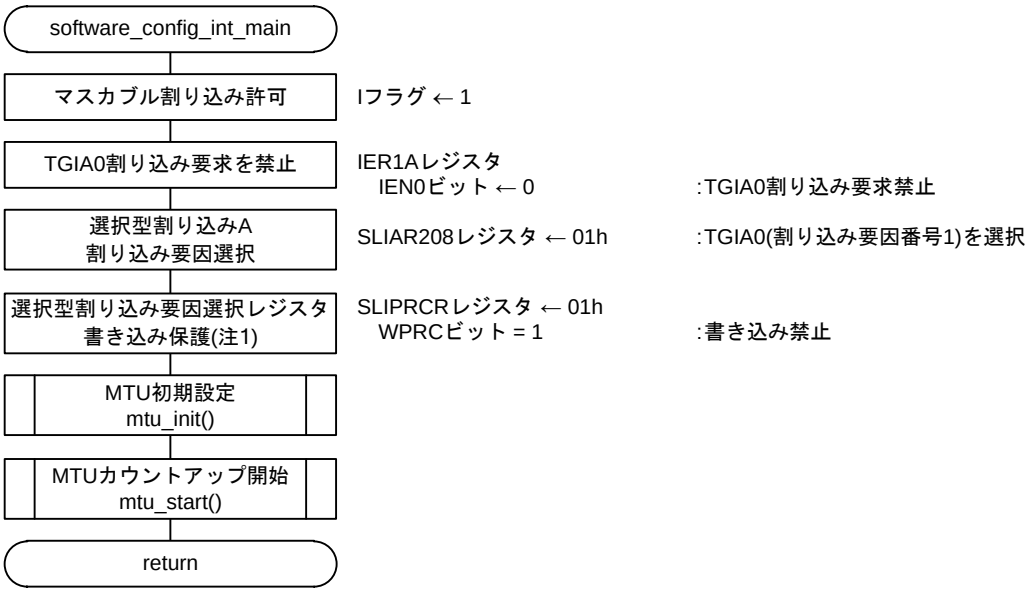


図 5.7　MTU 初期設定

5.9.6　選択型割り込みの設定

図 5.8 に選択型割り込みの設定のフローチャートを示します。



注1.　WPRCビットに“1”を書き込んだ後、WPRCビットが“1”になっていることを確認してください。
WPRCビットを一度“1”にするとソフトフェアでは“0”にできません。他の選択型割り込み要因選択レジスタの設定をする場合は、WPRCビットを“1”にする前に設定してください。

図 5.8　選択型割り込みの設定

5.9.7　MTU カウントアップ開始

図 5.9 に MTU カウントアップ開始のフローチャートを示します。

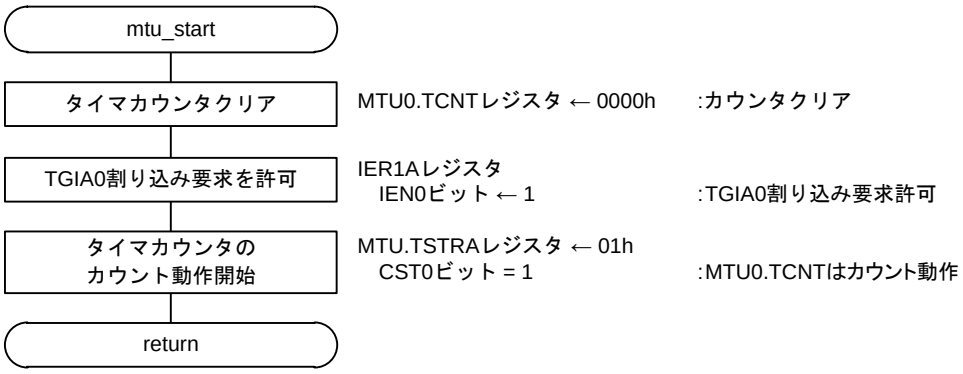


図 5.9　MTU カウントアップ開始

5.9.8　MTU カウントアップ停止

図 5.10 に MTU カウントアップ停止のフローチャートを示します。

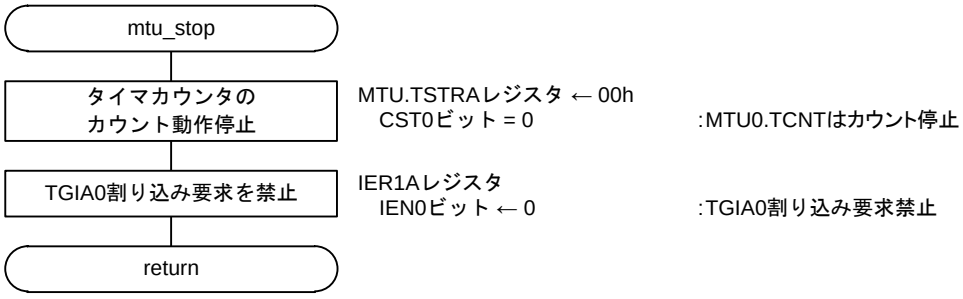


図 5.10　MTU カウントアップ停止

5.9.9　選択型割り込み処理(割り込みベクタ番号:208)

図 5.11 に選択型割り込み処理(割り込みベクタ番号:208)のフローチャートを示します。

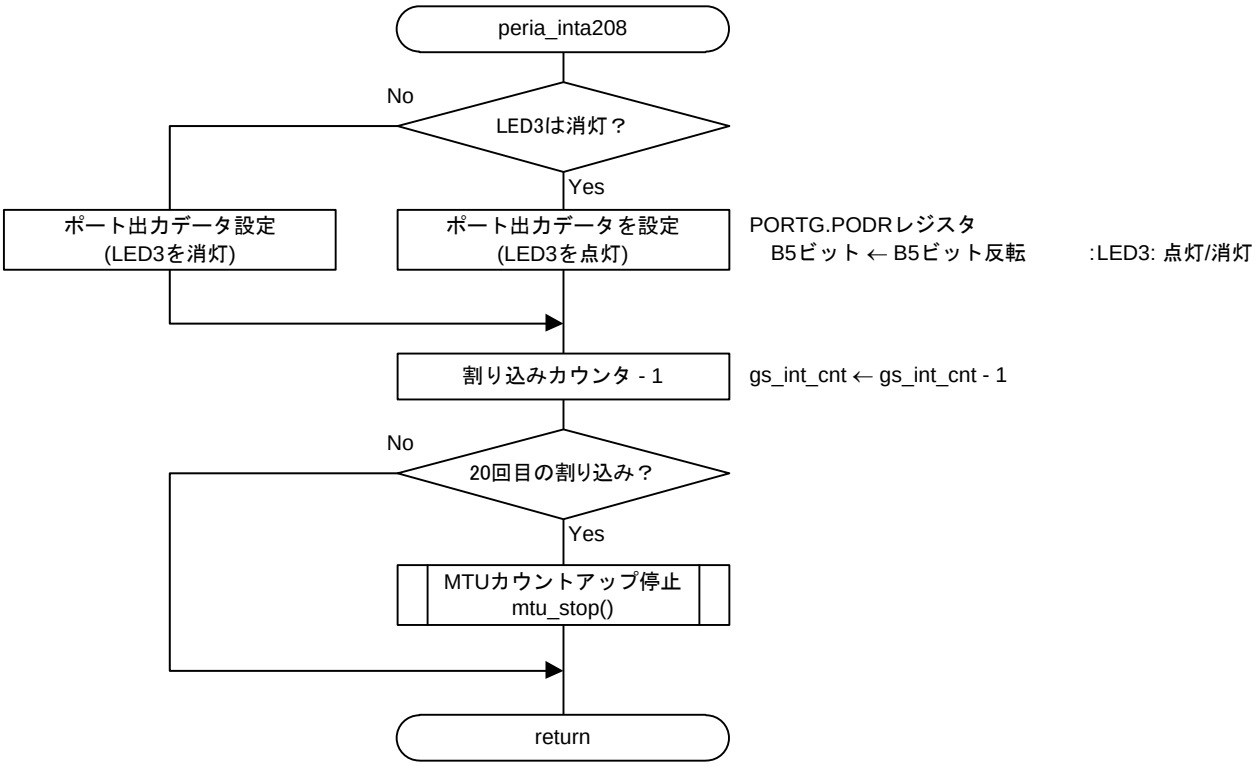


図 5.11　選択型割り込み処理(割り込みベクタ番号:208)

5.9.10 特権命令例外処理

図 5.12 に特権命令例外処理のフローチャートを示します。

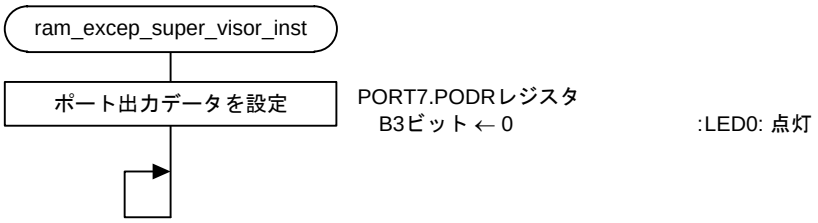


図 5.12　特権命令例外処理

5.9.11 ダミー処理

図 5.13 にダミー処理のフローチャートを示します。

図 5.13　ダミー処理

6.　他 RX ファミリにサンプルコードをポーティングする方法

本アプリケーションノートに同梱されているサンプルコードは例外ベクタテーブルと選択型割り込みを搭載する他 RX ファミリにポーティングできます。本章ではサンプルコードを RX66T(Renesas Starter Kit for RX66T)にポーティングする例を示します。

6.1　ポーティングをする前に

ポーティングする前に下記の仕様をあらかじめ確認してください。仕様に差異がある場合は、本章に記載する方法が適用できない場合があります。十分に確認をして、本アプリケーションを活用してください。

- ポーティング元とポーティング先の例外ベクタテーブルと選択型割り込みの仕様
- ポーティング元とポーティング先の MTU の仕様

6.2　ポーティング手順フロー

図 6.1 にポーティング手順フローを示します。

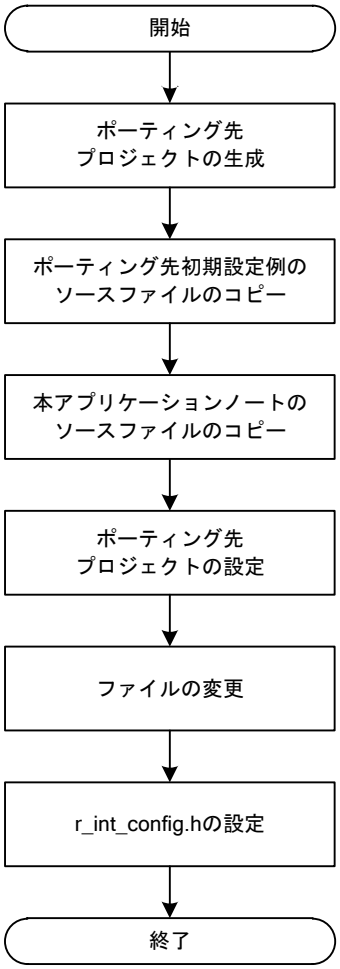


図 6.1　ポーティング手順フロー

6.3　ポーティング手順

6.3.1　ポーティング先プロジェクトの生成

e² studio を起動して、新規にプロジェクトを作成します。

1)　ポーティング先プロジェクトの生成

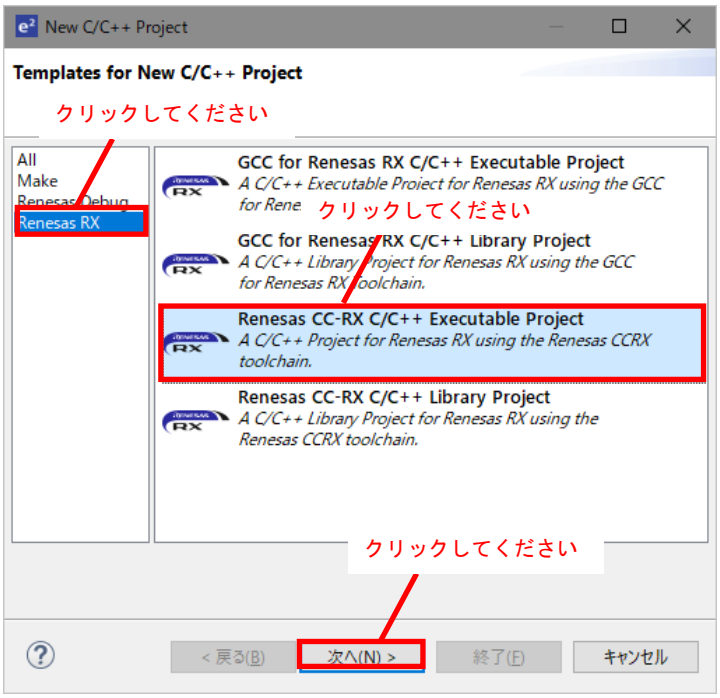
1-1) e² studio を起動して[ファイル(F)]をクリックしてください。

1-2) [新規(N)]の[C/C++ Project]をクリックして、New C/C++ Project ウィザードを起動します。

1-3) [Renesas RX]をクリックしてください。

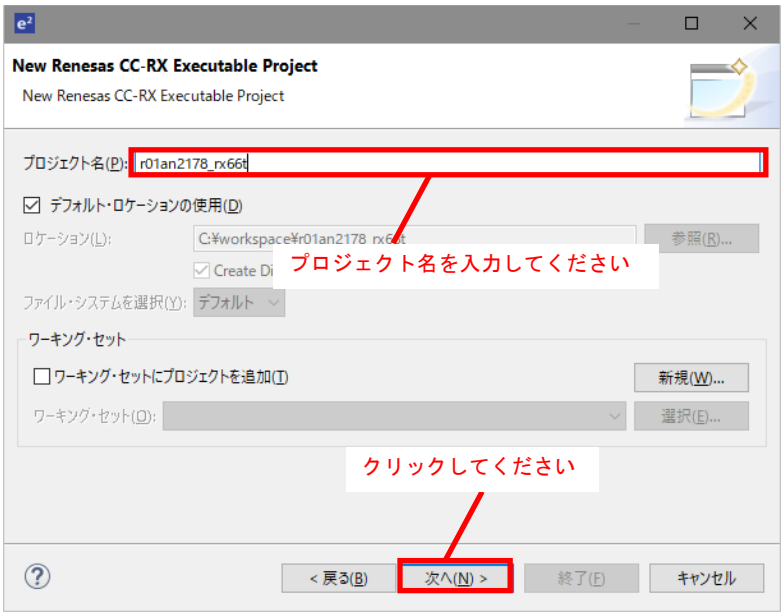
1-4) [Renesas CC-RX C/C++ Executable Project]をクリックしてください。

1-5) [次へ(N)>]をクリックしてください。



1-6) 任意のプロジェクト名を入力してください。

1-7) [次へ(N)>]をクリックしてください。

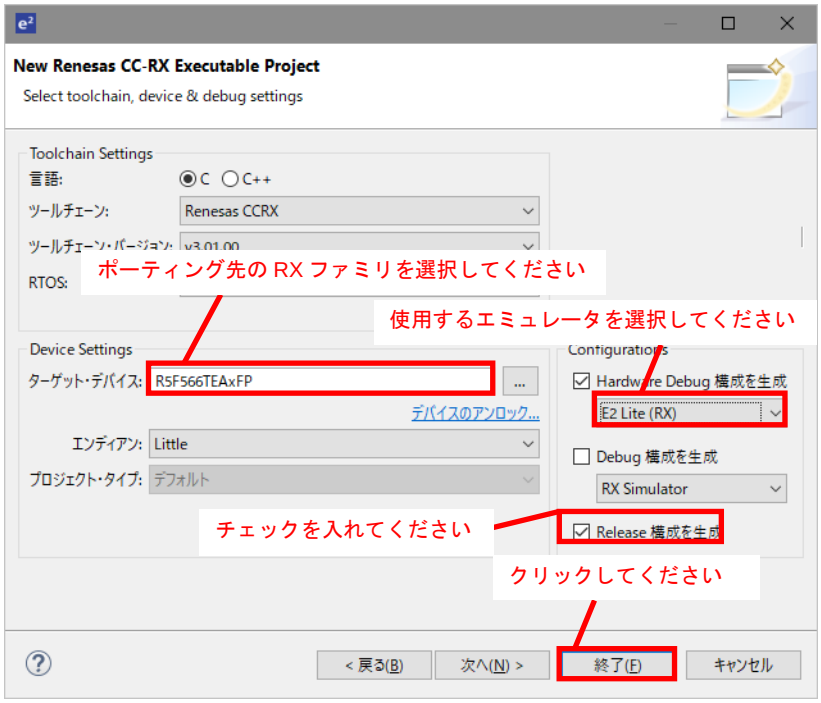


1-8) [ターゲットデバイス:]を[R5F566TEAxFP]に変更してください。
(他 RX ファミリにポーティングする場合は、ポーティング先の RX ファミリに変更してください)

1-9) 使用するエミュレータを選択してください。

1-10) [Release 構成を生成]にチェックを入れてください。

1-11) [終了(N)]をクリックしてください。



1-12) 生成したプロジェクトにある[<プロジェクト名>.c]を削除してください。

6.3.2　ポーティング先初期設定例のソースファイルのコピー

ポーティング先の RX ファミリの初期設定例アプリケーションノートのソースファイルを、新規生成したプロジェクトにコピーします。

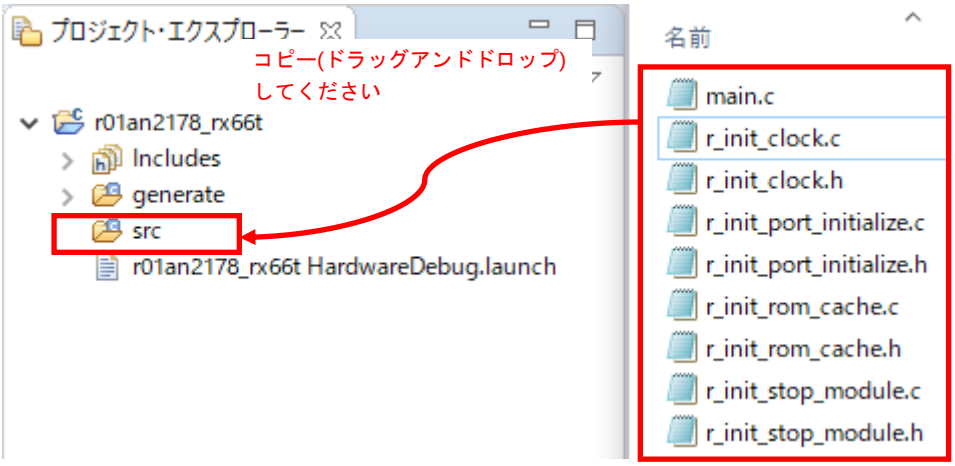
1)　初期設定例アプリケーションノートのダウンロード

1-1) [RX66T グループ 初期設定例(R01AN4057)]をルネサスエレクトロニクスホームページからダウンロードしてください。
(他 RX ファミリにポーティングする場合は、ポーティング先の RX ファミリに対応した初期設定例アプリケーションノートをダウンロードしてください)

1-2) ダウンロードした zip ファイルを任意の場所に解凍してください。

2)　初期設定例アプリケーションノートのソースファイルをプロジェクトにコピー

2-1) 解凍したフォルダをエクスプローラーで開き、[r01an4057_rx66t] -> [r01an4057_src]にあるすべてのファイルを生成したプロジェクトにコピーしてください。

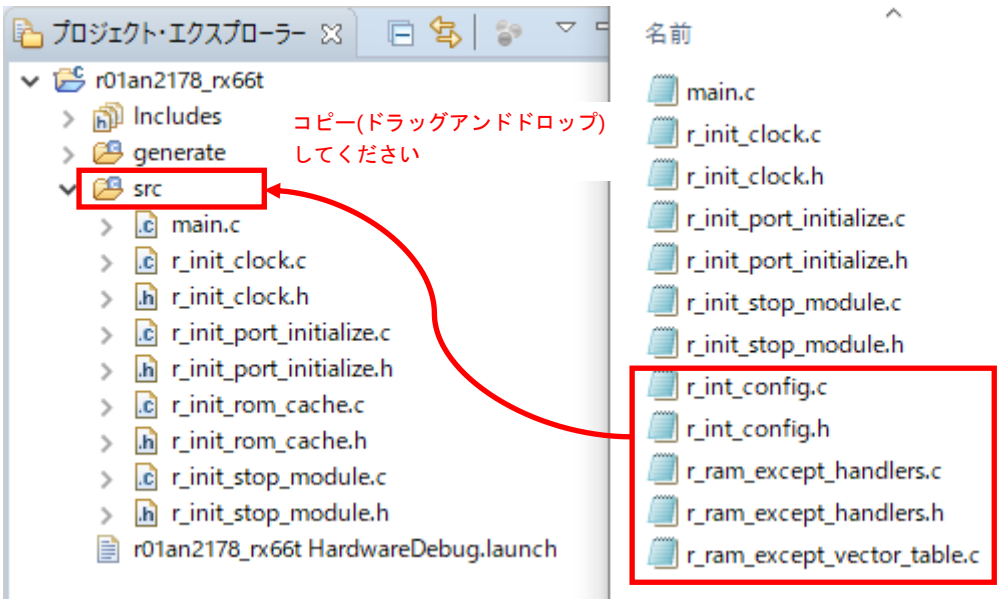


6.3.3　本アプリケーションノートのソースファイルのコピー

本アプリケーションのソースファイルを生成したプロジェクトにコピーします。

1)　本アプリケーションのソースファイルをプロジェクトにコピー

1-1) 本アプリケーションの[r01an2178_rx65n_2m] -> [r01an2178_src]にある[r_int_config.c]、[r_int_config.h]、[r_ram_except_handlers.c]、[r_ram_except_handlers.h]、[r_ram_except_vector_table.c]をプロジェクトにコピーします。

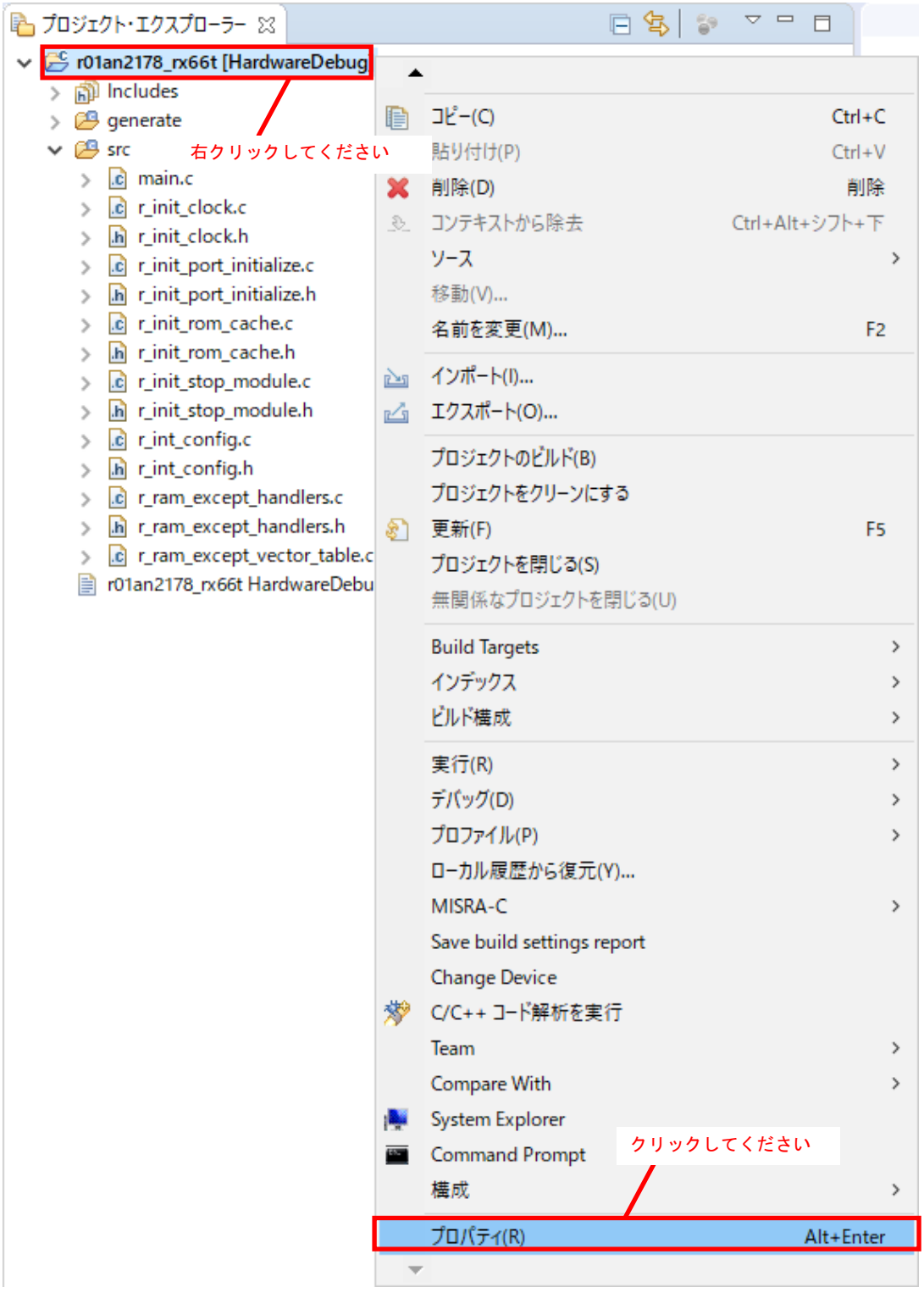


6.3.4　ポーティング先プロジェクトの設定

生成したプロジェクトのビルド設定を変更します。

1)　RAM の最終アドレスのセクションを追加

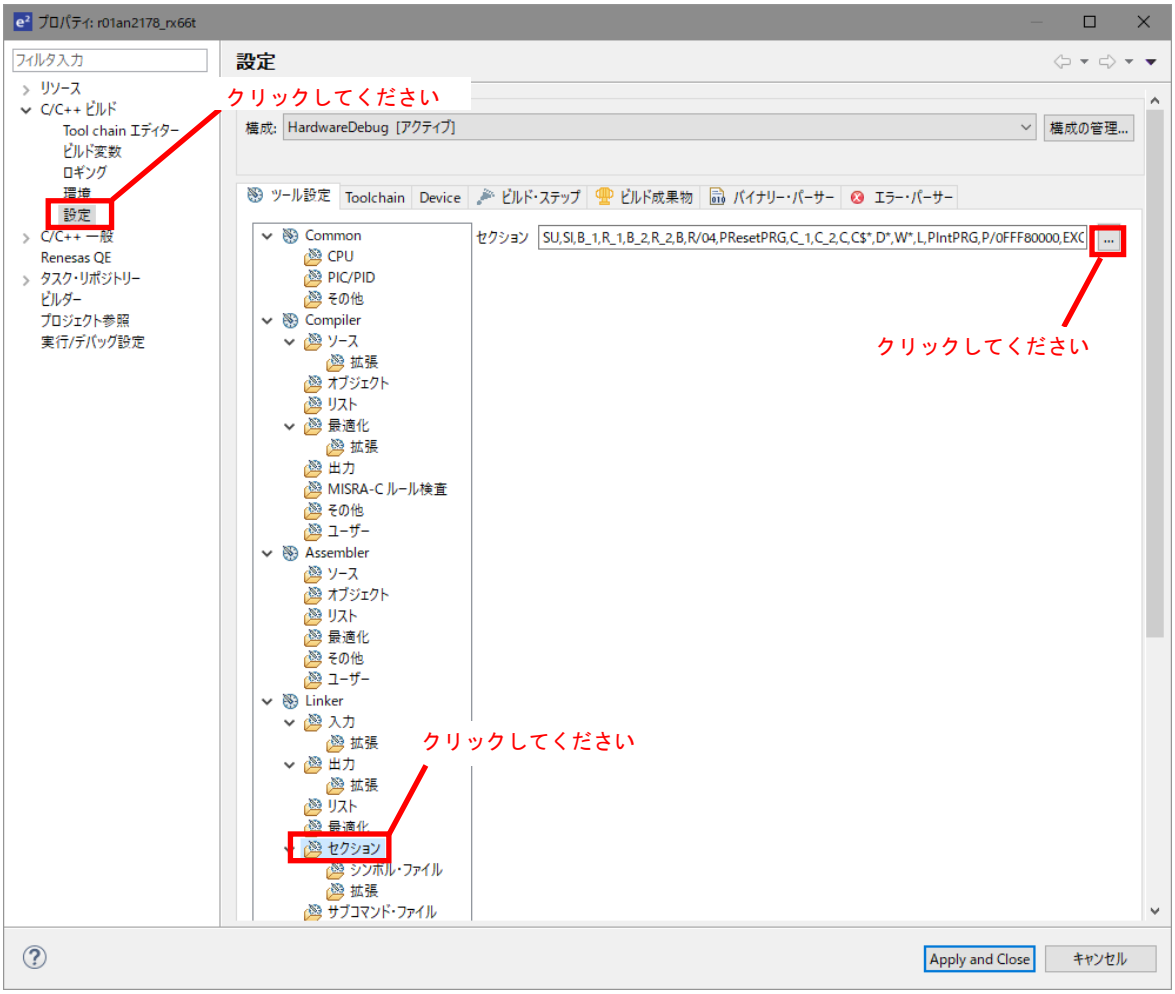
1-1) 生成したプロジェクトを右クリックして、[プロパティ(R)]をクリックしてください。



1-2) [C/C++ ビルド] -> [設定]をクリックしてください。

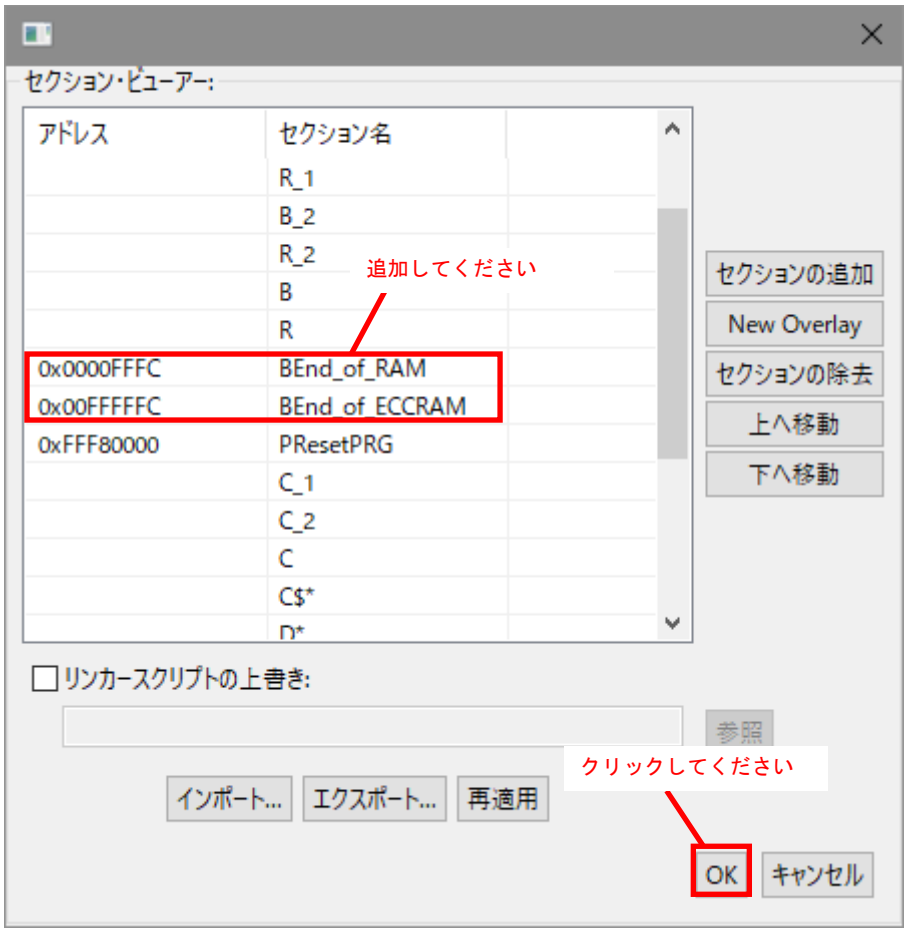
1-3) [ツール設定] -> [Linker] -> [セクション]をクリックしてください。

1-4) [セクション]の右端にある[…]をクリックしてください。



1-5) [End_of_RAM]セクションと[End_of_ECCRAM]セクションを追加してください。

1-6) [OK]をクリックしてください。



1-7) [Apply and Close]をクリックしてください。

2)　例外ベクタテーブル用のセクションを追加

2-1) 「5.1.2 例外ベクタテーブルを RAM 領域に移動する方法」を参考に例外ベクタテーブル用のセクションを追加してください。

6.3.5　ファイルの変更

本アプリケーションのサンプルコードを動作させるために、コピーした各ソースファイルを変更します。

1)　インクルードファイルのパスを変更

1-1) 初期設定例によってソースファイルのインクルードファイルパスが異なるため、ポーティング先のプロジェクトに合わせてインクルードファイルパスを見直してください。。

2)　[intprg.c]の変更

2-1) [intprg.c]に[r_int_config.h]へのインクルードパスを追加してください。

```
#include <machine.h>
#include "vect.h"
#include "../src/r_int_config.h"
#pragma section IntPRG

// Exception(Su                n)
void Excep_Super                brk(){  } */}

// Exception(Access Instruction)
void Excep_AccessInst(void){/* brk(); */}
```

追加してください

2-2) [intprg.c]の[Excep_PERIA_INTA208]関数に[peria_inta208]関数の呼び出し処理を追加してください。

```
// PERIA_INTA208
void Excep_PERIA_INTA208(void)
{
    peria_inta208();
}

// PERIA_INTA209
void Excep_PERIA_
```

追加してください

3)　[main.c]の変更

2-1) [main.c]に[r_int_config.h]へのインクルードパスを追加してください。

```
#include "r_init_clock.
#include "r_init_port_
#include "r_init_rom_cache.h"
#include "r_init_stop_module.h"
#include "r_int_config.h"

 Exported global variables and functions (to be accessed by other files)
void main (void);
```

追加してください

2-2) [main.c]の[main]関数に[R_INT_Config]関数の呼び出し処理を while 文の前に追加してください。

```
    /* ---- Initialization of the clock ---- */
    R_INIT_Clock();
                    追加してください
    /* ---- Initialization of ROM cache ---- */
    R_INIT_ROM_Cache();

    R_INT_Config();

    while (1)
    {
        /* Main loop */
    }
}
 End of function main
```

6.3.6　r_int_config.h の設定

ポーティング先の環境に合わせて[r_int_config.h]を変更します。

RX66T(Renesas Starter Kit for RX66T)にポーティングするときの設定値示します。他の RX ファミリにポーティングする場合は、ポーティング先の環境に合わせて設定値を変更してください。

1)　動作させる処理の設定

1-1) 動作させたい処理に合わせて定数 SEL_INT を設定します。例外ベクタテーブルの移動処理を動作させる場合、定数 SEL_INT に EXCEP_HANDL を設定してください。選択型割り込み処理を動作させる場合、定数 SEL_INT に SOFTWARE_CONFIG_INT を設定してください。

```
/* ==== Please select the interrupts ==== */
#define EXCEP_HANDL          (0)    /* Exception handling */
#define SOFTWARE_CO                 * Software configurable interrupts  */

/* This sample code does the exception handling.
   Please change the following settings as necessary. */
#define SEL_INT  (EXCEP_HANDL)
```

設定してください

2)　LED 点灯時/消灯時のポート出力データの設定

2-1) LED 点灯時/消灯時のポート出力データを定数 LED_ON および定数 LED_OFF に設定します。定数 LED_ON に[OUTPUT_LOW]、定数 LED_OFF に[OUTPUT_HIGH]を設定してください。

```
/* ==== Please select output data and output I/O ports ==== */
#define OUTPUT_HIGH (1) /* High output */
#define OUTPUT_LOW  (0) /* Low output */

/* This sample code                    and LED3 by low output,
   and turns off the LED0 and LED3 by high output.
   Please change the following settings as necessary. */
#define LED_ON  (OUTPUT_LOW)
#define LED_OFF (OUTPUT_HIGH)
```

設定してください

3)　LED に接続されているのポートの設定

3-1) LED0、LED3 に接続されているポートのポート出力データレジスタを定数 LED0_PODR および LED3_PODR に設定します。定数 LED0_PODR に[PORT9.PODR.BIT.B5]、
定数 LED3_PODR に[PORTE.PODR.BIT.B0]を設定してください。

```
/* This sample code controls the output d
   Please change the following setting
#define LED0_PODR   (PORT9.PODR.BIT.B5)
#define LED0_PDR    (PORT9.PDR.BIT.B5)
#define LED3_PODR   (PORTE.PODR.BIT.B0)
#define LED3_PDR    (PORTE.PDR.BIT.B0)
```

設定してください

設定してください

3-2) LED0、LED3 に接続されているポートのポート方向レジスタを定数 LED0_PDR および LED3_PDR に設定します。定数 LED0_PDR に[PORT9.PDR.BIT.B5]、定数 LED3_PDR に[PORTE.PDR.BIT.B0]を設定してください。

```
/* This sample code controls the output data of P73 and PG5.
   Please change the following settings as ---------- */
#define LED0_PODR    (PORT9.PODR.BIT.B5)
#define LED0_PDR     (PORT9.PDR.BIT.B5)
#define LED3_PODR    (PORTE.PODR.BIT.B0)
#define LED3_PDR     (PORTE.PDR.BIT.B0)
```

設定してください

設定してください

4) マルチファンクションタイマパルスユニット(MTU)の動作周波数の設定

2-1) マルチファンクションタイマパルスユニット(MTU)の動作周波数を定数 MTU_PCLK_HZ に Hz 単位で設定します。[160000000]を設定してください。

```
/* ==== Please select the MTU operating frequency ==== */
/* This sample code operates wi                    ıg frequency of 120 MHz.
   Please change the following settings as necessary. */
#define MTU_PCLK_HZ (160000000)
```

設定してください

7. サンプルコード

サンプルコードは、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。

8. 参考ドキュメント

ユーザーズマニュアル：ハードウェア

RX65N グループ、RX651 グループ ユーザーズマニュアル ハードウェア編（R01UH0659）
（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。）

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	Dec.01.2014	—	初版発行
1.01	Nov.02.2015	—	対応デバイスに RX71M グループを追加
1.10	May.31.2019	—	対応デバイスを例外ベクタテーブルと選択型割り込みを搭載する RX ファミリに変更
		4, 9, 10, 11, 23	RAM_EXFUNC_COPY のアドレスを変更 RAM_EXVECT_COPY のアドレスを変更
		5, 13, 19, 24, 25, 26	使用する選択型割り込みの割り込みベクタ番号および割り込み要因番号を変更 割り込み発生間隔と割り込み発生回数を変更
		7	LED 出力用に使用する端子を変更
		6	表 2.1 動作確認条件を変更
		14	表 5.3 サンプルコードで使用するオプション設定メモリを変更
		15	表 5.4 サンプルコードで使用する定数を変更
		16	表 5.5 static 型変数を変更
		17	5.8 関数仕様を変更
		21	5.9 フローチャートを変更
		29	6. 他 RX ファミリにサンプルコードをポーティングする方法を追加
		41	8. 参考ドキュメントを変更
		ﾌﾟﾛｸﾞﾗﾑ	・ターゲットデバイスを変更 ・ファイル削除 main.h ・ファイル追加 r_int_config.c r_int_config.h ・マクロ定義追加 LED_ON LED_OFF LED0_PODR LED0_PDR LED3_PODR LED3_PDR MTU_PCLK_HZ ・関数追加 R_INT_Config ・関数変更 port_init mtu_init software_config_int_main mtu_start mtu_stop peria_inta208 ram_excep_super_visor_inst

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。