RX ファミリ

SDRAMC を使用した SDRAM のリードライト制御

要旨

RX ファミリ搭載の SDRAM のインタフェースでは最大 128M バイト(1024M ビット)の SDRAM を直結することができ、CAS レイテンシ 1～3 の SDRAM を接続することができます。

本アプリケーションノートでは、以下のマイコン機種に搭載されている 128Mbit SDRAM(Micron 社製またはアライアンスメモリ社製)への書き込み、および読み出しを行う方法について示します。

- RX65N
- RX72M
- RX72N
- RX671

対象デバイス

・SDRAM 領域コントローラを搭載する RX ファミリ MCU

目次

1. 仕様

本アプリケーションノートのサンプルコードでは以下の処理が実装されています。

- SDRAMC を使用して、以下 128Mbit SDRAM のリード/ライトを行います。
  - ・2M-word×16bit×4bank MT48LC8M16A2P-6A : Micron 社製
  - ・1M-word×32bit×4bank MT48LC4M32B2P-6A : Micron 社製
  - ・2M-word×16bit×4bank AS4C8M16SA-7TCN : アライアンスメモリ社製
- リセット解除後、SDRAM の初期化を行った後、128Mbit の SDRAM 領域に対し、インクリメントデータを書き込みます。すべての領域に書き込み完了後、書いた値を読み出します。
- 読み出した値が期待値と一致した場合と異なる場合の動作を下記します。
  - ― Renesas Starter Kit+：読み出した値が期待値と一致した場合、LED0 を点灯します。期待値と異なる場合、LED1 を点灯します。
  - ― EK-RX671：読み出した値が期待値と一致した場合、LED１を点灯します。期待値と異なる場合、LED２を点灯します。

表 1.1 に使用する周辺機能と用途を、表 1.2 に SDRAM(MT48LC8M16A2P-6A)仕様、表 1.3 に SDRAM(MT48LC4M32B2P-6A)仕様、表 1.4 に SDRAM(AS4C8M16SA-7TCN)仕様を示します。

表 1.1　使用する周辺機能と用途

周辺機能	用途
外部バス	SDRAM との接続
I/O ポート	LED 点灯
CMT0	待ち時間用タイマ

表 1.2　SDRAM(MT48LC8M16A2P-6A)仕様

項目	内容
製品名	MT48LC8M16A2P-6A(Micron 社製)
構成	2M-word x 16bit x 4bank
容量	128M ビット
ロウアドレス	A11-A0
カラムアドレス	A8-A0
オートリフレッシュ間隔	64ms ごとの 4096 リフレッシュサイクル
CAS レイテンシ	2 または 3
初期化オートリフレッシュ回数	2 回
オートリフレッシュ期間(tRFC)	60ns(min)
ライトリカバリ期間(tWR)	22.67ns(min)(注 1)
プリチャージコマンド期間(tRP)	18ns(min)
アクティブコマンドからプリチャージコマンドまでの期間(tRAS)	42ns(min)
アクティブコマンドからリード／ライトコマンドまでの遅延時間(tRCD)	18ns(min)

注1.　ライトリカバリ期間は 1CLK+6ns です。本アプリケーションノートでは SDCLK=60MHz であるため、1CLK は 16.67ns となります。

表 1.3　SDRAM(MT48LC4M32B2P-6A)仕様

項目	内容
製品名	MT48LC4M32B2P-6A(Micron 社製)
構成	1M-word x 32bit x 4bank
容量	128M ビット
ロウアドレス	A11-A0
カラムアドレス	A7-A0
オートリフレッシュ間隔	64ms ごとの 4096 リフレッシュサイクル
CAS レイテンシ	2 または 3
初期化オートリフレッシュ回数	2 回
オートリフレッシュ期間(tRFC)	60ns(min)
ライトリカバリ期間(tWR)	19.5ns(min)(注 1)
プリチャージコマンド期間(tRP)	18ns(min)
アクティブコマンドからプリチャージコマンドまでの期間(tRAS)	42ns(min)
アクティブコマンドからリード／ライトコマンドまでの遅延時間(tRCD)	18ns(min)

注 1. ライトリカバリ期間は 1CLK+7ns です。本アプリケーションノートでは SDCLK=80MHz であるため、1CLK は 12.5ns となります。

表 1.4　SDRAM(AS4C8M16SA-7TCN)仕様

項目	内容
製品名	AS4C8M16SA-7TCN (アライアンスメモリ社製)
構成	2M-word x 16bit x 4bank
容量	128M ビット
ロウアドレス	A11-A0
カラムアドレス	A8-A0
オートリフレッシュ間隔	64ms ごとの 4096 リフレッシュサイクル
CAS レイテンシ	2 または 3
初期化オートリフレッシュ回数	2 回
Row サイクルタイム(tRC)	63ns(min)(同一バンク)
ライトリカバリ期間(tWR)	14ns(min)(注 1)
プリチャージコマンド期間(tRP)	21ns(min)
アクティブコマンドからプリチャージコマンドまでの期間(tRAS)	42ns(min)
アクティブコマンドからリード／ライトコマンドまでの遅延時間(tRCD)	21ns(min)

注 1. ライトリカバリ期間は 1CLK+6ns です。本アプリケーションノートでは SDCLK=60MHz であるため、1CLK は 16.67ns となります。

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(RX65N)

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 (PLL 2 分周) • 周辺モジュールクロック A (PCLKA): 120MHz (PLL 2 分周) • 周辺モジュールクロック B (PCLKB): 60MHz (PLL 4 分周) • 周辺モジュールクロック C (PCLKC): 60MHz (PLL 4 分周) • 周辺モジュールクロック D (PCLKD): 60MHz (PLL 4 分周) • SDCLK: 60MHz (PLL 4 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 2020-04
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 V.3.02
	コンパイルオプション 統合開発環境のデフォルト設定を使用しています
iodefine.h のバージョン	Version 2.30
エンディアン	リトルエンディアン
動作モード	内蔵 ROM 有効拡張モード
プロセッサモード	スーパバイザモード
サンプルコードのバージョン	Version 1.00
使用ボード	Renesas Starter Kit+ for RX65N-2MB (製品型名: RTK50565N2S80000BE) (SDRAM：MT48LC8M16A2P-6A 搭載品)

表 2.2　動作確認条件(RX72M)

項目	内容
使用マイコン	R5F572MNDDBD (RX72M 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): 240MHz (PLL 1 分周) • 周辺モジュールクロック A (PCLKA): 120MHz (PLL 2 分周) • 周辺モジュールクロック B (PCLKB): 60MHz (PLL 4 分周) • 周辺モジュールクロック C (PCLKC): 60MHz (PLL 4 分周) • 周辺モジュールクロック D (PCLKD): 60MHz (PLL 4 分周) • SDCLK: 80MHz (PLL 3 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 2020-04
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 V.3.02
	コンパイルオプション 統合開発環境のデフォルト設定を使用しています
iodefine.h のバージョン	Version 1.00C
エンディアン	リトルエンディアン
動作モード	内蔵 ROM 有効拡張モード
プロセッサモード	スーパバイザモード
サンプルコードのバージョン	Version 1.00
使用ボード	Renesas Starter Kit+ for RX72M (製品型名: RTK5572MNDS10000BE) (SDRAM：MT48LC4M32B2P-6A 搭載品)

表 2.3　動作確認条件(RX671:[RSK-RX671])

項目	内容
使用マイコン	R5F5671EHDFB (RX671 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 (PLL 2 分周) • 周辺モジュールクロック A (PCLKA): 120MHz (PLL 2 分周) • 周辺モジュールクロック B (PCLKB): 60MHz (PLL 4 分周) • 周辺モジュールクロック C (PCLKC): 60MHz (PLL 4 分周) • 周辺モジュールクロック D (PCLKD): 60MHz (PLL 4 分周) • SDCLK: 60MHz (PLL 4 分周)
動作電圧	3.3V(電源から供給)
統合開発環境	ルネサスエレクトロニクス製 e2studio Version 2023-10
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 V.3.05
	コンパイルオプション 統合開発環境のデフォルト設定
iodefine.h のバージョン	Version 1.00A
エンディアン	内蔵 ROM 有効拡張モード
動作モード	スーパバイザモード
プロセッサモード	Ver.1.10
サンプルコードのバージョン	3.3V(電源から供給)
使用ボード	Renesas Starter Kit+ for RX671 (製品型名: RTK55671EHS1000BE) (SDRAM：MT48LC8M16A2P-6A 搭載品)

表 2.4　動作確認条件(RX671:[EK-RX671])

項目	内容
使用マイコン	R5F5671EHDFB (RX671 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 (PLL 2 分周) • 周辺モジュールクロック A (PCLKA): 120MHz (PLL 2 分周) • 周辺モジュールクロック B (PCLKB): 60MHz (PLL 4 分周) • 周辺モジュールクロック C (PCLKC): 60MHz (PLL 4 分周) • 周辺モジュールクロック D (PCLKD): 60MHz (PLL 4 分周) • SDCLK: 60MHz (PLL 4 分周)
動作電圧	3.3V(電源から供給)
統合開発環境	ルネサスエレクトロニクス製 e2studio Version 2023-10
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 V.3.05
	コンパイルオプション 統合開発環境のデフォルト設定
iodefine.h のバージョン	Version 1.00A
エンディアン	内蔵 ROM 有効拡張モード
動作モード	スーパバイザモード
プロセッサモード	Ver.1.10
サンプルコードのバージョン	3.3V(電源から供給)
使用ボード	EK-RX671 (製品型名: RTK5EK6710S00001BE) (SDRAM：AS4C8M16SA-7TCN　搭載品)

表 2.5　動作確認条件(RX72N)

項目	内容
使用マイコン	R5F572NNDDBD (RX72N 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): 240MHz (PLL 1 分周) • 周辺モジュールクロック A (PCLKA): 120MHz (PLL 2 分周) • 周辺モジュールクロック B (PCLKB): 60MHz (PLL 4 分周) • 周辺モジュールクロック C (PCLKC): 60MHz (PLL 4 分周) • 周辺モジュールクロック D (PCLKD): 60MHz (PLL 4 分周) • SDCLK: 60MHz (PLL 4 分周)
動作電圧	3.3V(電源から供給)
統合開発環境	ルネサスエレクトロニクス製 e2studio Version 2023-10
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 V.3.05
	コンパイルオプション 統合開発環境のデフォルト設定
iodefine.h のバージョン	Version 1.00A
エンディアン	内蔵 ROM 有効拡張モード
動作モード	スーパバイザモード
プロセッサモード	Ver.1.10
サンプルコードのバージョン	3.3V(電源から供給)
使用ボード	Renesas Starter Kit+ for RX72N (製品型名: RTK5572NNHS10000BE) (SDRAM：MT48LC8M16A2P-6A　搭載品)

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 グループ、RX651 グループ 初期設定例 (R01AN3034)
- RX72M グループ 初期設定例 (R01AN4530)
- RX671 グループ 初期設定例 (R01AN5551)
- RX72N グループ 初期設定例 (R01AN4970)

最新版がある場合、最新版に差し替えて使用してください。最新版はルネサスエレクトロニクスホームページで確認および入手してください。

4.　周辺機能説明

SDRAMC について補足します。基本的な内容はユーザーズマニュアル ハードウェア編(以下、ユーザーズマニュアル)に記載しています。

4.1　SDRAMC の出力について

RX65N の SDRAMC では、SDRAM コマンド要求が発生すると、SDCLK の立ち上がりから一定時間遅延後に SDRAM 関連の端子状態が変化します。コマンドの判定は次の立ち上がりエッジで行われます。端子の出力遅延時間については、ユーザーズマニュアルの電気的特性を参照ください。

図 4.1 に SDRAM 関連端子の出力とコマンド判定タイミングを示します。

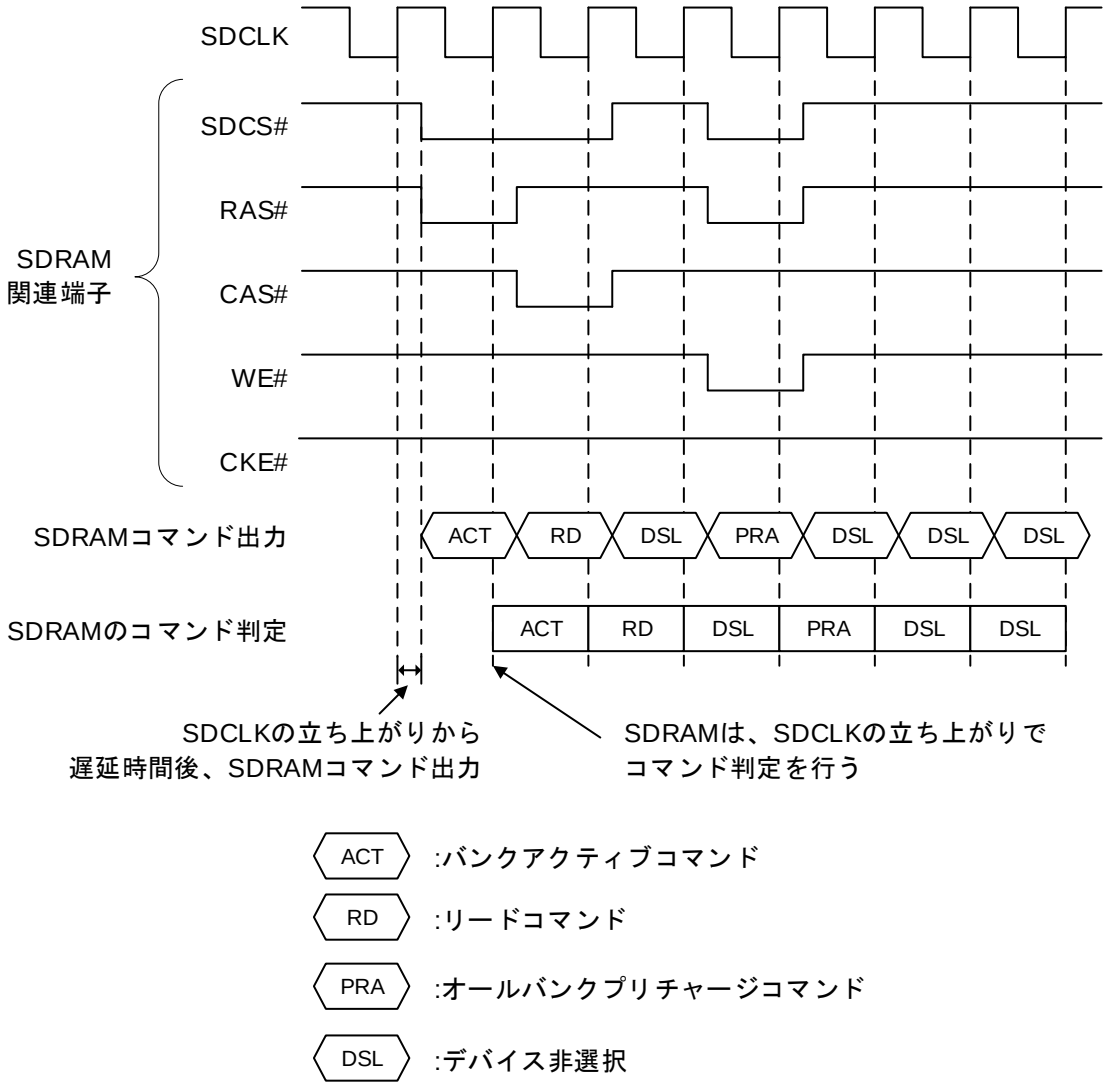
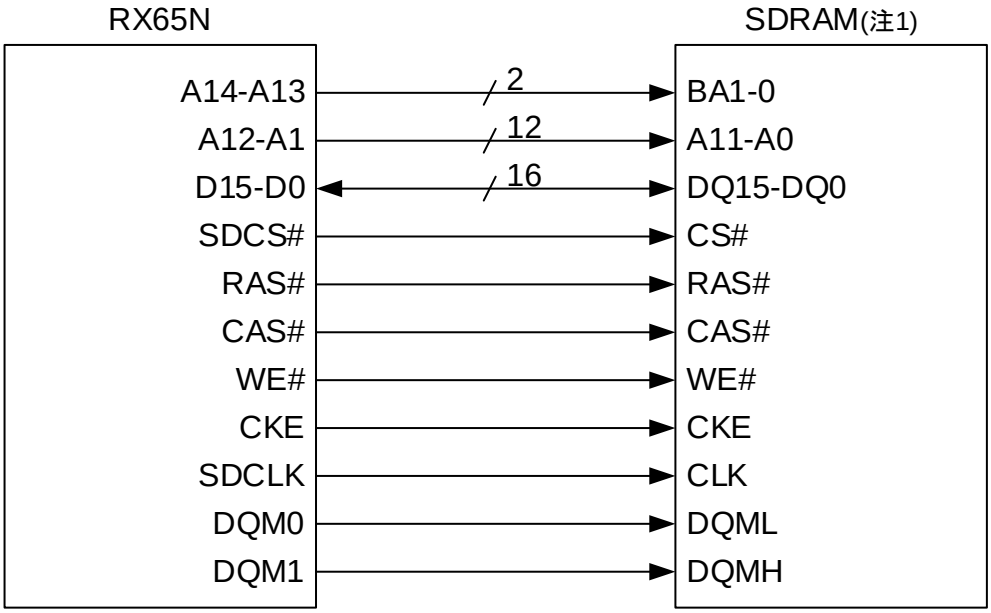


図 4.1　SDRAM 関連端子の出力とコマンド判定タイミング

5.　ハードウェア説明

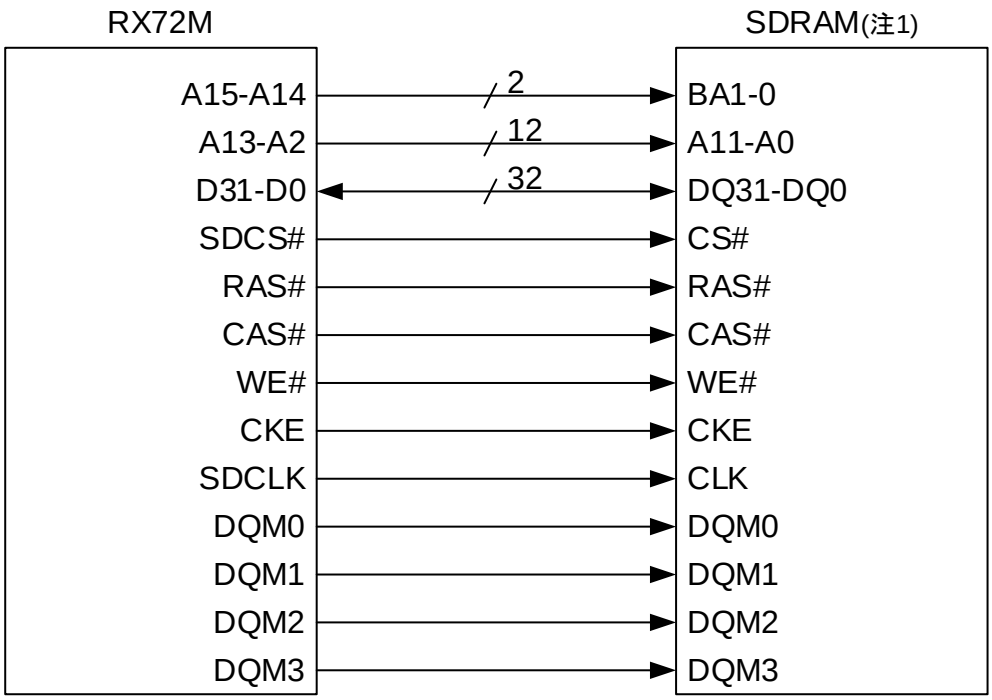
5.1　ハードウェア構成例

図 5.1～図 5.4 に接続例を示します。



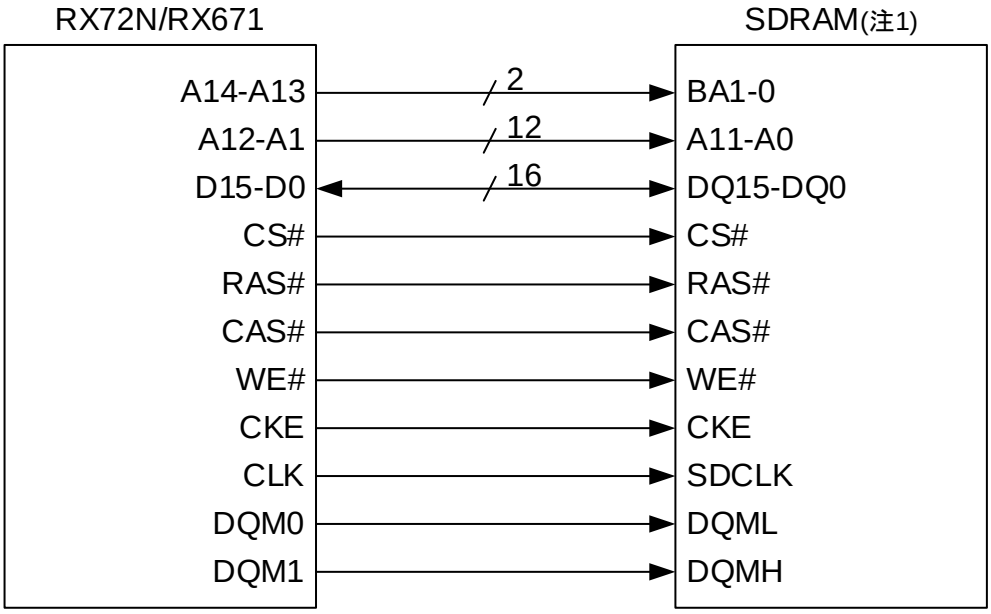
注1. MT48LC8M16A2P-6A (2M-word x 16bit x 4bank) を使用

図 5.1　MT48LC8M16A2P-6A 接続例



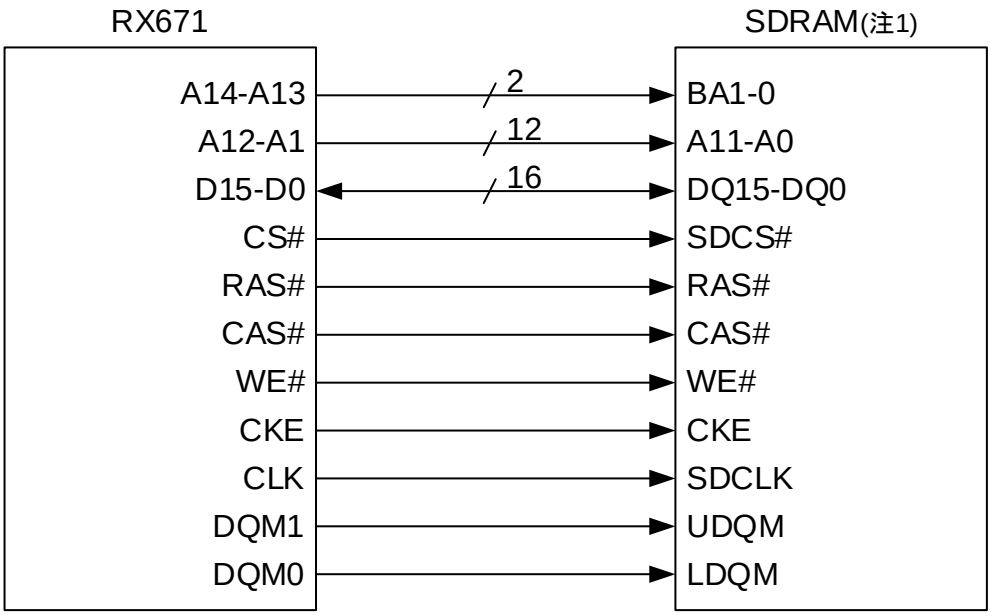
注1. MT48LC4M32B2P-6A (1M-word x 32bit x 4bank) を使用

図 5.2　MT48LC4M32B2P-6A 接続例



注1. MT48LC8M16A2P-6A (2M-word x 16bit x 4bank) を使用

図 5.3　MT48LC8M16A2P-6A 接続例



注1. AS4C8M16SA-7TCN (2M-word x 16bit x 4bank) を使用

図 5.4　AS4C8M16SA-7TCN 接続例

5.2　使用端子一覧

表 5.1～表 5.4 に使用端子と機能を示します。

表 5.1　使用端子と機能(RX65N)

端子名	入出力	内容
P73	出力	LED0 出力(ベリファイ完了)
PG7	出力	LED1 出力(ベリファイエラー)
PA7-PA0	出力	アドレス出力端子(A7-A0)
PB6-PB0	出力	アドレス出力端子(A14-A8)
PD7-PD0	入出力	データ入出力端子(D7-D0)
PE7-PE0	入出力	データ入出力端子(D15-D8)
P70	出力	SDCLK 端子出力
P61	出力	SDCS#端子出力
P62	出力	RAS#端子出力
P63	出力	CAS#端子出力
P64	出力	WE#端子出力
P65	出力	CKE 端子出力
P66	出力	DQM0 端子出力
P67	出力	DQM1 端子出力

表 5.2　使用端子と機能(RX72M)

端子名	入出力	内容
P42	出力	LED0 出力(ベリファイ完了)
PH0	出力	LED1 出力(ベリファイエラー)
PA7-PA2	出力	アドレス出力端子(A7-A2)
PB7-PB0	出力	アドレス出力端子(A15-A8)
PD7-PD0	入出力	データ入出力端子(D7-D0)
PE7-PE0	入出力	データ入出力端子(D15-D8)
P97-P90	入出力	データ入出力端子(D23-D16)
PG7-PG0	入出力	データ入出力端子(D31-D24)
P70	出力	SDCLK 端子出力
P61	出力	SDCS#端子出力
P62	出力	RAS#端子出力
P63	出力	CAS#端子出力
P64	出力	WE#端子出力
P65	出力	CKE 端子出力
P66	出力	DQM0 端子出力
P67	出力	DQM1 端子出力
PA0	出力	DQM2 端子出力
PA1	出力	DQM3 端子出力

表 5.3　使用端子と機能(RSK-RX671)

端子名	入出力	内容
P17	出力	LED0 出力(ベリファイ完了)
PF9	出力	LED1 出力(ベリファイエラー)
PA7-PA0	出力	アドレス出力端子(A7-A0)
PB6-PB0	出力	アドレス出力端子(A14-A8)
PD7-PD0	入出力	データ入出力端子(D7-D0)
PE7-PE0	入出力	データ入出力端子(D15-D8)
P70	出力	SDCLK 端子出力
P61	出力	SDCS#端子出力
P62	出力	RAS#端子出力
P63	出力	CAS#端子出力
P64	出力	WE#端子出力
P65	出力	CKE 端子出力
P66	出力	DQM0 端子出力
P67	出力	DQM1 端子出力

表 5.4　使用端子と機能(EK-RX671)

端子名	入出力	内容
P73	出力	LED1 出力(ベリファイ完了)
PG7	出力	LED2 出力(ベリファイエラー)
PA7-PA0	出力	アドレス出力端子(A7-A0)
PB6-PB0	出力	アドレス出力端子(A14-A8)
PD7-PD0	入出力	データ入出力端子(D7-D0)
PE7-PE0	入出力	データ入出力端子(D15-D8)
P70	出力	SDCLK 端子出力
P61	出力	SDCS#端子出力
P62	出力	RAS#端子出力
P63	出力	CAS#端子出力
P64	出力	WE#端子出力
P65	出力	CKE 端子出力
P66	出力	DQM0 端子出力
P67	出力	DQM1 端子出力

表 5.5　使用端子と機能(RSK-RX72N)

端子名	入出力	内容
P71	出力	LED0 出力(ベリファイ完了)
PH6	出力	LED1 出力(ベリファイエラー)
PA7-PA0	出力	アドレス出力端子(A7-A0)
PB6-PB0	出力	アドレス出力端子(A14-A8)
PD7-PD0	入出力	データ入出力端子(D7-D0)
PE7-PE0	入出力	データ入出力端子(D15-D8)
P70	出力	SDCLK 端子出力
P61	出力	SDCS#端子出力
P62	出力	RAS#端子出力
P63	出力	CAS#端子出力
P64	出力	WE#端子出力
P65	出力	CKE 端子出力
P66	出力	DQM0 端子出力
P67	出力	DQM1 端子出力

6.　ソフトウェア説明

6.1　動作概要

使用する SDRAM に合わせて初期化シーケンス、SDRAM のモードレジスタ、オートリフレッシュ間隔、SDRAM のリード／ライトタイミングの設定を行います。

本動作概要では、Micron 社製 SDRAM (MT48LC8M16A2P-6A)を使用した場合の、SDRAMC 設定例を示します。

6.1.1　SDRAM 初期化シーケンスの設定

リセット解除後､SDRAM を使用する前に､SDRAM の初期化を行う必要があります｡初期化シーケンスは、SDRAM のデータシートに記載されている、初期化オートリフレッシュ間隔(tRFC)、初期化リフレッシュ回数、初期化プリチャージサイクル(tRP)などを考慮して行います。

図 6.1 に SDRAM (MT48LC8M16A2P-6A)の初期化タイミングを、

表 6.1 に SDRAM (MT48LC8M16A2P-6A)接続時の SDRAMC 初期化シーケンス設定例を示します。

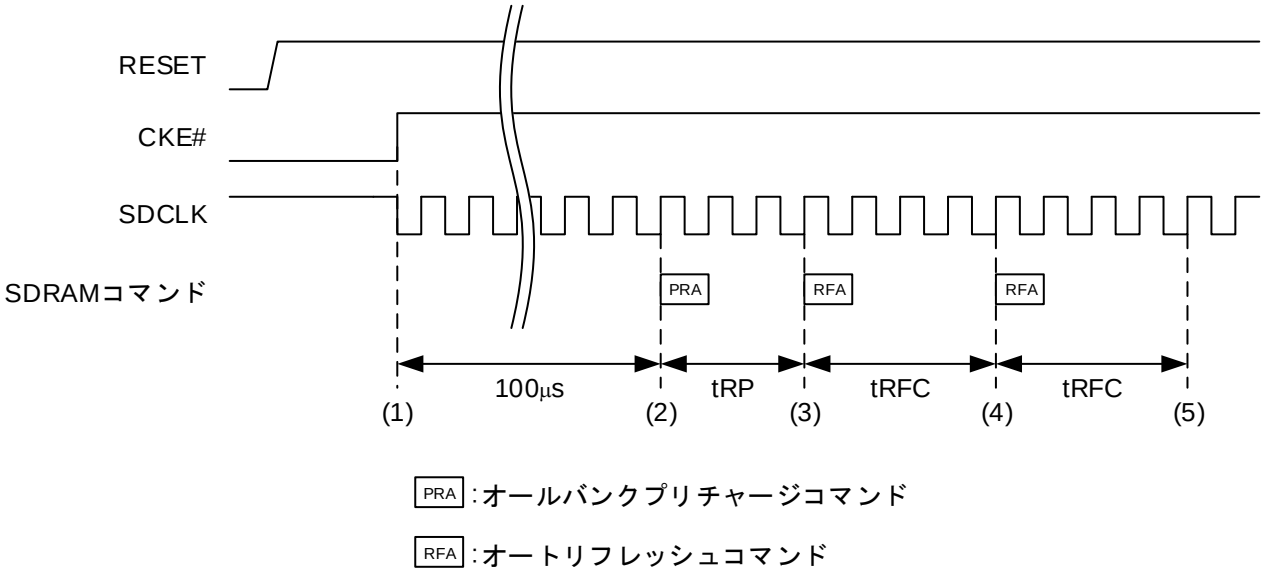


図 6.1　SDRAM (MT48LC8M16A2P-6A)の初期化タイミング

(1)　リセット解除後、SDRAM 関連端子の設定を行い、SYSCR0.EXBE ビットを“1”(外部バス有効)にすると、SDCLK 端子の出力を開始、CKE#端子から“H”が出力されます。(CKE#端子は、電源投入後“L”を入力する必要があるため、抵抗を介して GND に接続しています。)

(2)　クロック出力後、100μs 以上待ちます。(待ち時間の間、デバイス非選択コマンドを出力します。)
その後、SDIR レジスタに初期化タイミングを設定し、SDICR.INIRQ ビットを“1”にすると、オールバンクプリチャージコマンドが出力されます。

(3)　オールバンクプリチャージコマンド出力後、SDIR.PRC[2:0]ビットで設定したサイクル後にオートリフレッシュコマンドが出力されます。SDIR.PRC[2:0]ビットの値は、tRP 以上になるよう設定してください。

(4)　オートリフレッシュコマンド出力後、SDIR.ARFI[3:0]ビットで設定したサイクル数の待ち時間が挿入されます。SDIR.ARFI[3:0]ビットの値は、tRFC 以上になるよう設定してください。
SDIR.ARFC[3:0]ビットで初期化リフレッシュ回数を 2 回以上設定していた場合、再度オートリフレッシュコマンドが出力されます。

(5)　SDIR.ARFC[3:0]ビットで設定した回数、オートリフレッシュコマンドが出力されると、初期化シーケンスが完了します。

表 6.1　SDRAM (MT48LC8M16A2P-6A)接続時の SDRAMC 初期化シーケンス設定例

SDRAM タイミング	記号	内容	RX65N での SDRAMC 設定
SDCLK 入力後、 プリチャージコマンド 入力までの待ち時間	-	100μs	SDCLK 出力開始後、ソフトウェアで 100μs 待ってから、初期化シーケンスを開始
初期化プリチャージ サイクル	tRP	18ns(min)	SDIR.PRC[2:0]=“000b”：3 サイクル (SDCLK=60MHz のため、約 50ns)
初期化オート リフレッシュ間隔	tRFC	60ns(min)	SDIR.ARFI[3:0]=“0001b”：4 サイクル (SDCLK=60MHz のため、約 67ns)
初期化リフレッシュ回数	-	2 回	SDIR.ARFC[3:0]=“0010b”：2 回

6.1.2　SDRAM モードレジスタの設定

SDRAM の初期化後、SDRAM のモードを設定する必要があります。モードの設定は初期化後、1 回行ってください。SDRAM モードレジスタ(SDMOD)に値を書き込むと、モードレジスタ設定コマンドが出力されます。設定する値は、SDRAM のデータシートを参照してください。

表 6.2 に SDRAM (MT48LC8M16A2P-6A)の SDRAM モードレジスタを、図 6.2 にモードレジスタ設定コマンド動作タイミングを示します。

表 6.2　SDRAM (MT48LC8M16A2P-6A)の SDRAM モードレジスタ

ビット	シンボル	内容
b2-b0	Burst Length	バースト長の選択 0 0 0: 1 0 0 1: 2 0 1 0: 4 0 1 1: 8 1 1 1: Full Page (b3 = 1 の場合のみ) 上記以外の値を設定しないでください。
b3	Burst Type	バースト種別の選択 0: Sequential 1: Interleaved
b6-b4	CAS Latency	CAS レイテンシの選択 0 1 0: 2 0 1 1: 3 上記以外の値を設定しないでください。
b8-b7	Operating Mode	0 0: Standard Operation 上記以外の値を設定しないでください。
b9	Write Burst Mode	ライトバーストモードの選択 0: Programmed Burst Length 1: Single Location Access
b11-b10	Reserved	“00b”を書き込んでください。

RX65N の SDRAMC ではバースト長 1 で動作します。バースト長 1 以外を設定した場合、動作は保障されません。

本アプリケーションノートでは、“230h”(バースト長 1、CAS レイテンシ 3 サイクル)で使用します。

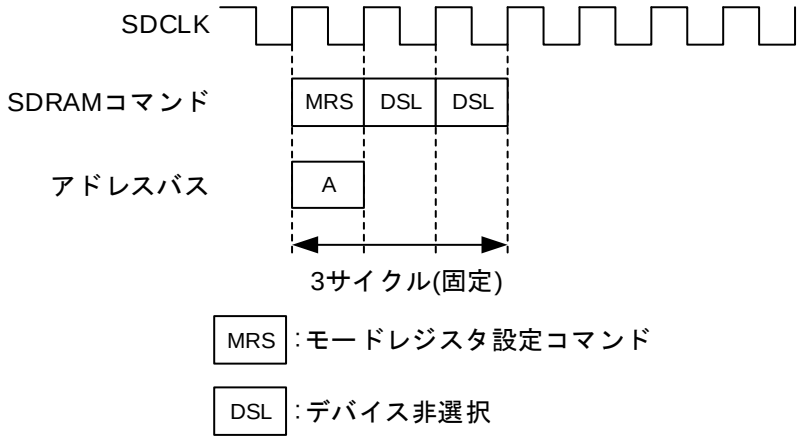


図 6.2　モードレジスタ設定コマンド動作タイミング

6.1.3　オートリフレッシュ間隔の設定

SDRAM のデータを保持するために、リフレッシュサイクル(tREF)の間に、ロウアドレス回数分のリフレッシュを行う必要があります。オートリフレッシュは、SDRAM のデータシートに記載されている、リフレッシュサイクル(tREF)、ロウアドレス数、オートリフレッシュ解除サイクル(tRFC)などを考慮して行います。

表 6.3 に SDRAM (MT48LC8M16A2P-6A)のオートリフレッシュタイミングを、図 6.3 にオートリフレッシュ動作タイミングを示します。

表 6.3　SDRAM (MT48LC8M16A2P-6A)のオートリフレッシュタイミング

SDRAM タイミング	記号	内容	RX65N での SDRAMC 設定
リフレッシュサイクル	tREF	64ms	(オートリフレッシュ間隔の計算式で使用)
ロウアドレス数	-	4,096	(オートリフレッシュ間隔の計算式で使用)
オートリフレッシュ間隔	-	15.625μs (tREF/ロウアドレス数)	SDRFCR.RFC[11:0]=“03A7h”：936 サイクル (SDCLK=60MHz のため、15.6μs)
オートリフレッシュ解除サイクル	tRFC	60ns(min)	SDRFCR.REFW[3:0]=“0011b”：4 サイクル (SDCLK=60MHz のため、約 67ns)

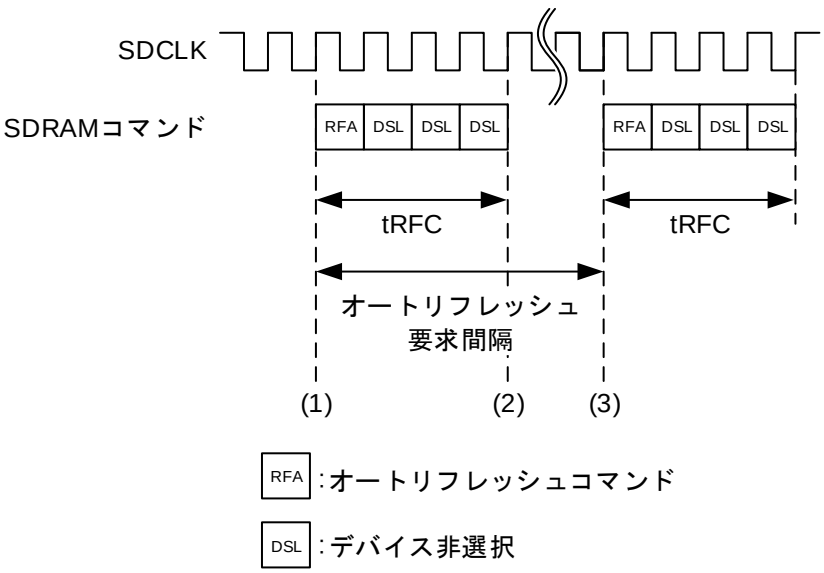


図 6.3　オートリフレッシュ動作タイミング

(1)　SDRFEN.RFEN ビットを“1”(オートリフレッシュ有効)にすると、オートリフレッシュコマンドが出力されます。

(2)　オートリフレッシュコマンド出力後、SDRFCR.REFW[3:0]ビットで設定したサイクルまで、デバイス非選択が出力されます。SDRFCR.REFW[3:0]ビットの値は、tRFC 以上になるよう設定してください。

(3)　SDRFCR.RFC[11:0]ビットで設定したサイクルごとに、オートリフレッシュコマンドが出力されます。SDRFCR.RFC[11:0]ビットの値は、オートリフレッシュ間隔(tREF/ロウアドレス数)以内になるよう設定してください。

6.1.4　SDRAM リード／ライトタイミングの設定

SDRAM のリード／ライトタイミングを、SDRAM のカラムレイテンシ(CL)設定、ライトリカバリ期間(tWR)、プリチャージコマンド期間(tRP)、ロウアクティブ期間(tRAS)、ロウカラムレイテンシ(tRCD)を考慮して設定します。

表 6.4 に SDRAM (MT48LC8M16A2P-6A)接続時のリード/ライトタイミングを、図 6.4 にリードタイミングを、

図 6.5 にライトタイミングを示します。

表 6.4　SDRAM (MT48LC8M16A2P-6A)接続時のリード/ライトタイミング

SDRAM タイミング	記号	内容	RX65N での SDRAMC 設定
カラムレイテンシ(注 2)	-	2 または 3(注 1)	SDTR.CL[2:0]=“011b” : 3 サイクル
ライトリカバリ期間	tWR	22.67ns(min)	SDTR.WR=“1” : 2 サイクル (SDCLK=60MHz のため、約 33ns)
ロウプリチャージ期間	tRP	18ns(min)	SDTR.RP[2:0]=“001b” : 2 サイクル (SDCLK=60MHz のため、約 33ns)
ロウアクティブ期間 (注 2)	tRAS	42ns(min)	SDTR.RAS[2:0]=“010b” : 3 サイクル (SDCLK=60MHz のため、約 50ns)
ロウカラムレイテンシ (注 2)	tRCD	18ns(min)	SDTR.RCD[1:0]=“01b” : 2 サイクル (SDCLK=60MHz のため、約 33ns)

(注 1) SDRAM モード設定にて、3 を選択。
(注 2) ロウアクティブ期間の設定は、ロウカラムレイテンシ(SDTR.RCD[1:0])＋カラムレイテンシ (SDTR.CL[2:0]) 以下に設定してください。

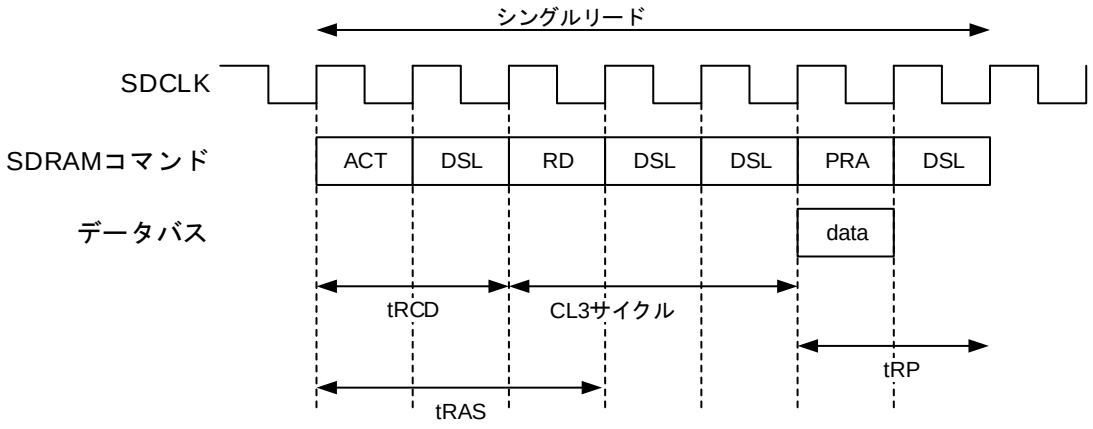


図 6.4　リードタイミング(表 6.4 で示した SDRAMC 設定時)

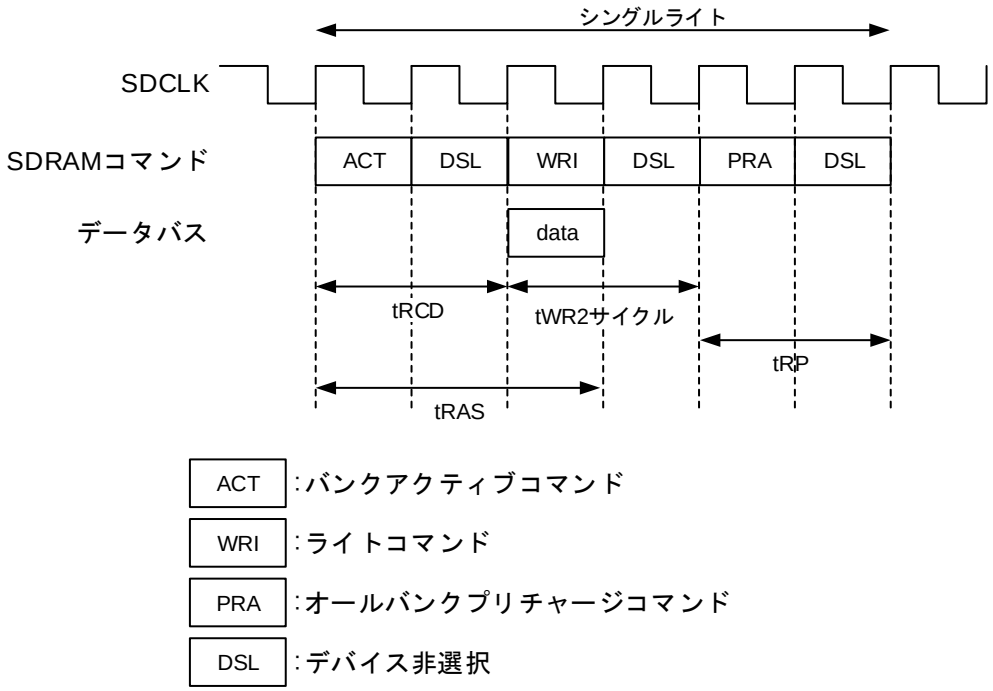


図 6.5　ライトタイミング(表 6.4 で示した SDRAMC 設定時)

6.2　ファイル構成

表 6.5 にサンプルコードで使用するファイルを示します。なお、本アプリケーションのサンプルコードはスマート・コンフィグレータを使用しているプロジェクトを RX671、RX72N、RX65N 及び RX72M、スマート・コンフィグレータを使用していないプロジェクトを RX65N 及び RX72M で用意しています。

統合開発環境及びスマート・コンフィグレータのコード生成機能によって生成されたソースコードをそのまま使用しているファイルは除きます。

表 6.5　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	スマート・コンフィグレータを使用しているかで処理が異なる
r_init_stop_module.c	リセット後に動作している周辺機能の停止	スマート・コンフィグレータを使用していないプロジェクトのみ
r_init_stop_module.h	r_init_stop_module.c のヘッダファイル	スマート・コンフィグレータを使用していないプロジェクトのみ
r_init_port_initialize.c	存在しないポートの初期設定	スマート・コンフィグレータを使用していないプロジェクトのみ
r_init_port_initialize.h	r_init_port_initialize.c のヘッダファイル	スマート・コンフィグレータを使用していないプロジェクトのみ
r_init_clock.c	クロック初期設定	スマート・コンフィグレータを使用していないプロジェクトのみ
r_init_clock.h	r_init_clock.c のヘッダファイル	スマート・コンフィグレータを使用していないプロジェクトのみ
r_init_rom_cache.c	ROM キャッシュ初期設定	RX72M のスマート・コンフィグレータを使用していないプロジェクトのみ
r_init_rom_cache.h	r_init_rom_cache.c のヘッダファイル	RX72M のスマート・コンフィグレータを使用していないプロジェクトのみ
r_cmt_wait.c	CMT を使用した時間待ち処理	スマート・コンフィグレータを使用していないプロジェクトのみ
r_cmt_wait.h	r_cmt_wait.c のヘッダファイル	スマート・コンフィグレータを使用していないプロジェクトのみ
r_sdram_api.c	SDRAMC の初期設定	スマート・コンフィグレータを使用していないプロジェクトのみ
r_sdram_api.h	r_sdram_api.c のヘッダファイル	スマート・コンフィグレータを使用していないプロジェクトのみ

6.3　オプション設定メモリ

表 6.6、表 6.7 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 6.6　RX65N 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 5D07h～FE7F 5D04h	FFFF FFFFh	リセット後、IWDT は停止 リセット後、WDT は停止
OFS1	FE7F 5D0Bh～FE7F 5D08h	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 5D03h～FE7F 5D00h	FFFF FFFFh	リトルエンディアン

表 6.7　RX72M 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 5D07h～FE7F 5D04h	FFFF FFFFh	リセット後、IWDT は停止 リセット後、WDT は停止
OFS1	FE7F 5D0Bh～FE7F 5D08h	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 5D03h～FE7F 5D00h	FFFF FFFFh	リトルエンディアン リニアモード

6.4　定数一覧

表 6.8、表 6.9 にサンプルコードで使用する定数を示します。

表 6.8　サンプルコードで使用する定数 (1/2)

定数名	設定値	内容
LED0_REG_PODR(注 3)	PORT7.PODR.BIT.B3(注 1) PORT4.PODR.BIT.B2(注 2)	LED0 出力データ格納ビット
LED0_REG_PDR(注 3)	PORT7.PDR.BIT.B3(注 1) PORT4.PDR.BIT.B2(注 2)	LED0 方向制御ビット
LED0_REG_PMR(注 3)	PORT7.PMR.BIT.B3(注 1) PORT4.PMR.BIT.B2(注 2)	LED0 端子モード制御ビット
LED1_REG_PODR(注 3)	PORTG.PODR.BIT.B7(注 1) PORTH.PODR.BIT.B0(注 2)	LED1 出力データ格納ビット
LED1_REG_PDR(注 3)	PORTG.PDR.BIT.B7(注 1) PORTH.PDR.BIT.B0(注 2)	LED1 方向制御ビット
LED1_REG_PMR(注 3)	PORTG.PMR.BIT.B7(注 1) PORTH.PMR.BIT.B0(注 2)	LED1 端子モード制御ビット
LED_ON(注 3)	0	LED 出力データ:　点灯
LED_OFF(注 3)	1	LED 出力データ:　消灯
SDRAM_TOP	(void*)(0x08000000)	SDRAM 領域の先頭番地
SDRAM_END	(void*)(0x09000000)	SDRAM 領域の最終番地
R_WT_CMT_CLOCK(注 3)	60000000L	CMT カウントソース周波数(PCLK)
R_WT_CMT_DIVIDE(注 3)	32L	CMT カウントソースの分周比
R_WT_BASE_US(注 3)	1000000L	1μs 単位での待ち時間計算値
R_WT_BASE_MS(注 3)	1000L	1ms 単位での待ち時間計算値
SDRAM_REG_MPC_PFAOE0(注 3)	0x7F(注 1) 0xFF(注 2)	MPC.PFAOE0 レジスタへの設定値
SDRAM_REG_MPC_PFAOE1(注 3)	0x00	MPC.PFAOE1 レジスタへの設定値
SDRAM_REG_MPC_PFBCR0(注 3)	0x11(注 1) 0x31(注 2)	MPC.PFBCR0 レジスタへの設定値
SDRAM_REG_MPC_PFBCR1(注 3)	0xD0	MPC.PFBCR1 レジスタへの設定値
SDRAM_REG_MPC_PFBCR2(注 3)	0x00	MPC.PFBCR2 レジスタへの設定値
SDRAM_REG_MPC_PFBCR3(注 3)	0x00(注 1) 0x40(注 2)	MPC.PFBCR3 レジスタへの設定値

注 1.RX65N のサンプルコードでの設定値です。
注 2.RX72M のサンプルコードでの設定値です。
注 3.スマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

表 6.9　サンプルコードで使用する定数 (2/2)

定数名	設定値	内容
SDRAM_REG_BSC_SDCCR(注 3)	0x00(注 1) 0x10(注 2)	BSC.SDCCR レジスタへの設定値
SDRAM_REG_BSC_SDCMOD(注 3)	0x00	BSC.SDCMOD レジスタへの設定値
SDRAM_REG_BSC_SDAMOD(注 3)	0x00	BSC.SDAMOD レジスタへの設定値
SDRAM_REG_BSC_SDRFCR(注 3)	0x33A7(注 1) 0x44E0(注 2)	BSC.SDRFCR レジスタへの設定値
SDRAM_REG_BSC_SDIR(注 3)	0x0021(注 1) 0x0022(注 2)	BSC.SDIR レジスタへの設定値
SDRAM_REG_BSC_SDADR(注 3)	0x01(注 1) 0x00(注 2)	BSC.SDADR レジスタへの設定値
SDRAM_REG_BSC_SDTR(注 3)	0x00021303(注 1) 0x00031303(注 2)	BSC.SDTR レジスタへの設定値
SDRAM_REG_BSC_SDMOD(注 3)	0x0230	BSC.SDMOD レジスタへの設定値
LED0_PORT_PIN(注 4)	GPIO_PORT_7_PIN_3(注 1) GPIO_PORT_4_PIN_2(注 2) GPIO_PORT_1_PIN_7(注 5) GPIO_PORT_7_PIN_1(注 7)	LED0 に対応する端子の列挙子
LED1_PORT_PIN(注 4)	GPIO_PORT_G_PIN_7(注 1) GPIO_PORT_H_PIN_0(注 2) GPIO_PORT_F_PIN_5(注 5) GPIO_PORT_8_PIN_2(注 6) GPIO_PORT_H_PIN_6(注 7)	LED1 に対応する端子の列挙子
LED2_PORT_PIN(注 4) (注 8)	GPIO_PORT_7_PIN_3	LED2 に対応する端子の列挙子

注 1.RX65N のサンプルコードでの設定値です。
注 2.RX72M のサンプルコードでの設定値です。
注 3.スマート・コンフィグレータを使用していないプロジェクトのみ定義されます。
注 4.スマート・コンフィグレータを使用しているプロジェクトのみ定義されます。
注 5. RX671(RSK-RX671)のサンプルコードでの設定値です。
注 6. RX671(EK-RX671) のサンプルコードでの設定値です。
注 7.RX72N のサンプルコードでの設定値です。
注 8.EK-RX671 向けのプロジェクトのみ定義されます。

6.5　変数一覧

表 6.10 にグローバル変数を示します。

表 6.10　グローバル変数

型	変数名	内容	使用関数
static volatile uint8_t	s_g_cmt_event_flag	コンペアマッチイベント発生フラグ	main cmt_event_cb

注.スマート・コンフィグレータを使用しているプロジェクトのみ定義されます。

6.6　関数一覧

表 6.11 に関数を示します。なお、スマート・コンフィグレータのコード生成機能によって生成されたソースコードをそのまま使用している関数については省略しています。

表 6.11　関数

関数名	概要
main	メイン処理
port_init	ポート初期設定
R_INIT_StopModule(注 3)	リセット後に動作している周辺機能の停止
R_INIT_Port_Initialize(注 3)	存在しないポートの初期設定
R_INIT_Clock(注 3)	クロック初期設定
R_INIT_ROM_Cache(注 2) (注 3)	ROM キャッシュ初期設定
R_INIT_CMT_Wait(注 3)	待ち時間用タイマ初期設定
R_CMT_Wait(注 3)	CMT による時間待ち処理
R_WAIT_US(注 3)	CMT による時間待ち処理(μs 単位)(注 1)
R_WAIT_MS(注 3)	CMT による時間待ち処理(ms 単位)(注 1)
R_SDRAM_Init(注 3)	SDRAMC 初期設定
sdram_verify_err	SDRAM ベリファイエラー処理
cmt_event_cb(注 4)	コンペアマッチイベントコールバック処理

注 1.本関数は関数形式マクロです。
注 2.RX72M のサンプルコードでのみ定義されます。
注 3.スマート・コンフィグレータを使用していないプロジェクトのみ定義されます。
注 4.スマート・コンフィグレータを使用しているプロジェクトのみ定義されます。

6.7　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、SDRAM の初期化を行い、SDRAM への書き込み、読み出しを行います。
引　数	なし
リターン値	なし
備　考	スマート・コンフィグレータを使用しているプロジェクトとスマート・コンフィグレータを使用していないプロジェクトで処理が異なります。

port_init	
概　要	ポート初期設定
ヘッダ	なし
宣　言	static void port_init(void)
説　明	ポートの初期設定を行います。
引　数	なし
リターン値	なし
備　考	スマート・コンフィグレータを使用しているプロジェクトとスマート・コンフィグレータを使用していないプロジェクトで処理が異なります。

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」「RX72M グループ 初期設定例」を参照してください。 本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

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	RX65N のサンプルコードでは、176 ピン版(PIN_SIZE=176)に設定しています。 RX72M のサンプルコードでは、224 ピン版(PIN_SIZE=224)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」「RX72M グループ 初期設定例」を参照してください。 本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

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、システムクロックを PLL 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」「RX72M グループ 初期設定例」を参照してください。 本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

R_INIT_ROM_Cache	
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_ROM_Cache.h
宣　言	void R_INIT_ROM_Cache(void)
説　明	ノンキャッシャブル領域を設定後、ROM キャッシュの動作許可を行います。
引　数	なし
リターン値	なし
備　考	サンプルコードでは、ROM キャッシュの動作許可のみ有効にしています。 この関数はシステムの起動後、ROM キャッシュ動作禁止の状態で呼び出されることを前提としています。 ROM キャッシュ動作許可後にノンキャッシャブル領域を設定するには ROM キャッシュ動作禁止にした後、この関数を呼び出してください。 本関数の詳細は、アプリケーションノート「RX72M グループ 初期設定例」を参照してください。 本関数は RX72M のスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

R_SDRAM_Init	
概　要	SDRAMC 初期設定
ヘッダ	なし
宣　言	void R_SDRAM_Init (void)
説　明	SDRAMC の初期設定を行います。
引　数	なし
リターン値	なし
備　考	本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

sdram_verify_err	
概　要	SDRAM ベリファイエラー処理
ヘッダ	なし
宣　言	static void sdram_verify_err(void)
説　明	SDRAM ベリファイエラー発生時、LED1 を点灯してループ処理を実行します。
引　数	なし
リターン値	なし
備　考	スマート・コンフィグレータを使用しているプロジェクトとスマート・コンフィグレータを使用していないプロジェクトで処理が異なります。

R_INIT_CMT_Wait	
概　要	待ち時間用タイマ初期設定
ヘッダ	r_cmt_wait.h
宣　言	void R_INIT_CMT_Wait (void)
説　明	待ち時間用タイマ(CMT0)の初期設定を行います。
引　数	なし
リターン値	なし
備　考	本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

R_CMT_Wait	
概　要	CMT による時間待ち処理
ヘッダ	r_cmt_wait.h
宣　言	void R_CMT_Wait (uint16_t cnt)
説　明	引数で指定した時間待ちます。
引　数	uint16_t cnt　　待ち時間
リターン値	なし
備考	R_CMT_WAIT_US(t_us)または、R_CMT_WAIT_MS(t_ms)で、本関数を使用します。 本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

R_CMT_WAIT_US	
概　要	CMT による時間待ち処理(μs 単位)
ヘッダ	r_cmt_wait.h
宣　言	R_CMT_WAIT_US(t_us)
説　明	引数で指定した時間(μs)待ちます。
引　数	u_int16 t_us　　待ち時間(μs)
リターン値	なし
備考	本関数は関数形式マクロです。 #define R_CMT_WAIT_US(t_us)　R_ CMT_Wait(t_us * (R_WT_CMT_CLOCK / R_WT_BASE_US) / R_WT_CMT_DIVIDE) 本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

R_ CMT_WAIT_MS	
概　要	CMT による時間待ち処理(ms 単位)
ヘッダ	r_cmt_wait.h
宣　言	R_CMT_WAIT_MS(t_ms)
説　明	引数で指定した時間(ms)待ちます。 本関数はスマート・コンフィグレータを使用していないプロジェクトのみで呼び出されます。
引　数	u_int16 t_ms　　待ち時間(ms)
リターン値	なし
備考	本関数は関数形式マクロです。 #define R_CMT_WAIT_MS(t_ms)　R_CMT_Wait(t_ms * (R_WT_CMT_CLOCK / R_WT_BASE_MS) / R_WT_CMT_DIVIDE) 本関数はスマート・コンフィグレータを使用していないプロジェクトのみ定義されます。

cmt_event_cb	
概　要	コンペアマッチイベントコールバック処理
ヘッダ	なし
宣　言	void cmt_event_cb (void* pdata)
説　明	コンペアマッチイベント発生時、s_g_cmt_event_flag に”1”を設定します。
引　数	CMT のチャネル番号
リターン値	なし
備考	本関数はスマート・コンフィグレータを使用しているプロジェクトのみ定義されます。

6.8　フローチャート

6.8.1　スマート・コンフィグレータを使用していないサンプルコード

6.8.1.1　メイン処理

図 6.6 にスマート・コンフィグレータを使用していないサンプルコードのメイン処理のフローチャートを示します。

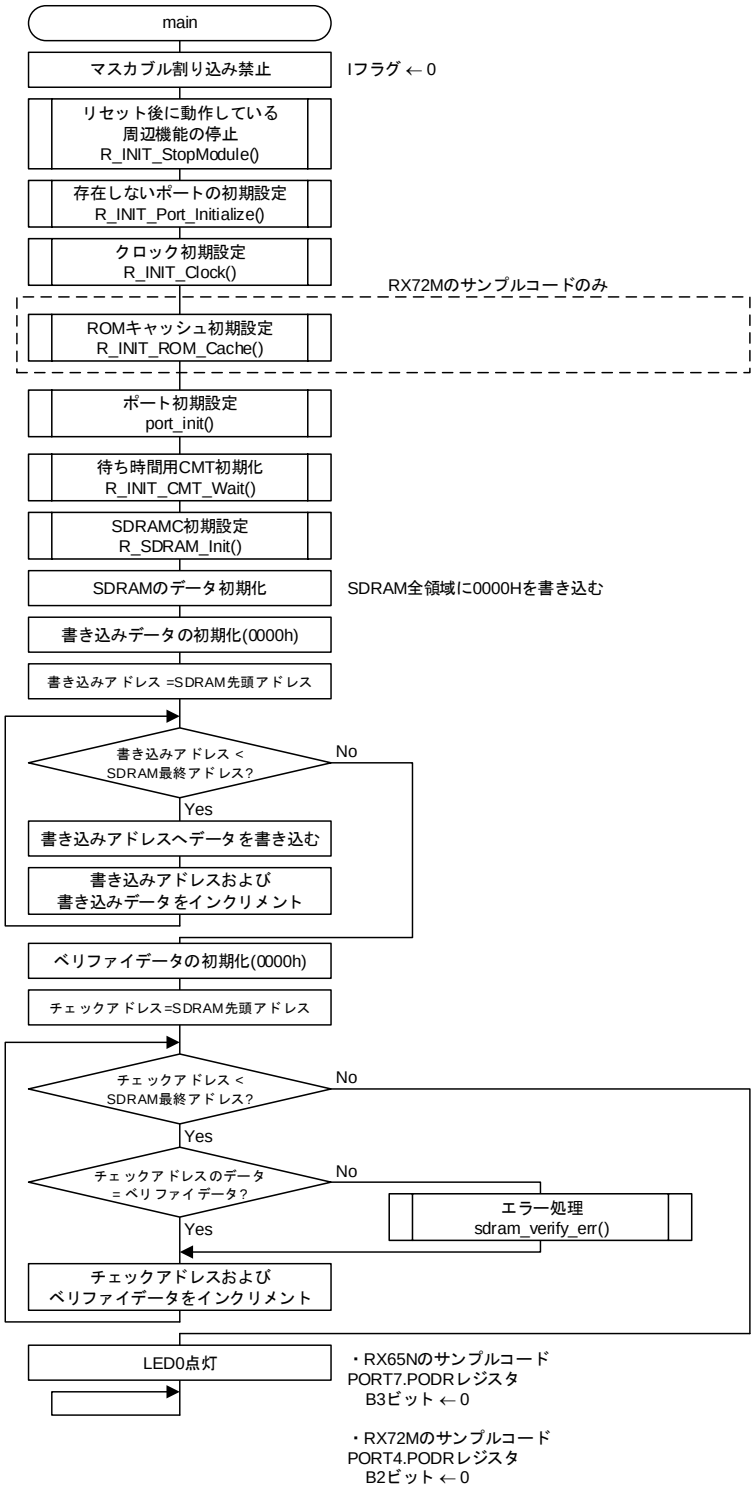


図 6.6　スマート・コンフィグレータを使用していないサンプルコードのメイン処理

6.8.1.2　SDRAMC 初期設定

図 6.7、図 6.8 に RX65N サンプルコードの SDRAMC 初期設定、図 6.9、図 6.10 に RX72M サンプルコードの SDRAMC 初期設定のフローチャートを示します。

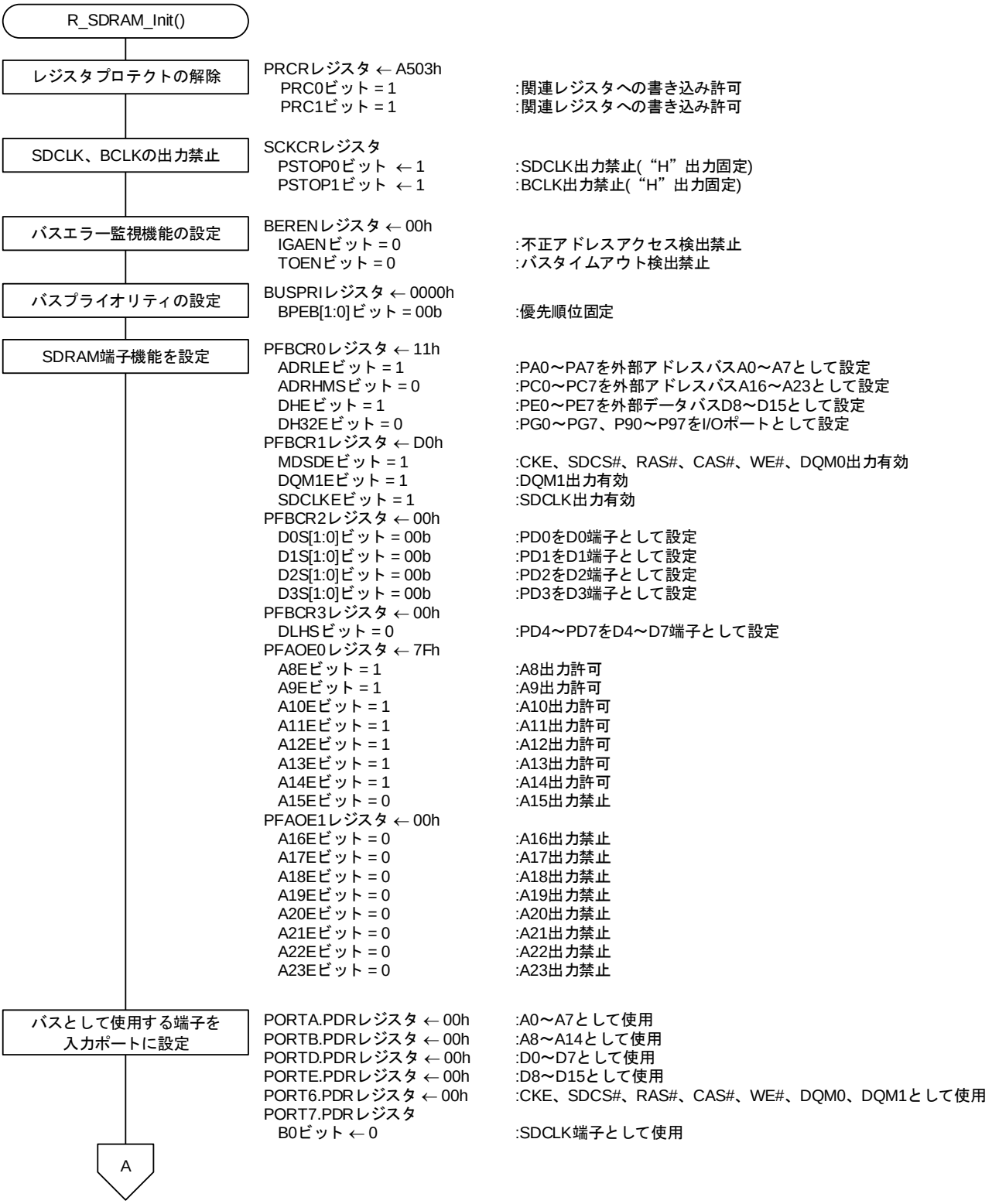


図 6.7　RX65N サンプルコードの SDRAMC 初期設定(1/2)

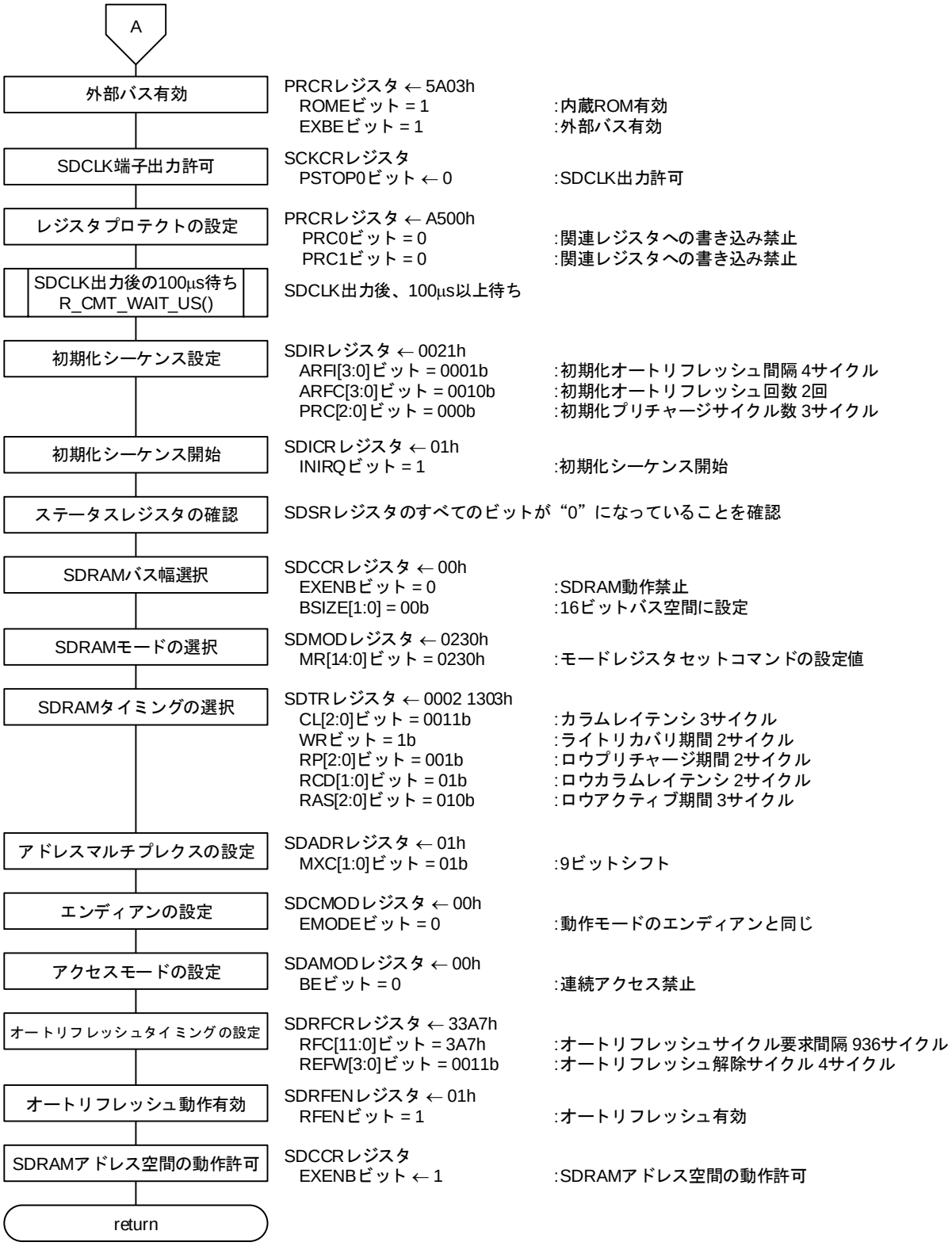


図 6.8　RX65N サンプルコードの SDRAMC 初期設定(2/2)

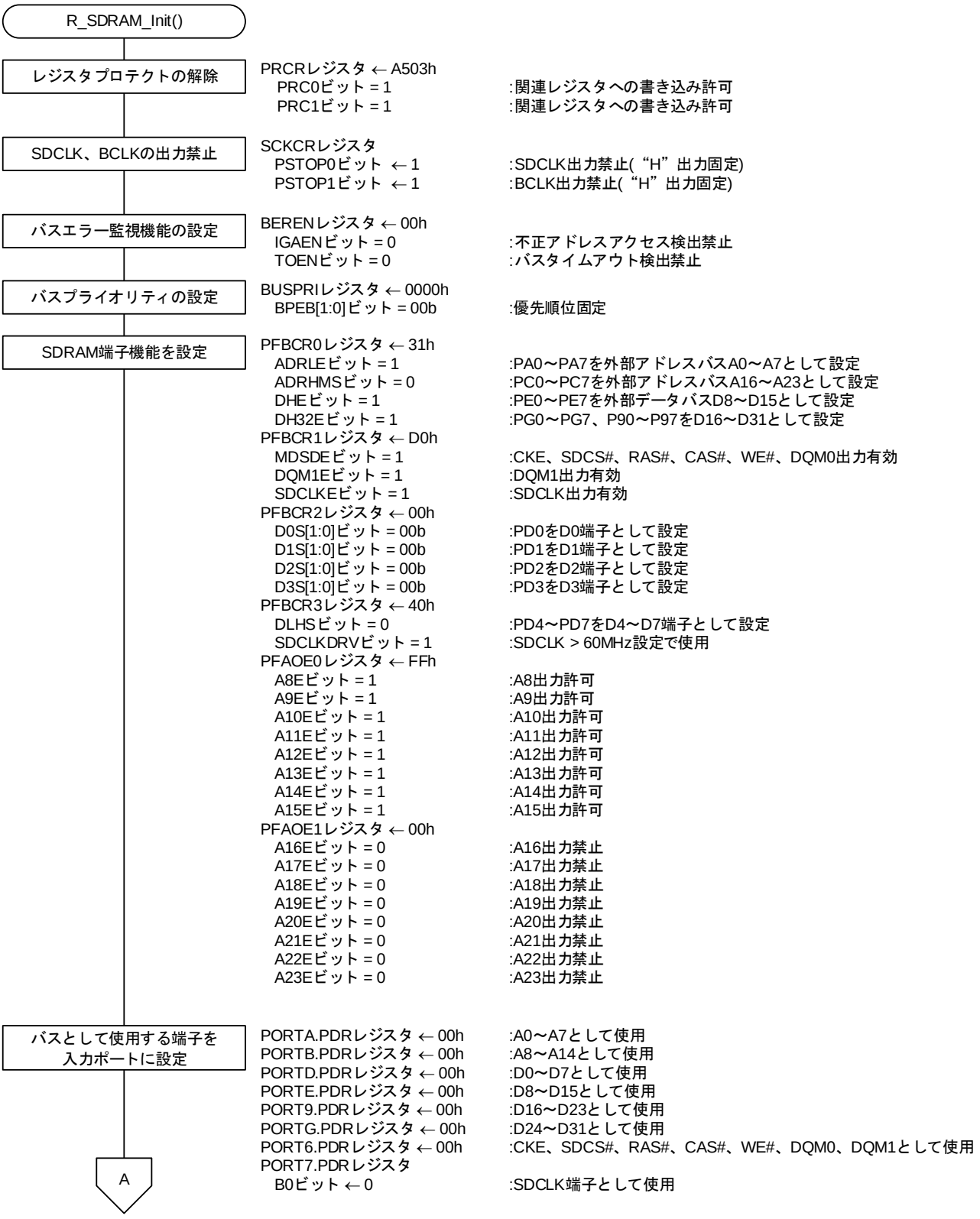


図 6.9　RX72M サンプルコードの SDRAM 初期設定(1/2)

図 6.10　RX72M サンプルコードの SDRAM 初期設定(2/2)

6.8.1.3　ポート初期化処理

図 6.11 に RX65N スマート・コンフィグレータを使用していないサンプルコードのポート初期化処理、図 6.12 に RX72M スマート・コンフィグレータを使用していないサンプルコードのポート初期化処理のフローチャートを示します。

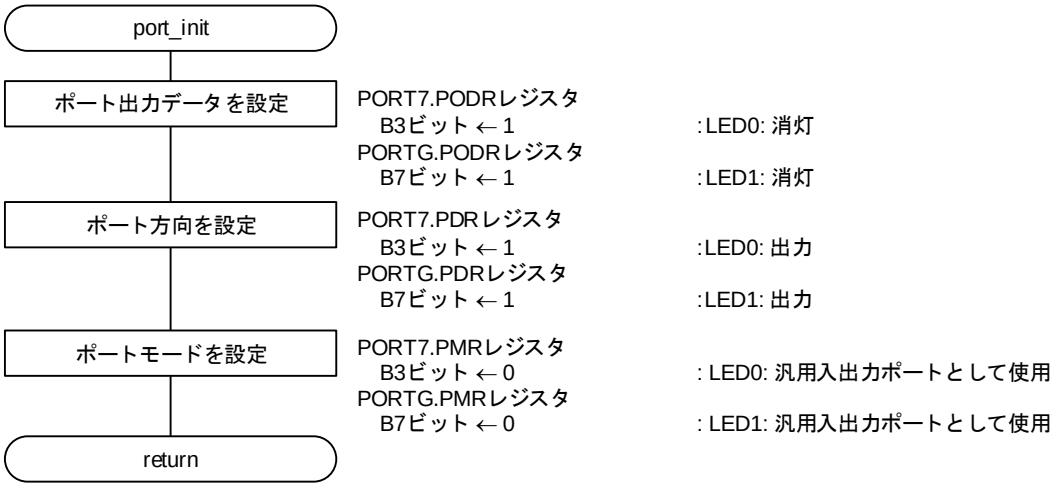


図 6.11　RX65N スマート・コンフィグレータを使用していないサンプルコードのポート初期化処理

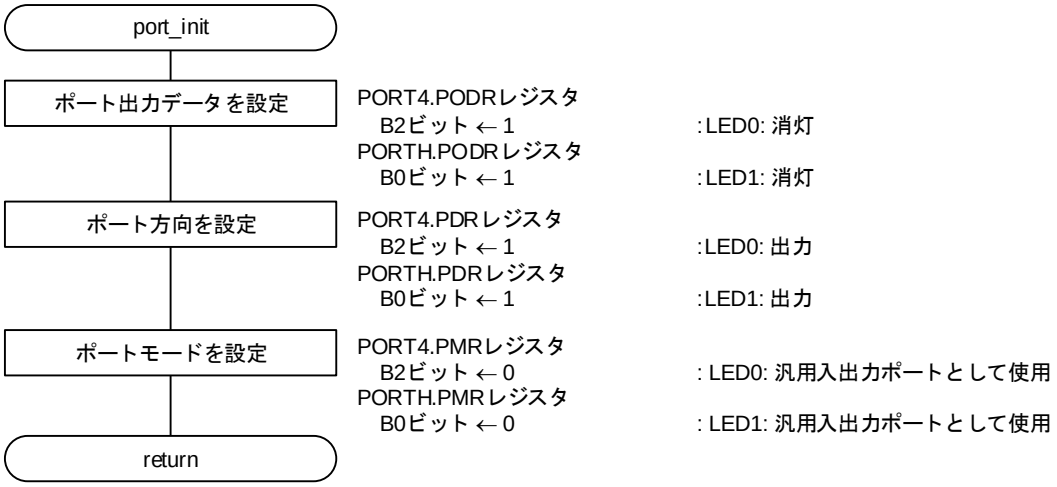


図 6.12 RX72M スマート・コンフィグレータを使用していないサンプルコードのポート初期化処理

6.8.1.4　待ち時間用タイマ初期設定

図 6.13 に待ち時間用タイマ初期設定のフローチャートを示します。

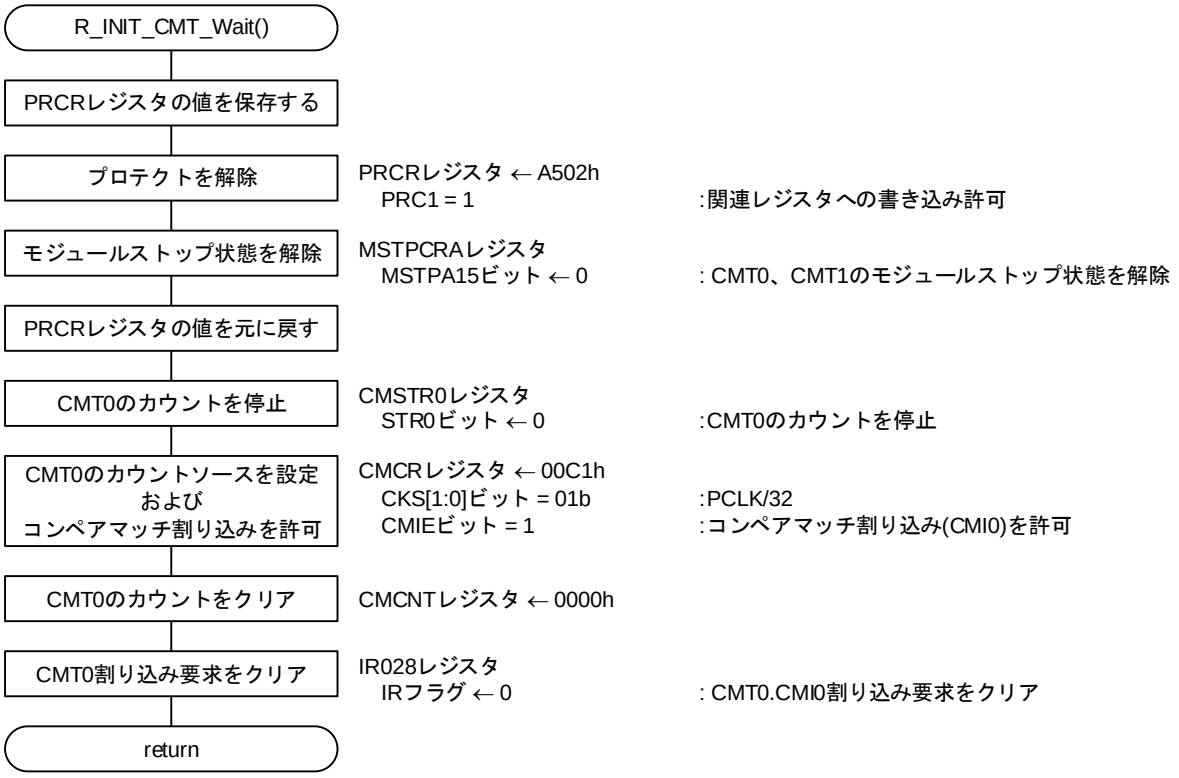


図 6.13　待ち時間用タイマ初期設定

6.8.1.5　CMT による待ち時間処理

図 6.14 に CMT による待ち時間処理のフローチャートを示します。

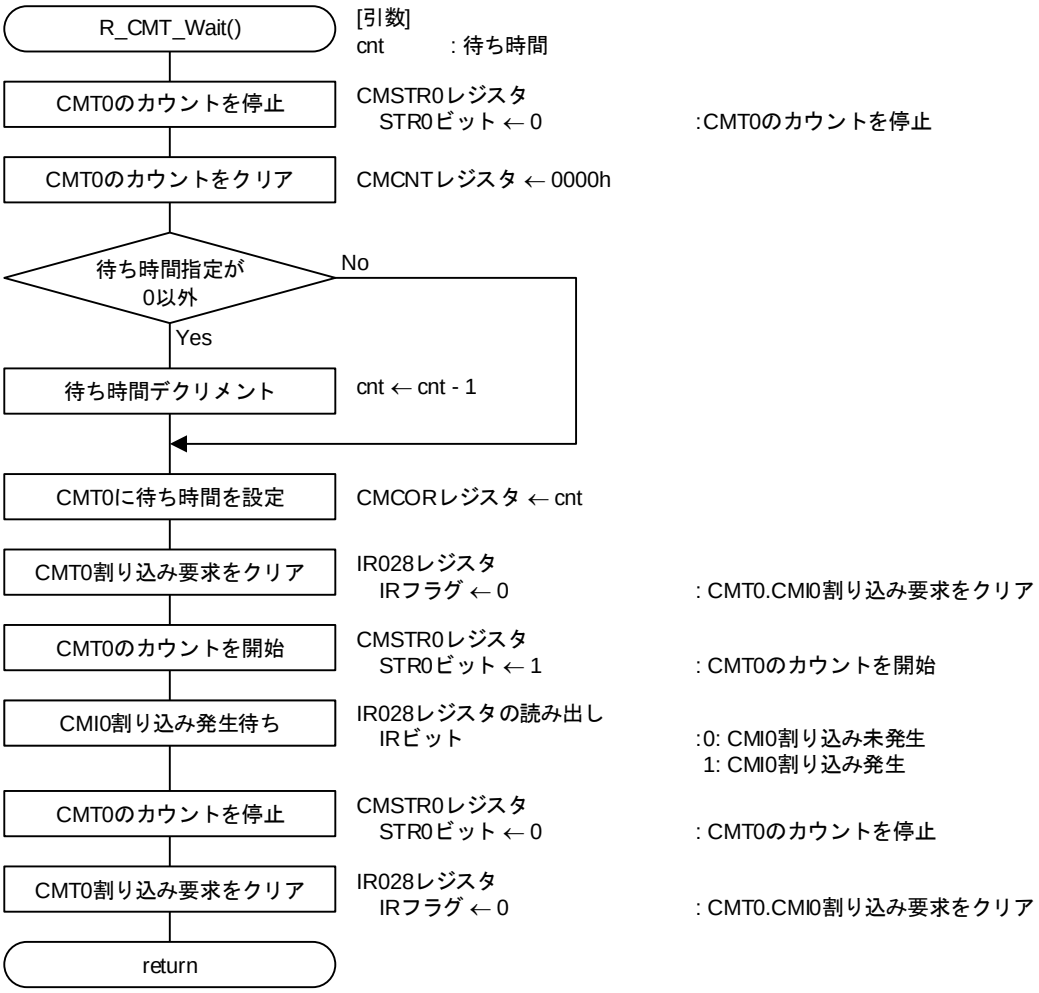


図 6.14　CMT による待ち時間処理

6.8.1.6　SDRAM ベリファイエラー処理

図 6.15 にスマート・コンフィグレータを使用していないサンプルコードの SDRAM ベリファイエラー処理のフローチャートを示します。

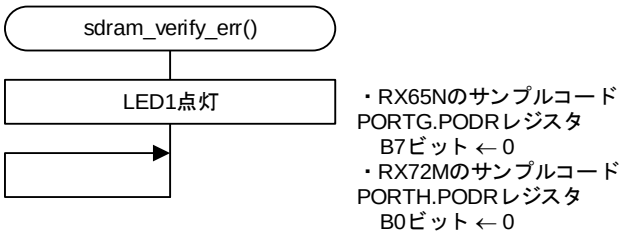


図 6.15 スマート・コンフィグレータを使用していないサンプルコードの SDRAM ベリファイエラー処理

6.8.2　スマート・コンフィグレータを使用しているサンプルコード

6.8.2.1　メイン処理

図 6.16 にスマート・コンフィグレータを使用しているサンプルコードのメイン処理フローチャートを、表 6.12 に R_GPIO_PinWrite()内データ設定を示します。スマート・コンフィグレータを使用しているサンプルコードでは、GPIO FIT の関数を使用しレジスタを書き換えています。

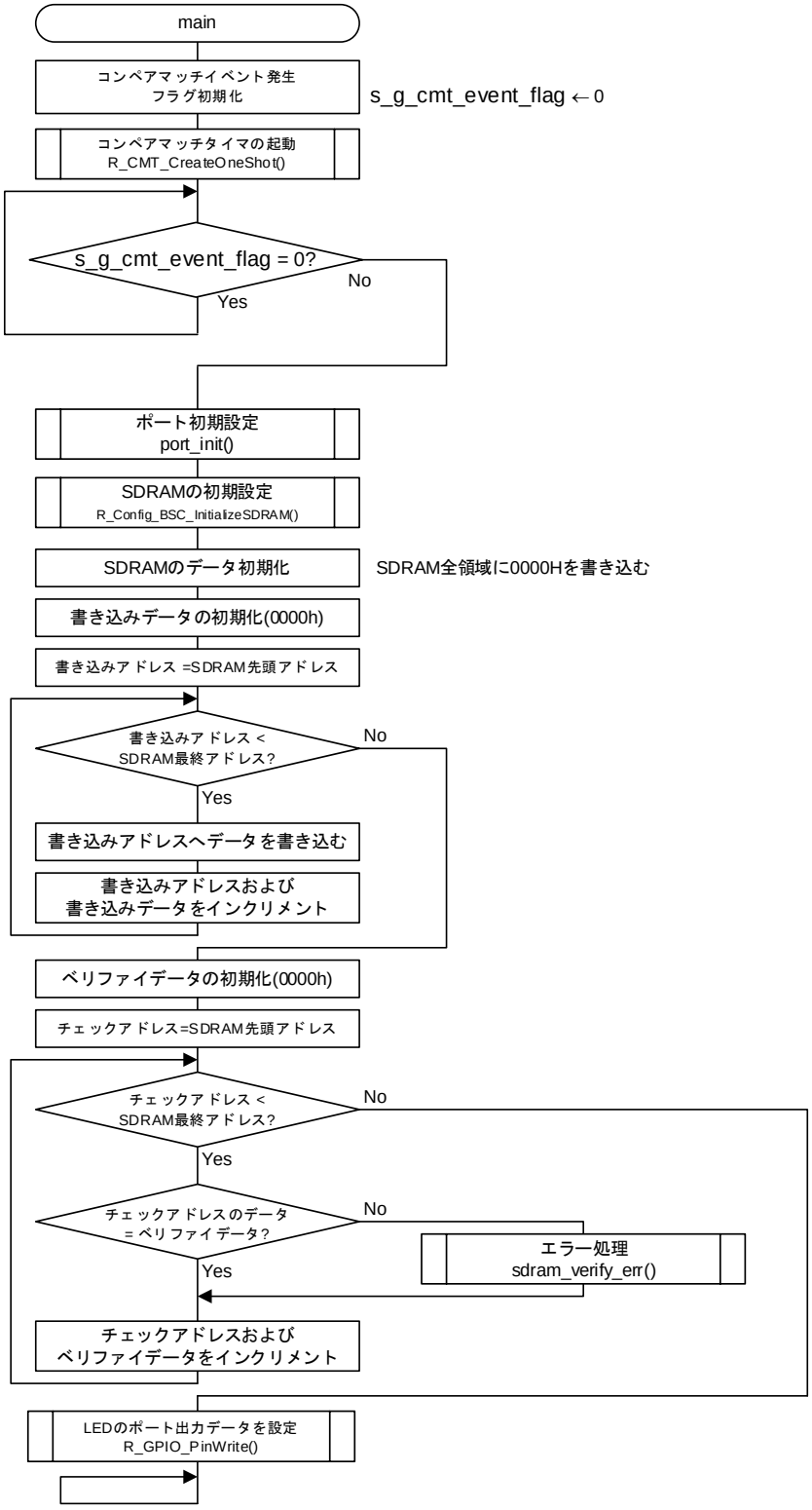


図 6.16　スマート・コンフィグレータを使用しているサンプルコードのメイン処理

表 6.12　R_GPIO_PinWrite()内データ設定

ボード名	設定ビット	設定値	動作
RSK- RX65N	PORT7.PODR レジスタ.B3 ビット	0	LED0 点灯
		1	LED0 消灯
RSK- RX72M	PORT4.PODR レジスタ.B2 ビット	0	LED0 点灯
		1	LED0 消灯
RSK- RX671	PORT1.PODR レジスタ.B7 ビット	0	LED0 点灯
		1	LED0 消灯
RSK- RX72N	PORT7.PODR レジスタ.B1 ビット	0	LED0 点灯
		1	LED0 消灯
EK-RX671	PORTH.PODR レジスタ.B6 ビット	0	LED1 点灯
		1	LED1 消灯

6.8.2.2　ポート初期化処理

図 6.17 にスマート・コンフィグレータを使用しているサンプルコードのポート初期化処理のフローチャートを、表 6.13 に R_GPIO_PinDirection()内データ設定を、表 6.14 に R_GPIO_PinControl()内データ設定を示します。スマート・コンフィグレータを使用しているサンプルコードでは、GPIO FIT の関数を使用しレジスタを書き換えています。

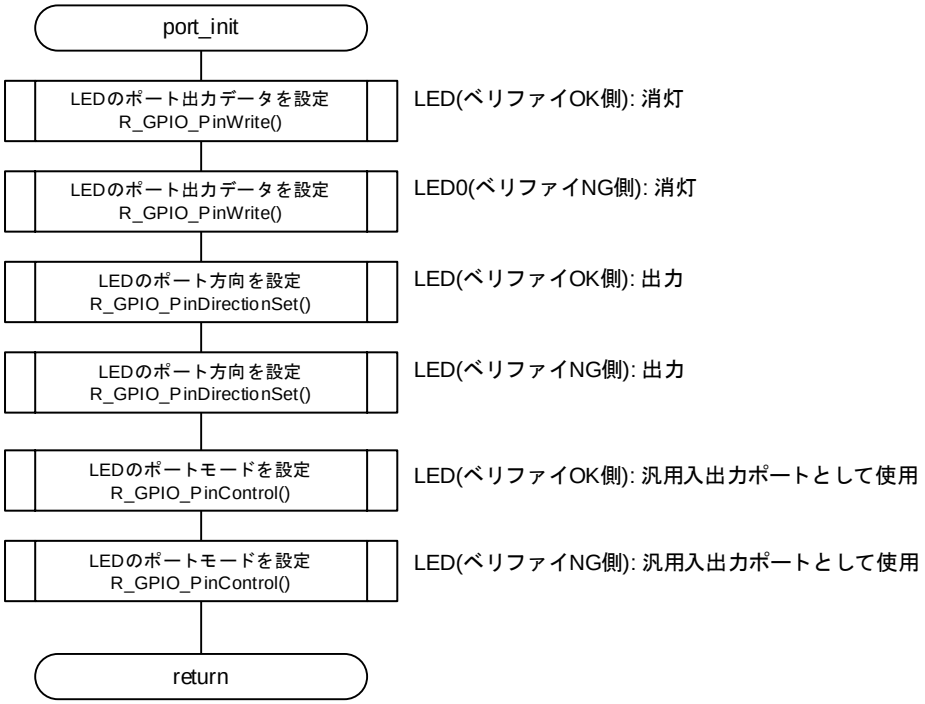


図 6.17 スマート・コンフィグレータを使用しているサンプルコードのポート初期化処理

表 6.13　R_GPIO_PinDirection()内データ設定

ボード名	設定ビット	設定値	動作
RSK- RX65N	PORT7.PDR レジスタ.B3 ビット	1	出力ポート設定
	PORTG.PDR レジスタ.B7 ビット	1	出力ポート設定
RSK- RX72M	PORT4.PDR レジスタ.B2 ビット	1	出力ポート設定
	PORTH.PDR レジスタ.B0 ビット	1	出力ポート設定
RSK- RX671	PORT1.PDR レジスタ.B7 ビット	1	出力ポート設定
	PORTF.PDR レジスタ.B5 ビット	1	出力ポート設定
RSK- RX72N	PORT7.PDR レジスタ.B1 ビット	1	出力ポート設定
	PORT8.PDR レジスタ.B2 ビット	1	出力ポート設定
EK-RX671	PORTH.PDR レジスタ.B6 ビット	1	出力ポート設定
	PORT7.PDR レジスタ.B3 ビット	1	出力ポート設定

表 6.14　R_GPIO_PinControl()内データ設定

ボード名	設定ビット	設定値	動作
RSK- RX65N	PORT7.PMR レジスタ.B3 ビット	0	汎用入出力ポート設定
	PORTG.PMR レジスタ.B7 ビット	0	汎用入出力ポート設定
RSK- RX72M	PORT4.PMR レジスタ.B2 ビット	0	汎用入出力ポート設定
	PORTH.PMR レジスタ.B0 ビット	0	汎用入出力ポート設定
RSK- RX671	PORT1.PMR レジスタ.B7 ビット	0	汎用入出力ポート設定
	PORTF.PMR レジスタ.B5 ビット	0	汎用入出力ポート設定
RSK- RX72N	PORT7.PMR レジスタ.B1 ビット	0	汎用入出力ポート設定
	PORT8.PMR レジスタ.B2 ビット	0	汎用入出力ポート設定
EK-RX671	PORTH.PMR レジスタ.B6 ビット	0	汎用入出力ポート設定
	PORT7.PMR レジスタ.B3 ビット	0	汎用入出力ポート設定

6.8.2.3　SDRAM ベリファイエラー処理

図 6.18 にスマート・コンフィグレータを使用しているサンプルコードの SDRAM ベリファイエラー処理のフローチャートを示します。スマート・コンフィグレータを使用しているサンプルコードでは、GPIO FIT の関数を使用しレジスタを書き換えています。

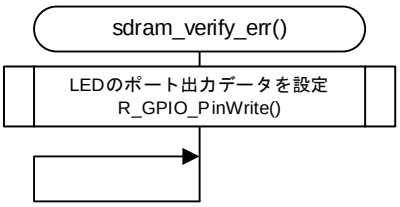


図 6.18 スマート・コンフィグレータを使用しているサンプルコードの SDRAM ベリファイエラー処理

6.8.2.4　コンペアマッチイベントコールバック処理

図 6.19 にコンペアマッチイベント発生コールバック処理のフローチャートを示します。

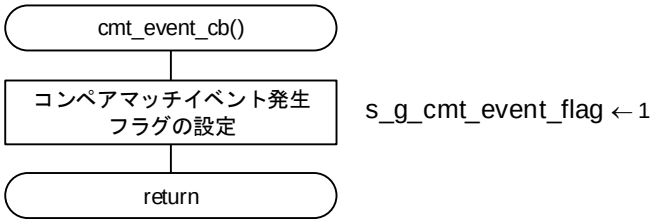


図 6.19　コンペアマッチイベント発生コールバック処理

.

7.　SDRAM の仕様に対する対象デバイスにおけるレジスタ設定の考え方

本章では、MT48LC4M32B2P-6A を例に SDRAM の仕様に対する対象デバイスにおける、レジスタ設定の考え方を説明します。

なお、SDRAMC で使用する端子の設定については、使用するデバイスのユーザーズマニュアル　ハードウェア編、IO ポート章内の「出力許可設定一覧」、またはマルチファンクションピン章内の「外部バスインタフェース設定方法」を確認してください。

7.1　BCLK(SDCLK)の設定

SDRAM の CLK 端子に供給するクロックは、SDRAM データシートの AC 特性に記載されている時間以上である必要があります。

MT48LC4M32B2P-6A、CL=3 の場合、1 サイクルが 7ns 以上(約 142MHz 以内)のクロックを供給する必要があります。また、使用するデバイスごとに SDCLK 端子から出力できる最大周波数が異なるため、各ユーザーズマニュアルを確認してください。

例として使用するデバイスが RX72M であり、クロックソースが 240MHz の場合、SCKCR レジスタの BCK[3:0]で 3 分周以上の設定かつ「ICLK≧BCLK」となる設定であれば仕様を満たすことができます。

7.2　SDC 制御レジスタ(SDCCR)

SDRAM 空間の動作許可(EXENB ビット)及び SDRAM バス幅選択(BSIZE ビット)ができます。

MT48LC4M32B2P-6A の場合 32 ビット幅でアクセスできるため、BSIZE[1:0]を”01b”(32 ビットバス空間)に設定することで仕様を満たすことができます。SDRAM 空間の動作許可は SDRAMC 関連のレジスタ設定が完了してから EXENB を”1”(動作許可)にしてください。

7.3　SDC モードレジスタ(SDCMOD)

SDRAM アドレス空間のエンディアン設定(EMODE ビット)ができます。

MT48LC4M32B2P-6A の場合で、デバイスの動作モードがリトルエンディアンの場合は、SDRAM アドレス空間のエンディアンは動作モードのエンディアンと同じになります。この場合は EMODE を”0”(SDRAM アドレス空間のエンディアンは動作モードのエンディアンと同じ)に設定することで動作させることができます。

7.4　SDC アクセスモードレジスタ(SDAMOD)

SDRAM 空間の連続アクセス許可(BE ビット)ができます。

MT48LC4M32B2P-6A の場合、連続アクセスが可能であるため、EXDMAC を使用してアクセスする設定を行った後で、BE を”1”(連続アクセス許可)に設定することで動作させることができます。

7.5　SDRAM リフレッシュ制御レジスタ(SDRFCR)

オートリフレッシュ要求間隔設定(RFC ビット)及びオートリフレッシュサイクル/セルフリフレッシュ解除サイクル数設定(REFW ビット)ができます。

オートリフレッシュ要求間隔は以下の式から求められます。

・リフレッシュサイクル / ロウアドレス数

MT48LC4M32B2P-6A の場合、リフレッシュサイクルが 64ms、ロウアドレス数が”4096”であるためオートリフレッシュ要求間隔は 15.625μs となります。オートリフレッシュ要求はこの時間以内に行う必要があるため、オートリフレッシュ要求間隔を RFC ビットで設定を行う必要があります。例として SDCLK が 60MHz の場合、RFC[11:0]は”3A7h”(936 サイクル)以下に設定することで 15.6μs 以下となり、仕様を満たすことができます。

オートリフレッシュサイクル/セルフリフレッシュ解除サイクル数設定は tRFC 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRFC は 60ns です。例として SDCLK が 60MHz の場合、REFW[3:0]は”0011b”(4 サイクル)以上に設定することで 67ns 以上となり、仕様を満たすことができます。

7.6　SDRAM 初期化レジスタ(SDIR)

初期化オートリフレッシュ間隔(ARFI ビット)、初期化オートリフレッシュ回数(ARFC ビット)、初期化プリチャージサイクル数(PRC ビット)の設定ができます。

初期化オートリフレッシュ間隔は tRFC 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRFC は 60ns です。例として SDCLK が 60MHz の場合、ARFI[3:0]は”0001b”(4 サイクル)以上に設定することで 67ns 以上となり、仕様を満たすことができます。

初期化オートリフレッシュ回数は、データシートに記載されている回数実施してください。MT48LC4M32B2P-6A の場合は 2 回以上であるため、ARFC[3:0]には”0010b”(2 回) 以上に設定することで 2 回以上となり、仕様を満たすことができます。

初期化プリチャージサイクル数は tRP 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRP は 18ns です。例として SDCLK が 60MHz の場合、PRC[3:0]は”0000b”(3 サイクル)以上に設定することで 50ns 以上となり、仕様を満たすことができます。

7.7　SDRAM アドレスレジスタ(SDADR)

アドレスマルチプレクスのシフト量選択(MXC ビット)ができます。アドレスマルチプレクスのシフト量はカラムアドレッシングの幅に合わせてください。MT48LC4M32B2P-6A の場合 8 ビット幅のため、MXC [1:0]を”00b”(8 ビットシフト)に設定することで仕様を満たすことができます。

7.8　SDRAM タイミングレジスタ(SDTR)

このレジスタでは、以下の設定を行うことができます。

- SDRAMC カラムレイテンシ設定(CL ビット)
- ライトリカバリ期間設定(WR ビット)
- ロウプリチャージ期間設定(RP ビット)
- ロウカラムレイテンシ設定(RCD ビット)
- ロウアクティブ期間設定(RAS ビット)

SDRAMC カラムレイテンシ設定は SDRAM に設定したカラムレイテンシと同じになるように設定してください。例として、SDRAM のカラムレイテンシ設定が”3”の場合、CL[2:0]は”011b”(3 サイクル)に設定することで仕様を満たすことができます。

ライトリカバリ期間設定は tWR 以上になるように設定してください。MT48LC4M32B2P-6A の場合で SDCLK が 60MHz であるとき、tWR は 23.67ns です。例として SDCLK が 60MHz の場合、WR ビットは”1”(2 サイクル)に設定することで 33ns となり、仕様を満たすことができます。

ロウプリチャージ期間設定は tRP 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRP は 18ns です。例として SDCLK が 60MHz の場合、RP[2:0]は”001b”(2 サイクル)以上に設定することで 33ns 以上となり、仕様を満たすことができます。

ロウカラムレイテンシ設定は tRCD 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRCD は 18ns です。例として SDCLK が 60MHz の場合、RCD[1:0]は”01b”(2 サイクル)以上に設定することで 33ns 以上となり、仕様を満たすことができます。

ロウアクティブ期間設定は tRAS 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRAS は 42ns です。例として SDCLK が 60MHz の場合、[2:0]は”010b”(3 サイクル)以上に設定することで 50ns 以上となり、仕様を満たすことができます。

7.9　SDRAM モードレジスタ(SDMOD)

SDRAM のモードレジスタへ設定を書き込むことができます。MT48LC4M32B2P-6A の場合でカラムレイテンシを”3”、バースト長を”1”に設定したい場合、SDRAM モードレジスタに”0x0230”を設定することで SDRAM に期待通りの設定を行うことができます。

8. 他 RX ファミリにスマート・コンフィグレータを使用していないサンプルコードをポーティングする方法

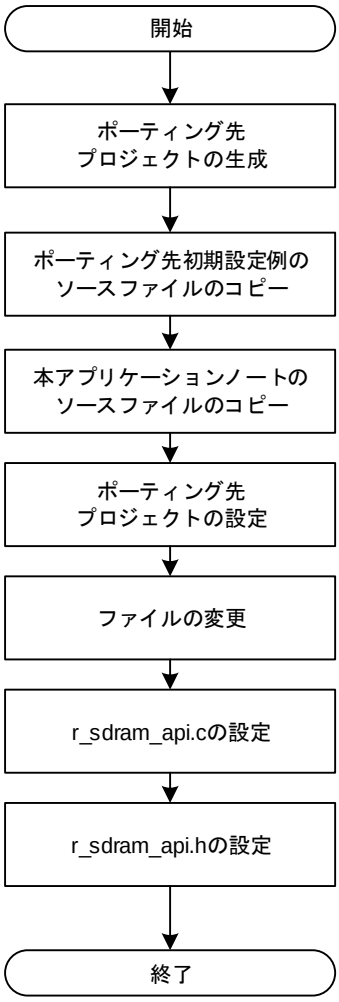
本アプリケーションノートに同梱されているスマート・コンフィグレータを使用していないサンプルコードは SDRAMC を搭載する他 RX ファミリにポーティングできます。本章では RX65N のスマート・コンフィグレータを使用していないサンプルコードを RX72M(Renesas Starter Kit+ for RX72M)にポーティングする例を示します。

8.1 ポーティングする前に

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 SDRAMC の仕様
- ポーティング元とポーティング先の CMT の仕様

8.2 ポーティング手順フロー



8.3　ポーティング手順

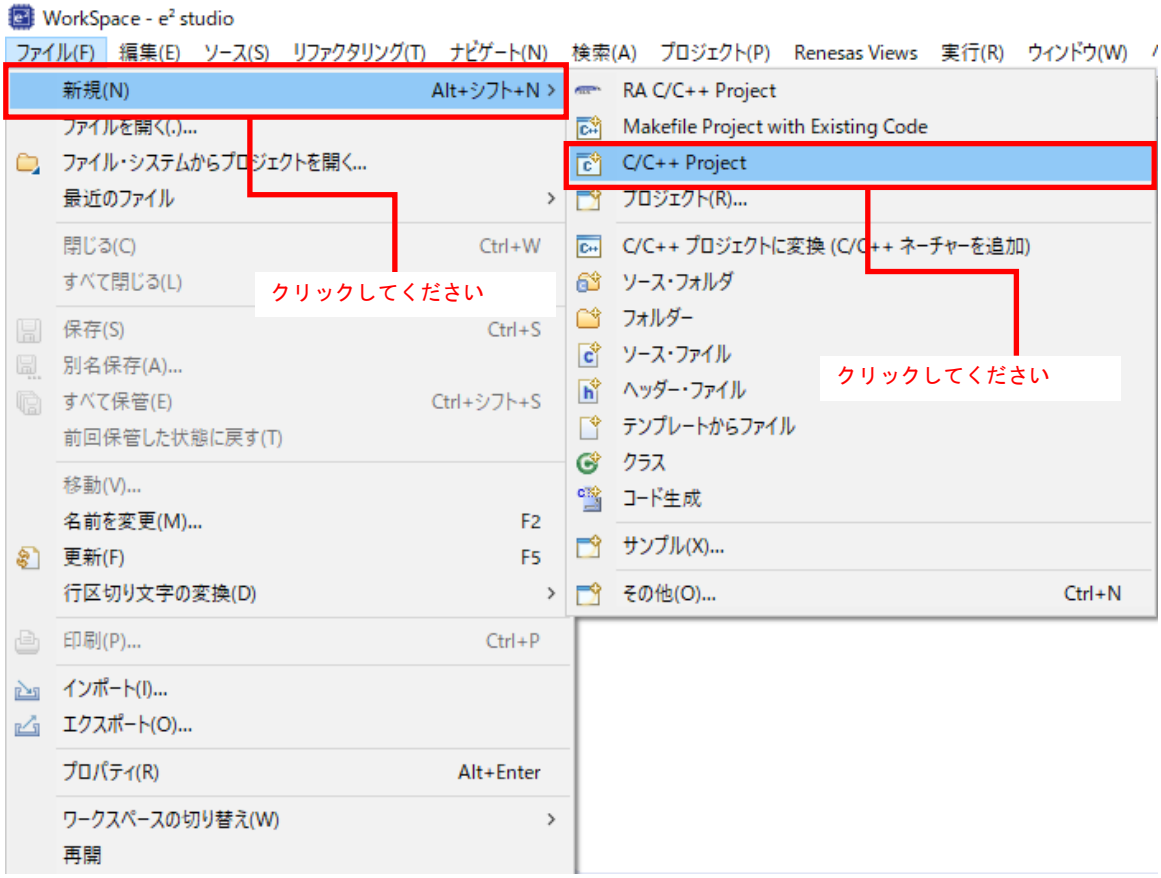
8.3.1　ポーティング先プロジェクトの生成

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。

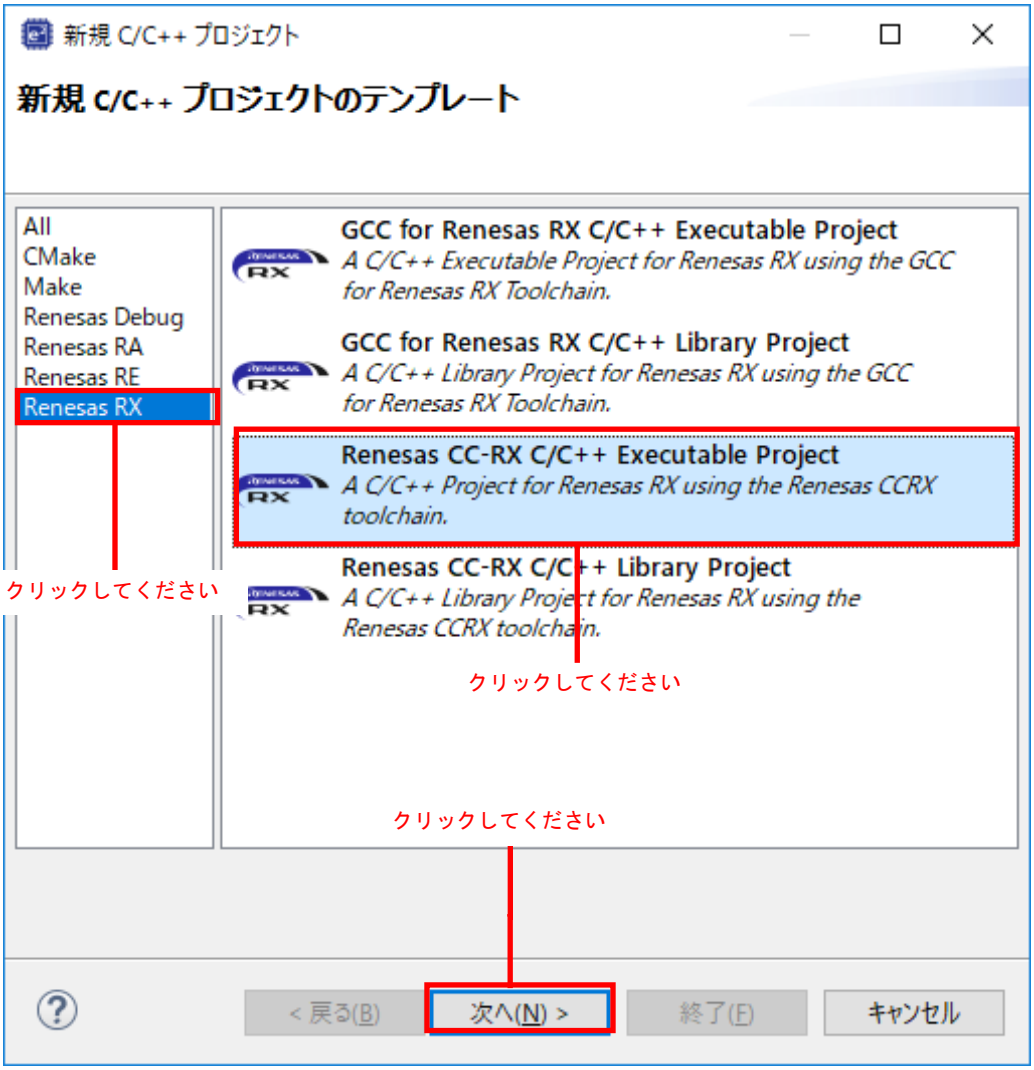
1-2) [新規(N)]の[C/C++ Project]をクリックして、New C/C++ Project ウィザードを起動します。



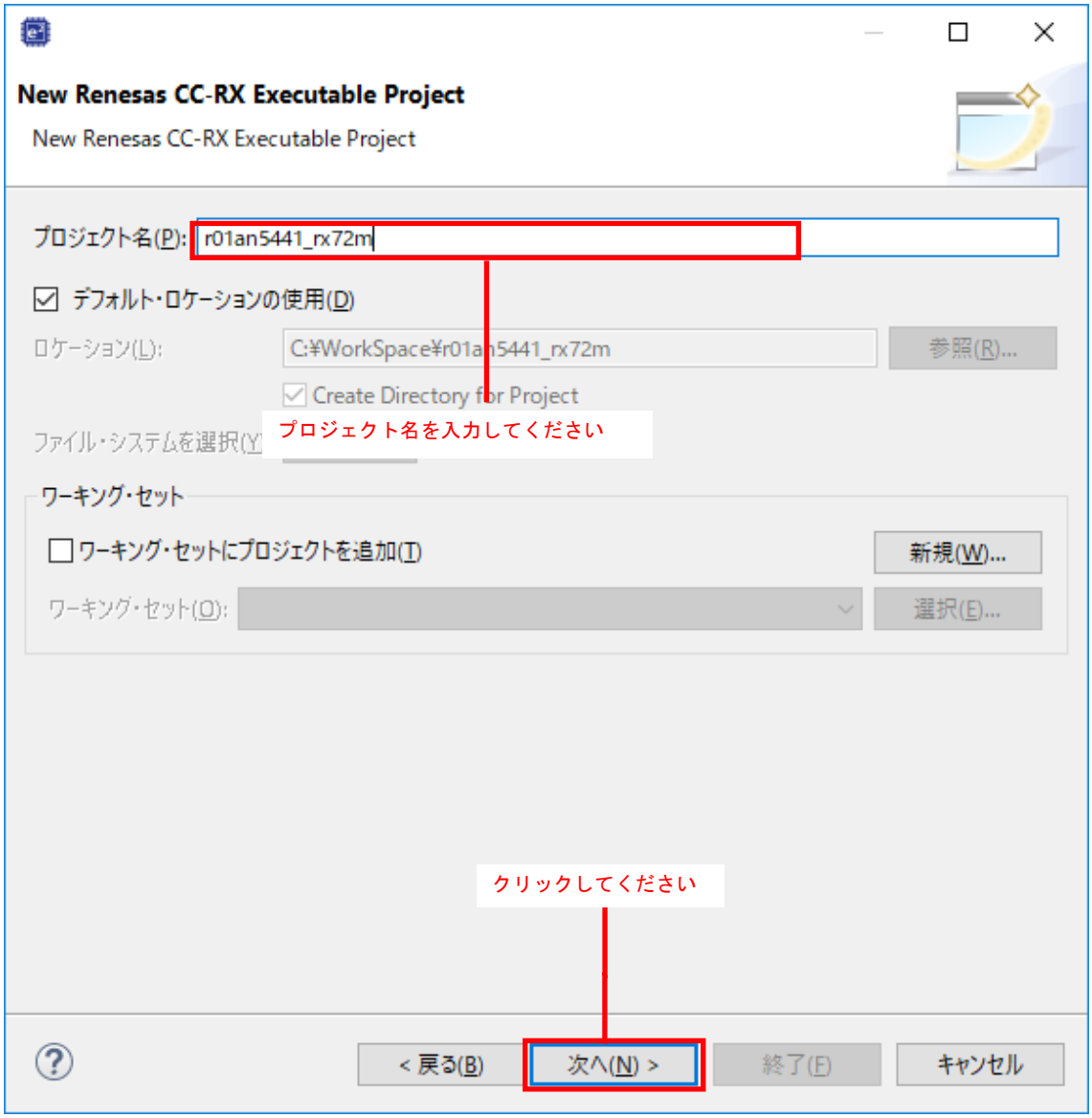
1-3) [Renesas RX]をクリックしてください。

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。



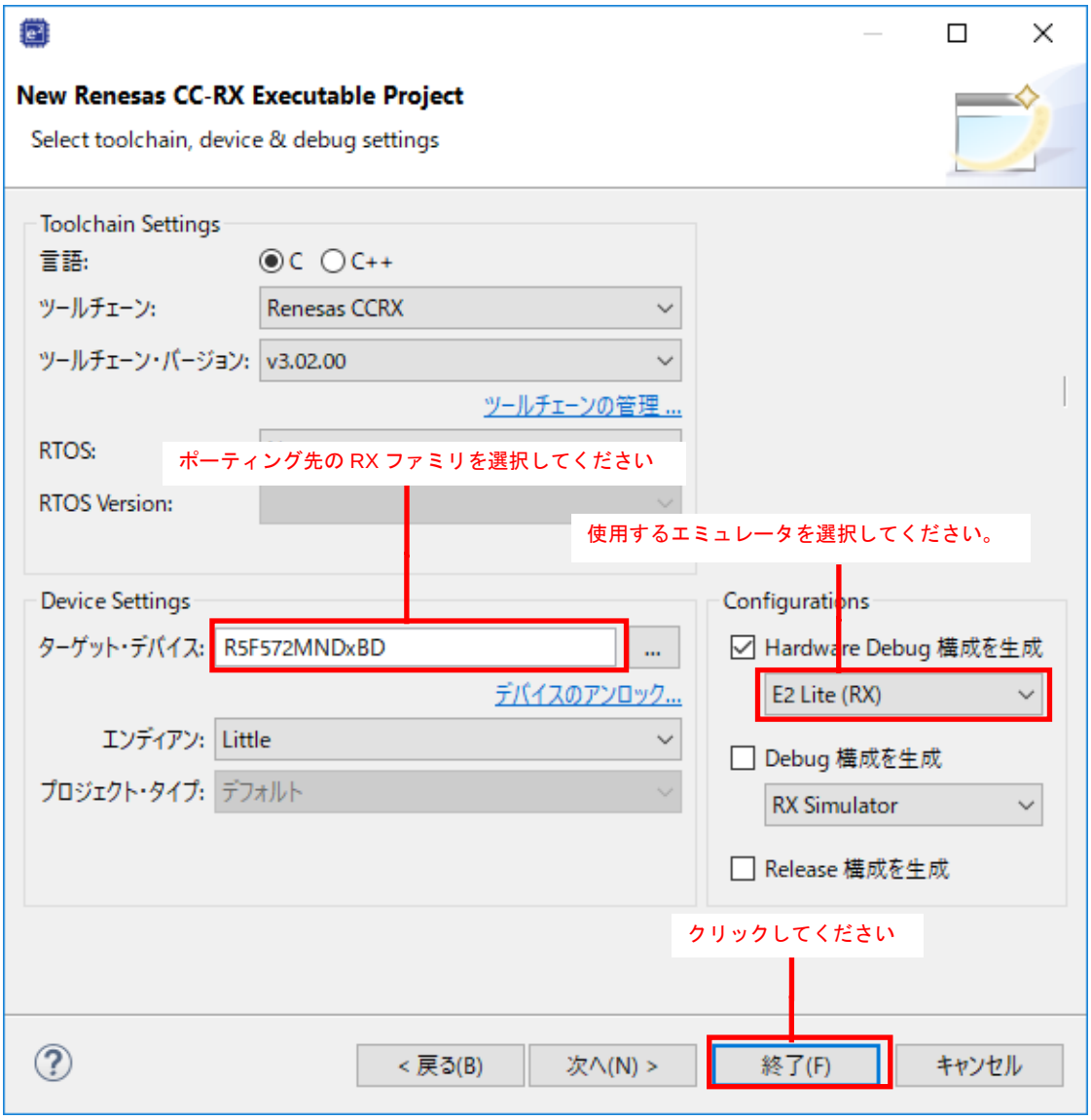
1-6) 任意のプロジェクト名を入力してください。
1-7) [次へ(N)>]をクリックしてください。



1-8) [ターゲットデバイス:]を[R5F572MNDxBD]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) [終了(F)]をクリックしてください。



1-11) 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。

8.3.2　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。

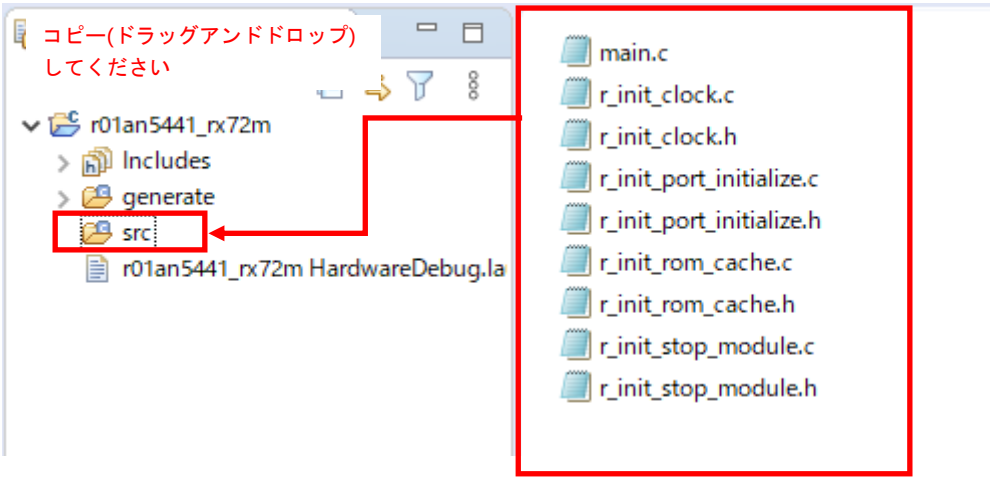
1)　初期設定例アプリケーションノートのダウンロード

1-1) [RX72M グループ 初期設定例(R01AN4530)]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、[r01an4530_rx72m] -> [r01an4530_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。

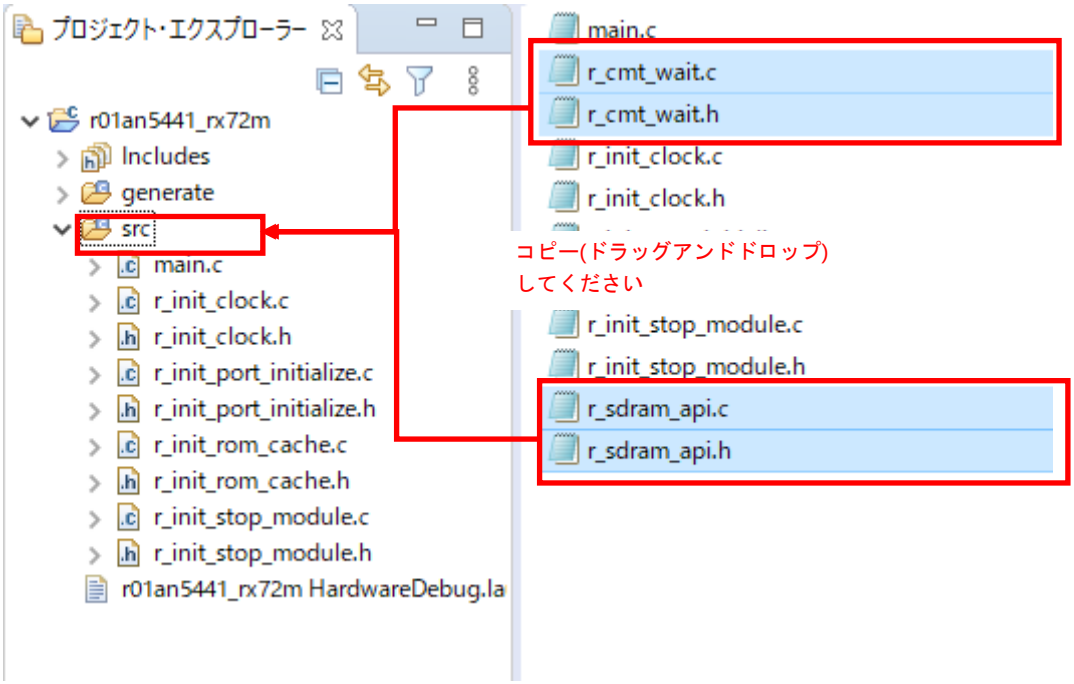


8.3.3　本アプリケーションノートのソースファイルのコピー

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[r01an5441_rx65n_sdram] -> [r01an5441_src]にある[r_cmt_wait.c]、[r_cmt_wait.h]、[r_sdram_api.c]、[r_sdram_api.h]をプロジェクトにコピーします。

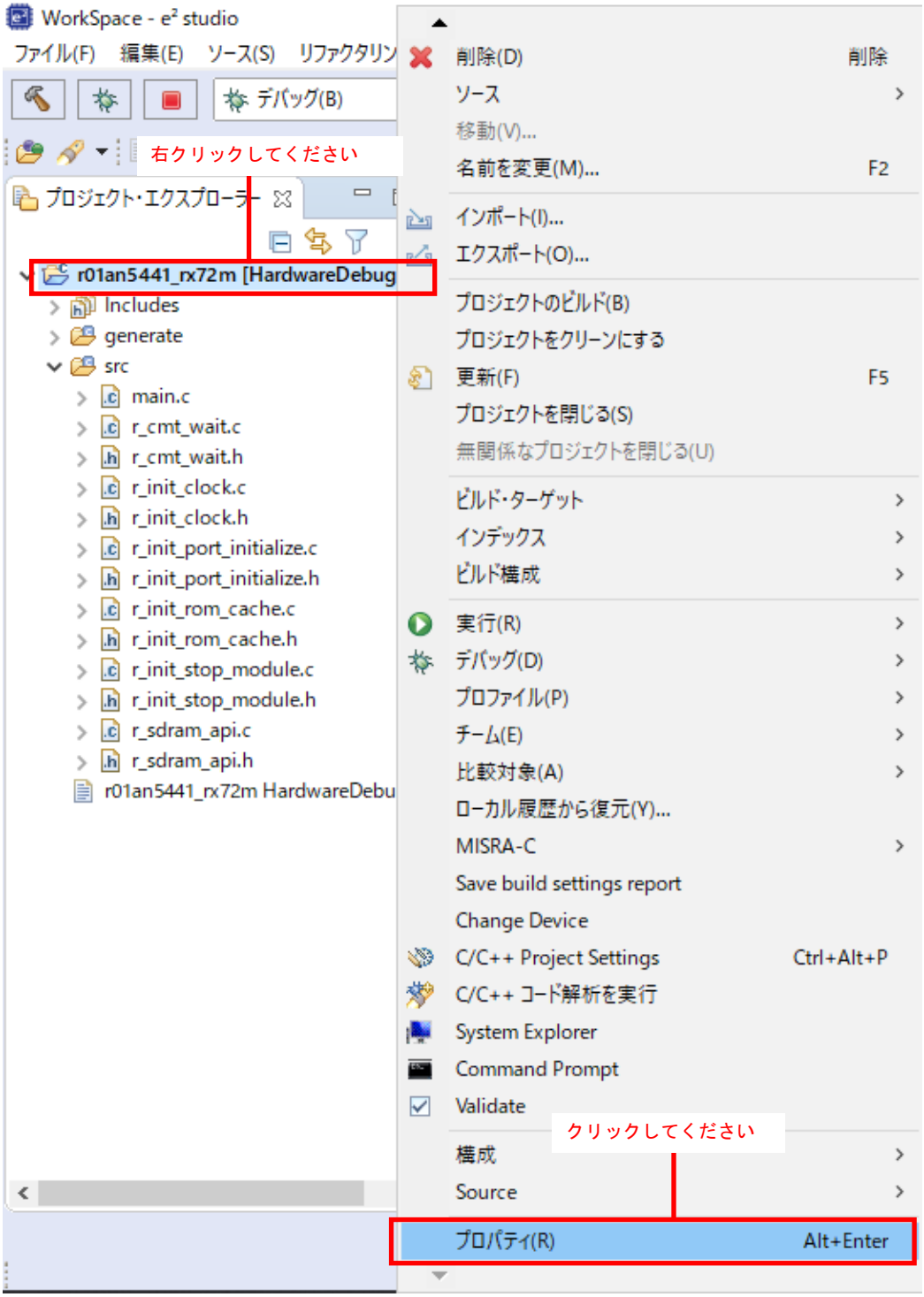


8.3.4　ポーティング先プロジェクトの設定

生成したプロジェクトのビルド設定を変更します。

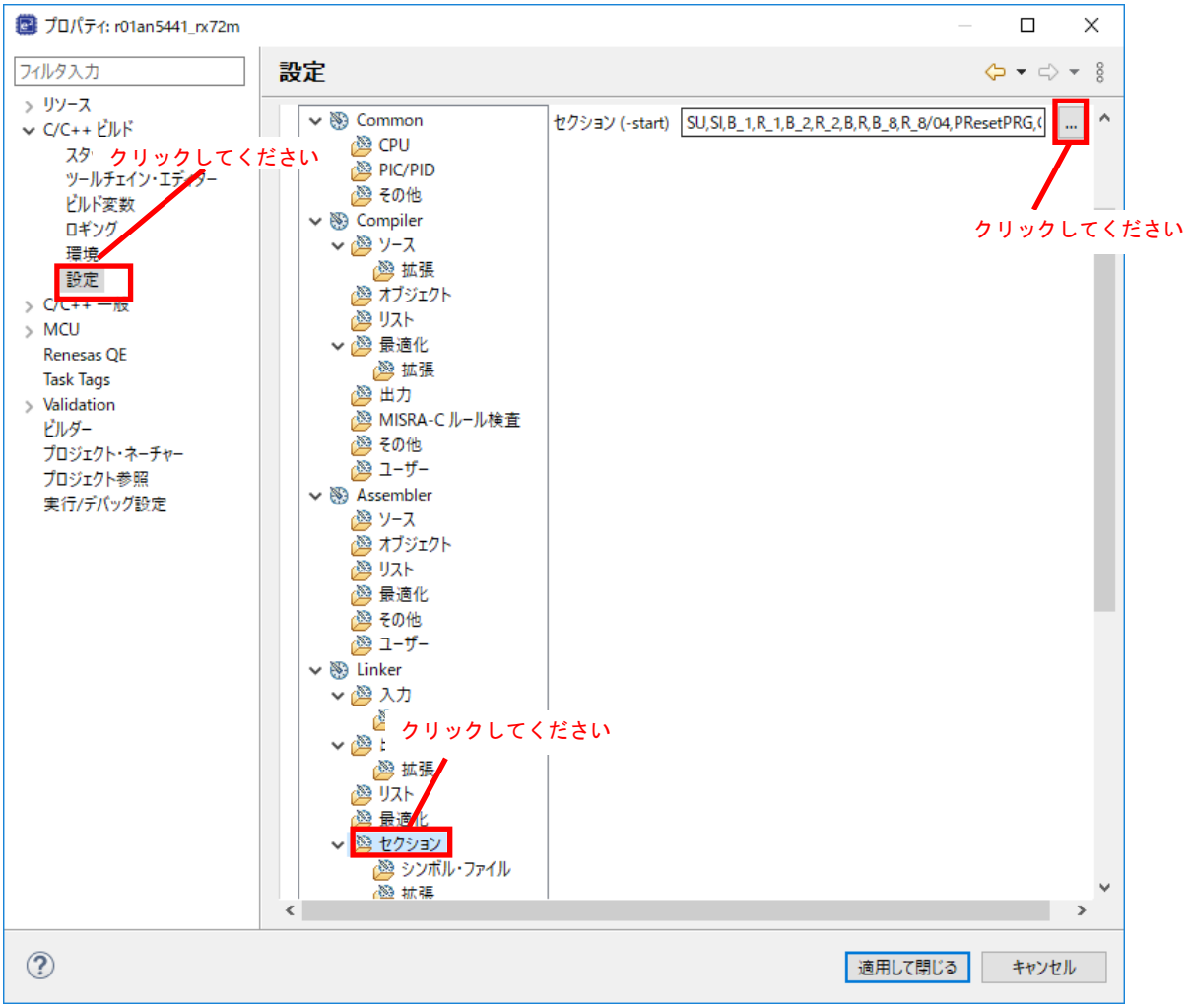
1)　RAM の最終アドレスのセクションを追加

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。

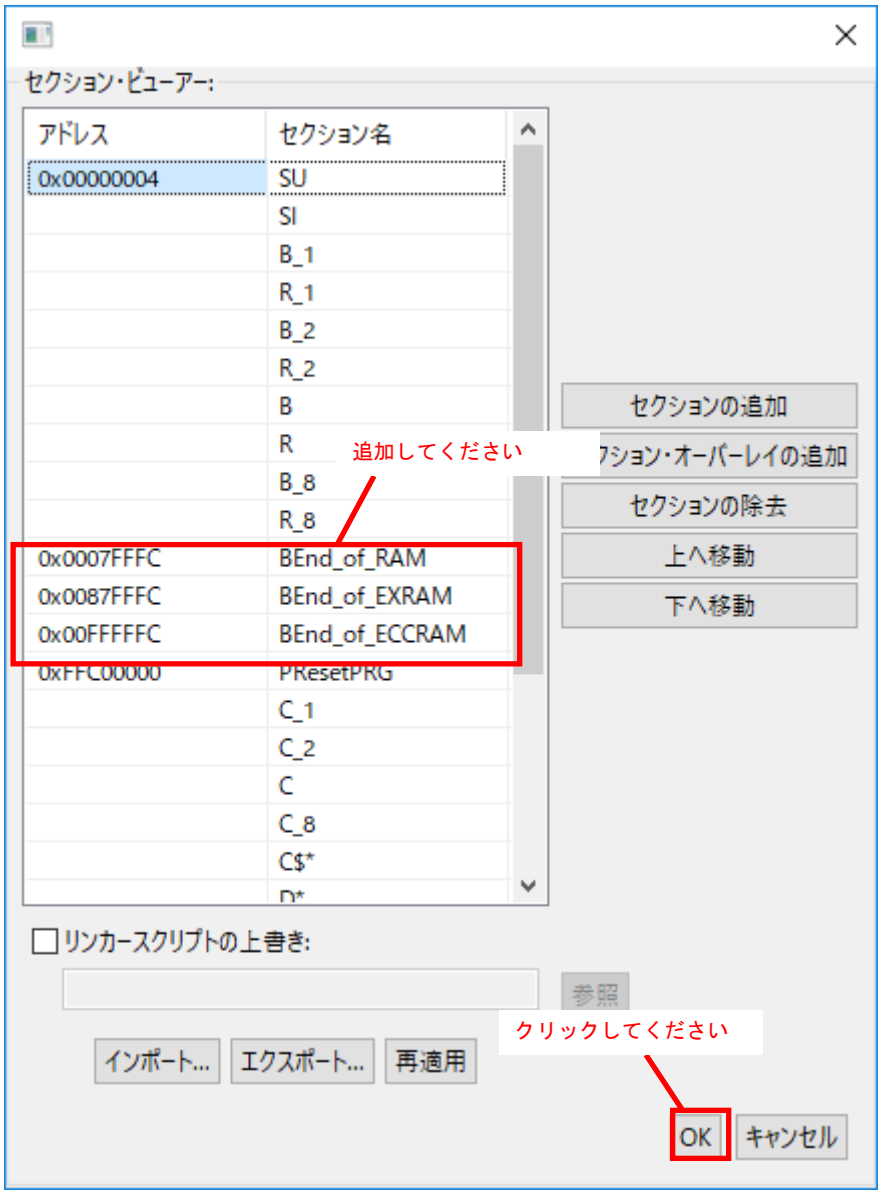


1-2) [C/C++ ビルド] -> [設定]をクリックしてください。
1-3) [ツール設定] -> [Linker] -> [セクション]をクリックしてください。

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]セクション及び[End_of_EXRAM]セクション、[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) [適用して閉じる]をクリックしてください。

8.3.5　ファイルの変更

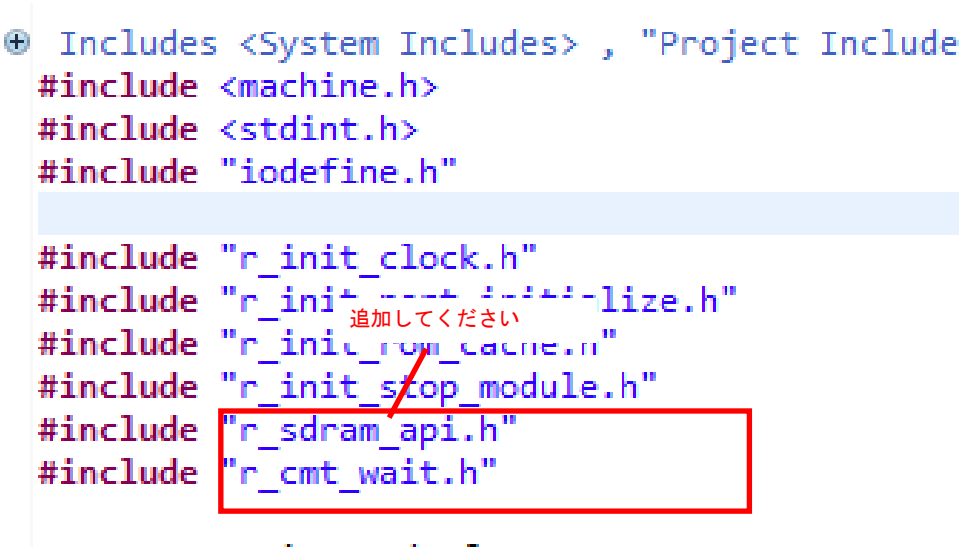
本アプリケーションのサンプルコードを動作させるために、コピーした各ソースファイルを変更します。

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)　[main.c]の変更

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



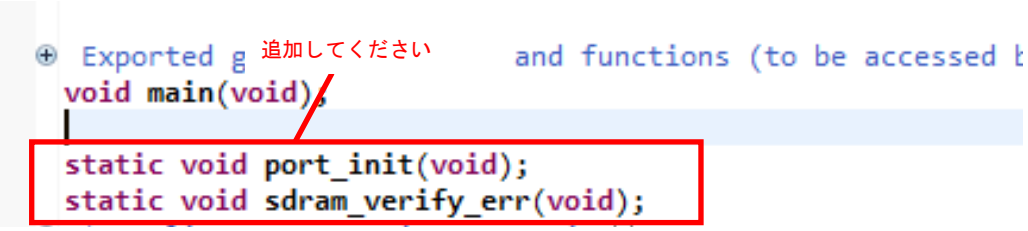
2-2) [main.c]の[#include “r_cmt_wait”]の下に、下記の define を追加し、LED0 及び LED1 に関連する define を LED0、LED1 で使用するポートに変更してください。今回は Renesas Starter Kit+ for RX72M の LED0、LED1 に合わせるため、LED0 を[P42]、LED1 を[PH0]に設定しています。

```
/***********************************************************************
Macro definitions
***********************************************************************/
/* **** LEDs **** */
/* ==== LED0 (SDRAM verify OK) ==== */
#define LED0_REG_PODR     PORT4.PODR.BIT.B2     /* LED0 Output data store bit */
#define LED0_REG_PDR      PORT4.PDR.BIT.B2      /* LED0 I/O select bit */
#define LED0_REG_PMR      PORT4.PMR.BIT.B2      /* LED0 Pin mode control bit */
/* ==== LED1 (SDRAM verify error) ==== */
#define LED1_REG_PODR     PORTH.PODR.BIT.B0     /* LED1 Output data store bit */
#define LED1_REG_PDR      PORTH.PDR.BIT.B0      /* LED1 I/O select bit */
#define LED1_REG_PMR      PORTH.PMR.BIT.B0      /* LED1 Pin mode control bit */
/* ==== LEDs output data ==== */
#define LED_ON            (0)                   /* LED on */
#define LED_OFF           (1)                   /* LED off */
```

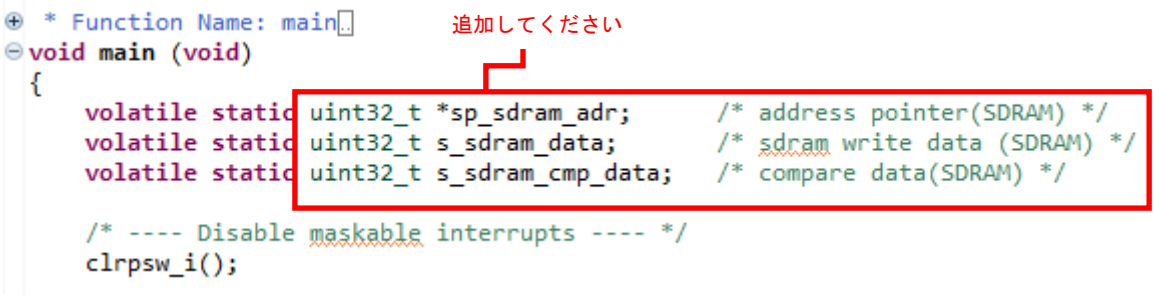
2-3) [main.c]の[#define LED_OFF (1)]の下に、[SDRAM_TOP]及び[SDRAM_END]の define を追加し、接続している SDRAM 及び使用しているデバイスの外部アドレス空間(SDRAM 領域)に合わせて変更してください。今回はアプリケーションノートの仕様に合わせるため、[SDRAM_TOP]を[(void*)(0x08000000)]、[SDRAM_END]を[(void*)(0x09000000)]に設定しています。

```
/* **** SDRAM address **** */
#define SDRAM_TOP     (void*)(0x08000000)          /* SDRAM top address 0x0800 0000 */
#define SDRAM_END     (void*)(0x09000000)          /* SDRAM end address 0x0900 0000 */
```

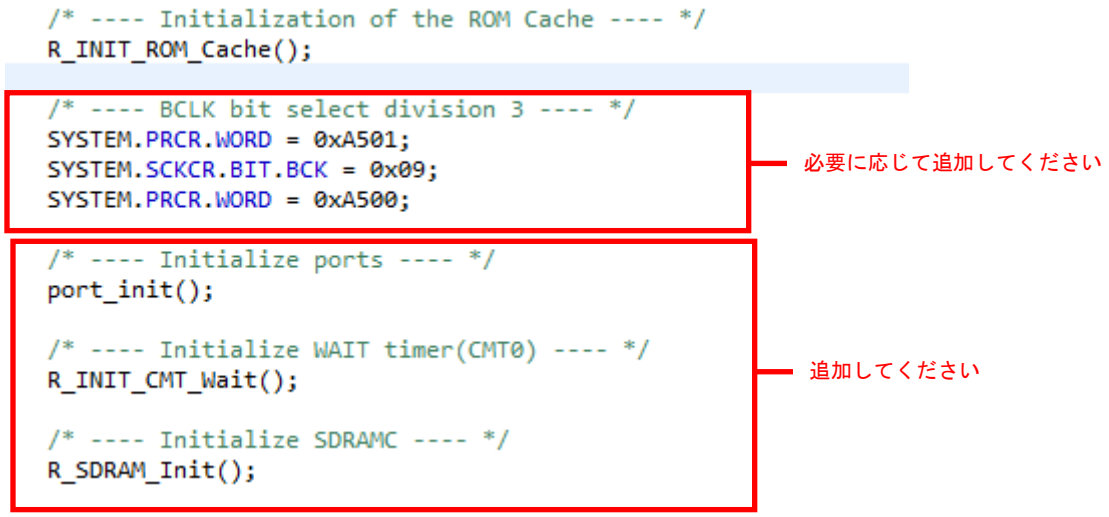
2-4) [main.c]に[port_init]関数及び[sdram_verify_err]関数のプロトタイプ宣言を追加してください。



2-5) [main.c] の[main]関数に[*sp_sdram_adr]及び[s_sdram_data]、[s_sdram_data]変数を、接続している SDRAM のデータバス幅に合わせた型で定義してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため[uint32_t]で定義しています。



2-6) [main.c]の[main]関数に[port_init]関数及び[R_INIT_CMT_Wait]関数、[R_SDRAM_Init]関数の呼び出し処理を while 文の前に追加してください。
また、接続している SDRAM の仕様に合わせて[外部バスクロック(BCLK)選択ビット]を変更する処理を追加してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、追加は不要です。



2-7) [main.c]の[main]関数、[R_SDRAM_Init]関数の下に、下記の SDRAM アクセス処理及び LED0 点灯処理を追加してください。この処理はアプリケーションノートで使用されている処理と同じものです。

```
/*********************************
 *    Memory access (SDRAM)
 *********************************/

/* ---- Data initialize for SDRAM    ---- */
for(sp_sdram_adr = SDRAM_TOP; sp_sdram_adr < SDRAM_END; sp_sdram_adr++)
{
      *sp_sdram_adr = 0x00000000;                /* Initialize data */
}

/* ---- Write data for SDRAM    ---- */
s_sdram_data = 0x00000000;                    /* SDRAM write data initialize */
for(sp_sdram_adr = SDRAM_TOP; sp_sdram_adr < SDRAM_END; sp_sdram_adr++)
{
      *sp_sdram_adr = s_sdram_data++;       /* Write increment data */
}

/* ---- Verify SDRAM data    ---- */
s_sdram_cmp_data = 0x00000000;                    /* SDRAM verify data initialize */
for(sp_sdram_adr = SDRAM_TOP; sp_sdram_adr<SDRAM_END; sp_sdram_adr++)
{
      /* ---- Verify error check    ---- */
      if(s_sdram_cmp_data != (*sp_sdram_adr))
      {
            /* ---- Verify error    ---- */
            sdram_verify_err();
      }
      s_sdram_cmp_data++;                         /* Verify data increment */

}

/* LED0 ON (SDRAM verify OK) */
LED0_REG_PODR = LED_ON;
```

2-8) [main.c]に下記の[port_init]関数及び[sdram_verify_err]関数の実体を定義してください。

```
static void port_init(void)
{

    /* ---- Initialize LEDs ---- */

    /* Set port output data - LEDs OFF */
    LED0_REG_PODR = LED_OFF;
    LED1_REG_PODR = LED_OFF;

    /* Set port direction - Output */
    LED0_REG_PDR = 1;
    LED1_REG_PDR = 1;

    /* Set port mode - Use pin as general I/O port */
    LED0_REG_PMR = 0;
    LED1_REG_PMR = 0;
}

static void sdram_verify_err(void)
{

    /* LED1 ON (SDRAM verify error) */
    LED1_REG_PODR = LED_ON;

    while (1)
    {
    }

}
```

8.3.6　r_sdram_api.c の設定

SDRAM 接続で使用する端子のポート方向を入力に変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、D16 から D31 に対応する PORT9 及び PORTG を入力ポートに設定しています。

```
/* ---- Set up pins to be used as i 設定してください      - */
PORTA.PDR.BYTE = 0x00;              /* PORTA(A0-A7)  input */
PORTB.PDR.BYTE = 0x00;              /* PORTB(A8-A14) input */
PORTD.PDR.BYTE = 0x00;              /* PORTD(D0-D7)  input */
PORTE.PDR.BYTE = 0x00;              /* PORTE(D8-D15) input */
PORT9.PDR.BYTE = 0x00;              /* PORT9(D16-D23) input */
PORTG.PDR.BYTE = 0x00;              /* PORTG(D24-D31) input */
PORT6.PDR.BYTE = 0x00;              /* SDCS#, RAS#,CAS#, WE#, (
PORT7.PDR.BIT.B0 = 0;               /* SDCLK input */
```

8.3.7　r_sdram_api.h の設定

ポーティング先の環境に合わせて[r_sdram_api.h]を変更します。

以下に RX72M(Renesas Starter Kit+ for RX72M)にポーティングするときの設定値を示します。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。また、7 SDRAM の仕様に対する対象デバイスにおけるレジスタ設定の考え方も参照してください。

1)　アドレス出力許可レジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_MPC_PFAOE0](アドレス出力許可レジスタ 0)及び[SDRAM_REG_MPC_PFAOE1](アドレス出力許可レジスタ 1)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、[SDRAM_REG_MPC_PFAOE0]を”0xFF”に変更しています。[SDRAM_REG_MPC_PFAOE1]の変更は不要です。

```
#define SDRAM_REG_MPC_PFAOE0  (0xFF)      /* Address Output Enable Register 0
#define SDRAM_REG_MPC_PFAOE1  (0x00)      /* Address Output Enable Register 1
                                          設定してください
```

2)　外部バス制御レジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_MPC_PFBCR0](外部バス制御レジスタ 0)、[SDRAM_REG_MPC_PFBCR1](外部バス制御レジスタ 1)、[SDRAM_REG_MPC_PFBCR2](外部バス制御レジスタ 2)及び[SDRAM_REG_MPC_PFBCR3](外部バス制御レジスタ 3)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、[SDRAM_REG_MPC_PFBCR0]を”0x31”、 [SDRAM_REG_MPC_PFBCR3]を”0x40”に変更しています。[SDRAM_REG_MPC_PFBCR1]及び[SDRAM_REG_MPC_PFBCR2]の変更は不要です。

```
#define SDRAM_REG_MPC_PFBCR0  (0x31)      /* External Bus Control Register 0 set value */
#define SDRAM_REG_MPC_PFBCR1  (0xD0)      /* External Bus Control Register 1 set value */
#define SDRAM_REG_MPC_PFBCR2  (0x00)      /* Ext 設定してください      gister 2 set value */
#define SDRAM_REG_MPC_PFBCR3  (0x40)      /* External Bus Control Register 3 set value */
```

3)　SDC 制御レジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDCCR](SDC 制御レジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、”0x10”に変更しています。

```
#define SDRAM_REG_BSC_SDCCR   (0x10)      /* SDC Control Register set value
                                          設定してください
```

4) SDC モードレジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDCMOD](SDC モードレジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、変更は不要です。

5) SDRAM アクセスモードレジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDAMOD](SDRAM アクセスモードレジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、変更は不要です。

6) SDRAM リフレッシュ制御レジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDRFCR](SDRAM リフレッシュ制御レジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、”0x44E0”に変更しています。

```
#define SDRAM_REG_BSC_SDRFCR  (0x44E0)      /* SDRAM Refresh Control Register set value */
```

設定してください

7) SDRAM 初期化レジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDIR](SDRAM 初期化レジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、”0x0022”に変更しています。

```
#define SDRAM_REG_BSC_SDIR    (0x0022)      /* SDRAM Initialization Register set value */
```

設定してください

8) SDRAM アドレスレジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDADR](SDRAM アドレスレジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、”0x00”に変更しています。

```
#define SDRAM_REG_BSC_SDADR   (0x00)      /* SDRAM Address Register set
```

設定してください

9) SDRAM タイミングレジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDTR](SDRAM タイミングレジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、”0x00031303”に変更しています。

```
#define SDRAM_REG_BSC_SDTR    (0x00031303)  /* SDRAM Timing Register set value */
```

設定してください

10) SDRAM モードレジスタの設定

接続している SDRAM の仕様に合わせて[SDRAM_REG_BSC_SDMOD](SDRAM モードレジスタ)の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、変更は不要です。

9. 他 RX ファミリにスマート・コンフィグレータを使用しているサンプルコードをポーティングする方法

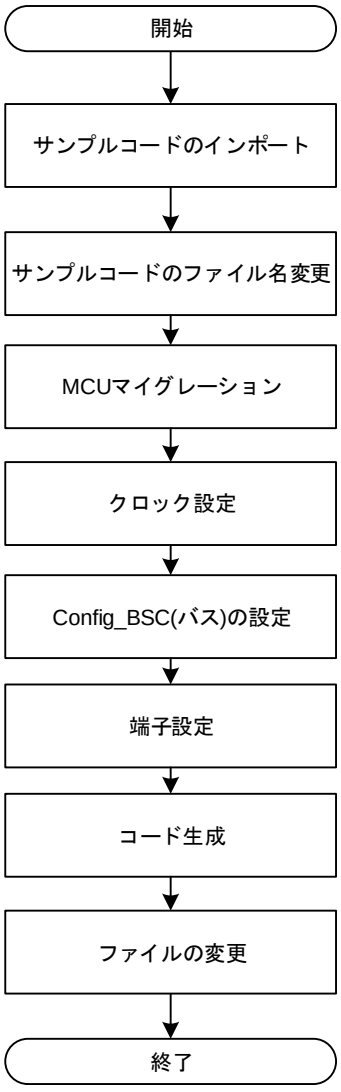
本アプリケーションノートに同梱されているスマート・コンフィグレータを使用しているサンプルコードは SDRAMC を搭載する他 RX ファミリにポーティングできます。本章では RX65N のスマート・コンフィグレータを使用しているサンプルコードを RX72M(Renesas Starter Kit+ for RX72M)にポーティングする例を示します。

9.1 ポーティングする前に

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 SDRAMC の仕様
- ポーティング元とポーティング先の CMT の仕様

9.2 ポーティング手順フロー



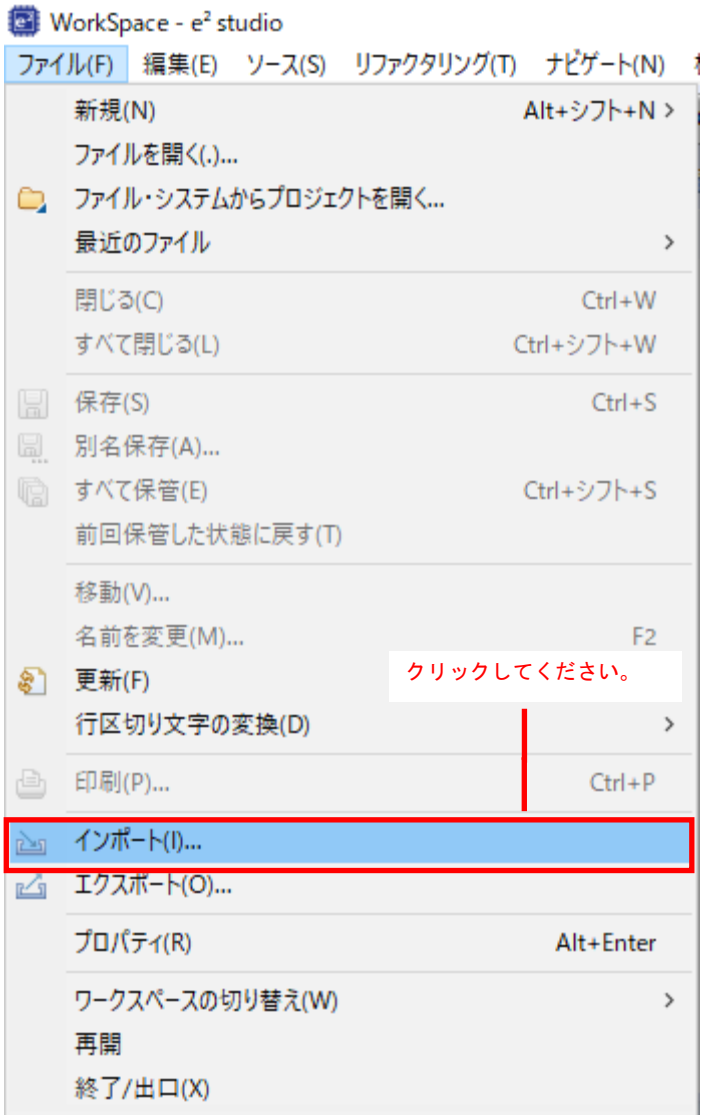
9.3　ポーティング手順

9.3.1　サンプルコードのインポート

本アプリケーションのサンプルコードをインポートします。

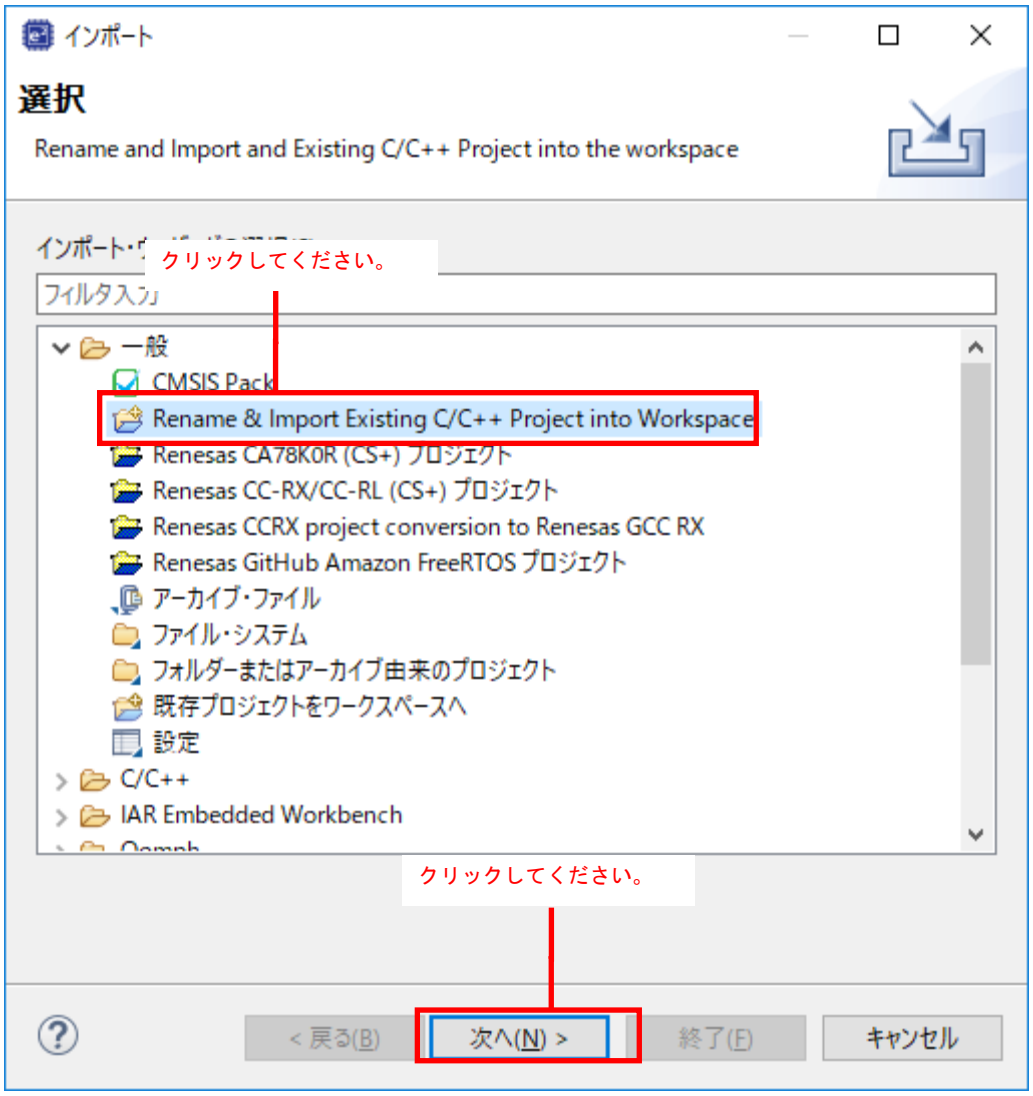
1) e² studio を起動して[ファイル(F)]をクリックしてください。

2) [インポート(I)]をクリックしてください。



3) [Rename & Import Existing C/C++ Project into Workspace]をクリックしてください。

4) [次へ(N)>]をクリックしてください。



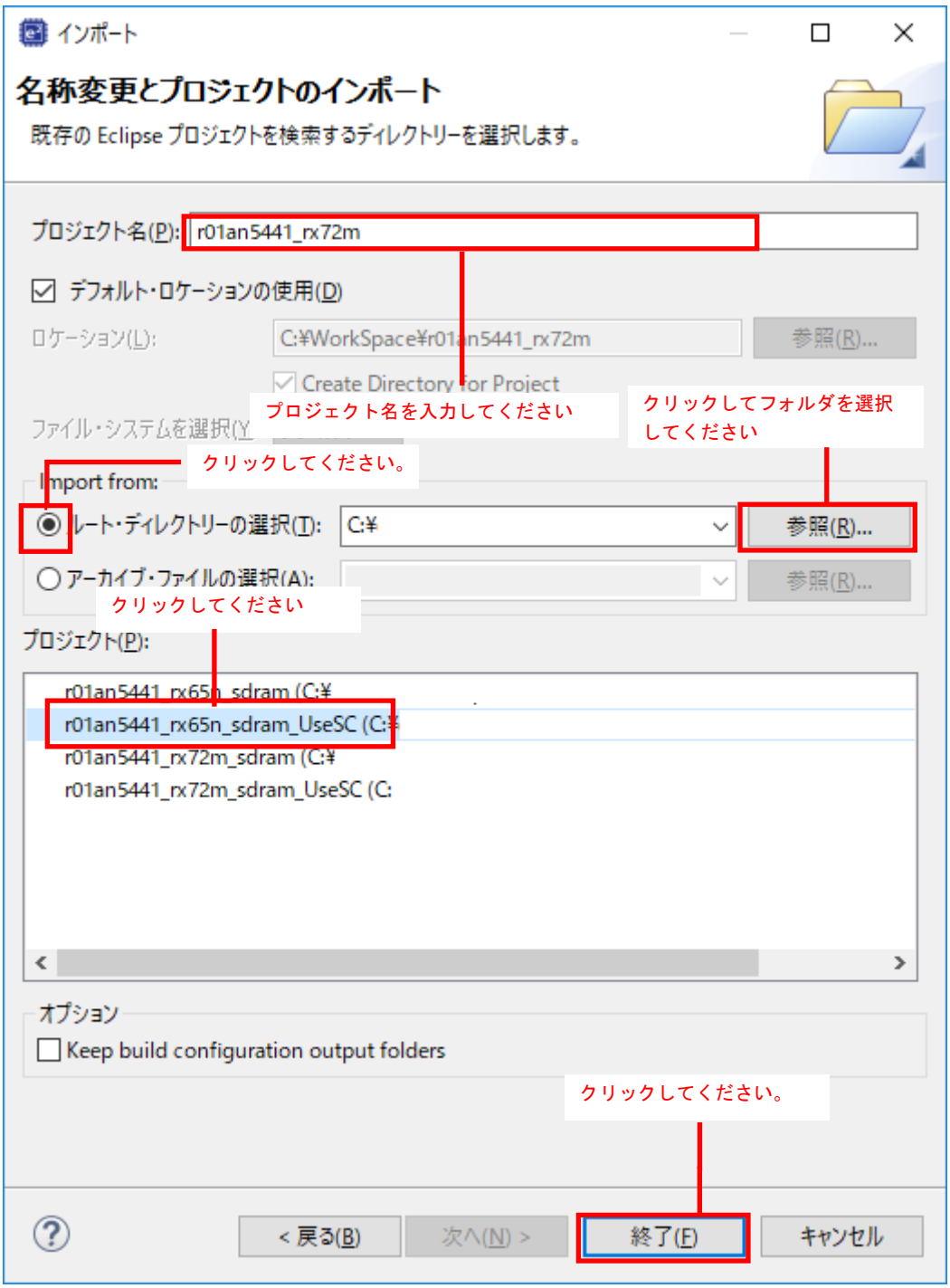
5) 任意のプロジェクト名を入力してください。

6) [ルート・ディレクトリの選択(T)]をクリックしてください。

7) [参照(R)]をクリックし、ダウンロードしたサンプルコードがあるフォルダを選択して下さい。

8) [r01an5441_rx65n_sdram_UseSC]をクリックしてください。

9) [終了(F)]をクリックしてください。

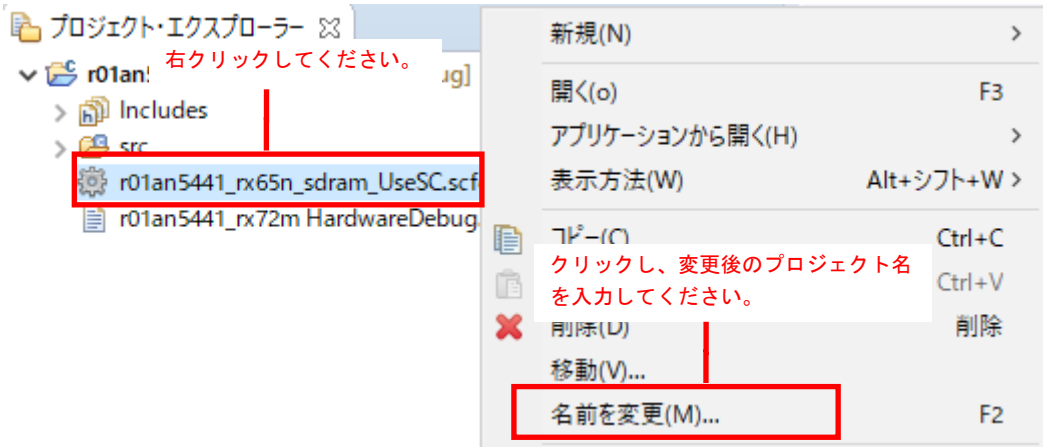


9.3.2　サンプルコードのファイル名変更

インポートしたサンプルコードのファイル名を変更します。

1) [r01an5441_rx65n_sdram_UseSC.scfg]を右クリックします

2) [名前を変更(M)]をクリックし、変更後のプロジェクト名を入力してください

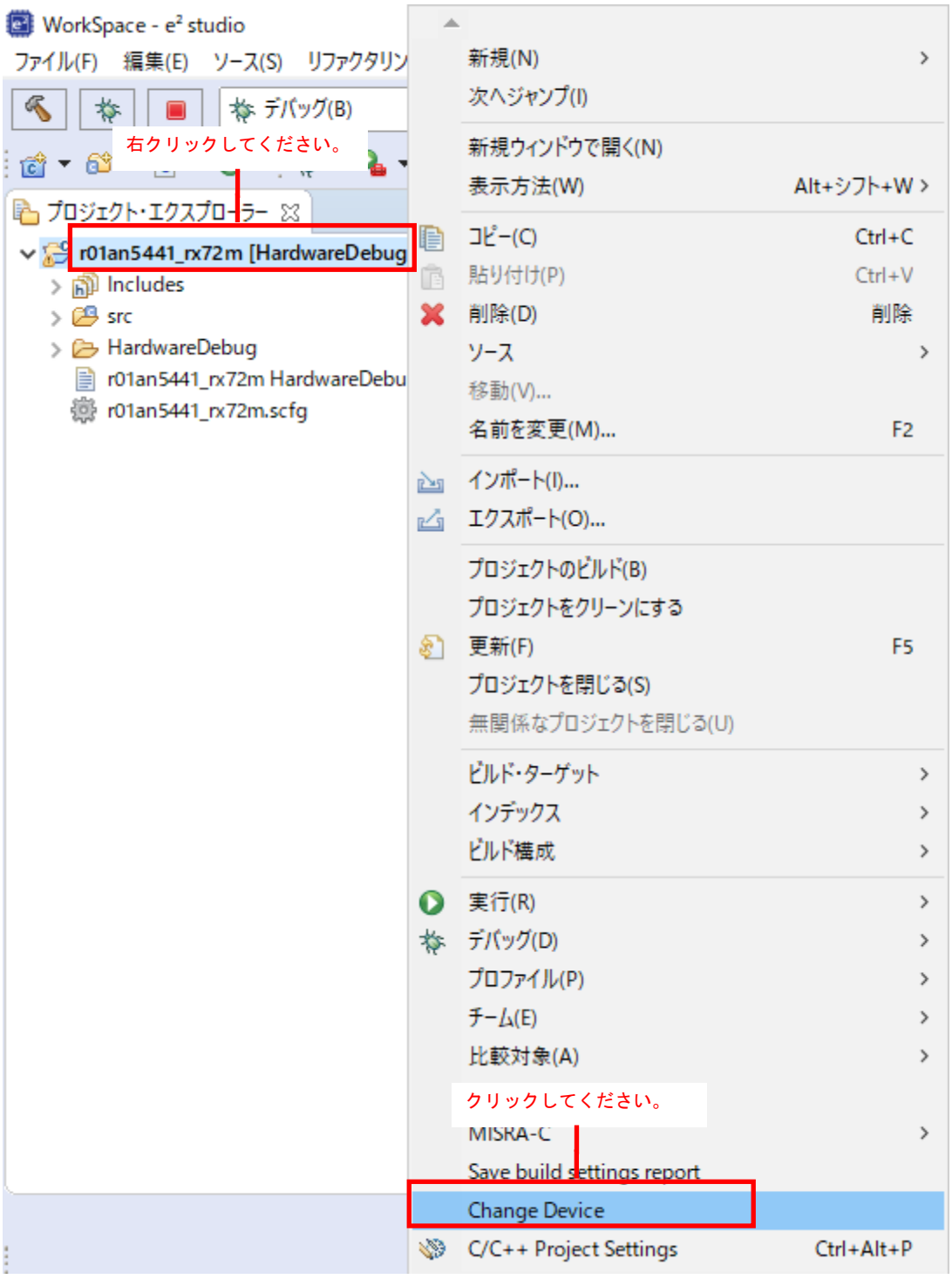


9.3.3　MCU マイグレーション

インポートしたサンプルコードの MCU マイグレーションを行います。

1) [<プロジェクト名>]のディレクトリを右クリックします

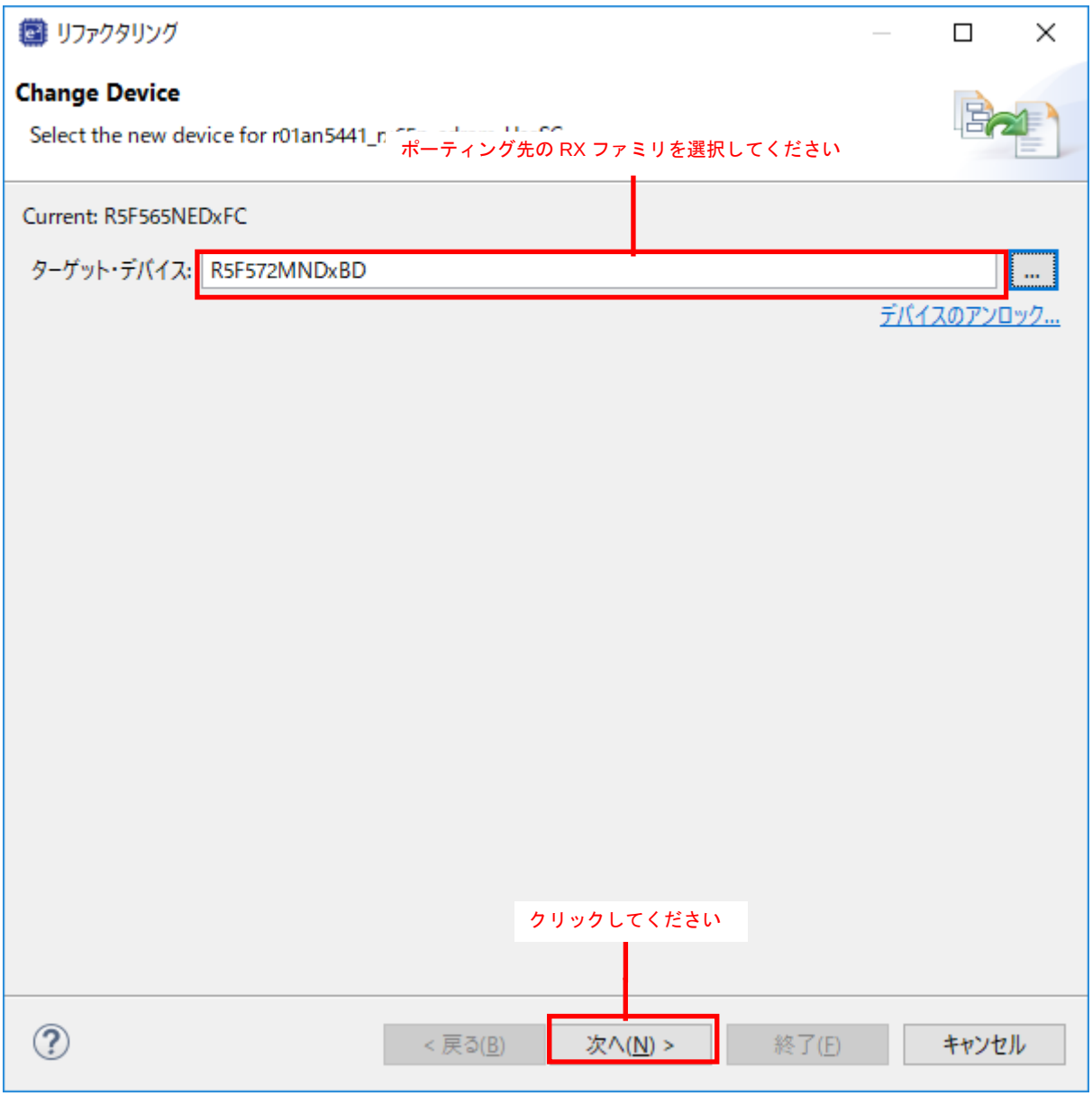
2) [Change Device]をクリックしてください



3) [ターゲットデバイス:]を[R5F572MNDxBD]に変更してください。

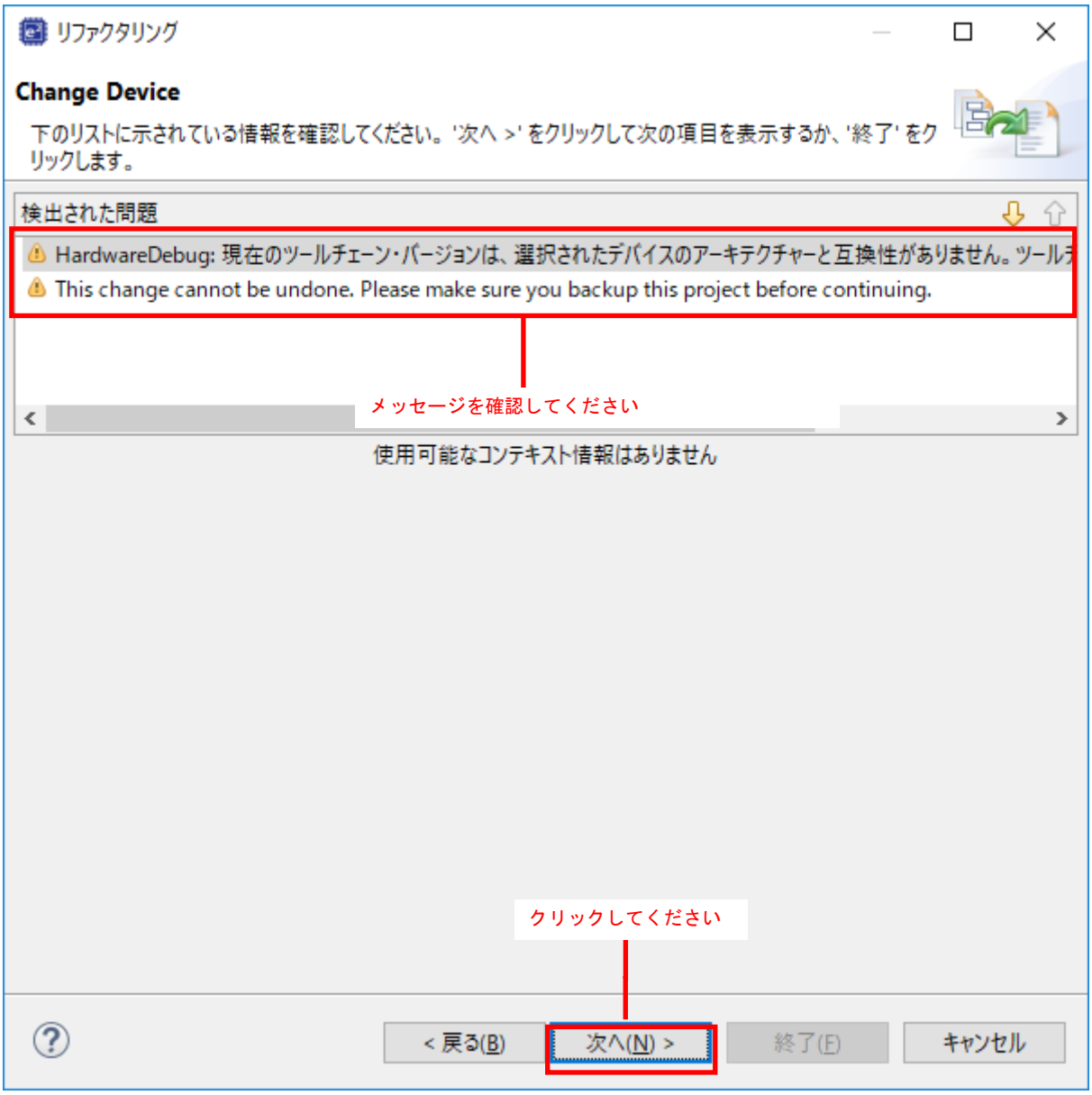
(他 RX ファミリにポーティングする場合は、ポーティング先の RX ファミリに変更してください)

4) [次へ(N)>]をクリックしてください。



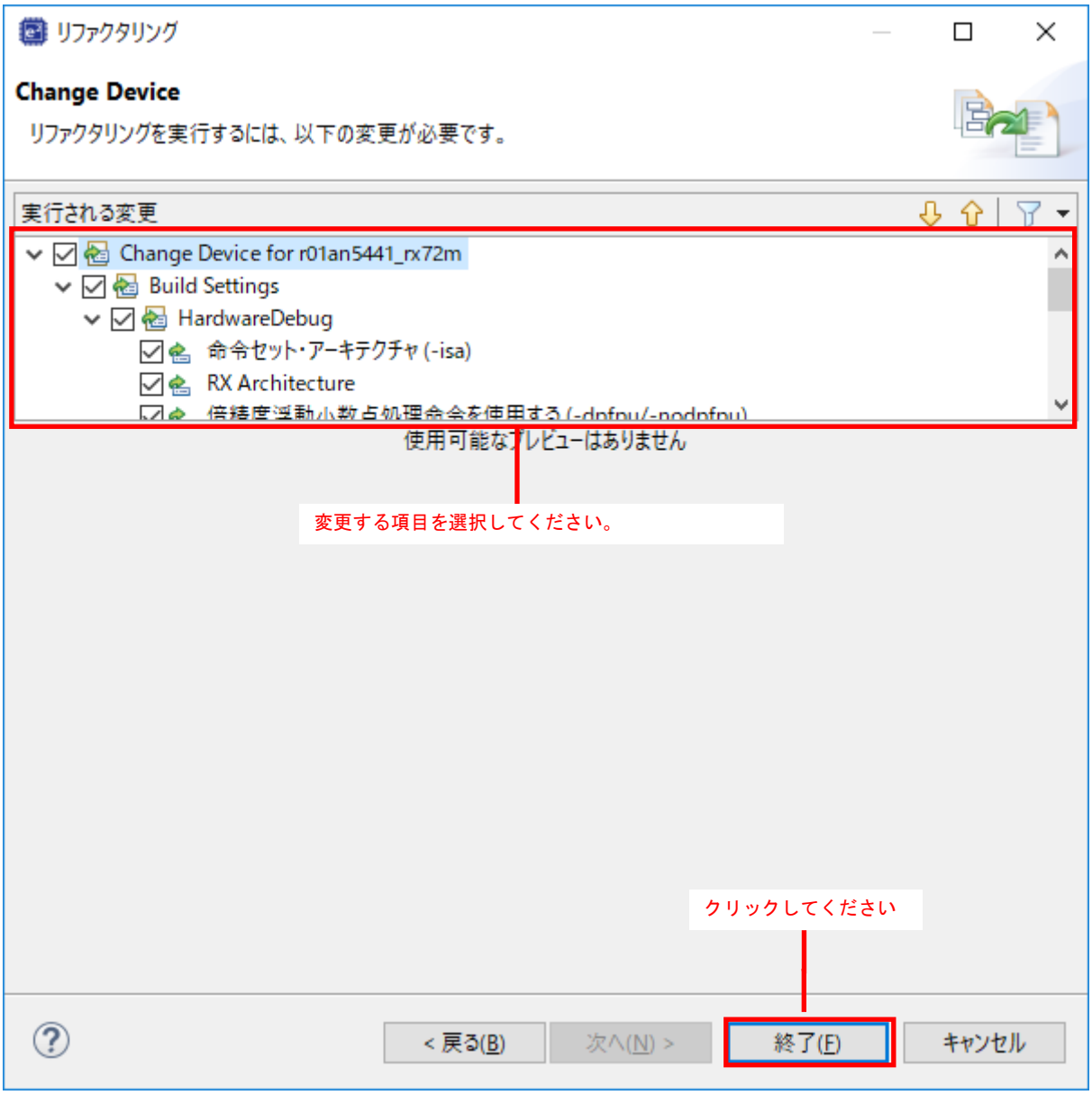
5) [検出された問題]に表示されたメッセージを確認してください。

6) [次へ(N)>]をクリックしてください。



7) [実行される変更]から変更する項目を選択して下さい。

8) [終了(F)]をクリックしてください。

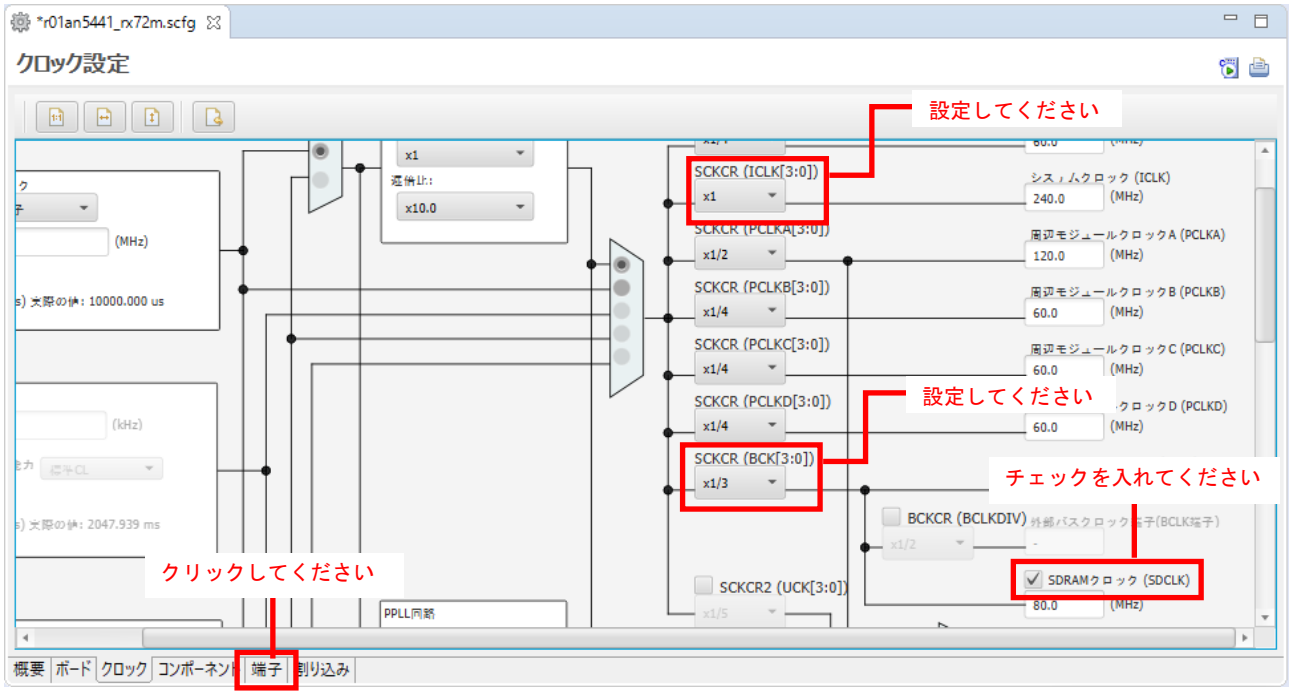


9.3.4　クロック設定

スマート・コンフィグレータを使用し、クロック設定を変更します。

1) 作成したプロジェクト内にある[<プロジェクト名>.scfg]を開き、下部にあるタブの[クロック]をクリックします。

2) 接続している SDRAM の仕様に合わせて[SCKCR(BCK[3:0])]を変更し、SDRAM クロック(SDCLK)にチェックを入れてください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様及びアプリケーションノートの仕様に合わせるため、[SCKCR(ICLK[3:0])]の設定を 1 分周、[SCKCR(BCK[3:0])]の設定を 3 分周に設定しています。



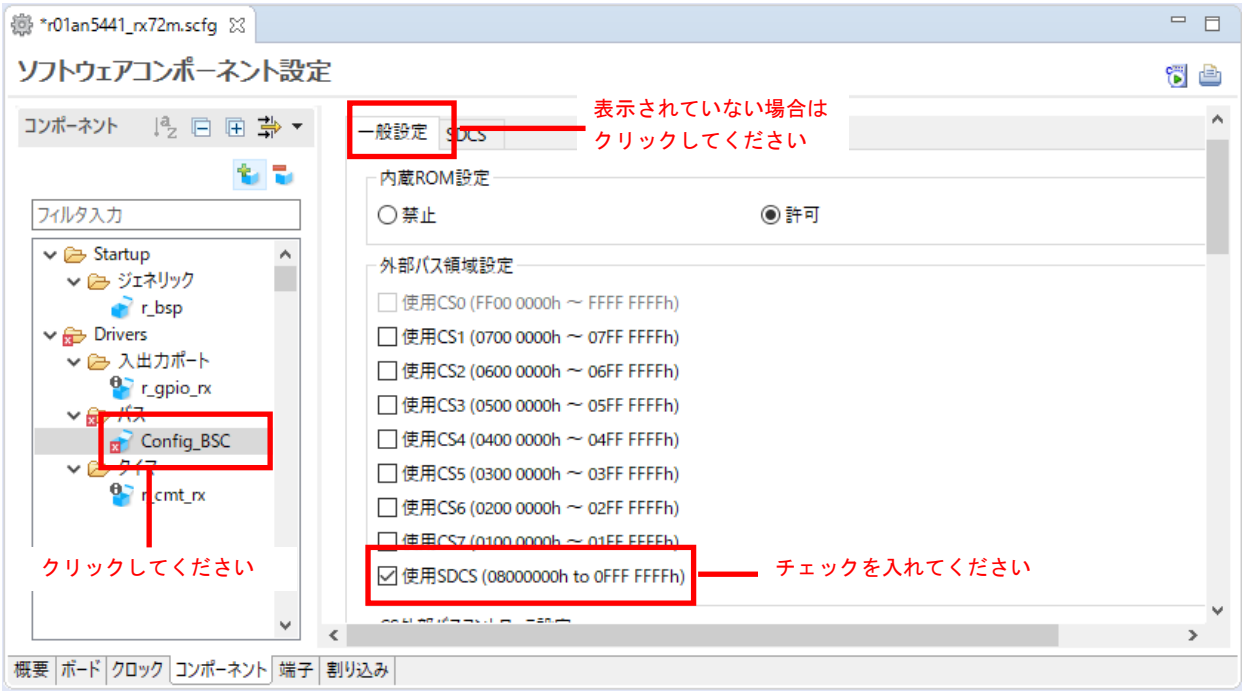
9.3.5　Config_BSC(バス)の設定

ポーティング先の環境に合わせて Config_BSC(バス)の設定を変更します。

以下に RX72M(Renesas Starter Kit+ for RX72M)にポーティングするときの設定値を示します。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[Config_BSC]の設定値を変更してください。また、7 SDRAM の仕様に対する対象デバイスにおけるレジスタ設定の考え方も参照してください。

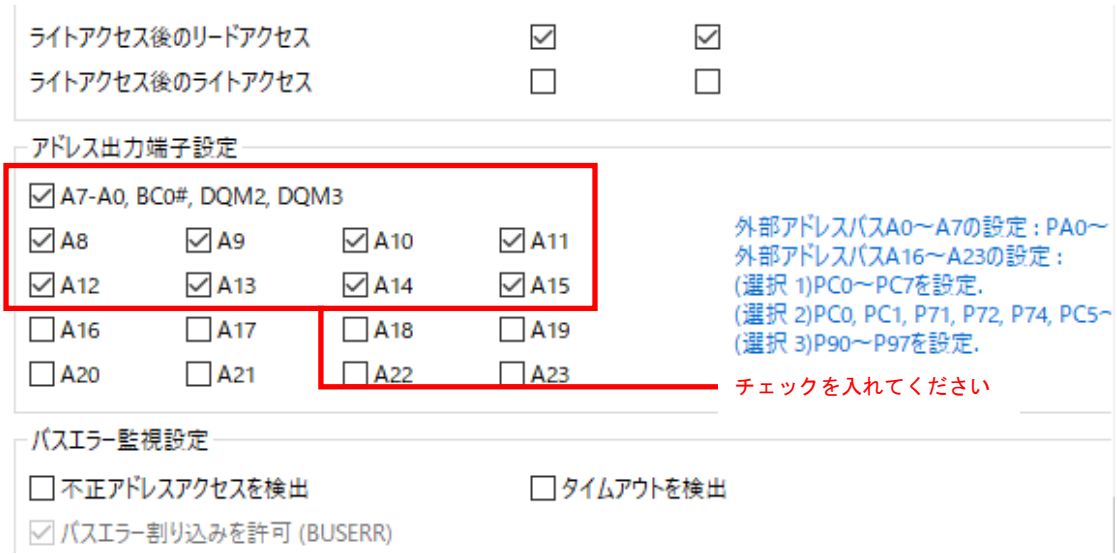
1)　外部バス領域設定

コンポーネントから[Config_BSC]クリックし設定画面を表示してください。表示された設定画面が[一般設定]タブであることを確認してください。SDRAM を使用するため、[外部バス領域設定]の[使用SDCS]にチェックを入れてください。



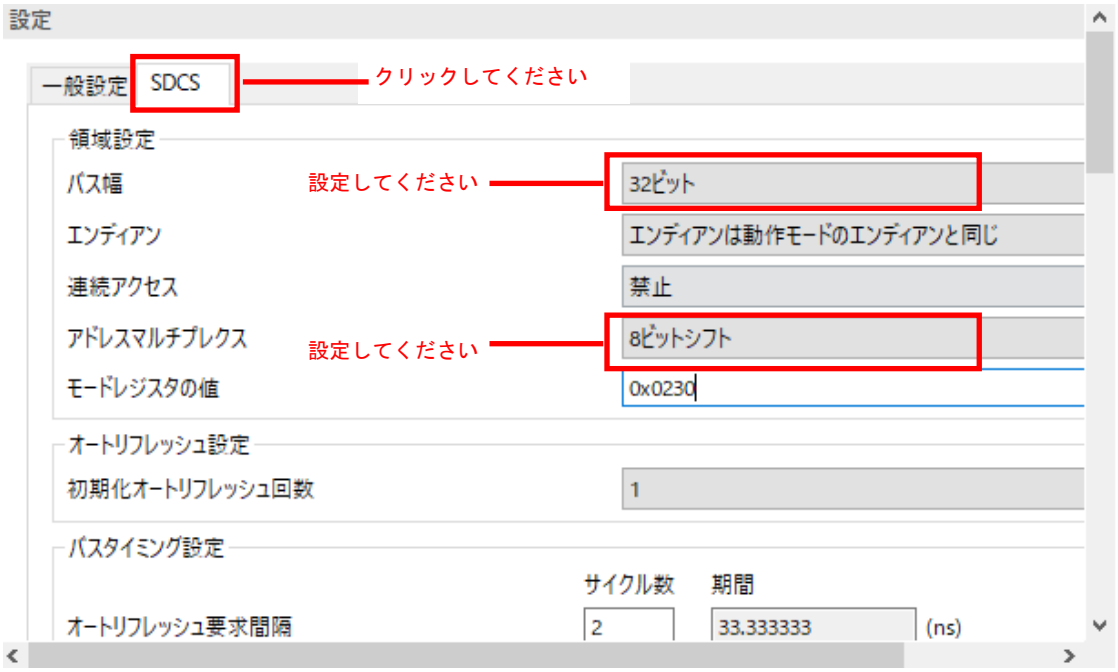
2)　アドレス出力端子設定

[一般設定]タブの[アドレス出力端子]を、接続している SDRAM の仕様に合わせて設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、[A7-A0, BC0#, DQM2, DQM3]及び[A8]から[A15]を出力する設定に変更しています。



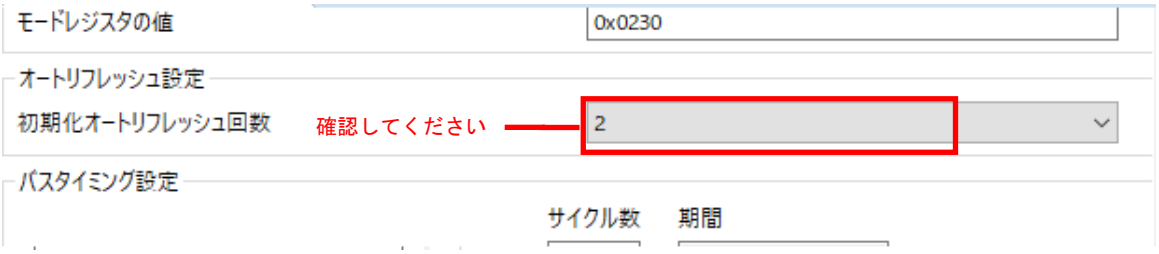
3)　領域設定

[SDCS]タブをクリックし[領域設定]の内容を、接続している SDRAM の仕様に合わせて設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、[バス幅]を[32 ビット]、[アドレスマルチプレクス]を[8 ビットシフト]に変更しています。



4)　オートリフレッシュ設定

[SDCS]タブの[オートリフレッシュ設定]の内容を接続している SDRAM の仕様に合わせて設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、変更は不要です。



5)　バスタイミング設定

[SDCS]タブの[バスタイミング設定]の内容を、接続している SDRAM の仕様に合わせて設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、下図のような設定に変更しています。

- [オートリフレッシュ要求間隔]は[リフレッシュサイクル / ロウアドレス数]以内になるようにしてください。MT48LC4M32B2P-6A の場合オートリフレッシュ要求間隔は 15625ns です。今回は 1249 サイクルに設定することで、15612.5ns に設定しています。
- [オートリフレッシュサイクル/セルフリフレッシュ解除サイクル数]は tRFC 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRFC は 60ns です。今回は 5 サイクルに設定することで、62.5ns に設定しています。
- [初期化オートリフレッシュ間隔]は tRFC 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRFC は 60ns です。今回は 5 サイクルに設定することで、62.5ns に設定しています。
- [初期化プリチャージサイクル数]は tRP 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRP は 18ns です。今回は 3 サイクルに設定することで、37.5ns に設定しています。
- [SDRAMC カラムレイテンシ]は SDRAM に設定したカラムレイテンシと同じになるように設定してください。今回は SDRAM の設定に合わせ 3 サイクルに設定しています。
- [ライトリカバリ期間]は tWR 以上になるように設定してください。MT48LC4M32B2P-6A の場合で SDCLK が 80MHz であるとき、tWR は 19.5ns です。今回は 2 サイクルに設定することで、25ns に設定しています。
- [ロウプリチャージ期間]は tRP 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRP は 18ns です。今回は 2 サイクルに設定することで、25ns に設定しています。
- [ロウカラムレイテンシ]は tRCD 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRCD は 18ns です。今回は 2 サイクルに設定することで、25ns に設定しています。
- [ロウアクティブ期間]は tRAS 以上になるように設定してください。MT48LC4M32B2P-6A の場合 tRAS は 42ns です。今回は 4 サイクルに設定することで、50ns に設定しています。

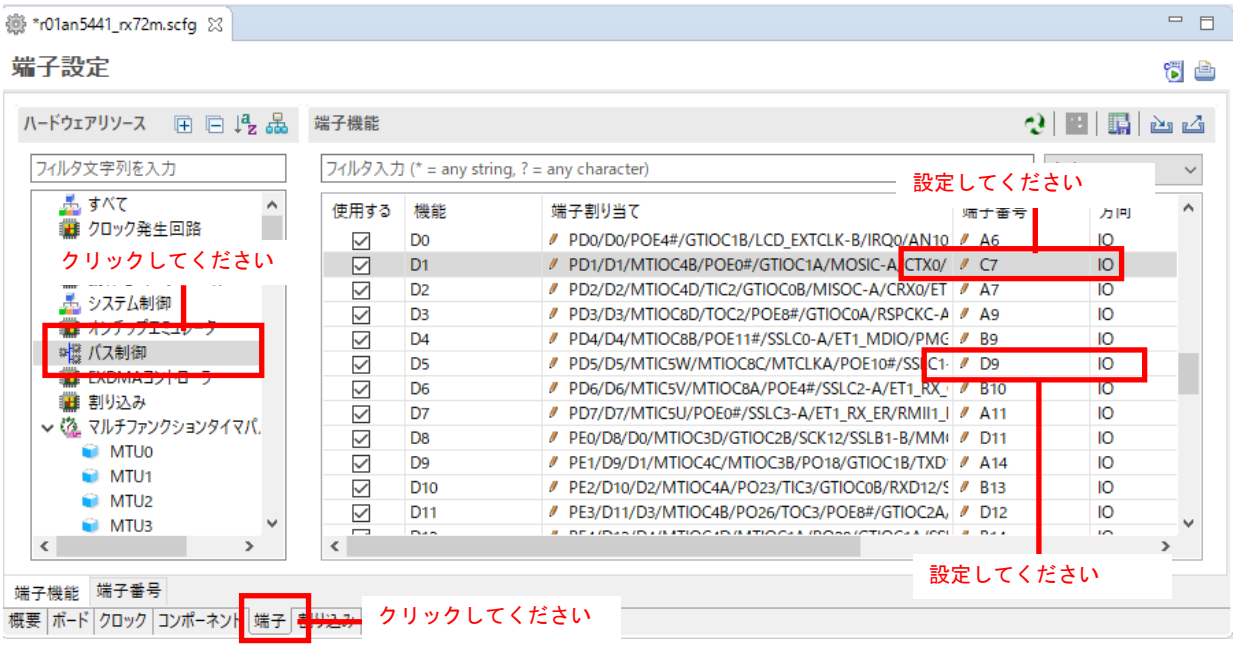
9.3.6　端子設定

スマート・コンフィグレータを使用し、使用する端子の設定を行います。

1) 作成したプロジェクト内にある[<プロジェクト名>.scfg]を開き、下部にあるタブの[端子]をクリックします。

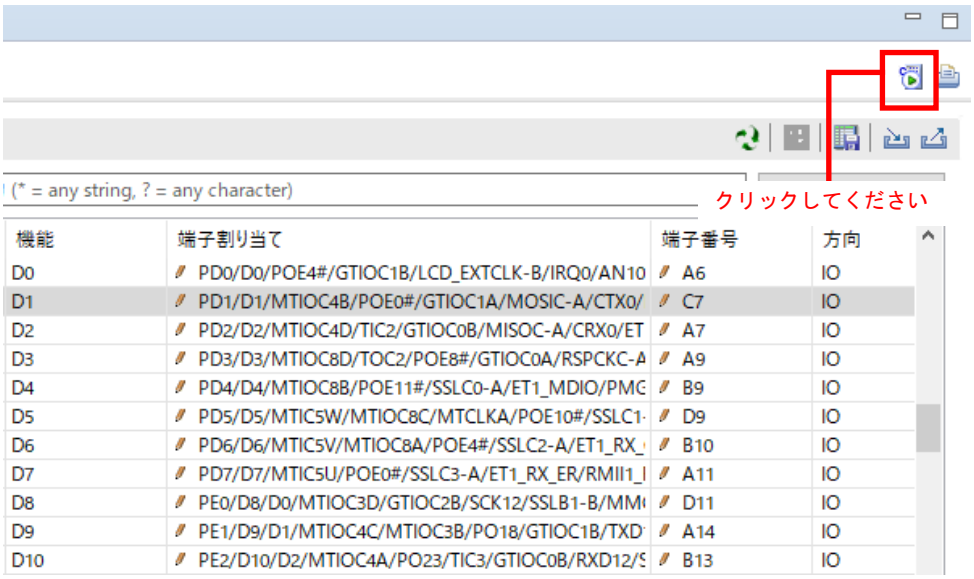
2) ハードウェアリソースの[バス制御]をクリックしてください。

3) 接続している SDRAM の仕様に合わせて端子設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため、[D1]の端子番号を[C7]、[D5]の端子番号を[D9]に変更しています。



9.3.7　コード生成

[コードの生成]ボタンをクリックし、コードを生成してください。



9.3.8　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、[main.c]を変更します。

1) 使用する LED 端子に合わせて、[LED0_PORT_PIN]及び[LED1_PORT_PIN]の設定を変更してください。今回は Renesas Starter Kit+ for RX72M に搭載されている LED に合わせるため[LED0_PORT_PIN]に[GPIO_PORT_4_PIN_2]、[LED1_PORT_PIN]に[GPIO_PORT_H_PIN_0]を設定しています。

```
Macro definitions
/* **** LEDs **** */
#define LED0_PORT_PIN    GPIO_PORT_4_PIN_2    /* LED0 Us
#define LED1_PORT_PIN    GPIO_PORT_H_PIN_0    /* LED1 Use port */
```

設定してください

2) [main]関数の[*sp_sdram_adr]及び[s_sdram_data]、[s_sdram_data]変数を、接続している SDRAM のデータバス幅に合わせた型で定義してください。今回は Renesas Starter Kit+ for RX72M に搭載されている SDRAM の仕様に合わせるため[uint32_t]で定義しています。

```
 * Function Name: main
void main (void)
{
    volatile static uint32_t *sp_sdram_adr;      /* address po
    volatile static uint32_t s_sdram_data;       /* sdram writ
    volatile static uint32_t s_sdram_cmp_data;   /* compare data(SDRAM) */
```

設定してください

10. 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。

11. 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（R01UH0590）

RX72M グループ　ユーザーズマニュアル　ハードウェア編（R01UH0804）

RX671 グループ　ユーザーズマニュアル　ハードウェア編（R01UH0899）

RX72N グループ　ユーザーズマニュアル　ハードウェア編（R01UH0824）

（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	Jun.30.20	ー	初版発行
1.10	Jan.10.24	全体	RSK-RX671、EK-RX671、RSK-RX72N 評価ボード 向けサンプルプログラムの追加
		1	「要旨」内のマイコングループ記載追加
		4	「1 .仕様」内の SDRAM 記載追加 EK-RX671 と RSK ボード別に LED 点灯動作追加
		5	表 1.4 追加
		6.7	表 2.1 表 2.2 名称変更
		8.9.10	表 2.3 表 2.4 表 2.5 追加
		10	RX671、RX72N 関連アプリケーションノート追加
		13	図 5.3 図 5.4 追加
		15.16	表 5.3 表 5.4 表 5.5 追加
		27	RSK-RX671,RSK-RX72N,EK-RX671 向け定数定義追加
		42	図 6.16 内記載変更
		43	表 6.12 追加
		44	図 6.17 追加
		45	表 6.13 表 6.14 追加
		46	図 6.18 内記載変更
		81	RX671,RX72N 参考ドキュメント追加

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。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、お客様の責任において、お客様の機器・システムを設計ください。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. 当社製品または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. 当社製品を組み込んだ製品の輸出入、製造、販売、利用、配布その他の行為を行うにあたり、第三者保有の技術の利用に関するライセンスが必要となる場合、当該ライセンス取得の判断および取得はお客様の責任において行ってください。
5. 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。
6. 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。
7. あらゆる半導体製品は、外部攻撃からの安全性を 100%保証されているわけではありません。当社ハードウェア／ソフトウェア製品にはセキュリティ対策が組み込まれているものもありますが、これによって、当社は、セキュリティ脆弱性または侵害（当社製品または当社製品が使用されているシステムに対する不正アクセス・不正使用を含みますが、これに限りません。）から生じる責任を負うものではありません。当社は、当社製品または当社製品が使用されたあらゆるシステムが、不正な改変、攻撃、ウイルス、干渉、ハッキング、データの破壊または窃盗その他の不正な侵入行為（「脆弱性問題」といいます。）によって影響を受けないことを保証しません。当社は、脆弱性問題に起因しまたはこれに関連して生じた損害について、一切責任を負いません。また、法令において認められる限りにおいて、本資料および当社ハードウェア／ソフトウェア製品について、商品性および特定目的との合致に関する保証ならびに第三者の権利を侵害しないことの保証を含め、明示または黙示のいかなる保証も行いません。
8. 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。
9. 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。
10. 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。
11. 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。
12. 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。
13. 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。
14. 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.5.0-1　2020.10)

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。