RL78/G23

SMS 7セグメント LED のダイナミック点灯制御

要旨

本アプリケーションノートでは、SNOOZE モード・シーケンサ (SMS) を使った 4 桁の 7 セグメント LED のダイナミック点灯を実現する方法を説明します。使用する 7 セグメント LED はカソードコモンです。

対象デバイス

RL78/G23

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

目次

1. 仕様

本アプリケーションノートでは、CPUがRTCから「分」と「秒」のデータを読み出して対応するセグメントデータに変換し、SMSが2ミリ秒ごとにCPUの介在なしで各桁のセグメントデータを読み出して7セグメントLEDに分と秒を表示します。

このため、CPUが停止中でも7セグメントLEDのダイナミック点灯が可能です。SNOOZEモード・シーケンサ(SMS)はCPUよりも動作電流が小さいため、CPUの代わりにSMSで処理を実行させることでシステムの低消費電力化を実現できます。

1.1 構成

表1-1に表1-1 使用する周辺機能と用途を示します。

表1-1 使用する周辺機能と用途

周辺機能	用途
32ビット・インターバル・タイマ	2msのインターバル・タイマ(SMS起動用)
P1	セグメント信号ドライブ用
P7	桁選択信号ドライブ用(P73、P72、P71、P70)
RTC	時間計測用

主な設定を説明します。

① 32ビット・インターバル・タイマの初期設定

表1-2に表1-2 ITL000の初期設定を示します。

表1-2 ITL000の初期設定

レジスタ名	設定値	設定項目
TML32EN	1	32ビット・インターバル・タイマにクロック供給
ITLCTL0	40H	16ビット・カウンタとして動作(動作は停止)
ITLMKF0	1	チャネル(0)コンペア一致ステータスをマスク
ITLS0	00H	チャネル(0)コンペア一致信号クリア
ITLIF	0	割り込みを要求クリア
ITLCSEL0	01H	カウント・クロック(f_{ITL0})はf_{IHP}を選択
ITLFDIV00	00H	f_{ITL0}を分周しないで使用
ITLCMP00	0F9FFH	インターバル時間は2ms

② RTCの初期設定

表1-3にRTCの初期設定を示します。

表1-3 RTCの初期設定

レジスタ名	設定値	設定項目
OSMC	0x00	サブシステム・クロックで動作
RTCWEN	1	16ビット・カウンタとして動作(動作は停止)
RTCMK	1	RTC割り込みマスク
RTCIF	0	RTC割り込み要求クリア
RTCC0	0x02	12時間制、1秒間隔割り込み
RTCC1	0x00	アラーム割り込み禁止

1.2 動作概要

7セグメントLEDのダイナミック点灯の動作の概要を示します。

CPUは、1秒ごとに発生するRTCの定周期割り込みでスタンバイ状態から復帰し、RTCから時刻情報を読み出します。読み出した時刻情報から分と秒の表示データ(セグメントデータ)を作成し、作成したデータをメモリに格納します。

SMSは、2ミリ秒ごとに発生する32ビット・インターバル・タイマのインターバル検出割り込みで起動され、表示桁のセグメントデータをメモリから読み出し、7セグメントLEDが接続されているポートにデータを出力します。

- 2ミリ秒ごとのインターバル検出割り込み(INTITL)でSMSを起動します。
- SMSが起動すると、コンペアー致検出フラグをクリアします。
- 全ての桁の選択信号をオフします。
- 表示のブランキング時間を待ちます。
- 桁の表示データを読み出し、7セグメントLEDが接続されているポートにデータを出力します。
- データの安定時間を待ちます。
- 桁の選択信号を出力します。
- 表示データのアドレスを更新します。
- 桁選択信号を右シフトして更新します。
- 全ての桁が完了したら、表示データの格納アドレスと桁選択信号を初期化します。
- SMS処理を終了し、次のインターバル検出割り込み(INTITL)を待ちます。

1.3 LEDの制御

カソードコモンの7セグメントLEDを使用し、ドライバICを使用せず、ポートで直接制御します。

P10 – P17でセグメント信号を制御し、P70 – P73でコモン信号を制御します。コモン信号の初期値としてポート7に0xFFF7を設定します。

2. SMS使用例

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	RL78/G23（R7F100GLG）
動作周波数	• 高速オンチップ・オシレータ・クロック：32MHz • CPU／周辺ハードウェア・クロック：32MHz
動作電圧	• 3.3V • LVD0動作（VLVD0）：リセット・モード 立ち上がり時TYP. 1.875V 立ち下がり時TYP. 1.835V
統合開発環境（CS+）	ルネサスエレクトロニクス製 CS+ for CC V8.07.00
Cコンパイラ（CS+）	ルネサスエレクトロニクス製 CC-RL V1.11
統合開発環境（e2 studio）	ルネサスエレクトロニクス製 e2 studio 2021-04 (21.4.0)
Cコンパイラ（e2 studio）	ルネサスエレクトロニクス製 CC-RL V1.11
統合開発環境（IAR）	IAR システム製 IAR Embedded Workbench for Renesas RL78 V4.21.1
Cコンパイラ（IAR）	
スマート・コンフィグレータ	V.1.0.1
ボードサポートパッケージ（r_bsp）	V.1.10
エミュレータ	CS＋、e2 studio：E2エミュレータLite IAR：E2エミュレータLite
使用ボード	RL78/G23-64 Fast Prototyping Board (RTK7RLG230CLG000BJ)

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。

RL78スマート・コンフィグレータ ユーザーガイド：CS+編（R20AN0580J）
RL78スマート・コンフィグレータ ユーザーガイド：e2 studio編（R20AN0579J）
RL78 スマート・コンフィグレータ ユーザーガイド：IAR 編（R20AN0581J）

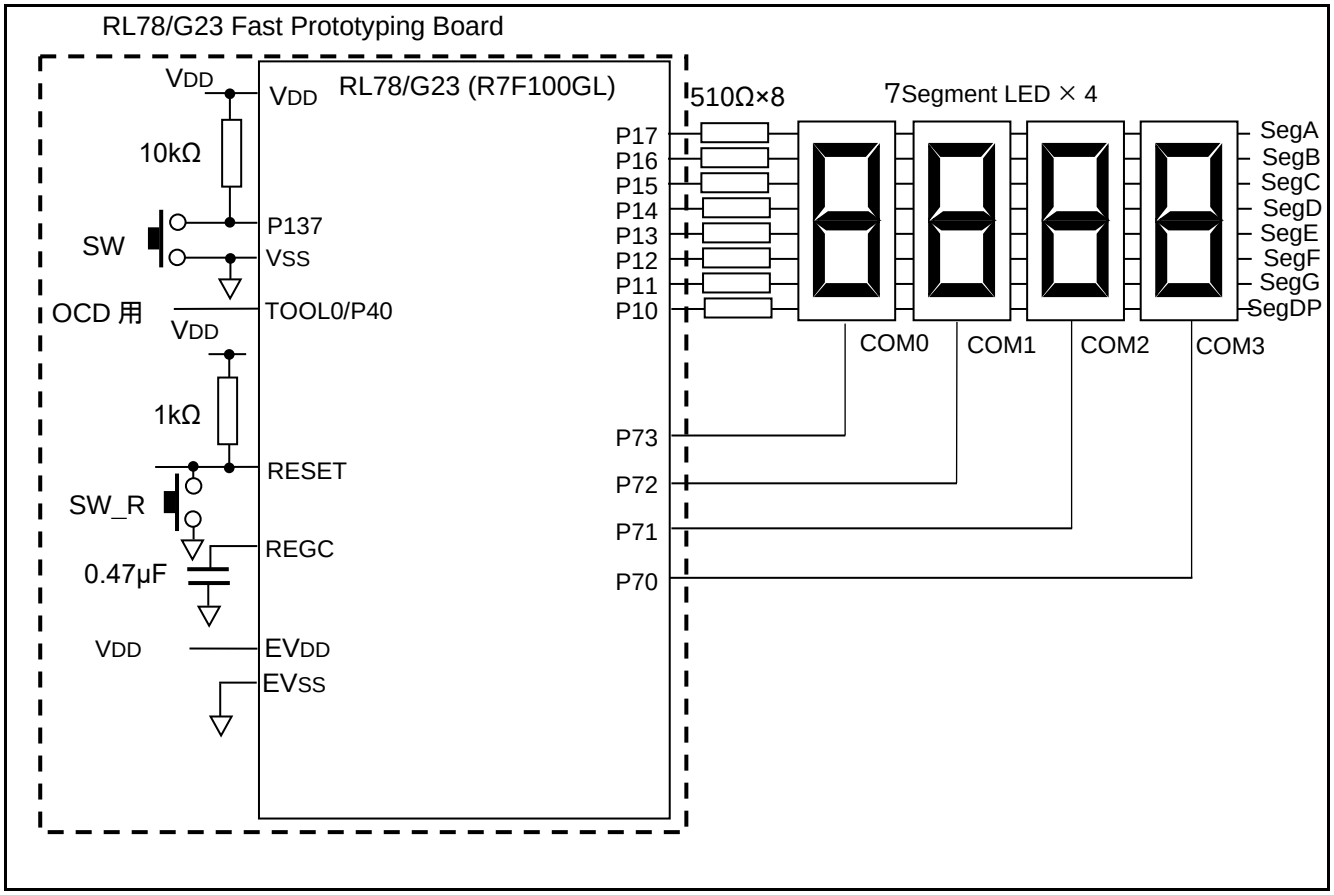
4. ハードウェア説明

4.1 ハードウェア構成例

CPU処理の一部（周辺機能の制御、演算及び判定処理）をSMSに置き換えた事例を紹介します。

図4-1に本アプリケーションノートで使用するハードウェア構成例を示します。

図4-1 ハードウェア構成



注 1. この回路イメージは接続の概要を示す為に簡略化しています。実際に回路を作成される場合は、端子処理などを適切に行い、電気的特性を満たすように設計してください (入力専用ポートは個別に抵抗を介して V_{DD} 又は V_{SS} に接続して下さい) 。

注 2. EV_{SS} で始まる名前の端子がある場合には V_{SS} に、EV_{DD} で始まる名前の端子がある場合には V_{DD} にそれぞれ接続してください。

注 3. V_{DD} はLVD0にて設定したリセット解除電圧 (V_{LVD0}) 以上にしてください。

4.2 使用端子一覧

表 4-1 に使用端子と機能を示します。

表 4-1 使用端子と機能

端子名	入出力	内容
P17	出力	Segment A 制御信号
P16	出力	Segment B 制御信号
P15	出力	Segment C 制御信号
P14	出力	Segment D 制御信号
P13	出力	Segment E 制御信号
P12	出力	Segment F 制御信号
P11	出力	Segment G 制御信号
P10	出力	Segment DP 制御信号
P73	出力	桁 0 (10 分の桁) 選択信号
P72	出力	桁 1 (1 分の桁)選択信号
P71	出力	桁 2 (10 秒の桁)選択信号
P70	出力	桁 3 (1 秒の桁) 選択信号

5. ソフトウェア説明

5.1 動作概要

CPUは、1秒ごとに発生するRTCの定周期割り込みでスタンバイ状態から復帰し、RTCから時刻情報を読み出します。読み出した時刻情報から分と秒の表示データ(セグメントデータ)を作成し、作成したデータをメモリに格納します。

SMSは、2ミリ秒ごとに発生する32ビット・インターバル・タイマのインターバル検出割り込みで　起動され、表示桁のセグメントデータをメモリから読み出し、7セグメントLEDが接続されているポートにデータを出力します。加えて、桁信号を制御することで4桁表示を行います。

表5-1に本サンプルコードの動作概要を示します。

表5-1 動作概要

No.	各機能の動作		
	CPU	SMS	RTC
(1)	SMSの処理コマンドをレジスタに格納	ー	ー
(2)	RTC起動	ー	時間カウント
(3)	SMS起動	起動トリガ待ち状態	↑
(4)	32ビット・インターバル・タイマの起動	↑	↑
(5)	HALT中	TML32のコンペアー致でSMS起動	↑
(6)	↑	TML32のコンペアー致検出フラグ(ITLS0レジスタ)をクリア”	↑
(7)	↑	桁選択信号をオフして、LEDを消灯	↑
(8)	↑	1μs待つ	↑
(9)	↑	次の桁のセグメントデータをメモリから読み出し、P1から出力	↑
(10)	↑	1μs待つ	↑
(11)	↑	桁選択信号をオンして、LEDを点灯	↑
(12)	↑	メモリアドレスを次の桁に更新	↑
(13)	↑	桁選択信号を更新	↑
(14)	↑	桁が終了していなければ(17)へ移動し、終了していれば(15)へ移動する	↑
(15)	↑	メモリアドレスを先頭に移動	↑
(16)	↑	桁データを初期値に戻す	↑
(17)	↑	SMSは待機状態へ移行し(5)へ	↑
(18)	RTCデータ読み出し	ー	定周期割り込み
(19)	セグメントデータを作成	ー	時間カウント
(20)	(5)へ戻りHALT	ー	↑

↑：同上

ー：動作しない

5.2 フォルダ構成

表 5-2にサンプルコードの使用しているソースファイル／ヘッダファイルの構成を示します。なお、統合開発環境で自動生成されるファイル、bsp環境のファイルは除きます。

表 5-2 フォルダ構成

フォルダ、ファイル名	説明	備考
¥r01an6429jj0100_sms_dynamic<DIR>	サンプルコードのフォルダ	
main.c	サンプルコードソースファイル	
¥src<DIR>	プログラム格納用フォルダ	
¥smc_gen<DIR>注 2	スマート・コンフィグレータ生成フォルダ	
¥Config_ITL000_ITL001<DIR>	TML32用プログラム格納フォルダ	
Config_ITL000_ITL001.c	TML32 用ソースファイル	
Config_ITL000_ITL001.h	TML32 用ヘッダファイル	
Config_ITL000_ITL001_user.c	TML32 用割り込みソースファイル	未使用
¥Config_PORT<DIR>	PORT 用プログラム格納フォルダ	
Config_PORT.c	PORT 用ソースファイル	
Config_PORT.h	PORT 用ヘッダファイル	
Config_PORT_user.c	PORT 用ユーザ処理ソースファイル	未使用
¥Config_RTC<DIR>	RTC 用プログラム格納フォルダ	
Config_RTC.c	RTC 用ソースファイル	
Config_RTC.h	RTC 用ヘッダファイル	
Config_RTC_user.c	RTC 用割り込みソースファイル	
¥Config_SMS<DIR>	SMS 用プログラム格納フォルダ	
Config_SMS.c	SMS 用ソースプログラム	
Config_SMS.h	SMS 用ヘッダファイル	
Config_SMS_ASM.smsasm	SMS 用 ASM ソースファイル	
Config_SMS_user.c	SMS 用割り込みソースファイル	未使用
¥general<DIR>	初期化、共通プログラム格納フォルダ	
¥r_bsp<DIR>	BSP プログラム格納フォルダ	
¥r_config<DIR>	BSP_CFG プログラム格納フォルダ	

補足 ”<DIR>” は、ディレクトリを意味します。

注 IAR版のサンプルコードは構成が異なります。詳細はIAR版のサンプルコードを確認してください。
また、r01an6429jj0110_sms_ dynamic.ipcf を格納しています。詳細は、「RL78 スマート・コンフィグレータ ユーザーガイド：IAR 編（R20AN0581）」を確認してください。

5.3 オプション・バイトの設定一覧

表 5-3 にオプション・バイト設定を示します。

表 5-3 オプション・バイト設定

アドレス	設定値	内容
000C0H / 010C0H	11101111B	ウォッチドッグ・タイマ 動作停止 (リセット解除後、カウント停止)
000C1H / 010C1H	11111011B	3.3V (2.82V～5.5V で動作可能) LVD0 検出電圧: リセット・モード 立ち上がり時 TYP. 2.97 V (2.88 V ～ 3.06V) 立ち下がり時 TYP. 2.91V (2.82 V ～ 3.00 V)
000C2H / 010C2H	11101000B	HS モード、高速オンチップ・オシレータ 32MHz
000C3H / 010C3H	10000100B	オンチップ・デバッグ許可

5.4 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	ファイル
SEG_TABLE[]		セグメントデータ用変換テーブル	main.c
INT_READY	0x01	RTC の定周期割り込みが発生した	Config_RTC_user.c
NOT_INT_READY	0x00	RTC の定周期割り込みは発生していない	main.c

5.5 変数一覧

表 5-5 にサンプルコードで使用するグローバル変数を示します。

表 5-5 サンプルコードで使用するグローバル変数

型	変数名	内容	使用関数
st_rtc_counter_value_t	g_RTC_init_data	RTC 初期化データ	main
st_rtc_counter_value_t	g_RTC_read_data	RTC 読み出しデータ	main
uint8_t	g_RTC_INT_flag	1 秒経過フラグ	main, r_Config_RTC_callback_constperiod
uint8_t	g_disp_data[4]	セグメントデータ	main, (SMS)

5.6 関数一覧

表 5-6 にサンプルコードで使用する関数を示します。ただし、スマート・コンフィグレータで生成された関数の内、変更を行っていないものは除きます。

表 5-6 関数一覧

関数名	概要	ソースファイル
main	メイン処理	main.c
R_MAIN_UserInit	使用機能の起動変数初期化	main.c
r_conv_2digit	8 ビットの BCD データを 2 桁分のセグメントデータに変換	main.c
r_conv_SEG	表示セグメントへの変換	main.c
r_Config_RTC_callback_constperiod	RTC の定周期割り込み	Config_RTC_user.c

5.7 関数仕様

サンプルコードの関数仕様を示します。

[関数名] main()

概 要	main 処理
ヘッダ	r_smc_entry.h （r_cg_macrodriver.h, Config_RTC.h,r_cg_userdefine.h 等を含む）
宣 言	void main(void)
説 明	使用する機能を起動後は HALT モードで SMS や RTC からの割り込みを待つ。 1 秒ごとに発生する RTC の定周期割り込みで、RTC から時刻情報を読み出し、セグメントデータに変換する。
引 数	なし
リターン値	なし

[関数名] R_MAIN_UserInit ()

概 要	使用する周辺機能の起動および変数初期化の処理
ヘッダ	r_smc_entry.h （r_cg_macrodriver.h, Config_RTC.h,r_cg_userdefine.h 等を含む）
宣 言	void R_MAIN_UserInit (void)
説 明	Main 関数の最初に呼び出されて、使用する周辺機能を起動する。RTC を起動した後に時間データを書き込む。その後、SMS を起動し、トリガ用のタイマも起動する。
引 数	なし
リターン値	なし

[関数名] r_conv_2digit()		
概　要	第 1 引数の BCD データを 2 桁のセグメントデータに変換する	
ヘッダ	r_smc_entry.h （r_cg_macrodriver.h, Config_RTC.h,r_cg_userdefine.h 等を含む）	
宣　言	void r_conv_2digit(uint8_t data,uint8_t digit)	
説　明	第 1 引数の BCD データを 2 桁のセグメントデータに変換して、セグメントデータの格納バッファ (g_disp_data) の第 2 引数で示された桁に格納する。	
引　数	uint8_t data	BCD データ
	uint8_t digit	変換したセグメントデータを格納するバッファのポインタ
リターン値	なし	

[関数名] r_conv_SEG()		
概　要	引数の下位 4 ビットの BCD をセグメントデータに変換する	
ヘッダ	r_smc_entry.h （r_cg_macrodriver.h, Config_RTC.h,r_cg_userdefine.h 等を含む）	
宣　言	uint8_t r_conv_SEG(uint8_t data)	
説　明	引数の BCD データの下位 4 ビットを対応するセグメントデータに変換する。	
引　数	uint8_t data	BCD データ
リターン値	uint8_t	セグメントデータ値

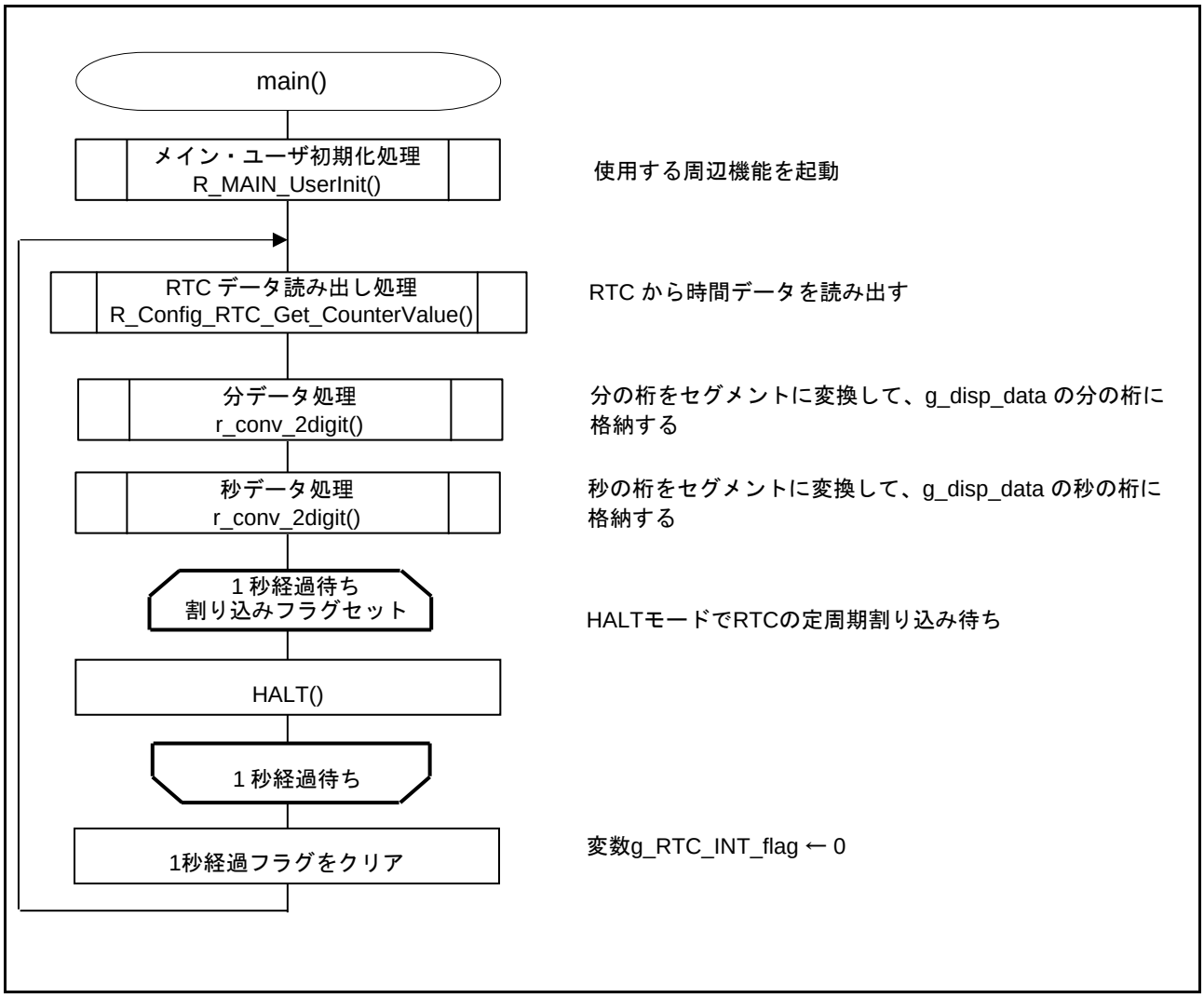
[関数名] r_Config_RTC_callback_constperiod()	
概　要	スタート・コンディションを生成
ヘッダ	r_cg_macrodriver.h, r_cg_userdefine.h, Config_RTC.h
宣　言	static void r_Config_RTC_callback_constperiod(void)
説　明	RTC の定周期割り込みでフラグ (g_RTC_INT_flag) をセットする
引　数	なし
リターン値	なし

5.8 フローチャート

5.8.1 メイン処理

図 5-1 にメイン処理のフローチャートを示します。

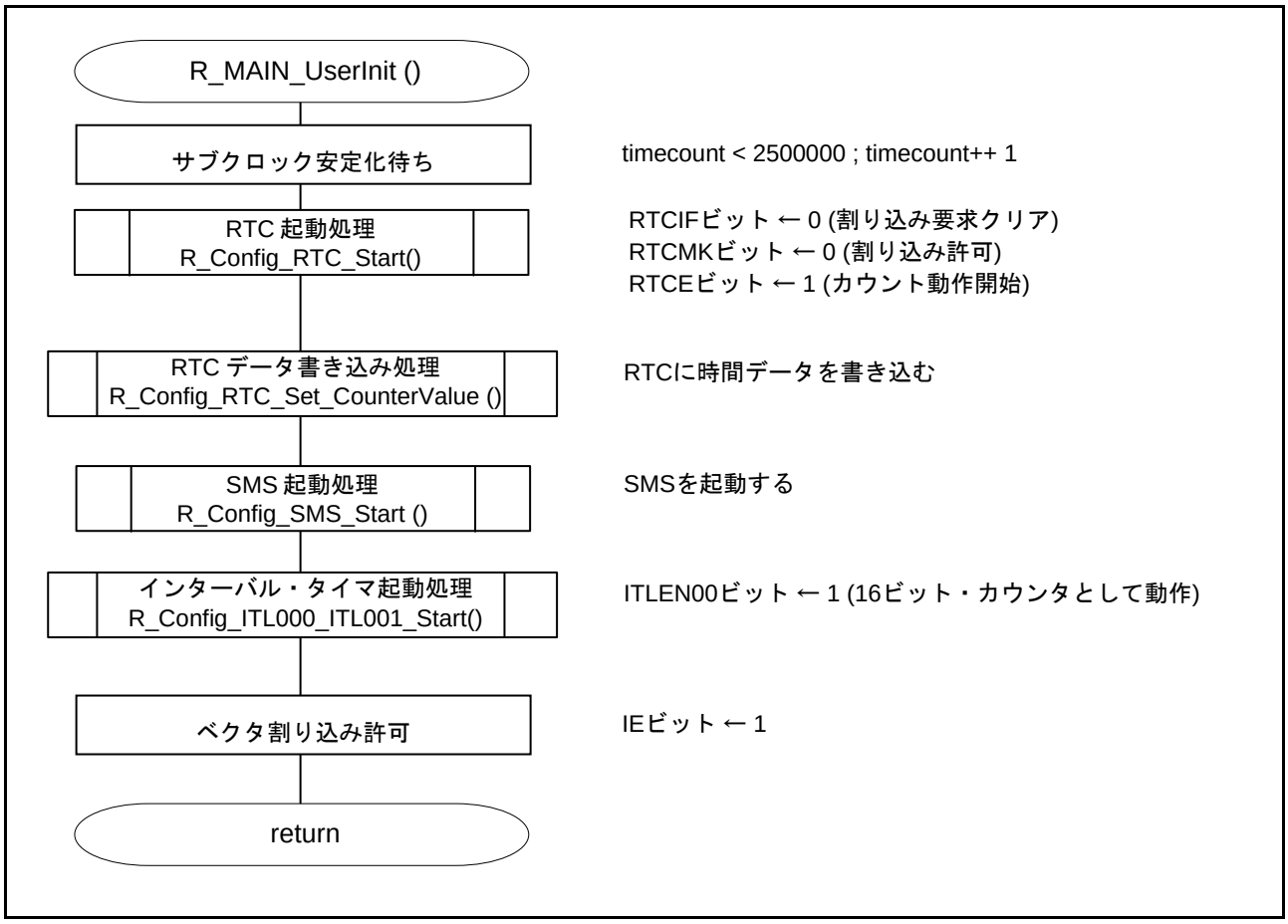
図 5-1 メイン処理



5.8.2 メイン・ユーザ初期化処理

図 5-2 にメイン・ユーザ初期化処理のフローチャートを示します。

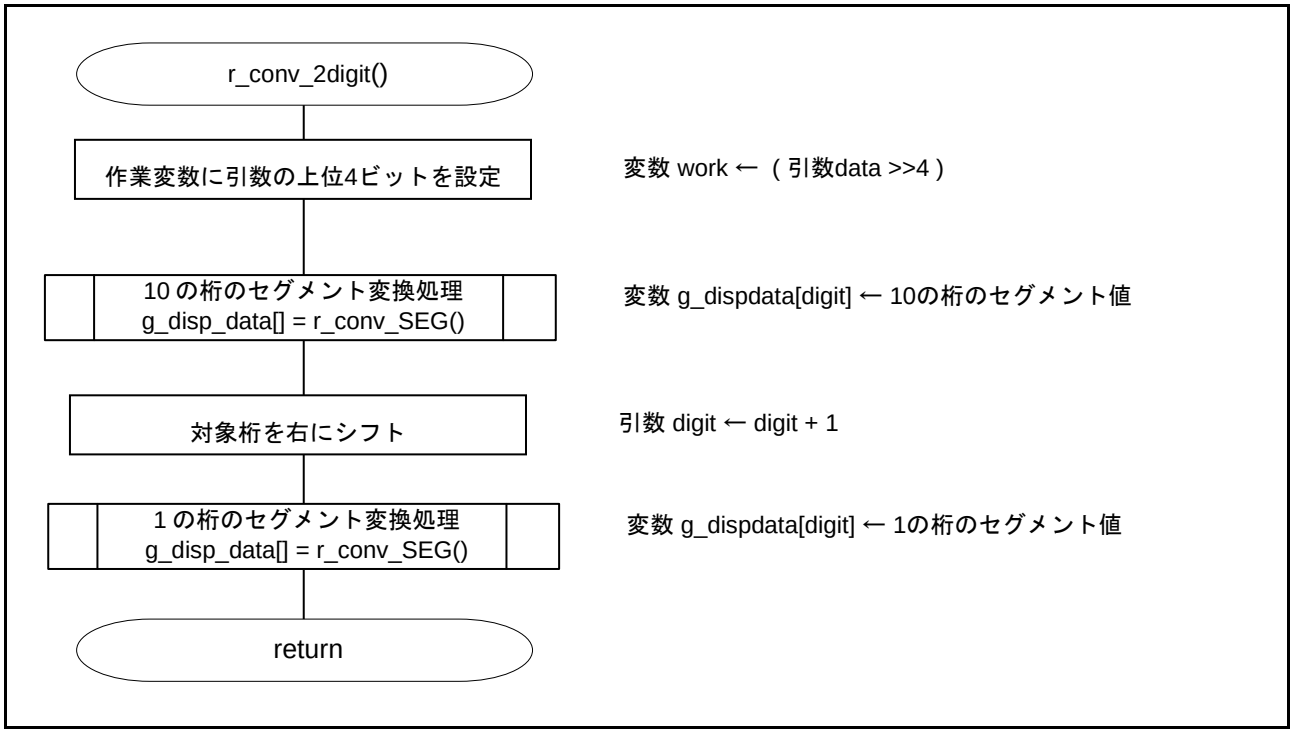
図 5-2 メイン・ユーザ初期化処理



5.8.3 2桁分セグメント変換処理

図 5-5 に 2 桁分セグメント変換処理のフローチャートを示します。

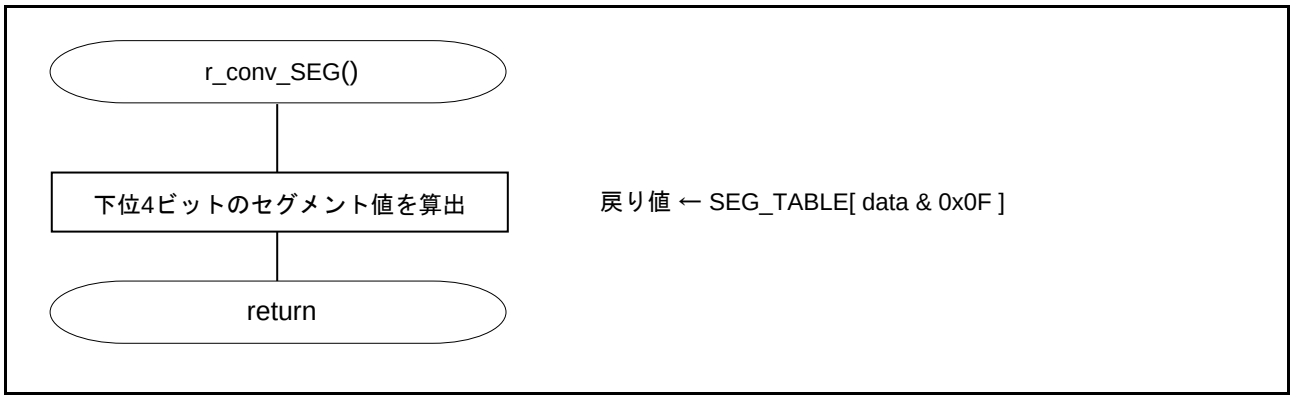
図 5-3 2 桁分セグメント変換処理



5.8.4 セグメント変換処理

図 5-4 にセグメント変換処理のフローチャートを示します。

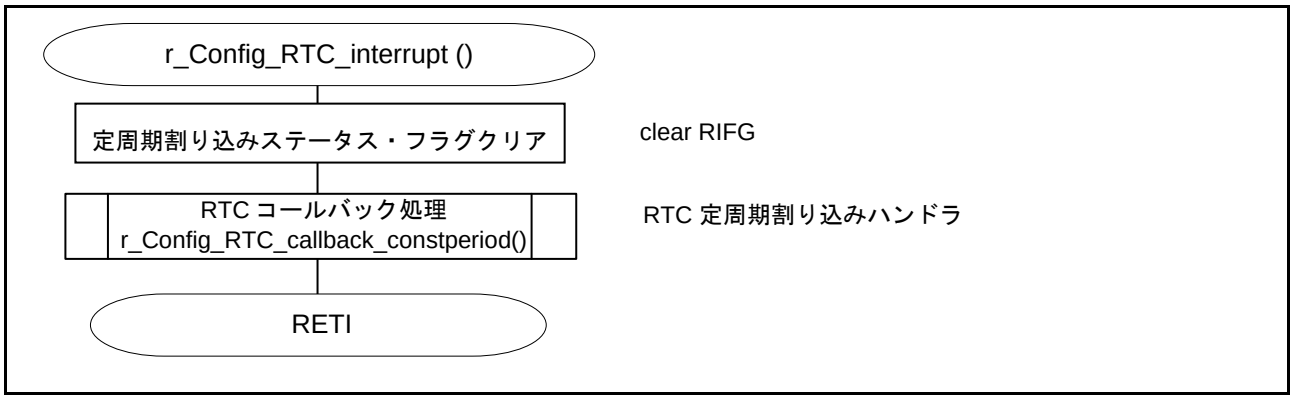
図 5-4 セグメント変換処理



5.8.5 RTC1 秒割り込み処理

図 5-5 に RTC1 秒割り込み処理のフローチャートを示します。

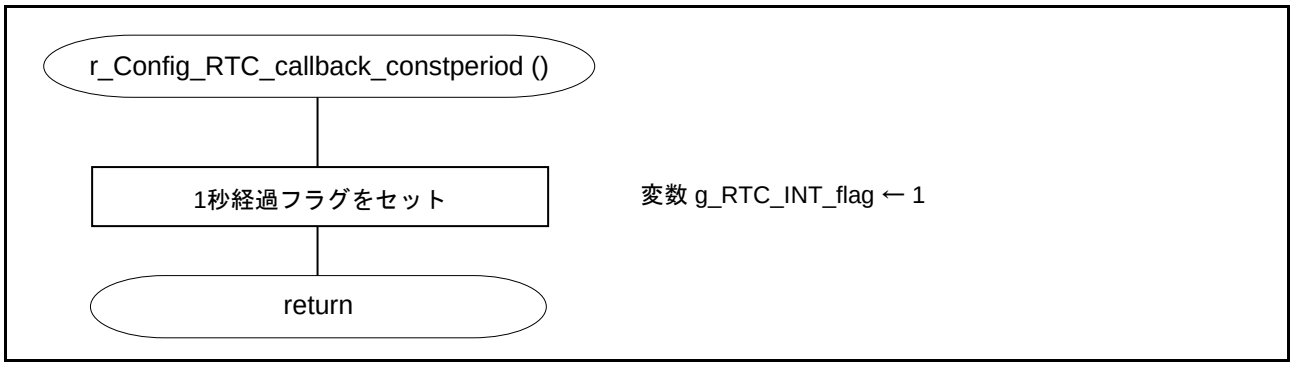
図 5-5 RTC1 秒割り込み処理



5.8.6 RTC コールバック処理

図 5-6 に RTC コールバック処理のフローチャートを示します。

図 5-6 RTC コールバック処理



5.9 SNOOZEモード・シーケンサの設定

起動トリガに設定したイベントが発生するとSMSはシーケンサ・インストラクション・レジスタ(SMSI0-31)に格納された処理コマンドを順次実行します。処理コマンドの実行では、シーケンサ汎用レジスタ（SMSG0-15）をソース・アドレスやデスティネーション・アドレス、演算データの格納などに使用します。

SMSI0-31とSMSG0-15は、SMS用プログラム(.SMSASMファイル)をアセンブリ言語で記述することで設定します。SMS用プログラムは、スマート・コンフィグレータのSNOOZE Mode Sequencerコンポーネントを使い処理ブロックを組み合わせることで作成することも可能です。作成したSMS用プログラムはSMS用アセンブラでC言語ファイルへ変換されプログラムに組み込まれます。

サンプルコードで実行するSMS処理の仕様を以下に示します。

概要	SMS処理
説明	TML32のインターバル検出割り込み(INTITL)によりSMSが起動し、main関数が設定したメモリの内容をP1とP7を制御して、7セグメントLEDを1桁ずつドライブすることで4桁の数値を表示します。
引数[注1]	DATAPOINTE, pointer
リターン値	なし
備考	なし

注1 R_Config_SMS_Start関数設定で指定する引数です。詳細は、6.2.1を参照してください。

図5-7にSMS処理のフローチャートを示します。
表5-7～表5-8にSNOOZEモード・シーケンサを制御するレジスタの設定値を示します。

図 5-7 SMS 処理

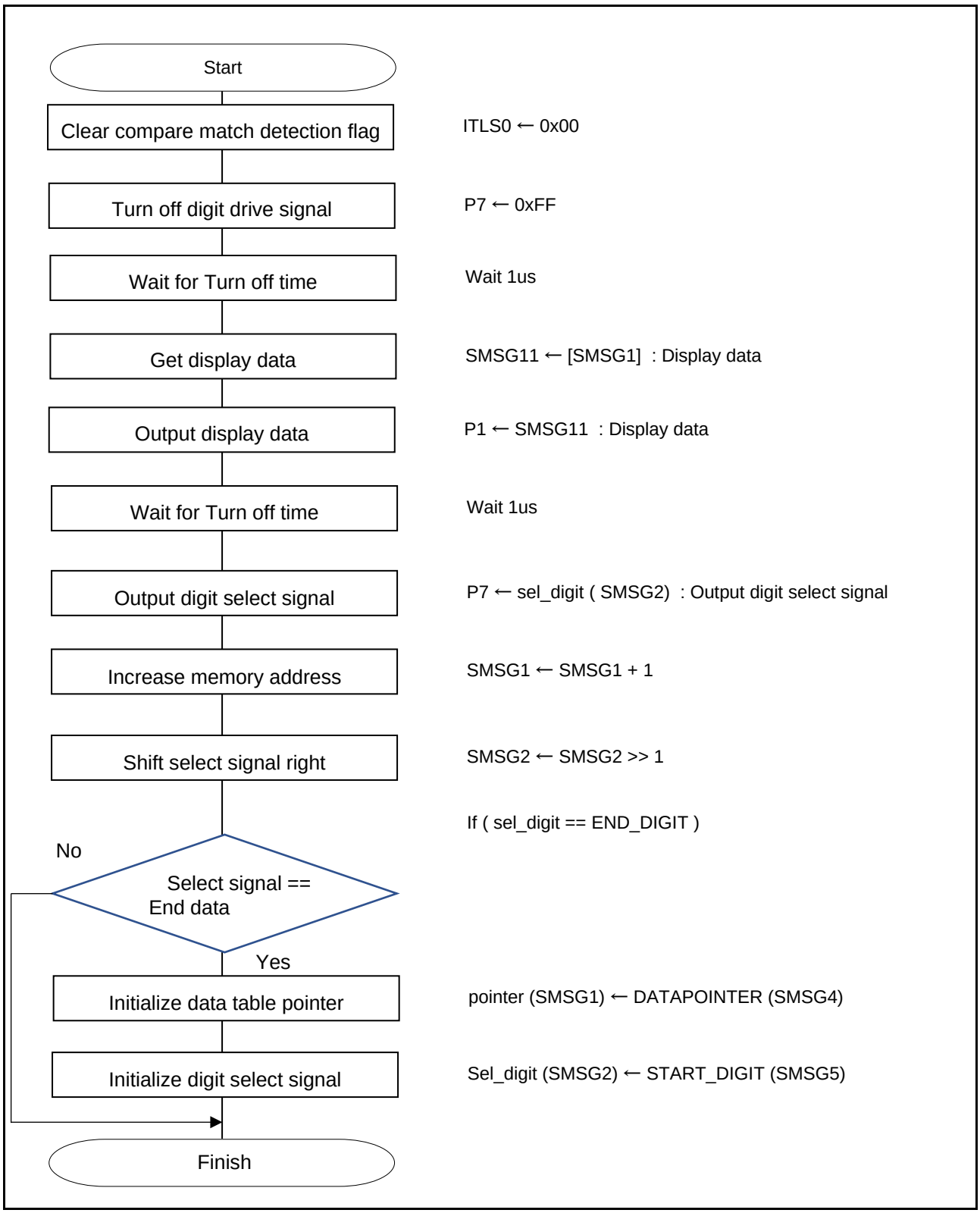

Start
Clear compare match detection flag
ITLS0 ← 0x00
Turn off digit drive signal
P7 ← 0xFF
Wait for Turn off time
Wait 1us
Get display data
SMSG11 ← [SMSG1] : Display data
Output display data
P1 ← SMSG11 : Display data
Wait for Turn off time
Wait 1us
Output digit select signal
P7 ← sel_digit (SMSG2) : Output digit select signal
Increase memory address
SMSG1 ← SMSG1 + 1
Shift select signal right
SMSG2 ← SMSG2 >> 1
If (sel_digit == END_DIGIT)
No
Select signal == End data
Yes
Initialize data table pointer
pointer (SMSG1) ← DATAPOINTER (SMSG4)
Initialize digit select signal
Sel_digit (SMSG2) ← START_DIGIT (SMSG5)
Finish

表 5-7 シーケンサ汎用レジスタ 0-15

レジスタ呼称	設定値	備考
SMSG0	0x0000	固定値：0000H
SMSG1	0x0000	表示データ用のポインタ
SMSG2	0xFFF7	桁選択データ
SMSG3	0x0FFF	最終桁用の比較データ
SMSG4	0x0000	表示データ用のポインタの初期値
SMSG5	0xFFF7	桁選択データの初期値
SMSG6	0x036B	ITLS0 レジスタのアドレス
SMSG7	0x03C1	SMSG1 のアドレス
SMSG8	0xFF01	P1 レジスタのアドレス
SMSG9	0x00FF	桁選択クリア用データ
SMSG10	0x0001	表示データの更新幅 (1 バイト)
SMSG11	0x0000	未使用
SMSG12	0x0000	未使用
SMSG13	0x0000	未使用
SMSG14	0x0000	未使用
SMSG15	0xFFFF	固定値：FFFFH

表 5-8 シーケンサ・インストラクション・レジスタ 0-31

レジスタ呼称	設定値	備考
SMSI0	0x0600	MOV [SMSG6+0], SMSG0
SMSI1	0x0896	MOV [SMSG8+6], SMSG9
SMSI2	0x9200	WAIT 32, 0
SMSI3	0x11B0	MOV SMSG11, [SMSG1+0]
SMSI4	0x08B0	MOV [SMSG8+0], SMSG11
SMSI5	0x9200	WAIT 32, 0
SMSI6	0x0826	MOV [SMSG8+6], SMSG2
SMSI7	0x71A0	ADDW SMSG1, SMSG10
SMSI8	0x7203	SHRW SMSG2
SMSI9	0x7232	CMPW SMSG2, SMSG3
SMSI10	0x8033	BNZ $3
SMSI11	0x2740	MOVW [SMSG7+0], SMSG4
SMSI12	0x0752	MOV [SMSG7+2], SMSG5
SMSI13	0xF000	FINISH
SMSI14-31	0x0000	未使用

6. 応用例

本アプリケーションノートは、サンプルコードの他に以下のスマート・コンフィグレータの設定ファイルが格納されています。

r01an6429_sms_dynamic.scfg

r01an6429_sms_dynamic.sms

次項以降でファイルの説明と使用する上での設定例および注意事項を示します。

6.1 r01an6429_sms_dynamic.scfg

サンプルコードで使用しているスマート・コンフィグレータの設定ファイルです。スマート・コンフィグレータで設定されている全ての機能が含まれています。サンプルコードの設定は以下の通りです。

表 6-1 スマート・コンフィグレータの設定値

タブ名	コンポーネント	内容
クロック	ー	動作モード：高速メインモード 2.7 (V) ~5.5 (V) EVDD 設定：2.7V≦EVDD0＜5.5V 高速オンチップ・オシレータ：32MHz fIHP：32MHz fCLK：32000kHz (高速オンチップ・オシレータ) fSXP：32.768kHz (サブシステム・クロック)
システム	ー	オンチップ・デバッグ動作設定：エミュレータを使う 疑似 RRM/DMM 機能設定：使用しない Start/Stop 関数機能設定：使用しない トレース機能設定：使用しない セキュリティ ID 設定：セキュリティ ID を設定する セキュリティ ID：0x00000000000000000000 セキュリティ ID 認証失敗時の設定：フラシュ・メモリのデータを消去する
コンポーネント	r_bsp	Start up select : Enable (use BSP startup) Control of invalid memory access detection : Disable RAM guard space (GRAM0-1) : Disabled Guard of control registers of port function (GPORT) : Disabled Guard of registers of interrupt function (GINT) : Disabled Guard of control registers of clock control function, voltage detector, and RAM parity error detection function (GCSC) : Disabled Data flash access control (DFLEN) : Disables Initialization of peripheral functions by Code Generator/Smart Configurator : Enable API functions disable : Enable Parameter check enable : Enable Setting for starting the high-speed on-chip oscillator at the times of release from STOP mode and of transitions to SNOOZE mode : High-speed Enable user warm start callback (PRE) : Unused Enable user warm start callback (POST) : Unused Watchdog Timer refresh enable : Unused
	Config_LVD0	動作モード設定：リセット・モード 電圧検出設定：リセット発生電圧 (VLVD0)：2.91 (V)

表 6-2 スマート・コンフィグレータの設定値

タブ名	コンポーネント	内容
コンポーネント	Config_ITL000 _ITL001	コンポーネント：インターバル・タイマ 動作モード：16 ビット・カウンタ・モード リソース：ITL000_ITL001 動作クロック：fIHP クロックソース：fITL0 インターバル時間：2ms 割り込み設定：使用しない
	Config_RTC	コンポーネント：リアルタイム・クロック クロックソース：サブシステム・クロック XR (fSXR) 時間制の選択：12 時間制 リアルタイム・クロック初期設定値：2022/04/01　12:00:00 RTC1HZ 端子の出力 (1Hz) 許可：使用しない アラーム検出機能：使用しない 割り込み設定： 　定周期割り込み機能 (INTRTC) ：1 秒に 1 度 　優先順位：レベル 3 (低優先順位)
	Config_SMS	コンポーネント：SNOOZE モード・シーケンサ 起動トリガ：インターバル検出割り込み (INTITL)
	Config_PORT	ポート選択：PORT1,PORT7 ポート・モード設定：Pmn レジスタ値を読み出す PORT1：全て出力 PORT7：P70～P73 を出力、1 を出力

6.1.1 クロック

サンプルコードで使用するクロックの設定を行います。

サンプルコードでは、fCLK に 32000KHz を、7 セグメント LED をマトリクス構成で使用しているので、動作モードを「高速メインモード 2.7 (V) ~5.5 (V)」に設定しています。設定を変更する際は注意してください。

6.1.2 システム

サンプルコードのオンチップ・デバッグ設定を行います。

「オンチップ・デバッグ動作設定」、「セキュリティ ID 認証失敗時の設定」は、「表 5-3 オプション・バイト設定」の「オンチップ・デバッグ動作許可」に影響を与えます。設定を変更する際は注意してください。

6.1.3 r_bsp

サンプルコードのスタートアップの設定を行います。

6.1.4 Config_LVD0

サンプルコードの電源管理の設定を行います。

「表 5-3 オプション・バイト設定」の「LVD0 の設定」に影響を与えます。設定を変更する際は注意してください。

6.1.5 Config__IT000__ITL001

サンプルコードのインターバル・タイマの初期設定を行います。

サンプルコードのSMSの起動にインターバル検出割り込み(INTITL)を使用します。そのため、「割り込み設定」を「使用しない」に設定しています。「割り込み設定」を「使用する」に変更することも可能です。R_Config_SMS_Start関数でINTITLはマスクされますので、HALTモード中にINTITLが発生してもCPUが起動することはありません。HALTモードから復帰後は、INTITLはマスクされた状態ですので、必要なときはINTITLのマスクを解除してください。

6.1.6 Config_RTC

サンプルコードのRTCの設定を行います。

サンプルコードでは、1秒の定周期でのINTRTC割り込みと時間カウント機能を使用して、分と秒の情報を得るために使用します。INTRTC割り込みはCPUのHALTモードを解除するために使用します。

6.1.7 Config_SMS

サンプルコードのSMSの設定を行います。

詳細は、「6.2 r01an6429_sms_dynamic.sms」を参照してください。

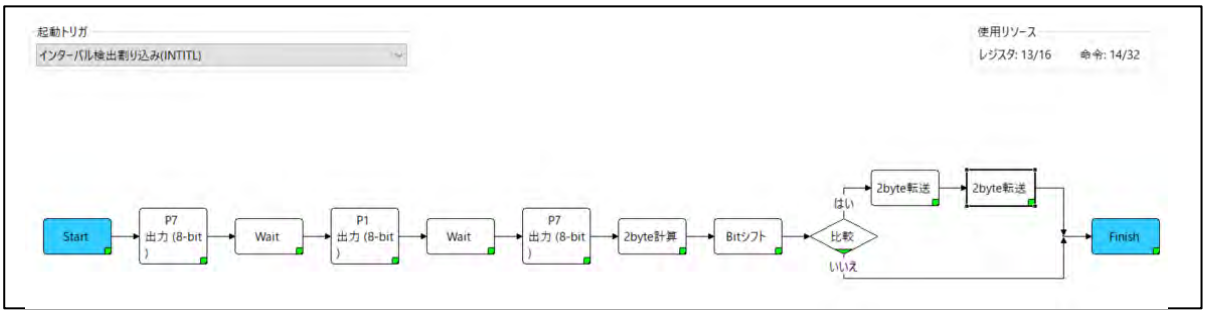
6.2 r01an6429_sms_dynamic.sms

Config_SMS単体のデータです。サンプルコードでは、SMSの起動にインターバル検出割り込み(INTITL)を使用し、CPUの起動と処理でRTCおよびポートを使用します。別途インターバル・タイマとRTCおよびポートの設定を行う必要がありますので注意してください。

また、r01an6429_sms_dynamic.smsは、別プロジェクトのスマート・コンフィグレータにインポートすることが可能です。別プロジェクトにSMSコンポーネントを設定後、［インポート］→［ブラウズ］で「r01an6429_sms_dynamic.sms」を選択するとインポートできます。

スマート・コンフィグレータにインポートすると図6-1のようなフロー図となります。このフロー図は、「図5-7 SMS処理」と同じです。

図6-1 Config_SMSのフロー図



各ブロックの説明を以下に示します。

6.2.1 Start

SMS起動時にSMS開始関数 (R_Config_SMS_Start関数) で引数として渡された値がPOINTER (表示データテーブルの読み出しアドレス) 、DATAPOINTER (データテーブル開始アドレス) に設定されます。

図 6-2 Start設定

6.2.2 P7出力 (8-bit)

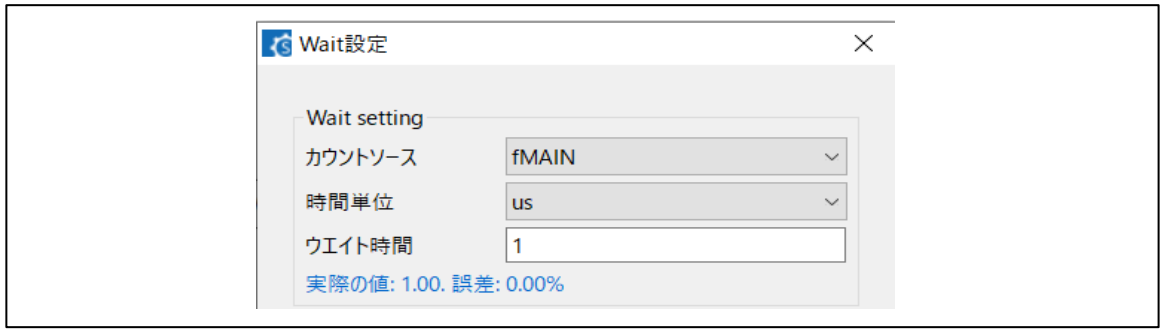
7セグメントLEDの表示を消すために、P7 (桁選択信号) に0xFFを出力して、コモン信号のドライブを中止します。

図 6-3 P7出力 (8-bit)設定

6.2.3 コモン信号のWait設定

桁選択信号の安定時間として1μs待ちます。外部にドライバICを追加する場合は、ドライバICの遅延時間を考慮して待ち時間を長くしてください。

図 6-4 コモン信号のウェイト設定



6.2.4 P1 出力 (8-bit)

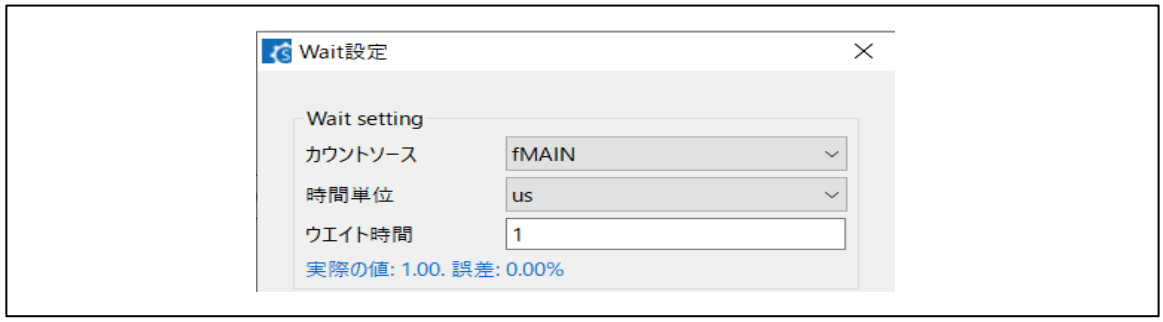
表示するデータに対応したセグメントデータを P1 に出力します。

図 6-5 P1 出力 (8-bit)設定

6.2.5 セグメント信号の Wait 設定

セグメント信号の安定時間として 1μs 待ちます。外部にドライバ IC を追加する場合は、ドライバ IC の遅延時間を考慮して待ち時間を長くしてください。

図 6-6 セグメント信号のウェイト設定



6.2.6 P7 出力 (8-bit)

表示する桁に対応するコモン信号をオンします。

図 6-7 P7 出力 (8-bit)設定

6.2.7 2byte 計算

表示するデータを指すポインタ (pointer) を次のデータに更新します。

図 6-8 2byte 計算設定

6.2.8 Bit シフト

次の桁の表示に備えて、桁選択信号 (sel_digit) を右に 1 ビット分シフトします。

図 6-9 Bit シフト設定

6.2.9 比較設定

桁選択信号 (sel_digit) が終了条件 (END_DIGIT) に達したかを比較します。

図 6-10 比較設定

6.2.10 ポインタの初期化用 2byte 転送

セグメントデータの読み出しポインタ (pointer) の値を初期値 (DATAPOINTER) に設定します。

図 6-11 ポインタの初期化用 2byte 転送

6.2.11 桁選択信号の初期化用 2byte 転送

次の桁を選択する値 (sel_digit) を初期値 (STAR_DIGIT) に戻します。

図 6-12 桁選択信号の初期化用 2byte 転送

6.2.12 Finish

HALTモードに移行します。サンプルコードでは、戻り値を使用しません。

図 6-13 Finish 設定

6.2.13 変数の設定

SMSで使用している変数の設定を以下に示します。

表 6-3 SMSで使用している変数

データ名	初期化
pointer	SMSの開始関数で引数渡し
sel_digit	SMSの開始関数で0xFFF7に初期化

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。

また、SCでコードを再生成した場合は、R_Config_SMS_Start関数内を下記赤枠の様に変更してください。

```
void R_Config_SMS_Start(uint16_t DATAPOINTER, uint16_t pointer)
{
    /* Set the sms data from arguments */
    SMSG4 = DATAPOINTER;
    SMSG1 = pointer;
    /* Initialize SMS data */
    SMSG2 = 65527U;
    SMSG5 = 65527U;
    SMSG3 = 4095U;
    /* Disable related interrupts */
    ITLMK = 1U;
    /* Start sms */
    //SMSEIF = 0U; /* clear INTSMSE interrupt flag */
    //SMSEMK = 0U; /* enable INTSMSE interrupt */
    SMSEMK = 1U; /* disable INTSMSE interrupt */
    SMSEIF = 1U; /* set INTSMSE interrupt flag */
    g_sms_wakeup_flag = 0U;
    ITLS0 = _00_INTITL_CLEAR;
    SMSSTART = 1U;
}
```

8. 参考ドキュメント

RL78/G23 ユーザーズマニュアル ハードウェア編 (R01UH0896J)
RL78 ファミリ ユーザーズマニュアル ソフトウェア編 (R01US0015J)
（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。）

ホームページとサポート窓口

ルネサス エレクトロニクスホームページ
http://japan.renesas.com

お問合せ先
http://japan.renesas.com/contact/

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2022.7.11	—	初版発行
1.10	2024.1.9	—	SMS 処理のフロー変更

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。