

RE01 1500KBグループ

R01AN4770JJ0130

Rev.1.30

R_SYSTEMドライバ詳細仕様書

2021.05.12

要旨

本書では、RE01 1500KB グループ CMSIS Driver Package のシステムドライバ R_SYSTEM の詳細仕様を説明します。

動作確認デバイス

RE01 1500KB グループ

目次

1. 概要	3
2. ソフトウェアコンポーネントの内部構成	4
2.1 ファイル構成	4
3. ソフトウェアコンポーネントの内部動作	7
4. ソフトウェアユニット詳細情報	8
4.1 コンフィグレーション	8
4.1.1 パラメタチェック	8
4.1.2 クリティカルセクション	8
4.1.3 レジスタプロテクション	8
4.1.4 APIのタイムアウト値	9
4.1.5 イベントリンク番号設定	9
4.1.6 関数のRAM配置	13
4.2 マクロ／型定義	14
4.3 関数仕様	16
4.3.1 R_SYS_CodeCopy関数	16
4.3.2 R_SYS_Initialize関数	17
4.3.3 R_SYS_BoostSpeedModeSet関数	18
4.3.4 R_SYS_HighSpeedModeSet関数	19
4.3.5 R_SYS_LowSpeedModeSet関数	22
4.3.6 R_SYS_32kHzSpeedModeSet関数	24
4.3.7 R_SYS_SpeedModeGet関数	25
4.3.8 R_SYS_SystemClockHOCOSet関数	26
4.3.9 R_SYS_SystemClockMOCOSet関数	29
4.3.10 R_SYS_SystemClockLOCOSet関数	30
4.3.11 R_SYS_SystemClockMOSCSet関数	31
4.3.12 R_SYS_SystemClockSOSCSet関数	32
4.3.13 R_SYS_SystemClockPLLSet関数	33
4.3.14 R_SYS_SystemClockFreqGet関数	36
4.3.15 R_SYS_PeripheralClockFreqGet関数	38
4.3.16 R_SYS_SystemClockDividerSet関数	40

4.3.17 R_SYS_MainOscSpeedClockStart関数	43
4.3.18 R_SYS_MainOscSpeedClockStop関数	44
4.3.19 R_SYS_HighSpeedClockStart関数	45
4.3.20 R_SYS_HighSpeedClockStop関数	47
4.3.21 R_SYS_MediumSpeedClockStart関数	48
4.3.22 R_SYS_MediumSpeedClockStop関数	49
4.3.23 R_SYS_LowSpeedClockStart関数	50
4.3.24 R_SYS_LowSpeedClockStop関数	51
4.3.25 R_SYS_SubOscSpeedClockStart関数	52
4.3.26 R_SYS_SubOscSpeedClockStop関数	53
4.3.27 R_SYS_PLLSpeedClockStart関数	54
4.3.28 R_SYS_PLLSpeedClockStop関数	56
4.3.29 R_SYS_OscStabilizationFlagGet関数	57
4.3.30 R_SYS_IrqEventLinkSet関数	58
4.3.31 R_SYS_IrqStatusGet関数	59
4.3.32 R_SYS_IrqStatusClear関数	60
4.3.33 R_SYS_EnterCriticalSection関数	61
4.3.34 R_SYS_ExitCriticalSection関数	61
4.3.35 R_SYS_ResourceLock関数	62
4.3.36 R_SYS_ResourceUnlock関数	62
4.3.37 R_SYS_RegisterProtectEnable関数	63
4.3.38 R_SYS_RegisterProtectDisable関数	63
4.3.39 R_SYS_SoftwareDelay関数	64
4.3.40 R_SYS_GetVersion関数	66
4.3.41 r_sys_BoostFlagGet関数	67
4.3.42 r_sys_BoostFlagSet関数	68
4.3.43 r_sys_BoostFlagClr関数	69
4.3.44 r_system_wdt_refresh関数	69
4.3.45 IELn_IRQHandler関数(n=0~31)	70
4.3.46 R_NVIC_EnableIRQ関数	71
4.3.47 R_NVIC_GetEnableIRQ関数	72
4.3.48 R_NVIC_DisableIRQ関数	73
4.3.49 R_NVIC_GetPendingIRQ関数	74
4.3.50 R_NVIC_SetPendingIRQ関数	75
4.3.51 R_NVIC_ClearPendingIRQ関数	76
4.3.52 R_NVIC_SetPriority関数	77
4.3.53 R_NVIC_GetPriority関数	78
4.3.54 R_NVIC_SetVector関数	78
4.3.55 R_NVIC_GetVector関数	79
4.3.56 R_NVIC_SystemReset関数	79

1. 概要

本書における略語一覧と関連文書一覧を示します。

表 1-1 略語一覧

名称	略語
RENESAS-DRIVER R_SYSTEM	R_SYSTEM ドライバ
RENESAS CMSIS-Core	R_CORE
RE01 グループ ユーザーズマニュアル ハードウェア編	UMH

表 1-2 関連文書一覧

文書名	文書番号
RE01 グループ(1.5M バイトフラッシュメモリ搭載製品) ユーザーズマニュアル ハードウェア編	R01UH0796
RE01 1500KB,256KB グループ CMSIS パッケージを用いた開発スタートアップガイド	R01AN4660

表 1-3 ROM/RAM サイズ一覧

ROM/RAM 名称	キャッシュタイプ	サイズ
ProgramROM	ROM/Flash	1.5 Mbytes
ROM	ROM/Flash	256 bytes
OptionSettingMEM	ROM/Flash	32 bytes
MemoryMirror	ROM/Flash	8 Mbytes
RAM	RAM	256 kbytes

表 1-4 最大スタックサイズ

最大スタックサイズ	0x400 (1 kbytes)
-----------	------------------

2. ソフトウェアコンポーネントの内部構成

2.1 ファイル構成

R_SYSTEM ドライバはCMSIS Driver Package の Device HAL に該当し、ベンダ独自ファイル格納ディレクトリ内の r_system_api.c、r_system_api.h、r_system_cfg.h の 3 個のファイルで構成されます。各ファイルの役割を表 2-1に示します。RE01 1500KB グループ CMSIS Driver Package における R_SYSTEM ドライバのファイル構成を図 2.1に示します。R_SYSTEM ドライバ関数は各機能に対し、図 2.2で示す関数で構成されます。

表 2-1 R_SYSTEM ドライバ 各ファイルの役割

ファイル名	内容
r_system_api.c	ドライバソースファイルです。 ドライバ関数の実体を用意します。 R_SYSTEM ドライバを使用する場合は、本ファイルをビルドする必要があります。
r_system_api.h	ドライバヘッダファイルです。 ユーザが参照可能なマクロ／型／プロトタイプ宣言を用意します。 R_SYSTEM ドライバを使用する場合は、本ファイルをインクルードする必要があります。
r_system_cfg.h	コンフィグレーション定義ファイルです。 ユーザが設定可能なコンフィグレーション定義を用意します。

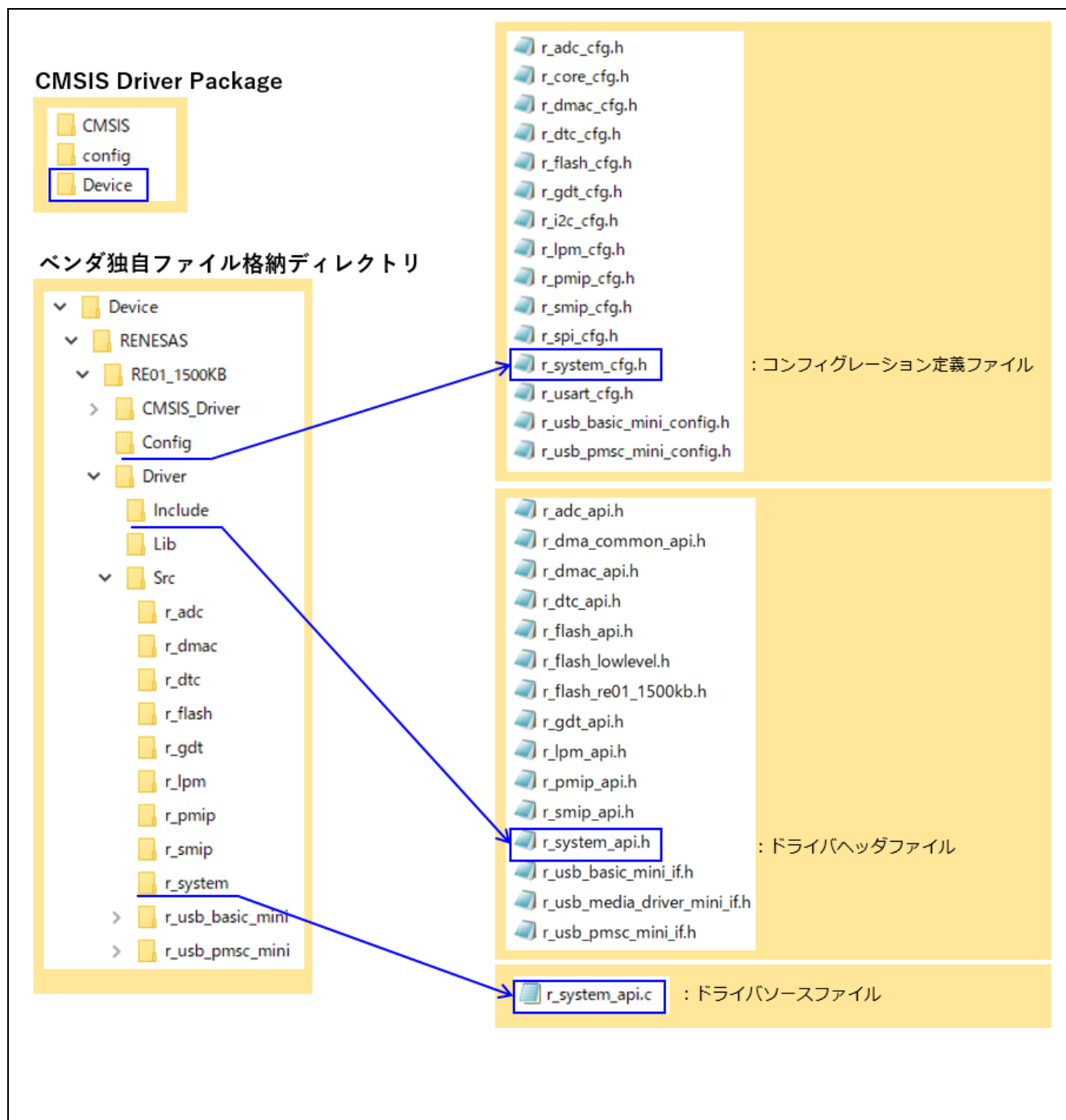


図 2.1 CMSIS Driver Package における R_SYSTEM ドライバファイル構成

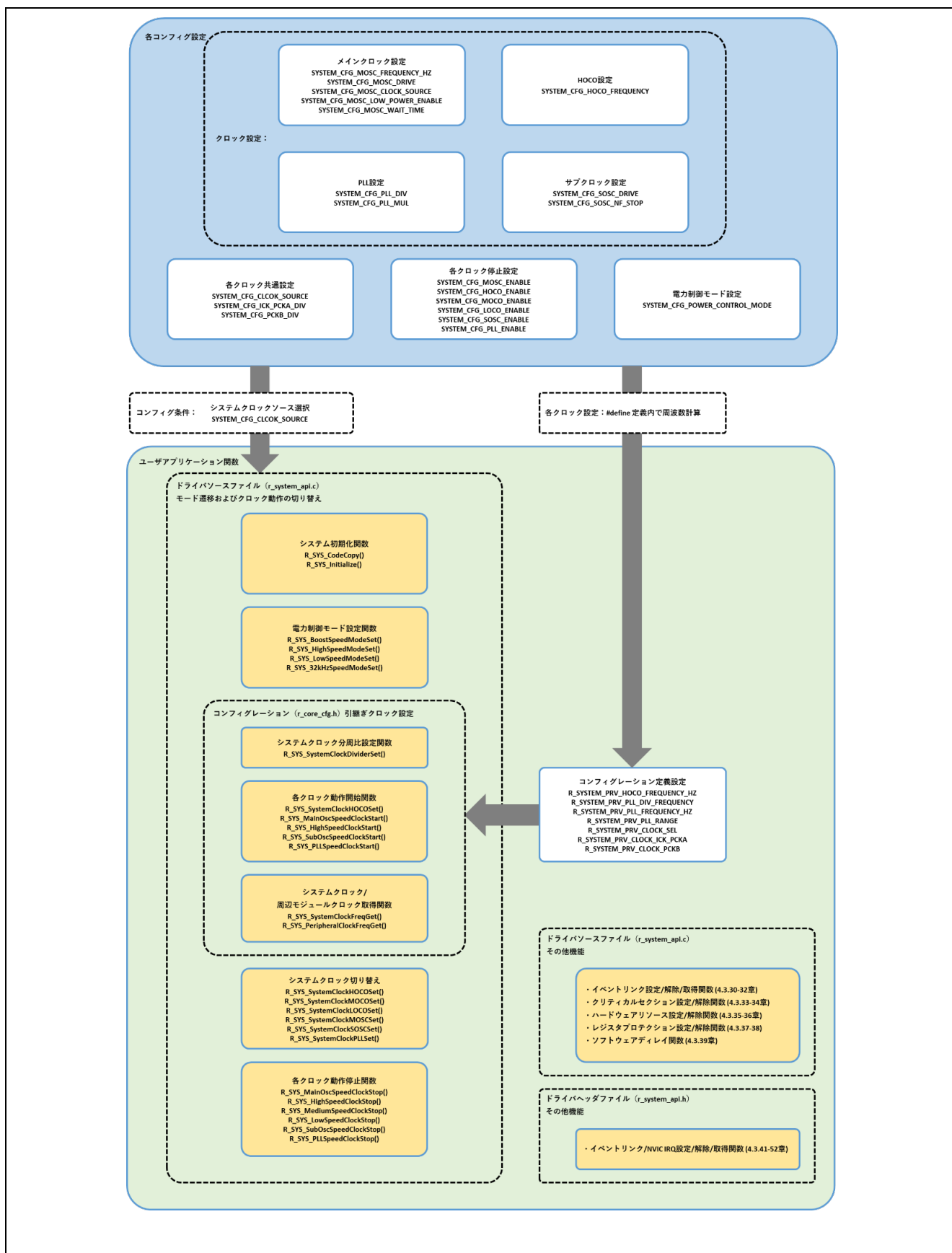


図 2.2 R_CORE のコンフィグレーション設定と R_SYSTEM ドライバ関数の関係

3. ソフトウェアコンポーネントの内部動作

R_SYSTEM ドライバは、モード遷移およびクロック動作の切り替えを実現します。本章では、モード遷移およびクロック動作切り替えを行う R_SYSTEM ドライバ関数呼び出し手順を示します。電源供給モードおよび VBB モード移行手順については、R_LPM のドライバ仕様書を参照ください。

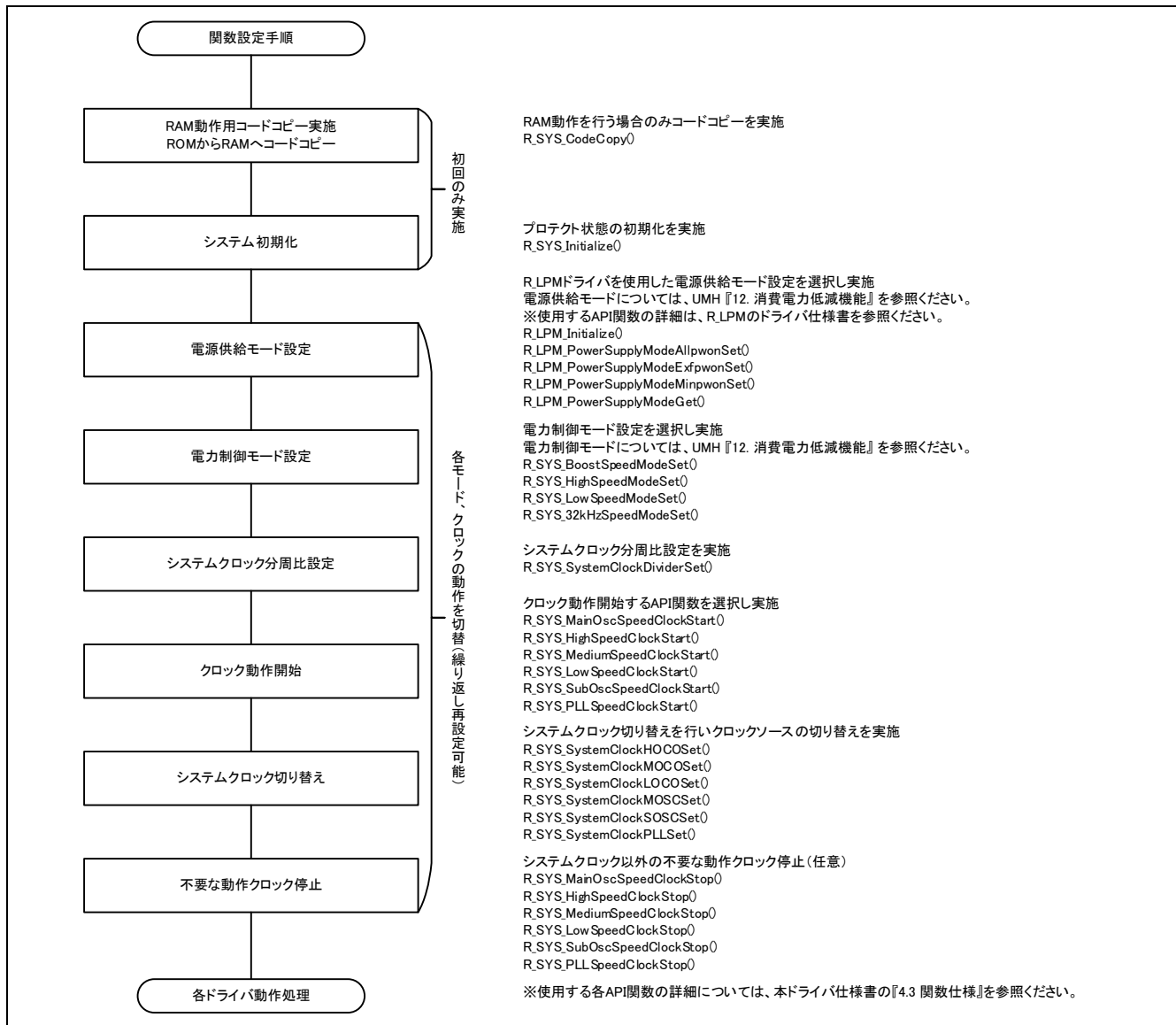


図 3.1 R_SYSTEM ドライバを使用した API 関数設定手順

4. ソフトウェアユニット詳細情報

4.1 コンフィグレーション

R_SYSTEM ドライバは、ユーザが設定可能なコンフィグレーションを `r_system_cfg.h` ファイルに用意します。

4.1.1 パラメタチェック

R_SYSTEM ドライバにおけるパラメタチェックの有効/無効を設定します。

名称：SYSTEM_CFG_PARAM_CHECKING_ENABLE

表 4-1 SYSTEM_CFG_PARAM_CHECKING_ENABLE の設定

設定値	内容
0	パラメタチェックを無効にします。 関数仕様に記載しているエラー条件の検出を行いません。
1 (初期値)	パラメタチェックを有効にします。 関数仕様に記載しているエラー条件の検出を行います。

4.1.2 クリティカルセクション

R_SYSTEM ドライバにおけるクリティカルセクション制御の有効/無効を設定します。

レジスタの一部ビットを変更するために、レジスタ値を読み出し後、一部変更して再設定を行うような場合は、途中で割り込みが入らないようにクリティカルセクションの制御をおこなう必要があります。

名称：SYSTEM_CFG_ENTER_CRITICAL_SECTION_ENABLE

表 4-2 SYSTEM_CFG_ENTER_CRITICAL_SECTION_ENABLE の設定

設定値	内容
0	クリティカルセクションの制御を無効にします。
1 (初期値)	クリティカルセクションの制御を有効にします。

4.1.3 レジスタプロテクション

R_SYSTEM ドライバにおけるレジスタライトプロテクション制御の有効/無効を設定します。

対象レジスタに書込みを行う場合は、レジスタライトプロテクションの制御を行う必要があります。

名称：SYSTEM_CFG_REGISTER_PROTECTION_ENABLE

表 4-3 SYSTEM_CFG_REGISTER_PROTECTION_ENABLE の設定

設定値	内容
0	レジスタライトプロテクションの制御を無効にします。
1 (初期値)	レジスタライトプロテクションの制御を有効にします。

4.1.4 APIのタイムアウト値

CMSIS ドライバAPI が値の反映待ちをする際のタイムアウト時間を設定します。

名称：SYSTEM_CFG_API_TIMEOUT_COUNT

表 4-4 SYSTEM_CFG_API_TIMEOUT_COUNT の設定

設定値	内容
268,435,456 (0x10000000)	CMSIS ドライバAPI が値の反映待ちをする際のタイムアウト時間を設定します。

4.1.5 イベントリンク番号設定

指定したイベントリンク番号の割り込みハンドラをコールバック関数として呼び出します。

本設定のリンクするイベント信号については、UMH を参照してください。

名称：SYSTEM_CFG_EVENT_NUMBER_****_****

表 4-5 SYSTEM_CFG_EVENT_NUMBER_****_**** の設定値

設定値	内容
0x00 (初期値) SYSTEM_IRQ_EVENT_NUMBER_NOT_USED	該当する周辺モジュールへのイベント出力は無効
0x01-0xAB SYSTEM_IRQ_EVENT_NUMBERn (n=0-31)	リンクするイベント信号の番号を指定

表 4-6 SYSTEM_CFG_EVENT_NUMBER_****_**** のイベント番号設定

イベント番号	割り込み要求の発生元	名称	イベント番号のコンフィグ設定(r_system_cfg.h)
01h	ポート	PORT_IRQ0	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ0
02h		PORT_IRQ1	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ1
03h		PORT_IRQ2	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ2
04h		PORT_IRQ3	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ3
05h		PORT_IRQ4	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ4
06h		PORT_IRQ5	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ5
07h		PORT_IRQ6	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ6
08h		PORT_IRQ7	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ7
09h	DMAC0	DMAC0_INT	SYSTEM_CFG_EVENT_NUMBER_DMAMC0_INT
0Ah	DMAC1	DMAC1_INT	SYSTEM_CFG_EVENT_NUMBER_DMAMC1_INT
0Bh	DMAC2	DMAC2_INT	SYSTEM_CFG_EVENT_NUMBER_DMAMC2_INT
0Ch	DMAC3	DMAC3_INT	SYSTEM_CFG_EVENT_NUMBER_DMAMC3_INT
0Dh	DTC	DTC_COMPLETE	SYSTEM_CFG_EVENT_NUMBER_DTC_COMPLETE
0Fh	ICU	ICU_SNZCANCEL	SYSTEM_CFG_EVENT_NUMBER_ICU_SNZCANCEL
10h	FCU	FCU_FIFERR	SYSTEM_CFG_EVENT_NUMBER_FCU_FIFERR
11h		FCU_FRDYI	SYSTEM_CFG_EVENT_NUMBER_FCU_FRDYI
12h	LVD	LVD_LVD1	SYSTEM_CFG_EVENT_NUMBER_LVD_LVD1
13h		LVD_LVDBAT	SYSTEM_CFG_EVENT_NUMBER_LVD_LVDBAT
14h	MOSC	MOSC_STOP	SYSTEM_CFG_EVENT_NUMBER_MOSC_STOP
15h	低消費電力モード	SYSTEM_SNZREQ	SYSTEM_CFG_EVENT_NUMBER_SYSTEM_SNZREQ
16h	EHC	SOL_DH	SYSTEM_CFG_EVENT_NUMBER_SOL_DH

17h		SOL_DL	SYSTEM_CFG_EVENT_NUMBER_SOL_DL
18h	AGT0	AGT0_AGTI	SYSTEM_CFG_EVENT_NUMBER_AGT0_AGTI
1Ah		AGT0_AGTCMBI	SYSTEM_CFG_EVENT_NUMBER_AGT0_AGTCMBI
1Bh	AGT1	AGT1_AGTI	SYSTEM_CFG_EVENT_NUMBER_AGT1_AGTI
1Ch		AGT1_AGTCMAI	SYSTEM_CFG_EVENT_NUMBER_AGT1_AGTCMAI
1Dh	AGT0	AGT0_AGTCMAI	SYSTEM_CFG_EVENT_NUMBER_AGT0_AGTCMAI
1Eh	IWDT	IWDT_NMIUNDF	SYSTEM_CFG_EVENT_NUMBER_IWDT_NMIUNDF
1Fh	WDT	WDT_NMIUNDF	SYSTEM_CFG_EVENT_NUMBER_WDT_NMIUNDF
20h	RTC	RTC_ALM	SYSTEM_CFG_EVENT_NUMBER_RTC_ALM
21h		RTC_PRD	SYSTEM_CFG_EVENT_NUMBER_RTC_PRD
22h		RTC_CUP	SYSTEM_CFG_EVENT_NUMBER_RTC_CUP
23h	S14AD	ADC140_ADI	SYSTEM_CFG_EVENT_NUMBER_ADC140_ADI
24h		ADC140_GBADI	SYSTEM_CFG_EVENT_NUMBER_ADC140_GBADI
25h		ADC140_CMPAI	SYSTEM_CFG_EVENT_NUMBER_ADC140_CMPAI
26h		ADC140_CMPBI	SYSTEM_CFG_EVENT_NUMBER_ADC140_CMPBI
27h		ADC140_WCMPPM	SYSTEM_CFG_EVENT_NUMBER_ADC140_WCMPPM
28h		ADC140_WCMPUM	SYSTEM_CFG_EVENT_NUMBER_ADC140_WCMPUM
29h		ADC140_GCADI	SYSTEM_CFG_EVENT_NUMBER_ADC140_GCADI
2Ah	ACMP	ACMP_CMPI	SYSTEM_CFG_EVENT_NUMBER_ACMP_CMPI
2Bh	USB	USBFS_D0FIFO	SYSTEM_CFG_EVENT_NUMBER_USBFS_D0FIFO
2Ch		USBFS_D1FIFO	SYSTEM_CFG_EVENT_NUMBER_USBFS_D1FIFO
2Dh		USBFS_USBI	SYSTEM_CFG_EVENT_NUMBER_USBFS_USBI
2Eh		USBFS_USBR	SYSTEM_CFG_EVENT_NUMBER_USBFS_USBR
2Fh	RIIC0	IIC0_RXI	SYSTEM_CFG_EVENT_NUMBER_IIC0_RXI
30h		IIC0_TXI	SYSTEM_CFG_EVENT_NUMBER_IIC0_TXI
31h		IIC0_TEI	SYSTEM_CFG_EVENT_NUMBER_IIC0_TEI
32h		IIC0_EEI	SYSTEM_CFG_EVENT_NUMBER_IIC0_EEI
33h	RIIC1	IIC1_RXI	SYSTEM_CFG_EVENT_NUMBER_IIC1_RXI
34h		IIC1_TXI	SYSTEM_CFG_EVENT_NUMBER_IIC1_TXI
35h		IIC1_TEI	SYSTEM_CFG_EVENT_NUMBER_IIC1_TEI
36h		IIC1_EEI	SYSTEM_CFG_EVENT_NUMBER_IIC1_EEI
37h	KINT	KEY_INTKR	SYSTEM_CFG_EVENT_NUMBER_KEY_INTKR
38h	DOC	DOC_DOPCI	SYSTEM_CFG_EVENT_NUMBER_DOC_DOPCI
39h	CAC	CAC_FEERI	SYSTEM_CFG_EVENT_NUMBER_CAC_FEERI
3Ah		CAC_MENDI	SYSTEM_CFG_EVENT_NUMBER_CAC_MENDI
3Bh		CAC_OVFI	SYSTEM_CFG_EVENT_NUMBER_CAC_OVFI
3Ch	I/Oポート	IOPORT_GROUP3	SYSTEM_CFG_EVENT_NUMBER_IOPORT_GROUP3
3Dh		IOPORT_GROUP2	SYSTEM_CFG_EVENT_NUMBER_IOPORT_GROUP2
3Eh	ELC	ELC_SWEVT0	SYSTEM_CFG_EVENT_NUMBER_ELC_SWEVT0
3Fh		ELC_SWEVT1	SYSTEM_CFG_EVENT_NUMBER_ELC_SWEVT1
40h	POE	POEG_GROUPA	SYSTEM_CFG_EVENT_NUMBER_POEG_GROUPA
41h		POEG_GROUPB	SYSTEM_CFG_EVENT_NUMBER_POEG_GROUPB
42h	TMR	TMR_CMIA0	SYSTEM_CFG_EVENT_NUMBER_TMR_CMIA0
43h		TMR_CMIB0	SYSTEM_CFG_EVENT_NUMBER_TMR_CMIB0
44h		TMR_OVF0	SYSTEM_CFG_EVENT_NUMBER_TMR_OVF0
45h		TMR_CMIA1	SYSTEM_CFG_EVENT_NUMBER_TMR_CMIA1
46h		TMR_CMIB1	SYSTEM_CFG_EVENT_NUMBER_TMR_CMIB1
47h		TMR_OVF1	SYSTEM_CFG_EVENT_NUMBER_TMR_OVF1
48h	CCC	CCC_PRD	SYSTEM_CFG_EVENT_NUMBER_CCC_PRD
49h		CCC_CUP	SYSTEM_CFG_EVENT_NUMBER_CCC_CUP
4Ah	LPG	CCC_ERR	SYSTEM_CFG_EVENT_NUMBER_CCC_ERR
4Bh	MTDV	MTDV_PM1INT	SYSTEM_CFG_EVENT_NUMBER_MTDV_PM1INT

4Ch		MTDV_PM25INT	SYSTEM_CFG_EVENT_NUMBER_MTDV_PM25INT
4Dh		MTDV_PM36INT	SYSTEM_CFG_EVENT_NUMBER_MTDV_PM36INT
4Eh	ELC	ELC_INT0	SYSTEM_CFG_EVENT_NUMBER_ELC_INT0
4Fh		ELC_INT1	SYSTEM_CFG_EVENT_NUMBER_ELC_INT1
50h	GPT320	GPT0_CCMPA	SYSTEM_CFG_EVENT_NUMBER_GPT0_CCMPA
51h		GPT0_CCMPB	SYSTEM_CFG_EVENT_NUMBER_GPT0_CCMPB
52h		GPT0_CMPC	SYSTEM_CFG_EVENT_NUMBER_GPT0_CMPC
53h		GPT0_CMPD	SYSTEM_CFG_EVENT_NUMBER_GPT0_CMPD
54h		GPT0_OVF	SYSTEM_CFG_EVENT_NUMBER_GPT0_OVF
55h		GPT0_UDF	SYSTEM_CFG_EVENT_NUMBER_GPT0_UDF
56h	GPT321	GPT1_CCMPA	SYSTEM_CFG_EVENT_NUMBER_GPT1_CCMPA
57h		GPT1_CCMPB	SYSTEM_CFG_EVENT_NUMBER_GPT1_CCMPB
58h		GPT1_CMPC	SYSTEM_CFG_EVENT_NUMBER_GPT1_CMPC
59h		GPT1_CMPD	SYSTEM_CFG_EVENT_NUMBER_GPT1_CMPD
5Ah		GPT1_OVF	SYSTEM_CFG_EVENT_NUMBER_GPT1_OVF
5Bh		GPT1_UDF	SYSTEM_CFG_EVENT_NUMBER_GPT1_UDF
5Ch	GPT162	GPT2_CCMPA	SYSTEM_CFG_EVENT_NUMBER_GPT2_CCMPA
5Dh		GPT2_CCMPB	SYSTEM_CFG_EVENT_NUMBER_GPT2_CCMPB
5Eh		GPT2_CMPC	SYSTEM_CFG_EVENT_NUMBER_GPT2_CMPC
5Fh		GPT2_CMPD	SYSTEM_CFG_EVENT_NUMBER_GPT2_CMPD
60h		GPT2_OVF	SYSTEM_CFG_EVENT_NUMBER_GPT2_OVF
61h		GPT2_UDF	SYSTEM_CFG_EVENT_NUMBER_GPT2_UDF
62h	GPT163	GPT3_CCMPA	SYSTEM_CFG_EVENT_NUMBER_GPT3_CCMPA
63h		GPT3_CCMPB	SYSTEM_CFG_EVENT_NUMBER_GPT3_CCMPB
64h		GPT3_CMPC	SYSTEM_CFG_EVENT_NUMBER_GPT3_CMPC
65h		GPT3_CMPD	SYSTEM_CFG_EVENT_NUMBER_GPT3_CMPD
66h		GPT3_OVF	SYSTEM_CFG_EVENT_NUMBER_GPT3_OVF
67h		GPT3_UDF	SYSTEM_CFG_EVENT_NUMBER_GPT3_UDF
68h	GPT164	GPT4_CCMPA	SYSTEM_CFG_EVENT_NUMBER_GPT4_CCMPA
69h		GPT4_CCMPB	SYSTEM_CFG_EVENT_NUMBER_GPT4_CCMPB
6Ah		GPT4_CMPC	SYSTEM_CFG_EVENT_NUMBER_GPT4_CMPC
6Bh		GPT4_CMPD	SYSTEM_CFG_EVENT_NUMBER_GPT4_CMPD
6Ch		GPT4_OVF	SYSTEM_CFG_EVENT_NUMBER_GPT4_OVF
6Dh		GPT4_UDF	SYSTEM_CFG_EVENT_NUMBER_GPT4_UDF
6Eh	GPT165	GPT5_CCMPA	SYSTEM_CFG_EVENT_NUMBER_GPT5_CCMPA
6Fh		GPT5_CCMPB	SYSTEM_CFG_EVENT_NUMBER_GPT5_CCMPB
70h		GPT5_CMPC	SYSTEM_CFG_EVENT_NUMBER_GPT5_CMPC
71h		GPT5_CMPD	SYSTEM_CFG_EVENT_NUMBER_GPT5_CMPD
72h		GPT5_OVF	SYSTEM_CFG_EVENT_NUMBER_GPT5_OVF
73h		GPT5_UDF	SYSTEM_CFG_EVENT_NUMBER_GPT5_UDF
74h	GPT	GPT_UVWEDGE	SYSTEM_CFG_EVENT_NUMBER_GPT_UVWEDGE
75h	SCI0	SCI0_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI0_RXI
76h		SCI0_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI0_TXI
77h		SCI0_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI0_TEI
78h		SCI0_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI0_ERI
79h		SCI0_AM	SYSTEM_CFG_EVENT_NUMBER_SCI0_AM
7Ah		SCI0_RXI_OR_ERI	未使用
7Bh	SCI1	SCI1_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI1_RXI
7Ch		SCI1_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI1_TXI
7Dh		SCI1_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI1_TEI
7Eh		SCI1_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI1_ERI
7Fh		SCI1_AM	SYSTEM_CFG_EVENT_NUMBER_SCI1_AM

80h	SCI2	SCI2_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI2_RXI
81h		SCI2_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI2_TXI
82h		SCI2_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI2_TEI
83h		SCI2_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI2_ERI
84h		SCI2_AM	SYSTEM_CFG_EVENT_NUMBER_SCI2_AM
85h	SCI3	SCI3_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI3_RXI
86h		SCI3_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI3_TXI
87h		SCI3_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI3_TEI
88h		SCI3_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI3_ERI
89h		SCI3_AM	SYSTEM_CFG_EVENT_NUMBER_SCI3_AM
8Ah	SCI4	SCI4_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI4_RXI
8Bh		SCI4_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI4_TXI
8Ch		SCI4_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI4_TEI
8Dh		SCI4_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI4_ERI
8Eh		SCI4_AM	SYSTEM_CFG_EVENT_NUMBER_SCI4_AM
8Fh	SCI5	SCI5_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI5_RXI
90h		SCI5_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI5_TXI
91h		SCI5_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI5_TEI
92h		SCI5_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI5_ERI
93h		SCI5_AM	SYSTEM_CFG_EVENT_NUMBER_SCI5_AM
94h	SCI9	SCI9_RXI	SYSTEM_CFG_EVENT_NUMBER_SCI9_RXI
95h		SCI9_TXI	SYSTEM_CFG_EVENT_NUMBER_SCI9_TXI
96h		SCI9_TEI	SYSTEM_CFG_EVENT_NUMBER_SCI9_TEI
97h		SCI9_ERI	SYSTEM_CFG_EVENT_NUMBER_SCI9_ERI
98h		SCI9_AM	SYSTEM_CFG_EVENT_NUMBER_SCI9_AM
99h	SPI0	SPI0_SPRI	SYSTEM_CFG_EVENT_NUMBER_SPI0_SPRI
9Ah		SPI0_SPTI	SYSTEM_CFG_EVENT_NUMBER_SPI0_SPTI
9Bh		SPI0_SPII	SYSTEM_CFG_EVENT_NUMBER_SPI0_SPII
9Ch		SPI0_SPEI	SYSTEM_CFG_EVENT_NUMBER_SPI0_SPEI
9Dh		SPI0_SPTEND	SYSTEM_CFG_EVENT_NUMBER_SPI0_SPTEND
9Eh	SPI1	SPI1_SPRI	SYSTEM_CFG_EVENT_NUMBER_SPI1_SPRI
9Fh		SPI1_SPTI	SYSTEM_CFG_EVENT_NUMBER_SPI1_SPTI
A0h		SPI1_SPII	SYSTEM_CFG_EVENT_NUMBER_SPI1_SPII
A1h		SPI1_SPEI	SYSTEM_CFG_EVENT_NUMBER_SPI1_SPEI
A2h		SPI1_SPTEND	SYSTEM_CFG_EVENT_NUMBER_SPI1_SPTEND
A3h	QSPI	QSPI_INTR	SYSTEM_CFG_EVENT_NUMBER_QSPI_INTR
A4h	DIV	DIV_CALCCOMP	SYSTEM_CFG_EVENT_NUMBER_DIV_CALCCOMP
A6h	MLCD	MLCD_TEI	SYSTEM_CFG_EVENT_NUMBER_MLCD_TEI
A7h		MLCD_TEMI	SYSTEM_CFG_EVENT_NUMBER_MLCD_TEMI
A8h	GDT	GDT_DATII	SYSTEM_CFG_EVENT_NUMBER_GDT_DATOI
A9h		GDT_DATOI	SYSTEM_CFG_EVENT_NUMBER_GDT_FDCENDI
AAh		GDT_FDCENDI	SYSTEM_CFG_EVENT_NUMBER_GDT_DATII
B4h	ポート	PORT_IRQ8	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ8
B5h		PORT_IRQ9	SYSTEM_CFG_EVENT_NUMBER_PORT_IRQ9

4.1.6 関数の RAM 配置

R_SYSTEM ドライバの特定関数を RAM で実行するための設定を行います。

Flash を遮断中に実行するプログラムは、RAM に配置し RAM で実行する必要があります。

関数の RAM 配置を設定するコンフィグレーションは、関数ごとに定義を持ちます。

名称：SYSTEM_CFG_SECTION_R_SYS_XXXXX

SYSTEM_CFG_SECTION_IELn_IRQHANDLER (n = 0～31)

XXXXX には API 名をすべて大文字で記載

例) R_SYS_Initialize 関数 → SYSTEM_CFG_SECTION_R_SYS_INITIALIZE

表 4-7 SYSTEM_CFG_SECTION_XXXXX の設定

設定値	内容
SYSTEM_SECTION_CODE	関数を RAM に配置しません
SYSTEM_SECTION_RAM_FUNC	関数を RAM に配置します

表 4-8 各関数の RAM 配置初期状態

番号	関数名	RAM 配置
1	R_SYS_Initialize	
2	R_SYS_BoostSpeedModeSet	
3	R_SYS_HighSpeedModeSet	✓
4	R_SYS_LowSpeedModeSet	✓
5	R_SYS_32kHzSpeedModeSet	✓
6	R_SYS_SpeedModeGet	✓
7	R_SYS_SystemClockHOCOSet	✓
8	R_SYS_SystemClockMOCOSet	✓
9	R_SYS_SystemClockLOCOSet	✓
10	R_SYS_SystemClockMOSCSet	✓
11	R_SYS_SystemClockSOSCSet	✓
12	R_SYS_SystemClockPLLSet	
13	R_SYS_SystemClockFreqGet	✓
14	R_SYS_PeripheralClockFreqGet	✓
15	R_SYS_SystemClockDividerSet	✓
16	R_SYS_MainOscSpeedClockStart	✓
17	R_SYS_MainOscSpeedClockStop	✓
18	R_SYS_HighSpeedClockStart	✓
19	R_SYS_HighSpeedClockStop	✓
20	R_SYS_MediumSpeedClockStart	✓
21	R_SYS_MediumSpeedClockStop	✓
22	R_SYS_LowSpeedClockStart	✓
23	R_SYS_LowSpeedClockStop	✓
24	R_SYS_SubOscSpeedClockStart	✓
25	R_SYS_SubOscSpeedClockStop	✓
26	R_SYS_PLLSpeedClockStart	
27	R_SYS_PLLSpeedClockStop	
28	R_SYS_OscStabilizationFlagGet	✓

29	R_SYS_IrqEventLinkSet	✓
30	R_SYS_IrqStatusGet	✓
31	R_SYS_IrqStatusClear	✓
32	R_SYS_EnterCriticalSection	✓
33	R_SYS_ExitCriticalSection	✓
34	R_SYS_ResourceLock	✓
35	R_SYS_ResourceUnlock	✓
36	R_SYS_RegisterProtectEnable	✓
37	R_SYS_RegisterProtectDisable	✓
38	R_SYS_SoftwareDelay	✓
39~ 70	IELn_IRQHandler (n = 0~31)	✓
71	R_SYS_GetVersion	

4.2 マクロ／型定義

R_SYSTEM ドライバは、ユーザが参照可能なマクロおよび型定義を `r_system_api.h` ファイルに用意します。

表 4-9 各マクロ定義一覧

マクロ定義	設定値	備考
R_SYSTEM_PRV_PRCR_KEY	(0xA500U)	PRCR register プロテクト解除
R_SYSTEM_PRV_IRQ_EVENT_NUMBER_TOTAL	(32)	IRQ イベントリンクの総割込み数 32 個
R_SYSTEM_PRV_LOCK_LOCKED	(0x01)	Valid <code>st_system_lock_t</code> のロック値 1
R_SYSTEM_PRV_LOCK_UNLOCKED	(0x00)	Valid <code>st_system_lock_t</code> のアンロック値 0
R_SYSTEM_PRV_IELSR_IR_MSK	(0x00010000)	ICU->IELSR register の IR 割込みステータス フラグ マスク設定値
R_SYSTEM_PRV_IELSR_IELS_MSK	(0x0000001F)	ICU->IELSR register の IELS マスク設定値
R_SYSTEM_PRV_OSCSF_HOCOSF_MSK	(0x01)	HOCO クロック発振安定フラグマスク設定 値
R_SYSTEM_PRV_OSCSF_MOSCSF_MSK	(0x08)	メインクロック発振安定フラグマスク設定値
R_SYSTEM_PRV_OSCSF_PLLSF_MSK	(0x20)	PLL クロック発振安定フラグマスク設定値
R_SYSTEM_PRV_SCKSCR_CKSEL_MSK	(0x07)	クロックソース選択マスク設定値
R_SYSTEM_PRV_SCKSCR_CKSEL_HOCO	(0x00)	クロックソースに HOCO を選択
R_SYSTEM_PRV_SCKSCR_CKSEL_MOCO	(0x01)	クロックソースに MOCO を選択
R_SYSTEM_PRV_SCKSCR_CKSEL_LOCO	(0x02)	クロックソースに LOCO を選択
R_SYSTEM_PRV_SCKSCR_CKSEL_MOSC	(0x03)	クロックソースにメインクロックを選択
R_SYSTEM_PRV_SCKSCR_CKSEL_SOSC	(0x04)	クロックソースにサブクロックを選択
R_SYSTEM_PRV_SCKSCR_CKSEL_PLL	(0x05)	クロックソースに PLL を選択
R_SYSTEM_PRV_HOCO_FREQU	(24000000U)	SYSTEM_CFG_HOCO_FREQUENCY = 0 の

ENCY_HZ		時、24MHz に設定
	(32000000U)	SYSTEM_CFG_HOCO_FREQUENCY = 1 の時、32MHz に設定
	(48000000U)	SYSTEM_CFG_HOCO_FREQUENCY = 2 の時、48MHz に設定
	(64000000U)	SYSTEM_CFG_HOCO_FREQUENCY = 3 の時、64MHz に設定
R_SYSTEM_PRV_MOCO_FREQUENCY_HZ	(2000000U)	MOCO 選択時、2MHz に設定
R_SYSTEM_PRV_LOCO_FREQUENCY_HZ	(32768U)	LOCO 選択時、32.768kHz に設定
R_SYSTEM_PRV_SUBCLOCK_FREQUENCY_HZ	(32768U)	サブクロック選択時、32.768kHz に設定
R_SYSTEM_PRV_PLL_DIV_FREQUENCY	(SYSTEM_CFG_MOSC_FREQUENCY_HZ / (SYSTEM_CFG_PLL_DIV+1))	メインクロックの周波数設定と PLL の分周より PLL 回路へ入力時の周波数を設定
R_SYSTEM_PRV_PLL_FREQUENCY_HZ	(R_SYSTEM_PRV_PLL_DIV_FREQUENCY*(SYSTEM_CFG_PLL_MUL+1))	PLL 回路への入力周波数設定と PLL の通倍設定により、PLL 動作時の周波数に設定
R_SYSTEM_PRV_PLL_RANGE	(1)	48000000 < PLL 回路の出力周波数 ≤ 64000000 の範囲の時に設定
	(0)	32000000 ≤ PLL 回路の出力周波数 ≤ 48000000 の範囲の時に設定
R_SYSTEM_PRV_CLOCK_SEL	(R_SYSTEM_PRV_HOCO_FREQUENCY_HZ)	HOCO 選択時、R_SYSTEM_PRV_HOCO_FREQUENCY_HZ で選択された周波数を設定
	(R_SYSTEM_PRV_MOCO_FREQUENCY_HZ)	MOCO 選択時、R_SYSTEM_PRV_MOCO_FREQUENCY_HZ で選択された周波数を設定
	(R_SYSTEM_PRV_LOCO_FREQUENCY_HZ)	LOCO 選択時、R_SYSTEM_PRV_LOCO_FREQUENCY_HZ で選択された周波数を設定
	(SYSTEM_CFG_MOSC_FREQUENCY_HZ)	メインクロック選択時、SYSTEM_CFG_MOSC_FREQUENCY_HZ で選択された周波数を設定
	(R_SYSTEM_PRV_SUBCLOCK_FREQUENCY_HZ)	サブクロック選択時、R_SYSTEM_PRV_SUBCLOCK_FREQUENCY_HZ で選択された周波数を設定
	(R_SYSTEM_PRV_PLL_FREQUENCY_HZ)	PLL 選択時、R_SYSTEM_PRV_PLL_FREQUENCY_HZ で選択された周波数を設定
R_SYSTEM_PRV_CLOCK_ICK_PCKA	(R_SYSTEM_PRV_CLOCK_SEL / (1 << SYSTEM_CFG_ICK_PCKA_DIV))	各クロックで決定した周波数を SYSTEM_CFG_ICK_PCKA_DIV の分周で計算した周波数に設定
R_SYSTEM_PRV_CLOCK_PCKB	(R_SYSTEM_PRV_CLOCK_SEL / (1 << SYSTEM_CFG_PCKB_DIV))	各クロックで決定した周波数を SYSTEM_CFG_PCKB_DIV の分周で計算した周波数に設定
R_SYSTEM_PRV_DELAY_LOOP_CYCLES	(4)	ディレイサイクル数を設定 4 サイクル

4.3 関数仕様

R_SYSTEM ドライバの各関数の仕様と処理フローを示します。

本章の関数仕様の表は、Doxygen に記載している内容に相当します。

処理フローのエラーチェックは、エラー条件のみを列挙し、具体的なチェック方法の記載は省略します。

処理フローの条件分岐には、条件判定に使用する対象を明確にするためレジスタ名および変数名を記載していますが、判定方法は処理フロー内の記述と必ずしも一致しません。

4.3.1 R_SYS_CodeCopy 関数

表 4-10 R_SYS_CodeCopy 関数仕様

書式	void R_SYS_CodeCopy(void)
仕様説明	ROM の規定位置に配置されているデータ/プログラムを RAM の規定位置に展開します。
引数	なし
戻り値	なし
備考	—

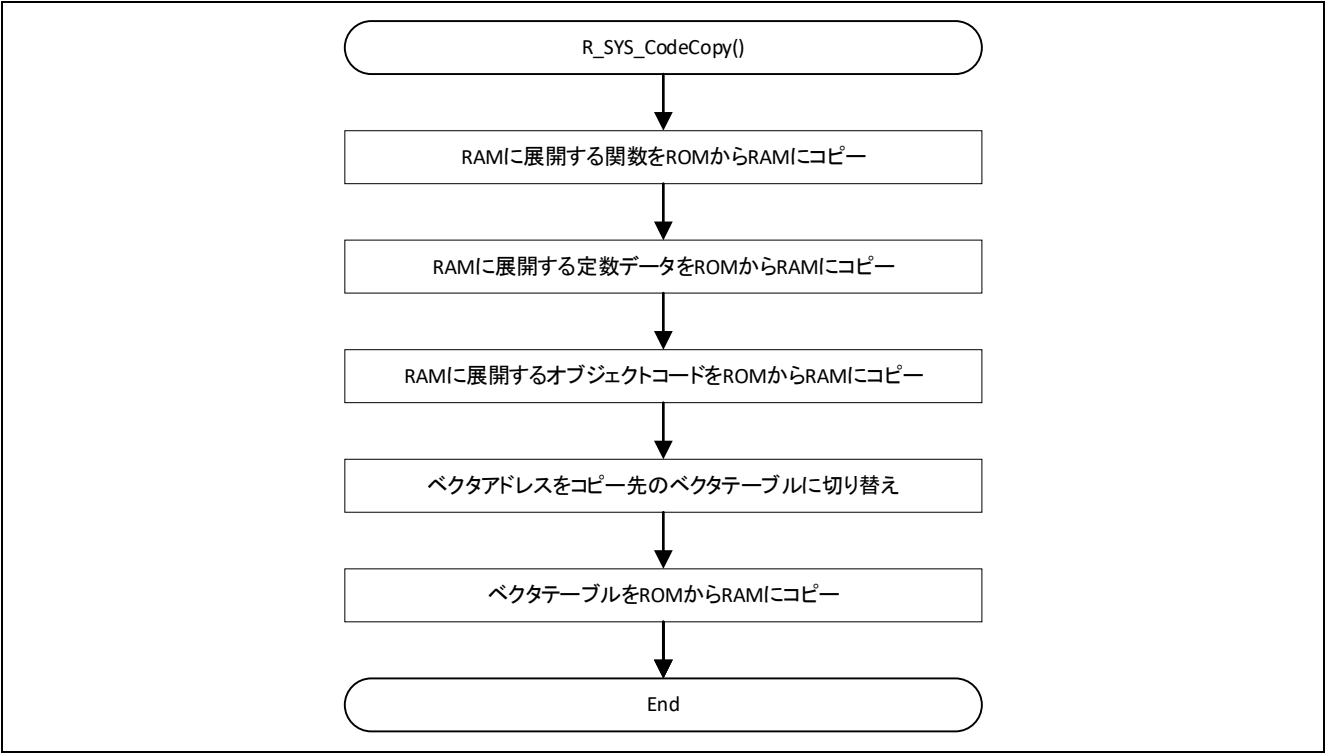


図 4.1 R_SYS_CodeCopy 関数処理フロー

4.3.2 R_SYS_Initialize 関数

表 4-11 R_SYS_Initialize 関数仕様

書式	void R_SYS_Initialize(void)
仕様説明	RAM 初期化処理(コールバック関数、リソースロック状態、レジスタプロテクト状態)を実行します。
引数	なし
戻り値	なし
備考	—

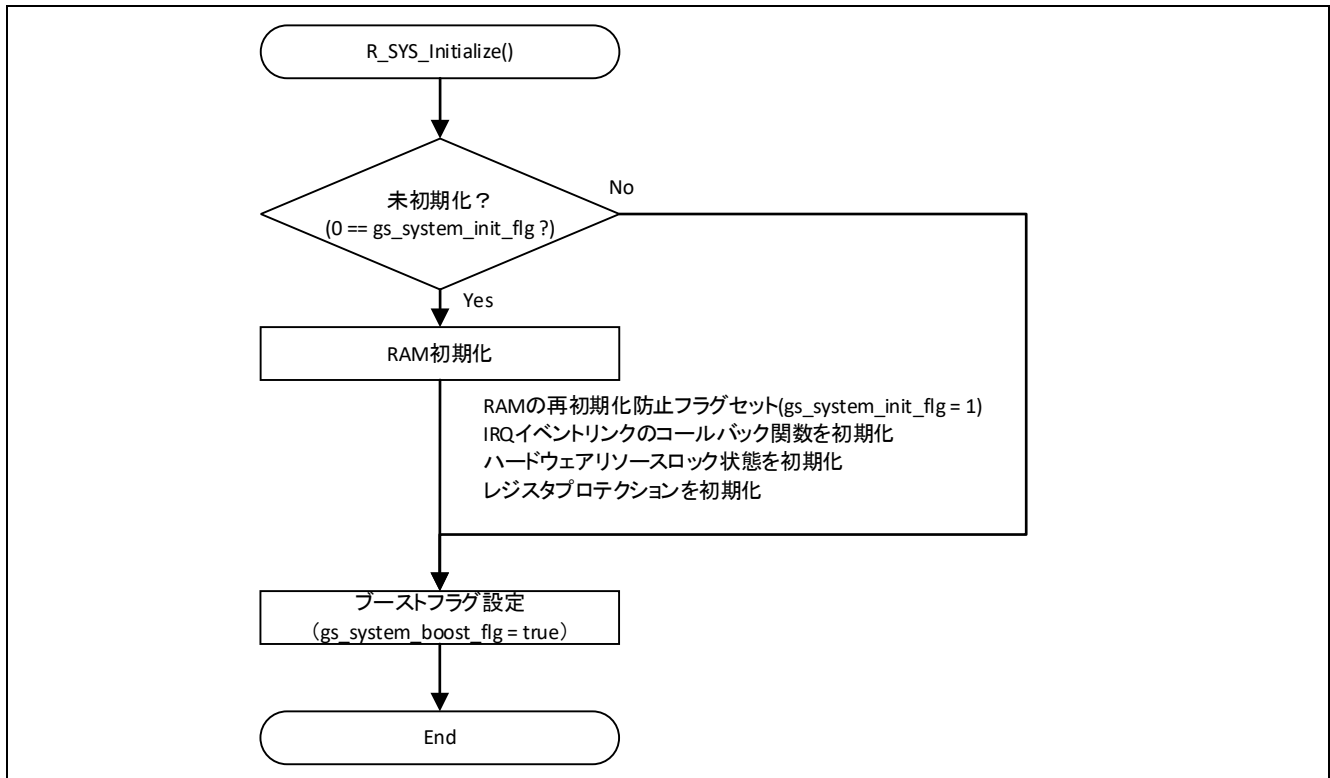


図 4.2 R_SYS_Initialize 関数処理フロー

4.3.3 R_SYS_BoostSpeedModeSet 関数

表 4-12 R_SYS_BoostSpeedModeSet 関数仕様

書式	int32_t R_SYS_BoostSpeedModeSet(void)
仕様説明	電力制御モードを Boost モードに設定します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

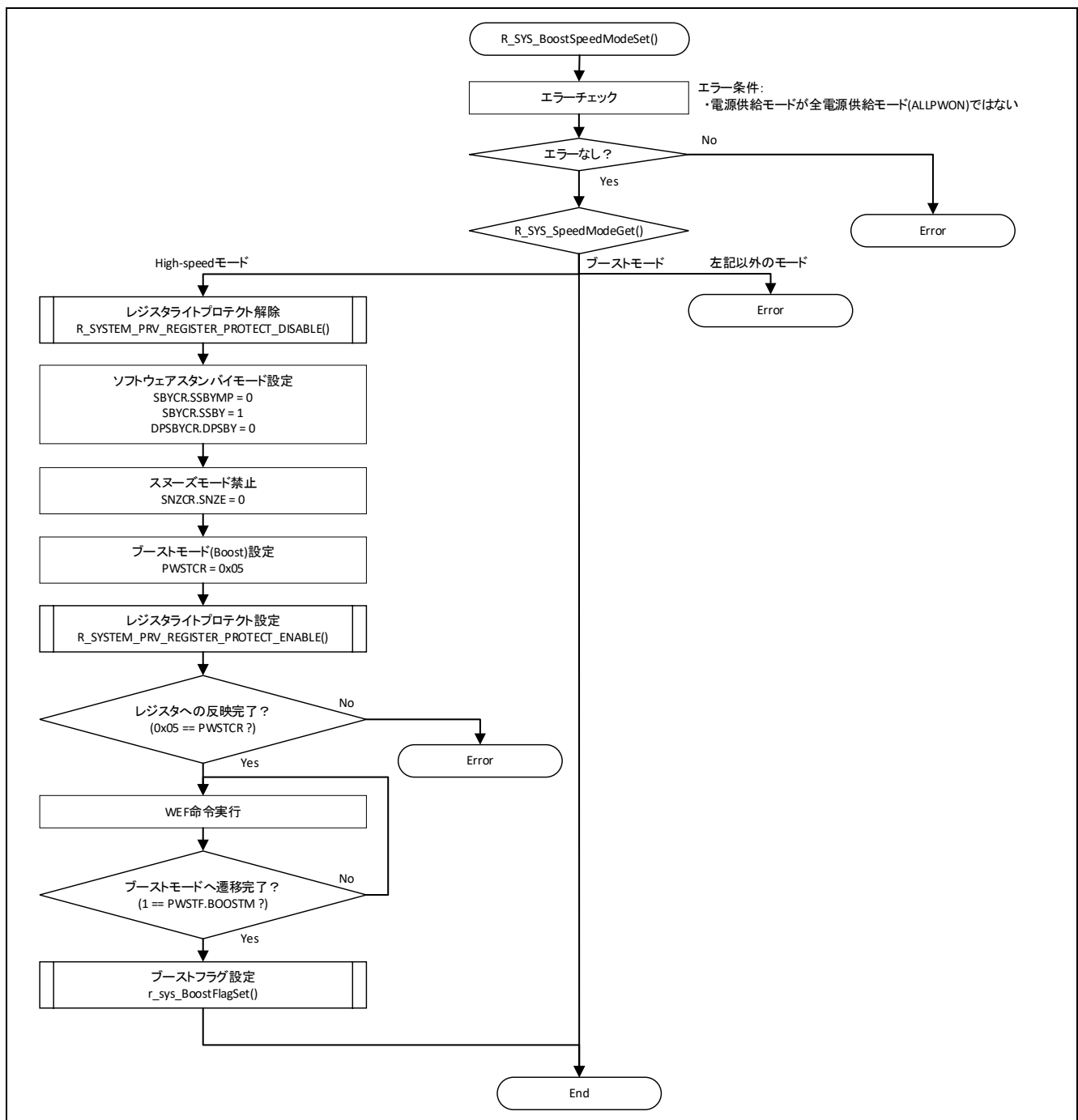


図 4.3 R_SYS_BoostSpeedModeSet 関数処理フロー

4.3.4 R_SYS_HighSpeedModeSet 関数

表 4-13 R_SYS_HighSpeedModeSet 関数仕様

書式	int32_t R_SYS_HighSpeedModeSet(void)
仕様説明	電力制御モードを High-speed モードに設定します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

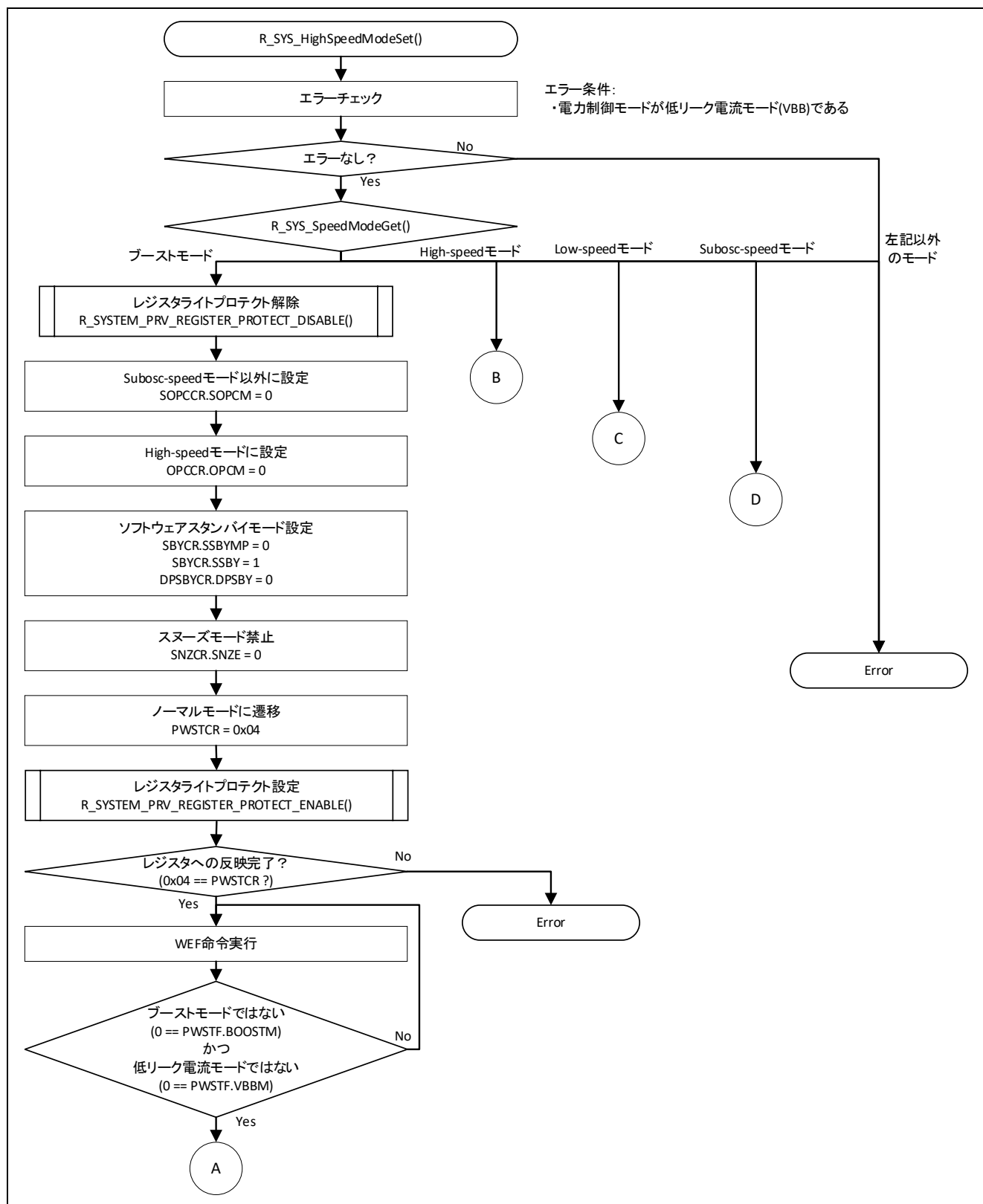


図 4.4 R SYS HighSpeedModeSet 関数処理フロー(1/2)

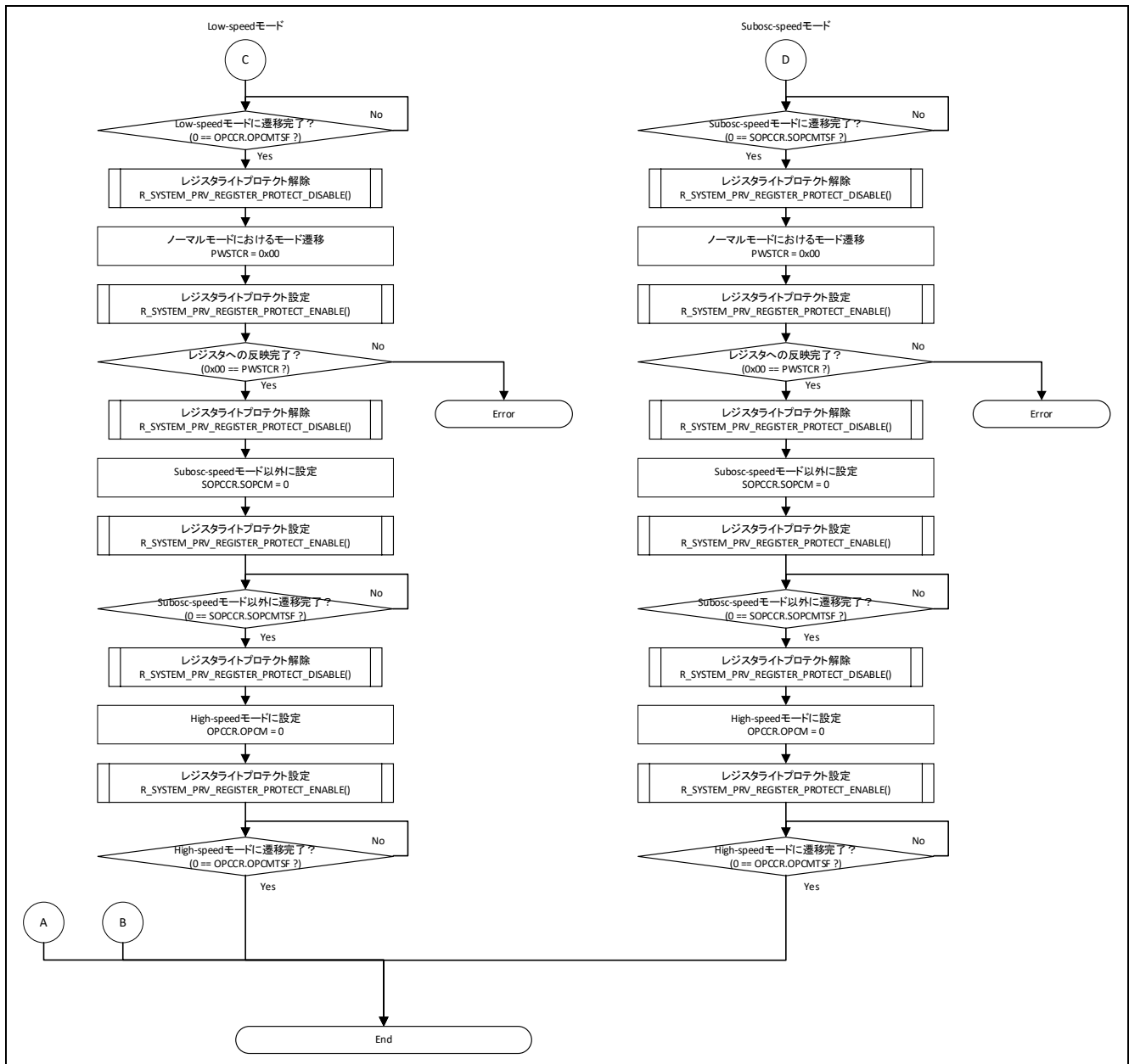


図 4.5 R_SYS_HighSpeedModeSet 関数処理フロー(2/2)

4.3.5 R_SYS_LowSpeedModeSet 関数

表 4-14 R_SYS_LowSpeedModeSet 関数仕様

書式	int32_t R_SYS_LowSpeedModeSet(void)
仕様説明	電力制御モードを Low-speed モードに設定します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

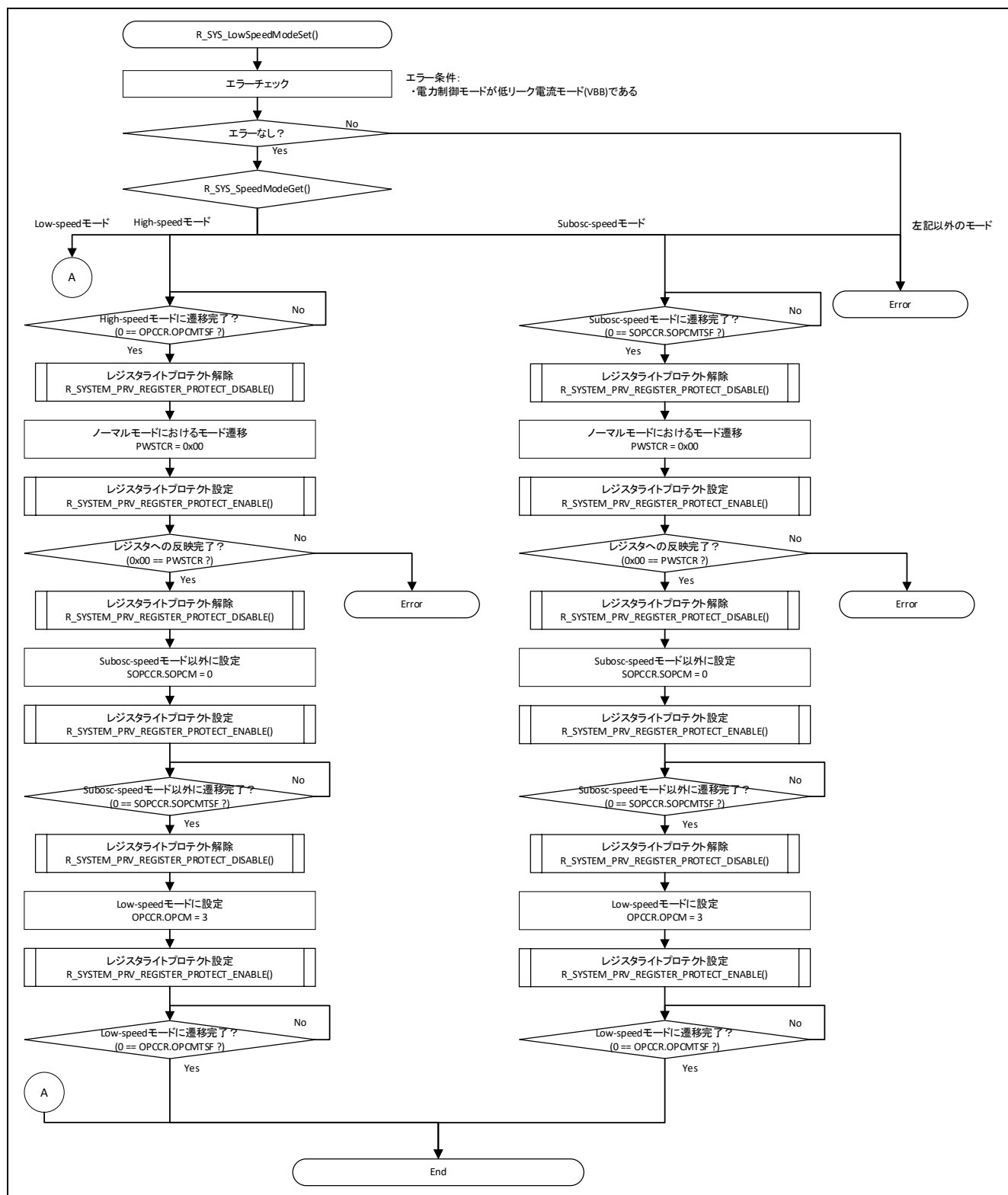


図 4.6 R_SYS_LowSpeedModeSet 関数処理フロー

4.3.6 R_SYS_32kHzSpeedModeSet 関数

表 4-15 R_SYS_32kHzSpeedModeSet 関数仕様

書式	int32_t R_SYS_32kHzSpeedModeSet(void)
仕様説明	電力制御モードを Subosc-speed モードに設定します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

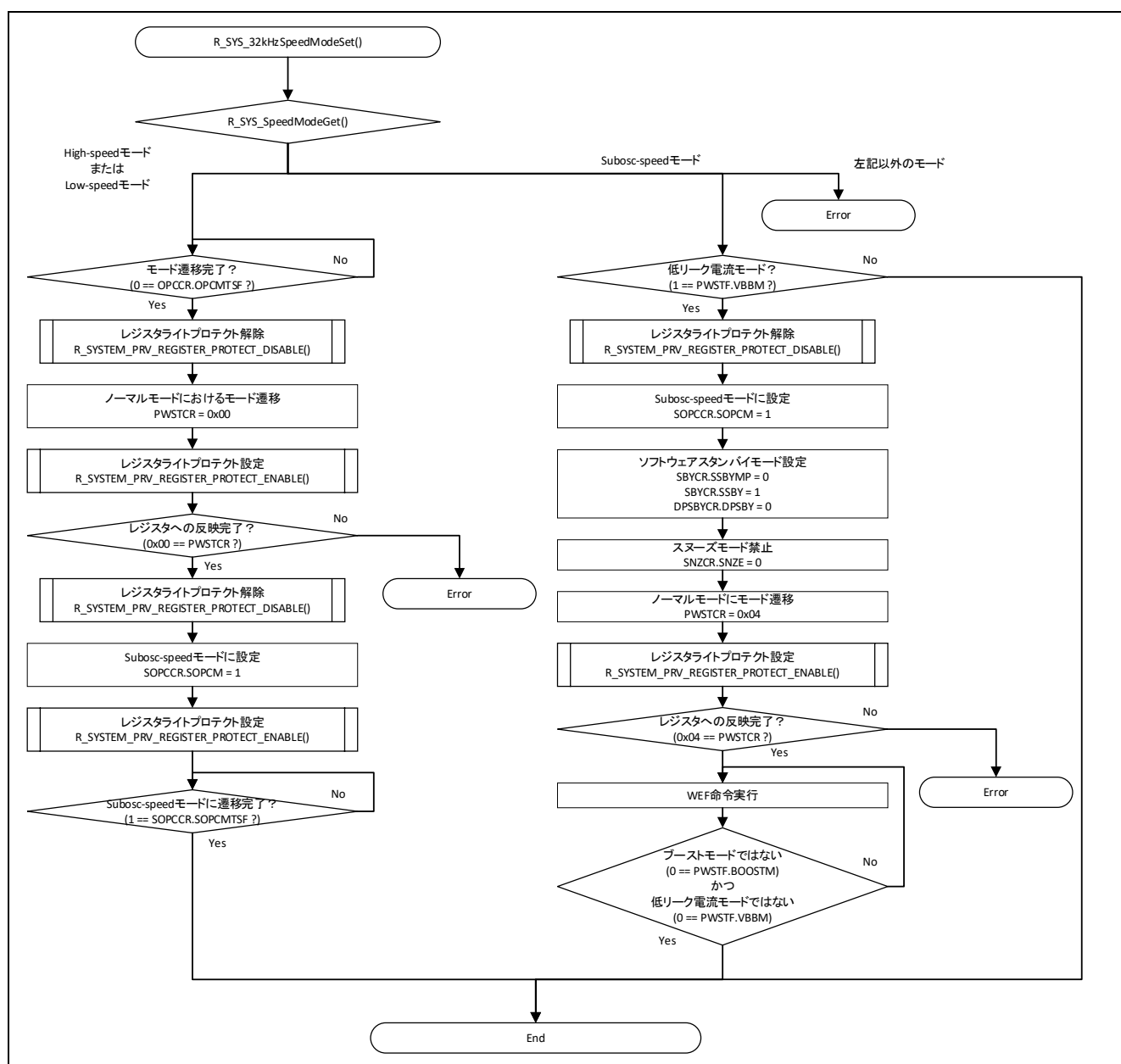


図 4.7 R_SYS_32kHzSpeedModeSet 関数処理フロー

4.3.7 R_SYS_SpeedModeGet 関数

表 4-16 R_SYS_SpeedModeGet 関数仕様

書式	e_system_speed_mode_t R_SYS_SpeedModeGet(void)
仕様説明	現在動作中の電力制御モードを取得します。
引数	なし
戻り値	Boost (0)
	High-speed (1)
	Low-speed (2)
	32kHz-speed (3)
備考	—

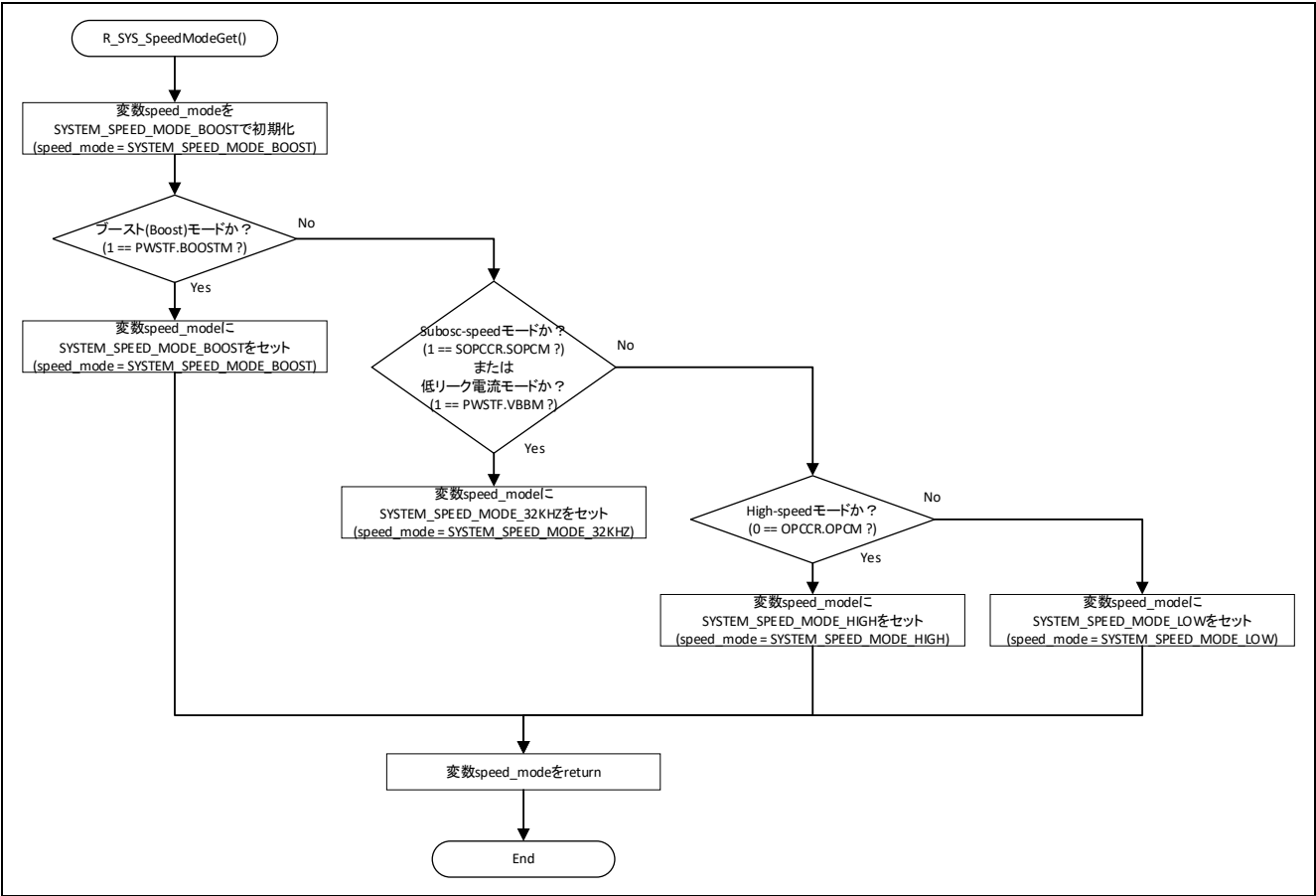


図 4.8 R_SYS_SpeedModeGet 関数処理フロー

4.3.8 R_SYS_SystemClockHOCOSet 関数

表 4-17 R_SYS_SystemClockHOCOSet 関数仕様

書式	int32_t R_SYS_SystemClockHOCOSet(void)
仕様説明	高速オンチップオシレータをシステムクロックに設定します。 動作周波数が 32MHz を超える場合はフラッシュアクセスサイクルを 1 ウェイトに、32MHz 以下の場合は 0 ウェイトに設定します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

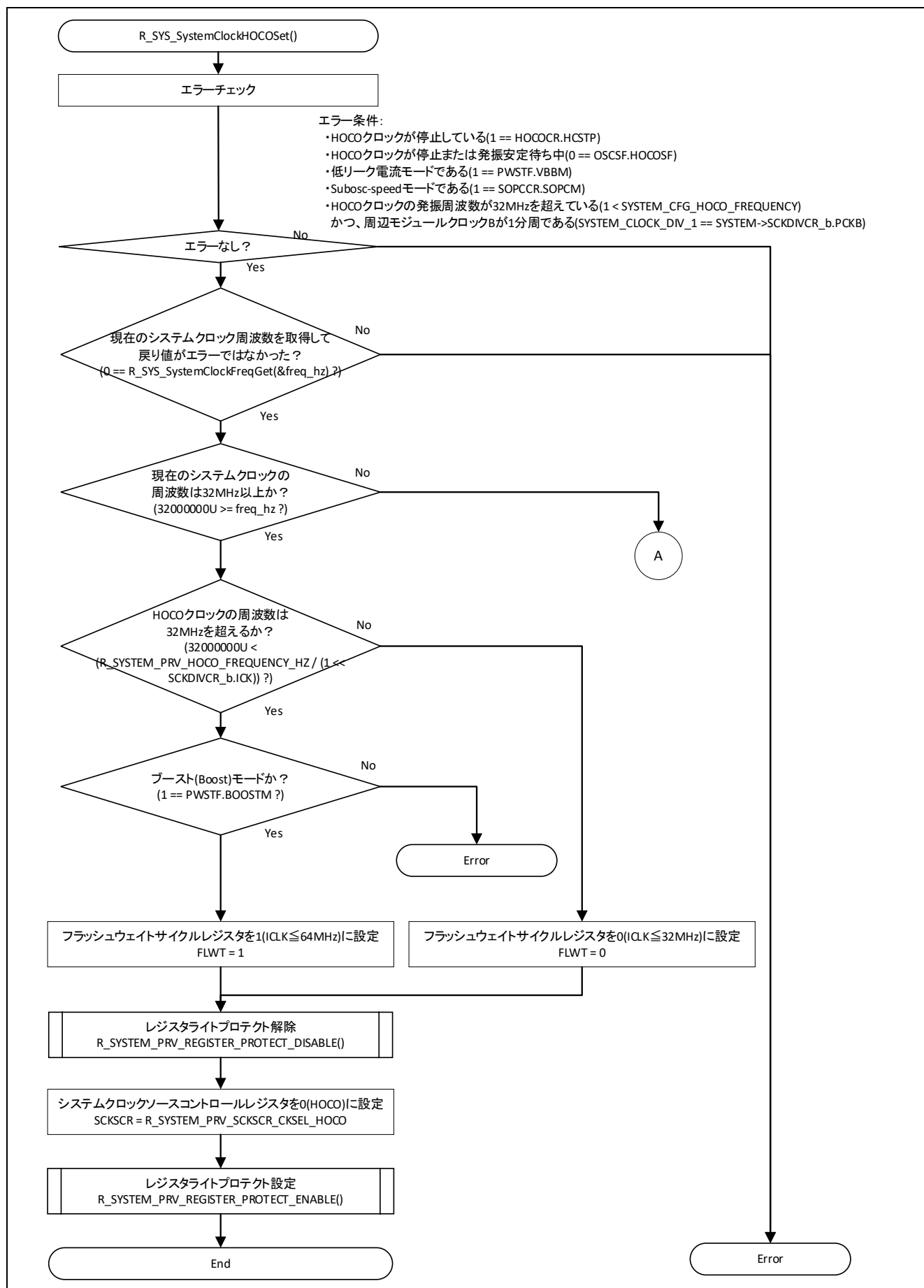


図 4.9 R_SYS_SystemClockHOCOSet 関数処理フロー(1/2)

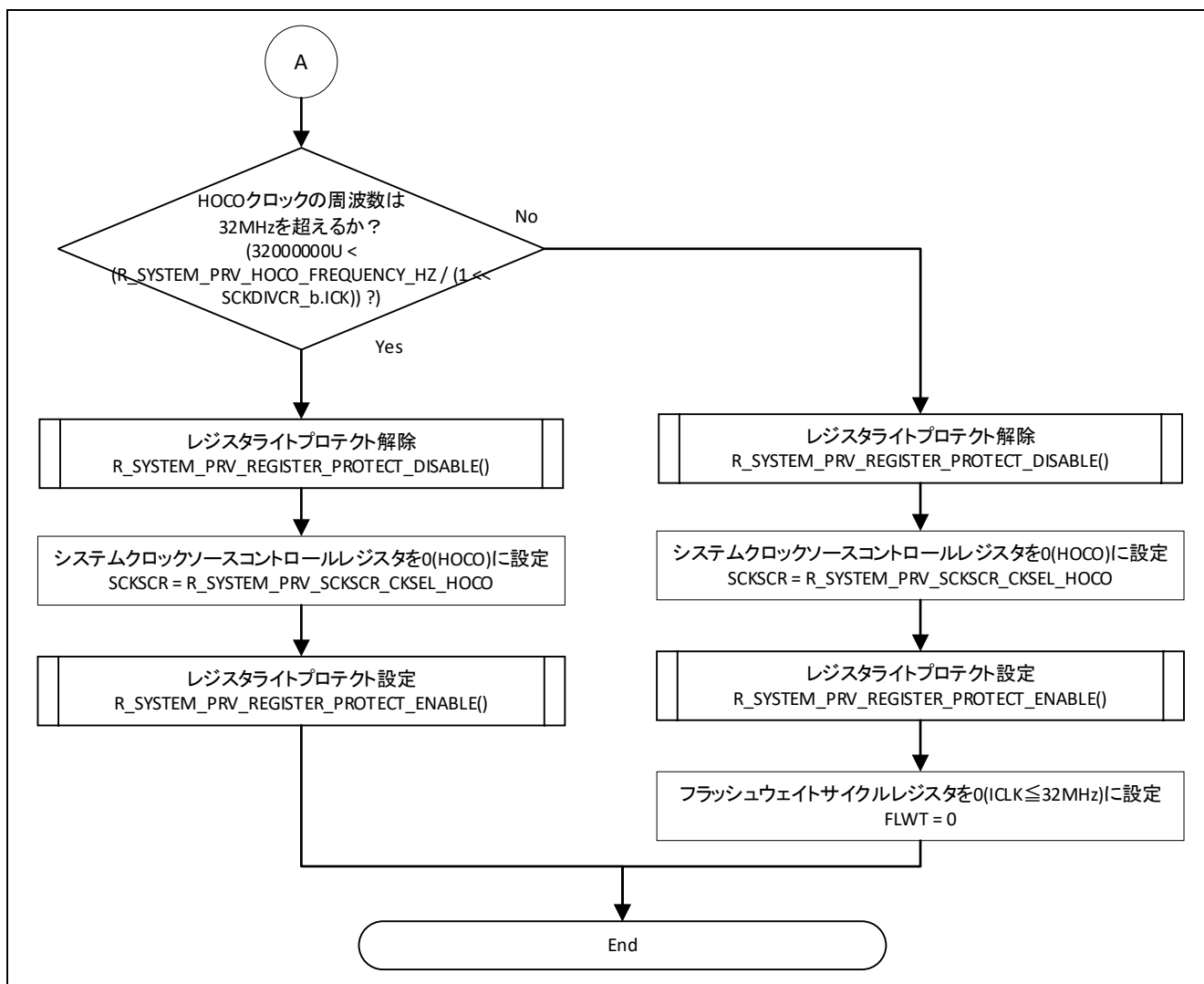


図 4.10 R_SYS_SystemClockHOCOSet 関数処理フロー(2/2)

4.3.9 R_SYS_SystemClockMOCOSet 関数

表 4-18 R_SYS_SystemClockMOCOSet 関数仕様

書式	int32_t R_SYS_SystemClockMOCOSet(void)
仕様説明	中速オンチップオシレータをシステムクロックに設定します。 フラッシュアクセスサイクルを0ウェイトに設定します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

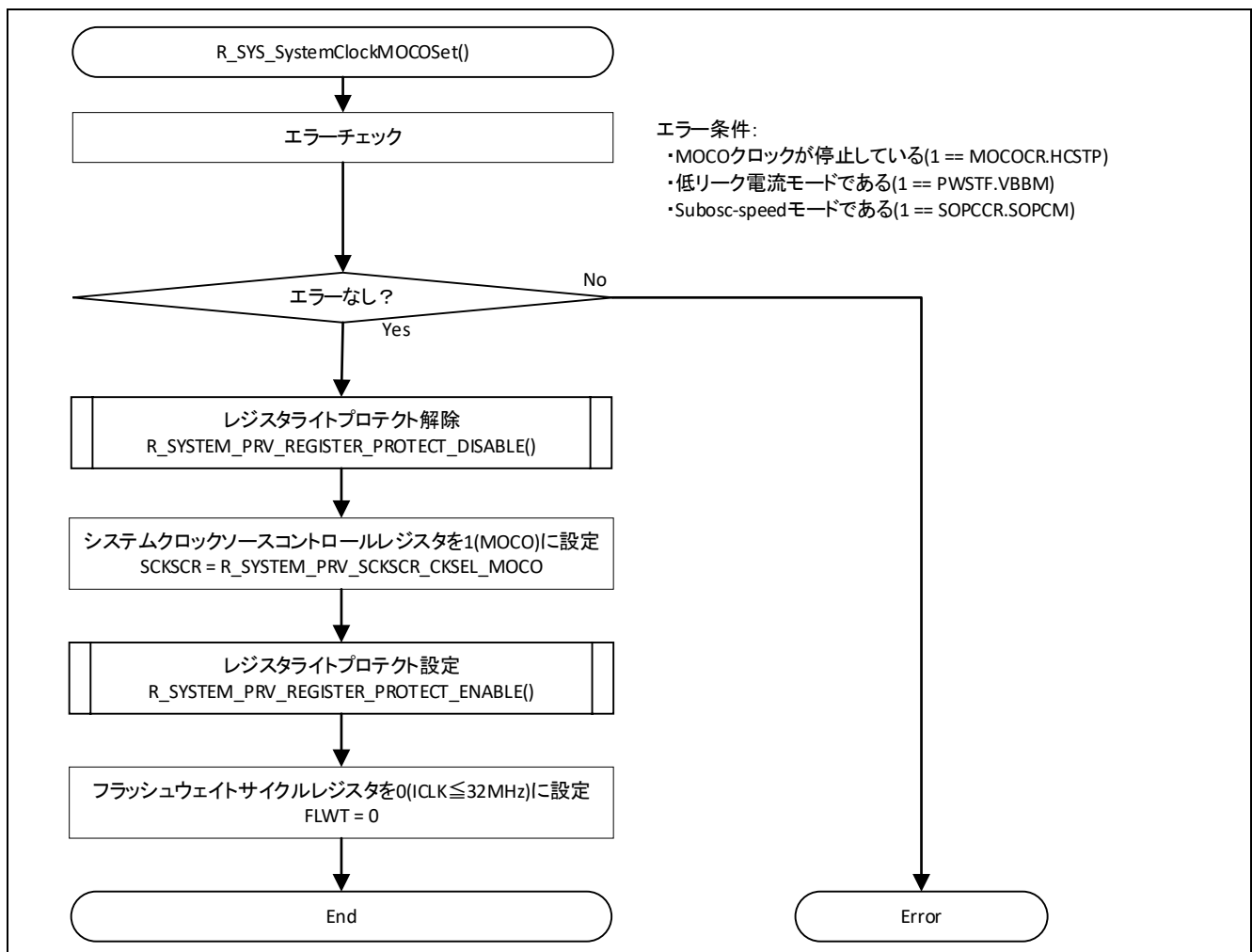


図 4.11 R_SYS_SystemClockMOCOSet 関数処理フロー

4.3.10 R_SYS_SystemClockLOCOSet 関数

表 4-19 R_SYS_SystemClockLOCOSet 関数仕様

書式	int32_t R_SYS_SystemClockLOCOSet(void)
仕様説明	低速オンチップオシレータをシステムクロックに設定します。 フラッシュアクセスサイクルを0ウェイトに設定します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

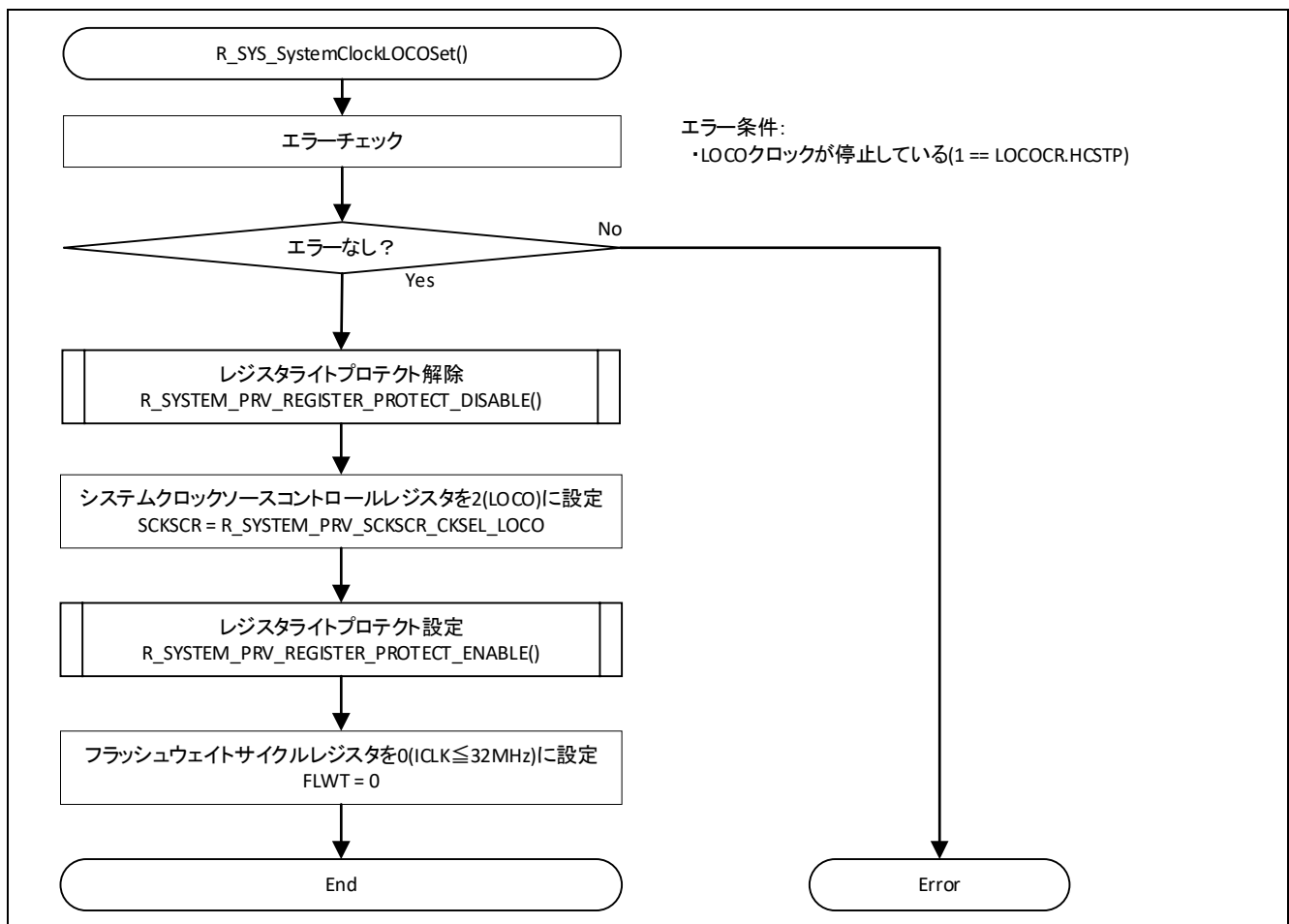


図 4.12 R_SYS_SystemClockLOCOSet 関数処理フロー

4.3.11 R_SYS_SystemClockMOSCSet 関数

表 4-20 R_SYS_SystemClockMOSCSet 関数仕様

書式	int32_t R_SYS_SystemClockMOSCSet(void)
仕様説明	メインクロック発振器をシステムクロックに設定します。 フラッシュアクセスサイクルを 0 ウェイトに設定します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

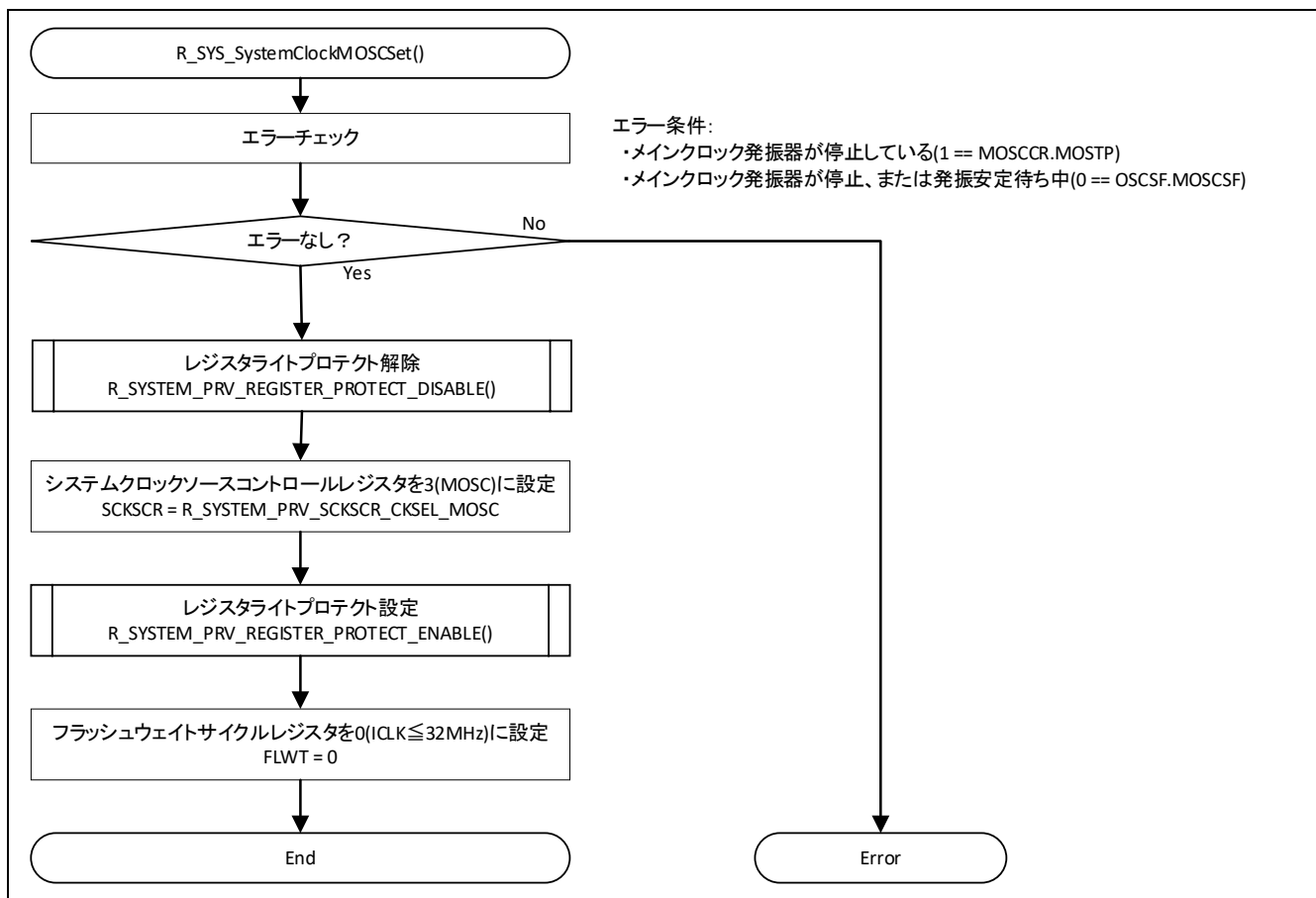


図 4.13 R_SYS_SystemClockMOSCSet 関数処理フロー

4.3.12 R_SYS_SystemClockSOSCSet 関数

表 4-21 R_SYS_SystemClockSOSCSet 関数仕様

書式	int32_t R_SYS_SystemClockSOSCSet(void)
仕様説明	サブクロック発振器をシステムクロックに設定します。 フラッシュアクセスサイクルを 0 ウェイトに設定します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

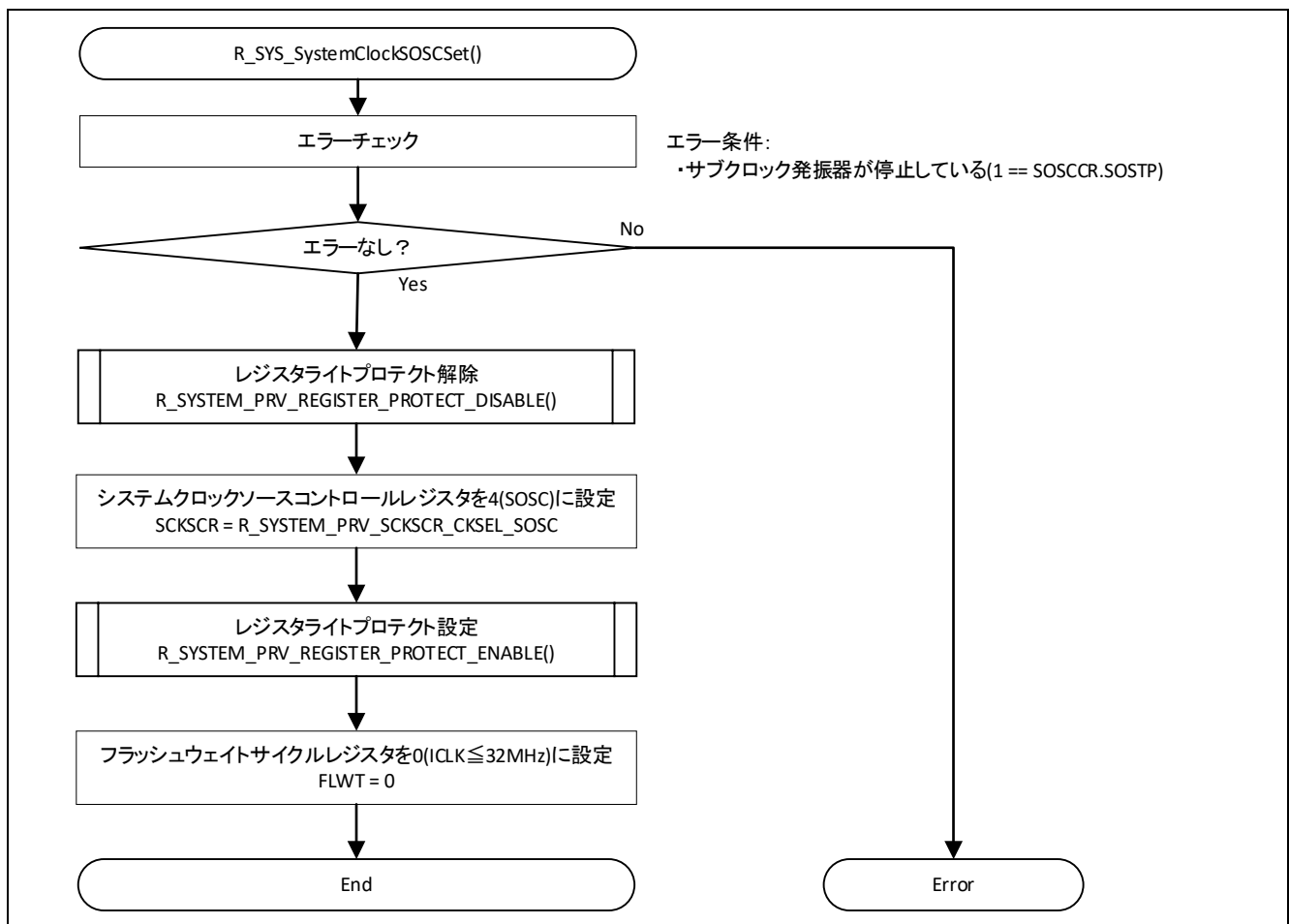


図 4.14 R_SYS_SystemClockSOSCSet 関数処理フロー

4.3.13 R_SYS_SystemClockPLLSet 関数

表 4-22 R_SYS_SystemClockPLLSet 関数仕様

書式	int32_t R_SYS_SystemClockPLLSet(void)
仕様説明	PLL 回路をシステムクロックに設定します。 動作周波数が 32MHz を超える場合はフラッシュアクセスサイクルを 1 ウェイトに、32MHz 以下の場合は 0 ウェイトに設定します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

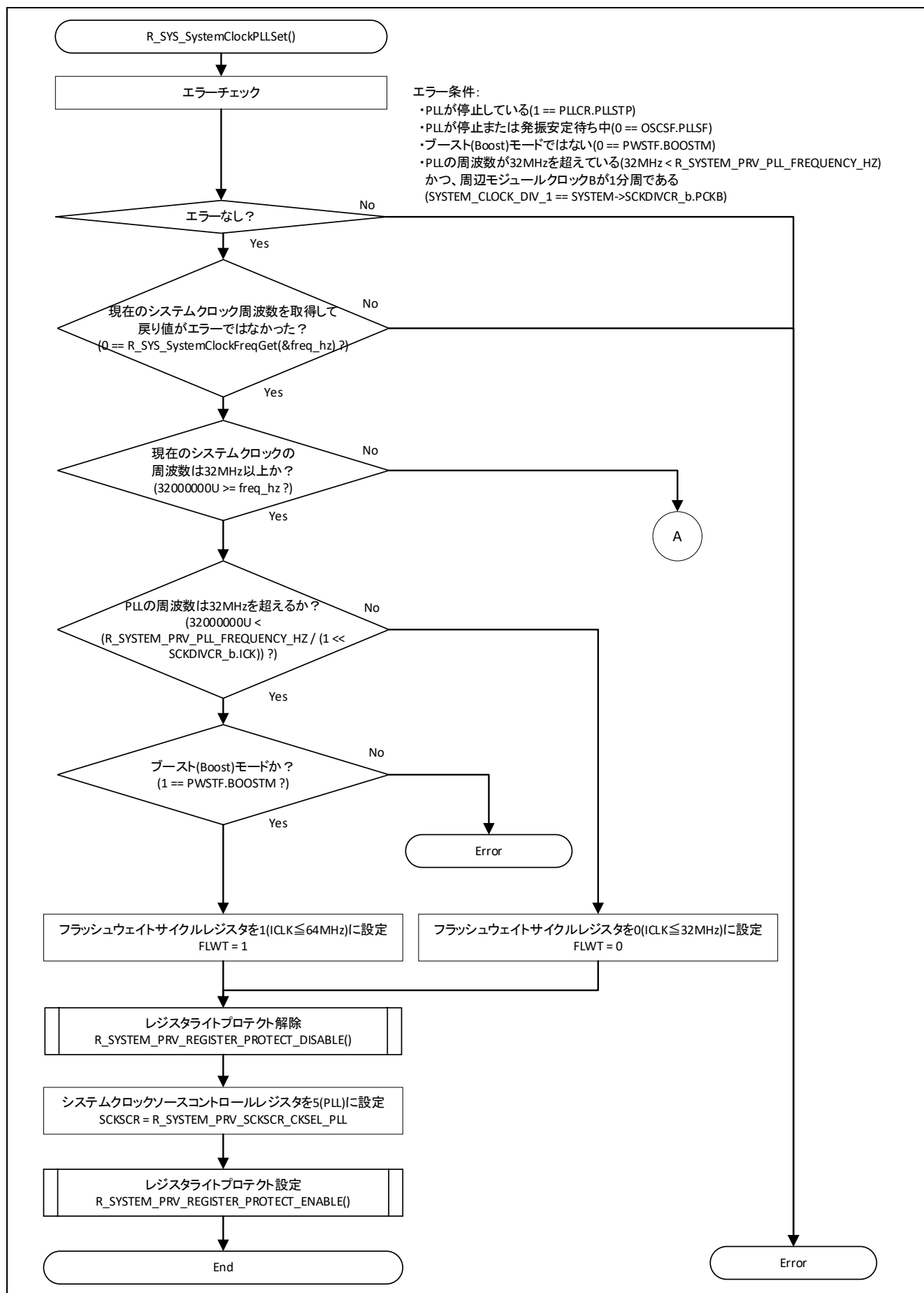


図 4.15 R_SYS_SystemClockPLLSet 関数処理フロー(1/2)

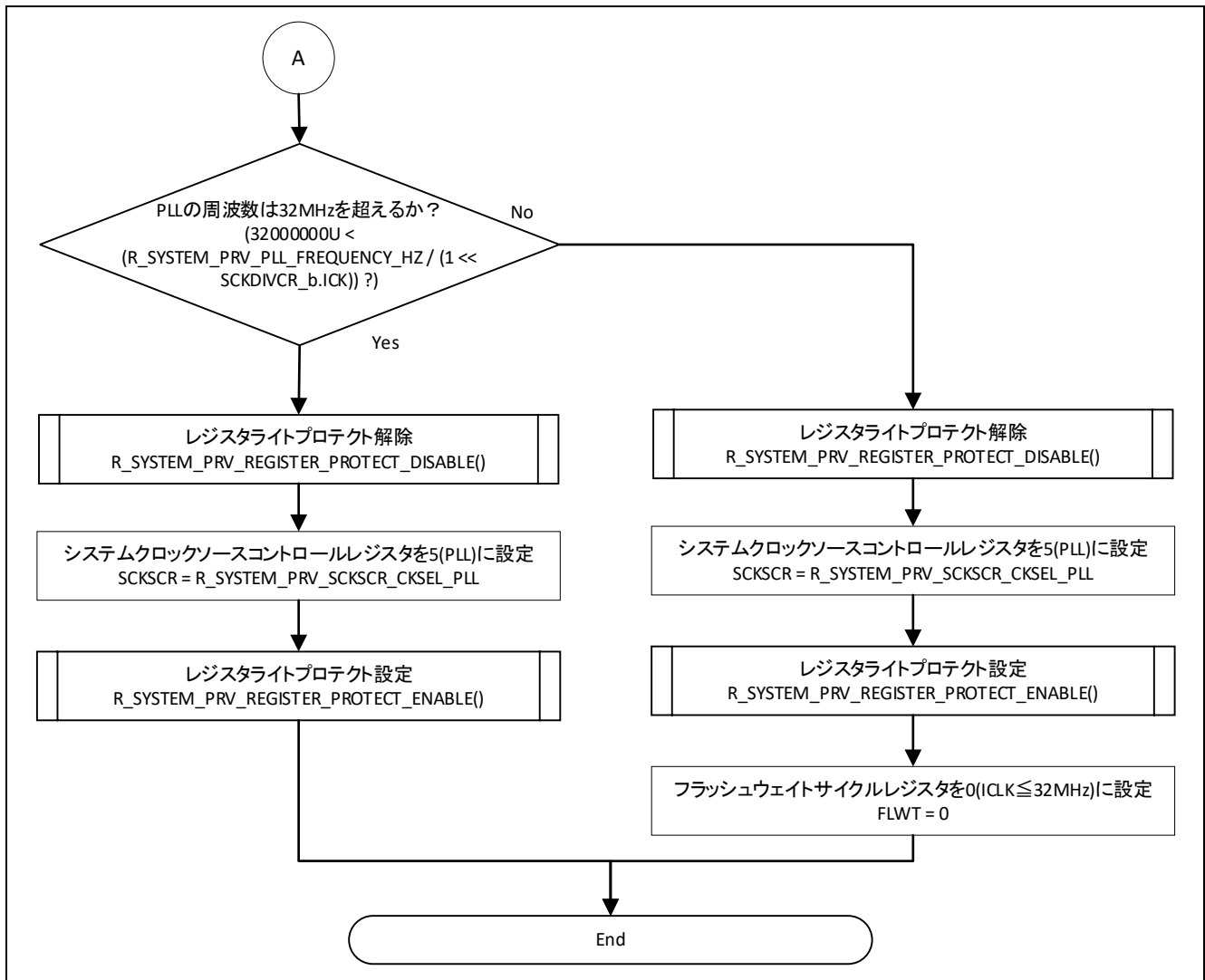


図 4.16 R_SYS_SystemClockPLLSet 関数処理フロー(2/2)

4.3.14 R_SYS_SystemClockFreqGet 関数

表 4-23 R_SYS_SystemClockFreqGet 関数仕様

書式	int32_t R_SYS_SystemClockFreqGet(uint32_t * p_freq_hz)
仕様説明	システムクロック(ICLK)/周辺モジュールクロック(PCLKA)の周波数を取得します。
引数	uint32_t * p_freq_hz[入力] : 取得した周波数の格納先を設定します。
戻り値	正常 (0)
	異常 (-1)
備考	—

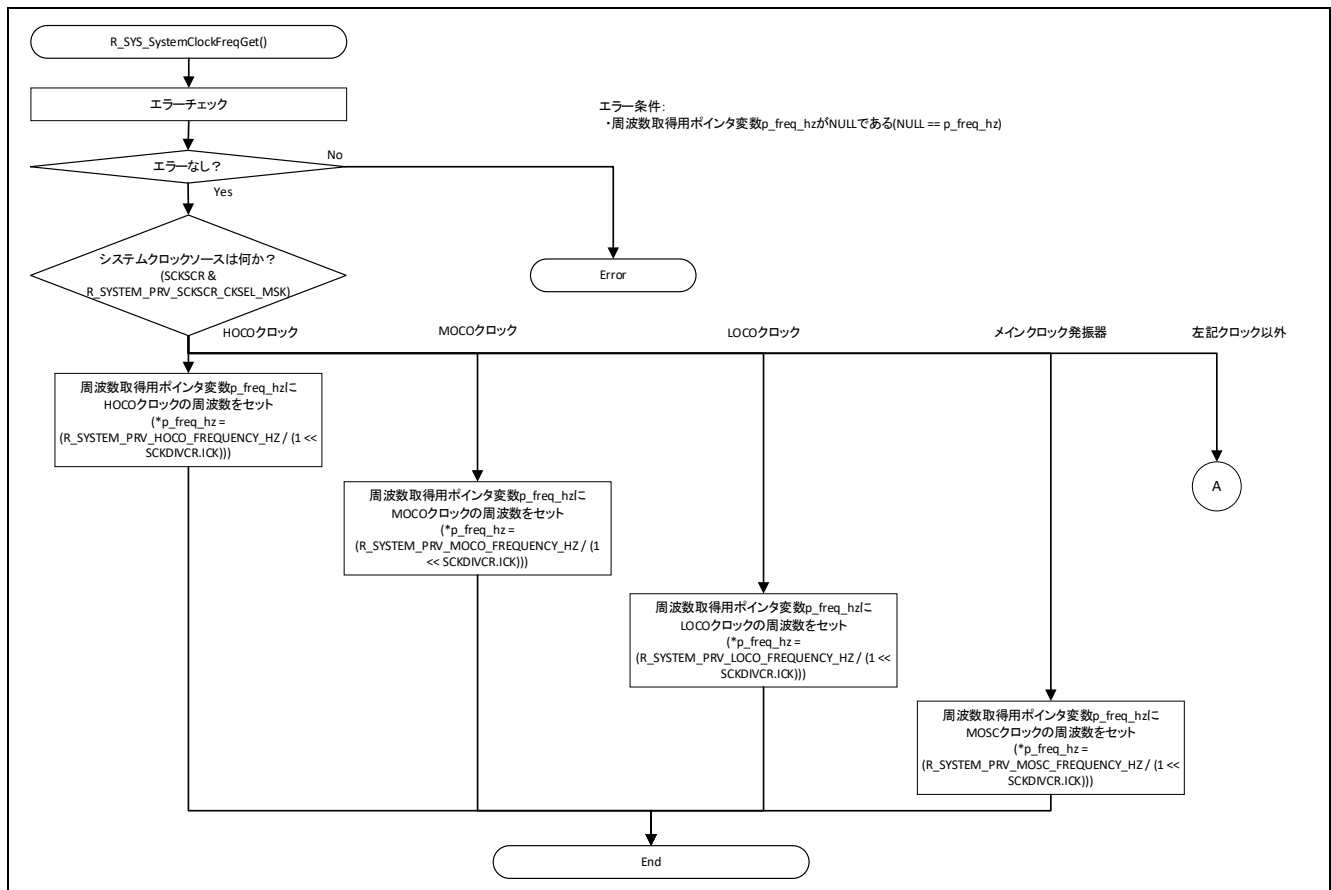


図 4.17 R_SYS_SystemClockFreqGet 関数処理フロー(1/2)

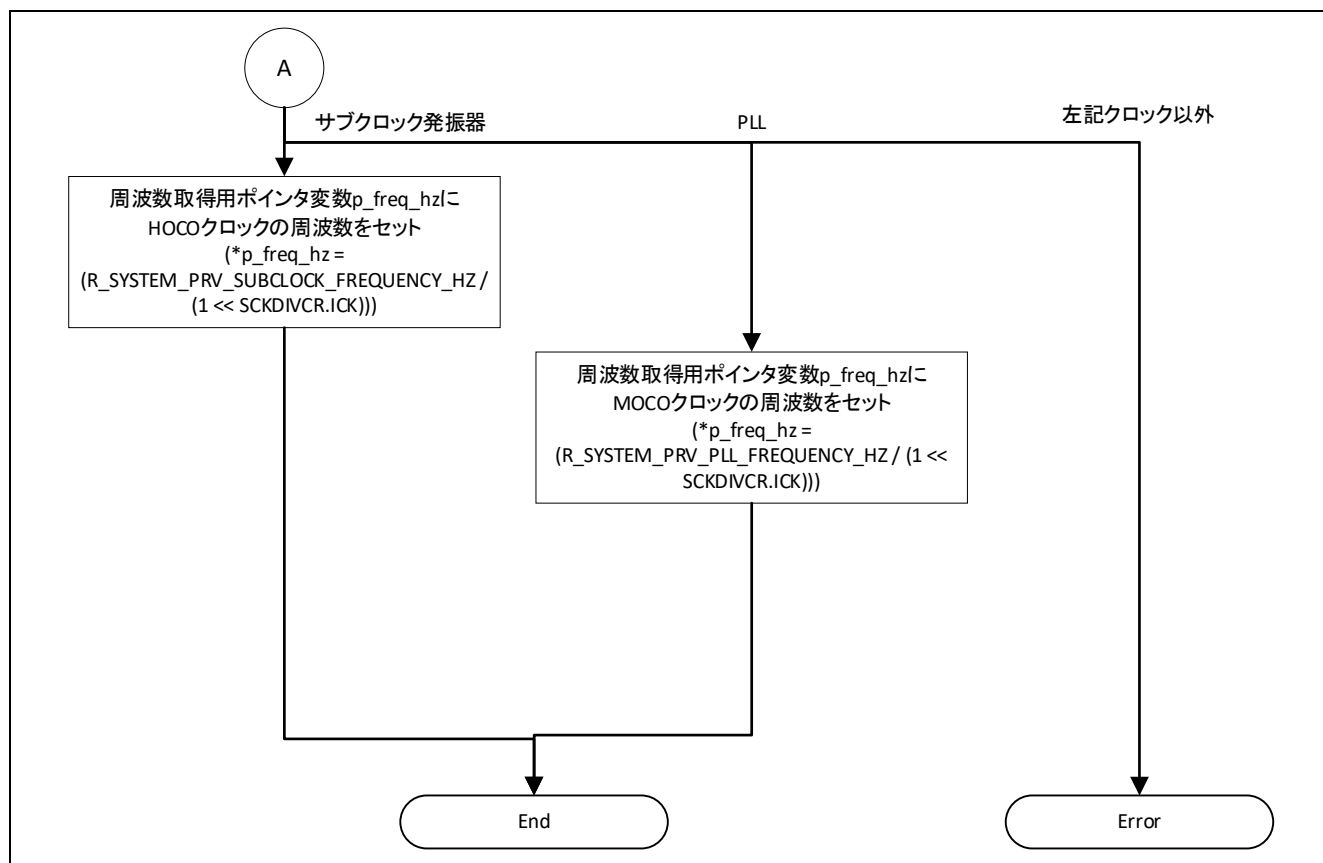


図 4.18 R_SYS_SystemClockFreqGet 関数処理フロー(2/2)

4.3.15 R_SYS_PeripheralClockFreqGet 関数

表 4-24 R_SYS_PeripheralClockFreqGet 関数仕様

書式	int32_t R_SYS_PeripheralClockFreqGet(uint32_t * p_freq_hz)
仕様説明	周辺モジュールクロック B(PCLKB)の周波数を取得します。
引数	uint32_t * p_freq_hz[入力]：取得した周波数の格納先を設定します。
戻り値	正常 (0)
	異常 (-1)
備考	—

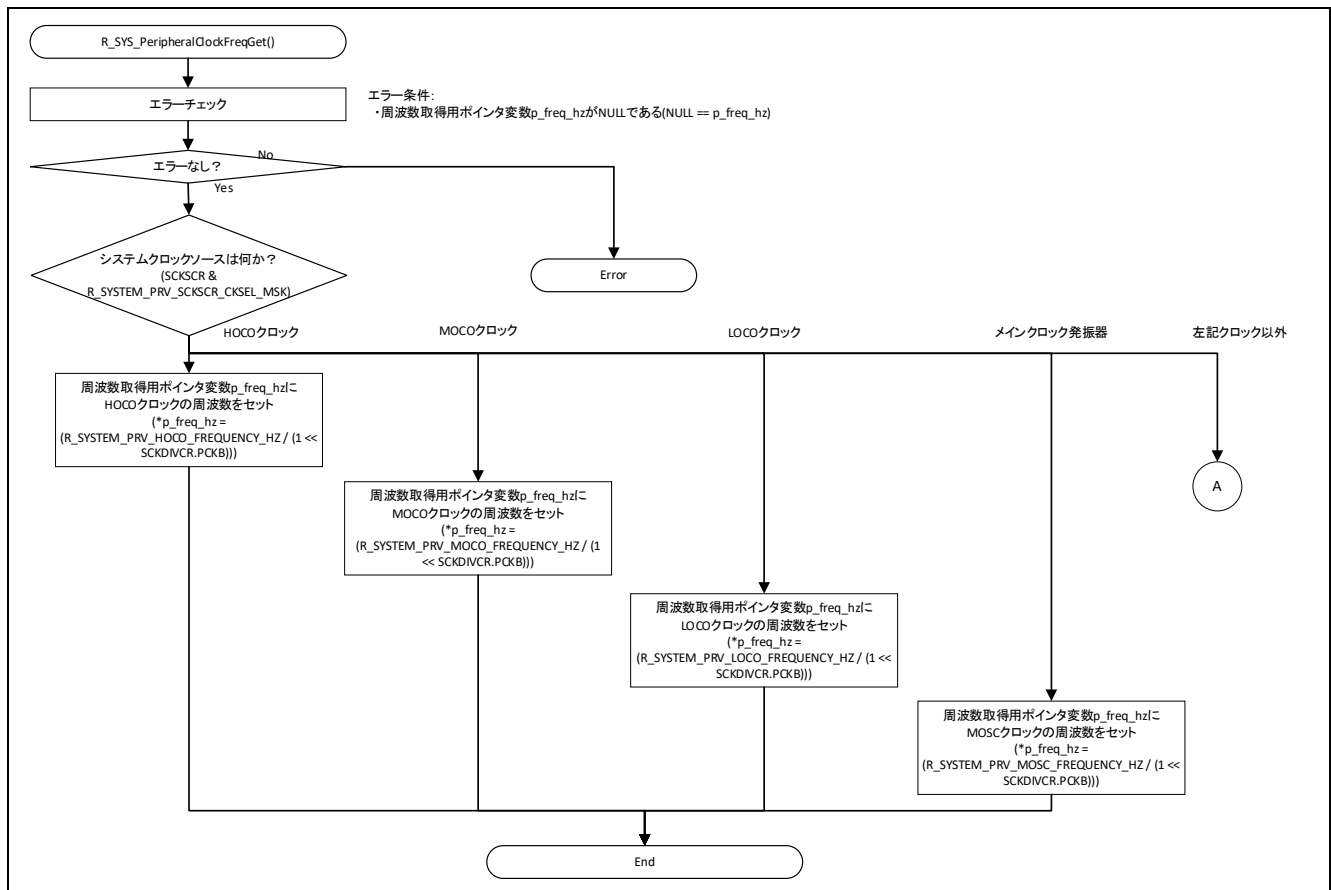


図 4.19 R_SYS_PeripheralClockFreqGet 関数処理フロー(1/2)

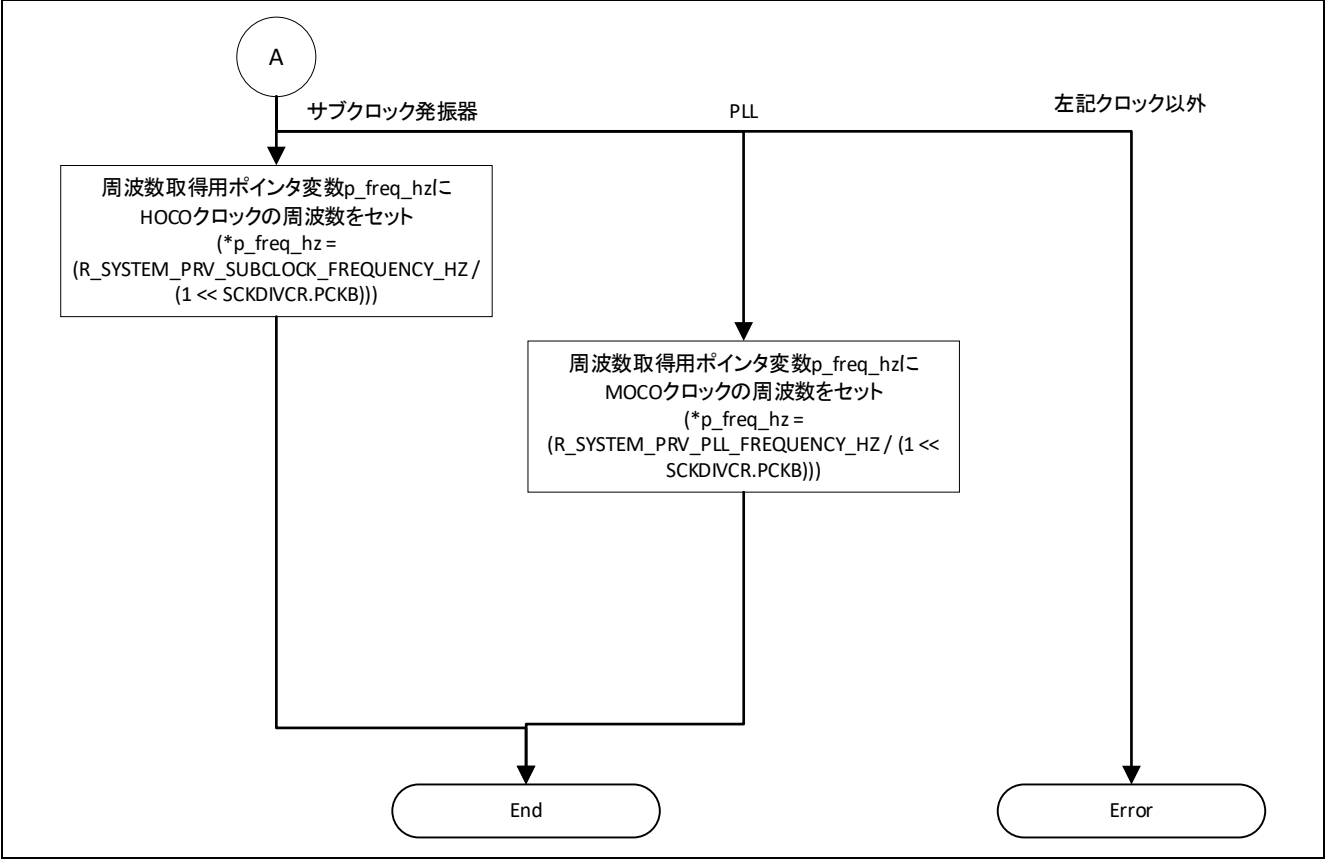


図 4.20 R_SYS_PeripheralClockFreqGet 関数処理フロー(2/2)

4.3.16 R_SYS_SystemClockDividerSet 関数

表 4-25 R_SYS_SystemClockDividerSet 関数仕様

書式	int32_t R_SYS_SystemClockDividerSet(e_system_sys_clock_div_t iclk_div, e_system_sys_clock_div_t pclk_div)
仕様説明	システムクロック(ICLK)/周辺モジュールクロック(PCLKA)および周辺モジュールクロックB(PCLKB)の分周値を設定します。
引数	e_system_sys_clock_div_t iclk_div[入力] : システムクロック(ICLK)/周辺モジュールクロック(PCLKA)の分周値を設定します。 e_system_sys_clock_div_t pclk_div[入力] : 周辺モジュールクロック(PCLKB)の分周値を設定します。
戻り値	正常 (0) 異常 (-1)
備考	—

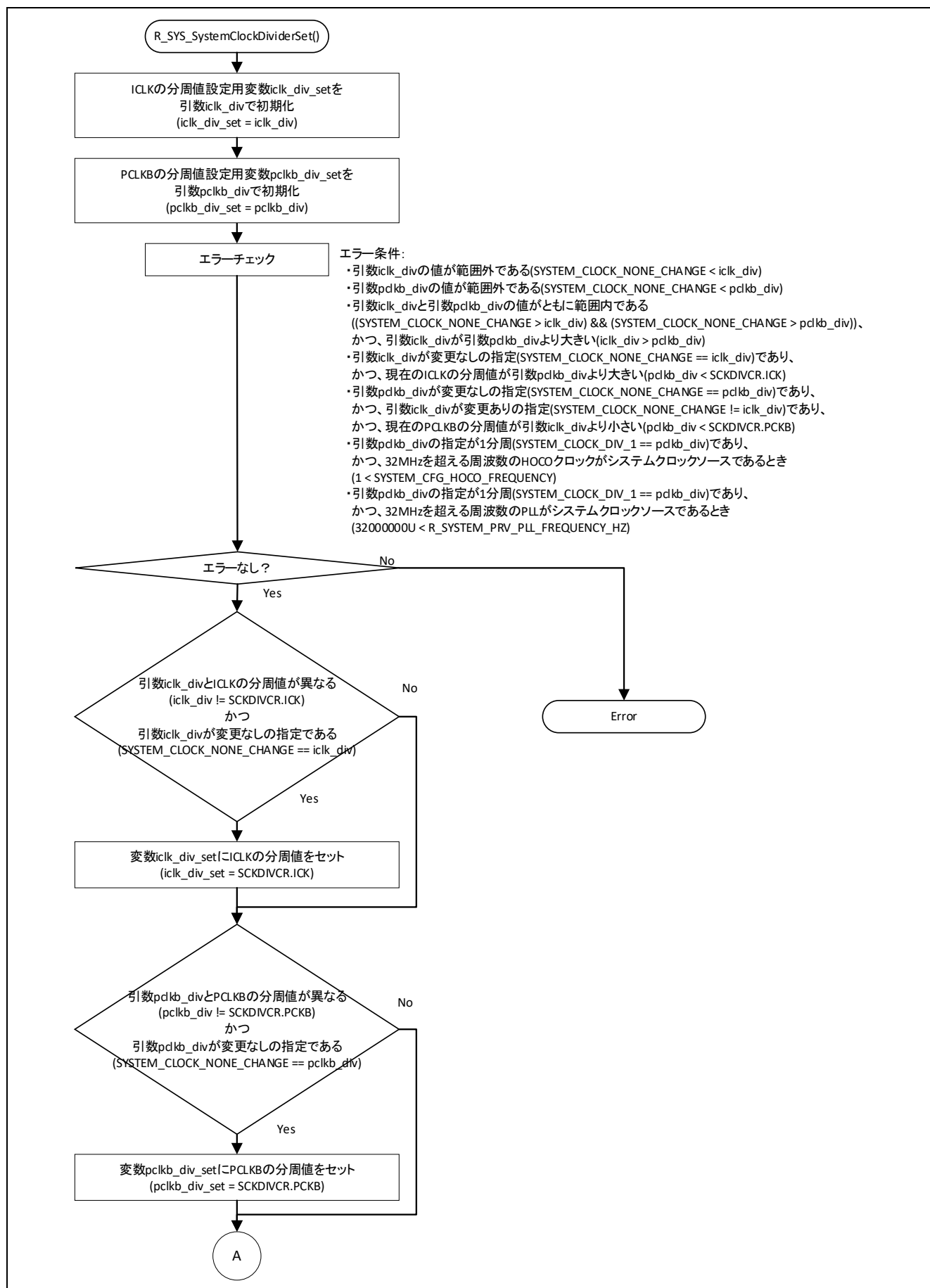


図 4.21 R_SYS_SystemClockDividerSet 関数処理フロー(1/2)

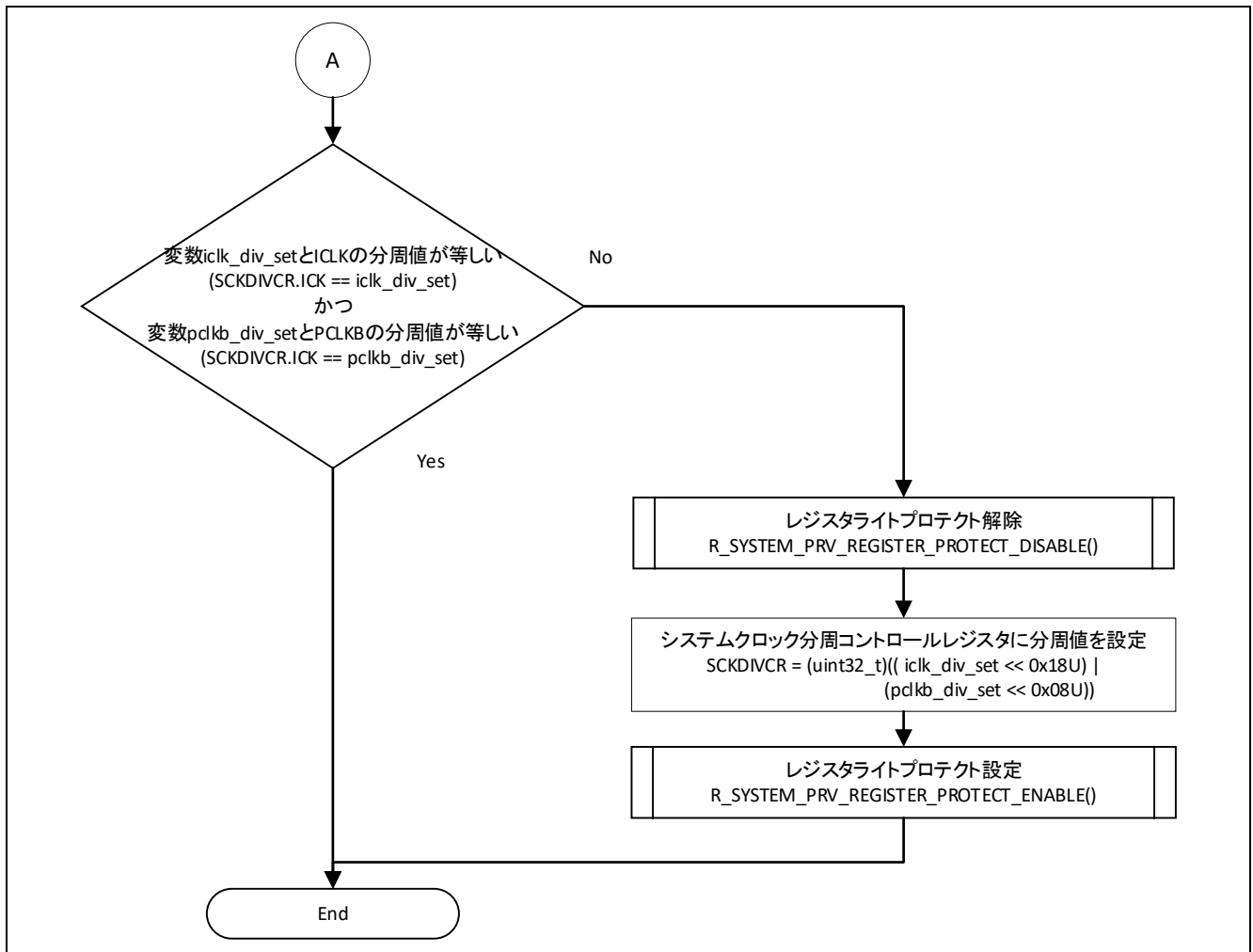


図 4.22 R_SYS_SystemClockDividerSet 関数処理フロー(2/2)

4.3.17 R_SYS_MainOscSpeedClockStart 関数

表 4-26 R_SYS_MainOscSpeedClockStart 関数仕様

書式	void R_SYS_MainOscSpeedClockStart(void)
仕様説明	メインクロック発振器の動作を開始します。
引数	なし
戻り値	なし
備考	—

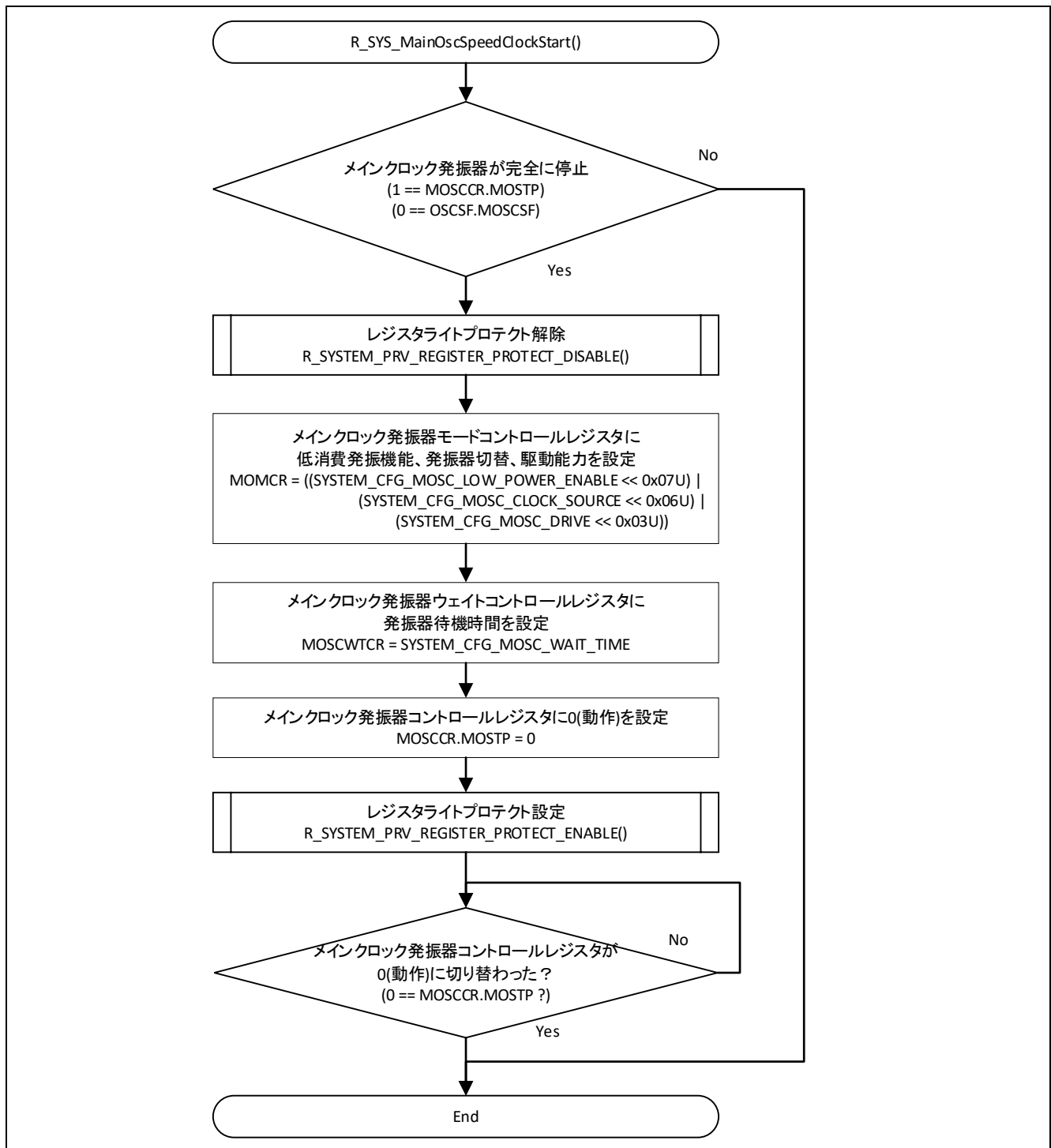


図 4.23 R_SYS_MainOscSpeedClockStart 関数処理フロー

4.3.18 R_SYS_MainOscSpeedClockStop 関数

表 4-27 R_SYS_MainOscSpeedClockStop 関数仕様

書式	int32_t R_SYS_MainOscSpeedClockStop(void)
仕様説明	メインクロック発振器の動作を停止します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

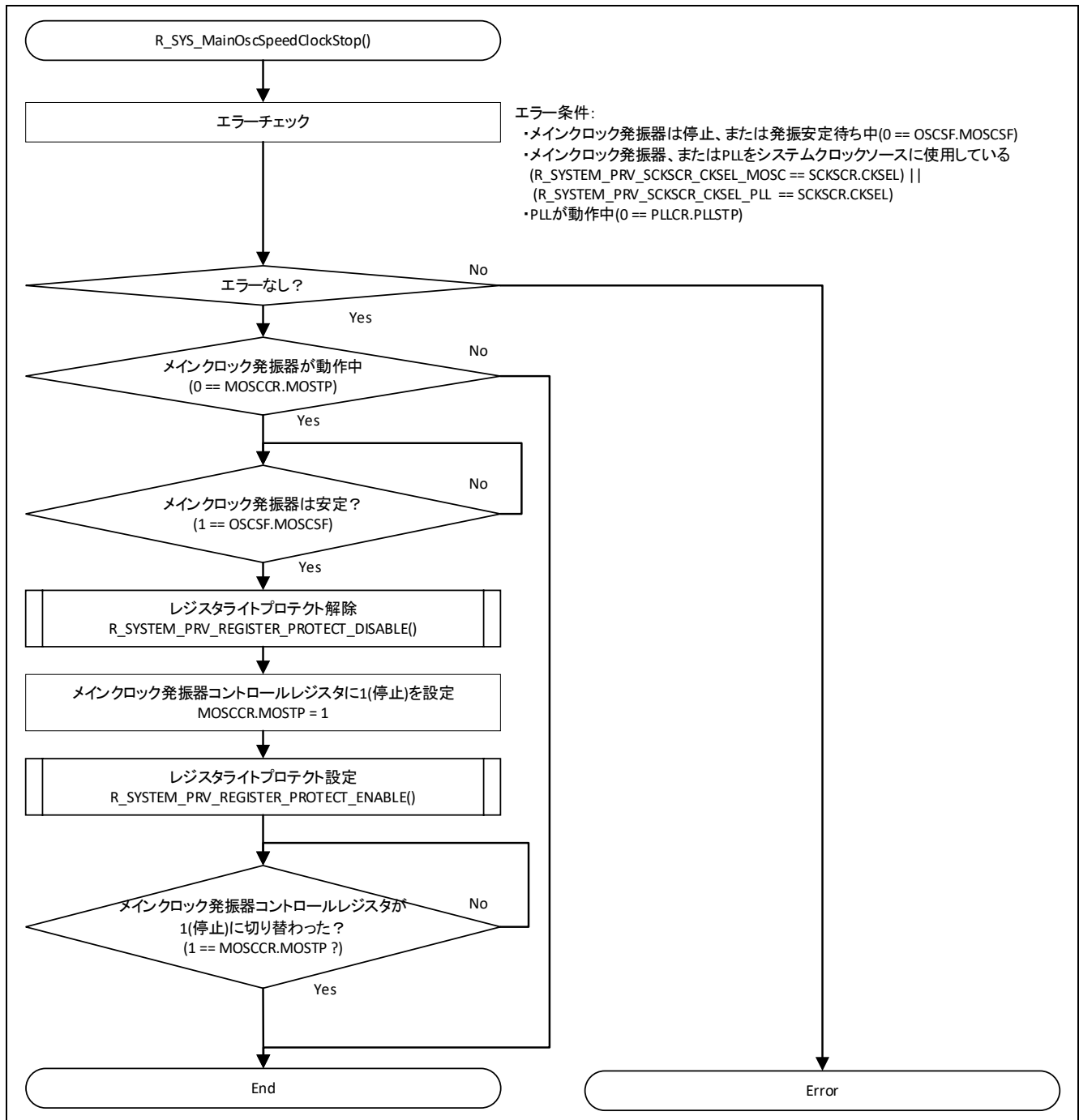
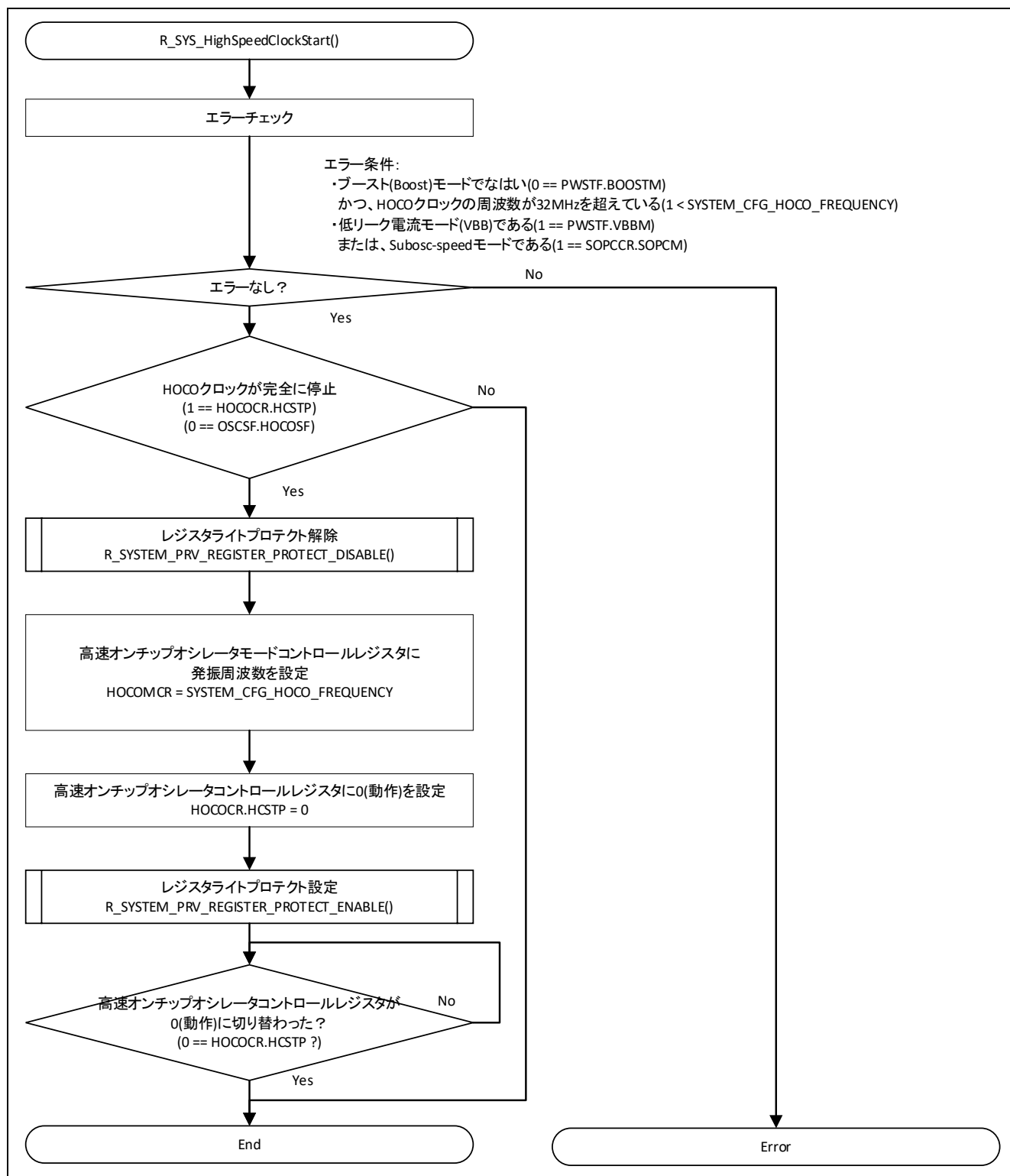


図 4.24 R_SYS_MainOscSpeedClockStop 関数処理フロー

4.3.19 R_SYS_HighSpeedClockStart 関数

表 4-28 R_SYS_HighSpeedClockStart 関数仕様

書式	int32_t R_SYS_HighSpeedClockStart(void)
仕様説明	高速オンチップオシレータの動作を開始します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—



4.3.20 R_SYS_HighSpeedClockStop 関数

表 4-29 R_SYS_HighSpeedClockStop 関数仕様

書式	int32_t R_SYS_HighSpeedClockStop(void)
仕様説明	高速オンチップオシレータの動作を停止します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

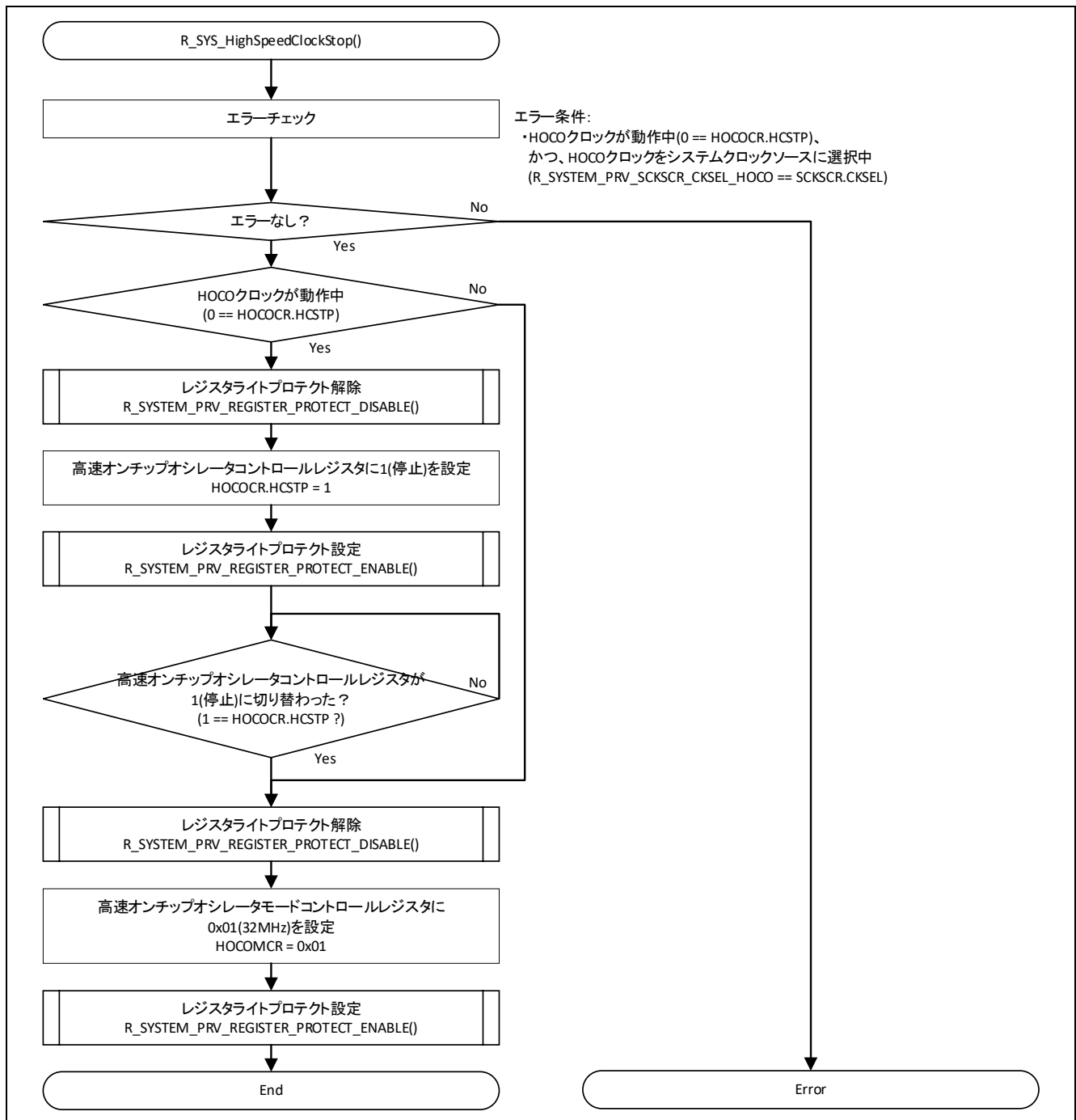


図 4.26 R_SYS_HighSpeedClockStop 関数処理フロー

4.3.21 R_SYS_MediumSpeedClockStart 関数

表 4-30 R_SYS_MediumSpeedClockStart 関数仕様

書式	int32_t R_SYS_MediumSpeedClockStart(void)
仕様説明	中速オンチップオシレータの動作を開始します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

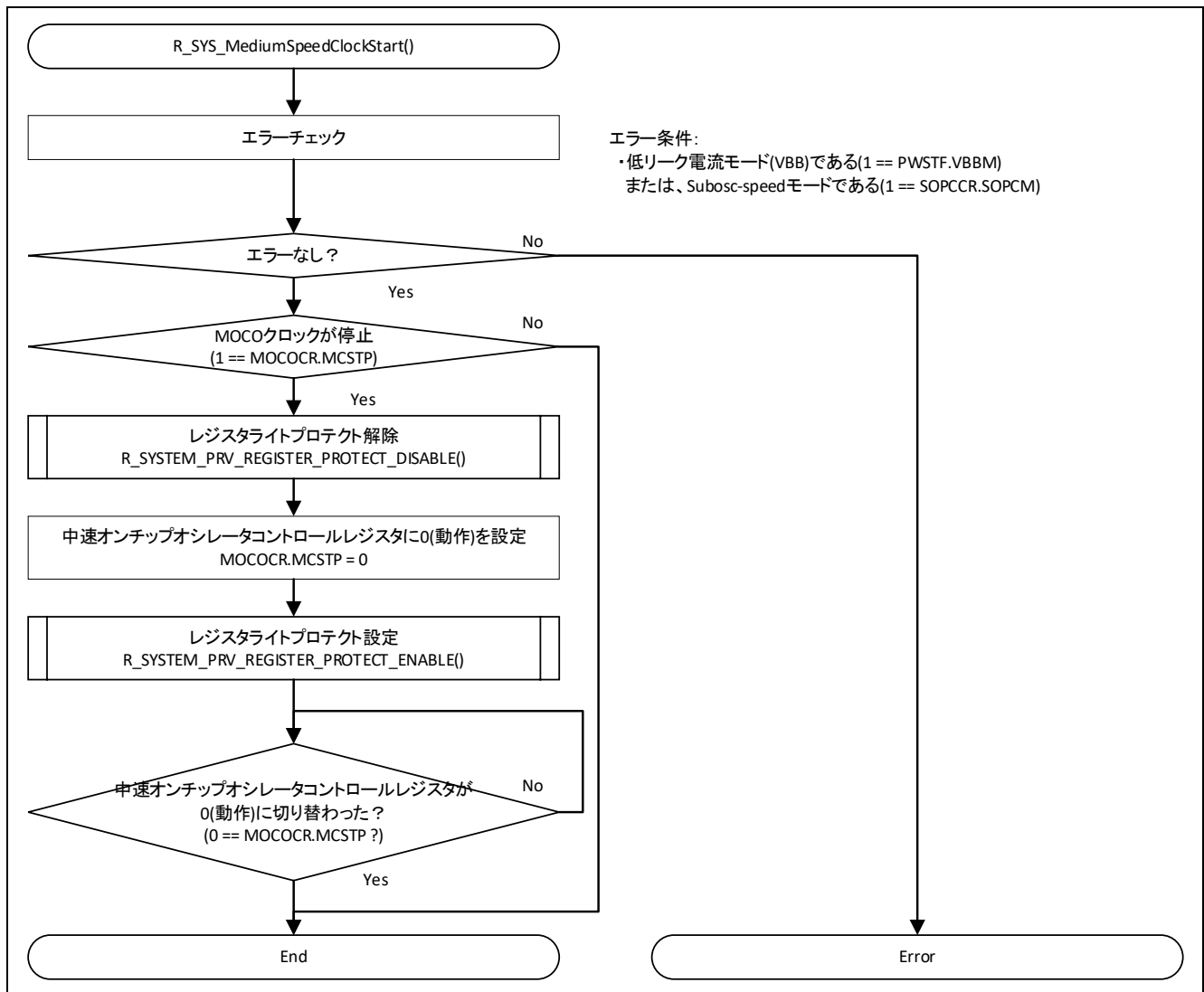


図 4.27 R_SYS_MediumSpeedClockStart 関数処理フロー

4.3.22 R_SYS_MediumSpeedClockStop 関数

表 4-31 R_SYS_MediumSpeedClockStop 関数仕様

書式	int32_t R_SYS_MediumSpeedClockStop(void)
仕様説明	中速オンチップオシレータの動作を停止します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

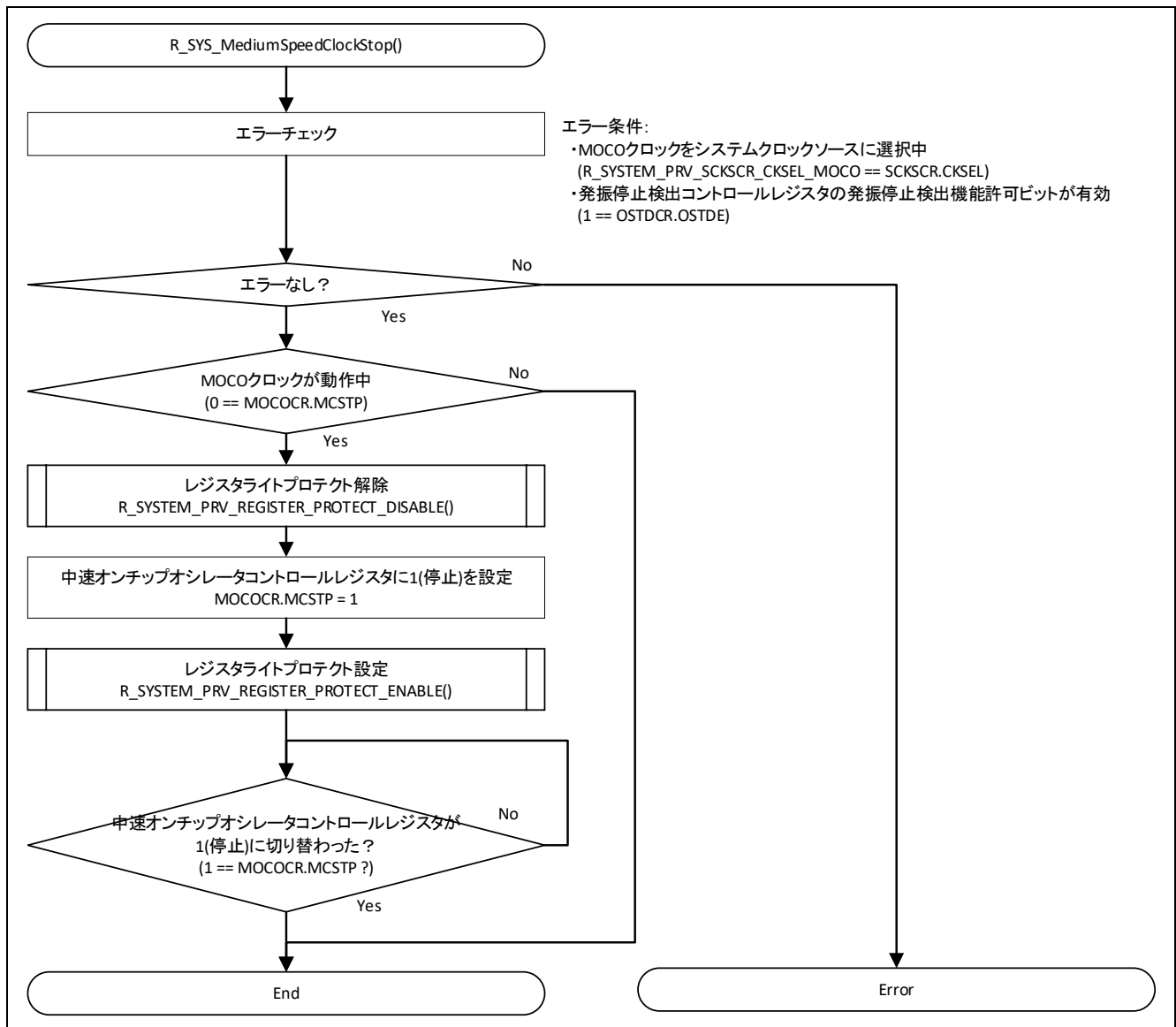


図 4.28 R_SYS_MediumSpeedClockStop 関数処理フロー

4.3.23 R_SYS_LowSpeedClockStart 関数

表 4-32 R_SYS_LowSpeedClockStart 関数仕様

書式	void R_SYS_LowSpeedClockStart(void)
仕様説明	低速オンチップオシレータの動作を開始します。
引数	なし
戻り値	なし
備考	—

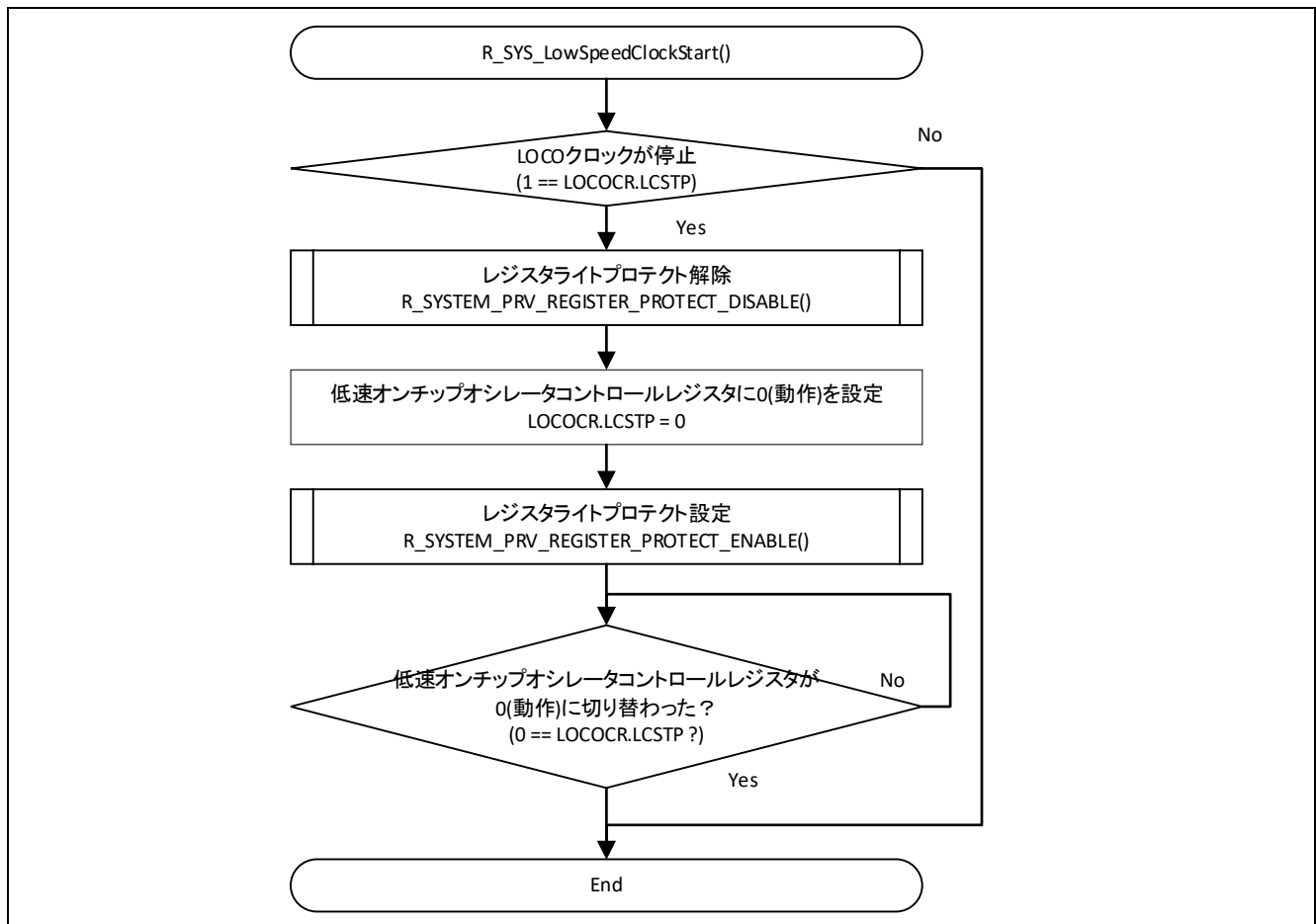


図 4.29 R_SYS_LowSpeedClockStart 関数処理フロー

4.3.24 R_SYS_LowSpeedClockStop 関数

表 4-33 R_SYS_LowSpeedClockStop 関数仕様

書式	int32_t R_SYS_LowSpeedClockStop(void)
仕様説明	低速オンチップオシレータの動作を停止します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

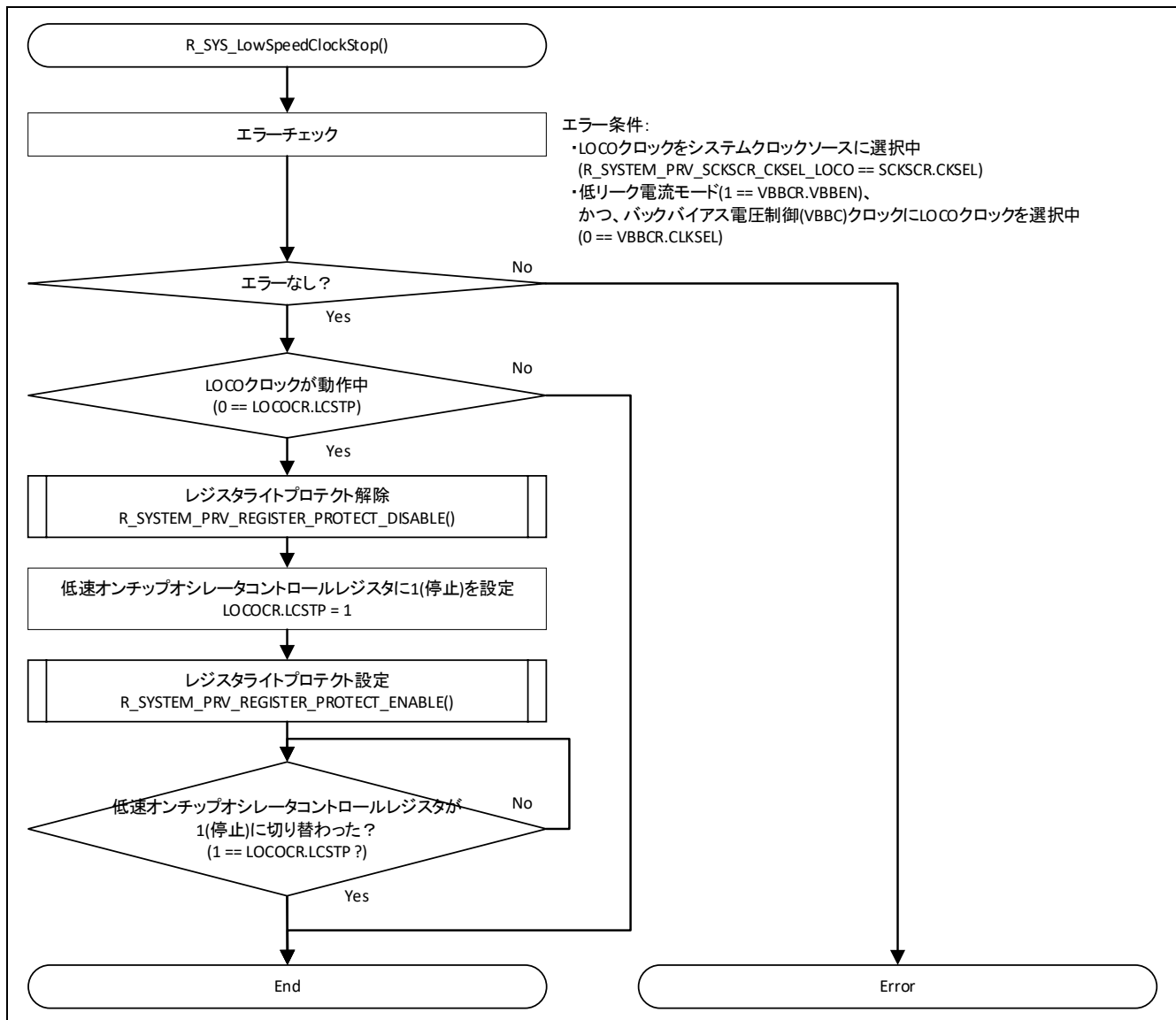


図 4.30 R_SYS_LowSpeedClockStop 関数処理フロー

4.3.25 R_SYS_SubOscSpeedClockStart 関数

表 4-34 R_SYS_SubOscSpeedClockStart 関数仕様

書式	void R_SYS_SubOscSpeedClockStart(void)
仕様説明	サブクロック発信器の動作を開始します。
引数	なし
戻り値	なし
備考	—

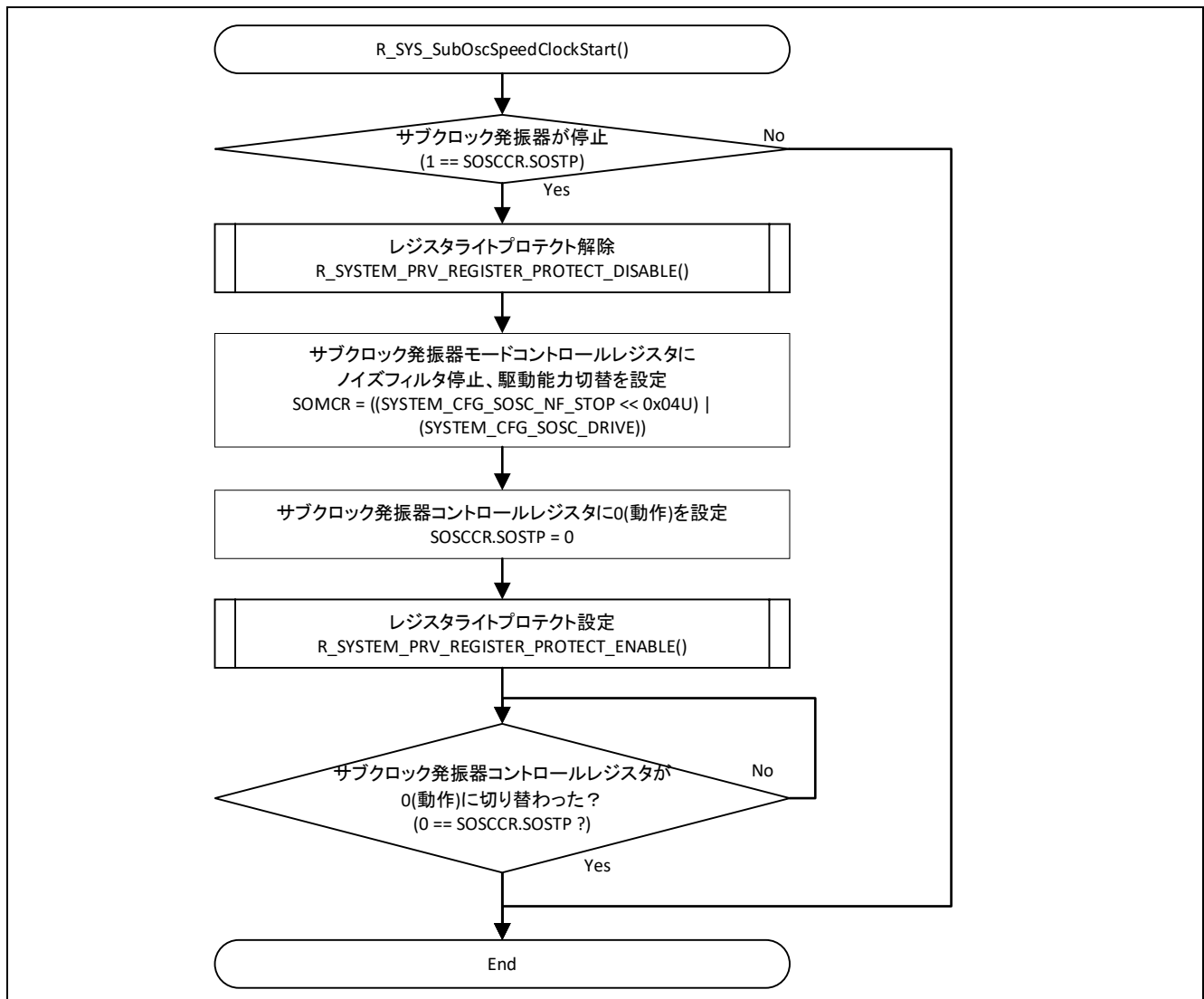


図 4.31 R_SYS_SubOscSpeedClockStart 関数処理フロー

4.3.26 R_SYS_SubOscSpeedClockStop 関数

表 4-35 R_SYS_SubOscSpeedClockStop 関数仕様

書式	int32_t R_SYS_SubOscSpeedClockStop(void)
仕様説明	サブクロック発信器の動作を停止します。
引数	なし
戻り値	正常 (0) 異常 (-1)
備考	—

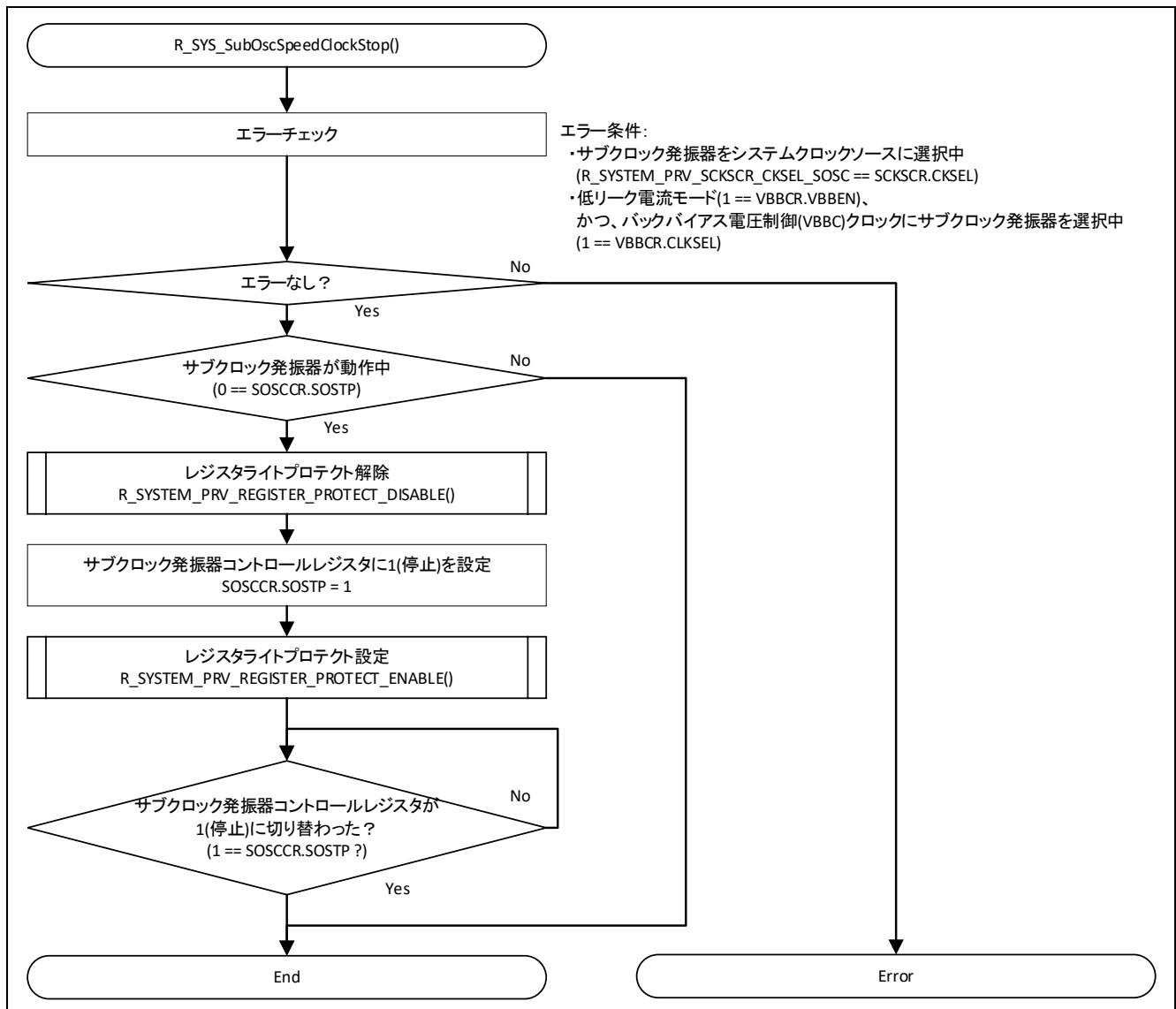


図 4.32 R_SYS_SubOscSpeedClockStop 関数処理フロー

4.3.27 R_SYS_PLLSpeedClockStart 関数

表 4-36 R_SYS_PLLSpeedClockStart 関数仕様

書式	int32_t R_SYS_PLLSpeedClockStart(void)
仕様説明	PLL 回路の動作を開始します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

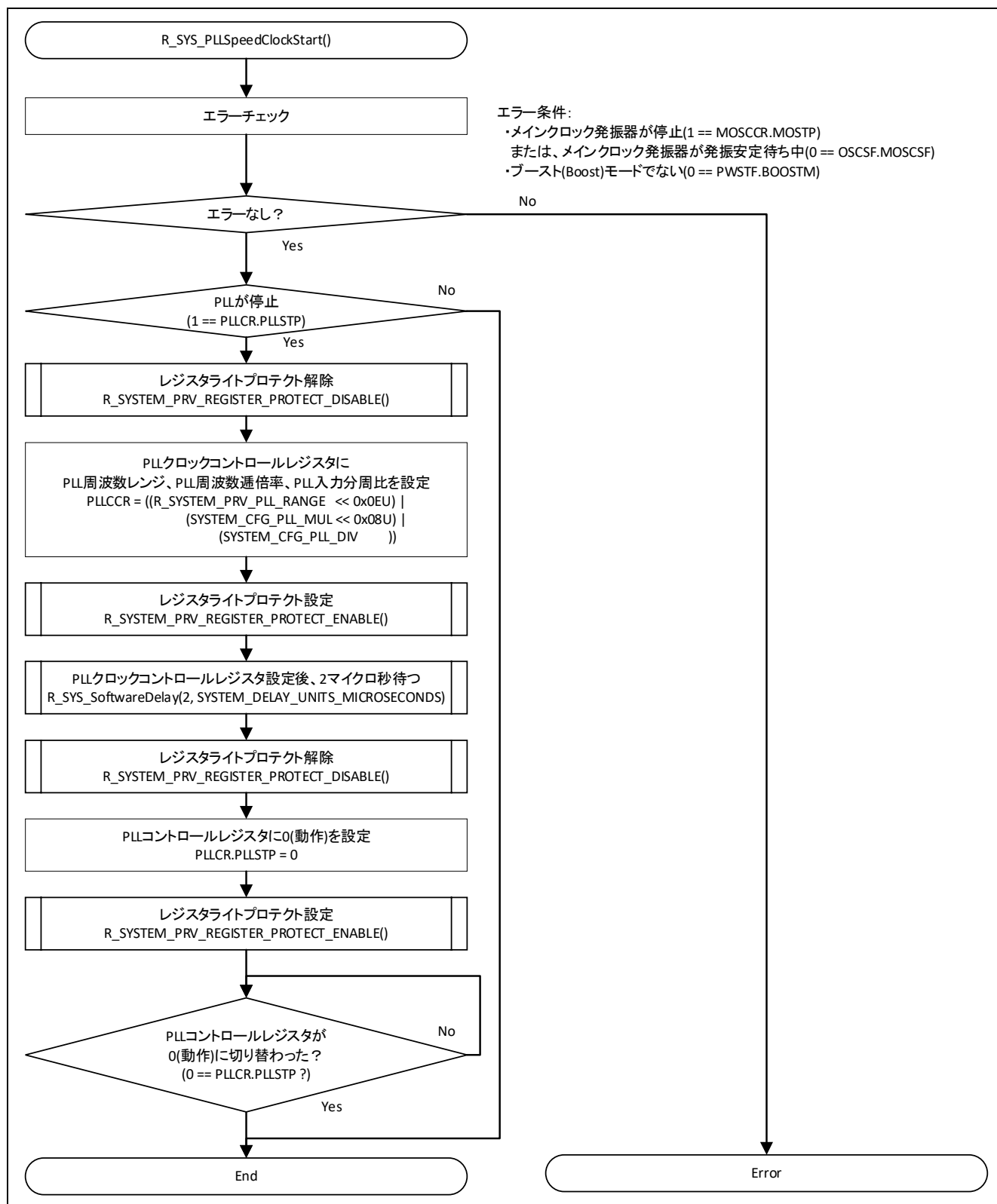


図 4.33 R_SYS_PLLSpeedClockStart 関数処理フロー

4.3.28 R_SYS_PLLSpeedClockStop 関数

表 4-37 R_SYS_PLLSpeedClockStop 関数

書式	int32_t R_SYS_PLLSpeedClockStop(void)
仕様説明	PLL 回路の動作を停止します。
引数	なし
戻り値	正常 (0)
	異常 (-1)
備考	—

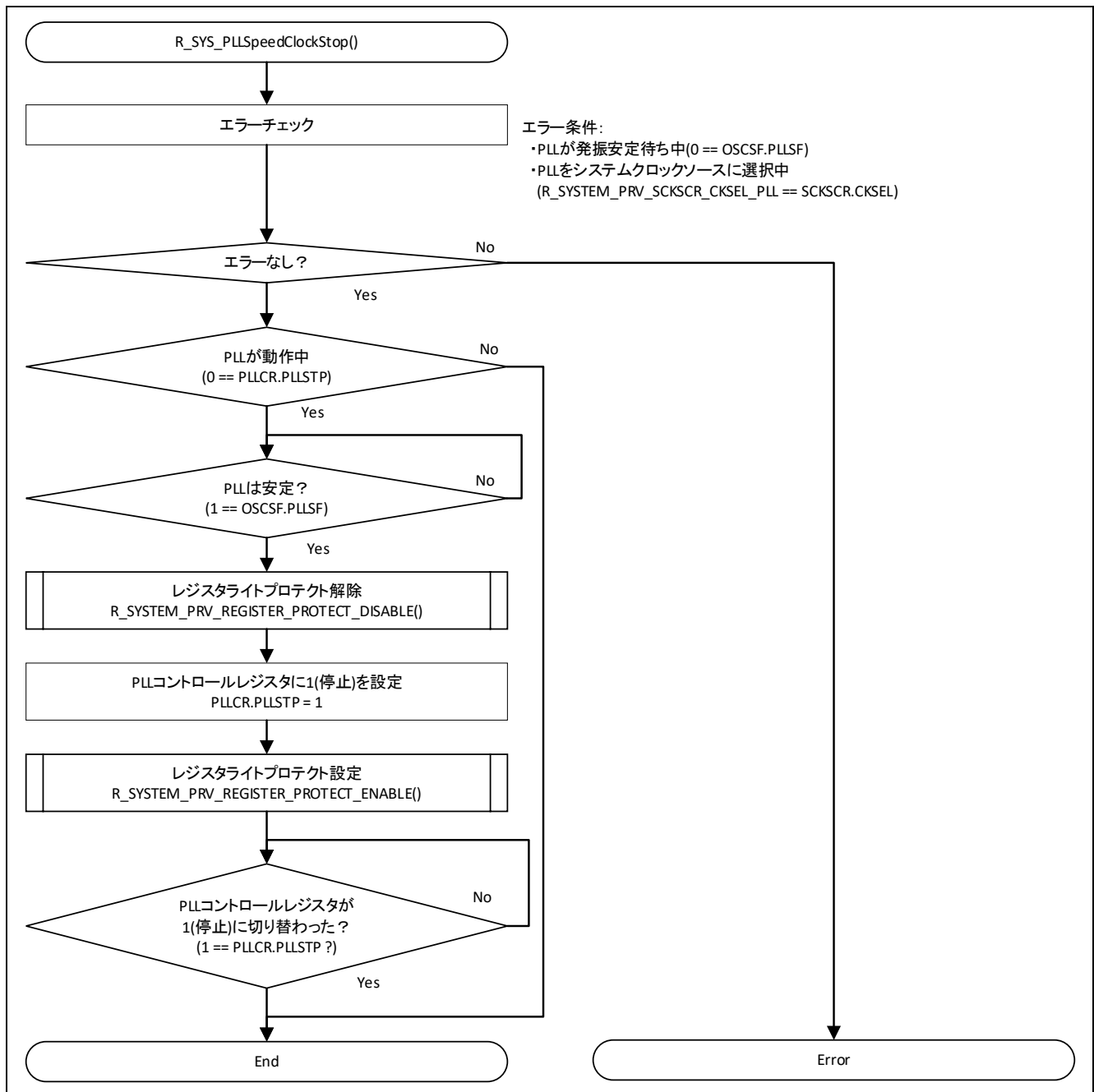


図 4.34 R_SYS_PLLSpeedClockStop 関数処理フロー

4.3.29 R_SYS_OscStabilizationFlagGet 関数

表 4-38 R_SYS_OscStabilizationFlagGet 関数

書式	uint8_t R_SYS_OscStabilizationFlagGet(void)
仕様説明	OSCSF レジスタの値を取得します。
引数	なし
戻り値	uint8_t : OSCSF レジスタの値を返します。
備考	—

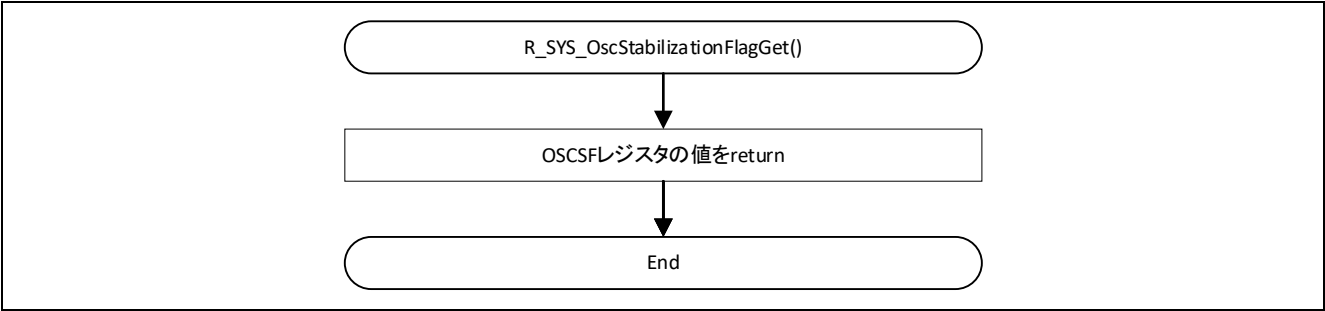


図 4.35 R_SYS_OscStabilizationFlagGet 関数処理フロー

4.3.30 R_SYS_IrqEventLinkSet 関数

表 4-39 R_SYS_IrqEventLinkSet 関数仕様

書式	int32_t R_SYS_IrqEventLinkSet(IRQn_Type irq, uint32_t iels_value, system_int_cb_t callback)
仕様説明	割り込みハンドラをコールバック関数として登録します。 このコールバック関数は、指定した IELx_IRQn 番号の割り込みハンドラから呼び出されます。
引数	IRQn_Type irq[入力] : イベントリンク番号(0~31)を設定します。 uint32_t iels_value[入力] : IELSRn.IELS レジスタにイベントリンク信号の値を設定します。 system_int_cb_t callback[入力] : コールバック関数を設定します。
戻り値	正常 (0) 異常 (-1)
備考	—

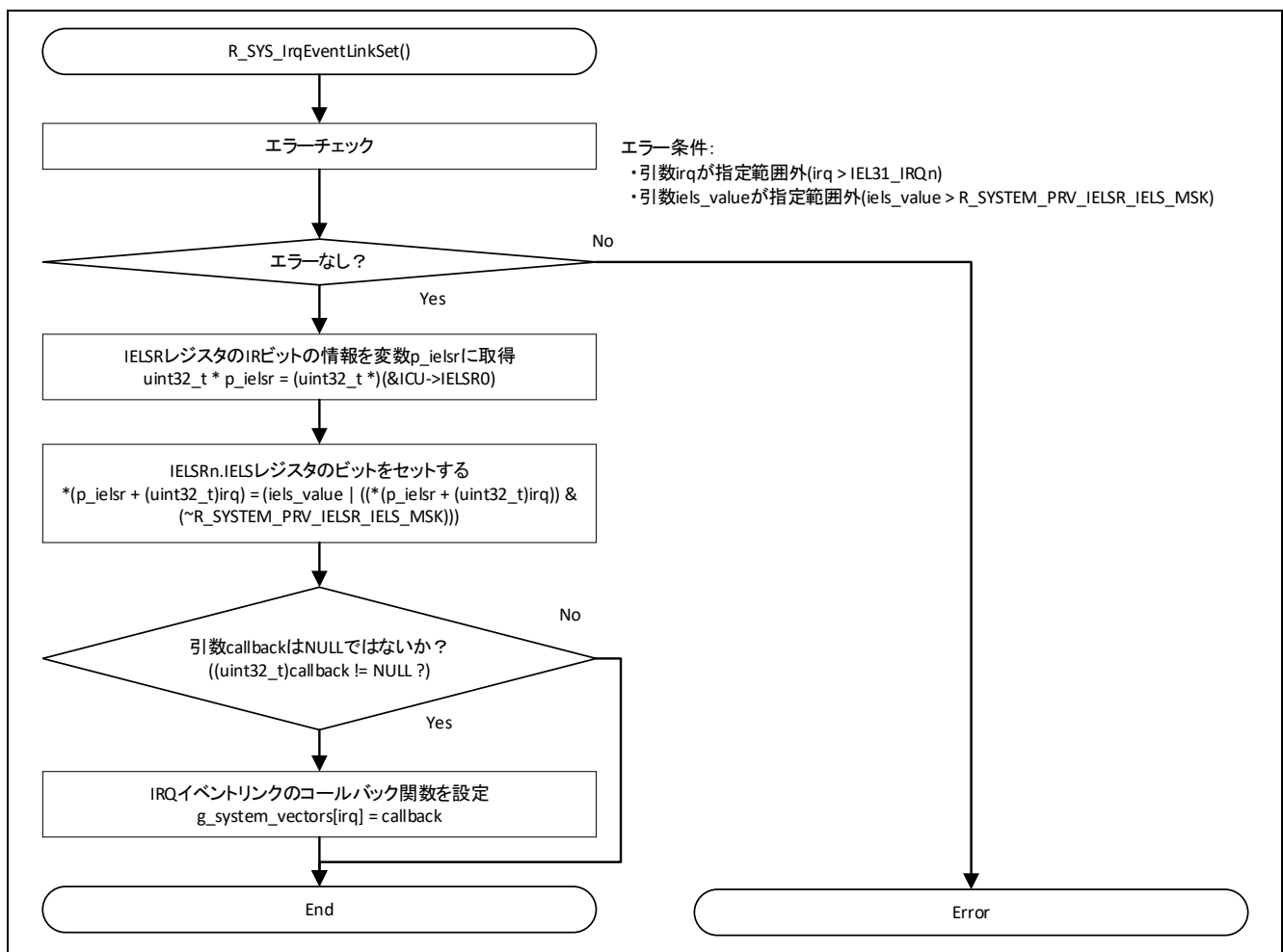


図 4.36 R_SYS_IrqEventLinkSet 関数処理フロー

4.3.31 R_SYS_IrqStatusGet 関数

表 4-40 R_SYS_IrqStatusGet 関数仕様

書式	int32_t R_SYS_IrqStatusGet(IRQn_Type irq, uint8_t * p_ir)
仕様説明	指定した IELx_IRQn 番号の IR フラグの状態を取得します。
引数	IRQn_Type irq[入力] : イベントリンク番号(0~31)を設定します。 uint8_t * p_ir[入力] : 取得した IR フラグの格納先を設定します。
戻り値	正常 (0) 異常 (-1)
備考	—

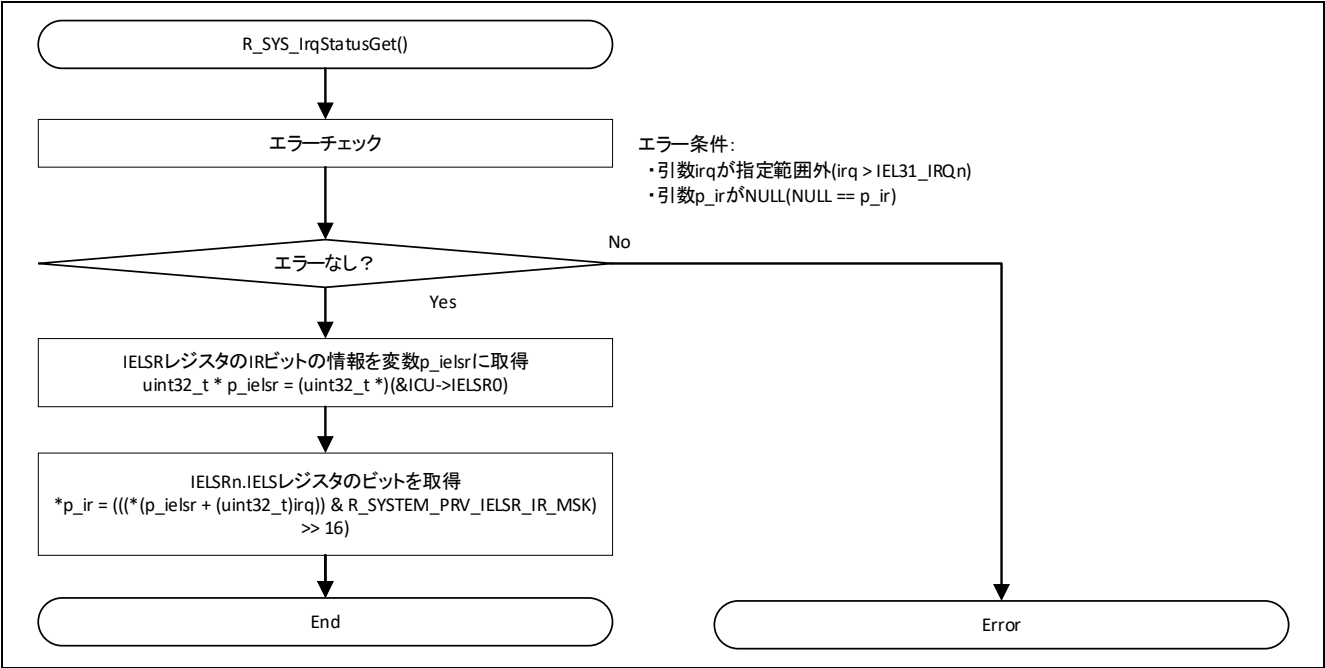


図 4.37 R_SYS_IrqStatusGet 関数処理フロー

4.3.32 R_SYS_IrqStatusClear 関数

表 4-41 R_SYS_IrqStatusClear 関数仕様

書式	int32_t R_SYS_IrqStatusClear(IRQn_Type irq)
仕様説明	指定した IELx_IRQn 番号の IR フラグの状態をクリアします。
引数	IRQn_Type irq[入力] : イベントリンク番号(0~31)を設定します。
戻り値	正常 (0)
	異常 (-1)
備考	—

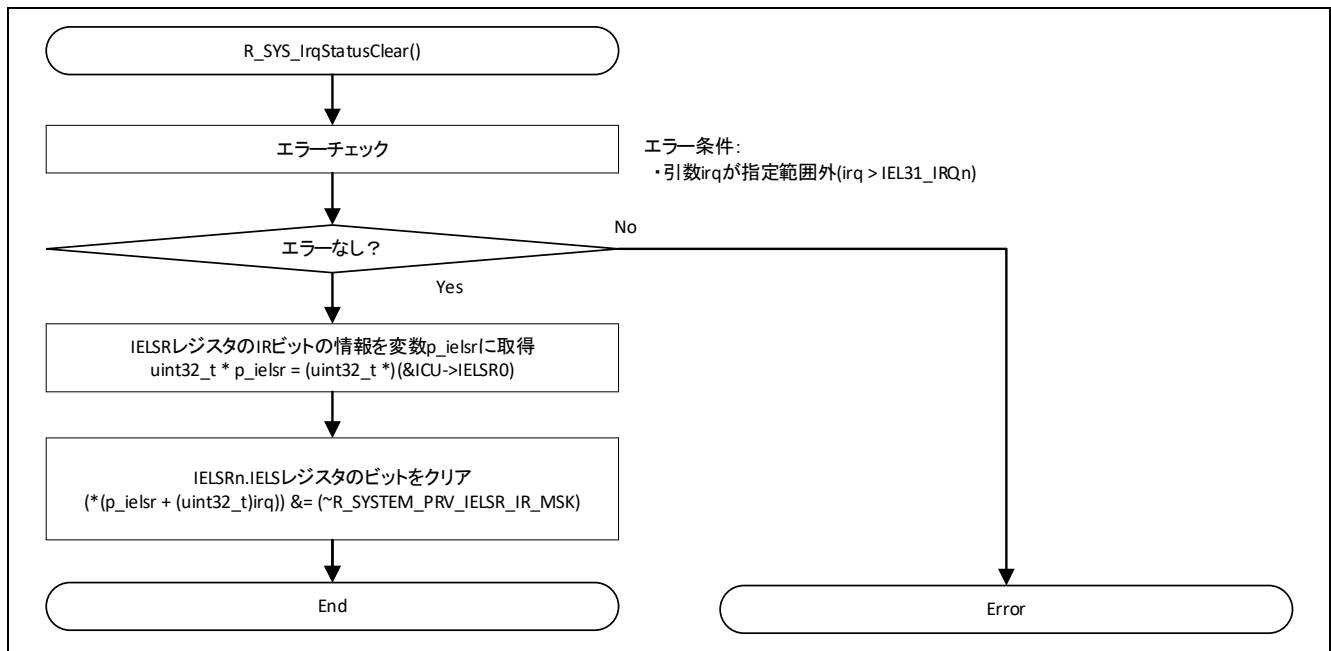


図 4.38 R_SYS_IrqStatusClear 関数処理フロー

4.3.33 R_SYS_EnterCriticalSection 関数

表 4-42 R_SYS_EnterCriticalSection 関数仕様

書式	void R_SYS_EnterCriticalSection(void)
仕様説明	割り込み禁止を有効にします。
引数	なし
戻り値	なし
備考	－

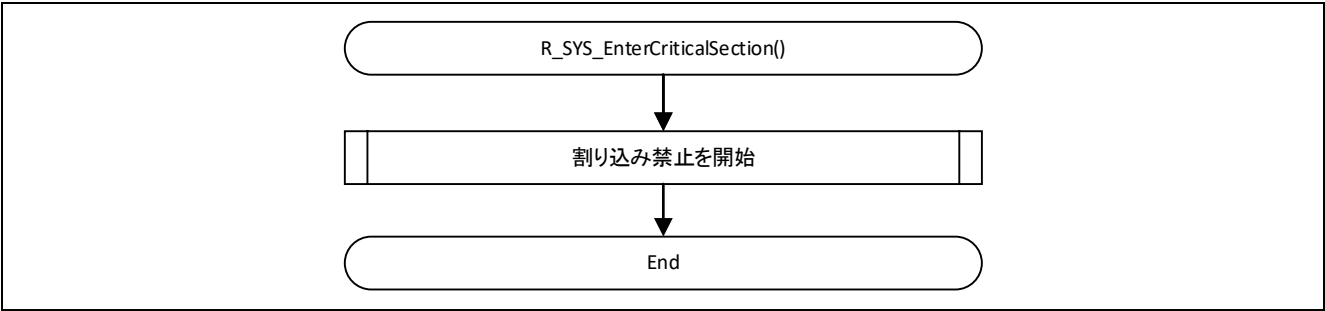


図 4.39 R_SYS_EnterCriticalSection 関数処理フロー

4.3.34 R_SYS_ExitCriticalSection 関数

表 4-43 R_SYS_ExitCriticalSection 関数仕様

書式	void R_SYS_ExitCriticalSection(void)
仕様説明	割り込み禁止を無効にします。
引数	なし
戻り値	なし
備考	－

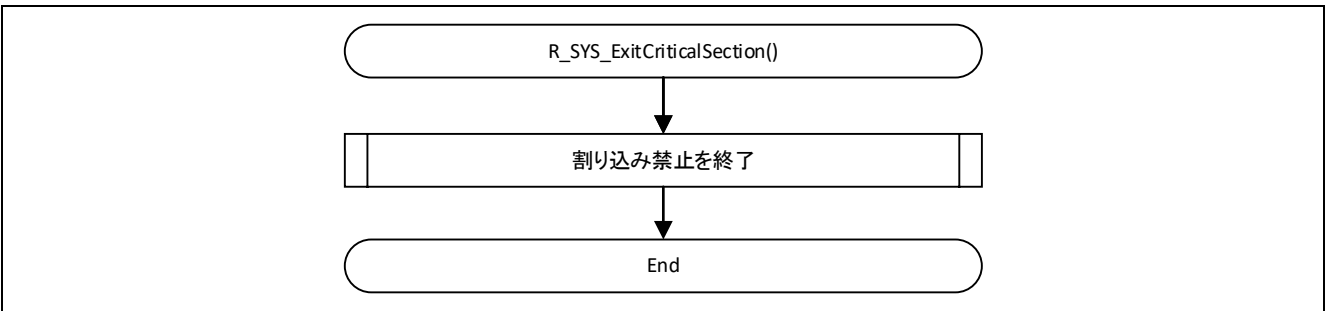


図 4.40 R_SYS_ExitCriticalSection 関数処理フロー

4.3.35 R_SYS_ResourceLock 関数

表 4-44 R_SYS_ResourceLock 関数仕様

書式	int32_t R_SYS_ResourceLock(e_system_mcu_lock_t hw_index)
仕様説明	ハードウェアリソースロックを設定します。
引数	e_system_mcu_lock_t hw_index[入力] : ハードウェアリソース番号を設定します。
戻り値	ロック成功 (0)
	ロック失敗 (-1)
備考	—

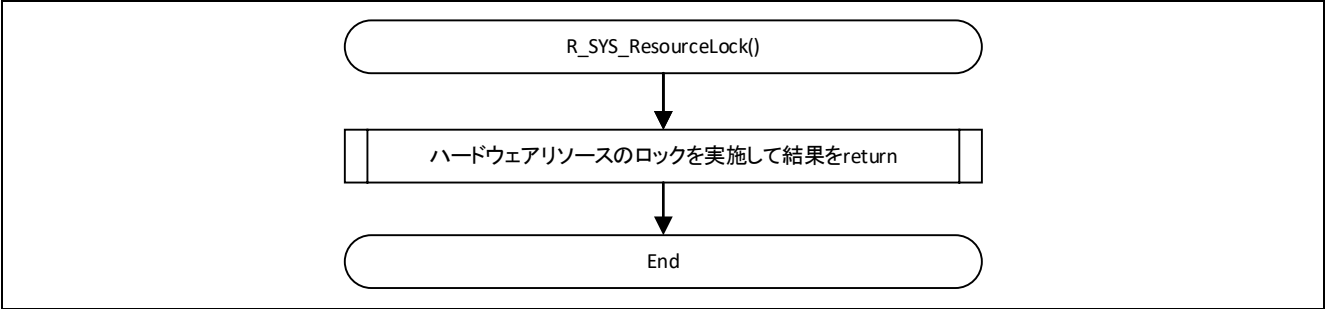


図 4.41 R_SYS_ResourceLock 関数処理フロー

4.3.36 R_SYS_ResourceUnlock 関数

表 4-45 R_SYS_ResourceUnlock 関数仕様

書式	void R_SYS_ResourceUnlock(e_system_mcu_lock_t hw_index)
仕様説明	ハードウェアリソースロックを解除します。
引数	e_system_mcu_lock_t hw_index[入力] : ハードウェアリソース番号を設定します。
戻り値	なし
備考	—

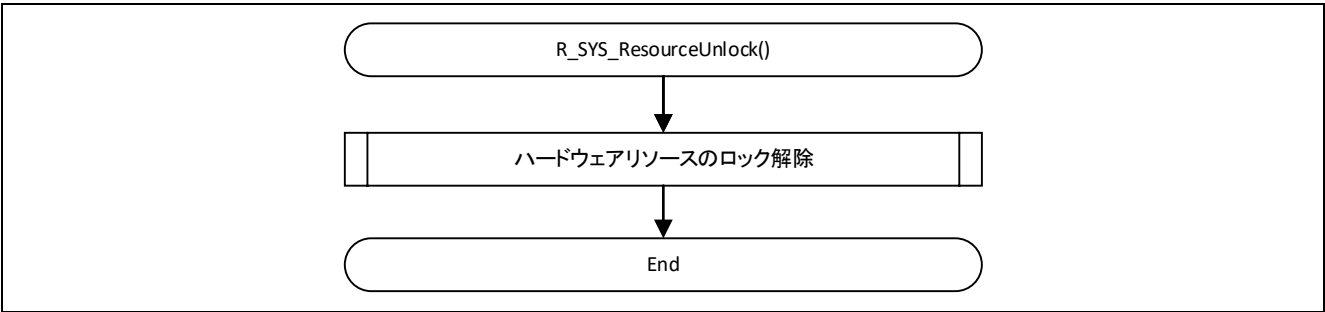


図 4.42 R_SYS_ResourceUnlock 関数処理フロー

4.3.37 R_SYS_RegisterProtectEnable 関数

表 4-46 R_SYS_RegisterProtectEnable 関数仕様

書式	void R_SYS_RegisterProtectEnable(e_system_reg_protect_t regs_to_protect)
仕様説明	レジスタプロテクションを有効にします。
引数	e_system_reg_protect_t regs_to_protect[入力] : レジスタプロテクション番号を設定します。
戻り値	なし
備考	—

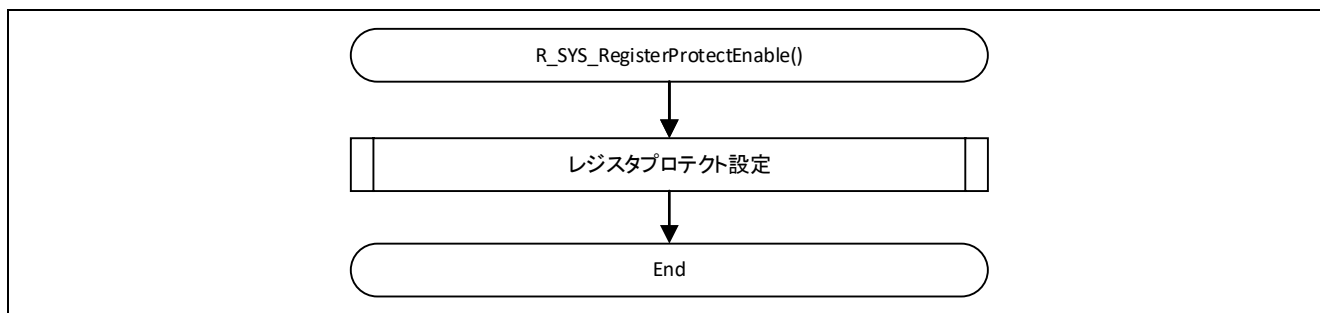


図 4.43 R_SYS_RegisterProtectEnable 関数処理フロー

4.3.38 R_SYS_RegisterProtectDisable 関数

表 4-47 R_SYS_RegisterProtectDisable 関数仕様

書式	void R_SYS_RegisterProtectDisable(e_system_reg_protect_t regs_to_unprotect)
仕様説明	レジスタプロテクションを無効にします。
引数	e_system_reg_protect_t regs_to_unprotect[入力] : レジスタプロテクション番号を設定します。
戻り値	なし
備考	—

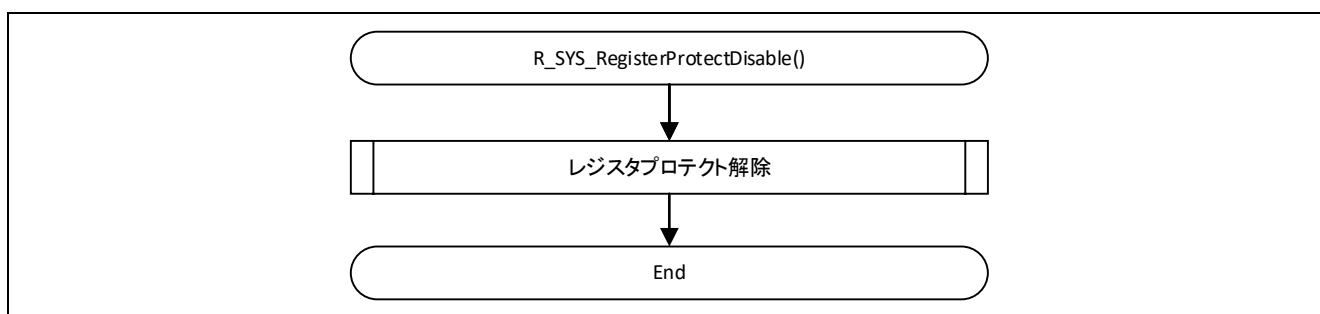


図 4.44 R_SYS_RegisterProtectDisable 関数処理フロー

4.3.39 R_SYS_SoftwareDelay 関数

表 4-48 R_SYS_SoftwareDelay 関数仕様

書式	void R_SYS_SoftwareDelay(uint32_t delay, e_system_delay_units_t units)
仕様説明	指定されたミリ秒、またはマイクロ秒のソフトウェアディレイを発生させます。
引数	uint32_t delay[入力] : ディレイ時間を設定します。
	e_system_delay_units_t units[入力] : ディレイ時間の単位(ミリ秒またはマイクロ秒)を設定します。
戻り値	なし
備考	—

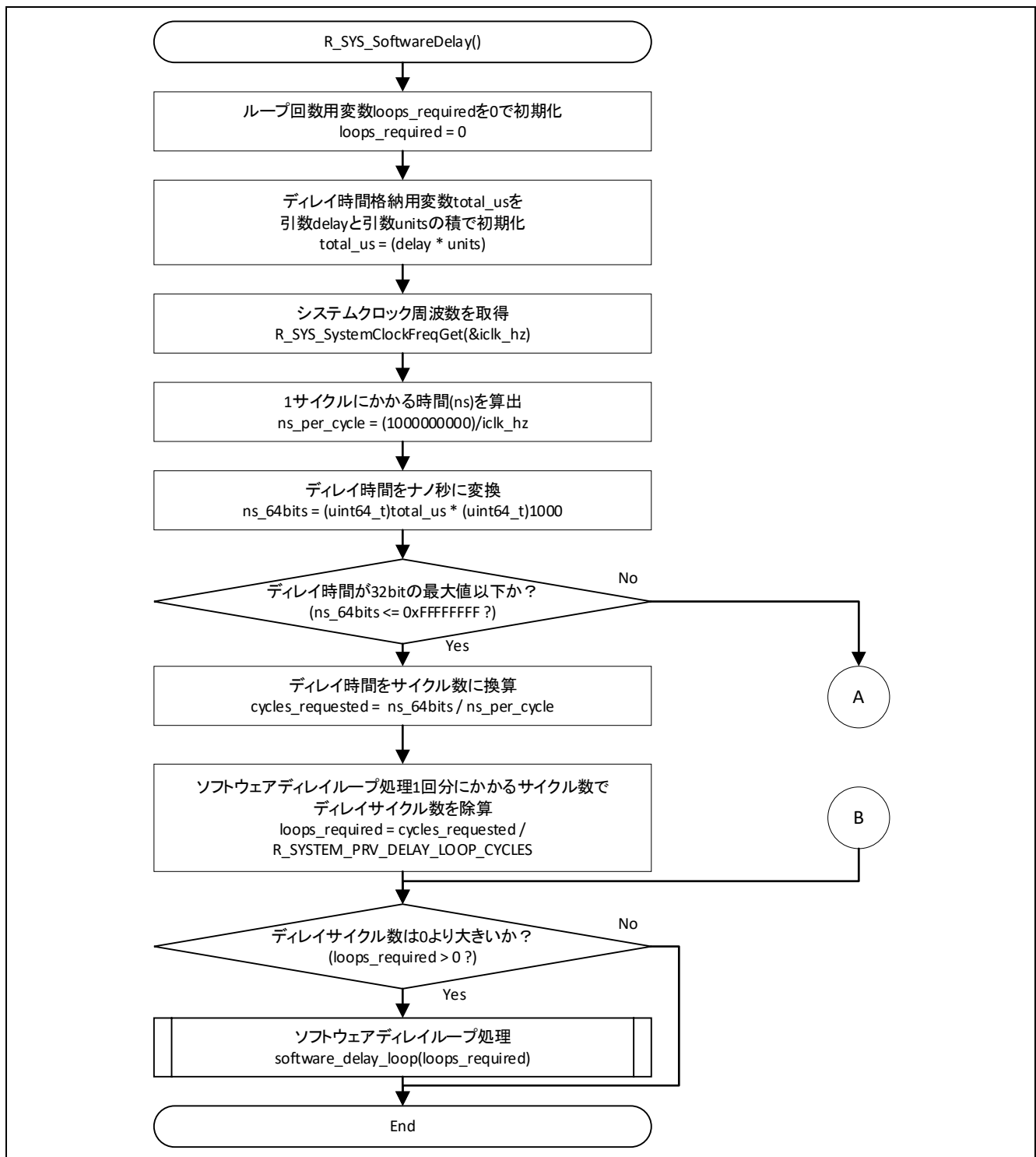


図 4.45 R_SYS_SoftwareDelay 関数処理フロー(1/2)

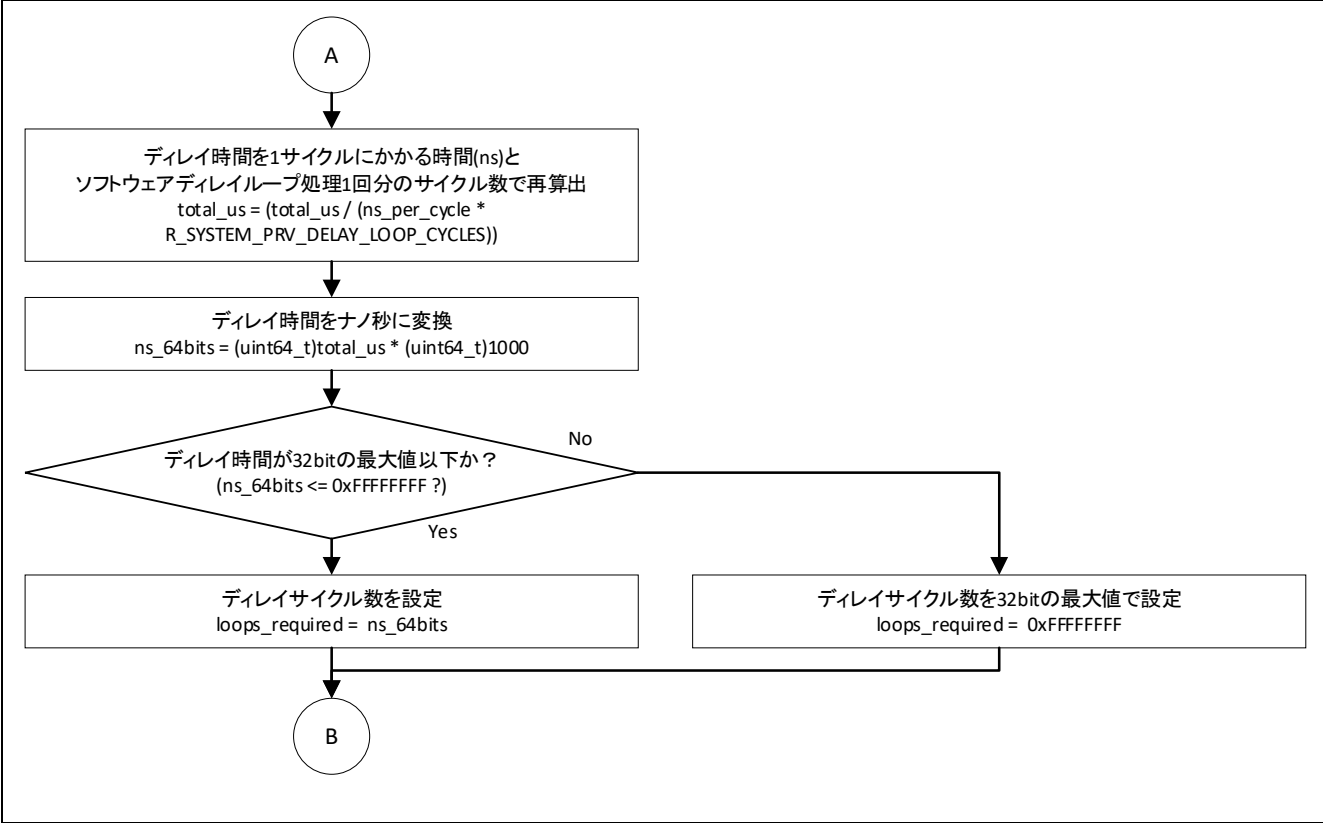


図 4.46 R_SYS_SoftwareDelay 関数処理フロー(2/2)

4.3.40 R_SYS_GetVersion 関数

表 4-49 R_SYS_GetVersion 関数仕様

書式	uint32_t R_SYS_GetVersion(void)
仕様説明	R_SYSTEM ドライバのバージョンを取得します。
引数	なし
戻り値	取得した R_SYSTEM ドライバのバージョン
備考	ー

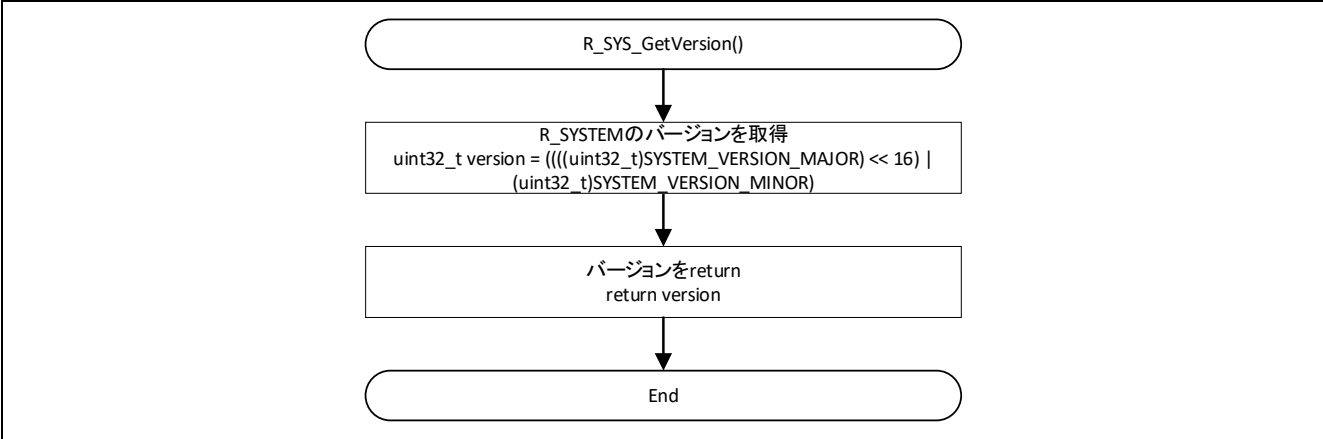


図 4.47 R_SYS_GetVersion 関数処理フロー

4.3.41 r_sys_BoostFlagGet 関数

表 4-50 r_sys_BoostFlagGet 関数仕様

書式	int32_t r_sys_BoostFlagGet(bool * boost_flg)
仕様説明	Boost モードへの遷移履歴を取得します。
引数	bool * boost_flg[入力] : 取得した遷移履歴の格納先を設定します。
戻り値	正常終了(0)
備考	boost_flg == true : 遷移履歴あり boost_flg == false : 遷移履歴なし

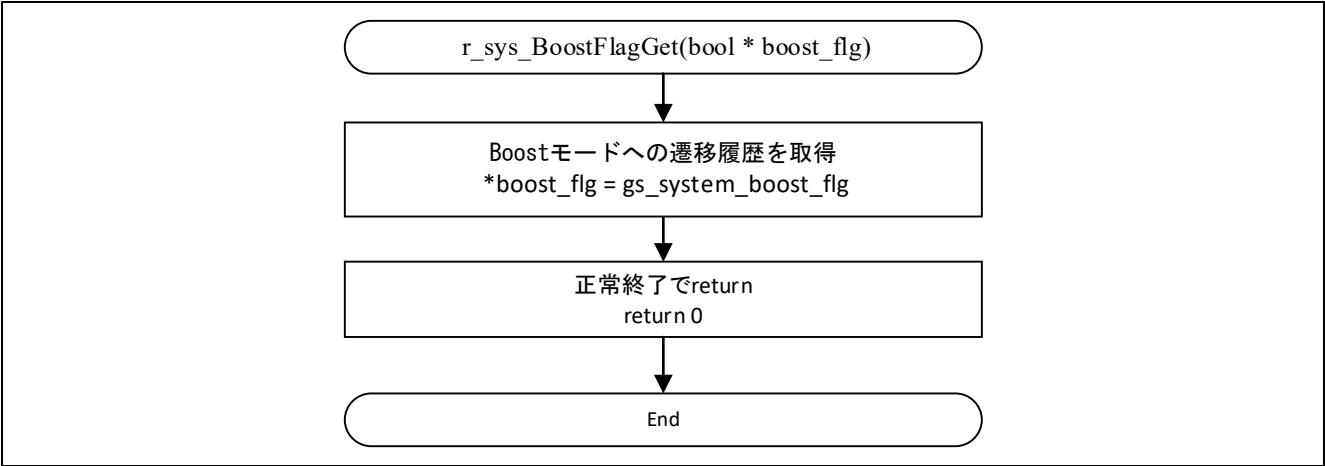


図 4.48 r_sys_BoostFlagGet 関数処理フロー

4.3.42 r_sys_BoostFlagSet 関数

表 4-51 r_sys_BoostFlagSet 関数仕様

書式	int32_t r_sys_BoostFlagSet(void)
仕様説明	Boost モードへの遷移履歴を設定します。
引数	なし
戻り値	正常終了(0)
備考	—

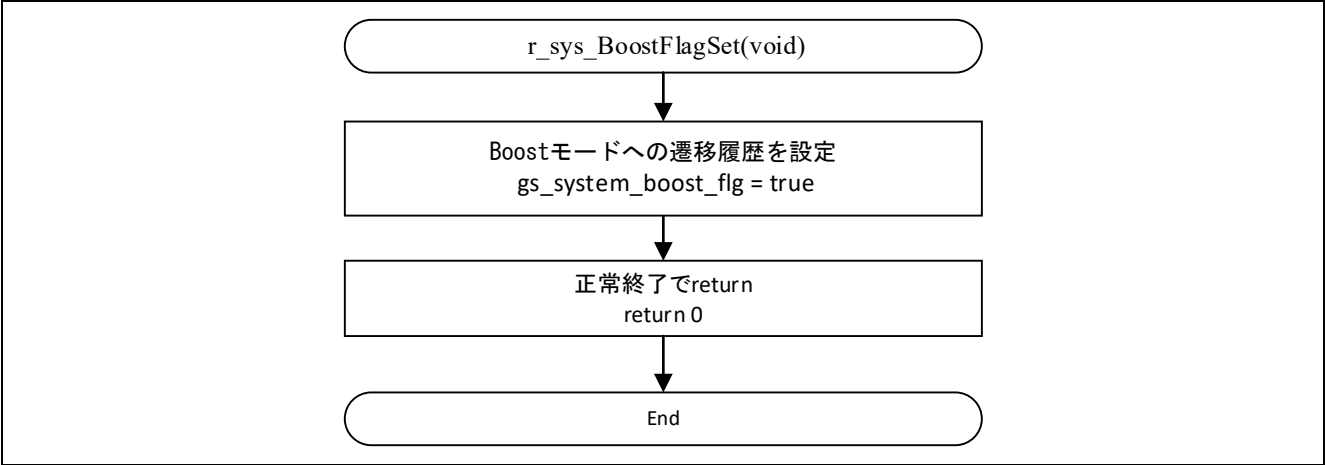


図 4.49 r_sys_BoostFlagSet 関数処理フロー

4.3.43 r_sys_BoostFlagClr 関数

表 4-52 r_sys_BoostFlagClr 関数仕様

書式	int32_t r_sys_BoostFlagClr(void)
仕様説明	Boost モードへの遷移履歴をクリアします。
引数	なし
戻り値	正常終了(0)
備考	—

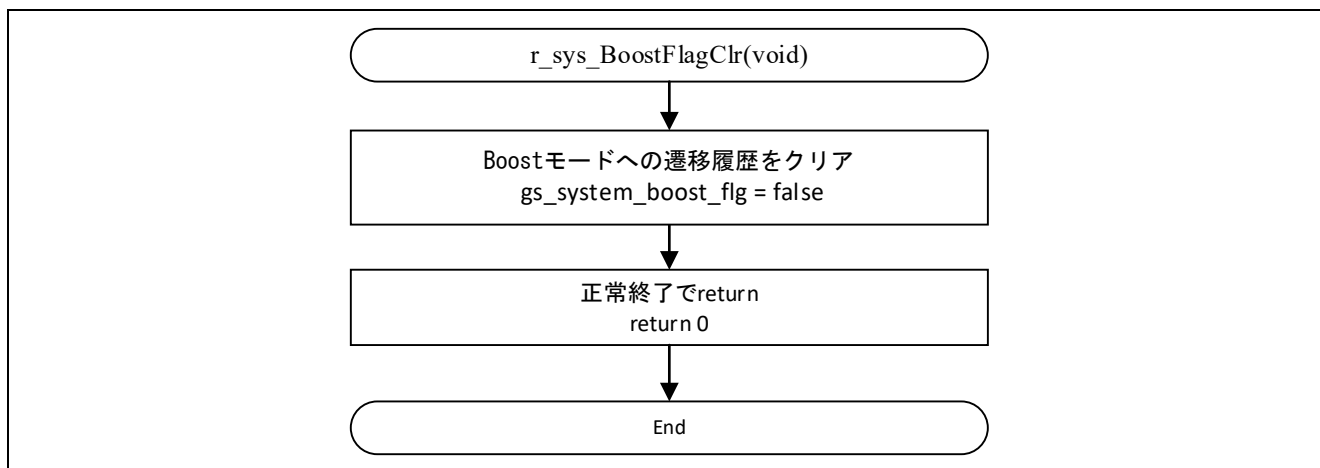


図 4.50 r_sys_BoostFlagClr 関数処理フロー

4.3.44 r_system_wdt_refresh 関数

表 4-53 r_system_wdt_refresh 関数仕様

書式	void r_system_wdt_refresh (void)
仕様説明	WDT のダウンカウンタをリフレッシュします。
引数	なし
戻り値	なし
備考	WEAK 関数として実装しています。同一名称の非 WEAK 関数を実装することで、本関数を無効にできます。

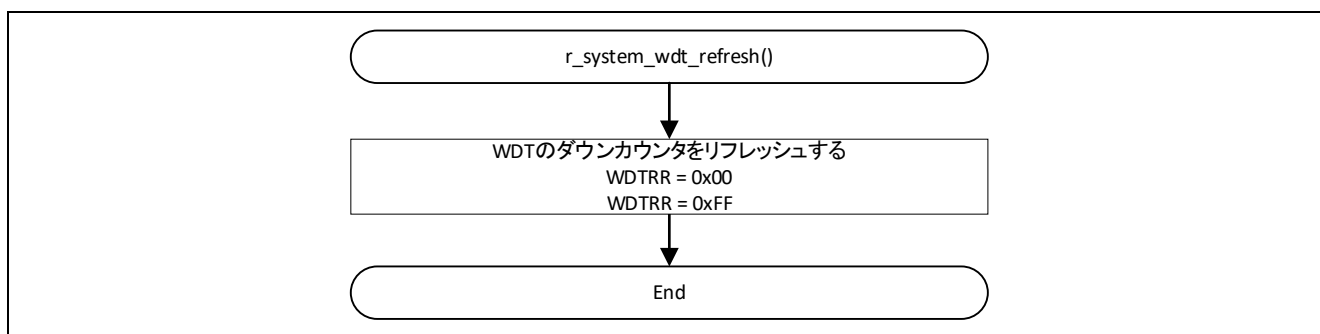


図 4.51 r_system_wdt_refresh 関数処理フロー

4.3.45 IELn_IRQHandler 関数(n=0~31)

表 4-54 IELn_IRQHandler 関数仕様

書式	void IELn_IRQHandler(void)
仕様説明	イベントリンクによって定義された IRQ 割り込みハンドラを実行します。
引数	なし
戻り値	なし
備考	ー

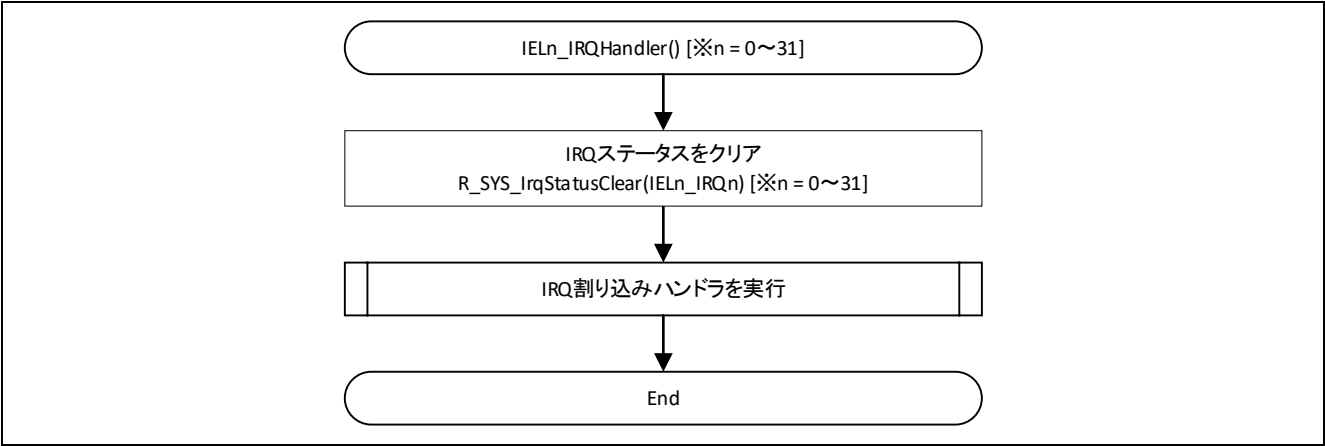


図 4.52 IELn_IRQHandler 関数処理フロー

4.3.46 R_NVIC_EnableIRQ 関数

表 4-55 R_NVIC_EnableIRQ 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_EnableIRQ(IRQn_Type IRQn)
仕様説明	Cortex-M0+ に定義されている NVIC の IRQ 番号に対応した割り込みを有効化します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	なし
備考	RAM 実行時に使用することで割り込みを有効化します。(Inline 展開される)

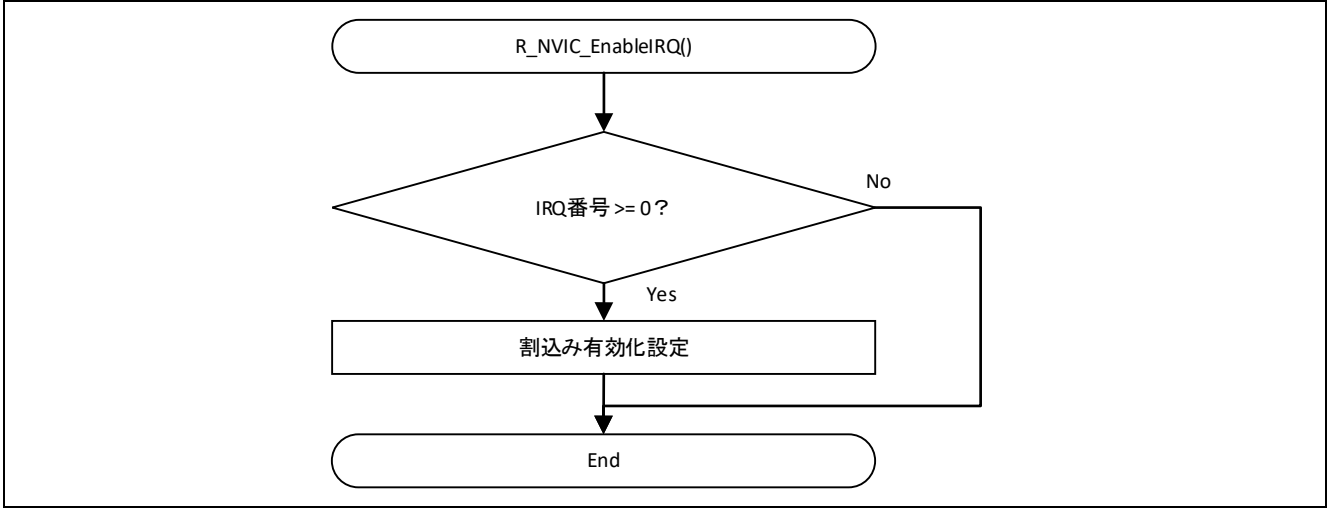


図 4.53 R_NVIC_EnableIRQ 関数処理フロー

4.3.47 R_NVIC_GetEnableIRQ 関数

表 4-56 R_NVIC_GetEnableIRQ 関数仕様

書式	__STATIC_FORCEINLINE uint32_t R_NVIC_GetEnableIRQ(IRQn_Type IRQn)
仕様説明	Cortex-M0+ に定義されている NVIC の IRQ 番号に対応した割込み設定を取得します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	無効化 (0)
	有効化 (1)
備考	RAM 実行時に使用することで割込み設定を取得します。(Inline 展開される)

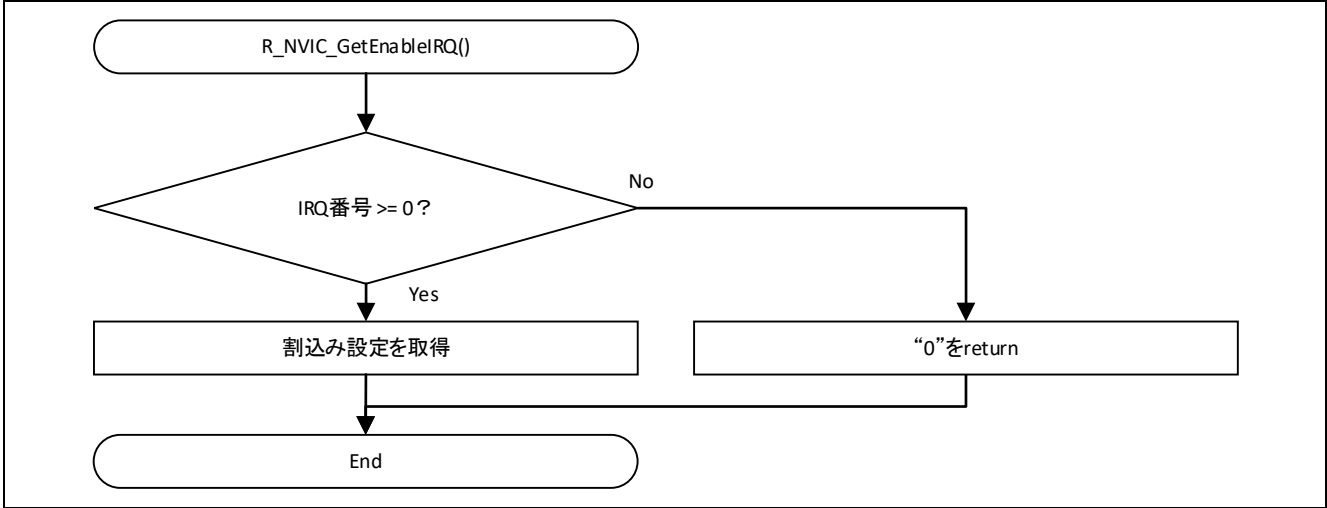


図 4.54 R_NVIC_GetEnableIRQ 関数処理フロー

4.3.48 R_NVIC_DisableIRQ 関数

表 4-57 R_NVIC_DisableIRQ 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_DisableIRQ(IRQn_Type IRQn)
仕様説明	Cortex-M0+ に定義されている NVIC の IRQ 番号に対応した割込みを無効化します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	なし
備考	RAM 実行時に使用することで割込みを無効化します。(Inline 展開される)

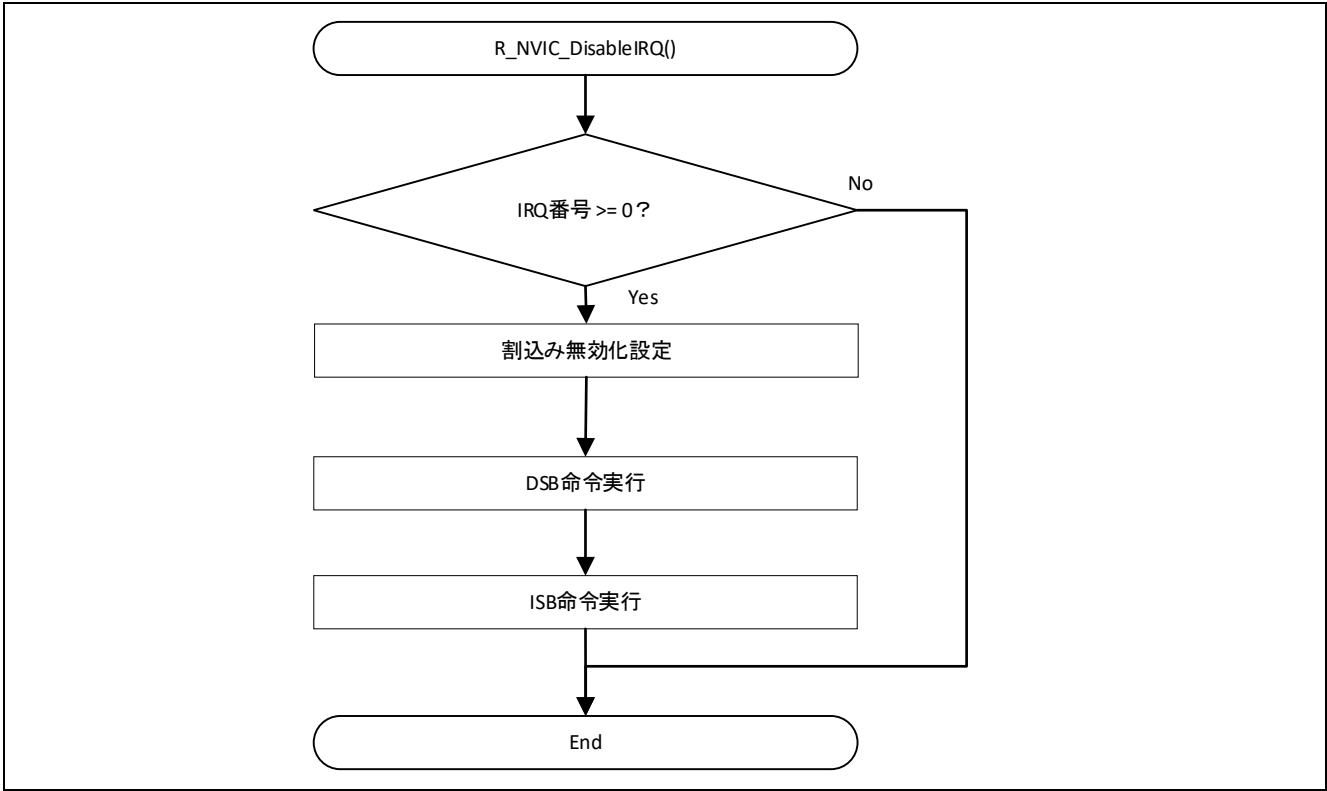


図 4.55 R_NVIC_DisableIRQ 関数処理フロー

4.3.49 R_NVIC_GetPendingIRQ 関数

表 4-58 R_NVIC_GetPendingIRQ 関数仕様

書式	__STATIC_FORCEINLINE uint32_t R_NVIC_GetPendingIRQ(IRQn_Type IRQn)
仕様説明	Cortex-M0+に定義されている NVIC の IRQ 番号に対応した割込みの保留状態を取得します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	割込み保留なし (0)
	割込み保留中 (1)
備考	RAM 実行時に使用することで割込み保留状態を取得します。(Inline 展開される)

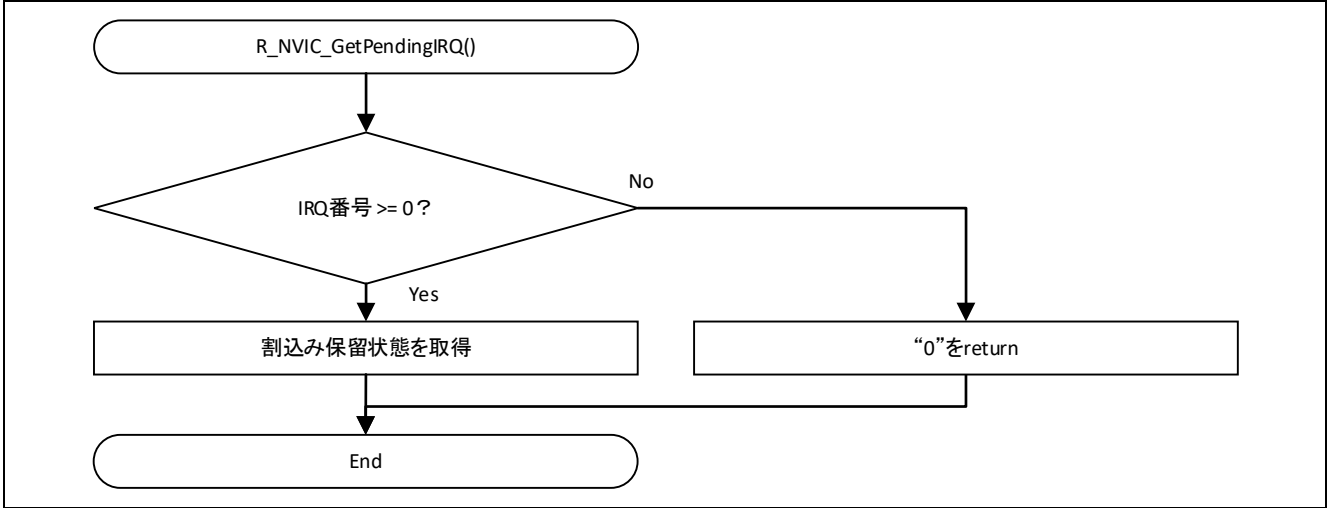


図 4.56 R_NVIC_GetPendingIRQ 関数処理フロー

4.3.50 R_NVIC_SetPendingIRQ 関数

表 4-59 R_NVIC_SetPendingIRQ 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_SetPendingIRQ(IRQn_Type IRQn)
仕様説明	Cortex-M0+に定義されている NVIC の IRQ 番号に対応した割込みの保留設定を有効化します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	なし
備考	RAM 実行時に使用することで割込みの保留設定を有効化します。(Inline 展開される)

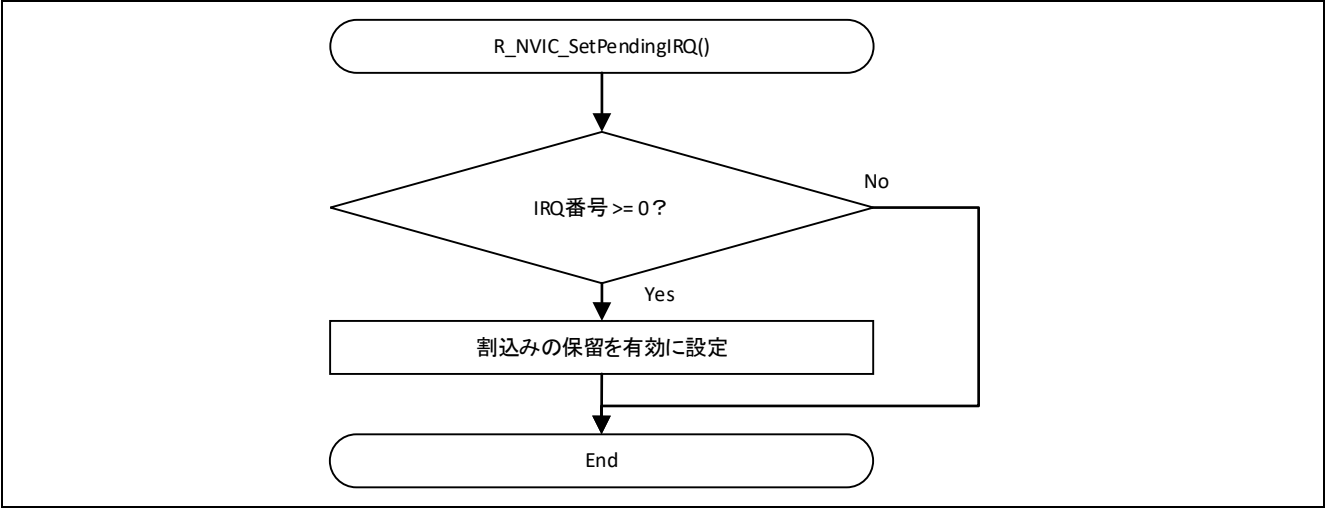


図 4.57 R_NVIC_SetPendingIRQ 関数処理フロー

4.3.51 R_NVIC_ClearPendingIRQ 関数

表 4-60 R_NVIC_ClearPendingIRQ 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_ClearPendingIRQ(IRQn_Type IRQn)
仕様説明	Cortex-M0+に定義されている NVIC の IRQ 番号に対応した割込みの保留状態をクリアします。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	なし
備考	RAM 実行時に使用することで割込みの保留状態をクリアします。(Inline 展開される)

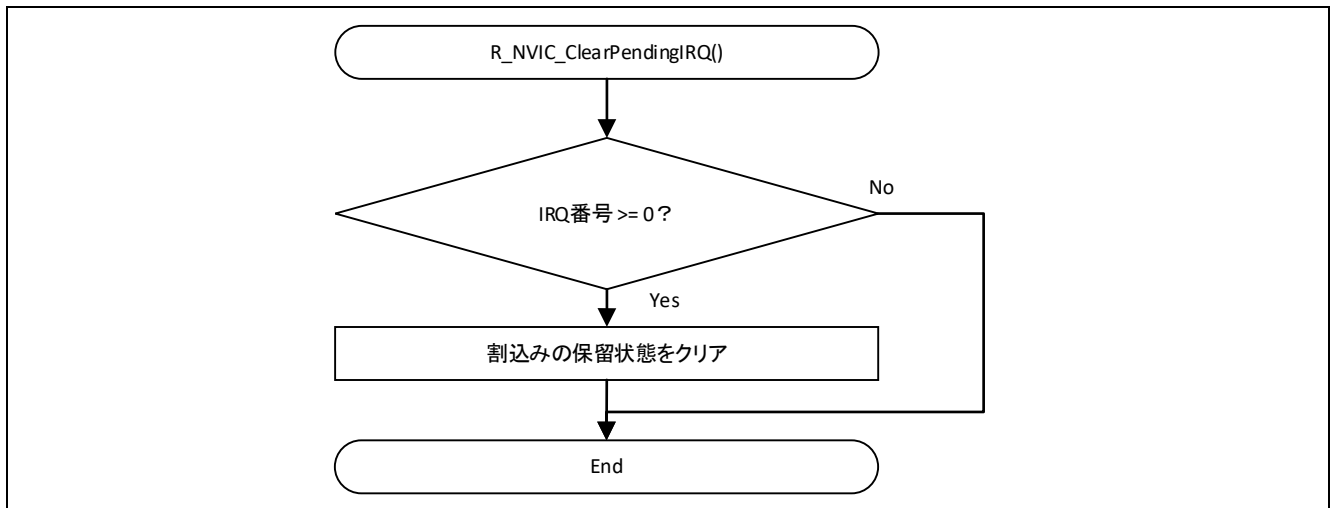


図 4.58 R_NVIC_ClearPendingIRQ 関数処理フロー

4.3.52 R_NVIC_SetPriority 関数

表 4-61 R_NVIC_SetPriority 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_SetPriority(IRQn_Type IRQn, uint32_t priority)
仕様説明	Cortex-M0+に定義されている NVIC の IRQ 番号に対応した割込みの優先度、またはシステムハンドラ優先度を設定します。
引数	IRQn_Type IRQn[入力] : IRQ 番号を設定します。 uint32_t priority[入力] : 割込みの優先度を設定します。
戻り値	なし
備考	RAM 実行時に使用することで割込みの優先度を設定します。(Inline 展開される) 割込みの優先度は、値が小さいほど対応する優先度は高くなります。

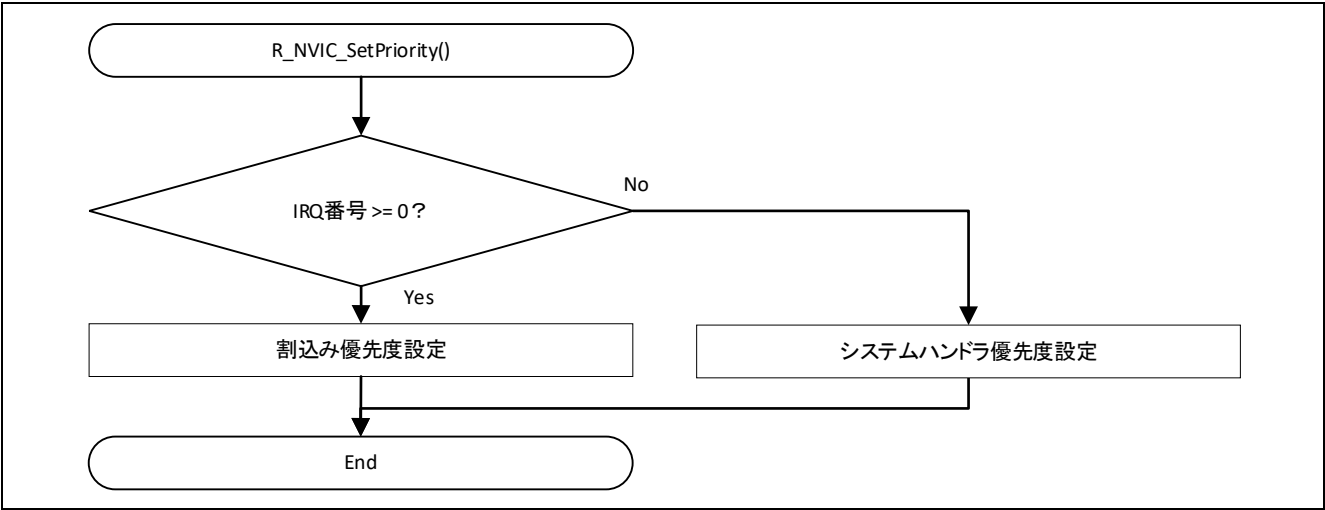


図 4.59 R_NVIC_SetPriority 関数処理フロー

4.3.53 R_NVIC_GetPriority 関数

表 4-62 R_NVIC_GetPriority 関数仕様

書式	__STATIC_FORCEINLINE uint32_t R_NVIC_GetPriority(IRQn_Type IRQn)
仕様説明	Cortex-M0+に定義されている NVIC の IRQ 番号に対応した割込みの優先度、またはシステムハンドラ優先度を取得します。
引数	IRQn_Type IRQn[入力] : IRQ 番号を設定します。
戻り値	割込みの優先度
備考	RAM 実行時に使用することで割込みの優先度を取得します。(Inline 展開される) 割込みの優先度は、値が小さいほど対応する優先度は高くなります。

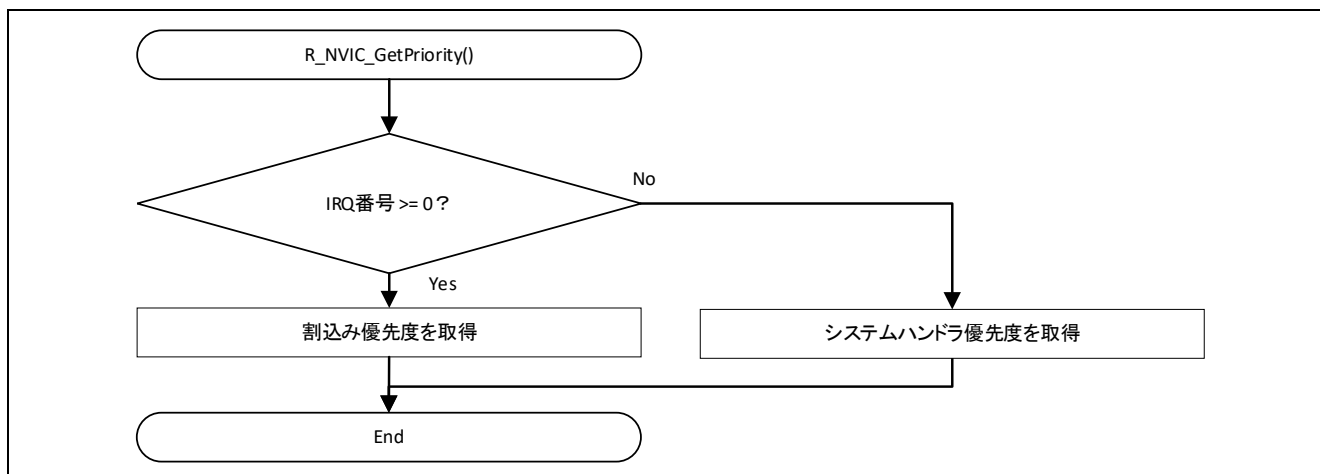


図 4.60 R_NVIC_GetPriority 関数処理フロー

4.3.54 R_NVIC_SetVector 関数

表 4-63 R_NVIC_SetVector 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_SetVector(IRQn_Type IRQn, uint32_t vector)
仕様説明	ベクタテーブルのベースアドレスからのオフセットアドレスを設定します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。 uint32_t vector[入力] : オフセットアドレスを設定します。
戻り値	なし
備考	RAM 実行時に使用することでオフセットアドレスを設定します。(Inline 展開される)

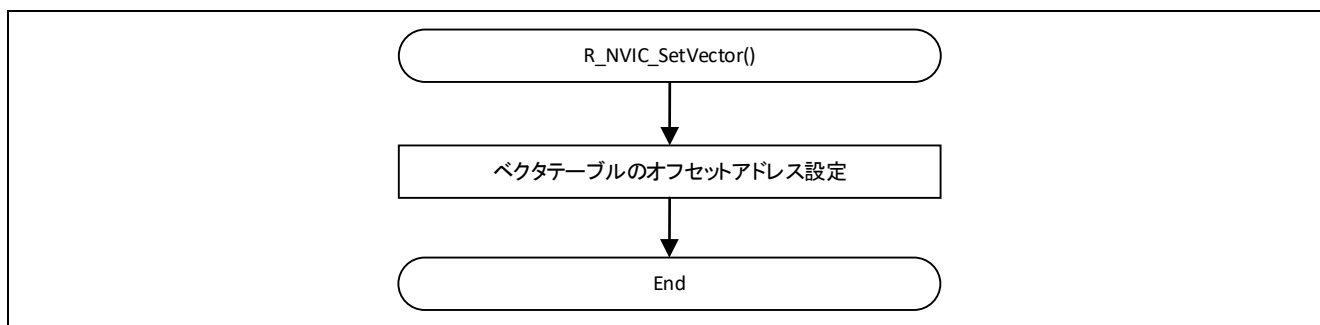


図 4.61 R_NVIC_SetVector 関数処理フロー

4.3.55 R_NVIC_GetVector 関数

表 4-64 R_NVIC_GetVector 関数仕様

書式	__STATIC_FORCEINLINE uint32_t R_NVIC_GetVector(IRQn_Type IRQn)
仕様説明	ベクタテーブルのベースアドレスからのオフセットアドレスを取得します。
引数	IRQn_Type IRQn[入力] : IRQ 番号(0~31)を設定します。
戻り値	オフセットアドレス
備考	RAM 実行時に使用することでオフセットアドレスを取得します。(Inline 展開される)

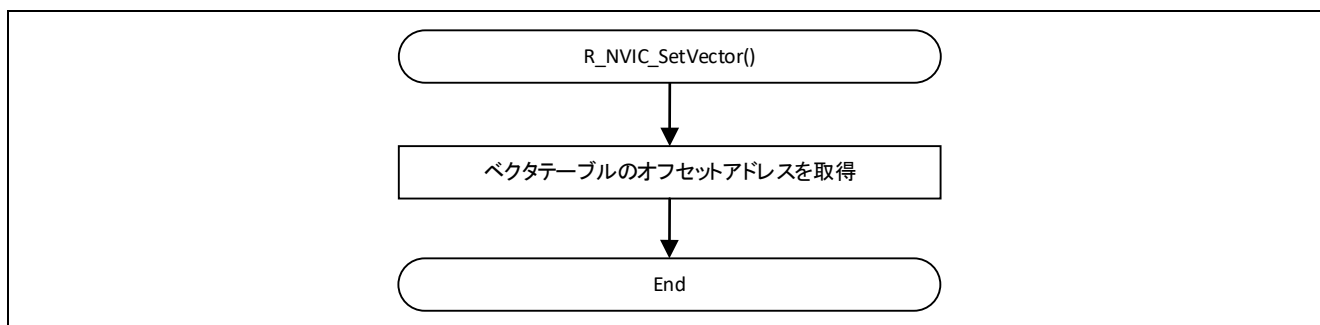


図 4.62 R_NVIC_GetVector 関数処理フロー

4.3.56 R_NVIC_SystemReset 関数

表 4-65 R_NVIC_SystemReset 関数仕様

書式	__STATIC_FORCEINLINE void R_NVIC_SystemReset(void)
仕様説明	システムレベルでのリセットを要求します。
引数	なし
戻り値	なし
備考	RAM 実行時に使用することでリセット要求をします。

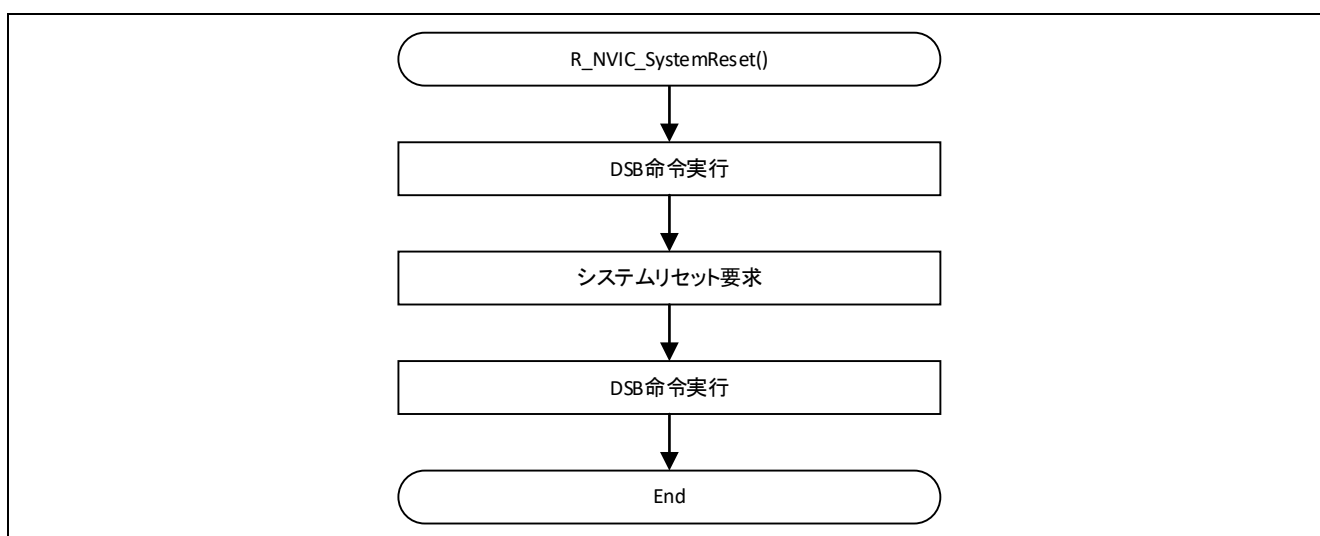


図 4.63 R_NVIC_SystemReset 関数処理フロー

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/>

お問い合わせ先

<http://japan.renesas.com/contact/>

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。

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
0.72	2019.07.01	—	初版
1.00	2019.08.09	—	R_SYSTEM ドライババージョン 1.00 に対応
		—	シリーズ名の決定に伴う文書タイトルおよびファイル名の変更 文書内のシリーズ名、グループ名の変更
		3	関連文書一覧に、CMSIS パッケージを用いた開発スタートアップガイドを追加
		4	シリーズ名の決定に伴うファイル構成の変更
		19	R_SYS_HighSpeedModeSet 関数のフロー修正
		25	R_SYS_32kHzSpeedModeSet 関数のフロー修正
		58	R_SYS_SubOscSpeedClockStart 関数のフロー修正
		59	R_SYS_SubOscSpeedClockStop 関数のフロー修正
		69	r_system_wdt_refresh 関数を追加
1.10	2020.03.09	18	R_SYS_BoostSpeedModeSet 関数のフロー修正
		20 - 21	R_SYS_HighSpeedModeSet 関数のフロー修正
		23	R_SYS_LowSpeedModeSet 関数のフロー修正
		24	R_SYS_32kHzSpeedModeSet 関数のフロー修正
		43	R_SYS_MainOscSpeedClockStart 関数のフロー修正
		44	R_SYS_MainOscSpeedClockStop 関数のフロー修正
		46	R_SYS_HighSpeedClockStart 関数のフロー修正
		47	R_SYS_HighSpeedClockStop 関数のフロー修正
		48	R_SYS_MediumSpeedClockStart 関数のフロー修正
		49	R_SYS_MediumSpeedClockStop 関数のフロー修正
		50	R_SYS_LowSpeedClockStart 関数のフロー修正
		51	R_SYS_LowSpeedClockStop 関数のフロー修正
		52	R_SYS_SubOscSpeedClockStart 関数のフロー修正
		53	R_SYS_SubOscSpeedClockStop 関数のフロー修正
		55	R_SYS_PLLSpeedClockStart 関数のフロー修正
		56	R_SYS_PLLSpeedClockStop 関数のフロー修正
1.20	2020.08.17	52	R_SYS_SubOscSpeedClockStart 関数のフロー修正
1.30	2021.05.12	77	R_NVIC_SetPriority 関数の関数仕様およびフロー修正
		78	R_NVIC_GetPriority 関数の関数仕様およびフロー修正

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、入力信号や入出力ブルアップ電源を入れないでください。入力信号や入出力ブ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、 $V_{IL}(\text{Max.})$ から $V_{IH}(\text{Min.})$ 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、 $V_{IL}(\text{Max.})$ から $V_{IH}(\text{Min.})$ 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くと、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、ノイズ輻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、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、販売または転移等する場合は、「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

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。

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/