RL78/G23

複数スレーブ・アドレス対応 I2C (マスタ)

要旨

本アプリケーションノートでは、シリアル・インタフェース IICA を使用した I2C バスのマスタ機能を使用する方法を示します。異なるスレーブ・アドレスで指定される 4 つのシリアル・メモリ (256 バイト×4) を操作します。

本アプリケーションノートは、「RL78/G23　複数スレーブ・アドレス対応 I2C (スレーブ) 」のアプリケーションノートに対応しています。

動作確認デバイス

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

1.1 I2C バスのマスタとしての基本的な仕様

I2C バスに関する仕様は以下の通りです。

・接続する I2C バス: ファースト・モード (最大 400kbps)

対象スレーブ (4 つの容量 256 バイトのシリアル・メモリ)

・スレーブ・アドレス 1: 0010000B、(1 番目のシリアル・メモリ)

・スレーブ・アドレス 2: 0100100B、(2 番目のシリアル・メモリ)

・スレーブ・アドレス 3: 1011010B、(3 番目のシリアル・メモリ)

・スレーブ・アドレス 4: 1101011B、(4 番目のシリアル・メモリ)

注意 RL78 ファミリでは、7 ビットの自局アドレスを SVA0 レジスタの上位 7 ビットで表現します。SVA0 レジスタの最下位ビットは 0 固定です。
アドレス送信は、スレーブのアドレスと転送方向 (R / $\overline{W}$) を合わせて 8 ビットとして IICA シフト・レジスタ 0 (IICA0) に書き込みます。

表 1-1 に 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
IICA0	I2C バスのマスタ機能として動作する
TM01	50 µs のインターバル・タイマ割り込み
TM03	500 ms のインターバル・タイマ割り込み
TM07	10 ms のインターバル・タイマ割り込み
P52	動作表示 LED ドライブ
P53	エラー表示 LED のドライブ
P137	動作開始スイッチ入力

主な設定を説明します。

① IICA0 の初期設定

表 1-2 に IICA0 の初期設定を示します。

表 1-2 IICA0 の初期設定

レジスタ名	設定値	設定項目
PM6	03H	初期設定期間、SCL、SDA 信号の兼用端子を入力に設定
P6	00H	SCL、SDA 信号の兼用端子を 0 に設定
IICCTL01	0DH	ファースト・モード、デジタル・フィルタ・オン、 IICA0 動作クロック f_{CLK}/2 に設定
IICWL0	15H	SCLA0 ロウ幅設定
IICWH0	14H	SCLA0 ハイ幅設定
SVA0	10H	スレーブ・アドレスを 10H に設定
IICF0	03H	初期状態はバス解放、通信予約禁止
IICCTL00	8CH	SPD 割り込み禁止、9 クロック目でウエイト、ACK 応答許可

② TM01 の初期設定

表 1-3 に TM01 初期設定を示します。

表 1-3 TM01 初期設定

レジスタ名	設定値	設定項目
TPS0	0018H	CK00 を 125kHz に、CK01、CK02、CK03 を 32 MHz に設定
TT0	0002H	チャネル 1 動作停止
TMPR001	0	割り込み優先順位はレベル 2
TMR01	8000H	TM01 のカウント・クロックを CK01 (32 MHz) に設定
TDR01	063FH	インターバル時間を 50 µs に設定
TOM0	00H	TM01 はマスタ・モード
TOL0	00H	TM01 の出力は正論理
TO0	00H	TM01 の出力は 0
TOE0	00H	TM01 の出力禁止

③ TM03 の初期設定

表 1-4 に TM03 初期設定を示します。

表 1-4 TM03 初期設定

レジスタ名	設定値	設定項目
TPS0	0018H	CK00 を 125 kHz に、CK01、CK02、CK03 を 32MHz に設定
TT0	0008H	チャネル 3 動作停止
TMR03	8000H	TM01 のカウント・クロックを CK00 (125kHz) に設定
TDR03	F423H	インターバル時間を 500ms に設定
TOM0	00H	TM03 はマスタ・モード
TOL0	00H	TM03 の出力は正論理
TO0	00H	TM03 の出力は 0
TOE0	00H	TM03 の出力禁止

④ TM07 の初期設定

表 1-5 に TM07 初期設定を示します。

表 1-5 TM07 初期設定

レジスタ名	設定値	設定項目
TPS0	0008H	CK00 を 125kHz に、CK01、CK02、CK03 を 32MHz に設定
TT0	0080H	チャネル 7 動作停止
TMPR007	0	割り込み優先順位はレベル 2
TMR07	0000H	TM07 のカウント・クロックを CK00 (125kHz) に設定
TDR07	04E1H	インターバル時間を 10ms に設定
TOM0	00H	TM07 はマスタ・モード
TOL0	00H	TM07 の出力は正論理
TO0	00H	TM07 の出力は 0
TOE0	00H	TM07 の出力禁止

1.2 動作概要

スレーブ内部のアドレスを示すレジスタ・アドレスを 1 バイト持つ 4 つのシリアル・メモリに対してデータを送受信します。

4 つのシリアル・メモリに対して、次の動作を順番に行います。

・256 バイト分を連続して読み出す。

・読み出したデータが期待値と等しいか否かを確認する。(2 巡目以降)

・0x00～0xFF のデータを 16 バイト単位で書き込む。

・16 バイトずつ 16 回に分けて読み出す。

・読み出したデータが期待値と等しいか否かを確認する。

・0xFF～0x00 のデータを 256 バイト単位で書き込む。

1.3 I2C バスの制御

表 1-6 に I2C バスの状態と g_status の関係を表します。

表 1-6 I2C バスの状態と g_status の関係

I2C バスの状態	g_status	備考
I2C バスは解放中	0x00	IICBSY0 = 0
他のマスタが I2C バスを使用中	0x0F	IICBSY0 = 1, MSTS0 = 0
マスタ動作で I2C バスを使用中	0x20	MSTS0 = 1
マスタ動作でスレーブ・アドレスを送信中	0x18	MSTS0 = 1
マスタ動作でスレーブのレジスタ・アドレスを送信中	0x16	MSTS0 = 1
マスタ動作でデータ受信中	0x14	MSTS0 = 1
マスタ動作でデータ送信中	0x12	MSTS0 = 1
データ受信が正常終了	0x24	MSTS0 = 1
データ送信が正常終了	0x22	MSTS0 = 1
IICA0 処理中もしくは他のマスタが I2C バスを使用中	0x8F	
スレーブ・アドレスに対して NACK を検出	0x80	
スレーブからの NACK を検出	0xC0	

1.4 スレーブへの通信ライブラリ

本アプリケーションノートでは、つぎの関数をサポートしています。

① 指定したスレーブに指定したバイト数のデータ書き込みを開始する。

② 指定したスレーブの指定したアドレスから指定したバイト数のデータを書き込む。

③ 指定したスレーブから指定したデータの読出しを開始する。

④ 指定したスレーブの指定したアドレスから指定したバイト数のデータを読み出す。

⑤ ①または③で起動した通信の終了をポーリングする。

⑥ ①または③で起動した通信の完了を待つ。

⑦ ストップ・コンディションを生成する。

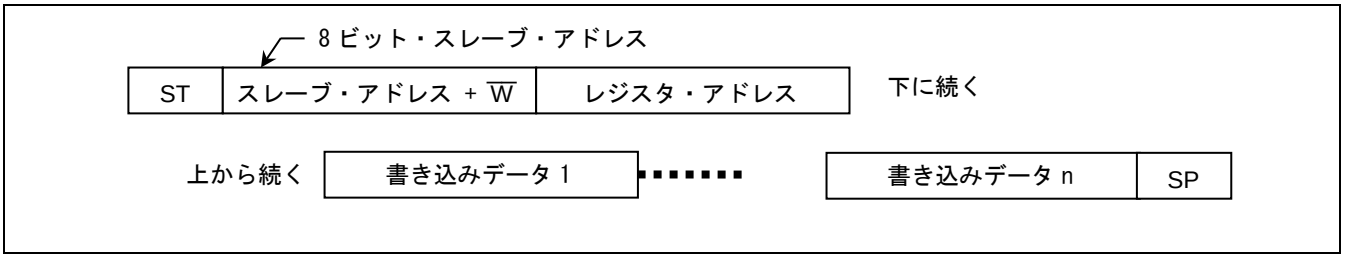
1.5 シリアル・メモリ制御

スレーブ・アドレスと次の 1 バイト・データ (レジスタ・アドレス) で指定されたシリアル・メモリのアドレスに対して、データの書き込みまたは読み出しができます。

レジスタ・アドレスを指定して連続してスレーブのシリアル・メモリにデータを書き込む場合について、シーケンスを図 1-1 に示します。

マスタからはスタート・コンディション (ST) に続いて、7 ビット・スレーブ・アドレス と転送方向 W を合わせた 8 ビット・スレーブ・アドレス を送信します。スレーブ・アドレスに続いて、シリアル・メモリの内部アドレスを指定するレジスタ・アドレスを送信します。その後、書き込みデータを順番に送信します。最終のデータ送信後、ストップ・コンディション（SP）を生成して通信を完了します。

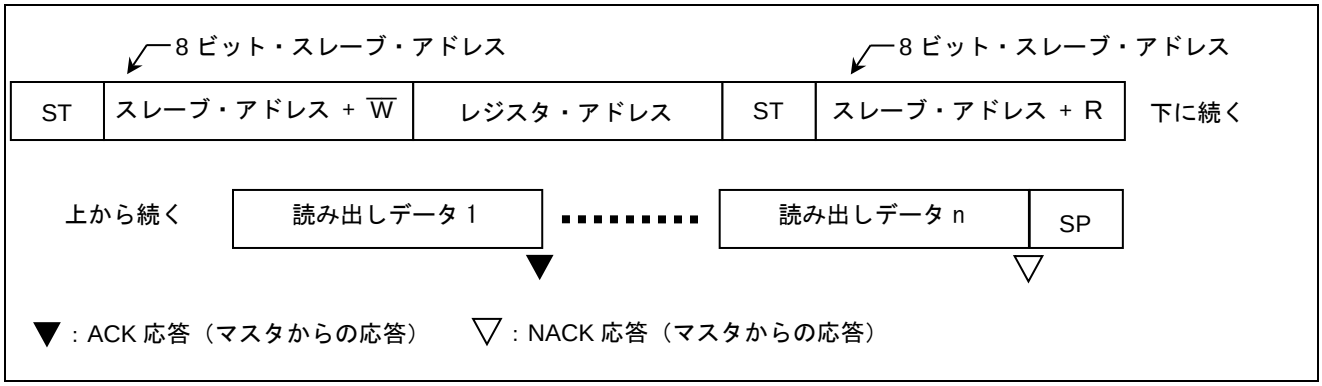
図 1-1 レジスタ・アドレス指定での連続データ書き込みシーケンス



レジスタ・アドレスを指定して連続してスレーブのシリアル・メモリからデータを読み込む場合について、シーケンスをに示します。

マスタからはスタート・コンディション (ST) に続いて、7 ビット・スレーブ・アドレス と転送方向 W を合わせた 8 ビット・スレーブ・アドレス を送信します。スレーブ・アドレスに続いて、シリアル・メモリの内部アドレスを指定するレジスタ・アドレスを送信します。その後、リスタート・コンディション（ST）に続いて、7 ビット・スレーブ・アドレス と転送方向 R を合わせた 8 ビット・スレーブ・アドレス を送信します。その後は、指定されたレジスタ・アドレスから順番にデータが送信されます (シーケンシャル・リード)。受信データに対してマスタが NACK 応答すると、スレーブは送信を中止します。最後にストップ・コンディション (SP) を生成して通信を完了します。

図 1-2 レジスタ・アドレス指定での連続データ読み出しシーケンス



2. 動作確認条件

本アプリケーションノートのサンプルコードは、下記の条件で動作を確認しています。

表 2-1 動作確認条件

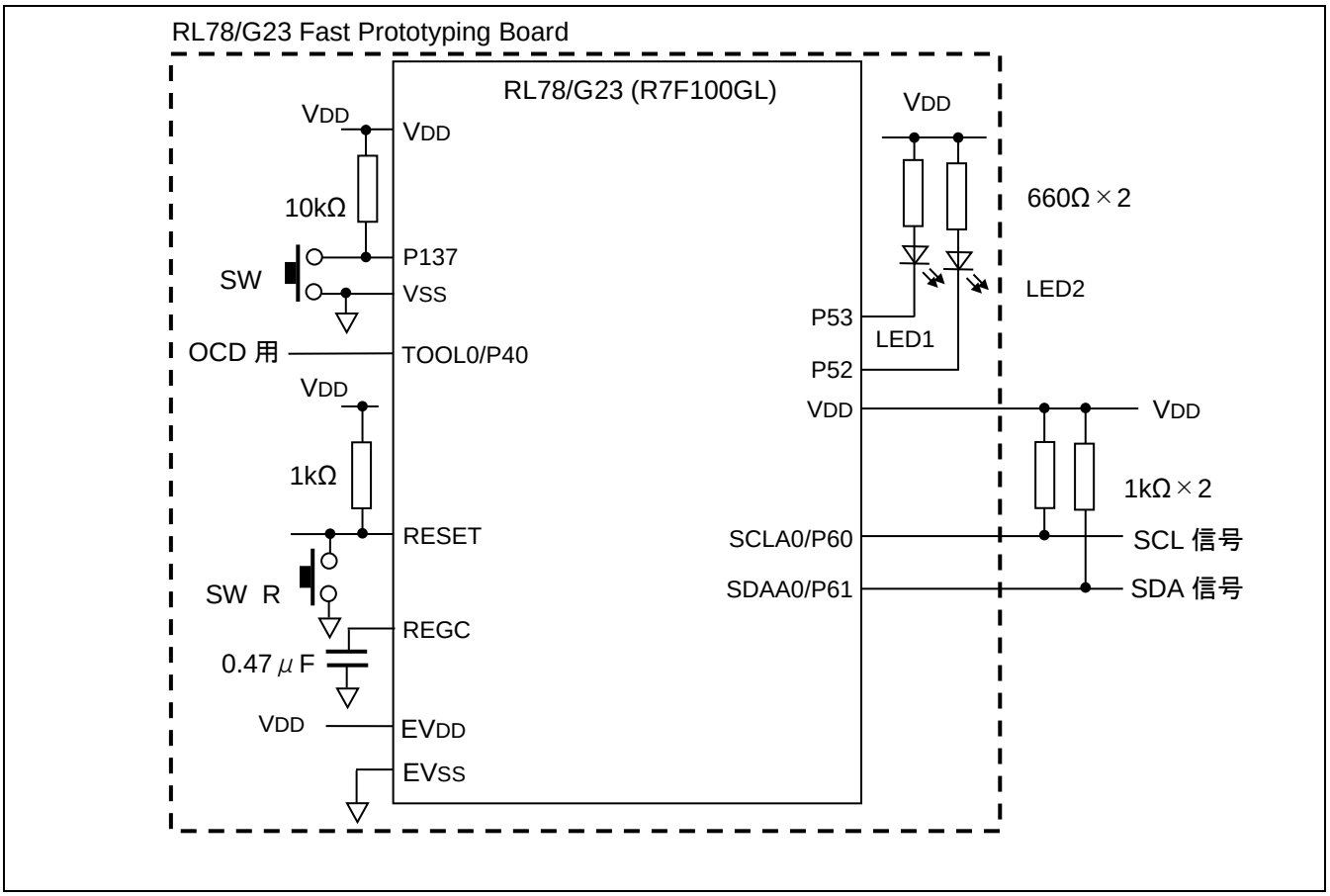
項目	内容
使用マイコン	RL78/G23 (R7F100GLG)
使用ボード	RL78/G23 Fast Prototyping Board (RTK7RLG230CLG000BJ)
動作周波数	高速オンチップ・オシレータ・クロック: 32MHz CPU／周辺ハードウエア・クロック: 32MHz
動作電圧	5.0 V (4.0V～5.5V で動作可能) LVD0 検出電圧: リセット・モード 立ち上がり時 TYP. 3.96 V (3.84 V ～ 4.08 V) 立ち下がり時 TYP. 3.88 V (3.76 V ～ 4.00 V)
統合開発環境 (CS+)	ルネサス エレクトロニクス製 CS+ V8.05.00
C コンパイラ (CS+)	ルネサス エレクトロニクス製 CC-RL V1.10.00
統合開発環境 (e2studio)	ルネサス エレクトロニクス製 e2 studio V2021-04 (21.4.0)
C コンパイラ (e2studio)	ルネサス エレクトロニクス製 CC-RL V1.10.00
統合開発環境 (IAR)	IAR Systems 製 IAR Embedded Workbench for Renesas RL78 V4.21.1
C コンパイラ (IAR)	IAR Systems 製 IAR C/C++ Compiler for Renesas RL78 V4.21.1
スマート・コンフィグレータ (SC)	ルネサス エレクトロニクス製 V1.0.1
ボードサポートパッケージ (BSP)	ルネサス エレクトロニクス製 V1.00

3. ハードウェア説明

3.1 ハードウェア構成例

図 3-1 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 3-1 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。

3.2 使用端子一覧

表 3-1 に 使用端子と機能を示します。

表 3-1 使用端子と機能

端子名	入出力	内容
P60 (D15)	入出力	SCL 信号
P61 (D14)	入出力	SDA 信号
P53 (D16)	出力	LED1 (動作表示) ドライブ
P52 (D17)	出力	LED2 (エラー表示) ドライブ
P137 (D18)	入力	スイッチ (SW) 入力

4. ソフトウェア説明

4.1 オプション・バイトの設定一覧

表 4-1 にオプション・バイト設定を示します。

表 4-1 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	11111010B	LVD0 検出電圧: リセットモード 立ち上がり時 TYP. 3.96 V 立ち下がり時 TYP. 3.88 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

4.2 定数一覧

表 4-2 に サンプルコードで使用する定数を示します。

表 4-2 サンプルコードで使用する定数

定数名	設定値	内容
LED1_pin	P5_bit.no3	D16 端子に対応するポート
LED2_pin	P5_bit.no2	D17 端子に対応するポート
SW_IN	P13_bit.no7	D18 端子に対応するポート。オンボード SW の入力端子。
LED_ON	0	LED 点灯用コモン・データの値
LED_OFF	1	LED 消灯用コモン・データの値
INC_DATA[16][16]	0x00～0xFF	RAM への書き込みデータ 1
RAM1	0010000b	RAM1 の 7 ビット・スレーブ・アドレス
RAM2	0100100b	RAM2 の 7 ビット・スレーブ・アドレス
RAM3	1011010b	RAM3 の 7 ビット・スレーブ・アドレス
RAM4	1101011b	RAM4 の 7 ビット・スレーブ・アドレス
PAGE_SIZE	256	RAM のページ・サイズ
HIGH	0x01	ハイ・レベル
LOW	0x00	ロウ・レベル
WAITTIME	1000	スタート・コンディション等検出リミット時間。ループ回数。
BUS_FREE	0x00	I2C バスは解放中
BUS_BUSY	0x0F	他のマスタが I2C バスを使用中
BUS_HOLD	0x20	マスタ動作で I2C バスを使用中
TX_SADDR	0x18	マスタ動作でスレーブ・アドレス送信中
TX_ADDR_REG	0x16	マスタ動作でレジスタ・アドレスの値を送信中
RX_MODE	0x14	マスタ動作で I2C 受信処理中
TX_MODE	0x12	マスタ動作で I2C 送信処理中
OP_END	0x20	マスタ動作で I2C 通信完了
RX_END	0x24	マスタ動作で I2C 受信完了
TX_END	0x22	マスタ動作で I2C 送信完了
SUCCESS	0x00	正常動作
COM_ERROR	0xFF	コマンド・パラメータ要因のエラー
BUS_ERROR	0x8F	I2C バス使用中のためエラー
NO_SLAVE	0x80	スレーブ・アドレスに対して NACK を検出
NO_ACK	0xC0	送信データに対して NACK を検出

4.3 変数一覧

表 4-3 に グローバル変数を示します。

表 4-3 グローバル変数

型	変数名	内容	使用関数
uint8_t	g_Tx_buff[]	257 バイトの送信バッファ	main()
uint8_t	g_Rx_buff[]	256 バイトの受信バッファ	main()
uint8_t	g_sl_addr_T[]	RAM1, RAM2, RAM3, RAM4	main()
uint8_t	g_status	現在の通信状態 0x00: I2C バス解放中 0x0F: 他のマスタが I2C バス使用中 0x12: データ送信中 0x14: データ受信中 0x16: レジスタ・アドレス送信中 0x18: スレーブ・アドレス送信中 0x20: マスタ動作中 0x22: データ送信正常終了 0x24: データ受信正常終了 0x8F: IICA0 処理中 or バス使用中 0x80: スレーブ・アドレスに対して NACK 検出 0xC0: NACK 検出	r_IICA0_master_handler(), R_IICA0_check_comstate(), R_IICA0_poll(), R_IICA0_StartCondition(), R_IICA0_bus_check(),
uint8_t *	gp_ rx_address	受信データの格納ポインタ	r_IICA0_master_handler(), R_IICA0_Master_Receive(), R_IICA0_Rx()
uint16_t	g_ rx_len	受信するデータ数	r_IICA0_master_handler(), R_IICA0_Master_Receive(), R_IICA0_Rx()
uint16_t	g_ rx_cnt	受信したデータ数	r_IICA0_master_handler(), R_IICA0_Master_Receive(), R_IICA0_Rx()
uint8_t*	gp _tx_address	送信するデータのポインタ	r_IICA0_master_handler(), R_IICA0_Master_Send(), R_IICA0_Tx()
uint16_t	g_ tx_cnt	送信するデータ数	r_IICA0_master_handler(), R_IICA0_Master_Send(), R_IICA0_Tx()
uint8_t	g_sl_addr	8 ビット・スレーブ・アドレス	r_IICA0_master_handler(), R_IICA0_Tx(),R_IICA0_Rx()
uint8_t	g_adr_reg	レジスタ・アドレス領域	r_IICA0_master_handler(), R_IICA0_Tx(),R_IICA0_Rx()
uint8_t	g_adr_flag	レジスタ・アドレス有効フラグ 0x00: 無効。データ送受信中。 0x01: 有効。データ送信中。 0x10: 有効。データ受信中。	r_IICA0_master_handler(), R_IICA0_Tx(),R_IICA0_Rx()
uint8_t	g_SW_state	スイッチの状態	main(), r_Config_TAU0_7_interrupt()

4.4 関数一覧

表 4-4 に 関数一覧を示します。

表 4-4 関数一覧

関数名	概要
R_Config_IICA0_Create()	IICA0 初期設定処理
r_Config_IICA0_interrupt()	IICA0 割り込み処理
r_IICA0_master_handler()	IICA0 マスタ割り込み処理
R_IICA0_StartCondition()	スタート・コンディション生成
R_IICA0_StopCondition()	ストップ・コンディション生成
R_ IICA0_Master_Send()	スレーブへのデータ送信起動
R_IICA0_Tx()	スレーブの指定アドレスへのデータ送信
R_ IICA0_Master_Receive()	スレーブからのデータ受信起動
R_IICA0_Rx()	スレーブの指定アドレスからのデータ受信
R_IICA0_poll()	通信状態ポーリング
wait_time()	50us 待ち
R_IICA0_wait_comend()	通信完了待ち
R_IICA0_check_comstate()	IICA0 の通信状態確認
R_IICA0_bus_check()	I2C バスの状態確認
r_Config_TAU0_7_interrupt()	10ms インターバル・タイマ割り込み処理

4.5 関数一覧

サンプルコードの関数仕様を示します。

[関数名]	R_ Config_IICA0_Create()	
概　要	IICA0 初期設定処理	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	void R_ Config_IICA0_Create(void)	
説　明	IICA0 の初期設定（ファースト・モード、マスタ）を行う。	
引　数	なし	
リターン値	なし	

[関数名]	r_Config_IICA0_interrupt()	
概　要	IICA0 割り込み処理	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	static void __near r_iica0_interrupt(void)	
説　明	IICA0 割り込み要求の受け付け処理を行う。	
引　数	なし	
リターン値	なし	

[関数名]	r_IICA0_master_handler()	
概　要	IICA0 割り込みのマスタ処理関数	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	static void r_ IICA0_master_handler(void)	
説　明	マスタ動作中の IICA0 割り込み処理。スレーブ・アドレスに対して NACK を検出した場合、ストップ・コンディションを生成して通信を終了。スレーブ・アドレスに対して ACK を検出した場合、送信モードではレジスタ・アドレスまたはデータを送信する。受信モードではデータを受信し、受信データを格納する。	
引　数	なし	
リターン値	なし	

[関数名]	R_IICA0_StartCondition()	
概　要	スタート・コンディションを生成	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_StartCondition(void)	
説　明	スタート・コンディションを生成し、その結果を確認する。	
引　数	なし	
リターン値	uint8_t	0x00 (SUCCESS): スタート・コンディション生成完了 0x8F (BUS_ERROR): エラー発生

[関数名]	R_IICA0_StopCondition()	
概　要	ストップ・コンディションを生成	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_StopCondition(void)	
説　明	ストップ・コンディションを生成し、その結果を検出する。	
引　数	なし	
リターン値	uint8_t	0x00 (SUCCESS): ストップ・コンディション生成。通信完了。 0x8F (BUS_ERROR): エラー発生

[関数名]	R_IICA0_Master_Send()	
概　要	スレーブへのデータ送信起動処理	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_Master_Send(uint8_t sladr8, uint8_t * const tx_buf, uint16_t tx_num)	
説　明	IICA0 のデータ送受信が正常終了している場合、つぎの動作を行う。 ・I2C バスが解放されている場合、スタート・コンディションを生成する。 ・IICA0 がマスタとして動作している場合、送信パラメータをコピーして スレーブ・アドレスを送信する。 上記以外は、エラーで戻る。	
引　数	uint8_t sladr8	スレーブ・アドレス
	uint8_t * const tx_buf	送信するデータの格納ポインタ
	uint16_t tx_num	送信するデータのバイト数
リターン値	uint8_t	ステータス 0x00 (SUCCESS): 正常に通信開始 0x8F (BUS_ERROR): IICA0 処理中もしくは I2C バス使用中

[関数名]	R_IICA0_Tx()	
概　要	スレーブの指定アドレスにデータを送信	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h,, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_Tx(uint8_t sladr7, uint8_t adr, uint8_t * const tx_buf, uint16_t tx_num)	
説　明	① I2C バスの状態を確認する。 I2C バスが解放されている場合、スタート・コンディションを生成する。 I2C バスが使用されている場合、エラーで戻る。 ② スタート・コンディションを生成後、レジスタ・アドレス有効 (g_adr_flag = 0x01) に設定する。 ③ スレーブ・アドレスを 7 ビットから 8 ビットに変換し、データ送信を開始する。 ④ 通信完了を待つ。	
引　数	uint8_t sladr7	7 ビット・スレーブ・アドレス
	uint8_t adr	レジスタ・アドレスへの設定値
	uint8_t * const tx_buf	送信するデータの格納ポインタ
	uint16_t tx_num	送信するデータのバイト数
リターン値	uint8_t	結果 0x00 (SUCCESS): 送信完了 0x8F (BUS_ERROR): IICA0 処理中 or I2C バス使用中

[関数名]	R_IICA0_Master_Receive()	
概　要	指定スレーブへのデータ送信起動処理	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_Master_Receive (uint8_t sladr8, uint8_t * const tx_buf, uint16_t tx_num)	
説　明	IICA0 のデータ送受信が正常終了している場合、つぎの動作を行う。 ・I2C バスが解放されている場合、スタート・コンディションを生成する。 ・IICA0 がマスタとして動作している場合、送信パラメータをコピーして スレーブ・アドレスを送信する。 上記以外は、エラーで戻る。	
引　数	uint8_t sladr8	スレーブ・アドレス
	uint8_t * const rx_buf	受信したデータの格納ポインタ
	uint16_t rx_num	受信するデータのバイト数
リターン値	uint8_t	ステータス 0x00 (SUCCESS): 正常に通信開始 0x8F (BUS_ERROR): IICA0 処理中もしくは I2C バス使用中 0xFF (COM_ERROR): 受信データ数エラー

[関数名]	R_IICA0_Rx()	
概　要	スレーブの指定アドレスからデータを受信	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h,, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_Rx(uint8_t sladr7, uint8_t adr, uint8_t * const tx_buf, uint16_t tx_num)	
説　明	① I2C バスの状態を確認する。 I2C バスが解放されている場合、スタート・コンディションを生成する。 I2C バスが使用されている場合、エラーで戻る。 ② スタート・コンディションを生成後、通信パラメータを設定する。 ③ レジスタ・アドレス有効 (g_adr_flag = 0x10) に設定する。 ④ スレーブ・アドレスを 7 ビットから 8 ビットに変換し、データ受信を開始する。 ⑤ 通信完了を待つ。	
引　数	uint8_t sladr7	7 ビット・スレーブ・アドレス
	uint8_t adr	レジスタ・アドレスへの設定値
	uint8_t * const tx_buf	送信するデータの格納ポインタ
	uint16_t tx_num	送信するデータのバイト数
リターン値	uint8_t	結果 0x00 (SUCCESS): 受信完了 0x8F (BUS_ERROR): IICA0 処理中 or I2C バス使用中 0xFF (COM_ERROR): 受信データ数エラー

[関数名]	R_IICA0_poll()	
概　要	IICA0 通信状態ポーリング処理	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_poll(void)	
説　明	処理中の通信状態を確認する。	
引　数	なし	
リターン値	uint8_t	通信状況 0x00 (SUCCESS): 通信完了 (通信成功) 0x01 (ON_COMMU): 通信中 0x8F (BUS_ERROR): IICA0 処理中もしくは I2C バス使用中 0x80 (NO_SLAVE): スレーブ・アドレスに対して NACK 検出 0xC0 (NACK): NACK 検出

[関数名]	wait_time()	
概　要	50us 待ち	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	void wait_time(void)	
説　明	50us 待ち合わせる。	
引　数	なし	
リターン値	なし	

[関数名]	R_IICA0_wait_comend()	
概　要	IICA0 通信完了待ち処理	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA_wait_comend(uint8_t stop)	
説　明	処理中の通信完了を待つ。通信完了後にエラーを検出した場合、ストップ・コンディションを生成して通信を終了する。	
引　数	uint8_t　stop	0x00：正常終了ならストップ・コンディション生成なし 0x01：通信完了時ストップ・コンディション生成
リターン値	uint8_t	通信結果 0x00 (SUCCESS): 通信完了 (通信成功) 0x8F (BUS_ERROR): IICA0 処理中もしくは I2C バス使用中 0x80 (NO_SLAVE): スレーブ・アドレスに対して NACK 検出 0xC0 (NACK): NACK 検出

[関数名]	R_IICA0_check_comstate()	
概　要	IICA0 通信状態確認	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_check_comstate(void)	
説　明	IICA0 の状態を確認する (g_status を読み出す)。	
引　数	なし	
リターン値	uint8_t	g_status の値

[関数名]	R_IICA0_bus_check()	
概　要	I2C バスの状態確認	
ヘッダ	r_cg_macrodriver.h, Config_IICA0.h, r_cg_userdefine.h	
宣　言	uint8_t R_IICA0_bus_check(void)	
説　明	I2C バスの状態を確認する。IICA0 がデータ処理中の場合、エラーで戻る。 I2C バスを確保しているが通信は終了しているか I2C バスが解放されている場合、スタート・コンディションを生成する。	
引　数	なし	
リターン値	uint8_t	結果 0x00 (SUCCESS): スタート・コンディション生成完了 0x8F (BUS_ERROR): IICA0 処理中もしくは I2C バス使用中

[関数名]	r_tau0_channel7_interrupt()
概　要	10ms インターバル割り込み処理
ヘッダ	Config_TAU0_7.h、r_cg_macrodriver.h、r_cg_userdefine.h
宣　言	static void r_tau0_channel7_interrupt(void)
説　明	スイッチの状態のサンプリングを行う。
引　数	なし
リターン値	なし

4.6 フローチャート

4.6.1 メイン処理

図 4-1～図 4-3 にメイン処理のフローチャートを示します。

図 4-1 メイン処理 (1/3)

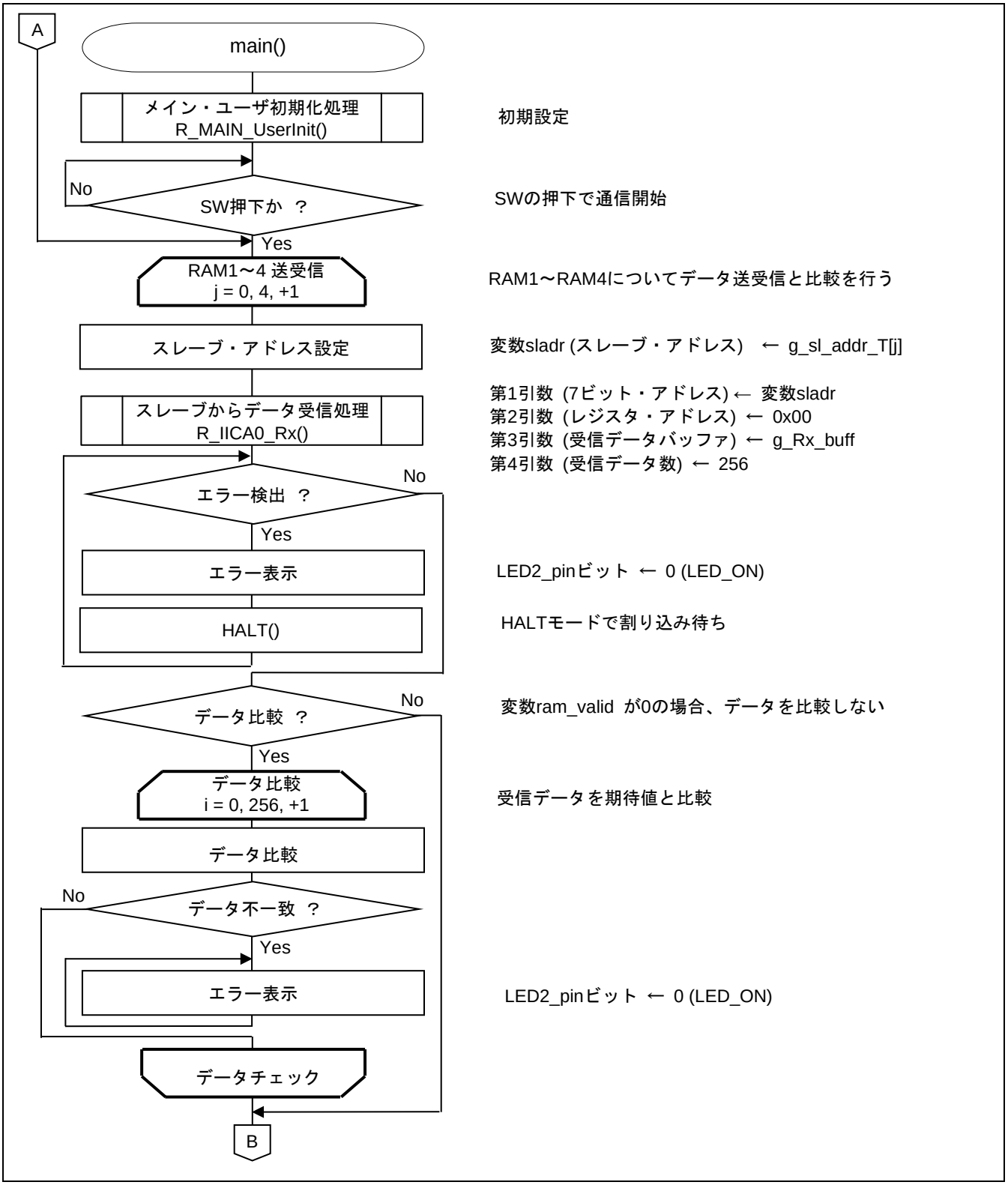


図 4-2 メイン処理 (2/3)

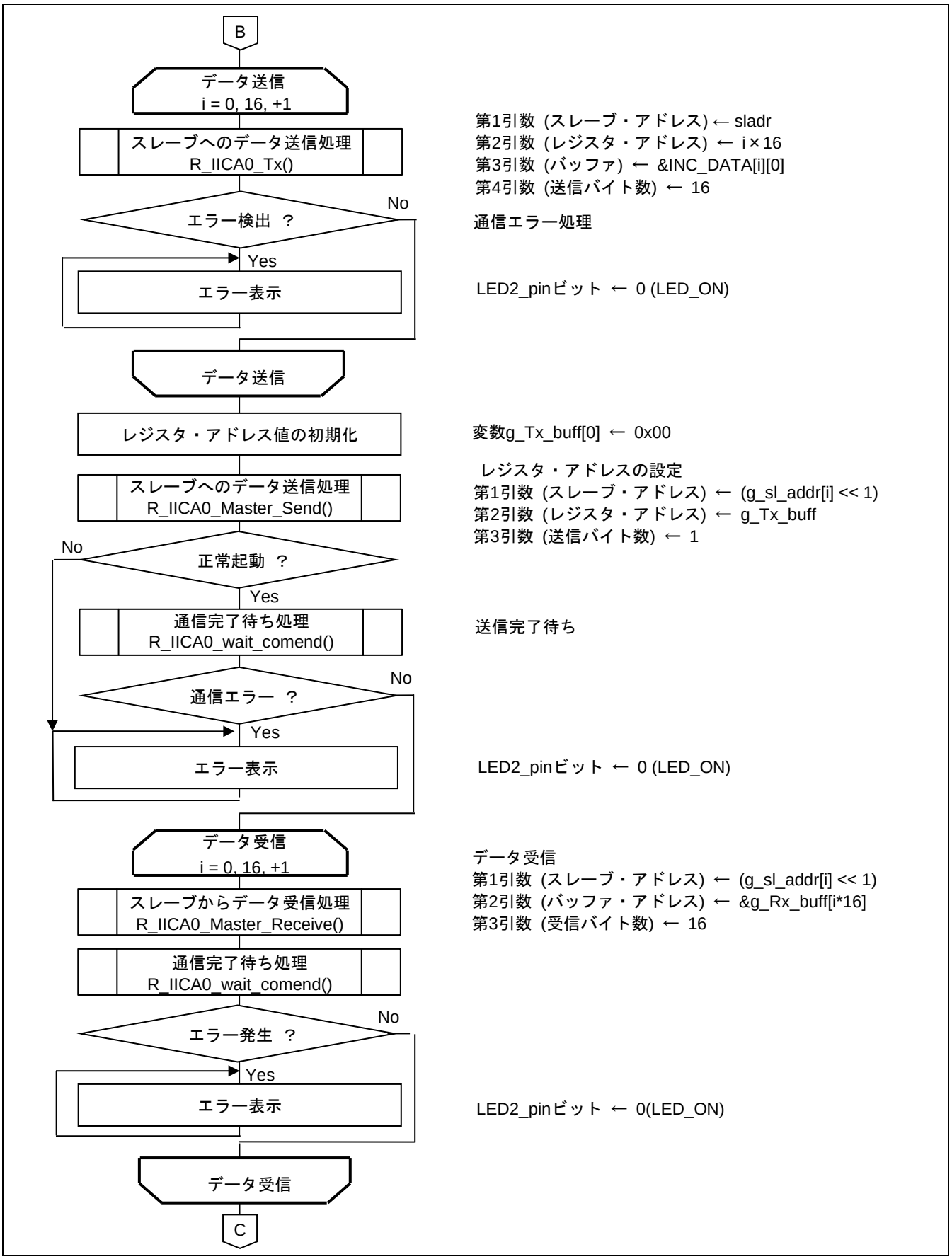
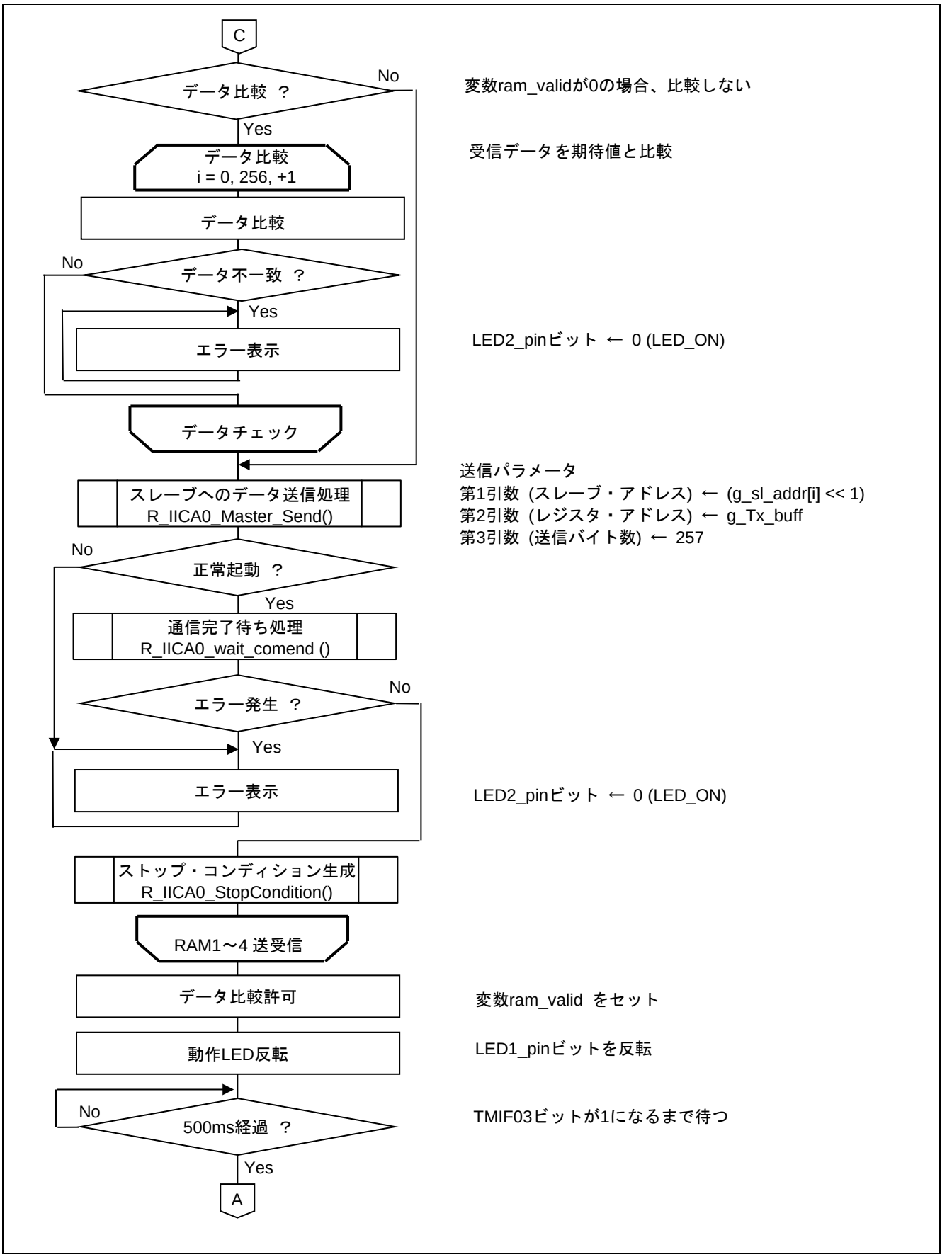

B
データ送信
i = 0, 16, +1
スレーブへのデータ送信処理
R_IICA0_Tx()
第1引数 (スレーブ・アドレス) ← sladr
第2引数 (レジスタ・アドレス) ← i×16
第3引数 (バッファ) ← &INC_DATA[i][0]
第4引数 (送信バイト数) ← 16
エラー検出 ？
No
Yes
通信エラー処理
エラー表示
LED2_pinビット ← 0 (LED_ON)
データ送信
レジスタ・アドレス値の初期化
変数g_Tx_buff[0] ← 0x00
スレーブへのデータ送信処理
R_IICA0_Master_Send()
レジスタ・アドレスの設定
第1引数 (スレーブ・アドレス) ← (g_sl_addr[i] << 1)
第2引数 (レジスタ・アドレス) ← g_Tx_buff
第3引数 (送信バイト数) ← 1
正常起動 ？
No
Yes
通信完了待ち処理
R_IICA0_wait_comend()
送信完了待ち
通信エラー ？
No
Yes
エラー表示
LED2_pinビット ← 0 (LED_ON)
データ受信
i = 0, 16, +1
データ受信
第1引数 (スレーブ・アドレス) ← (g_sl_addr[i] << 1)
第2引数 (バッファ・アドレス) ← &g_Rx_buff[i*16]
第3引数 (受信バイト数) ← 16
スレーブからデータ受信処理
R_IICA0_Master_Receive()
通信完了待ち処理
R_IICA0_wait_comend()
エラー発生 ？
No
Yes
エラー表示
LED2_pinビット ← 0(LED_ON)
データ受信
C

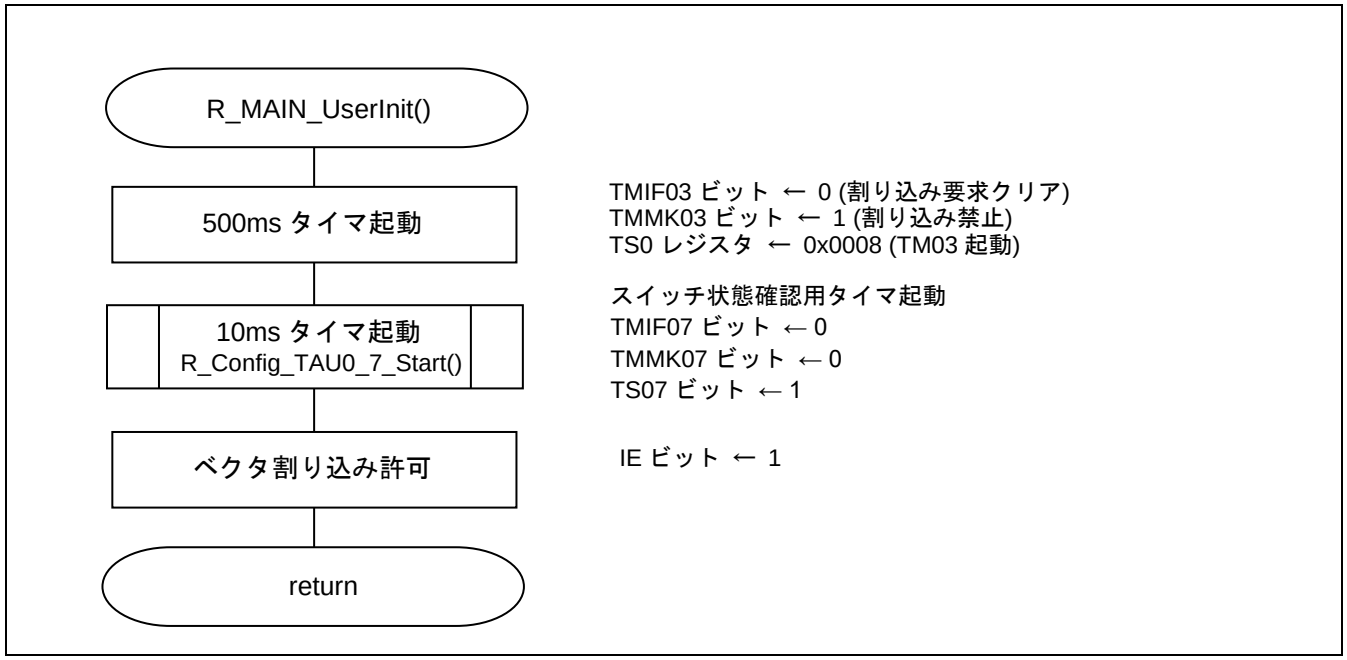
図 4-3 メイン処理 (3/3)


C
データ比較 ？
No
Yes
変数ram_validが0の場合、比較しない
データ比較
i = 0, 256, +1
受信データを期待値と比較
データ比較
No
データ不一致 ？
Yes
エラー表示
LED2_pinビット ← 0 (LED_ON)
データチェック
スレーブへのデータ送信処理
R_IICA0_Master_Send()
送信パラメータ
第1引数 (スレーブ・アドレス) ← (g_sl_addr[i] << 1)
第2引数 (レジスタ・アドレス) ← g_Tx_buff
第3引数 (送信バイト数) ← 257
No
正常起動 ？
Yes
通信完了待ち処理
R_IICA0_wait_comend ()
No
エラー発生 ？
Yes
エラー表示
LED2_pinビット ← 0 (LED_ON)
ストップ・コンディション生成
R_IICA0_StopCondition()
RAM1～4 送受信
データ比較許可
変数ram_valid をセット
動作LED反転
LED1_pinビットを反転
No
500ms経過 ？
TMIF03ビットが1になるまで待つ
Yes
A

4.6.2 変数初期化処理

図 4-4 に変数初期化関数のフローチャートを示します。

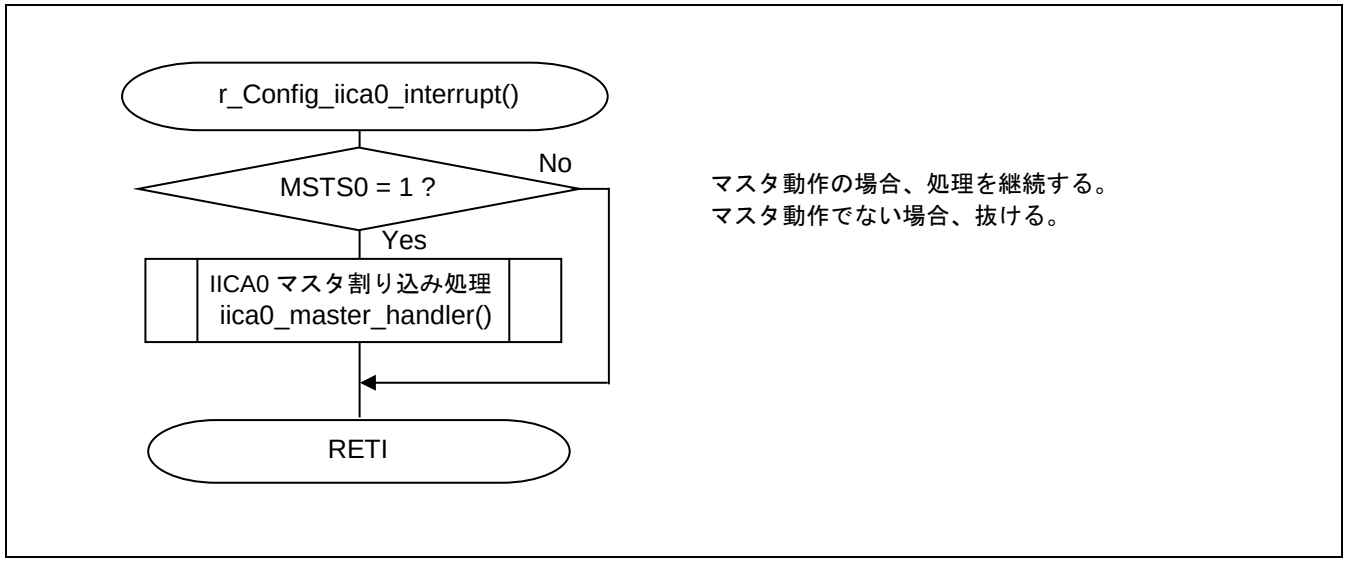
図 4-4 変数初期化処理



4.6.3 IICA0 割り込み処理関数

図 4-5 に IICA0 割り込み処理関数のフローチャートを示します。

図 4-5 IICA0 割り込み処理関数



4.6.4 IICA0 マスタ割り込み処理関数

図 4-6～図 4-9 に IICA0 マスタ割り込み処理関数のフローチャートを示します。

図 4-6 IICA0 マスタ割り込み処理関数 (1/4)

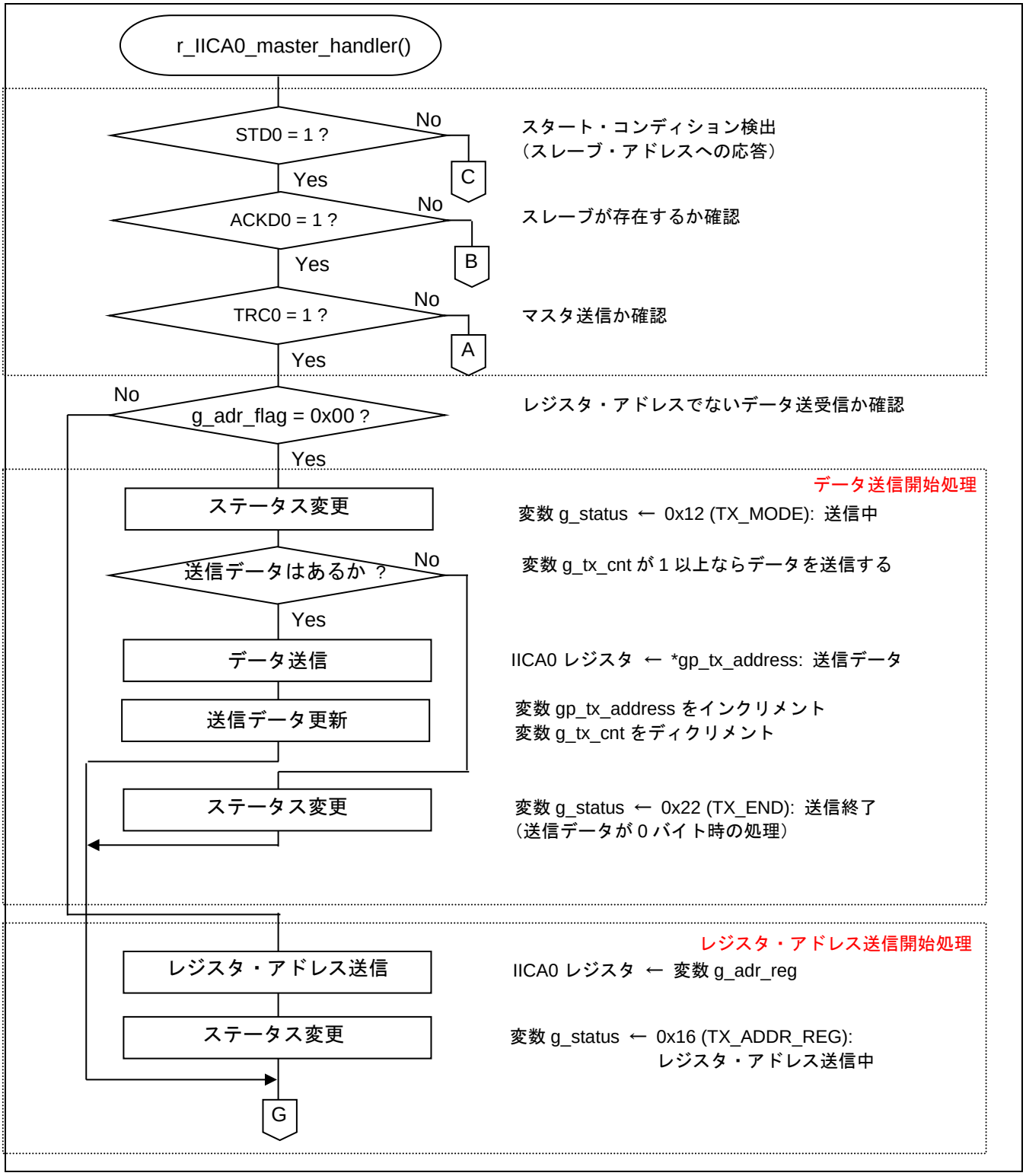


図 4-7 IICA0 マスタ割り込み処理関数 (2/4)

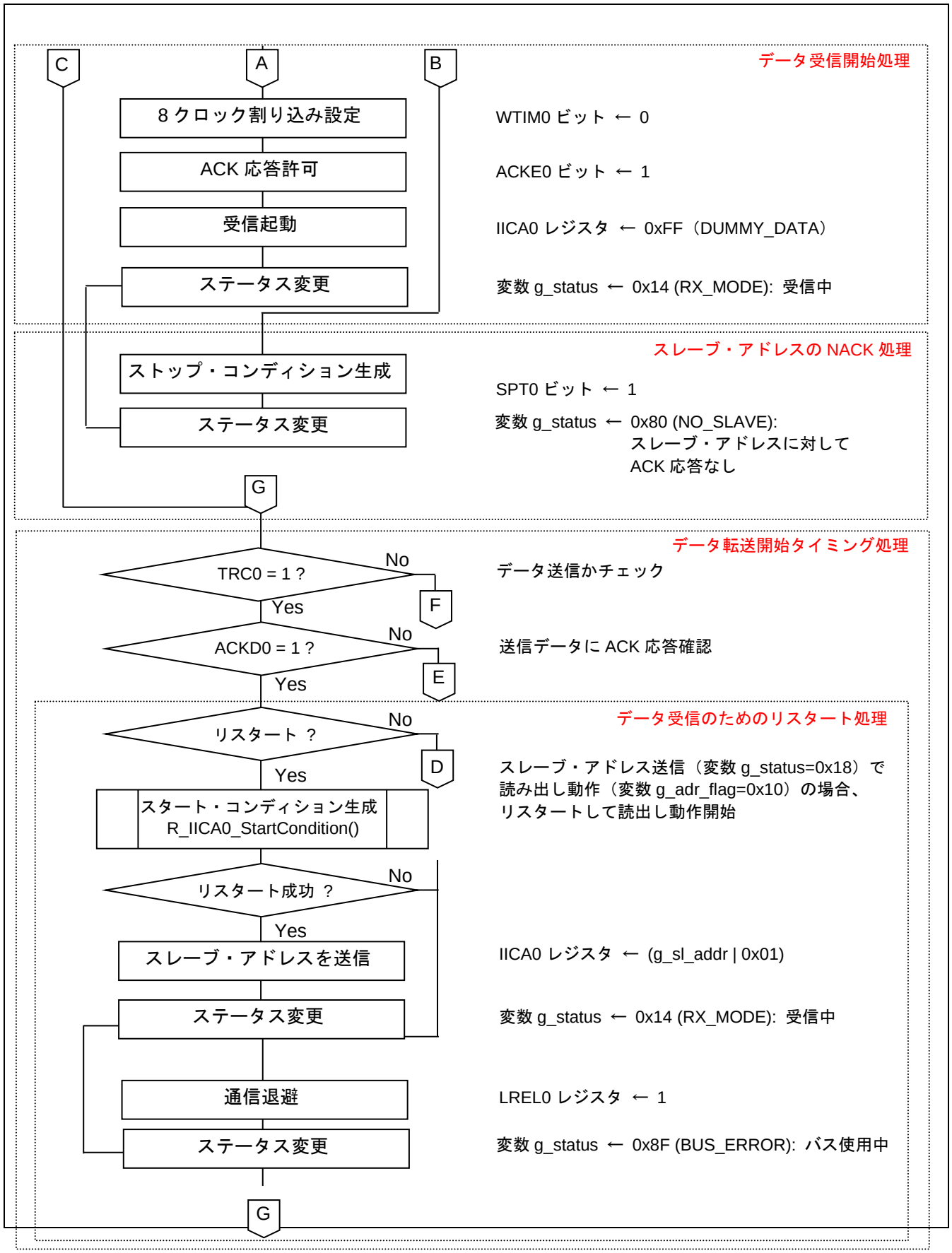


図 4-8 IICA0 マスタ割り込み処理関数 (3/4)

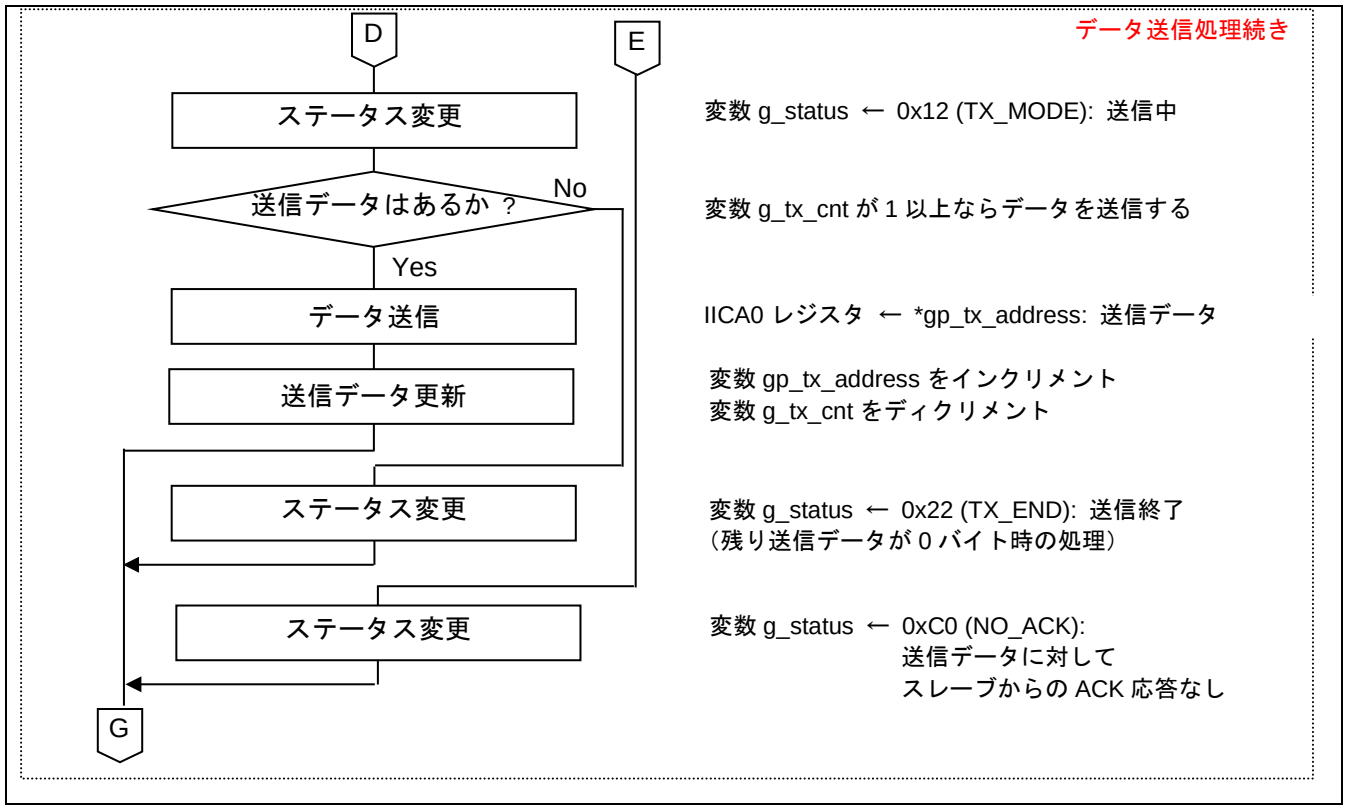
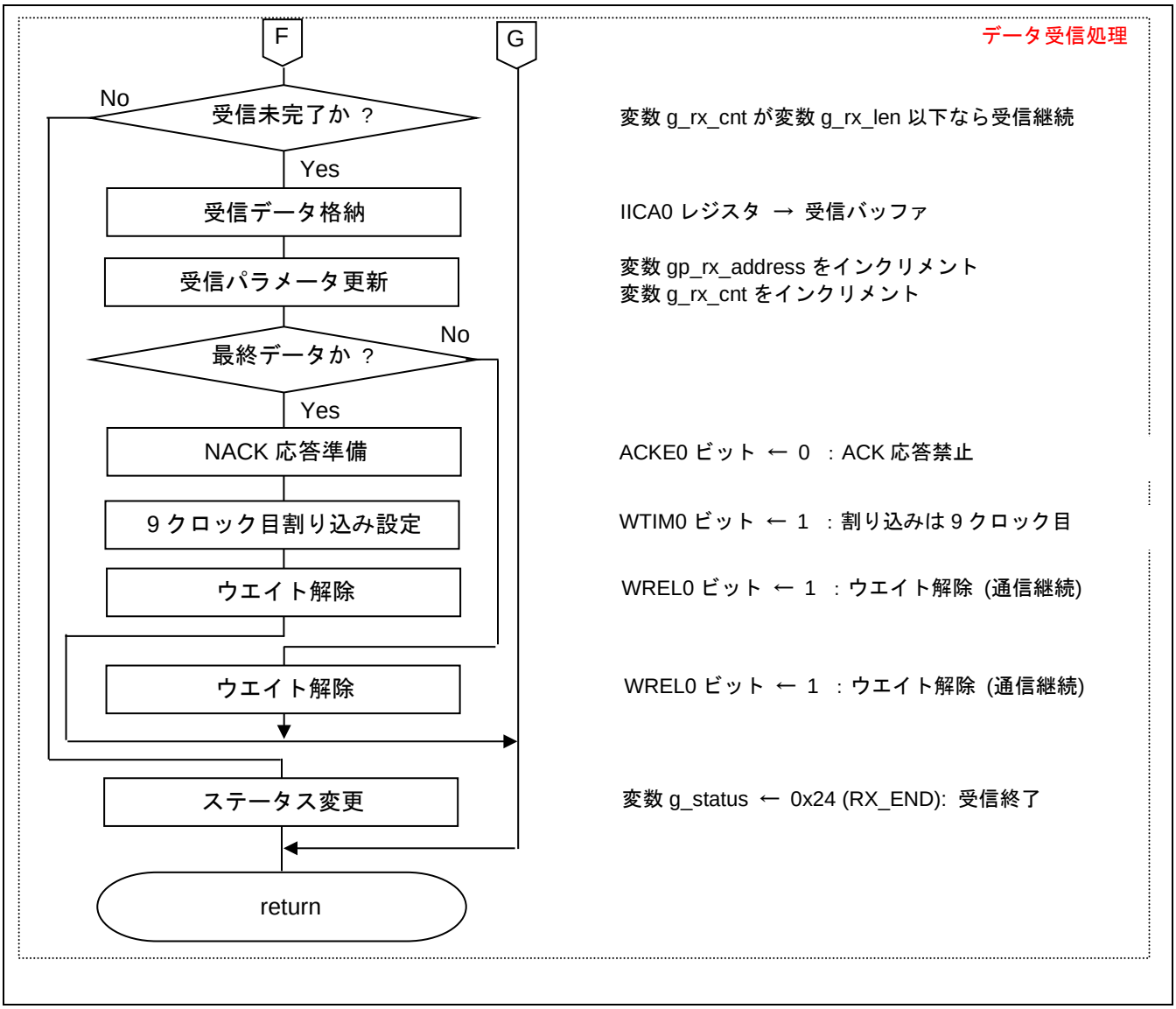


図 4-9 IICA0 マスタ割り込み処理関数 (4/4)

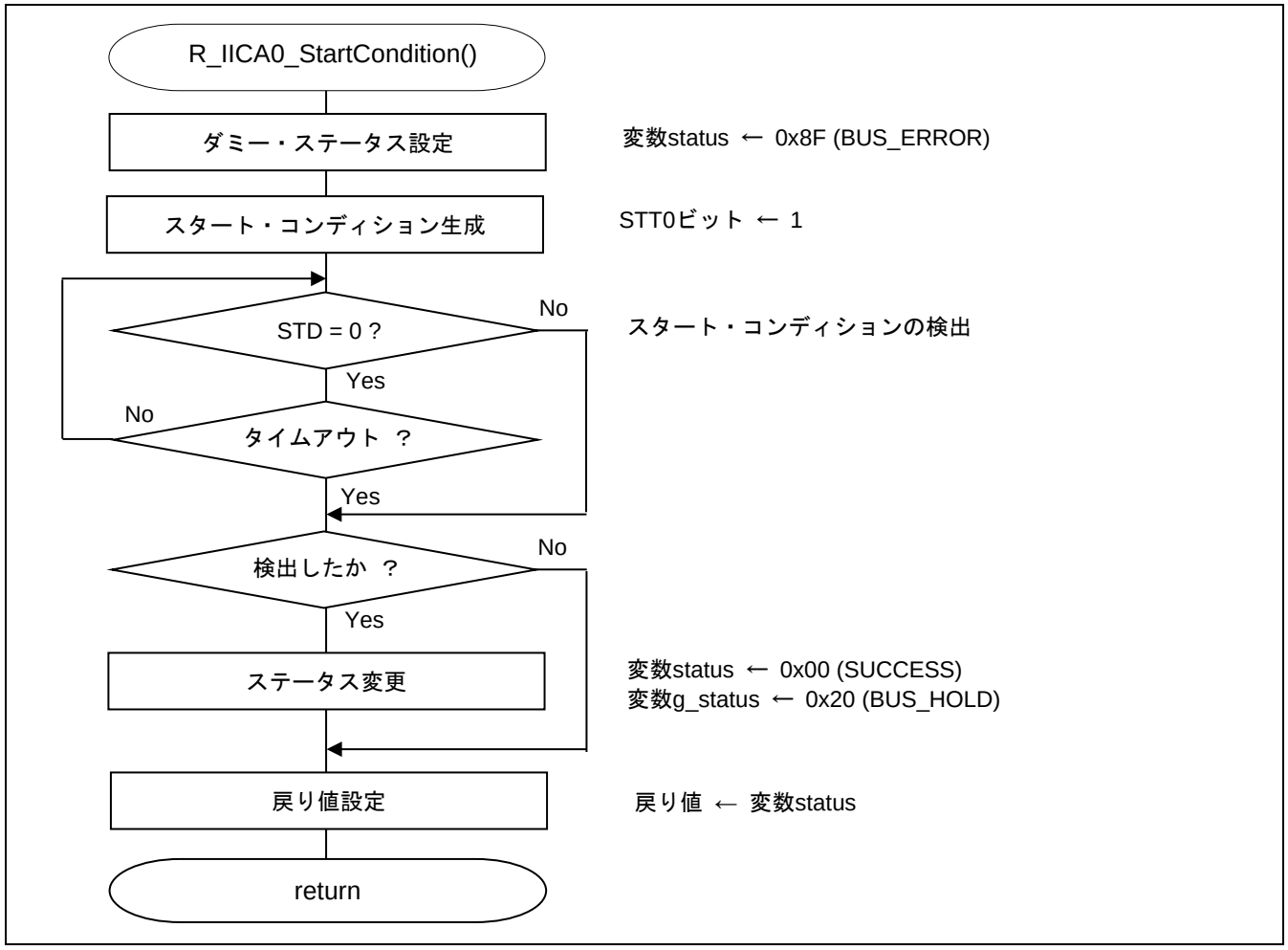


変数 g_rx_cnt が変数 g_rx_len 以下なら受信継続

4.6.5 IICA0 スタート・コンディション生成関数

図 4-10 に IICA0 スタート・コンディション生成関数のフローチャートを示します。

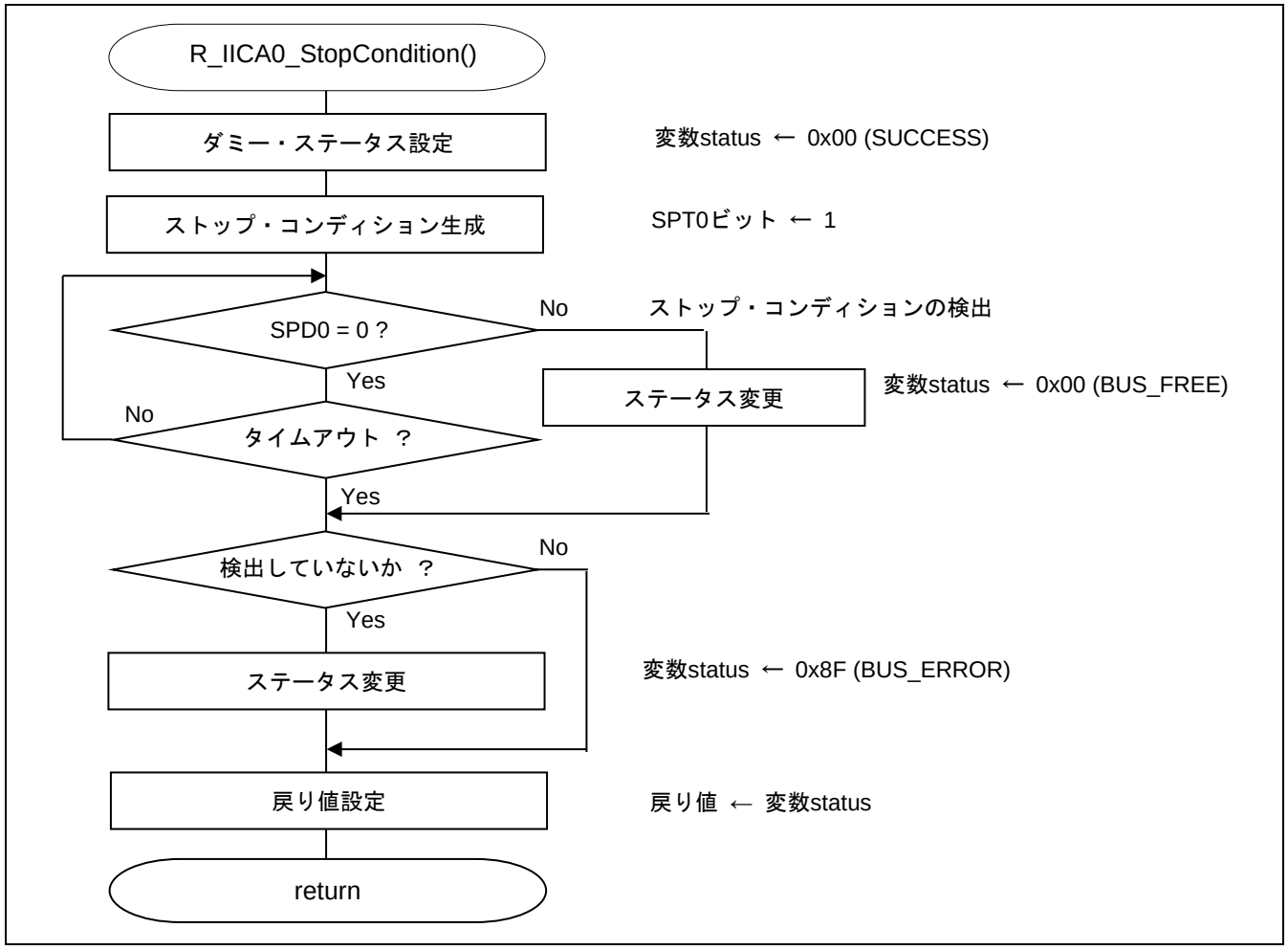
図 4-10 IICA0 スタート・コンディション生成関数



4.6.6 IICA0 ストップ・コンディション生成関数

図 4-11 に IICA0 ストップ・コンディション生成関数のフローチャートを示します。

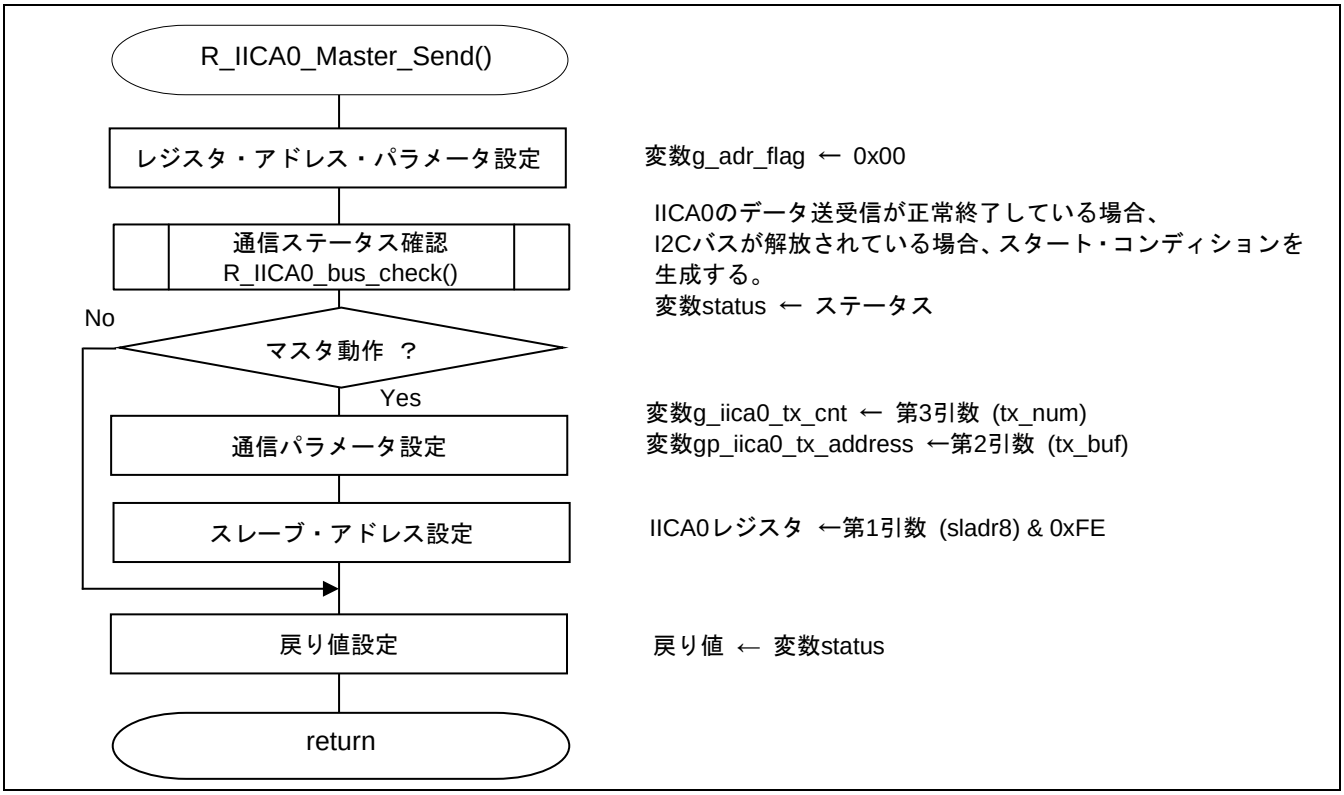
図 4-11 IICA0 ストップ・コンディション生成関数



4.6.7 IICA0 マスタ送信起動関数

図 4-12 に IICA0 マスタ送信起動関数のフローチャートを示します。

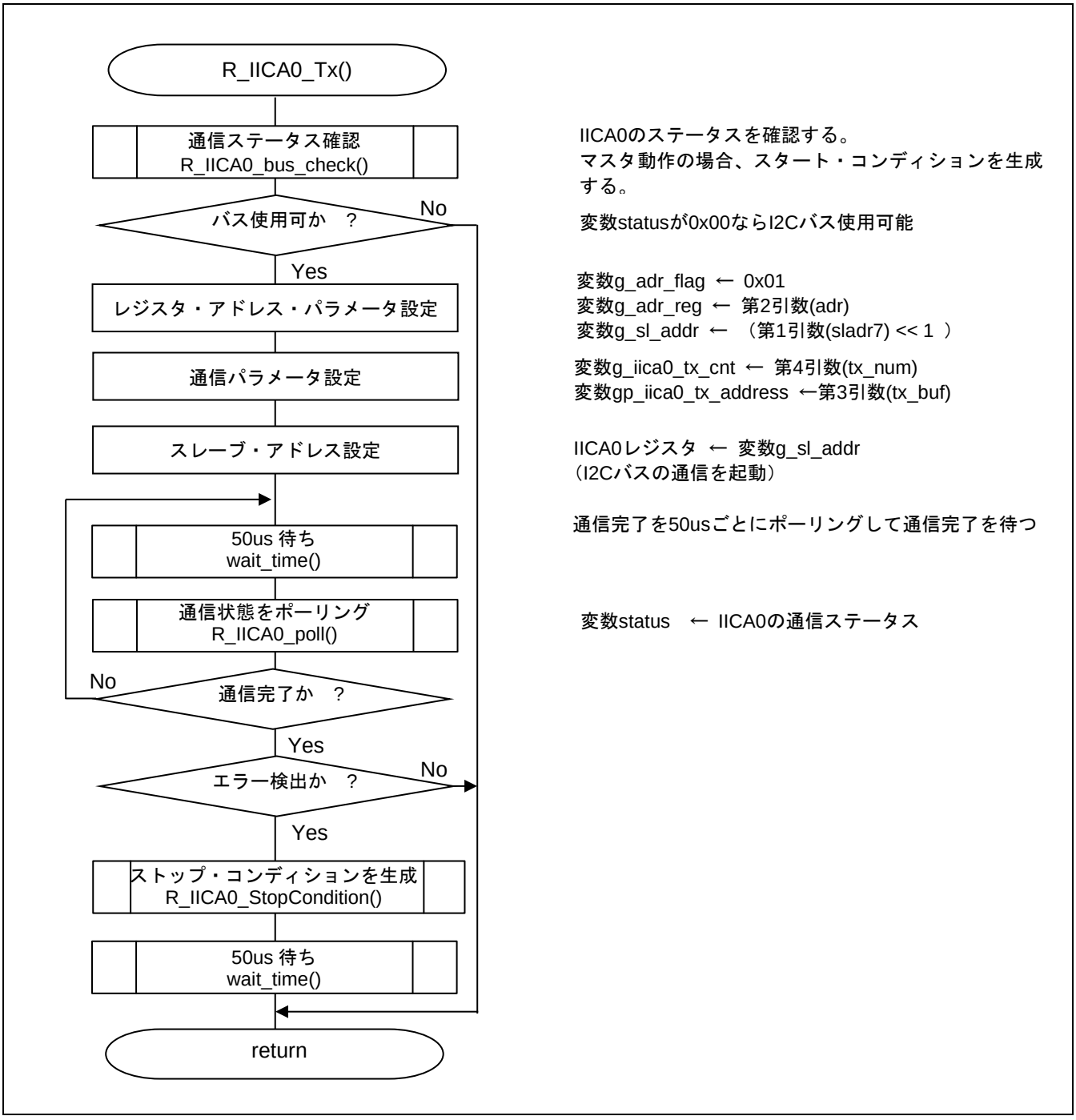
図 4-12 IICA0 マスタ送信起動関数



4.6.8 IICA0 データ送信処理関数

図 4-13 に IICA0 データ送信処理関数のフローチャートを示します。

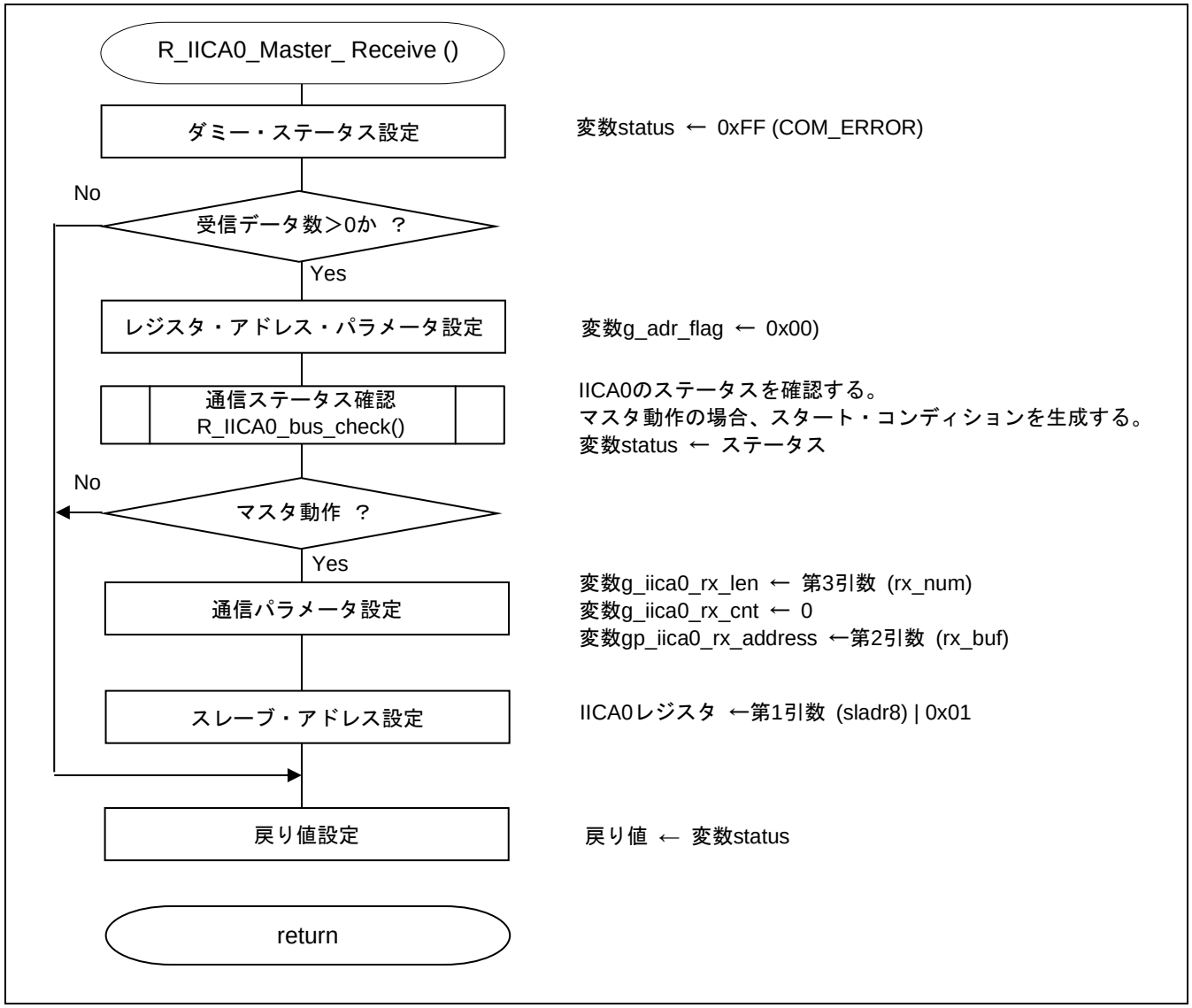
図 4-13 IICA0 データ送信処理関数



4.6.9 IICA0 マスタ受信起動関数

図 4-14 に IICA0 マスタ受信起動関数のフローチャートを示します。

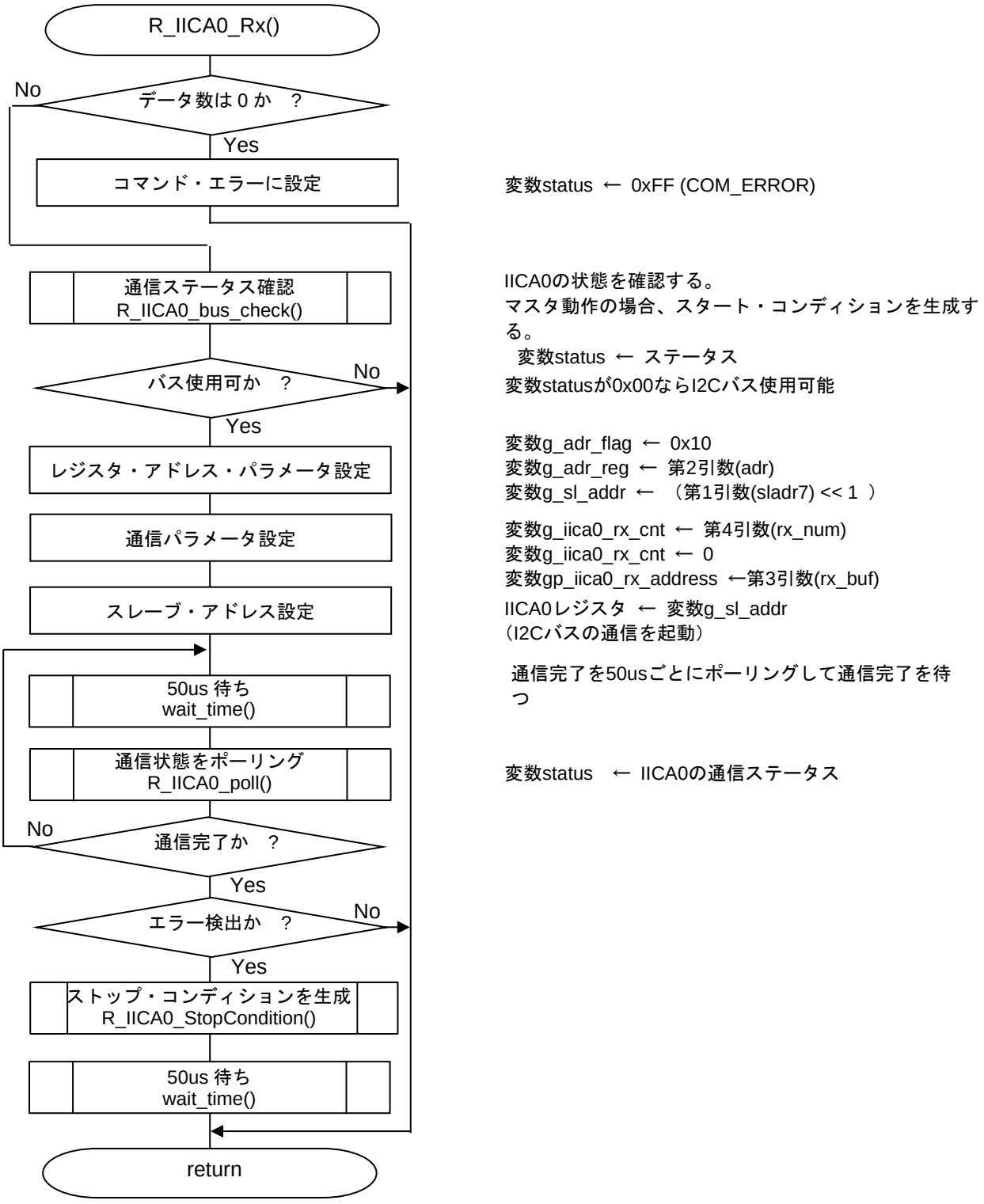
図 4-14 IICA0 マスタ受信起動関数



4.6.10 IICA0 データ受信処理関数

図 4-15 に IICA0 データ受信処理関数のフローチャートを示します。

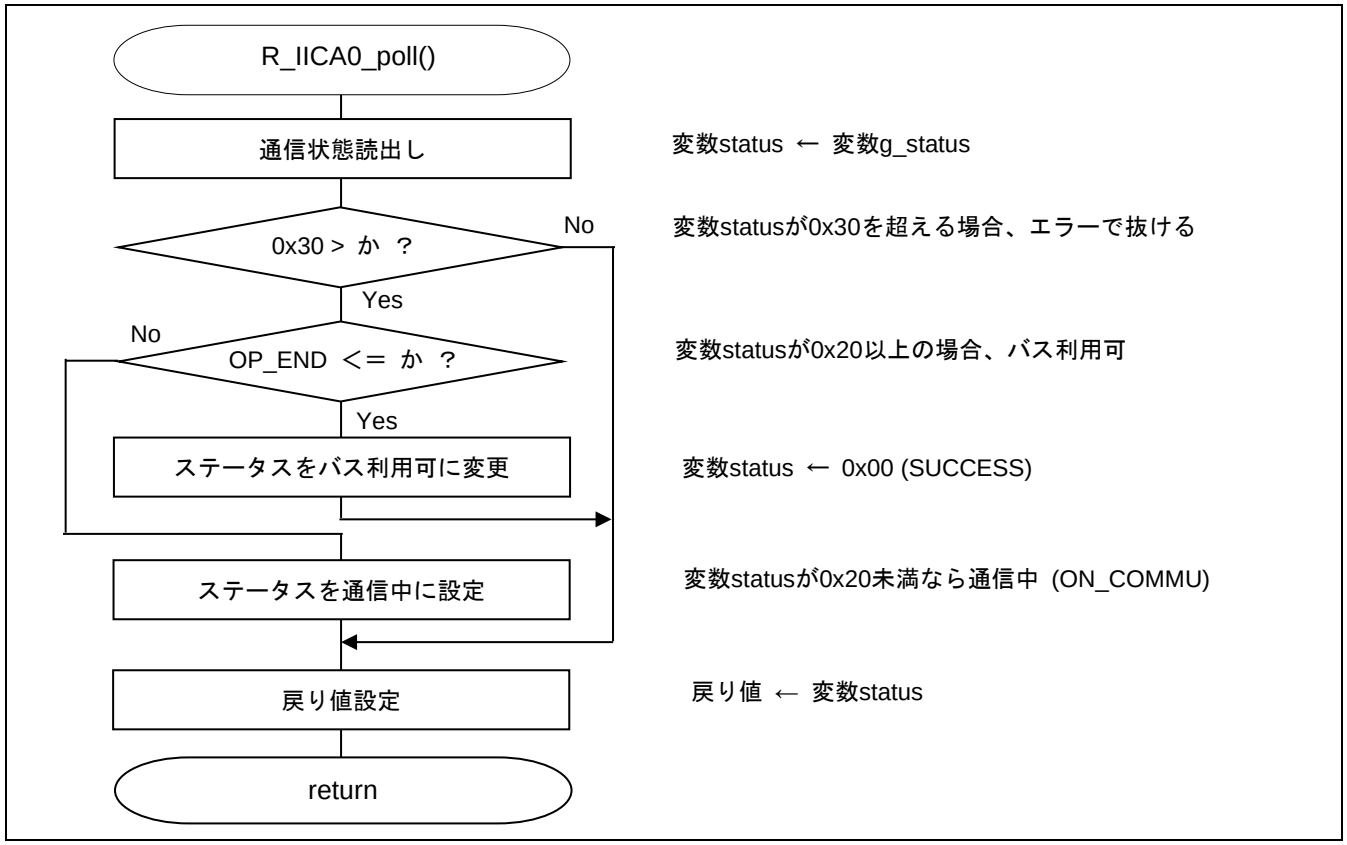
図 4-15　IICA0 データ受信処理関数



4.6.11 通信状態ポーリング関数

図 4-16 に通信状態ポーリング関数のフローチャートを示します。

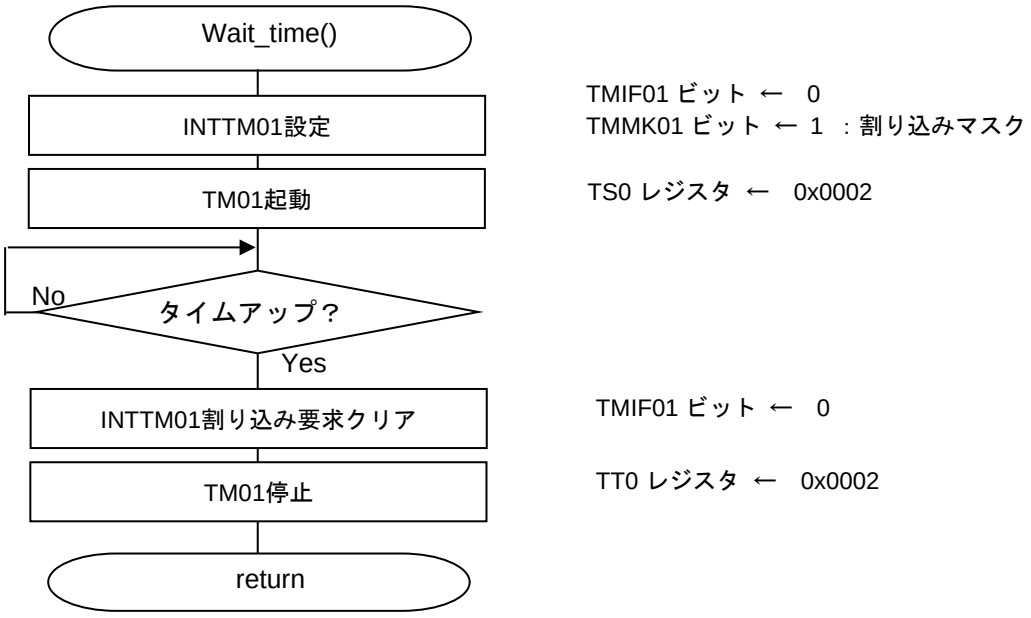
図 4-16 通信状態ポーリング関数



4.6.12 50μs 待ち関数

図 4-17 に 50μs 待ち関数のフローチャートを示します。

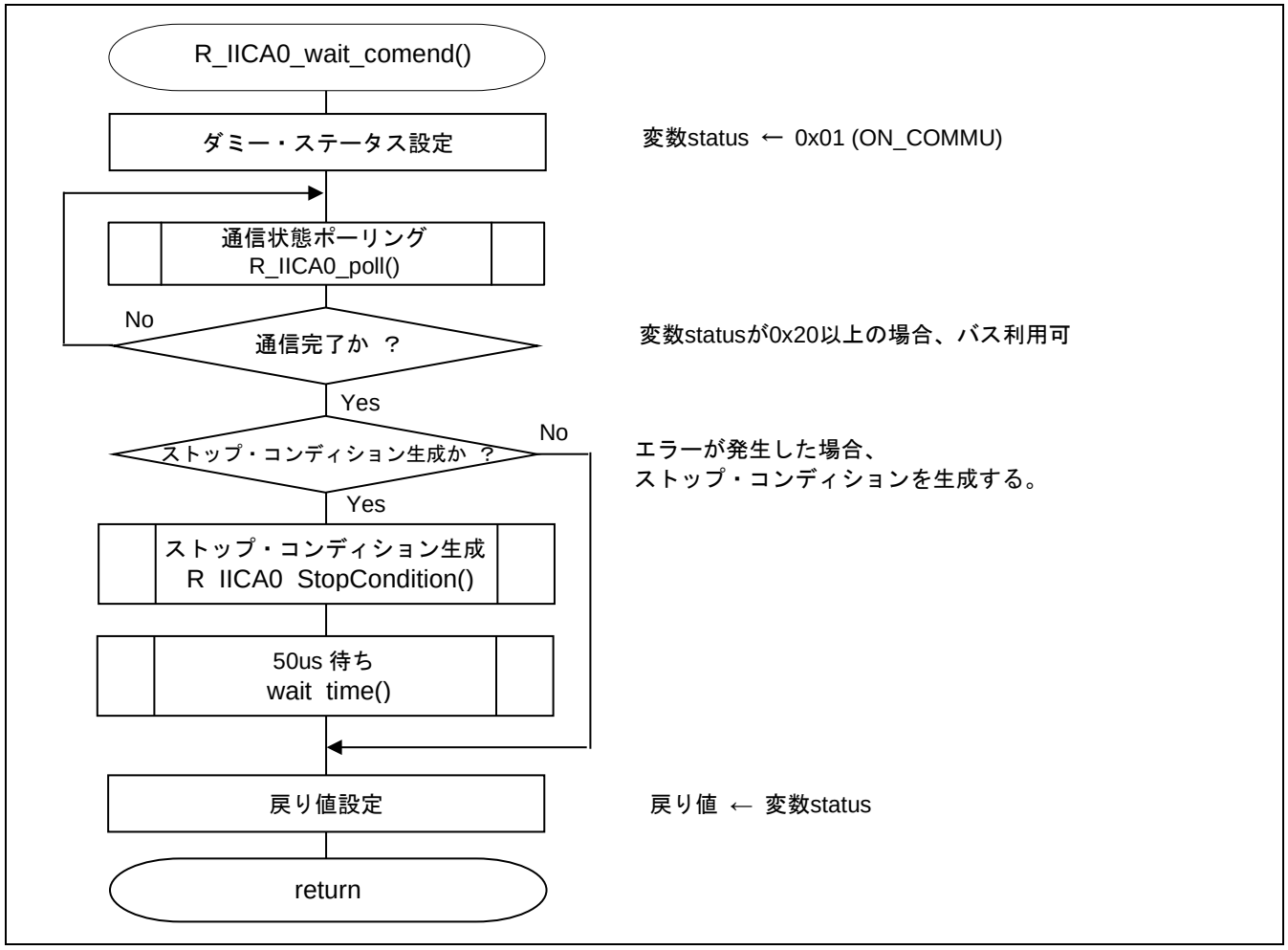
図 4-17　50μs 待ち関数



4.6.13 通信完了待ち関数

図 4-18 に通信完了待ち関数のフローチャートを示します。

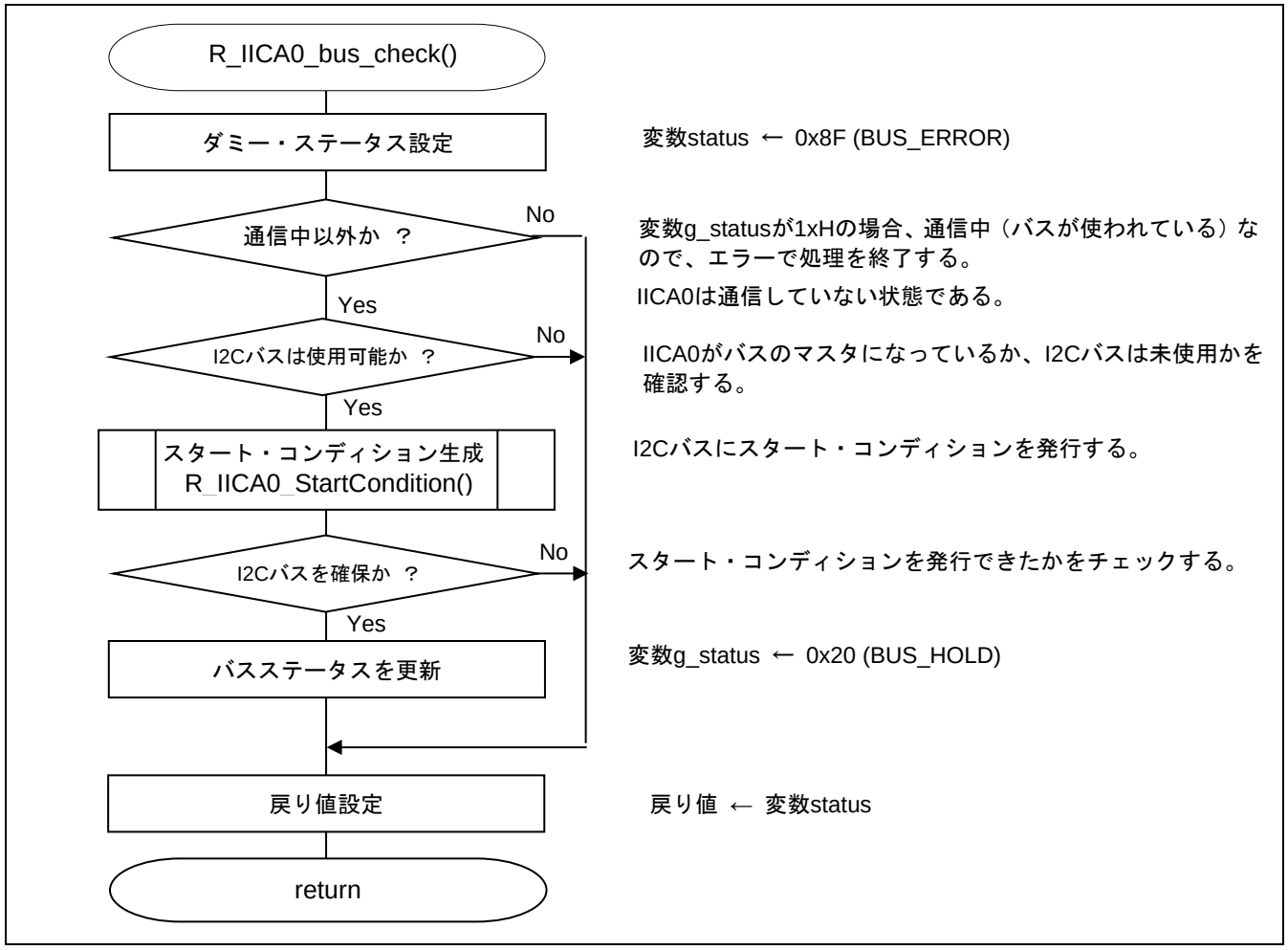
図 4-18　通信完了待ち関数



4.6.14 I2C バスの状態確認関数

図 4-19 に I2C バスの状態確認関数のフローチャートを示します。

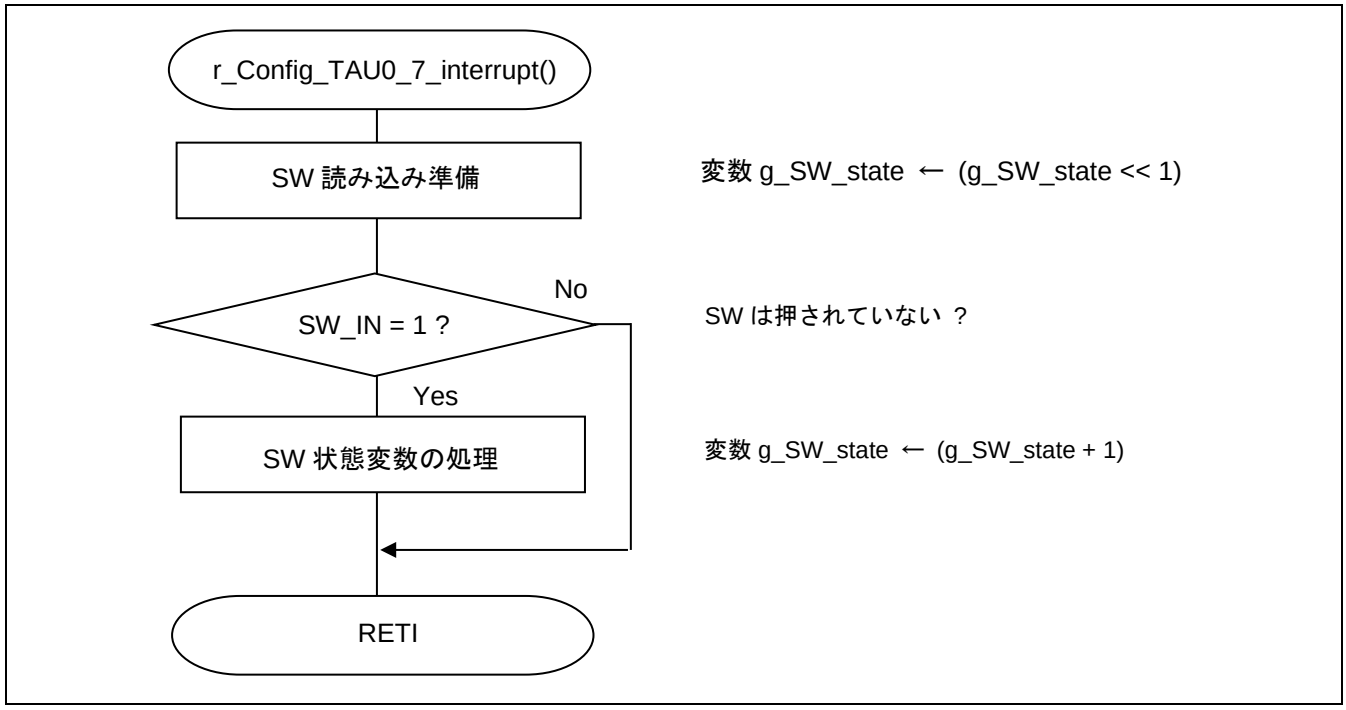
図 4-19 I2C バスの状態確認関数



4.6.15 10ms インターバル・タイマ割り込み処理関数

図 4-20 に 10ms インターバル・タイマ割り込み処理関数のフローチャートを示します。

図 4-20　10ms インターバル・タイマ割り込み処理関数



5. 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。

6. 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 (R01UH0896)
RL78 ファミリ ユーザーズマニュアル ソフトウェア編 (R01US0015)
(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。)

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2021.07.01	-	初版発行

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。