SH7262/SH7264 グループ

I²C バスインタフェース 3
割り込みを用いた EEPROM リードライト例

R01AN0068JJ0101
Rev.1.01
2012.02.10

要旨

本アプリケーションノートでは、SH7262/SH7264 の I²C バスインタフェース 3 (IIC3) のシングルマスタ送受信機能と割り込み機能を使用して EEPROM にリードライトアクセスする例について説明しています。

動作確認デバイス

SH7262/SH7264

以下、総称して「SH7264」として説明します。

目次

1. はじめに

1.1 仕様

- マスタデバイスを SH7264、スレーブデバイスを EEPROM として、EEPROM へデータをライトします。
- 同様に EEPROM からデータをリードします。
- 転送レートを 391kHz に設定しています。
- 送信終了および受信データフルには割り込みを使用します。

【注】EEPROM の電気的特性を満足するように転送レートを設定してください。

1.2 使用機能

- I²C バスインタフェース 3 (IIC3)
- 割り込みコントローラ（INTC）

1.3 適用条件

マイコン	SH7262/SH7264
動作周波数	内部クロック：144 MHz バスクロック：72 MHz 周辺クロック：36 MHz
統合開発環境	ルネサスエレクトロニクス製 High-performance Embedded Workshop Ver.4.07.00
C コンパイラ	ルネサスエレクトロニクス製 SuperH RISC engine ファミリ C/C++コンパイラパッケージ Ver.9.03 Release00
コンパイルオプション	High-performance Embedded Workshop でのデフォルト設定 (-cpu=sh2afpu -fpu=single -object="$(CONFIGDIR)¥$(FILELEAF).obj" -debug -gbr=auto -chgincpath -errorpath -global_volatile=0 -opt_range=all -infinite_loop=0 -del_vacant_loop=0 -struct_alloc=1 –nologo)

1.4 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。合わせて参照してください。

- SH7262/SH7264 グループ　初期設定例
- SH7262/SH7264 グループ　I²C バスインタフェース 3 シングルマスタ送信（EEPROM のライト）
- SH7262/SH7264 グループ　I²C バスインタフェース 3 シングルマスタ受信（EEPROM のリード）

1.5 "L"アクティブ端子（信号）の表記について

端子名（信号名）末尾の＃は"L"アクティブ端子（信号）であることを示します。

2. 応用例の説明

参考プログラムでは、I²C バスインタフェース 3 (IIC3)を使用し、マスタデバイスの SH7264 がスレーブデバイスの EEPROM へデータをライトした後、続けてリードします。送信終了および受信データフルには割り込みを使用します。

2.1 使用機能の動作概要

I²C バスインタフェース 3 (IIC3)は、フィリップス社が提唱する I²C バス（Inter IC Bus）インタフェース方式に準拠しており、サブセット機能を備えています。ただし I²C バスを制御するレジスタの構成が一部フィリップス社のものと異なるため注意してください。

SH7264 の I²C バスインタフェース 3 (IIC3)には以下に示す特長があります。

- I²C バスフォーマットまたはクロック同期式シリアルフォーマットを選択可能
- 連続送信／受信可能
  シフトレジスタ、送信データレジスタ、受信データレジスタがそれぞれ独立しているため、連続送信／受信が可能

表 1に フォーマット別の特長を示します。図 1に IIC3 のブロック図を示します。IIC3 についての詳細は、「SH7262/SH7264 グループ ハードウェアマニュアル　I²Cバスインタフェース 3」の章を参照してください。

表1　フォーマット別の特長

フォーマット	特長
I2C バスフォーマット	• マスタモードでは開始条件、停止条件を自動生成 • 受信時、アクノリッジの出力レベルを選択可能 • 送信時、アクノリッジビットを自動ロード • ビット同期／ウェイト機能内蔵 マスタモードではビットごとに SCL の状態をモニタして自動的に同期を取ります。転送準備ができていない場合には、SCL をローレベルにして待機させます。 • 割り込み要因：6 種類 ① 送信データエンプティ（スレーブアドレス一致時を含む） ② 送信終了 ③ 受信データフル（スレーブアドレス一致時を含む） ④ アービトレーションロスと ⑤ NACK 検出 ⑥ 停止条件検出 • 送信データエンプティ割り込みと受信データフル割り込みにより、ダイレクトメモリアクセスコントローラ（DMAC）を起動させてデータを転送可能。 • バスを直接駆動可能 SCL、SDA の 2 端子は、バス駆動機能選択時 NMOS オープンドレイン出力
クロック同期式シリアルフォーマット	• 割り込み要因：4 種類 ① 送信データエンプティ ② 送信終了 ③ 受信データフル ④オーバランエラー • 送信データエンプティ割り込みと受信データフル割り込みにより、ダイレクトメモリアクセスコントローラ（DMAC）を起動させてデータを転送可能。

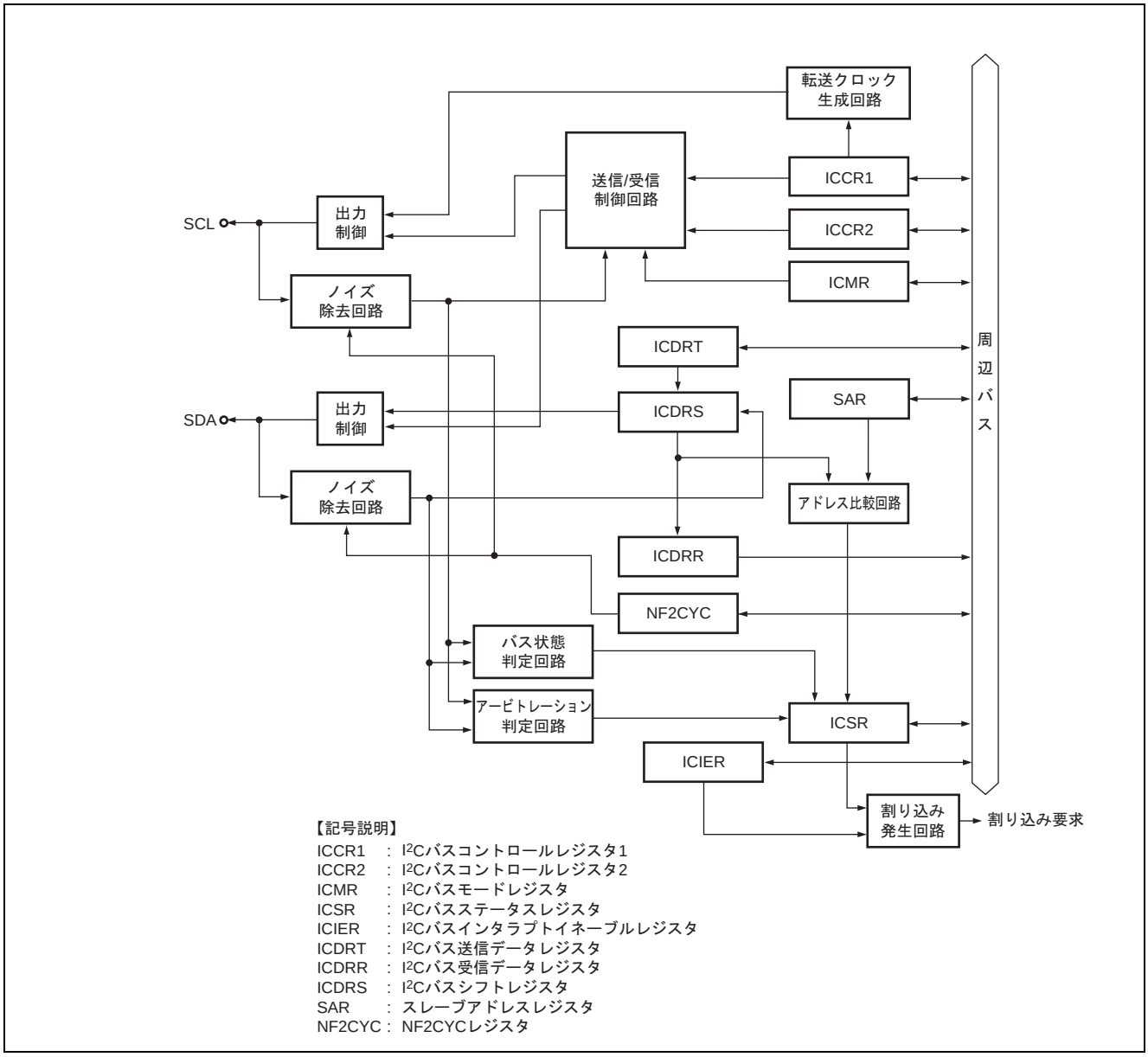


図1　IIC3 のブロック図

2.2　使用機能の設定手順

ここでは、IIC3 の初期設定手順について説明します。転送レートはEEPROMの電気的特性を満足するように設定してください。参考プログラムではPφ/92 を選択しています。図 2に IIC3 の初期設定フロー例を示します。なお、各レジスタ設定の詳細は、「SH7262/SH7264 グループ　ハードウェアマニュアル」を参照してください。

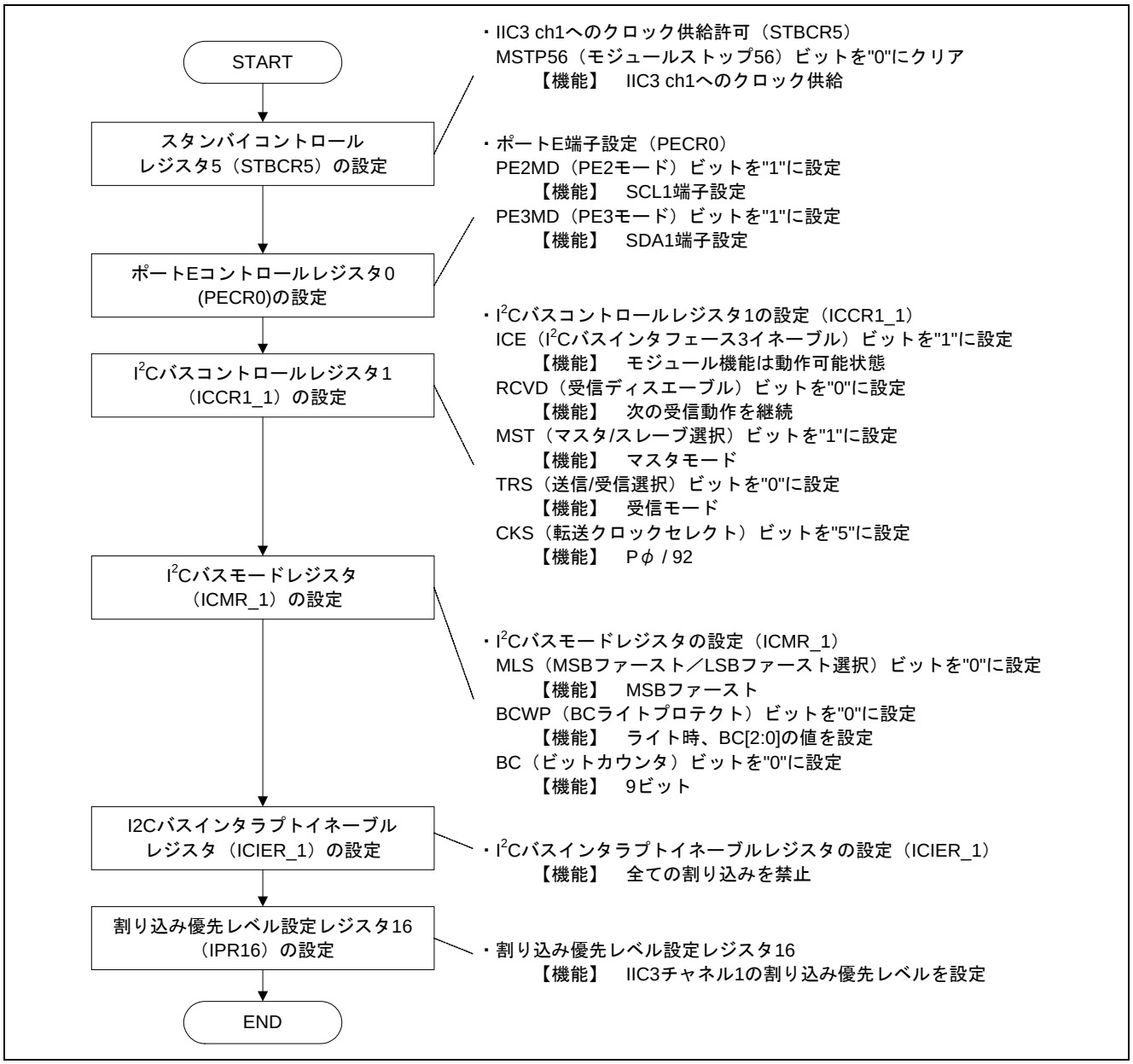


図2　IIC3 の初期設定フロー例

2.3 参考プログラムの動作

参考プログラムでは、IIC3 をマスタ送信モードに設定し、10 バイトのページライトを行います。2 バイト目以降は、送信終了割り込み（TEI）を使用してデータをレジスタに書き込みます。その後、IIC3 をマスタ受信モードに設定し 10 バイトのシーケンシャルリードを行います。受信データフル割り込み（RXI）を使用してレジスタからデータを読み出します。

デバイスコードについては EEPROM のデータシートを確認してください。参考プログラムでは、デバイスコード"B'1010"を使用します。

参考プログラムでは、デバイスアドレス"B'000"を使用します。デバイスアドレスについては、EEPROM のデータシートを確認してください。

メモリアドレスはEEPROMの書き込みまたは読み込み開始アドレスを示し、ライトまたはリードする度にEEPROM側でアドレスがインクリメントされます。図 3にページライト動作図、図 4にシーケンシャルリード動作図を示します。また、図 5に参考プログラムの動作環境を示します。

参考プログラムは、ルネサス エレクトロニクス製 EEPROM（型名：R1EX24128ASA00A）で動作を確認しています。

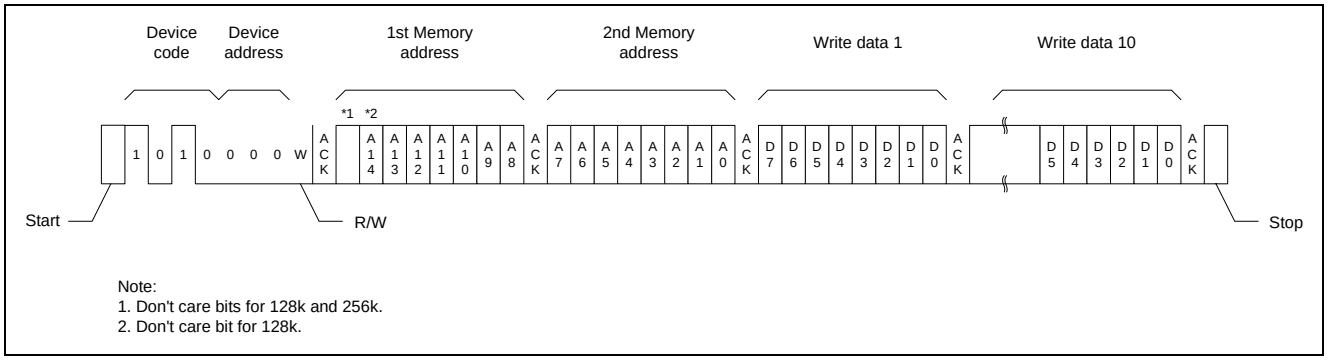


図3　ページライト動作図

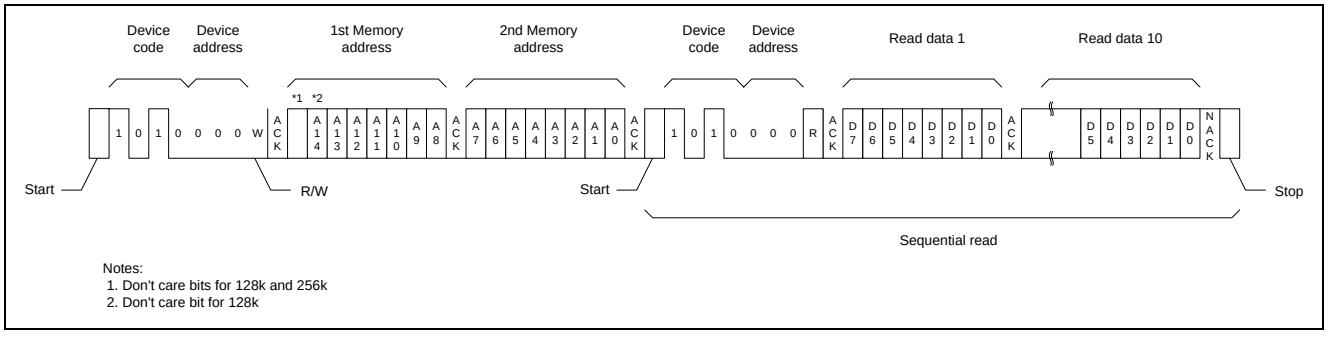


図4　シーケンシャルリード動作図

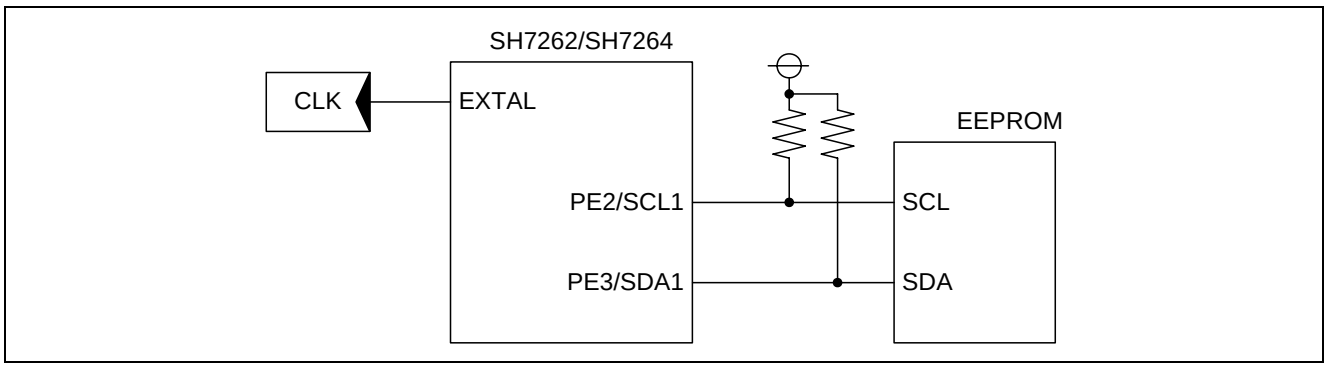


図5 参考プログラムの動作環境

2.4　参考プログラムの処理手順

表 2、表 3、図 6に参考プログラムにおけるレジスタ設定およびマクロ定義、構造体を示します。図 7～図 17に参考プログラムの処理フローを示します。

表2　参考プログラムのレジスタ設定（初期設定時）

レジスタ名	アドレス	設定値	機能
スタンバイコントロール レジスタ 5 (STBCR5)	H'FFFE 0410	H'00	MSTP56="0": IIC3 の ch1 は動作
I²C バスコントロールレジスタ 1 (ICCR1_1)	H'FFFE E400	H'B5	ICE = "1": SCL/SDA はバス駆動状態 RCVD = "0": 次の受信動作を継続 MST = "1"、TRS = "1": マスタ送信モード CKS = "B'0101": 転送レート Pφ/92
I²C バスモードレジスタ (ICMR_1)	H'FFFE E402	H'30	MLS = "0": MSB ファースト BCWP = "0": ライト時、BC の値を設定 BC = "B'000":9 ビット
I²C バスインタラプトイネーブル レジスタ（ICIER_1）	H'FFFE E403	H'00	TIE = "0": 送信データエンプティ 割り込み要求（TXI）の禁止 TEIE = "0": 送信終了割り込み要求 （TEI）の禁止 RIE = "0": 受信データフル割り込み要求 （RXI）の禁止 NAKIE = "0": NACK 受信割り込み要求 （NAKI）の禁止 STIE = "0": 停止条件検出割り込み （STPI）の禁止
割り込み優先レベル設定 レジスタ 16	H'FFFE 0C14	H'0050	IIC3 チャネル 1 の割り込み優先レベルを 5 に設定

表3　参考プログラムで使用しているマクロ定義

マクロ定義	設定値	機能
EEPROM_MEM_ADDR	H'0000	EEPROM 開始アドレス
DEVICE_CODE	H'A0	デバイスコード
DEVICE_ADDR	H'00	デバイスアドレス
IIC_DATA_WR	H'00	ライトコード
IIC_DATA_RD	H'01	リードコード
DATA_LENGTH	10	データ転送サイズ
E_OK	0	正常終了
E_ERR	−1	エラー終了
IIC3_IDOL	0	IIC3 の制御がアイドル状態であることを示す
IIC3_NACK	1	IIC3 の制御が NACK 受信で終了したことを示す
IIC3_SEND	2	IIC3 の制御が送信中であることを示す
IIC3_RECV	3	IIC3 の制御が受信中であることを示す

■IIC3の制御情報を定義する構造体変数（iic3_info）

```
typedef enum{
        IIC3_IDOL=0,                    /* アイドル状態(送受信完了状態) */
        IIC3_NACK,                      /* NACK受信による終了 */
        IIC3_SEND,                      /* 送信中(送信系の割り込み使用中) */
        IIC3_RECV,                      /* 受信中(受信系の割り込み使用中) */
}IIC3_MODE;

typedef struct{
        volatile IIC3_MODE mode;        /* IIC3の制御状態 */
        unsigned char *buffer;          /* 連続送受信時のアクセス先を管理するポインタ */
        int count;                      /* 連続送受信時の残りデータサイズ */
}IIC3_INFO;

static IIC3_INFO iic3_info;
```

■iic3_info.modeの状態遷移図

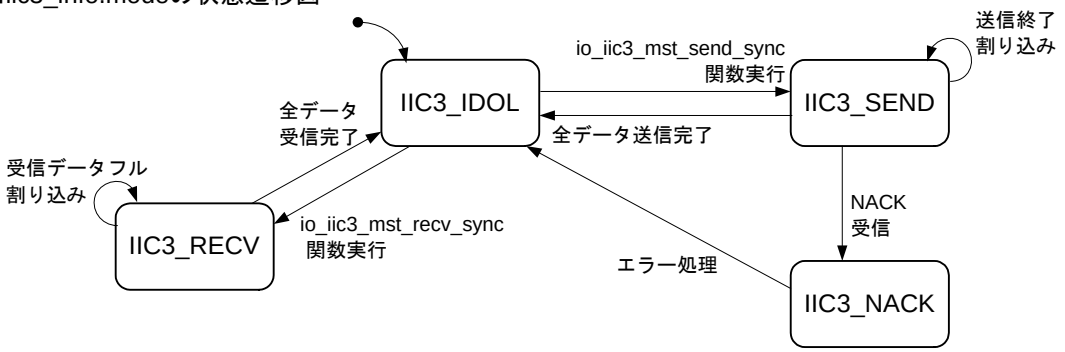


図6　参考プログラムで使用する構造体

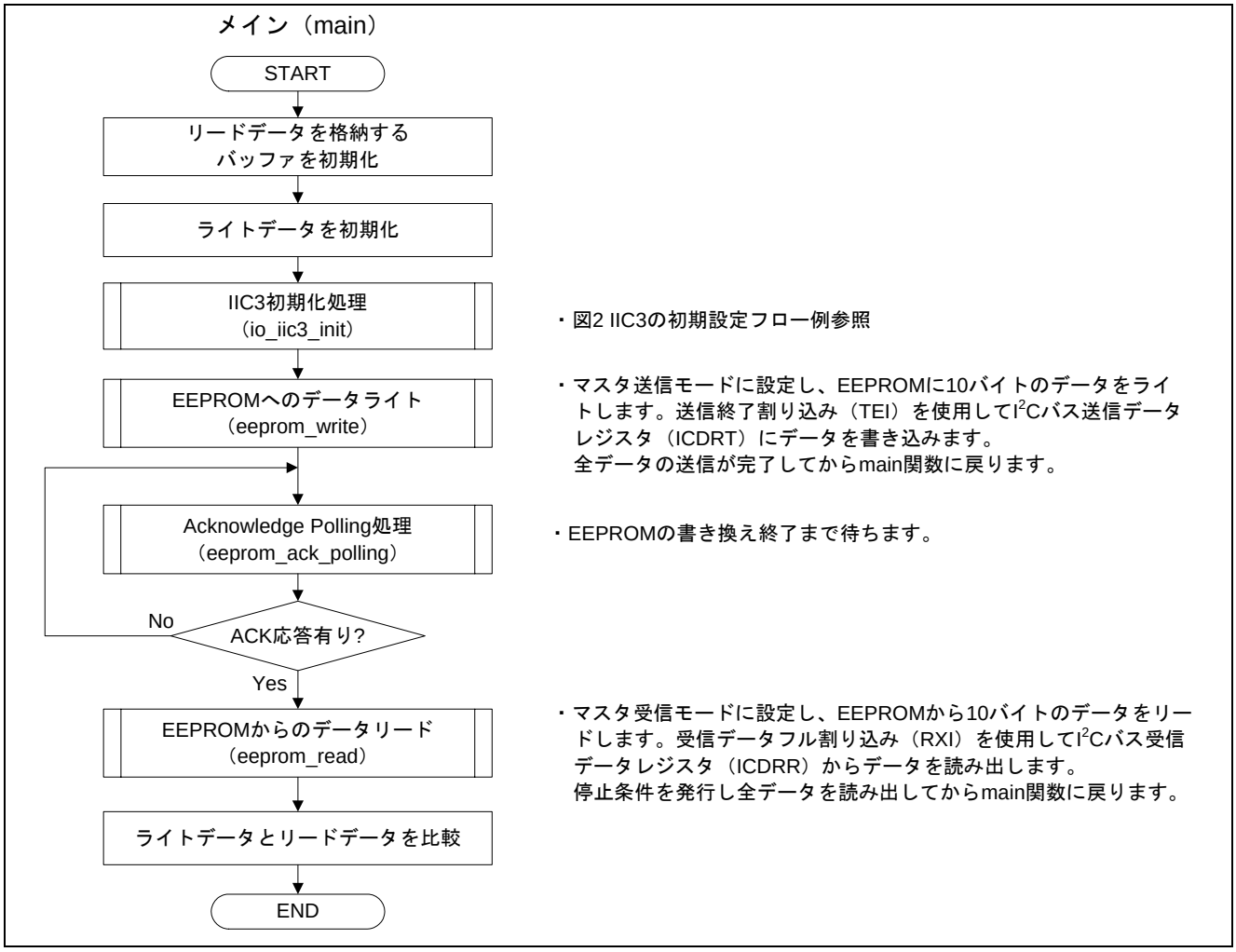


図7　参考プログラムの処理フロー（1）

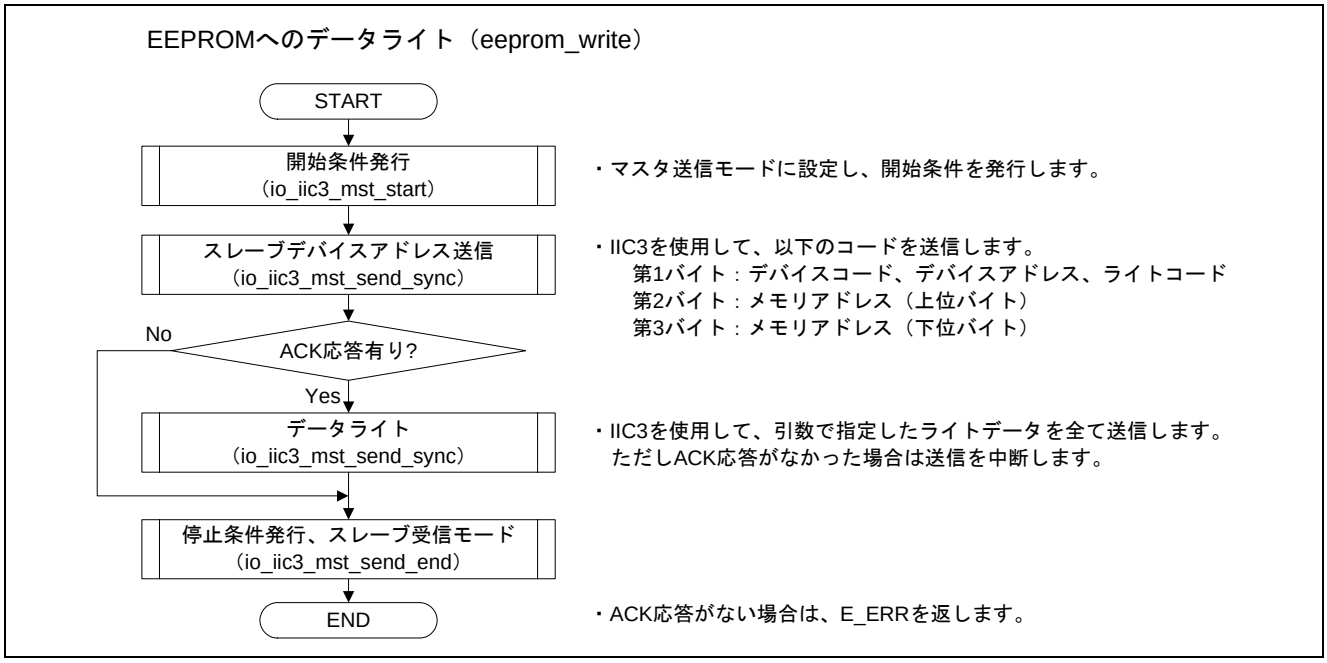


図8　参考プログラムの処理フロー（2）

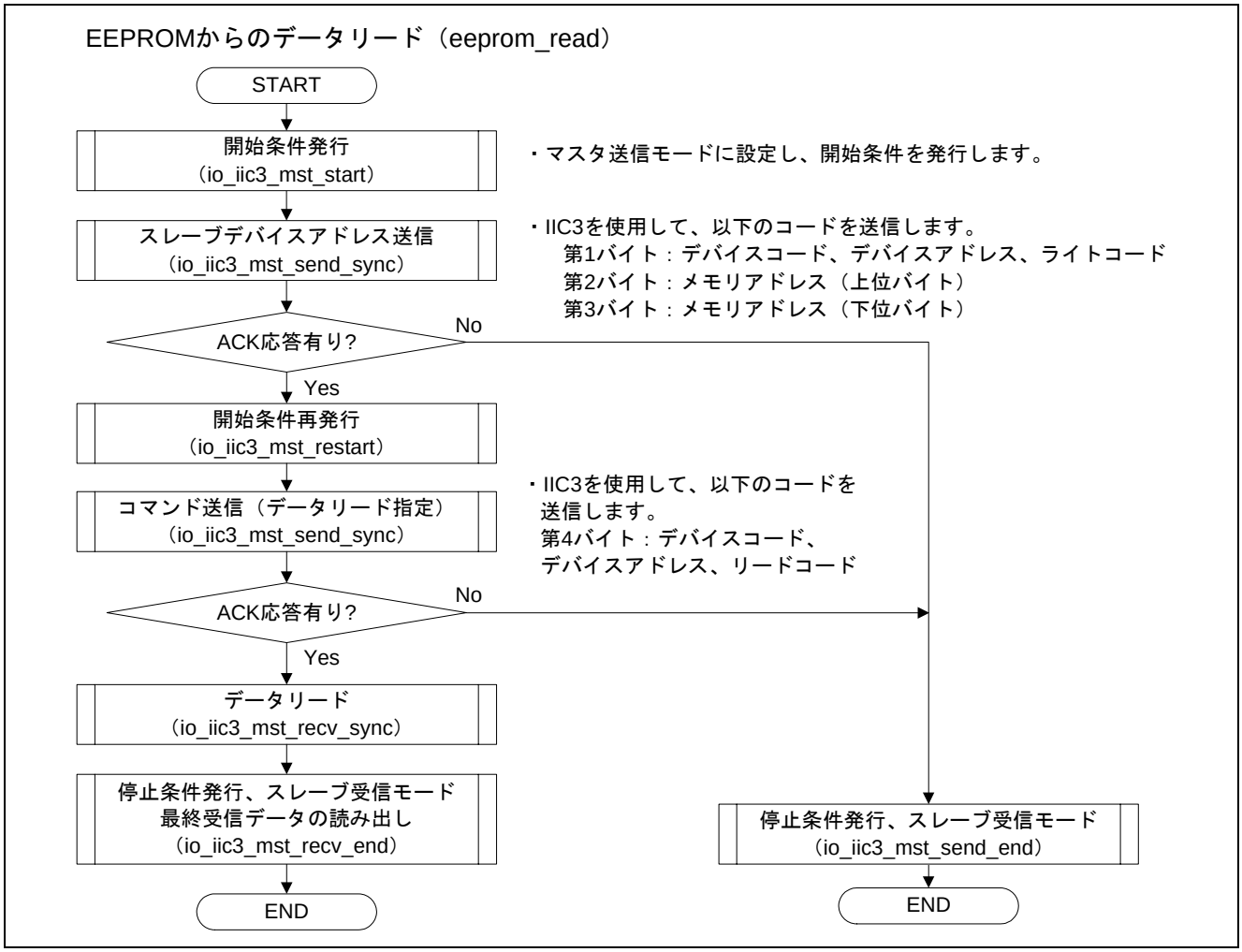


図9　参考プログラムの処理フロー（3）

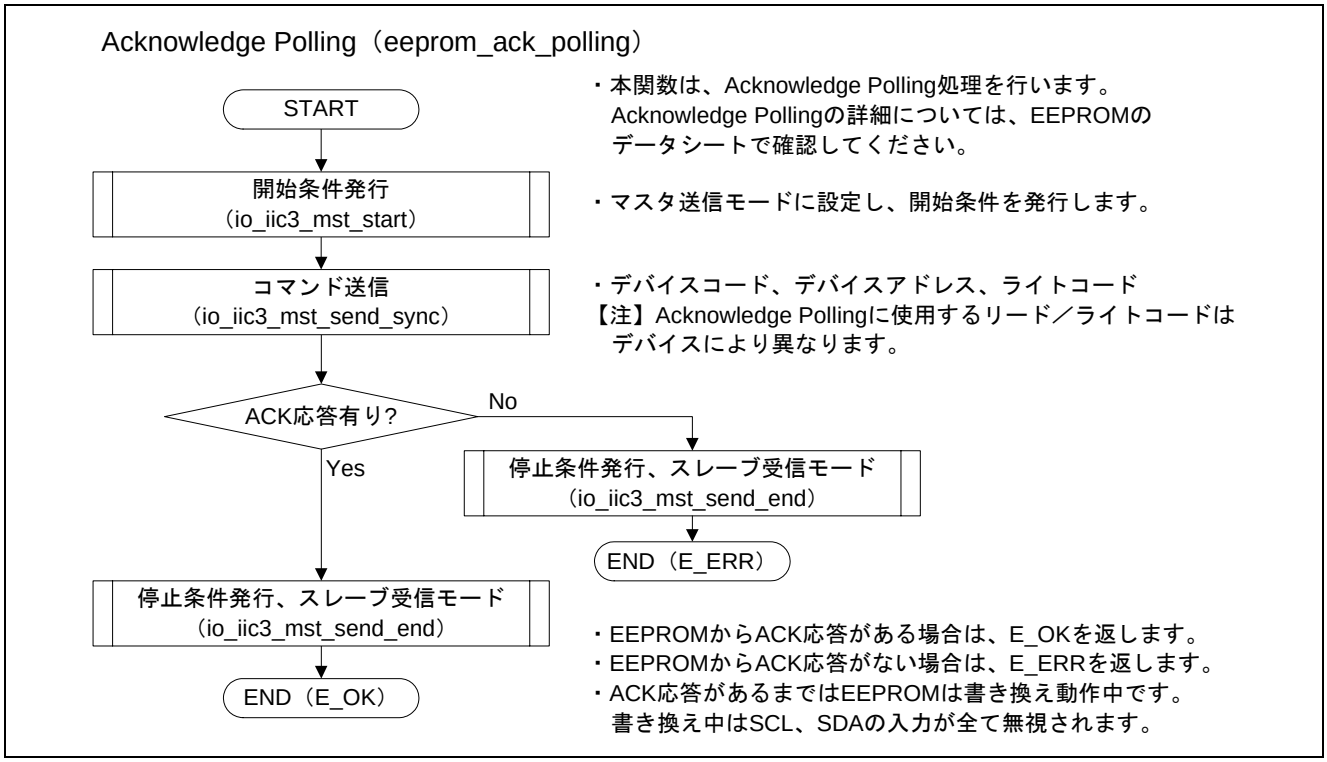


図10　参考プログラムの処理フロー（4）

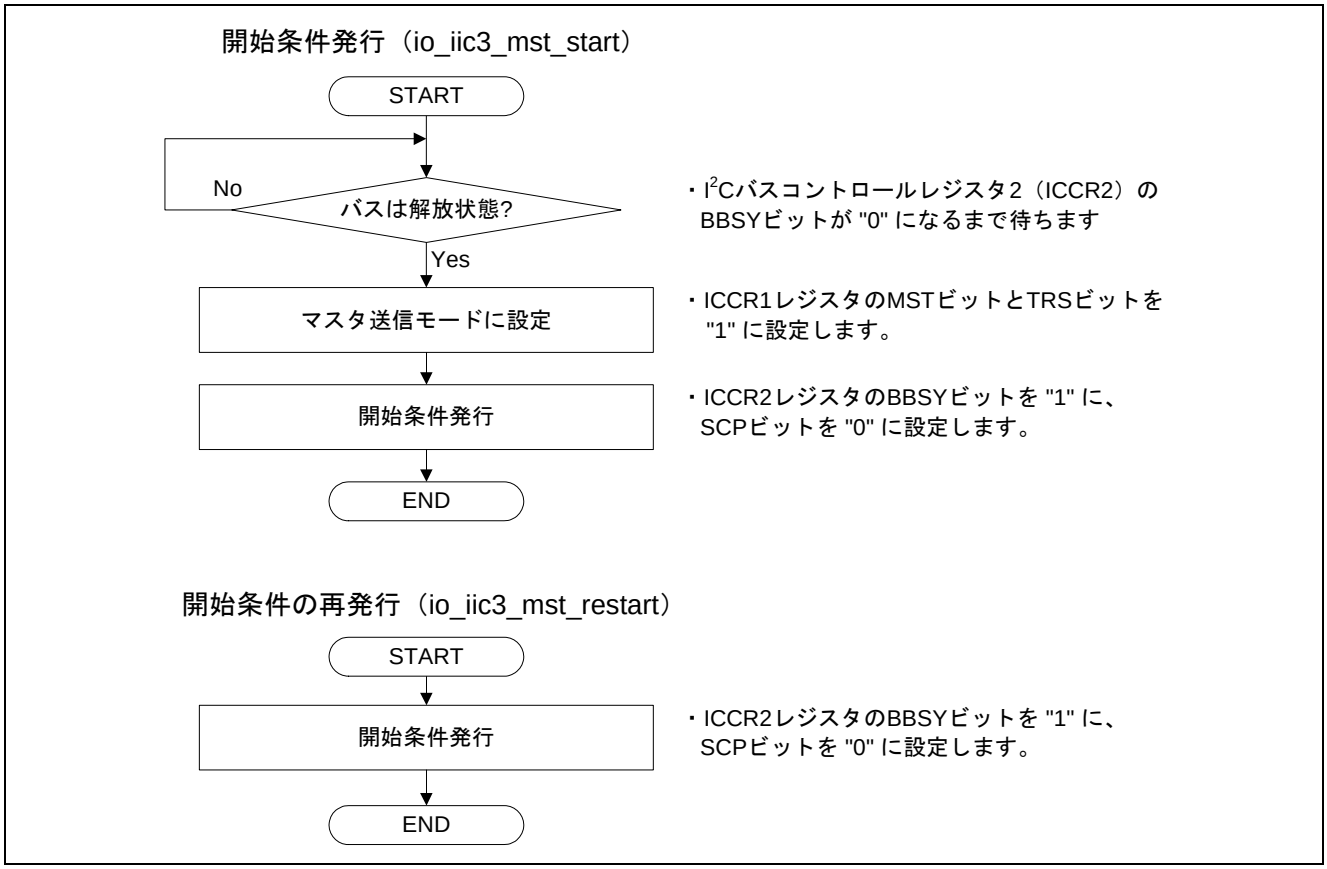
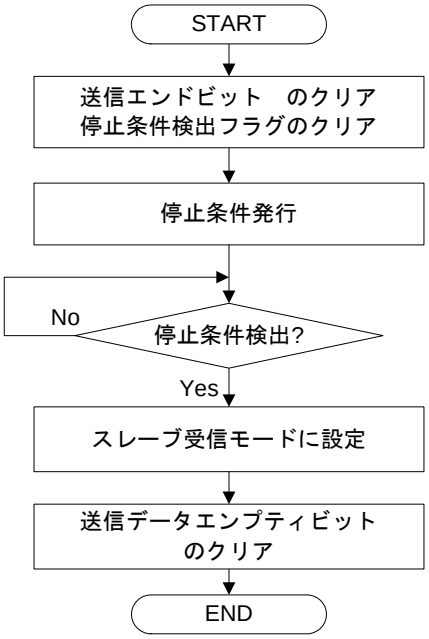


図11　参考プログラムの処理フロー（5）

停止条件発行、スレーブ受信モード（io_iic3_mst_send_end）



・本関数はマスタ送信後の停止条件発行に使用します。

・I²Cバスステータスレジスタ（ICSR）の
TENDビットとSTOPビットをクリアします。

・ICCR2レジスタのBBSYビットとSCPビットを
"0" に設定します。

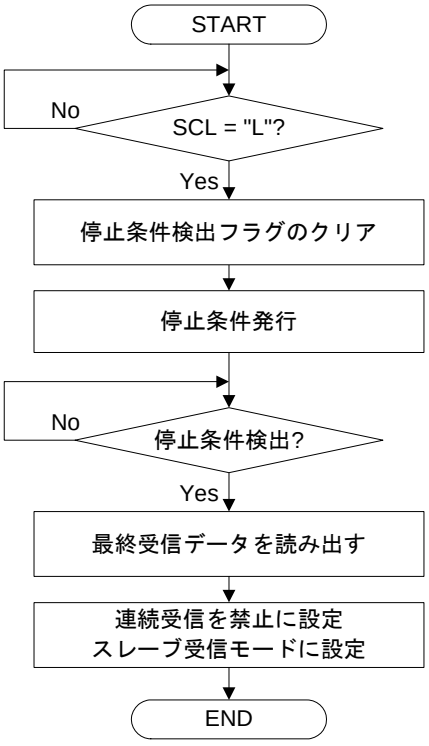
・STOPビットが "1" になるまで待ちます。

・ICCR1レジスタのMSTビットとTRSビットを
"0" に設定します。

・ICSRレジスタのTDREビットをクリアします。

図12　参考プログラムの処理フロー（6）

停止条件発行、スレーブ受信モード、最終受信データの読み出し
（io_iic3_mst_recv_end）



・本関数はマスタ受信後の停止条件発行に使用します。

・SCLの9クロック目の立ち下がりを待ちます。
（Technical Update [TN-MC*-A020A/J]参照）

・I²Cバスステータスレジスタ（ICSR）の
TENDビットとSTOPビットをクリアします。

・ICCR2レジスタのBBSYビットとSCPビットを
"0" に設定します。

・STOPビットが "1" になるまで待ちます。

・ICCR1レジスタのRCVDビットとMSTビット、
TRSビットを"0" に設定します。

図13　参考プログラムの処理フロー（7）

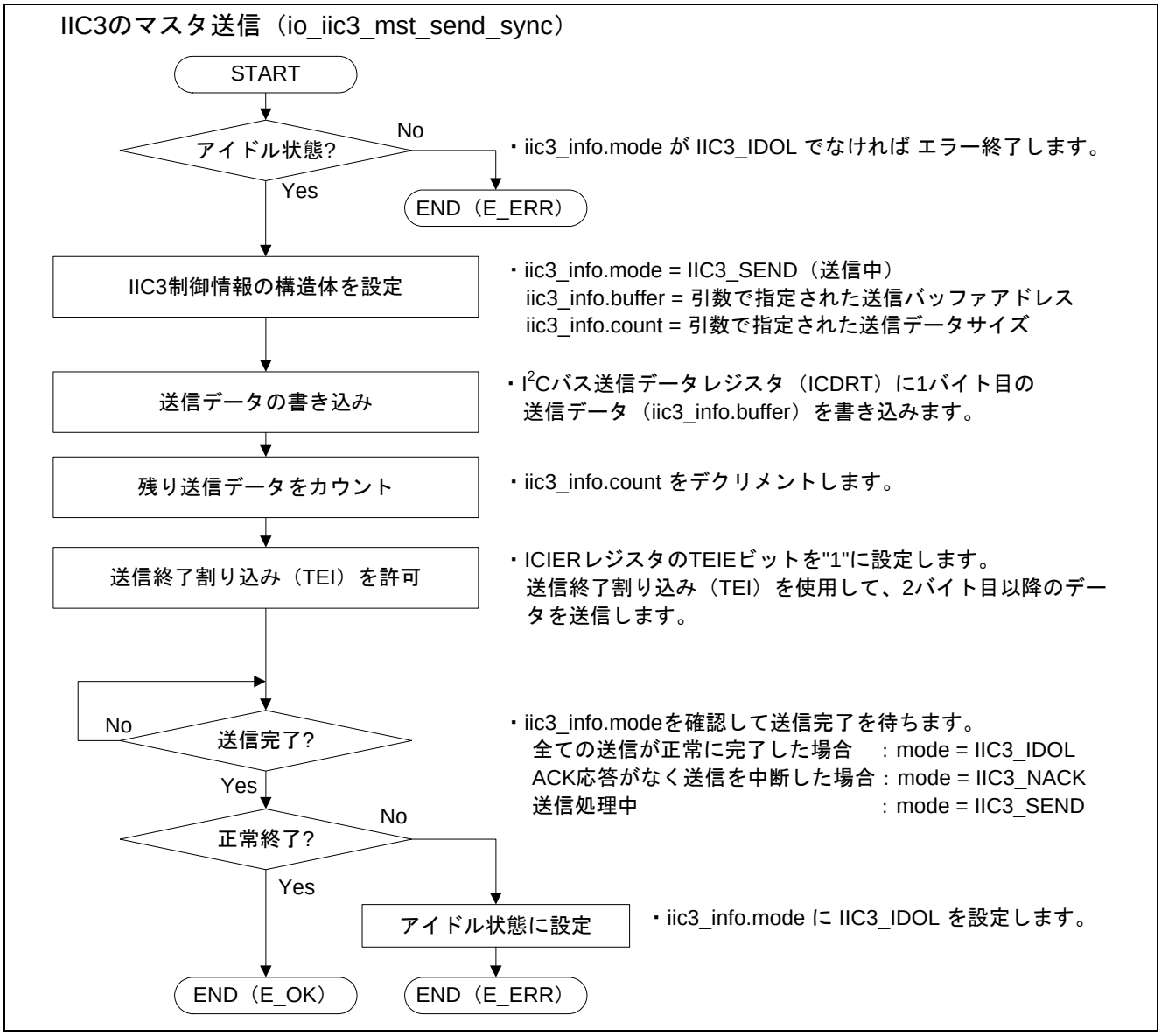


図14　参考プログラムの処理フロー（8）

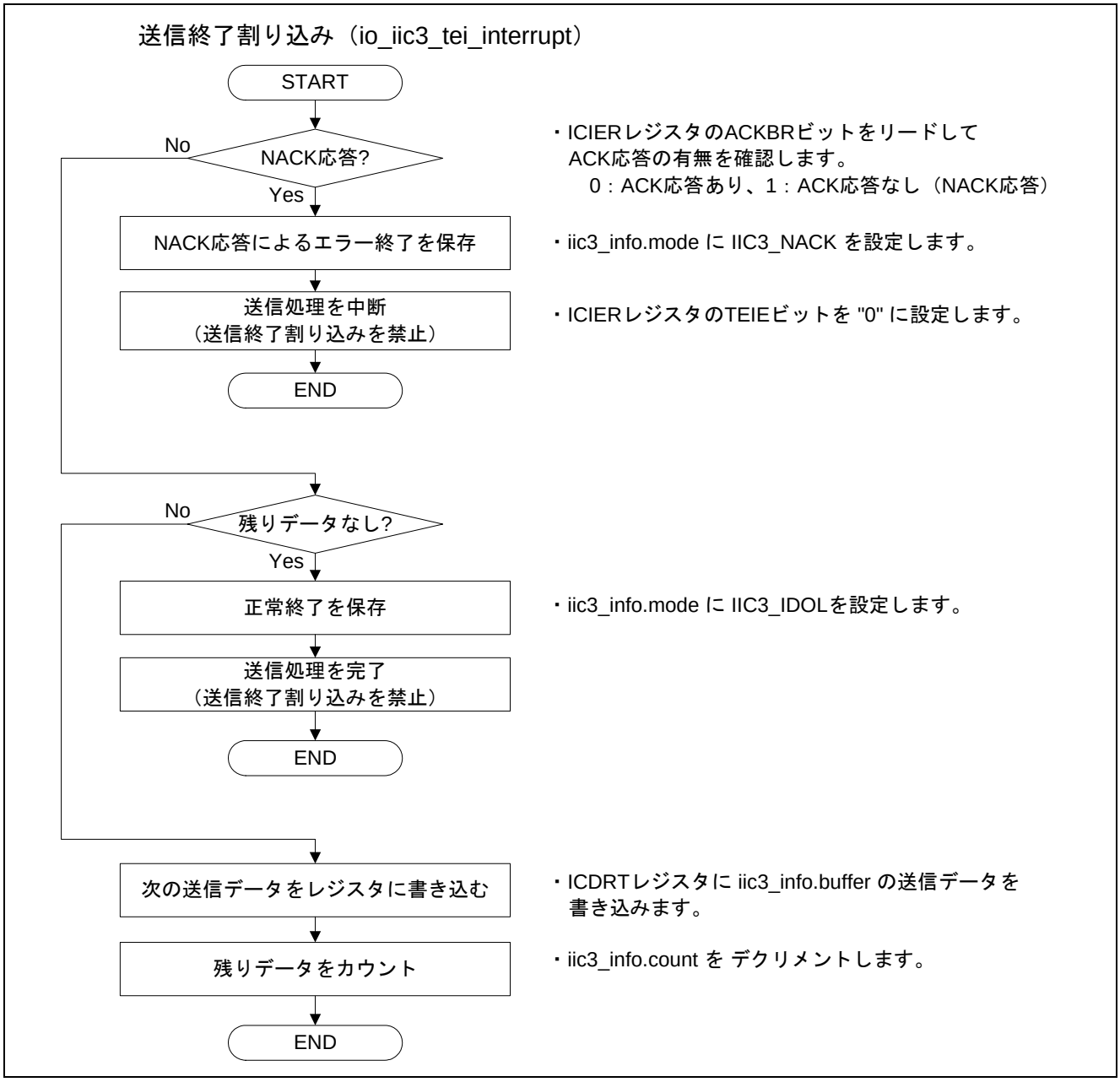


図15　参考プログラムの処理フロー（9）

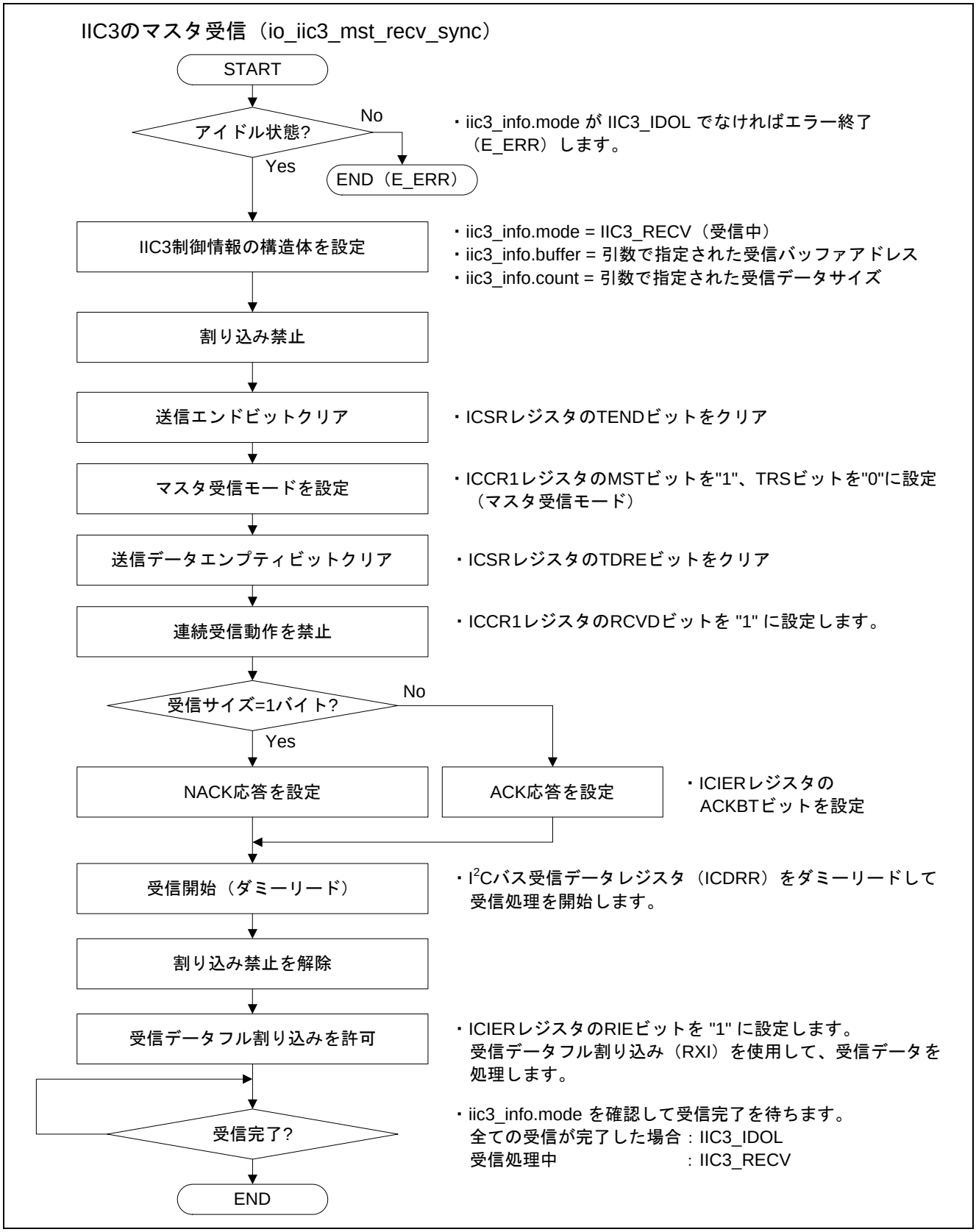


図16　参考プログラムの処理フロー（10）

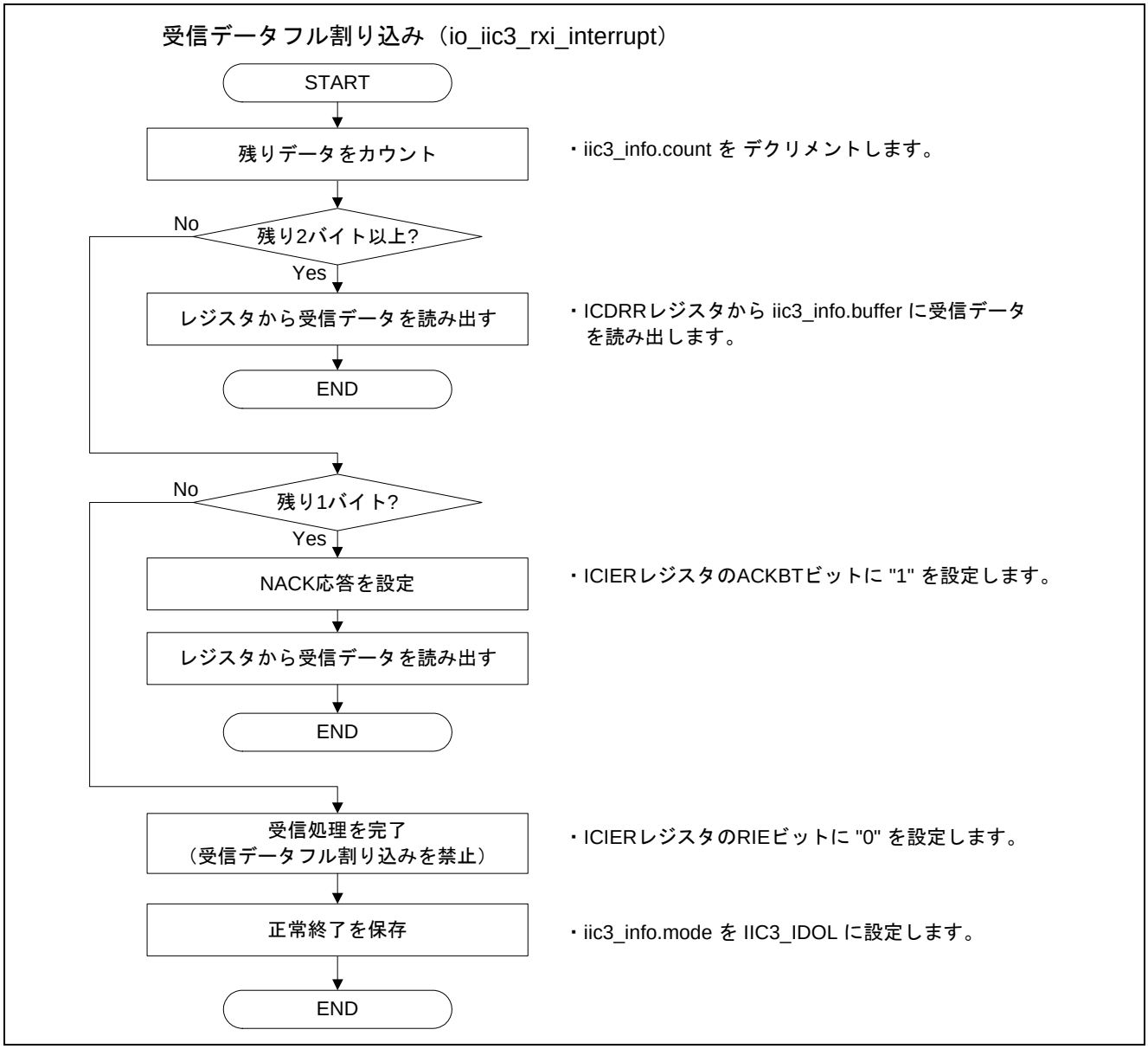


図17　参考プログラムの処理フロー（11）

2.5　マスタ受信モード時の注意

8 クロック目の立ち下がり付近で I²C バス受信データレジスタ（ICDRR）をリードすると、受信データが取得できない場合があります。

また、受信バッファフルかつ 8 クロック目の立ち下がり付近で I²C バスコントローラレジスタの受信ディスエーブルビット（RCVD）を 1 に設定すると、停止条件の発行ができなくなる場合があります。そのため、以下の 1.もしくは 2.どちらかの方法で対応する必要があります。

参考プログラムでは、RCVD を 1 にセットして 1 バイトごとの通信を行っています。

1. マスタ受信モードで ICDRR をリードする処理は 8 クロックの立ち上がりまでに行ってください。
2. マスタ受信モードは RCVD を 1 にセットし、1 バイトごとの通信で処理してください。

2.6　マスタ受信モード、ACKBT 設定時の注意

マスタ受信モード動作時、連続転送している最終データの 8 つ目の SCL が立ち下がる前に ACKBT を設定してください。スレーブ送信側デバイスがオーバランする可能性があります。

参考プログラムでは、RCVD を 1 にセットして 1 バイトごとの通信を行っているため、本注意事項には該当しません。

2.7　マスタ受信モード、停止条件発行または開始条件の再発行時の注意

停止条件発行または開始条件の再発行が SCL の 9 クロック目の立ち下がりと重なった場合、9 クロック目の後に SCL が 1 クロック余分に出力されます。そのため、マスタ受信完了後、SCL の 9 クロック目の立ち下がりを確認してから、停止条件を発行または開始条件を再発行してください。

SCL の 9 クロック目の立ち下がりは次の方法で確認します。

- ICSR レジスタの RDRF ビット（受信データレジスタフルフラグ）が 1 になったことを確認後、ICCR2 レジスタの SCLO ビット（SCL モニタフラグ）が 0（SCL 端子は "L"）になったことを確認してください。

本注意事項の詳細については、ルネサス テクニカルアップデート（発行番号：TN-MC*-A020A/J）を参照してください。

2.8　IICRST 使用上の注意

I²C バス動作中に、ICCR1 レジスタの ICE ビットに 0 をライトもしくは ICCR2 レジスタの IICRST ビットに 1 をライトすると、ICCR2 レジスタの BBSY ビットと ICSR レジスタの STOP ビットは不定となります。

本注意事項の詳細についてはルネサス テクニカルアップデート（発行番号：TN-MC*-A022A/J）を参照してください。

3. 参考プログラム例

3.1 参考プログラムについての補足

SH7264 は、製品によって大容量内蔵 RAM の容量が 1MB または 640KB と異なるため、参考プログラムのセクション配置やレジスタの設定を一部変更する必要があります。そのため本アプリケーションノートでは 1MB 用と 640KB 用の 2 つのワークスペースを用意しています。

640KB 版はライトプロテクトを解除しなければ保持用内蔵 RAM へ書き込むことができないため、640KB 版のワークスペースは、システムコントロールレジスタ 5（SYSCR5）にライトプロテクトの解除を設定しています。

使用する製品を確認した上で、対応するワークスペースを使用してください。

3.2 サンプルプログラムリスト"main.c（1）"

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Electronics Corporation and is only
5     *   intended for use with Renesas products.  No other uses are authorized.
6     *
7     *   This software is owned by Renesas Electronics Corporation and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ***********************************************************************
28    *   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29    *""FILE COMMENT""*********** Technical reference data *************************
30    *   System Name : SH7264 Sample Program
31    *   File Name   : main.c
32    *   Abstract    : IIC3 割り込みを用いた EEPROM リードライト例
33    *   Version     : 1.00.00
34    *   Device      : SH7262/SH7264
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                          (Ver.9.03 Release00).
38    *   OS          : None
39    *   H/W Platform: M3A-HS64G50(CPU board)
40    *   Description :
41    ***********************************************************************
42    *   History     : Aug.17,2010 Ver.1.00.00
43    *""FILE COMMENT END""*************************************************************/
44    #include <machine.h>
45    #include "iodefine.h"      /* SH7264 iodefine  */
46
```

3.3　サンプルプログラムリスト"main.c（2）"

```
47    /* ==== symbol definition ==== */
48    #define EEPROM_MEM_ADDR 0x0000
49    #define DEVICE_CODE 0xA0   /* EEPROM device code   :b'1010    */
50    #define DEVICE_ADDR 0x00   /* EEPROM device address:b'000     */
51    #define IIC_DATA_WR 0x00   /* Data write code      :b'0       */
52    #define IIC_DATA_RD 0x01   /* Data read code       :b'1       */
53    #define DATA_LENGTH 10
54
55    #define E_OK 0
56    #define E_ERR -1
57
58    /* ==== RAM allocation variable declaration ==== */
59    unsigned char ReadData[DATA_LENGTH];
60    unsigned char WriteData[DATA_LENGTH];
61
62    /* ==== prototype declaration ==== */
63    void main(void);
64    int eeprom_write(unsigned char d_code,unsigned char d_adr,unsigned short w_adr,
65                                  unsigned int w_size,unsigned char* w_buf);
66    int eeprom_read(unsigned char d_code,unsigned char d_adr,unsigned short r_adr,
67                                 unsigned int r_size,unsigned char* r_buf);
68    int eeprom_ack_polling(unsigned char d_code,unsigned char d_adr);
69    int  io_iic3_init(void);
70    void io_iic3_mst_start(void);
71    void io_iic3_mst_restart(void);
72    void io_iic3_mst_send_end(void);
73    unsigned char io_iic3_mst_recv_end(void);
74    int  io_iic3_mst_send_sync( unsigned char *buffer, int size);
75    int  io_iic3_mst_send( unsigned char *buffer, int size);
76    int  io_iic3_mst_recv_sync( unsigned char *buffer, int size);
77    int  io_iic3_mst_recv( unsigned char *buffer, int size);
78
79    /*""FUNC COMMENT""*****************************************************************
80     * ID          :
81     * Outline     : サンプルプログラムメイン
82     *-------------------------------------------------------------------------------
83     * Include     :
84     *-------------------------------------------------------------------------------
85     * Declaration  : void main(void);
86     *-------------------------------------------------------------------------------
87     * Description  : IIC3 のマスタ送信モードを用いて、EEPROM にデータをライトします。
88     *              : IIC3 のマスタ受信モードを用いて、EEPROM からデータをリードします。
89     *              : データの送受信には、送信終了割り込みおよび受信データフル割り込み
90     *              :を使用します。
91     *-------------------------------------------------------------------------------
92     * Argument    : void
93     *-------------------------------------------------------------------------------
94     * Return Value : void
95     *-------------------------------------------------------------------------------
96     * Note        : None
97     *""FUNC COMMENT END""*************************************************************/
```

3.4 サンプルプログラムリスト"main.c（3）"

```
98    void main(void)
99    {
100     int i,ack;
101
102     /* ==== データ格納場所クリア ==== */
103     for(i=0;i<DATA_LENGTH;i++){
104       ReadData[i] = 0x00;
105     }
106     /* ==== 書き込みデータ作成 ==== */
107     for(i=0;i<DATA_LENGTH;i++){
108       WriteData[i] = DATA_LENGTH+i;
109     }
110
111     /* ==== IIC3 初期設定 ==== */
112     io_iic3_init();
113
114     /* ==== EEPROM へのデータライト ==== */
115     eeprom_write(DEVICE_CODE,    /* デバイスコード       */
116                  DEVICE_ADDR,    /* デバイスアドレス     */
117                  0x0000,         /* 書き込み開始アドレス */
118                  sizeof(WriteData), /* 書き込みデータサイズ */
119                  WriteData);     /* データ格納場所       */
120
121     /* ==== Acknowledge Polling ==== */
122     while( eeprom_ack_polling(DEVICE_CODE, DEVICE_ADDR) != E_OK){
123       /* EEPROM の内部書き換えが完了するまで待つ */
124     }
125
126     /* ==== EEPROM からのデータリード ==== */
127     eeprom_read(DEVICE_CODE,     /* デバイスコード       */
128                 DEVICE_ADDR,     /* デバイスアドレス     */
129                 0x0000,          /* 読み出し開始アドレス */
130                 sizeof(ReadData),  /* 読み出しデータサイズ */
131                 ReadData);       /* データ格納場所       */
132
133     /* ==== 結果比較 ==== */
134     for(i=0; i<DATA_LENGTH; i++){
135       if( WriteData[i] != ReadData[i] ){
136           while(1){
137               /* error */
138           }
139       }
140     }
141     while(1){
142       /* Loop */
143     }
144   }
```

3.5　サンプルプログラムリスト"main.c（4）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : EEPROM へのデータライト
 *-----------------------------------------------------------------------------
 * Include     :
 *-----------------------------------------------------------------------------
 * Declaration  : int eeprom_write(unsigned char d_code,unsigned char d_adr,
 *              : unsigned short w_adr,unsigned int w_size,unsigned char* w_buf);
 *-----------------------------------------------------------------------------
 * Description  : デバイスコード d_code、デバイスアドレス d_adr で指定した EEPROM へ
 *              : w_buf で指定した領域のデータを w_size バイト分書き込みます。
 *              : EEPROM のメモリアドレスは w_adr で指定します。
 *-----------------------------------------------------------------------------
 * Argument    : unsigned char d_code ; I : デバイスコード
 *              : unsigned char d_adr  ; I : デバイスアドレス
 *              : unsigned short w_adr ; I : 書き込み開始アドレス
 *              : unsigned int w_size  ; I : 書き込みデータサイズ
 *              : unsigned char* w_buf ; I : 書き込みデータ格納先
 *-----------------------------------------------------------------------------
 * Return Value : ACK 応答有り : E_OK
 *              : ACK 応答無し : E_ERR
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
int eeprom_write(unsigned char d_code,unsigned char d_adr,unsigned short w_adr,
                                unsigned int w_size,unsigned char* w_buf)
{
  int ack = E_OK;
  unsigned char send[3];

  send[0] = (unsigned char)(d_code|((d_adr & 0x7)<<1)|IIC_DATA_WR);
  send[1] = (unsigned char)((w_adr>>8) & 0x00ff);
  send[2] = (unsigned char)(w_adr & 0x00ff);

  /* ==== 開始条件発行 ==== */
  io_iic3_mst_start();

  /* ==== スレーブデバイスアドレス送信 ==== */
  ack = io_iic3_mst_send_sync( send, 3);         /* 送信完了後にリターン */
  if( ack != E_OK ){
    io_iic3_mst_send_end();
    return ack;
  }
  /* ==== データライト ==== */
  ack = io_iic3_mst_send_sync( w_buf, w_size );  /* 送信完了後にリターン */

  /* ==== 停止条件発行 ==== */
  io_iic3_mst_send_end();

  return ack;
}
```

3.6　サンプルプログラムリスト"main.c（5）"

```
196  /*""FUNC COMMENT""*****************************************************************
197   * ID          :
198   * Outline     : EEPROMからのデータリード
199   *-------------------------------------------------------------------------------
200   * Include     :
201   *-------------------------------------------------------------------------------
202   * Declaration  : int eeprom_read(unsigned char d_code,unsigned char d_adr,
203   *              : unsigned short r_adr,unsigned int r_size,unsigned char* r_buf);
204   *-------------------------------------------------------------------------------
205   * Description  : デバイスコードd_code、デバイスアドレスd_adrで指定したEEPROMから
206   *              : r_sizeバイト分データを読み出し、r_bufで指定した領域に読み出した
207   *              : データを格納します。EEPROMのメモリアドレスはr_adrで指定します。
208   *-------------------------------------------------------------------------------
209   * Argument    : unsigned char d_code ; I : デバイスコード
210   *              : unsigned char d_adr  ; I : デバイスアドレス
211   *              : unsigned short r_adr ; I : 読み出し開始アドレス
212   *              : unsigned int r_size  ; I : 読み出しデータサイズ
213   *              : unsigned char* r_buf ; O : 読み出しデータ格納先
214   *-------------------------------------------------------------------------------
215   * Return Value : ACK応答有り : E_OK
216   *              : ACK応答無し : E_ERR
217   *-------------------------------------------------------------------------------
218   * Note        : None
219   *""FUNC COMMENT END""*************************************************************/
220  int eeprom_read(unsigned char d_code,unsigned char d_adr,unsigned short r_adr,
221                                  unsigned int r_size,unsigned char* r_buf)
222  {
223    int ack = E_OK;
224    unsigned char send[4];
225
226    send[0] = (unsigned char)(d_code|((d_adr & 0x7)<<1)|IIC_DATA_WR);
227    send[1] = (unsigned char)((r_adr>>8) & 0x00ff);
228    send[2] = (unsigned char)(r_adr & 0x00ff);
229    send[3] = (unsigned char)(d_code|((d_adr & 0x7)<<1)|IIC_DATA_RD);
230
231    /* ==== 開始条件発行 ==== */
232    io_iic3_mst_start();
233
234    /* ==== スレーブデバイスアドレス送信 ==== */
235    ack = io_iic3_mst_send_sync( send, 3);          /* 送信完了後にリターン */
236    if( ack != E_OK ){
237      io_iic3_mst_send_end();
238      return ack;
239    }
240    /* ==== 開始条件の再発行 ==== */
241    io_iic3_mst_restart();
242
243    /* ==== コマンド送信（データリード指定） ==== */
244    ack = io_iic3_mst_send_sync( &send[3], 1);      /* 送信完了後にリターン */
245    if( ack != E_OK ){
246      io_iic3_mst_send_end();
247      return ack;
248    }
```

3.7　サンプルプログラムリスト"main.c（6）"

```
249     /* ==== データリード==== */
250     ack = io_iic3_mst_recv_sync( r_buf, r_size);       /* 受信完了後にリターン */
251
252     /* ==== 停止条件発行(最終データの読み出しも行います) ==== */
253     r_buf[r_size - 1] = io_iic3_mst_recv_end();
254
255     return ack;
256   }
257   /*""FUNC COMMENT""*****************************************************************
258    * ID          :
259    * Outline     : Acknowledge Polling
260    *-------------------------------------------------------------------------------
261    * Include     : iodefine.h
262    *-------------------------------------------------------------------------------
263    * Declaration  : int eeprom_ack_polling(unsigned char d_code,unsigned char d_adr);
264    *-------------------------------------------------------------------------------
265    * Description  : 本関数は、EEPROM が書き換え中か否かを判定します。
266    *              : EEPROM は、内部で書き換え中の場合入力コマンドを無視し、
267    *              : ACK を返しません。本関数で EEPROM が書き換え中ではないこと
268    *              : を確認してから EEPROM にアクセスしてください。
269    *              : Acknowledge Polling 時に送信する Read/Write コードは、
270    *              : 使用するデバイスにより異なります。使用する EEPROM の
271    *              : データシートで確認してください。
272    *-------------------------------------------------------------------------------
273    * Argument    : unsigned char d_code ; I : デバイスコード
274    *             : unsigned char d_adr  ; I : デバイスアドレス
275    *-------------------------------------------------------------------------------
276    * Return Value : E_OK  : NOT_BUSY
277    *              : E_ERR : BUSY (EEPROM は書き換え中)
278    *-------------------------------------------------------------------------------
279    * Note        : None
280    *""FUNC COMMENT END""*************************************************************/
281   int eeprom_ack_polling(unsigned char d_code,unsigned char d_adr)
282   {
283     int ack = E_OK;
284     unsigned char send[1];
285
286     send[0] = (unsigned char)(d_code|((d_adr & 0x7)<<1)|IIC_DATA_WR);
287
288     /* ==== アクノリッジポーリングを実施 ==== */
289     io_iic3_mst_start();                        /* 開始条件発行 */
290     ack = io_iic3_mst_send_sync( send, 1);      /* 送信完了後にリターン */
291     io_iic3_mst_send_end();                     /* 停止条件発行 */
292
293     return ack;
294   }
295   /* End of File */
```

3.8　サンプルプログラムリスト"iic3.c（1）"

```
1     /***********************************************************************
2     *   DISCLAIMER
3     *
4     *   This software is supplied by Renesas Electronics Corporation and is only
5     *   intended for use with Renesas products.  No other uses are authorized.
6     *
7     *   This software is owned by Renesas Electronics Corporation and is protected under
8     *   all applicable laws, including copyright laws.
9     *
10    *   THIS SOFTWARE IS PROVIDED "AS IS" AND RENESAS MAKES NO WARRANTIES
11    *   REGARDING THIS SOFTWARE, WHETHER EXPRESS, IMPLIED OR STATUTORY,
12    *   INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
13    *   PARTICULAR PURPOSE AND NON-INFRINGEMENT.  ALL SUCH WARRANTIES ARE EXPRESSLY
14    *   DISCLAIMED.
15    *
16    *   TO THE MAXIMUM EXTENT PERMITTED NOT PROHIBITED BY LAW, NEITHER RENESAS
17    *   ELECTRONICS CORPORATION NOR ANY OF ITS AFFILIATED COMPANIES SHALL BE LIABLE
18    *   FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES
19    *   FOR ANY REASON RELATED TO THIS SOFTWARE, EVEN IF RENESAS OR ITS
20    *   AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
21    *
22    *   Renesas reserves the right, without notice, to make changes to this
23    *   software and to discontinue the availability of this software.
24    *   By using this software, you agree to the additional terms and
25    *   conditions found by accessing the following link:
26    *   http://www.renesas.com/disclaimer
27    ***********************************************************************
28    *   Copyright (C) 2010 Renesas Electronics Corporation. All rights reserved.
29    *""FILE COMMENT""*********** Technical reference data *************************
30    *   System Name : SH7264 Sample Program
31    *   File Name   : iic3.c
32    *   Abstract    : IIC3 割り込みを用いた EEPROM リードライト例
33    *   Version     : 1.00.00
34    *   Device      : SH7262/SH7264
35    *   Tool-Chain  : High-performance Embedded Workshop (Ver.4.07.00).
36    *               : C/C++ compiler package for the SuperH RISC engine family
37    *               :                          (Ver.9.03 Release00).
38    *   OS          : None
39    *   H/W Platform: M3A-HS64G50(CPU board)
40    *   Description :
41    ***********************************************************************
42    *   History     : Dec.07,2010 Ver.1.00.00
43    *""FILE COMMENT END""*****************************************************/
44    #include <machine.h>
45    #include "iodefine.h"          /* SH7264 iodefine  */
46
```

3.9　サンプルプログラムリスト"iic3.c（2）"

```
47    /* ==== symbol definition ==== */
48    typedef enum{
49      IIC3_IDOL=0,                /* アイドル状態(送受信完了状態) */
50      IIC3_NACK,                  /* NACK 受信による終了 */
51      IIC3_SEND,                  /* 送信中(送信系の割り込み使用中) */
52      IIC3_RECV,                  /* 受信中(受信系の割り込み使用中) */
53    }IIC3_MODE;
54
55    typedef struct{
56      volatile IIC3_MODE mode;    /* IIC3 の制御状態 */
57      unsigned char *buffer;      /* 連続送受信時のアクセス先を管理するポインタ */
58      int count;                  /* 連続送受信時の残りデータサイズ */
59    }IIC3_INFO;
60
61    #define E_OK 0
62    #define E_ERR -1
63
64    /* ==== RAM allocation variable declaration ==== */
65    static IIC3_INFO iic3_info;
66
67    /* ==== prototype declaration ==== */
68    int  io_iic3_init(void);
69    void io_iic3_mst_start(void);
70    void io_iic3_mst_restart(void);
71    void io_iic3_mst_send_end(void);
72    unsigned char io_iic3_mst_recv_end(void);
73    int  io_iic3_mst_send_sync( unsigned char *buffer, int size);
74    int  io_iic3_mst_send( unsigned char *buffer, int size);
75    int  io_iic3_mst_recv_sync( unsigned char *buffer, int size);
76    int  io_iic3_mst_recv( unsigned char *buffer, int size);
77    void io_iic3_tei_interrupt(void);
78    void io_iic3_rxi_interrupt(void);
79
```

3.10　サンプルプログラムリスト"iic3.c（3）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : IIC3 モジュール初期設定
 *-----------------------------------------------------------------------------
 * Include     : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : int io_iic3_init(void);
 *-----------------------------------------------------------------------------
 * Description  : IIC3 のチャネル 1 の初期設定を行います。
 *-----------------------------------------------------------------------------
 * Argument    : void
 *-----------------------------------------------------------------------------
 * Return Value : E_OK
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
int io_iic3_init(void)
{
  /* ---- STBCR5 ---- */
  CPG.STBCR5.BIT.MSTP56   = 0;    /* IIC3 チャネル 1 動作   */

  /* ---- PORT ---- */
  PORT.PECR0.BIT.PE2MD = 0x01;/* SCL1 select */
  PORT.PECR0.BIT.PE3MD = 0x01;/* SDA1 select */


  /* ----IIC31 module operation enabled ---- */
  IIC3_1.ICCR1.BYTE = 0xB5;   /* IIC3 モジュール動作可能           */
                              /* 次の受信動作を継続               */
                              /* マスタ選択                       */
                              /* 送信選択                         */
                              /* 転送クロックレート：Pφ/92(391kHz) */
  /*  ---IIC bus mode register (ICMR) setting --- */
  IIC3_1.ICMR.BYTE = 0x30;
           /*
                bit7    : MLS:0 ---------------------   MSB first
                bit6-4   : Reserve:1 -----------------  Reserve bit
                bit3    : BCWP:0---------------------   Unsetting
                bit2-0   : BC0:0, BC1:0,BC0:0---------  IIC format 9-bit
           */
  /* ---- 割り込みの禁止許可設定 ---- */
  IIC3_1.ICIER.BYTE = 0x00;       /* IIC3 の全ての割り込みを禁止 */
  INTC.IPR16.BIT._IIC31 = 5;      /* 割り込み優先レベル 5 */

  /* ---- 管理情報の初期化 ---- */
  iic3_info.mode   = IIC3_IDOL;
  iic3_info.buffer = (void *)0;
  iic3_info.count  = 0;

  return(E_OK);
}
```

3.11　サンプルプログラムリスト"iic3.c（4）"

```
131   /*""FUNC COMMENT""*****************************************************************
132    * ID          :
133    * Outline     : 開始条件発行
134    *-------------------------------------------------------------------------------
135    * Include     : iodefine.h
136    *-------------------------------------------------------------------------------
137    * Declaration  : void io_iic3_mst_start(void);
138    *-------------------------------------------------------------------------------
139    * Description  : 開始条件を発行します。
140    *              : 開始条件を発行する前にバスが解放されていることを確認し、マスタ
141    *              : 送信モードに設定しています。
142    *-------------------------------------------------------------------------------
143    * Argument    : void
144    *-------------------------------------------------------------------------------
145    * Return Value : void
146    *-------------------------------------------------------------------------------
147    * Note        : None
148    *""FUNC COMMENT END""*************************************************************/
149   void io_iic3_mst_start(void)
150   {
151     while(IIC3_1.ICCR2.BIT.BBSY == 1){
152       /* バス解放待ち */
153     }
154     IIC3_1.ICCR1.BYTE |= 0x30;                    /* マスタ送信モードに設定 */
155     IIC3_1.ICCR2.BYTE = ((IIC3_1.ICCR2.BYTE & 0xbf)|0x80); /* 開始条件発行 */
156   }
157   /*""FUNC COMMENT""*****************************************************************
158    * ID          :
159    * Outline     : 開始条件の再発行
160    *-------------------------------------------------------------------------------
161    * Include     : iodefine.h
162    *-------------------------------------------------------------------------------
163    * Declaration  : void io_iic3_mst_restart(void);
164    *-------------------------------------------------------------------------------
165    * Description  : 開始条件を再発行します。開始条件の発行以外の処理は行いません。
166    *-------------------------------------------------------------------------------
167    * Argument    : void
168    *-------------------------------------------------------------------------------
169    * Return Value : void
170    *-------------------------------------------------------------------------------
171    * Note        : None
172    *""FUNC COMMENT END""*************************************************************/
173   void io_iic3_mst_restart(void)
174   {
175     IIC3_1.ICCR2.BYTE = ((IIC3_1.ICCR2.BYTE & 0xbf)|0x80); /* 開始条件発行 */
176   }
```

3.12　サンプルプログラムリスト"iic3.c（5）"

```
177   /*""FUNC COMMENT""*****************************************************************
178    * ID          :
179    * Outline     : 停止条件発行
180    *-------------------------------------------------------------------------------
181    * Include     : iodefine.h
182    *-------------------------------------------------------------------------------
183    * Declaration  : void io_iic3_mst_send_end(void);
184    *-------------------------------------------------------------------------------
185    * Description  : 停止条件を発行し、スレーブ受信モードに切り替えます。
186    *              : 本関数はマスタ送信後に停止条件を発行する場合に使用します。
187    *-------------------------------------------------------------------------------
188    * Argument    : void
189    *-------------------------------------------------------------------------------
190    * Return Value : void
191    *-------------------------------------------------------------------------------
192    * Note        : None
193    *""FUNC COMMENT END""*************************************************************/
194   void io_iic3_mst_send_end(void)
195   {
196     /* ==== 停止条件発行 ==== */
197     IIC3_1.ICSR.BIT.TEND = 0;       /* TEND フラグクリア */
198     IIC3_1.ICSR.BIT.STOP = 0;       /* STOP フラグクリア */
199     IIC3_1.ICCR2.BYTE &= 0x3f;      /* 停止条件発行 */
200
201     /* ==== バス解放待ち ==== */
202     while(IIC3_1.ICSR.BIT.STOP == 0){
203       /* wait */
204     }
205     /* ==== スレーブ受信モードに設定 ==== */
206     IIC3_1.ICCR1.BYTE &= 0xcf;      /* スレーブ受信モード */
207     IIC3_1.ICSR.BIT.TDRE = 0;       /* TDRE をクリア */
208   }
```

3.13　サンプルプログラムリスト"iic3.c（6）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : 停止条件発行
 *-----------------------------------------------------------------------------
 * Include     : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : unsigned char io_iic3_mst_recv_end(void);
 *-----------------------------------------------------------------------------
 * Description  : 停止条件を発行し、スレーブ受信モードに切り替えます。
 *              : 本関数はマスタ受信後に停止条件を発行する場合に使用します。
 *-----------------------------------------------------------------------------
 * Argument    : void
 *-----------------------------------------------------------------------------
 * Return Value : 最後の受信データ
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
unsigned char io_iic3_mst_recv_end(void)
{
  unsigned char data;

  /* ==== SCL の 9 クロック目の立ち下がりを待つ ==== */
  while(IIC3_1.ICCR2.BIT.SCLO == 1){  /* Technical Update [TN-MC*-A020A/J] */
    /* wait */
  }
  /* ==== 停止条件発行 ==== */
  IIC3_1.ICSR.BIT.STOP = 0;           /* STOP フラグクリア */
  IIC3_1.ICCR2.BYTE &= 0x3f;          /* 停止条件発行 */

  /* ==== バス解放待ち ==== */
  while(IIC3_1.ICSR.BIT.STOP == 0){
    /* wait */
  }
  /* ==== 最終バイトのデータをリード==== */
  data = IIC3_1.ICDRR;                /* レジスタから最終データをリード */

  /* ==== スレーブ受信モードに戻す ==== */
  IIC3_1.ICCR1.BIT.RCVD = 0;          /* RCVD をクリア */
  IIC3_1.ICCR1.BYTE &= 0xcf;          /* スレーブ受信モード */

  return data;
}
```

3.14　サンプルプログラムリスト"iic3.c（7）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : マスタ送信（同期型）
 *-----------------------------------------------------------------------------
 * Include     : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : int io_iic3_mst_send_sync( unsigned char *buffer, int size);
 *-----------------------------------------------------------------------------
 * Description  : マスタ送信モードで 引数 buffer で指定したアドレスから
 *              : 引数 size で指定したバイト数だけデータを送信します。
 *              : データの送信には、送信終了割り込み（TEI）を使用します。
 *              : 送信完了後に呼び出し元に戻ります。
 *-----------------------------------------------------------------------------
 * Argument    : unsigned char *buffer ; I : 送信データを格納したバッファ
 *              : int          size   ; I : 送信データサイズ
 *-----------------------------------------------------------------------------
 * Return Value : ACK 応答有り : E_OK
 *              : ACK 応答無し : E_ERR
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
int io_iic3_mst_send_sync( unsigned char *buffer, int size)
{
  int ack = E_OK;

  /* ==== マスタ送信開始 ==== */
  ack = io_iic3_mst_send( buffer, size);
  if( ack == E_OK ){
    /* ==== 送信完了待ち ==== */
    while( iic3_info.mode == IIC3_SEND ){
        /* wait */
    }
    /* ==== NACK を受信した場合はエラー終了 ==== */
    if( iic3_info.mode == IIC3_NACK ){
        iic3_info.mode = IIC3_IDOL;
        ack = E_ERR;
    }
  }
  return ack;
}
```

3.15　サンプルプログラムリスト"iic3.c（8）"

```
291  /*""FUNC COMMENT""*****************************************************************
292   * ID          :
293   * Outline     : マスタ送信（非同期型）
294   *-------------------------------------------------------------------------------
295   * Include     : iodefine.h
296   *-------------------------------------------------------------------------------
297   * Declaration  : int io_iic3_mst_send( unsigned char *buffer, int size);
298   *-------------------------------------------------------------------------------
299   * Description  : マスタ送信モードで 引数 buffer で指定したアドレスから
300   *              : 引数 size で指定したバイト数だけデータを送信します。
301   *              : データの送信には、送信終了割り込み（TEI）を使用します。
302   *              : 送信完了を待たずに呼び出し元に戻ります。
303   *-------------------------------------------------------------------------------
304   * Argument    : unsigned char *buffer ; I : 送信データを格納したバッファ
305   *              : int          size   ; I : 送信データサイズ
306   *-------------------------------------------------------------------------------
307   * Return Value : ACK 応答有り : E_OK
308   *              : ACK 応答無し : E_ERR
309   *-------------------------------------------------------------------------------
310   * Note        : None
311   *""FUNC COMMENT END""*************************************************************/
312  int io_iic3_mst_send( unsigned char *buffer, int size)
313  {
314    /* ==== 転送中でないことを確認 ==== */
315    if( iic3_info.mode != IIC3_IDOL ){
316      return E_ERR;
317    }
318    /* ==== IIC3 制御情報の構造体を設定 ==== */
319    iic3_info.mode   = IIC3_SEND;
320    iic3_info.buffer = buffer;
321    iic3_info.count  = size;
322
323    /* ==== 送信開始 ==== */
324    IIC3_1.ICDRT = *(iic3_info.buffer)++;
325    iic3_info.count--;
326
327    /* ==== 送信終了割り込みを許可 ==== */
328    IIC3_1.ICIER.BIT.TEIE = 1;
329
330    return E_OK;
331  }
```

3.16 サンプルプログラムリスト"iic3.c（9）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : マスタ受信（同期型）
 *-----------------------------------------------------------------------------
 * Include     : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : int io_iic3_mst_recv_sync( unsigned char *buffer, int size);
 *-----------------------------------------------------------------------------
 * Description  : マスタ受信モードで 引数 buffer で指定したアドレスに
 *              : 引数 size で指定したバイト数だけデータを受信します。
 *              : データの受信には、受信データフル割り込み（RXI）を使用します。
 *              : 受信完了後に呼び出し元に戻ります。
 *-----------------------------------------------------------------------------
 * Argument    : unsigned char *buffer ; O : 受信データを格納するバッファ
 *              : int          size   ; I : 受信データサイズ
 *-----------------------------------------------------------------------------
 * Return Value : 正常終了 : E_OK
 *              : 異常終了 : E_ERR
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
int io_iic3_mst_recv_sync( unsigned char *buffer, int size)
{
  int ack = E_OK;

  /* ==== マスタ受信開始 ==== */
  ack = io_iic3_mst_recv( buffer, size);
  if( ack == E_OK ){
    /* ==== 受信完了待ち ==== */
    while( iic3_info.mode == IIC3_RECV ){
        /* wait */
    }
  }
  return ack;
}
```

3.17　サンプルプログラムリスト"iic3.c（10）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : マスタ受信（非同期型）
 *-----------------------------------------------------------------------------
 * Include     : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : int io_iic3_mst_recv( unsigned char *buffer, int size);
 *-----------------------------------------------------------------------------
 * Description  : マスタ受信モードで 引数 buffer で指定したアドレスに
 *              : 引数 size で指定したバイト数だけデータを受信します。
 *              : データの受信には、受信データフル割り込み（RXI）を使用します。
 *              : 受信完了を待たずに呼び出し元に戻ります。
 *-----------------------------------------------------------------------------
 * Argument    : unsigned char *buffer ; O : 受信データを格納するバッファ
 *              : int          size    ; I : 受信データサイズ
 *-----------------------------------------------------------------------------
 * Return Value : 正常終了 : E_OK
 *              : 異常終了 : E_ERR
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
int io_iic3_mst_recv( unsigned char *buffer, int size)
{
  int mask;
  unsigned char dummy;

  /* ==== 転送中でないことを確認 ==== */
  if( iic3_info.mode != IIC3_IDOL ){
    return E_ERR;
  }
  /* ==== IIC3 制御情報の構造体を設定 ==== */
  iic3_info.mode   = IIC3_RECV;
  iic3_info.buffer = buffer;
  iic3_info.count  = size;

  mask = get_imask();
  set_imask(15);                    /* ↓↓↓↓ 割り込み禁止 ↓↓↓↓ */

  /* ==== マスタ受信モード(非連続受信)を設定 ==== */
  IIC3_1.ICSR.BIT.TEND = 0;         /* TEND クリア */
  IIC3_1.ICCR1.BIT.MST = 1;         /* マスタモード */
  IIC3_1.ICCR1.BIT.TRS = 0;         /* 受信モード */
  IIC3_1.ICSR.BIT.TDRE = 0;         /* TDRE クリア */
  IIC3_1.ICCR1.BIT.RCVD = 1;        /* 連続受信禁止 */

  /* ==== アクノリッジを設定 ==== */
  if(iic3_info.count == 1){         /* 1 バイト受信時 */
    IIC3_1.ICIER.BIT.ACKBT = 1;     /* アクノリッジ設定"H" */
  }
  else{
    IIC3_1.ICIER.BIT.ACKBT = 0;     /* アクノリッジ設定"L" */
  }
```

3.18 サンプルプログラムリスト"iic3.c（11）"

```
419     /* ==== 受信開始 ==== */
420     dummy = IIC3_1.ICDRR;               /* ダミーリード */
421     set_imask(mask);                    /* ↑↑↑↑ 割り込み許可 ↑↑↑↑ */
422
423     /* ==== 受信データフル割り込みを許可 ==== */
424     IIC3_1.ICIER.BIT.RIE = 1;
425
426     return E_OK;
427   }
428   /*""FUNC COMMENT""*****************************************************************
429    * ID          :
430    * Outline     : 送信終了割り込み（TEI）
431    *-------------------------------------------------------------------------------
432    * Include     : iodefine.h
433    *-------------------------------------------------------------------------------
434    * Declaration  : void io_iic3_tei_interrupt(void);
435    *-------------------------------------------------------------------------------
436    * Description  : 送信終了割り込み（TEI）発生時に実行します。
437    *              : ACK 応答を受信できなかった場合は、動作状態を IIC3_NACK に設定して
438    *              : 送信処理を終了します。正常に全てのデータを送信終了した場合は
439    *              : 動作状態に IIC3_IDOL を設定して送信処理を終了します。
440    *              : 送信データが残っている場合は送信処理を継続します。
441    *-------------------------------------------------------------------------------
442    * Argument    : void
443    *-------------------------------------------------------------------------------
444    * Return Value : void
445    *-------------------------------------------------------------------------------
446    * Note        : None
447    *""FUNC COMMENT END""*************************************************************/
448   void io_iic3_tei_interrupt(void)
449   {
450     unsigned char dummy;
451
452     /* ==== NACK 受信の場合は送信終了 ==== */
453     if(IIC3_1.ICIER.BIT.ACKBR == 1){
454       iic3_info.mode = IIC3_NACK;
455       IIC3_1.ICIER.BIT.TEIE = 0;            /* 送信終了割り込みを禁止 */
456     }
457     /* ==== 残りデータがあれば送信を継続 ==== */
458     else if( iic3_info.count > 0 ){
459       IIC3_1.ICDRT = *(iic3_info.buffer)++;
460       iic3_info.count--;
461     }
462     /* ==== 全データ送信したら正常終了 ==== */
463     else{
464       IIC3_1.ICIER.BIT.TEIE = 0;            /* 送信終了割り込みを禁止 */
465       iic3_info.mode = IIC3_IDOL;
466     }
467     /* ==== 割り込み要因のクリア時間を確保 ==== */
468     dummy = IIC3_1.ICSR.BYTE;
469   }
```

3.19　サンプルプログラムリスト"iic3.c（11）"

```
/*""FUNC COMMENT""*****************************************************************
 * ID          :
 * Outline     : 受信データフル割り込み（RXI）
 *-----------------------------------------------------------------------------
 * Include     : iodefine.h
 *-----------------------------------------------------------------------------
 * Declaration  : void io_iic3_rxi_interrupt(void);
 *-----------------------------------------------------------------------------
 * Description  : 受信データフル割り込み（RXI）発生時に実行します。
 *              : 2 バイト以上受信データが残っている場合は、受信処理を継続します。
 *              : 残り 1 バイトの場合は、NACK 応答を設定してから受信処理を継続します。
 *              : 全てのデータを受信した場合は、受信処理を終了します。
 *              : ただし、本関数は最後のデータをリードしません。停止条件発行後に
 *              : リードしてください。
*-----------------------------------------------------------------------------
 * Argument    : void
 *-----------------------------------------------------------------------------
 * Return Value : void
 *-----------------------------------------------------------------------------
 * Note        : None
 *""FUNC COMMENT END""*************************************************************/
void io_iic3_rxi_interrupt(void)
{
  unsigned char dummy;

  /* ==== 受信をカウント ==== */
  iic3_info.count--;

  /* ==== 2 バイト以上の残りデータあり ==== */
  if( iic3_info.count >= 2 ){
    *(iic3_info.buffer)++ = IIC3_1.ICDRR; /* レジスタから受信データをリード */
  }
  /* ==== 残り 1 バイトのみ ==== */
  else if( iic3_info.count == 1 ){
    IIC3_1.ICIER.BIT.ACKBT = 1;           /* アクノリッジ設定"H" */
    *(iic3_info.buffer)++ = IIC3_1.ICDRR; /* レジスタから受信データをリード */
  }
  /* ==== 全データ受信したら正常終了 ==== */
  else{
    IIC3_1.ICIER.BIT.RIE = 0;             /* 受信データフル割り込みを禁止 */
    iic3_info.mode = IIC3_IDOL;
  }
  /* ==== 割り込み要因のクリア時間を確保 ==== */
  dummy = IIC3_1.ICSR.BYTE;
}
/* End of File */
```

4. 参考ドキュメント

- ソフトウェアマニュアル
  SH-2A、SH2A-FPU　ソフトウェアマニュアル Rev.3.00
  （最新版をルネサスエレクトロニクスのホームページから入手してください。）

- ハードウェアマニュアル
  SH7262 グループ、SH7264 グループ　ハードウェアマニュアル Rev.2.00
  （最新版をルネサスエレクトロニクスの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/inquiry

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2010.12.07	—	初版発行
1.01	2012.02.10	17	マスタ受信モード時の注意の誤記修正 正）8 クロックの立ち上がりまでに行ってください 誤）8 クロックの立ち下がりまでに行ってください

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、本文を参照してください。なお、本マニュアルの本文と異なる記載がある場合は、本文の記載が優先するものとします。

1. 未使用端子の処理

   【注意】未使用端子は、本文の「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。未使用端子は、本文「未使用端子の処理」で説明する指示に従い処理してください。

2. 電源投入時の処置

   【注意】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. リザーブアドレスのアクセス禁止

   【注意】リザーブアドレスのアクセスを禁止します。

   アドレス領域には、将来の機能拡張用に割り付けられているリザーブアドレスがあります。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。

4. クロックについて

   【注意】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。

5. 製品間の相違について

   【注意】型名の異なる製品に変更する場合は、事前に問題ないことをご確認下さい。

   同じグループのマイコンでも型名が違うと、内部メモリ、レイアウトパターンの相違などにより、特性が異なる場合があります。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。

ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。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。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、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に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したものですが、誤りがないことを保証するものではありません。万一、本資料に記載されている情報の誤りに起因する損害がお客様に生じた場合においても、当社は、一切その責任を負いません。
7. 当社は、当社製品の品質水準を「標準水準」、「高品質水準」および「特定水準」に分類しております。また、各品質水準は、以下に示す用途に製品が使われることを意図しておりますので、当社製品の品質水準をご確認ください。お客様は、当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途に当社製品を使用することができません。また、お客様は、当社の文書による事前の承諾を得ることなく、意図されていない用途に当社製品を使用することができません。当社の文書による事前の承諾を得ることなく、「特定水準」に分類された用途または意図されていない用途に当社製品を使用したことによりお客様または第三者に生じた損害等に関し、当社は、一切その責任を負いません。なお、当社製品のデータ・シート、データ・ブック等の資料で特に品質水準の表示がない場合は、標準水準製品であることを表します。

   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

   高品質水準： 輸送機器（自動車、電車、船舶等）、交通用信号機器、防災・防犯装置、各種安全装置、生命維持を目的として設計されていない医療機器（厚生労働省定義の管理医療機器に相当）

   特定水準： 航空機器、航空宇宙機器、海底中継機器、原子力制御システム、生命維持のための医療機器（生命維持装置、人体に埋め込み使用するもの、治療行為（患部切り出し等）を行うもの、その他直接人命に影響を与えるもの）（厚生労働省定義の高度管理医療機器に相当）またはシステム等
8. 本資料に記載された当社製品のご使用につき、特に、最大定格、動作電源電圧範囲、放熱特性、実装条件その他諸条件につきましては、当社保証範囲内でご使用ください。当社保証範囲を超えて当社製品をご使用された場合の故障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おりますが、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、当社製品は耐放射線設計については行っておりません。当社製品の故障または誤動作が生じた場合も、人身事故、火災事故、社会的損害など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、機器またはシステムとしての出荷保証をお願いいたします。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、お客様が製造された最終の機器・システムとしての安全検証をお願いいたします。
10. 当社製品の環境適合性等、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。お客様が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. 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固くお断りいたします。
12. 本資料に関する詳細についてのお問い合わせその他お気付きの点等がございましたら当社営業窓口までご照会ください。

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その総株主の議決権の過半数を直接または間接に保有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。

ルネサス エレクトロニクス株式会社

■営業お問合せ窓口 http://www.renesas.com

※営業お問合せ窓口の住所・電話番号は変更になることがあります。最新情報につきましては、弊社ホームページをご覧ください。

ルネサス エレクトロニクス販売株式会社　〒100-0004　千代田区大手町2-6-2（日本ビル） (03)5201-5307

■技術的なお問合せおよび資料のご請求は下記へどうぞ。
総合お問合せ窓口：http://japan.renesas.com/inquiry

Colophon 1.0