お客様各位

カタログ等資料中の旧社名の扱いについて

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ（http://www.renesas.com）

2010年4月1日
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社（http://www.renesas.com）

【問い合わせ先】http://japan.renesas.com/inquiry

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。

H8/3664

シリアル EEPROM（I²C EEPROM）のアクセス

要旨

H8/3664 グループは、高速 H8/300H CPU を核に、システム構成に必要な周辺機能を集積したシングルチップマイクロコンピュータです。H8/300H CPU は、H8/300 CPU と互換性のある命令体系を備えています。

H8/3664 グループは、システム構成に必要な周辺機能として、4 種類のタイマ、I²C バスインタフェース、シリアルコミュニケーションインタフェース、10 ビット A/D 変換器を内蔵しており、高度な制御システムの組み込み用マイコンとして活用できます。

H8/300H シリーズ-H8/3664-アプリケーションノートは､H8/3664 グループの内蔵周辺機能を単独で使用した場合の動作例を示した"基礎編"により構成されており、ユーザにてソフトウェア設計およびハードウェア設計の際、ご参考として役立てていただけるようにまとめたものです。

なお、本アプリケーションノートに掲載されているプログラム、回路等の動作は確認しておりますが、実際にご使用になる場合は、必ず動作確認の上ご使用くださいますようにお願い致します。

動作確認デバイス

H8/3664

目次

1. 概要

H8/3664 の I²C インタフェースにより、I²C EEPROM の読み書きを行います。

2. 構成

図 2.1 に、I²C EEPROM との接続図を示します。

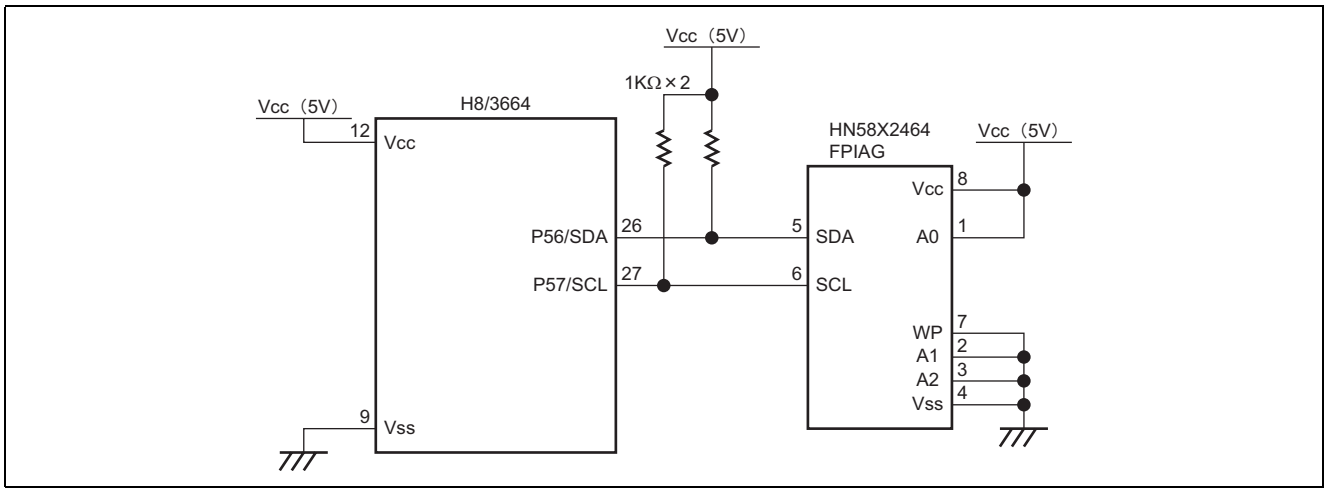


図 2.1　I²C EEPROM との接続図

仕様

- H8/3664 の動作周波数：16MHz
- I²C EEPROM（HN58X2464FPIAG）のピン仕様を表 2.1 に示します。
- I²C EEPROM 仕様：64Kbit（8192×8bit）

表 2.1　I²C EEPROM ピン仕様

ピン名称	機　　能
A0～A2	Device address（A0 は High、A1 および A2 は Low に固定）
SCL	Serial clock input
SDA	Serial data input/output
WP	Write protect
Vcc	Power supply
Vss	Ground

3. サンプルプログラム

3.1 機能

1. I²C EEPROM に 1 バイトのデータを書き込みます。（Byte Write）
2. I²C EEPROM から 1 バイトのデータを読み出します。（Random Read）
3. I²C EEPROM に連続してデータを書き込みます。（Page Write）
4. I²C EEPROM から連続してデータを読み出します。（Sequential Read）

3.2 組み込み方法

1. サンプルプログラム 1-A　define 定義を組み込んでください。
2. サンプルプログラム 1-B　プロトタイプ宣言を組み込んでください。
3. サンプルプログラム 1-C　ソースプログラムを組み込んでください。

3.3 サンプルプログラムの変更

サンプルプログラムそのままでは、システムが動作しないことがあります。お客様のプログラムやシステム環境に合わせて修正を行う必要があります。

1. レジスタの構造体定義は、ルネサス Web
   http://www.renesasu.com/jpn/products/mpumcu/tool/crosstool/iodef/index.html　で無償入手できる定義ファイルをご利用になるとサンプルプログラムをそのまま使用することができます。独自に作成される場合は、サンプルプログラム中に使用している IO レジスタの構造体を適宜変更して下さい。
2. サンプルプログラム中、I²C インタフェースの状態監視のためにタイマ W を 10ms ごとに起動し 5 秒でタイムアウトするよう設計してあります。タイマ処理は、お客様の都合に合わせて変更して構いません。もちろんそのまま使用することも可能です。サンプルプログラムのタイマ処理をそのまま使用する場合は、次の変更を行ってください。
   A. サンプルプログラム 1-E
   - タイマ W のリセットベクタを追加してください。
   - 共通変数として、com_timer を追加して下さい。
     （タイマ W が 10ms に割り込むようご使用になるマイコンの動作周波数に合わせて GRA の設定値を変更してください。設定値は、H8/3644 のハードウェアマニュアルを、変更箇所はサンプルプログラム中の program note を参照して下さい。）
   - タイマ W の割り込み処理を追加して下さい。
3. ターゲットデバイスのスペックとマイコンの動作周波数に合せて I²C インタフェースの転送レート ICMR（CKS2 : 0）および TSCR（IICX）を設定して下さい。設定値は、H8/3644 のハードウェアマニュアルを、変更箇所はサンプルプログラム中の program note を参照して下さい。本サンプルプログラムでは、転送レートは 200kbps に設定してあります。

3.4 使用方法

1. I²C EEPROM に 1 バイトのデータを書き込みます。

```
unsigned int com_i2c_eeprom_write
    ( unsigned char slave_addr , unsigned int rom_addr , unsigned char rom_data )
```

引数	説　　明
slave_addr	I²C EEPROM のスレーブアドレスを指定します。スレーブアドレスは、デバイスコード（固定値 1010）と I²C EEPROM のアドレス端子（A2：0）で決まる値です。 本サンプルプログラムでは、A2：1＝Low、A0＝High slave_addr フォーマット デバイスコード | アドレス端子 | 1 | 0 | 1 | 0 | A2 | A1 | A0 | 0 であるから、slave_addr への設定値は 0xA2 です。
rom_addr	書き込みを行う ROM アドレスを指定します。
rom_data	書き込む 1 バイトのデータを指定します。

戻り値	説　　明
0	正常終了
1	異常終了（バスビジータイムアウト）
2	異常終了（転送準備完了待ちタイムアウト）
3	異常終了（アクノリッジタイムアウト）
4	異常終了（転送完了待ちタイムアウト）
5	異常終了（受信完了待ちタイムアウト）
6	異常終了（停止条件検出タイムアウト）

使用例

```
int ret ;
unsigned char slave_addr , rom_data ;
unsigned int rom_addr ;

ret = com_i2c_eeprom_write (slave_addr , rom_addr , rom_data )
```

2. I²C EEPROM に 1 バイトのデータを読み出します。

```
unsigned int com_i2c_eeprom_read
    ( unsigned char slave_addr , unsigned int rom_addr , unsigned char *rom_data )
```

引数	説　　明
slave_addr	I²C EEPROM のスレーブアドレスを指定します。スレーブアドレスは、デバイスコード（固定値 1010）と I²C EEPROM のアドレス端子（A2：0）で決まる値です。 本サンプルプログラムでは、A2：1＝Low、A0＝High slave_addr フォーマット デバイスコード ／ アドレス端子 ／ 1 0 1 0 ／ A2 A1 A0 ／ 0 であるから、slave_addr への設定値は 0xA2 です。
rom_addr	読み出しを行う ROM アドレスを指定します。
*rom_data	読み出したデータを格納するアドレスを指定します。

戻り値	説　　明
0	正常終了
1	異常終了（バスビジターイムアウト）
2	異常終了（転送準備完了待ちタイムアウト）
3	異常終了（アクノリッジタイムアウト）
4	異常終了（転送完了待ちタイムアウト）
5	異常終了（受信完了待ちタイムアウト）
6	異常終了（停止条件検出タイムアウト）

使用例

```
int ret ;
unsigned char slave_addr , *rom_data ;
unsigned int rom_addr ;

ret = com_i2c_eeprom_read (slave_addr , rom_addr , *rom_data )
```

3. I²C EEPROM に連続してデータを書き込みます。

```
unsigned int com_i2c_eeprom_page_write
    ( unsigned char slave_addr , unsigned int rom_addr , unsigned int rom_length ,
      unsigned char *rom_data )
```

引数	説　　明
slave_addr	I²C EEPROM のスレーブアドレスを指定します。スレーブアドレスは、デバイスコード（固定値 1010）と I²C EEPROM のアドレス端子（A2：0）で決まる値です。 本サンプルプログラムでは、A2：1＝Low、A0＝High slave_addr フォーマット デバイスコード ｜ アドレス端子 ｜ 1 ｜ 0 ｜ 1 ｜ 0 ｜ A2 ｜ A1 ｜ A0 ｜ 0 であるから、slave_addr への設定値は 0xA2 です。
rom_addr	書き込みを行う先頭の ROM アドレスをしてします。
rom_length	書き込みを行うバイト数を指定します。本デバイスの場合、Max32 バイトです。
*rom_data	書き込みデータを格納する先頭アドレスを指定します。

戻り値	説　　明
0	正常終了
1	異常終了（バスビジータイムアウト）
2	異常終了（転送準備完了待ちタイムアウト）
3	異常終了（アクノリッジタイムアウト）
4	異常終了（転送完了待ちタイムアウト）
5	異常終了（受信完了待ちタイムアウト）
6	異常終了（停止条件検出タイムアウト）

使用例

```
int ret ;
unsigned char slave_addr , *rom_data ;
unsigned int rom_length , rom_addr ;

ret = com_i2c_eeprom_page_write (slave_addr , rom_addr , rom_length , *rom_data )
```

4. I²C EEPROM に連続してデータを読み出します。

```
unsigned int com_i2c_eeprom_page_read
    ( unsigned char slave_addr , unsigned int rom_addr , unsigned int rom_length ,
      unsigned char *rom_data )
```

引数	説　　明
slave_addr	I²C EEPROM のスレーブアドレスを指定します。スレーブアドレスは、デバイスコード（固定値 1010）と I²C EEPROM のアドレス端子（A2：0）で決まる値です。 本サンプルプログラムでは、A2：1＝Low、A0＝High slave_addr フォーマット デバイスコード | アドレス端子 | 1 0 1 0 | A2 A1 A0 | 0 であるから、slave_addr への設定値は 0xA2 です。
rom_addr	読み出しを行う ROM アドレスをしてします。
rom_length	読み出しを行うバイト数を指定します。
*rom_data	読み出したデータを格納する先頭アドレスを指定します。

戻り値	説　　明
0	正常終了
1	異常終了（バスビジータータイムアウト）
2	異常終了（転送準備完了待ちタイムアウト）
3	異常終了（アクノリッジタイムアウト）
4	異常終了（転送完了待ちタイムアウト）
5	異常終了（受信完了待ちタイムアウト）
6	異常終了（停止条件検出タイムアウト）

使用例

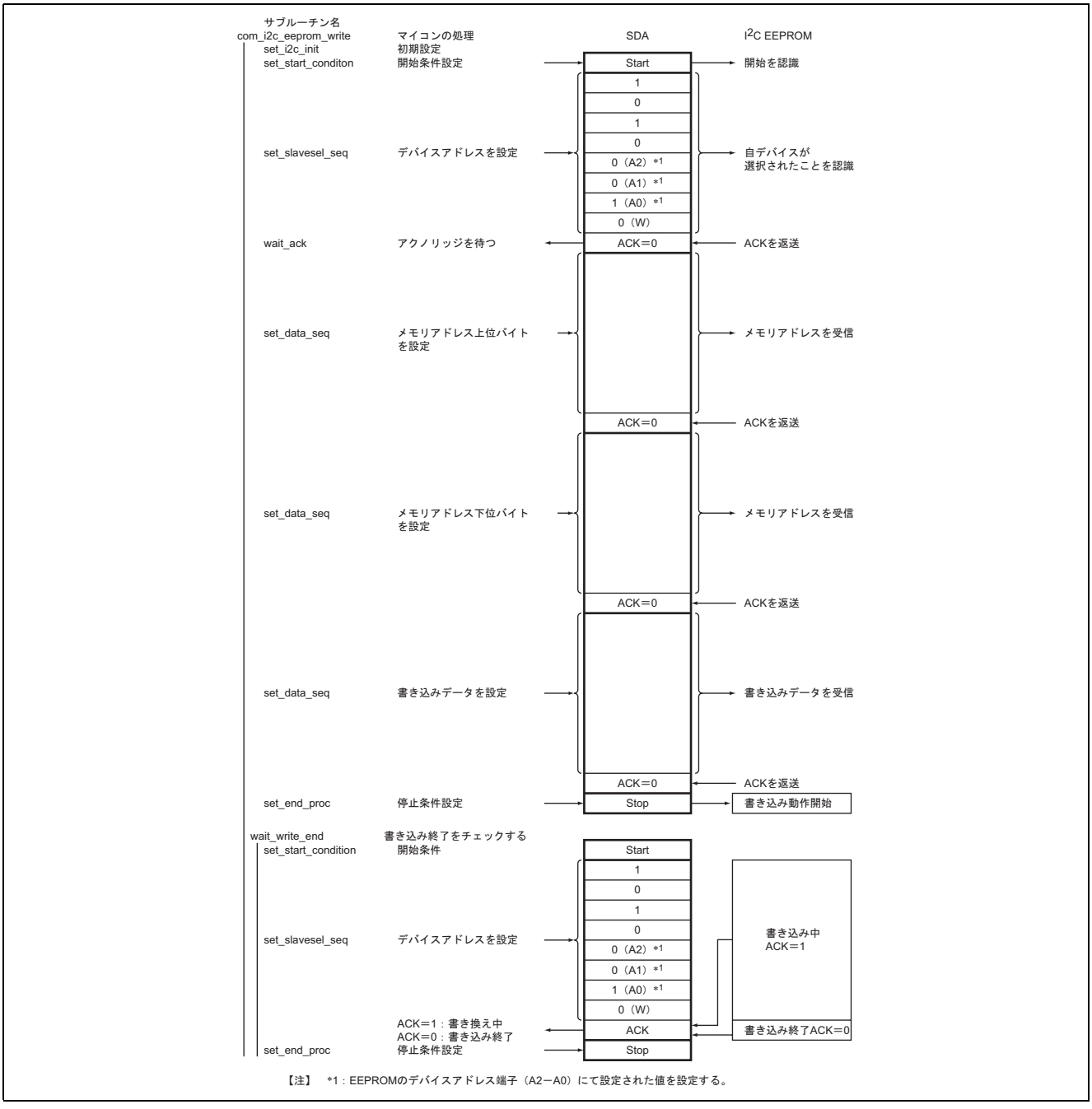
```
int ret ;
unsigned char slave_addr , *rom_data ;
unsigned int rom_length , rom_addr ;

ret = com_i2c_eeprom_page_read (slave_addr , rom_addr , rom_length , *rom_data )
```

3.5 動作説明

以下に、動作説明を示します。

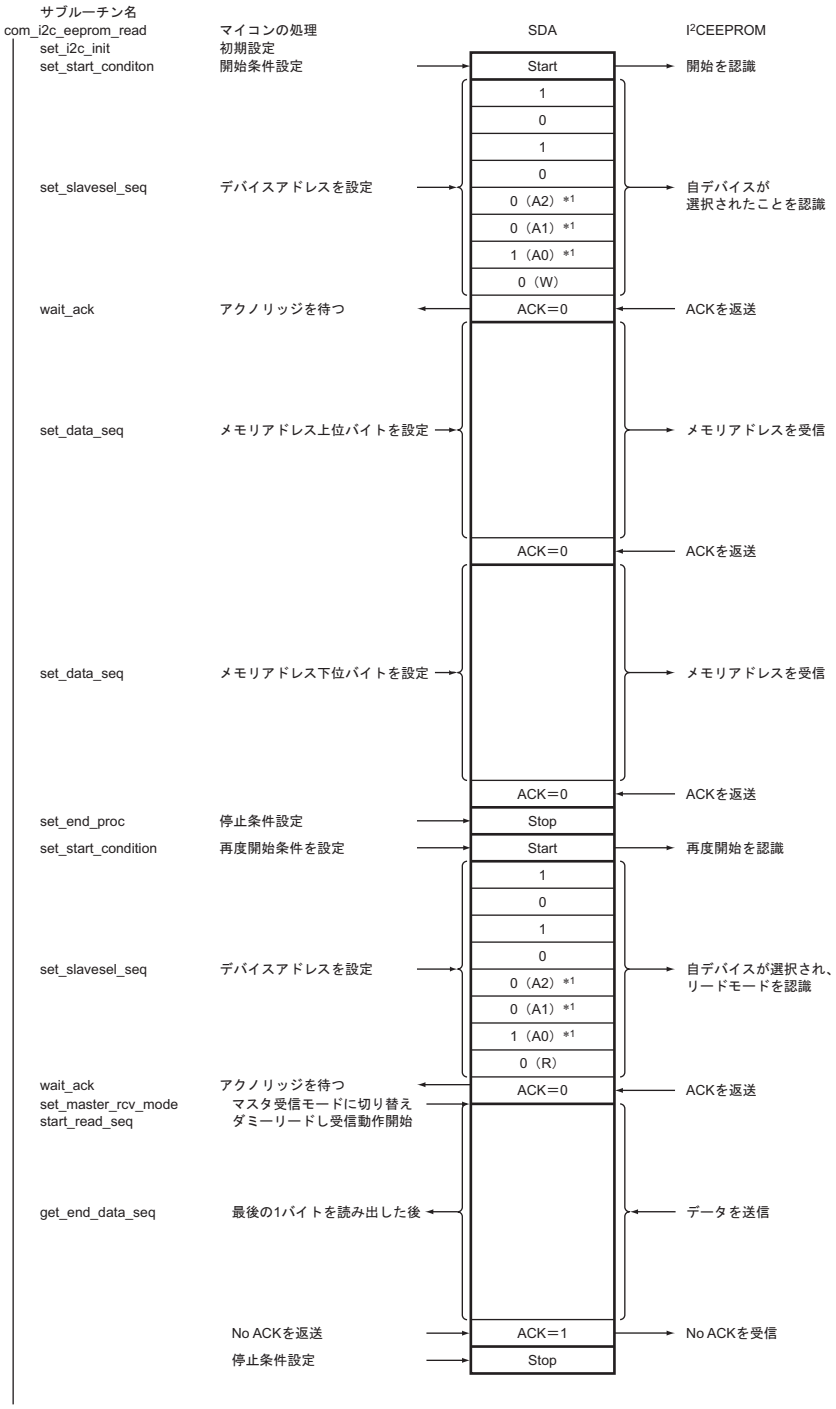
SDA のデータの流れに対する H8 マイコンと EEEPROM の動作を下図に示します。

1. I²C EEPROM に 1 バイトのデータを書き込む



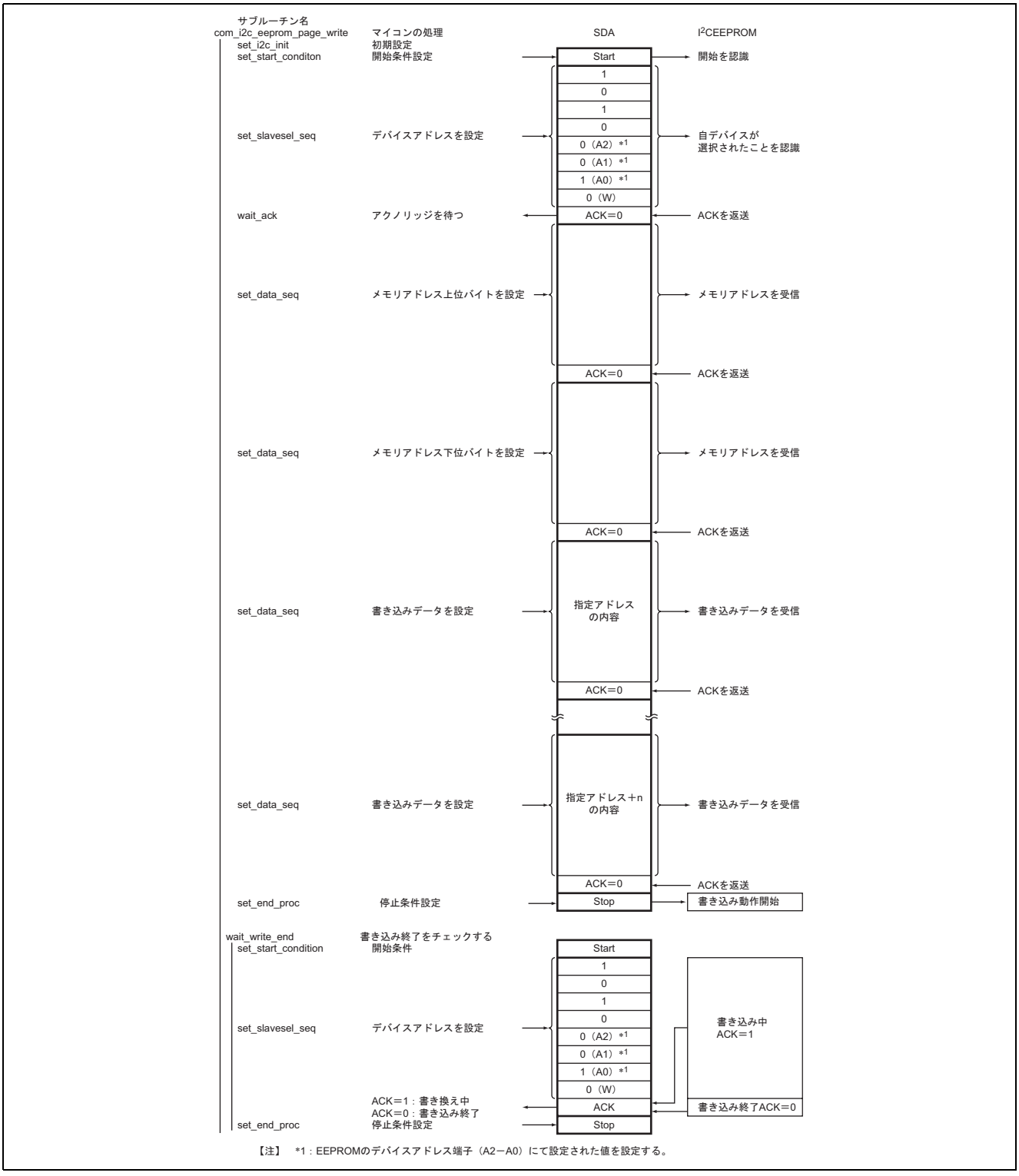
【注】 *1：EEPROMのデバイスアドレス端子（A2－A0）にて設定された値を設定する。

2. I²C EEPROM に 1 バイトのデータを読み出す



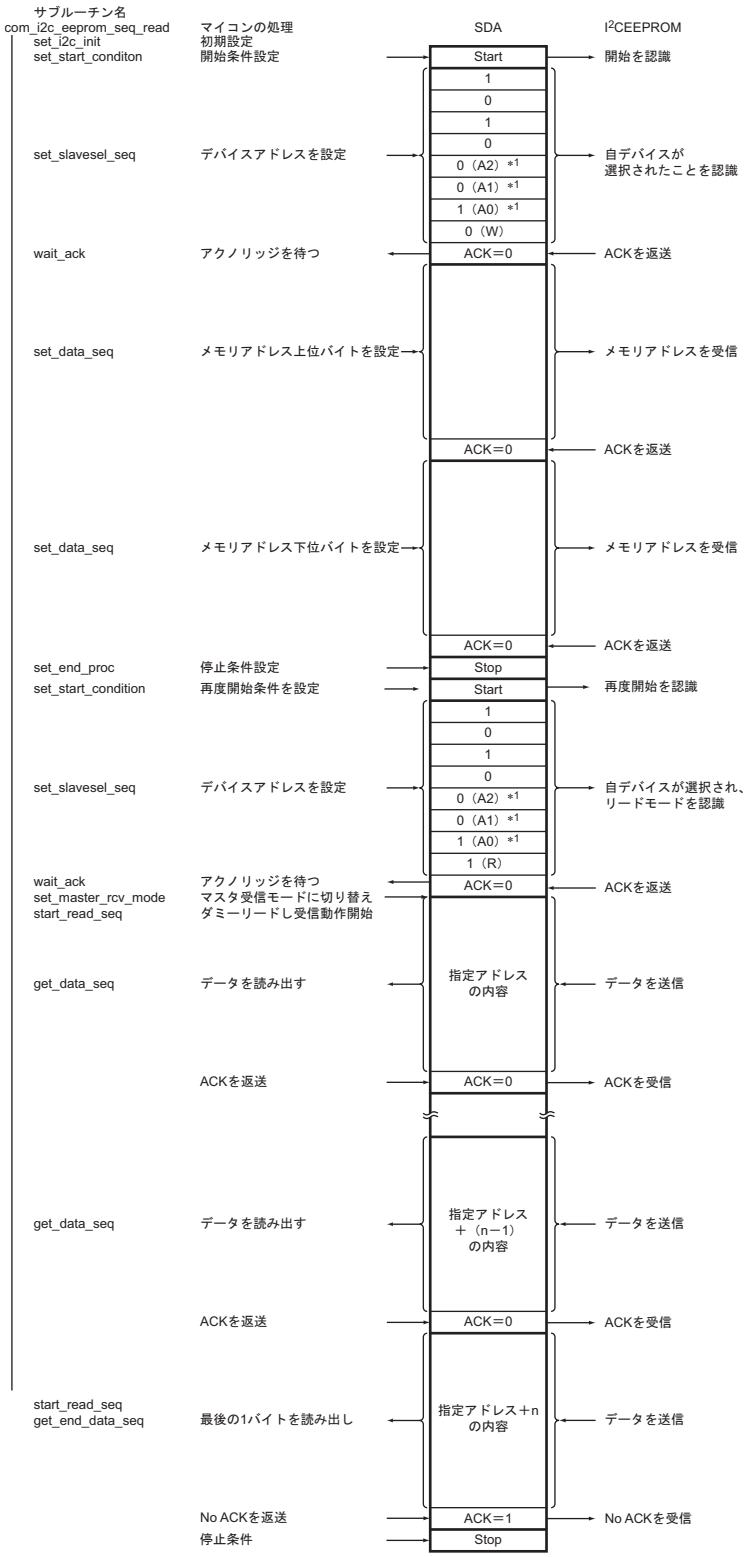
【注】 *1：EEPROMのデバイスアドレス端子（A2－A0）にて設定された値を設定する。

3. I²C EEPROM に連続してデータを書き込む



【注】 *1：EEPROMのデバイスアドレス端子（A2−A0）にて設定された値を設定する。

4. I²C EEPROM に連続してデータを読み出す



【注】 *1：EEPROMのデバイスアドレス端子（A2－A0）にて設定された値を設定する。

3.6 使用レジスター覧

本サンプルプログラムで使用する H8 マイコンの内部レジスタの一覧を示します。内容の詳細は、H8/3664 グループハードウェアマニュアルを参照してください。

1. I²C 関連レジスタ

名　　称	概　　要
I²C バスデータレジスタ（ICDR）	8 ビットのリード／ライト可能なレジスタで、送信時は送信用データレジスタとして、受信時は受信用データレジスタとして機能する。
スレーブアドレスレジスタ（SAR）	スレーブアドレスと転送フォーマットを設定する。
第 2 スレーブアドレスレジスタ（SARX）	第 2 スレーブアドレスと転送フォーマットを設定する。
I²C バスモードレジスタ（ICMR）	転送フォーマットと転送レートを設定する。 ICCR の ICE ビットが 1 のときだけアクセスできる。
I²C バスコントロールレジスタ（ICCR）	I²C バスインタフェースの制御ビットと割り込み要求フラグ
I²C バスステータスレジスタ（ICSR）	ステータスフラグ
タイマシリアルコントロールレジスタ（TSCR）	8 ビットのリード／ライト可能なレジスタで動作モードの制御を行う。

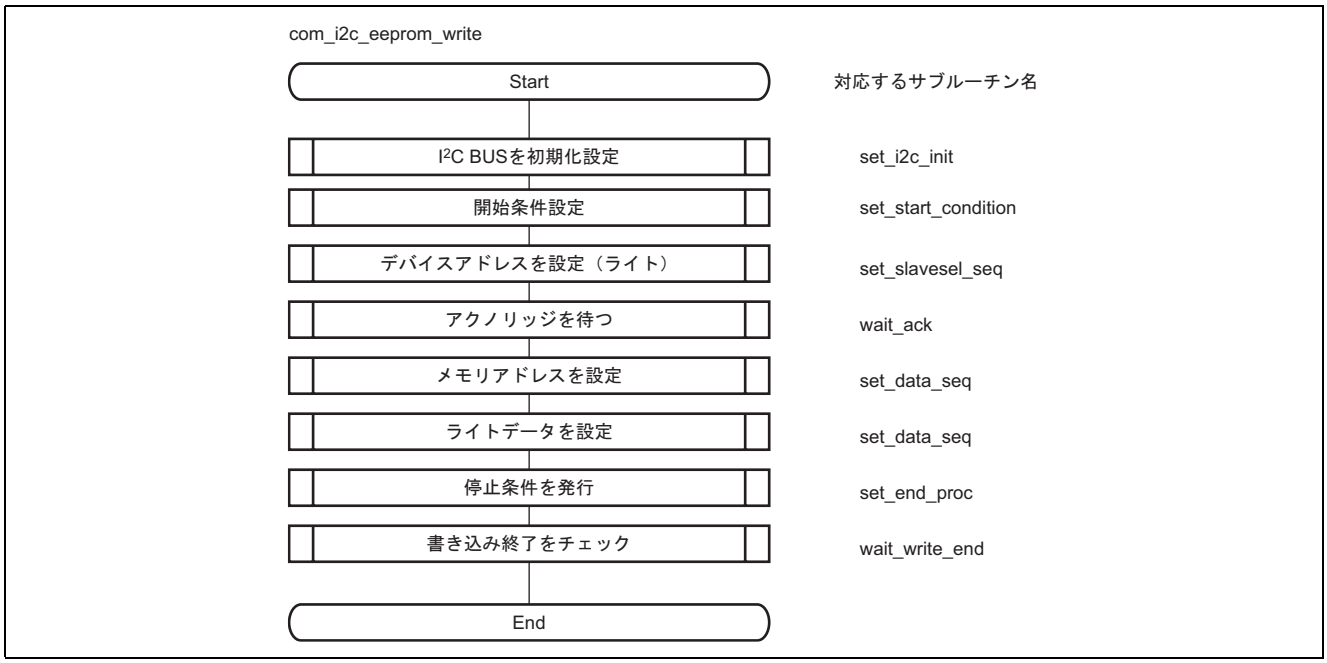
2. タイマ W 関連レジスタ

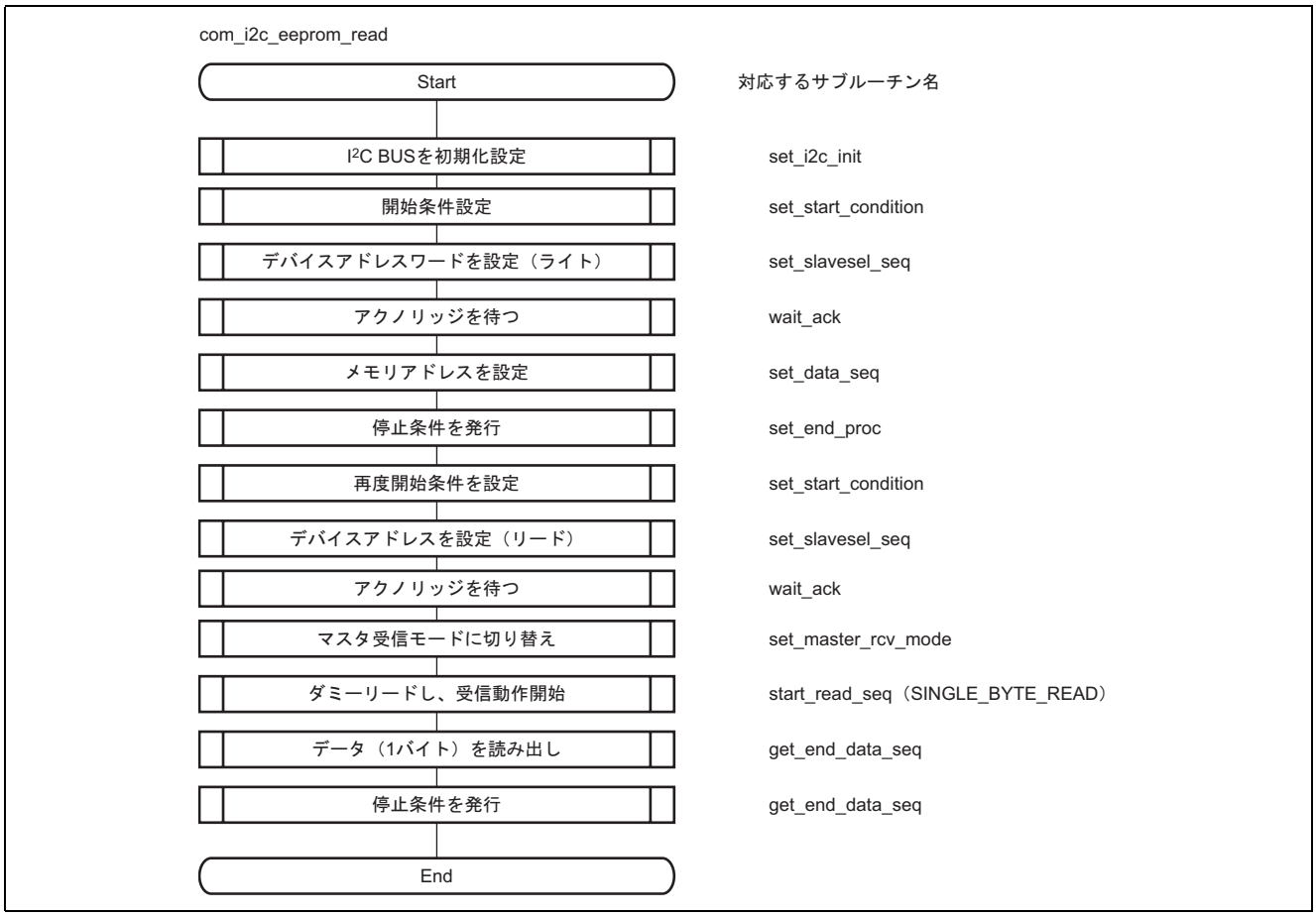
タイマ W は多様な機能を持ちますが、本サンプルプログラムでは、GRA レジスタのコンペアマッチ機能で 10ms ごとの割り込みを発生させるようにしています。

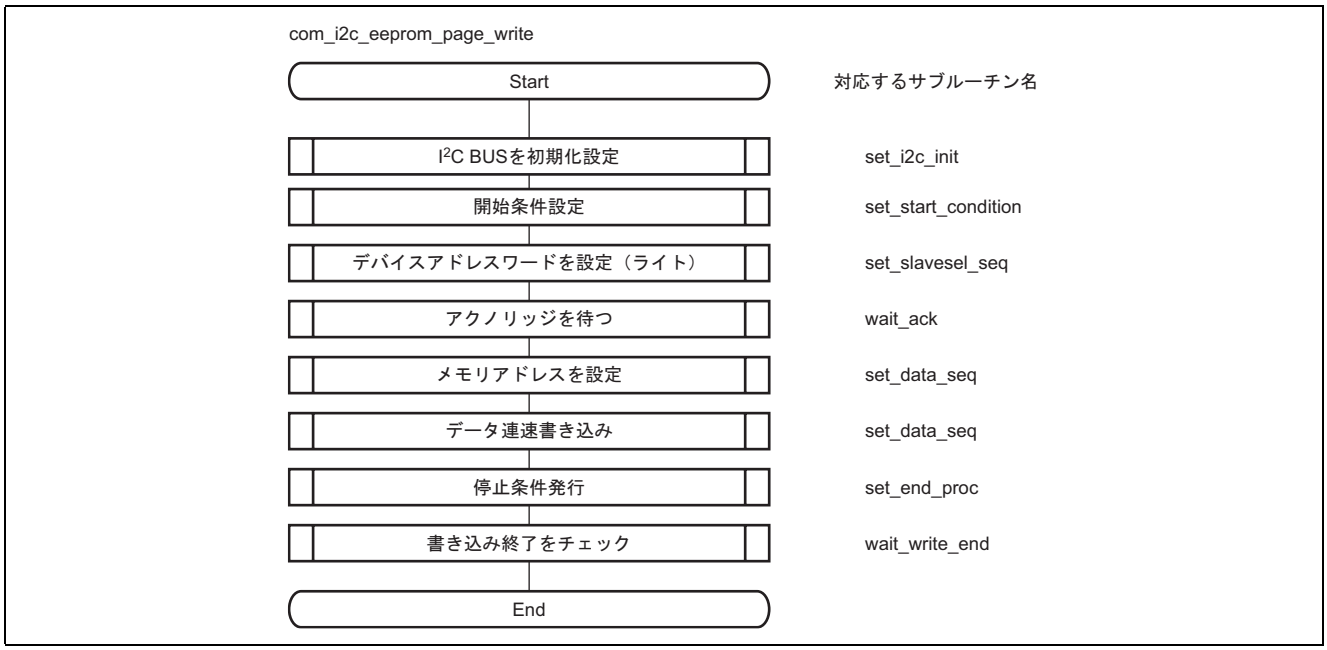
名　　称	概　　要
タイマモードレジスタ W（TMRW）	ジェネラルレジスタの機能やタイマの出力モードなどを選択をする。
タイマコントロールレジスタ W（TCRW）	TCNT のカウンタクロックの選択、カウンタのクリア条件やタイマの初期出力レベルの設定を選択する。
タイマインタラプトイネーブルレジスタ W（TIERW）	タイマWの割り込み要求を制御する。
タイマ I/O コントロールレジスタ 0（TIOR0）	GRA、GRB および FTIOA、FTIOB 端子の機能を選択する。
タイマ I/O コントロールレジスタ 1（TIOR1）	GRC、GRD および FTIOC、FTIOD 端子の機能を選択する。 本サンプルプログラムでは使用しません。
タイマカウンタ（TCNT）	16 ビットのリード／ライト可能なアップカウンタ
ジェネラルレジスタ A、B、C、D（GRA、GRB、GRC、GRD）	16 ビットのリード／ライト可能なレジスタで、アウトプットコンペアレジスタとしてもインプットキャプチャレジスタとしても使用可能

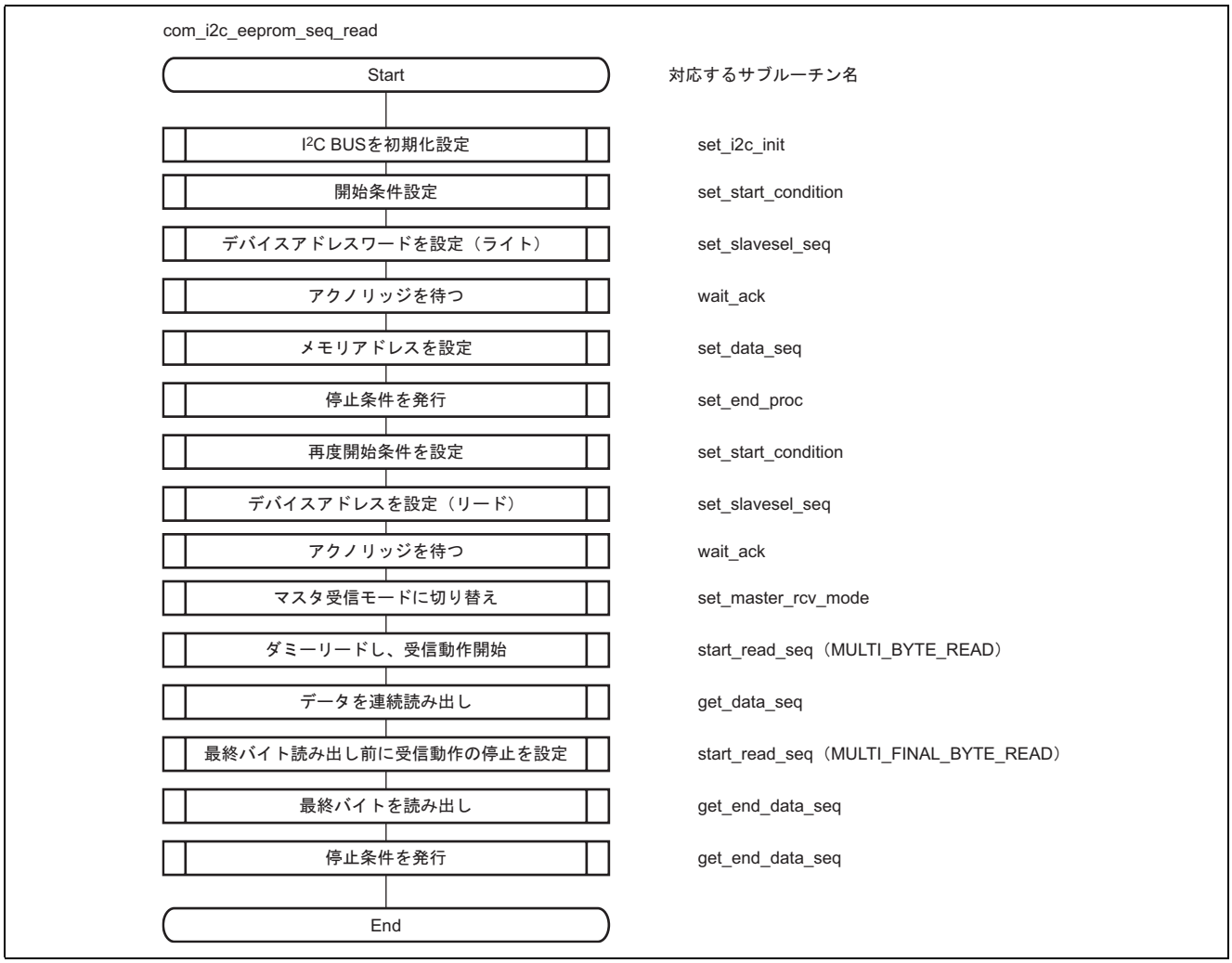
3.7 フローチャート

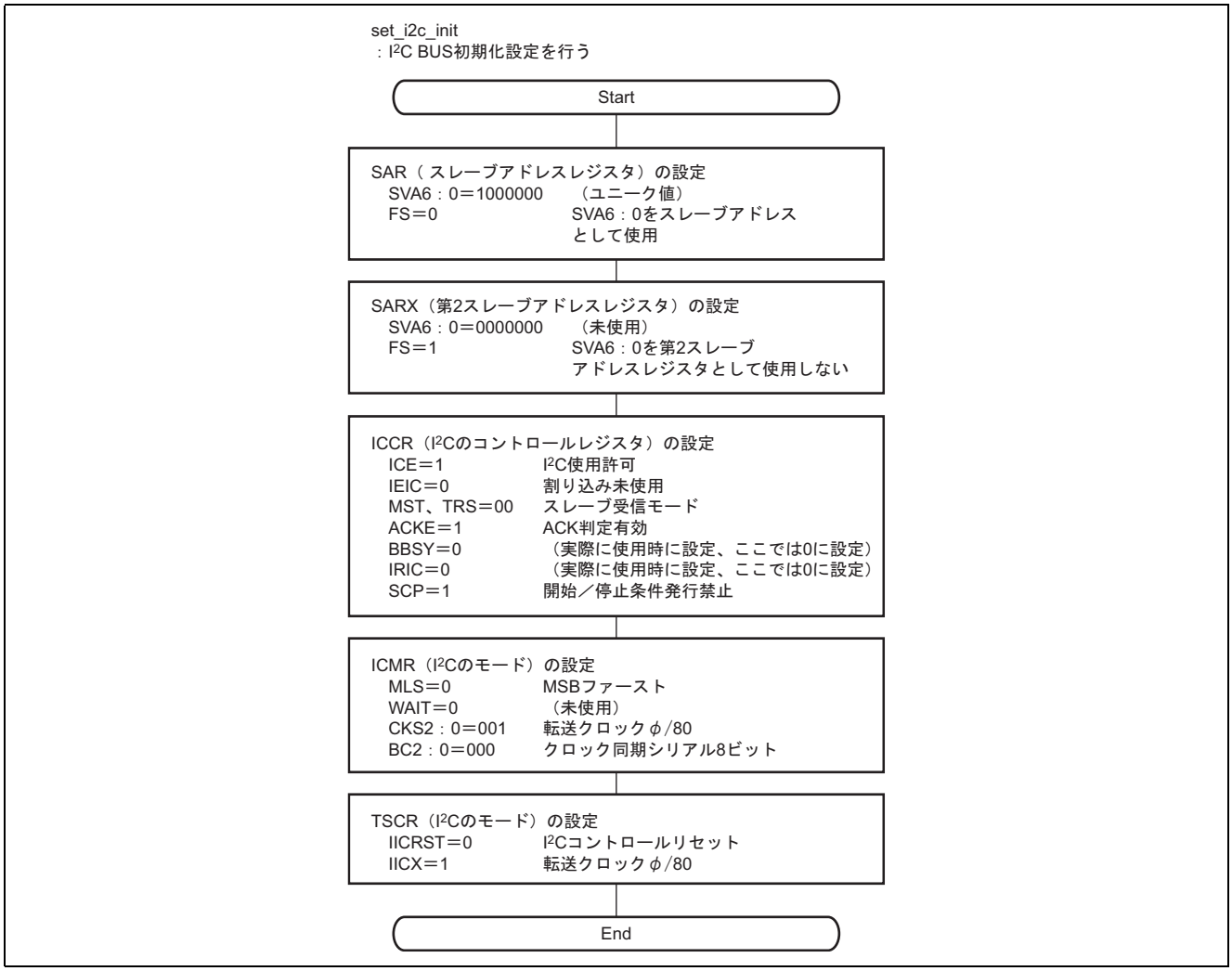
プログラムの処理フローを以下に示します。

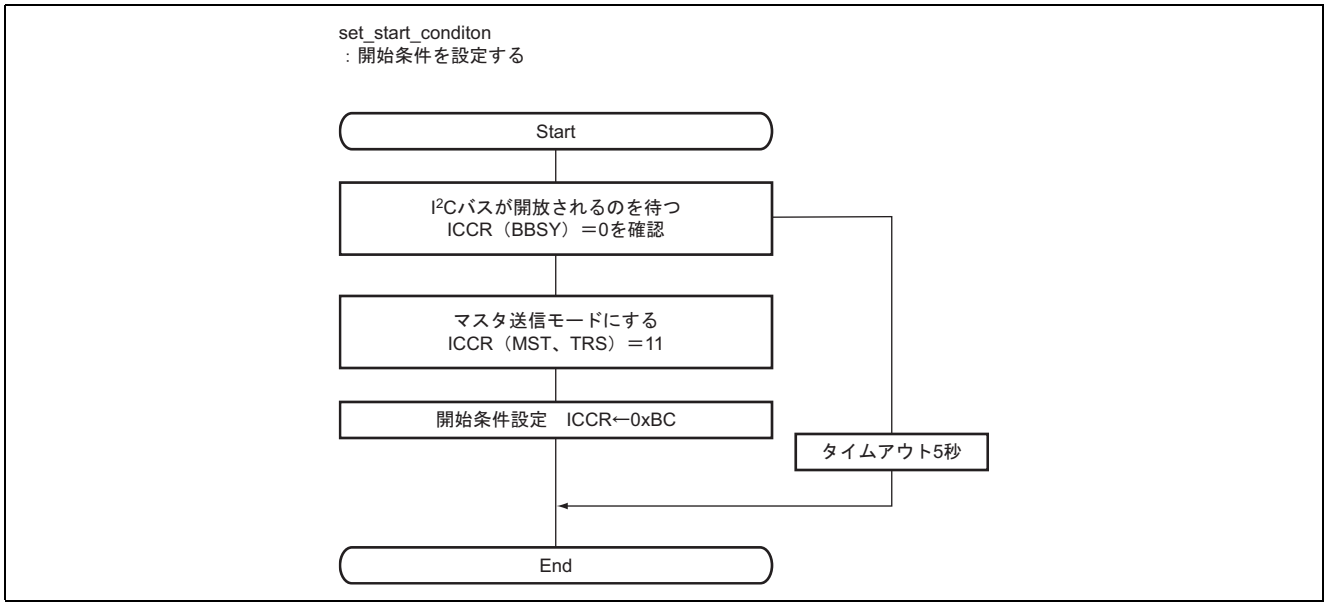




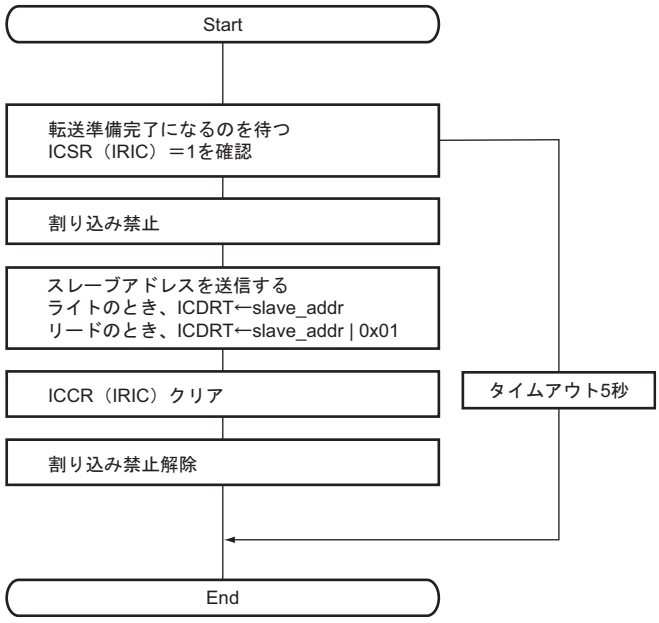

com_i2c_eeprom_page_write
Start
I²C BUSを初期化設定
開始条件設定
デバイスアドレスワードを設定（ライト）
アクノリッジを待つ
メモリアドレスを設定
データ連速書き込み
停止条件発行
書き込み終了をチェック
End
対応するサブルーチン名
set_i2c_init
set_start_condition
set_slavesel_seq
wait_ack
set_data_seq
set_data_seq
set_end_proc
wait_write_end


com_i2c_eeprom_seq_read
Start
I²C BUSを初期化設定
開始条件設定
デバイスアドレスワードを設定（ライト）
アクノリッジを待つ
メモリアドレスを設定
停止条件を発行
再度開始条件を設定
デバイスアドレスを設定（リード）
アクノリッジを待つ
マスタ受信モードに切り替え
ダミーリードし、受信動作開始
データを連続読み出し
最終バイト読み出し前に受信動作の停止を設定
最終バイトを読み出し
停止条件を発行
End
対応するサブルーチン名
set_i2c_init
set_start_condition
set_slavesel_seq
wait_ack
set_data_seq
set_end_proc
set_start_condition
set_slavesel_seq
wait_ack
set_master_rcv_mode
start_read_seq（MULTI_BYTE_READ）
get_data_seq
start_read_seq（MULTI_FINAL_BYTE_READ）
get_end_data_seq
get_end_data_seq

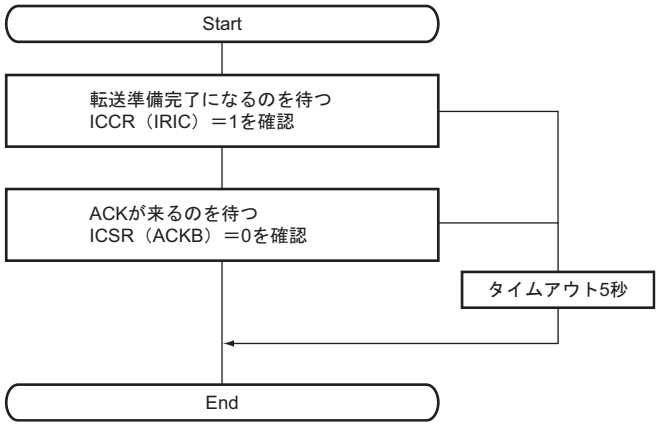

set_i2c_init
：I²C BUS初期化設定を行う
Start
SAR（スレーブアドレスレジスタ）の設定
SVA6：0＝1000000　（ユニーク値）
FS＝0　SVA6：0をスレーブアドレス
として使用
SARX（第2スレーブアドレスレジスタ）の設定
SVA6：0＝0000000　（未使用）
FS＝1　SVA6：0を第2スレーブ
アドレスレジスタとして使用しない
ICCR（I²Cのコントロールレジスタ）の設定
ICE＝1　I²C使用許可
IEIC＝0　割り込み未使用
MST、TRS＝00　スレーブ受信モード
ACKE＝1　ACK判定有効
BBSY＝0　（実際に使用時に設定、ここでは0に設定）
IRIC＝0　（実際に使用時に設定、ここでは0に設定）
SCP＝1　開始／停止条件発行禁止
ICMR（I²Cのモード）の設定
MLS＝0　MSBファースト
WAIT＝0　（未使用）
CKS2：0＝001　転送クロックφ/80
BC2：0＝000　クロック同期シリアル8ビット
TSCR（I²Cのモード）の設定
IICRST＝0　I²Cコントロールリセット
IICX＝1　転送クロックφ/80
End


set_start_conditon
：開始条件を設定する
Start
I²Cバスが開放されるのを待つ
ICCR（BBSY）＝0を確認
マスタ送信モードにする
ICCR（MST、TRS）＝11
開始条件設定　ICCR←0xBC
タイムアウト5秒
End

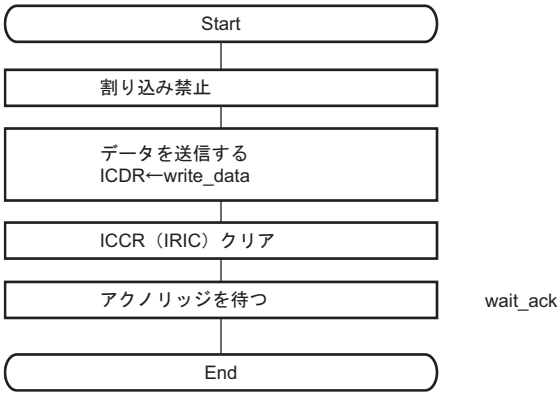
set_slavesel_seq（unsigned char mode , unsigned char slave_addr）
：スレーブ選択処理を実行する
mode：ライト／リードの種別
　0：ライト
　1：リード
slave_addr：EEPROMのデバイスアドレス



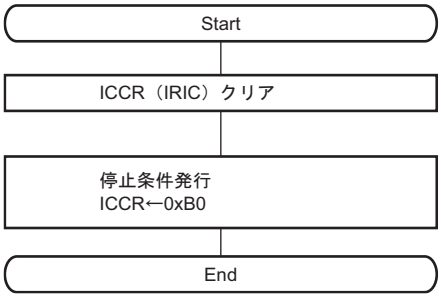
wait_ack
：Ackを待つ



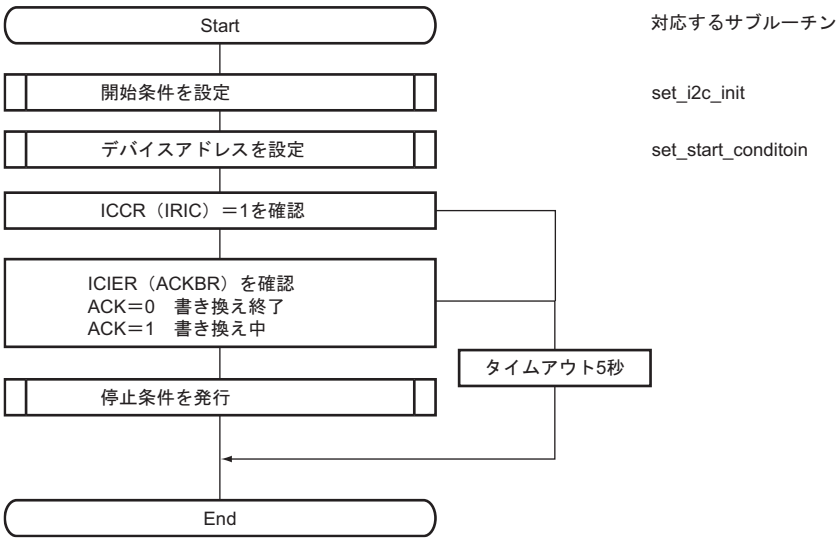
set_data_seq（unsigned char write_data）
：データの設定を行う
write_data：送信するデータ



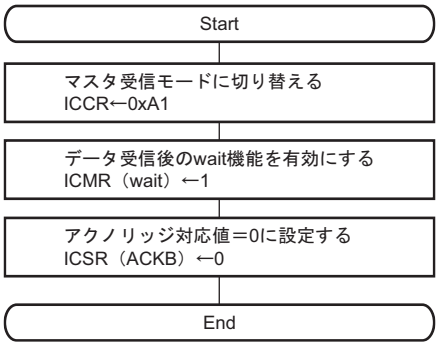
set_end_proc
：I²C終了シーケンスを実行する



wait_write_end（unsigned char slave_addr）
：I²C書き込み終了をチェックする
slave_addr：EEPROMのデバイスアドレス

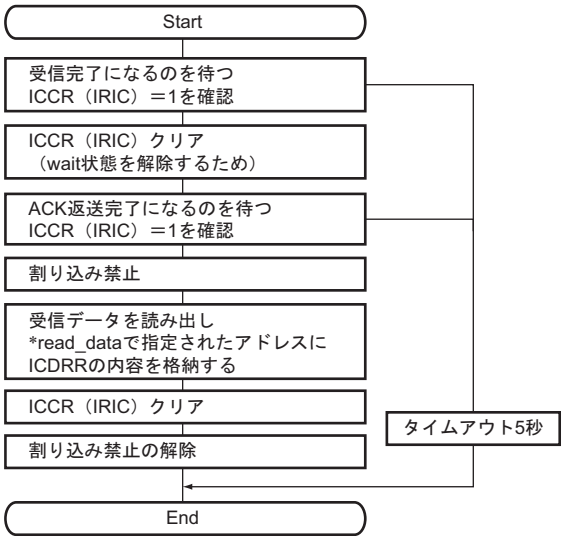


set_master_rcv_mode
：マスタ受信モードに切り替える

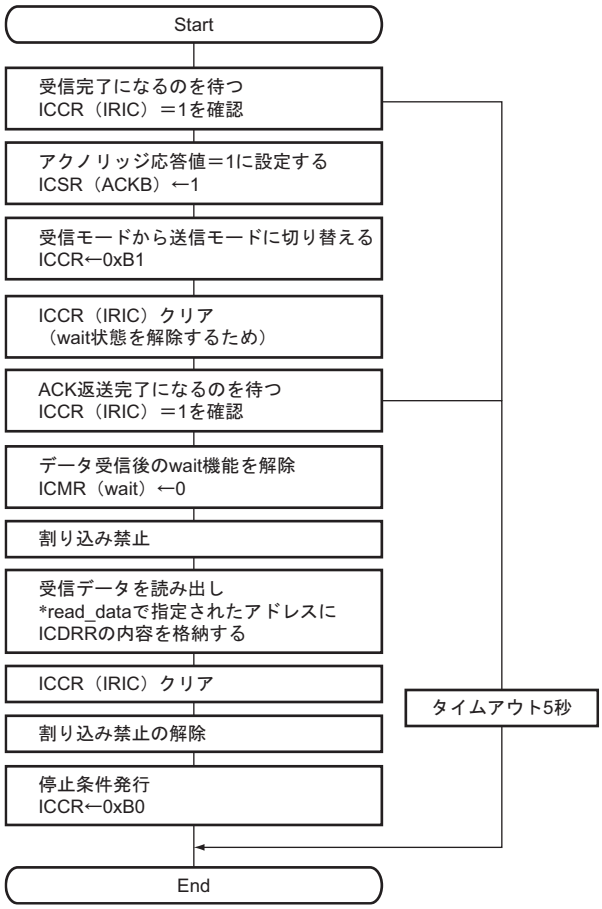


start_read_seq
：リード処理の最初にダミーリードを行う

get_data_seq（unsigned char *read_data）
：データを受信する
*read_data：読み出しデータを格納するアドレス



get_end_data_seq（unsigned char *read_data）
：最終データを受信する
*read_data：読み出しデータを格納するアドレス



3.8 プログラムリスト

```
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 1 . サンプルプログラム 1-A define 定義---------------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/************************************************************************************************************************/
/*   I2CEEPROM アクセス用                                                                                                 */
/************************************************************************************************************************/
#define   CMD_WRITE_OPERATION      0
#define   DATA_READ_OPERATION      1

#define   MULTI_BYTE_READ          0
#define   SINGLE_BYTE_READ         1
#define   MULTI_FINAL_BYTE_READ    2

/************************************************************************************************************************/
/*   I2CEEPROM アクセスエラーコード(0 以外)                                                                               */
/************************************************************************************************************************/
#define   I2C_BBSY_TOUT       1
#define   I2C_IRIC_TOUT       2
#define   I2C_ACKB_TOUT       3
#define   I2C_IRTR_TOUT       4
#define   I2C_TRS_TOUT        5


/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 2 . サンプルプログラム 1-B プロトタイプ宣言------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/************************************************************************************************************************/
/************************************************************************************************************************/
/*   I2C BUS アクセス処理                                                                                                 */
/************************************************************************************************************************/
/************************************************************************************************************************/
void set_i2c_init( );
unsigned int set_start_condition( );
unsigned int wait_ack();
unsigned int set_slavesel_seq (unsigned char mode ,unsigned char slave_addr ) ;
unsigned int set_data_seq(unsigned char write_data);
unsigned int set_end_proc ();
unsigned int set_master_rcv_mode () ;
unsigned int wait_write_end (unsigned char device_addr_code ) ;

void start_read_seq () ;
unsigned int get_end_data_seq (unsigned char *read_data);
unsigned int get_data_seq (unsigned char *read_data);

unsigned int com_i2c_eeprom_read( unsigned char device_addr_code , unsigned int rom_addr , unsigned char *rom_data );
unsigned int com_i2c_eeprom_write( unsigned char device_addr_code , unsigned int rom_addr , unsigned char rom_data );
unsigned int com_i2c_eeprom_seq_read
               ( unsigned char device_addr_code , unsigned int rom_addr , unsigned int rom_length , unsigned char *rom_data ) ;
unsigned int com_i2c_eeprom_page_write
               ( unsigned char device_addr_code , unsigned int rom_addr , unsigned int rom_length , unsigned char *rom_data ) ;
```

```
/* ------------------------------------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------------------------------------ */
/* 3 . サンプルプログラム 1-C ソースコード  ------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------------------------------------ */
/*****************************************************************************************************************************************************/
/*****************************************************************************************************************************************************/
/*****************************************************************************************************************************************************/
/*                                                                                                                                                   */
/*                                    I2C EEPROM 制御                                                                                                */
/*                                                                                                                                                   */
/*****************************************************************************************************************************************************/
/*****************************************************************************************************************************************************/
/*****************************************************************************************************************************************************/

/*****************************************************************************************************************************************************/
/*    1.モジュール名称:set_i2c_init                                                                                                                  */
/*    2.機能概要:I2 アクセス前の初期設定を行う                                                                                                       */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                                       */
/*          000   2002.12.14         上田             新規作成                                                                                       */
/*****************************************************************************************************************************************************/
void set_i2c_init( )
{
     /*************************************************************************************************************************************************/
     /*    SAR             スレーブアドレスレジスタ                                                                                               */
     /*         SVA6:0     = 1000000 (ユニーク値)                                                                                                 */
     /*         FS         = 0   SVA6:0 をスレーブアドレスとして使用                                                                              */
     /*************************************************************************************************************************************************/
          /* ##(program note)################################################################################################################## */
          /* ## SVA6:0 は、スレーブモードで使用します。I2C BUS につながる他のスレーブデバイスと異なるユニークにアドレスを設定すること                ## */
          /* ################################################################################################################################## */
     IIC.SAR.BYTE= 0x80 ;

     /*************************************************************************************************************************************************/
     /*    SARX        第二スレーブアドレスレジスタ                                                                                               */
     /*         SVAX6:0    = 0000000 (未使用)                                                                                                     */
     /*         FS         = 1   SVAX6:0 を第二スレーブアドレスとして使用しない                                                                   */
     /*************************************************************************************************************************************************/
     IIC.SARX.BYTE= 0x01 ;

     /*************************************************************************************************************************************************/
     /*    ICCR       I2C のコントロールレジスタ設定                                                                                              */
     /*         ICE        = 1 I2C 使用許可                                                                                                       */
     /*         IEIC       = 0 割込未使用                                                                                                         */
     /*         MST,TRS    = 00 スレーブ受信モード                                                                                                */
     /*         ACKE       = 1 ACK 判定有効                                                                                                       */
     /*         BBSY       = 0 (実際に使用時に設定、ここでは 0 に設定)                                                                            */
     /*         IRIC       = 0 (実際に使用時に設定、ここでは 0 に設定)                                                                            */
     /*         SCP        = 1 開始/停止条件発行禁止                                                                                              */
     /*************************************************************************************************************************************************/
     IIC.ICCR.BYTE  = 0x89 ;
```

```
      /*****************************************************************************************************************************/
      /*   ICMR      I2C のモード設定                                                                                              */
      /*        MLS       = 0 MSB ファースト                                                                                       */
      /*        WAIT      = 0 (未使用)                                                                                             */
      /*        CKS2:0    = 001 転送クロックφ/80                                                                                   */
      /*        BC2:0     = 000 クロック同期シリアル 8bit                                                                          */
      /*****************************************************************************************************************************/
      IIC.ICMR.BYTE= 0x08 ;
           /* ##(program note)############################################################################################## */
           /* ## CKS2:0 は、必要とする転送レートに従って設定値を変更します                                                    ## */
           /* ## 詳細は、H8/3687 ハードウェアマニュアルを参照してください                                                     ## */
           /* ############################################################################################################## */

      /*****************************************************************************************************************************/
      /*   TSCR      I2C のモード設定                                                                                              */
      /*        IICRST    = 0 I2C コントロールリセット                                                                             */
      /*        IICX      = 1 転送クロックφ/80                                                                                    */
      /*****************************************************************************************************************************/
      TSCR.BYTE= 0x01 ;
           /* ##(program note)############################################################################################## */
           /* ## IICX は、必要とする転送レートに従って設定値を変更します                                                      ## */
           /* ## 詳細は、H8/3687 ハードウェアマニュアルを参照してください                                                     ## */
           /* ############################################################################################################## */

}

/*****************************************************************************************************************************/
/*   1.モジュール名称:set_start_condition                                                                                        */
/*   2.機能概要:I2C 開始条件を設定する                                                                                           */
/*   3.来歴:REV  作成/改訂日付     作成/改訂者     改訂内容                                                                      */
/*         000   2002.12.14       上田          新規作成                                                                      */
/*****************************************************************************************************************************/
unsigned int set_start_condition( )
{
      int ret , timer_wk;

      ret = NORMAL_END ;

      /*****************************************************************************************************************************/
      /*   ICCR(BBSY)=0 を確認                                                                                                   */
      /*****************************************************************************************************************************/
      com_timer.wait_100ms_scan = 50 ;
      while (IIC.ICCR.BIT.BBSY == 1){                                   /* I2C BUS が開放されるのを待つ                        */
           timer_wk = com_timer.wait_100ms_scan ;
           if (timer_wk == 0){                                          /* 5s 間 1 のままならエラーで抜ける                    */
                ret = I2C_BBSY_TOUT;                                    /* 異常終了(timeout)                                   */
                goto exit ;
           }

           #ifdef UT
                IIC.ICCR.BIT.BBSY = 0;
           #endif

      }
```

```
     /*****************************************************************************************************************************************/
     /*    マスタ送信モードにする                                                                                                             */
     /*****************************************************************************************************************************************/
     IIC.ICCR.BYTE  = 0xB9 ;                                                    /* マスタ送信モードにする                                    */
          /* ##(program note)################################################################################################################ */
          /* ## ACKE(ACK 判定有効)をリセットしてしまわないように注意のこと                                                                  ## */
          /* ################################################################################################################################ */

     /*****************************************************************************************************************************************/
     /*    開始条件設定                                                                                                                       */
     /*****************************************************************************************************************************************/
     IIC.ICCR.BYTE  = 0xBC ;
          /* ##(program note)################################################################################################################ */
          /* ## 開始条件設定のための bit2,0 の設定は同時に設定する必要があるため Byte 単位に書き込む必要がある。                            ## */
          /* ## 1bit ずつ設定すると正しく開始条件が設定されないので注意。                                                                    ## */
          /* ################################################################################################################################ */
          /* ##(program note)################################################################################################################ */
          /* ## ACKE(ACK 判定有効)をリセットしてしまわないように注意のこと                                                                  ## */
          /* ################################################################################################################################ */

exit :
     return (ret);

}

/*****************************************************************************************************************************************/
/*    1.モジュール名称:wait_ack                                                                                                          */
/*    2.機能概要:I2C ACK を待つ                                                                                                          */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                           */
/*          000   2002.12.14        上田             新規作成                                                                           */
/*****************************************************************************************************************************************/
unsigned int wait_ack ()
{
     int ret , timer_wk;

     ret = NORMAL_END ;

     /*****************************************************************************************************************************************/
     /*    ICCR(IRIC)=1 を確認                                                                                                             */
     /*****************************************************************************************************************************************/
     com_timer.wait_100ms_scan = 50 ;
     while (IIC.ICCR.BIT.IRIC == 0){                                            /* 転送準備完了になるまで待つ                                */
          timer_wk = com_timer.wait_100ms_scan ;
          if (timer_wk == 0){                                                   /* 5s 間 1 のままならエラーで抜ける                          */
               ret = I2C_IRIC_TOUT;                                             /* 異常終了(timeout)                                         */
               goto exit ;
          }

          #ifdef UT
               IIC.ICCR.BIT.IRIC = 1 ;
          #endif
     }
```

```
     /*****************************************************************************************************************************************/
     /*    ICSR(ACKB)=0 を確認                                                                                                                */
     /*****************************************************************************************************************************************/
     com_timer.wait_100ms_scan = 50 ;
     while (IIC.ICSR.BIT.ACKB == 1){                                        /* ACKB=0 が戻るのを待つ                                        */
          timer_wk = com_timer.wait_100ms_scan ;
          if (timer_wk == 0){                                               /* 5s 間 1 のままならエラーで抜ける                             */
               ret = I2C_ACKB_TOUT;                                         /* 異常終了(timeout)                                            */
               goto exit ;
          }

          #ifdef UT
               IIC.ICSR.BIT.ACKB = 0 ;
          #endif

     }

exit :
     return (ret);

}

/*****************************************************************************************************************************************/
/*    1.モジュール名称:set_slavesel_seq                                                                                                  */
/*    2.機能概要:I2C スレーブ選択処理を実行する                                                                                          */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                           */
/*          000   2002.12.14        上田          新規作成                                                                               */
/*****************************************************************************************************************************************/
unsigned int set_slavesel_seq (unsigned char mode ,unsigned char slave_addr )
{
     int ret , timer_wk;
     unsigned char write_data ;

     ret = NORMAL_END ;

     /*****************************************************************************************************************************************/
     /*    ICCR(IRIC)=1 を確認                                                                                                                */
     /*****************************************************************************************************************************************/
     com_timer.wait_100ms_scan = 50 ;
     while (IIC.ICCR.BIT.IRIC == 0){                                        /* 転送準備完了になるまで待つ                                   */
          timer_wk = com_timer.wait_100ms_scan ;
          if (timer_wk == 0){                                               /* 5s 間 1 のままならエラーで抜ける                             */
               ret = I2C_IRIC_TOUT;                                         /* 異常終了(timeout)                                            */
               goto exit ;
          }

          #ifdef UT
               IIC.ICCR.BIT.IRIC = 1 ;
          #endif
     }

     /*****************************************************************************************************************************************/
     /*    スレーブアドレスの設定                                                                                                             */
     /*****************************************************************************************************************************************/
     if (mode == DATA_READ_OPERATION){
          slave_addr = slave_addr | 0x01 ;
     }
```

```
    /*****************************************************************************************************************************/
    /*    割込み禁止                                                                                                             */
    /*****************************************************************************************************************************/
    set_imask_ccr(1);                                                   /* 割込み禁止                                          */

    /*****************************************************************************************************************************/
    /*    データの書き込み                                                                                                       */
    /*****************************************************************************************************************************/
    IIC.ICDR = slave_addr ;

    /*****************************************************************************************************************************/
    /*    ICCR(IRIC)クリア                                                                                                       */
    /*****************************************************************************************************************************/
    IIC.ICCR.BIT.IRIC = 0 ;
            /* ##(program note)################################################################################################ */
            /* ## ICDR のライトと IRIC のクリアは 1byte 分のデータ転送時間が経過しない間に、連続的に write すること。                ## */
            /* ## 。このため、この操作の間は全ての割込みを禁止状態にする。                                                        ## */
            /* ################################################################################################################ */

    /*****************************************************************************************************************************/
    /*    割込み禁止の解除                                                                                                       */
    /*****************************************************************************************************************************/
    set_imask_ccr(0);                                                   /* 割込み禁止解除                                      */

exit :
    return (ret);

}

/*****************************************************************************************************************************/
/*    1.モジュール名称:set_data_seq                                                                                              */
/*    2.機能概要:I2C データ設定処理を実行する                                                                                    */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                   */
/*          000   2002.12.14         上田           新規作成                                                                   */
/*****************************************************************************************************************************/
unsigned int set_data_seq (unsigned char write_data)
{
    int ret , timer_wk;

    ret = NORMAL_END ;

    /*****************************************************************************************************************************/
    /*    割込み禁止                                                                                                             */
    /*****************************************************************************************************************************/
    set_imask_ccr(1);                                                   /* 割込み禁止                                          */

    /*****************************************************************************************************************************/
    /*    データの書き込み                                                                                                       */
    /*****************************************************************************************************************************/
    IIC.ICDR = write_data ;

    /*****************************************************************************************************************************/
    /*    ICCR(IRIC)クリア                                                                                                       */
    /*****************************************************************************************************************************/
    IIC.ICCR.BIT.IRIC = 0 ;
            /* ##(program note)################################################################################################ */
            /* ## ICDR のライトと IRIC のクリアは 1byte 分のデータ転送時間が経過しない間に、連続的に write すること。                ## */
            /* ## このため、この操作の間は全ての割込みを禁止状態にする。                                                          ## */
            /* ################################################################################################################ */
```

```
    /*****************************************************************************************************************************************/
    /*    割込み禁止の解除                                                                                                                   */
    /*****************************************************************************************************************************************/
    set_imask_ccr(0);                                                    /* 割込み禁止解除                                                  */

    /*****************************************************************************************************************************************/
    /*    Acknowlege を待つ                                                                                                                  */
    /*****************************************************************************************************************************************/
    ret = wait_ack() ;
    if (ret !=0) { goto exit ;}

exit :
    return (ret);
}
/*****************************************************************************************************************************************/
/*    1.モジュール名称:set_end_proc                                                                                                          */
/*    2.機能概要:I2C 終了シーケンスを実行する                                                                                                */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                               */
/*          000   2002.12.14        上田          新規作成                                                                                 */
/*****************************************************************************************************************************************/
unsigned int set_end_proc ()
{
    int ret , timer_wk;

    ret = NORMAL_END ;

    /*****************************************************************************************************************************************/
    /*    ICCR(IRIC)クリア                                                                                                                   */
    /*****************************************************************************************************************************************/
    IIC.ICCR.BIT.IRIC = 0 ;

    /*****************************************************************************************************************************************/
    /*    停止条件発行                                                                                                                       */
    /*****************************************************************************************************************************************/
    IIC.ICCR.BYTE = 0xB0 ;

exit :
    return (ret);

}

/*****************************************************************************************************************************************/
/*    1.モジュール名称:wait_write_end                                                                                                        */
/*    2.機能概要:I2C 書き込み終了をチェックする                                                                                              */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                               */
/*          000   2002.12.14        上田          新規作成                                                                                 */
/*****************************************************************************************************************************************/
unsigned int wait_write_end (unsigned char slave_addr )
{
    int ret , timer_wk;
    unsigned char ack_status ;

    ret = NORMAL_END ;

    com_timer.wait_100ms = 50 ;

    do{
```

```
        /*****************************************************************************************************************************/
        /*    開始条件を設定する                                                                                                       */
        /*****************************************************************************************************************************/
        ret = set_start_condition() ;                                       /* 開始条件を設定する                                    */
             if (ret !=0) { goto exit ;}

        /*****************************************************************************************************************************/
        /*    デバイスアドレスワードを設定する(Write)                                                                                  */
        /*****************************************************************************************************************************/
        ret = set_slavesel_seq ( CMD_WRITE_OPERATION , slave_addr ) ;
             if (ret !=0) { goto exit ;}

        /*****************************************************************************************************************************/
        /*    ICCR(IRIC)=1 を確認                                                                                                      */
        /*****************************************************************************************************************************/
        com_timer.wait_100ms_scan = 50 ;
        while (IIC.ICCR.BIT.IRIC == 0){                                     /* 転送準備完了になるまで待つ                            */
             timer_wk = com_timer.wait_100ms_scan ;
             if (timer_wk == 0){                                            /* 5s 間 1 のままならエラーで抜ける                      */
                  ret = I2C_IRIC_TOUT;                                      /* 異常終了(timeout)                                     */
                  goto exit ;
             }

             #ifdef UT
                  IIC.ICCR.BIT.IRIC = 1 ;
             #endif

        }

        /*****************************************************************************************************************************/
        /*    ICIER(ACKBR)を確認    ACK=0 書き換え終了、=1 書き換え中                                                                  */
        /*****************************************************************************************************************************/
        if (com_timer.wait_100ms == 0){                                     /* 5s 間 1 のままならエラーで抜ける                      */
             ret = I2C_ACKB_TOUT;                                           /* 異常終了(timeout)                                     */
             goto exit ;
        }

        #ifdef UT
             IIC.ICSR.BIT.ACKB = 1 ;
        #endif
        ack_status = IIC.ICSR.BIT.ACKB ;

        /*****************************************************************************************************************************/
        /*    停止条件を発行する                                                                                                       */
        /*****************************************************************************************************************************/
        ret = set_end_proc ( ) ;
             if (ret !=0) { goto exit ;}

    } while (ack_status == 1) ;                                             /* ACK=1 の間 loop                                       */

exit :
    return (ret);

}
```

```
/*****************************************************************************************************************************************************/
/*    1.モジュール名称:set_master_rcv_mode                                                                                                           */
/*    2.機能概要:Master 受信モードに切り替える                                                                                                       */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                                       */
/*          000  2002.12.14        上田           新規作成                                                                                         */
/*****************************************************************************************************************************************************/
unsigned int set_master_rcv_mode ()
{
     int ret , timer_wk;
     unsigned char dummy_data ;

     ret = NORMAL_END ;


     /*****************************************************************************************************************************************************/
     /*    マスタ受信モードに切り替える                                                                                                               */
     /*****************************************************************************************************************************************************/
     IIC.ICCR.BYTE = 0xA1     ;

     /*****************************************************************************************************************************************************/
     /*    ICMR(WAIT)をセットする                                                                                                                     */
     /*****************************************************************************************************************************************************/
     IIC.ICMR.BIT.WAIT = 1 ;

     /*****************************************************************************************************************************************************/
     /*    ICSR(ACKB)を 0 クリアする(アクノリッジデータの設定)                                                                                        */
     /*****************************************************************************************************************************************************/
     IIC.ICSR.BIT.ACKB = 0 ;

exit :
     return (ret);
}

/*****************************************************************************************************************************************************/
/*    1.モジュール名称:start_read_seq                                                                                                                */
/*    2.機能概要:read 処理の最初に dummy read を行う                                                                                                 */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                                       */
/*          000   2002.12.14        上田           新規作成                                                                                         */
/*****************************************************************************************************************************************************/
void start_read_seq ()
{
     unsigned char dummy_data ;

     /*****************************************************************************************************************************************************/
     /*    割込み禁止                                                                                                                                 */
     /*****************************************************************************************************************************************************/
     set_imask_ccr(1);                                                    /* 割込み禁止                                                             */

     /*****************************************************************************************************************************************************/
     /*    duumy read を行うと受信動作が開始される                                                                                                    */
     /*****************************************************************************************************************************************************/
     dummy_data = IIC.ICDR ;
          /* ##(program note)################################################################################################################ */
          /* ## duumy read により受信動作が始まり、SCL に同期してデバイスよりデータが送られてくる。                                                ## */
          /* ## 先の ICSR(ACKB) = 0 により、9 回目の SCL に同期してデバイスへ Low が送出される。                                                   ## */
          /* ################################################################################################################################ */
```

```
    /*****************************************************************************************************************************************/
    /*   ICCR(IRIC)クリア                                                                                                                    */
    /*****************************************************************************************************************************************/
    IIC.ICCR.BIT.IRIC = 0 ;
            /* ##(program note)################################################################################################ */
            /* ## ICDR のライトと IRIC のクリアは 1byte 分のデータ転送時間が経過しない間に、連続的に write すること。                    ## */
            /* ## このため、この操作の間は全ての割込みを禁止状態にする。                                                            ## */
            /* ################################################################################################################ */

    /*****************************************************************************************************************************************/
    /*   割込み禁止の解除                                                                                                                    */
    /*****************************************************************************************************************************************/
    set_imask_ccr(0);                                                   /* 割込み禁止解除                                                  */

}

/*****************************************************************************************************************************************/
/*   1.モジュール名称:get_data_seq                                                                                                       */
/*   2.機能概要:I2C ターゲットデバイスからデータを読み出す                                                                               */
/*   3.来歴:REV  作成/改訂日付     作成/改訂者     改訂内容                                                                              */
/*         000  2002.12.14       上田          新規作成                                                                                  */
/*****************************************************************************************************************************************/
unsigned int get_data_seq (unsigned char *read_data)
{
    int ret , timer_wk;
    unsigned char dummy_data ;

    ret = NORMAL_END ;

    /*****************************************************************************************************************************************/
    /*   ICCR(IRIC)=1 を確認                                                                                                                 */
    /*****************************************************************************************************************************************/
    com_timer.wait_100ms_scan = 50 ;
    while (IIC.ICCR.BIT.IRIC == 0){                                     /* 転送準備完了になるまで待つ                                      */
        timer_wk = com_timer.wait_100ms_scan ;
        if (timer_wk == 0){                                             /* 5s 間 1 のままならエラーで抜ける                                */
            ret = I2C_IRIC_TOUT;                                        /* 異常終了(timeout)                                               */
            goto exit ;
        }

        #ifdef UT
            IIC.ICCR.BIT.IRIC = 1 ;
        #endif
    }

    /*****************************************************************************************************************************************/
    /*   ICCR(IRIC)クリア(wait 状態を解除するため)                                                                                           */
    /*****************************************************************************************************************************************/
    IIC.ICCR.BIT.IRIC = 0 ;
```

```
	/*****************************************************************************************************************************************/
	/*	ICCR(IRIC)=1を確認                                                                                                                   */
	/*****************************************************************************************************************************************/
	com_timer.wait_100ms_scan = 50 ;
	while (IIC.ICCR.BIT.IRIC == 0){                                         /* 転送準備完了になるまで待つ                                   */
		timer_wk = com_timer.wait_100ms_scan ;
		if (timer_wk == 0){                                                 /* 5s間1のままならエラーで抜ける                                */
			ret = I2C_IRIC_TOUT;                                            /* 異常終了(timeout)                                            */
			goto exit ;
		}

		#ifdef UT
			IIC.ICCR.BIT.IRIC = 1 ;
		#endif
	}

	/*****************************************************************************************************************************************/
	/*	割込み禁止                                                                                                                           */
	/*****************************************************************************************************************************************/
	set_imask_ccr(1);                                                       /* 割込み禁止                                                   */

	/*****************************************************************************************************************************************/
	/*	受信データを読み出す                                                                                                                 */
	/*****************************************************************************************************************************************/
	*read_data = IIC.ICDR ;                                                 /* data read                                                    */

	/*****************************************************************************************************************************************/
	/*	ICCR(IRIC)クリア                                                                                                                     */
	/*****************************************************************************************************************************************/
	IIC.ICCR.BIT.IRIC = 0 ;
		/* ##(program note)#################################################################################################### */
		/* ## ICDRのライトとIRICのクリアは1byte分のデータ転送時間が経過しない間に、連続的にwriteすること。                          ## */
		/* ## このため、この操作の間は全ての割込みを禁止状態にする。                                                             ## */
		/* ################################################################################################################### */

	/*****************************************************************************************************************************************/
	/*	割込み禁止の解除                                                                                                                     */
	/*****************************************************************************************************************************************/
	set_imask_ccr(0);                                                       /* 割込み禁止解除                                               */

exit :
	return (ret);
}

/*****************************************************************************************************************************************/
/*	1.モジュール名称:get_end_data_seq                                                                                                    */
/*	2.機能概要:I2C ターゲットデバイスからデータを読み出す                                                                                */
/*	3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                             */
/*	        000   2002.12.14        上田          新規作成                                                                               */
/*****************************************************************************************************************************************/
unsigned int get_end_data_seq (unsigned char *read_data)
{
	int ret , timer_wk;
	unsigned char dummy_data ;

	ret = NORMAL_END ;
```

```
/*****************************************************************************************************************************************/
/*    ICCR(IRIC)=1 を確認                                                                                                                */
/*****************************************************************************************************************************************/
com_timer.wait_100ms_scan = 50 ;
while (IIC.ICCR.BIT.IRIC == 0){                                          /* 転送準備完了になるまで待つ                                */
     timer_wk = com_timer.wait_100ms_scan ;
     if (timer_wk == 0){                                                 /* 5s 間 1 のままならエラーで抜ける                          */
          ret = I2C_IRIC_TOUT;                                           /* 異常終了(timeout)                                         */
          goto exit ;
     }

     #ifdef UT
          IIC.ICCR.BIT.IRIC = 1 ;
     #endif
}

/*****************************************************************************************************************************************/
/*    データ受信後の ACK 返送時の設定値を"1"(NOACK)とする                                                                               */
/*****************************************************************************************************************************************/
IIC.ICSR.BIT.ACKB = 1 ;

/*****************************************************************************************************************************************/
/*    受信モードから送信モードに切り替える                                                                                               */
/*****************************************************************************************************************************************/
IIC.ICCR.BYTE = 0xB1 ;

/*****************************************************************************************************************************************/
/* ICCR(IRIC)クリア(wait 状態を解除するため)                                                                                             */
/*****************************************************************************************************************************************/
IIC.ICCR.BIT.IRIC = 0 ;

/*****************************************************************************************************************************************/
/*    ICCR(IRIC)=1 を確認                                                                                                                */
/*****************************************************************************************************************************************/
com_timer.wait_100ms_scan = 50 ;
while (IIC.ICCR.BIT.IRIC == 0){                                          /* 転送準備完了になるまで待つ                                */
     timer_wk = com_timer.wait_100ms_scan ;
     if (timer_wk == 0){                                                 /* 5s 間 1 のままならエラーで抜ける                          */
          ret = I2C_IRIC_TOUT;                                           /* 異常終了(timeout)                                         */
          goto exit ;
     }

     #ifdef UT
          IIC.ICCR.BIT.IRIC = 1 ;
     #endif
}

/*****************************************************************************************************************************************/
/*    ICMR(WAIT)をリセットする                                                                                                           */
/*****************************************************************************************************************************************/
IIC.ICMR.BIT.WAIT = 0 ;

/*****************************************************************************************************************************************/
/*    割込み禁止                                                                                                                         */
/*****************************************************************************************************************************************/
set_imask_ccr(1);                                                        /* 割込み禁止                                                */
```

```
      /*****************************************************************************************************************************************/
      /*    受信データを読み出す                                                                                                               */
      /*****************************************************************************************************************************************/
      *read_data = IIC.ICDR ;                                                  /* data read                                                  */

      /*****************************************************************************************************************************************/
      /*    ICCR(IRIC)クリア(wait 状態を解除するため)                                                                                          */
      /*****************************************************************************************************************************************/
      IIC.ICCR.BIT.IRIC = 0 ;
                /* ##(program note)################################################################################################ */
                /* ## ICDR のライトと IRIC のクリアは 1byte 分のデータ転送時間が経過しない間に、連続的に write すること。                  ## */
                /* ## このため、この操作の間は全ての割込みを禁止状態にする。                                                          ## */
                /* ################################################################################################################ */

      /*****************************************************************************************************************************************/
      /*    割込み禁止の解除                                                                                                                   */
      /*****************************************************************************************************************************************/
      set_imask_ccr(0);                                                        /* 割込み禁止解除                                             */

      /*****************************************************************************************************************************************/
      /*    ICCR(IRIC)クリア,停止条件発行                                                                                                      */
      /*****************************************************************************************************************************************/
      IIC.ICCR.BYTE = 0xB0 ;

exit :

      return (ret);
}

/*****************************************************************************************************************************************/
/*    1.モジュール名称:com_i2c_eeprom_write                                                                                                  */
/*    2.機能概要:I2CEEPROM に 1byte データを書きこむ                                                                                         */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                               */
/*          000  2002.12.14        上田          新規作成                                                                                   */
/*****************************************************************************************************************************************/
unsigned int com_i2c_eeprom_write ( unsigned char slave_addr , unsigned int rom_addr , unsigned char rom_data )
{
      int ret ;
      union {
            unsigned int  d_int ;
            unsigned char      d_byte[2];
      } buf;


      ret = NORMAL_END ;

      /*****************************************************************************************************************************************/
      /*    I2C BUS の初期化を行う                                                                                                             */
      /*****************************************************************************************************************************************/
      set_i2c_init () ;

      /*****************************************************************************************************************************************/
      /*    開始条件を設定する                                                                                                                 */
      /*****************************************************************************************************************************************/
      ret = set_start_condition() ;                                            /* 開始条件を設定する                                         */
            if (ret !=0) { goto exit ;}
```

```
    /*****************************************************************************************************************************************/
    /*    デバイスアドレスワードを設定する(Write)                                                                                            */
    /*****************************************************************************************************************************************/
    ret = set_slavesel_seq ( CMD_WRITE_OPERATION , slave_addr ) ;
        if (ret !=0) { goto exit ;}

    /*****************************************************************************************************************************************/
    /*    Acknowlege を待つ                                                                                                                  */
    /*****************************************************************************************************************************************/
    ret = wait_ack() ;
        if (ret !=0) { goto exit ;}

    /*****************************************************************************************************************************************/
    /*    Memory address を設定する                                                                                                          */
    /*****************************************************************************************************************************************/
    buf.d_int = rom_addr ;

    ret = set_data_seq ( buf.d_byte[0]) ;                                   /* 1st Memory address                        */
        if (ret !=0) { goto exit ;}

    ret = set_data_seq ( buf.d_byte[1]) ;                                   /* 2nd Memory address                        */
        if (ret !=0) { goto exit ;}

    /*****************************************************************************************************************************************/
    /*    write data を設定する                                                                                                              */
    /*****************************************************************************************************************************************/
    ret = set_data_seq ( rom_data) ;
        if (ret !=0) { goto exit ;}

    /*****************************************************************************************************************************************/
    /*    停止条件を発行する                                                                                                                 */
    /*****************************************************************************************************************************************/
    ret = set_end_proc ( ) ;
        if (ret !=0) { goto exit ;}

    /*****************************************************************************************************************************************/
    /*    書き込み終了をチェックする                                                                                                         */
    /*****************************************************************************************************************************************/
    ret = wait_write_end ( slave_addr ) ;
        if (ret !=0) { goto exit ;}
        /* ##(program note)#################################################################################################################### */
        /* ## I2CEEPROM は停止条件を受信後、書き込み動作を開始する。書き込み動作には、最大 15ms と時間がかかるため                          ## */
        /* ## Acknowledge Pooling という方法で書き込み終了をチェックする                                                                  ## */
        /* ##################################################################################################################################### */

    return (ret);

exit :
    /*****************************************************************************************************************************************/
    /*    エラーの時、リセット、停止条件を発行する                                                                                           */
    /*****************************************************************************************************************************************/
    set_end_proc ( ) ;
        return (ret);

}
```

```
/*****************************************************************************************************************************************************/
/*    1.モジュール名称:com_i2c_eeprom_write                                                                                                          */
/*    2.機能概要:I2CEEPROM に連続してデータを書きこむ                                                                                                */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                                       */
/*          000   2002.12.14        上田           新規作成                                                                                       */
/*****************************************************************************************************************************************************/
unsigned int com_i2c_eeprom_page_write
               ( unsigned char slave_addr , unsigned int rom_addr , unsigned int rom_length , unsigned char *rom_data )
{
     int ret , i ;
     union {
           unsigned int  d_int ;
           unsigned char      d_byte[2];
     } buf;


     ret = NORMAL_END ;

     /*****************************************************************************************************************************************************/
     /*   I2C BUS の初期化を行う                                                                                                                       */
     /*****************************************************************************************************************************************************/
     set_i2c_init () ;

     /*****************************************************************************************************************************************************/
     /*    開始条件を設定する                                                                                                                          */
     /*****************************************************************************************************************************************************/
     ret = set_start_condition() ;                                          /* 開始条件を設定する                                                    */
          if (ret !=0) { goto exit ;}

     /*****************************************************************************************************************************************************/
     /*    デバイスアドレスワードを設定する(Write)                                                                                                     */
     /*****************************************************************************************************************************************************/
     ret = set_slavesel_seq ( CMD_WRITE_OPERATION , slave_addr ) ;
          if (ret !=0) { goto exit ;}

     /*****************************************************************************************************************************************************/
     /*   Acknowlege を待つ                                                                                                                            */
     /*****************************************************************************************************************************************************/
     ret = wait_ack() ;
          if (ret !=0) { goto exit ;}

     /*****************************************************************************************************************************************************/
     /*   Memory address を設定する                                                                                                                    */
     /*****************************************************************************************************************************************************/
     buf.d_int = rom_addr ;

     ret = set_data_seq ( buf.d_byte[0]) ;                                  /* 1st Memory address                                                    */
          if (ret !=0) { goto exit ;}

     ret = set_data_seq ( buf.d_byte[1]) ;                                  /* 2nd Memory address                                                    */
          if (ret !=0) { goto exit ;}
```

```
	/*****************************************************************************************************************************************/
	/*	データを連続書き込みする                                                                                                             */
	/*****************************************************************************************************************************************/
	for (i=0; i< rom_length ; i++){
		buf.d_byte[0] = *rom_data  ;
		ret = set_data_seq ( buf.d_byte[0] ) ;
			if (ret !=0) { goto exit ;}
			*rom_data ++ ;
	}

	/*****************************************************************************************************************************************/
	/*	停止条件を発行する                                                                                                                   */
	/*****************************************************************************************************************************************/
	ret = set_end_proc ( ) ;
		if (ret !=0) { goto exit ;}

	/*****************************************************************************************************************************************/
	/*	書き込み終了をチェックする                                                                                                           */
	/*****************************************************************************************************************************************/
	ret = wait_write_end ( slave_addr ) ;
		if (ret !=0) { goto exit ;}
		/* ##(program note)################################################################################################################ */
		/* ## I2CEEPROM は停止条件を受信後、書き込み動作を開始する。書き込み動作には、最大 15ms と時間がかかるため                          ## */
		/* ## Acknowledge Pooling という方法で書き込み終了をチェックする                                                               ## */
		/* ################################################################################################################################ */

	return (ret);

exit :
	/*****************************************************************************************************************************************/
	/*	エラーの時、リセット、停止条件を発行する                                                                                             */
	/*****************************************************************************************************************************************/
	set_end_proc ( ) ;
		return (ret);


}

/*****************************************************************************************************************************************/
/*	1.モジュール名称:com_i2c_eeprom_read                                                                                                 */
/*	2.機能概要:I2CEEPROM から 1byte データを読み出す                                                                                     */
/*	3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                             */
/*	        000   2002.12.14        上田          新規作成                                                                               */
/*****************************************************************************************************************************************/
unsigned int com_i2c_eeprom_read ( unsigned char slave_addr , unsigned int rom_addr , unsigned char *rom_data )
{
	int ret ;
	union {
		unsigned int  d_int ;
		unsigned char      d_byte[2];
	} buf;


	ret = NORMAL_END ;
```

```
/*****************************************************************************************************************************************/
/*    I2C BUS の初期化を行う                                                                                                             */
/*****************************************************************************************************************************************/
set_i2c_init () ;

/*****************************************************************************************************************************************/
/*    開始条件を設定する                                                                                                                 */
/*****************************************************************************************************************************************/
ret = set_start_condition() ;                                          /* 開始条件を設定する                                          */
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    デバイスアドレスワードを設定する(Write)                                                                                            */
/*****************************************************************************************************************************************/
ret = set_slavesel_seq ( CMD_WRITE_OPERATION , slave_addr ) ;
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    Acknowlege を待つ                                                                                                                  */
/*****************************************************************************************************************************************/
ret = wait_ack() ;
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    Memory address を設定する                                                                                                          */
/*****************************************************************************************************************************************/
buf.d_int = rom_addr ;

ret = set_data_seq ( buf.d_byte[0]) ;                                  /* 1st Memory address                                          */
      if (ret !=0) { goto exit ;}

ret = set_data_seq ( buf.d_byte[1]) ;                                  /* 2nd Memory address                                          */
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    停止条件を発行することで一旦 SDA を High にする                                                                                    */
/*****************************************************************************************************************************************/
ret = set_end_proc ( ) ;
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    再び開始条件を設定する                                                                                                             */
/*****************************************************************************************************************************************/
ret = set_start_condition() ;                                          /* 開始条件を設定する                                          */
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    デバイスアドレスワードを設定する(read)                                                                                             */
/*****************************************************************************************************************************************/
ret = set_slavesel_seq ( DATA_READ_OPERATION , slave_addr ) ;
      if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    Acknowlege を待つ                                                                                                                  */
/*****************************************************************************************************************************************/
ret = wait_ack() ;
      if (ret !=0) { goto exit ;}
```

```
     /*****************************************************************************************************************************************/
     /*    マスタ受信モードに切り替える                                                                                                       */
     /*****************************************************************************************************************************************/
     ret = set_master_rcv_mode () ;
          if (ret !=0) { goto exit ;}

     /*****************************************************************************************************************************************/
     /*    データ読み出し開始の dummy read を行う                                                                                             */
     /*****************************************************************************************************************************************/
     start_read_seq () ;

     /*****************************************************************************************************************************************/
     /*    データ(1byte)を読み出し後、停止条件を発行する                                                                                      */
     /*****************************************************************************************************************************************/
     ret = get_end_data_seq ( &buf.d_byte[0] ) ;
          if (ret !=0) { goto exit ;}

          *rom_data = buf.d_byte[0] ;

          return (ret);

exit :
     /*****************************************************************************************************************************************/
     /*    エラーの時、リセット、停止条件を発行する                                                                                           */
     /*****************************************************************************************************************************************/
     set_end_proc ( ) ;
          return (ret);

}
/*************************************************************************************************************************************************/
/*    1.モジュール名称:com_i2c_eeprom_seq_read                                                                                                   */
/*    2.機能概要:I2CEEPROM から指定長分データを読み出す                                                                                          */
/*    3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                                                   */
/*          000   2002.12.14        上田          新規作成                                                                                     */
/*************************************************************************************************************************************************/
unsigned int com_i2c_eeprom_seq_read
               ( unsigned char slave_addr , unsigned int rom_addr , unsigned int rom_length , unsigned char *rom_data )
{
     int ret , i  ;
     union {
           unsigned int  d_int ;
           unsigned char      d_byte[2];
     } buf;


     ret = NORMAL_END ;

     /*****************************************************************************************************************************************/
     /*    I2C BUS の初期化を行う                                                                                                             */
     /*****************************************************************************************************************************************/
     set_i2c_init () ;

     /*****************************************************************************************************************************************/
     /*    開始条件を設定する                                                                                                                 */
     /*****************************************************************************************************************************************/
     ret = set_start_condition() ;                                          /* 開始条件を設定する                                          */
          if (ret !=0) { goto exit ;}
```

```
/*****************************************************************************************************************************************/
/*    デバイスアドレスワードを設定する(Write)                                                                                            */
/*****************************************************************************************************************************************/
ret = set_slavesel_seq ( CMD_WRITE_OPERATION , slave_addr ) ;
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    Acknowlege を待つ                                                                                                                  */
/*****************************************************************************************************************************************/
ret = wait_ack() ;
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    Memory address を設定する                                                                                                          */
/*****************************************************************************************************************************************/
buf.d_int = rom_addr ;

ret = set_data_seq ( buf.d_byte[0]) ;                                  /* 1st Memory address                        */
     if (ret !=0) { goto exit ;}

ret = set_data_seq ( buf.d_byte[1]) ;                                  /* 2nd Memory address                        */
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    停止条件を発行することで一旦 SDA を High にする                                                                                    */
/*****************************************************************************************************************************************/
ret = set_end_proc ( ) ;
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    再び開始条件を設定する                                                                                                             */
/*****************************************************************************************************************************************/
ret = set_start_condition() ;                                          /* 開始条件を設定する                        */
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    デバイスアドレスワードを設定する(read)                                                                                             */
/*****************************************************************************************************************************************/
ret = set_slavesel_seq ( DATA_READ_OPERATION , slave_addr ) ;
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    Acknowlege を待つ                                                                                                                  */
/*****************************************************************************************************************************************/
ret = wait_ack() ;
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    マスタ受信モードに切り替える                                                                                                       */
/*****************************************************************************************************************************************/
ret = set_master_rcv_mode () ;
     if (ret !=0) { goto exit ;}

/*****************************************************************************************************************************************/
/*    データ読み出し開始の dummy read を行う                                                                                             */
/*****************************************************************************************************************************************/
start_read_seq () ;
```

```
    /*****************************************************************************************************************************************/
    /*    データを連続読み出しする                                                                                                           */
    /*****************************************************************************************************************************************/
    for (i=0; i< (rom_length-1) ; i++){
        ret = get_data_seq ( &buf.d_byte[0] ) ;
            if (ret !=0) { goto exit ;}

            *rom_data = buf.d_byte[0] ;
            *rom_data ++ ;
    }

    /*****************************************************************************************************************************************/
    /*    最後のデータ(1byte)を読み出し後、停止条件を発行する                                                                                */
    /*****************************************************************************************************************************************/
    ret = get_end_data_seq ( &buf.d_byte[0] ) ;
        if (ret !=0) { goto exit ;}

        *rom_data = buf.d_byte[0] ;

        return (ret);

exit :
    /*****************************************************************************************************************************************/
    /*    エラーの時、リセット、停止条件を発行する                                                                                           */
    /*****************************************************************************************************************************************/
    set_end_proc ( ) ;
        return (ret);
}
```

```
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 4 . サンプルプログラム 1-D TimerW 処理---------------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
/* ------------------------------------------------------------------------------------------------------------------------ */

/* ------------------------------------------------------------------------------------------------------------------------ */
/* 4.1 リセットベクターの追加---------------------------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
/*   飛び先を h8_timerw に設定してください                                                                                 */

/* ------------------------------------------------------------------------------------------------------------------------ */
/* 4.2 TimerW 用共通変数定義----------------------------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
    struct    {
         int counter;                                          /* 100ms カウンタ                          */
         int wait_10ms;                                        /* 10mswait 用                             */
         int wait_100ms;                                       /* 100ms 単位 wait 用(共通)                */
         int wait_100ms_scan;                                  /* 100ms 単位 wait 用(for I2C)             */
    }com_timer;

/* ------------------------------------------------------------------------------------------------------------------------ */
/* 4.3 TimerW の初期設定--------------------------------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
    /* ###################################################################################################################### */
    /* ###################################################################################################################### */
    /*                                                                                                                        */
    /*        TimerW の設定                                                                                                   */
    /*                                                                                                                        */
    /* ###################################################################################################################### */
    /* ###################################################################################################################### */
    /************************************************************************************************************************/
    /*   timerw を初期設定                                                                                                    */
    /************************************************************************************************************************/
    TW.TCRW.BIT.CKS     = 3 ;                                 /* 内部クロックΦ/8 でカウント              */
    TW.TCRW.BIT.CCLR    = 1 ;                                 /* GRA コンペアマッチでカウンタクリア       */
    TW.TIOR0.BIT.IOA    = 0 ;                                 /* GRA はアウトプットコンペア               */
    TW.TIERW.BIT.IMIEA  = 1 ;                                 /* IMFA イネーブル                         */
    TW.GRA              = 20000 ;                             /* 10msec 毎に割り込み                     */
         /* ##(program note)################################################################################################ */
         /* ## マイコンの動作周波数により設定値は異なる。設定内容は、H8/3687 のハードウェアマニュアルを参照のこと。        ## */
         /* ###################################################################################################################### */

    TW.TCNT             = 0 ;                                 /* タイマカウンタクリア                    */

    /************************************************************************************************************************/
    /*   割り込み禁止解除                                                                                                     */
    /************************************************************************************************************************/
    set_imask_ccr(0);                                         /* 割込み許可                              */

    /************************************************************************************************************************/
    /*   timerw を start させる                                                                                               */
    /*   「com_change_frq」実行前に timer を start させる事により、直接遷移割込みを使用せず                                 */
    /*   TimerW 割込みにより sleep 状態から復帰し、周波数変更を行っている。                                                   */
    /************************************************************************************************************************/
    TW.TMRW.BIT.CTS     = 1 ;                                 /* imer start                              */
```

```
/* ------------------------------------------------------------------------------------------------------------------------ */
/* 4.4 TimerW 割込み処理---------------------------------------------------------------------------------------------------- */
/* ------------------------------------------------------------------------------------------------------------------------ */
/************************************************************************************************************************/
/*   1.モジュール名称:h8_timerw                                                                                             */
/*   2.機能概要:10msec 毎のインターバルタイマ処理                                                                           */
/*   3.来歴:REV  作成/改訂日付      作成/改訂者      改訂内容                                                               */
/*          000   2002.02.11        上田          新規作成                                                                 */
/************************************************************************************************************************/
#pragma interrupt( h8_timerw )
void h8_timerw( void )
{

     /************************************************************************************************************************/
     /*    要因クリア                                                                                                        */
     /************************************************************************************************************************/
     com_global.dummy = TW.TSRW.BYTE;                                        /* dummy read                                */

     TW.TSRW.BIT.IMFA = 0;                                                   /* IMFA clear                                */

     /************************************************************************************************************************/
     /*   10msec 単位で -1                                                                                                   */
     /************************************************************************************************************************/
     if( com_timer.wait_10ms>0 )
          com_timer.wait_10ms --;

     /************************************************************************************************************************/
     /*    カウントアップ                                                                                                    */
     /************************************************************************************************************************/
     com_timer.counter++;
     if( com_timer.counter >= 10 ){
          /************************************************************************************************************************/
          /*   100msec 単位で -1                                                                                               */
          /************************************************************************************************************************/
          if( com_timer.wait_100ms>0 )
               com_timer.wait_100ms --;
          if( com_timer.wait_100ms_scan>0 )
               com_timer.wait_100ms_scan --;

          com_timer.counter = 0;
     }

}
```

4. 参考文献

- H8/3664 グループ　ハードウェアマニュアル（ルネサス テクノロジ発行）
- HN58X2464FPIAG データシート（ルネサス テクノロジ発行）
- I²C バス使用法（Philips 発行）

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2003.09.24	—	初版発行

安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(http://www.renesas.com)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものですが万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任は負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気付きの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。