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。

H8/300H Tiny シリーズ

低電圧検出時の EEPROM バックアップ処理

要旨

内蔵低電圧検出回路を使用し，低電圧時に RAM のデータを EEPROM へバックアップします。

動作確認デバイス

H8/3687G

目次

1. 仕様

1. 内蔵低電圧検出回路を使用し，動作状態を変化します。
2. アクティブモード時，外部 EEPROM から読み込んだ点滅間隔データにより LED を点滅します。
3. IRQ1 スイッチにより LED 点滅間隔を変更し，点滅間隔データを RAM に格納します。
4. アクティブモード時に電圧が 3.7V 以下になると，RAM 上の点滅間隔データを外部 EEPROM に書き込み，スタンバイモードに遷移します。
5. スタンバイモード時に電圧が 4.0V 以上になると，アクティブモードに復帰します。
6. 電圧が 2.3V 以下になると内部リセット信号が発生します。
7. 本タスクの接続例を図 1.1 に示します。

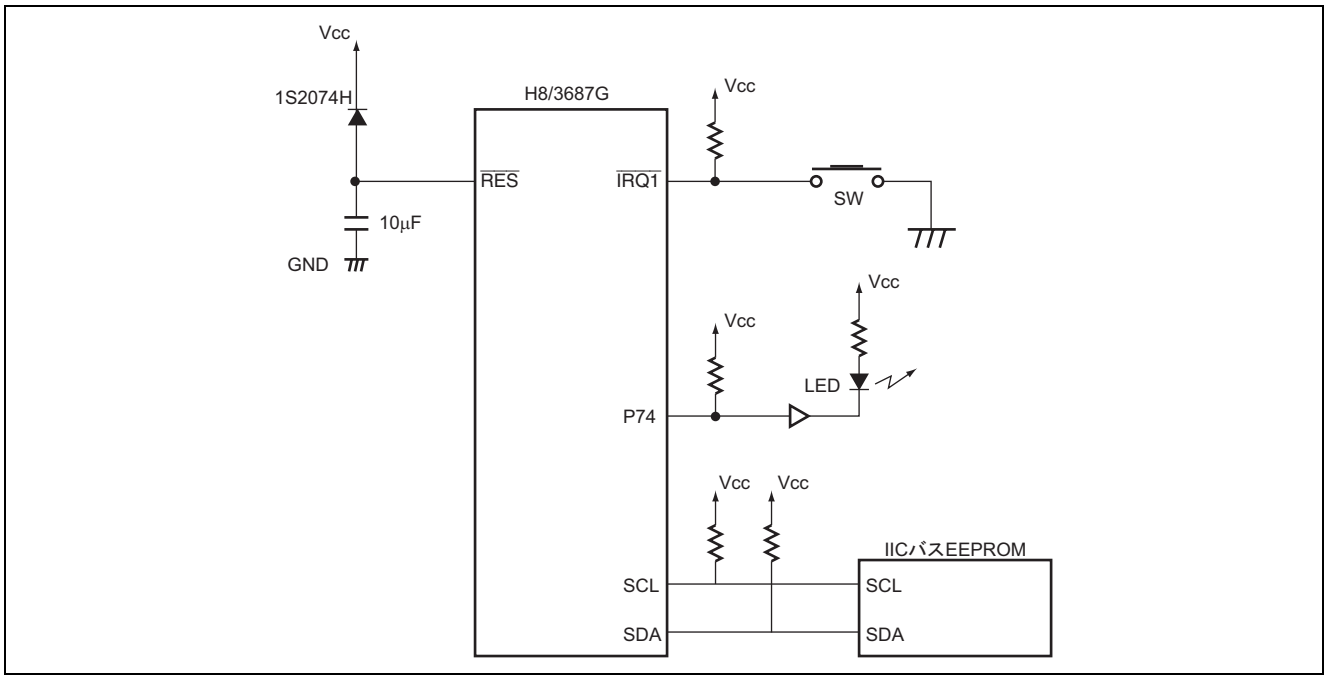


図 1.1　本タスクの接続例

2. 使用機能説明

1. 本タスク例では，オプションの内蔵低電圧検出回路を使用して，低電圧時の動作状態を制御します。低電圧検出回路のブロック図を図 2.1 に示します。以下に低電圧検出回路のブロック図について説明します。

- システムクロック (φ)
  16MHz のクロックで，CPU および周辺機能を動作させるための基準クロックです。
- プリスケーラ S (PSS)
  φを入力とする 13 ビットのカウンタで，1 サイクルごとにカウントアップします。
- 低電圧検出コントロールレジスタ (LVDCR)
  低電圧検出回路を制御します。本タスク例では，低電圧検出回路を使用し，電圧降下／上昇時に IRQ0 割り込みを発生，リセット検知電圧を 2.3V に設定しています。
- 低電圧検出ステータスレジスタ (LVDSR)
  電源電圧がある一定電圧より降下または上昇したかを示すフラグです。

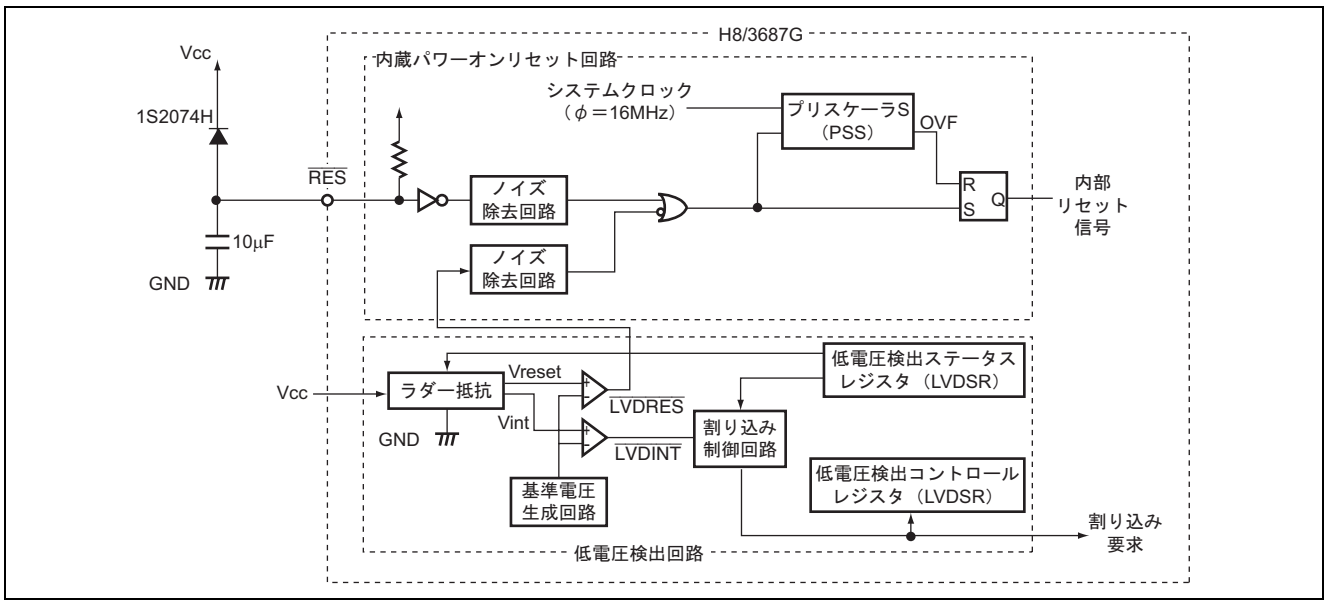


図 2.1　低電圧検出回路のブロック図

IIC バスインタフェース基本フォーマット (EEPROM バイトライト) を図 2.2 に示します。

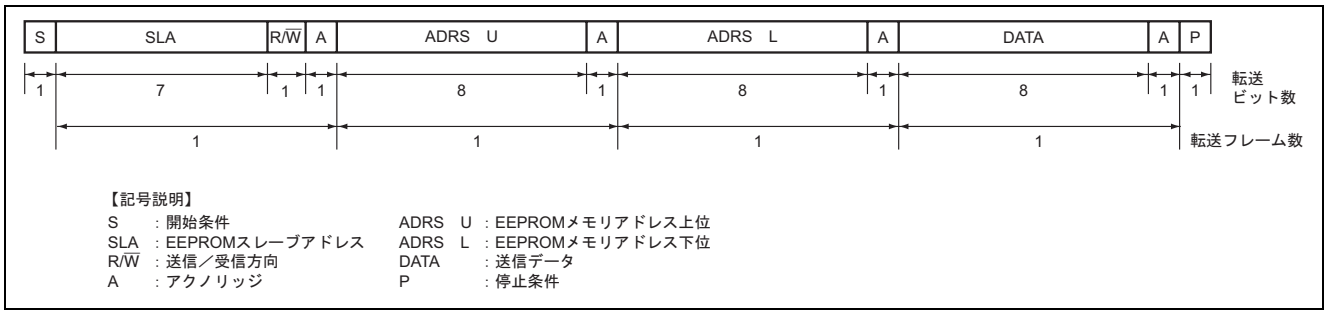


図 2.2　IIC バスインタフェースフォーマット

IIC バス EEPROM との接続例を図 2.3 に示します。

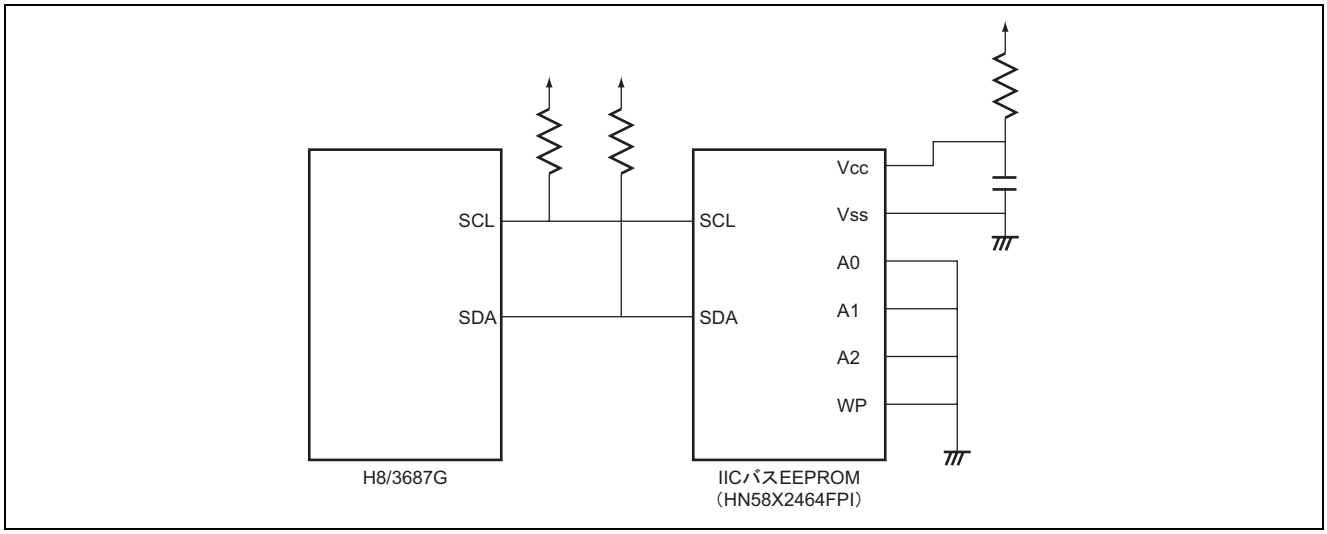


図 2.3　IIC バス EEPROM との接続例

2. 本タスク例の機能割り付けを表 2.1 に示します。表 2.1 に示すように機能を割り付け，低電圧検出時の EEPROM バックアップ処理を行います。

表 2.1　機能割り付け

機能	機能割り付け
PSS	システムクロックを入力とする 13 ビットのカウンタ
LVDCR	低電圧検出回路の動作／解除を制御
LVDSR	電源電圧がある一定電圧より降下または上昇したかを示すフラグ
PDR7	動作モードを確認するため，P74 端子に接続した LED を点滅する
PCR7	P74 端子を出力端子に設定
SYSCR1	低消費電力モードの制御
SYSCR2	低消費電力モードの制御
IRQ1	LED 点滅間隔の変更スイッチ
SCL	EEPROM との接続端子
SDA	EEPROM との接続端子

3. 本タスク例で使用している IIC バス EEPROM の仕様について説明します。
IIC バス EEPROM は，2 線式シリアルインタフェースの EEPROM (電気的に書き換え可能な ROM) です。
本タスク例では，Renesas Technology 社製 64k ビット EEPROM (HN58X2464FPI) を使用しています。本タスク例で使用している EEPROM の特長を表 2.2 に示します。

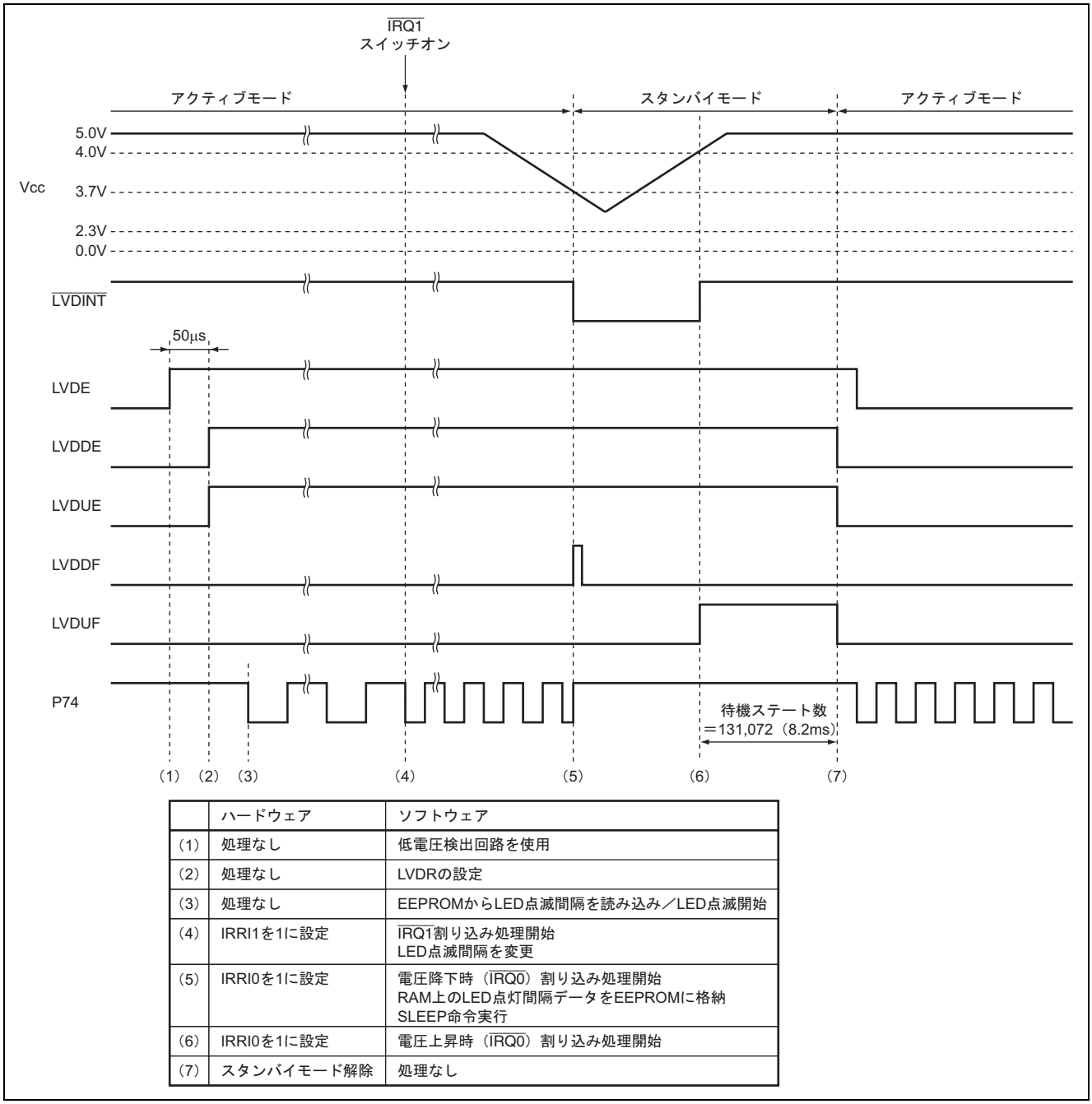
表 2.2　Renesas Technology 社製 64k ビット EEPROM (HN58X2464FPI)

単一電源		1.8V～5.5V
2 線式シリアルインタフェース		IIC バスインタフェース
動作周波数		400kHz
消費電流	スタンバイ時	3μA (max)
	読み出し時	1mA (max)
	書き換え時	3mA (max)
ページ書き換え		ページサイズ 32 バイト
書き換え時間		10ms (2.7～5.5V 以上)／15ms (1.8V～2.7V)
書き換え回数		10^5 回 (ページ書き換え時)

3. 動作説明

1. タイミングチャート

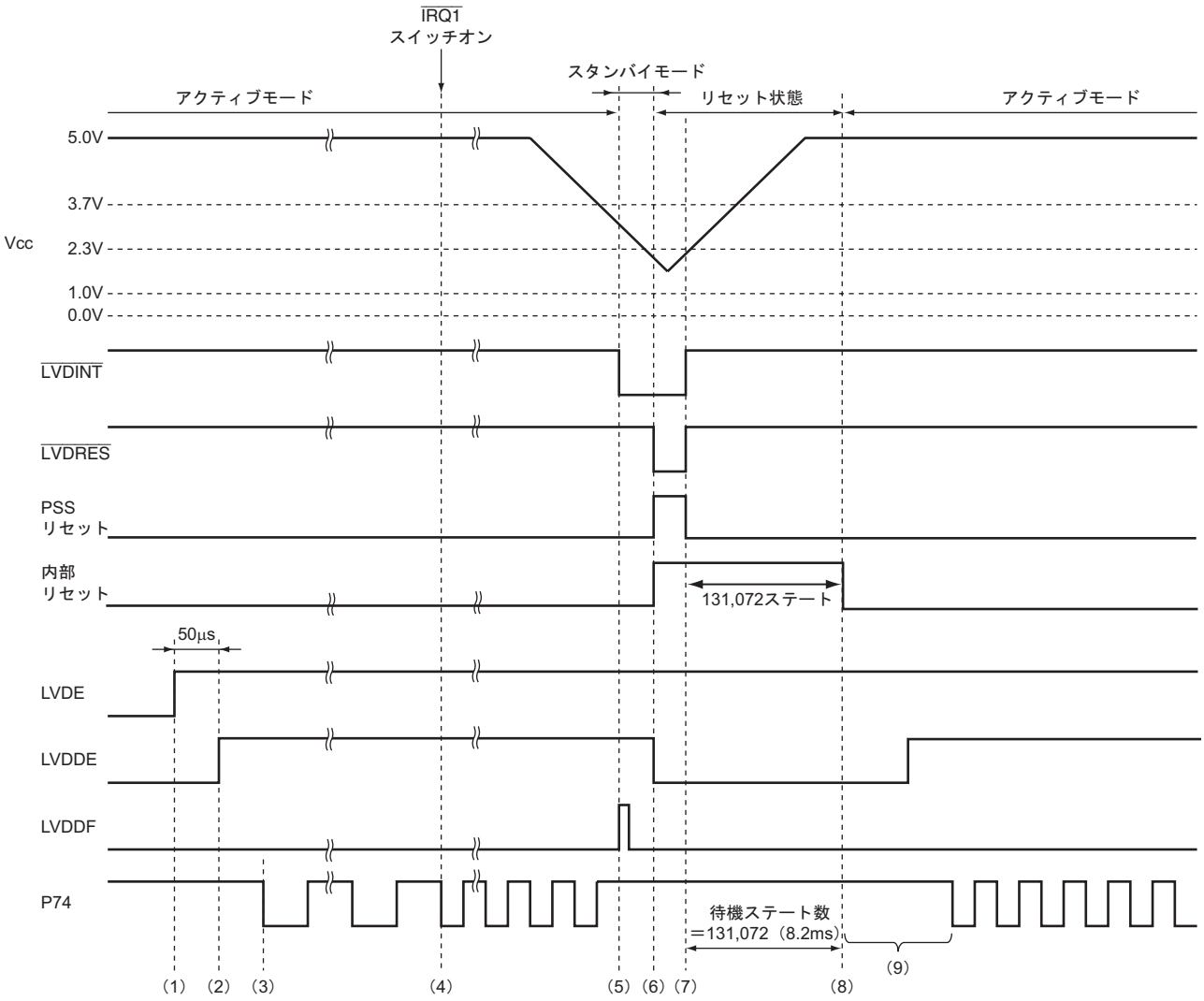
LVDI の設定／解除手順と，低電圧検出時割り込みをトリガとしたスタンバイモードへの遷移を図 3.1 に示します。



	ハードウェア	ソフトウェア
(1)	処理なし	低電圧検出回路を使用
(2)	処理なし	LVDRの設定
(3)	処理なし	EEPROMからLED点滅間隔を読み込み／LED点滅開始
(4)	IRRI1を1に設定	IRQ1割り込み処理開始 LED点滅間隔を変更
(5)	IRRI0を1に設定	電圧降下時（IRQ0）割り込み処理開始 RAM上のLED点灯間隔データをEEPROMに格納 SLEEP命令実行
(6)	IRRI0を1に設定	電圧上昇時（IRQ0）割り込み処理開始
(7)	スタンバイモード解除	処理なし

図 3.1　動作説明 (1)

低電圧検出時割り込みをトリガとしたスタンバイモードへの遷移と，低電圧検出時リセット動作を図 3.2 に示します。



	ハードウェア	ソフトウェア
(1)	処理なし	低電圧検出回路を使用
(2)	処理なし	LVDRの設定
(3)	処理なし	EEPROMからLED点滅間隔を読み込み／LED点滅開始
(4)	IRRI1を1に設定	IRQ1割り込み処理開始 LED点滅間隔を変更
(5)	IRRI0を1に設定	電圧降下時（IRQ0）割り込み処理開始 RAM上のLED点灯間隔データをEEPROMに格納 SLEEP命令実行
(6)	内部リセット発生	処理なし
(7)	PSSリセットカウント開始	処理なし
(8)	内部リセット解除	処理なし
(9)	処理なし	（1）、（2）、（3）を実行

図 3.2　動作説明 (2)

2. EEPROM 書き込み時間
   本タスク例では，低電圧時 RAM 上のデータ (4 バイト) を EEPROM にバックアップします。低電圧時割り込みの開始から EEPROM 書き込み終了までの処理時間を表 3.1 に示します。バックアップ処理が完了するまでの間，電源電圧が，動作保証下限電圧 (3.0V) 以上を保つようにしてください。

表 3.1　EEPROM バックアップ処理時間

書き込みデータサイズ	処理時間
4 バイト	330µs

4. ソフトウェア説明

4.1 モジュール説明

本タスク例のモジュールを表 4.1 に示します。

表 4.1 メインモジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	低電圧検出回路の設定，割り込みの許可，EEPROM データ読み込み，LED (P74) の制御，および IRQ0 に接続したスイッチの判定を行う
低電圧検出割り込み	irq0int	IRQ0 割り込み処理 LVD フラグクリア，lpcnt を 0 または 1 にセット，EEPROM バックアップ処理
スイッチオン	irq1int	IRQ1 割り込み処理 lpcnt を 2 にセット
EEPROM アクセスルーチン	Read_n_EEPROM	EEPROM から n バイト読み込み
	Write_n_EEPROM	EEPROM へ n バイト書き込み
	Write_data_EEPROM	EEPROM データ書き込み
	Write_data_End_EEPROM	EEPROM 最終データ書き込み
	Set_adrs_EEPROM	EEPROM アドレス指定
	Recv_datan_EEPROM	IIC を利用したデータ n バイト受信

4.2 引数の説明

各関数の引数を以下に示します。

- Read_n_EEPROM 関数

引数	機能	データ長
adrs	読み込みアドレス指定	2 バイト
*rd_ptr	読み込みデータ格納アドレス	1 バイト
no	読み込みデータ数	2 バイト

- Write_n_EEPROM 関数

引数	機能	データ長
adrs	書き込みアドレス指定	2 バイト
*wr_ptr	書き込みデータ格納アドレス	1 バイト
no	書き込みデータ数	2 バイト

- Write_data_EEPROM 関数

引数	機能	データ長
wr_data	書き込みデータ	1 バイト

- Write_data_End_EEPROM 関数

引数	機能	データ長
wr_data	書き込みデータ	1 バイト

- Set_adrs_EEPROM 関数

引数	機能	データ長
adrs	書き込み／読み込みアドレス	2 バイト

- Recv_datan_EEPROM 関数

引数	機能	データ長
*rd_ptr	読み込みデータ格納アドレス	1 バイト
no	読み込みバイト数 1～64	2 バイト

4.3 使用内部レジスタ説明

本タスク例の使用内部レジスタを以下に示します。

- LVDCR　低電圧検出コントロールレジスタ　　アドレス：0xF730

ビット	ビット名	設定値	機能
7	LVDE	1	LVD イネーブル LVDE＝0：低電圧検出回路は未使用（スタンバイ状態） LVDE＝1：低電圧検出回路を使用
3	LVDSEL	0	LVDR 検出レベル選択 LVDSEL＝0：リセット検知電圧を 2.3V に設定 LVDSEL＝1：リセット検知電圧を 3.6V に設定
2	LVDRE	1	LVDR イネーブル LVDRE＝0：LVDR によるリセットを禁止 LVDRE＝1：LVDR によるリセットを許可
1	LVDDE	1	電圧降下時の割り込みイネーブル LVDDE＝0：電圧降下時の割り込み要求を禁止 LVDDE＝1：電圧降下時の割り込み要求を許可
0	LVDUE	1	電圧上昇時の割り込みイネーブル LVDUE＝0：電圧上昇時の割り込み要求を禁止 LVDUE＝1：電圧上昇時の割り込み要求を許可

- LVDSR　低電圧検出ステータスレジスタ　　アドレス：0xF731

ビット	ビット名	設定値	機能
1	LVDDF	0	LVD 電源電圧降下フラグ LVDDF＝0：0 クリア状態 LVDDF＝1：電源電圧が 3.7V 以下に降下した
0	LVDUF	0	LVD 電源電圧上昇フラグ LVDUF＝0：0 クリア状態 LVDUF＝1：LVDCR の LVDUE を 1 にセットした状態で電源電圧が 3.7V 以下に降下し，Vreset1 (2.3V) 以下に降下する前に 4.0V 以上に再上昇した

- PDR7　ポートデータレジスタ 7　　アドレス：0xFFDA

ビット	ビット名	設定値	機能
4	P74	0	ポートデータレジスタ 74 P74＝0：P74 端子の出力レベルは Low P74＝1：P74 端子の出力レベルは High

- PMR1　ポートモードレジスタ 1　　アドレス：0xFFE0

ビット	ビット名	設定値	機能
5	IRQ1	1	P15/IRQ1 端子の機能を選択 IRQ1＝0：P15/IRQ1 端子を P15 入出力端子に設定 IRQ1＝1：P15/IRQ1 端子を IRQ1 入力端子に設定

- PCR7　ポートコントロールレジスタ 7　　アドレス：0xFFEA

ビット	ビット名	設定値	機能
4	PCR74	1	ポートコントロールレジスタ 74 PCR74＝0：P74 端子を P74 入力端子に設定 PCR74＝1：P74 端子を P74 出力端子に設定

- SYSCR1　システムコントロールレジスタ 1　　アドレス：0xFFF0

ビット	ビット名	設定値	機能
7	SSBY	1	ソフトウェアスタンバイ DTON＝0，SSBY＝1 ：アクティブモードで SLEEP 命令実行後，スタンバイモードに遷移
6 5 4	STS2 STS1 STS0	STS2＝1 STS1＝0 STS0＝0	スタンバイタイマセレクト 2～0 STS2＝1，STS1＝0，STS0＝0 のとき，待機ステート数を 131,072 ステートに設定

- SYSCR2　システムコントロールレジスタ 2　　アドレス：0xFFF1

ビット	ビット名	設定値	機能
5	DTON	0	ダイレクトトランスファオンフラグ DTON＝0，SSBY＝1 ：アクティブモードで SLEEP 命令実行後，スタンバイモードに遷移
4 3 2	MA2 MA1 MA0	MA2＝0 MA1＝× MA0＝×	アクティブモードクロックセレクト 2～0 MA2＝0，MA1＝×，MA0＝× ：アクティブモード／スリープモードの動作クロックをφosc に設定 (×：Don't care)

- IEGR1　割り込みエッジセレクトレジスタ 1　　アドレス：0xFFF2

ビット	ビット名	設定値	機能
0	IEG1	1	IRQ1 エッジセレクト IEG1＝0：IRQ1 端子入力の検出エッジに立ち下がりエッジを選択 IEG1＝1：IRQ1 端子入力の検出エッジに立ち上がりエッジを選択

- IENR1　割り込みイネーブルレジスタ 1　　アドレス：0xFFF4

ビット	ビット名	設定値	機能
1	IEN1	1	IRQ1 割り込み要求イネーブル IEN1＝0：IRQ1 端子の割り込み要求を禁止 IEN1＝1：IRQ1 端子の割り込み要求を許可

- IRR1　割り込みフラグレジスタ 1　　アドレス：0xFFF6

ビット	ビット名	設定値	機能
1	IRRI1	0	IRQ1 割り込み要求フラグ IRRI1＝0：IRQ1 端子の割り込みが要求されていない IRRI1＝1：IRQ1 端子の割り込みが要求されている
0	IRRI0	0	IRQ0 割り込み要求フラグ IRRI0＝0：IRQ0 端子の割り込みが要求されていない IRRI0＝1：IRQ0 端子の割り込みが要求されている

- ICCR1　IIC バスコントロールレジスタ 1　　アドレス：0xF748

ビット	ビット名	設定値	機能
7	ICE	1	IIC バスインタフェースイネーブル ICE＝0：IIC2 モジュールは機能停止状態 (SCL/SDA 端子はポート機能) ICE＝1：IIC2 モジュールは転送動作可能状態(SCL/SDA 端子はバス駆動機能)
6	RCVD	0	受信ディスエーブル RCVD＝0：次の受信動作を継続 RCVD＝1：次の受信動作を禁止
5	MST	0	マスタ／スレーブ選択 MST＝0：スレーブを選択 MST＝1：マスタを選択
4	TRS	0	送信／受信選択 TRS＝0：受信モード TRS＝1：送信モード
3 2 1 0	CKS3 CKS2 CKS1 CKS0	CKS3＝0 CKS2＝0 CKS1＝0 CKS0＝1	転送クロック選択 3～0 CKS3＝0，CKS2＝0，CKS1＝0，CKS0＝1 ：φ＝16 のとき，転送レートを 400kHz に設定

- ICCR2　IIC バスコントロールレジスタ 2　　アドレス：0xF749

ビット	ビット名	設定値	機能
7	BBSY	1	バスビジー BBSY＝0：IIC バス使用中 BBSY＝1：IIC バス未使用
6	SCP	0	開始／停止条件発行禁止ビット RCVD＝0：発行可能 RCVD＝1：発行禁止
5	SDAO	1	SDA 出力値制御 SDAO＝0：Low レベル SDAO＝1：High レベル
4	SDAOP	1	SDAO ライトプロテクト SDAOP＝0：書き込み有効 SDAOP＝1：書き込み無効
3	SCLO	1	SCL 出力レベルモニタ SCLO＝0：SCL 出力は Low レベル SCLO＝1：SCL 出力は High レベル
1	IICRST	0	IIC コントロール部リセット IICRST＝0：通常 IICRST＝1：リセット

- ICIER　IIC バスインタラプトイネーブルレジスタ　　アドレス：0xF74B

ビット	ビット名	設定値	機能
1	ACKBR	－	受信アクノリッジ ACKBR＝0：受信アクノリッジ＝0 ACKBR＝1：受信アクノリッジ＝1
0	ACKBT	0	送信アクノリッジ ACKBT＝0：送信アクノリッジ＝0 ACKBT＝1：送信アクノリッジ＝1

- ICSR　IIC バスステータスレジスタ　　アドレス：0xF74C

ビット	ビット名	設定値	機能
7	TDRE	－	トランスミットデータエンプティ TDRE＝0：0 クリア状態 TDRE＝1：ICDRT から ICDRS にデータが転送された
6	TEND	－	トランスミットエンド TEND＝0：0 クリア状態 TEND＝1：IIC バスフォーマットの場合，TDRE が 1 の状態で，SCL の 9 クロック目が立ち上がった
5	RDRF	－	レシーブデータレジスタフル RDRF＝0：0 クリア状態 RDRF＝1：ICDRS から ICDRR に受信データが転送された
3	STOP	0	停止条件検出フラグ STOP＝0：0 クリア状態 STOP＝1：フレーム転送の完了時に停止条件を検出した

- ICDRT　IIC バス送信データレジスタ　　アドレス：0xF74E

  機能：　送信データを格納するレジスタです。ICDRS の空きを検出すると ICDRT から ICDRS へ転送し，データ送信を開始します。

  設定値：－

- ICDRR　IIC バス受信データレジスタ　　アドレス：0xF74F

  機能：　受信データを格納するレジスタです。1 バイトのデータ受信が終了すると，ICDRS から ICDRR へ転送し，次のデータを受信可能にします。

  設定値：－

4.4　使用 RAM 説明

本タスク例の使用 RAM を表 4.2 に示します。

表 4.2　使用 RAM 説明

ラベル名	機能	メモリ消費量	使用モジュール名
lpcnt	低電圧検出状態を判定するフラグ lpcnt＝0：通常モード復帰 lpcnt＝1：低電圧時，RAM データを EEPROM にバックアップし，モジュールスタンバイ lpcnt＝2：IRQ1 割り込み，低電圧検出回路を解除	1 バイト	メインルーチン 低電圧検出割り込み スイッチオン

4.5 モジュール階層図

本タスク例のモジュール階層図を図 4.1 に示します。

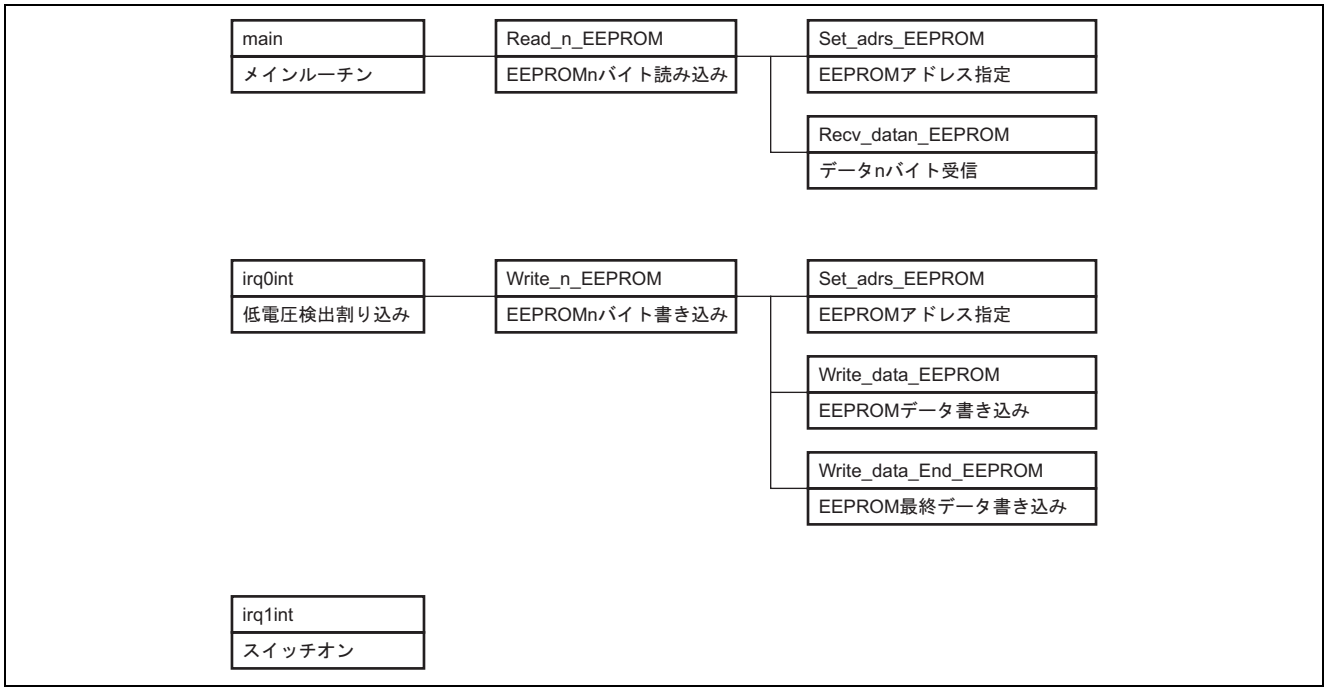
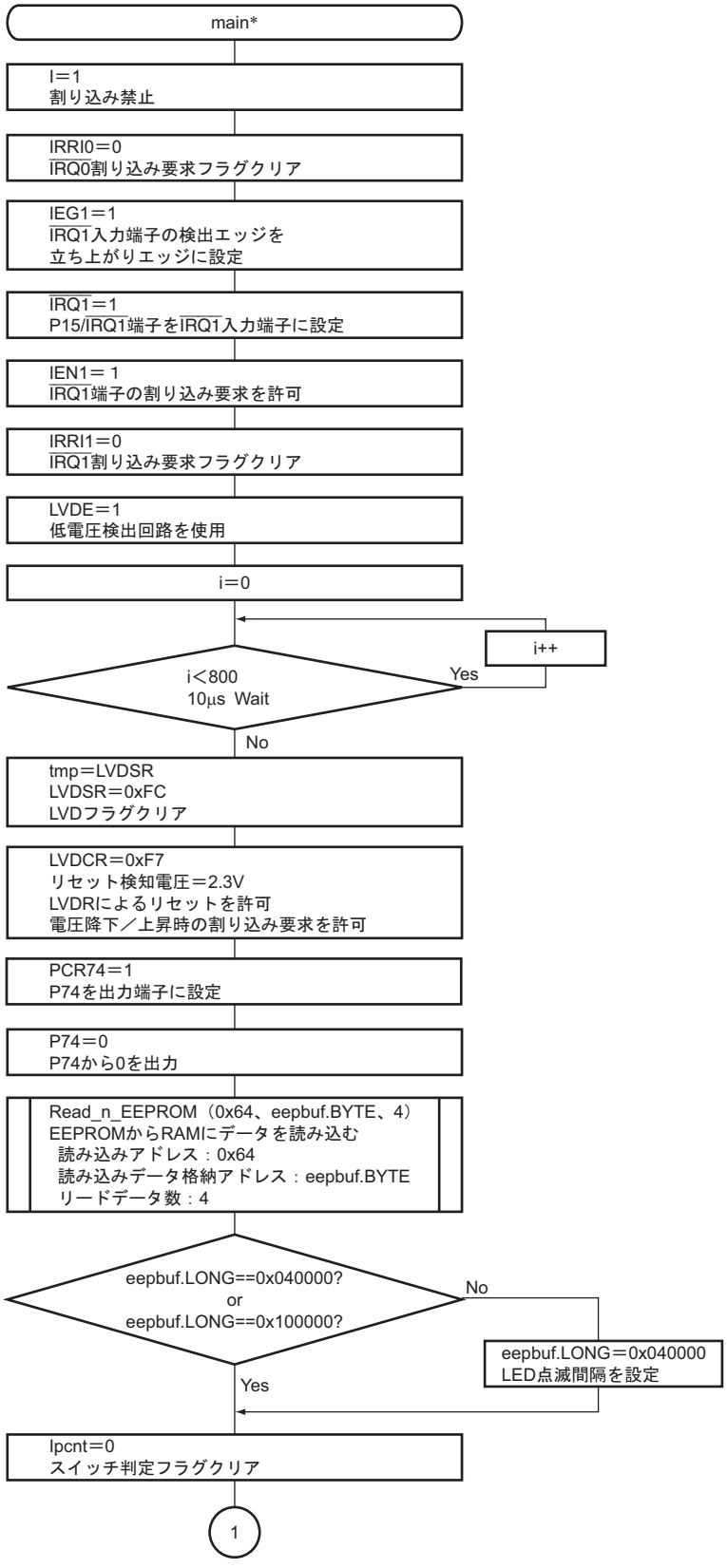


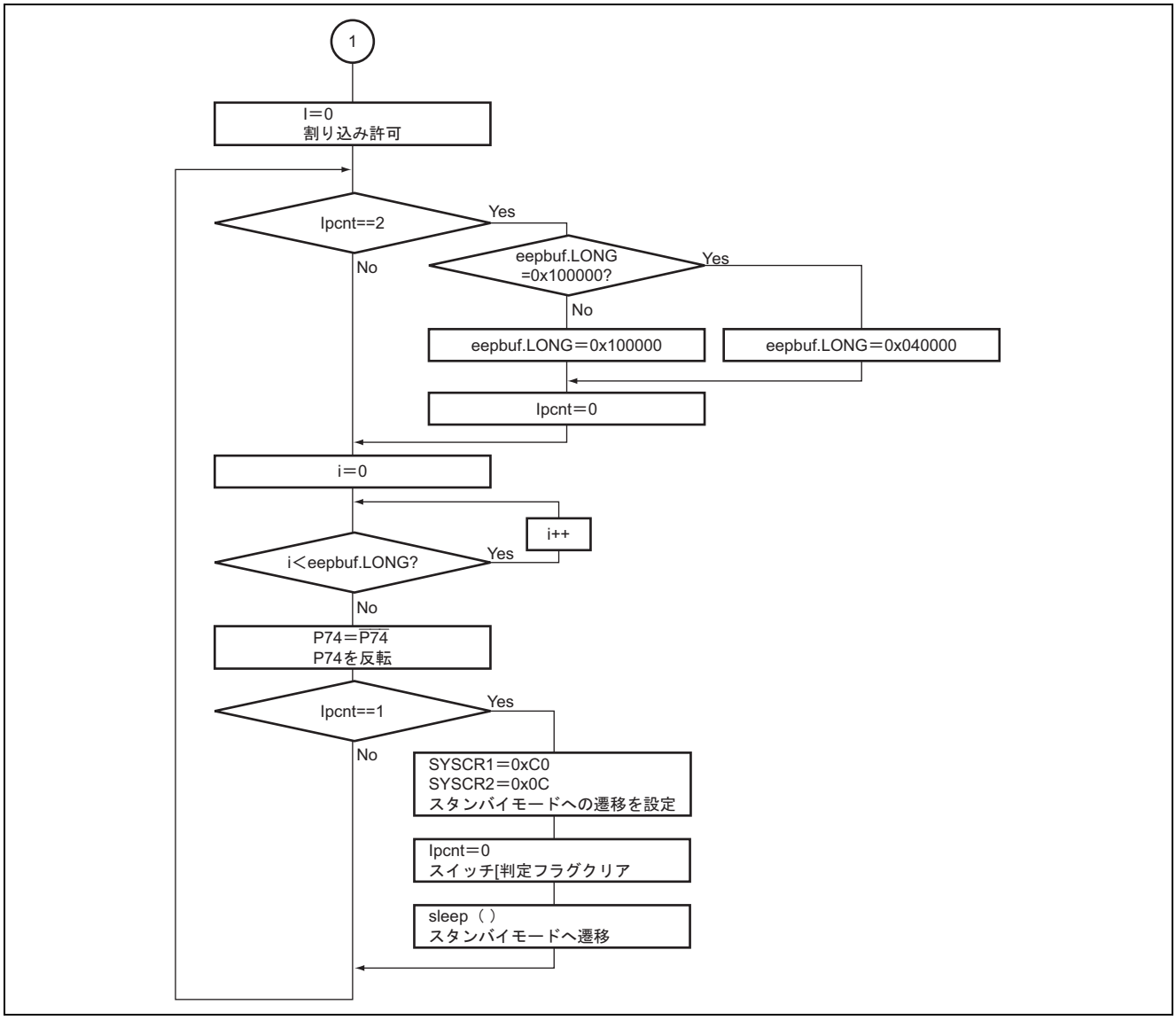
図 4.1 モジュール階層図

5. フローチャート

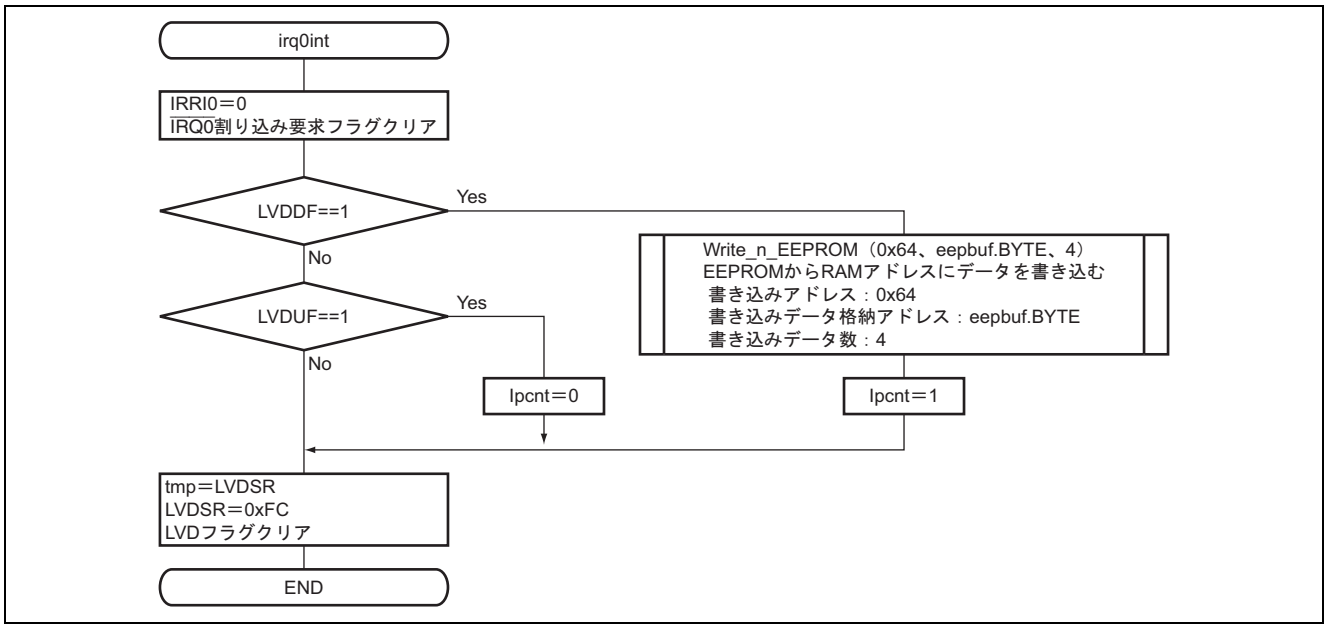
1. メインルーチン



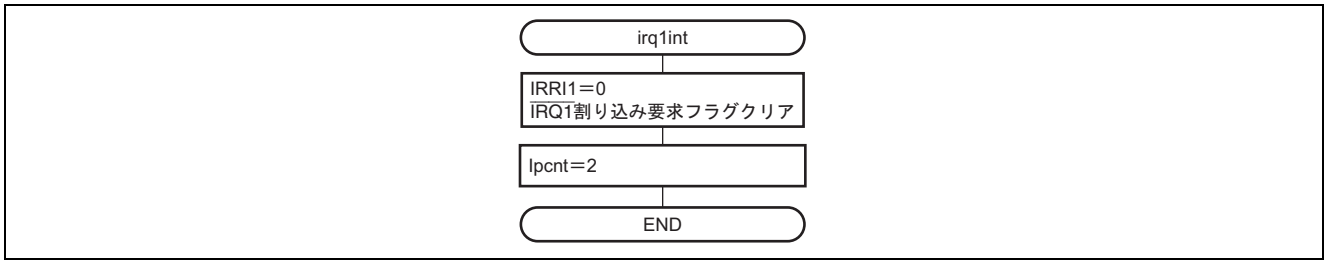
【注】* 本例ではスタックポインタはINIT.SRC（アセンブル言語）で設定してあります。


1
I=0
割り込み許可
lpcnt==2
Yes
No
eepbuf.LONG
=0x100000?
Yes
No
eepbuf.LONG=0x100000
eepbuf.LONG=0x040000
lpcnt=0
i=0
i++
i<eepbuf.LONG?
Yes
No
P74=P74
P74を反転
lpcnt==1
Yes
No
SYSCR1=0xC0
SYSCR2=0x0C
スタンバイモードへの遷移を設定
lpcnt=0
スイッチ[判定フラグクリア
sleep（）
スタンバイモードへ遷移

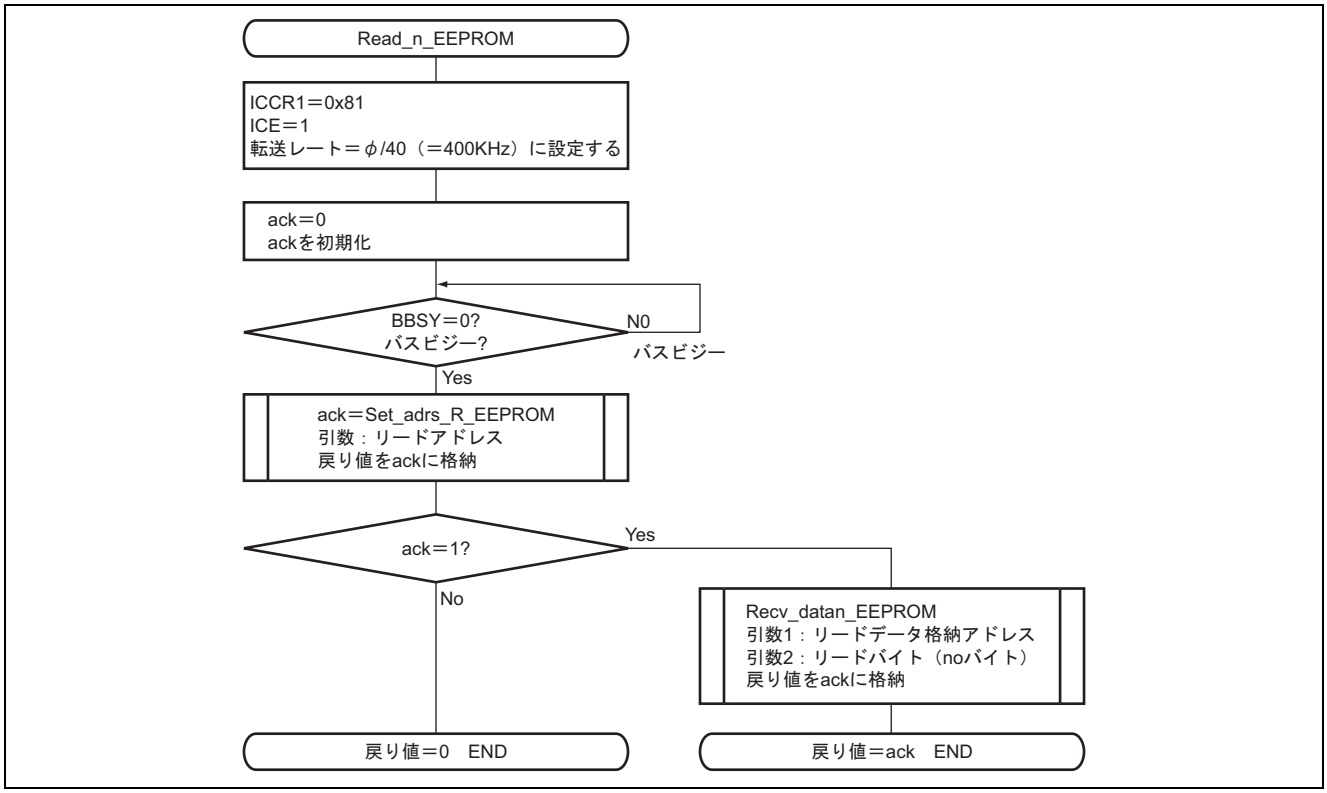
2. 低電圧検出割り込み

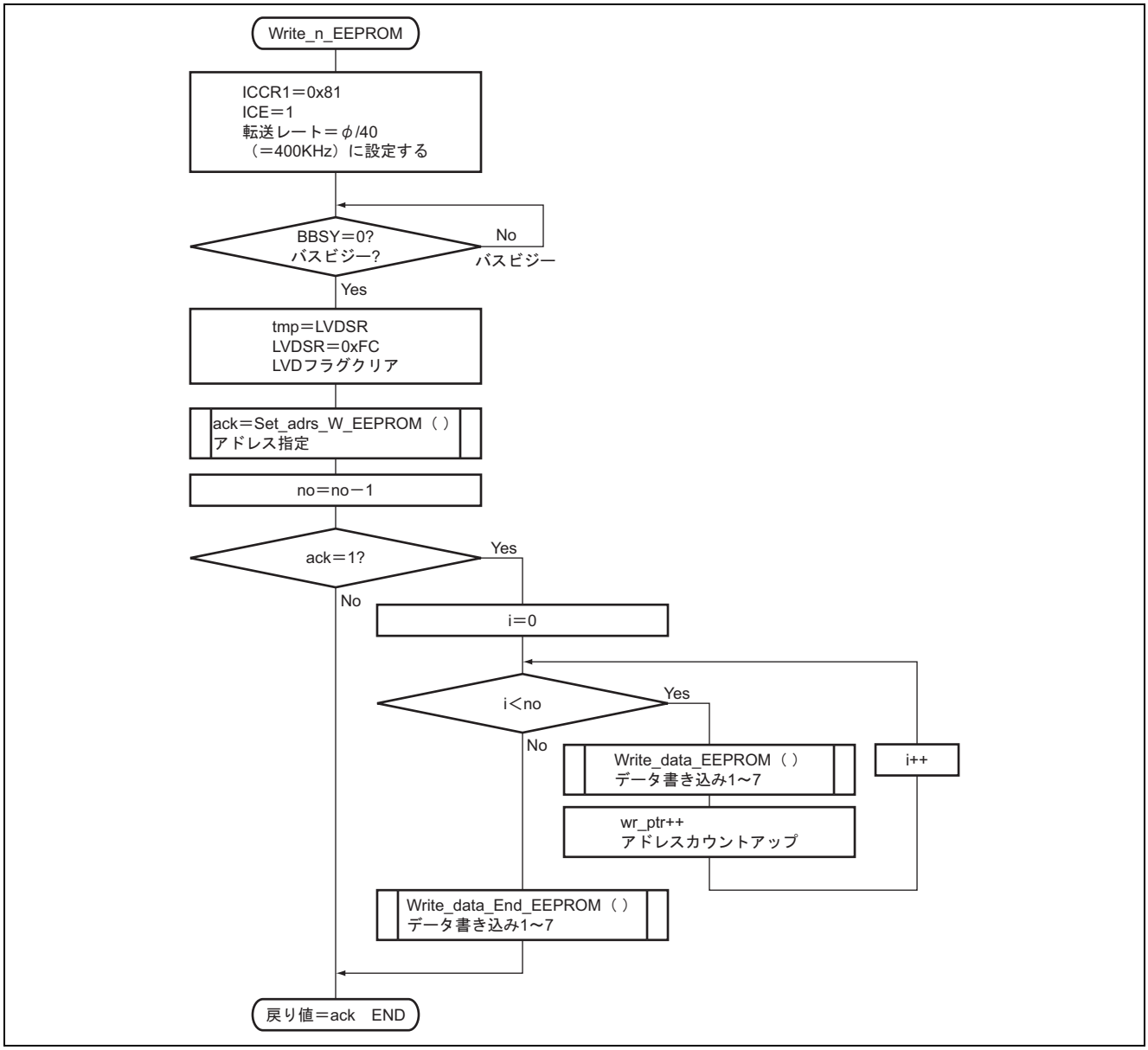

irq0int
IRRI0=0
IRQ0割り込み要求フラグクリア
LVDDF==1
Yes
No
Write_n_EEPROM（0x64、eepbuf.BYTE、4）
EEPROMからRAMアドレスにデータを書き込む
書き込みアドレス：0x64
書き込みデータ格納アドレス：eepbuf.BYTE
書き込みデータ数：4
LVDUF==1
Yes
No
lpcnt=0
lpcnt=1
tmp=LVDSR
LVDSR=0xFC
LVDフラグクリア
END

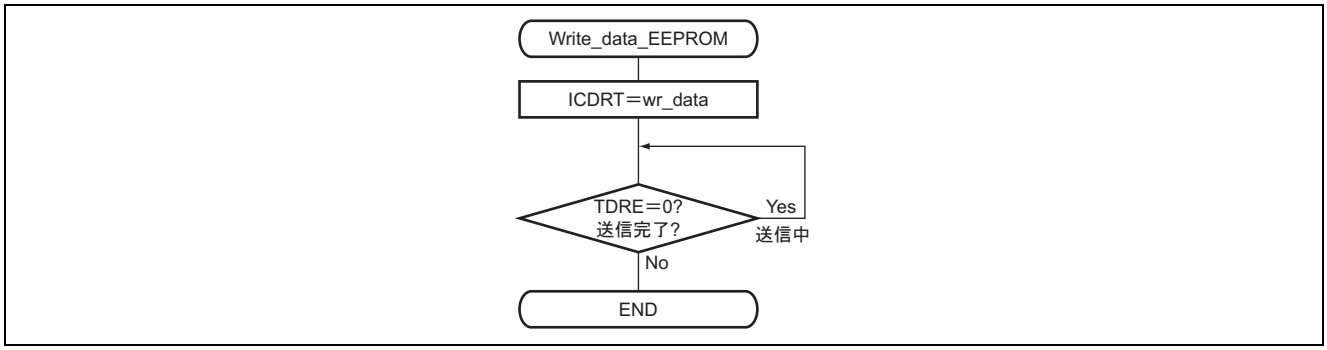
3. スイッチオン

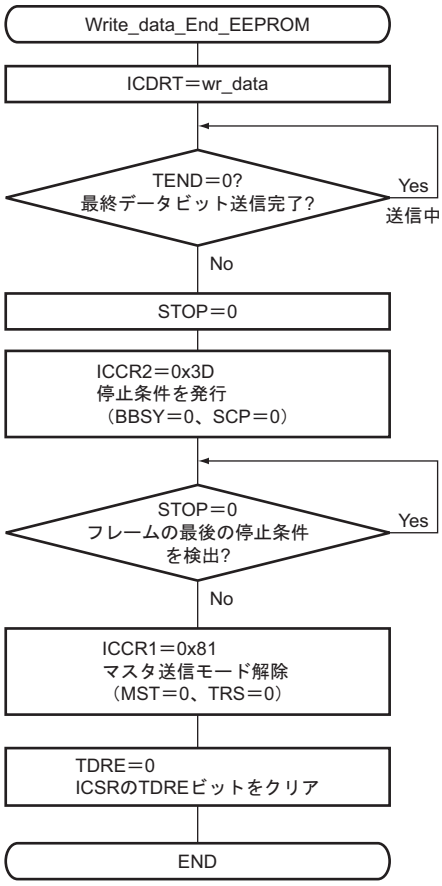


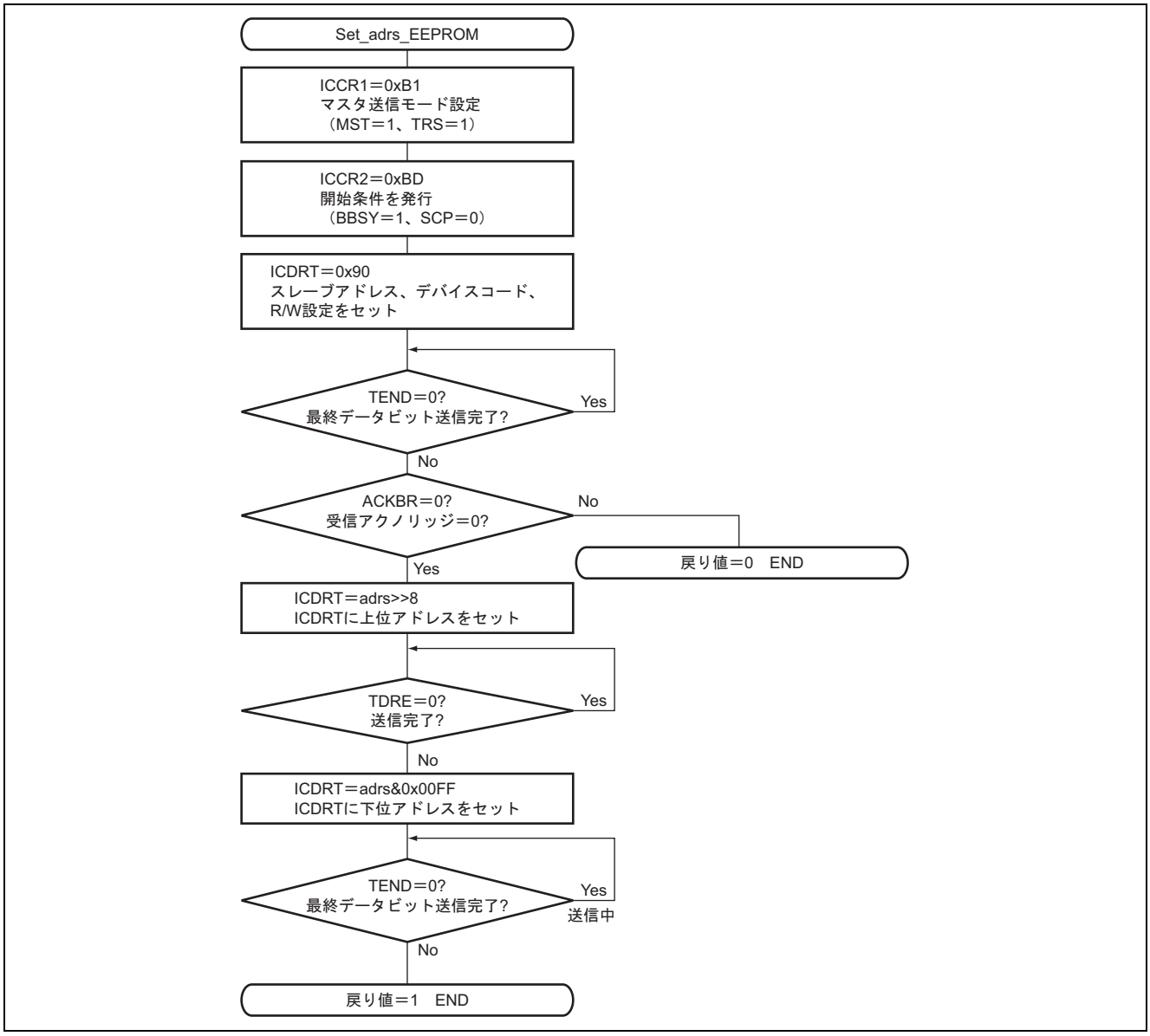
4. EEPROM アクセスルーチン

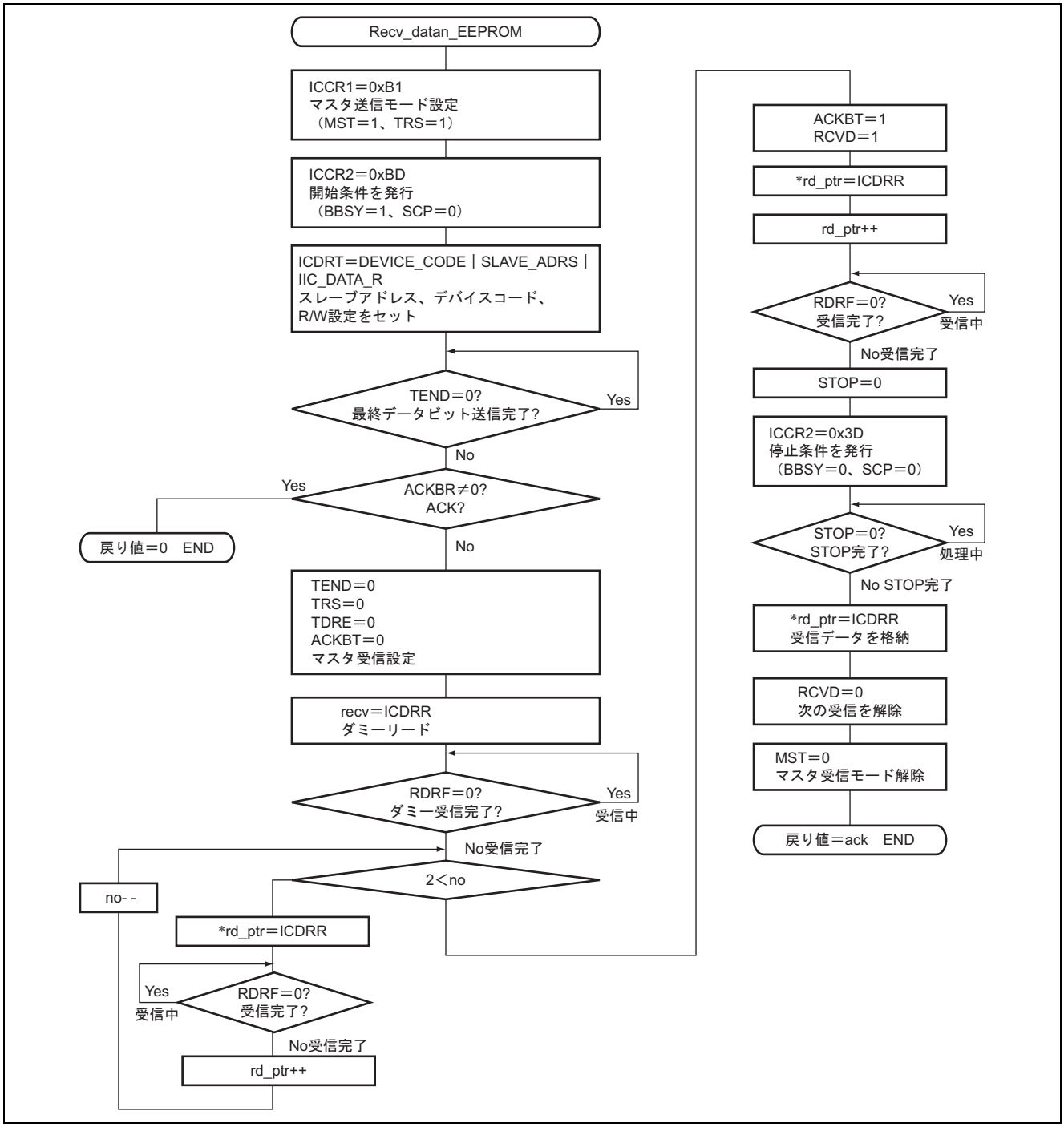



Write_n_EEPROM
ICCR1＝0x81
ICE＝1
転送レート＝φ/40
（＝400KHz）に設定する
BBSY＝0?
バスビジー?
No
バスビジー
Yes
tmp＝LVDSR
LVDSR＝0xFC
LVDフラグクリア
ack＝Set_adrs_W_EEPROM（）
アドレス指定
no＝no－1
ack＝1?
Yes
No
i＝0
i＜no
Yes
No
Write_data_EEPROM（）
データ書き込み1～7
i++
wr_ptr++
アドレスカウントアップ
Write_data_End_EEPROM（）
データ書き込み1～7
戻り値＝ack　END


Write_data_EEPROM
ICDRT＝wr_data
TDRE＝0?
送信完了?
Yes
送信中
No
END


Write_data_End_EEPROM
ICDRT＝wr_data
TEND＝0?
最終データビット送信完了?
Yes
送信中
No
STOP＝0
ICCR2＝0x3D
停止条件を発行
（BBSY＝0、SCP＝0）
STOP＝0
フレームの最後の停止条件
を検出?
Yes
No
ICCR1＝0x81
マスタ送信モード解除
（MST＝0、TRS＝0）
TDRE＝0
ICSRのTDREビットをクリア
END


Set_adrs_EEPROM
ICCR1＝0xB1
マスタ送信モード設定
（MST＝1、TRS＝1）
ICCR2＝0xBD
開始条件を発行
（BBSY＝1、SCP＝0）
ICDRT＝0x90
スレーブアドレス、デバイスコード、
R/W設定をセット
TEND＝0?
最終データビット送信完了?
Yes
No
ACKBR＝0?
受信アクノリッジ＝0?
No
戻り値＝0　END
Yes
ICDRT＝adrs>>8
ICDRTに上位アドレスをセット
TDRE＝0?
送信完了?
Yes
No
ICDRT＝adrs&0x00FF
ICDRTに下位アドレスをセット
TEND＝0?
最終データビット送信完了?
Yes
送信中
No
戻り値＝1　END


Recv_datan_EEPROM
ICCR1＝0xB1
マスタ送信モード設定
（MST＝1、TRS＝1）
ICCR2＝0xBD
開始条件を発行
（BBSY＝1、SCP＝0）
ICDRT＝DEVICE_CODE｜SLAVE_ADRS｜IIC_DATA_R
スレーブアドレス、デバイスコード、R/W設定をセット
TEND＝0?
最終データビット送信完了?
Yes
No
ACKBR≠0?
ACK?
Yes
戻り値＝0　END
No
TEND＝0
TRS＝0
TDRE＝0
ACKBT＝0
マスタ受信設定
recv＝ICDRR
ダミーリード
RDRF＝0?
ダミー受信完了?
Yes
受信中
No受信完了
2＜no
no- -
*rd_ptr＝ICDRR
Yes
受信中
RDRF＝0?
受信完了?
No受信完了
rd_ptr++
ACKBT＝1
RCVD＝1
*rd_ptr＝ICDRR
rd_ptr++
RDRF＝0?
受信完了?
Yes
受信中
No受信完了
STOP＝0
ICCR2＝0x3D
停止条件を発行
（BBSY＝0、SCP＝0）
STOP＝0?
STOP完了?
Yes
処理中
No STOP完了
*rd_ptr＝ICDRR
受信データを格納
RCVD＝0
次の受信を解除
MST＝0
マスタ受信モード解除
戻り値＝ack　END

6. プログラムリスト

```
/************************************************************************************************************************/
/*                                                                                                                      */
/*   H8/300HN Series -H8/3687G-                                                                                         */
/*   Application Note                                                                                                   */
/*                                                                                                                      */
/*   'Memory Backup to EEPROM by LVDI'                                                                                  */
/*                                                                                                                      */
/*   Function                                                                                                           */
/*   : LVD (Interrupt by lowvoltage detect)                                                                             */
/*   : IIC Bus Interface 2 (EEPROM Access)                                                                              */
/*                                                                                                                      */
/*   External Clock : 16MHz                                                                                             */
/*   Internal Clock : 16MHz                                                                                             */
/*   Sub Clock      : 32.768kHz                                                                                         */
/*                                                                                                                      */
/************************************************************************************************************************/

#include   <machine.h>

/************************************************************************************************************************/
/*   Symbol Definition                                                                                                  */
/************************************************************************************************************************/
struct BIT {
     unsigned char   b7:1;          /* bit7 */
     unsigned char   b6:1;          /* bit6 */
     unsigned char   b5:1;          /* bit5 */
     unsigned char   b4:1;          /* bit4 */
     unsigned char   b3:1;          /* bit3 */
     unsigned char   b2:1;          /* bit2 */
     unsigned char   b1:1;          /* bit1 */
     unsigned char   b0:1;          /* bit0 */
};

#define    LVDCR         *(volatile unsigned char *)0xF730                  /* Low-voltage-detection control Register   */
#define    LVDCR_BIT     (*(struct BIT *)0xF730)                            /* Low-voltage-detection control Register   */
#define    LVDE          LVDCR_BIT.b7                                       /* LVD Enable                               */
#define    LVDSEL        LVDCR_BIT.b3                                       /* LVDI Detection Level Select              */
#define    LVDRE         LVDCR_BIT.b2                                       /* LVDR Enable                              */
#define    LVDSR         *(volatile unsigned char *)0xF731                  /* Low-Voltage-Detection Status Register    */
#define    LVDSR_BIT     (*(struct BIT *)0xF731)                            /* Low-Voltage-Detection Status Register    */
#define    LVDDF         LVDSR_BIT.b1                                       /* LVD Power-Supply Voltage Fall            */
#define    LVDUF         LVDSR_BIT.b0                                       /* LVD Power-Supply Voltage Rise            */
#define    PDR7_BIT      (*(struct BIT *)0xFFDA)                            /* Port Data Register 7                     */
#define    P74           PDR7_BIT.b4                                        /* Port Data Register 7 bit4                */
#define    PMR1_BIT      (*(struct BIT *)0xFFE0)                            /* Port mode Register 1                     */
#define    IRQ1          PMR1_BIT.b5                                        /* P15/IRQ1 Pin Function Switch             */
#define    IRQ0          PMR1_BIT.b4                                        /* P14/IRQ0 Pin Function Switch             */
#define    PCR7_BIT      (*(struct BIT *)0xFFEA)                            /* Port Control Register 7                  */
#define    PCR74         PCR7_BIT.b4                                        /* Port Control Register 7 bit4             */
#define    SYSCR1        *(volatile unsigned char *)0xFFF0                  /* System Control Register 1                */
#define    SYSCR2        *(volatile unsigned char *)0xFFF1                  /* System Control Register 2                */
#define    IEGR1_BIT     (*(struct BIT *)0xFFF2)                            /* Interrupt Edge Select Register 1         */
#define    IEG1          IEGR1_BIT.b1                                       /* IRQ0 Edge Select                         */
#define    IEG0          IEGR1_BIT.b0                                       /* IRQ0 Edge Select                         */
```

```
#define    IENR1_BIT     (*(struct BIT *)0xFFF4)                                  /* Interrupt Enable Register 1             */
#define    IEN1          IENR1_BIT.b1                                             /* IRQ0 Interrupt Enable                   */
#define    IEN0          IENR1_BIT.b0                                             /* IRQ0 Interrupt Enable                   */
#define    IRR1_BIT      (*(struct BIT *)0xFFF6)                                  /* Interrupt Request Register 1            */
#define    IRRI1         IRR1_BIT.b1                                              /* IRQ1 Interrupt Request Flag             */
#define    IRRI0         IRR1_BIT.b0                                              /* IRQ0 Interrupt Request Flag             */

#pragma interrupt  (irq0int)
#pragma interrupt  (irq1int)
/************************************************************************************************************************/
/*   Function define                                                                                                    */
/************************************************************************************************************************/
extern void INIT ( void );                                                        /* SP Set                                  */
extern unsigned char Write_byte_EEPROM ( unsigned short adrs , unsigned char wr_data );
extern unsigned char Read_byte_EEPROM ( unsigned short adrs );
void main ( void );
void irq0int ( void );
void irq1int ( void );
void sleep ( void );

/************************************************************************************************************************/
/*   RAM define                                                                                                         */
/************************************************************************************************************************/
volatile unsigned char lpcnt;
union addt{
     unsigned long    LONG;
     unsigned char    BYTE[4];
}eepbuf;

/************************************************************************************************************************/
/*   Vector Address                                                                                                     */
/************************************************************************************************************************/
#pragma section    V1                                                             /* VECTOR SECTOIN SET                      */
void (*const VEC_TBL1[])(void) = {                                                /* 0x00 - 0x0f                             */
     INIT                                                                         /* 00 Reset                                */
};
#pragma section    V2                                                             /* VECTOR SECTOIN SET                      */
void (*const VEC_TBL2[])(void) = {
     irq0int                                                                      /* 1C IRQ0 Interrupt                       */
};
#pragma section    V3                                                             /* VECTOR SECTOIN SET                      */
void (*const VEC_TBL3[])(void) = {
     irq1int                                                                      /* 1E IRQ1 Interrupt                       */
};

#pragma section                                                                   /* P                                       */
```

```
/*****************************************************************************************************************************************/
/*   Main Program                                                                                                                        */
/*****************************************************************************************************************************************/
void main ( void )
{
     unsigned long  i;
     unsigned char  tmp;

     set_imask_ccr(1);                                                    /* Interrupt Disable                    */

     IRRI0 = 0;                                                           /* Clear IRRI0                          */
     IEG1 = 1;                                                            /* Rising Edge of IRQ1 Input            */
     IRQ1 = 1;                    /* Initialize IRQ1 Terminal Input       */
     IEN1 = 1;                    /* IRQ1 Interruput Enable               */
     IRRI1 = 0;                                                           /* Clear IRRI1                          */

     LVDE = 1;                                                            /* LVD Enable                           */
     for(i=0; i<800; i++);                                                /* 50us Wait                            */
     tmp = LVDSR;
     LVDSR = 0xFC;                                                        /* Clear LVDDF,LVDUF                    */
     LVDCR = 0xF7;                                                        /* Set LVDRE,LVDDE,LVDUE                */

     PCR74 = 1;                                                           /* P74 Output Pin                       */
     P74 = 0;                                                             /* P74 is Low                           */

     Read_n_EEPROM(0x64, eepbuf.BYTE, 4);                                 /* Read EEPROM DATA -> eepbuf           */
     if(!((eepbuf.LONG==0x040000)|(eepbuf.LONG==0x100000)))
          eepbuf.LONG = 0x040000;                                         /* Set eepbuf                           */

     lpcnt = 0;                                                           /* Clear lpcnt                          */
     set_imask_ccr(0);                                                    /* Interrupt Enable                     */
     while(1){
          if(lpcnt == 2){                                                 /* IRQ1 SW On?                          */
               if(eepbuf.LONG == 0x100000)                                /* Change eepbuf data                   */
                    eepbuf.LONG = 0x040000;
               else
                    eepbuf.LONG = 0x100000;
               lpcnt = 0;                                                 /* Clear lpcnt                          */
          }

          for(i=0; i<eepbuf.LONG; i++);                                   /* Wait Loop                            */
          P74 = ~P74;

          if(lpcnt == 1){                                                 /* Lowvoltage detect ?                  */
               SYSCR1 = 0xC0;
               SYSCR2 = 0x0C;
               lpcnt = 0;                                                 /* Clear lpcnt                          */
               sleep();                                                   /* Transition to Standby Mode           */
          }
     }
}
```

```
/*****************************************************************************************************************************************/
/*   IRQ0 Interrupt                                                                                                                      */
/*****************************************************************************************************************************************/
void irq0int ( void )
{
     unsigned char   tmp;

     IRRI0 = 0;                                                                 /* Clear IRRI0                                 */
     if(LVDDF == 1){                                                            /* LVD Power-Supply Voltage Fall?              */
          Write_n_EEPROM(0x64, eepbuf.BYTE, 4);                                 /* Memory Backup to EEPROM                     */
          lpcnt = 1;                                                            /* Set Standby Mode flag                       */
     }
     else if(LVDUF == 1)                                                        /* LVD Power-Supply Voltage Rise?              */
          lpcnt = 0;                                                            /* IRQ0 Interrupt / Active Mode                */

     tmp = LVDSR;
     LVDSR = 0xFC;                                                              /* Clear LVDDF,LVDUF                           */
}

/*****************************************************************************************************************************************/
/*   IRQ1 Interrupt                                                                                                                      */
/*****************************************************************************************************************************************/
void irq1int ( void )
{
     IRRI1 = 0;                                                                 /* Clear IRRI1                                 */
     lpcnt = 2;                                                                 /* Set lpcnt                                   */
}

/*****************************************************************************************************************************************/
/*                                                                                                                                       */
/*   IIC2 EEPROM read/write                                                                                                              */
/*        H8/3687,H8/3694 EEPROM Function                                                                                                */
/*                                                                                                                                       */
/*****************************************************************************************************************************************/
#include    <machine.h>
#include    "H8_3687_IIC2.H"

/*****************************************************************************************************************************************/
/*   Symbol Definition                                                                                                                   */
/*****************************************************************************************************************************************/
#define     DEVICE_CODE 0xA0                                                    /* EEPROM DEVICE CODE:1010                     */
#define     SLAVE_ADRS  0x00                                                    /* SLAVE ADRS:0                                */
#define     IIC_DATA_W  0x00                                                    /* WRITE_DATA                                  */
#define     IIC_DATA_R  0x01                                                    /* READ_DATA                                   */

/*****************************************************************************************************************************************/
/*   Function define                                                                                                                     */
/*****************************************************************************************************************************************/
unsigned char Set_adrs_EEPROM ( unsigned short adrs );
void Write_data_EEPROM ( unsigned char wr_data );
void Write_data_End_EEPROM ( unsigned char wr_data );
unsigned char Recv_datan_EEPROM ( unsigned char *rd_ptr , unsigned short no );
```

```
/*****************************************************************************************************************************************/
/*   Main Program                                                                                                                        */
/*   Read_n_EEPROM   (n:2-512 byte)                                                                                                      */
/*        argument1:read address(unsigned short)                                                                                         */
/*        argument2:read data address(unsigned char *)                                                                                   */
/*        argument3:read data number(unsigned short)                                                                                     */
/*        return: 1:OK/0:NG EEPROM NOACK   (unsigned char)                                                                               */
/*****************************************************************************************************************************************/
unsigned char Read_n_EEPROM ( unsigned short adrs , unsigned char *rd_ptr , unsigned short no )
{
     unsigned char   ack;

     IIC2.ICCR1.BYTE = 0x81;                                          /* Initialize (ICE=1,CKS=0001)             */

     ack = 0;
     while(IIC2.ICCR2.BIT.BBSY != 0);                                 /* Bus busy?                               */

     ack = Set_adrs_EEPROM(adrs);                                     /* Set address (dummy write)               */
     if(ack == 1)
          ack = Recv_datan_EEPROM(rd_ptr,no);                         /* Data read n byte                        */

     return(ack);
}

/*****************************************************************************************************************************************/
/*   Write_page_EEPROM   (8byte)                                                                                                         */
/*        argument1:write address(unsigned short)                                                                                        */
/*        argument2:write data address(unsigned char *)                                                                                  */
/*        argument3:write data number(unsigned short)                                                                                    */
/*        return: 1:OK/0:NG EEPROM NOACK   (unsigned char)                                                                               */
/*****************************************************************************************************************************************/
unsigned char Write_n_EEPROM( unsigned short adrs , unsigned char *wr_ptr , unsigned short no )
{
     unsigned short  i;
     unsigned char   ack;

     IIC2.ICCR1.BYTE = 0x81;                                          /* Initialize (ICE=1,CKS=0001)             */

     while(IIC2.ICCR2.BIT.BBSY != 0);                                 /* Bus busy?                               */

     ack = Set_adrs_EEPROM(adrs);                                     /* Set address (dummy write)               */
     no = no-1;
     if(ack == 1){
          for(i = 0; i < no; i++){
               Write_data_EEPROM(*wr_ptr);                           /* Data write1-7                           */
               wr_ptr++;
          }
          Write_data_End_EEPROM(*wr_ptr);                            /* Data write8 (last data)                 */
     }

     return(ack);
}
```

```
/*****************************************************************************************************************************************/
/*   Write_data_EEPROM                                                                                                                   */
/*        argument1:write data(unsigned char)                                                                                            */
/*        return: none                                                                                                                   */
/*****************************************************************************************************************************************/
void Write_data_EEPROM( unsigned char wr_data )
{
     IIC2.ICDRT = wr_data;                                                  /* < >Data set                                              */
     while(IIC2.ICSR.BIT.TDRE == 0);                                        /* < >Finish Send?                                          */
}

/*****************************************************************************************************************************************/
/*   Write_data_End_EEPROM                                                                                                               */
/*   argument1:write data(unsigned char)                                                                                                 */
/*   return: none                                                                                                                        */
/*****************************************************************************************************************************************/
void  Write_data_End_EEPROM( unsigned char wr_data )
{
     IIC2.ICDRT = wr_data;                                                  /* <3>Set low address                                       */
     while(IIC2.ICSR.BIT.TEND == 0);                                        /* <3>send end?                                             */

     IIC2.ICSR.BIT.STOP = 0;                                                /* (STOP=0)                                                 */
     IIC2.ICCR2.BYTE = 0x3D;                                                /* (BSY=0,SCP=0)                                            */

     while(IIC2.ICSR.BIT.STOP == 0);                                        /* STOP end?                                                */

     IIC2.ICCR1.BYTE = 0x81;                                                /* End Master send(MST=0,TRS=0)                             */
     IIC2.ICSR.BIT.TDRE = 0;                                                /* TDRE = 0                                                 */
}
```

```
/*****************************************************************************************************************************************/
/*   Set_adrs_EEPROM                                                                                                                     */
/*        argument1:write/read address (unsigned short)                                                                                  */
/*        return: 1:OK/0:NG EEPROM NOACK   (unsigned char)                                                                               */
/*****************************************************************************************************************************************/
/*****************************************************************************************************************************************/
/*   (ADDRESS SET ACTION / DUMMY WRITE ACTION)                                                                                           */
/*   <1>            <2>            <3>                                                                                                  */
/*   123456789      123456789      123456789                                                                                            */
/*   101000000      000000000      000000000                                                                                            */
/*   slave  WA      addressHI A    addressLO A                                                                                          */
/*****************************************************************************************************************************************/
unsigned char Set_adrs_EEPROM( unsigned short adrs )
{
     IIC2.ICCR1.BYTE = 0xB1;                                                    /* Set Master send(MST=1,TRS=1)            */
     IIC2.ICCR2.BYTE = 0xBD;                                                    /* (BSY=1,SCP=0)                           */

                                                                                /* <1>                                     */
     IIC2.ICDRT = (unsigned char)(DEVICE_CODE | SLAVE_ADRS | IIC_DATA_W);       /* <1>Set slave address                    */

     while(IIC2.ICSR.BIT.TEND == 0);                                            /* <1>send end?                            */

     if(IIC2.ICIER.BIT.ACKBR != 0)                                              /* ACK?                                    */
          return(0);

                                                                                /* <2>                                     */
     IIC2.ICDRT = (unsigned char)(adrs >> 8);                                   /* <2>Set high address                     */
     while(IIC2.ICSR.BIT.TDRE == 0);                                            /* <2>send end?                            */
                                                                                /* <3>                                     */
     IIC2.ICDRT = (unsigned char)(adrs & 0x00FF);                               /* <3>Set low address                      */
     while(IIC2.ICSR.BIT.TEND == 0);                                            /* <3>send end?                            */

     return(1);
}
```

```
/*****************************************************************************************************************************************/
/*   Recv_datan_EEPROM                                                                                                                   */
/*        argument1:read data address(unsigned char *)                                                                                   */
/*        argument2:read byte 1to64(unsigned char)                                                                                       */
/*        return: 1:OK/0:NG EEPROM NOACK   (unsigned char)                                                                               */
/*****************************************************************************************************************************************/
/*****************************************************************************************************************************************/
/*   (CURRENT ADDRESS READ ACTION)                                                  */
/*   <1>            <2>            <3>           <n>                                                                                     */
/*   123456789      123456789      123456789     123456789                                                                               */
/*   101000010      000000000      000000000     000000000                                                                               */
/*   slave  RA      readdataA readdataA          readdataA                                                                               */
/*****************************************************************************************************************************************/
unsigned char Recv_datan_EEPROM( unsigned char *rd_ptr , unsigned short no )
{
     unsigned char   recv;

     IIC2.ICCR1.BYTE = 0xB1;                                                    /* (ICE=1,TRS=1)                                          */
     IIC2.ICCR2.BYTE = 0xBD;                                                    /* (BSY=1,SCP=0)                                          */

                                                                                /* <1>                                                    */
     IIC2.ICDRT = (unsigned char)(DEVICE_CODE | SLAVE_ADRS | IIC_DATA_R);       /* <1>Set slave address                                   */

     while(IIC2.ICSR.BIT.TEND == 0);                                            /* <1>send end?                                           */

     if(IIC2.ICIER.BIT.ACKBR != 0)                                              /* ACK?                                                   */
          return(0);

                                                                                /* Set Master recv                                        */
     IIC2.ICSR.BIT.TEND = 0;                                                    /* TEND = 0                                               */
     IIC2.ICCR1.BIT.TRS = 0;                                                    /* Set Master recv(TRS = 0)                               */
     IIC2.ICSR.BIT.TDRE = 0;                                                    /* TDRE = 0                                               */
     IIC2.ICIER.BIT.ACKBT = 0;                                                  /* (ACKBT = 0)                                            */

     recv = IIC2.ICDRR;                                                         /* dummy read                                             */
     while(IIC2.ICSR.BIT.RDRF == 0);                                            /* dummy recv end?                                        */
/*---------------------------------------------------------------------------------------------------------------------------------------*/
     while(2 < no){
          *rd_ptr = IIC2.ICDRR;                                                 /* <>read                                                 */
          while(IIC2.ICSR.BIT.RDRF == 0);                                       /* <>recv end?                                            */
          rd_ptr++;                                                             /* address increment                                      */
          no--;
     }
/*---------------------------------------------------------------------------------------------------------------------------------------*/
     IIC2.ICIER.BIT.ACKBT = 1;                                                  /* (ACKBT = 1)                                            */
     IIC2.ICCR1.BIT.RCVD = 1;                                                   /* (RCVD = 1)                                             */

     *rd_ptr = IIC2.ICDRR;                                                      /* <no-1>read                                             */
     rd_ptr++;                                                                  /* address increment                                      */
     while(IIC2.ICSR.BIT.RDRF == 0);                                            /* <no-1>recv end?                                        */
/*---------------------------------------------------------------------------------------------------------------------------------------*/
     IIC2.ICSR.BIT.STOP = 0;                                                    /* (STOP=0)                                               */
     IIC2.ICCR2.BYTE = 0x3D;                                                    /* (BSY=0,SCP=0)                                          */
     while(IIC2.ICSR.BIT.STOP == 0);                                            /* STOP end?                                              */

     *rd_ptr = IIC2.ICDRR;                                                      /* <no>read                                               */
     IIC2.ICCR1.BIT.RCVD = 0;                                                   /* (RCVD = 0)                                             */
     IIC2.ICCR1.BIT.MST = 0;                                                    /* (MST=0)                                                */

     return(1);
}
```

リンクアドレス指定

セクション名	アドレス
CV1	0x0000
CV2	0x001C
CV3	0x001E
P	0x0100
B	0xFB80

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
2.00	2004.05.07	—	誤記修正

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。