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。

H8/3687F シリーズ
LIN(Local Interconnect Network)アプリケーションノート スレーブ編

RJJ05B0303-0141Z
Rev.1.41
2003.06.20

要旨

LIN(Local Interconnect Network)アプリケーションノートスレーブ編は、H8/300H Tiny シリーズマイコンの内蔵周辺機能を使用し、LIN 通信プロトコルによる通信を行う仕様例、設定例を記しており、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。

尚、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

H8/300H Tiny シリーズ　―H8/3687F―

目次

1. LIN 通信システム概要 3
1.1 LIN バスへの接続 3
1.1.1 システム構成 3
1.1.2 LIN バス(シングルワイヤバス)インタフェース 3
1.2 LIN 通信概要 4
1.2.1 メッセージフレーム構成 4
1.2.2 メッセージフレームの送受信 5
2. ライブラリソフトウェア仕様 6
2.1 動作環境 6
2.2 ファイル構成 6
2.3 ROM/RAM 使用容量 6
2.4 機能仕様 7
2.4.1 LIN 通信仕様 7
2.4.2 LINID.h ファイル設定例 8
2.4.3 ユーザアプリケーションインタフェース 13
2.5 動作説明 16
2.5.1 ヘッダフレームの受信 16
2.5.1.1 シンクブレークフィールドの検出 16
2.5.1.2 シンクフィールドの受信 17
2.5.1.3 ID フィールドの受信 19
2.5.2 レスポンスフレームの送受信 20
2.5.2.1 データフィールドの送受信(チェックサムフィールドの送信) 20
2.5.2.2 チェックサムフィールドの受信 21
2.5.3 ウェイクアップ信号の送受信 22
2.5.3.1 ウェイクアップ信号の送信 22
2.5.3.2 ウェイクアップ信号の受信 23
2.5.4 スリープコマンドの受信 23
2.6 ソフトウェア説明 24
2.6.1 ヘッダファイルの組み込み 24
2.6.2 関数定義 24
2.6.3 ライブラリ内部定数・変数定義 25
2.6.4 グローバル変数定義 29
2.6.5 初期設定用関数 30
2.6.6 LIN 通信制御停止用関数 33
2.6.7 LIN 通信制御再開準備用関数 33
2.6.8 ウェイクアップ信号送信用関数 34
2.6.9 ウェイクアップ信号受信準備用関数 34
2.6.10 ライブラリ内部シンクブレークフィールド検出準備用関数 35
2.6.11 IRQ 割込み関数 36
2.6.12 タイマ Z 割込み関数 37
2.6.13 SCI3 割込み関数 41
3. 参考ドキュメント 49

1. LIN 通信システム概要

本アプリケーションノートに掲載している LIN 通信用ソフトウェアライブラリサンプル(以下ライブラリと称す)を組み込んだシステムによる LIN 通信の概要を示します。

1.1 LIN バスへの接続

LIN バスによりネットワーク接続されたシステム(図１)に LIN バスインタフェース回路(または LIN トランシーバ)を介して接続することにより、マスターノードとしてヘッダフレームの送信、レスポンスフレームの送受信等の LIN 通信を行います。

1.1.1 システム構成

図１に LIN バスネットワークシステム構成例を示します。

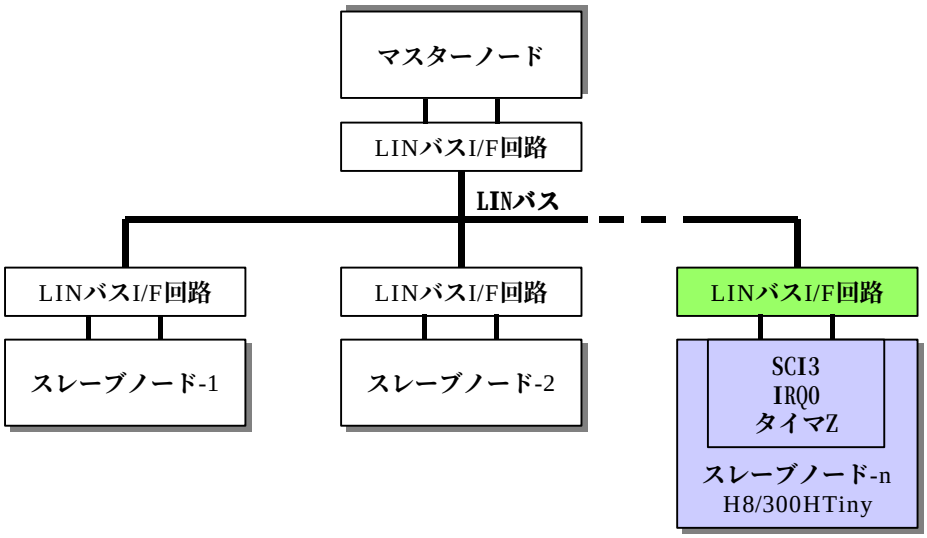


図 1 LIN バス接続によるシステムブロック図

1.1.2 LIN バス(シングルワイヤバス)インタフェース

図２に H8/300HTiny シリーズマイコン(以下マイコンと称す)内蔵機能の入出力端子と LIN バスとのインターフェイス回路例を示します。

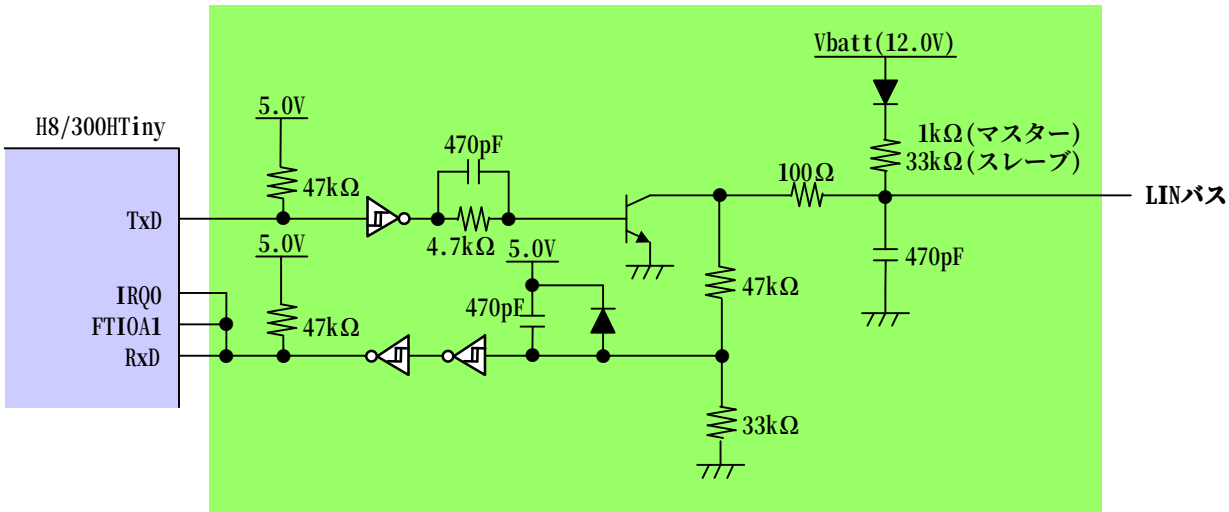


図 2 LIN バスインタフェース回路例

1.2 LIN 通信概要

LIN 通信プロトコルにより送受信されるメッセージフレームの概要を示します。

1.2.1 メッセージフレーム構成

図３にメッセージフレームの構成を示します。メッセージフレームは、マスターノードから送信されるヘッダフレームと、マスターノードもしくはスレーブノードから送信されるレスポンスフレームから構成されます。

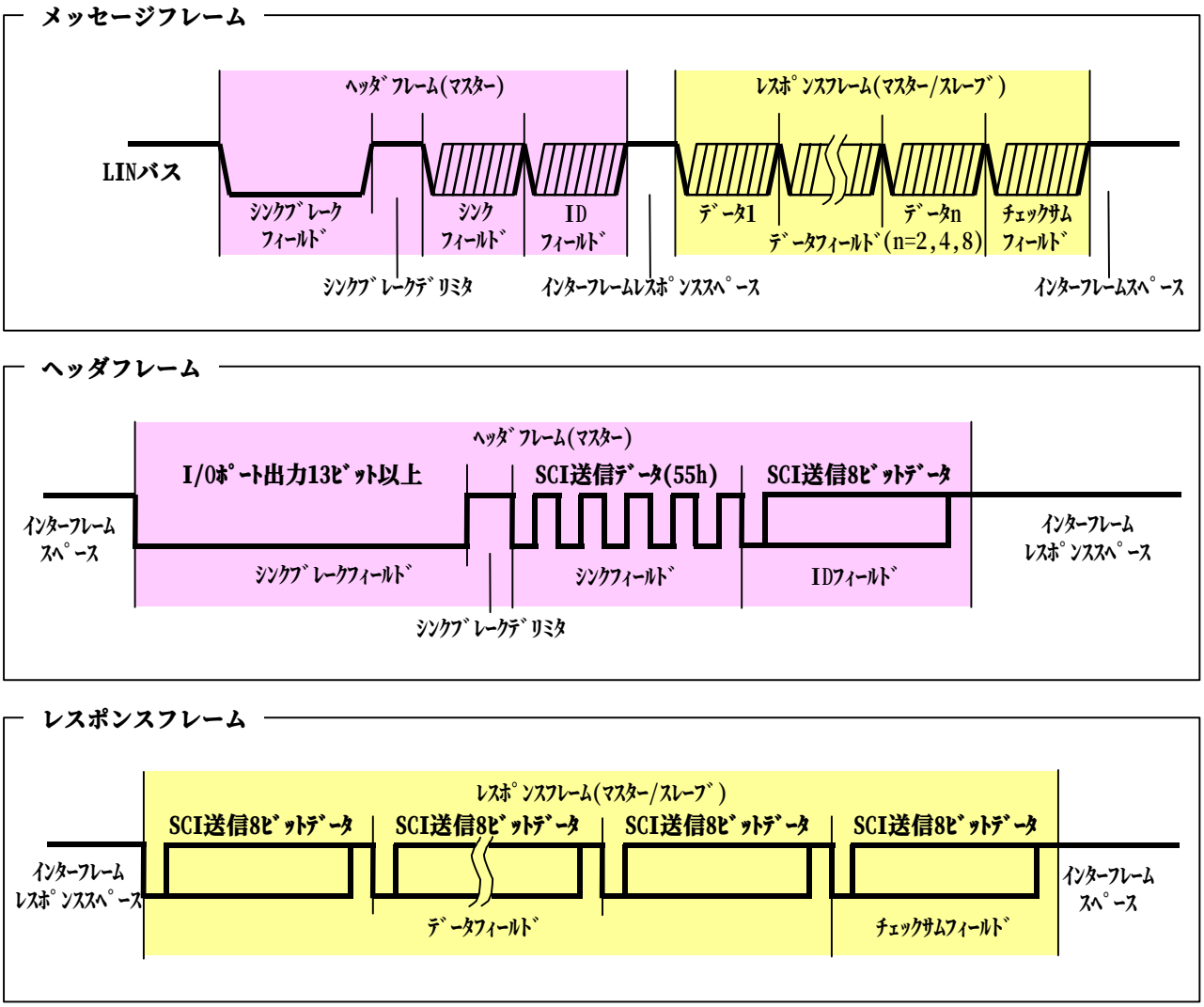


図 3 メッセージフレームの構成

1.2.2 メッセージフレームの送受信

図４にマスター/スレーブ各ノードにおけるメッセージフレームの送受信イメージを示します。

- マスターノードがヘッダフレームを送信します。
- スレーブノードは、受信したヘッダフレームからIDを判定し、自ノードに対するIDの時レスポンスを送信します。(マスターノードは送信時にIDを判定)

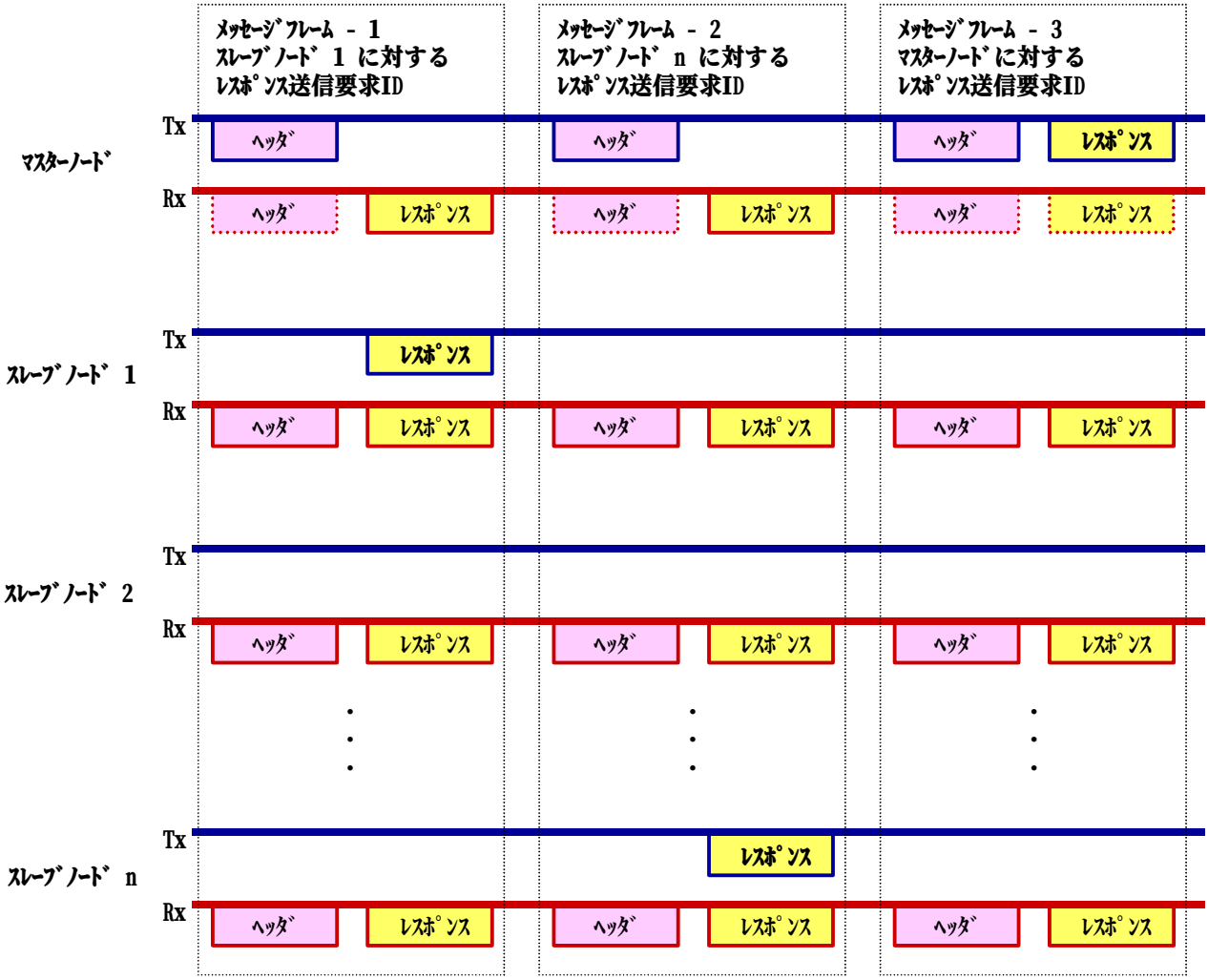


図 4 メッセージフレームの送受信イメージ

2. ライブラリソフトウェア仕様

ライブラリをユーザアプリケーションプログラムに組み込む事によりマイコン内蔵機能を使用し、スレーブノードとして LIN 通信を行います。

2.1 動作環境

使用デバイス：H8/300HTiny シリーズマイコン(H8/3687F)
動作周波数範囲(システム搭載クロック(φosc))：デバイス動作周波数と同等範囲。但し、LIN 通信速度及びユーザアプリケーションプログラムの処理条件により考慮し、LINID.h 内にφosc を定義する必要があります。(2.4.2 LINID.h ファイル設定例参照)
使用機能：ライブラリにおいて使用するマイコンの内蔵周辺機能と使用用途を表 1 に示します。

表 1 マイコン内蔵周辺機能使用表

機能		端子機能(端子No.)	用途	説明
SCI3 (channel-0)	送信	TXD (46pin)	レスポンスフレーム送信 ウェイクアップ信号送信	調歩同期式モード データ長8ビット パリティービット無し 1ストップビット(スタートビット付加) LSBファースト
	受信	RXD (45pin)	レスポンスフレーム受信	
			通信エラー検出	モジュール内エラー検出機能
タイマZ (channel-1)		FTIOA1 (37pin)	シンクブレークフィールドドミナント期間計測 シンクフィールド期間計測 ウェイト期間計測(ライブラリ内部機能) タイムアウト検出	φosc/8周期でカウント動作し、各時間間隔を計測
IRQ		/IRQ0 (51pin)	ウェイクアップ信号検出	スタンバイ時にLINバスを監視し立下りエッジを検出

2.2 ファイル構成

- **LINslvZ.c(Ver.1.41)**
  H8/300HTiny シリーズ(タイマ Z 内蔵版)における LIN 通信用(スレーブ)マイコン機能設定、及び通信制御を行うための C ソースファイルです。

- **LINID.h(Ver.1.41)**
  ユーザにより通信転送速度の設定や ID の設定等を行い LINslvZ.c(Ver.1.41)コンパイル時に組み込むためのインクルードファイルです。また、ユーザアプリケーションインタフェース用関数、変数定義も同ファイル内で行っているため、ユーザアプリケーションプログラムコンパイル時にも組み込む必要があります。

- **H8_3687.h(Ver.1.00)**
  H8/3687F シリーズ用内蔵 I/O レジスタ定義ファイルです。

2.3 ROM/RAM 使用容量

(H8S,H8300 シリーズ C コンパイラ CH38.exe Ver.2.0C 使用時)

ROM/RAM 使用容量は、ID 設定数等により変化します。

- **ROM 容量：約** 2.0 kByte
- **RAM 容量：約** 40 Byte

2.4 機能仕様

2.4.1 LIN 通信仕様

- **ノード**：スレーブノードに対応

- **ID**：ユーザ定義 ID

レスポンス送信 ID

LINID.h により 0 個から 61 個(00h～3bh,3dh)まで設定可能。

(但し、同一 LIN バス上に同じ ID を持つノードを設定すると正常動作しません)

LIN プロトコル定義 ID

a)マスターリクエストフレーム ID　3ch(ID フィールドデータ：3Ch)

マスターノードからレスポンスフレーム(データ長 8 バイト)が送信されます。データフィールド 1 バイト目が 00h であった場合、スリープコマンド受信と認識しステータスフラグ(表 4 参照)をセットします。

b)スレーブレスポンスフレーム ID　3dh(ID フィールドデータ：7Dh)

この ID を持つスレーブノードがレスポンスフレーム(データ長 8 バイト)を送信します。

c)拡張フレーム ID　3eh,3fh(ID フィールドデータ：FEh,BFh)

本ライブラリ(Ver.1.41)では、対応しておりません。

(受信時は、次のメッセージフレーム(シンクブレークフィールド検出)待機状態となります。)

ID 設定方法

LINID.h 内でレスポンス送信 ID として設定する ID 以外の定義文(#define　__ID**m**　0xnn (**m**=**00**h～**3b**h,**3d**h))

を削除、またはコメント文とし、設定したい ID のみを定義した状態で LINslvZ.c をコンパイルします。

- **レスポンスデータ長**：受信 ID フィールド内 DLC(データレングスコントロール)ビットを判定。

- **通信転送速度**：LINID.h 内で使用する通信転送速度を定義します。

システム搭載クロック(φosc)定義値、及び通信転送速度定義値からライブラリ内で使用する定数、および SCI3 モジュール設定値を自動的に算出します。(注意：通信転送速度はφosc により制限されることがあります。詳しくはハードウェアマニュアルの SCI3 モジュール：ビットレートに対する BRR 設定例(調歩同期式モード)をご参考下さい)

- **ウェイクアップ信号送受信**：ウェイクアップ信号の送信機能、受信機能を組み込みます。

ウェイクアップ信号送信機能の組込み

LINID.h 内定義文(#define　__T_WAKEUP　__ON)により、ウェイクアップ送信機能を組み込み、ユーザアプリケーションプログラム等から関数(LIN_transmit_wake_up)を呼び出す事により LIN バス上にウェイクアップ信号を送信します。

ウェイクアップ信号受信機能の組込み

LINID.h 内定義文(#define　__R_WAKEUP　__ON)により、ウェイクアップ受信機能を組み込み、マイコンがスタンバイ状態等にあっても LIN バス上のウェイクアップ信号を IRQ0(外部割込み入力)により検出(立下りエッジ検出)します。

2.4.2 LINID.h ファイル設定例

LINID.h の設定例を示します。

a)ウェイクアップ信号の送信をしないノードとする。

b)IRQ0(外部割込み)によるウェイクアップ信号の検出(立下りエッジ検出)は行う。(他ノードからのウェイクアップ信号有り)

c)以下４個の ID に対してレスポンスフレームを送信する。

ID(ID ビット+DLC ビット)	(パリティービット含む)
03h	(03h)
3ah	(D3h)

d)システム搭載クロック(φosc)を 20[MHz]とする。

e)LIN 通信転送速度を 9600[bit/sec]とする。

f) シンクフィールド計測による LIN 通信転送速度の補正は行わない。

上記 a)～g)の仕様に基づき設定した例を以下に示します。

(太字以外の定義文を削除もしくはコメント行として下さい)

```
/*****************************************************************************************************************************************************/
/*                                                                                                                                                   */
/*                              LINID.h      Ver.1.41                                                                                                */
/*                                                                                                                                                   */
/*****************************************************************************************************************************************************/

#define     __ON     1                  /*  この行は変更または削除しないで下さい                                                                    */
#define     __OFF    0                  /*  この行は変更または削除しないで下さい                                                                    */

/*****************************************************************************************************************************************************/
/*      ウェイクアップ信号送信機能設定                                                                                                               */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
/*#define     __T_WAKEUP     __ON        /*  ウェイクアップ信号の送信を行う場合はこの行を定義                                                        */
#define     __T_WAKEUP     __OFF       /*  ウェイクアップ信号の送信を行わない場合はこの行を定義                                                    */

/*****************************************************************************************************************************************************/
/*      ウェイクアップ信号検出機能設定                                                                                                               */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
#define     __R_WAKEUP     __ON        /*  ウェイクアップ信号の検出(立下りエッジ検出)を行う場合はこの行を定義                                      */
/*#define     __R_WAKEUP     __OFF       /*  ウェイクアップ信号の検出を行わない場合はこの行を定義                                                    */

/*****************************************************************************************************************************************************/
/*      レスポンス送信 ID 設定                                                                                                                       */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
/*      2 バイトデータ                                                                                                                               */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
```

```
#define     __Res2byte_ID    __ON      /*  2 バイトのレスポンスデータ送信 ID を持つ場合はこの行を定義                */
/*#define    __Res2byte_ID    __OFF     /*  2 バイトのレスポンスデータ送信 ID を持たない場合はこの行を定義          */

#if    __Res2byte_ID    ==    __ON

/*#define    __ID00    0x80                                                    /*                    */
/*#define    __ID01    0xC1                                                    /*                    */
/*#define    __ID02    0x42                                                    /*                    */
#define     __ID03    0x03              /*  ID フィールド 03h に対してレスポンスを送信                    */
/*#define    __ID04    0xC4                                                    /*                    */
/*#define    __ID05    0x85                                                    /*                    */
/*#define    __ID06    0x06                                                    /*                    */
/*#define    __ID07    0x47                                                    /*                    */
/*#define    __ID08    0x08                                                    /*                    */
/*#define    __ID09    0x49                                                    /*                    */
/*#define    __ID0a    0xCA                                                    /*                    */
/*#define    __ID0b    0x8B                                                    /*                    */
/*#define    __ID0c    0x4C                                                    /*                    */
/*#define    __ID0d    0x0D                                                    /*                    */
/*#define    __ID0e    0x8E                                                    /*                    */
/*#define    __ID0f    0xCF                                                    /*                    */
/*#define    __ID10    0x50                                                    /*                    */
/*#define    __ID11    0x11                                                    /*                    */
/*#define    __ID12    0x92                                                    /*                    */
/*#define    __ID13    0xD3                                                    /*                    */
/*#define    __ID14    0x14                                                    /*                    */
/*#define    __ID15    0x55                                                    /*                    */
/*#define    __ID16    0xD6                                                    /*                    */
/*#define    __ID17    0x97                                                    /*                    */
/*#define    __ID18    0xD8                                                    /*                    */
/*#define    __ID19    0x99                                                    /*                    */
/*#define    __ID1a    0x1A                                                    /*                    */
/*#define    __ID1b    0x5B                                                    /*                    */
/*#define    __ID1c    0x9C                                                    /*                    */
/*#define    __ID1d    0xDD                                                    /*                    */
/*#define    __ID1e    0x5E                                                    /*                    */
/*#define    __ID1f    0x1F                                                    /*                    */

#endif

/*-------------------------------------------------------------------------------------------------------------------*/
/*          4 バイトデータ                                                                                           */
/*-------------------------------------------------------------------------------------------------------------------*/

/*#define    __Res4byte_ID    __ON      /*  4 バイトのレスポンスデータ送信 ID を持つ場合はこの行を定義              */
#define     __Res4byte_ID    __OFF     /*  4 バイトのレスポンスデータ送信 ID を持たない場合はこの行を定義          */

#if    __Res4byte_ID    ==    __ON

/*#define    __ID20    0x20                                                    /*                    */
/*#define    __ID21    0x61                                                    /*                    */
```

```
/*#define    __ID22    0xE2                                                    /*                    */
/*#define    __ID23    0xA3                                                    /*                    */
/*#define    __ID24    0x64                                                    /*                    */
/*#define    __ID25    0x25                                                    /*                    */
/*#define    __ID26    0xA6                                                    /*                    */
/*#define    __ID27    0xE7                                                    /*                    */
/*#define    __ID28    0xA8                                                    /*                    */
/*#define    __ID29    0xE9                                                    /*                    */
/*#define    __ID2a    0x6A                                                    /*                    */
/*#define    __ID2b    0x2B                                                    /*                    */
/*#define    __ID2c    0xEC                                                    /*                    */
/*#define    __ID2d    0xAD                                                    /*                    */
/*#define    __ID2e    0x2E                                                    /*                    */
/*#define    __ID2f    0x6F                                                    /*                    */

#endif

/*-------------------------------------------------------------------------------------------------------------------*/
/*        8 バイトデータ                                                                                             */
/*-------------------------------------------------------------------------------------------------------------------*/

#define     __Res8byte_ID    __ON      /*  8 バイトのレスポンスデータ送信 ID を持つ場合はこの行を定義                */
/*#define    __Res8byte_ID    __OFF     /*  8 バイトのレスポンスデータ送信 ID を持たない場合はこの行を定義           */

#if    __Res8byte_ID    ==    __ON

/*#define    __ID30    0xF0                                                    /*                    */
/*#define    __ID31    0xB1                                                    /*                    */
/*#define    __ID32    0x32                                                    /*                    */
/*#define    __ID33    0x73                                                    /*                    */
/*#define    __ID34    0xB4                                                    /*                    */
/*#define    __ID35    0xF5                                                    /*                    */
/*#define    __ID36    0x76                                                    /*                    */
/*#define    __ID37    0x37                                                    /*                    */
/*#define    __ID38    0x78                                                    /*                    */
/*#define    __ID39    0x39                                                    /*                    */
#define     __ID3a    0xBA          /*  ID フィールド BAh に対してレスポンスを送信                                */
/*#define    __ID3b    0xFB                                                    /*                    */
/*#define    __ID3d    0x7D                                                    /*                    */

#endif

/*********************************************************************************************************************/
/*        システム搭載クロック(φosc)定義部                                                                          */
/*-------------------------------------------------------------------------------------------------------------------*/
#define     OSC_Hz    20000000      /*  φosc=20.000MHz → 20000000                                                */
/*#define    OSC_Hz    16000000                                                /*  φosc=16.000MHz → 16000000  */
/*#define    OSC_Hz    10486000                                                /*  φosc=10.486MHz → 10486000  */
/*#define    OSC_Hz    10000000                                                /*  φosc=10MHz     → 10000000  */
/*#define    OSC_Hz    9830400                                                 /*  φosc=9.8304MHz →  9830400  */
/*#define    OSC_Hz    8000000                                                 /*  φosc=8MHz      →  8000000  */
/*#define    OSC_Hz    7372800                                                 /*  φosc=7.3728MHz →  7372800  */
```

```
/*#define     OSC_Hz     6000000                                    /* φosc=6MHz      →  6000000   */
/*#define     OSC_Hz     4915200                                    /* φosc=4.9152MHz →  4915200   */
/*#define     OSC_Hz     3579545                                    /* φosc=3.5795MHz →  3579545   */
/*#define     OSC_Hz     2457600                                    /* φosc=2.4576MHz →  2457600   */

/*****************************************************************************************************************************************************/
/*          ボーレート定義部                                                                                                                         */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
/*#define     B_rate     2400                                                    /*  2400bps  →  2400          */
/*#define     B_rate     4800                                                    /*  4800bps  →  4800          */
#define     B_rate     9600                    /*  9600bps  →  9600                                          */
/*#define     B_rate     10000                                                   /* 10000bps  → 10000          */
/*#define     B_rate     14400                                                   /* 14400bps  → 14400          */
/*#define     B_rate     15000                                                   /* 15000bps  → 15000          */
/*#define     B_rate     19200                                                   /* 19200bps  → 19200          */
/*#define     B_rate     20000                                                   /* 20000bps  → 20000          */

/*****************************************************************************************************************************************************/
/*          ボーレート補正機能設定                                                                                                                   */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
#define     __Corrects_baud_rate     __ON    /*  シンクフィールド計測によるボーレート補正を行う場合はこの行を定義       */
/*#define     __Corrects_baud_rate     __OFF   /*  シンクフィールド計測によるボーレート補正を行わない場合はこの行を定義 */

/*****************************************************************************************************************************************************/
/*          ライブラリ用定数算出部      以下は変更または削除しないで下さい                                                                           */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
#define     t_1_data            ((((OSC_Hz) / (B_rate)) + 0x04) >>3)
#define     t_11_data           ((((11 *   ((OSC_Hz) >>2)) / (B_rate)) + 0x01) >>1)
#define     t_128_data          (((((OSC_Hz) <<5) / (B_rate)) + 0x01) >>1)
#define     t_15k_data          (((0xEA6 * ((OSC_Hz) / (B_rate))) + 0x01) >>1)
#define     t_25k_data          (((0x186A * ((OSC_Hz) / (B_rate))) + 0x01) >>1)
#define     t_2byte_data        ((((91 *   ((OSC_Hz) >>2)) / (B_rate)) + 0x01) >>1)
#define     t_4byte_data        ((((119 * ((OSC_Hz) >>2)) / (B_rate)) + 0x01) >>1)
#define     t_8byte_data        ((((175 * ((OSC_Hz) >>2)) / (B_rate)) + 0x01) >>1)

#define     baudrate_data       ((((((OSC_Hz) >>4) / (B_rate)) + 0x01) >>1) - 1)

/*****************************************************************************************************************************************************/
/*          関数・変数定義部            以下は変更または削除しないで下さい                                                                           */
/*---------------------------------------------------------------------------------------------------------------------------------------------------*/
#if     ((__Res2byte_ID)  || (__Res4byte_ID)  ||  (__Res8byte_ID))
#define     __RESPONSE      __ON
#else
#define     __RESPONSE      __OFF
#endif

#ifndef     __LIN_LIB

extern      void      LIN_initialize(void);
extern      void      LIN_end(void);
extern      void      LIN_sleep(void);
extern      void      LIN_error(void);
```

```
extern      void      LIN_start(void);
extern      void      LIN_stop(void);

#if      __RESPONSE      ==      __ON
extern      void      LIN_data_Set(void);
#endif

#if      __T_WAKEUP      ==      __ON
extern      void      LIN_transmit_wake_up(void);
#endif

#if      __R_WAKEUP      ==      __ON
extern      void      LIN_wake_up(void);
extern      void      LIN_wake_up_PR(void);
#endif

#if      __RESPONSE      ==      __ON
extern      volatile      unsigned char      LIN_tx_data[8];
#endif

extern      volatile      unsigned char      LIN_rx_id;
extern      volatile      unsigned char      LIN_rx_data[8];
extern      volatile      union {
                              unsigned char      BYTE;
                              struct {
                                   unsigned char      NBA        :1;
                                   unsigned char      CSE        :1;
                                   unsigned char      ISFE       :1;
                                   unsigned char      TOA3B      :1;
                                   unsigned char      SNRE       :1;
                                   unsigned char      SCI        :1;
                                   unsigned char      SUC        :1;
                                   unsigned char      SLEEP      :1;
                              }    BIT;
}      LIN_status;
extern      volatile      union {
                              unsigned char BYTE;
                              struct {
                                   unsigned char      CBR        :1;
                                   unsigned char      wk6        :1;
                                   unsigned char      WU         :1;
                                   unsigned char      wk4        :5;
                              }    BIT;
}      LIN_control;

#endif
```

2.4.3 ユーザアプリケーションインタフェース

本ライブラリとユーザアプリケーションプログラムとのインタフェース仕様を示します。

- **関数(モジュール)呼び出し**によるインタフェース

ユーザアプリケーションプログラムからライブラリ内の関数を呼び出す事により、LIN 通信制御に必要なマイコン内蔵周辺機能の初期化、LIN 通信制御の停止・再開ウェイクアップ信号の送信制御、ウェイクアップ信号の受信準備を行います。

表 2 ユーザアプリケーションプログラムからのライブラリ内関数呼び出し

関数名	機能説明
LIN_initialize	LIN通信制御に必要なマイコン内蔵周辺機能の初期設定を行い通信制御を開始します。 LIN_start関数を呼び出す必要はありません。
LIN_stop	LIN通信制御を停止します。
LIN_start	LIN通信制御を再開します。(電源ON時にはLIN_initialize関数を呼び出して下さい)
LIN_transmit_wake_up	ウェイクアップ信号を送信します。
LIN_wake_up_PR	ウェイクアップ信号の受信準備を行います。

ライブラリ内から呼び出される関数をユーザアプリケーションプログラム内に準備する事により、LIN 通信中の各タイミング(送受信完了、通信エラー検出時等)での処理を行います。

表 3 ライブラリからのユーザアプリケーション制御用関数呼び出し

関数名	機能説明
LIN_wake_up	ウェイクアップ信号検出時のユーザアプリケーション制御用関数
LIN_sleep	スリープコマンド受信時のユーザアプリケーション制御用関数
LIN_data_set	レスポンスフレーム送信前のユーザアプリケーション制御用関数
LIN_end	メッセージフレーム送受信完了後のユーザアプリケーション制御用関数
LIN_error	LIN通信エラー検出時のユーザアプリケーション制御用関数

・ **動作概要**

図５、図６に動作概要を示します。

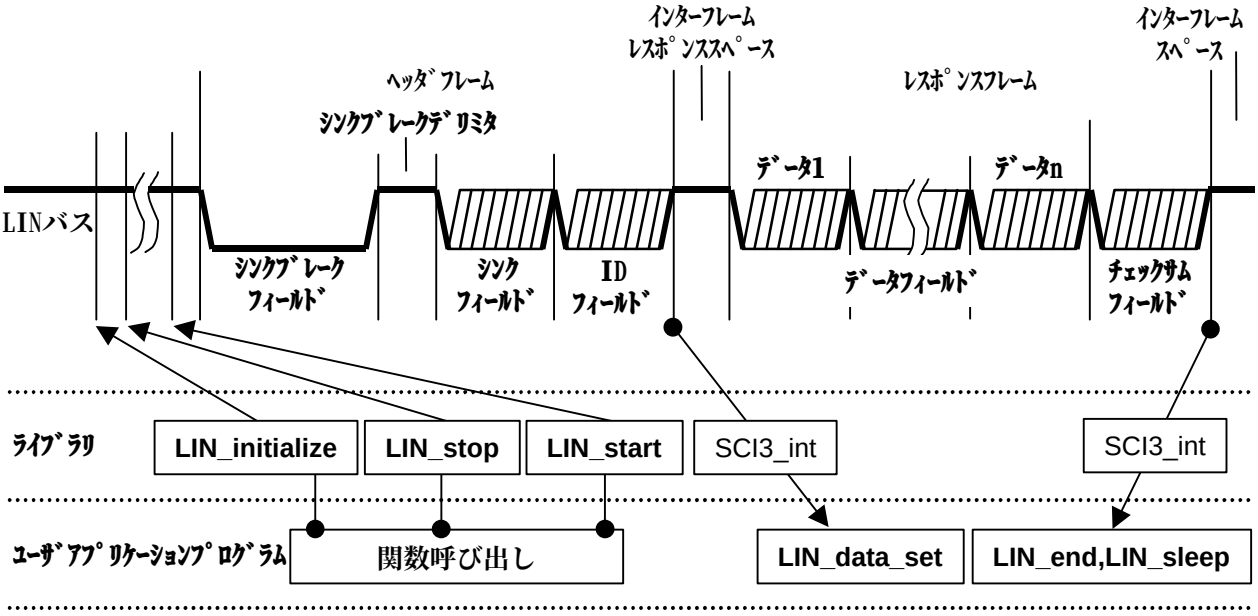


図 5　メッセージフレーム送受信時動作概要

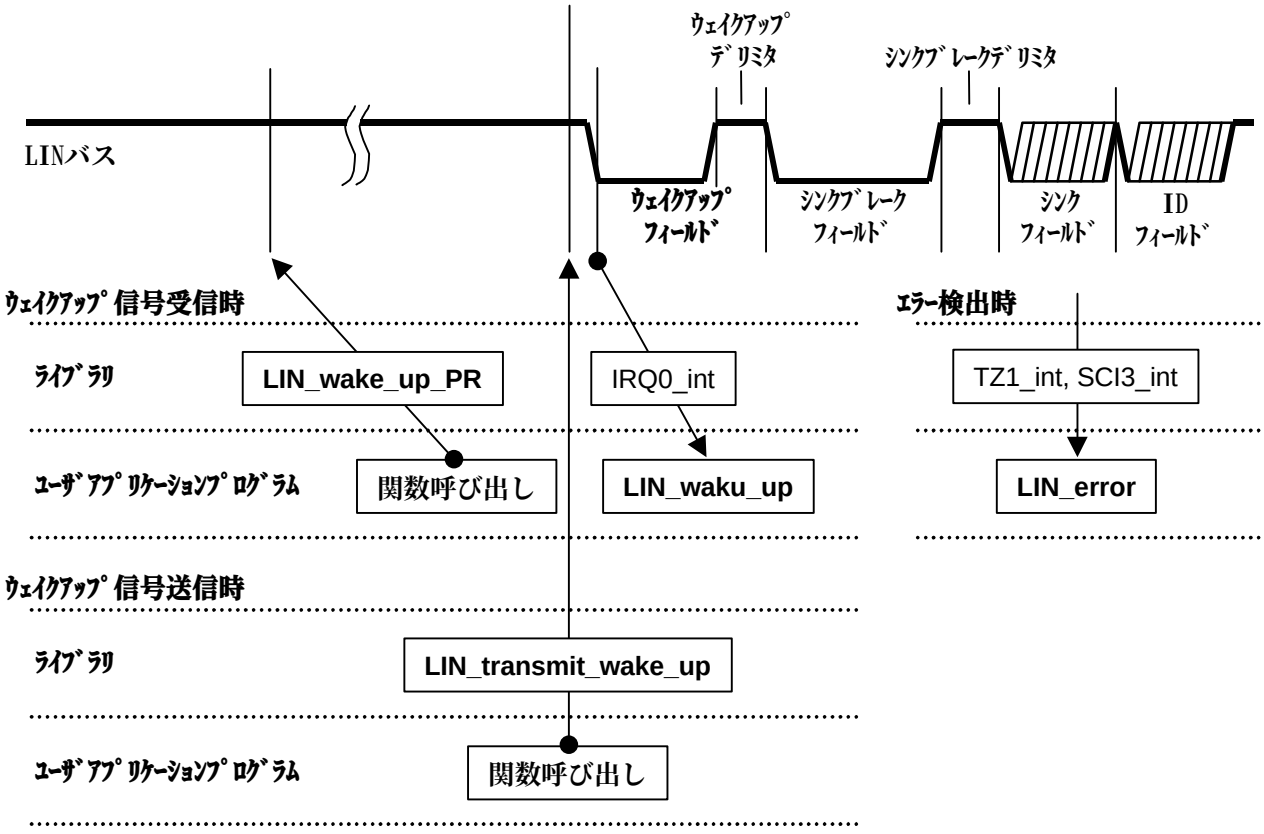


図 6　エラー検出、ウェイクアップ信号送受信時動作概要

・ **グローバル変数(RAM 領域格納データ)**によるインタフェース
ユーザアプリケーションプログラム、ライブラリでデータを共有する事によりインタフェースを行います。

表 4　ユーザアプリケーション・ライブラリ共有データ(グローバル変数)

ラベル名(変数名)	データ型	機能説明
LIN_tx_data[0]～[7]	unsigned char (配列)	レスポンスフレーム送信時の送信データを設定
LIN_rx_id	unsigned char	受信IDを格納
LIN_rx_data[0]～[7]	unsigned char (配列)	受信したレスポンスデータを格納
LIN_status	(構造体)	通信ステータス
LIN_status.BYTE	バイトアクセス unsigned char	
	ビットアクセス	
LIN_status.BIT.NBA	ビット- 7	ノーバスアクティブエラー セット条件：一定時間LINバス動作が無い場合
LIN_status.BIT.CSE	ビット- 6	チェックサムエラーフラグ セット条件：レスポンス受信時にチェックサムエラーを検出した場合
LIN_status.BIT.ISFE	ビット- 5	シンクフィールドエラー セット条件：受信したシンクフィールドデータ(SCI3モジュールによる受信データ)が55hでなかった場合
LIN_status.BIT.TOA3B	ビット- 4	ウェイクアップタイムアウト セット条件：ウェイクアップリトライ信号(初回を含め3回)送信後一定時間内にマスターから送信されるヘッダーフレームの検出が行われなかった場合
LIN_status.BIT.SNRE	ビット- 3	スレーブノットレスポンディングエラー セット条件：メッセージフレーム送受信中、一定時間内にスレーブからのレスポンスフレームが受信完了されなかった場合
LIN_status.BIT.SCI	ビット- 2	SCIエラー セット条件：SCI3モジュールによるエラー(オーバーランエラー,フレーミングエラー)を検出した場合
LIN_status.BIT.SUC	ビット- 1	メッセージフレーム正常受信完了フラグ セット条件：レスポンスフレームを正常受信完了した時 クリア条件：IDフレームが受信された時
LIN_status.BIT.SLEEP	ビット- 0	スリープコマンド受信フラグ セット条件：スリープコマンド受信時
LIN_control	(構造体)	通信コントロール
LIN_control.BYTE	バイトアクセス unsigned char	
	ビットアクセス	
LIN_control.BIT.CBR	ビット- 7	通信転送レート補正機能制御 (2.5.1.2シンクフィールドの受信参照)
LIN_control.BIT.wk6	ビット- 6	リザーブビット
LIN_control.BIT.WU	ビット- 5	(ウェイクアップコントロールビット) (2.5.3 ウェイクアップ信号の送受信参照)
LIN_control.BIT.wk4	ビット- 4～0	リザーブビット

2.5 動作説明

以下にライブラリによる送受信動作説明を示します。

2.5.1 ヘッダフレームの受信

2.5.1.1 シンクブレークフィールドの検出

タイマ W インプットキャプチャ機能により、シンクブレークフィールドドミナント期間を計測します。

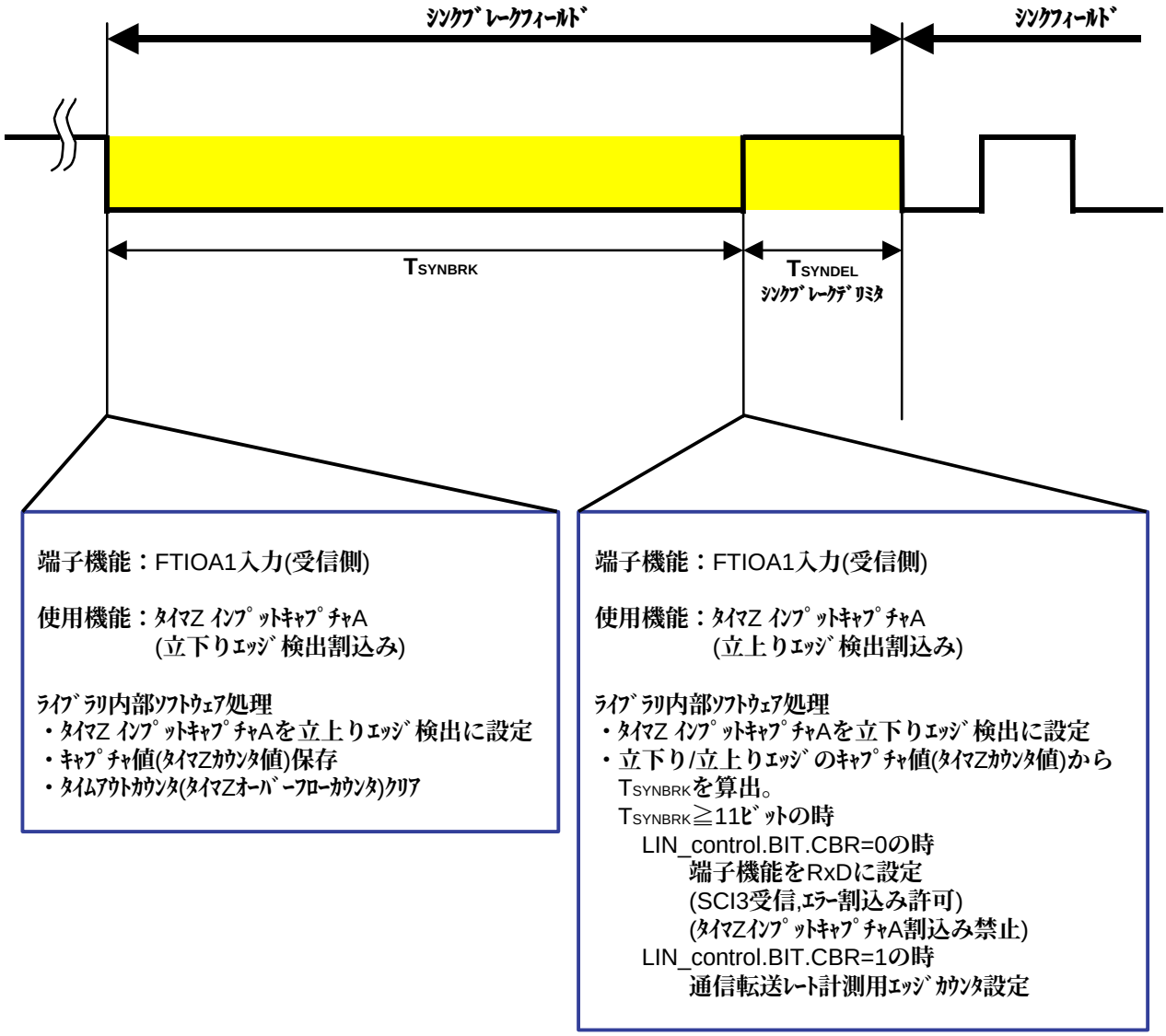


図 7 シンクブレークフィールドの検出

2.5.1.2 シンクフィールドの受信

LIN_control 内 CBR ビット(表 4 参照)の設定により、シンクフィールドの受信制御方法は以下のようになります。

CBR=0： SCI3 受信機能によりシンクフィールド受信データ(55h)の判定を行います。
(図 8 SCI3 受信機能によるシンクフィールドの受信と判定)

CBR=1： タイマ Z インプットキャプチャ機能により、シンクフィールドのビット幅を計測し通信転送レートを補正(SCI3 モジュール内 BRR 等を再設定)します。
(図 9 タイマ Z インプットキャプチャ機能による通信転送レートの補正)

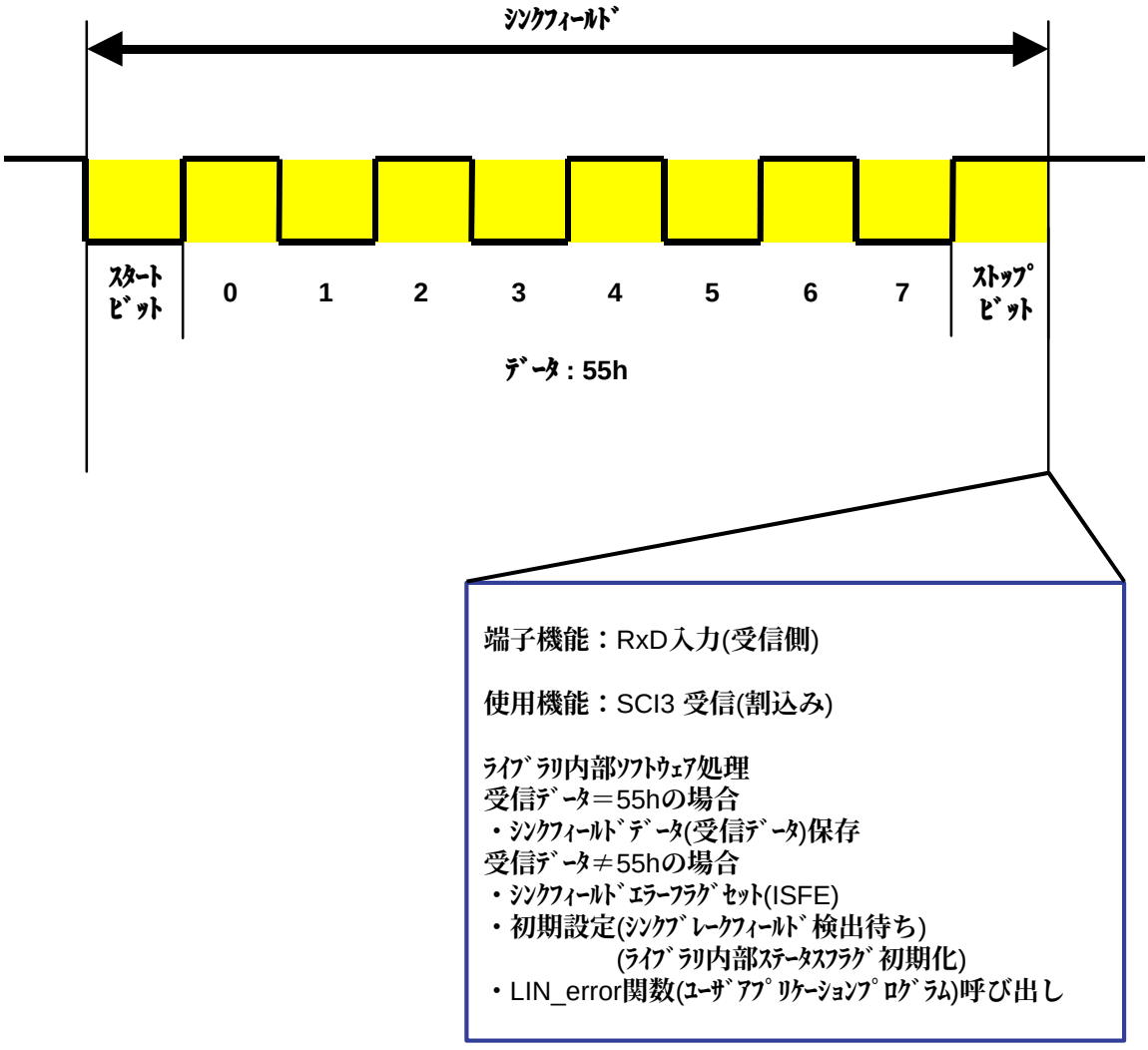


図 8 SCI3 受信機能によるシンクフィールドの受信と判定

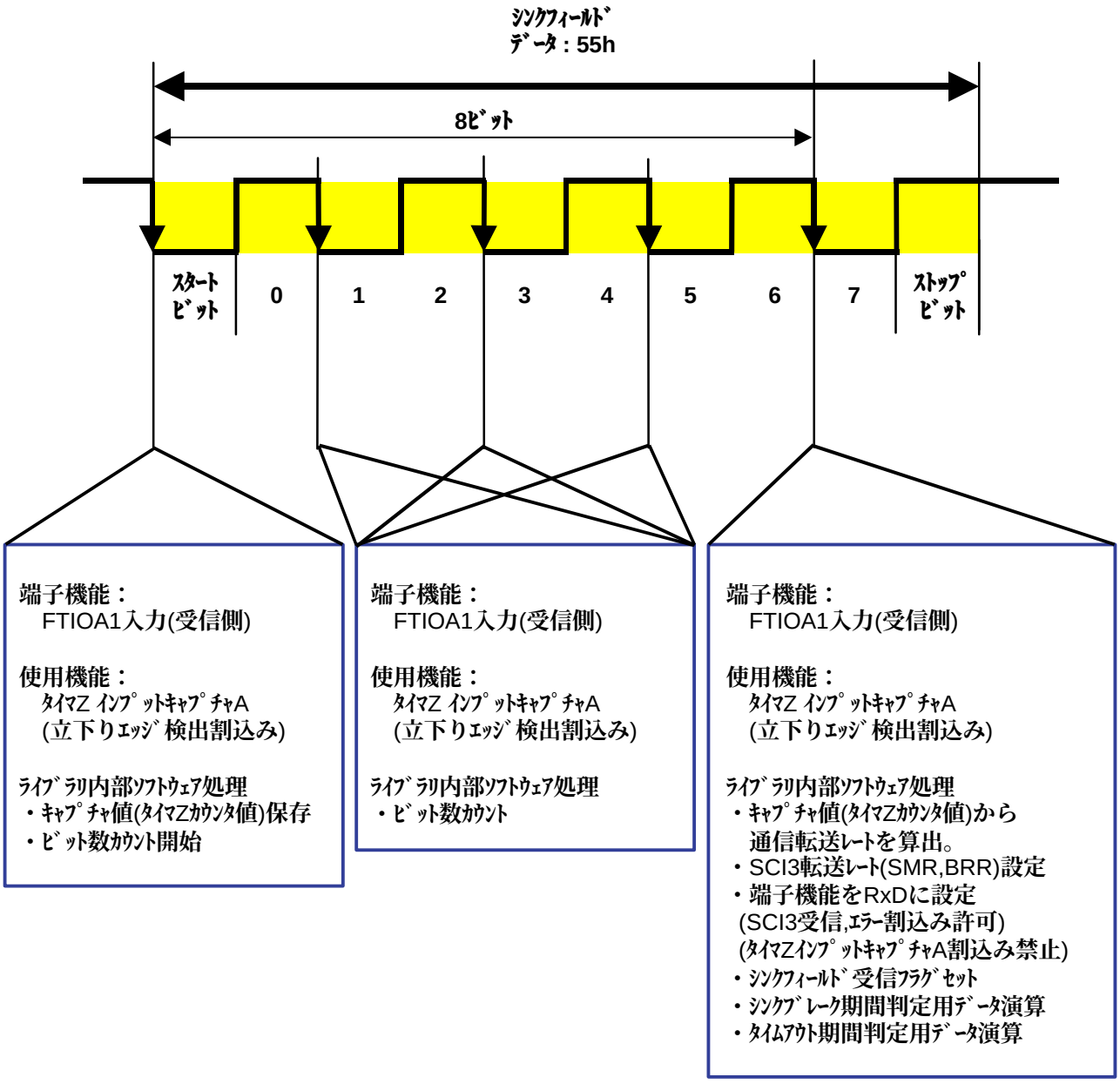


図 9　タイマＺインプットキャプチャ機能による通信転送レートの補正

2.5.1.3 ID フィールドの受信

SCI3 受信機能により、ID フィールド受信データ中の ID(DLC,パリティー含む)を判定します。

自ノードへのレスポンス送信要求 ID であった場合レスポンスフレームの送信を開始します。

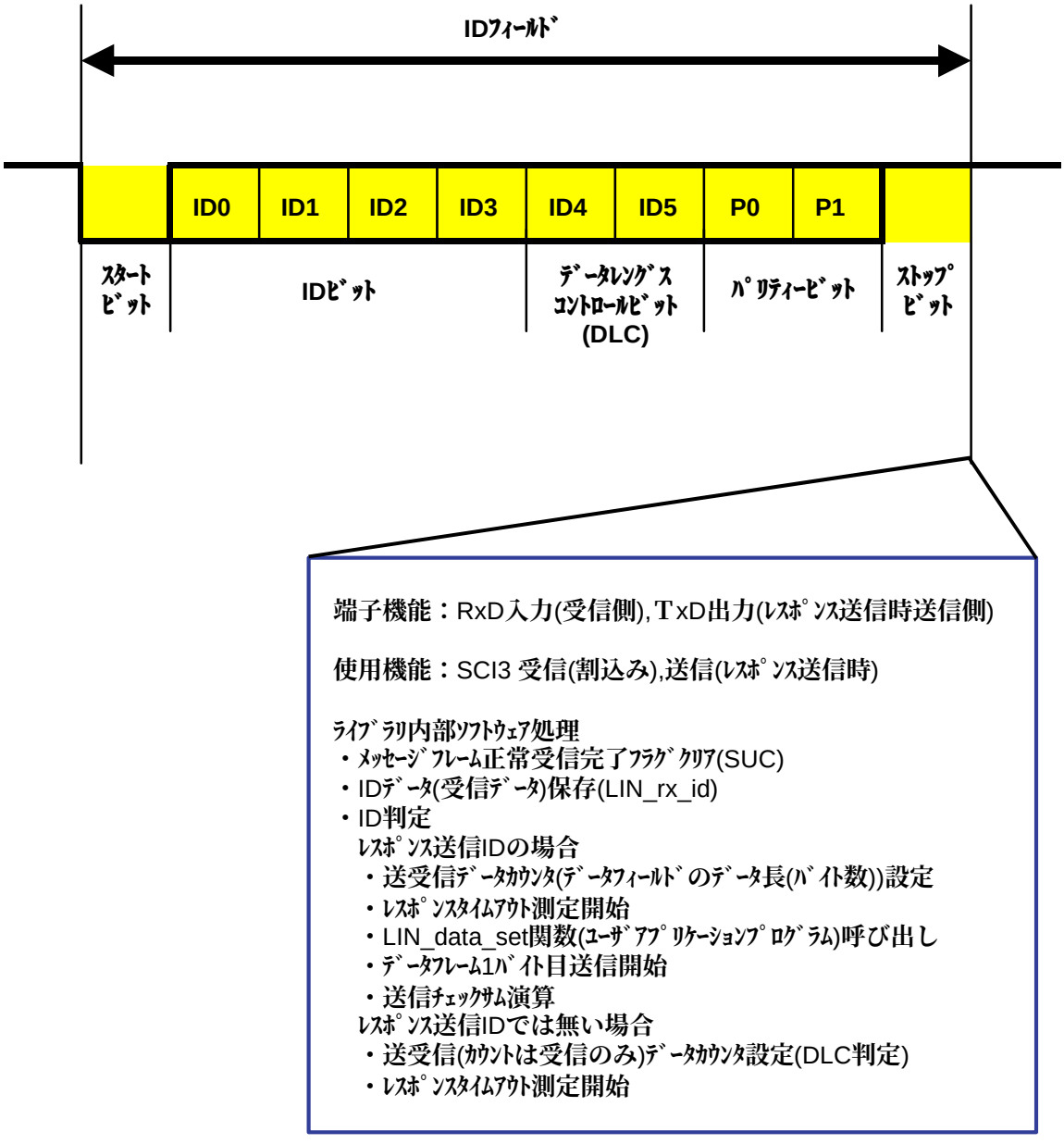


図 10 ID フィールドの受信と判定

2.5.2 レスポンスフレームの送受信

2.5.2.1 データフィールドの送受信(チェックサムフィールドの送信)

SCI3 受信機能により受信したデータを保存し、受信チェックサムデータ演算を行います。

レスポンス送信時は、次データを送信し、送信チェックサムデータ演算を行います。(受信割込み内)

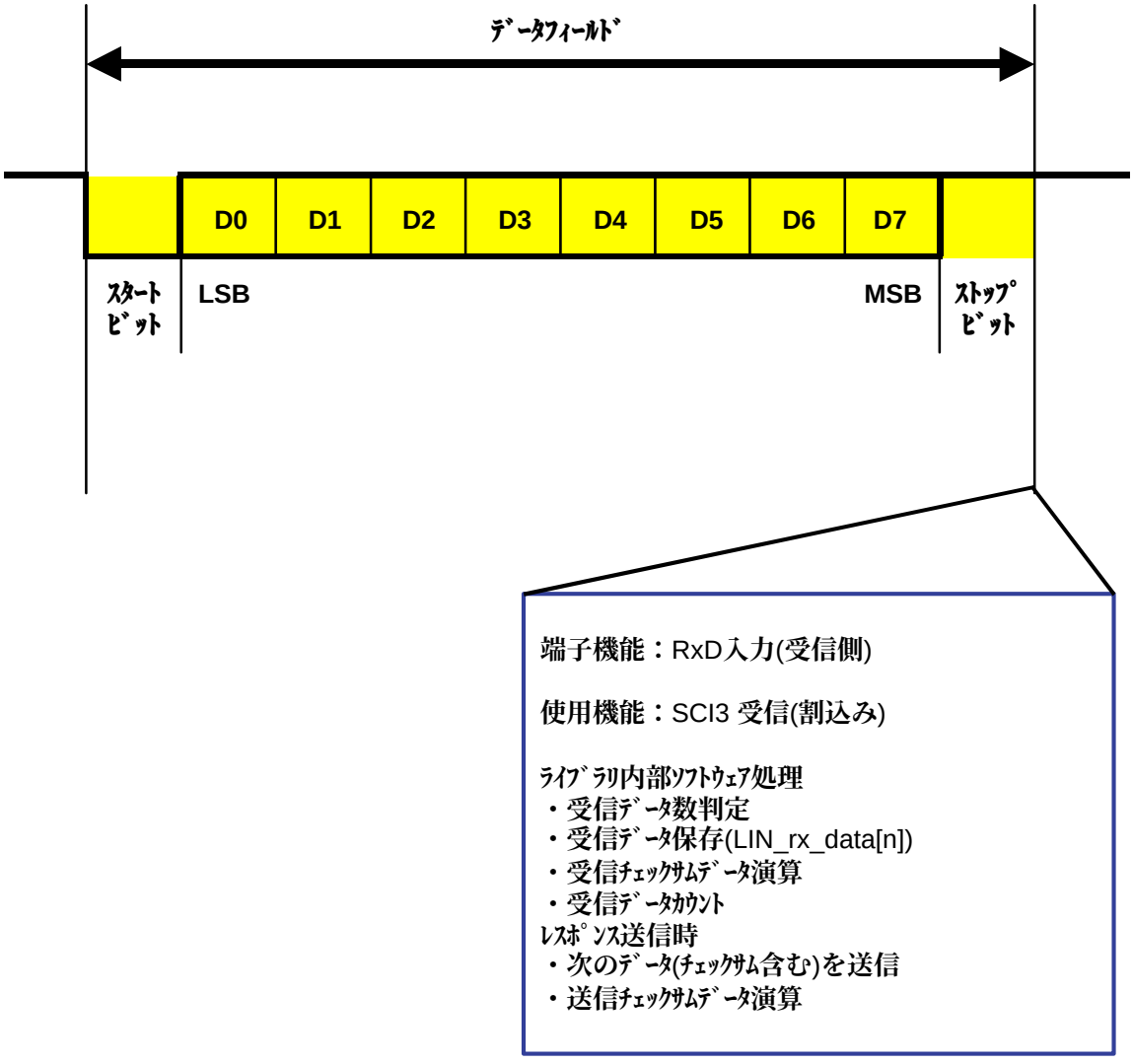
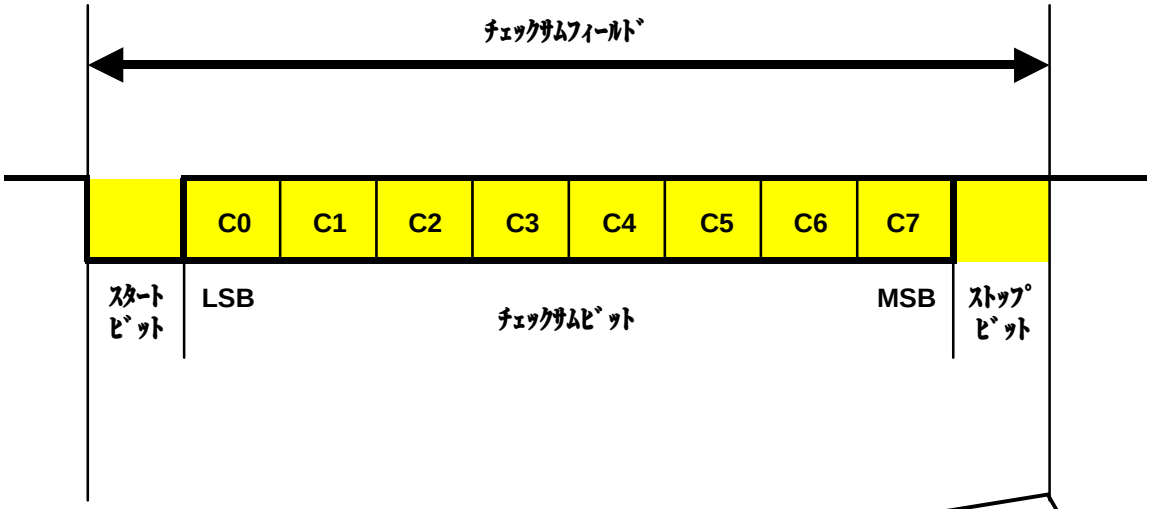


図 11 データフィールドの送受信とチェックサムデータの送信

2.5.2.2 チェックサムフィールドの受信

SCI3 受信機能により受信したチェックサムフィールドデータと、データフィールドから演算した受信チェックサムデータにより判定を行います。



端子機能：RxD入力(受信側)

使用機能：SCI3 受信(割込み)

ﾗｲﾌﾞﾗﾘ内部ｿﾌﾄｳｪｱ処理
・受信ﾃﾞｰﾀ数判定
・ﾁｪｯｸｻﾑ判定
・初期設定(ｼﾝｸﾌﾞﾚｰｸﾌｨｰﾙﾄﾞ検出待ち)
OKの場合
・ﾒｯｾｰｼﾞﾌﾚｰﾑ正常受信完了ﾌﾗｸﾞｾｯﾄ(SUC)
・ｽﾘｰﾌﾟｺﾏﾝﾄﾞ判定
ｽﾘｰﾌﾟｺﾏﾝﾄﾞの場合
・ｽﾘｰﾌﾟﾌﾗｸﾞｾｯﾄ(SLEEP)
・LIN_sleep関数(ﾕｰｻﾞｱﾌﾟﾘｹｰｼｮﾝﾌﾟﾛｸﾞﾗﾑ)呼び出し
ｽﾘｰﾌﾟｺﾏﾝﾄﾞでは無い(通常のﾒｯｾｰｼﾞﾌﾚｰﾑ)場合
・LIN_end関数(ﾕｰｻﾞｱﾌﾟﾘｹｰｼｮﾝﾌﾟﾛｸﾞﾗﾑ)呼び出し
NGの場合
・ﾁｪｯｸｻﾑｴﾗｰﾌﾗｸﾞｾｯﾄ(CSE)
・LIN_error関数(ﾕｰｻﾞｱﾌﾟﾘｹｰｼｮﾝﾌﾟﾛｸﾞﾗﾑ)呼び出し

図 12 チェックサムフィールドの受信と判定

2.5.3 ウェイクアップ信号の送受信

SCI3 送信機能によりウェイクアップ信号(送信データ：80h)を送信します。

IRQ0 立下りエッジ検出機能により他ノードからのウェイクアップ信号を検出します。

2.5.3.1 ウェイクアップ信号の送信

LINID.h 内定義文(#define __T_WAKEUP __ON)によりコンパイル時にウェイクアップ信号送信機能が組み込まれ、ユーザアプリケーションプログラムから **LIN_transmit_wake_up** 関数を呼び出す事で SCI3 送信機能によりウェイクアップ信号を送信します。本ライブラリでは、ウェイクアップデリミタ出力制御は行っておりません。

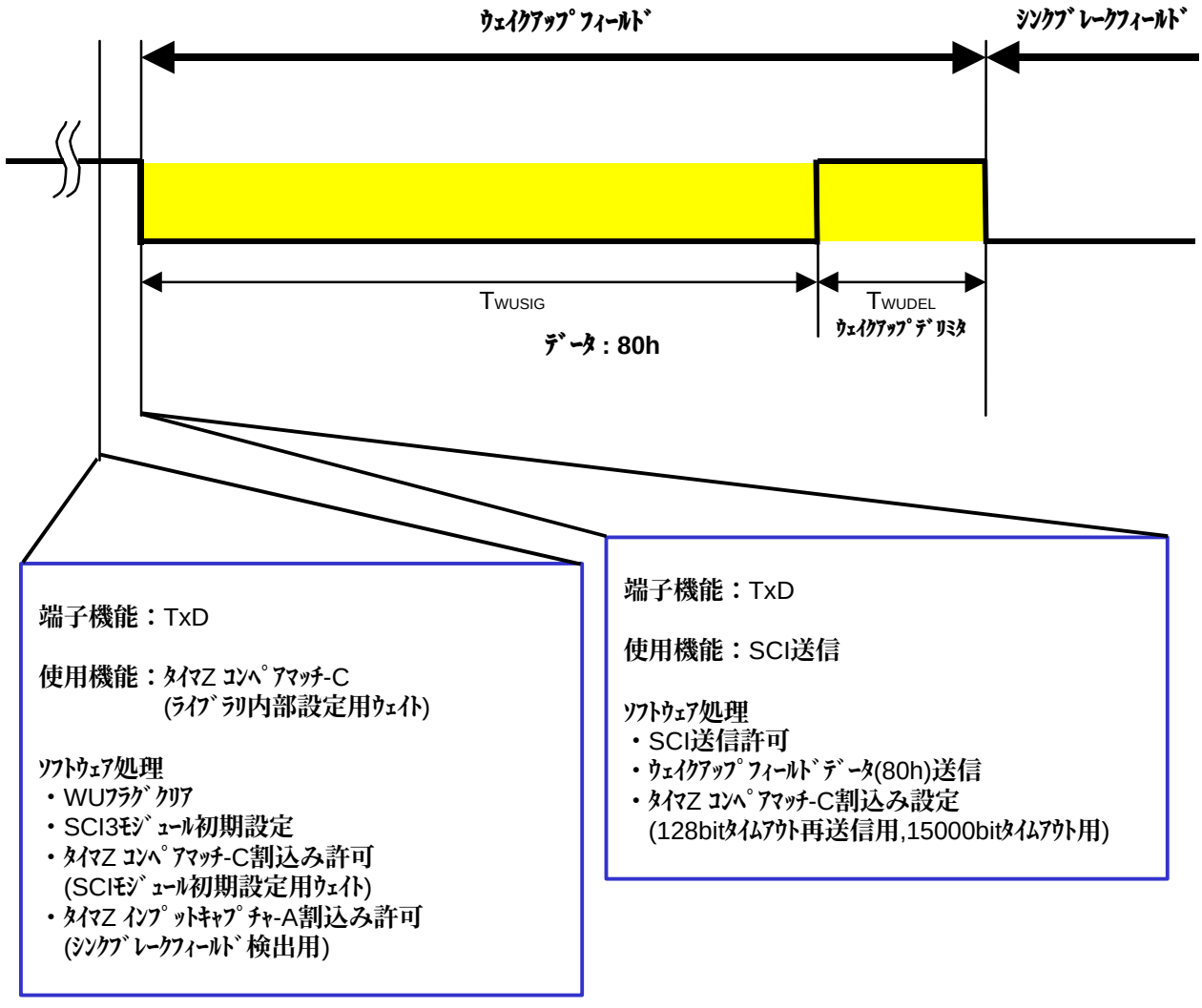


図 13 ウェイクアップ信号の送信

2.5.3.2 ウェイクアップ信号の受信

LINID.h 内定義文(#define __R_WAKEUP __ON)によりコンパイル時にウェイクアップ信号受信機能が組み込まれ、ユーザアプリケーションプログラムから **LIN_wake_up_PR** 関数を呼び出す事で IRQ 立下りエッジ検出機能による他ノードからのウェイクアップ信号の受信待ち状態となります。

本ライブラリは、ウェイクアップフィールドデータの検証は行わず、立下りエッジのみの検出となります。

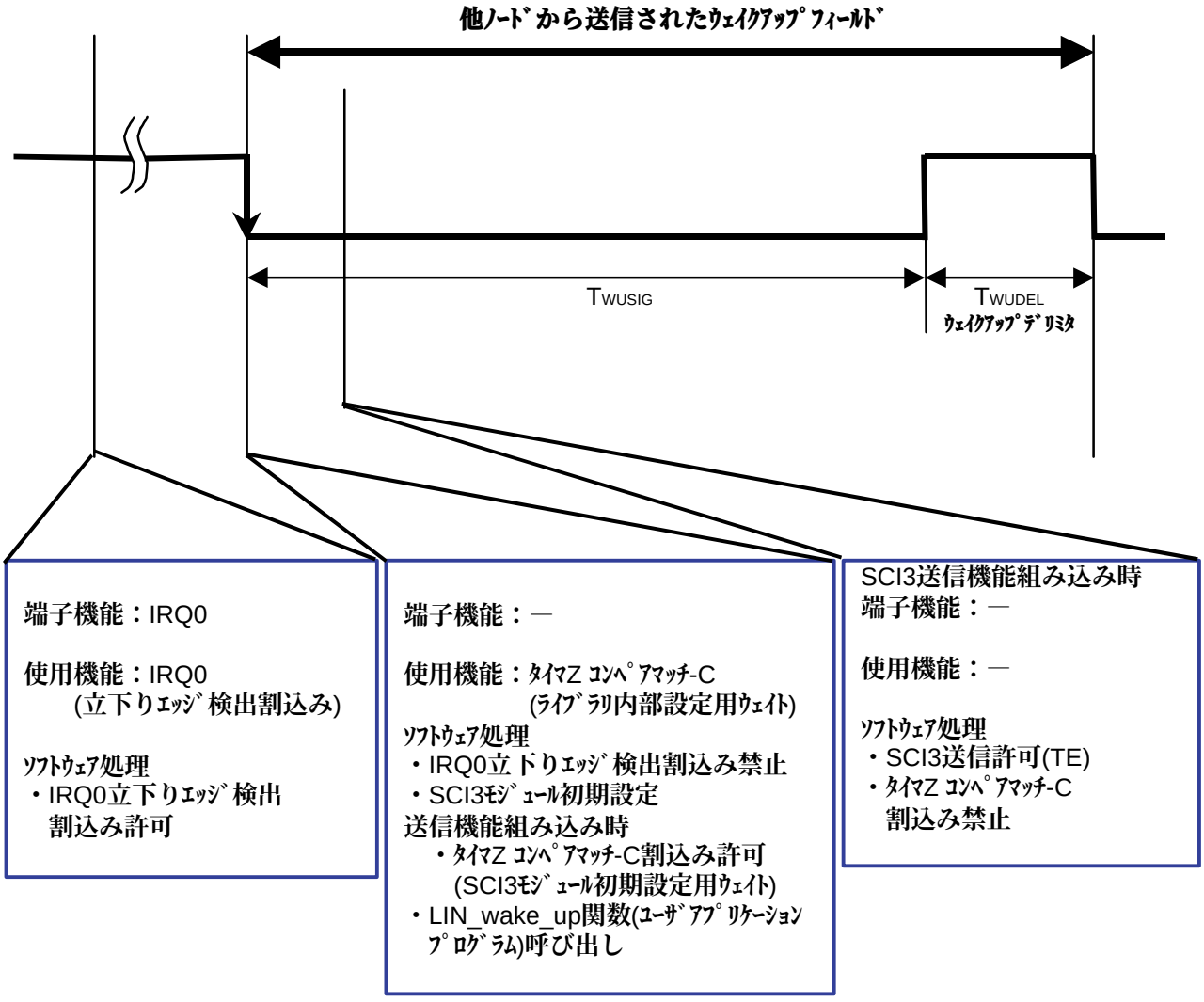


図 14 ウェイクアップ信号の受信

2.5.4 スリープコマンドの受信

メッセージフレームを正常受信完了後、受信した ID フィールドデータが 3Ch 且つレスポンスデータ 1 バイト目が 00h であった場合スリープコマンドと認識し、スリープフラグ(SLEEP)をセットし、LIN_sleep 関数(ユーザアプリケーションプログラム)を呼び出します。(その際のメッセージフレームでは LIN_end 関数呼び出しはありません)

2.6 ソフトウェア説明

以下に本ライブラリソフトウェアの説明を示します。

2.6.1 ヘッダファイルの組み込み

標準ライブラリ(machine.h)、LIN ライブラリ用定義ファイル(LINID.h)、マイコン内蔵周辺レジスタ定義ファイル(H8_3687f.h)の組み込みを行います。

```
#include      <machine.h>

#define       __LIN_LIB
#include      "LINID.h"

#include      "H8_3687f.h"
```

2.6.2 関数定義

ライブラリ内の関数(モジュール)定義を行います。

LIN_data_set 関数は、LINID.h 内の__Res2byte_ID、__Res4byte_ID、__Res8byte_ID のいずれかの定義により関数の組み込みを選択されます。

LIN_transmit_wake_up 関数は、__T_WAKEUP 定義により組み込みを選択されます。

LIN_intc_init 関数、LIN_wake_up 関数、LIN_wake_up_PR 関数は、__R_WAKEUP 定義により組み込みを選択されます。

```
void      LIN_initialize(void);
void      LIN_system_init(void);
void      LIN_port_init(void);
void      LIN_sci_init(void);
void      LIN_timerZ_init(void);
void      LIN_Sflag_init(void);
void      LIN_end(void);
void      LIN_sleep(void);
void      LIN_error(void);
void      LIN_break_reception_PR(void);
void      LIN_start(void);
void      LIN_stop(void);

#if       __RESPONSE      ==      __ON
void      LIN_data_set(void);
#endif

#if       __T_WAKEUP      ==      __ON
void      LIN_transmit_wake_up(void);
#endif
```

```
#if     __R_WAKEUP     ==     __ON
void     LIN_intc_init(void);
void     LIN_wake_up(void);
void     LIN_wake_up_PR(void);
#endif
```

2.6.3 ライブラリ内部定数・変数定義

ライブラリ内部で使用する定数および変数を定義します。

表 5 ライブラリ内部定数・変数定義

ラベル名(変数名)	データ型	機能説明
t_1	unsigned short	1ビット長カウンタ値(SCI3初期設定時ウェイト用)
t_11	unsigned long	11ビット長カウンタ値 (シンクブレークフィールド検出用)
t_128	(構造体)	128ビット長カウンタ値 (ウェイクアップ送信時シンクブレークフィールド検出 タイムアウト用)
t_128.LONG	unsigned long	
t_128.WORD.h	unsigned short	
t_128.WORD.l	unsigned short	
t_15k	unsigned long	15000ビット長カウンタ値 (ウェイクアップリトライ送信(3回)後タイムアウト用) (LIN_status.BIT.TOA3B)
t_25k	unsigned long	25000ビット長カウンタ値(ノーバスアクティブ検出用) (LIN_status.BIT.NBA)
flame_max_2	(構造体)	レスポンスタイムアウトMAX値 (LIN_status.BIT.SNRE)
flame_max_2.LONG	unsigned long	
flame_max_2.WORD.h	unsigned short	
flame_max_2.WORD.l	unsigned short	
flame_max_4	(構造体)	
flame_max_4.LONG	unsigned long	
flame_max_4.WORD.h	unsigned short	
flame_max_4.WORD.l	unsigned short	
flame_max_8	(構造体)	
flame_max_8.LONG	unsigned long	
flame_max_8.WORD.h	unsigned short	
flame_max_8.WORD.l	unsigned short	
baudrate	(構造体)	SCI3モジュール用ボーレート設定値
baudrate.WORD	unsigned short	
baudrate.BYTE.smr	unsigned char	
baudrate.BYTE.brr	unsigned char	
ex_counter	(構造体)	タイマW拡張カウンタ
ex_counter.LONG	unsigned long	
ex_counter.WORD.h	unsigned short	
ex_counter.WORD.l	unsigned short	
flame_max	unsigned short	レスポンスタイムアウト設定値 (タイマW オーバーフローカウント値)

表 6　ライブラリ内部定数・変数定義(続き)

ラベル名(変数名)	データ型	機能説明
counter	unsigned char	送受信データカウンタ
t_checksum	(構造体)	送信データチェックサム演算値
t_checksum.WORD	unsigned short	
t_checksum.BYTE.carry	unsigned char	
t_checksum.BYTE.data	unsigned char	
r_checksum	(構造体)	受信データチェックサム演算値
r_checksum.WORD	unsigned short	
r_checksum.BYTE.carry	unsigned char	
r_checksum.BYTE.data	unsigned char	
in_status	(構造体)	ライブラリ内部ステータス
in_status.BYTE	unsigned char	
in_status.BIT.wk7	ビット- 7	リザーブビット
in_status.BIT.sync_field	ビット- 6	シンクフィールド受信フラグ
in_status.BIT.wk5	ビット- 5	リザーブビット
in_status.BIT.r_id	ビット- 4	レスポンスID判定フラグ レスポンスデータ送信時： 1 受信時： 0
in_status.BIT.sci	ビット- 3	SCI3モジュール初期化フラグ
in_status.BIT.brr	ビット- 2	ボーレート補正フラグ
in_status.BIT.wu	ビット- 1～0	ウェイクアップ信号送信用フラグ (内部設定用送信カウンタ)

```
#if     __Corrects_baud_rate    ==    __ON
static    unsigned short    t_1;
static    unsigned long     t_11;
static    union {
              unsigned short    WORD;
              struct {
                   unsigned char    smr;
                   unsigned char    brr;
              }    BYTE;
}    baudrate;
#elif    __Corrects_baud_rate    ==    __OFF
const    unsigned short    t_1    =    t_1_data;
const    unsigned long     t_11   =    t_11_data;
const    union {
              unsigned short    WORD;
              struct {
                   unsigned char    smr;
                   unsigned char    brr;
              }    BYTE;
}    baudrate    =    baudrate_data;
#endif

const    unsigned long    t_25k    =    t_25k_data;
```

```
const	union {
		unsigned long	LONG;
		struct {
			unsigned short	h;
			unsigned short	l;
		}	WORD;
}	flame_max_2	=	t_2byte_data;
const	union {
		unsigned long	LONG;
		struct {
			unsigned short	h;
			unsigned short	l;
		}	WORD;
}	flame_max_4	=	t_4byte_data;
const	union {
		unsigned long	LONG;
		struct {
			unsigned short	h;
			unsigned short	l;
		}	WORD;
}	flame_max_8	=	t_8byte_data;
static	union {
		unsigned long	LONG;
		struct {
			unsigned short	h;
			unsigned short	l;
		}	WORD;
}	ex_counter;
static	unsigned short	flame_max;
static	unsigned char	counter;

#if	__T_WAKEUP	==	__ON
const	union {
		unsigned long	LONG;
		struct {
			unsigned short	h;
			unsigned short	l;
		}	WORD;
}	t_128	=	t_128_data;
const	unsigned long	t_15k	=	t_15k_data;
#endif
```

```
#if     __RESPONSE     ==     __ON
static     union {
                unsigned short     WORD;
                struct {
                    unsigned char     carry;
                    unsigned char     data;
                }    BYTE;
}     t_checksum;
#endif

static     union {
                unsigned short     WORD;
                struct {
                    unsigned char     carry;
                    unsigned char     data;
                }    BYTE;
}     r_checksum;
static     union {
                unsigned char     BYTE;
                struct {
                    unsigned char     wk7              :1;
                    unsigned char     sync_field       :1;
                    unsigned char     wk5              :1;
                    unsigned char     r_id             :1;
                    unsigned char     sci              :1;
                    unsigned char     brr              :1;
                    unsigned char     wu               :2;
                }    BIT;
}     in_status;
```

2.6.4 グローバル変数定義

ユーザアプリケーションプログラム、及びライブラリで共有する変数を定義します。

(表 4 参照)

```
#if     __RESPONSE     ==     __ON
volatile     unsigned char     LIN_tx_data[8];
#endif

volatile     unsigned char     LIN_rx_id;
volatile     unsigned char     LIN_rx_data[8];
volatile     union {
                    unsigned char     BYTE;
                    struct {
                         unsigned char     NBA       :1;
                         unsigned char     CSE       :1;
                         unsigned char     ISFE      :1;
                         unsigned char     TOA3B     :1;
                         unsigned char     SNRE      :1;
                         unsigned char     SCI       :1;
                         unsigned char     SUC       :1;
                         unsigned char     SLEEP     :1;
                    }    BIT;
}     LIN_status;
volatile     union {
                    unsigned char BYTE;
                    struct {
                         unsigned char     CBR     :1;
                         unsigned char     wk6     :1;
                         unsigned char     WU      :1;
                         unsigned char     wk4     :5;
                    }    BIT;
}     LIN_control;
```

2.6.5 初期設定用関数

LIN 通信制御に使用する H8/3687 内蔵周辺機能の初期設定、及びライブラリ内部で使用するソフトウェアフラグ等の初期化を行います。

注意：端子 P14(IRQ0),P21(RxD),P22(TxD),P64(FTIOA1)は LIN 通信で使用します。ユーザアプリケーションでポート 1,ポート 2,ポート 6 におけるその他の端子(P10～P12,P15～P17,P20,P23,P24,P60～P63,P65～P67)を使用する場合、下記ソースファイル内 LIN_port_init 関数における PCR2,PCR6、及び LIN_intc_init 関数における PCR1 の設定文により端子設定が変更されるおそれがあります。前述の端子をご使用の際には、各 PCR の設定をユーザアプリケーションプログラム内で行ったうえ、下記ソースファイル内の PCR1, PCR2, PCR8 の設定文を削除またはコメントアウトして下さい。

図 15　初期設定用関数フローチャート

```
void LIN_initialize(void){
      LIN_status.BYTE      =      0;
      LIN_system_init();
      LIN_port_init();

#if      __Corrects_baud_rate      ==      __ON
      t_1         =      t_1_data;
      t_11        =      t_11_data;
      baudrate.WORD      =        baudrate_data;
#endif

      LIN_timerZ_init();
      LIN_Sflag_init();
      LIN_sci_init();

#if      __R_WAKEUP      ==      __ON
      LIN_intc_init();
#endif

      ex_counter.WORD.h        =        0;
      LIN_control.BYTE      =      0;
}


void LIN_system_init(void){
      MSTCR1.BYTE      &=      0xDF;
      MSTCR2.BYTE      &=      0xFD;
}


void LIN_port_init(void){
#if         __R_WAKEUP      ==      __ON
      IO.PMR1.BYTE      |=      0x12;
#elif      __R_WAKEUP      ==      __OFF
      IO.PMR1.BYTE      |=      0x02;
#endif

      IO.PDR2.BIT.B2      =      1;
/*   IO.PCR2      =      0;              /* 注意:ポート 2,ポート 6 をユーザアプリケーションで使用する場合、          */
/*   IO.PCR6      =      0;              /*       システム保護の為 LIN で使用する端子の設定用ビットを “ 0”          */
                                        /*       (入力端子)にしてユーザアプリケーション側で設定し、本設定          */
                                        /*       文は削除して下さい                                                */
}
```

```
void LIN_sci_init(void){
    SCI3.SCR3.BYTE    =    0;
    SCI3.SMR.BYTE     =    baudrate.BYTE.smr;
    SCI3.BRR    =    baudrate.BYTE.brr;

#if    ((__RESPONSE   ==   __ON)   ||   (__T_WAKEUP   ==   __ON))
    TZ.GRC1    =    TZ.TCNT1    +    t_1;
    TZ.TSR1.BIT.IMFC    =    0;
    TZ.TIER1.BIT.IMIEC    =    1;
    in_status.BIT.sci    =    1;
#endif
}


void LIN_timerZ_init(void){
    TZ.TSTR.BIT.STR1    =    0;
    TZ.TCR1.BYTE    =    0x03;
    TZ.TMDR.BYTE    =    0x0E;
    TZ.TOER.BYTE    |=    0xF0;
    TZ.TPMR.BYTE    &=    0x8F;
    TZ.TIORA1.BYTE    =    0x8D;
    TZ.TIORC1.BYTE    =    0xF8;
    TZ.TSR1.BYTE    &=    0xC0;
    TZ.TIER1.BYTE    =    0x11;
    TZ.TSTR.BIT.STR1    =    1;
}


#if    __R_WAKEUP    ==    __ON
void LIN_intc_init(void){
/*  IO.PCR1    =    0;              /* 注意:ポート 1 をユーザアプリケーションで使用する場合、          */
                                    /*       システム保護の為 LIN で使用する端子の設定用ビットを “ 0” */
                                    /*       (入力端子)にしてユーザアプリケーション側で設定し、本設    */
                                    /*       定文は削除して下さい                                    */
    IEGR1.BIT.IEG0    =    0;
    IRR1.BIT.IRRI0    =    0;
    IENR1.BIT.IEN0    =    0;
}
#endif


void LIN_Sflag_init(void){
    counter    =    0;
    in_status.BYTE    =    0;
}
```

2.6.6 LIN 通信制御停止用関数

LIN 通信制御を停止し、接続されている LIN バス上の通信に関与しなくなります。

図 16 LIN 通信制御停止用関数フローチャート

```
void LIN_stop(void){
    SCI3.SSR.BYTE     &=     0x84;
    SCI3.SCR3.BYTE    =      0x00;
    TZ.TIER1.BYTE     &=     0xE0;

#if     __R_WAKEUP     ==     __ON
    IENR1.BIT.IEN0    =      0;
#endif
}
```

2.6.7 LIN 通信制御再開準備用関数

2.6.6 LIN 通信制御停止用関数等により停止状態となっている LIN 通信制御を再開し、シンクブレークフィールドの受信待ち状態となります。

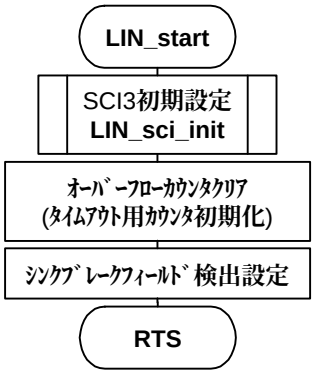


図 17 LIN 通信制御再開準備用関数フローチャート

```
void LIN_start(void){
    LIN_sci_init();
    ex_counter.WORD.h     =     0;
    TZ.TSR1.BYTE      &=      0xC0;
    TZ.TIER1.BYTE     |=      0x11;
}
```

2.6.8 ウェイクアップ信号送信用関数

ウェイクアップ信号を送信します。送信時より 128bit 期間内にシンクブレークフィールドの検出が無い場合、初回送信を含めて 3 回までリトライ送信を行います(タイマ Z 割込み関数内で送信制御)。3 回送信後、15000bit 期間内にシンクブレークフィールドの検出が無い場合には、タイムアウトフラグ(LIN_status.BIT.TOA3B)をセットし、LIN_error 関数(ユーザアプリケーションプログラム)を呼び出します。

図 18 ウェイクアップ信号送信用関数フローチャート

```
#if     __T_WAKEUP      ==      __ON
void LIN_transmit_wake_up(void){
      LIN_control.BIT.WU      =      0;
      in_status.BIT.wu      =      1;

#if     __R_WAKEUP      ==      __ON
      IENR1.BIT.IEN0      =      0;
#endif

      LIN_start();
}
#endif
```

2.6.9 ウェイクアップ信号受信準備用関数

他ノードからのウェイクアップ信号受信準備を行います。

図 19 ウェイクアップ信号受信準備用関数フローチャート

```
#if     __R_WAKEUP     ==     __ON
void LIN_wake_up_PR(void){
      IRR1.BIT.IRRI0     =     0;
      IENR1.BIT.IEN0     =     1;
}
#endif
```

2.6.10 ライブラリ内部シンクブレークフィールド検出準備用関数

メッセージフレーム送受信完了時、拡張フレーム ID 受信時、エラー検出時にライブラリ内部で次のメッセージフレームの受信準備(シンクブレークフィールド検出準備)を行います。

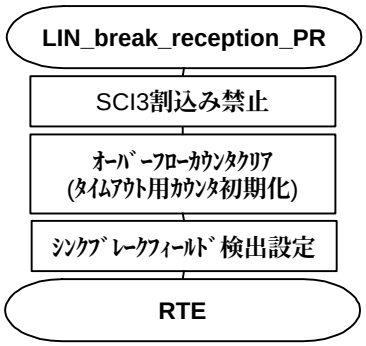


図 20 シンクブレークフィールド検出受信準備用関数

```
void LIN_break_reception_PR(void){
      SCI3.SSR.BYTE      &=      0x84;
#if    (( __RESPONSE   ==   __ON)   ||   (__T_WAKEUP   ==   __ON))
      SCI3.SCR3.BYTE      =      0x20;
#else
      SCI3.SCR3.BYTE      =      0x00;
#endif
      ex_counter.WORD.h      =      0;
      TZ.TSR1.BYTE      &=      0xC0;
      TZ.TIER1.BYTE      =      0x11;
}
```

2.6.11 IRQ 割込み関数

IRQ0 立下りエッジ検出割込み処理を行います。2.6.9 ウェイクアップ信号受信準備用関数による設定後、LIN バス上の立下りエッジを検出し、LIN 通信制御の準備を行います。

図 21 IRQ 割込み関数フローチャート

```
#if     __R_WAKEUP      ==      __ON
#pragma     interrupt(IRQ0_int)
void        IRQ0_int(void){
    LIN_status.BIT.SLEEP     =     0;
    IRR1.BIT.IRRI0     =     0;
    IENR1.BIT.IEN0     =     0;
#if     __Corrects_baud_rate     ==     __ON
    t_1         =     t_1_data;
    t_11        =     t_11_data;
    baudrate.WORD     =     baudrate_data;
#endif
    LIN_start();
    LIN_wake_up();
}
#endif
```

2.6.12 タイマ Z 割込み関数

タイマ Z(channel-1)のインプットキャプチャ-A 割込み(I/C-A)、コンペアマッチ-B 割込み(O/C-B)、コンペアマッチ-C 割込み(O/C-C)、及びオーバーフロー割込み(OVF)処理を行います。

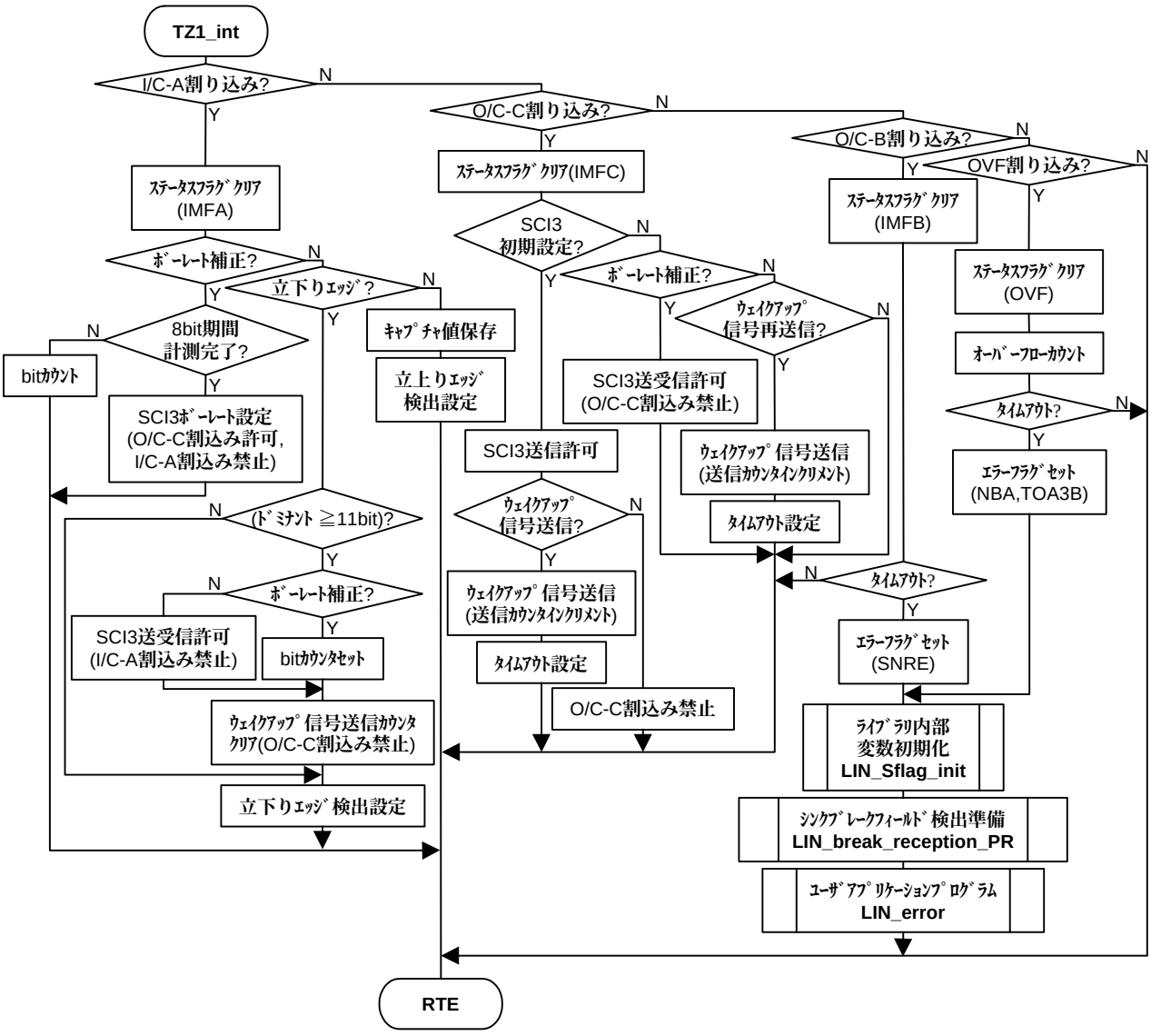


図 22 タイマ Z(channel-1)割込み関数フローチャート

```
#pragma       interrupt(TZ1_int)
void       TZ1_int(void){
      unsigned long       i;
      unsigned short       N,w;

      if((TZ.TSR1.BIT.IMFA)    &&    (TZ.TIER1.BIT.IMIEA)){
            TZ.TSR1.BIT.IMFA      =      0;
```

```
#if     __Corrects_baud_rate     ==     __ON
          if(in_status.BIT.brr   ==   0){
#endif

               if(TZ.TIORA1.BIT.IOA0){
                    TZ.TIORA1.BIT.IOA0     =     0;
                    ex_counter.LONG     =     (unsigned long)TZ.GRA1;
                    TZ.GRB1     =     TZ.GRA1;
                    if((TZ.TSR1.BIT.OVF)   &&   (ex_counter.WORD.l   <   0x00FF)){
                         TZ.TSR1.BIT.OVF     =     0;
                    }
               }else{
                    w     =     ex_counter.WORD.l;
                    ex_counter.WORD.l     =     TZ.GRA1;
                    if((TZ.TSR1.BIT.OVF)   &&   (ex_counter.WORD.l   <   0x00FF)){
                         ex_counter.WORD.h     +=     1;
                         TZ.TSR1.BIT.OVF     =     0;
                    }
                    ex_counter.LONG     -=     w;
                    if(ex_counter.LONG   >=   t_11){

#if     __Corrects_baud_rate     ==     __ON
                         if(LIN_control.BIT.CBR){
                              in_status.BIT.brr     =     1;
                              LIN_control.BIT.CBR     =     0;
                              counter     =     4;
                         }else{
#endif

                              SCI3.SSR.BYTE     &=     0x84;

#if        __RESPONSE     ==     __ON
                              SCI3.SCR3.BYTE     =     0x70;
#elif     __RESPONSE     ==     __OFF
                              SCI3.SCR3.BYTE     =     0x50;
#endif

                              TZ.TIER1.BIT.IMIEA     =     0;

#if     __Corrects_baud_rate     ==     __ON
                         }
#endif

#if     __T_WAKEUP     ==     __ON
                         in_status.BIT.wu     =     0;
                         TZ.TIER1.BIT.IMIEC     =     0;
#endif

                    }
                    TZ.TIORA1.BIT.IOA0     =     1;
               }
```

```
#if     __Corrects_baud_rate     ==     __ON
        }else{
            if(counter){
                if(counter  ==  4){
                    ex_counter.LONG     =     (unsigned long)TZ.GRA1;
                    SCI3.SCR3.BYTE     =     0;
                    SCI3.SMR.BYTE     =     0;
                    if((TZ.TSR1.BIT.OVF)  &&  (ex_counter.WORD.l  <  0x00FF)){
                        TZ.TSR1.BIT.OVF     =     0;
                    }
                }
                counter     -=     1;
            }else{
                TZ.TIER1.BYTE     =     0xF4;
                w     =     ex_counter.WORD.l;
                ex_counter.WORD.l     =     TZ.GRA1;
                if((TZ.TSR1.BIT.OVF)  &&  (ex_counter.WORD.l  <  0x00FF)){
                    ex_counter.WORD.h     +=     1;
                    TZ.TSR1.BIT.OVF     =     0;
                }
                t_1     =     (ex_counter.LONG     -     w)     >>     3;
                for(N     =     ((t_1  +  2)  >>  2);  N  >  0x0100;  N  >>=  2){
                    SCI3.SMR.BIT.CKS     +=     1;
                }
                SCI3.BRR     =     N     -     1;
                TZ.GRC1     =     (TZ.TCNT1     +     t_1);
                TZ.TSR1.BYTE     &=     0xF0;
                ex_counter.WORD.h     =     0;
                in_status.BIT.sync_field     =     1;
                t_11     =     t_1     *     11;
            }
        }
#endif

    }else if((TZ.TSR1.BIT.IMFC)  &&  (TZ.TIER1.BIT.IMIEC)){
        TZ.TSR1.BIT.IMFC     =     0;
        if(in_status.BIT.sci){
            SCI3.SSR.BYTE     &=     0x84;
            SCI3.SCR3.BIT.TE     =     1;

#if     __T_WAKEUP     ==     __ON
            if(in_status.BIT.wu  ==  1){
                TZ.GRC1     =     TZ.TCNT1     +     t_128.WORD.l;
                SCI3.TDR     =     0x80;
                in_status.BIT.wu     +=     1;
            }else{
#endif

                TZ.TIER1.BIT.IMIEC     =     0;

#if     __T_WAKEUP     ==     __ON
            }
#endif
```

```
                in_status.BIT.sci       =       0;

#if     __Corrects_baud_rate      ==      __ON
          }else if(in_status.BIT.brr){
                SCI3.SSR.BYTE     &=      0x84;

#if     ((__RESPONSE   ==   __ON)   ||   (__T_WAKEUP   ==   __ON))
                SCI3.SCR3.BYTE    =       0x70;
#else
                SCI3.SCR3.BYTE    =       0x50;
#endif

                TZ.TIER1.BIT.IMIEC      =       0;
#endif

#if     __T_WAKEUP      ==      __ON
          }else if((in_status.BIT.wu   ==   2)   &&   (ex_counter.WORD.h   >=   t_128.WORD.h)){
                SCI3.TDR      =       0x80;
                ex_counter.WORD.h       =       0;
                TZ.GRC1       =       TZ.TCNT1      +       t_128.WORD.l;
                in_status.BIT.wu        +=      1;
          }else if((in_status.BIT.wu   ==   3)   &&   (ex_counter.WORD.h   >=   t_128.WORD.h)){
                SCI3.TDR      =       0x80;
                ex_counter.WORD.h       =       0;
                TZ.TIER1.BIT.IMIEC      =       0;
#endif
          }
     }else if((TZ.TSR1.BIT.IMFB)   &&   (TZ.TIER1.BIT.IMIEB)){
          TZ.TSR1.BIT.IMFB      =       0;
          if(ex_counter.WORD.h   >=   flame_max){
                LIN_status.BIT.SNRE       =       1;
                LIN_Sflag_init();
                LIN_break_reception_PR();
                LIN_error();
          }
     }else if((TZ.TSR1.BIT.OVF)   &&   (TZ.TIER1.BIT.OVIE)){
          TZ.TSR1.BIT.OVF       =       0;
          ex_counter.WORD.h       +=      1;
          if((ex_counter.LONG   >   t_25k)){
                LIN_status.BIT.NBA      =       1;
                LIN_Sflag_init();
                LIN_break_reception_PR();
                LIN_error();

#if     __T_WAKEUP      ==      __ON
          }else if((ex_counter.LONG   >=   t_15k)   &&   (in_status.BIT.wu   ==   3)){
                in_status.BIT.wu = 0;
                LIN_status.BIT.TOA3B      =       1;
                LIN_error();
#endif
          }
     }
}
```

2.6.13 SCI3 割込み関数

SCI3 のエラー検出割込み、及び受信割込み処理を行います。

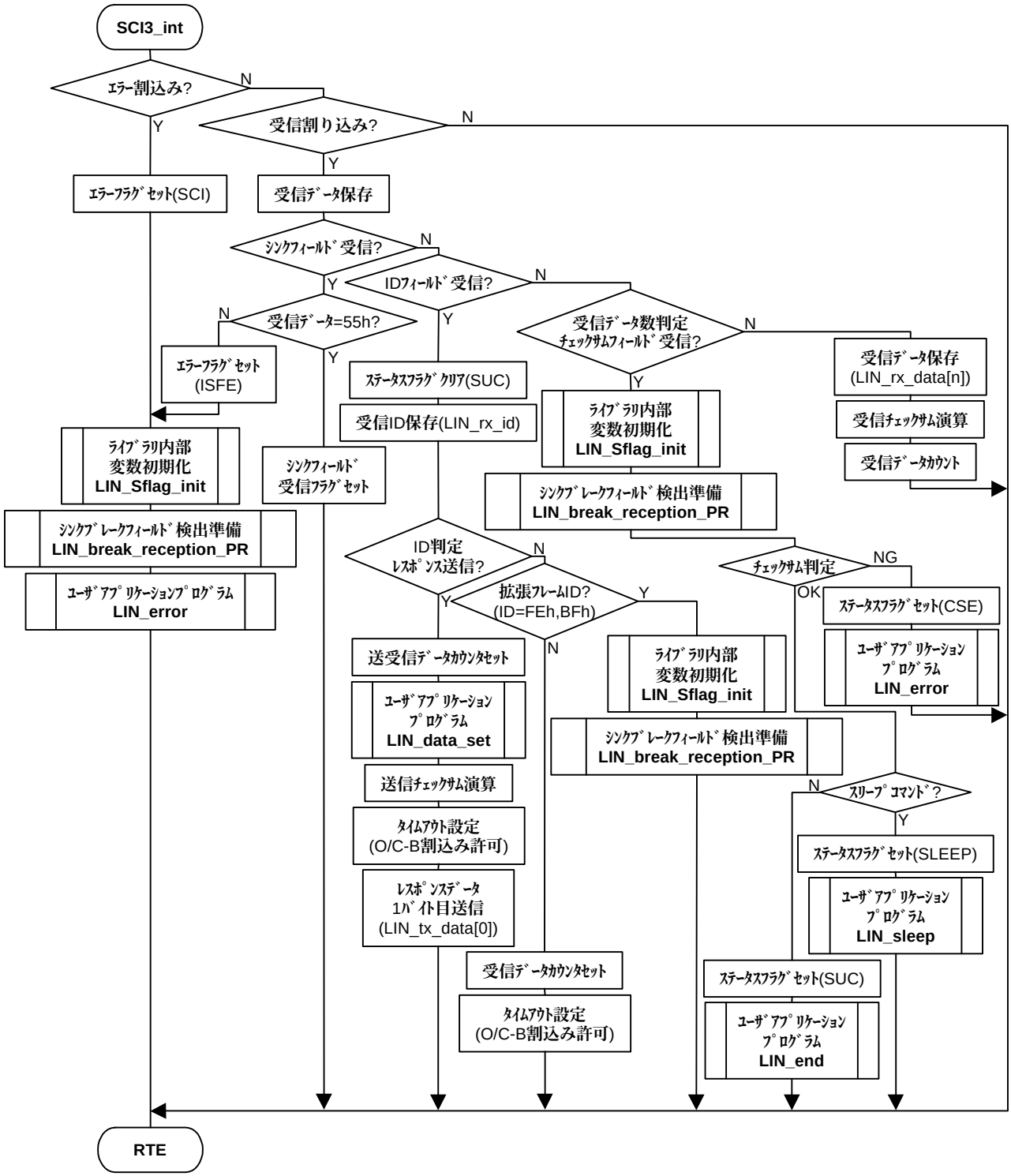


図 23 SCI3 割込み関数フローチャート

```
#pragma      interrupt(SCI3_int)
void      SCI3_int(void){
     unsigned char      buff,nmbr,nm,id,dlc;

     if(SCI3.SSR.BYTE   &   0x38){
          LIN_status.BIT.SCI      =      1;
          LIN_Sflag_init();
          LIN_break_reception_PR();
          LIN_error();
     }else if(SCI3.SSR.BIT.RDRF){
          buff      =      SCI3.RDR;
          if(in_status.BIT.sync_field){
               if(counter){
                    nm      =      counter      &      0x0F;
                    nmbr      =      (counter      >>      4)      -      nm;
                    if(nm){
                         LIN_rx_data[nmbr]      =      buff;
                         r_checksum.WORD      +=      (unsigned short)LIN_rx_data[nmbr];
                         r_checksum.BYTE.data      +=      r_checksum.BYTE.carry;
                         r_checksum.BYTE.carry      =      0;
                         counter      -=      1;

#if      __RESPONSE      ==      __ON
                         if(in_status.BIT.r_id){
                              if(nm   -   1){
                                   buff      =      LIN_tx_data[(nmbr   +   1)];
                                   SCI3.TDR      =      buff;
                                   t_checksum.WORD      +=      buff;
                                   t_checksum.BYTE.data      +=      t_checksum.BYTE.carry;
                                   t_checksum.BYTE.carry      =      0;
                              }else{
                                   t_checksum.BYTE.data      =      ~(t_checksum.BYTE.data);
                                   SCI3.TDR      =      t_checksum.BYTE.data;
                                   in_status.BIT.r_id      =      0;
                              }
                         }
#endif

                    }else{
                         LIN_Sflag_init();
                         LIN_break_reception_PR();
                         if((r_checksum.BYTE.data   ^   buff)   !=   0xFF){
                              LIN_status.BIT.CSE      =      1;
                              LIN_error();
                         }else{
                              if((LIN_rx_id   ==   0x3C)   &&   (LIN_rx_data[0]   ==   0)){
                                   LIN_status.BIT.SLEEP      =      1;
                                   LIN_sleep();
                              }else{
                                   LIN_status.BIT.SUC      =      1;
                                   LIN_end();
                              }
```

```
                        }
                    }
                }else{
                    in_status.BYTE      &=      0x40;
                    LIN_status.BIT.SUC      =       0;
                    LIN_rx_id       =       buff;
                    switch(LIN_rx_id){
#if     __Res2byte_ID       ==      __ON
#ifdef      __ID00
                        case        __ID00:
#endif
#ifdef      __ID01
                        case        __ID01:
#endif
#ifdef      __ID02
                        case        __ID02:
#endif
#ifdef      __ID03
                        case        __ID03:
#endif
#ifdef      __ID04
                        case        __ID04:
#endif
#ifdef      __ID05
                        case        __ID05:
#endif
#ifdef      __ID06
                        case        __ID06:
#endif
#ifdef      __ID07
                        case        __ID07:
#endif
#ifdef      __ID08
                        case        __ID08:
#endif
#ifdef      __ID09
                        case        __ID09:
#endif
#ifdef      __ID0a
                        case        __ID0a:
#endif
#ifdef      __ID0b
                        case        __ID0b:
#endif
#ifdef      __ID0c
                        case        __ID0c:
#endif
#ifdef      __ID0d
                        case        __ID0d:
#endif
```

```
#ifdef    __ID0e
                    case    __ID0e:
#endif
#ifdef    __ID0f
                    case    __ID0f:
#endif
#ifdef    __ID10
                    case    __ID10:
#endif
#ifdef    __ID11
                    case    __ID11:
#endif
#ifdef    __ID12
                    case    __ID12:
#endif
#ifdef    __ID13
                    case    __ID13:
#endif
#ifdef    __ID14
                    case    __ID14:
#endif
#ifdef    __ID15
                    case    __ID15:
#endif
#ifdef    __ID16
                    case    __ID16:
#endif
#ifdef    __ID17
                    case    __ID17:
#endif
#ifdef    __ID18
                    case    __ID18:
#endif
#ifdef    __ID19
                    case    __ID19:
#endif
#ifdef    __ID1a
                    case    __ID1a:
#endif
#ifdef    __ID1b
                    case    __ID1b:
#endif
#ifdef    __ID1c
                    case    __ID1c:
#endif
#ifdef    __ID1d
                    case    __ID1d:
#endif
#ifdef    __ID1e
                    case    __ID1e:
#endif
```

```
#ifdef	__ID1f
		case	__ID1f:
#endif
			counter	=	0x22;
			in_status.BIT.r_id	=	1;
			r_checksum.WORD	=	0;
			LIN_data_set();
			buff	=	LIN_tx_data[0];
			t_checksum.WORD	=	(unsigned short)buff;
			TZ.GRB1	+=	flame_max_2.WORD.l;
			flame_max	=	flame_max_2.WORD.h;
			TZ.TSR1.BIT.IMFB	=	0;
			TZ.TIER1.BIT.IMIEB	=	1;
			SCI3.TDR	=	buff;
			break;
#endif
#if	__Res4byte_ID	==	__ON
#ifdef	__ID20
		case	__ID20:
#endif
#ifdef	__ID21
		case	__ID21:
#endif
#ifdef	__ID22
		case	__ID22:
#endif
#ifdef	__ID23
		case	__ID23:
#endif
#ifdef	__ID24
		case	__ID24:
#endif
#ifdef	__ID25
		case	__ID25:
#endif
#ifdef	__ID26
		case	__ID26:
#endif
#ifdef	__ID27
		case	__ID27:
#endif
#ifdef	__ID28
		case	__ID28:
#endif
#ifdef	__ID29
		case	__ID29:
#endif
#ifdef	__ID2a
		case	__ID2a:
#endif
```

```
#ifdef    __ID2b
                    case    __ID2b:
#endif
#ifdef    __ID2c
                    case    __ID2c:
#endif
#ifdef    __ID2d
                    case    __ID2d:
#endif
#ifdef    __ID2e
                    case    __ID2e:
#endif
#ifdef    __ID2f
                    case    __ID2f:
#endif
                        counter     =     0x44;
                        in_status.BIT.r_id     =     1;
                        r_checksum.WORD     =     0;
                        LIN_data_set();
                        buff     =     LIN_tx_data[0];
                        t_checksum.WORD     =     (unsigned short)buff;
                        TZ.GRB1     +=     flame_max_4.WORD.l;
                        flame_max     =     flame_max_4.WORD.h;
                        TZ.TSR1.BIT.IMFB     =     0;
                        TZ.TIER1.BIT.IMIEB     =     1;
                        SCI3.TDR     =     buff;
                        break;
#endif
#if    __Res8byte_ID    ==    __ON
#ifdef    __ID30
                    case    __ID30:
#endif
#ifdef    __ID31
                    case    __ID31:
#endif
#ifdef    __ID32
                    case    __ID32:
#endif
#ifdef    __ID33
                    case    __ID33:
#endif
#ifdef    __ID34
                    case    __ID34:
#endif
#ifdef    __ID35
                    case    __ID35:
#endif
#ifdef    __ID36
                    case    __ID36:
#endif
```

```
#ifdef    __ID37
                    case    __ID37:
#endif
#ifdef    __ID38
                    case    __ID38:
#endif
#ifdef    __ID39
                    case    __ID39:
#endif
#ifdef    __ID3a
                    case    __ID3a:
#endif
#ifdef    __ID3b
                    case    __ID3b:
#endif
#ifdef    __ID3d
                    case    __ID3d:
#endif
                        counter     =     0x88;
                        in_status.BIT.r_id     =     1;
                        r_checksum.WORD     =     0;
                        LIN_data_set();
                        buff     =     LIN_tx_data[0];
                        t_checksum.WORD     =     (unsigned short)buff;
                        TZ.GRB1     +=     flame_max_8.WORD.l;
                        flame_max     =     flame_max_8.WORD.h;
                        TZ.TSR1.BIT.IMFB     =     0;
                        TZ.TIER1.BIT.IMIEB     =     1;
                        SCI3.TDR     =     buff;
                        break;
#endif
/*                  case    0x3C:
                        counter     =     0x88;
                        r_checksum.WORD     =     0;
                        TZ.GRB1     +=     flame_max_8.WORD.l;
                        flame_max     =     flame_max_8.WORD.h;
                        TZ.TSR1.BIT.IMFB     =     0;
                        TZ.TIER1.BIT.IMIEB     =     1;
                        break;
*/
                    case    0xFE:
                    case    0xBF:
                        LIN_Sflag_init();
                        LIN_break_reception_PR();
                        break;
                    default :
                        dlc     =     buff     &     0x30;
                        if(dlc   ==   0x20){
                            counter     =     0x44;
                            TZ.GRB1     +=     flame_max_4.WORD.l;
                            flame_max     =     flame_max_4.WORD.h;
```

```
                        }else if(dlc   ==   0x30){
                            counter      =      0x88;
                            TZ.GRB1      +=      flame_max_8.WORD.l;
                            flame_max      =      flame_max_8.WORD.h;
                        }else{
                            counter      =      0x22;
                            TZ.GRB1      +=      flame_max_2.WORD.l;
                            flame_max      =      flame_max_2.WORD.h;
                        }
                        r_checksum.WORD = 0;
                        TZ.TSR1.BIT.IMFB      =      0;
                        TZ.TIER1.BIT.IMIEB      =      1;
                        break;
                    }
                }
            }else if(SCI3.RDR   ==   0x55){
                in_status.BIT.sync_field      =      1;
            }else{
                LIN_status.BIT.ISFE      =      1;
                LIN_Sflag_init();
                LIN_break_reception_PR();
                LIN_error();
            }
        }
    }
```

3. 参考ドキュメント

- LIN Protocol Specification Revision 1.2
- LIN Protocol Specification Revision 1.3 Draft 7
- H8/3687 シリーズハードウェアマニュアル　第一版

改訂記録<revision history,rh>

Rev.	発行日	改訂内容	
		ページ	ポイント
1.41	2003.06.20	—	初版発行

安全設計に関するお願い

・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させじ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

・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。

・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。

・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。

・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。

・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。

・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。

・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。

・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。