CAN スタータキット RX/RA
CAN スタータキット SmartRX
RSCAN モジュール編
ソフトウェアマニュアル

ルネサス エレクトロニクス社　RX/RA マイコン搭載
HSB シリーズマイコンボード　評価キット

-本書を必ずよく読み、ご理解された上でご利用ください

株式会社北斗電子
REV.1.9.0.0

ー目　次ー

注意事項 **1**

安全上のご注意 **2**

概要 **4**

1.　ソースファイル構成 **5**

2.　サンプルプログラムの説明(SAMPLE1) **6**

2.1.　プログラム仕様 6
2.2.　送受信バッファの設定 7
2.3.　動作説明 7
2.4.　データの受信 9
2.5.　SAMPLE1 フローチャート 10

3.　サンプルプログラムの説明(SAMPLE2) **11**

3.1.　プログラム仕様 11
3.2.　使用する割り込み 11
3.2.1.　RX200 系マイコン RSCAN モジュール 11
3.3.　送受信 FIFO バッファの設定 11
3.3.1.　受信ルール設定 12
3.4.　動作説明 12
3.5.　SAMPLE2 フローチャート 14

4.　サンプルプログラムの説明(SAMPLE3) **16**

4.1.　プログラム仕様 16
4.2.　受信ルール設定 17
4.3.　動作説明 17
4.4.　SAMPLE3 フローチャート 19

5.　サンプルプログラムで使用している関数の説明 **21**

5.1.　関数仕様 21
5.2.　プログラムで使用している変数・定数 27
5.2.1.　グローバル変数 27
5.2.2.　定数定義 27

取扱説明書改定記録 29
お問合せ窓口 29

注意事項

本書を必ずよく読み、ご理解された上でご利用ください

【ご利用にあたって】

1. 本製品をご利用になる前には必ず取扱説明書をよく読んで下さい。また、本書は必ず保管し、使用上不明な点がある場合は再読し、よく理解して使用して下さい。
2. 本書は株式会社北斗電子製マイコンボードの使用方法について説明するものであり、ユーザシステムは対象ではありません。
3. 本書及び製品は著作権及び工業所有権によって保護されており、全ての権利は弊社に帰属します。本書の無断複写・複製・転載はできません。
4. 弊社のマイコンボードの仕様は全て使用しているマイコンの仕様に準じております。マイコンの仕様に関しましては製造元にお問い合わせ下さい。弊社製品のデザイン・機能・仕様は性能や安全性の向上を目的に、予告無しに変更することがあります。また価格を変更する場合や本書の図は実物と異なる場合もありますので、御了承下さい。
5. 本製品のご使用にあたっては、十分に評価の上ご使用下さい。
6. 未実装の部品に関してはサポート対象外です。お客様の責任においてご使用下さい。

【限定保証】

1. 弊社は本製品が頒布されているご利用条件に従って製造されたもので、本書に記載された動作を保証致します。
2. 本製品の保証期間は購入戴いた日から1年間です。

【保証規定】

保証期間内でも次のような場合は保証対象外となり有料修理となります

1. 火災・地震・第三者による行為その他の事故により本製品に不具合が生じた場合
2. お客様の故意・過失・誤用・異常な条件でのご利用で本製品に不具合が生じた場合
3. 本製品及び付属品のご利用方法に起因した損害が発生した場合
4. お客様によって本製品及び付属品へ改造・修理がなされた場合

【免責事項】

弊社は特定の目的･用途に関する保証や特許権侵害に対する保証等、本保証条件以外のものは明示・黙示に拘わらず一切の保証は致し兼ねます。また、直接的・間接的損害金もしくは欠陥製品や製品の使用方法に起因する損失金・費用には一切責任を負いません。損害の発生についてあらかじめ知らされていた場合でも保証は致し兼ねます。
ただし、明示的に保証責任または担保責任を負う場合でも、その理由のいかんを問わず、累積的な損害賠償責任は、弊社が受領した対価を上限とします。本製品は「現状」で販売されているものであり、使用に際してはお客様がその結果に一切の責任を負うものとします。弊社は使用または使用不能から生ずる損害に関して一切責任を負いません。
保証は最初の購入者であるお客様ご本人にのみ適用され、お客様が転売された第三者には適用されません。よって転売による第三者またはその為になすお客様からのいかなる請求についても責任を負いません。
本製品を使った二次製品の保証は致し兼ねます。

安全上のご注意

製品を安全にお使いいただくための項目を次のように記載しています。絵表示の意味をよく理解した上でお読み下さい。

表記の意味

警告 取扱を誤った場合、人が死亡または重傷を負う危険が切迫して生じる可能性がある事が想定される

注意 取扱を誤った場合、人が軽傷を負う可能性又は、物的損害のみを引き起こすが可能性がある事が想定される

絵記号の意味

	**一般指示** 使用者に対して指示に基づく行為を強制するものを示します		**一般禁止** 一般的な禁止事項を示します
	**電源プラグを抜く** 使用者に対して電源プラグをコンセントから抜くように指示します		**一般注意** 一般的な注意を示しています

警告

以下の警告に反する操作をされた場合、本製品及びユーザシステムの破壊・発煙・発火の危険があります。マイコン内蔵プログラムを破壊する場合もあります。

1. 本製品及びユーザシステムに電源が入ったままケーブルの抜き差しを行わないでください。
2. 本製品及びユーザシステムに電源が入ったままで、ユーザシステム上に実装されたマイコンまたはIC等の抜き差しを行わないでください。
3. 本製品及びユーザシステムは規定の電圧範囲でご利用ください。
4. 本製品及びユーザシステムは、コネクタのピン番号及びユーザシステム上のマイコンとの接続を確認の上正しく扱ってください。

発煙・異音・異臭にお気付きの際はすぐに使用を中止してください。

電源がある場合は電源を切って、コンセントから電源プラグを抜いてください。そのままご使用すると火災や感電の原因になります。

⚠注意

以下のことをされると故障の原因となる場合があります。

1. 静電気が流れ、部品が破壊される恐れがありますので、ボード製品のコネクタ部分や部品面には直接手を触れないでください。
2. 次の様な場所での使用、保管をしないでください。

   ホコリが多い場所、長時間直射日光があたる場所、不安定な場所、衝撃や振動が加わる場所、落下の可能性がある場所、水分や湿気の多い場所、磁気を発するものの近く
3. 落としたり、衝撃を与えたり、重いものを乗せないでください。
4. 製品の上に水などの液体や、クリップなどの金属を置かないでください。
5. 製品の傍で飲食や喫煙をしないでください。

ボード製品では、裏面にハンダ付けの跡があり、尖っている場合があります。

取り付け、取り外しの際は製品の両端を持ってください。裏面のハンダ付け跡で、誤って手など怪我をする場合があります。

CD メディア、フロッピーディスク付属の製品では、故障に備えてバックアップ(複製)をお取りください。

製品をご使用中にデータなどが消失した場合、データなどの保証は一切致しかねます。

アクセスランプがある製品では、アクセスランプの点灯中に電源を切ったり、パソコンをリセットをしないでください。

製品の故障や、データ消失の原因となります。

本製品は、医療、航空宇宙、原子力、輸送などの人命に関わる機器やシステム及び高度な信頼性を必要とする設備や機器などに用いられる事を目的として、設計及び製造されておりません。

医療、航空宇宙、原子力、輸送などの設備や機器、システムなどに本製品を使用され、本製品の故障により、人身や火災事故、社会的な損害などが生じても、弊社では責任を負いかねます。お客様ご自身にて対策を期されるようご注意ください。

概要

本書は、「CAN スタータキット RX/RA」「CAN スタータキット SmartRX」付属 CD に含まれる、サンプルプログラムの解説を行う資料となります。

従来のマニュアルでは、複数のモジュールの動作を併記していましたが、本バージョンからモジュール毎に分割を行う事と致しました。本書は、「RSCAN モジュール編」のマニュアルです。RSCAN モジュール搭載マイコンの場合、本書を参照してください。

1. ソースファイル構成

・RSCAN モジュール向け

フォルダ	ファイル	説明
source¥RSCAN_module¥rscan		RSCAN モジュール向け共通フォルダ
	rscan.h	各種定義ファイル
	rscan.c	ch0 関数
	rscan_s1.c	SAMPLE1 向け割り込み関数定義（中身は空）
	rscan_s2.c	SAMPLE2 向け割り込み関数定義
	rscan_s3.c	SAMPLE3 向け割り込み関数定義
source¥RSCAN_module¥main		メイン関数
	main_s1.c	SAMPLE1 メイン関数
	main_s2.c	SAMPLE2 メイン関数
	main_s3.c	SAMPLE3 メイン関数

※RSCAN は CAN の ch が 2ch 以上となるマイコンが存在しないため、ch によるソースファイルの分割はありません

2. サンプルプログラムの説明(SAMPLE1)

2.1. プログラム仕様

・データフレームの送信
・データフレームの受信

を行うサンプルプログラムとします。

・CAN ID は拡張フォーマット(29bit)とする（標準フォーマットのデータも受信する）(*1)
・送信に使用する ID は 0x0000000~0x0000003 までの 4 種
・受信する ID は、0x0000000~0x0000003 までの 4 種（それ以外の ID のデータは受信しない）
・送信データは"s"コマンドで拡張フォーマットと標準フォーマットの切り替えが可能
・複数の CAN ch を持っているボードは、全ポート受信を行い、送信は"c"コマンドで ch の変更を行う
・データフレームのみ受信（リモートフレームは受信しない）
・端末のキーボード入力を読み取り 0~3 の入力に応じて ID，送信データバイト数を変えて送信する
・プッシュスイッチが付いているボードでは、プッシュスイッチを押すと 1 バイト送信する（キーボードの 0 と等価）
・LED が付いているボードはデータを受信する度に LED の点灯・消灯が切り替わる
（SmartRX!!!ボードでは、受信 ID に応じた LED が点灯する）

－キーボードから入力したキーと送信データの関係－

キーボードからの入力	ID	送信バイト数	送信データ
0	0x0000000	1	0x 01
1	0x0000001	2	0x 01 23
2	0x0000002	4	0x 01 23 45 67
3	0x0000003	8	0x 01 23 45 67 89 AB CD EF

(*1)can_operation.h の定義で、標準フォーマットのみ取り扱う様に変更可能

2.2. 送受信バッファの設定

受信ルール番号	フォーマット	ID	受信バッファ番号
0	標準(SID)	0x000	0
1	標準(SID)	0x001	1
2	標準(SID)	0x002	2
3	標準(SID)	0x003	3
4	拡張(EID)	0x0000000	4
5	拡張(EID)	0x0000001	5
6	拡張(EID)	0x0000002	6
7	拡張(EID)	0x0000003	7

RSCAN モジュールは、16 個の受信ルールが設定可能で、ルールにマッチした（ID が一致した）データの格納先を設定できますが、本サンプルプログラムでは、受信バッファに割り当てています。（受信バッファは、16 個ありますが、受信ルールと、受信バッファは自由に紐付けできます。本サンプルプログラムでは、受信ルール番号と受信バッファの同じ番号のものを 1:1 で紐付けさせていますが、設定は自由です）

2.3. 動作説明

実際に CAN の通信を行う際、関数をどのような流れで呼び出すかを以下で説明します。

(1)初期化を行う

can_init();

CAN の初期化をする。

(2)受信ルールの設定

//引数:　　受信ルール番号　バッファ／FIFO 区分　フォーマット区分　データフレーム／リモートフレーム区分　ID
can_receive_rule_set(0, CAN_RULE_BUF, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000000);

can_receive_rule_set(1, CAN_RULE_BUF, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000001);
…
can_receive_rule_set(4, CAN_RULE_BUF, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000000);
…
can_receive_rule_set(7, CAN_RULE_BUF, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000003);

can_start();

受信ルール番号 0~7 を受信設定する例。
ルール設定後、can_start を呼び出す。

(3)データの受信

```
for(i=0; i<=7; i++)  //受信ルール 0~7 が設定済み
{
   //引数:      受信ルール番号  フォーマット区分  データフレーム／リモートフレーム区分  ID  データ  タイムスタンプ
   r_ret = can_buf_receive((unsigned char)i, &r_ide, &r_rtr, &r_id, &r_data[0], &r_ts);
```

受信ルール番号：can_receive_rule_set()で設定した番号
フォーマット区分：受信した標準フォーマット(CAN_ID_FORMAT_SID)，拡張フォーマット(CAN_ID_FORMAT_EID)
データフレーム／リモートフレーム区分：受信したデータフレーム(CAN_DATA_FRAME)，リモートフレーム(CAN_REMOTE_FRAME)
ID：受信した ID 値
データ：受信データ
タイムスタンプ：タイムスタンプ値（受信側で付与）

r_ret が 0 ならば、受信したデータはなし。1~8 であれば、戻り値に対応するバイト数のデータを受信しています。

(4)データの送信

```
unsigned char s_data[8]={0x01,0x23,0x45,0x67,0x89,0xAB,0xCD,0xEF}; //送信データ

//引数:      バッファ番号  フォーマット区分  データフレーム／リモートフレーム区分  ID  送信バイト数  データ
s_ret = can_buf_send(0, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x0000, 1, &s_data[0]);
```

バッファ番号：送信に使用する送信バッファ番号
フォーマット区分：標準フォーマット(CAN_ID_FORMAT_SID)，拡張フォーマット(CAN_ID_FORMAT_EID)
データフレーム／リモートフレーム区分：データフレーム(CAN_DATA_FRAME)，リモートフレーム(CAN_REMOTE_FRAME)
送信バイト数：1~8
データ：送信データ

送信バッファ 0 から ID=0x000、データ 0x01 を 1 バイト送信する。

2.4. データの受信

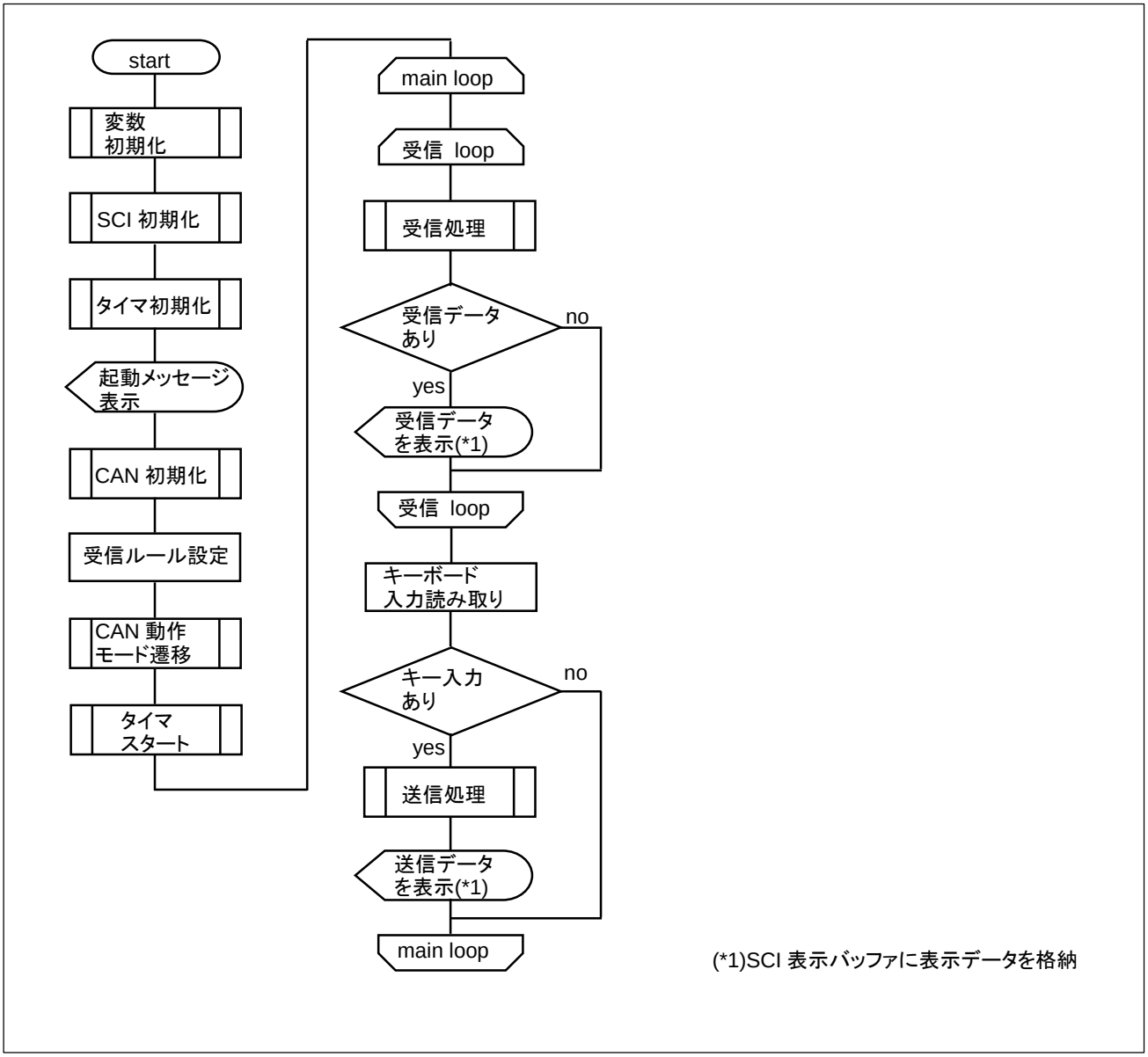
SAMPLE1 でのデータの受信に関しては、受信バッファまでのデータ格納に関しては、（初期化、メールボックス、受信ルール設定が済んでいれば）マイコンのハードウェアが行います。メールボックス、受信バッファにデータが格納されているかは、プログラムで受信関数を呼び出す事で確認を行っています。そのため、常に受信関数を呼び出して確認を行わないと、データの取りこぼしが生じる可能性があります。受信データの確認に CPU リソースを食うため、スムーズな手法とはいえないと考えます。

（なお、次のサンプルプログラム、SAMPLE2 ではデータ受信時に割り込みが入る様に設定しており、受信の処理が割り込みによって処理されます。）

2.5. SAMPLE1 フローチャート

－処理フロー－

メイン関数　main_s1()



3. サンプルプログラムの説明(SAMPLE2)

3.1. プログラム仕様

・データフレームの送信
・データフレームの受信

を行うサンプルプログラムとします。

SAMPLE1 との相違点は、データ送信後の処理と、受信処理を割り込みを使用して行う事とします。

3.2. 使用する割り込み

3.2.1. RX200 系マイコン RSCAN モジュール

RX200 系マイコンでは、FIFO に割り込みが割り当てられているため、SAMPLE2 では、FIFO を使用してデータの送受信を行う事とします。

・RX24U

割り込み 要因番号	割り込み 要求元	名称
60	RSCAN	RXFINT
61	RSCAN	TXINT

・RX231

割り込み 要因番号	割り込み 要求元	名称
53	RSCAN	RXFINT
54	RSCAN	TXINT

3.3. 送受信 FIFO バッファの設定

RSCAN モジュール系では、SAMPLE1 では、受信・送信バッファを使用していましたが、SAMPLE2 では受信には「受信 FIFO」。送信には「送受信 FIFO」を送信の設定で使用する事とします。

合計 16 個の FIFO バッファが使用できますので、本サンプルプログラムでは、
受信 FIFO 8 バッファ
送受信 FIFO（送信に使用） 8 バッファ
を割り当てる事とします。

3.3.1. 受信ルール設定

受信ルール番号	フォーマット	ID	使用 FIFO
0	標準(SID)	0x000	FIFO(0)
1	標準(SID)	0x001	FIFO(0)
2	標準(SID)	0x002	FIFO(0)
3	標準(SID)	0x003	FIFO(0)
4	拡張(EID)	0x0000000	FIFO(0)
5	拡張(EID)	0x0000001	FIFO(0)
6	拡張(EID)	0x0000002	FIFO(0)
7	拡張(EID)	0x0000003	FIFO(0)

SAMPLE2 での受信ルールでは、データを受信した際に、受信 FIFO の 0 番側にデータが格納される様設定します。

3.4. 動作説明

実際に CAN の通信を行う際、関数をどのような流れで呼び出すかを以下で説明します。SAMPLE1 と同一な点は説明を省略します。

(1)初期化を行う

can_init();

(2)受信ルールの設定[SAMPLE1 との相違点]

//引数:　　受信ルール番号　バッファ／FIFO 区分　フォーマット区分　データフレーム／リモートフレーム区分　ID
can_receive_rule_set(0, **CAN_RULE_FIFO0**, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000000);
…
can_start();

受信ルール設定時、受信バッファではなく FIFO(0)を使うように設定。
（受信割り込み機能が、FIFO と関連しているためです）

(3)データの送信

//引数:　　フォーマット区分　データフレーム／リモートフレーム区分　ID　送信バイト数　データ
s_ret = **can_fifo_send**(s_format, CAN_DATA_FRAME, s_id, s_dlc, &s_data[0]);

送受信 FIFO を使用する関数で送信。

(1)~(3)の処理はメイン関数で処理され、(2)(3)が SAMPLE1 と異なります。

(i1)受信割り込み関数[SAMPLE1 との相違点]

```
while(RSCAN.RFSTS0.BIT.RFEMP != 1)
{
    ret = ret = can_fifo_receive(&ide, &rtr, &id, &data[0], &ts);
    [受信データの表示]
}
```

割り込みフラグクリアと、受信 FIFO にデータが存在する内は FIFO からの読み出しと表示を行う。
受信の関数が、FIFO 向けに変更されています

(i2)送信割り込み関数[SAMPLE1 との相違点]

割り込みフラグクリアと、データ送信済みの画面表示を行う。

受信割り込み関数、送信割り込み関数は

rscan_s2.c　　（SAMPLE2 の割り込み関数は_s2.c 内で定義）

内で定義されています。

・割り込み関数名

受信割り込み関数	送信割り込み関数
Intr_RSCAN_RXFINT_s(void)	Intr_RSCAN_TXINT_s(void)

※元々の割り込み関数は、Intr_RSCAN_RXFINT(void)の様に_s がない形で定義されています、_s 付きの関数は

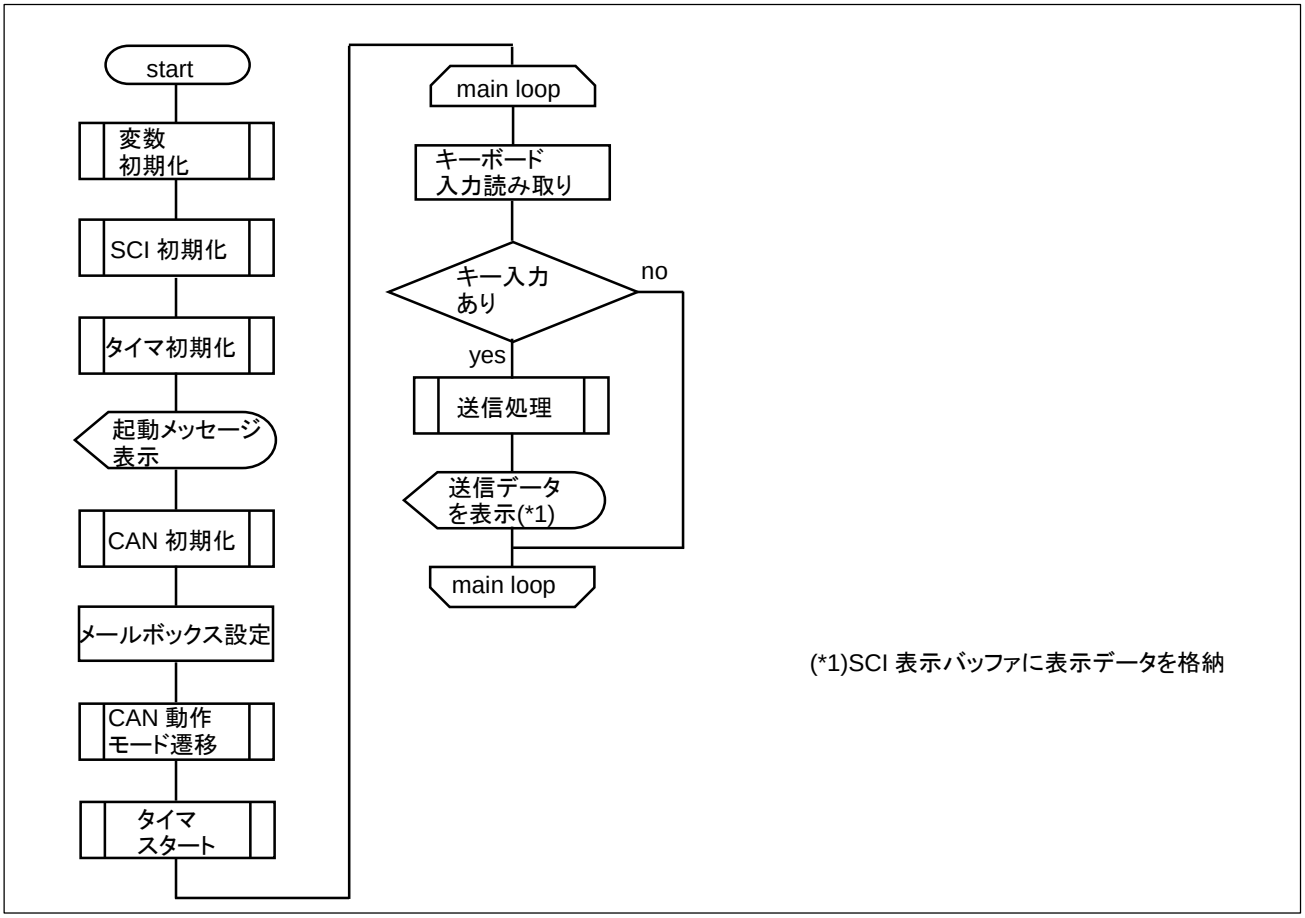
```
Intr_RSCAN_RXFINT(void)      //システムで呼び出される割り込み関数
{
//この部分にユーザ関数外で処理する内容を追加するケースを想定して分けている
        Intr_RSCAN_RXFINT_s();      //ユーザ定義の割り込み関数
}
```

の様にワンクッション入れて呼び出しています。（現状、特段深い意味合いはありません。）

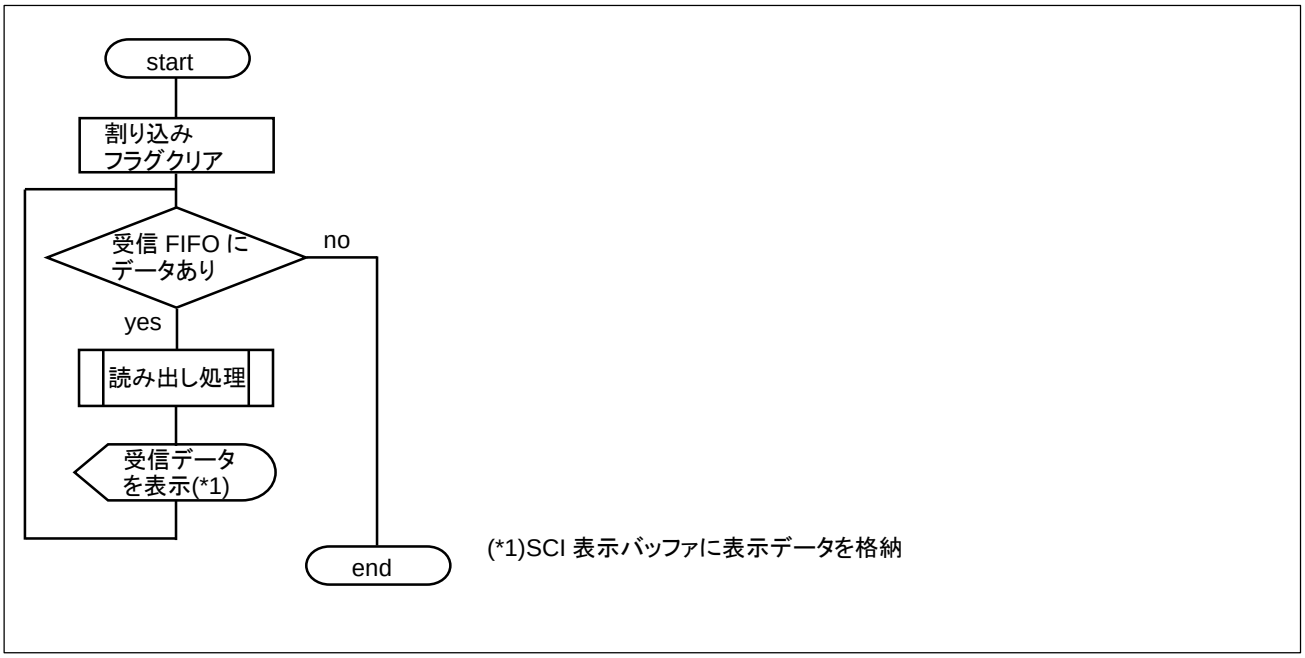
3.5. SAMPLE2 フローチャート

－処理フロー－

メイン関数　main_s2()



受信割り込み関数　Intr_RSCAN_RXFINT()



送信完了割り込み関数　Intr_RSCAN_TXINT()

4. サンプルプログラムの説明(SAMPLE3)

4.1. プログラム仕様

・データフレームの送信
・データフレームの受信
・リモートフレームの送信
・リモートフレームを受信した際応答（データ送信を行う）する

を行うサンプルプログラムとします。

SAMPLE2 との相違点は、リモートフレームに対応している事です。

－キーボードから入力したキーと送信データの関係－

・データフレーム送信
キーボード 0~3（SAMPLE1 と同一）

・リモートフレーム送信

キーボードからの入力	ID	送信要求バイト数(DLC)
q	0x0000000	1
w	0x0000001	2
e	0x0000002	4
r	0x0000003	8

・リモートフレーム応答
0xA1 A2 A3 A4 A5 A6 A7 A8
を、要求元の送信要求バイト数に応じて返答
（DLC=4 のときは、0xA1 A2 A3 A4 を返す）

※本サンプルプログラムは、ID=0x0000000 ~ 0x0000003 でリモートフレームを受信すると、応答（データフレームを送信）しますので、本サンプルプログラムが動作するボードを 3 台（以上）同一バスに接続すると、2 台（以上）が同時に応答します　CAN バス上では、1 つのリモートフレームに続き 2 つ（以上）のデータが流がれますので、多少動作が見難くなるかと思います

※SAMPLE3 を 3 台以上同時に接続させて動作させる場合は、ボード毎に応答する ID を変更する事を推奨致します

4.2. 受信ルール設定

受信ルール番号	フォーマット	データフレーム／リモートフレーム	ID	使用 FIFO
0	標準(SID)	データフレーム	0x000	FIFO(0)
1	標準(SID)	データフレーム	0x001	FIFO(0)
2	標準(SID)	データフレーム	0x002	FIFO(0)
3	標準(SID)	データフレーム	0x003	FIFO(0)
4	拡張(EID)	データフレーム	0x0000000	FIFO(0)
5	拡張(EID)	データフレーム	0x0000001	FIFO(0)
6	拡張(EID)	データフレーム	0x0000002	FIFO(0)
7	拡張(EID)	データフレーム	0x0000003	FIFO(0)
8	標準(SID)	リモートフレーム	0x000	FIFO(0)
9	標準(SID)	リモートフレーム	0x001	FIFO(0)
10	標準(SID)	リモートフレーム	0x002	FIFO(0)
11	標準(SID)	リモートフレーム	0x003	FIFO(0)
12	拡張(EID)	リモートフレーム	0x0000000	FIFO(0)
13	拡張(EID)	リモートフレーム	0x0000001	FIFO(0)
14	拡張(EID)	リモートフレーム	0x0000002	FIFO(0)
15	拡張(EID)	リモートフレーム	0x0000003	FIFO(0)

8~15 をリモートフレーム受信用に割り当て。

4.3. 動作説明

実際に CAN の通信を行う際、関数をどのような流れで呼び出すかを以下で説明します。SAMPLE2 と同一な点は説明を省略します。

(1)初期化を行う

(2)受信ルールの設定[SAMPLE2 との相違点]

//引数:　　受信ルール番号　バッファ／FIFO 区分　フォーマット区分　データフレーム／リモートフレーム区分　ID

…

can_receive_rule_set(8, CAN_RULE_FIFO0, CAN_ID_FORMAT_SID, **CAN_REMOTE_FRAME**, 0x00000000);

…

can_start();

受信ルール番号 8-15 をリモートフレーム受信向けに追加。

(3)データの送信

//引数:　　バッファ番号　フォーマット区分　データフレーム／リモートフレーム区分　ID　送信バイト数　データ

s_ret = can_fifo_send(s_format, **s_rtr**, s_id, s_dlc, &s_data[0]);

SAMPLE1~2 では、データフレーム固定のところが、SAMPLE3 では、コマンド(0~3, **q~r**)に応じて、送信する値を変更しています。

(i1)受信割り込み関数[SAMPLE2 との相違点]

```
while(RSCAN.RFSTS0.BIT.RFEMP != 1)
{
    ret = can_fifo_receive(&ide, &rtr, &id, &data[0], &ts);
    [データフレームの場合：受信データの表示]
    [リモートフレームの場合：データ表示，データ送信]
    ret2 = can_fifo_send(ide, CAN_DATA_FRAME, id, dlc, &remote_frame_response_data[0]);
}
```

割り込みフラグクリアと、受信 FIFO にデータが存在する内は FIFO からの読み出しと表示を行う。リモートフレームの際はデータの送信を行う。データ送信に関しては、通常のデータフレーム送信と同じ関数(can_fifo_send)で行う。但し、送信バイト数はリモートフレーム要求元から送信されてきた値(DLC)を使用する。

(i2)送信割り込み関数[SAMPLE2 との相違点]

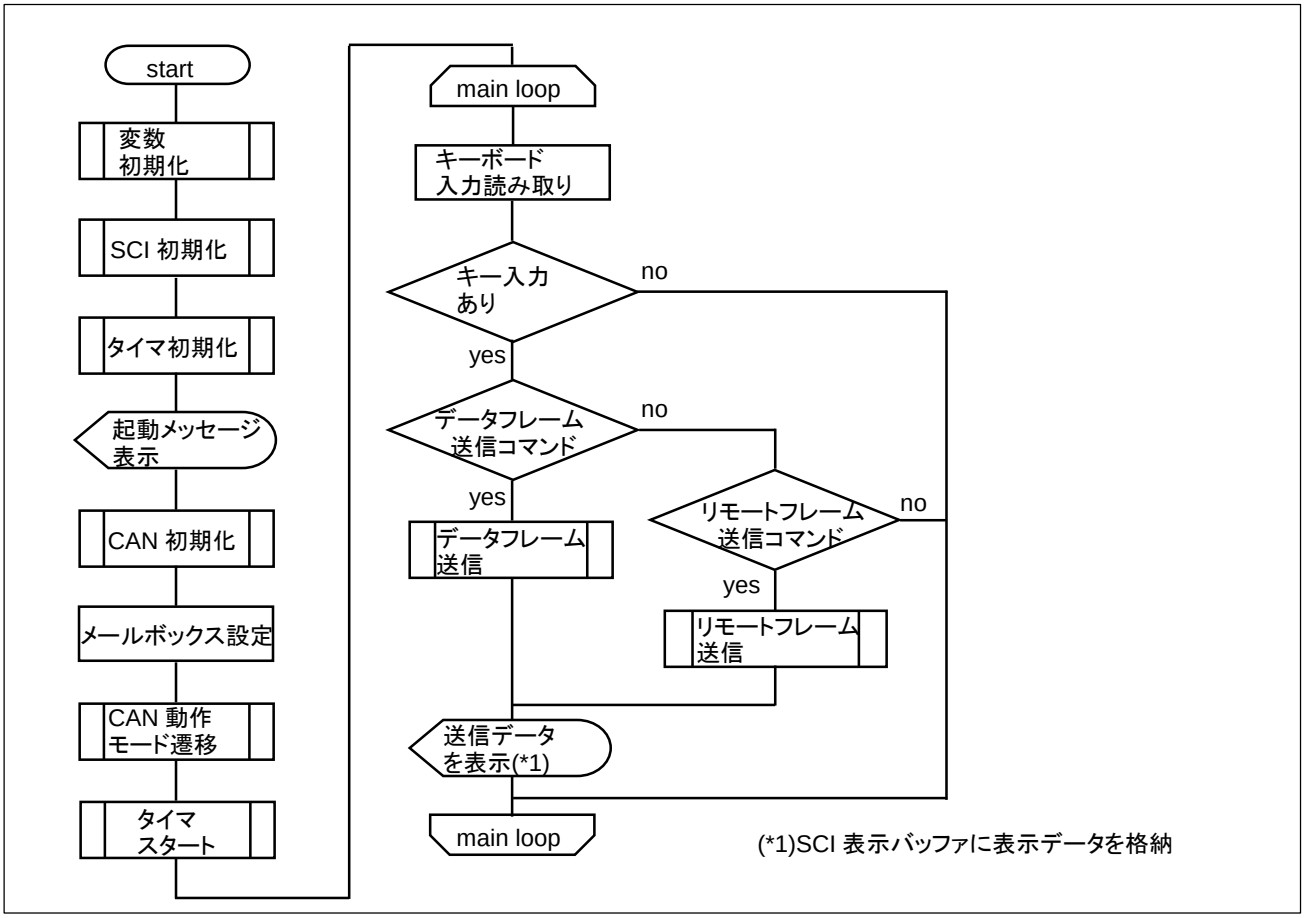
```
if(can0_remote_frame_request != 1) sciPrintBuf("--¥n");
```

リモートフレームの場合はデータの区切りが判る様表示を追加しています。

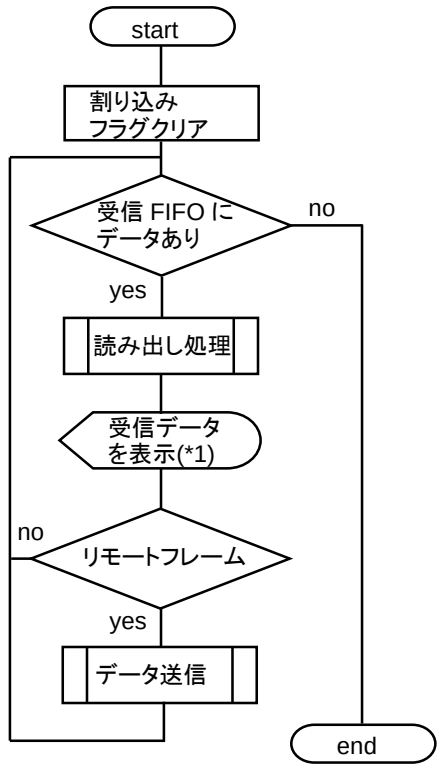
4.4. SAMPLE3 フローチャート

－処理フロー－

メイン関数　main_s3()



受信割り込み関数（RSCAN モジュール系） Intr_RSCAN_RXFINT()



(*1)SCI 表示バッファに表示データを格納

5. サンプルプログラムで使用している関数の説明

5.1. 関数仕様

can_init

概要:初期化関数
宣言:
　　int can_init(void)

説明:
　・モジュールストップ解除
　・端子設定
　・通信設定
を行います

引数:
　　なし
戻り値:
　　0: 正常終了

　初期化関数は、CAN を 1Mbps 設定(*1)で初期化します。

(*1)通信速度は、can_operation.h の定義で変更可能です
通信速度の異なるボード同士は通信が行えません

can_receive_rule_set

概要:受信ルール設定関数

宣言:

int can_receive_rule_set(unsigned char num, unsigned char mode, unsigned char ide, unsigned char rtr, unsigned long id);

説明:

・受信ルール設定

を行います

引数:

num: 受信ルール番号(0~15)

mode: 受信先

CAN_RULE_BUF(0x01)	受信バッファで受信
CAN_RULE_FIFO0(0x02)	受信 FIFO(0)で受信
CAN_RULE_FIFO1(0x04)	受信 FIFO(1)で受信
CAN_RULE_SR_FIFO(0x08)	送受信 FIFO で受信

ide: 標準／拡張フォーマット区分

CAN_ID_FORMAT_SID(0)	標準フォーマット(ID=11bit)
CAN_ID_FORMAT_EID(1)	拡張フォーマット(ID=29bit)

rtr: データフレーム／リモートフレーム区分

CAN_DATA_FRAME(0)	データフレーム（データ送信）
CAN_REMOTE_FRAME(1)	リモートフレーム（相手にデータ要求）

id: ID を指定

戻り値:

0: 正常終了

-1: 引数エラー

-2: 受信ルール設定不可の動作モード

RSCAN モジュール系では本関数で「受信ルール」の設定を行います。受信ルールにマッチしたデータが、本関数で指定した受信先にコピーされるという動作です。

can_start　　[RSCAN モジュール系]

概要:動作モード変更関数
宣言:
　　int can_start(void)
説明:
　・CAN モジュールを動作モードに移行する
を行います
引数:　なし
戻り値:
　　0:　正常終了
　　-2:　動作モード変更 NG

RSCAN モジュール系では、受信ルール設定完了後、本関数を呼び出す事により、CAN モジュールを動作モードに変更します。

can_buf_send　　[RSCAN モジュール系]

概要:データ送信関数
宣言:
　　int can_buf_send(unsigned char num, unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char *data);
説明:
　・送信バッファを使用してデータの送信
を行います
引数:
　　num:　送信バッファ番号(0~3)
　　ide:　標準／拡張フォーマット区分
　　　CAN_ID_FORMAT_SID(0)　標準フォーマット(ID=11bit)
　　　CAN_ID_FORMAT_EID(1)　拡張フォーマット(ID=29bit)
　　rtr:　データフレーム／リモートフレーム区分
　　　CAN_DATA_FRAME(0)　　データフレーム（データ送信）
　　　CAN_REMOTE_FRAME(1)　リモートフレーム（相手にデータ要求）
　　id:　送信する ID を指定します
　　dlc:　送信バイト数を指定します(1~8)
　　*data:　送信するデータを指定します
戻り値:
　　0:　正常終了
　　-1:　引数チェックエラー
　　-3:　送信バッファ使用中

RSCAN モジュール系は、送信関数を呼び出す際に、全ての情報を指定する仕様です。

can_fifo_send

概要:データ送信関数
宣言:
　　int can_fifo_send(unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char *data)
説明:
　・送受信 FIFO を使用してデータの送信
を行います
引数:
　　ide: 標準／拡張フォーマット区分
　　CAN_ID_FORMAT_SID(0)　標準フォーマット(ID=11bit)
　　CAN_ID_FORMAT_EID(1)　拡張フォーマット(ID=29bit)
　　rtr: データフレーム／リモートフレーム区分
　　CAN_DATA_FRAME(0)　　データフレーム（データ送信）
　　CAN_REMOTE_FRAME(1)　リモートフレーム（相手にデータ要求）
　　id: 送信する ID を指定します
　　dlc: 送信バイト数を指定します(1~8)
　　*data: 送信するデータを指定します
戻り値:
　　0: 正常終了
　　-1: 引数チェックエラー
　　-2: FIFO フル使用中

データ送信関数は、送信バッファを使う関数と、FIFO を使う関数の 2 種類を用意しています。送信割り込みを使いたい場合は、本関数を使用してください。

can_buf_receive

概要:受信関数

宣言:

int can_buf_receive(unsigned char num, unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts)

説明:

・受信バッファを使用したデータの受信

を行います

引数:

num: 受信ルール番号(＝受信バッファ番号)(0~15)

*ide: 標準／拡張フォーマット区分を格納するポインタ(unsigned char x1)

CAN_ID_FORMAT_SID(0) 標準フォーマット(ID=11bit)

CAN_ID_FORMAT_EID(1) 拡張フォーマット(ID=29bit)

*rtr: データフレーム／リモートフレーム区分を格納するポインタ(unsigned char x1)

CAN_DATA_FRAME(0) データフレーム(データ送信)

CAN_REMOTE_FRAME(1) リモートフレーム(相手にデータ要求)

*id: 受信した ID を格納するポインタ(unsigned long x1)

*data: 受信したデータを格納するポインタ(最大 unsigned char x8)

*ts: データ受信時のタイムスタンプ(unsigned short x1)

戻り値:

0: 受信データなし

1~8: 受信したデータのバイト数

-1: 引数チェックエラー

-3: 受信ルールが設定されていない

関数の動作としては、CAN モジュール系と同様です。

CAN モジュールの受信関数は、メールボックス番号を指定し、RSCAN モジュールの受信関数は、受信ルール番号(＝受信バッファ番号で設定)を指定する様になっています。

can_fifo_receive

概要:受信関数

宣言:

int can_fifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts)

説明:

・受信 FIFO を使用したデータの受信

を行います

引数:

*ide: 標準／拡張フォーマット区分を格納するポインタ(unsigned char x1)

CAN_ID_FORMAT_SID(0) 標準フォーマット(ID=11bit)

CAN_ID_FORMAT_EID(1) 拡張フォーマット(ID=29bit)

*rtr: データフレーム／リモートフレーム区分を格納するポインタ(unsigned char x1)

CAN_DATA_FRAME(0) データフレーム（データ送信）

CAN_REMOTE_FRAME(1) リモートフレーム（相手にデータ要求）

*id: 受信した ID を格納するポインタ(unsigned long x1)

*data: 受信したデータを格納するポインタ（最大 unsigned char x8）

*ts: データ受信時のタイムスタンプ(unsigned short x1)

戻り値:

0: 受信データなし（DLC=0 のデータはデータなしとして扱う）

1~8: 受信したデータのバイト数

-1: 引数チェックエラー

-2: 受信 FIFO にデータなし

※データ受信の割り込みで本関数を呼び出した際には FIFO にはデータが入っているはずなので戻り値(-2)にはならないはずです。データ受信の割り込み時以外で本関数を呼び出した場合は、戻り値(-2)となる事が考えられます。

5.2. プログラムで使用している変数・定数

5.2.1. グローバル変数

unsigned long can_remote_frame_request

リモートフレームであることを示すフラグ変数。リモートフレーム送信前にフラグを立て、リモートフレームの処理完了時にフラグを落とす。

5.2.2. 定数定義

rscan¥rscan.h
で定義

```
#define  CAN_BPS_1M  1          //通信速度  1Mbps
#define  CAN_BPS_500K  2        //通信速度  500kbps
#define  CAN_BPS_250K  3        //通信速度  250kbps
#define  CAN_BPS_125K  4        //通信速度  125kbps
```

通信速度設定。common¥can_operation.h 内で速度を変更する際に指定。
（数値は、クロック分周比と対応していますので、定義値を任意の数値に変更できる訳ではありません）

```
#define CAN_ID_FORMAT_SID  0        //標準フォーマット(ID=11bit)
#define  CAN_ID_FORMAT_EID  1       //拡張フォーマット(ID=11bit)
```

標準／拡張フォーマットール定義値。

```
#define CAN_DATA_FRAME0         //データフレーム
#define CAN_REMOTE_FRAME1       //リモートフレーム
```

データフレーム／リモートフレーム定義値。

```
#define CAN_CLOCK_PCLK  0       //PCLK(B)を CAN のクロックソースとして使用
#define CAN_CLOCK_XTAL  1       //XTAL を CAN のクロックソースとして使用
```

使用するクロックソース定義。

```
#defineCAN_RULE_BUF   0x01          //バッファモード
#defineCAN_RULE_FIFO0   0x02          //受信 FIFO0 で受信
#defineCAN_RULE_FIFO1   0x04          //受信 FIFO1 で受信
#defineCAN_RULE_SR_FIFO   0x08          //送受信 FIFO で受信
```

受信先設定定義値（bit0=1:バッファ, bit1=1:FIFO0, bit2=2:FIFO1, bit3=1:送受信 FIFO）。

取扱説明書改定記録

バージョン	発行日	ページ	改定内容
REV.1.8.0.0	2022.8.10	—	ソフトウェア編マニュアルから分離、独立
REV.1.9.0.0	2023.6.23		タイマ割り込み関数の削除

お問合せ窓口

最新情報については弊社ホームページをご活用ください。

ご不明点は弊社サポート窓口までお問合せください。

株式会社 北斗電子

〒060-0042 札幌市中央区大通西 16 丁目 3 番地 7

TEL 011-640-8800　FAX 011-640-8801

e-mail：support@hokutodenshi.co.jp (サポート用)、order@hokutodenshi.co.jp (ご注文用)

URL:https://www.hokutodenshi.co.jp

ルネサス エレクトロニクス RX, RA マイコン搭載
HSB シリーズマイコンボード 評価キット

CAN スタータキット RX/RA
CAN スタータキット SmartRX
RSCAN モジュール編
ソフトウェアマニュアル

株式会社 北斗電子
 2023 年 6 月 23 日改訂 REV.1.9.0.0 (230623)