CAN スタータキット RX/RA
CAN スタータキット SmartRX
CANFD 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

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、「CANFD モジュール編」のマニュアルです。CANFD, CANFD_B, CANFD-Lite 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

・CANFD モジュール向け

フォルダ	ファイル	説明
source¥CANFD_module¥canfd		CANFD モジュール向け共通フォルダ
	canfd.h	各種定義ファイル
	canfd.c	ch に依存しない処理関数
	canfd_s1.c	SAMPLE1 向け割り込み関数定義（中身は空）
	canfd_s2.c	SAMPLE2 向け割り込み関数定義
	canfd_s3.c	SAMPLE3 向け割り込み関数定義
	canfd_s4.c	SAMPLE4 向け割り込み関数定義
	canfd_pin_def.h	ch に依存しないデバッグ端子定義
source¥CANFD_module¥canfd0		ch0 向けフォルダ
	canfd_ch0.h	ch0 関数用ヘッダ
	canfd_ch0.c	ch0 関数
	can_ch0_pin_def.h	ch0 端子設定
source¥CANFD_module¥canfd1		ch1 向けフォルダ
	canfd_ch1.h	ch1 関数用ヘッダ
	canfd_ch1.c	ch1 関数
	can_ch1_pin_def.h	ch1 端子設定
source¥CANFD_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
	main_s4.c	SAMPLE4 メイン関数

※CAN モジュール向けは割り込み関数を ch 毎に分けていましたが、CANFD の割り込みは ch 毎に独立していないため、canfd フォルダの下に割り込み関数定義を入れています

・CANFD_B モジュール向け

フォルダ	ファイル	説明
source¥CANFD_B_module¥canfd_b		CANFD_B モジュール向け共通フォルダ
	canfd_b.h	各種定義ファイル
	canfd_b.c	ch に依存しない処理関数
	canfd_b_s1.c	SAMPLE1 向け割り込み関数定義（中身は空）
	canfd_b_s2.c	SAMPLE2 向け割り込み関数定義
	canfd_b_s3.c	SAMPLE3 向け割り込み関数定義
	canfd_b_s4.c	SAMPLE4 向け割り込み関数定義
	canfd_b_pin_def.h	デバッグ端子定義
source¥CANFD_B_module¥canfd_b0		ch0 向けフォルダ
	canfd_b_ch0.h	ch0 関数用ヘッダ
	canfd_b_ch0.c	ch0 関数
	can_ch0_pin_def.h	ch0 端子設定
source¥CANFD_B_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
	main_s4.c	SAMPLE4 メイン関数

※CANFD_B は現状 1ch のみのサポートですが、ch 固有の設定等は分離させています

・CANFD-Lite モジュール向け

フォルダ	ファイル	説明
source¥CANFDLite_module¥canfd_lite		CANFD-Lite モジュール向け共通フォルダ
	canfd_lite.h	各種定義ファイル
	canfd_lite.c	ch に依存しない処理関数
	canfd_lite_s1.c	SAMPLE1 向け割り込み関数定義 （中身は空）
	canfd_lite_s2.c	SAMPLE2 向け割り込み関数定義
	canfd_lite_s3.c	SAMPLE3 向け割り込み関数定義
	canfd_lite_s4.c	SAMPLE4 向け割り込み関数定義
	canfd_lite_pin_def.h	デバッグ端子定義
source¥CANFDLite_module¥canfd_lite0		ch0 向けフォルダ
	canfd_lite_ch0.h	ch0 関数用ヘッダ
	canfd_lite_ch0.c	ch0 関数
	can_ch0_pin_def.h	ch0 端子設定
source¥CANFDLite_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
	main_s4.c	SAMPLE4 メイン関数

※CANFD-Lite は現状 1ch のみのサポートですが、ch 固有の設定等は分離させています

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

－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

2.2.1. 受信バッファ設定

・CAN-ch0

アクセプタンス フィルタ番号 page=0	フォーマット	ID	受信メッセージ 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

・CAN-ch1（CANFD モジュールのみ）

アクセプタンス フィルタ番号 page=1(*1)	フォーマット	ID	受信メッセージ バッファ番号
0	標準(SID)	0x000	16
1	標準(SID)	0x001	17
2	標準(SID)	0x002	18
3	標準(SID)	0x003	19
4	拡張(EID)	0x0000000	20
5	拡張(EID)	0x0000001	21
6	拡張(EID)	0x0000002	22
7	拡張(EID)	0x0000003	23

CANFD では、アクセプタンスフィルタリスト（16 ルール×複数ページ）で設定した、ルールにマッチした（ID 等が一致した）データの格納先を最大 8 個設定できます。本サンプルプログラムでは、データの格納先を受信メッセージバッファに割り当てています。（受信メッセージバッファは、CAN-ch 共有で 32 個あり、アクセプタンスフィルタのルールと、受信メッセージバッファは自由に紐付けできます。）

(*)CAN-ch0 を使用しない様に設定（サンプルプログラムのデフォルト）した場合は、CAN-ch1 のページ番号は 0 になります

※CANFD_B, CANFD-Lite では、アクセプタンスフィルタリスト（16 ルール×2 ページ）
データの格納先は最大 2 個

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_reset();　　//CANFD モジュールのリセット
can0_init();　　//CAN-ch0 の初期化（CAN-ch0 使用時）(*1)
can1_init();　　//CAN-ch1 の初期化（CAN-ch1 使用時）(*1) [CANFD モジュールのみ]
receive_buf_conf();　　//受信メッセージバッファの設定

CAN の初期化をする。
(*1)どちらか先でも構いません

(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*n*_receive_rule_set(0, CAN_RULE_BUF, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000000);

can*n*_receive_rule_set(1, CAN_RULE_BUF, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000001);
…
can*n*_receive_rule_set(4, CAN_RULE_BUF, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000000);
…
can*n*_receive_rule_set(7, CAN_RULE_BUF, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000003);

n=0 or 1（CAN-ch に応じた値）[n=1 CANFD モジュールのみ]

can_operate();　　//全体の動作設定
can0_operate();　　//CAN-ch0 の動作設定(*2)
can1_operate();　　//CAN-ch1 の動作設定(*2) [CANFD モジュールのみ]

受信ルール番号 0~7 を受信設定する例。
ルール設定後、can_operate, can*n*_operate を呼び出す。
(*2)どちらが先でも構いません

(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 n_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。

※CAN-ch0, CAN-ch1 の両方の受信処理を行いたい場合は、can0_buf_receive(), can1_buf_receive()を呼び出してください。 [can1_buf_receive() CANFD モジュールのみ]

(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 = can0_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

CAN-ch0 で、送信バッファ 0 から ID=0x000、データ 0x01 を 1 バイト送信する。

※*n*:送信したい方の CAN-ch を指定してください [n=1 CANFD モジュールのみ]

2.4. データの受信

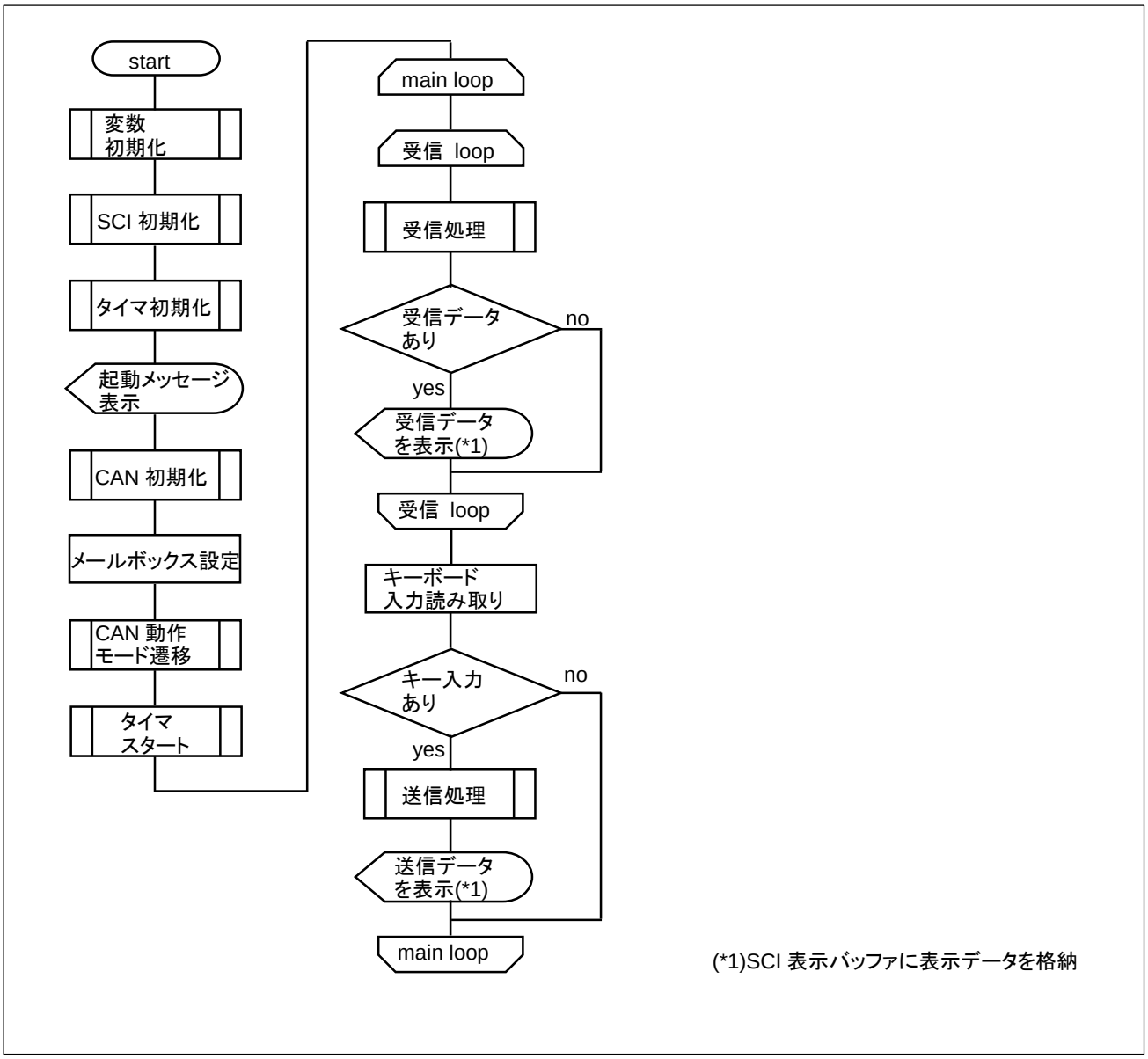
SAMPLE1 でのデータの受信に関しては、受信メッセージバッファまでのデータ格納に関しては、（初期化、メールボックス、受信ルール設定が済んでいれば）マイコンのハードウェアが行います。メールボックス、受信バッファにデータが格納されているかは、プログラムで受信関数を呼び出す事で確認を行っています。そのため、常に受信関数を呼び出して確認を行わないと、データの取りこぼしが生じる可能性があります。受信データの確認に CPU リソースを食うため、スムーズな手法とはいえないと考えます。

（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. RA, CANFD 系マイコン CANFD モジュール

CANFD モジュールでは、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。

RA マイコンでは、CAN モジュールの割り込みは、ICU イベントリンク設定で割り込み番号の割り当てを行います。「ソフトウェア編」3.2 章の GUI 上での設定で割り込み番号が割り当て済みです。

・CANFD モジュール(RA6M5)

割り込み 要求元	名称	割り込み条件	対応する 割り込み番号	呼び出される 割り込み関数名
CAN	CAN_RXF	受信 FIFO	4	intr_can_rxf(*1)
CAN-CAN0	CAN0_TX	チャネル送信	5	intr_can0_tx(*2)
CAN-CAN0	CAN0_COMFRX	共通 FIFO 受信	6	intr_can0_comfrx(*3)
CAN-CAN1	CAN1_TX	チャネル送信	7	intr_can1_tx(*4)
CAN-CAN1	CAN1_COMFRX	共通 FIFO 受信	8	intr_can1_comfrx(*5)

SAMPLE2～の受信は、
canfd/canfd.h 内で
・受信 FIFO を使用（デフォルト）
・共通 FIFO を使用
のどちらかを選択できる様になっています。受信 FIFO を選択した場合、受信は CAN ch0, ch1 共(*1)の割り込み関数で処理されます。共通 FIFO を使った場合は、ch0 の受信割り込みは(*3)で、ch1 の受信割り込みは(*5)で処理されます。また、送信の割り込み処理は、ch0 が(*2)、ch1 が(*4)で処理されます。

ここで注意が必要で、共通 FIFO を使った場合、ch0 の割り込みが必ず CAN0_TX, CAN0_COMFRX で処理される訳ではありません。共通 FIFO と割り込みの関係は以下の様になっています。

共通 FIFO	対応する 送信割り込み	対応する 受信割り込み
CFIFO0	CAN0_TX	CAN0_COMFRX
CFIFO1	CAN0_TX	CAN0_COMFRX
CFIFO2	CAN0_TX	CAN0_COMFRX
CFIFO3	CAN1_TX	CAN1_COMFRX
CFIFO4	CAN1_TX	CAN1_COMFRX
CFIFO5	CAN1_TX	CAN1_COMFRX

CAN-ch1 の受信を CFIFO1 で行った場合、受信割り込みは CAN0_COMFRX で処理されます
※本サンプルプログラムでは、対応が判り易い様に、CAN-ch0 の送信・受信を、CAN0_TX, CAN0_COMFRX の割り込みで処理しています。CAN-ch1 も同様に、CAN1_TX, CAN1_COMFRX で処理しています。

SAMPLE2 では、

CAN-ch	送信で 使用している 共通 FIFO 番号	受信で 使用している 共通 FIFO 番号	備考
CAN-ch0	0	1	ch0 に 0~2 の FIFO 番号を割り当て
CAN-ch1	3	4	ch1 に 3~5 の FIFO 番号を割り当て

としています。そのため、ch0 側が CAN0_TX, CAN0_COMFRX、ch1 側が CAN1_TX, CAN1_COMFRX 割り込みで処理されるようになっていますが、本来 CAN の物理 ch*n*(ch0, ch1)と CAN*n*_TX, CAN*n*_COMFRX の *n* には関連がありません。

3.2.2. RA, CANFD_B 系マイコン CANFD_B モジュール

CANFD_B モジュールでも、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。

RA マイコンでは、CAN モジュールの割り込みは、ICU イベントリンク設定で割り込み番号の割り当てを行います。「ソフトウェア編」3.2 章の GUI 上での設定で割り込み番号が割り当て済みです。

・CANFD_B モジュール

割り込み 要求元	名称	割り込み条件	対応する 割り込み番号	呼び出される 割り込み関数名
CAN	CAN_RXF	受信 FIFO	4	intr_can_rxf
CAN-CAN0	CAN0_TX	チャネル送信	5	intr_can0_tx
CAN-CAN0	CAN0_COMFRX	共通 FIFO 受信	6	intr_can0_comfrx
CAN-CAN0	CAN0_RXMB	受信メッセージバッファ	7	intr_can0_rxmb

SAMPLE2～の受信は、
canfd_b/canfd_b.h 内で
・受信 FIFO を使用（デフォルト）
・受信メッセージバッファを使用(*1)
・共通 FIFO を使用(*2)

のいずれかを選択できる様になっています。

(*1)CANFD モジュールでは、受信メッセージバッファと割り込みの紐付けが出来ない（受信メッセージバッファ割り込みが存在しない）ために、この選択肢を用意していませんが、CANFD_B では受信メッセージバッファを使用した際でも受信割り込みの使用が可能です

(*2)共通 FIFO を使った受信、受信時の割り込みは CANFD_B モジュールでも有効ですが、CANFD_B モジュールでは共通 FIFO は 1 本しかないため、受信で共通 FIFO を使った場合、送信で共通 FIFO を使う事が出来ません（この選択肢を選んだ場合、送信時の割り込みが使用できません）

3.2.3. RX, CANFD-Lite 系マイコン CANFD-Lite モジュール

CANFD-Lite モジュールの割り込みは、CANFD_B モジュールと同等です。

RX マイコンでは、CANFD-Lite の割り込みは、グループ BL2 割り込みとしてグループ化されており、割り込みベクタは 1 個で共有。割り込みフラグで割り込みの要求元を特定する方式となっています。

・CANFD-Lite モジュール

割り込み 要求元	名称	割り込み条件	呼び出される 割り込み関数名
CANFD0	CFRI	共通 FIFO 受信	intr_canfd0_cfri_s
CANFD	RFRI	受信 FIFO	intr_canfd_rfri_s
CANFD0	CHTI	チャネル送信	intr_canfd0_chti_s
CANFD	RMRI	受信メッセージバッファ	intr_canfd_rmri_s

・受信 FIFO を使用（デフォルト）
・受信メッセージバッファを使用
・共通 FIFO を使用

の 3 種のどの受信方法を選択可能な点や、共通 FIFO を受信で使用した場合、送信割り込みが使えない点等は、CANFD_B と同様です。

3.3. 受信ルールの設定

CANFD モジュール系では、SAMPLE1 では、受信メッセージバッファ・送信メッセージバッファを使用していましたが、SAMPLE2 では受信には「受信 FIFO」または「共通 FIFO」。送信には「共通 FIFO」（共通 FIFO には送信メッセージバッファを割り当て、送信メッセージバッファの手前に FIFO が追加されるイメージ）を送信の設定で使用する事とします。FIFO は、128 段（CANFD_B, CANFD-Lite では 48 段）まで設定可能ですが、本サンプルプログラムでは 8 段に設定しています。

CANFD_B, CANFD-Lite モジュール系では、CANFD データを取り扱わない SAMPLE2~3 では、FIFO を 8 段に設定、CANFD データを取り扱う SAMPLE4 では FIFO を 4 段に設定しています。

（※FIFO 段数設定に関しては、SAMPLE4 の項で詳細を記載）

3.3.1. 受信ルール設定(CANFD モジュール系）

CANFD では、アクセプタンスフィルタリスト（AFL, ID 等の条件比較）にマッチしたデータを、

・受信メッセージバッファ（最大 32）

・受信 FIFO（8 個）

・共通 FIFO（6 個）

に、最大 8 個の宛先を指定して送る事が出来ます。サンプルプログラムでは宛先は以下の設定です。

CAN-ch	SAMPLE1	SAMPLE2~ 受信 FIFO を選択	SAMPLE2~ 共通 FIFO を選択
0	受信メッセージバッファ 0~7	RXFIFO(0)	CFIFO(1)
1	受信メッセージバッファ 16~23	RXFIFO(1)	CFIFO(4)

・CAN-ch0

アクセプタンス フィルタ番号 page=0	フォーマット	ID	受信 FIFO 使用時	共通 FIFO 使用時
0	標準(SID)	0x000	RXFIFO(0)	CFIFO(1)
1	標準(SID)	0x001	RXFIFO(0)	CFIFO(1)
2	標準(SID)	0x002	RXFIFO(0)	CFIFO(1)
3	標準(SID)	0x003	RXFIFO(0)	CFIFO(1)
4	拡張(EID)	0x0000000	RXFIFO(0)	CFIFO(1)
5	拡張(EID)	0x0000001	RXFIFO(0)	CFIFO(1)
6	拡張(EID)	0x0000002	RXFIFO(0)	CFIFO(1)
7	拡張(EID)	0x0000003	RXFIFO(0)	CFIFO(1)

・CAN-ch1

アクセプタンス フィルタ番号 page=1	フォーマット	ID	受信 FIFO 使用時	共通 FIFO 使用時
0	標準(SID)	0x000	RXFIFO(1)	CFIFO(4)
1	標準(SID)	0x001	RXFIFO(1)	CFIFO(4)
2	標準(SID)	0x002	RXFIFO(1)	CFIFO(4)
3	標準(SID)	0x003	RXFIFO(1)	CFIFO(4)
4	拡張(EID)	0x0000000	RXFIFO(1)	CFIFO(4)
5	拡張(EID)	0x0000001	RXFIFO(1)	CFIFO(4)
6	拡張(EID)	0x0000002	RXFIFO(1)	CFIFO(4)
7	拡張(EID)	0x0000003	RXFIFO(1)	CFIFO(4)

3.3.2. 送信設定(CANFD モジュール系)

CAN-ch	SAMPLE1	SAMPLE2~
0	送信メッセージバッファ 0~3	共通 FIFO(0)
1	送信メッセージバッファ 64~67	共通 FIFO(3)

3.3.3. 受信ルール設定(CANFD_B, CANFD-Lite モジュール系)

CANFD_B, CANFD-Lite では、アクセプタンスフィルタリスト(AFL, ID 等の条件比較)にマッチしたデータを、

・受信メッセージバッファ(最大 32)

・受信 FIFO(2 個)

・共通 FIFO(1 個)

に、最大 2 個の宛先を指定して送る事が出来ます。サンプルプログラムでは宛先は以下の設定です。

CAN-ch	SAMPLE1	SAMPLE2~ 受信 FIFO を選択	SAMPLE2 受信メッセージバッファを選択	SAMPLE2~ 共通 FIFO を選択
0	受信メッセージバッファ 0~7	受信 FIFO(0)	受信メッセージバッファ 0~7	共通 FIFO(0)

・CAN-ch0

アクセプタンスフィルタ番号 page=0	フォーマット	ID	受信 FIFO 使用時	受信メッセージバッファ使用時 番号	共通 FIFO 使用時
0	標準(SID)	0x000	RXFIFO(0)	0	CFIFO(0)
1	標準(SID)	0x001	RXFIFO(0)	1	CFIFO(0)
2	標準(SID)	0x002	RXFIFO(0)	2	CFIFO(0)
3	標準(SID)	0x003	RXFIFO(0)	3	CFIFO(0)
4	拡張(EID)	0x0000000	RXFIFO(0)	4	CFIFO(0)
5	拡張(EID)	0x0000001	RXFIFO(0)	5	CFIFO(0)
6	拡張(EID)	0x0000002	RXFIFO(0)	6	CFIFO(0)
7	拡張(EID)	0x0000003	RXFIFO(0)	7	CFIFO(0)

3.3.4. 送信設定(CANFD_B, CANFD-Lite モジュール系)

CAN-ch	SAMPLE1	SAMPLE2~4
0	送信メッセージバッファ 0~3	共通 FIFO(0)

※受信で共通 FIFO を選択した場合は、SAMPLE2~での送信に関しては送信メッセージバッファを使用します

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

(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
can0_receive_rule_set(0, **CAN_RULE_RXFIFO0**, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000000); //CAN-ch0

can1_receive_rule_set(0, **CAN_RULE_RXFIFO1**, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000000); //CAN-ch1　[CANFD モジュールのみ]
…

受信ルール設定時、受信バッファではなく CAN-ch0：受信 FIFO(0), CAN-ch1：受信 FIFO(1)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*n*_cfifo_send**(s_format, CAN_DATA_FRAME, s_id, s_dlc, &s_data[0]);

共通 FIFO を使用する関数で送信。
[CANFD_B, CANFD-Lite モジュール]受信に共通 FIFO を使った場合は、送信は SAMPLE1 同様 can0_buf_send()での送信となります（送信割り込みは使用できない）

(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]

(a)受信 FIFO を使う場合（デフォルト）

intr_can_rxf_s() [RA]
intr_canfd_rfri_s() [RX]

```
while(R_CANFD->CFDRFSTS_b[CAN0_RX_RXFIFO_NO].RFEMP != 1) [RA]
while(CANFD.RFSR[CAN0_RX_RXFIFO_NO].BIT.EMPTY != 1) [RX]
{
    r_ret = can0_rxfifo_receive(&ide, &rtr, &id, &data[0], &ts);       //受信 FIFO を使った受信
    [受信データの表示]
}
```

FIFO が空であるというフラグが立つまでループ

[CANFD モジュールのみ CAN-ch1 側]

```
while(R_CANFD->CFDRFSTS_b[CAN1_RX_RXFIFO_NO].RFEMP != 1) [RA]
{
    r_ret = can1_rxfifo_receive(&ide, &rtr, &id, &data[0], &ts);       //受信 FIFO を使った受信
    [受信データの表示]
}
```

受信 FIFO には複数のデータが格納されているかもしれないので、FIFO が空になるまでデータの読み出し(can*n*_rxfifo_receive の実行）を行います。

(b)共通 FIFO を使う場合

intr_can0_comfrx_s() [RA]
intr_canfd0_cfri_s() [RX]

while(R_CANFD->CFDCFSTS_b[CAN0_RX_CFIFO_NO].CFEMP != 1) [RA]
while(CANFD.CFSR0.BIT.EMPTY != 1) [RX]
{
    r_ret = **can0_cfifo_receive**(&ide, &rtr, &id, &data[0], &ts);　　//共通 FIFO を使った受信
    [受信データの表示]
}

intr_can1_comfrx_s() [RA]　[CANFD モジュールのみ]

while(R_CANFD->CFDCFSTS_b[CAN1_RX_CFIFO_NO].CFEMP != 1) [RA]
{
    r_ret = **can1_cfifo_receive**(&ide, &rtr, &id, &data[0], &ts);　　//共通 FIFO を使った受信
    [受信データの表示]
}

共通 FIFO を使う場合も、FIFO が空になるまでデータの読み出しを行います（受信 FIFO を使う場合と同様です）。

CANFD モジュールでは、CAN-ch1 の共通 FIFO を使った受信割り込みは、別関数となっています。（受信 FIFO を使った場合は、CAN-ch0, CAN-ch1 どちらの受信でも同じ割り込み関数に飛んできます。）

受信に共通 FIFO を使う場合、(2)の受信ルール設定の際に FIFO 区分を共通 FIFO を使う様に設定する必要があります。

具体的には、

can0_receive_rule_set(0, **CAN_RULE_CFIFO1**,… [CANFD モジュール]
can1_receive_rule_set(0, **CAN_RULE_CFIFO4**,… [CANFD モジュール]

can0_receive_rule_set(0, **CAN_RULE_CFIFO0**,… [CANFD_B, CANFD-Lite モジュール]

の様になります。

※CANFD_B, CANFD-Lite モジュール系は共通 FIFO は 1 本しかないので CAN_RULE_CFIFO0 のみ指定可能です

(c)受信メッセージバッファを使う場合　[CANFD_B, CANFD-Lite モジュールのみ]

intr_can0_rxmb_s() [RA]
intr_canfd_rmri_s() [RX]

```
for(i=0; i<8; i++)
{
    r_ret = can0_buf_receive((unsigned char)i, &r_ide, &r_rtr, &r_id, &data[0], &r_ts); //受信メッセージバッファを使った受信
    [受信データの表示]
}
```

受信メッセージバッファを使う場合、受信データにより受信メッセージバッファの 0~7 のいずれかに格納されますので、受信割り込みの際 8 回受信関数を呼び出します。（データを受信していないときは、戻り値 0 です。）

受信に受信メッセージバッファを使う場合、(2)の受信ルール設定のメッセージ格納先として受信メッセージバッファを使う様に設定する必要があります。

can0_receive_rule_set(0, **CAN_RULE_BUF**,…

(i2)送信割り込み関数[SAMPLE1 との相違点]

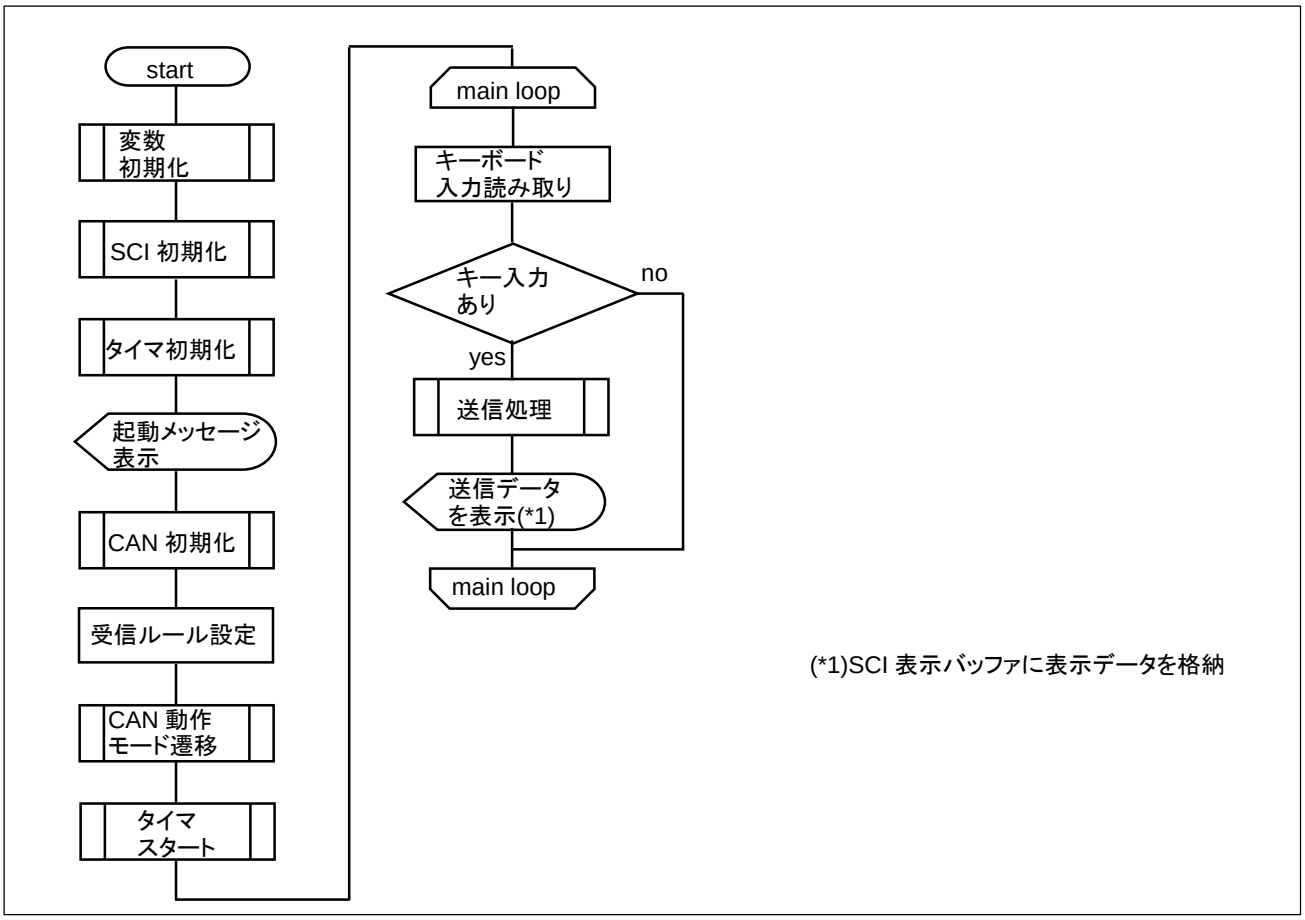
intr_can0_tx_s() [RA]
intr_can1_tx_s() [RA]
intr_canfd0_chti_s() [RX]

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。

3.5. SAMPLE2 フローチャート

－処理フロー－

メイン関数　main_s2()

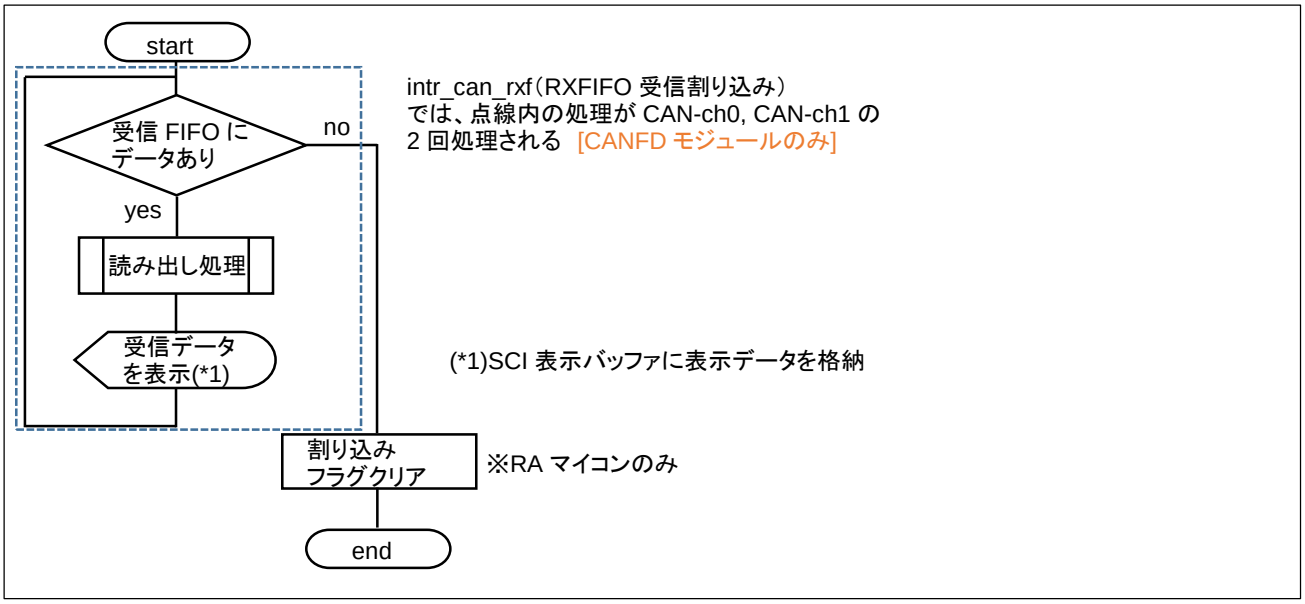


受信割り込み関数（受信 FIFO、共通 FIFO 使用）

（CANFD モジュール系） intr_can_rxf(),intr_can0_comfrx(), intr_can1_comfrx()

（CANFD_B モジュール系） intr_can_rxf(), intr_can0_comrxf()

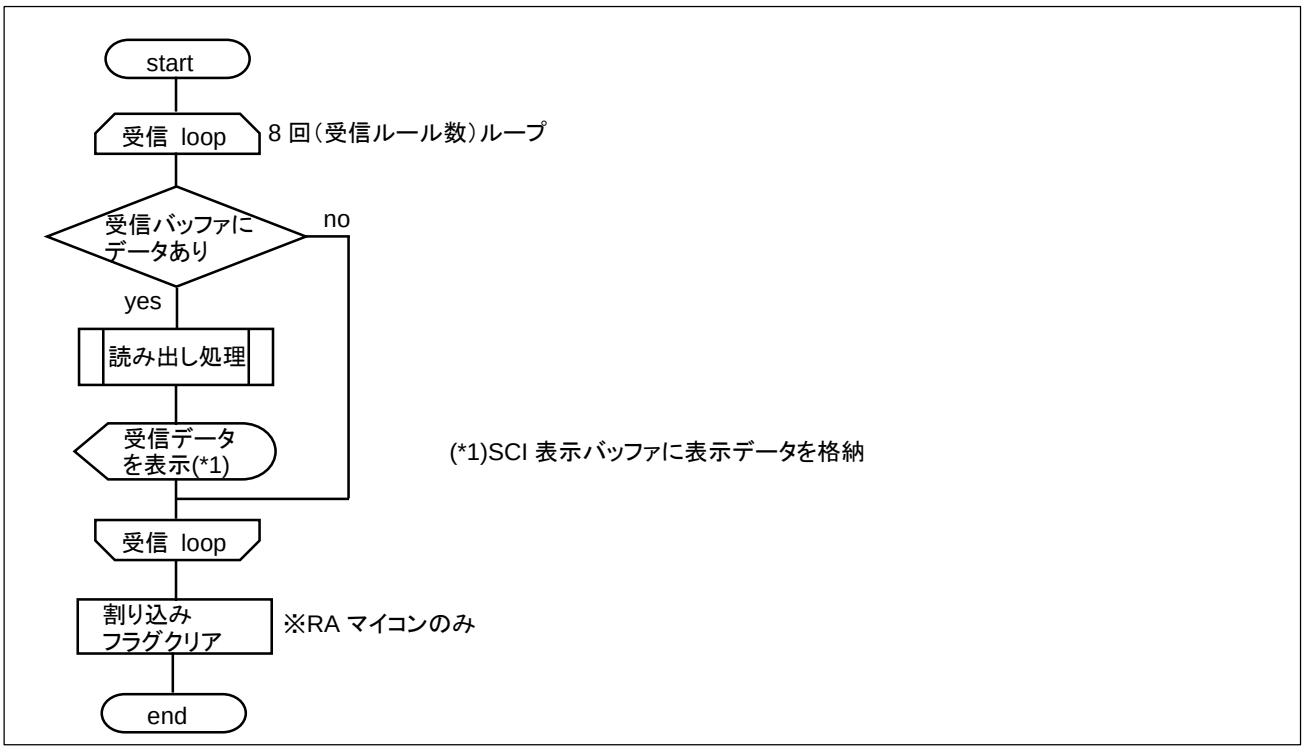
（CANFD-Lite モジュール系） intr_canfd0_cfri_s(), intr_canfd_rfri_s()



受信割り込み関数（受信メッセージバッファ使用）

（CANFD_B モジュール系） intr_can0_rxmb()

（CANFD-Lite モジュール系） intr_canfd_rmri_s()

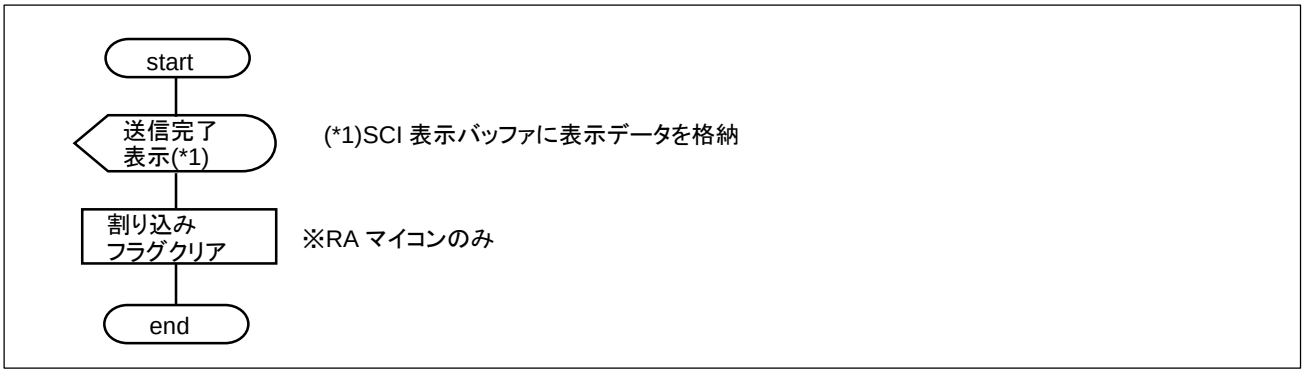


送信完了割り込み関数

（CANFD モジュール系） Intr_can0_tx(), intr_can1_tx()

（CANFD_B モジュール系） intr_can0_tx()

（CANFD-Lite モジュール系） intr_canfd0_chti_s()



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

4.2.1. 受信ルール設定(CANFD モジュール系)

・CAN-ch0

アクセプタンス フィルタ番号 page=0	フォーマット	データフレーム／ リモートフレーム	ID	受信 FIFO 使用時	共通 FIFO 使用時
0	標準(SID)	データフレーム	0x000	RXFIFO(0)	CFIFO(1)
1	標準(SID)	データフレーム	0x001	RXFIFO(0)	CFIFO(1)
2	標準(SID)	データフレーム	0x002	RXFIFO(0)	CFIFO(1)
3	標準(SID)	データフレーム	0x003	RXFIFO(0)	CFIFO(1)
4	標準(SID)	リモートフレーム	0x000	RXFIFO(0)	CFIFO(1)
5	標準(SID)	リモートフレーム	0x001	RXFIFO(0)	CFIFO(1)
6	標準(SID)	リモートフレーム	0x002	RXFIFO(0)	CFIFO(1)
7	標準(SID)	リモートフレーム	0x003	RXFIFO(0)	CFIFO(1)
8	拡張(EID)	データフレーム	0x0000000	RXFIFO(0)	CFIFO(1)
9	拡張(EID)	データフレーム	0x0000001	RXFIFO(0)	CFIFO(1)
10	拡張(EID)	データフレーム	0x0000002	RXFIFO(0)	CFIFO(1)
11	拡張(EID)	データフレーム	0x0000003	RXFIFO(0)	CFIFO(1)
12	拡張(EID)	リモートフレーム	0x0000000	RXFIFO(0)	CFIFO(1)
13	拡張(EID)	リモートフレーム	0x0000001	RXFIFO(0)	CFIFO(1)
14	拡張(EID)	リモートフレーム	0x0000002	RXFIFO(0)	CFIFO(1)
15	拡張(EID)	リモートフレーム	0x0000003	RXFIFO(0)	CFIFO(1)

・CAN-ch1

アクセプタンス フィルタ番号 page=1	フォーマット	データフレーム／ リモートフレーム	ID	受信 FIFO 使用時	共通 FIFO 使用時
0	標準(SID)	データフレーム	0x000	RXFIFO(1)	CFIFO(4)
1	標準(SID)	データフレーム	0x001	RXFIFO(1)	CFIFO(4)
2	標準(SID)	データフレーム	0x002	RXFIFO(1)	CFIFO(4)
3	標準(SID)	データフレーム	0x003	RXFIFO(1)	CFIFO(4)
4	標準(SID)	リモートフレーム	0x000	RXFIFO(1)	CFIFO(4)
5	標準(SID)	リモートフレーム	0x001	RXFIFO(1)	CFIFO(4)
6	標準(SID)	リモートフレーム	0x002	RXFIFO(1)	CFIFO(4)
7	標準(SID)	リモートフレーム	0x003	RXFIFO(1)	CFIFO(4)
8	拡張(EID)	データフレーム	0x0000000	RXFIFO(1)	CFIFO(4)
9	拡張(EID)	データフレーム	0x0000001	RXFIFO(1)	CFIFO(4)
10	拡張(EID)	データフレーム	0x0000002	RXFIFO(1)	CFIFO(4)
11	拡張(EID)	データフレーム	0x0000003	RXFIFO(1)	CFIFO(4)
12	拡張(EID)	リモートフレーム	0x0000000	RXFIFO(1)	CFIFO(4)
13	拡張(EID)	リモートフレーム	0x0000001	RXFIFO(1)	CFIFO(4)
14	拡張(EID)	リモートフレーム	0x0000002	RXFIFO(1)	CFIFO(4)
15	拡張(EID)	リモートフレーム	0x0000003	RXFIFO(1)	CFIFO(4)

4~7, 12~15 をリモートフレーム受信用に割り当て。

※RSCAN では、後ろにリモートフレームの設定を追加していますが、CANFD では途中に追加しています。プログラムの初期設定で、拡張 ID を扱わないように設定した場合、アクセプタンスフィルタ番号は 0~7 が設定されます。アクセプタンスフィルタは、未設定の項目がある場合（例えば 0~3 と 8~11 を設定、4~7 が未設定）未設定の後のルールにマッチしない（4~7 が未設定だと、8~11 にはマッチしない）ためです。アクセプタンスフィルタの設定(can*n*_receive_rule_set)を行う場合は、0 から詰めていき、途中の番号に空きが出ない様にしてください。

4.2.2. 受信ルール設定(CANFD_B, CANFD-Lite モジュール系）

・CAN-ch0

アクセプタンスフィルタ番号 page=0	フォーマット	データフレーム／リモートフレーム	ID	受信 FIFO 使用時	受信メッセージバッファ使用時番号	共通 FIFO 使用時
0	標準(SID)	データフレーム	0x000	RXFIFO(0)	0	CFIFO(0)
1	標準(SID)	データフレーム	0x001	RXFIFO(0)	1	CFIFO(0)
2	標準(SID)	データフレーム	0x002	RXFIFO(0)	2	CFIFO(0)
3	標準(SID)	データフレーム	0x003	RXFIFO(0)	3	CFIFO(0)
4	標準(SID)	リモートフレーム	0x000	RXFIFO(0)	4	CFIFO(0)
5	標準(SID)	リモートフレーム	0x001	RXFIFO(0)	5	CFIFO(0)
6	標準(SID)	リモートフレーム	0x002	RXFIFO(0)	6	CFIFO(0)
7	標準(SID)	リモートフレーム	0x003	RXFIFO(0)	7	CFIFO(0)
8	拡張(EID)	データフレーム	0x0000000	RXFIFO(0)	8	CFIFO(0)
9	拡張(EID)	データフレーム	0x0000001	RXFIFO(0)	9	CFIFO(0)
10	拡張(EID)	データフレーム	0x0000002	RXFIFO(0)	10	CFIFO(0)
11	拡張(EID)	データフレーム	0x0000003	RXFIFO(0)	11	CFIFO(0)
12	拡張(EID)	リモートフレーム	0x0000000	RXFIFO(0)	12	CFIFO(0)
13	拡張(EID)	リモートフレーム	0x0000001	RXFIFO(0)	13	CFIFO(0)
14	拡張(EID)	リモートフレーム	0x0000002	RXFIFO(0)	14	CFIFO(0)
15	拡張(EID)	リモートフレーム	0x0000003	RXFIFO(0)	15	CFIFO(0)

CANFD 同様、標準 ID のみ取り扱う設定としてもアクセプタンスフィルタ番号の途中に欠番が出ない様に、0~7 を標準 ID、8~15 を拡張 ID の設定としています。

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*n*_receive_rule_set(4, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, **CAN_REMOTE_FRAME**, 0x00000000);
…

受信ルール番号 4~7, 12~15 をリモートフレーム受信向けに追加。
[n=1 は CANFD モジュールのみ]

(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*n*_cfifo_send(s_format, **s_rtr**, s_id, s_dlc, &s_data[0]);

SAMPLE2 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。(0~3 の時は s_rtr=0, q~r の時は s_rtr=1)

[CANFD_B, CANFD-Lite モジュール]受信に共通 FIFO を使った場合は、送信は SAMPLE1 同様 can0_buf_send()での送信となります(送信割り込みは使用できない)

(i1)受信割り込み関数[SAMPLE2 との相違点]

FIFO が空になるまでループ(SAMPLE2 同様）

```
{
    ret = cann_rxfifo_receive(&ide, &rtr, &id, &data[0], &ts);       //受信 FIFO を使った受信
    if(rtr ==  CAN_DATA_FRAME)
    {
        [データフレームの場合：受信データの表示]
    }
    else if(rtr ==  CAN_REMOTE_FRAME)
    {
        [リモートフレームの場合：データ表示，データ送信]
        dlc = ret;  //送信バイト数は受信した DLC 値とする
        ret2 = cann_cfifo_send(ide, CAN_DATA_FRAME, id, dlc, &remote_frame_response_data[0]);
    }
}
```

FIFO が空になるまでデータ受信を行い、受信したデータが rtr == CAN_DATA_FRAME(0)の時は、SAMPLE2 同様、受信データを表示して終了。rtr == CAN_REMOTE_FRAME(1)の場合は、画面表示と「データフレーム」の送信を行います。

（リモートフレームに対する応答なので、送信するデータは「データフレーム」とします。）

送信するデータバイト数は、リモートフレーム要求元から送られてきた DLC 値を使用します。

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(cann_cfifo_send)で行う。
[CANFD_B, CANFD-Lite モジュール]受信に共通 FIFO を使った場合は、送信は SAMPLE1 同様 can0_buf_send()での送信となります

上記は、受信に受信 FIFO を使った場合ですが、共通 FIFO、受信メッセージバッファを使った場合でも同様の処理となります。

(i2)送信割り込み関数[SAMPLE2 との相違点]

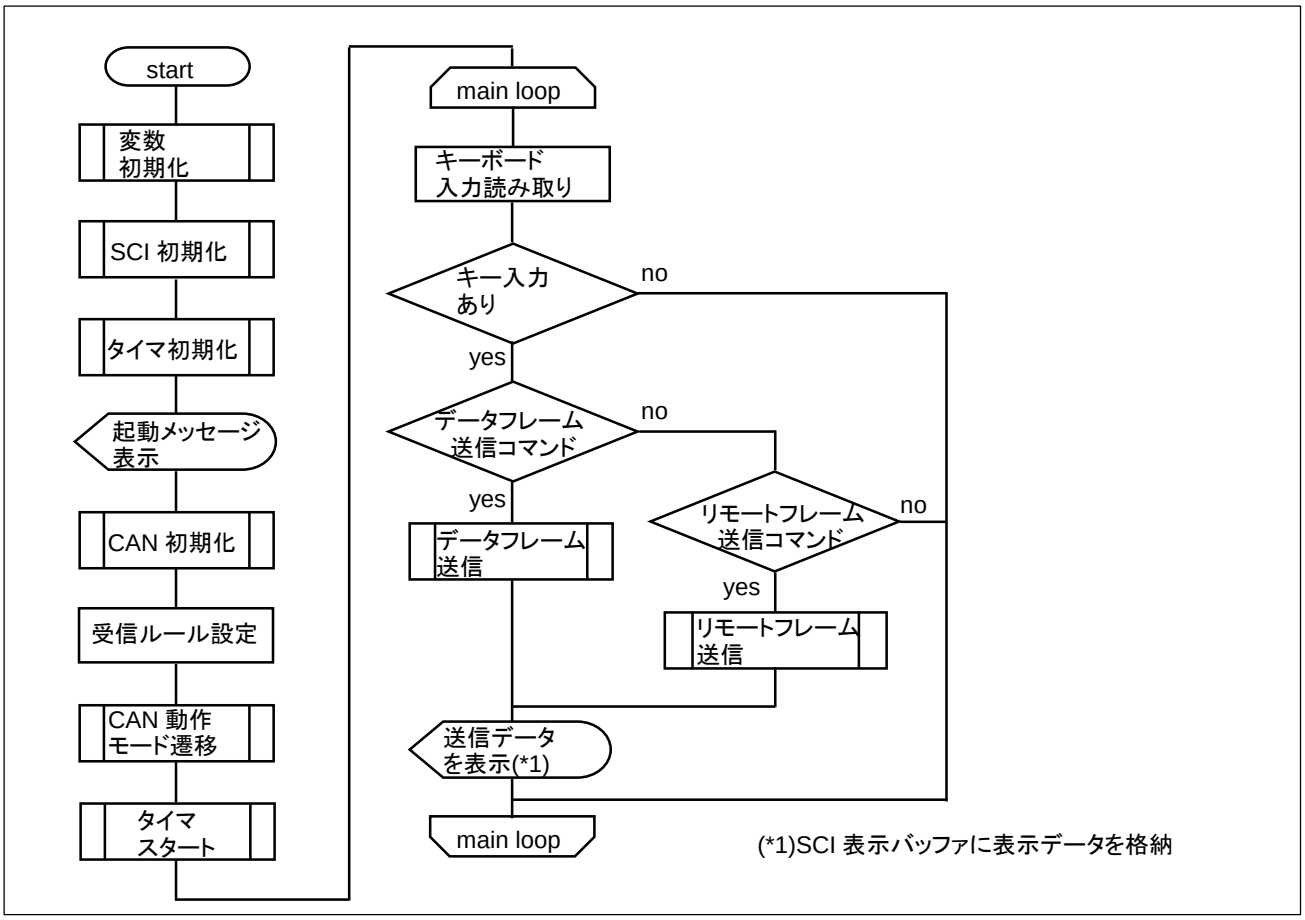
```
if(can0_remote_frame_request != 1) sciPrintBuf("--¥n");
```

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()

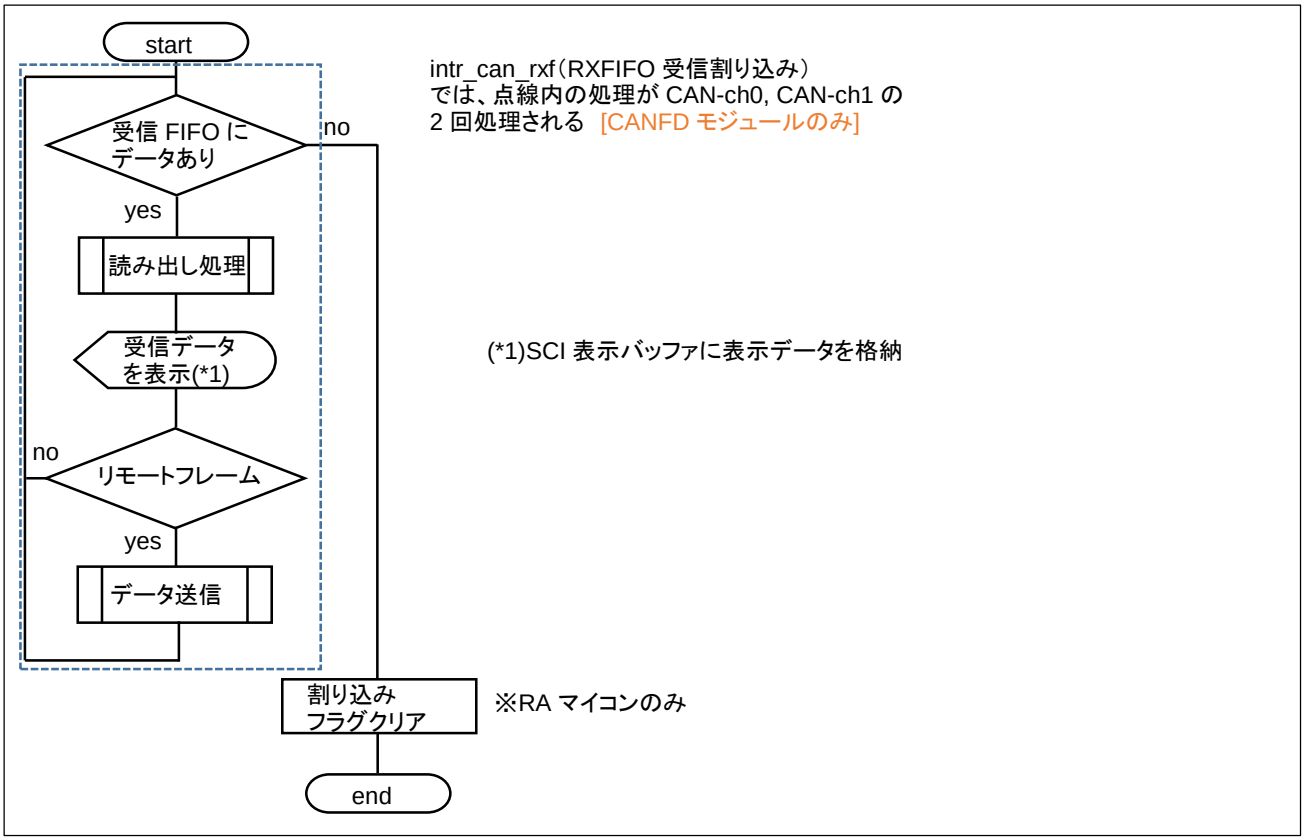


受信割り込み関数（受信 FIFO、共通 FIFO 使用）

（CANFD モジュール系） intr_can_rxf(),intr_can0_comfrx(), intr_can1_comfrx()

（CANFD_B モジュール系） intr_can_rxf(), intr_can0_comrxf()

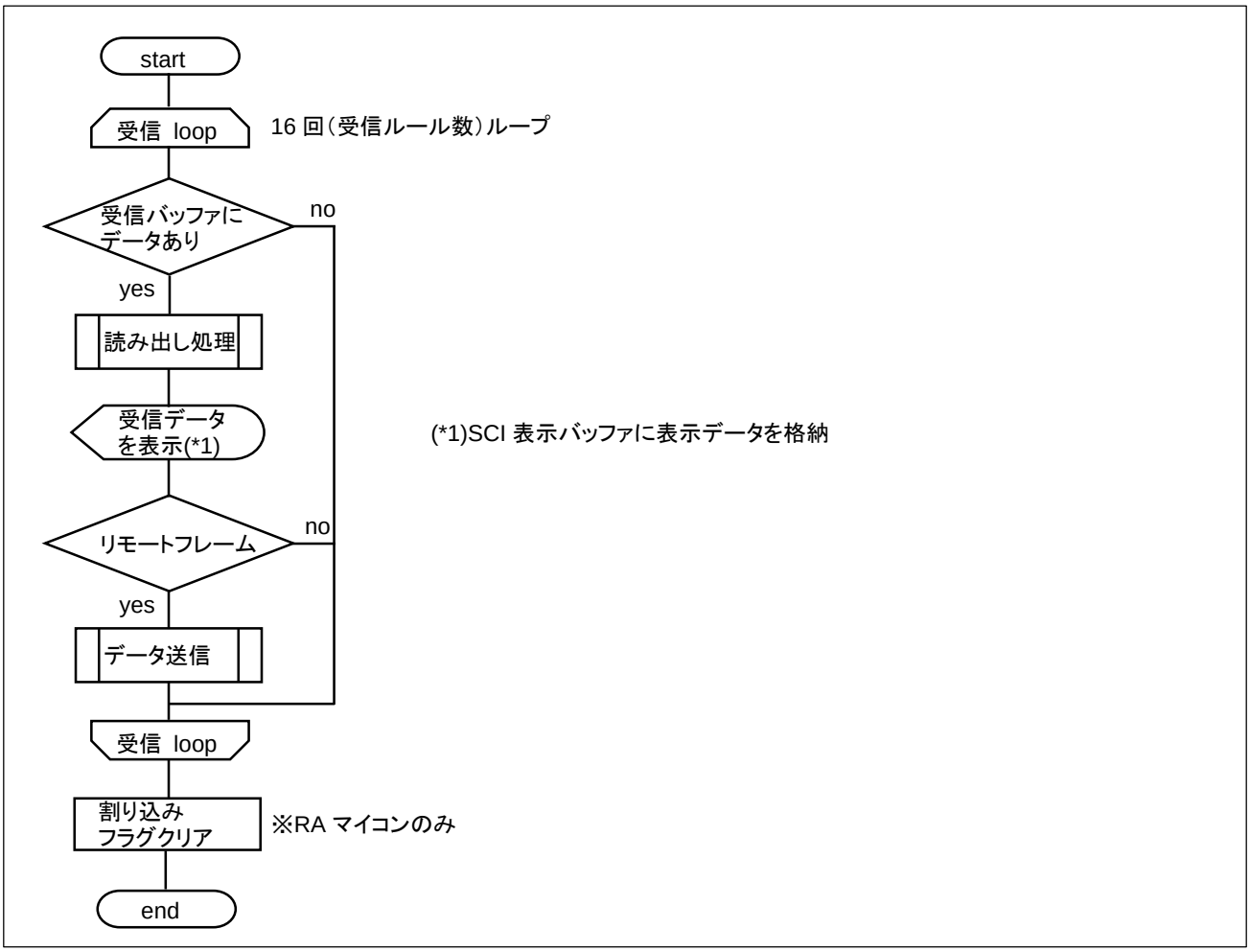
（CANFD-Lite モジュール系） intr_canfd0_cfri_s(), intr_canfd_rfri_s()



受信割り込み関数（受信メッセージバッファ使用）

（CANFD_B モジュール系） intr_can0_rxmb()

（CANFD-Lite モジュール系） intr_canfd_rmri_s()



5. サンプルプログラムの説明(SAMPLE4)

5.1. プログラム仕様

・データフレーム（CANFD フレーム）の送信
・データフレーム（CANFD 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（CANFD フレーム）を行う）する

を行うサンプルプログラムとします。動作としては、SAMPLE3 と同様ですが、赤字の部分を CANFD フレームを使って送受信します。（リモートフレームの送信のみ、従来の CAN フレームでの通信となります。…CANFD の仕様）

－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

キーボードからの入力	ID	送信バイト数(DLC)
0	0x0000000	1
1	0x0000001	8
2	0x0000002	16
3	0x0000003	64

送信データは、0x00, 0x01, 0x02, 0x03.....（0 からインクリメントしたデータ）で、最大 64 バイトです。

・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	8
e	0x0000002	16
r	0x0000003	64

・リモートフレーム応答
0xFF, 0xFE, 0xFD, 0xFC.....（0xFF からデクリメントしたデータ）
を、要求元の送信要求バイト数に応じて返答
（DLC=8 のときは、0xFF FE FD FC FB FA F9 F8 を返す）

CANFD では、データのパケットサイズは最大 64 バイトに拡張されています。SAMPLE4 では、最大 64 バイトのデータを取り扱う様にしています。

SAMPLE4 のプログラムをボードに書き込んで起動すると、通信速度を聞かれますので、通信を行う 2 台のボードで同じ値を設定してください。

※通信速度設定が異なるボード同士は通信が通りません

```
HSBRA6M5F176 CAN Starter kit program boot.

Copyright (C) 2022 HokutoDenshi. All Rights Reserved.



SAMPLE4: CANFD [Data frame] send/receive program(with interrupt, use FIFO).

                [Remote frame] request/response program(with interrupt, use FIFO).



CAN ID mode -> EID                                    通信速度の入力

CANFD speed [Mbps](2-5,8, default:2)) >
```

のプロンプトに対し、2~5 の数値をキーボードから入力するか、[Enter]を押して先に進んでください。
（[Enter]を押した場合は、デフォルト値 2Mbps 設定となります。）

```
CAN transceiver delay compensation(y/n, default:n) >    [Enter]
                                                        [Enter]
```

とりあえず動作を見る場合、[Enter]で先へ進めて問題ありません。

※設定項目に関しては、「CAN スタータキット RX/RA　CAN スタータキット SmartRX 取扱説明書」の 3.5 章に説明がありますので、そちらを参照してください

```
Input summary:

  CANFD speed [kbps] > 2000
  CAN transceiver delay compensation > n
  [not use] CAN transceiver secondary sampling point > delay+offset
  [not use] CAN transceiver secondary sampling point delay > 0
  CAN transceiver RX edge filter > n

Command Usage:
  0123: Data flame send
  qwer: Remote frame send(data request)
  z: LED blink test(for board identify)
  s: send format EID <-> SID
  (Push-SW: Data frame send [=keyboard 0])
```

CAN 通信速度の入力が終了すると、入力項目の一覧(Input summary:)が表示され、コマンド一覧(Command Usage:)が表示されます。

この状態は、通信が行える状態（受信待機）となります。

※起動後、CANFD のパラメータ設定の入力を完了していない状態では、データを受信できませんので注意願います

※main/main_s4.c 内の `#define CANFD_PARAMETER_SETTING` の行をコメントアウトした場合、起動時のパラメータの対話入力は省略され、common/can_operation.h 内で定義された速度他のパラメータが有効になります

5.2. 受信ルール設定

5.2.1. 受信ルール設定(CANFD モジュール系)

・CAN-ch0

アクセプタンス フィルタ番号 page=0	フォーマット	データフレーム／ リモートフレーム	ID	受信 FIFO 使用時	共通 FIFO 使用時
0	標準(SID)	データフレーム	0x000	RXFIFO(0)	CFIFO(1)
1	標準(SID)	データフレーム	0x001	RXFIFO(0)	CFIFO(1)
2	標準(SID)	データフレーム	0x002	RXFIFO(0)	CFIFO(1)
3	標準(SID)	データフレーム	0x003	RXFIFO(0)	CFIFO(1)
4	標準(SID)	リモートフレーム	0x000	RXFIFO(0)	CFIFO(1)
5	標準(SID)	リモートフレーム	0x001	RXFIFO(0)	CFIFO(1)
6	標準(SID)	リモートフレーム	0x002	RXFIFO(0)	CFIFO(1)
7	標準(SID)	リモートフレーム	0x003	RXFIFO(0)	CFIFO(1)
8	拡張(EID)	データフレーム	0x0000000	RXFIFO(0)	CFIFO(1)
9	拡張(EID)	データフレーム	0x0000001	RXFIFO(0)	CFIFO(1)
10	拡張(EID)	データフレーム	0x0000002	RXFIFO(0)	CFIFO(1)
11	拡張(EID)	データフレーム	0x0000003	RXFIFO(0)	CFIFO(1)
12	拡張(EID)	リモートフレーム	0x0000000	RXFIFO(0)	CFIFO(1)
13	拡張(EID)	リモートフレーム	0x0000001	RXFIFO(0)	CFIFO(1)
14	拡張(EID)	リモートフレーム	0x0000002	RXFIFO(0)	CFIFO(1)
15	拡張(EID)	リモートフレーム	0x0000003	RXFIFO(0)	CFIFO(1)

・CAN-ch1

アクセプタンス フィルタ番号 page=1	フォーマット	データフレーム／ リモートフレーム	ID	受信 FIFO 使用時	共通 FIFO 使用時
0	標準(SID)	データフレーム	0x000	RXFIFO(1)	CFIFO(4)
1	標準(SID)	データフレーム	0x001	RXFIFO(1)	CFIFO(4)
2	標準(SID)	データフレーム	0x002	RXFIFO(1)	CFIFO(4)
3	標準(SID)	データフレーム	0x003	RXFIFO(1)	CFIFO(4)
4	標準(SID)	リモートフレーム	0x000	RXFIFO(1)	CFIFO(4)
5	標準(SID)	リモートフレーム	0x001	RXFIFO(1)	CFIFO(4)
6	標準(SID)	リモートフレーム	0x002	RXFIFO(1)	CFIFO(4)
7	標準(SID)	リモートフレーム	0x003	RXFIFO(1)	CFIFO(4)
8	拡張(EID)	データフレーム	0x0000000	RXFIFO(1)	CFIFO(4)
9	拡張(EID)	データフレーム	0x0000001	RXFIFO(1)	CFIFO(4)
10	拡張(EID)	データフレーム	0x0000002	RXFIFO(1)	CFIFO(4)
11	拡張(EID)	データフレーム	0x0000003	RXFIFO(1)	CFIFO(4)
12	拡張(EID)	リモートフレーム	0x0000000	RXFIFO(1)	CFIFO(4)
13	拡張(EID)	リモートフレーム	0x0000001	RXFIFO(1)	CFIFO(4)
14	拡張(EID)	リモートフレーム	0x0000002	RXFIFO(1)	CFIFO(4)
15	拡張(EID)	リモートフレーム	0x0000003	RXFIFO(1)	CFIFO(4)

SAMPLE3 と同様の設定です。FIFO の段数は 8 段の設定（SAMPLE2~3 と同じ）です。

5.2.2. 受信ルール設定(CANFD_B, CANFD-Lite モジュール系)

・CAN-ch0

アクセプタンスフィルタ番号 page=0	フォーマット	データフレーム／リモートフレーム	ID	受信 FIFO 使用時	受信メッセージバッファ使用時番号	共通 FIFO 使用時
0	標準(SID)	データフレーム	0x000	RXFIFO(0)	割り当てなし	CFIFO(0)
1	標準(SID)	データフレーム	0x001	RXFIFO(0)	割り当てなし	CFIFO(0)
2	標準(SID)	データフレーム	0x002	RXFIFO(0)	割り当てなし	CFIFO(0)
3	標準(SID)	データフレーム	0x003	RXFIFO(0)	割り当てなし	CFIFO(0)
4	標準(SID)	リモートフレーム	0x000	RXFIFO(0)	割り当てなし	CFIFO(0)
5	標準(SID)	リモートフレーム	0x001	RXFIFO(0)	割り当てなし	CFIFO(0)
6	標準(SID)	リモートフレーム	0x002	RXFIFO(0)	割り当てなし	CFIFO(0)
7	標準(SID)	リモートフレーム	0x003	RXFIFO(0)	割り当てなし	CFIFO(0)
8	拡張(EID)	データフレーム	0x0000000	RXFIFO(0)	0	CFIFO(0)
9	拡張(EID)	データフレーム	0x0000001	RXFIFO(0)	1	CFIFO(0)
10	拡張(EID)	データフレーム	0x0000002	RXFIFO(0)	2	CFIFO(0)
11	拡張(EID)	データフレーム	0x0000003	RXFIFO(0)	3	CFIFO(0)
12	拡張(EID)	リモートフレーム	0x0000000	RXFIFO(0)	4	CFIFO(0)
13	拡張(EID)	リモートフレーム	0x0000001	RXFIFO(0)	5	CFIFO(0)
14	拡張(EID)	リモートフレーム	0x0000002	RXFIFO(0)	6	CFIFO(0)
15	拡張(EID)	リモートフレーム	0x0000003	RXFIFO(0)	7	CFIFO(0)

受信メッセージバッファを使った場合、SID は未サポートとなります。

受信 FIFO、共通 FIFO(受信として使用)、共通 FIFO(送信として使用)の段数はいずれも 4 段に設定しています。(SAMPLE2, SAMPLE3 では 8 段)

CANFD_B, CANFD-Lite では、バッファメモリの制約が厳しく、用意されているバッファメモリは 1 メッセージのデータサイズを 64 バイトとすると、16 メッセージ分です。

受信メッセージバッファを 16 個確保すると、受信メッセージバッファのみで全てのバッファメモリを使い切るので、SAMPLE4 では受信メッセージバッファを 8 個(512 バイトのメモリを消費)としています。

(SAMPLE1~3 では、CAN パケットのみを取り扱う設定で、データサイズが 8 バイトでしたので、16 個のバッファを確保しても、128 バイトのメモリ消費量に納まります。)

・CANFD モードでのメッセージ数設定(1 メッセージのデータサイズ 64 バイト)

	受信 FIFO 使用時	受信メッセージバッファ使用時	受信に共通 FIFO 使用時
受信メッセージバッファ	8	8	8
RXFIFO0	4	0	0
RXFIFO1	0	0	0
CFIFO	4(送信で使用)	4(送信で使用)	4
合計	16	12	12

※合計が 16 以下である必要があります(データサイズ 64 バイトの場合)
※赤字の受信メッセージバッファは未使用なので 0 に設定しても問題ありません

FIFO 段数の設定と受信メッセージバッファ数の設定を調整して、容量をオーバしないようにしてください。

※CANFD フレームを取り扱う場合でも、データサイズを 64 バイトではなくもっと小さな値に制限した場合、16 を超えるメッセージ（例えば 32 バイトにした場合は 32 メッセージまで）を取り扱う事も可能です（その場合、データサイズの設定を超えるパケットは受信できません）

CANFD モジュールでは、メッセージ数はデータサイズ 64 バイトの場合でも 512 メッセージとなりますので、FIFO 段数の設定や受信メッセージバッファ数の設定に余裕がありますが、CANFD_B と CANFD-Lite モジュールでは、意外に少ない容量ですので、注意が必要です。

（※メッセージ数の設定をオーバした場合、受信データで AFL テーブルが上書きされる等の、意図しないデータ破壊が起きます。メッセージ数の設定オーバに関しては、マイコン側にチェックする手段がないため、プログラムを作成する人が計算する必要があります。）

5.3. 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[SAMPLE3 との相違点]

受信 FIFO 使用時、共通 FIFO を受信に使用する場合、受信ルール番号 0~3, 8~11 をデータフレーム受信、受信ルール番号 4~7, 12~15 をリモートフレーム受信向けに設定。(SAMPLE3 と同じ)

受信メッセージバッファを受信に使用する場合、受信ルール番号 0~3 をデータフレーム受信、受信ルール番号 4~7 をリモートフレーム受信向けに設定。拡張 ID のみ受信設定を行い、標準 ID のデータは受信設定を行わない。

(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　**FDF(=1)**　**BRS(=1)**　データ
s_rtr = CAN_DATA_FLAME; //(=0)
s_fdf = 1;
s_brs = 1;
s_ret = can**fd*n***_cfifo_send(s_format, s_rtr, s_id, s_dlc, **s_fdf**, **s_brs**, &s_data[0]);

データ送信時、FDF と BRS の項目が追加されています。FDF=1 の時は CANFD フォーマットであることを示しており、BRS=1 の時は、データフィールドを高速で通信する事を示しています。(取扱説明書の 2.5 章　CANFD のデータパケットの項を参照)

[CANFD_B, CANFD-Lite モジュール]受信に共通 FIFO を使った場合は、送信は canfd0_buf_send()での送信となります(送信割り込みは使用できない)

s_ret = can**fd**0_buf_send(0, s_format, s_rtr, s_id, s_dlc, **s_fdf**, **s_brs**, &s_data[0]); //受信に共通 FIFO を使用した場合

送信バッファで送信する関数も、CANFD 版(can**fd**0_buf_send)となり、FDF と BRS の項が従来の関数に対して増えています。

※BRS=1 を指定しているので、データフィールドの通信速度は、起動に設定した速度(デフォルト 2Mbps)となります

リモートフレーム要求の場合は、

//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　**FDF(=0)**　**BRS(=0)**　データ
s_rtr = CAN_REMOTE_FLAME; //(=1)
s_fdf = 0;
s_brs = 0;
s_ret = can**fd*n***_cfifo_send(s_format, s_rtr, s_id, s_dlc, **s_fdf**, **s_brs**, &s_data[0]);

となります。FDF と BRS は 0 となり、SAMPLE3 でのリモートフレーム要求のパケットと同じになります。
（但し、DLC の値が 9 以上を出す事がある（e,r コマンド）点が SAMPLE3 とは異なります）

※リモートフレーム送信では、通信速度は通信開始時の速度のまま（デフォルト設定では 1Mbps）です

(i1)受信割り込み関数[SAMPLE3 との相違点]

FIFO が空になるまでループ(SAMPLE2~3 同様）
{
　ret = **canfd*n*_rxfifo_receive**(&ide, &rtr, &id, &fdf, &brs, &esi, &data[0], &ts);　　//受信 FIFO を使った受信
　if(rtr == CAN_DATA_FRAME)
　{
　　[データフレームの場合：受信データの表示]
　}
　else if(rtr == CAN_REMOTE_FRAME)
　{
　　[リモートフレームの場合：データ表示，データ送信]
　　dlc = ret;　//送信バイト数は受信した DLC 値とする
　　ret2 = **canfd*n*_cfifo_send**(ide, CAN_DATA_FRAME, id, ,dlc, CANFD_DATA_FORMAT, CANFD_BITRATE, &remote_frame_response_data[0]);
　}
}

受信関数が、CANFD 版となり、&fdf, &brs, &esi の引数が追加されています。

受信したデータがリモートフレームの場合のデータ送信関数も CANFD 版となり、
FDF= CANFD_DATA_FORMAT(1), BRS= CANFD_BITRATE(1)を指定して送信する事としています（返信するデータは CANFD フレーム）。

上記は、受信に受信 FIFO を使った場合ですが、共通 FIFO、受信メッセージバッファを使った場合でも同様の処理となります。

5.4. SAMPLE4 フローチャート

データ送信、受信の関数が CANFD 版に変わっているだけで、フローチャートとしては SAMPLE3 に同じ。

6. サンプルプログラムで使用している関数の説明

6.1. 関数仕様

can_reset
can*n*_init　　(n=0~1)

概要:初期化関数
宣言:
　　int can_reset(void)
　　int can0_init(void)　[ch0 向け]
　　int can1_init(void)　[ch1 向け] [CANFD モジュールのみ]

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_reset()の実行後、can*n*_init()を実行してください。

can_reset()は全体の処理、can*n*_init()は ch 毎の処理です。

receive_buf_conf

概要:AFL(アクセプタンスフィルタリスト), 受信メッセージバッファ数設定関数
宣言:
　　int receive_buf_conf(void)

説明:
　・AFL ロック解除
　・受信メッセージバッファ数設定
を行います

can*n*_init() 後、can*n*_receive_rule_set()実行前に実行してください。

引数:
　　なし
戻り値:
　　0: 正常終了

can*n*_receive_rule_set　　(n=0~1)

概要:受信ルール設定関数
宣言:
　　int can0_receive_rule_set(unsigned char num, unsigned char mode, unsigned char ide, unsigned char rtr, unsigned long id);
　　int can1_receive_rule_set(unsigned char num, unsigned char mode, unsigned char ide, unsigned char rtr, unsigned long id); [CANFD モジュールのみ]

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)(*1) [CANFD モジュール]
　　num: 受信ルール番号(0~31) [CANFD_B, CANFD-Lite モジュール]
　　mode: 受信先
　　CAN_RULE_BUF(0x0)　　受信バッファで受信
　　CAN_RULE_RXFIFO0(0x0001)　　受信 FIFO(0)で受信
　　CAN_RULE_RXFIFO1(0x0002)　　受信 FIFO(0)で受信
　　CAN_RULE_RXFIFO2(0x0003)　　受信 FIFO(0)で受信 [CANFD モジュールのみ]
…
　　CAN_RULE_RXFIFO7(0x0080)　　受信 FIFO(7)で受信 [CANFD モジュールのみ]

CAN_RULE_CFIFO0(0x0100)　共通 FIFO(0)で受信
CAN_RULE_CFIFO1(0x0200)　共通 FIFO(1)で受信 [CANFD モジュールのみ]
…
CAN_RULE_CFIFO5(0x2000)　共通 FIFO(5)で受信 [CANFD モジュールのみ]
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

補足:

CANFD_B, CANFD-Lite モジュールでは、受信 FIFO は 2 本、共通 FIFO は 1 本となります
(*1) [CANFD モジュール]ch 毎に 16 個ずつ割り振る事としたので、本関数では 0-15 までとしています。
※プログラムを変更すれば 128 個/ch まで拡張可能です

can_operate
can*n*_operate　(n=0~1)

概要:動作モード変更関数

宣言:

int can_operate(void)
int can0_operate(void)
int can1_operate(void) [CANFD モジュールのみ]

説明:

・CANFD 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

can_operate 実行（CANFD モジュール全体を動作モードに変更）後 ch 毎の動作モードを変更(can*n*_operate)を実行してください。

cann_buf_send　　(n=0~1)
canfdn_buf_send　　(n=0~1)

概要:データ送信関数
宣言:
　　int can0_buf_send(unsigned char num, unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char *data);
　　int can1_buf_send(unsigned char num, unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char *data);　[CANFD モジュールのみ]
　　int canfd0_buf_send(unsigned char num, unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char fdf, unsigned char brs, unsigned char *data);
　　int canfd1_buf_send(unsigned char num, unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char fdf, unsigned char brs, unsigned char *data);　[CANFD モジュールのみ]

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(0~7)(*1)　[CANFD モジュール]
　　num: 送信バッファ番号(0~3)(*2)　[CANFD_B, CANFD-Lite モジュール]
　　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):1~8 バイト　[cann_buf_send]
　　dlc: 送信バイト数を指定します(1~8):1~8 バイト、(9~15):12~64 バイト　[canfdn_buf_send]
　　fdf: CANFD フレーム区分　[canfdn_buf_send のみ]
　　　CAN_DATA_FORMAT(0)　　CAN フレームのフォーマットで送信
　　　CANFD_DATA_DORMAT(1)　CANFD フレームのフォーマットで送信
　　brs: ビットレートスイッチ　[canfdn_buf_send のみ]
　　　CAN_BITRATE(0)　　データフィールドの通信速度は通常
　　　CANFD_BITRATE(1)　データフィールドの通信速度は高速
　　*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

補足:

(*1)[CANFD モジュール]引数として指定する送信バッファ番号は 0~7 です。CAN-ch0 の場合は、送信メッセージバッファ 0~7 が使用されます。CAN-ch1 の場合は、送信メッセージバッファ 64~71（num+64)が使われます。

(*2)[CANFD_B, CANFD-Lite モジュール]引数として指定する送信バッファ番号は 0~3 です。

cann_cfifo_send (n=0~1)

canfdn_cfifo_send (n=0~1)

概要:データ送信関数

宣言:

int can0_cfifo_send(unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char *data)

int can1_cfifo_send(unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char *data) [CANFD モジュールのみ]

int canfd0_cfifo_send(unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char fdf, unsigned char brs, unsigned char *data)

int canfd1_cfifo_send(unsigned char ide, unsigned char rtr, unsigned long id, unsigned char dlc, unsigned char fdf, unsigned char brs, unsigned char *data) [CANFD モジュールのみ]

説明:

・共通 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):1~8 バイト [cann_cfifo_send]

dlc: 送信バイト数を指定します(1~8):1~8 バイト、(9~15):12~64 バイト [canfdn_cfifo_send]

fdf: CANFD フレーム区分 [canfdn_cfifo_send のみ]

CAN_DATA_FORMAT(0) CAN フレームのフォーマットで送信

CANFD_DATA_DORMAT(1) CANFD フレームのフォーマットで送信

brs: ビットレートスイッチ [canfdn_cfifo_send のみ]

CAN_BITRATE(0) データフィールドの通信速度は通常

CANFD_BITRATE(1) データフィールドの通信速度は高速

*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

補足:

[CANFD モジュール] CAN ch0 は CFIFO(0)で送信を行います

[CANFD モジュール] CAN ch1 は CFIFO(3)で送信を行います

[CANFD_B, CANFD-Lite モジュール] CFIFO(0)で送信を行います

cann_buf_receive (n=0~1)
canfdn_buf_receive (n=0~1)

概要:受信関数

宣言:

int can0_buf_receive(unsigned char num, unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts)

int can1_buf_receive(unsigned char num, unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts) [CANFD モジュール]

int canfd0_buf_receive(unsigned char num, unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *fdf, unsigned char *brs, unsigned char *esi, unsigned char *data, unsigned short *ts)

int canfd1_buf_receive(unsigned char num, unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *fdf, unsigned char *brs, unsigned char *esi, unsigned char *data, unsigned short *ts) [CANFD モジュール]

説明:

・受信メッセージ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(0~15)(*1) [CANFD モジュール]

num: 受信ルール番号(0~31)(*2) [CANFD_B, CANFD-Lite モジュール]

*ide: 標準／拡張フォーマット区分を格納するポインタ(unsigned char [1])

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 [1])

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 [1])

*fdf: 受信した FDF を格納するポインタ(unsigned char [1])

CAN_DATA_FORMAT(0)　　CAN フレームのフォーマット

CANFD_DATA_DORMAT(1)　CANFD フレームのフォーマット

*brs: 受信した BRS を格納するポインタ(unsigned char [1])

CAN_BITRATE(0)　　データフィールドの通信速度は通常

CANFD_BITRATE(1)　データフィールドの通信速度は高速

*esi: 受信した ESI を格納するポインタ(unsigned char [1])

CANFD_ERROR_ACTIVE(0)　　エラーアクティブ

CANFD_ERROR_PASSIVE(1)　エラーアクティブパッシブ

*data: 受信したデータを格納するポインタ（最大 unsigned char [8]） [cann_buf_receive]

*data: 受信したデータを格納するポインタ（最大 unsigned char [64]） [canfdn_buf_receive]

*ts: データ受信時のタイムスタンプ(unsigned short [1])

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　[cann_buf_receive]

1~15: 受信したデータのバイト数に相当する DLC 値　[canfdn_buf_receive]

-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

補足:

(*1) [CANFD モジュール]引数として指定する受信ルール番号は 0~15 です。CAN-ch0 の場合は、受信メッセージバッファ 0~15 が使用されます。CAN-ch1 の場合は、受信メッセージバッファ 16~31（num+16)が使われます。（受信メッセージバッファは全体で 32 個です、CANFD モジュールでは CAN-ch0 で 16 個、CAN-ch1 で 16 個定義しているので can0_buf_receive()/canfd0_buf_receive()の num=0~15 が受信メッセージバッファ 0~15 に対応し、can1_buf_receive()/canfd1_buf_receive()の num=0~15 が受信メッセージバッファ 16~31 に対応します。）

(*2) [CANFD_B, CANFD-Lite モジュール]受信ルール番号 0~31 を指定できます。受信ルール番号=受信メッセージバッファの対応としています。

※CANFD パケットを取り扱う際（かつ、メッセージサイズを 64 バイトとした場合は）は、受信メッセージバッファは最大 16 バッファです（FIFO バッファを使用しない場合）

can*n*_rxfifo_receive
canfd*n*_rxfifo_receive

概要:受信 FIFO 受信関数

宣言:

int can0_rxfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts)

int can1_rxfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts) [CANFD モジュールのみ]

int canfd0_rxfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *fdf, unsigned char *brs, unsigned char *esi, unsigned char *data, unsigned short *ts)

int canfd1_rxfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *fdf, unsigned char *brs, unsigned char *esi, unsigned char *data, unsigned short *ts) [CANFD モジュールのみ]

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 [1])

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 [1])

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 [1])

*fdf: 受信した FDF を格納するポインタ(unsigned char [1])

CAN_DATA_FORMAT(0) CAN フレームのフォーマット

CANFD_DATA_DORMAT(1) CANFD フレームのフォーマット

*brs: 受信した BRS を格納するポインタ(unsigned char [1])

CAN_BITRATE(0) データフィールドの通信速度は通常

CANFD_BITRATE(1) データフィールドの通信速度は高速

*esi: 受信した ESI を格納するポインタ(unsigned char [1])

CANFD_ERROR_ACTIVE(0) エラーアクティブ

CANFD_ERROR_PASSIVE(1) エラーアクティブパッシブ

*data: 受信したデータを格納するポインタ（最大 unsigned char [8]） [cann_rxfifo_receive]

*data: 受信したデータを格納するポインタ（最大 unsigned char [64]） [canfdn_rxfifo_receive]

*ts: データ受信時のタイムスタンプ(unsigned short [1])

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 [cann_rxfifo_receive]

1~15: 受信したデータのバイト数に相当する DLC 値 [canfdn_rxfifo_receive]

-1: 引数チェックエラー

-2: 受信 FIFO にデータなし

補足:

CAN ch0 は RXFIFO(0)で送信を行います

[CANFD モジュール]CAN ch1 は RXFIFO(1)で送信を行います

(canfd¥canfd.h/canfd_b¥canfd_b.h/canfd_lite¥canfd_lite.h 内で使用 FIFO 番号を定義)

cann_cfifo_receive
canfdn_cfifo_receive

概要:共通 FIFO 受信関数

宣言:

int can0_cfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts)

int can1_cfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *data, unsigned short *ts) [CANFD モジュールのみ]

int canfd0_cfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *fdf, unsigned char *brs, unsigned char *esi, unsigned char *data, unsigned short *ts)

int canfd1_cfifo_receive(unsigned char *ide, unsigned char *rtr, unsigned long *id, unsigned char *fdf, unsigned char *brs, unsigned char *esi, unsigned char *data, unsigned short *ts) [CANFD モジュールのみ]

説明:

・共通 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 [1])

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 [1])

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 [1])

*fdf: 受信した FDF を格納するポインタ(unsigned char [1])

CAN_DATA_FORMAT(0) CAN フレームのフォーマット

CANFD_DATA_DORMAT(1) CANFD フレームのフォーマット

*brs: 受信した BRS を格納するポインタ(unsigned char [1])

CAN_BITRATE(0) データフィールドの通信速度は通常

CANFD_BITRATE(1) データフィールドの通信速度は高速

*esi: 受信した ESI を格納するポインタ(unsigned char [1])

CANFD_ERROR_ACTIVE(0)　　エラーアクティブ

CANFD_ERROR_PASSIVE(1)　　エラーアクティブパッシブ

*data: 受信したデータを格納するポインタ（最大 unsigned char [8]）　[cann_cfifo_receive]

*data: 受信したデータを格納するポインタ（最大 unsigned char [64]）[canfdn_cfifo_receive]

*ts: データ受信時のタイムスタンプ(unsigned short [1])

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　[cann_rxfifo_receive]

1~15: 受信したデータのバイト数に相当する DLC 値　[canfdn_rxfifo_receive]

-1: 引数チェックエラー

-2: 受信 FIFO にデータなし

補足:

[CANFD モジュール]CAN ch0 は CFIFO(1)で受信を行います、CAN ch1 は CFIFO(4)で受信を行います

[CANFD_B, CANFD-Lite モジュール]CFIFO(0)で受信を行います

(canfd¥canfd.h/canfd_b¥canfd_b.h/canfd_lite¥canfd_lite.h 内で使用 FIFO 番号を定義）

6.2. プログラムで使用している変数・定数

6.2.1. グローバル変数

unsigned long can*n*_remote_frame_request; (n=[], 0~1)

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。

6.2.2. 定数定義

canfd¥canfd.h [CANFD モジュール]
canfd_b¥canfd_b.h [CANFD_B モジュール]
canfd_lite¥canfd_lite.h [CANFD-Lite モジュール]
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 CANFD_BPS_12M   12000        //12M bps  ※テスト用(*1)(*2)
#define CANFD_BPS_10M   10000        //10M bps  ※テスト用(*1)(*2)
#define CANFD_BPS_8M   8000          //8M bps  ※テスト用(*1)(*3)
#define CANFD_BPS_7_5M   7500        //7.5M bps  ※テスト用(*1)(*2)
#define CANFD_BPS_6_6M   6667        //6.6M bps  ※テスト用(*1)
#define CANFD_BPS_6M   6000          //6M bps  ※テスト用(*1)(*2)
#define CANFD_BPS_5M   5000          //5M bps
#define CANFD_BPS_4M   4000          //4M bps
#define CANFD_BPS_3_3M   3333        //3.3M bps
#define CANFD_BPS_3M   3000          //3M bps (*2)
#define CANFD_BPS_2M   2000          //2M bps
```

CANFD のデータ部分速度設定。
(*1)テスト用に定義されています
(*2)CANFDCLK=60MHz の場合に定義されています
(*3) CANFDCLK=40MHz の場合に定義されています

```
#define ENABLE   1
#define DISABLE   0
```

パラメータの有効/無効設定定数。

```
#define DELAY_OFFSET   0          //測定値＋オフセット値
#define OFFSET_ONLY   1           //オフセット値のみ
```

CANFDの第2サンプルポイントの設定。

```
#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。(IDE)

```
#define CAN_DATA_FRAME   0             //データフレーム
#define CAN_REMOTE_FRAME   1           //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。(RTR)

```
#define CAN_DATA_FORMAT   0            //CANデータフォーマット
#define CANFD_DATA_FORMAT   1          //CANFDデータフォーマット
```

CANFDデータフォーマット区分。(FDF)

```
#define CAN_BITRATE   0            //CANビットレート
#define CANFD_BITRATE   1          //CANFDビットレート
```

CANFDデータビットレート。(BRS)

```
#define CANFD_ERROR_ACTIVE   0          //エラーアクティブ
#define CANFD_ERROR_PASSIVE   1         //エラーパッシブ
```

CANFD のエラー区分。(ESI)

```
#define  CAN_RULE_BUF  0x0        //受信メッセージバッファモード
#define  CAN_RULE_RXFIFO0  0x0001        //受信 FIFO0 で受信
#define  CAN_RULE_RXFIFO1  0x0002        //受信 FIFO1 で受信
#define  CAN_RULE_RXFIFO2  0x0004        //受信 FIFO2 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_RXFIFO3  0x0008        //受信 FIFO3 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_RXFIFO4  0x0010        //受信 FIFO4 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_RXFIFO5  0x0020        //受信 FIFO5 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_RXFIFO6  0x0040        //受信 FIFO6 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_RXFIFO7  0x0080        //受信 FIFO7 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_CFIFO0  0x0100        //共通 FIFO0 で受信
#define  CAN_RULE_CFIFO1  0x0200        //共通 FIFO1 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_CFIFO2  0x0400        //共通 FIFO2 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_CFIFO3  0x0800        //共通 FIFO3 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_CFIFO4  0x1000        //共通 FIFO4 で受信 [CANFD モジュールのみ]
#define  CAN_RULE_CFIFO5  0x2000        //共通 FIFO5 で受信 [CANFD モジュールのみ]
```

受信データ格納先定義。

```
#define CAN_RX_METHOD  RXFIFO        //受信FIFOで受信
#define CAN_RX_METHOD  RXBUF        //受信メッセージバッファで受信 [CANFD_B, CANFD-Liteモジュールのみ]
#define CAN_RX_METHOD  CFIFO        //共通 FIFO で受信
```

データ受信方法の定義（いずれかの定義を有効化する）。

[CANFD モジュール]

```
#define CAN0_RX_RXFIFO_NO  0        //CAN-ch0受信に使用する受信FIFO番号
#define CAN0_RX_CFIFO_NO  1        //CAN-ch0受信に使用する共通FIFO番号
#define CAN0_TX_CFIFO_NO  0        //CAN-ch0送信に使用する共通FIFO番号

#define CAN1_RX_RXFIFO_NO  1        //CAN-ch1受信に使用するRXFIFO番号
#define CAN1_RX_CFIFO_NO  4        //CAN-ch1受信に使用する共通FIFO番号
#define CAN1_TX_CFIFO_NO  3        //CAN-ch1送信に使用する共通FIFO番号
```

[CANFD_B, CANFD-Liteモジュール]

```
#define CAN0_RX_RXFIFO_NO  0        //受信に使用する受信FIFO番号
#define CAN0_RX_CFIFO_NO  0        //受信に使用する共通FIFO番号
#define CAN0_TX_CFIFO_NO  0        //送信に使用する共通FIFO番号
```

使用する FIFO 番号の定義。[CANFD モジュール]では、RXFIFO 0~7、CFIFO 0~5、[CANFD_B, CANFD-Lite モジュール]では、RXFIFO 0~1、CFIFO 0 が使用可能。

#define CAN_INTERRUPT_DEBUG

定義時、デバッグ向けに、割り込み時にポートを反転させる。

6.3. 受信ルール設定に関して

```
                                                (1)                 (2)                 (3)
can0_receive_rule_set(0, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000000);
can0_receive_rule_set(1, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000001);
can0_receive_rule_set(2, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000002);
can0_receive_rule_set(3, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_DATA_FRAME, 0x00000003);
can0_receive_rule_set(4, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_REMOTE_FRAME, 0x00000000);
can0_receive_rule_set(5, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_REMOTE_FRAME, 0x00000001);
can0_receive_rule_set(6, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_REMOTE_FRAME, 0x00000002);
can0_receive_rule_set(7, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_SID, CAN_REMOTE_FRAME, 0x00000003);
can0_receive_rule_set(8, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000000);
can0_receive_rule_set(9, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000001);
can0_receive_rule_set(10, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000002);
can0_receive_rule_set(11, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_DATA_FRAME, 0x00000003);
can0_receive_rule_set(12, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_REMOTE_FRAME, 0x00000000);
can0_receive_rule_set(13, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_REMOTE_FRAME, 0x00000001);
can0_receive_rule_set(14, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_REMOTE_FRAME, 0x00000002);
can0_receive_rule_set(15, CAN_RULE_RXFIFO0, CAN_ID_FORMAT_EID, CAN_REMOTE_FRAME, 0x00000003);
```

本サンプルプログラムでは、can*n*_receive_rule_set() 関数で

(1)ID 区分　CANID_FORMAT_SID(0)か CANID_FORMAT_EID(1)
(2)データ／リモートフレーム区分　CAN_DATA_FRAME(0)か CAN_REMOTE_FRAME(1)
(3)ID 値

を設定し、(1)~(3)の全てが合致した場合のみ、受信するようになっています。受信ルールは 32 個[CANFD_B, CANFD-Lite モジュール], 128 個/ch[CANFD モジュール]という上限がありますので、「どのような ID 値であっても受信したい」とか、1 つのルールで SID（標準 ID）と EID（拡張 ID）の両方を受信したいという事があるかと思います。

その様な場合は、can*n*_receive_rule_set() 関数に手を加えてください。この関数内のマスクレジスタが(1)~(3)を比較対象とするかどうかを決めています。can*n*_receive_rule_set() 関数ではマスクビットを 0b1 に設定しているので、一致した場合のみルールにマッチするようになっています。

（サンプルプログラムでは ID 決め打ちで受信を行っていますが、受信ルールの設定次第で変えることが可能です。）

また、本サンプルプログラムでは DLC 値に関しては受信制限を設けていない（DLC がどの値であっても受信する）としていますが、受信する DLC 値に関して制約を掛ける事（DLC=8 以上のメッセージしか受信しない）も可能です。

受信ルールにマッチしたデータがデータ格納先（第 2 引数、上記では RXFIFO の 0 番）に格納され、複数のルールがマッチする場合は、番号の若い方のルールが優先されます。

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	2022.8.19	—	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	P13-15	RA の割り込み番号の変更、特定のマイコン型名からモジュール名への変更 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 スターターキット RX/RA
CAN スターターキット SmartRX
CANFD ソフトウェア編　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)