

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ひ申し上げます。

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のですが、誤りが無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、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と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。

本説明書はV850ES/Kx1, V850ES/Kx1+用のシリアル通信系のサンプルソフトウェアの動作を説明したものです。

ご注意

本ソフトウェアはあくまで参考用のソフトであり、当社がこの動作を保証するものではありません。

本ソフトウェアを使用する場合、お客様のセット上で十分な評価の上ご使用いただきますようお願いいたします。

目次

シリアル・インターフェース	UART 送信 (UARTn)	P 4
シリアル・インターフェース	UART 受信 (UARTn)	P 7
シリアル・インターフェース	UART 送受信 (UARTn)	P10
シリアル・インターフェース	CSI 送信 (CSIn)	P13
シリアル・インターフェース	CSI 受信 (CSIn)	P16
シリアル・インターフェース	CSI 送受信 (CSIn)	P19
シリアル・インターフェース	CSI 繰り返しモード送信 (CSIn)	P23
シリアル・インターフェース	CSI 繰り返しモード受信 (CSIn)	P27
シリアル・インターフェース	CSI 繰り返しモード送受信 (CSIn)	P31
シリアル・インターフェース	CSIA 単発送信 (CSIA _n)	P35
シリアル・インターフェース	CSIA 単発受信 (CSIA _n)	P38
シリアル・インターフェース	CSIA 単発送受信 (CSIA _n)	P41
シリアル・インターフェース	CSIA 自動モード送信 (CSIA _n)	P45
シリアル・インターフェース	CSIA 自動モード受信 (CSIA _n)	P48
シリアル・インターフェース	CSIA 自動モード送受信 (CSIA _n)	P51

関数一覧表は以下のように構成されています。

テーマ (ハードウェア略号)

【機能】	テーマの説明
【関数名】	サンプル関数の名前
【引数】	引数の型と概要
【処理内容】	サンプル関数の処理内容
【起動方法】	関数の呼び出し条件
【使用SFR】	レジスタ名と設定内容
【call関数】	呼び出し関数の名前と機能
【変数】	サンプル関数での使用変数の型、名前、概要
【割り込み】	関数名
【割り込み要因】	名称
【使用ハード】	その他使用リソース
【ファイル名】	対応するサンプルプログラム・ファイル名
【注意事項】	関数使用上の注意。使い方

割り込み関数

【関数名】	
【概要】	処理の目的
【要因】	指定無し
【call関数】	無し
【変数】	変数名 機能
【ファイル名】	対応するサンプルプログラム・ファイル名
【注意事項】	無し

シリアル・インターフェース UART 送信 (UARTn) KF1 / KG1 : n = 0-1

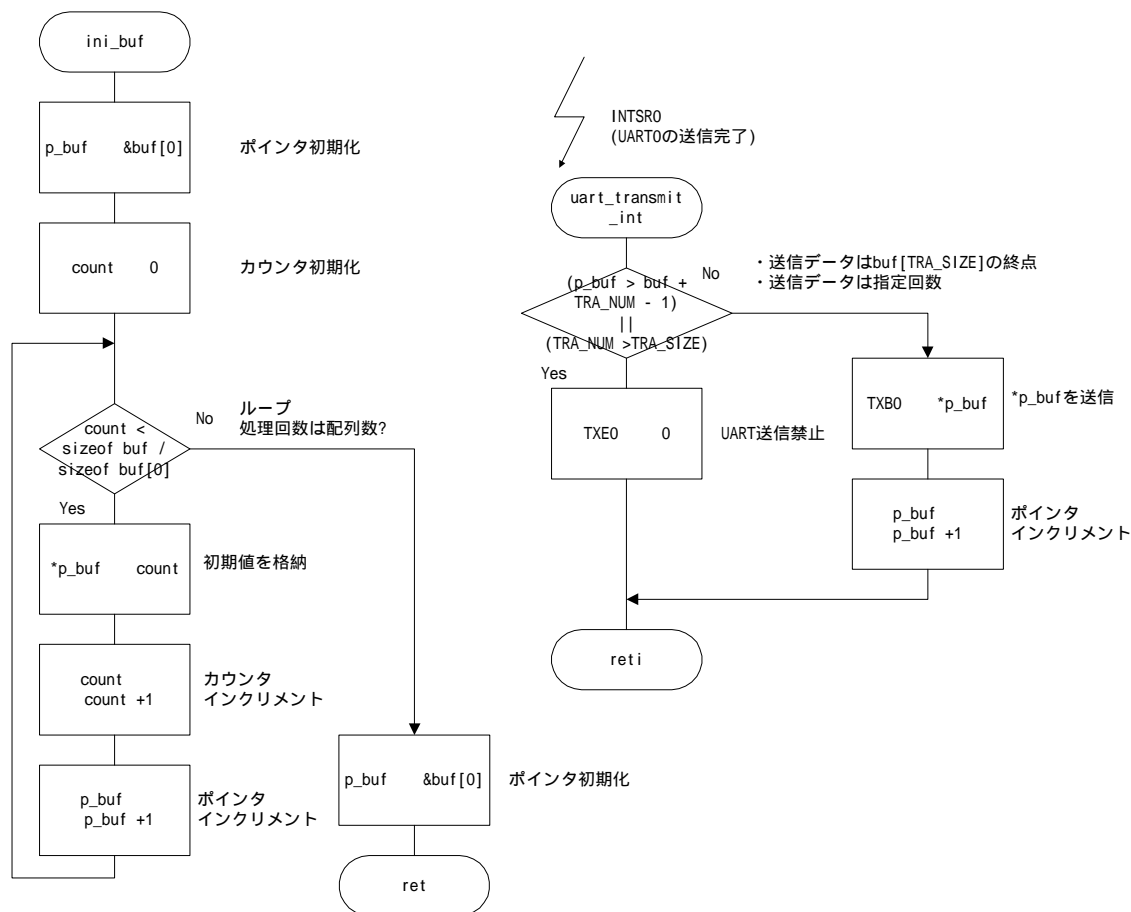
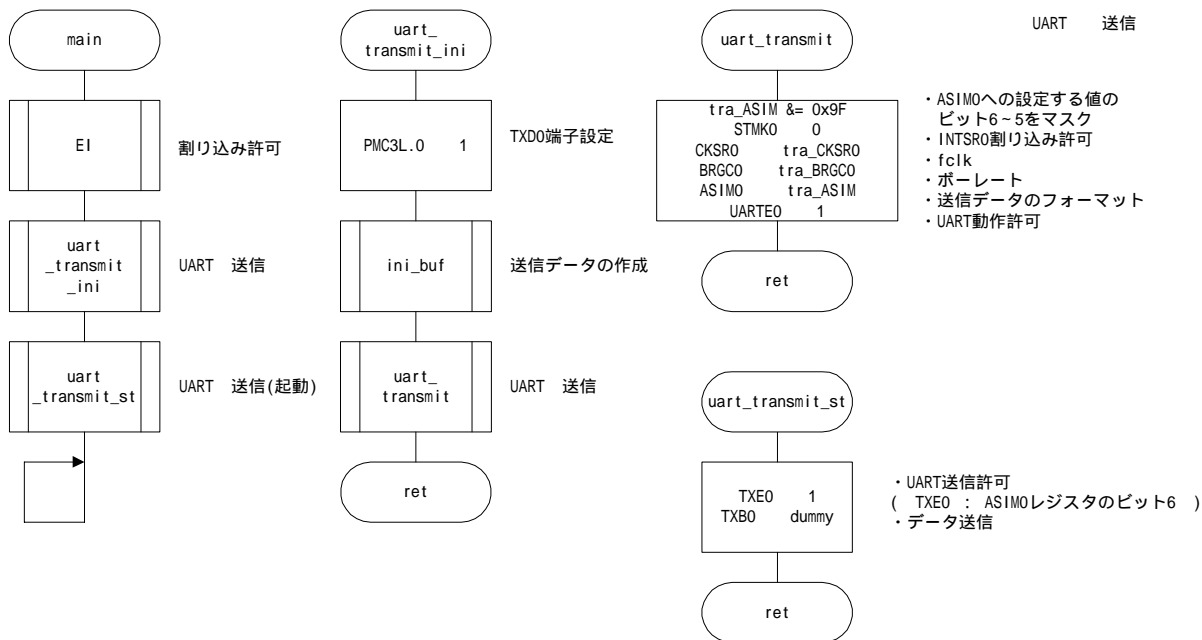
KJ1 : n = 0-2

【機能】	送信準備(URTE0 = 1、TEX0 = 1)の設定を行ったあとで送信バッファ0レジスタに(TXB0)に送信データを書きこむことで、データを出力します。	
【関数名】	uart_transmit	
【引数】	unsigned char tra_CKSRO	クロックの分周
	unsigned char tra_BRGCO	ボーレートの設定
	unsigned char tra_ASIM	送信データのフォーマット
【処理内容】	UART送信動作の設定を行います。設定は以下の通りです。 ボーレート：9600bps パリティ・ビット：なし データ長：8ビット ストップ・ビット：1ビット 送信データ：0xA5、0~9(10進) 転送方向：LSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールしてください。 ・コールする前にTXD0端子の設定を行ってください。 ・メインクロックを16MHzに設定してください。 ・uart_transmit_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	STIC0	INTST0割り込みのマスクとレベルの設定
	CKSRO	基本クロックの選択
	BRGCO	ボーレートを制御
	ASIMO	UART0の転送動作を制御
【call関数】	main	メイン関数
【変数】	無し	
【割り込み】	uart_transmit_int	
【割り込み要因】	INTST0	
【ファイル名】	uart_transmit¥UART_1.c , uart_transmit¥MAIN.C	
【注意事項】	<ul style="list-style-type: none"> ・UARTnを使用する場合は、必ずUARTnに関連する外部端子をコントロール・モードに設定した後で、UARTnの設定を行ってください。 ・TXD端子と送信データの初期化はuart_transmit_ini関数で行っています。 	

【関 数 名】	uart_transmit_st
【引 数】	unsigned char dummy 初回送信データ
【処 理 内 容】	uart_transmitの起動関数です。
【起 動 方 法】	uart_transmit関数の後にコールしてください。
【使 用 S F R】	TXE0 送信許可 TXB0 送信バッファ
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	uart_transmit¥UART_1.c
【注 意 事 項】	無し

割り込み関数

【関 数 名】	uart_transmit_int
【概 要】	・ポインタ操作により、buf[TRA_SIZE]のデータを1バイトずつ送信します。 ・ポインタのインクリメントで、次の送信データ格納先を指定します。
【要 因】	INTST0 UART0送信完了
【 call 関数】	無し
【変 数】	unsigned char buf[TRA_SIZE] 送信データを格納 unsigned char *p_buf 送信データのアドレスを格納
【フ ァ イ ル 名】	uart_transmit¥UART_1.c
【注 意 事 項】	無し



シリアル・インターフェース UART 受信 (UARTn) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

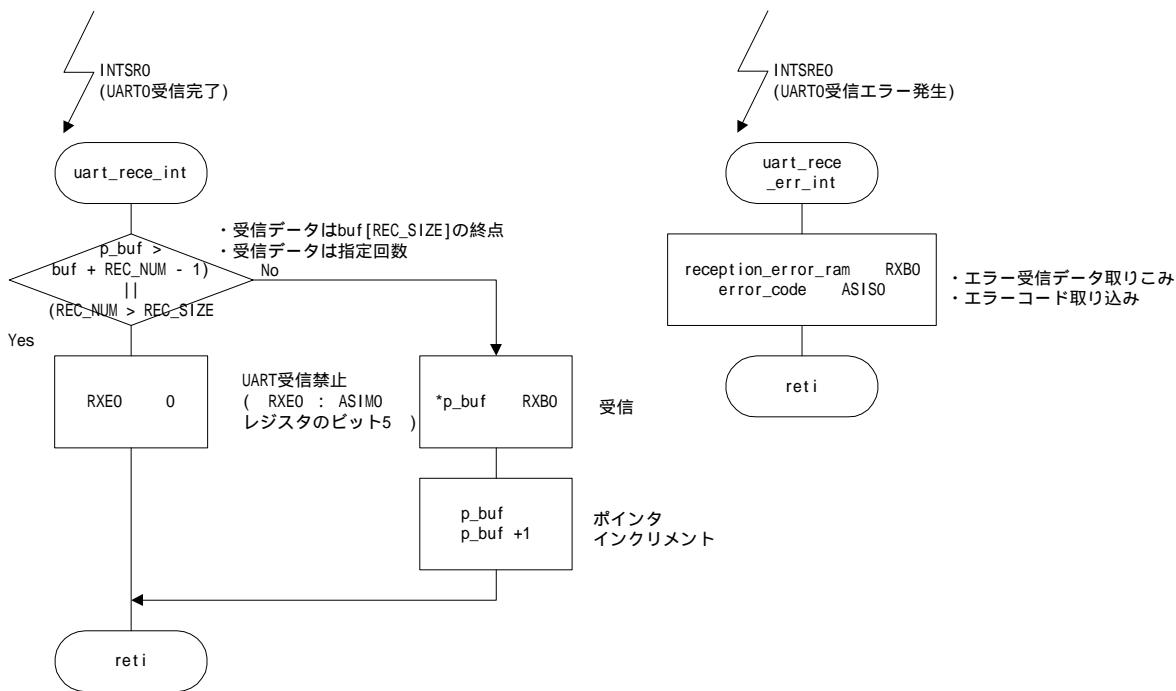
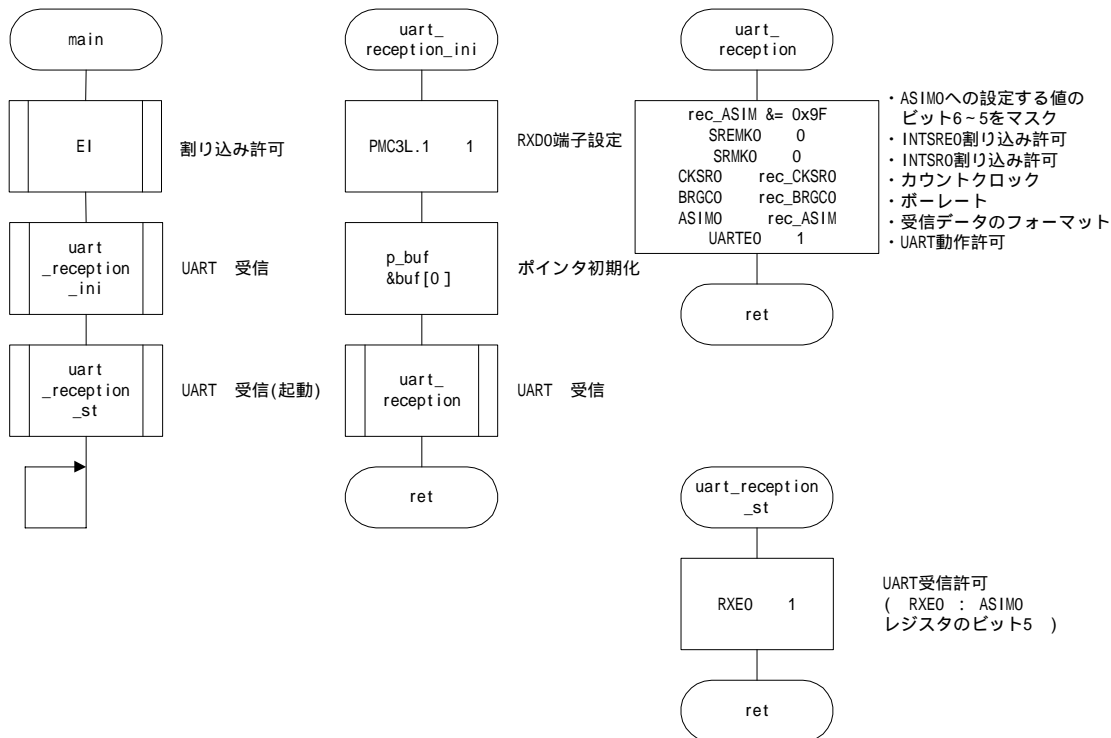
【機能】	受信準備 (URTE0 = 1、RXE0 = 1) の設定を行うことで、受信待ちの状態になります。この状態でRXE0端子から、スタートビットの検出により設定されたボーレートでデータを受信します。	
【関数名】	uart_reception	
【引数】	unsigned char rec_CKSRO	クロックの分周
	unsigned char rec_BRGCO	ボーレートの設定
	unsigned char rec_ASIM	受信データのフォーマット
【処理内容】	UART受信動作の設定を行います。設定は以下の通りです。 ボーレート： 9600bps パリティ・ビット：なし データ長：8ビット ストップ・ビット：1ビット 転送方向：LSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールしてください。 ・コールする前にRXD0端子の設定を行ってください。 ・メインクロックを16MHzに設定してください。 ・uart_reception_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	SREICO	INTSRE0割り込みのマスクとレベルの設定
	SRICO	INTSRO割り込みのマスクとレベルの設定
	CKSRO	基本クロックの選択
	BRGCO	ボー・レートを制御
	ASIMO	UART0の転送動作を制御
【call関数】	main	メイン関数
【変数】	無し	
【割り込み】	uart_rece_int	
	uart_rece_err_int	
【割り込み要因】	INTSRO	
	INTSRE0	
【ファイル名】	uart_reception¥UART_2.c , uart_reception¥MAIN.C	
【注意事項】	<ul style="list-style-type: none"> ・UARTnを使用する場合は、必ずUARTnに関連する外部端子をコントロール・モードに設定した後で、UARTnの設定を行ってください。 RXE端子と受信データ格納先の初期化はuart_reception_ini関数で行っています。 	

【関 数 名】	uart_reception_st
【引 数】	無し
【処 理 内 容】	uart_receptionの起動関数です。
【起 動 方 法】	uart_reception関数の後にコールしてください。
【使 用 S F R】	RXE0 受信許可
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	uart_reception¥UART_2.c
【注 意 事 項】	無し

割り込み関数

【関 数 名】	uart_rece_int
【概 要】	<ul style="list-style-type: none"> ・ポインタ操作により、受信データをbuf[REC_SIZE]に格納します。 ・ポインタインクリメントで次の受信データ格納先を指定します。
【要 因】	INTSR0 UART0受信完了
【 call 関数】	無し
【変 数】	unsigned char buf[10] 受信データを格納 unsigned char *p_buf 受信データのアドレスを格納
【フ ァ イ ル 名】	uart_reception¥UART_2.c
【注 意 事 項】	無し

【関 数 名】	uart_rece_err_int
【概 要】	受信エラー発生の原因をerror_codeに格納します。
【要 因】	INTSRE0 UART0受信エラー発生
【 call 関数】	無し
【変 数】	unsigned char reception_error_ram エラーデータを格納 unsigned char error_code エラーの発生要因を格納
【フ ァ イ ル 名】	uart_reception¥UART_2.c
【注 意 事 項】	無し



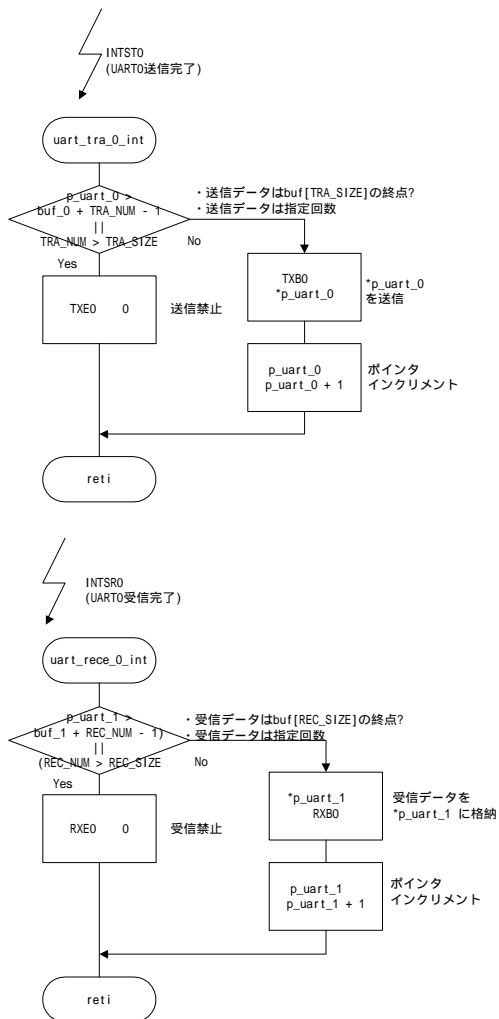
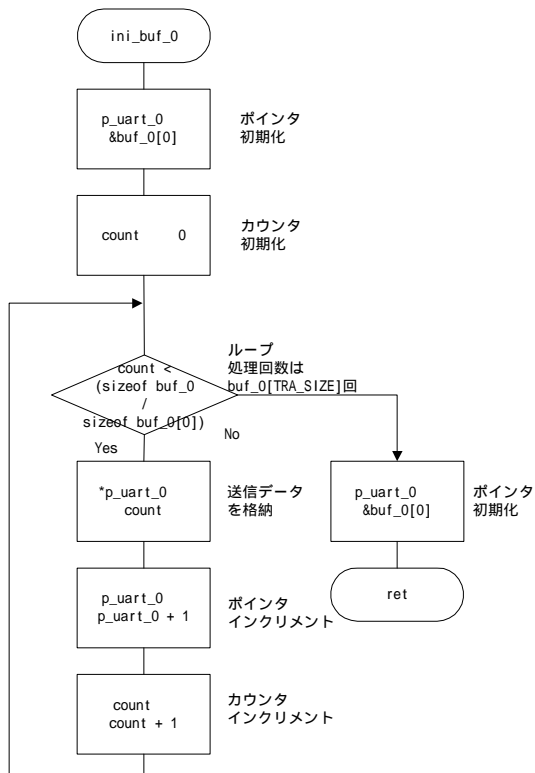
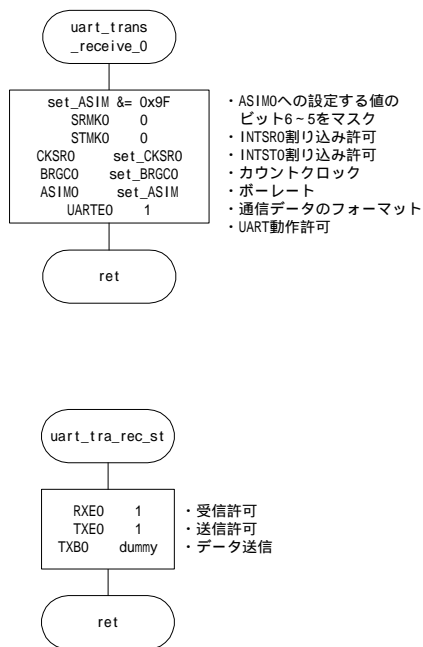
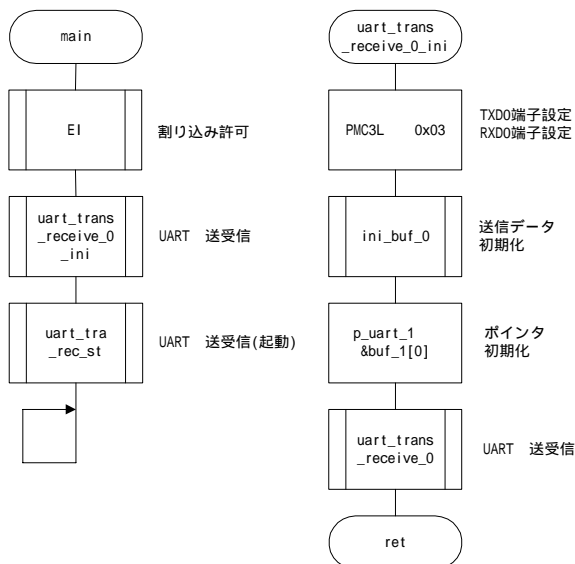
【機能】	送受信準備 (URTE0 = 1、TEX0 = 1、RXE0 = 1)の設定を行ったあとで送信バッファレジスタに(TXB0)に送信データ書き込むことで、データを送信します。 またスタートビットの検出によりデータを受信します	
【関数名】	uart_trans_receive_0	
【引数】	unsigned char set_CKSRO unsigned char set_BRGCO unsigned char set_ASIM	クロックの分周 ボーレートの設定 転送データのフォーマット
【処理内容】	UART送受信動作の設定を行います。設定は以下の通りです。 ボーレート：9600bps、パリティ・ビット：なし データ長：8ビット、ストップ・ビット：1ビット 送信データ：0~9、転送方向：LSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールしてください。 ・コールする前にRXD0端子の設定を行ってください。 ・メインクロックを16MHzに設定してください。 ・uart_tra_rec_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	SRICO STICO CKSRO BRGCO ASIMO	INTSRO割り込みのマスクとレベルの設定 INTSTO割り込みのマスクとレベルの設定 基本クロックの選択 ボー・レートを制御 UART0の転送動作を制御
【call関数】	main	メイン関数
【変数】	unsigned char buf_0[TRA_SIZE] unsigned char buf_1[REC_SIZE] unsigned char p_uart_0 unsigned char p_uart_1	送信データを格納 受信データを格納 送信データ格納先のアドレス 受信データ格納先のアドレス
【割り込み】	uart_tra_0_int uart_rece_0_int	
【割り込み要因】	INTSTO INTSRO	
【ファイル名】	uart_trans_receive_0\UART_3.c , uart_trans_receive_0\MAIN.C	
【注意事項】	<ul style="list-style-type: none"> ・UARTnを使用する場合は、必ずUARTnに関連する外部端子をコントロール・モードに設定した後で、UARTnの設定を行ってください。 ・RXE端子と送信、受信データ格納先の初期化は、uart_trans_receive_0_ini関数で行っています。 	

【関 数 名】	uart_tra_rec_st
【引 数】	unsigned char dummy 初回送信データ
【処 理 内 容】	uart_trans_receive_0の起動関数です。
【起 動 方 法】	uart_trans_receive_0関数の後にコールしてください。
【使 用 S F R】	TXE0 送信許可 TXB0 送信バッファ RXE0 受信許可
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	uart_trans_receive_0UART_3.c
【注 意 事 項】	無し

割り込み関数

【関 数 名】	uart_rece_0_int
【概 要】	・ポインタ操作により、受信データをにbuf_1[REC_SIZE]に格納します。 ・ポインタインクリメントで次の受信データ格納先を指定します。
【要 因】	INTSR0 UART0受信完了
【 call 関数】	無し
【変 数】	unsigned char buf_1[REC_SIZE] 受信データを格納 unsigned char p_uart_1 受信データ格納先のアドレス
【フ ァ イ ル 名】	uart_trans_receive_0UART_3.c
【注 意 事 項】	無し

【関 数 名】	uart_tra_0_int
【概 要】	・ポインタ操作により、buf_0[TRA_SIZE] のデータを送信します。 ・ポインタのインクリメントで、次の送信データ格納先を指定します。
【要 因】	INTST0 UART0送信完了
【 call 関数】	無し
【変 数】	unsigned char buf_0[TRA_SIZE] 送信データを格納 unsigned char p_uart_0 送信データ格納先のアドレス
【フ ァ イ ル 名】	uart_trans_receive_0UART_3.c
【注 意 事 項】	無し



シリアル・インターフェース CSI 送信(CSIn) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

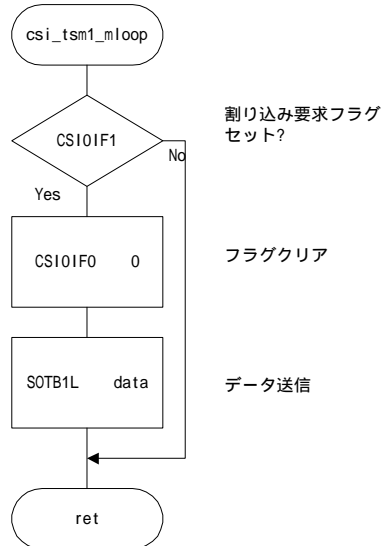
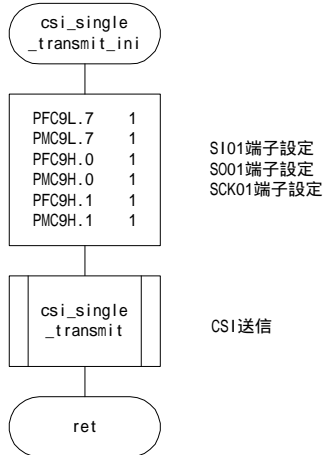
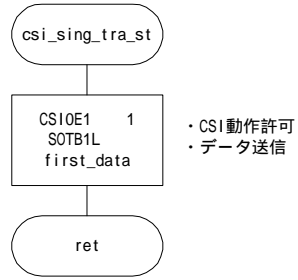
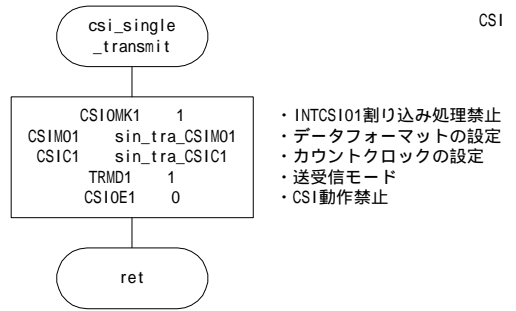
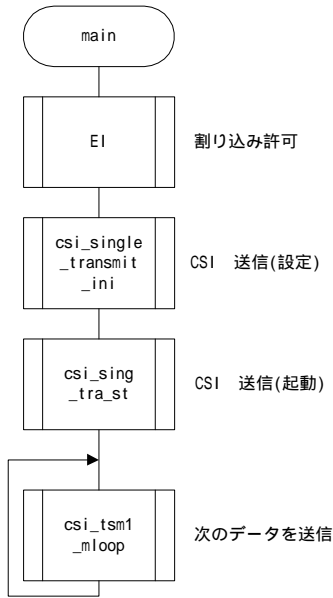
【機能】	送受信モード(CSIMA1レジスタのTRMD1ビット= 1) は、送信データバッファレジスタSOTB1(16ビット) / SOTB1L (8ビット) へのライトで送信動作を開始します。
【関数名】	csi_single_transmit
【引数】	unsigned char sin_tra_CSIM01 転送データのフォーマット unsigned char sin_tra_CSIC1 クロックの分周
【処理内容】	CSI送信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 データ長：8ビット 転送方向：MSB
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にS001、SCK01端子の設定を行ってください。 ・スレーブ側の通信準備が終わった後でコールして下さい。 ・csi_sing_tra_st関数のコールにより通信を開始して下さい。
【設定】	CSIOIC1 INTCSI01割り込みのマスキとレベルの選択 CSIM01 CSIO1の動作を制御 CSIC1 転送動作、クロックの選択
【call関数】	main メイン関数
【変数】	無し
【割り込み】	無し
【ファイル名】	csi_single_transmit%csi_tsm1.c , csi_single_transmit%MAIN.C
【注意事項】	<ul style="list-style-type: none"> ・ S001、SCK01端子と送信データの初期化はcsi_single_transmit_ini関数で行っています。 ・通信を継続する為にメインループ内にcsi_tsm1_mloop関数をコールして下さい。

シリアル・インターフェース CSI 送信 (CSIn) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

【関 数 名】	csi_tsm1_mloop	
【引 数】	unsigned char data	次の転送データ
【概 要】	次のデータを送信します。	
【起 動 方 法】	メインループでコールして下さい。	
【 call 関数】	無し	
【変 数】	無し	
【フ ァ イ ル 名】	csi_single_transmit¥csi_tsm1.c	
【注 意 事 項】	無し	

【関 数 名】	csi_sing_tra_st	
【引 数】	unsigned char first_data	初回送信データ
【処 理 内 容】	csi_single_transmitの起動関数です。	
【起 動 方 法】	csi_single_transmit関数の後にコールしてください。	
【使 用 S F R】	CS10E1 CS11動作許可 S0TB1L 送信バッファ	
【 call 関数】	無し	
【変 数】	無し	
【フ ァ イ ル 名】	csi_single_transmit¥csi_tsm1.c	
【注 意 事 項】	無し	



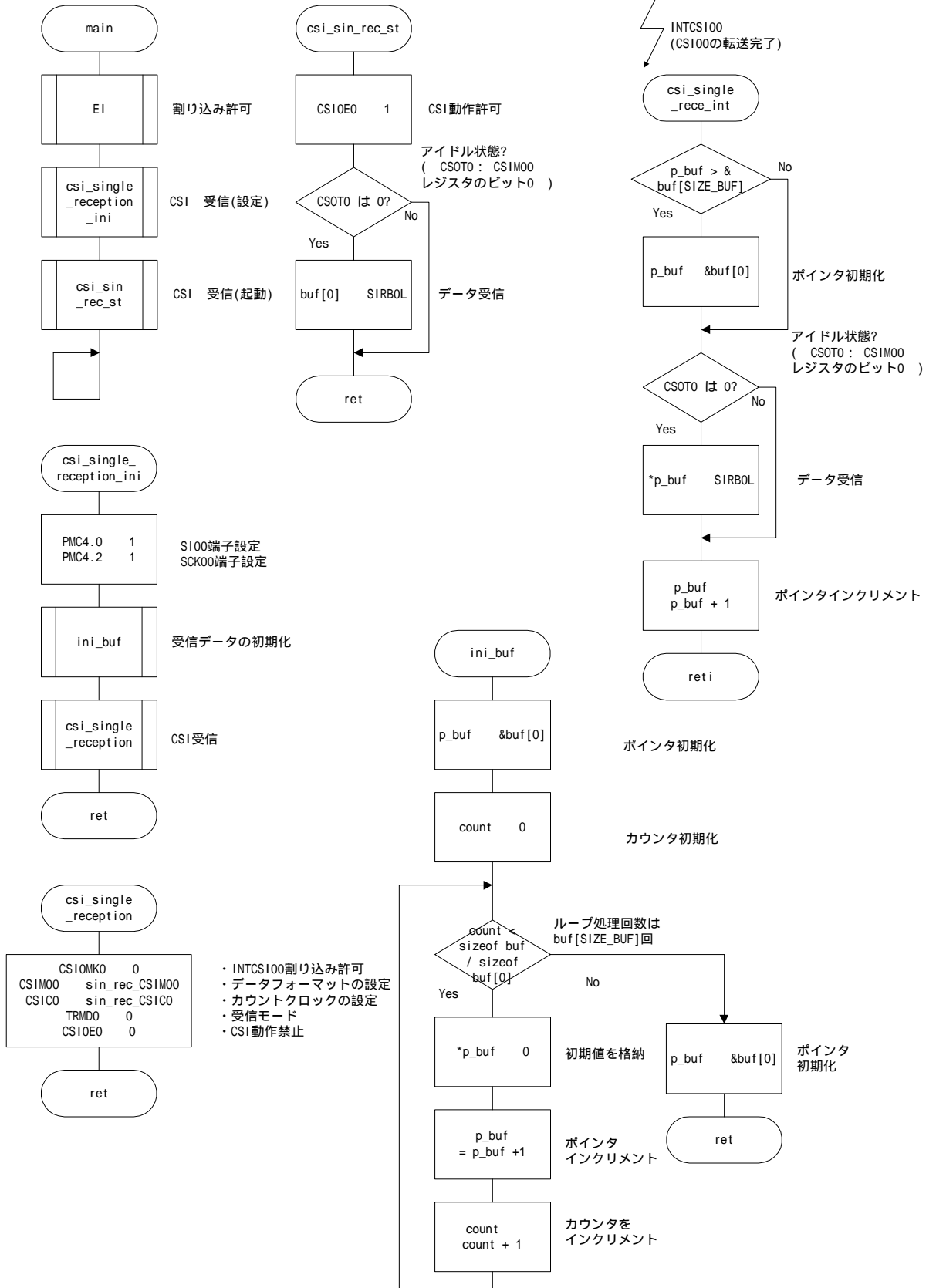
シリアル・インターフェース CSI 受信 (CSIn) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

【機能】	受信専用モード(CSIMAO レジスタのTRMD0ビット= 0) の場合、受信データバッファレジスタSIRB0(16ビット) / SIRB0L (8ビット) のリードで受信動作を開始します。	
【関数名】	csi_single_reception	
【引数】	unsigned char sin_rec_CSIM00 unsigned char sin_rec_CSIC0	転送データのフォーマット クロックの分周
【処理内容】	CSI受信動作の設定を行います。設定は以下の通りです。 カウントクロック：スレーブモード データ長：8ビット 転送方向：MSB	
【起動方法】	<ul style="list-style-type: none"> ・コールする前にS100、SCK00端子の設定を行ってください。 初期設定でコールして下さい。 ・マスタ側の通信動作が開始前にコールして下さい。 ・csi_sin_rec_st関数のコールにより通信を開始して下さい。 	
【設定】	CSI0IC0 CSIM00 CSIC0	INTCSI00割り込みのマスクとレベルの選択 CSI00の動作を制御 転送動作、クロックの選択
【call 関数】	main	メイン関数
【変数】	unsigned char buf [SIZE_BUF] unsigned char *p_buf	受信データを格納 受信データ格納先のアドレス
【割り込み】	csi_single_reception_interrupt	
【ファイル名】	csi_single_reception¥csi_rss0.c , csi_single_reception¥MAIN.C	
【注意事項】	<ul style="list-style-type: none"> ・S100、SCK00端子の設定と送信、受信データ格納先の初期化は、csi_single_reception_ini関数で行っています。 	

【関 数 名】	csi_sin_rec_st
【引 数】	無し
【処 理 内 容】	csi_single_receptionの起動関数です。
【起 動 方 法】	csi_single_reception関数の後にコールしてください。
【使 用 S F R】	CSIOE0 CSIO動作許可 SIRBOL 受信バッファ
【 call 関数】	無し
【変 数】	unsigned char buf[SIZE_BUF] 受信データを格納
【フ ァ イ ル 名】	csi_single_reception¥csi_rss0.c
【注 意 事 項】	無し

割り込み関数

【関 数 名】	csi_single_rece_int
【概 要】	<ul style="list-style-type: none"> ・受信バッファレジスタからデータをリードし、ポインタ操作により、buf[SIZE_BUF]の受信データを1バイトずつ格納します。 ・ポインタのインクリメントで、次の送信データ格納先を指定します。
【要 因】	INTCS100 CS100の転送完了
【 call 関数】	無し
【変 数】	unsigned char buf[SIZE_BUF] 受信データを格納 unsigned char *p_buf 受信データ格納先のアドレス
【フ ァ イ ル 名】	csi_single_reception¥csi_rss0.c
【注 意 事 項】	無し



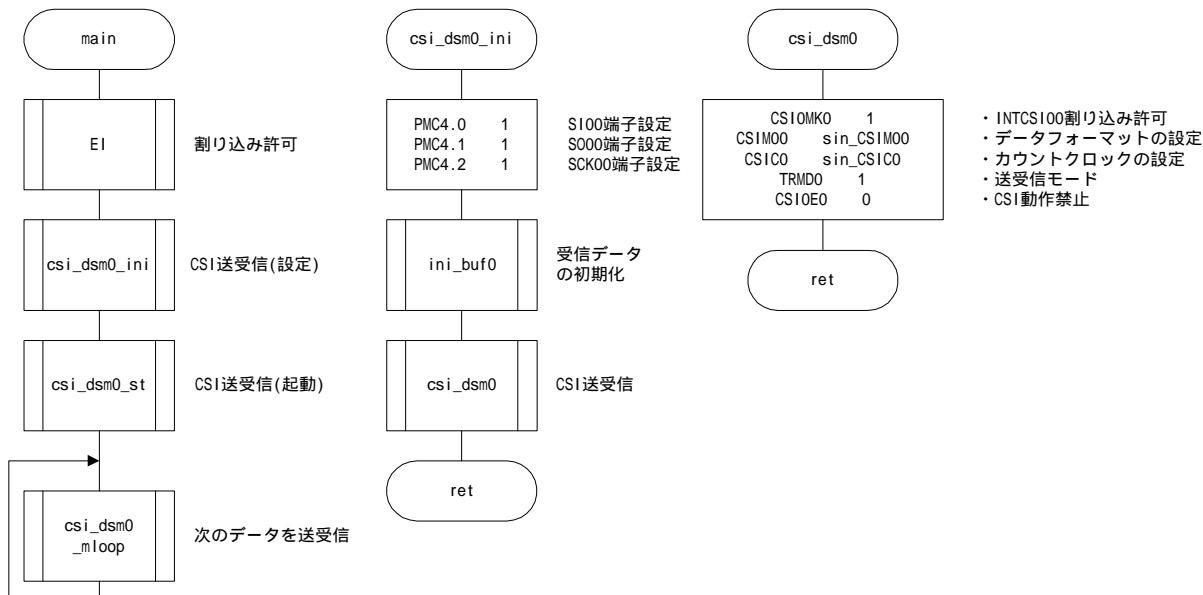
【機能】	CSIMA0 レジスタの(TRMDOビット= 1)の場合、送信データバッファレジスタSOTB0(16ビット) / SOTBL0 (8ビット)へのライトで送受信動作を開始します。
【関数名】	csi_dsm0
【引数】	unsigned char sin_CSIM00 転送データのフォーマット unsigned char sin_CSIC0 クロックの分周
【処理内容】	CSI送受信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 データ長：8ビット 転送方向：MSB
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前に、S100 S000、SCK00端子の設定を行ってください。 ・スレーブ側の通信準備が終わった後でコールして下さい。 ・csi_dsm0_st関数のコールにより通信を開始して下さい。
【設定】	CS10IC0 INTCS100割り込みのマスクとレベルの選択 CSIM00 CS100の動作を制御 CSIC0 転送動作、クロックの選択
【call 関数】	main メイン関数
【変数】	buf_rec0 [SIZE_BUF_REC0] 受信データを格納 first_data 送信データを格納
【割り込み】	無し
【ファイル名】	csi_dsm0¥csi_dsm0.c , csi_dsm0¥MAIN.C
【注意事項】	<ul style="list-style-type: none"> ・S100、S000、SCK00端子の設定と送信、受信データ格納先の初期化は、bis_receive_ini関数で行っています。 ・通信を継続する為にメインループ内にcsi_dsm0_mloop関数をコールして下さい。

シリアル・インターフェース CSI 送受信 (CSI_n) KF1 / KG1 : n = 0-1

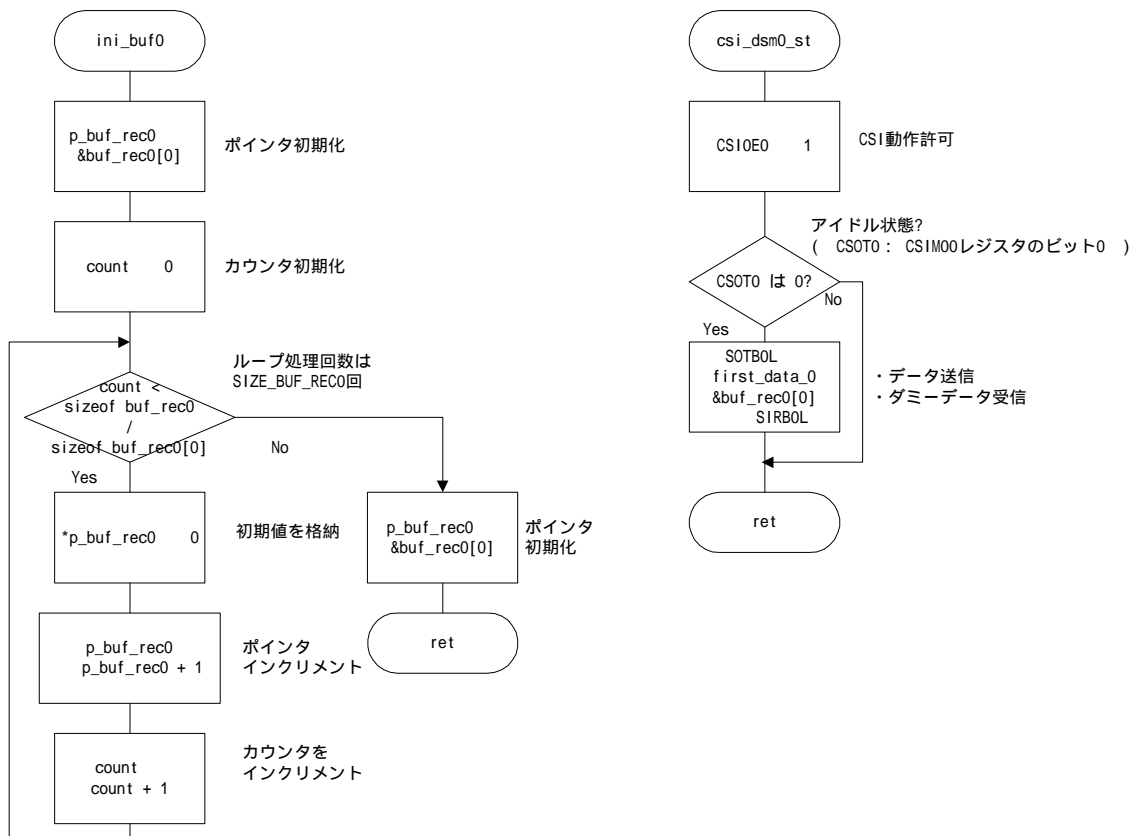
KJ1 : n = 0-2

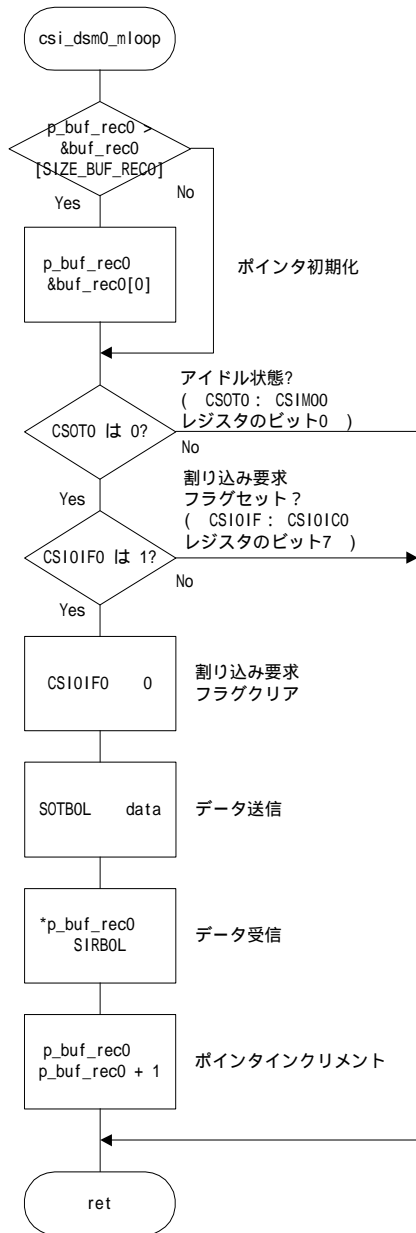
【関 数 名】	csi_dsm0_mloop	
【引 数】	unsigned char data	送信データを格納
【概 要】	次のデータの送受信	
【起 動 方 法】	メインループでコールして下さい。	
【 call 関 数】	無し	
【変 数】	unsigned char buf_rec0 [SIZE_BUF_REC0]	受信データを格納
	unsigned char p_buf_rec0	受信データ格納先のアドレス
【フ ァ イ ル 名】	csi_dsm0¥csi_dsm0.c	
【注 意 事 項】	無し	

【関 数 名】	csi_dsm0_st	
【引 数】	unsigned char first_data	初回送信データ
【処 理 内 容】	csi_dsm0の起動関数です。	
【起 動 方 法】	csi_dsm0関数の後にコールしてください。	
【使 用 S F R】	CSI0E10	CSI0動作許可
	SOTBOL	送信バッファ
	SIRBOL	受信バッファ
【 call 関数】	無し	
【変 数】	buf_rec0[SIZE_BUF_REC0]	受信データを格納
【フ ァ イ ル 名】	csi_dsm0¥csi_dsm0.c	
【注 意 事 項】	無し	



- ・INTCS100割り込み許可
- ・データフォーマットの設定
- ・カウントクロックの設定
- ・送受信モード
- ・CSI動作禁止





シリアル・インターフェース CSI 繰り返しモード送信 (CSI_n) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

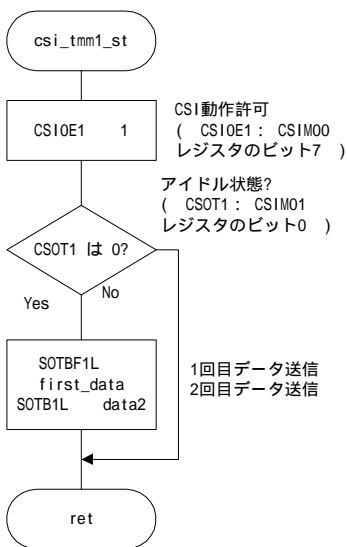
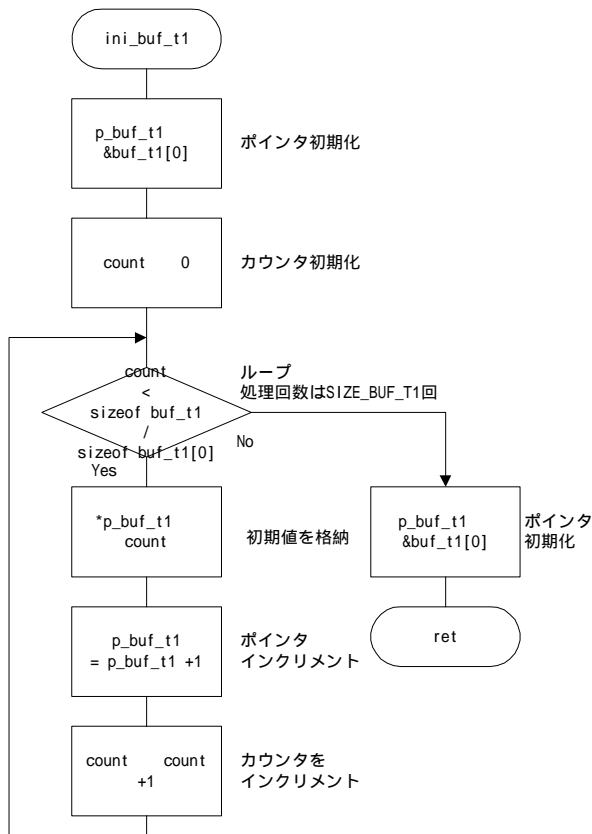
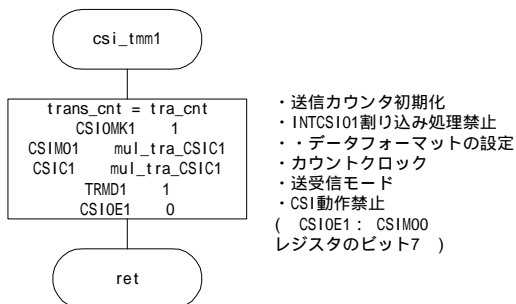
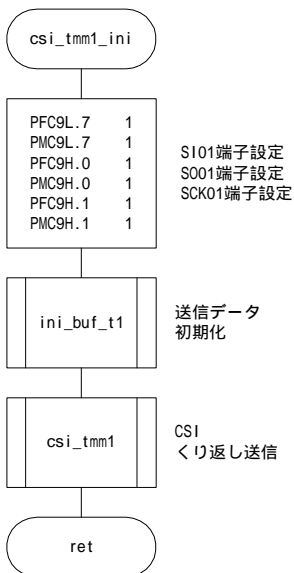
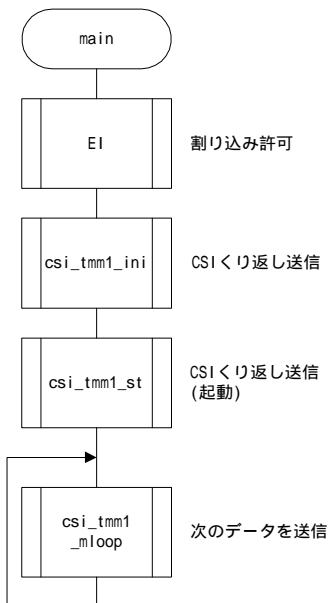
【機能】	割り込み要求発生時に次の転送データを予約する事で可変長のデータを送信することができます。
【関数名】	csi_tmm1
【引数】	unsigned char mul_tra_CSIM01 転送データのフォーマット unsigned char mul_tra_CSIC1 カウントクロック unsigned int tra_cnt データ転送回数
【処理内容】	CSI繰り返しモードの送信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 データ長：8ビット 転送方向：MSB
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にS001、SCK01端子の設定を行ってください。 ・スレーブ側の通信準備が終わった後でコールして下さい。 ・csi_tmm1_st関数のコールにより通信を開始して下さい。
【使用SFR】	CSI0IC1 INTCSI01割り込みのマスクとレベルの選択 CSIM01 CSI01の動作を制御 CSIC1 転送動作、クロックの選択
【call関数】	main メイン関数
【変数】	unsigned char buf_t1[SIZE_BUF_T1] 送信データを格納 unsigned char p_buf_t1 送信データ格納先のアドレス unsigned int trans_cnt 転送回数カウンタ
【ファイル名】	csi_tmm1¥csi_tmm1.c , csi_tmm1¥MAIN.C
【注意事項】	<ul style="list-style-type: none"> ・S001、SCK1端子の初期化は、csi_tmm1_ini関数で行っています。 ・通信を継続する為にメインループ内にcsi_tmm1_mloop関数をコールして下さい。

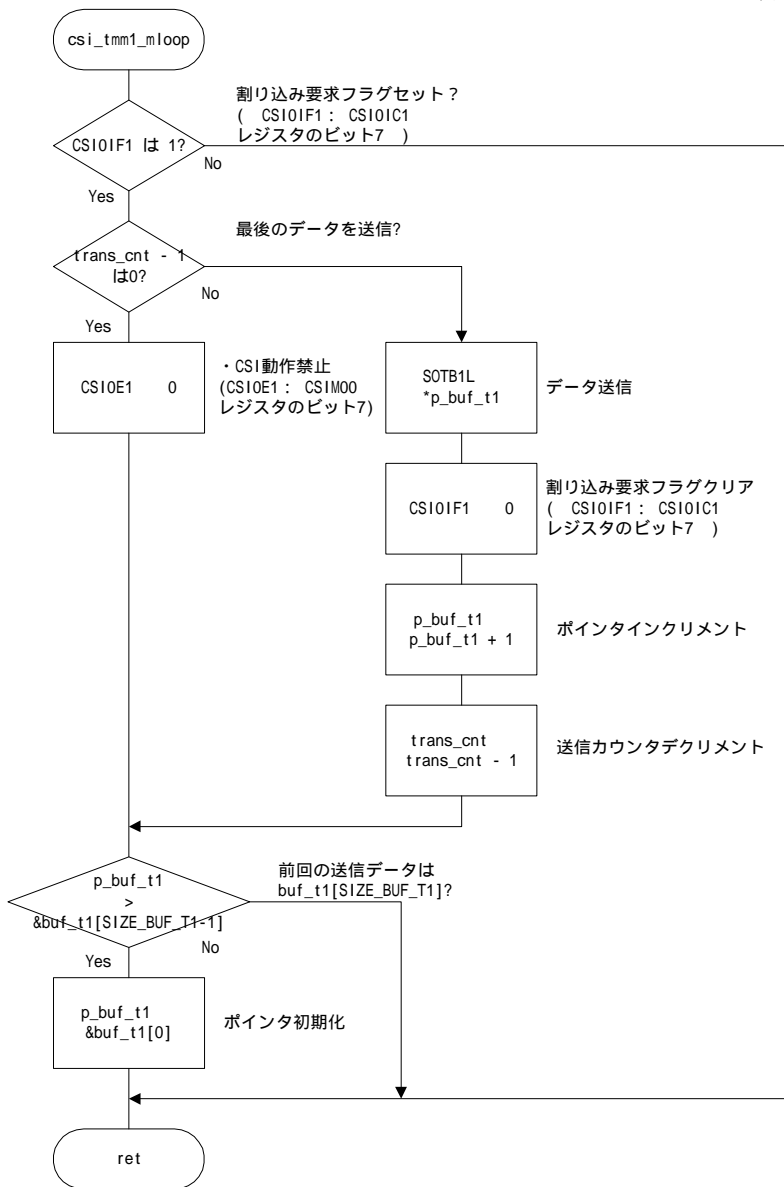
シリアル・インターフェース CSI 繰り返しモード送信 (CSI_n) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

【機能】	割り込み要求発生時に次の転送データを予約する事での可変長のデータを送信することができます。
【関数名】	csi_tmm1_mloop
【引数】	無し
【処理内容】	・CSI0IF1がセットされたときに次のデータの送信予約を行います。 ・転送動作回数のカウンタを監視し、変数trans_cnt回転送後に送信を停止します。
【起動方法】	メインループにてコールして下さい。
【使用SFR】	無し
【call関数】	main メイン関数
【変数】	unsigned char buf_t1[SIZE_BUF_T1] 送信データ unsigned char p_buf_t1 送信データの格納先のアドレス unsigned int trans_cnt 転送回数カウンタ
【ファイル名】	csi_tmm1¥csi_tmm1.c , csi_tmm1¥MAIN.C
【注意事項】	無し

【関数名】	csi_tmm1_st
【引数】	unsigned char first_data 初回送信データ unsigned char data_2 2回目送信データ
【処理内容】	csi_tmm1の起動関数です。
【起動方法】	csi_tmm1関数の後にコールしてください。
【使用SFR】	CSI0E1 CSI1動作許可 S0TBF1L 初回送信バッファ S0TB1L 送信バッファ
【call関数】	無し
【変数】	無し
【ファイル名】	csi_tmm1¥csi_tmm1.c , csi_tmm1¥MAIN.C
【注意事項】	無し NEC Electronics Corporation January 2004





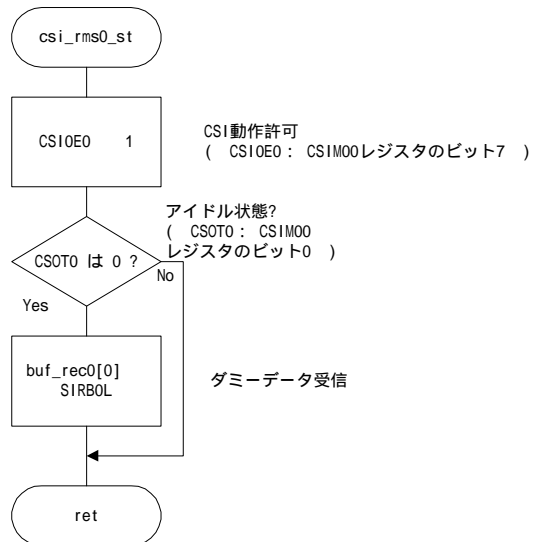
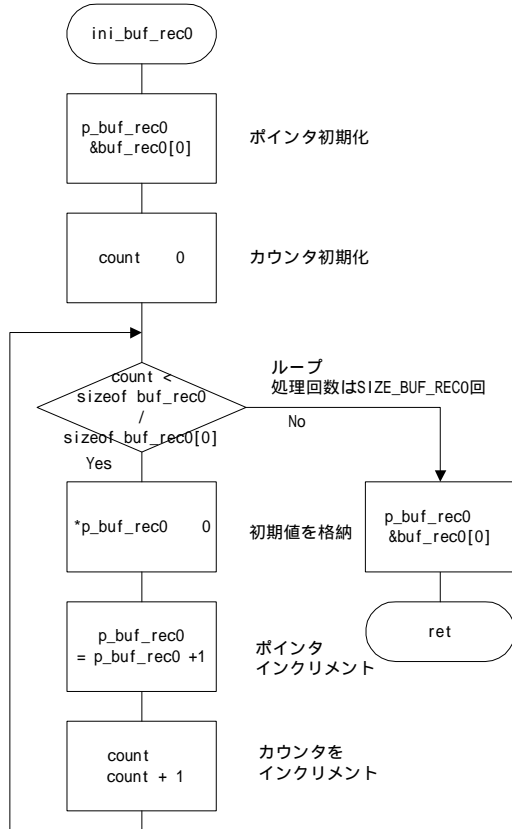
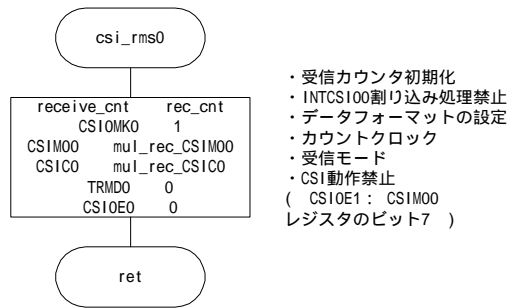
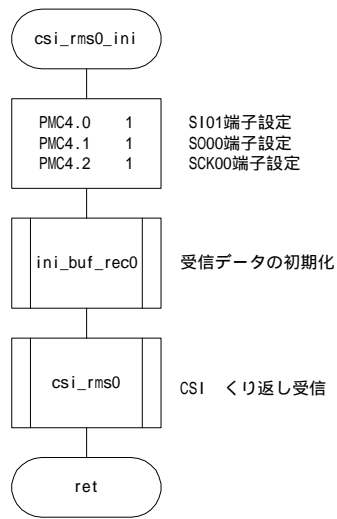
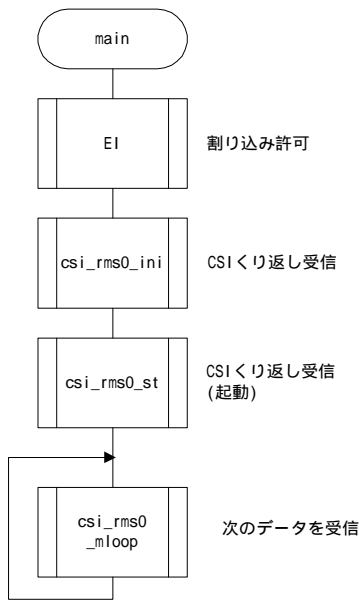
シリアル・インターフェース CSI 繰り返しモード受信(CSIn) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

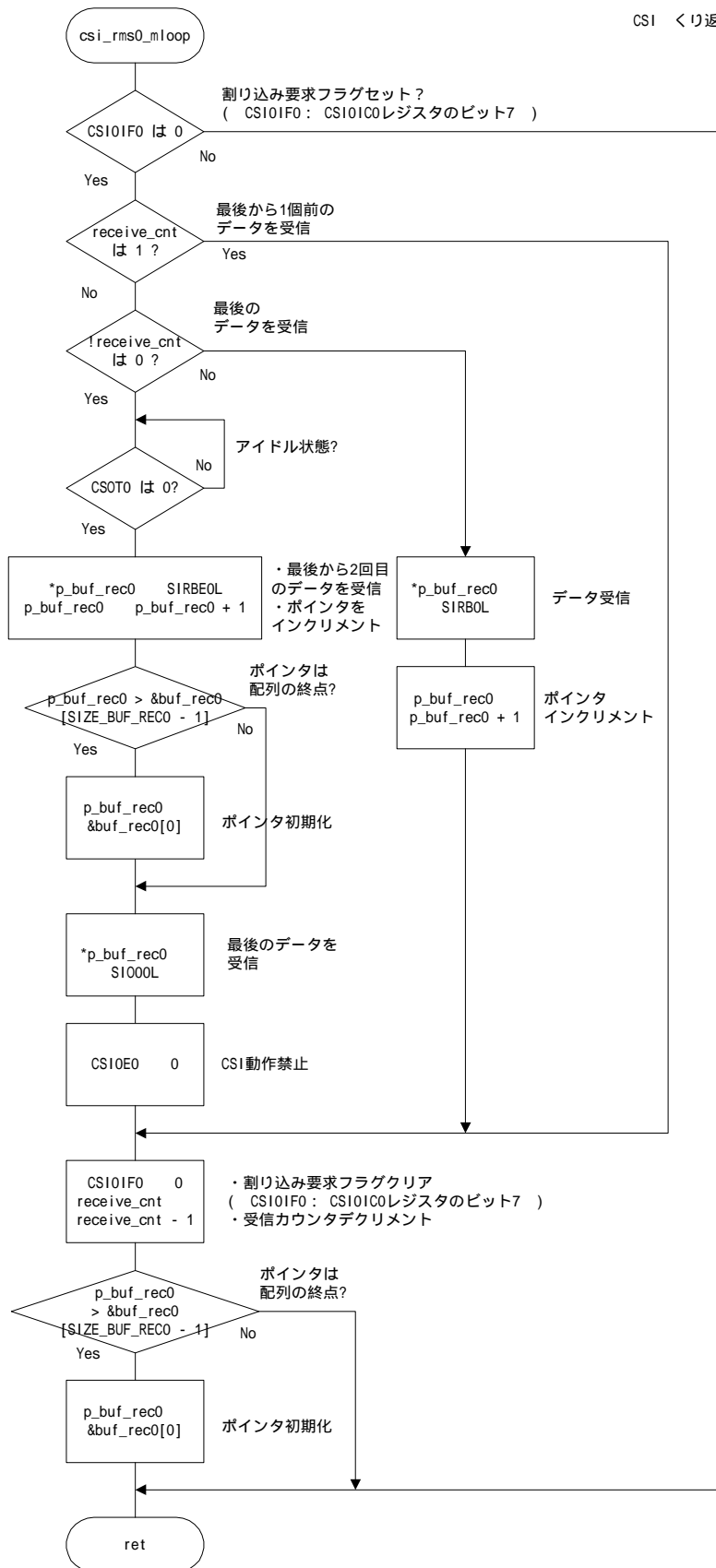
【機能】	割り込み要求発生時に次の転送データを予約する事での可変長のデータを受信することができます。
【関数名】	csi_rms0
【引数】	unsigned char mul_rec_CSIM00 転送データのフォーマット unsigned char mul_rec_CSIC0 カウントクロック unsigned int rec_cnt データ転送回数
【処理内容】	CSI繰り返しモードの受信動作の設定を行います。設定は以下の通りです。 カウントクロック：スレーブモード データ長：8ビット 転送回数：10回 転送方向：MSB
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にS100、SCK00端子の設定を行ってください。 ・マスタ側の通信動作が開始前にコールして下さい。 ・csi_rms0_st関数のコールにより通信を開始して下さい。
【使用SFR】	CSIC0 INTCS100割り込みのマスクとレベルの選択 CSIM00 CS100の動作を制御 CSIC0 転送動作、クロックの選択
【call関数】	main メイン関数
【変数】	unsigned char buf_rec0 [SIZE_BUF_REC0] 受信データを格納 unsigned char *p_buf_rec0 受信データ格納先のアドレス unsigned int receive_cnt 転送回数カウンタ
【ファイル名】	csi_rms0%csi_rms0.c , csi_rms0%MAIN.C
【注意事項】	<ul style="list-style-type: none"> ・S100、SCK00端子の設定の初期化は、csi_rms0_ini関数で行っています。 ・通信を継続する為にメインループ内にcsi_rms0_mloop関数をコールして下さい。

シリアル・インターフェース CSI 繰り返しモード受信 (CSI_n) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

【機能】	割り込み要求発生時に次の転送データを予約する事での可変長のデータを受信することができます。
【関数名】	csi_rms0_mloop
【引数】	無し
【処理内容】	CSI0IF0がセットされたときに次のデータの受信予約を行います。
【起動方法】	メインループでコールして下さい。
【使用 S F R】	無し
【call 関数】	無し
【変数】	unsigned buf_rec0[SIZE_BUF_REC0] 受信データを格納 unsigned char *p_buf_rec0 受信データ格納先のアドレス unsigned int receive_cnt 転送回数カウンタ
【ファイル名】	csi_rms0¥csi_rms0.c
【注意事項】	無し

【関数名】	csi_rms0_st
【引数】	無し
【処理内容】	csi_rms0の起動関数です。
【起動方法】	csi_rms0関数の後にコールしてください。
【使用 S F R】	CSI0E0 CSI0動作許可 SIRBOL 受信バッファ
【call 関数】	無し
【変数】	unsigned char buf_rec0[SIZE_BUF_REC0] 受信バッファ
【ファイル名】	csi_rms0¥csi_rms0.c
【注意事項】	無し



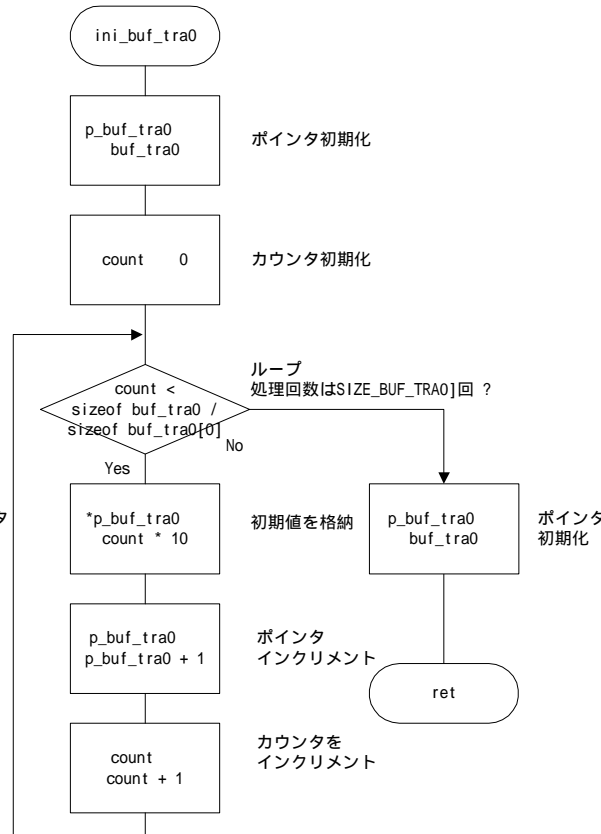
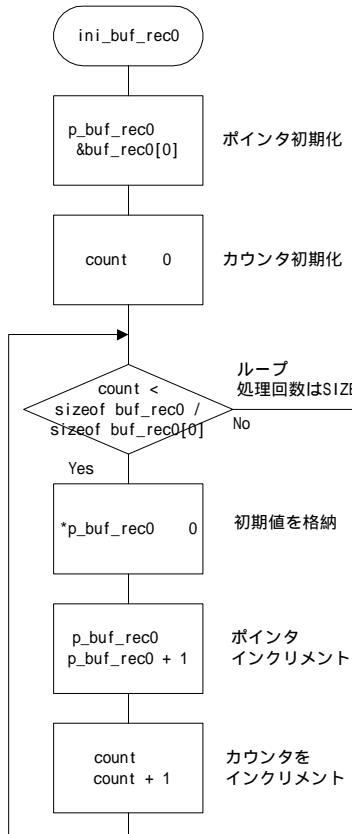
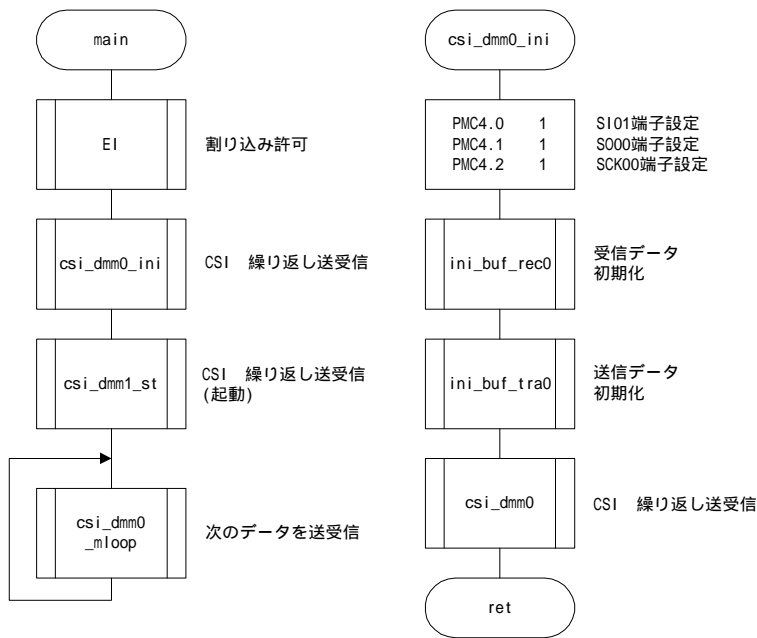


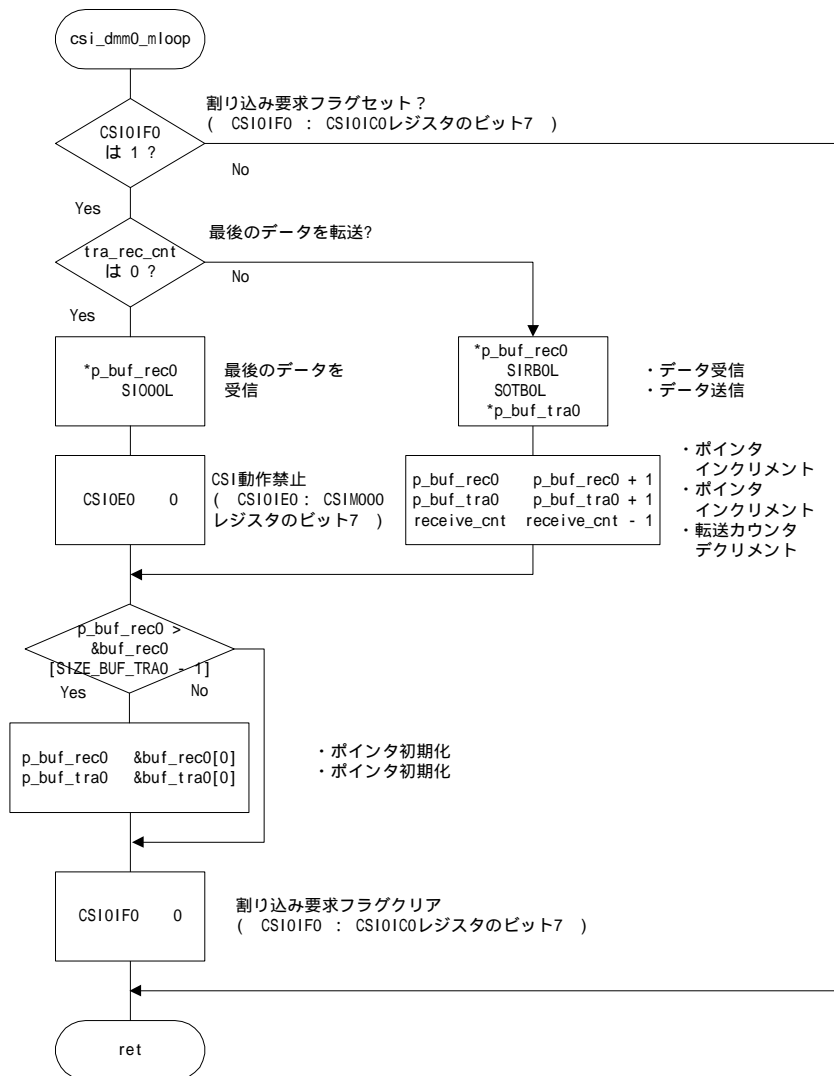
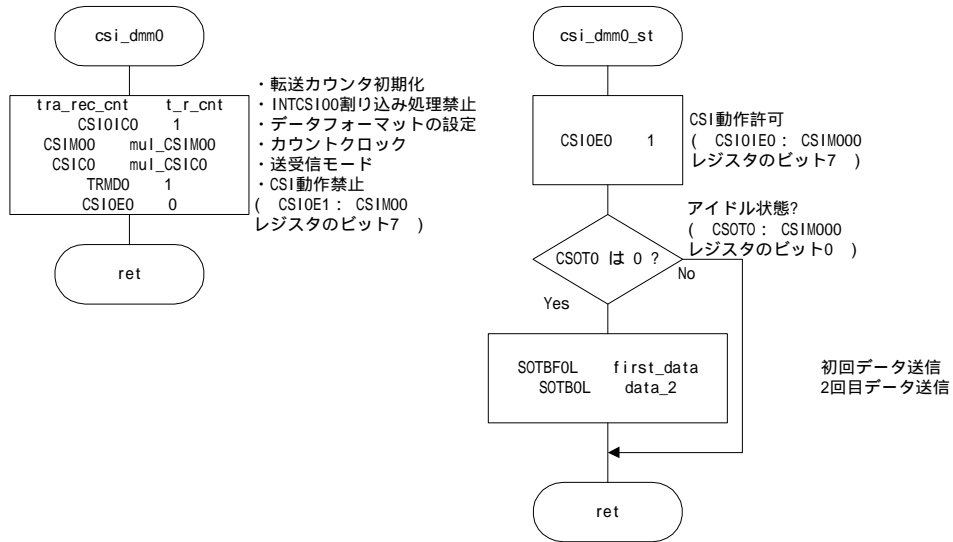
シリアル・インターフェース CSI 繰り返しモード送受信 (CSIn) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

【機能】	割り込み要求発生時に次の転送データを予約する事での可変長のデータを送受信することができます。	
【関数名】	csi_dmm0	
【引数】	unsigned char mul_CSIM00 unsigned char mul_CSIC0 unsigned int t_r_cnt	転送データのフォーマット カウントクロック データ転送回数
【処理内容】	CSI繰り返しモードの送受信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/64 データ長：8ビット 初回送信データ：0xA5 2回目以降の送信データ：0xA5、0~70(10進) 転送回数：10回 転送方向：MSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にS100、S000、SCK00端子の設定を行ってください。 ・スレーブ側の通信準備が終わった後でコールして下さい。 ・csi_dmm0_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	CS10IC0 CS1M00 CS1C0	INTCS100割り込みのマスクとレベルの選択 CS100の動作を制御 転送動作、クロックの選択
【call関数】	main	メイン関数
【変数】	unsigned int tra_rec_cnt	データ転送回数カウンタ
【ファイル名】	csi_dmm0¥csi_dmm0.c , csi_dmm0¥MAIN.C	
【注意事項】	<ul style="list-style-type: none"> ・S100、S000、SCK00端子の設定と送信、受信データ格納先の初期化は、csi_dmm0_ini関数で行っています。 ・通信を継続する為にメインループ内にcsi_dmm0_mloop関数をコールして下さい。 	

【機能】	割り込み要求発生時に次の転送データを予約する事での可変長のデータを送信、受信することができます。
【関数名】	csi_dmm0_mloop
【引数】	無し
【処理内容】	転送動作回数がTRA_REC_CNT回転送後に通信を停止します。
【起動方法】	メインループにてコールして下さい。
【使用 S F R】	無し
【call 関数】	main メイン関数
【変数】	unsigned char buf_tra0[SIZE_BUF_TRA0] 送信データ unsigned char p_buf_tra0 送信データの格納先のアドレス unsigned char buf_rec0[SIZE_BUF_REC0] 受信データ unsigned char p_buf_rec0 受信データの格納先のアドレス unsigned int tra_rec_cnt データ転送回数 カウンタ
【ファイル名】	csi_dmm0¥csi_dmm0.c
【注意事項】	無し

【関数名】	csi_dmm0_st
【引数】	unsigned char first_data 初回送信データ unsigned char data_2 2回目送信データ
【処理内容】	csi_dmm0の起動関数です。
【起動方法】	csi_dmm0関数の後にコールしてください。
【使用 S F R】	CSIOE10 CSIO動作許可 SOTBF0L 初回送信バッファ SOTB0L 送信バッファ
【call 関数】	無し
【変数】	unsigned char buf_rec0[SIZE_BUF_REC0] 受信バッファ
【ファイル名】	csi_dmm0¥csi_dmm0.c
【注意事項】	無し



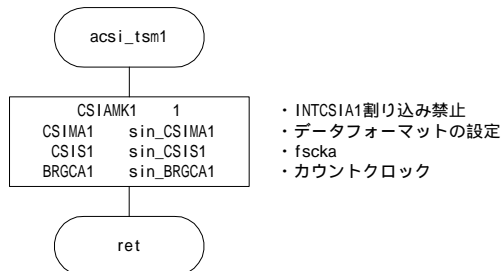
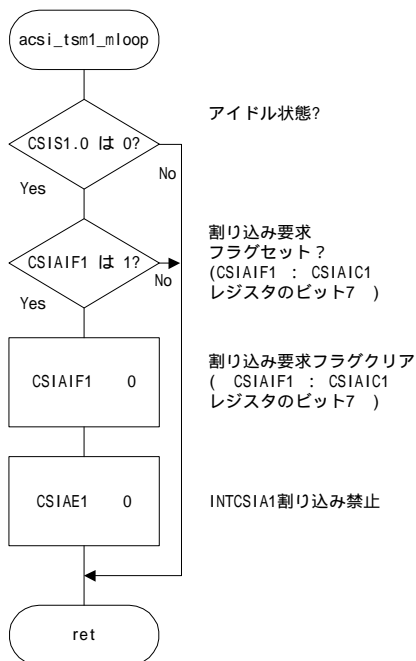
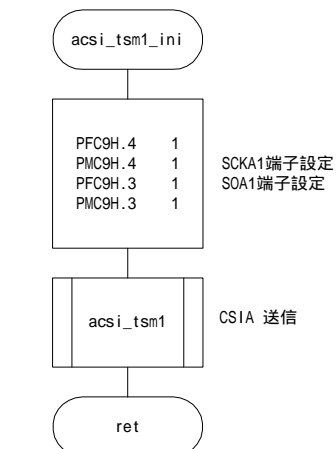
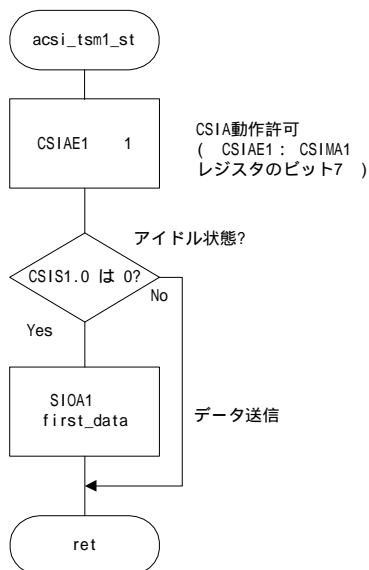
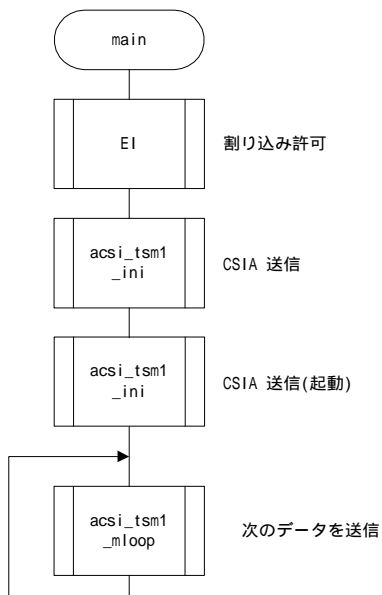


シリアル・インターフェース CSIA 単発送信 (CSIA_n) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

【機能】	CSIMA1 レジスタのCSIAE1ビット= 1、ATE1ビット = 0のとき、送信データバッファレジスタSIOA1へのライトで送信動作を開始します。	
【関数名】	acsi_tsm1	
【引数】	unsigned char sin_CSIMA1 unsigned char sin_CSIS1 unsigned char sin_BRGCA1	転送データのフォーマット fscka(入力クロック) クロックの分周
【処理内容】	CSIA単発モードの送信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 転送方向：MSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にSOA1、SCKA1端子の設定を行ってください。 ・スレーブ側の通信準備が終わった後でコールして下さい。 ・acsi_tsm1_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	CSIAIC1 CSIMA1 CSIS1 BRGCA1	INTCSIA1割り込みのマスクとレベルの選択 CSIA1の動作を制御 入力クロックの選択と動作ステータス シリアル転送スピードを制御
【call関数】	main	メイン関数
【変数】	無し	
【ファイル名】	acsi_tsm1¥acsi_tsm1.c , acsi_tsm1¥MAIN.C	
【注意事項】	SOA1、SCKA1端子の設定はacsi_tsm1_ini関数で行っています。	

【関 数 名】	acsi_tsm1_mloop
【概 要】	1バイトデータ転送確認後送信動作を停止します。
【起 動 方 法】	メインループでコールして下さい。
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_tsm1¥acsi_tsm1.c
【注 意 事 項】	無し

【関 数 名】	acsi_tsm1_st
【引 数】	unsigned char first_data 初回送信データ
【処 理 内 容】	acsi_tsm1の起動関数です。
【起 動 方 法】	acsi_tsm1関数の後にコールしてください。
【使 用 S F R】	CSIAE1 CSIA1動作許可 SIOA1 送信バッファ
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_tsm1¥acsi_tsm1.c
【注 意 事 項】	無し



シリアル・インターフェース CSIA 単発受信 (CSIA_n) KF1 / KG1 : n = 0-1

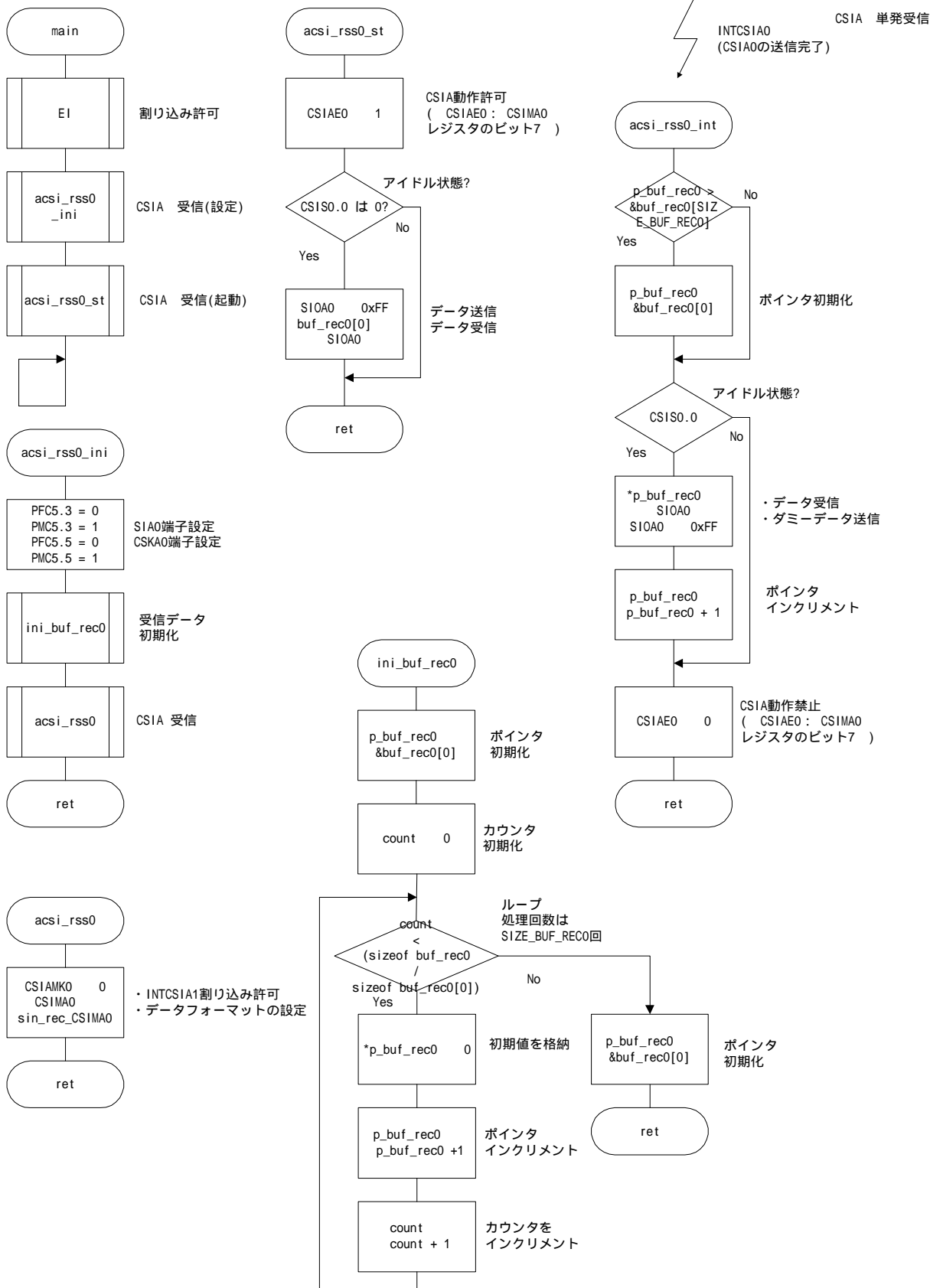
KJ1 : n = 0-2

【機能】	CSIMA0レジスタのCSIAE0ビット= 1、ATE0ビット = 0のとき、送信データバッファレジスタSIOA0へのダミーデータのライトで受信動作を開始します。	
【関数名】	acsi_rss0	
【引数】	unsigned char sin_rec_CSIMA0	転送データのフォーマット
【処理内容】	CSIA単発モードの受信動作の設定を行います。設定は以下の通りです。 カウントクロック：スレーブモード 転送方向：MSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にSIA0、SCKA0端子の設定を行ってください。 ・マスタ側の通信動作が開始前にコールして下さい。 ・acsi_rss0_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	CSIAIC0 CSIMA0 CSISO	INTCSIA0割り込みのマスクとレベルの選択 CSIA0の動作を制御 入力クロックの選択と動作ステータス
【call関数】	main	メイン関数
【変数】	unsigned char buf_rec0[SIZE_BUF_REC0]	受信データを格納
【割り込み】	acsi_rss0_int	
【割り込み要因】	INTCSIA0	
【ファイル名】	acsi_rss0¥acsi_rss0.c , acsi_rss0¥MAIN.C	
【注意事項】	SIA0、SCKA0端子、受信データ初期値の設定はacsi_rss0_ini関数で行っています。	

【関 数 名】	acsi_rss0_st
【引 数】	無し
【処 理 内 容】	acsi_rss0の起動関数です。
【起 動 方 法】	acsi_rss0関数の後にコールしてください。
【使 用 S F R】	CSIAEO CSIAO動作許可 SIOAO 受信バッファ
【 call 関数】	無し
【変 数】	unsigned char buf_rec0[SIZE_BUF_REC0] 受信データを格納
【フ ァ イ ル 名】	acsi_rss0¥acsi_rss0.c , acsi_rss0¥MAIN.C
【注 意 事 項】	無し

割り込み関数

【関 数 名】	acsi_rss0_int
【概 要】	・受信バッファレジスタからデータをリードし、ポインタ操作により、buf_rec0[SIZE_BUF_REC0]の受信データを1バイト格納し、CSIAOを停止します。
【要 因】	INTCSIAO データの受信
【 call 関数】	無し
【変 数】	unsigned char buf_rec0[SIZE_BUF_REC0] 受信データを格納 unsigned char *p_buf_rec0 受信データ格納先のアドレス
【フ ァ イ ル 名】	acsi_rss0¥acsi_rss0.c , acsi_rss0¥MAIN.C
【注 意 事 項】	無し



シリアル・インターフェース CSIA 単発送受信 (CSIA_n) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

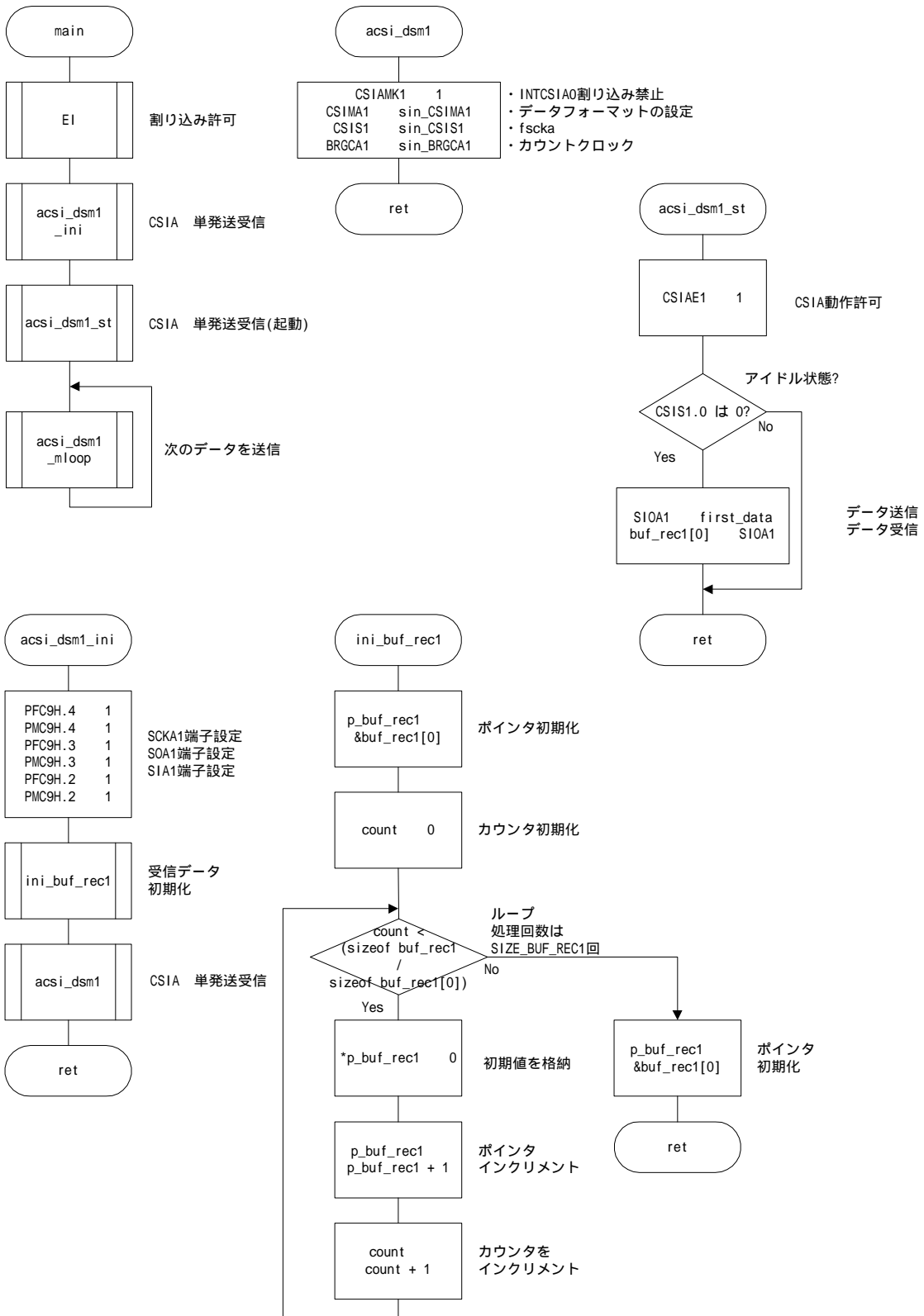
【機能】	CSIMA0 レジスタのCSIAE0ビット= 1、ATE0ビット = 0で、送信データバッファレジスタSIOA0へのライトで送受信動作を開始します。
【関数名】	acsi_dsm1
【引数】	unsigned char sin_CSIMA1 転送データのフォーマット unsigned char sin_CSIS1 fsccka(入力クロック) unsigned char sin_BRGCA1 クロックの分周
【処理内容】	CSIA送信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 転送方向：MSB
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にSIA1、SOA1、SCKA1端子の設定を行ってください。 ・スレーブ側の通信準備が終わった後でコールして下さい。 ・acsi_dsm1_st関数のコールにより通信を開始して下さい。
【使用SFR】	CSIAIC1 INTCSIA1割り込みのマスクとレベルの選択 CSIMA1 CSIA1の動作を制御 CSIS1 入力クロックの選択と動作ステータス BRGCA1 シリアル転送スピードを制御
【call関数】	main メイン関数
【変数】	無し
【ファイル名】	acsi_dsm1¥acsi_dsm1.c , acsi_dsm1¥MAIN.C
【注意事項】	<ul style="list-style-type: none"> ・SIA1、SOA1、SCKA1端子、送信、受信データ初期値の設定はacsi_dsm1_ini関数で行っています。 ・通信を継続する為にメインループ内にacsi_dsm1_mloop関数をコールして下さい。

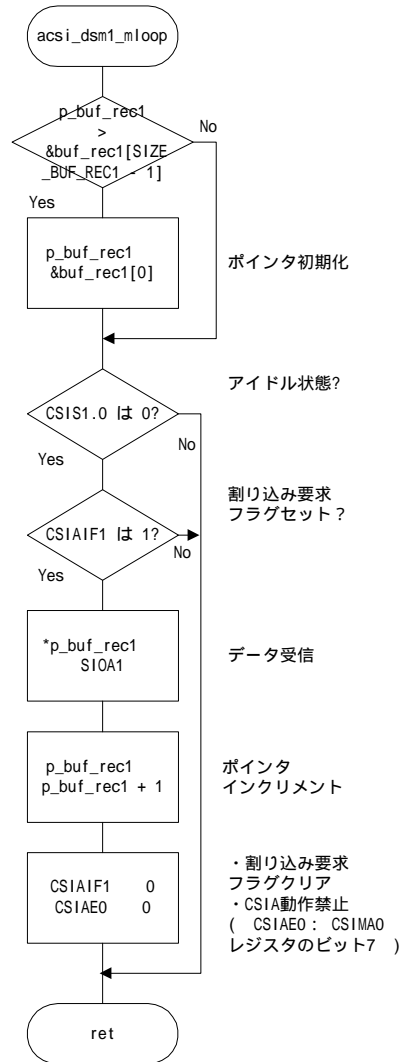
シリアル・インターフェース CSIA 単発送受信 (CSIA_n) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

【関 数 名】	acsi_dsm1_mloop
【概 要】	1バイトデータ転送確認後送受信動作を停止します
【起 動 方 法】	メインループでコールして下さい。
【 call 関数】	無し
【変 数】	unsigned char buf_rec1[SIZE_BUF_REC1] 受信データを格納 unsigned char p_buf_rec1 受信データ格納先のアドレス
【フ ァ イ ル 名】	acsi_dsm1¥acsi_dsm1.c
【注 意 事 項】	・ SIA1、 SOA1、 SCKA1端子、送信データ初期値の設定は acsi_tmm1_ini関数で行っています。

【関 数 名】	acsi_dsm1_st
【引 数】	無し
【処 理 内 容】	acsi_dsm1の起動関数です。
【起 動 方 法】	acsi_dsm1関数の後にコールしてください。
【使 用 S F R】	CSIAE1 CSIA1動作許可 SIOA1 初回受信バッファ SIOA1 初回総信バッファ
【 call 関数】	無し
【変 数】	unsigned char buf_rec1[SIZE_BUF_REC1] 受信データを格納
【フ ァ イ ル 名】	acsi_dsm1¥acsi_dsm1.c
【注 意 事 項】	無し





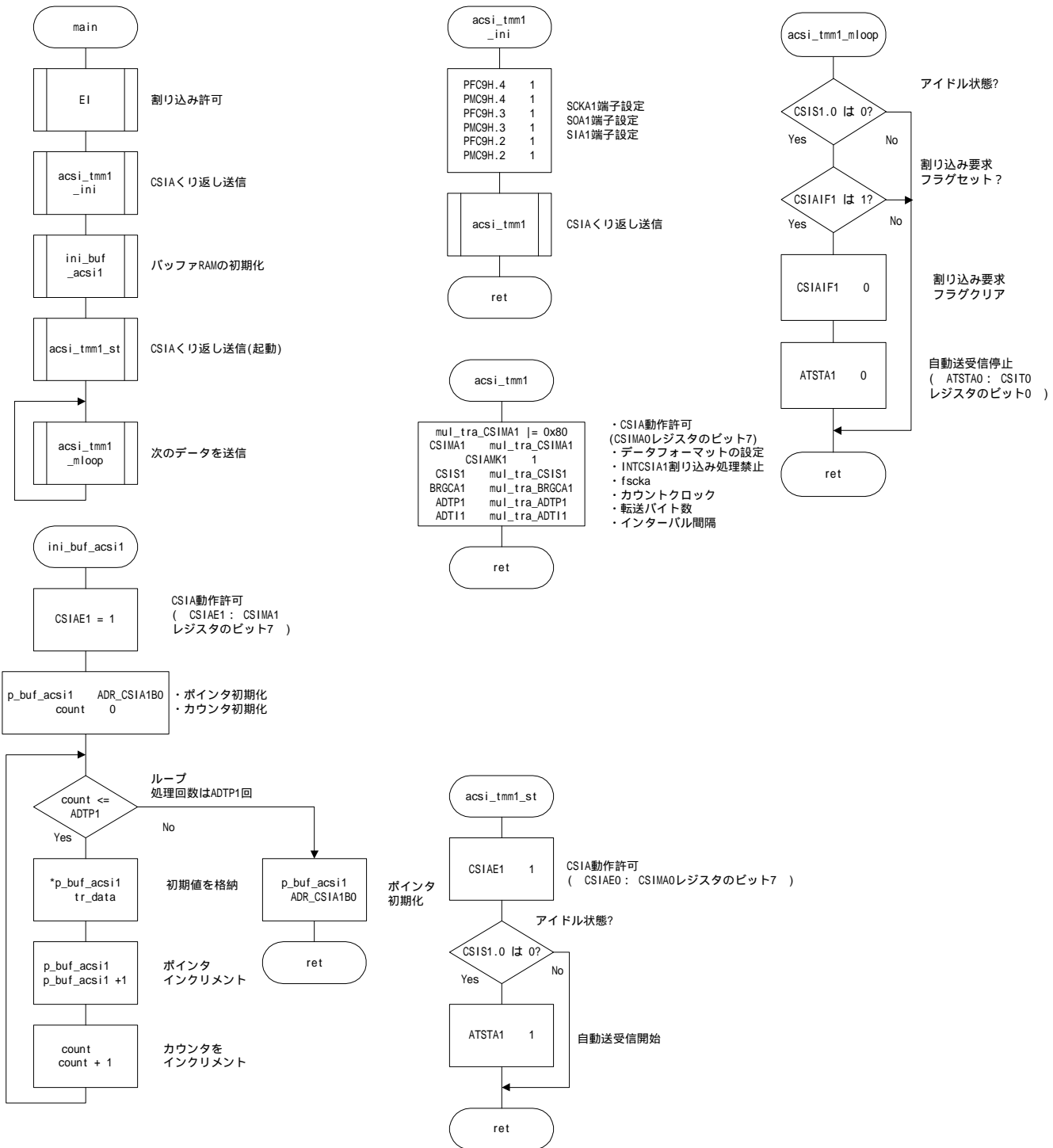
シリアル・インターフェース CSIA 自動モード送信 (CSIA_n) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

【機能】	CSIMA1 レジスタのCSIAE1ビット= 1、ATE1ビット = 1の時 CSIT1レジスタのATST1ビット = 1により、32バイトバッファRAMを 使用した自動送受信を行います。
【関数名】	acsi_tmm1
【引数】	unsigned char mul_tra_CSIS1 fsccka(入力クロック) unsigned char mul_tra_BRGCA1 クロックの分周 unsigned char mul_tra_ADTP1 転送バイト数 unsigned char mul_tra_ADTI1 インターバル間隔 unsigned char mul_tra_CSIMA1 転送データのフォーマット
【処理内容】	CSIA自動モード送信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 データ長：32バイト 転送方向：MSB
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にSOA1、SCKA1端子の設定を行ってください。 ・acsi_tmm1_st関数のコールにより通信を開始して下さい。
【使用SFR】	CSIAIC1 INTCSIA1割り込みのマスクとレベルの選択 CSIMA1 CSIA1の動作を制御 CSIS1 入力クロックの選択と動作ステータス ADTP1 自動データ転送時の転送バイト数 ADTI1 自動データ転送時の1バイト間のインターバル時間 BRGCA1 シリアル転送スピードを制御
【call関数】	main メイン関数
【変数】	無し
【ファイル名】	acsi_tmm1¥acsi_tmm1.c , aaci_tmm1¥MAIN.C
【注意事項】	<ul style="list-style-type: none"> ・SIA1、SOA1、SCKA1端子の設定はacsi_tmm1_ini関数で行っています。 ・通信を継続する為にメインループ内にacsi_tmm1_mloop関数をコールして下さい。

【関 数 名】	acsi_tmm1_mloop
【概 要】	mul_tra_ADTP1で指定したデータ転送確認後送信動作を停止します。
【起 動 方 法】	メインループでコールして下さい。
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_tmm1¥acsi_tmm1.c
【注 意 事 項】	無し

【関 数 名】	ini_buf_acsi1
【引 数】	unsigned char tr_data バッファRAMの初期値
【処 理 内 容】	バッファRAMの初期化を行ないます。
【起 動 方 法】	acsi_tmm1関数の後にコールしてください。
【使 用 S F R】	ADTP1 自動データ転送時の転送バイト数
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_tmm1¥acsi_tmm1.c
【注 意 事 項】	無し

【関 数 名】	acsi_tmm1_st
【引 数】	無し
【処 理 内 容】	acsi_tmm1の起動関数です。
【起 動 方 法】	acsi_tmm1関数の後にコールしてください。
【使 用 S F R】	CSIAE1 CSIA1動作許可 ATSTA1 自動転送開始
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_tmm1¥acsi_tmm1.c
【注 意 事 項】	無し



シリアル・インターフェース CSIA 自動モード受信 (CSIA_n) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

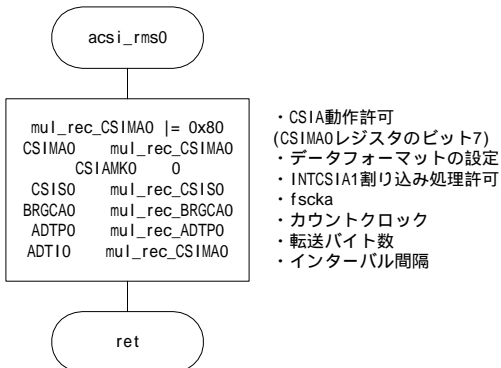
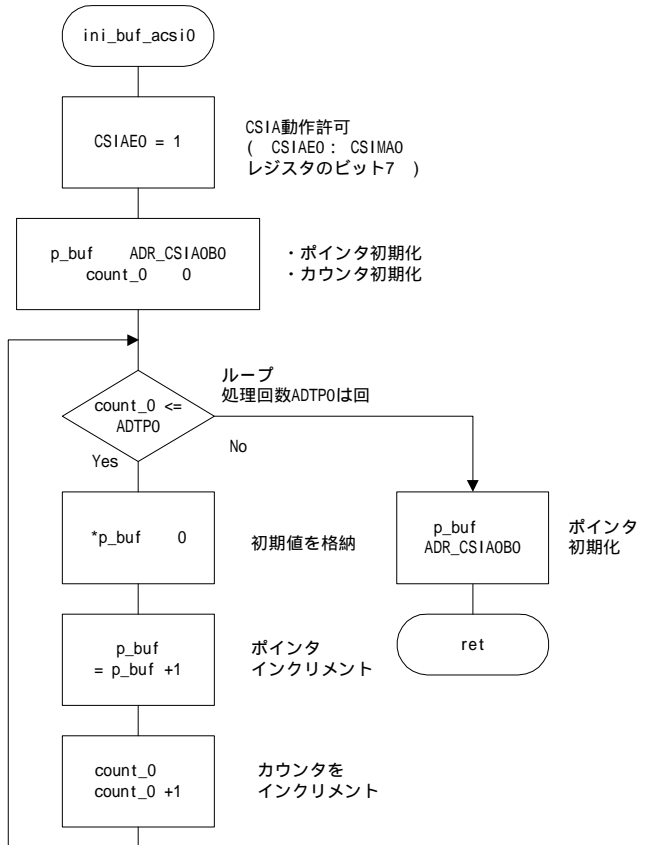
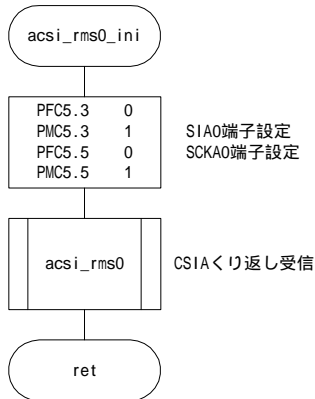
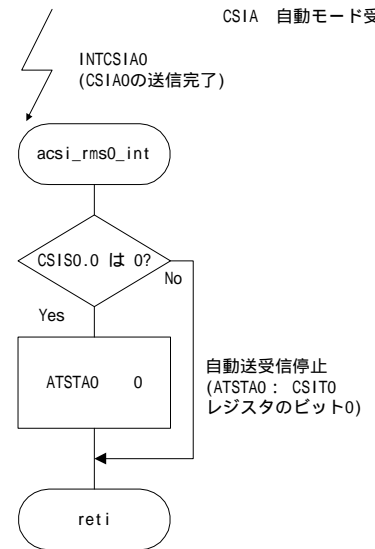
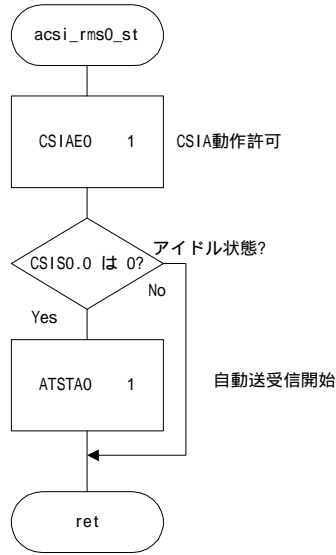
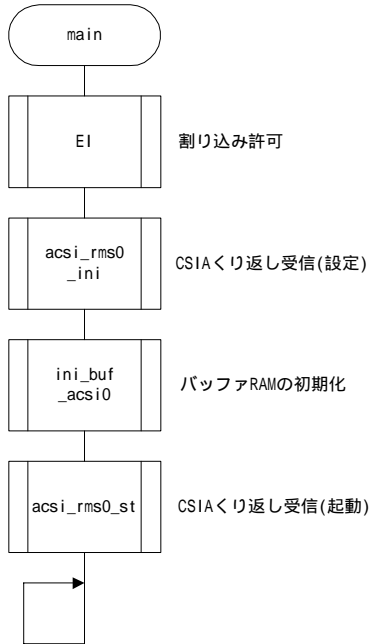
【機能】	CSIMA0 レジスタのCSIAEOビット= 1、ATE0ビット = 1の時 CSIT0レジスタのATST0ビット = 1により、32バイトバッファRAMを使用した自動送受信を行います。										
【関数名】	acsi_rms0										
【引数】	<table> <tr> <td>unsigned char mul_rec_CSIS0</td> <td>fscka(入力クロック)</td> </tr> <tr> <td>unsigned char mul_rec_BRGCA0</td> <td>クロックの分周</td> </tr> <tr> <td>unsigned char mul_rec_ADTP0</td> <td>転送バイト数</td> </tr> <tr> <td>unsigned char mul_rec_ADTI0</td> <td>インターバル間隔</td> </tr> <tr> <td>unsigned char mul_rec_CSIMA0</td> <td>転送データのフォーマット</td> </tr> </table>	unsigned char mul_rec_CSIS0	fscka(入力クロック)	unsigned char mul_rec_BRGCA0	クロックの分周	unsigned char mul_rec_ADTP0	転送バイト数	unsigned char mul_rec_ADTI0	インターバル間隔	unsigned char mul_rec_CSIMA0	転送データのフォーマット
unsigned char mul_rec_CSIS0	fscka(入力クロック)										
unsigned char mul_rec_BRGCA0	クロックの分周										
unsigned char mul_rec_ADTP0	転送バイト数										
unsigned char mul_rec_ADTI0	インターバル間隔										
unsigned char mul_rec_CSIMA0	転送データのフォーマット										
【処理内容】	CSIA自動モード受信動作の設定を行います。設定は以下の通りです。 カウントクロック：スレーブモード データ長：32バイト 転送方向：MSB										
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にSIA0、SCKA0端子の設定を行ってください。 ・acsi_rms0_st関数のコールにより通信を開始して下さい。 										
【使用SFR】	<table> <tr> <td>CSIAIC0</td> <td>INTCSIA0割り込みのマスクとレベルの選択</td> </tr> <tr> <td>CSIMA0</td> <td>CSIA0の動作を制御</td> </tr> <tr> <td>CSIS0</td> <td>入力クロックの選択と動作ステータス</td> </tr> <tr> <td>ADTP0</td> <td>自動データ転送時の転送バイト数</td> </tr> <tr> <td>ADTI0</td> <td>自動データ転送時の1バイト間のインターバル時間</td> </tr> </table>	CSIAIC0	INTCSIA0割り込みのマスクとレベルの選択	CSIMA0	CSIA0の動作を制御	CSIS0	入力クロックの選択と動作ステータス	ADTP0	自動データ転送時の転送バイト数	ADTI0	自動データ転送時の1バイト間のインターバル時間
CSIAIC0	INTCSIA0割り込みのマスクとレベルの選択										
CSIMA0	CSIA0の動作を制御										
CSIS0	入力クロックの選択と動作ステータス										
ADTP0	自動データ転送時の転送バイト数										
ADTI0	自動データ転送時の1バイト間のインターバル時間										
【call関数】	main メイン関数										
【変数】	無し										
【割り込み】	acsi_rms0_int										
【割り込み要因】	INTCSIA0										
【ファイル名】	acsi_rms0¥acsi_rms0.c , acsi_rms0¥MAIN.C										
【注意事項】	・SIA0、SCKA0端子の設定は acsi_rms0_ini関数で行っています。										

【関 数 名】	acsi_rms0_st
【引 数】	無し
【処 理 内 容】	acsi_rms0の起動関数です。
【起 動 方 法】	acsi_rms0関数の後にコールしてください。
【使 用 S F R】	CSIAE0 CSIA0動作許可 ATSTA0 自動転送開始
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_rms0¥acsi_rms0.c , acsi_rms0¥MAIN.C
【注 意 事 項】	無し

【関 数 名】	ini_buf_acsi0
【引 数】	無し
【処 理 内 容】	バッファRAMの初期化を行ないます。
【起 動 方 法】	acsi_rms0関数の後にコールしてください。
【使 用 S F R】	ADTP0 自動データ転送時の転送バイト数
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_rms0¥acsi_rms0.c
【注 意 事 項】	無し

割り込み関数

【関 数 名】	acsi_rms0_int
【概 要】	mul_rec_ADTP0で指定したデータ転送確認後受信動作を停止します。
【要 因】	INTCSIA0 CSIA0の転送完了
【 call 関数】	無し
【変 数】	無し
【フ ァ イ ル 名】	acsi_rms0¥acsi_rms0.c
【注 意 事 項】	無し



シリアル・インターフェース CSIA 自動モード送受信 (CSIA_n) KF1 / KG1 : n = 0-1

KJ1 : n = 0-2

【機能】	CSIMAOレジスタのCSIAEOビット= 1、ATEOビット = 1の時 CSITOレジスタのATSTOビット = 1により、32バイトバッファRAMを使用した自動送受信を行います。	
【関数名】	acsi_dmm0	
【引数】	unsigned char mul_CSIS0 unsigned char mul_BRGCAO unsigned char mul_ADTP0 unsigned char mul_ADTIO unsigned char mul_CSIMAO	fscka(入力クロック) クロックの分周 転送バイト数 インターバル間隔 転送データのフォーマット
【処理内容】	CSIA自動モード送受信動作の設定を行います。設定は以下の通りです。 カウントクロック：fxx/16 データ長：32バイト 転送方向：MSB	
【起動方法】	<ul style="list-style-type: none"> ・初期設定でコールして下さい。 ・コールする前にSIA0、SOA0、SCKA0端子の設定を行ってください。 ・acsi_dmm0_st関数のコールにより通信を開始して下さい。 	
【使用SFR】	CSIAIC0 CSIMAO CSIS0 ADTP0 ADTIO BRGCAO	INTCSIA0割り込みのマスクとレベルの選択 CSIA0の動作を制御 入力クロックの選択と動作ステータス 自動データ転送時の転送バイト数 自動データ転送時の1バイト間のインターバル時間 シリアル転送スピードを制御
【call関数】	main ini_buf_acsi0	メイン関数 バッファRAMを初期化
【変数】	無し	
【ファイル名】	acsi_dmm0¥acsi_dmm0.c , acsi_dmm0¥MAIN.C	
【注意事項】	<ul style="list-style-type: none"> ・SIA0、SOA0、SCKA0端子の設定はacsi_dmm0_ini関数で行っています。 ・通信を継続する為にメインループ内にacsi_dmm0_mloop関数をコールして下さい。 	

シリアル・インターフェース CSIA 自動モード送受信 (CSIA_n) KF1 / KG1 : n = 0-1
 KJ1 : n = 0-2

【関 数 名】	acsi_dmm0_st	
【引 数】	無し	
【処 理 内 容】	acsi_dmm0の起動関数です。	
【起 動 方 法】	acsi_dmm0関数の後にコールしてください。	
【使 用 S F R】	CSIAE0 ATSTAO	CSIA0動作許可 自動転送開始
【 call 関数】	無し	
【変 数】	無し	
【フ ァ イ ル 名】	acsi_dmm0¥acsi_dmm0.c , acsi_dmm0¥MAIN.C	
【注 意 事 項】	無し	

【関 数 名】	ini_buf_acsi0	
【引 数】	unsigned char data	バッファRAMの初期値
【処 理 内 容】	バッファRAMの初期化を行ないます。	
【起 動 方 法】	acsi_rms0関数の後にコールしてください。	
【使 用 S F R】	ADTP0	自動データ転送時の転送バイト数
【 call 関数】	無し	
【変 数】	無し	
【フ ァ イ ル 名】	acsi_dmm0¥acsi_dmm0.c , acsi_dmm0¥MAIN.C	
【注 意 事 項】	無し	

【関 数 名】	acsi_dmm0_mloop	
【概 要】	mul_ADTP0で指定したデータ転送確認後受信動作を停止します。	
【起 動 方 法】	メインループでコールして下さい。	
【 call 関数】	無し	
【変 数】	無し	
【フ ァ イ ル 名】	acsi_dmm0¥acsi_dmm0.c , acsi_dmm0¥MAIN.C	
【注 意 事 項】	無し	

