

お客様各位

---

## カタログ等資料中の旧社名の扱いについて

---

2010年4月1日を以ってNECエレクトロニクス株式会社及び株式会社ルネサステクノロジが合併し、両社の全ての事業が当社に承継されております。従いまして、本資料中には旧社名での表記が残っておりますが、当社の資料として有効ですので、ご理解の程宜しくお願い申し上げます。

ルネサスエレクトロニクス ホームページ (<http://www.renesas.com>)

2010年4月1日  
ルネサスエレクトロニクス株式会社

【発行】ルネサスエレクトロニクス株式会社 (<http://www.renesas.com>)

【問い合わせ先】 <http://japan.renesas.com/inquiry>

## ご注意書き

1. 本資料に記載されている内容は本資料発行時点のものであり、予告なく変更することがあります。当社製品のご購入およびご使用にあたりましては、事前に当社営業窓口で最新の情報をご確認いただきますとともに、当社ホームページなどを通じて公開される情報に常にご注意ください。
2. 本資料に記載された当社製品および技術情報の使用に関連し発生した第三者の特許権、著作権その他の知的財産権の侵害等に関し、当社は、一切その責任を負いません。当社は、本資料に基づき当社または第三者の特許権、著作権その他の知的財産権を何ら許諾するものではありません。
3. 当社製品を改造、改変、複製等しないでください。
4. 本資料に記載された回路、ソフトウェアおよびこれらに関連する情報は、半導体製品の動作例、応用例を説明するものです。お客様の機器の設計において、回路、ソフトウェアおよびこれらに関連する情報を使用する場合には、お客様の責任において行ってください。これらの使用に起因しお客様または第三者に生じた損害に関し、当社は、一切その責任を負いません。
5. 輸出に際しては、「外国為替及び外国貿易法」その他輸出関連法令を遵守し、かかる法令の定めるところにより必要な手続を行ってください。本資料に記載されている当社製品および技術を大量破壊兵器の開発等の目的、軍事利用の目的その他軍事用途の目的で使用しないでください。また、当社製品および技術を国内外の法令および規則により製造・使用・販売を禁止されている機器に使用することができません。
6. 本資料に記載されている情報は、正確を期すため慎重に作成したのですが、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。

# H8/300L SLP シリーズ

## 調歩同期式シリアルデータ同時送受信

### 要旨

調歩同期式シリアル転送機能を使用して、4 バイトの 8 ビットデータの同時送受を行いません。

送信データの通信フォーマットは、データ長を 8 ビット、奇数パリティ、ストップビット長を 1 ビットに設定します。ビットレートは 31250(bit/s)で送信します。4 バイトのデータを送受信すると終了します。

### 動作確認デバイス

H8/38024

### 目次

1. 仕様 .....	2
2. 使用機能説明 .....	3
3. 動作説明 .....	8
4. ソフトウェア説明 .....	9
5. フローチャート .....	13
6. プログラムリスト .....	15

### 1. 仕様

- (1) 図 1 に示すように調歩同期式シリアル転送機能を使用して、4 バイトの 8 ビットデータの同時送受を行います。
- (2) 送信データの通信フォーマットは、データ長が 8 ビット、奇数パリティ、ストップビット長が 1 ビットに設定します。
- (3) ビットレートは 31250(bit/s)で送信します。4 バイトのデータを送受信すると終了します。

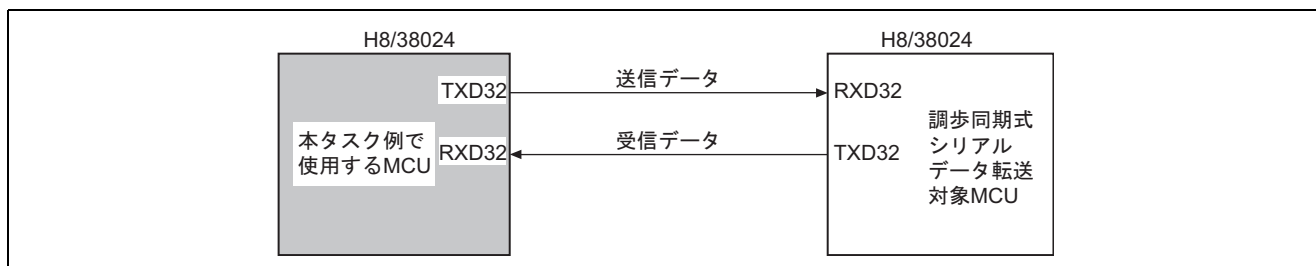
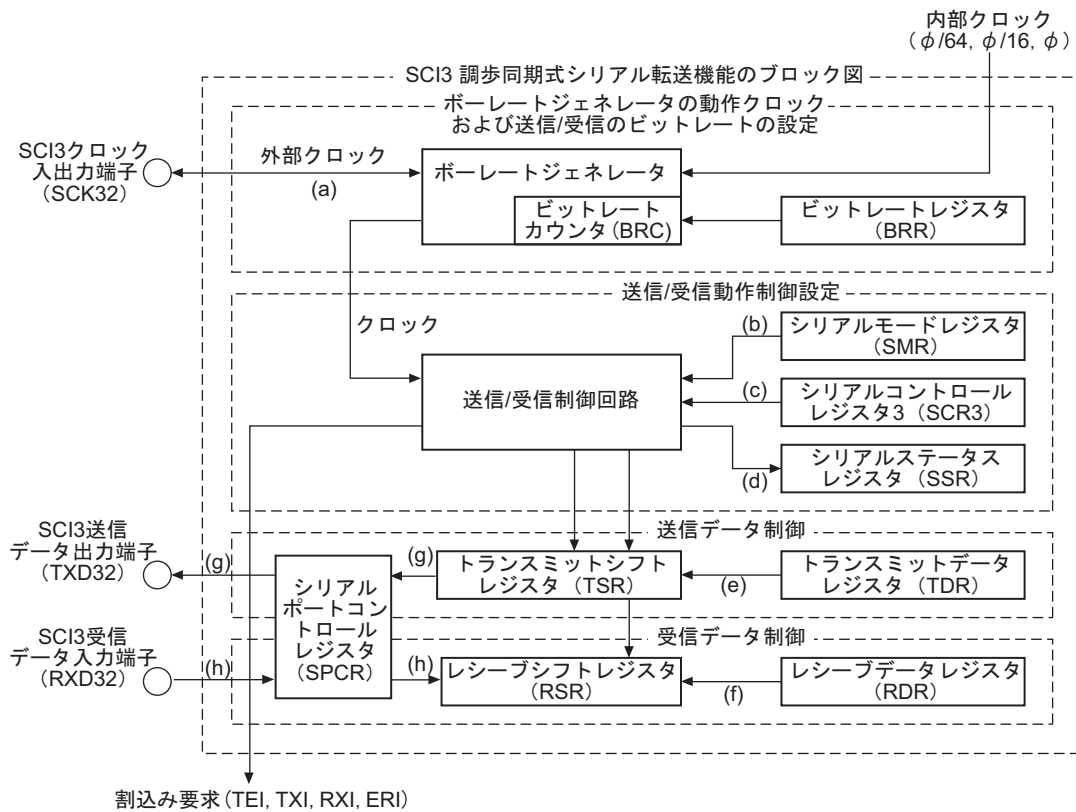


図 1 調歩同期式シリアルデータ同時送受信

## 2. 使用機能説明

- (1) 本タスク例では、シリアルコミュニケーションインタフェース(SCI: Serial Communication Interface)を使用して、調歩同期式のシリアルデータの同時送受信を行ないます。図 2 に調歩同期式シリアルデータ同時送受信のブロック図を示します。以下に調歩同期式シリアルデータ同時送受信のブロック図について説明します。
- 調歩同期モードは、キャラクタ単位で同期をとる調歩同期方式でシリアルデータ通信を行ないます。
  - Universal Asynchronous Receiver/Transmitter(UART)や、Asynchronous Communication Interface Adapter(ACIA)などの標準の調歩同期式通信用 LSI とのシリアルデータ通信ができます。
  - 複数のプロセッサとシリアル通信ができるマルチプロセッサ間通信機能を備えています。
  - 通信フォーマットを 16 種類のフォーマットから選択できます。
  - 独立した送信部と受信部を備えているので、送信と受信を同時に行なうことができます。また、送信部および受信部ともにダブルバッファ構造になっているため、連続送信・連続受信ができます。
  - 内蔵のポーレートジェネレータで任意のビットレートを選択可能です。
  - 送受信クロックソースを内部クロック、または外部クロックから選択可能です。
  - 割込み要因には送信終了、送信データエンプティ、受信データフル、オーバランエラー、フレーミングエラー、パリティエラーの 6 種類の割込み要因があります。
  - レシーブシフトレジスタ(RSR)は、シリアルデータを受信するためのレジスタです。RSR に RXD32 端子から入力されたシリアルデータを、LSB(ビット 0)から受信した順にセットしパラレルデータに変換します。1 バイトのデータを受信すると、データは自動的に RDR へ転送されます。CPU から RSR を直接リード/ライトすることはできません。
  - レシーブデータレジスタ(RDR)は、受信したシリアルデータを格納する 8 ビットのレジスタです。1 バイトのデータの受信が終了すると、受信したデータを RSR から RDR へ転送し、受信動作を完了します。その後、RSR は受信可能となります。RSR と RDR はダブルバッファになっているため連続した受信動作が可能です。RDR は受信専用レジスタなので CPU からライトできません。
  - トランスミットシフトレジスタ(TSR)は、シリアルデータを送信するためのレジスタです。TDR から送信データをいったん TSR に転送し、LSB(ビット 0)から順に TXD32 端子に送出することでシリアルデータ送信を行ないます。1 バイトのデータを送信すると、自動的に TDR から TSR へ次の送信データを転送し、送信を開始します。ただし、TDR にデータが書き込まれていない(TDRE に"1"がセットされている)場合には TDR から TSR へのデータ転送は行ないません。CPU から TSR を直接リード/ライトすることはできません。
  - トランスミットデータレジスタ(TDR)は、送信データを格納する 8 ビットのレジスタです。TSR の“空”を検出すると、TDR に書き込まれた送信データを TSR に転送し、シリアルデータ送信を開始します。TSR のシリアルデータ送信中に、TDR に次の送信データをライトしておくこと、連続送信が可能です。TDR は、常に CPU によるリード/ライトが可能です。
  - シリアルモードレジスタ(SMR)は、シリアルデータ通信フォーマットの設定と、ポーレートジェネレータのクロックソースを選択するための 8 ビットのレジスタです。SMR は、常に CPU によるリード/ライトが可能です。
  - シリアルコントロールレジスタ 3(SCR3)は、送信/受信動作、調歩同期モードでのクロック出力、割込み要求の許可/禁止、および送信/受信クロックソースの選択を行なう 8 ビットのレジスタです。SCR3 は、常に CPU によるリード/ライトが可能です。



割り込み要求 (TEI, TXI, RXI, ERI)

【注】

- (a) SMR で選択されるポータレートジェネレータの動作クロックと合わせて、送信/受信のビットレートを設定します。本タスク例では、送信のビットレートを 31250(bit/s)に設定しています。
- (b) シリアルデータ通信フォーマットの設定と、ポータレートジェネレータのクロックソースを選択します。本タスク例でシリアルデータ通信フォーマットは、動作モードを調歩同期式に、データ長を 8 ビットに、パリティビットあり、パリティモードを奇数パリティに、ストップビット長を 1 ビットに、内蔵ポータレートジェネレータのクロックソースを クロックに設定しています。
- (c) 送信/受信動作、調歩同期式モードでのクロック出力、割り込み要求の許可/禁止を選択します。本タスク例では、調歩同期式モードでのクロック出力は、コミュニケーションモードを調歩同期式に、クロックソースを内部クロックに、SCK32 端子機能を出力クロックに設定しています。また、割り込み要求の許可/禁止は送信データエンプティ割り込み要求を禁止、受信データフル割り込み要求を禁止、受信エラー割り込み要求を禁止に設定しています。
- (d) ステータスフラグ(トランスミットデータレジスタエンプティ、レシーブデータレジスタフル、オーバランエラー、フレーミングエラー、パリティエラー、トランスミットエンド)により SCI3 の動作状態を示します。
- (e) TSR の"空"を検出することにより、TDR に書き込まれた送信データを TSR に転送します。
- (f) 1 バイトのデータの受信が終了すると、受信したデータを RSR から RDR へ転送します。
- (g) 送信データを送信します。
- (h) 受信データを受信します。

図 2 調歩同期式シリアルデータ同時送受信のブロック図

- シリアルステータスレジスタ(SSR)は、SCI3 の動作状態を示すステータスフラグと、マルチプロセッサビットを内蔵した 8 ビットのレジスタです。SSR は常に CPU からリード/ライトできます。ただし、TDRE, RDRF, OER, PER, FER へ "1" をライトすることはできません。また、これらに "0" をライトしてクリアするためには、あらかじめ "1" をリードしておく必要があります。また、TEND および MPBR はリード専用であり、ライトすることはできません。
- ビットレートレジスタ(BRR)は、SMR の CKS1, CKS0 で選択されるボーレートジェネレータの動作クロックとあわせて、送信/受信のビットレートを設定する 8 ビットのレジスタです。BRR は常に CPU によるリード/ライトが可能です。
- 表 1 に、調歩同期式モードの BRR の設定例を示します。表 1 はアクティブモードで、OSC が 10MHz のときの値を示しています。

表 1 ビットレートに対する BRR の設定例(調歩同期式モード)

R ビットレート (bit/s)	110	150	200	250	1200	2400	31250
n	2	2	2	2	0	0	0
N	88	64	48	38	129	64	4
誤差(%)	-0.25	0.16	-0.35	0.16	0.16	0.16	0.00

- 【注】 1. 誤差は、1%以内となるように設定します。  
2. BRR の設定値は以下の計算式で求められます。

$$N = \frac{OSC}{64 \times 2^{2n} \times B} \times 10^6 - 1$$

B: ビットレート(bit/s)

N: ボーレートジェネレータの BRR の設定値(0 ≤ N ≤ 255)

OSC: OSC の値(MHz)= 10MHz または サブクロック w の値 = 32.768kHz

n: SMR の CKS1, CKS0 の設定値(0 ≤ n ≤ 3) n とクロックの関係は表 2 を参照

表 2 n とクロックの関係

n	クロック	SMR の設定値	
		CKS1	CKS0
0		0	0
1	w/4, w	0	1
2	/16	1	0
3	/64	1	1

3. 表 1 の誤差は、以下計算式で求めた値の小数点第 3 位を四捨五入したものです。

$$Error (\%) = \left\{ \frac{\phi \times 10^6}{(N+1) \times B \times 64 \times 2^{2n-1}} - 1 \right\} \times 100$$

4. OSC が 10MHz のときの最大ビットレート(調歩同期式モード)は、31250(bit/s)になります。ただし、設定値は、n=0, N=4 のときです。

- 調歩同期モードは、通信開始を意味するスタートビットと通信終了を意味するストップビットとをデータに付加したキャラクタを送信/受信し、1キャラクタ単位で同期をとりながらシリアル通信を行なうモードです。
- SCI3 内部では、送信部と受信部は独立しているため、全二重通信を行なうことができます。また、送信部と受信部がともにダブルバッファ構造になっているため、送信中にデータのライト、受信中にデータのリードができ、連続送信/受信が可能です。
- 図3に調歩同期式通信のデータフォーマットを示します。調歩同期式通信では、通信回線は通常マーク状態("High"レベル)に保たれています。SCI3では通信回線を監視し、スペース("Low"レベル)になったところをスタートビットとみなしてシリアル通信を開始します。
- 通信データの1キャラクタはスタートビット("Low"レベル)から始まり、送信/受信データ(LSBファースト:最下位ビットから)、パリティビット("High"または"Low"レベル)、最後にストップビット("High"レベル)の順で構成されます。
- 調歩同期モードでは、受信時にスタートビットの立ち下がりエッジで同期化を行いません。また、データを1ビット期間の16倍の周波数のクロックの8番目でサンプリングするので、各ビットの中央で通信データを取り込みます。

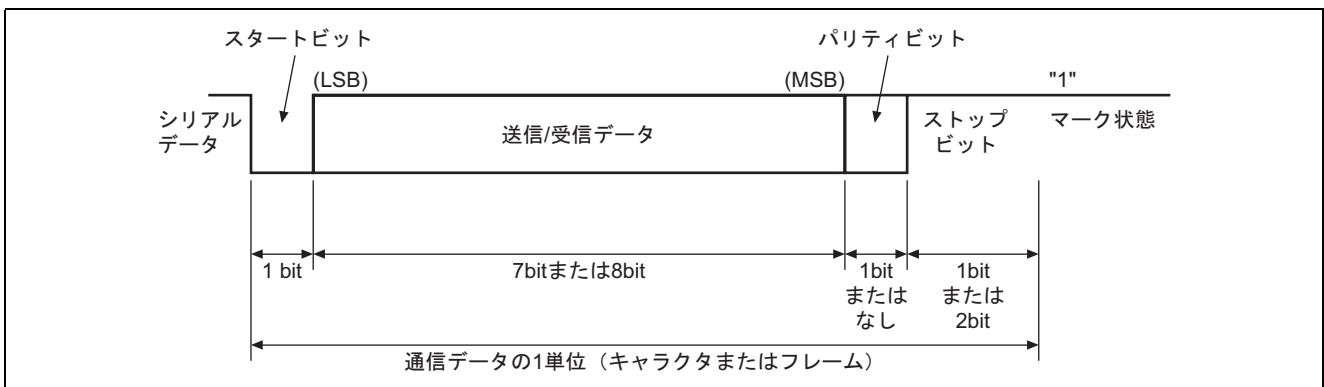


図3 調歩同期式通信のデータフォーマット

- SCI3 クロック(SCK32)は、SCI3 のクロック入出力端子です。
- SCI3 レシーブデータ入力(RXD32)は、SCI3 の受信データ入力端子です。
- SCI3 トランスミットデータ出力(TXD32)は、SCI3 の送信データ出力端子です。
- SCI3 の割り込み要因には、送信終了、送信データエンプティ、受信データフルおよび3種類の受信エラー(オーバランエラー、フレーミングエラー、パリティエラー)の計6種類があり、共通のベクタアドレスが割り付けられています。
- 各割り込み要求は、SCR3 の TIE, RIE で許可/禁止できます。
- SSR の TDRE が"1"にセットされると TXI が発生します。SSR の TEND が"1"にセットされると、TEI が発生します。この2つの割り込みは送信時に発生します。
- SSR の TDRE は初期値が"1"になっています。したがって送信データを TDR へ転送する前に SCR3 の TIE を"1"にセットして送信データエンプティ割り込み要求(TXI)を許可すると、送信データが準備されていなくても TXI が発生します。
- SSR の TEND は初期値が"1"になっています。したがって、送信データを TDR へ転送する前に SCR3 の TEIE を"1"にセットして送信終了割り込み要求(TEI)を許可すると、送信データが送信されていなくても TEI が発生します。
- 送信データを TDR へ転送する処理を割り込み処理ルーチンの中で行なうようにすることで、これらの割り込みを有効に利用できます。また、これらの割り込み要求(TXI, TEI)の発生を防ぐためには、送信データを TDR へ転送した後に、これらの割り込み要求に対応する許可ビット(TIE, TEIE)を"1"にセットします。
- SSR の RDRF が"1"にセットされると RXI が発生します。OER, PER, FER のいずれかが"1"にセットされると ERI が発生します。この2つの割り込み要求は受信時に発生します。



(2)表 3 に本タスク例の機能割付けを示します。表 3 に示すように機能を割り付け、調歩同期式シリアルデータ同時送受信を行ないます。

表 3 機能割付け

機能	機能割付け
TSR	シリアルデータを送信するためのレジスタ
TDR	送信データを格納するレジスタ
RSR	シリアルデータを受信するためのレジスタ
RDR	受信データを格納するレジスタ
SMR	シリアルデータ通信フォーマット, ポーレートジェネレータのクロックソースの設定
SSR	SCI3 の動作状態を示すステータスフラグ
BRR	送信/受信のビットレートを設定
SCR3	送受信動作の許可, TXD32 出力端子設定, RXD32 入力端子設定
TXD32	SCI3 の送信データ出力端子
RXD32	SCI3 の受信データ入力端子
SPCR	TXD32 出力端子設定

3. 動作説明

(1) 図4に動作説明を示します。図4に示すようなハードウェア処理, およびソフトウェア処理により調歩同期式シリアルデータ同時送受信を行ないます。

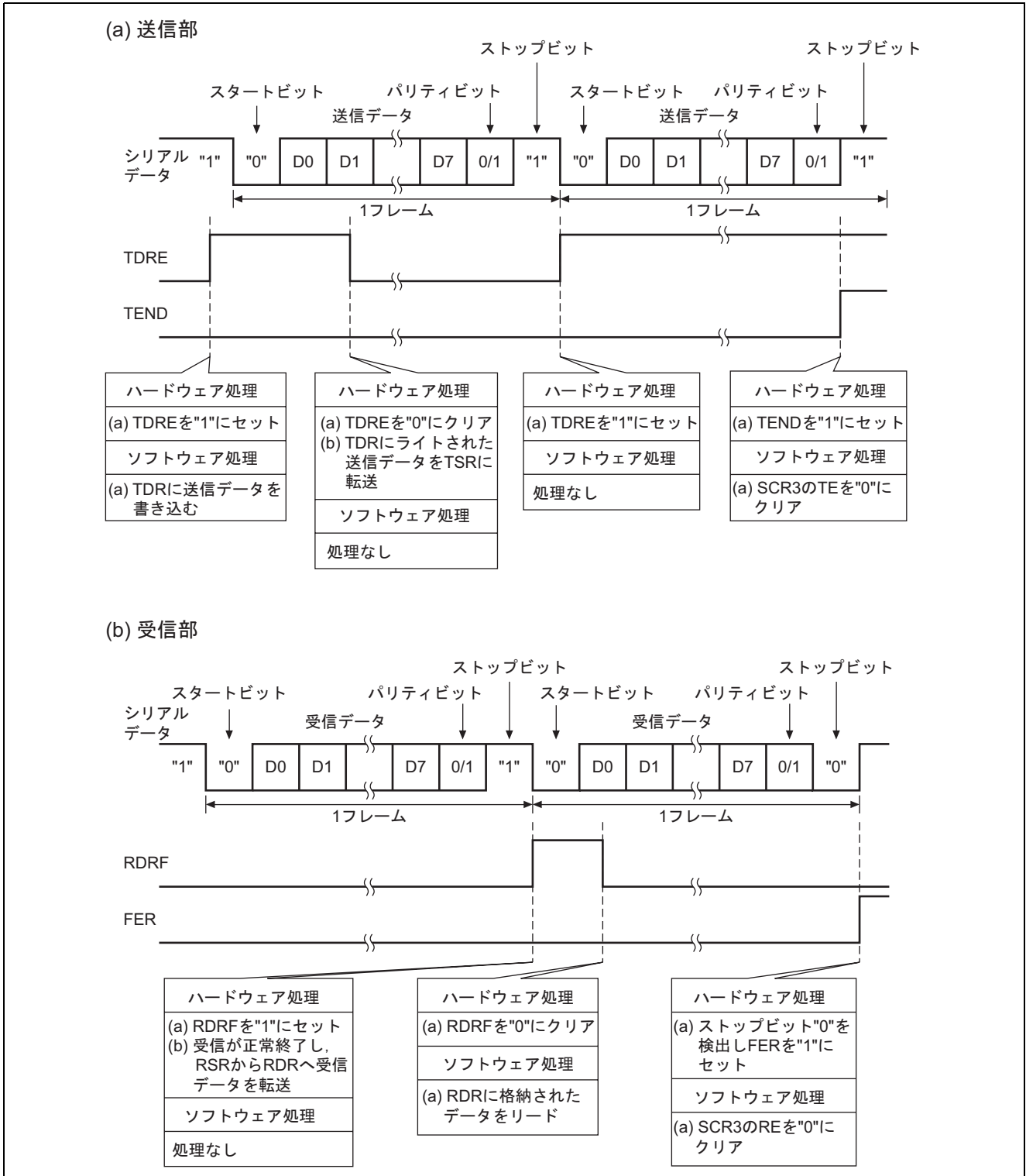


図4 調歩同期式シリアルデータ同時送受信の動作説明

#### 4. ソフトウェア説明

##### (1) モジュール説明

本タスク例のモジュールを表 4 に示します。

表 4 モジュール説明

モジュール名	ラベル名	機能
メインルーチン	main	調歩同期式シリアルデータ送受信の設定，受信エラーが発生した場合には受信エラー処理サブルーチンへ分岐，4 バイトのデータを送受信すると終了
受信エラー処理	er_sub	OER，FER，PER のどのエラーかを判定し，所定のエラー処理を行なう

##### (2) 引数の説明

本タスク例の引数を表 5 に示します。

表 5 引数の説明

引数名	機能	使用モジュール名	データ長	入出力
STD[0] ~ STD[3]	調歩同期式シリアル送信データ	メインルーチン	1 バイト	入力
SRD[0] ~ SRD[3]	調歩同期式シリアル受信データ	メインルーチン	1 バイト	出力

## (3) 使用内部レジスタ説明

本タスク例の使用内部レジスタを表 6 に示します。

表 6 使用内部レジスタ説明

レジスタ名		機能	アドレス	設定値
SMR	COM	シリアルモードレジスタ(コミュニケーションモード) : COM="0"のとき, コミュニケーションモードを調歩同期式モードに設定 : COM="1"のとき, コミュニケーションモードをクロック同期式モードに設定	H'FFA8 ビット 7	0
	CHR	シリアルモードレジスタ(キャラクタレングス) : CHR="0"のとき, 調歩同期式モード時におけるデータ長を 8 ビットデータに設定 : CHR="1"のとき, 調歩同期式モード時におけるデータ長を 7 ビットデータに設定	H'FFA8 ビット 6	0
	PE	シリアルモードレジスタ(パリティイネーブル) : PE="0"のとき, 調歩同期式モードで, 送信時にパリティビットの付加およびチェックを禁止 : PE="1"のとき, 調歩同期式モードで, 送信時にパリティビットの付加およびチェックを許可	H'FFA8 ビット 5	1
	PM	シリアルモードレジスタ(パリティモード) : PM="0"のとき, パリティの付加やチェックを偶数パリティに設定 : PM="1"のとき, パリティの付加やチェックを奇数パリティに設定	H'FFA8 ビット 4	1
	STOP	シリアルモードレジスタ(ストップビットレングス) : STOP="0"のとき, 調歩同期式モードで, ストップビットの長さを 1 ビットに設定 : STOP="1"のとき, 調歩同期式モードで, ストップビットの長さを 2 ビットに設定	H'FFA8 ビット 3	0
	MP	シリアルモードレジスタ(マルチプロセッサモード) : MP="0"のとき, マルチプロセッサ通信機能を禁止 : MP="1"のとき, マルチプロセッサ通信機能を許可	H'FFA8 ビット 2	0
	CKS1 CKS0	シリアルモードレジスタ(クロックセレクト 1, 0) : CKS1="0", CKS0="0"のとき, 内蔵ポーレートジェネレータのクロックソースを クロックに設定	H'FFA8 ビット 1 ビット 0	CKS1="0" CKS0="0"
BRR		ビットレートレジスタ : BRR=H'04 のとき, SMR の CKS1, CKS0 で選択されるポーレートジェネレータの動作クロックとあわせた送信のビットレートを 31250(bit/s)に設定	H'FFA9	H'04
SCR3	TE	シリアルコントロールレジスタ 3(トランスミットイネーブル) : TE="0"のとき, 送信動作を禁止 : TE="1"のとき, 送信動作を許可	H'FFAA ビット 5	0
	RE	シリアルコントロールレジスタ 3(レシーブイネーブル) : RE="0"のとき, 受信動作を禁止 : RE="1"のとき, 受信動作を許可	H'FFAA ビット 4	0
	CKE1 CKE0	シリアルコントロールレジスタ 3(クロックイネーブル 1, 0) : CKE1="0", CKE0="1"のとき, 調歩同期式モードにおいてクロックソースを内部クロック, SCK32 端子機能をクロック出力に設定	H'FFAA ビット 1 ビット 0	CKE1="0" CKE0="1"

表 6 使用内部レジスタ説明(つづき)

レジスタ名	機能	アドレス	設定値
TDR	トランスミットデータレジスタ : 送信データを格納する 8 ビットのレジスタ	H'FFAB	—
SSR	TDRE シリアルステータスレジスタ (トランスミットデータレジスタエンプティ) : TDRE="0"のとき, TDR にライトされた送信データが TSR に転送されていないことを示す : TDRE="1"のとき, TDR に送信データがライトされていない, または TDR にライトされた送信データが TSR に転送されてい ないことを示す	H'FFAC ビット 7	1
	RDRF シリアルステータスレジスタ(レシーブデータレジスタフル) : RDRF="0"のとき, RDR に受信データが格納されていない : RDRF="1"のとき, RDR に受信データが格納されている	H'FFAC ビット 6	1
	OER シリアルステータスレジスタ(オーバランエラー) : OER="0"のとき, 受信中, または受信を完了したことを示す : OER="1"のとき, 受信時にオーバランエラーが発生したことを 示す	H'FFAC ビット 5	0
	FER シリアルステータスレジスタ(フレーミングエラー) : FER="0"のとき, 受信中, または受信を完了したことを示す : FER="1"のとき, 受信時にフレーミングエラーが発生したこ とを示す	H'FFAC ビット 4	0
	PER シリアルステータスレジスタ(パリティエラー) : PER="0"のとき, 受信中, または受信を完了したことを示す : PER="1"のとき, 受信時にパリティエラーが発生したことを 示す	H'FFAC ビット 3	0
	TEND シリアルステータスレジスタ(トランスミットエンド) : TEND="0"のとき, 送信中であることを示す : TEND="1"のとき, 送信を終了したことを示す	H'FFAC ビット 2	—
RDR	レシーブデータレジスタ : 受信データを格納する 8 ビットのレジスタ	H'FFAD	—
SPCR	SPC32 シリアルポートコントロールレジスタ (P42/TXD32 端子機能切り替え) : SPC32="0"のとき, P42/TXD32 端子を P42 端子機能に設定 : SPC32="1"のとき, P42/TXD32 端子を TXD32 端子機能に設 定	H'FF91 ビット 5	1
	SCINV3 シリアルポートコントロールレジスタ (TXD32 端子出力データ反転切り替え) : SCINV3="0"のとき, TXD32 の出力データを反転しない : SCINV3="1"のとき, TXD32 の出力データを反転する	H'FF91 ビット 3	0
	SCINV2 シリアルポートコントロールレジスタ (RXD32 端子入力データ反転切り替え) : SCINV2="0"のとき, RXD32 の入力データを反転しない : SCINV2="1"のとき, RXD32 の入力データを反転する	H'FF91 ビット 2	0

(4) 使用 RAM 説明

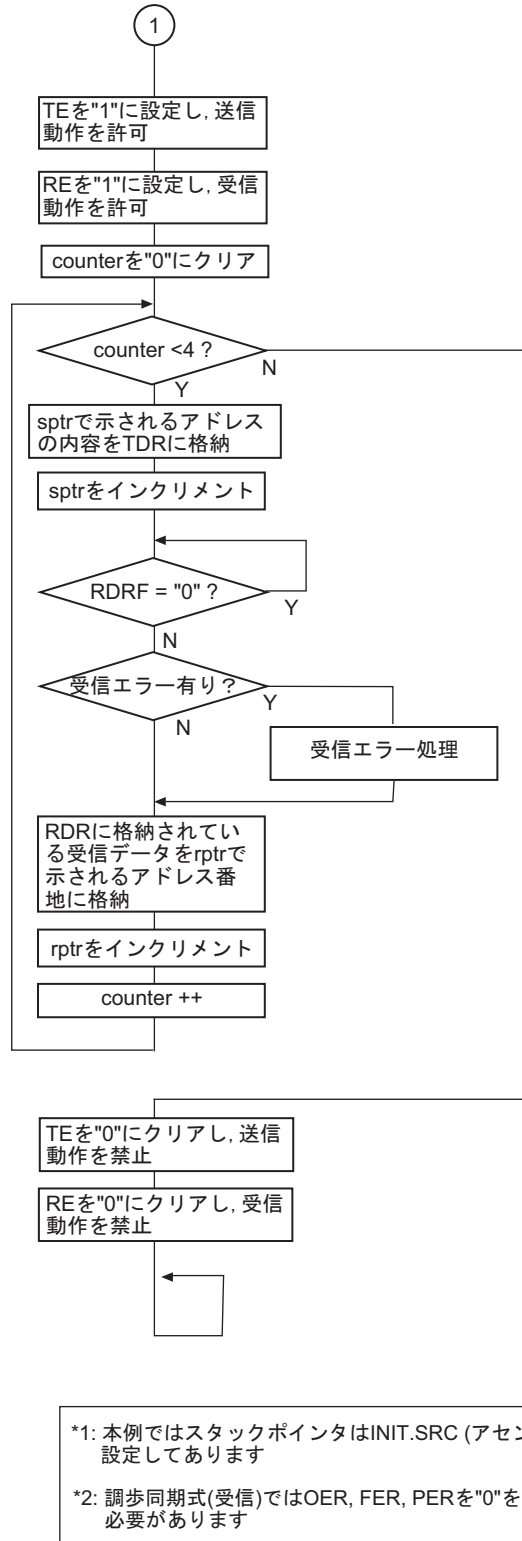
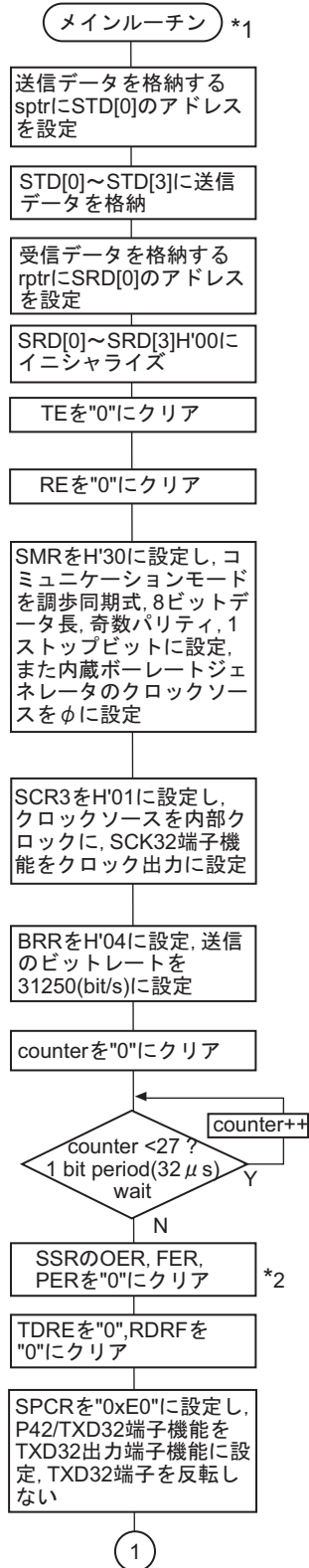
本タスク例の使用 RAM を表 7 に示します。

表 7 使用 RAM 説明

ラベル名	機能	アドレス	使用モジュール名
STD[0]	調歩同期式シリアル送信データの 1 バイト目を格納	H'FB80	メインルーチン
STD[1]	調歩同期式シリアル送信データの 2 バイト目を格納	H'FB81	メインルーチン
STD[2]	調歩同期式シリアル送信データの 3 バイト目を格納	H'FB82	メインルーチン
STD[3]	調歩同期式シリアル送信データの 4 バイト目を格納	H'FB83	メインルーチン
SRD[0]	調歩同期式シリアル受信データの 1 バイト目を格納	H'FB84	メインルーチン
SRD[1]	調歩同期式シリアル受信データの 2 バイト目を格納	H'FB85	メインルーチン
SRD[2]	調歩同期式シリアル受信データの 3 バイト目を格納	H'FB86	メインルーチン
SRD[3]	調歩同期式シリアル受信データの 4 バイト目を格納	H'FB87	メインルーチン

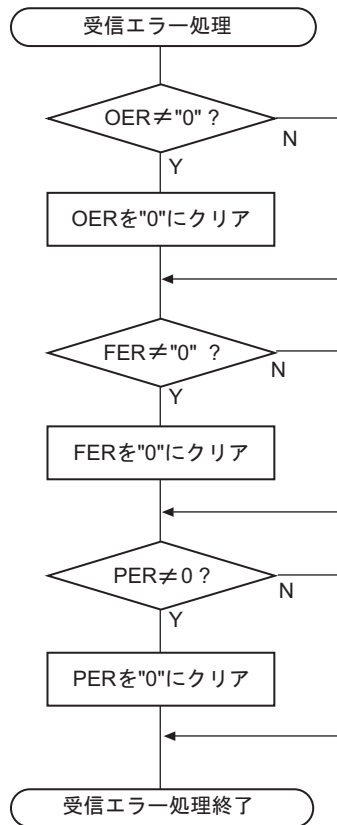
### 5. フローチャート

(1) メインルーチン



\*1: 本例ではスタックポインタはINIT.SRC (アセンブリ言語)で  
設定してあります  
\*2: 調歩同期式(受信)ではOER, FER, PERを"0"を設定しておく  
必要があります

(2) サブルーチン





## 6. プログラムリスト

### 6.1 INIT.SRC(プログラムリスト)

```

        .EXPORT  _INIT
        .IMPORT  _main
;
        .SECTION P, CODE
        _INIT:
        MOV.W   #'FF80,R7
        LDC.B   #B'10000000,CCR
        JMP     @_main
;
        .END

```

```

/*****/
/*                                     */
/* H8/300L Super Low Power Series      */
/*   -H8/38024 Series-                  */
/* Application Note                      */
/*                                     */
/* 'Asynchronous Serial Data Simultaneous */
/*   Transmission and Reception'        */
/*                                     */
/* Function                              */
/* : Serial Communication Interface      */
/*   Asynchronous Serial Interface      */
/*   -Transmitting/Receiving            */
/*                                     */
/* External Clock : 16MHz                */
/* Internal Clock : 16MHz                */
/* Sub Clock      : 32.768kHz            */
/*                                     */
/*****/

#include <machine.h>

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

#define SMR          *(volatile unsigned char *)0xFFA8 /* Serial Mode Register */
#define SMR_BIT      (*(struct BIT *)0xFFA8)          /* Serial Mode Register */
#define COM          SMR_BIT.b7                      /* Communication Mode */
#define CHR          SMR_BIT.b6                      /* Character Length */
#define PE           SMR_BIT.b5                      /* Parity Enable */
#define PM           SMR_BIT.b4                      /* Parity Mode */
#define STOP        SMR_BIT.b3                      /* Stop Bit Length

```

```

#define MP          SMR_BIT.b2          /* Multiprocessor Mode          */
#define CKS1        SMR_BIT.b1          /* Clock Select 1               */
#define CKS0        SMR_BIT.b0          /* Clock Select 0               */
#define BRR         *(volatile unsigned char *)0xFFA9 /* Bit Rate Register           */
#define SCR3        *(volatile unsigned char *)0xFFAA /* Serial Control Register 3    */
#define SCR3_BIT    (*(struct BIT *)0xFFAA) /* Serial Control Register 3    */
#define TIE         SCR3_BIT.b7         /* Transmit Interrupt Enable    */
#define RIE         SCR3_BIT.b6         /* Receive Interrupt Enable     */
#define TE          SCR3_BIT.b5         /* Transmit Enable              */
#define RE          SCR3_BIT.b4         /* Receive Enable               */
#define MPIE        SCR3_BIT.b3         /* Multiprocessor Interrupt Enable */
#define TEIE        SCR3_BIT.b2         /* Transmit End Interrupt Enable */
#define CKE1        SCR3_BIT.b1         /* Clock Enable 1               */
#define CKE0        SCR3_BIT.b0         /* Clock Enable 0               */
#define TDR         *(volatile unsigned char *)0xFFAB /* Transmit Data Register      */
#define SSR         *(volatile unsigned char *)0xFFAC /* Serial Status Register       */
#define SSR_BIT     (*(struct BIT *)0xFFAC) /* Serial Status Register       */
#define TDRE        SSR_BIT.b7         /* Transmit Data Register Empty */
#define RDRF        SSR_BIT.b6         /* Receive Data Register Full   */
#define OER         SSR_BIT.b5         /* Overrun Error                */
#define FER         SSR_BIT.b4         /* Framing Error                */
#define PER         SSR_BIT.b3         /* Parity Error                  */
#define TEND        SSR_BIT.b2         /* Transmit End                  */
#define MPBR        SSR_BIT.b1         /* Multiprocessor Bit Receive    */
#define MPBT        SSR_BIT.b0         /* Multiprocessor Bit Transfer   */
#define SPCR        *(volatile unsigned char *)0xFF91 /* Transmit Data Register      */
#define SPCR_BIT    (*(struct BIT *)0xFF91) /* Port Mode Register 1         */
#define SPC32       SPCR_BIT.b5        /* TXD Output Terminal          */

#define RDR         *(volatile unsigned char *)0xFFAD /* Receive data Register        */

/*****/
/* Function define */
/*****/
extern void INIT ( void ); /* SP Set */
void main ( void );
void er_sub ( void );

/*****/
/* RAM define */
/*****/
unsigned char STD[4];
unsigned char SRD[4];

/*****/
/* Vector Address */
/*****/
#pragma section V1 /* VECTOR SECTOIN SET */
void (*const VEC_TBL1[])(void) = {
/* 0x00 - 0x0f */
INIT /* 00 Reset */
};

#pragma section /* P */
/*****/
/* Main Program */
/*****/

```

```

void main ( void )
{
    unsigned char  stus;
    unsigned char  *sptr,*rptr;
    unsigned char  counter;

    sptr = &STD[0];                /* Initialize Serial Transmitting Data Address */

    STD[0] = 0x00;                  /* Set Serial Transfer Data 0 */
    STD[1] = 0x55;                  /* Set Serial Transfer Data 1 */
    STD[2] = 0xAA;                  /* Set Serial Transfer Data 2 */
    STD[3] = 0xbb;                  /* Set Serial Transfer Data 3 */

    rptr = &SRD[0];                /* Initialize Serial Receiving Data Address */

    SRD[0] = 0x00;                  /* Initialize Serial Receiving Data 0 */
    SRD[1] = 0x00;                  /* Initialize Serial Receiving Data 1 */
    SRD[2] = 0x00;                  /* Initialize Serial Receiving Data 2 */
    SRD[3] = 0x00;                  /* Initialize Serial Receiving Data 3 */

    TE = 0;                         /* Clear Serial Transmitting */
    RE = 0;                         /* Clear Serial Receiving */

    SMR = 0x30;                     /* Initialize Serial Mode Register */
    SCR3 = 0x01;                    /* Initialize Serial Control Register 3 */

    BRR = 4;                        /* Initialize Bit Rate Register */
    for(counter = 0; counter < 27; counter++); /* dummy wait */

    OER = 0;                        /* Clear OER */
    FER = 0;                        /* Clear FER */
    PER = 0;                        /* Clear PER */

    TDRE = 0;                       /* Clear TDRE */
    RDRF = 0;                       /* Clear RDRF */

    SPCR = 0xE0;                   /* Initialize Output Port TXD */

    TE = 1;                         /* Start Serial Transmitting */
    RE = 1;                         /* Start Serial Receiving */

    for(counter = 0; counter < 4; counter++){ /* Serial Transmitting/Receiving Data Counter */
        /*                               4 Loop */
        TDR = *sptr;                 /* Save Serial Transmitting Data */
        sptr++;                      /* Increment Serial Transmitting Data Address */

        while(RDRF == 0){           /* End Serial Receive End ? */
            ;
        }

        if((SSR & 0x38) != 0){      /* Error Flag = 1 ? */
            er_sub();
        }
        else{
            *rptr = RDR;             /* Save Serial Receiving Data */
        }
    }
}

```

```

    }

    rptr++;          /* Increment Serial Receiving Data Address    */
}

TE = 0;            /* Initialize Transmitting Enable      */
RE = 0;            /* Initialize Receiving Enable         */

while(1){
    ;
}

void er_sub(void)
{
    if(OER != 0){
        OER = 0;          /* Clear OER                          */
    }
    if(FER != 0){
        FER = 0;          /* Clear FER                            */
    }
    if(PER != 0){
        PER = 0;          /* Clear PER                            */
    }
}

```

リンクアドレス指定

セクション名	アドレス
CV1	H'0000
P	H'0100
B	H'FB80

改訂記録

Rev.	発行日	改訂内容	
		ページ	ポイント
1.00	2003.12.19	—	初版発行

### 安全設計に関するお願い

1. 弊社は品質、信頼性の向上に努めておりますが、半導体製品は故障が発生したり、誤動作する場合があります。弊社の半導体製品の故障又は誤動作によって結果として、人身事故、火災事故、社会的損害などを生じさせないような安全性を考慮した冗長設計、延焼対策設計、誤動作防止設計などの安全設計に十分ご留意ください。

### 本資料ご利用に際しての留意事項

1. 本資料は、お客様が用途に応じた適切なルネサス テクノロジ製品をご購入いただくための参考資料であり、本資料中に記載の技術情報についてルネサス テクノロジが所有する知的財産権その他の権利の実施、使用を許諾するものではありません。
2. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他応用回路例の使用に起因する損害、第三者所有の権利に対する侵害に関し、ルネサス テクノロジは責任を負いません。
3. 本資料に記載の製品データ、図、表、プログラム、アルゴリズムその他全ての情報は本資料発行時点のものであり、ルネサス テクノロジは、予告なしに、本資料に記載した製品または仕様を変更することがあります。ルネサス テクノロジ半導体製品のご購入に当たりましては、事前にルネサス テクノロジ、ルネサス販売または特約店へ最新の情報をご確認頂きますとともに、ルネサス テクノロジホームページ(<http://www.renesas.com>)などを通じて公開される情報に常にご注意ください。
4. 本資料に記載した情報は、正確を期すため、慎重に制作したものです。万一本資料の記述誤りに起因する損害がお客様に生じた場合には、ルネサス テクノロジはその責任を負いません。
5. 本資料に記載の製品データ、図、表に示す技術的な内容、プログラム及びアルゴリズムを流用する場合は、技術内容、プログラム、アルゴリズム単位で評価するだけでなく、システム全体で十分に評価し、お客様の責任において適用可否を判断してください。ルネサス テクノロジは、適用可否に対する責任を負いません。
6. 本資料に記載された製品は、人命にかかわるような状況の下で使用される機器あるいはシステムに用いられることを目的として設計、製造されたものではありません。本資料に記載の製品を運輸、移動体用、医療用、航空宇宙用、原子力制御用、海底中継用機器あるいはシステムなど、特殊用途へのご利用をご検討の際には、ルネサス テクノロジ、ルネサス販売または特約店へご照会ください。
7. 本資料の転載、複製については、文書によるルネサス テクノロジの事前の承諾が必要です。
8. 本資料に関し詳細についてのお問い合わせ、その他お気づきの点がございましたらルネサス テクノロジ、ルネサス販売または特約店までご照会ください。