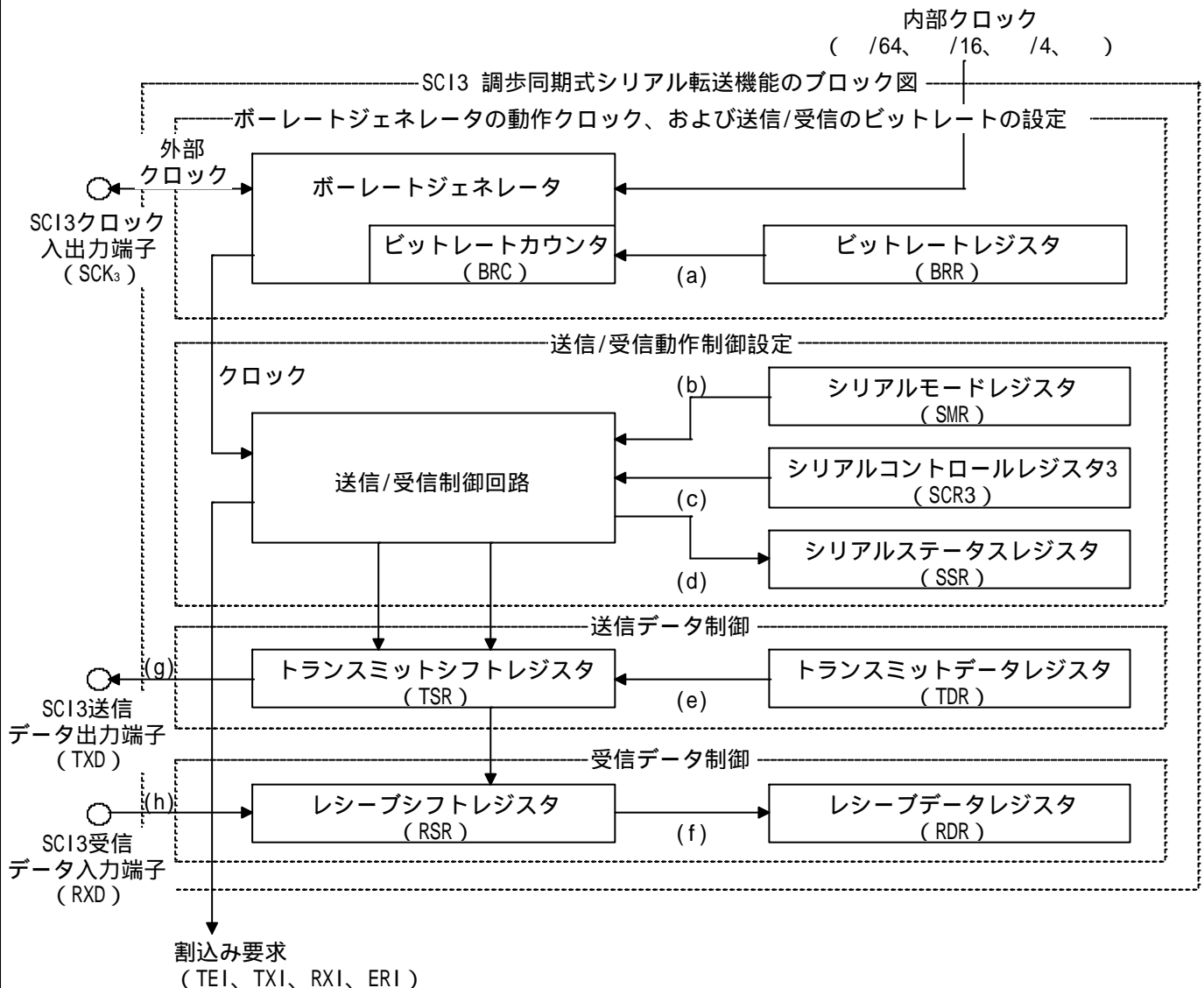


2.14 調歩同期式シリアルデータ送信

調歩同期式シリアルデータ送信	使用機能	SCI3 : 調歩同期式シリアル転送機能
仕様		
<p>(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ビットデータの送信を行います。</p> <p>(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。</p> <p>(3) ビットレートは31250 (bit/s) で送信します。またデータ送信の終了時にブレークを出力します。</p>		
図1 調歩同期式シリアルデータ送信		

使用機能説明		
<p>(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送信を行ないます。図2に調歩同期式シリアルデータ送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送信のブロック図について説明します。</p> <ul style="list-style-type: none"> ・調歩同期式モードは、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12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。 ・内蔵のボ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pty、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RXD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TXD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、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、ボ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、調歩同期式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。 		

使用機能説明



- 【注】 (a) SMRで選択されるポ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、SCK₃端子機能をクロック出力に設定しています。また、割り込み要求の許可/禁止は送信データエンpty割り込みを禁止に、受信データフル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pty、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。
- (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。
- (g) 送信データ。
- (h) 受信データ。

図2 調歩同期式シリアルデータ送信のブロック図

使用機能説明

- ・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16MHz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	300	600	1200	2400	4800	9600	19200	31250	38400
n	3	2	2	1	1	0	0	0	0	0	0
N	70	207	103	207	103	207	103	51	25	15	12
誤差 (%)	0.03	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.16	0.00	0.16

- 【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) = 16MHz

n : SMRのCKS1、CKS2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	/4	0	1
2	/16	1	0
3	/64	1	1

3. 表1に誤差は以下の計算式で求めた値を小数点第3位を四捨五入して表示してあります。

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

4. OSCが16MHzのときの最大ビットレート (調歩同期式モード) は、500000 (bit/s) になります。ただし、設定値は、n=0、N=0のときです。

- ・調歩同期式モードは、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ので、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ので、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。
- ・調歩同期式モードでは、受信時にスタートビットの立ち上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。

使用機能説明

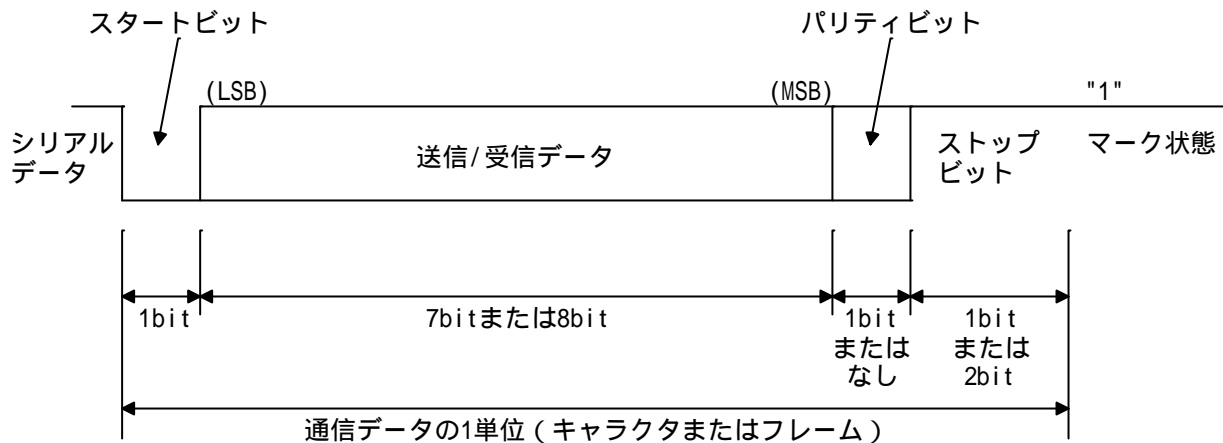


図3 調歩同期式通信のデータフォーマット

- SCI3クロック (SCK₃) は、SCI3のクロック入出力端子です。
- SCI3レシーブデータ入力 (RXD) 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 (TXD) 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送信を行ないます。

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
SMR	シリアルデータ通信フォーマット、ボ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
PMR1	TXD出力端子設定
SCK ₃	SCI3のクロック出力端子
TXD	SCI3の送信データ出力端子

動作原理

(1) 図4に動作原理を示します。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送信を行ないます。

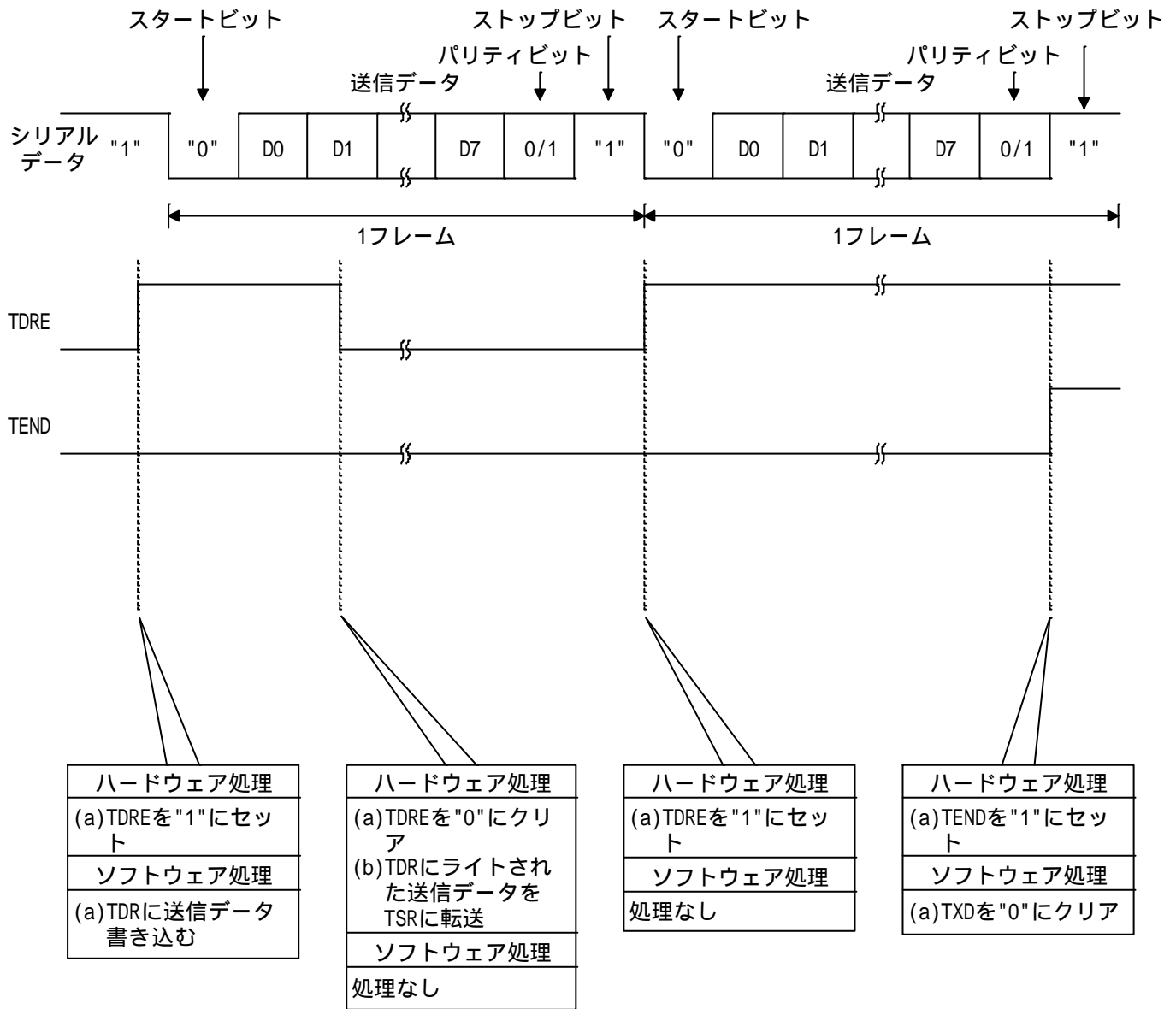


図4 調歩同期式シリアルデータ送信の動作原理

ソフトウェア説明

(1) モジュール説明

表4に本タスク例におけるモジュール説明を示します。

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	転送データの設定、調歩同期式シリアルデータ送信の設定、4バイトのデータを送信したところで終了

(2) 引数の説明

表5に本タスク例で使用する引数を示します。

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
STD0 ~ STD3	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

(3) 使用内部レジスタ説明

表6に本タスク例における使用内部レジスタ説明を示します。

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	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	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="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="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のストップビットの長さを1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	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'0F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'0F	

ソフトウェア説明

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	
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 (TXD端子はトランスミットデータ端子)	H'FFAA ビット5	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、SCK ₃ 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"
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エンpty) : 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 : TSRE="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に転送されたことを示す	H'FFAC ビット7	1
	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	1
PMR1	PMR11	ポートモードレジスタ1 (P2 ₂ /TXD端子機能切り替え) : PMR11="1"のとき、P2 ₂ /TXD端子をTXD端子機能に設定	H'FFE0 ビット1	1

(4) 使用RAM説明

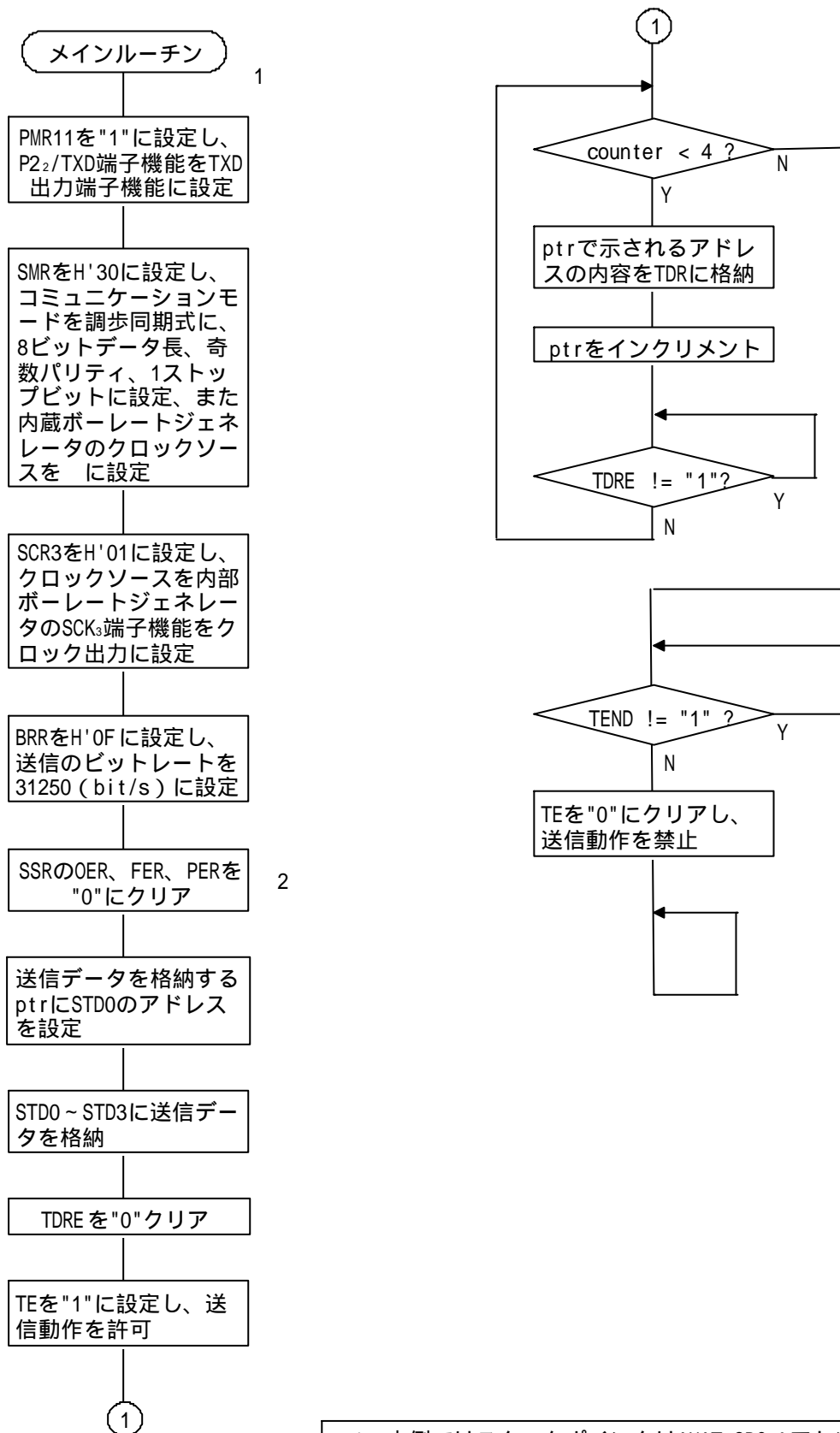
表7に本タスク例における使用RAM説明を示します。

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
STD0	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
STD1	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
STD2	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
STD3	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
counter	調歩同期式シリアル送信動作を4カウントする8ビットカウンタ	H'FB84	メインルーチン

フローチャート

(a) 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 (送信) ではOER、FER、PERを"0"クリアしておく必要がある。

プログラムリスト

INIT.SRC (プログラムリスト)

```

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

/*****/
/*                                     */
/* H8/300H Tiny Series -H8/3664-      */
/* Application Note                    */
/*                                     */
/* 'Asynchronous Serial Data Transmission' */
/*                                     */
/* Function                             */
/* : Serial Communication Interface     */
/*   Asynchronous Serial Interface      */
/*   -Transmitting                      */
/*                                     */
/* 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 */

```

プログラムリスト

```

#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 PMR1_BIT (*(struct BIT *)0xFFE0) /* Port Mode Register 1 */
#define PMR11    PMR1_BIT.b1 /* Port Mode Register 1 Bit1 */
#define PCR2_BIT (*(struct BIT *)0xFFE5) /* Serial Control Register 2 */
#define PCR22    PCR2_BIT.b2 /* Port Control Register 2 bit2 */
#define PCR21    PCR2_BIT.b1 /* Port Control Register 2 bit1 */
#define PCR20    PCR2_BIT.b0 /* Port Control Register 2 bit0 */

/*****
/* 関数定義 */
*****/
extern void INIT( void ); /* SP Set */
void main ( void );

/*****
/* RAM Allocation */
*****/
unsigned char STD0;
unsigned char STD1;
unsigned char STD2;
unsigned char STD3;
unsigned char counter;

/*****
/* 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 *ptr;

    PMR11 = 1; /* Initialize Output Port TXD */
    SMR = 0x30; /* Initialize Serial Mode Register */
    SCR3 = 0x01; /* Initialize Serial Control Register 3 */
    BRR = 0x0F; /* Initialize Bit Rate Register */

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

    ptr = &STD0; /* Initialize Serial Transmitting Data Address */

    STD0 = 0x22; /* Set Serial Transfer Data 0 */
    STD1 = 0x44; /* Set Serial Transfer Data 1 */
    STD2 = 0x66; /* Set Serial Transfer Data 2 */
    STD3 = 0x88; /* Set Serial Transfer Data 3 */

```

プログラムリスト

```

TDRE = 0;                /* Clear TDRE                */
TE = 1;                  /* Start Serial Transmitting */
for(counter = 0 ; counter < 4 ; counter++){ /* Serial Transmitting Data Counter 4 Loop */
    TDR = *ptr;          /* Save Serial Transmitting Data */
    ptr++;               /* Increment Serial Transmitting Data Address */
    while(TDRE != 1){   /* End Serial Transmitting */
        ;
    }
}
while(TEND != 1){       /* End Serial Transmitting */
    ;
}
TE = 0;                 /* Initialize Transmitting Enable */

while(1) {
    ;
}
}

```

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