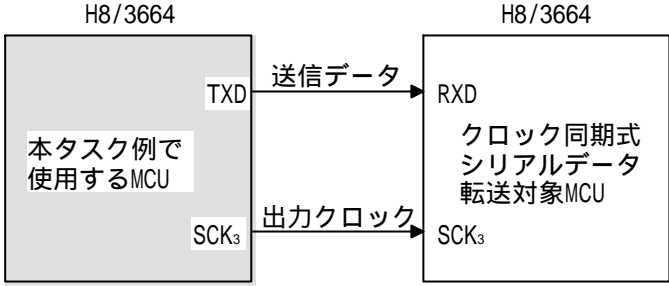


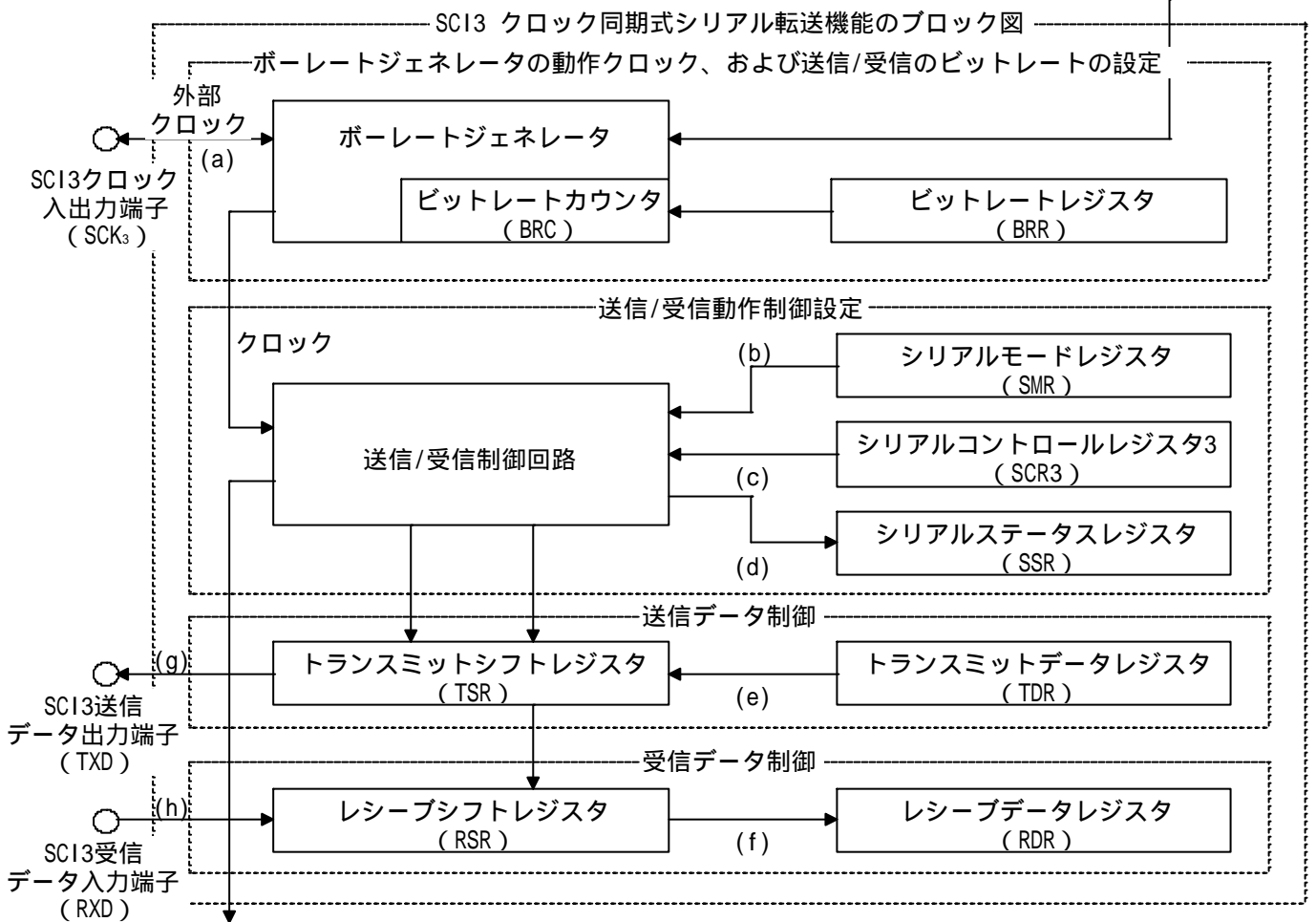
2.11 クロック同期式シリアルデータ送信

クロック同期式シリアルデータ送信	使用機能	SCI3 : クロック同期式シリアル転送機能
<p>仕様</p> <p>(1) 図1に示すようにクロック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) 転送クロックは、内部クロックを使用し4μsの転送クロック周期でデータを転送します。</p> <p>(3) 転送するデータのデータ長は8ビットで、データの最下位ビットから送信するLSBファースト方式による送信を行います。</p> <div style="text-align: center;">  </div> <p>図1 クロック同期式シリアルデータ送信</p>		

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、クロック同期式のシリアルデータの送信を行います。図2にクロック同期式シリアルデータ送信のブロック図を示します。以下にクロック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"> ・システムクロック () は、16MHzのOSCクロックで、CPUおよび周辺機能を動作させるための基準クロックです。 ・クロック同期モードではデータ長は8ビットになります。 ・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、内蔵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。 ・シリアルコントロールレジスタ3 (SCR3) は、送信/受信動作および送信/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。 ・シリアルステータスレジスタ (SSR) は、SCI3のステータスフラグと送受信マルチプロセッサビットで構成されています。TDRE、RDRF、OER、PER、FERはクリアのみ可能です。 ・転送クロックは、8種類の内部クロックと外部クロックから選択できます。内部クロックを選択した場合は、SCK₃端子は出力端子となります。クロック連続出力モードに設定すると選択したクロックをSCK₃端子から連続して出力します。外部クロックを選択した場合は、SCK₃端子はクロック入力端子となります。 ・本タスク例では、転送クロックソースを内蔵ポーレートジェネレータの /64分周クロックにし、転送クロック周期を4μsに設定しています。 ・SCI3の転送フォーマットは8ビットのデータを選択可能です。データの最下位ビットから送受信されるLSBファースト方式による転送を行います。送信データは、転送クロックの立ち上がりから次の立ち上がりまで出力されます。また、受信データは転送クロックの立ち上がりで取り込まれます。 ・本タスク例では、動作モードを8ビットモードに設定し、8ビットのデータ送信を行います。 ・SCI3クロック (SCK₃) は、SCI₃のクロック入出力端子です。 ・SCI3トランスミットデータ出力 (TXD) は、SCI₃の送信データの出力端子です。
--------	--

使用機能説明

図2 クロック同期式シリアルデータ送信のブロック図 (内部クロック /64、 /16、 /4、)



割り込み要求 (TEI、TXI、RXI、ERI)

- 【注】 (a) SMRで選択されたボーレートジェネレータの動作クロック (/64分周) を出力する。
 (b) 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します。
 (c) 送信/受信動作、クロック同期式モードでのクロック出力端子の選択をします。
 (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ブティ) 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) 送信データを送信する。
 (h) 受信データを受信する。

(2) 表1に本タスク例の機能割付け示します。表1に示すように機能を割り付け、クロック同期式シリアルデータ送信を行ないます。

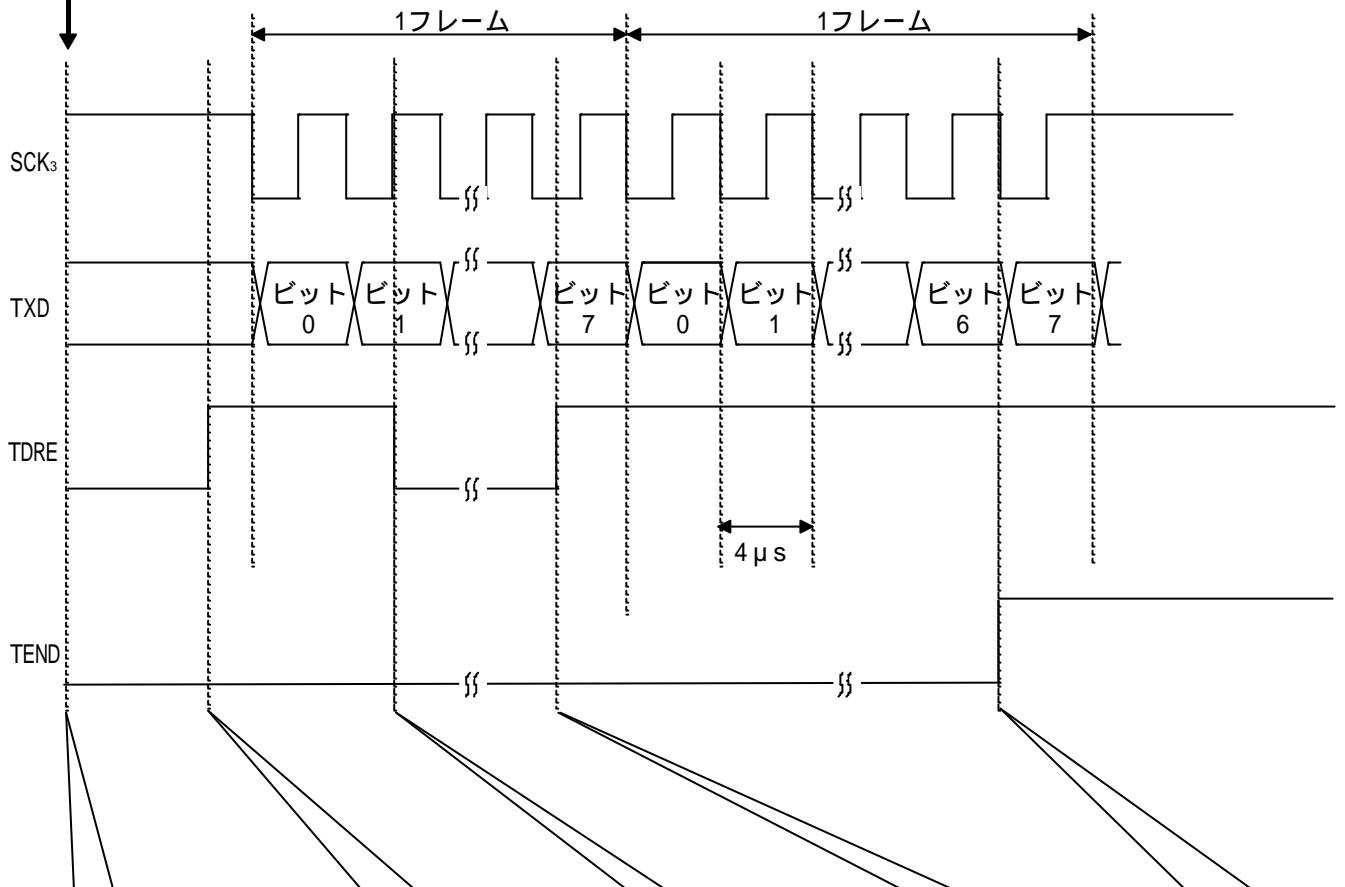
表1 機能割付け

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
SCR3	送信動作、SCK ₃ の端子機能をクロック出力端子設定
SCK ₃	SCI3のクロック出力端子
TXD	SCI3の送信データ出力端子
PMR1	TXD出力端子設定

動作原理

(1) 図3に動作原理を示します。図3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クロック同期式シリアルデータ送信を行います。

リセット直後



ハードウェア処理 処理なし	ハードウェア処理 (a) TDREを"1"に設定	ハードウェア処理 (a) TDREを"0"にクリア (b) TDRにライトされた送信データをTSRに転送	ハードウェア処理 (a) TDREを"1"に設定	ハードウェア処理 (a) TENDを"1"に設定
ソフトウェア処理 初期設定 (a) TXD出力端子、SCK ₃ 入出力端子の設定 (b) 8ビットクロック同期式シリアル転送機能の設定 (c) 送信データの設定	ソフトウェア処理 (a) 1バイト目の送信データをTDRに設定	ソフトウェア処理 (a) シリアル送信カウンタをインクリメント	ソフトウェア処理 (a) 2バイト目の送信データをTDRに設定	ソフトウェア処理 (a) TEを"0"にクリアし、送信動作終了

図3 クロック同期式シリアル送信の動作原理

クロック同期式シリアルデータ送信	使用機能	SCI3 : クロック同期式シリアル転送機能
------------------	------	------------------------

ソフトウェア説明

(1) モジュール説明

表2に本タスク例におけるモジュール説明を示します。

表2 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、クロック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3に本タスク例で使用する引数を示します。

表3 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	クロック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4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4 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="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	1
	MP	シリアルモードレジスタ (マルチプロセッサモード) : クロック同期式モードではこのビットは"0"に設定する。	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="1"、CKS0="1" のとき、内蔵ボーレートジェネレータのクロックソースを /64分周クロックに設定	H'FFA8 ビット1 ビット0	CKS1="1" CKS0="1"
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="1" のとき、送信動作が可能になります。	H'FFAA ビット5	1
	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="0" のとき、クロック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="0"
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 : 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 (P ₂ /TXD端子機能切り替え) : PMR11="1" のとき、P 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

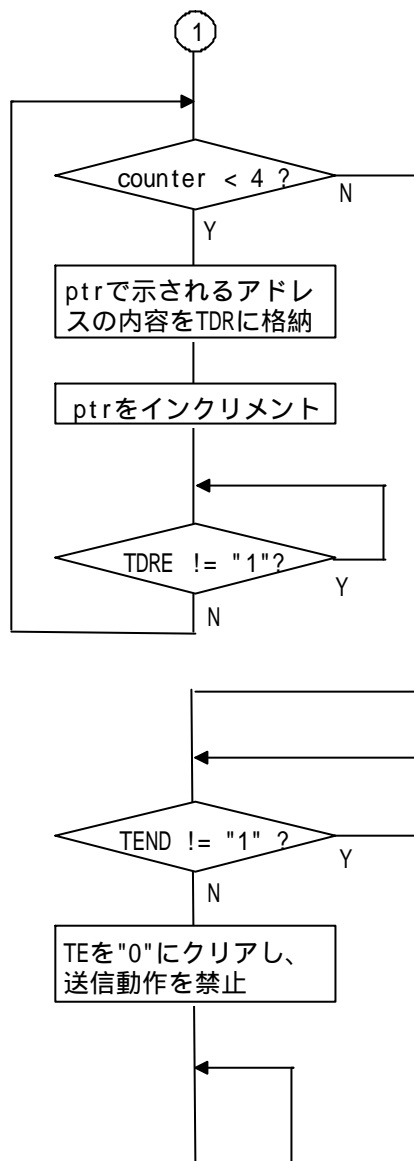
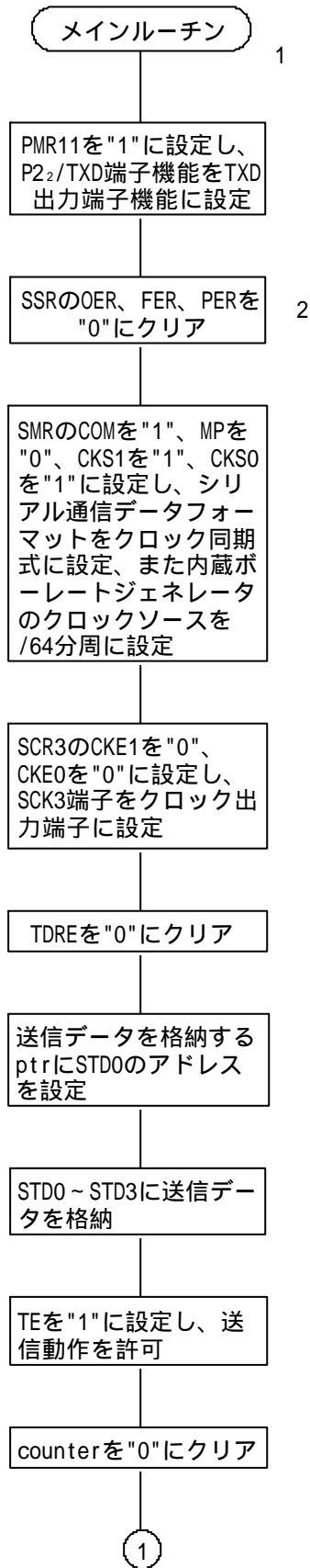
表5に本タスク例における使用RAM説明を示します。

表5 使用RAM説明

ラベル名	機能	アドレス	使用モジュール名
STD0	クロック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	クロック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	クロック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	クロック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	クロック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: クロック同期式 (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
/*
/* 'Synchronous Serial Data Transmission'
/*
/* Function
/* : Serial Communication Interface
/* Synchronous 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_BIT (*(struct BIT *)0xFFA8)    /* Serial Mode Register */
#define COM      SMR_BIT.b7                /* Communication Mode */
#define MP       SMR_BIT.b2                /* Multiprocessor Mode */
#define CKS1     SMR_BIT.b1                /* Clock Select 1 */
#define CKS0     SMR_BIT.b0                /* Clock Select 0 */
#define SCR3_BIT (*(struct BIT *)0xFFAA)   /* Serial Control Register 3 */
#define TE       SCR3_BIT.b5                /* Transmit Enable */
#define CE1     SCR3_BIT.b1                /* Clock Enable 1 */
#define CE0     SCR3_BIT.b0                /* Clock Enable 0 */
#define TDR      *(volatile unsigned char *)0xFFAB /* Transmit Data Register */
#define SSR_BIT (*(struct BIT *)0xFFAC)   /* Serial Status Register */
#define TDRE     SSR_BIT.b7                /* Transmit Data Register Empty */
#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 PMR1_BIT (*(struct BIT *)0xFFE0)  /* Port Mode Register 1 */
#define PMR11    PMR1_BIT.b1              /* TXD Output Terminal */

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

```

プログラムリスト

```

/*****/
/* 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 */

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

COM = 1; /* Initialize Communication Mode */
MP = 0; /* Initialize Multiprocessor Mode */
CKS1 = 1; /* Initialize Clock Select 1 /64 */
CKS0 = 1; /* Initialize Clock Select 0 /64 */

CKE1 = 0; /* Initialize Clock Enable 1 */
CKE0 = 0; /* Initialize Clock Enable 0 */

TDRE = 0; /* Clear TDRE */

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

STD0 = 0x00; /* Set Serial Transfer Data 0 */
STD1 = 0x55; /* Set Serial Transfer Data 1 */
STD2 = 0xAA; /* Set Serial Transfer Data 2 */
STD3 = 0xFF; /* Set Serial Transfer Data 3 */

TE = 1; /* Start Serial Transmitting */

for(counter = 0 ; counter < 4 ; counter++){ /* Serial Transmitting Data Counter 4 Loop */
TDR = *ptr; /* Write Serial Transmit Data to TDR */
ptr++; /* Increment Serial Transmitting Data Address */
while(TDRE != 1){ /* TDRE = 1 ? */
;
}
}
}

```

プログラムリスト

```

while(TEND != 1){
    ;
}
TE = 0;
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