```
/************************************************************************************
    sci.c           H8/300H Serial Input/Output module

       This module controls the serial input and output with receiving interrupt handler.

       communication parameters(fixed)
              19200 bps
              data 8 bits
              stop bit
              none XON/XOFF
              none hardware flow control
       Can chose echo back function at receving process
        Can support the specific function call when the module in an idling time.

       Original source
              Copyright (C) 1998, Office SoftBone
              SoftBone Repesentative Tanaka, Toshihide
              ttanaka@cb3.so-net.ne.jp
              http://member.nifty.ne.jp/softbone/index.html

       Ammended by Kenji Arai / JH1PJL            April         24,1998
        Modified comments                  March         15,1999
              Modified for H8_3664                April         21,2001
              Add toupper function                November      11,2001
              Modified SciGetS                          November      14,2001
                                                                      November      14,2001
              September 18,2005     Change to H8/3694F
              October  1,2005
************************************************************************************/
#pragma interrupt (Sci3ISR(vect=23))

/* -----< Include Files >--------------------------------------------------- */
#include "sci.h"
#include "iodefine.h"
//#include "3664F.H"

/* -----< Definition >------------------------------------------------------ */
#defineTOUPPER 1     /* 1= To upper char                */

static struct{
       char *buf;
       short size;
       short head, tail;
       void (*idleFunc)( void );
} sciInfo;

/* -----< Control program >------------------------------------------------- */
static void nop( void )
{
       ;
}

void SciInit( void* rxBuf, short rxBufSize, void (*idleFunc)( void ) )
{
       short i;

       /* Set parameters */
       sciInfo.buf = (char*)rxBuf;
```

```
        sciInfo.size = rxBufSize;
        sciInfo.head = 0;
        sciInfo.tail = 0;
        sciInfo.idleFunc = idleFunc;

        /* Set up the controller */
        SCI3.SCR3.BYTE = 0;          /* TIE=0,RIE=0,TE=0.RE=0,MPIE=0, TEIE=0, CKE1,0=0 */
        SCI3.SMR.BYTE = 0;           /* Com=0,CHR=0,PE=0,PM=N/A=0,STOP=0,MP=0,CKS1,0=0 */
        SCI3.BRR = SCI19200;  /* Set baud rate */
        for ( i = 0 ; i < 100 ; i++ ){
                nop();
        }

        SCI3.SSR.BIT.OER = 0;
        SCI3.SSR.BIT.FER = 0;
        SCI3.SSR.BIT.PER = 0;
        SCI3.SCR3.BIT.TE = 1;
        SCI3.SCR3.BIT.RE = 1;
        SCI3.SCR3.BIT.RIE = 1;
        IO.PMR1.BIT.TXD =1;          /* Port set to SCI */
}

void SciPutC( char c )
{
        while ( SCI3.SSR.BIT.TDRE == 0 ){
                nop();
        }
        SCI3.TDR = c;
/*      SSR.BIT.TDRE = 0;  Oct.8,2001 */
}

void SciPutS( char *buf )
{
        while ( *buf )
                SciPutC( *(buf++) );
}

char SciGetC( short echo )
{
        char *buf = sciInfo.buf;
        short size = sciInfo.size;
        short head = sciInfo.head;
        short tail = sciInfo.tail;
        void (*idleFunc)( void ) = sciInfo.idleFunc;
        short next;char c;

        while ( ! SciRxReady() )    /* Receiving buffer is empty */
        {
                if ( idleFunc ){
                        idleFunc();
                }
        }

        next = tail + 1;
        if ( next >= size )
                next = 0;
        c = buf[tail];
        sciInfo.tail = next;
```

```
        if ( echo )
                SciPutC( c );
        return c;
}

/* Add new function April 24,1998 Kenji Arai */

char AsmGetC( void )
{
        return SciGetC( 0 );         /* Read received data with No echo mode */
}

void SciGetS( char *buf, short echo )
{
        char *org = buf;
        char c;

        while ( 1 ){
                c = SciGetC( 0 );
                if ( c == '\r' ){
                        *buf = 0;
                        if ( echo ){
                                SciPutC(0x0d);
                                SciPutC(0x0a);
                        }
                        break;
                }
                if ( c == '\b' && buf > org ){
                        SciPutC( '\b' );
                        SciPutC( ' ' );
                        SciPutC( '\b' );
                        buf--;
                }
                else{
                        if ( echo ){
                                SciPutC( c );
                        }
#ifdef TOUPPER
                        if( c >= ' ' ){
                                c = toupper(c);
                        }
#endif
                        *(buf++) = c;
                }
        }
}

char toupper( char c )
{
        return( 'a' <= c && c <= 'z' ? c  - 'a' + 'A' : c );
}

short SciRxReady( void )
{
        return sciInfo.head != sciInfo.tail;
}

void SciRxISR( char data )
```

```
{
        char *buf = sciInfo.buf;
        short size = sciInfo.size;
        short head = sciInfo.head;
        short tail = sciInfo.tail;
        short next;

        next = head + 1;
        if ( next >= size )
                next = 0;
         if ( next == tail )
                return;/* Buffer is full */
        buf[head] = data;
        sciInfo.head = next;
}


/* Add new ISR function June 27,1998 Kenji Arai */

/*      -------------
        SCI Interrupt
        ------------- */
void Sci3ISR( void )
{
        char c;

        c = SCI3.RDR;
        SciRxISR( c );
        SCI3.SSR.BIT.RDRF = 0;
        /* error reset */
        SCI3.SSR.BIT.OER = 0;
        SCI3.SSR.BIT.FER = 0;
        SCI3.SSR.BIT.PER = 0;
}
```