```
/*-------------------------------------------------------------------------------------------+
|                            Interfacing the PC AT Keyboard                                 |

|                              (MPU AT89C51 with 32K Ram)                                   |
| Name : ATKeyboard.c                                                                       |
| Purpose:                                                                                  |
| Provide the interface for the ATKeyboard, convert the keyboard scancode into ascii code and |
| then transmit the codes through RS232 port.                                               |
+-------------------------------------------------------------------------------------------*/
#include "config.h"

void main(void)
{
    static unsigned char rcvdata,key;
    init_uart();                            // 9600 baud @ 11.0592MHz
    init_int0();                            // enable INT0

    while(1)                                // use polling method to handle AT keyboard
    {                                       // interface.
        rcvdata=getrcvdata();
    //   rcvdata=simscancode(0x15,0xf0,0x15,0x15,0xf0,0x15,0x2D,0xf0,0x2D);
    //   print_hexdata(rcvdata);

        if(isvalidcode(rcvdata))
        {
            key=scancode2Ascii(rcvdata);
            if(isdispkey(rcvdata))
                printf("%c",key);
        }
    }
}
```

```
/*-------------------------------------------------------------------------------------------+
| Name : uart.c                                                                              |
| Purpose:                                                                                   |
| Initalize the com port interface in 9600 baud with crystal 11.0592MHz                     |
| Initalize the external interrupt 0 (P3.2).                                                 |
+-------------------------------------------------------------------------------------------*/
#include "config.h"

// com port with 9600 baud with crystal 11.0592MHz.
void init_uart(void)
{
    SCON  = 0x50;
    TMOD |= 0x20;
    TH1   = 253;
    TR1   = 1;
    TI    = 1;
}

// initalize external interrupt 0 (P3.2)
void init_int0(void)
{
    PX0=1;  //Define Int0 high priority
    IE0=0;  //External Interrupt 0 edge flag, set when external interrupt detected,
            //cleared when interrupt is processed.
    IT0=1;  //set to specific falling edge produce interrupt
    EX0=1;  //enable External Interrupt 0
    EA=1;   //enable all interrupt
}
```

```
/*-------------------------------------------------------------------------------------------+
| Name : config.h                                                                            |
| Purpose:                                                                                   |
| Define the common include files for the program, hardware pin connection and function      |
| protocol.                                                                                  |
+-------------------------------------------------------------------------------------------*/
#include <reg51.h>
#include <stdio.h>

// define hardware pin connection
sbit CLK = P3^2;            // keyboard clock input pin
sbit KBDATA = P1^0;         // keyboard data input pin

// uart.c
void init_uart(void);
void init_int0(void);

// decode.c
bit isdispkey(unsigned char rcvdata);
bit isvalidcode(unsigned char rcvdata);
unsigned char scancode2Ascii(unsigned char sc);

//hex.c
void print_hexdata(unsigned char hexdata);
unsigned char hex2ascii(unsigned char bits_data);

//kbdata.c
unsigned char getrcvdata(void);

//simscancode.c
unsigned char simscancode(unsigned char a,b,c,d,e,f,g,h,i);
```

```
/*-------------------------------------------------------------------------------------------+
| Name : hex.c                                                                               |
| Purpose:                                                                                   |
| Provide the function that will print the hexdata on screen through RS232 port.             |
+-------------------------------------------------------------------------------------------*/
#include"config.h"

/*
    print the value of hex data on screen
*/
void print_hexdata(unsigned char hexdata)
{
    static unsigned char hex_h;
    static unsigned char hex_l;
    hex_l = hex2ascii(hexdata&0x0f);                    // convert low byte
    hex_h = hex2ascii((hexdata&0xf0)/0x0f);             // convert high byte
    printf("%c%c",hex_h,hex_l);                         // print out
}

/*
    convert the hex data(4 bits) to ascii code(8 bits)
*/
unsigned char hex2ascii(unsigned char bits_data)
{
    if(bits_data >=0x0f)                                // all invalid data return 'F'
    {
        return 0x46;
    }
    if (bits_data >=0x00 & bits_data <=0x09)            // return '0'-'9'
    {
        bits_data +=0x30;
        return bits_data;
    }
    if (bits_data >=0x0a & bits_data <=0x0f);           // return 'A'-'F'
    {
        bits_data +=0x37;
        return bits_data;
    }
}
```

```
/*-------------------------------------------------------------------------------------------+
| Name : decode.c                                                                            |
| Purpose:                                                                                   |
| The file provide the functions for the convertion of the keyboard scancode into ascii code. |
| Before send the code to the convertion function, it is necessary to determine which one is  |
| vaild code and which one is not, a function isvalidcode() will help to do so. The scan code |
| like shift, caps etc will required to pass to the convertion function for the ascii code    |
| convertion. The return value form scancode2ascii(), however is incorrect, it is necessary to|
| hide the key, so isdispkey() will implemented, becasue the usage of printf(ascii).         |
+-------------------------------------------------------------------------------------------*/
#include "config.h"
#include "scanCodes.h"

/*
    If scancode is a function key, the return value from scancode2ascii is wrong. so monitor
    should not display it. 0 means should not display, and 1 means can display the key.
*/
bit isdispkey(unsigned char rcvdata)
{
    // caps,left shift, right shift, key release
    if(rcvdata ==0x58|rcvdata ==0x12 |rcvdata ==0x59 |rcvdata ==0xE0 |rcvdata ==0xF0)
        return 0;
    else
        return 1;
}

/*
    This function can determine whether the rcvdata from keyboard is a valid key or not. For a
    example, when key 'q' is pressed keyboard will return 15F015 to indicate the key pressed and
    then released, so only the 15F0 is a vaild key, and later 15 is invalid. Another example is
    the 'Home' key, the keyboard will return E06CE0F06C, E0 indicate the extend key. Also E06CE0F0
    is valid code, and later 6C is invalid.
*/
bit isvalidcode(unsigned char rcvdata)
{
    static bit releasekey=0;
    if(rcvdata == 0xF0)                 // when 0xF0, the following rcvdata is invalid
    {
        releasekey = 1;
        return 1;
    }
    else
    {
        if(releasekey)                  // if releasekey is true, the current rcvdate is invalid
        {
            releasekey=0;
            return 0;
        }
        else                            // else the current rcvdata is valid
        {
            return 1;
        }
    }
}

/*
    the scancode decoder for the ascii output. with memory for extend, shift and caps
    if fine scancode not match, a alert bell will sound.
*/
unsigned char scancode2Ascii(unsigned char sc)
{
    static bit extend = 0;
    static bit shift = 0;
    static bit caps  = 0;
    static unsigned char kbdata;
    unsigned int i;

        switch (sc)
        {
            // FUNCTION CODE...
            case    0xE0:                       // if extended code
                    extend = 1;
                    break;
            case    0xF0 :                      // The key is released, no extended
                    extend = 0;                 // code
                    break;
            case    0x12 :                      // Left SHIFT
```

```
                shift ^=1;
                break;
        case    0x59 :                          // Right SHIFT
                shift ^= 1;
                break;
        case    0x58:                           // cap
                caps ^= 1;
                if(!caps)
                    shift=0;
                if(caps)
                    shift=1;
                break;
        // DATA CODE...
        default:
        {
            // IF EXTENDED CODE IS IMPLEMENTED...
            if(extend)
            {
                extend = 0;
                // do a table look-up
                for(i = 0; extended[i][0]!=sc && extended[i][0]; i++);
                    if (extended[i][0] == sc) { // end look-up if end search
                        kbdata = extended[i][1];
                    }
                    else                        // not match return a alert bell
                    {
                        kbdata='\a';
                    }
            }
            // IF NOT EXTENDED CODE...
            else
            {
                // IF NOT SHIFTED...
                if(!shift)
                {   // do a table look-up
                    for(i = 0; unshifted[i][0]!=sc && unshifted[i][0]; i++);
                    if (unshifted[i][0] == sc) {    // end look-up if end search
                        kbdata = unshifted[i][1];
                    }
                    else                            // not match return a alert bell
                    {
                        kbdata='\a';
                    }
                    if(caps)
                    {
                        shift=1;
                    }
                }
                // IF SHIFTED...
                else
                {
                    // do a table look-up
                    for(i = 0; shifted[i][0]!=sc && shifted[i][0]; i++);
                    if (shifted[i][0] == sc) {      // end look-up if end search
                        kbdata = shifted[i][1];
                    }
                    else                            // not match return a alert bell
                    {
                        kbdata='\a';
                    }
                    if(!caps)
                    {
                        shift = 0;
                    }
                }// end of !shiftIf
            }// end of extendIf
            break;
        }// end of default
    }// end of switch
    return kbdata;
}
```

```
/*-------------------------------------------------------------------------------------------+
| Name : kbdata.c                                                                            |
| Purpose:                                                                                   |
| Get the keyboard scancode form the keyboard, and export it to other function.             |
| Ps: I have try to export and implement convertion directly when receive a newer scancode,  |
|     the method, however, is failed. The reason, i guess is the time is not enough before   |
|     another interrupt occur. So, i use a variable to store the scancode, process it and use |
|     the buffer that provided by the standard function printf().                            |
+-------------------------------------------------------------------------------------------*/
#include"config.h"

static unsigned char rcvdata;                   // keyboard scancode data
static bit ok;                                  // scancode data validlity

/*
    export the keyboard scancode, wait until a new scancode is received.
*/
unsigned char getrcvdata(void)
{
    while(!ok);                                 // wait until a scancode received.
    ok=0;                                       // when get the scancode, clear the ok state
    return rcvdata;
}

/*
    External interrupt 0 will triger the clock pin, as the bit count continues, the scancode
    data can be obtained and stored in variable rcvdata. Flag ok indicate the validable of the
    scancode data.
*/
void getkbData(void) interrupt 0
{
    static unsigned char maskdata,bitcount;
    bitcount++;
    // bit 2 to 9 is data, 10 is parity, 11 is stop bit and 1 is start bit
    switch(bitcount)
    {
        case 1:
                break;
        case 2:
                if (KBDATA==1)
                maskdata |= 0x01;
                break;
        case 3:
                if (KBDATA==1)
                maskdata |= 0x02;
                break;
        case 4:
                if (KBDATA==1)
                maskdata |= 0x04;
                break;
        case 5:
                if (KBDATA==1)
                maskdata |= 0x08;
                break;
        case 6:
                if (KBDATA==1)
                maskdata |= 0x10;
                break;
        case 7:
                if (KBDATA==1)
                maskdata |= 0x20;
                break;
        case 8:
                if (KBDATA==1)
                maskdata |= 0x40;
                break;
        case 9:
                if (KBDATA==1)
                maskdata |= 0x80;
                rcvdata=maskdata;
                ok=1;
                break;
        case 10:
                break;
        case 11:
                bitcount=0;
                maskdata=0x00;
```

```
                break;
    }// end of switch
}
```

```
/*-------------------------------------------------------------------------------------------+
| Name : simscancode.c                                                                       |
| Purpose:                                                                                   |
| simulate 9 keyboard scancodes                                                              |
+-------------------------------------------------------------------------------------------*/
// EXAMPLE:
// rcvdata=simscancode(0x15,0xf0,0x15,0x15,0xf0,0x15,0x2D,0xf0,0x2D);    >> 'q' 'q' 'w'
// rcvdata=simscancode(0xe0,0x6c,0xe0,0xf0,0x6c,0x15,0xf0,0x15,0xf0);    >> 'Home' 'q'
//                                                                        the last one 0xf0 is wrong.
#include"config.h"

// simluate 9 keyboard scancodes
unsigned char simscancode(unsigned char a,b,c,d,e,f,g,h,i)
{
    static unsigned int z;
    switch(z)
    {
        case 0:
            z=1;
            return a;
            break;
        case 1:
            z=2;
            return b;
            break;
        case 2:
            z=3;
            return c;
            break;
        case 3:
            z=4;
            return d;
            break;
        case 4:
            z=5;
            return e;
            break;
        case 5:
            z=6;
            return f;
            break;
        case 6:
            z=7;
            return g;
            break;
        case 7:
            z=8;
            return h;
            break;
        case 9:
            z=0;
            return i;
            break;
    }
}
```

```
/*-------------------------------------------------------------------------------------------+
| Name : scancodes.h                                                                         |
| Purpose:                                                                                   |
| The database for the AT keyboard scan code convert to ascii code                           |
+-------------------------------------------------------------------------------------------*/

// Unshifted characters
unsigned char xdata unshifted[][2] = {
    0x0d,'\t',
    0x0e,'`',
    0x15,'q',
    0x16,'1',
    0x1a,'z',
    0x1b,'s',
    0x1c,'a',
    0x1d,'w',
    0x1e,'2',
    0x21,'c',
    0x22,'x',
    0x23,'d',
    0x24,'e',
    0x25,'4',
    0x26,'3',
    0x29,' ',
    0x2a,'v',
    0x2b,'f',
    0x2c,'t',
    0x2d,'r',
    0x2e,'5',
    0x31,'n',
    0x32,'b',
    0x33,'h',
    0x34,'g',
    0x35,'y',
    0x36,'6',
    0x39,',',
    0x3a,'m',
    0x3b,'j',
    0x3c,'u',
    0x3d,'7',
    0x3e,'8',
    0x41,',',
    0x42,'k',
    0x43,'i',
    0x44,'o',
    0x45,'0',
    0x46,'9',
    0x49,'.',
    0x4a,'/',
    0x4b,'l',
    0x4c,';',
    0x4d,'p',
    0x4e,'-',
    0x52,'\'',
    0x54,'[',
    0x55,'=',
    0x5a,'\n',
    0x5b,']',
    0x5d,'\\',
    0x61,'<',
    0x66,8,
    0x69,'1',
    0x6b,'4',
    0x6c,'7',
    0x70,'0',
    0x71,'.',
    0x72,'2',
    0x73,'5',
    0x74,'6',
    0x75,'8',
    0x76,'\a',
    0x79,'+',
    0x7a,'3',
    0x7b,'-',
    0x7c,'*',
    0x7d,'9',
    0,0
```

```
};
// Shifted characters
unsigned char xdata shifted[][2] = {
    0x0d,'\t',
    0x0e,'~',
    0x15,'Q',
    0x16,'!',
    0x1a,'Z',
    0x1b,'S',
    0x1c,'A',
    0x1d,'W',
    0x1e,'@',
    0x21,'C',
    0x22,'X',
    0x23,'D',
    0x24,'E',
    0x25,'$',
    0x26,'#',
    0x29,' ',
    0x2a,'V',
    0x2b,'F',
    0x2c,'T',
    0x2d,'R',
    0x2e,'%',
    0x31,'N',
    0x32,'B',
    0x33,'H',
    0x34,'G',
    0x35,'Y',
    0x36,'^',
    0x39,'L',
    0x3a,'M',
    0x3b,'J',
    0x3c,'U',
    0x3d,'&',
    0x3e,'*',
    0x41,'<',
    0x42,'K',
    0x43,'I',
    0x44,'O',
    0x45,')',
    0x46,'(',
    0x49,'>',
    0x4a,'?',
    0x4b,'L',
    0x4c,':',
    0x4d,'P',
    0x4e,'_',
    0x52,'"',
    0x54,'{',
    0x55,'+',
    0x5a,'\n',
    0x5b,'}',
    0x5d,'|',
    0x61,'>',
    0x66,8,
    0x69,'1',
    0x6b,'4',
    0x6c,'7',
    0x70,'0',
    0x71,'.',
    0x72,'2',
    0x73,'5',
    0x74,'6',
    0x75,'8',
    0x76,'\a',
    0x79,'+',
    0x7a,'3',
    0x7b,'-',
    0x7c,'*',
    0x7d,'9',
    0,0
};

// extended code characters
unsigned char xdata extended[][2] = {
    0x75,11,
```

```
    0x6b,12,
    0x72,13,
    0x74,14,
    0x6c,'\r',
    0x4a,'/',
    0x5a,'\n',
    0,0
};
```