```
/**************************************************
 MSSP i2c LCD TEST of PIC12F1827
             By nobcha all right reserved

  Ver 1.0 07/04/2012 for i2c LCD TEST
  Ver 4.0 09/26/2012 for STRAWBERY LINUX i2c
                     & 4bit LCD

  LCDとしてのお約束
     １．電源立ち上げイニシャライズは自分でやるので、電源立ち上げ後
     ３００ｍｓ待ってからアクセスする。
     ２．LCDへのコマンド発行はi2cアドレス0x7CへRSバイト（：データ時
     は0x40、制御コマンド時0x40）とデータあるいは制御コマンドの２バ
     イトを連続ペアで送る。たとえば画面クリアなら、0x00,0x01と送る。
  キーパッドのお約束
     １．キースキャン開始は0x10を送る。スキャンストップは0x11。
     ２．キースキャン時はLCDコマンド発行不可。もし重なるとデッドロック
     になる。
     ３．キーコードはASCII、長押しがあると７ビット目が１になる。

         RA0-3:SC1602は4ビットモ―ドとし、RA0－3で接続します。
         RB6:LCD EN bit (enable)はRB6につなぎます。
         RB7:LCD RS bit (RS)はRB7につなぎます。
         RB1:SDA1 MSSP  sl LCD+KEY, sn
         RB3:SW2
         RB4:SCL1 MSSP
         RA7:動作確認用LED
         RB2:1:normal 0:Debug

         SC1602 pin connection via 4bit mode
                 #1      Vdd=5V
                 #2      Vss=GND
                 #3      LCD contrust center of 5k VOL
                 #4      RS      RB7
                 #5      R/W     GND
                 #6      EN      RB6

                 #11-14  DATA    RA0-3

    OSC INT 8MHz
   Development Circumstance
   MPLAB IDE V8.85   HiTECH C V9.83
************************************************** */

#define _XTAL_FREQ 8000000
#define PIC_CLOCK  8000000

#define HEART_BEAT LATA7
#define SW2 RB3


#include <htc.h>
#include "mssp_i2c.h"
#include "lcd_key_i2c_sl_mssp.h"
#include "lcd.h"

__CONFIG(
    FOSC_INTOSC & WDTE_OFF & PWRTE_ON & MCLRE_ON & CP_OFF
    & CPD_OFF & BOREN_OFF & CLKOUTEN_ON & IESO_OFF & FCMEN_OFF
        );

__CONFIG(
        WRT_OFF & PLLEN_OFF & STVREN_ON &  LVP_OFF
        );

// function prototyping
void itostring(char digit, unsigned int data, char *buffer);
void mssp_init(void);

unsigned char Msg1[] = "i2c LCD&KEY test";
unsigned char Msg2[] = "CNT=xxxxx       ";
unsigned char Msg3[] = " k=";

void main(void)
{
```

```
        unsigned int Count;
        unsigned char i, key_data;

        /* INITIALIZE REGISTER      */
        PORTA = 0b00001001;                     //
        PORTB = 0b00000000;                     //
        TRISA = 0b01110000;                     /* RA0-3は出力、RA4-6入力、RA7出力  */
        TRISB = 0b00011111;                     /* RB0ー4は、ｓｗ、i2c用で入力、RB5-7出力設定  */
        OSCCON = 0b01110000;                    /* 内部クロック8MHz            */

        GIE=0;
        ANSELA = 0x00;                          /* 全ポートデジタル            */
        ANSELB = 0x00;
        OPTION_REG = 0x00;                      /* オプション設定なし          */

        CM1CON0 = 0b00000111;                   // No using compalator
        CM2CON0 = 0b00000111;                   // No using compalator

        mssp_init();                            // MSSP initialize
        Count = 0;
        RA0 = 1;

//      i2c_sl_lcd_init();                      // bad effect against nobcha LCD
        lcd_init();
        __delay_ms(100);


        while(1) {
                __delay_ms(100);

                HEART_BEAT ^= 1;                // Heart beat
                lcd_goto(0x00);                 // Move cursor 1st line
                lcd_puts(Msg1);                 // Display test message
                itostring(5, Count++, Msg2+4);
                lcd_goto(0x40);                 // Move cursor to 2nd line
                lcd_puts(Msg2);                 // Display count message

                i2c_sl_lcd_cmd(0x80);           // Move cursor 1st line
                i2c_sl_lcd_str(Msg1);           // Display test message

                i2c_sl_lcd_cmd(0xC0);           // Move cursor to 2nd line
                i2c_sl_lcd_str(Msg2);           // Display count message

                i2c_sl_lcd_cmd(0xCA);           // Move cursor to 2nd line 12th chs
                i2c_sl_lcd_str(Msg3);           // Key-in
                lcd_goto(0x49);                 // Move cursor to 2nd line 12th chs
                lcd_puts(Msg3);                 // Key-in

//              i++;
//              lcd_putch((i&0x07)| 0x30);              // loop count
//              i2c_sl_lcd_data((i&0x07)| 0x30);        // loop count

                while(SW2){};
                i2c_sl_key_run(1);                      // key start
                __delay_ms(100);                        // key in wait
                key_data=i2c_sl_key_get();              // get key code
                lcd_putch(key_data);                    // debug

                i2c_sl_key_run(0);                      // key scan stop

//                      i2c_sl_lcd_cmd(0xCC);           // Move cursor to 2nd line 14th chs
                        i2c_sl_lcd_data(key_data&0x7F);
//                      i2c_sl_lcd_data(((key_data&0x80)>>7) | 0x20);
//                      lcd_goto(0x4C);                 // Move cursor to 2nd line 14th chs
//                      lcd_putch(0x22);
//                      lcd_putch(key_data&0x3F);
//                      lcd_putch(((key_data&0x80)>>7) | 0x20);

                for(i=5;i>0;i--){               // 500ms waiting
                        __delay_ms(100);

                }
        }
}
/****************************************
```

```
* Converting binary to ASCII
****************************************/
void itostring(char digit, unsigned int data, char *buffer)
{
        char i;
        buffer += digit;                        // last data
        for(i=digit; i>0; i--) {                //
                buffer--;                       //
                *buffer=(data%10)+'0';          // ASCII code
                data = data / 10;               // next digit
        }
}

/****************************************
* MSSP initialize
****************************************/
void mssp_init(void){
       /* SSP1CON1 REGISTERS */
       SSP1CON1bits.SSPEN = 1;  //Enables Serial Port Mode
       SSP1CON1bits.SSPM3 = 1;  ////////
       SSP1CON1bits.SSPM2 = 0;  //I2C Master Mode
       SSP1CON1bits.SSPM1 = 0;  // clock= Fosc/(4*(SSP1ADD+1))
       SSP1CON1bits.SSPM0 = 0;  ////////

       /* SSPCON2 REGISTERS */
       SSP1CON2   = 0x00;
       /* SSPCON3 REGISTERS */
       SSP1CON3   = 0x00;


       /* SSP1STAT REGISTERS */
       SSP1STATbits.SMP = 1;    //SPI MASTER MODE
       SSP1STATbits.CKE = 1;    //SMBus Specific Inputs Enabled

       //SSP1ADD    = 0x19;     //~75kHz
       //SSP1ADD    = 0x13;     //~100kHz
       //SSP1ADD    = 0x07;     //~400kHz
       SSP1ADD    = 0x50;
}
```