

```

main_slave
/* ////////////////////////////////////// */
/* PIC16F1823にSC1602互換LCD表示パネルを接続してi2cのスレー
/* ブとして表示を行う実験用ルーチンです。sc1602とは4ビット
/* モード接続されます。
*
* PIC16F88 V0.1 2011.09.30 V1.0 2011.10.03 をベースに改造
* PIC16F1823 V2.0 2011.10.21
* MPLAB v8.73&H1-TECH C V9.82
*
* by nobcha (c)2011
*
* i2c経由データは必ず2バイト単位でくるとする。MAX6(2,4,6)
* 1バイト目はRSビット、2バイト目がLCDに書き込むデータです
* ストロベリーリナックスの拡張コマンドはサポートしてません
*
* RC2-5:SC1602は4ビットモードとし、RC2-5で接続します。
* RA0:LCD EN bit (enable)はRA0につなぎます。
* RA1:LCD RS bit (RS)はRA1につなぎます。
* RC1:SDA MSSP
* RC0:SCL MSSP
* RA5:動作確認用LED
* RA2:1:normal 0:Debug
*
* SC1602 pin connection via 4bit mode
* #1 Vdd=5V
* #2 Vss=GND
* #3 LCD contrast center of 2k VOL
* #4 RS RA1
* #5 R/W GND
* #6 E RA0
*
* #11-14 DATA RC2-5
*
* ////////////////////////////////////// */
#include <htc.h>
#include "lcd.h"
#include "delay.h"

#define XTAL_FREQ = 8000000
#define PIC_CLOCK = 8000000
#define I2C_ADR 0x7C
#define DEBUG RA2
#define MON_LED LATA5

__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
);
/* ////////////////////////////////////// */

unsigned char buffer[20]; /* 受信バッファ */
unsigned char stat[20];
char i, j, rcv_count=0, cnt=0;

void init(void){
    OSCCON = 0b01110000; /* 内部クロック8MHz */
    ANSELA = 0x00; /* 全ポートデジタル */
    ANSELB = 0;
    OPTION_REG = 0x00; /* オプション設定なし */
    TRISA = 0b00001100; /* RA2,3は入力 */
    TRISC = 0b00000011; /* RC0とRC1はi2c用で入力設定 */
    LATA = 0b00000000; // port direction: 1:input
    LATC = 0b00000000; //
}

```

```

main_slave
    delay(100);
}

void ssp_init(void){
    SSP1CON1 = 0x36;           // SSP1EN, CKP, SSP_slave_7bit
    SSP1ADD = I2C_ADR;        // SSP_ADDRESS SET
    PEIE = 1;                 // PEIE enable
    SSP1IE = 1;
}

unsigned char lcdrs;
interrupt i2c_slave() {      // i2c slave receive int func
    GIE=0;
    if(SSP1IF==1){
        SSP1IE=0;
        SSP1IF=0;
        buffer[rcv_count]=SSP1BUF;           /* dataをバッファに入れる */
        stat[rcv_count]=SSP1STAT;           /* statusをステータスバッファに入れる */

        if ((SSP1STAT&0x20)==0x20) {         // DATA?
            if ((rcv_count==1) | (rcv_count==3) | (rcv_count==5)){
                if ((buffer[rcv_count]&0x40)==0x40) {lcdrs=1;}
                else {lcdrs=0;}
            }
            else {
                lcd_write_rs(buffer[rcv_count], lcdrs);
            }
        }
        rcv_count++;
        if(rcv_count>7) { lcd_putch(0x3F);} /* 受信バイト数 */
        SSP1IE=1;
    }
}

GIE=1;
}

char MSG1[] = "PIC i2c slave v2.0 ";

void main(void){
    init(); /* ポート初期化 */
    MON_LED=1;
    _delay(100);
    _delay(100);
    lcd_init(); /* LCDの初期化 */
    cnt=0;
    MON_LED=0;
    _delay(100);
    _delay(100);

    if ((DEBUG)==1){ /* debug mode */
        lcd_goto(40); /* 2行目にLCD動作表示 */
        lcd_puts(MSG1); /* LCDに初期表示 */
    }

    ssp_init(); /* SSPの初期化 */
    SSP1IF = 0; /* SSPIF clear */
    MON_LED = 1; /* 動作モニター用LED消灯 */

    rcv_count=0;

    while(1){ /* 繰り返し */
        SSP1IE=1;
        PEIE=1;
        GIE=1;
        SSPEN=1;
        while((SSP1STAT&0x10) ==0x00) { /* 受信のストップ状態チェック */
            GIE=0; /* int disable */

            if ((DEBUG)==0x1){ /* debug mode */
                for(i=0; i<(rcv_count+1); i++){
                    lcd_goto(i*5 | ((cnt&0x01)<<6);
                    lcd_putch(0x23); /* # */
                    lcd_putch(((buffer[i]>>4)&0x0F)|0x30);
                }
            }
        }
    }
}

```

```

                                main_slave
                                lcd_putch(((buffer[i])&0x0F)|0x30);
                                lcd_putch(((stat[i]>>4)&0x0F)|0x30);
                                lcd_putch(((stat[i])&0x0F)|0x30);
                                }
                                lcd_goto(0x53);                                /* 2行目にLCD動作表示 */
                                lcd_putch(cnt++ | 0x30);
                                if (cnt>0x09) {cnt=0;}
                                lcd_putch(0x3C);

                                lcd_putch(((rcv_count)& 0x0F) |0x30); /* 下位4ビット取り出しASCII */
                                lcd_putch(0x3E);
                                lcd_putch(0x20);

                                for (j=0;j<10;j++){                                /* 表示時間を待たせる */
                                    delay(100);
                                }
                                }
                                rcv_count=0;
                                MON_LED = MON_LED ^ 1;
                                }
}

```