Source Codes for Demo kit DAD02

V2.00

This document contains source codes for demo kit DAD02 of compass pressure sensor module DSPC01 and is used only for reference. No gurantee is promised for any potential mistakes. Please refer to datasheet of kit DAD02 in accessories page and schematics in compass pressure sensor page for operation details.

```
#include<pic.h>
#include<math.h>


#define COMMUNI_SUCCEED      1
#define COMMUNI_FAILED       0

#define noACK 0
#define ACK 1


#define PRESSURE_MODE        0
#define Temperature_MODE     1
#define ALTITUDE_MODE        2
#define COMPASS_MODE         3
#define WEATHER_MODE         4
#define POWER_OFF_MODE       5


#define MODE_KEY                              0x1e   //RA0
#define START_COMPASS_CALIBRATION_KEY         0x1d   //RA1
#define EXIT_COMPASS_CALIBRATION_KEY          0x1b   //RA2
#define POWER_ON_KEY                          0x17   //RA3
#define POWER_OFF_KEY                         0x0f   //RA4


#define POWER_ON        0
#define POWER_OFF       1

#define        COMPASS_CALIBRATION_COMMAND             0xe0
#define        EXIT_COMPASS_CALIBRATION_COMMAND        0xe1
#define        PRESS_WRITE_COMMAND                     0xB0
#define        TEMPERATURE_WRITE_COMMAND               0x80
#define        ALTITUDE_WRITE_COMMAND                  0xA0
#define        COMPASS_WRITE_COMMAND                   0xC0
#define        WEATHER_WRITE_COMMAND                   0xB3

#define          MODULE_WAKEUP_WRITE_COMMAND           0X70
#define        MODULE_SLEEP_WRITE_COMMAND              0X71



#define        LED_A            RD0
```

```
#define       LED_B           RD1
#define       LED_C           RD2
#define       LED_D           RD3
#define       LED_E           RD4
#define       LED_F           RD5
#define       LED_G           RD6
#define       LED_DP          RD7

#define       LED_1           RB0     //RB0 //RA0
#define       LED_2           RB1     //RB1 //RA1
#define       LED_3           RB2     //RB2
#define       LED_4           RB3     //RB3 //RA3
#define       LED_5           RB4     //RB4
#define       LED_6           RB5     //RB5 //RA3

#define       SCK       RC0
#define       DATA      RC1

#define       DATA_SET  TRISC1
#define       DATA_INPUT    1
#define       DATA_OUTPUT   0
#define       DELAY100US    10
#define       LED_ON        1
#define       LED_OFF   0

#define       LED_PEDO              RC2
#define       LED_TEMPERATURE       RC3
#define       LED_PRESSURE          RC4
#define       LED_ALTIDUTE          RC5
#define       LED_COMPASS           RC6
#define       LED_WEATHER           RC7

//        0    1  2     3    4    5    6    7   8    9     a   b
const unsigned char
Tab_DispCoed[12]={0xc0,0xf9,0xa4,0xb0,0x99,0x92,0x82,0xf8,0x80,0x90,0xff,0xbf};
unsigned char DispData1;
unsigned char DispData2;
unsigned char DispData3;
unsigned char DispData4;
unsigned char DispData5;
unsigned char DispData6;
```

```
unsigned char DispCount;

unsigned char key;
unsigned char mode;

unsigned char press_h, press_m, press_l;
unsigned char altitude_h, altitude_m, altitude_l;
unsigned char weather;
unsigned char compass_h, compass_l;
unsigned char temperature_h, temperature_l;
unsigned char communicate_status;


unsigned long int cal_temp_ram;

typedef struct
{

    unsigned char reach_1s              : 1;

    unsigned char reach_5hz             : 1;

    unsigned char temp_is_positive      : 1;
    unsigned char is_disp_press         : 1;
    unsigned char is_disp_altitude      : 1;
    unsigned char is_disp_compass       : 1;
    unsigned char is_disp_pedo          : 1;
    unsigned char is_disp_temp          : 1;
    unsigned char is_disp_weather       : 1;

}   FlagType;

FlagType  Flag;

void port_init(void);

void timer2_init(void);

void IIC_SCL_HIGH(void);
void IIC_SCL_LOW(void);
void IIC_Start(void);
void IIC_Stop(void);
void IIC_ACK(void);
```

```
void IIC_NoAck(void);
unsigned char IIC_ReadByte(void);
unsigned char IIC_WriteByte( unsigned char ucData );

void SysDelay1ms(unsigned int x);
void SysDelay(unsigned int t);

unsigned char key_scan_main_program(void);
void pressure_led_display(void);
void temperature_led_display(void);
void altitude_led_display(void);
void compass_led_display(void);
void weather_led_display(void);
void poweroff_led_display(void);

void send_SPC01_write_command(unsigned char command);
void send_read_press_command(void);
void send_read_altitude_command(void);
void send_read_compass_command(void);
void send_read_weather_command(void);

void display_press(void);
void display_altitude(void);
void display_temperature(void);
void display_compass(void);
void display_weather(void);
void key_handle_program(void);
void mode_handle_program(void);
void send_read_temperature_command(void);
void display_compass_calibrate(void);
void display_power_off(void);

void main()
{
    OSCCON = 0X70;     // 8M crystal

    WDTCON = 0X00;
    DispCount = 2;
    SysDelay1ms(1000);
    timer2_init(); //
    port_init();

     INTCON = 0xc0;   // Enable interrupt, 5MS interruption for display
```

```
        DispData1 = 0x01;
        DispData2 = 0x02;
        DispData3 = 3;
        DispData4 = 4;
        DispData5 = 5;
        DispData6 = 6;


    send_SPC01_write_command(MODULE_WAKEUP_WRITE_COMMAND);
    while( communicate_status != COMMUNI_SUCCEED)
    {
        send_SPC01_write_command(MODULE_WAKEUP_WRITE_COMMAND);
    }

    SysDelay1ms(1000);

     mode = PRESSURE_MODE;

     while(1)
     {
        key = key_scan_main_program();
        key_handle_program();
        mode_handle_program();

    }

}

unsigned char key_scan_main_program(void)
{
    unsigned char i,j;

    i = PORTA;
    i &= 0x1f;   // only RA0 --RA4 , normal high
    if ( i != 0x1f)
    {
        SysDelay1ms(30);
        j = PORTA;
        j &= 0x1f;
        if(i == j)
        {
            SysDelay1ms(100);
```

```
            return (i);   // key pressed

        }
        else
        {
            return(0);
        }
    }
    else return(0);
}


void timer2_init(void)
{
    T2CON = 0x7f; // timer2 on and 16 pre, postscale
    PR2 = 15;    //156;  // 50hZ*4, 4m/4/16/16/50 = 5ms
    TMR2IE = 1;
    NOP();
}


void interrupt ISR_timer(void)
{
    unsigned char i;
    if(TMR2IF)
    {


            switch(DispCount)
        {
            case 2:
                PORTD = Tab_DispCoed[DispData1];
                LED_1 = LED_ON ;
                LED_2 = LED_OFF;
                LED_3 = LED_OFF;
                LED_4 = LED_OFF;
                LED_5 = LED_OFF;
                LED_6 = LED_OFF;


                        DispCount <<= 1;
                break;
```

```
case 4:
     PORTD = Tab_DispCoed[DispData2];
     LED_1 = LED_OFF ;
     LED_2 = LED_ON;
     LED_3 = LED_OFF;
     LED_4 = LED_OFF;
     LED_5 = LED_OFF;
     LED_6 = LED_OFF;

          DispCount <<= 1;
     break;

     case 8:
          PORTD = Tab_DispCoed[DispData3];
          LED_1 = LED_OFF ;
     LED_2 = LED_OFF;
     LED_3 = LED_ON;
     LED_4 = LED_OFF;
     LED_5 = LED_OFF;
     LED_6 = LED_OFF;

     DispCount <<= 1;
     break;

case 16:
     PORTD = Tab_DispCoed[DispData4];
     if(Flag.is_disp_press)
     {
          PORTD &= 0x7f;     // display dot
     }
     LED_1 = LED_OFF ;
     LED_2 = LED_OFF;
     LED_3 = LED_OFF;
     LED_4 = LED_ON;
     LED_5 = LED_OFF;
     LED_6 = LED_OFF;
     DispCount <<= 1;
     break;
case 32:
     PORTD = Tab_DispCoed[DispData5];
     if((Flag.is_disp_temp)||(Flag.is_disp_altitude))
     {
```

```
                    PORTD &= 0x7f;     // display dot
                }
                LED_1 = LED_OFF ;
                LED_2 = LED_OFF;
                LED_3 = LED_OFF;
                LED_4 = LED_OFF;
                LED_5 = LED_ON;
                LED_6 = LED_OFF;
                DispCount <<= 1;
                break;
              case 64:
                PORTD = Tab_DispCoed[DispData6];
                LED_1 = LED_OFF ;
                LED_2 = LED_OFF;
                LED_3 = LED_OFF;
                LED_4 = LED_OFF;
                LED_5 = LED_OFF;
                LED_6 = LED_ON;
                    DispCount = 2;
                 break;
        }
        TMR2IF=0;
    }

}



void port_init(void)
{
    ANSELA = 0;
    ANSELB = 0;
    ANSELD = 0;
    ANSELE = 0;

    TRISA = 0xFF;                          //  RA PORT IS INPUT ( KEY + OSC)
    TRISB = 0x00;                          // rb IS OUTPTU
    TRISC = 0X00;                          // PORT C IS OUTPUT
    TRISD = 0X0;                           // PORT D IS OUTPUT
    TRISE = 0x00;                         // PORTE is output
    PORTB = PORTC = PORTD= PORTE =  0xff;   // all port is high

}
```

```
//=============================================

void SysDelay(unsigned int t)
{
     t *= 2;
     for(;t>0;t--)
     {
          #asm
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          NOP
          #endasm
     }

}

//=============================================

void SysDelay1ms(unsigned int x)
{
     unsigned char i;
     x *= 2;
     for (; x>0; x--)
     {
          for(i = 200; i>1; i--)
          {
               NOP();
          }
     }

}


void pressure_led_display(void)
{
     LED_PRESSURE = 1;
     LED_ALTIDUTE = 0;
```

```
    LED_TEMPERATURE = 0;
    LED_COMPASS = 0;
    LED_WEATHER = 0;
    LED_PEDO = 0;
}


void temperature_led_display(void)
{
    LED_PRESSURE = 0;
    LED_ALTIDUTE = 0;
    LED_TEMPERATURE = 1;
    LED_COMPASS = 0;
    LED_WEATHER = 0;
    LED_PEDO = 0;
}
void altitude_led_display(void)
{
    LED_PRESSURE = 0;
    LED_ALTIDUTE = 1;
    LED_TEMPERATURE = 0;
    LED_COMPASS = 0;
    LED_WEATHER = 0;
    LED_PEDO = 0;
}
void compass_led_display(void)
{
    LED_PRESSURE = 0;
    LED_ALTIDUTE = 0;
    LED_TEMPERATURE = 0;
    LED_COMPASS = 1;
    LED_WEATHER = 0;
    LED_PEDO = 0;
}
void weather_led_display(void)
{
    LED_PRESSURE = 0;
    LED_ALTIDUTE = 0;
    LED_TEMPERATURE = 0;
    LED_COMPASS = 0;
    LED_WEATHER = 1;
    LED_PEDO = 0;
}
```

```
void poweroff_led_display(void)
{
    LED_PRESSURE = 0;
    LED_ALTIDUTE = 0;
    LED_TEMPERATURE = 0;
    LED_COMPASS = 0;
    LED_WEATHER = 0;
    LED_PEDO = 0;
}

void key_handle_program(void)
{

    switch(key)
    {
        case MODE_KEY:
        {
            if(mode == POWER_OFF_MODE)
            {
                mode = 0;
            }
            else
            {
                mode ++;
                if(mode == 0x05)
                {
                    mode = 0;
                }
            }
            break;
        }

        case POWER_ON_KEY:
        {
            mode = PRESSURE_MODE;
            send_SPC01_write_command(MODULE_WAKEUP_WRITE_COMMAND);
            while( communicate_status != COMMUNI_SUCCEED)
            {
                send_SPC01_write_command(MODULE_WAKEUP_WRITE_COMMAND);
            }
            SysDelay1ms(200);

            break;
```

```
            }
            case POWER_OFF_KEY:
            {
                mode = POWER_OFF_MODE;
                display_power_off();
                poweroff_led_display();
                send_SPC01_write_command(MODULE_SLEEP_WRITE_COMMAND);
                while( communicate_status != COMMUNI_SUCCEED)
                {
                    send_SPC01_write_command(MODULE_SLEEP_WRITE_COMMAND);
                }

                SysDelay1ms(200);


                break;
            }
            case START_COMPASS_CALIBRATION_KEY:
            {
                display_compass_calibrate();
                mode = COMPASS_MODE;
                send_SPC01_write_command(COMPASS_CALIBRATION_COMMAND);
                while( communicate_status != COMMUNI_SUCCEED)
                {
                    send_SPC01_write_command(COMPASS_CALIBRATION_COMMAND);
                }

                break;
            }
            case EXIT_COMPASS_CALIBRATION_KEY:
            {
                mode = COMPASS_MODE;
                send_SPC01_write_command(EXIT_COMPASS_CALIBRATION_COMMAND);
                while( communicate_status != COMMUNI_SUCCEED)
                {
                    send_SPC01_write_command(EXIT_COMPASS_CALIBRATION_COMMAND);
                }
                break;
            }
        }
}

void mode_handle_program(void)
```

```
{
    if(mode == PRESSURE_MODE)
    {
        pressure_led_display();
        send_SPC01_write_command(PRESS_WRITE_COMMAND);
        while( communicate_status != COMMUNI_SUCCEED)
        {
            send_SPC01_write_command(PRESS_WRITE_COMMAND);
        }

        SysDelay1ms(200);
        send_read_press_command();
        display_press();

        SysDelay1ms(10);
    }
    else if(mode == Temperature_MODE)
    {
        temperature_led_display();
        send_SPC01_write_command(TEMPERATURE_WRITE_COMMAND);
        while( communicate_status != COMMUNI_SUCCEED)
        {
            send_SPC01_write_command(TEMPERATURE_WRITE_COMMAND);
        }

        SysDelay1ms(200);
        send_read_temperature_command();
        display_temperature();

        SysDelay1ms(10);
    }
    else if(mode == ALTITUDE_MODE)
    {
        altitude_led_display();
        send_SPC01_write_command(ALTITUDE_WRITE_COMMAND);
        while( communicate_status != COMMUNI_SUCCEED)
        {
            send_SPC01_write_command(ALTITUDE_WRITE_COMMAND);
        }
        SysDelay1ms(200);
        send_read_altitude_command();
        display_altitude();
```

```
            SysDelay1ms(10);
        }
        else if(mode == COMPASS_MODE)
        {
            compass_led_display();
            send_SPC01_write_command(COMPASS_WRITE_COMMAND);
            while( communicate_status != COMMUNI_SUCCEED)
            {
                send_SPC01_write_command(COMPASS_WRITE_COMMAND);
            }
            SysDelay1ms(200);
            send_read_compass_command();
            display_compass();

            SysDelay1ms(10);
        }
        else if(mode == WEATHER_MODE)
        {
            weather_led_display();
            send_SPC01_write_command(WEATHER_WRITE_COMMAND);
            while( communicate_status != COMMUNI_SUCCEED)
            {
                send_SPC01_write_command(WEATHER_WRITE_COMMAND);
            }
            SysDelay1ms(200);
            send_read_weather_command();
            display_weather();

            SysDelay1ms(10);
        }
}


void display_press(void)
{
    unsigned long int  j;

    Flag.is_disp_press = 1;
    Flag.is_disp_altitude = 0;
    Flag.is_disp_compass = 0;
    Flag.is_disp_pedo = 0;
    Flag.is_disp_temp = 0;
    Flag.is_disp_weather = 0;
```

```
    NOP();

    cal_temp_ram = press_h;
    cal_temp_ram *= 256;
    cal_temp_ram *= 256;


    cal_temp_ram += (unsigned int)(press_m <<8);
    cal_temp_ram += press_l;

    DispData6 = cal_temp_ram%10;

    j= cal_temp_ram/10;
    DispData5 = j%10;

    j= cal_temp_ram/100;
    DispData4 = j%10;


    j= cal_temp_ram/1000;
    DispData3 = j%10;

    j= cal_temp_ram/10000;
    DispData2 = j%10;

    j= cal_temp_ram/100000;
    DispData1 = j%10;
    if(DispData1 ==0)
    {
        DispData1 = 0x0a;
        if( DispData2 ==0)
        {
            DispData2 = 0x0a;
        }
    }
}
void display_altitude(void)
{
    unsigned long int i, j;


    Flag.is_disp_press = 0;
```

```
Flag.is_disp_altitude = 1;
Flag.is_disp_compass = 0;
Flag.is_disp_pedo = 0;
Flag.is_disp_temp = 0;
Flag.is_disp_weather = 0;

if((altitude_h &0x80) ==0x80)
{
     DispData1= 0x0b  ;   // display -  altitude is negative
     altitude_h &= 0x7f;
}
else
{
     DispData1= 0x0a  ;
}

i = altitude_h;
i <<= 8;
i <<= 8;
i += (unsigned int)(altitude_m <<8);
i += altitude_l;

DispData6 = i%10;

j= i/10;
DispData5 = j%10;

j= i/100;
DispData4 = j%10;

j= i/1000;
DispData3 = j%10;

j= i/10000;
DispData2 = j%10;

if( DispData2 ==0)
{
     DispData2 = 0x0a;
     if(  DispData3 ==0 )
     {
          DispData3 = 0x0a;
          if(DispData4 == 0x0)
```

```
                {
                    DispData4 = 0x0a;
                }
            }
        }
}
void display_temperature(void)
{
    unsigned int i, j;


    Flag.is_disp_press = 0;
    Flag.is_disp_altitude = 0;
    Flag.is_disp_compass = 0;
    Flag.is_disp_pedo = 0;
    Flag.is_disp_temp = 1;
    Flag.is_disp_weather = 0;

    if((temperature_h &0x80) ==0x80)
    {
        DispData1= 0x0b  ;   // display -  ,temperature is negative
        temperature_h &= 0x7f;
    }
    else
    {
        DispData1= 0x0a  ;
    }

    i =(unsigned int)( temperature_h<<8);
    i += temperature_l;

    DispData6 = i%10;

    j= i/10;
    DispData5 = j%10;

    j= i/100;
    DispData4 = j%10;

    j= i/1000;
    DispData3 = j%10;

    j= i/10000;
```

```
    DispData2 = j%10;

    if( DispData2 ==0)
    {
        DispData2 = 0x0a;
        if(  DispData3 ==0 )
        {
            DispData3 = 0x0a;
            if(DispData4 == 0x0)
            {
                DispData4 = 0x0a;
            }
        }
    }
}
void display_compass(void)
{
    unsigned int i, j;

    Flag.is_disp_press = 0;
    Flag.is_disp_altitude = 0;
    Flag.is_disp_compass = 1;
    Flag.is_disp_pedo = 0;
    Flag.is_disp_temp = 0;
    Flag.is_disp_weather = 0;

    i = (unsigned int)(compass_h<<8);
    i += compass_l;

    DispData6 = i%10;

    j= i/10;
    DispData5 = j%10;

    j= i/100;
    DispData4 = j%10;

    DispData1 = 0x0a;
    DispData2 = 0x0a;
    DispData3 = 0x0a;

}
void display_power_off(void)
```

```
{

    Flag.is_disp_press = 0;
    Flag.is_disp_altitude = 0;
    Flag.is_disp_compass = 0;
    Flag.is_disp_pedo = 0;
    Flag.is_disp_temp = 0;
    Flag.is_disp_weather = 0;


    DispData6 = 0x0a;
    DispData5 = 0x0a;
    DispData4 = 0x0a;

    DispData1 = 0x0a;
    DispData2 = 0x0a;
    DispData3 = 0x0a;

}
void display_compass_calibrate(void)
{


    Flag.is_disp_press = 0;
    Flag.is_disp_altitude = 0;
    Flag.is_disp_compass = 1;
    Flag.is_disp_pedo = 0;
    Flag.is_disp_temp = 0;
    Flag.is_disp_weather = 0;


    DispData6 = 0x0b;
    DispData5 = 0x0b;
    DispData4 = 0x0b;

    DispData1 = 0x0a;
    DispData2 = 0x0a;
    DispData3 = 0x0a;

}

void display_weather(void)
{
```

```
    Flag.is_disp_press = 0;
    Flag.is_disp_altitude = 0;
    Flag.is_disp_compass = 0;
    Flag.is_disp_pedo = 0;
    Flag.is_disp_temp = 0;
    Flag.is_disp_weather = 1;

    DispData1 = 0x0a;
    DispData2 = 0x0a;
    DispData3 = 0x0a;
    DispData4 = 0x0a;
    DispData5 = 0x0a;

    DispData6 = weather;
}



void send_SPC01_write_command(unsigned char command)
{
    IIC_Start();
    communicate_status = IIC_WriteByte(0x20);
    if( communicate_status == COMMUNI_SUCCEED)
    {
        communicate_status = IIC_WriteByte(command);
        if( communicate_status == COMMUNI_SUCCEED)
        {
            IIC_Stop();
        }
    }
}

void send_read_press_command(void)
{
    IIC_Start();
    communicate_status = IIC_WriteByte(0x21);
    if( communicate_status == COMMUNI_SUCCEED)
    {
        press_h = IIC_ReadByte();
        IIC_ACK();

        press_m = IIC_ReadByte();
        IIC_ACK();
```

```
            press_l = IIC_ReadByte();

            IIC_NoAck();
            IIC_Stop();
        }
}
void send_read_temperature_command(void)
{
        IIC_Start();
        communicate_status = IIC_WriteByte(0x21);
        if( communicate_status == COMMUNI_SUCCEED)
        {
            temperature_h = IIC_ReadByte();
            IIC_ACK();

            temperature_l = IIC_ReadByte();

            IIC_NoAck();
            IIC_Stop();
        }
}
void send_read_altitude_command(void)
{
        IIC_Start();
        communicate_status = IIC_WriteByte(0x21);
        if( communicate_status == COMMUNI_SUCCEED)
        {
            altitude_h = IIC_ReadByte();
            IIC_ACK();

            altitude_m = IIC_ReadByte();
            IIC_ACK();

            altitude_l = IIC_ReadByte();

            IIC_NoAck();
            IIC_Stop();
        }
}
void send_read_compass_command(void)
{
        IIC_Start();
```

```
    communicate_status = IIC_WriteByte(0x21);
    if( communicate_status == COMMUNI_SUCCEED)
    {
        compass_h = IIC_ReadByte();
        IIC_ACK();

        compass_l = IIC_ReadByte();

        IIC_NoAck();
        IIC_Stop();
    }
}
void send_read_weather_command(void)
{
    IIC_Start();
    communicate_status = IIC_WriteByte(0x21);
    if( communicate_status == COMMUNI_SUCCEED)
    {
        weather = IIC_ReadByte();

        IIC_NoAck();
        IIC_Stop();
    }
}



//=============================================

void IIC_SCL_HIGH(void)
{
    SCK=1;
}
//=============================================

void IIC_SCL_LOW(void)
{
    SCK=0;
}

void IIC_Start(void)
{
    DATA_SET = DATA_OUTPUT;
```

```
    DATA=1;
    SysDelay(DELAY100US);

    IIC_SCL_HIGH();
    SysDelay(DELAY100US);

    DATA=0;
    SysDelay(DELAY100US);

    IIC_SCL_LOW();
    SysDelay(DELAY100US);

}
//==============================================

void IIC_Stop(void)
{
    DATA_SET = DATA_OUTPUT;
    IIC_SCL_LOW();
    SysDelay(DELAY100US);

    DATA=0;
    SysDelay(DELAY100US);

    IIC_SCL_HIGH();
    SysDelay(DELAY100US);

    DATA=1;
    SysDelay(DELAY100US);
}
//==============================================

void IIC_ACK(void)
{
    DATA_SET = DATA_OUTPUT;
    SysDelay(DELAY100US);

    DATA=0;
    SysDelay(DELAY100US);

    IIC_SCL_HIGH();
    SysDelay(DELAY100US);
```

```
    IIC_SCL_LOW();
    SysDelay(DELAY100US);

}
//=============================================


void IIC_NoAck(void)
{
    DATA_SET = DATA_OUTPUT;
    SysDelay(DELAY100US);

    DATA=1;
    SysDelay(DELAY100US);

    IIC_SCL_HIGH();
    SysDelay(DELAY100US);

    IIC_SCL_LOW();
    SysDelay(DELAY100US);

}


//=============================================

unsigned char IIC_ReadByte(void)
{
    unsigned char ucValue;
    unsigned char ucIndex;

    ucValue = 0;
    DATA_SET = DATA_INPUT;
    SysDelay(DELAY100US);


    for ( ucIndex = 0; ucIndex < 8; ucIndex++ )
    {
        ucValue <<= 1;

        IIC_SCL_LOW();
        SysDelay(DELAY100US);
```

```
        IIC_SCL_HIGH();
        SysDelay(DELAY100US);

        if(DATA)  ucValue |= 1;


        SysDelay(DELAY100US);
        IIC_SCL_LOW();
        SysDelay(DELAY100US);

    }
    return ucValue;
}
//=============================================

unsigned char IIC_WriteByte( unsigned char ucData )
{
    unsigned char i;

    DATA_SET = DATA_OUTPUT;
    SysDelay(DELAY100US);
    for( i = 0; i < 8; i++ )
    {
        IIC_SCL_LOW();
        SysDelay(DELAY100US);

        if((ucData & 0x80) == 0x80)
        {
            DATA=1;
            SysDelay(DELAY100US);
        }
        else
        {
            DATA=0;
            SysDelay(DELAY100US);
        }


        IIC_SCL_HIGH();
            SysDelay(DELAY100US);
        ucData <<= 1;
        IIC_SCL_LOW();
    }
```

```
    DATA_SET = DATA_INPUT;
    SysDelay(DELAY100US);
    IIC_SCL_LOW();
    SysDelay(DELAY100US);

    IIC_SCL_HIGH();
    SysDelay(DELAY100US);


    if( DATA != 0)
    {
        IIC_SCL_LOW();
        SysDelay(DELAY100US);
        SysDelay1ms(10);
        return(COMMUNI_FAILED);
    }
    else
    {
        IIC_SCL_LOW();
        SysDelay(DELAY100US);
        return(COMMUNI_SUCCEED);
    }
}
```

Dorji Applied Technologies
A division of ***Dorji Industrial Group Co., Ltd***

Add.: Xinchenhuayuan 2, Dalangnanlu, Longhua, Baoan district, Shenzhen, China 518109
Tel: 0086-755-28156122
Fax.: 0086-755-28156133
Email: sales@dorji.com
Web: http://www.dorji.com