

```

/*****
*****
*Header File
*****
*****/

#include "msp430x22x4.h"
#include "image.h"

/*****
*****
* IO Port Define
*****
*****/

#define SDA_H      (P1OUT |=BIT7)           // P1.7
#define SDA_L      (P1OUT &=~BIT7)
#define SCLK_H     (P1OUT |=BIT6)           // P1.6
#define SCLK_L     (P1OUT &=~BIT6)
#define nCS_H      (P1OUT |=BIT5)           // P1.5
#define nCS_L      (P1OUT &=~BIT5)
#define nDC_H      (P1OUT |=BIT4)           // P1.4
#define nDC_L      (P1OUT &=~BIT4)
#define nRST_H     (P1OUT |=BIT3)           // P1.3
#define nRST_L     (P1OUT &=~BIT3)
#define R_SDA      0x80
//P1.7

/*****
*****
* Image-Display define
*****
*****/

#define PIC_BLACK   252
#define PIC_WHITE   255
#define PIC_A       1
#define PIC_B       2
#define PIC_HLINE   3
#define PIC_VLINE   4
#define PIC_C       5
#define PIC_D       6
#define PIC_E       7
#define PIC_R       8
#define PIC_Yellow   9

/*****
*****
* Globle Variable definitions
*****
*****/

```

```

unsigned char Temp_H,Temp_L;
unsigned char tempvalue;
unsigned char temp1;
unsigned char temp2;
unsigned char CRC_Byte;
unsigned char CRC_bite1,CRC_bite2,CRC_bite3,CRC_bite4;
unsigned char Byte1,Byte2,Byte3,Byte4;

/*****
*****
* MCU delay configuration
*****
*****/

void DELAY_100nS(int delaytime)                                // 30us
{
    int i,j;

    for(i=0;i<delaytime;i++)
        for(j=0;j<8;j++);
}

void DELAY_mS(int delaytime)                                    // 1ms
{
    int i,j;

    for(i=0;i<delaytime;i++)
        for(j=0;j<1600;j++);
}

void DELAY_S(int delaytime)                                     // 1s
{
    int i,j,k;

    for(i=0;i<delaytime;i++)
        for(j=0;j<1000;j++)
            for(k=0;k<1600;k++);
}

/*****
*****
*
* EPD function
*
*****
*****/

```

```

/*****
*****
* IC reset
*****
*****/

void RESET()
{
    nRST_L;
    DELAY_mS(20);
    nRST_H;
    DELAY_mS(20);
}

/*****
*****
* IC Read Busy
*****
*****/

void READBUSY()
{
    while(1)
    {
        _NOP();
        if((P1IN & 0x04)==0x04)
            break;
    }
}

/*****
*****
* 4line SPI Write Command
*****
*****/

void SPI4W_WRITECOM(unsigned char INIT_COM)
{
    unsigned char TEMPCOM;
    unsigned char scnt;
    P1DIR |= R_SDA;
    TEMPCOM=INIT_COM;
    nCS_H;
    nCS_L;
    SCLK_L;
    nDC_L;
    for(scnt=0;scnt<8;scnt++)
    {
        if(TEMPCOM&0x80)
            SDA_H;
        else
            SDA_L;
        SCLK_H;
    }
}

```

```

        SCLK_L;
        TEMPCOM=TEMPCOM<<1;
    }
    nCS_H;
}

/*****
*****
* 4line SPI Write Data
*****
*****/

void SPI4W_WRITEDATA(unsigned char INIT_DATA)
{
    unsigned char TEMPCOM;
    unsigned char scnt;
    P1DIR |= R_SDA;
    TEMPCOM=INIT_DATA;
    nCS_H;
    nCS_L;
    SCLK_L;
    nDC_H;
    for(scnt=0;scnt<8;scnt++)
    {
        if(TEMPCOM&0x80)
            SDA_H;
        else
            SDA_L;
        SCLK_H;
        SCLK_L;
        TEMPCOM=TEMPCOM<<1;
    }
    nCS_H;
}

/*****
*****
* 4line SPI Read Data
*****
*****/

unsigned char SPI4W_READDATA(void)
{
    unsigned char scnt,temp;
    P1DIR &=~ R_SDA;
    nCS_H;
    nCS_L;
    SCLK_H;
    nDC_H;
    for(scnt=0;scnt<8;scnt++)
    {
        SCLK_H;
        if(P1IN&R_SDA)

```

```

        temp=(temp<<1)|0x01;
    else
        temp=temp<<1;
        SCLK_L;
    }
    nCS_H;
    tempvalue= temp;
    return temp;
}

/*****
*****
* Temperature sensing
*****
*****/

void read_temperature(void)
{

    SPI4W_WRITECOM(0x40);
    DELAY_S(1);
    temp1=SPI4W_READDATA();
    temp2=SPI4W_READDATA();

}

/*****
*****
* Read Revision function
*****
*****/

void read_Revision()
{
    SPI4W_WRITECOM(0x70);
    Byte1=SPI4W_READDATA();//chip id
    Byte2=SPI4W_READDATA();//lut_ver[0:7]
    Byte3=SPI4W_READDATA();//lut_ver[8:15]
    Byte4=SPI4W_READDATA();//lut_ver[23:16]
}

/*****
*****
* EPD init

```

```
*****  
*****/
```

```
void INIT_JD79667()
```

```
{
```

```
    //FITI cmd.
```

```
    SPI4W_WRITECOM(0x4D);
```

```
    SPI4W_WRITEDATA(0x78);
```

```
    // datasheet user cmd.
```

```
    SPI4W_WRITECOM(0x00);
```

```
    SPI4W_WRITEDATA(0x0F);
```

```
    SPI4W_WRITEDATA(0x29);
```

```
    SPI4W_WRITECOM(0x01);
```

```
    SPI4W_WRITEDATA(0x07);
```

```
    SPI4W_WRITECOM(0x03);
```

```
    SPI4W_WRITEDATA(0x10);
```

```
    SPI4W_WRITEDATA(0x54);
```

```
    SPI4W_WRITEDATA(0x44);
```

```
    SPI4W_WRITECOM(0x06);    //47uH bst level modify
```

```
    SPI4W_WRITEDATA(0x0F);
```

```
    SPI4W_WRITEDATA(0x0A);
```

```
    SPI4W_WRITEDATA(0x2F);
```

```
    SPI4W_WRITEDATA(0x25);
```

```
    SPI4W_WRITEDATA(0x22);
```

```
    SPI4W_WRITEDATA(0x2E);
```

```
    SPI4W_WRITEDATA(0x21);
```

```
    SPI4W_WRITECOM(0x50);
```

```
    SPI4W_WRITEDATA(0x37);    //border
```

```
    SPI4W_WRITECOM(0x60);
```

```
    SPI4W_WRITEDATA(0x02);
```

```
    SPI4W_WRITEDATA(0x02);
```

```
    SPI4W_WRITECOM(0x61);
```

```
    SPI4W_WRITEDATA(0x00);
```

```
    SPI4W_WRITEDATA(0xC8);    //200
```

```
    SPI4W_WRITEDATA(0x01);
```

```
    SPI4W_WRITEDATA(0x2C);    //300
```

```

    SPI4W_WRITECOM(0xE7);
    SPI4W_WRITEDATA(0x1C);

    SPI4W_WRITECOM(0xE3);
    SPI4W_WRITEDATA(0x22);

    SPI4W_WRITECOM(0xB4);
    SPI4W_WRITEDATA(0xD0);

    SPI4W_WRITECOM(0xB5);
    SPI4W_WRITEDATA(0x03);

    SPI4W_WRITECOM(0xE9);
    SPI4W_WRITEDATA(0x01);

    SPI4W_WRITECOM(0x30); //frame rate from OTP
    SPI4W_WRITEDATA(0x08);
}

/*****
*****
* Enter DeepSleep
*****
*****/

void enterdeepsleep()
{
    SPI4W_WRITECOM(0x07); // Deep sleep
    SPI4W_WRITEDATA(0xA5);
}

/*****
*****
* Display Test Image
*****
*****/

void dis_img_red(unsigned char num)
{
    unsigned int row, col;
    unsigned int pcnt;

    SPI4W_WRITECOM(0x10); // DTML Write
    READBUSY();

```

```

    pcnt = 0;
    for(col=0; col<264; col++)
    {
        for(row=0; row<44; row++)
        {
            switch (num)
            {

                case PIC_A:

SPI4W_WRITEDATA(gImage_2D7_E5_ZITI[pcnt]);

                break;


                case PIC_WHITE:
                    SPI4W_WRITEDATA(0x55);
                    break;

                case PIC_BLACK:
                    SPI4W_WRITEDATA(0x00);
                    break;

                case PIC_R:

                    SPI4W_WRITEDATA(0xFF);
                    break;


                case PIC_Yellow:

                    SPI4W_WRITEDATA(0xAA);
                    break;

                default:
                    break;
            }
            pcnt++;
        }
    }

```

```

SPI4W_WRITECOM(0x04); // Power ON
READBUSY();
DELAY_mS(10);

SPI4W_WRITECOM(0x12); // Display Refresh
SPI4W_WRITEDATA(0x00);
DELAY_mS(10);
READBUSY();

```

```

    SPI4W_WRITECOM(0x02); // Power OFF
    SPI4W_WRITEDATA(0x00);
    READBUSY();
    DELAY_mS(20);

}

/*****
*****
*
* Main Funcation
*
*****
*****/

void main( void )
{

    WDTCTL = WDTPW + WDTHOLD; // Stop watchdog
    timer to prevent time out reset
    BCSCTL1 = CALBC1_8MHZ; // set DCO frequency
    1MHZ
    DCOCTL = CALDCO_8MHZ;

    P1DIR |=0xF8; // set P1.3~7 output

    RESET();
    DELAY_mS(20);
    read_temperature();

    RESET();
    READBUSY();
    INIT_JD79667();
    dis_img_red(PIC_WHITE);
    enterdeepsleep();
    DELAY_S(5);

    RESET();
    READBUSY();
    INIT_JD79667();
    dis_img_red(PIC_A);
    enterdeepsleep();
    DELAY_S(5);

```

