```
////Note: For the best effect, we suggest you view the program with the Source
Insight software.

//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
//  Prorgam:         EPD2.9_JD79661
//  Author:          CHENG WEI
//  Date:            2023.03.06.
//  Rev:             1.1   //Update the initialization frequency command
//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

//
//xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx



//xx Includes
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxxxxxx
#include "msp430x22x4.h"
#include "image.h"



//xx Private macro
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxx

#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



unsigned char tempvalue;
unsigned char tempvalue1;
unsigned char temp1;
unsigned char temp2;


#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

/*********************************************************************
*********
* Delay time function
*********************************************************************
********/
void DELAY_100nS(int delaytime)                          // 30us
{
    int i,j;

    for(i=0;i<delaytime;i++)
        for(j=0;j<1;j++);
}

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

    for(i=0;i<delaytime;i++)
        for(j=0;j<200;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<200;k++);
}

/*********************************************************************
*********
* Reset function
*********************************************************************
********/
void RESET()
{
    nRST_L;
    DELAY_mS(20);
    nRST_H;
    DELAY_mS(20);
}

/*********************************************************************
*********
* Read busy function
*********************************************************************
********/
void READBUSY()
{
    while(1)
    {
```

```
      _NOP();
      if((P1IN & 0x04)==0x04)
        break;
    }
}

/*********************************************************************
*********
* 4line SPI Write Command function
*********************************************************************
********/
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 function
*********************************************************************
********/
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;
    SCLK_L;
    nDC_H;
}

/*********************************************************************
*********
* 4line SPI Read Data function
*********************************************************************
********/
unsigned char SPI4W_READDATA(void)
{
    unsigned char scnt,temp;
    P1DIR &=~ R_SDA;
    nDC_H;
    nCS_H;
    nCS_L;
    SCLK_L;

    for(scnt=0;scnt<8;scnt++)
    {

      if(P1IN&R_SDA)
        temp=(temp<<1)|0x01;
      else
        temp=temp<<1;
      SCLK_H;
      SCLK_L;
    }
    nCS_H;
    tempvalue= temp;
    return temp;

}



// Temperature sensing
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxxxxxxxx
   void read_temperture(void)
   {

      unsigned char temp1,temp2;

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

      tempvalue=temp1;
      tempvalue=temp2;
```

```
   }


/*********************************************************************
*********
* Initialization function
*********************************************************************
********/


void INIT_JD79667()
{
   //FITI cmd.
   SPI4W_WRITECOM(0x4D);
   SPI4W_WRITEDATA(0x78);



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


}

/*********************************************************************
*********
* Depp-sleep function
*********************************************************************
********/
void enterdeepsleep()
{
    SPI4W_WRITECOM(0x07);  // Deep sleep
    SPI4W_WRITEDATA(0xA5);
}




/*********************************************************************
*********
* Display  picture function
*********************************************************************
********/

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



    SPI4W_WRITECOM(0x10);   // DTM1 Write
        READBUSY();
        pcnt = 0;
        for(col=0; col<384; col++)
        {
            for(row=0; row<42; row++)
```

```
{
        switch (num)
    {
        case PIC_A:

             SPI4W_WRITEDATA(gImage_2D9_168X384_1[pcnt]);

            break;
        case PIC_B:

             SPI4W_WRITEDATA(0X00);

            break;



         case PIC_VLINE:
             if(col<96)
             SPI4W_WRITEDATA(0xAA);
             else if(col<192)
             SPI4W_WRITEDATA(0xFF);
                            else if(col<288)
                            SPI4W_WRITEDATA(0x55);
             else
                            SPI4W_WRITEDATA(0x00);
             break;




        case PIC_HLINE:
             if(row<11)
             SPI4W_WRITEDATA(0xAA);
             else if(row<22)
             SPI4W_WRITEDATA(0xFF);
                            else if(row<32)
                            SPI4W_WRITEDATA(0x55);
                            else
                            SPI4W_WRITEDATA(0x00);
             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 function
************************************************************************
********/

void main( void )
{

    unsigned int i;
    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_temperture();
```

```
    RESET();
    READBUSY();
    INIT_JD79667();
    dis_img_red(PIC_WHITE);
    enterdeepsleep();
    DELAY_S5;
    _NOP();



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









}
```