```
//-------------------------------------------------------------------------------
//Prorgam:2D9_VL_296X128_1680_Debug
//Version:A0     Description:  First version      Date:2020-11-17
//Author:Hu Feng Yao
//-------------------------------------------------------------------------------
//
//Includes ------------------------------------------------------------------
#include "msp430x22x4.h"
//#include "waveform_T2530.h"
#include "image11.h"

// IO_Configuration----------------------------------------------------------

#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 LED_OFF     (P4OUT |=BIT3)                                      // P4.3
#define LED_ON      (P4OUT &=~BIT3)

// Variable definitions------------------------------------------------------
#define R_SDA                   0x80
#define DELAY_TIME         1
#define MODE1

unsigned char tempvalue;
signed char temp1,temp2;

// Display function----------------------------------------------------------
#define PIC_BLACK       252
#define PIC_BLACK       252                                          //Disply black
#define PIC_WHITE       255                                          //Disply white
#define PIC_A           1                                            //Disply char
#define PIC_B              2                                            //Disply
lebel
#define PIC_C              3                                            //Disply QR
code
#define PIC_HLINE       4                                            //Disply hline
#define PIC_VLINE          5                                            //Disply
vline
#define PIC_D                 6                                             //Disply
red
```

```
// MCU delay configuration ----------------------------------------------------

void DELAY_mS(int delaytime)
{
  int i,j;

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

void DELAY_S(int delaytime)
{
  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(10);
    nRST_H;
    DELAY_mS(10);
}

// IC read BUSY --------------------------------------------------------------

void READBUSY()
{
  while(1)
  {
    _NOP();
    if((P1IN & 0x04)==0)                                          // Read
BUSY electrica level
    break;
  }
}
// 4line SPI Write Command ----------------------------------------------------

// Enter command xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
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;
}

// Enter data
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
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;
}
```

```
// Read data
xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxxxx
 unsigned char SPI4W_READDATA1()
 {
      P1DIR &=~ R_SDA;

      unsigned char scnt,temp;
         temp=0;

     // nCS_H;
     // nCS_L;
     // SCLK_H;
      nDC_H;
      for(scnt=0;scnt<8;scnt++)
      {

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


unsigned char SPI4W_READDATA()
  {
  P1DIR &=~ R_SDA;
  unsigned char scnt,temp=0;
  nCS_L;
  nDC_H;
  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;
  }



// EPD INIT ------------------------------------------------------------------

void INIT_SSD1682()
{

    SPI4W_WRITECOM(0x01);
//Driver output control
    SPI4W_WRITEDATA(0xF9);
    SPI4W_WRITEDATA(0x00);


    SPI4W_WRITECOM(0x11);                                                    //
Data entry mode setting
    SPI4W_WRITEDATA(0x01);

    SPI4W_WRITECOM(0x44);                                                    // set
RAM x address start/end
    SPI4W_WRITEDATA(0x00);                                                   //
Start 0
    SPI4W_WRITEDATA(0x0F);                                                   // End
0x0C=12 (12+1)x8=104

    SPI4W_WRITECOM(0x45);                                                    // set
RAM y address start/end
    SPI4W_WRITEDATA(0xF9);                                                   //
Start 0xD3=211+1=212
    SPI4W_WRITEDATA(0x00);


    SPI4W_WRITECOM(0x3C);                                                    //
board
    SPI4W_WRITEDATA(0x01);                                                   //
0x01 border white  0x00 black



     SPI4W_WRITECOM(0x37);  // Waveform ID register
   SPI4W_WRITEDATA(0x40); // ByteA
   SPI4W_WRITEDATA(0x80); // ByteB        DM[7:0]
   SPI4W_WRITEDATA(0x03); // ByteC        DM[[15:8]
   SPI4W_WRITEDATA(0x0E); // ByteD        DM[[23:16]
```

```
}


// IC Sleep ------------------------------------------------------------------
void enterdeepsleep()
{
  SPI4W_WRITECOM(0x10);
  SPI4W_WRITEDATA(0x03);
  LED_OFF;
}

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

  SPI4W_WRITECOM(0x21);                                                    //
Display update control
    SPI4W_WRITEDATA(0x40);

      SPI4W_WRITECOM(0x4E);                                                  //
set RAM x address
      SPI4W_WRITEDATA(0x00);
      SPI4W_WRITECOM(0x4F);                                                  //
set RAM y address
      SPI4W_WRITEDATA(0xF9);

      SPI4W_WRITECOM(0x24);

      pcnt = 0;
      for(col=0; col<250; col++)
      {
          for(row=0; row<16; row++)
          {
              switch (num)
              {
                case PIC_A:
                  SPI4W_WRITEDATA(~gImage_BW[pcnt]);
                break;
                case PIC_B:
                    if(col%2==0)
                           SPI4W_WRITEDATA(0xAA);
                           else
                           SPI4W_WRITEDATA(0x55);
                           break;
                case PIC_C:
                //SPI4W_WRITEDATA(gImage_2[pcnt]);
                break;
```

```
                case PIC_VLINE:
                if(col<125)
                SPI4W_WRITEDATA(0x00);
                else
                SPI4W_WRITEDATA(0xff);
                break;


                case PIC_HLINE:
                if(row<7)
                SPI4W_WRITEDATA(0x00);
                else
                SPI4W_WRITEDATA(0xff);
                break;

                case PIC_WHITE:
                SPI4W_WRITEDATA(0xff);
                break;

                case PIC_BLACK:
                SPI4W_WRITEDATA(0x00);
                break;
                default:
                break;
            }
            pcnt++;
        }
    }



  SPI4W_WRITECOM(0x18);                                                    // set
RAM x address
  SPI4W_WRITEDATA(0x80);
     //Last Temperature Value of WS

  //LED_ON;
  SPI4W_WRITECOM(0x22);
  SPI4W_WRITEDATA(0xF7);                                                   //Load
LUT from MCU(0x32), Display update
  SPI4W_WRITECOM(0x20);
  DELAY_mS(1);
  READBUSY();
}


void dis_img_partial(unsigned char num,unsigned char num1)
{
```

```
  unsigned int row, col;
  unsigned int pcnt;

  SPI4W_WRITECOM(0x21);                                                    //
Display update control
    SPI4W_WRITEDATA(0x00);

      SPI4W_WRITECOM(0x4E);                                                //
set RAM x address
      SPI4W_WRITEDATA(0x00);
      SPI4W_WRITECOM(0x4F);                                                //
set RAM y address
      SPI4W_WRITEDATA(0xF9);

      SPI4W_WRITECOM(0x26);

      pcnt = 0;
      for(col=0; col<250; col++)
      {
          for(row=0; row<16; row++)
          {
              switch (num)
              {
                 case PIC_A:
              SPI4W_WRITEDATA(gImage_4[pcnt]);


            break;

            case PIC_B:
              SPI4W_WRITEDATA(~gImage_BW1[pcnt]);


            break;


                case PIC_VLINE:
                if(col<125)
                SPI4W_WRITEDATA(0x00);
                else
                SPI4W_WRITEDATA(0xff);
                break;


                case PIC_HLINE:
                if(row<7)
                SPI4W_WRITEDATA(0x00);
                else
                SPI4W_WRITEDATA(0xff);
```

```
                break;

                case PIC_WHITE:
                SPI4W_WRITEDATA(0xff);
                break;

                case PIC_BLACK:
                SPI4W_WRITEDATA(0x00);
                break;
                default:
                break;
              }
              pcnt++;
          }
      }

      SPI4W_WRITECOM(0x4E);                                                    //
set RAM x address
      SPI4W_WRITEDATA(0x00);
      SPI4W_WRITECOM(0x4F);                                                    //
set RAM y address
      SPI4W_WRITEDATA(0xF9);

      SPI4W_WRITECOM(0x24);

      pcnt = 0;
      for(col=0; col<250; col++)
      {
          for(row=0; row<16; row++)
          {
              switch (num1)
              {
                 case PIC_A:
              SPI4W_WRITEDATA(gImage_4[pcnt]);


            break;

            case PIC_B:
              SPI4W_WRITEDATA(~gImage_BW1[pcnt]);


            break;


                case PIC_VLINE:
                if(col<125)
                SPI4W_WRITEDATA(0x00);
                else
```

```
                SPI4W_WRITEDATA(0xff);
                break;


                case PIC_HLINE:
                if(row<7)
                SPI4W_WRITEDATA(0x00);
                else
                SPI4W_WRITEDATA(0xff);
                break;

                case PIC_WHITE:
                SPI4W_WRITEDATA(0xff);
                break;

                case PIC_BLACK:
                SPI4W_WRITEDATA(0x00);
                break;
                default:
                break;
              }
              pcnt++;
          }
      }


    SPI4W_WRITECOM(0x3C);                                                    //
board
    SPI4W_WRITEDATA(0x81);


  SPI4W_WRITECOM(0x18);                                                      // set
RAM x address
  SPI4W_WRITEDATA(0x80);


  //LED_ON;
  SPI4W_WRITECOM(0x22);
  SPI4W_WRITEDATA(0xFF);                                                     //Load
LUT from MCU(0x32), Display update
  SPI4W_WRITECOM(0x20);
  DELAY_mS(1);
  READBUSY();
}


//------------------------------Supply Voltage ---------------------------------

void VCI_2_3V()
```

```
{
    P3DIR &=0x00;
    P3DIR |=0x00;                                                    //2.3V
}


void VCI_2_4V()
{

    P3DIR &=0x00;
    P3DIR |= 0x02;
    P3OUT &=~BIT1;                                                   //2.4V
}


void VCI_2_5V()
{
    P3DIR &=0x00;
    P3DIR |= 0x04;
    P3OUT &=~BIT2;                                                   //2.5V
}

void VCI_3_0V()
{
    P3DIR &=0x00;
    P3DIR |= 0x01;
    P3OUT &=~BIT0;                                                   //3.0V
}


void VCI_3_6V()
{
    P3DIR &=0x00;
    P3DIR |= 0x08;
    P3OUT &=~BIT3;                                                   //3.6V
}

// Function Main-----------------------------------------------------------------

void main( void )
{

  int i=30;

  WDTCTL = WDTPW + WDTHOLD;                                          // Stop
watchdog timer to prevent time out reset
  BCSCTL1 = CALBC1_8MHZ;                                             // set
DCO frequency 12 MHZ
  DCOCTL = CALDCO_8MHZ;
```

```
  P1DIR |=0x78;                                                    // set
P1.3~7 output
  P4DIR |=0x08;                                                    // set
P4.3 output


  VCI_3_0V();
  DELAY_mS(10);



    RESET();
    SPI4W_WRITECOM(0x12);
    READBUSY();
    INIT_SSD1682();
    dis_img(PIC_WHITE);
    enterdeepsleep();
    DELAY_S(1);



    RESET();
    SPI4W_WRITECOM(0x12);
    READBUSY();
    INIT_SSD1682();
    dis_img_partial(PIC_WHITE,PIC_A);
    enterdeepsleep();
    DELAY_S(1);


    RESET();
    SPI4W_WRITECOM(0x12);
    READBUSY();
    INIT_SSD1682();
    dis_img_partial(PIC_A,PIC_B);
    enterdeepsleep();
    DELAY_S(1);



    while(1){}
}
```