



Winstar Display Co., LTD

華凌光電股份有限公司

住址: 407 台中市中清路 163 號
No.163 Chung Ching RD.,
Taichune, Taiwan, R.O.C

WEB: <http://www.winstar.com.tw>

E-mail: winstar@winstar.com.tw

Tel:886-4-24262208 Fax : 886-4-24262207



TFT Module

Application Note

(For S1D13A04)

Contents

Chapter-1. Reference Circuit

Chapter-2. Initial Data

Chapter-3. Sample Program

Chapter-4. Initial Tool For S1d13A04

(1) MCS8051 Demo Board circuit design :



2. Initial data

By using S1D13A04 as TFT module's controller, user have to set up the internal parameter before send data onto the display; The parameter include **“pixel set”**, **“frequency set”**, and **“ address set”** etc. . so Epson have to provide **“S1D13Ax Initial Tool”** for user easily to set all parameter (see Chapter 4).

3. Sample Program

The demo program is designed for MCS-8051 demo board (see fig-1), the following program can drive the TFT module of Winstar's which with S1D13A04 control board . the program have divide into 2 parts, one is original program, the other is program with short comments.

3.1 Original Program

/*Module : Main .c*/

```
#define _MAIN_C_
#include "reg52.h"           // define 8051 registers
#include <intrins.h>
#include "eps13a04.h"

static UW32 dp;
static UW32 address;

void SHOW_PICTURE1();
void SHOW_PICTURE2();
void SHOW_PICTURE3();
void SHOW_PICTURE4();

main()
{
    UB8 i,j,k,x;
    MCU_INIT();
    for(j=0;j<108;j++)
    {
        for(i=0;i<108;i++)
        {}
    }
    EPS13A04_INIT();
    for(j=0;j<108;j++)
    {
```

```

        for(i=0;i<108;i++)
        {}

while(1)
{

full_on(0x0000);

        SHOW_PICTURE();

    }
}
//=====================================================
//=====================================================
void step_color(UW16 step,UW16 end,UW16 start)
{
UB8 i;
UW16 color,j;
    address = 0x00000000;
    write_address(address);
    for(i = 0;i<240;i++)
    {
        color = start;
        for(j=0;j<320;j++)
        {
            write_address(address);
            address ++;
            DATA_DMODE;
            A04_BUS(color);
            color += step;
            if(color == end)
                color = start;
        }
    }
}
//=====================================================
void show_word(UW32 start_address,UB8 length,UB8 wide,sword,UW16 word_color)
UW16 *sword;
{
UB8 k,l;
UW16 x;

```

```

//UW16 i=0;
    address = start_address;
    write_address(address);

for(l=0;l<wide;l++)
{
for(k = 0;k<(length/16);k++)
    {
        x = *sword;
        sb_out(x>>8,word_color);
        sb_out((UB8)x,word_color);
        sword++;
    }
address+=(320-length);
}
}

//=====================================================
void sb_out(UB8 x,UW16 word_color)
{
UB8 y;
    for(y = 0;y<8;y++)
        {
            write_address(address++);
            DATA_DMODE;
            if(x&0x80)
                {
                    A04_BUS(word_color);
                }
            else
                {
                    A04_BUS(0x0000);
                }
            x<<=1;
        }
}

//=====================================================
void line_bar2()
{
UB8 i,j,l;
UW16 color;
    address = 0x00000000;

```

```

color = 0x00f0;
write_address(address);
for(i = 0;i<240;i++)
{
    for(l=0;l<16;l++)
    {
        if(color == 0x0000)
            {color = 0x000f;}
        color<<=4;
        for(j=0;j<20;j++)
        {
            write_address(address);
            address ++;
            DATA_DMODE;
            A04_BUS(color);
            }}}
//=====
void full_on(UW16 color)
{
    UB8 i;
    address = 0x00000000;
    write_address(address);
    for(i = 0;i<240;i++)
    {
        color_bar(0x0140,color);//320
    }
}
//=====
void color_bar(UW16 wide,UW16 color)
{
    UW16 j;
    _nop_();
    for(j=0;j<wide;j++)
    {
        write_address(address);
        address ++;
        DATA_DMODE;
        P1 =(UB8)  color;
        P3 =  color >> 8;
        CS_ON;
        CS_OFF;
    }
}

```

```

        }

_nop_();
    }

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

void SHOW_PICTURE1()
{
    UB8 i;
    UW16 j;
    dp = 0x0000;
    address = 0x00000000;
    write_address(address);
    for(i = 0;i<120;i++)
    {
        CS_OFF;
        for(j=0;j<160;j++)
        {
            write_address(address);
            address ++;
            DATA_DMODE;
            P3 = PIC1[dp];
            dp++;
            P1 = PIC1[dp];
            dp++;
            CS_ON;
            _nop_();
            _nop_();
            CS_OFF;
        }
        address +=160;
    }
}

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

void SHOW_PICTURE2()
{
    UB8 i;
    UW16 j;
    dp = 0x0000;
    address = 160;

```

```

write_address(address);
for(i = 0;i<120;i++)
{
    CS_OFF;
    for(j=0;j<160;j++)
    {
        write_address(address);
        address ++;
        DATA_DMODE;
        P3 = PIC1[dp];
        dp++;
        P1 = PIC1[dp];
        dp++;
        CS_ON;
        _nop_();
        _nop_();
        CS_OFF;
    }
    address +=160;
}
}

//=====================================================
//=====================================================
void SHOW_PICTURE3()
{
    UB8 i;
    UW16 j;
    dp = 0x0000;
    address = 38400;
    write_address(address);
    for(i = 0;i<120;i++)
    {
        CS_OFF;
        for(j=0;j<160;j++)
        {
            write_address(address);
            address ++;
            DATA_DMODE;
            P3 = PIC1[dp];
            dp++;
            P1 = PIC1[dp];

```

```

        dp++;
        CS_ON;
        _nop_();
        _nop_();
        CS_OFF;
    }
    address +=160;
}
}

//=====================================================
//=====================================================
void SHOW_PICTURE4()
{
    UB8 i;
    UW16 j;
    dp = 0x0000;
    address = 38560;
    write_address(address);
    for(i = 0;i<120;i++)
    {
        CS_OFF;
        for(j=0;j<160;j++)
        {
            write_address(address);
            address ++;
            DATA_DMODE;
            P3 = PIC1[dp];
            dp++;
            P1 = PIC1[dp];
            dp++;
            CS_ON;
            _nop_();
            _nop_();
            CS_OFF;
        }
        address +=160;
    }
}

//=====================================================
//  INIT FUNCTION
//=====================================================

```

```

void MCU_INIT()
{
    EA  = 0;

    P0 = 0xff;

    P1 = 0x00;
    P2 = 0x00;
    P3 = 0x00;

    P0 = 0x00;

}
//=====================================================
void delay()
{
    UW16 i,j,k;
    for(k=0;k<10;k++)
    {
        for(j=0;j<200;j++)
        {
            for(i=0;i<200;i++)
            {}
        }
    }
}

//=====================================================
//
void SHOW_PICTURE()
{
    SHOW_PICTURE1();
    SHOW_PICTURE3();
    SHOW_PICTURE4();
    SHOW_PICTURE2();
    delay();//
    delay();//
    delay();//
    delay();//
    delay();//
}

```

```

    delay();//
}
//=====================================================
void A04_BUS(UW16 x)
{
P1 =(UB8) x;
P3 = x >> 8;
CS_ON;
CS_OFF;
}

```

/*Module Eps13a04.c*/(Write initial)

```

#define    _EPS13A04_C_
#include    "eps13a04.h"
#include    "a04ini.h"
#include    <intrins.h>
//=====================================================
//=====================================================
void EPS13A04_INIT()
{
UB8data y;
for(y=0;y<29;y++)
{
    write_command(S1D_REGS[y].Index,S1D_REGS[y].Value);

}
}
//=====================================================
void write_command(UB8 command,UW32 information)
{
CS_OFF;
COMMAND_AMODE;
P2 = 0x00;
P3 = 0x00;
P1 = command>>1;
COMMAND_DMODE;
A04_BUS(information >>16 );

A04_BUS((UW16)information);

}

```

```
//=====
/*void write_data(UW16 information)
{
CS_OFF;
DBUS;
DATA_MODE;
A04_BUS(information);
CS_ON;
_nop_();
_nop_();
CS_OFF;
}*/
//=====
void write_address(UW32 information)
{
    UW16 x;
    DATA_AMODE;
    P1 =(UB8)information;
    x=information;
    x=x>>8;
    P3 =(UB8) x;
    if(information& 0x00010000)
        P2 =1;
    else
        P2 =0;
}
```

/*a04ini.h*/(Initial Register Set)

```
struct easy code S1D_REGS[]=
{
    { 0x14,          0x00000000 },    /* Power Save Configuration Register
*/ \
    { 0x64,          0x28D70000 },    /* GPIO Status and Control Register
*/ \
    { 0x04,          0x00000000 },    /* Memory Clock Configuration Register
*/ \
    { 0x08,          0x00000042 },    /* Pixel Clock Configuration Register
*/ \
    { 0x0C,          0x00000061 },    /* Panel Type and MOD Rate Register
*/ \
    { 0x10,          0x00000010 },    /* Display Settings Register
```

```

*/ \
    { 0x20,          0x00000037 }, /* Horizontal Total Register
*/ \
    { 0x24,          0x00000027 }, /* Horizontal Display Period Register
*/ \
    { 0x28,          0x0000000F }, /* Horizontal Display Period Start Position
Register */ \
    { 0x2C,          0x00870156 }, /* FPLINE Register
*/ \
    { 0x30,          0x00000105 }, /* Vertical Total Register
*/ \
    { 0x34,          0x000000EF }, /* Vertical Display Period Register
*/ \
    { 0x38,          0x0000000D }, /* Vertical Display Period Start Position
Register */ \
    { 0x3C,          0x00800000 }, /* FPFRAME Register
*/ \
    { 0x40,          0x00000000 }, /* Main Window Display Start Address Register
*/ \
    { 0x44,          0x000000A0 }, /* Main Window Line Address Offset Register
*/ \
    { 0x50,          0x00000000 }, /* PIP+ Window Display Start Address Register
*/ \
    { 0x54,          0x000000A0 }, /* PIP+ Window Line Address Offset Register
*/ \
    { 0x58,          0x00000000 }, /* PIP+ Window X Positions Register
*/ \
    { 0x5C,          0x00000000 }, /* PIP+ Window Y Positions Register
*/ \
    { 0x60,          0x00000000 }, /* Special Purpose Register
*/ \
    { 0x70,          0x00000000 }, /* PWM Clock Configuration Register
*/ \
    { 0x74,          0x00000000 }, /* PWMOUT Duty Cycle Register
*/ \
    { 0x80,          0x00000000 }, /* Scratch Pad A Register
*/ \
    { 0x84,          0x00000000 }, /* Scratch Pad B Register
*/ \
    { 0x88,          0x00000000 }, /* Scratch Pad C Register
*/ \

```

```

        { S1D_REGDELAYON,    0x00000032 },    /* LCD Panel Power On Delay (in ms)
*/ \
        { 0x64,              0x28D70001 },    /* GPIO Status and Control Register
*/ \
        { 0x14,              0x00000000 }     /* Power Save Configuration Register
*/ \
    };

```

/*eps13a04.h*/

```

#ifndef _eps13a04_h_
#define _eps13a04_h_
#include "reg52.h"
//=====
#define S1D_REGDELAYON                0xFE
//=====

sbit E2P_A18    =    P3^2;
sbit E2P_A17    =    P3^1;
sbit E2P_A16    =    P3^0;
sbit E2P_OE     =    P3^7;
sbit E2P_WE     =    P3^6;

#define EXTERN          extern
#define ON               1
#define OFF              (!ON)
// contral function pin set
#define ADDRESS_BUS      0x10
#define DATA_BUS        0x20
#define A04_CS           0x40
#define EEPROM_BUS       0x01
#define S1D_BHE           0x01    //16 bit data
#define S1D_WE            0x02    //write mode set "0"
#define S1D_RD            0x04    //read mode set "0"
#define S1D_MR            0x08    //0:command 1:display data
#define S1D_A0            0x10    //address bit "0"
//=====
#define DATA_AMODE       (P0 = S1D_RD|ADDRESS_BUS|S1D_MR)
#define DATA_DMODE       (P0 = S1D_RD|DATA_BUS|S1D_MR)
#define COMMAND_AMODE     (P0 = S1D_RD|ADDRESS_BUS&(~S1D_MR))
#define COMMAND_DMODE     (P0 = S1D_RD|DATA_BUS&(~S1D_MR))

```

```
#define CS_ON          (P0 |= A04_CS)
#define CS_OFF         (P0 &=(~A04_CS))
```

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

```
#define address_base    0x02
#define picture1        0x00
#define picture2        (picture1 + address_base)
#define picture3        (picture2 + address_base)
#define picture4        (picture3 + address_base)
#define picture5        (picture4 + address_base)
#define picture6        (picture5 + address_base)
#define picture7        (picture6 + address_base)
#define picture8        (picture7 + address_base)
```

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

```
typedef bit            BOOL;        /* Boolean */
typedef unsigned char   UB8;        /* 8-bit data */
typedef unsigned short int UW16;    /* 16-bit data */
typedef unsigned long   UW32;       /* 32-bit data */
typedef signed char     SB8;        /* 8-bit data */
typedef signed short int SW16;      /* 16-bit data */
typedef signed long     SW32;       /* 32-bit data */
```

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

```
struct easy
```

```
{
    UB8    Index;
    UW32 Value;
};
```

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

```
====
```

```
void MCU_INIT(void);
void delay();
void line_bar2();
EXTERN void step_color(UW16,UW16,UW16);
EXTERN void SHOW_PICTURE();
EXTERN void write_address(UW32);
EXTERN void write_command(UB8,UW32);
EXTERN void EPS13A04_INIT(void);
EXTERN void A04_BUS(UW16);
EXTERN UB8 code PIC1[];
EXTERN void color_bar(UW16,UW16);
EXTERN void full_on(UW16);
```

```

EXTERN UB8 code se[];
EXTERN UB8 code se2[];
void show_word(UW32,UB8,UB8,UW16 *,UW16);
void sb_out(UB8,UW16);
#endif

```

3.2 Program function of Sub-Routine

3.2.1 Declaration of Data Type

```

typedef bit          BOOL;          /* Boolean */
typedef unsigned char UB8;          /* 8-bit data */
typedef unsigned short int UW16;    /* 16-bit data */
typedef unsigned long UW32;         /* 32-bit data */
typedef signed char   SB8;          /* 8-bit data */
typedef signed short int SW16;      /* 16-bit data */
typedef signed long   SW32;         /* 32-bit data */

```

3.2.2 EPS13A04_INIT

Prototype :

```
void EPS13A04_INIT()
```

Parameters :

None

Returns:

Description:

To initialize the S1D13A04 module.

3.2.3 Write Command

Prototype :

```
void write_command(UB8 command,UW32 information)
```

Parameters :

UB8 command //13a04 Register Address

UW32 information //13a04 Register 32bit data

Returns:

Description:

Write 13A04 initial code

3.2.4 Write Address (Set Cursor address)

Prototype :

```
void write_address(UW32 information)
```

Parameters :

```
UW32  information    //32bit Address data
```

Returns:

Description:

Set Display Ram Address

3.2.5 Set up Color

Prototype :

```
void step_color(UW16 step,UW16 end,UW16 start)
```

Parameters :

```
UW16  step
```

```
UW16  end    //End color
```

```
UW16  start  //Start color
```

Returns:

Description:

Change color while being progressive

3.2.6 Show Word

Prototype :

```
void show_word(UW32 start_address,UB8 length,UB8 wide,sword,  
               UW16 word_color)
```

Parameters :

```
UW32  start_address  //Picture start address
```

```
UB8   length         //Picture length
```

```
UB8   wide           //Picture wide
```

```
UW16  *sword         // Picture data
```

```
UW16  word_color
```

Returns:

Description:

Show Character picture

3.2.7 Line Bar (For vertical line display)

Prototype :

```
void line_bar2()
```

Parameters : None

Returns:

Description:

RGB color bar

3.2.8 Full On (For full screen display)

Prototype :

void full_on(UW16 color)

Parameters :

UW16 color Display color

Returns:

Description:

One color full of the display

3.2.9 Show Picture

Prototype :

void SHOW_PICTURE()

Parameters :

Returns:

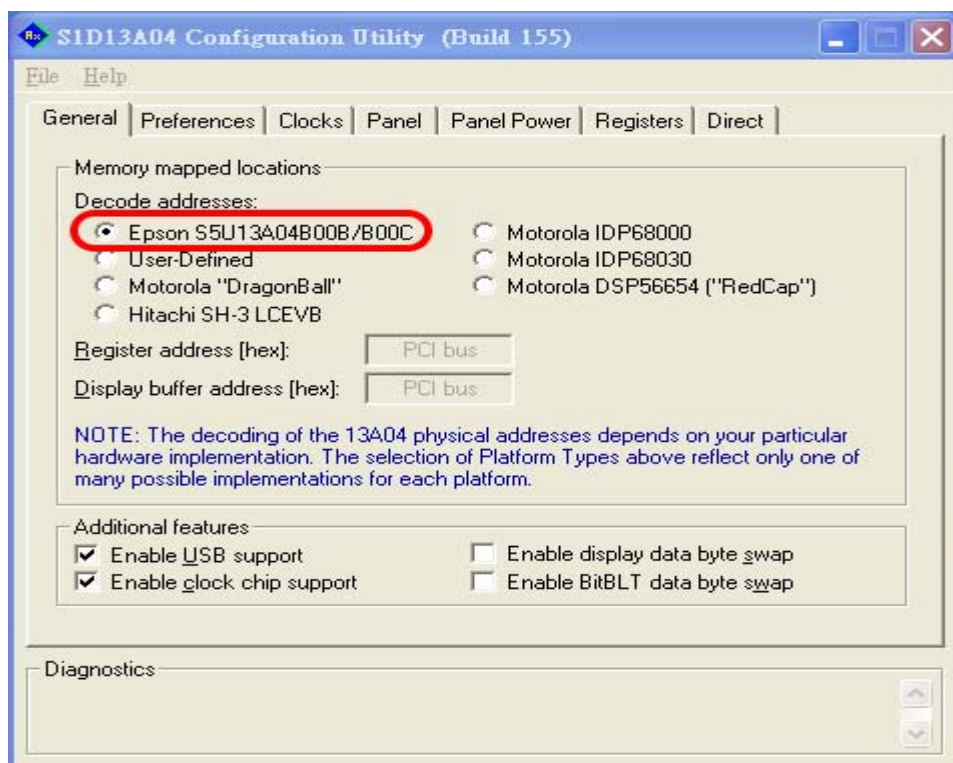
Description:

140*220 65K Color Picture

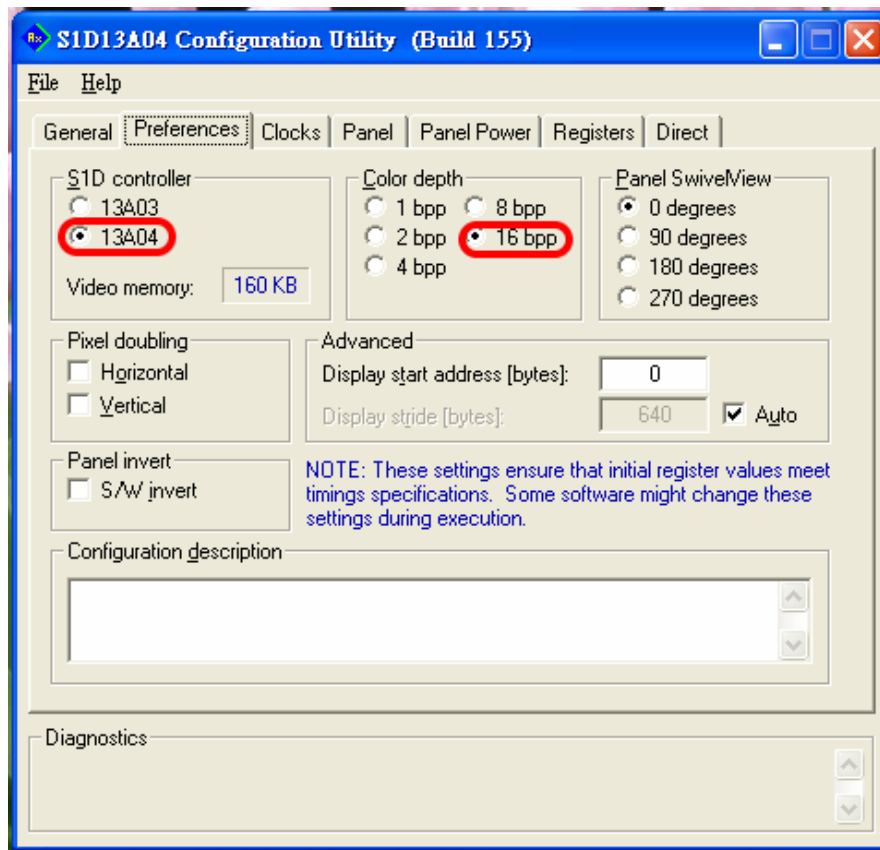
4. Initial tool for S1D13A04

(1)Download “13A04cfg.exe”

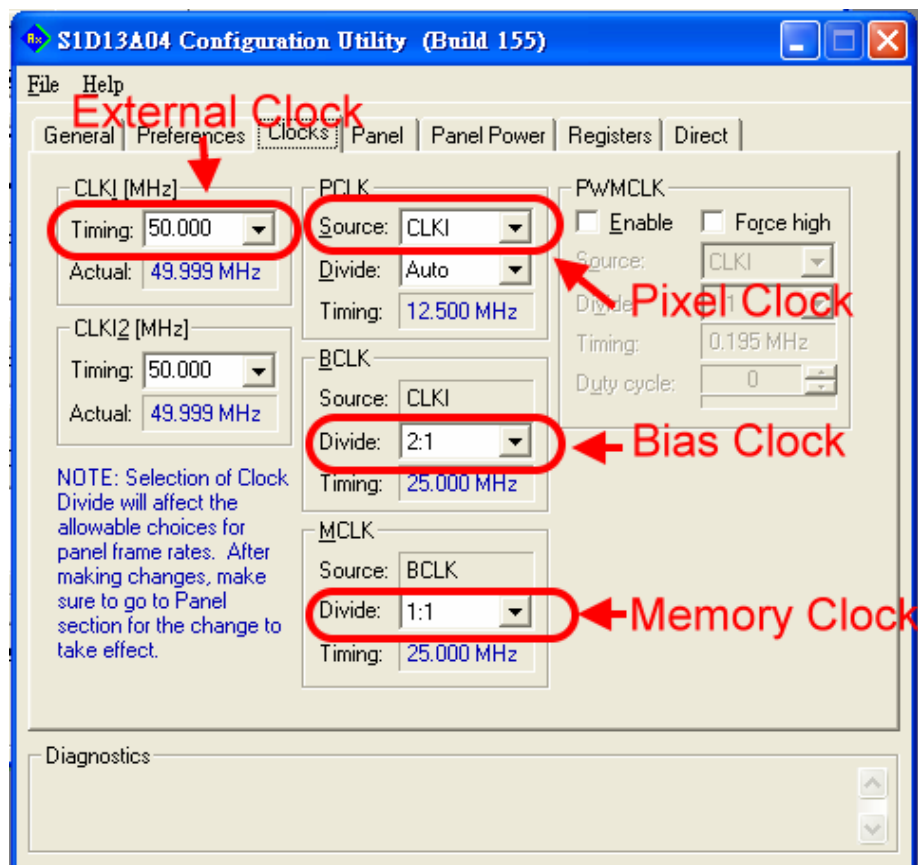
(2)Decode addresses →EPSON 13A04



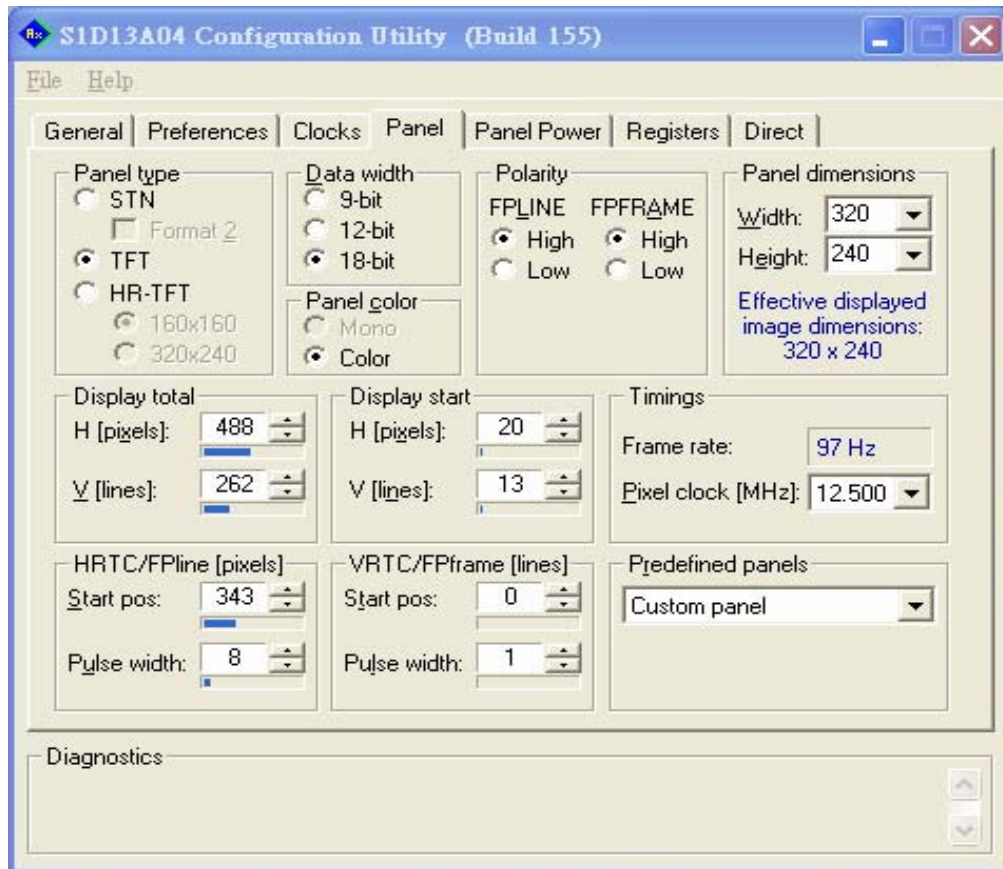
(3) S1D Controller → 13A04 : Color depth 16bpp



(4) Clocks Set

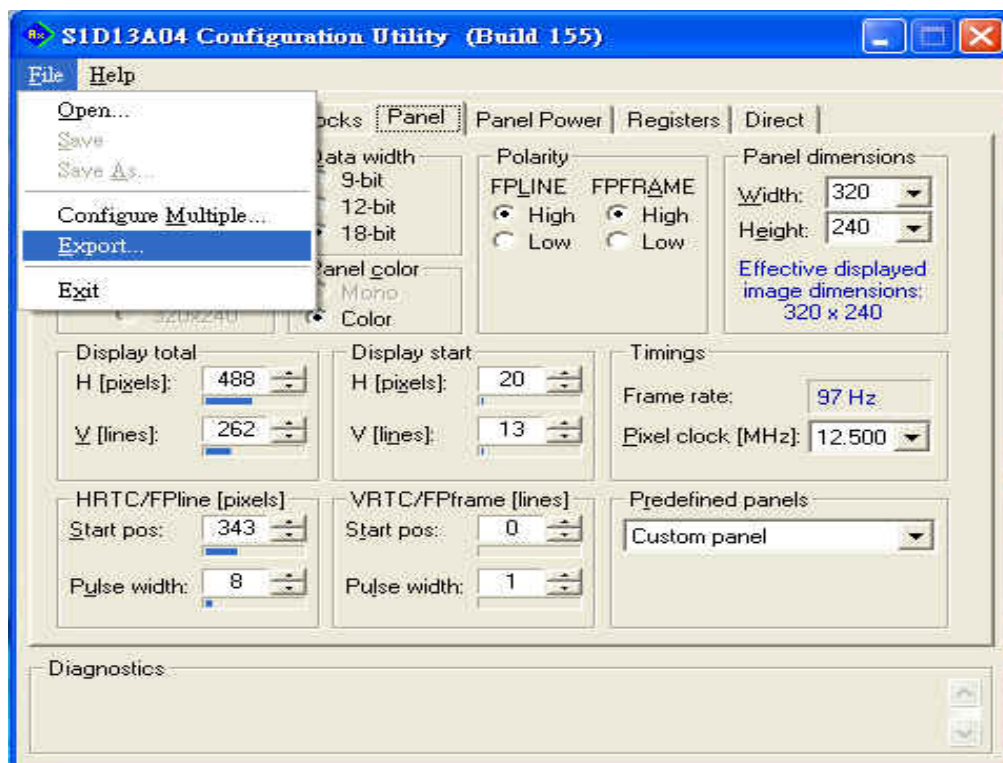


(5)Panel Data Set

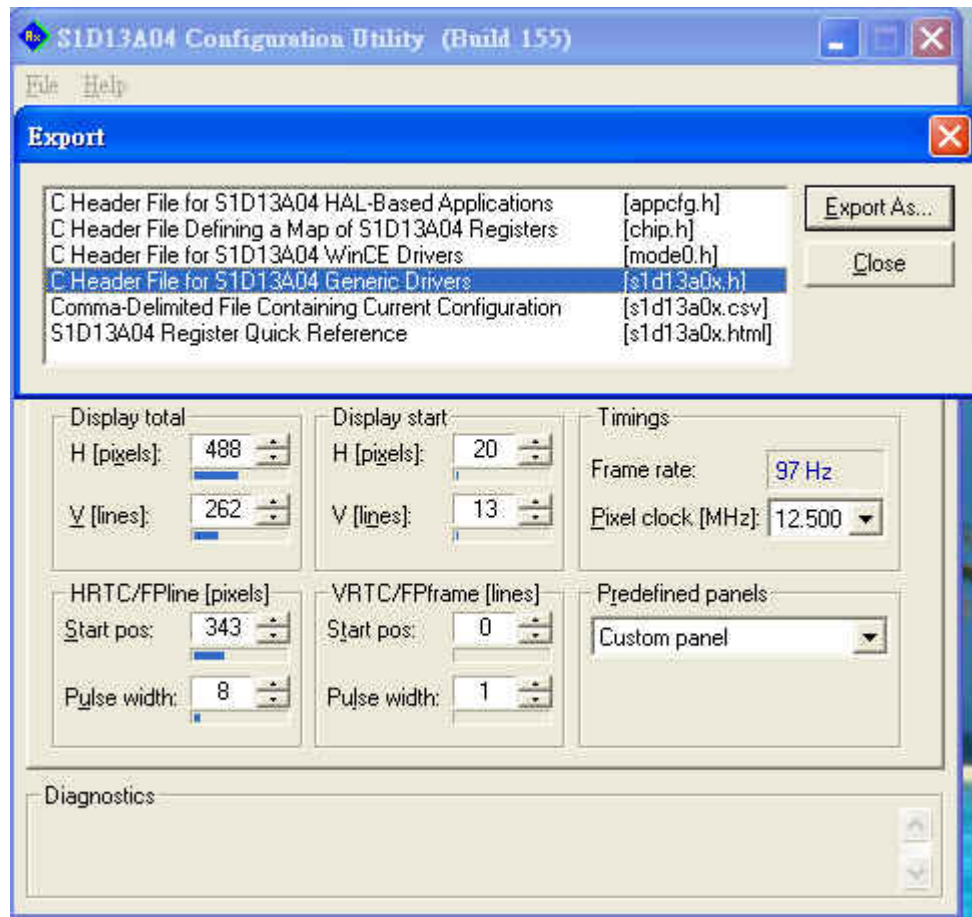


Note: Other settlement does not change

(6)Export Initial Register



(7)Select s1d13a0x.h Export



(8)Export file eps13a04.h .

(9)It finishes to establish S1D13A04 Initial Register.