Keywords: Pixel clock speed, Solomon SSD1963

Pixel clock setup for Solomon SSD1963 Display Controller

Introduction

This document explains how to configure different pixel clock speed for Solomon SSD1963 Display Controller IC

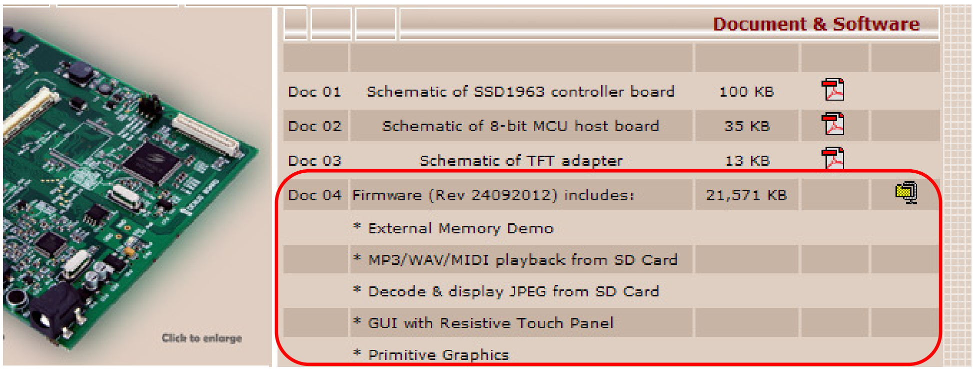
Hardware

List of components:

- SSD1963 Evaluation Kit Ultima Rev4.1
- Microchip PIC32 USB Start Kit
- 7" WVGA 800x480 TFT screen
- The workstation for development is a Pentium PC Dual Core E2160 running Windows XP SP3

Software

The demo application was developed under Microchip MPLAB version 8.83 with C32 compiler v2.20. Source code is available from our web site in the main page of SSD1963 EVK Ultima R4.1 at Doc04 (http://www.techtoys.com.hk/Displays/Solomon%20SSD1963%20EVK%20R4_1/SSD1963%20EVK%20Ultima%20R4_1.htm)



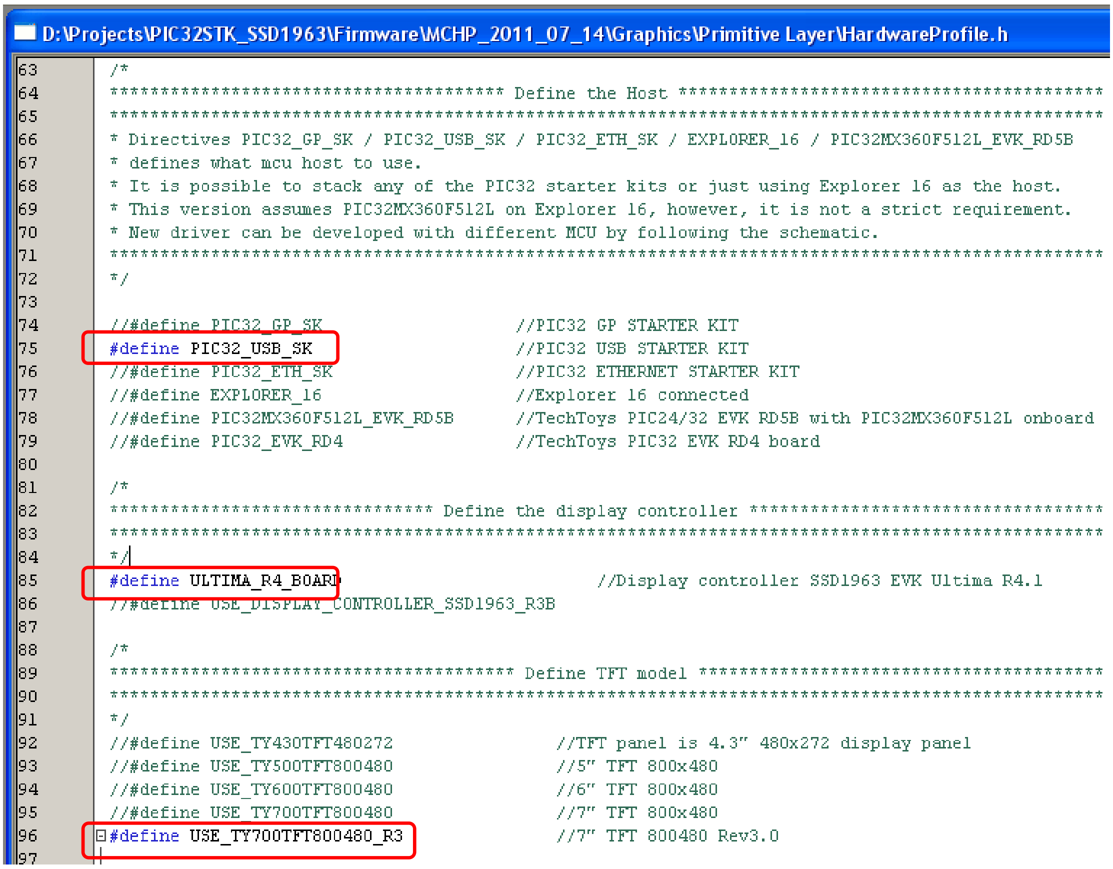
Document & Software

Doc 01	Schematic of SSD1963 controller board	100 KB		
Doc 02	Schematic of 8-bit MCU host board	35 KB		
Doc 03	Schematic of TFT adapter	13 KB		
Doc 04	Firmware (Rev 24092012) includes:	21,571 KB		
	* External Memory Demo			
	* MP3/WAV/MIDI playback from SD Card			
	* Decode & display JPEG from SD Card			
	* GUI with Resistive Touch Panel			
	* Primitive Graphics			

Figure 1 Location of source code

The project for this note is located at the directory ..\Graphics\Primitive Layer. There are three MCU variants PIC32MX360F512L , PIC32MX460F512L, & PIC32MX795F512L available. It is PIC32MX460F512L on PIC32 USB Starter Kit.

The options as shown in Figure 2(*HardwareProfile.h*)were chosen for our particular hardware combination.



D:\Projects\PIC32STK_SSD1963\Firmware\MCHP_2011_07_14\Graphics\Primitive Layer\HardwareProfile.h

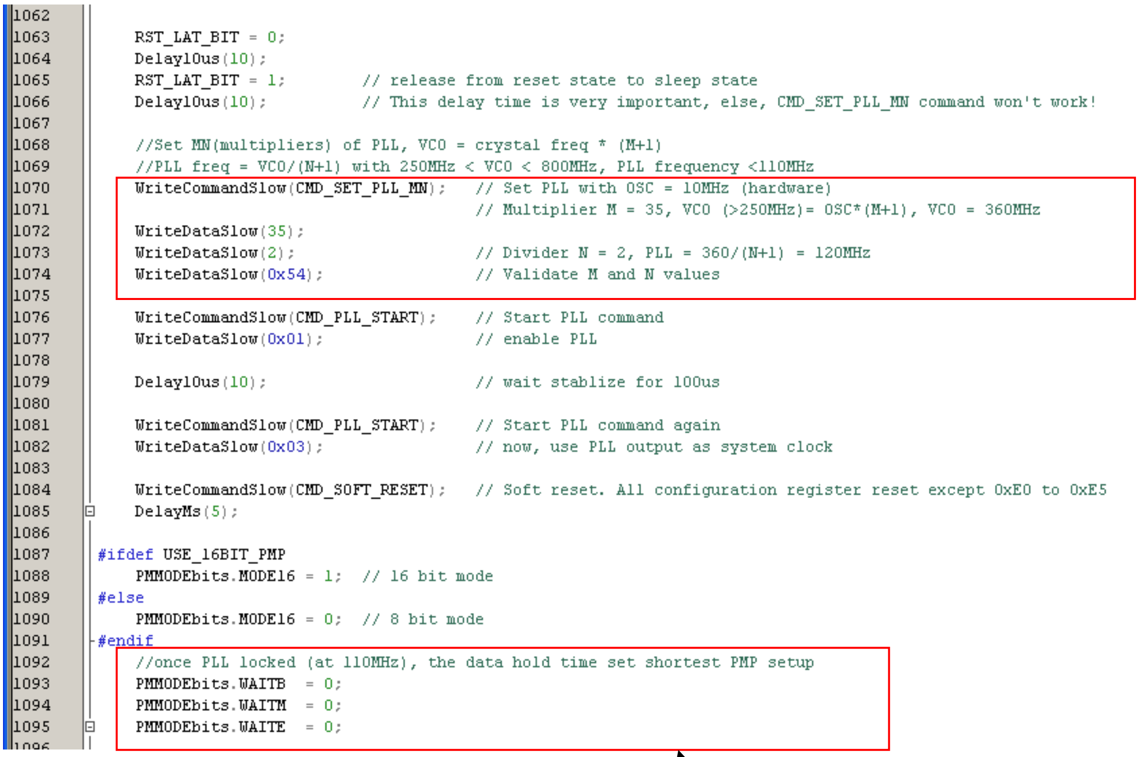
```
/*
***************************************** Define the Host *******************************************
*****************************************************************************************************
* Directives PIC32_GP_SK / PIC32_USB_SK / PIC32_ETH_SK / EXPLORER_16 / PIC32MX360F512L_EVK_RD5B
* defines what mcu host to use.
* It is possible to stack any of the PIC32 starter kits or just using Explorer 16 as the host.
* This version assumes PIC32MX360F512L on Explorer 16, however, it is not a strict requirement.
* New driver can be developed with different MCU by following the schematic.
*****************************************************************************************************
*/

//#define PIC32_GP_SK                       //PIC32 GP STARTER KIT
#define PIC32_USB_SK                        //PIC32 USB STARTER KIT
//#define PIC32_ETH_SK                      //PIC32 ETHERNET STARTER KIT
//#define EXPLORER_16                       //Explorer 16 connected
//#define PIC32MX360F512L_EVK_RD5B          //TechToys PIC24/32 EVK RD5B with PIC32MX360F512L onboard
//#define PIC32_EVK_RD4                     //TechToys PIC32 EVK RD4 board

/*
********************************** Define the display controller ************************************
*****************************************************************************************************
*/
#define ULTIMA_R4_BOARD                              //Display controller SSD1963 EVK Ultima R4.1
//#define USE_DISPLAY_CONTROLLER_SSD1963_R3B

/*
****************************************** Define TFT model *****************************************
*****************************************************************************************************
*/
//#define USE_TY430TFT480272                    //TFT panel is 4.3" 480x272 display panel
//#define USE_TY500TFT800480                    //5" TFT 800x480
//#define USE_TY600TFT800480                    //6" TFT 800x480
//#define USE_TY700TFT800480                    //7" TFT 800x480
#define USE_TY700TFT800480_R3                   //7" TFT 800480 Rev3.0
```

Figure 2 Options in HardwareProfile.h

From the project workspace, open the file SSD1963.c.

Browse to ResetDevice(void).

1. It is important to notice that, before the SSD1963 is configured to run at any higher speed, it is running at 10MHz which is exactly the crystal frequency in hardware. Therefore, it is not possible to write to SSD1963 at any speed higher than 10MHz before the PLL is locked. That's why the function WriteDataSlow() is used to match the speed of 10MHz. Figure 3 shows an extract from the source code SSD1963.c.



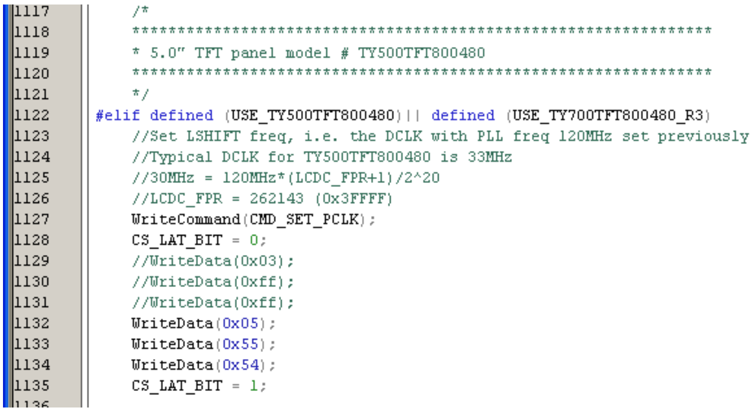
```
    RST_LAT_BIT = 0;
    Delay10us(10);
    RST_LAT_BIT = 1;        // release from reset state to sleep state
    Delay10us(10);          // This delay time is very important, else, CMD_SET_PLL_MN command won't work!

    //Set MN(multipliers) of PLL, VCO = crystal freq * (M+1)
    //PLL freq = VCO/(N+1) with 250MHz < VCO < 800MHz, PLL frequency <110MHz
    WriteCommandSlow(CMD_SET_PLL_MN);   // Set PLL with OSC = 10MHz (hardware)
                                        // Multiplier M = 35, VCO (>250MHz)= OSC*(M+1), VCO = 360MHz
    WriteDataSlow(35);
    WriteDataSlow(2);                   // Divider N = 2, PLL = 360/(N+1) = 120MHz
    WriteDataSlow(0x54);                // Validate M and N values

    WriteCommandSlow(CMD_PLL_START);    // Start PLL command
    WriteDataSlow(0x01);                // enable PLL

    Delay10us(10);                      // wait stablize for 100us

    WriteCommandSlow(CMD_PLL_START);    // Start PLL command again
    WriteDataSlow(0x03);                // now, use PLL output as system clock

    WriteCommandSlow(CMD_SOFT_RESET);   // Soft reset. All configuration register reset except 0xE0 to 0xE5
    DelayMs(5);

#ifdef USE_16BIT_PMP
    PMMODEbits.MODE16 = 1;  // 16 bit mode
#else
    PMMODEbits.MODE16 = 0;  // 8 bit mode
#endif
    //once PLL locked (at 110MHz), the data hold time set shortest PMP setup
    PMMODEbits.WAITB  = 0;
    PMMODEbits.WAITM  = 0;
    PMMODEbits.WAITE  = 0;
```

Figure 3   An extract from SSD1963.c in ResetDevice(void)

After PLL has been locked at 120MHz, we may use a higher writing speed. For Microchip PIC32, it is possible to set PMP waiting periods at the lowest values.

OK, now we have a SSD1963 running at 120MHz.

The pixel clock is divided from the master clock with divisors at register LCDC_FPR. Down the function ResetDevice(void) you may see the configuration for 5" or 7" TFT panels as below (Figure 4).



```
    /*
    ****************************************************************
    * 5.0" TFT panel model # TY500TFT800480
    ****************************************************************
    */
#elif defined (USE_TY500TFT800480)|| defined (USE_TY700TFT800480_R3)
    //Set LSHIFT freq, i.e. the DCLK with PLL freq 120MHz set previously
    //Typical DCLK for TY500TFT800480 is 33MHz
    //30MHz = 120MHz*(LCDC_FPR+1)/2^20
    //LCDC_FPR = 262143 (0x3FFFF)
    WriteCommand(CMD_SET_PCLK);
    CS_LAT_BIT = 0;
    //WriteData(0x03);
    //WriteData(0xff);
    //WriteData(0xff);
    WriteData(0x05);
    WriteData(0x55);
    WriteData(0x54);
    CS_LAT_BIT = 1;
```

Figure 4   Register LCDC_FPR for PCLK speed

The equation to obtain LCDC_FPR is:

PCLK = Master Clock * (LCDC_FPR+1)/2^20.

If we need a PCLK at 40MHz, the value of LCDC_FPR is calculated as:

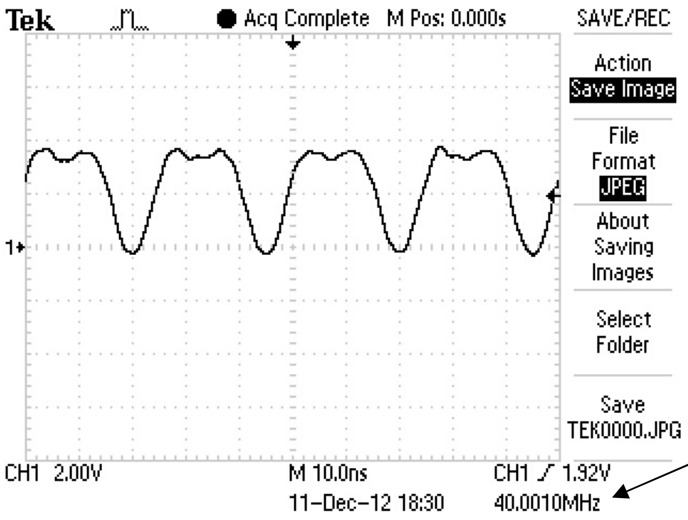
LCDC_FPR = 2^20*40/120 – 1

= 349524 (0x55554).

Similar, the value of LCDC_FPR for PCLK@30MHz = 2^20*30/120 – 1

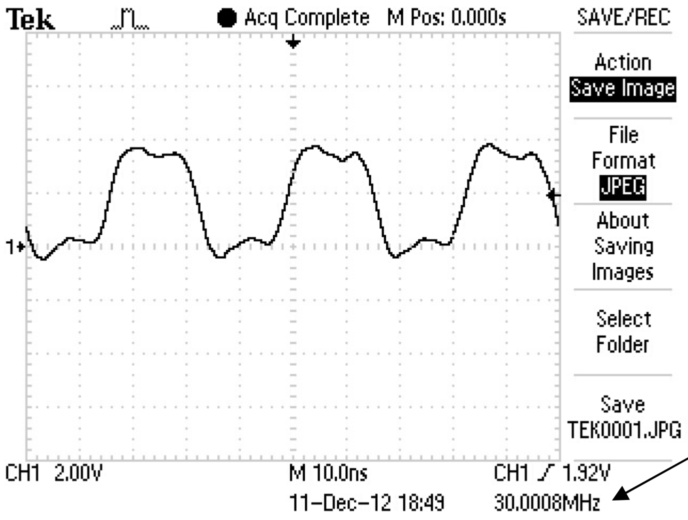
= 262143 (0x3FFFF)

Figures below show the waveform of PCLK output from SSD1963.



```
WriteCommand(CMD_SET_PCLK);

CS_LAT_BIT = 0;

WriteData(0x05);

WriteData(0x55);

WriteData(0x54);

CS_LAT_BIT = 1;
```

Figure 5 Set LCDC_FPR a value of 0x55554 for PCLK at 40MHz



```
WriteCommand(CMD_SET_PCLK);

CS_LAT_BIT = 0;

WriteData(0x03);

WriteData(0xFF);

WriteData(0xFF);

CS_LAT_BIT = 1;
```

Figure 6 Set LCDC_FPR a value of 0x3FFFF for PCLK at 30MHz

This page is empty