Keywords: Tearing Effect, Solomon SSD1963, Double Buffering

Tearing Effect with Solomon SSD1963 Display Controller

Introduction

This document explains how the Tearing Effect Signal (TE) of Solomon SSD1963 Display Controller IC can be used to void display flicker and its limitation. A demo application is described here to update a 5" WVGA TFT screen of resolution 800x480 at 50 frames per second driven by Microchip 32-bit MCU PIC32MX360F512L. Timing analysis has been measured by a logic analyzer to learn the relationship between the blank period of display and the time required to render an image if tearing effect is to be avoided.

Hardware

The following hardware (Figure 1) was used to aid in development.

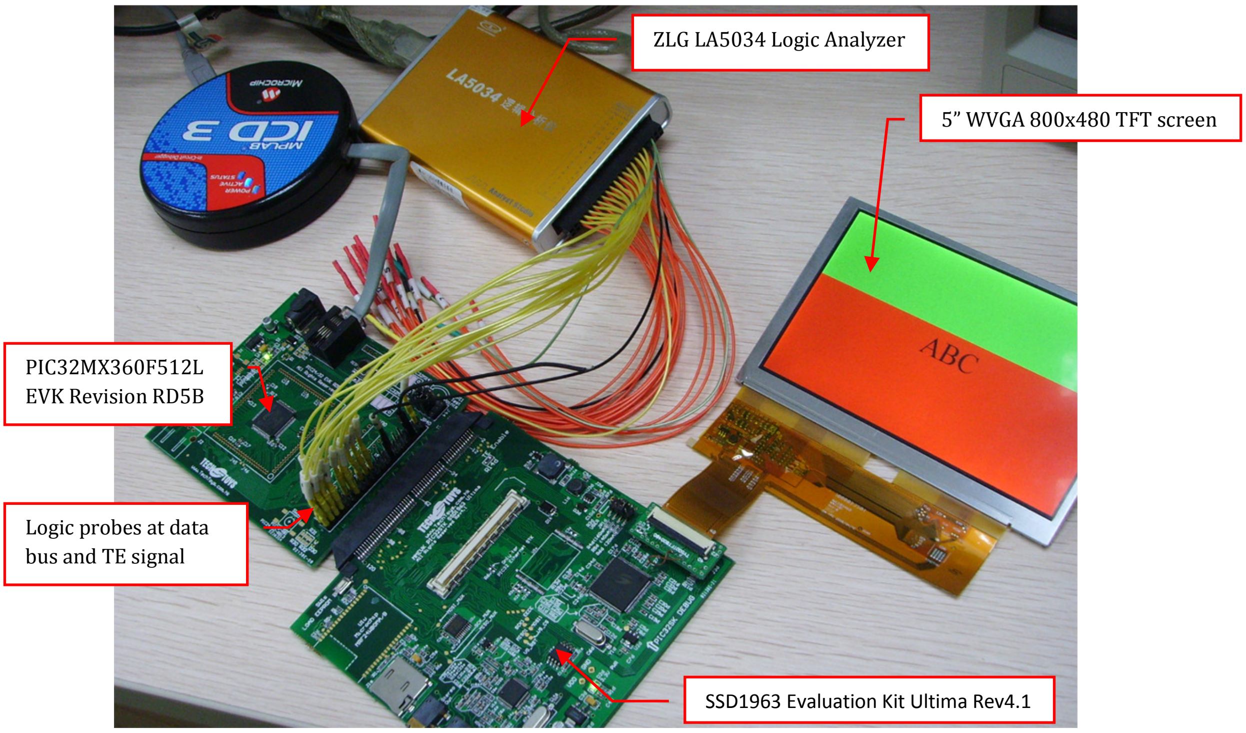

ZLG LA5034 Logic Analyzer
5" WVGA 800x480 TFT screen
PIC32MX360F512L EVK Revision RD5B
Logic probes at data bus and TE signal
SSD1963 Evaluation Kit Ultima Rev4.1

Figure 1 Hardware

List of components:

- SSD1963 Evaluation Kit Ultima Rev4.1
- PIC32MX360F512L EVK Revision RD5B
- 5" WVGA 800x480 TFT screen
- Microchip ICD3
- ZLG LA5034 Logic Analyzer
- The workstation for development is a Pentium PC Dual Core E2160 running Windows XP SP3

Full schematics of the SSD1963 Evaluation Kit Ultima[i] and PIC32MX360F512LEVK RD5B[ii] are available from our web site by following the hyperlink in End Notes. An extract from the schematic for the MCU-SSD1963 interface is shown in Figure 2. The signals required for our demo are RESET#, CS#, DC, RD#, WR#, TE, and D[15:0]. A spared pin RB1 of the MCU is used as an output for timing analysis. This pin is not shown in the diagram here.

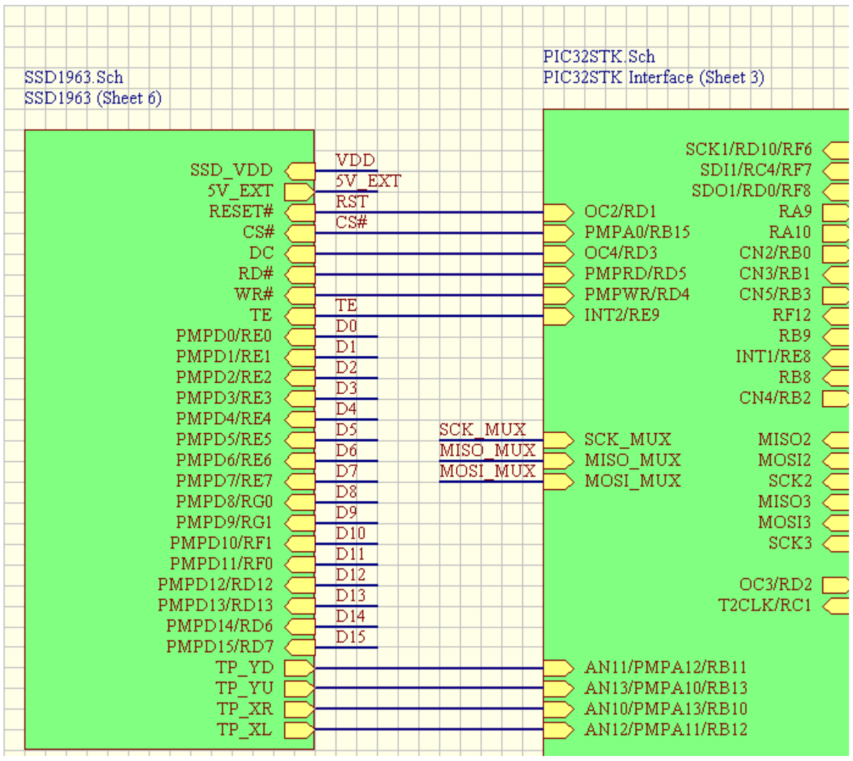


Figure 2 MCU – SSD1963 Interface

A patch resistor is required at R1 4d (TE Enable) to bridge the TE signal from SSD1963 to INT2/RE9 pin of the MCU. A 0 Ohm chip resistor of 0805 package is good enough for this purpose.

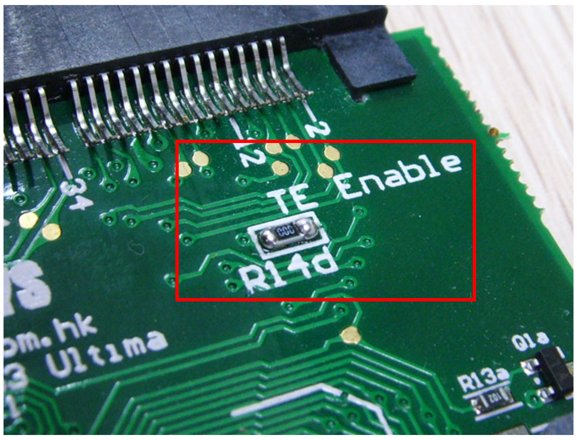
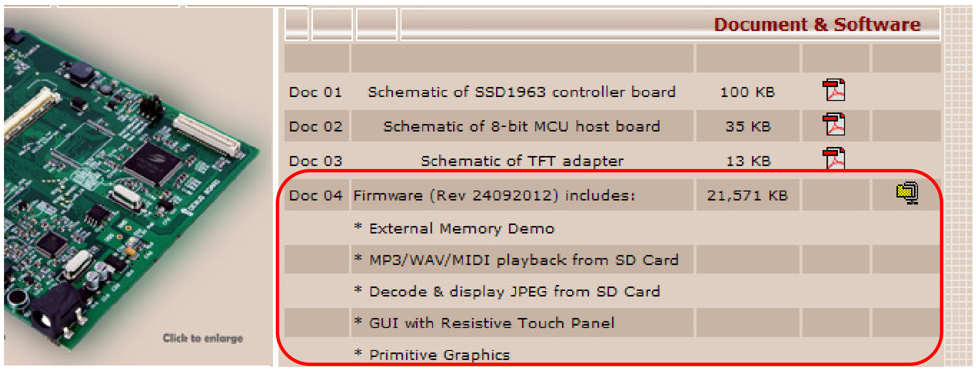


Figure 3 A patch resistor is required at R1 4d

Software

The demo application was developed under Microchip MPLAB version 8.83 with C32 compiler v2.20. Source code of the application 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)



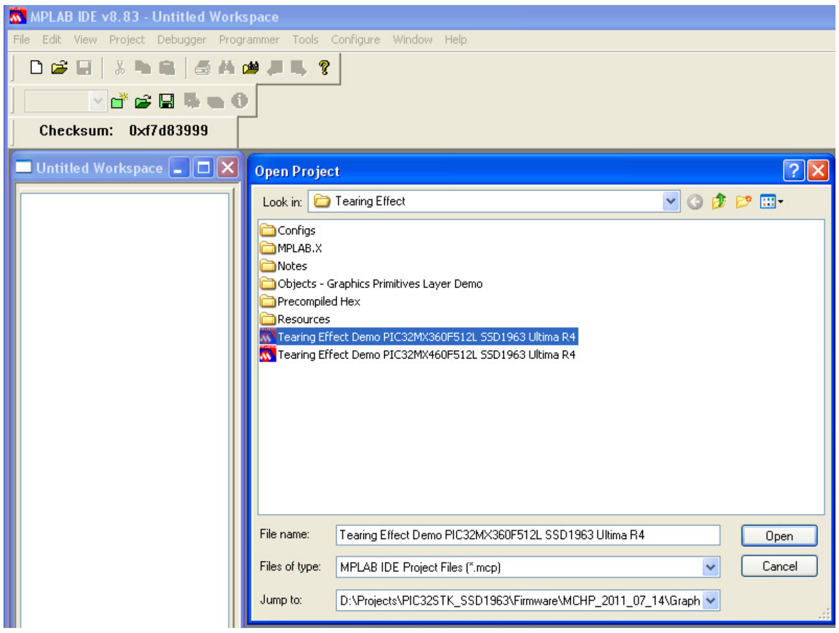
Click to enlarge

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 4 Location of source code

Projects are located at the directory ..\Graphics\Tearing Effect. There are two MCU variants PIC32MX360F512L & PIC32MX460F512L supported. We are using the first model for this document. From the project folder under \Tearing Effect, open the project file *Tearing Effect Demo PIC32MX360F512L SSD1963 Ultima R4.mcp*.



MPLAB IDE v8.83 - Untitled Workspace

Checksum: 0xf7d83999

Untitled Workspace

Open Project

Look in: Tearing Effect

Configs
MPLAB.X
Notes
Objects - Graphics Primitives Layer Demo
Precompiled Hex
Resources
Tearing Effect Demo PIC32MX360F512L SSD1963 Ultima R4
Tearing Effect Demo PIC32MX460F512L SSD1963 Ultima R4

File name: Tearing Effect Demo PIC32MX360F512L SSD1963 Ultima R4

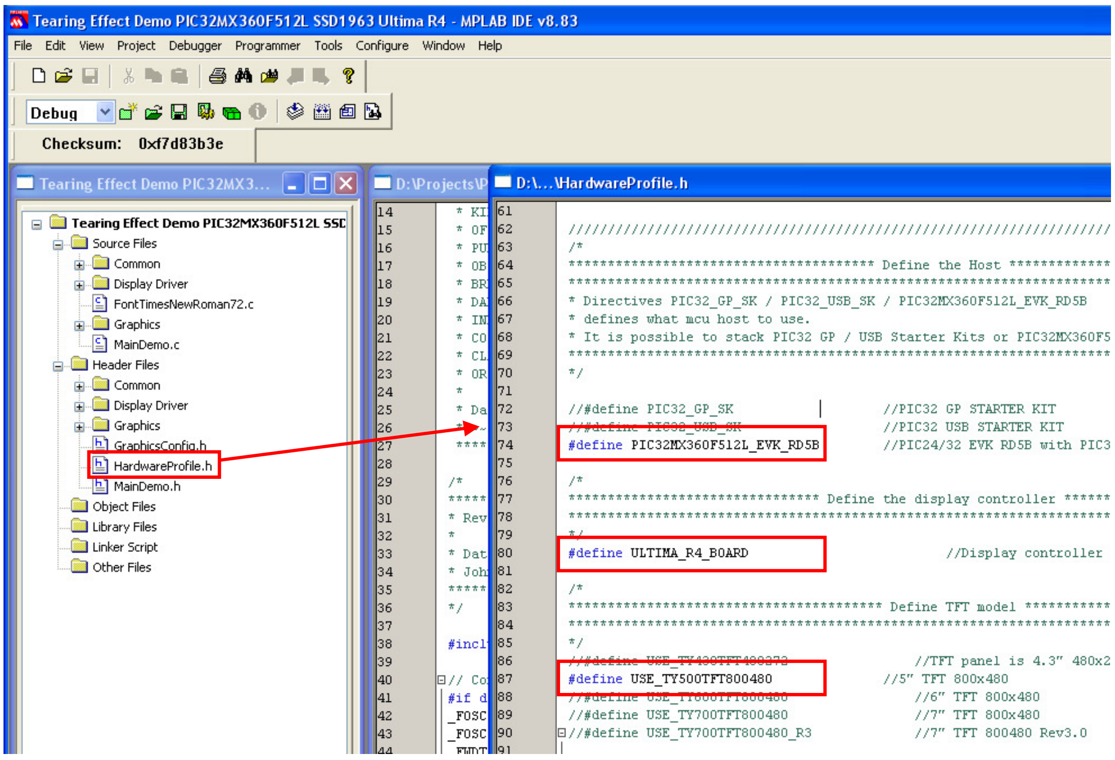
Files of type: MPLAB IDE Project Files (*.mcp)

Jump to: D:\Projects\PIC32STK_SSD1963\Firmware\MCHP_2011_07_14\Graph

File Edit View Project Debugger Programmer Tools Configure Window Help
Open
Cancel

Figure 5 Project location

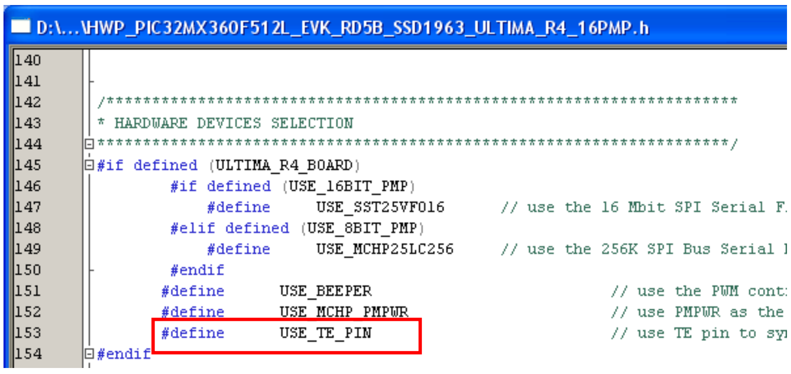
First examine the file *HardwareProfile.h* from project workspace. Make sure the definitions as shown in Figure 6 have been selected. If one is using a 4.3" TFT or 7" WVGA TFT, it is possible to change USE_TY500TFT800480 to USE_TY430TFT480272 or USE_TY700TFT800480_R3. It is also possible to use a PIC32 General Purpose Starter Kit or PIC32 USB Starter Kit for this demo. The PIC32MX360F512L EVK has been chosen because a 2x20 breakout pin is on board for Logic Analyzer.



```
////////////////////////////////////////////////////////////////////
/*
**************************************** Define the Host *************
************************************************************************
* Directives PIC32_GP_SK / PIC32_USB_SK / PIC32MX360F512L_EVK_RD5B
* defines what mcu host to use.
* It is possible to stack PIC32 GP / USB Starter Kits or PIC32MX360F5
************************************************************************
*/

//#define PIC32_GP_SK                    //PIC32 GP STARTER KIT
//#define PIC32_USB_SK                   //PIC32 USB STARTER KIT
#define PIC32MX360F512L_EVK_RD5B         //PIC24/32 EVK RD5B with PIC3

/*
******************************** Define the display controller ******
************************************************************************
*/
#define ULTIMA_R4_BOARD                           //Display controller

/*
***************************************** Define TFT model ***********
************************************************************************
*/
//#define USE_TY430TFT480272                 //TFT panel is 4.3" 480x2
#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 6 HardwareProfile options

Next, browse to the folder ..\Tearing Effect\Configs and locate the file *HWP_PIC32MX360F512L_EVK_RD5B_SSD1963_ULTIMA_R4_16PMP.h.* Locate the definition *#define USE_TE_PIN* necessary for this demo. Leave this option "as-is" for now. We will be changing this option later.


D:\...\HWP_PIC32MX360F512L_EVK_RD5B_SSD1963_ULTIMA_R4_16PMP.h

```


/****************************************************************************
 * HARDWARE DEVICES SELECTION
 ****************************************************************************/
#if defined (ULTIMA_R4_BOARD)
        #if defined (USE_16BIT_PMP)
            #define     USE_SST25VF016      // use the 16 Mbit SPI Serial F.
        #elif defined (USE_8BIT_PMP)
            #define     USE_MCHP25LC256     // use the 256K SPI Bus Serial I
        #endif
       #define      USE_BEEPER                         // use the PWM cont:
       #define      USE_MCHP_PMPWR                     // use PMPWR as the
       #define      USE_TE_PIN                         // use TE pin to sy
#endif
```

Figure 7 Options to use PMPWR and TE pin

Important: Optimization level under Project→Project Options→MPLAB PIC32 C Compiler→Optimization should be set to level 1 or above to use this demo.

About Tearing Effect

An extract from Wiki[iii] about Tearing Effect is summarized in Italic below.

"Screen tearing is a visual artifact in video display where a display device shows information from two or more frames in a single screen draw. The artifact occurs when the video feed to the device isn't in sync with the display's refresh…

During video motion, screen tearing creates a torn look as edges of objects."

The Tearing Effect Signal (TE pin) is a feedback signal from SSD1963 to MCU. This signal reveals the display status about the non-display period that is basically the vertical blanking interval[iv] between frames. During this non-display period, the TE pin goes high. Therefore, this signal enables the MCU to write data by observing the TE pin to avoid tearing.

Figure 8 below captures the waveform with TE pin goes high when Vsync goes low during the demo running. The time interval when TE pin high is measured to be 1.57 ms with Vsync low measured 70.40µs. TE high interval is much longer than Vsync because there are back porch and front porch intervals taking account of the blanking interval.

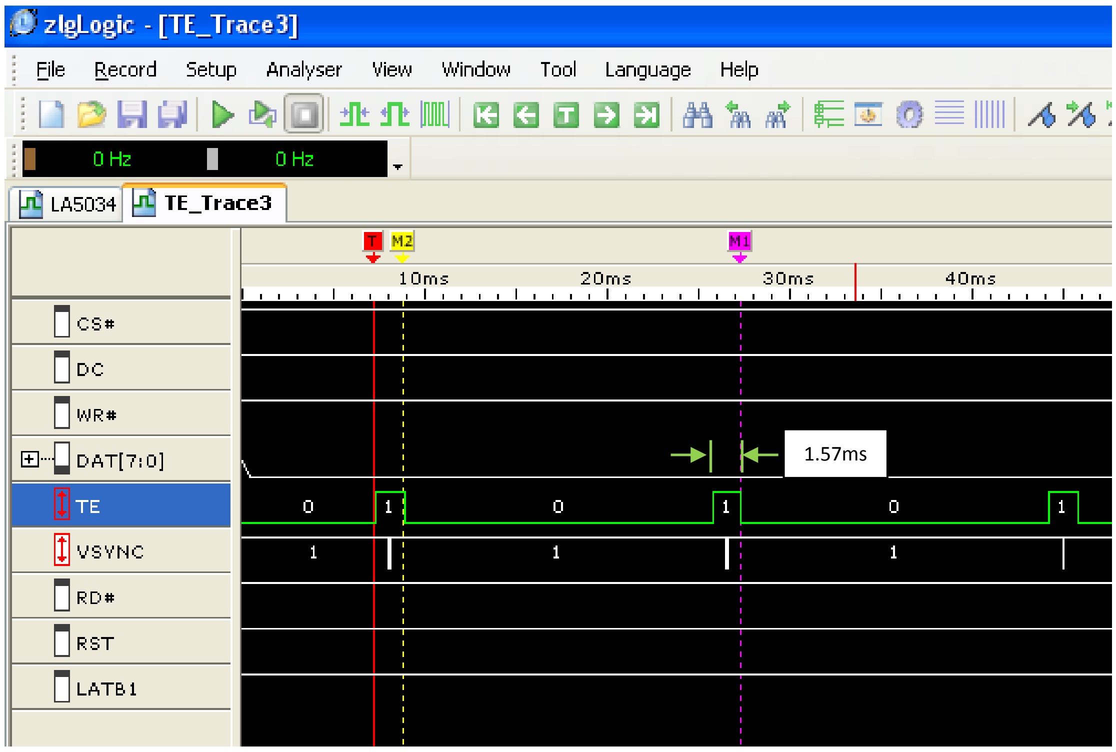


Figure 8 TE captured with correlation to Vsync

The result of measurement on TE and Vsync is matching the hardware profile of TY500TFT800480 in the file.

The blanking period is calculated by the equation below.

Blanking period =
(DISP_VER_PULSE_WIDTH+DISP_VER_BACK_PORCH+DISP_VER_FRONT_PORCH)*1056*T_{CPH}

= (2+25+18)*1056*(1/30MHz)

=1584µs

=1.584ms

The value of 1056 (in T_{CPH}) is given by the horizontal period (T_H) of the display from its specifications. T_{CPH} is given by 1/30MHz because it is the pixel clock period (PCLK) configured in the function ResetDevice(void) in SSD1963.c source file.

Listing 1 shows the relevant lines extracted from ResetDevice(void).

```
#elif defined (USE_TY500TFT800480)|| defined (USE_TY700TFT800480_R3)
WriteCommand(CMD_SET_PCLK);                                        (1)
CS_LAT_BIT = 0;                                                    (2)
WriteData(0x03);                                                   (3)
WriteData(0xff);                                                   (4)
WriteData(0xff);                                                   (5)
CS_LAT_BIT = 1;                                                    (6)
```

Listing 1　　　　**PCLK configuration**

Line (1) writes command CMD_SET_PCLK (0xE6) to set the pixel clock driven by SSD1963. The equation for PCLK is given by

PCLK = PLL Frequency * (LCDC_FPR + 1)/2^{20}

LCDC_FPR is the 20-bit parameter following command CMD_SET_PCLK.

PLL freq = 120MHz configured previously in ResetDevice(void).

Therefore PCLK　　= 120MHz * (0x03FFFF+1)/ 2^{20}

　　　　　　　　= 30MHz

Line (2) sets CS# low to start writing data

Lines (3) – (5) write parameters 0x03, 0xFF, & 0xFF

Line (6) finishes the write cycle by taking CS# high to de-select SSD1963.

The pixel clock is somehow related to the frame rate. With a PCLK set at 30MHz for the 800x480 TFT screen, the frame rate obtained is around 54Hz. It is indeed possible to set a very low PCLK (say, 15MHz) to lengthen the TE time for MCU data write, but the penalty is a flickering display with half the frame rate at 27Hz.

Listing 2 below shows the section to use TE pin from MainDemo.c

```
#if defined (USE_TE_PIN)                                            (1)
SetTearingCfg(1, 0);        //enable TE pin for Vsync               (2)
INTCONbits.INT2EP = 0;   //falling edge detect                      (3)
//debug pin to tell the write time
TRISBbits.TRISB1 = 0;                                               (4)
LATBbits.LATB1 = 1;                                                 (5)
#endif
```

Listing 2 **TE enabled in MainDemo.c**

Line (1) is a directive to choose if TE pin is enabled or not. USE_TE_PIN is defined in the hardware configuration file *HWP_PIC32MX360F512L_EVK_RD5B_SSD1963_ULTIMA_R4_16PMP.h* in the path ..\Tearing Effect\Configs.

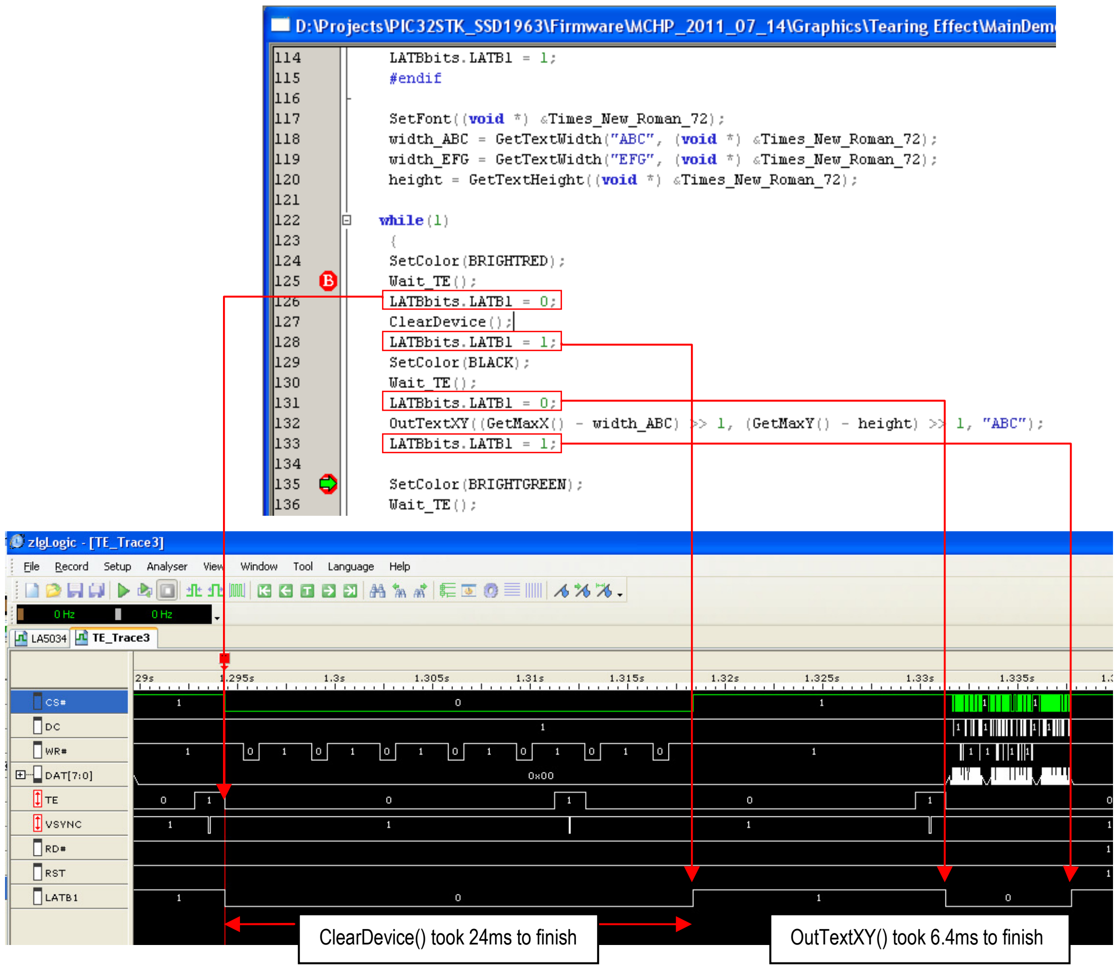
Line (2) switches TE pin ON with Vsync information only.

Line (3) configures INT2 of PIC32MX360F512L to detect the falling edge of TE. A simple polling method is used to synchronize TE pin with the MCU via INT2. Under the hardware configuration file *HWP_PIC32MX360F512L_EVK_RD5B_SSD1963_ULTIMA_R4_16PMP.h* Wait_TE() is defined to poll the TE pin for either a rising edge (INTCONbits.INT2EP = 1) or falling edge (INTCONbits.INT2EP = 0).

```
#if defined (USE_TE_PIN)
        #define Wait_TE()  {IFS0bits.INT2IF = 0; while(!IFS0bits.INT2IF); }
#else
        #define Wait_TE()
#endif
```

Line (4) & (5) configures RB1 as an output pin for timing analysis.

Running the program with breakpoints as shown in Figure 9 reveals the relationship between TE and ClearDevice( ) & OutTextXY( ).



D:\Projects\PIC32STK_SSD1963\Firmware\MCHP_2011_07_14\Graphics\Tearing Effect\MainDem

```
    LATBbits.LATB1 = 1;
    #endif

    SetFont((void *) &Times_New_Roman_72);
    width_ABC = GetTextWidth("ABC", (void *) &Times_New_Roman_72);
    width_EFG = GetTextWidth("EFG", (void *) &Times_New_Roman_72);
    height = GetTextHeight((void *) &Times_New_Roman_72);

   while(1)
    {
    SetColor(BRIGHTRED);
    Wait_TE();
    LATBbits.LATB1 = 0;
    ClearDevice();
    LATBbits.LATB1 = 1;
    SetColor(BLACK);
    Wait_TE();
    LATBbits.LATB1 = 0;
    OutTextXY((GetMaxX() - width_ABC) >> 1, (GetMaxY() - height) >> 1, "ABC");
    LATBbits.LATB1 = 1;

    SetColor(BRIGHTGREEN);
    Wait_TE();
```

zlgLogic - [TE_Trace3]
File Record Setup Analyser View Window Tool Language Help
LA5034
TE_Trace3
CS#
DC
WR#
DAT[7:0]
TE
VSYNC
RD#
RST
LATB1
ClearDevice() took 24ms to finish
OutTextXY() took 6.4ms to finish

Figure 9 ClearDevice(void) took 24ms while OutTextXY() took 6.4ms to finish with compiler optimization option at -Os

The function ClearDevice() simply writes all pixels in pure color, in this case writing BRIGHTRED. It took 24ms to finish rendering the whole screen in 800x480 pixels in 16-bit color depth and the time for OutTextXY() is much shorter at 6.4ms.

There are two options for fast & slow MCUs. If the MCU is faster than the LCD controller it is possible to update the display buffer starting from the rising edge of the TE-high. In this case, the LCD controller will always get the newly updated data prior to each display period. This option is not valid for us.

If the MCU writing speed is slow, it is possible to catch the falling edge of the TE pin and starts writing pixels. Then the LCD controller will always display the old memory content until the next frame.

Figure 10 shows the relationship between Tearing Effect signal and MCU memory writing copied from the data sheet of SSD1963. For us, we simply use the option Slow Write MCU in blue dotted line to remove screen flicker.

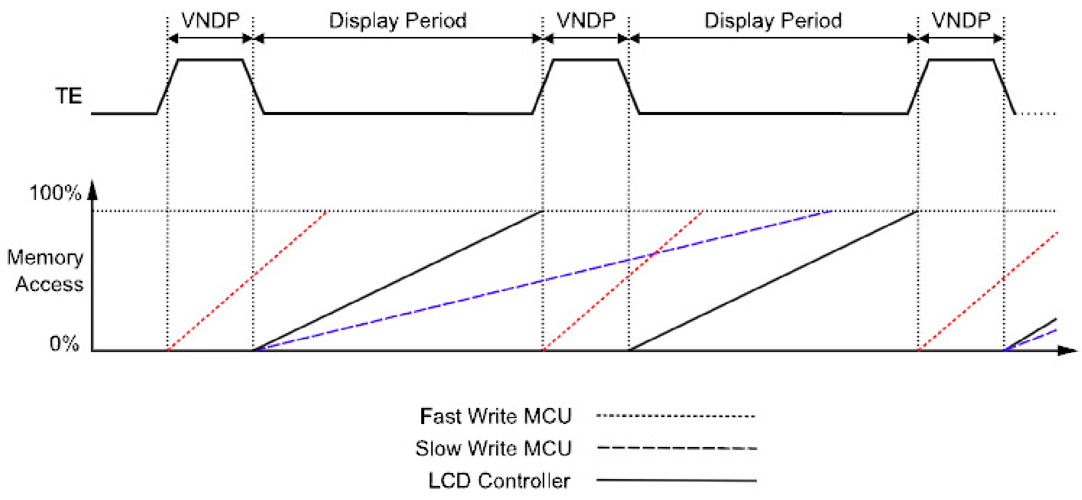


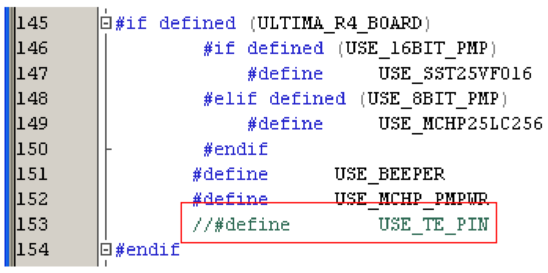
Figure 10 Tearing Effect signal and MCU memory write relationship

Now open the hardware configuration file *HWP_PIC32MX360F512L_EVK_RD5B_SSD1963_ULTIMA_R4_16PMP.h*.

Comment the line #define USE_TE_PIN as shown in Figure 11.

In MPLAB, recompile the program from Project → Build All (or Ctrl + F10).

Under Debugger → Program, and press Run (F9).



```
#if defined (ULTIMA_R4_BOARD)
        #if defined (USE_16BIT_PMP)
            #define     USE_SST25VF016
        #elif defined (USE_8BIT_PMP)
            #define     USE_MCHP25LC256
        #endif
       #define      USE_BEEPER
       #define      USE_MCHP_PMPWR
       //#define        USE_TE_PIN
#endif
```

Figure 11 Comment #define USE_TE_PIN to show screen flicker

The result with TE switched off is illustrated by Figure 12 with ICD3 halted at random moment to for illustration purpose because it is difficult to catch the torn screen with a digital camera which is also refreshing with a certain frequency unknown to us. However, it would be very easy to tell if screen torn occurs because there is a very prominent horizontal line at random location of the screen observable by naked eyes.

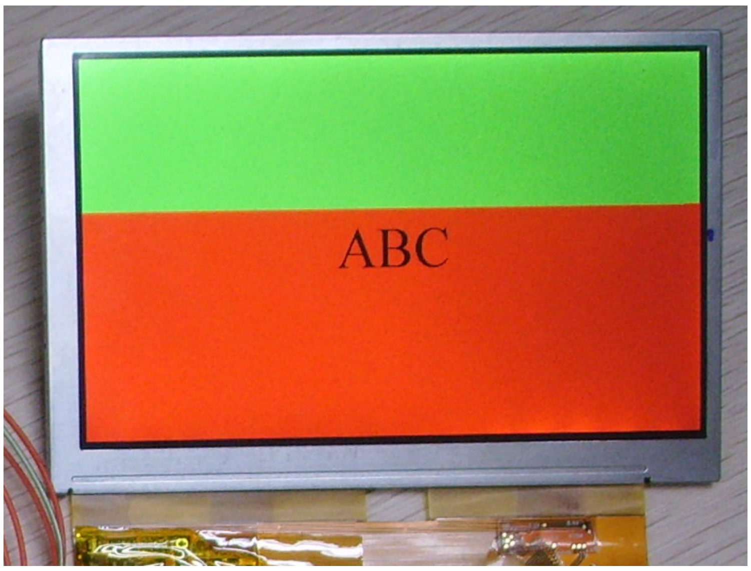


Figure 12 Screen torn at random horizontal positions

One may enable #define USE_TE_PIN again from *HWP_PIC32MX360F512L_EVK_RD5B_SSD1963_ULTIMA_R4_16PMP.h* to notice the difference with and without using TE.

Limitation

Availability of a TE signal alone is not enough to solve all problems. If the MCU takes more than two frames to finish rendering a single picture, flickering is still going to occur as shown in Figure 13.

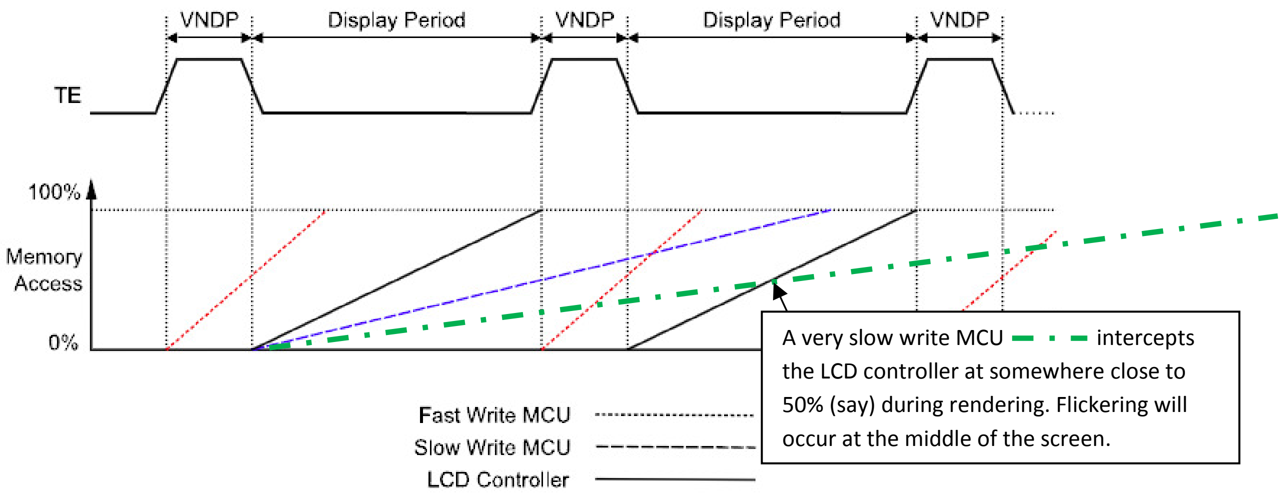

VNDP
Display Period
VNDP
Display Period
VNDP
TE
100%
Memory Access
0%
Fast Write MCU
Slow Write MCU
LCD Controller
A very slow write MCU intercepts the LCD controller at somewhere close to 50% (say) during rendering. Flickering will occur at the middle of the screen.

Figure 13 A very slow MCU renders an image in more than 2 frames

Double-buffering can be used to avoid this problem if the frame buffer is big enough. There is a frame buffer of 1,215Kbytes in SSD1963. Memory requirement is calculated by the equation as below.

Memory required for 16-bit color in 800x480 = 800x480x2 Bytes

= 768,000bytes

This memory footprint is enough for more than a single page but not enough for two pages. Unfortunately there is no blit (BitBlt, BITmap Block Transfer) function available with SSD1963 therefore it is not possible to copy a small block of memory from an off-screen area to the active screen at blanking period as shown in Figure 14.

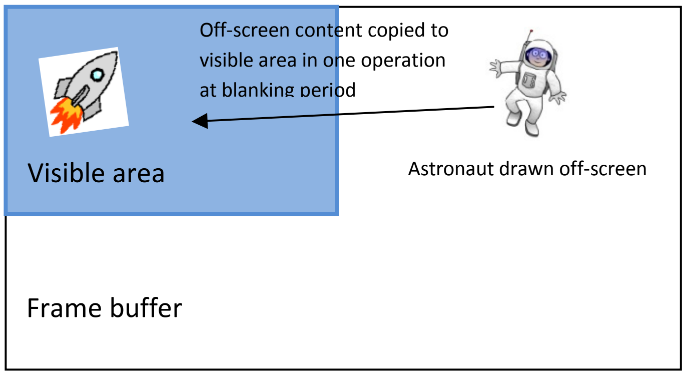

Off-screen content copied to visible area in one operation at blanking period
Astronaut drawn off-screen
Visible area
Frame buffer

Figure 14 BitBlt operation is not possible with SSD1963

This situation is different when a smaller screen is used with SSD1963. For example, the memory requirement for 4.3" TFT of 480x272 pixels in 16-bit color is calculated as follows.

Memory required for 16-bit color in 480x272	= 480x272x2 Bytes
	= 261,120 Bytes.

The frame buffer of SSD1963 is enough to hold more than 4 pages with illustration in Figure 15. Whenever screen update is required, one may use the function SetScrollStart( ) to jump between pages while the MCU will have plenty of time to finish rendering an off-screen pages. This demonstration is available from the Primitive Layer Demo for 4.3" TFT screen TY430TFT480272.

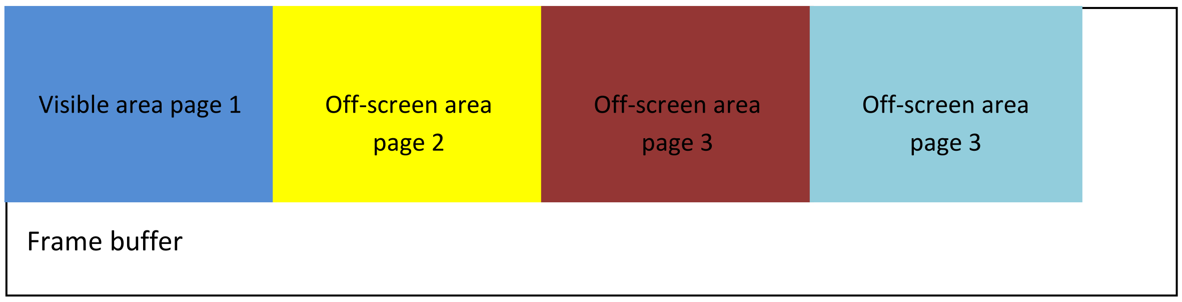


Figure 15 Jump between visual and off-screen pages with SetScrollStart ().

End Notes

[i] http://www.techtoys.com.hk/Displays/Solomon%20SSD1963%20EVK%20R4_1/SSD1963%20EVK%20Ultima%20R4_1.htm
[ii] http://www.techtoys.com.hk/PIC_boards/PIC2432EVK-RD5b/PIC2432%20EVK%20RD5B.htm
[iii] http://en.wikipedia.org/wiki/Screen_tearing
[iv] Resources available including Wiki at http://en.wikipedia.org/wiki/Vertical_synchronization#Vertical_synchronization & further information on horizontal and vertical blanking periods at Section 43, **"Graphics Controller Module (GFX)"** (DS39731) in the "*PIC24F Family Reference Manual"*.