LVDS Display Integration with the AM3517 EVM, AM3517 eXperimenter, OMAP35x Torpedo, OMAP35x SOM-LV, DM3730 Torpedo, and DM3730 SOM-LV Development Kits

Application Note 502

Logic PD // Products
Published: October 2011
Last revised: August 2012

Revision History

REV	EDITOR	DESCRIPTION	APPROVAL	DATE
A	BSB	-Initial Release	RAH	10/21/11
B	RAH	-Throughout: Updated for DM3730 Torpedo and DM3730 SOM-LV Development Kits; updated for LogicLoader v2.5.x; tested against DM37x Windows CE BSP; -Section 1: Added note regarding update of document to include DM3730 SOMs; added note regarding AM3517 SOM support of up to 24 bits; -Added Figure 3.2, Figure 3.3, and Figure 3.4; -Added Section 3.1, 3.2, 7.1, 7.2, and 7.3; -Section 7.4: Added note about how to find information on custom Linux configurations for DM3730 SOMs	BSB, RAH	08/16/12

Table of Contents

1 Introduction

This application note guides users through an example of interfacing a Low Voltage Differential Signaling (LVDS) display to a Logic PD Zoom™ Development Kit Baseboard. Since each LCD, System on Module (SOM) LCD controller, and software configuration is unique, the hope is that this example can be used by customers to assist in their LVDS display integration. The intention of this document is to point users in the right direction so that they may confidently approach their particular situation.

Logic PD does offer support contracts for customers seeking assistance integrating their custom LCD panel to a Logic PD development kit or custom platform. See the Logic PD support packages web page[1] for additional information on available packages.

Additional information on interfacing LCDs to Logic PD's development kits can be found in the *Interfacing LCDs to Logic's Development Kit Baseboards Application Note*.[2] Information on integrating OLED LCDs is available in the *OLED Display Integration with OMAP35x Development Kits Application Note*.[3]

Information on interfacing LCDs specifically to the DM3730 Torpedo Development Kit and DM3730 SOM-LV Development Kit can be found in Logic PD's *LCD Integration with DM3730 Development Kits Application Note*.[4]

In the example below, we are connecting a Shanghai Tianma Micro-Electronics TM104SDH01 LVDS display to a Logic PD OMAP35x SOM-LV Development Kit. Appendix A of this document contains the schematic of the LVDS Display Adaptor Board created by Logic PD to complete this task.

NOTE: This application note has been updated to include instructions for interfacing the DM3730/AM3703 SOMs to an LVDS panel and to 24-bit panels; however, the procedures have not been validated.

NOTE: While the AM3517 eXperimenter Kit and AM3517 EVM Development Kit do not provide access to the *uP_DSS_DOUT[24:16]* signals, the AM3517 SOM-M2 does support up to 24 bits.

2 Reviewing the Specifications

Below is a list of items to consider prior to interfacing an LVDS display panel to a Zoom Development Kit Baseboard.

2.1 Resolution

1. What is the resolution required by the LVDS display panel?
2. Is the required resolution supported by the microprocessor LCD controller?
3. Is there support for the LVDS display's resolution in the Operating System (OS)?

2.2 Color Depth

1. What is the color depth supported by the panel?
2. What is the color depth required by the design?
3. What modifications must be made to the LCD driver of the OS to support the desired color depth?

[1] http://www.logicpd.com/support/support-packages/
[2] http://support.logicpd.com/downloads/105/
[3] http://support.logicpd.com/downloads/1404/
[4] http://support.logicpd.com/downloads/1543/

2.3 Power Supply

Is there a power management chip for the LCD and backlight that is recommended by the manufacturer?

If not:

1. What voltage levels are needed?
2. What maximum currents are needed?
3. What are the power on/off and reset sequence requirements for the LCD power supplies?

2.4 Reset Sequence

Is there additional support logic required to meet the LCD reset requirements or can microprocessor GPIO signals perform the required reset sequence for the LCD?

2.5 I/O Voltage Levels

1. What are the required I/O voltage levels for the targeted LVDS interface chip?
2. Is it compatible with the 1.8V display I/O voltage levels or is voltage translation logic required?

3 Interfacing RGB

There are three different types of bits per pixel:

1. Bits per pixel in the software driver.
2. Bits per pixel in the hardware interface from the SOM.
3. Bits per pixel in the hardware interface to the LCD panel.

Many SOMs can support 12-bit, 16-bit, 18-bit and 24-bit LCD panels. For the OMAP35x SOM, LogicLoader v2.4.x supports 16-bit LCD displays. This is generally the default support in Windows CE and Linux LCD drivers as well. For the DM3730 SOM, LogicLoader v2.5.x and Windows CE BSP 3.0.1 support 16-bit and 24-bit displays. Other color depths can be supported but will require modifications to LogicLoader and the OS LCD display driver. Contact Logic PD[5] for support if this assistance is needed to get your LCD working.

The hardware interface from the SOM supports 16-bit, 18-bit and 24-bit outputs. Each type of output provides color data on specific bit lanes, as discussed below. These can be connected directly to the LCD panel interface. If there is a mismatch because more color bits are available on the panel than on the SOM hardware interface, low-order color bits can be duplicated from high-order color bits.

[5] http://support.logicpd.com/support/askaquestion.php

Figure 3.1 below provides the pin-out of the 60-pin LCD header at J11 on the SDK2 Baseboard, used in the OMAP35x SOM-LV and DM3730 SOM-LV Development Kits.

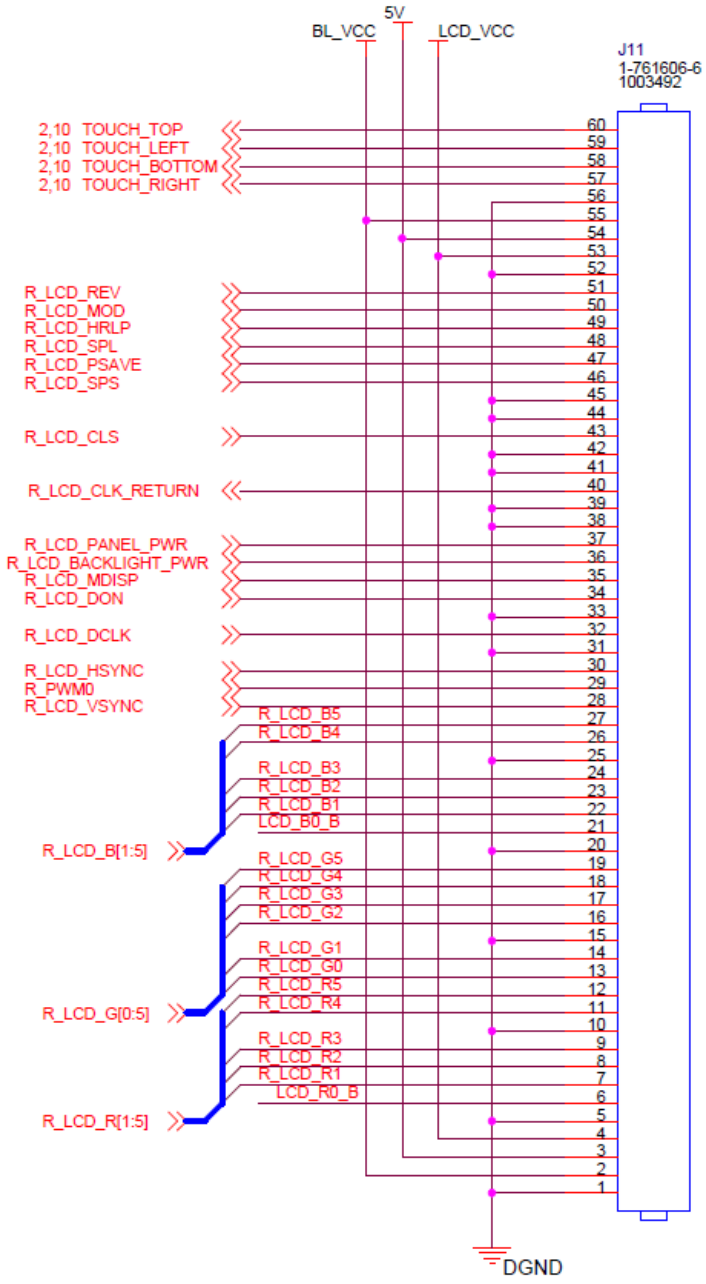


Figure 3.1: SDK2 Baseboard J11 60-pin 0.100" Header Pinout

Figure 3.2 below provides the pin-out of the 60-pin LCD header at J3 on the Torpedo Launcher Baseboard, used in the OMAP35x Torpedo and DM3730 Torpedo Development Kits. The signal names represent the colors for connecting a 16-bit LCD.

NOTE: The Torpedo Launcher Baseboard does not have a separate BL_VCC power rail. Instead, use LCD_VCC to provide 5V. Also, several signals that are no longer useful have been removed on J3. The most important change is R_LCD_DON; instead tie the panel on/off to J_LCD_PANEL_PWR.

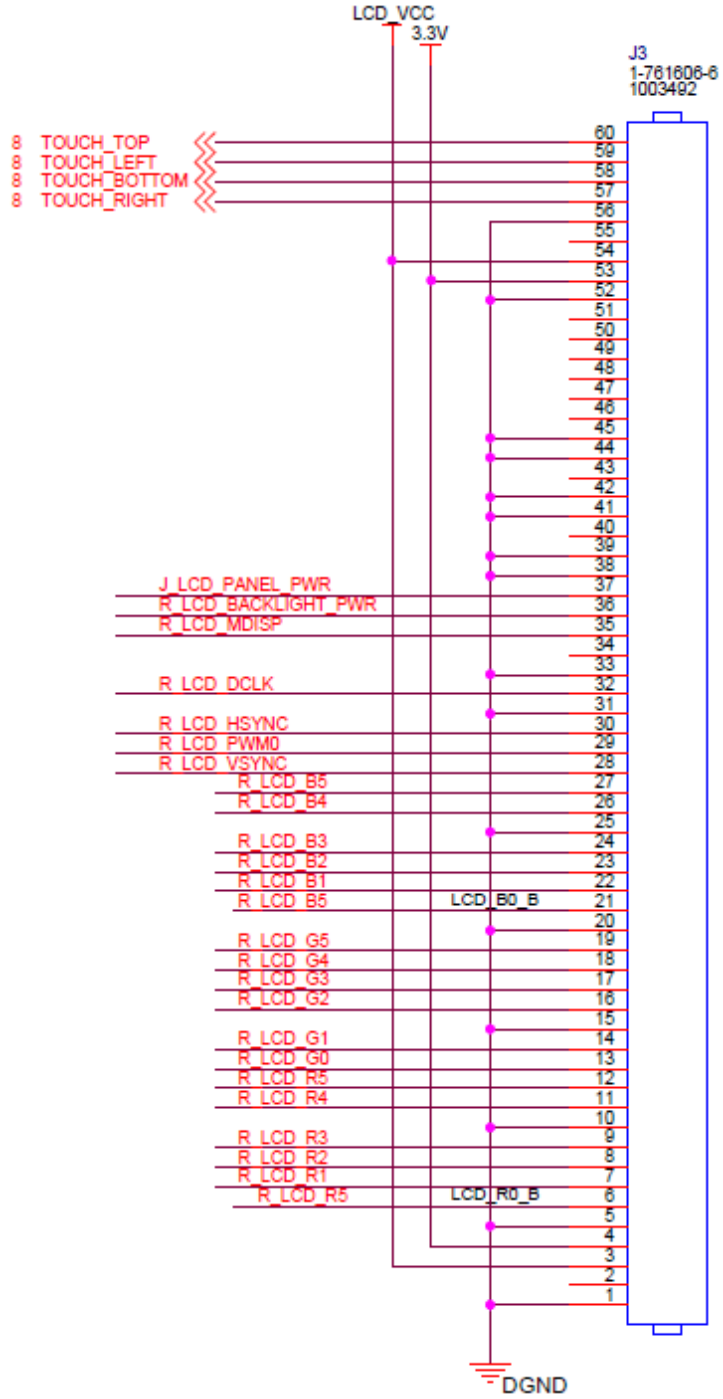


Figure 3.2: Torpedo Launcher Baseboard J3 60-pin 0.100" Header Pinout

J9 on the SDK2 Baseboard and J5 on the Torpedo Launcher Baseboard have different pinouts, as shown in Figure 3.3 and Figure 3.4 below. These connectors have the additional signals from the processor necessary for 18-bit and 24-bit interfacing. Because the SDK2 Baseboard was originally designed for the Freescale i.MX31 series of processors, the signal names for J9 do not align with how the signals are used in 24-bit mode on TI processors. Be sure to verify with the tables given in Section 3.2.

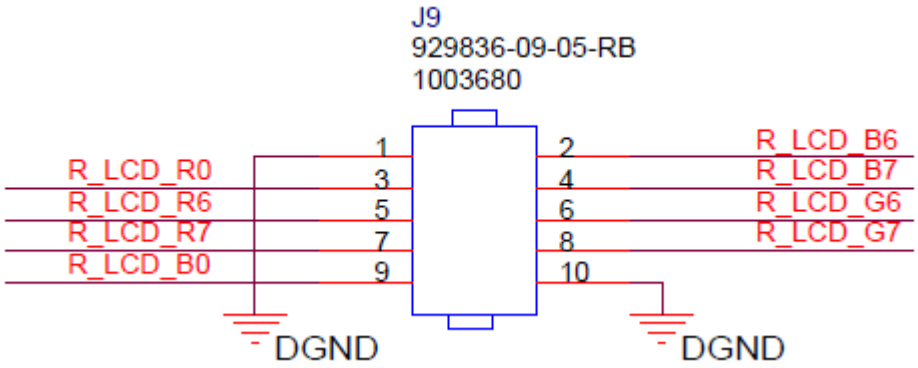


Figure 3.3: SDK2 Baseboard J9 10-pin 0.100" Header Pinout

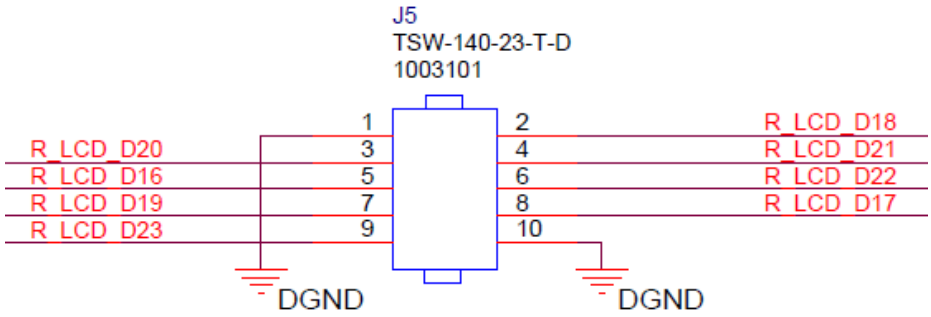


Figure 3.4: Torpedo Launcher 3 J5 10-pin 0.100" Header Pinout

In the examples below, the control signals LCD_DCLK, LCD_HSYNC, LCD_VSYNC, LCD_MDISP and PWM are assumed to be connected and not discussed. Some advanced panels might not need one or more of these signals; validate necessary signals against the LCD data sheet.

3.1 16-bit SOM Hardware Interface to 18-bit LCD Panel

Before the OS display driver can work, the microprocessor LCD controller must be connected correctly to the LCD color signals. In this example, we connect a 16-bit CPU hardware interface to an 18-bit LCD panel. The advantage of using 16-bit is that the software driver support is already available and does not cause the same loss of performance that may occur when supporting 18-bit display panels due to extra data being fetched across the memory bus. The extra low-order bits for red and blue on the LCD panel are simply driven with copies of the high order bits.

Table 3.1 shows how the signals proceed from the CPU signal on the left, through the connectors, and to the LVDS signal on the right. The pin names and pin numbers are identical for J11 on the SDK2 Baseboard and J3 on the Torpedo Launcher Baseboard.

Table 3.1: Signal Mapping from 16-bit LCD Interface to 18-bit LVDS Interface

CPU Signal	SOM Signal Name	Pin Name	Pin Number	LVDS Signal
DSS_D15	LCD_D15	LCD_R0_B	6	R0
DSS_D11	LCD_D11	R_LCD_R1	7	R1
DSS_D12	LCD_D12	R_LCD_R2	8	R2
DSS_D13	LCD_D13	R_LCD_R3	9	R3
DSS_D14	LCD_D14	R_LCD_R4	11	R4
DSS_D15	LCD_D15	R_LCD_R5	12	R5
DSS_D5	LCD_D5	R_LCD_G0	13	G0
DSS_D6	LCD_D6	R_LCD_G1	14	G1
DSS_D7	LCD_D7	R_LCD_G2	16	G2
DSS_D8	LCD_D8	R_LCD_G3	17	G3
DSS_D9	LCD_D9	R_LCD_G4	18	G4
DSS_D10	LCD_D10	R_LCD_G5	19	G5
DSS_D4	LCD_D4	LCD_B0_B	21	B0
DSS_D0	LCD_D0	R_LCD_B1	22	B1
DSS_D1	LCD_D1	R_LCD_B2	23	B2
DSS_D2	LCD_D2	R_LCD_B3	24	B3
DSS_D3	LCD_D3	R_LCD_B4	25	B4
DSS_D4	LCD_D4	R_LCD_B5	26	B5

Figure 3.5 provides the LCD output and data format for the parallel interface when interfacing to 16-bit displays. The signals represent different colors on the SOM using a 16-bit color depth.

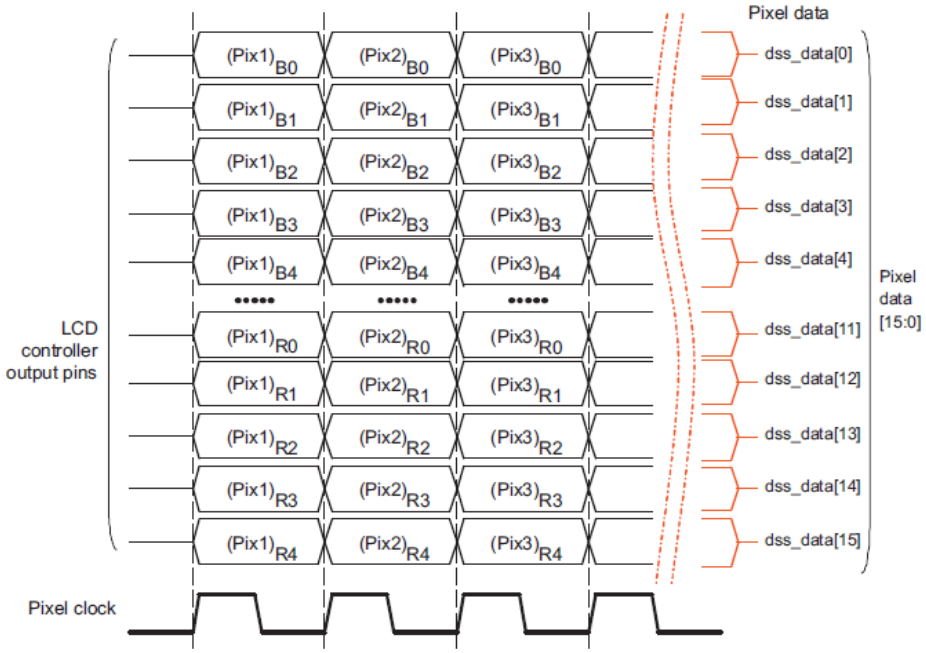


Figure 3.5: CPU 16-bit LCD Output and Data Format for Parallel Interface

3.2 24-bit CPU Hardware Interface to 24-bit LCD Panel

In this example, we connect a 24-bit CPU hardware interface to a 24-bit LCD panel. A 24-bit panel allows for full color range, although 50% additional bandwidth is required from the memory interface to keep the LCD controller FIFOs full. The pin names and pin numbers are not the same for J9 on the SDK2 Baseboard and J5 on the Torpedo Launcher Baseboard; therefore, separate signal mappings have been provided for each baseboard below.

NOTE: The colors associated with each signal are different in 24-bit mode as compared to 16-bit mode.

Table 3.2 shows how the signals proceed for the SDK2 Baseboard from the CPU signal on the left, through the connectors, and to the LVDS signal on the right.

Table 3.2: Signal Mapping from 24-bit SDK2 Baseboard to 24-bit LVDS Interface

CPU signal	SOM-LV signal	J11/J9 Pin Name	J11/J9 Pin number	LVDS signal
DSS_D16	LCD_D16	LCD_R0	J9.3	R0
DSS_D17	LCD_D17	LCD_R6	J9.5	R1
DSS_D18	LCD_D18	LCD_R7	J9.7	R2
DSS_D19	LCD_D19	LCD_B0	J9.9	R3
DSS_D20	LCD_D20	LCD_B6	J9.2	R4
DSS_D21	LCD_D21	LCD_B7	J9.4	R5
DSS_D22	LCD_D22	LCD_G6	J9.6	R6
DSS_D23	LCD_D23	LCD_G7	J9.8	R7
DSS_D8	LCD_D8	R_LCD_G3	J11.17	G0
DSS_D9	LCD_D9	R_LCD_G4	J11.18	G1
DSS_D10	LCD_D10	R_LCD_G5	J11.19	G2
DSS_D11	LCD_D11	R_LCD_R1	J11.7	G3
DSS_D12	LCD_D12	R_LCD_R2	J11.8	G4
DSS_D13	LCD_D13	R_LCD_R3	J11.9	G5
DSS_D14	LCD_D14	R_LCD_R4	J11.11	G6
DSS_D15	LCD_D15	R_LCD_R5	J11.12	G7
DSS_D0	LCD_D0	R_LCD_B1	J11.22	B0
DSS_D1	LCD_D1	R_LCD_B2	J11.23	B1
DSS_D2	LCD_D2	R_LCD_B3	J11.24	B2
DSS_D3	LCD_D3	R_LCD_B4	J11.26	B3
DSS_D4	LCD_D4	R_LCD_B5	J11.27	B4
DSS_D5	LCD_D5	R_LCD_G0	J11.13	B5
DSS_D6	LCD_D6	R_LCD_G1	J11.14	B6
DSS_D7	LCD_D7	R_LCD_G2	J11.16	B7

Table 3.3 shows how the signals proceed for the Torpedo Launcher Baseboard from the CPU signal on the left, through the connectors, and to the LVDS signal on the right.

Table 3.3: Signal Mapping from 24-bit Torpedo Launcher Baseboard to 24-bit LVDS Interface

CPU signal	Torpedo signal	J3/J5 Pin Name	J3/J5 Pin number	LVDS signal
DSS_D16	LCD_D16	R_LCD_D16	J5.5	R0
DSS_D17	LCD_D17	R_LCD_D17	J5.8	R1
DSS_D18	LCD_D18	R_LCD_D18	J5.2	R2
DSS_D19	LCD_D19	R_LCD_D19	J5.7	R3
DSS_D20	LCD_D20	R_LCD_D20	J5.3	R4
DSS_D21	LCD_D21	R_LCD_D21	J5.4	R5
DSS_D22	LCD_D22	R_LCD_D22	J5.6	R6
DSS_D23	LCD_D23	R_LCD_D23	J5.9	R7
DSS_D8	LCD_D8	R_LCD_G3	J3.17	G0
DSS_D9	LCD_D9	R_LCD_G4	J3.18	G1
DSS_D10	LCD_D10	R_LCD_G5	J3.19	G2
DSS_D11	LCD_D11	R_LCD_R1	J3.7	G3
DSS_D12	LCD_D12	R_LCD_R2	J3.8	G4
DSS_D13	LCD_D13	R_LCD_R3	J3.9	G5
DSS_D14	LCD_D14	R_LCD_R4	J3.11	G6
DSS_D15	LCD_D15	R_LCD_R5	J3.12	G7
DSS_D0	LCD_D0	R_LCD_B1	J3.22	B0
DSS_D1	LCD_D1	R_LCD_B2	J3.23	B1
DSS_D2	LCD_D2	R_LCD_B3	J3.24	B2
DSS_D3	LCD_D3	R_LCD_B4	J3.26	B3
DSS_D4	LCD_D4	R_LCD_B5	J3.27	B4
DSS_D5	LCD_D5	R_LCD_G0	J3.13	B5
DSS_D6	LCD_D6	R_LCD_G1	J3.14	B6
DSS_D7	LCD_D7	R_LCD_G2	J3.16	B7

Figure 3.6 provides the LCD output and data format for the parallel interface when interfacing to 24-bit displays. The signals represent different colors on the CPU using a 24-bit color depth.

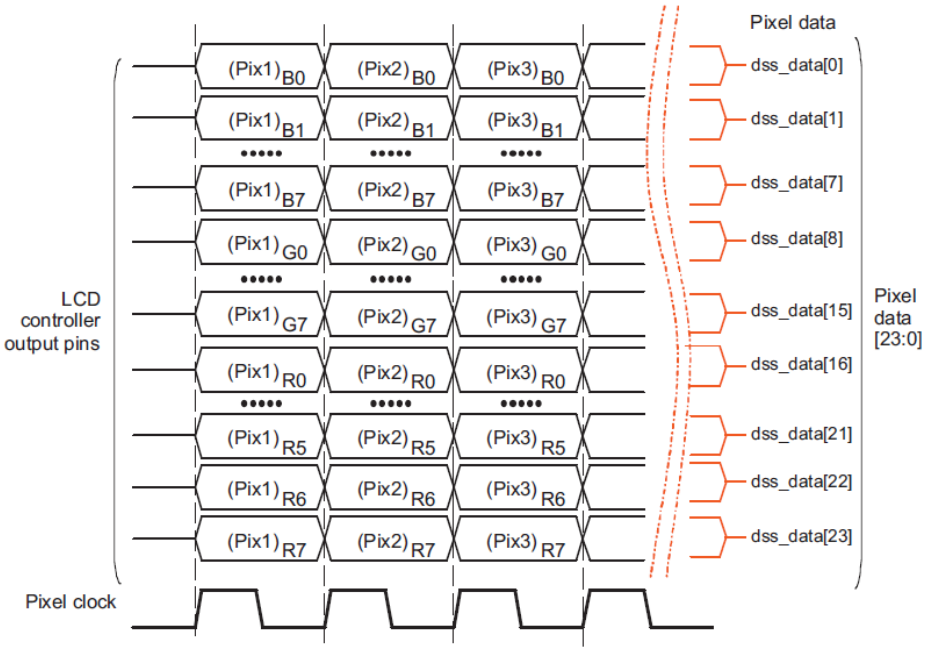


Figure 3.6: CPU 24-bit LCD Output and Data Format for Parallel Interface

4 Selecting Companion Chips

This design example interfaces a 16-bit CPU hardware interface to an 18-bit LCD. Several companion chips are required to interface the TM104SDH01 LCD panel. In this example, the LVDS18B was chosen to connect a parallel LCD controller to an LVDS interface, based on TI's recommendation in their *LCD connectivity wiki article*. [6]

Another companion chip was needed to control the backlight on the TM104SDH01 LCD display. Logic PD recommends contacting the LCD manufacturer for their recommendation when selecting a backlight driver companion chip. In this case, Tianma recommended using TI's TPS61165DBVR backlight driver.

NOTE: When integrating an LCD, designers should request as much information from the LCD manufacturer as possible. This includes reference schematics, software, errata sheets, application notes, and any additional documentation that may be helpful in getting the LCD working.

Based on the components selected for this design, the block diagram in Figure 4.1 was developed.

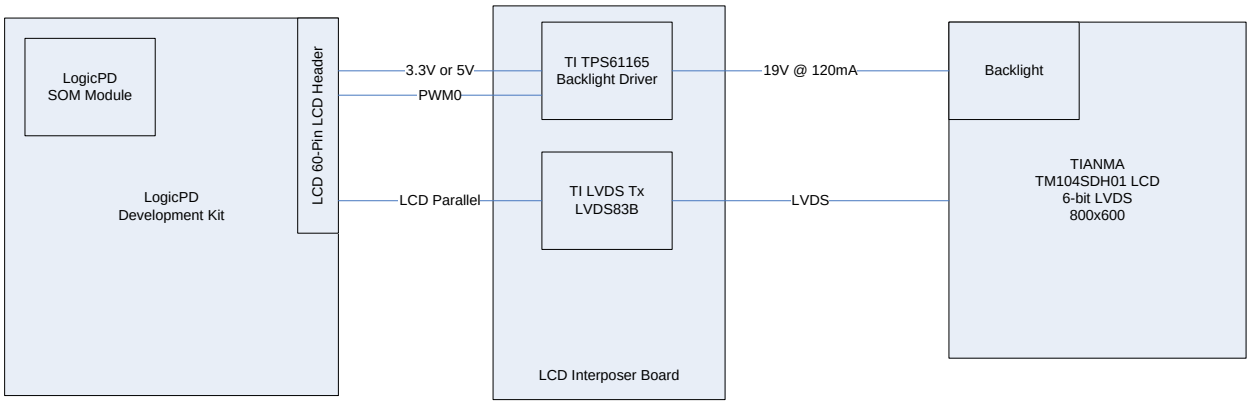


Figure 4.1: LVDS Interposer Board Block Diagram

[6] http://processors.wiki.ti.com/index.php/LCD_connectivity

5 LCD, Backlight and Touch Interface Considerations

Many small LCD panels come with a ribbon cable that easily connects to the manufacturer's recommended mating connector. In these cases, it is beneficial to get the manufacturer's recommended mating connector for the LCD ribbon cable, touch cable, and backlight cable. Plan to design these mating connectors into the interposer PCB that interfaces to the 60-pin header.

For large display panels, custom cables are commonly required to interface your LCD interposer board and your custom LCD panel. This was the case when designing in the TM104SDH01 LCD panel. There can be a long lead time to get custom cabling, so Logic PD recommends allocating the cables prior to beginning the schematic phase of your design.

NOTE: Once the required cabling is determined, it may be beneficial to look for pre-assembled cables or pre-crimped wires to incorporate into the design. Pre-crimped wires allow designers to build the cables by adding the pre-crimped wires to the connectors with the correct number of positions.

6 Schematic and Layout Techniques

In this design, some techniques were used that can assist in debugging newly integrated LCD designs. Logic PD recommends considering these debug techniques if the board real estate or the schedule allows for a second re-spin of a tighter board following a proof-of-concept board.

6.1 Technique 1: Design an Interposer

Throughout this application note, we have discussed interfacing to an interposer. An interposer connects to Logic PD's development kit, allowing designers to verify their LCD interface without having to worry about other issues associated with their customer design.

6.2 Technique 2: Provide Test Points

Provide test points to all signals that might not be otherwise accessible, like signals under a BGA. This could prevent a re-spin of the PCB to get the display panel working in cases where a design error occurs on a signal that would have otherwise been inaccessible.

6.3 Technique 3: Place Components Opposite of Interfacing Connector

Place all components and test points on the interposer opposite of the 60-pin interfacing connector. This will allow the designer to gain access to signals and test points without having to solder wires if they become squeezed between the LCD interposer board and the development kit.

6.4 Technique 4: Use Shorted Resistors

For signals that might require rework, it is recommended to design in a zero ohm resistor. In this design, we provided many shorted resistors that allow a net to pass through without populating a zero ohm resistor. However, if access to a net is needed, the trace between the pads of the resistor can simply be cut and the signal can be intercepted using the pads of the resistor. The repair of the cut trace is as simple as populating a zero ohm resistor. Figure 6.1 provides the layout note that was added to the schematic informing the layout engineer about the purpose of the shorted resistor.

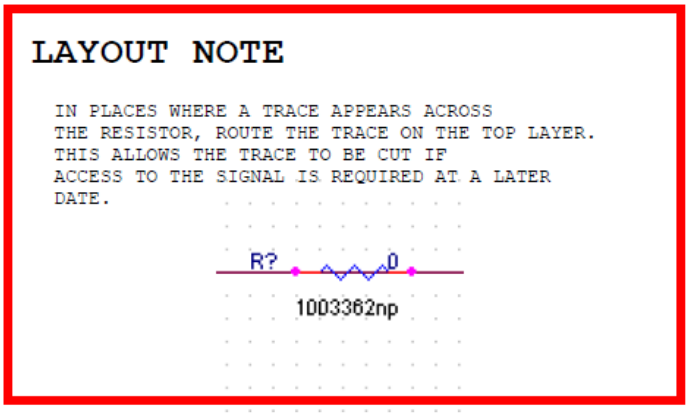


Figure 6.1: Shorted Resistor Symbol

6.5 Technique 5: Connect Power Supply Enable Signals

Power supply chips that are recommended for powering the LCD and/or the backlight generally have enable signals. Certain LCD panels require a power-up sequence which may or may not be documented at the time of development. For those cases, it is recommended to connect each power supply enable signal to an unused zero ohm resistor if a pull-up or pull-down is planned for the final design.

7 Programming

Information on programing the LCD is generally extracted from the target LCD datasheet. Table 7.1 below provides a parameters table similar to that found in Shanghai Tianma's *TM104SDH01 Datasheet*.[7] Designers are encouraged to look for similar parameters in their LCD datasheet.

Table 7.1: LCD Parameters

	Parameter	Symbol	Min.	Typ.	Max.	Unit
Dclk Frequency		1/Tclk	32	40	50	MHz
Horizontal Section	Horizontal total	Th	866	1056	1064	Tclk
	Horizontal blanking	Thb	66	256	264	Tclk
	Valid Data Width	Thd	800	800	800	Tclk
Vertical Section	Frame Rate	-	-	60	70	Hz
	Vertical total	Tv	604	628	800	Th
	Vertical blanking	Tvb	4	28	200	Th
	Valid Data Width	Tvd	600	600	600	Th

Table 7.2 provides information gathered from both Table 7.1 and values extracted from the data to program the appropriate LCD configuration registers.

Table 7.2: LCD Settings

Settings	Calculation	Value
Width DISPC_SIZE_LCD[10:0] DISPC_GFX_SIZE[10:0]	Horizontal Valid Data Width	800
Height DISPC_SIZE_LCD[26:16] DISPC_GFX_SIZE[26:16]	Vertical Valid Data Width	600
Horizontal Back Porch DISPC_TIMING_H[31:20]	Set to 1	1
Horizontal Front Porch DISPC_TIMING_H[19:8]	Horizontal blanking (typ) - HSYNC Pulse Width - Horizontal Back Porch	254
HSYNC Pulse Width DISPC_TIMING_H[7:0]	Set to 1	1
Vertical Back Porch DISPC_TIMING_V[31:20]	Set to 1	1
Vertical Front Porch DISPC_TIMING_V[19:8]	Vertical blanking (typ) – VSYNC Pulse Width – Vertical Back Porch	26
VSYNC Pulse Width DISPC_TIMING_V[7:0]	Set to 1	1
CM_SLKSEL_DSS CM_CLKSEL_DSS1[7:0]	DCLK Frequency (typ) 40MHz – This is the target frequency. DPLL4 FCLK_OUT_x2 = 864MHz (default value set in OS) CM_CLKSEL_DSS1[7:0] = 11 DCLK = DPLL4 FCLK_OUT_x2 / CM_CLKSEL_DSS1[7:0] / (DISPC_DIVISOR[LCD] * DISPC_DIVISOR[PCD]) DCLK = 864 / 11 / (1 * 2) DCLK = 39.27 MHz	11
LCD_CNTR Divider DISPC_DIVISOR[23:16]		1
Pixel Divider DISPC_DIVISOR[7:0]		2

[7] http://tianma-europe.com/downloads/tm104sdh01_f_v2.1.pdf

7.1 LogicLoader 2.4.x

Using the values collected in Table 7.2 and reviewing both the *AM35x ARM Microprocessor Technical Reference Manual (TRM)*[8] and the *OMAP35x TRM*,[9] the following LogicLoader commands were created to configure the LCD and verify its functionality.

To configure your custom LCD in LogicLoader's configuration block, complete the following steps:

1. Create a config block.

```
losh> config CREATE
```

2. Open a default display.

```
losh> video-open 15 16
```

3. Update the display control registers with your LCD custom values.

```
losh> w /w 0x48050440 0x00018109    # DISPC_CONTROL
losh> w /w 0x4805047C 0x0257031F    # DISPC_SIZE_LCD
losh> w /w 0x4805048C 0x0257031F    # DISPC_GFX_SIZE
losh> w /w 0x48050464 0x0000FD00    # DISPC_TIMING_H
losh> w /w 0x48050468 0x00001900    # DISPC_TIMING_V
losh> w /w 0x48050470 0x00010002    # DISPC_DIVISOR
losh> w /w 0x48050444 0x00000000    # DISPC_CONFIG
losh> w /w 0x4805046C 0x00000000    # DISPC_POL_FREQ
losh> w /w 0x48004E40 0x0001000B    # CM_CLKSEL_DSS
```

4. Save your custom LCD display control register settings. In this example, we have saved them under the name *LVDS*.

```
losh> config V LVDS 800 600
```

5. Test the LVDS display settings.

```
losh> video-open LVDS 16
losh> draw-test
```

[8] http://www.ti.com/product/am3517
[9] http://www.ti.com/product/omap3530

With the correct setting on the display, you should see a framed red, green, blue, and stipple test pattern as seen in Figure 7.1.

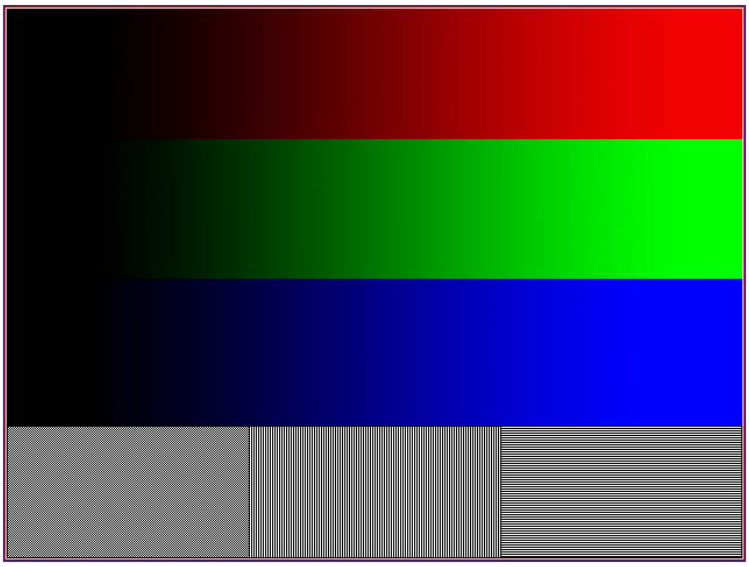


Figure 7.1: LogicLoader 2.4.x Draw-Test Display

6. Now reset your system to test the saved settings.
7. Open the LVDS display.

```
losh> video-open LVDS 16
```

8. Execute the draw-test command.

```
losh> draw-test
```

7.2 LogicLoader 2.5.x

With LogicLoader 2.5.x, Logic PD has discontinued use of a separately stored config block. LogicLoader now uses several setup and script files saved in */lboot* and custom video configurations must be recreated each time the system boots. *lboot.lol* can be modified to execute the video configuration commands during the boot process. The example boot script below shows the custom LCD configuration for 16-bit LVDS in *lboot.lol.*

```
# This is an example boot script.

video-init 15 16

w /w 0x48050440 0x00018109    # DISPC_CONTROL
w /w 0x4805047C 0x0257031F    # DISPC_SIZE_LCD
w /w 0x4805048C 0x0257031F    # DISPC_GFX_SIZE
w /w 0x48050464 0x0000FD00    # DISPC_TIMING_H
w /w 0x48050468 0x00001900    # DISPC_TIMING_V
w /w 0x48050470 0x00010002    # DISPC_DIVISOR
w /w 0x48050444 0x00000000    # DISPC_CONFIG
w /w 0x4805046C 0x00000000    # DISPC_POL_FREQ

w /w 0x48004E40 0x0001000B    # CM_CLKSEL_DSS
```



```
video-add LVDS 800 600
```

Once the boot script completes the *video-add* command, the *video-open* command can be used with the LVDS display name. These commands can be used within the *lboot.lol* script or after it has completed execution.

```
losh> video-open LVDS 16
losh> draw-test
```

7.3 Windows CE

If an LCD display is configured and enabled in LogicLoader, Windows CE will automatically read the LCD registers and use the dimensions for the display.

7.4 Linux

For the DM3730 SOMs, information on custom Linux configurations can be found in Logic PD's *LCD Integration with DM3730 Development Kits Application Note*.[10]

The Tianma display was verified running in Linux on the AM3517 SOM-M2 development platform. TI has information available on the display driver in both the *V4L2 Driver* wiki article[11] and the *Display Timing* wiki article.[12]

In order to get the display working in Linux, one of the Sharp-lq043t1dg01 display drivers was updated in the source using the patch in Figure 7.2 below.

[10] http://support.logicpd.com/downloads/1543/
[11] http://processors.wiki.ti.com/index.php/UserGuideDisplayDrivers_PSP_03.00.00.05#Command_Line_arguments
[12] http://processors.wiki.ti.com/index.php?title=DSS2_SYSFS_Examples#Display_timing


```
This patch modifies the default display timings to support the LVDS
display.  It also clears the following bits which were previously
set in the DSPC_POLL_FREQ register:
        IEO - Invert Output Enable
        IHS - Invert Horizontal Sync
        IVS - Invert Vertical Sync

To adjust the settings at runtime, manipulate the file:
        /sys/devices/platform/omapdss/display0/timings


---
 .../video/omap2/displays/panel-sharp-lq043t1dg01.c |   37 ++++++++++++--------
 1 files changed, 22 insertions(+), 15 deletions(-)

diff --git a/drivers/video/omap2/displays/panel-sharp-lq043t1dg01.c
b/drivers/video/omap2/displays/panel-sharp-lq043t1dg01.c
index e75798e..3e9ce7c 100644
--- a/drivers/video/omap2/displays/panel-sharp-lq043t1dg01.c
+++ b/drivers/video/omap2/displays/panel-sharp-lq043t1dg01.c
@@ -25,25 +25,32 @@
 #include <plat/display.h>

 static struct omap_video_timings sharp_lq_timings = {
-       .x_res = 480,
-       .y_res = 272,
-
-       .pixel_clock    = 9000,
-
-       .hsw            = 42,
-       .hfp            = 3,
-       .hbp            = 2,
-
-       .vsw            = 11,
-       .vfp            = 3,
-       .vbp            = 2,
+       /* LVDS--
+        * Changes to panel timing for LVDS 800x600 display.
+        * 40000,800/254/1/1,600/26/1/1
+        */
+       .x_res = 800,
+       .y_res = 600,
+
+       .pixel_clock    = 40000,
+
+       .hsw            = 1,
+       .hfp            = 254,
+       .hbp            = 1,
+
+       .vsw            = 1,
+       .vfp            = 26,
+       .vbp            = 1,
+       /* --LVDS */
 };

 static int sharp_lq_panel_probe(struct omap_dss_device *dssdev)
 {
-       dssdev->panel.config = OMAP_DSS_LCD_TFT | OMAP_DSS_LCD_IVS |
-                       OMAP_DSS_LCD_IHS | OMAP_DSS_LCD_IEO;
+       /* LVDS--
+        * Changes to get rid of the inverted signals.
+        */
+       dssdev->panel.config = OMAP_DSS_LCD_TFT;
+       /* --LVDS */
        dssdev->panel.acb = 0x0;
        dssdev->panel.timings = sharp_lq_timings;

--
```

Figure 7.2: Panel-sharp-lq043t1dg01.c Patch

After applying the patch and building a kernel with an updated display driver, the steps below were followed to display the test image in Linux.

1. Launch U-Boot.
2. Adjust *otherbootargs* using the commands below.

```
u-boot> setenv otherbootargs "vram=3M omapfb.mode=lcd:800x600-16@72"
u-boot> setenv otherbootargs "omapfb.debug=y  $otherbootargs"
u-boot> setenv otherbootargs "omapfb.test=y   $otherbootargs"
u-boot> setenv otherbootargs "omapdss.debug=y $otherbootargs"
```

3. Boot Linux.

   In this example, PSP 03.00.00.05 for the AM3517 SOM-M2 was used to receive the test pattern seen in Figure 7.3 with *omapfb.debug*, *omapfb.test* and *omapdss.debug* set to ‘y’ in U-Boot.

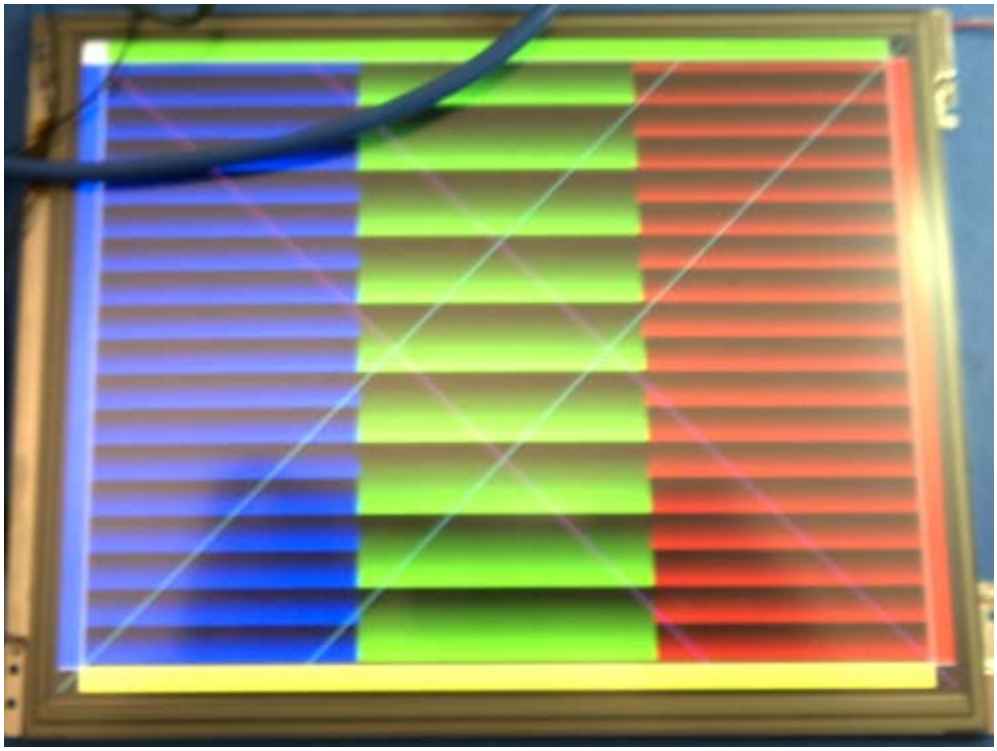


Figure 7.3: Linux LCD Test Pattern

If the display image is not quite clear, it may be possible to make adjustments by accessing some of the parameters at the Linux prompt. Information about how to modify the available LCD controller parameters while running your Linux image has been provided below.

Manual tweaks of the front/back porch and synchronization width settings can be done using the following files and by completing the steps below:

- */sys/devices/platform/omapdss/display0/timing*
- */sys/devices/omapdss/display0/enabled*

1. Read the files using the *cat* command.

```
$ cat /sys/devices/platform/omapdss/display0/timing
```

When complete, output similar to that included below will be seen.

```
PCKL,x-res/fp/bp/sw,y-res/fp/bp/sw

40000,800/254/1/1,600/26/1/1

mapdss/display0/timings

PCKL  : pixel clock (referred to as Dclk in the data sheet)

x-res : x-resolution (800 in this example)

y-res : y-resolution (600 in this example)

fp    : front-porch

bp    : back-porch

sw    : width of synchronization signal
```

2. Writes to a file using an *echo* command, as seen below, allow changes to the existing driver. In this example we change the vertical front porch from twenty-six to twenty-four. There should be no noticeable difference in this change.

```
$ echo 0 > /sys/devices/platform/omapdss/display0/enabled

$ fbset -fb /dev/fb0 -xres 800 -yres 600

$ echo 1 > /sys/devices/platform/omapdss/display0/enabled

$ echo '40000,800/254/1/1,600/24/1/1' >
/sys/devices/platform/omapdss/display0/timings
```

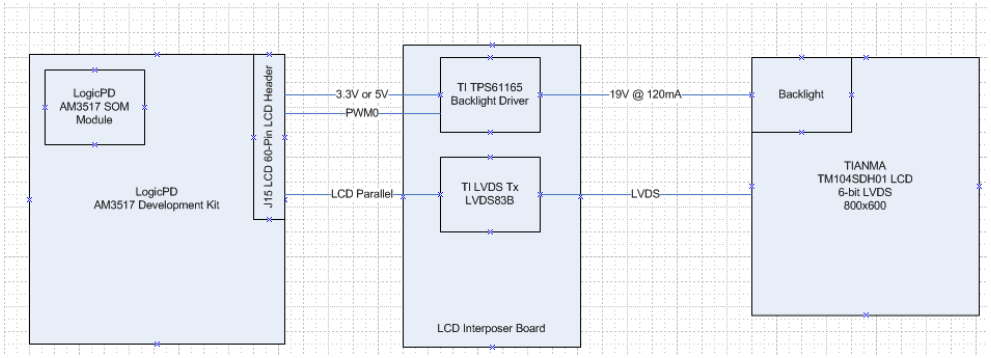
8 Summary

The information in this application note is meant to provide a sample process that can be used to begin integrating LCD displays into your Logic PD SOM designs. If you experience difficulties with your integration, Logic PD offers design support packages through our Products Applications Engineering team. Additional information can be found on the Logic PD support packages web page.

PROJECT	SDK2 LCD ADAPTOR BOARD LVDS
PART NUMBER:	101xxxx
ASSEMBLY NAME:	SDK2-ADP-LCD-LVDS
SCHEMATICS:	BSB

TABLE OF CONTENTS

BLOCK DIAGRAM



IMPORTANT NOTES ABOUT THIS SCHEMATIC

DESIGN NOTE: Example text for the design note to show the note inside the colored box.

1) DESIGN NOTES in grey are information notes.

DESIGN NOTE: Example text for the design note to show the note inside the colored box.

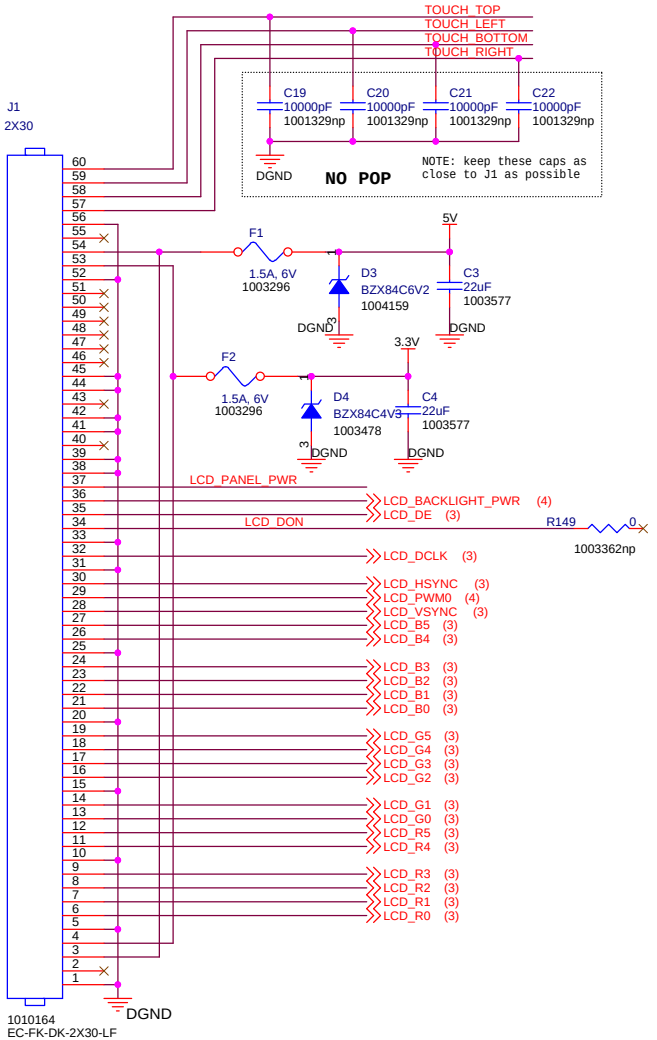
2) DESIGN NOTES in red are critical, and must be understood and followed.

LOGIC PD
LOGIC PD
411 WASHINGTON AVE. N
MINNEAPOLIS, MN 55401
PHONE: (612) 672-9495
FAX: (612) 672-9489

Title SDK2 LCD ADAPTOR BOARD LVDS DISPLAY		Project	SDK2-ADP-LCD-LVDS
Number 101xxxx	Monday, March 07, 2011	Rev A	Sheet 1 Of 5

BASEBOARD I/F, TOUCH, PWR

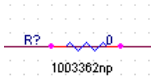
BASE BOARD CONNECTOR



4-WIRE RESISTIVE TOUCH (OPTIONAL)

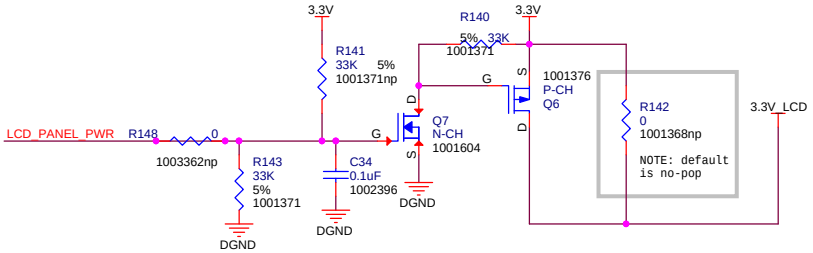
LAYOUT NOTE

IN PLACES WHERE A TRACE APPEARS ACROSS THE RESISTOR, ROUTE THE TRACE ON THE TOP LAYER. THIS ALLOWS THE TRACE TO BE CUT IF ACCESS TO THE SIGNAL IS REQUIRED AT A LATER DATE.



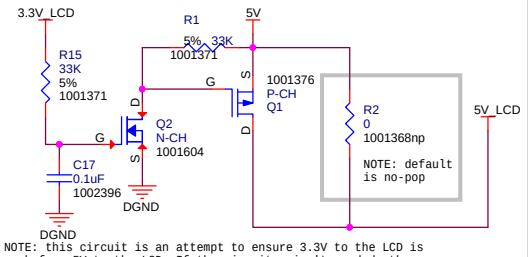
3.3V FOR LCD

(sequencing occurs automatically)



5.0V FOR LCD

(sequencing occurs automatically)

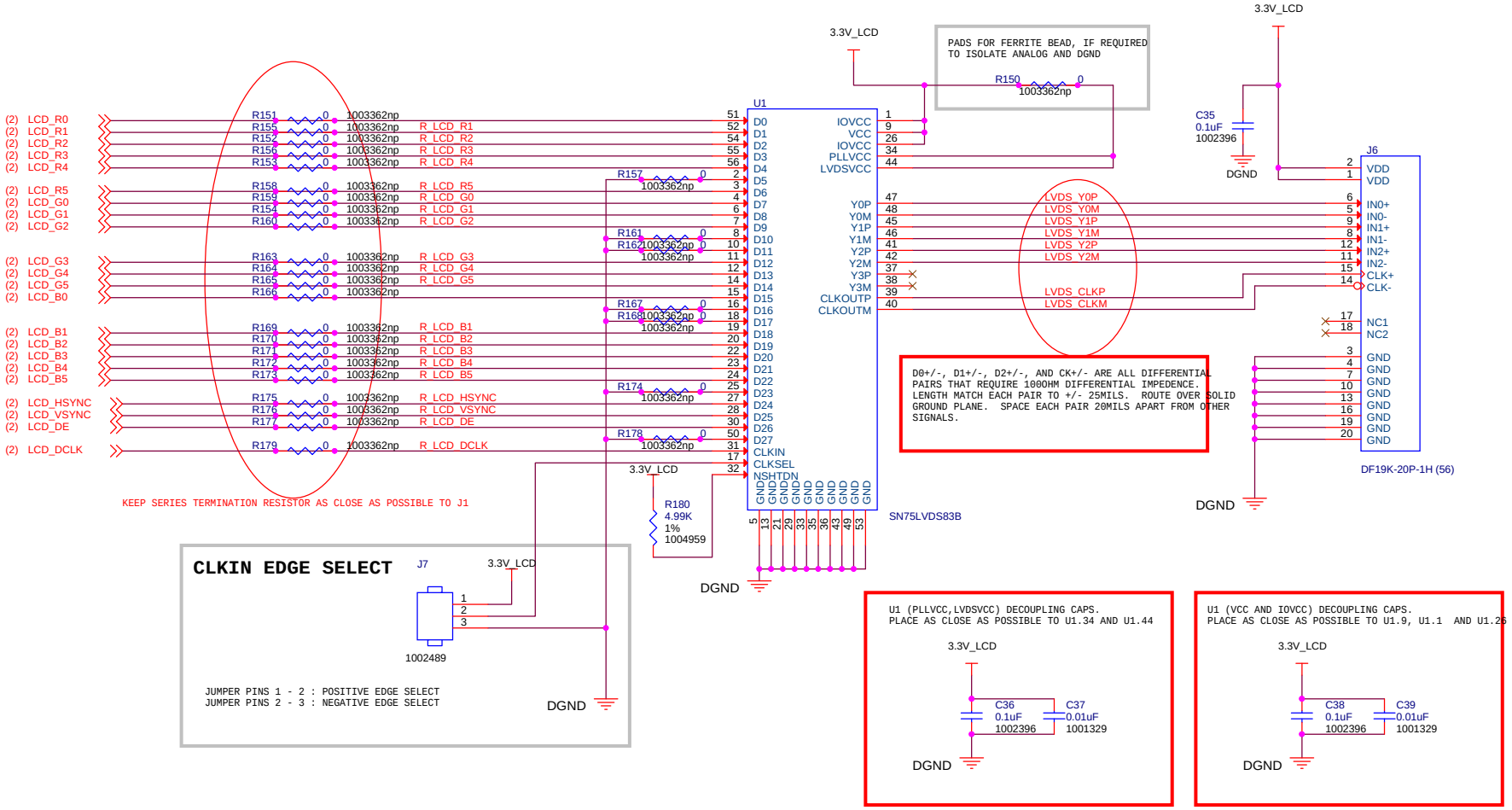


NOTE: this circuit is an attempt to ensure 3.3V to the LCD is up before 5V to the LCD. If the circuitry isn't needed, then no-pop everything except the zero-ohm resistor.

LOGIC PD
411 WASHINGTON AVE. N
MINNEAPOLIS, MN 55401
PHONE: (612) 672-9495
FAX: (612) 672-9489

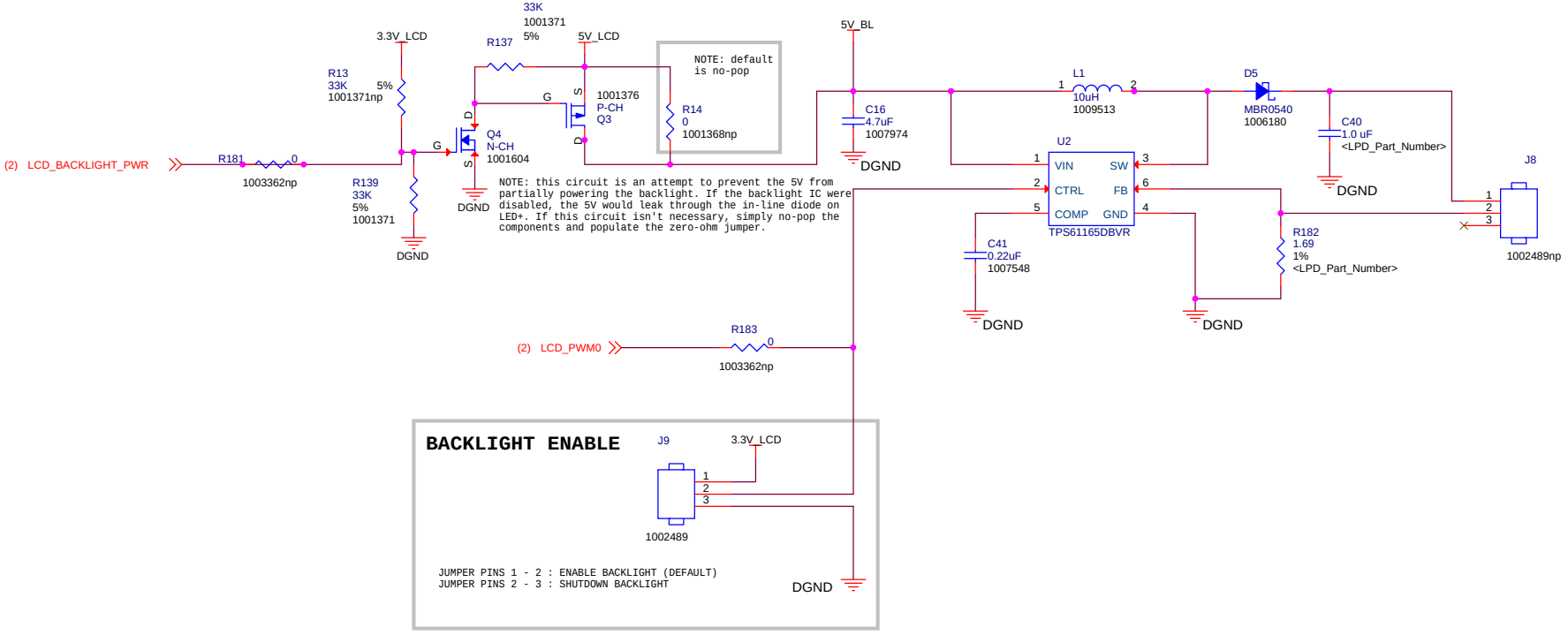
Title SDK2 LCD ADAPTOR BOARD LVDS DISPLAY		Project	SDK2-ADP-LCD-LVDS
Number 101xxxx	Monday, March 07, 2011	Rev A	Sheet 2 Of 5

PARALLEL CMOS TO LVDS



LOGIC PD	LOGIC PD	411 WASHINGTON AVE. N MINNEAPOLIS, MN 55401 PHONE: (612) 672-9495 FAX: (612) 672-9489
Title SDK2 LCD ADAPTOR BOARD LVDS DISPLAY		Project SDK2-ADP-LCD-LVDS
Number 101xxxx	Monday, March 07, 2011	Rev A / Sheet 3 Of 5

BACKLIGHT DRIVER



BACKLIGHT ENABLE

LOGIC PD

LOGIC PD
411 WASHINGTON AVE. N
MINNEAPOLIS, MN 55401
PHONE: (612) 672-9495
FAX: (612) 672-9489

Title SDK2 LCD ADAPTOR BOARD LVDS DISPLAY		Project	SDK2-ADP-LCD-LVDS
Number 101xxxx	Monday, March 07, 2011	Rev A	Sheet 4 Of 5

ECO LIST

REVISION CONTROL			
P/N	REV	DESCRIPTION OF CHANGE	DATE
apha 1	A	INITIAL RELEASE -	03-04-11

LOGIC PD™

LOGIC PD

411 WASHINGTON AVE. N
MINNEAPOLIS, MN 55401
PHONE: (612) 672-9495
FAX: (612) 672-9489

Title SDK2 LCD ADAPTOR BOARD LVDS DISPLAY			Project SDK2-ADP-LCD-LVDS
Number 101xxxx	Monday, March 07, 2011	Rev A	Sheet 5 Of 5