iCE65™ mobileFPGA™ as an LVDS, FPD-Link Display Driver

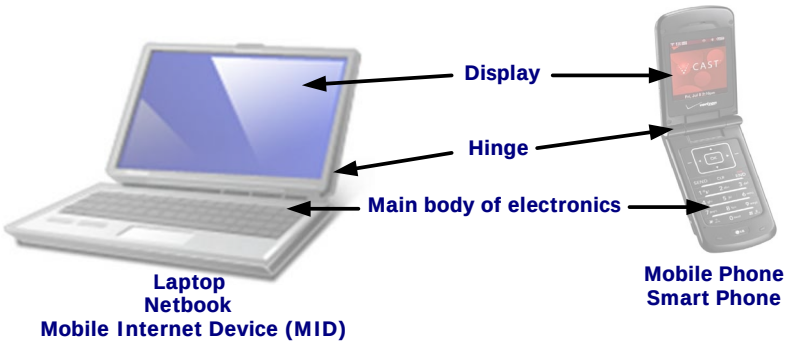
December 1, 2010 (1.2) Application Note AN014

Introduction

Traditionally, color graphic displays employ a TTL or LVCMOS signaling interface. Three separate binary-weighted values represent the red, green, and blue constituent colors of a pixel. Such an interface requires between 21 and 28 signal wires for data and control.

By contrast, many modern high-resolution, large-format graphic displays use an FPD-Link (Flat Panel Display Link) interface, based around the Low-Voltage Differential Swing (LVDS) I/O standard or an LVDS Display Interface (LDI). Both interfaces reduce the number of signal wires required, which is critical for the compact packaging of today's electronics, as shown in Figure 1. The display is typically separated from the main body containing most of the electronics. The main body and the display are mechanically connected by some sort of hinge mechanism. Sending a wide, LVCMOS signal cable bundle across the hinge to the display is simply impractical. Likewise, a custom, wide, flex-cable is prohibitively expensive.

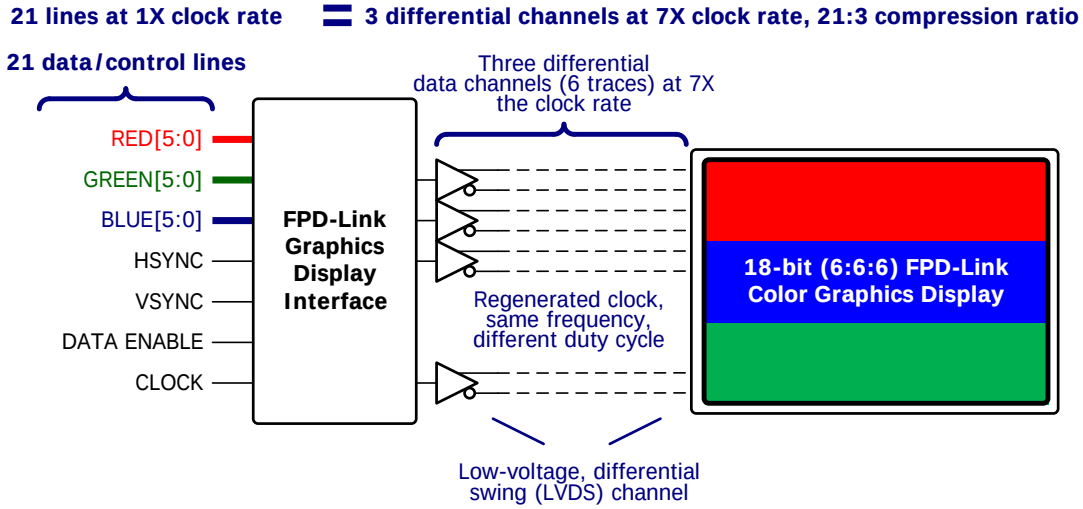
Figure 1: **LVDS-based Display Interfaces Are Critical for Today's Compact Electronics**



FPD-Link Display Basics

Figure 2 shows an example RGB-to-FPD-Link interface. The LVDS outputs from the interface connect directly to an FPD-Link or LDI color graphics display. Figure 2 depicts an 18-bit color interface where three separate 6-bit values represent the balance of red, green, and blue. This interface is sometime represented as (6:6:6), representing three 6-bit values for red, green, and blue. A 24-bit color interface, (8:8:8), adds another LVDS output pair and recodes the output data. Both interfaces have identical timing and electrical requirements, but use a different number of LVDS output pairs and have different data encoding, as described in "18-bit Color (6:6:6) Encoding" and "24-bit Color (8:8:8) Encoding" starting on page 2. For a given resolution, the data bandwidth at each LVDS output put is identical between 18-bit and 24-bit color modes.

Figure 2: **18-bit (6:6:6) FPD-Link Interface to Color Graphics Display**

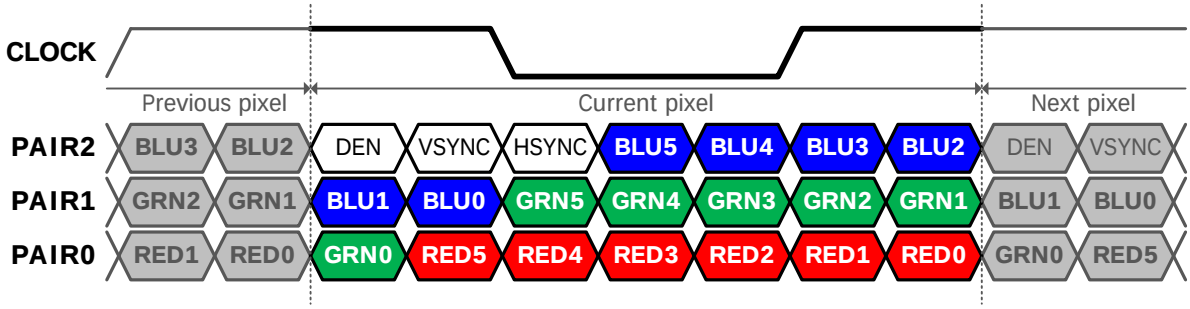


In an 18-bit (6:6:6) design, the 21 data and control signals from the RGB interface are re-encoded onto three LVDS output pairs plus a regenerated clock output. As described later, the output clock has specific duty-cycle and data alignment requirements.

18-bit Color (6:6:6) Encoding

Figure 3 shows the data encoding for the 21 signals in an 18-bit (6:6:6) RGB interface onto three LVDS output data pairs. Each pixel is recoded into seven data transfers across three LVDS pairs. The regenerated clock output has the same frequency as the input clock, but has specific duty-cycle requirements. The rising and falling edges of the associated regenerated output clock signal also provide data alignment information to the receiver.

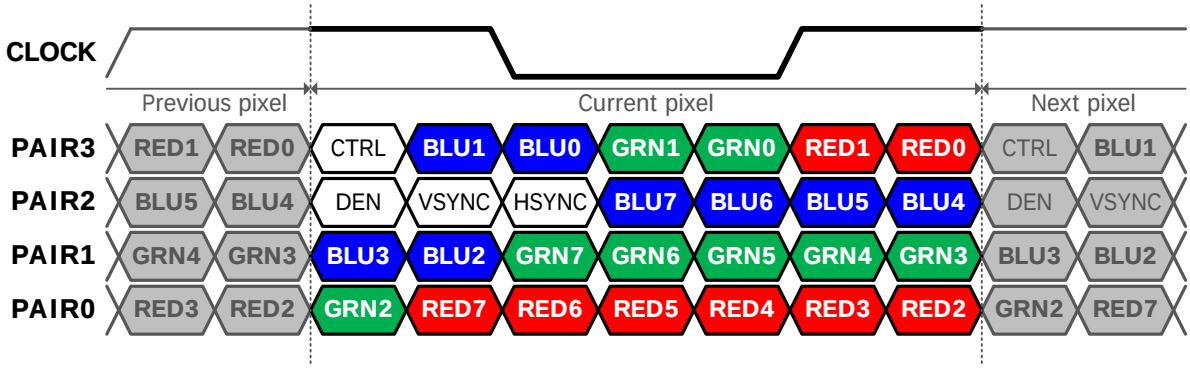
Figure 3: **18-bit Color (6:6:6) FPD-Link Data Encoding**



24-bit Color (8:8:8) Encoding

Figure 4 shows the data encoding for the 27 or 28 signals in a 24-bit (8:8:8) RGB interface onto four LVDS output data pairs. Each pixel is recoded into seven data transfers across four LVDS pairs. Again, the regenerated clock output has the same frequency as the input clock, but has specific duty-cycle requirements. The rising and falling edges of the associated regenerated output clock signal also provide data alignment information to the receiver.

Figure 4: **24-bit Color (8:8:8) FPD-Link Data Encoding**



Data/Clock Rates by Screen Resolution

The iCE65P mobileFPGA family supports LVDS output data rates up to the maximum data shown in Table 1.

Table 1: **Maximum Data Rates for SubLVDS and LVDS on iCE65 mobileFPGAs**

	SubLVDS	LVDS	Units
VCCO_# Voltage	1.8	2.5	Volts
Maximum Data Rate (MAX_RATE)	210	350	Million bits/sec (Mbps)

Equation 1 shows the relationship between the maximum data rate and supported FPD-Link screen resolutions, based on the product of the number of pixels in a video line (including blanking), the number of lines in a vide frame (including blanking), the number of frames per second, and the number of color bits per pixel. This product is then divided by the number of data channels, which is three (3) for 18-bit color and four (4) for 24-bit color.

Equation 1

$$\frac{\left(\frac{\text{Pixels}}{\text{Video Line}} \times \frac{\text{Video Lines}}{\text{Frame}} \times \frac{\text{Frames}}{\text{Second}} \times \frac{\text{Color Bits}}{\text{Pixel}}\right)}{\text{Data Channels}} \leq \text{MAX\_RATE} \frac{\text{bits}}{\text{second} \times \text{channel}}$$

Table 2 lists a few of the supported FPD-Link screen resolutions for iCE65 mobileFPGAs. For higher-resolution displays, there are two competing limits. The iCE65 LVDS I/O pins support data rates up to 350 Mbps, meaning that the display's input clock rate must be 50 MHz or less. Many high-resolution displays support a 50 MHz input clock rate but some require an ever higher clock rate, depending on vendor.

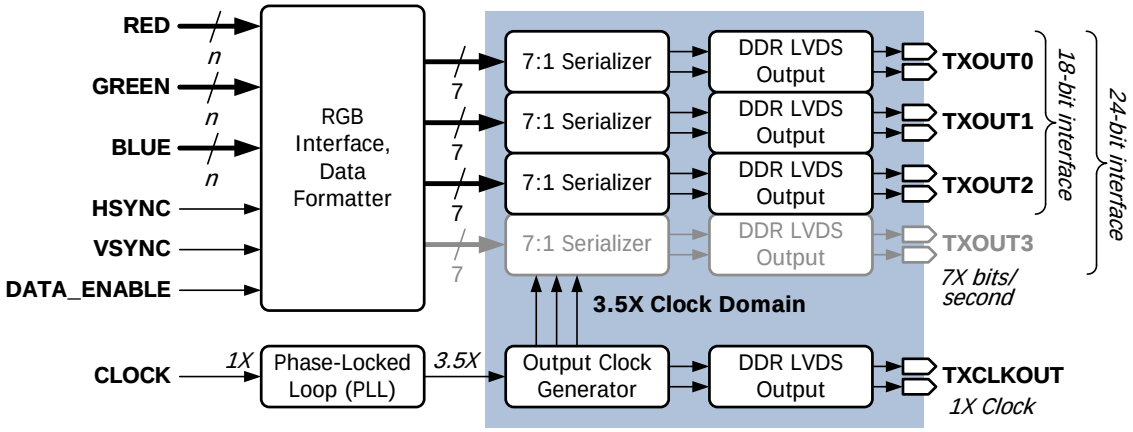
Table 2: **Popular FPD-Link LCD Screen Resolutions Supported by iCE65 mobileFPGAs**

LCD Screen Format	Line Width	Number of Lines	Pixel per Line	Lines per Frame	Frames per Second	Pixel Clock (MHz)	Data Rate per Channel (Mbps)
VGA	640	480	800	525	60	25.20	176
SDTV, 480p	704	480	858	525	59.94	27.00	189
PAL	768	576	960	625	50	30.00	210
SVGA	800	600	1,056	632	60	40.04	280
WSVGA	1,024	600	1,240	638	65	45.00	315
WXGA (HD-ready)	1,280	800	1,300	821	55	49.21	345
HD (720p)	1,366	768	1,494	803	48	50.00	350

iCE65 mobileFPGA Solution

Figure 5 provides a block diagram for an RGB-to-FPD-Link solution implemented in an iCE65 mobileFPGA. The design essentially supports both 18-bit (6:6:6) and 24-bit (8:8:8) color interfaces with slight modification. A 24-bit interface requires an additional serializer and DDR LVDS output, as shown by comparing Figure 4 to Figure 3. RGB data is first formatted into multiple seven-bit bundles for the 7-to-1 (7:1) serializers. The data encoding is different between 18-bit and 24-bit applications, again as shown in Figure 3 and Figure 4.

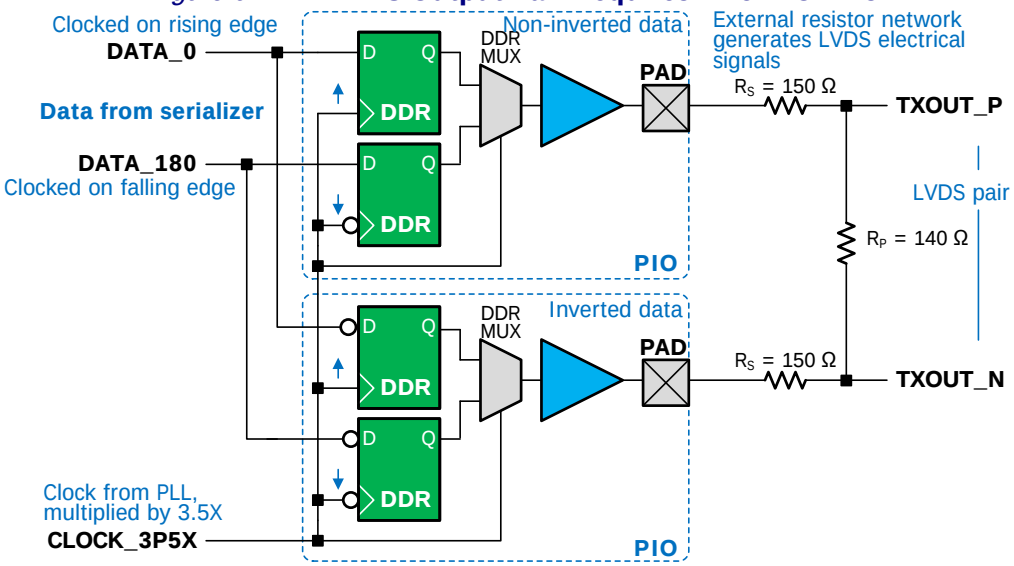
Figure 5: **iCE65 mobileFPGA RGB-to-FPD-Link Solution**



The output data rate is seven times (7X) the input clock frequency. Because of the potentially high output data rates, the design uses Double Data Rate (DDR) flip-flops located in each iCE65 PIO pin.

Each serializer produces two outputs for the DDR LVDS output stage. Each DDR LVDS output, shown in Figure 6, consists of two PIO pins and an external resistor compensation network to generate LVDS signals. The external resistors shown define LVDS output voltage levels; SubLVDS requires different values. One output from the serializer (DATA_0) is clocked out on the rising edge of the 3.5X clock while the other output (DATA_180) is clocked out on the falling edge. The two PIO pins form a complementary LVDS pair. One PIO pin (TXOUT_P) is non-inverting while the other (TXOUT_N) inverts the input data into the DDR output flip-flop.

Figure 6: **DDR LVDS Output Pair Requires Two PIO Pins**



The DDR flip-flops essentially multiply the output data rate by a factor of two. Consequently, most of the internal mobileFPGA logic is clocked at 3.5 times (3.5X) the input clock rate instead of 7 times the clock rate, which has the following effects on the design.

1. Clocking internal logic at half the output rate effective doubles the available timing margin for most of the design.

2. Clocking internal logic at half the output rate also requires that data is alternatively clocked on rising and falling clock edges.

The 3.5X internal clock is generated by the iCE65P's internal phase-locked loop (PLL). The output from the PLL is phase aligned to the input clock.

Based on available characterization data, keep the LVDS output data rate below 350 million bits per second (350 Mbps), which equates to an input clock rate 50 MHz (350 Mbps divided by 7) or slower. Faster performance may be possible pending further characterization.

Clocking

Generating Serializer Clock

FPD-Link applications require a data rate that is seven times (7X) faster than the basic pixel clock rate.

Instead of using a 7X clock within the application, many FPD-Link solutions employ an internal clock that is 3.5 times (3.5X) faster than the pixel clock. The 3.5X clock then drives double-data-rate (DDR) output flip-flops that generate a 7X output data rate (3.5X times 2).

There are several advantages to using a 3.5X clock instead of a 7X clock.

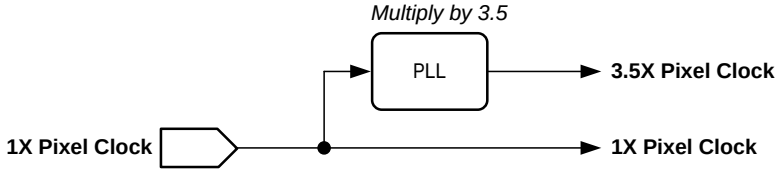
- Because the 3.5X clock operates at half the frequency of a 7X clock, it doubles the clock period which allows more levels of logic between clock edges and simplifies timing requirements.
- Because the 3.5X clock operates at half the frequency, it also cuts the power consumption in half for the design.

Generating a 3.5X generally requires a Phase-Locked Loop (PLL). Fortunately, the iCE65 P-Series mobileFPGAs include an on-chip PLL. The two possible solutions appear below.

Generating 3.5X from Pixel Clock Input

If the pixel clock is already available in the application, then the PLL in the iCE65P mobileFPGA multiplies the input clock by 3.5 to generate a 3.5X pixel clock, as shown in Figure 7.

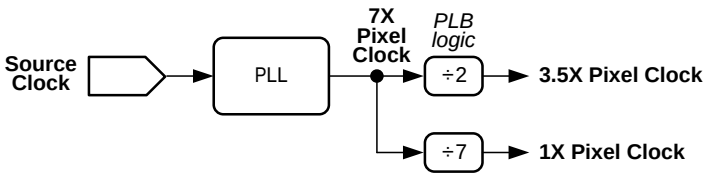
Figure 7: **Generating 3.5X Pixel Clock using PLL**



Generating 1X and 3.5X Pixel Clock from General Clock Frequency

In many applications, the system clock is not a desirable pixel clock. In these applications, use the PLL to generate a frequency that is seven times (7X) faster than the desired pixel clock frequency. Using PLB logic, divide the 7X clock by 7 to generate the desired pixel clock and divide by 2 to generate the 3.5X pixel clock, as shown in Figure 8.

Figure 8: **Generating 1X and 3.5X Pixel Clock from General Clock Frequency**

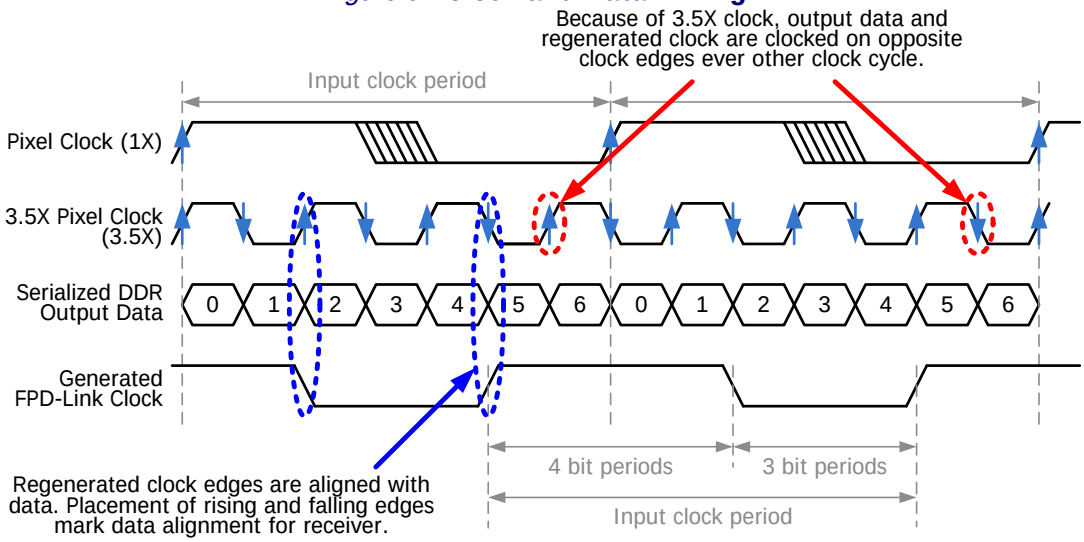


Due to the very high frequency of the 7X clock, the divide-by-7 circuitry requires careful design and placement. See "Placing Logic using a Logic Region" on page 11 for detailed step-by-step instructions. The divide-by-2 circuit is a simple toggle flip-flop.

FPD-Link Clock and Data Timing

Figure 9 shows the clock timing that drives most of the decisions in the design. The Phase-Locked Loop (PLL) and all logic driving the FPD-Link interface are clocked by the rising edge of the input clock. The PLL generates an internal clock that is 3.5 times faster than the input clock.

Figure 9: **Clock and Data Timing**



New graphic data is clocked on every rising edge of the input clock. The Output Clock Generator detects the rising clock edge and generates all the load controls for the serializers, as described in "Serializer Design" on page 6.

The outputs from the 7-to-1 serializers are clocked into the DDR LVDS outputs on both the rising and falling edges of the 3.5X clock form the PLL. One of the tricks of the serializer design is that data is clocked on the rising edge during one input clock cycle and clocked on the falling edge during the next cycle.

The generated LVDS clock has the same frequency as the input clock signal. However, the output clock has a different duty-cycle and phase relationship. The rising or falling edge of the output clock defines the data alignment for the display's receiver. The falling clock edge always coincides with the start of data output bit 2 and the rising edge always coincides with the start of data output bit 5. Similarly, the generated output clock is synchronized with the data output.

Serializer Design

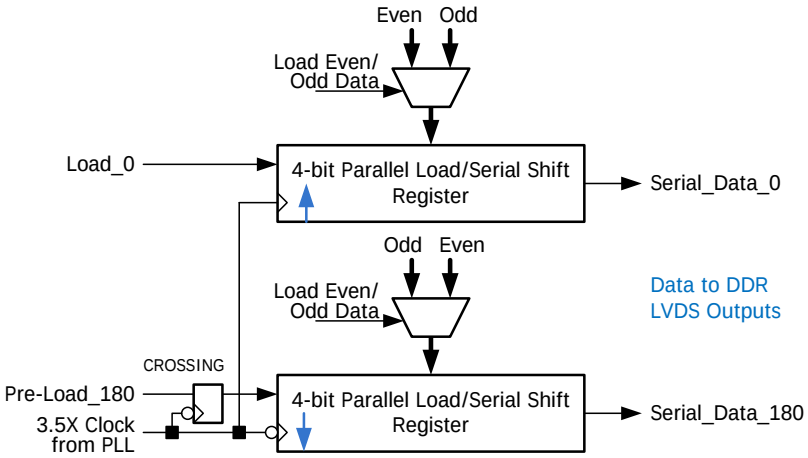
The serializer design is perhaps the most difficult challenge in the application because of the 3.5X clock. Data is clocked on both edges of the 3.5X so that the final output data is seven times the input data rate. Various solutions are possible but the challenges are as follows.

- Serialized data is clocked on both edges of the 3.5X clock.
- At various points in the design, there is a critical timing path that crosses from one 3.5X clock edge to the other. Essentially, there are paths that operate at 7X the input clock rate. Ideally, for maximum performance, these crossings are few and have a single load or connection to other logic.
- Various design trade-offs are possible that use additional logic to improve performance.

Figure 10 provides an overview of the solution used in the demonstration design. Two bits of serialized data are sent to the DDR LVDS output during every 3.5X clock cycle—one bit for the rising edge, one bit for the falling edge.

The seven bits of parallel data are loaded into two four-bit shift registers. One register is clocked on the rising edge of the 3.5X clock, the other on the falling edge. Because seven bits are clocked out every 3.5X clock cycle, the shift registers load even-numbered bits during one serialization period and the odd-numbered bits during the next, as shown in Figure 11.

Figure 10: **Serializer Design Block Diagram**



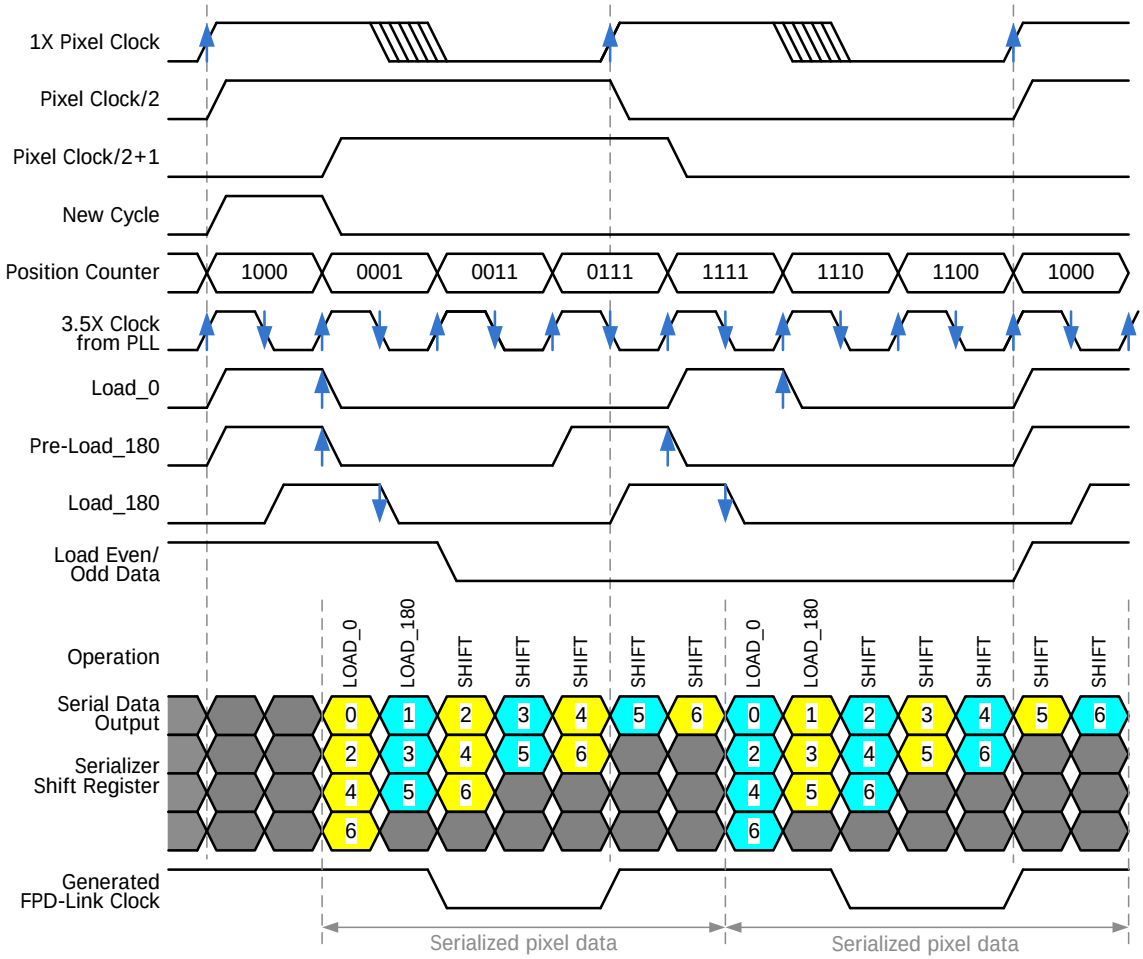
All of the control logic is generated by the Output Clock Generator, shown in Figure 5 and clocked by the rising edge of the 3.5X clock to simplify timing closure. Additionally, all timing paths are as relaxed as possible. Parallel input data and all controls are stable at least a full 3.5X clock cycle before they are used.

The controller synchronizes to the rising edge of the input clock, when new pixel data is available. The controller detects a New Cycle and resynchronizes every two incoming pixels.

A Johnson counter tracks the serializer bit position between synchronizations. The Johnson counter reduces overall power consumption and to simplify timing. The counter tracks data for two input clock cycles or two pixels. The Johnson counter also simplifies the decoding for all control outputs.

The control logic indicates when new parallel data is loaded into the serializer's shift registers and whether to load even or odd data. The load control for the falling-edge clocked shift register, Pre-Load_180, is sent to the serializers one cycle early, on the rising edge of the 3.5X clock. The load control is then re-clocked in each serializer onto the falling edge as Load_180, making the load control stable for a full 3.5X clock cycle before it is used.

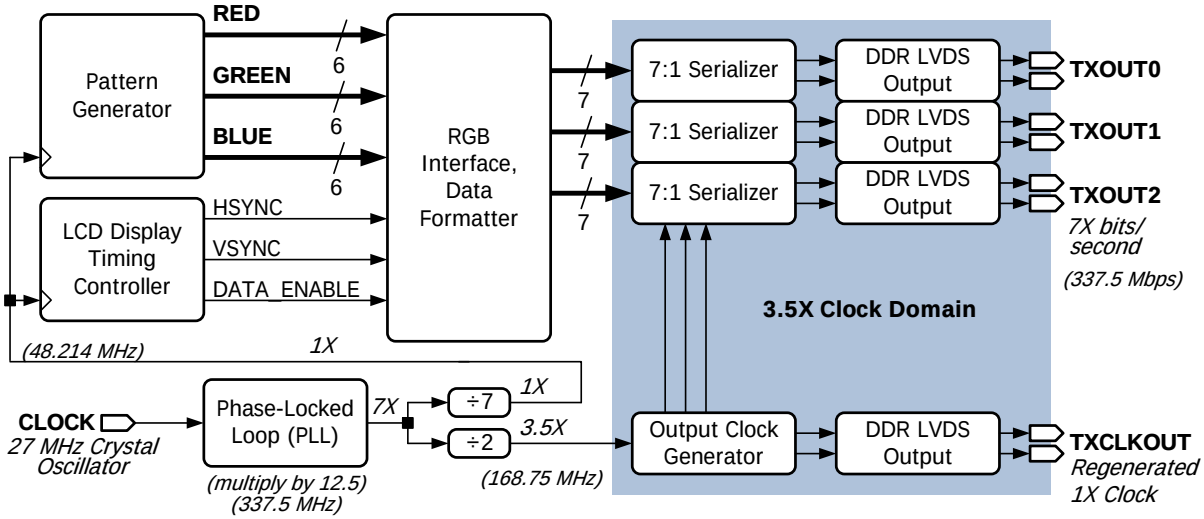
Figure 11: **Serializer Control Timing**



iCEman65P Demonstration Application

The design was validated in hardware by connecting a SiliconBlue iCEman65P evaluation kit to a HannStar 10-inch, 1024x600 WSVGA color display. The HannStar display supports an 18-bit (6:6:6) FPD-Link style interface. As shown in Figure 12, the demonstration design consists of the RGB-to-FPD-Link interface described above. A timing control block generates the proper timing for a 1024x600 WSVGA display. A pattern generator provides the red, green, and blue output data.

Figure 12: **FPD-Link Demonstration Application for iCEman65P**



Clocking for Application Example

The iCEman65P board includes a 27 MHz crystal oscillator. Using the mobileFPGA's PLL, the 27 MHz clock is multiplied by 12.5 to generate a 337.5 MHz clock, which is 7 times faster than the desired pixel clock. The 7X clock is then divided by 7 to generate the 48.214 MHz pixel clock and the 168.75 MHz clock for the serializer (3.5X clock). The 48.214 MHz pixel clock generates a rate slightly faster than 60 frames per second, for demonstration purposes. Figure 13 shows the Synplicity clock constraints.

Figure 13: **Synplicity Clock Constraints**

Clock Object	Clock Alias	Frequency (MHz)	Period (ns)	Clock Group
n:PIXEL_CLOCK_GENERATOR.CLK_7X	CLK_7X	337.5	2.962962963	CLK_GRP_7X
n:PIXEL_CLOCK_GENERATOR.CLK_1X	CLK_1X	48.21	20.74258453	CLK_GRP_1X
n:PIXEL_CLOCK_GENERATOR.CLK_3P5X	CLK_3.5X	168.75	5.925925926	CLK_GRP_3P5X
p:CLOCK_27MHz	CLK_27MHz	27	37.03703704	CLK_GRP_7X

Connector to LCD Display

The LCD panel connects to I/O Bank 1 on the iCEman65P board, which already includes the external LVDS output compensation resistors on selected pairs. Table 3 provides a detailed connection list. The connection to the iCEman65P board is via Header J18, a 40-pin stake-pin header with 0.1" (2.54 mm) pin spacing. The connection to the HannStar display is via a FI-XB30SSRL-HF16 (JAE or equivalent) connector.

Table 3: **iCEman65P Connection to HannStar Display**

HannStar Display FI-XB30SSRL-HF16 (JAE or equiv)		iCEman65P04 Header J18		Example Application	FPGA Ball
Display	Signal	Pin	Signal		
Pin3:	VDD(+3.3V DC)	29	VCCIO_1	VCCIO_1	VCCIO_1
Pin4:	VDD(+3.3V DC)	29	VCCIO_1	VCCIO_1	VCCIO_1
Pin5:	Backlight ADJ	1	B1-IO00	—	D20
Pin8:	Data0-	22	B1-DP04_N	TXOUT_N[0]	P16
Pin9:	Data0+	21	B1-DP04_P	TXOUT_P[0]	R18
Pin10:	GND	12	GND	GND	GND
Pin11:	Data1-	18	B1-DP02_N	TXOUT_N[1]	P18
Pin12:	Data1+	17	B1-DP02_P	TXOUT_P[1]	P15
Pin13:	GND	12	GND	GND	GND
Pin14:	Data2-	16	B1-DP01_N	TXOUT_N[2]	N15
Pin15:	Data2+	15	B1-DP01_P	TXOUT_P[2]	N16
Pin16:	GND	30	GND	GND	GND
Pin17:	CLK-	14	B1-DP00_N	TXCLKOUT_N	E20
Pin18:	CLK+	13	B1-DP00_P	TXCLKOUT_P	F20
Pin26:	VDD_LED_Backlight(+5V DC)	11	+5 VDC	—	—

Color Bar Application Example

Figure 14 is a photograph showing the HannStar display panel, driven by the iCEman65P evaluation kit board, with a simple color-bar test pattern. The HannStar display panel connects to the 40-pin header connected to the mobileFPGA's I/O Bank 1 (note the orientation of the board in the figure). The display panel also requires a 3.3V supply, which is provided from the iCEman65P board by setting VCCIO_1 to 3.3V.

Figure 14: **HannStar 10-inch LCD Display with FPD-Link Interface Driven by iCEman65P Evaluation Kit**

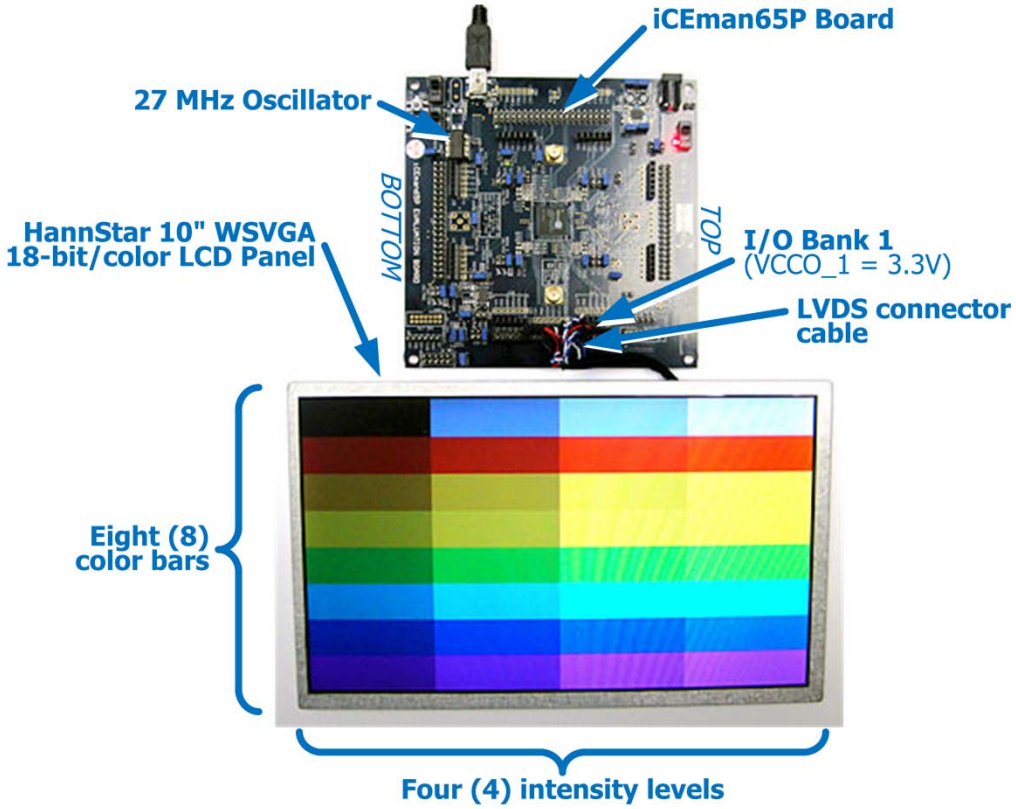
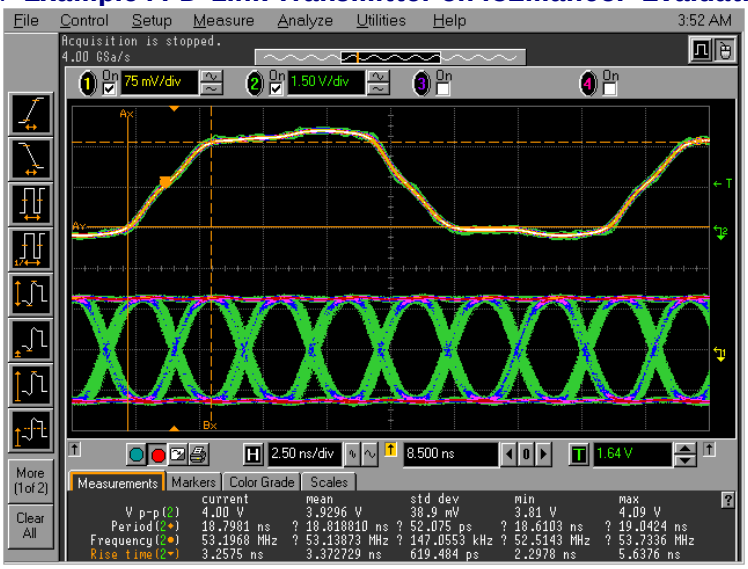


Figure 15 shows an example oscilloscope output of the FPD-Link interface. The top trace is the generated output clock signal and the bottom trace is one of the LVDS data outputs.

Figure 15: **Example FPD-Link Transmitter on iCEman65P Evaluation Board**



Application Example Design Files

The application design files are available from your local SiliconBlue sales representative.

Appendix

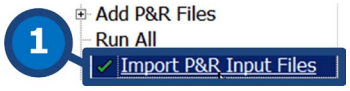
Placing Logic using a Logic Region

The following steps describe how to create a logic region, how to create a logic group, and how to assign logic to a group.

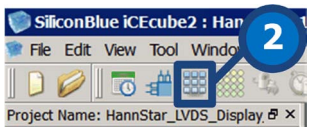
Creating a Logic Region

Follow these steps to create a logic region.

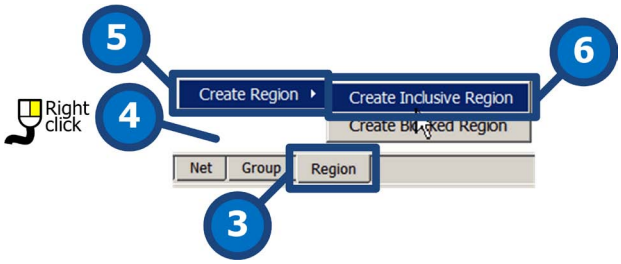
1. After synthesizing the design in Synplicty, import the design into iCEcube by double-clicking **Import P&R Input Files**.



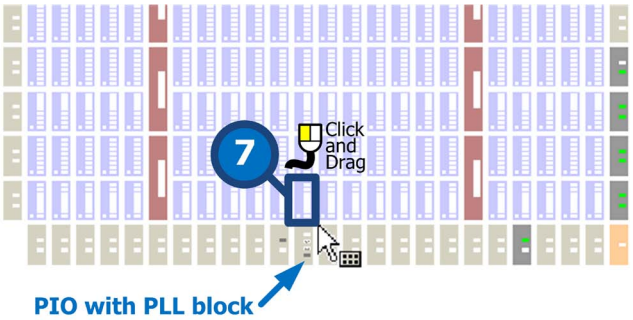
2. Open the **Floor Planner** by clicking on the Floor Planner icon in the iCEcube tool bar.



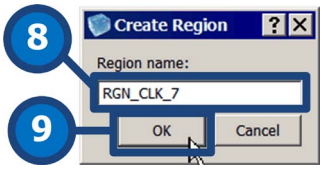
3. Click the **Region** tab.
4. **Right-click** in the pane.
5. Click **Create Region**.
6. Click **Create Inclusive Region**.



7. Locate the Programmable Logic Block (PLB) immediately above the PLL at the center of the bottom edge of the iCE65P04 floor plan. Click and drag to select the entire PLB.



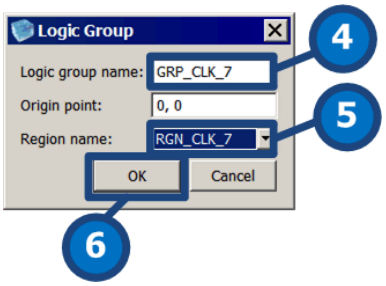
8. Type a name for the region, `RGN_CLK_7`.
9. Click **OK**. The region is now defined.



Creating a Logic Group

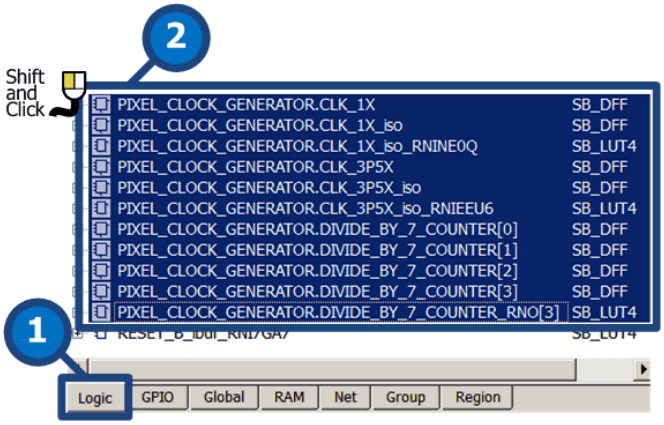
1. Click the **Group** tab.
2. **Right-click** in the pane.
3. Select **Create Logic Group**.

4. Type a name for the logic group, `GRP_CLK_7`.
5. Assign the logic group to a region from the drop list.
6. Click **OK**. The logic group is defined and is assigned to a region.

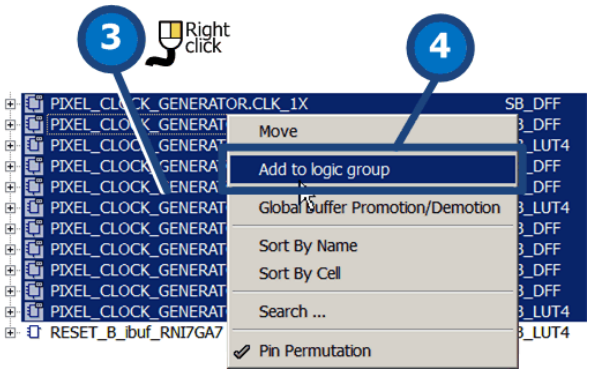


Assigning Logic to a Group

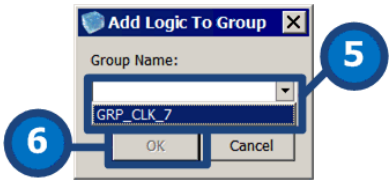
1. Click the **Logic** tab.
2. While holding either the **Shift** or the **Ctrl** key, select the desired logic to be added the group. In this example, it is all the logic in the `PIXEL_CLOCK_GENERATOR` module.



3. **Right-click** on the selection.
4. Select **Add to logic group**.

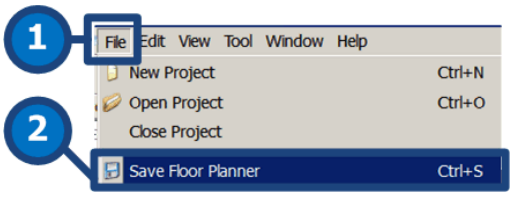


5. Assign the selected logic to a previously defined logic group name using the drop list.
6. Click **OK**. The selected logic is now assigned to a logic group and the logic group was previously assigned to a region.

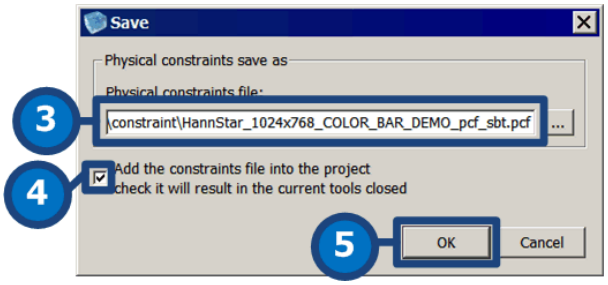


Save the Floor Plan

1. From the iCEcube menu bar, select **File → Save Floor Planner**.

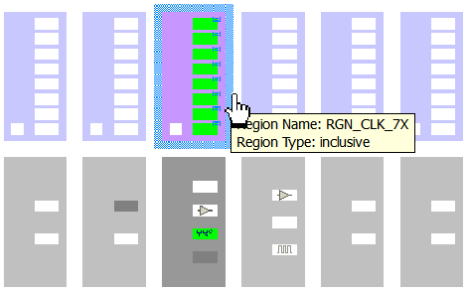


3. Generally, use the default name. Otherwise, specify a name for the constraints file.
4. Check **Add the constraint file** option
5. Click **OK** to save the constraints file and add it to the current project.



Process Design

The design must be processed using the iCEcube software.



Related Documents

- HannStar, *HSD100IFW1 10" Color TFT-LCD Module*
  www.siliconbluetech.com/media/downloads/Hannstar10inchHSD100IFW1-A00.pdf
- SiliconBlue, AN008: *Using Differential I/O (LVDS, SubLVDS, LVPECL) in iCE65 mobileFPGAs*
  www.siliconbluetech.com/media/downloads/SiliconBlue_AN008_LVDS.pdf
- National Semiconductor, AN-1032: *An Introduction to FPD-Link*
  www.national.com/an/AN/AN-1032.pdf
- National Semiconductor, AN-1127: *LVDS Display Interface (LDI) TFT Data Mapping for Interoperability with FPD-Link*
  www.national.com/an/AN/AN-1127.pdf

Revision History

Version	Date	Description
1.2	1-DEC-2010	Corrected I/O bank and voltage settings in Figure 14. Updated Figure 3 and Figure 4.
1.1	19-NOV-2010	Added support for 1280x800 (WXGA, HD-ready) and 1366x768 (HD, 720p) displays in "Data/Clock Rates by Screen Resolution" section on page 3.
1.0	5-OCT-2010	Initial release.

SiliconBlue Technologies Corporation

3255 Scott Blvd.
Building 7, Suite 101
Santa Clara, California 95054
United States of America

Tel: +1 408-727-6101
Fax: +1 408-727-6085

www.SiliconBlueTech.com