LCD Controller Design

System Overview

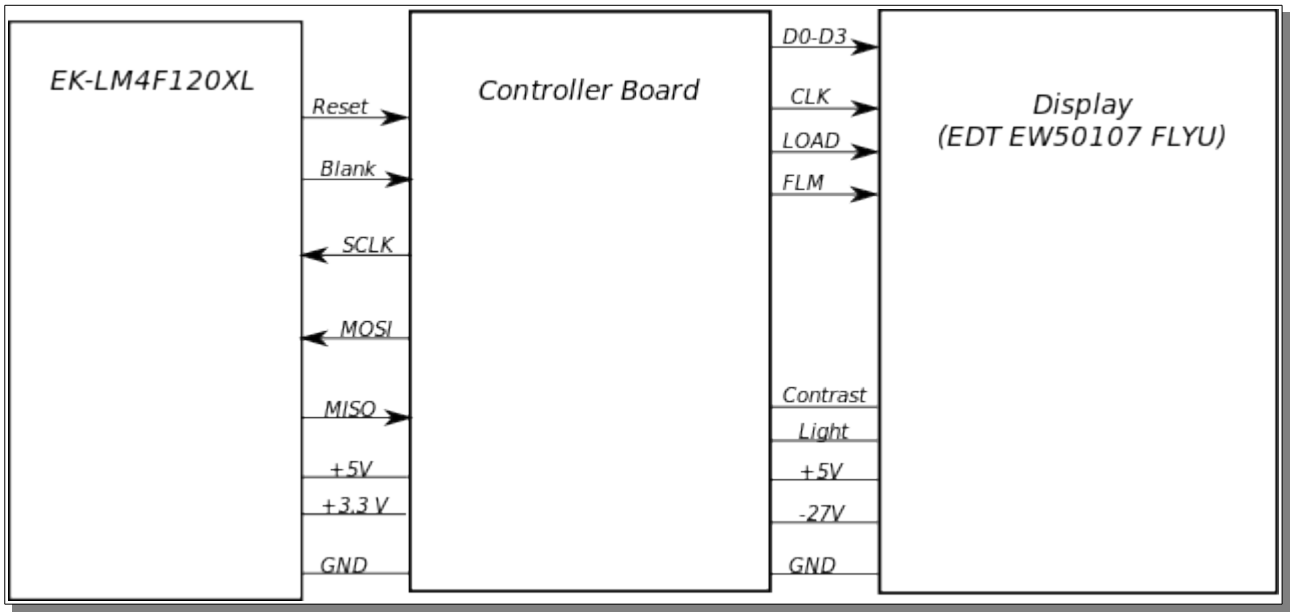

Figure 1:

Name	Type	Funciton
Reset	GPIO	Reset the MSP on the Controller Board
Blank	GPIO	Signal the MSP to stop SPI and turn off the display. Also used to start the SPI transfer after the Stellaris is ready
SCLK,MOSI,MISO	SPI	SPI bus pins. Controller has jumpers to select one of the four controllers on the Stellaris
+5V,+3.3V,GND	Power	Power Supply to controller and display

Table 1: Stellaris Pin Functions

Buffer Memory Layout

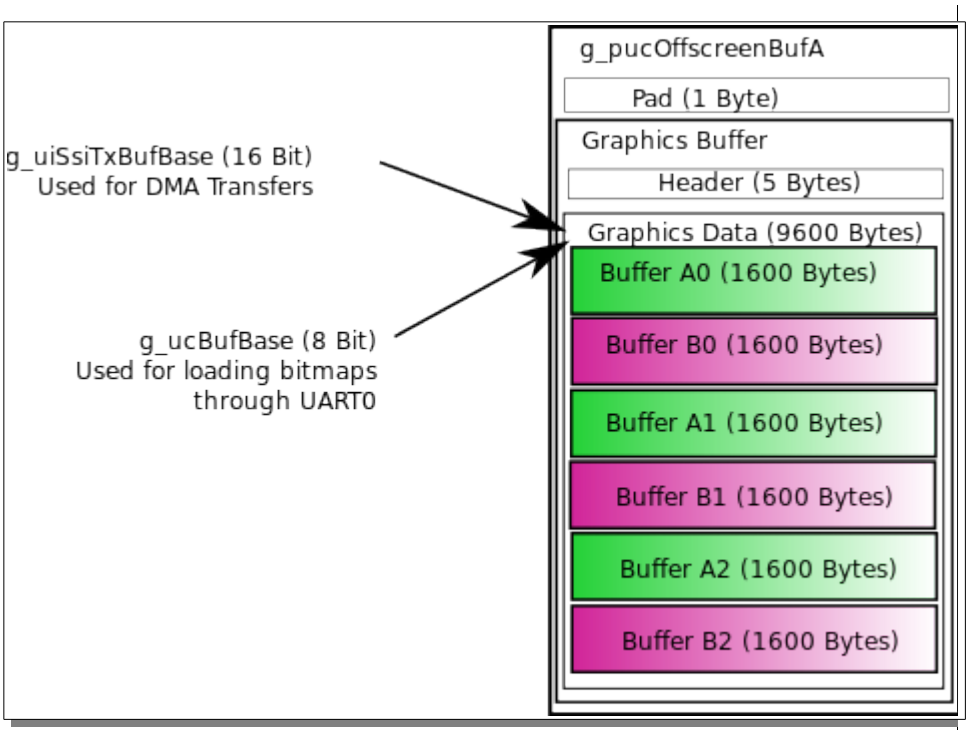

Figure 2: Graphics Buffer Layout

The display buffer layout is shown in Figure 2. In order for the uDMA transferrs to work correctly, the Graphics data needs to be aligned on an even address. The grlib 1bpp format adds a 5 byte headder in front of the actual graphics data. In order get the alignment correct, the g_pucOffscreenBufA array is defined using the *__attribute__ ((aligned(2)))* gcc directive, forcing the array to be on an even boundry. In addition, we leave a 1 byte pad before the grlib image structure.

Data Transfer Design

The Stellaris is set up as a 16bit SPI slave using the uDMA controller to keep the buffer filled and to discard the recieved data into a dummy buffer. The uDMA transfers are set up as a ping pong buffer. See Figure 2 for the details of the buffer layout. After initialization, the primary uDMA buffer is set to Buffer A0 and the secondary buffer is set to Buffer B0. When the A0 buffer has been transfered, the uDMA controller switches to the B0 buffer and genrates an interrupt. The interrupt handler detects that Buffer A0 is done and sets up Buffer A1 as the new primary buffer. When Buffer B0 is empty, the transfers now come from Buffer A1 and the secondary buffer is set to B1. Next Buffers A2/B2 are used. On the next interrupt the sequence starts back at Buffers A0/B0.

The uDMA transferrs are triggerd when the SSI FIFO buffer has 4 empty slots and is configured as a burst transfer of 4 16bit values. This keeps CPU overhead to a minimum.