Bit banging VGA with the Cypress PSoC

David Lawrence

June 5, 2013

Abstract

Bit banging VGA from an embedded device presents a challenge due to the extremely high bandwidth of a video signal: a typical VGA signal has a pixel clock in excess of 20 MHz. This document describes how the Cypress Programmable System on a Chip (PSoC) may be used to efficiently display an color image of approximately 12000 pixels at 60 frames per second.

1 VGA hardware

VGA is a mixed analog and digital signal. It is comprised of two TTL signals for horizontal and vertical synchronization, and three analog signals that represent the red, green, and blue components of each pixel's color.

Each analog signal is driven using two digital IO pins connected to a resistor ladder—i.e. a crude 2-bit DAC. This limits the color space to 64 possible values, but offers extremely high bandwidth. Each sync signal is driven directly from a digital IO pin. Thus the VGA signal is produced by a total of eight digital outputs from the PSoC, as shown in Figure 1.

2 Clocks and timing

A VGA signal with 1024 by 768 resolution at 60 frames per second requires a pixel clock of 65 MHz [1]. This is just within reach of the PSoC, whose maximum processor clock speed is 67 MHz. The PSoC is configured to generate a 65 MHz master clock using its internal clock scaler locked to a 24 MHz crystal.

As per [1], each line consists of 1024 pixels of color information, followed by 24 blank pixels, 136 blank pixels with the horizontal sync asserted, and 160 more blank pixels. Each frame consists of 768 lines of color information, 3 blank lines, 6 blank lines with the vertical sync asserted, and 29 more blank lines.

Hardware timers are used to count the pixels in each line and the lines in each frame, as well as to control the timing of the sync pulse within the blanking period for each line and frame. This arrangement is shown in Figure 2. Note that the sync signals are generated entirely by the hardware, and so no code is needed to ensure their proper operation. A hardware mux is used to shut off the color channels during blanking periods.

3 Displaying pixels

1024 by 768 is too large a resolution for a PSoC-based embedded device: a single frame would consume 768 kilobytes of memory, while the PSoC only has 64 kilobytes of SRAM. Furthermore, it would take minutes for a client device to send a single frame to the PSoC, even over a fast serial connection.

For this reason, the "native resolution" of the PSoC's VGA output is chosen to be 128 by 96. Each 8 by 8 square of pixels in the VGA output is grouped into a single logical pixel in the PSoC software. This reduces the output bandwidth required by a factor of 8, since the PSoC's digital outputs only need to be updated once every eight pixels. To complete the scaling, each line is repeated eight times.

The PSoC's direct memory access (DMA) controller is used to copy one byte at a time from SRAM to the digital outputs. Because the DMA controller uses dedicated hardware, the processor core will be left free for other tasks. When the DMA controller is in free-running mode, it copies one byte every eight clock cycles—exactly what is needed.

The PSoC DMA component is configured to begin copying at the start of each line and to fire an interrupt as soon as the last pixel of the line has been copied. The interrupt handler points the DMA controller to the next line to be displayed by setting the `CY_DMA_TDMEM_STRUCT_PTR[dma_td].TD1` register to contain the address of that line in memory.

4 Memory layout and bus contention

Some care must be taken to ensure that other PSoC components do not interfere with the operation of the DMA controller. For example, if the processor core attempts to access SRAM at the same time as the DMA controller tries to copy a byte from SRAM, the PSoC's bus arbitration will delay the DMA controller by one clock cycle. At lower speeds this would not be an issue, but with the DMA controller in free running mode the bus arbitration is clearly visible as a rightward shift and jitter in the VGA output.

This is quite a serious problem, since it is almost impossible for the processor to do anything useful without accessing variables in SRAM. Fortunately, the PSoC's memory architecture allows a solution: system memory is actually split into two 32 kilobyte SRAMs connected to separate memory buses. If the display buffer were located in one SRAM and everything else (including the heap and stack) was located in the other SRAM, all would be well.

The desired memory layout is achieved by modifying the linker settings used by PSoC Creator. System memory still begins at address 0x1fff8000, but its length is reduced by 32 kilobytes. A new section called "vram" is defined to begin at address 0x20000000 (formerly occupied by regular system memory), also of length 32 kilobytes. By default the linker places everything in system memory; to override this, the definition of the video buffer is annotated with "`__attribute__ ((section(".vram")))`". The full linker configuration is in the file `custom.ld` in the PSoC project directory.

5 Memory operations and the serial API

The frame buffer can only be modified by the processor during blanking periods, since any access to the buffer while the DMA controller was active would cause bus contention. Therefore a second frame buffer is stored in the regular system memory. When the processor is ready to have the system frame buffer copied into the video frame buffer, it sets a flag. The end-of-line interrupt handler watches for that flag and, once it is set, copies the system buffer into the video buffer during the lengthy vertical blanking period.

The processor modifies the system frame buffer in response to commands issued over the serial port. The serial API is byte-based and uses a system of command bytes and data bytes. A "command byte" is any byte with the high-order bit set, and a "data byte" is any byte with the high-order bit clear. Any command byte may be followed by zero or more associated data bytes. The commands are listed in the following table:

Command	Byte	# data bytes
Send 16 by 12 image	0x80	192
Send 32 by 24 image	0x81	768
Send 64 by 48 image	0x82	3072
Send 128 by 96 image	0x83	12288

The serial port is driven by a PSoC UART component running at 19200 baud, which is connected to a MAX202 line driver. Thus it takes approximately 0.1 seconds to send a 16 by 12 image, and approximately 10 seconds to send a 128 by 96 image. It is expected that a user with higher video bandwidth requirements would implement additional command bytes as needed (e.g. "change color of pixel at coordinates", "draw circle", etc), because although the serial bus is saturated, the PSoC processor core is very much underutilized.

6 Python driver

A simple Python driver is provided which displays the contents of a two-dimensional array of RGB color values. The function `test()` in `vga.py` will write a 128 by 96 test image to an attached VGA board.

References

[1] XGA Signal 1024 x 768 @ 60 Hz timing
`http://tinyvga.com/vga-timing/1024x768@60Hz`

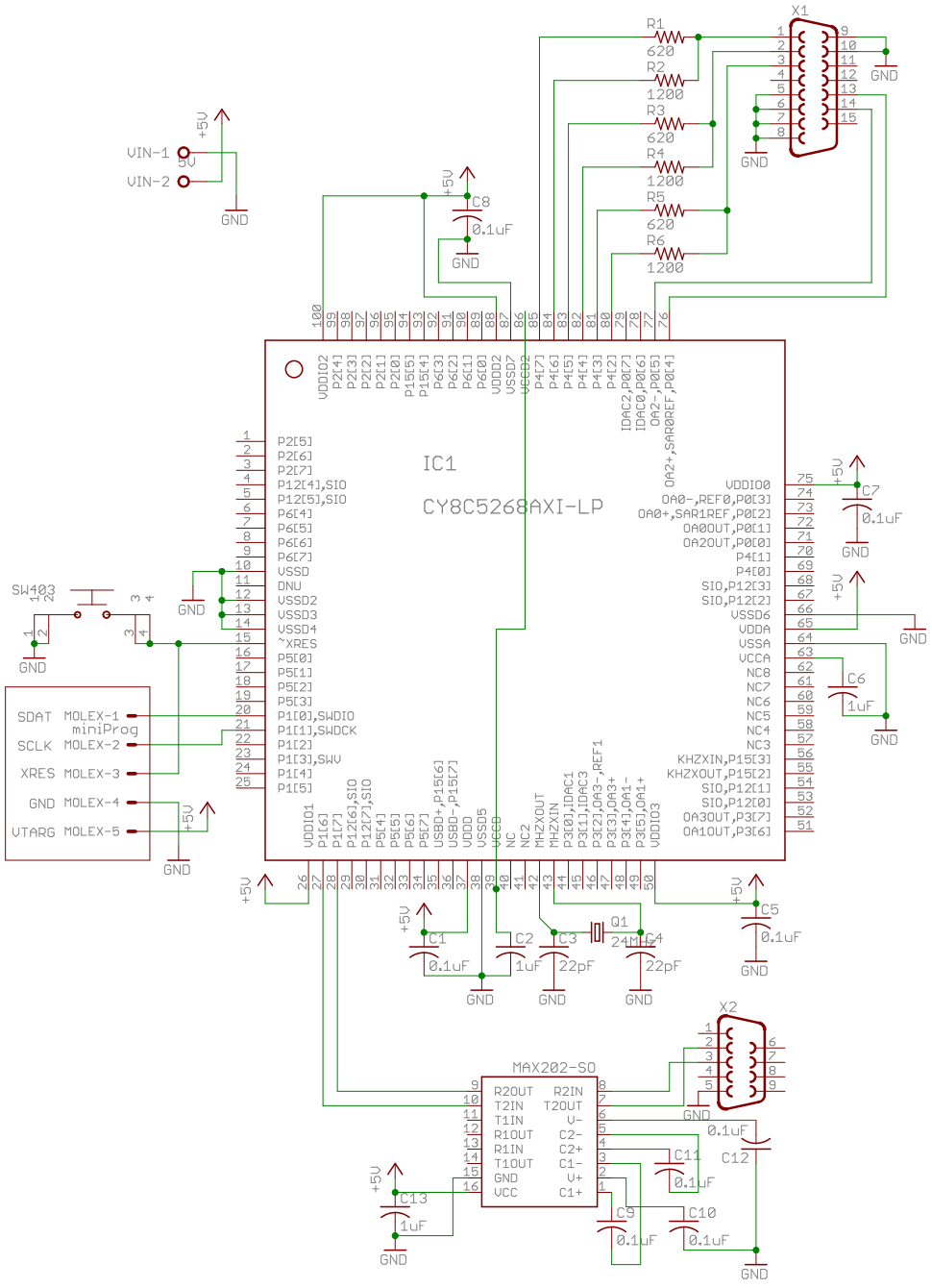

Figure 1: Hardware connections to the PSoC.

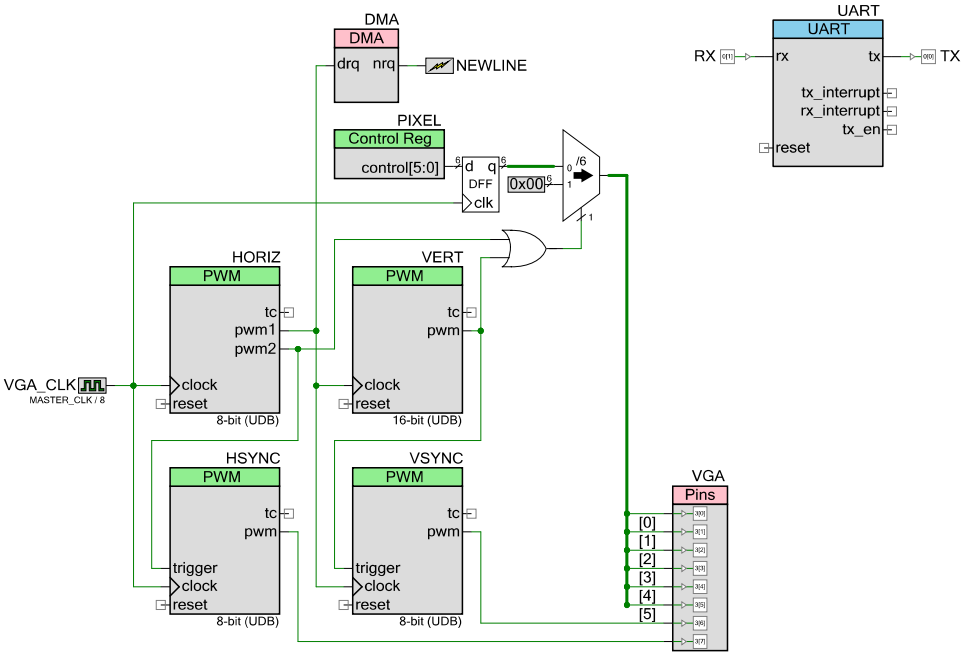

Figure 2: The arrangement of timers generating VGA output signals.