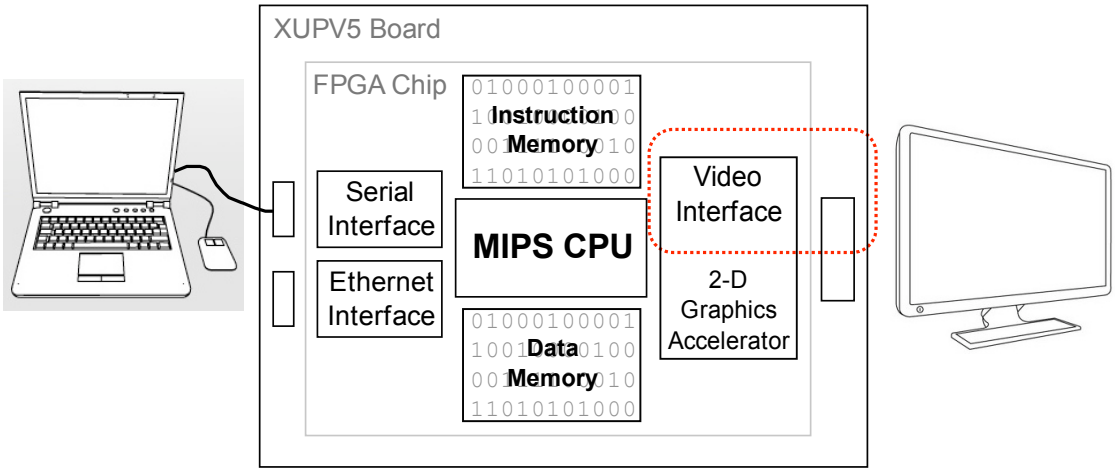
EECS150 - Digital Design

Lecture 15 - Project Description, Part 4

March 8, 2010

John Wawrzynek

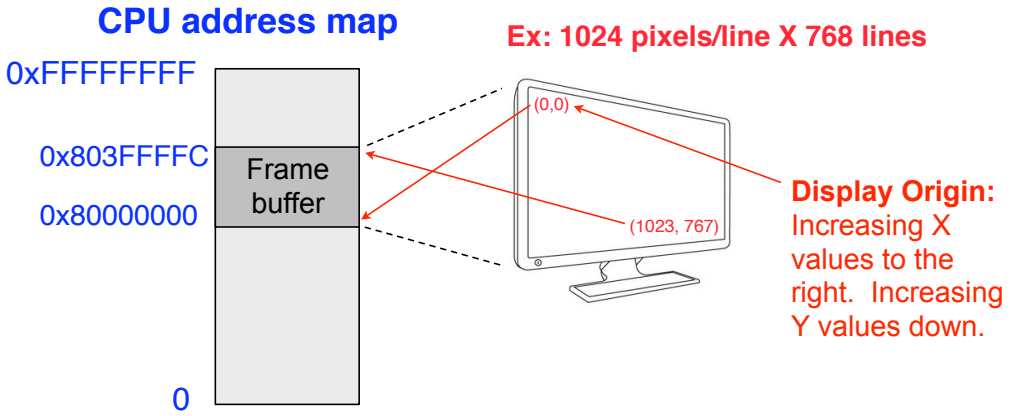
MIPS150 Video Subsystem



- Gives software ability to display information on screen.
- Equivalent to standard graphics cards:
  - Processor can directly write the display bit map
  - 2D Graphics acceleration

"Framebuffer" HW/SW Interface

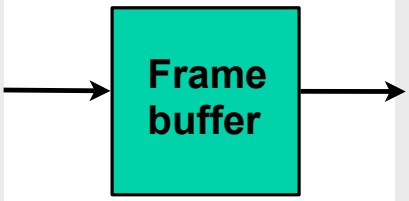
- A range of memory addresses correspond to the display.
- CPU writes (using sw instruction) pixel values to change display.
- No synchronization required. Independent process reads pixels from memory and sends them to the display interface at the required rate.



Framebuffer Implementation

- Framebuffer is a simple dual-ported memory. Two independent processes access framebuffer:

CPU writes pixel locations. Could be in random order, e.g. drawing an object, or sequentially, e.g. clearing the screen.



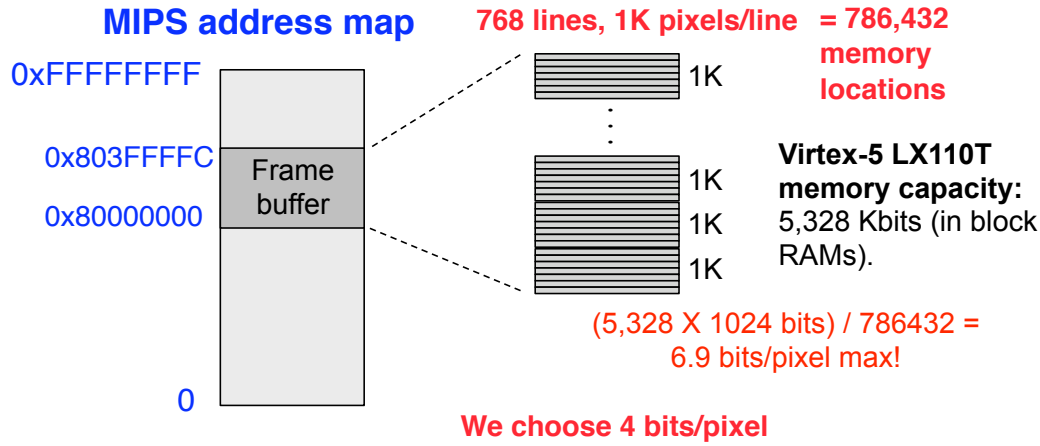
Video Interface continuously reads pixel locations in scan-line order and sends to physical display.

- How big is this memory and how do we implement it?

1024 x 768 pixels/frame x 24 bits/pixel

Framebuffer Details last year

- One pixel value per memory location.



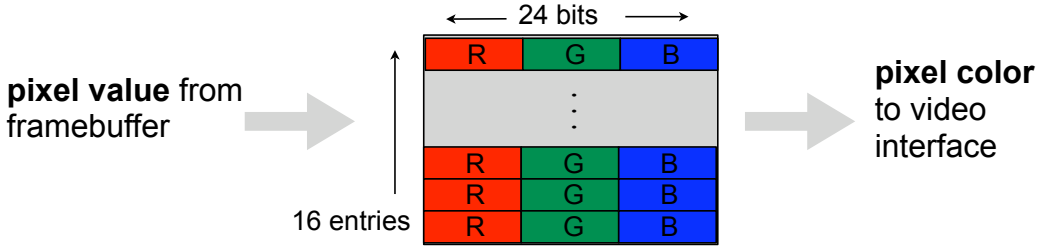
- Note, that with only 4 bits/pixel, we could assign more than one pixel per memory location. Ruled out by us, as it complicated software.

Color Map

4 bits per pixel, allows software to assign each screen location, one of 16 different colors.

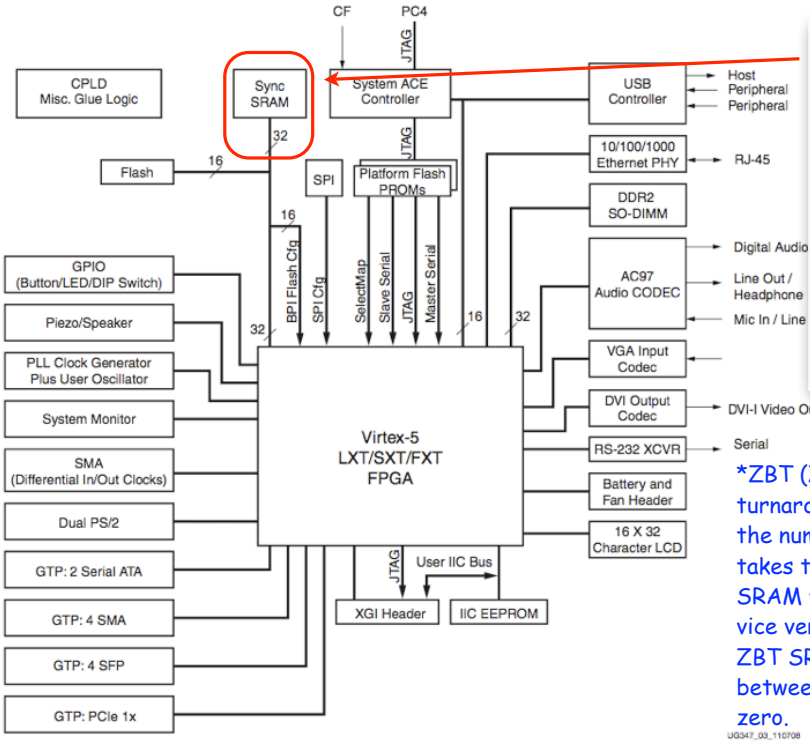
However, physical display interface uses 8 bits / pixel-color. Therefore entire pallet is 2^{24} colors.

Color Map converts 4 bit pixel values to 24 bit colors.



Color map is memory mapped to CPU address space, so software can set the color table. Addresses: **0x8040_0000 0x8040_003C**, one 24-bit entry per memory address.

XUP Board External SRAM



"ZBT" synchronous SRAM, 9 Mb on 32-bit data bus, with four "parity" bits
256K x 36 bits
(located under the removable LCD)

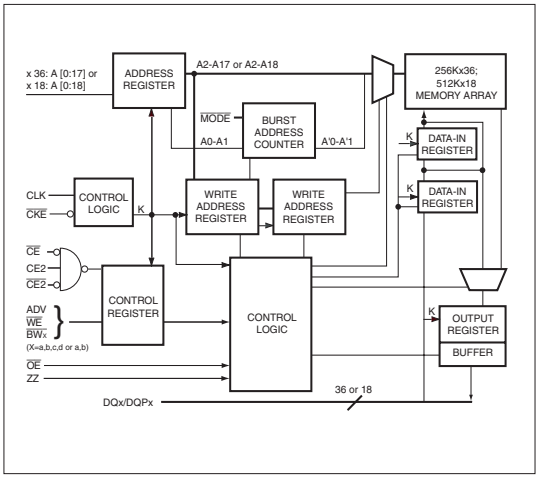
*ZBT (ZBT stands for zero bus turnaround) — the turnaround is the number of clock cycles it takes to change access to the SRAM from write to read and vice versa. The turnaround for ZBT SRAMs or the latency between read and write cycle is zero.

IS61NLP25636A/IS61NVP25636A
IS61NLP51218A/IS61NVP51218A

256K x 36 and 512K x 18
9Mb, PIPELINE 'NO WAIT' STATE BUS SRAM

MARCH 2008

BLOCK DIAGRAM



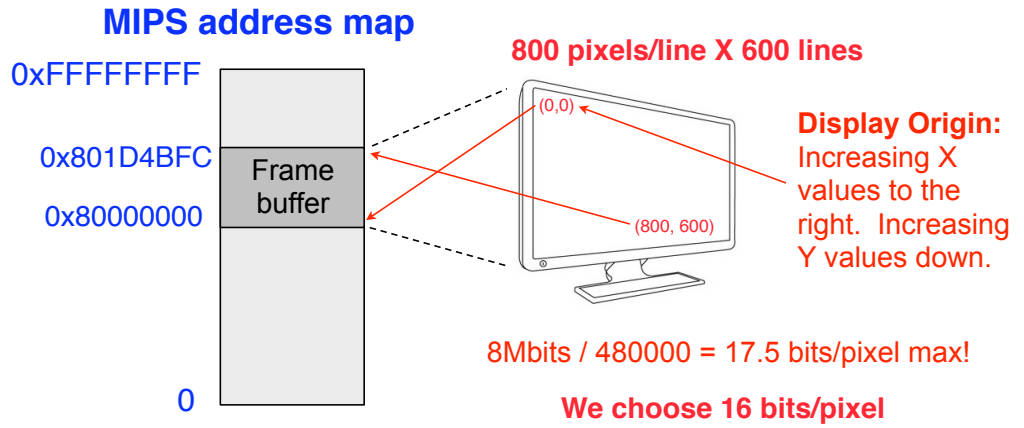
PIN DESCRIPTIONS

A0, A1	Synchronous Address Inputs. These pins must tied to the two LSBs of the address bus.
A	Synchronous Address Inputs
CLK	Synchronous Clock
ADV	Synchronous Burst Address Advance
$\overline{BW}$a-$\overline{BW}$d	Synchronous Byte Write Enable
$\overline{WE}$	Write Enable
$\overline{CKE}$	Clock Enable
Vss	Ground for Core
NC	Not Connected
$\overline{CE}$, CE2, $\overline{CE2}$	Synchronous Chip Enable
$\overline{OE}$	Output Enable
DQa-DQd	Synchronous Data Input/Output
DQPa-DQPd	Parity Data I/O
MODE	Burst Sequence Selection
V_{DD}	+3.3V/2.5V Power Supply
V_{SS}	Ground for output Buffer
V_{DDQ}	Isolated Output Buffer Supply: +3.3V/2.5V
ZZ	Snooze Enable

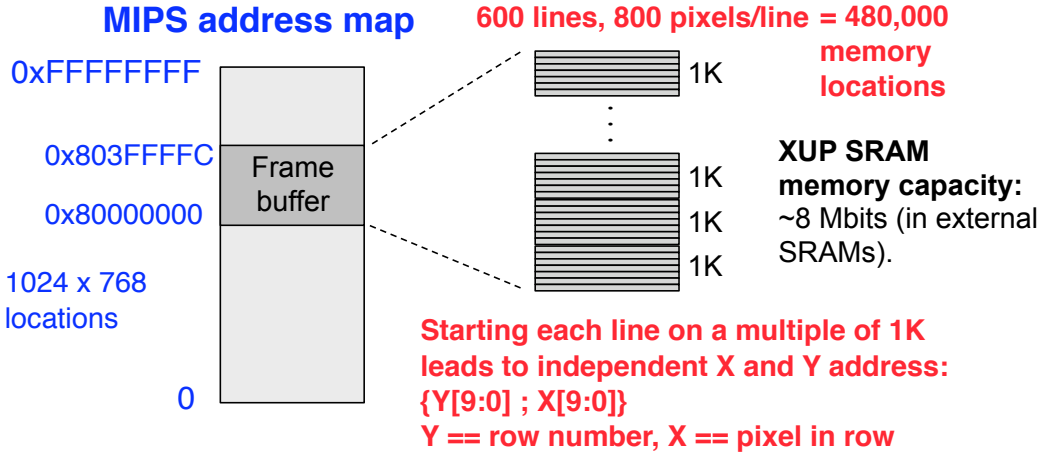
What frame buffer configuration is possible?

Memory Mapped Framebuffer

- A range of memory addresses correspond to the display.
- CPU writes (using sw instruction) pixel values to change display.
- No handshaking required. Independent process reads pixels from memory and sends them to the display interface at the required rate.

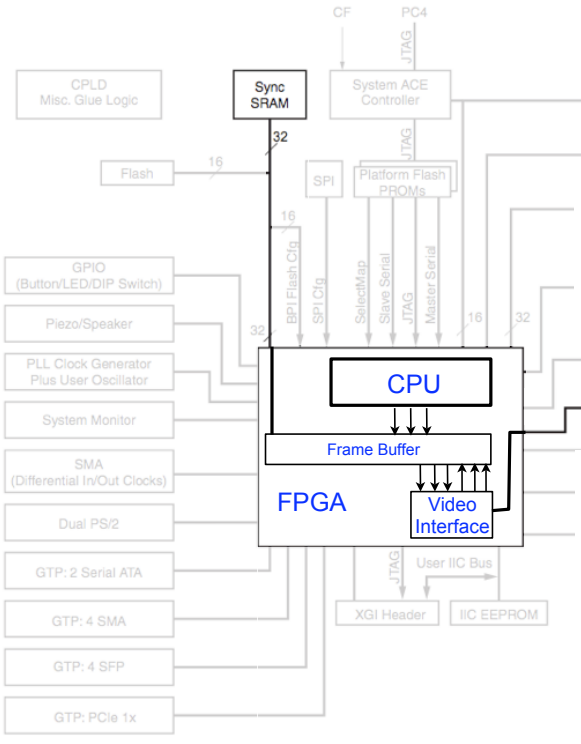


Framebuffer Details



- Note, that we assign only one 16 bit pixel per memory location.
- Two pixel address map to one address in the SRAM (it is 32bits wide).
- Only part of the mapped memory range occupied with physical memory.

Video Interface



Frame Buffer: provides a memory mapped programming interface to video display.

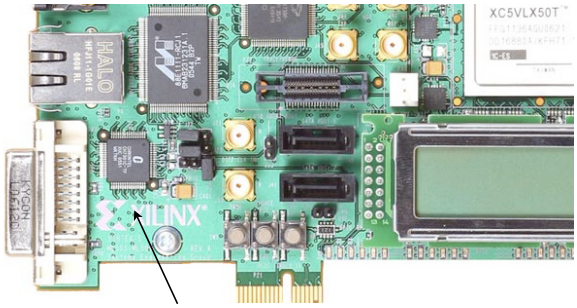
You do!

Video Interface Block: accepts pixel values from FB, streams pixels values and control signals to physical device.

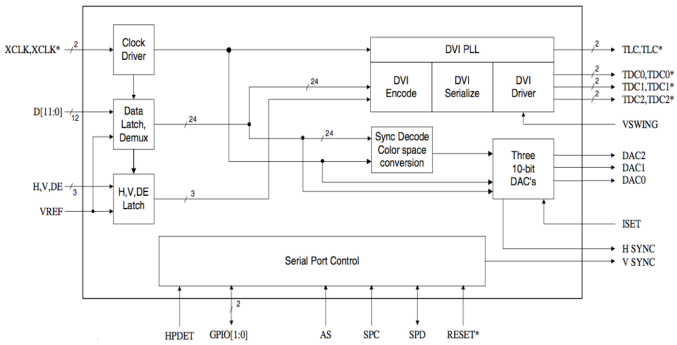
We do!

Physical Video Interface

DVI connector: accommodates analog and digital formats



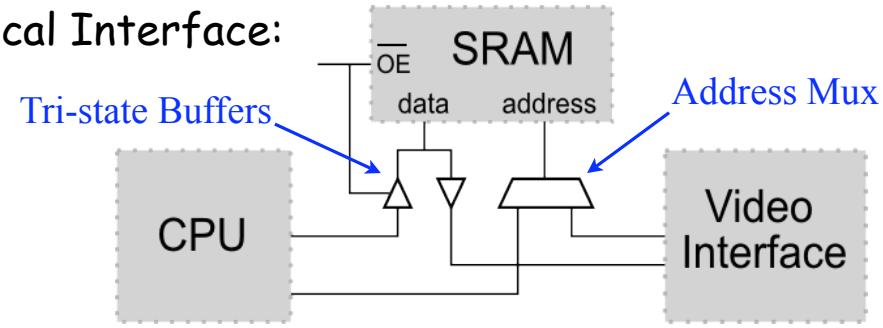
DVI Transmitter Chip, Chrontel 7301C.

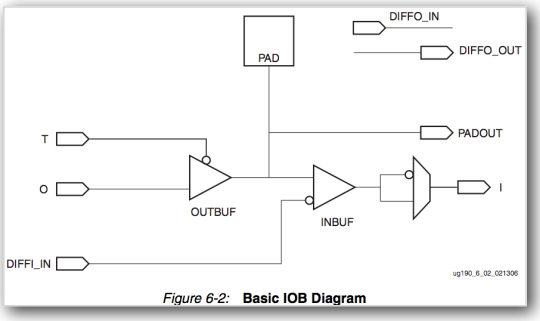


Implements standard signaling voltage levels for video monitors. Digital to analog conversion for analog display formats.

Video Interface Details

- Physical Interface:

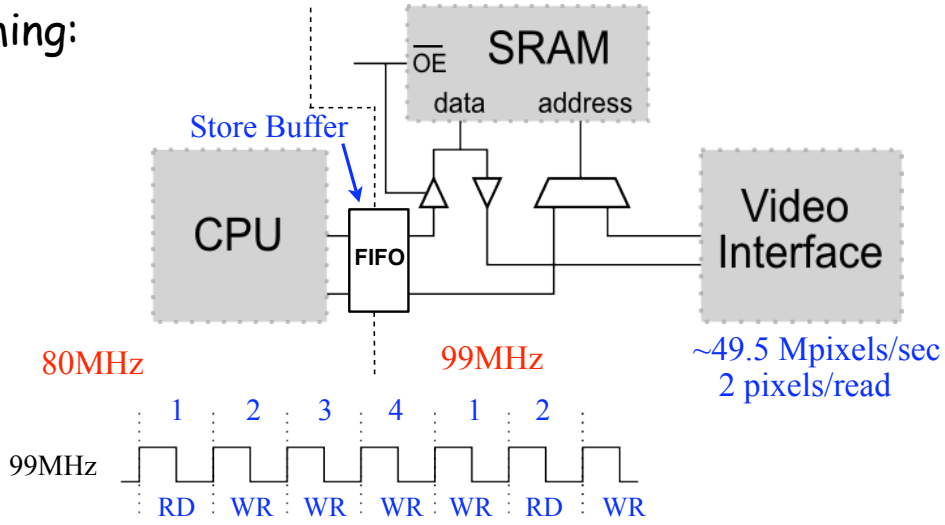




Xilinx I/O buffer

Video Interface Details

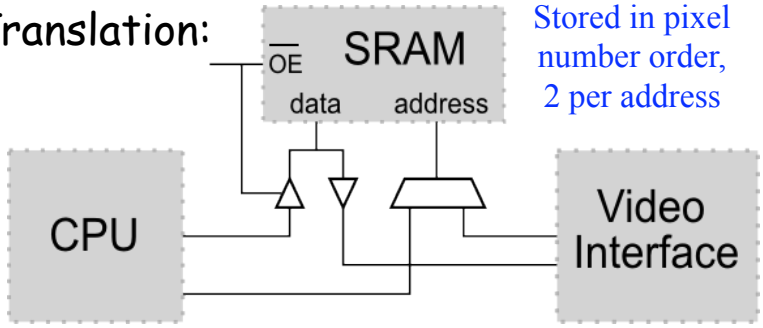
- Timing:



- All CPU frame buffer writes go through FIFO (and crosses clock domain boundary).
- Store Buffer writes to SRAM 3/4 SRAM cycles.
- Can Store Buffer fill up? What if CPU runs at lower clock rate?

Video Interface Details

- Address Translation:



Stored in pixel number order, 2 per address

Random writes using frame buffer addresses: Y[9:0], X[9:0]

Reads in pixel number order: 0 ... 480000

CPU writes need translation to convert from 20-bit frame buffer address to 19-bit SRAM address:

PN = X + 800*Y

How to do this on FPGA? 800 = 0x320 = 512 + 256 + 32

How do we write a single pixel? Use SRAM "Byte Write Enables"