CSEE 4840 Embedded System Design: NES FPGA Emulator

Jeff Jaquith, Minghao Li, and Zach Schuermann

Table of Contents

Block Diagrams and Communication Protocols:

On a high level, the CPU reads from the program ROM, which contains essential game information, and constructs the basic logics of the game. It then informs the PPU of how to specifically produce a pixel by pixel output on the screen.

- Program ROM: contains essential game data (sound, color, logic and etc.)
- PPU VRAM: high level graphics data
- Sprite RAM: max 64 sprites to be displayed in any given frame (and only up to 8 per scan line)
- Pattern ROM: patterns for game graphics

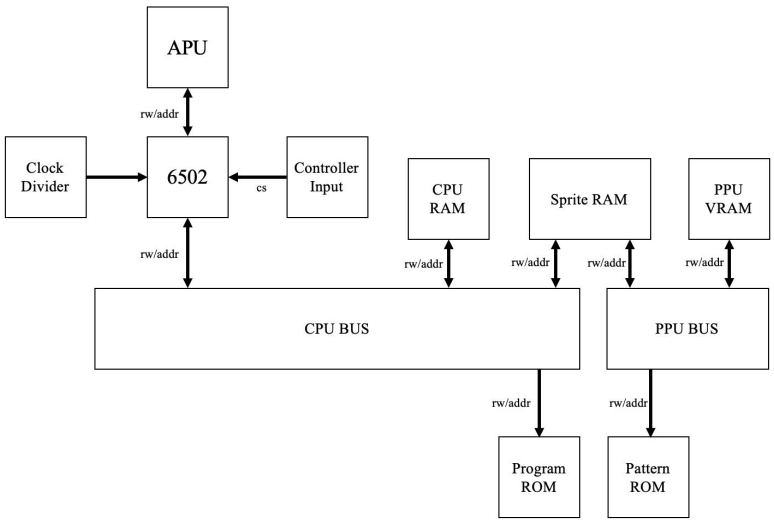

Figure 1. System Block Diagram

- The CPU is integrated on an ASIC (labelled RP2A03), that also integrates the Audio Processor, a DMA Unit, a clock divider and a few additional pins related to controller input.
- The 16-bit address bus is used for the address of a requested location. It is directly generated by the CPU. The CPU is all together able to address up to 64KB of memory or memory-mapped peripherals.
- The 8-bit control bus informs the connected components of whether the request is read or write.
- The 8-bit bidirectional data bus is used to read or write a byte to the selected address. Due to its bidirectionality, each peripheral is able to write to it at different times as specified by the address bus.
- To access the read-only ROM, a MMC is used for bank switching.
- The I/O registers are used to communicate with other components of the system.

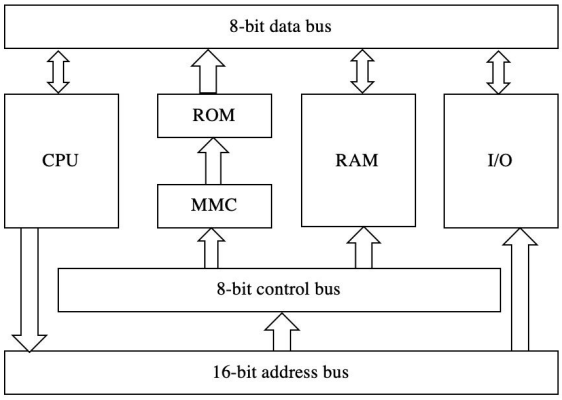

Figure 2. CPU Memory Communications

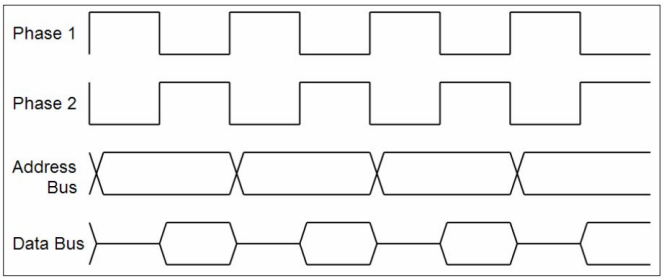

Figure 3. 6502 Bus Timing Diagram

- There are two modules to render the background and sprite respectively on a per pixel basis
- There are 8 PPU registers and they are accessed by the CPU's address and data lines.
    - Some are used to keep track of what is displayed on the screen
    - Some are used to hold information of sprites
    - Some are used for memory writes

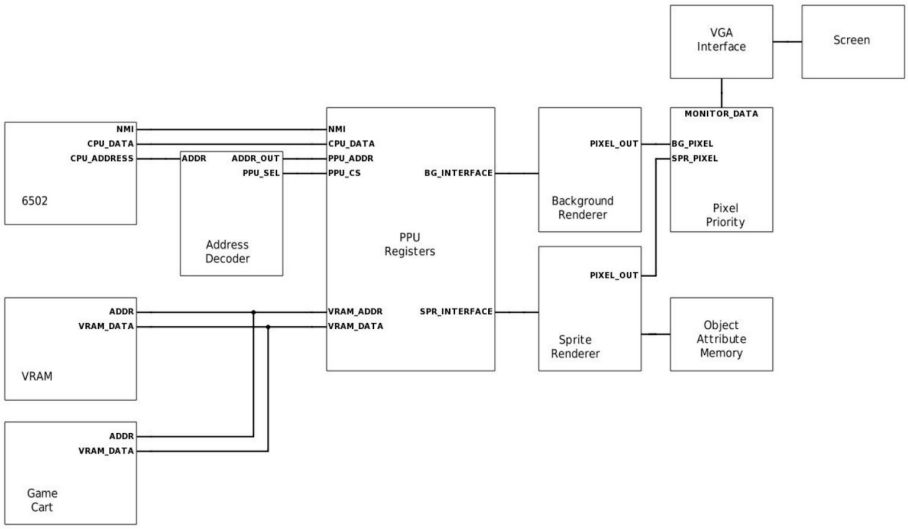

Figure 4. PPU Register and Other Relating Components

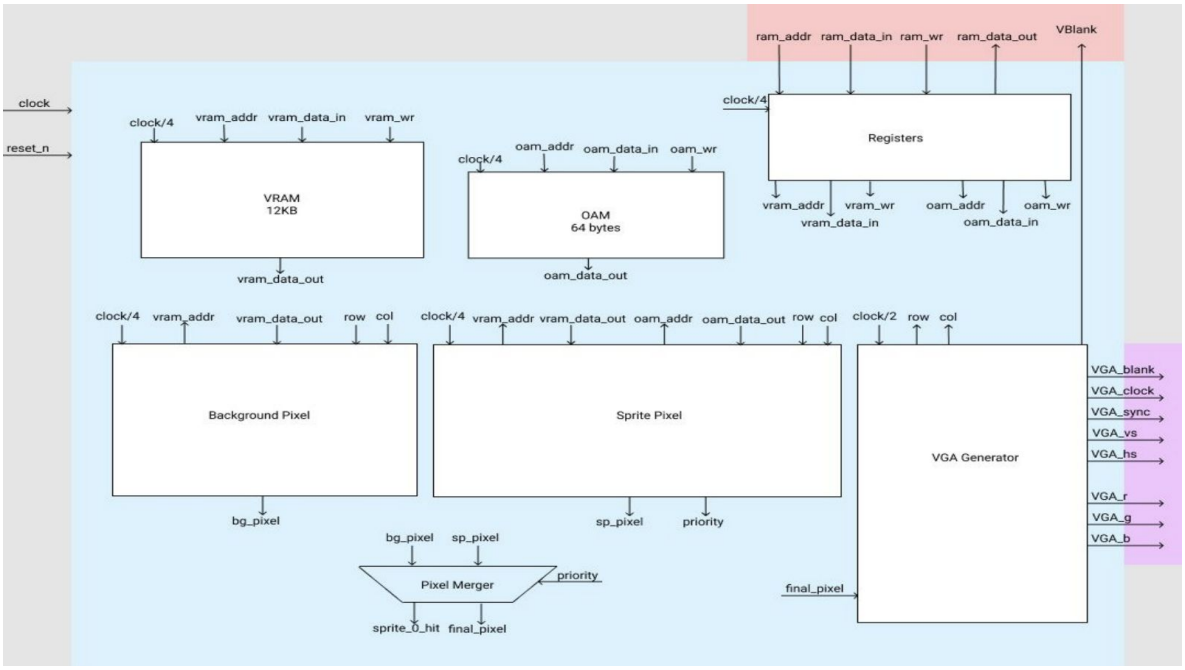

Figure 5. PPU IO Implementation Diagram

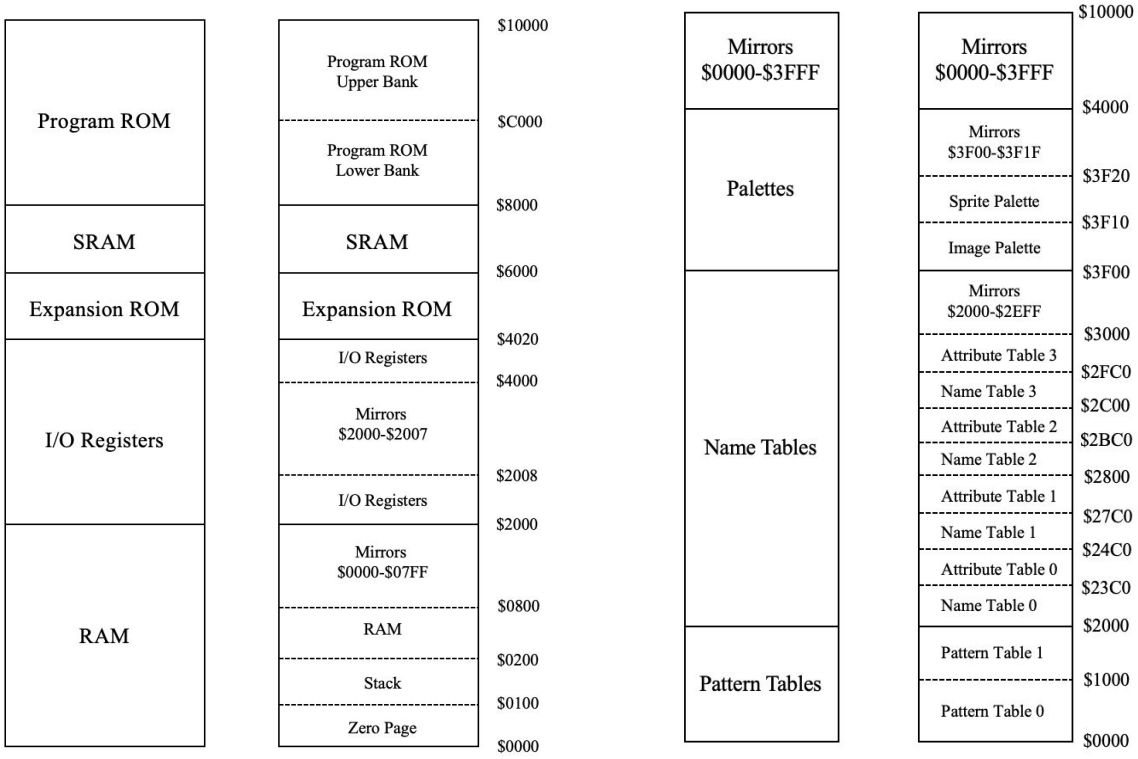

Figure 6. CPU (left) and PPU (right) Memory Map

Timing Figures:

- FPGA 50 and 25 MHz clocks. Per DE1-SoC Manual for 60 Hz 640 x 480 resolution needs 25MHz.
- 25 MHZ can also be used for ROM loader.
- CPU runs at 1.79 MHz - this is the PPU frequency / 3, or every third cycle
- PPU runs at 5.37 MHz. Generate this frequency using a phase lock loop (PLL) in Quartus.
    - Drop 50 MHz to 5.73MHz -- (50 * 189 / 220) / 8

PPU Rendering:

- PPU renders 262 scan lines per frame
    - 240 visible scan lines
    - 20 fetching data (vblank)
    - 2 dummy
- Only can write one pixel per PPU cycle
    - Takes 341 PPU cycles per scanline
      256 for rendering; remaining are used to fetch data from nametables, etc.
- For each frame:
    - -1 scanline: prefetch tile info for first two tiles
    - 0-239 scanline: render background and sprite
    - 240 scanline: idle
    - 241-260 scanline: vblank lines, CPU can access VRAM
- For each visible scanline:
    - 0 cycle: idle
    - 1-256 cycle: visible pixels
        - Output pixel based on VRAM
        - Prefetch next tiles
        - Sprite evaluation for next scanline
    - 257-340: prefetch tile data for next line’s first two tiles

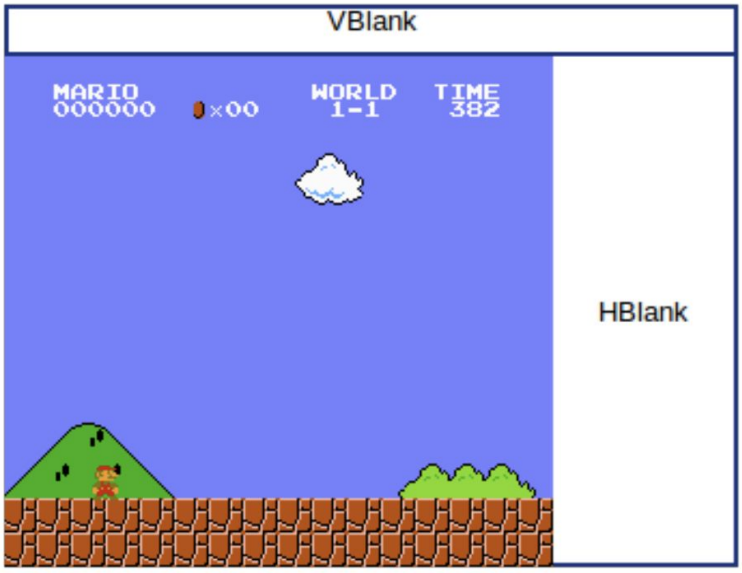

Figure 7. Blank Areas Used During CPU Cycles/Data Fetch

Arbitration of shared resources
CPU/PPU share memory, CPU updates mem (VRAM) during VBlank.
Can also use a dual port block ram. (DPBRAM) Allows PPU and CPU to access memory separately.

Hardware/Software Interface
The hardware/software interface exists between the FPGA implementation of the NES and the Linux host which will load ROM's (game cartridges) onto the board.

Resource Requirements:
The DE1-SoC has 64MB SDRAM, far greater than any combination of NES + ROM cartridges with additional RAM (no more than 1MB). Furthermore, the original 6502 had ~3,200 transistors, while our FPGA has 85,000. The 6502 core we plan to use occupied only 8% of the flops and 7% of the LUTs in the Xilinx xc3s500e FPGA.

Appendix:
Sprite DMA Unit: For updating to the internal PPU memory, there are memory mapped registers at $2003 and $2004. $2003 is used to set the internal address, $2004 writes a value with auto-increment.
Since most games would want to update all 256 bytes of sprite data in each frame, the 2A03 integrates a unit that can halt the CPU and copy one memory page to $2004 directly.

References:

https://www.quora.com/q/oefyspdckxhlovmr/How-NES-Graphics-Work-Pattern-tables
http://web.mit.edu/6.111/www/f2004/projects/dkm_report.pdf
http://nesdev.com/NESDoc.pdf
https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/ug_altpll.pdf
http://www.aholme.co.uk/6502/Main.htm