EE 4840: Embedded System Design

TupacMan: Design Document

March 28, 2007

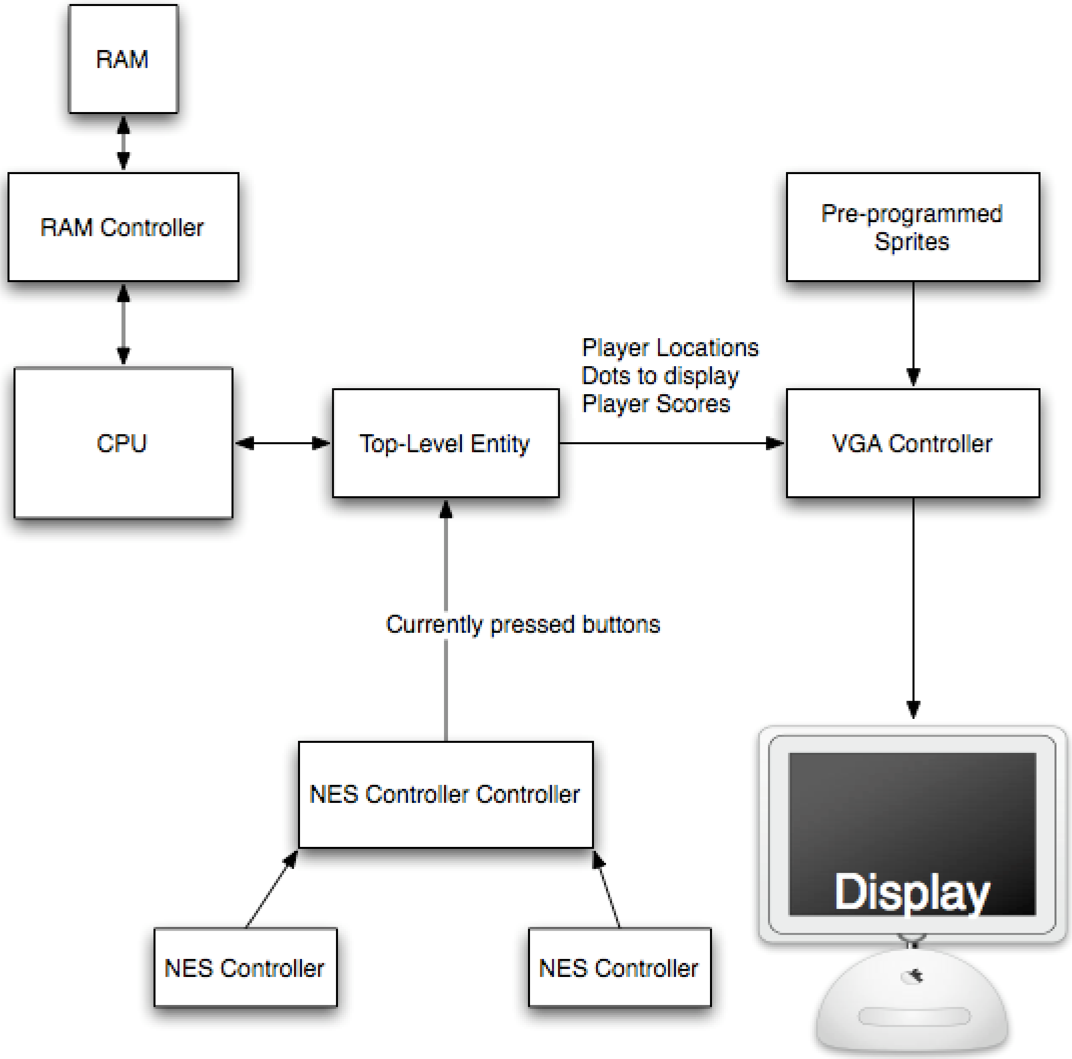
Varun Mehta (vkm2101)
Michael Pierorazio (mbp2103)
Jeffrey Cropsey (jdc2103)

INTRODUCTION

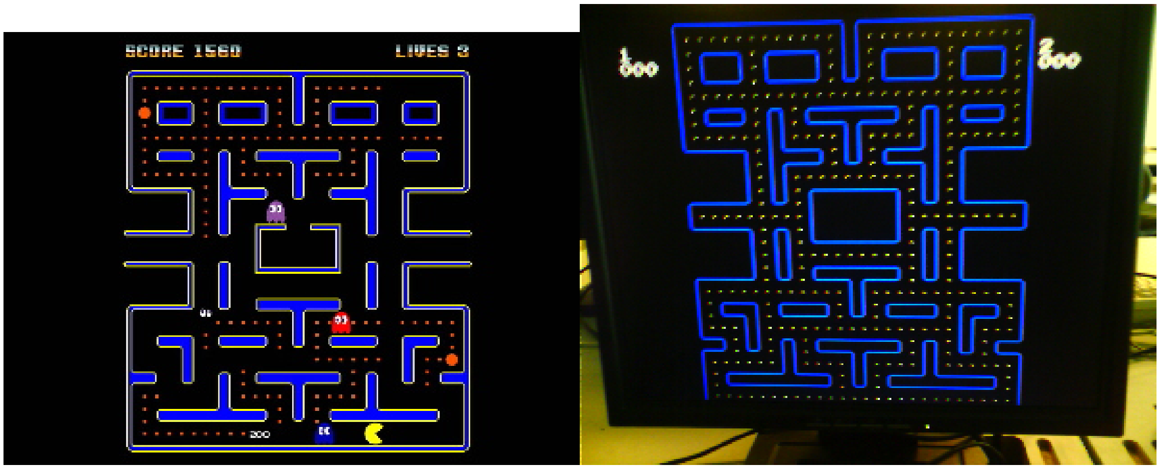
Our goal is to design an arcade game that consists of a mix of custom designed hardware and software. In hardware we will design (sometimes calling upon examples used in class or provided by the lab software) a VGA controller, a CPU, a RAM controller and an interface for the pre-existing NES controller. The game we are designing is a Pac-Man variant tentatively named “TupacMan”. The variation is that the game is designed for two players. This will be accomplished by having the two players race to eat the most pellets.

We plan to use the VGA output of the Altera board to present the game's graphics. Images will be initialized in hardware and sprite movement done in software. The software will also control game logic. Players will play the game using NES controllers.

ARCHITECTURE



GRAPHICS



Left: Pac-Man screenshot, Right: Development Screenshot of TupacMan

Currently we have vhdl code that defines and draws the game board, draws the players, pellets as well as numbers, which will be used to track the scores of each player. As you can see from the image on the left we used the same classic design as Pac-Man. There is a RAM mapped to the pellets on screen, so the software can communicate which pellets are to be drawn. Each line of the game board has its own 32-bit word, which corresponds to the pellets on that line, with bit 0 being the right-most pellet. All of the sprites for players, lines, numbers, curves and pellets are currently being stored in a RAM onboard the chip. The sprites for the players are also stored in hardware. Memory between software and the graphics component is mapped as follows:

Address |	**32-bit words**
0	Player 1’s y-position (16), Player 1’s x-position(16)
1	Player 2’s y-position (16), Player 2’s x-position(16)
2	Garbage(23), Player 1’s 3-digit score (4,4,4)
3	Garbage(23), Player 2’s 3-digit score(4,4,4)
4	Garbage(6), Pellet Data for line 0 (26)
…	
32	Garbage(6), Pellet Data for line 27 (26)

Currently all of these graphical components and their interface with software have been completed.

GAME LOGIC

The game commences after both players press ‘Start’. Each characters sprite will move in the direction of the sprite’s mouth until direction is changed by the player using the directional pad. Only the directional pad is used during game play. Each time a player moves over a pellet it will be ‘eaten’, his score will increase and the pellet sprite will be removed from the game board. To restart after completion of the game both players should press ‘Start’.

NES CONTROLLER INPUT

```
          +----> Power
          |
5 +---------+  7
  | x  x  o   \
  | o  o  o  o |
4 +------------+ 1
    |  |  |  |
    |  |  |  +-> Ground
    |  |  +----> Pulse
    |  +-------> Latch
    +----------> Data
```

Pin-Out of NES controller

The NES polls its controller every 60 Hz in order to determine the state of the eight buttons. It accomplishes this by sending a high signal to the ‘Latch’ pin; the controller then latches the state of the buttons. The ‘Latch’ signal is followed by eight high pulses of the ‘Pulse’ pin.

After the latch signal, ‘Data’ pin goes high for the duration of the eight high pulses of ‘Pulse’ before returning to low (if no buttons have been pressed). Each of the eight pulses relates to each of the eight buttons on the NES in this order: A, B, Select, Start, Up, Down, Left, Right. If one of these buttons was pressed when the ‘Latch’ signal was received then ‘Data’ will be low during that corresponding pulse of ‘Pulse’.

Input from the controller will be interpreted by a state machine and will be available to the CPU as signals.

RAM

It seems that everything all of the images and game states will be able to be stored in the main chip on the Altera board either in our custom RAM or as signals.