PAC-MAN HHZ

CSEE 4840 Embedded System Design

Yuedong Huang
Electrical Engineering Department
yh2566@columbia.edu

Youwei Hou
Electrical Engineering Department
yh2551@columbia.edu

Yizhong Zhang
Electrical Engineering Department
yz2482@columbia.edu

MARCH 26, 2013

COLUMBIA UNIVERSITY IN THE CITY OF NEW YORK

1. Overview

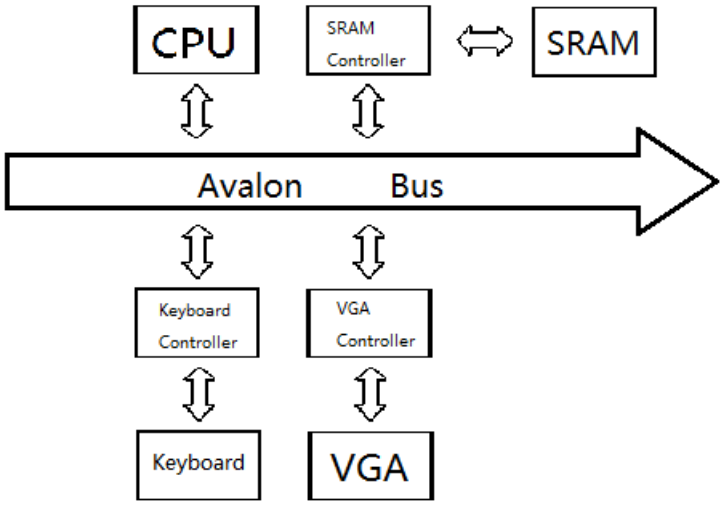
The theme of our project is to create a Pac-Man-Like video game. The game is like the classical Pac-Man game and it should be running on a FPGA board and displayed on a VGA screen.

The player will control Pac-Man through keyboard's arrow keys to direct Pac-Man through a maze, eating Pac-dots. When all Pac-dots are eaten, Pac-Man is taken to the next stage. Ghosts roam the maze, trying to catch Pac-Man. If a ghost touches Pac-Man, a life is lost. When all lives have been lost, the game ends. Some extra feature may be added, e.g. Pac-pellets the enables Pac-Man to temporarily have the ability to eat ghosts.

2. Architecture

Based on the experience of lab3, in this project, we will also build a nios system component which contains several sub-component, such as VGA, keyboard, CPU, SRAM. The peripherals connected with this system are keyboard and VGA.

Hardware Architecture



3. Hardware Implementation

Hardware is responsible for building maze, drawing ghosts and Pac-man. The keyboard is used to control the movement of Pac-man. In order to accomplish our goal, the following component is needed:
Keyboard controller;
VGA controller.

Keyboard controller:
The keyboard controller is used to receive and store the code number of keyboard, we will use up, down, left and right to control the movement of host.

There will be a 1*16 RAM in the controller to store the code number which is used for CPU to read.

VGA controller:
The VGA controller is used to display host, sprites and maze on VGA. We will use two arrays for the host, one for open mouth and the other for close mouth. There should also have several arrays for sprites to simulate the movement. On the other hand, there will be many arrays for the background of the game, including score, numbers of lives, maze etc.
Thus, we need a RAM in the controller to store the coordinates of the sprites and host. For we have four sprites and one host, the RAM should be 5*16.

4. Critical Path

We think the critical path is determined by the sequence of operation as follows: first I press a keyboard to change the direction of Pac-man. Then the keyboard controller store the number code of the button for the reading operation of CPU. CPU read the key word and determine which direction should turn and put it on the bus and write it in the RAM of VGA controller. At last, the VGA controller tells VGA which array and where the array should be displayed next. This is the longest path in this game, so if the player enter the button too fast, then the VGA may fail to display the right array.

5. Software Implementation

With the properly set up hardware, the software can access to each component through the Avalon bus. As for Pac-man, it will continue moving forward until an arrow key on the keyboard has been pressed and Pac-man is allowed to turn (e.g. at a cross). The software will interpret the key code send from keyboard controller and adjust Pac-man's moving direction accordingly. Four ghosts will be released from the central chamber gradually and they will move toward Pac-man. The software will calculate their moving direction and make sure they don't just roam randomly. The ghosts' speed will increase as the game gets harder. This is also controller by software.

Also, some extra feature like Pac-pellets that enables Pac-man to temporarily eat ghosts can be realized by software. The software will tell when the Pac-man touches one of the pellets and set up a counter. As long as the pellet remains active, the software will tell the hardware to paint the ghost in a grey color indicating they are vulnerable.