Snappers

Project Design

March 2012

Yuhan Dai
Electrical Engineering Department
yd2233@columbia.edu

Lianyi Ding
Electrical Engineering Department
ld2504@columbia.edu

Dian Wang
Electrical Engineering Department
dw2504@columbia.edu

Chi Zhang
Electrical Engineering Department
cz2244@columbia.edu

Introduction

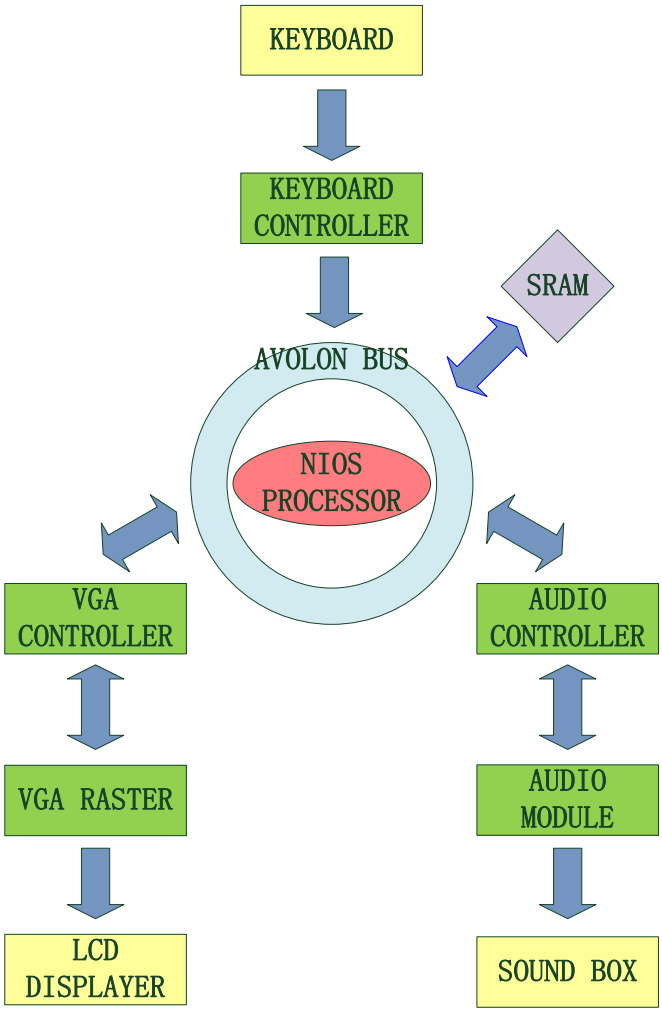
Snappers is a popular puzzle game on iOS platform developed by Emerging Banking LLC in 2011. The goal of this game is to eliminate as many snappers (the red and green objects in the picture) as possible. The player needs to choose a snapper and turn it into four bullets in horizontal and vertical directions to shoot other snappers. Snappers of different colors can resist different times of shooting before it disappears. Once a snapper is shot, its color will change correspondingly. This game requires thinking, determination and trial-and-error. Our goal is to implement this game on the Altera Cyclone II FPGA board. We will use the FPGA board to display the graphics and the C program to control the movement and variation of the objects shown on the screen. We will add some sound effect to the game and we will implement some functional button to enable pause/resume and forward/backward. In addition, the players will learn how to play by following the tutorial instruction.

Design Architecture

In our project, there are mainly three parts of work. The block diagram below shows the fundamental architecture of Snappers.

First, we use the keyboard as the game controller. The keyboard controller should receive the input from the keyboard and send data into the Avalon Bus. Secondly, through the VGA control, we can edit the movement or variation of the figures which implemented by VGA raster. It will finally display on the LCD screen. Thirdly, as the procedure of VGA, we use the audio control to manage different sounds and then produce them through the sound box.

We can see from the diagram: Keyboard, LCD displayer and sound box are the peripherals of this design; all the three controllers pass the signal back and forward to CPU—NIOS processor through the Avalon Bus; SRAM can store all the data in the bus.



VGA blocks

We will use 5X7 squares as the game screen and each of them is formed by 60X60 pixels. The other part of the screen will be put on some functional buttons. The VGA part should have following functions:

- Loading the scenario setup (background) from the bus sequentially and put images at proper positions on the screen.
- Adjust the shape or positions of snappers and bullets when the relevant commands received from the bus.

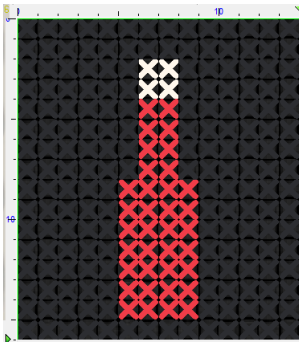
We will use a signal called "color" which is a 3-bit vector so that it can represent 8 different colors in this project. The value of this signal will indicates different colors and the vhdl code is as follows:

```
if color = '000' then        --white
VGA_R <= "1111111111";
VGA_G <= "1111111111";
VGA_B <= "1111111111";
if color = '001' then        --black
VGA_R <= "0000000000";
VGA_G <= "0000000000";
VGA_B <= "0000000000";
if color = '010' then        --green
VGA_R <= "0000000000";
VGA_G <= "1111111111";
VGA_B <= "0000000000";
if color = '011' then        --red
VGA_R <= "1111111111";
VGA_G <= "0000000000";
VGA_B <= "0000000000";
if color = '100' then        --blue
VGA_R <= "0000000000";
VGA_G <= "0000000000";
VGA_B <= "1111111111";
if color = '101' then        --yellow
VGA_R <= "1111111111";
VGA_G <= "1111111111";
VGA_B <= "0000000000";
if color = '110' then        --purple
VGA_R <= "1111111111";
VGA_G <= "0000000000";
VGA_B <= "1111111111";
if color = '111' then        --light blue
VGA_R <= "1111111111";
VGA_G <= "0000000000";
```

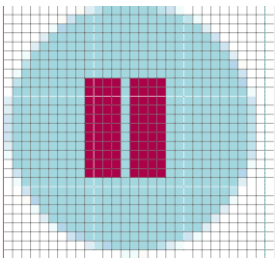
VGA_B <= "1111111111";

The icon of the “snapper” will be generated in 50X50 pixels:

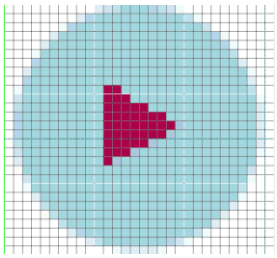
Bullet (16X14 pixels):



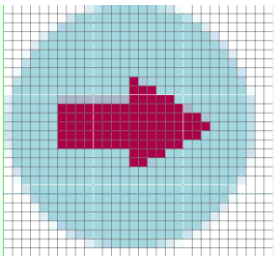
Function buttons: 30X30 pixels:



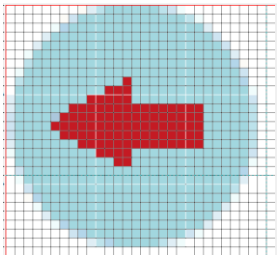
pause icon



start icon



next stage icon



back stage icon

Keyboard

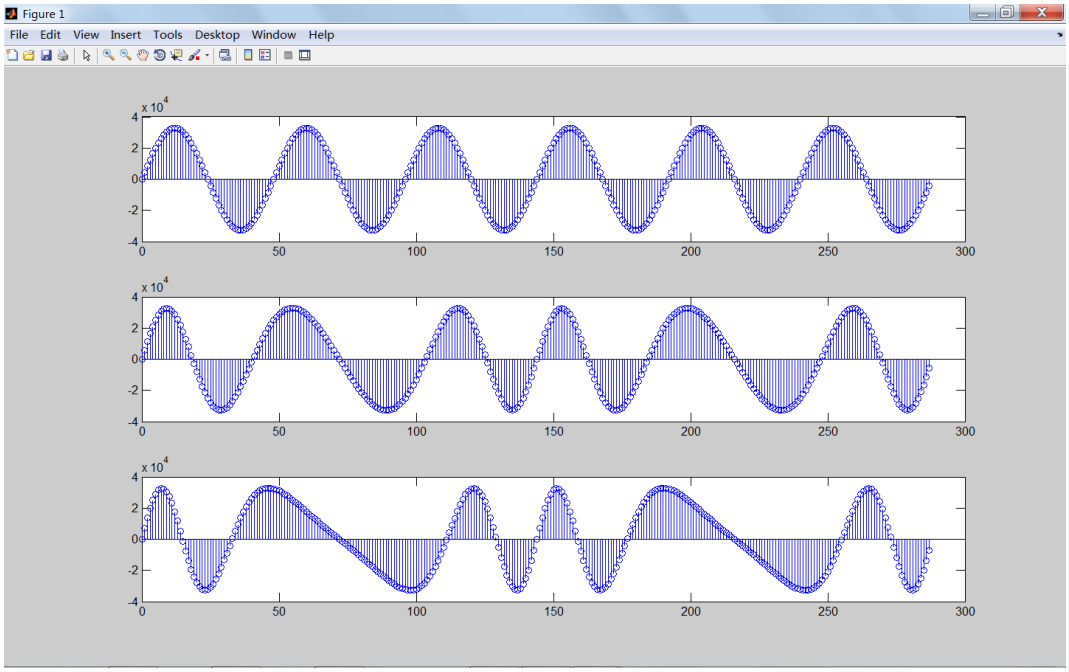
In our project PS2 keyboard is used to control both the video and audio part. We use VGA and audio function in our processor system to read the data from keyboard and then send the data to Avalon bus. The Nios2 system will connect the keyboard controller to the VGA and audio controller through the Avalon bus. The keyboard interface will be implemented in C program.

Audio Block

In our game option, we can choose to turn on music or not. When music is on, when the player clicked to explode one snapper, it will sound like bomb explosion and it will play a piece of melody after victory. We use half of the 50MHz clock to sample and calculate the sampled value of the audio wave by MATLAB. Finally, we store different audio waveforms in the hardware, and connect the PS2 keyboard to FPGA board and use C program to select the amplitude of the harmonic sinoidal wave in order to change the timbre of the sound.

The richness of a sound is described in terms of a sum of distinct frequencies. We maintain the fundamental frequency to keep the pitch and add a multiple of harmonics.

Three quasi-sinoidal waveform for the same frequency but different timbre including artificial harmonics:



Finally, we can use either a speaker or an earphone to receive the sound.

Reference

http://en.wikipedia.org/wiki/Timbre
http://itunes.apple.com/us/app/snappers/id479852117?mt=8