Smasher Video Game Design

CSEE 4840 Spring 2012 Project Design

Prepared for

Prof. Stephen Edwards

Prepared By

Jian Liu, jl3784

Tong Zhang, tz2176,

Chaoying Kang, ck2566

Yunfan Dong, yd2238

Mar 20th,2012

1. Abstract

Our team proposes to implement a video game named Smasher. This is a game in which the player controls boards to bounce the ball and break as many blocks as he can. There are four boards in the four sides of the screen. If the ball fails to bounce and goes out of the screen, the game is over. We plan to set several missions to complete the game.

The picture above is just to show the basic idea of our design. We will do some upgrade and the final result will be different. Detailed description is as follows.

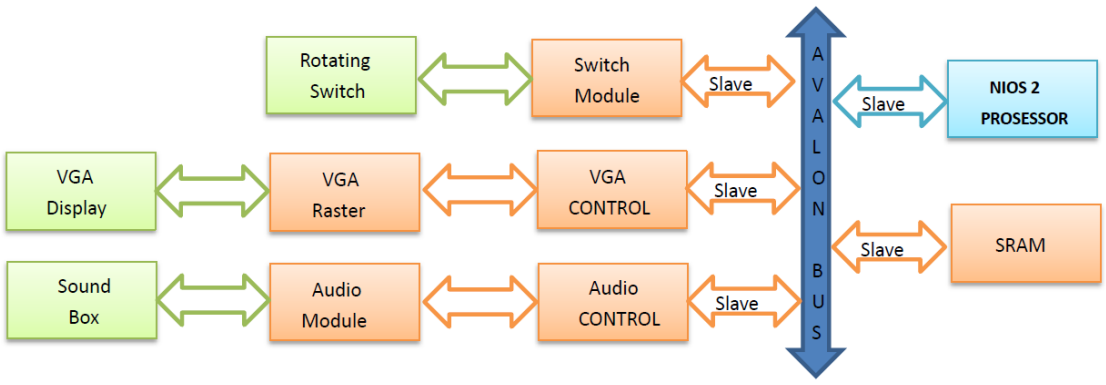
2.Design Feature

- Players control the board using rotating switch.
- Game interface is shown on the VGA
- Two pairs of boards located on four sides of the screen. Each of them can bounce the ball.
- The ball bounces back when it hits the board according to the reflection law.break all the blocks.

- We plan to place different kinds of blocks in the game.
- There can be a “worm-hole” in some missions, which is an invisible pipe. If the ball appears in the entrance of the hole, it will immediately be transported to the exit of the hole. The motion status will remain the same.
- When the ball breaks the block, gifts may randomly appear in screen and gradually fall out of screen. If the board touches the gift, It can be up-grated or down-graded.
- Scores will be calculated, and the high score will be recorded.
- If the ball hits the block, it also bounces according to the reflection law, and the block breaks. The goal of a mission is to

3.Block Diagram

The following is the design block of our project. It is composed of CPU, VGA Control, Audio Control, SRAM, Switch Module, VGA Raster, Audio Module. As shown in the diagram, the green blocks indicate the real world hardware while orange blocks represent the Avalonslaves and the blue block stand for the CPU. All these blocks will be bind to the Avalon bus with SOPC builder in the Quartus. The VGA raster is the block which actually communicates with the LCD displayer through VGA port and displays the gram graphics. In the VGA block, the basic graphic patterns will be pre-set and the software will send the coordinates of positions to the VGA raster hence realize the control of the game graphics. The audio controller and audio module works in a similar way to the VGA controller and VGA raster. Software will tell the audio block when to play which kind of sound. The details of the implementation will be provided in the upcoming sections.



4.Software Implementation

Software is mainly responsible for board controlling, the tracking of the ball, event processing like break the brick and get bonus. The track of the ball is quite similar to that of the lab3.We try to use the C code to control the position of the ball. By the way, the ball’s moving position and speed can be changed depends on the board’s hitting angle. This function will also be realized by the C code.

Another big part of the software is the communication between the player’s control

and the board's movement. The software will get the data from the input hardware and convert it to the position of the board. Then the program will sent the position of the board to the VGA raster the FPGA will update the position of the board in the screen.

5. Hardware Implementation

Hardware is responsible for the cartoon display and audio synthesize. The sprite arrays will be store in the FPGA board's memory and will be displayed on the screen, following the software's program.

5.1 Rotary Switch

Another big part of the hardware is the input hardware: rotary switch. A rotary switch is a switch operated by rotation. When we rotate the switch of this component, the rotary switch will give us a chain of binary number. We use these data to control the board's movement.

5.2 Audio Generation

The audio generation is also a big part. We try to use what we learn in Lab3 to generate some sound effects, such as the brick's breaking, game's background music and bonus' sound. Each sound will be triggered and terminated by software.

5.3 VGA Display

In our design, the colors are designated as follows to obtain better graphic performance:

Black: "00000"
Dark Grey: "00001"
Gray: "00010"
White: "00011"

Dark Red: "00100"
Red: "00101"
Light Red: "00110"

Dark Orange: "01000"
Orange: "01001"
Light Orange: "01010"

Dark Yellow: "01100"
Yellow: "01101"
Light Yellow: "01110"

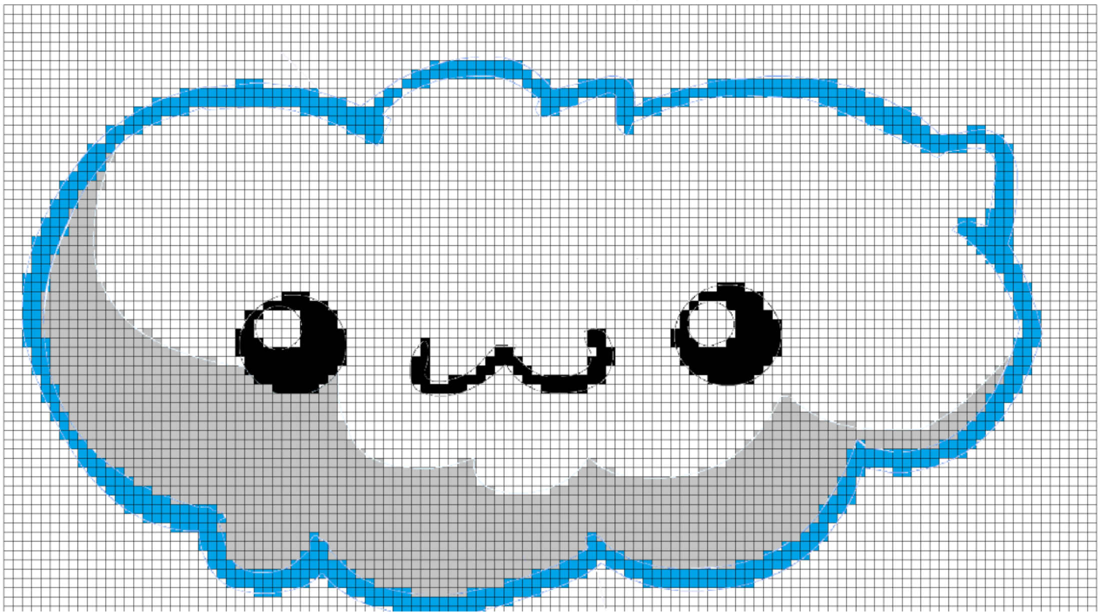
Dark Green: "10000"

Green: “10001”
Light Green: “10010”

Dark Blue: “10100”
Blue: “10101”
Light Blue: “10110”

Dark Purple: “11000”
Purple: “11001”
Light Purple: “11010”

The sprite of the board is given:



matrix for the first line:=
(“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”)
For a typical line #16:=
(“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“10101”,
“10101”,“10101”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,

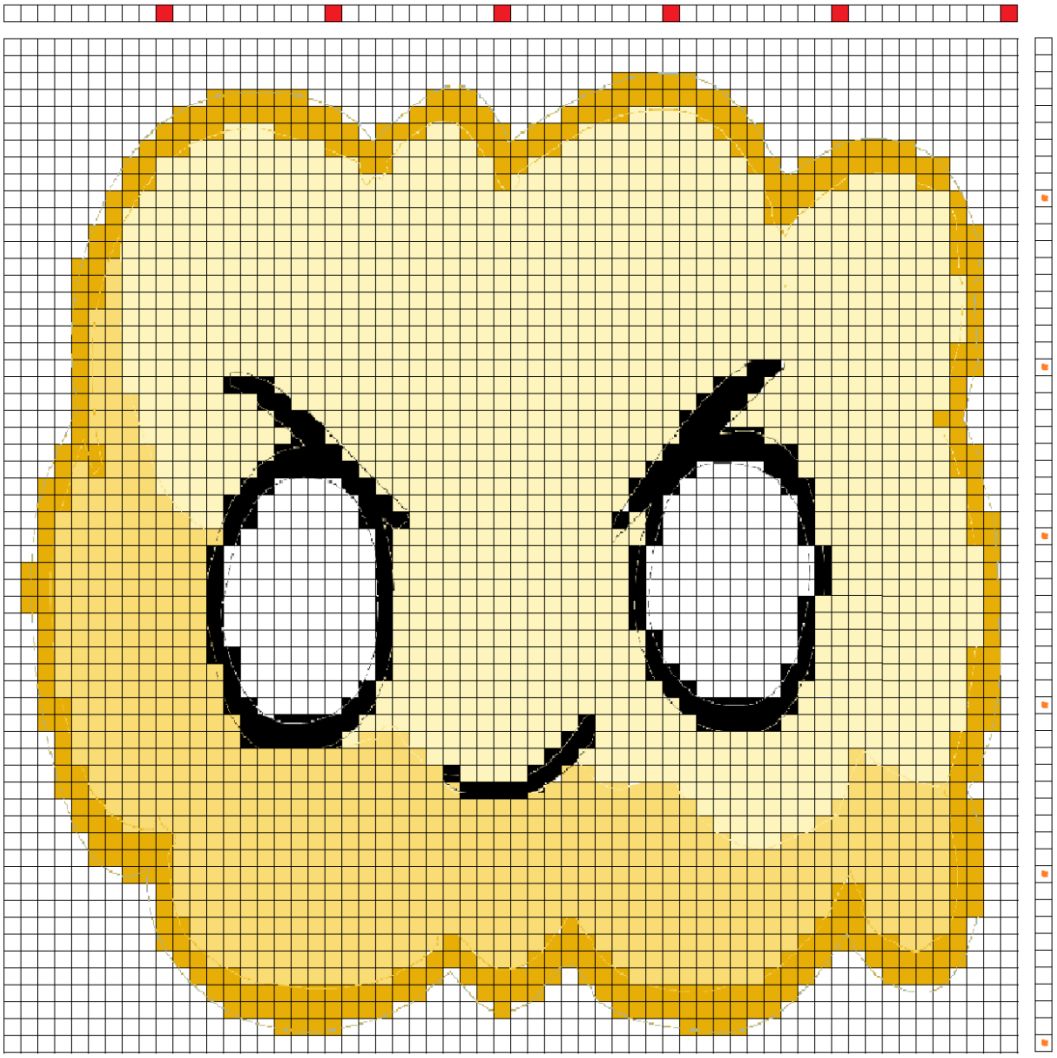
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,
“00011”,“10101”, “10101”,“10101”,“10101”,“10101”,“00011”,“00011”,“10101”,“10101”,
“10101”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”)

The sprite of the brick is given:

The matrix for a typical line #10 is given:
(“00011”,“00011”, “00011”,“00011”,“00011”,“00011”,“10100”,“10100”,“10110”,“10110”,
“10110”,“10110”, “10110”,“10110”,“10110”,“10110”, “10110”,“10110”,“10110”,“10110”,
“10110”,“10110”, “10110”,“10110”,“10110”,“10110”, “10110”,“10110”,“10110”,“10110”,
“10110”,“10110”, “10110”,“10110”,“10110”,“10110”, “10110”,“10110”,“10110”,“10110”,

“10110”,“10110”, “10110”,“10110”,“10110”,“10110”, “10110”,“10110”,“10110”,“10110”,
“10110”,“10110”,“10110”,“10110”, “10110”,“10110”,“10100”,“00011”,“00011”,“00011”,)

The sprite of one other brick is given:



The matrix for a typical line #51 is given:
(“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“00011”,“01100”,
“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,
“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,
“01110”,“01110”,”01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,
“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01110”,“01100”,
“01100”,“01101”,“01101”,“01101”,“01101”,“01101”,“01100”,“01100”,“00011”,“00011”)