Target-Driven Smoke Simulator

CSEE 4840 Embedded System Design

Yun Fei
Jonathan Chang
Tianming Miao
Guanduo Li

3/27/2014

1. Project Introduction

In this project, we are going to implement a target-driven smoke simulator, more specifically, the system will generate a smoke that has a sequence of different states and it will exhibit natural-looking smoke-like behavior [1]. Moreover, the smoke will be in its initial state, and then diffuse towards the shape that we define in the system. An overview of the project is shown in figure 1. Throughout the project, we are going to use the IEEE 754 half precision stand to do the calculation for the good estimation.

The algorithm will be based on Raanan Fattal and Dani Lischinski's paper published in 2004 [1]. The paper claims that typical smoke animations are generally created by the inviscid Euler equations:

$$u_t = -u \cdot \nabla u - \nabla p + f \qquad (1)$$

$$\nabla u = 0 \qquad (2)$$

In above equations, u_t is the smoke fluid velocity vector's temporal derivative. P is the hydrostatic pressure and f is the external forces. Equation (1) is basically Newton's second law. Although this describes smoke behavior well, it does not provide good control of it. The paper brings an improved algorithm based on the previous one and allows us to control the movement of smoke animation.

$$u_t = -u \cdot \nabla u - \nabla p + v_f F(\rho, \rho^*) - v_d u \qquad (3)$$

In equation (3), we have a new term F, which is the driving force term to carry the smoke to the defined target density ρ^*. v_f and v_d are control parameters that tune the strength of the driving force and attenuate momentum.

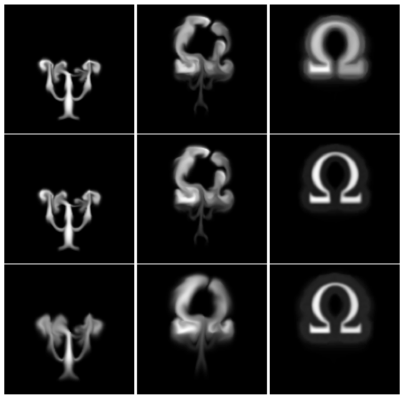
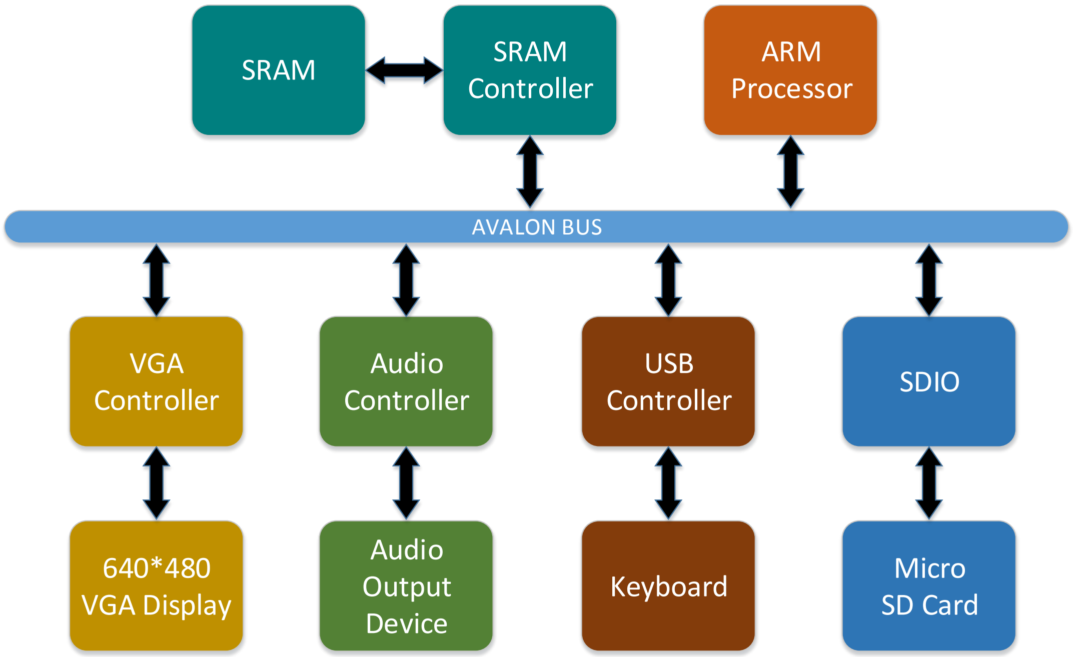


Figure 1: Simulation result

2. Block Diagram

The following figure includes block diagrams of our system. The main blocks are below:



The hardware components of the design include the USB keyboard to select the different effect, a VGA monitor to display the smoke simulator, and an audio output display the background music stored in the micro SD card.

The main blocks are listed below:

1. SRAM Controller and Audio controller
2. SRAM Controller and SDIO
3. SRAM Controller and USB Controller
4. SRAM Controller and VGA Controller
5. SRAM and SRAM Controller

IEEE 754 Half Precision floating point standard

we are going to use the IEEE 754 half precision floating point formats to represent all the bits information throughout the project after we consider the simulation color precision and the capability of the FPGA calculating ability.

2.1 SRAM memory storage

The total screen is 640*480 pixels, and for the pictures which diffused by the states with various shapes are in 256*128 pixels, since there are 16 bits in the IEEE 754 half precision format, so the total memories for a single picture is 256*128*16 = 524,288 bits which is 64Kb of SRAM. And we store 6 pictures in the SRAM, therefore, the total memory stored in the SRAM is 384Kb.

2.2 Audio Module

Our Audio block is working as the background music, so user can choose to turn on or off the music. And the music is coming from the SD card.

2.3 Keyboard

Up/down/left/right keyboards target the positions where the smoke simulations will ends.

2.4 SRAM and Floating Point Unit Athematic

Initially, we store all the pixel RGB bits information in the SRAM, and then the SRAM controller will call the memory data to feed the pixel color bits into the Floating Point Unit to calculate out the Smoke Simulation information bits based on the smoke animation algorithms. After calculating the complicated floating point calculations, the output of the Floating Point Unit will feed into VGA controller to display the images.

According to the smoke simulation algorithm, for each pixel, there are around 200 floating calculations to get the corresponding values go to VGA controller. And the below steps are the floating point algorithms:

1. Convert image from RGB to LAB Color space. (13 steps calculations)
2. Gaussian Blur in X direction. (3 steps calculations)
3. Gaussian Blur in Y direction.(3 steps calculations)
4. Add force to move smoke.(4 steps calculations)
5. Advect fluids.(41 steps calculations)
6. Gauss-Seidel solving for incompressibility.(16 steps calculations)
7. Apply the corrected pressure to velocity. (6 steps calculations)
8. Gather the smoke towards the input image.(70 steps)
9. Combine the smoke in the L channel with A-B channels. (37 steps calculations)

2.5 VGA Module

The VGA controller controls all RGB variables that determine the images on the 640*480 display. Therefore, according to the Sockit user guide, we can set the inputs and outputs. (Figure 2)

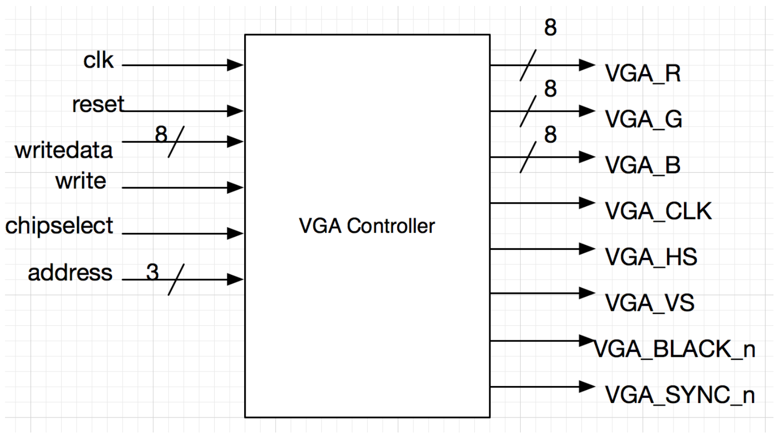


Figure 2 VGA Controller inputs and outputs

Inputs:

Clk: the clock of the entire system.

Reset: reset the whole system.

Writedata: the data calculated by the Floating point Unit

Write: on if the user switches between images

Address: the memory address being handled

Outputs:

VGA_CLK: the clock of the VGA display.

VGA_HS: the horizontal sync signal of the VGA display.

VGA_BLANK: The blank signal of the VGA Display.

RGB: The color of the current pixel

<u>Reference</u>

[1] Fattal, Raanan, and Dani Lischinski. "Target-driven Smoke Animation." ACM Transactions on Graphics 23.3 (2004): 441. Print.