Embedded Systems Design Doc

Tameem Asif
tma2153

Patrick Puma
pmp2149

Abstract

This document contains a high-level overview of what the project will look like with some technical details (including diagrams) as well as potentials milestones for the rest of the semester.

Contents

1 Project Overview

In our project, we aim to imitate and improve upon the 1977 Atari 2600 game, Combat. The original game includes various numbers of tanks and planes levels but we will use the core gameplay from the tanks levels. We will have 3 game modes (a mix of levels from the original game and our own inventions) with various static obstacles. For more on game modes, please refer to section 3 on algorithms.

2 System Block Diagram

Please refer to Figure 1 below for our block diagram.

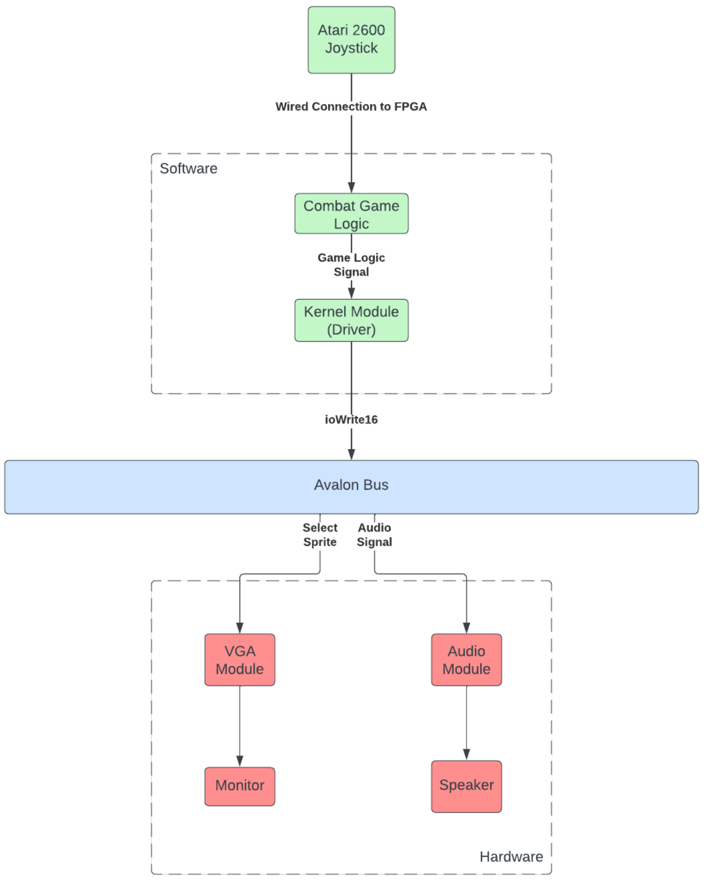

Figure 1: Block Diagram for the entire system

Figure 1 shows what our entire system is supposed to look like at a high level. On the software side, we will have all the C code that contains the logic of the game and the movement of the sprites. The logic uses the device driver (codes as a kernel module) to send the sprite and audio selection information to the hardware through the Avalon bus. On the hardware side, there are 2 modules: the VGA module and the audio module. The VGA module receives information about which sprites to display where and displays that on the VGA monitor. The audio module receives information about the proper audio file to play and plays it through the attached speaker.

3 Algorithms

We intend to have multiple game modes and the first game mode will be the classic tank vs. tank game mode. Each tank will be able to shoot a single bullet at a time. The bullet's trajectory will turn as the tank rotates in place. If the bullet hits the opposing tank, the opposing tank is stunned and knocked back while spinning in a circle. The shooting tank receives a point and there is a cool-down before either tank can move and shoot. The game lasts for a minute after both tanks cannot move and the winner is the tank with the most points.

The second game mode is the same as the first except that the tanks will be invisible during movement. The only times that the tanks will be visible is when the tank shoots a bullet, bumps into a wall, or gets hit.

The third game mode (a bit more of a stretch depending on whether we complete the first two game modes) is a cooperative game mode where the two tanks will play against endless waves of computer-controlled tanks. The controls will be the same except each player tank will have 3 lives and the game ends when both tanks lose all their lives. The goal is to survive the longest and see how many enemy tanks the players can defeat cooperatively.

Additionally, since the gameplay is straightforward, we will add more features to increase engagement. A staple of older games are inputs that led to secret abilities or to enable cheat codes in games (aka easter eggs). In the same nature, we will build that in as a feature to enable a charge shot for the tank.

Hardware: We will be creating a Main.sv module that takes in the input from the Avalon bus and then as the raster is being created, it will reference two other modules using the sprite and tile method, which we will go over in more detail in the hardware-software interfaces section.

Software: The code will be written in C and through a kernel module of our design, we will write and read data from the Avalon bus to interpret where the tanks will be located. The scripts will handle where the tanks move, the borders of the map, and how far a bullet will travel. It will also handle scoring and the timeouts for the stuns when a tank is hit.

Movement of Combat Tanks:

- Rotate in place
- Move forward (Not backward!)
- Turn while moving

- Fire bullet that disappears after traveling a set distance
- Knockback when hit
- Clipping behavior when knockback close to walls

For the (bonus) mode when the players are fighting computer tanks. They will be controlled by the A* algorithm and then when in range of the player will shoot bullets at them.

4 Resource Budgets

We will be store all the tiles and the audio in the FPGA TOM IP Core Module which will allow our VGA module to quickly access the data.

Each tile in Figure 2 and obstacles (not shown here) will be 22x22 pixels while each tank will be 28x28 pixels as shown in Figure 3.

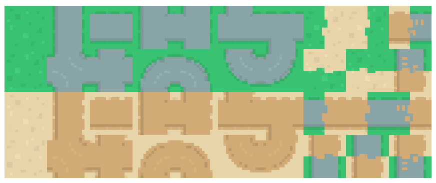

Figure 2: Tilemap we intend to use.

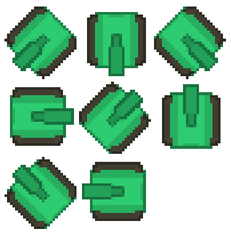

Figure 3: The green tank sprite sheet. There will be two more identical ones but with different colors: one for the opponent tank and another for the computer-controlled enemy.

Since we are doing an updated version of tanks, we will be using a much higher resolution and more varied colors than that of the original game. We will be attempting to do a resolution of 1280x720. Figure 4 shows the background in reference to the tank sizes.

After accounting for the sizes of all the sprites that we will be using, Figure 5 shows the estimated sizes for the audio and the sprites that we will be using in the game.

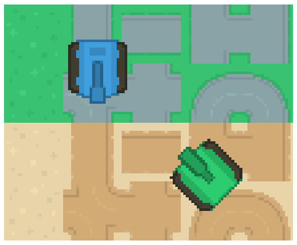

Figure 4: Tank overlaid on tile map

Object	Pixel	Size
Tank Spritesheets	3*30*30*8	~2KB
Map Tiles	22*22*8	~4KB
Objects	22*22*8	~4KB
Score Board	22*22*8	~4KB
Audio	--	~250KB
Total	-----------	264KB

Figure 5: Estimated resource usages

5 Hardware/Software Interface

An `iowrite16` will be sending the data between the hardware and software through the Avalon bus.

Figure 6 above shows the registers we will need for this game. Because we aim to have a screen resolution of `1280 x 720`, we have allocated the registers properly to account for keeping track of the positions of the two tanks as well as the two bullets that each tank shoots. To store game information that is constantly being updated, we have allocated another register for score and timekeeping.

Figure 7 above shows what the inputs of the Atari 2600 Joystick will be and how the software interpret them.

Graphics: Instead of using the very simple sprites that the original game used, we will create our own pixel art sprite sheet with the help of a copyright free vector art sprite sheet we found the internet[1]. We will be using GIMP to convert the art into the properly sized images for our game. See section 4 for more details on resources.

[1]https://kenney.nl/assets/top-down-tanks-redux

Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Remark
00	X-Position for Tank																Tank 1 X-Position
01	Y-Position for Tank																Tank 1 Y-Position
02	X-Position for Tank																Tank 2 X-Position
03	Y-Position for Tank																Tank 2 Y-Position
04	X-Position for Bullet																Bullet 1 X-Position
05	Y-Position for Bullet																Bullet 1 Y-Position
06	X-Position for Bullet																Bullet 2 X-Position
07	Y-Position for Bullet																Bullet 2 Y-Position
08	Score								Timer								Game Info

Figure 6: This is our registers table.

Button	**Function**
Up	Tank move up
Down	---
Left	Rotate left
Right	Rotate right
Fire Button	Fire bullet

Figure 7: This is our table mapping the inputs of the controller to their function.

Controllers: We intend to use 2 original Atari 2600 controllers (see Figure 8) for the game. We have found the pinout for the controllers[2] and will use that for connecting to the FPGA.

6 Milestones

6.1 Milestone 1

Our first milestone will be the ability to move a single tank with one of the Atari joysticks. Additionally, we want to be able to fire bullets from the tank and have them move across the screen. Finally, we want to incorporate one of the sounds effects (specifically the sound of the bullet firing from the tank) into the game.

[2]https://old.pinouts.ru/InputCables/JoystickAtari2600_pinout.shtml

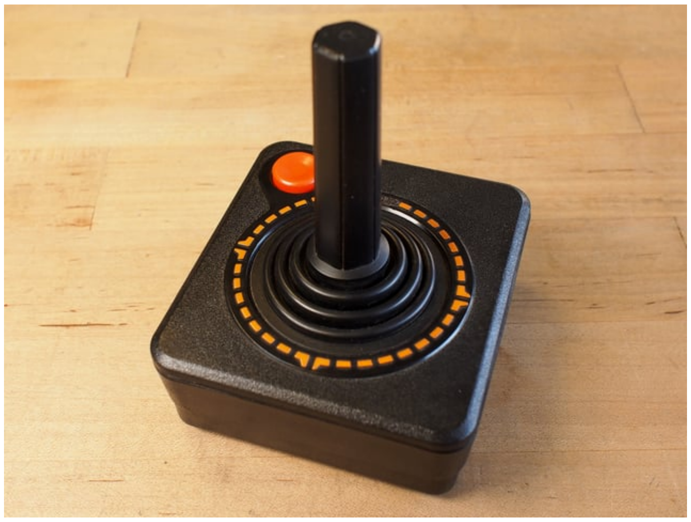

Figure 8: These are the controllers that we intend to use.

6.2 Milestone 2

In the second milestone, we want to have both tanks moving and firing bullets. We also want to incorporate proper knock-back mechanics and keep track of the score plus the timing aspect. In terms of audio, we would now want to incorporate the rest of the audio sound effects.

6.3 Milestone 3

For the third and final milestone, we want to add the different game modes. We will add the static obstacles along with the invisible tanks game mode and the bonus game mode. Then, it is just a matter of tuning and putting the finishing touches.