Asteroids

ECE 287

Hanqing Nie

Joshua Smith

Dr. Rajasekhar

The goal of this project was to use an FPGA board and vhdl code to recreate the classic arcade game Asteroids from 1979. In this game, the users control a ship to dodge the asteroids and try to fire bullets to destroy the asteroids. This game was a test of programing ability and self-study skills.

For this project, the software Quartus was used to program an Altera Cyclone V FPGA board using the coding language vhdl. Some knowledge used for this project came from lecture, but was not enough to complete this project. It was necessary to understand more details about this language, such as signals, variables, case statements, and graphics in order to display on a monitor, take inputs from the FPGA board, and set up a clock. These components were then used to move a ship and asteroids, fire bullets, and detect collisions.

The first part attempted was the motion of the ship, bullets and asteroids. Integer signals were used to keep track of each component's position. The ship movement and firing was done with the first five switches on the FPGA board. The first two switches move the ship left and right, the third and fourth switches move the ship up and down, and the fifth switch fires the bullets. The ship can also move diagonally by simultaneously flipping two switches. For the graphics, sample code from the Internet about VGA was used. This sample code initially displayed two colors on the screen, but was altered to display the motion of the ship, bullets, and asteroids [1]. Collisions were detected by comparing the distance from the center of one object to another with a predetermined collision distance. If the ship collides with an asteroid, the game ends. When the game is over, all the pixels on the screen are briefly set to red. After that, the game resets and puts the ship back to its starting location. The game can also be manually reset with the farthest switch left on the board which was chosen so that the user would not accidentally press the switch while playing the game. If a collision is detected between one of the bullets and one of the asteroids, the asteroid will disappear and the score increases. A bar at the top of the screen displays the score, and includes a green rectangle that indicates the high score for each game session.

This project is fully functional, but did not quite meet the requirements laid out in the proposal. The ship can be moved in eight directions and fire the bullets by switches as opposed to moving by having two switches rotate the ship and one switch accelerate the ship. There was also not enough time to implement the keyboard. The asteroids are fully functional, instantiating pseudo-randomly and moving automatically. Once an asteroid is destroyed, it will disappear and the score will increase. After a short delay, the asteroid reappears in a pseudo-random location. The bullet firing is also a fully functional feature, allowing the user to fire one bullet and then a second bullet if they choose before the first has decayed. At the beginning, only one bullet could be fired at a time, but it was found that such a setup made it difficult for players to hit an asteroid.

The most difficult part of the project was the VGA. Sample code from the internet was ultimately used to display graphics on the monitor. Every pixel clock cycle, the entire screen refreshes by sweeping through each pixel in one row and sweeping through every row on the screen. Each pixel is assigned a three-bit color code according to its location on the screen. Various if statements were used to display each separate component.

This project achieved the fundamental requirements and expectations that were set at the beginning. It consists of a fully functional interface that includes user input that controls the ship in order to interact with the rest of the components in the game.

**Program Operation:**

When the code is uploaded to the board, the game immediately begins. The ship appears in its initial location as do five asteroids which immediately begin moving. The user uses switch 0 to move right, switch 1 to move left, switch 2 to move up, switch 3 to move down. The ship can also move diagonally with the appropriate combination of switches. Switch 4 fires a bullet which moves in the direction the ship was facing upon firing, and decays after a short time. A second bullet can be fired a few moments after the first bullet is fired if the first bullet has not yet decayed. If a bullet strikes an asteroid, the asteroid disappears and adds one to the score. The asteroid then reappears a few moments later with a pseudo-random location and velocity.

The score is continuously displayed at the top of the screen in the form of a yellow bar which gets longer as the score increases. A green rectangle at the top of the screen indicates the high score for the current session and continuously updates. The score is limited to approximately one fifth the number of pixels on the screen, because at that point the right side of the score bar will be off the side of the screen.

If an asteroid strikes the ship, the screen flashes red and the game resets. Resetting game sets the score to zero and puts the ship back at its starting location. The asteroids are started in a pseudo-random location with a pseudo-random velocity using an algorithm that updates each reset. The game runs infinitely while the board is running.

References

1. Larson, Scott. "VGA Controller." Eewiki.net. Digi-Key, 01 Aug. 2013. Web. 11 Dec. 2015.

<https://github.com/smithjg4/Asteroids>