EE314 DIGITAL ELECTRONICS LABORATORY

SPRING 2018-2019 TERM PROJECT:

Introduction:

This document contains the project definition of the EE314 laboratory. Here are the important points about the project:

• Note that this is not a weekend project. Start working on it now. If you would like to test your designs, you can use the equipment in the EE314 lab in working hours unless there is a laboratory session proceeding. During weekends, laboratory will be closed. However, after last experiment FPGA development boards will be lent to groups (probably 1 board for 3 groups).

• There will be two projects and groups will be assigned to one of the projects randomly.

• The aim of this project work is to make you more familiar with some subjects you were introduced in digital electronics and logic design class. However, you may need to do some research and study extra material to accomplish the task. This will be a good step for 4th year graduation projects.

• The project groups will contain at most 2 students. Although it is not recommended, you may do your project alone. So, determine your project partner as soon as possible. It is not necessary that your lab partner and project partner is the same person.

• You are free and encouraged to use your own ideas. Although your design approach is not limited, the systems are supposed to be economical.

• All assistants are responsible for the project. Primary contact mechanism with the assistants is via email.

• Early demonstration will be and it will be rewarded by 10 Bonus points.

CAUTION: Each Verilog code will be compared with the other codes via plagiarism detection program. Any copy attempt will yield 0 point from the project and disciplinary actions will be taken.

Important Dates:

-**31st May**: Final Report (There will be no late submission)

-**1st -2nd June**: Project Demonstrations

Report Format

Final Report: The final report should be in the IEEE double column paper format (please check the IEEE paper format) and it should not exceed 10 pages in total, any more pages will decrease your grade. The formatting is one of the most important parts of the project. If the final report is not in the IEEE paper format, the project will not be graded and you will get zero from the whole project. Any formatting mistake (such as no figure captions, not referral to the figure in your main text, etc.) will result in grade deduction. You have to upload your proposal report in pdf format to ODTUCLASS until 31st of May, 17:00. Late submissions will not be accepted. Your report should include the following items:

- Theoretical background and literature research
- Design methodology and mathematical analysis of the subsystems
- State diagram and timing diagram of the project
- Simulation results (with Modelsim)*
- Simulation results verifying that your subsystems and overall system is working properly. (with Modelsim)*
- Experimental results
- Comparison of the experimental results with the simulation results and mathematical calculations and explanation of any discrepancies.

*Note that, the most important part of the report is the simulation result and you have to present your simulation result performed in **ModelSim**. Required testbench codes are needed to be written by you and submitted with your report.

You should upload your verilog and testbench codes as seperate files.

Project Video: Each group must submit a video about their project. This video must be in English. The evaluation of the video will be based on the oral skills not the technical situation of the project. You do not have to show a working project. The explanation of the overall project, the solution method, and the final situation of the project are the main criteria. The total time of the video should be around 5 minutes and each group member must talk in this duration.

Grading:

-Project Demonstrations + Video: 60 pts

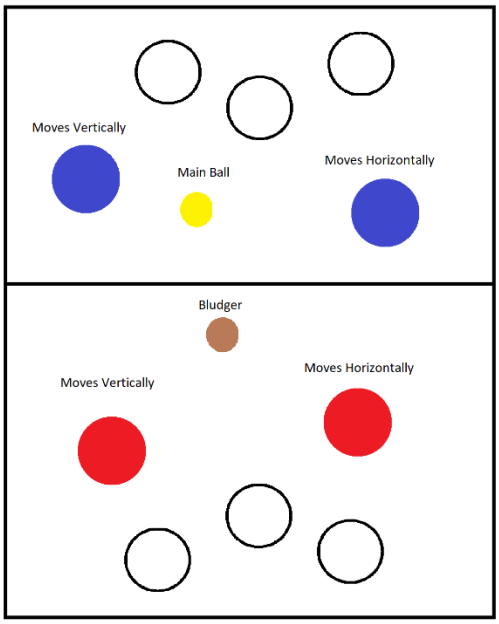
-Final Report: 40 pts

Project Definition

Project 1: Fake Quidditch

In this project, you are required to design a game, for which a representative gameplay screenshot is given in Figure 1. Note that this figure is a rough sketch, which is shown to give you an idea. You **SHOULD** implement your own design.

- The aim of the game is to score goals by passing the main ball (shown as yellow) through one of the opponents' circles. Please note that the goals should be fairly larger than the ball so as not to make scoring extremely difficult.
- Game consists of four players, two players in each team. One of the players in team is only able to move on a horizontal axis and the other one is able to move on a vertical axis. There is a main ball which is controlled by players by hitting it. In addition, there is another ball called "bludger", which scatters randomly from field boundaries. If it hits one of the players, that player will be disabled for 10 seconds.
- Players and balls should have circular forms as shown Figure 1. When balls hit a surface (player or field boundary) it should be reflected properly. In other words, reflection angles should be calculated according to basic physical principles. In addition, players should not be able to go outside of the field.
- Players will be controlled by the buttons in the FPGA board. However, the number of buttons may not be sufficient. In this case, you can plug your own buttons to GPIO pins of the board.
- The players are rigid; they cannot pass through each other.
- There should be a score board, which shows the current scores of each team.
- There should be a timer, which shows the remaining time for the game. In addition, if a player is disabled, remaining penalty time should be presented.
- Each game lasts 3 minutes.
- Orientation of the field should be the same as shown in Figure 1.
- You will use VGA interface to obtain colorful images on the computer.



Project 2: Pinball

In this project, you are required to design a game, for which a representative gameplay screenshot is given in Figure 2. Note that this figure is a rough sketch, which is shown to give you an idea. You **SHOULD** implement your own design.

- Aim of the game is to score points by striking the **randomly** placed targets by a ball(YELLOW). The game consists of several parts.

1. **Plunger:** Plunger provides the initial movement to the ball.
2. **Flippers:** Flippers are used to control the direction of the ball. In addition, if the ball passes between the flippers, the game ends.
3. **Targets:** There are three types of targets.
    a. **Circular targets:** If the ball hits one of these targets, it will bounce back according to simple physical principles. Which results in 10 points. (RED Circle)
    b. **Scattering targets:** If the ball passes through a scattering target, it will be scattered randomly every time. Scattering angle should change randomly at each time. Which results in 20 points. (HEXAGON)
    c. **Penalty targets:** Penalty targets have the same physical properties as circular targets; however, it results -15 points. (GREEN Circle)

- Flippers are controlled by the buttons in the FPGA. When a button is pressed, corresponding flipper should start to rotate around a pin point. This movement should be done in a **CONTINUOUS** manner. This is important since, direction of the ball will be controlled by the angle of flipper at the collision instant.
- When the ball hits a flipper, it will be reflected and its reflection angle will be perpendicular to the flipper at the collision instant. This is shown in Figure 3.
- Speed of the ball should decrease as it goes towards to upper boundary and it should speed up as it goes down. Note that when flipper hits the ball, the ball should gain enough speed to reach the boundary.
- The ball cannot go across the boundaries; it should be reflected properly.
- In addition, lower boundary should be designed such that the ball always reaches one of the flippers.
- There should be a score board, which shows the current score.
- There should be a timer showing the time passed during the game.
- Each group **SHOULD** place their targets **RANDOMLY**. There will be two of each targets.

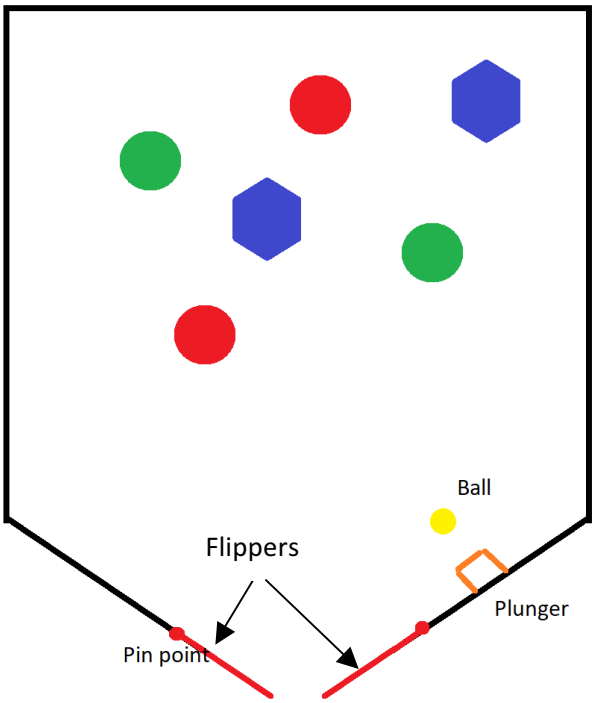


Figure 2

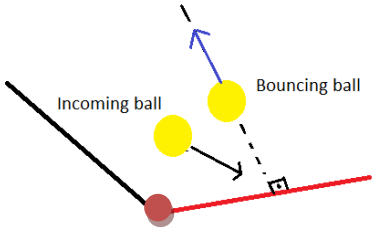


Figure 3 Bouncing of the ball from flipper

Appendix: VGA Interface

VGA is a widely used standard in video industry for the transmission of video signals from a computer or microprocessor into a monitor or TV. Each 640x480 image is called a 'frame' and each frame contains 480 lines which are made up of 640 pixels.

The monitor starts displaying each frame by beginning from the first line and then the first pixel of this line. In each line, the display order is from left to right; and each frame is written in an order from top to bottom. So, your first pixel is always at the top left corner, while the last pixel at the bottom right.

You will need to generate an image buffer with at least 640x480 = 307200 bits to store each line and frame in order to form a coherent image; however you will also need to adjust two synchronization signals called HSync (Horizontal Synchronization) and VSync (Vertical Synchronization) in order to see a video. These signals tell the monitor when a line or frame is finished, and the monitor should start from the next line or frame.

As shown in Figure 4, VGA interface is actually very simple, and you will only need to make 3 connections, namely R-G-B. For example, for a white pixel all three inputs should be high, and for a black pixel the inputs should be low. The FPGA cards in the laboratory already have a VGA output port with color outputs, so you will only need to supply the R-G-B data digitally to the VGA port. Necessary pins for these assignments can be found in the user manual.

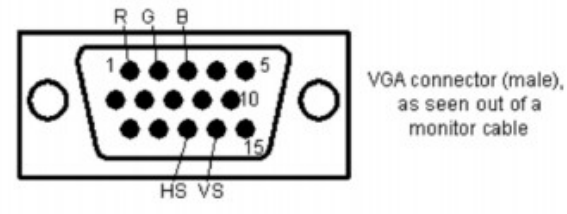


Figure 4: VGA interface.

HSync and VSync: HSync and VSync are necessary in order to tell the monitor to 'start' or 'stop' writing a line or frame. You will need to build the necessary digital blocks in order to correctly form these two signals. These blocks are basically counters with some modifications and are very easy to implement in Verilog. You can see the horizontal and vertical synchronization signals in Figure 5 with the corresponding timing in Table 2.

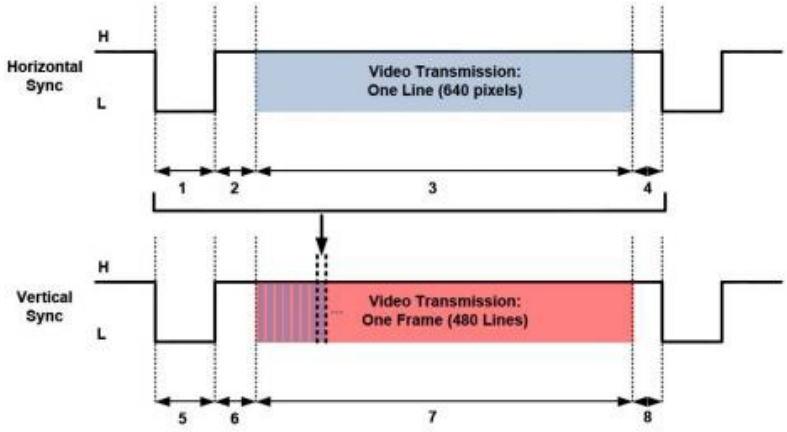


Figure 5: HS and VS.

Timeline # on Fig. 1	Name	Duration	Clock Count
1	H. Sync	3.84 µs	96
2	Back Porch (H)	1.92 µs	48
3	Video Signal (One Line)	25.6 µs	640
4	Front Porch (H)	0.64 µs	16
5	V. Sync	0.064 ms	2
6	Back Porch (V)	1.056 ms	33
7	Video Signal (One Frame)	15.36 ms	480
8	Front Porch (V)	0.32 ms	10

Table 2: Timing.

By observing Figure 5 and Table 2, we can understand that the HSync signal is used to synchronize one line in a frame, while VSync is used to synchronize each frame. Basically, when HSync or VSync is low, the monitor understands that it needs to switch from one line or frame to the next. Back and front porch are idle stages where the monitor is getting ready to write the next pixel or line. They also include 8 pixel and line over scan or 'border' pixel/lines outside our standard view of the monitor.

IMPORTANT NOTE: The video input signals (R, G, B) of a VGA monitor should be off (or black) during H. or V. Sync stages, and front/back porch stages. The video input signals should only be active during an active video transmission stage, which are highlighted in Figure 3.
In order to construct these HSync and VSync signals and to achieve transmission of each line/pixel, you will need a 25 MHz clock signal. This will also mean that each pixel will be transmitted at 25 MHz to the monitor during active video stages.
Internal clock information about ALTERA can be found under the Clock Circuitry part of the user manual.

http://www.epanorama.net/documents/pc/vga_timing.html
http://martin.hinner.info/vga/640x480_60.html