Project Design and Planning Proposal

Vikas Dhaiman

William Poole and Austin Morin

11/19/2021

1. What do you understand by the pong game implementation? What will be the final product?

Pong is a game where a block or ball moves around the screen and bounces off 4 select surfaces. Two of the surfaces are known as the walls that are either invisible or coloured differently to the other surfaces, they are also uncontrolled and immobile. There is typically a controlled surface or paddle that is only allowed to move orthogonally to the walls. And on the other end there is either another player paddle or for our implementation another wall to practice on. In the player vs player mode there is a score keeper for when the ball goes beyond a player paddle.

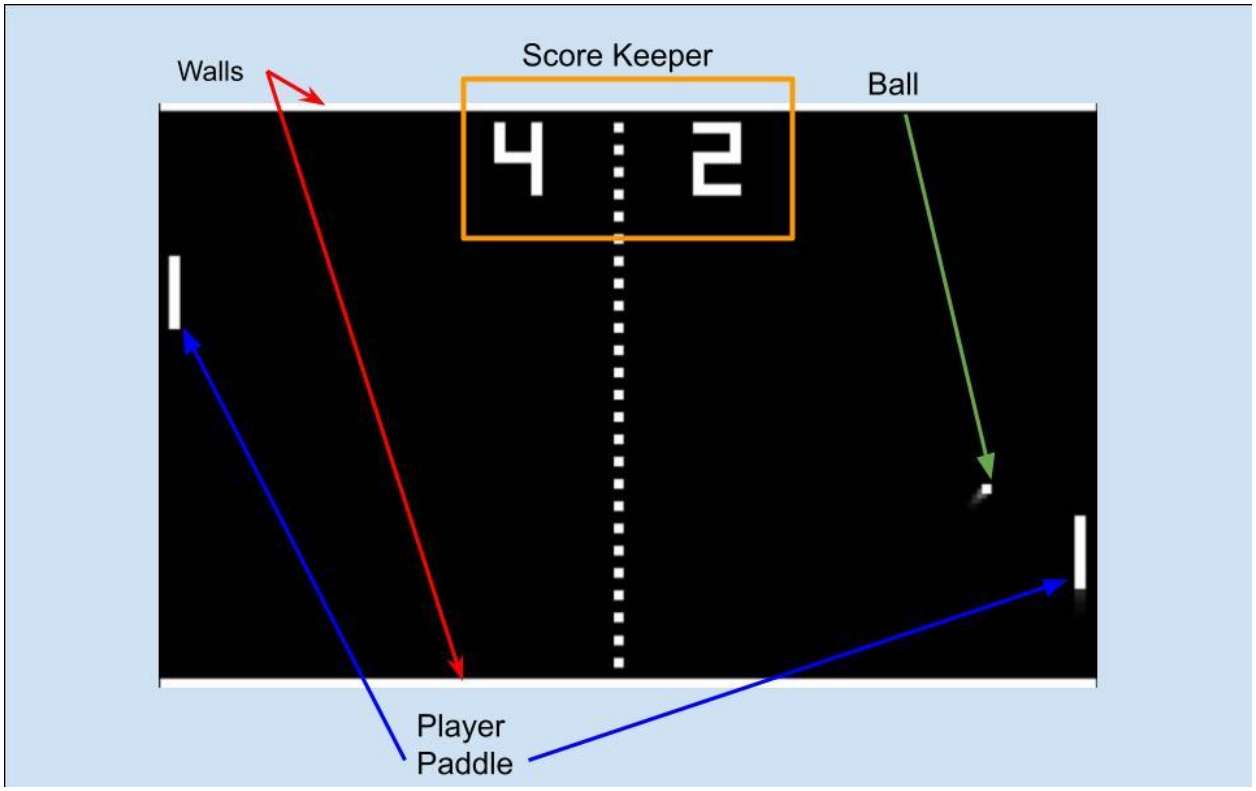

Figure One: Normal Pong Game

(Source https://www.imaginarycloud.com/blog/content/images/2019/02/Pong.jpg)

(Starting point): Reproduce the VGA interface demo, from DE0 demonstrations.

(Minimal): A moving ball demo.

Setup the ball to move in a many angles across the screen using VGA output

(Stage 2): A bouncing ball demo

Setup top two walls to block the ball and bounce it around the screen, and verify that the ball bounces around in a playable manner.

(Stage 3): A ball that bounces off a paddle

Add a player paddle to bounce off of and a block on the right and have score keeper. As well as wall on the other side of screen, using BUTTON0 & BUTTON1

(Stage 4): Control the paddle with human input

Allow the paddle to be controlled by human input.

(Ambitious): Insert your crazy idea here.

Make mode from stage 1 as a "pause" screen for stage 4.

2. How can each outcome be tested? What is the expected range of inputs? What is the expected output for each input?

(minimal):

Get VGA driver code to output correctly. Program a ball that can move depending on an (x,y) vector. The ball will have to be reset before the next input can be executed. The vector direction will be a clock based random vector with a pause switch that stops the ball and inputs a new vector to change the direction of the ball.

(stage 2):

Once again have the ball spawn on the screen but, be surrounded by walls, or encased in a rectangle that keeps it within the walls and has it bounce around. Same inputs as minimal to verify if it bounces off of every surface correctly.

(stage 3):

Change one of the walls to a smaller block with the possible areas for scoring, one again same inputs to check for each possible location for glitches and scoring capabilities.

(stage 4):

Monitor the the paddle movements by human input, the range will be two inputs just switches one controls move up and the other controls movement down. Scorekeeping will be enabled to make the game playable.

Ambitious:

Same as stage 4 but, now another switch will turn it into stage 2 so that the game becomes some sort of a pause screen.

4. What are the input and output ports, switches, LEDs on the FPGA board that you are going to use. Are you going to develop and implement your own protocol or use existing modules for interacting with the FPGA board?

Ideally, the only inputs utilized would be the three switches or two buttons and a switch (two for paddle movement and one for pause screen). Outputs would just be the VGA onto the monitor. I believe that we would use an existing module for VGA utilization, and maybe some for bouncing mechanics other than that all others would be self made.

5. How will the project be broken down into smaller modules? The minimum number of modules is three: input, core, and output. The core of your implementation can be further broken down into smaller modules. Each module's signature and documentation (what it is expected to do) will be specified at this stage.

We will use multiple modules listed below:

-Input

- player input (Buttons/switches)
- game feedback (previous state; screen, score, and ball direction)

-Core

- ball (spawning and location)
- wall/barriers (collision detection)
    - direction vectors (math for ricochet)
- game start (initialization and setup)
- scorekeeping (detection of if the ball scored and point tracking)

-Output

-VGA Display Driver (takes all necessary inputs and displays them)

6. Which modules will be borrowed from external sources like Altera cookbook or the internet? Which modules will be programmed by you? How will you automatically test the modules programmed by you and those found from unreliable sources?

Depending on how the VGA display functionality works, I believe testing the version given to us would be the only external module we will have to test. Other than that everything else should be completely original.

7. Anticipated timeline for the project, including research on finding the desired modules; learning new Verilog features; evaluating, adapting, testing and debugging the existing modules; anticipated manhours for each step. Since the project is spread over 3-weeks, we expect you to spend total 5-7 hours per week per person. Since, you are new to Verilog, it is okay to anticipate your time in terms of C/C++ and then multiply it by 2.

	Task	**Person time (30 hrs)**	**Depen dency**	**Schedule**
T1	Research, implement ball the core modules starting with the ball (spawning and location) and then wall/barriers (collision detection) & direction vectors (math for ricochet)	5 hrs		Nov 20-26
T2	Implement the core modules using the VGA Driver module to display the ball and walls on screen	5 hrs	T1/T4	Nov 26-30
T3	Implement the paddle and human input modules on top	6 hrs	T2	Nov 29-Dec 1

	of the previously implemented modules			
T4	Learn VGA driver	3 hrs		Nov 25-Nov 30
T5	Implement output_adaptor and test independently.	7 hrs	T4	Dec 1-Dec 6
T6	Connect all modules together	10 hrs	T2,T3, T5	Dec 6-Dec 10