G-Pong

Leonardo Barzacchi Filippo Bertelli Francesco Urbani

University of Pisa

PSD Final Project Report

Fall 2018

January 16, 2019

Abstract

This project proposes a revisitation of the popular arcade game Pong with some user-interaction-related improvements, using the hardware description language Verilog, an Intel FPGA and Intel Quartus Prime design suite for the verification and synthesis.

Contents

List of Figures

7 Conclusions

Functional simulations and timing analysis verification confirms the achievement of the objective. They also indicate that we have room to increase the frequency of operation.

8 Further improvements

Further improvements include the use of two boards and the instantiation of an additional paddle to implement a multiplayer game setup. The two boards may be communicating over a known protocol or a specific one using the on-board GPIOs. The role of one board would be the slave, meaning just sending to the master board the necessary signals to move the paddle, whereas the role of the master board would be receiving and processing the data coming from the slave and simultaneously generating the correct signals for the display, as well as processing the signals coming from its own on-board accelerometer.

Appendices

A File hierarchy

```
.
└── gpong
    ├── pong
    │   ├── game_controller
    │   └── display
    │       ├── clock_div
    │       ├── vga_controller
    │       │
    │       ├── figures
    │       │   ├── ball
    │       │   ├── paddle
    │       │   └── outline
    │       │
    │       ├── collision_detector
    │       ├── rgb
    │       ├── gameover
    │       ├── score
    │       │   ├── current_score
    │       │   └── highest_score
    │       │
    │       └── seven_seg_driver
    │           ├── lut_7_seg_current_score_upright
    │           ├── lut_7_seg_current_score_upsidedown
    │           ├── lut_7_seg_highest_score_upright
    │           └── lut_7_seg_highest_score_upsidedown
    │
    ├── reset_sync
    ├── led_to_xvel_cn
    └── G_Sensor
        ├── gen_clk_200
        ├── G_sensor_read
        ├── G_sensor_write
        └── converter
```

B RTL schematics

C Verilog code

Head over this [9] URL to access the repository:
every Verilog design file, Verilog test bench file, side script, Quartus Prime project file and a handful of `.vcd`[10] waveform files are herein contained.

[9] (Request)
[10] to be opened with GTKWave or similar software

References

[1] lut_7_seg_generator.py

[2] https://onlinelibrary.wiley.com/doi/book/10.1002/9780470374283

[3] http://www.eng.ucy.ac.cy/theocharides/Courses/ECE664/VGA.pdf

[4] https://www.digikey.com/eewiki/pages/viewpage.action?pageId=15925278

[5] https://github.com/WillGreen/timetoexplore

[6] https://timetoexplore.net/blog/video-timings-vga-720p-1080p

[7] http://pages.hmc.edu/jspjut/class/f2013/e155/lectures/131028_vga.pdf

[8] http://atelier.inf.unisi.ch/ esposem/documents/myurop.pdf

[9] https://gamedev.stackexchange.com/a/4255

[10] https://electronics.stackexchange.com/a/80345