



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	3
1 Introduction	4
1.1 Preface	4
1.2 Usage	4
1.3 Tools and Software used	4
2 System overview	5
3 Blocks implementation and design	5
4 Accelerometer	7
4.1 Specifications	7
4.2 I ² C digital interface	7
4.2.1 Design choices	7
4.3 G-sensor	8
4.4 Verilog files description	9
4.4.1 Write module	9
4.4.2 Read module	9
4.4.3 Gen_clk_200 module	10
4.4.4 Converter module	10
4.5 Design choices	10
4.6 Simulations	10
5 Pong	12
5.1 Game controller	12
5.1.1 Simulation	12
5.2 Display	13
5.3 VGA Controller	14
5.3.1 Introduction	14
5.3.2 VGA Controller module	14
5.3.3 Simulation	15
5.4 Figures	15
5.4.1 Game physics	16
5.5 Collision detector	16
5.6 Score	16
5.7 Current score	17
5.7.1 Score assignment	17
5.7.2 Simulation	17
5.8 Seven-segment display driver	18
5.9 LUTs	18
5.10 Game Over	19
5.11 RGB driver	20
6 Post-fitting analysis	20
6.1 Flow summary	20
6.2 Timing analysis	21
7 Conclusions	24
8 Further improvements	24
Appendices	25

A File hierarchy	25
B RTL schematics	25
C Verilog code	25
References	26

List of Figures

1	System overview	5
2	Top level RTL overview	5
3	G-sensor placement on Terasic DE10-Lite board	7
4	FPGA to on-board accelerometer wiring	7
5	I ² C device addressing	8
6	I ² C timing diagram	8
7	<i>G_Sensor</i> module	9
8	Data formatting of full-resolution and $\pm 2g$, 10-bit modes of operation when output data is right justified	10
9	I ² C read and write mode	10
10	I ² C write mode	11
11	I ² C read mode	11
12	Pong module	12
13	Game controller FSM with meaningful transitions	13
14	Game controller waveforms simulation	13
15	Display instance, functional overview	13
16	VGA timing	14
17	<i>Figure</i> module	15
18	Ball – paddle collision scenario	17
19	Score rtl	17
20	<code>o_score</code> assignment rule	17
21	Current score waveforms simulation	18
22	Seven-segment display driver block diagram	18
23	LUT generation terminal command	19
24	RTL view of one of the four Lookup Table instances	19
25	Fitter resource Usage summary – without <i>game over</i>	20
26	Fitter resource Usage summary – with <i>game over</i>	21

7 Conclusions

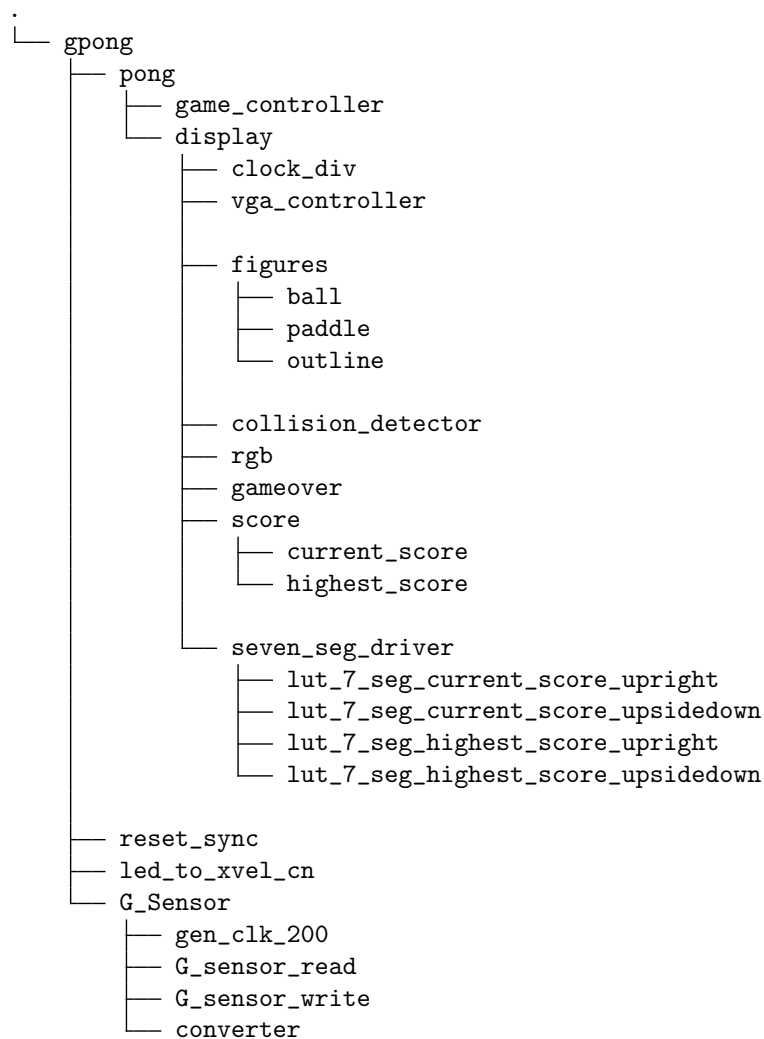
Functional simulations and timing analysis verification confirms the achievement of the objective. They also indicate that we have room to increase the frequency of operation.

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



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>