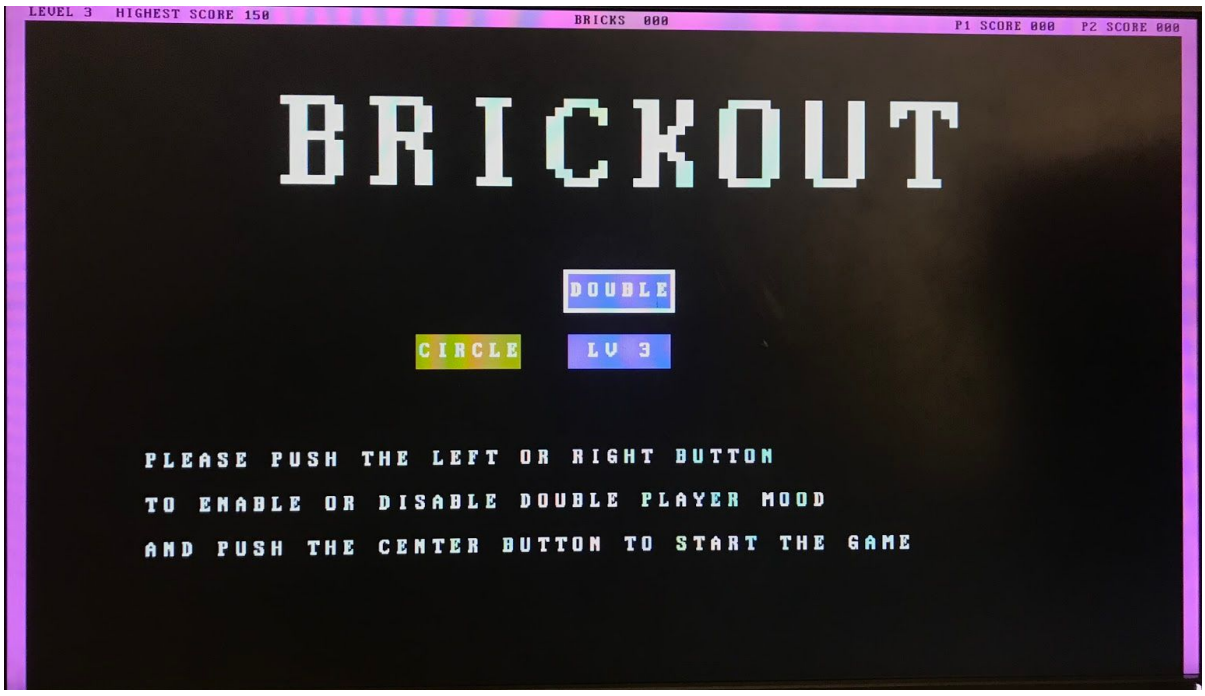

Group Report

ENSC 452 Final Project Report, Group 14

Jianglin(Arlene) Fu, Ruisi(Sam) Wang
09/04/2018

Table of Contents

1. Introduction

The goal of this ENSC 452 course is to implement a program on ZedBoard with both Xilinx's system software and hardware. We decide our project to be a video game similar to the brick breaker. In our game, the user can move the bar at the bottom of the screen left and right in order to bounce the movable ball up and try to hit the bricks. Once the ball hit any brick, the brick will be downgraded and eventually wiped out. We will split our job into 6 milestones and use Agile methodology to accept greater flexibility and to get higher efficiency. Each part will be tested by a specific test plan to ensure the completeness.

2. Background

Breakout is a traditional arcade game developed and published by Atari, Inc. And Brick Breaker is a clone of Breakout. The game consists of 50 bricks on the top of the screen, a flying ball and a movable paddle on the bottom of the screen. The paddle is controlled by the user to move either left or right. Player need to use this paddle to prevent the ball from falling out of the screen. Three unbreakable walls on the side of the screen are used to deflect the ball. The collision between the ball and bricks can make brick downgraded or disappear. The speed of ball gradually goes up with time. The level of bricks is also upgraded as time goes on, which means it requires more than one collision to smash the upgraded bricks. The game will continuously proceed as long as the ball is not falling out of the screen. During playing, the user is able to pause the game and continuous again.

This game connects to a standard keyboard to get user input and use VGA port to display the whole video screen. If we have enough time, we will play audio and make sound synchronous to break a brick. And try to use the mouse to control the paddle.

3. System overview

a. System Block Diagram

Figure 1 is our system block diagram. The blocks with white background is the Xilinx IP. Yellow block is designed by Arlene Fu, green block is designed by Rui Si Wang. Arlene Fu has more resiblsibleslity on hardware of the project, and Rui Si Wang has more in software of the project.

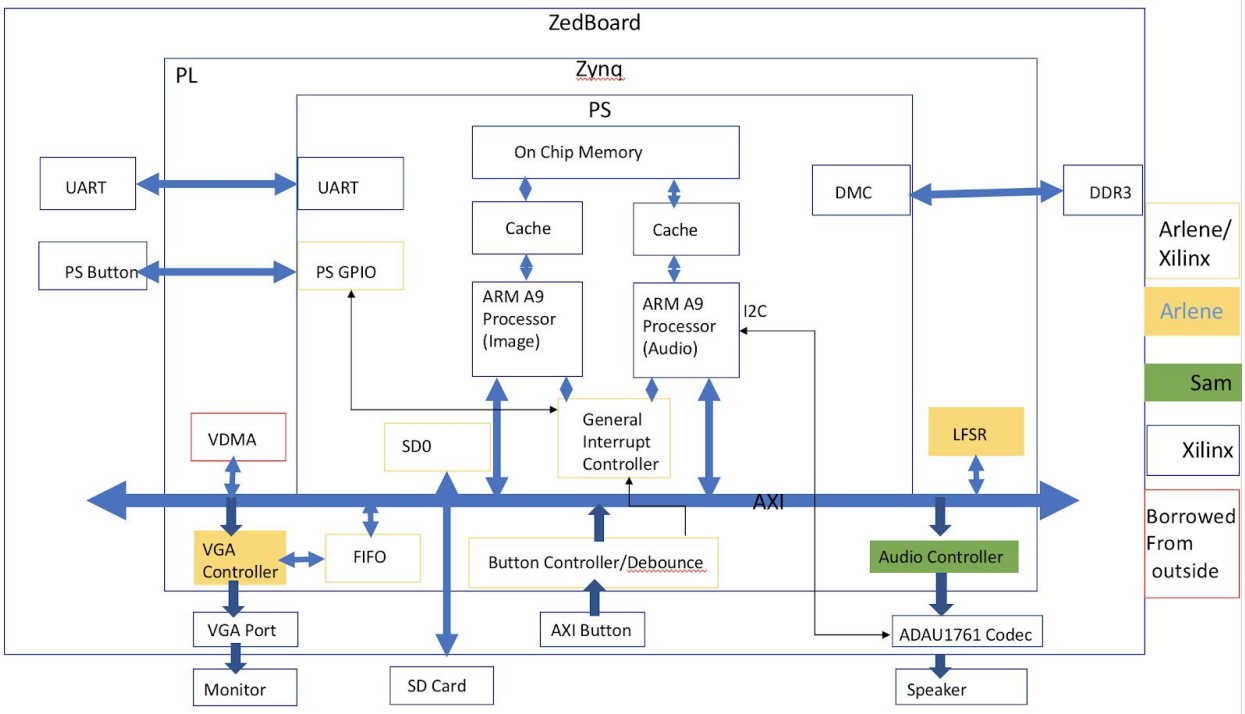

Figure 1. System block diagram

b. IP used

- Most of the IP cores used in our project are provided by Xilinx, which are presented within blue border.
- The VGA controller is modified by Arlene, the original code is from: http://blog.dev-flow.com/en/11-create-a-vga-controller-for-the-zedboard/ [1]
- The Audio controller is modified by Sam, the original code is from: https://github.com/ems-kl/zedboard_audio [2]
- LFSR is the pseudo-random number generator designed by Arlene
- Button Controller / Debounce is AXI GPIO IP with a debounce block designed by Arlene
- FIFO is a Stream FIFO, which is a Xilinx IP block but modified settings by Arlene
- VDMA is the IP block from Xilinx, but the driver code is borrowed from Wesley Kendall, https://piazza.com/class/jbenuzkh1151xw?cid=9 [3]
- PS GPIO and General Interrupt Controller are Xilinx IP blocks but Arlene implemented the interrupt service routine

c. User Manual

Before playing the game, user need to be familiarize with 7 buttons on the ZedBoard. As shown in Figure 2, the left two buttons are the control buttons for the second player. For easy understanding, the left two buttons will be called as BTN8 and BTN9 respectively. There are five buttons on the right hand side in Figure 2. For these five buttons, here define the upper, left, center, right and lower button as BTNU, BTNL, BTNC, BTNR and BTND respectively.

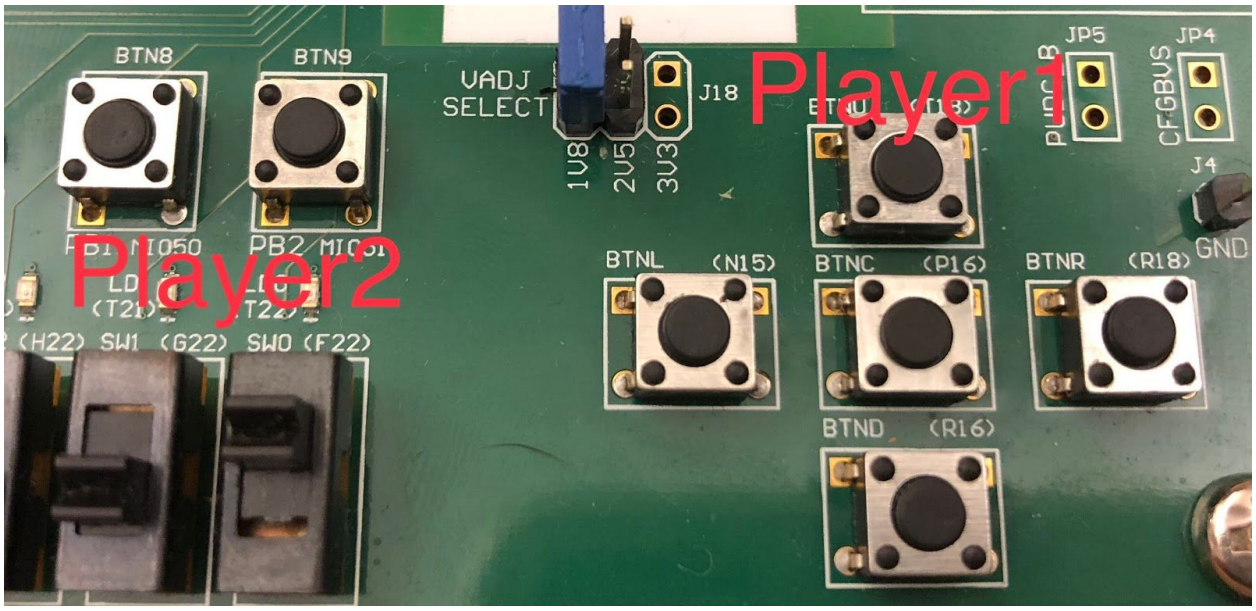

Figure 2.

To play our game, all the game logic is implemented by Sam, player can choose to boot up the ZedBoard with or without an SD card. Figure 3 shows the home menu of this game. In the top-left corner of the screen, the difficulty level and the historical highest score has been displayed. If user boot the board without SD card, then the highest score will be 0. In the top-middle part, it shows the current total brick number on the screen. Since it is in the menu page, the brick number will be zero. The upper-right corner displays the current score.

In the middle part of the screen, user can see two buttons. The first one is the place to choose single-player mode or double-player mode. User can alter the setting by press BTNL and BTNR.

If user want to change the game level, user can move to the second button by pressing BTND. The highlight around the button tells which button does user current located.

Now user can press the BTNC to increase the level. The highest level is 9, if current level is 9 and user presse the BTNC again, the the level will loop back to 1.
If user want to start the game, just press BTNU to back to the first button and press BTNC again to start the game.

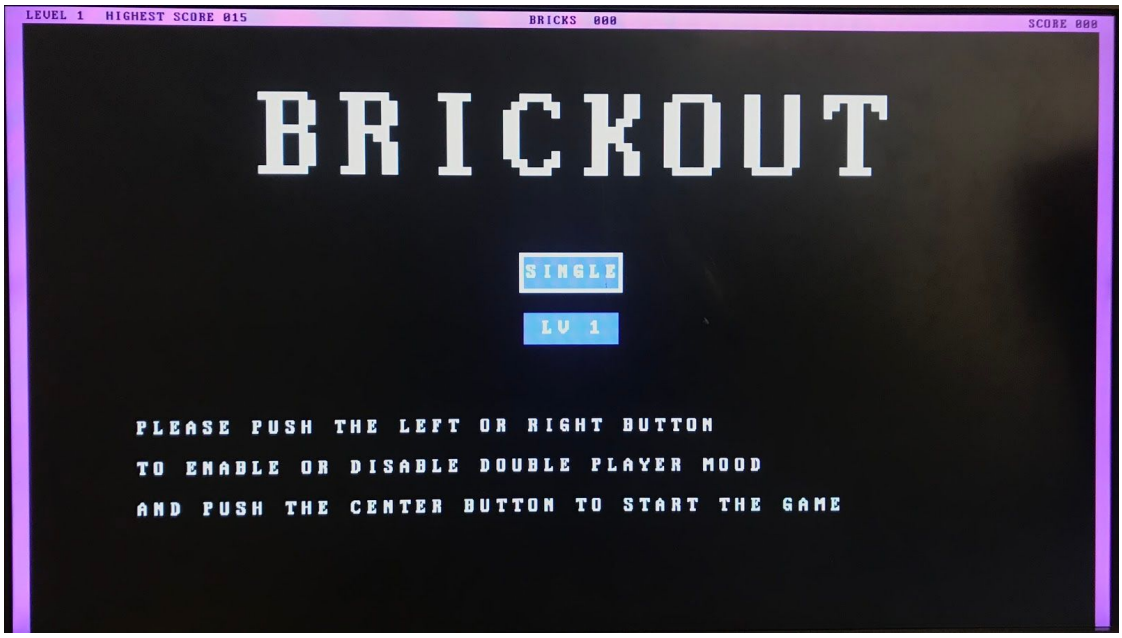

Figure 3.

After starting the game, the game interface looks like Figure 4.

There is a block of bricks located on the top two third of the screen and a paddle is in the lower part. The bricks with different color represent different levels. The brick has 4 levels in total. The darkest color represents level 4, and the lightest color is level 1. User should keep ball not falling out by controlling the panel to bounce the ball upwards. If user chooses the single-player mode, then the paddle can be controlled by pressing BTNL to move left and pressing BTNR to move right.

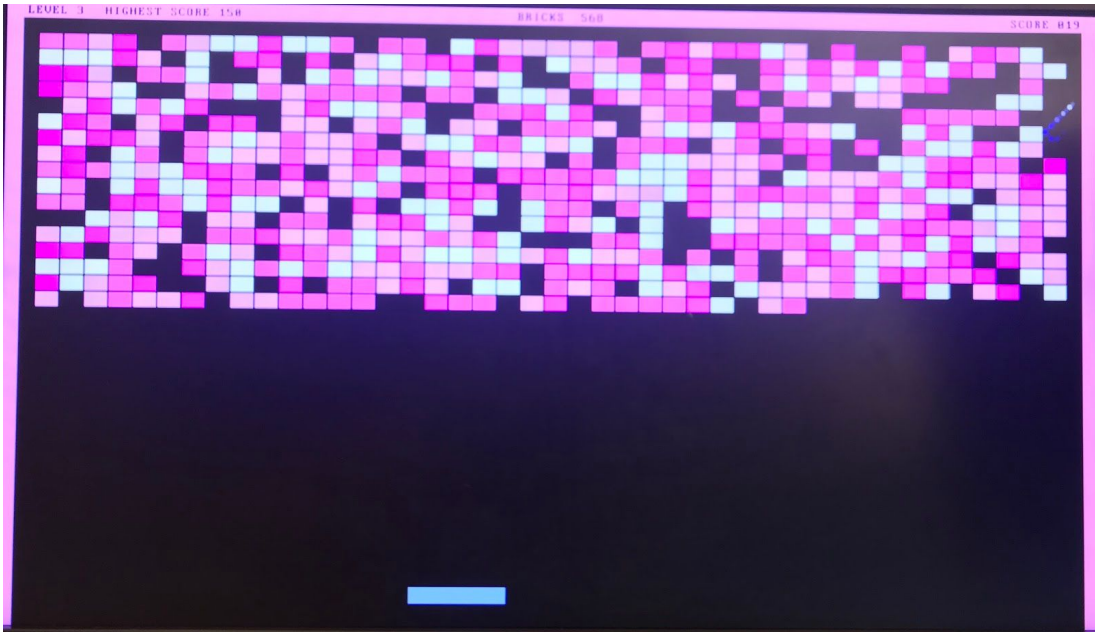

Figure 4.

If user selects double-player mode, the game interface looks like Figure 5. In this case, the blue paddle represents play 1 and player 2 controls the green paddle. As mentioned above, player 1 uses BTNL and BTNR while player 2 uses BTN8 and BTN9 to move the paddle to the left and the right, respectively.

During the game, BTNC always reset the game, move the ball to its initial state.

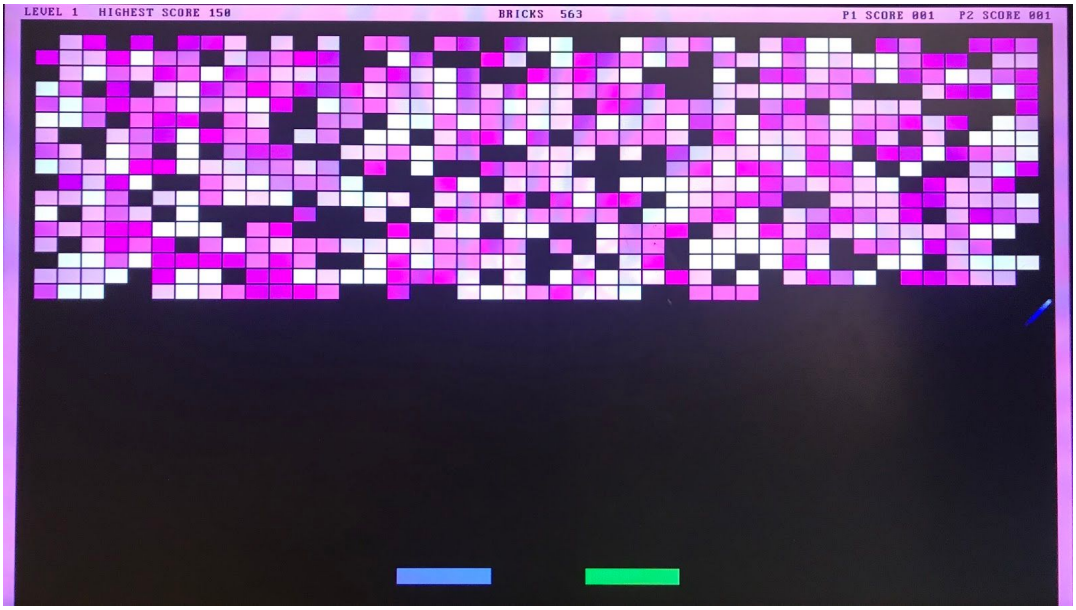

Figure 5.

During the game, press BTNU can pause the game and the interface will change to Figure 6. Press BTNU second time can resume the game.

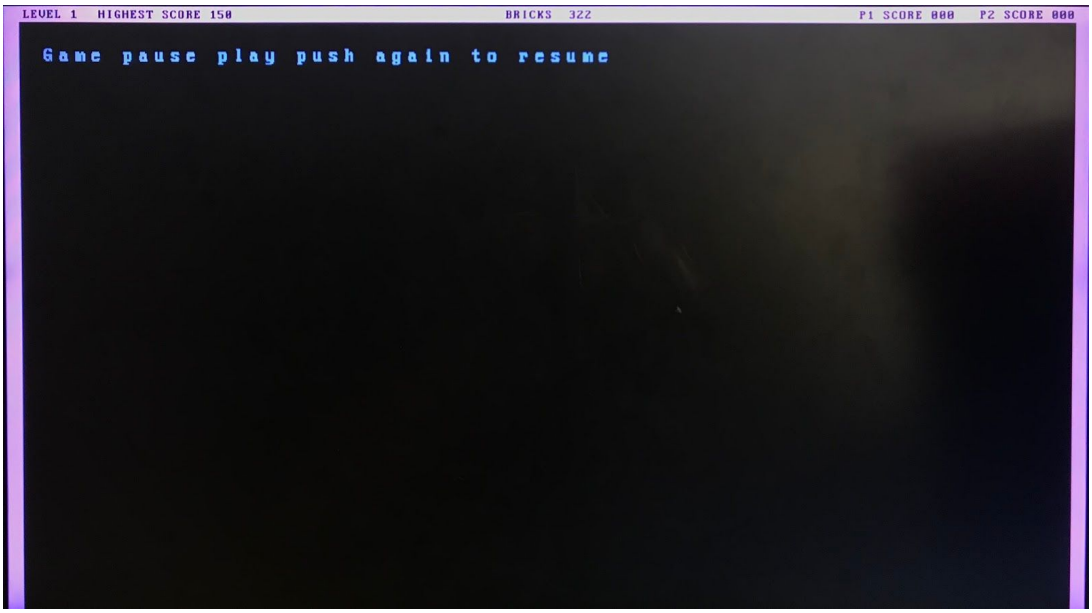

Figure 6.

BTND is the button for quit the game.

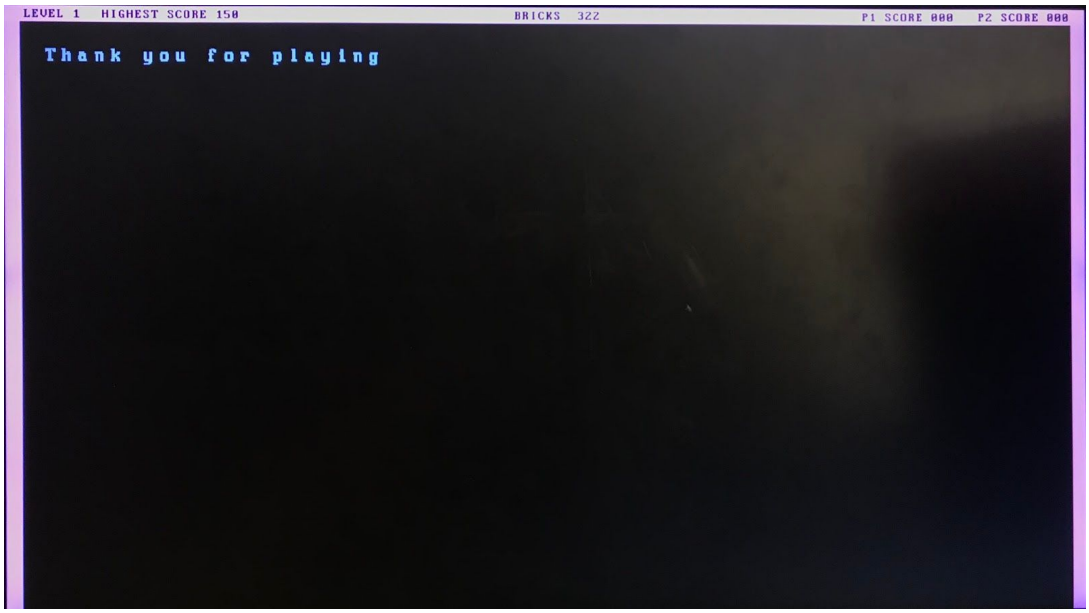

Figure 7.

Once user's current score is higher than 150, after game over for current round, user will surprisingly see an extra option on the home page, which is the circular mode, as shown in Figure 8.

Press BTND and then BTNL will move the focus to the third button. Then, press BTNC to start the game. The method to control the paddle is the same as regular mode mentioned above, but the interface looks like Figure 9 and Figure 10.

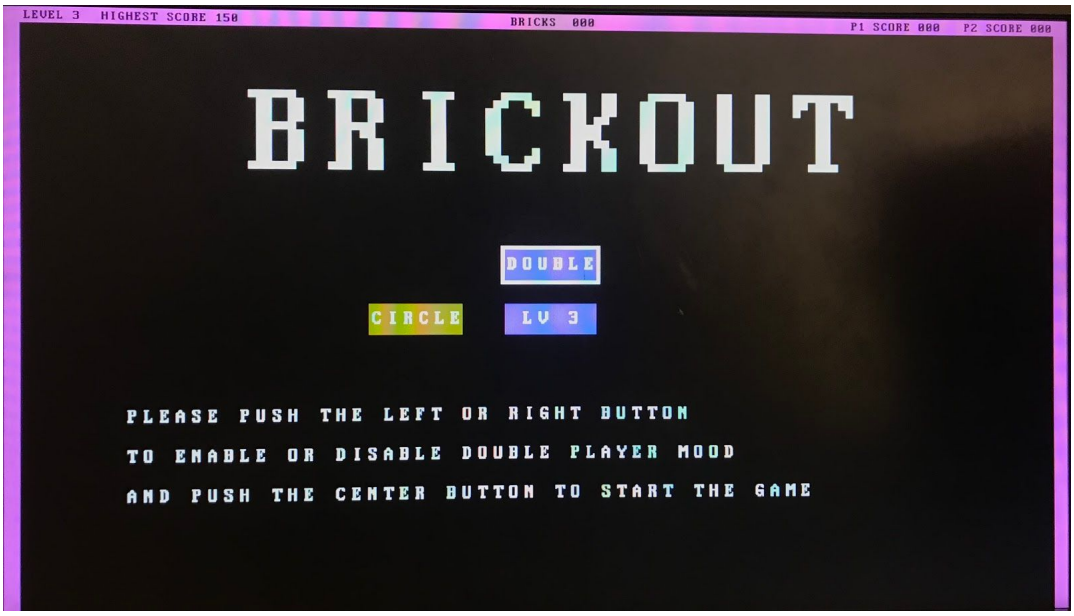

Figure 8.

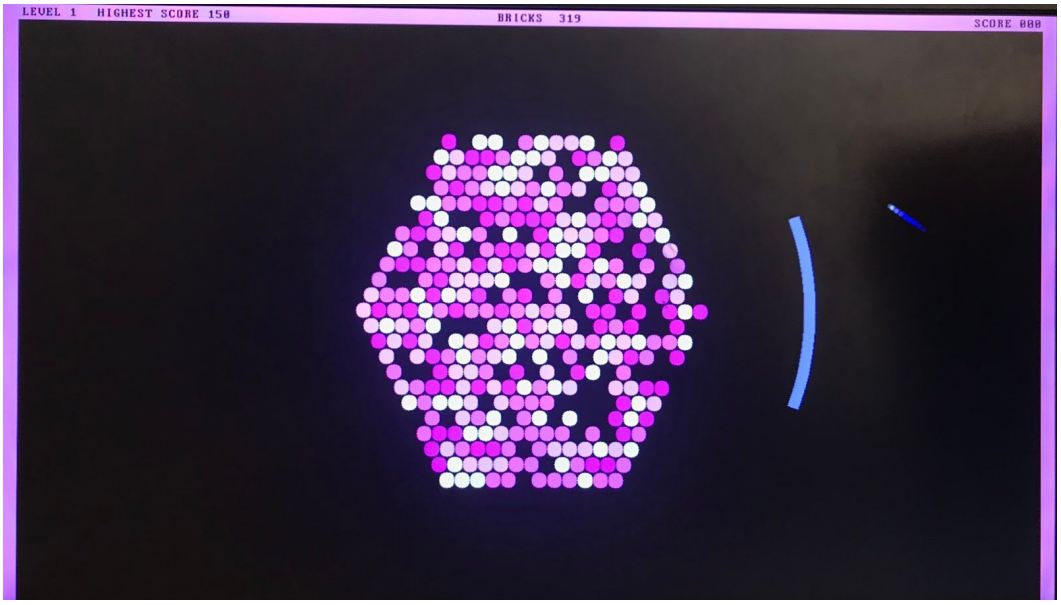

Figure 9.

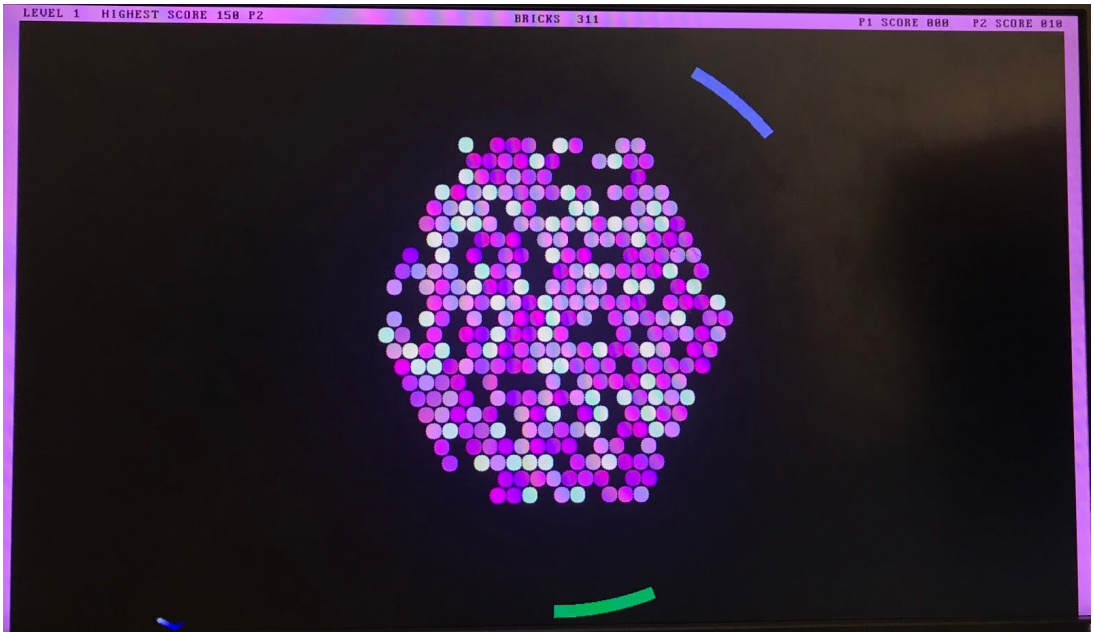

Figure 10.

4. Project Outcome

Our brick out project in ENSC 452 spring 2018 has been a total success. Its performance, efficiency, and comprehensiveness are satisfactory. Our detail outcome results are listed below:

Hardware Result

- **VGA**

  Arlene designed the VGA controller based on [1]. And the controller successfully interacted with DDR via VDMA[3]. The implementation of the double buffer helps displaying image without flicking.

- **Button Interface**

  By using the GPIO IP block provided by Xilinx, Arlene designed the interrupt service routine and debounce logic in both hardware and software. As the result, the buttons performed pretty good response towards user action.

- **SD Card**

  Arlene successfully enabled SD0 host controller on PS and implement corresponding C code to read and write the file on SD card. Now the historical highest score can be loaded every time boot up the machine.

- **Audio**

  Audio core is implemented by Sam, it is able to output tones to the audio jack, in-line input music and sound can be played to the jack as

needed. Output tones and volume and can be modified through the GPIO controller.

- **Dual-Core Processing**

  Our project is running on both CPU0 and CPU1 on the zedboard, it is implemented by Sam, each core have their own tasks and they will communicate with each other through a memory address (0X10100000) and interact with each other from time to time. Each CPU execution time will not depend on each other but exchanging information is secured and managed.

Software Result

- **Game Play System**

  Brick Out gameplay system is implemented by Sam, including general, double-play, and circular are all carefully planned and designed. Each system and module design as the black box design which means they will not affect each other other than the input. Systems are well structured, and fun for players to play with.

- **Performance**

  The project is in 1368X768 resolution, 60Hz, which is relatively high, and the two image buffers are working at a high enough speed that gives the user a smooth gameplay experience which is is implemented by Sam. Even with the double player and maximum level speed, the game will still be fluent and the paddle response will still be quick to fun with.

  By looking at the implementation design analysis in Vivado, the resource occupied by the project is relatively small. Our project used

1.842W on-chip power (see Figure 11), 5.58% of LUTs, 3.75% flip flops and 2.14% BRAM.

As for the critical path, there are three critical paths in our program. Two of them resulted from our audio part and the other came from VGA part. The slack value for three critical paths are -5.313ns, -4.648ns and -2.834ns. The audio part is a little slower than vga part.

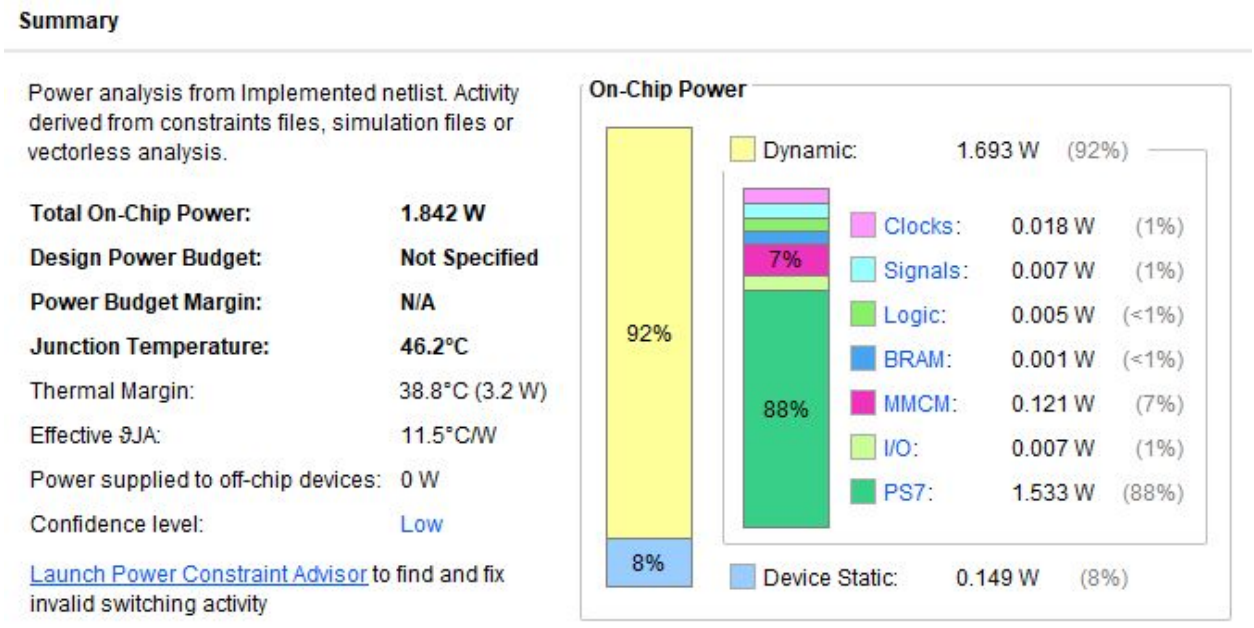

Figure 11. Power analysis

Summary

Name	Path 8
Slack	-5.313ns
Source	dma_design_1_i/audio_testbench_0/U0/i_audio/hphone_r_freeze_100_reg[13]/C (rising edge-triggered cell FDRE clocked by clk_fpga_0 {rise@0.000ns fall@5.000ns period=10.000ns})
Destination	dma_design_1_i/audio_testbench_0/U0/i_audio/Inst_adau1761_izedboard/Inst_i2s_data_interface/sr_out_reg[21]/D (rising edge-triggered cell FDRE clocked by zed_audio_clk_48M {rise@0.000ns fall
Path Group	zed_audio_clk_48M
Path Type	Setup (Max at Slow Process Corner)
Requirement	0.833ns (zed_audio_clk_48M rise@20.833ns - clk_fpga_0 rise@20.000ns)
Data Path Delay	8.982ns (logic 0.580ns (6.457%) route 8.402ns (93.543%))
Logic Levels	1 (LUT4=1)
Clock Path Skew	3.325ns
Clock Un...rtainty	0.520ns

Source Clock Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
(clock clk_fpga_0 rise edge)	(r) 20.000	20.000		
PS7	(r) 0.000	20.000	Site: PS7_X0Y0	dma_design_1_i/processing_system7_0/inst/PS7_i/FCLKCLK[0]
net (fo=1, routed)	1.193	21.193		dma_design_1_i/processing_system7_0/inst/FCLK_CLK_unbuffered[0]
BUFG (Prop_bufg_I_O)	(r) 0.101	21.294	Site: BUFGCTRL_X0Y16	dma_design_1_i/processing_system7_0/inst/buffer_fclk_clk_0.FCLK_CLK_0_BUFG/O
net (fo=4053, routed)	1.843	23.137		dma_design_1_i/audio_testbench_0/U0/i_audio/CLK
FDRE			Site: SLICE_X43Y106	dma_design_1_i/audio_testbench_0/U0/i_audio/hphone_r_freeze_100_reg[13]/C

Data Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
FDRE (Prop_fdre_C_Q)	(r) 0.456	23.593	Site: SLICE_X43Y106	dma_design_1_i/audio_testbench_0/U0/i_audio/hphone_r_freeze_100_reg[13]/Q
net (fo=1, routed)	8.402	31.995		dma_design_1_i/audio_testbench_0/U0/i_audio/Inst_adau1761_izedboard/Inst_i2s_data_interface/hphone_r_freeze_100_reg[23][13]
LUT4 (Prop_lut4_I0_O)	(r) 0.124	32.119	Site: SLICE_X40Y106	dma_design_1_i/audio_testbench_0/U0/i_audio/Inst_adau1761_izedboard/Inst_i2s_data_interface/sr_out[21]_i_1/O
net (fo=1, routed)	0.000	32.119		dma_design_1_i/audio_testbench_0/U0/i_audio/Inst_adau1761_izedboard/Inst_i2s_data_interface/sr_out[21]_i_1_n_0
FDRE			Site: SLICE_X40Y106	dma_design_1_i/audio_testbench_0/U0/i_audio/Inst_adau1761_izedboard/Inst_i2s_data_interface/sr_out_reg[21]/D
Arrival Time		32.119		

Destination Clock Path

Delay Type	Incr (ns)	Path (...	Location	Netlist Resource(s)
(clock zed_audio_clk_48M rise edge)	(r) 20.833	20.833		

Figure 12. Critical path

- **Completion**

  Brick Out project is completed. the game and its details meet all the requirements that we agreed on and all the components were design are all implemented.

- **Algorithms**

  The calculations and algorithms in our game is implemented by Sam, it act like a complex physics engine that will let the players feel "normal". the movement of the Ball and the Paddle are all well considered and corner cases are more than 90 percent covered. Although the calculations are relatively large, we still get a quick response from it.

5. Description of The Blocks

- AXI GPIO and PS GPIO were used to transfer data and trigger interrupt.
- Debounce block for AXI GPIO was designed by Arlene.
- Arlene enabled the SD/SDIO host controller on the board and added the code for accessing the non-volatile memory.
- Button controller is the AXI GPIO IP block provided by Xilinx and the interrupt controller of that was designed by Arlene.
- Pseudo-random number generator (LFSR) was designed by Arlene. It is used to generate random brick level.
- VGA Controller is modified by Arlene, the original code is from: http://blog.dev-flow.com/en/11-create-a-vga-controller-for-the-zedboard/ [1]. This IP block is used for displaying pixels on the monitor.
    - Changes had been made to the source code including:
    - Change the resolution of the display to 1368x768, including changing all the parameters such as h_sync, v_sync and back porch.
    - Add logic to detect the end of writing each frame, and output an interrupt each time finish drawing one frame
    - Instead of hardcode the pixels, change the input to a continuous signal fed by a stream FIFO
- VDMA IP Core is provided by Xilinx, modified the settings by Arlene. The driver for VDMA was borrowed from Wesley. Arlene added a Stream FIFO instead of Video Timing Controller IP in Wesley's design and design the structure as shown in Figure 13.

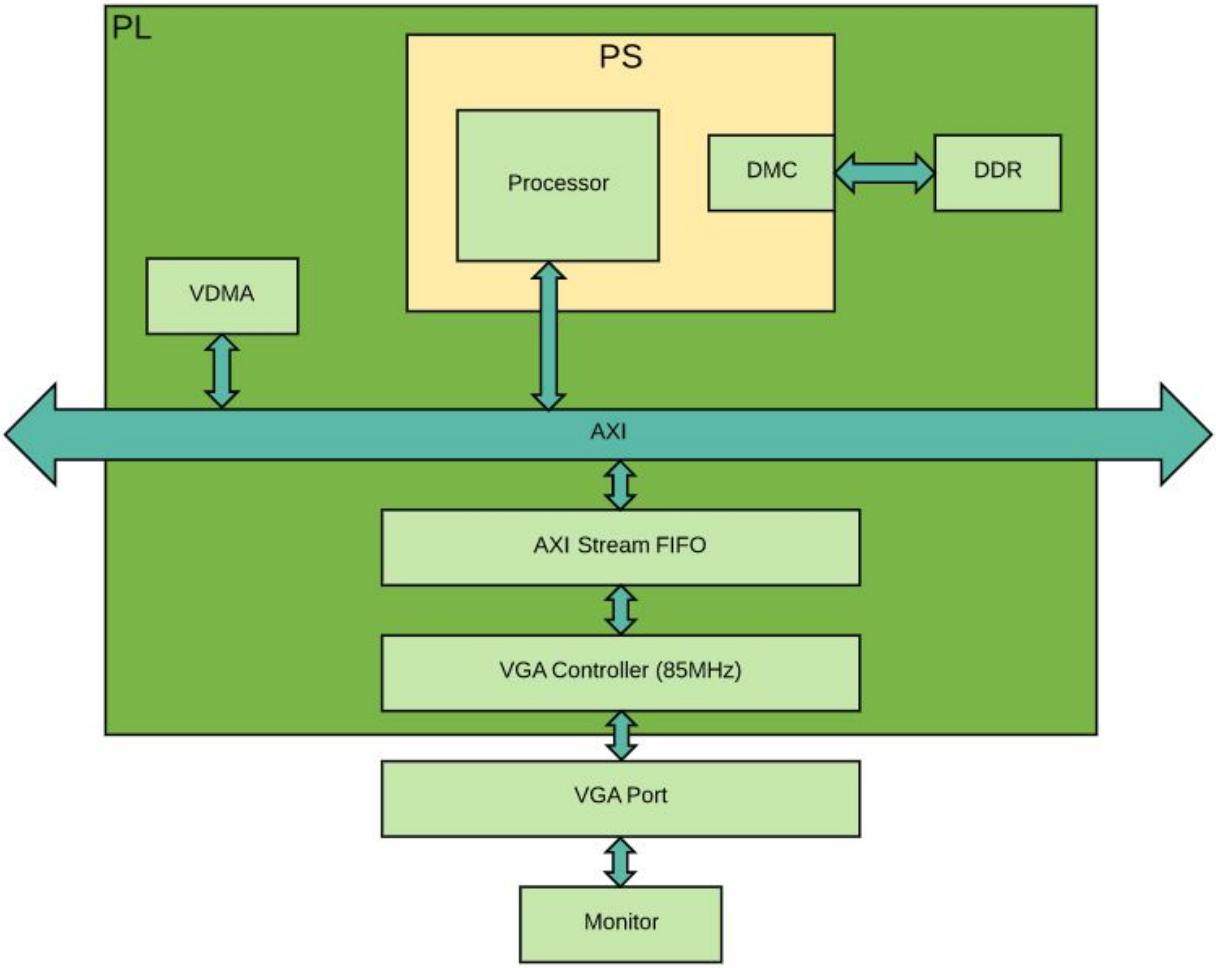

Figure 13. Block design for final VGA structure

- ADAU1761 Audio Core
- reference from https://github.com/ems-kl/zedboard_audio[2] by Sam, and re-customized the IP using the following settings:
    - Tone input (8 bits, GPIO memory)
    - Volume input (24 bits, GPIO memory)
    - In-line sound play to the jack
    - Communication protocol: I2C bus
    - Audio codec: ADAU1761
    - Clock frequency: 100 MHz

For more detail about the IP block, please refer to our individual report. [4][5]

The following are the customized setting we applied to Xilinx IP.

ZYNQ7 Processing System

-Xilinx IP block with with version 5.5

-re-customize IP using the following settings:

-PS-PL Configuration: check HP Slave AXI Interface->S AXI HP0 Interface

-MIO Configuration->I/O Peripherals:

-check ENET0, USB0, SD0, UART1, I2C0

-I2C0 IO: MIO 50.. 51

-SD0: check CD and WP, set IO to MIO47 and MIO46 respectively

-Disable all the Pullup under SD0/Power

-GPIO: enable GPIO MIO

-Clock Configuration->PL Fabric Clocks:

-FCLK_CLK0: Requested Frequency: 100MHz

-FCLK_CLK1: Requested Frequency: 85.86MHz

-Interrupts->PL-PS Interrupt Ports:

-check IRQ_F2P[15:0]

AXI Video Direct Memory Access (VDMA)

-Xilinx IP block with with version 6.3

-re-customize IP using the following settings:

-disable write channel

-change GenLock Mode under Write Channel Options to Master

-change Fsync Options under Read Channel Options to mm2s_fsync

FIFO Generator (Stream FIFO)

-Xilinx IP block with with version 13.2

-re-customize IP using the following settings:

-under Basic tab:

-Interface Type: AXI Stream

-Clock Type AXI: Independent Clock

-under AXI4 Stream Ports tab:

-TDATA NUM BYTES: 4

-TUSER WIDTH:0

Processor System Reset

-Xilinx IP block with with version 5.0

-keep all the settings in default

Concat

-Xilinx IP block with with version 2.1

-keep all the settings in default

6. Description of Design Tree

- The package we sent contains the whole project
- The folder structure:

Project Directory

|-Project
|-final.srcs
|-final.sdk
|-final.xpr

To launch the whole project, just go to Project folder, open the .xpr project.
Launch SDK from Vivado.
In SDK, compile the FPGA first then go the Run Configuration, Compile the file in System Debugger mode.

7. Reference

[1] Florent, “Create a VGA controller for the ZedBoard,” [Online]. Available: http://blog.dev-flow.com/en/11-create-a-vga-controller-for-the-ZedBoard/ [Accessed 15 January 2018].

[2]ems-kl, “zedboard_audio” [Online]. Available: https://github.com/ems-kl/zedboard_audio. [Accessed 31 January 2018].

[3] Wesley Kendall, VDMA https://piazza.com/class/jbenuzkh1151xw?cid=9. [Accessed 31 January 2018].

[4] Individual report, Arlene Fu

[5] Individual report, Sam Wang