Breakout Game Remastered

Youfeng Chen (yc3999)
Angzi Xu (ax2157)
Wang Chen (wc2794)
Zheyuan Song (zs2527)
Overview

- Rebuild the classic Breakout game.
- Destroy all bricks with the ball to win the game.
- If the player failed to catch the ball for a total of three time in one stage, player will lose the game.
Top-level Architecture

Hardware
- ROM: *.mif files
- Objects’ shape
- Color info
- Sound info

Software
- Main.c
  - Game logic:
  - Controls
  - EVERYTHING
- Driver

Take command
Send out command

Avalon

Sound IP Core
- Sound
- Audio CODAC

Screen
- Vega visual elements

Controller.c
- Input Translate

USB

Speaker
HW Design - Graphics

Tiles and sprites

- Tiles: pre-made graphical materials - assign to certain locations on the screen in case of need.
- Sprites: place things in layers - ensure proper overlay.
HW Design - Graphics

Tile example:

- Letter G, which utilizes two colors: “11” and “00”.
- 00 means the first color (#000000) and 11 means the forth (#FF0000).
- Each tile can use up to 4 colors.
Sprite:

```verbatim
always_comb begin
    (VGA_R, VGA_G, VGA_B) = (8'h0, 8'h0, 8'h0);
    if (VGA_BLANK_n)
        if (circle) //Ball
            (VGA_R, VGA_G, VGA_B) = (8'hff, 8'hff, 8'hff);
        else if (paddle) //Pad
            (VGA_R, VGA_G, VGA_B) = (8'h0, 8'hff, 8'hff);
        else if (waste) //Gray needless area
            (VGA_R, VGA_G, VGA_B) = (8'h69, 8'h69, 8'h69);
    else if (((tile_x <= 27 && tile_y == 2) || //Corners + Top
                (tile_x == 0 || tile_x == 27) && tile_y == 3) || //Side
               (tile_x <= 16 && tile_x >= 12 && tile_y == 0) || //SCORE
               (tile_x <= 16 && tile_x >= 13 && tile_y == 1) || //Score Number
               (tile_x <= 27 && tile_x >= 23 && tile_y == 0) || //STAGE
               (tile_x == 27 && tile_y == 1) || //Stage Number
               ((tile_x == 1 || tile_x == 2) && tile_y == 29) || //HP Indicator
               (tile_y == 15 && tile_x <= 18 && tile_x >= 10) || //Win or Lose
               (tile_x >= 1 && tile_x <= 26 && tile_y >= 5 && tile_y <= 10) || //Bricks
           )
        (VGA_R, VGA_G, VGA_B) = color_output;
    else //Background
        (VGA_R, VGA_G, VGA_B) = (8'h0, 8'h0, 8'h0);
end
```

Upper elements will display on top of lower elements in case of conflict
HW Design - Graphics

Block Diagram:
HW Design - Sound

Block Diagram:
HW Design - Sound

- Audio_ppl_0 configuration:
HW Design - Sound

- Audio IP Core:
HW Design - Sound

- Audio and Video Config.
HW Design - Sound

- Final Qsys connections
SW Design - Input

Buttons’ Function of the controller

<table>
<thead>
<tr>
<th>Button</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left arrow</td>
<td>Move the paddle to the left</td>
</tr>
<tr>
<td>Right arrow</td>
<td>Move the paddle to the right</td>
</tr>
<tr>
<td>Start</td>
<td>When one round of the game ends, restart</td>
</tr>
<tr>
<td>A</td>
<td>Launch the ball from the paddle</td>
</tr>
<tr>
<td>X+Y</td>
<td>Implement cheating mode to quickly end the game</td>
</tr>
</tbody>
</table>

Data received for each key press:

```c
// left: 0 127 0 128 128 15
// right: 255 127 0 128 128 15
// up: 127 0 0 128 128 15
// down: 127 255 0 128 128 15
// A: 127 127 0 128 128 47
// restart: 127 127 0 128 128 15 9
// X + Y: 127 127 0 128 128 15 32
// X: 127 127 0 128 128 31
// Y: 127 127 0 128 128 143
```
SW Design: Game Logic

- Hit the "A" button to launch the pinball
  - Automatically enter Stage 2 after hitting all the bricks in stage 1
    - Use the cheat button "X+Y" to finish the game quickly
    - Hit all the bricks in Stage 2 to win, and the screen will show "CONGRATS!"
  - Lose all lives before all bricks are destroyed and the game is lost and over. The screen displays "GG!"
    - Lose a life when paddle doesn't catch the ball
    - Lose a life when paddle doesn't catch the ball
    - Lose last life when paddle doesn't catch the ball
  - After the game is over, press the "START" button to restart a new game
Initialization

\[
\text{ball}_h = 208; \text{ & } \text{ball}_v = 425; \text{data.x_pad} = 208;
\]

Assign brick’s data:

```c
// assign data
data.brick1 = convert2bin( brick_matrix[0], 0 );
data.brick2 = convert2bin( brick_matrix[1], 1 );
data.brick3 = convert2bin( brick_matrix[2], 2 );
data.brick4 = convert2bin( brick_matrix[3], 3 );
data.brick5 = convert2bin( brick_matrix[4], 4 );
data.brick6 = convert2bin( brick_matrix[5], 5 );
```

Movement logic

The moving vector of the ball is a compose of its horizontal and vertical velocity.
SW Design - Hit Logic

Wall:

- Top wall y coordinate: 53
- Right wall x coordinate: 411
- Left wall x coordinate: 5

1. Hits the right wall:
   - Horizontal movement: reversed
   - Vertical movement: unchanged

2. Hits the top wall:
   - Horizontal movement: unchanged
   - Vertical movement: reversed

3. Hits the left wall:
   - Horizontal movement: reversed
   - Vertical movement: unchanged
SW Design - Hit Logic

Paddle “hitbox”:
Rectangle zone on the paddle.
L = length of the paddle
H = radius of the ball

Brick “hitbox”:
Four rectangle zones at each side of a brick

Yellow areas as shown on the right
SW Design – Other information

I. Score
A four-digit "score" at the top. Break one brick = +10 points
Capable of handling the highest number of bricks possible
(6 rows * 13 bricks per row * 2 stages * 10 pts per brick = 1560 points maximum)

II. HP indicator
When the ordinate of the ball is greater than or equal to 475,
execute "game_hp -= 1;".
When game_hp = 0, the game ends.

III. Game stage number
Stage 1 is relatively easy: fewer bricks and a slow ball
Stage 2 is more difficult: more bricks with a faster ball

```c
// stage 1
int brick_matrix[6][13] = {
    {0,0,0,0,0,0,0,0,0,0,0,0,0},
    {0,1,1,1,0,0,1,0,1,0,1,0,0},
    {0,1,0,0,0,0,1,0,1,1,1,0,0},
    {0,1,0,0,0,0,0,1,0,1,0,0,0},
    {0,1,1,1,0,0,1,1,1,0,0,0,0},
    {0,0,0,0,0,0,0,0,0,0,0,0,0}
};

// stage 2 map
int stage2matrix[6][13] = {
    {1,0,1,0,1,0,0,1,0,1,0,1,0},
    {1,0,1,0,0,1,0,1,0,0,1,0,1},
    {1,0,1,0,1,0,0,1,0,1,0,1,0},
    {1,0,1,0,1,0,0,1,0,0,0,1,0},
    {1,0,1,0,1,0,0,1,0,0,0,1,0},
    {1,0,1,0,1,0,0,1,0,0,0,1,0}
};
```
SW Design - Win & Loss Logic

- Clear all balls in both stages → CONGRATS!
- Fail to catch the ball for three time in one stage → GG!

- Think the game is too hard? We made a “cheating mode” button for you with love ❤️
Thank you!