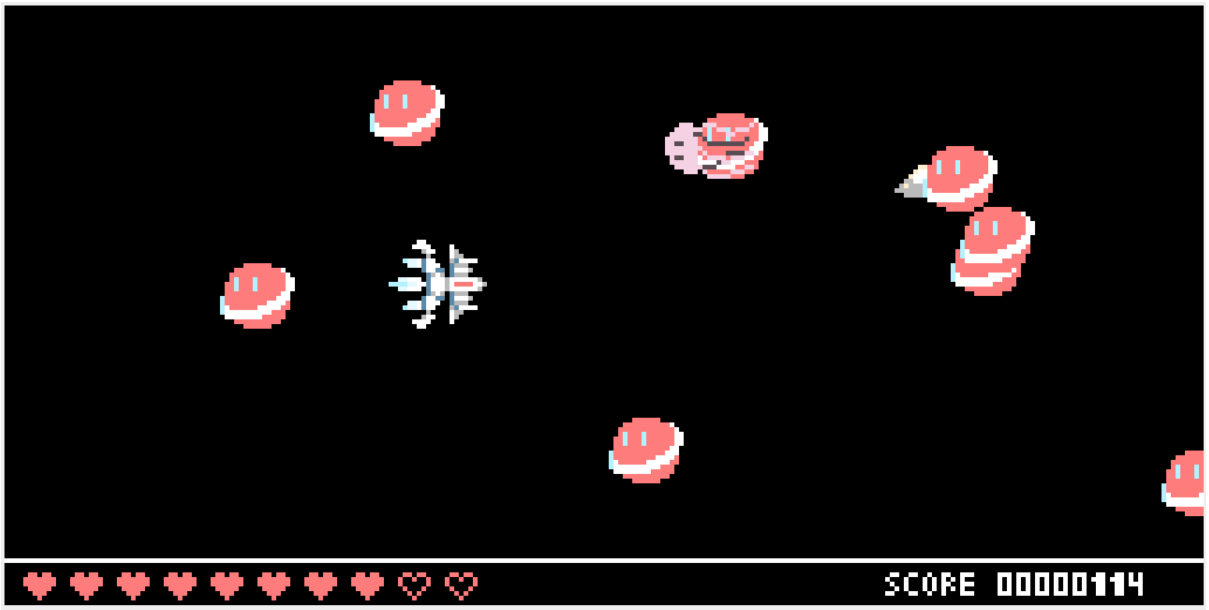
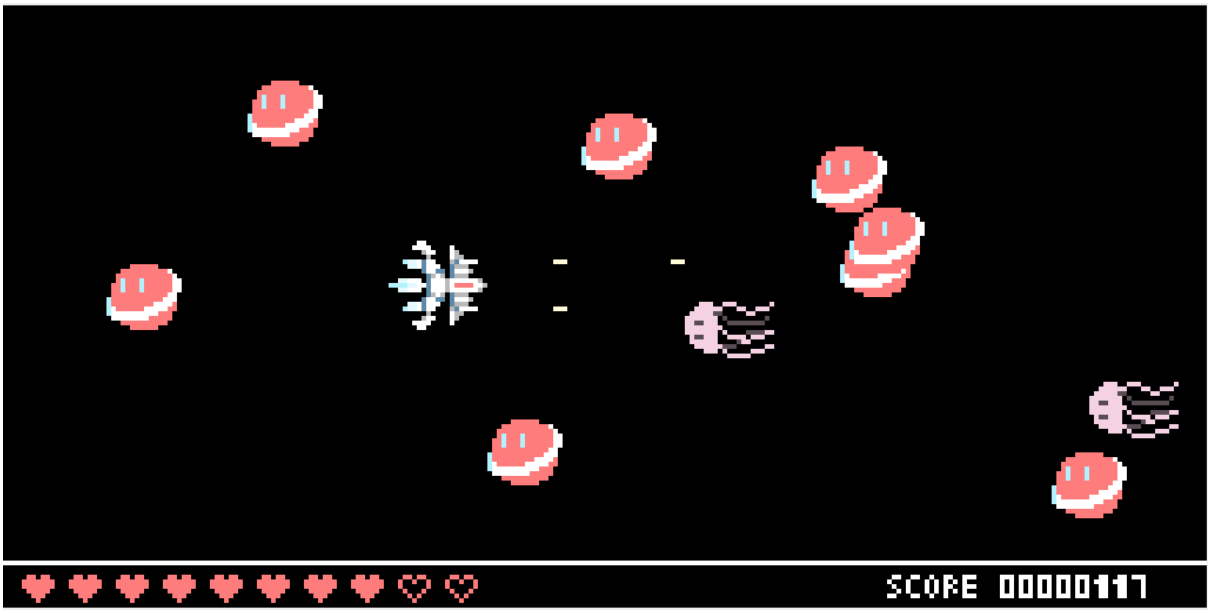
CSCB58 Individual Project - MIPS assembly shmups game

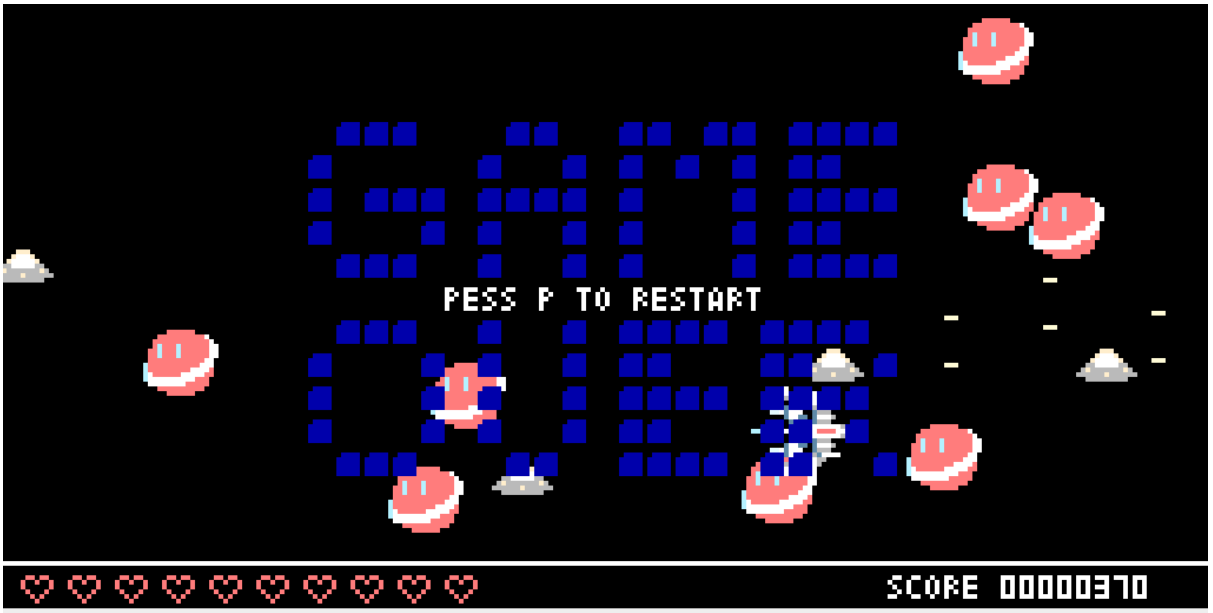
Jiale,Shang
1006580022
shangj22

Bitmap Display Configuration:

- Unit width in pixels: 2
- Unit height in pixels: 2
- Display width in pixels: 512 (256 units)
- Display height in pixels: 256 (128 units)
- Base Address for Display: 0x10010000 ($sp)




Text Display:

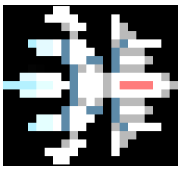
There are two kinds of text, big (for title and “game over” sign) and small (for score, infos like “press key to start”...).

Basic Function:

1. After the title and start animation, a “press any key to start” sign will show up at the center of the play area. the player can press any key to start the game.
2. player can press “p” to restart anytime in game or after game.
3. the player can see the current spaceship hp(hearts) at the left down side of the screen.
   The remaining health and maximum health of the spaceship will be presented in the form of hearts and heart containers.
4. the player can see the current score at the right down side of the screen.

5. The player can see a “gameover” sign when the spaceship is reduced to 0.

Player abilities:



1. move: player can press “w” to move up, press “a” to move left, press “s” to move down, press “d” to move left. The spaceship will not go out of the play area.
2. shoot: player can press “space” to shoot 2 bullets from the upper and lower weapon of the spaceship, after each shoot, the spaceship will have 25 shootcd.
3. take add-ons: will be explained later.

Obstacle abilities:

New obstacle appears from the right side of the screen when specific time passes or when an old obstacle moves out of the screen or is crashed.

obstacle1:

hp: 1

movecd:0

pattern: move 2 units leftward and 2 units upward or downward(screen rolling speed excluded), it will change its direction on y axis every time it reaches the top or bottom edge of the game area.

obstacle2:

hp:3

movecd:4

pattern: everytime movecd counts to 0, move 1 unit (screen rolling speed excluded)

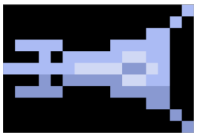
obstacle3:

hp:1

movecd:1

pattern: everytime movecd counts to 0, first move 2 units rightward and then move 4 units leftward (screen rolling speed excluded)

enemy(planned): a special obstacle that can shoot and follow player



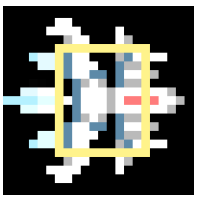
hp:3

movecd:1

shootcd:10

pattern: everytime movecd counts to 0, move 2 units toward player’s spaceship

Object collisions:

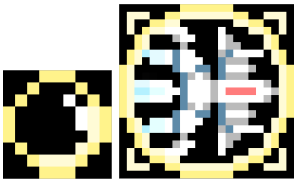


1. when an obstacle collapses with the spaceship hit box, remove this obstacle, add new obstacle, and reduce spaceship hp or remove the shield.
2. when an obstacle collapses with a bullet from the spaceship, reduce this obstacle's hp by buttet-damage, and remove the bullet from the game.
3. When a spaceship collapses with an add-on, pick-up the add-on and activate it.

Add-ons:

add-ons show up randomly on the screen. Note: when they appear, they won't overlap the spaceship.

1. shield: when picked up, the spaceship will get a shield or get its shield time +20. Shield disappears when shield time counts to 0. When having a shield, the spaceship can defend one damage, and then the shield brokes.



2. spaceship repair: when picked up, increase spaceship's hp by 2. The hp after repair won't exceed the max hp. 

Score:

Score increases when an obstacle is crashed or shooted, and when the spaceship is very close to some obstacles.

Other:

1. A lot of time is spent on pixel art design. This kind of classic shmups game art style is also a feature of this game.
2. start animation: after cleaning the screen, initializing everything and before the game starts, the spaceship moves slowly into the screen and stops at its starter position.
3. clear screen: when the first time the program is runned, it will clear the screen by writing all pixels black. And after that, everytime a player presses "p" to restart, it will only erase the drawings(spaceship,UI,obstacles,...) on the screen.
4.

Proposed methodology:

1. I will create a function to draw objects, taking an array of colors, an array of pixels' coords of each color, and an array of the number of pixels of each color as inputs.
2. To make obstacles appear randomly, I will use random number generators to decide which type of obstacle to be created, and what the x and y coord will be.
3. To make objects(obstacles/bullets,...) move, I will loop through the array of this kind of objects, get their current position to erase the previous ones, make changes to the x,y coord of each object by its status and movement pattern, and then draw the object at a new position.
4. In order to allow players to operate the ship, I will look at the keyboard input and write branches for different inputs, and change the next movement status of the spaceship accordingly.
5. For the shooting part, I will create a bullet array, and add new bullets when the player presses “space” and shootcd is 0. After a shoot, I will reset the shootcd.
6. To find collisions between the spaceship and the obstacles/add-on, I will loop through the obstacles array, do some math on the obst coord, spaceship coord, obst width,height and spaceship width, height to check whether there is a collision.
7. To find collisions between the obstacles and the bullets, I will loop through obst array and bullet array to check whether there is any bullet inside any obstacle’s hitbox.
8. In order to alleviate some "flickering" problems, the erasing process of some object will be skipped when it is not moving or changing(e.g. when the spaceship doesn’t receive a valid key event, and when an obstacle moved isn’t 0; only erase/draw changed heart and digit of score).
9. To display small text (score, infos), I will first create separate functions for drawing different letters at specific coord, and then combine them together.
10. To display big text (title, game over sign), I will first create functions to draw squares, and then create functions to draw letters with squares.
11. To implement the “add-on” feature, I will make “add-on” appear at random places on the screen without overlapping with the spaceship, by using the random number generator. And I will then check collisions between spaceship and “add-on”s (like what i did for checking collisions between spaceship and obstacles) when spaceship moves,
12. When updating the score, I will first compare the previous score with the current score digit by digit), and only redraw these changed digits.