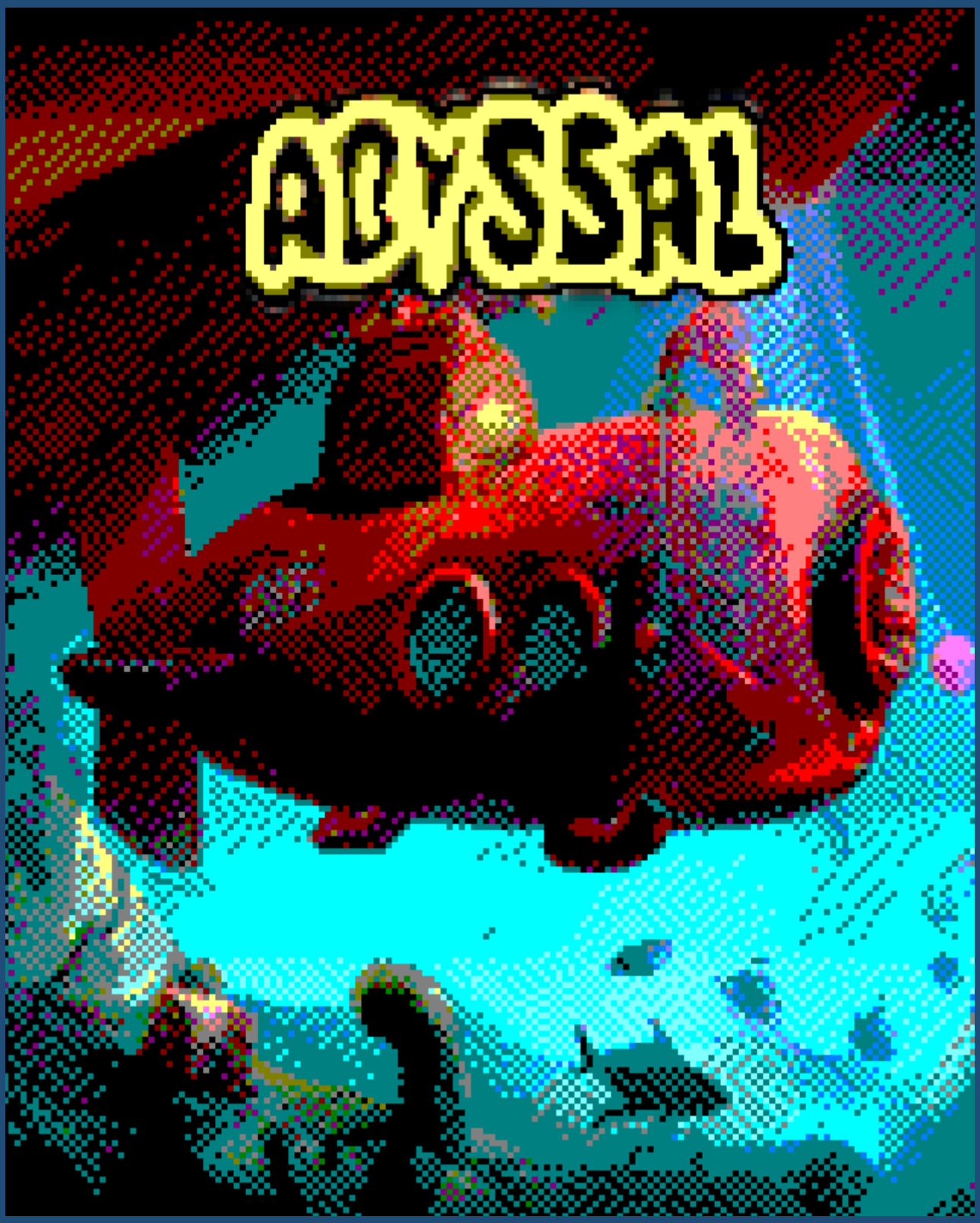


MAKING OF

A game developed by Cabbage Corp

INDEX

CONCEPT

The idea first came to us as we were thinking of different genres of games to develop. Since all we could come up with were just the generic, and nothing concrete, we opted to take a focus on the mechanics of the game, and that's when we though of the "energy" element. Instead of having a set number of lives like in other games of the shoot'em up genre, our game would have a health or energy bar, and each hit the player took decreased it. You could draw a relation between the amount of life lost in a hit to a life in any other game, but what our game does different is, instead of regaining full lives when going over a score threshold, the player can regenerate it partially by not shooting.

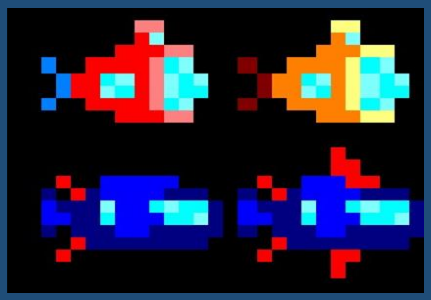
This forces the player to not be able to increase the score while they try to regenerate by not shooting down enemies, or allows them to space out the regeneration across level segments. Being more efficient when destroying enemies gives more time for the player to heal between waves.

As for the setting, since shoot'em ups are usually in space, we decided with an underwater setting, since it allowed the same range of movement.

ART AND MUSIC

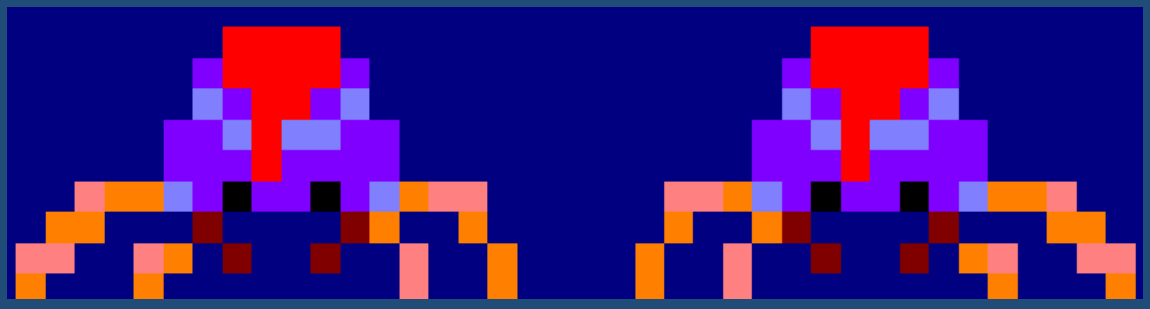
Since the setting we decided on was underwater, the player had to be a submarine, and the enemies sea creatures, with the occasional cyborg enemy.

These the initial designs we came up with for the player submarine:



In the end we decided to use the top left one for the game.

We didn't do the same thing for the enemies though. Since there were many different enemies, we just decided on what each one would be based on and did a single design. We did however cut one enemy from the game, the magma crab:



The magma crab was supposed to be an enemy that moved beneath the border of the playable area, shooting vertically instead of horizontally. It would be an enemy the player wouldn't be able to kill, and instead they would have to focus on dodging bullets from a different direction than usual. However border checking to destroy entities leaving the playable area didn't allow the magma crab to be on the HUD, and instead would have to be placed on the playable area. This however brought another problem, the enemy needed to "invade" the player area to be able to shoot the player, and since it was there it could collide and damage the player, however due to our collision optimisations, there is no collision checking between player bullets and enemies on the left half of the screen, which made the matchup not only unfair, but jarring, as it was the only enemy that bullet would go though on (aside from rubber rings and wheels, but those are just supposed to be obstacles, they don't shoot).

As for the music, we composed 6 melodies in total, one for each level, one for the main menu, another for the game over screen, and a final one for the victory screen. The melodies were composed first on a piano, and then trasposed in Arkos Tracker 2. However a compatibility problem forced us to redo every song on Arkos Tracker 1. We then had a small setback here when we discovered the predefined instruments Arkos Traker had, and had to adapt every song again using the instruments, however since the song was still the same, changing the instruments didn't take long.

AI

We didn't run into much trouble while developing the AI, or more precisely, the movement patterns for most of the enemies. The only two things that took a bit more work to get going were the circular motion of the first boss, and the bullet that follows the player.

During the initial tests of the circular movement, the enemy flew out of the screen. In the end, the problem was solved by defining "regions" in the enemy's path, and its speed would change when it reached certain points in the path, to end up making an ellipse-like movement.

As for the special bullet, well, it isn't really a bullet, not internally at least. Since the bullets don't have AI, we used an enemy instead, and added player-enemy collisions to the game, which weren't a thing yet at that point, so the bullet could hurt the player if the two of them collided.

The acceleration towards the player's vertical coordinate is done simply comparing the relative position of both, and adding to the speed of the bullet. Since this would increase the speed too quickly if done every frame, instead we use the time alive counter every enemy has for checks like firing rates and movement patterns, and use it to only increase the speed every 8 frames. Now, that we have more experience in development, we would have implemented the acceleration by using fixed point notation for the speed, but it's too late for these kind of changes at this point in development, and the current effect is good enough.

COLLISIONS

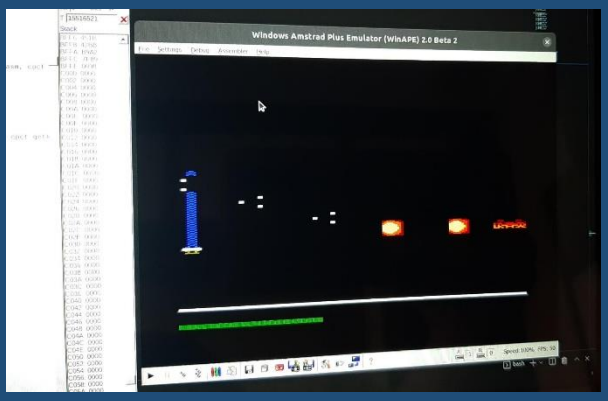
Since we wanted a big amount of bullets to be able to be on screen at once, we focused our optimisation for the game on the collision system, and doing so also required changing the core of the engine by changing how entities are stored in memory. At first every entity used the same space and was stored in the same array as every other entity. This meant that every time we a system needed to call a function on every entity of a specific type, despite not needing to call the function for the majority of the entities, the engine still had to iterate through them.

What we did was split the different types of entities in three arrays, one for enemies, one for enemy bullets, and one for bullets fired by the player. This way the engine wouldn't iterate through entities that didn't need to be called for a function that was applied to a different type of entity, and also made sure that when checking pairs for collisions, it would never even check if a pair of entities of the same type could collide with each other.

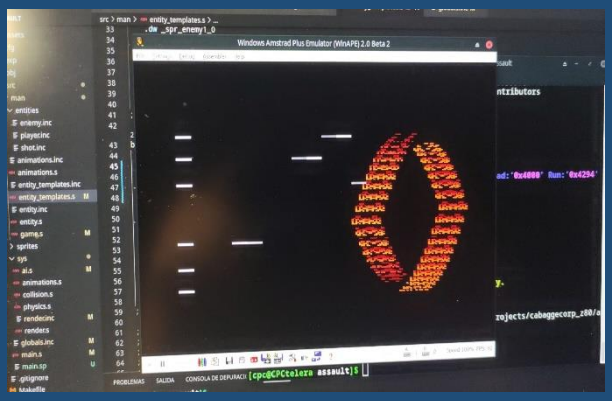
To optimise further, at the end of development we changed the function that check collision between enemy bullets and enemies to assume the bullets were points with no width or height. Since they're so small, not adding the height barely does a difference, and not including it isn't noticeable while playing. On the other hand, not adding the width makes the collisions more apparent, and it gives a better feedback on hitting the enemies.

DEVELOPMENT

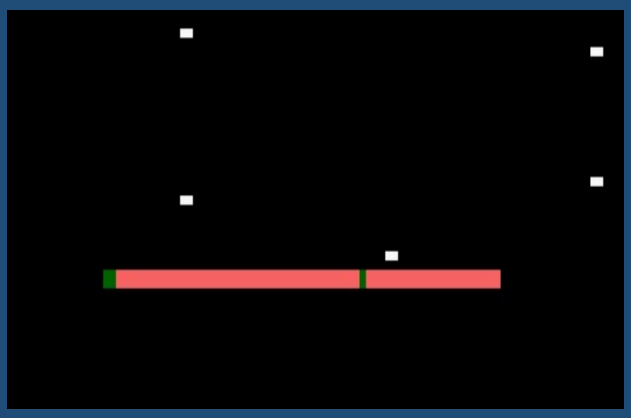
In this section you can see several pictures of the game in different stages of development:



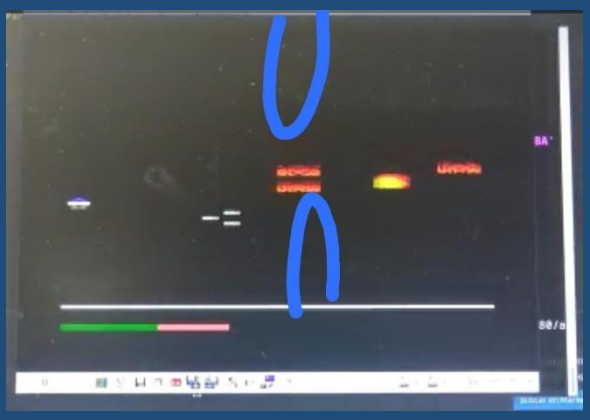
Testing different shot patterns



First success on making circular movement



A visual bug on the energy bar



A sketch of a boss fight's concept

TECHNOLOGY USED

To develop this game, we used the following resources:

- CPCTelera: Game engine for development on Z80 assembly
- WinAPE: Amstrad CPC emulator
- Retro Virtual Machine: Amstrad CPC and ZX Spectrum emulator
- Git: Version-control system
- GitHub: Internet hosting service for git
- Visual Studio Code: Text editor
- Arkos Tracker
- GIMP: Sprite editor

CREDITS

This game was made by

Cabbage Corp are:

- Alberto Rius Poveda
- Eliseo Fuentes Martínez
- Alfonso Ruiz Martínez