Cannibalator development notes:

The program is written fully in asm;
I've been a java / cpp programmer for 15 years and since 2013 I decided to get reasonably fluent in z80, which has been quite fun and accidentaly taught me a lot !

I had an initial project from 2013 that was a maze game with tile movement, and no double buffer or sound.
Some preparation was done to test double buffering and sound playback using the firmware.
Development was started in mid/late august , and done in 30 minutes/1 hour chuncks (because of full time job and family).

This is my first assembly game with "platform like" physics, and real coordinates instead of "tile" movement, so some time was spent to write the collision methods necessary at the beginning.

I didn't see coming the major difficulty: implementing a system to flag which part of the map were damaged to reblit only them.
As such it was implemented in a rush after mid september, and in the crunch, I introduced a bug that stopped me for quite some time.
After putting some real effort into finding the source of the memory corruption, I resumed in packaging the existing physics into a minimal game.

It has been an interesting phase of the project, as I couldn't add anything by risk of missing the deadline I had to think just in term of level design, and I think the result is reasonably fun !

Lessons learnt:
In asm, you can't rush it. You end up corrupting some registers, and it is not really obvious something is wrong at first. If you don't test enough and solution problems AS SOON as they arise, the backlog is very hard to entangle.
I would say when I was really in a rush to deliver, the lack of scoping of what I was writing was getting in the way all the time.

The game idea I initially had was multi screen and less hard, I will definitely come back to it and improve this foundation in the future.