GDD

Working title

```
Your game’s title should communicate the gameplay and the style of the game
```

"Gorodith"
Go(dot) Ro(bot) Di(ploma) Th(esis)

Concept statement

```
The game in a tweet: one or two sentences at most that say what the game is and why it’s fun.
```

A singleplayer third person platform, action, and puzzle game that mixes 2D and 3D gameplay. Main purpose of the game is to cover all areas of game development outlined in the curriculum, so that it is a good tool for students to experiment and learn with.

Genre

```
Single genre is clearer but often less interesting. Genre combinations can be risky. Beware of ‘tired’ genres.
```

Platformer (3D and 2D), Action, Puzzle, Metroidvania (during the gameplay players will gain new abilities, which will help them reach new areas)

Target Audience

```
Motivations and relevant interests; potentially age, gender, etc.; and the desired ESRB rating for the game.
```

The audience of the game should be fans of 3D platforming games and people who like experimental and indie games. The important thing is to create a easily modifiable source code of the game for the students to experiment with and fulfill tasks in. These students are mainly the Students of the bachelor GameDev program at CTU FEE.

Unique Selling Points

```
Critically important. What makes your game stand out? How is it different from all other games?
```

- Two unique 2D in 3D minigames/puzzles
- 3D platformer - not that saturated market
- Computer science theme
- Open-source learning tool

Player Experience and Game POV

```
Who is the player? What is the setting? What is the fantasy the game grants the player? What emotions do you want the player to feel? What keeps the player engaged for the duration of their play?
```

The player is a nano-robot, which is tasked to enter a computer and repair all corrupted parts. The setting is a stylized version of the inside of a computer.

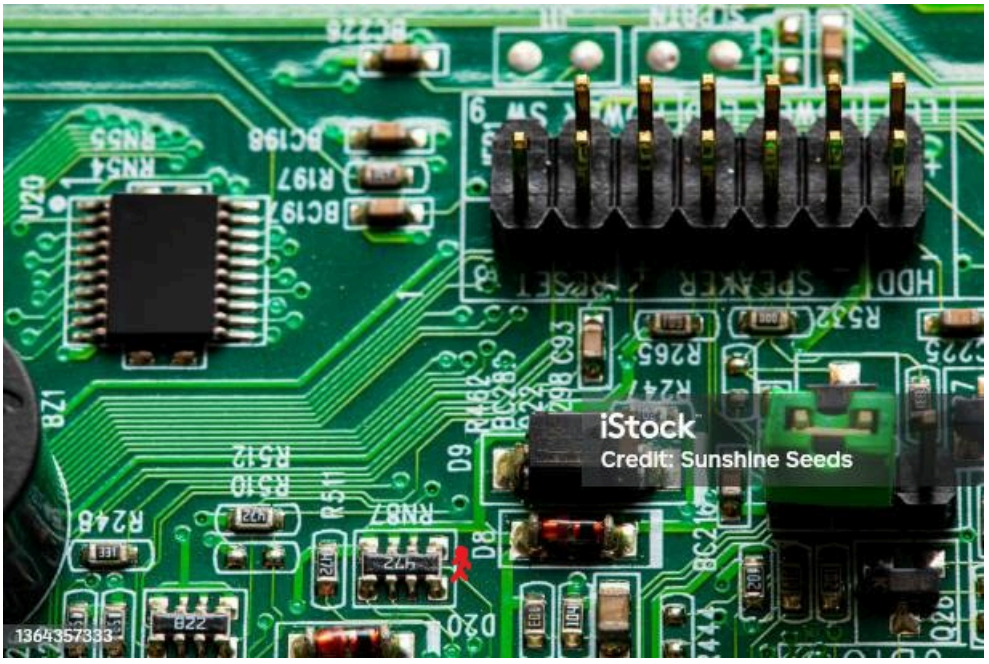
The fantasy of the player should be the wonder of seeing a very complex machine from the inside and being able to interact with its parts.
I want the player to feel a sense of scale and being a part of a complex interwoven system that does not take them into account.
The main driving force for the player should be the loop of completing a level. This will be done using fun and engaging platforming and puzzle solving mechanics.

Visual and Audio Style

```
What is the “look and feel” of the game? How does this support the desired player’s experience?
What concept art or reference art can you show to give the feel of the game?
```

Visual

The game takes place somewhere between the microscopic and macroscopic world inside of a computer. The main esthetic should be similar to the image below and image in Game World Fiction section with the player being quite small (stick figure near the bottom). One could image the pins in the top right as the size of a smaller block of flats from the players perspective.



Audio

Most of the audio will be done in an electric or synth theme to match the visual aesthetic.

Game World Fiction

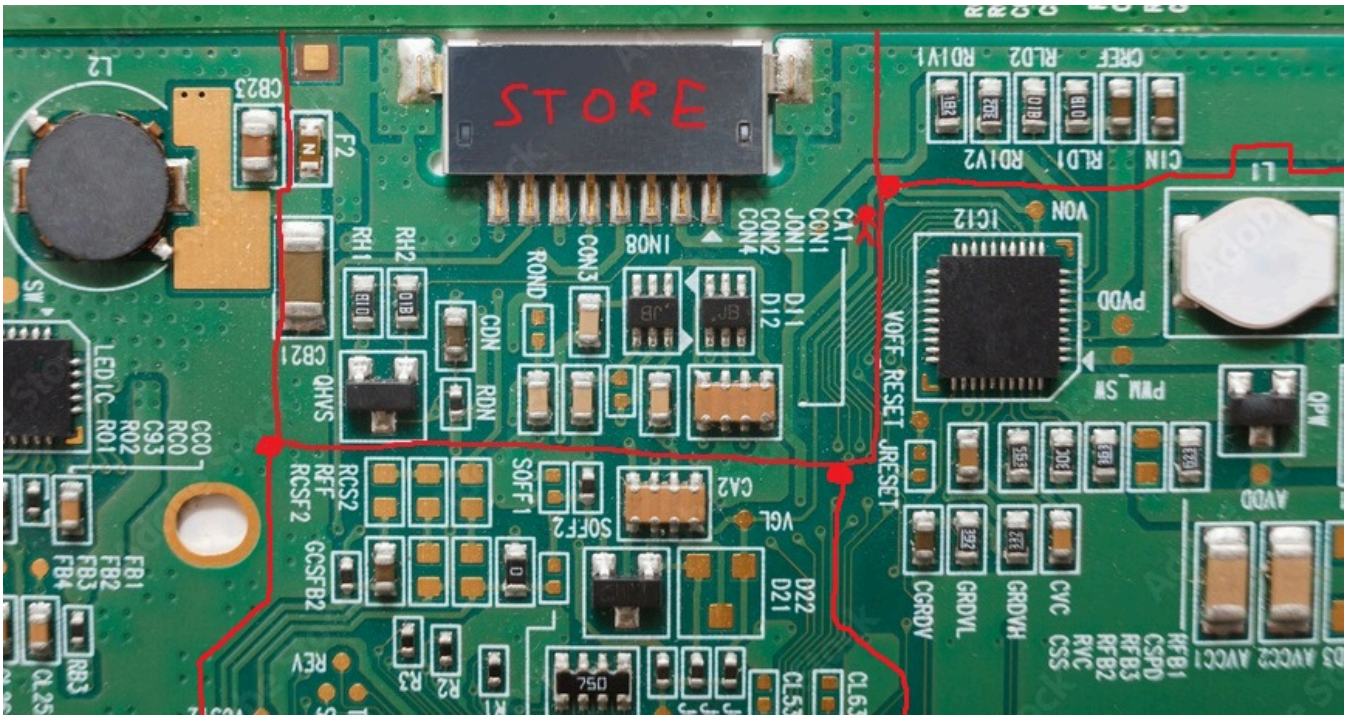
```
Briefly describe the game world and any narrative in player-relevant terms (as presented to the
player).
```

The world will be an artistic representation of what the inside of the computer looks like. The brief story will be about a nano-robot (the player) that has to go inside the computer to fight off a malware that has infected the computer. (or some sort of corruption/rust/etc.)

However as the player enters the computer, the infection will also get to them and disable their special power modules (Abilities in Objectives and progression section). These will be fixed one-by-one when a component has been cleared of the infection making the malware weak enough to gain control of one of their infected module back.

At the beginning they go through a cable, which will serve as a tutorial level for basic mechanics. Once complete they will get to the center (or the main part) of the motherboard, which will serve as the hub area (red area in the motherboard picture). This area will feature a "Store" of sorts. I imagine it could be a port (similar to the one in the

image), which can be used to send information about the malware to the outside in exchange for upgrades. The hub zone will feature buses (in computer terms, as in memory bus, system bus etc., red lines in the image), which will take the player to the specific components (CPU, GPU, RAM, etc.).



The majority of gameplay will take place in smaller self-contained levels. These levels will be present in the components of the computer (smaller hub areas). The levels themselves in the component could be represented as a section of the real hardware (eg. CPU - cores=levels). The player's goal is to clear each level of the corruption the malware caused and to gather more information about it.

(if there is enough time)
Once all parts of the computer have been fixed. The computer will be fixed and the game will end.

Monetization

```
How will the game make money? Premium purchase? F2P? How do you justify this within the design
```

No monetization or money making.

Technology

```
PC or mobile? Table or phone? 2D or 3D? Unity or Javascript?
```

PC only because the main point of the game is to modify the source code and learn with it. The engine in use will be Godot Engine. Most of the game will be in 3D with some 2D parts used as puzzles or as means of transportation (2D playable wall mural which after completion will help the player to reach some area).

Scope

```
How long to make, and how big a team? How long to first-playable? How long to complete the game?
Major risks?
```

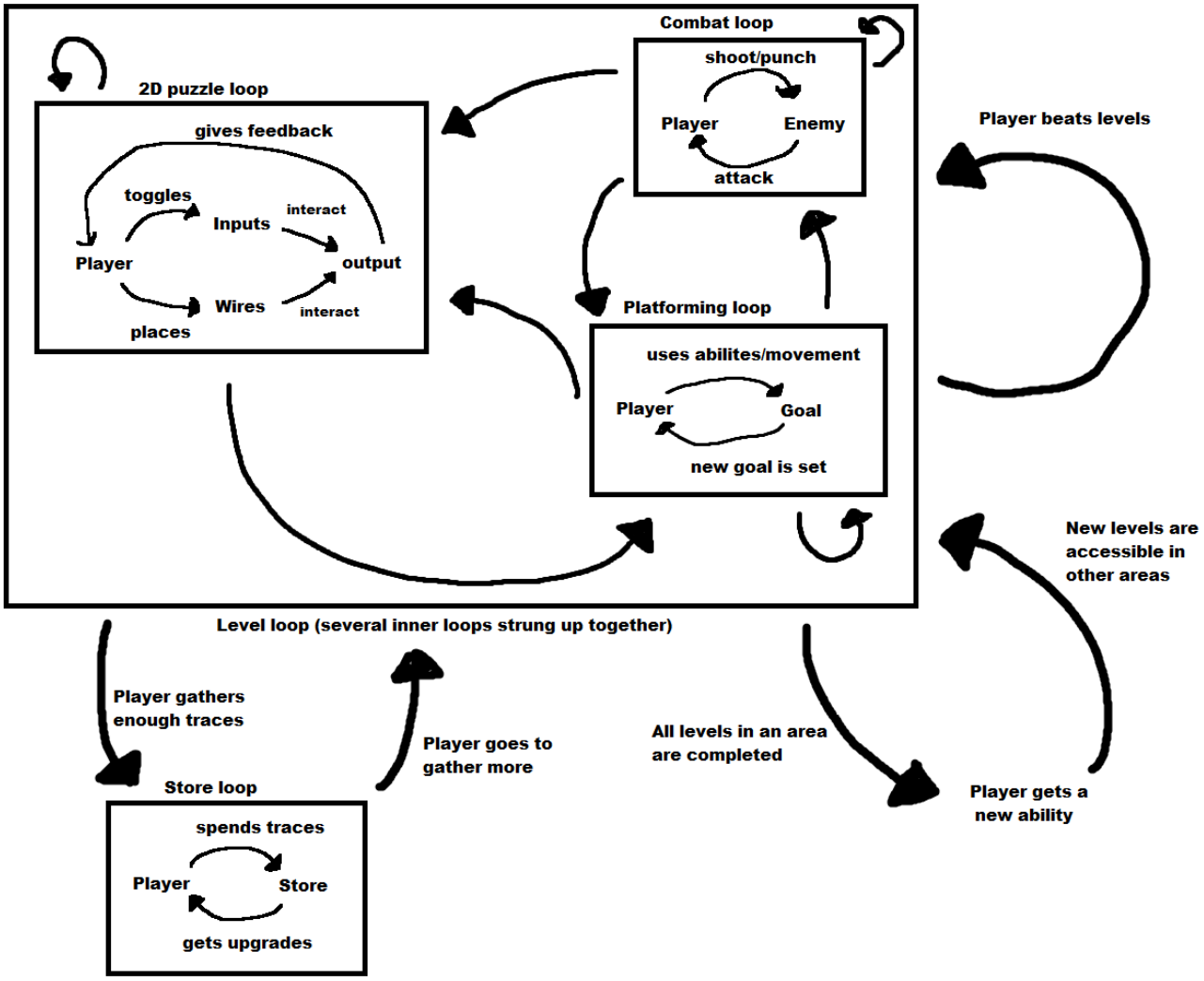
I am making this game and the accompanying course alone. I will have approximately 5 months to make a prototype of the game with a mostly complete course behind it. After that around 4 months to polish the game, finish it (mainly from the audio and visual style), and write the Diploma thesis around it.

I would like the game to feature at least the main hub area (motherboard) and one area of levels (eg. CPU). I think this would be enough for the learning part of the project while delivering a solid vertical slice of the game. The full project,

that will be a part of my diploma theses could be expanded with one more area if time allows.

Core Loops

```
How do game objects and the player's actions form loops? Why is this engaging? How does this support player goals? What emergent results do you expect/hope to see? If F2P, where are the monetization points?
```



Objectives and Progression

```
How does the player move through the game, literally and figuratively, from tutorial to end? What are their short-term and long-term goals (explicit or implicit)? How do these support the game concept, style, and player-fantasy?
```

Path from start to end

Tutorial -> Motherboard (hub area with store) -> Component (CPU, GPU, etc., smaller hub area) -> Individual levels -> all levels in component cleared -> new ability -> all levels cleared -> win

Short-term goals

Their short-term goal is to clear the level of corruption and collect traces of the malware source code (some of them cleverly hidden). A single level could be composed of several combat, platforming, and puzzle solving sections. These sections will sometimes be strung up one, after another in a linear fashion. Other times it will be a bit more open with the possibility to choose the order of the sections.

Long-term goals

The main objective of the player will be to fix the computer by clearing all the levels of the game. The player will achieve this by clear all their modules of the infection from the malware making them powerful enough to fight it and defeat it.

After completing around 2 levels the player should have enough traces of the malware in a form of collectibles (similar to coins in other games) for a stat upgrade in the store (more health, more damage, more speed etc.).

Abilities

Some levels (the last one in a component probably?) will grant the player a special ability, the use of which will help them reach areas and levels inaccessible before. These will be now accessible because the player cleared a large chunk of the infection, which weakened it enough to take control of their module again.

List of abilities:

- Double jump - the player will push out steam making them jump again in the air
- EMP Blast - the player will stop (even in midair) charge up and blast an EMP that destroys any robot enemies in the blast radius, will be used to break unstable walls to gain access to new areas
- Hologram - they will turn themselves into a hologram granting them the ability to pass through special walls.
- ... this should be enough for the scope

Game Systems

```
What systems are needed to make this game? Which ones are internal (simulation, etc.) and which does the player interact with?
```

Platforming

The player will be at the minimum able to move and jump. An easy addition would be a double jump, which could be a nice educative showcase. In an ideal case I would like to expand the platforming with some form of wall jumping, long jumping or other interesting movement mechanics.

Combat

The combat system will be very simple to show the basics to the students. The player will have 2 attacks. One will be ranged with the player shooting a projectile in the direction they are looking. The other one will be a melee attack (punch, kick, or something like that).

The player will have a basic health bar and an indicator of the amount of malware traces they have.

Enemies

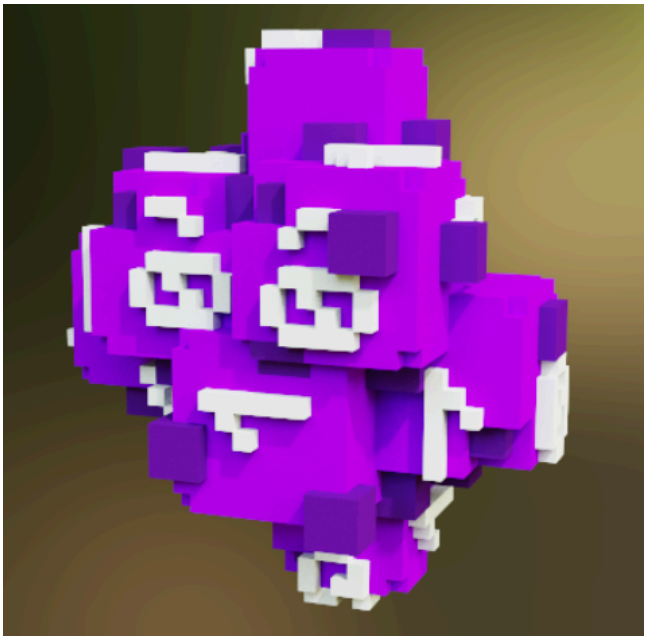
The game will feature 2 types of enemies. One ground unit, which will use the NavMesh system and one air unit, whose AI will be controlled with steering (like boids). This design choice is done to showcase both of these very common AI implementations (same as in the HRY unity course).

To spice up the gameplay one enemy will be immune to the shooting attack and one to the melee attack. I have not decided, which enemy will be immune to which projectile because this is something that needs to be tested in gameplay. (To be decided)

The enemies will drop a set amount of traces of malware source code. To motivate the player to destroy them.

Ground Enemy Prototype

Air Enemy Prototype



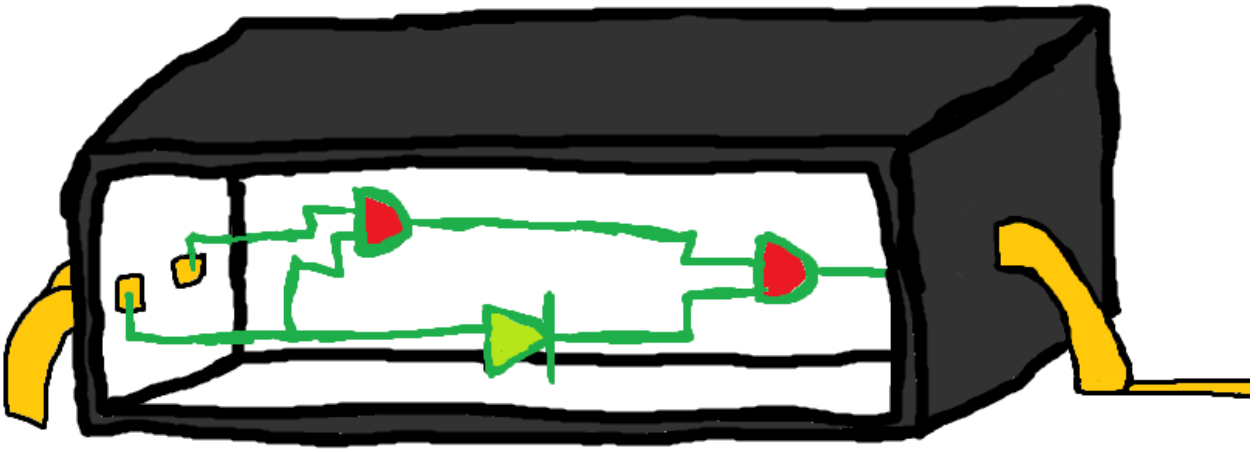
Puzzles and the 2D elements

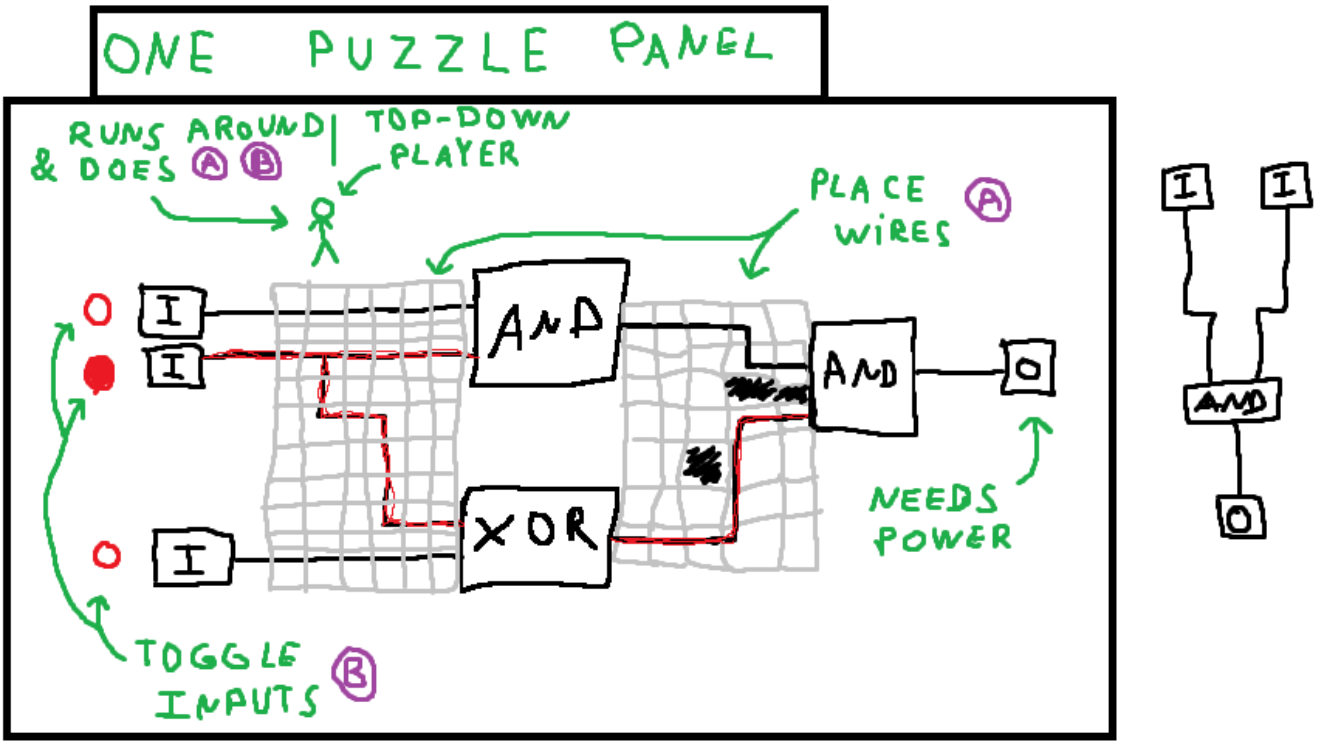
2D Puzzle

One possibility is to mix 2D gameplay and the puzzles. Special "transistors" (or the the black cockroach-like blocks) would host these puzzles and 2D sections. The side of them (or maybe top-side) would have a 2D image (viewport texture) with the puzzle panel.

Upon interacting with the transistor the player would be transported into the 2D world. The puzzle panel itself would control in a top-down manner. The 2D player would walk around the environment and have 2 interactions. The first one would be to switch the inputs on/off ("I" in image) and the second one would be to place wires to connect components (and, or, xor etc. gates) in desired spots. The goal would be to provide power to the output node ("O" in image).

I feel like the main problem would be that the showcase of 2D development would not be in-depth enough or too specialized. Another hard thing could be to make the puzzle design not too difficult while not being too boring.





2D platforming

This is and alternative solution to be used if the 2D implementation of the first possibility does not show enough complexity from the 2D development workflow.

The puzzle sections outlined above would be done in 3D or skipped all together. The 2D sections would be used for getting items or getting to otherwise inaccessible places. An example of what I mean can be seen in the picture below from the game "Ruffy and the Riverside".

(2D puzzle in a 3D world)

(Player in said 2D world platforming)

Sources

One-page GDD from GDC link
GDD template link
Ruffy and the riverside (3D platformer with 2D->3D) link