Yes, there was a **decompilation project** for *The Legend of Zelda: Ocarina of Time*, similar to the **Super Mario 64 decompilation**. The project was completed by a team of reverse engineers and preservationists, aiming to **recreate the original source code** from the compiled game.

---

1. Ocarina of Time Decompilation Project

- The project is known as **"The Ocarina of Time Decompilation Project"**.
- The team **reverse-engineered the game's compiled ROM** into C code.
- This allows **modders and game developers** to better understand how the game works.
- The project itself **does not contain Nintendo's assets** (textures, music, models) but provides a **fully readable and compilable version of the original code**.

🚨 **Legal Status:**
While decompilation itself is a **gray area**, Nintendo has **not yet taken legal action** against the project—probably because it does not distribute copyrighted assets (just code). However, if someone were to **use this code to create a full, stand-alone game**, that would likely **trigger a Nintendo takedown**.

---

2. Identifying Nintendo Rendering Techniques in Open-Source Projects

If you suspect an open-source project **borrows rendering techniques from Nintendo**, here's how you can check:

✅ A. Compare Function Names & Variables

- Decompilation projects often **retain function names** if they were recovered.

If an open-source project has functions like:
```
void Zelda_RenderShadows();
void Mario64_RasterizeTriangle();
```

- …it's a strong indicator they **borrowed from Nintendo's rendering pipeline**.

✅ B. Look for Unique Nintendo-Specific Graphics Code

Nintendo games, especially on **N64, GameCube, and Wii**, use **unique rendering techniques**:

- **Microcode Rendering** (custom RSP microcode on N64)
- **F3DEX2 Graphics Pipeline** (used in Ocarina of Time & Mario 64)
- **EFB (Embedded Frame Buffer) effects** (GameCube/Wii)
- **Depth Buffer Shadows & Fog Tricks** (Zelda games)

If an open-source project **uses these terms** or has **identical-looking graphics code**, it likely **borrowed from Nintendo**.

✅ **C. Check the Game's Memory Structure**

- Nintendo's games often have **unique memory layouts** (especially for handling textures, objects, and shaders).
- If an open-source project **matches Nintendo's memory layout**, it likely reverse-engineered the code.

✅ **D. Compare Assembly Code**

If you have access to **both the original game's disassembly** and the open-source project, you can **compare**:

1. **Disassemble the original ROM** using tools like **IDA Pro** or **Ghidra**.
2. **Compare the low-level assembly functions** with the open-source project.
3. If function calls or memory access patterns are **too similar**, it's likely **borrowed from Nintendo**.

✅ **E. Look at the Code's Development History**

- Check the **GitHub commit history** of the project.
- If early commits **contain Nintendo's function names or references to original ROM files**, it's a red flag.
- Some decompiled projects rename Nintendo's functions to **avoid legal issues**, but old commits might still have traces.

---

3. How Would Nintendo Detect This?

Nintendo uses **legal teams and automated tracking** to detect unauthorized use of its rendering techniques.

- **They monitor GitHub, game mods, and ROM hacking forums**.
- **They use pattern recognition** to identify Nintendo's rendering methods.
- **They take down projects** that closely mimic their games (even if no original assets are used).

🚨 *Example: When a team made a PC port of Mario 64 using decompiled code, Nintendo sent legal takedowns even though no official assets were included.*

---

4. Can Someone Use Nintendo's Rendering Techniques Without Legal Issues?

✅ **Yes, if they write their own engine from scratch and only use similar ideas.**
✅ **Yes, if they develop a retro-style engine inspired by Nintendo without copying code.**
❌ **No, if they directly copy code from decompiled Nintendo projects or official ROMs.**
❌ **No, if they distribute commercial projects using reverse-engineered Nintendo code.**

Final Thoughts: Should You Be Concerned?

If you're making a **game engine or project** and you want to avoid legal issues:

- **Do NOT use Nintendo's decompiled code** (even if it's open-source).
- **Do NOT copy rendering techniques line-by-line**—instead, study them and implement your own version.
- **If using an open-source project, check its history** to see if it contains Nintendo-derived code.

If your goal is to **work with Nintendo**, it's best to **stay clear of anything legally risky** and focus on **creating original tech**.

Would you like help analyzing a specific project to see if it borrows from Nintendo? 🚀