NES Emulator

Sebestyén Bence

March 8, 2019

Contents

Chapter 1

Introduction

The Nintendo Entertainment System, frequently called NES, is an home gaming console developed by the Japanese company called Nitendo Company, Limited. The hardware itself can be divided up to five big different part.

The console main chip was manufactured by Richo, which contains the CPU (Central Processing Unit) and the APU (Audio Processing Unit) [2]. The processor itself is an 8-bit MOS Technology 6502 with a little difference that the decimal mode is not presented.

The PPU (Pixel Processing Unit), which was also shipped by Richo, is technically a primitive graphics card which is used by the system to colour and render the graphics pixel by pixel to the Television screens.

The Cartrdige which meant to provide the necessary binary code of the games and also the graphics data for the system. Also gave an opportunity to the developers to implement their own cartridge builds and use it to extend the console's capabilities one great example for that is the first The legend of Zelda game. The game's cartridge also contains a battery powered RAM extension for players to save their game state [1].

Two 8 button, these buttons are up, down, left, right, select, start, A, B, controller provided the interface for user input to the system. The NES controller was the first controller which introduced the single button plus symbol shaped DPAD. Each Nintendo system were brought some revolutionary design idea to the world of gaming console controllers.

The RAM (Random Access Memory) is the central piece of the hardware which not only holds data but trough memory mapping it also connect all the other pieces to the CPU. Therefore the developers can control the full hardware

behaviours trough with specific read and write operations to certain memory slots.

In this project these core parts of the hardware will be emulated on an x86-64 machines. As a result, provide an application which is able to run those games which were developed for the original NES system.

1.1 History

This console was the second console of which was created by the Nitnedo Co. Ltd. and the first which was planned to be sold world-wide. [5] The gaming console was first released in japan 1983. The Japanese version were called Famicon and had a bright red toy-like design unlike the version which was released world wide. The homeland release were followed by USA and parts of Europe in 1986 and later in 1987 Australia and rest of Europe.

However, these releases were concerning for Nintendo as '83 was also the year of the great video game crash happened in North America.[3]. But it turned out the perfect opportunity for the console. As the hugely saturated gaming console market shrunk down, most of the competitors get bankrupt, Nintendo rebranded their console and released it as Nintendo Entertainment System worldwide.

It was a huge success not just because the competitors fall out but also due to two important principle which Nintendo following since then. One of them is unlike other consoles before only certified games could be released to the platform, which meant degradation in quantity and increase in the quality of the games. The other principle was that the actual hardware built from not bleeding edge components therefore making it cheap, more easily accessible.

Chapter 2

Project Goal

Main target of the project is to build and application which is able to run games developed for the original Nintendo Entertainment System. The main target platform for the emulator to support is 64bit Linux, but the project design allows easy implementation for other platforms too. This goal is achieved by emulating all the necessary hardware of the original NES system programmatically thus creating a medium for the game binaries to run without any modification at all to it.
However, as the sounds system (APU) is not necessary to be able to run the games this part of the system is not implemented.
The project goal can be divided to multiple stages which are the core and other advanced stages. The main goal is to achieve the Minimal Core stage by the project deadline. However if the time allows it advanced features will be implemented.

Core - Minimal The emulator can emulate the perfectly the RAM and its mirroring an mapping behaviours properly.
The CPU capable to execute all the official operation codes, as the MOS 6502 has unofficial operation codes as well, also able to properly handle interrupts.
Graphics displayed on the screen with the usage of the PPU without any restriction about how it is doing it.
The user capable to interact with the system with keyboard.
Digitalized cartridge format (iNes) red and mapped with the basic NROM mapper (Mapper 0).

Core - Full) The graphics CPU and the whole system itself running with 100% cycle accuracy compared to the original gaming console. Some games are heavily rely on cycle accuracy, especially to provide some nice graphics implementations and also to provide more authentic experience for end users.

Advanced - Mappers Implementing additional Cartridge mappers other than the basic NROM mapper[4]. This would allow the emulator to run a larger pool of games.

Advanced - Multiple Platform Due to the nature of the project design, see below, support for other systems could be more easily implemented due to the modular approach which was used to develop the emulator. Therefore for instance Windows and Android could be easily supported by the emulator which would further boost the user experience.

Advanced - State Save Allowing the user to make a snapshot of the current CPU PPU and RAM state and be able to load it back at later point in case if the user would like to continue the game right from the place where they saved it.

Advanced - Online Multiplayer The NES system supports two-player local multiplayer, as the system provides hardware to connect two controllers, by default which could be extended over network therefore players could play two player games with the comfort of their homes.

Chapter 3

Project Execution

3.1 Design

Due to the fact that the NES gaming console's hardware could easily be divided up to 5 distinct, loosely coupled module it was clear that the best way is to design the implementation around these modules:

1. Ram
2. CPU
3. PPU
4. Controller
5. Cartridge / Mapper

This thanks to the fact that the system gives control over all of these pieces by wiring them in to given memory slots and using CPU instructions to perform read and write instructions on them to trigger given actions on them, all of this without letting the actual CPU Hardware know that it is accessing anything else. Therefore when the system could been implemented module by module in a careful order. Due to this modular separations the it was easy and straightforward to build each module easily by separating them up to smaller task and gradually build each module.

Besides the separation of the emulator code to modules, the GUI, Graphical User Interface, is also separated from the actual emulator logic. This behaviour is implemented by compiling the NES emu logic into a shared objects, dynamic library for Windows users, and linked it with the GUI implementation. This decoupling has numerous advantages over shipping the whole application as a single executable.
Due to this separation the emulator logic became more easy to test on its own. The when a new patch is ready for the applications only part of the project needs

to be recompiled and updated. Which makes easyer to implement a versioning and updating system.
The GUI implementation can be changed without affecting any of the core solution therefore it is easier to port it to different platforms and devices.

3.2 Tools

3.3 Management

Chapter 4

Specification and implementation

4.1 RAM

4.1.1 Specification

4.1.2 Implementation

4.1.3 Testing

4.2 Cartridge (ROM)

4.2.1 Specification

4.2.2 Implementation

4.2.3 Testing

4.3 CPU

4.3.1 Specification

4.3.2 Implementation

4.3.3 Testing

4.4 PPU

4.4.1 Specification

4.4.2 Implementation

4.4.3 Testing

Chapter 5

Evaluation

5.1 Performance and Precision

5.2 Game Performance

Chapter 6

Conclusions

Bibliography

[1] Jeff Gerstmann. *The Legend of Zelda Review*. Nov. 22, 2006. URL: https://www.gamespot.com/reviews/the-legend-of-zelda-review/1900-6162256/.

[2] NesDev. *CPU - Nesdev wiki*. Mar. 7, 2019. URL: https://wiki.nesdev.com/w/index.php/CPU.

[3] Nadia Oxford. *TEN FACTS ABOUT THE GREAT VIDEO GAME CRASH OF '83*. Sept. 21, 2011. URL: https://uk.ign.com/articles/2011/09/21/ten-facts-about-the-great-video-game-crash-of-83.

[4] Mapper - Nesdev wiki. *TEN FACTS ABOUT THE GREAT VIDEO GAME CRASH OF '83*. Mar. 7, 2019. URL: https://wiki.nesdev.com/w/index.php/Mapper.

[5] Wikipedia. *Nintendo Entertainment System - Wikipedia*. Mar. 7, 2019. URL: https://en.wikipedia.org/wiki/Nintendo_Entertainment_System.