SUPER NES 系统需求

文档编号：001

组长：李嘉睿

小组成员：李嘉睿

2021 年 5 月

目录

1.项目背景

1.1 项目背景

上世纪 8，90 年代，红白机风靡全球。超级马里奥，魂斗罗、吃豆人等游戏陪伴了一代人度过自己的青春时光。红白机的官方名称是 NES（Nintendo Entertainment System），由任天堂公司设计。自 1983 年上市到 2004 年正式停产，二十多年里该平台共推出了 800 多款游戏，总销量达到 6192 万台。然而随着硬件制造工艺、图形处理器性能的提升，现如今家用游戏机市场早已被微软的 X-Box，索尼的 PlayStation 以及任天堂的 Switch 等占据，红白机渐渐从历史舞台退出。然而抛开硬件平台不谈，从游戏设计者和玩家的角度，平台其实没有那么重要，游戏本质上是娱乐的工具，最重要的是游戏性和艺术性。现如今移动平台用户广泛，那么是否可以在其上实现 NES 系统的模拟器，让如今已是而立之人在闲暇之时重温青少年时期的快乐呢。本项目旨在为安卓移动平台开发一款 NES 模拟器，并移植一些经典的 NES 游戏。后续考虑提供 6502 汇编器、游戏交流社区等功能。

1.2 现状分析

目前安卓平台的游戏开发在图像渲染以及音视频处理方面一般通过 NDK（Native Developement Kit，原生开发套件）使用 C/C++语言。上层的 Java/Kotlin 代码通过 JNI 接口和 C/C++交互。这种方式具有更好的性能，但是开发难度较大，且需要音视频领域的相关知识。本项目旨在学习 Kotlin 语言和安卓平台的开发，为降低开发难度仍使用安卓平台封装好的 Canvas API 和 OpenGL ES 接口。

2.系统建设目标

本项目希望使用 Kotlin 语言在安卓平台实现一个本地 NES 模拟器。最终目标是支持用户游戏收藏、试玩功能。

3.需求分析

3.1 系统模块划分

本系统从整体上划分为三大模块：模拟器模块、视图模块、游戏框架模块。其中模拟器模块细分为：

- CPU 模块
- PPU 模块
- 总线模块
- 只读游戏卡带（ROM Cartridges）模块
- 游戏手柄模块
- APU 模块（待定）

游戏框架模块细分为：

- 应用程序和窗体处理模块
- 输入模块：处理用户和外设输入
- 文件 IO 模块
- 图形模块：处理图像渲染
- 声音模块（待定）
- 游戏顶层框架模块

视图模块分为：

- 进入页模块
- 游戏大厅模块
- 游戏介绍模块
- 游戏模块

- 搜索模块

  核心页面的原型图如下

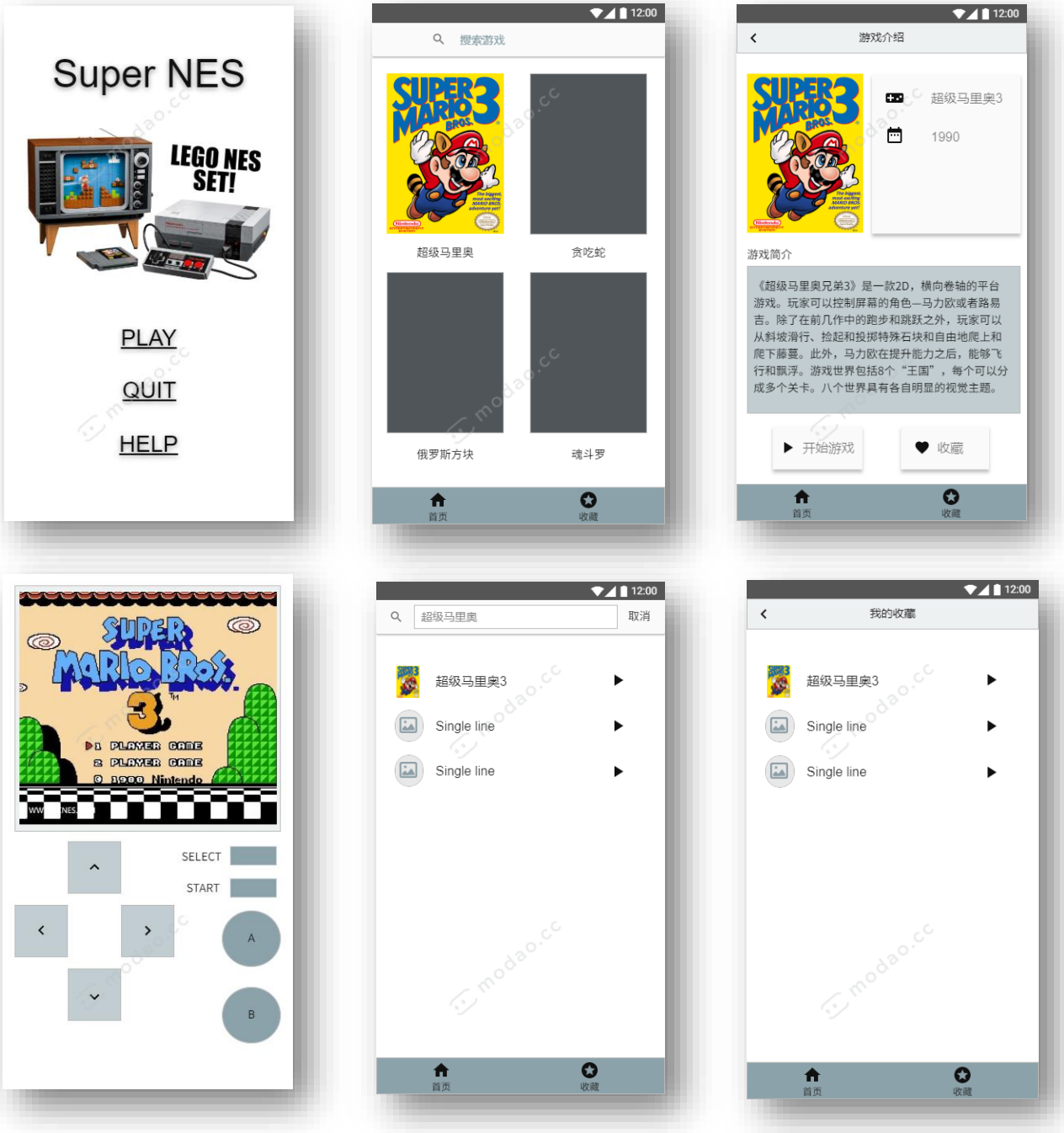

图 1 系统原型图

3.2 需求具体分析

3.2.1 模拟器模块

对模拟器模块，CPU 应当支持累加器架构的 6502 指令集以及扩展的 2A03 指令集，64kB 地址空间，总线应当包括地址、数据和控制总线，游戏卡带应当支持模拟 PRG 和 CHR ROM 卡带，PPU 应当支持中断生成、CHR 卡带的解析、图形图像的渲染以及游戏窗口的滚动，游戏手柄应当支持 joypads 手柄（包括上下左右四个方向、SELECT、START 以及 A、B 键）。

3.2.2 游戏框架模块

对游戏框架模块，应用程序和窗口管理部分负责窗口创建、活动的暂停和启动，输入部分负责记录用户输入事件（触摸、键盘输入、外设），文件 IO 部分负责文件读取，图像部分负责图像渲染，声音部分（待定）负责声音文件的装载和播放，顶层框架模块负责提供游戏开发的 API。

3.2.3 视图模块

对视图模块，至少应当包括应用程序进入页面、游戏大厅、游戏介绍页面、游戏页面、游戏搜索页面五个部分，具体见原型图。

3.3 关键业务流程

由于本项目重点是模拟器的实现，核心业务是游戏的选择和试玩。对应的流程如下：

1) 用户通过进入页面的 PLAY 按钮进入首页
2) 用户进入游戏大厅页面可以浏览游戏并选择，选择方式有下面几种
    a) 选择游戏
    b) 通过游戏搜索框搜索后选择游戏
    c) 进入收藏页面后选择游戏

3) 进入游戏介绍页面后，在试玩前用户还可以执行下面的操作

a) 用户点击收藏按钮，收藏游戏

b) 返回游戏大厅页面，重新执行从 2）流程开始执行

4) 用户点击开始游戏按钮开始游戏

流程图如下

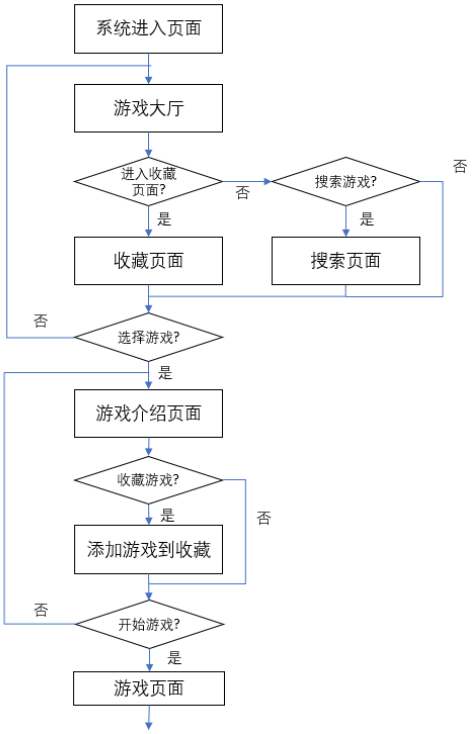

图 2 游戏试玩流程图

3.4 其他需求

3.4.1 开发和部署工具需求

本项目仅使用 Kotlin 语言进行开发。游戏框架模块的实现，参考《Beginning Android Games》中 Java 版本的游戏框架使用 Kotlin 语言重写，底层借助 Canvas API 以及 OpenGL ES。该书中的代码遵循 Apache 2.0 协议，无版权纠纷。

考虑到实际情况，拟采用的技术如表 1 所示：（均无版权纠纷）

表 1 系统技术表

序号	名称	用途
1	OpenGL ES/Canvas API	渲染 2D 图像
2	《Beginning Android Games》游戏框架	游戏音视频处理接口
3	Kotlin	实现语言

系统拟采用的环境如表 2 所示：

表 2 系统环境表

用途	建议配置		
	硬件	操作系统	应用软件
安卓系统模拟器	Pixel2	Android 11.0	SUPER NES
数据存储	Pixel2	Android 11.0	Sqlite

3.4.2 系统性能要求

系统要求独立于帧速率的图像移动速度，即对于不同帧率（FPS）的系统，游戏窗体的移动速度应当相同。

3.4.3 文档规范要求

除本文文档外，还应当提供向系统添加新游戏的操作文档。对文档不做具体要求。但是接口中方法级注释需要至少包括下面方面：

- 方法名和类型
- REQUIRES：对参数的预期（precondition）
- ENSURES：对返回结果的保证（postcondition）

3.4 集成与外部接口需求

3.4.1 用户接口

用户主要通过屏幕点击的方式和模拟器交互。

3.4.2 数据接口

本系统中游戏元数据以及用户收藏记录使用本地 Sqlite 数据库存储。游戏元数据表如下：

表 3 游戏元数据表

字段名	类型	作用
name	字符串	游戏名
info	字符串	游戏介绍
year	整型	游戏发行年份
file_name	字符串	游戏文件名

用户游戏收藏记录表如下：

表 4 用户游戏收藏记录表

字段名	类型	作用
game_name	字符串	游戏名

此外游戏二进制文件通过文件存储在本地。

4.实施进度

表 5 实施进度表

任务	截止日期
实现 CPU 模拟器并测试	2021/5/4
实现 Android 游戏框架	2021/5/5
测试贪吃蛇游戏	2021/5/5
实现总线和游戏卡带模拟器	2021/5/9
实现 PPU 模拟器和游戏手柄模拟器	2021/5/16
实现 PPU 滚动并完善各个 Activity 视图	2021/5/23

5.总结

由于个人经验和水平有限，目前技术预研阶段还无法判断是否能实现游戏记录的保存和恢复功能，因此上述需求文档中并未涉及该功能。此外一些扩展的功能希望能在之后的迭代版本中尝试加入。主要考虑从两方面扩展：一是实现基于 6502 CPU 的汇编器，支持断点调试、可重定位编译最终达到用户可以自制游戏或修改现有游戏源文件的目的；另一方面是游戏社区的实现，提供游戏经验分享、游戏得分排行榜等功能。