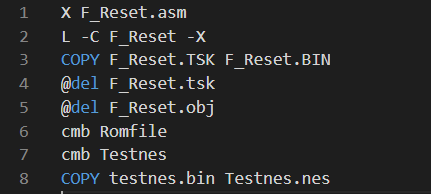
­FC游戏ROM的建立学习笔记

1. ROM建立

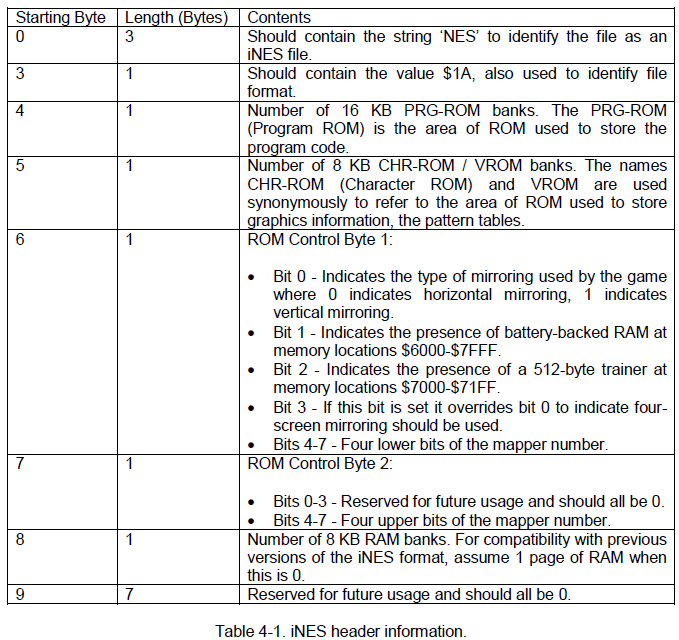
整个ROM建立可以用一个批处理命令完成 (nes.bat) 。

用X和 L 命令将F\_Reset.asm 文件编译，链接。然后用cmb 命令通过testnes.idx 文件

第6条，第7条语句疑似重复？？

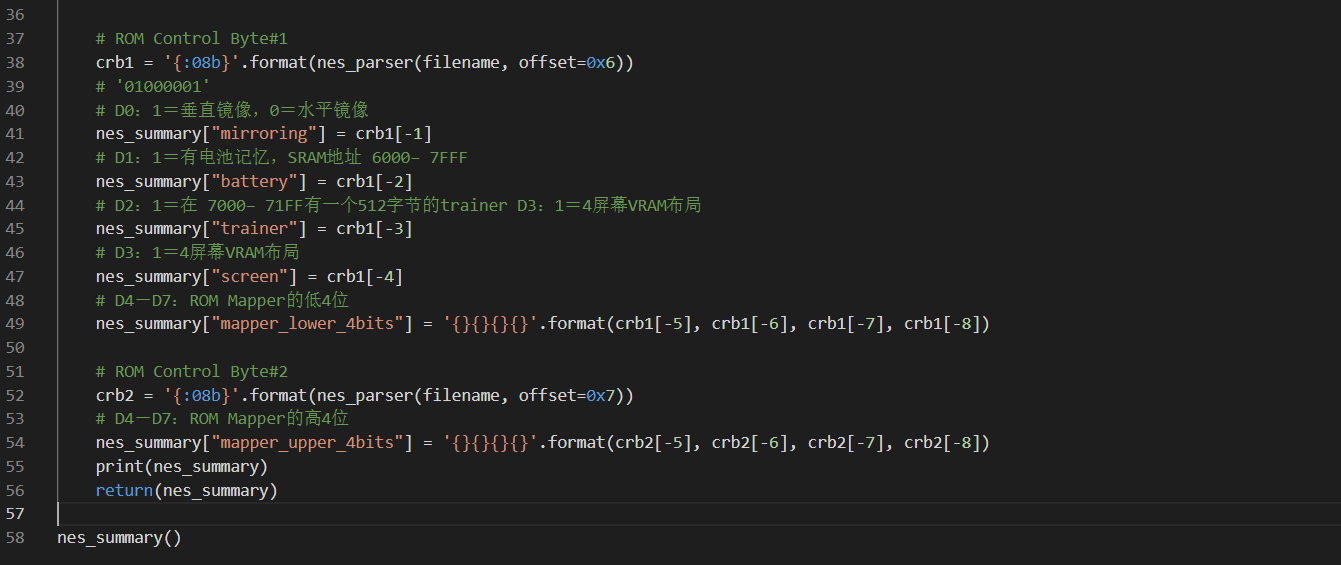
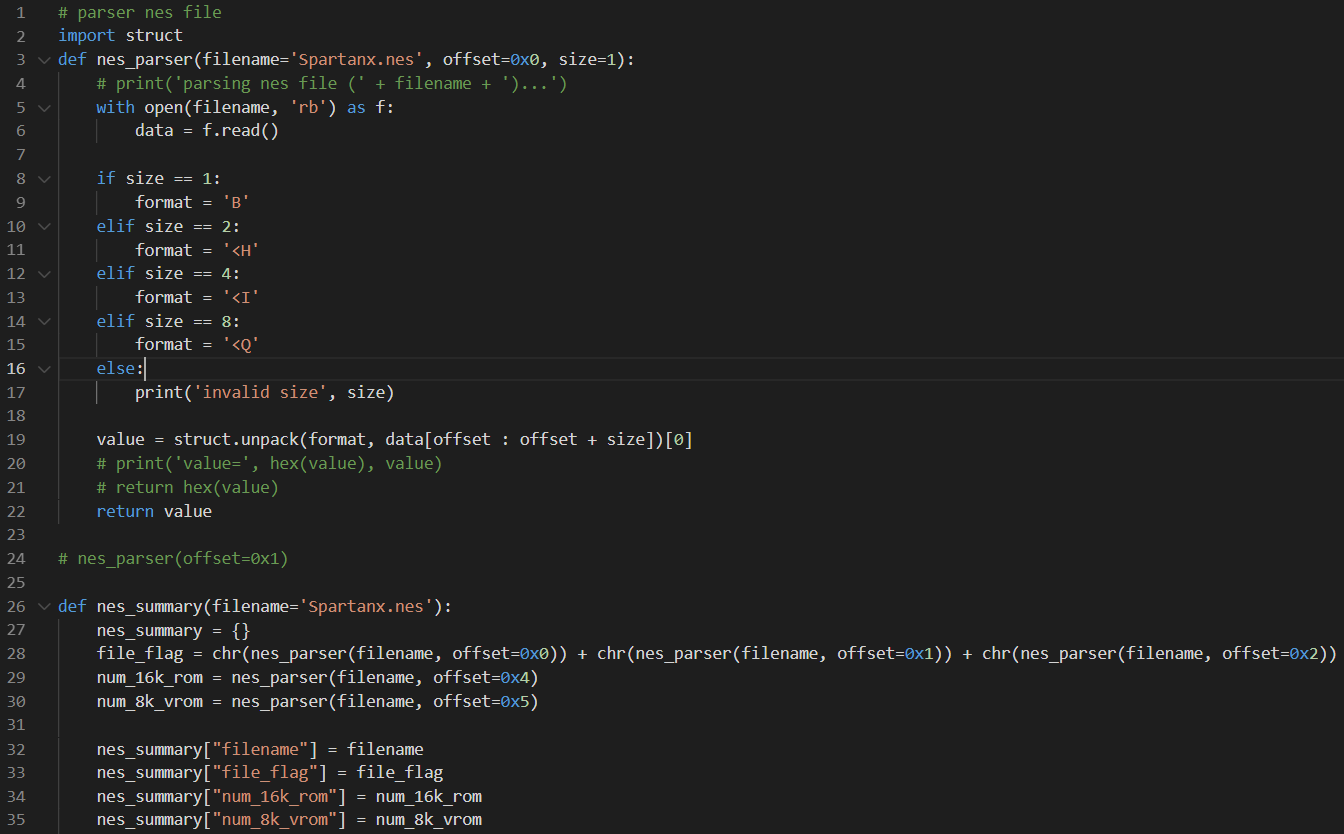
|  |  |  |
| --- | --- | --- |
| 序号 | 语句 | 解释 |
| 1. | X F\_Reset.asm | X.EXE 是6502系统汇编程序，用于编译\*.asm， 如果没有错误则输出.obj文件。 |
| 2. | L –C F\_Reset –X | L.EXE 是6502汇编程序的连接工具，常用参数为 -C 和 -X ，把编译工具生成的obj文件，转变成纯二进制文件 \*.tsk 。 |
| 3 | COPY F\_Reset.TSK F\_Reset.BIN | 将.tsk文件重命名为.bin文件 |
| 4 | Del F\_Reset.tsk | 删除.tsk中间临时文件 |
| 5 | Del F\_Reset.obj | 删除.obj中间临时文件 |
| 6 | - | - |
| 7 | Cmb Testnes | CMB.EXE 是二进制文件组合工具，从\*.idx文件中第一列读取需要组合的文件名字，第二列中读取组合时该文件的起点，从第三列中读取存放到组合文件中的位置，生成一个和\*.idx同名的文件，但文件的扩展名为bin，比如执行 CMB Testnes ，就是从Testnes.idx文件中读取组合信息，产生Testnes.bin的组合文件。 |
|  |  |  |

1. NES格式

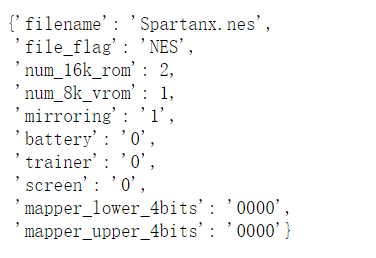


| **偏移** | **字节数** | **内容** |
| --- | --- | --- |
| 0－3 | 4 | 字符串“NES^Z”用来识别.NES文件 |
| 4 | 1 | 16kB ROM的数目 PRG-ROM page count |
| 5 | 1 | 8kB VROM的数目 CHR-ROM page count |
| 6 | 1 | ROM Control Byte #1 D0：1＝垂直镜像，0＝水平镜像 D1：1＝有电池记忆，SRAM地址6000−7FFF D2：1＝在7000−71FF有一个512字节的trainer  D3：1＝4屏幕VRAM布局 D4－D7：ROM Mapper的低4位 |
| 7 | 1 | ROM Control Byte #2 D0－D3：保留，必须是0 D4－D7：ROM Mapper的高4位 |
| 8－F | 8 | 保留，必须是0 |
| 16 - ? | 16KxM | ROM段升序排列，如果存在trainer，它的512字节摆在ROM段之前 |
| ? - EOF | 8KxN | VROM段, 升序排列 |

NES格式文件分析程序：



分析结果输出：



1. 核心文件

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 文件名 | 内容 | 代码 |
| 1. | F\_Reset.asm | 指定游戏的程序数据和图像数据在闪存中的位置、容量、镜像、游戏入口地址、是否需要预先加载特定数据等的汇编文件 |  |
| 2. | Testnes.idx | 产生二进制的组合信息文件 |  |

1. 核心文件中需要修改的部分

主要部分，这部分内容还需要进一步学习和理解。

3.1 F\_Reset.asm

3.2 Testnes.idx

四、自动化

4.1 总信息输入为EXCEL文件

4.2 建立代码模板

4.3 自动生成或完善核心文件（F\_Reset.asm, Testnes.idx）

五、测试

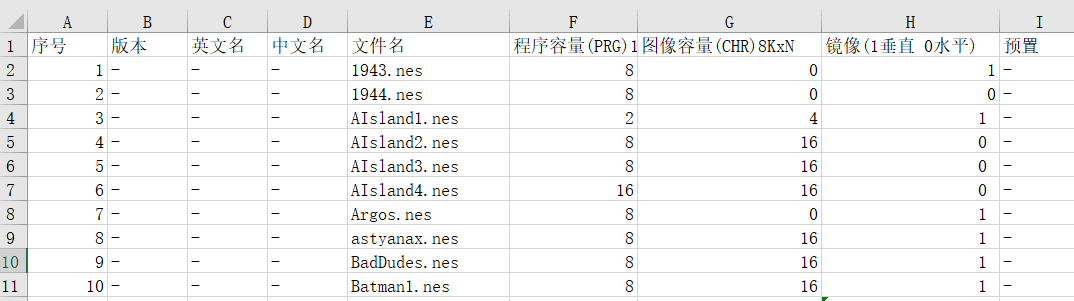
* 1. 基于FC模拟器的自动化测试工具

1. 疑问与沟通

6.1 建议以游戏列表EXCEL 文件为总信息源输入

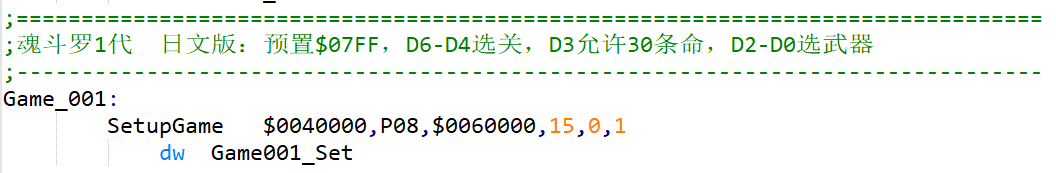
我的思路是：

* 以游戏列表EXCEL 文件（Nes-Games-List.xlsx）为总信息源输入，自动化脚本通过这个文件中的详细信息，自动生成全部或部分核心代码（F\_reset.asm, Testnes.idx）,这样即便用户需求变了，我们只需更新游戏列表文件，可极大提高效率。
* 游戏列表EXCEL 文件也是通过自动化脚本自动生成的。该脚本分析游戏库中.NES 文件，提取相应信息到EXCEL文件中。如下图：



* 但是目前仍有“版本”，“英文名”，“中文名”，“预设”字段的信息不知道这么获取。例如“英文名”如果能填充进去，自动化脚本就可以根据此字段生成F\_reset.asm 中关于游戏文件名部分的代码。

6.2 关于F\_reset.asm 代码中的游戏预置



上次文档中解释的已经很清楚了，但是仍有几个地方需再次确认（红色部分）。

上图中 $040000对应程序数据安放的绝对地址 （也就是.idx 文件中的第3列数据？）

P08对应8X16K, （P32对应32X6K, P16对应16X16K, P04对应4X16K, P02对应2X16K ?）

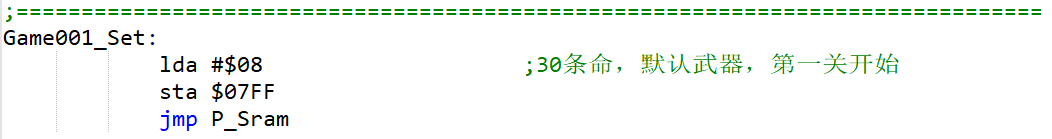
0060000对应图像数据安放的绝对地址 (这个数据怎么来的？安放的绝对地址是指在大文件中的绝地地址？)

16对应16X8K

0对应纵向镜像，

1对应需要预置数据（这个数据怎么来的？1表示有预置数据，0表示无预置数据？）

如果有预置数据则跳转对应语句，如下图：



Lda #$08

Sta $07FF

此两条语句含义是什么？（还需要进一步学习6502 ASM）08, 07FF 数据来源是什么？

七、文档记录

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | 初始版本 | 2021.03.15 | Kzeng |
|  |  |  |  |