**Y86-Simulator实验报告**

**一、后端：**

**实现框架\文件依赖关系：**

main.cpp

↓

header.h

↓

Fetch.h, Decode.h, Execute.h, Memory.h, Wback.h, Renewpc.h以及同名.cpp文件

↓

variables\_declaration.h, variables\_definition.cpp

↓

Register.h, Register.cpp

自底向上展开，各文件的主要功能：

**1.*Register.h, Register.cpp***

//定义Data的union结构，可视情况翻译为长整型或者字符数组（能正确翻译是因为.yo文件中为小端法存储，而longlong对位模式的解释也是小端），作为字符数组从imemory中读取或写入数据，作为长整型方便REG类变量计算或作为imemory索引寻址

union Data{

char ch[8];

long long ll;

};

//定义REG类，私有成员Data类型变量，分别有作为字符数组、长整型数据的读和写接口，既可以作为寄存器使用，也可以存储从imemory中读出的数据

class REG{

public:

REG(){data.ll=0;}

char\* get\_data\_ch();

long long get\_data\_ll();

bool write\_data(const char\*);

bool write\_data(long long);

REG& operator=(REG r);

private:

Data data;

};

//定义REGS类为寄存器文件，可获得寄存器变量，向寄存器写数据

class REGS{

public:

REG get\_reg(int);

bool write\_data(int, const char\*);

bool write\_data(int, REG&);

private:

REG reg[15];

};

**2.***variables\_definition.cpp*定义所有全局变量

*variables\_declaration.h*声明所有全局变量，其他文件只需包含此文件即可正常使用全局变量

char imemory[800000];//作为内存空间

REGS registers;//寄存器文件

REG pc, valC, valP, valA, valB, valE, valM, dstE, dstM, srcA, srcB;//pc寄存器，不同阶段产生的值

char icode, ifun, rA, rB;

int my\_stat = 1;//状态码:1为正常,2为HALT, 3为地址错误, 4为指令无效

char Cnd = 1;//是否选择分支

bool zf = 1, of = 0, sf = 0;//条件码

bool imem\_error, instr\_valid, dmem\_error;//用于在memory阶段产生状态码

**3.*Fetch.h, Decode.h, Execute.h, Memory.h, Wback.h, Renewpc.h*以及同名.cpp文件**

*Fetch:*从pc终读取icode, ifun, rA, rB(如果需要)、设置valP, valC(如果需要)

*Decode:* 确定srcA和srcB, 设置valA, valB

*Execute:* 获得执行操作数aluA, aluB, 进行相应的运算, 设置条件码(如果需要), 设置Cnd

*Memory:*获得读\写用的地址,数据、进行读或写、设置stat状态码

*Wback：*确定 dstE, dstM, 向相应寄存器写回

*Renewpc:* 更新pc

**4.*header.h***

整合包含了各个阶段的头文件

**5.*main.cpp***

int s16toi(const string&);//将字符串翻译成十六进制数作为imemory索引

void make\_imem1(const string&);//被imem2调用

void make\_imem2(istream&);//从标准输入流读取文件到全局imemory数组

void out\_mem();//十进制以一个byte为单位输出非零内存信息

void out\_reg();//输出寄存器信息

void out\_cc();//输出条件码

void run\_out();当状态码为1时运行六个阶段的函数并调用以上函数进行输出

注：将寄存器的值以char\*类型和long long类型组成的union表示参考了崔晨昊、祁昊学长ICS-Y86-master中Val的实现, 也因此对于小端法、课本上联合一节“联合不改变位模式只改变翻译方式“有了更为深入的理解。

**生成可执行文件：**

使用Mac在makefile所在文件夹下输入命令make会自动生成可执行文件。

或在vscode中使用release模式在release文件夹会生成可执行文件

**二、前端**

**前端使用方法：**

1.打开网页前先要在当前文件夹目录下输入“python（或者python3） connect.py 8000”打开本地服务器端口

2.打开网页，先在“选取文件”处选择test文件夹中的.yo文件，点击“next”按钮就会依次输出每条指令实行之后的register、memory等的对应值。

**前端实现思路：**

**语言**：html+css+js

**文件内容介绍**：

1.**web.html**文件

<head>部分主要是链接.css文件，调用jquery库

<body>部分主要是界面元素的排布，界面用到的所有元素都在这里进行类的定义和命名，方便之后在.css中美化和.js中定义函数。界面元素为我们要展示的内容，包括REGISTER、MEMORY、PC、STAT、CC和对应指令。

2.**style.css**文件

我们本次web页面设计基于小王子主题，设置了小王子背景、文字元素和美化设计。

首先，我们使用.title{}那部分的动画处理，实现标题和署名的彩色动画显示。

其次，我们将要显示的内容分为四个模块：regbox（显示十五个寄存器）、membox（显示MEM中不为0的内存值）、cmdbox（显示当前运行指令）和otherbox（显示PC、STAT、CC），分别进行了美化处理：

1. 为每个模块精心设计了边框、背景颜色以及字体，突出了STAT和OP即当前指令
2. 综合运用模块嵌套的方式，例如#regbox > div ,和overflow：auto、overflow-y:scroll等，对regbox和membox的内容设置了滑动条自适应，可以适应不同电脑屏幕，也可以在页面窗口缩放时由鼠标控制滑动显示；
3. 用active对regbox、otherbox、cmdbox里的显示值设计了光标点击可放大的功能。

接着，我们利用button设计了选择文件功能和next功能。点击“选择文件”按钮，可以选择.yo的测试文件，系统会据此读取对应的.json文件，点击“next”可实现分步运行。

最后，我们结合js内部的函数功能，实现了页面与用户的交互功能。

1. 进入页面，需要输入您的名字，页面提示“Welcome to B612!”；
2. 当STAT不为1时，程序终止运行，会提示一句话，背景由夜晚切换至白天；
3. 程序运行完毕后，再次选择文件，页面会初始化，点击“next”即可继续运行。

3.**script.js**文件

**filechange()函数：**当input选取的文件改变时会触发这个函数，并将界面设置到初始状态。首先通过document.getElementById获取用户选择的文件，利用.name操作获取文件名，之后用字符串的处理方法获得对应的json文件名，最后使用XMLHTTPRequest的API向服务器请求资源文件，将json文件内容保存在json全局变量中，方便在下一个函数中调用，调用Initial()函数将界面显示恢复到初始状态。

**button.onclick=function()函数：**点击next按钮触发，显示进行下一指令后的数值。首先从json变量中获取各个变量的值，然后在对应的位置显示，并将i+1。当STAT不为1的时候会调用EndCondition()函数切换背景。

**EndCondition()函数：**遇到stat不为1的时候切换背景，表示出错, 程序显示完毕。此时会弹窗小王子里的句子（与背景呼应），然后背景切换成白天状态。

**Initial()函数：**更换文件时触发。把背景改为晚上状态，不显示register、memory等的对应值。将i的值设为0。

**设计理念：**

把REGISTER块、MEMORY块、PC+STAT+CC块看作宇宙中的星球，夜晚降临，程序按部就班地执行，遇到STAT不为1或程序执行完，这表示夜晚终止，迎接白天。

“All men have the stars, but they are not the same things for different people.”给本次PJ留一点小浪漫～

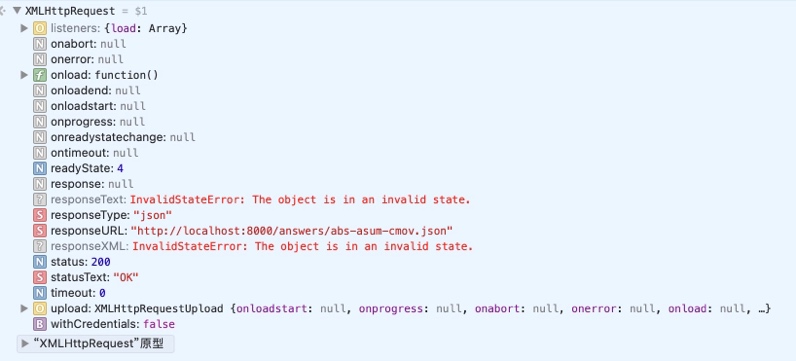
**遇到的问题：**

1.问题：前端读取本地json文件跨域错误(XMLHttpRequest cannot load <http://localhost:8000/answers/xxx.json> due to access control checks)，尝试了多种方法，用了不同的电脑和浏览器都存在跨域问题。

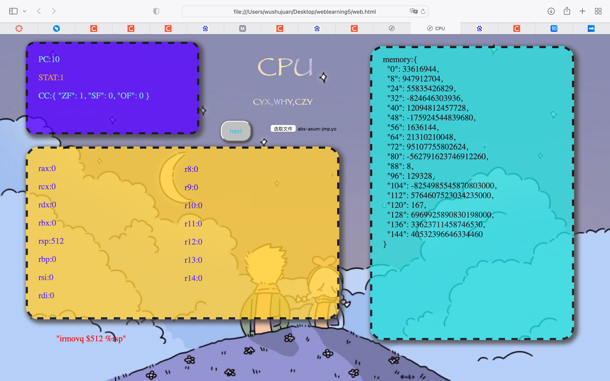
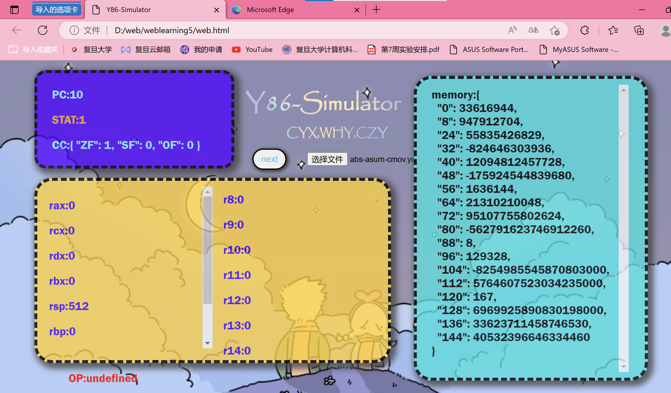
解决方法：利用python3的http:server本地服务，将文件发布到本地代理服务器上面就可以直接访问了。（脚本参考<https://blog.csdn.net/weixin_43633385/article/details/109508549>）

2.问题：电脑、网站的不兼容性（暂时没有找到解决方案）

（1）使用mac访问本地json文件时出现了InvalidStateError: The object is in an invalid state，提示对象处于无效状态，可能是Safari无法读取json文件，小组其他成员均可访问读取



（2）有一位成员的电脑上会出现某一变量undefined的情况



**感想：**

前端：第一次接触前端，心理历程：畏难->看了一些简单介绍觉得不是很难->通过搜索学习比较顺利地完成了大部分->最后被跨域问题搞疯->终于解决，不断美化网站。关于跨域问题的反思是要先搞清楚背后的原理，而不是不停地试别的方法最终发现都是同一种报错。网站还有很多要完善的地方，但通过这次pj对网页的制作有了一定的认识。

后端：心路历程：到底要干嘛->.yo文件是什么,.json文件又是什么,到底怎么输入, 要把.yo文件内容全部存起来吗->询问别人:需要开辟一块数组作为内存->假装已经成功把.yo文件存入memory数组实现功能先->弄清楚.yo的格式，读入好像也不难->ifstream读入几份文件debug->调整输出格式符合json规范->通过测试->辅助网页制作添加一个输出指令的函数输出到json文件, 由于没有头绪拖了很久才开始，真正开始后发现其实思路很清晰，不足之处是许多地方用了笨办法，比如输出到json文件完全手工调制，一份一份地生成十几份json文件…还应该进行进一步学习。最开始的时候我们都没什么底气，但最后也一起完成了任务，感谢队友~