

ICS-PJ REPORT.....	2
1.前期准备: .....	2
1.1 开发环境: .....	2
1.2 文件结构: .....	2
2.功能概述: .....	3
3.使用方法: .....	3
3.1 打开命令行.....	3
3.2 打开网页 .....	4
3.3 页面各板块介绍&板块使用.....	4
3.3.1 文件上传板块: .....	4
3.3.2 代码显示板块:.....	5
3.3.3 流水线控制板块: .....	6
3.3.4 流水线板块: .....	7
3.3.5 速度控制板块: .....	7
3.3.6 状态板块: .....	8
3.3.7 寄存器/内存板块: .....	8
3.3.8 评论区板块: .....	9
4.流水线实现 .....	11
5.前后端交互 .....	11
6.开发流程.....	12
第一周 11.14~11.17: .....	12
第二周 11.18~11.24.....	12
第三周 11.25~11.27.....	12
第三周 11.28~12.1.....	13
第四周 12.2~12.8.....	14
第五周 12.9~12.11.....	14
7.新技能 get: .....	14
8.感想.....	14
9.致谢.....	15

# ICS-PJ REPORT

## 1. 前期准备:

需要安装 Python 3.8.4、 Django 2.1, 在 Windows 上运行.

### 1.1 开发环境:

- 开发语言: Javascript/HTML/CSS/Python
- 浏览器环境: Chrome
- 第三方库: jQuery

### 1.2 文件结构:

#### *后端代码:*

##### **simulator/backend**

Stage/init.py 空文件-建包	run.py 运行测试样例
Stage/Fetch.py 流水线取指阶段	cpu.py 流水线逻辑控制
Stage/Execute.py 流水线执行阶段	const.py 常量说明
Stage/Decode.py 流水线译码阶段	Memory.py 内存类的定义, 以及加载、
Stage/MemoryAccess.py 流水线访存	读取函数
阶段	Pipeline.py 流水线类的定义
Stage/WriteBack.py 流水线写回阶段	register.py 寄存器类的定义, 以及加
Stage/PCUpdate.py 流水线更新阶段	载、读取函数
y86-code 测试样例	Alu.py 数字逻辑运算单元
init.py 空文件-建包	StringOperation.py 字符串操作函数

#### *前端代码:*

##### **simulator/static**

font/iconfont.css 定义了 icon 图标的样式

backimg.jpg 网页背景图

book.jpg 网页默认图标

style.css 定义了 main.html 中的各个 div 样式

logic.js 实现安排了流水线详细数据在页面的布置+异步刷新数据

index.js 主要是使用 JQuery 库对评论区功能的实现

simulator/templates

main.html 网页主体

**django 框架:**

manage.py 命令行工具

views.py 对应 urls.py 的路径

settings.py 设置 template、static

wsgi.py 服务器路口

urls.py 配置路径

**文件:**

Comment.json 储存了评论区输入的数据

Show.json 储存了流水线详细数据

## 2. 功能概述:

- 实现了 Y86 所有指令集
- 实现流水线控制逻辑，完整展示流水线的信息：包括条件码、CPI、寄存器、内存、各个阶段信息

在此基础上，还添加了新的功能：

- 用于交流的评论区，可提交评论、点赞、点踩、删除、记录时间，功能强大。
- web 页面简洁大方，每个版块精心设计，真的很养眼，人机交互及其快乐。
- 支持运行、步进、步退、暂停、重置操作并且能以不同频率运行
- 在代码显示框能动态查看正在运行的指令
- 能够选择查看寄存器还是内存的数据
- 增加了 iaddq 指令

## 3. 使用方法:

### 3.1 打开命令行

进入 `django-simulator/simulator` 目录，按住键盘上面的 `shift` 键，然后在当前文件夹空白的地方鼠标右击，在弹出的选项卡中可以看到一个在此处打开命令行窗口的选项，输入 `python manage.py`，

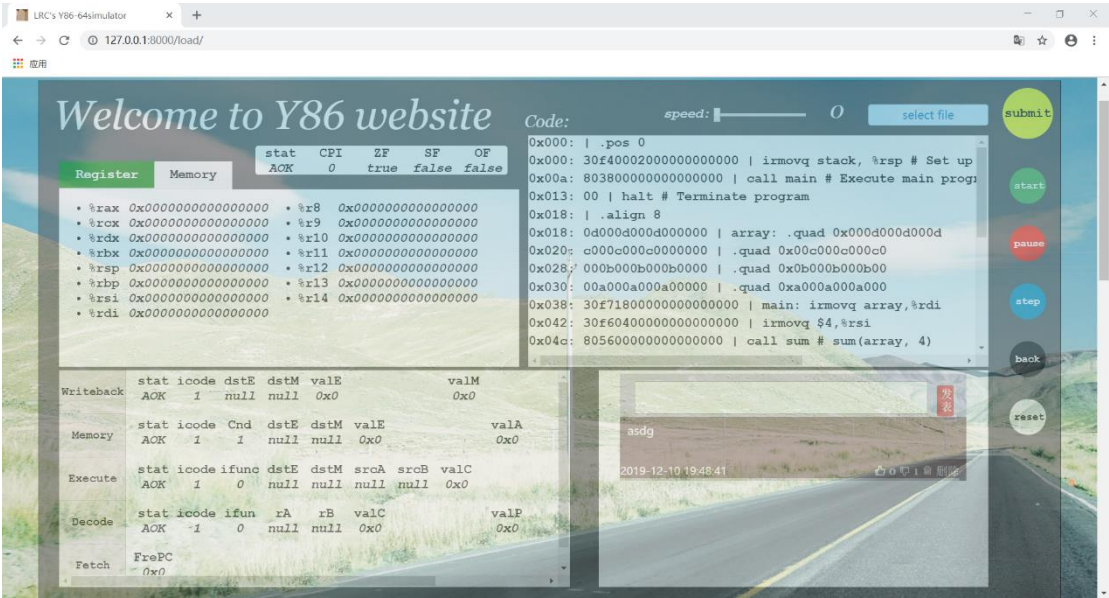
```
Windows PowerShell
PS D:\Desktop\ICS-PJ\django-simulator\django-simulator> python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 17 unapplied migration(s). Your project may not work properly until you apply them.
Run 'python manage.py migrate' to apply them.
December 11, 2019 - 12:02:07
Django version 2.2.7, using settings 'simulator.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

3.2 打开网页

在 Chrome 浏览器中输入 <http://127.0.0.1:8000/>，得到如下页面：



3.3 页面各板块介绍&板块使用

3.3.1 文件上传板块：

**特点：**使用 form 表单进行前后端数据的交互。通过 HTML 中的 input、button 标签实现前端数据的传入，并美化标签，还设置了鼠标的悬停效果；

**使用&实现：**点击蓝色方框，会弹出文件选择页面，选择文件后，点击 submit 提交，这会将文件名传递给后端处理。

**如果文件名有效（以.yo 后缀），**后端处理得到数据类型为 dict，键为流水线当前 cycle 数，对应的值为类型为字典，保存了流水线的详细数据，在通过 json.dump（）处理为 json 数据写入 show.txt 文件中。同时后端将筛选输入文件中以 0x 开头代码行传递给 HTML，通过模板语言显示在代码展示板块中；

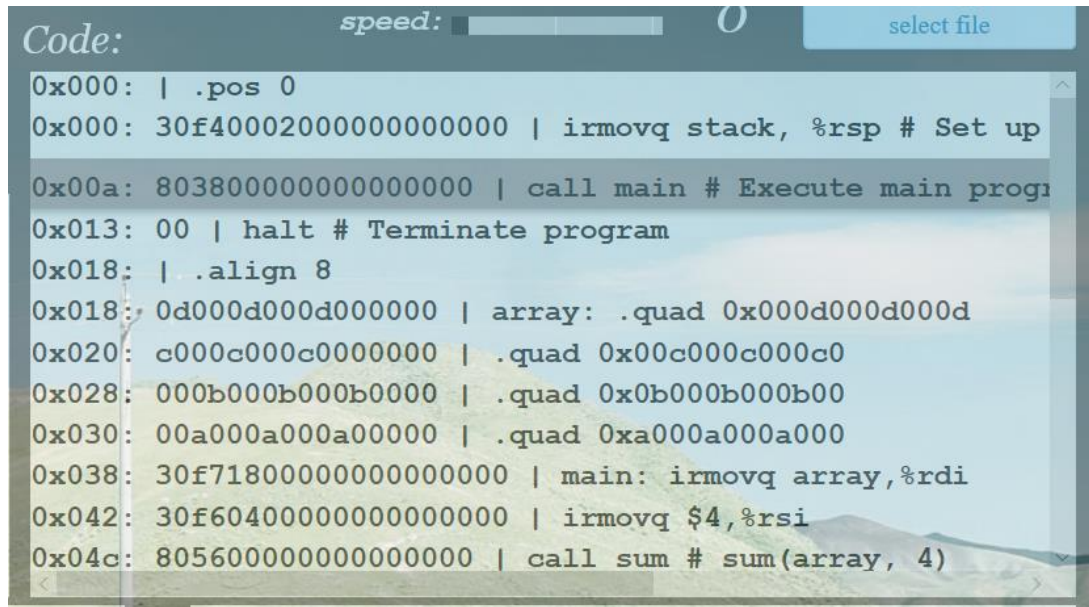
**如果文件名无效，**后端将给 HTML 返回一个 0000000000000000，通过模板语言 if 判断文件无效，将在代码展示模块中显示“文件输入失败，请输入.yo 后缀文件”



### 3.3.2 代码显示板块：

**特点：**设置了代码运行时的突出显示效果

**实现：**通过得到当前流水线的 `f_pc` 值可以得到当前运行指令对应的内存地址，获取每一行代码，对比地址值，相等则特别设置该行阴影、颜色属性，得到动态的突出的效果显示。



The screenshot shows a web-based code editor interface. At the top, there is a 'speed:' label followed by a slider bar and a '0' value. To the right is a 'select file' button. The main area displays assembly code with the following lines:

```
Code:
0x000: | .pos 0
0x000: 30f40002000000000000 | irmovq stack, %rsp # Set up
0x00a: 80380000000000000000 | call main # Execute main progr
0x013: 00 | halt # Terminate program
0x018: | .align 8
0x018: 0d000d000d000000 | array: .quad 0x000d000d000d
0x020: c000c000c0000000 | .quad 0x00c000c000c0
0x028: 000b000b000b0000 | .quad 0x0b000b000b00
0x030: 00a000a000a00000 | .quad 0xa000a000a000
0x038: 30f71800000000000000 | main: irmovq array,%rdi
0x042: 30f60400000000000000 | irmovq $4,%rsi
0x04c: 80560000000000000000 | call sum # sum(array, 4)
```

### 3.3.3 流水线控制板块：

**特点：**通过 Ajax 异步刷新网页数据；设置了按钮悬停效果，实现开始、停止、下一步、返回、重置功能；

**实现：**主要实现了一个 `step` 函数，通过 jQuery 的 `ajax` 方法获取 `show.json` 中的数据，用 `cnt` 下标索引得到数据的键值对。每次 `step()` 都会得到下一个键值对。

- Start 函数： `setInterval(step,speed)` 让 `step` 函数以 `speed(ms)`/条指令速度运行。
- Pause 函数：通过 `clearInterval` 杀死计时器。
- Step 函数： `step` 函数。

- Back 函数：将索引 cnt 的值-2，运行 step 一次。
- Reset：将索引 cnt 的值设置为 0，并运行 step 一次。



3.3.4 流水线板块：

特点：通过滚动条查看，字体好看。

实现：通过控制板块按钮控制运行，每次调用 step 函数，ajax 把对应的数据显示到页面上；val 数据之间距离大是因为储存数据长。

Writeback	AOK	1	null	null	0x0			0x0
Memory	stat	icode	Cnd	dstE	dstM	valE		valA
	AOK	1	1	null	null	0x0		0x0
Execute	stat	icode	ifunc	dstE	dstM	srcA	srcB	valC
	AOK	1	0	null	null	null	null	0x0
Decode	stat	icode	ifun	rA	rB	valC		valP
	AOK	1	0	null	null	0x0		0x0
Fetch	FrePC							
	0x000							

3.3.5 速度控制板块：

特点：动态显示滑块的值，滑条值在 0~30，计算公式  $(500-x*15)$ ，可以实现

50~500ms/step 的运行速度

**实现：**拉动滑条可以改变 setinterval (step,speed) 计时器的 speed，调节流水线运行速度。另外每次拉动滑条改变数据时，要求重新点击 start 才能改变运行速度，这样可以防止误调节滑条。



### 3.3.6 状态板块：

**特点：**CPI 的值设置为流水线运行结束时显示

**实现：**每次调用 step 函数用 ajax 方法将数据显示到页面上

stat	CPI	ZF	SF	OF
AOK	0	true	false	false

### 3.3.7 寄存器/内存板块：

**特点：**通过点击板块上方的 Register、Memory 标签可以选择查看的内容，并且设置文件没有正确上传时不能进行选择。

**实现：**通过 click 事件动态设置对应标签的 class 值，通过 css 特别设置 class 值对应的样式就可以控制对应标签内容显示与隐藏。



Register	Memory	AOK	0	true	false	false
• 0x8	0x0000000000000000	• 0x110	0x0000000000000000			
• 0x10	0x0000000000000000	• 0x118	0x0000000000000000			
• 0x18	0x0000000000000000	• 0x120	0x0000000000000000			
• 0x20	0x0000000000000000	• 0x128	0x0000000000000000			
• 0x28	0x0000000000000000	• 0x130	0x0000000000000000			
• 0x30	0x0000000000000000	• 0x138	0x0000000000000000			
• 0x38	0x0000000000000000	• 0x140	0x0000000000000000			
• 0x40	0x0000000000000000	• 0x148	0x0000000000000000			
• 0x48	0x0000000000000000	• 0x150	0x0000000000000000			
• 0x50	0x0000000000000000	• 0x158	0x0000000000000000			
• 0x58	0x0000000000000000	• 0x160	0x0000000000000000			
• 0x60	0x0000000000000000	• 0x168	0x0000000000000000			

### 3.3.8 评论区板块：

可吹可黑，可赞可踩。

**灵感来源：**希望 web 提供一个可以交流互动分享的平台（或者说更像一个 web？）

**实现：**textarea 标签实现文本输入，button 标签实现提交。点击事件主要通过 jQuery 中的 .delegate() 方法来处理。数据交互 ajax 实现。（代码在 index.js 中）。

**功能特点：**处心积虑+精心设计；逻辑可以分为两部分，一部分 ajax 实现，控制前后端数据的交互（即 js 与 python 交互）；一部分 jQuery 选择器直接控制 HTML 的显示（即 js 与 HTML 交互）。并且因为前后端数据交互，刷新页面后评论会被重新渲染加载。

- 不输入文本内容不给 submit，submit 点击后，js 会生成一个 <div></div> 插入到 HTML 中显示，并且每次 submit 都通过 ajax 方法将数据写入 comment.json

文件中；

- 赞、踩：（点击文字部分，不要点图标）可以实现数据变化，并通过 ajax 将后端对应数据修改。
- 删除，你删除的只是删掉你看到的，删不掉你看不到的：删除功能可以删除页面上对应评论，但无法删除后台对应数据，所以只能做到眼不见心为净。刷新页面之后删除的评论将重新显示在评论区。（您宝贵的意见、评价将被永远保存）



## 4. 流水线实现

**Reference:** 流水线的实现非常感谢聂希瑞学长，没有他在逻辑上的指导，我不可能这么顺利地理清流水线的逻辑，以及对一些对数据的处理方法也是向他学习。

在完成流水线的实现过程中我主要梳理了以下问题：

1. 各个阶段的逻辑。（这个通过看书可以了解）。
2. 流水线各个阶段如何更新。更新让我联想到 swap 函数，通过类似方式就容易解决了，就逐渐想到把前后状态封装成一个类，但对于五个阶段倒序执行这一逻辑我确实没能很快想到。
3. 原始数据（.yo 文件）的处理。这一块本来不难却琢磨了很久。

## 5. 前后端交互

Django 框架下开发 web，我没有使用 django 的 model，而是使用了.json 文件来储存后端的数据。最后进行了多种方式，不同语言之间数据的交互：

**Html 与 python:** 通过在 javascript 文件的 ajax 方法获取数据并在 js 文件中处理，实现了异步刷新；通过 Html 的 form 表单标签发送请求，并通过使用 django 模板语言获取返回的数据。

**Javascript 与 Html:** 通过 jQuery 选择器、document 对象获取 html 的元素，以此进行 web 页面上数据的修改……

## 6. 开发流程

### 第一周 11.14~11.17:

开始阶段算是一头雾水，包括语言的选择、前后端如何交互、选择 linux 还是 window，terminal 是什么……。经过漫长的百度了解，包括 python (CGI)、c 自带 GUI、Web、Unity 都了解了一些。过程中被推荐了 python，因为它里面封装了很多制作图形界面相关的库函数（虽然我也还不知道有哪些）。

在王辰浩学长的帮助下，我理解了前后端，真正明白这个 PJ 要做的事情并在学长的指导下，最后大概决定使用 python 后端，django+web 前端。

### 第二周 11.18~11.24

学习了 python，学会使用了给的 sim 样例在图形界面、终端的输出，重新仔细阅读了教材熟悉了流水线的各种细节并对如何用 python 实现有了想法。在 11.24 晚上写出了大致框架，并做了 memory 方面的调试。

### 第三周 11.25~11.27

11.25 调试过 prog1~prog10。

11.26 调试过其它函数。

11.27 学习 markdownpro 书写实验报告，实现了 CPI 的计算，实现了 IADDP 指令

以下是大概的问题描述,还有许多小问题遗忘了（辣眼睛）:

- prog1~4: 问题多到数不清, python 全局变量、变量是否可变……的 python

基础知识。

- prog5: writeMemory 参数设置为 int; PCUpdata 中的条件问题
- prog6: hex () 自带“0x”问题
- prog7: 由于 jne 的 ifunc 没读到 4, 没有设置成功 Cnd, 原因是 ifunc 与 iFunc 的统一
- prog8: d\_srcA 不用 mrmvoq , d\_srcB 不用 rrmovq
- prog9: 处理了 invalid\_instr 的输出
- prog10: 处理地址越界的情况
- cjrr: 大改 ALU 逻辑
- ret-hazard: 修改 PCUpdata, +not D\_stall
- abs-asum-movq: PCUpdata 逻辑的优先级
- asumr: Execute 中 now.E[icode]不用, IRRMOVQ
- 加入 iaddq 时: 不要忘了 set\_cc 的设置
- 实现 CPI: W\_bubble 不用计算, 因为此时指令已经结束
- Cjr: 重写了 CPI 计算方法

### 第三周 11.28~12.1

11.28 第一阶段提交, 加入通过命令行控制后端功能

11.29~11.30 学习 html、css

12.1 学习 jQuery、javascript、bootstrap

解决问题: 前端 http 请求, 使得 python 文件读取 .yo 文件, 再将 python 产生的字典数据传递到 js 处理, 用 html 文件确定页面布局, 将按钮、输入框等交互元素与 js 文件中函数绑定, 因此可以将要显示出来的结果显示在页面对应位置, 网

页样式通过.css 文件修饰。

#### 第四周 12.2~12.8

12.2~12.6 通过菜鸟教程、csdn 阅读了各种 html、js、css 文件，期间在学习理解 ajax、jQuery 上分别花了大量时间。

12.7~12.8 初步完成了前端 web 页面的布局，吐槽一下 css 写的真是又臭又长。

#### 第五周 12.9~12.11

12.9~12.10 完成了评论区功能和代码板块动态指示当前运行指令功能

12.11 完善了一下评论区功能，书写实验报告。

### 7. 新技能 get:

- Python
- HTML5/CSS3/Javascript
- Django
- jQuery Ajax/Documnet
- JSON
- Github

### 8. 感想

体会过搜寻半天也没结果的迷茫、体会过代码看不懂的忧伤、体会过苦思半天却发现走了弯路的苦涩；但也记得一步步找到解决问题逻辑的振奋、记到 debug 后样例正确输出的手舞足蹈、记得一朝顿悟的痛快、记得美化修饰一个个

web 元素的满满成就、记得实现一个个功能后与室友诉说分享的喜悦。

老实说，自己是个佛系的人，加入拔尖班就是希望挑战自己，测试自己的潜力。从第一阶段参考学习到第二阶段通过搜索百度独立完成交互、修改的所有逻辑，虽然两个阶段逻辑侧重点不同，不可一概而论，但我是真心欢喜自己的变化。还记得蔡学长在学期开始前就教导我们：自主学习、充分利用网络这项技能很重要。我一直记在心里，我想，现在自己也该有了一定进步，知道对于一个问题搜什么？怎么有效率地搜？

另外，学习新知识的过程是那么充满挑战而充实快乐。除了 c++ 知识作为基础外，这次 PJ 所用的各种语言 Python、Html……还有一些库都是从头到尾自学，看博客、看视频……摸索它们中适合，能够应用到这道 PJ 的方法。学了这么多东西，做到了自己希望做到的事，修饰出了一个还算满意的 web 界面。

回顾历程，其实，坚持是关键，每天的投入让我每天都在成长。

总之，和同学们之间差距仍在，“革命尚未成功，我辈仍需努力”。

## 9. 致谢

感谢王辰浩学长给我的 PJ 指明了方向。

感谢聂希瑞学长提供他的优秀 PJ 供我参考。

感谢在实验报告书写方面以张作柏学长、周芯怡学姐 PJ 的书写格式参考。