计算机系统基础
Programming Assignment

PA 0 – 在线实验平台与PA简介

2020年9月10日 / 2020年9月11日

南京大学《计算机系统基础》课程组

目录

- 课程基本信息
- 在线实验平台的使用
- PA课程内容和原理简介

目录

- 课程基本信息
- 在线实验平台的使用
- PA课程内容和原理简介

课程基本信息

- 上课时间地点

课程 节次	四	五
1--2 节	离散数学 仙 I -102	
3--4 节	数据结构 （一）仙 II-306 （二）仙 II-406	数字电路与数字系统实验（一） 基础实验楼乙 125、126 数字电路与数字系统 仙 I -102
5--6 节	程序设计基础实验 基础实验楼乙 126 计算机系统基础 （一）仙 II-504	计算机系统基础 （二）逸 B-104
7--8 节	离散数学习题 仙 I -102	
9--10 节	数据结构（实验），基础实验楼乙 124、125	
11 节		

- 在线实验平台

  http://114.212.10.200
  使用方法接下来详细演示

- 课程QQ群

  群名称：2020秋-计算机系统基础
  群　　号：452987148
  加群时/后提供学号姓名

- 上学期在B站的视频（PA 0-1不看）

  https://space.bilibili.com/284613991/channel/detail?cid=103368

目录

- 课程基本信息
- 在线实验平台的使用
- PA课程内容和原理简介

在线实验平台的使用说明

开发

汪亮

Email : wl#nju.edu.cn (replace # with @)

QQ/微信: 715764994

庄宇乾

Email: zhuangyq#smail.nju.edu.cn

张明亚

Email: DG20330034#smail.nju.edu.cn

PANEMU OL

- 无须自己部署环境!
- 更强的防作弊功能!
- 更好的学习体验?

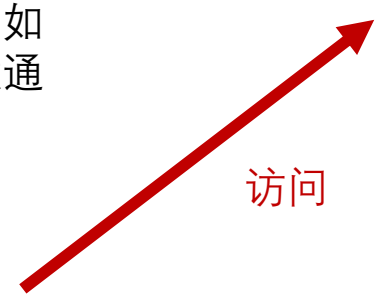
Driven by ~~magic~~.

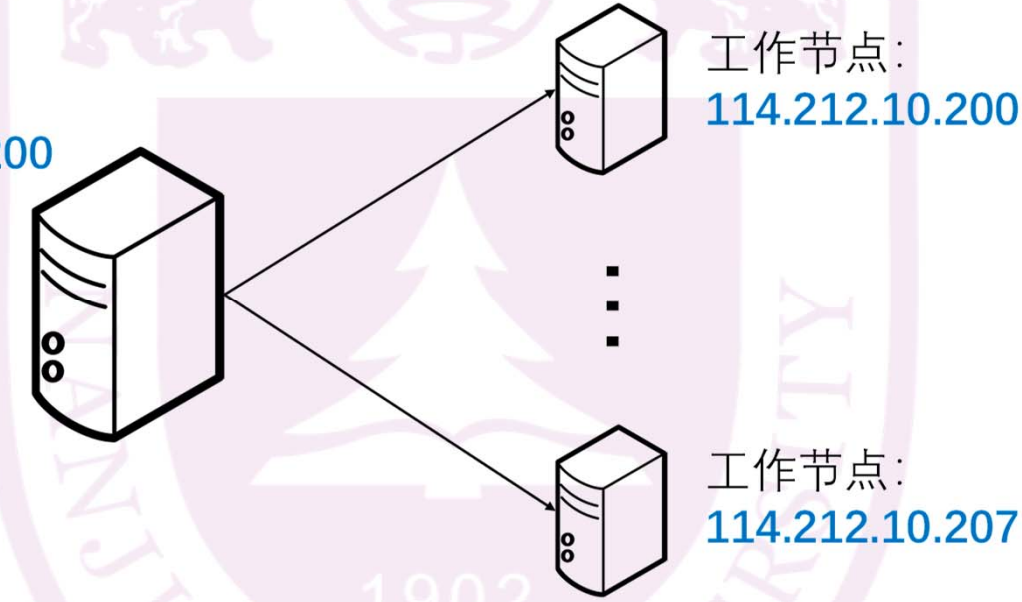
great passion and a few bugs

在线实验平台的使用说明

确保自己连接了校园网，如果在校外访问，则需要拨通南京大学VPN

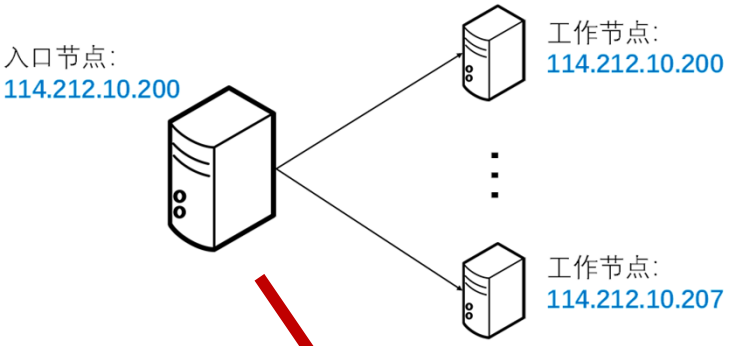

使用我们推荐的浏览器访问在线实验平台，若使用的是firefox，按照教程首页上的说明进行设置





在线实验平台架构

（1）访问平台



汪亮

Email : wl#nju.edu.cn (replace # with @)

QQ/微信: 715764994

庄宇乾

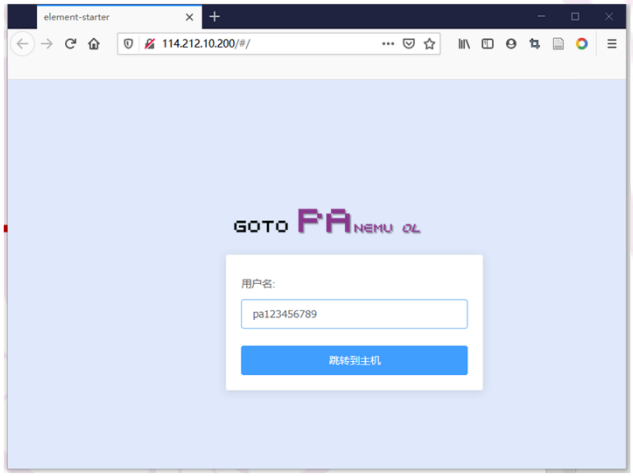
Email: zhuangyq#smail.nju.edu.cn

张明亚

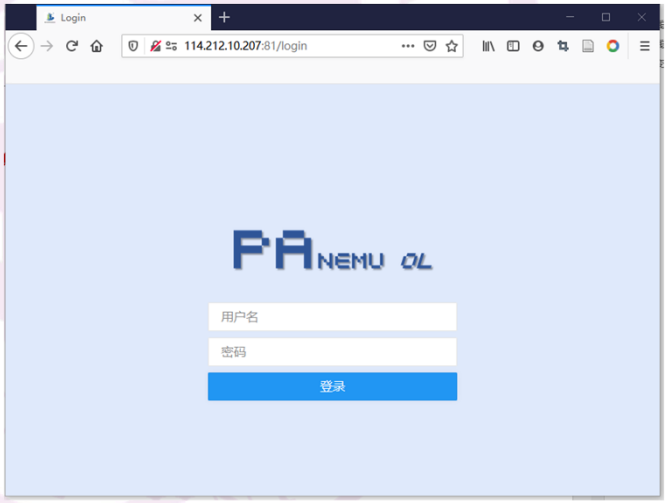
Email: DG20330034#smail.nju.edu.cn

跳转失败:
群里找老师和助教

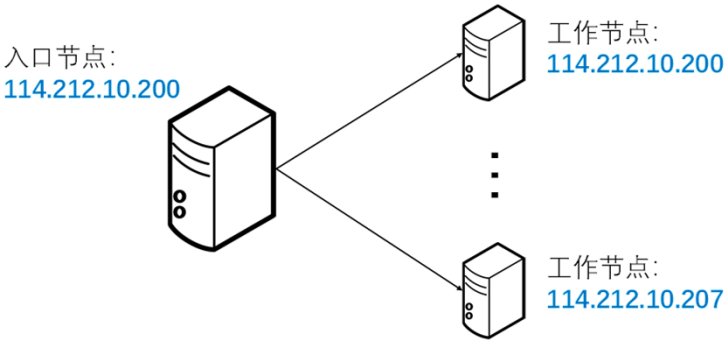
1. 访问入口节点
2. 输入**pa学号**作为用户名
3. 点击'**跳转到主机**'按钮



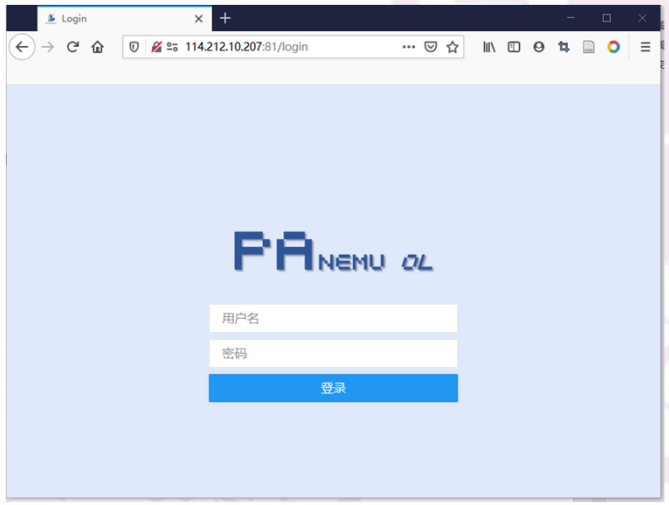
跳转成功:
来到工作节点



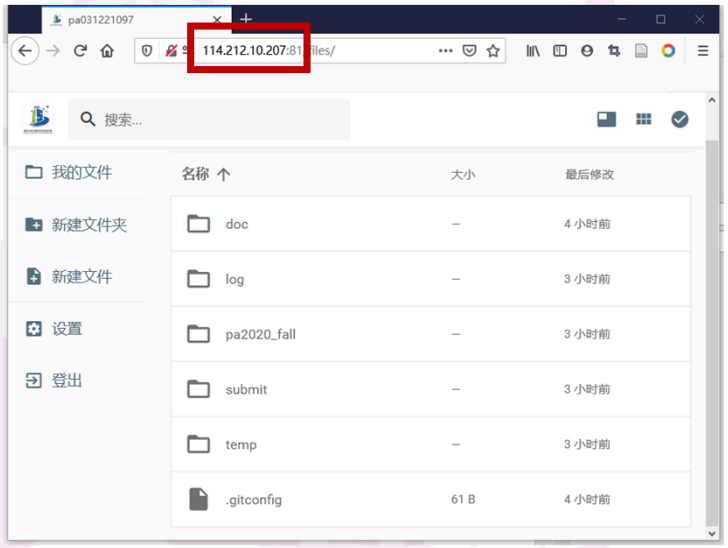
（1）访问平台



1. 输入**pa学号**作为用户名
2. 输入**pa学号**作为密码
3. 点击‘**登陆**’按钮
4. 如登陆有问题，找老师和助教



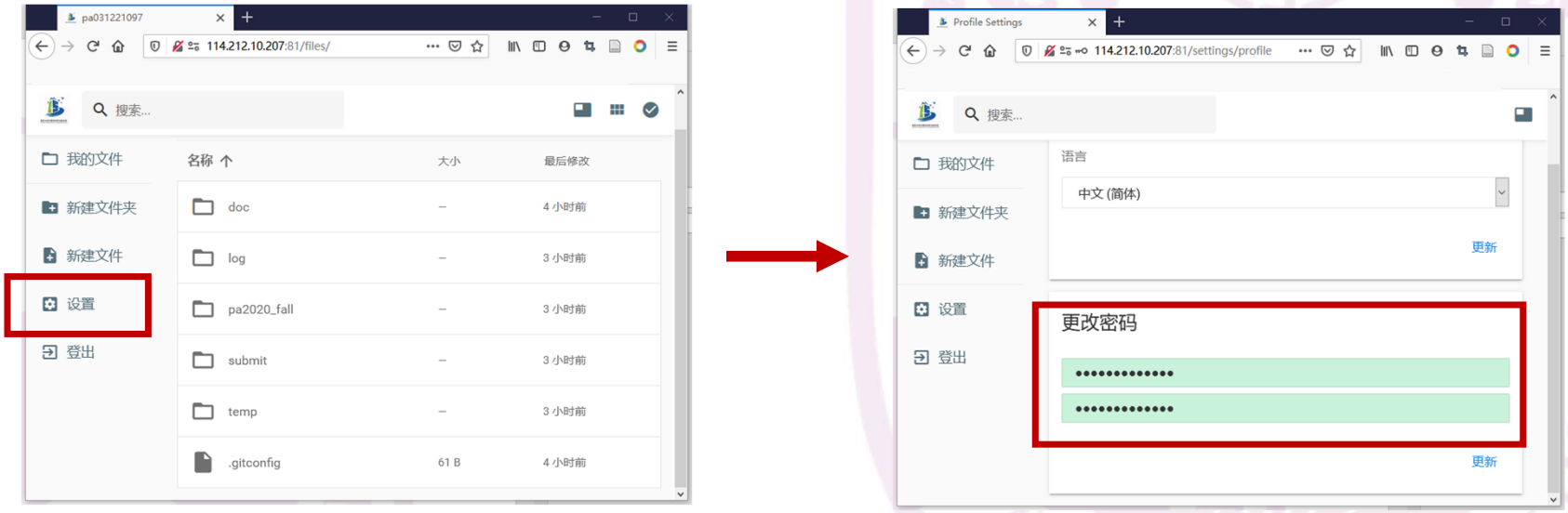
记住自己的工作节点IP地址和端口号（81），
下次可无需再访问入口节点



登陆成功!

✿✿ヽ(°▽°)ノ✿

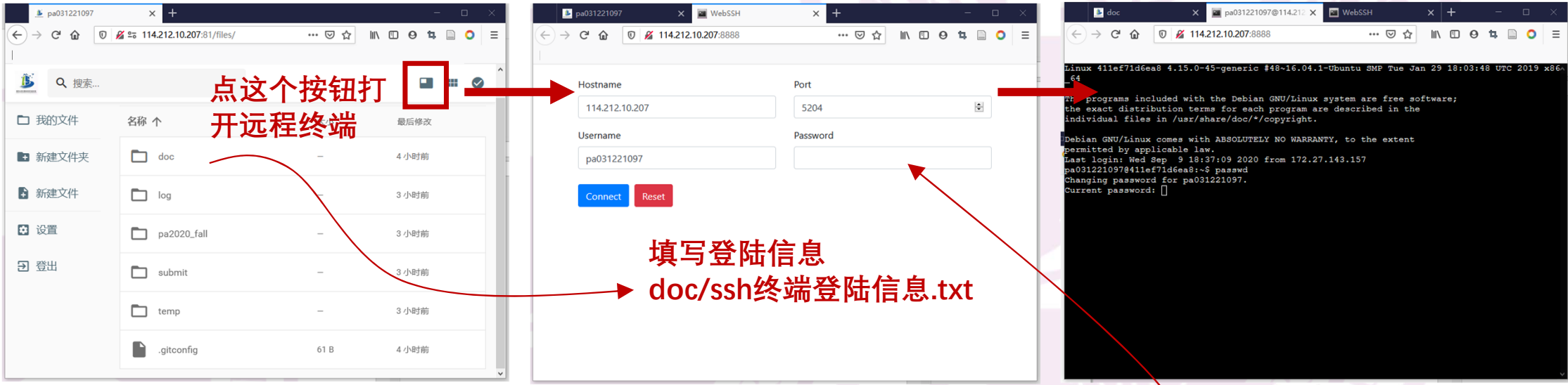
(2) 更改密码



第一处：更改网页登陆密码

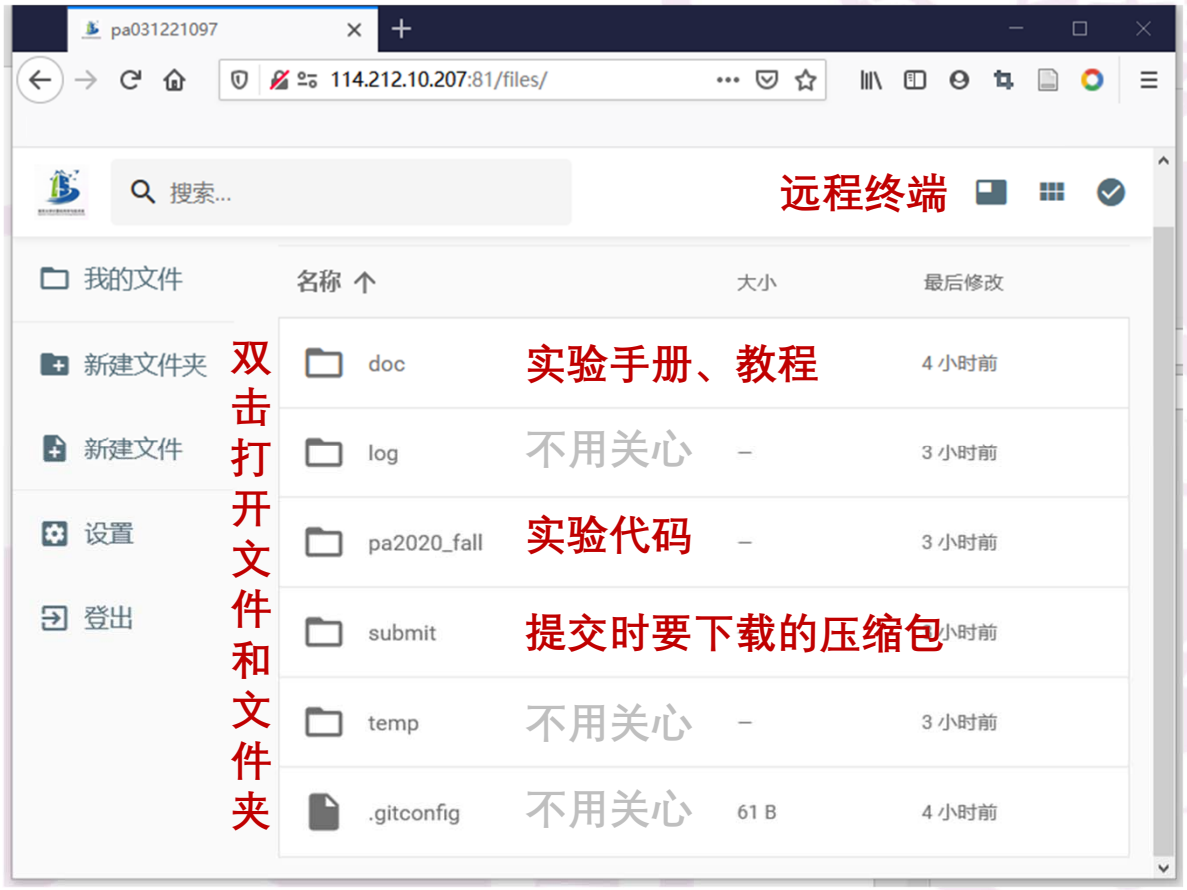
(2) 更改密码

命令passwd

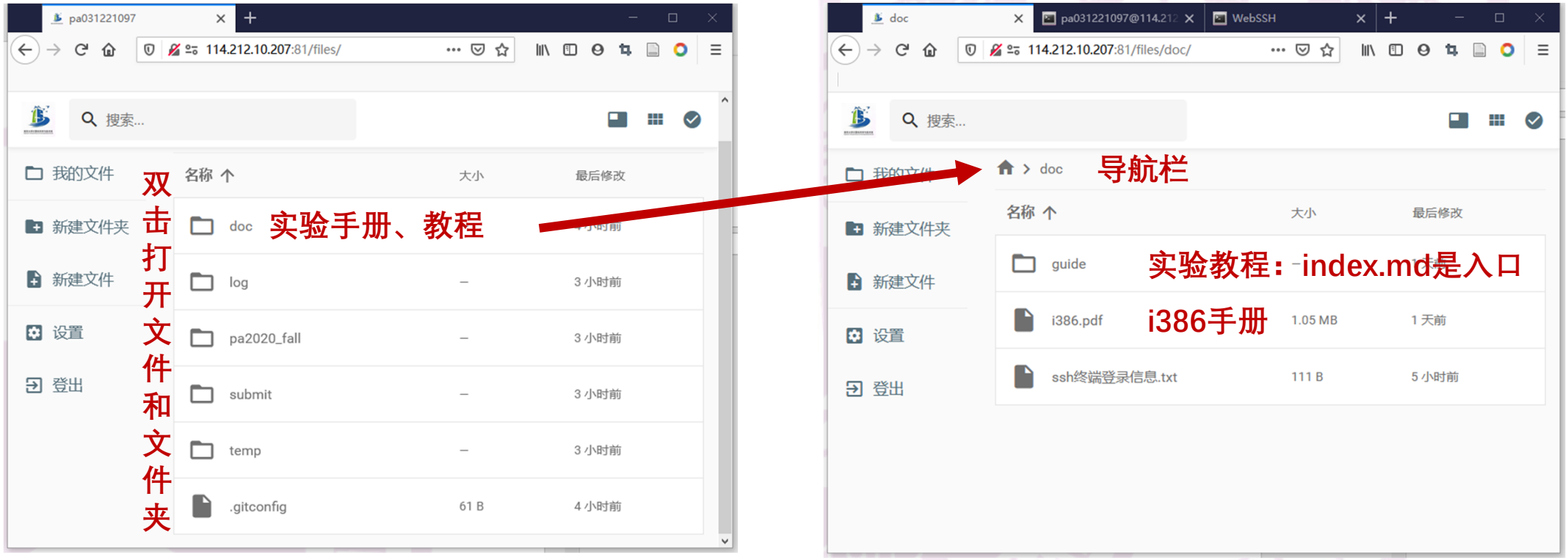


第二处：更改个人docker密码

(3) 熟悉环境

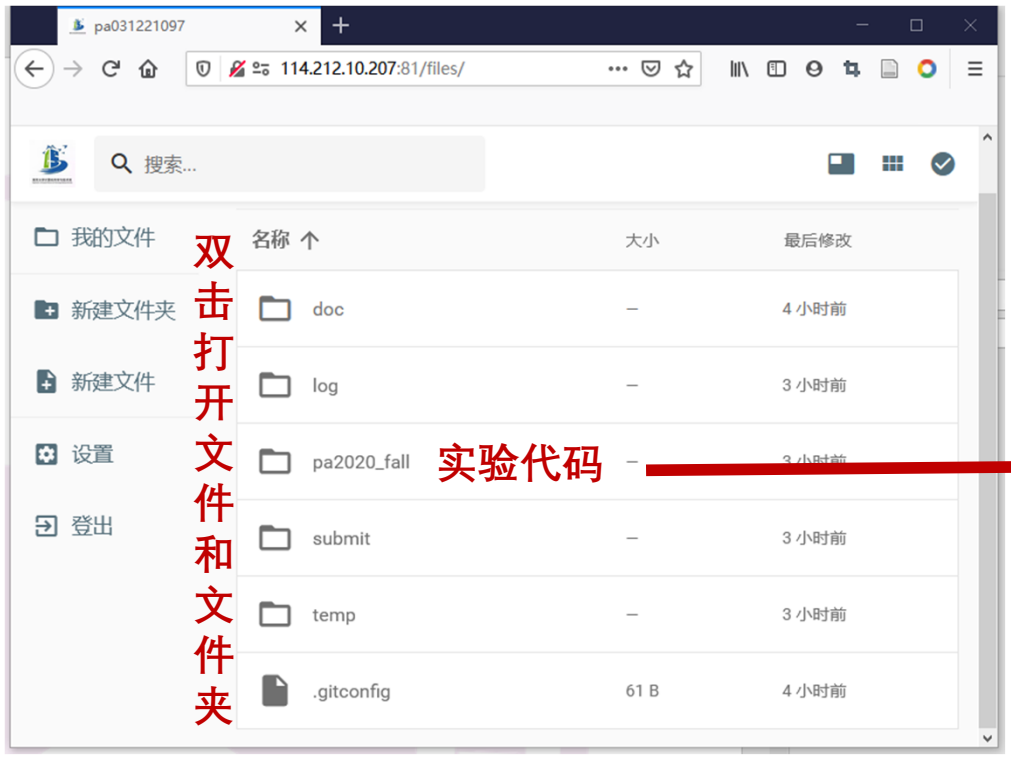

pa031221097
114.212.10.207:81/files/
搜索...
远程终端
我的文件
新建文件夹
新建文件
设置
登出
双击打开文件和文件夹
名称
大小
最后修改
doc
实验手册、教程
4 小时前
log
不用关心
3 小时前
pa2020_fall
实验代码
3 小时前
submit
提交时要下载的压缩包
temp
不用关心
3 小时前
.gitconfig
不用关心
61 B
4 小时前

（3）熟悉环境



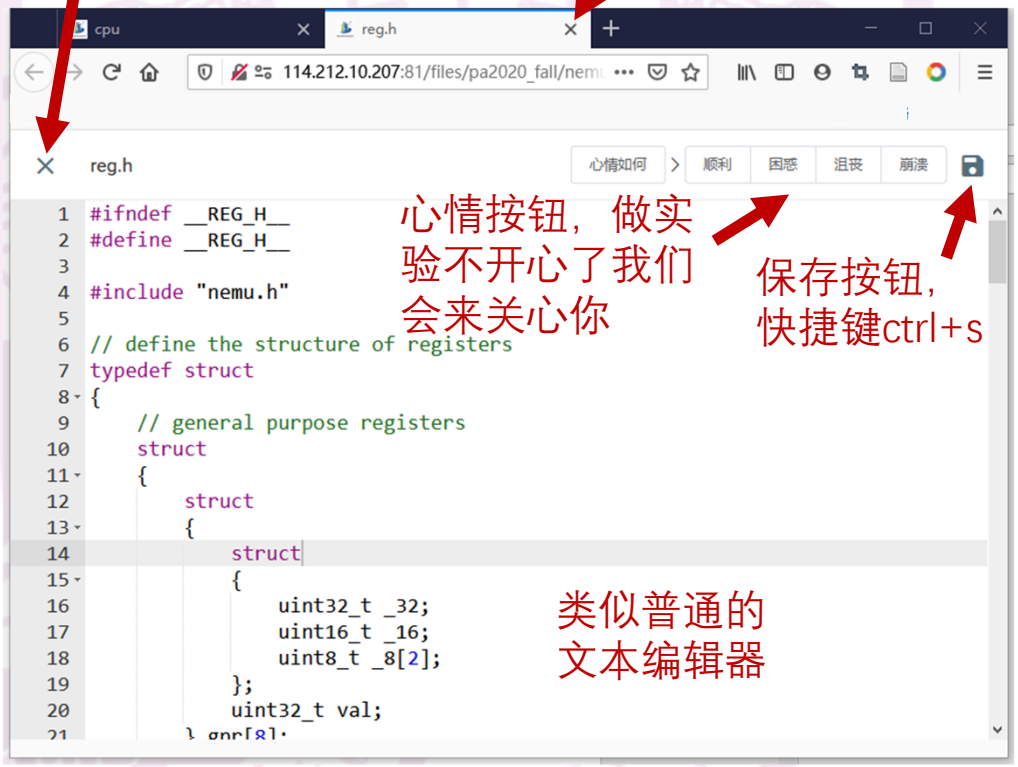
找到实验的相关手册和教程

（3）熟悉环境



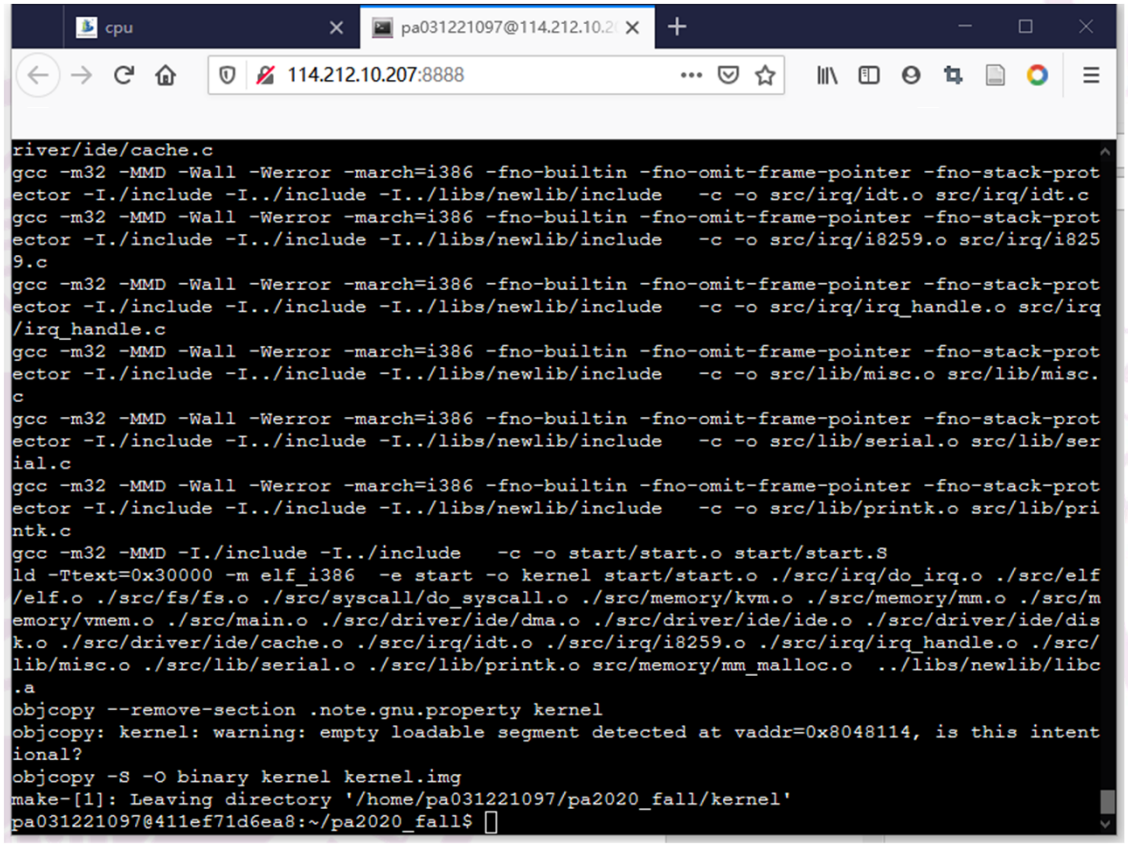
关闭按钮2：保存了也会再弹个窗提醒你

关闭按钮1：没有保存也不会有任何提示



熟悉代码编辑界面

(4) 尝试一下make



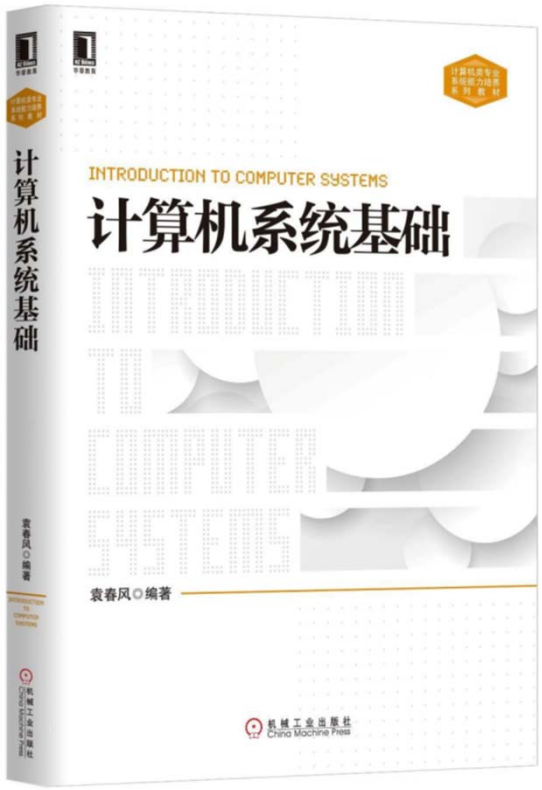
> cd pa2020_fall/
> make clean
> make

✤✤ヽ(°▽°)ノ✤

目录

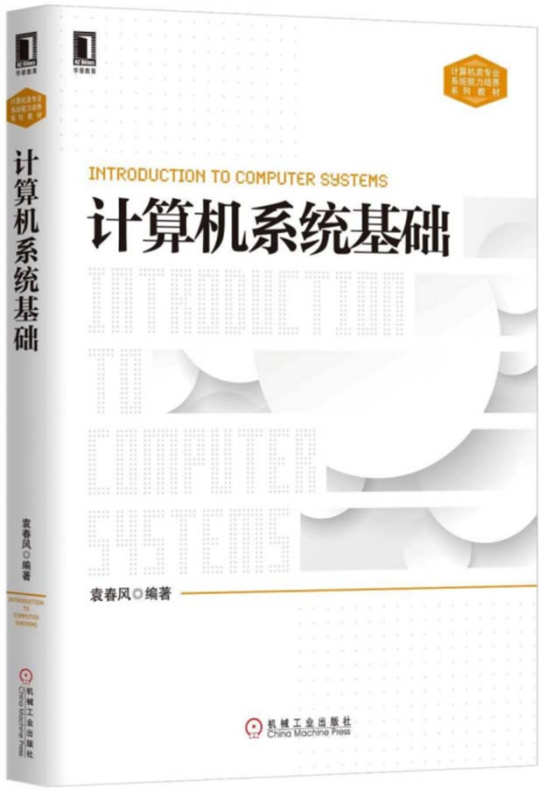
- 课程基本信息
- 在线实验平台的使用
- PA课程内容和原理简介

PA简介



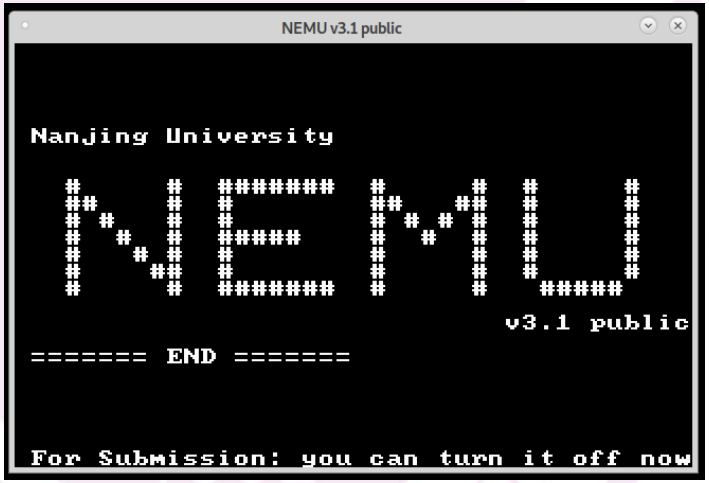
计算机系统基础（第二版）

PA简介



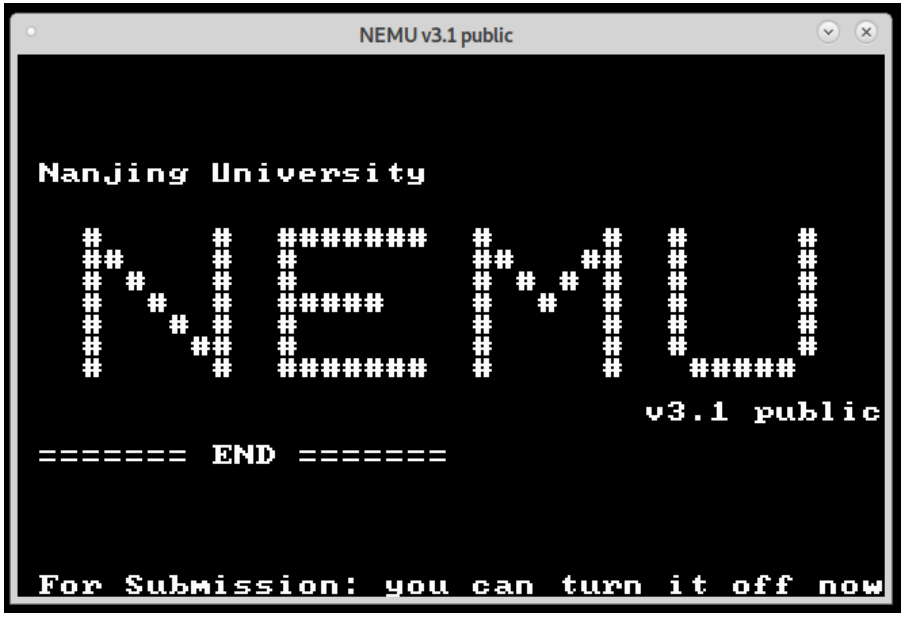
编程实践部分

Programming Assignment，简称PA



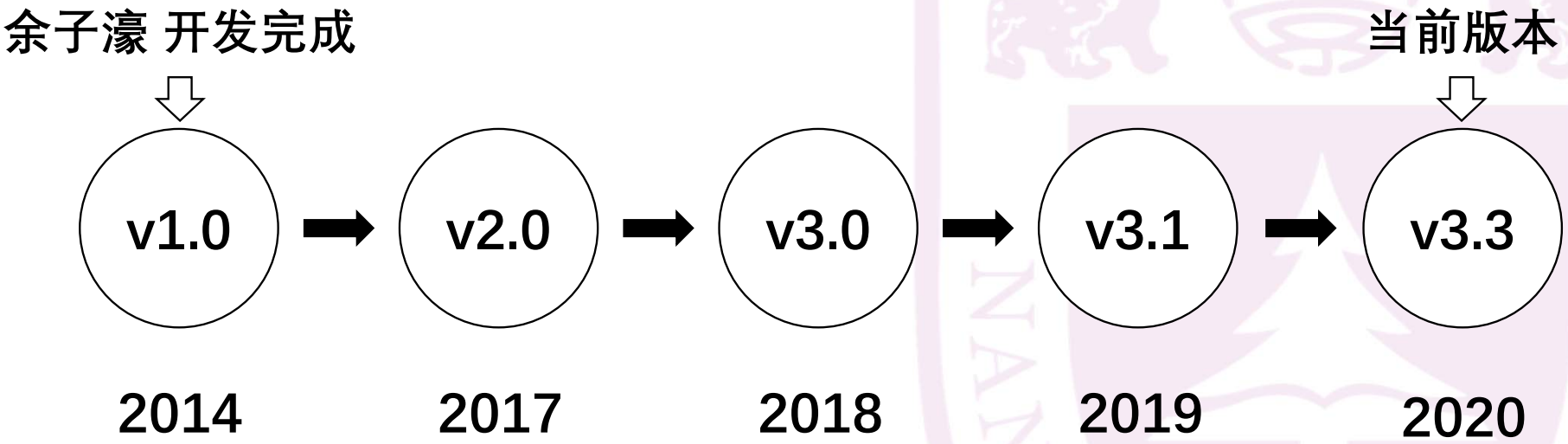
PA简介

PA实验的目的：实现NEMU，一个简化的i386模拟器



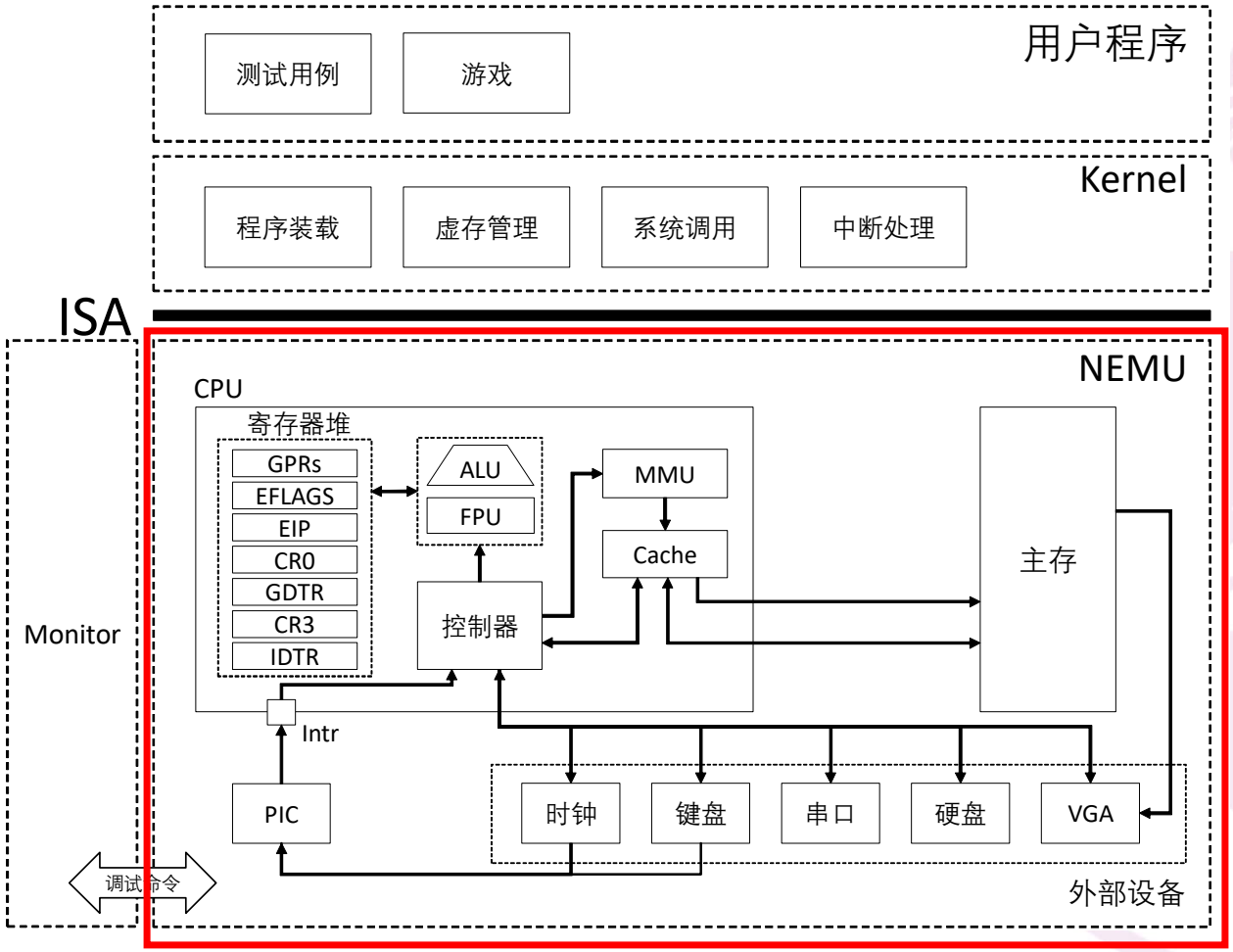
- 由C语言编写
- 以用户软件的形态运行
- 能够执行i386指令集程序

PA简介



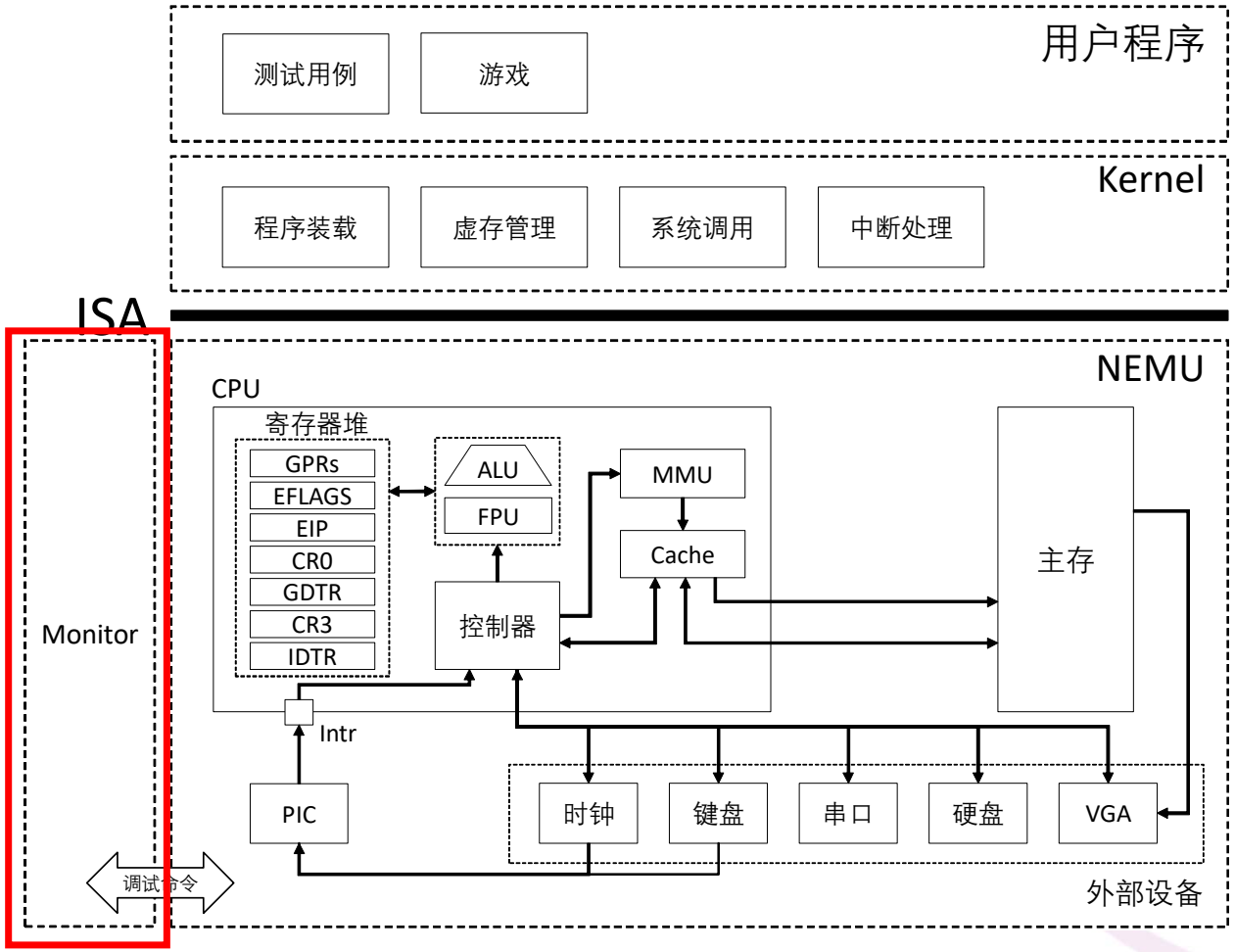
PA的简要历史

PA简介



NEMU模拟器
(NJU Emulator)

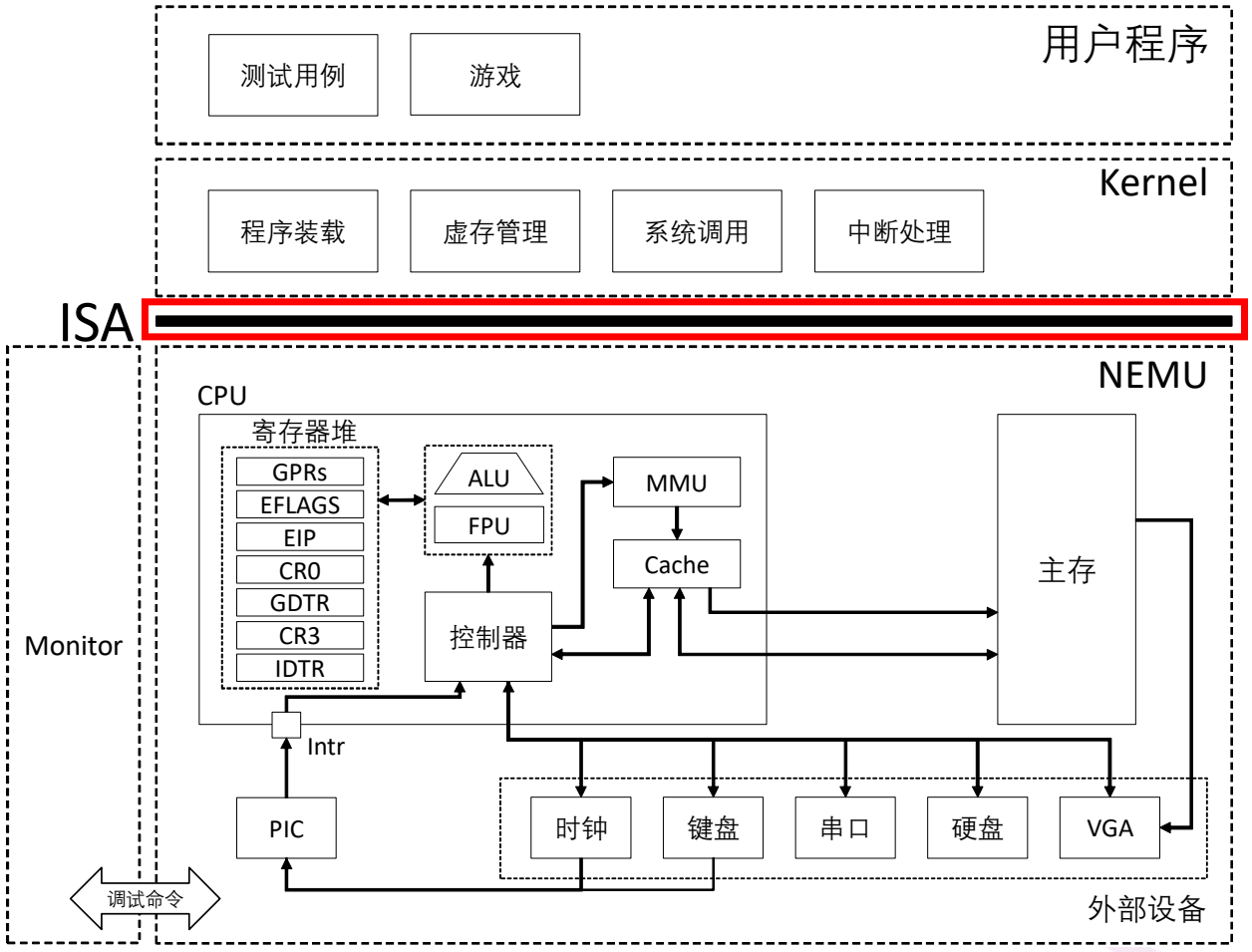
PA简介



内建调试器

NEMU模拟器
(NJU Emulator)

PA简介



内建调试器

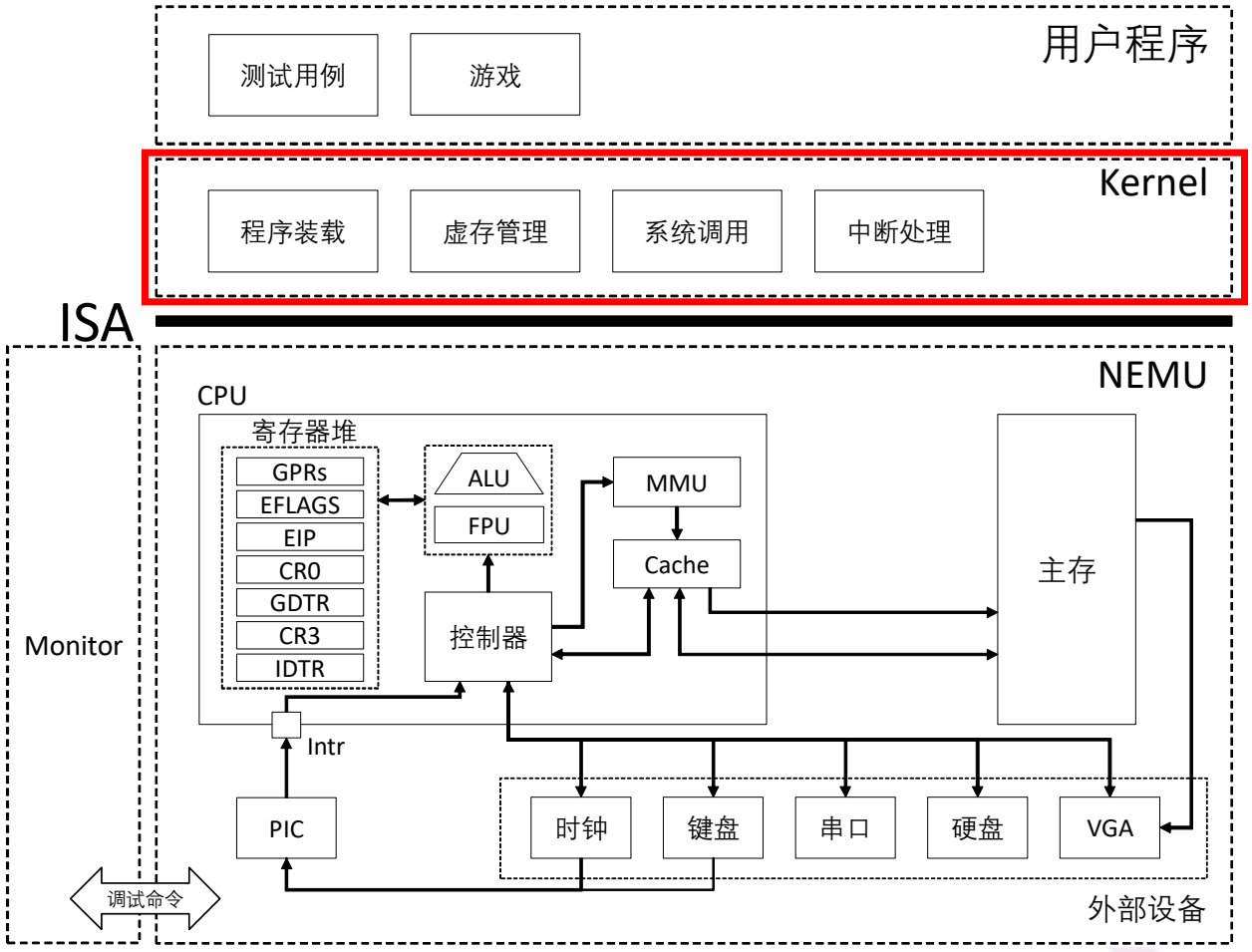
模拟硬件

简化的i386指令集体系结构

实现

NEMU模拟器
(NJU Emulator)

PA简介



内建调试器

模拟硬件

微型操作系统

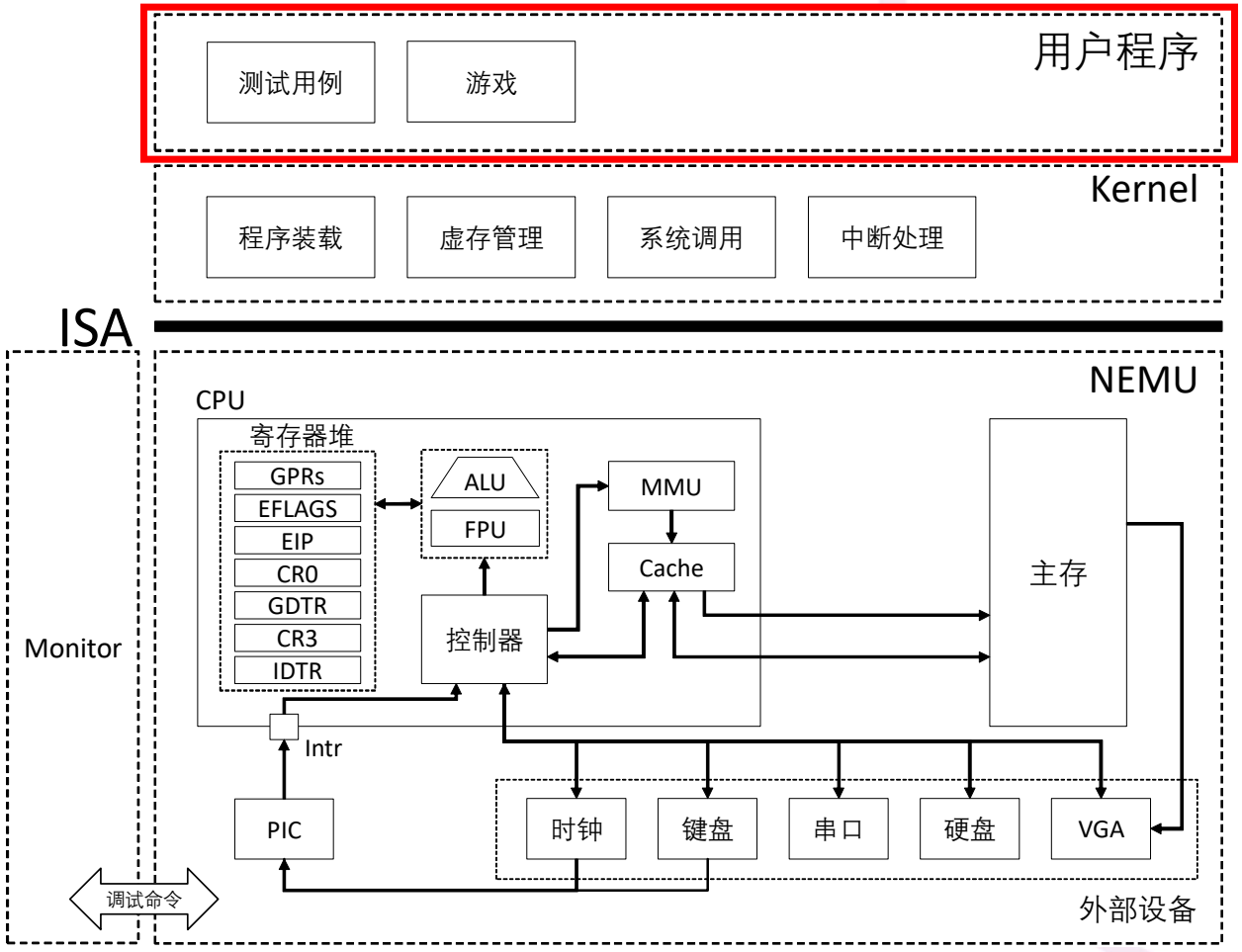
运行

简化的i386指令集体系结构

实现

NEMU模拟器
(NJU Emulator)

PA简介



软件

模拟硬件

内建调试器

用户程序

微型操作系统

运行

简化的i386指令集体系结构

实现

NEMU模拟器
(NJU Emulator)

PA简介

PA分为四个大的阶段，每个大阶段又拆分为三个小阶段

- PA 1 - 数据的表示、存取和运算
- PA 2 - 程序的执行
- PA 3 - 存储管理
- PA 4 - 异常、中断与I/O

PA的原理

如何编写软件模拟一台计算机？

PA的原理

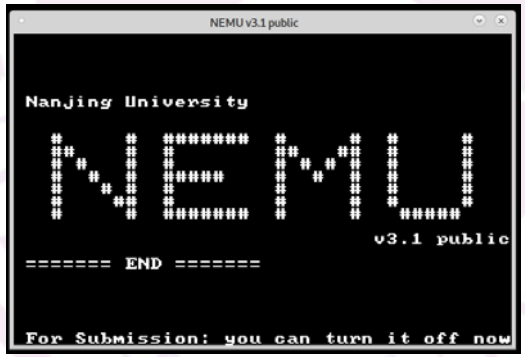
可以使用软件模拟一台计算机！

√ 模拟PC机　　√ 模拟手机　　√ 模拟游戏机

PA的原理

同样的软件

√ 真机



√ 模拟器

PA的原理

- 从机器的视角来看软件

hello_world.c

```
#include<stdio.h>

int main() {
        printf("Hello World!\n");
        return 0;
}
```

编译得到可执行文件

```
$ gcc –o hello_world hello_world.c
```

机器运行可执行文件

```
$ ./hello_world
```

得到执行结果

```
Hello World!
```

PA的原理

- 从机器的视角来看软件

hello_world.c

```
#include<stdio.h>

int main() {
        printf("Hello World!\n");
        return 0;
}
```

编译得到
可执行文件

机器运行
可执行文件

得到
执行结果

```
$ gcc –o hello_world hello_world.c

$ ./hello_world

Hello World!
```

软件

PA的原理

- 从机器的视角来看软件

```
$ gcc –c –o hello_world.o hello_world.S
$ hexdump hello_world.o | less
```

hello_world.c

```
#include<stdio.h>

int main() {
        printf("Hello World!\n");
        return 0;
}
```

编译

低位

hello_world.o 查看其内容

```
0000000 457f 464c 0101 0001 0000 0000 0000 0000
0000010 0001 0003 0001 0000 0000 0000 0000 0000
0000020 02fc 0000 0000 0000 0034 0000 0000 0028
0000030 000f 000e 0001 0000 0007 0000 4c8d 0424
0000040 e483 fff0 fc71 8955 53e5 e851 fffc ffff
0000050 0105 0000 8300 0cec 908d 0000 0000 8952
0000060 e8c3 fffc ffff c483 b810 0000 0000 658d
0000070 59f8 5d5b 618d c3fc 6548 6c6c 206f 6f57
0000080 6c72 2164 8b00 2404 00c3 4347 3a43 2820
0000090 6544 6962 6e61 3620 332e 302e 312d 2938
```

偏移量

数据，两字节一组，小端

```
0000030 000f 000e 0001 0000 0007 0000 4c8d 0424
0000040 e483 fff0 fc71 8955 53e5 e851 fffc ffff
0000050 0105 0000 8300 0cec 908d 0000 0000 8952
0000060 e8c3 fffc ffff c483 b810 0000 0000 658d
0000070 59f8 5d5b 618d c3fc 6548 6c6c 206f 6f57
0000080 6c72 2164 8b00 2404 00c3 4347 3a43 2820
0000090 6544 6962 6e61 3620 332e 302e 312d 2938
```

```
$ gcc –c –o hello_world.o hello_world.S
$ objdump -d hello_world.o | less
```

hello_world.o 反汇编其内容

```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

```
0000030 000f 000e 0001 0000 0007 0000 4c8d 0424
0000040 e483 fff0 fc71 8955 53e5 e851 fffc ffff
0000050 0105 0000 8300 0cec 908d 0000 0000 8952
0000060 e8c3 fffc ffff c483 b810 0000 0000 658d
0000070 59f8 5d5b 618d c3fc 6548 6c6c 206f 6f57
0000080 6c72 2164 8b00 2404 00c3 4347 3a43 2820
0000090 6544 6962 6e61 3620 332e 302e 312d 2938
```

```
$ gcc –c –o hello_world.o hello_world.S
$ objdump -d hello_world.o | less
```

hello_world.o 反汇编其内容

```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

```
0000030 000f 000e 0001 0000 0007 0000 4c8d 0424
0000040 e483 fff0 fc71 8955 53e5 e851 fffc ffff
0000050 0105 0000 8300 0cec 908d 0000 0000 8952
0000060 e8c3 fffc ffff c483 b810 0000 0000 658d
0000070 59f8 5d5b 618d c3fc 6548 6c6c 206f 6f57
0000080 6c72 2164 8b00 2404 00c3 4347 3a43 2820
0000090 6544 6962 6e61 3620 332e 302e 312d 2938
```

```
$ gcc –c –o hello_world.o hello_world.S
$ objdump -d hello_world.o | less
```

hello_world.o 反汇编其内容

```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

交给机器执行的
就是指令的序列

PA的原理


```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

编写

软硬件遵守共同的规范

指令集体系结构
Instruction Set Architecture, ISA

实现

用硬件实现ISA

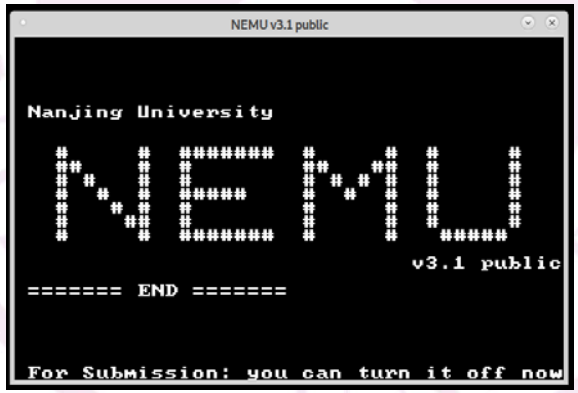
PA的原理


```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

编写

软硬件遵守共同的规范

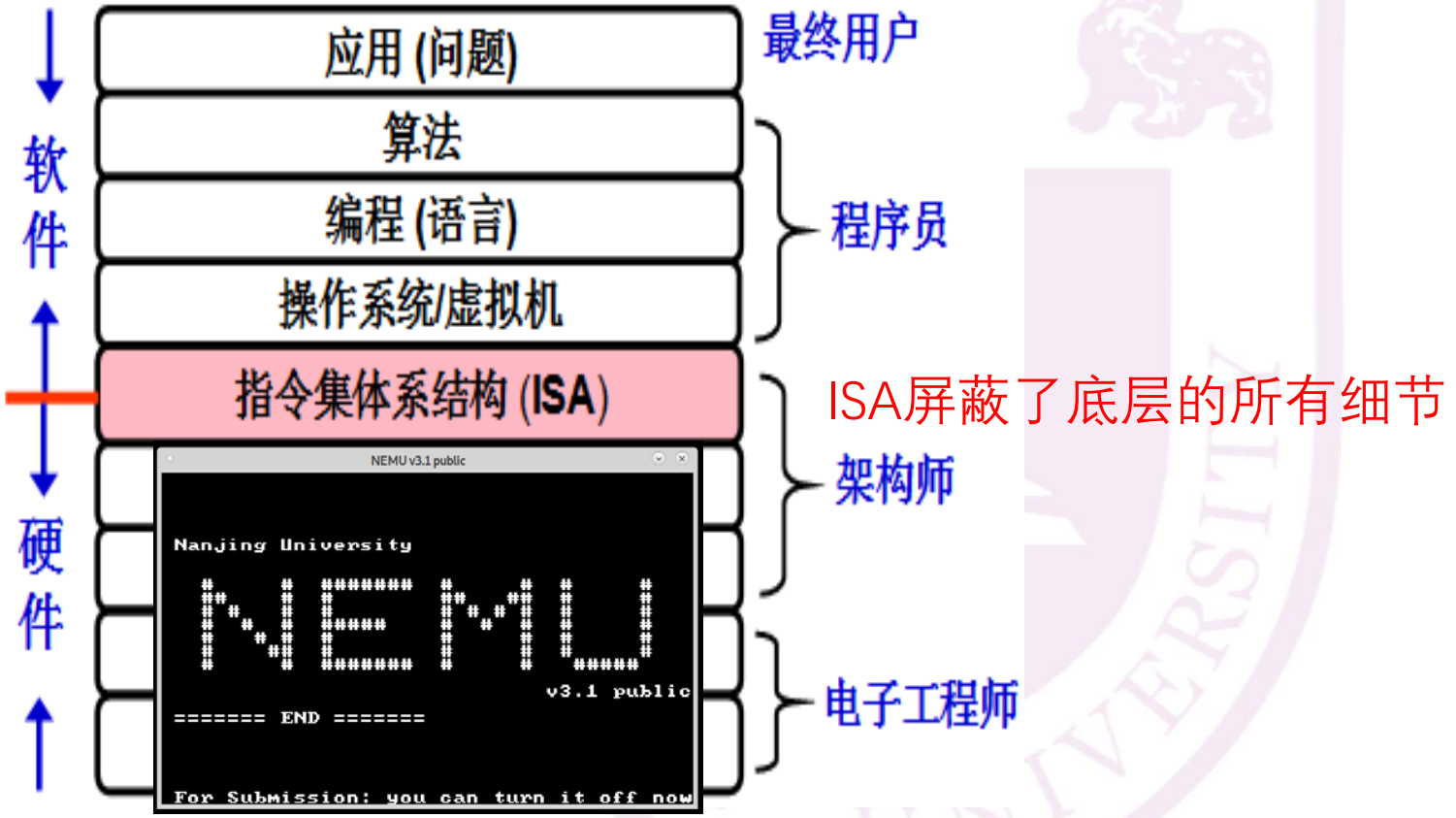
指令集体系结构
Instruction Set Architecture, ISA



实现

替换ISA实现方案，用软件模拟

PA的原理


软件
硬件
应用 (问题)
算法
编程 (语言)
操作系统/虚拟机
指令集体系结构 (ISA)
最终用户
程序员
架构师
电子工程师
ISA屏蔽了底层的所有细节
NEMU v3.1 public
Nanjing University
v3.1 public
======= END =======
For Submission: you can turn it off now

PA的原理

软件

编译得到指令的序列

```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

```
pa2020_fall/
├──── game                  // 包含游戏相关代码
├──── include               // PA整体依赖的一些文件
│     ├──── config.h        // 一些配置用的宏
│     └──── trap.h          // trap相关定义，不可改动
├──── kernel                // 一个微型操作系统内核
├──── Makefile              // 帮助编译和执行工程的Makefile
├──── Makefile.git          // 和git有关的部分
├──── nemu                  // NEMU
│     └──── src
│           └──── main.c    // NEMU入口
├──── testcase              // 测试用例
├──── scripts               // 框架代码功能脚本，不可改动
└──── libs                  // 框架代码所使用的库，不可改动
```

PA的原理

软件

编译得到指令的序列

```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

ISA: Intel i386

```
pa2020_fall/
├──── game                  // 包含游戏相关代码
├──── include               // PA整体依赖的一些文件
│     ├──── config.h        // 一些配置用的宏
│     └──── trap.h          // trap相关定义，不可改动
├──── kernel                // 一个微型操作系统内核
├──── Makefile              // 帮助编译和执行工程的Makefile
├──── Makefile.git          // 和git有关的部分
├──── nemu                  // NEMU
│     └──── src
│           └──── main.c    // NEMU入口
├──── testcase              // 测试用例
├──── scripts               // 框架代码功能脚本，不可改动
└──── libs                  // 框架代码所使用的库，不可改动
```

PA的原理

软件

编译得到指令的序列

```
hello_world.o:     file format elf32-i386


Disassembly of section .text:

00000000 <main>:
   0:   8d 4c 24 04             lea    0x4(%esp),%ecx
   4:   83 e4 f0                and    $0xfffffff0,%esp
   7:   ff 71 fc                pushl  -0x4(%ecx)
   a:   55                      push   %ebp
   b:   89 e5                   mov    %esp,%ebp
   d:   53                      push   %ebx
   e:   51                      push   %ecx
   f:   e8 fc ff ff ff          call   10 <main+0x10>
```

ISA: Intel i386

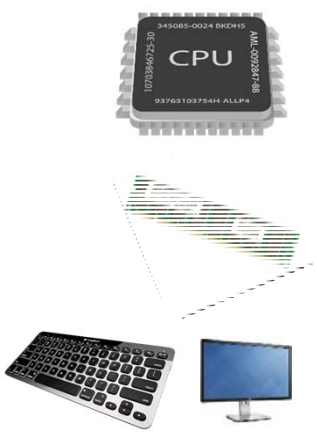
模拟实现

```
pa2020_fall/
├──── game                  // 包含游戏相关代码
├──── include               // PA整体依赖的一些文件
│     ├──── config.h        // 一些配置用的宏
│     └──── trap.h          // trap相关定义，不可改动
├──── kernel                // 一个微型操作系统内核
├──── Makefile              // 帮助编译和执行工程的Makefile
├──── Makefile.git          // 和git有关的部分
├──── nemu                  // NEMU
│     └──── src
│           └──── main.c    // NEMU入口
├──── testcase              // 测试用例
├──── scripts               // 框架代码功能脚本，不可改动
└──── libs                  // 框架代码所使用的库，不可改动
```

可以解释执行i386指令

PA的原理

PA分为四个大的阶段，每个大阶段又拆分为三个小阶段

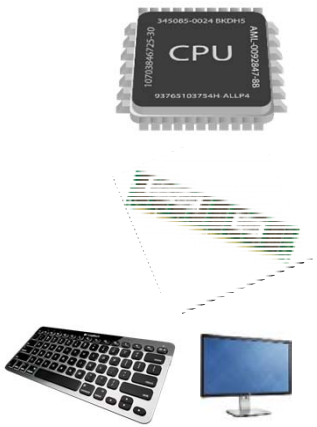


- PA 1 - 数据的表示、存取和运算
- PA 2 - 程序的执行
- PA 3 - 存储管理
- PA 4 - 异常、中断与I/O

```
pa2020_fall/
└── nemu
    └── src
        └── main.c
        └── cpu
        └── memory
        └── device
```

PA的原理

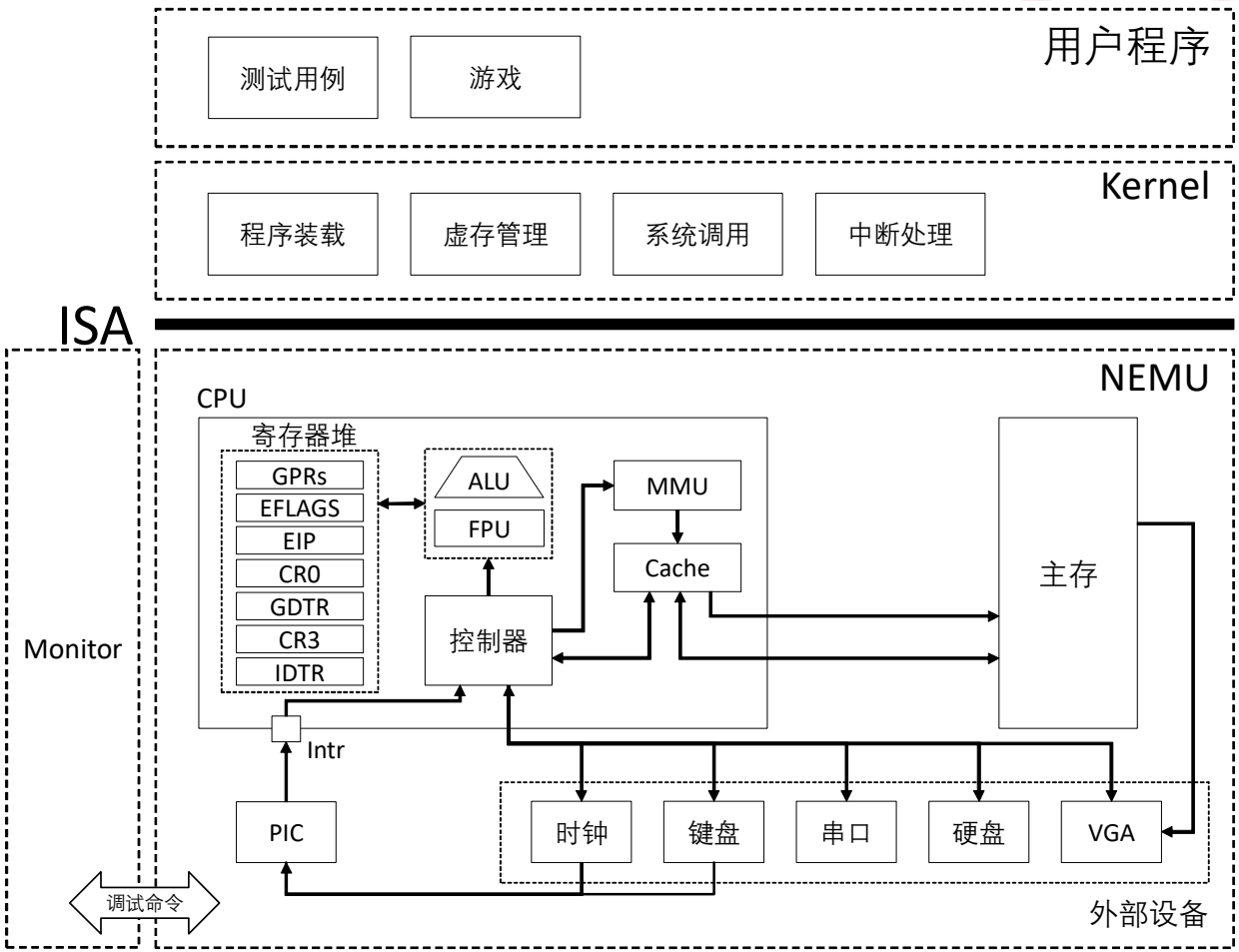
PA分为四个大的阶段，每个大阶段又拆分为三个小阶段



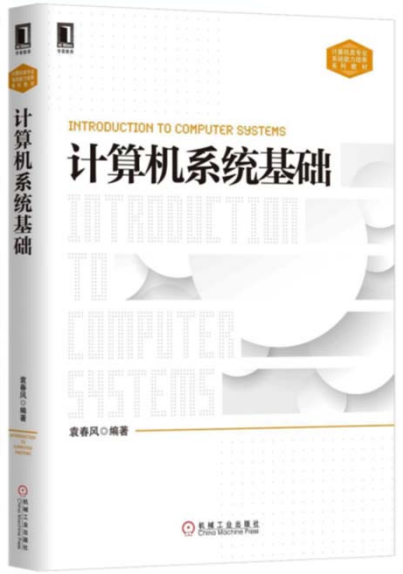
- PA 1
- PA 2
- PA 3
- PA 4


```
pa2020_fall/
└── nemu
    └── src
        ├── main.c
        ├── cpu
        ├── memory
        └── device
```

PA简介


用户程序
测试用例
游戏
Kernel
程序装载
虚存管理
系统调用
中断处理
ISA
NEMU
CPU
寄存器堆
GPRs
EFLAGS
EIP
CR0
GDTR
CR3
IDTR
ALU
FPU
MMU
Cache
控制器
主存
Monitor
Intr
PIC
时钟
键盘
串口
硬盘
VGA
调试命令
外部设备


EXPERT


INTRODUCTION TO COMPUTER SYSTEMS
计算机系统基础
袁春风 编著

完整的PA

PA 0 完成

重要事项

- 不许抄袭！必须通过在线平台完成整个实验！
- 记得时常git commit保存进度
- 一次只运行一个nemu进程，防止服务器过载
- 有问题找老师和助教

今天需要完成的内容

- 保证自己能够登陆在线实验平台
- 修改网站和SSH两处密码
- 熟悉网页的操作