

计算机系统基础
Programming Assignment

PA 0 实验环境配置与实验总览

2019年9月4日

课程信息

- PA和实验课上课时间：
 - (一班) 周三5-6节 仙II-504
 - (二班) 周五5-6节 仙II-405
- 授课团队
 - 讲师：汪亮，助教：待定，QQ群里已经有几位
- 信息
 - 课程gitlab: <http://114.212.80.187>
 - cslab系统: <http://cslabcms.nju.edu.cn/>
 - 课程QQ消息群: 708861469
 - 只是消息通知，不开展技术问答
 - 验证消息输入: 大班PA实验 学号 姓名
 - 修改群名片为: 学号 姓名
 - 讨论和问答在课程gitlab的项目主页中开展:
http://114.212.80.187/wl/pa2019_fall/issues

开始PA最重要的事情：获取Guide

http://114.212.80.187/wl/pa2019_fall_guide/

Liang Wang > pa2019_fall_guide > Details

pa2019_fall_guide Project ID: 2

☆ Star 0 Clone

No license. All rights reserved 3 Commits 1 Branch 0 Tags 8.9 MB Files

master pa2019_fall_guide History Find file

update guide Liang Wang authored 9 minutes ago 399c4a0b

README

Name	Last commit	Last update
PA_2019_fall_Guide.pdf	update guide	9 minutes ago
README.md	Initial commit	5 days ago

README.md

pa2019_fall_guide

点击这个pdf文件

开始PA最重要的事情：获取Guide

http://114.212.80.187/wl/pa2019_fall_guide/

Liang Wang > pa2019_fall_guide > Details

pa2019_fall_guide
Project ID: 2

No license. All rights reserved 3 Commits 1 Branch 0 Tags 8.9 MB Files

master pa2019_fall_guide

update guide
Liang Wang authored 9 minutes ago

README

Name	Last commit
PA_2019_fall_Guide.pdf	update guide
README.md	Initial commit

README.md

pa2019_fall_guide

点击这个pdf文件

update guide
Liang Wang authored 10 minutes ago

PA_2019_fall_Guide.pdf 5.25 MB

下载Guide

PA 2019 秋季 实验指导

南京大学《计算机系统基础课程组》 汪亮, 2019 年 8 月 22 日星期四

本教程是针对 2019 年秋季《计算机系统基础》Programming Assignment (PA) 实验的指导。开展实验的流程请参见 PA 0 的内容。其中一些关键注意事项如下所述：

注意事项：

1. 我们使用 `git` 来管理代码并接受提交，使用学号注册，不要泄露你的账号信息；若无法登陆，请及时联系助教或讲师；
2. 环境一旦安装完毕，最好不要移动自己的本地项目路径，或将代码放到别人的机器上去执行，否则可能出现无法提交的异常；
3. 框架代码会使用 `git commit` 命令来追踪整个项目开发的过程，并用于包括抄袭检查、行为分析在内的教学和科研目的，此功能不应被关闭，也禁止对相应组件进行攻击；
4. 框架代码每隔一段时间会弹出一个情绪调查问卷，用于教学质量评估和科研目的，此调研自愿参加，可以选择跳过；
5. 经常执行 `git push` 操作，可随时保存代码防止意外，同时让我们能够及时掌握项目进展情况并调整授课计划；
6. 讨论和问答在课程 `gitlab` 的项目主页中开展：http://114.212.80.187/wl/pa2019_fall/issues；
7. 严禁任何形式的抄袭或对框架代码包括 `scripts/`和 `libs/`文件夹下数据与脚本的逆向工程和攻击！
8. 我们所使用的私有 `GitLab` 服务器因不可预知的原因有崩溃的危险，因此请注意以下几点
 - a) 我们会在每天凌晨 5 点重启服务器、每周日凌晨 1 点备份服务器数据，请勿在这个时间段进行 `pull`、`push` 等与服务器通信的操作；
 - b) 你的本地仓库是安全的，当出现服务器数据与本地仓库不一致的时候，不要随意改动本地仓库内的代码和数据；出现异常时及时与教学团队联系；
 - c) 每次 `submit` 提交时，同时将生成的压缩包上传到 `cslab` 的备用提交窗口保证安全。

实验环境配置 – 虚拟机

- 安装虚拟机软件
 - Vmware
 - VirtualBox
- 下载debian操作系统iso镜像
 - 32位 (i386)
 - 最小化安装 (network install)
- 新建虚拟机并安装32位debian操作系统
 - 语言选择US_en
 - 地区选择中国并选择国内的源 (source)
 - Locale选择united states
 - 选择一个桌面环境, 推荐MATE

实验环境配置 – 配置用户和环境

- 配置客户操作系统和开发环境
 - 将用户添加到sudoer list, 不允许使用root用户做实验
 - 具体过程参见Guide
- 安装vmware-tools或virtualbox增强功能
 - 具体过程参见Guide或者网上自行搜索解决
- 为虚拟机安装必要的工具和软件

```
sudo apt-get install build-essential libreadline-dev libsdl1.2-dev vim  
git tmux dialog python python-rsa openssl
```

参见Guide PA 0和附录 C中的详细说明

实验环境配置 – 配置用户和环境

- 参考的软件版本信息如下

类型	名称	版本	查看命令
客户操作系统	Debian 10 buster	OS: Debian 10 buster Kernel: i686 Linux 4.19.0-5-686	uname -a
编译器	gcc	8.3.0	gcc -v
readline开发包	libreadline-dev	7.0-5 i386	apt list libreadline-dev
SDL开发包	libsdl1.2-dev	1.2.15+dfsg2-4 i386	apt list libsdl1.2-dev

请注意版本号： gcc 版本是否是8.x很关键， SDL库必须是1.x版本而非2.x版本

实验环境配置 – 获取PA框架代码

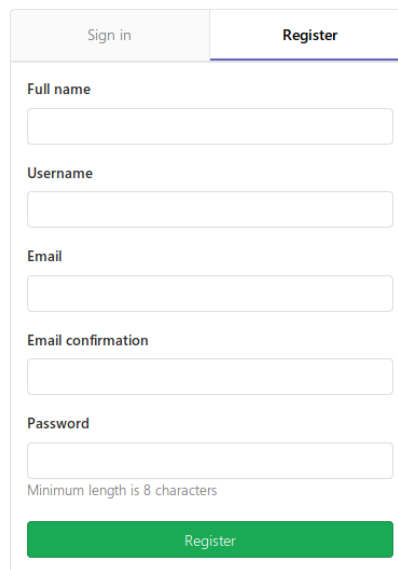
- 连接校园网并访问<http://114.212.80.187>

GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

我们会删除用户名不是学号，邮箱不是smail，或学号不在选课名单上的账号



The image shows the 'Register' tab of the GitLab registration form. It contains five input fields: 'Full name', 'Username', 'Email', 'Email confirmation', and 'Password'. Below the password field is a note 'Minimum length is 8 characters'. At the bottom is a green 'Register' button. Red arrows point from the Chinese text on the right to each of the five input fields.

学号

学号

学号@smail.nju.edu.cn

学号@smail.nju.edu.cn

设置强密码

- 成功注册后，fork框架代码、clone到本地、并执行setup
 - 具体请参见教程和现场演示

PA实验的过程和提交方式

- pa2019_fall/的组织结构

```
pa2019_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
├── kernel        // 一个微型操作系统内核
├── libs           // 库文件
├── Makefile       // 帮助编译和执行工程的Makefile
├── Makefile.git   // 和git以及提交信息有关的部分
├── nemu           // NEMU
│   └── src
│       └── main.c // NEMU入口
├── scripts        // 一些重要的脚本和数据
└── testcase       // 测试用例
```

- 不要改动以下文件

- main.c, parse_args.c, nemu_trap.c
- scripts/和libs/文件夹下的任意文件

PA实验的过程和提交方式

- 参照Guide的提示完成各个阶段
- 运行本地测试， 执行

make test_pa-阶段

- 所有的test系列目标已经列举在Makefile中

PA实验的过程和提交方式

- 提交PA实验代码

- 在pa2019_fall/目录下输入以下命令 **自动打包和上传代码**

make submit_pa-阶段

- 所有submit系列目标已经列举在Makefile中，不要改动此部分
 - 同时将压缩包上传到**cslab备用提交窗口**
- 在Guide中有对应阶段的书面小问题，在每个大阶段截止时，将该阶段所有小问题的答卷提交到**cslab系统中对应的PA课后问题提交窗口**
- PA各阶段的截止
 - 应该在规定的提交截止时间前提交代码和答卷
 - 若迟交，则会进行相应扣分惩罚
- **不允许抄袭！我们会比对今年和往年的所有代码，若发现抄袭，则对应阶段0分。不要试图破解或攻击submit相关逻辑和服务器，若发现，提交校方进行处理。**

PA实验的过程和提交方式

- 我们所使用的私有GitLab服务器因不可预知的原因有崩溃的危险，因此请注意以下几点
 - 我们会在每天凌晨5点重启服务器、每周日凌晨1点备份服务器数据，请勿在这个时间段进行pull、push等与服务器通信的操作；
 - 你的本地仓库是安全的，当出现服务器数据与本地仓库不一致的时候，不要随意改动本地仓库内的代码和数据；出现异常时及时与教学团队联系；
 - 每次submit提交时，同时将生成的压缩包上传到cslab的备用提交窗口保证安全。
- 讨论和问答在课程gitlab的项目主页中开展：
http://114.212.80.187/wl/pa2019_fall/issues

PA实验的过程和提交方式

- PA环境配置了两个追踪功能
 - 针对实验代码开发过程的追踪，必须启用，用于辅助判断抄袭和科研目的
 - 针对实验过程中的心理调查问卷，自愿参加，用于教学过程管理和科研目的
 - 相关数据仅用于教学和科研目的，若需要公开发表，则会先进行匿名处理

下面开始讲PA

PA 0 基础知识部分

- PA的目标
- PA的阶段构成
- PA的原理
- PA的成果

PA 0 基础知识部分

- PA的目标
- PA的阶段构成
- PA的原理
- PA的成果

PA的目标

- 要创建NEMU（一个简化的x86模拟器）
 - 由C语言编写
 - 以用户软件的形态运行
 - 能够执行通过交叉编译得到的i386指令集程序

PA 0 基础知识部分

- PA的目标
- PA的阶段构成
- PA的原理
- PA的成果

PA的阶段构成

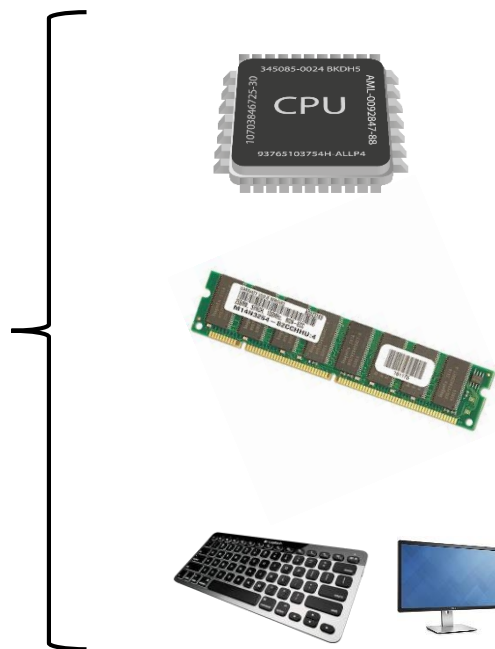
- PA 1 – 数据的表示、存取和运算
 - PA 1-1 – 数据的存储、寄存器模拟
 - PA 1-2 – 整数运算、ALU模拟
 - PA 1-3 – 浮点数运算、FPU模拟
- PA 2 – 程序的执行
 - PA 2-1 – 指令解码与执行
 - PA 2-2 – 装载程序的Loader
 - PA 2-3 – 可选任务：完善调试器
- PA 3 – 存储管理
 - 分三个阶段
- PA 4 – 异常、中断与I/O
 - 分三个阶段

PA的阶段构成

大致对应的器件



NEMU



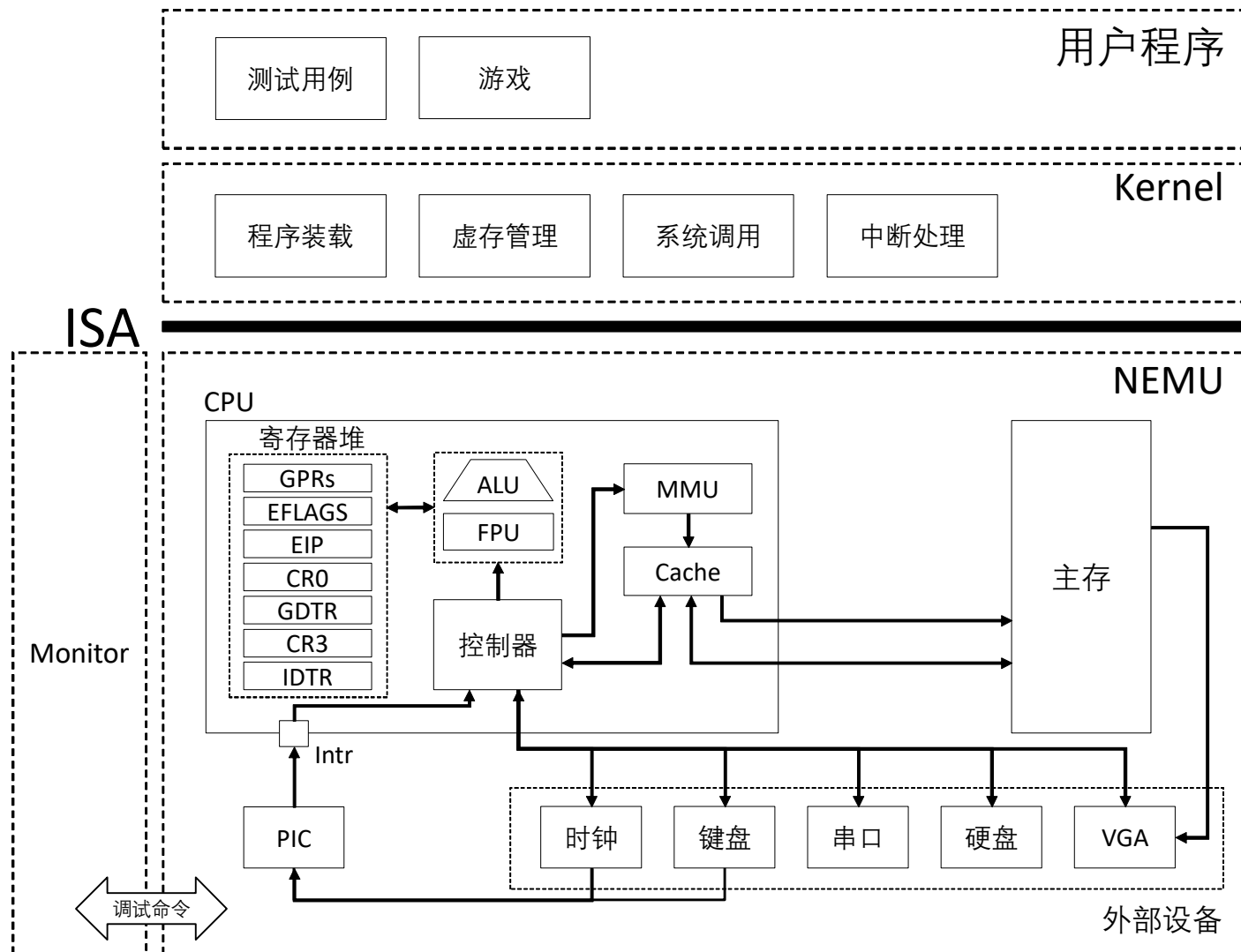
PA的四个大阶段

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的阶段构成



PA 0 基础知识部分

- PA的目标
- PA的阶段构成
- PA的原理
- PA的成果

PA的原理

- 以一个软件来模拟硬件可行吗？
 - 答案是肯定的，有好多现成的例子



虚拟机



安卓模拟器



NES模拟器

PA的原理

- 以一个软件来模拟硬件可行吗？
 - 答案是肯定的，安装模拟器后的系统栈

模拟器上执行的 程序 (OS, Hello World, Super Mario, ...)
模拟器 /虚拟机 (NEMU , Vmware, x Emulator, ...)
操作系统 (Windows, GNU/Linux, Mac, ...)
机器硬件 (Lenovo, Dell, Mac, ...)

从程序的角度看，什么是一个计算机？

PA的原理

- Hello World究竟是怎么运行起来的？

hello_world.c

```
#include<stdio.h>

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

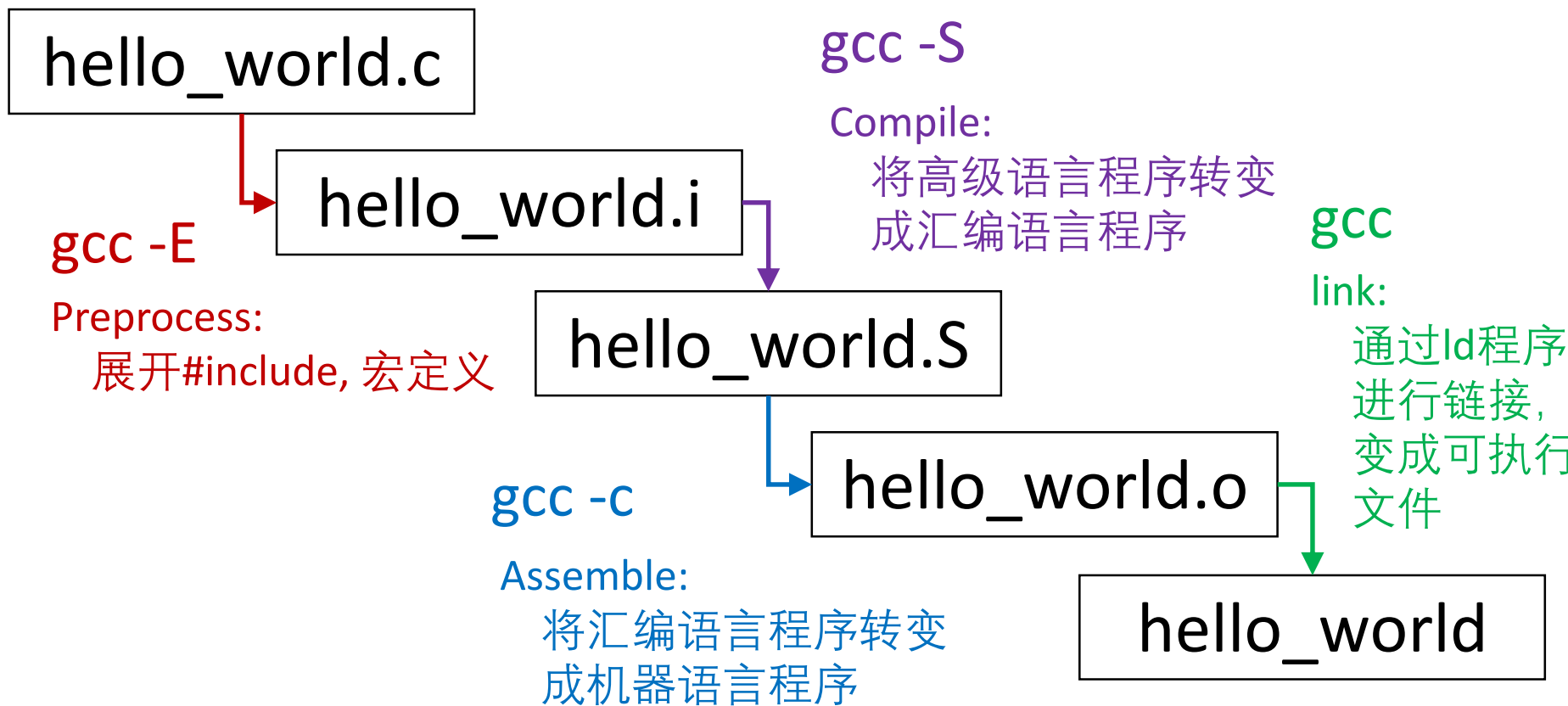
```
$ gcc -o hello_world hello_world.c
$ ./hello_world
Hello World!
```

编译
运行

PA的原理

```
$ gcc -o hello_world hello_world.c
```

一条命令执行了四个步骤



PA的原理

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

hello_world.o 查看其内容

00000000	457f	464c	0101	0001	0000	0000	0000	0000
00000010	0001	0003	0001	0000	0000	0000	0000	0000
00000020	02fc	0000	0000	0000	0034	0000	0000	0028
00000030	000f	000e	0001	0000	0007	0000	4c8d	0424
00000040	e483	fff0	fc71	8955	53e5	e851	fffc	ffff
00000050	0105	0000	8300	0cec	908d	0000	0000	8952
00000060	e8c3	fffc	ffff	c483	b810	0000	0000	658d
00000070	59f8	5d5b	618d	c3fc	6548	6c6c	206f	6f57
00000080	6c72	2164	8b00	2404	00c3	4347	3a43	2820
00000090	6544	6962	6e61	3620	332e	302e	312d	2938

偏移量

数据，两字节一组，小端

机器语言程序!

PA的原理

```
$ 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	\$0xfffffffff0,%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>

PA的原理

```
$ 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:

Disassembly of

一系列的指令
对程序而言能执行指令的就是计算机

00000000 <main>:

0: 8d 4c 24 04

4: 83 e4 f0

7: ff 71 fc

a: 55

b: 89 e5

d: 53

e: 51

f: e8 fc ff ff ff

```
lea    0x4(%esp),%ecx  
and     $0xfffffffff0,%esp  
pushl   -0x4(%ecx)  
push    %ebp  
mov     %esp,%ebp  
push    %ebx  
push    %ecx  
call    10 <main+0x10>
```

PA的原理

- 为什么可以用软件来模拟计算机执行程序？
 - 所谓程序，在形式上就是一串指令的序列
 - 硬件设计者和软件开发者约定好有哪些指令可以用（ISA）
 - 只要能够读懂指令并正确完成对应动作（运算、数据操作、输入输出等）的东西就是一台合格的计算机



PA的原理

```
$ 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
```

```
4:  83 e4 f0
```

```
7:  ff 71 fc
```

```
a:  55
```

```
b:  89 e5
```

```
d:  53
```

```
e:  51
```

```
f:  e8 fc ff ff ff
```



```
lea    0x4(%esp),%ecx
and     $0xfffffffff0,%esp
pushl   -0x4(%ecx)
push    %ebp
mov     %esp,%ebp
push    %ebx
push    %ecx
call    10 <main+0x10>
```

指令里有各种数
据移动、运算、
控制操作

PA的原理

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

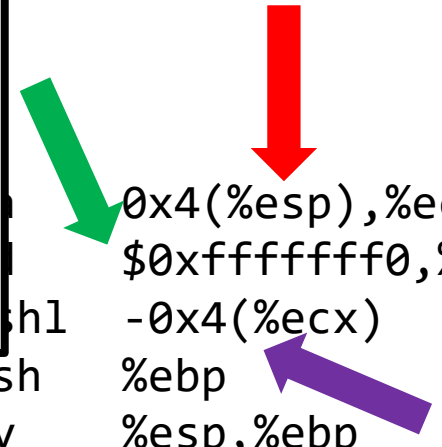
hello_world.o 反汇编其内容

hello_world.o: file format elf32-i386

Disassembly of section .text:

00000000	0:		h1	0x4(%esp),%ecx
	4:			\$0xfffffffff0,%esp
	7:			-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>	

指令中涉及寄存器、立即数、内存地址等操作对象，相对应的就有寄存器和内存这样的设备对程序可见



PA的原理

```
#include<stdio.h>

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

```
$ gcc -o hello_world hello_world.c
$ ./hello_world
Hello World!
```

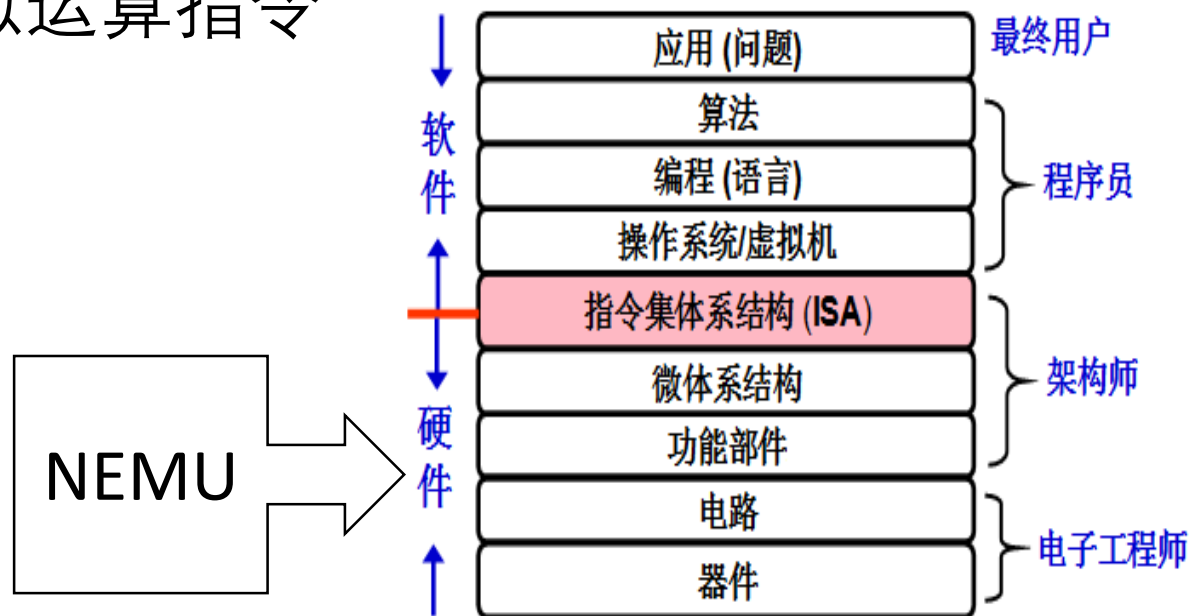


指令中也包含对键盘、屏幕等外部设备的读写指令，因此外部设备也对程序可见

PA的原理

用C语言模拟机器的功能，实现指令的解释执行，模拟实现所有对程序可见的机器组件和功能：

- 变量模拟寄存器
- 数组模拟内存
- 变量赋值模拟数据移动指令
- 运算操作模拟运算指令
- ...



PA 0 基础知识部分

- PA的目标
- PA的阶段构成
- PA的原理
- PA的成果

PA的成果



当然，最重要的是掌握了计算机系统的重要知识！

PA 0 到此结束

没有需要提交的内容