Programming Assignment Lecture II

PA1-1

Xie

xiemhemail@gmail.com

Sep 8th, 2017

The simplest computer- Turing machine

Architecture of the simplest computer

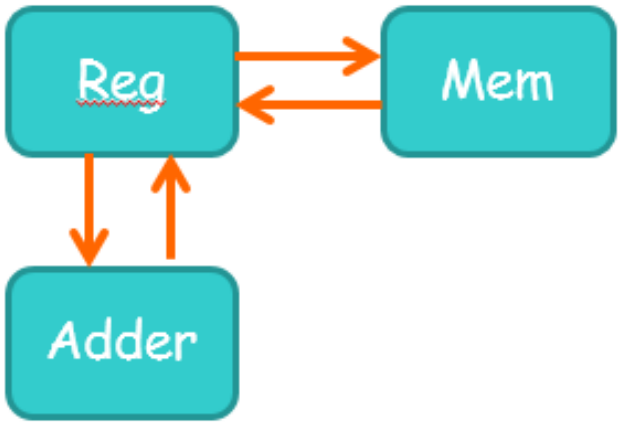
To place programs Memory

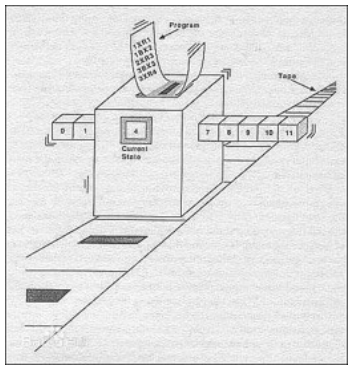
To process data Adder

To store temporary results efficiently Reg

Working mode of the simplest computer

- Fetch instruction from Mem using PC.
- Execute instruction.
- Update PC.





Programming Assignment Lecture II

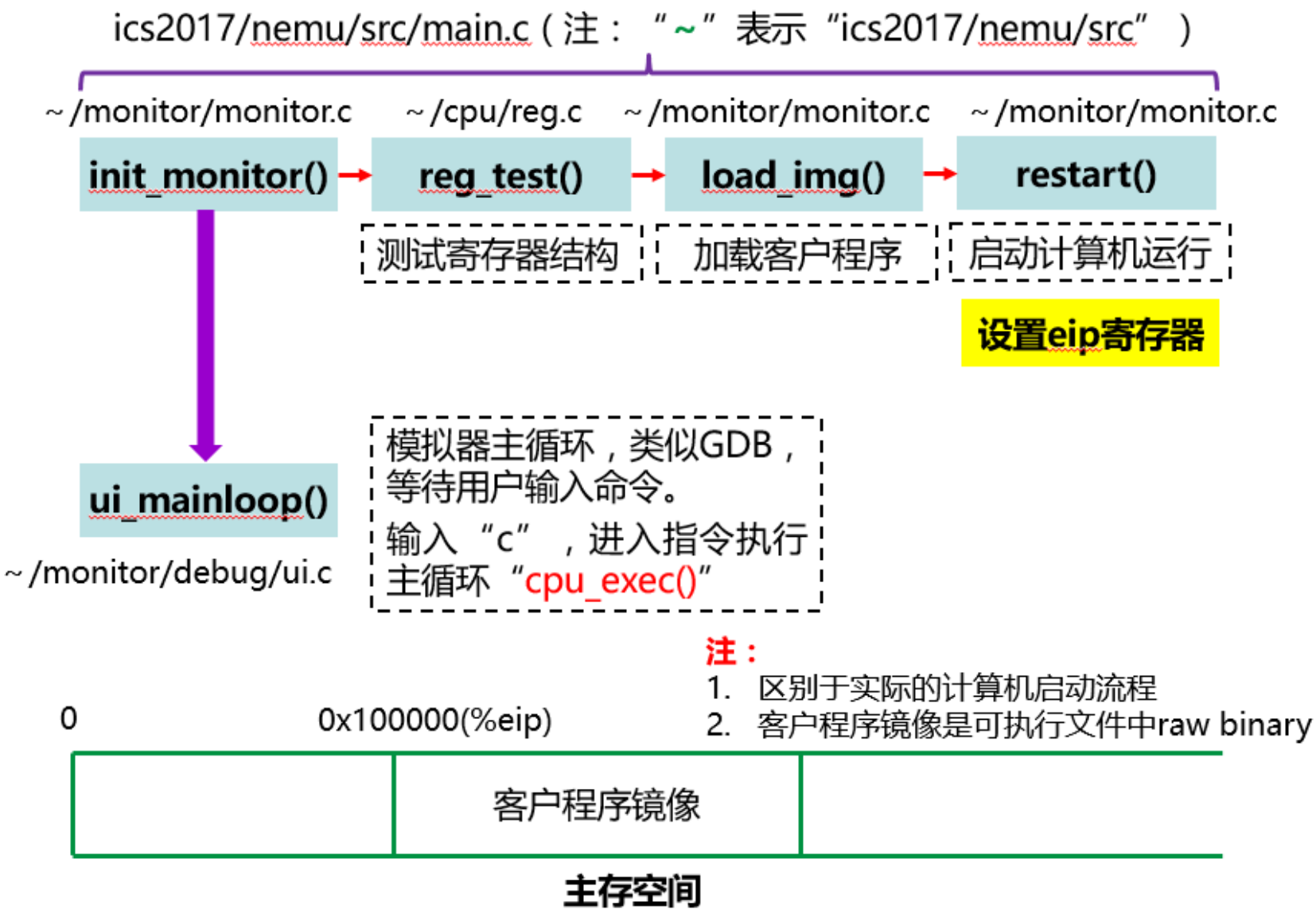
Structure of PA

```
ics2017
|---nanos-lite     # mini operating system kernel
|---navy-apps      # apps
|---nemu           # NEMU
|---nexus-am       # abstract machine
```

Structure of NEMU

```
nemu
├── include                    # 存放全局使用的
│   ├── common.h               # 公用的头文件
│   ├── cpu
│   │   ├── decode.h           # 译码相关
│   │   ├── exec.h             # 执行相关
│   │   ├── reg.h              # 寄存器结构体的
│   │   └── rtl.h              # RTL指令
│   ├── debug.h                # 一些方便调试用
│   ├── device                 # 设备相关
│   ├── macro.h                # 一些方便的宏定
│   ├── memory                 # 访问内存相关
│   ├── monitor
│   │   ├── expr.h             # 表达式求值相关
│   │   ├── monitor.h
│   │   └── watchpoint.h       # 监视点相关
│   └── nemu.h
├── Makefile                   # 指示NEMU的编译
├── Makefile.git               # git版本控制相关
├── runall.sh                  # 一键测试脚本
└── src                        # 源文件
    ├── cpu
    │   ├── decode             # 译码相关
    │   ├── exec               # 执行相关
    │   ├── intr.c             # 中断处理相关
    │   └── reg.c              # 寄存器相关
    ├── device                 # 设备相关
    ├── main.c                 # 你知道的...
    ├── memory
    │   └── memory.c           # 访问内存的接口
    ├── misc
    │   └── logo.c             # "i386"的logo
    └── monitor
        ├── cpu-exec.c         # 指令执行的主循
        ├── diff-test
```

Flow of execution



main()

nemu/src/main.c

init_monitor() Initialize monitor

ui_mainloop() Ui main loop

Tip - Ctags

Ctags is a programming tool that generates an index (or tag) file of names found in source and header files of various programming languages. Depending on the language, functions, variables, class members, macros and so on may be indexed. These tags allow definitions to be quickly and easily located by a text editor.

init_monitor()

nemu/src/monitor/monitor.c

init_log()	Initialize log file
reg_test()	Test the CPU_State struct
load_img()	Load the image to memory
restart()	Set %eip
init_*()	Do some else initialization work
welcome()	Output Welcome to NEMU!

ui_mainloop()

nemu/src/monitor/debug/ui.c

```
while(1)
{
    read the user command
    execute the user command
}
```

We already have implemented some commands

c,q,help

Question

In cmd_c(), we call the function cpu_exec(-1), why -1?

Commands in monitor

命令	格式	使用举例	说明
帮助(1)	`help`	`help`	打印命令的帮助信息
继续运行(1)	`c`	`c`	继续运行被暂停的程序
退出(1)	`q`	`q`	退出NEMU
单步执行	`si [N]`	`si 10`	让程序单步执行 `N` 条指令后暂停执行, 当 `N` 没有给出时, 缺省为 `1`
打印程序状态	`info SUBCMD`	`info r` `info w`	打印寄存器状态 打印监视点信息
表达式求值	`p EXPR`	`p $eax + 1`	求出表达式 `EXPR` 的值, `EXPR` 支持的 运算请见调试中的表达式求值小节
扫描内存(2)	`x N EXPR`	`x 10 $esp`	求出表达式 `EXPR` 的值, 将结果作为起始内存 地址, 以十六进制形式输出连续的 `N` 个4字节
设置监视点	`w EXPR`	`w *0x2000`	当表达式 `EXPR` 的值发生变化时, 暂停程序执行
删除监视点	`d N`	`d 2`	删除序号为 `N` 的监视点

备注:

- (1) 命令已实现

Infrastructure

- Improve developing efficiency.

Examples

- Makefile
- Vivado
- Google
  - Adder
  - Multiplier

Programming Assignment Lecture II

Requirements of PA1-1

nemu/include/cpu/reg.h Implementing the struct of Regs

nemu/src/monitor/debug/ui.c Parsing commands

nemu/src/monitor/debug/ui.c Implementing some commands

Struct of Regs

- nemu/include/cpu/reg.h
- The function reg_test() in nemu/src/cpu/reg.c will test your implementing.Assertion fail will be triggered if you are wrong.
- If right, you will hit good trap if you enter the command 'c' in monitor.

Tip

- Understand the structure of CPU regs.
- Understand the differences between struct and union in C language.

Parsing commands

- nemu/src/monitor/debug/ui.c ui_mainloop()
- Nothing to say.
- RTFSC,RTFM

Tip

- man readline
- man strtok
- man sscanf

Implementing some commands -1

- nemu/src/monitor/debug/ui.c
- si
- info r
- x

Question

Do you know what type the array **opcode_table** is ?

Tip

- Understand function pointer

Implementing some commands -2

si

- Understand the meaning of **cpu_exec()**.RTFSC

info r

- So easy!

x

- Try to find the interface of accessing memory.

Tip

Function **load_default_img()** in nemu/src/monitor/monitor.c will tell you whether your command **x** is right.

The end

Thanks!