Lab 3
Architecture lab, Part A
Y86 Programming

CS429
Fall 2012
Yousuk Seung

Background

- IA32 Programming
  - Chapter 3.1-3.7
  - Lectures on “Machine Programming”
  - Need to understand
    - IA32 assembly programming basics
    - Data movement
    - Logical, arithmetic operations
    - Conditional statements, control
    - Procedures
- You will **NOT** be programming in IA32 though

Background (con't)

- Y86 Architecture
  - Simplified version of IA32 instruction set
  - Lab 3,6,7 will all be in Y86
  - Chapter 4.1
- Understanding of IA32 necessary

Lab setup

- Setting up may need some time
  - Read the assignment PDF document
- Part of today's goal
  - Download and extract lab packages
  - Compile/setup the lab
  - Assemble and run a dummy Y86 program

Need Complete Y86 Codes

- Your program needs to be **complete**
  - Not a snippet of procedures
  - Should include
    - Initial setup for the stack, SP, and BP
    - Call to your procedure with arguments
    - HALT after return from your procedure

Sample Codes

- Initialization (Figure 4.7 **SUM**)
- Note that **NO** arguments are passed here
  - But you need to

```
#Execution begins at address 0
            .pos 0
Init:       irmovl Stack, %esp      # Set up stack pointer
            Irmovl Stack, %ebp      # Set up base pointer
            call Main               # Execute main program
            halt                    # Terminate program
```

Sample Codes (con't)

- Stack (Figure 4.7 **SUM**)
  - Need to reserve enough space for the stack
  - Why do we need ".pos 0x100"?
    - We will reserve **more** stack space

```
  # The stack starts here and grows to lower addresses
                .pos  0x100
stack:
```

Code Specifications

- **Important**
- Label list elements as on PDF
  - ele1, ele2, ..., ele10, ele11, ...
  - sum.ys and rsum.ys
- Ensure 4096 bytes for the stack
  - e.g.

```
# assuming your code is smaller than 1024 words
            .pos 0x2000
stack:
```

Assembling Your Codes

- Read the assignment PDF document
- Will print compile-time error messages
- Assembler will generate a **.yo** file.

```
Linux> yas sum.ys
```

Running Your Codes

- Sample output for sum.ys

```
Linux> yis sum.yo

Stopped in 33 steps at PC = 0x19.  Status 'HLT', CC Z=1 S=0 O=0
Changes to registers:
%eax:   0x00000000  0x00000cba
%edx:   0x00000000  0x00000c00
%esp:   0x00000000  0x000000fc
%ebp:   0x00000000  0x00000100

Changes to memory:
...
```

Running Your Codes

- Sample output for rsum.ys

```
Linux> yis rsum.yo

Stopped in 60 steps at PC = 0x19.  Status 'HLT', CC Z=0 S=0 O=0
Changes to registers:
%eax:    0x00000000  0x00000cba
%ecx:    0x00000000  0x0000000a
%esp:    0x00000000  0x000000fc
%ebp:    0x00000000  0x00000100

Changes to memory:
...
```

Running Your Codes

- Sample output for copy.ys

```
Linux> yis copy.yo

Stopped in 71 steps at PC = 0x29.  Status 'HLT', CC Z=1 S=0 O=0
Changes to registers:
%eax:    0x00000000  0x00000cba
%ecx:    0x00000000  0x00000c00
%esp:    0x00000000  0x000000f4
%ebp:    0x00000000  0x00000100

Changes to memory:
0x0038:  0x00000111  0x0000000a
0x003c:  0x00000222  0x000000b0
0x0040:  0x00000333  0x00000c00
...
```

Turning In

- Include your name, EID, and CSID
- Three .ys files
  - sum.ys, rsum.ys, copy.ys
- Brief report
  - doc