10220CS410001
Computer Architecture

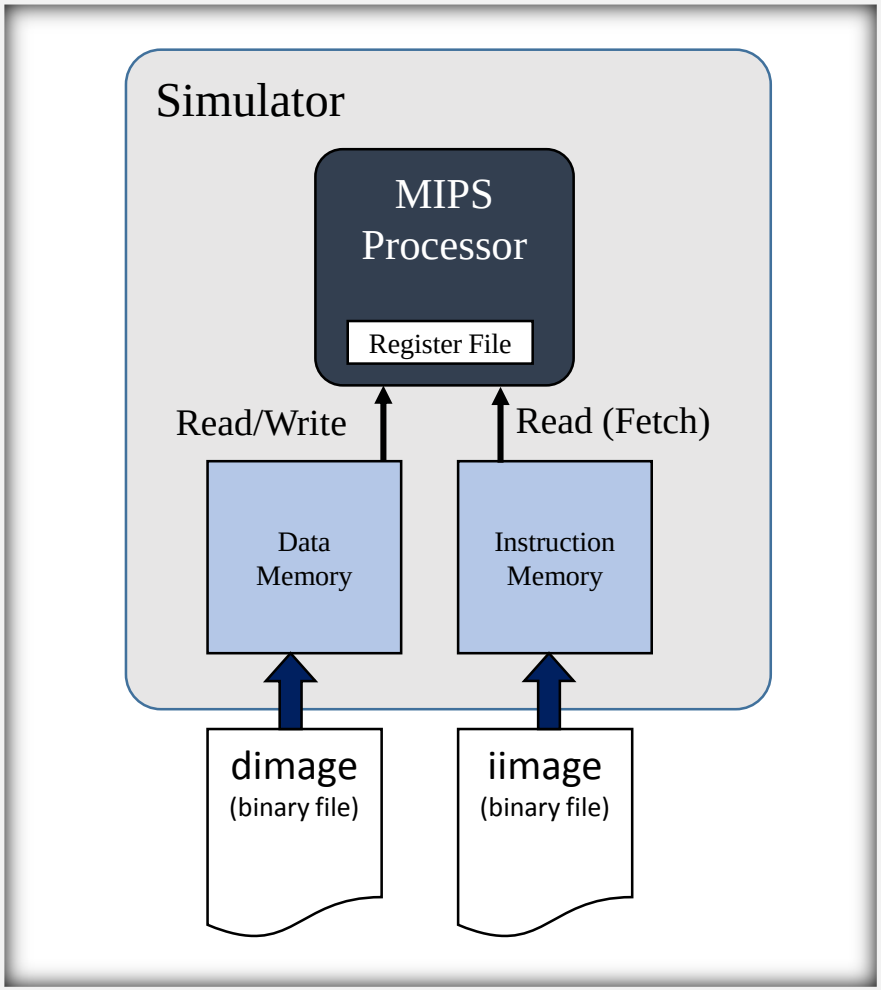
Tutorial 1

1st Project Overview

Access Server Remotely

Development Environment

Single Cycle Processor Simulator

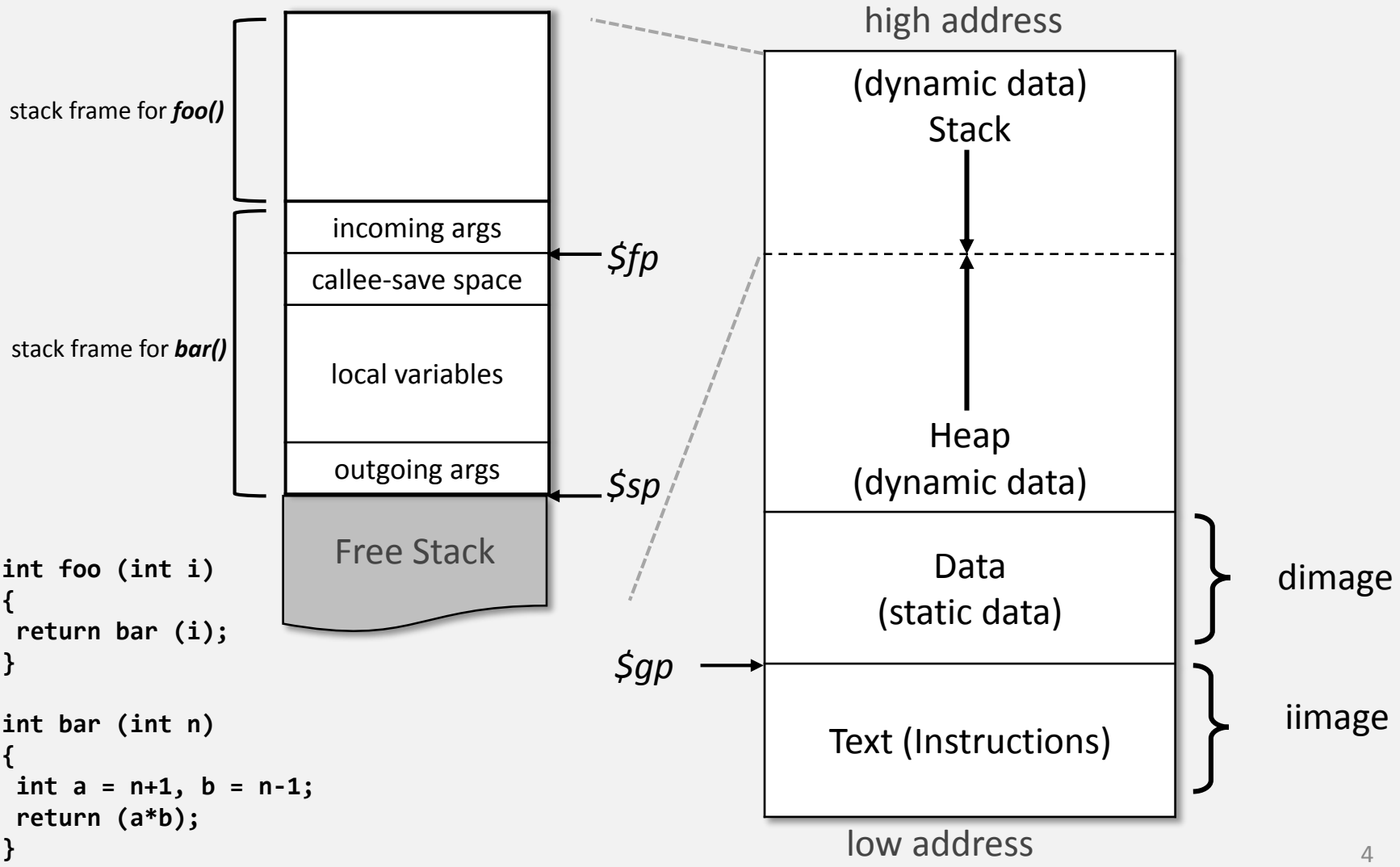


- **Simulation steps:**
  (1) fetch Instruction → (2) decoding → (3) execution(translation/emulation)
- **Fetch Instruction:**
  - **big-endian / little-endian (memory/register)**
- **Decoding**
  - **R-type**
  - **I-type**
  - **J-type**

Register Usage in MIPS ABI

	Name		Usage		Callee must preserve?
	O32	N32/N64			
$0	zero		constant 0		N/A
$1	at		assembler temporary		NO
$2,$3	v0,v1		values for function returns and expression evaluation		NO
$4-$7	a0-a3		function arguments		NO
$8-$11	t0-t3	a4-a7	temporaries	function arguments	NO
$12-$15	t4-t7	t0-t3	temporaries		NO
$24,$25	t8,t9	t8,t9	temporaries		NO
$16-$23	s0-s7		saved temporaries		YES
$26,$27	k0,k1		reserved for OS kernel		N/A
$28	gp		global pointer		YES
$29	sp		stack pointer		YES
$30	s8/fp		frame pointer		YES
$31	ra		return address		N/A

$sp, $fp, $gp




```
int foo (int i)
{
 return bar (i);
}

int bar (int n)
{
 int a = n+1, b = n-1;
 return (a*b);
}
```

Note

- objdump -d -j .text [object file]
- objdump -d -j .data [object file]
- objdump -d -j .bss [object file]

- readelf

- http://blog.linux.org.tw/~jserv/archives/002064.html

Connect to a Remote Server

Secure Shell (**SSH**) is a cryptographic network protocol

- secure data communication (sftp)
- remote command-line login
- remote command execution
- other secure network services

Client Utility

- remote command execution
  - OpenSSH
    - ssh [user_name]@[host_name/IP_address]
  - PieTTY / PuTTY
- secure data communication
  - OpenSSH
    - sftp [user_name]@[host_name/IP_address]
      - ls, cd, get
    - scp [source_user]@[source_host]:[source_file] [target_user]@[target_host]:[target_file]
  - Filezilla

iimage.bin

initial value of PC
: 0x00000207

number of words to be loaded into I memory
: 0x0000001b = 27

```
[/tmp/foo] xxd open_testcase/recur/iimage.bin
0000000: 7002 0000 1b00 0000 fcff bd23 0000 bfaf  p..........#....
0000010: 0000 048c a900 000c 0400 02ac 0000 bf8f  ................
0000020: 0400 bd23 0400 038c ffff ffff ffff ffff  ...#............
0000030: ffff ffff ffff ffff ffff ffff 0200 8828  ...............(
0000040: 0200 0011 2510 8000 0800 e003 f8ff bd23  ....%..........#
0000050: 0400 bfaf 0000 a4af ffff 8420 a900 000c  .......... ....
0000060: 0000 a48f 0400 bf8f 0800 bd23 2010 4400  ...........# .D.
0000070: 0800 e003
```

```
addi $sp, $sp, -4   # i:0x270
        sw   $ra, 0($sp)    # push $ra
        lw   $a0, 0($0)     # $a0 = n
        jal  sumToN
        sw   $v0, 4($0)     # store ans, i:0x280
        lw   $ra, 0($sp)    # pop $ra
        addi $sp, $sp, 4
        lw   $v1, 4($0)
        halt
        halt
        halt
        halt
        halt
sumToN: slti $t0, $a0, 2  # whether in base cases,
                                :0x2A4
        beq  $t0, $0, recur
        or   $v0, $a0, $0
        jr   $ra            #03e00008
 recur: addi $sp, $sp, -8
        sw   $ra, 4($sp)    # push $ra
        sw   $a0, 0($sp)    # push old $a0
        addi $a0, $a0, -1   # new $a0
        jal  sumToN
        lw   $a0, 0($sp)    # pop old $a0, i:0x2C8
        lw   $ra, 4($sp)    # pop $ra
        addi $sp, $sp, 8
        add  $v0, $v0, $a0
        jr   $ra            #03e00008
```

dimage.bin

initial value of $sp
:0x00000300

```
[/tmp/foo] xxd open_testcase/recur/dimage.bin
0000000: 0003 0000 0200 0000 0300 0000 2143 6587  ............!Ce.
```

0x00000002 words to be loaded into D memory

Note

- xxd [object file]
- hexdump [object file]
- od [object file]

- vim –b [object file]
  - %!xxd
  - %!xxd -p
  - %!xxd –r

Project 1 Directory Structure

```
> tree archi00/
archi00/
├── archi00_report.pdf
├── simulator
│   ├── Makefile
│   ├── README
│   ├── XXX.c
│   └── XXX.h
└── testcase
```

Note

- cd [directory]
  - cd –
  - cd ..
  - cd ~
- mkdir [directory]
- mv [source_file] [target_file]
- cp [source_file] [target_file]
- passwd

.tar.gz

- compress: tar zcxvf archi00.tar.gz archi00/
- uncompress: tar zxvf archi00.tar.gz

.tar

- archive: tar cvf [file_name]. tar DirName
- extract: tar xvf [file_name]. tar

.gz

- compress: gzip [file_name]
- uncompress: gunzip [file_name].gz
- uncompress: gzip -d [file_name]. gz

Integrated Development Environment (IDE)

- **Editor**
  - Vim – VI improvement
  - nano
  - Notepad++
  - Sublime
- **Compiler**
  - GCC: GNU Compiler Collection
- **Build Automation Utility**
  - GNU make
- **Debugger**
  - GDB: GNU Debugger
- **Version Control System**
  - git: unpleasant person
  - SVN
- **Source Code Bug & Quality Checker**
  - cpplint
  - pylint

為什麼要學 VIM

- 它是一個古老、強大、普遍的編輯器
- VIM其實沒那麼好用，但是卻又不得不用
  - 工作站有時只提供你一個純文字介面
  - 寫 C#、Android App 時用VIM不會比較有效率
  - 當你是第一個環境開發者、沒有IDE怎麼辦???
- 會了VIM不會比較厲害，但是不會就遜掉了

VIM

Movement

- h: move left
- j: move down
- k: move up
- l: move right
- w: move one word forward
- b: move one word backword
- [#n][h | j | k | l|w|b]: moves *#n* characters/word
- gg: move to first line
- G: move to last line
- ^/home: move to begin of line
- $/end: move to end of line
- :[#n]: move to [#n]-st line

VIM

Copy(Yank), Paste, Undo, Repeat

- yy: yank current line
- [#n]yy: yank current [#n] linew
- yaw: yank current word
- y$: yank until end of line
- dd: delete current line
- [#n]dd: delete current [#n] linew
- daw: delter current word
- d$: delete until end of line
- p: paste
- u: undo
- .: repeat

VIM

Saving, Quitting

- :w: Save
- :q: Quit
- :x: Save & Quit
- :wq: Save & Quit
- :q!: Quit Vim without saving

- gg=G: automatically fix indentation for the entire file

VIM

Mode

- normal [*ESC*]
  - For navigation and manipulation of text. This is the mode that vim will usually start in, which you can usually get back to with ESC.
- insert [*i*]
  - For inserting new text.
- visual [*v*]
  - For navigation and manipulation of text selections, this mode allows you to perform most normal commands, and a few extra commands, on selected text.
- command-line
  - For entering editor commands
- select
- Ex-mode

Note

- :set nu
- .vimrc
- .vim/

GNU Compiler

gcc, g++

```c
#include "stdio.h“
int main(void) {
    printf("Hello World\n");
    return 0;
}
```

- The standard way to compile this program is with the command:
  - `gcc hello.C -o hello`
- Alternatively, the above program could be compiled using the following two commands:
  - `gcc -c hello.C`

    compiles hello.C into a machine code file named "hello.o"
  - gcc hello.o –o hello

    links hello.o with some system libraries to produce the final program (executable file) "hello"

GNU Compiler

Frequently used compilation options

- Enables all compiler's warning messages
  - gcc -Wall hello.C -o hello
- Optimization
  - gcc -o0 hello.c -o hello
  - gcc -o1 hello.c -o hello
  - gcc -o2 hello.c -o hello
  - gcc -o3 hello.c -o hello
  - gcc -os hello.c -o hello
- Add the directory of header files
  - gcc –I”path/of/header/file” hello.c -o hello
- Add the directory of library
  - gcc –L”path/of/library” hello.c -o hello

Note

- clang
- clang++
- llvm-gcc
- llvm-g++

GNU Make

Make

- A utility that automatically builds executable programs and libraries from source code.

Make-Files (***Makefile***)

- Contains a set of rules used to build an application.

Makefile Example

The first rule seen by make is used as the ***default rule***.

A *rule* consists of three parts: ***target***, ***prerequisite(s)***, ***command(s)***

```
TARGET=hello_world

$(TARGET): hello_world.o
    gcc hello_world.o -o $(TARGET)

hello_world.o: hello_world.c
    gcc -c hello_world.c

clean:
    rm -f *.o $(TARGET)
```

The *target* is the file or thing that must be made.
The *prerequisites* or *dependents* are those files that must exist before the target can be successfully created.
The *commands* are those shell commands that will create the target from the prerequisites.

為什麼要會 GDB

- 你是不是常常用 print 大法 !!!
  - 每次都要重新 Compile
  - 除錯完還要把所有的「printf(”Bug is here???”)」給刪掉
- 我想要知道程式中連 print 都看不到的東西
  - stack、heap、$PC ……
- 它還可以
  - 動態的改變記憶體、暫存器、參數… 的內容，或是程式執行的路徑
  - ……

GNU Debugger

gdb

- file [file_name]
- list [#start], [#end]
- break #line
- break function
- delete [#break_point]
- info break
- run
- continue
- step
- next
- print
- print/x
- p $pc
- display/i $pc

Note

- http://www.cis.nctu.edu.tw/~is93007/acd.htm

為什麼需要版本控制

- 改錯程式、誤刪檔案不用怕
- 脫離「/*程式碼*/」的苦海
- 適合團隊合作開發
  - 還可以找到是誰把程式碼改爆了
- 分支! 學會不用再資料夾管理版本
  - ver1/、ver2/、ver3/、.....、final/、real_final/、last_final/

git

init

```
[/tmp/foo] cd project_1/archi00/
[/tmp/foo] git init .
[/tmp/foo] git add .
[/tmp/foo] git commit –m "initial commit"
```

git

.git

```
[/tmp/foo] $ ls –lah
total 12K
drwxr-xr-x+ 1  Kyle None 0 Feb 17 11:46 .
drwxrwxrwt+ 1  Kyle root 8.0K Feb 17 11:46 ..
drwxr-xr-x+ 1  Kyle None 4.0K Feb 17 11:19 .git
-rw-r--r--  1  Kyle None 0 Feb 17 11:46 Hello.txt
-rw-r--r--  1  Klye None 0 Feb 17 11:46 README
```

- .git/:
  usually is a repository

git

add file

```
[/tmp/foo] $ git status
# On branch master
# Changed but not updated:
#     (use "git add <file>..." to update what will be committed)
#     (use "git checkout -- <file>..." to discard changes in working directory)
#           modified: README
#
# Untracked files:
#     (use "git add <file>..." to include in what will be committed)
#           Hello
no changes added to commit (use "git add" and/or "git commit -a")
```

git

add file

```
[/tmp/foo] $ git add .
[/tmp/foo] $ git status
# On branch master
# Changes to be committed:
#     (use "git reset HEAD <file>..." to unstage)
#           new file: Hello
#           modified: README
```

git

Commit Changes to The Repository

```
[/tmp/foo] $ git add .
[/tmp/foo] $ git status
# On branch master
# Changes to be committed:
#     (use "git reset HEAD <file>..." to unstage)
#         new file: Hello
#         modified: README
[/tmp/foo] $ git commit -m 'add hello, refactor README‘
[master 356bbef] add hello, refactor README
  1 files changed, 1 insertions(+), 0 deletions(-)
  create mode 100644 Hello
[/tmp/foo] $ git status
# On branch master
nothing to commit (working directory clean)
```

git

History

```
[/tmp/git] $ git log
commit 5673d695fcce217b26d1a5956c1184ff62dc74f1
Author: Junio C Hamano <gitster@pobox.com>
Date: Wed Feb 16 14:33:22 2011 -0800

   Merge branch 'maint'
   * maint:
     parse_tag_buffer(): do not prefixcmp() out of range

commit 759e84f07fd0fba2f3466b11b74146173d42cb6b
Author: Junio C Hamano <gitster@pobox.com>
Date: Wed Feb 16 14:33:11 2011 -0800

   Merge branch 'maint-1.7.3' into maint
   * maint-1.7.3:
```

git

- git checkout HEAD hello.c
- git checkout (HEAD^) hello.c
- git checkout xxxx hello.c
- git rm
- git mv
- git diff

Note

- .ignore
- GitHub
  - https://github.com/
- git with Dropbox
- http://try.github.io/

Cpplint

- The ***cpplint*** is a tool that reads a source file and identifies many style errors.
- **Google C++ Style Guide**
  - http://google-styleguide.googlecode.com/svn/trunk/cppguide.xml

➢ Cpplint [file_name]