政大資科系

作業系統

Operating System

廖峻鋒

cfliao@nccu.edu.tw

- **轉達系辦重要通知:**
  - **如果有要加簽系上必修課，且為資科系、所、學士班或雙主修的同學，請透過加退選機制直接加簽，系統會優先對您進行遞補。**
  - **人工加簽**
    - **授課老師送出的加簽單中，若有資科系、所、學士班或雙主修的同學，系辦不會受理。**
    - 2021/9/16 **系務會議: 只受理系統印出之加簽單**
  - **如果有任何問題，請直接詢問系辦課務助教周小姐，感謝!**

加選意願表

- https://forms.gle/b5TQ8f8429wQqytD9

Operating System

前言: 從碼農到碼神

Chun-Feng Liao

廖峻鋒

Department of Computer Science

National Chengchi University

如何精進程式開發能力

- 「學會某個程式語言」不等於能建構完整系統 (Thomas 03)
  - 以學習語文為例，我們無法期待學完必要的單字和文法，就可以順利寫出一本小說
  - 學完程式語言所有語法後，就能寫出一個系統?
- 之後還要…
  1. 克服「撞牆者症侯群」
  2. 鍛練程式識讀(Program Comprehension)能力
  3. 了解平台(軟、硬體)，培養Mechanical Sympathy
  4. 學習軟體工程
     - 高品質、大規模軟體系統設計 (軟體模組、軟體架構、軟體樣式)

第一步: 克服撞牆期症侯群

(Sonmez 17)

- 何時
  - 學習一個程式語言時，在初步理解大部份語法，也能了解一些簡單的範例，甚至於能運用語言的核心函式庫開始解決一些問題
- 症狀
  - 學習者會「感覺無法輕鬆活用所學的程式語言」
    - 覺得可以理解一切運作方式
    - 但不知道如何整合，寫出一個(滿足自己想要的)完整應用程式
    - 就像是學習說出第二外語時，有點卡住那種感覺
  - 初學者在此階段感到非常挫折，認為永遠無法真正學會

第一步: 克服撞牆期症侯群 (2)

- 處方
  - 看程式碼，而且是一行一行看
  - 確定自已完全了解每一行的敍述在做什麼
  - 遇到部份程式無法理解，隨時查詢釐清任何語意上的困惑
- 理由
  - 不了解個別單字或片語的意義，就無法了解句子; 如果不了解每個句子，就無法精確了解文章要傳達的概念
  - 跳過這個步驟，就沒機會徹底了解程式碼「為什麼」要這樣寫的原因，或是說「它們為何要這樣被組合在一起」的原因
    - 了解組件為何這樣放在一起，是學習者成長為具開發系統能力過程不可省略的

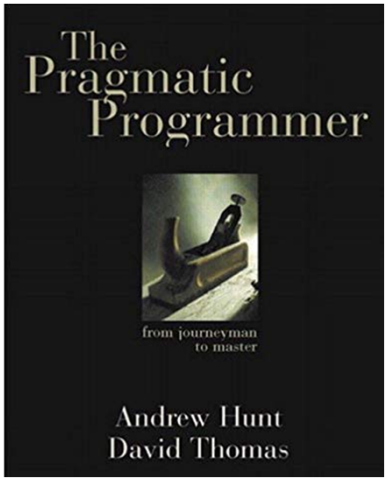
第二步: 鍛練程式識讀能力 (Program Comprehension)

Dave Thomas: Eclipse IDE創作者、The Pragmatic Programmer作者

- Dave Thomas: **程式碼識讀是寫作優質程式碼的前提**
  - **目前大部份程式設計教學忽視識讀能力的重要性，這是「悲劇(tragedy)」**
  - **理由**
    - **不藉由識讀程式碼加以修改重用，會花費時間重覆寫作功能相同程式**
    - **無法從他人傑出設計或致命錯誤中學習**

傳統程式教育並不重視程式碼識讀，其中一個主要原因是早期許多軟體公司將原始碼視為重要機密資產，因此具有特定功能的好的原始碼素材不易取得。

當今，大部份軟體公司已經採取開源策略，大量的程式碼可以無償在開源儲存庫如Github中取得，而這些公司也都提供了很好的工具，可以讀者直接線上閱讀與註解或下傳到本地端試著修改、建構與執行。



鍛練程式識讀能力

- 定義
  - 不只單純閱讀並理解，還包含理解程式碼之後，內化模仿或修改為合用，甚至於在其上新增功能的過程
- 閱讀是優質寫作的基礎
  - 藉由觀摩專業人員寫作的程式碼，能學到
    - 解決同一個問題較好的做法
    - 學習較佳的編程風格，讓程式碼可讀性更高
    - 訓練運用多種角度看待同一個問題

第三步: 了解平台: Mechanical Sympathy

Mechanical Sympathy

Mechanical sympathy is when you use a tool or system with an understanding of how it operates best.

"You don't have to be an engineer to be be a racing driver, but you do have to have Mechanical Sympathy." Jackie Stewart, racing driver

When you understand how a system is designed to be used, you can align with the design to gain optimal performance. For example, if you know that a certain type of memory is more efficient when addresses are multiples of a factor, you can optimize your performance by using data structure alignment.

傑奇·史都華

體育評論員

約翰·楊·「傑奇」·史都華爵士，OBE，蘇格蘭F1車手，有「飛行蘇格蘭人」之稱。在1965年至1973年之間，他共贏得3座世界冠軍獎盃。2009年，在著名記者凱文·伊森所列的F1史上最偉大的50位賽車手中他排名第五。伊森這樣寫道：「他不僅是一位偉大的車手，更是賽車界一個偉大的靈魂人物。」 維基百科

案例一：想一想

- 為什麼主流語言的陣列，索引都要從0開始算?
  - A[0] → A陣列的第一個元素

案例一: 想一想

- 為什麼主流語言的陣列，索引都要從0開始算?
  - A[0] → A陣列的第一個元素

索引從1開始算
一格的資料大小 = 4bytes(Why?)

&A[1] = 200 // 第一格的編號
&A[2] = 204
&A[3] = 208
…
// 第n格的編號
&A[n] = &A[1] + 4*(n-1)
= &A[1] + ((n-1) << 2)

索引從0開始算
一格的資料大小= 4bytes

&A[0] = 200
&A[1] = 204
&A[2] = 208

&A[n] = &A[0] + 4*n
&A[n] = &A[0] + (n<<2)

觀察1: 每次陣列存取，會差一次加法指令
觀察2: shift和乘法，效能是不同檔次

案例二

- 在Raspberry Pi上執行同一支程式，有時可以跑，有時出現Null錯誤，為什麼?

案例三

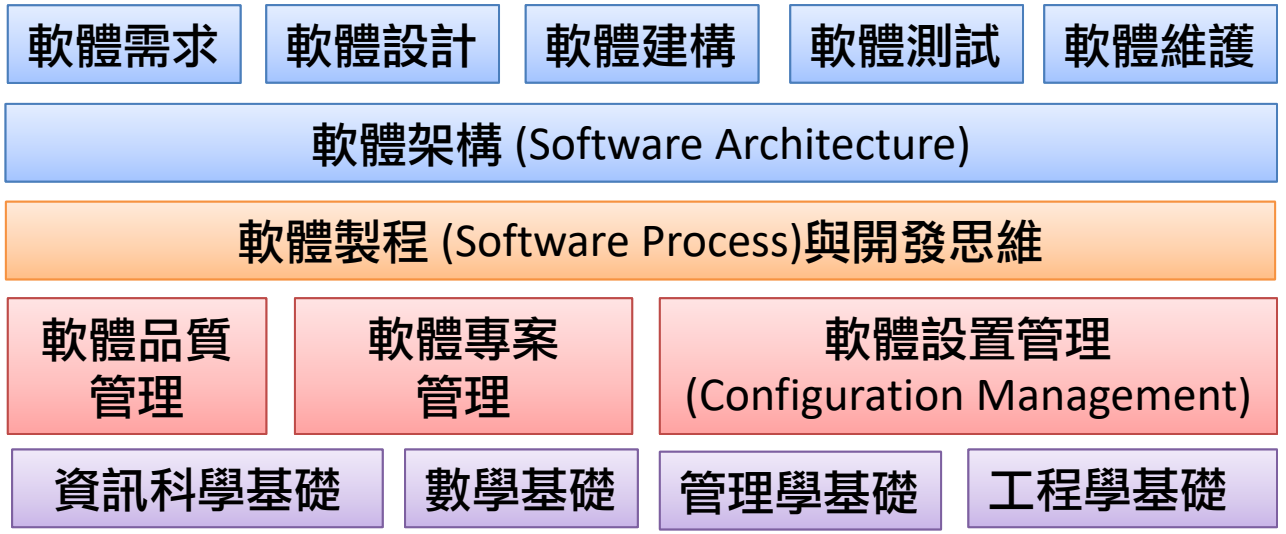
- Kafka速度快的秘密

Memory

The normal mode of operation for a Kafka consumer is reading from the end of the partitions, where the consumer is caught up and lagging behind the producers very little, if at all. In this situation, the messages the consumer is reading are optimally stored in the system's page cache, resulting in faster reads than if the broker has to reread the messages from disk. Therefore, having more memory available to the system for page cache will improve the performance of consumer clients.

Kafka itself does not need much heap memory configured for the Java Virtual Machine (JVM). Even a broker that is handling 150,000 messages per second and a data rate of 200 megabits per second can run with a 5 GB heap. The rest of the system memory will be used by the page cache and will benefit Kafka by allowing the system to cache log segments in use. This is the main reason it is not recommended to have Kafka colocated on a system with any other significant application, as it will have to share the use of the page cache. This will decrease the consumer performance for Kafka.

第四步: 軟體工程的專業知識



開發思維: 設計建構軟體的思考方式，例如: 結構化, 物件導向, 剖面導向, 函式導向...

Operating System

Course Introduction

Chun-Feng Liao

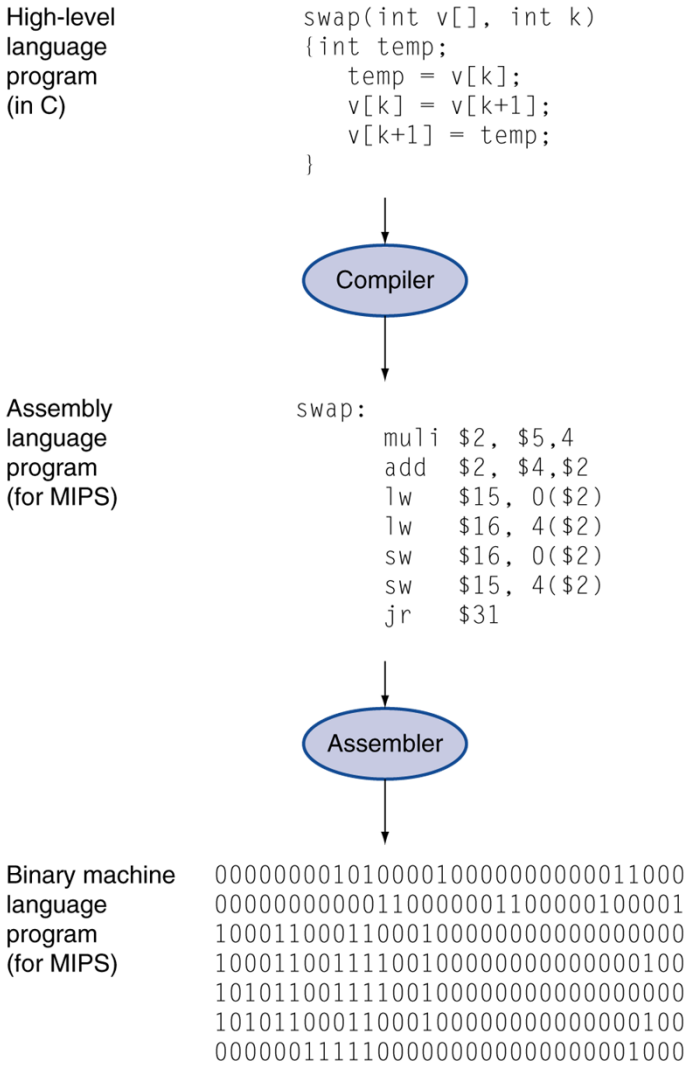
廖峻鋒

Department of Computer Science

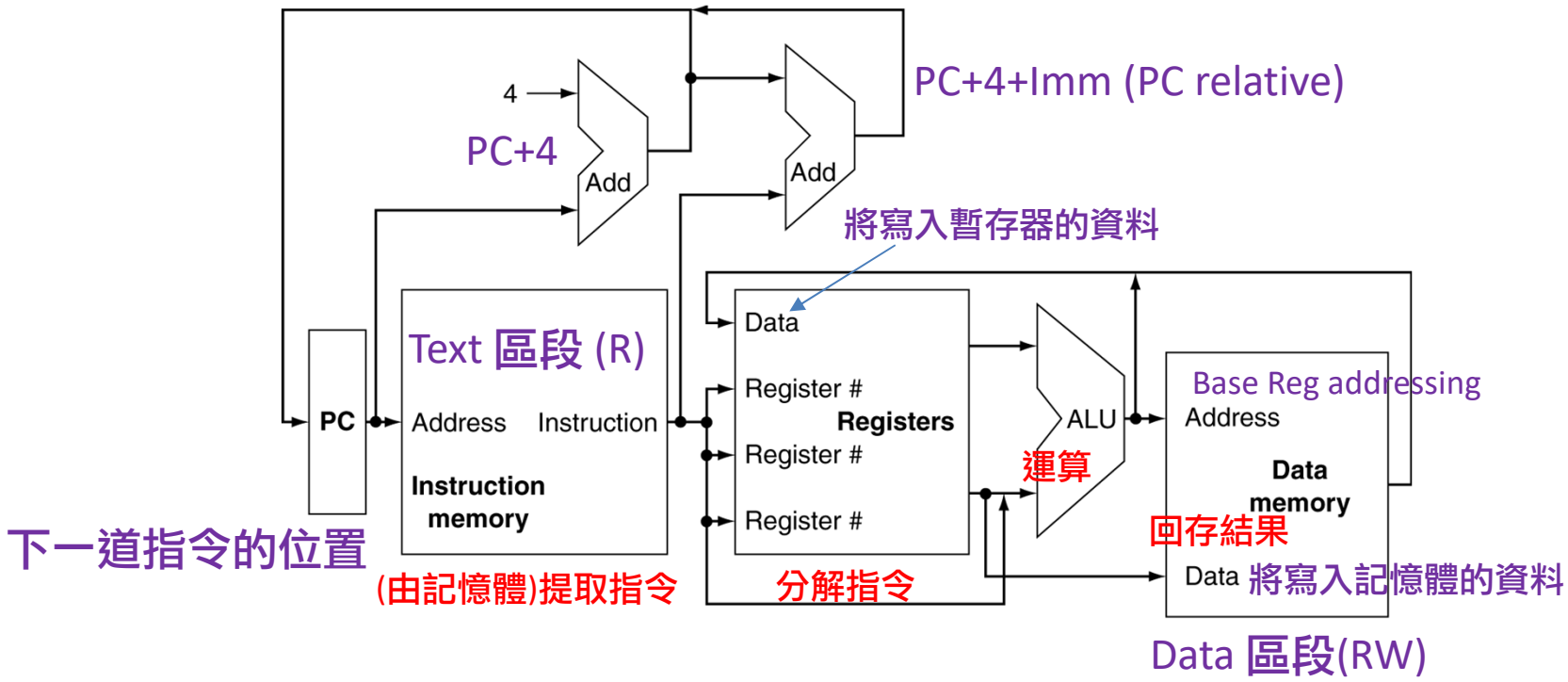
National Chengchi University

Levels of Program Code

- High-level language
  - Closer to problem domain
  - For productivity and portability
- Assembly language
  - Textual representation of instructions
- Hardware representation
  - Binary digits (bits)
  - Encoded instructions and data

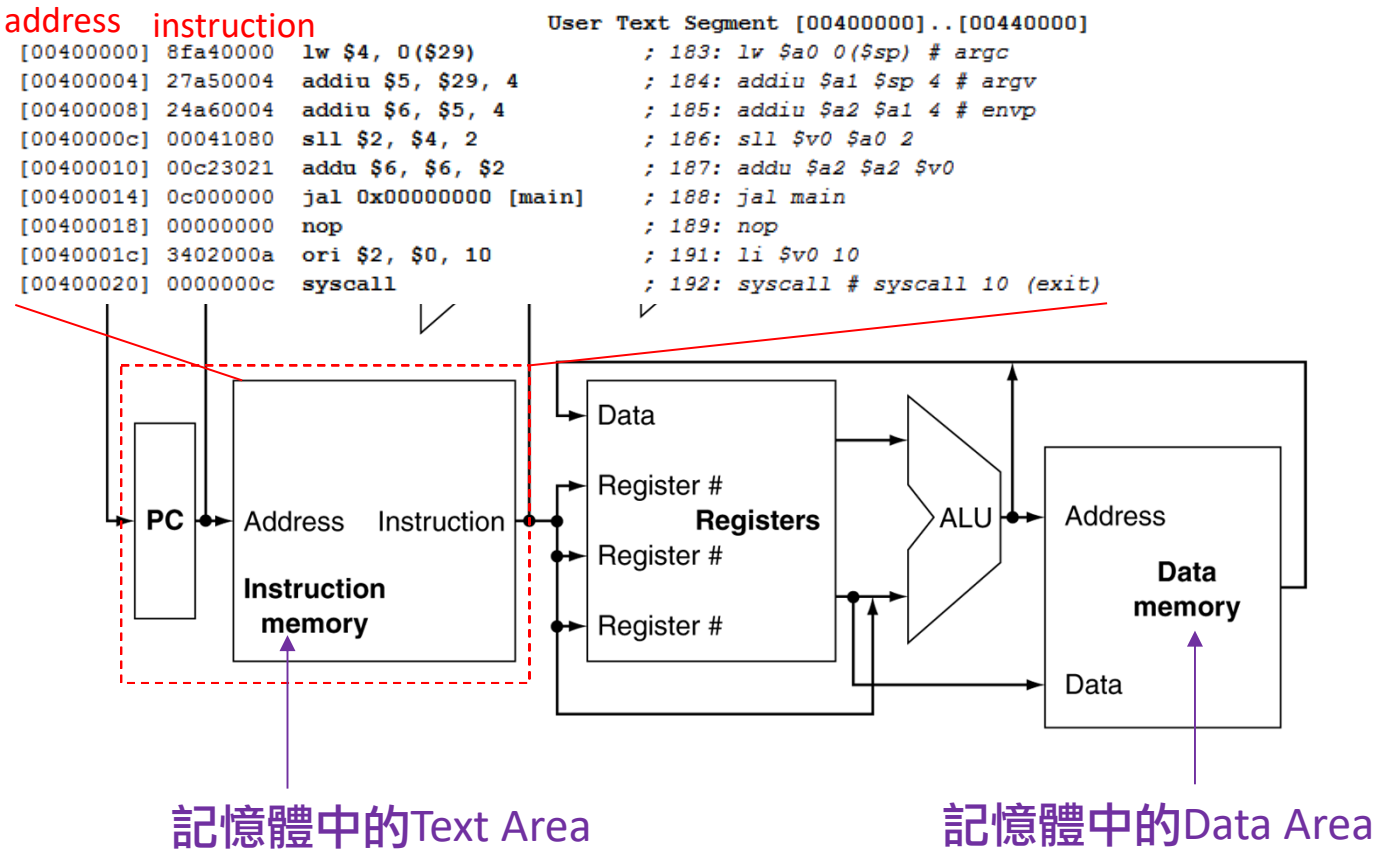


簡化的CPU架構


4
PC+4
Add
Add
PC+4+Imm (PC relative)
將寫入暫存器的資料
PC
Text 區段 (R)
Address
Instruction
Instruction memory
Data
Register #
Registers
Register #
Register #
ALU
運算
Base Reg addressing
Address
Data memory
回存結果
Data
將寫入記憶體的資料
下一道指令的位置
(由記憶體)提取指令
分解指令
Data 區段(RW)

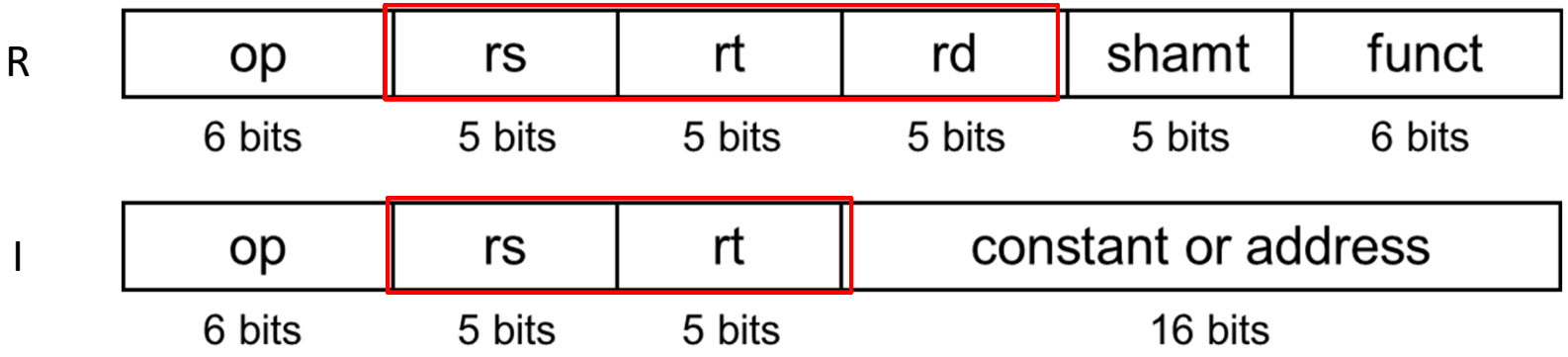
執行指令的流程

- 提取指令(Instruction Fetch, IF)
  - 依PC到instruction memory (也就是text區段)提取指令



執行指令的流程

- **讀取暫存器內容**(Instruction Decode, ID)
  - Register numbers → read registers

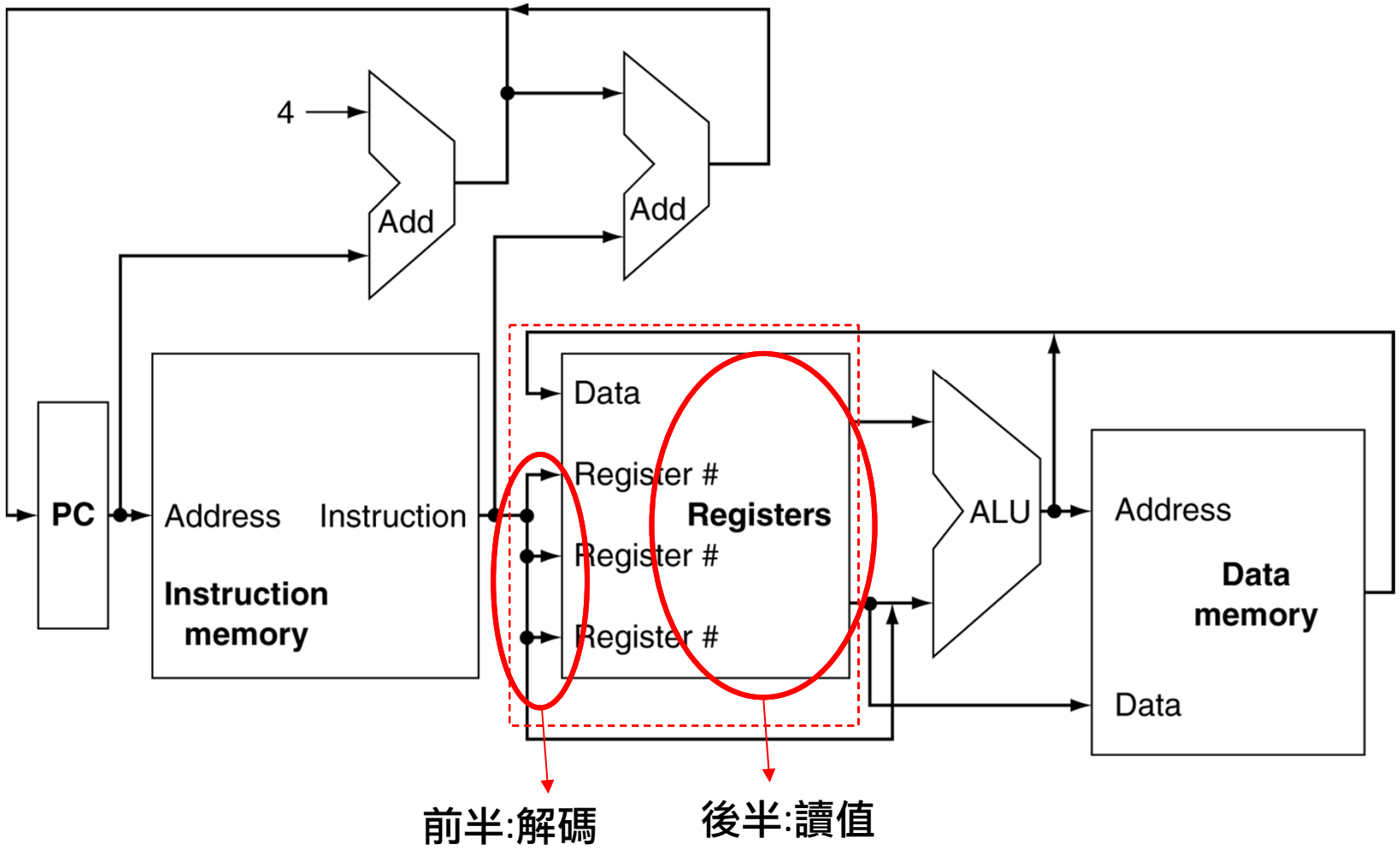


以R-type為例，要讀取rs,rt內容
以 記憶體存取類(I-type)的lw為例，要讀取rs (base reg)內容 ;

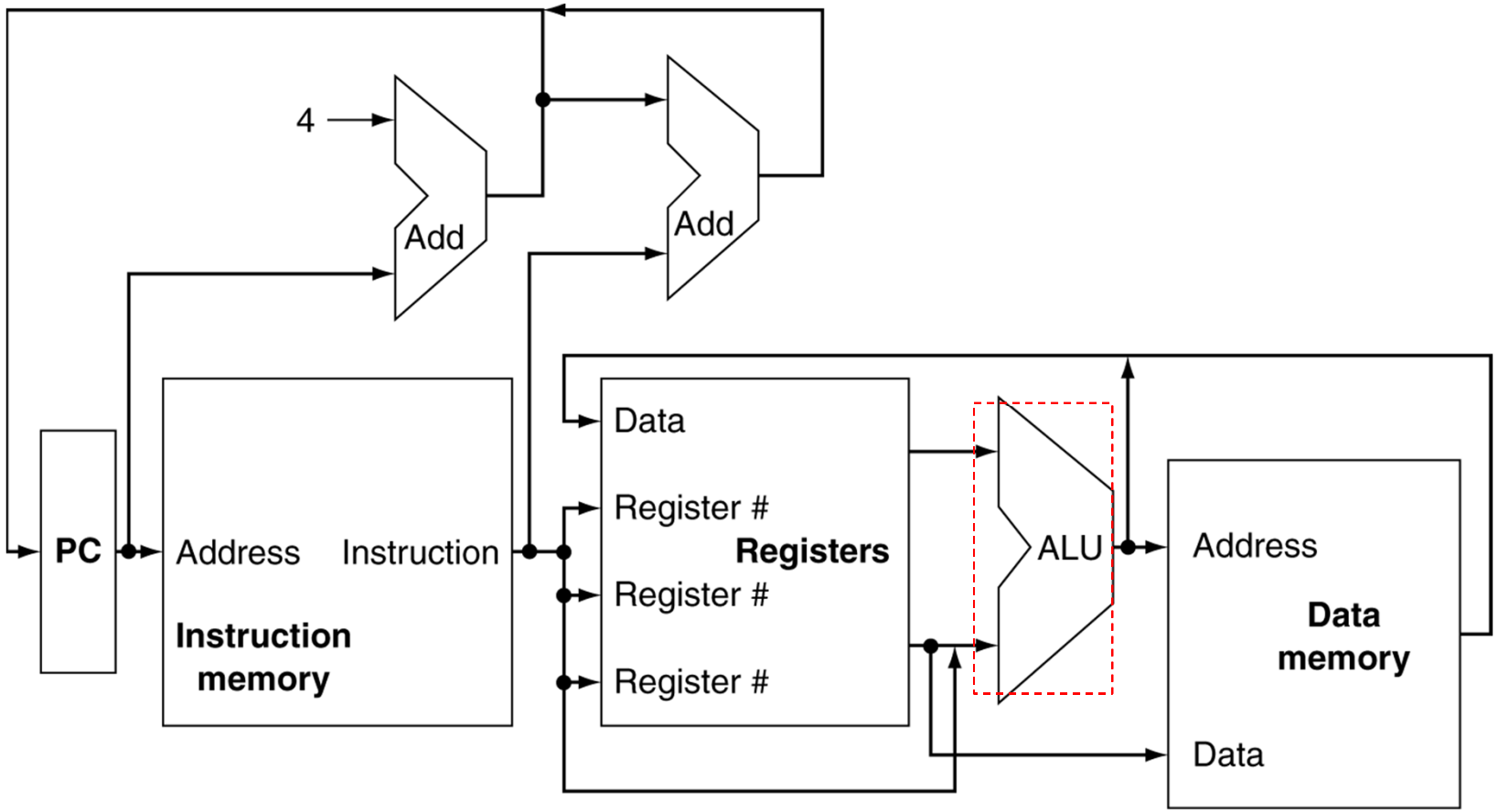
address instruction

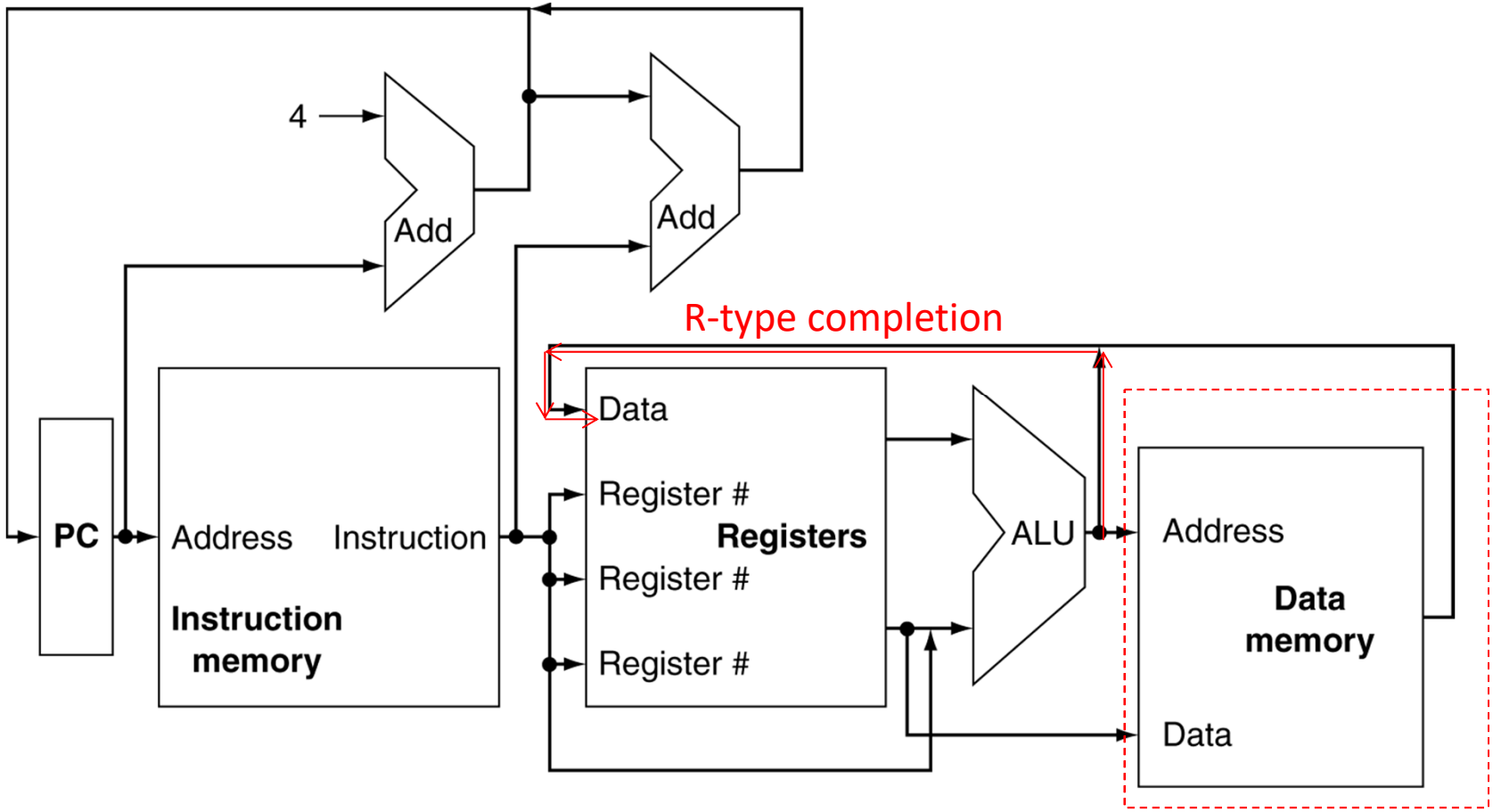
```
User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)           ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4        ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4         ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080  sll $2, $4, 2           ; 186: sll $v0 $a0 2
[00400010] 00c23021  addu $6, $6, $2         ; 187: addu $a2 $a2 $v0
[00400014] 0c000000  jal 0x00000000 [main]   ; 188: jal main
[00400018] 00000000  nop                     ; 189: nop
[0040001c] 3402000a  ori $2, $0, 10          ; 191: li $v0 10
[00400020] 0000000c  syscall                 ; 192: syscall # syscall 10 (exit)
```

執行指令的流程


4
Add
Add
PC
Address
Instruction
Instruction memory
Data
Register #
Registers
Register #
Register #
ALU
Address
Data memory
Data
前半:解碼
後半:讀值

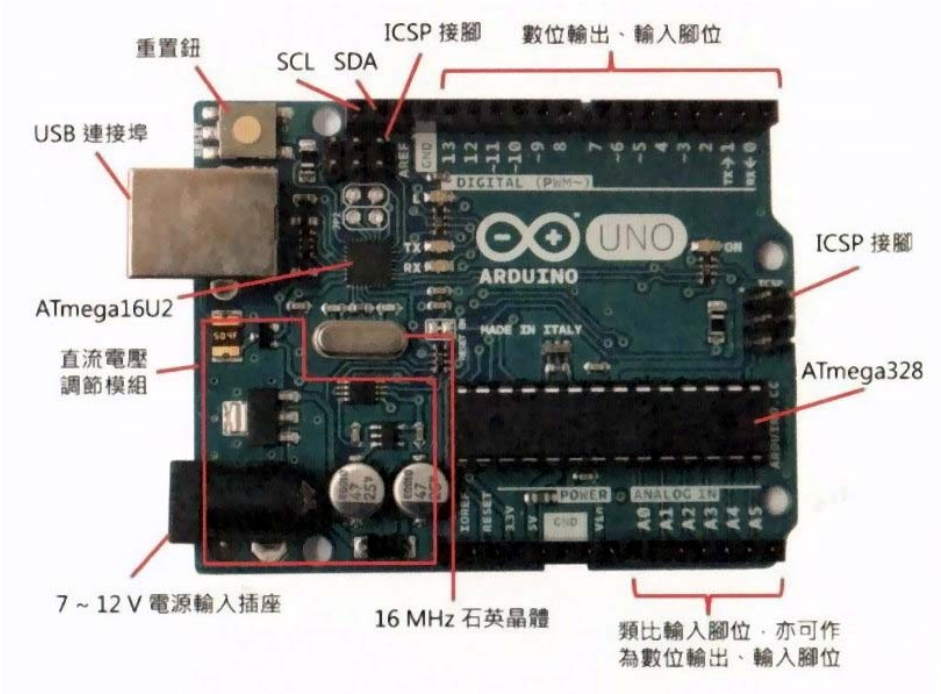
執行指令的流程（EX）


4
Add
Add
PC
Address
Instruction
Instruction memory
Data
Register #
Registers
Register #
Register #
ALU
Address
Data memory
Data

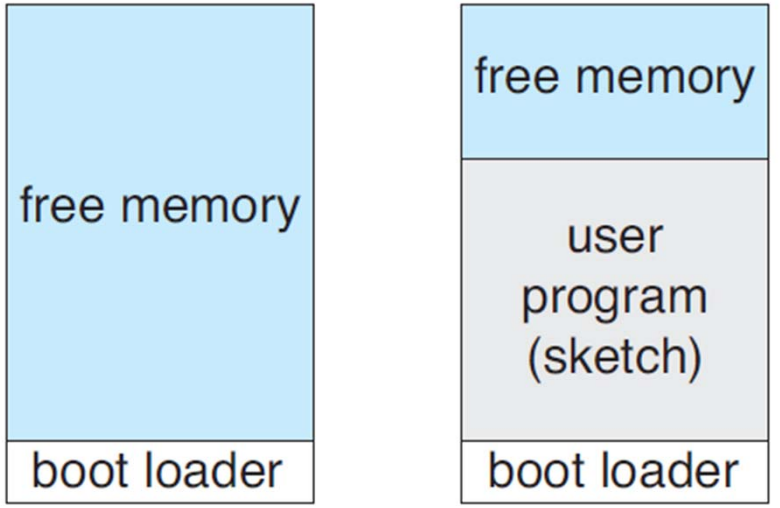

4
Add
Add
R-type completion
PC
Address
Instruction
Instruction memory
Data
Register #
Registers
Register #
Register #
ALU
Address
Data memory
Data

- 有了CPU、有了程式碼，如何讓系統動起來?
  - 開啟電源後，第一件事要做什麼?

Arduino



Program Memory of Aruidno



System startup

Running a sketch

誰來系統初始化、運算與儲存資源調配、管理?

為什麼需要作業系統?

- 有CPU可以執行指令，為什麼還要作業系統(OS)?
  - 早期電腦的確沒有OS
  - 為什麼要OS? 沒有OS會怎樣?

沒有作業系統時

- No access to C library (libc) and headers
- No memory protection
- No floating-point operations support
  - 每個平台(CPU)有自己的機制
- No concurrency support
- No development support
  - No compiler and debugger
  - 要另有機制設置開發環境

為什麼需要作業系統?

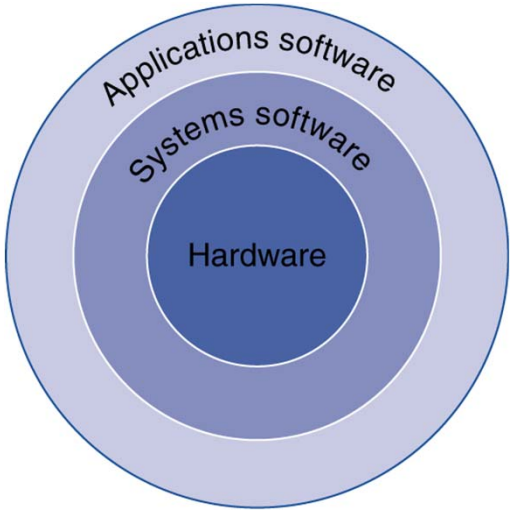
- 早期電腦的確沒有作業系統 (OS)
- 所以，為什麼要OS? 沒有OS會怎樣?
  - 沒有檔案系統 (應用程式直接管理)
  - 缺乏UI (CLI)
  - 只能執行一個程式
  - 程式難以除錯
  - 沒有安全保護

為什麼需要作業系統?

- CPU: 從記憶體提取指令循序執行
  - 一開始記憶體是空的
  - 第一個指令存那? 怎麼開始?
- 如何和CPU之外的硬體溝通?
  - 記憶體、硬碟、網路卡…
  - 指令如何能操作各項裝置?
- 進階應用
  - 如何在循序執行指令的CPU上，讓很多程式同時被執行?
  - 程式開發: 怎麼寫指令? 用紙筆? 寫完後怎麼放到記憶體?
- 其它問題…
  - 模組化
  - 效率
  - 安全

如果沒有作業系統，電腦應用程式的開發人員，必須自行處理這些工作!

Below Your Program



- 應用程式 Application software
  - Written in high-level language
- 系統程式 System software
  - Compiler: 轉換程式語言為機器碼
  - OS
    - Handling input/output
    - Managing memory and storage
    - Scheduling tasks & sharing resources
- 硬體 Hardware
  - Processor, memory, I/O controllers

Challenges for OS in Modern Platforms

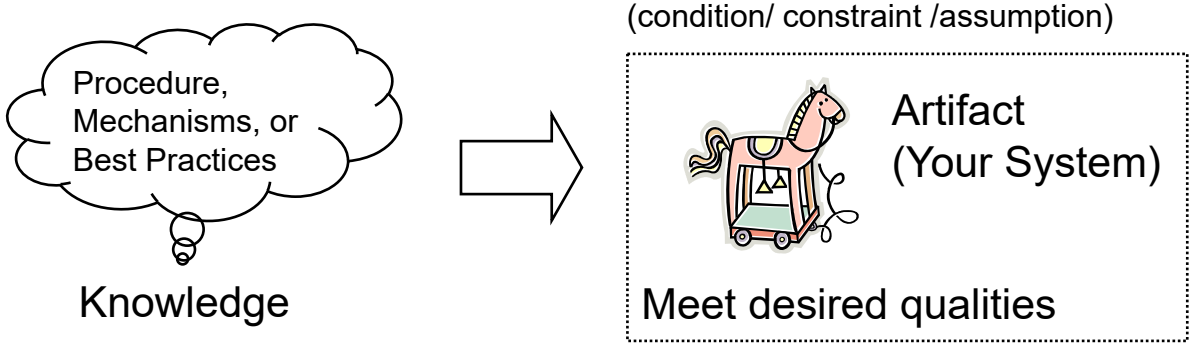
- Mobile devices
  - Power consumption become a critical concern
- Cloud computing
  - Virtualized hardware
- IoT
  - Limited resources
- Diversity architectures
  - x86, ARM, Apple M series, RISC-V

學科本質

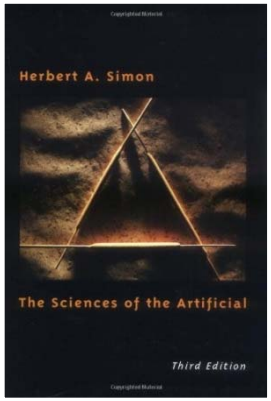
- Operating System
  - Software system
  - Belongs to the Engineering discipline
- Engineering and Science
  - Engineering is the Science of the Artificial
  - Engineering is about the Design of artifacts
- Design
  - To invent and bring into being (Webster's Dictionary)

Engineering

- Objective
  - Designing an artifact that meets desired qualities
- Valuable knowledge
  - The mechanisms for such artifact



H.A.Simon, *The Sciences of the Artificial*, 1996



這門課如何上

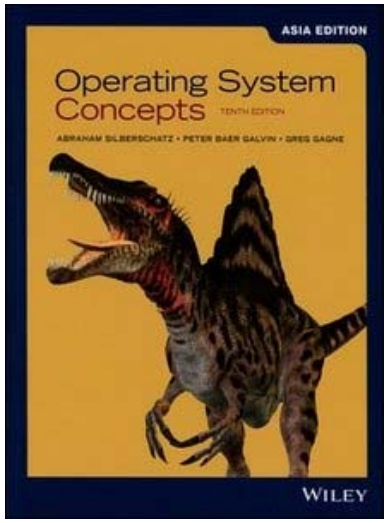
- 大學如何協助學生學習?
  - 提供學習動力
    - 訂定必修課、控管學習品質
    - Mind set
    - 考試
  - 營造學習環境
    - 固定學習時間、作業、討論
  - 講授必要知識
    - 針對必要重點講解 (自己看不容易懂的部份)

這門課如何上

- 課堂講授
  - 建立基本知識
  - 投影片、黑板、隨堂抽問
  - 投影片原則
    - 定位:為幫助你理解書中重要概念，不是要取代教科書!
    - 中文或英文: 依出現時機，以同學容易理解為考量
    - 下課後放上

Text book

- 教科書
  - Abraham Silberschatz, Greg Gagne, Peter Galvin
  - Operating System Concepts, 10th ed., John Wiley



亞洲英文版

應購買原文書

本書是經典中的經典，工作經常會用到

第10版(2018出版)更新了許多重要當代OS知識

上課教材會搭配原文頁碼

學習不能只靠片斷知識

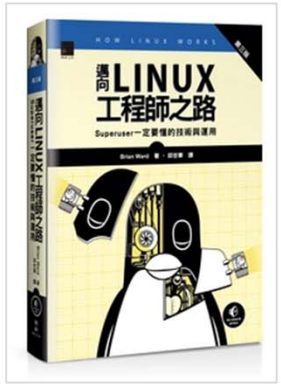
投影片只能提示重要概念，無法完全深入理解

Reference

- 有用的參考書

邁向 Linux 工程師之路：Superuser 一定要懂的技術與運用, 3/e (How Linux Works : What Every Superuser Should Know, 3/e)

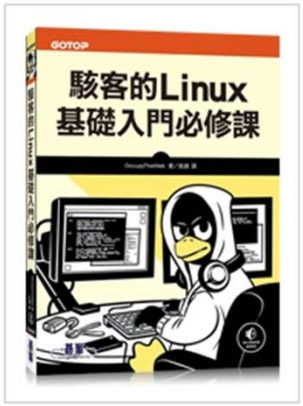
Brian Ward 著 邱世華 譯



預覽內頁

出版商:	博碩文化
出版日期:	2023-03-30
定價:	$780
售價:	7.1 折 $553 (限時優惠至 2023-08-27)
語言:	繁體中文
頁數:	528
裝訂:	平裝
ISBN:	6263334126
ISBN-13:	9786263334120
相關分類:	Linux
此書翻譯自:	How Linux Works : What Every Superuser Should Know, 3/e (Paperback)
相關翻譯:	Linux 是怎樣工作的 (簡中版)

立即出貨 (庫存 > 10) 讚 0 分享



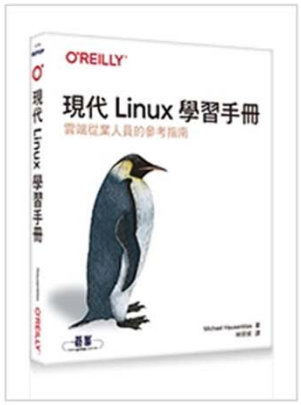
預覽內頁

出版商:	碁峰資訊
出版日期:	2019-07-16
定價:	$420
售價:	7.9 折 $332
語言:	繁體中文
ISBN:	9865021994
ISBN-13:	9789865021993
相關分類:	Linux、資訊安全、駭客 Hack
此書翻譯自:	Linux Basics for Hackers: Getting Started with Networking, Scripting, and Security in Kali

立即出貨 (庫存 > 10) 讚 10 分享

加入購物車 加入追蹤清單

Reference



預覽內頁

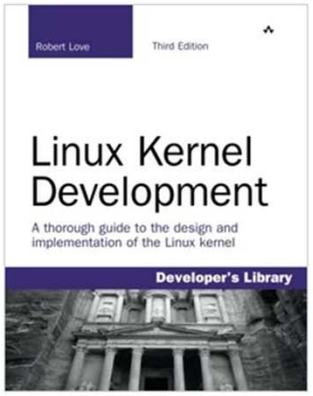
出版商:	歐萊禮
出版日期:	2023-07-05
定價:	$580
售價:	*7.9* 折 *$458*
語言:	繁體中文
頁數:	280
ISBN:	6263244380
ISBN-13:	9786263244382
相關分類:	Linux
此書翻譯自:	Learning Modern Linux: A Handbook for the Cloud Native Practitioner (Paperback)

立即出貨 (庫存 > 10) 讚 2 分享

加入購物車 加入追蹤清單

Linux Kernel Development, 3/e (Paperback)

Robert Love



出版商:	Addison Wesley
出版日期:	2010-06-25
售價:	*$1,750*
貴賓價:	*9.5* 折 *$1,663*
語言:	英文
頁數:	480
裝訂:	Paperback
ISBN:	0672329468
ISBN-13:	9780672329463
相關分類:	Linux
相關翻譯:	Linux 內核設計與實現, 3/e (Linux Kernel Development, 3/e) (簡中版)

立即出貨 讚 0 分享

給分標準

- 課堂表現/印象分數/點名 10%
- 作業 20%
- 期中考 35%
- 期末考 35%

課堂表現/印象分數 (10%)

- 每節課隨堂抽同學問問題
  - 若人到，答對加分、答錯不扣分
  - 若人未到，扣分
    - 不接受事後補點
    - 常沒來的同學，會一直點你
  - 相關問題
    - 點完名偷跑被老師發現?
    - 可不可吃東西?
    - 上課玩手機?

程式作業 (20%)

- 程式作業繳交方式
  - 三次
    - Kernel Module
    - eBPF
    - Container (Docker)
  - 派出後二週內
    - 依格式規定，Zip後上傳moodles
    - 逾時不收，未交者0分
  - 注意事項
    - 抄襲者依學校相關規定辦理

期中與期末考 (70%)

- 考古題約佔30% (公佈在moodle上)
- 可帶正反A4小抄
- 考試試題來源
  - 課本內容 (含延伸閱讀)
  - 投影片
  - 作業

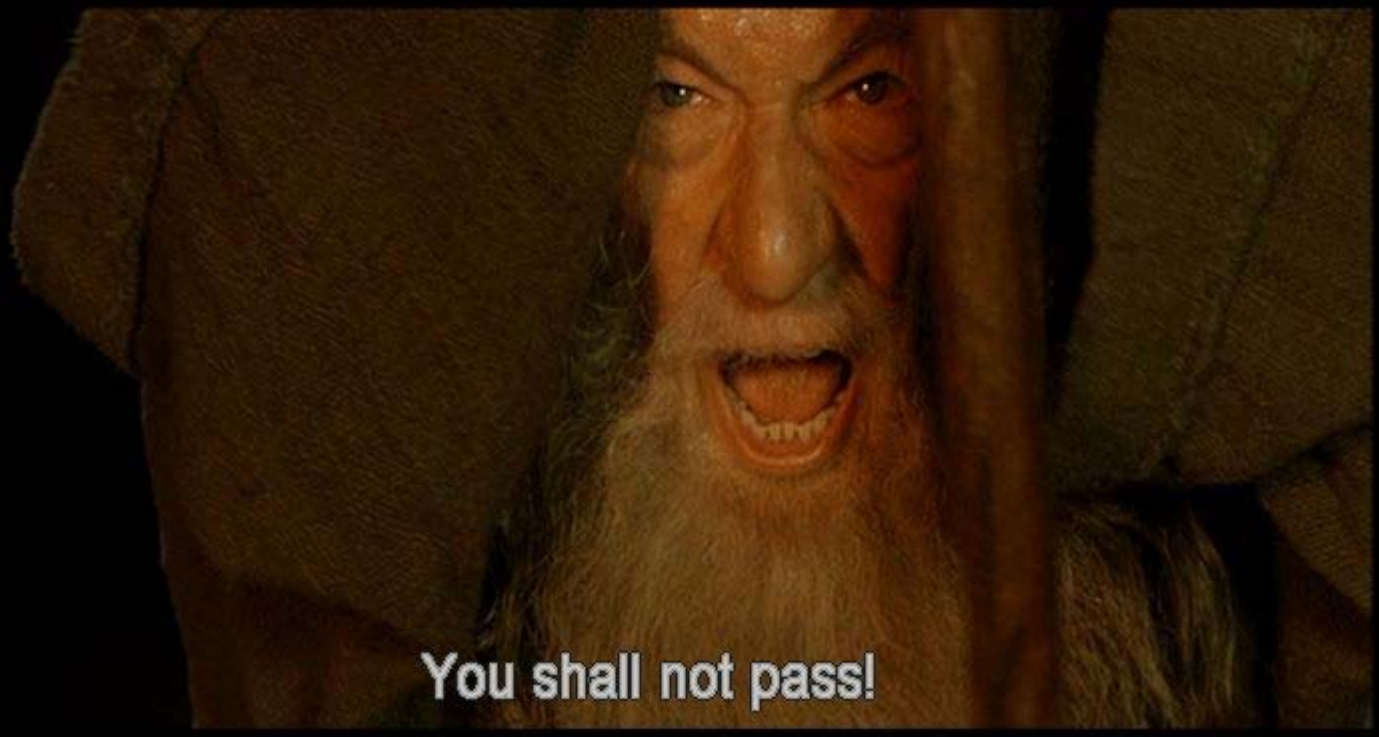
課程助教

- 助教
  - 資訊碩 吳泓澈
  - 資訊碩 黃薇倫

Syllabus

- 進度視同學學習情況調整，請以Moodle為主

如何被當掉?



常常缺課

不上Moodle

不交作業或抄作業

上課不專心

最後的一些提醒

不調分

被當的時候該怎麼辦

你只能重修

求情成功率=0%

Final Reminder

- 請同學儘可能加入FB，可即時得知最新課程訊息
  - https://www.facebook.com/groups/236713895592851
  - 可以請助教幫您加入
- Moodle已公告事項未留意或因未收登錄於學校的email導致任何後續問題，後果自行負責
- 到教材官網看一下
  - https://www.os-book.com/OS10/
  - 延伸閱讀
  - 線上教材
  - 課本錯誤更正

Q & A