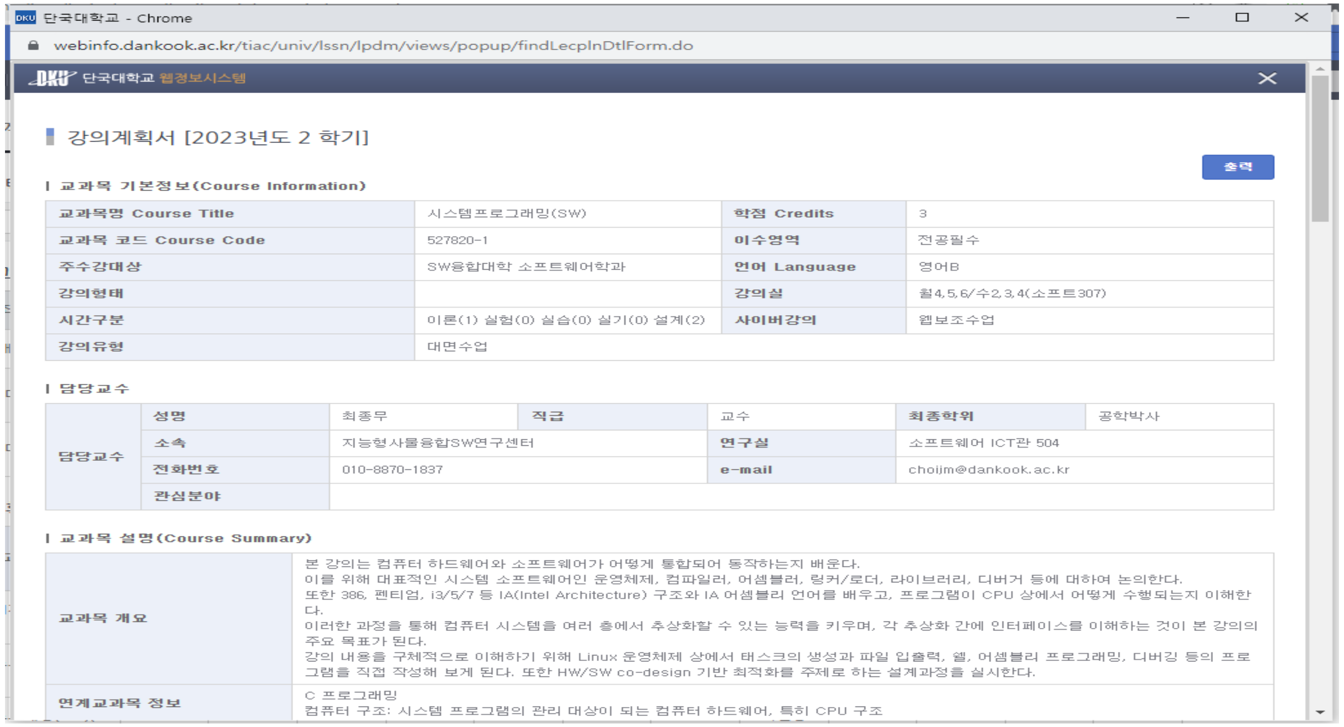
Lecture Note 0.
Lecture Overview

September 4, 2023

Jongmoo Choi
Dept. of Software
Dankook University
http://embedded.dankook.ac.kr/~choijm

Contents

- Course objectives
  - ✓ What can we learn in this semester?
- Course contents
  - ✓ Text book, Lecture notes, …
- Course methods
  - ✓ Assignment, Grade, …



단국대학교 - Chrome
webinfo.dankook.ac.kr/tiac/univ/lssn/lpdm/views/popup/findLecplnDtlForm.do
DKU 단국대학교 웹정보시스템
출력

강의계획서 [2023년도 2 학기]

교과목 기본정보(Course Information)

교과목명 Course Title	시스템프로그래밍(SW)	학점 Credits	3
교과목 코드 Course Code	527820-1	이수영역	전공필수
주수강대상	SW융합대학 소프트웨어학과	언어 Language	영어B
강의형태		강의실	월4,5,6/수2,3,4(소프트307)
시간구분	이론(1) 실험(0) 실습(0) 실기(0) 설계(2)	사이버강의	웹보조수업
강의유형	대면수업		

담당교수

담당교수	성명	최종무	직급	교수	최종학위	공학박사
	소속	지능형사물융합SW연구센터		연구실	소프트웨어 ICT관 504	
	전화번호	010-8870-1837		e-mail	choijm@dankook.ac.kr	
	관심분야					

교과목 설명(Course Summary)

교과목 개요	본 강의는 컴퓨터 하드웨어와 소프트웨어가 어떻게 통합되어 동작하는지 배운다. 이를 위해 대표적인 시스템 소프트웨어인 운영체제, 컴파일러, 어셈블러, 링커/로더, 라이브러리, 디버거 등에 대하여 논의한다. 또한 386, 펜티엄, i3/5/7 등 IA(Intel Architecture) 구조와 IA 어셈블리 언어를 배우고, 프로그램이 CPU 상에서 어떻게 수행되는지 이해한다. 이러한 과정을 통해 컴퓨터 시스템을 여러 층에서 추상화할 수 있는 능력을 키우며, 각 추상화 간에 인터페이스를 이해하는 것이 본 강의의 주요 목표가 된다. 강의 내용을 구체적으로 이해하기 위해 Linux 운영체제 상에서 태스크의 생성과 파일 입출력, 쉘, 어셈블리 프로그래밍, 디버깅 등의 프로그램을 직접 작성해 보게 된다. 또한 HW/SW co-design 기반 최적화를 주제로 하는 설계과정을 실시한다.
연계교과목 정보	C 프로그래밍 컴퓨터 구조: 시스템 프로그램의 관리 대상이 되는 컴퓨터 하드웨어, 특히 CPU 구조

Course Objectives (1/2)

- What is System Programming?
  - ✓ Application program vs. System program

```
#include <stdio.h>

int main()
{
    printf(“Hello, World₩n”);
}
```

- ☞ **How to run this program on CPU?**
- ☞ **What is the role of printf()?**
- ☞ **How the string is displayed on Monitor?**
- ☞ **How this program can be executed with other programs concurrently?**
- ☞ **What are the differences between local and global variables?**
- ☞ **What if we split the string “Hello, World\n” into two strings with two printf()s?**

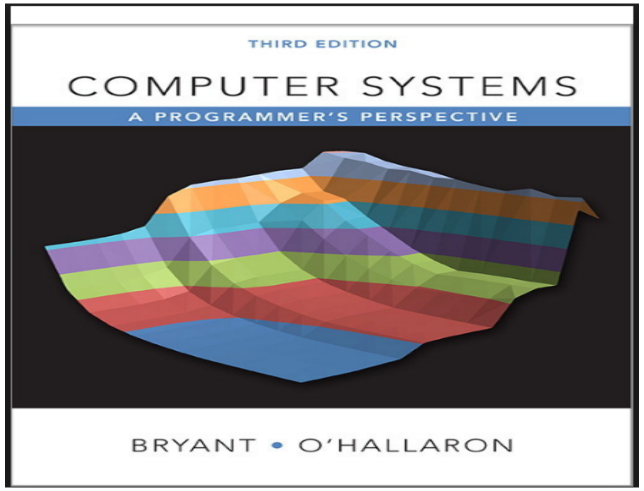
Course Objectives (2/2)

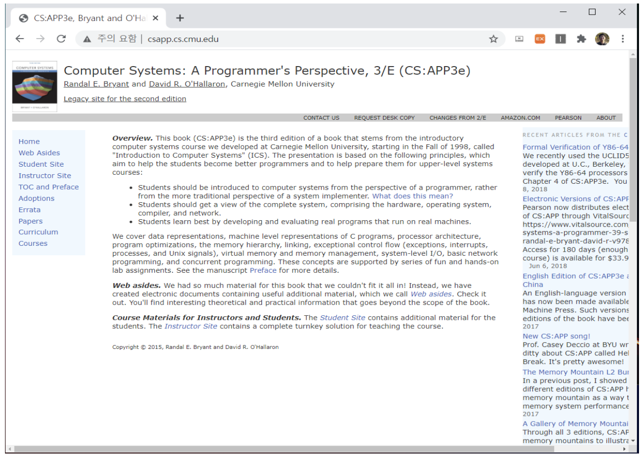
- Understand how software runs on hardware (or how software and hardware are connected)
  - ✓ High-level program for human vs. Binary for CPU
  - ✓ Compiler, Assembler, Linker, Loader, Debugger, Library (dll), …
  - ✓ File system, Device driver
  - ✓ Concept of Process, Scheduling for multiple processes
  - ✓ Memory management (data/stack/heap, virtual memory)
  - ✓ Software-level optimizations: code motion, loop unrolling, …
  - ✓ Hardware-level optimizations: pipeline, cache, …
  - ✓ Recent technologies in Intel CPU

- Grasp the concept of abstraction
  - ✓ Information hiding
  - ✓ Interface vs. Implementation
  - ✓ Layered architecture

Course Contents (1/4)

- Textbook 1: CSAPP
  - ✓ Computer Systems: A Programmer's Perspective, by R. Bryant and D. O'Hallaron
  - ✓ Contents
    1. A Tour of Computer Systems
    2. Representing and Manipulating Information
    3. Machine-level Representation of Programs
    4. Processor Architecture
    5. Optimizing Program Performance
    6. The Memory Hierarchy
    7. Linking
    8. Exceptional Control Flow
    9. Virtual Memory
    10. System-Level I/O
    11. Network Programming
    12. Concurrent Programming





Computer Systems: A Programmer's Perspective, 3/E (CS:APP3e)

Randal E. Bryant and David R. O'Hallaron, Carnegie Mellon University

Legacy site for the second edition

CONTACT US | REQUEST DESK COPY | CHANGES FROM 2/E | AMAZON.COM | PEARSON | ABOUT

Home
Web Asides
Student Site
Instructor Site
TOC and Preface
Adoptions
Errata
Papers
Curriculum
Courses

Overview. This book (CS:APP3e) is the third edition of a book that stems from the introductory computer systems course we developed at Carnegie Mellon University, starting in the Fall of 1998, called "Introduction to Computer Systems" (ICS). The presentation is based on the following principles, which aim to help the students become better programmers and to help prepare them for upper-level systems courses:

- Students should be introduced to computer systems from the perspective of a programmer, rather from the more traditional perspective of a system implementer. What does this mean?
- Students should get a view of the complete system, comprising the hardware, operating system, compiler, and network.
- Students learn best by developing and evaluating real programs that run on real machines.

We cover data representations, machine level representations of C programs, processor architecture, program optimizations, the memory hierarchy, linking, exceptional control flow (exceptions, interrupts, processes, and Unix signals), virtual memory and memory management, system-level I/O, basic network programming, and concurrent programming. These concepts are supported by series of fun and hands-on lab assignments. See the manuscript Preface for more details.

Web asides. We had so much material for this book that we couldn't fit it all in! Instead, we have created electronic documents containing useful additional material, which we call *Web asides*. Check it out. You'll find interesting theoretical and practical information that goes beyond the scope of the book.

Course Materials for Instructors and Students. The *Student Site* contains additional material for the students. The *Instructor Site* contains a complete turnkey solution for teaching the course.

Copyright © 2015, Randal E. Bryant and David R. O'Hallaron

RECENT ARTICLES FROM THE C

Formal Verification of Y86-64
We recently used the UCLID5
developed at U.C., Berkeley,
verify the Y86-64 processors
Chapter 4 of CS:APP3e. You
8, 2018

Electronic Versions of CS:APP
Pearson now distributes elect
of CS:APP through VitalSourc
https://www.vitalsource.com
systems-a-programmer-39-s
randal-e-bryant-david-r-v978
Access for 180 days (enough
course) is available for $33.9
Jun 6, 2018

English Edition of CS:APP3e
China
An English-language version
has now been made available
Machine Press. Such versions
editions of the book have bee
2017

New CS:APP song!
Prof. Casey Deccio at BYU wr
ditty about CS:APP called Hel
Break. It's pretty awesome!

The Memory Mountain L2 Bu
In a previous post, I showed
different editions of CS:APP
memory mountain as a way
memory system performance
2017

A Gallery of Memory Mountai
Through all 3 editions, CS:AP
memory mountains to illustra

(http://csapp.cs.cmu.edu/)

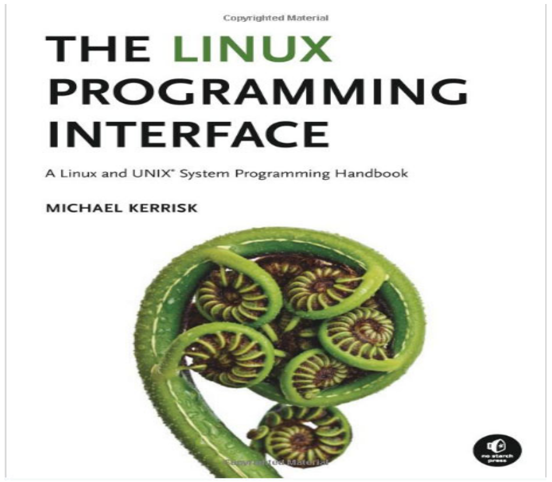
Course Contents (2/4)

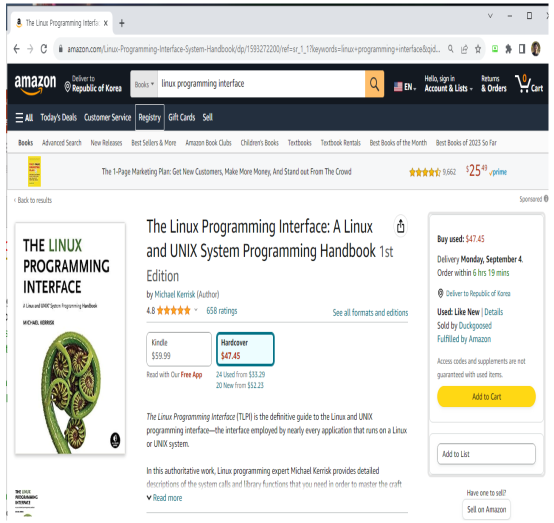
- Textbook 2: LPI
  - ✓ The Linux Programming Interface: A Linux and UNIX System Programming Handbook
    1. History and Standards
    2. Fundamental Concepts
    3. System programming concepts
    4. File I/O: The Universal I/O Model
    5. File I/O: Further Details
    6. Process
    7. Memory Allocation
    8. Users and Groups

    …

    24. Process Creation
    25. Process Termination
    26. Monitoring Child Processes
    27. Program Execution

    … /* total 64 chapters */


Copyrighted Material
THE LINUX PROGRAMMING INTERFACE
A Linux and UNIX® System Programming Handbook
MICHAEL KERRISK



The 1-Page Marketing Plan: Get New Customers, Make More Money, And Stand out From The Crowd 9,662 $25.49 prime

The Linux Programming Interface: A Linux and UNIX System Programming Handbook 1st Edition

by Michael Kerrisk (Author)

4.8 658 ratings See all formats and editions

Kindle $59.99 Read with Our Free App

Hardcover $47.45 24 Used from $33.29 20 New from $52.23

The Linux Programming Interface (TLPI) is the definitive guide to the Linux and UNIX programming interface—the interface employed by nearly every application that runs on a Linux or UNIX system.

In this authoritative work, Linux programming expert Michael Kerrisk provides detailed descriptions of the system calls and library functions that you need in order to master the craft

Buy used: $47.45

Delivery Monday, September 4. Order within 6 hrs 19 mins

Deliver to Republic of Korea

Used: Like New | Details

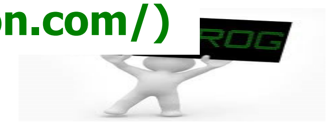
Sold by Duckgoosed

Fulfilled by Amazon

Access codes and supplements are not guaranteed with used items.

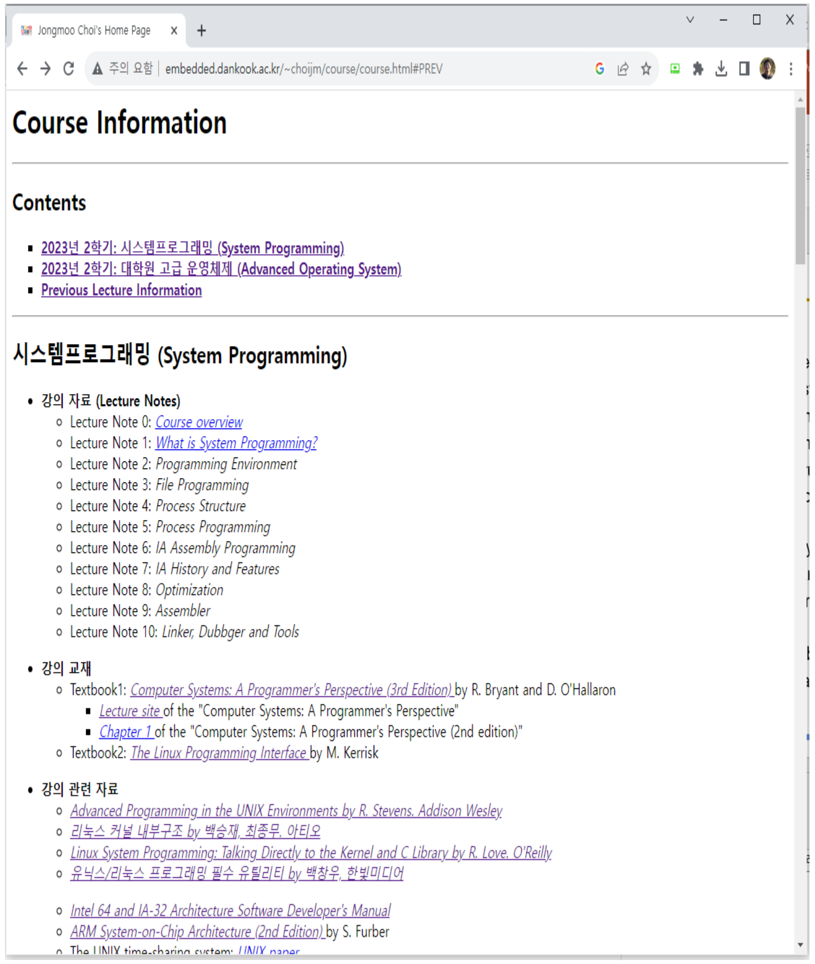
Have one to sell?

(https://www.amazon.com/)



Course Contents (3/4)

- Lecture Notes
  - ✓ LN0: Course Overview
  - ✓ LN1: What is System Programming?
  - ✓ LN2: Programming Environments
  - ✓ LN3: File Programming
  - ✓ LN4: Process Structure
  - ✓ LN5: Process Programming
  - ✓ LN6: IA Assembly Programming
  - ✓ LN7: IA History and Features
  - ✓ LN8: Optimization Practice
  - ✓ LN9: Assembler
  - ✓ LN10: Linker, Debugger and Tools



Course Information

Contents

- 2023년 2학기: 시스템프로그래밍 (System Programming)
- 2023년 2학기: 대학원 고급 운영체제 (Advanced Operating System)
- Previous Lecture Information

시스템프로그래밍 (System Programming)

- 강의 자료 (Lecture Notes)
  - Lecture Note 0: *Course overview*
  - Lecture Note 1: *What is System Programming?*
  - Lecture Note 2: *Programming Environment*
  - Lecture Note 3: *File Programming*
  - Lecture Note 4: *Process Structure*
  - Lecture Note 5: *Process Programming*
  - Lecture Note 6: *IA Assembly Programming*
  - Lecture Note 7: *IA History and Features*
  - Lecture Note 8: *Optimization*
  - Lecture Note 9: *Assembler*
  - Lecture Note 10: *Linker, Dubbger and Tools*
- 강의 교재
  - Textbook1: *Computer Systems: A Programmer's Perspective (3rd Edition)* by R. Bryant and D. O'Hallaron
    - *Lecture site* of the "Computer Systems: A Programmer's Perspective"
    - *Chapter 1* of the "Computer Systems: A Programmer's Perspective (2nd edition)"
  - Textbook2: *The Linux Programming Interface* by M. Kerrisk
- 강의 관련 자료
  - *Advanced Programming in the UNIX Environments by R. Stevens. Addison Wesley*
  - *리눅스 커널 내부구조 by 백승재, 최종무. 아티오*
  - *Linux System Programming: Talking Directly to the Kernel and C Library by R. Love. O'Reilly*
  - *유닉스/리눅스 프로그래밍 필수 유틸리티 by 백창우, 한빛미디어*
  - *Intel 64 and IA-32 Architecture Software Developer's Manual*
  - *ARM System-on-Chip Architecture (2nd Edition)* by S. Furber
  - The UNIX time-sharing system: *UNIX paper*

(http://embedded.dankook.ac.kr/~choijm/)

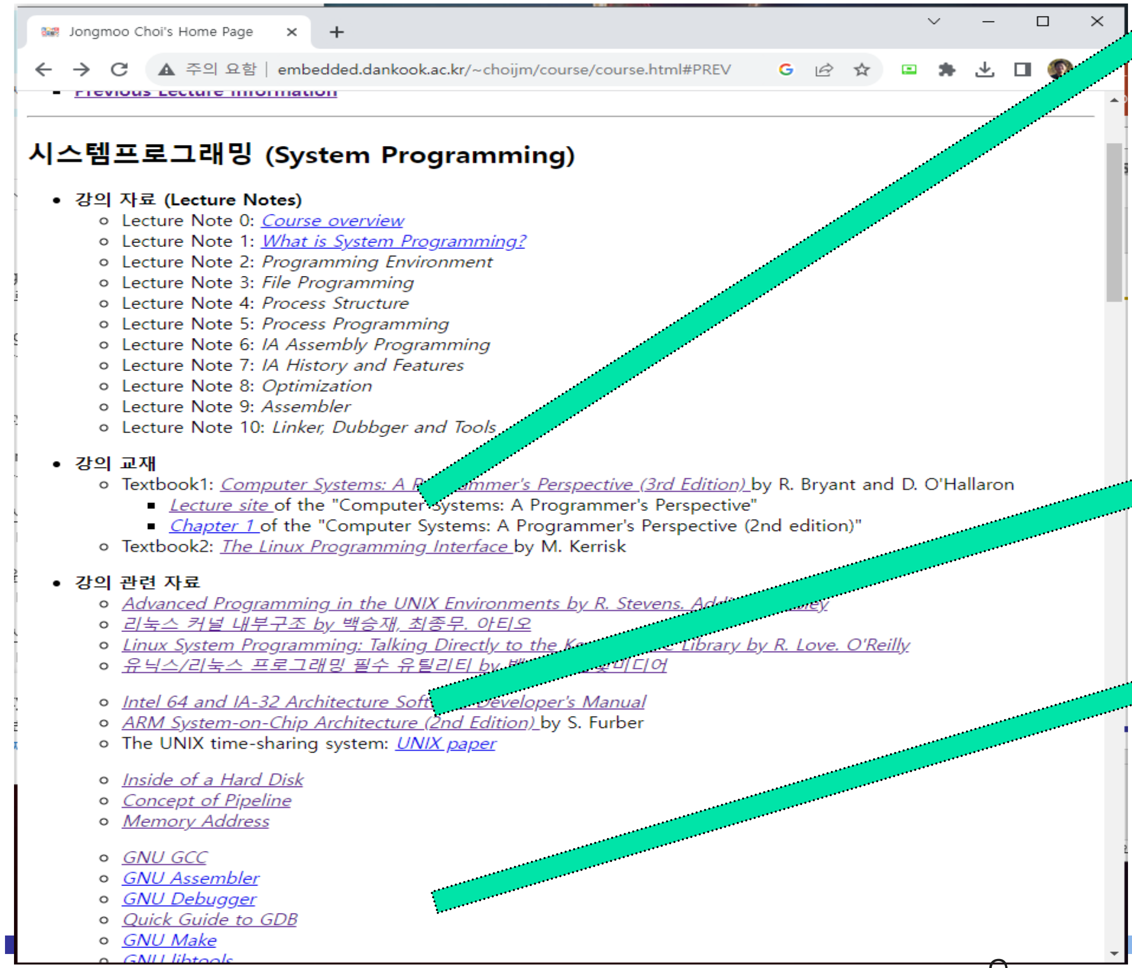
Course Contents (4/4)

- Suggestion
  - ✓ Lecture notes are sufficient for this class
  - ✓ But, text books are powerful tools to improve your knowledge

- Relation btw Lecture Notes and Textbooks
  - ✓ LN1. What is System Programming? : CSAPP Chap. 1
  - ✓ LN2. Programming Environment: LPI Chap. 1, 2, 3
  - ✓ LN3. File Programming: LPI Chap. 4, 5 / CSAPP Chap. 10
  - ✓ LN4. Process Structure: LPI Chap. 6 / CSAPP Chap. 8, 9
  - ✓ LN5. Process Programming: LPI Chap. 24, 25, 27, 29 / CSAPP Chap. 8, 12
  - ✓ LN6. IA assembly Programming: CSAPP Chap. 2, 3 / Intel Dev. Manual
  - ✓ LN7. IA History and Features: CSAPP Chap. 4 / Intel Dev. Manual
  - ✓ LN8. Optimization Practice: CSAPP Chap. 5, 6 / LPI Chap. 23
  - ✓ LN9. Assembler: CSAPP Chap. 3, 7
  - ✓ LN10. Linker, Debugger and Tools: CSAPP Chap. 7

Course Methods (1/3)

- Class hour
  - ✓ Lecturing and Discussion (Q&A)
    - Using ppt from lecture site
    - Q&A is quite important (especially I like questions from students)

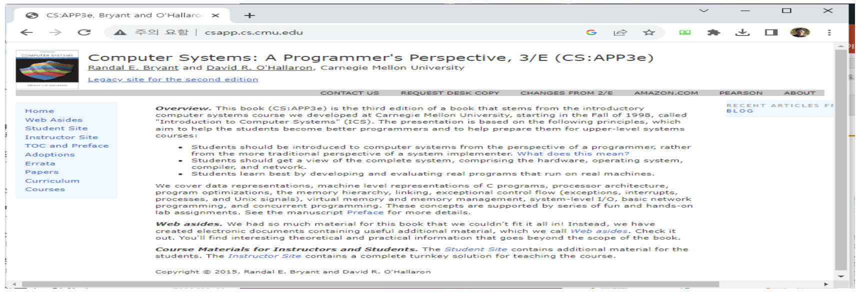


Jongmoo Choi's Home Page

주의 요함 | embedded.dankook.ac.kr/~choijm/course/course.html#PREV

시스템프로그래밍 (System Programming)

- 강의 자료 (Lecture Notes)
  - Lecture Note 0: *Course overview*
  - Lecture Note 1: *What is System Programming?*
  - Lecture Note 2: *Programming Environment*
  - Lecture Note 3: *File Programming*
  - Lecture Note 4: *Process Structure*
  - Lecture Note 5: *Process Programming*
  - Lecture Note 6: *IA Assembly Programming*
  - Lecture Note 7: *IA History and Features*
  - Lecture Note 8: *Optimization*
  - Lecture Note 9: *Assembler*
  - Lecture Note 10: *Linker, Dubbger and Tools*
- 강의 교재
  - Textbook1: *Computer Systems: A R[illegible]mmer's Perspective (3rd Edition)* by R. Bryant and D. O'Hallaron
    - *Lecture site* of the "Computer Systems: A Programmer's Perspective"
    - *Chapter 1* of the "Computer Systems: A Programmer's Perspective (2nd edition)"
  - Textbook2: *The Linux Programming Interface* by M. Kerrisk
- 강의 관련 자료
  - *Advanced Programming in the UNIX Environments by R. Stevens. Add[illegible]ley*
  - *리눅스 커널 내부구조 by 백승재, 최종무. 아티오*
  - *Linux System Programming: Talking Directly to the K[illegible] Library by R. Love. O'Reilly*
  - *유닉스/리눅스 프로그래밍 필수 유틸리티 by [illegible]미디어*
  - *Intel 64 and IA-32 Architecture Soft[illegible]Developer's Manual*
  - *ARM System-on-Chip Architecture (2nd Edition)* by S. Furber
  - The UNIX time-sharing system: *UNIX paper*
  - *Inside of a Hard Disk*
  - *Concept of Pipeline*
  - *Memory Address*
  - *GNU GCC*
  - *GNU Assembler*
  - *GNU Debugger*
  - *Quick Guide to GDB*
  - *GNU Make*
  - *GNU libtools*



CS:APP3e, Bryant and O'Hallaron

주의 요함 | csapp.cs.cmu.edu

Computer Systems: A Programmer's Perspective, 3/E (CS:APP3e)
Randal E. Bryant and David R. O'Hallaron, Carnegie Mellon University
Legacy site for the second edition

CONTACT US | REQUEST DESK COPY | CHANGES FROM 2/E | AMAZON.COM | PEARSON | ABOUT

Home
Web Asides
Student Site
Instructor Site
TOC and Preface
Adoptions
Errata
Papers
Curriculum
Courses

Overview. This book (CS:APP3e) is the third edition of a book that stems from the introductory computer systems course we developed at Carnegie Mellon University, starting in the Fall of 1998, called "Introduction to Computer Systems" (ICS). The presentation is based on the following principles, which aim to help the students become better programmers and to help prepare them for upper-level systems courses:

- Students should be introduced to computer systems from the perspective of a programmer, rather from the more traditional perspective of a system implementer. What does this mean?
- Students should get a view of the complete system, comprising the hardware, operating system, compiler, and network.
- Students learn best by developing and evaluating real programs that run on real machines.

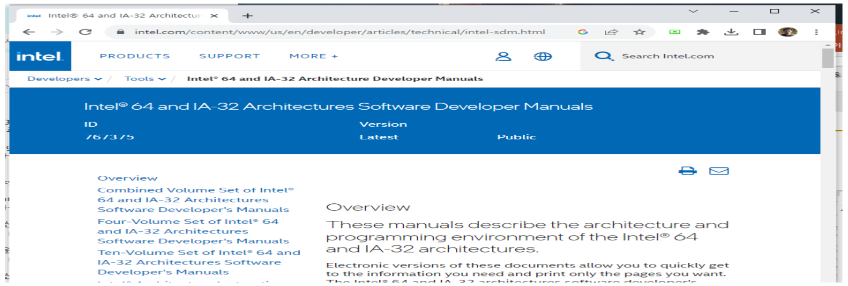
We cover data representations, machine level representations of C programs, processor architecture, program optimizations, the memory hierarchy, linking, exceptional control flow (exceptions, interrupts, processes, and Unix signals), virtual memory and memory management, system-level I/O, basic network programming, and concurrent programming. These concepts are supported by series of fun and hands-on lab assignments. See the manuscript Preface for more details.

Web asides. We had so much material for this book that we couldn't fit it all in! Instead, we have created electronic documents containing useful additional material, which we call *Web asides*. Check it out. You'll find interesting theoretical and practical information that goes beyond the scope of the book.

Course Materials for Instructors and Students. The *Student Site* contains additional material for the students. The *Instructor Site* contains a complete turnkey solution for teaching the course.

Copyright © 2015, Randal E. Bryant and David R. O'Hallaron

RECENT ARTICLES F
BLOG



Intel® 64 and IA-32 Architectur

intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html

intel. PRODUCTS SUPPORT MORE + Search Intel.com

Developers / Tools / Intel® 64 and IA-32 Architecture Developer Manuals

Intel® 64 and IA-32 Architectures Software Developer Manuals

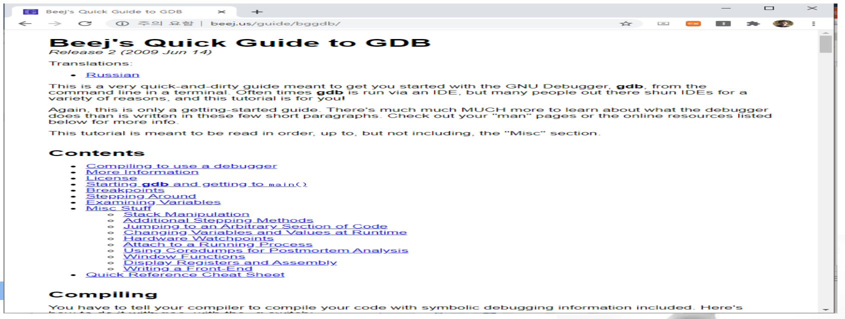
ID 767375 | Version Latest | Public

Overview
Combined Volume Set of Intel® 64 and IA-32 Architectures Software Developer's Manuals
Four-Volume Set of Intel® 64 and IA-32 Architectures Software Developer's Manuals
Ten-Volume Set of Intel® 64 and IA-32 Architectures Software Developer's Manuals

Overview

These manuals describe the architecture and programming environment of the Intel® 64 and IA-32 architectures.

Electronic versions of these documents allow you to quickly get to the information you need and print only the pages you want.



Beej's Quick Guide to GDB

주의 요함 | beej.us/guide/bggdb/

Beej's Quick Guide to GDB
Release 2 (2009 Jun 14)

Translations:

- Russian

This is a very quick-and-dirty guide meant to get you started with the GNU Debugger, **gdb**, from the command line in a terminal. Often times **gdb** is run via an IDE, but many people out there shun IDEs for a variety of reasons, and this tutorial is for you!

Again, this is only a getting-started guide. There's much much MUCH more to learn about what the debugger does than is written in these few short paragraphs. Check out your "man" pages or the online resources listed below for more info.

This tutorial is meant to be read in order, up to, but not including, the "Misc" section.

Contents

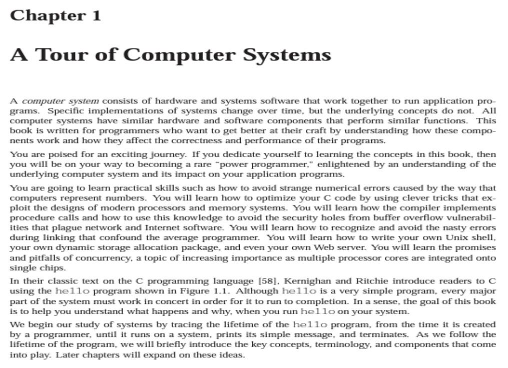
- Compiling to use a debugger
- More Information
- License
- Starting **gdb** and getting to `main()`
- Breakpoints
- Stepping Around
- Examining Variables
- Misc Stuff
  - Stack Manipulation
  - Additional Stepping Methods
  - Jumping to an Arbitrary Section of Code
  - Changing Variables and Values at Runtime
  - Hardware Watchpoints
  - Attach to a Running Process
  - Using Coredumps for Postmortem Analysis
  - Window Functions
  - Display Registers and Assembly
  - Writing a Front-End
- Quick Reference Cheat Sheet

Compiling

You have to tell your compiler to compile your code with symbolic debugging information included. Here's

Course Methods (2/3)

- Assignment: personal
  - ✓ Programming assignment: 4 or 5
    - Make programs in Linux Environment!!
      - Linux Server: 220.149.236.2 (primary), 220.149.236.4 (secondary)
      - TA: Minguk Choi (Room 515, SW-ICT Bldg)
    - Program examples
      - Using vi editor, file I/O, process manipulation, shell, assembly, optimization, …
  - ✓ Documentation assignment: 1 or 2
    - Reading a chapter in our textbooks
      - E.g. Chapter 1 in CSAPP or Chapter 3 in LPI
    - Reading a well-known paper
      - E.g. UNIX paper



Chapter 1

A Tour of Computer Systems

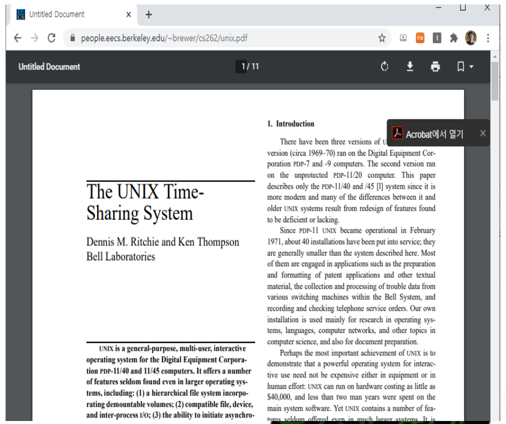
A *computer system* consists of hardware and systems software that work together to run application programs. Specific implementations of systems change over time, but the underlying concepts do not. All computer systems have similar hardware and software components that perform similar functions. This book is written for programmers who want to get better at their craft by understanding how these components work and how they affect the correctness and performance of their programs.

You are poised for an exciting journey. If you dedicate yourself to learning the concepts in this book, then you will be on your way to becoming a rare "power programmer," enlightened by an understanding of the underlying computer system and its impact on your application programs.

You are going to learn practical skills such as how to avoid strange numerical errors caused by the way that computers represent numbers. You will learn how to optimize your C code by using clever tricks that exploit the designs of modern processors and memory systems. You will learn how the compiler implements procedure calls and how to use this knowledge to avoid the security holes from buffer overflow vulnerabilities that plague network and Internet software. You will learn how to recognize and avoid the nasty errors during linking that confound the average programmer. You will learn how to write your own Unix shell, your own dynamic storage allocation package, and even your own Web server. You will learn the promises and pitfalls of concurrency, a topic of increasing importance as multiple processor cores are integrated onto single chips.

In their classic text on the C programming language [58], Kernighan and Ritchie introduce readers to C using the `hello` program shown in Figure 1.1. Although `hello` is a very simple program, every major part of the system must work in concert in order for it to run to completion. In a sense, the goal of this book is to help you understand what happens and why, when you run `hello` on your system.

We begin our study of systems by tracing the lifetime of the `hello` program, from the time it is created by a programmer, until it runs on a system, prints its simple message, and terminates. As we follow the lifetime of the program, we will briefly introduce the key concepts, terminology, and components that come into play. Later chapters will expand on these ideas.



The UNIX Time-Sharing System

Dennis M. Ritchie and Ken Thompson
Bell Laboratories

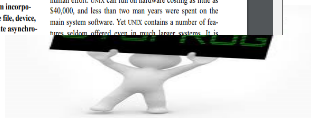
UNIX is a general-purpose, multi-user, interactive operating system for the Digital Equipment Corporation PDP-11/40 and 11/45 computers. It offers a number of features seldom found even in larger operating systems, including: (1) a hierarchical file system incorporating demountable volumes; (2) compatible file, device, and inter-process I/O; (3) the ability to initiate asynchro-

1. Introduction

There have been three versions of U version (circa 1969–70) ran on the Digital Equipment Corporation PDP-7 and -9 computers. The second version ran on the unprotected PDP-11/20 computer. This paper describes only the PDP-11/40 and /45 [l] system since it is more modern and many of the differences between it and older UNIX systems result from redesign of features found to be deficient or lacking.

Since PDP-11 UNIX became operational in February 1971, about 40 installations have been put into service; they are generally smaller than the system described here. Most of them are engaged in applications such as the preparation and formatting of patent applications and other textual material, the collection and processing of trouble data from various switching machines within the Bell System, and recording and checking telephone service orders. Our own installation is used mainly for research in operating systems, languages, computer networks, and other topics in computer science, and also for document preparation.

Perhaps the most important achievement of UNIX is to demonstrate that a powerful operating system for interactive use need not be expensive either in equipment or in human effort: UNIX can run on hardware costing as little as $40,000, and less than two man years were spent on the main system software. Yet UNIX contains a number of features seldom offered even in much larger systems. It is



Course Methods (3/3)

- Evaluation
  - ✓ Mid exam.: 30%
  - ✓ Final exam.: 30%
  - ✓ Assignment: 30%
  - ✓ Attendance/Q&A: 10%
    - Can be changed according to the progress

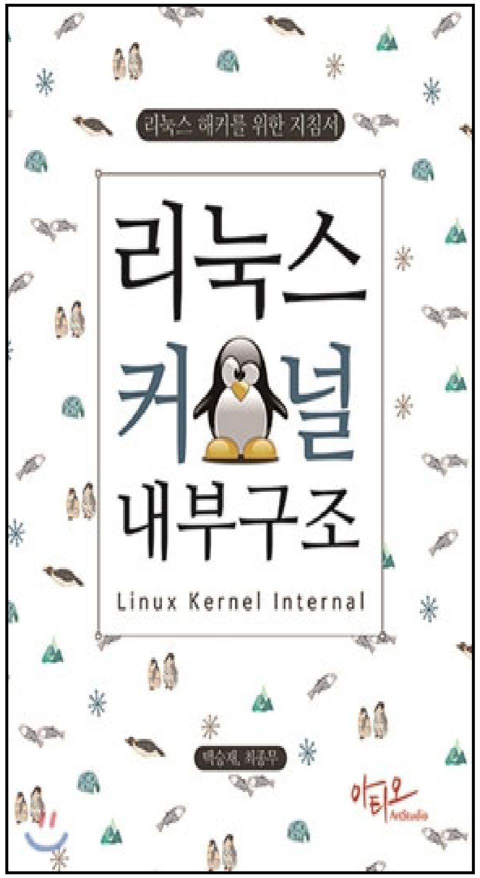
- Grade
  - ✓ Roughly, 20% students are expected to get the A grade.
    - 45% for B, others for C or D
  - ✓ Absence more than 5 times or Mid and Final Exam. Score below 20 or No assignment ➔ F

Discussion

- Q&A
  - ✓ Email: choijm@dankook.ac.kr

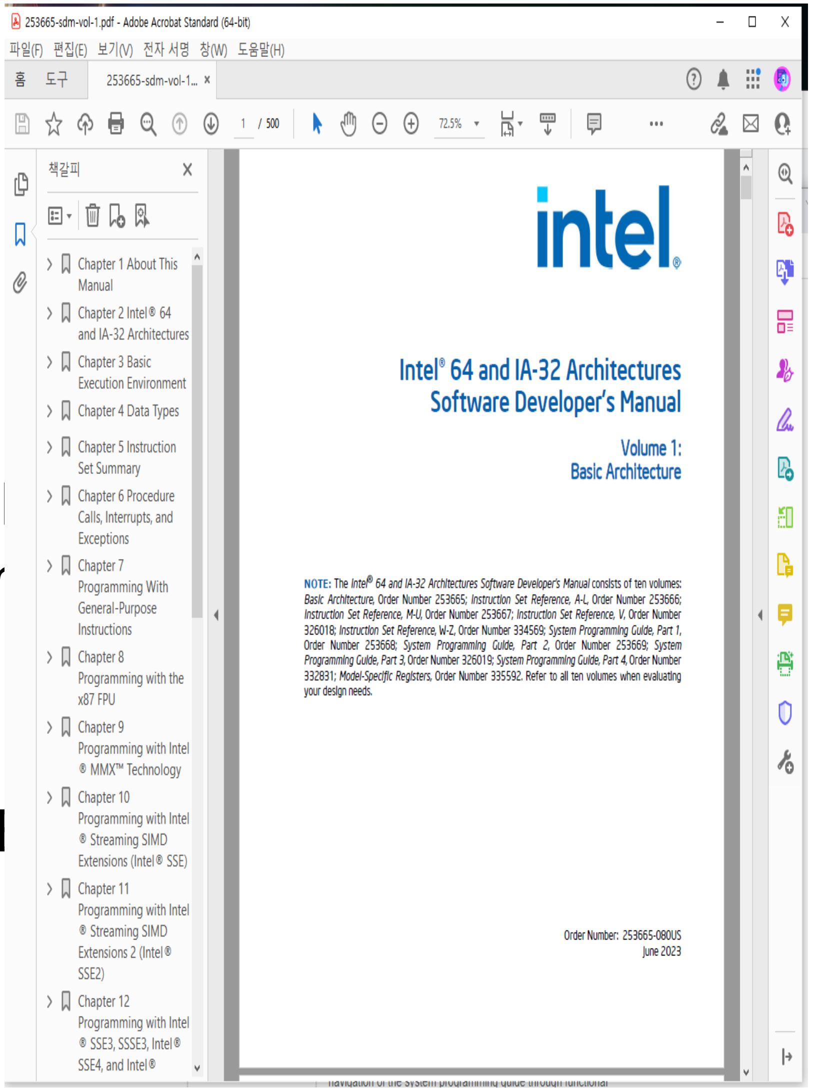
Appendix: Good book for Learning Linux

- Linux Kernel Internal (리눅스 커널 내부 구조)
  - ✓ 0장. 운영체제 이야기
  - ✓ 1장. 리눅스 소개
  - ✓ 2장. 리눅스 커널 구조
  - ✓ 3장. 태스크 관리
  - ✓ 4장. 메모리 관리
  - ✓ 5장. 파일 시스템과 가상 파일 시스템
  - ✓ 6장. 인터럽트와 트랩 그리고 시스템 호출
  - ✓ 7장. 리눅스 모듈 프로그래밍
  - ✓ 8장. 디바이스 드라이버
  - ✓ 9장. 네트워킹
  - ✓ 10장. 운영체제 관련 실습
  - ✓ 부록 A. 리눅스와 가상화 그리고 XEN
  - ✓ 부록 B. MTD와 YAFFS
  - ✓ 부록 C: Map of the Linux



Appendix: Intel Developer's Manual

- Intel®64 & IA-32 Architectures Software Developer's Manual (Volume 1: Basic Architecture)
  1. About This Manual
  2. Intel® 64 and IA-32 Architecture
  3. Basic Execution Environment
  4. Data type
  5. Instruction Set Summary
  6. Procedure Calls, Interrupts, and Exce
  7. Programming with General Purpose Ir
  8. Programming with the x87 FPU
  9. Programming with Intel MMX Technol
  10. Programming with Streaming SIMD I
  11. …


253665-sdm-vol-1.pdf - Adobe Acrobat Standard (64-bit)
파일(F) 편집(E) 보기(V) 전자 서명 창(W) 도움말(H)
홈 도구 253665-sdm-vol-1... ×
1 / 500
72.5%
책갈피

Chapter 1 About This Manual
Chapter 2 Intel® 64 and IA-32 Architectures
Chapter 3 Basic Execution Environment
Chapter 4 Data Types
Chapter 5 Instruction Set Summary
Chapter 6 Procedure Calls, Interrupts, and Exceptions
Chapter 7 Programming With General-Purpose Instructions
Chapter 8 Programming with the x87 FPU
Chapter 9 Programming with Intel® MMX™ Technology
Chapter 10 Programming with Intel® Streaming SIMD Extensions (Intel® SSE)
Chapter 11 Programming with Intel® Streaming SIMD Extensions 2 (Intel® SSE2)
Chapter 12 Programming with Intel® SSE3, SSSE3, Intel® SSE4, and Intel®

intel®

Intel® 64 and IA-32 Architectures
Software Developer's Manual

Volume 1:
Basic Architecture

NOTE: The *Intel® 64 and IA-32 Architectures Software Developer's Manual* consists of ten volumes: *Basic Architecture*, Order Number 253665; *Instruction Set Reference, A-L*, Order Number 253666; *Instruction Set Reference, M-U*, Order Number 253667; *Instruction Set Reference, V*, Order Number 326018; *Instruction Set Reference, W-Z*, Order Number 334569; *System Programming Guide, Part 1*, Order Number 253668; *System Programming Guide, Part 2*, Order Number 253669; *System Programming Guide, Part 3*, Order Number 326019; *System Programming Guide, Part 4*, Order Number 332831; *Model-Specific Registers*, Order Number 335592. Refer to all ten volumes when evaluating your design needs.

Order Number: 253665-080US
June 2023