Machine Called Computer

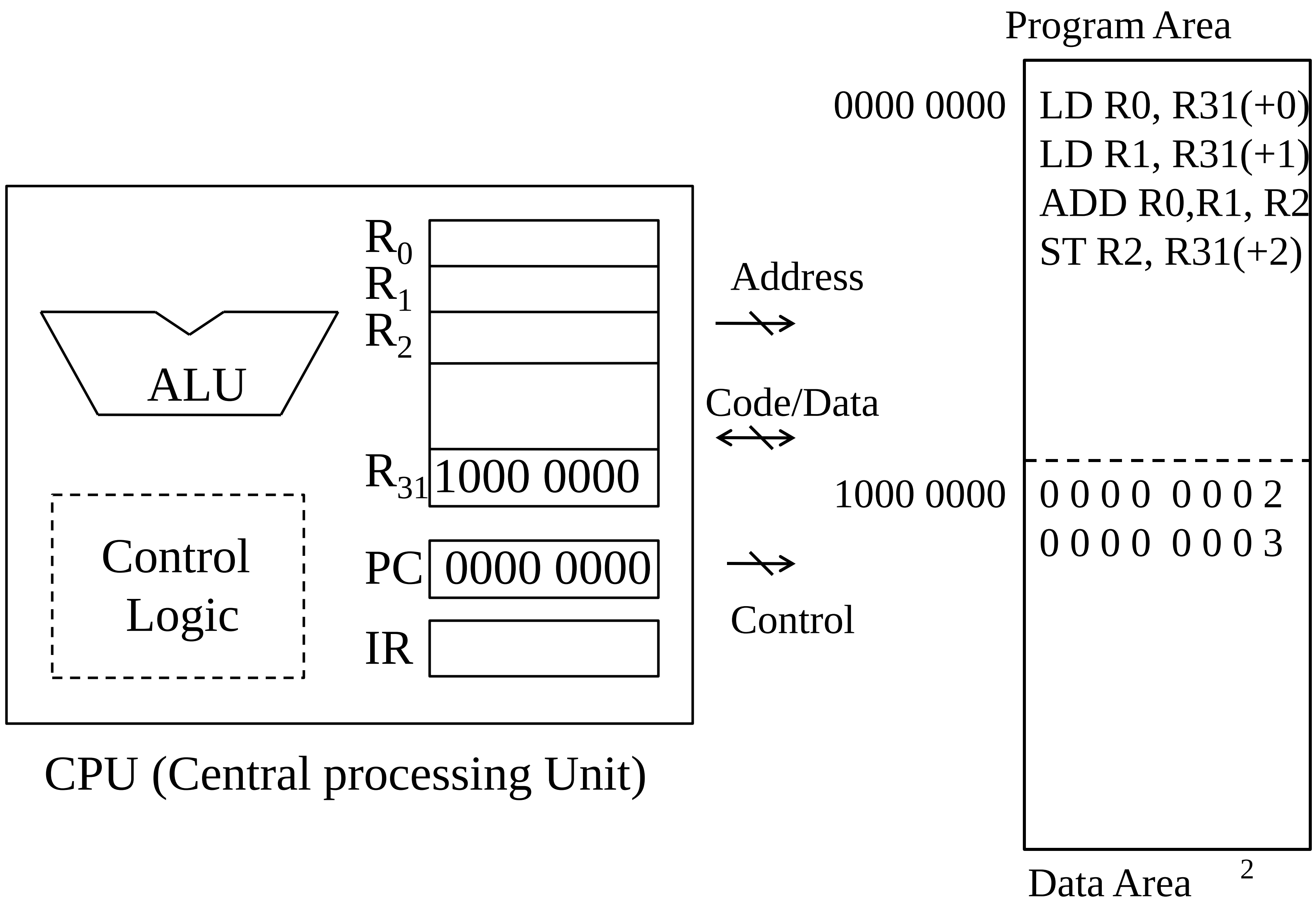
Part 5
Fetch-decode-execute,
Stored Program Computer,
ISA (Instruction Set Architecture)

instruction을 몇개 만들어야 컴퓨터가 인간의 생각을 실행할 수 있겠는가? 큰 이슈!

References:

1. Computer Organization and Design & Computer Architecture, Hennessy and Patterson (slides are adapted from those by the authors)

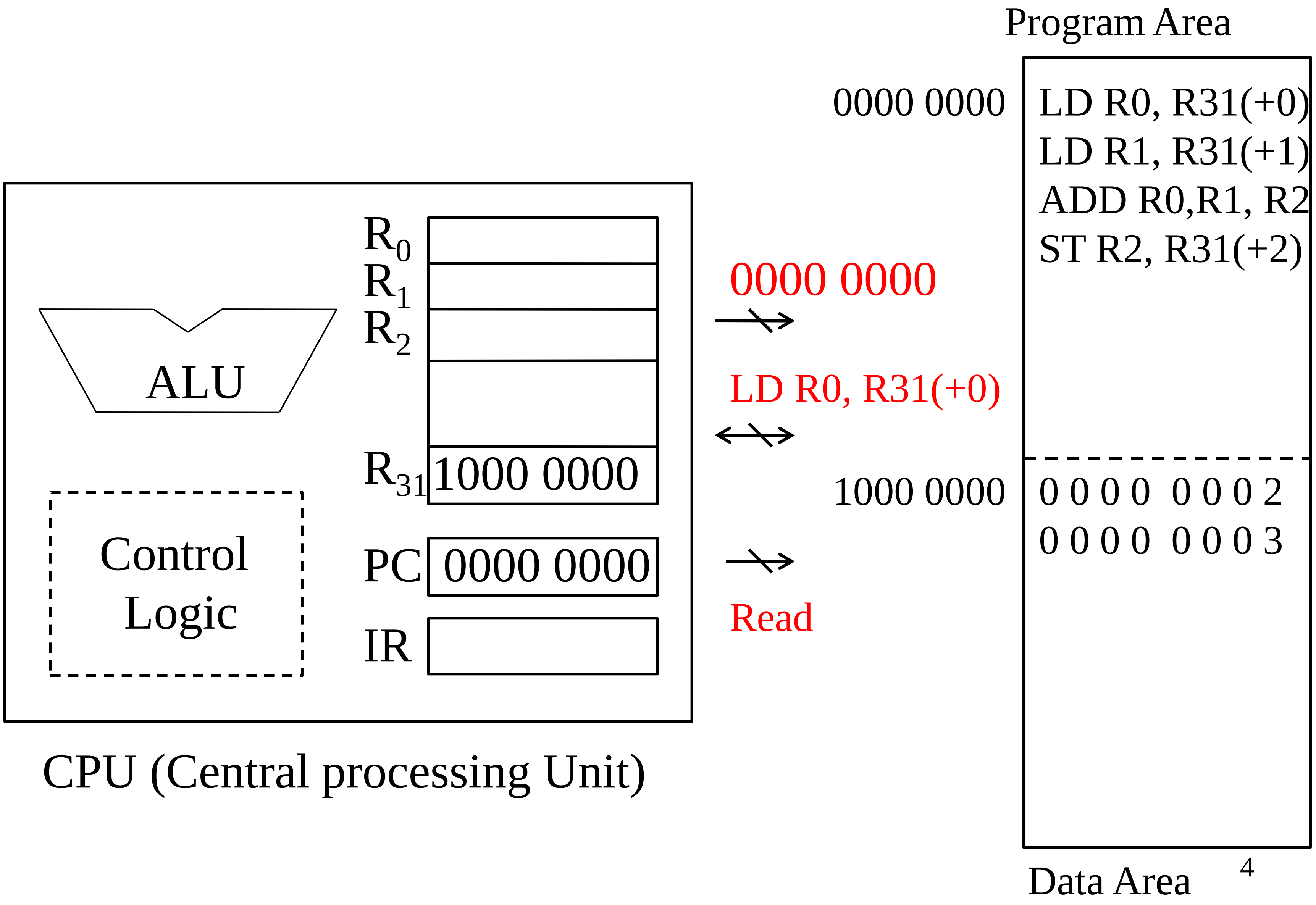
To Start Program Execution



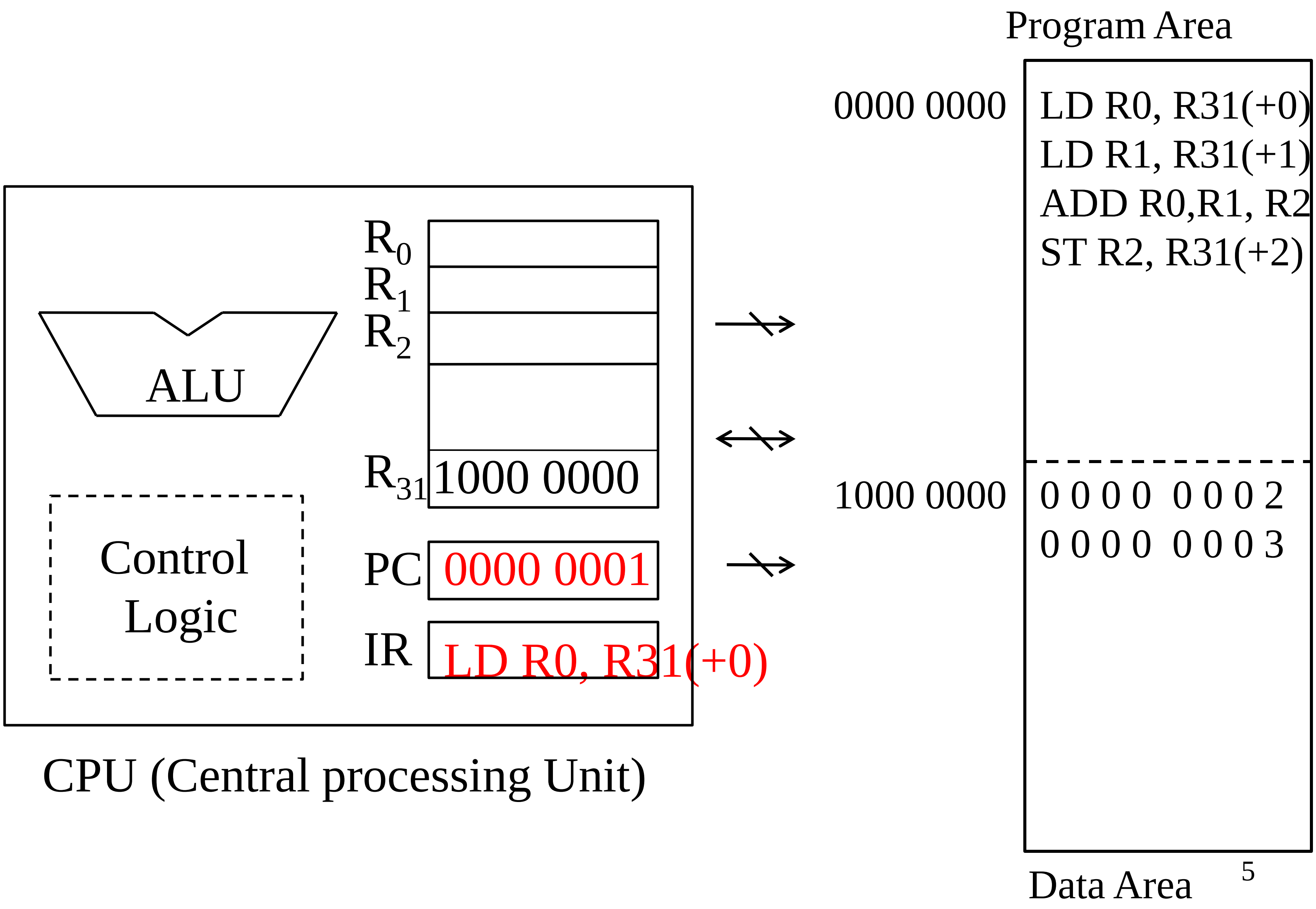
Program Execution

- Set up for execution
  - Program at known location
    - PC (program counter) = $0000\ 0000_{HEX}$
      - † PC: address of instruction to execute next
  - Data at known location
    - R31 = $1000\ 0000_{HEX}$
- Program for adding two numbers
  - 4 machine instructions

Step 1: Instruction Fetch (IF)



Step 1: Instruction Fetch (IF)



CPU (Central processing Unit)

Step 2: Instruction Decode (ID)

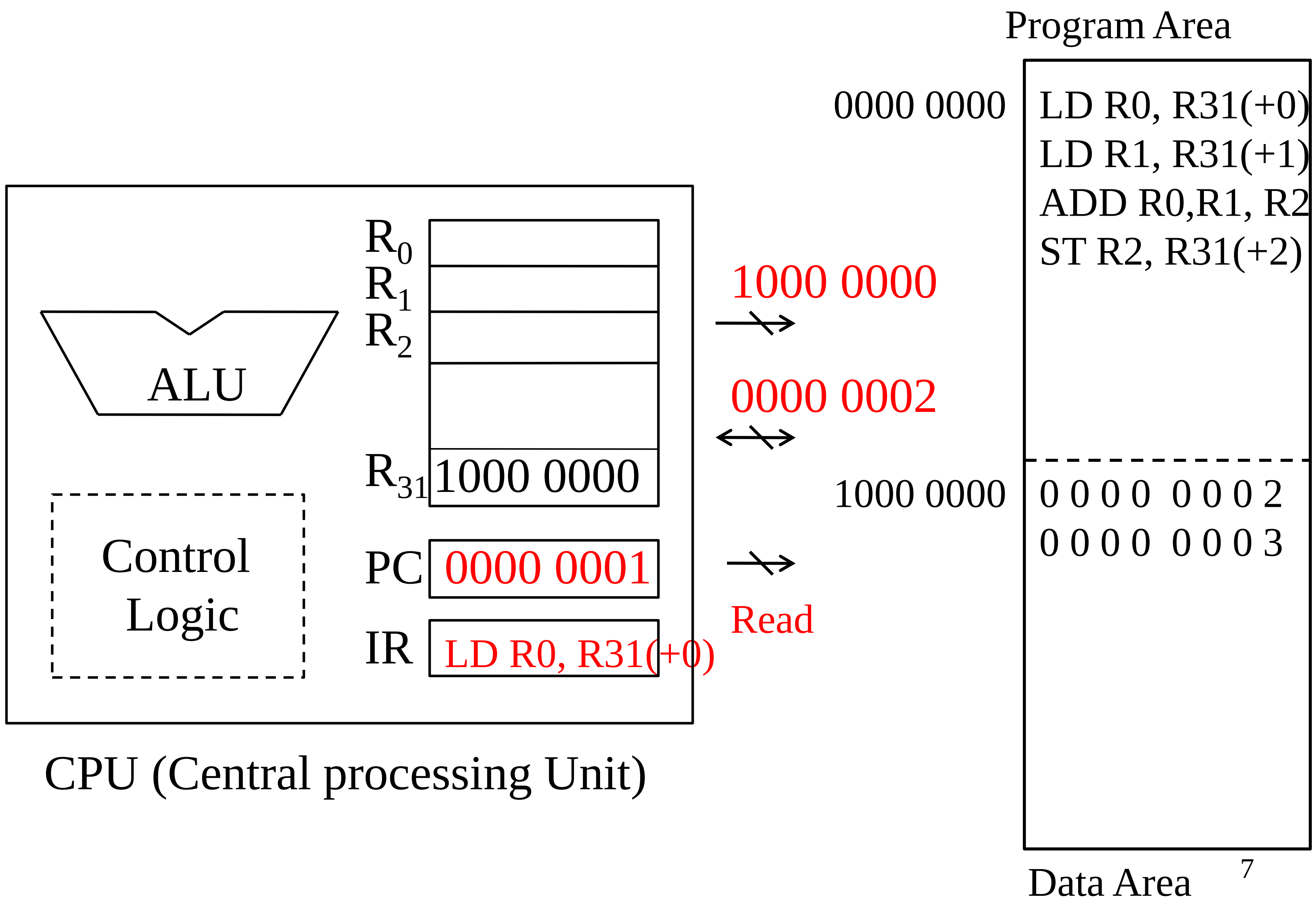
- Control logic in CPU
  - Examine the content of IR (i.e., fetched instruction)
  - Understand what it is

    LD, R0, R31(0)

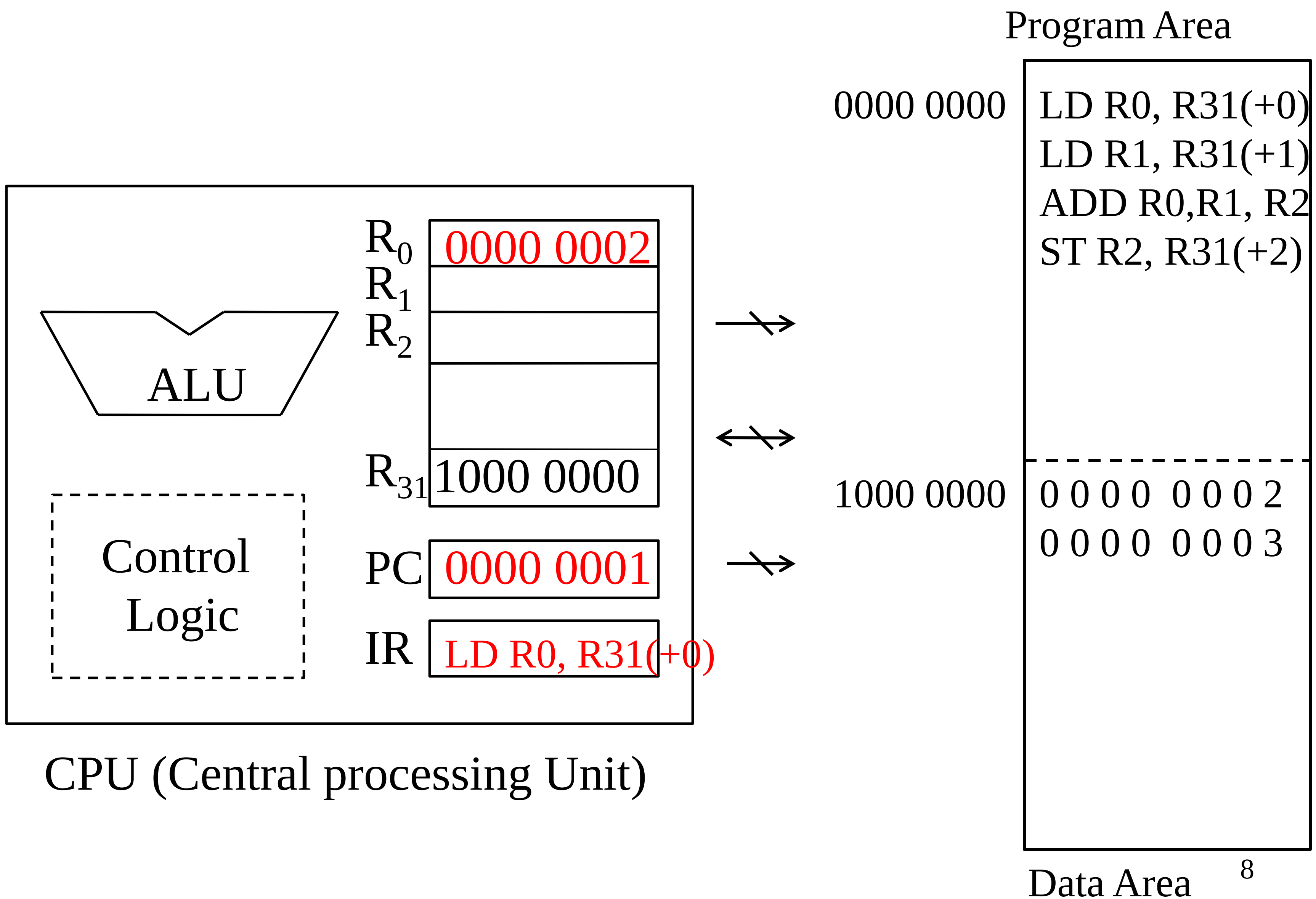
    $R0 \leftarrow M[\,R31 + 0\,]$

    $R0 \leftarrow M[1000\ 0000]$
  - Read what is in memory address 1000 0000 and copy it to register 0
    - Why not use absolute address ("LD, R0, 10000000")?

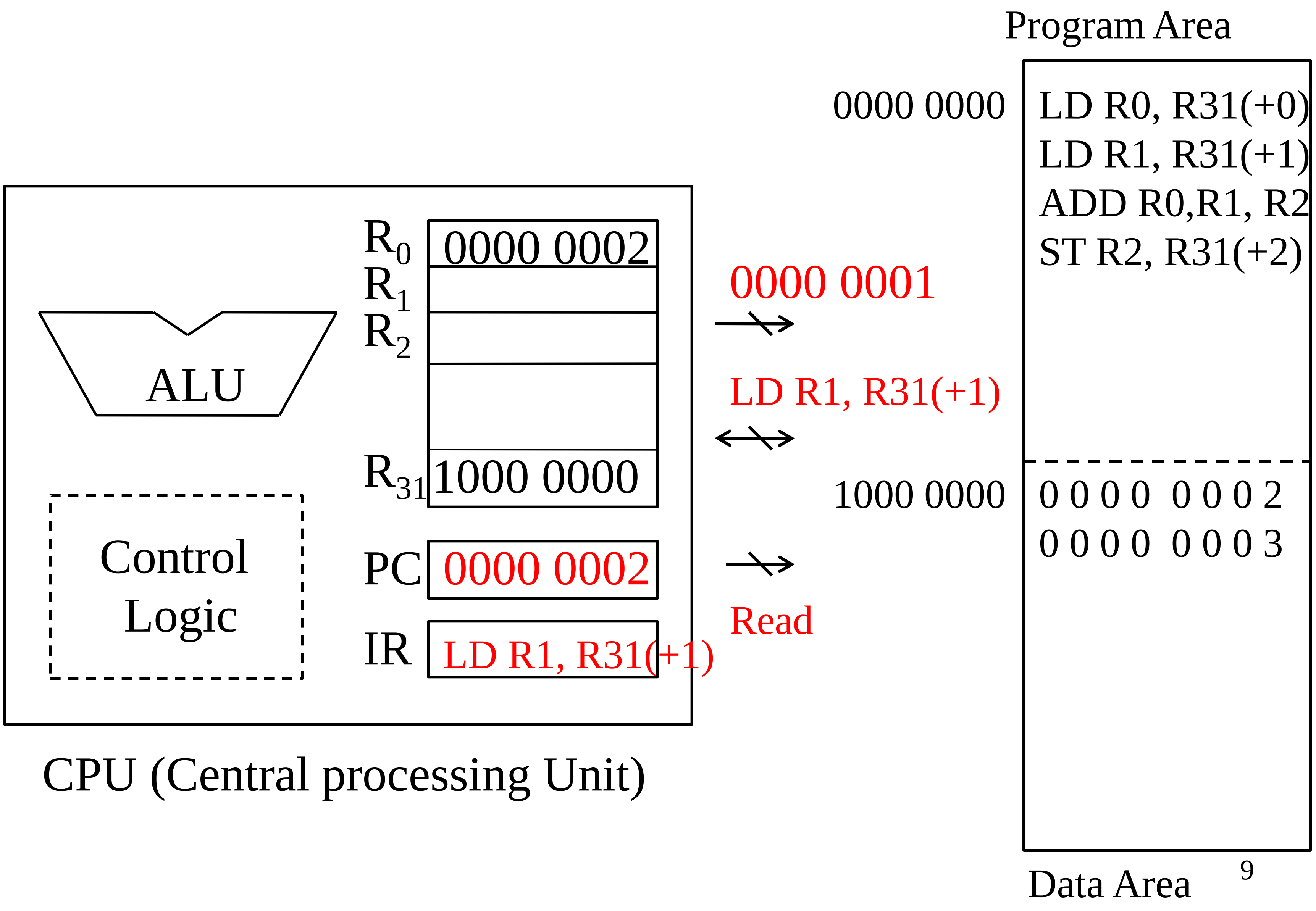
Step 3: Instruction Execute (EX)



Step 3: Instruction Execute (EX)

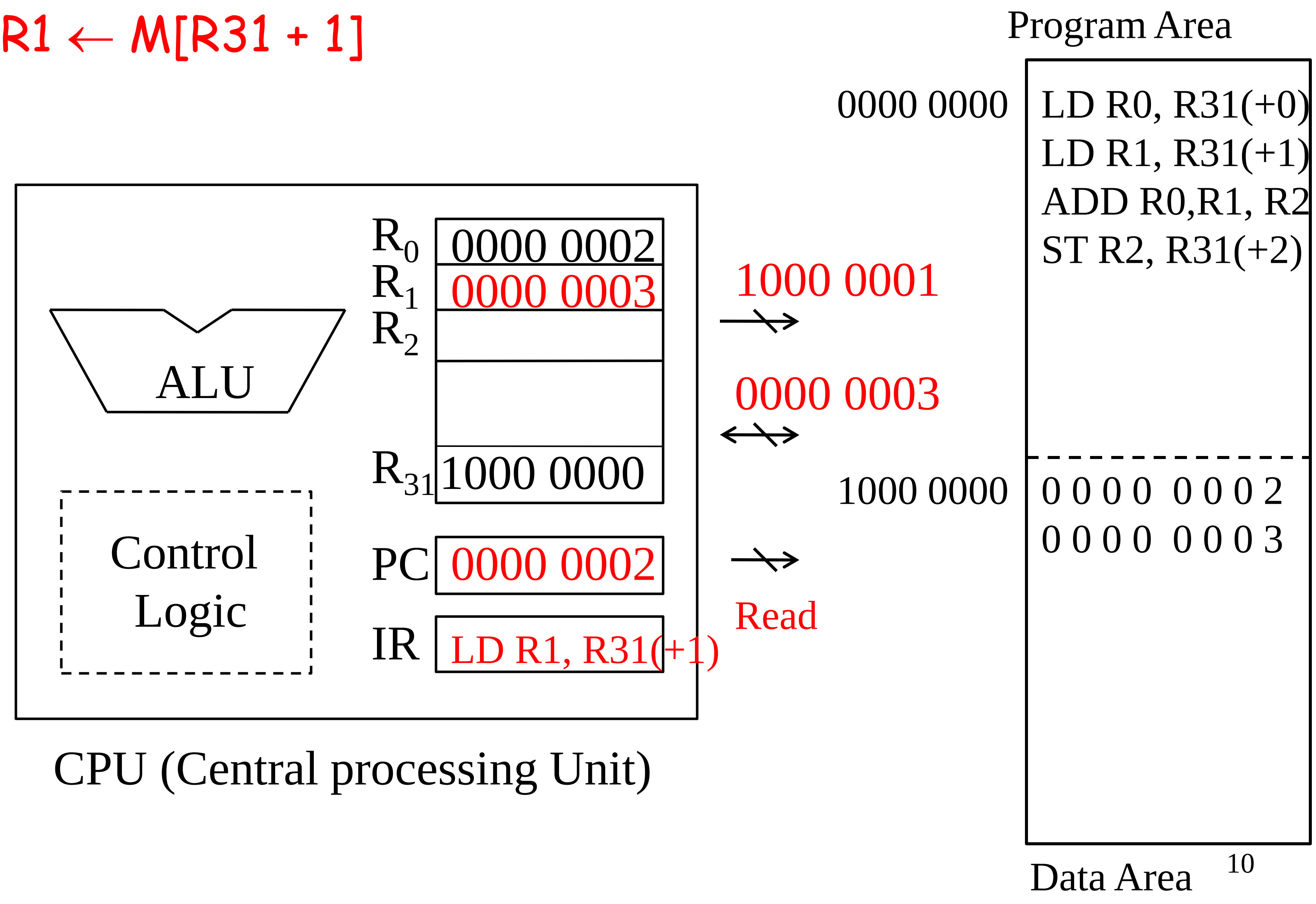


Fetch of 2nd Instruction

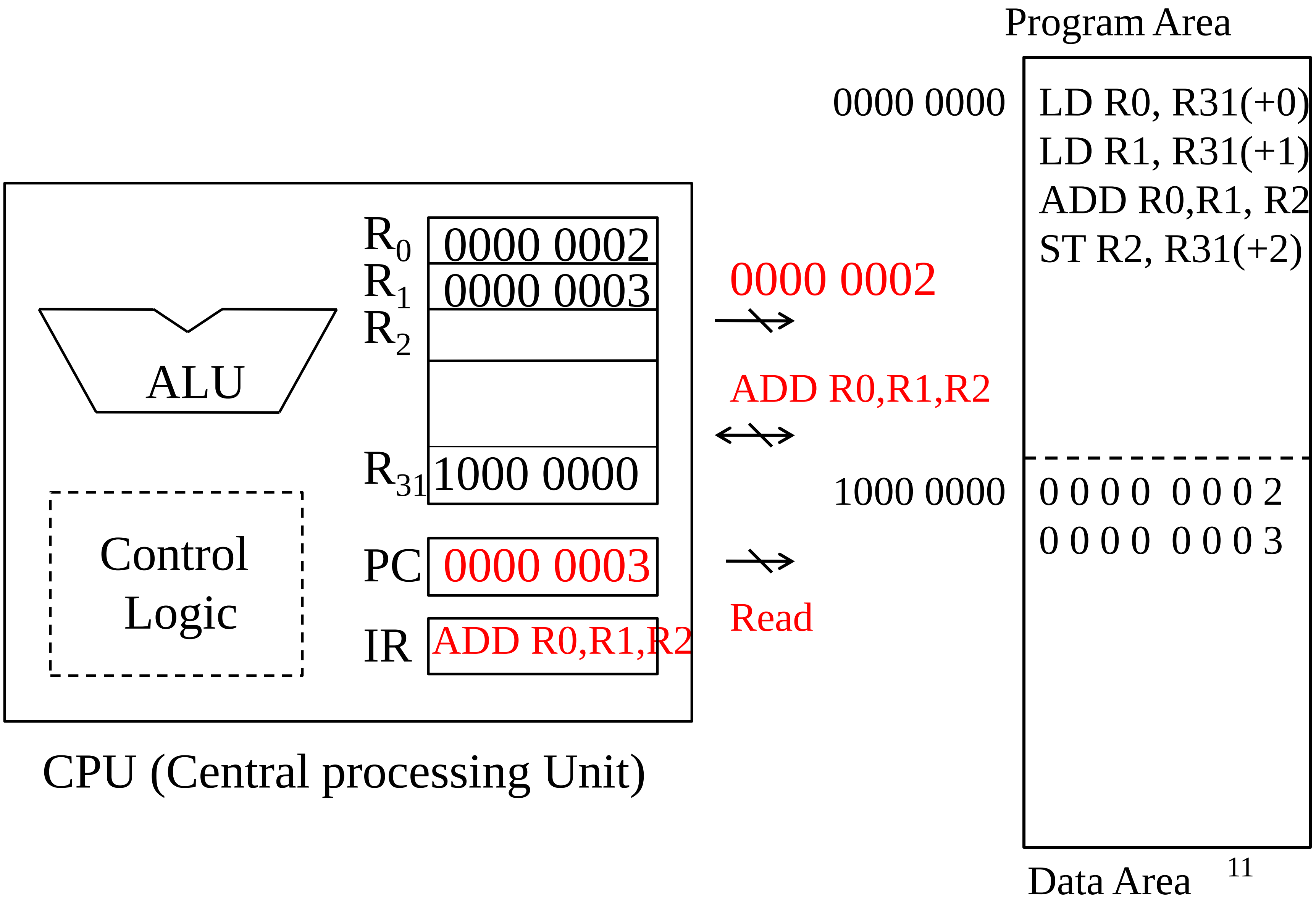


Decode and Execute 2nd Instruction

R1 ← M[R31 + 1]



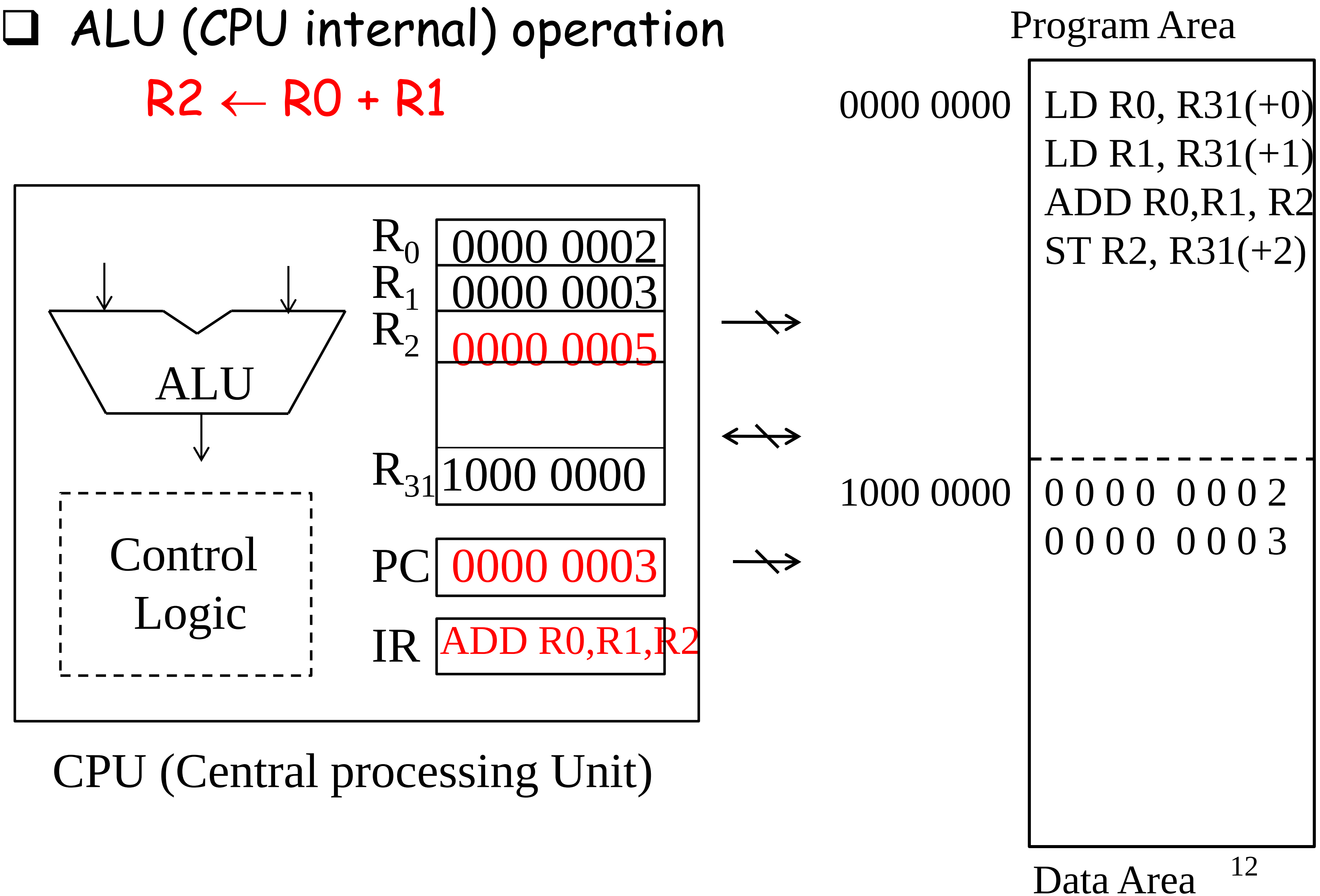
Fetch of 3rd Instruction



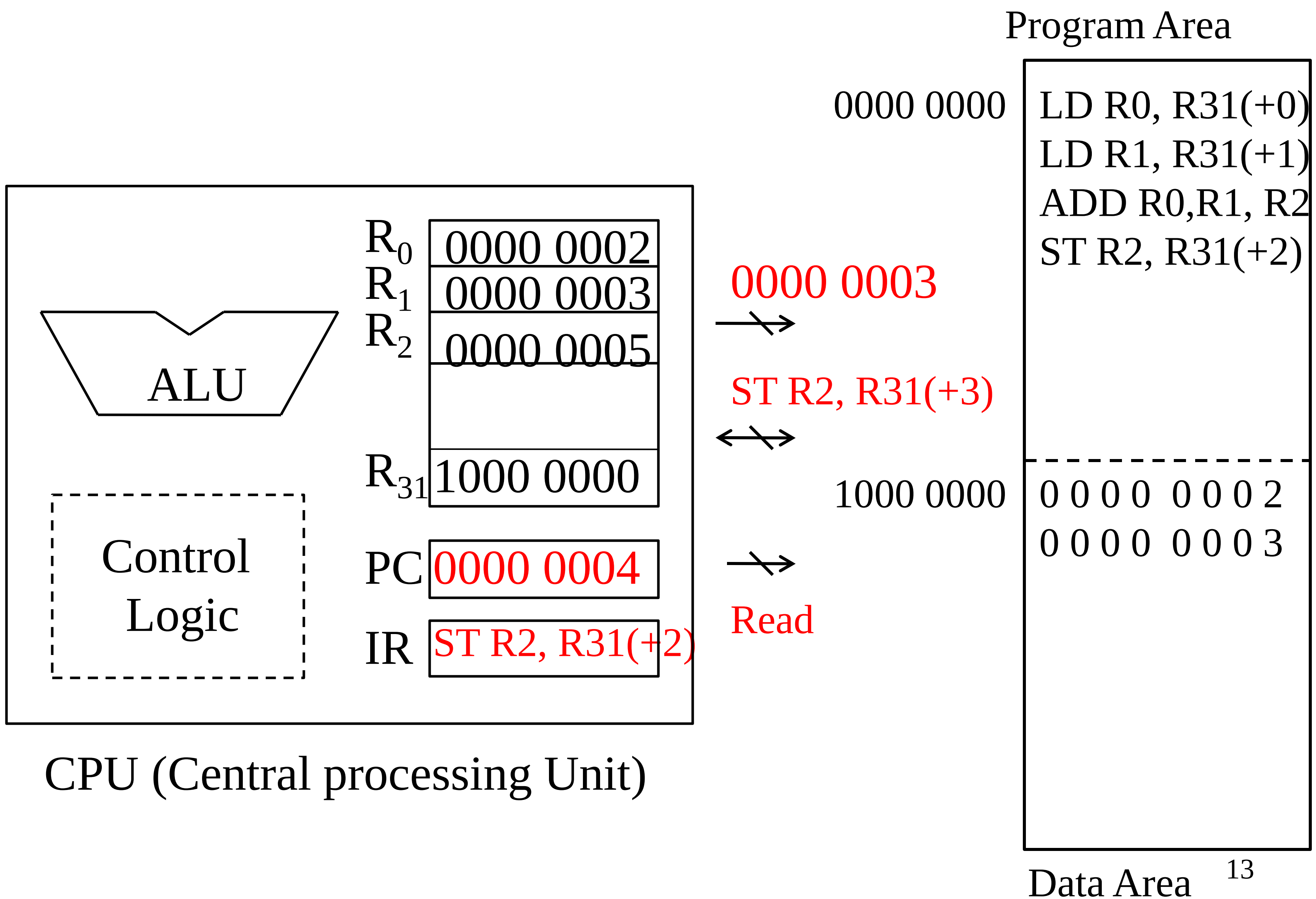
Decode and Execute 3rd Instruction

- ALU (CPU internal) operation

  $R2 \leftarrow R0 + R1$

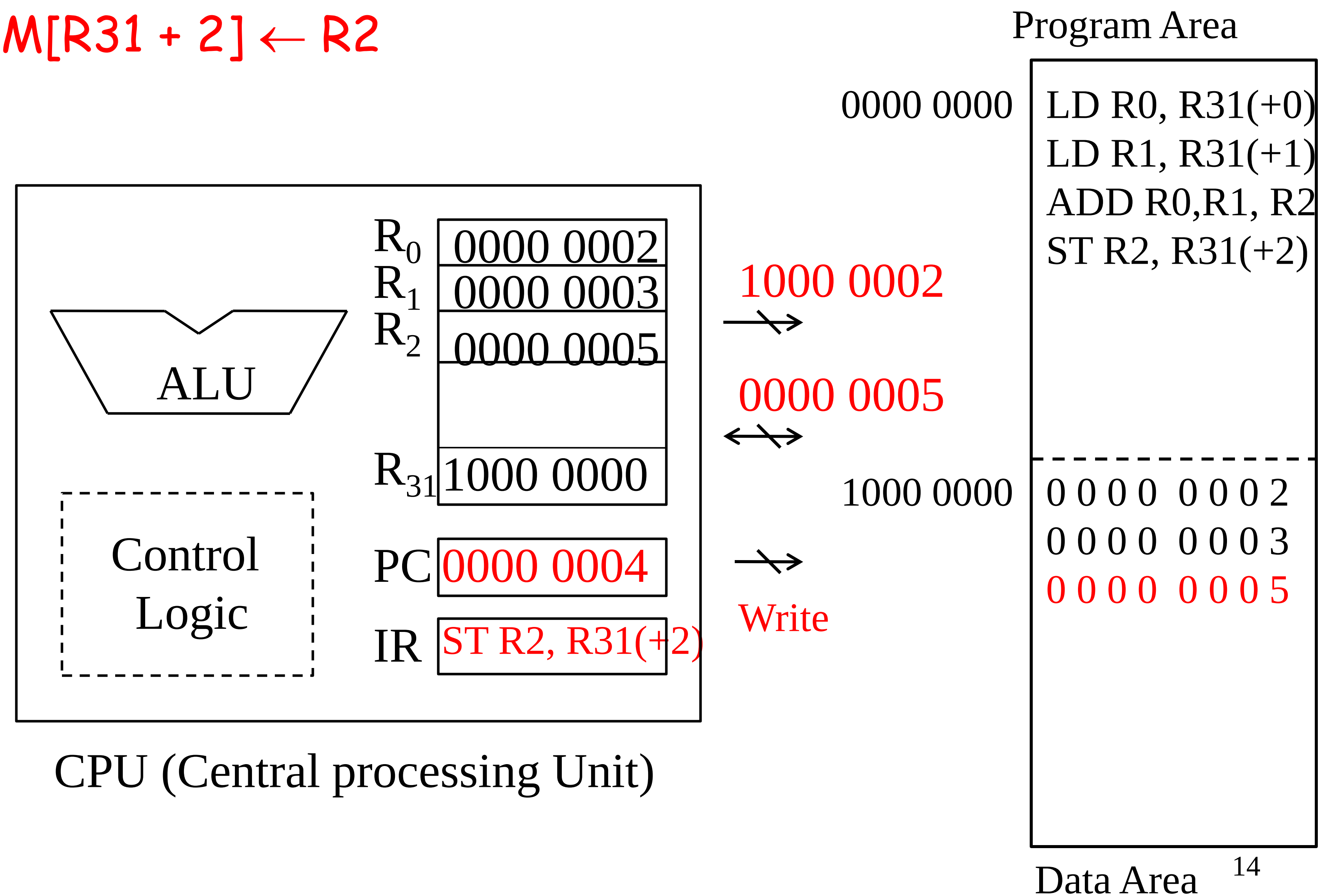


Fetch of 4th Instruction



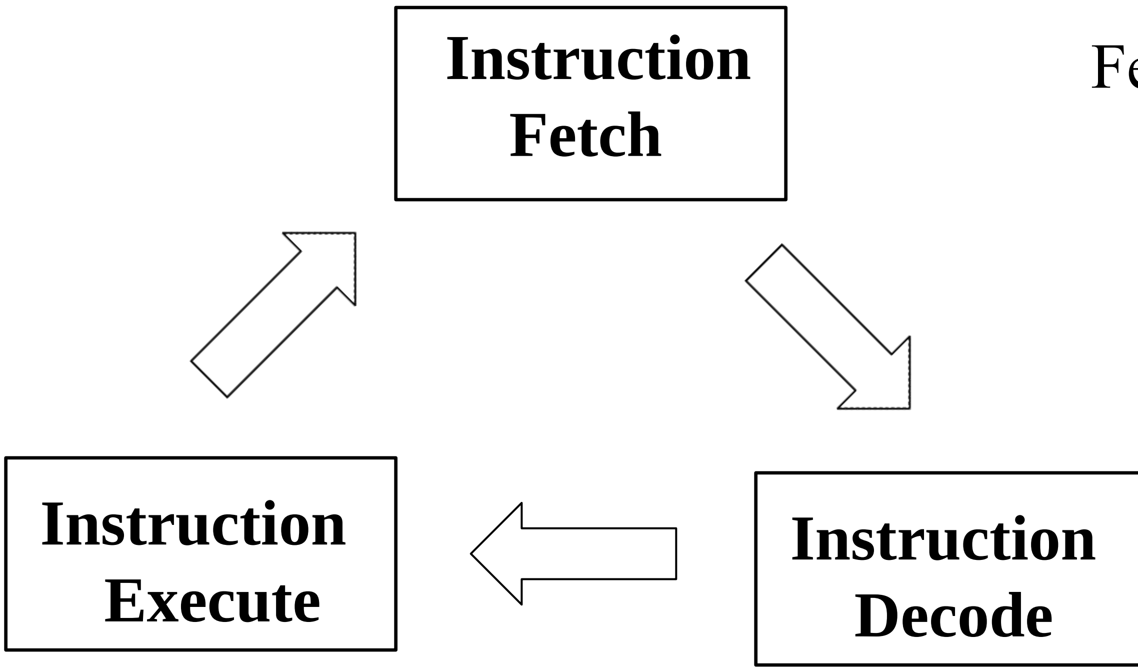
Decode and Execute 4th Instruction

M[R31 + 2] ← R2



Machine Cycle

- What is CPU? What is computer?

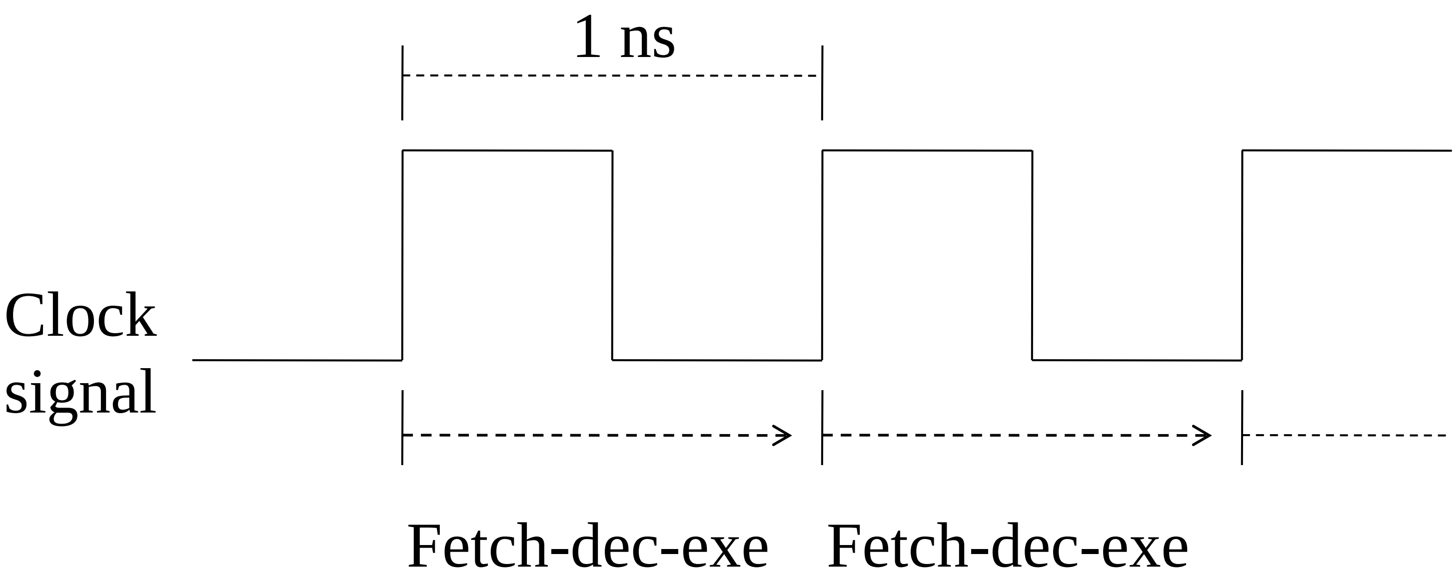


Fetch: $IR \leftarrow M[PC]$

$PC \leftarrow PC+1$

Time behavior

- Imagine 1 GHz CPU
  - Use 1 GHz clock
  - Clock period of 1 ns - how long is 1 ns?



ISA (Instruction Set Architecture)

Processors' external interface
(what programmers must know)

Final Piece of Machine Called Computer

- Hardware components
  - CPU, memory, I/O devices
  - ALU, registers, control logic in CPU
    - Digital logic design, abstraction
  - Program and data in memory
  - Notion of address (for memory locations, I/O devices)
- Operation
  - Fetch-decode-execute cycle, timing
- What is missing?
  - ISA (Instruction Set Architecture)
    - Kind of instructions needed to be a "computer"

Need Jump Instructions

- AND, OR, NOT (Boolean algebra)
  - CPU, memory, ALU (사칙연산, 논리연산)
- IF: jump (essential for problem solving, programming)

  IF (고객이 노인) 입장료 = 정가 * 0.7

  - Conditional jump instruction

```
SUB  R1, R9, #65          //  R1 = age - 65
JUMP-NEG  R1, #1          // if negative, PC ← PC + 1
MULT  R3, R3, #0.7        // if negative, skip (no discount)
    ----
```

  - Six types of jump instructions: $=, \neq, >, <, \leq, \geq$

Instruction Set (Architecture)

- ALU instructions
  - add, sub, mult, div, and, or, not // ADD R1, R2, R3

이 세가지로 인간의 생각 표현 가능 (경험적으로 증명됨)

- Data transfer instructions (for external memory, I/O)
  - load, store // LD R1, R31(#1)
- Jump instructions
  - jump if =, ≠, >, <, ≤, ≥

† With these, we have been computing for 70 years!

† Power of AND, OR, NOT, IF by G. Boole

– "The Laws of Thought" versus machine called computer

Machine and Programming

- Machine called computer
  - Function determined by program
  - Service provided by CPU
    - A few tens ~ a few hundreds of instructions
      - Use them for problem solving (programming)
- What is programming?
  - Telling computer what to do
  - Programming with machine instructions
    - Set of instructions define machine language

Assembly vs. Binary Language (참고자료)

사람의 인지를 위해서 만들어짐

- ALU instructions: 32-bit long (opcode, operands)

ADD R1, R2, R3

0000 1000	00001	00010	00011	
Opcode(8)	Reg(5)	Reg(5)	Reg(5)	unused(9)

OR R2, R4, R6

0000 1100	00010	00100	00110	

- The two are identical – both called machine language
  - Simple 1:1 translation
  - Assembly language: mnemonic

Assembly vs. Binary Language (참고자료)

- Jump instructions: 32-bit long

JUMP-NZ R1, 8

0100 0000	00001	000 0000 0000 00001000
Opcode(8)	Reg(5)	Jump distance(19)

JUMP-POS R2, 4

0110 0000	00010	000 0000 0000 0000 0100

- What is instruction decoding?

Assembly vs. Binary Language (참고자료)

- Load and store instructions: 32-bit long

LD R1, R31(2)

0000 0010	00001	11111	00 0000 0000 0010
Opcode(8)	Reg(5)	Reg(5)	Constant(14)

ST R2, R31(4)

0000 0011	00010	11111	00 0000 0000 0100

- Why not use absolute memory address?

R31(2) vs 1000 0002
1000 0002가 아닌 이유??
32bit이기 때문에 fetch가 2번 일어나게 된다. 따라서 instruction을 32bit으로(fetch 1번) 맞추기 위해서 R31(2)으로 표현

Machine Called Computer

- Function determined by program
- Doing useful work with computer require:
  - Programming, software design/development
    - The term problem-solving

Programmer

Machine-level language

ISA (Pentium, PowerPC)

Machine (CPU)

Engineering = Building Abstractions

- Programmer
  - Use machine instructions for programming
    - "Interface" (사용법)
  - Not know computer design/organization/operation
    - "Implementation" (설계/구조/동작)
- Computer (CPU) 를 포함한 모든 공학 도구/물건
  - Implementation 몰라도 interface 알면 사용 가능
    - † Fundamental concept of abstraction
- Complex engineering product (e.g., 컴퓨터, 자동차, 건물)
  - 작은 부품들 사서 복잡한 모듈 만듬, 모듈들로 더 복잡한 ...
    - † Recursive abstraction building

Processor (Machine Called Computer)

- Why do you buy it? For what service?
- What kind of interface does it provide?
- What kind of abstraction does it provide?
  - ISA
- What is processor?
  - What implements ISA
- Does it look intelligent?
  - Once the software for solving differential equation is developed, then the problem is solved!
  - Reliable, very fast

Engineering Design (and the term "Architecture")

- ❑ Marketing and requirements analysis
  - New (or existing) product; can we sell it?
- ❑ Design and implementation
  - External interface internal implementation에 대해 정확히 알아야 interface 설계 가능
    - How the user will use the product
  - Internal implementation
- ❑ Testing and release
- ❑ Continual enhancement

† Architecture (major interface), architect, abstraction

What is Computer Science?

- Study of problem-solving with computational devices
- What kind of problem did we solve?
  - How to build computer (i.e., machine that compute)

Machine-level language

Programmer

Machine (CPU)

ISA (Pentium, PowerPC)

† Architecture (major interface), architect, abstraction

Stored Program Concept

- Old term: von Neumann architecture
  - Fetch-decode-execute

Modern Digital Computer

- Gradual evolutions to meet human computational need
  - Capabilities, design techniques, supporting technology
- ENIAC (1943-1946)
  - First fully-electronic, general-purpose computer
  - U. Penn., Eckert, Mauchly
  - Program not in memory, vacuum tube program을 기계적으로 실행 (vaccum tube을 손으로 연결)
- What was a brilliant idea?
  - Stored program concept in 1945
  - Natural consequence: fetch-decode-execute
    - von Neumann architecture/bottleneck cpu 빨라져도 memory 느리다.
    - C. Babbage's work in early 19C

ENIAC (1943-1946)

Image of ENIAC:

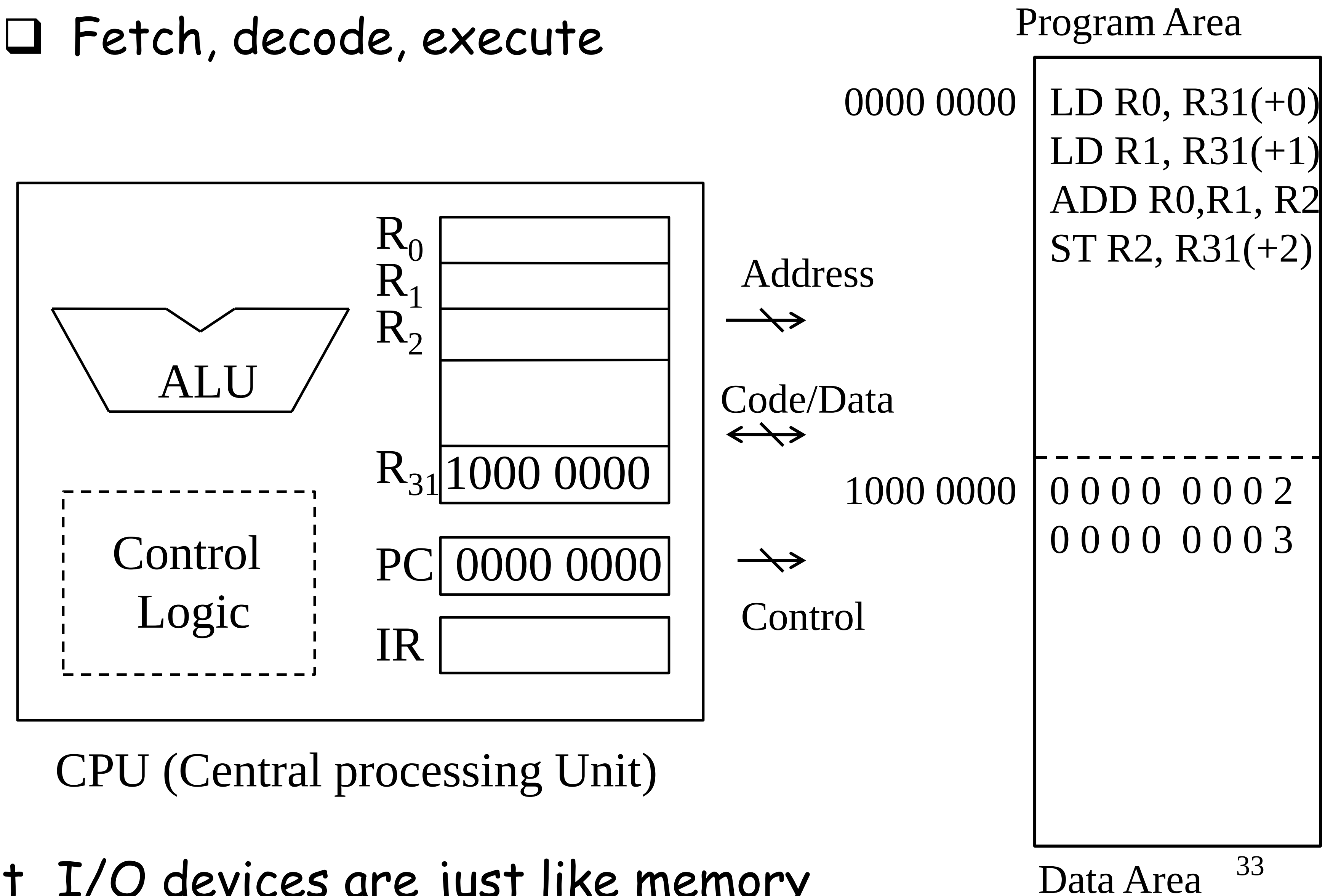
http://en.wikipedia.org/wiki/File:Classic_shot_of_the_ENIAC.jpg

Image of ENIAC:

http://en.wikipedia.org/wiki/File:Eniac.jpg

Stored Program Concept

- Fetch, decode, execute



- I/O devices are just like memory

Modern Digital Computer

- Completed by stored program concept in 1945
  - First stored program computers
    - UNIVAC I (1951), EDVAC (1952)
    - Earlier, smaller-scale British/US computers
- Next 60 years of evolution for performance
- Search continues
  - Non-von Neumann architectures
  - Alternate forms of computing
    - Biological, optical, quantum
  - Possibly, new definition of "computing"

UNIVAC I (1947-1951)

Image of UNIVAC I:

http://en.wikipedia.org/wiki/File:UNIVAC-I-BRL61-0977.jpg

Image of UNIVAC I:

http://en.wikipedia.org/wiki/File:Museum_of_Science,_Boston,_MA_-_IMG_3163.JPG

EDVAC (1945-1952)

Image of EDVAC:

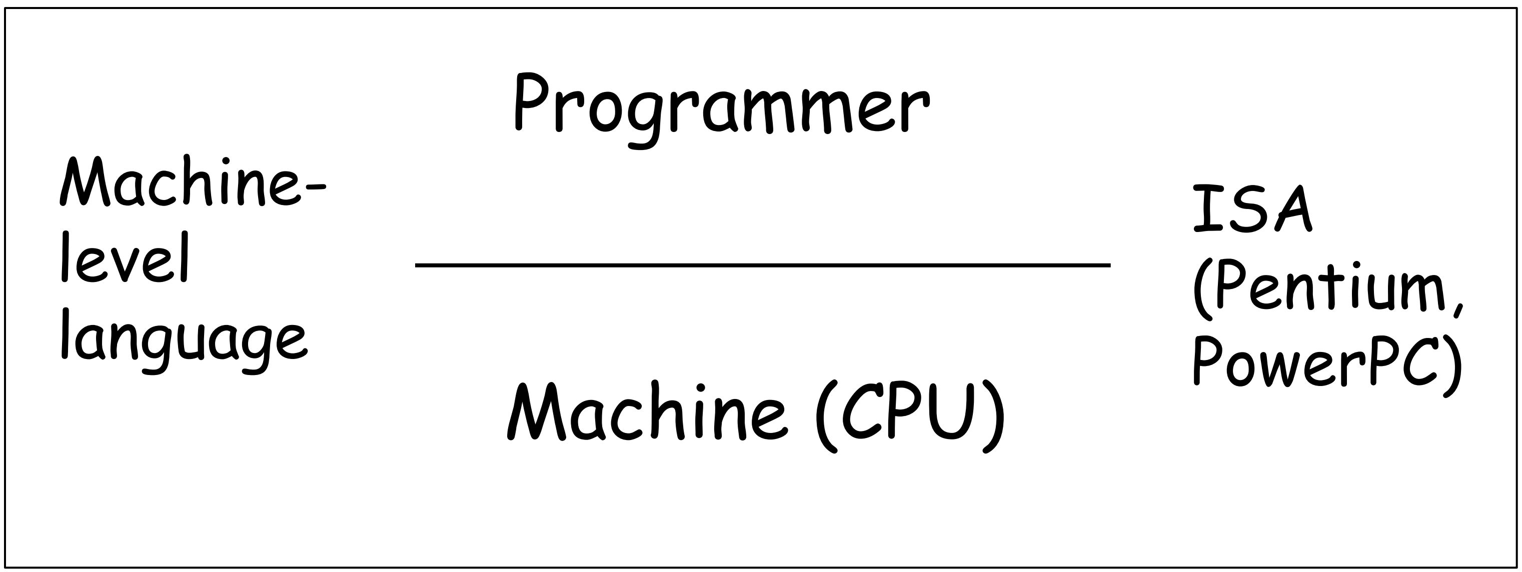
http://en.wikipedia.org/wiki/File:Edvac.jpg

Machine, Software, Internet

(not separate entities but combined whole)

Programming

- Telling computer what to do
- Machine define a (very low-level) language
  - Productive?



Program to add two numbers

```
1000    LOAD  R1, (2000)    //  load from address 2000 to R1
1004    LOAD  R2, (2004)    //  load from address 2004 to R2
1008    ADD  R3, R1, R2     //  add
100C    STORE  R3, (2008)   //  store result to address 2008
1010    HALT
        …
2000    25                  //  first operand
2004    31                  //  second operand
2008    -                   //  sum of two operands
```

C program:

```
int  a, b, c;
a = 25;
b = 31;
c = a + b;
```

Two Major Interfaces in Computer

Developers

High-level language — C, C++, Java

Compiler

(executable) Machine-level language — ISA (Pentium, PowerPC)

Machine (CPU)

- Computer language vs. natural language?
- Abstractions supported by ISA and HLL adequate? (skip)

Two Major Interfaces in Computer

- Two major interfaces
  - High-level language
  - Machine (or assembly) language
- Two major products
  - Processor
  - Compiler
- What kind of service (or abstraction) do they provide?
  - † The term “abstraction” in engineering
- If you must choose one, what will be your choice?
  - Why do we program in high-level language?

Programming Paradigms/Languages
(참고자료)

Procedural:

- Fortran, 1957 and after
- Algol, 1958
- Cobol, 1960
- Basic, 1964
- Pascal, 1970
- C, 1973
- Ada, 1983

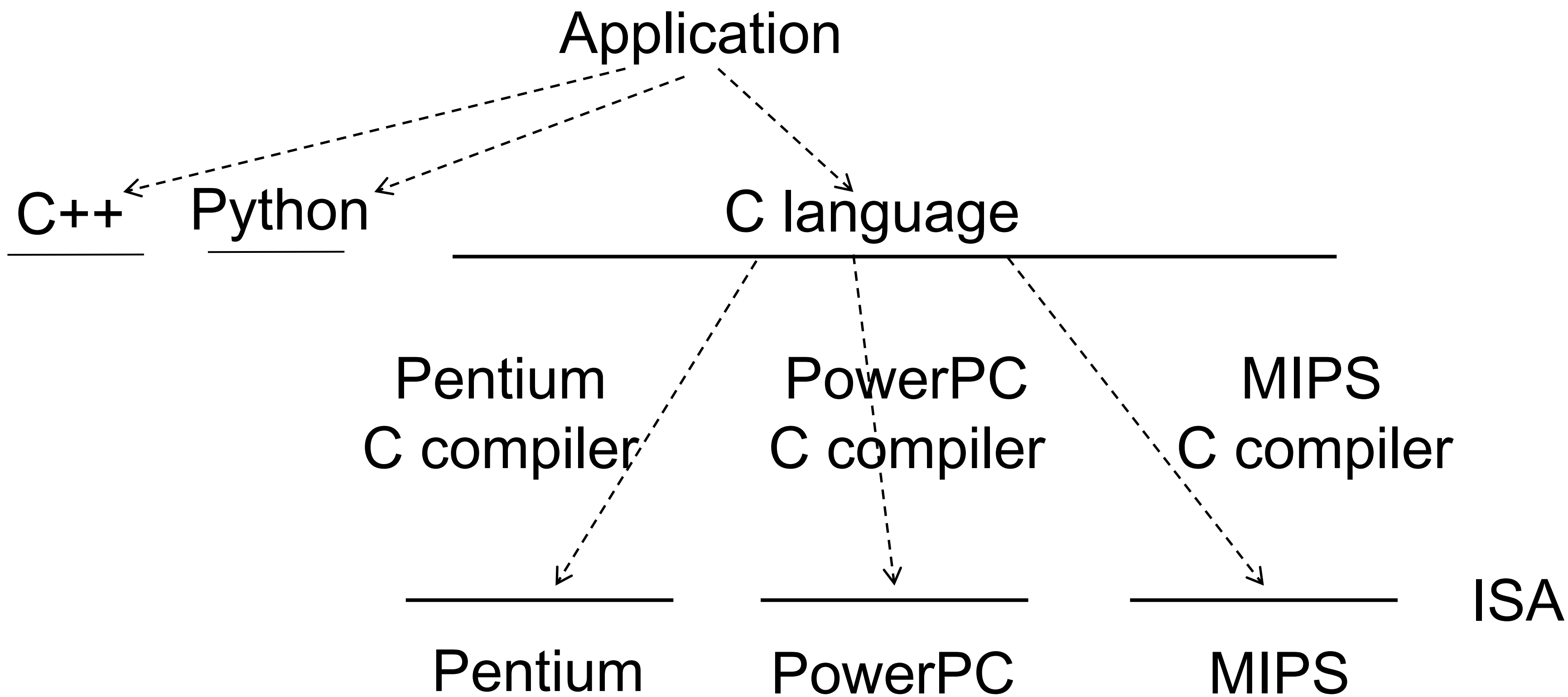
Object-oriented:

- Simula, 1967
- Smalltalk, 1980
- C++, 1985
- Perl, 1987
- Python, 1990
- Visual C++, 1993
- PHP, 1994
- Java, 1995
- Ruby, 1995
- C#, 2002

Functional: Lisp (1958)

Logic: Prolog (1972)

CPU Dependency



- You buy compiled code (e.g., Word for Pentium)
- When upgrading your PC, you choose Pentium (독점성)
  - Similar dependency exist for OS also

What is Computer Science?

- Study of <u>problem-solving</u> with <u>computational devices</u>
- What kind of problems did we solve?
  - How can we build a machine that computes?
  - How can we boost productivity in programming?

Million Lines of Source Code

Developers

Many design steps (manual)
to fill semantic gap

High-level language

C, C++, Java

Compiler

(executable) Machine-level language

ISA (Pentium, PowerPC)

Machine (CPU)

Software Complexity
(data from Wikipedia)

- Operating System (OS)
  - 1 billion source lines of code (SLOC) in C++
  - Debian 2.2 (55M): 14,005 man-years, 1.9 billion US$
- How do we go about this?

Year	OS	SLOC (Million)
1993	Windows NT 3.1	4-5
1994	NT 3.5	7-8
1996	NT 4.0	11-12
2000	2000	> 29
2001	XP	45
2003	Server 2003	50

Year	OS	SLOC (Million)
2000	Debian 2.2	55-59
2002	3.0	104
2005	3.1	215
2007	4.0	283
2009	5.0	324
2005	Mac OX X 10.4	86

What is Computer Science?

- Given Pentium/C, what kinds of problems did we solve?
  - How to kill: solve differential equations
  - How to make my computer easier to use (OS)
  - How to manage the information on things (database)
  - How to connect all computers in the world (Internet)
    - Given Internet, how to share information (web)
  - Given the web, how to find what I want (search engine)
  - Given web, how to sell my products (e-commerce)
  - How to make documentation/publishing easier (Word)
  - Yet to solve: big data challenge
    - Bioinformatics, SNS data

Computer Science and Engineering

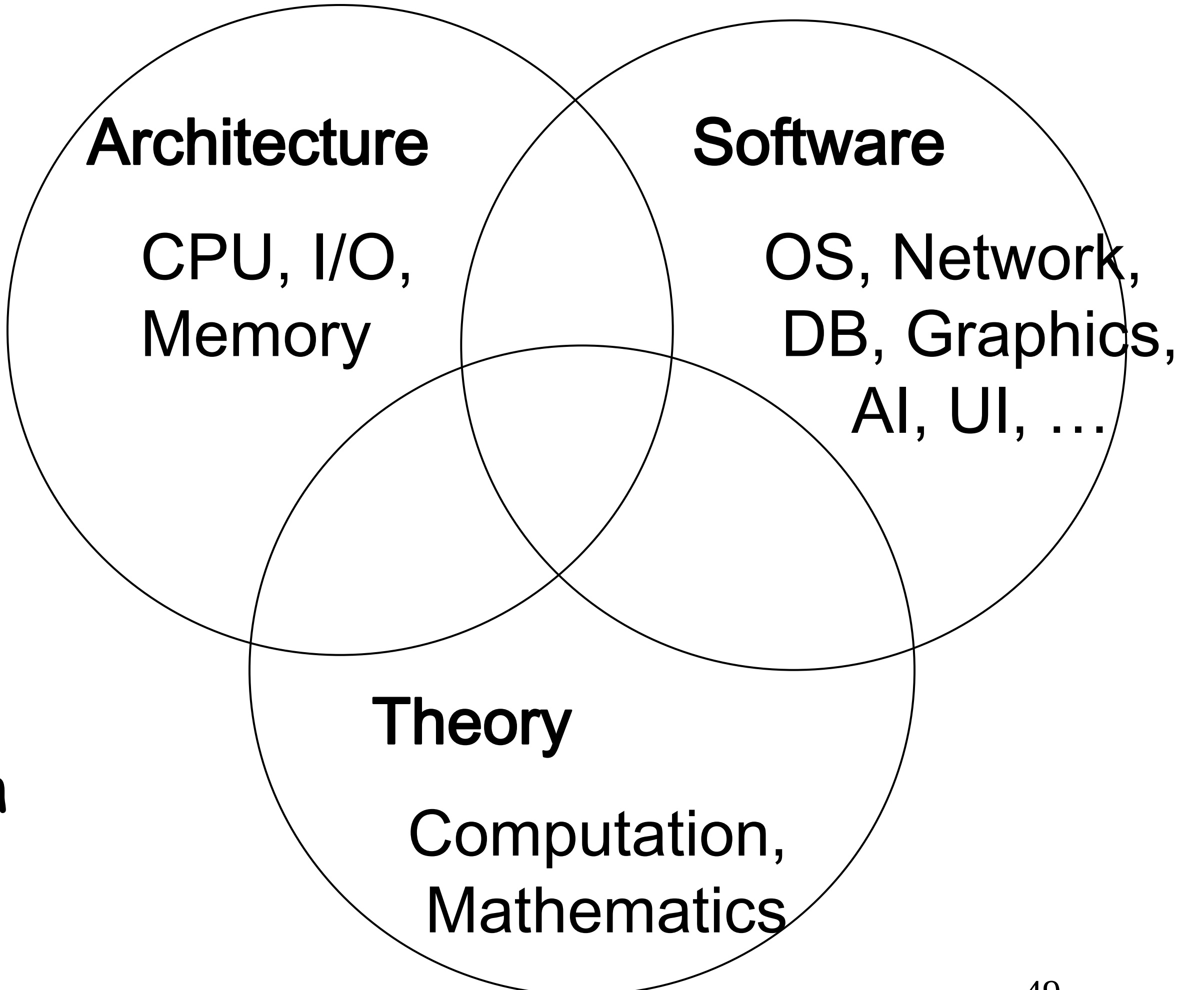
- Science vs. engineering
  - Science pursue a major new piece of knowledge
  - Engineering is about tools
    - Accumulation of knowledge facilitate engineering
- Recognition of problems, establish mathematical approaches
- Programming is about tools, processor is a tool
  - Tool development
    - Smaller-scale problem-solving by many engineers

† CS, CSE, CE

Computer Science and Engineering

- Core IT

- IT convergence
  - Management
  - Finance
  - Law
  - Automotive
  - Education
  - Transportation
  - Silver, …



IT Convergence

- 인간의 지식과 기술이 software 형태로 집약됨(국가경쟁력)
- Not by CSE major but by all elites
  - How can we build software infrastructure?

† SW 비전공자 양성 과정 (현재 준비 중임; 잠정적 자료임)
  - 재학 중 본인의 전공과목과 함께 SW 기초교육 받음
  - 신청자 중 선발
  - 2-4 학년 동안 학기당 2과목씩 총 12 과목 이수
  - 방학 중에는 SW 현장교육과 인턴 기회
  - 통섭형 인재

IT Gold Rush (in USA)

IT Gold Rush in USA

- 1950 through 1970: innovation and transition
  - Transistorized computers, start using IC
  - Many big companies jump into computer ventures
    - IBM System/360 in 1965
  - Software technology
    - OS (time sharing, virtual memory, file system, …)
    - High-level programming languages, applications
  - Computers penetrate into industry
    - Minicomputers in mid 1960s
  - ISV (Independent software vendor): 2,800 in 1970
  - Service bureaus ("cloud" today)

IT Gold Rush in USA

- 1970s and 1980s: extraordinary evolution and growth
  - Processor, memory: semiconductor VLSI technology
    - Smaller/faster, exponential growth (Moore's law)
  - UNIX and C language (1969/1973)
    - Open/free OS source, renaissance of CPU design
  - Full bloom of minicomputers, relational database
  - Personal computer revolution (since 1975)
    - Whole new "shrink-wrapped software" business
    - Microsoft, Apple, Lotus, …
  - "Silicon valley"; computer networks; America Online
  - Computer/software penetrate into all vertical markets
- † BY 1990, USA far ahead of Europe and Asia

IT Gold Rush: Web, Internet

- Success of Internet: one and the only network
  - TCP and IP standard protocol of ARPAnet in 1983
  - Web: killer application [Tim Berners-Lee, 1989-1991]
  - Graphic browser by Andreessen/Netscape (1993/1994)
  - U.S. transition for commercial use (1991-1995)
- Internet, web, PC explosion in 1990s
  - Electronic commerce, information revolution
  - New business models
    - Yahoo,Google,Twitter/Facebook,YouTube,Wikipedia
- IT bubble bust in 2000/2001
  - Smartphones in 2000s
- IT convergence: IT-driven changes in all industries

Information Revolution

- Current economic/social/technological trends
- Social perspective
  - Control of info., propagation speed, people's reaction
- Electronic commerce
  - Distribution channel of goods and service
    - Eliminate distance; must be globally competitive
    - Only one economy and only one market
  - † Computer to steam engine, e-commerce to railroad
- Routinization and new business models
  - Knowledge leading factor in production
- Part of Scientific/Technical Revolution since mid-20C
  - Industrial Revolution: series of changes over 200 years
    - Initial quick changes, then hard innovations

Classes of Computers

Two Types of Computers

- General-purpose computer (범용컴퓨터)
  - 인간이 주는 (다양한 종류의) 프로그램을 실행함
    - PC, 한양대 데이터베이스 서버

새로운 프로그램이 나오면 잘 돌리기 위해 진화
1. general-purpose network - internet
2. special-purpose network - 전화

- Embedded computer (내장형컴퓨터)
  - Machines 과 결합하여 다양하고 강력한 자동형 기계 형성
    - 항공기, 우주선, 자동차, 청소기, drone, 로봇, ...
  - Many different types, so many of them
    - 프로그램은 한 가지로 고정되어 있음
  - 컴퓨터는 기계를 조종하는 머리 역할 수행
    - 컴퓨터는 작고 기계에 안에 내장되어 잘 보이지 않음

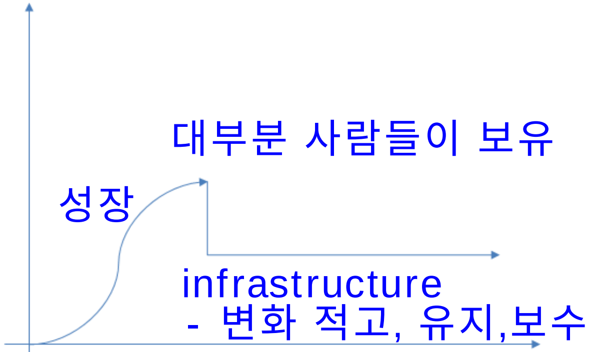
† Special-purpose computer, dedicated computer

Classes of Computer Applications

- PCs (or desktops)
  - Good performance for single user at low cost
  - Third-party software 부품사듯이 구매
- Servers
  - Large workload: single complex application or many small jobs (supercomputers, web servers)
  - Software from another source (database or simulation system), but customized
- Embedded computers
  - Fixed applications integrated with software, delivered as single system
  - Widest range, not seen as computers, cost and power

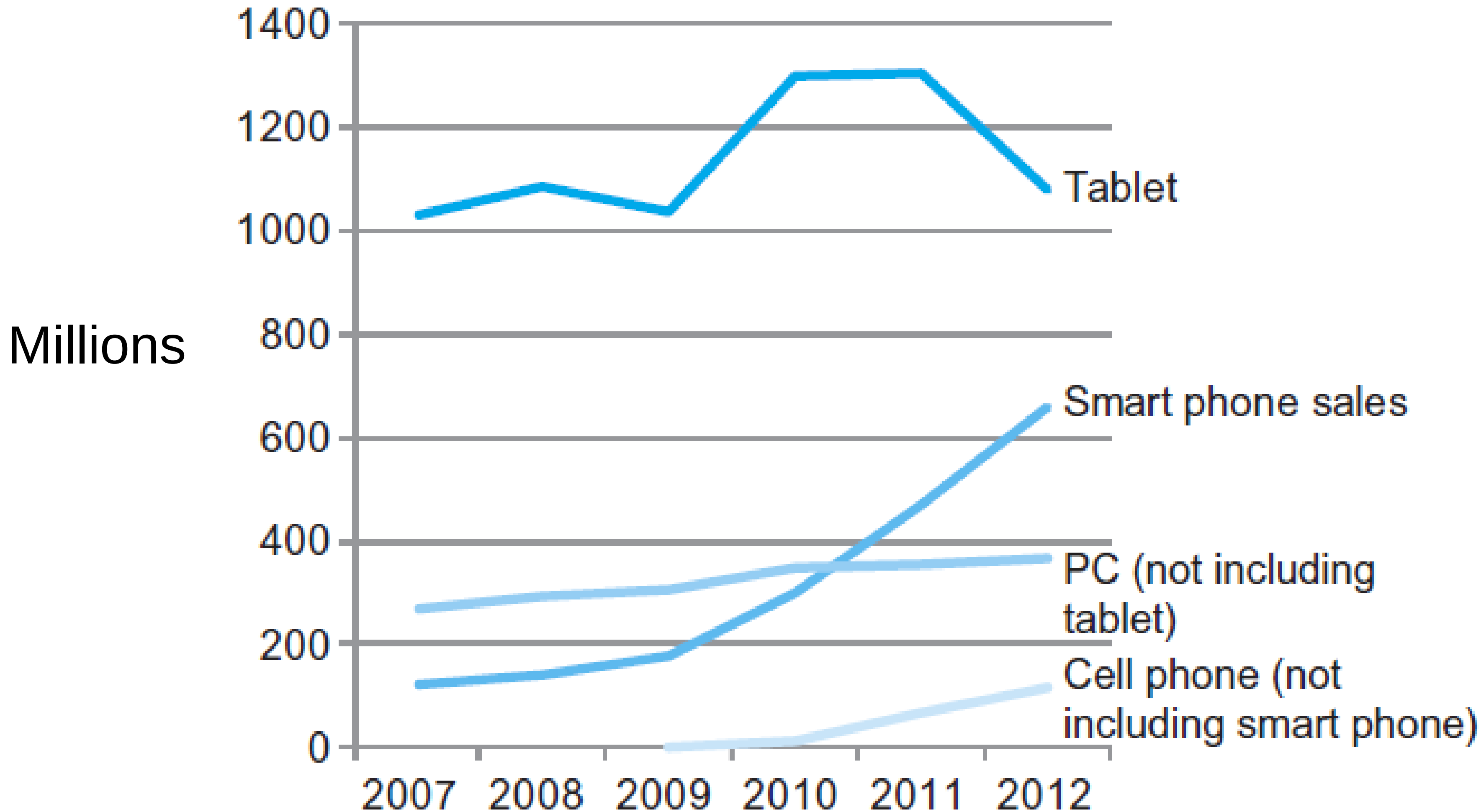
Evolutions Turning into Infrastructure

- Introduction of electronic computers (around 1950)
  - Evolve into business mainframes
    - Create new markets and billionaires
    - HW + OS + service (+ applications)
- Minicomputers, open Unix servers (and LAN) – 1970s
- PC (and Internet) – 1980s and 1990s
  - Shrink-wrapped software, Wintel, plug and play
- Web (or Internet) explosion in 1990s
- What's next (around 2000) – Post PC
  - Personal mobile devices and cloud
- What's next? (today's question – history repeats!)



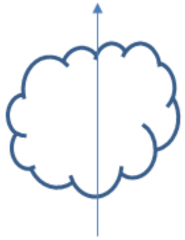
The Post-PC Era

Source: Computer Organization and Design, Hennessy and Patterson)



The Post-PC Era

- ❑ Hardware perspective
  - Personal mobile device (PMD)
    - Smartphone or tablet (maybe glasses or wearables)
    - Battery operated, wireless connectivity to Internet
    - Download software (“apps”)
    - Touch-sensitive screen or even speech input
  - Could computing (computer as utility)
    - Warehouse-scale computers (WSCs)
      - Giant data center with 100,000 servers



cloud 의미
저편 어딘가에 있다.

data center 의미
1. 회사가 대용량의 데이터를 보유하여 소비자에게 정보 제공
2. 개인이나 기업의 사용료를 받고 hw/sw 제공

- ❑ Software as a Service (SaaS)
  - Software developer has a portion of application on PMD and a portion in the cloud (e.g., web search, SNS)

Cloud Computing

2번에 해당 규모의 경제학으로 실현 가능

- H대 전산실의 경우
  - Hardware: 중대형 서버들
  - Software: OS, database, applications
  - 자체 응용 소프트웨어 유지 보수 (HY-IN)
  - 자체 인력, 예산
- IBM say: 우리한테 외주 주시오, 연 xx억만 내시오
  - HW only, HW + SW, or everything
  - 그러면 우리는 인터넷 저편 어디의 ("somewhere over the cloud") IBM 서비스를 web 을 통해 사용하게 됨
    - Will my data be safe? 보안,프라이버시 문제
- 비슷한 예: web hard, free email, …

Cloud Computing

Computing resources delivered as service over network (Internet)

Image of cloud computing in Wikipedia: http://en.wikipedia.org/wiki/File:Cloud_computing.svg

SaaS: software as a service

PaaS: platform as a service

IaaS: infrastructure as a service

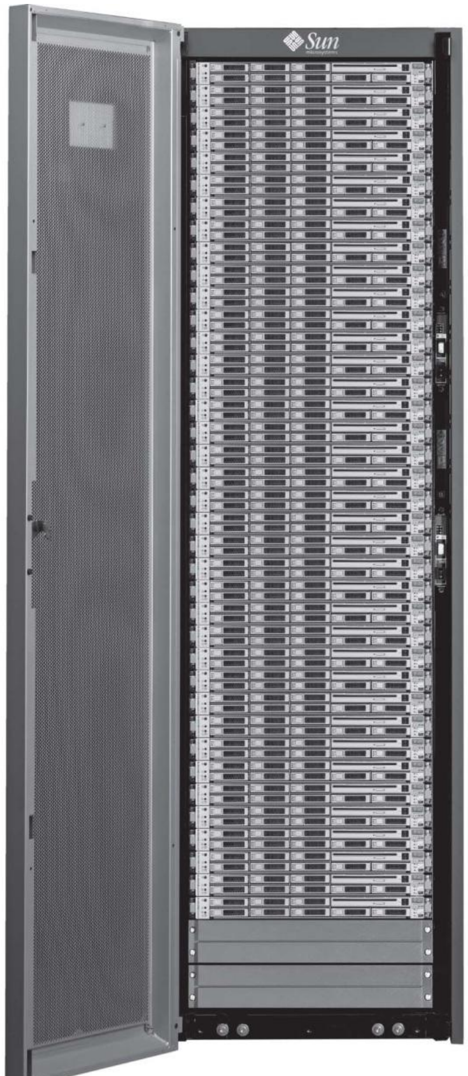
Large Data Centers

- Cloud computing
  - Computers and disks used by millions of users
  - Computer as utility
- Warehouse-scale computer
  - Space, cooling, networking, storage
- Physical design standard
  - Rack mount computer
    - 19" wide (482.6 mm)
    - 1.75" (44.45 mm) high – rack unit or unit (U)
    - Most popular: 42 U high
- Standard container filled with racks and interconnection

19-inch Rack with 42 1U servers

(Source: Computer Organization and Design, Hennessy and Patterson)

- Standard container

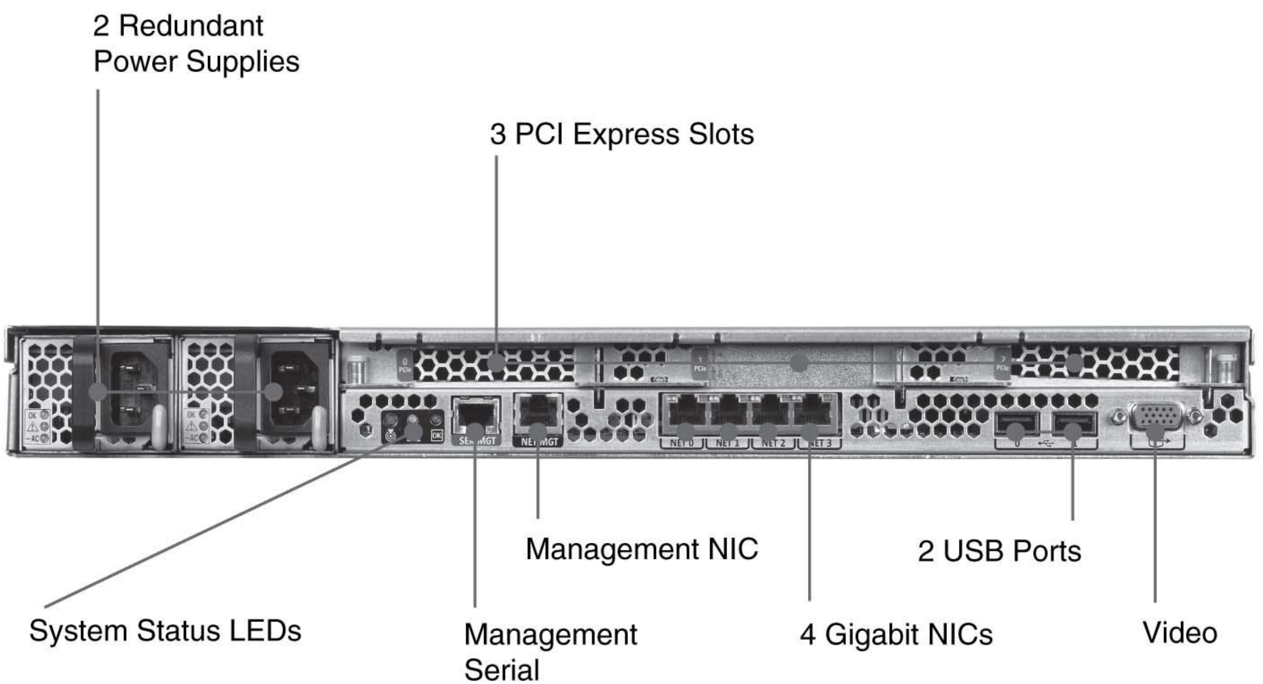


Sun Fire x4150 1U Server

(Source: Computer Organization and Design, Hennessy and Patterson)

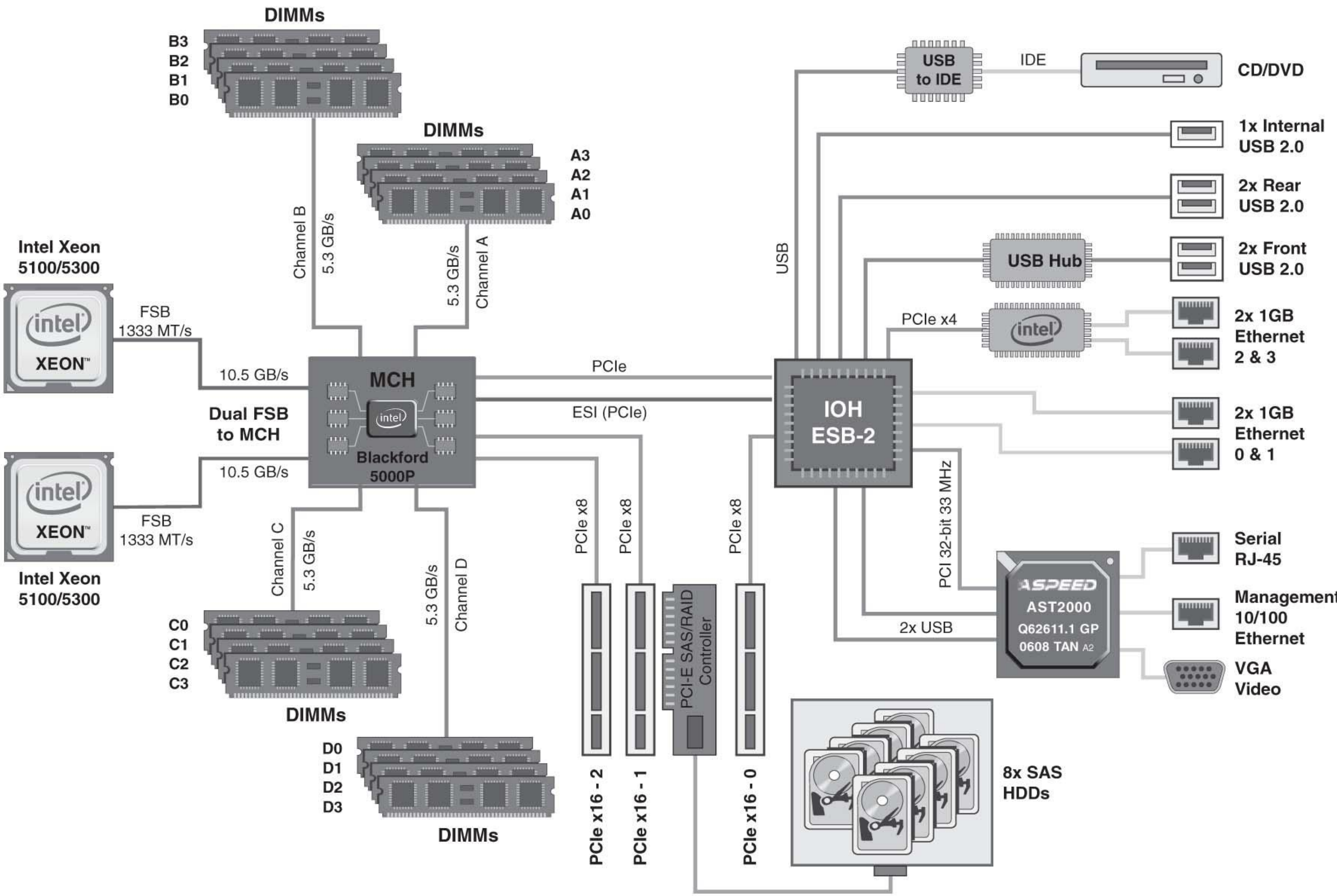
Front

Rear



Inside of Sun Fire x4150

(Source: Computer Organization and Design, Hennessy and Patterson)



Google Bigtable

Supercomputers

- Desire for highest performance possible
  - IBM, NEC, Fujitsu, Europe
  - US startups in late 1980s
    - Cray, Thinking machines, Kendal Square Research
- Financial difficulty
  - Especially due to end of cold war
- Demand for supercomputers strong and growing in commercial applications
  - Commercial aircraft and automobile design
  - Chemical engineering, weather forecasting, and so on
  - Industry must deal with cost/performance

Supercomputer

- Cray in 1980s
  - Fastest hardware with best existing technologies

Image of CRAY-2 supercomputer:

http://en.wikipedia.org/wiki/File:Cray2.jpeg

Image of CRAY T3E processor board:

http://en.wikipedia.org/wiki/File:Processor_board_cray-2_hg.jpg

Supercomputer

- 2012 fastest supercomputer: IBM Sequoia Blue Gene/Q
  - 1,572,864 processor cores, 1.6 petabytes of memory
  - 91 refrigerator-sized server racks
  - 1.6 petaFLOPS
- Why not possible with Cray style of design?
- IBM Watson

Image of IBM Blue Gene P supercomputer:
http://en.wikipedia.org/wiki/File:IBM_Blue_Gene_P_supercomputer.jpg
Image of LLNL Blue Gene L Diagram:
http://en.wikipedia.org/wiki/File:LLNL_BGL_Diagram.png

Summary

- Machine called computer
  - Basic computer organization
    - Components and interconnection
  - Operational principle
    - Fetch-decode-execute cycle
  - Service provided by CPU (or processor or computer)
    - Instruction Set Architecture
  - † Completion of modern digital computer in 1945
- Programming and problem solving
- IT Gold Rush in USA
- Classes of computers