Computer Architecture and Logic Design

Top Level View of Computer Structure and Function

Engr. Rimsha

Last Lecture Review

- Difference between ISA and microarchitecture?
  - ISA :Agreed upon interface between software and hardware
  - Microarchitecture : Specific implementation of an ISA
- Hardware vs. Software Knowledge requirements?
  - Can develop better software if you understand the underlying hardware
  - Can design better hardware if you understand what software it will execute
  - Can design a better computing system if you understand both
- Why study Computer Architecture?
  - Enable Better Systems, Enable new applications, Enable better solutions to problems

+

Today’s Lecture Topics

- Computer Structure and Function

Structure and Function

- Hierarchical system
  - Set of interrelated subsystems
- Hierarchical nature of complex systems is essential to both their design and their description
- Designer need only deal with a particular level of the system at a time
  - Concerned with structure and function at each level

- Structure
  - The way in which components relate to each other

- Function
  - The operation of individual components as part of the structure

Function

A computer can perform four basic functions:

- Data processing
- Data storage
- Data movement
- Control

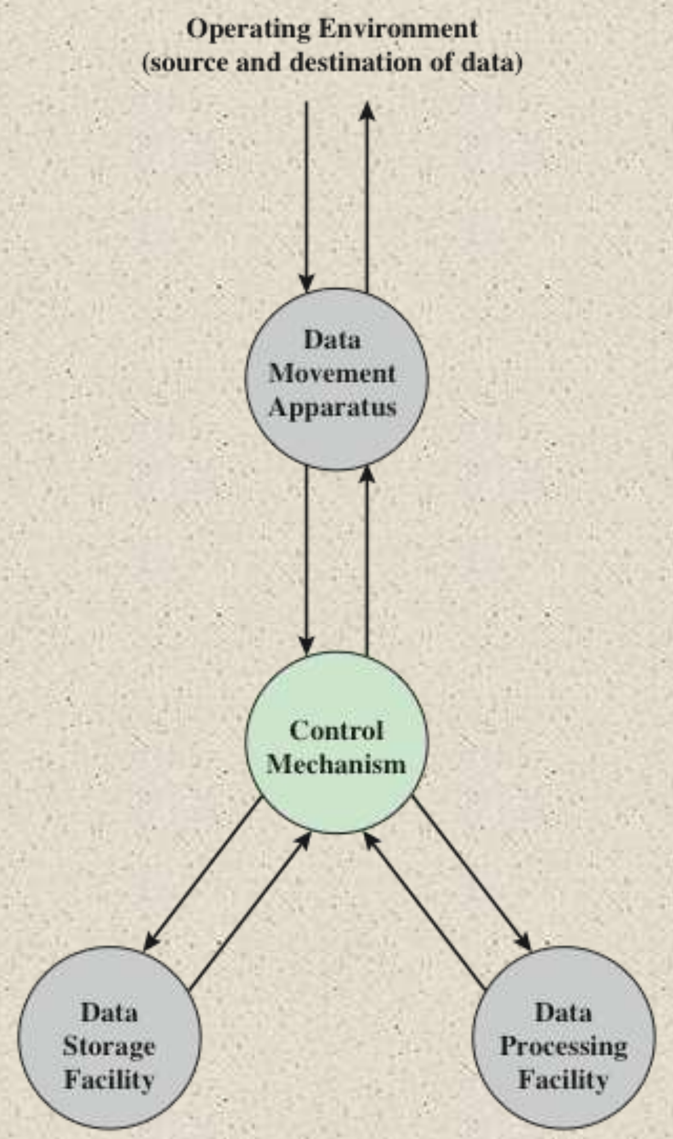


Figure 1.1 A Functional View of the Computer

Operations

(a)
Data movement

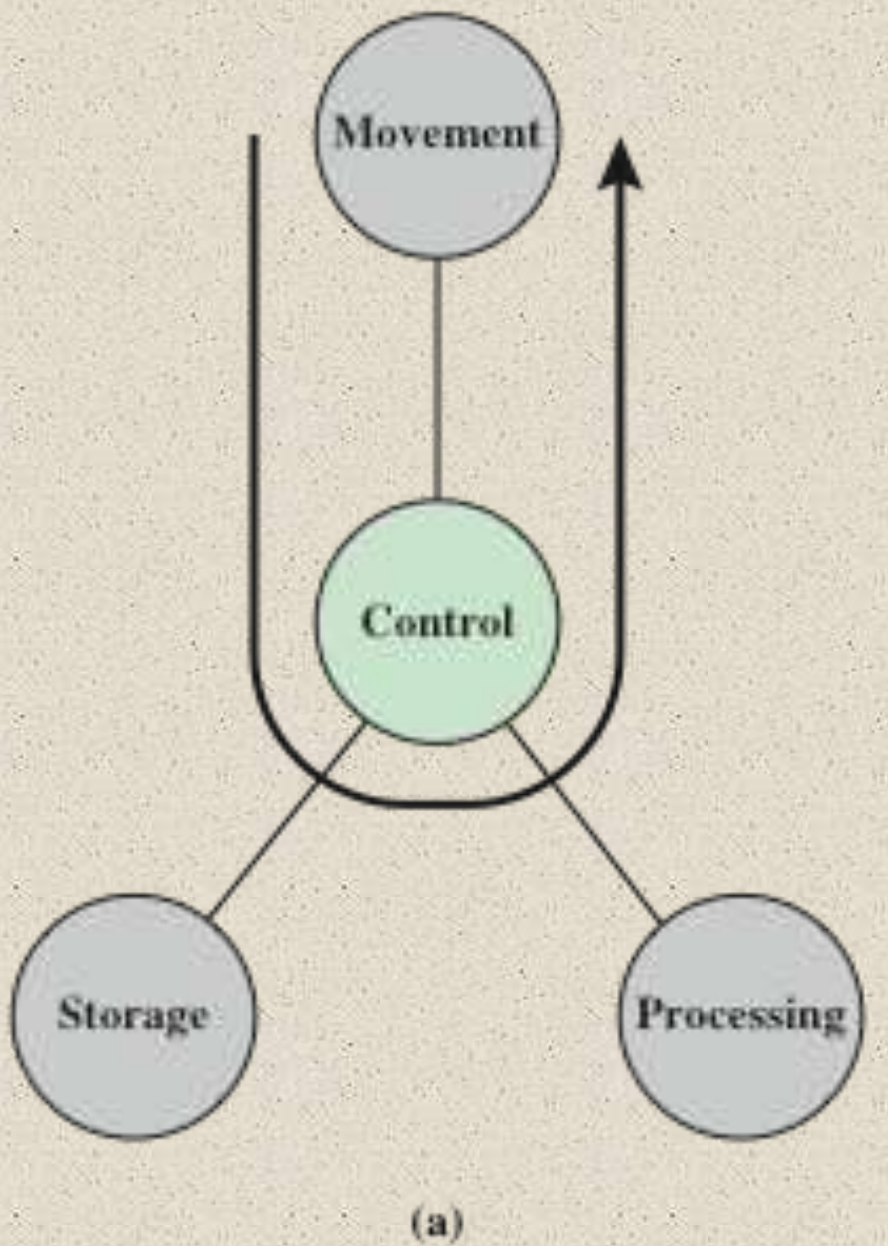
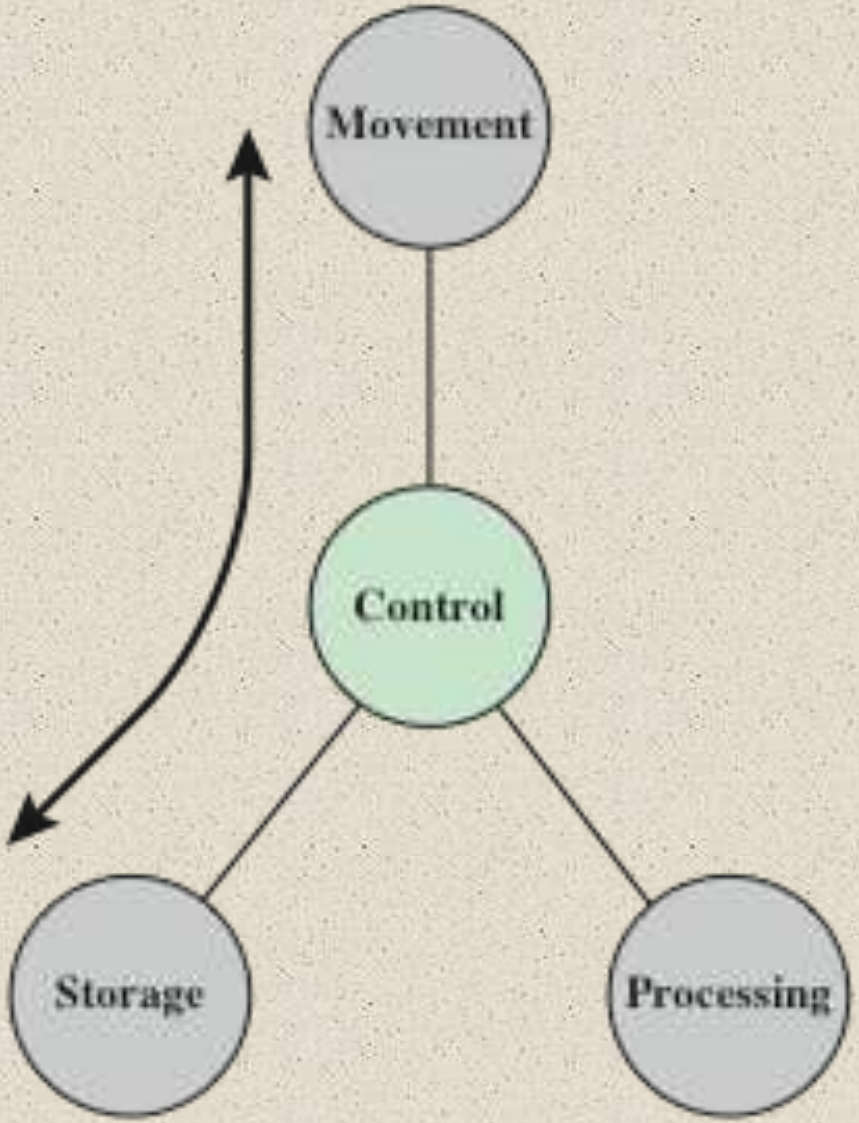


Figure 1.2 Possible Computer Operations

Operations

(b)
Data storage



(b)

Figure 1.2 Possible Computer Operations

Operations

(c)
Data movement

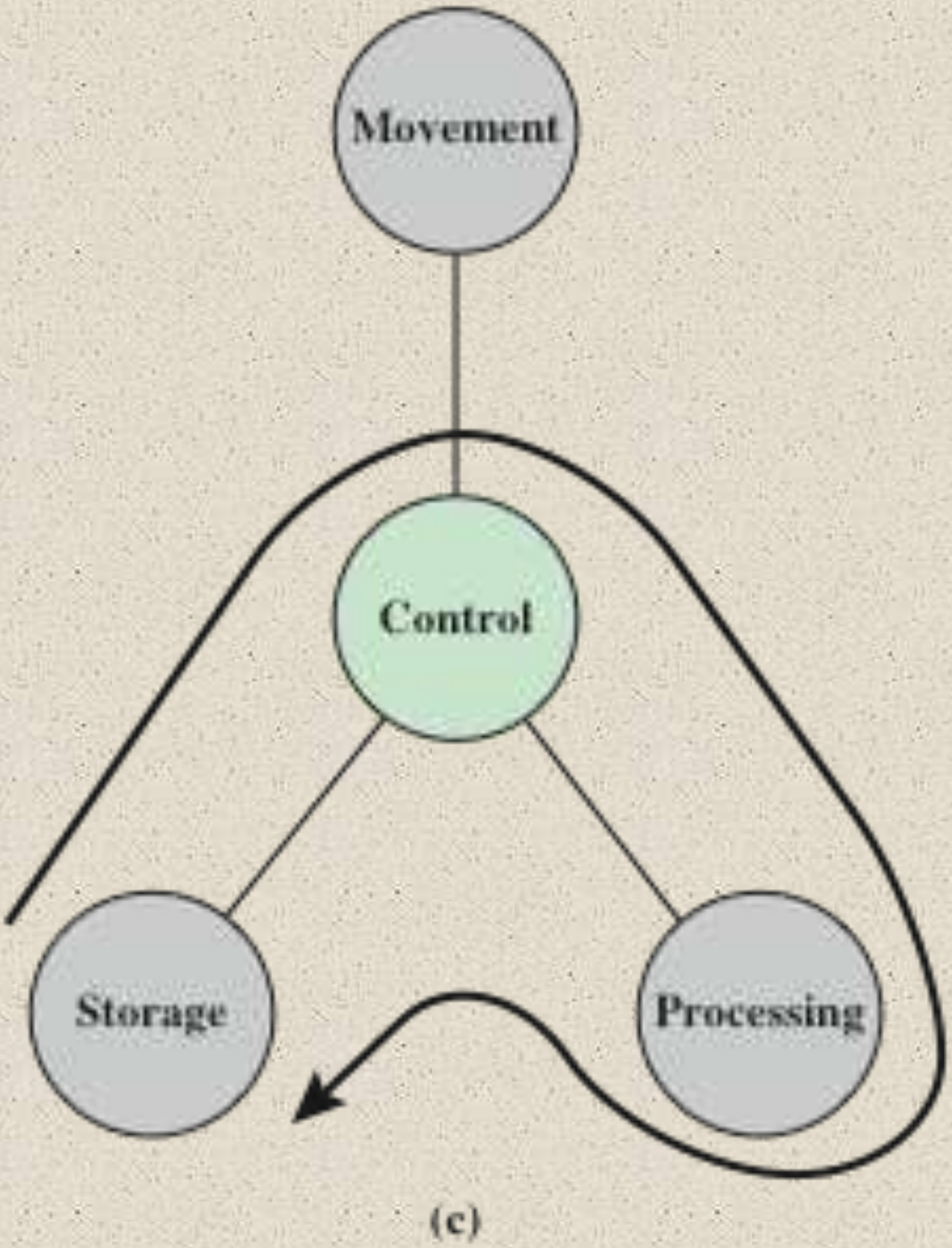


Figure 1.2 Possible Computer Operations

Operations

(d) Control

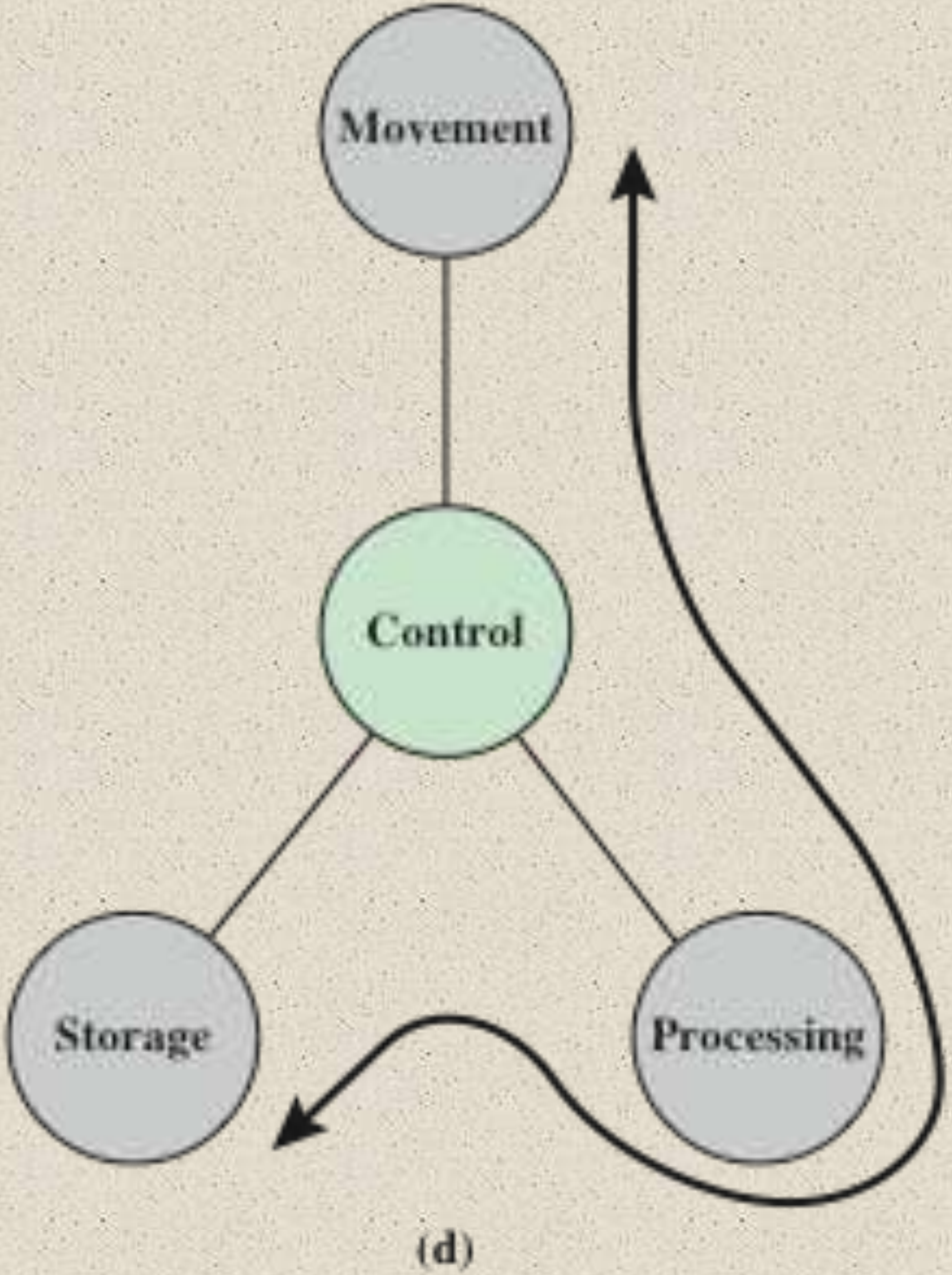
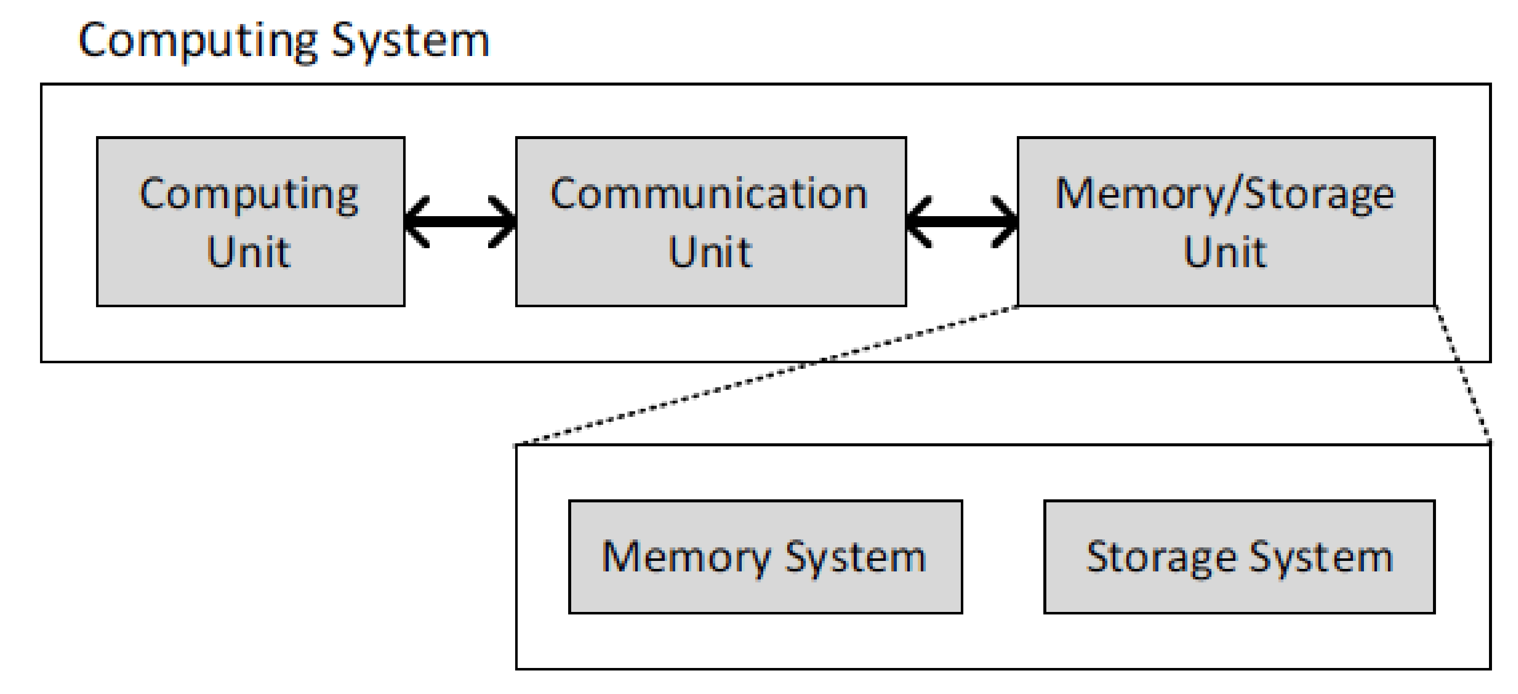


Figure 1.2 Possible Computer Operations

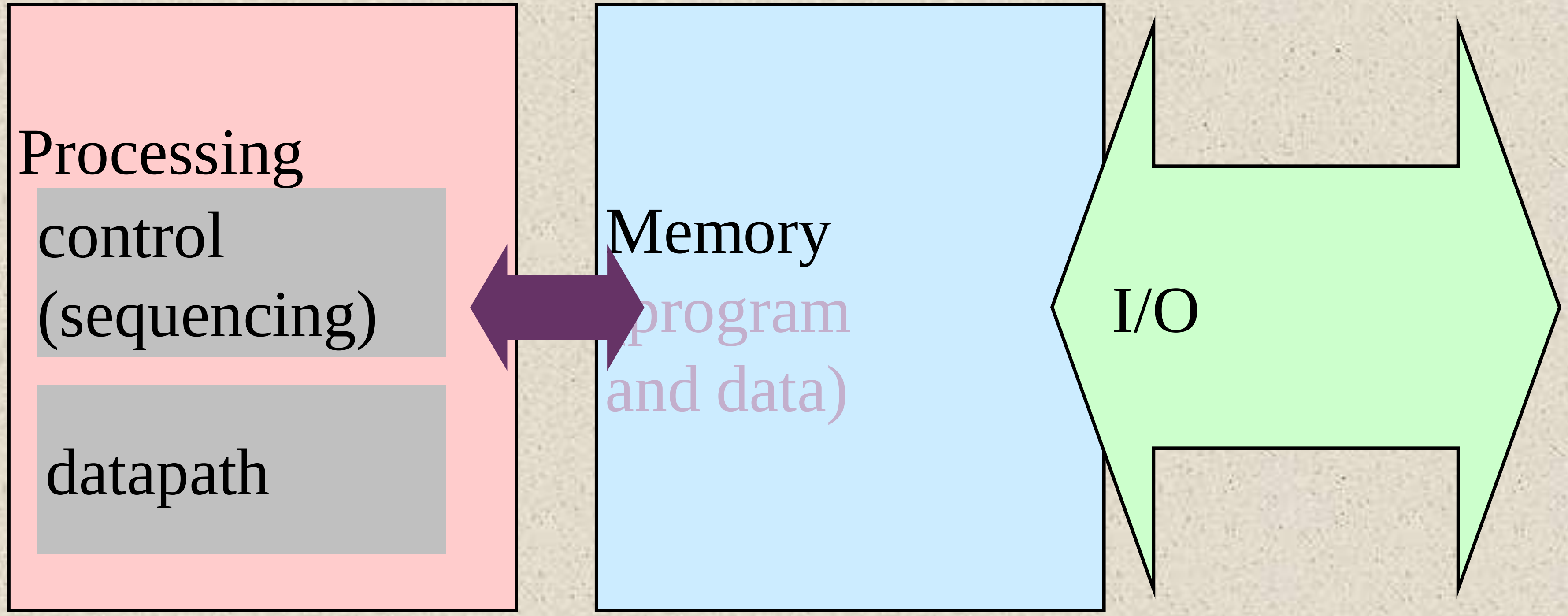
+
What is A Computer?

- Three key components
- Computation
- Communication
- Storage (memory)



+ What is A Computer?

- We will cover all three components



The Computer

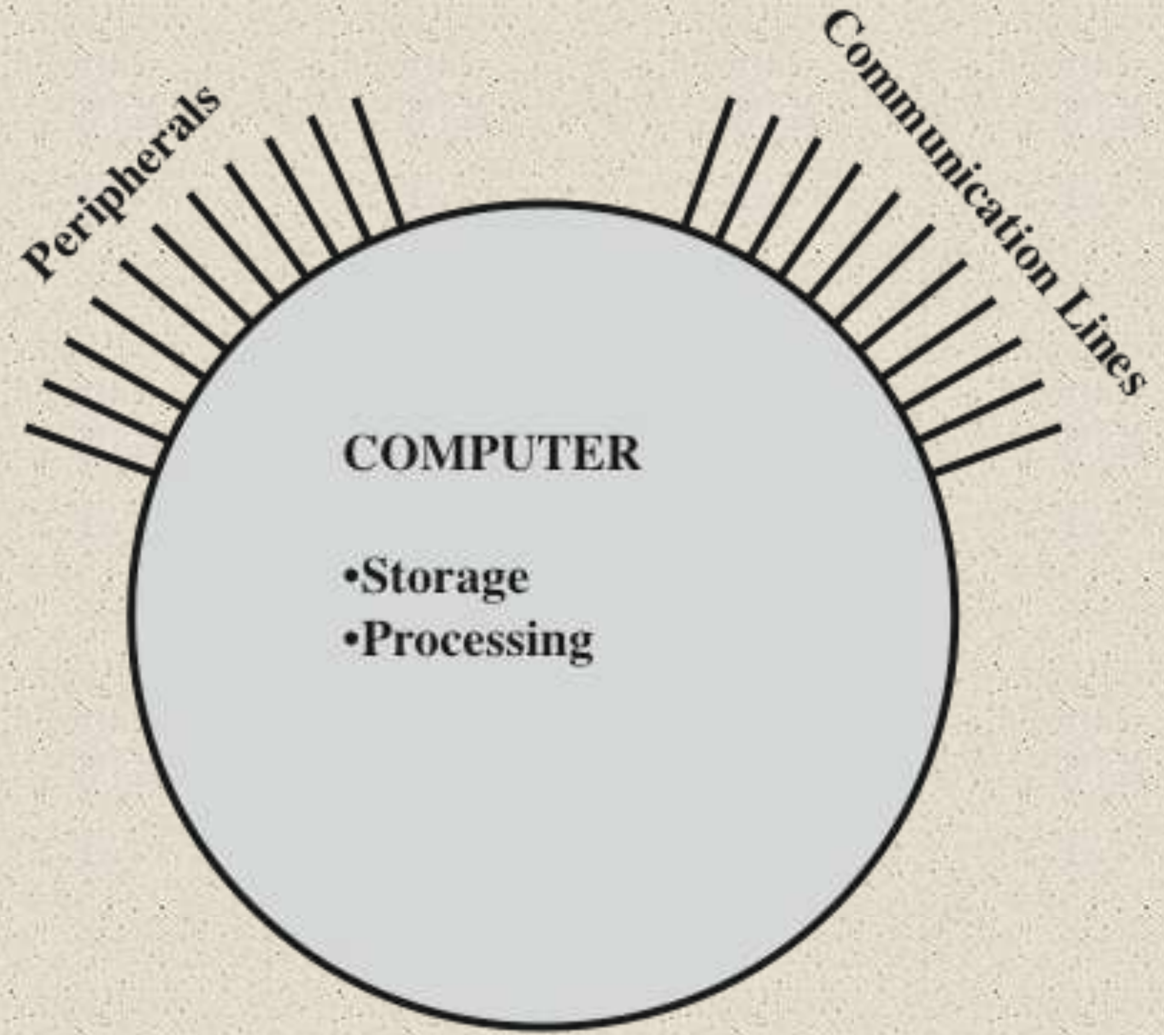
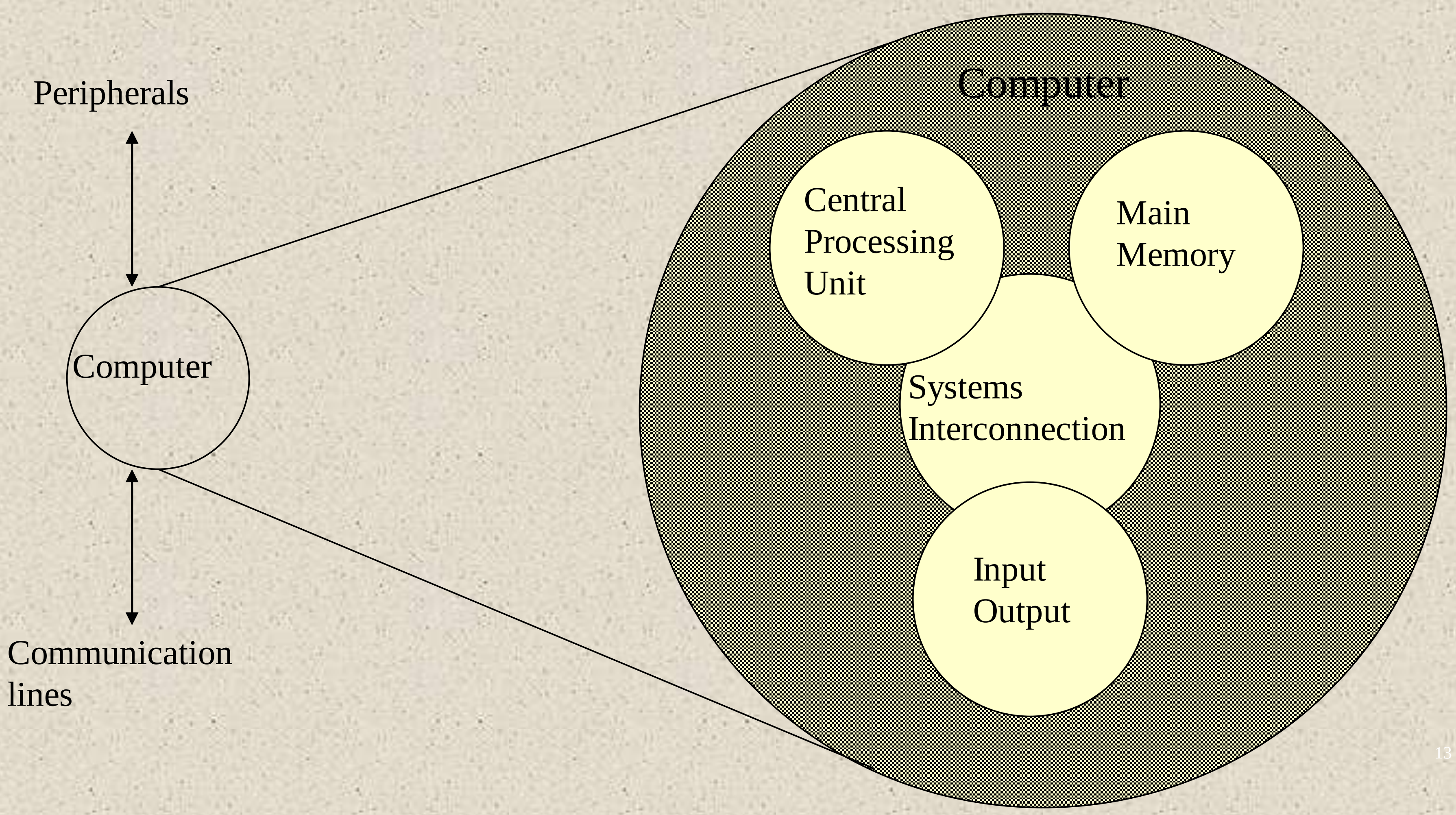
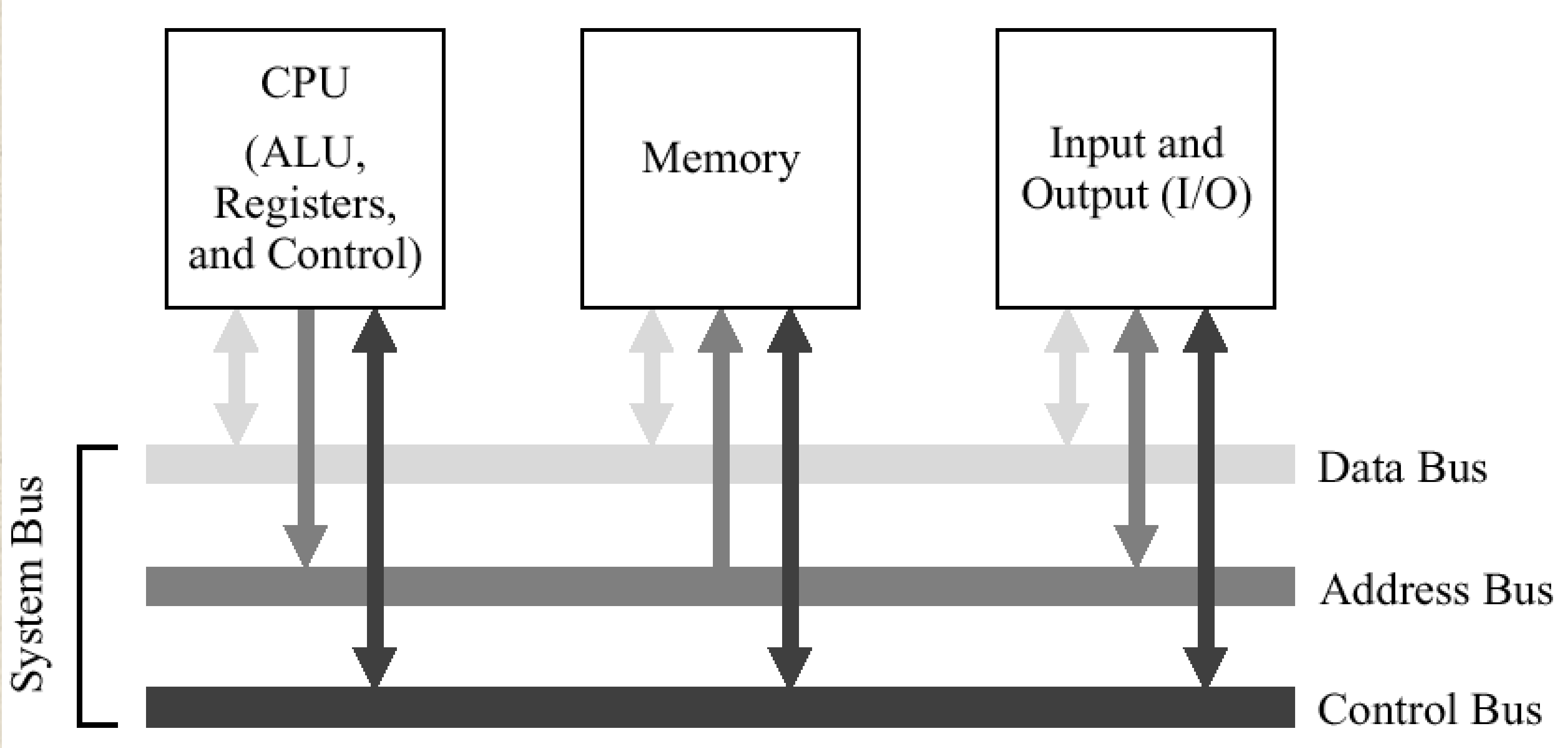


Figure 1.3 The Computer

Structure - Top Level



Generic System Bus

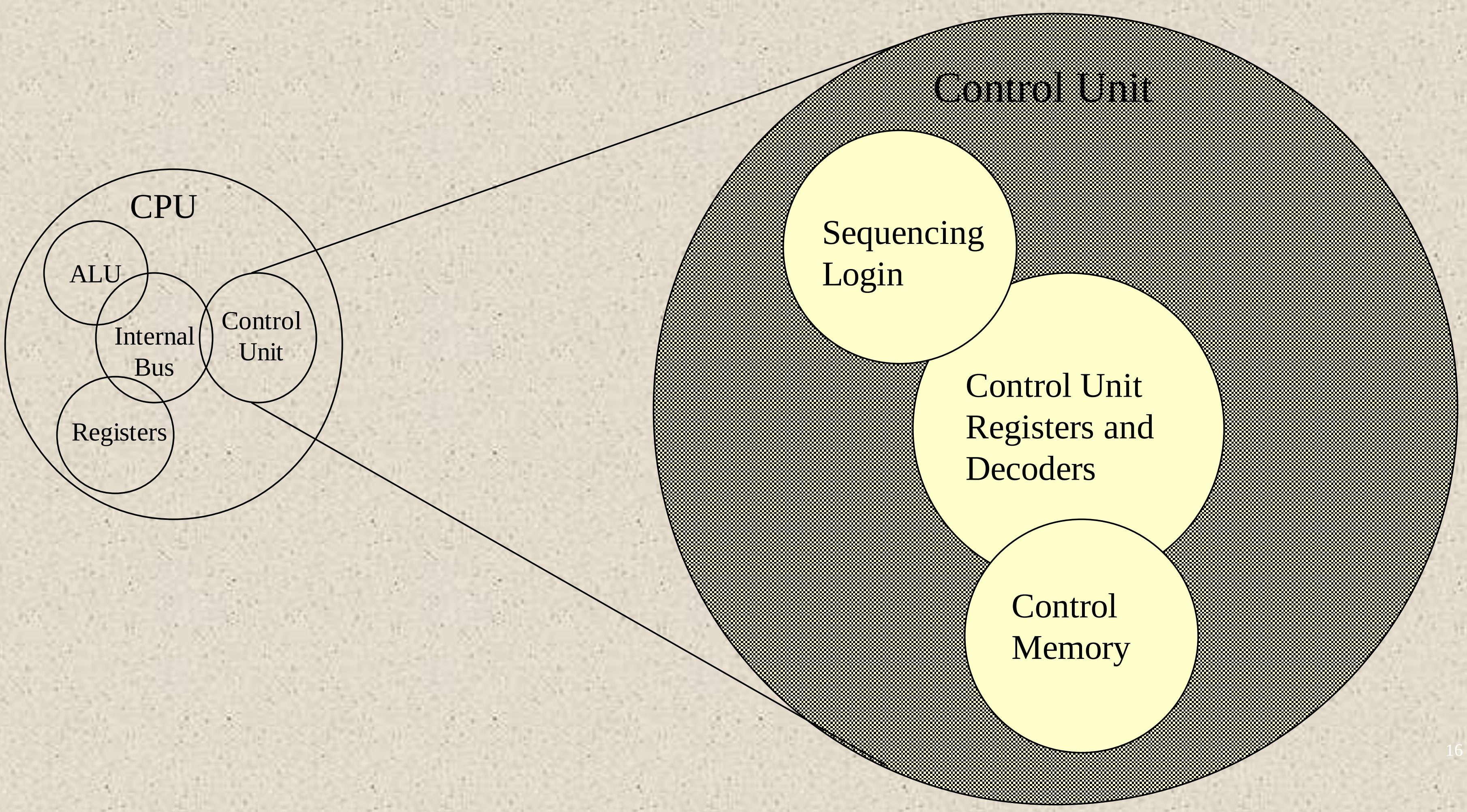


System Bus = Data, Address, and Control Bus (set of wires, e.g. 32 wires each)
Typically multiple I/O buses, power bus, etc.

Structure - CPU

- Major components of the CPU
  - Control Unit (CU) – Controls the operation of the CPU
  - Arithmetic and Logic Unit (ALU) – Performs data processing functions, e.g. arithmetic operations
  - Registers – Fast storage internal to the CPU, but contents can be copied to/from main memory
  - CPU Interconnect – Some mechanism that provides for communication among the control unit, ALU, and registers

Structure – A Microprogrammed Control Unit



Structure

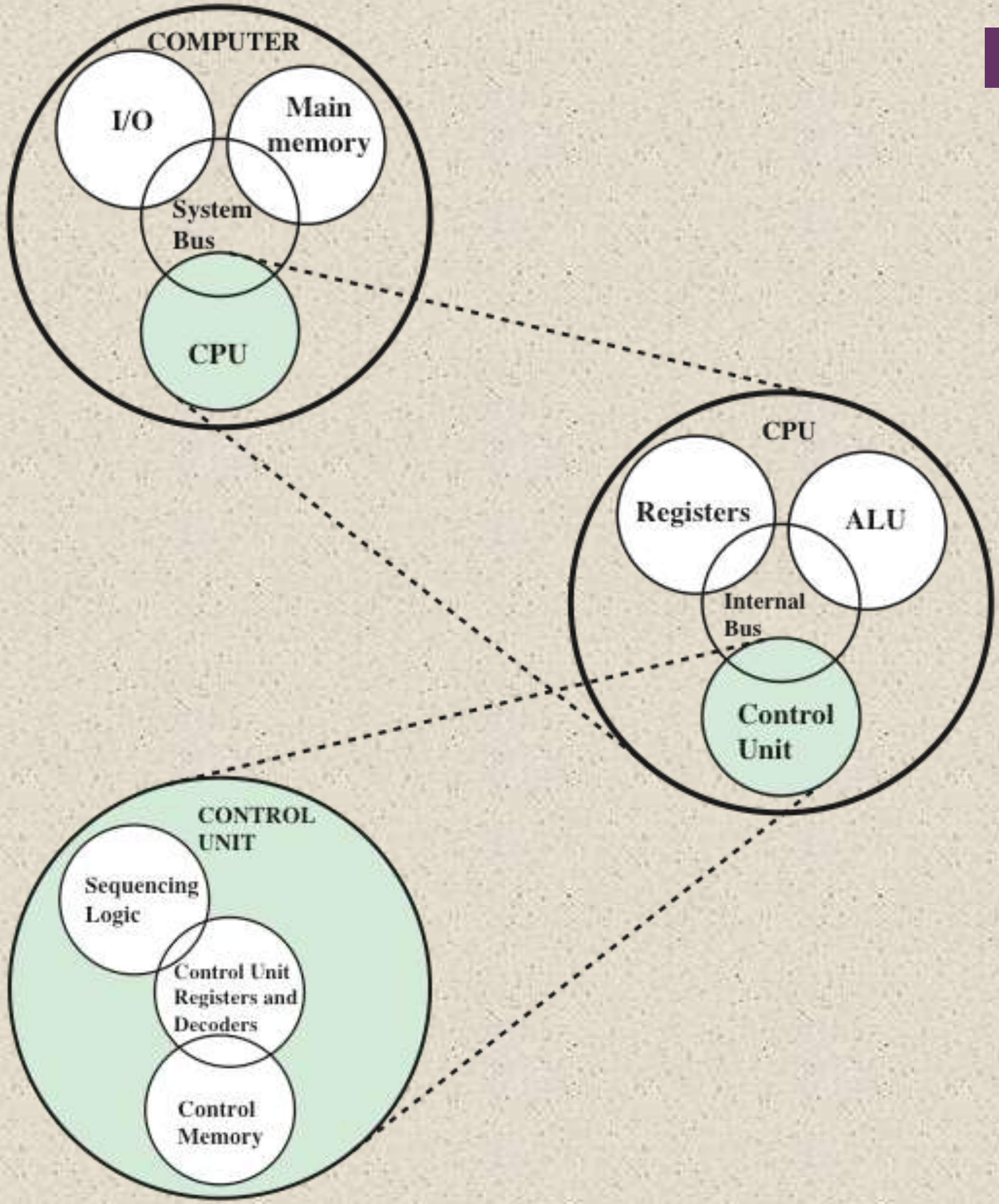


Figure 1.4 A Top-Down View of a Computer

+

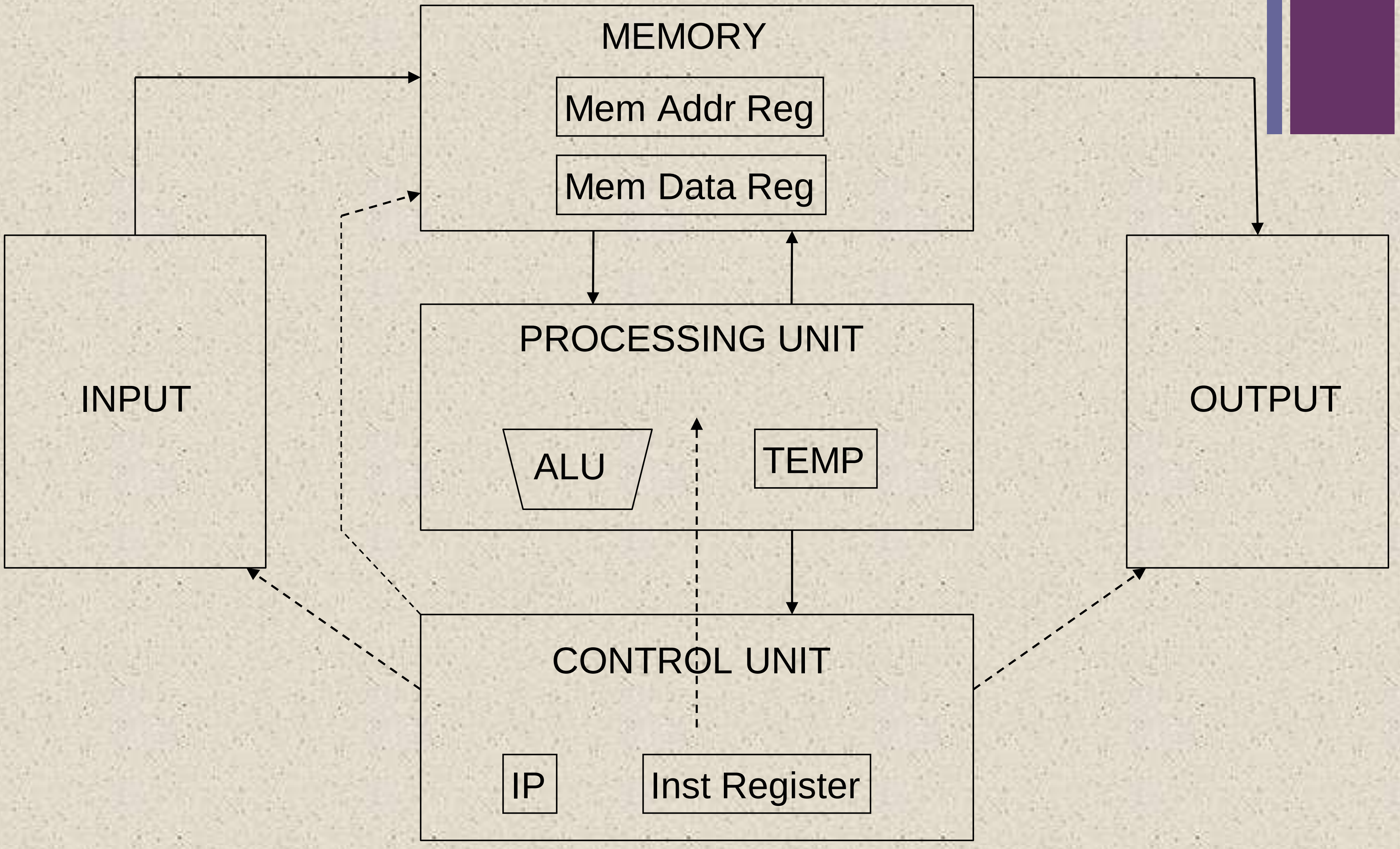
The Von Neumann Model/Architecture

- Also called stored program computer (instructions in memory). Two key properties:

- Stored program
  - Instructions stored in a linear memory array
  - Memory is unified between instructions and data
    - The interpretation of a stored value depends on the control signals

- Sequential instruction processing
  - One instruction processed (fetched, executed, and completed) at a time
  - Program counter (instruction pointer) identifies the current instr.
  - Program counter is advanced sequentially except for control transfer instructions

The Von Neumann Model (of a Computer)



+

The Von Neumann Model (of a Computer)

- Q: Is this the only way that a computer can operate?

- A: No.

- Qualified Answer: But, it has been the dominant way
  - i.e., the dominant paradigm for computing
  - for N decades

The Dataflow Model (of a Computer)

- Von Neumann model: An instruction is fetched and executed in control flow order
  - As specified by the instruction pointer
  - Sequential unless explicit control flow instruction

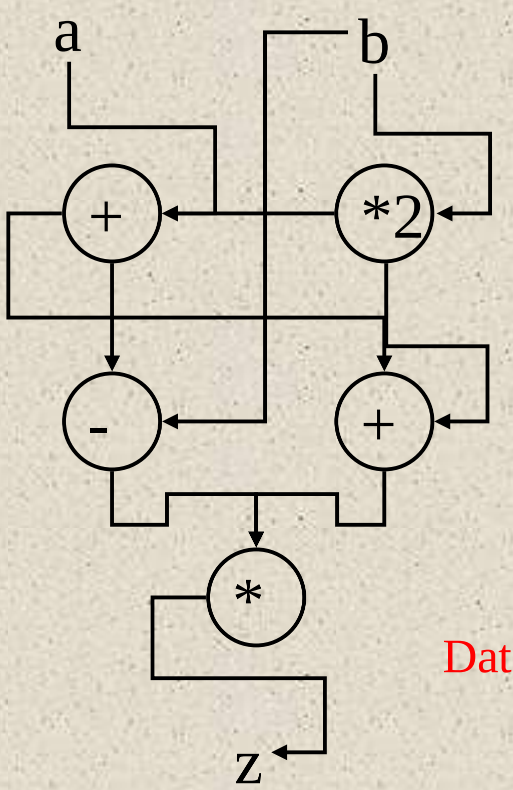
- Dataflow model: An instruction is fetched and executed in data flow order
  - i.e., when its operands are ready
  - i.e., there is no instruction pointer
  - Instruction ordering specified by data flow dependence
    - Each instruction specifies “who” should receive the result
    - An instruction can “fire” whenever all operands are received
  - Potentially many instructions can execute at the same time
    - Inherently more parallel

Von Neumann vs Dataflow

- Consider a Von Neumann program
  - What is the significance of the program order?
  - What is the significance of the storage locations?

```
v <= a + b;
w <= b * 2;
x <= v - w
y <= v + w
z <= x * y
```

Sequential

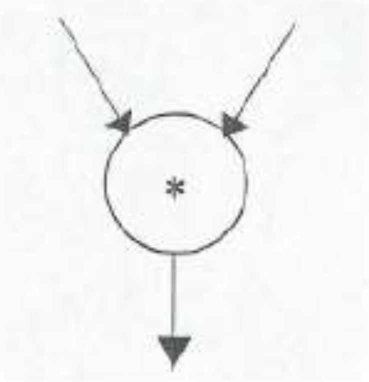


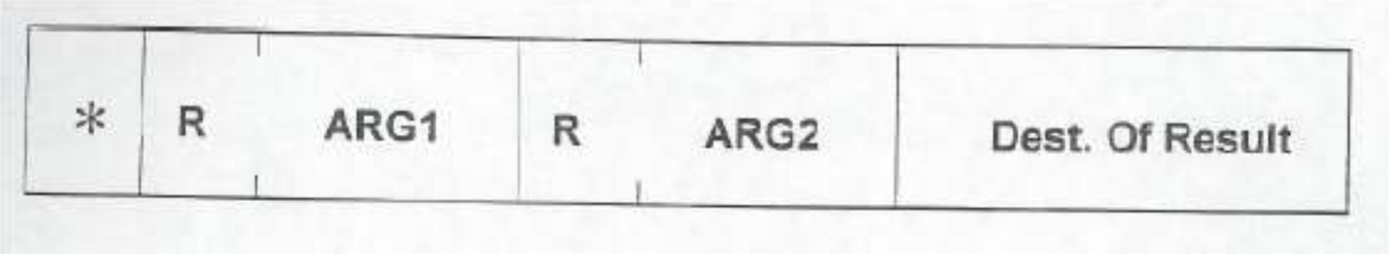
Dataflow

- Which model is more natural to you as a programmer?

More on Data Flow

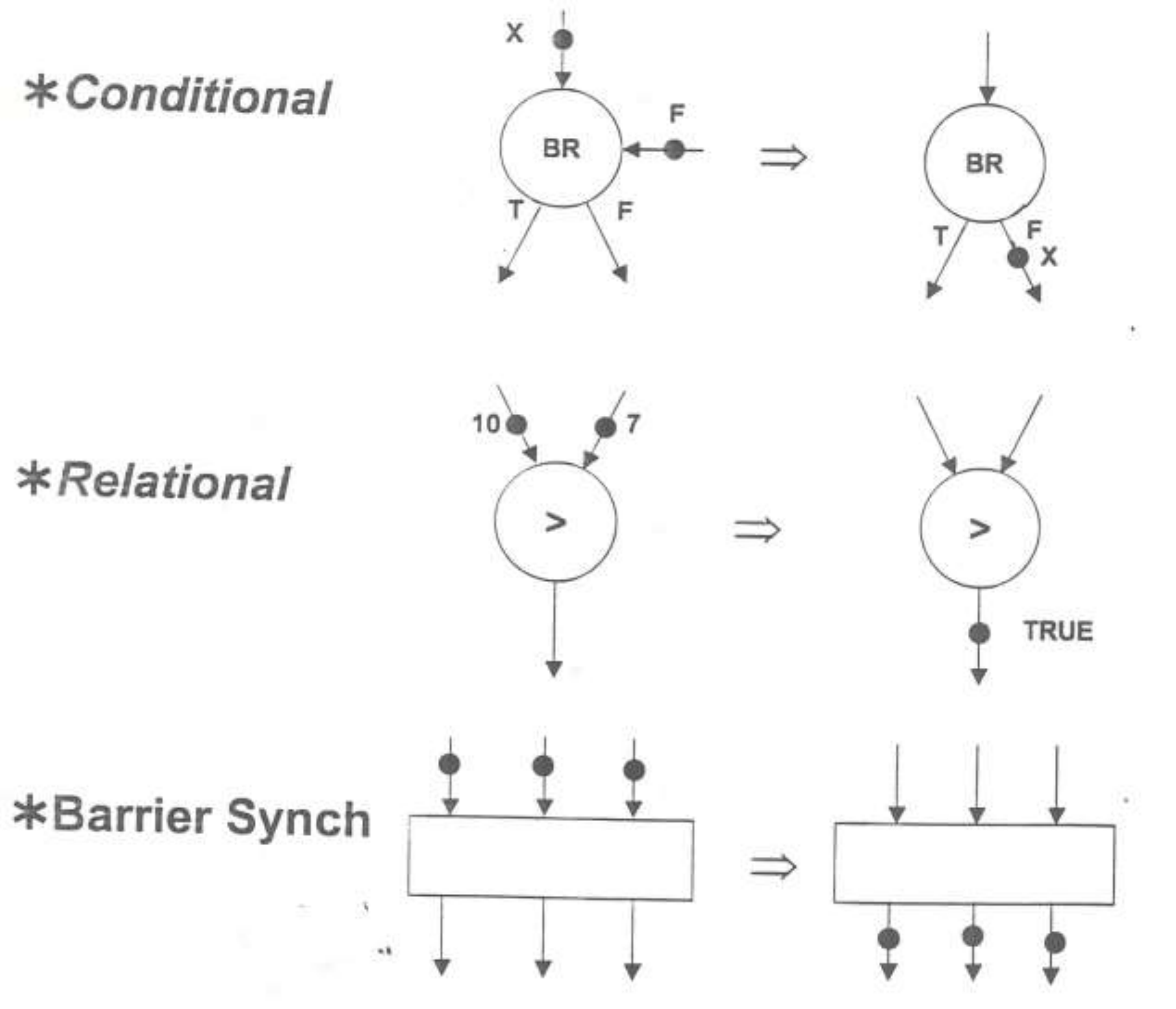
- In a data flow machine, a program consists of data flow nodes
  - A data flow node fires (fetched and executed) when all it inputs are ready
    - i.e. when all inputs have tokens
- Data flow node and its ISA representation



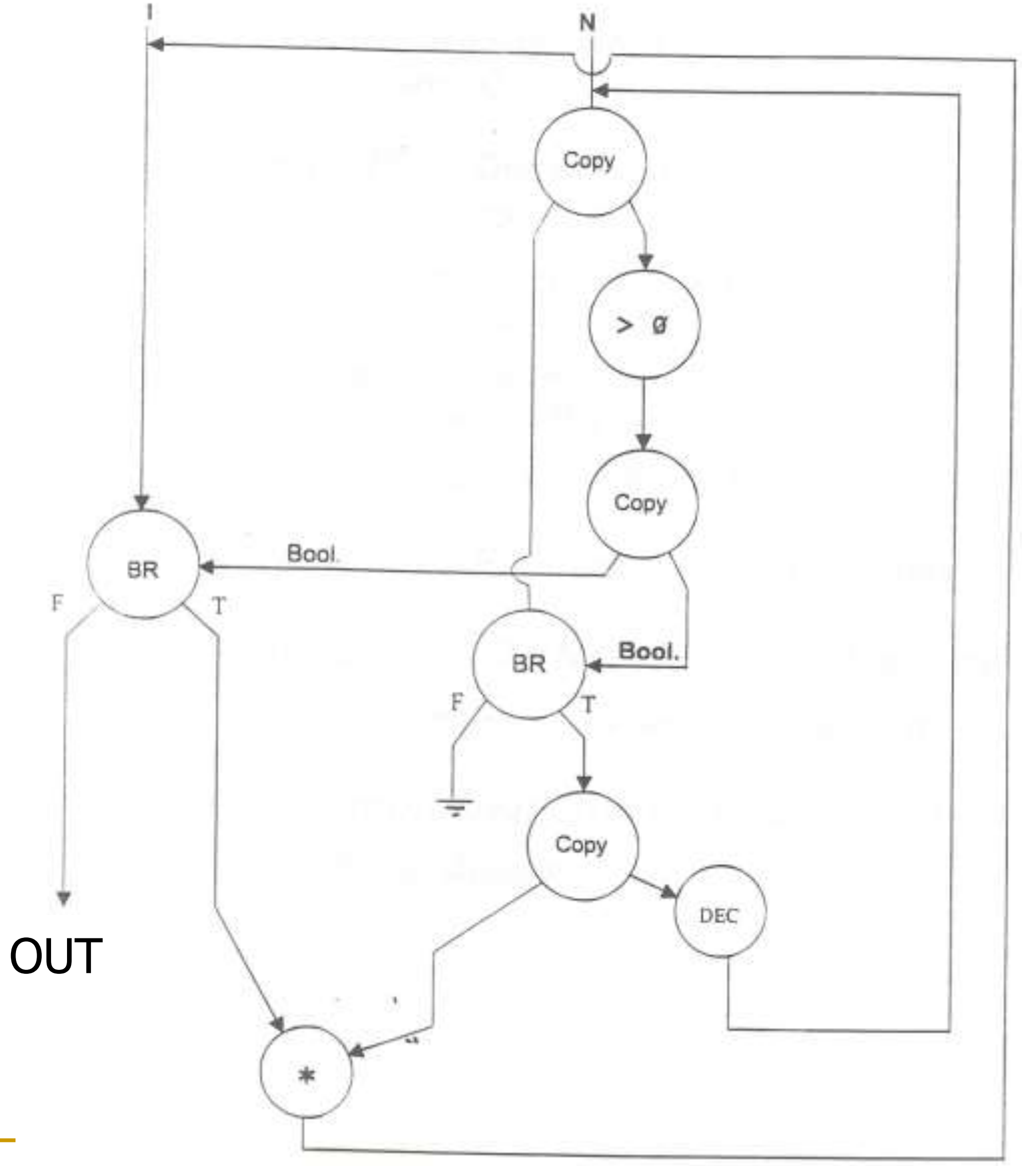


*	R	ARG1	R	ARG2	Dest. Of Result

Data Flow Nodes



An Example Data Flow Program



OUT

+

Memory Hierarchy

- Design constraints on a computer’s memory can be summed up by three questions:
  - How much, how fast, how expensive

- There is a trade-off among capacity, access time, and cost
  - Faster access time, greater cost per bit
  - Greater capacity, smaller cost per bit
  - Greater capacity, slower access time

- The way out of the memory dilemma is not to rely on a single memory component or technology, but to employ a memory hierarchy

Memory Hierarchy - Diagram

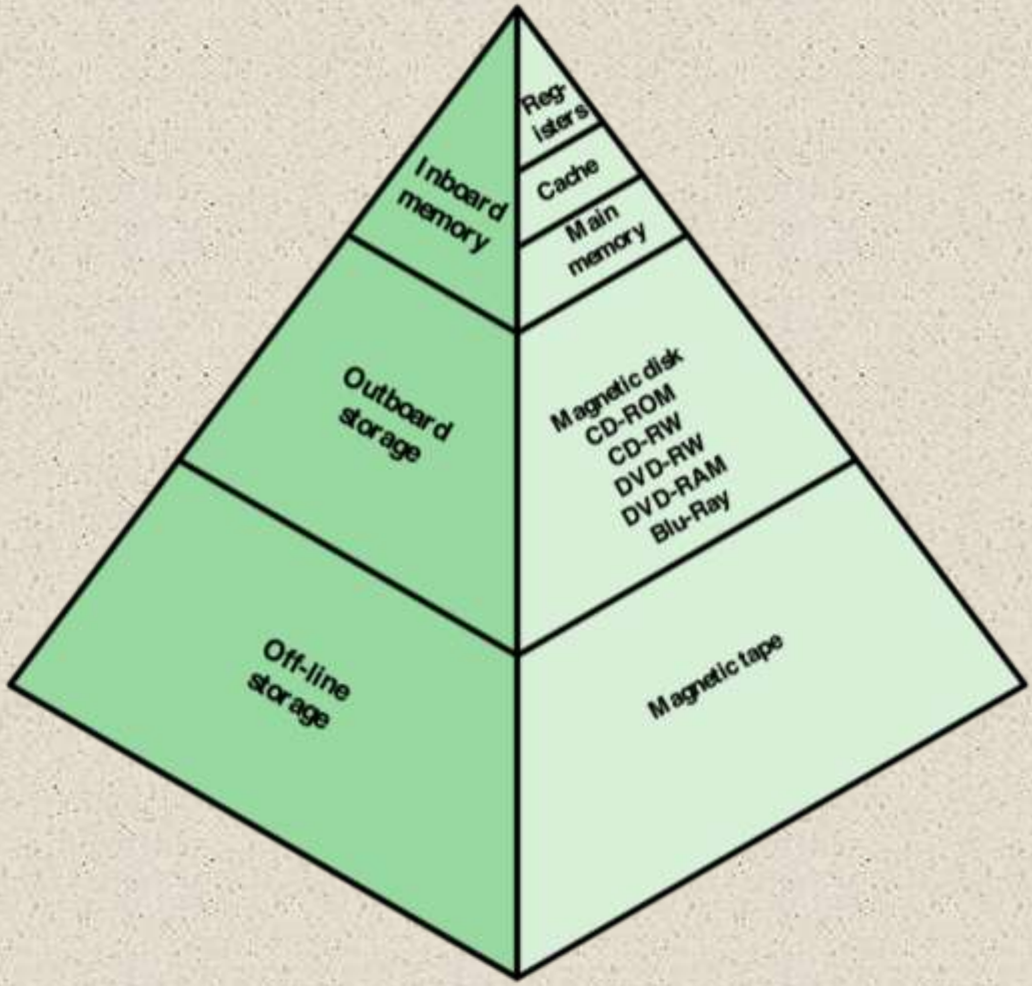
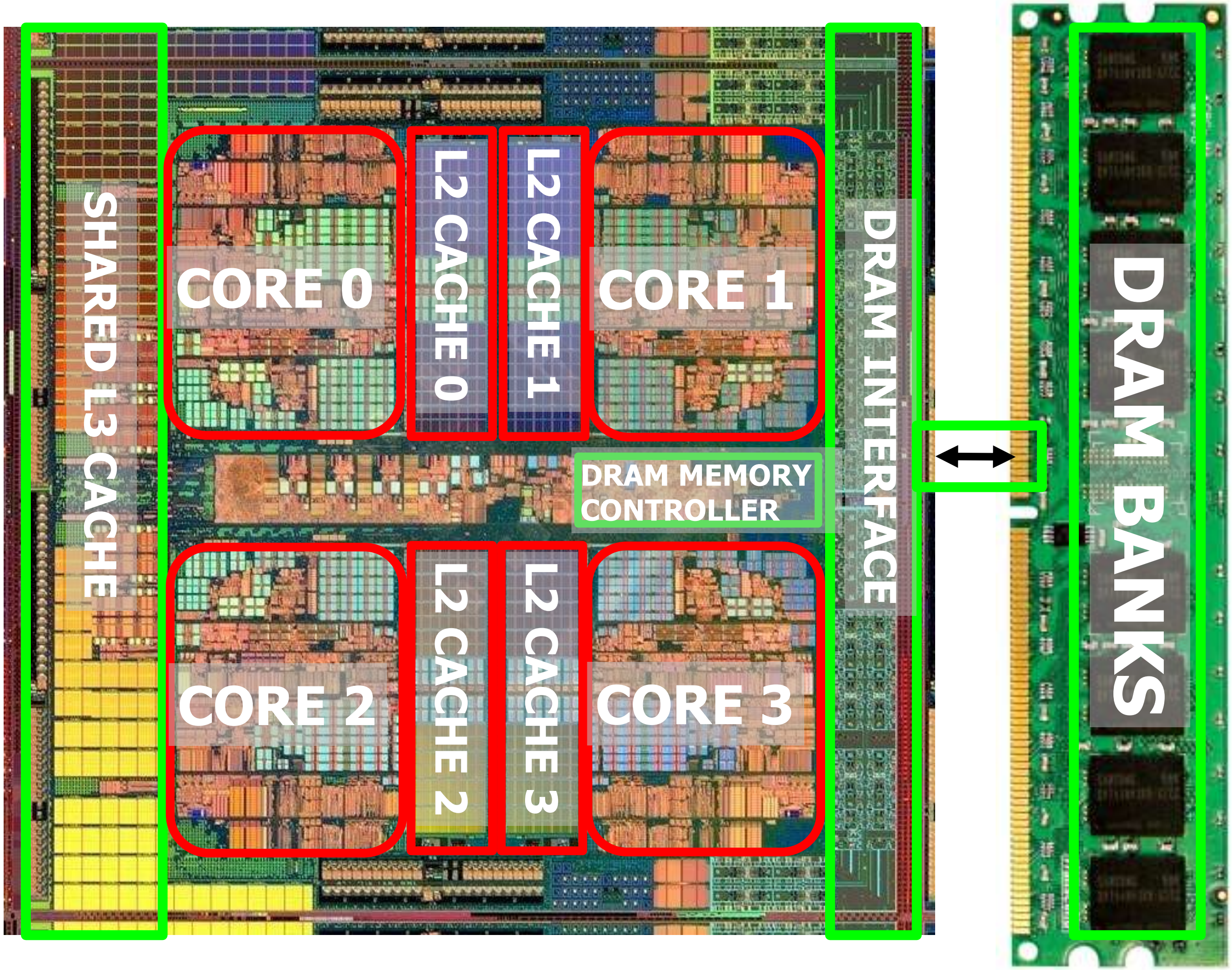
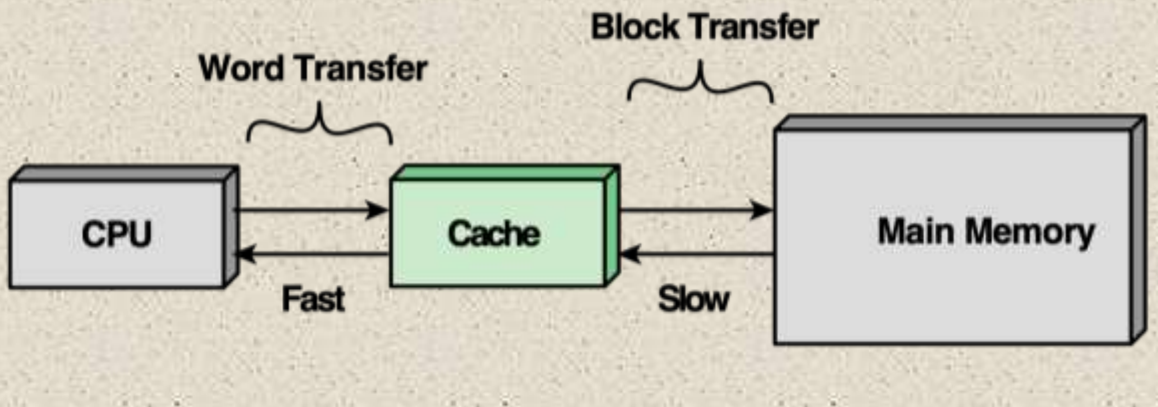


Figure 4.1 The Memory Hierarchy

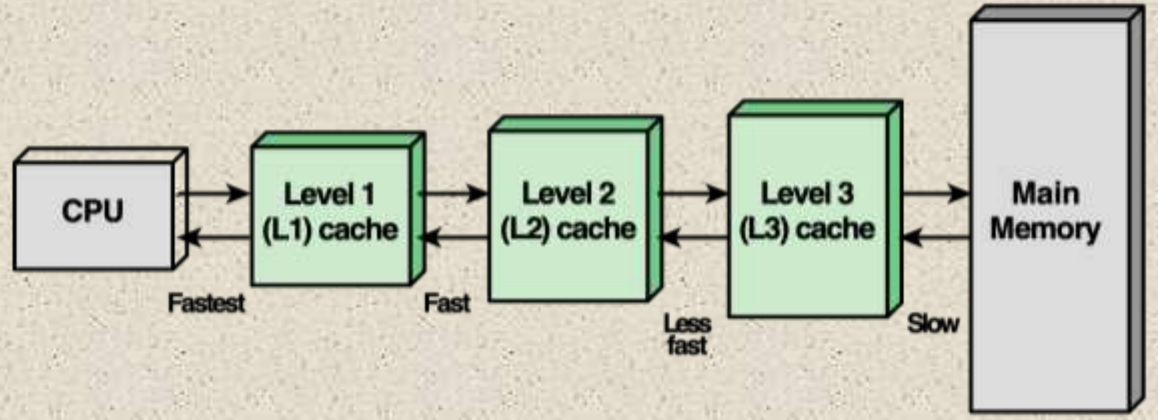
Memory in a Modern System



Cache and Main Memory



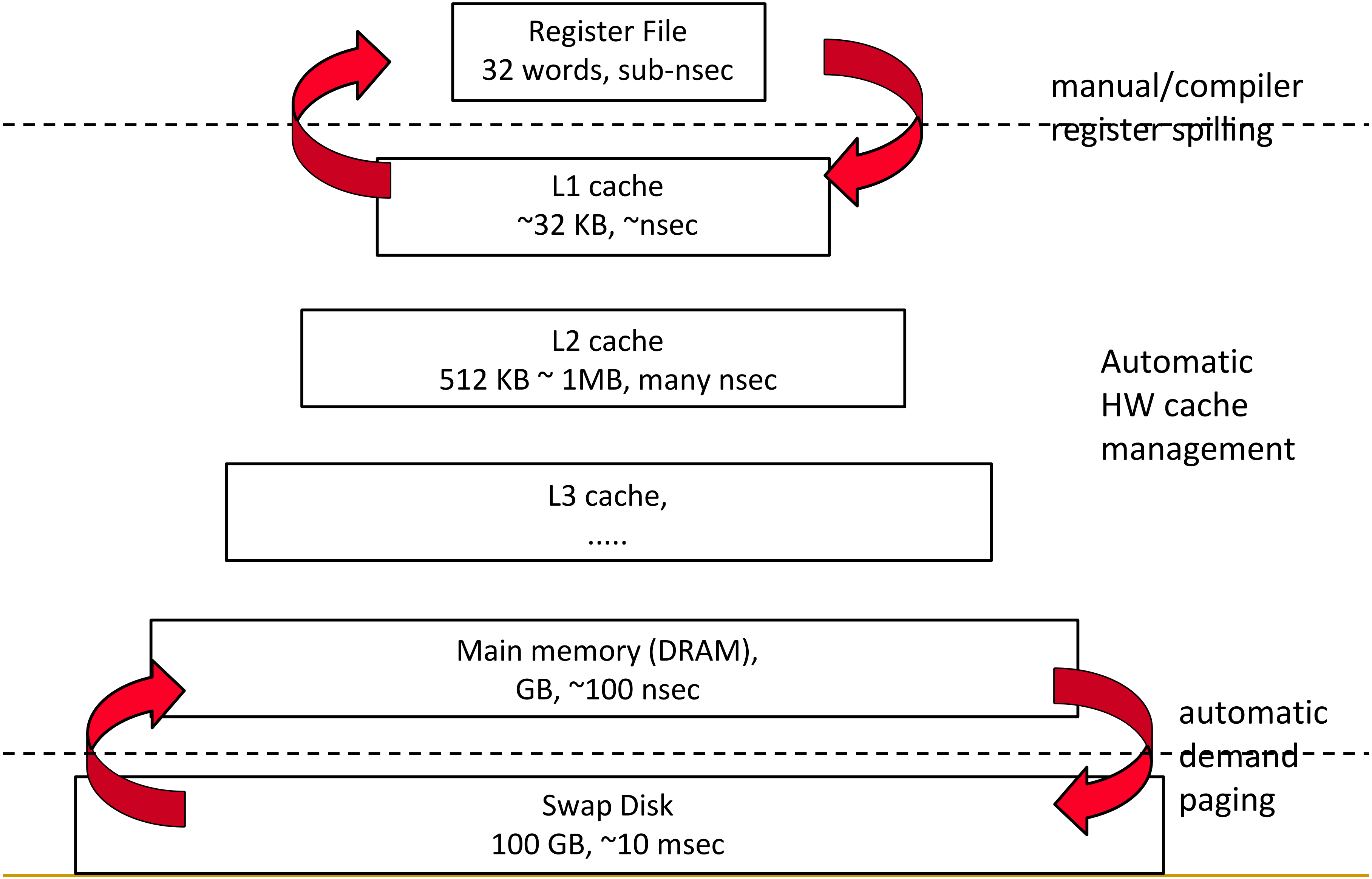
(a) Single cache



(b) Three-level cache organization

Figure 4.3 Cache and Main Memory

A Modern Memory Hierarchy



Register File
32 words, sub-nsec

manual/compiler
register spilling

L1 cache
~32 KB, ~nsec

L2 cache
512 KB ~ 1MB, many nsec

Automatic
HW cache
management

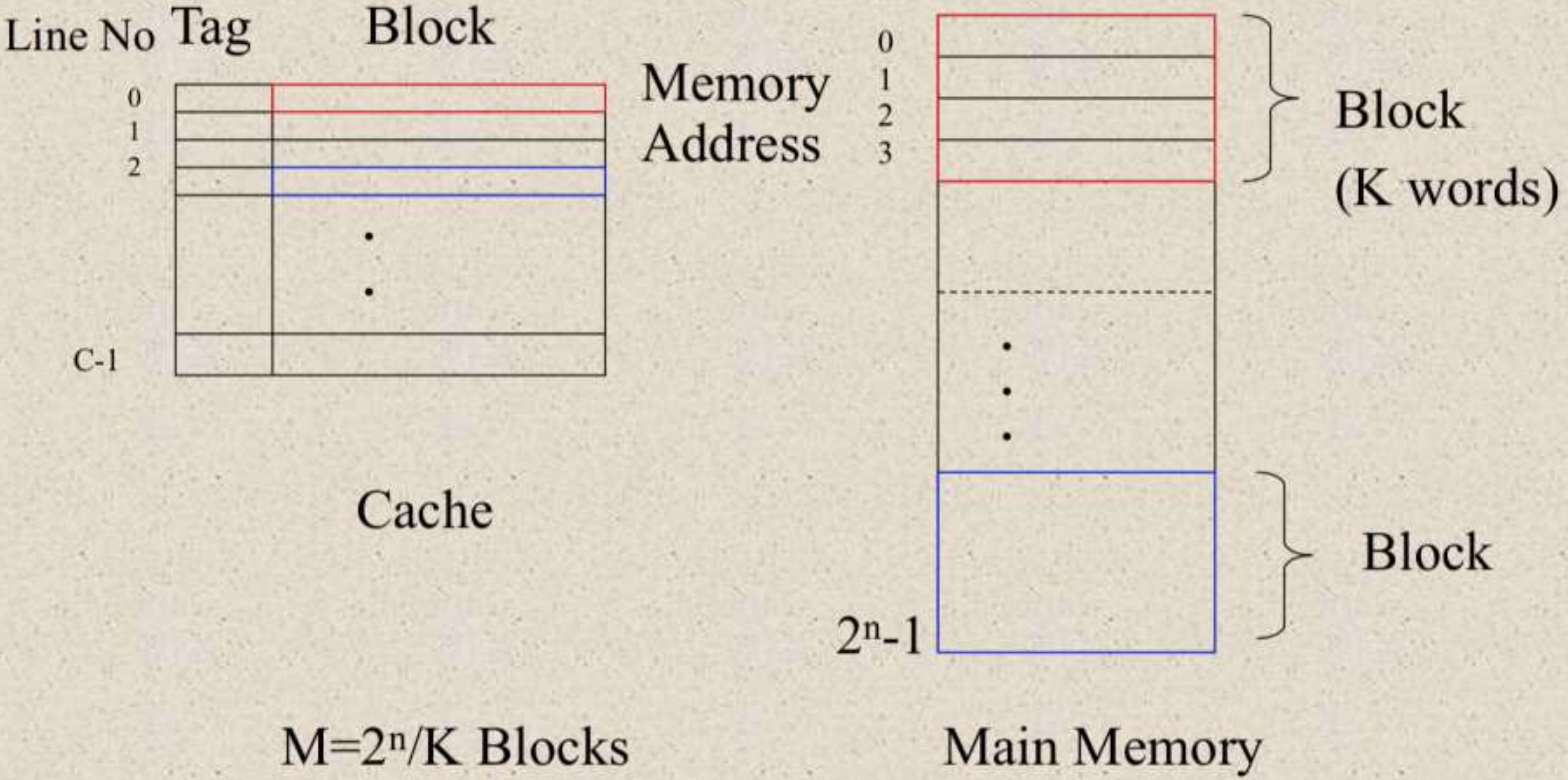
L3 cache,
.....

Main memory (DRAM),
GB, ~100 nsec

automatic
demand
paging

Swap Disk
100 GB, ~10 msec

Cache/Main Memory Structure



Typical Cache Organization

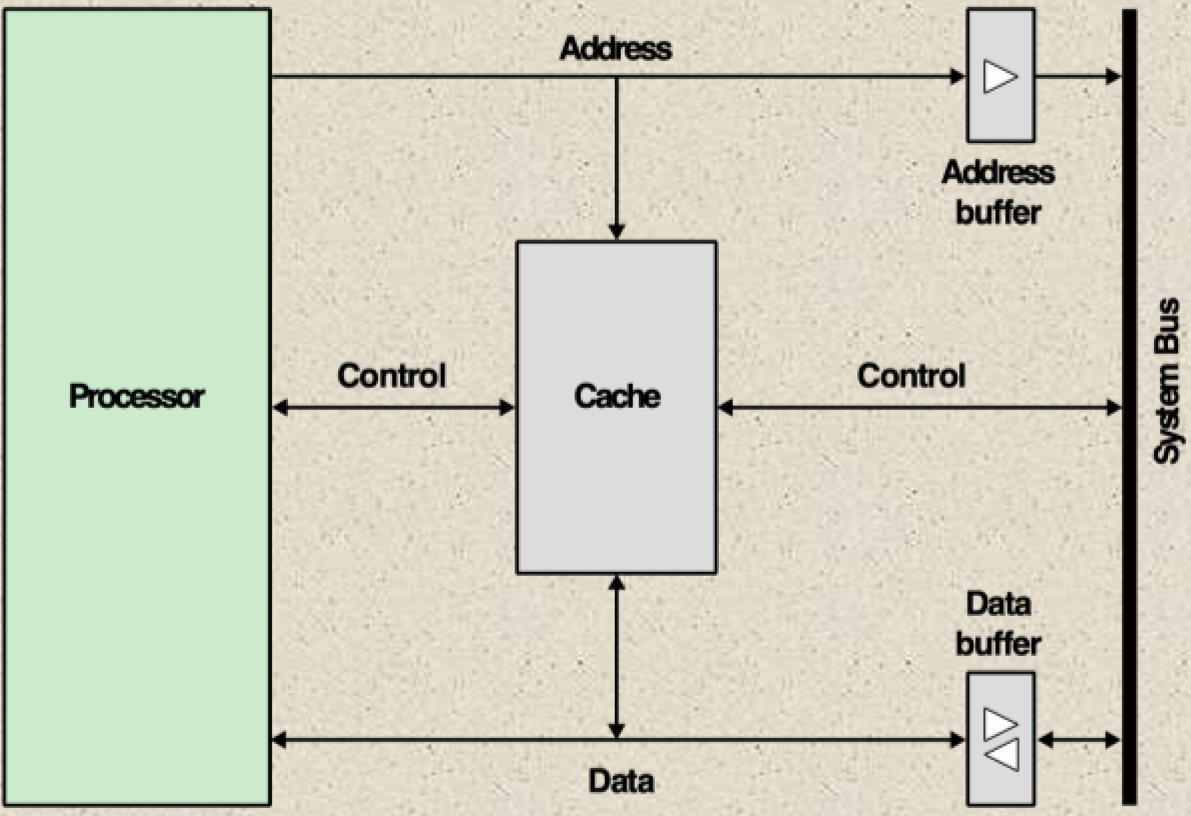


Figure 4.6 Typical Cache Organization

Cache Read Operation

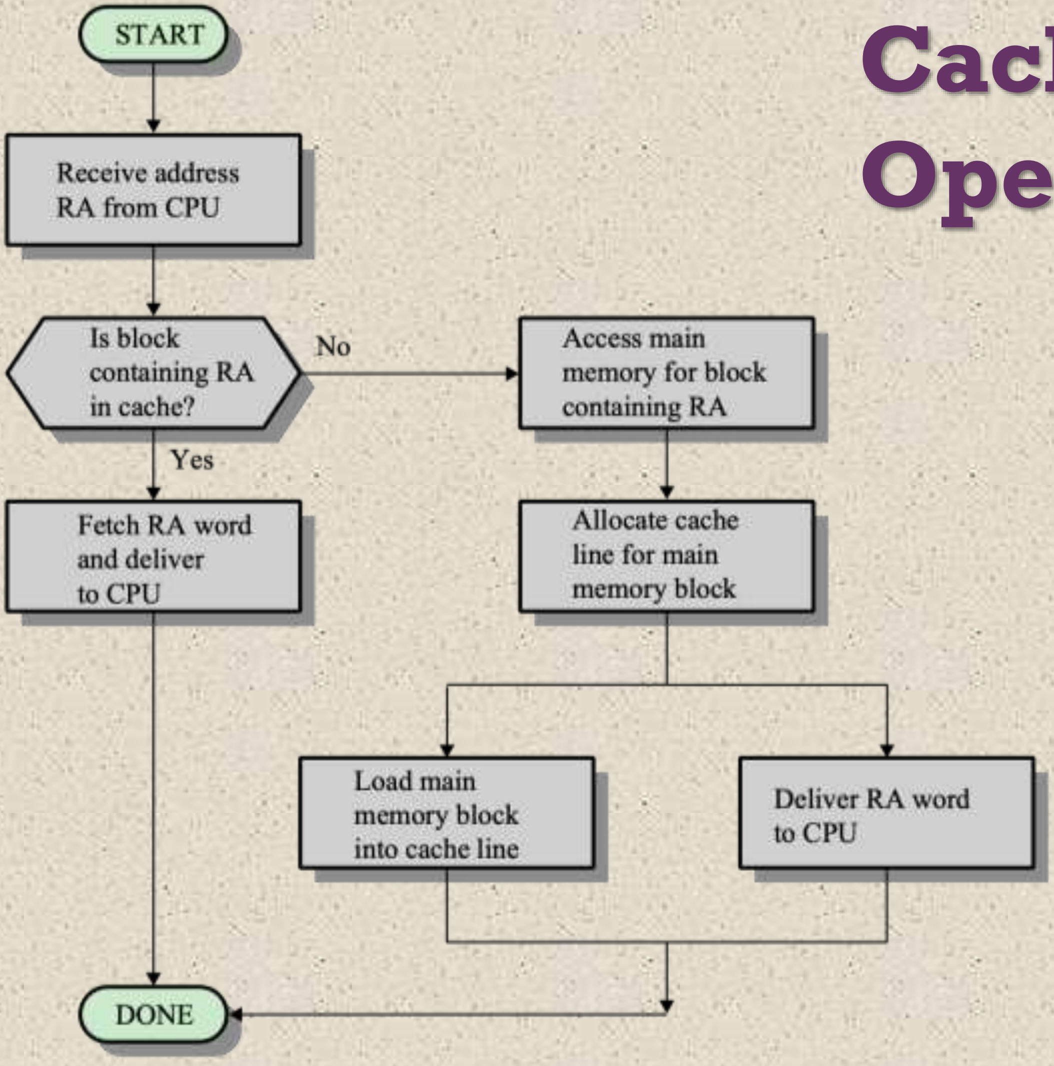
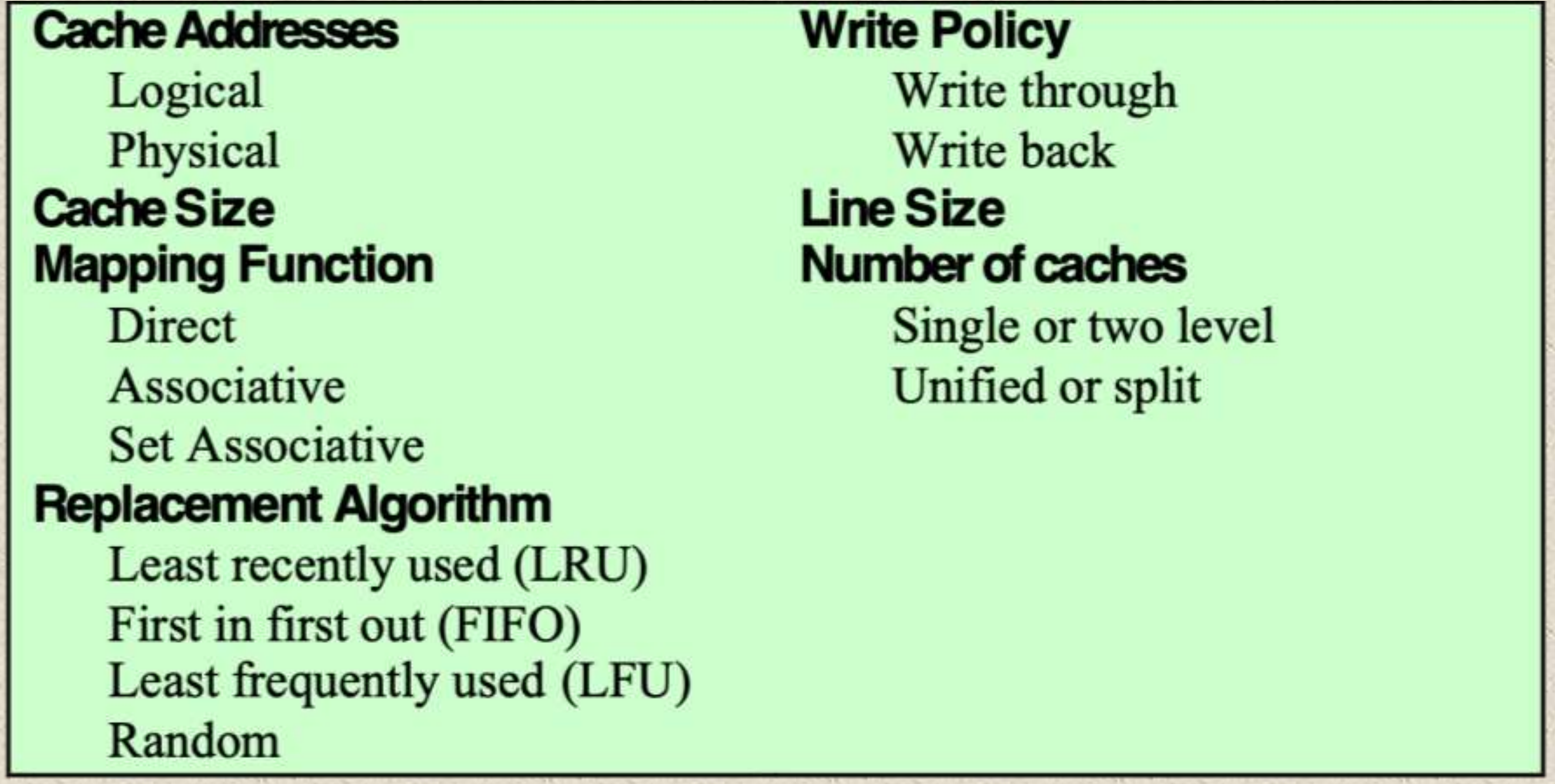

START
Receive address RA from CPU
Is block containing RA in cache?
No
Access main memory for block containing RA
Yes
Fetch RA word and deliver to CPU
Allocate cache line for main memory block
Load main memory block into cache line
Deliver RA word to CPU
DONE

Figure 4.5 Cache Read Operation

Elements of Cache Design



Cache Addresses
- Logical
- Physical

Cache Size

Mapping Function
- Direct
- Associative
- Set Associative

Replacement Algorithm
- Least recently used (LRU)
- First in first out (FIFO)
- Least frequently used (LFU)
- Random

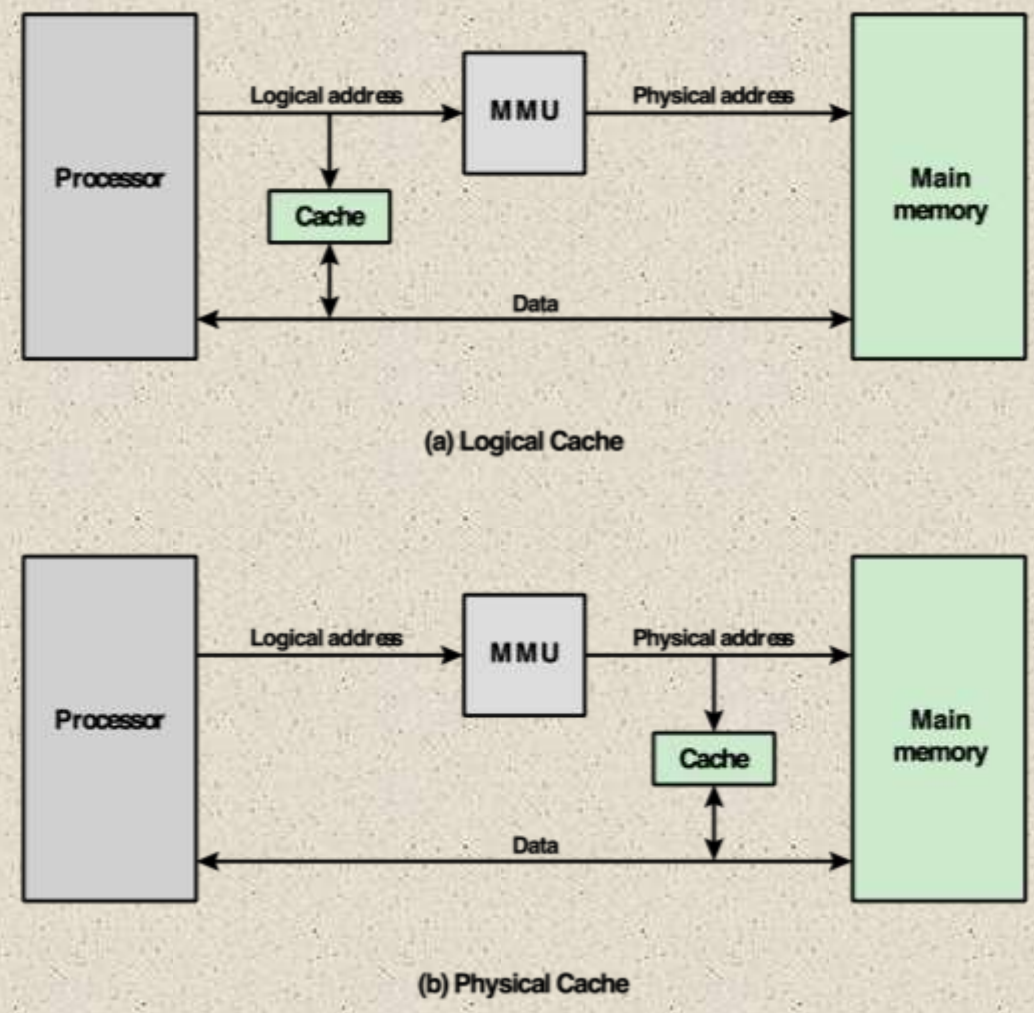
Write Policy
- Write through
- Write back

Line Size

Number of caches
- Single or two level
- Unified or split

Table 4.2 Elements of Cache Design

Logical and Physical Caches



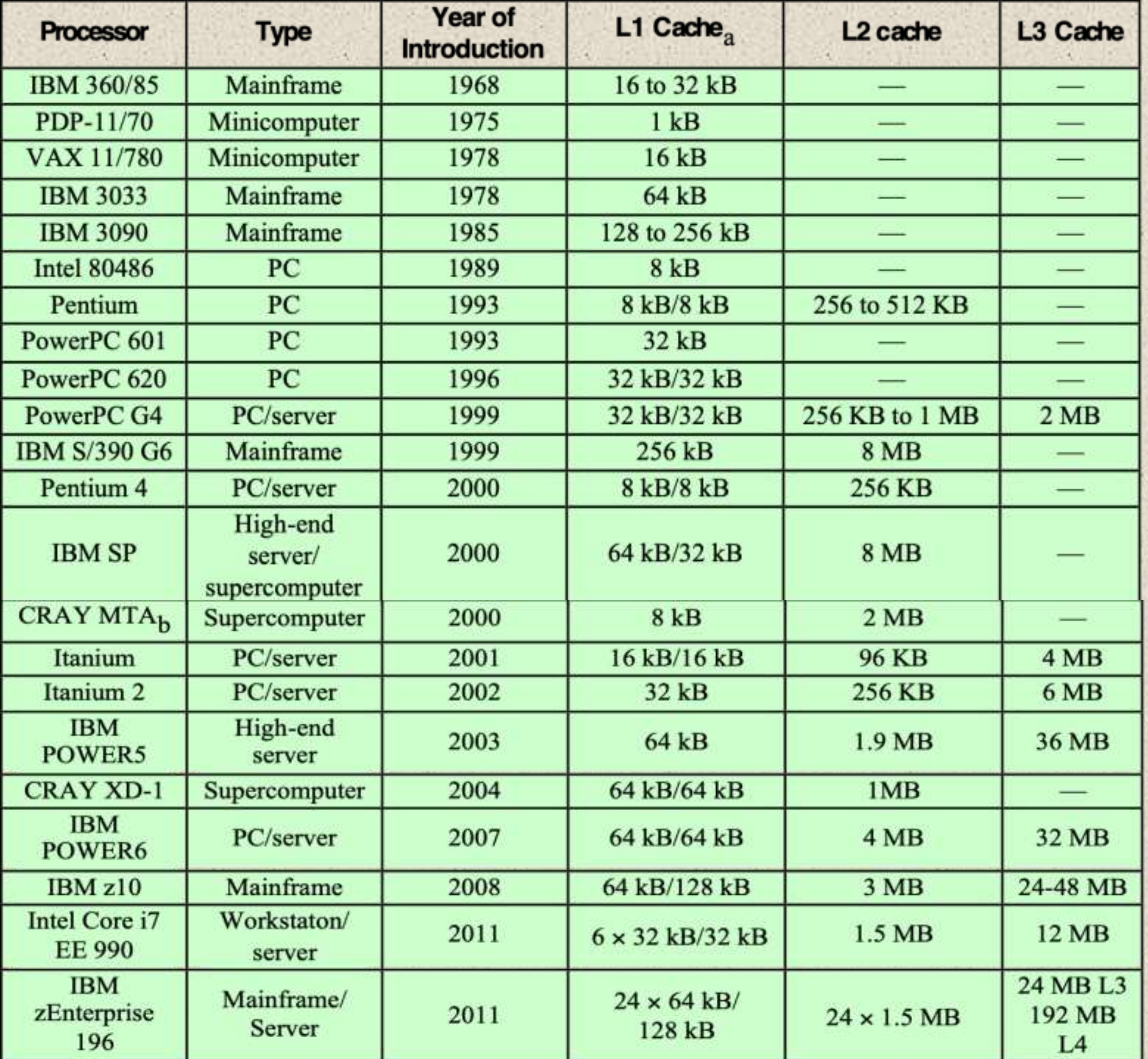
(a) Logical Cache

(b) Physical Cache

Figure 4.7 Logical and Physical Caches

Table 4.3

Cache Sizes of Some Processors



Processor	Type	Year of Introduction	L1 Cache[a]	L2 cache	L3 Cache
IBM 360/85	Mainframe	1968	16 to 32 kB	—	—
PDP-11/70	Minicomputer	1975	1 kB	—	—
VAX 11/780	Minicomputer	1978	16 kB	—	—
IBM 3033	Mainframe	1978	64 kB	—	—
IBM 3090	Mainframe	1985	128 to 256 kB	—	—
Intel 80486	PC	1989	8 kB	—	—
Pentium	PC	1993	8 kB/8 kB	256 to 512 KB	—
PowerPC 601	PC	1993	32 kB	—	—
PowerPC 620	PC	1996	32 kB/32 kB	—	—
PowerPC G4	PC/server	1999	32 kB/32 kB	256 KB to 1 MB	2 MB
IBM S/390 G6	Mainframe	1999	256 kB	8 MB	—
Pentium 4	PC/server	2000	8 kB/8 kB	256 KB	—
IBM SP	High-end server/ supercomputer	2000	64 kB/32 kB	8 MB	—
CRAY MTA[b]	Supercomputer	2000	8 kB	2 MB	—
Itanium	PC/server	2001	16 kB/16 kB	96 KB	4 MB
Itanium 2	PC/server	2002	32 kB	256 KB	6 MB
IBM POWER5	High-end server	2003	64 kB	1.9 MB	36 MB
CRAY XD-1	Supercomputer	2004	64 kB/64 kB	1MB	—
IBM POWER6	PC/server	2007	64 kB/64 kB	4 MB	32 MB
IBM z10	Mainframe	2008	64 kB/128 kB	3 MB	24-48 MB
Intel Core i7 EE 990	Workstaton/ server	2011	6 × 32 kB/32 kB	1.5 MB	12 MB
IBM zEnterprise 196	Mainframe/ Server	2011	24 × 64 kB/ 128 kB	24 × 1.5 MB	24 MB L3 192 MB L4

[a] Two values separated by a slash refer to instruction and data caches.

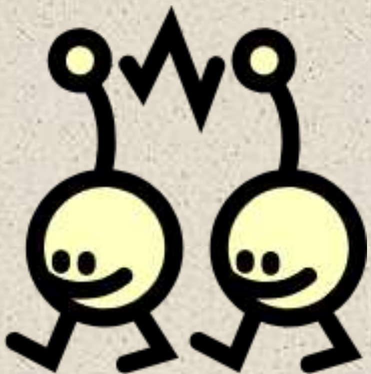
[b] Both caches are instruction only; no data caches.

+

Cache Addresses

Virtual Memory

- Virtual memory
  - Facility that allows programs to address memory from a logical point of view, without regard to the amount of main memory physically available
  - When used, the address fields of machine instructions contain virtual addresses
  - For reads to and writes from main memory, a hardware memory management unit (MMU) translates each virtual address into a physical address in main memory



- Mapping Function
  - Direct Mapping
  - Associative Mapping
  - Set Associative Mapping

End of Lecture