CS 451 / 551 / ECE 541

ADVANCED
COMPUTER ARCHITECTURE

SESSION no. 15

University of Idaho

ADDRESS SPACE

n BITS ⇒ 2^n MEMORY LOCATIONS

LOGICAL ADDRESSES → PHYSICAL ADDRESSES

n INCREASES ~ 1.5 BITS/YR.

$2^{1.5}$ = 3.8 × per year

DEC: DIGITAL EQUIPMENT CORP.

PDP-8	12 bits	4096 = 4K ADDRESSES
PDP-11	16 bits	64 K-ADDRESSES
		└ I & D SPACE
VAX	32 bits	└ 4 G-ADDR

University of Idaho

DO DATA SIZE & ADDRESS WIDTH HAVE TO BE THE SAME?

NO - BUT IF THEY ARE NOT, IT GETS COMPLICATED.

DATA OPERATIONS ON:

- PC
- REGISTERS (ADDR)
- MEMORY WORD
- DATA PATH: ADDERS (EFFECTIVE ADDR CALC.)
- INTERCONNECT - BUSSES

INTEL 8086

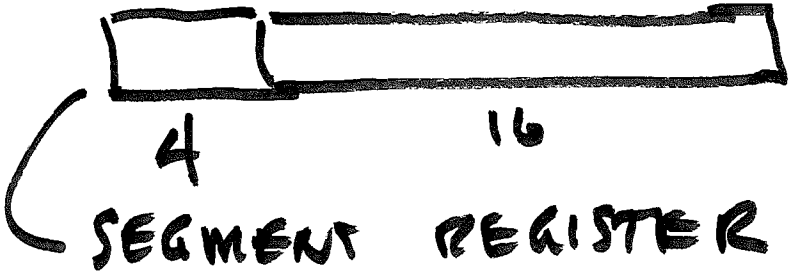
16 BIT INTERNAL ADDR & DATA

20 BIT EXTERNAL ADDR. =>

2^{20} = 1 M. ADDR

+ 8 BIT I/O

SEGMENTED MEMORY



INTEL 80x86

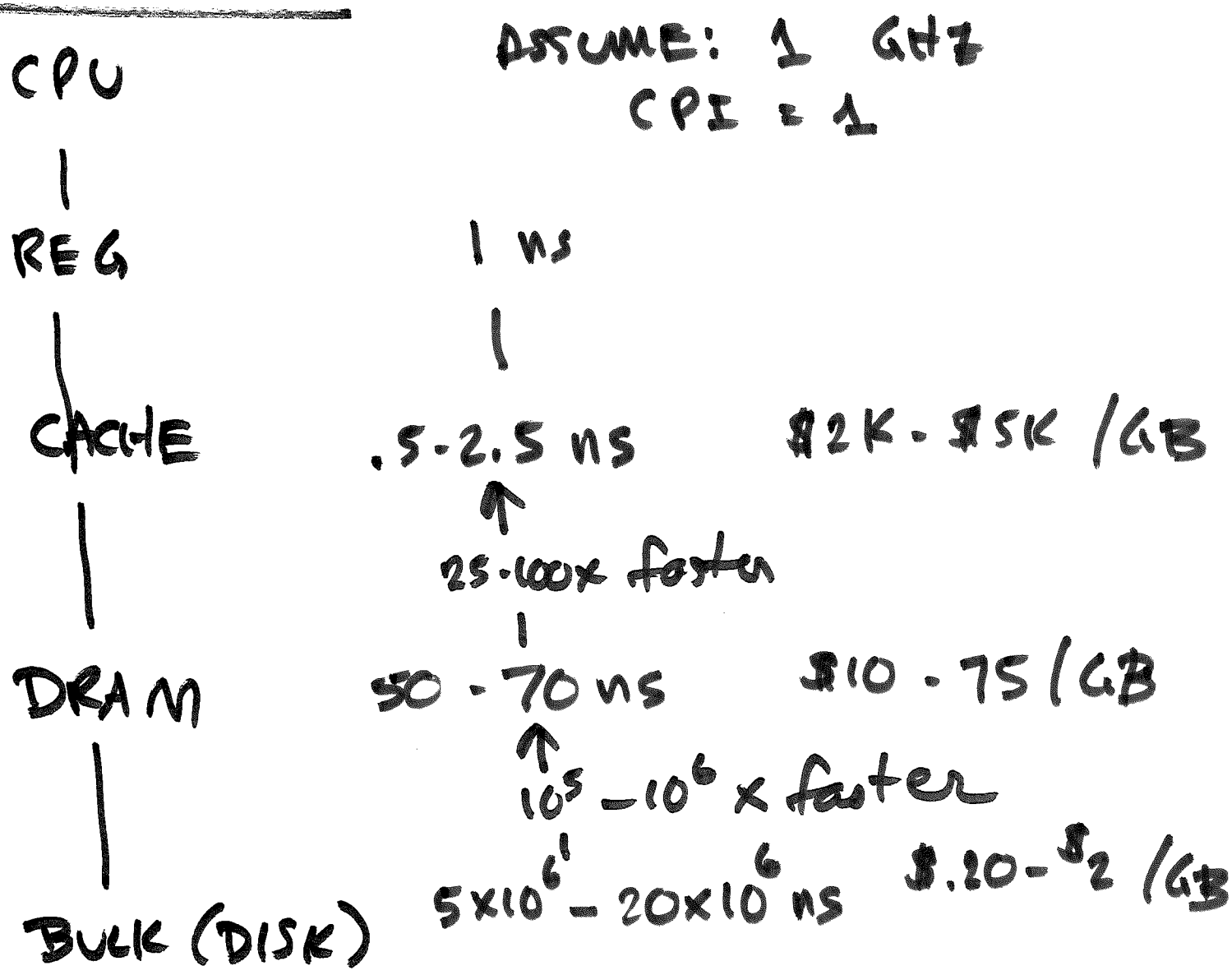
8086 16 bit

80386 32 bits (1985)

PENTIUM, OPTERON
64 BIT

University of Idaho

HIERARCHY



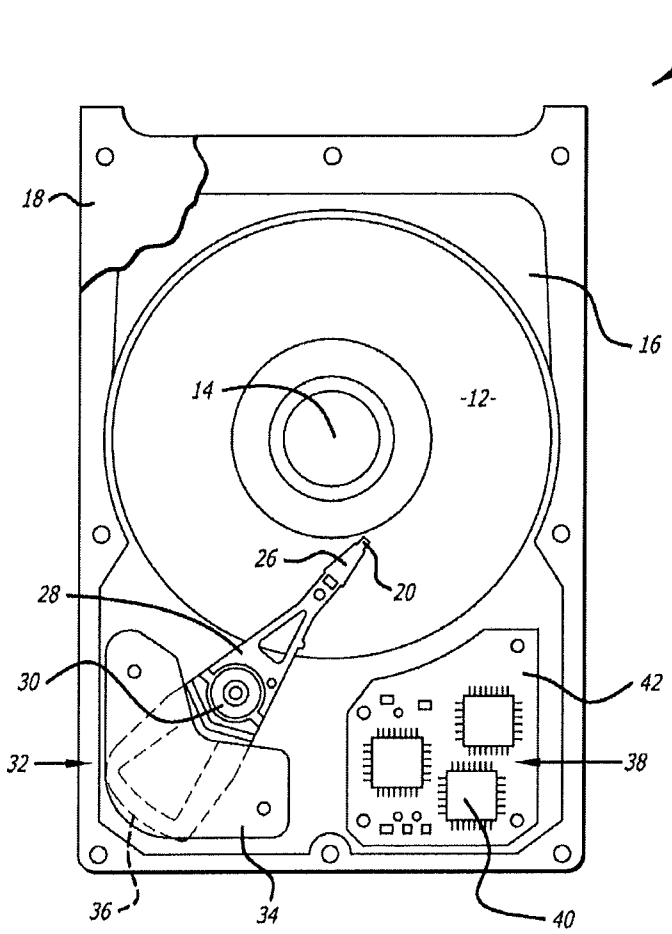
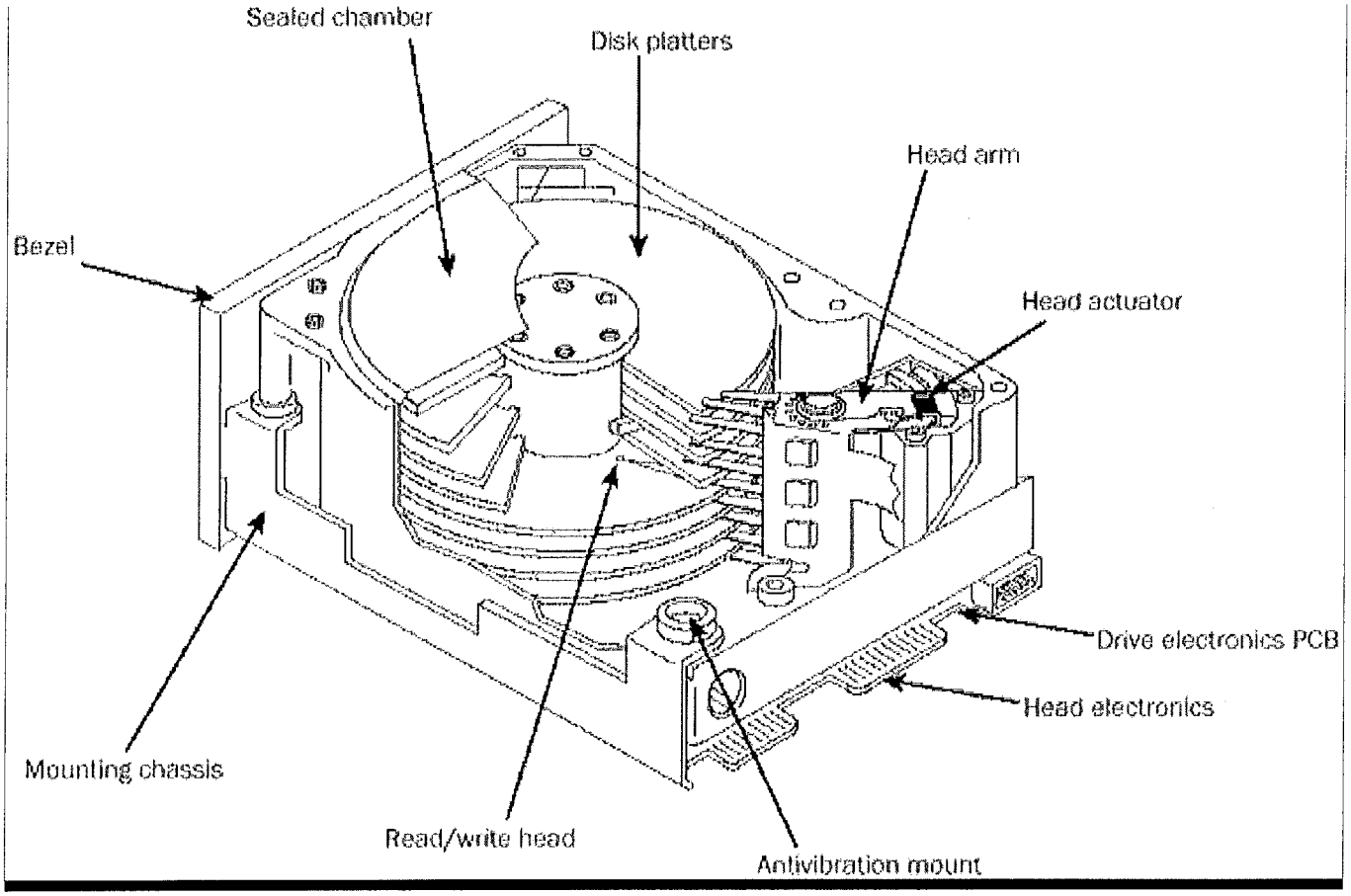


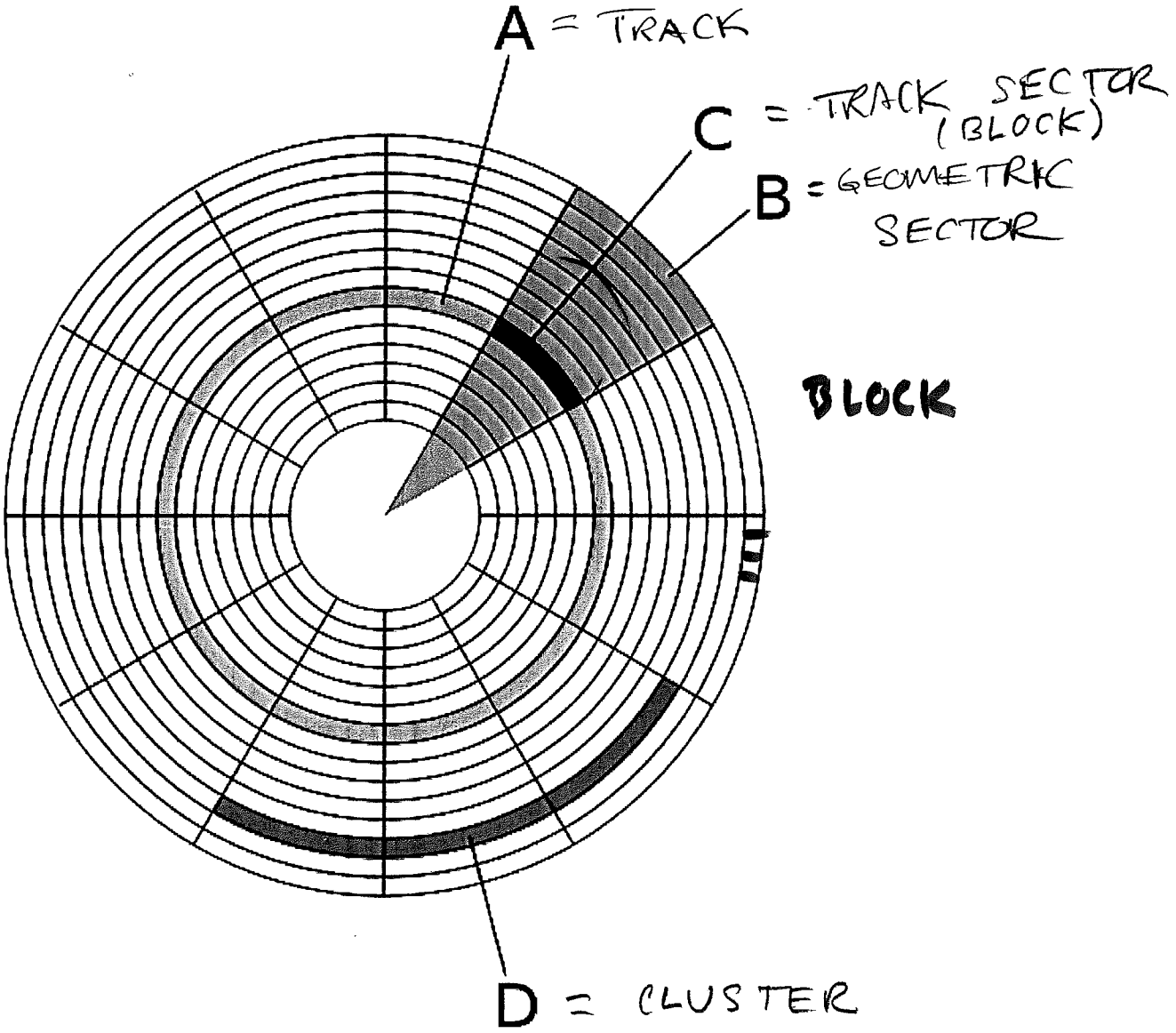
FIG. 1


Sealed chamber
Disk platters
Head arm
Bezel
Head actuator
Drive electronics PCB
Head electronics
Mounting chassis
Read/write head
Antivibration mount

7

File:Disk-structure2.svg

From Wikipedia, the free encyclopedia



Disk-structure2.svg (SVG file, nominally 600 × 600 pixels, file size: 7 KB)

This image rendered as PNG in other sizes: 200px, 500px, 1000px, 2000px.

This is a file from the Wikimedia Commons. Information from its **description page there** is shown below.
Commons is a freely licensed media file repository. You can help.

Summary

Description

Disk structure showing a track (A), a sector (B), a sector of track (C) and a cluster of sectors

CACHE ⟷ DRAM TRANSLATION

- FEW CYCLES
- HARDWARE
- TRANSPARENT TO CPU
  READ & WRITE WHOLE ADDR SPACE

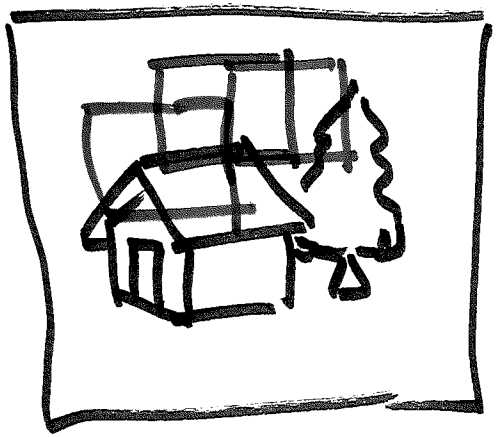
VIRTUAL MEMORY

- EXTEND PHYSICAL ADDR SPACE BEYOND DRAM
- 1 CPI @ 1 GHz
  10^7 ns access time ⇒
  10^7 instructions
- SOFTWARE — OS FUNCTION

University of Idaho

OLD DAYS - PROGRAMMER DID MAPPING

DATA: READ & WRITE BLOCKS

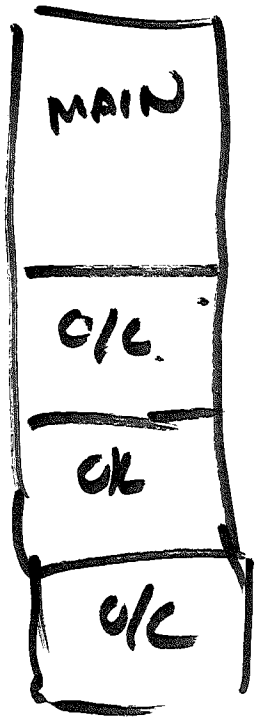


BIG IMPACT ON PROGRAMMER PRODUCTIVITY

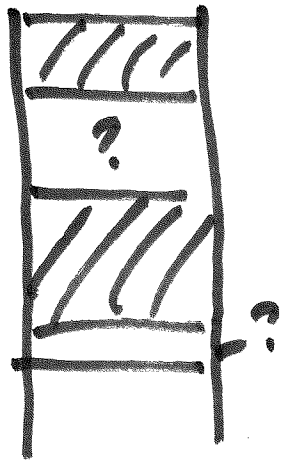
CODE: "OVERLAYS"

MOVED TO O/S: VIRTUAL MEMORY

malloc(sizeof(image))



TERMINOLOGY



VM. PAGE

PAGE FAULT

{ ADDRESS TRANSLATION
MEMORY MAPPING

PAGE - FIXED SIZE. 1 WORD ID

4096 · 892 BYTES

FRAGMENTATION - WASTED MEMORY

SEGMENT - VARIABLE SIZE

2 WORDS: ID, LENGTH

PAGED SEGMENTS

PAGE = min block size, multiple pages/segment

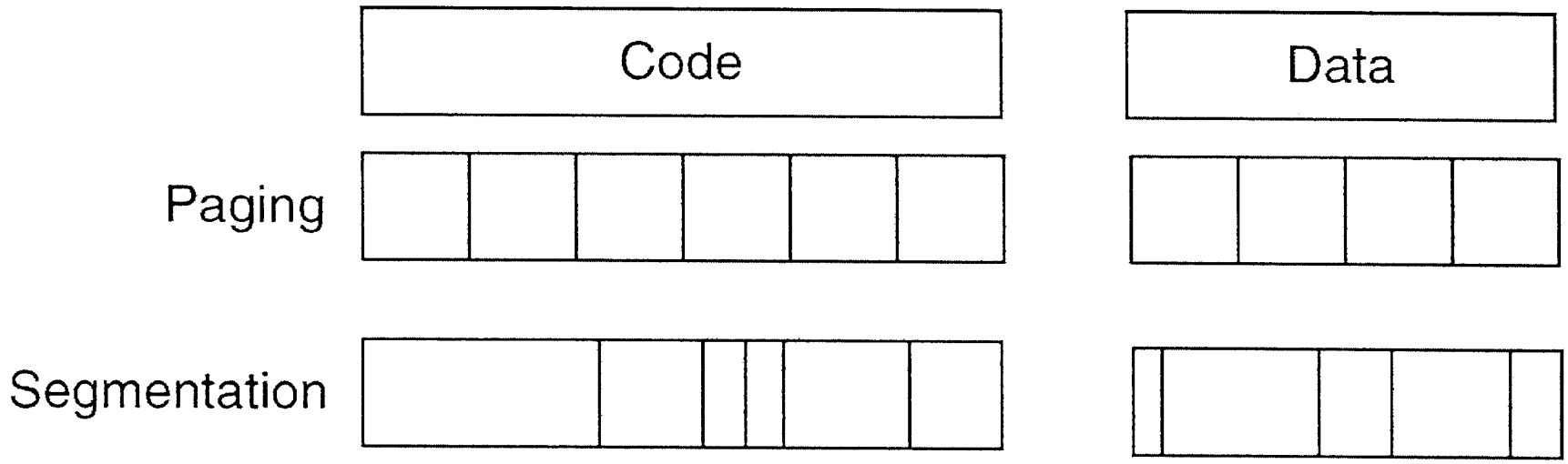


Figure B.21 Example of how paging and segmentation divide a program.

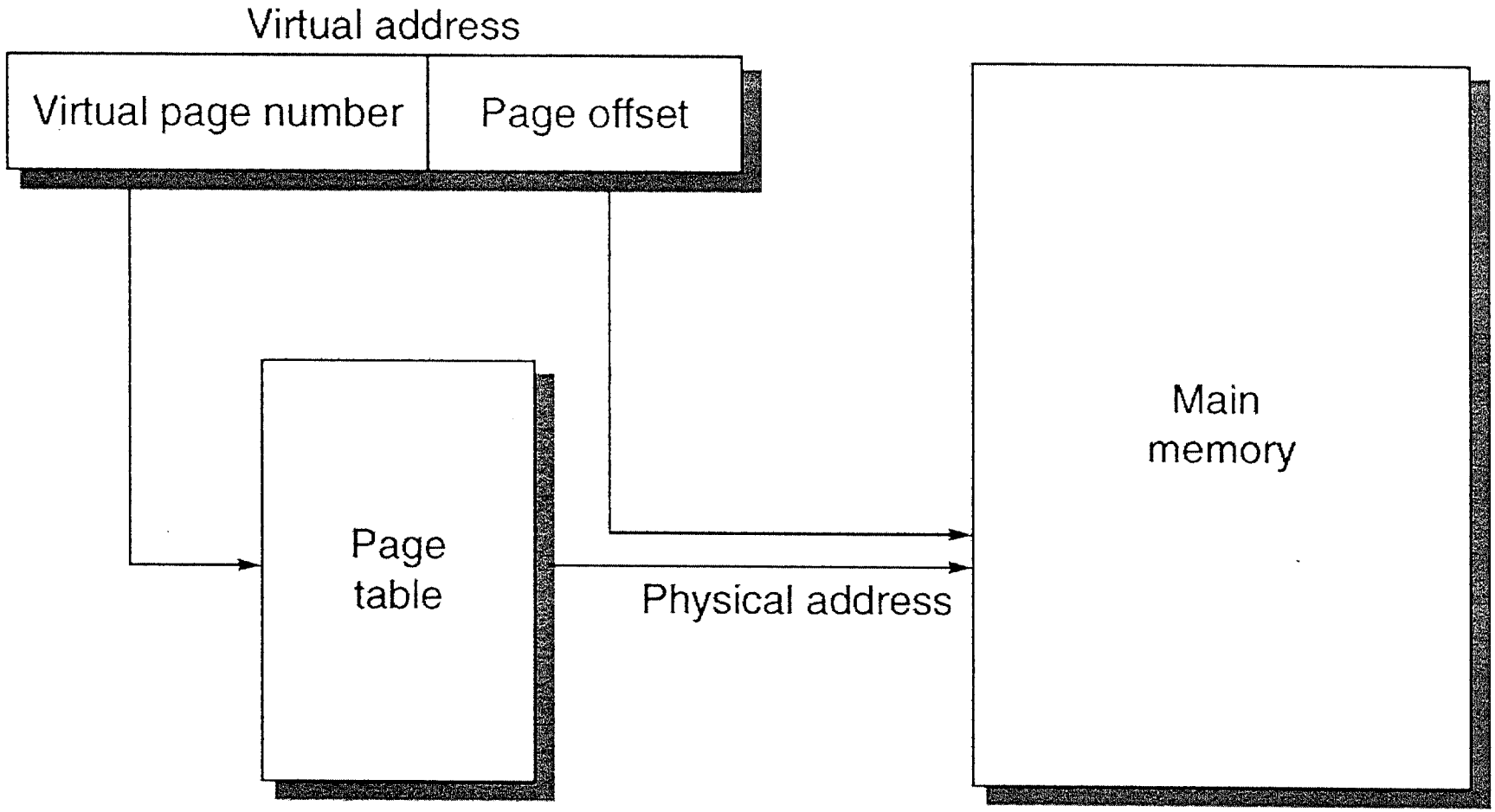


Figure B.23 The mapping of a virtual address to a physical address via a page table.

MAPPING IN SOFTWARE

PAGE TABLE

EX: 32 BIT ADDR SPACE $\Rightarrow 2^{32}$ locations

4 KB pages $\Rightarrow 2^2 \times 2^{10} = 2^{12}$ pages

Page table size:

$$\left(\frac{2^{32}}{2^{12}}\right) 2^2 = 4 \text{ MB}$$

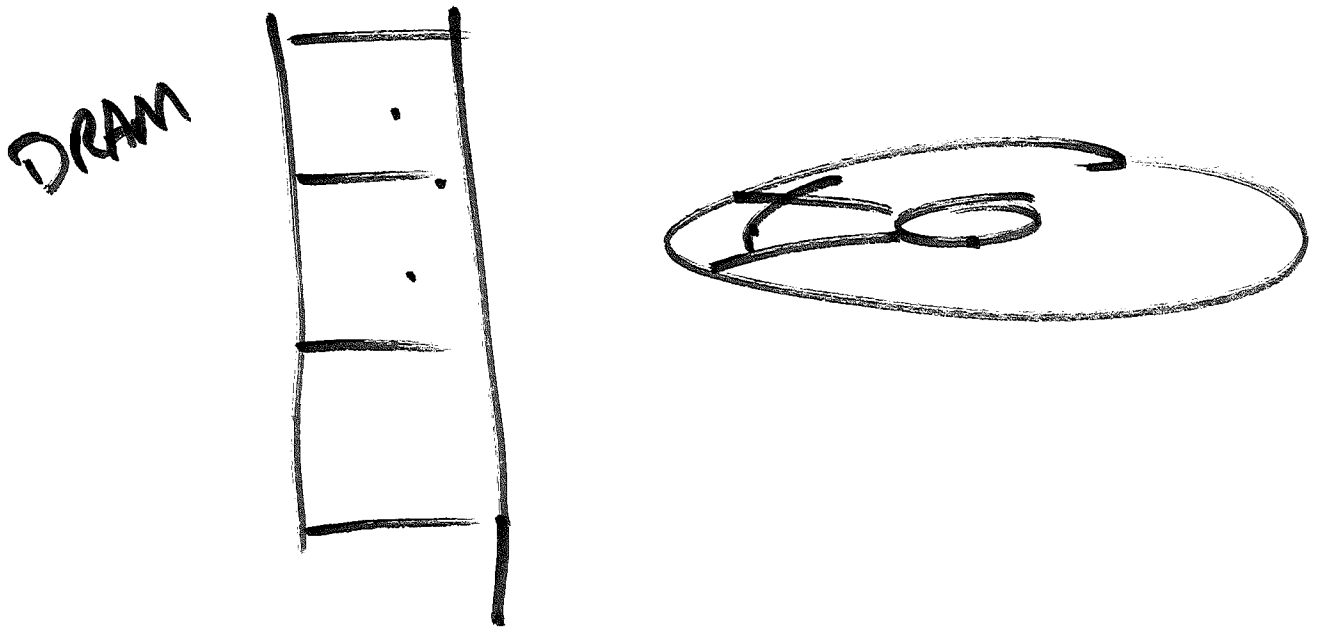
- SIZE OF A CACHE
  - SPECIAL CACHE JUST FOR PAGE TABLES
  - HARDWARE TO ACCELERATE ADDRESS MAPPING

TLB: TRANSLATION LOOKASIDE BUFFER

University of Idaho

FOUR QUESTIONS

Q1: WHERE CAN WE PLACE A BLOCK IN MEMORY?



A1: ANYWHERE!

COMPLEXITY NOT AN ISSUE.

University of Idaho

Q2: How Do WE FIND A BLOCK IN MEM?

A2: PAGE TABLE - TLB

Q3: WHICH BLOCK DO WE REPLACE?

A3: LRU - OPTIMAL USE OF LOCALITY

COMPLEX, BUT WHO CARES?

Q4: WHAT HAPPENS ON A WRITE?

~~WRITE THRU~~ - WRITE AS SOON AS CHANGE OCCURS

A4: WRITE BACK - WRITE WHEN WE REPLACE PAGE

LONG DISK ACCESS TIME - PREFER BLOCKS.