Memory Hierarchy

Primary memory / physical / main mem. (original)

RAM → constant power supply, volatile
ROM → non-volatile
} chip memory, expensive, fast

secondary mem (to increase storage capacity)

HD →
FD (floppy disk) } magnetic disk

CD →
DVD → } optical disk

pendrive - semiconductor

} writtable
non volatile,
cheap
need large power

HD, FD, CD, DVD

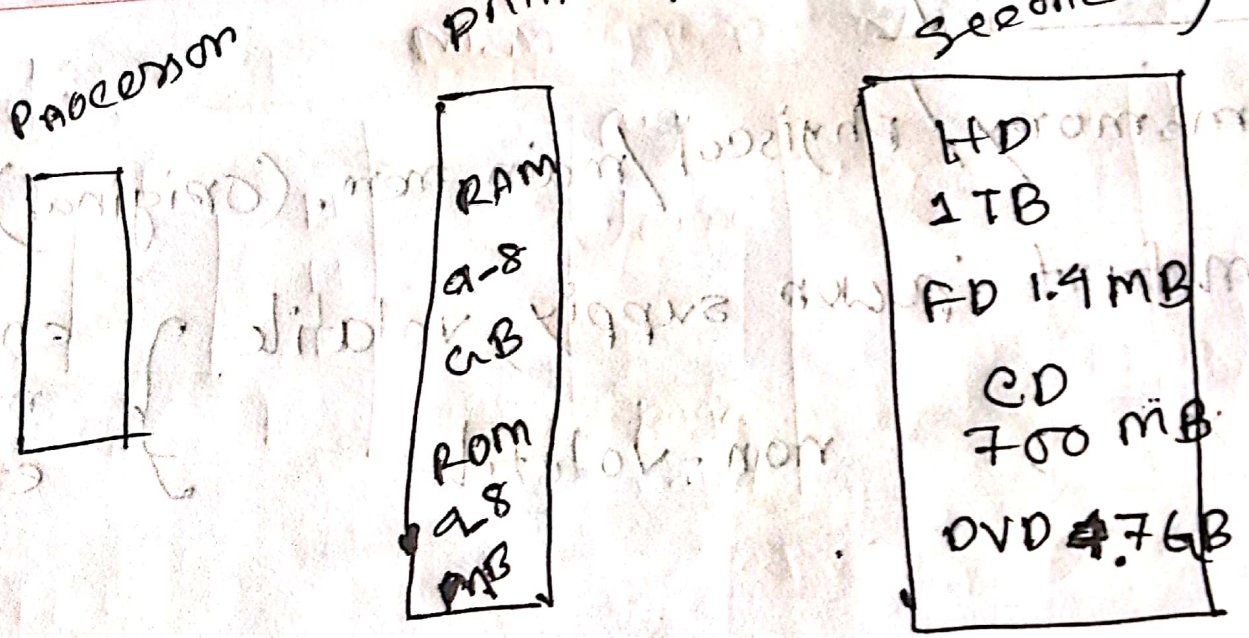
data acquining rate =
rate of disk rotation (rpm)

(fast + cheap) → secondary mem

for operation / for storage

Q. How speed increase of . by increasing RAM size?

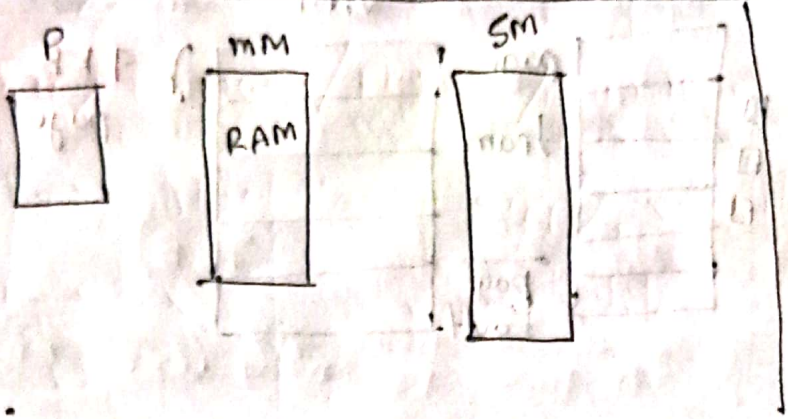
আমির জোরে চলার সময় কোন ব্রেক ই আসে
চড়ার মত বন্ধু কে আছে?



Lecture - 20

Date: 27.06.22

Virtual Memory Management through Paging:



pendrive (-flash ROM)
written and
non volatile.
SSD → Solid state disk.

ROM:

BIOS:

basic i/p o/p systems

OP run → load in RAM

Paging: It is a virtual memory management system. It tells how information come from hard disk into RAM.

~~Paging~~ **Page:** Division of virtual memory.

Page frame: Division of main memory.

Hit: If processor find something in MM.

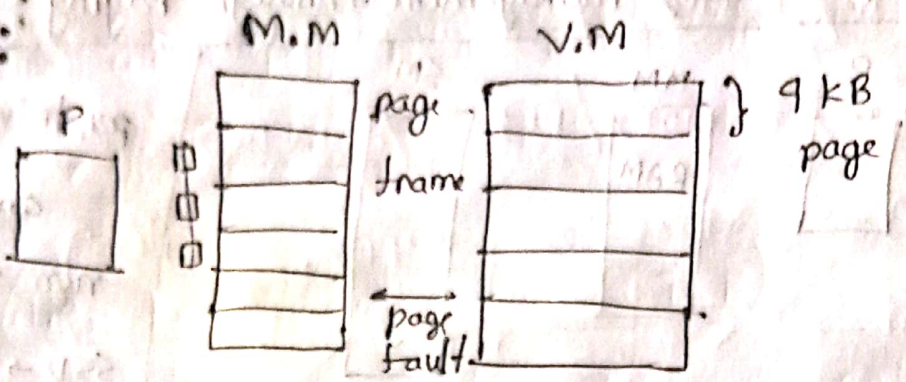
Page fault: If processor don't find something in MM, go to V.M.

Hit Ratio: $\frac{N_1}{N_1+M_2}$ = num of hit upon total attempt.

↙ how many times found

↘ and not found

Paging:



- All page size is 9kB.
- If RAM is full, then extra part for keeping some portion of RAM will be replaced.

Page Replacement:

What are the conditions for replacement?

1. Page Fault
2. Page should be empty.

Whenever there is page fault and there is no empty page available in main memory, page from the main memory is replaced by the page from virtual memory.

Dirty bit: by default 0, when write operation: 1

if dirty bit = 0, replace

" " " = 1, copy and replace dirty page.

Page replacement algo:

i) FIFO → First in first out

ii) LRU → Least recently used

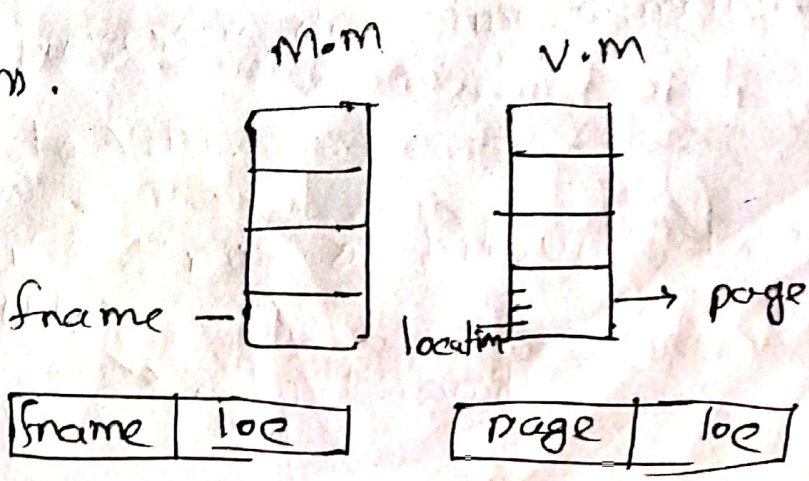
iii) LFU → " frequently "

Thrashing: when too many page fault.

LRU: Page that have not used for the longest time will be replaced.

LFU: Page in the lowest count will be replaced.

Address translation: Converting Virtual Address into physical Address.



To know which page is in which frame, there is a table called page table.

Translation look ahead buffer: (TLB)

TLB keep the most recently used entry / page

1st search, TLB → then page table (slow)

simultaneously search 'TLB' and page table

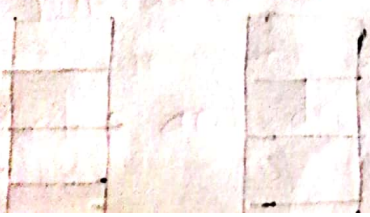
TLB: 32 most recently page.

CT-3: pipelining, Hazard

Addressing mode

without state diagram

mapping

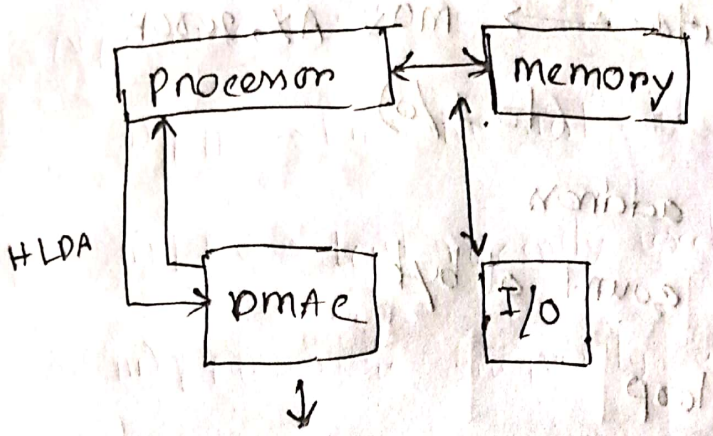


Lecture-21

Date: 16.07.22

DMA : Direct Memory Access

↳ Transferring data between memory and I/O without utilizing processor.



DMA controller

To do data transfer control signal is needed. that's why memory and I/O unit transfer data.

→ DMA is used for bulk amount of data transfer.

Bus master → control bus

processor → generates control signal

Read → transfering data memory to processor

Q. Why can we not transfer data between memory and I/O ?

Q. What is DMA concept? When we use it?

4 steps for 2 bytes, small amount. DMAC is needed when heavy amount of data is required. It's a controller, hand ware based.

Example of instruction:

take data ⟶ MOV AX, 2000H
send (to I/O)
INC address
DEC count of byte
check loop

Here, 5 fetch and 5 decodes will happen for 1 byte.
for 1 byte, loop will run 1 cycle
" 1KB " " 1000 cycle and 1kB data transfer

Q. Why DMA is efficient?

⇒ 1. Here data transfer happens directly without processor. So, cycle reduces here.

2. As, it is hand ware based, so no instruction should be given for fetch and decoding. So, it requires no time for fetching and decoding.

Steps: 1. DMAC give hold=1

2. P relinquishes system bus

3. P gives HLDA

4. DMAC perform data transfer

5. DMA gives hold=0

6. P again become bus master

DMAC contains two reg →

CAR → current address register

(gives the starting address of transferred file)

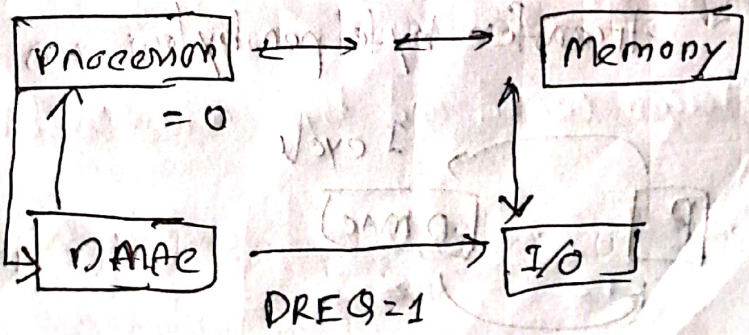
CWCR → Current Word count Register

(gives the size of transferred file)

P → DMAC (CAR and CWCR) → wait for a confirmation

→ DMAR; h=1, → HLDA → DACK to I/O →

DMAC start to data transfer → CAR++, CWCR=0 (terminal



Lecture-22 Date: 18-07-22

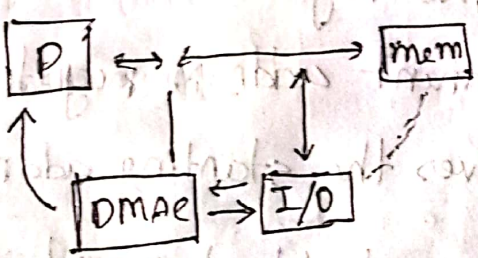
Types of DMA

i) Block transfer / Burst Method → continous use of bus ✓ fast

ii) Cycle stealing / 1 byte transfer

iii) Demand transfer

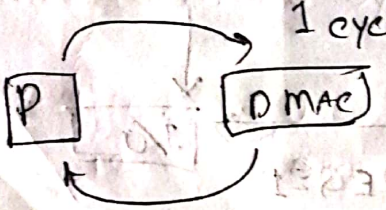
iv) Hidden Mode / No need to fetch and decode, only execution



Add inc
Byte transfer dec (CWCR) reg → 0

i) Block transfer → continous use of bus : Advantage : fast
Dis - processor lies idely.

ii) Cycle stealing : Advantage : fast but slower than block transfer, twice of it.
↓
transfer byte per byte.



iii) Demand transfer (added feature) DREQ = 1/0

take some intermidate time to get ready for next stage. It is ensured by checking DREQ.

iv) Hidden mode:

when processor remains idle, this DMA work or take data from processor.

CISC: i) Complex instruction set computer

ii) Flexible inst. format

iii) more addressing mode

iv) memory based

v) less register

vi) poor pipelining (may have gap)

vii) variable execution time

viii) microprogrammed CU

ix) microprocessor 8085, 8086 (example)

RISC: i) Reduced instruction set computer.

ii) regid inst. format

iii) less addressing mode

iv) register based (load, store mem)

v) more reg

vi) Exellent pipelining (bus free)

vii) fixed execution time

viii) hardware CU

ix) microcontroller, ARM, PIC