

Indira Gandhi Govt PG College

Vaishali Nagar, Bhilai

Department of Computer Science

UNIT III

Computer Fundamental & Operating System

CSSC/CSGE - 01

Memory: Detailed Hindi-English Notes
With Relevant Diagrams | 5 and 10 Marks Questions

Head of Department

Suresh Kumar Thakur

Asst Prof.

B Sc/B A/B H Sc 1st Sem

Unit III: Memory — Study Guide

यह booklet B Sc/B A/B H Sc 1st Semester के विद्यार्थियों के लिए है। सभी syllabus topics सरल Hindi-English में दिए गए हैं। Diagrams conceptual हैं और scale पर नहीं बने हैं।

Topic	मुख्य Subtopics
Memory hierarchy	Registers, cache, RAM, secondary storage; speed/capacity/cost
Primary / Secondary memory	RAM, ROM, types, comparison and storage
Cache memory	Hit/miss, locality, mapping and write policies
Virtual memory	Pages, frames, page table and page faults
DASD	Disk structure, access time and direct access
Instruction format	Opcode, operands, addressing fields and formats
Program / Data memory	Stored instructions, variables and architectures
MMU	Address translation, protection and TLB

5 और 10 अंक का उत्तर कैसे लिखें?

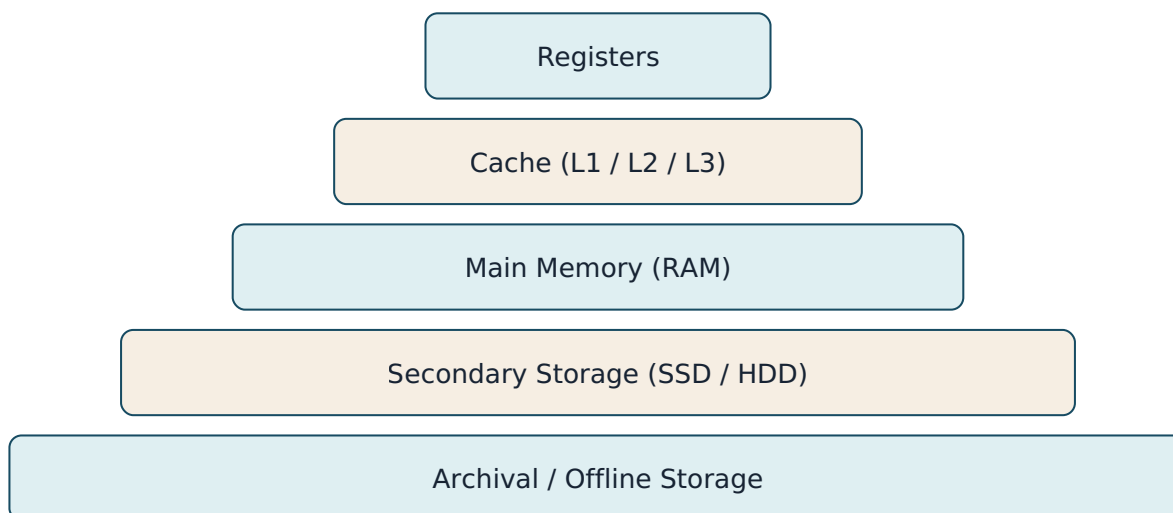
5 अंक: Definition, labelled diagram, मुख्य बिंदु तथा example। **10 अंक:** Introduction, labelled diagram, components, working, types, advantages/limitations तथा example या numerical। Question के अनुसार headings चुनें।

Basic Units

1 bit = binary 0 या 1; 1 byte = 8 bits। Binary units: 1 KiB = 1024 bytes, 1 MiB = 1024 KiB। Decimal units: 1 kB = 1000 bytes। Questions में दिए unit convention का पालन करें।

1. Memory Hierarchy

Memory hierarchy computer में memories को speed, capacity और cost per bit के आधार पर levels में व्यवस्थित करने की प्रणाली है। एक ही memory से बहुत अधिक speed, बड़ी capacity और कम cost प्राप्त करना कठिन होता है। इसलिए कई levels मिलकर काम करते हैं।



Up: faster, smaller, higher cost per bit

Down: larger, slower, lower cost per bit (general trend)

Figure 1. Memory hierarchy: ऊपर speed अधिक; नीचे capacity सामान्यतः अधिक।

Levels का वर्णन

- **Registers:** CPU के अंदर छोटे storage locations; current operands, addresses और intermediate results रखते हैं।
- **Cache:** CPU के पास frequently/recently used memory blocks की copies; सामान्यतः SRAM आधारित।
- **Main memory:** RAM में currently running programs और data रहते हैं।
- **Secondary storage:** SSD/HDD पर programs और files लंबे समय तक रहती हैं।
- **Archival storage:** Backup और long-term retention; tape इसका उदाहरण है।

Hierarchy कैसे लाभ देती है?

CPU को आवश्यक data पहले तेज level पर खोजा जाता है। Nearby/frequently used data cache में रखने से अधिकांश accesses तेज हो सकते हैं, जबकि बड़ा data नीचे के levels में रखा जाता है।

महत्वपूर्ण: Capacity और speed की तुलना सामान्य प्रवृत्ति है; हर device के लिए एक fixed numerical value नहीं होती।

1.1 Memory Characteristics & Locality

मुख्य विशेषताएँ

Term	Meaning
Capacity	कितना data store हो सकता है
Access time / latency	Request और data मिलने के बीच समय
Transfer rate	प्रति second transfer होने वाला data
Cost per bit	प्रत्येक bit storage की लागत
Volatility	Power हटने पर data बना रहता है या नहीं
Memory cycle time	Memory operations के बीच आवश्यक समय

Locality of Reference

Programs अक्सर सीमित समय में कुछ instructions/data को बार-बार उपयोग करते हैं। इसी व्यवहार से cache और virtual memory प्रभावी बनते हैं।

- **Temporal locality:** हाल में उपयोग हुआ data/instruction फिर उपयोग हो सकता है। Example: loop का instruction।
- **Spatial locality:** उपयोग किए address के पास के locations जल्द उपयोग हो सकते हैं। Example: array को क्रम से पढ़ना।

Capacity Numerical

यदि memory में 1024 locations हैं और हर location 8 bits की है, capacity = $1024 \times 8 = 8192$ bits = 1024 bytes = 1 KiB।

Address Lines Numerical

यदि memory byte-addressable है और address lines 16 हैं, तो locations = $2^{16} = 65,536$ । Capacity = 65,536 bytes = 64 KiB। Word-addressable system में capacity निकालते समय word size से गुणा करना होगा।

Hierarchy के लाभ और सीमा

लाभ: तेज average access, बड़ी practical capacity और cost का संतुलन। सीमा: data movement, replacement और consistency management में complexity।

5 अंक का प्रश्न: Memory hierarchy diagram के साथ समझाइए। **10 अंक:** Levels, characteristics, locality तथा advantages सहित समझाइए।

2. Primary Memory

Primary memory CPU द्वारा directly addressable internal memory को कहा जाता है। Introductory classification में RAM और ROM इसके मुख्य रूप हैं। Cache और registers को hierarchy में अलग तेज levels के रूप में भी पढ़ा जाता है।

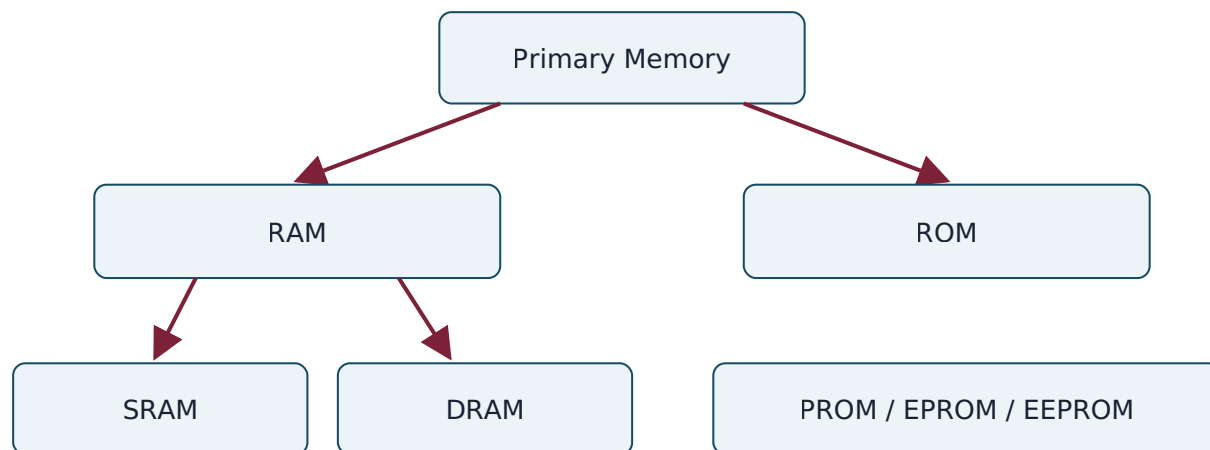


Figure 2. Primary memory की सामान्य introductory classification।

RAM — Random Access Memory

RAM read/write working memory है। Computer में चल रहे programs, current data और intermediate results सामान्यतः RAM में होते हैं। Main-memory RAM सामान्यतः volatile होती है; power हटने पर उसका data चला जाता है। Random access का अर्थ location के address से access है।

SRAM — Static RAM

Bits bistable circuits में store होते हैं; powered रहने पर periodic refresh की आवश्यकता नहीं होती। यह तेज लेकिन प्रति bit महँगी और कम dense होती है। Cache में सामान्य उपयोग। Static का अर्थ non-volatile नहीं है।

DRAM — Dynamic RAM

Bits capacitor charge से represent होते हैं। Charge leak होने के कारण refresh आवश्यक है। यह SRAM की तुलना में सामान्यतः अधिक dense और सस्ती होती है; main memory में उपयोग होती है।

Basis	SRAM	DRAM
Storage	Bistable circuit	Capacitor charge
Refresh	Periodic refresh नहीं	आवश्यक
Speed / Cost	सामान्यतः तेज / महँगी	सामान्यतः धीमी / सस्ती
Use	Cache	Main memory
Volatility	Volatile	Volatile

2.1 ROM and Its Types

ROM = Read-Only Memory. यह non-volatile memory family है जिसका उपयोग firmware तथा relatively permanent instructions के लिए किया जाता है। सभी ROM types को सामान्य runtime RAM की तरह बार-बार नहीं लिखा जाता।

Type	Detailed explanation
Mask ROM	Manufacturing के समय contents निर्धारित; सामान्य उपयोगकर्ता reprogram नहीं कर सकता।
PROM	Programmable ROM; सामान्यतः एक बार program की जा सकती है।
EPROM	Ultraviolet light से erase करके reprogram; सामान्यतः chip को UV exposure देना पड़ता है।
EEPROM	Electrical तरीके से erase और reprogram; fine-grained updates संभव।
Flash memory	Non-volatile technology; सामान्यतः blocks में erase। Firmware, SSD और pen drives में उपयोग।

Firmware

Firmware hardware control के लिए stored software है। Modern systems में firmware अक्सर rewritable flash memory में रहता है। इसे केवल permanently fixed ROM मानना सही नहीं है।

RAM vs ROM

RAM	ROM family
Working programs/data	Firmware/permanent instructions
Normal read/write operations	Writing capability type पर निर्भर
Typical main-memory RAM volatile	Non-volatile
Running applications	Startup/control software

Primary Memory के लाभ एवं सीमाएँ

Primary working memory CPU को comparatively fast access देती है। RAM की capacity secondary storage से सामान्यतः कम और cost per bit अधिक होती है। Volatility के कारण persistent files को secondary storage पर save करना चाहिए।

परीक्षा में: RAM/ROM definition, classification diagram, SRAM-DRAM तथा ROM types के उदाहरण लिखें।

3. Secondary Memory

Secondary memory programs और data को long-term रखने का storage है। यह सामान्यतः non-volatile, अधिक capacity वाली और RAM से कम cost per bit वाली होती है। सामान्य desktop execution में stored program RAM में load होकर execute होता है।

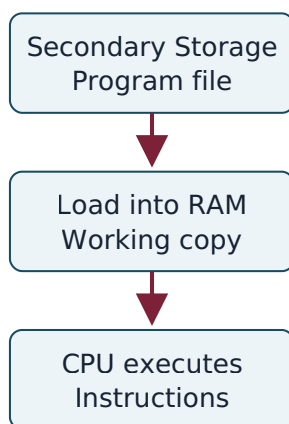


Figure 3. सामान्य program loading और execution का conceptual क्रम।

Device	Working / Use
HDD	Magnetic rotating platters; large-capacity storage
SSD	Flash-based storage; moving mechanical parts नहीं
USB flash drive	Portable flash storage
Optical disc	Laser-based reading; CD/DVD/Blu-ray
Magnetic tape	Sequential access; backup/archive

Primary vs Secondary Memory

Primary working memory	Secondary storage
Fast working area	Long-term storage
CPU memory operations के लिए उपलब्ध	I/O और storage controllers द्वारा access
Typical RAM volatile	आमतौर पर non-volatile
Comparatively smaller capacity	Comparatively larger capacity
Examples: RAM	Examples: SSD, HDD

महत्व: OS, applications, student records, documents और backups power off होने पर भी सुरक्षित रखे जा सकते हैं। Storage device की reliability, backups और access controls फिर भी आवश्यक हैं।

4. Cache Memory: Meaning & Working

Cache CPU के पास स्थित छोटी और तेज memory है जो lower-level memory के selected blocks की copies रखती है। इसका उद्देश्य average memory access time घटाना है। यह सामान्यतः SRAM आधारित होती है।

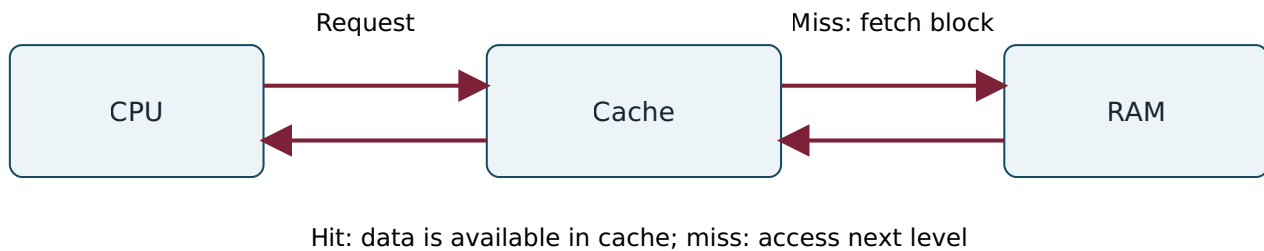


Figure 4. CPU-cache-RAM interaction; वास्तविक systems में multiple cache levels होते हैं।

Cache Hit and Miss

Hit: Requested block cache में मिल गया। **Miss:** Block नहीं मिला; अगले memory level से लाना होगा। Cache पूरी RAM की copy नहीं रखती।

Working Steps

- CPU memory address generate करता है।
- Cache address tag तथा relevant entry की जाँच करती है।
- Hit होने पर data/instruction दिया जाता है।
- Miss होने पर next level से block fetch होता है।
- जरूरत हो तो पुराने block को replacement policy से हटाया जाता है।

Levels

L1: CPU core के बहुत पास, छोटी और तेज; instruction/data अलग हो सकती है। **L2:** सामान्यतः बड़ी और कुछ धीमी। **L3:** कई designs में बड़ी और cores के बीच shared। Exact arrangement processor पर निर्भर है।

Locality

Temporal locality repeated access में और spatial locality पास के addresses के access में cache को लाभ देती है। Cache line/block में कई consecutive bytes रखे जा सकते हैं।

लाभ: CPU waiting कम और average access तेज। **सीमाएँ:** capacity कम, cost अधिक तथा coherence/replacement complexity।

4.1 Cache Mapping, Policies & Numerical

Mapping Techniques

Technique	Explanation
Direct mapping	हर memory block केवल एक निर्धारित cache line में जा सकता है। सरल, लेकिन conflict misses अधिक हो सकते हैं।
Fully associative	Block किसी भी cache line में जा सकता है। Flexible, लेकिन associative search महंगा।
Set associative	Block एक निर्धारित set में जाता है, लेकिन उस set की कई ways में से किसी में रह सकता है।

Direct mapping: Cache line index = memory block number mod number of cache lines। Example: 8 lines वाली cache में block 18 → line 2।

Replacement Policies

LRU: सबसे लंबे समय से उपयोग न हुए block को हटाना। **FIFO:** पहले आए block को पहले हटाना। **Random:** Random choice। Replacement की जरूरत उस समय होती है जब allowed locations भरी हों।

Write Policies

Write-through: Write cache और next level तक भेजा जाता है। **Write-back:** Modified block cache में रहता है; eviction पर next level update होता है। Write-allocate/no-write-allocate write miss की अलग policies हैं।

Hit Ratio

Hit ratio = hits / total accesses। Miss ratio = 1 – hit ratio। यदि 1000 accesses में 920 hits हैं, hit ratio = 0.92 = 92%; miss ratio = 8%।

Average Memory Access Time

AMAT = Hit time + Miss rate × Miss penalty. यहाँ miss penalty, miss के कारण लगने वाला अतिरिक्त समय है।

Example: hit time 2 ns, miss rate 0.05 और additional miss penalty 80 ns। $AMAT = 2 + 0.05 \times 80 = 6$ ns।

Trap: यदि question में miss का total service time दिया हो, तो timing definition देखकर formula लगाएँ; penalty और total miss time समान नहीं हैं।

5. Virtual Memory

Virtual memory एक memory-management technique है जो processes को virtual address spaces देती है और केवल जरूरी pages को RAM में रख सकती है। इससे program के सभी भागों का एक साथ RAM में होना आवश्यक नहीं रहता।

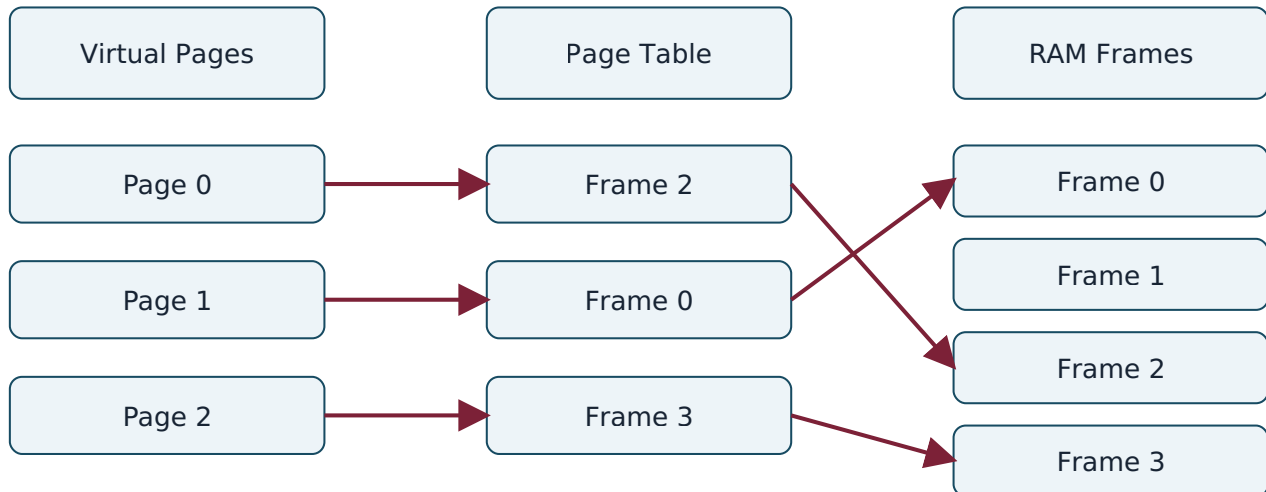


Figure 5. Illustrative page-to-frame mapping; virtual pages अलग physical frames में जा सकती हैं।

Pages, Frames and Page Table

Virtual memory fixed-size **pages** में और physical RAM समान size के **frames** में बाँटी जा सकती है। Page table virtual page को physical frame से map करती है और valid/present तथा protection information रख सकती है।

Demand Paging

Page को आवश्यकता पड़ने पर RAM में load किया जाता है। Required page resident न हो तो page fault हो सकता है। OS उसकी वैधता जाँचकर storage से page लाता है और mapping update करता है।

लाभ

- Processes को स्वतंत्र address spaces और protection।
- केवल needed pages RAM में रखने से memory utilization।
- Available RAM से बड़े address space वाले programs चलाना संभव।

सीमाएँ

Page faults expensive हैं; page tables और translation overhead होता है। बहुत अधिक paging होने पर thrashing से performance गिर सकती है। Virtual memory RAM की speed नहीं बढ़ाती।

5.1 Page Fault & Address Numerical

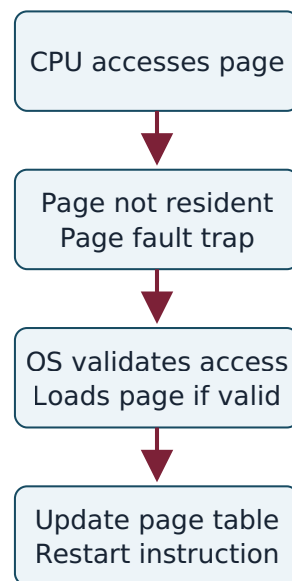


Figure 6. Valid non-resident page के fault की सरल handling।

Page Fault के चरण

OS access की वैधता जाँचता है। Invalid access पर error/termination हो सकता है। Valid page के लिए free frame लिया जाता है या victim page चुना जाता है। Modified victim को आवश्यक होने पर storage पर लिखा जाता है। Page load होने के बाद table update और instruction restart होती है।

Address Translation Numerical

Page size = 1024 bytes। Virtual address = 2500। Page number = $\text{floor}(2500/1024) = 2$; offset = $2500 \bmod 1024 = 452$ । यदि page 2 frame 5 में है, physical address = $5 \times 1024 + 452 = \mathbf{5572}$ ।

Thrashing

जब system useful execution की तुलना में pages लाने/निकालने में बहुत अधिक समय लगाता है, उसे thrashing कहते हैं। कारणों में insufficient frames तथा working sets के मुकाबले बहुत अधिक active processes शामिल हो सकते हैं।

Cache vs Virtual memory: Cache access speed बढ़ाने के लिए copies रखती है। Virtual memory address-space management, protection और demand loading की सुविधा देती है। दोनों locality से लाभ लेते हैं।

6. Direct Access Storage Devices (DASD)

DASD ऐसे storage devices हैं जिनमें किसी block तक पहुँचने के लिए सभी पिछले records को क्रम से पढ़ना आवश्यक नहीं होता। Magnetic disks इसके पारंपरिक उदाहरण हैं। SSD भी direct/random block access देते हैं, लेकिन उनमें mechanical disk movement नहीं होता।

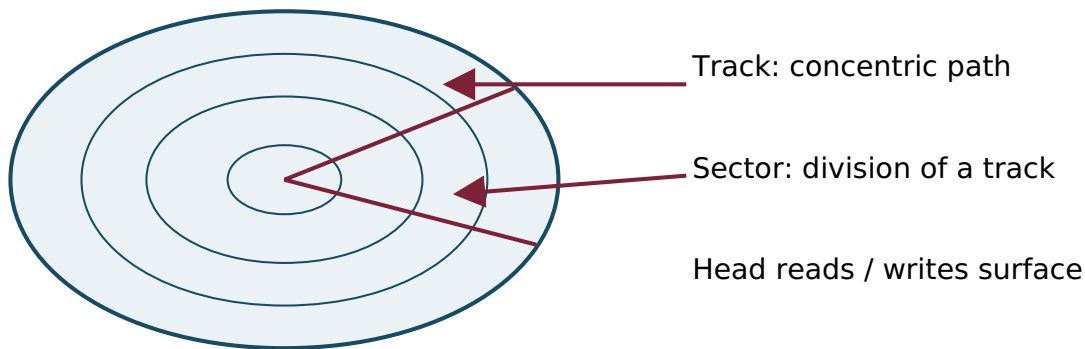


Figure 7. Magnetic disk की conceptual surface: tracks और sector division।

Magnetic Disk Components

- **Platter:** Magnetic coating वाली circular disk।
- **Track:** Surface पर concentric circular recording path।
- **Sector:** Track का storage division।
- **Read/write head:** Surface से data पढ़ता/लिखता है।
- **Cylinder:** Multiple surfaces पर समान radius के tracks का समूह।
- **Controller:** Requests, transfers और device operations manage करता है।

Disk Access Time

Seek time: Head को required track तक ले जाने का समय। **Rotational latency:** Required sector के head तक आने की प्रतीक्षा। **Transfer time:** Actual data transfer का समय।

Simplified total = seek time + rotational latency + transfer time। Queue और controller overhead अलग से हो सकते हैं।

Numerical: 8 ms + 4 ms + 1 ms = **13 ms**। SSD पर mechanical seek और rotational latency लागू नहीं होती।

6.1 Access Methods & Storage Comparison

Method	Explanation / Example
Sequential	पहले के data से गुजरकर आगे पहुँचना; magnetic tape।
Direct	Block का address देकर access; magnetic disk।
Random	Location को address से access, access time लगभग समान माना जाता है; RAM।
Associative	Address के बजाय content/key match से lookup; associative memory।

DASD के लाभ

- Records के लिए direct block access।
- Files/databases में convenient updates।
- Large non-volatile storage।
- Sequential medium की तुलना में arbitrary records तक जल्दी पहुँचना।

सीमाएँ

HDD में mechanical delay, wear और shock sensitivity हो सकती है। SSD में flash endurance तथा controller behavior महत्वपूर्ण हैं। किसी भी device पर backup आवश्यक है।

HDD	SSD
Magnetic rotating media	Flash-based, no moving parts
Seek और rotational delay	Mechanical seek/rotation नहीं
अक्सर lower cost per capacity	अक्सर lower latency
Shock और mechanical wear का प्रभाव	Write endurance एवं controller factors

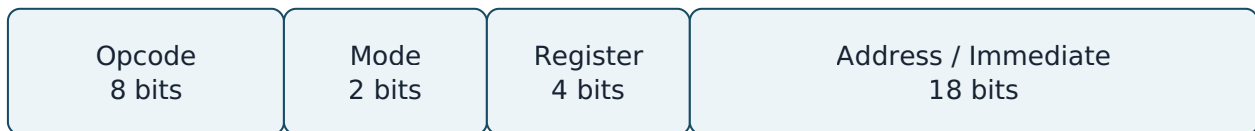
Rotational Latency Numerical

7200 RPM disk: एक rotation का समय = $60/7200$ second ≈ 8.33 ms। Uniform arrival मानने पर average rotational latency लगभग half rotation = **4.17 ms**।

10 अंक में: Definition, disk diagram, components, access-time formula, numerical तथा advantages/limitations लिखें।

7. Instruction Format

Instruction format machine instruction के bits की layout है। यह बताती है कि operation, operands और addressing information कहाँ encoded हैं। Actual formats processor architecture पर निर्भर होते हैं।



Illustrative 32-bit format: $8 + 2 + 4 + 18 = 32$ bits

Figure 8. उदाहरण के लिए बनाया गया instruction format; किसी विशेष CPU का standard नहीं।

Main Fields

- **Opcode:** कौन-सा operation करना है; जैसे ADD, LOAD या STORE।
- **Operand / address fields:** Data कहाँ है या किस operand पर operation होना है।
- **Register field:** प्रयुक्त register की पहचान।
- **Addressing-mode field:** Operand/address interpret करने की विधि।
- **Immediate field:** Instruction में encoded constant, यदि format अनुमति दे।

Addressing Examples

Immediate: Operand instruction में constant; जैसे ADD R1, #5। **Register:** Operand register में।

Direct: Address field memory location दर्शाता है। **Indirect:** Address किसी अन्य location/register में मिलता है। Exact assembly syntax architecture-specific है।

Fixed vs Variable Length

Fixed-length instructions का size समान होता है; decoding नियमित हो सकती है। Variable-length formats अलग instructions के लिए अलग size देते हैं; compact encoding संभव लेकिन decoding जटिल हो सकती है।

Numerical

8-bit opcode field अधिकतम $2^8 = 256$ distinct patterns देता है। 4-bit register field अधिकतम 16 register identifiers encode कर सकता है। सभी patterns actual ISA में उपयोग हों, यह आवश्यक नहीं।

7.1 Types of Instruction Formats

Address-count classification में explicit operand/address fields की संख्या देखी जाती है। Registers भी operands हो सकते हैं; प्रत्येक field को memory address समझना आवश्यक नहीं।

Format	Illustrative instruction	Meaning
Three-address	ADD R1, R2, R3	$R1 \leftarrow R2 + R3$
Two-address	ADD R1, R2	$R1 \leftarrow R1 + R2$
One-address	ADD X	Accumulator $\leftarrow$ Accumulator + M[X]
Zero-address	ADD	Stack के top operands जोड़ना

Three-address

Destination और दोनों source operands स्पष्ट होते हैं। कम instructions में expression बन सकता है, लेकिन operand fields अधिक bits ले सकते हैं।

Two-address

एक operand destination भी होता है। Encoding compact हो सकती है, लेकिन original operand overwrite होने से extra copy की आवश्यकता हो सकती है।

One-address

Accumulator implicit operand होता है। LOAD A; ADD B; STORE X द्वारा $X = A + B$ निकाला जा सकता है।

Zero-address

Stack machine में arithmetic operands implicit stack top पर रहते हैं। PUSH A; PUSH B; ADD के बाद result stack पर रहता है। Zero-address का अर्थ पूरी instruction set में कभी operand field न होना नहीं है।

Expression Example: $X = (A + B) \times C$

Three-address: ADD R1, A, B
MUL X, R1, C

Accumulator: LOAD A
ADD B
MUL C
STORE X

Stack: PUSH A
PUSH B
ADD
PUSH C
MUL
POP X

ये teaching examples हैं; वास्तविक machine में memory/register operand restrictions अलग हो सकती हैं।

8. Program and Data Memory

Program memory instructions store करती है। **Data memory** variables, input, intermediate values और results store करती है। ये logical roles हैं; जरूरी नहीं कि दोनों अलग physical chips हों।

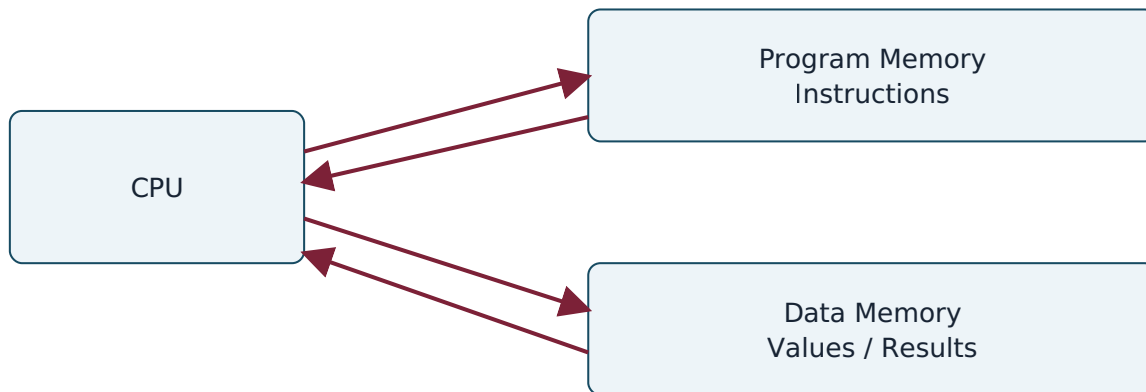


Figure 9. Instructions और data के logical roles; separate physical memories architecture पर निर्भर हैं।

Program Memory

CPU instructions fetch करके decode और execute करता है। Embedded systems में program flash/ROM में हो सकता है। General-purpose computers में executable storage से RAM में load होता है; code pages सामान्यतः suitable protection के साथ रखे जाते हैं।

Data Memory

Variables, arrays, stack और dynamically allocated objects रख सकती है। Runtime values बदलते हैं, इसलिए working data सामान्यतः writable RAM में रहता है। Persistent data storage पर save किया जाता है।

Program memory	Data memory
Instructions	Operands और results
Fetch/decode/execute में उपयोग	Read/write computations में उपयोग
Code / firmware	Variables, stack, heap
अक्सर executable और protected	अक्सर writable; permissions लागू

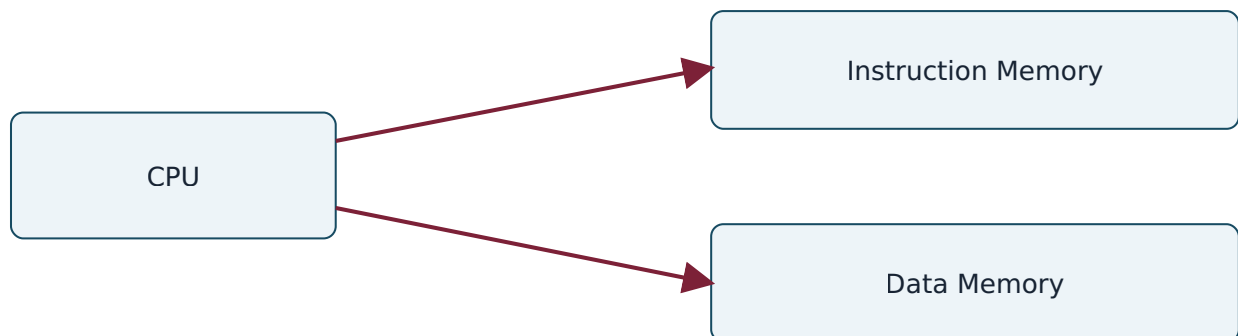
Example

C statement $sum = a + b$; के machine instructions program region में होते हैं। a , b और sum के runtime values data storage में होते हैं। Compiler optimization के कारण कुछ values registers में भी रह सकती हैं।

8.1 Von Neumann & Harvard Architecture



Von Neumann: shared logical memory / path



Harvard: separate instruction and data paths

Figure 10. दो architectures का simplified comparison।

Von Neumann	Harvard
Instructions/data का shared address space	Separate instruction/data memories या address spaces
Shared access path bottleneck हो सकता है	Independent paths simultaneous access दे सकते हैं
Flexible common-memory organization	Different widths/technologies संभव
Conceptually simpler shared system	Organization अधिक complex हो सकती है

Modified Harvard

कई modern processors separate instruction/data caches रखते हैं लेकिन lower levels पर unified memory उपयोग करते हैं। इसलिए किसी modern computer को केवल basic diagram से पूरी तरह describe नहीं किया जा सकता।

Exam tip: दोनों diagrams बनाकर memory arrangement, buses, simultaneous access तथा complexity के आधार पर differences लिखें।

9. Memory Management Unit (MMU)

MMU hardware component है जो CPU-generated virtual/logical addresses को physical addresses में translate करने और access protection लागू करने में सहायता करता है। यह processor के अंदर integrated हो सकती है।

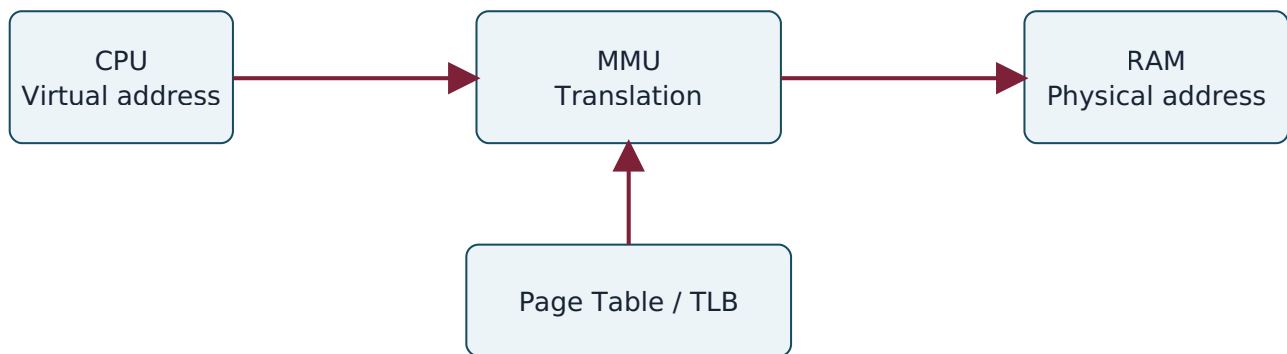


Figure 11. MMU translation और mapping information का उपयोग।

मुख्य कार्य

- Virtual-to-physical address translation।
- Page-table entries और permissions का उपयोग।
- Read/write/execute तथा privilege restrictions की जाँच।
- Translation cache अर्थात TLB के साथ काम।
- Invalid/non-resident mapping पर appropriate fault signal करना।

OS और MMU की भूमिका

OS page tables बनाता/update करता है, frames allocate करता है और page faults handle करता है। MMU hardware memory access के समय configured translation/protection लागू करता है। MMU स्वयं disk से missing page load नहीं करता।

TLB — Translation Lookaside Buffer

TLB recently used address translations की छोटी तेज cache है। TLB hit पर mapping जल्दी मिलती है। TLB miss पर page table walk होता है—hardware या software द्वारा, architecture पर निर्भर।

महत्वपूर्ण: TLB miss हमेशा page fault नहीं है। Mapping page table में valid/resident मिल जाए तो page fault के बिना access पूरा हो सकता है।

9.1 MMU Translation & Protection

Paging में Translation

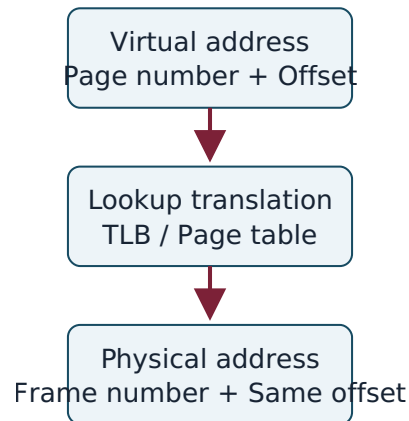


Figure 12. Paging में page number translate होता है; offset अपरिवर्तित रहता है।

Worked Example

Page size 4 KiB = 4096 bytes। Virtual address 13,000। Page number = $\text{floor}(13000/4096) = 3$; offset = 712। यदि page 3 frame 9 में map है, physical address = $9 \times 4096 + 712 = 37,576$ ।

Offset Bits

4 KiB page size = 2^{12} bytes; इसलिए byte-addressable system में offset 12 bits का है। 32-bit virtual address में virtual page number के लिए 20 bits बचते हैं।

Base and Limit: सरल Protection Model

एक simple relocation model में $\text{physical address} = \text{base} + \text{logical address}$, बशर्ते logical address permitted limit के अंदर हो। Example: base 5000 और valid logical address 120 → physical 5120। Modern paging इससे अधिक विस्तृत mechanism है।

MMU का महत्व

Process isolation, controlled sharing, relocation और virtual memory support। MMU translation overhead ला सकती है, जिसे TLB कम करता है। OS policies और hardware support मिलकर memory protection देते हैं।

10. Revision & Examination Questions

Quick Revision

- SRAM और DRAM दोनों सामान्यतः volatile हैं।
- ROM family non-volatile है; EEPROM/flash rewrite की जा सकती हैं।
- Cache copies रखती है; hit ratio = hits / accesses।
- AMAT में miss penalty अतिरिक्त समय है।
- Virtual memory address-space management देती है; RAM तेज नहीं बनाती।
- Page और frame size paging में समान होते हैं।
- Page fault और TLB miss अलग हैं।
- DASD direct block access देता है; tape sequential है।
- Instruction format में opcode और operand-related fields होते हैं।
- Program/data memory roles अलग हैं; physical separation architecture पर निर्भर।
- MMU translates/protects; OS page allocation/fault handling करता है।

5 Marks Questions

- Memory hierarchy diagram सहित समझाइए।
- SRAM-DRAM तथा RAM-ROM में अंतर लिखिए।
- Cache hit, miss और locality समझाइए।
- Virtual memory और page fault समझाइए।
- DASD और disk access time लिखिए।
- Instruction format के fields समझाइए।
- Program/data memory में अंतर लिखिए।
- MMU तथा TLB की भूमिका लिखिए।

10 Marks Questions

- Hierarchy के levels, characteristics और locality समझाइए।
- Primary/secondary memory तथा RAM/ROM types विस्तार से लिखिए।
- Cache working, mapping और AMAT numerical लिखिए।
- Virtual memory, demand paging और address translation समझाइए।
- DASD disk diagram, components तथा timing numerical लिखिए।
- Instruction formats और address-count types उदाहरण सहित समझाइए।
- Von Neumann/Harvard diagrams तथा program/data memory समझाइए।
- MMU working, protection, TLB और translation numerical लिखिए।