DEPARTMENT OF COMPUTER SCIENCE AND ENGG.

UNIVERSITY INSTITUTE OF ENGINEERING

CHANDIGARH UNIVERSITY, MOHALI

CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

COMPUTER ORGANIZATION & ARCHITECTURE

CST-203/ITT-203

Session : July- Dec 2018

Isha Sharma E2892
Asst.prof
Chandigarh University

COURSE INTRODUCTION

Programme Name: BE CSE /IT

Duration: 4 YEARS

Subject Name: Computer Organization and Architecture

Subject Code: CST-203/ITT-203

Total Hours: 45 (LTP: 3:0:0)

Credits: 3

RELEVANCE OF COURSE

- To familiarize students with the architecture of a processor.
- To have a good understanding of various functional units of computer.
- To understand the design of a basic computer system.

SYALLABUS -UNIT 2

UNIT-II **[15h]**

Design of control unit - Hardwired control unit, Micro-Programmed control unit and comparative study.

Memory organization-Memory hierarchy, Cache Memory Associative Memory, Cache memory with associative memory, Virtual Memory: Paging, Segmentation.

Input output organization –Asynchronous Data transfer: Source Initiated, Destination Initiated, Handshaking, Programmed I/O, Interrupts DMA, and IOP

LEARNING OBJCETIVES :UNIT2

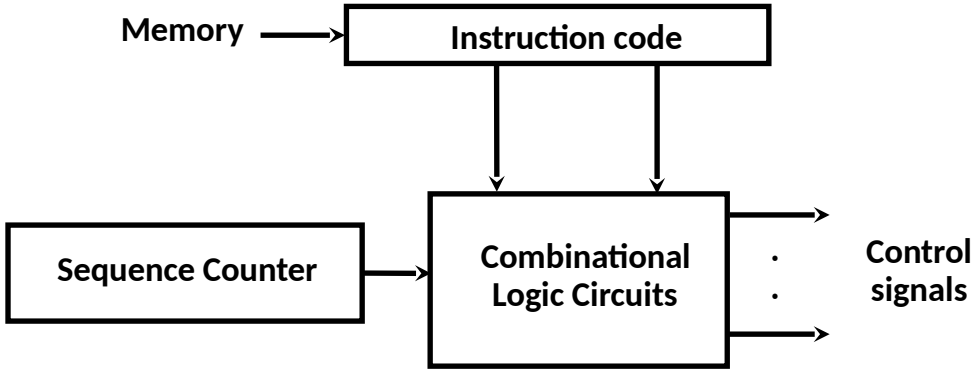
- To understand input/output mechanisms
- To understand the various parts of a system memory hierarchy.
- To understand various CPU mapping techniques with numerical problems

PEROQUISITES

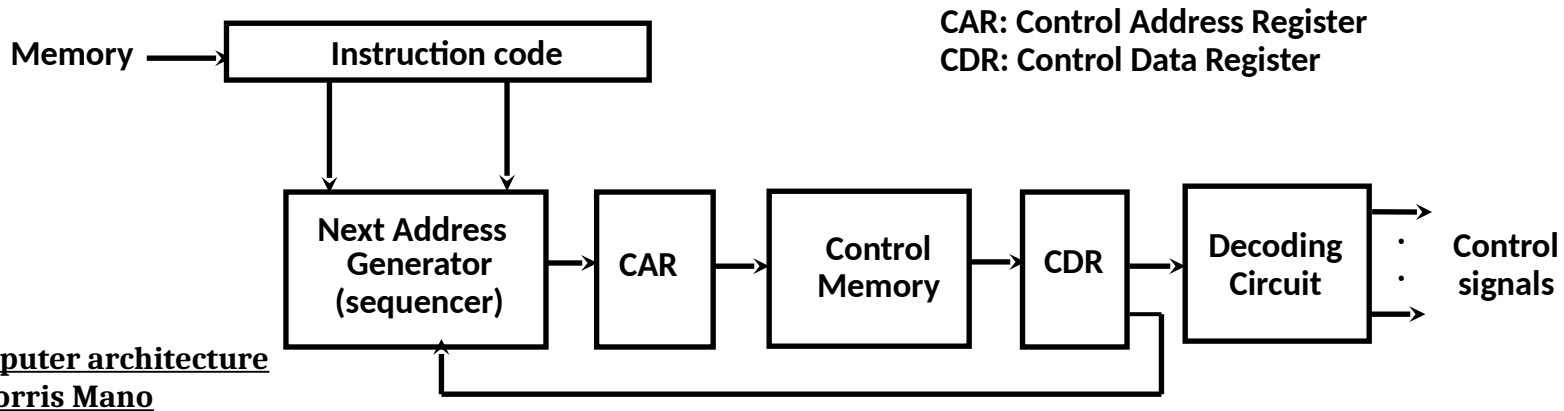
- Functional Units of the Computer
  - CPU, ALU, Memory etc.
- Memory Units and its types
  - main memory
  - secondary memory
  - Cache
  - Virtual
- Multiprogramming environment

CONTROL UNIT: Implementation

- Hardwired



- Microprogrammed



SOURCE: Computer architecture by Morris Mano

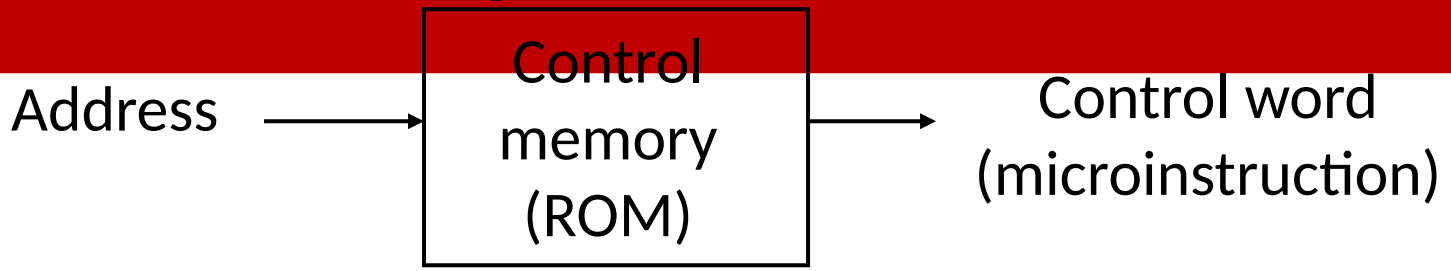
MICRO PROGRAMMED CONTROL UNIT

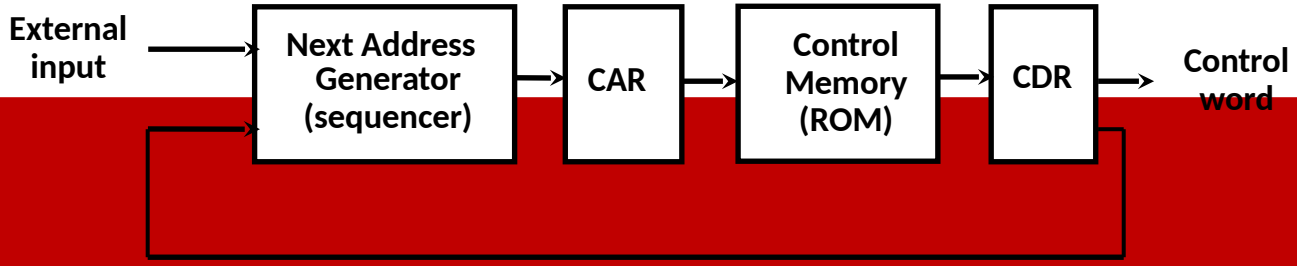
- Control signals
  - Group of bits used to select paths in multiplexers, decoders, arithmetic logic units
- Control variables
  - Binary variables specify microoperations
  - Certain microoperations initiated while others idle
- Control word
  - String of 1's and 0's represent control variables

- Control memory
  - Memory contains control words
- Microinstructions
  - Control words stored in control memory
  - Specify control signals for execution of micro operations
- Micro program
  - Sequence of microinstructions

Control Memory

- Read-only memory (ROM)
- Content of word in ROM at given address specifies microinstruction
- Each computer instruction initiates series of microinstructions (microprogram) in control memory
- These microinstructions generate microoperations to
  - Fetch instruction from main memory
  - Evaluate effective address
  - Execute operation specified by instruction
  - Return control to fetch phase for next instruction

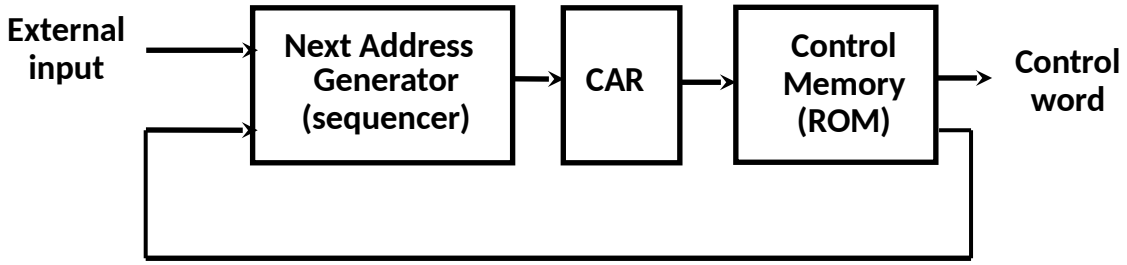




- Control memory
  - Contains microprograms (set of microinstructions)
  - Microinstruction contains
    - Bits initiate microoperations
    - Bits determine address of next microinstruction
- Control address register (CAR)
  - Specifies address of next microinstruction

- Next address generator (microprogram sequencer)
  - Determines address sequence for control memory
- Micro program sequencer functions
  - Increment CAR by one
  - Transfer external address into CAR
  - Load initial address into CAR to start control operations

- Control data register (CDR)- or pipeline register
  - Holds microinstruction read from control memory
  - Allows execution of micro operations specified by control word simultaneously with generation of next microinstruction
- Control unit can operate without CDR



Micro program Routines

- Routine
  - Group of microinstructions stored in control memory
- Each computer instruction has its own micro program routine to generate micro operations that execute the instruction

- Subroutine
  - Sequence of microinstructions used by other routines to accomplish particular task
- Example
  - Subroutine to generate effective address of operand for memory reference instruction
- Subroutine register (SBR)
  - Stores return address during subroutine call

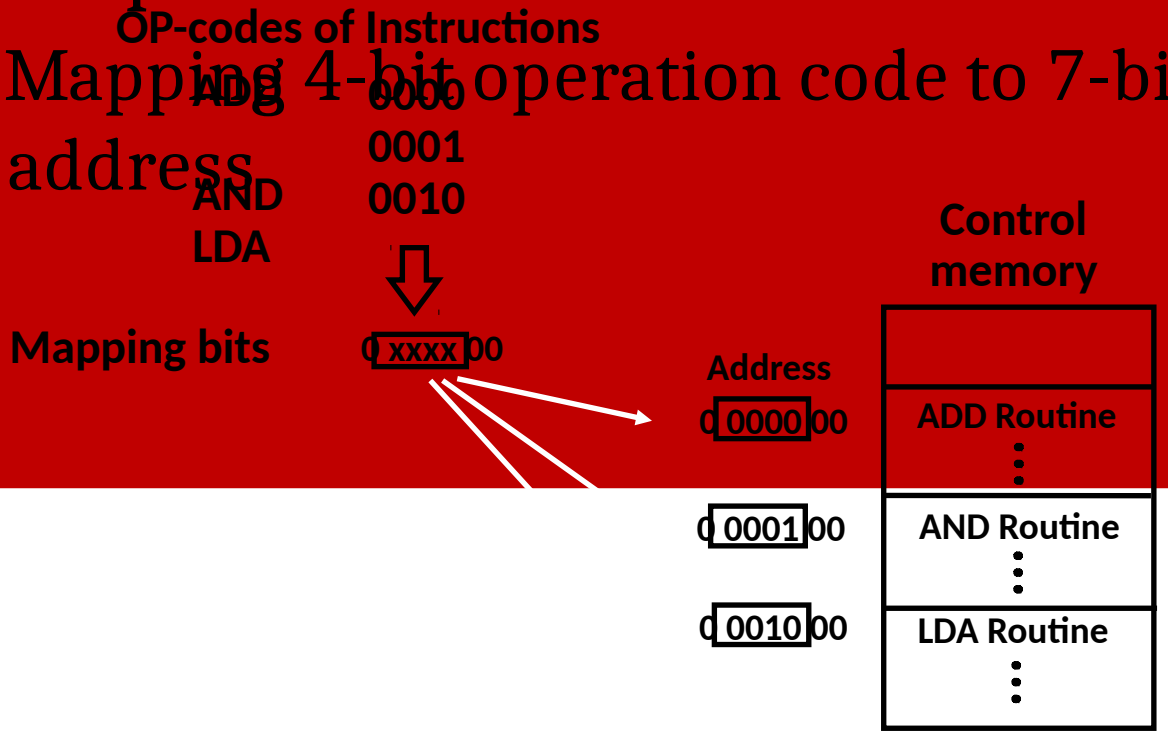
Conditional Branching

- Branching from one routine to another depends on status bit conditions
- Status bits provide parameter info such as
  - Carry-out of adder
  - Sign bit of number
  - Mode bits of instruction
- Info in status bits can be tested and actions initiated based on their conditions: 1 or 0
- Unconditional branch
  - Fix value of status bit to 1

Mapping of Instruction

- Each computer instruction has its own micro program routine stored in a given location of the control memory
- Mapping
  - Transformation from instruction code bits to address in control memory where routine is located

- Example
  - Mapping 4-bit operation code to 7-bit address



Address Sequencing

- Address sequencing capabilities required in control unit
  - Incrementing CAR
  - Unconditional or conditional branch, depending on status bit conditions
  - Mapping from bits of instruction to address for control memory
  - Facility for subroutine call and return

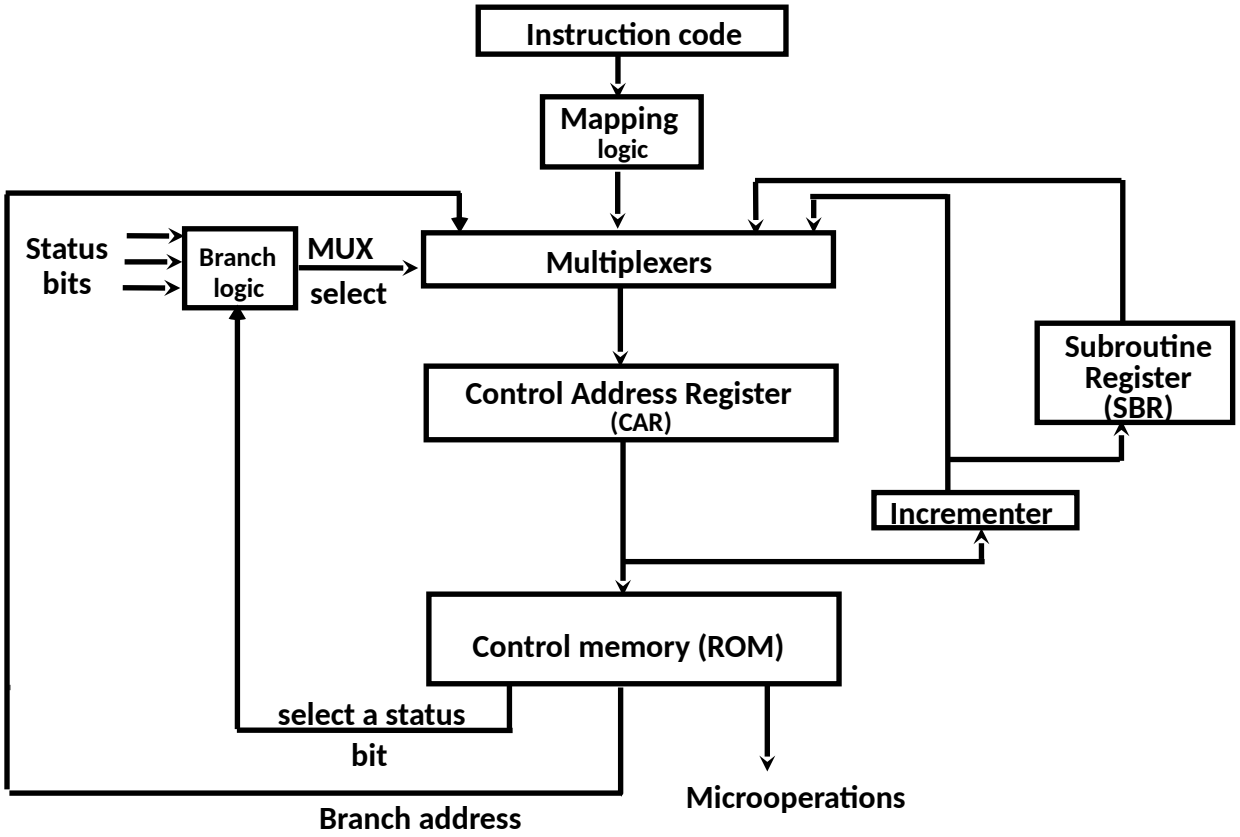
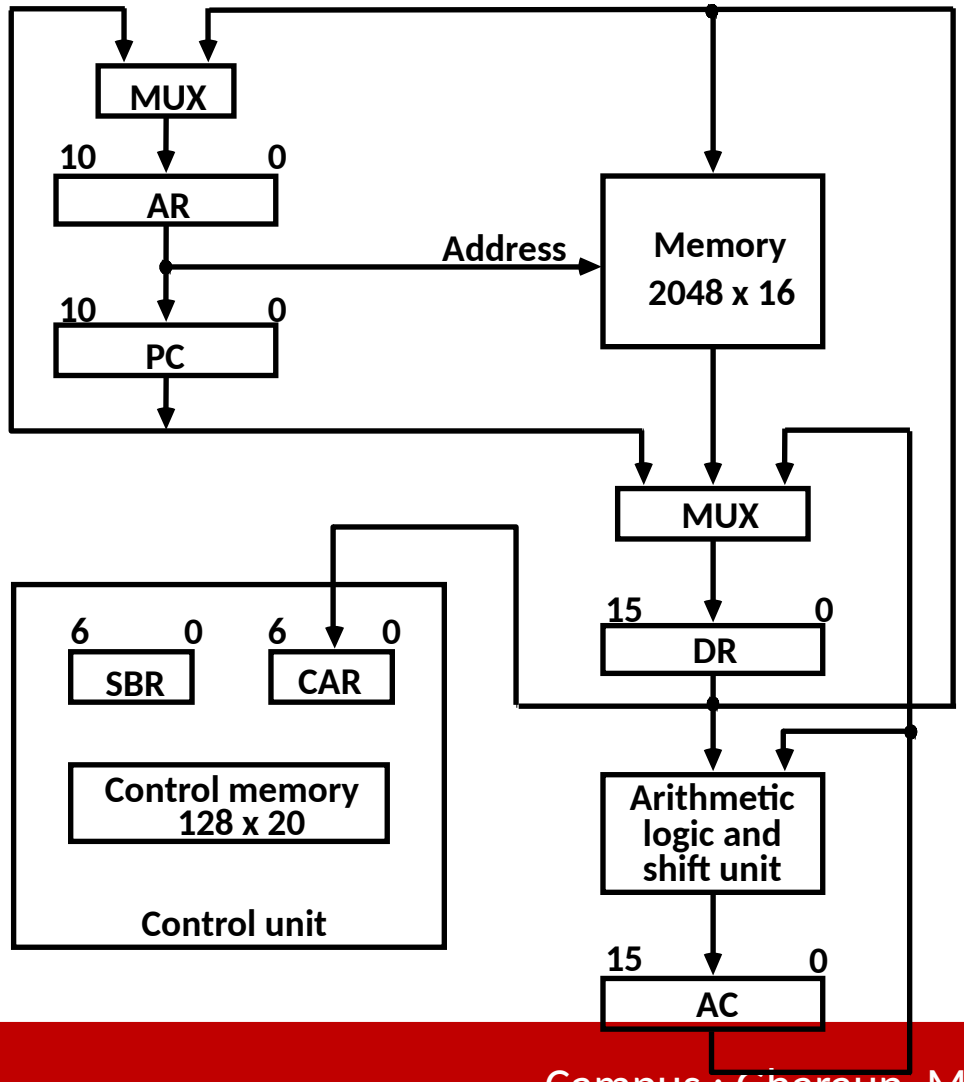


Diagram Address Sequencer

SOURCE: Computer architecture by Morris Mano

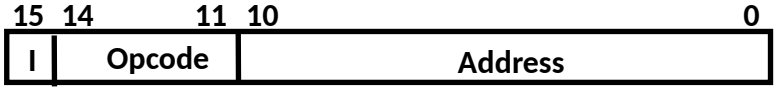
Micro program Example

Computer Configuration



SOURCE: Computer architecture by Morris Mano

Computer instruction format

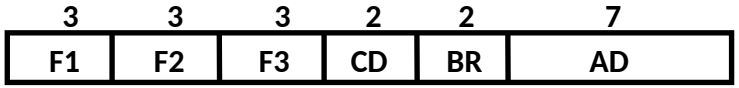


Four computer instructions

Symbol	OP-code	Description
ADD	0000	AC ← AC + M[EA]
BRANCH	0001	if (AC < 0) then (PC ← EA)
STORE	0010	M[EA] ← AC
EXCHANGE	0011	AC ← M[EA], M[EA] ← AC

EA is the effective address

Microinstruction Format



F1, F2, F3: Microoperation fields
CD: Condition for branching
BR: Branch field
AD: Address field

SOURCE: Computer architecture by Morris Mano

Microinstruction Fields

F1	Microoperation	Symbol
000	None	NOP
001	AC ← AC + DR	ADD
010	AC ← 0	CLRAC
011	AC ← AC + 1	INCAC
100	AC ← DR	DRTAC
101	AR ← DR(0-10)	DRTAR
110	AR ← PC	PCTAR
111	M[AR] ← DR	WRITE

F2	Microoperation	Symbol
000	None	NOP
001	AC ← AC - DR	SUB
010	AC ← AC $\vee$ DR	OR
011	AC ← AC $\wedge$ DR	AND
100	DR ← M[AR]	READ
101	DR ← AC	ACTDR
110	DR ← DR + 1	INCDR
111	DR(0-10) ← PC	PCTDR

F3	Microoperation	Symbol
000	None	NOP
001	AC ← AC ⊕ DR	XOR
010	AC ← AC'	COM
011	AC ← shl AC	SHL
100	AC ← shr AC	SHR
101	PC ← PC + 1	INCPC
110	PC ← AR	ARTPC
111	Reserved	

SOURCE: Computer architecture by Morris Mano

Microinstruction Fields

CD	Condition	Symbol	Comments
00	Always = 1	U	Unconditional branch
01	DR(15)	I	Indirect address bit
10	AC(15)	S	Sign bit of AC
11	AC = 0	Z	Zero value in AC

BR	Symbol	Function
00	JMP	CAR ← AD if condition = 1
		CAR ← CAR + 1 if condition = 0
01	CALL	CAR ← AD, SBR ← CAR + 1 if condition = 1
		CAR ← CAR + 1 if condition = 0
10	RET	CAR ← SBR (Return from subroutine)
11	MAP	CAR(2-5) ← DR(11-14), CAR(0,1,6) ← 0

SOURCE: Computer architecture by Morris Mano

Symbolic Microinstruction

- **Sample Format Label: Micro-ops CD BR AD**

- Label may be empty or may specify symbolic address terminated with colon

- Micro-ops consists of 1, 2, or 3 symbols separated by commas

- CD one of {U, I, S, Z}
  U: Unconditional Branch
  I: Indirect address bit
  S: Sign of AC
  Z: Zero value in AC

- BR one of {JMP, CALL, RET, MAP}

- AD one of {Symbolic address, NEXT, empty}

Fetch Routine

- Fetch routine
  - Read instruction from memory
  - Decode instruction and update PC

Microinstructions for fetch routine:

```
AR ← PC
DR ← M[AR], PC ← PC + 1
AR ← DR(0-10), CAR(2-5) ← DR(11-14), CAR(0,1,6) ← 0
```

Symbolic microprogram for fetch routine:

```
FETCH:  ORG 64
        PCTAR          U  JMP  NEXT
        READ, INCPC    U  JMP  NEXT
        DRTAR          U  MAP
```

Binary microporgram for fetch routine:

SOURCE: Computer architecture by Morris Mano

Binary address	F1	F2	F3	CD	BR	AD
1000000	110	000	000	00	00	1000001
1000001	000	100	101	00	00	1000010
1000010	101	000	000	00	11	0000000

Symbolic Micro program

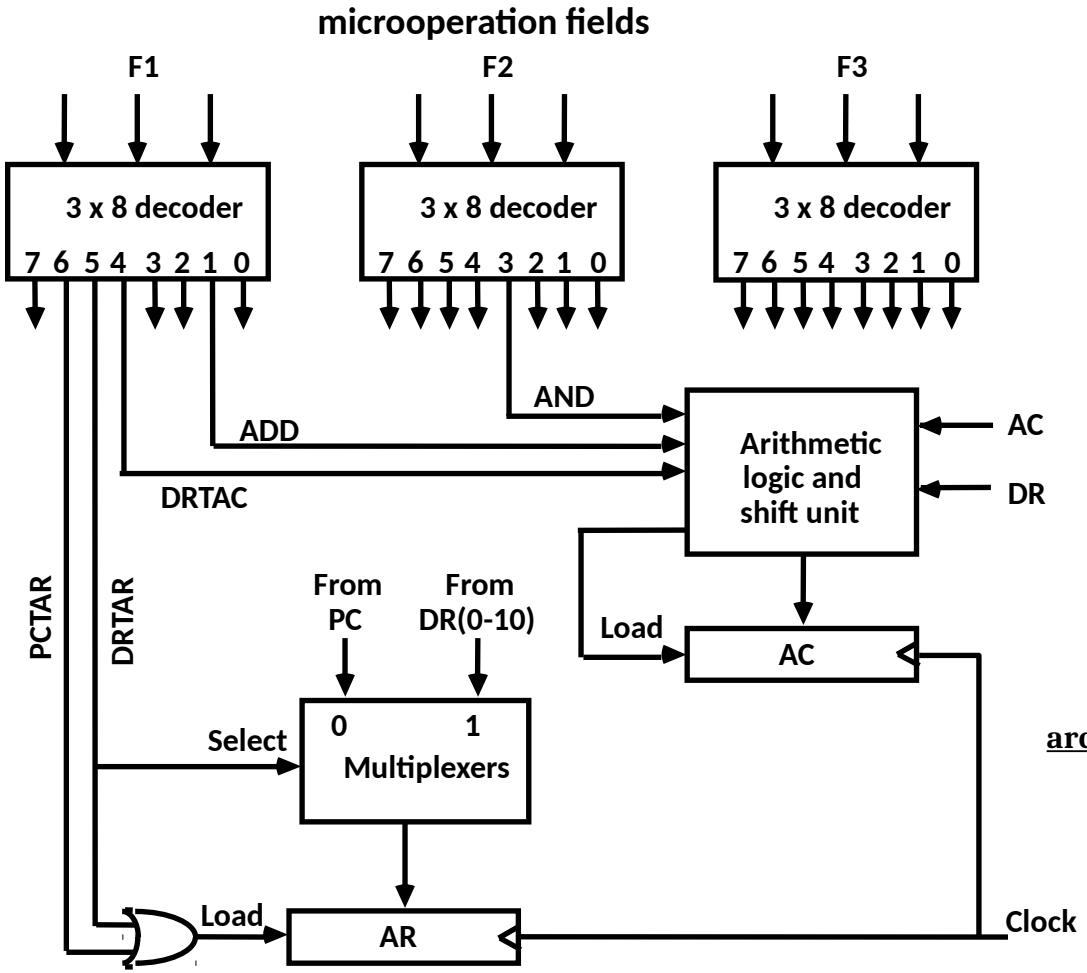
- Control memory: 128 20-bit words
- First 64 words: Routines for 16 machine instructions
- Last 64 words: Used for other purpose (e.g., fetch routine and other subroutines)
- Mapping: OP-code XXXX into 0XXXX00, first address for 16 routines are 0(0 0000 00), 4(0 0001 00), 8, 12, 16, 20, ..., 60

Partial Symbolic Microprogram

Label	Microops	CD	BR	AD
	ORG 0			
ADD:	NOP	I	CALL	INDRCT
	READ	U	JMP	NEXT
	ADD	U	JMP	FETCH
	ORG 4			
BRANCH:	NOP	S	JMP	OVER
	NOP	U	JMP	FETCH
OVER:	NOP	I	CALL	INDRCT
	ARTPC	U	JMP	FETCH
	ORG 8			
STORE:	NOP	I	CALL	INDRCT
	ACTDR	U	JMP	NEXT
	WRITE	U	JMP	FETCH
	ORG 12			
EXCHANGE:	NOP	I	CALL	INDRCT
	READ	U	JMP	NEXT
	ACTDR, DRTAC	U	JMP	NEXT
	WRITE	U	JMP	FETCH
	ORG 64			
FETCH:	PCTAR	U	JMP	NEXT
	READ, INCPC	U	JMP	NEXT
	DRTAR	U	MAP	
INDRCT:	READ	U	JMP	NEXT
	DRTAR	U	RET	

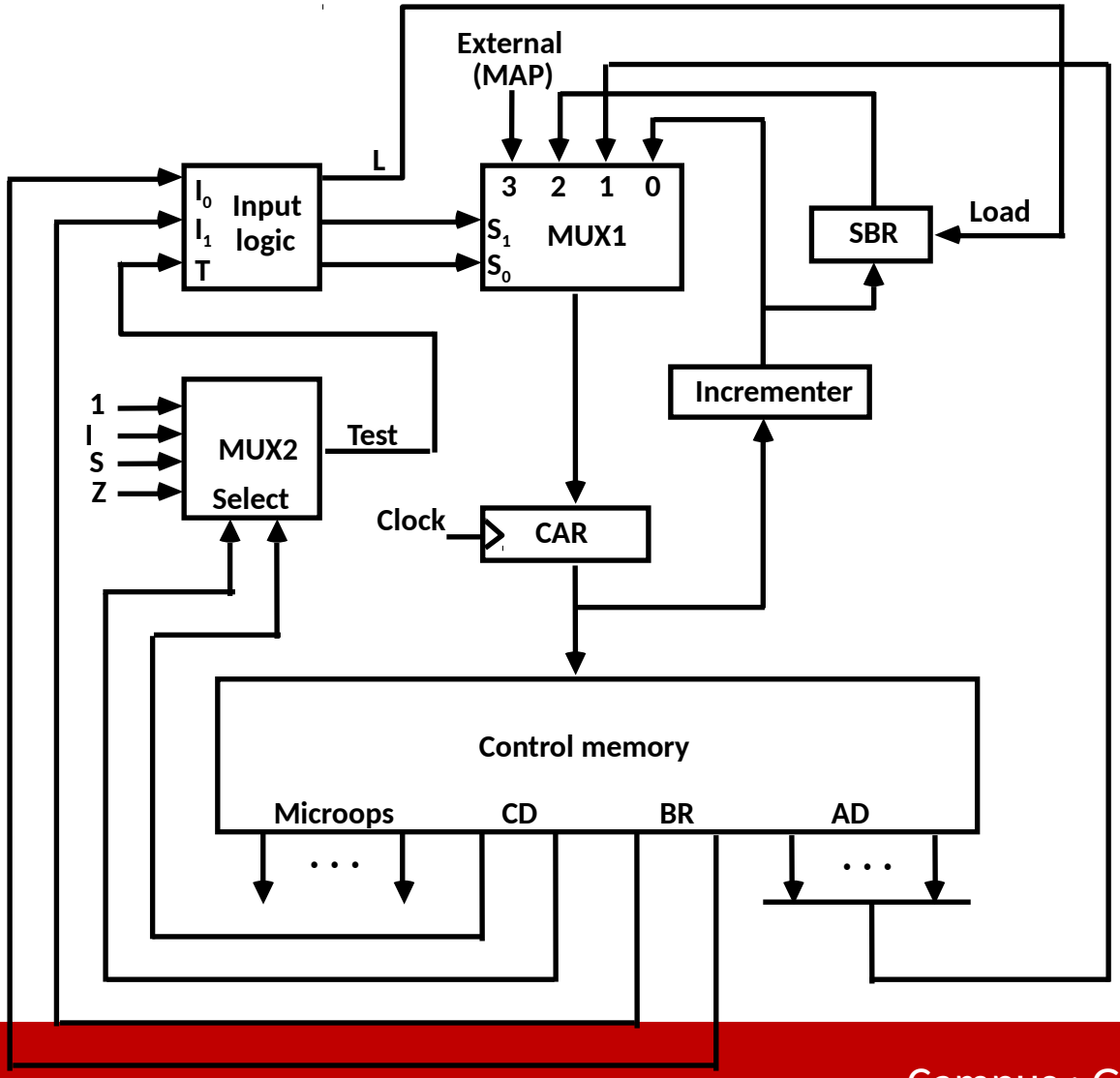
SOURCE: Computer architecture by Morris Mano

DESIGN OF CONTROL UNIT



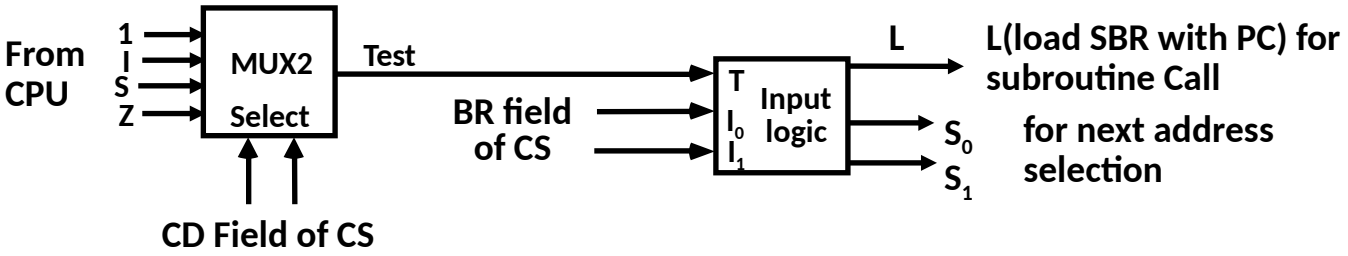
SOURCE: Computer architecture by Patterson

Microprogram Sequencer



SOURCE: Computer architecture by Patterson

Input Logic for Microprogram Sequencer



Input Logic

I1I0T	Meaning	Source of Address	S_1S_0	L
000	In-Line	CAR+1	00	0
001	JMP	CS(AD)	01	0
010	In-Line	CAR+1	00	0
011	CALL	CS(AD) and SBR <- CAR+1	01	1
10x	RET	SBR	10	0
11x	MAP	DR(11-14)	11	0

$S1 = I1$

$S_0 = I_0I_1 + I1'T$

$L = I_1'I0T$

SOURCE: Computer architecture by Patterson

Problems with hardwired control unit

- Sequencing & micro-operation logic gets complex
- Difficult to design, prototype, and test
- Resultant design is inflexible, and difficult to build upon (Pipeline, multiple computation units, etc.)
- Adding new instructions requires major design and adds complexity quickly

Advantages and disadvantages of microprogramming

Advantage:

- Simplifies design of control unit
  - Cheaper
  - Less error-prone
  - Easier to modify

Disadvantage:

- Slower

COMPARTIVE STUDY

Micro programmed control

- Micro programmed control is a control mechanism to generate control signals by using a memory called control storage (CS), which contains the control signals. Although micro programmed control seems to be advantageous to CISC machines, since CISC requires systematic development of sophisticated control signals, there is no intrinsic difference between these 2 control mechanisms.

Hard-wired control

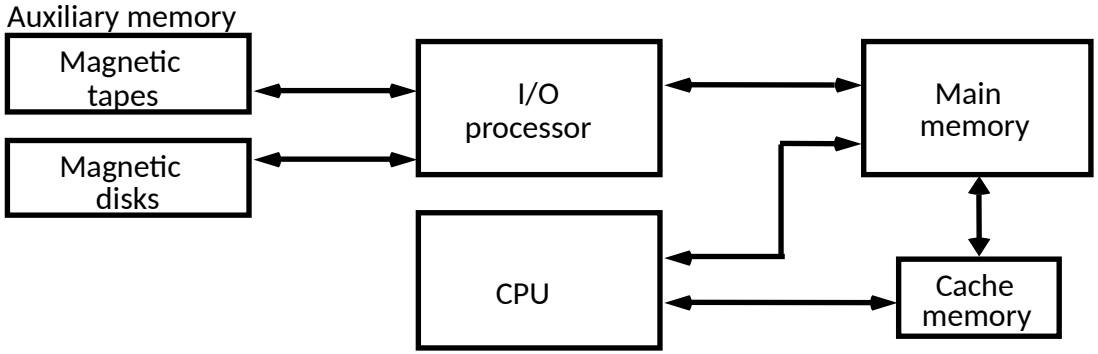
- Hardwired control is a control mechanism to generate control signals by using appropriate finite state machine (FSM). The pair of "microinstruction-register" and "control storage address register" can be regarded as a "state register" for the hardwired control. Note that the control storage can be regarded as a kind of combinational logic circuit. We can assign any 0, 1 values to each output corresponding to each address, which can be regarded as the input for a combinational logic circuit. This is a truth table.

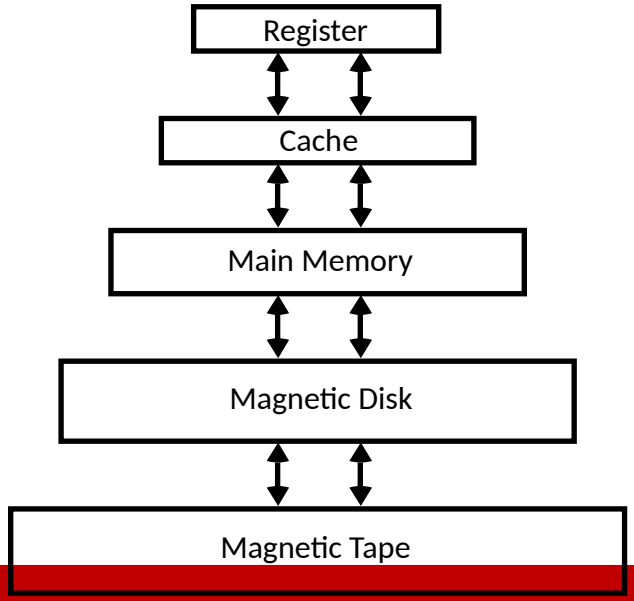
MEMORY ORGANIZATION

- Memory Hierarchy
- Main Memory
- Auxiliary Memory
- Associative Memory
- Cache Memory
- Virtual Memory
- Memory Management Hardware

MEMORY HIERARCHY

Memory Hierarchy is to obtain the highest possible access speed while minimizing the total cost of the memory system



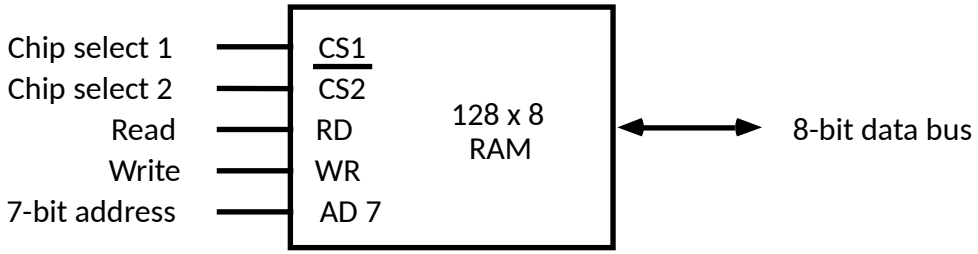


SOURCE: Computer architecture by M. Mano

MAIN MEMORY

RAM and ROM Chips

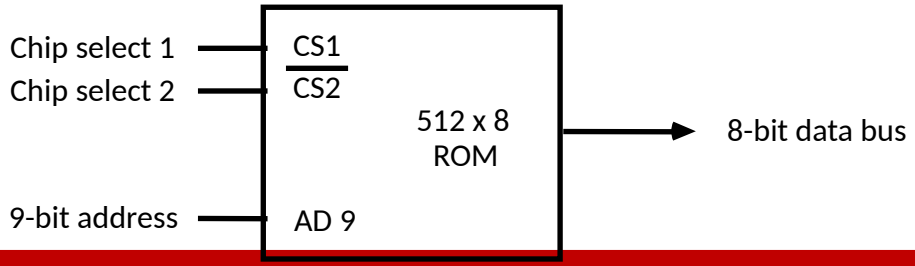
Typical RAM chip



CS1	$\overline{CS2}$	RD	WR	Memory function	State of data bus
0	0	x	x	Inhibit	High-impedence
0	1	x	x	Inhibit	High-impedence
1	0	0	0	Inhibit	High-impedence
1	0	0	1	Write	Input data to RAM
1	0	1	x	Read	Output data from RAM
1	1	x	x	Inhibit	High-impedence

Typical ROM chip

SOURCE: Computer architecture by M. Mano



MEMORY ADDRESS MAP

Address space assignment to each memory chip

Example: 512 bytes RAM and 512 bytes ROM

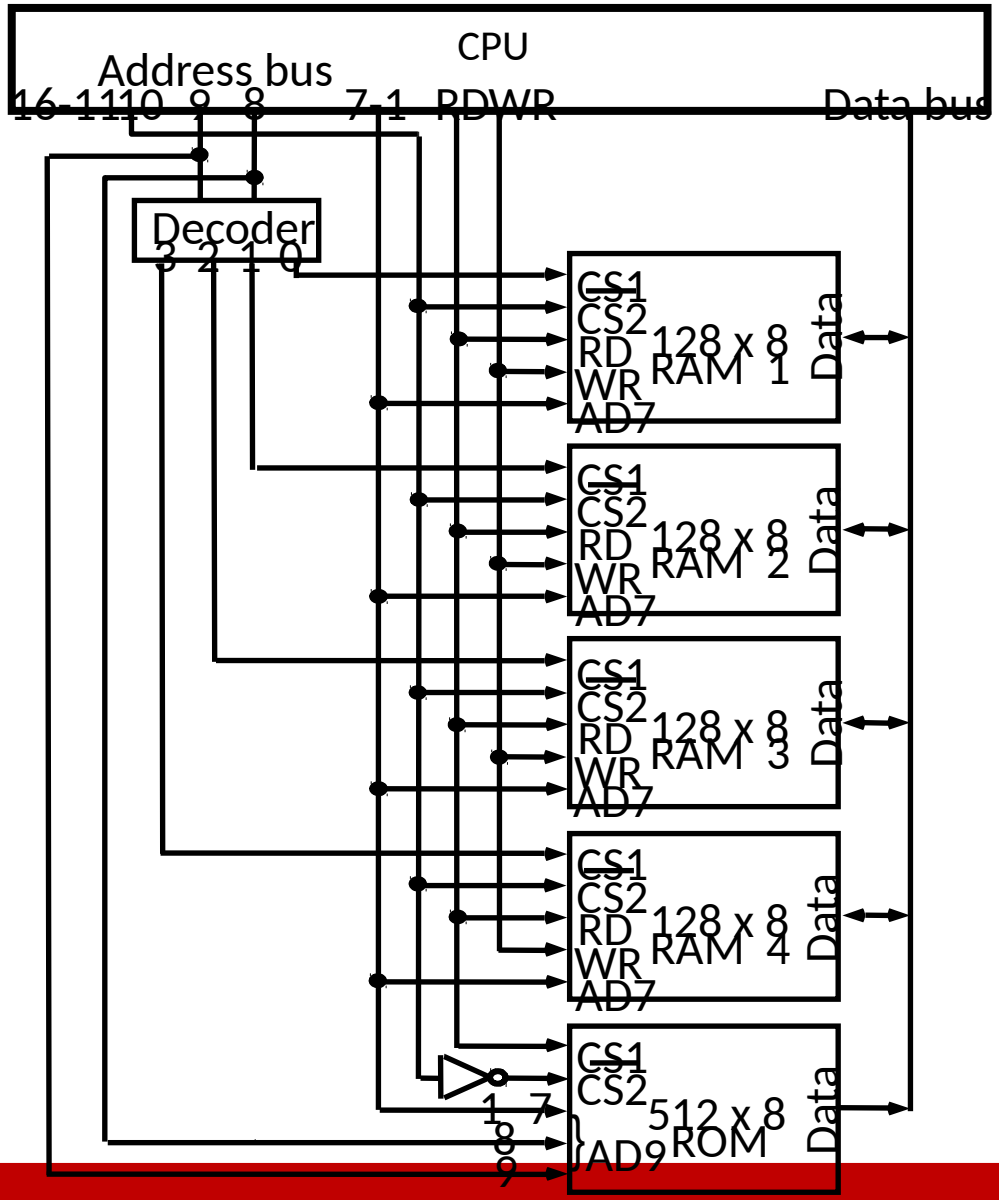
Component	Hexa address	Address bus 10 9 8 7 6 5 4 3 2 1
RAM 1	0000 - 007F	0 0 0 x x x x x x x
RAM 2	0080 - 00FF	0 0 1 x x x x x x x
RAM 3	0100 - 017F	0 1 0 x x x x x x x
RAM 4	0180 - 01FF	0 1 1 x x x x x x x
ROM	0200 - 03FF	1 x x x x x x x x x

SOURCE: Computer architecture by Patterson

Memory Connection to CPU

- RAM and ROM chips are connected to a CPU through the data and address buses

- The low-order lines in the address bus select the byte within the chips and other lines in the address bus select a particular chip through its chip select inputs



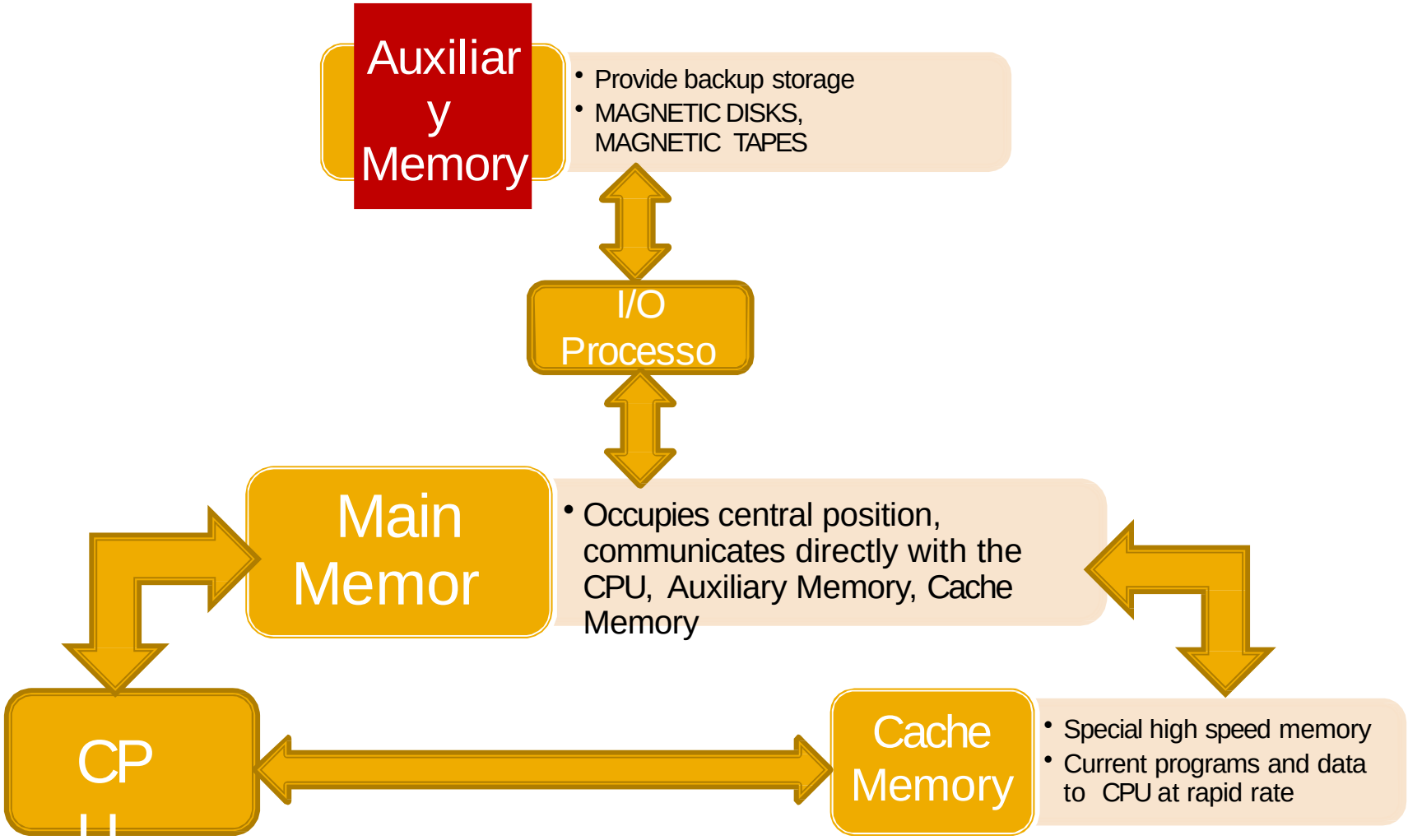
CONNECTION OF MEMORY TO CPU

SOURCE: Computer architecture by Morris Mano

MEMORY ORGANIZATION

- Memory Hierarchy
- Main Memory
- Auxiliary Memory
- Associative
- Memory
- Cache Memory
- Virtual Memory

MEMORY HIERACHY



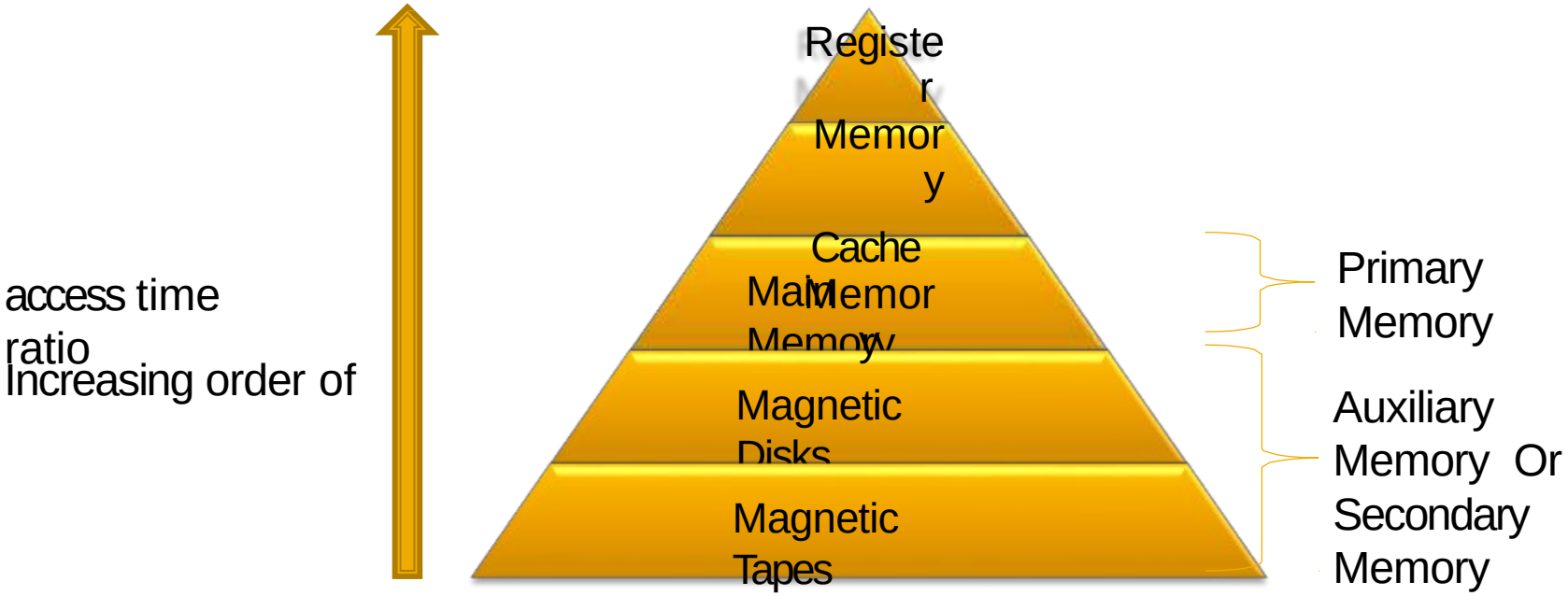
CPU logic is usually faster than main memory access time, with the result that processing speed is limited primarily by the speed of main memory.

The cache is used for storing segments of programs currently being executed in the CPU and temporary data frequently needed in the present calculations.

The typical access time ratio between cache and main memory is about 1 to 7~10 .

Auxiliary memory access time is usually 1000 times that of main memory.

The memory hierarchy system consists of all storage devices employed in a computer system from the slow by high- capacity auxiliary memory to a relatively faster main memory, to an even smaller and faster cache memory.



ACCESS METHODS

Each memory is a collection of various memory location. Accessing the memory means finding and reaching desired location and than reading information from memory location. The information from locations can be accessed as follows:

1. Random access
2. Sequential access
3. Direct access

Random Access: It is the access mode where each memory location has a unique address. Using these unique addresses each memory location can be addressed independently in any order in equal amount of time. Generally, main memories are random access memories(RAM).

Sequential Access: If storage locations can be accessed only in a certain predetermined sequence, the access method is known as serial or sequential access.
Opposite of RAM: Serial Access Memory (SAM). SAM works very well for memory buffers, where the data is normally stored in the order in which it will be used (a good example is the texture buffer memory on a video card , magnetic tapes, etc.).

Direct Access: In this access information is stored on tracks and each track has a separate read/write head. This features makes it a semi random mode which is generally used in magnetic disks.

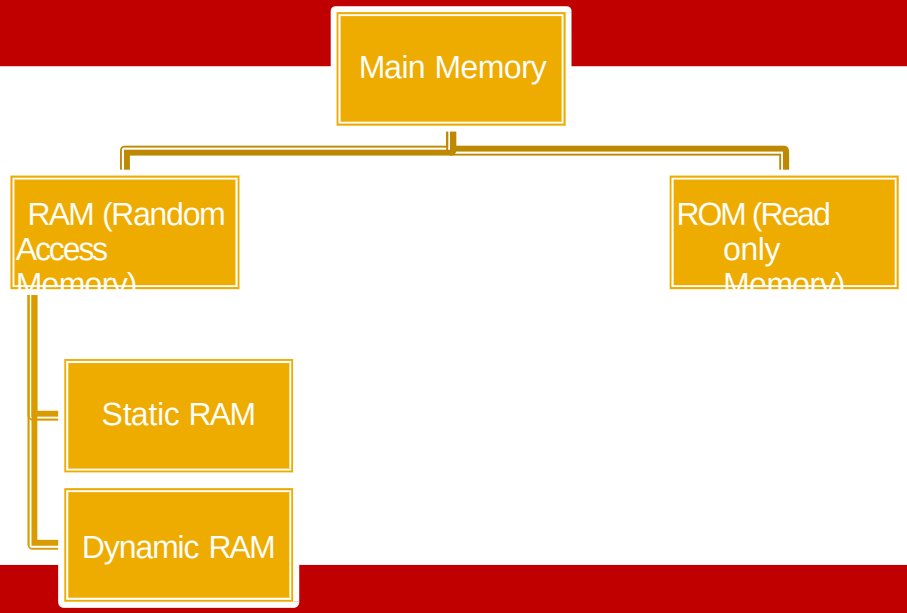
MAIN MEMORY

Most of the main memory in a general purpose computer is made up of **RAM** integrated circuits chips, but a portion of the memory may be constructed with **ROM** chips.

RAM– Random Access memory

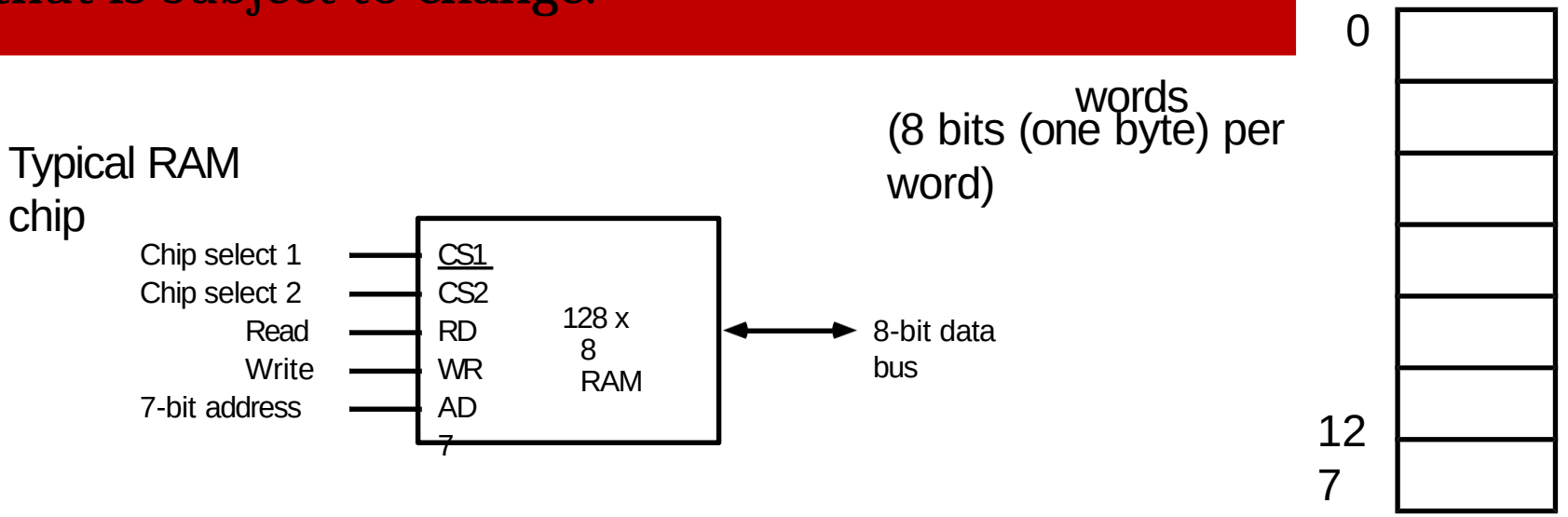
Integrated RAM are available in two possible operating modes, *Static* and *Dynamic*.

ROM– Read Only memory



RANDOM ACCESS MEMORY

RAM is used for storing bulk of programs and data that is subject to change.



CS1	$\overline{CS2}$	RD	WR	Memory function	State of data bus
0	0	x	x	Inhibit	High-impedence
0	1			Inhibit	High-impedence
1	0	x	x	Inhibit	High-impedence
1	0			Write	Input data to RAM
1	0	0	0	Read	Output data from RAM
1	1	0	1	Inhibit	High Impedence
		1	x		
		x			
			x		

SOURCE: Computer architecture by Patterson

TYPES OF RANDOM ACCESS MEMORY

Static RAM (SRAM)

- Each cell stores bit with a six-transistor circuits
- Retains value indefinitely, as long as it is kept powered
- Faster (8-16 times faster) and more expensive (8-16 times more expensive as well) than DRAM

Dynamic RAM (DRAM)

- Each cell stores bit with a capacitor and transistor.
- Value must be refreshed every 10-100 ms.
- Slower and cheaper than SRAM. Has reduced power consumption, and a large storage capacity.

Non Volatile RAM (NVRAM)

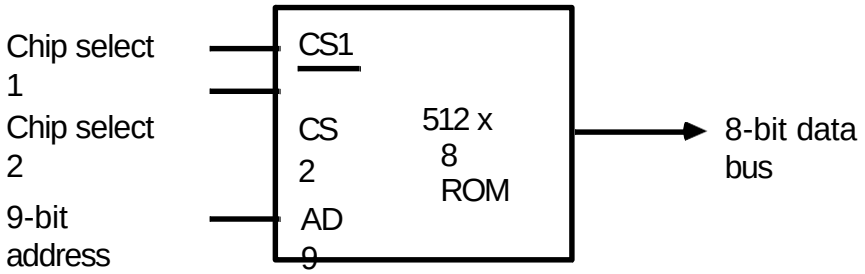
- retains its information when power is turned off (non volatile).
- best-known form of NVRAM memory today is flash memory.

READ ONLY MEMORY

- It is non-volatile memory, which retains the data even when power is removed from this memory. Programs and data that can not be altered are stored in ROM.

- ROM is used for storing programs that are **PERMANENTLY** resident in the computer and for tables of constants that do not change in value once the production of the computer is completed.
- The ROM portion of main memory is needed for storing an initial program called ***bootstrap loader***, which is to start the computer operating system when power is turned on.

Typical ROM chip:



- Since the ROM can only READ, the data bus can only be in output mode.
- No need of READ and WRITE control.
- Same sized RAM and ROM chip , it is possible to have more bits of ROM than of RAM , because the internal binary cells in ROM occupy less space than in RAM.

TYPES OF READ ONLY MEMORY

The required paths in a ROM may be programmed in four different ways:

1. Mask Programming: It is done by the company during the fabrication process of the unit. The procedure for fabricating a ROM requires that the customer fills out the truth table he wishes the ROM to satisfy.

2. Programmable Read only memory(PROM):

PROM contain all the fuses intact giving all 1's in the bits of the stored words. A blown fuse defines binary 0 state and an intact fuse give a binary 1 state. This allows the user to program the PROM by using a special instruments called PROM programmer.

TYPES OF READ ONLY MEMORY

3.Erasable PROM(EPROM): In a PROM once fixed pattern is permanent and can not be altered. The EPROM can be restructured to the initial state even through it has been programmed previously. When EPROM is placed under a special ultra-violet light for a given period of time all the data are erased. After erase, the EPROM returns to it initial state and can be programmed to a new set of values.

4.Electrically Erasable PROM(EEPROM): It is similar to EPROM except that the previously programmed connections can be erased with an electrical signal instead of ultra violet light. The advantage is that device can be erased without removing it from it's socket.

AUXILIARY MEMORY

- Also called as Secondary Memory, used to store large chunks of data at a lesser cost per byte than a primary memory for backup.
  It does not lose the data when the device is powered down

  It is not directly accessible by the CPU, they are accessed via the input/output channels.

  The most common form of auxiliary memory devices used in consumer systems is flash memory, optical discs, and magnetic disks, magnetic tapes.

TYPES OF AUXILIARY MEMORY

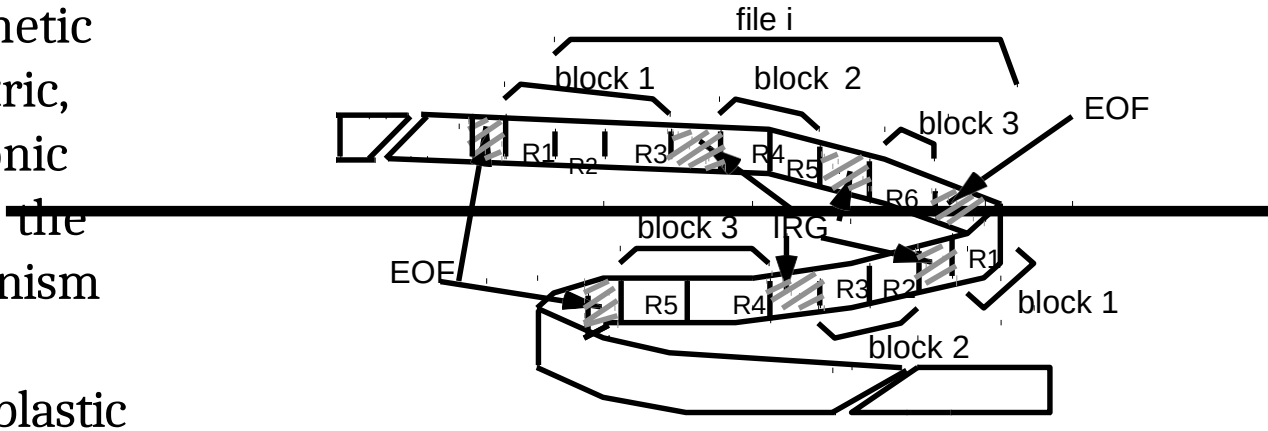
Flash memory: An electronic non-volatile computer storage device that can be electrically erased and reprogrammed, and works without any moving parts. Examples of this are USB flash drives and solid state drives.

Optical disc: Its a storage medium from which data is read and to which it is written by lasers. There are three basic types of optical disks: CD- ROM (read-only), WORM (write-once read- many) & EO (erasable optical disks).

TYPES OF AUXILIARY MEMORY

Magnetic tapes: A magnetic tape consists of electric, mechanical and electronic components to provide the parts and control mechanism for a magnetic tape unit.

The tape itself is a strip of plastic coated with a magnetic recording medium. Bits are recorded as magnetic spots on tape along several tracks called RECORDS.



SOURCE: Computer architecture by Patterson

TYPES OF AUXILIARY MEMORY

Magnetic Disk:

- A magnetic disk is a circular plate constructed of metal or plastic coated with magnetized material.
- Both sides of the disk are used and several disks may be stacked on one spindle with read/write heads available on each surface.
- Bits are stored in magnetized surface in spots along concentric circles called tracks. Tracks are commonly divided into sections called sectors.
- Disk that are permanently attached and cannot removed by occasional user are called hard disks.

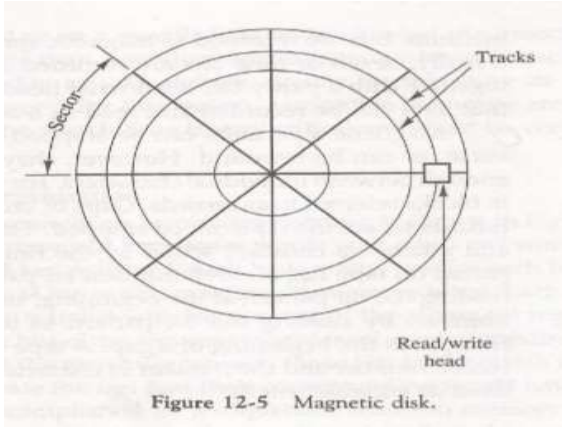


Figure 12-5 Magnetic disk.

SOURCE: Computer architecture by Patterson

ASSOCIATIVE MEMORY

- A memory unit accessed by contents is called an associative memory or content addressable memory(CAM).
- This type of memory is accessed simultaneously and in parallel on the basis of data content rather than by specific address or location.

READ/ WRITE OPERATION IN ASSOCIATIVE MEMORY

Write operation:

- When a word is written in in an associative memory, no address is given
- The memory is capable of finding an unused location to store the word.

Read operation:

- When a word is to be read from an associative memory, the contents of the word, or a part of the word is specified.
- The memory locates all the words which match the specified content and marks them for reading.

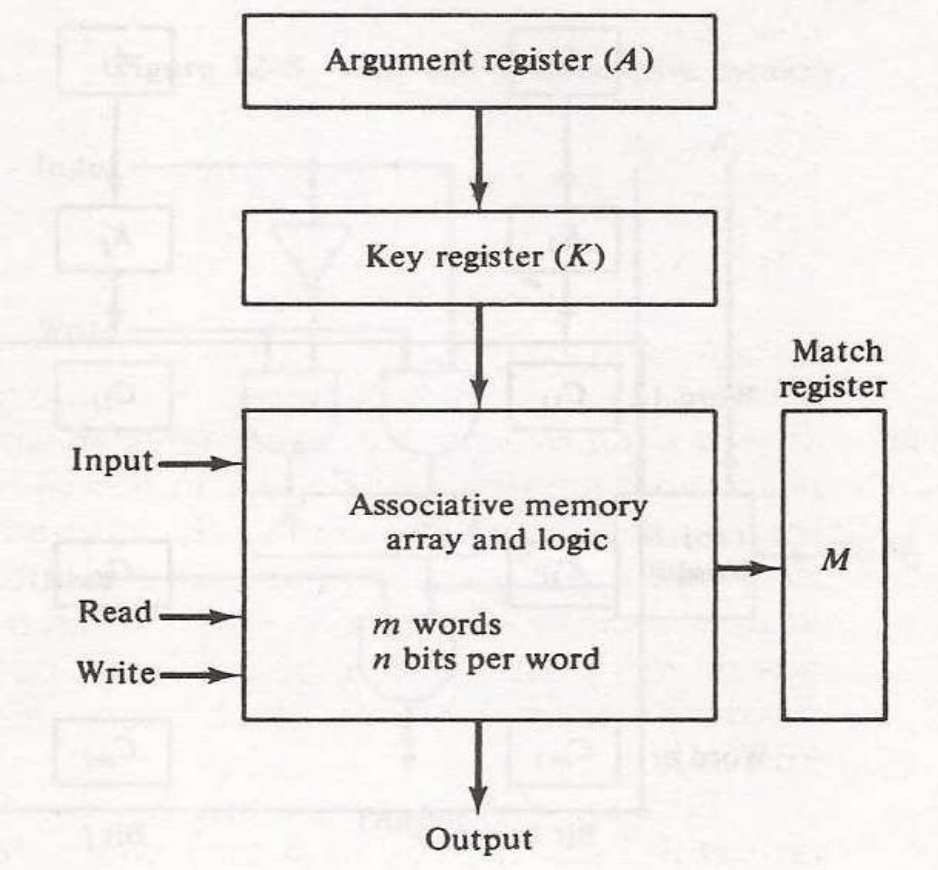
HARDWARE ORGANIZATION

Argument register(A): It contains the word to be searched. It has *n* bits(one for each bit of the word).

Key Register(K): It provides mask for choosing a particular field or key in the argument word. It also has *n* bits.

Associative memory array: It contains the words which are to be compared with the argument word.

Match Register(M): It has m bits, one bit corresponding to each word in the memory array . After the matching process, the bits corresponding to matching words in



SOURCE: Computer architecture by morris mano

MATCHING PROCESS

- The entire argument word is compared with each memory word, if the key register contains all 1's. Otherwise, only those bits in the argument that have 1's in their corresponding position of the key register are compared.
- Thus the key provides a mask or identifying piece of information which specifies how the reference to memory is made.
- To illustrate with a numerical example, suppose that the argument register A and the key register K have the bit configuration as shown below.
- Only the three left most bits of A are compared with the memory words because K has 1's in these three positions only.

A	101	111100	
K	111	000000	
Word 1	100	111100	no match
Word 2	101	000001	match

SOURCE: Computer architecture by Stallings, W

DISADVANTAGES

- An associative memory is more expensive than a random access memory because each cell must have an extra storage capability as well as logic circuits for matching its content with an external argument.

- For this reason, associative memories are used in applications where the search time is very critical and must be very short.

CACHE MEMORY

- If the active portions of the program and data are placed in a fast small memory, the **average memory access time** can be reduced.
- Thus reducing the **total execution time** of the program
- Such a fast small memory is referred to as cache memory
- The cache is the fastest component in the memory hierarchy and approaches the speed of CPU component

BASIC OPERATION OF CACHE MEMORY

- When CPU needs to access memory, the cache is examined.
- If the word is found in the cache, it is read from the cache memory.
- If the word addressed by the CPU is not found in the cache, the main memory is accessed to read the word.
- A block of words containing the one just accessed is then transferred from main memory to cache memory.
- If the cache is full, then a block equivalent to the size of the used word is replaced according to the replacement algorithm being used.

HIT RATIO

- When the CPU refers to memory and finds the word in cache, it is said to produce a **hit**

  Otherwise, it is a **miss**

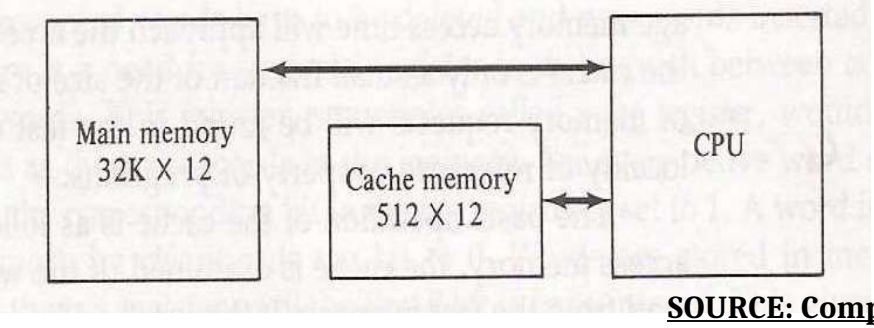
- The performance of cache memory is frequently measured in terms of a quantity called **hit ratio**

Hit ratio = hit / (hit+miss)

MAPPING PROCESS

- The transformation of data from main memory to cache memory is referred to as a **mapping** process, there are three types of mapping:
  - Associative mapping
  - Direct mapping
  - Set-associative mapping

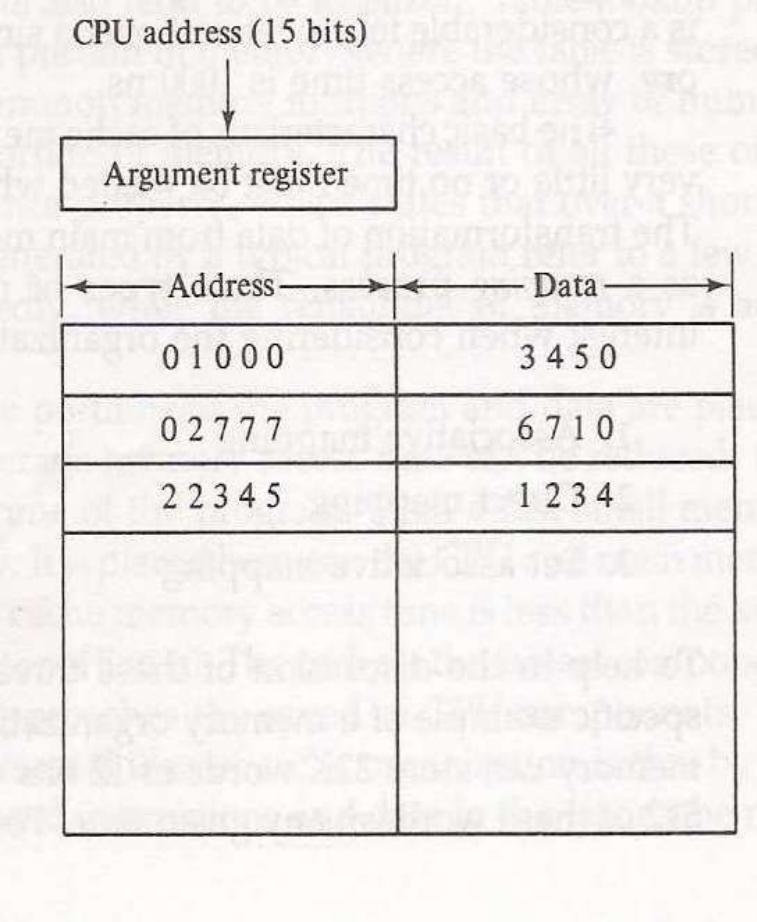
- To help understand the mapping procedure, we have the following example:



SOURCE: Computer architecture by Patterson and Hennessy

ASSOCIATIVE MAPPING

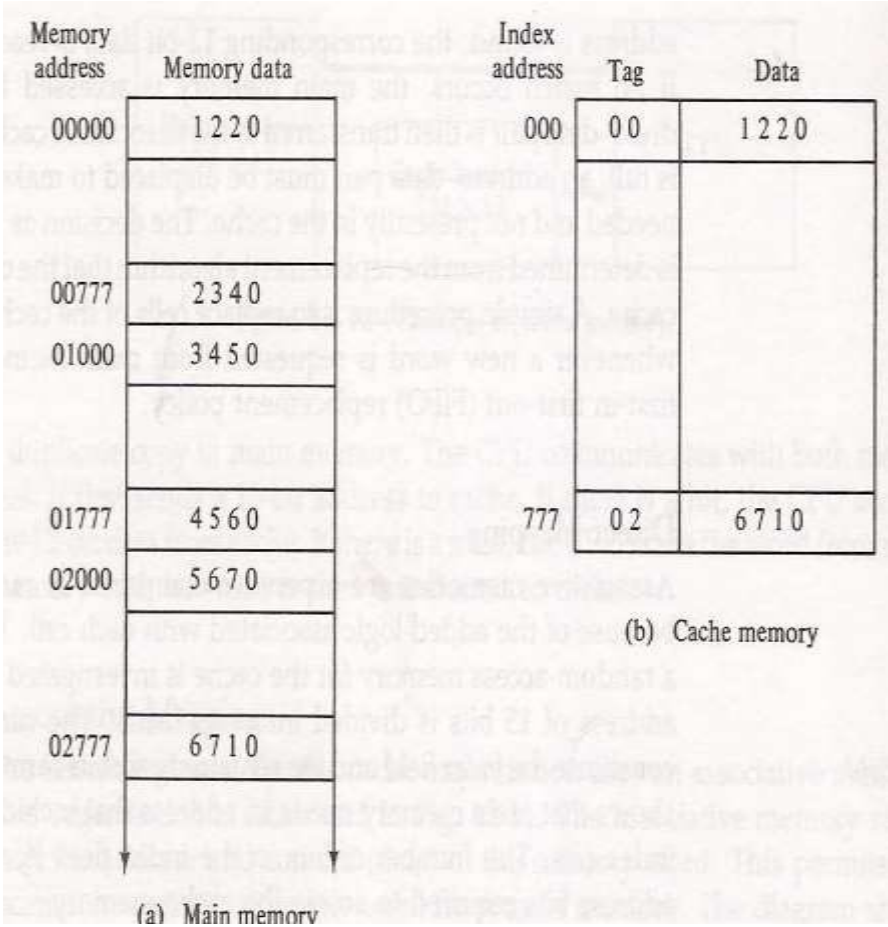
- The fastest and most flexible cache organization uses an associative memory.
- The associative memory stores both the address and data of the memory word.
- This permits any location in cache to store a word from main memory.
- The address value of 15 bits is shown as a five-digit **octal** number and its corresponding 12-bit word is shown as a four-digit octal



SOURCE: Computer architecture by Morris Mano

DIRECT MAPPING

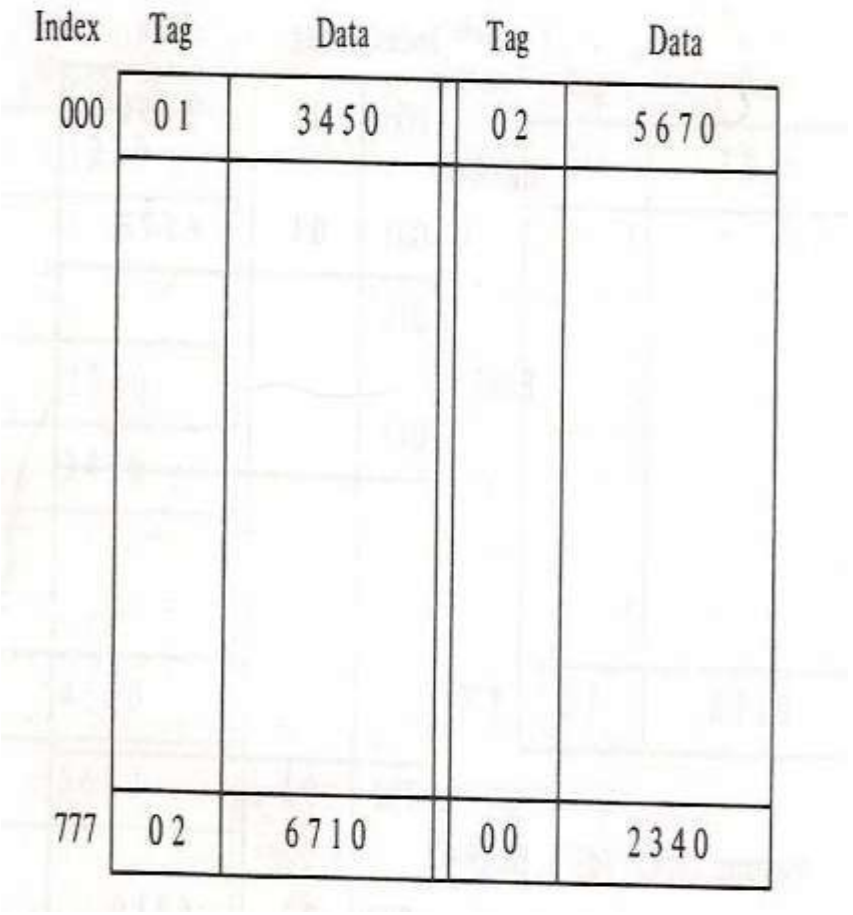
- Associative memory is expensive compared to RAM.
- In general case, there are 2^k words in cache memory and 2^n words in main memory (in our case, k=9, n=15).
- The n bit memory address is divided into two fields: k-bits for the index and
- n-k bits for the tag field.



SOURCE: Computer architecture by Morris Mano

SET ASSOCIATIVE MAPPING

- The disadvantage of direct mapping is that two words with the same index in their address but with different tag values cannot reside in cache memory at the same time.
- Set-Associative Mapping is an improvement over the direct-mapping in that each word of cache can store two or more word of memory under the same index address.



Index	Tag	Data	Tag	Data
000	01	3450	02	5670
777	02	6710	00	2340

SOURCE: Computer architecture by Morris Mano

REPLACEMENT ALGORITHMS

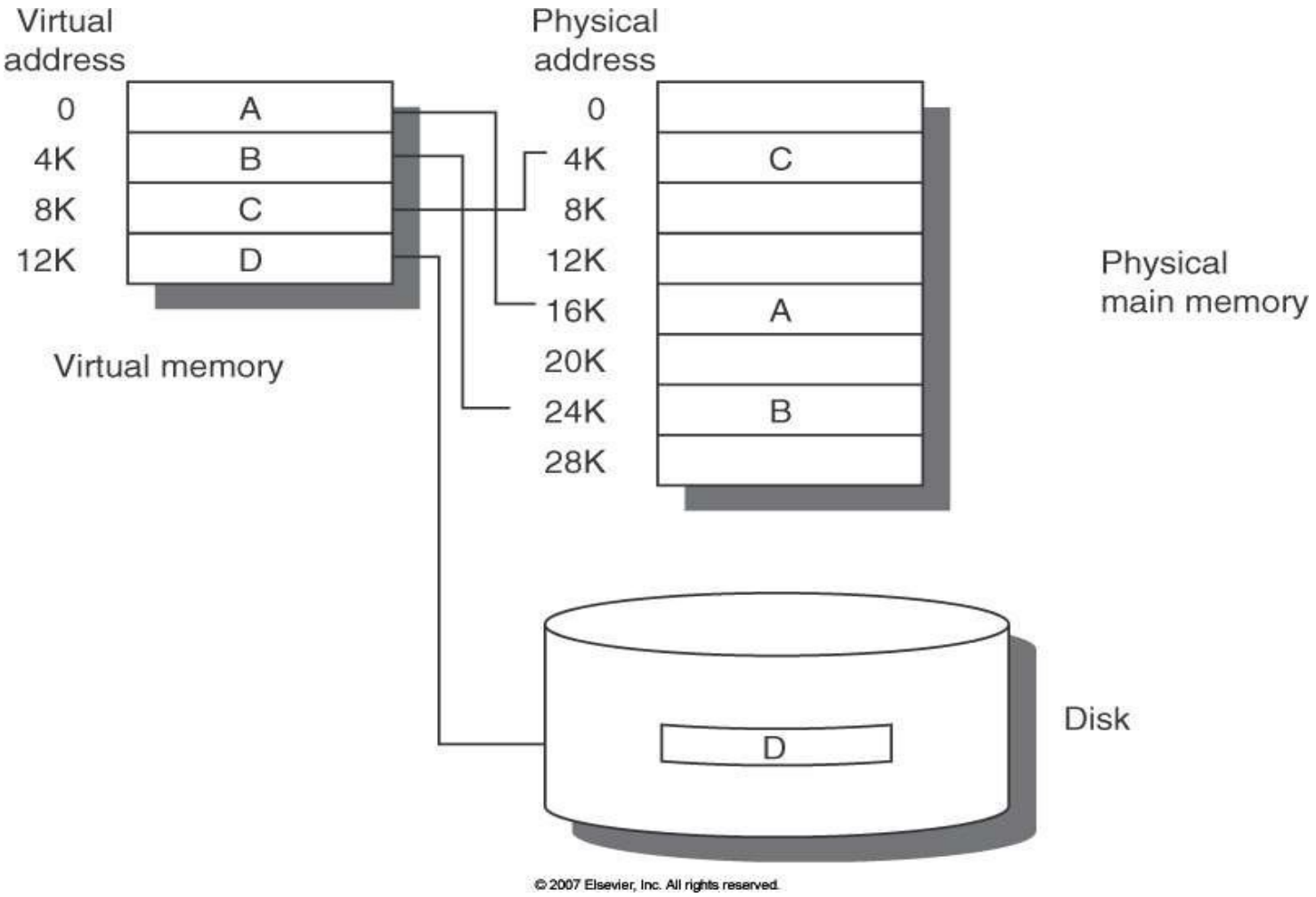
- Optimal replacement algorithm – find the block for replacement that has minimum chance to be referenced next time.

Two algorithms:

- FIFO: Selects the item which has been in the set the longest.
- LRU: Selects the item which has been least recently used by the CPU.

VIRTUAL MEMORY

- Virtual memory is a common part of operating system on desktop computers.
- The term Virtual Memory refers to something which appears to be present but actually is not.
- This technique allows users to use more memory for a program than the real memory of a computer.



SOURCE: Computer architecture by J.P. Hayes

NEED FOR VIRTUAL MEMORY

- Virtual Memory is a imaginary memory which we assume or use, when we have a material that exceeds our memory at that time.

- Virtual Memory is temporary memory which is used along with the ram of the system.

ADVANTAGES

- Allows Processes whose aggregate memory requirement is greater than the amount of physical memory, as infrequently used pages can reside on the disk.
- Virtual memory allows speed gain when only a particular segment of the program is required for the execution of the program.
- This concept is very helpful in implementing multiprogramming environment.

DISADVANTAGES

- Applications run rather slower when they are using virtual memory.
- It takes more time to switch between applications.
- Reduces system stability.

COA

Chapter-5(Unit-ii)
Input/output organization

TOPICS COVERED

- Asynchronous Data transfer:
  Source Initiated, Destination Initiated,
- Handshaking,
- Programmed I/O
- Interrupts
- DMA
- IOP

SYNCHRONOUS DATA TRANSFER

- Synchronous transfers usually occur when peripherals are located within the same computer as the CPU because their close proximity allows them to share a common clock and because data does not have to travel very far physically, which becomes a concern at a higher clock frequencies.

ASYNCHRONOUS DATA TRANSFER

- A computer can make use of asynchronous data transfers when synchronous transfers are not viable.
- Asynchronous transfers use control signals and their associated hardware to coordinate the movement of data. These data transfer do not require that the source and destination use the same system clock.
- There are four types of asynchronous data transfers.

TYPES OF ASYNCHRONOUS DATA TRANSFER

- Source initiated without handshaking
- Destination initiated without handshaking
- Source initiated with handshaking
- Destination initiated with handshaking

HANDSHAKING

- For some devices, particularly electromechanical devices, do not require the same amount of time for every transfer, they can use handshaking to coordinate their transfers.

- Handshaking uses an additional control signal to indicate that data is ready or has been read in.

SOURCE INITIATED WITHOUT HANDSHAKING

- The source outputs its data.
- Then strobes a control signal for a set amount of time.
- The destination device reads in the data during this time.
- The source device next desserts the strobe and stops outputting data.

DESTINATION INITIATED WITHOUT HANDSHAKING

- The destination device transmits a data strobe signal to the source device which, after a brief delay, makes data available.
- The destination device reads in this data
- The destination device deasserts the data strobe.

SOURCE INITIATED WITH HANDSHAKING

- The source sets the data request signal high.
- Makes valid data available to the destination device
- The destination device reads in data.
- The destination device sends a data acknowledge signal to the source.
- The source sets its data request line low and stops sending data.
- The destination then resets its data acknowledge signal.

DESTINATION INITIATED WITH HANDSHAKING

- Similar to that of the source-initiated data transfer using handshaking, except that the data-acknowledge signal is replaced by a data-ready signal.

PROGRAMMED INPUT OUTPUT

- Definition:

  Programmed I/O is exactly what its name implies: A program instruction causes the CPU to input or output data.
- Programmed I/O can be either isolated or memory mapped.
- Isolated I/O uses separate instructions to access I/O ports.
- Memory-mapped I/O treats I/O ports as memory locations.

INTERRUPTS

- In program-controlled I/O, the program enters a wait loop in which it repeatedly tests the device status. During the period, the processor is not performing any useful computation.
- However, in many situations other tasks can be performed while waiting for an I/O device to become ready.
- Let the device alert the processor.

Enabling and Disabling Interrupts

- Since the interrupt request can come at any time, it may alter the sequence of events from that envisaged by the programmer.
- Interrupts must be controlled.

Enabling and Disabling Interrupts

- The interrupt request signal will be active until it learns that the processor has responded to its request. This must be handled to avoid successive interruptions.
- Let the interrupt be disabled/enabled in the interrupt-service routine.
- Let the processor automatically disable interrupts before starting the execution of the interrupt-service routine.

Handling Multiple Devices

- How can the processor recognize the device requesting an interrupt?
- Given that different devices are likely to require different interrupt-service routines, how can the processor obtain the starting address of the appropriate routine in each case?
- (Vectored interrupts)
- Should a device be allowed to interrupt the processor while another interrupt is being serviced?
- (Interrupt nesting)
- How should two or more simultaneous interrupt requests be handled?
- (Daisy-chain)

Vectored Interrupts

- A device requesting an interrupt can identify itself by sending a special code to the processor over the bus.
- Interrupt vector
- Avoid bus collision

Interrupt Nesting

- Simple solution: only accept one interrupt at a time, then disable all others.
- Problem: some interrupts cannot be held too long.
- Priority structure

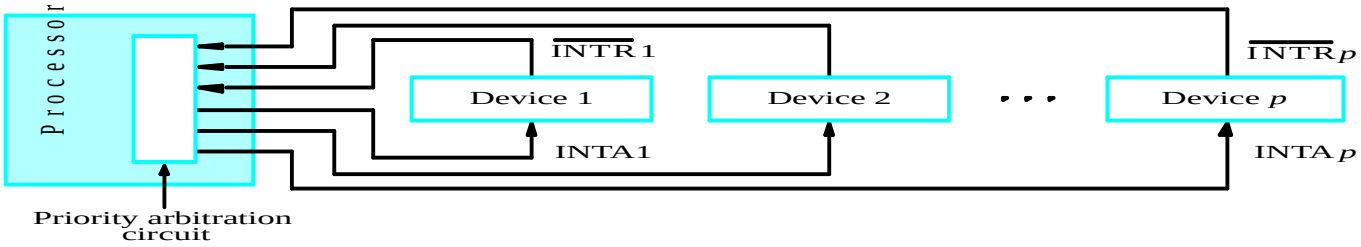


Figure 4.7. Implementation of interrupt priority using individual interrupt-request and acknowledge lines.

SOURCE: Carpinelli J.D," Computer systems organization &Architecture", Fourth Edition, Addison Wesley.

Simultaneous Requests

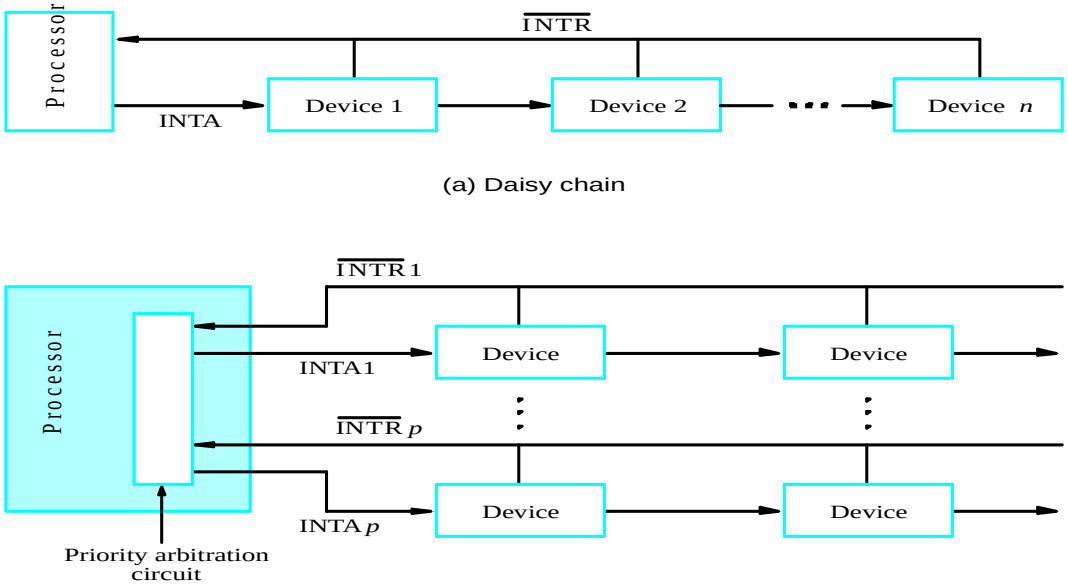


Figure 4.8. Interrupt priority schemes.

SOURCE: Carpinelli J.D," Computer systems organization &Architecture", Fourth Edition, Addison Wesley.

Controlling Device Requests

- Some I/O devices may not be allowed to issue interrupt requests to the processor.
- At device end, an interrupt-enable bit in a control register determines whether the device is allowed to generate an interrupt request.
- At processor end, either an interrupt enable bit in the PS register or a priority structure determines whether a given interrupt request will be accepted.

Exceptions

- Recovery from errors
- Debugging
  - Trace
  - Breakpoint
- Privilege exception

Use of Interrupts in Operating Systems

- The OS and the application program pass control back and forth using software interrupts.
- Supervisor mode / user mode
- Multitasking (time-slicing)
- Process – running, runnable, blocked
- Program state

DMA

- Think about the overhead in both polling and interrupting mechanisms when a large block of data need to be transferred between the processor and the I/O device.
- A special control unit may be provided to allow transfer of a block of data directly between an external device and the main memory, without continuous intervention by the processor – direct memory access (DMA).
- The DMA controller provides the memory address and all the bus signals needed for data transfer, increment the memory address for successive words, and keep track of the number of transfers.

- Processor sends the starting address, the number of data, and the direction of transfer to DMA controller.
- Processor suspends the application program requesting DMA, starts DMA transfer, and starts another program.
- After the DMA transfer is done, DMA controller sends an interrupt signal to the processor.
- The processor puts the suspended program in the Runnable state.

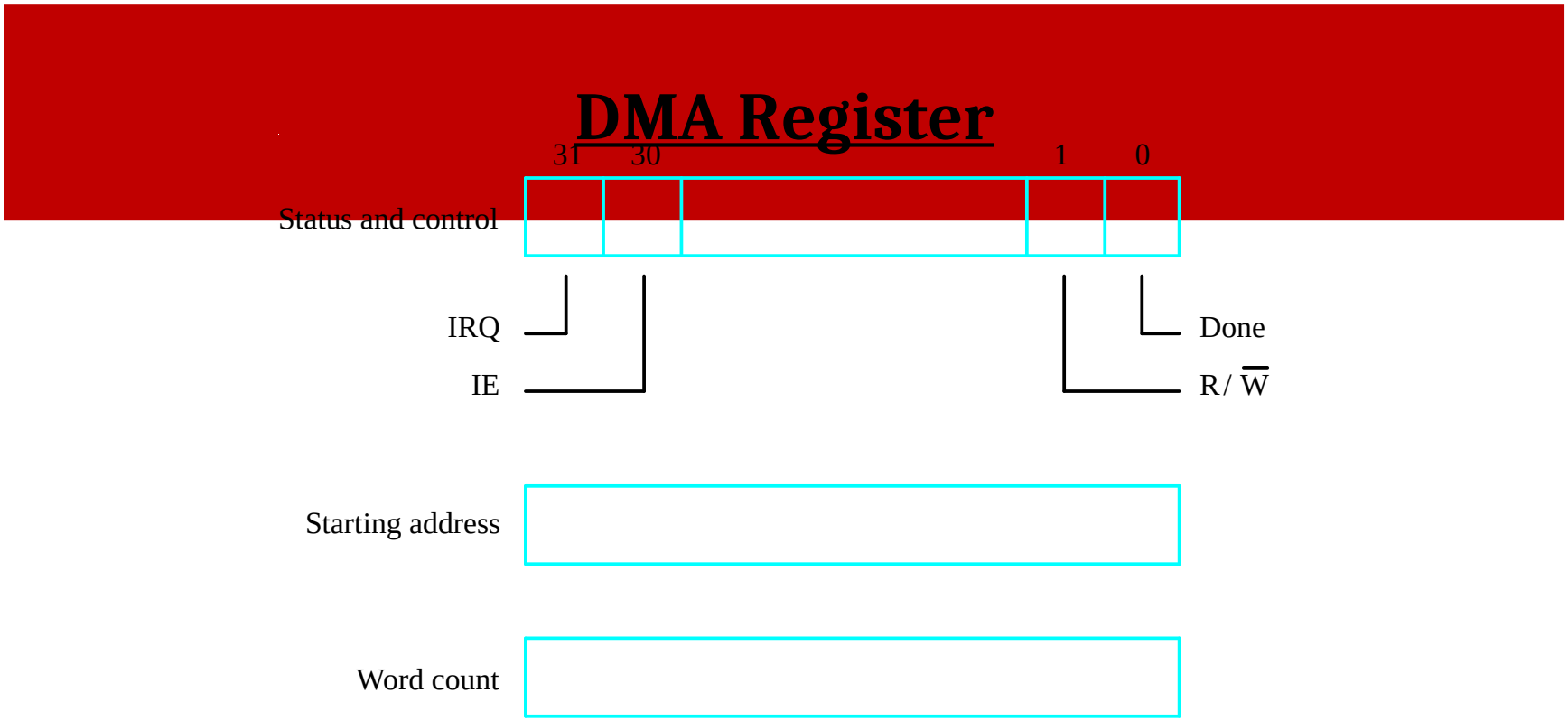


Figure 4.18. Registers in a DMA interface.

SOURCE: Carpinelli J.D," Computer systems organization &Architecture", Fourth Edition, Addison Wesley.

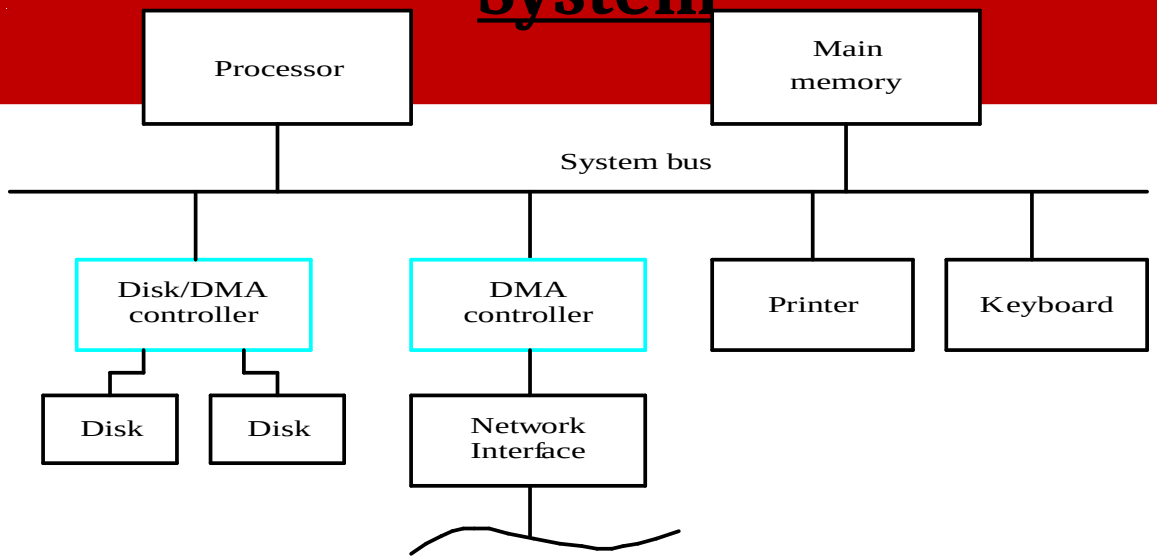


Figure 4.19. Use of DMA controllers in a computer system.

SOURCE: Carpinelli J.D," Computer systems organization &Architecture", Fourth Edition, Addison Wesley.

Memory Access

- Memory access by the processor and the DMA controller are interwoven.
- DMA device has higher priority.
- Among all DMA requests, top priority is given to high-speed peripherals.
- Cycle stealing
- Block (burst) mode
- Data buffer
- Conflicts

Course Outcomes

- Students will understand input/output mechanisms
- Students will understand the various parts of a system memory hierarchy.
- Students will understand various CPU mapping techniques with numerical problems

REFERNCES

Text Books:

•Carpinelli J.D," Computer systems organization &Architecture", Fourth Edition, Addison Wesley.
•Patterson and Hennessy, "Computer Architecture" , Fifth Edition Morgaon Kauffman.

Reference Books:

•J.P. Hayes, "Computer Architecture and Organization", Third Edition.
•Mano, M., "Computer System Architecture", Third Edition, Prentice Hall.
•Stallings, W., "Computer Organization and Architecture", Eighth Edition, Pearson Education.

COURSE APPLICATIONS

•Computer organization and architecture course deals with instruction set architecture, micro architecture and efficient implementation of micro architecture.

• Understanding the computer architecture concepts is essential for students interested in hardware, processor design, compilers, and operating systems.

CONTENT / APPLICATIONS BEYOND SYALLABUS

- Concept of Thrashing
- Paging and Segmentation Combined Mapping Procedure

FAQS

Q1. Give the comparison between & examples of hardwired control unit and micro programmed control unit.
Q2. How Cache Memory is useful in memory hierarchy?
Q3.Difference between direct and associative mapping?
Q4.What are the advantages of microprogrammed control unit over hardwired control unit.
Q5.What is the role of microprogram sequencer?
Q6.Discuss Microinstruction format?
Q7.Discuss the following terms:
Microcode
Microprogram
Microinstruction
Q8. How Cache Memory is useful in memory hierarchy?
Q.9. What do you mean by initialisation of DMA controller?
Q10. Why is set associative memory so called?
Q11.Difference between paging and segmentation?
Q12.How is decoding of microoperations is done in microprogrammed control unit?
Q13.What are the limitations of microprogrammed control unit

Thank you