UNIT 4

THE MEMORY SYSTEM

Basic Concepts

- The maximum size of the Main Memory is determined by its addressing scheme.
- A 16-bit computer that generates 16-bit addresses is capable of addressing upto 2^{16} =64K memory locations.
- If a machine generates 32-bit addresses, it can access upto 2^{32} = 4G memory locations.
- This number represents the size of *address space* of the computer.

Contd - Word-address and Byte-address

- If the smallest addressable unit of information is a memory word, the machine is called word-addressable.

- If individual memory bytes are assigned distinct addresses, the computer is called byte-addressable

- A word-address assignment for a 32-bit computer:

Word Address	**Byte Address**			
0	0	1	2	3
4	4	5	6	7
8	8	9	10	11
.		…..		

Contd – Measure of speed of memory unit

- **Memory Access Time -** the time between Read/Write and MFC

It is the time that elapses between the initiation of an operation and the completion of that operation

- **Memory Cycle Time -** the time between two successive Read/Write operations

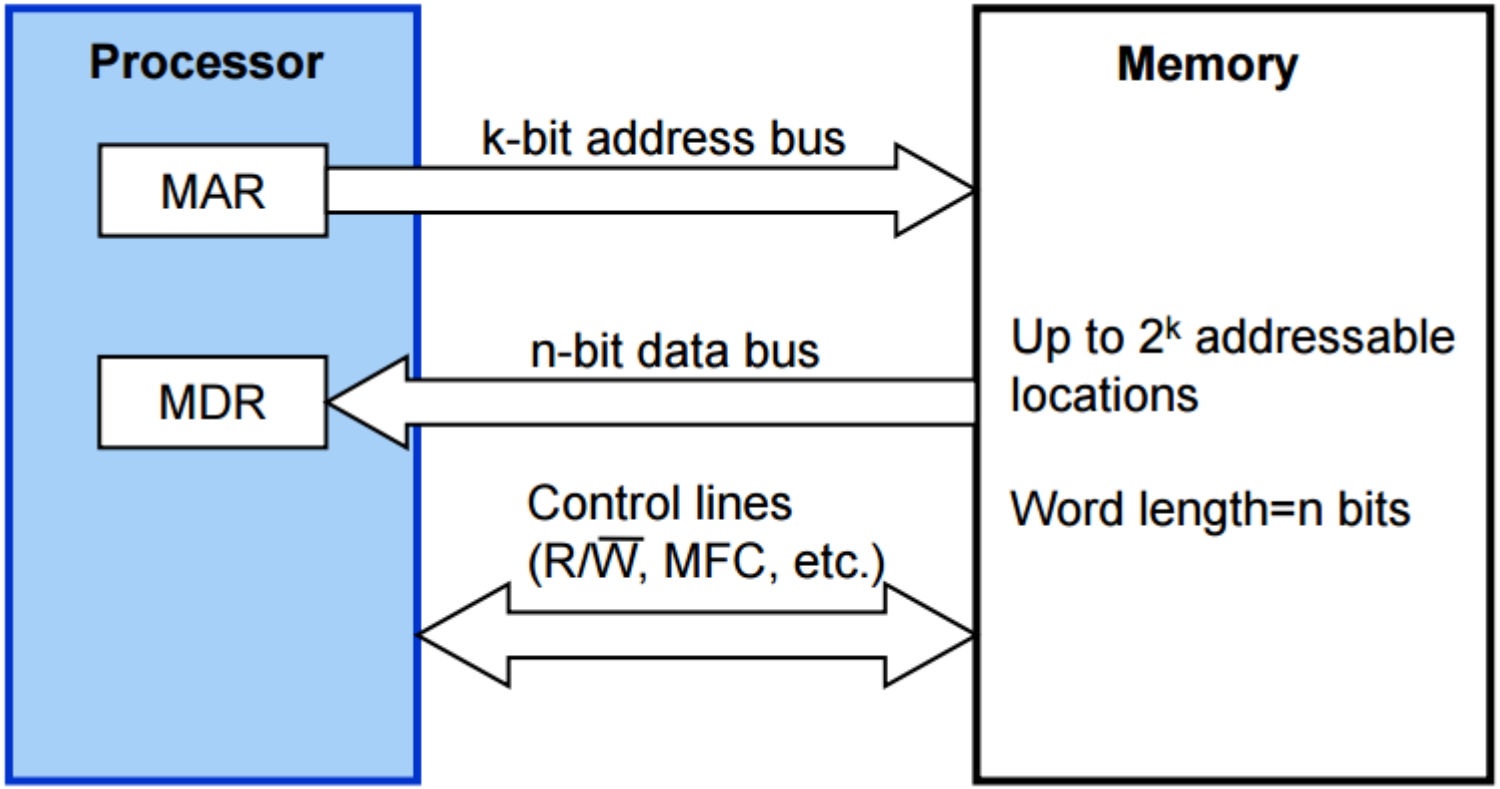
It is the minimum time delay required between the initiations of two successive memory operations

- The cycle time is usually slightly longer than the access time.

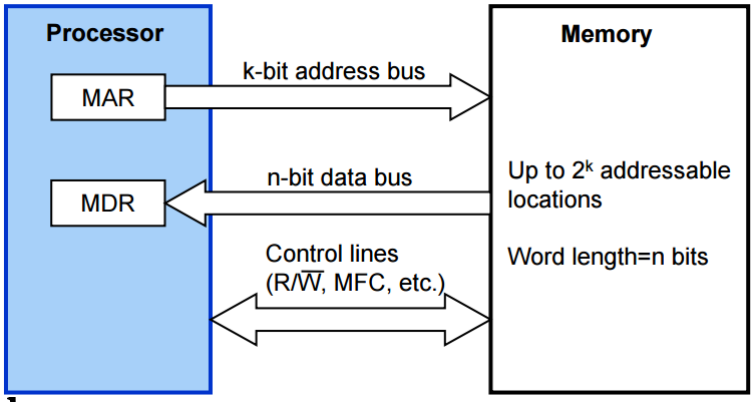
Contd.. - Random Access Memory (RAM)

- Access time for any location is independent of the location's address.
- Any location can be accessed in some fixed amount of time
- i.e access time for all locations is same
- In serial access storage devices (magnetic tapes & disks) access time is different for different locations

CPU-Main Memory Connection



Memory Read operation :



- CPU loads the address into MAR
- R/W' is set to 1
- data from the MM is placed on the data bus
- set MFC (memory function complete) to 1
- Copy the data from data lines to MDR

Memory Write operation :

- CPU loads the address into MAR and data to MDR
- R/W' is set to 0
- MM control circuitry loads the data into appropriate location
- sets MFC to 1

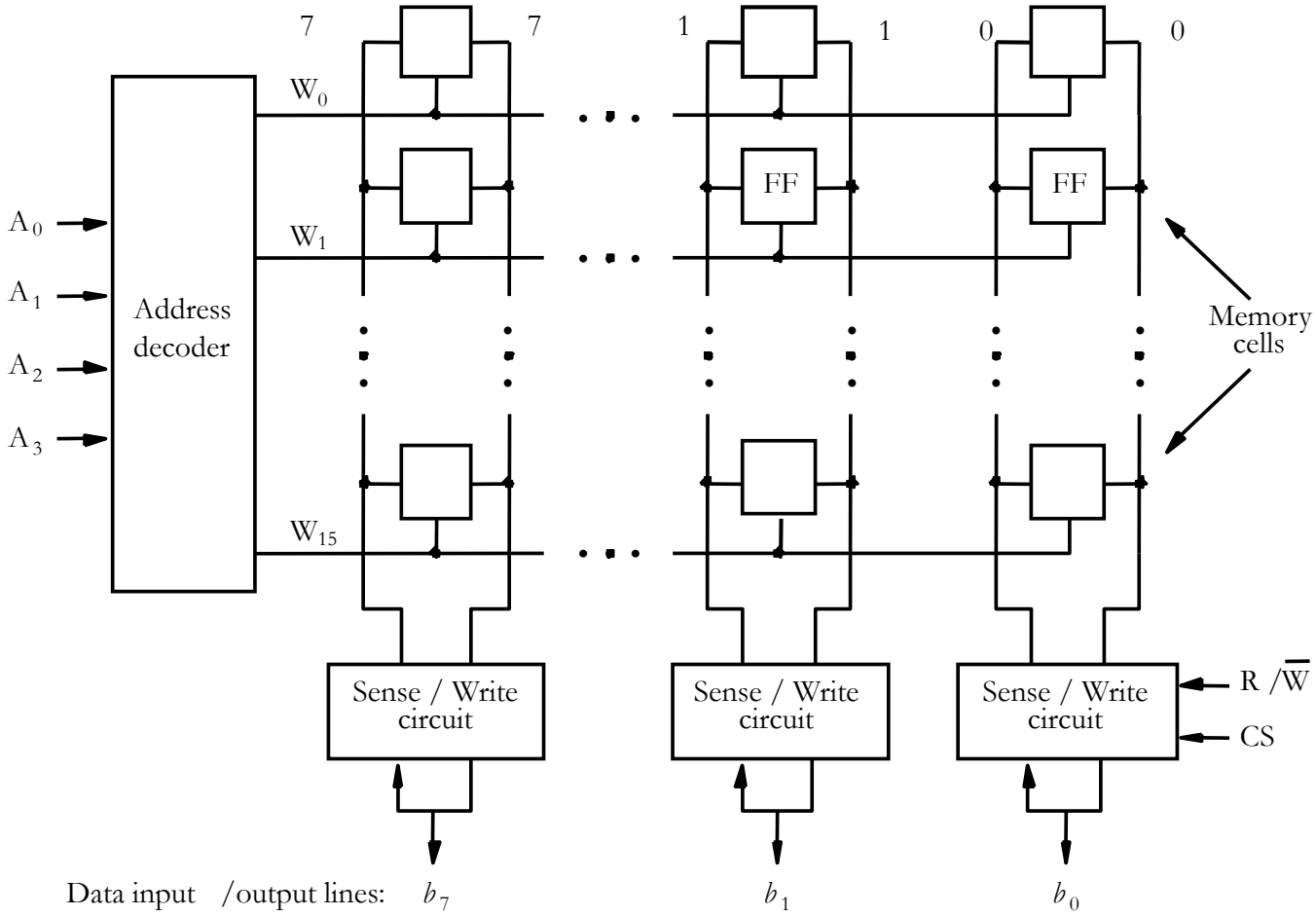
Internal Organization of Semiconductor Memory Chips

- A memory cell is capable of storing one bit of information
- Cells are organized in the form of an array
- A row of cells constitutes a **memory word**
- All cells of a row are connected to the **word line**
- word line is driven by the address decoder

Contd..

- Cells in each column are connected to a sense/write circuit by two lines known as **bit lines**
- sense/write circuits are connected to the data input/output lines
- During a READ operation Sense/Write circuits sense, or read, the information stored in the cells selected by a word line
- During a WRITE operation, they receive input information and store it in the cells of the selected word

Internal organization of a 16x8 memory chip



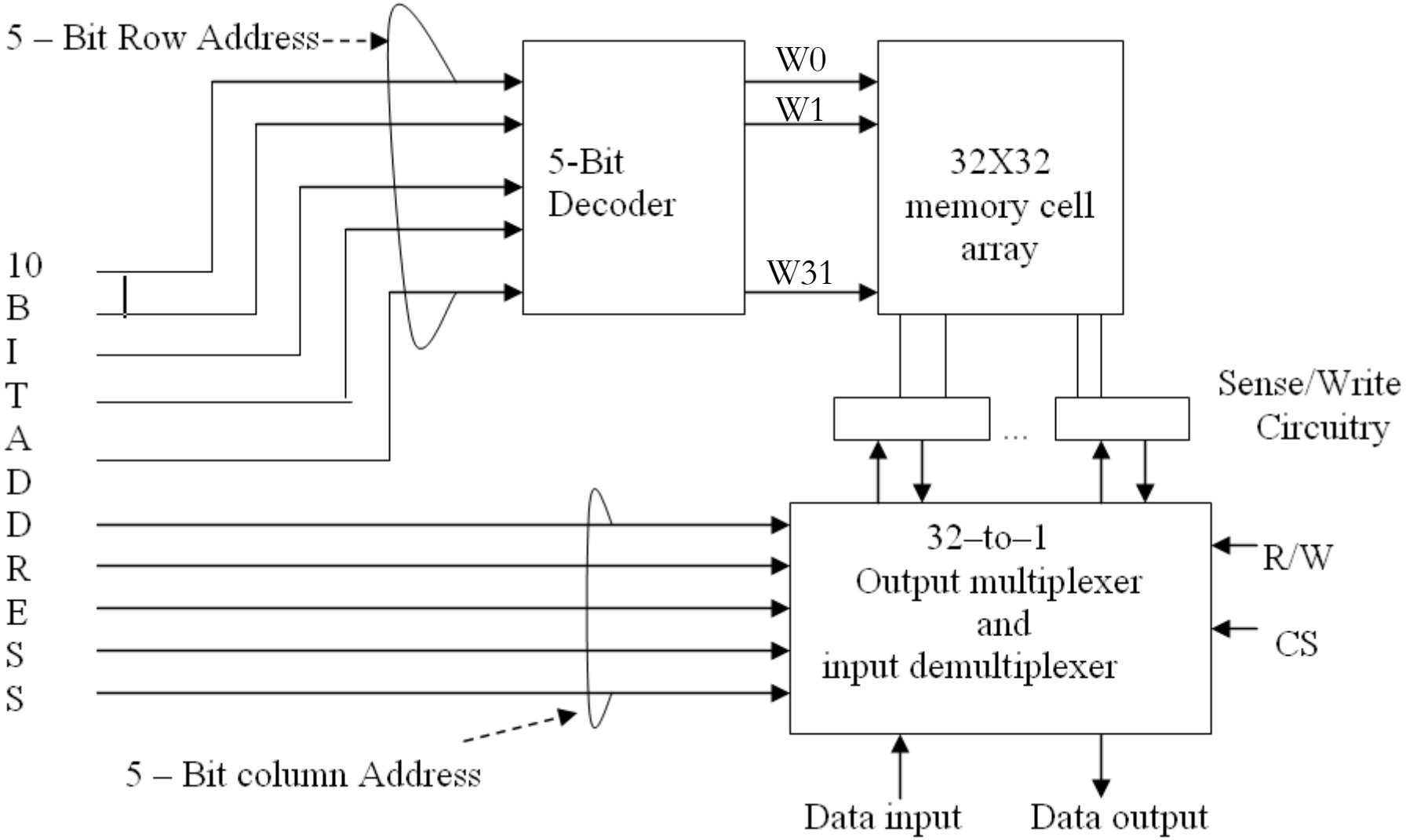
Contd..

- This chip consists of 16 words of 8 bits each
- Usually referred to as a 16 x 8 organization; stores 128 bits
- Requires 14 + 2 external connections

1K x 1 Memory chip

- 10-bit address is split into 5 bit row address and 5 bit column address
- 32 cells in a row are accessed in parallel
- Only one external data line
- The o/p mux and i/p demux, using the column address connect at a time only one of the cells in a row to the external data line
- Stores 1024 bits
- 16-pins=10 addr + 2 data + 2 contrl + 2 power
- CS –chip select, used in multichip organization

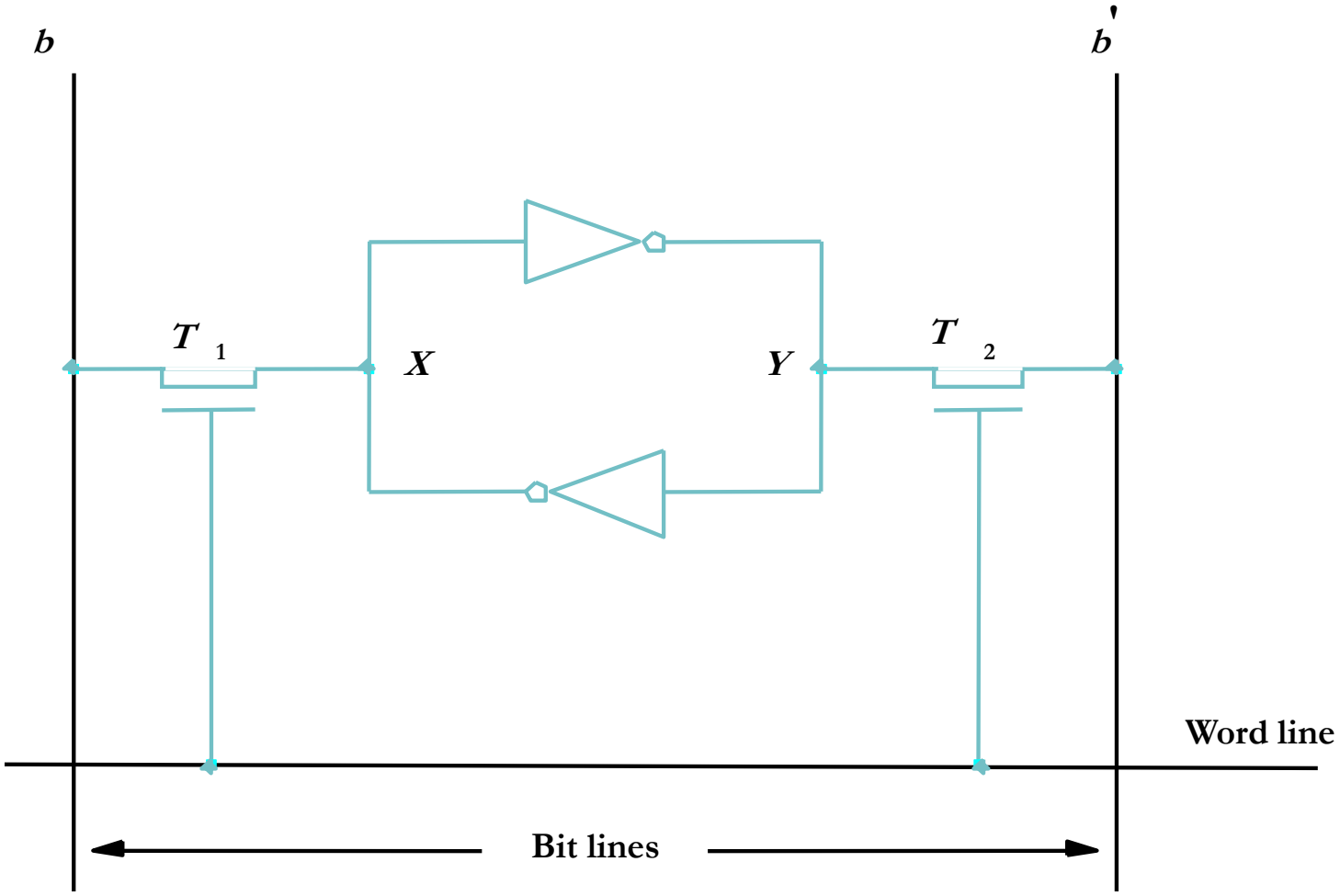
Organization of 1K X 1 memory chip



Static and Dynamic Memory

- **Static memory** : can store information as long as current flow to the cell is maintained.
- Also called Static RAM (SRAM)
- **Dynamic memory** : requires not only the maintaining of a power supply, but also a periodic "refresh" to maintain the information stored in them.
- Also called Dynamic RAM (DRAM)

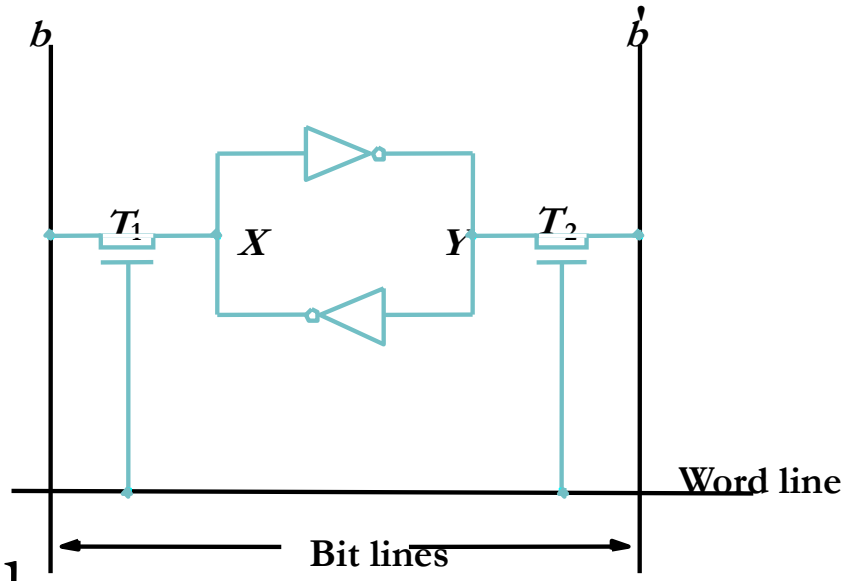
A static memory cell



Contd…

- 2 inverters form a latch
- T1 and T2 connect the latch to bit lines b and b’
- T1 and T2 act as switches; closed if the word line is high.
- If word line is low, T1 and T2 will be off ; latch retains its state.

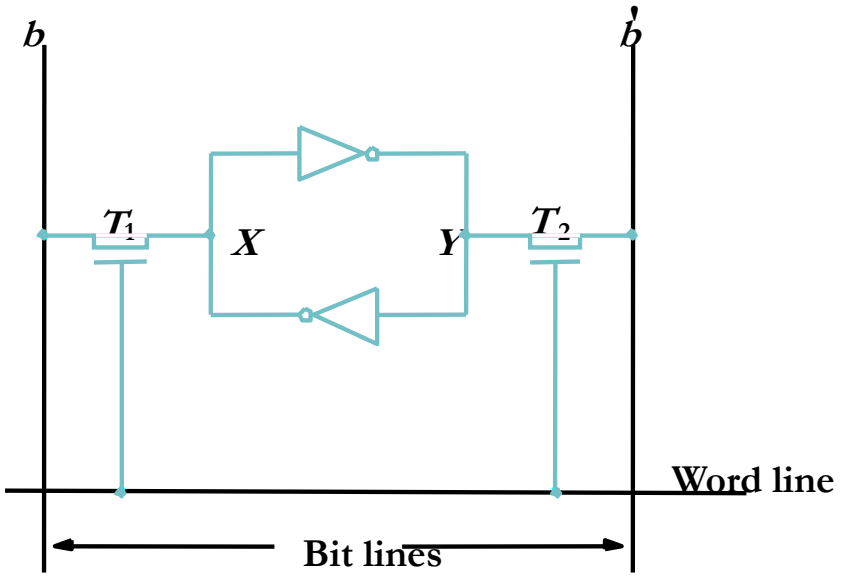
Contd..



Read operation:

- activate the word line
- Now, T1 and T2 are closed
- If the cell is in state 1, b becomes 1 and b' becomes 0 (and vice versa)
- Sense/Write circuit at the end of bit line set the o/p accordingly

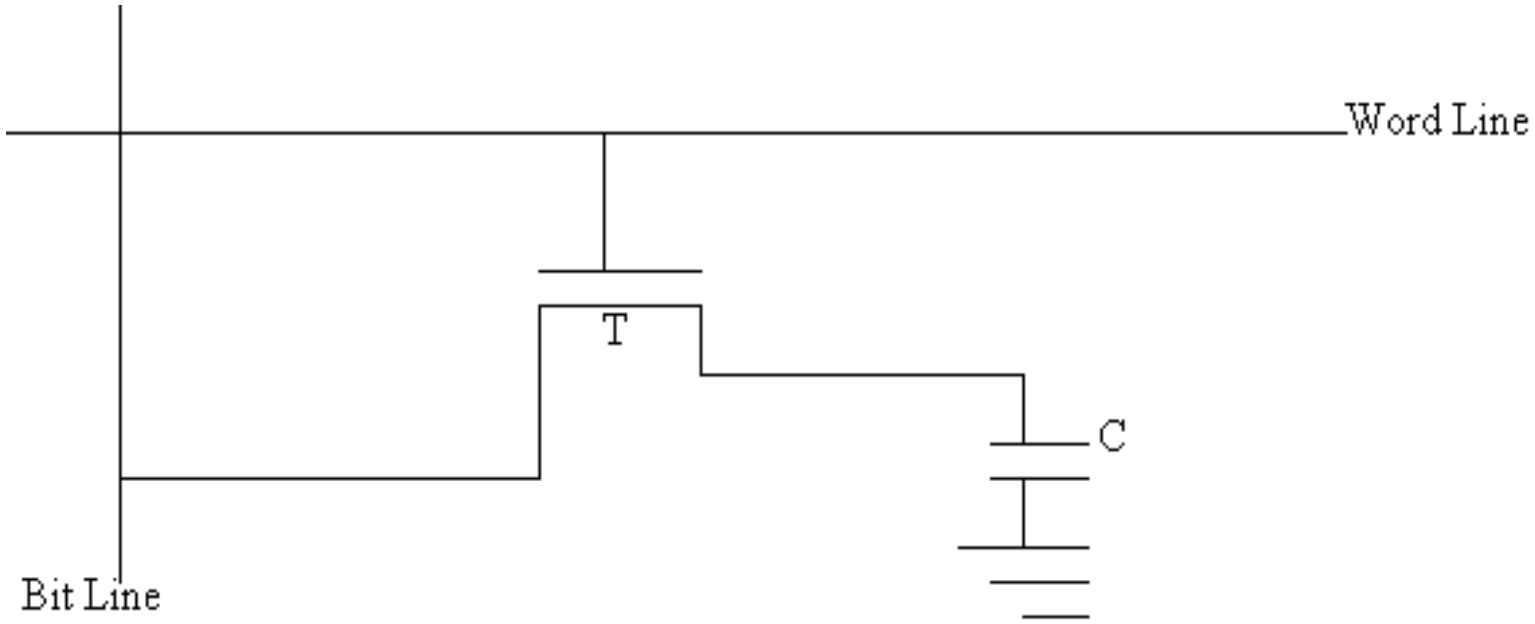
Contd..



Write operation:

- Place appropriate value on b (and its complement on b')
- Activate the word line
- Now, the cell will attain the state of bit line b

A dynamic memory cell

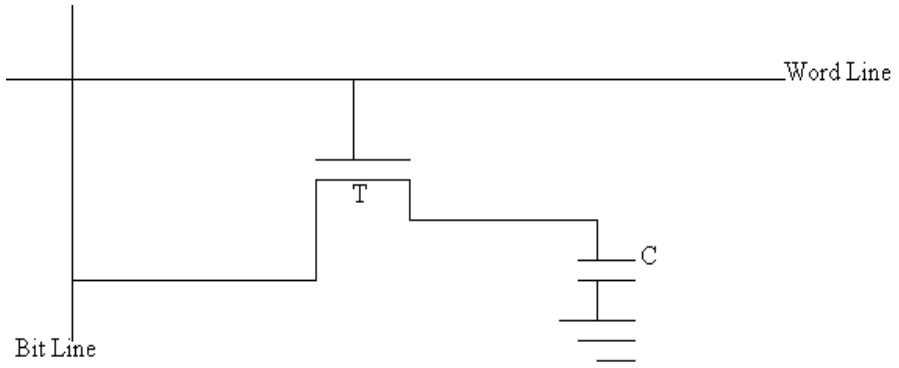


Information is stored in the form of charge on the capacitor

Contd..

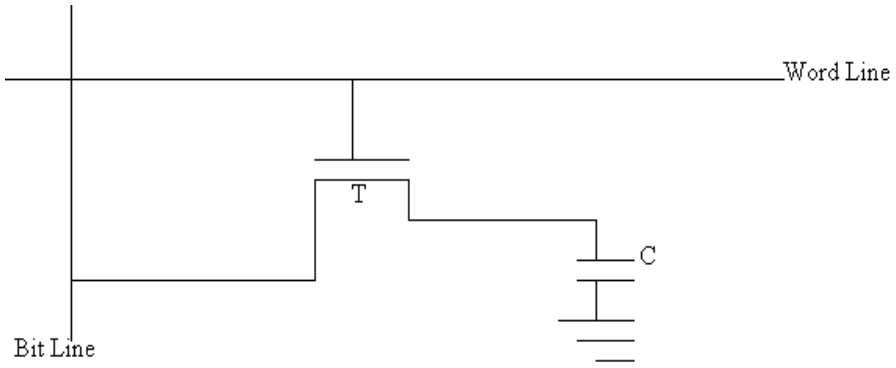
Write operation :

- Transistor T is turned on by activating the word line
- Appropriate voltage is applied to the bit line
- Now, the capacitor is charged to the level of voltage in the bit line



Contd. - Refreshing

- After the transistor is turned off, capacitor begins to discharge due to
  - its leakage resistance
  - transistor conducting a tiny current in the off state
- Hence the need of refreshing, to restore the capacitor's charge
- The information is read correctly only if the cell is read before the charge drops below a threshold value



Contd. - refreshing

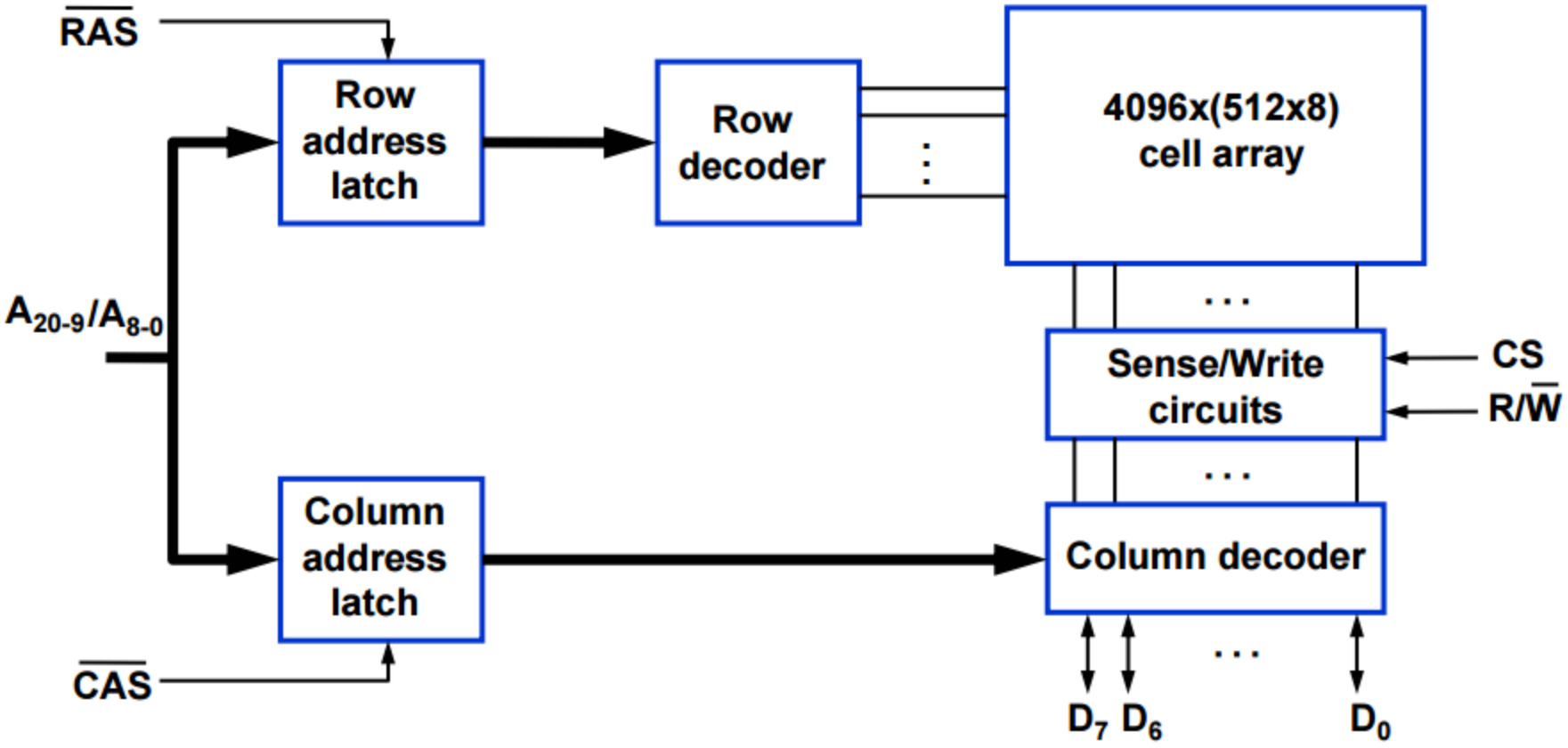
- During Read operation, the sense amplifier connected to the bit line checks whether the charge on the capacitor is above the threshold value.
- If so, a full voltage is placed in the bit line. This charges the capacitor to full voltage, representing a logic 1 .
- Else, voltage on the bit line is made 0, and now the capacitor will have no charge, representing a logic 0.
- Thus, reading a row refreshes all cells in that row

Asynchronous DRAMS

- **Timing of the memory device is controlled by a specialized memory controller circuit which generates RAS and CAS.**
- **No separate clock signal.**
- **This type of DRAM is known as asynchronous DRAM**

2M x 8 asynchronous dynamic memory chip

Explain the operation of a 16 Mega bit DRAM chip configured as 2Mx8 – 10 marks



Contd..

- Cells are organized in 4K x 4K array (4096 x 4096 bits= 16 mega bits)
- 4096 cells in each row are divided into 512 groups of 8 cells each
- So, each row stores 512 bytes
- 12 address bits are needed to address 4096 rows (i.e, 2^{12}=4096)
- 9 bits are needed to address 512 coloumns (2^9=512). A column is a group of 8 bits
- Hence, 21 bit address

Properties

- Read operation refreshes a cell

  Write operation overwrites a cell

- Refreshing is performed automatically

  Each row is accessed periodically (irrespective of read/write operations)

- Asynchronous memory – RAS and CAS govern the timing. No clock signal.

FAST PAGE MODE

- It is an operation to read consecutive columns in a row
- Transferring the bytes in sequential order is achieved by applying the consecutive sequence of column address under the control of successive CAS signals.
- This scheme allows transferring a block of data at a faster rate. The block transfer capability is called as **Fast Page Mode.**

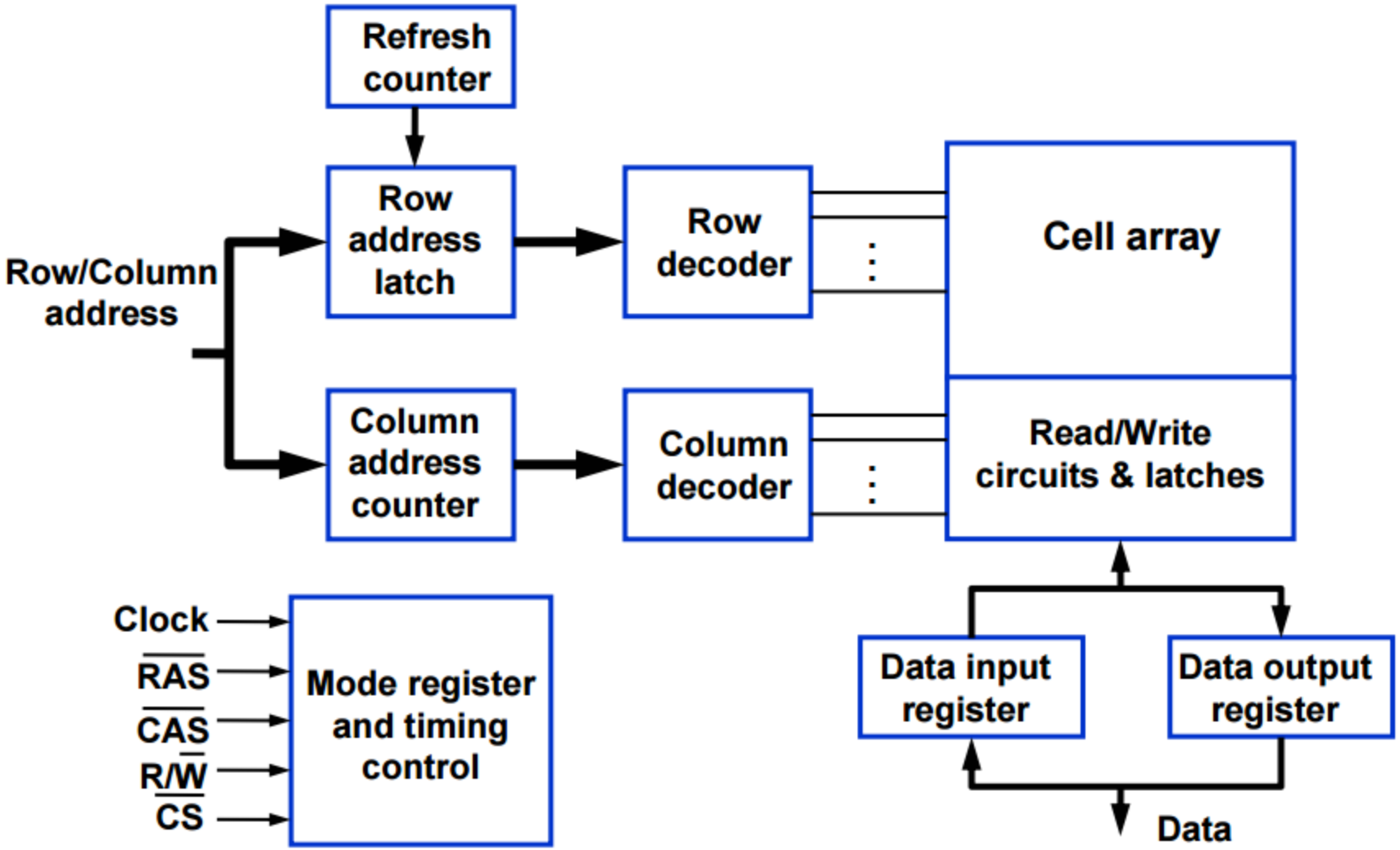
Contd..

- Row and column addresses are multiplexed on 12 pins. (hence no. of pins are reduced)

Read Operation

- row addr is applied. It is loaded to Row Address Latch when RAS (row address strobe) signal is active
- All the cells in the selected row are refreshed
- Now, the column address is applied. It is loaded to Column Address Latch when CAS signal is active

Synchronous DRAMS



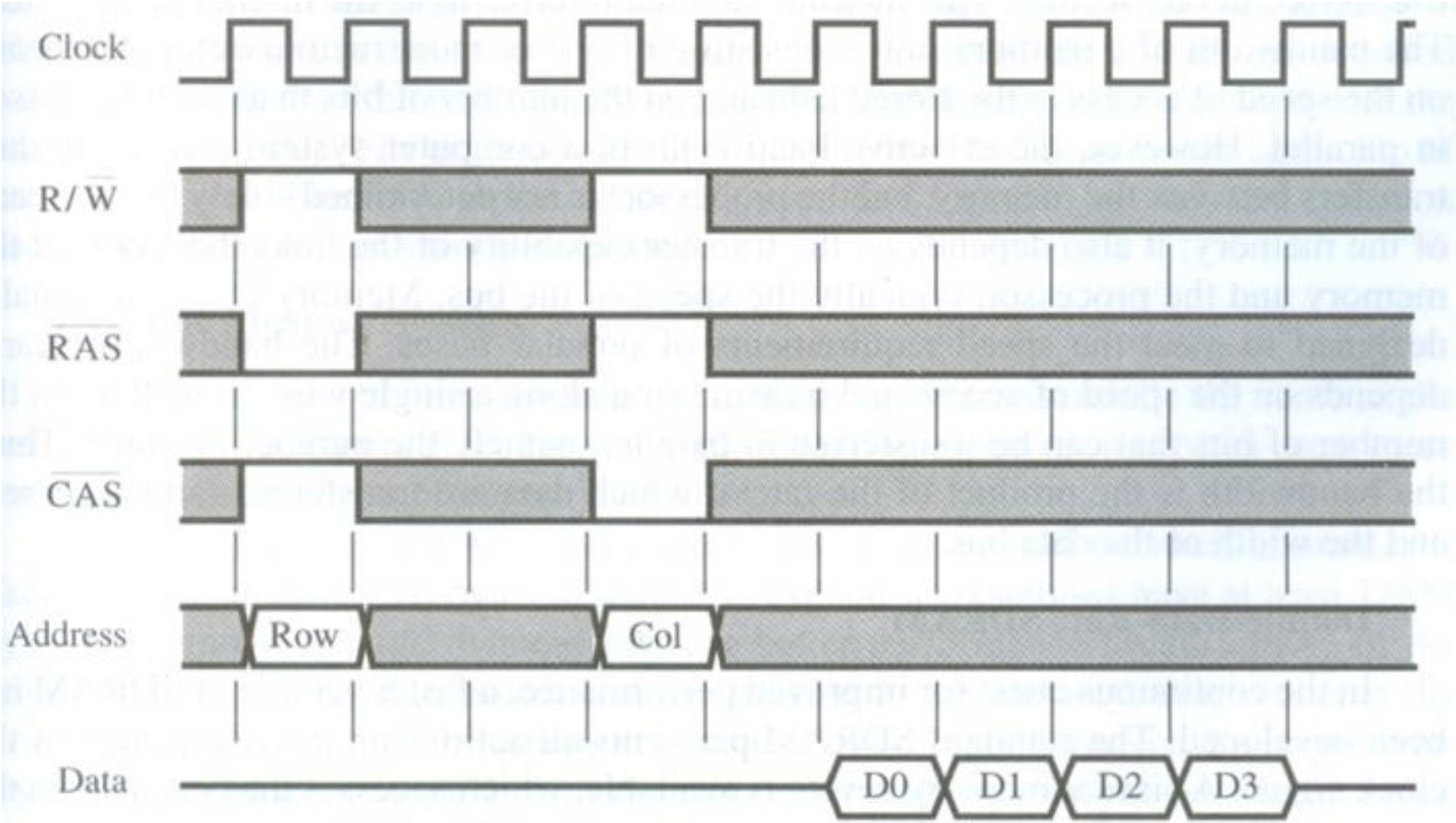
Synchronous DRAMS

- **Operation of SDRAM is directly synchronized with clock signal.**
- The outputs of the Read/Write circuits are connected to latches.
- During a Read operation, the contents of the cells in a row are loaded onto the latches.
- During a refresh operation, the contents of the cells are refreshed without changing the contents of the latches.
- Data held in the latches that correspond to the selected columns are transferred to the output.

Properties of SDRAM

- For a burst mode of operation, successive columns are selected using column address counter and clock.
- CAS signal need not be generated externally.
- A new data is placed during rising edge of the clock
- Mode of operation is programmable using mode register
- Burst mode : to read/write successive columns

Burst read of length 4 in an SDRAM



Burst read of length 4 in an SDRAM

- First ,the row address is latched under control of RAS signal.
- The memory typically takes 2 or 3 clock cycles to activate the selected row.
- Then the column address is latched under the control of CAS signal.
- After a delay of one clock cycle, the first set of data bits is placed on the data lines.
- The SDRAM automatically increments the column address to access the next 3 sets of bits in the selected row, which are placed on the data lines in the next 3 clock cycles.

Latency & bandwidth

- Latency : time taken to transfer a word to or from memory
- In burst operation : latency is the time taken to transfer the first word.

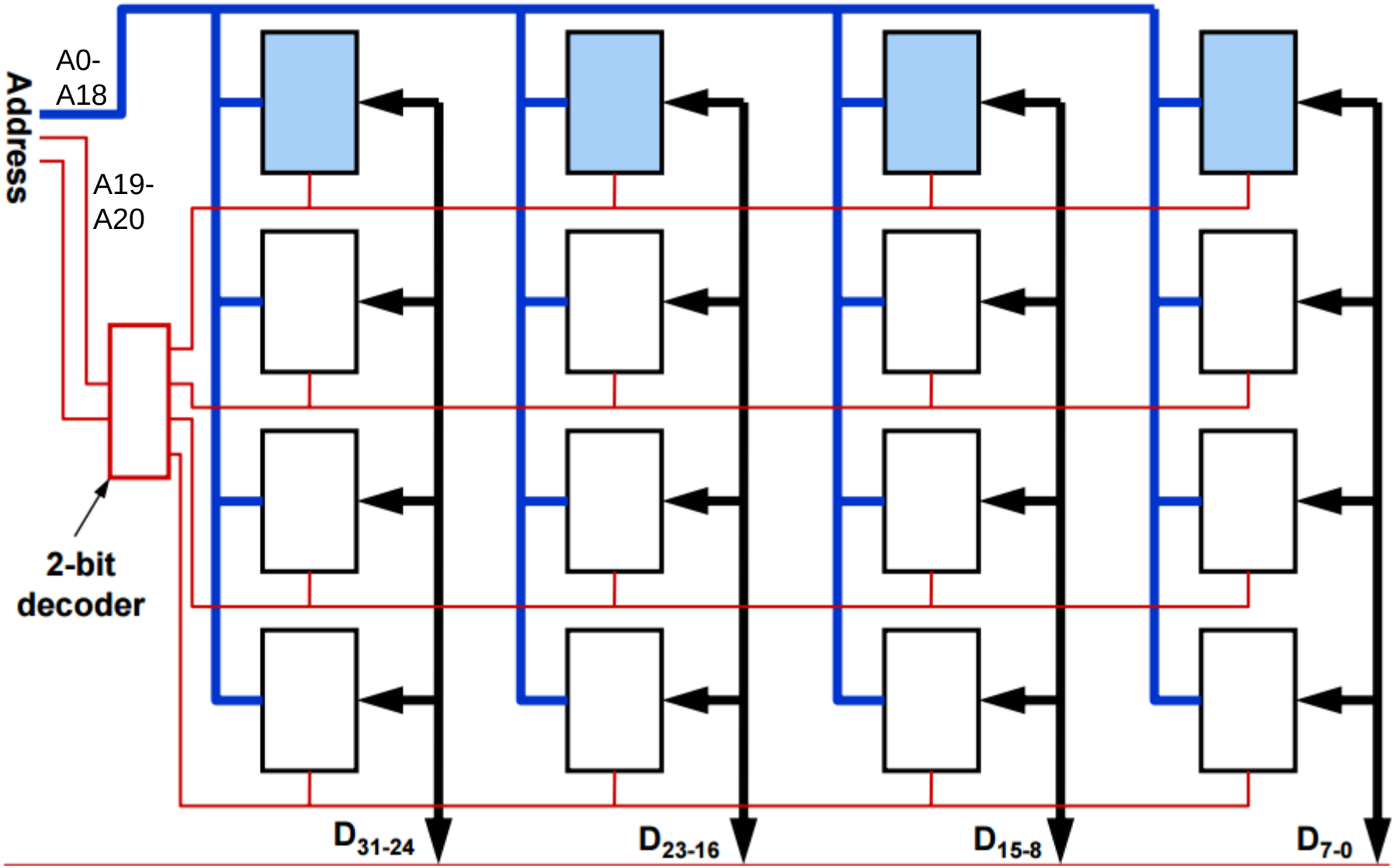
- Bandwidth : bits/second

Double Data Rate SDRAM's

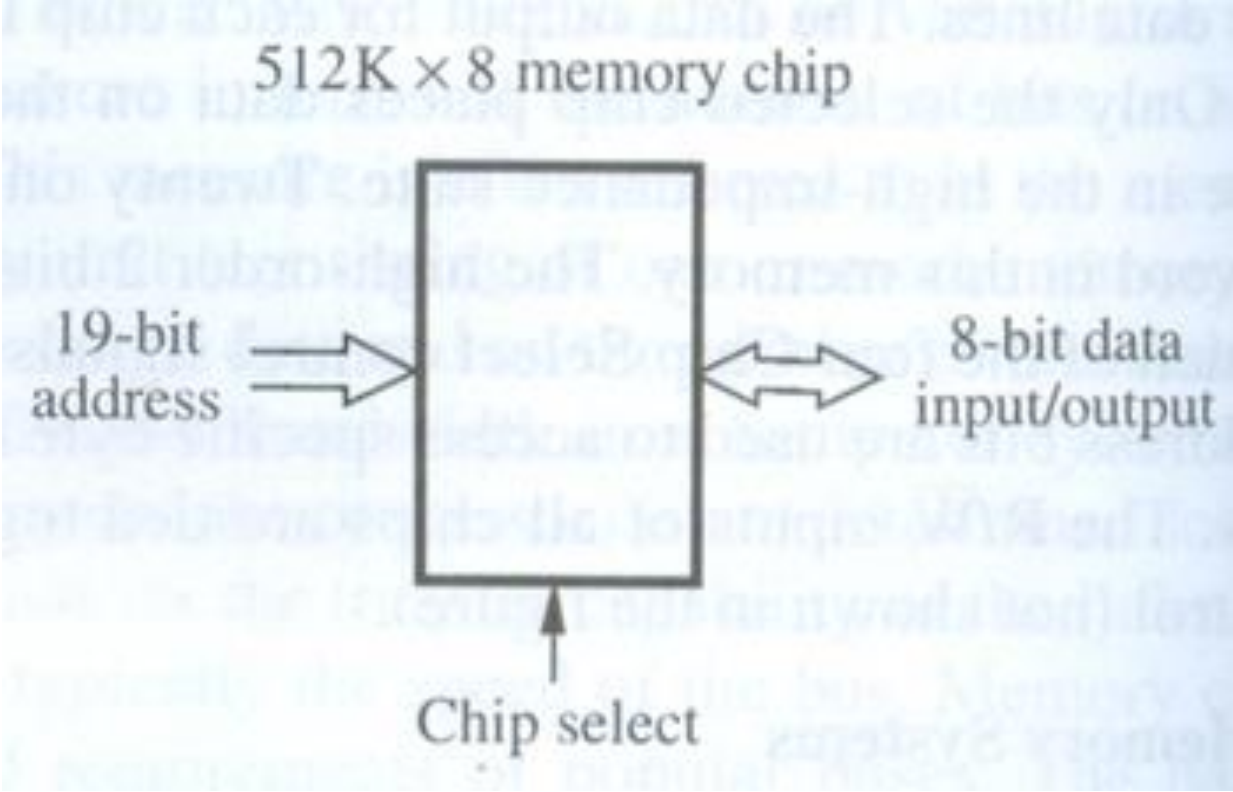
- Transfer of data takes place on both the edge of the clock.
- The band width will be doubled.
- To make it possible the cell arrays must be organized in two banks.
- Used in applications where block transfers are common.

Structure of Larger Memories

2M x 32 memory module using 512K x 8 memory chips – using SRAM chips



512 K x 8 memory chip



Contd…

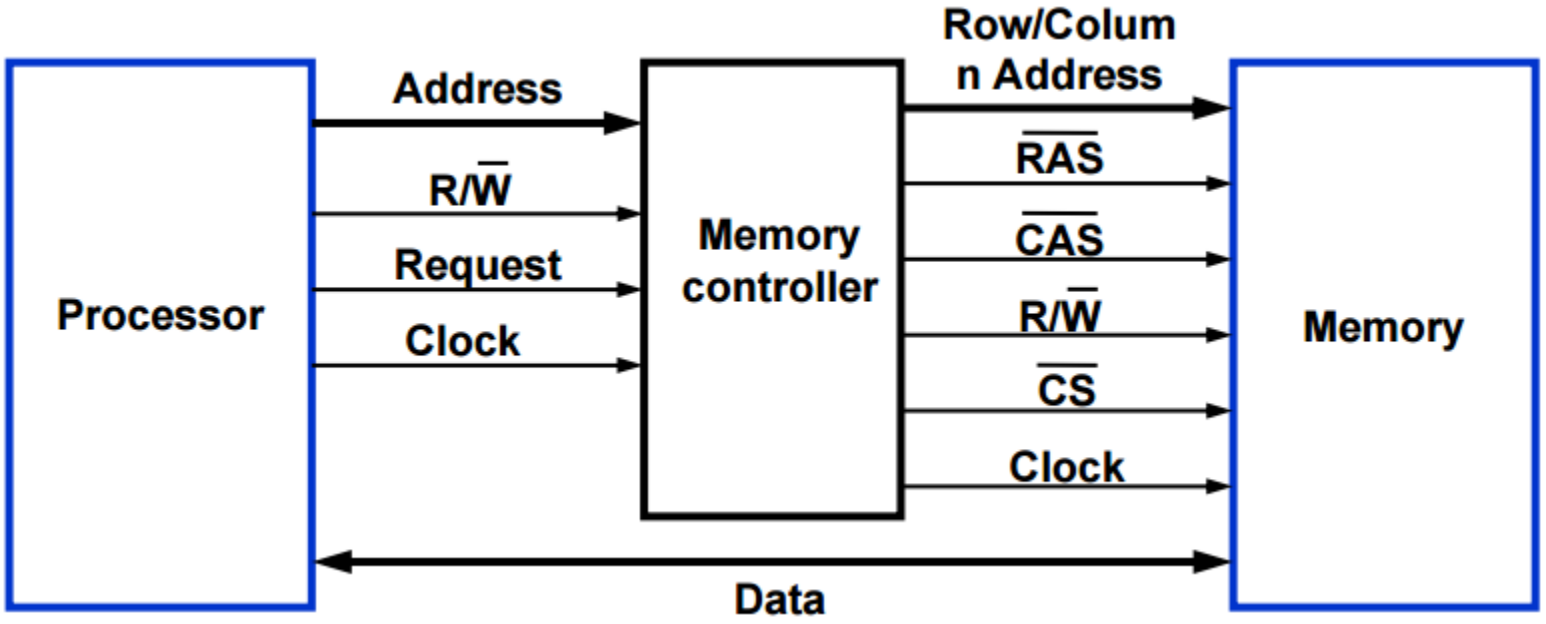
- Each chip has 8 cols. Four chips in a row provide 32 bits.
- Each chip has 512K rows. Chips in 4 rows give 2M rows.
- Thus 2M x 32 memory module
- 2 bits of address (A19 & A20) to select any one row chips (using Chip Select)
- 19 bits (A0 to A18) to address 512K rows in a selected row of chips

Large memory using DRAM chips

- Organization is same as SRAM
- But DRAM chips are packaged in the form of modules
- SIMM and DIMM (single/dual inline memory module)
- Several chips on a small board – plugged on to a socket on the motherboard
- Reduces size of motherboard
- Facilitates future expansion

Memory Controller

- Row and Col addresses are multiplexed to reduce the no. of pins
- Multiplexing is performed by Memory Controller



Refresh overhead

- Refresh period for a typical SDRAM is 64ms
- Example: consider a chip having 8K rows
- Clock cycles needed to access a row is 4
- Then, cycles needed to refresh all rows=8192x4= 32,768 cycles
- If the clock rate is 133MHz,
  - the time needed to refresh all rows =32768/133MHz = 0.246ms.
  - Refresh overhead is 0.246/64= 0.0038
  - Less than 0.4 percent

RAMBUS memory

- Speed of the transfers is not just a function of speed of memory unit. Also depends on speed of the bus.
- More data lines – More speed –widens the bus – increases cost – more space on the mother board
- An alternative is RAMBUS
- RAMBUS: A narrow bus – but faster

Contd..

- Fast signaling – faster transfer of info. btwn chips
- Conventional signal uses - 0 volt for logic 0 and Vcc for logic 1
- More transition time
- RAMBUS signaling:– 0.3v swing above and below a Vref of 2 volts.
- Also called as differential signaling- faster
- Needs special techniques to implement bus lines. Hence fewer lines

contd

RAMBUS specifications:

- Allows 400MHz clock frequency
- Rambus DRAMs : accessing multiple words at a time. Chips organized in multiple banks
- Communication- in the form of packets
  - Request, Acknowledge, data packets
- RAMBUS vs DDRAM – proprietary vs Open standard

ROM

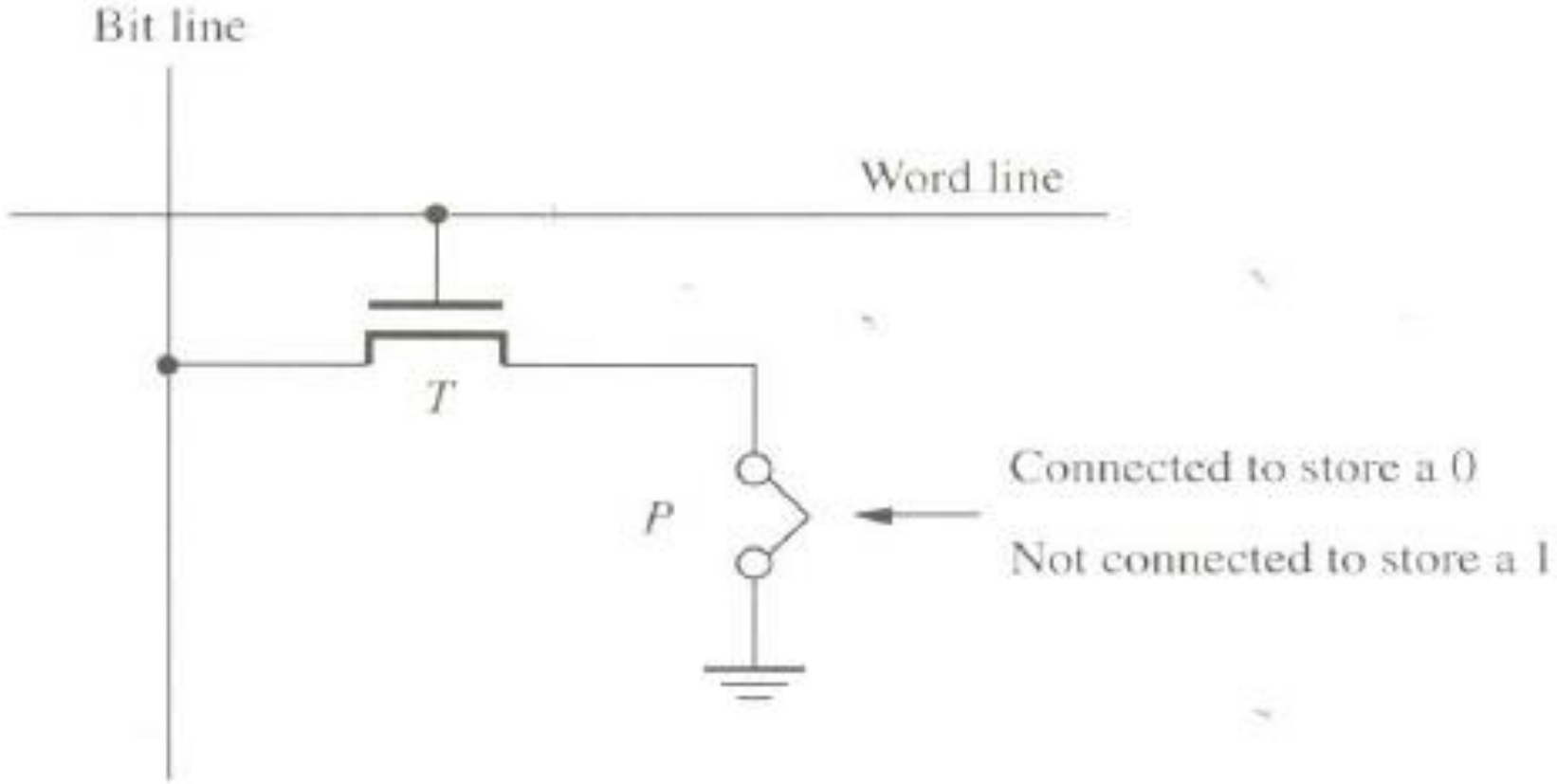
- Data are written during manufacturing

- The need:
  - Eg : **boot** program : Operating System software has to be loaded from disk to memory. This is done by boot program which is stored in non-volatile memory.
  - Non-volatile memory is also used in **embedded** system.

- Since the normal operation involves only reading of stored data, a memory of this type is called ROM.

A ROM CELL

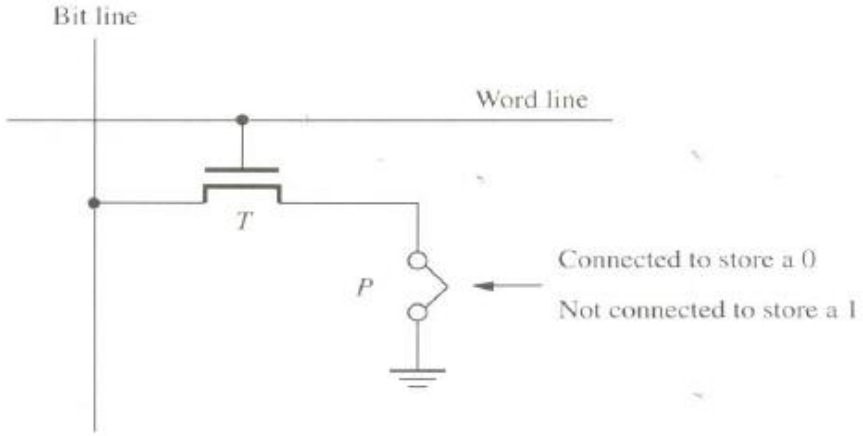
Bit line connected
to power supply



- The bit line is connected to the power supply
- **At Logic value '0' - Transistor(T) is connected to the ground point(P).**
  - Transistor switch is closed & voltage on bitline nearly drops to zero.
- **At Logic value '1' - Transistor switch is open.**
  - The bitline remains at high voltage.

- To read the state of the cell, the word line is activated.

- A Sense circuit at the end of the bitline generates the proper output value.



ROM types

- PROM
  - ROM is expensive when manufactured in small qty. So the PROM

- EPROM – erasable – a special transistor is used at point P in the previous ckt - charges are injected to this transistor - trapped charges make this transistor act as open switch – ie a logical 1 – UV light to erase- ie, dissipates trapped charges, ie a logical 0.

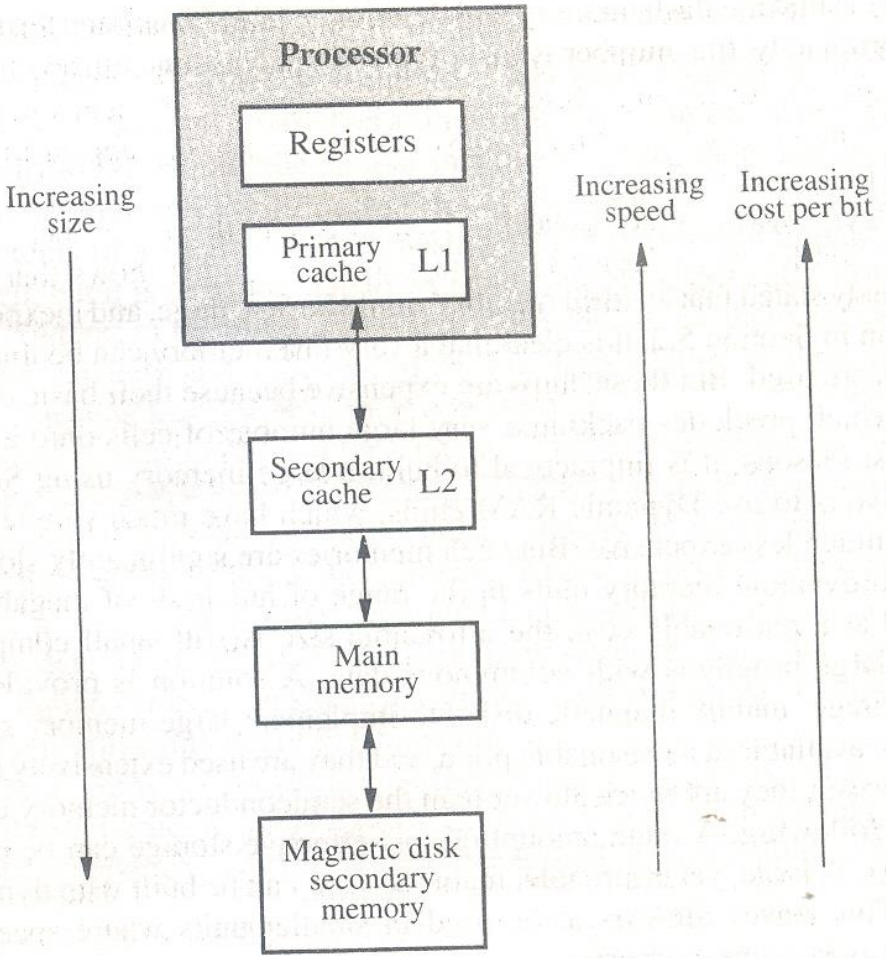
- EEPROM –allows selective erasure

- FLASH memory – similar to EEPROM; reading a single cell; writing a block of cells

Contd..

- Flash cards – larger module of flash chips- up to several GBs
- FLASH drive (pen drive) - to replace HDD – up to giga bytes.
- FASH drive VS HDD.

Speed, size and COST



Cache Memory

- Main memory is very slow compared to the processor
- To reduce the time needed to access the necessary information Cache memory is used
- The cache mechanism is based on the property of programs called **locality of reference**

Contd..

- Most of the execution time of programs is spent on routines in which many instructions are executed repeatedly
- These instructions may constitute a simple loop, nested loops or few procedure that repeatedly call each other

- The main observation is that *many instructions in a few localized areas of the program are repeatedly executed and that the remainder of the program is accessed relatively infrequently*

- This property is called as **Locality of Reference**

Temporal aspect of LOR

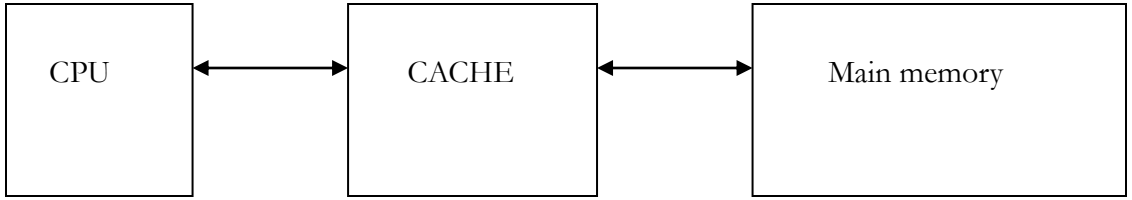
- Temporal aspect means that - recently executed instruction is likely to be executed again very soon

- The above aspect suggests that whenever an information item is first needed, this item should be brought to the cache where it will hopefully remain until it is needed again

Spatial aspect of LOR

- Spatial aspect means that – instructions in close proximity to a recently executed instruction are also likely to be executed very soon
- This suggests that instead of fetching just one item from the main memory to cache, it is useful to fetch several items that reside at adjacent address as well.
- A **block** is a set of contiguous addr. locations

Use of cache



- Read operation- copy the block containing the required word to cache. During program execution, the desired contents are directly read from the cache

Terms…

- Cache mapping – the correspondence between the main memory blocks & cache blocks
- Cache replacement – Done by replacement algorithms.
- Read Hit – the requested word exists in the cache. The requested word is read from the appropriate cache location
- Write Hit - the word to be written exists in the cache.

Contd..

- Two ways to write :
  1. Write-through – cache and main memory locations are updated simultaneously.

     Results in unnecessary memory-write operations when a word is updated frequently in the cache (but is simplest method)

  2. Write-back : main memory is updated later. Update only the cache location, and mark it as updated using dirty or modified bit

     Also results in unnecessary Write operations.

     During write-back all words of the block are written to memory even if a single word has been changed.

Mapping Functions

Assumptions:

- Cache has 128 blocks
- Each block has 16 words
- 4bits to address 16 words

- Main memory has 4K (4096) blocks
- 12 bits to address 4096 blocks

- Total 16 bits in the address field

Direct Mapping

- Mapping function:

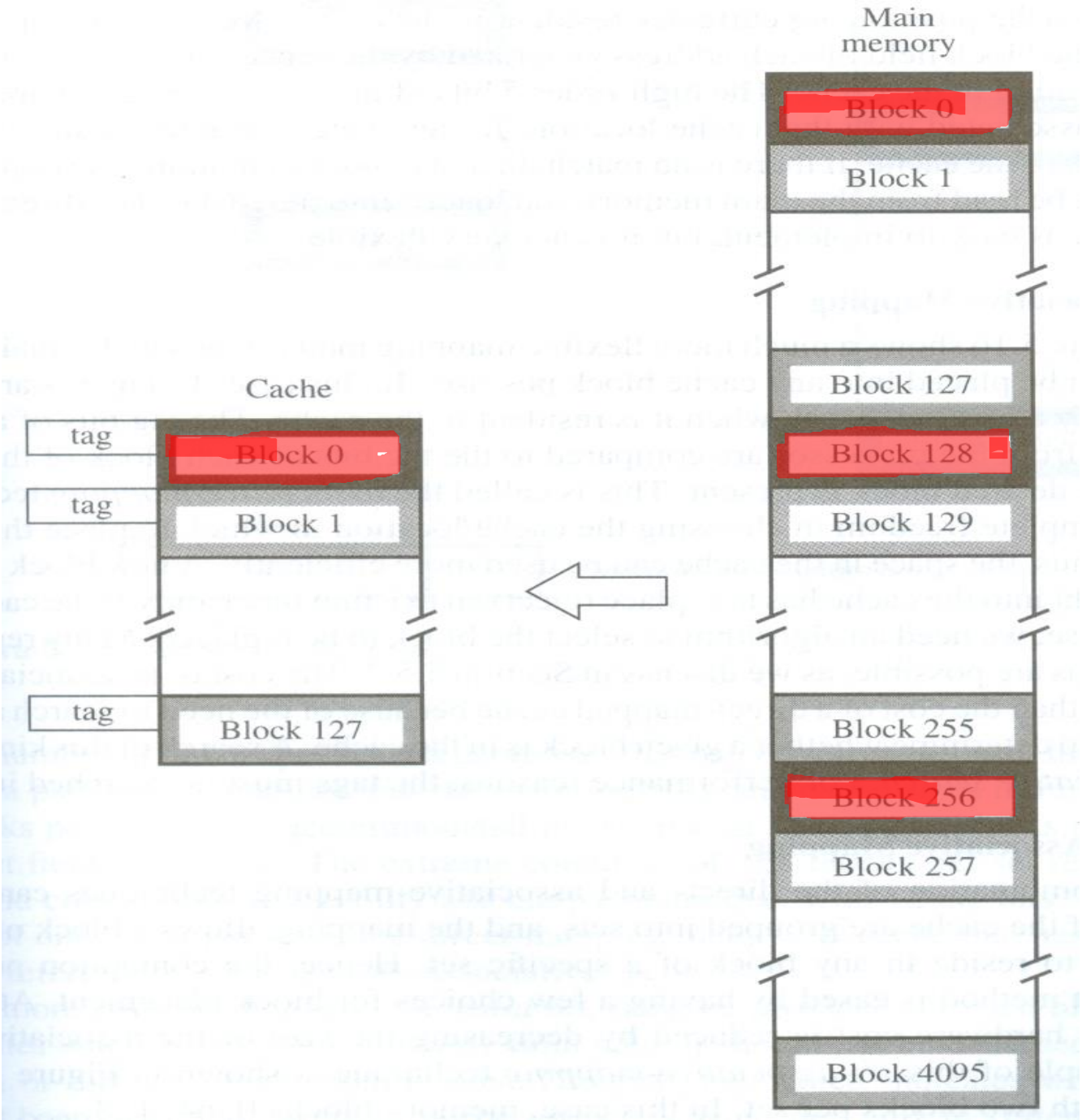
  (MM *Block address*) **MOD** (*Number of blocks in cache*)

- Block j of the main memory maps onto block j mod 128 of the cache.

- Blocks 0, 128, 256,… of main memory mapped to block 0 of cache

- Blocks 1, 129, 257,… of main memory mapped to block 1 of cache

- So, a total of 32 blocks of main memory will be mapped to each cache block (ie 4096/128)

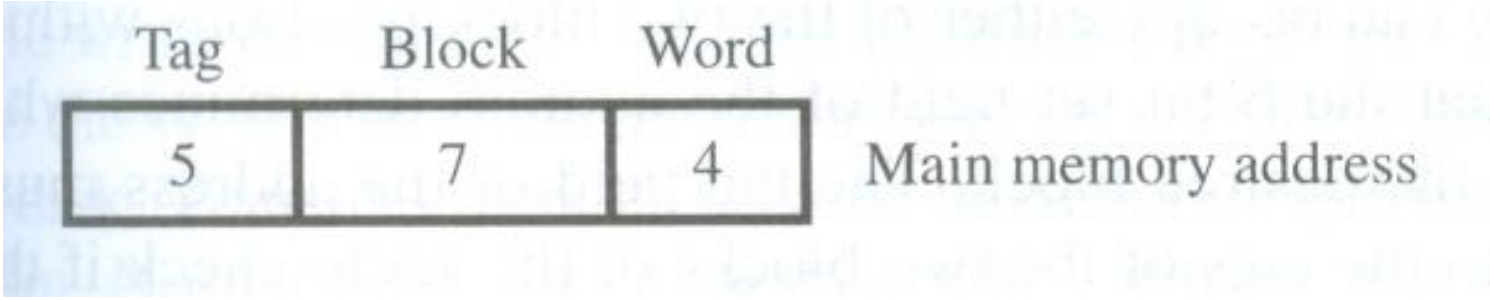

Main memory
Block 0
Block 1
Block 127
Block 128
Block 129
Block 255
Block 256
Block 257
Block 4095
Cache
tag
Block 0
tag
Block 1
tag
Block 127

Contd..

- Hence the contention….

- Uses a replacement algorithm

Placement of a memory block in cache



- 4 bit word address – addresses the required **word** in the block of memory
- 7 bit block address - addresses the cache **block** where this memory block will be stored
- 5 bit tag – Identifies one of the 32 blocks mapped on to a cache block

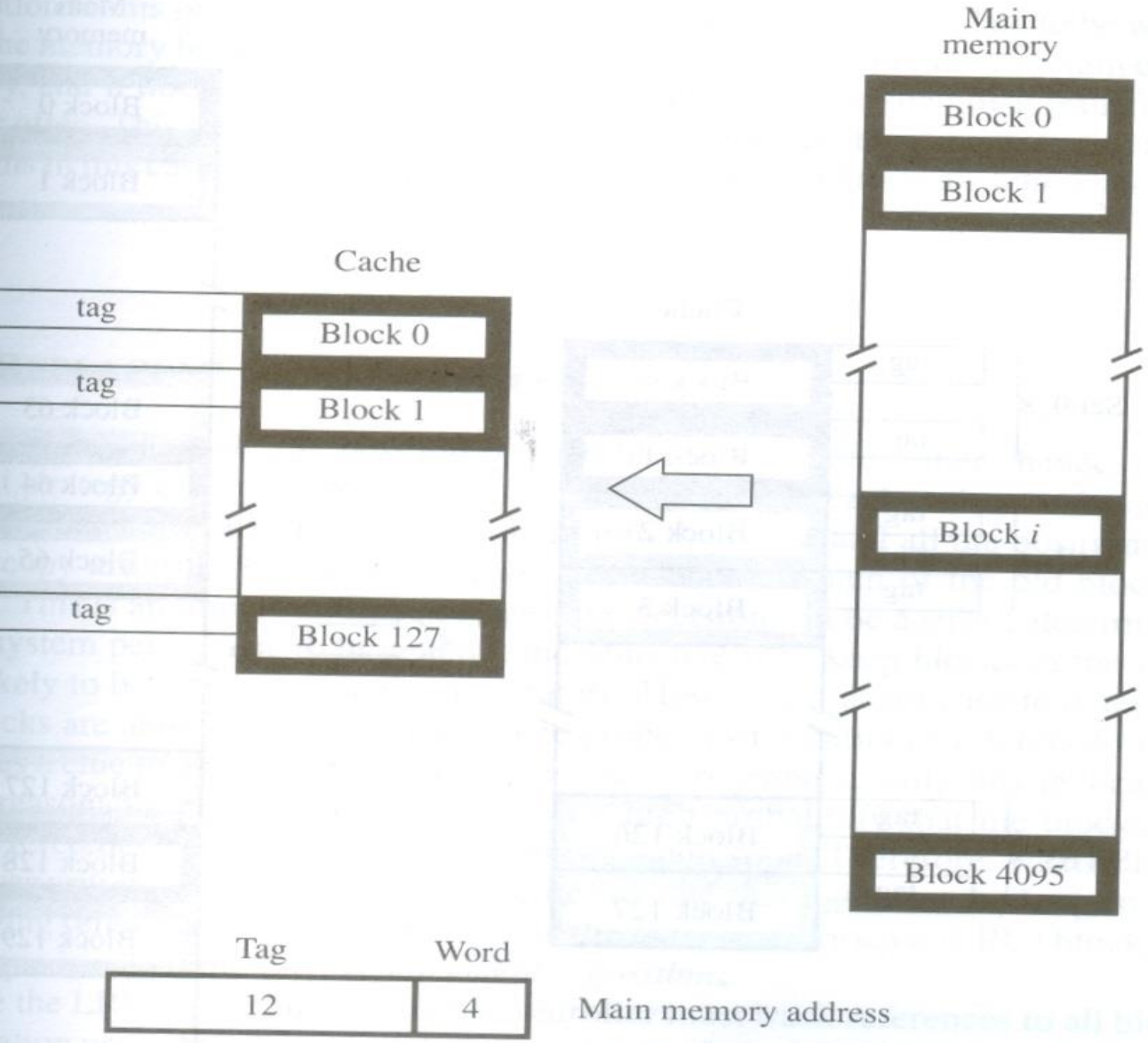
contd..

Accessing from cache:

- The 7 bits of cache block field of address generated by the processor points to a cache block
- Compare the high order 5 bits with the tag bits associated with that cache
- If they match, the desired word is in that block of the cache.
- Else read that block from MM & load into cache
- Easy to implement; very flexible

Associative Mapping

- According to this mapping a memory block can be placed into any cache block
- 12 tag bits to identify 4096 memory blocks
- The tag bits of the Addr received from the processor are compared to the tag bits of each block of the cache.
- Replacement – only if the cache is full
- Cost of searching is more



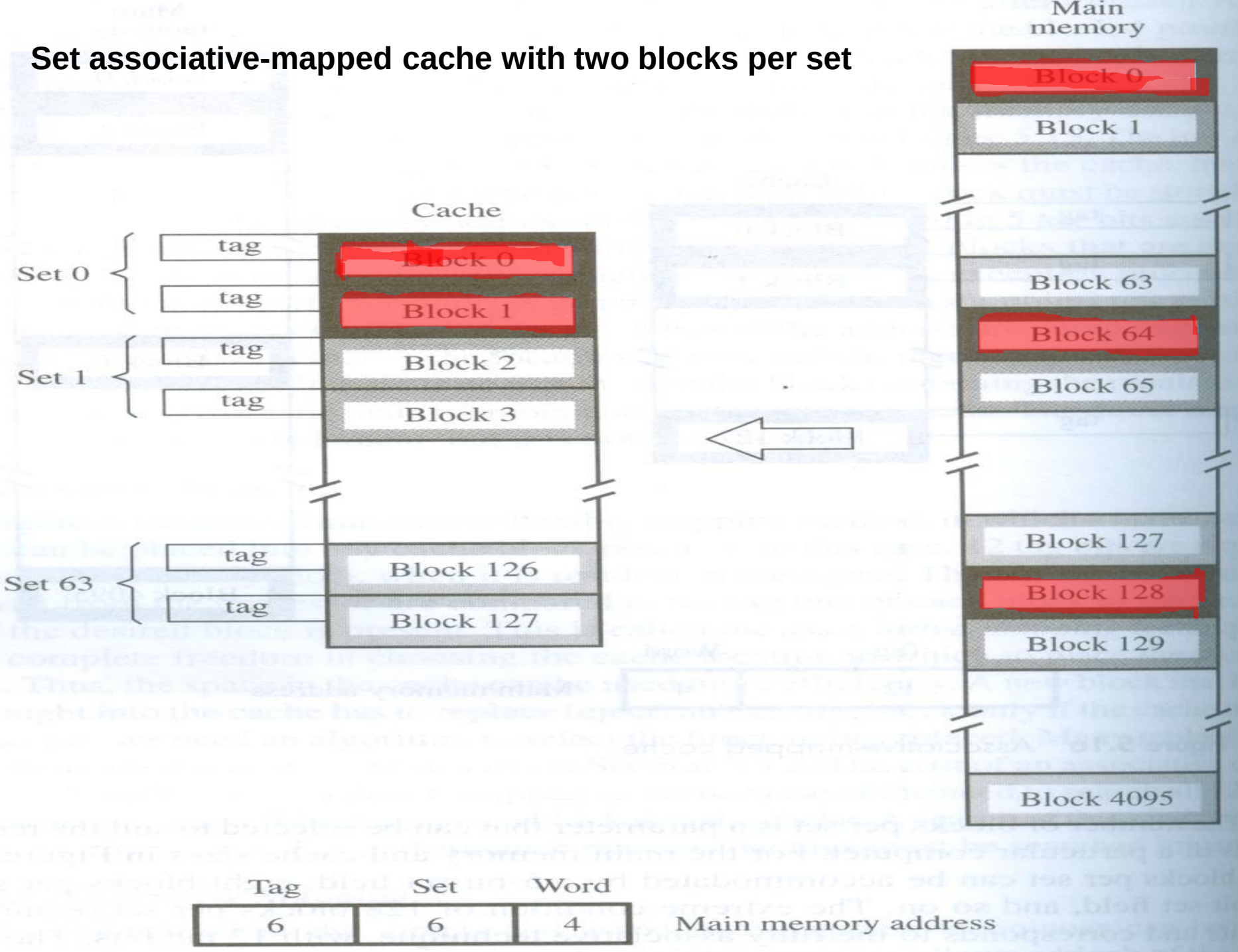
e 5.16 Associative-mapped cache.

Set Associative Mapping

- Blocks of the cache are grouped into sets
- Mapping function:

  (*block address*) **MOD** (*Number of sets in a cache*)
- A main memory block which is mapped to a set can reside in any of the blocks in the set
- Has a choice for block placement. Hence less contention.
- More h/w cost

Set associative-mapped cache with two blocks per set



Contd..

- 64 groups.
- Mapping function – j mod 64
- 0,64, 128, 192, 256… blocks of MM mapped to cache set 0. (And so on…)
- They can occupy either of the two blocks in that set
- 4096/64=64 blocks are mapped to a set . So, 6 bit tag field
- 64 sets. Hence 6 bit set field

Contd…

- Reduces the search and contention.

Replacement Algorithms

- When a new block is to be brought into cache, and all the positions that the new block may occupy are full, the replacement algorithms decide which of the old blocks to overwrite.

- No replacement strategy for direct mapped cache, because position of each block is predetermined

- Strategy exists only for associative and set-associative caches

LRU Replacement Algorithm

- According to the property of LOR, blocks that have been referenced recently will be referenced again soon.
- Therefore, when a block is to be overwritten, overwrite the block that has gone longest time without being referenced
- This block is called as *least recently used block*, and hence the name.

LRU replacement- The method

- Using a counter, track the references to all blocks as the execution proceeds
- Consider a set-associative cache with 4 blocks/set
- A 2 bit counter is used to track the LRU block.
- When a miss occurs and the set is not full, the counter associated with the new block loaded is set to 0 and counters of all other blocks are increased by one.

Contd..

- When the miss occurs and the set is full, the block with counter value 3 is removed. New block is put in its place, and its counter is set 0. The other three block counters are incremented by one.

- When a hit occurs, the counter of the referenced block is set to 0. Counters with values lower than the referenced block are incremented by one .

- LRU algorithm is used extensively.

Performance considerations

What are the key factors that affect the performance and cost of a computer system w.r.t memory ?

- Performance depends on how fast m/c instructions are brought into processor for execution
  - i.e shorter access time ->high performance

- To achieve this, parallelism is used in accessing slower memory

Factors affecting the performance -1

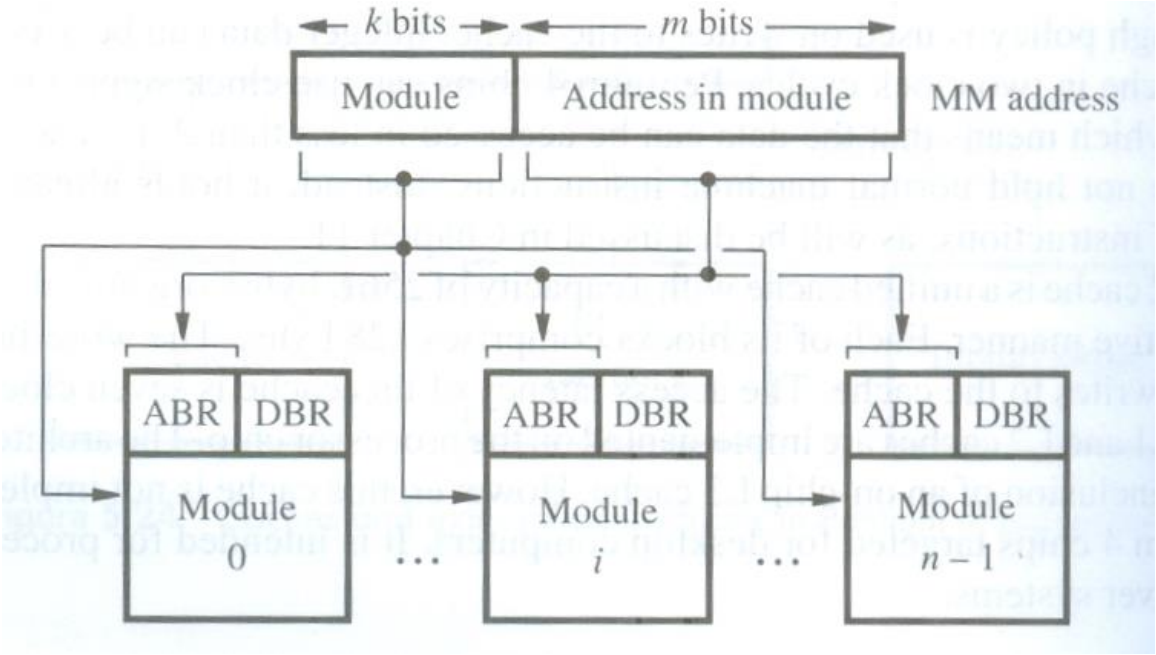
Parallelism using Interleaving

- Main memory of a computer is structured as separate modules.
- Each module has its own Address Buffer Register (ABR) and Data Buffer Register (DBR)
- Advantage - All modules can be accessed simultaneously.

Addressing multiple modules

Method-1 : Consecutive words in a module

- High-order k bits name one of n modules
- Low-order m bits name a word within that module



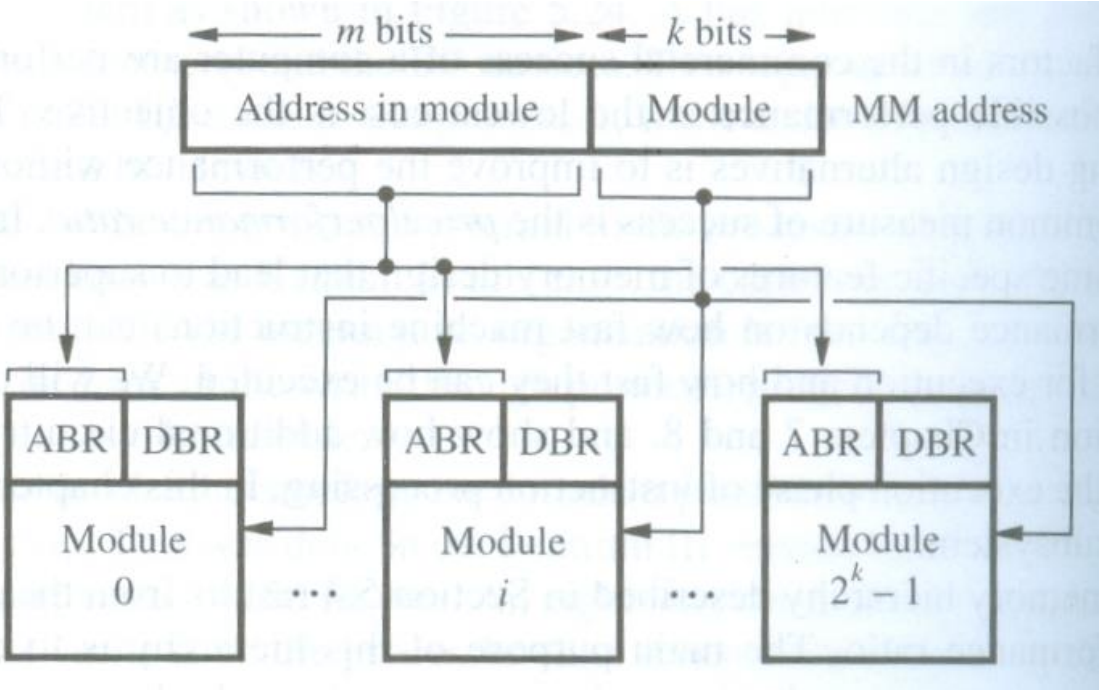
Contd..

- Drawback - When consecutive locations are accessed only one module is involved.

Addressing multiple modules

Method-2 :Consecutive words in consecutive modules

- Also called as *memory interleaving*
- Low-order k bits name a module
- High-order m bits name a word within that module



Contd..

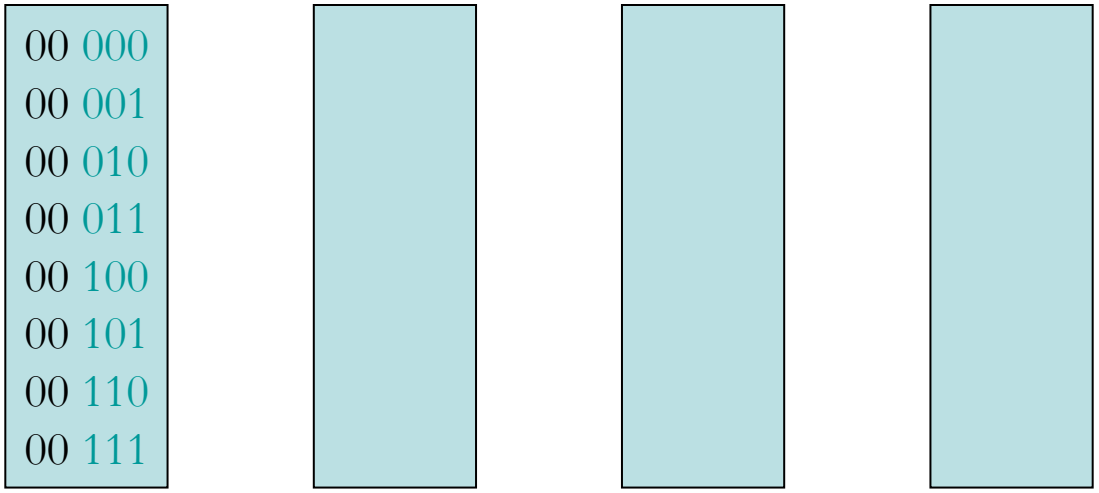
- Consecutive locations are addressed in successive modules
- When consecutive locations are accessed, several module are involved – results in faster access.

Effect of interleaving – An example

- Consider that cache has 8 words per block and No. of modules=4
- On a read miss, the desired word must be brought to cache from main memory
- Assumptions- No. of clock cycles needed to:
  - send an address to main memory = 1
  - access the first word = 8
  - Access each of the subsequent word = 4
  - send the last word to main memory = 1

Contd...

- Let the address of 8 words be 00000 through 00111
- Distribution of words in main memory modules according to the 1st method :



Contd..

- The no. of clock cycles needed to load the desired block into cache is

  1 – to send the address

  \+ 8 – to access the first word

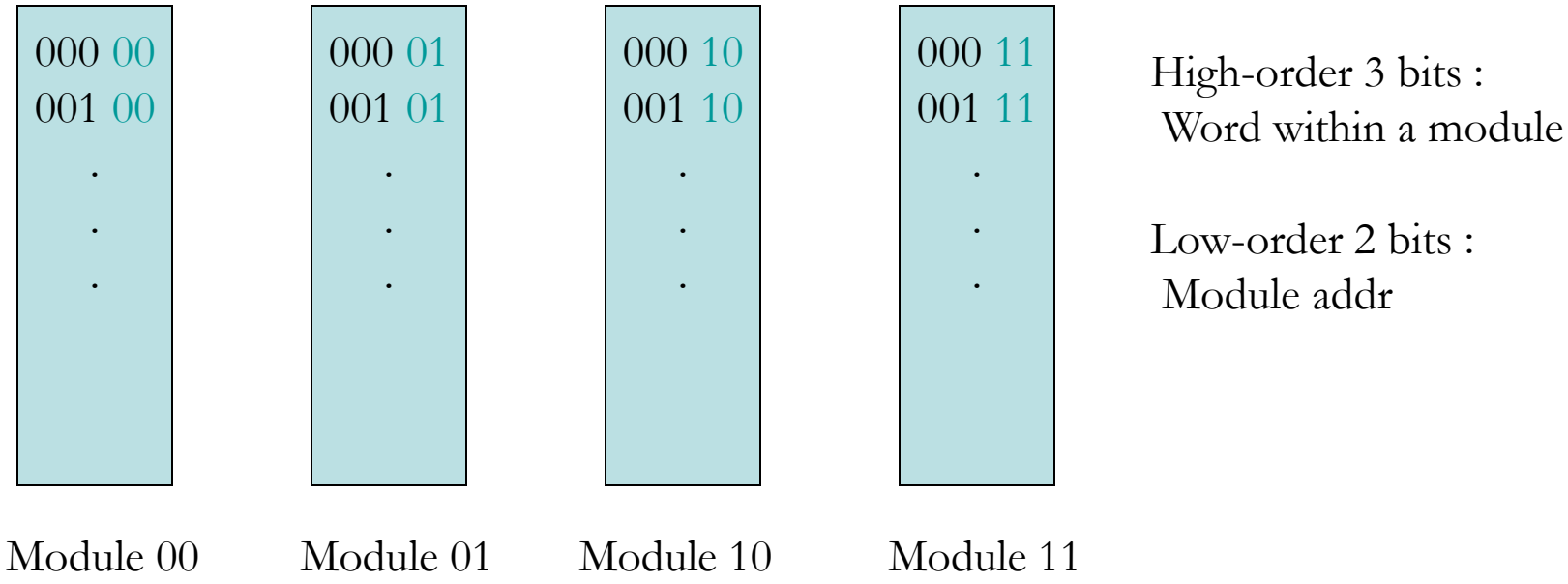
  \+ (7x4) - to access remaining words

  \+ 1 - to send the last word to cache

  = 38 cycles

Contd..

- Distribution of words according to the 2nd method



Contd..

- The no. of clock cycles needed to load the desired block into cache is

  1 – to send the address

  \+ 8 – to access the first word from each module

  \+ (1x4) - to access the second word in each module

  \+ 4 - to send the second word from each moule to cache

  = 17 cycles

Factors affecting the performance -2

Hit Rate & Miss Penalty – The usage of Cache

- Hit - Successful access to data in the cache
- Hit Rate = $\dfrac{\text{No. of hits}}{\text{Total no. of accesses}}$

- Miss Rate = $\dfrac{\text{No. of misses}}{\text{Total no. of accesses}}$

- For a high performance computer, Hit rate must be more than 0.9

Contd…

- Miss Penalty : The extra time needed to bring the desired information into the cache after a cache miss.
- To improve the performance, it is necessary to reduce the miss penalty
- Average access time experienced by the processor $t_{ave} = hC + (1 - h)M$

  Where,

  h – hit rate, (1-h) – miss rate

  M – miss penalty (the time needed to access information from the main memory)

  C – the time needed to access information from the cache

An example for impact of cache on performance

Assumptions:

- No. of instructions executed = 100
- No. instructions that perform data read/write operations = 30
- Hence the total no. of memory accesses=130
- In a system that has no cache,
  - the no. of clock cycles for **each memory read access** = 10

Contd…

- In a system having cache,
  - No. of words per cache block = 8
  - Hit rate for instructions =0.95
  - Hit rate for data =0.9
  - Computer uses interleaved MM
  - The miss penalty M

    = No. of clock cycles needed to **load a block** from interleaved MM to cache

    = 17

Therefore -

$$\text{Improvement in Performance} = \frac{\text{Time without cache}}{\text{Time with cache}}$$

Contd…

- Access Time without cache for 130 accesses= 130x10
- Time with cache = (t_{ave} for accessing 100 instructions)
  + (t_{ave} for accessing 30 data)

$$\text{speedup} = \frac{130 \times 10}{100(0.95 \times 1 + 0.05 \times 17) + 30(0.9 \times 1 + 0.1 \times 1\ 7)} = 5.04$$

- Above result suggests that the computer with the cache performs five times better

Contd.. Ideal cache Vs actual cache

- Hit rate for ideal cache=100%
- Therefore t_{ave} for 130 cache accesses

  = 130

- The relative performance= t_{ave} for actual cache/ t_{ave} for ideal cache

$$= \frac{100(0.95 \times 1 + 0.05 \times 17) + 30(0.9 \times 1 + 0.1 \times 1\ 7)}{130}$$

$$= \underline{1.98}$$

- Above result shows that actual cache is only 2 times slower than the ideal cache.

Contd.. Processor clock Vs System bus clock

- A high performance processor uses a separate clock which is much faster than system bus clock
- On-chip cache uses processor clock
- Main memory uses system bus clock.

An example to illustrate the effect of using separate clocks:

Assumptions:

System bus clock is four times slower than the processor clock

Contd..

- No. of words in a cache block=8
- Miss penalty (time needed to bring a MM block to cache) in terms of no. of *system bus clock* cycles = 15
- Miss penalty in terms of no. of *processor clock* cycles= 15 x 4 =60
- No. of *system bus clock* cycles needed to access a word from the MM = 9
- The above value in terms of *processor clock* cycles = 9 x 4 =36

Contd..

- No. of *processor clock* cycles needed to access a word from the cache= 1

$$\text{Improvement in Performance} = \frac{\text{Time without cache}}{\text{Time with cache}}$$

$$= \frac{130 \times 36}{100(0.95 \times 1 + 0.05 \times 60) + 30(0.9 \times 1 + 0.1 \times 60)} = 7.77$$

- Thus, the cache that uses processor clock improves the performance.

Caches on the processor chip

- Cache can be an external chip to processor
- During data transfer betwn processor and a chip, delay is introduced in the driver and receiver gates of the chip
- To avoid it, cache is placed on the processor chip
- Size of the cache is limited by the availability of space in the processor chip

Contd..

- High-performance processors use two levels of caches
- A smaller and faster, L1 cache on the processor chip – ensures shorter access time – size in Kilo bytes
- Larger and slower L2 cache, external to processor chip – ensures high hit rate – size in Mega bytes
- L2 cache affects only the miss penalty of the L1 cache

Contd..

- Average access time experienced by the processor with two level caches is :

$$t_{ave}=h_1C_1 + (1-h_1)h_2C_2+(1-h_1)(1-h_2)M$$

where,

h1 – hit rate in L1 cache

h2 – hit rate in L2 cache

C1- access time in the L1 cache

C2 -access time in the L2 cache

M - access time in the memory

- If h1 & h2 > 90% then no. of misses will be less than 1 % of total memory accesses

Example

If h1=0.8, h2=0.9, out of 100 accesses how many accesses are satisfied at each level of cache?

- $t_{ave}=h_1C_1 + (1-h_1)h_2C_2+(1-h_1)(1-h_2)M$
  $=0.8 \times C1+0.2 \times 0.9 \times C2 + 0.2 \times 0.1 \times M$

  $=0.8C1 + 0.18C2 + 0.02M$

Level / hit rate	Accesses made	Accesses satisfied	Accesses reach the next level
L1 - 0.8	100	80	20
L2 – 0.9	20	18	2
MM - 1	2	2	Nill

Virtual Memory

- Main memory of a computer is not as large as its address space.

- For example, a processor that uses 32 bit address has an address space of 2^{32}=4Gbytes. But memory in a typical computer ranges from few hundred mega bytes to 1G bytes.

- Large programs cannot be fitted completely into main memory for execution.

Contd..

- Therefore, only those parts of the program which are currently required for execution are stored in main memory
- The remaining parts of it are stored on secondary storage device like hard disk.
- When a new block is needed for execution it is brought to main memory from hard disk. If main memory is full the new block replaces an already existing block.

Contd..

- Techniques that are used for moving pgm and data blocks into main memory are called virtual-memory techniques

- Virtual or Logical addresses :
  - The addresses of instruction and data ***used in a program***

- Processor issues these logical addresses for instruction and data

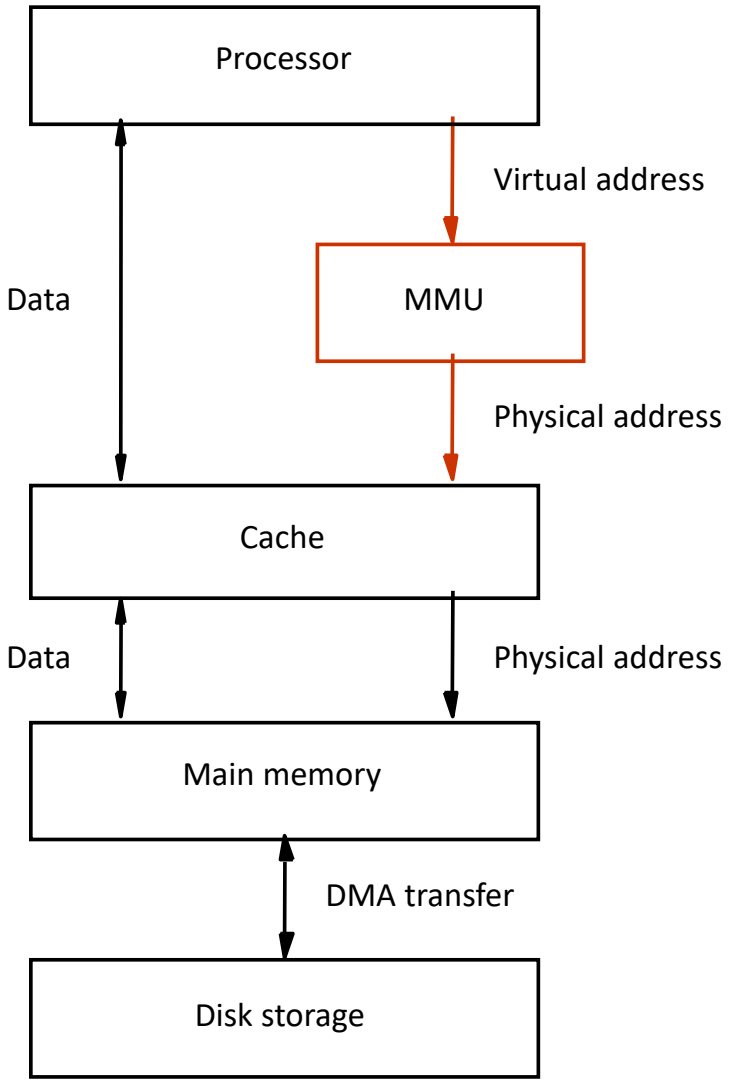
Contd..

- Physical addresses :
  - Addresses of main memory locations (physical memory) where instructions and data are stored after a program is loaded to main memory for execution

- Address Translation :
  - Translating virtual addresses into physical addresses
  - This is needed to access the instruction or data of the program in the main memory
  - MMU (Memory Management Unit) a h/w unit does the translation

Virtual Memory Organization

- When the desired data (or instructions) are in the main memory, these data are fetched.
- If the data are not in the main memory, MMU causes the operating system to bring the data into the main memory from the disk



Address Translation

- Assume that all programs and data in the secondary storage are made up of fixed length units called ***pages***
- A page contains block of words that occupy contiguous locations in the main memory
- Page length may vary from 2KB to 16 KB
- Page is the basic unit that is moved between main memory disk

Contd..

- Main memory is made up of fixed length units called ***page frames***
- Size of a page and page frame are equal

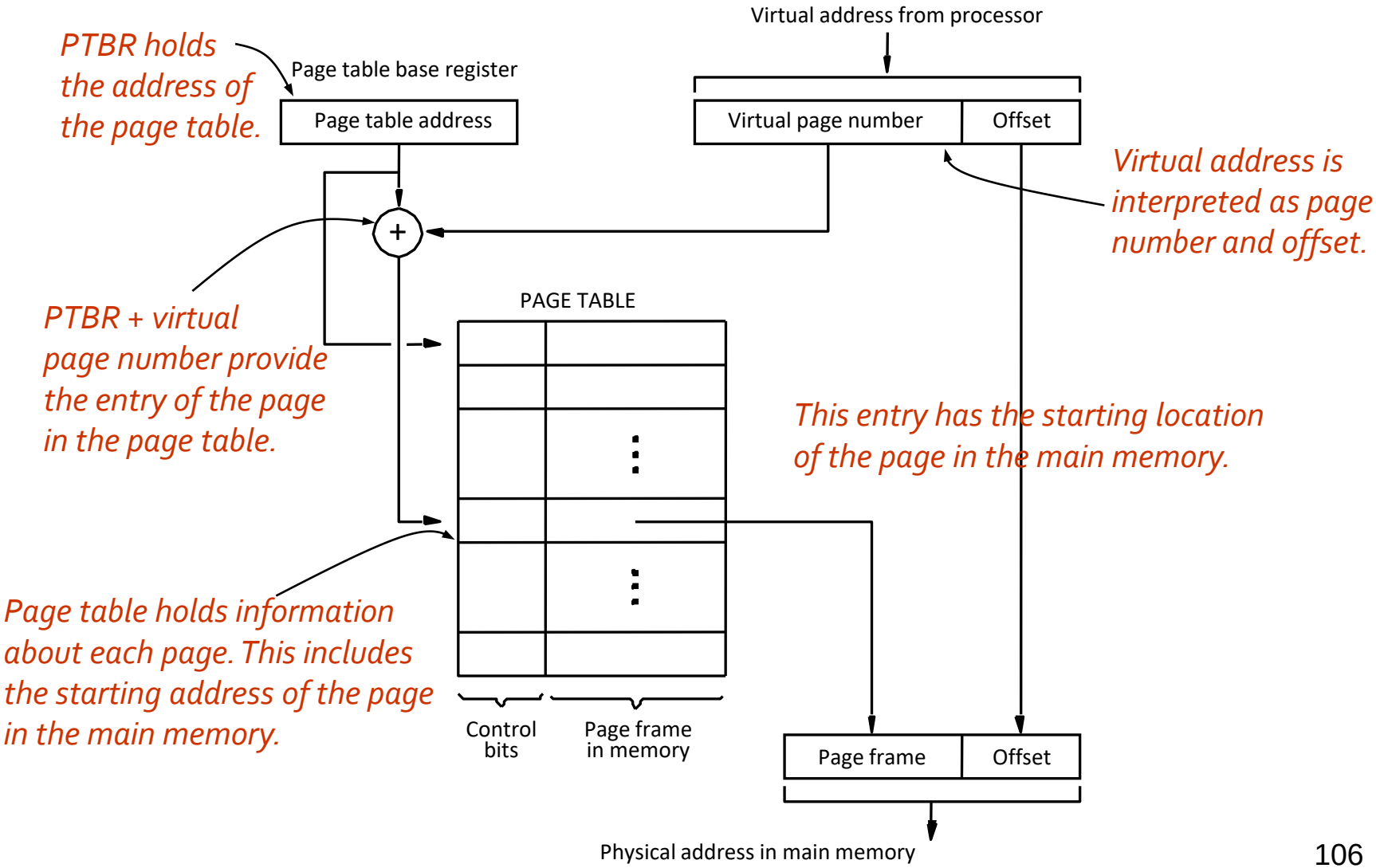
- ***Page table*** – holds
  - Address of a page in the main memory
  - Current status of the page
- First row of the page table contains entry for first page of the program, 2nd row for second page and so on..

- ***Page Table Base Register*** – holds the starting address of the page table in main memory

Contd..

- Virtual address – generated by processor for instruction or data access
  - High-order bits represent page number
  - Low-order bits represent offset , ie. Location of a byte within a page
- Address of the entry for a page in the page table =contents of the PTBR + virtual page number
- An entry in the page table gives :
  - the starting address of the page in the main memory (if the page currently resides in the main memory).
    - By adding the offset to this starting address the ***physical address*** is generated

Virtual memory address translation



Contd..

– some control bits that describe the status of the page, such as –
  - valid / invalid bit – indicates whether the page is actually loaded in the main memory. An invalid page is one that is present in memory but not accessible
  - dirty bit – indicates whether the page has been modified. Modified page must be written back to disk before it is removed from main memory
  - Access control bit – gives read and write or read only permission to a program

Contd..

- Page table is used by MMU for every read and write access. Therefore it is suitable to keep it in MMU (cache) for faster access. Since page table is too large it is kept in main memory

- But a copy of small portion of the page table is kept in a small cache.

- This small cache is called as Translation Lookaside Buffer (TLB), which consists of entries corresponding to most recently accessed pages

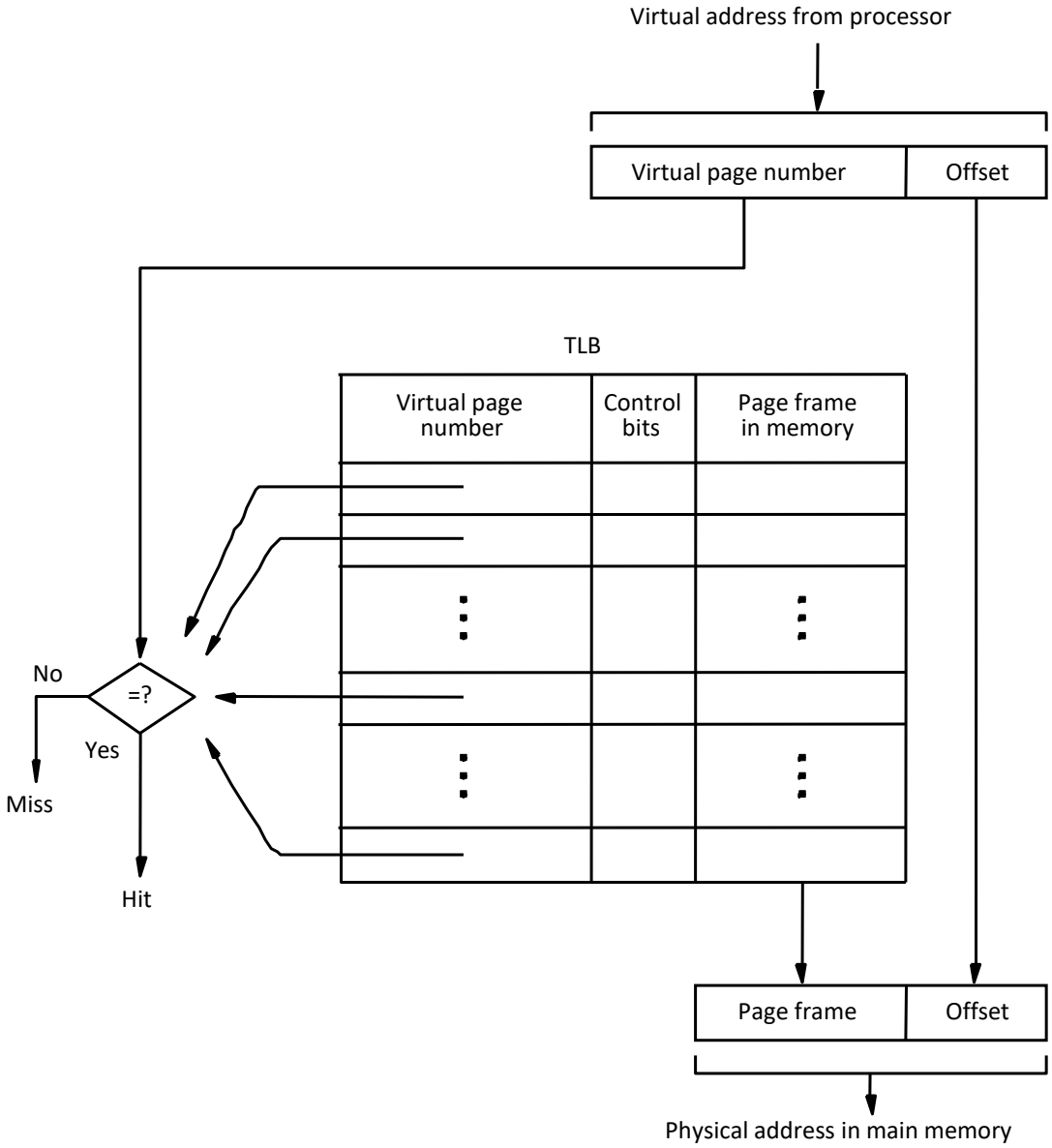
Contd..

- Every entry in the TLB also includes the virtual page number along with frame address and control bits
- Associative mapping technique is used for storing the entries in the TLB. (ie, an entry for a page can be stored in any row of the TLB)

Contd..

- Address translation using TLB
  - Given a virtual address, the MMU looks in the TLB for the referenced page
  - If the page table entry for this page is found in the TLB, the physical address is obtained immediately
  - If there is a miss in the TLB, then the required entry is obtained from the page table in the main memory and this entry is also copied to TLB
- If the required page is not found in the main memory also then a *page fault* occurs. Now the whole page must be brought from the disk into main memory

Virtual memory system with TLB



Cache and VM – The difference

- Cache bridges the speed gap btwn processor and main memory
- Cache is implemented in hardware

- Virtual memory mechanism bridges the size and speed gaps btwn the main memory and secondary storage
- VM is the combination of hardware and software components

End of Unit 4

Acknowledgements

Internet

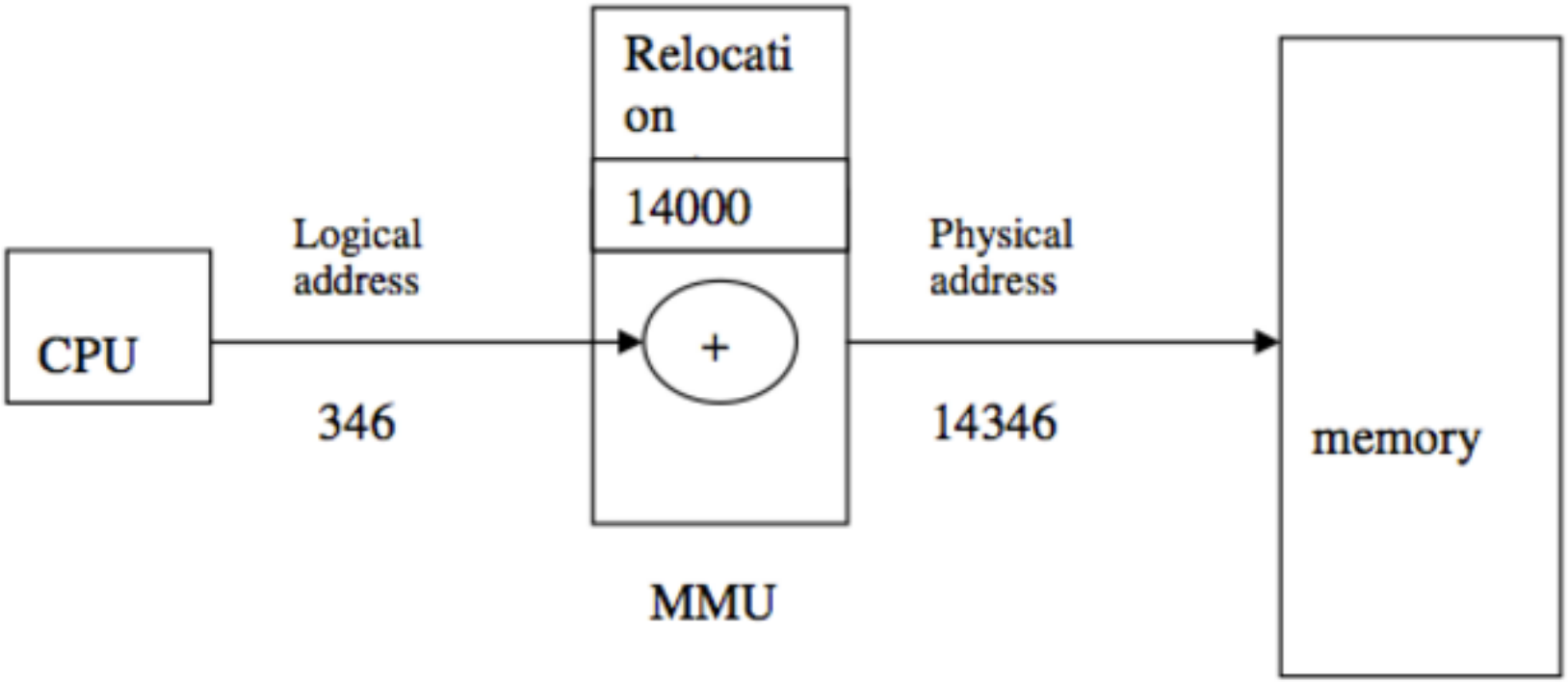
PPT compiled/prepared by:

Dr. Raju K., Assoc. Prof.

CSE Dept, NMAMIT

Line	Loc	Source statement			Object code
5	0000	COPY	START	0	
10	0000	FIRST	STL	RETADR	140033
15	0003	CLOOP	JSUB	RDREC	481039
20	0006		LDA	LENGTH	000036
25	0009		COMP	ZERO	280030
30	000C		JEQ	ENDFIL	300015
35	000F		JSUB	WRREC	481061
40	0012		J	CLOOP	3C0003
45	0015	ENDFIL	LDA	EOF	00002A
50	0018		STA	BUFFER	0C0039
55	001B		LDA	THREE	00002D
60	001E		STA	LENGTH	0C0036
65	0021		JSUB	WRREC	481061
70	0024		LDL	RETADR	080033
75	0027		RSUB		4C0000
80	002A	EOF	BYTE	C'EOF'	454F46
85	002D	THREE	WORD	3	000003
90	0030	ZERO	WORD	0	000000
95	0033	RETADR	RESW	1	
100	0036	LENGTH	RESW	1	
105	0039	BUFFER	RESB	4096	
110		.			
115		.	SUBROUTINE TO READ RECORD INTO BUFFER		
120		.			
125	1039	RDREC	LDX	ZERO	040030
130	103C		LDA	ZERO	000030
135	103F	RLOOP	TD	INPUT	E0105D
140	1042		JEQ	RLOOP	30103F
145	1045		RD	INPUT	D8105D
150	1048		COMP	ZERO	280030
155	104B		JEQ	EXIT	301057
160	104E		STCH	BUFFER,X	548039
165	1051		TIX	MAXLEN	2C105E
170	1054		JLT	RLOOP	38103F
175	1057	EXIT	STX	LENGTH	100036
180	105A		RSUB		4C0000
185	105D	INPUT	BYTE	X'F1'	F1
190	105E	MAXLEN	WORD	4096	001000
195		.			
200		.	SUBROUTINE TO WRITE RECORD FROM BUFFER		
205		.			
210	1061	WRREC	LDX	ZERO	040030
215	1064	WLOOP	TD	OUTPUT	E01079
220	1067		JEQ	WLOOP	301064
225	106A		LDCH	BUFFER,X	508039
230	106D		WD	OUTPUT	DC1079
235	1070		TIX	LENGTH	2C0036
240	1073		JLT	LOOP	381064
245	1076		RSUB		4C0000
250	1079	OUTPUT	BYTE	X'05'	05
255			END	FIRST	

Relocation


Relocati
on
14000
Logical
address
Physical
address
CPU
+
346
14346
memory
MMU