CPE300: Digital System Architecture and Design

Fall 2011

MW 17:30-18:45 CBC C316

Memory System Design

11212011

http://www.egr.unlv.edu/~b1morris/cpe300/

Outline

- Review Pipelining
- Memory Components
- Memory Boards
- Memory Modules

Pipelining

- Process of issuing a new instruction before the previous one has completed execution
  - Favorite technique for RISC processors
  - Hide latency of instruction execution (multiple clock cycles for a single instruction)
- Goal to keep equipment busy as much of the time as possible
  - Total throughput may be increased by decreasing the amount of work done at a given stage and increasing the number of stages (simple tasks to accomplish instruction execution)
- Consequences for fetch-execute cycle
  - Previous instruction not guaranteed to be completed before next operation begins
  - Results of previous operation not free available at next operation

Pipeline Hazards

- Deterministic events that are a side-effect of having instructions in pipeline
  - Parallel execution
  - Instruction dependence – instruction depends on result of previous instruction that is not yet completely executed
- Two categories of hazards
  - Data hazards – incorrect use of old and new data
    - RAW in SRC – data not written when needed in pipeline
  - Branch hazards – fetch of wrong instruction on a change in the `PC`
    - Use branch prediction to minimize

Dealing with Hazards

- Pairs of instructions must be considered to detect hazards
  - Data is normally available after being written to a register
- Hazard detection
  - Require minimum spacing between dependent instructions (restrictive)
- Data correction
  - `nop` bubbles to delay pipeline
  - Detect dependence in pipeline and forward data to needed stage as soon as available (without waiting for write)

Restrictions After Forwarding

1. Branch delay slot
   - Instruction after branch is always executed no matter if the branch succeeds or not
2. Load delay slot
   - Register loaded from memory cannot be used as operand in the next instruction
   - Register loaded from memory cannot be used as a branch target for the next 2 instructions
3. Branch target
   - Results register of alu or ladr instruction cannot be used as a branch target by next instruction

```
br r4
add . . .
```

```
ld r4, 4(r5)
nop
neg r6, r4
```

```
ld r0, 1000
nop
nop
br r0
```

```
not r0, r1
nop
br r0
```

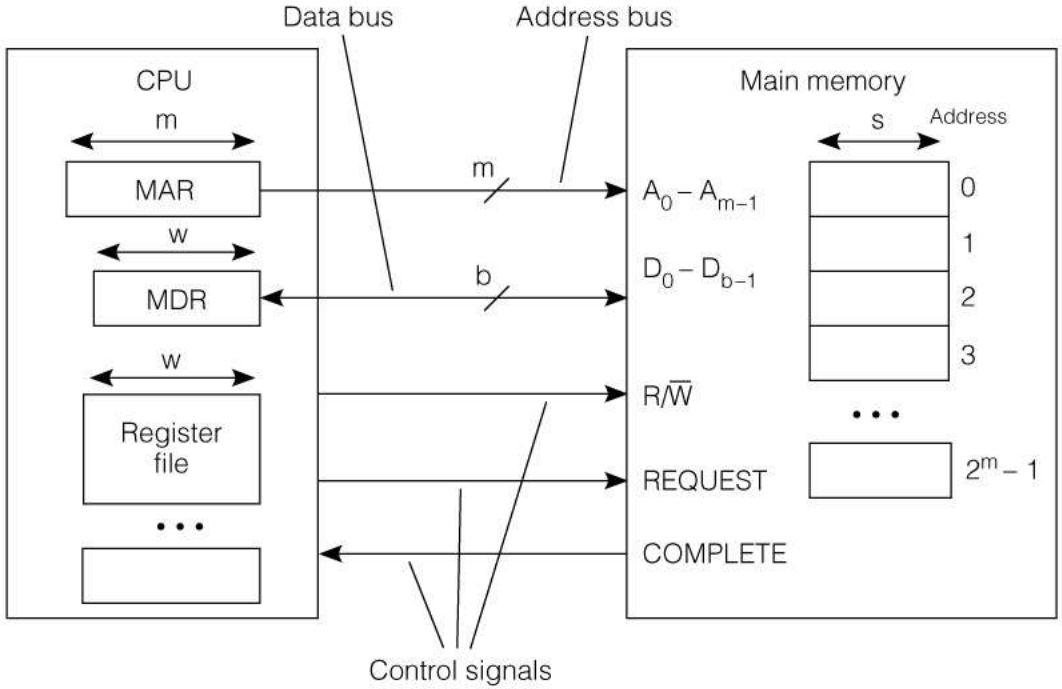
Chapter 7

- RAM Structure: Cells and Chips
- Memory Boards and Modules
- Two-Level Memory Hierarchy
- Cache
- Virtual Memory
- Memory as Computer Sub-System

Memory System Design

- Memory has been treated as an array of words limited in size only by number of address bits
- Real world design issues include:
  - Cost
  - Speed
  - Size
  - Power consumption
  - Volatility
  - Etc.
- These affect how a memory system is designed

CPU-Memory Interface



- Read sequence
  1. CPU loads `MAR`, issues `Read` and `REQUEST`
  2. Main memory transmits words to `MDR`, asserts `COMPLETE`
- Write Sequence
  1. CPU loads `MAR` and `MDR`, assert `WRITE` and `REQUEST`
  2. Value in `MDR` written to address in `MAR`
  3. Main memory asserts `COMPLETE`

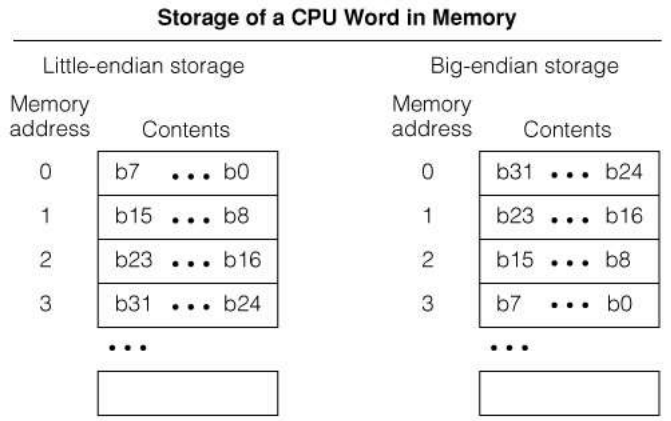
- w = CPU word size
- m = bits in memory address
- s = bits in smallest addressable unit (e.g. 1 byte = 8-bits)
- b = data bus size

- If b<w (bus size smaller than word),
  - main memory must make w/b b-bit transfers
- Some CPUs allow reading and writing of words sizes <w
  - 16-bit words but 16 or 8 bit values can be read or written (word or half word)
- `COMPLETE` signal could be omitted if memory is fast or has a predictable response
- `Read` and `Write` (`R/W`) lines are sometimes separate
  - Omit `REQUEST`

Word-Ordering Endian-ness

- Data types have word size larger than smallest addressable unit
  - Little endian – little end first
    - Least significant portion of word stored at lowest address
  - Big endian – big end first
    - Most significant portion of word stored at lowest address

	CPU Word			
CPU word bit locations	b31 . . . b24	b23 . . . b16	b15 . . . b8	b7 . . . b0
Big-endian byte addresses	0	1	2	3
Little-endian byte addresses	3	2	1	0



RAM and ROM

- RAM – random access memory
  - Memory cells can be accessed in equal time
  - Read/write semiconductor memory
- ROM – read-only memory
  - Programmed memory that can only be read
  - Also is random access
- Random access is compelling
  - Access independent of location within memory
  - Contrast with a disk that is dependent on current location of read-write head and location of data on disk (e.g. platter, sector)

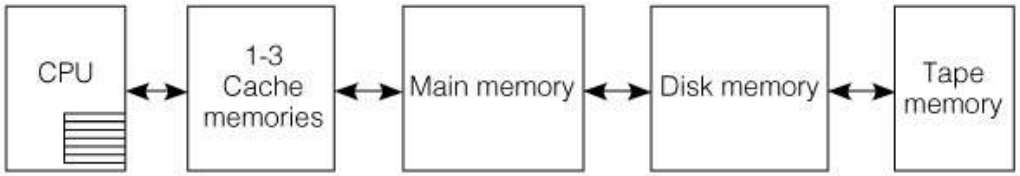
Memory Performance Parameters

Symbol	Definition	Units	Meaning
t_a	Access time	time	Time to access a memory word
t_c	Cycle time	time	time from start of read to start of next
k	Block size	words	Number of words per block
ω	Bandwidth	words/sec	Word transmission rate
t_l	Latency	time	Time to access first of sequence of words
$t_{bl} = t_l + k/\omega$	block access time	time	Time to access entire block from start of read

- Information often stored and moved in blocks at cache and disk level

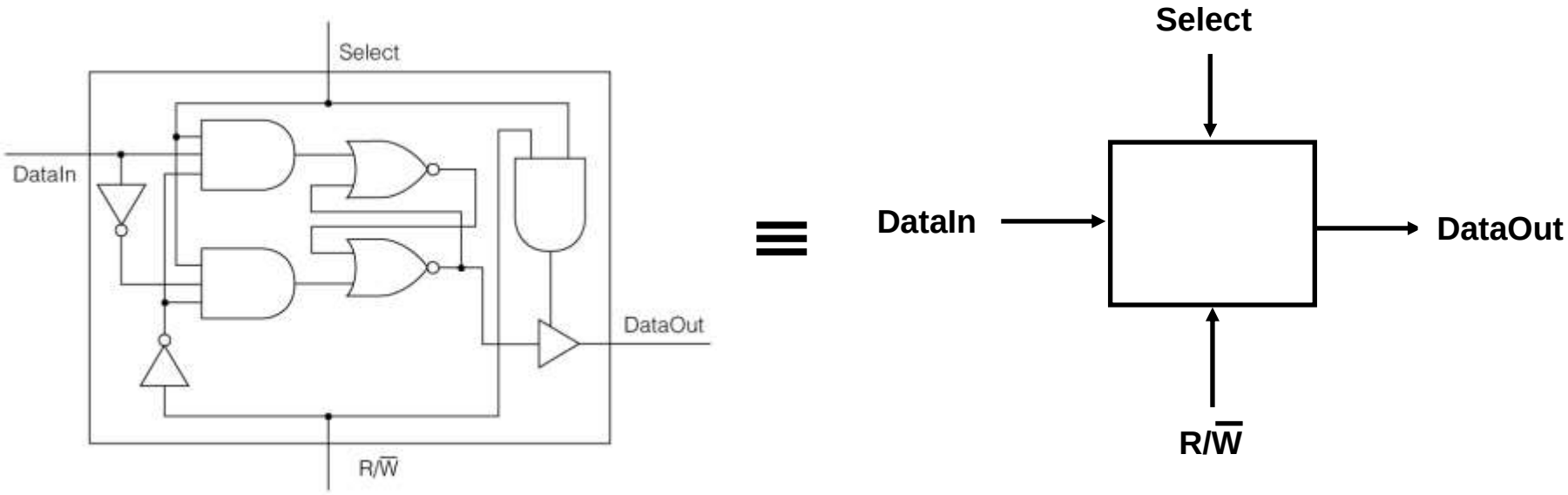
Memory Hierarchy, Cost, Performance

1. Registers – internal to CPU
2. Cache levels
3. Main memory

Component					
Access type	Random access	Random access	Random access	Direct access	Sequential access
Capacity, bytes	64–1024	8 KB–4 MB	64 MB–2 GB	10–200 GB	1 TB
Latency	.4–10 ns	0.4–20 ns	10–50 ns	10 ms	10 ms–10 s
Block size	1 word	16 words	16 words	4 KB	4 KB
Bandwidth	System clock rate	system clock rate - 80 MB/s	10–4000 MB/s	50 MB/s	1 MB/s
Cost/MB	High	$10	$0.25	$0.002	$0.01

Conceptual Structure of Memory Cell

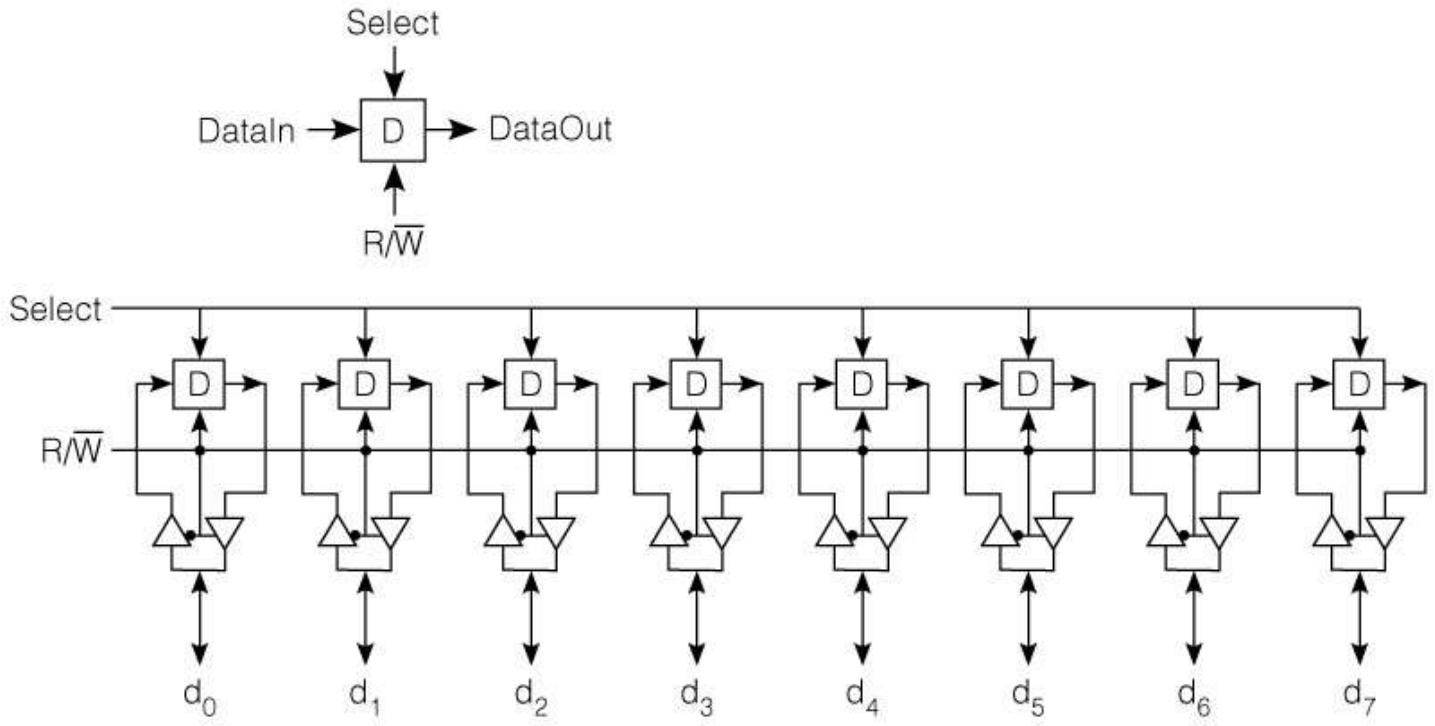
- RAM cell must provide four functions
  - Select, DataIn, DataOut, and R/W



Cross-coupled gates for memory unit – not a practical design

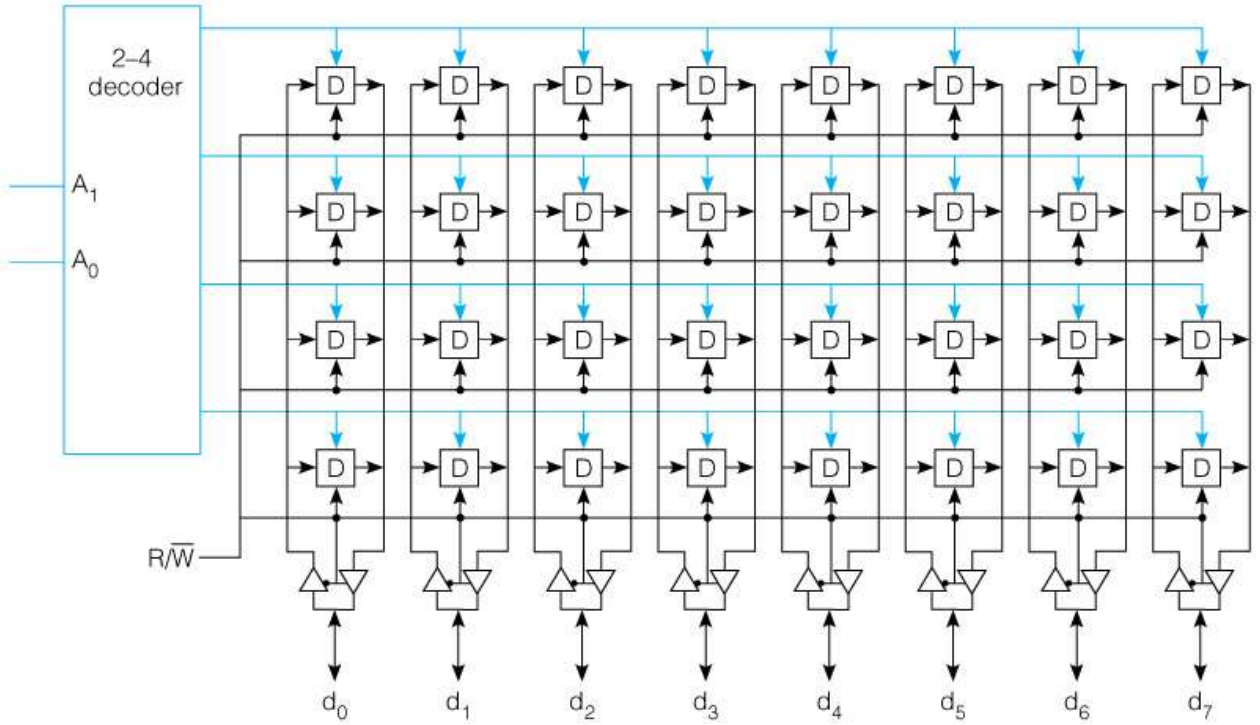
8-Bit Register as 1D RAM Array

- Combine smaller cells into array
  - Single select line used for entire register
  - Single R/W line

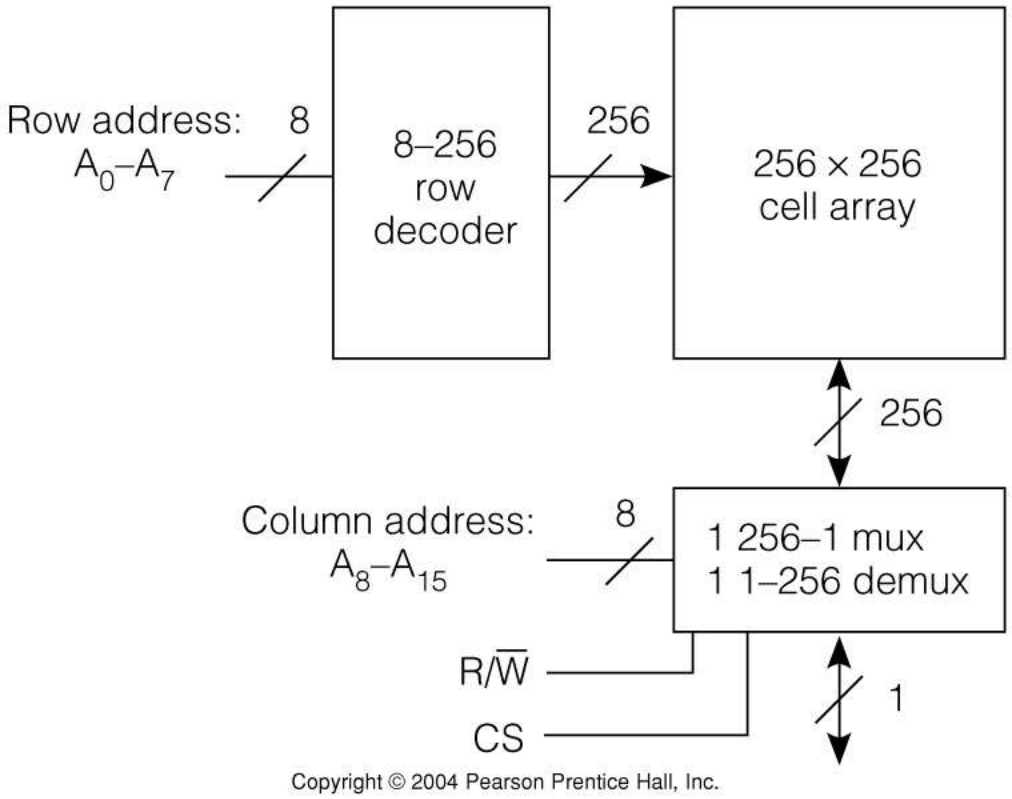


2D Memory Cell Array

- Larger array structure
  - Easy to design with modern layout tools
- Use address decoder to select a register row
  - 2-4 decoder uses two address bits

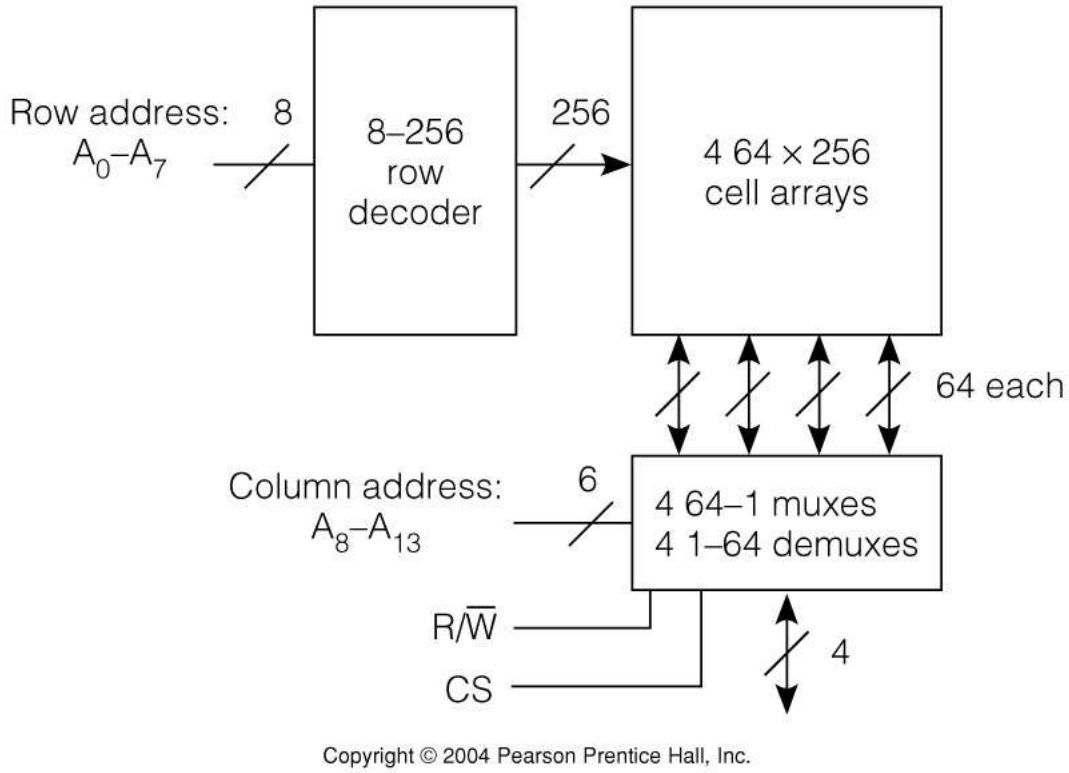


64 K x 1 Static RAM Chip



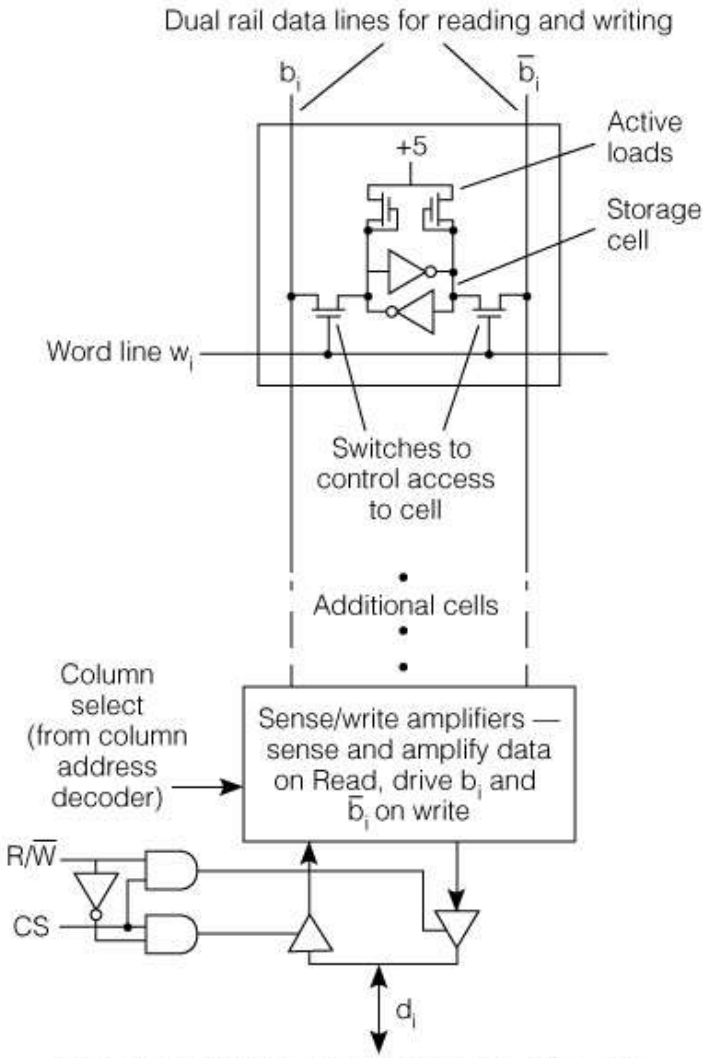
- Large address decoders are impractical because of large gate count and fan-in
  - Use row and column decoders
  - Design RAM to be 1-it wide to save pins
- Row decoder is select
- Column decoder is both a
  - Mux (read)
  - Demux (write)

2.5D Alternate Design



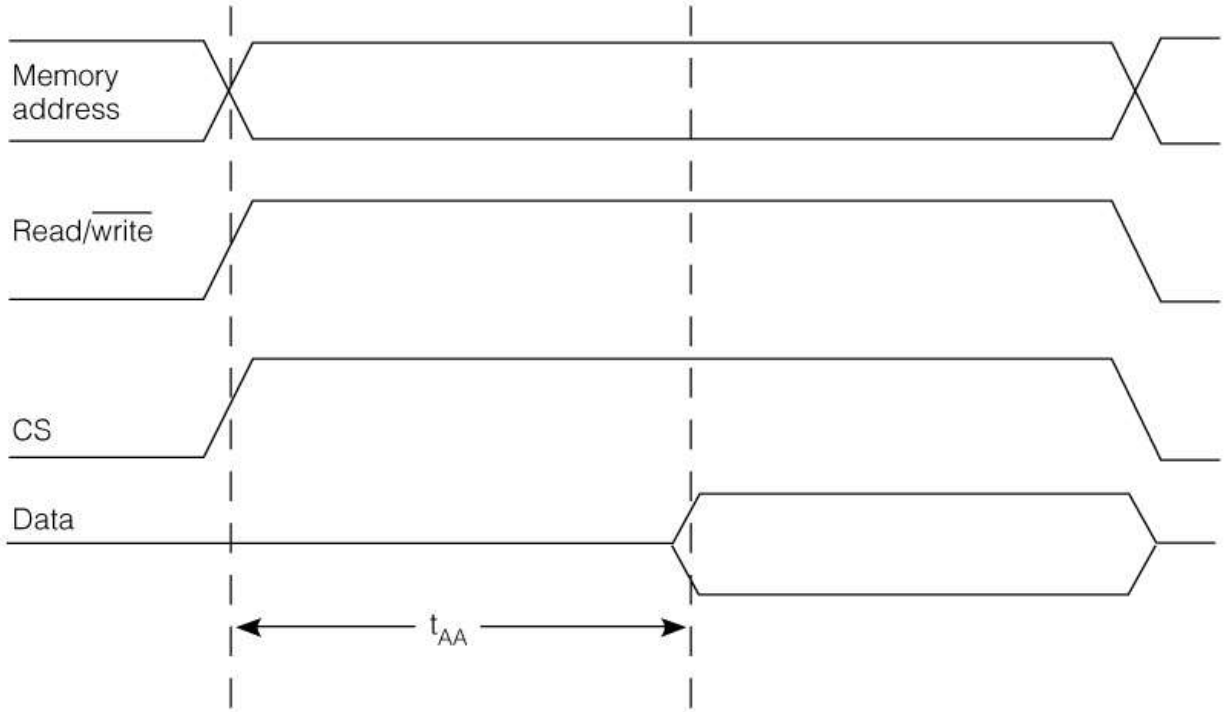
- Memory array does not have to be square
- Use multiple column selects
  - Return 4-bits rather than single bit
- Could be possible to work in 3D design
  - Need structures that support design
  - Divide address into 3 fields
    - Row
    - Column
    - Plane

Static RAM Cell Design



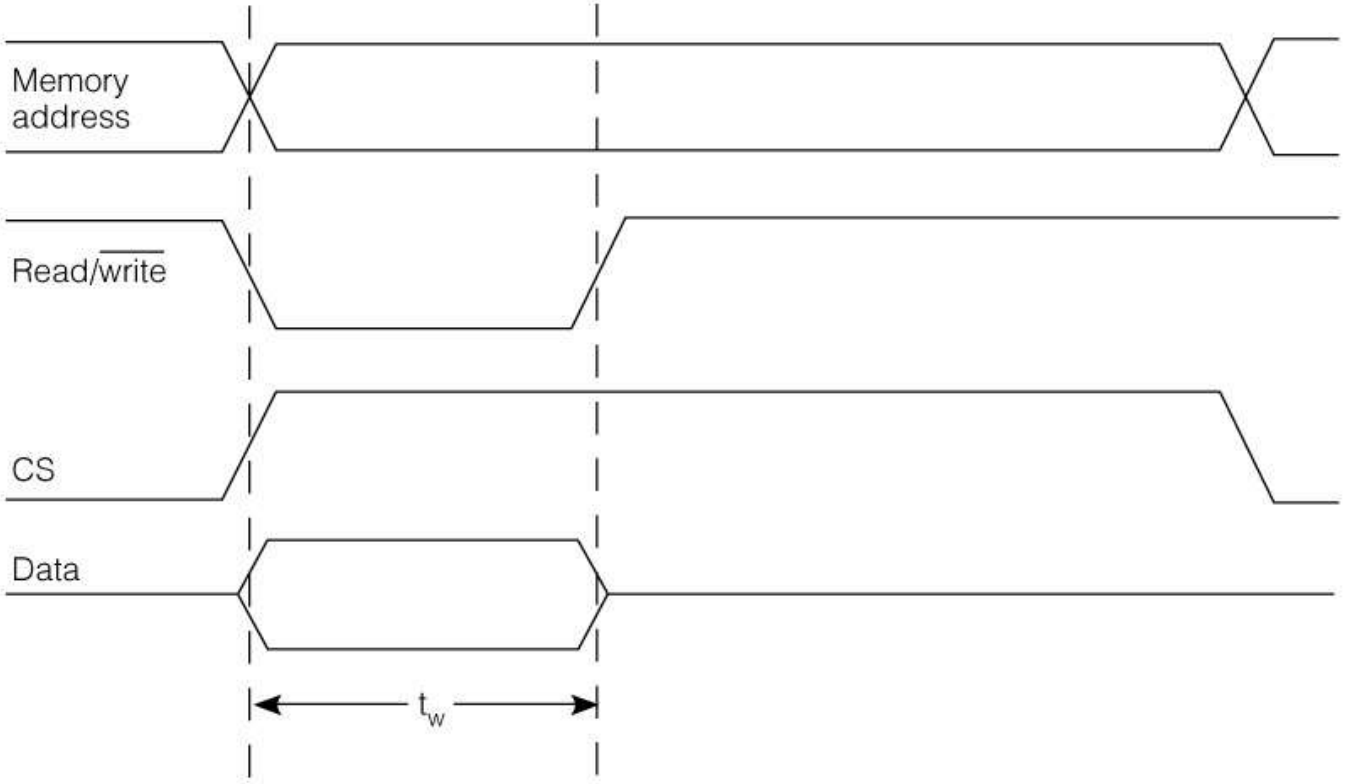
- 6-transistor cell
  - Use of cross-coupled inverters
  - 6-gate design more practical than previous 8-gate
    - Single transistor for a inverter
    - 2 transistors for control
    - 2 transistors for active loads
- Value in read by precharging bit lines to value halfway between 0 and 1 (2.5 V)
  - Assert word line
  - Bit lines driven to value stored in latch

Static RAM Read Timing



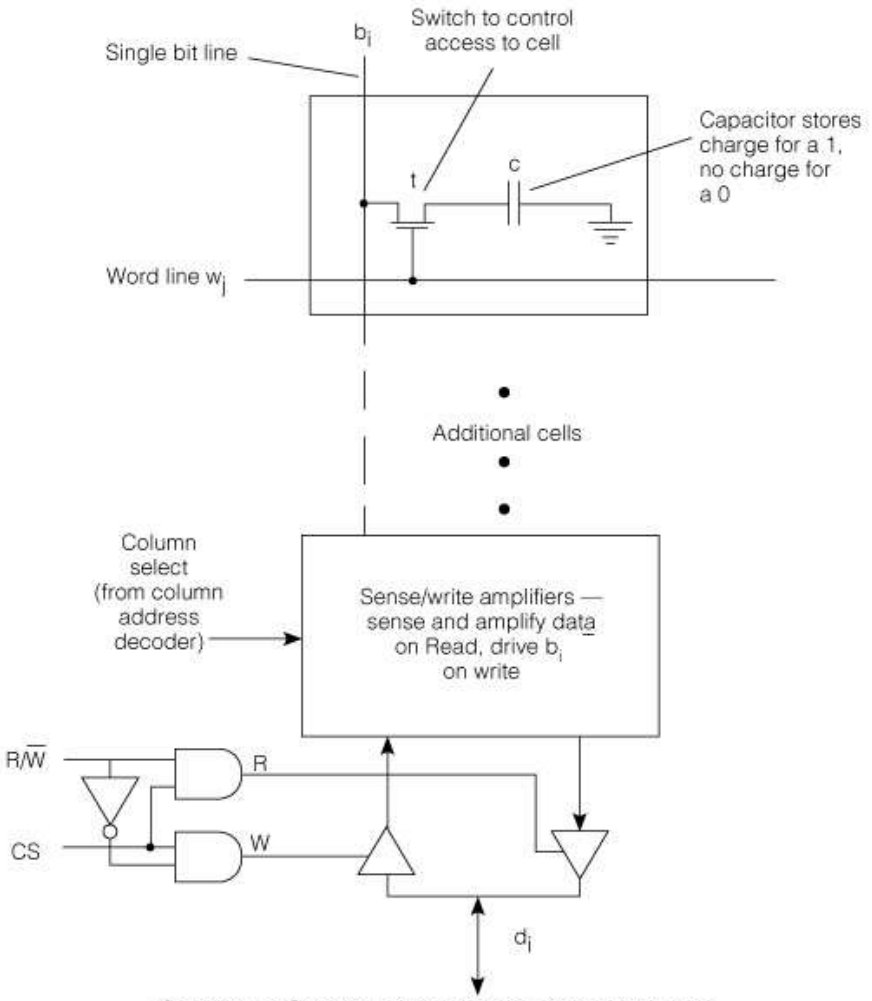
- t_{AA}- access time from address
  - Time required for RAM array to decode the address and provide value to data bus

Static RAM Write Timing



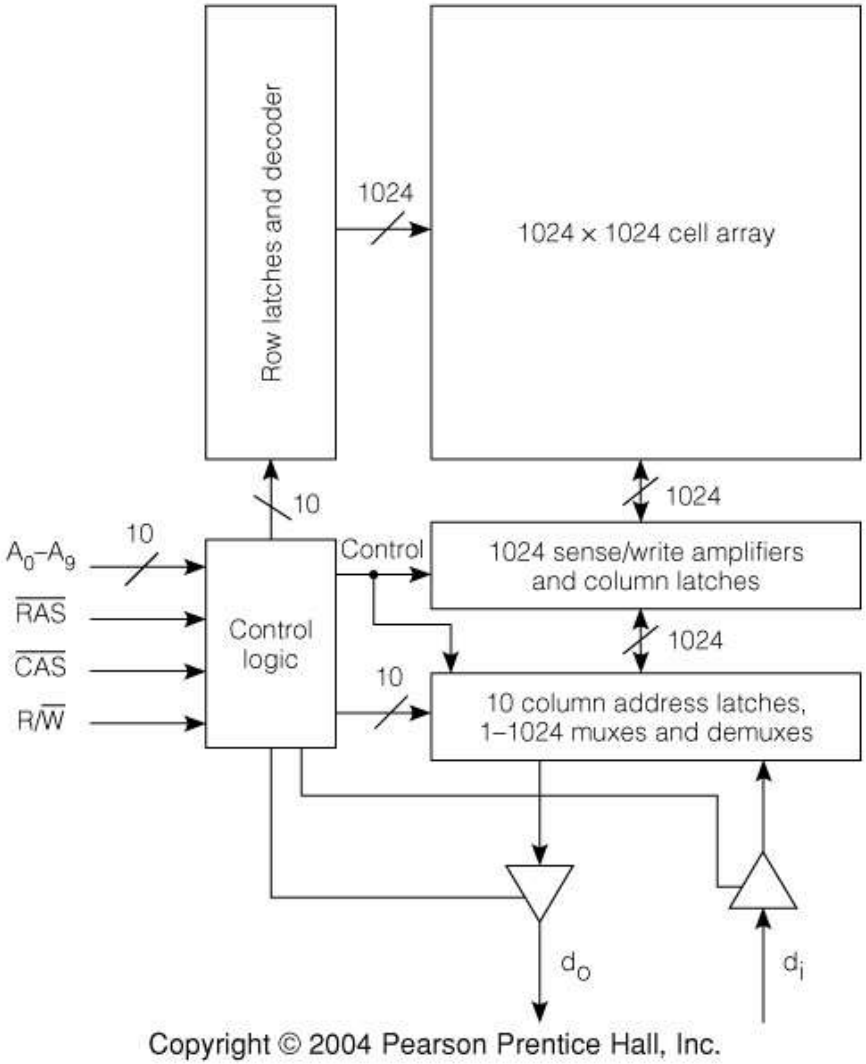
- t_W- write time
  - Time data must be held valid in order to decode address and store value in memory cells

Dynamic RAM (DRAM) Cell



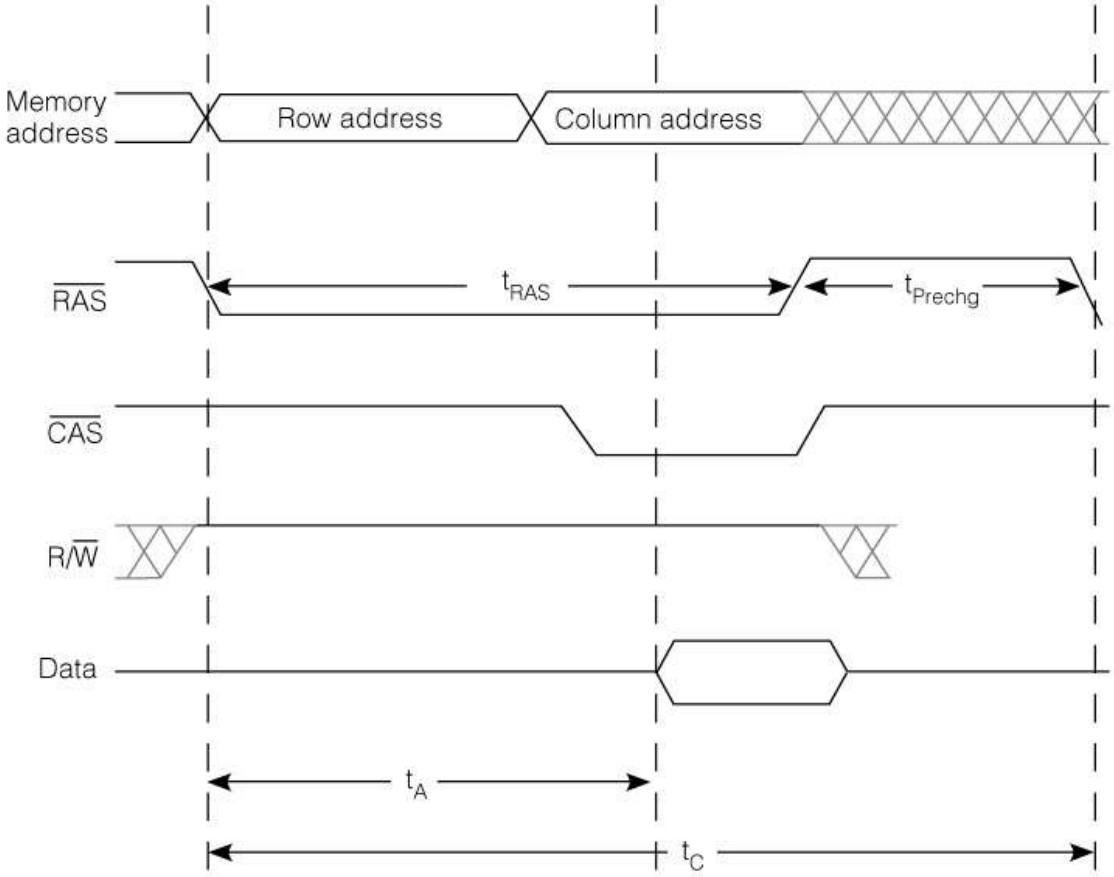
- State saved in single capacitor rather than cross-coupled gates
  - Significant savings in cell area
  - 1 transistor and capacitor
  - Single data bit line
- Capacitor discharges (4-15 msec range)
  - Refresh capacitor value by reading (sensing) bit line
    - Amplifies capacitor to restore value
- Write
  - Place value on bit line and assert word line
- Read
  - Precharge bit line
  - Sense value with sense/amplify circuitry

DRAM Chip Organization



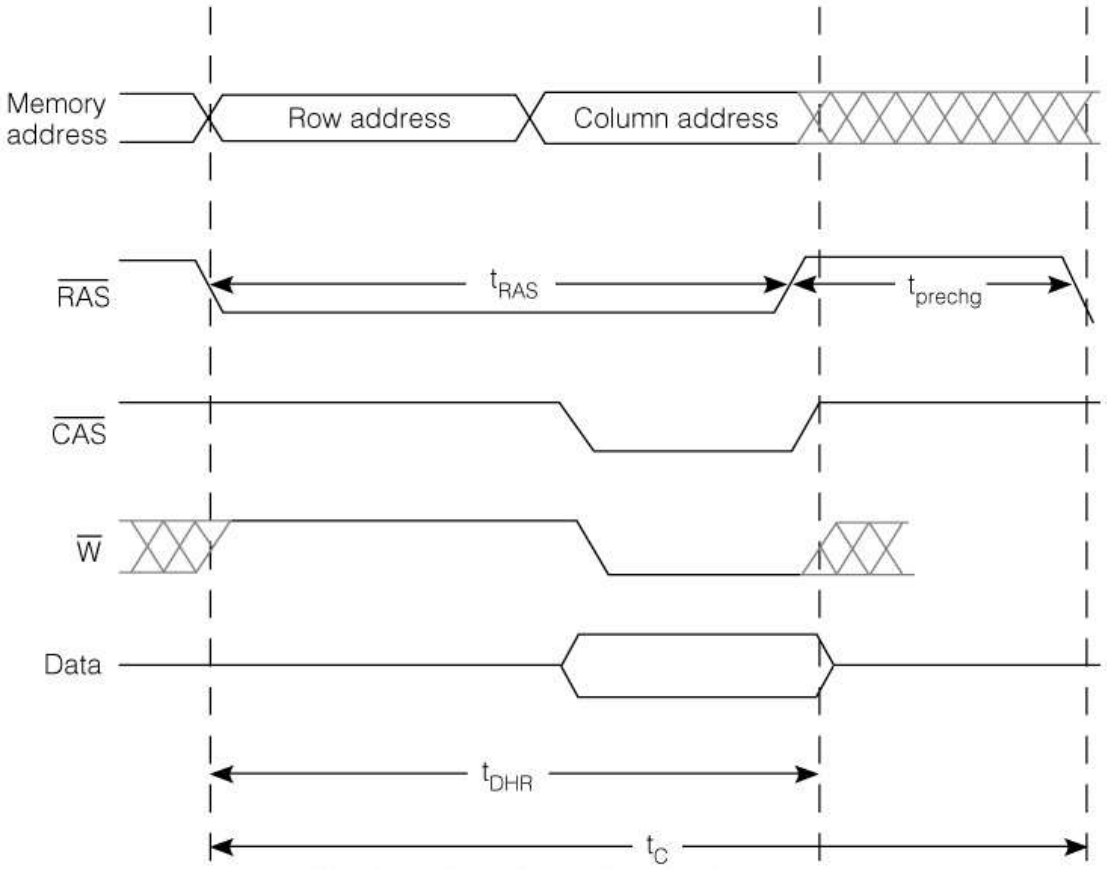
- Addresses are time-multiplexed onto bus
  - RAS – row address strobe
  - CAS – column address strobe
    - Used as CS function
- Design influenced mainly by number of pins and need to refresh cells
  - Time-multiplexing sends row and column address on same bus sequentially
    - 27 pins without time-multiplexing address (includes power and ground)
    - 17 pins with time-multiplexing

DRAM Read Timing



- t_A- access time
- Bit precharge operation causes difference between access time and cycle time

DRAM Read Timing

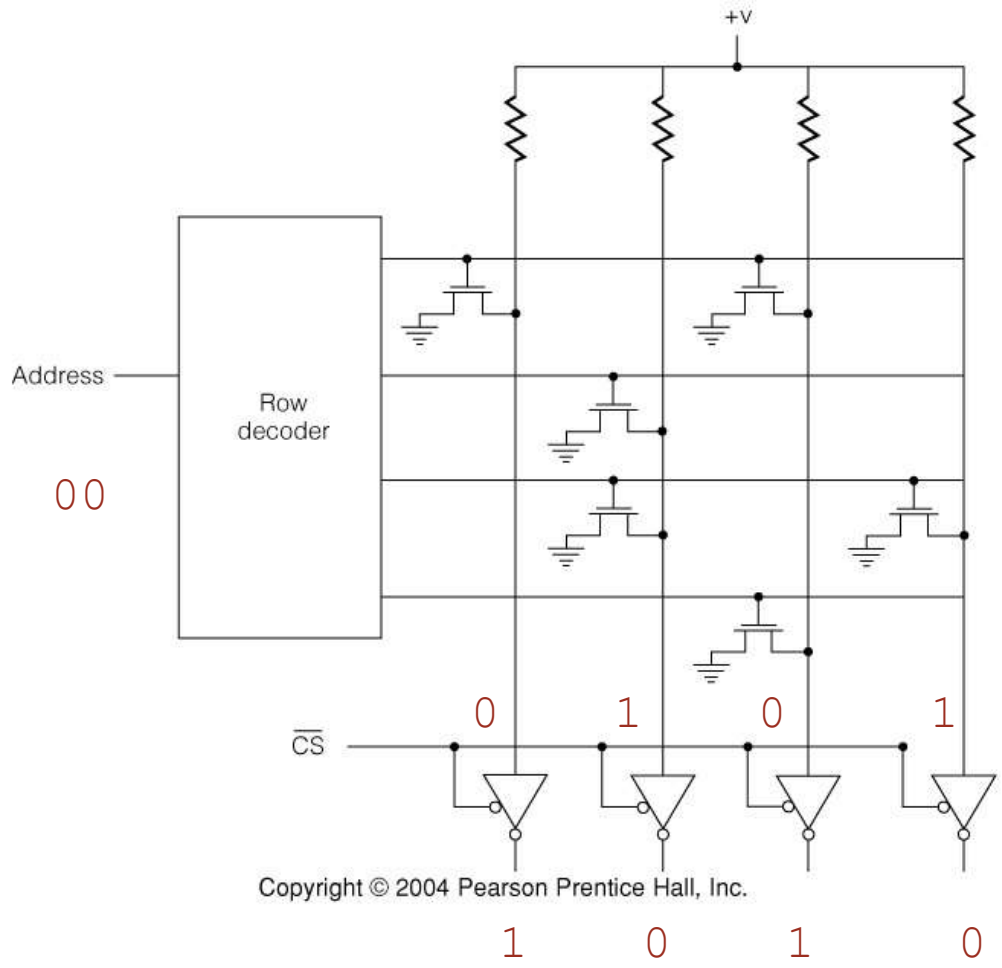


- t_{DHR} - data hold time from RAS

DRAM Refresh and Row Access

- Refresh is usually accomplished by a “RAS-only” cycle. T
  - Row address is placed on the address lines and RAS asserted to refresh entire row
  - Absence of a CAS phase signals the chip that a row refresh is requested, and thus no data is placed on the external data lines.
- Many chips use “CAS before RAS” to signal a refresh
  - Chip has an internal counter, and whenever CAS is asserted before RAS, it is a signal to refresh the row pointed to by the counter, and to increment the counter.
- Most DRAM vendors also supply one-chip DRAM controllers that encapsulate the refresh and other functions.
- Page mode, nibble mode, and static column mode allow rapid access to the entire row that has been read into the column latches.
- Video RAMS, VRAMS, clock an entire row into a shift register where it can be rapidly read out, bit by bit, for display.

CMOS ROM Chip



- Nonvolatile memory
  - Retains information when power is removed
- Necessary when machine code must be available at power-up (things like code)
  - Automobile engine control parameters
  - Video game cartridge
- Mask-programmed ROM
  - Inexpensive
  - Specify one-time the mask
    - Place transistor at every location where a one is stored

ROM Types

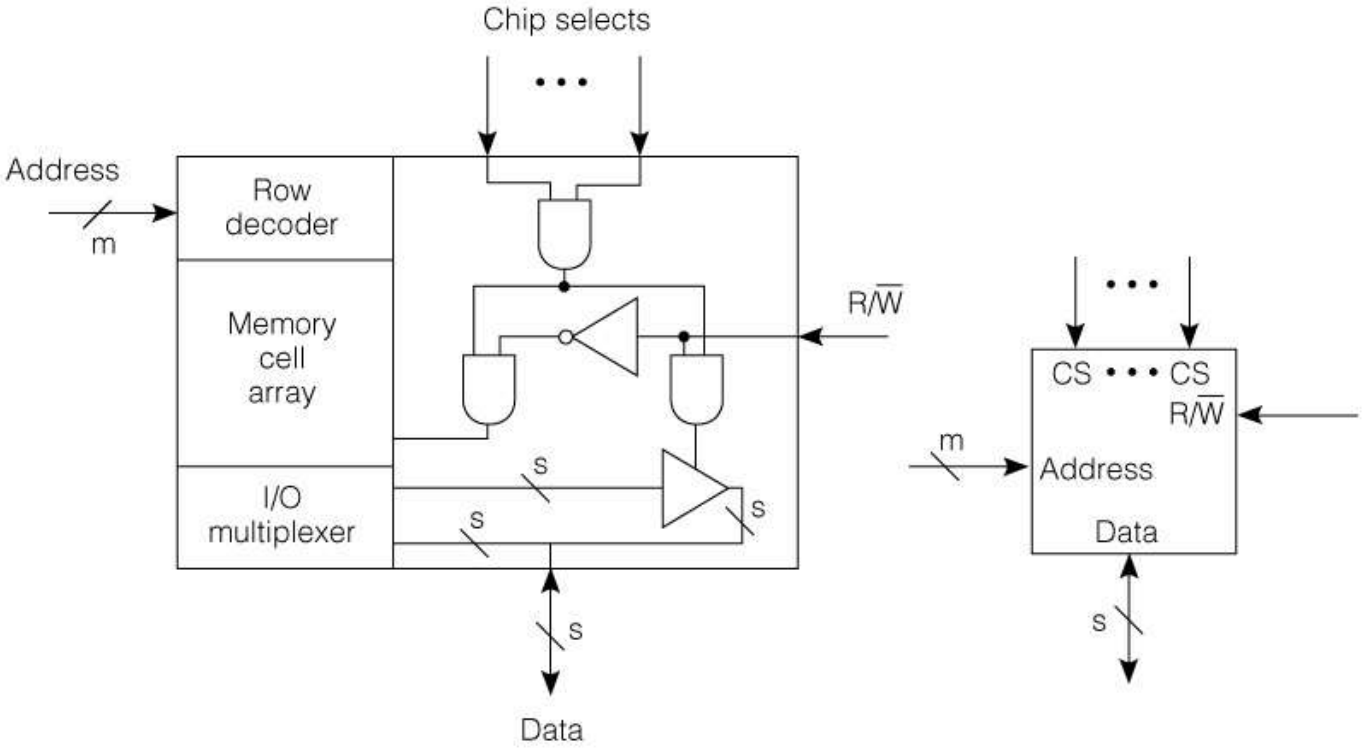
ROM Type	Cost	Programmability	Time to Program	Time to Erase
Mask-programmed ROM	Very inexpensive	At factory only	Weeks	N/A
PROM	Inexpensive	Once, by end user	Seconds	N/A
EPROM	Moderate	Many times	Seconds	20 minutes
Flash EPROM	Expensive	Many times	100 μsec	1 sec, large block
EEPROM	Very expensive	Many times	100 μsec	10 msec, byte

Memory Boards and Modules

- Need memories larger and wider than a single chip
- Chips organized into boards
  - Do not have to be physical boards
  - Consist of structured chip array present on mother board
- Collection of boards make up a memory module
  - Satisfy processor-main memory interface requirements
  - May have DRAM refresh capability
  - May expand the total main memory capacity
  - May be interleaved to provide faster access to blocks of words

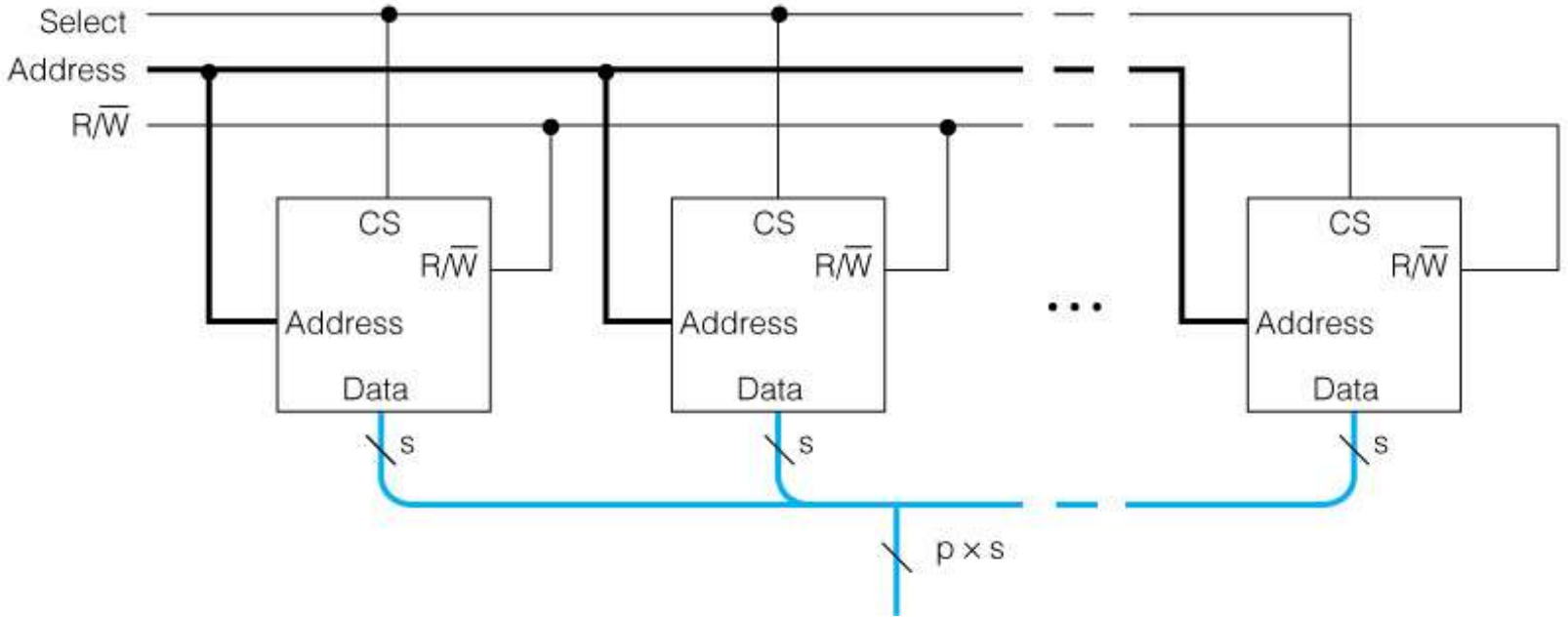
General Memory Chip Structure

- Slightly different view than previously
  - Multiple chip selects lines ease assembly of chips into chip arrays



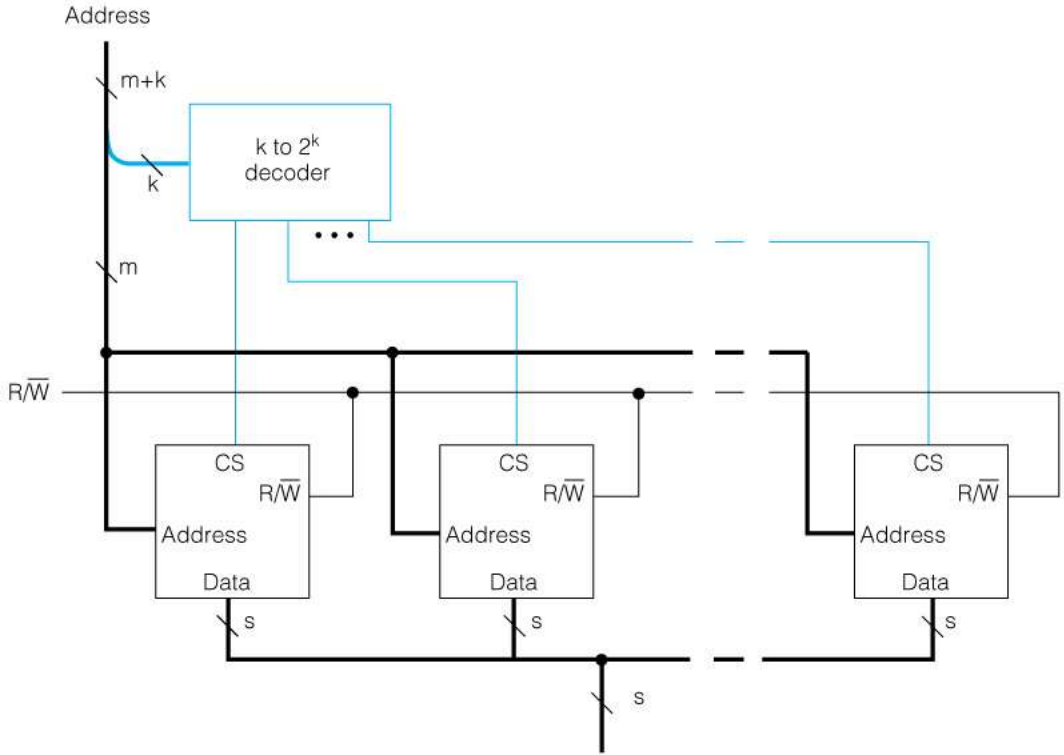
Expanding Memory Word Size

- Combine chips to have increased output size
  - Parallel connection of address, select, and chip selects
- P chips expand word size from s bits to $p \cdot s$ bits
  - Each chip provides a fixed location in word
- What are the memory capacity effects due to changing the address size?
  - RAM vs. DRAM

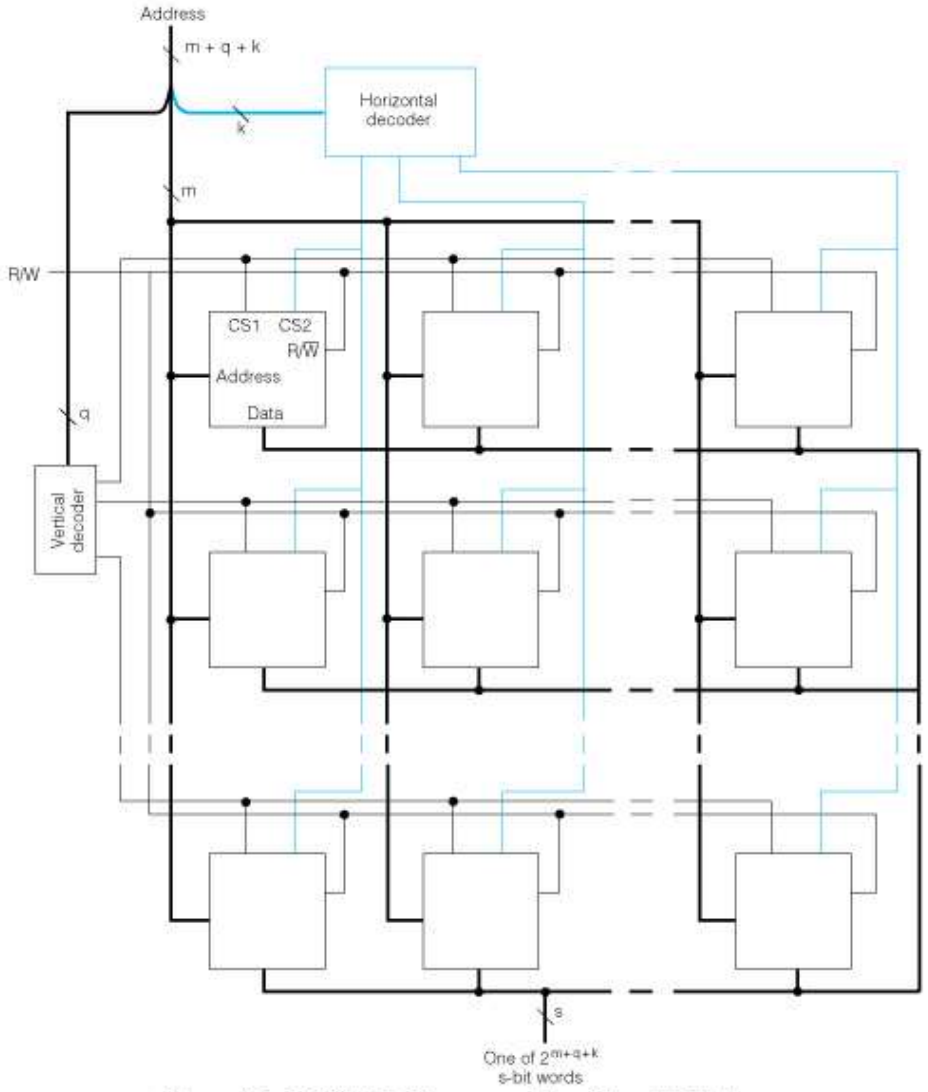


Increasing Number of Words

- Additional k address bits are used as a chip select signal
  - 2^k chips, each with 2^m words
    - What is memory size in number of words?
  - Word size remains at s bits



Chip Matrix Using Two Chip Selects

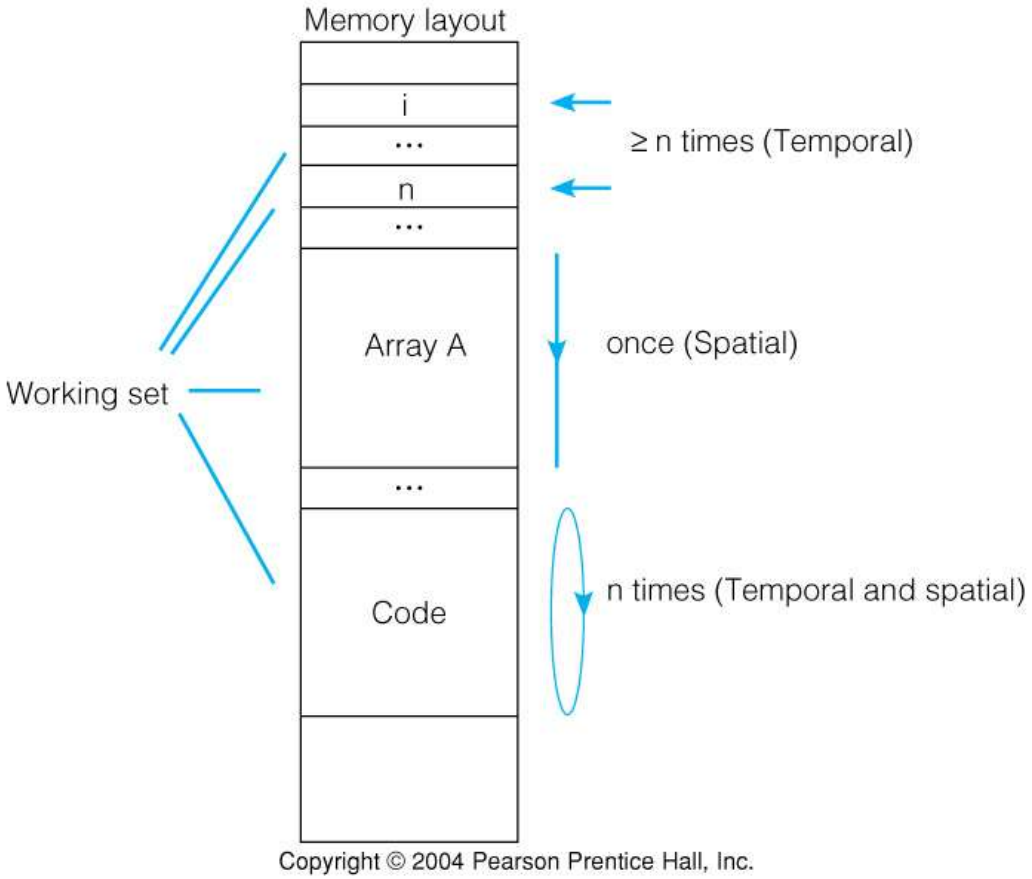


- Multiple chip select lines used to replace last level of gates in the matrix decoder
- Simplifies decoding using $(q + k)$-bit decoder
  - Single q-bit decoder
  - Single k-bit decoder

Memory Hierarchy

- Combine smaller, faster memory with lower, larger memory
  - Primary and secondary levels (e.g. cache and main memory)
- Move data efficiently from slow to fast memory using principle of locality
  - Programs tend to reference a confined area of memory repeatedly
  - Spatial locality – if a given memory location is referenced, addresses near it will likely be referenced soon
  - Temporal locality – if a given memory location is referenced, it is likely to be referenced again soon
  - Working set – set of memory locations referenced over a fixed time window

Temporal and Spatial Locality Example



- C `for` loop
- `for(int i=0; i<n; i+=1)`
  `A[i]=0;`

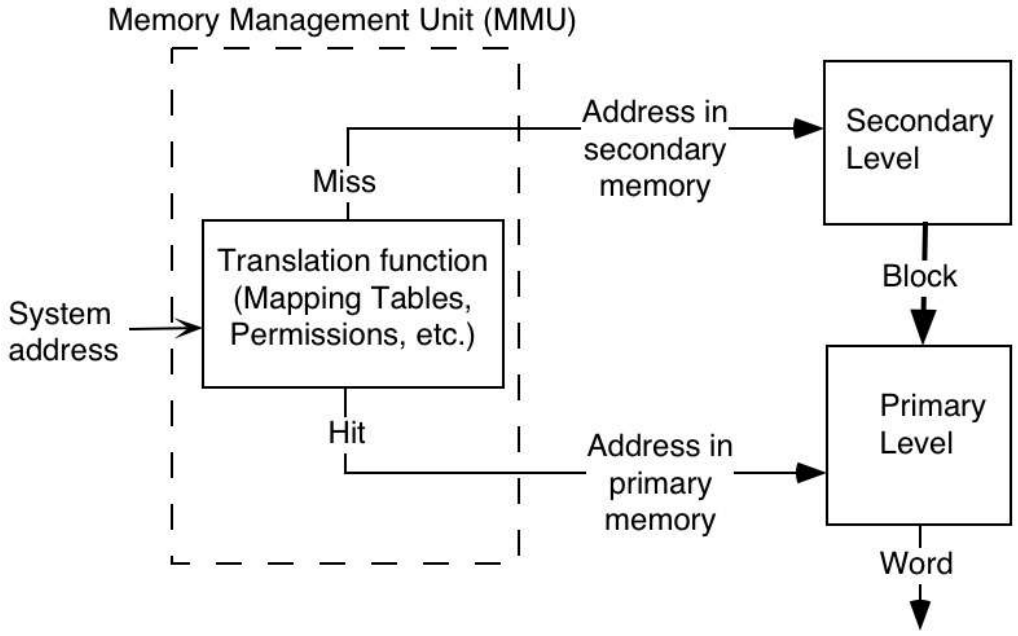
- Loop variables accessed many times
- Array sequentially accessed
- Loop code is sequentially accessed and repeated

Primary/Secondary Memory Levels

- Speed difference between levels
  - Latency – time to access first word
  - Bandwidth – number of words/sec transmitted
- Block – consecutive memory words
  - Transmitted between levels
  - Primary blocks are subset of secondary level information
- Blocks moved back/forth through hierarchy to satisfy memory access requests as working set changes
- Different addresses depending on the level
  - Primary address – address at primary level
  - Secondary address – address at secondary level

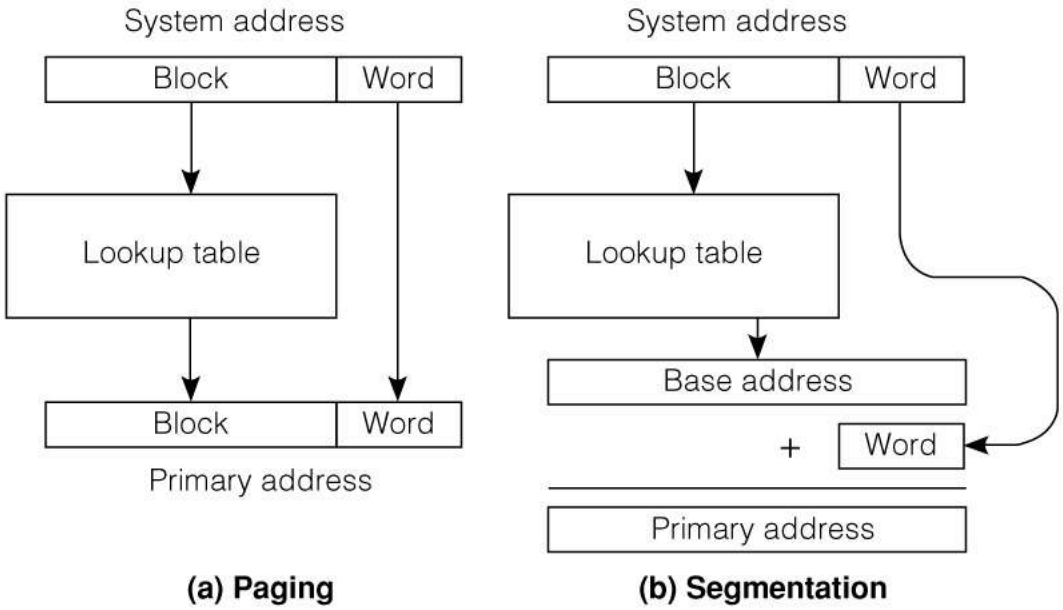
Addressing and Accessing a Two-Level Hierarchy

- Address translation (hardware or software) is needed to determine where to get value
  - Hit – primary address is found
    - Must be fast
  - Miss – must go to secondary level
    - Allowed to be slower, infrequent occurrence



Primary Address Formation

- Paging and Segmentation
  - Paging more common
  - Segmentation is limited to disk/tape where blocks are of variable length and random locations



2-Level Hierarchy Design Decisions

- Translation procedure to translate from system address to primary address.
- Block size – block transfer efficiency and miss ratio will be affected.
- Processor dispatch on miss – processor wait or processor multiprogrammed.
- Primary level placement – direct, associative, or a combination.
- Replacement policy – which block is to be replaced upon a miss.
- Direct access to secondary level – in the cache regime, can the processor directly access main memory upon a cache miss?
- Write through – can the processor write directly to main memory upon a cache miss?
- Read through – can the processor read directly from main memory upon a cache miss as the cache is being updated?
- Read or write bypass – can certain infrequent read or write misses be satisfied by a direct access of main memory without any block movement?