CPE300: Digital System Architecture and Design

Fall 2011
MW 17:30-18:45 CBC C316

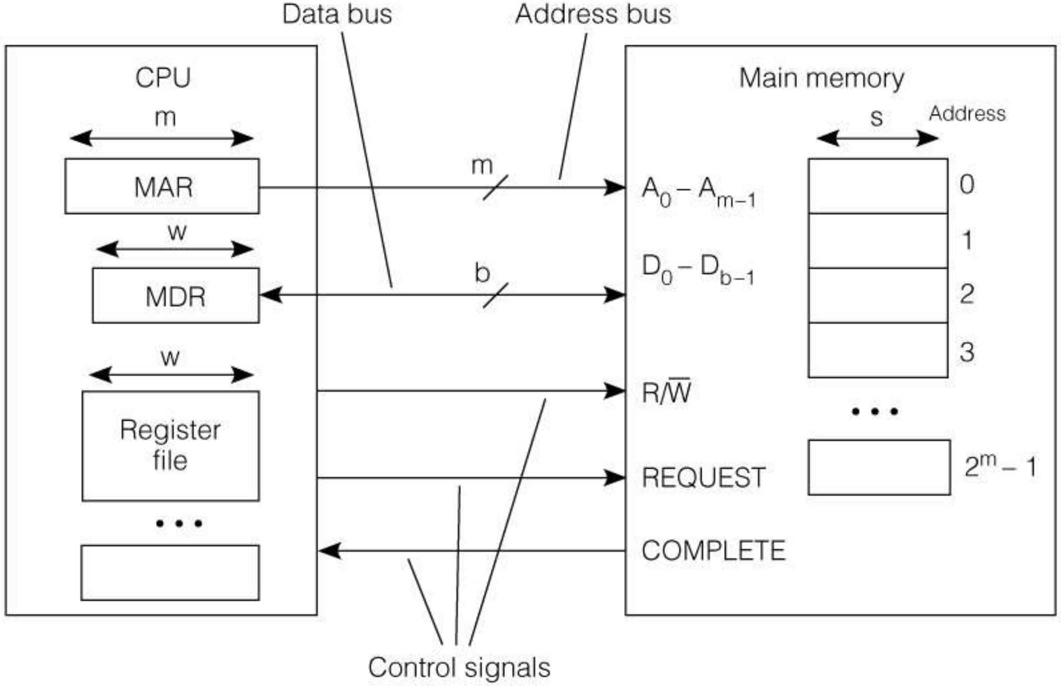
Cache
11232011

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

Outline

- Review Memory Components/Boards
- Two-Level Memory Hierarchy
- Cache
- Virtual Memory

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

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)

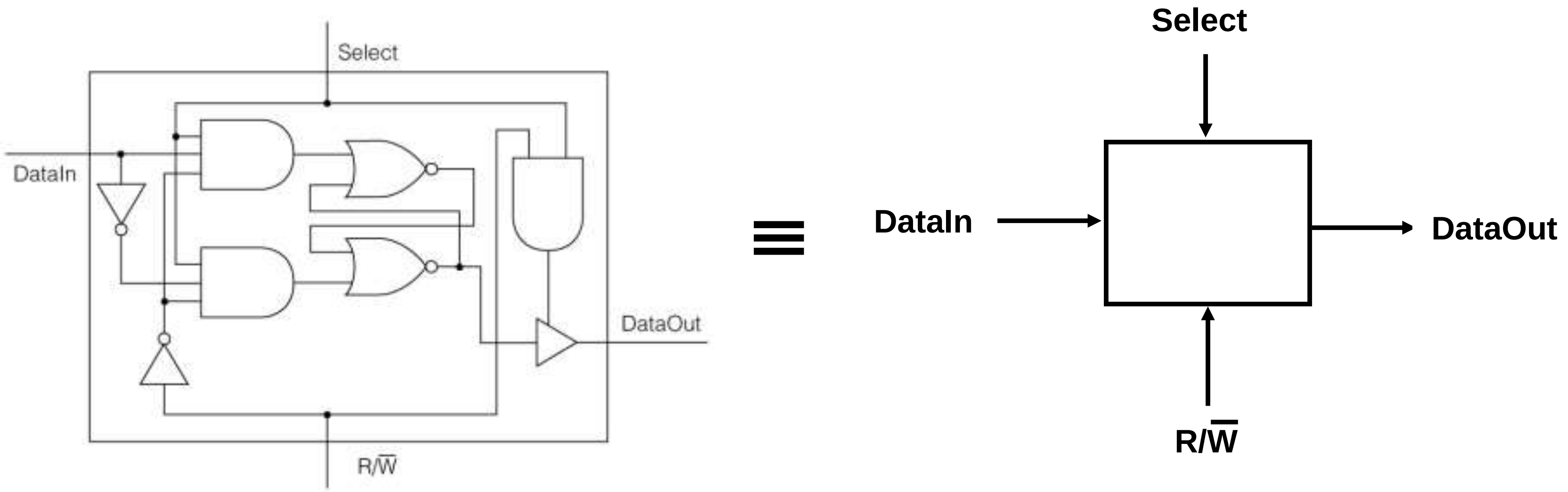
Memory Hierarchy, Cost, Performance

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

Component	CPU	1-3 Cache memories	Main memory	Disk memory	Tape memory
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

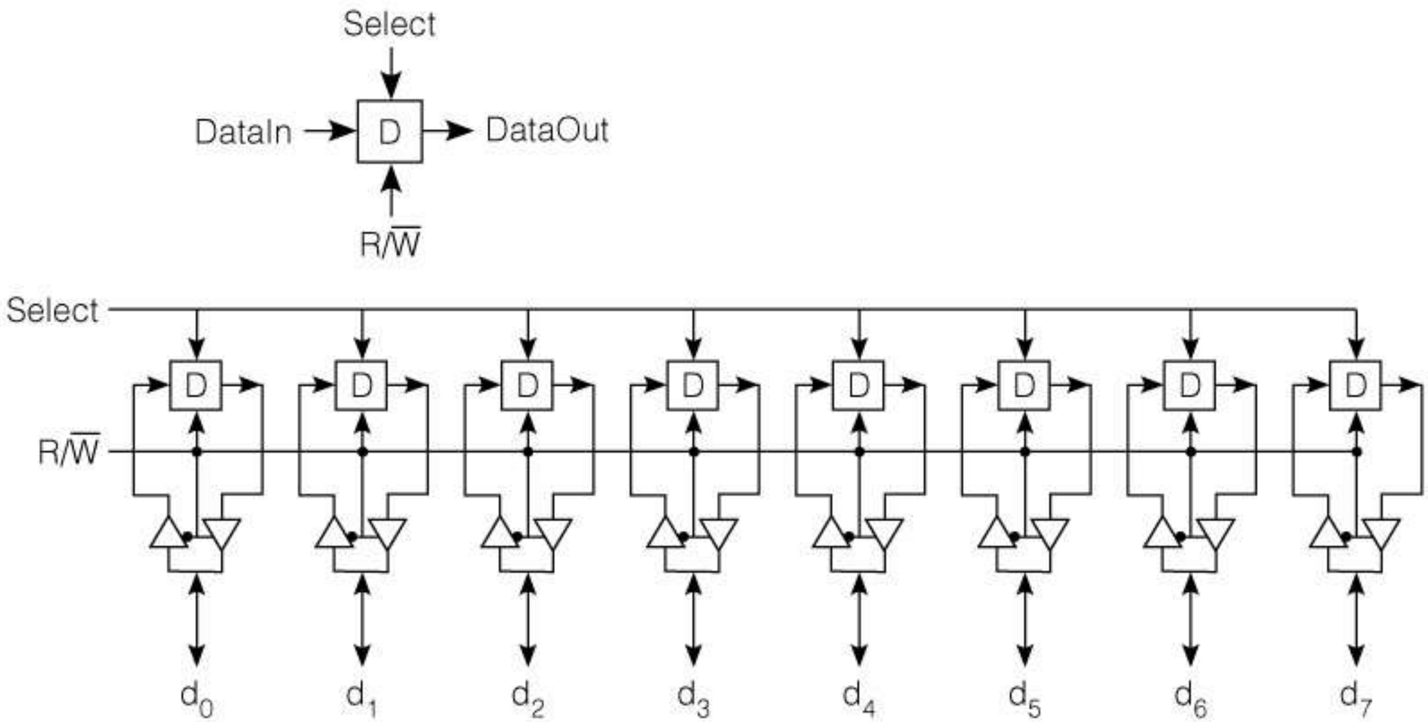
- RAM cell must provide four functions
  - Select, DataIn, DataOut, and R/W



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

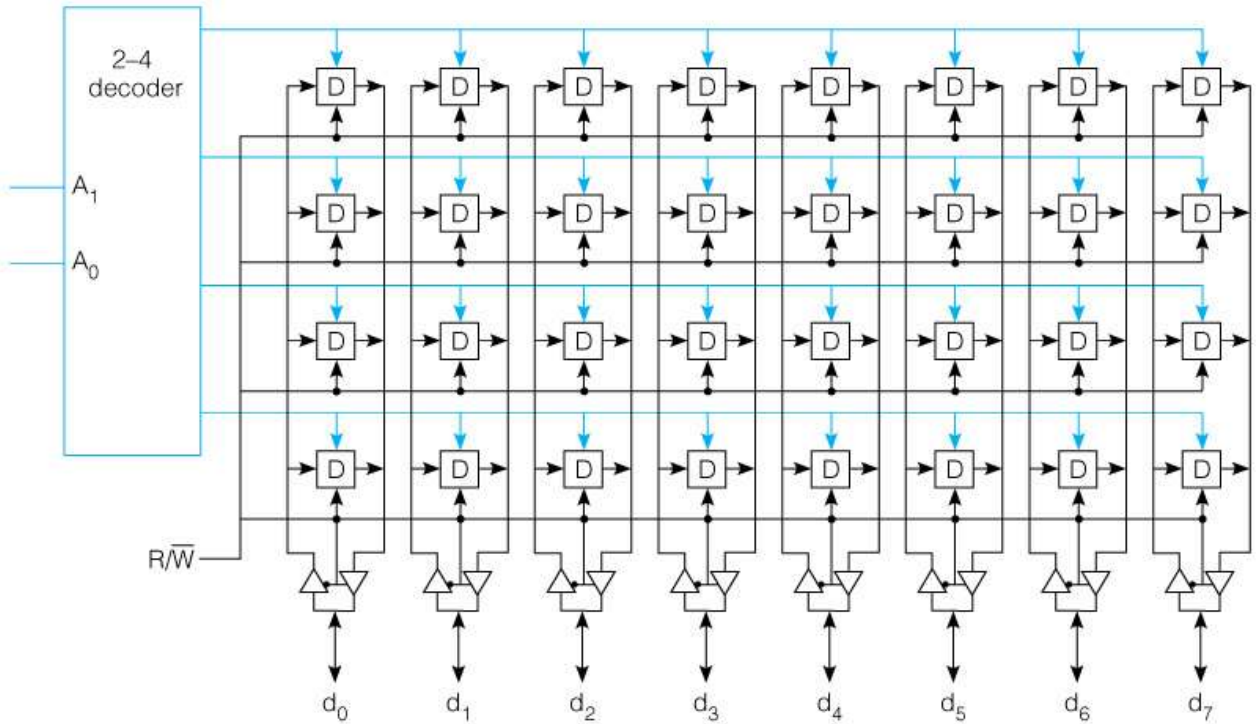
8-Bit Register as 1D RAM Array

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

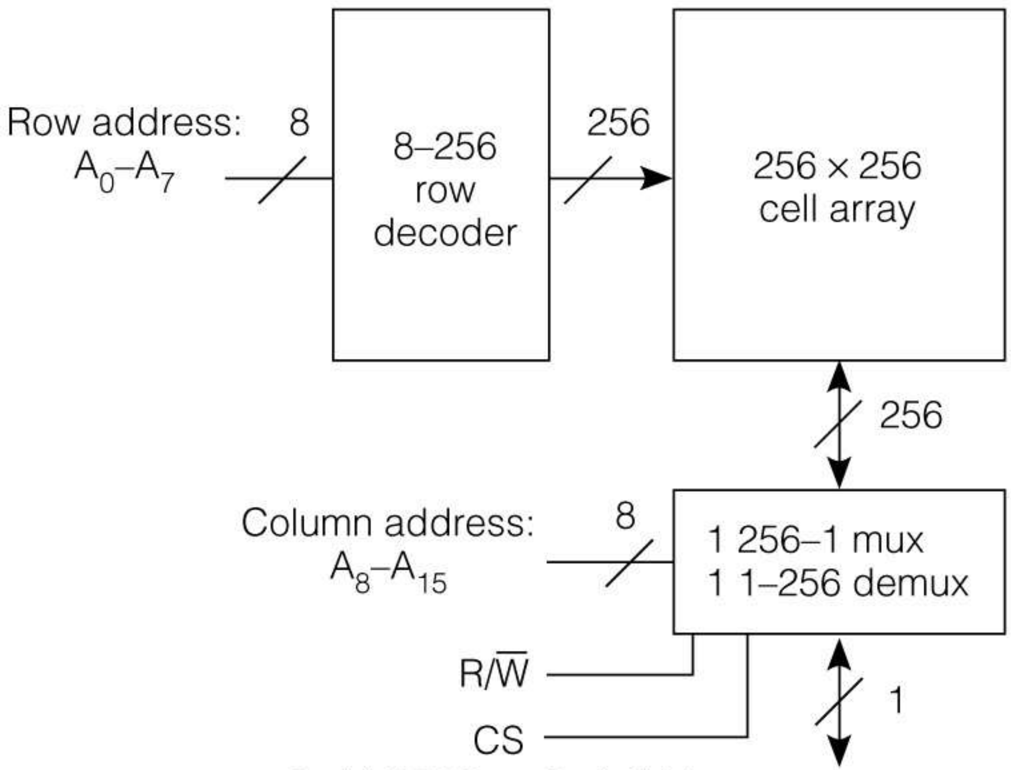


2D Memory Cell Array

- 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

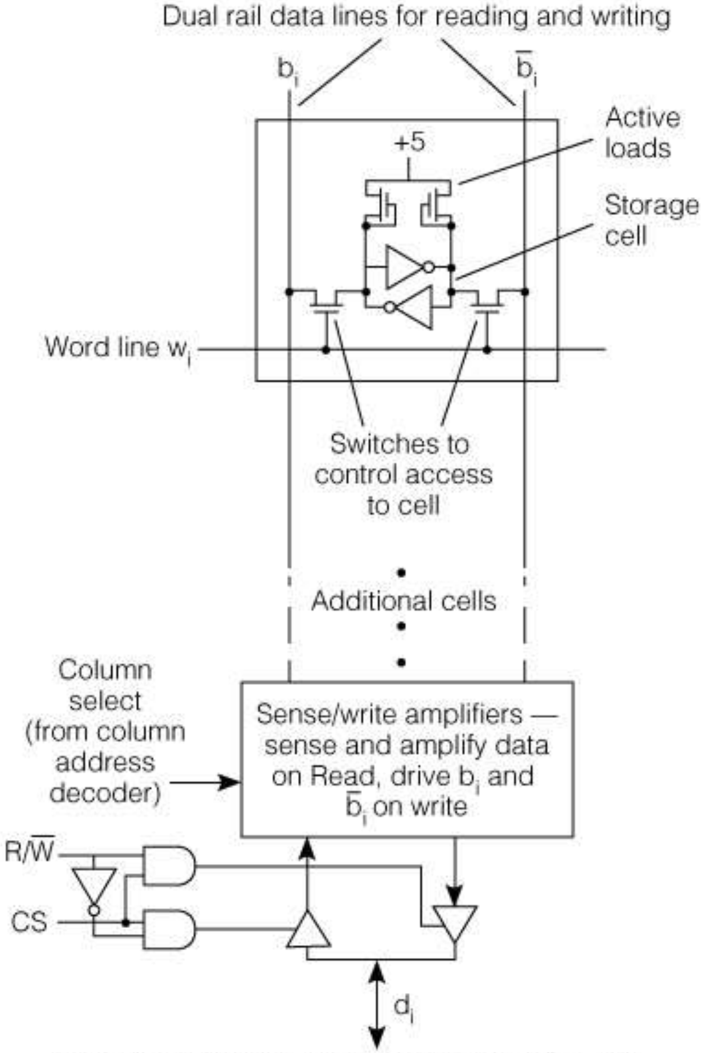


64 K x 1 Static RAM Chip



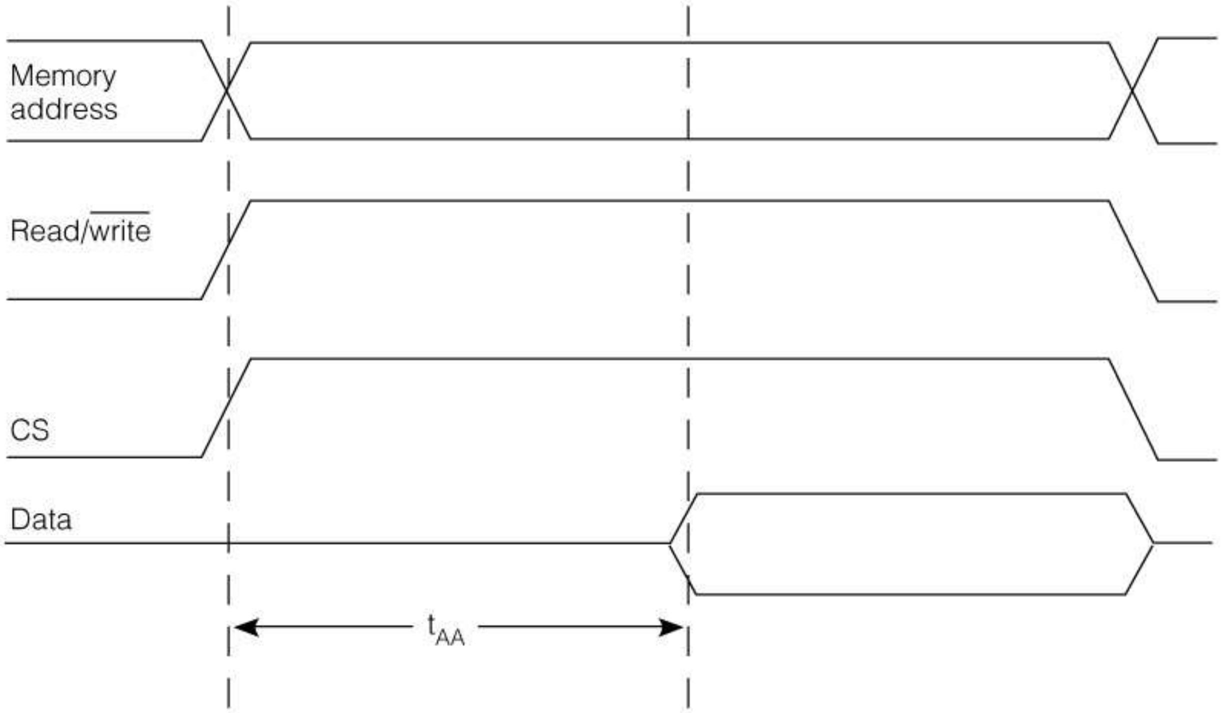
- Large address decoders are impractical because of large gate count and fan-in
  - Use row and column decoders
  - Design RAM to be 1-it wide to save pins
- Row decoder is select
- Column decoder is both a
  - Mux (read)
  - Demux (write)

Static RAM Cell Design



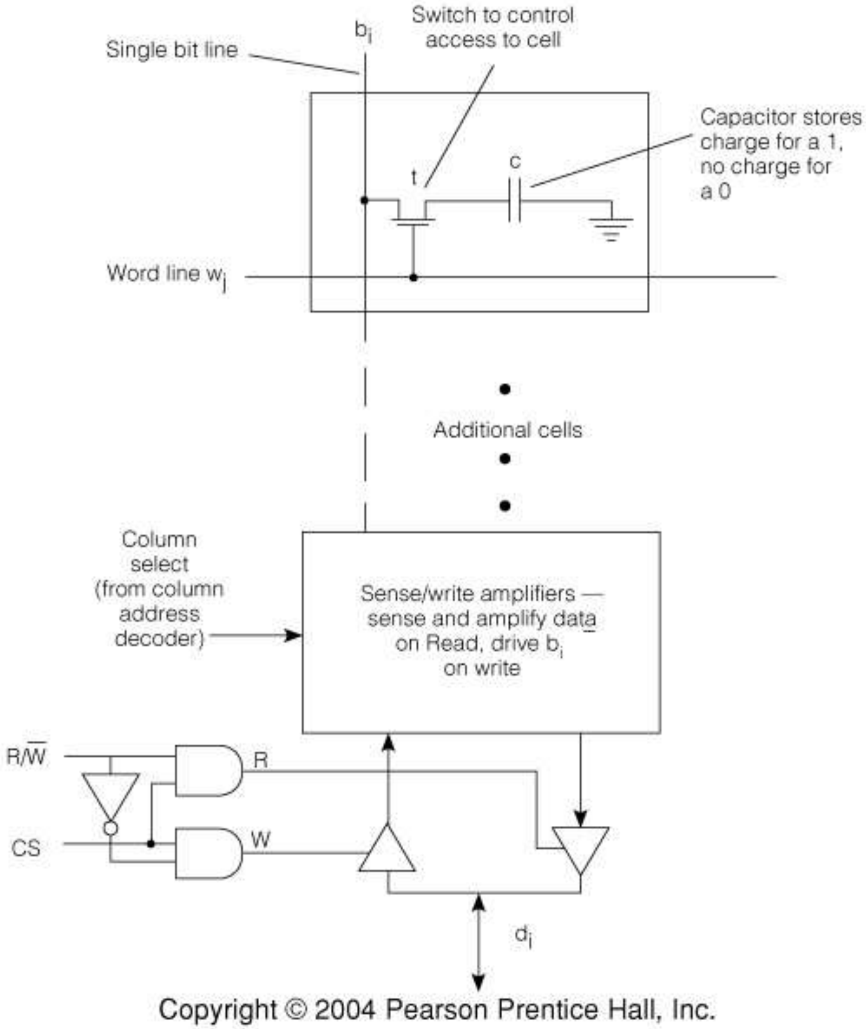
- 6-transistor cell
  - Use of cross-coupled inverters
  - 6-gate design more practical than previous 8-gate
    - Single transistor for a inverter
    - 2 transistors for control
    - 2 transistors for active loads
- Value in read by precharging bit lines to value halfway between 0 and 1 (2.5 V)
 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

Static RAM Read Timing



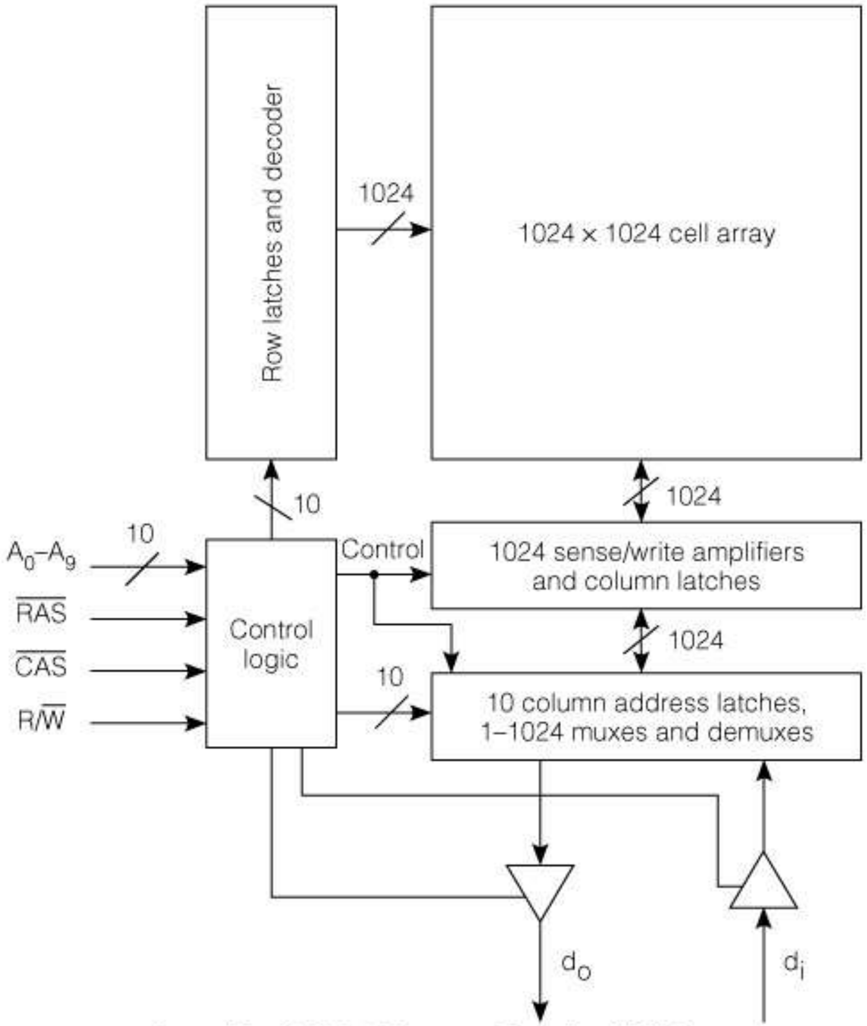
- t_{AA}- access time from address
  - Time required for RAM array to decode the address and provide value to data bus

Dynamic RAM (DRAM) Cell



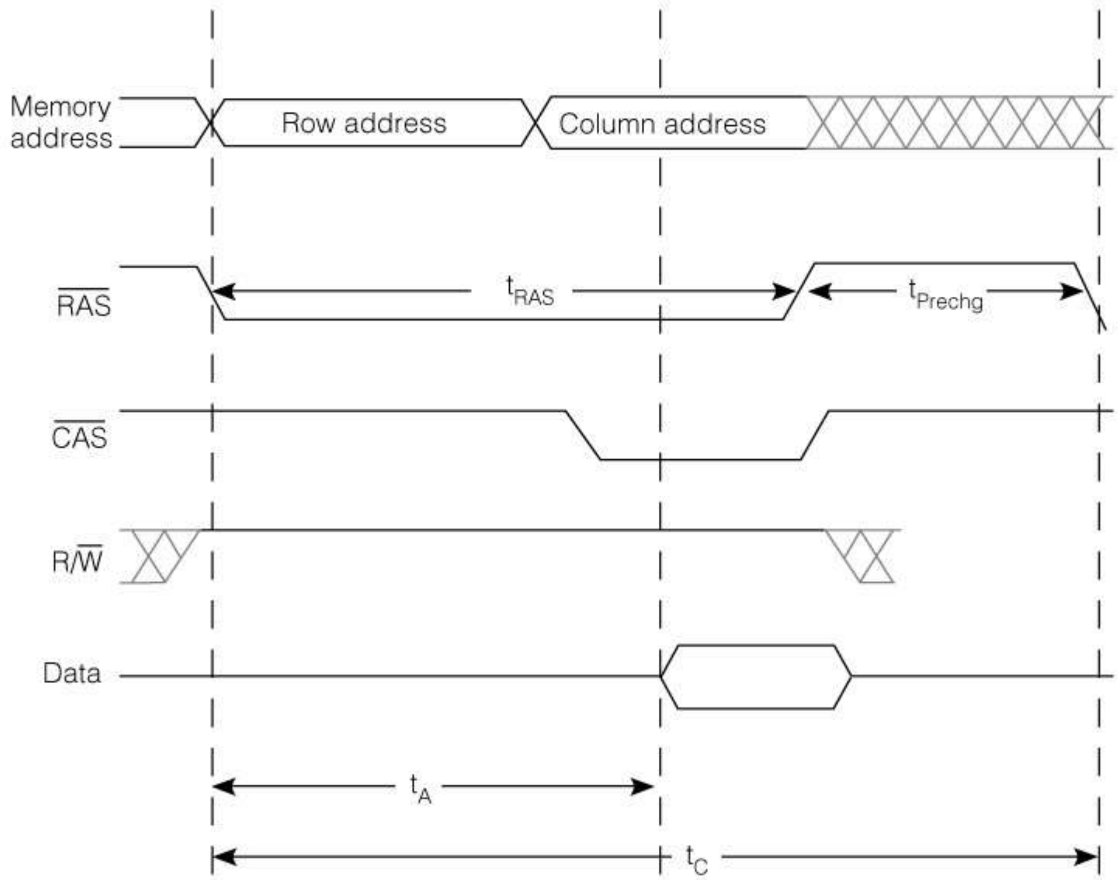
- State saved in single capacitor rather than cross-coupled gates
  - Significant savings in cell area
  - 1 transistor and capacitor
  - Single data bit line
- Capacitor discharges (4-15 msec range)
  - Refresh capacitor value by reading (sensing) bit line
    - Amplifies capacitor to restore value
- 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

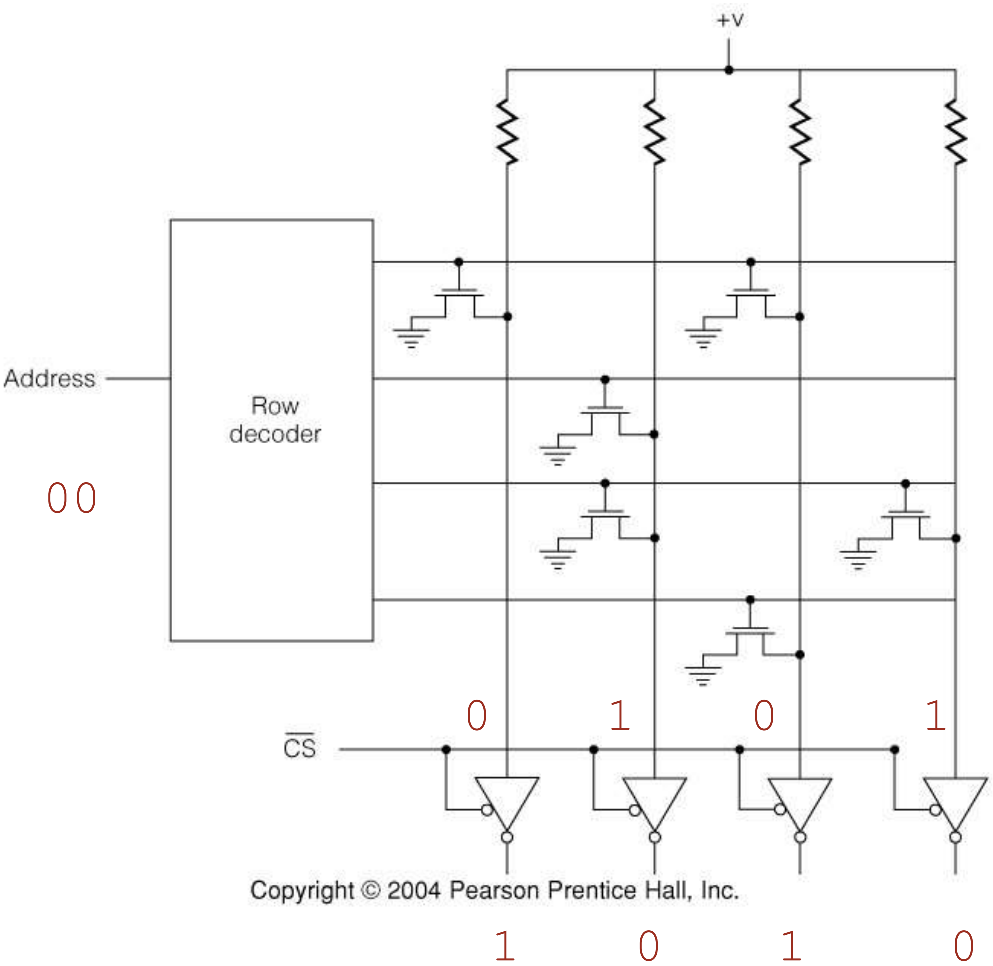


- t_A- access time
- Bit precharge operation causes difference between access time and cycle time

DRAM 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



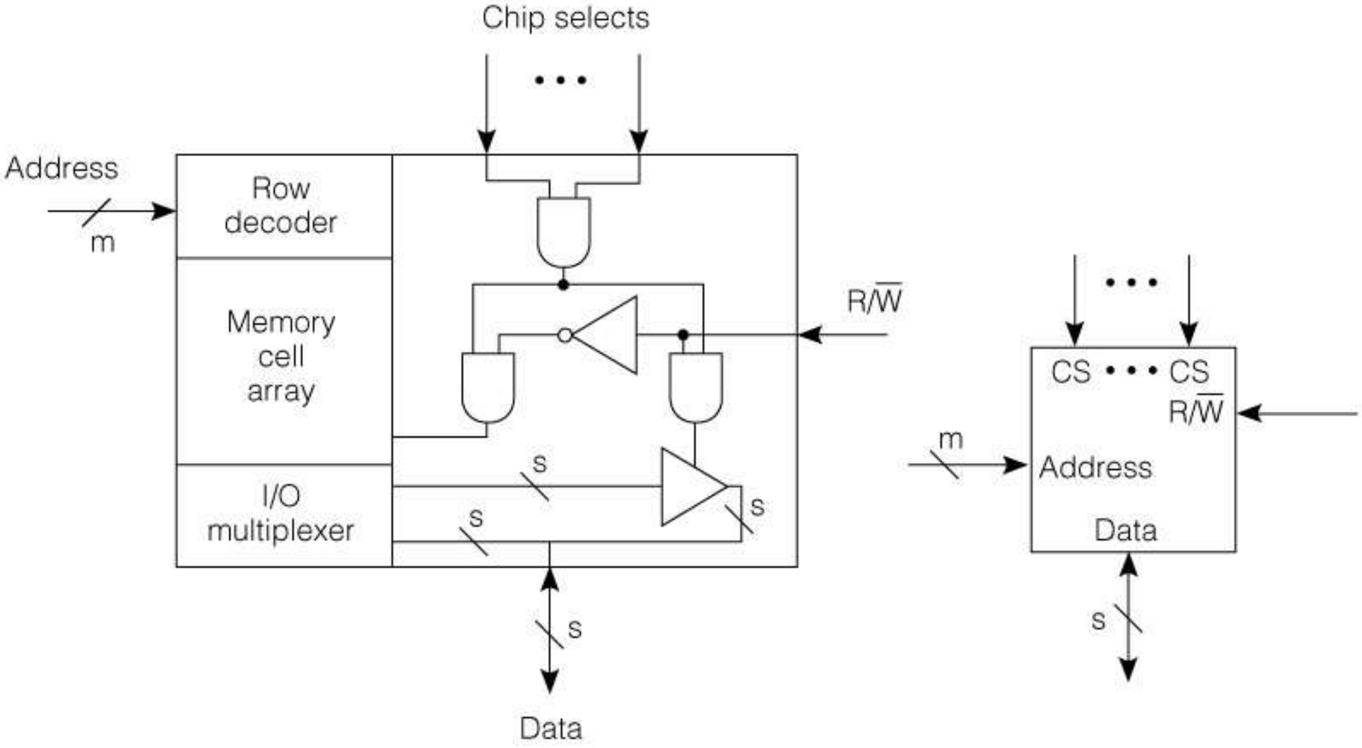
- Nonvolatile memory
  - Retains information when power is removed
- Necessary when machine code must be available at power-up (things like code)
  - Automobile engine control parameters
  - Video game cartridge
- Mask-programmed ROM
  - Inexpensive
  - Specify one-time the mask
    - Place transistor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

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

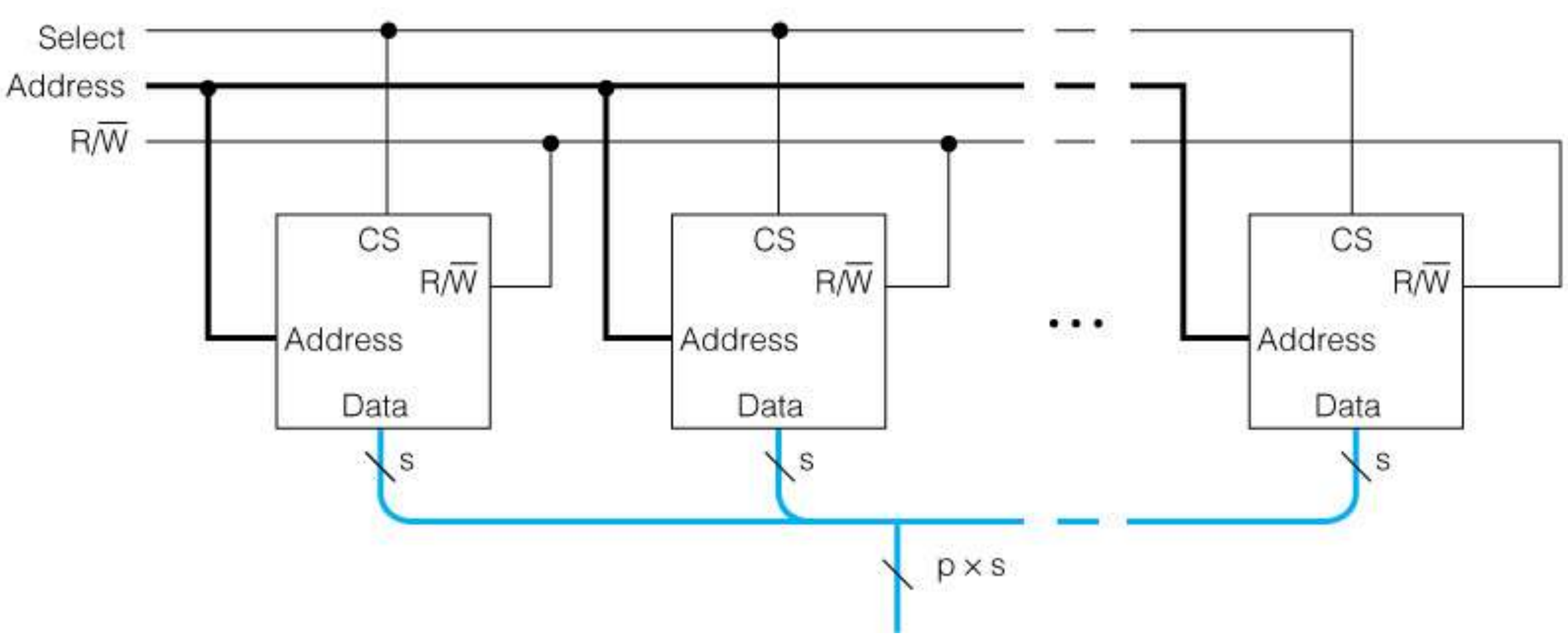
General Memory Chip Structure

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



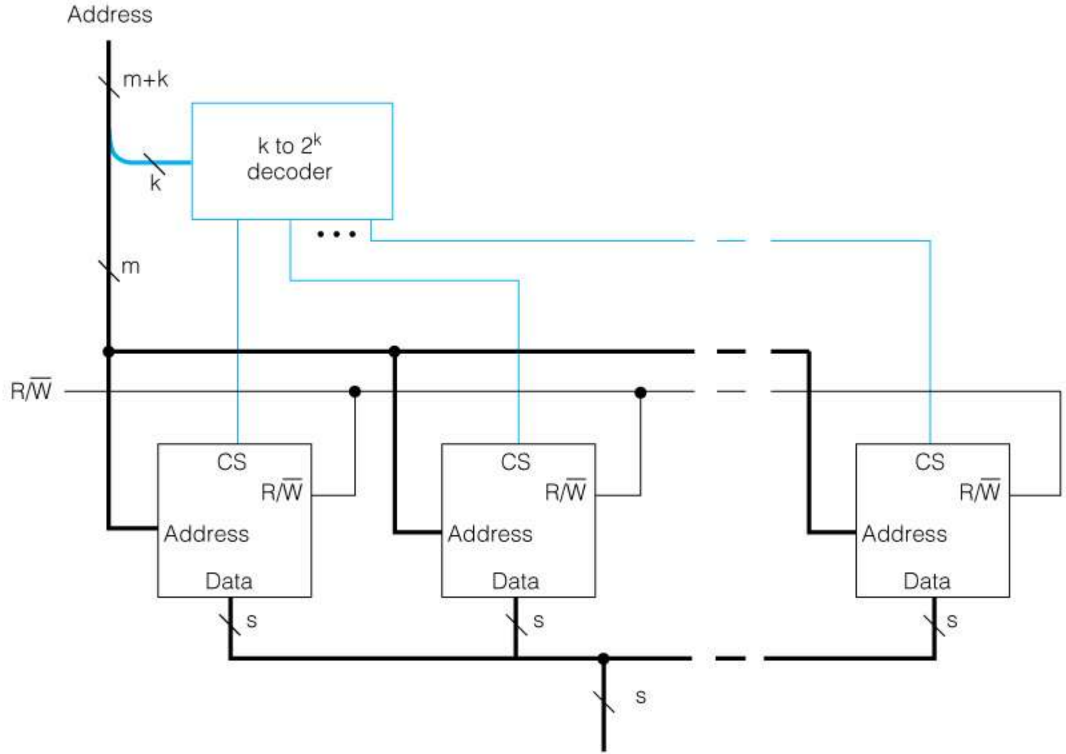
Expanding Memory Word Size

- 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

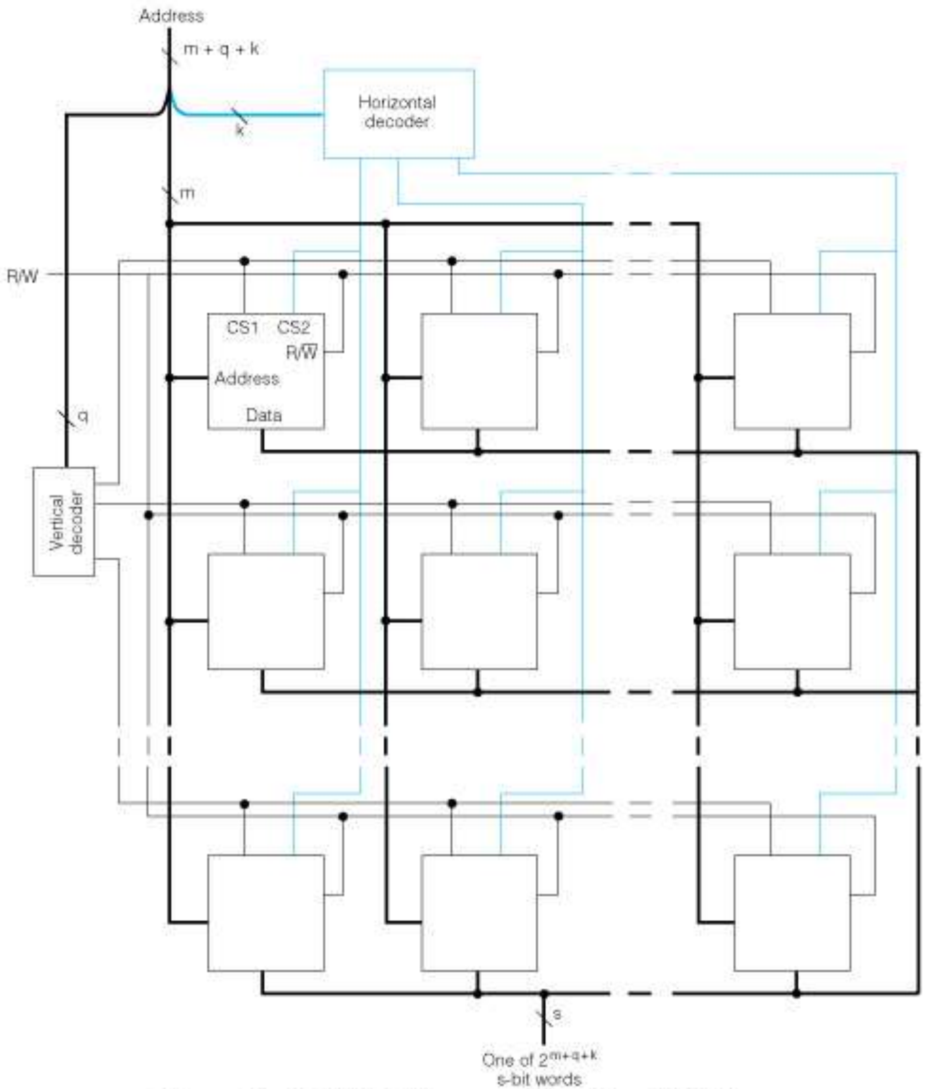


Increasing Number of Words

- 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

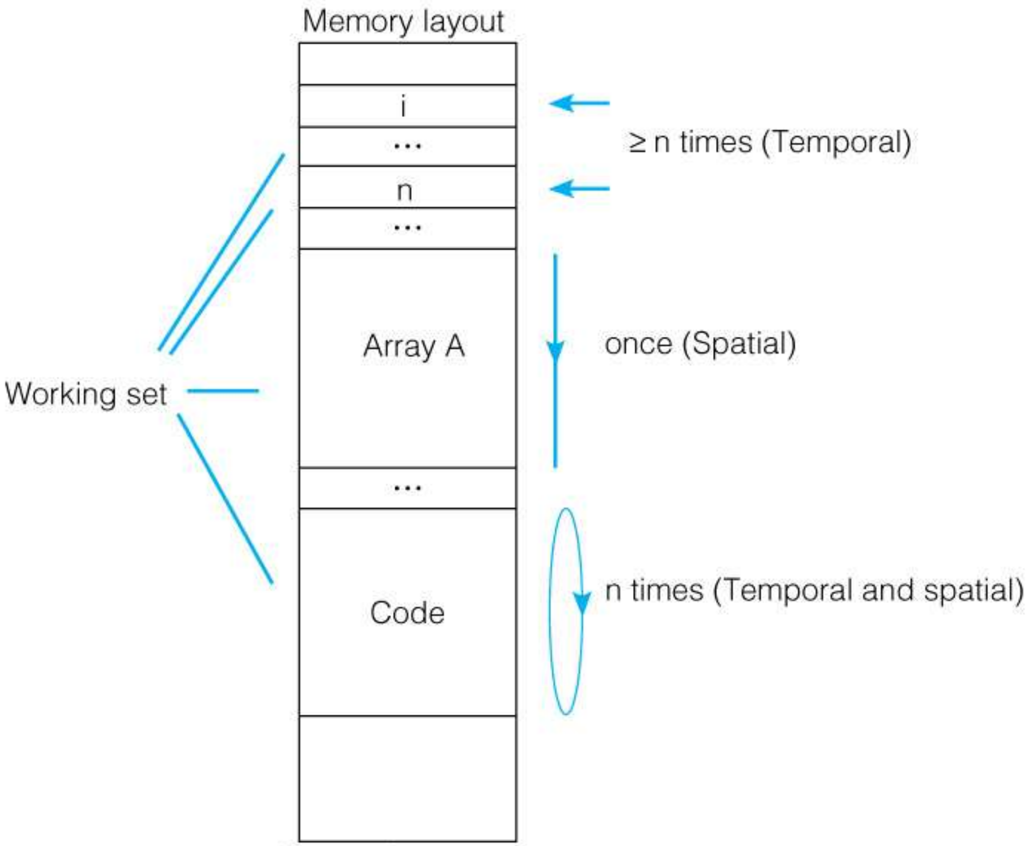


- Multiple chip select lines used to replace last level of gates in the matrix decoder
- Simplifies decoding using $(q + k)$-bit decoder
  - Single q-bit decoder
  - Single k-bit decoder

Memory Hierarchy

- Combine smaller, faster memory with slower, 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;`

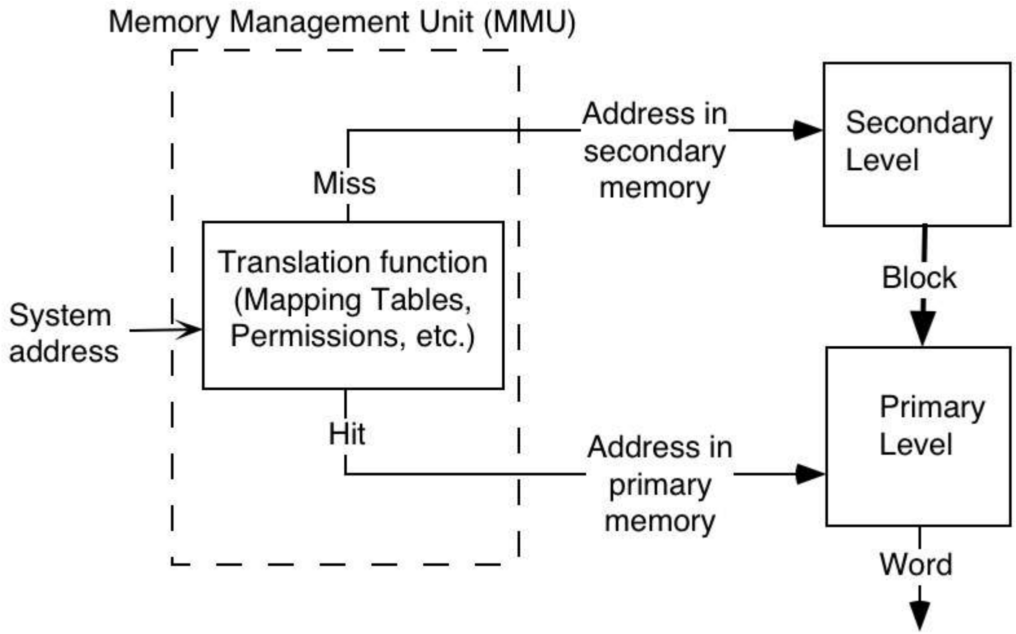
- Loop variables accessed many times
- Array sequentially accessed
- Loop code is sequentially accessed and repeated

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

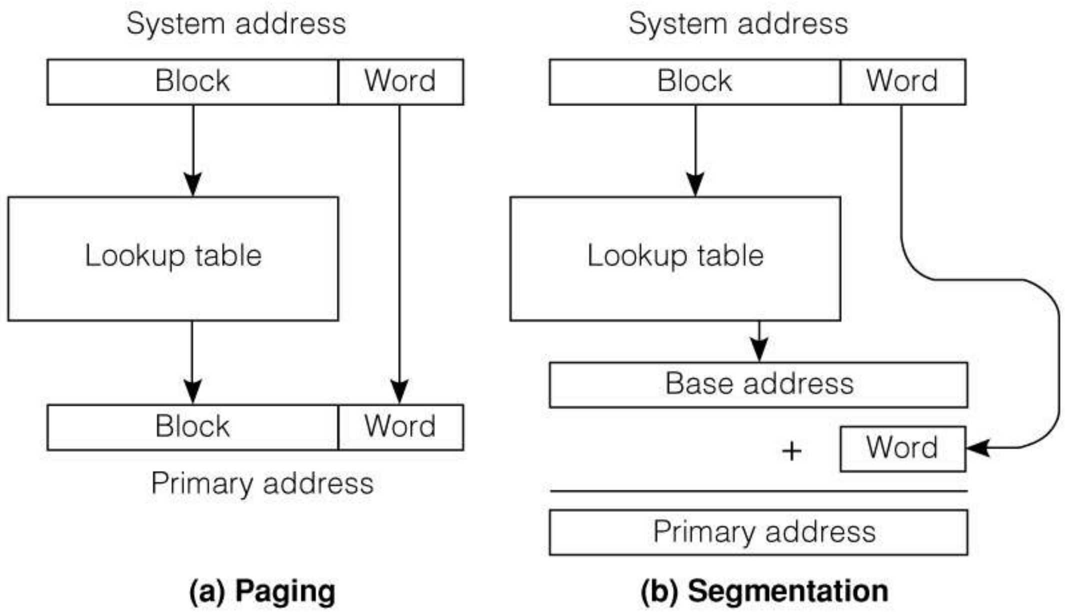
Addressing and Accessing a Two-Level Hierarchy

- 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



Hits and Misses

- Hit – word is found at level requested
  - Hit ratio (hit rate) - $h = \frac{\text{\# hits}}{\text{total \# references}}$
- Miss – word not found at level requested
  - Must request for containing block in the next higher level in memory hierarchy
  - Miss ratio = $1 - h$
- Access time
  - $t_a = ht_p + (1 - h)t_s$
  - t_p - primary memory access time
  - t_s - secondary memory access time

Virtual Memory

- Memory hierarchy usually of main memory and disk
- Enormous speed difference between main memory and disk
  - Order of 10^6 factor
  - Processor should not be kept waiting for transfer into memory upon miss
- Multiprogramming shares the processor among independent programs stored in memory
  - On miss switch to another program
- Miss response can be assisted by processor
  - I/O, placement/replacement decisions, computations of disk addresses

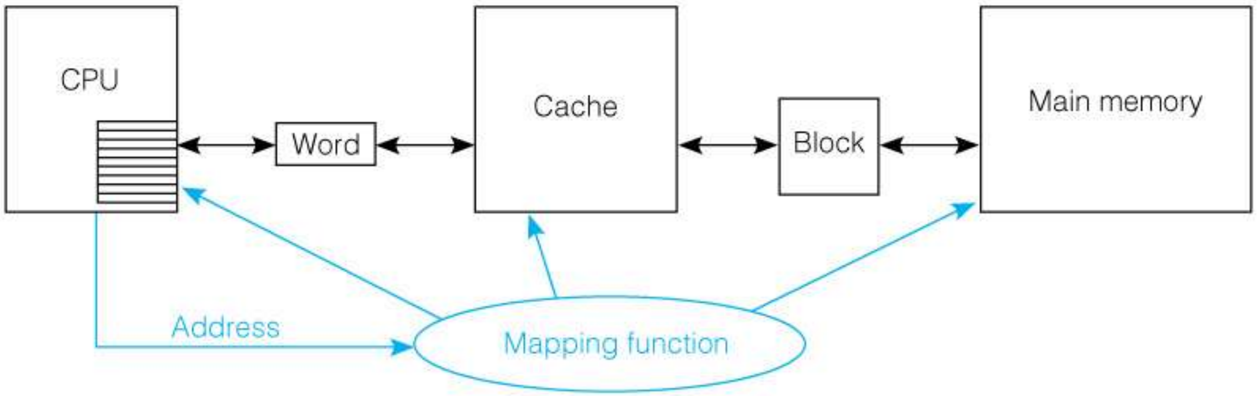
2-Level Hierarchy Design Decisions

- 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?

Cache

- Insertion of high speed memory between CPU and main memory
  - May have more than one cache level
- Caching is usually transparent to programmer
- Caching operations must be handled in hardware
  - Why?
- Cache blocks are item of commerce
  - Block sizes in range of 16-256 bytes

Cache Mapping Function



- Responsible for all cache operations
  - Placement strategy – where to place an incoming block in cache
  - Replacement strategy – which block to replace upon miss
  - Read/write policy – how to handle reads and writes upon cache hits and misses
- Three common mapping functions
  - Associative
  - Direct-mapped
  - Block-set-associative – combination of associative and direct-mapped

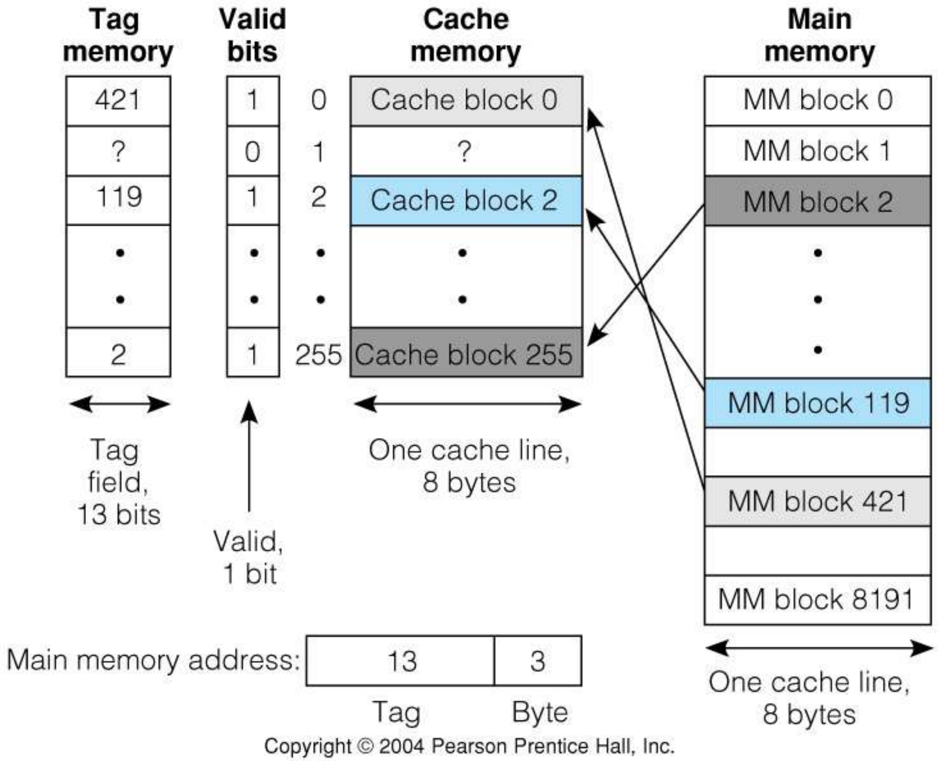
Memory Fields and Address Translation

- Partition processor main memory address (virtual address)

	31	0
32-bit Address	32-bits	
	Block Number	Word
Memory Fields	26	6
Example	00 … 00 1001	001011

  - Block field – represents 2^{26} blocks
  - Word field – $2^6 = 64$ word offset in block
  - Example: Word 11 in block 9

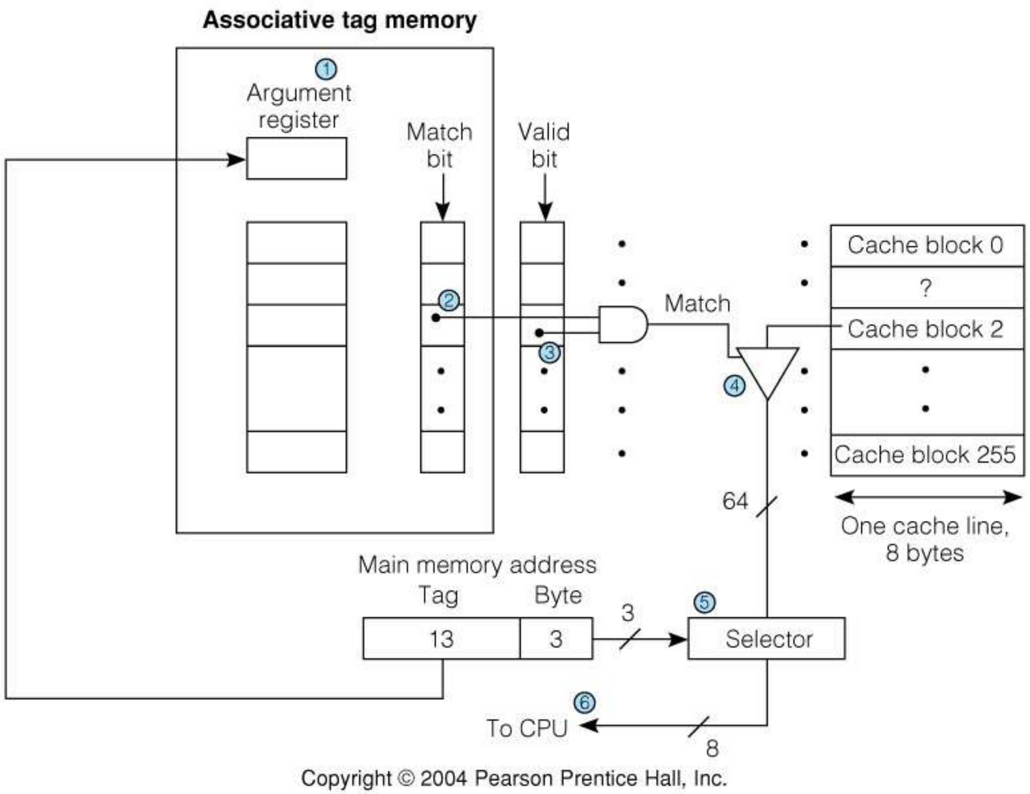
Associative Mapped Cache



- Any block from main memory can be put anywhere in cache

- Example: 16-bit address
- Cache structure:
  - One set of 256 lines – 256 block capacity
    - $2^8 = 256$ 8-byte blocks
  - 3-bits for byte sized word
    - Main memory has $2^{13} = 8192$ 8-byte blocks
  - 256 x 13-bit tag memory
    - Indicates block number in cache position
  - 256 x 1-bit valid memory
    - Indicates if cache location has a value

Associative Cache Operation



- All cache location searched simultaneously
  - Associative (content addressable) memory

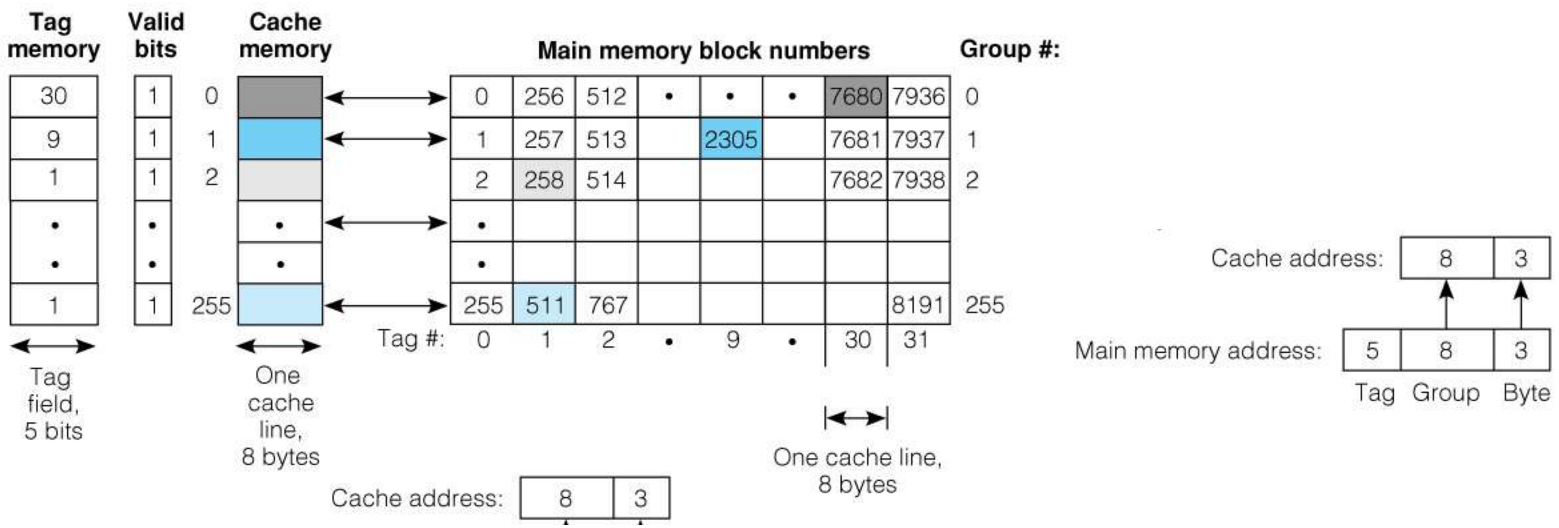
1. Argument register of the associative tag memory is filled
2. All tag memory contents compared in parallel to find match
3. Check a match is also valid
4. Gate cache memory item
5. Offset field used to select the byte (word) of interest

Properties of Associative Cache

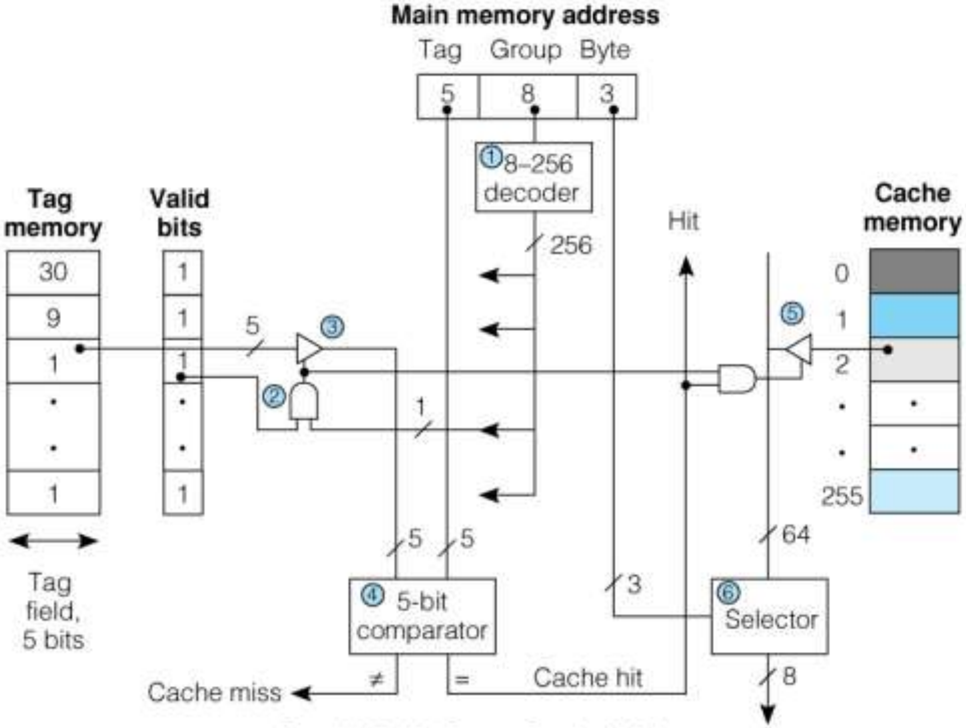
- Advantage
  - Most flexible mapping because a main memory block can go anywhere in the cache
- Disadvantage
  - Large tag memory required
  - Must search entire tag memory simultaneously → lots of hardware required
  - Replacement policy when cache is full causes issue

Direct Mapped Cache

- Divide main memory into sets
  - All blocks in a set (group) can go into only one cache location
- Example: 16-bit main memory address
  - 256 x 8-byte cache
  - The number of cache lines determines the number of sets
  - Cache only examines single group



Direct Mapped Cache Operation



- Single cache location references set memory locations by group number

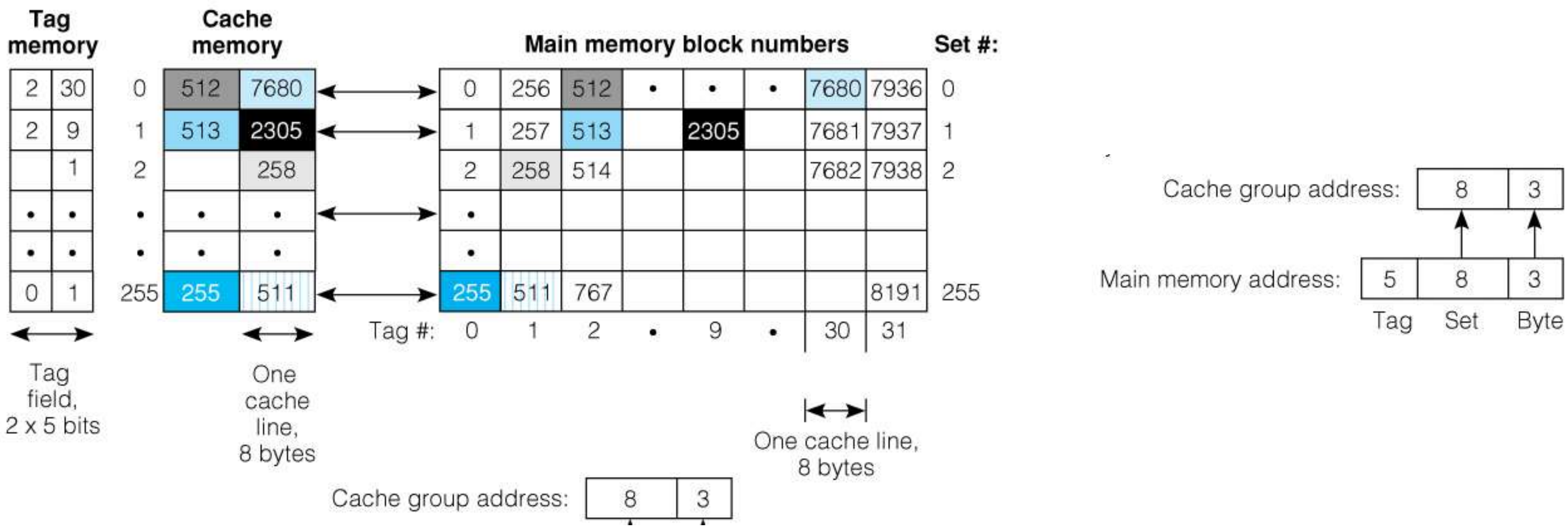
1. Decode the group number to select cache location
2. Check if cache has valid data
3. Gate out tag field (offset in group)
4. Compare address tag with cache tag
5. On hit, gate out cache line
6. Use word field offset to select desired word

Properties of Direct Mapped Cache

- Advantage
  - Requires less hardware than associative
  - Simple (trivial) replacement policy
- Disadvantage
  - Simple replacement policy
    - Restrictive – poor use of cache space
    - Thrashing – two blocks from the same group that are frequently referenced will compete for the same cache location
      - Cause frequent switching of cache data and performance degradation

Block-Set-Associative Cache

- Compromise between associative and direct-mapped to allow several cache blocks for each memory group
- Example: 2-way set associative cache
  - A set of 2 cache values per group
    - 256 x 2 x 8-byte cache
    - 256 sets of 2 lines each
  - Operation is same as direct-mapped
    - Must do associative comparison between tag and cache memory
    - Copy of direct mapped hardware for each set



Intel Pentium Cache

- 2 separate caches
  - 1 instruction, 1 data
  - 2-way set associative caches
  - A cache is 8K = 2^{13} bytes in size
  - 2^5 = 32 bytes per line
- Group (set) calculation
  - 2-way x 32 bytes = 64 = 2^6 bytes/set
  - $2^{13}/2^6 = 2^7 = 128$ groups
- Tag field – 32– 5– 7 = 20 bits

	Tag	Set	Word
Memory Fields	20	7	5
	31		0

Cache Read/Write Policies

- Hit policies
  - Write-through – updates both cache and main memory upon each write
  - Write-back – updates only cache
    - Update main memory only upon removal of block
    - Dirty bit is set upon first write to indicate block must be written back
- Miss Policies
  - Read miss – bring block in from main memory
    - Forward word as brought into cache
    - Wait until entire line is filled then repeat cache request
  - Write miss
    - Write allocate – bring block into cache, then update
    - Write-no allocate – write word to main memory without bringing block into cache

Block Replacement Strategies

- Not needed with direct-mapped cache
- Least recently used (LRU)
  - Track cache usage with counter
  - Each block access causes
    - Clear counter of accessed block
    - Increment counters with values less than block being accessed
    - All others remain unchanged
  - When set is full, remove line with highest count
- Random replacement – replace block at random
  - Actually effective strategy in practice

Cache Performance

- Recall: Access time
  - $t_a = ht_p + (1-h)t_s$
  - Primary is cache
  - Secondary is main memory
- Define speedup as ratio of access times
  - $S = \frac{T_{wo}}{T_w}$
  - T_{wo} is time without cache
  - T_w is time with cache
- Example: improved cache hit rate, 20 nsec cache and 70 nsec main memory
  - $h = 0.91 \rightarrow 0.96$
  - $T_w = 0.91(20) + (1-0.91)(70) = 24.5$
  - $T_w = 0.96(20) + (1-0.96)(70) = 22.0$
  - $S = \frac{T_{wo}}{T_w} = \frac{24.5}{22.0} = 1.11$

Virtual Memory

Paging and Block Placement

- Page – commonly used name for a disk block
- Page fault – synonymous with a miss
- Demand paging – pages moved from disk to main memory only when a word in the page is requested by the processor
- Block placement/replacement decisions must be made each time a block is moved
  - Placement – where a block should go
  - Replacement – what blocks can be removed to make room for new block