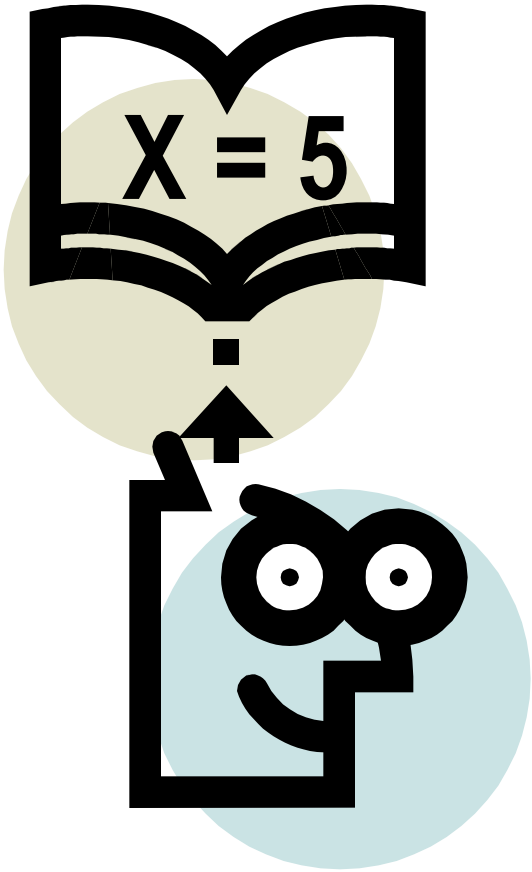
Lecture 16

Memory



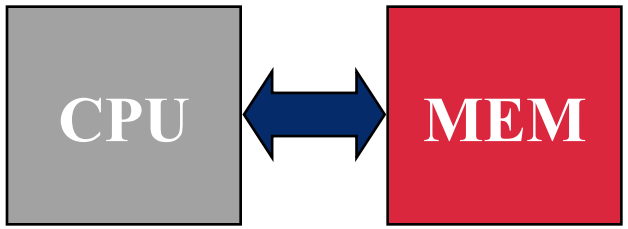
Adapted from slides originally developed by Profs. Hardavellas, Hill, Falsafi, Marculescu, Patterson, Rutenbar and Vijaykumar of Northwestern, Carnegie Mellon, Purdue, UC-Berkley, UWisconsin

Review: Improving Cache Performance

- **Boiled it all down to three basic types of optimizations**
  - O1: Reducing the cache miss rate
    - Larger blocks, higher associativity, victim caches, prefetching
  - O2: Reducing the cache miss penalty
    - Loads before stores, sub-block, early restart, 2nd level caches
  - O3: Reducing the time to hit in the cache
    - Small caches, address translation

Today's Menu:

- **Basic Memory Technology**
  - SRAM—Static RAM
  - DRAM—Dynamic RAM



Random Access Memory (RAM) Technology

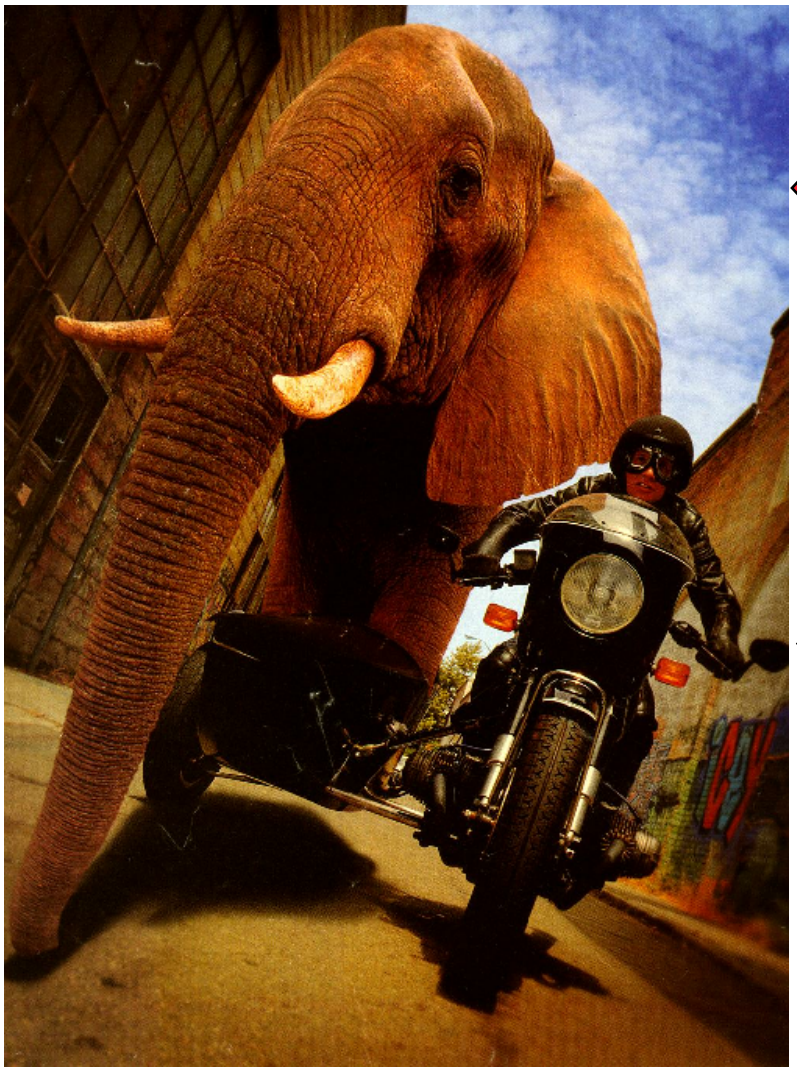
- **Why do we need to know about RAM technology?**
  - Processor performance is usually limited by memory bandwidth
  - As IC densities increase, lots of memory fits on processor chip
  - Tailor on-chip memories to very specific needs
    - Instruction cache
    - Data cache
    - Write buffer

- **What makes RAM different from a bunch of flip-flops?**
  - **Density**: RAM is much more dense
  - **Speed**: depending on how you design it, RAM may be fast, or very slow

Memory Technologies

- **Random Access Memories**
  - **DRAM**: *Dynamic* Random Access Memory
    - High density, low power (0.1–0.5 W active, 0.25–10 mW standby), cheap, slow
    - Dynamic: needs to be "refreshed" regularly
    - Dynamic means "***I write a bit, and if I ignore it long enough, it will evaporate***"
    - Dynamic also means "***I read a bit, and then I have to write it back to make sure it is still correctly stored there***"
  - **SRAM**: *Static* Random Access Memory
    - Low density (SRAM cell is 4-8x bigger than DRAM cell)
    - Fast (SRAM is 8-16x faster than DRAM)
    - High power, expensive
    - Static: content will last "forever"(until lose power)
    - Static means "***I write a bit, and it stays there. Period***"
- **What gets used where (typically)?**
  - Main memory is **DRAM**: you need it big, so you need it cheap
  - CPU cache memory is **SRAM**: you need it fast, so it's more expensive, so it's smaller than you would usually want due to resource limitations

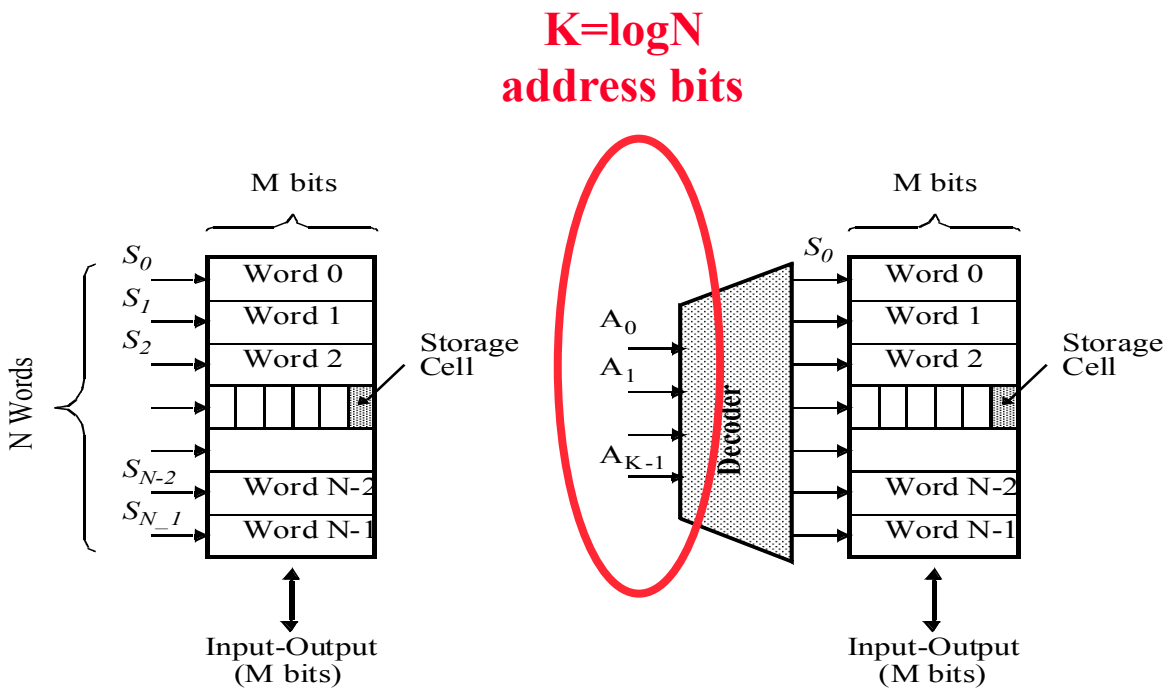
Illustrative Analogy…



This is your CPU's
Main memory subsystem
You want it big

This is your CPU
You want it fast

Memory 2D Array Issues: Decoders



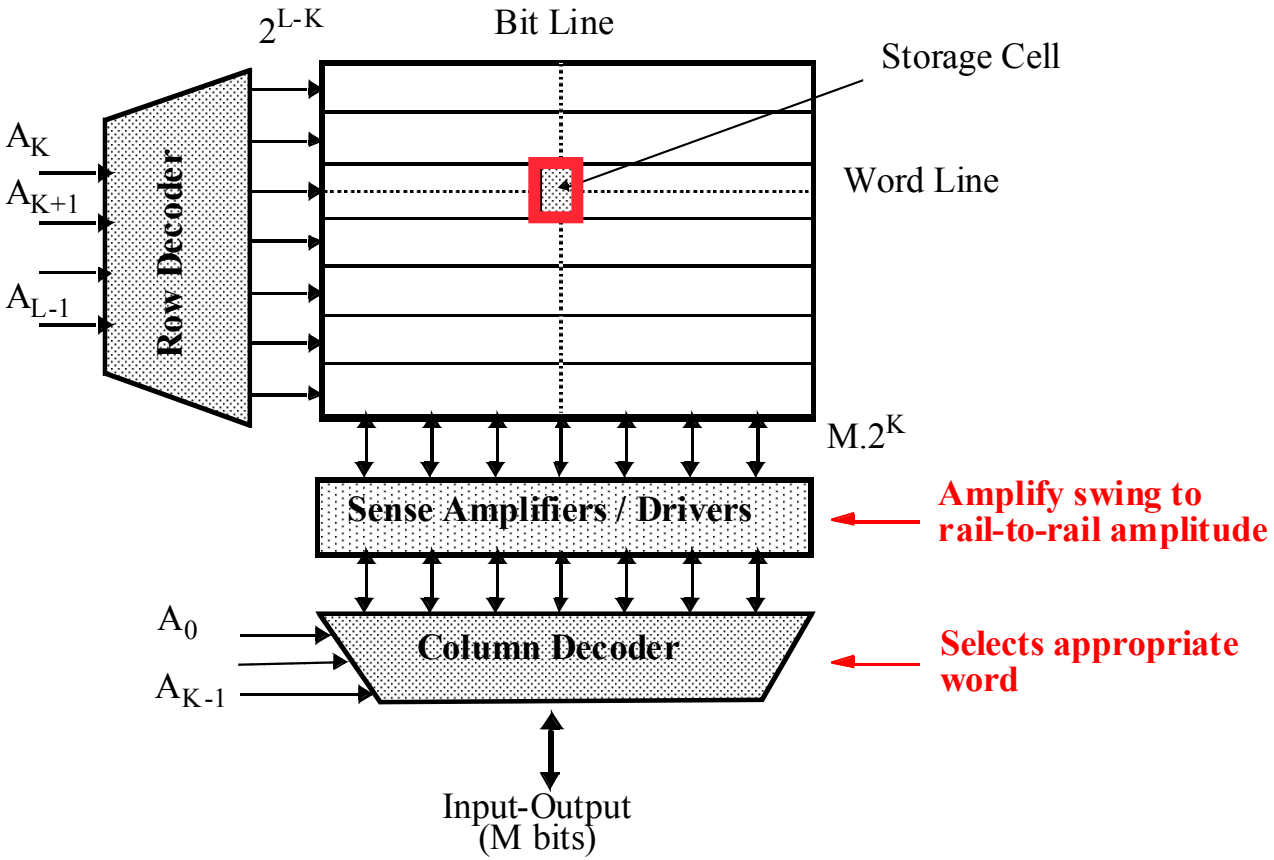
N words => N select signals
Too many select signals

Decoder reduces # of select signals
$K = log_2 N$

Array-Structured Memory Architecture

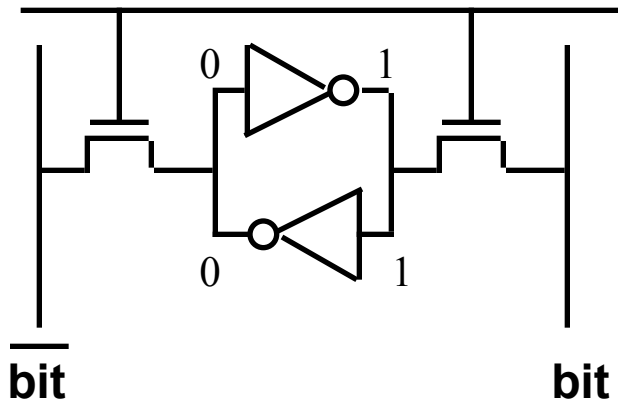
- If you have 1M 1-bit words, you *don't* want a 1M x 1 2D array

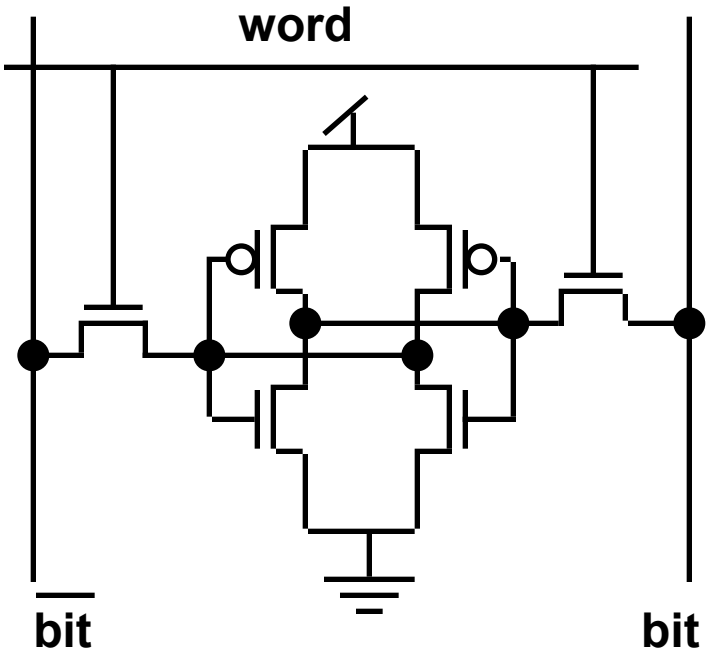
Problem: ASPECT RATIO or HEIGHT >> WIDTH



Static RAM Cell: Circuits View (if you know ckts)

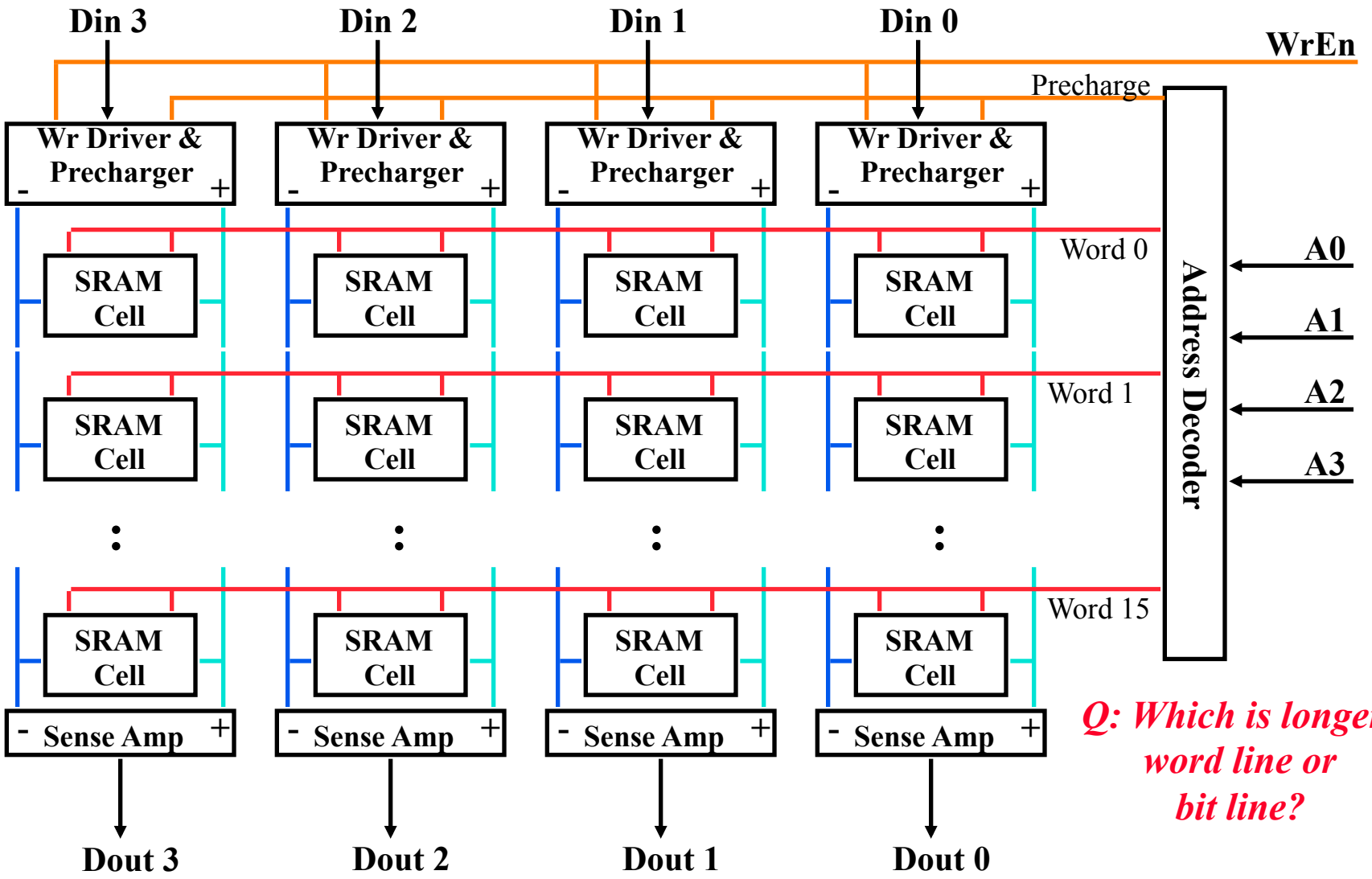
6-Transistor SRAM Cell





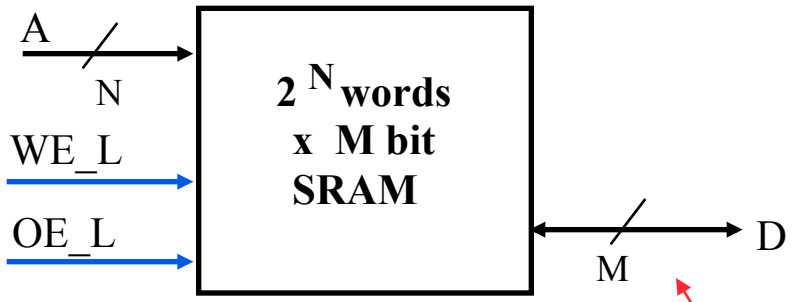
- **Write:**
  1. Drive bit lines (bit = x, $\overline{bit}$ = $\overline{x}$)
  2. Select row (set word line to Vdd)
- **Read:**
  1. Precharge bit and $\overline{bit}$ to Vdd
  2. Select row
  3. Cell pulls one line low
  4. Special analog circuit called a "*Sense amplifier*" (or usually just "sense-amp") on column detects the difference between bit and $\overline{bit}$, which denotes what we read

Typical SRAM Organization: 16-word x 4-bit



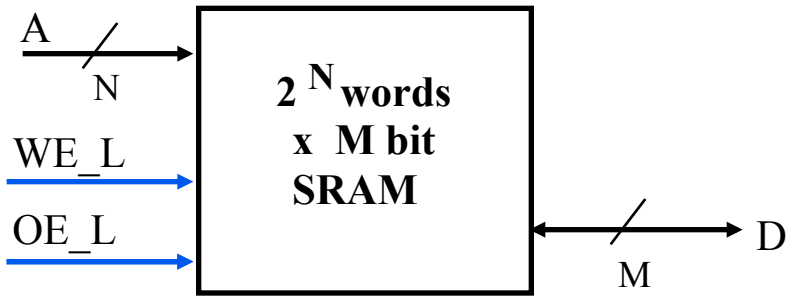
Q: Which is longer: word line or bit line?

Logic Diagram of a Typical SRAM



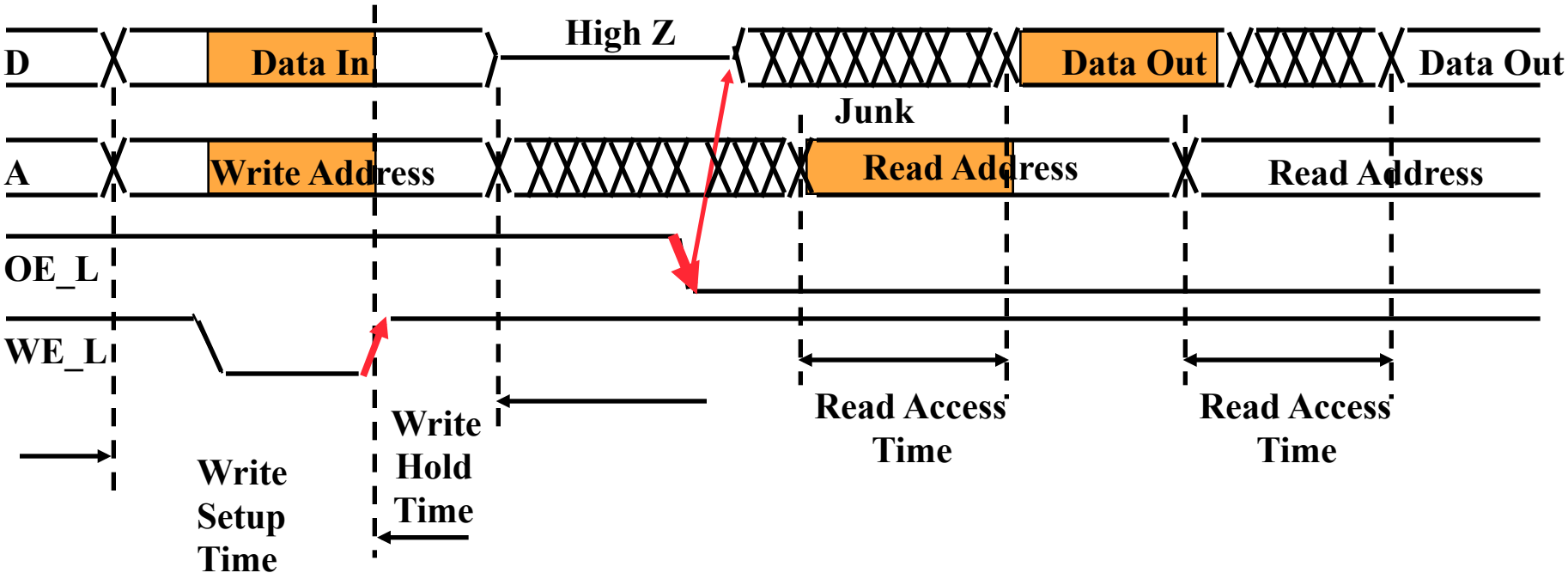
- **Basic plumbing**
  - *WriteEnable* is usually active low (*WE_L*)
  - *Din* and *Dout* are combined to save pins:
- **A new signal, *OutputEnable* (*OE_L*) is needed**
  - *WE_L* is asserted (Low), *OE_L* is de-asserted (High)
    - *D* serves as the data **input** pins
  - *WE_L* is de-asserted (High), *OE_L* is asserted (Low)
    - *D* is the data **output** pins
  - Both *WE_L* and *OE_L* are asserted:
    - Result is **unknown**. ***Don't do that!!!***

Typical SRAM Timing

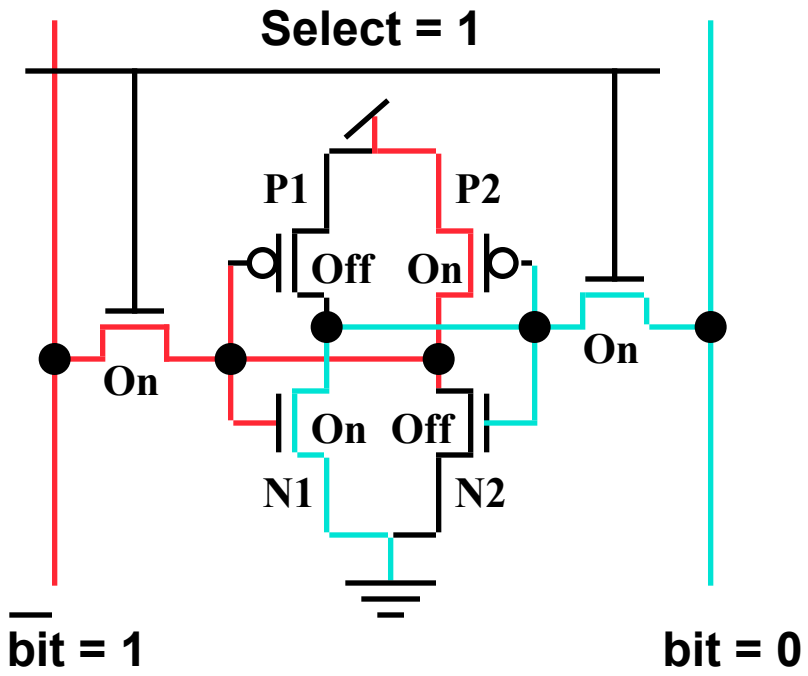


Write Timing:

Read Timing:



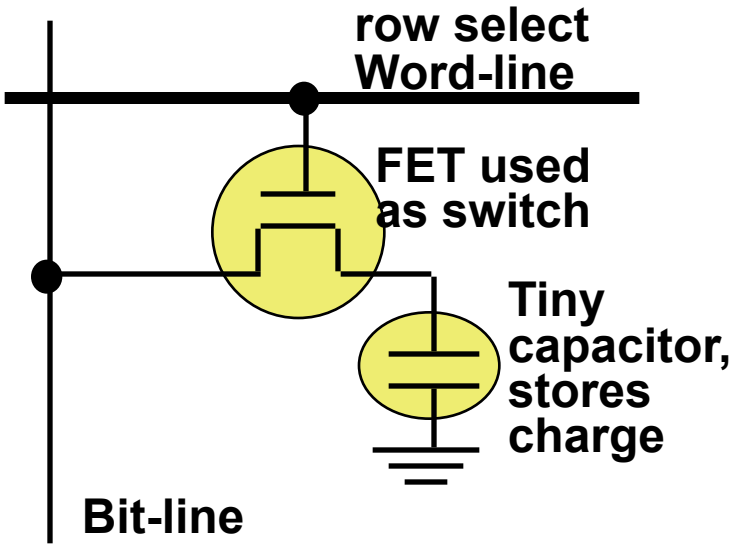
Problems with SRAM



- **Issues**
  - 6 transistors is a lot of area, if want millions of stored bits
  - It's fast, and it's static (ie, it's cross-coupled like a flip-flop, so it regenerates state from inputs)…
  - …but, as a result, it's electrically "on" all the time (depending on ckt design)
  - Can be a major power dissipation issue

Other option: 1-Transistor Memory Cell (DRAM)

- **Write:**
  - 1. Drive bit line
  - 2. Select row (called word-line)
- **Read:**
  - 1. Precharge bit line to Vdd/2
  - 2. Select row
  - 3. Cell and bit line share charges
    - Very small voltage changes on the bit line
  - 4. Sense (fancy sense amp)
    - Can detect changes of ~1 million electrons
  - 5. Write: restore the value
- **Refresh**
  - 1. Just do a dummy read to every cell.



Closer Look at DRAM Cell

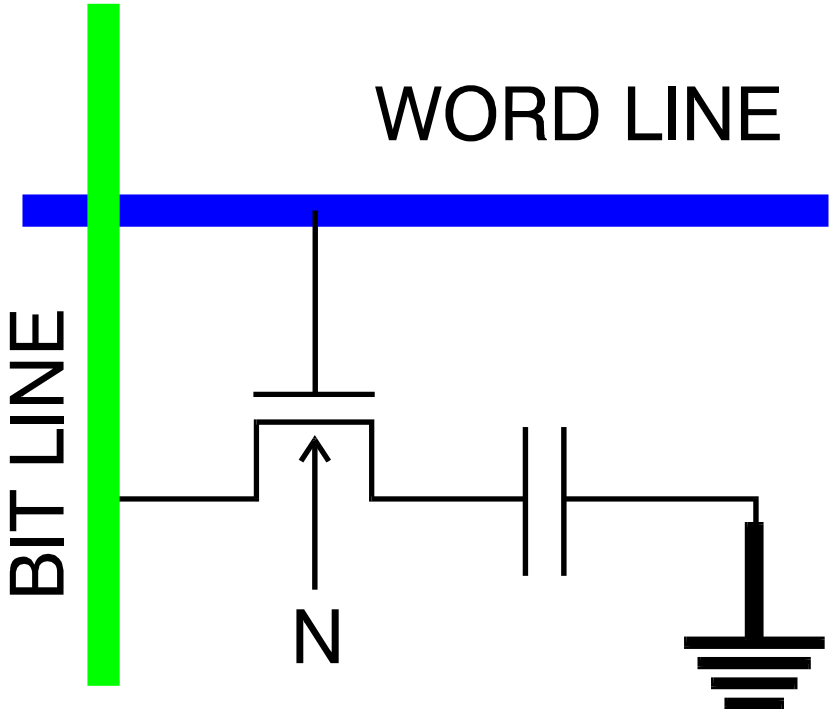
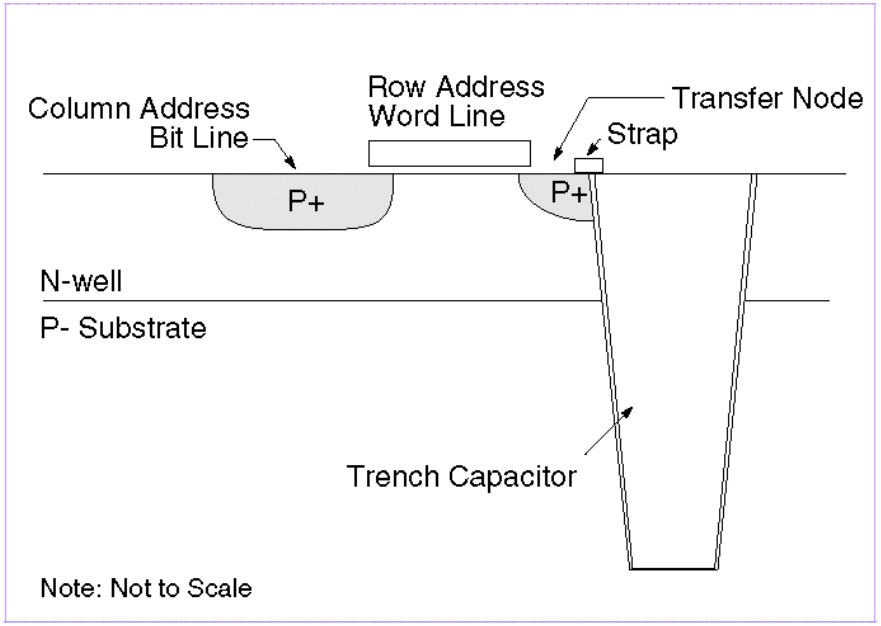
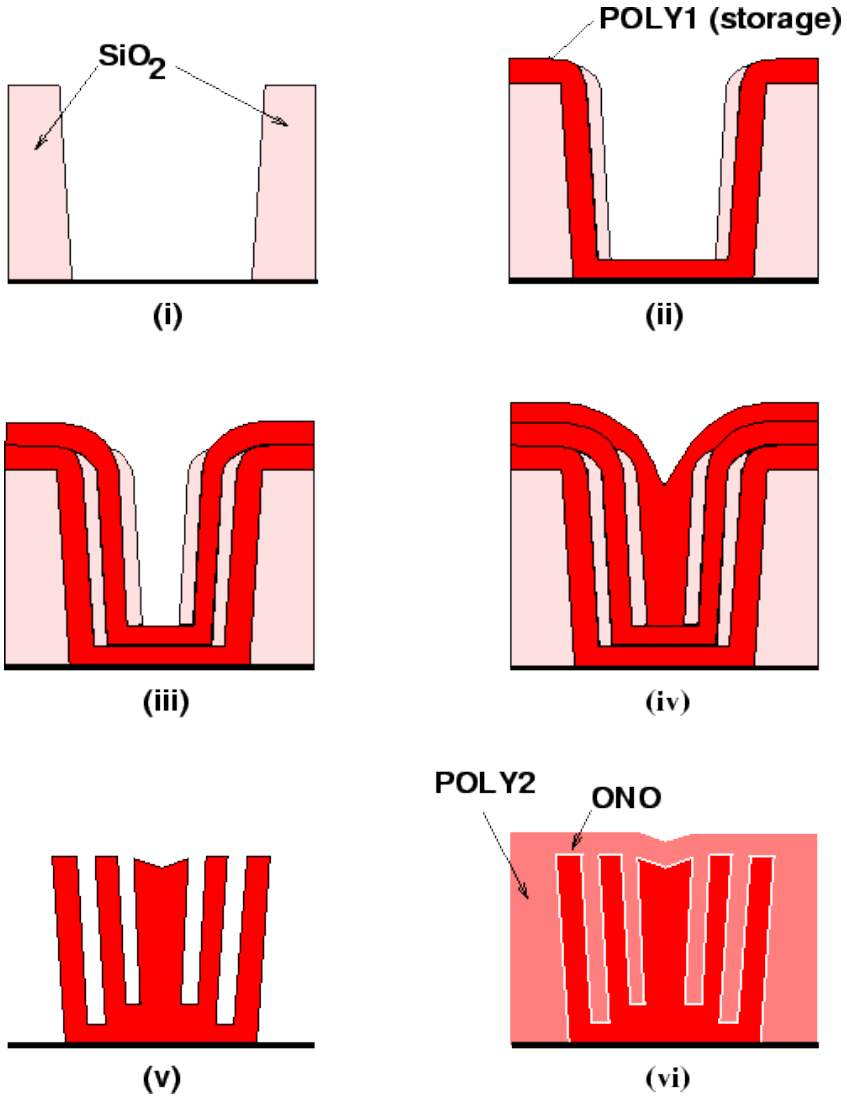


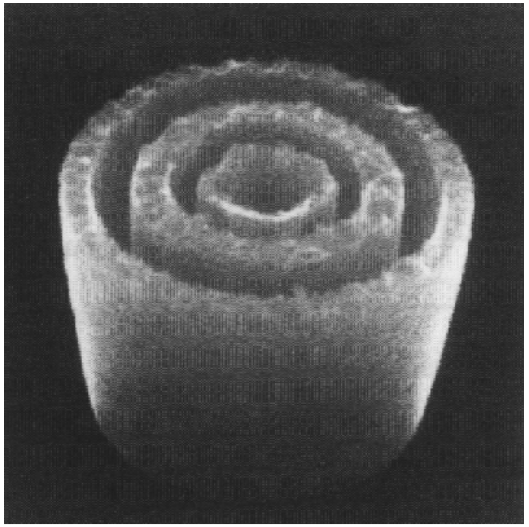
Figure 1: IBM Trench Capacitor Memory Cell

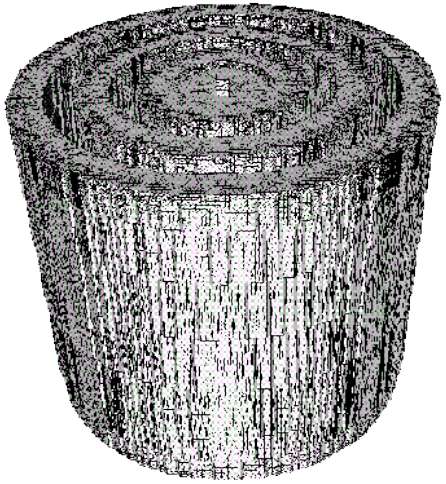


Problem is: you want more and more bits/chip.
So, each bit gets smaller and smaller.
So, the capacitor gets smaller and smaller.
But, the capacitor needs physical area to store enough charge.
Solution: dig a hole in silicon, called a "*trench*"

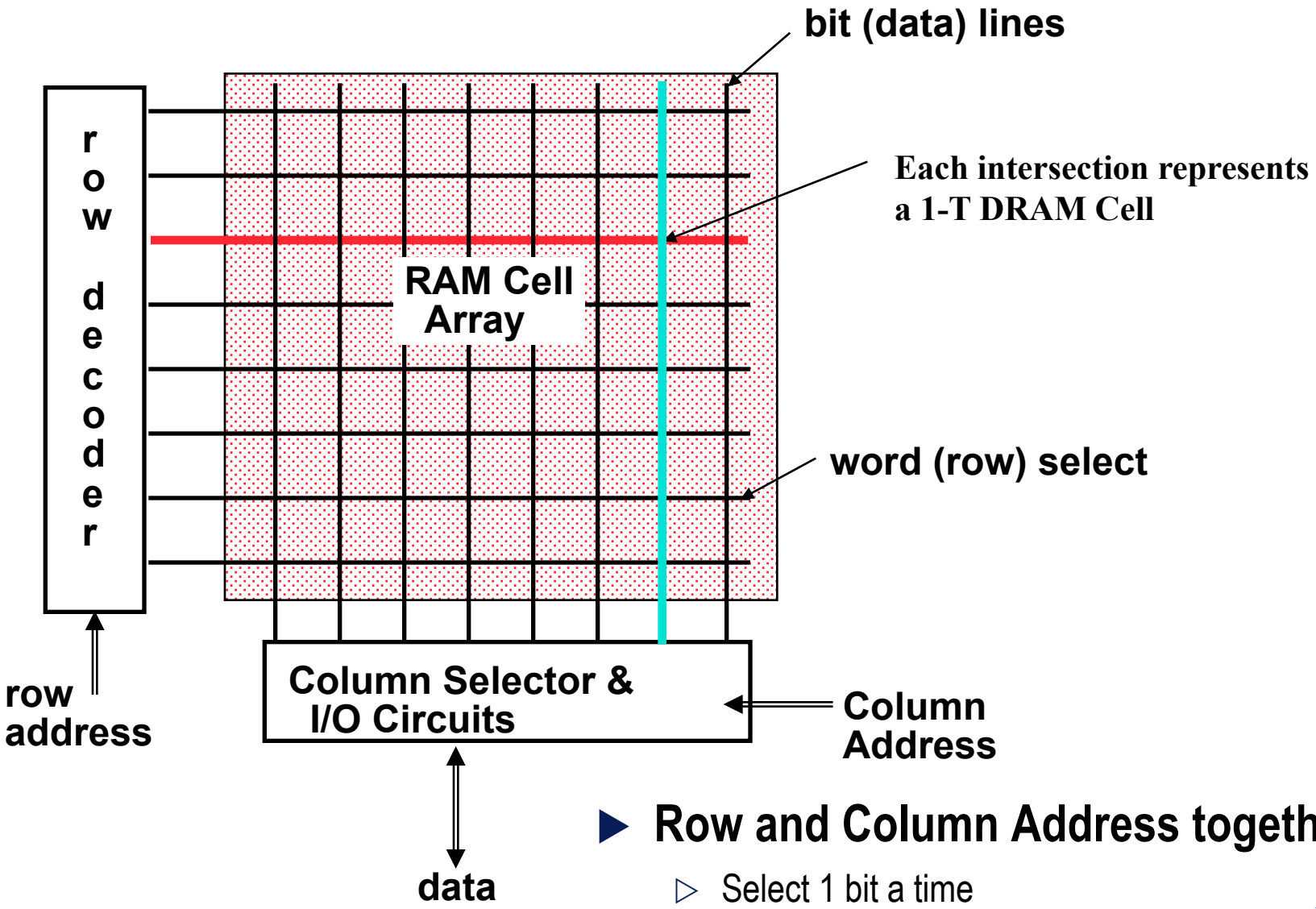
DRAMs with Stack Capacitors







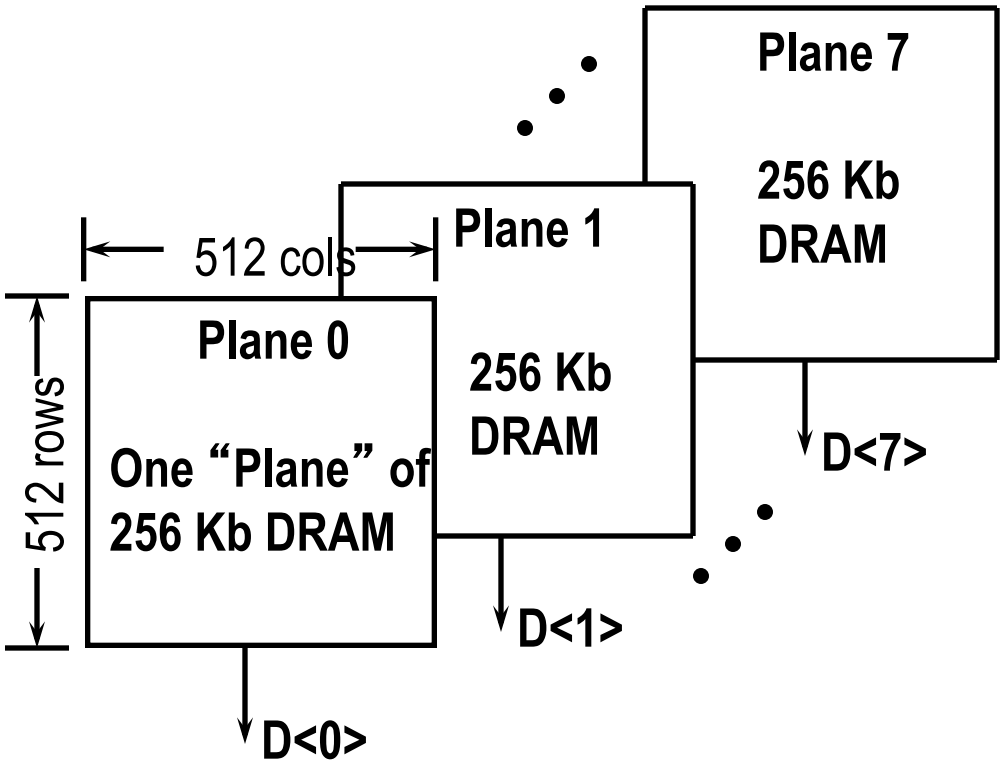
Classical DRAM Organization (square)



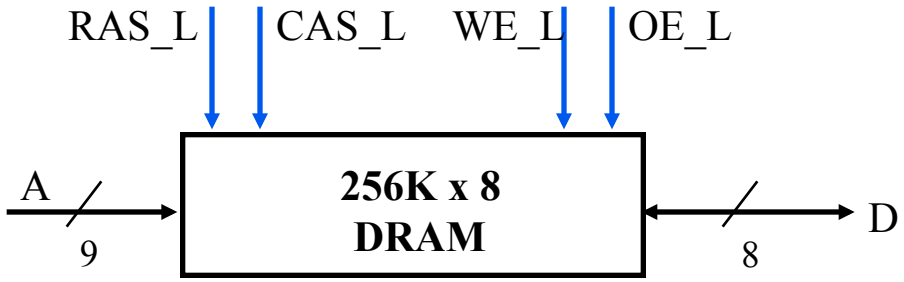
- Row and Column Address together:
  - Select 1 bit a time

Typical DRAM Organization

- **Typical DRAMs: access multiple bits in parallel**
  - Example: 2 Mb DRAM = 256K x 8 = 512 rows x 512 cols x 8 bits
  - Row and column addresses are applied to all 8 planes in parallel



Logic Diagram of a Typical DRAM

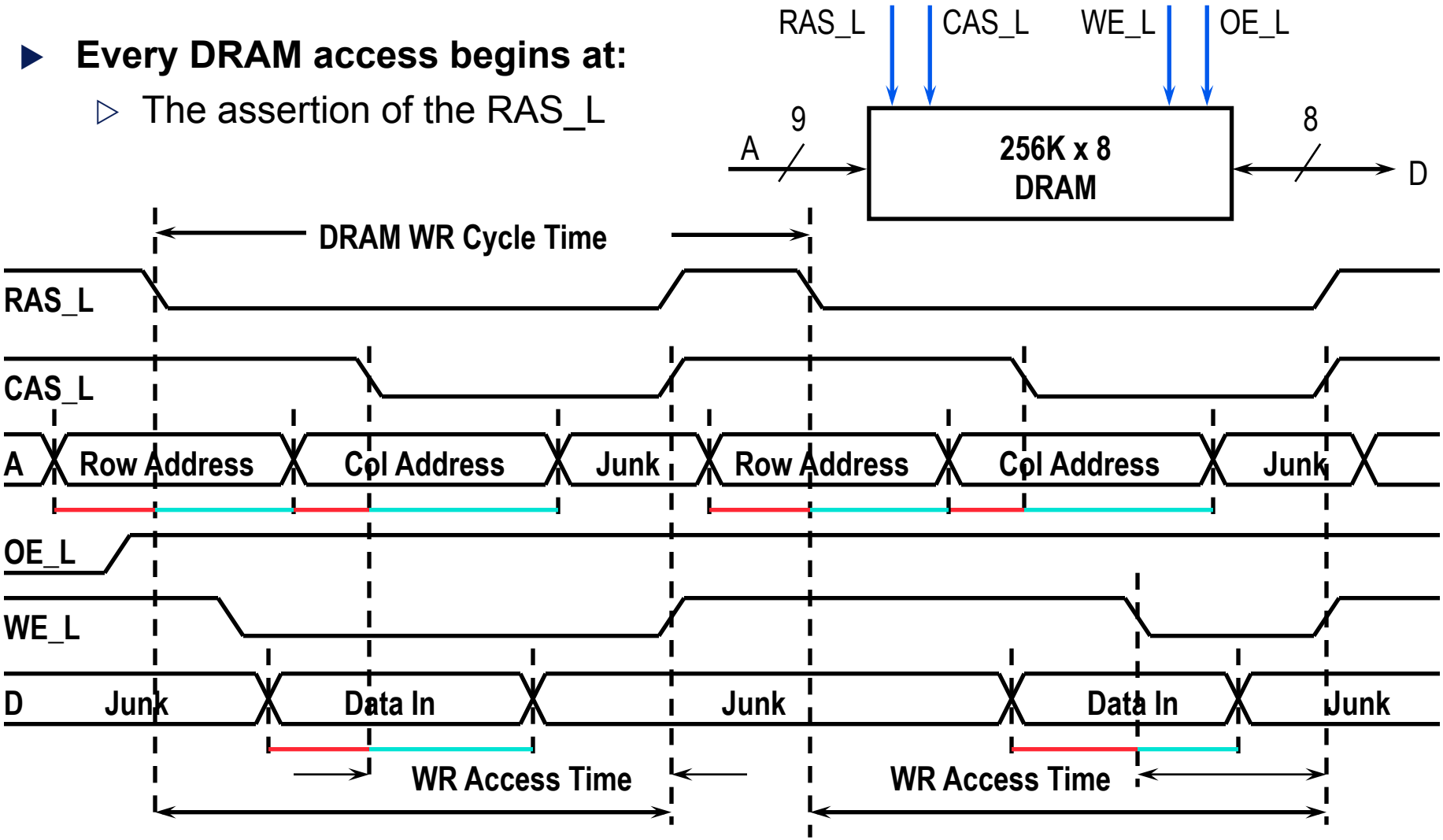


- **Control Signals (*RAS_L, CAS_L, WE_L, OE_L*)**
  - All active low
- ***Din* and *Dout* are combined (*D*):**
  - *WE_L* is asserted (Low), *OE_L* is disasserted (High): *D* serves as the data **input** pin
  - *WE_L* is disasserted (High), *OE_L* is asserted (Low): *D* is the data **output** pin
- **Row and column addresses *share the same pins* (*A*)**
  - *RAS_L* goes low: Pins A are latched in as row address
  - *CAS_L* goes low: Pins A are latched in as column address
  - *RAS/CAS* edge-sensitive

Weird but common on DRAMs…

DRAM Write Timing

- **Every DRAM access begins at:**
  - The assertion of the RAS_L

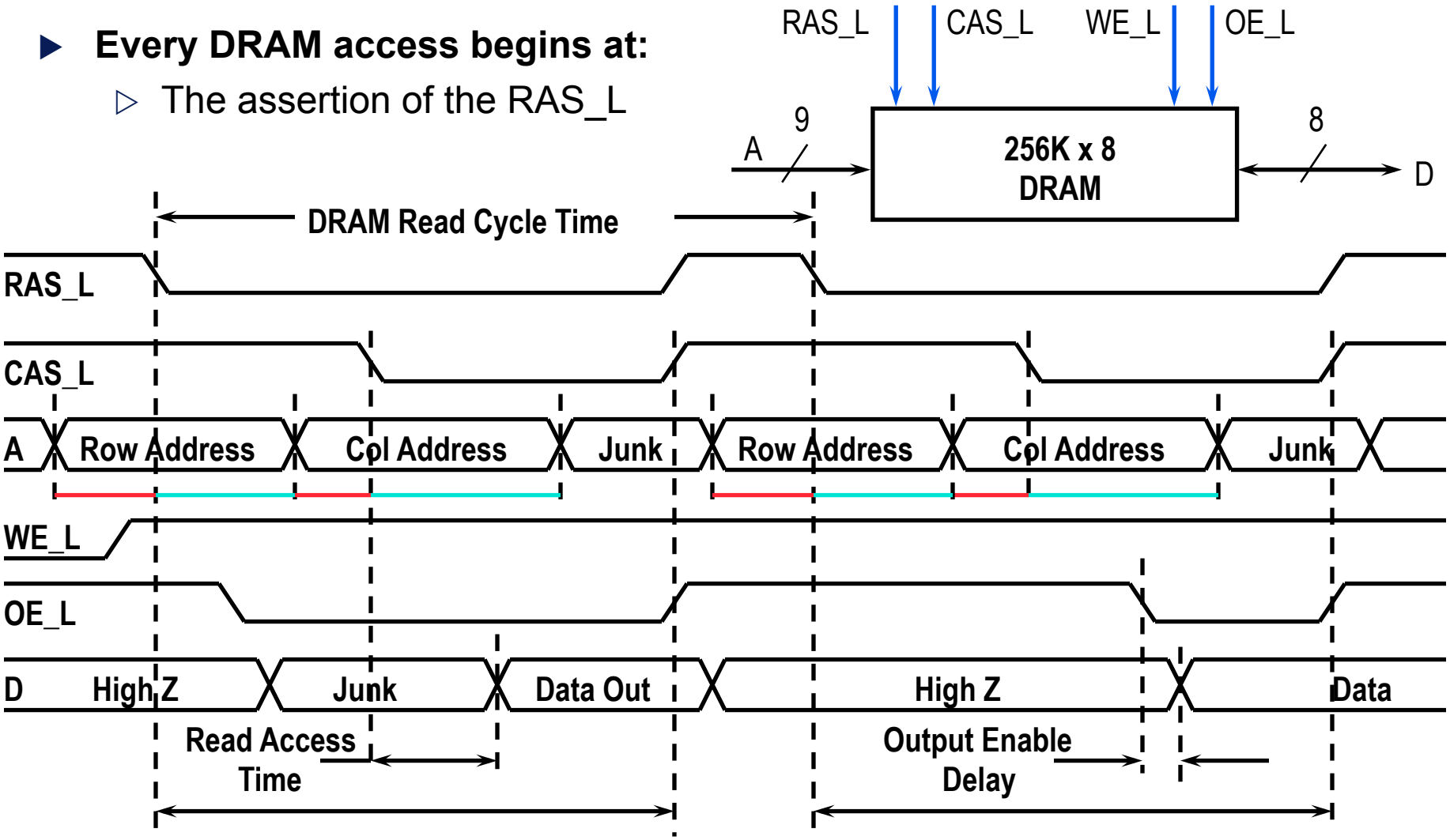


Early Wr Cycle: WE_L asserted before CAS_L

Late Wr Cycle: WE_L asserted after CAS_L

DRAM Read Timing

- **Every DRAM access begins at:**
  - The assertion of the RAS_L

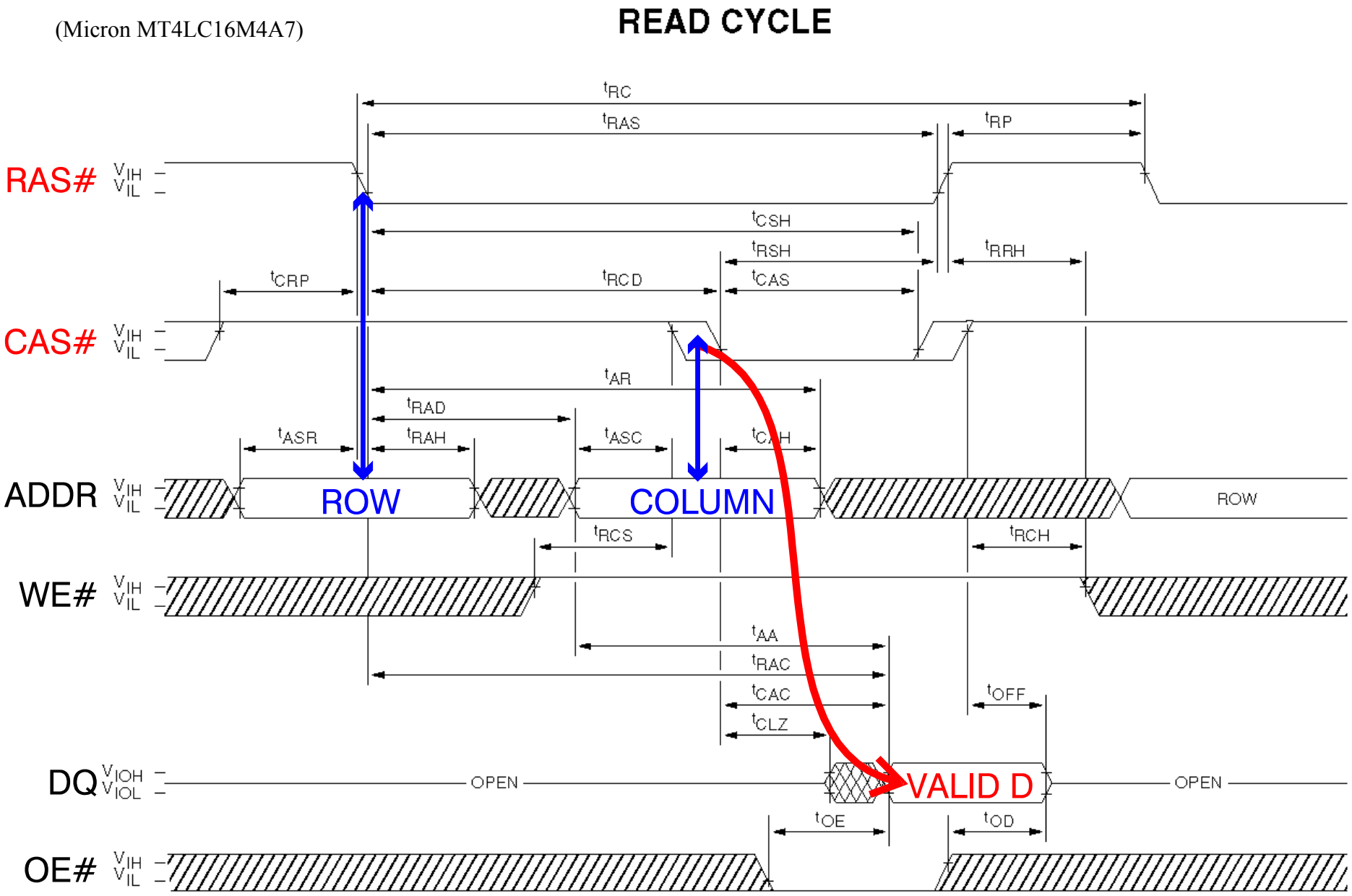


Early Read Cycle: OE_L asserted before CAS_L

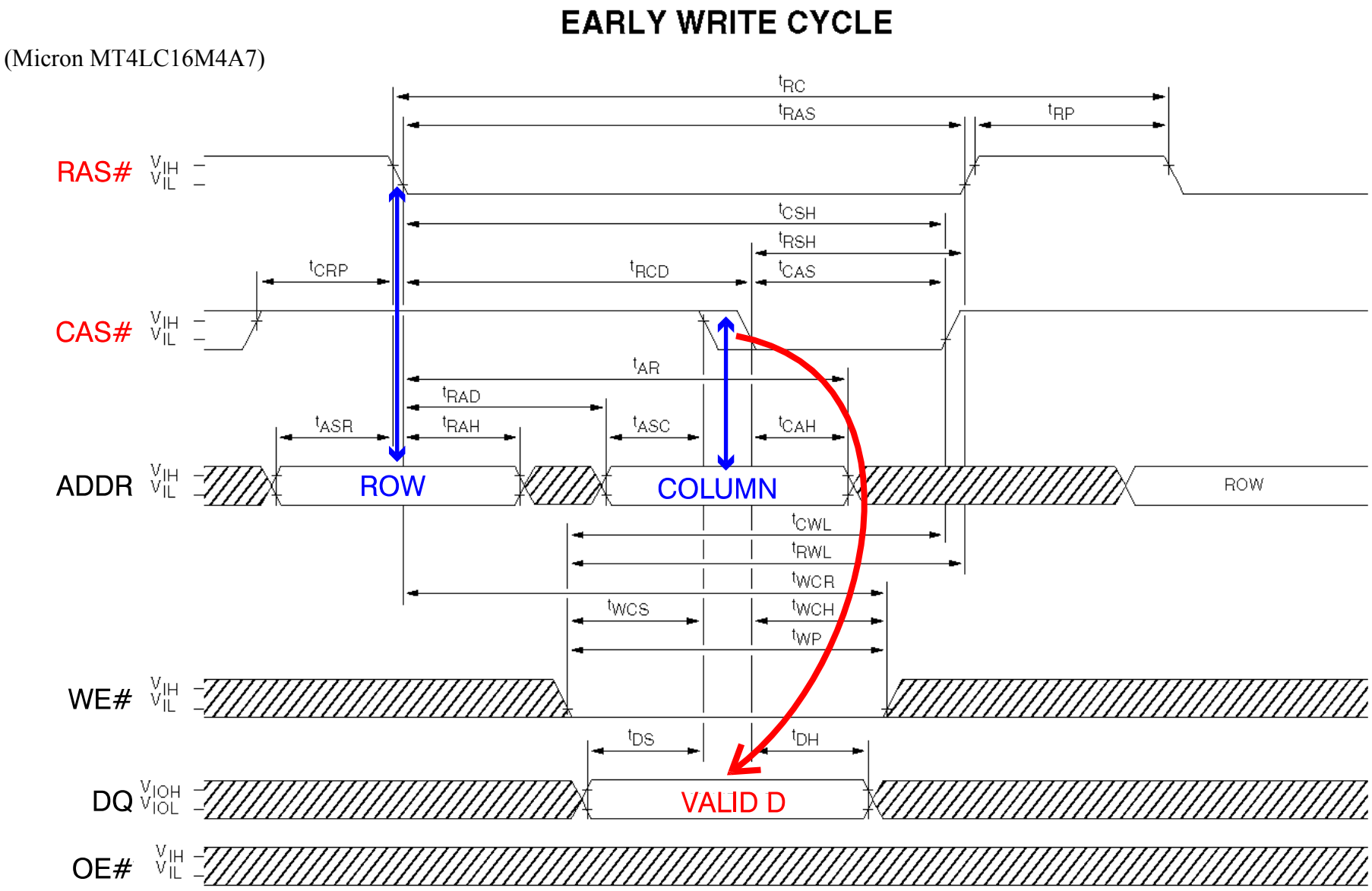
Late Read Cycle: OE_L asserted after CAS_L

EXAMPLE DRAM READ

(Micron MT4LC16M4A7)



EXAMPLE DRAM WRITE

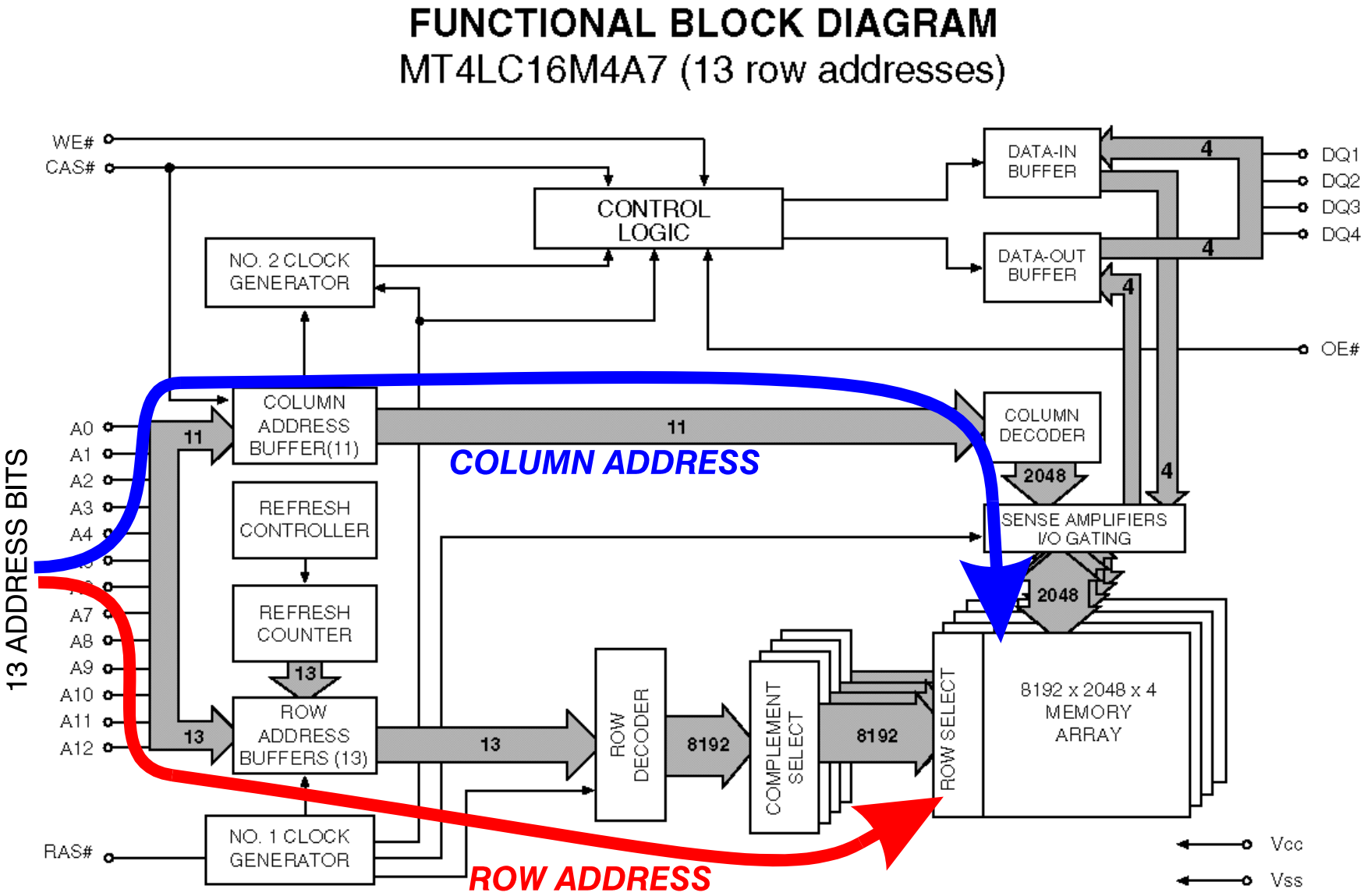


About DRAM Timing

- **It's just plain ugly.**
  - Not at all like a nice friendly flip flop
  - Lots of complex timing on control signals
  - Lots of complex variations

- **Why?**
  - DRAMs are intrinsically "Dump"
  - DRAMs are intrinsically slow
  - Lots of low-level tricks to disguise or mitigate this fact
  - Lots of optimizations for particular hardware applications

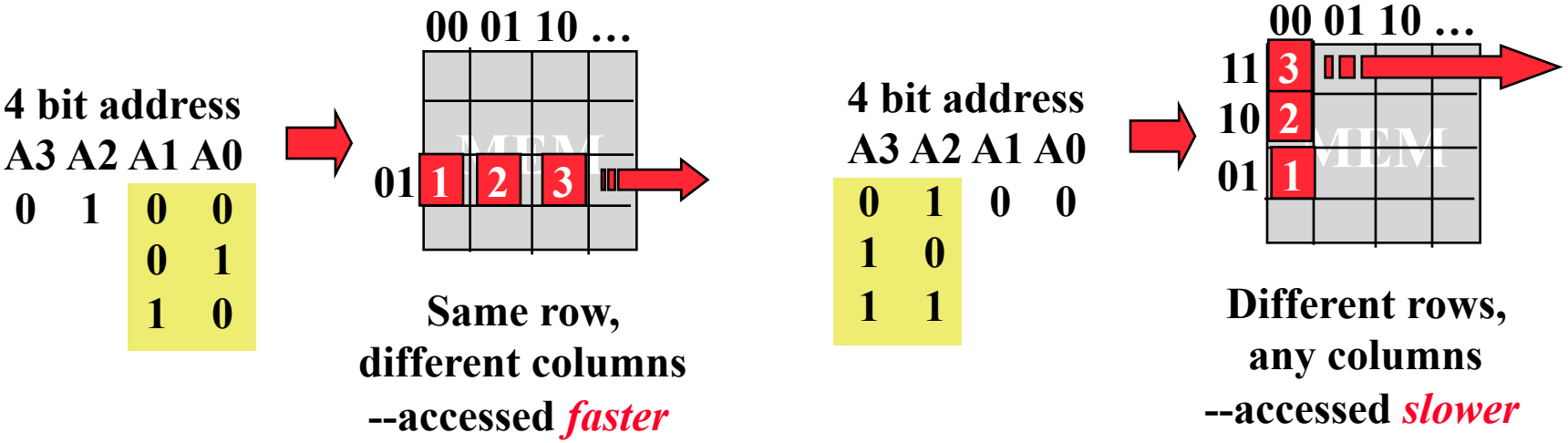
Example 64Mbit DRAM: Micron



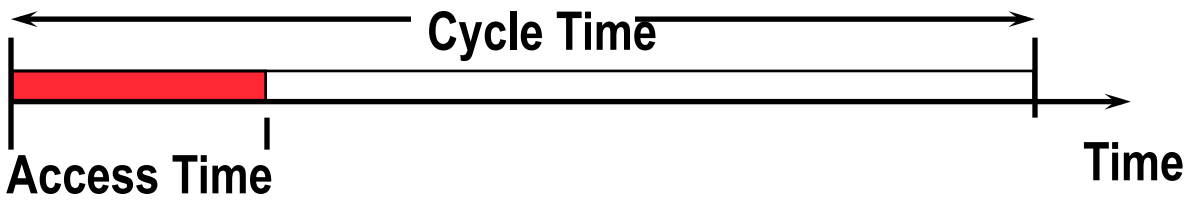
DRAM Performance

- **A 60 ns (t_{RAC}) DRAM can…**
  - …perform a row access only every 110 ns (t_{RC})
  - …perform column access (t_{CAC}) in 15 ns…
  - … but time *between* successive column accesses is at least 35 ns (t_{PC}).
  - (In practice, external address delays and bus wire delays make it 40 to 50 ns)

- **Consequence**



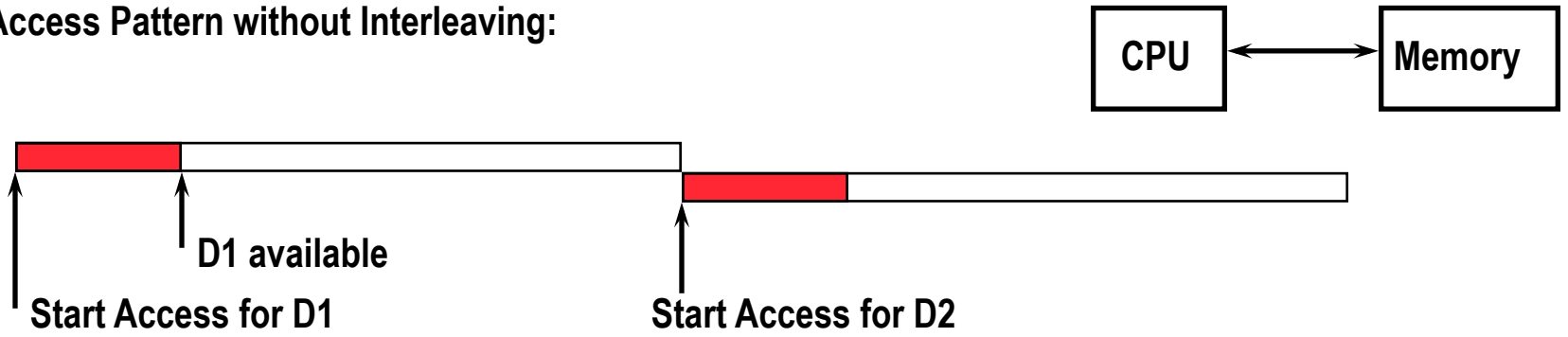
Cycle Time versus Access Time



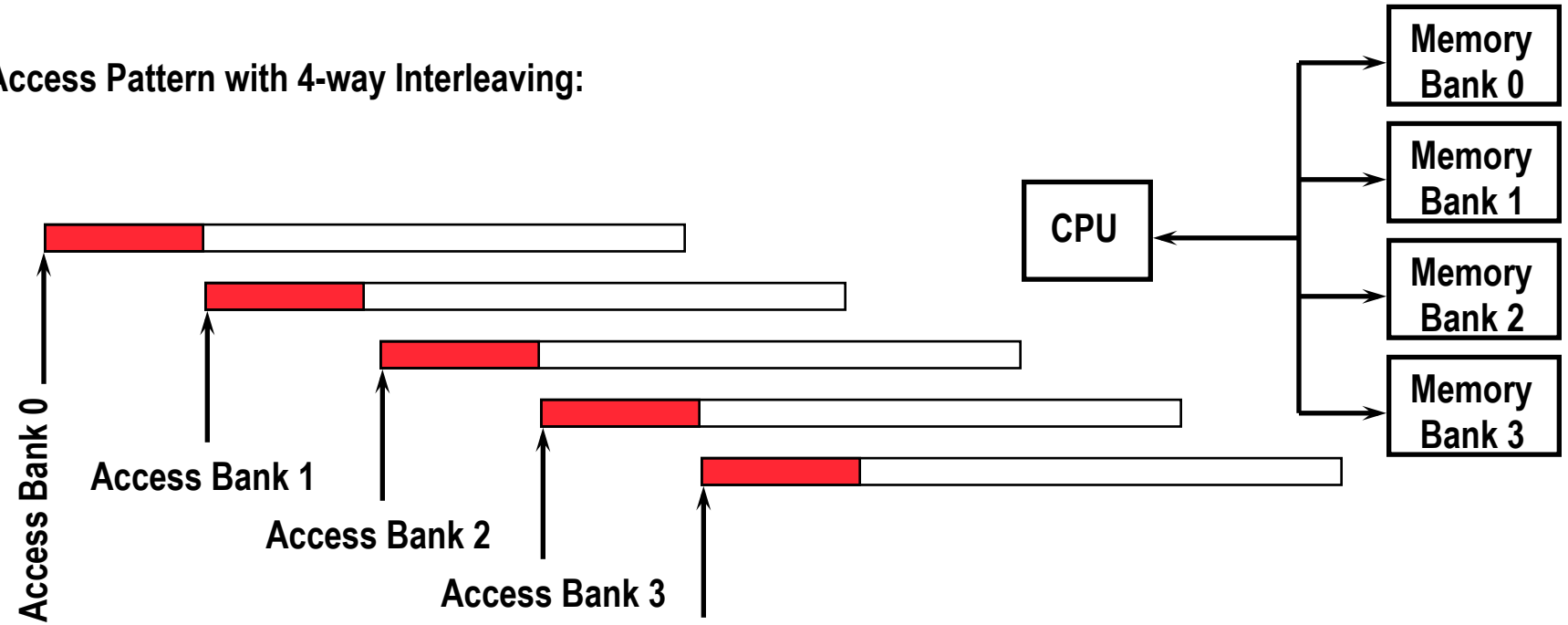
- **DRAM (Read/Write) Cycle Time >> DRAM (Read/Write) Access Time**
- **DRAM (Read/Write) Cycle Time :**
  - How frequent can you initiate an access?
  - Analogy: A little kid can only ask his father for money on Saturday
- **DRAM (Read/Write) Access Time:**
  - How quickly will you get what you want once you initiate an access?
  - Analogy: As soon as he asks, his father will give him the money
- **DRAM Bandwidth Limitation analogy:**
  - What happens if he runs out of money on Wednesday?

Increasing Bandwidth - Interleaving

Access Pattern without Interleaving:

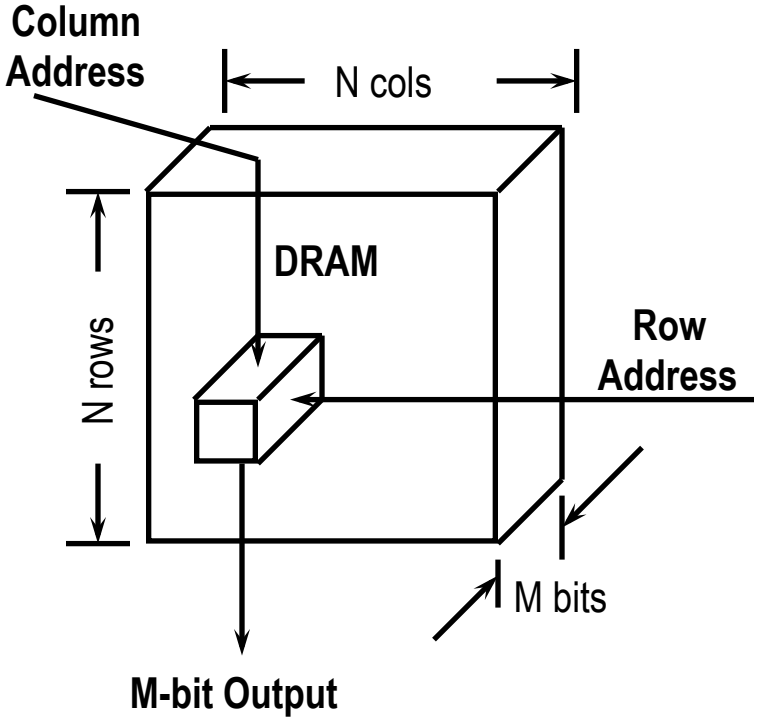


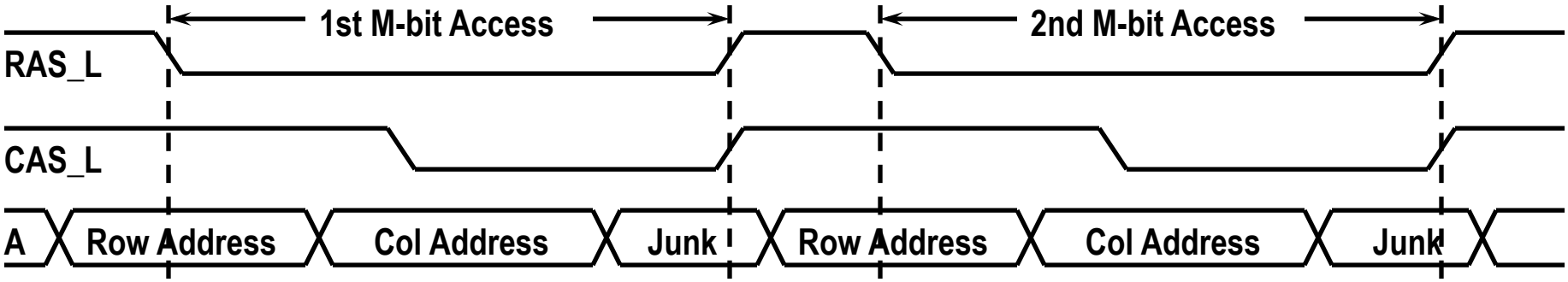
Access Pattern with 4-way Interleaving:



Fast Page Mode DRAM

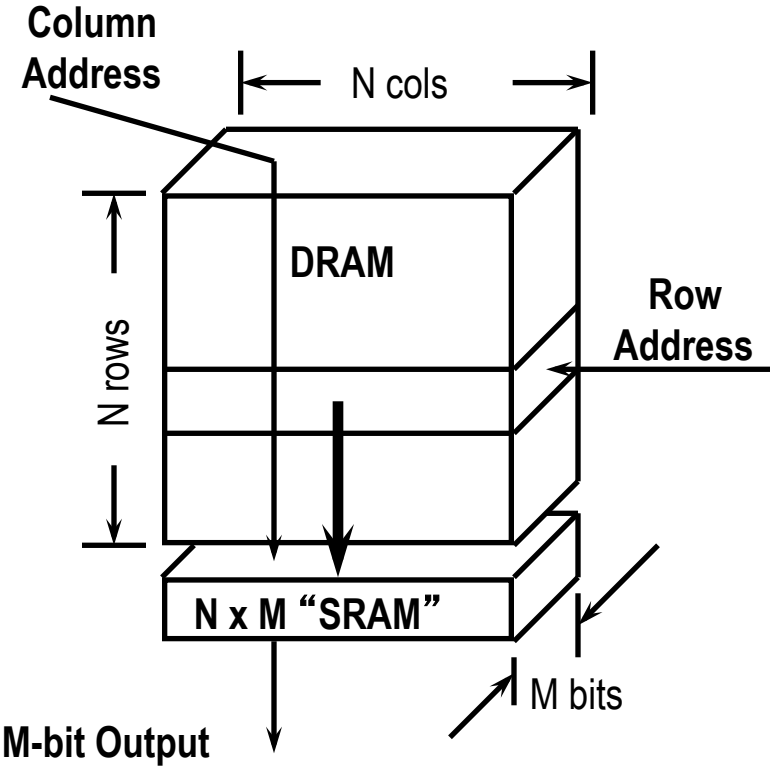
- **Regular DRAM Organization:**
  - N rows x N column x M-bit
  - Read & Write M-bit at a time
  - Each M-bit access requires a RAS / CAS cycle
- **Fast Page Mode DRAM**
  - N x M "register" to save a row

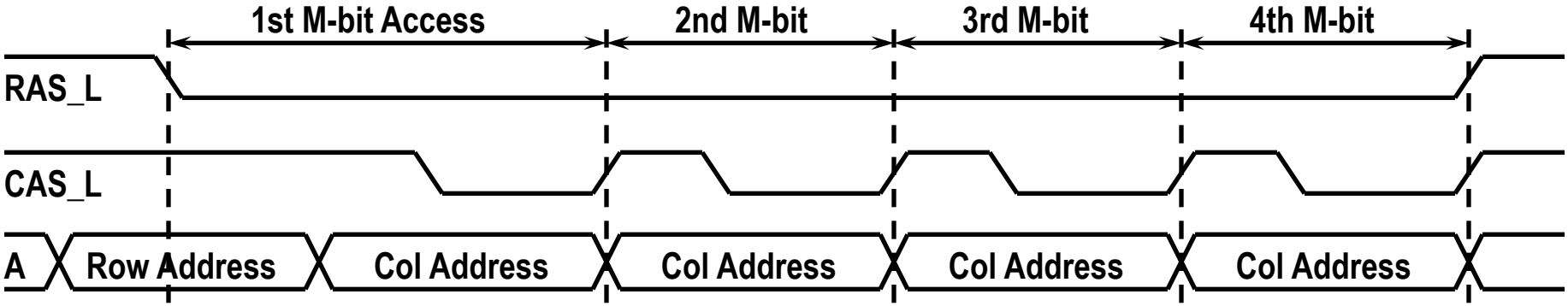




Fast Page Mode Operation

- **Fast Page Mode DRAM**
  - N x M "SRAM" to save a row
- **After a row is read into the register**
  - Only CAS is needed to access other M-bit blocks on that row
  - RAS_L remains asserted while CAS_L is toggled





DDR SDRAM Timing Diagram

- **Use positive and negative edges of clock**
- **Request "bursts" of data**
  - 4-256 words on the bus; don't re-transfer address

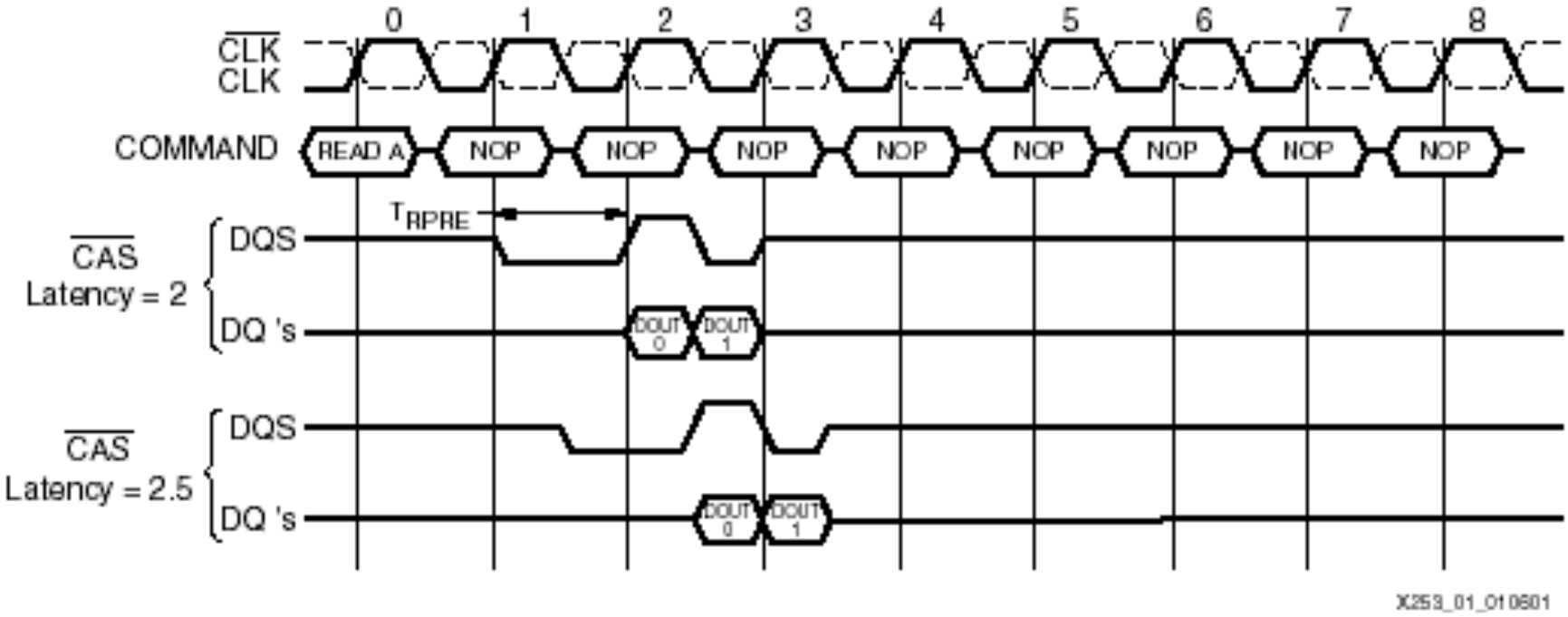


Figure 1: **Burst Read Operation Timing**

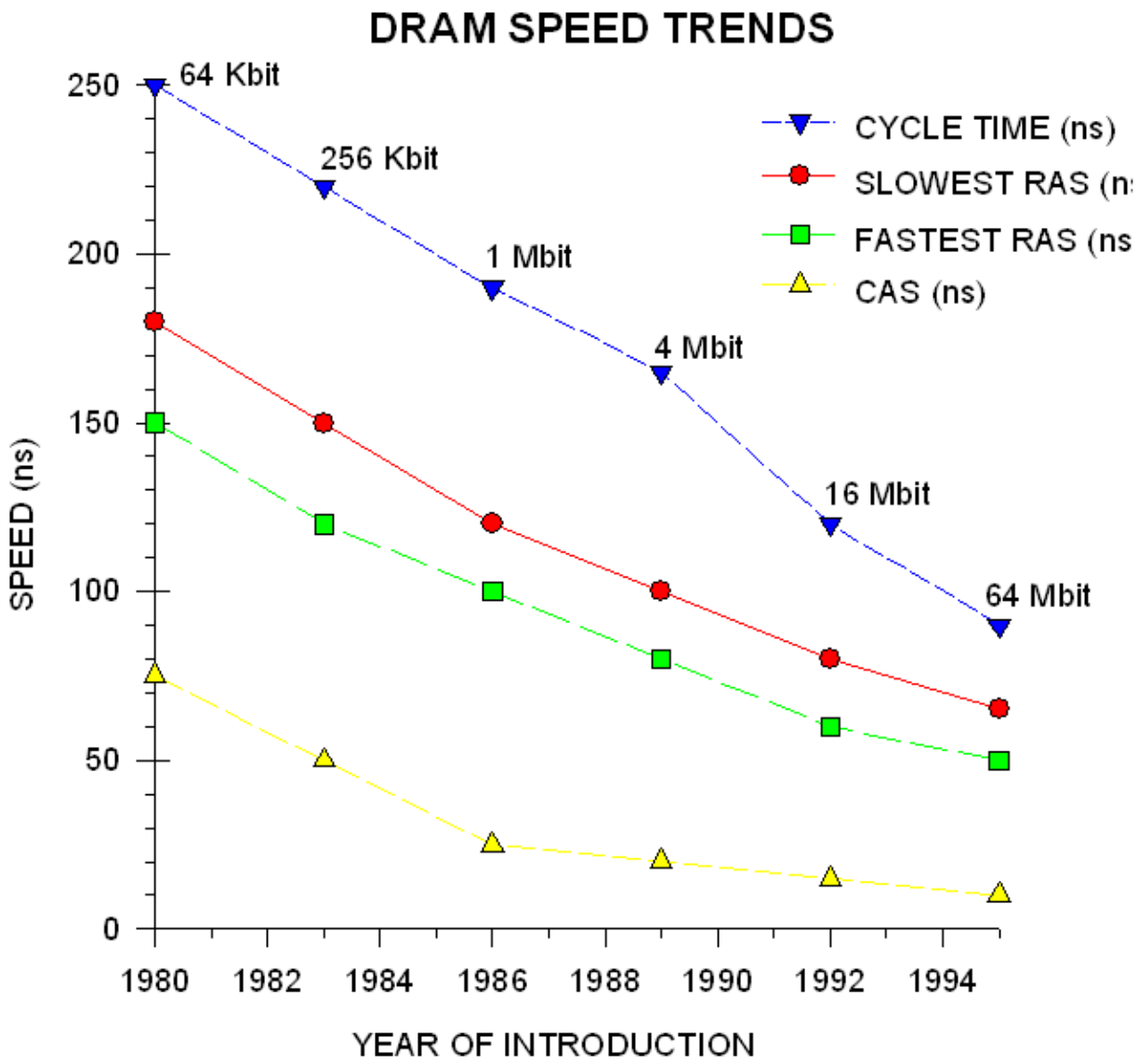
DRAM History

- **DRAMs: capacity +60%/yr, cost –30%/yr**
  - 2.5X cells/area, 1.5X die size in -3 years
  - DRAM fab line costs $1B to $2B
  - Commodity industry: high-volume, low-profit/part, conservative

- **Problems**
  - They’re big and cheap, but they’re still SLOW
  - SLOW is increasingly the difficult problem…

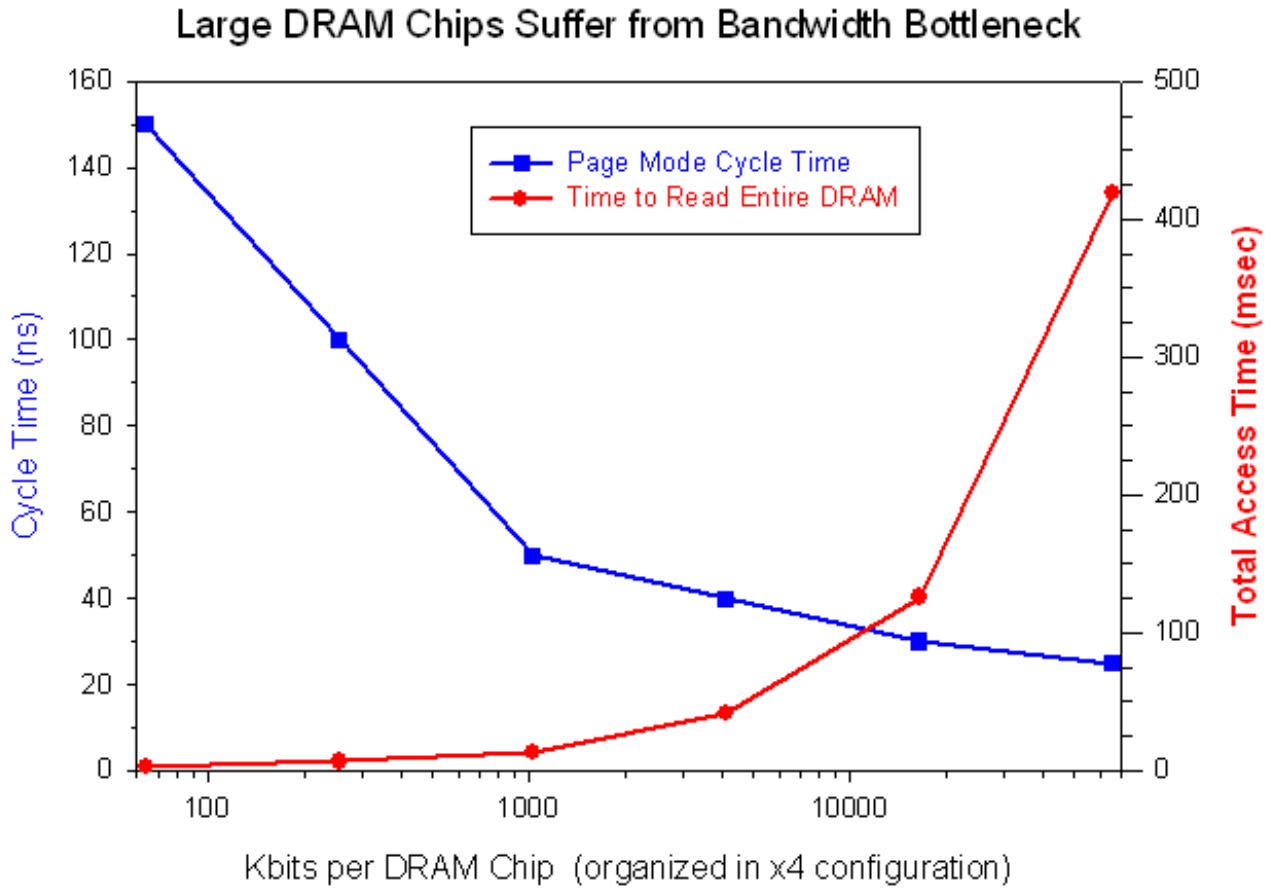
DRAM Trends

- Exponential growth in size (bits/chip)
- Only linear improvement in speed



DRAM Bandwidth Gap

- **Means that the time to read an entire DRAM is growing slower than the size (capacity) of DRAMS; takes longer and longer to read out a whole DRAM.**



DRAM: Organizational Changes

- **Modest organizational innovations in last 20 years**
  - Example: Synchronous DRAM, Rambus DRAM
  - Basically changes in how signaling occurs
  - Leverage the fact that an entire row gets read every RAS cycle

- **More recent focus on radical signaling changes**
  - Gbit signaling over essentially narrow (sometimes 1-bit serial), super-fast lines.
  - Example: RAMBUS: 10X bandwidth, +30% cost
  - What circuit designers worry about here (in order): (1) Cost/bit, (2) Capacity

Memory Technology Summary

- **RAM comes in many flavors**
  - 2 big ones: SRAM vs DRAM
  - SRAM: fast, expensive, larger silicon footprint.
  - DRAM: slow, cheap, smaller silicon footprint
  - Use SRAM in fast caches. Use DRAM in main memory

- **DRAM comes in many flavors**
  - DRAM optimized for capacity more than for speed
  - Secondarily, DRAM is optimized for different signaling modes to speed access
  - "Strange" signaling is common. Most common is need to save pins and send row-address and col-address on same wires. Leads to RAS CAS timing stuff.
  - Advanced DRAM chips with sophisticated signaling are increasingly common (eg, Synch DRAM, Rambus, etc.) Goal is to break speed bottleneck in DRAM.