

# HY220

## Εργαστήριο Ψηφιακών Κυκλωμάτων

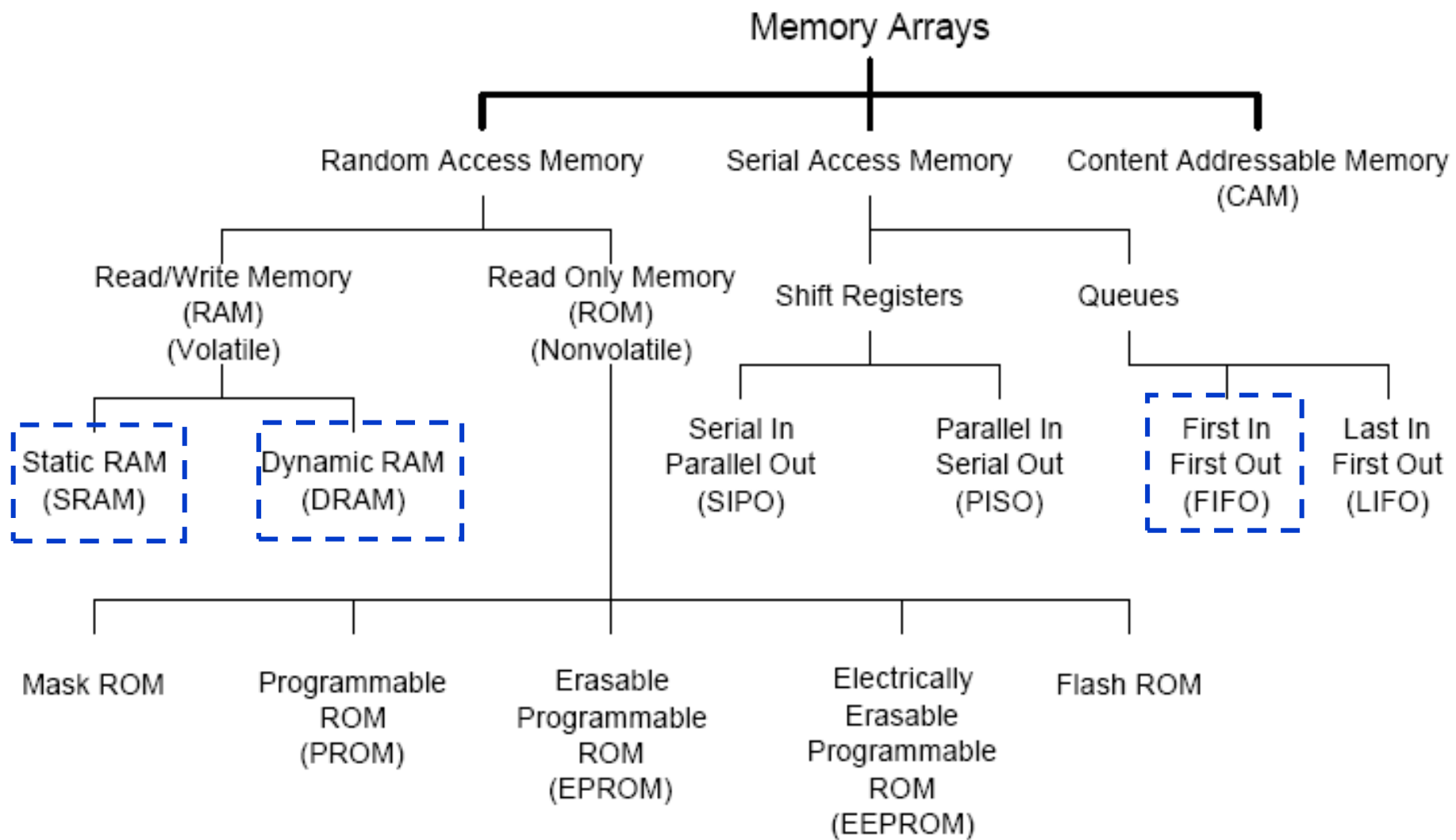
Εαρινό Εξάμηνο  
2024

Στατικές Μνήμες - SRAM

# Περίληψη

- Μνήμη είναι μια συλλογή από κελιά αποθήκευσης μαζί με κατάλληλα κυκλώματα για είσοδο και έξοδο από και προς την μνήμη.
  - Μπορούμε να γράφουμε και να διαβάζουμε κελιά
- Η μνήμη RAM (Random Access Memory) οργανώνει τα δεδομένα σε λέξεις
- Τα δεδομένα προσπελαυνονται μέσω μιας ακολουθίας από σήματα
  - Κυματομορφές χρονισμού (timing waveforms)
- Οι αποκωδικοποιητές είναι από τα πιο σημαντικά κομμάτια των μνημών
  - Επιλέγουν συγκεκριμένα δεδομένα
- Οι στατικές μνήμες χάνουν τα δεδομένα τους όταν τις αποσυνδέσουμε από το ρεύμα.

# Τύποι Memory Arrays



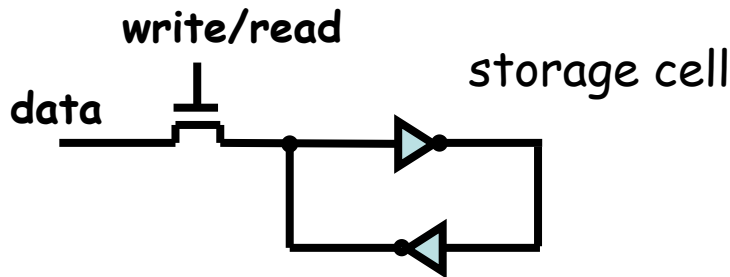
# Τύποι RAMs

- **Static Random Access Memory (SRAM)**
  - Operates like a collection of latches
  - Once value is written, it is guaranteed to remain in the memory as long as power is applied
  - Generally expensive
  - Used inside processors (like the Pentium)
- **Dynamic Random Access Memory (DRAM)**
  - Generally, simpler internal design than SRAM
  - Requires data to be rewritten (refreshed), otherwise data is lost
  - Often hold larger amount of data than SRAM
  - Longer access times than SRAM
  - Used as main memory in computer systems

# SRAM vs. DRAM

## Static RAM (SRAM)

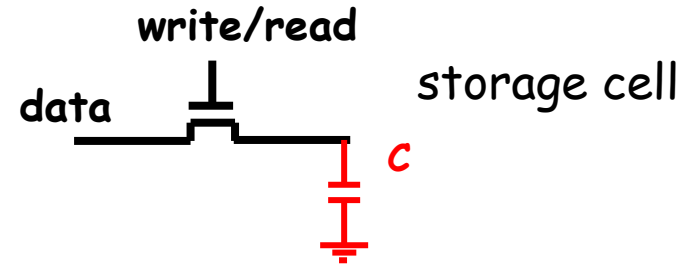
Δεδομένα αποθηκεύονται σε Latch.



- +Ταχύτερη προσπέλαση μνήμης.
- +Δεν χρειάζεται refreshing.
- +Καλή συμπεριφορά στον θόρυβο.
- Μεγαλύτερο μέγεθος/bit από DRAM.

## Dynamic RAM (DRAM)

Δεδομένα αποθηκεύονται σε φορτίο dynamic node.



- +Μικρό μέγεθος/bit μνήμης.
- Χρειάζεται refreshing λόγω leakage. Πιο αργή από SRAM.
- Προβλήματα με θόρυβο (noise).

# Τα βασικά σήματα των RAMs

- Γραμμές εισόδου και εξόδου δεδομένων (input and output lines, DIN - DOUT)
- Η μνήμη περιέχει  $2^k$  λέξεις (memory words)
  - $K$  γραμμές διεύθυνσης (address lines - ADDR) επιλέγουν 1 λέξη από τις  $2^k$
- Θέτουμε το σήμα Read (asserted) για να μεταφέρουμε δεδομένα στην έξοδο. (OE - output enable)
- Θέτουμε το σήμα Write (asserted) για να αποθηκεύσουμε τα δεδομένα της εισόδου. (WE - write enable)
- Chip Enable/ Chip Select (CE-CS) σαν γενικός διακόπτης λειτουργίας

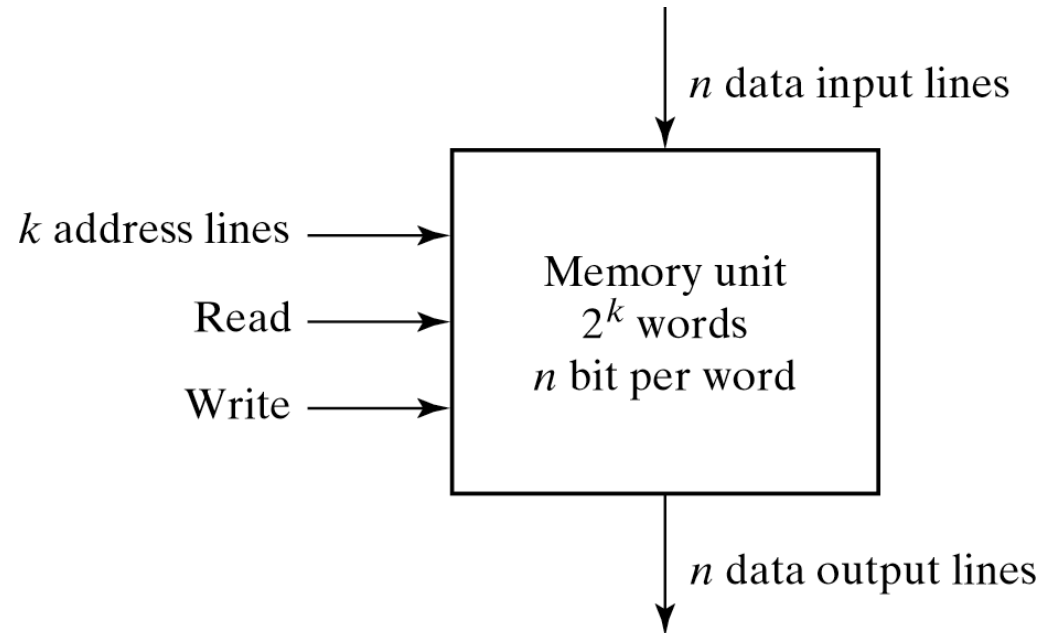
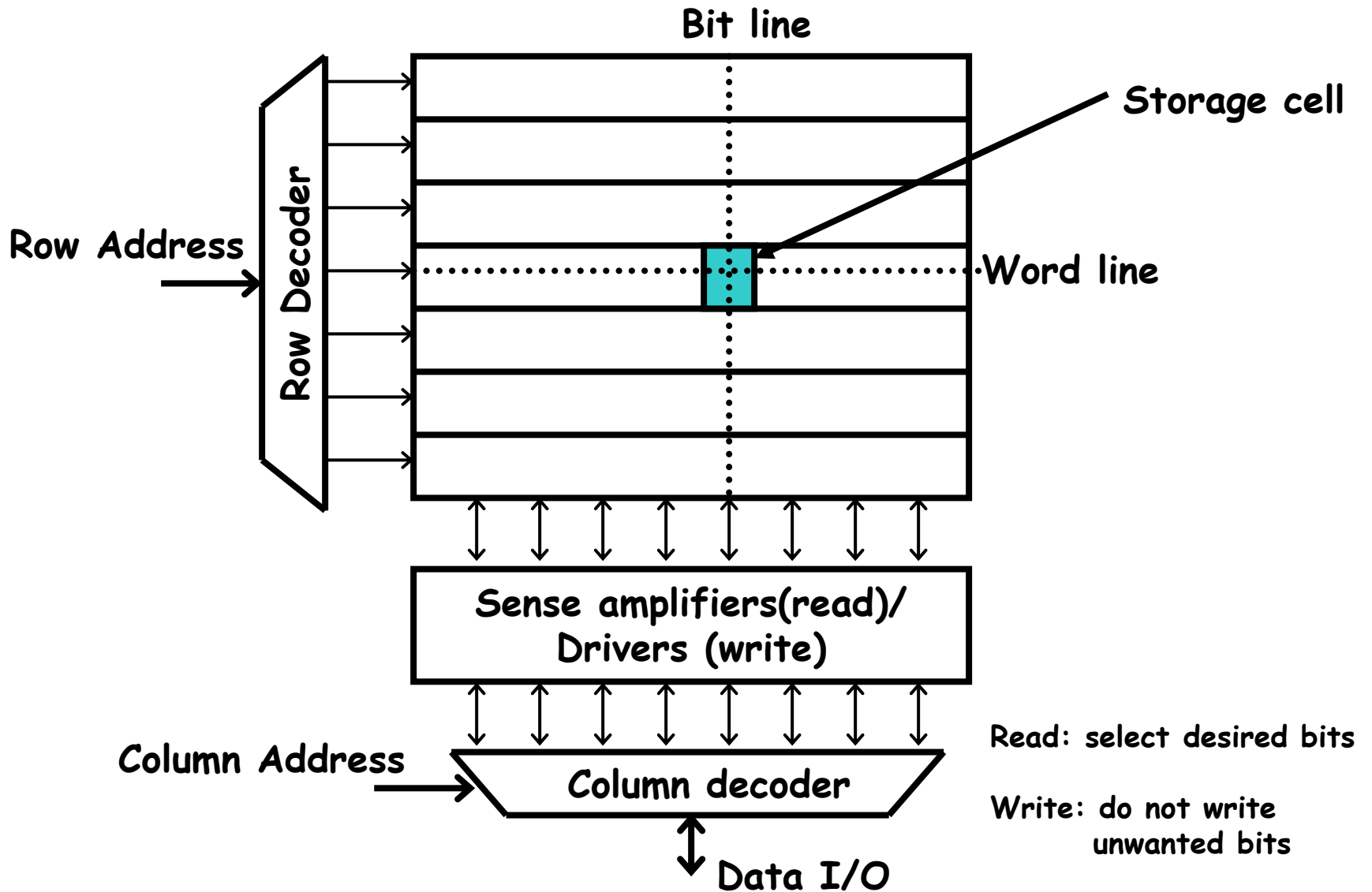


Fig. 7-2 Block Diagram of a Memory Unit

# Αρχιτεκτονική Μνημών



# Τα εσωτερικά της RAM

- Η διεύθυνση πάει στον decoder
  - Μόνο μια έξοδος ενεργή
- Η Word line επιλέγει μια γραμμή (row) από bits (word)
- Κάθε binary cell (BC) αποθηκεύει 1 bit
- Τα δεδομένα εισόδου αποθηκεύονται όταν το Read/Write είναι 0
- Τα δεδομένα εξόδου βγαίνουν όταν το Read/Write είναι 1

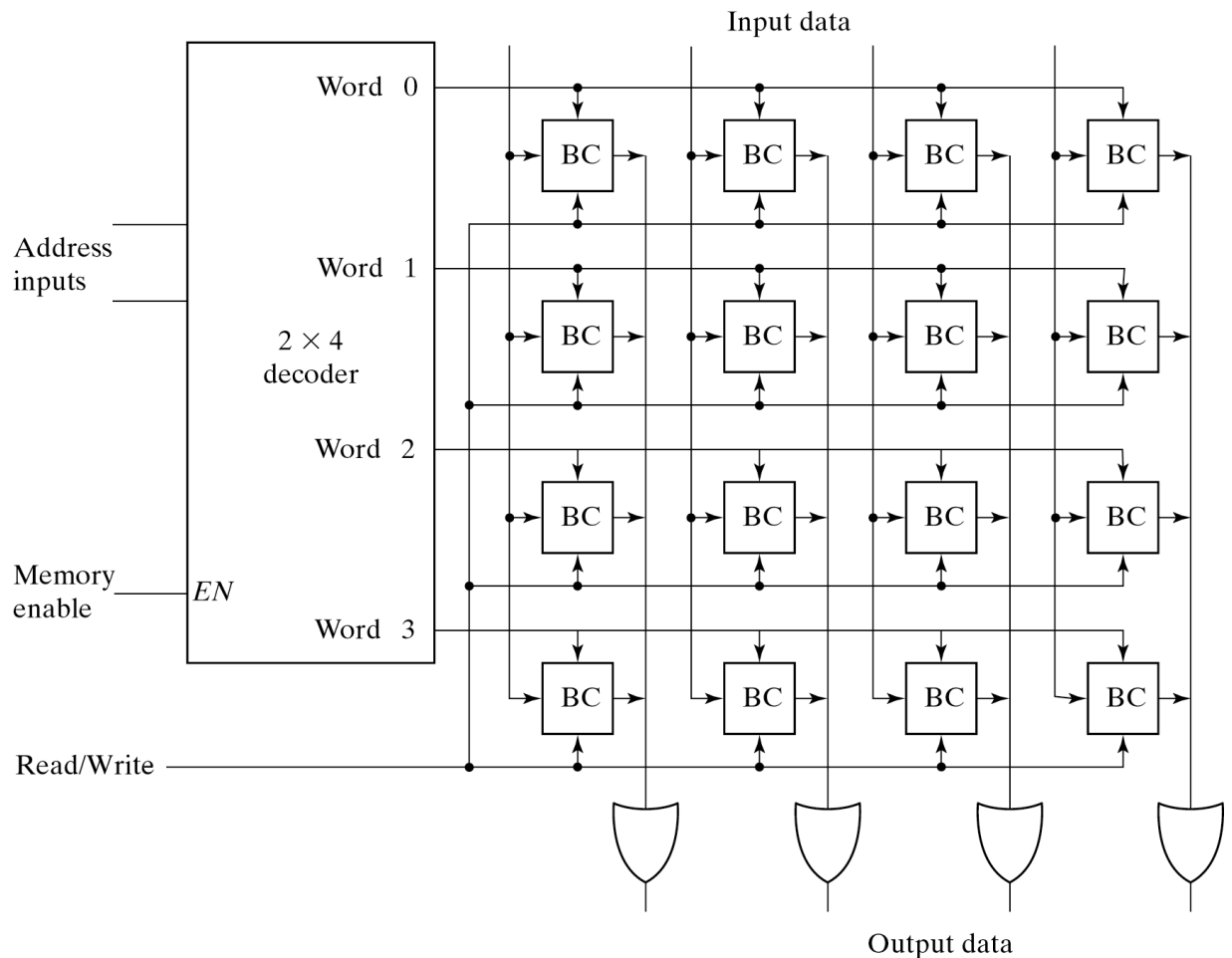


Fig. 7-6 Diagram of a 4 × 4 RAM



# Τα εσωτερικά της SRAM

- Note: η καθυστέρηση εξαρτάται κυρίως από τον αριθμό των λέξεων
- Η καθυστέρηση δεν επηρεάζεται από το μήκος των λέξεων
- Πόσα address bits χρειαζόμαστε για 16 words?

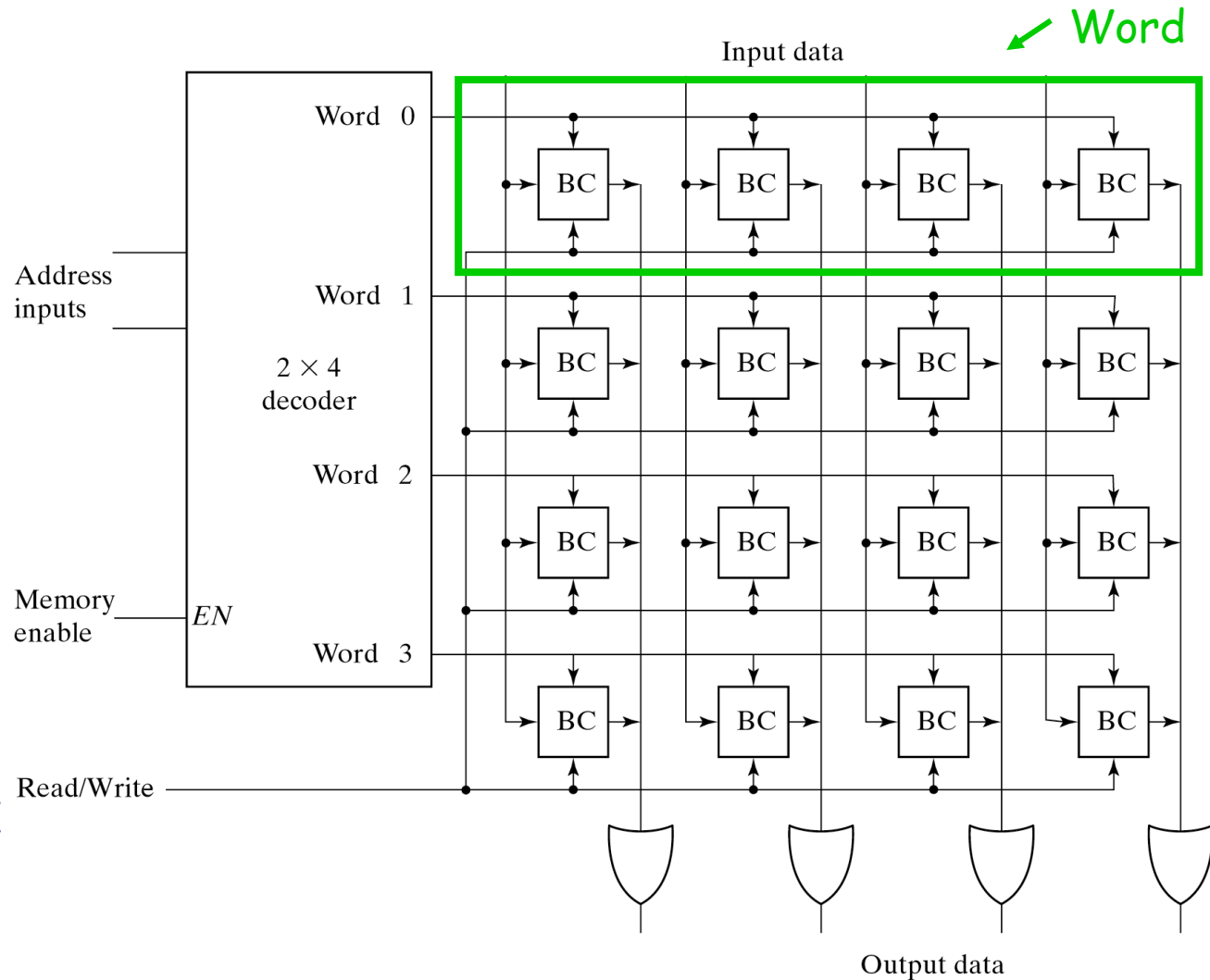
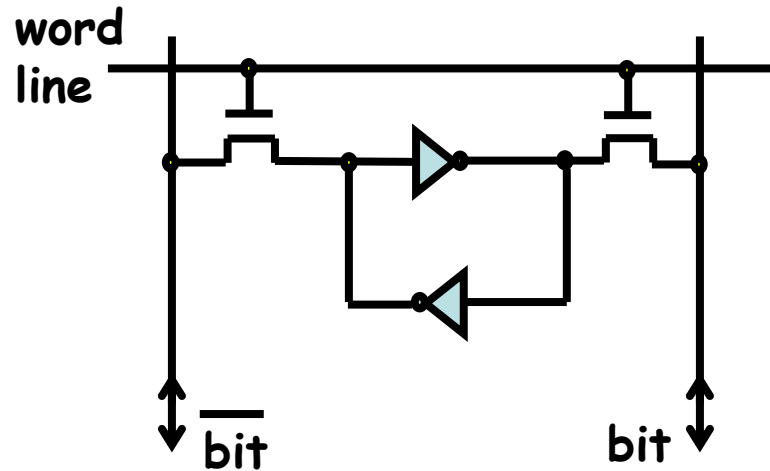


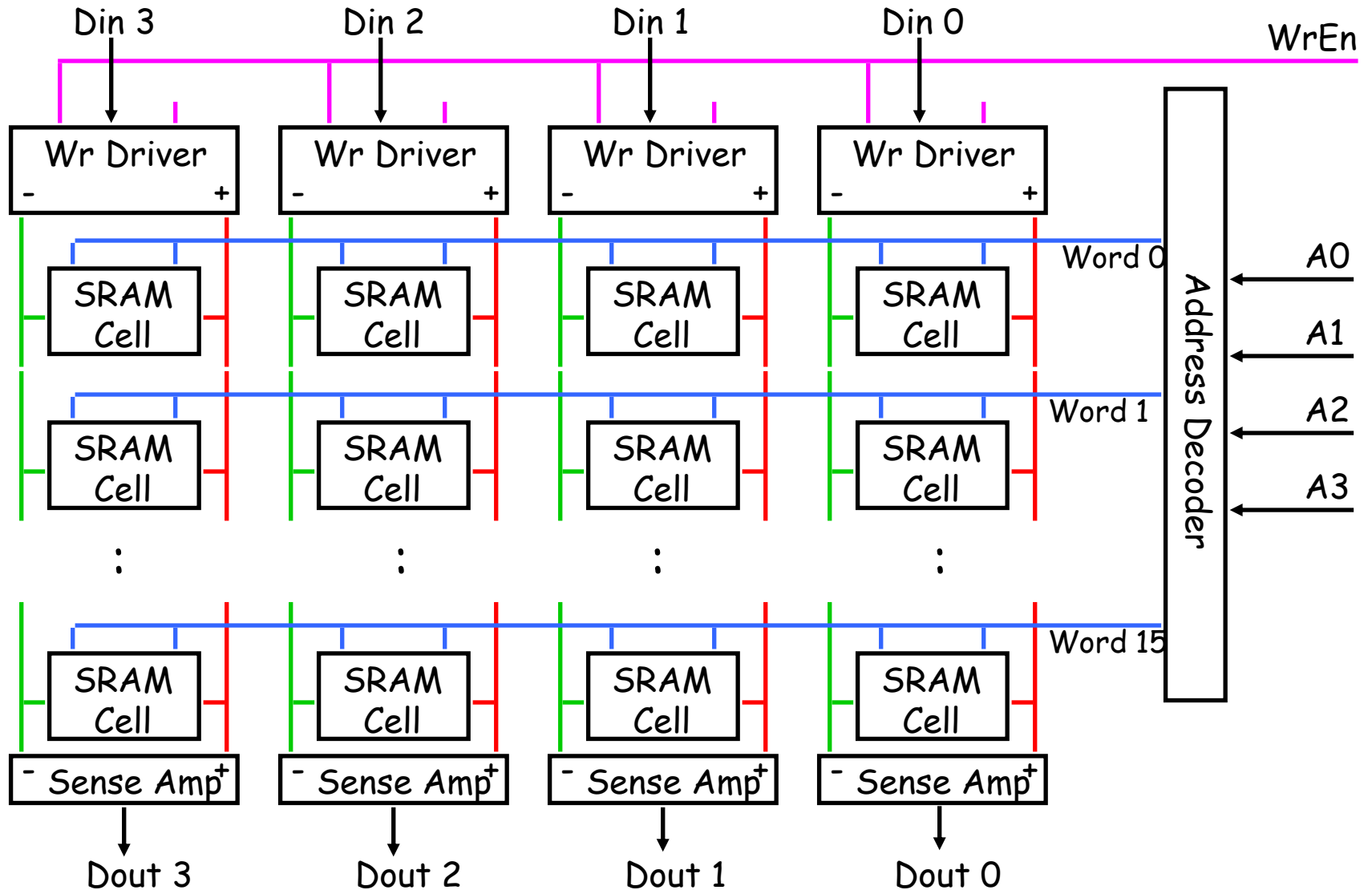
Fig. 7-6 Diagram of a 4 × 4 RAM

# Ένα κελί SRAM - 6 transistors

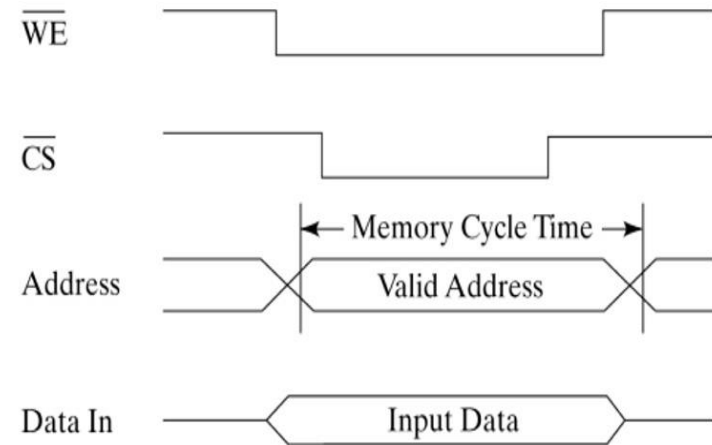
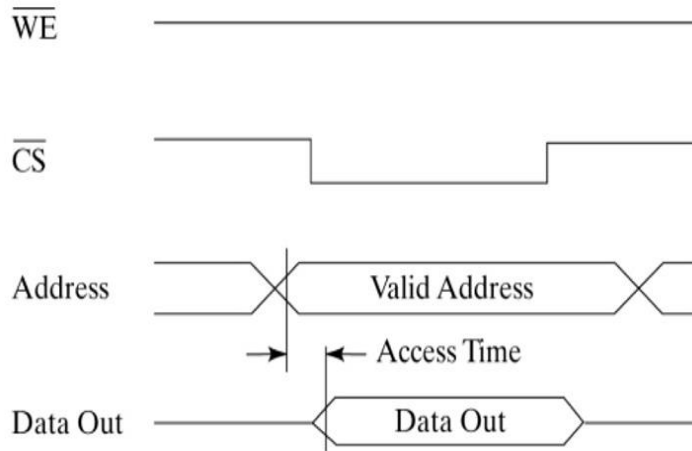


Όταν η word line ενεργοποιείται ( $V_{DD}$ ) τότε η τιμή του Latch διαβάζεται στα bit και  $\overline{\text{bit}}$  κατά το διάβασμα της μνήμης ή η τιμή του Latch γράφεται από τα bit και  $\overline{\text{bit}}$  κατά την εγγραφή της μνήμης.

# Μια Τυπική οργάνωση SRAM : 16-word x 4-bit



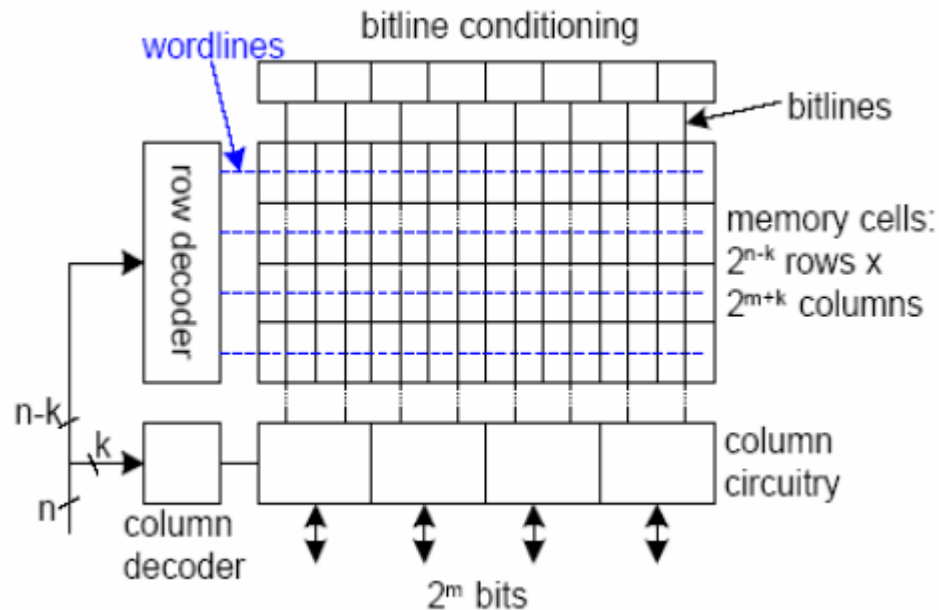
# Απλοποιημένο διάγραμμα χρονισμού SRAM



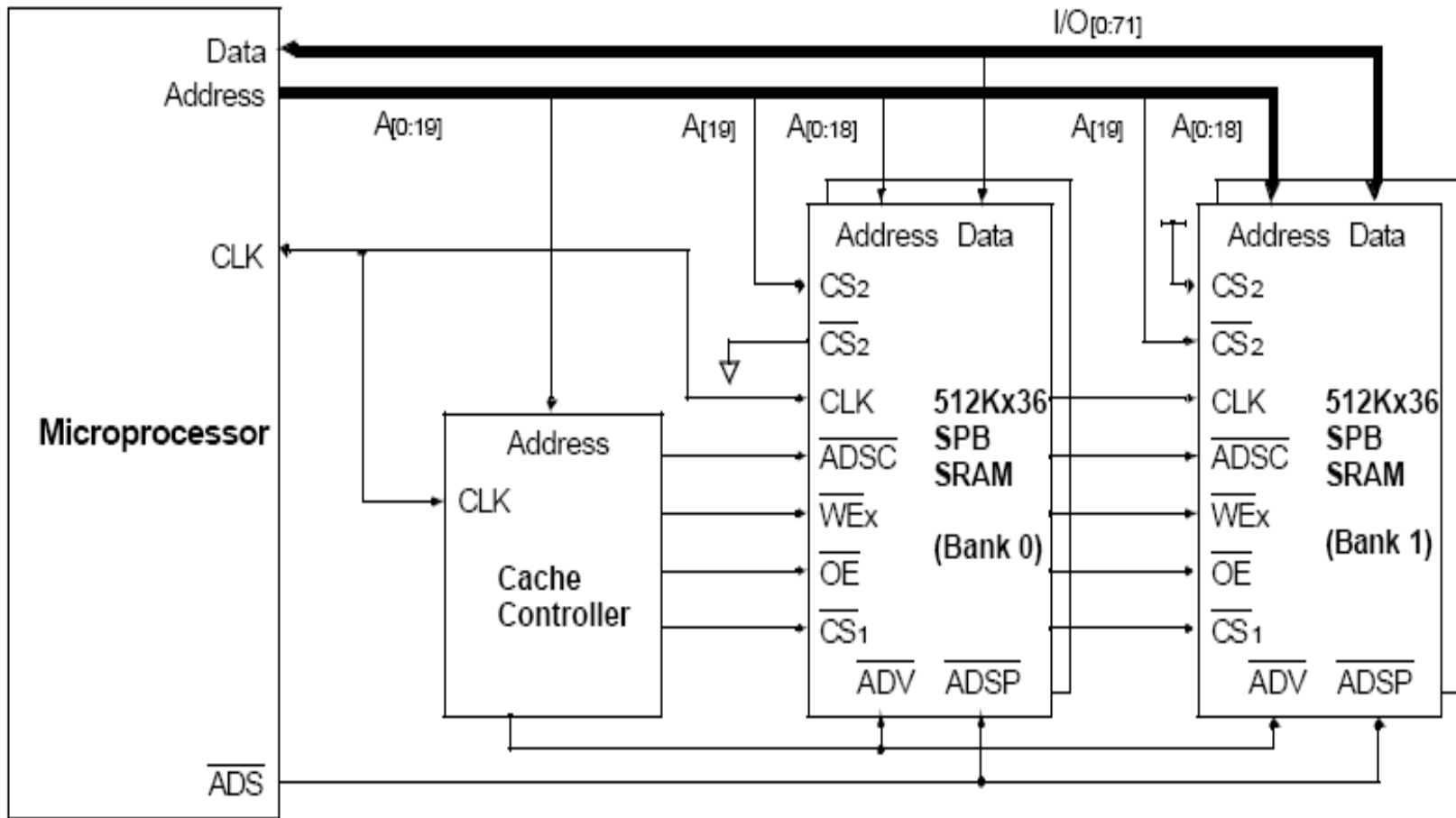
- Read: Έγκυρη address, και μετά Chip Select ( $CS$ )
- Access Time: Ο χρόνος για την έξοδο των δεδομένων μετά από έγκυρη address
- Cycle Time: Ελάχιστος χρόνος μεταξύ συνεχόμενων λειτουργιών της μνήμης
- Write: Έγκυρη address και data μαζί με  $WE$ , μετά  $CS$ 
  - Η Address πρέπει να είναι έγκυρη setup time πριν το  $WE$  και το  $CS$  ενεργοποιηθούν.
  - Και hold time αφού απενεργοποιηθούν

# Αρχιτεκτονική για Μεγαλύτερες Μνήμες

- $2^n$  λέξεις από  $2^m$  bits η καθεμιά
- Εάν  $n \gg m$  μπορούμε να την διπλώσουμε (fold) κατά  $2^k$  σε λιγότερες γραμμές και πιο πολλές στήλες.
  - Αποκωδικοποιητής στηλών !
- Κανονικότητα στη σχεδίαση - εύκολη σχεδίαση

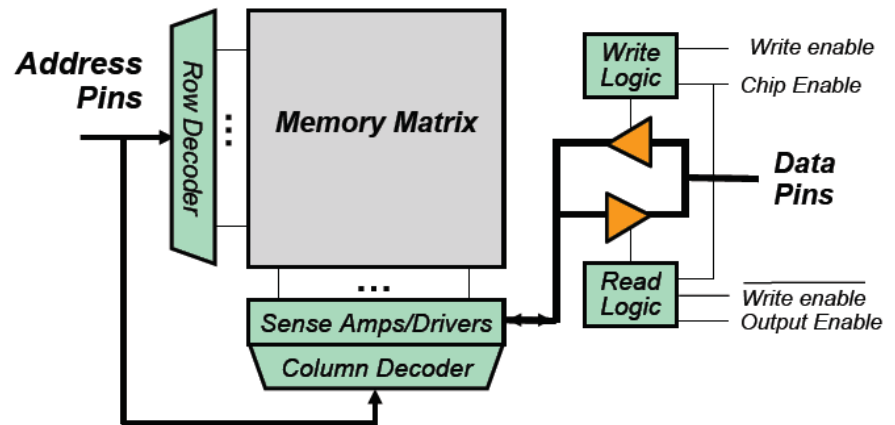


# SRAM Banks



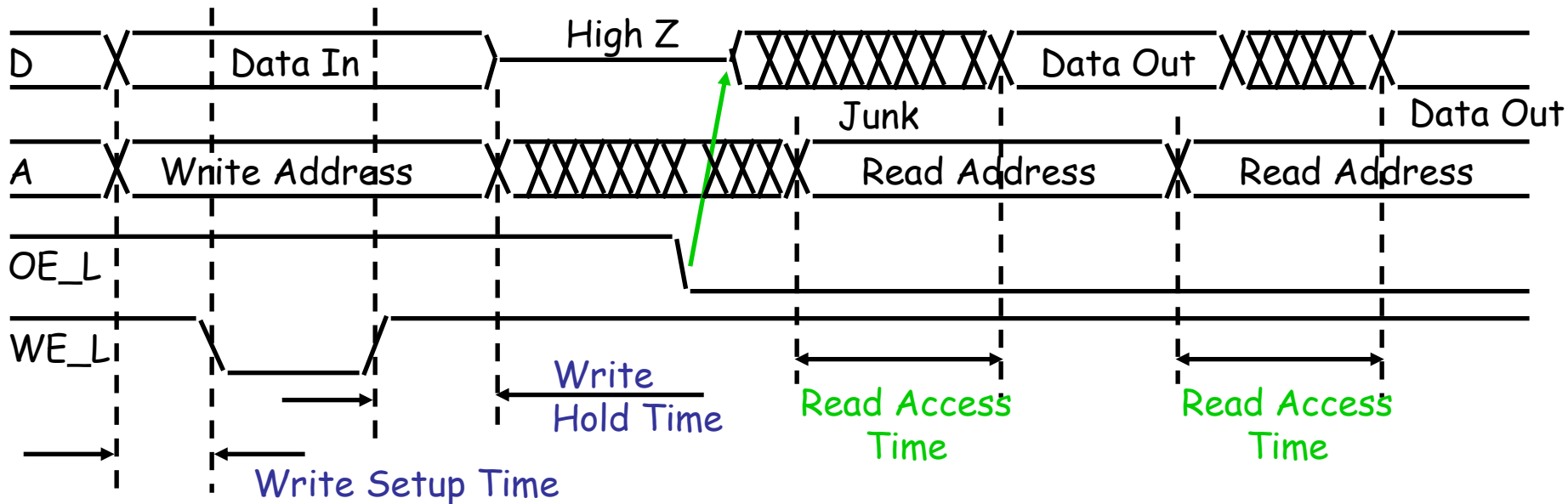
# Τυπικός χρονισμός Ασύγχρονης SRAM

- Asynchronous SRAM
  - Χρονισμός μόνο βάση των σημάτων



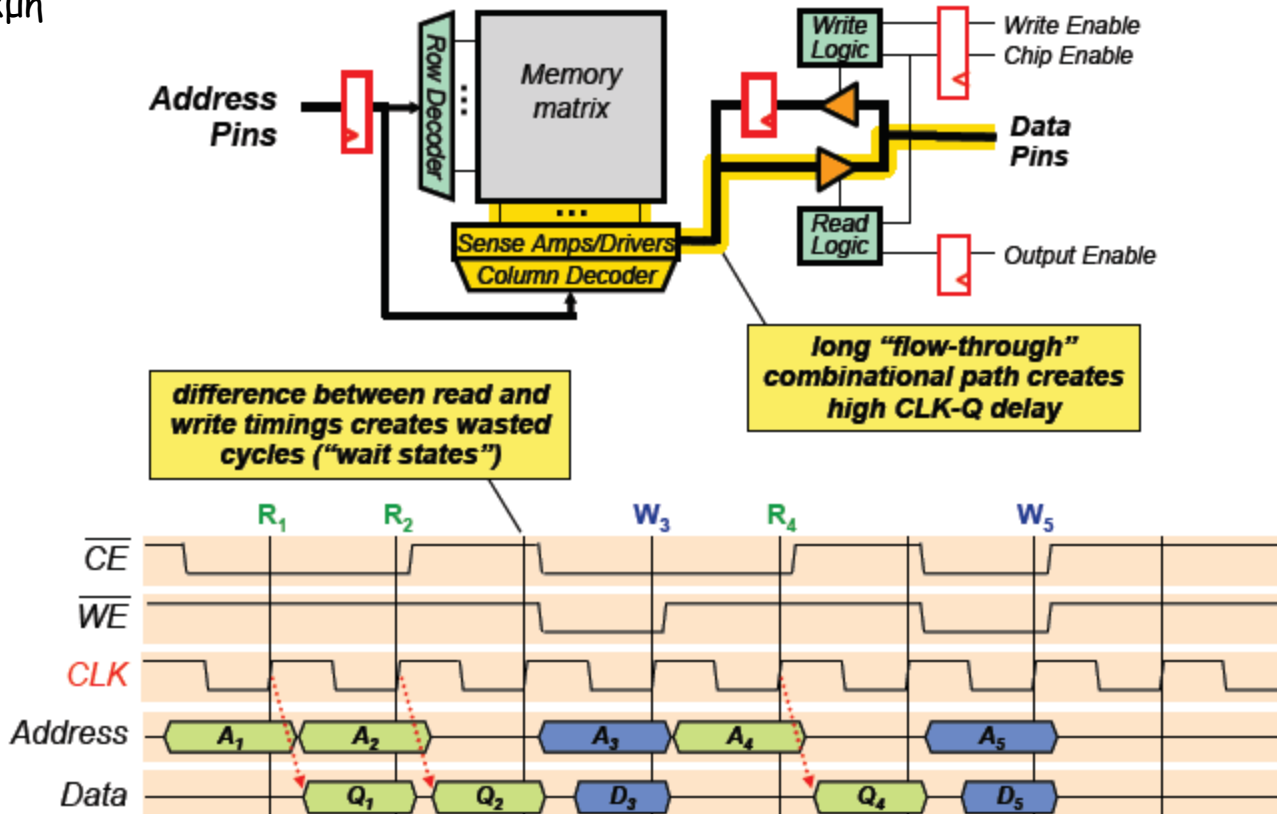
Write Timing:

Read Timing:



# Τυπικός χρονισμός Σύγχρονης SRAM

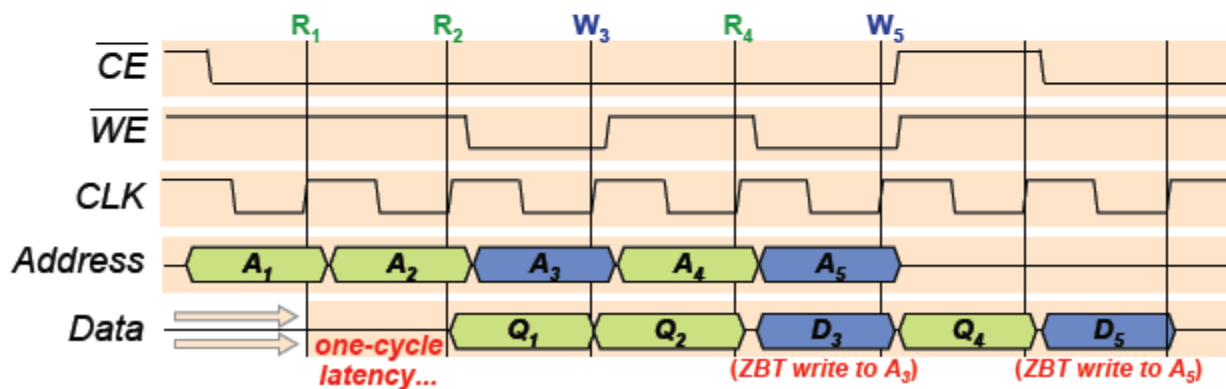
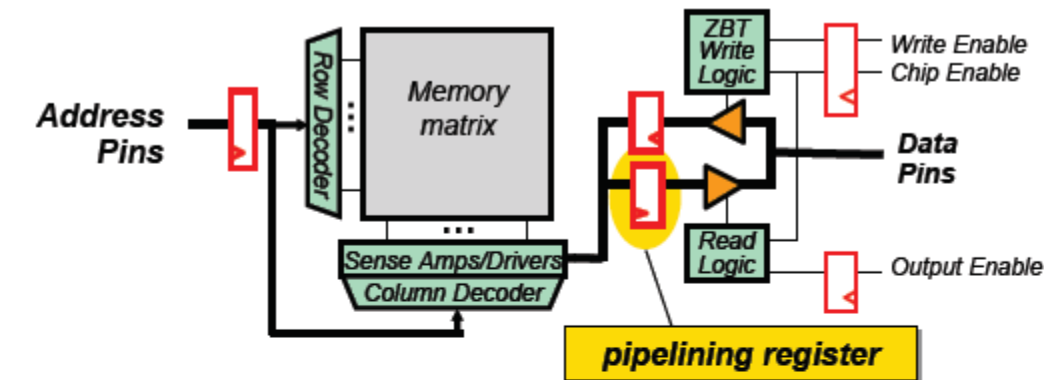
- Synchronous - SRAM
  - Χρονισμός με βάση τα σήματα στην ακμή του ρολογιού
- Write-after-Read πρόβλημα - Wait states - Bus Turnaround
  - Στα read τα data βγαίνουν μετά την ακμή ενώ στα writes τα data πρέπει να μπουν πριν την ακμή



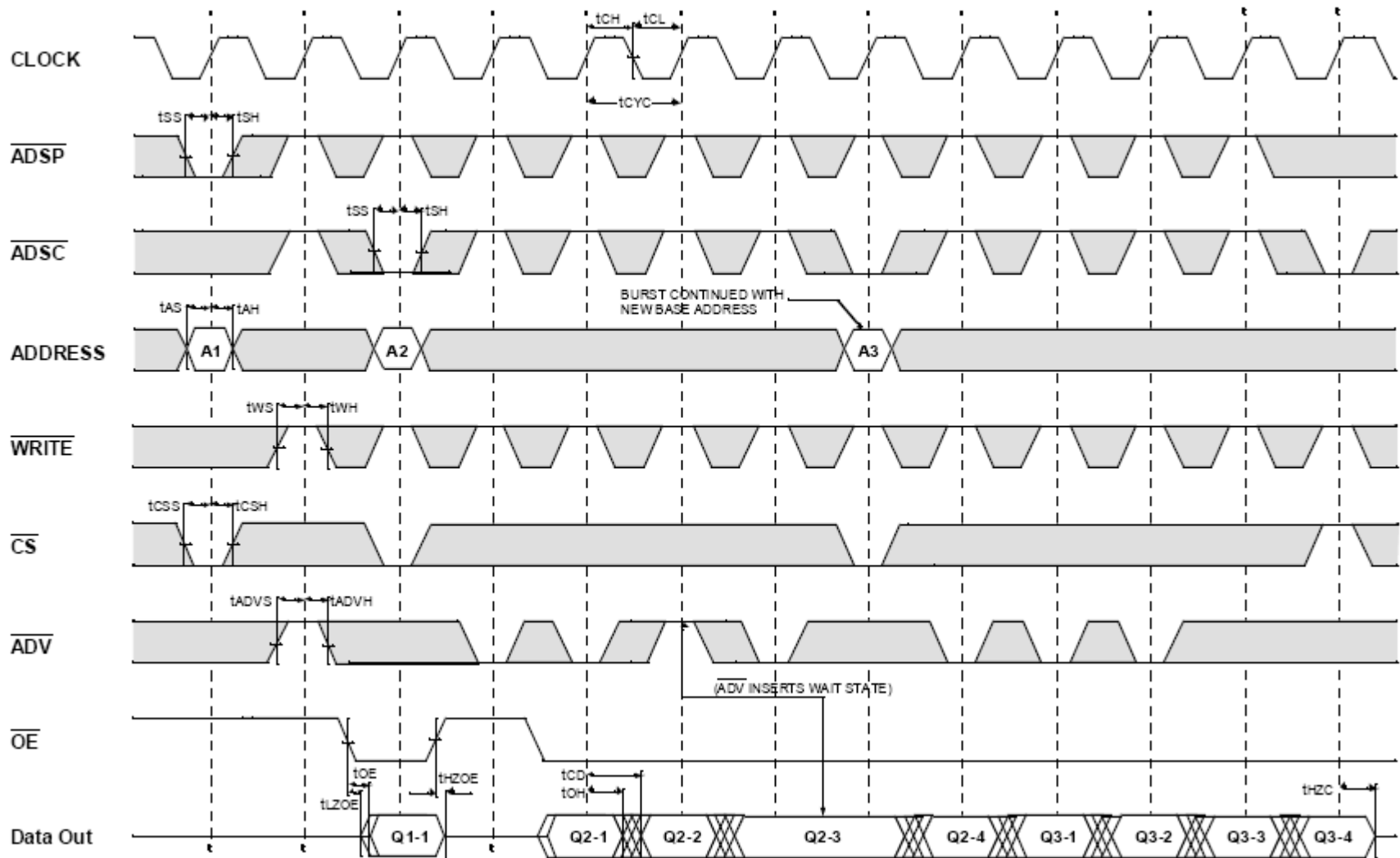


# Τυπικός χρονισμός ZBT Synchronous SRAM

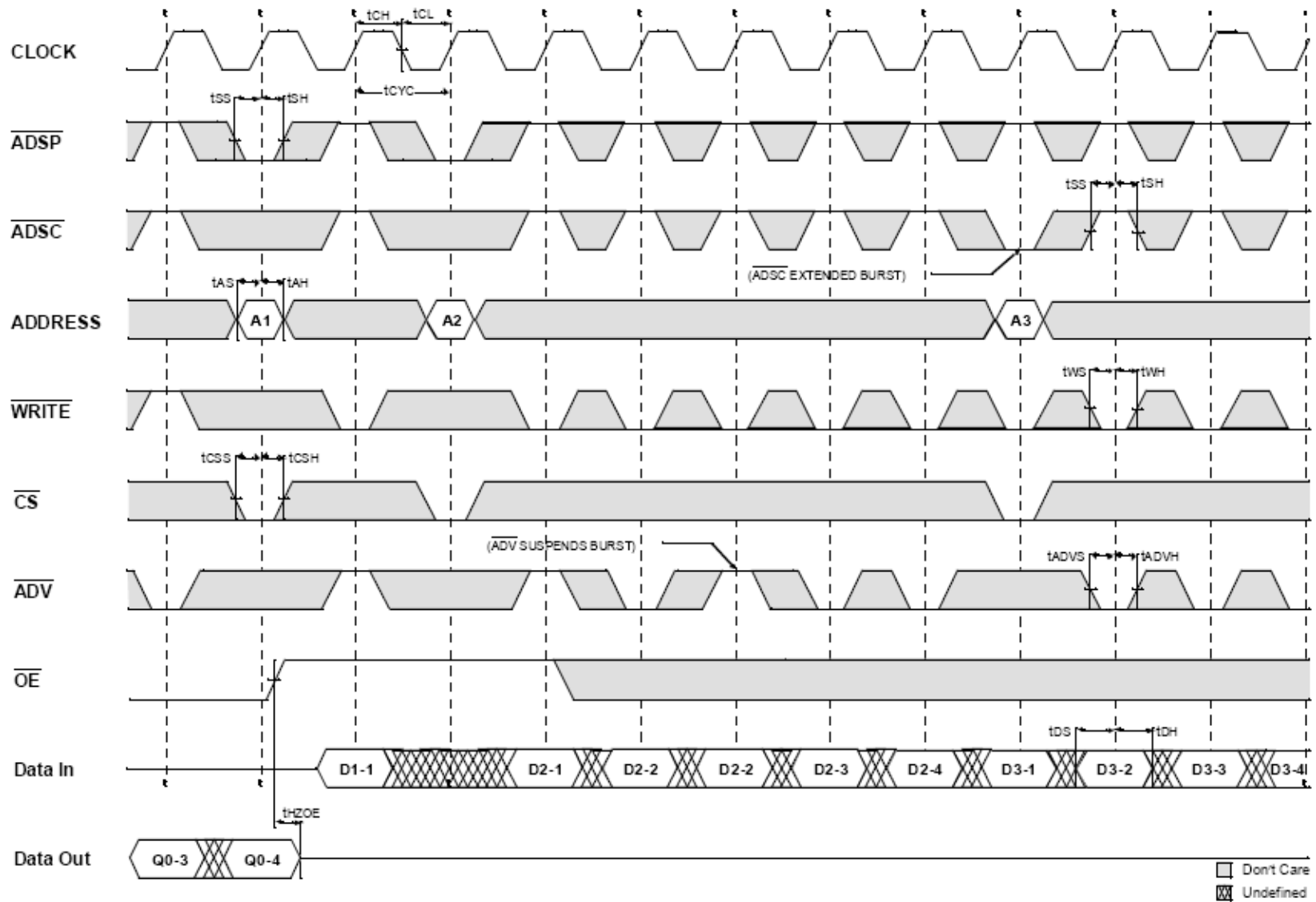
- Zero-Bus-Turnaround (ZBT) ή No Bus Latency (NoBL)
  - Pipelined έξοδος λύνει το πρόβλημα Write-after-Read
  - Μειώνει  $Clk2Q$  delay και επιτρέπει υψηλότερη συχνότητα ρολογιού
  - Έκτρα κύκλος καθυστέρηση στην ανάγνωση



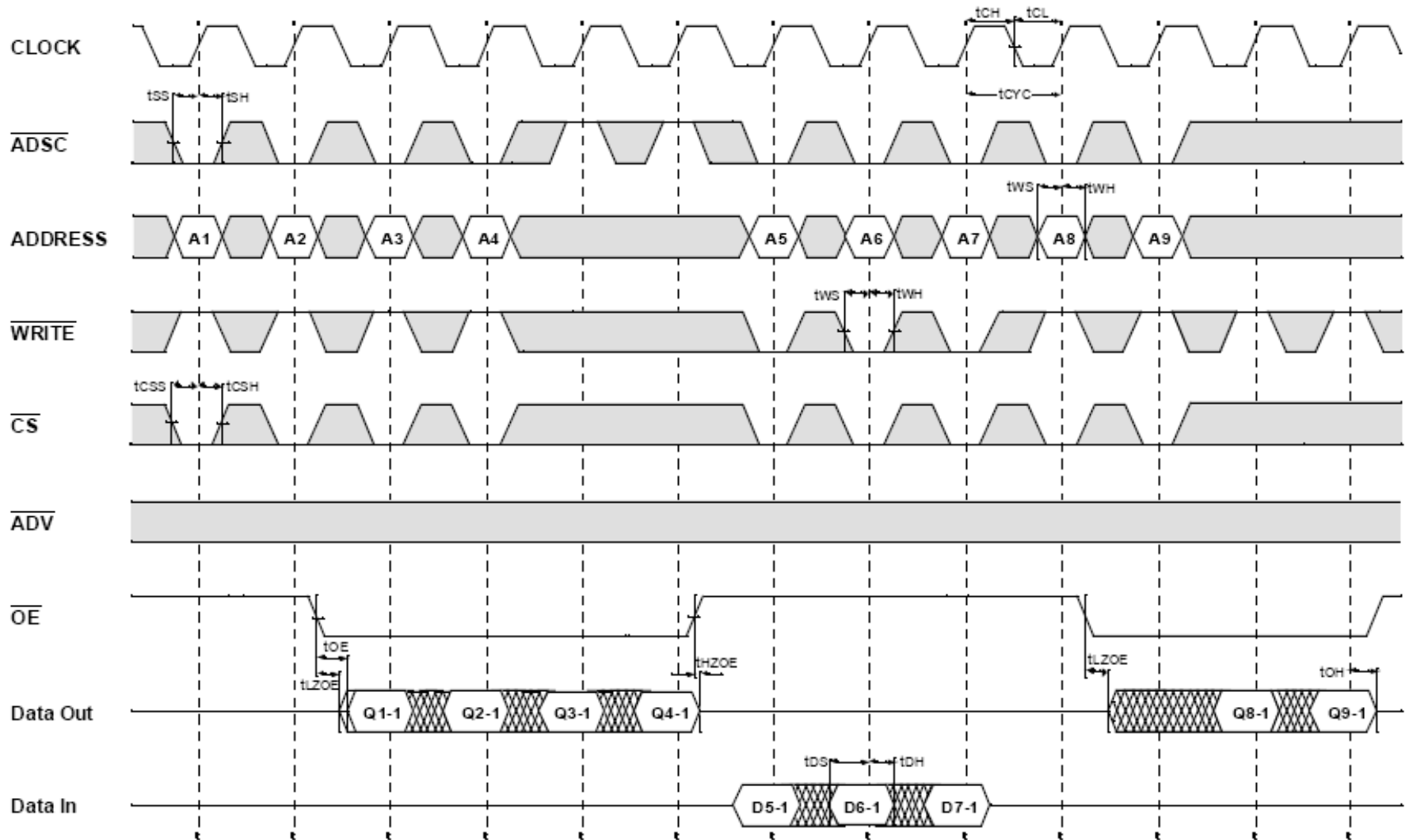
# Χρονισμός SRAM για Read από Datasheet



# Χρονισμός SRAM για Write από Datasheet



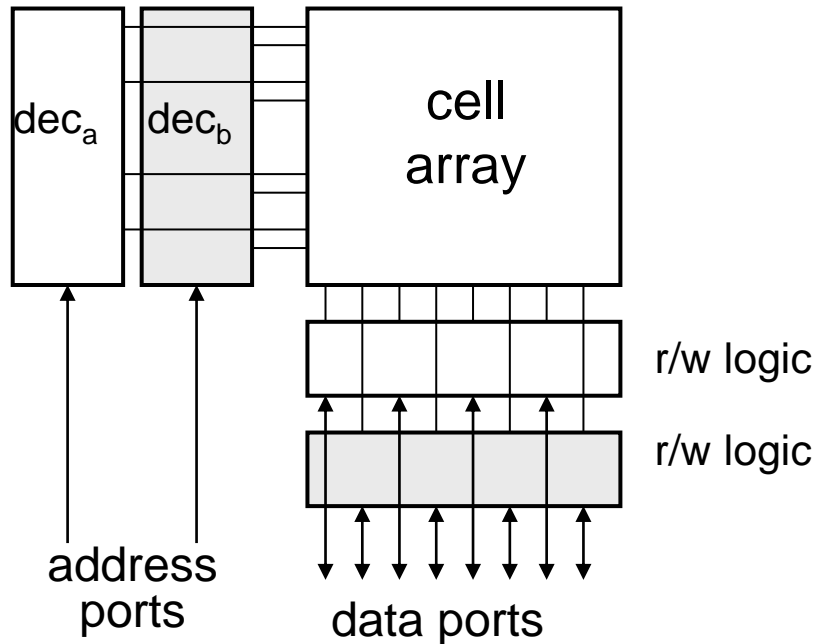
# Χρονισμός SRAM για Read/Write από Datasheet



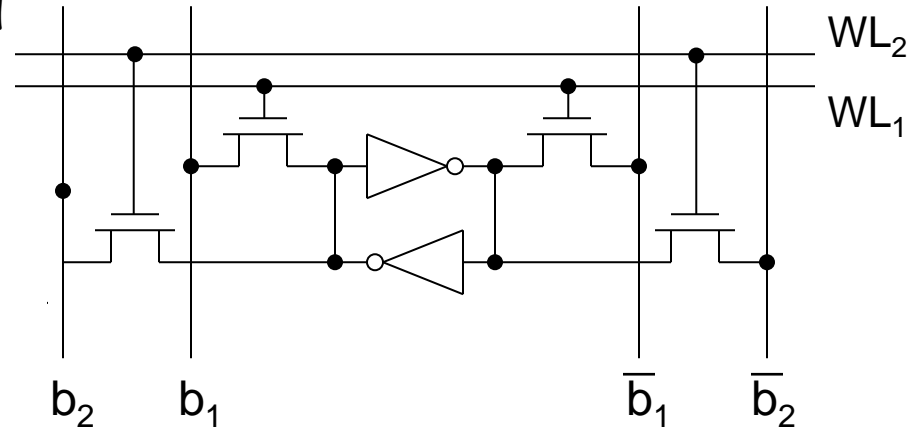
□ Don't Care  
 ▨ Undefined

# Dual-ported Memory Internals

- Add decoder, another set of read/write logic, bits lines, word lines:



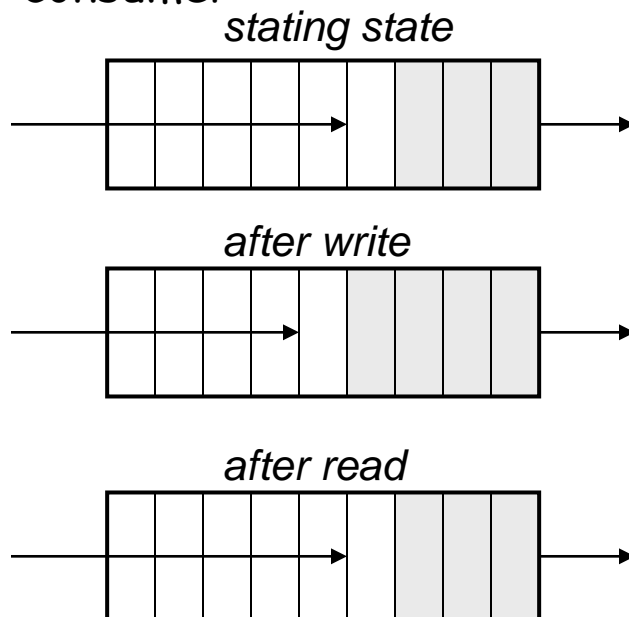
- Example cell: SRAM



- Repeat everything but cross-coupled inverters.
- This scheme extends up to a couple more ports, then need to add additional transistors.

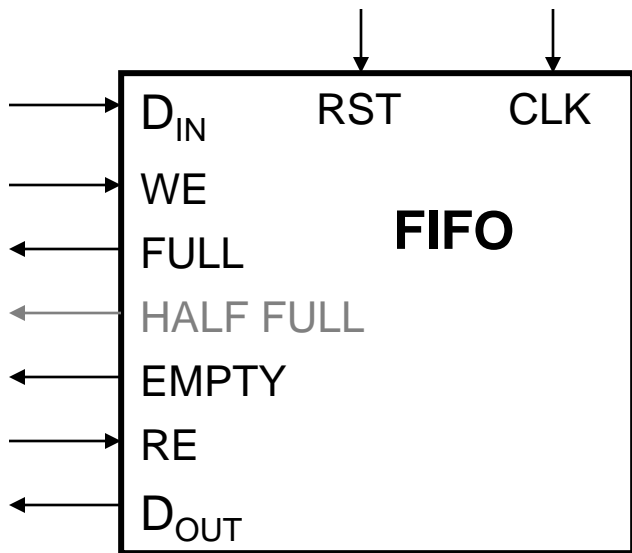
# First-in-first-out (FIFO) Memory

- Used to implement *queues*.
- These find common use in computers and communication circuits.
- Generally, used for rate matching data producer and consumer:

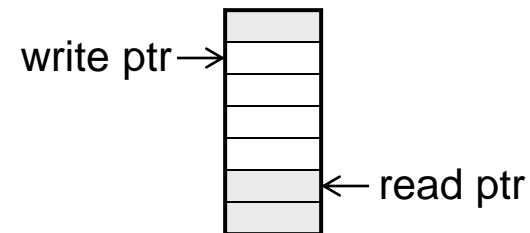


- Producer can perform many writes without consumer performing any reads (or vice versa). However, because of finite buffer size, on average, need equal number of reads and writes.
- Typical uses:
  - interfacing I/O devices. Example network interface. Data bursts from network, then processor bursts to memory buffer (or reads one word at a time from interface). Operations not synchronized.
  - Example: Audio output. Processor produces output samples in bursts (during process swap-in time). Audio DAC clocks it out at constant sample rate.

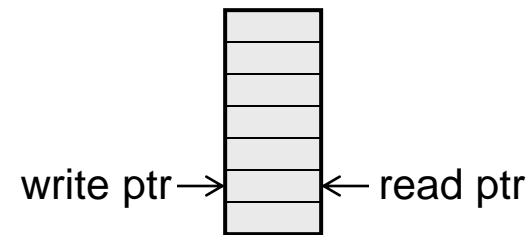
# FIFO Interfaces



- Address pointers are used internally to keep next write position and next read position into a dual-port memory.



- If pointers equal after write  $\Rightarrow$  FULL:



- After write or read operation, FULL and EMPTY indicate status of buffer.
- Used by external logic to control own reading from or writing to the buffer.
- FIFO resets to EMPTY state.
- HALF FULL (or other indicator of partial fullness) is optional.

- If pointers equal after read  $\Rightarrow$  EMPTY:

