ELECTRICAL ENGINEERING
UNIVERSITY *of* WASHINGTON

Lecture 13: Memory Structures

Based on material prepared by prof. Visvesh S. Sathe

ELECTRICAL & COMPUTER ENGINEERING

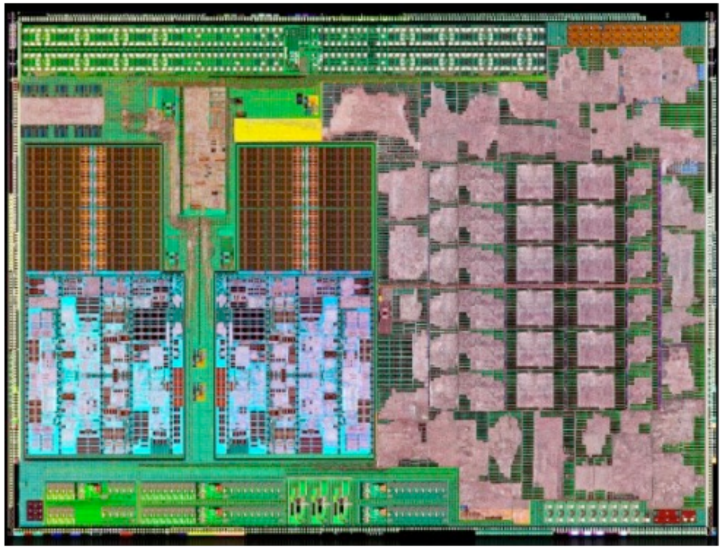
Acknowledgements

All class materials (lectures, assignments, etc.) based on material prepared by Prof. Visvesh S. Sathe, and reproduced with his permission

Visvesh S. Sathe
Associate Professor
Georgia Institute of Technology
https://psylab.ece.gatech.edu

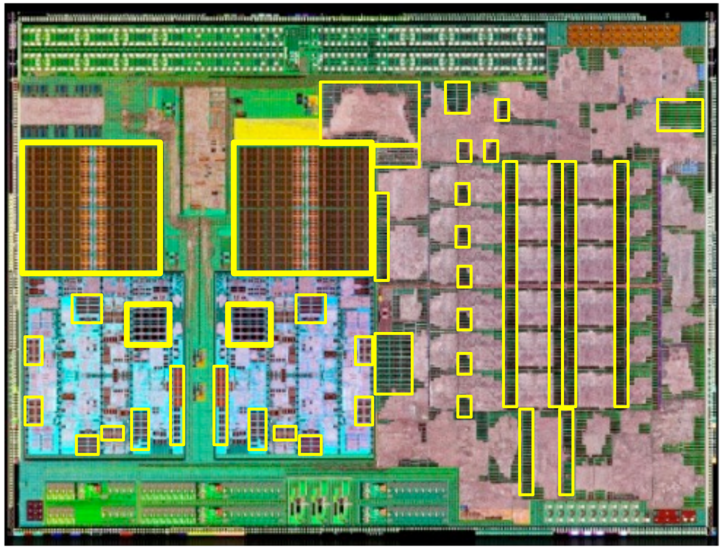
UW (2013-2022)
GaTech (2022-present)

Memory Structures



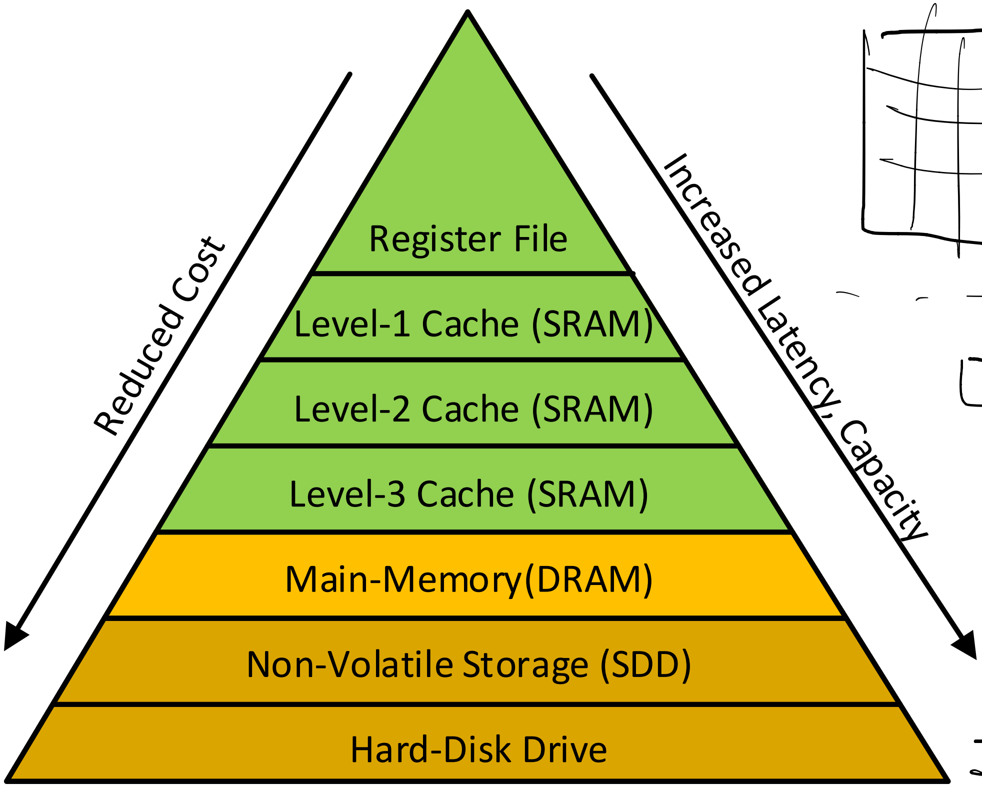
- Dominant source of Tx count, area in many modern CMOS systems
- Increasingly key to system performance
  - On-chip: Register files, Cache memory
  - Off-chip : Main memory storage (DDR), Solid-state drives
- Key properties
  - Density (bits per mm^2)
  - Latency (e.g. cycles to access)
  - Bandwidth

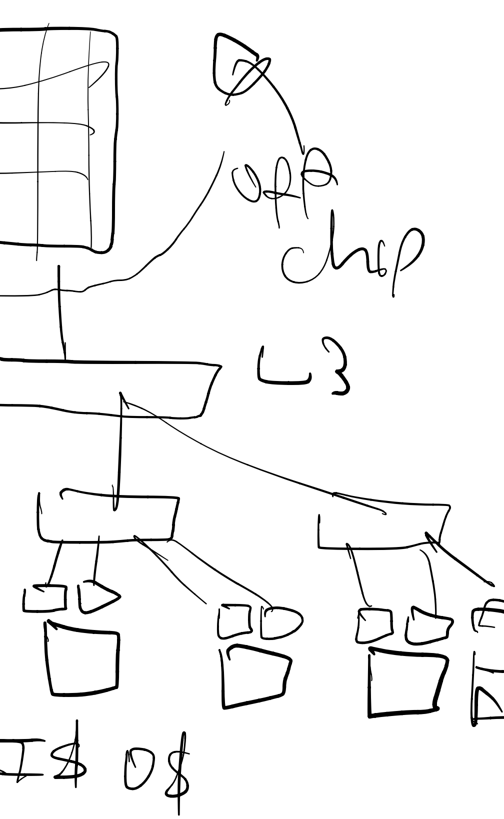
Memory Structures



- Dominant source of Tx count, area in many modern CMOS systems
- Increasingly key to system performance
  - On-chip: Register files, Cache memory
  - Off-chip : Main memory storage (DDR), Solid-state drives
- Key properties
  - Density (bits per mm^2)
  - Latency (e.g. cycles to access)
  - Bandwidth

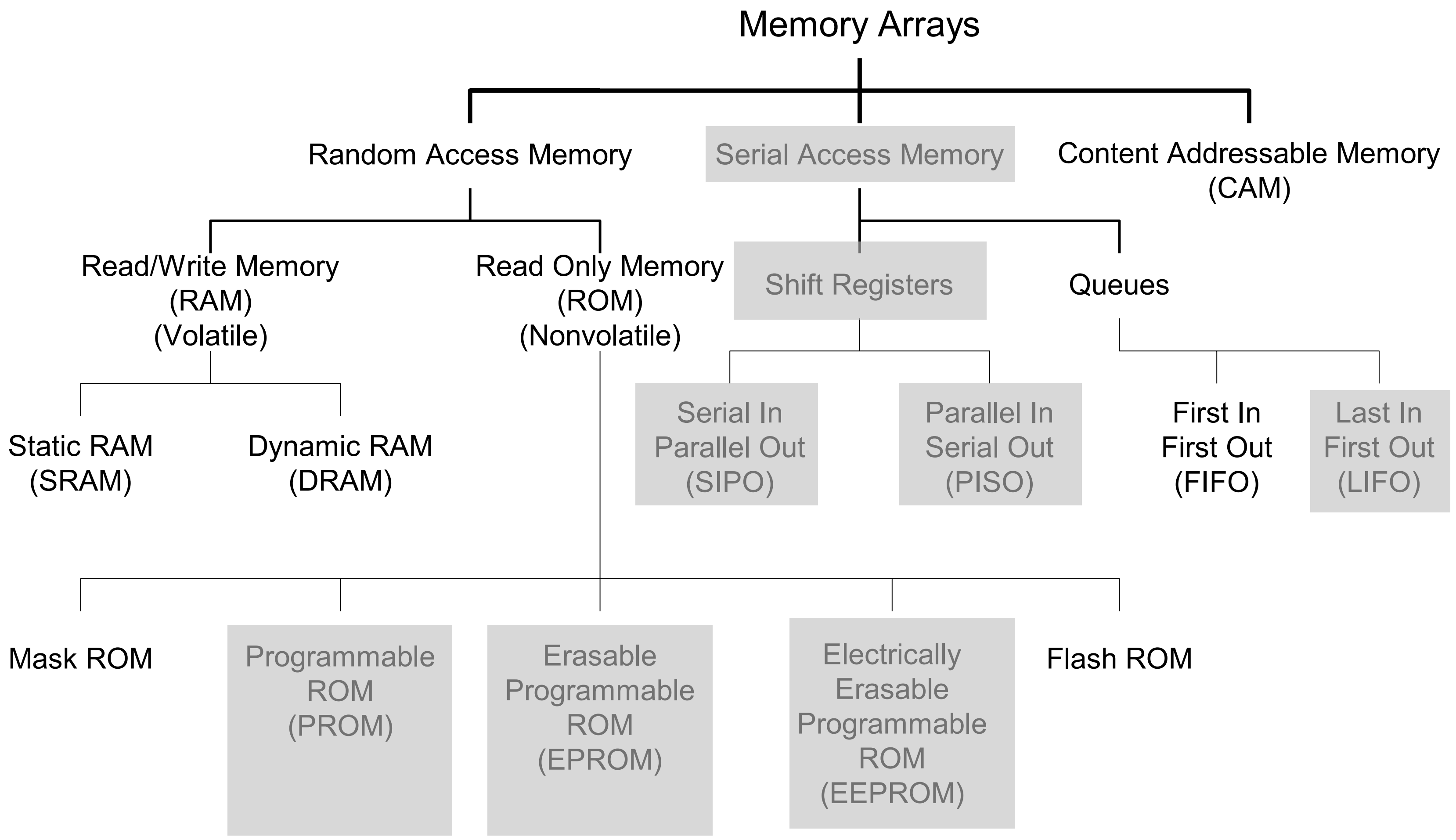
Memory Hierarchy





- Memory systems typically made up of *hierarchy* of different memory structures (cost, latency, bandwidth, capacity)
- Performance impact mitigated by
  - Locality of reference
  - Arithmetic intensity

We Will Cover…



Memory Arrays

Random Access Memory

Serial Access Memory

Content Addressable Memory (CAM)

Read/Write Memory (RAM) (Volatile)

Read Only Memory (ROM) (Nonvolatile)

Shift Registers

Queues

Static RAM (SRAM)

Dynamic RAM (DRAM)

Serial In Parallel Out (SIPO)

Parallel In Serial Out (PISO)

First In First Out (FIFO)

Last In First Out (LIFO)

Mask ROM

Programmable ROM (PROM)

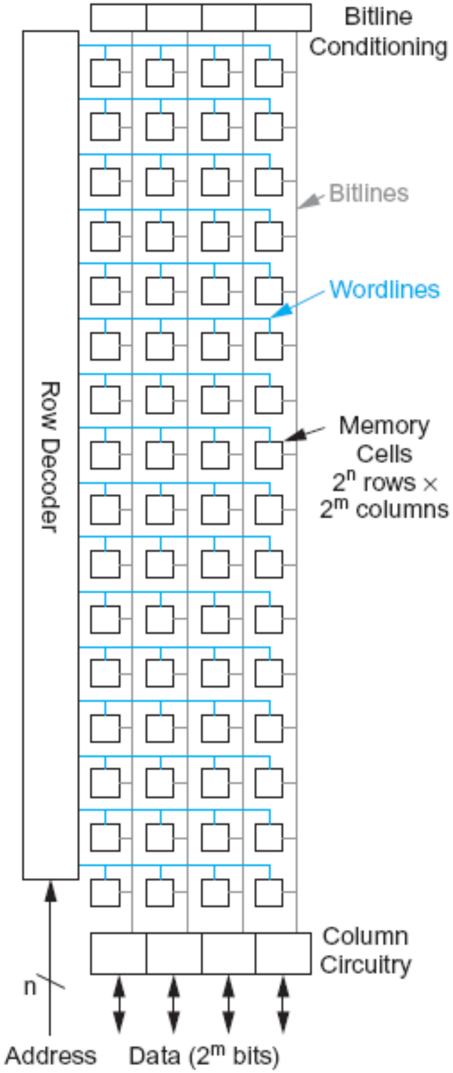
Erasable Programmable ROM (EPROM)

Electrically Erasable Programmable ROM (EEPROM)

Flash ROM

Source:W&H

Basic Random Access Memory Array Structure

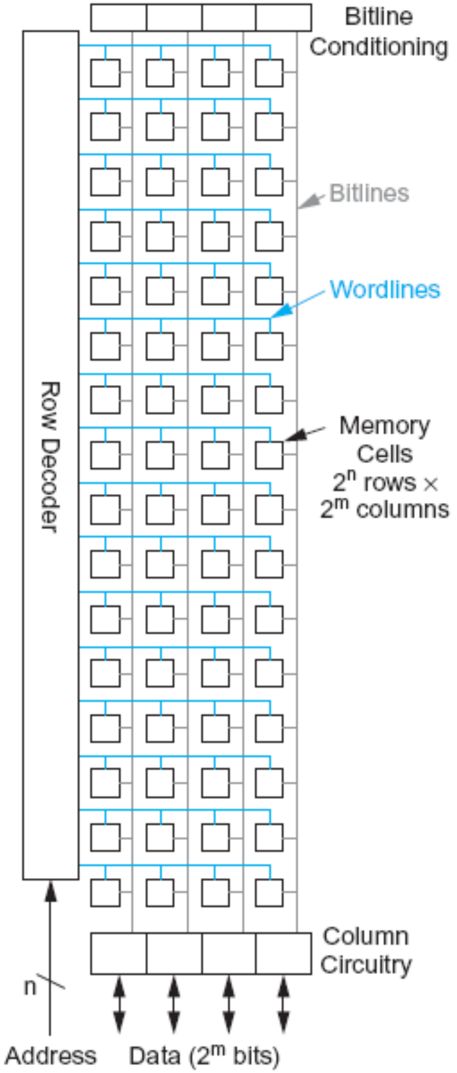


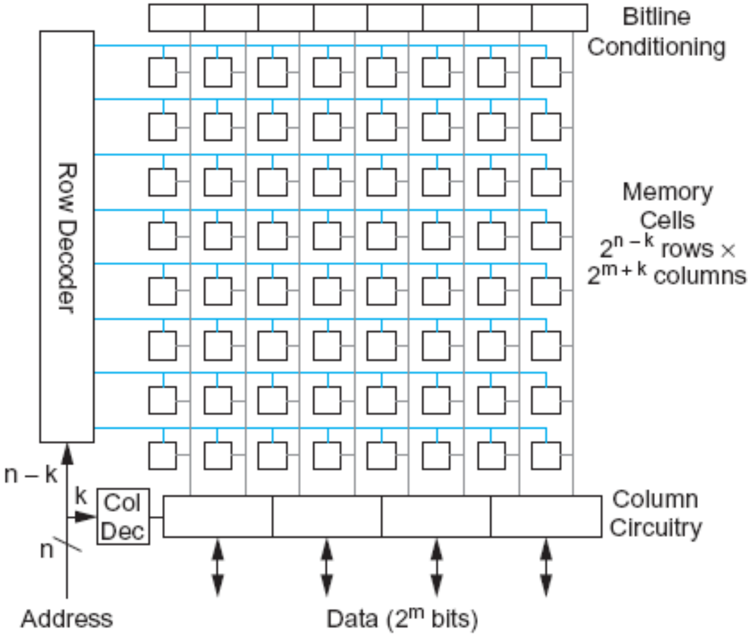
- Random Access →Arbitrary data access order
- 2^n entry RAM
  - Query SRAM with n-bit address A[n-1:0]
  - Obtain 2^m bit data
  - 2MB memory with 64-bit data
    - m=6, n=15(32000 rows!)
    - Folding yields improved aspect ratio

Source:W&H

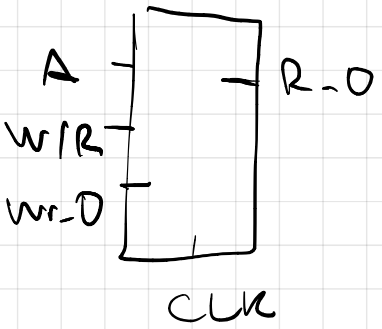
Basic Random Access Memory Array Structure

- Random Access →Arbitrary data access order
- 2^n entry RAM
  - Query SRAM with n-bit address A[n-1:0]
  - Obtain 2^m bit data
  - 2MB memory with 64-bit data
    - m=6, n=15(32000 rows!)
    - Folding yields improved aspect ratio



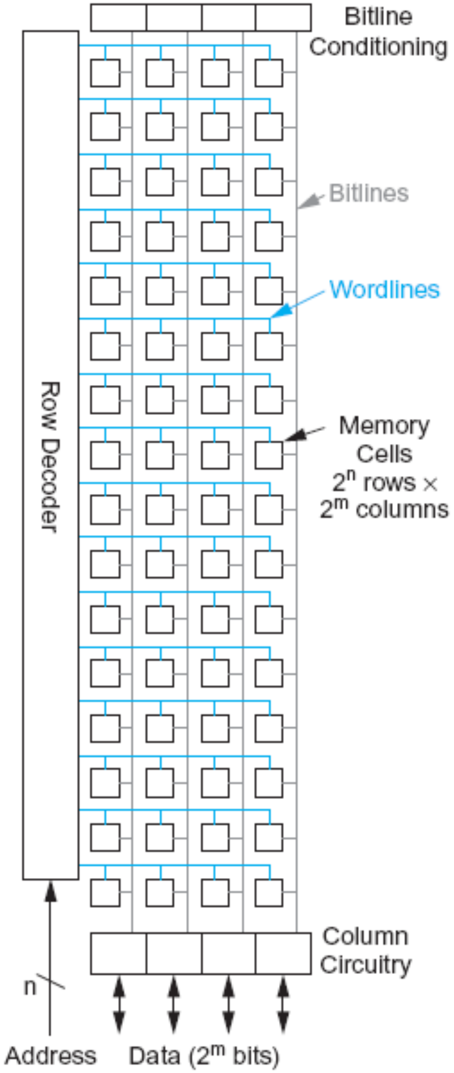


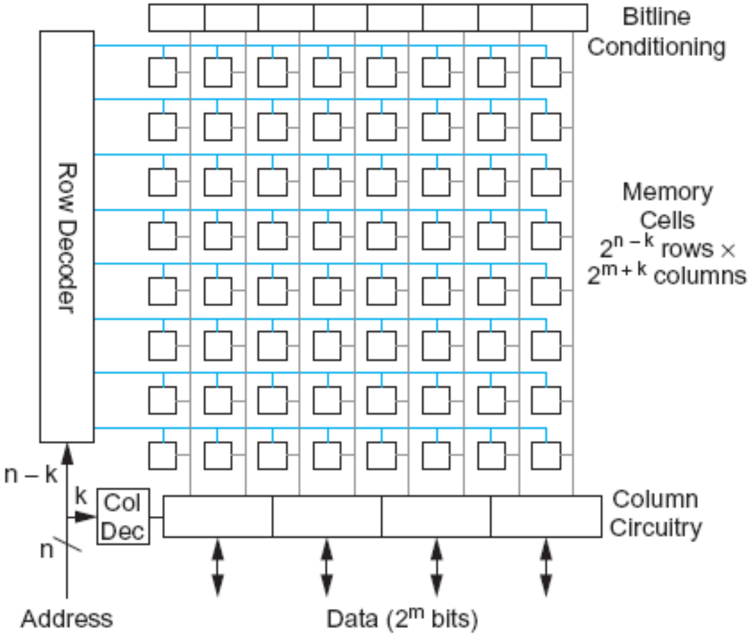
Source:W&H


A
W/R
Wr_D
R_O
CLK

Basic Random Access Memory Array Structure

- Random Access →Arbitrary data access order
- 2^n entry RAM
  - Query SRAM with n-bit address A[n-1:0]
  - Obtain 2^m bit data
  - 2MB memory with 64-bit data
    - m=6, n=15(32000 rows!)
    - Folding yields improved aspect ratio



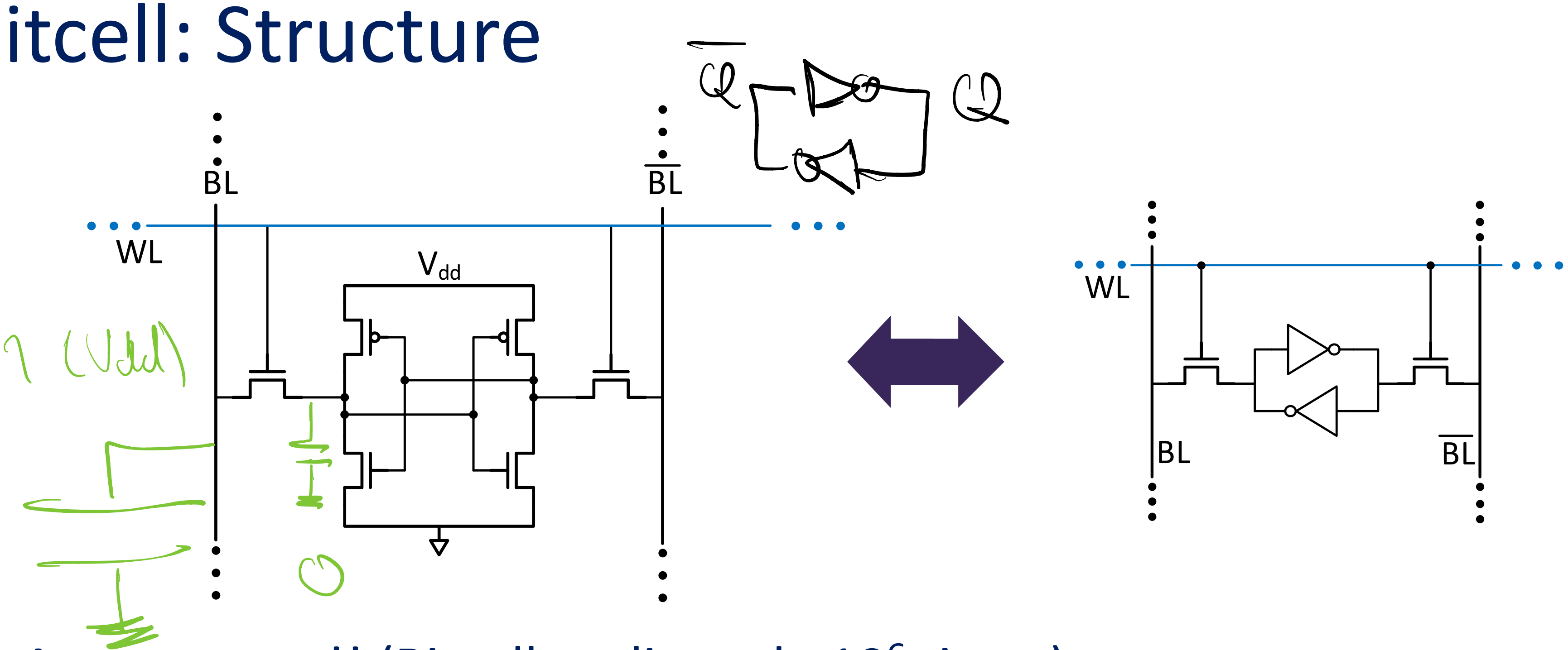


Additional notation

- Bitcell
- Ports

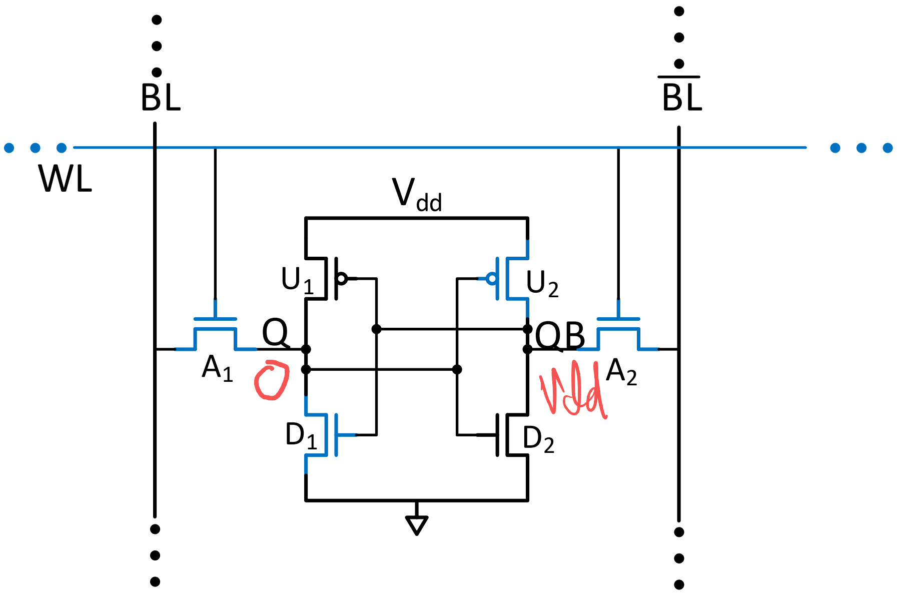
Source:W&H

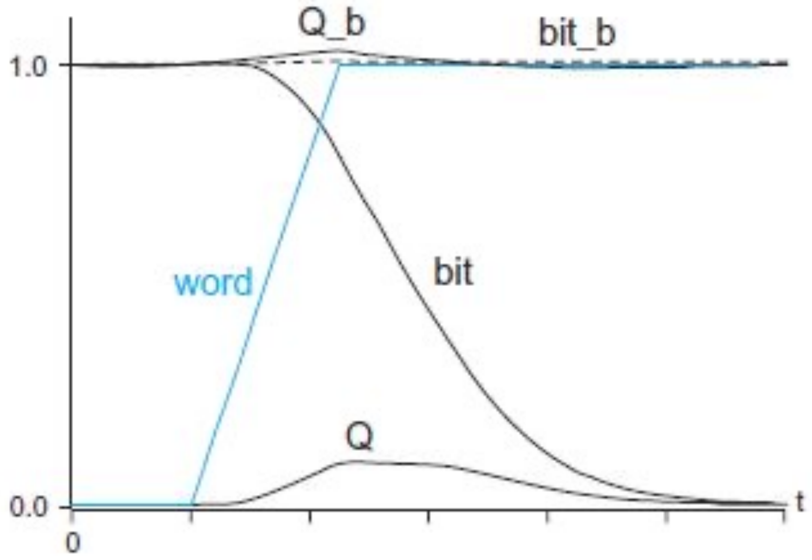
Bitcell: Structure

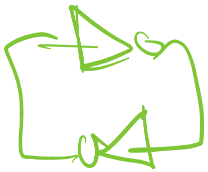


- Area matters!! (Bitcell replicated >10^6 times)
- Wordline turns-on access transistors (only NMOS to save area)
- Back-to-back inverter structures enable static data storage
- Bitline is highly capacitive (wire-load, bit-cell loading)
  - Driving bitline to 1 is difficult, area consuming
  - Pre-charge the bitline to V_{dd}, bitline only responsible for pulldown
  - Read circuits in large memory structures don't wait for bitline to reach rails*
  - Need for dual-rail bitlines (Differential vs. single-ended sensing)

Operation (Read)

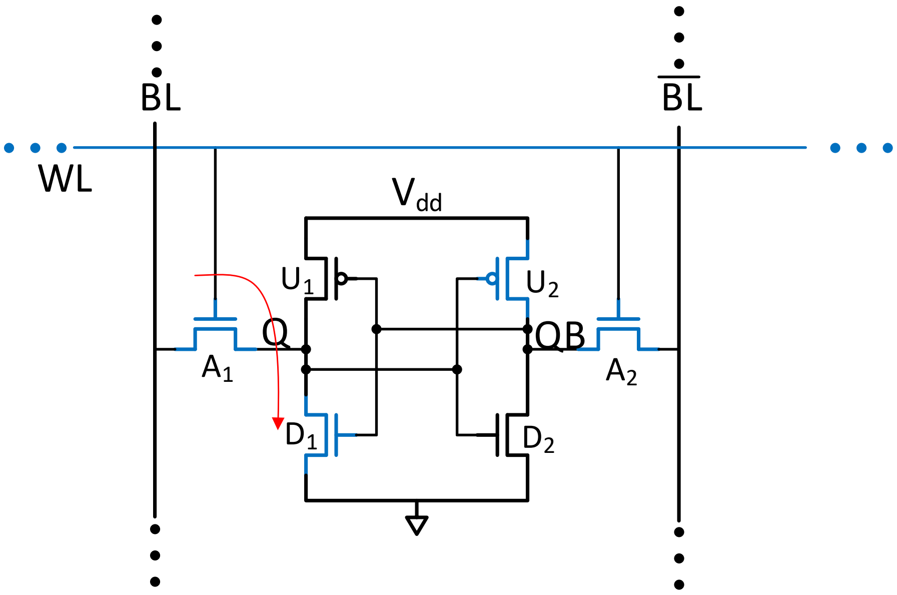


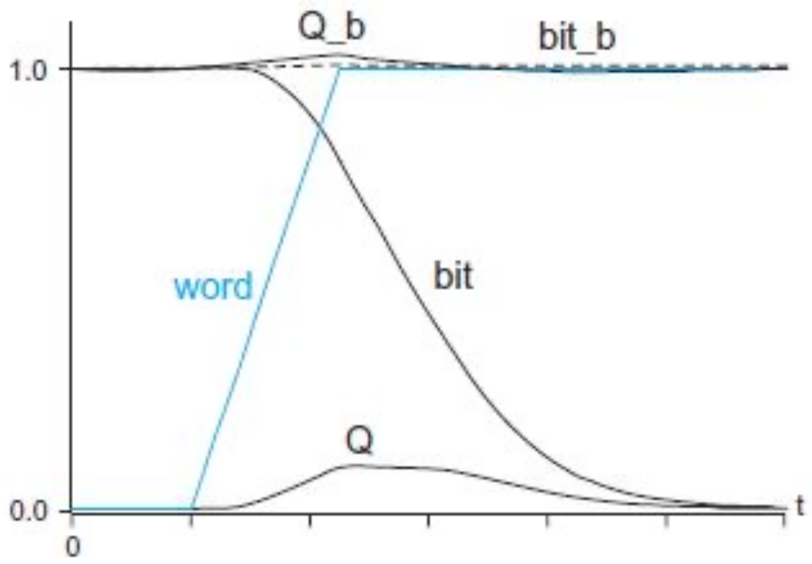




- Successful Read
  - Precharge bit-lines to V_{dd}
  - Activate word-line (turn on access transistors)
  - Voltage divider with A_1, D_1 as Q pulls *bit* low*, experiences voltage rise
  - Q_b, *bit_b* polarities match (*bit_b* weakly helps stabilize Q_b)
  - Q pulled down to gnd through D_1

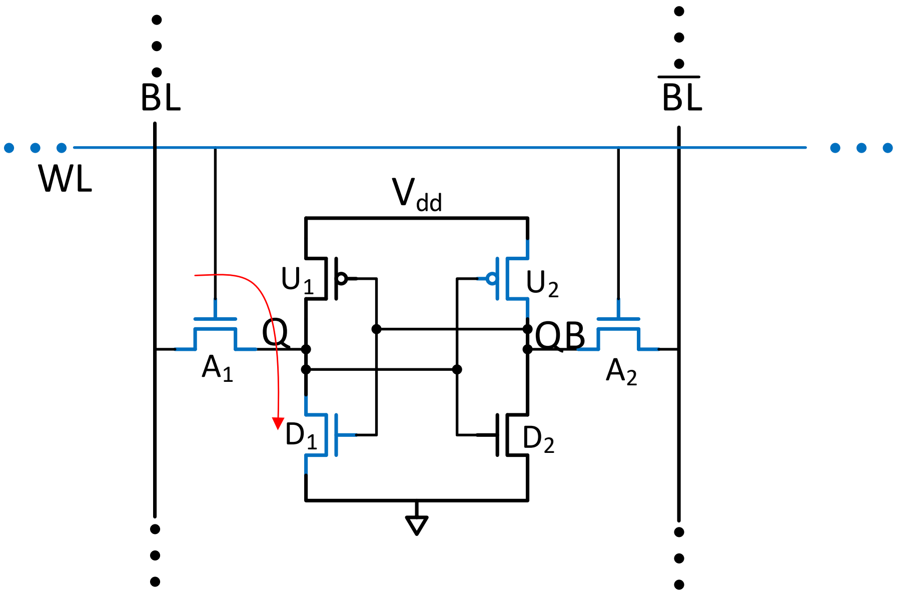
Operation (Read)

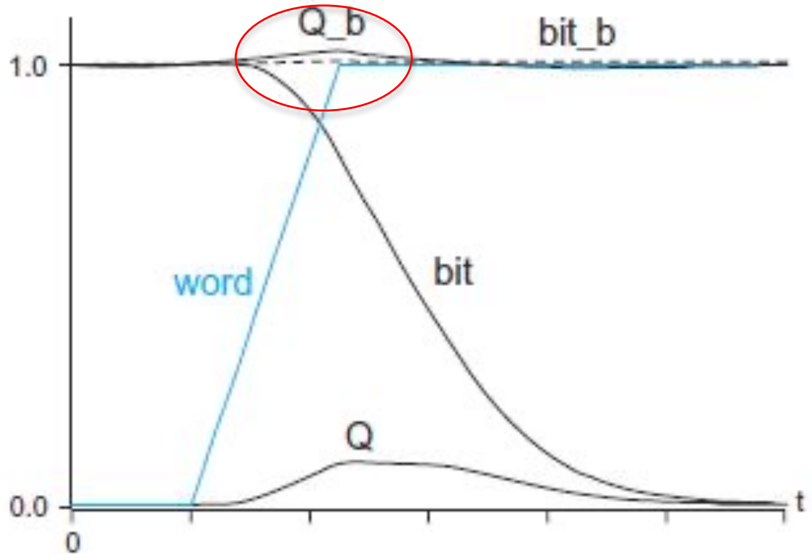




- Successful Read
  - Precharge bit-lines to V_{dd}
  - Activate word-line (turn on access transistors)
  - Voltage divider with A_1, D_1 as Q pulls *bit* low*, experiences voltage rise
  - Q_b, *bit_b* polarities match (*bit_b* weakly helps stabilize Q_b)
  - Q pulled down to gnd through D_1

Operation (Read)

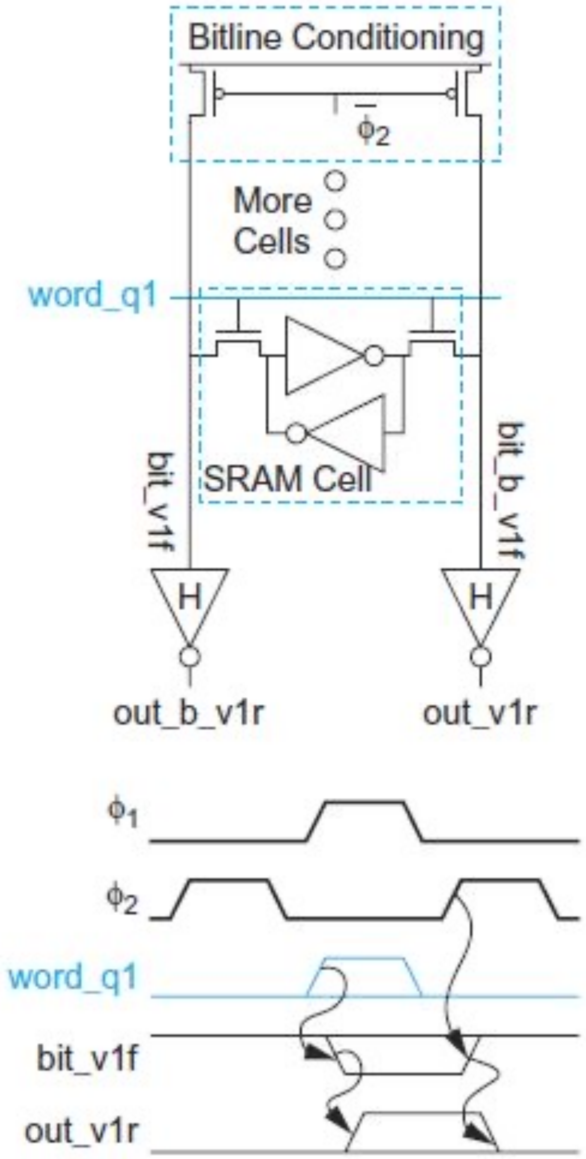




- Successful Read
  - Precharge bit-lines to V_{dd}
  - Activate word-line (turn on access transistors)
  - Voltage divider with A_1, D_1 as Q pulls *bit* low*, experiences voltage rise
  - Q_b, *bit_b* polarities match (*bit_b* weakly helps stabilize Q_b)
  - Q pulled down to gnd through D_1

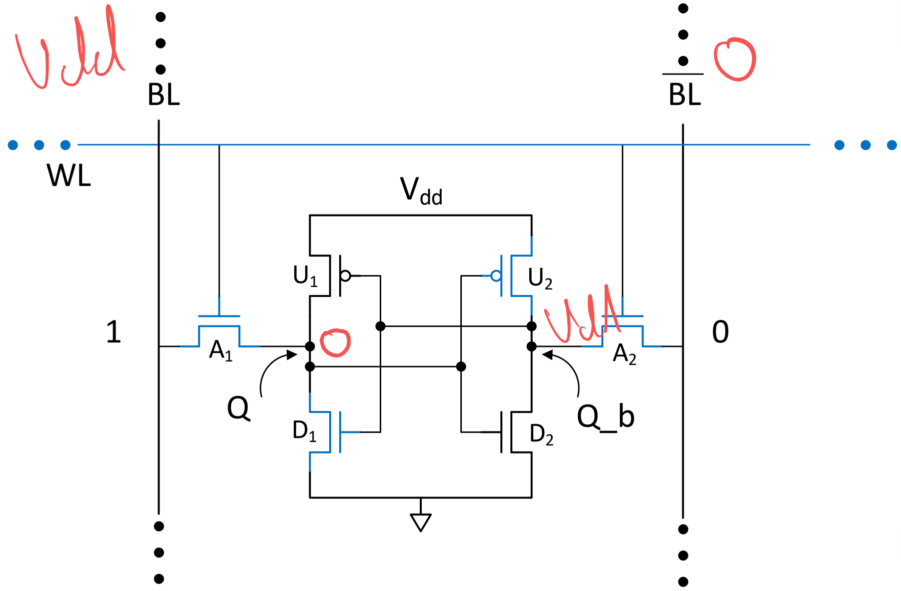
Array Read Timing

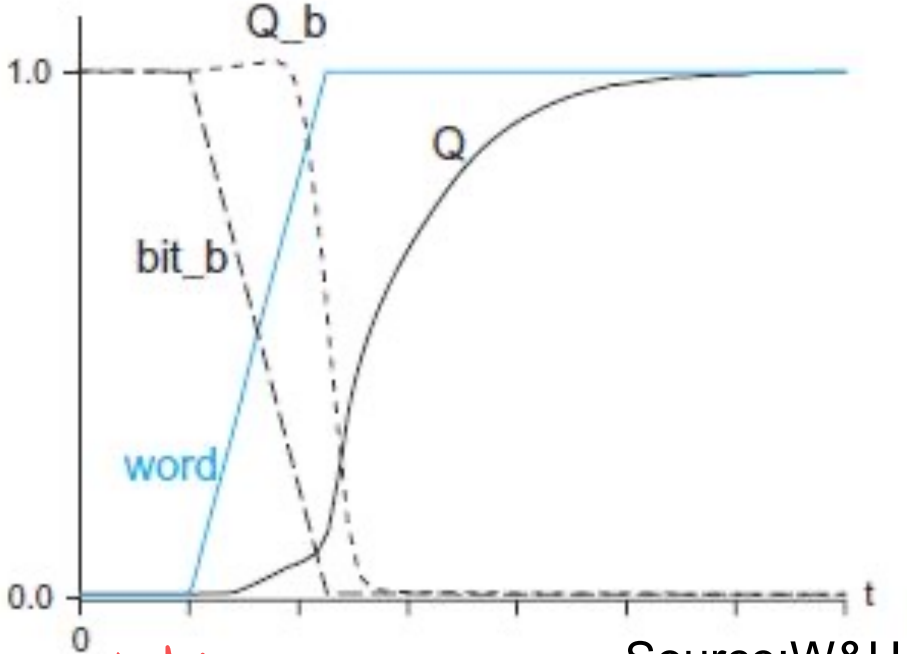
- 2-stage process
  - 1st stage: precharge bitlines
  - 2nd stage: access bitcell
- *bit_v1f* does not have to go to 0
  - Differential read performed by sense-amplifiers



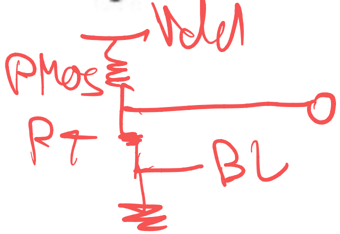
Source:W&H

Operation (Write)



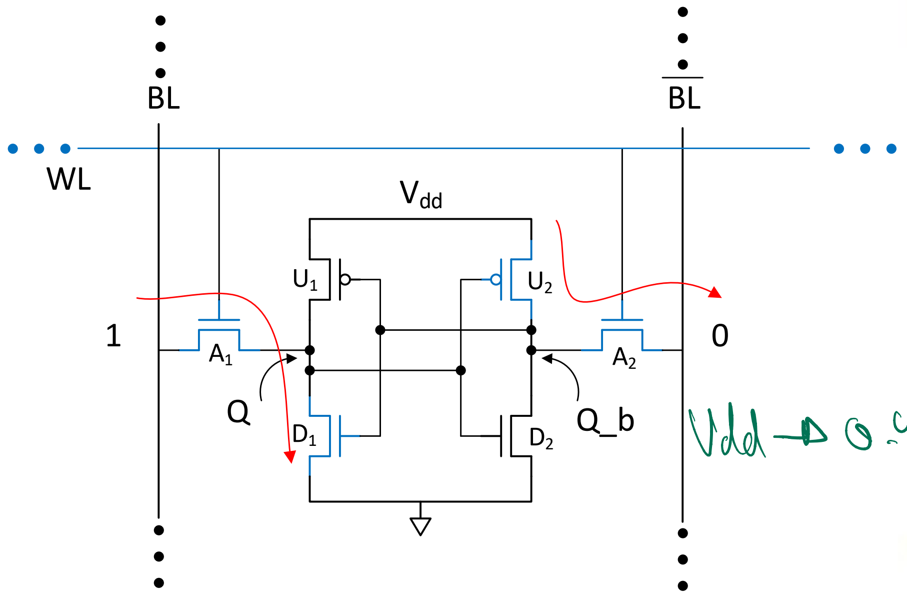


Source:W&H

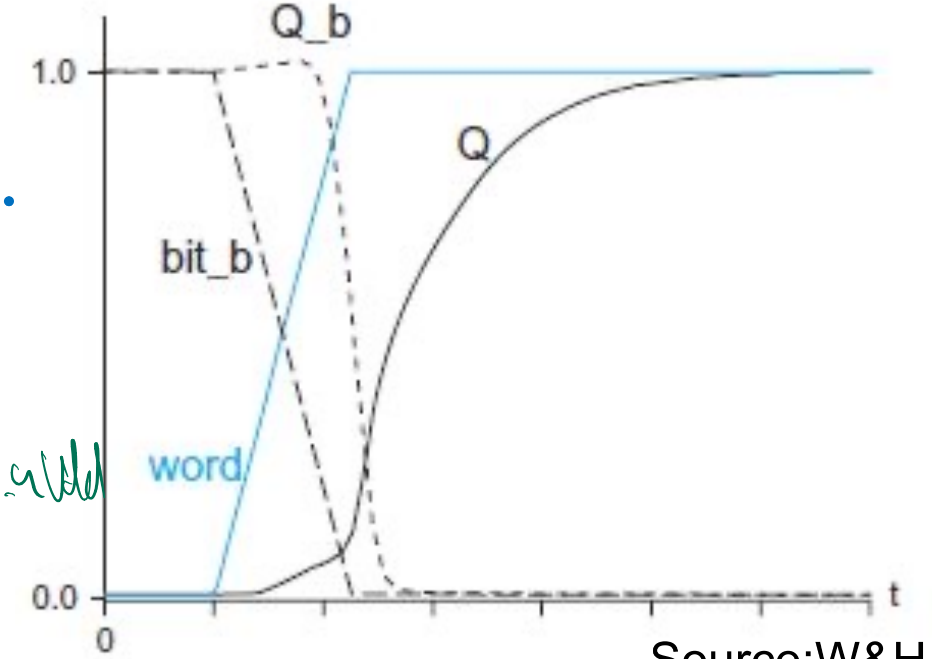


- Successful Write
  - *bit, bit_b* driven as per write-data
  - Activate word-line (turn on access transistors)
  - If stored data disagrees with write-data, fight ensues
    - Voltage divider arrangement between A_2, U_2* and A_1,D_1*
    - *Q_b* pulled toward gnd. → U_1 and A_1 combine to overcome D_1 *Q* transitions upward
    - Positive feedback: Q Transition → D_2 assists in pulldown, U_1 asserts pull-up

Operation (Write)



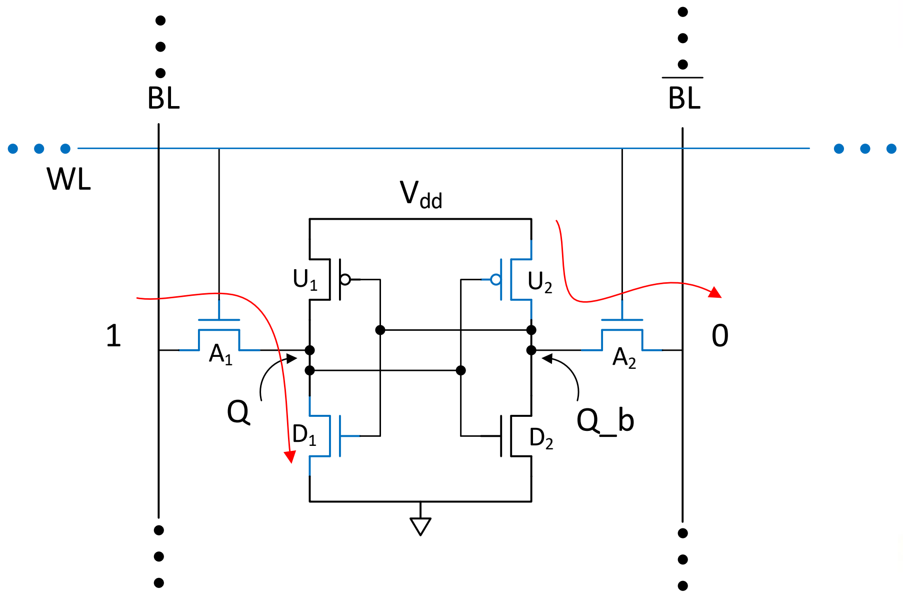
Vdd → 0.9Vdd

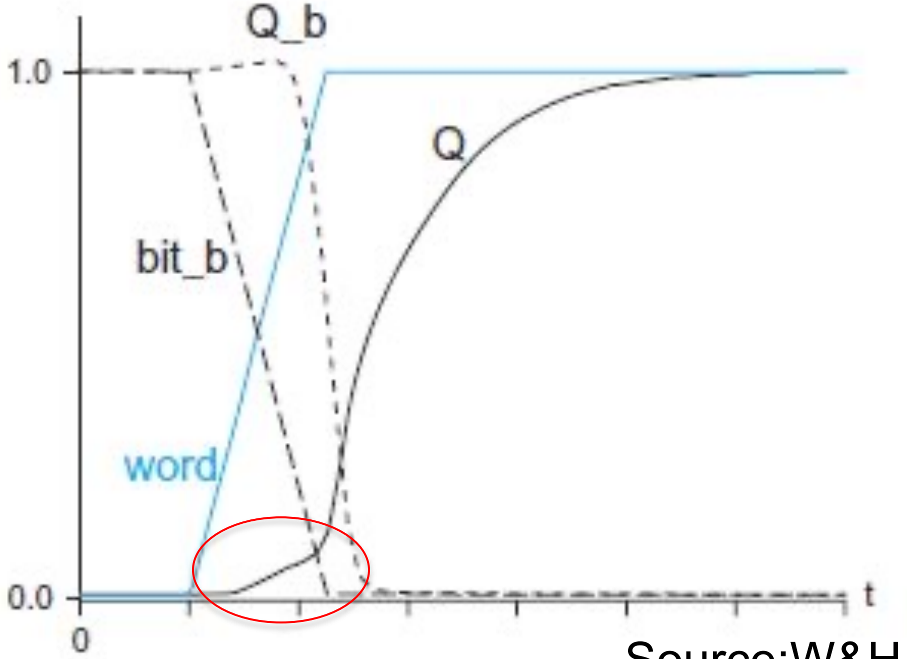


Source:W&H

- Successful Write
  - *bit, bit_b* driven as per write-data
  - Activate word-line (turn on access transistors)
  - If stored data disagrees with write-data, fight ensues
    - Voltage divider arrangement between A_2, U_2* and A_1,D_1*
    - *Q_b* pulled toward gnd. → U_1 and A_1 combine to overcome D_1 *Q* transitions upward
    - Positive feedback: Q Transition → D_2 assists in pulldown, U_1 asserts pull-up

Operation (Write)

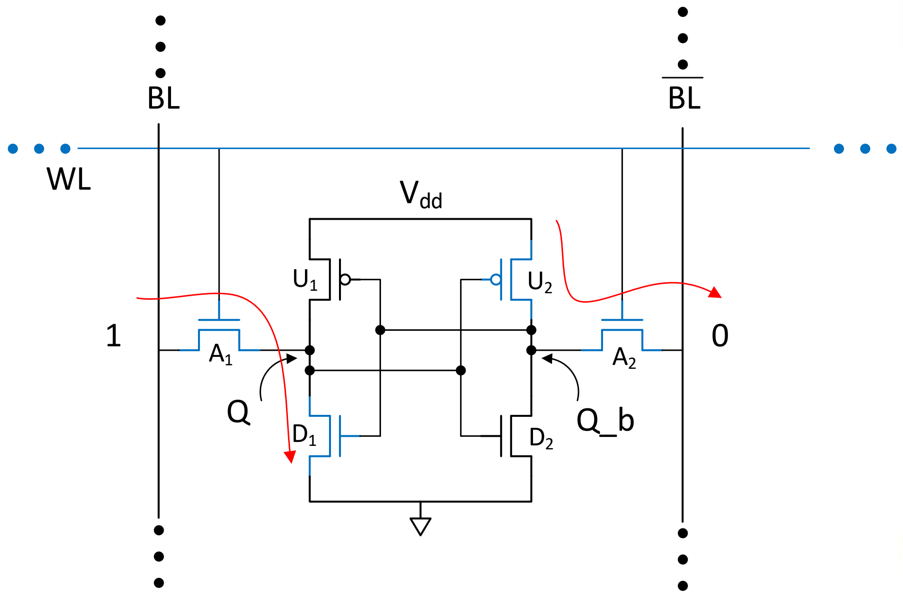


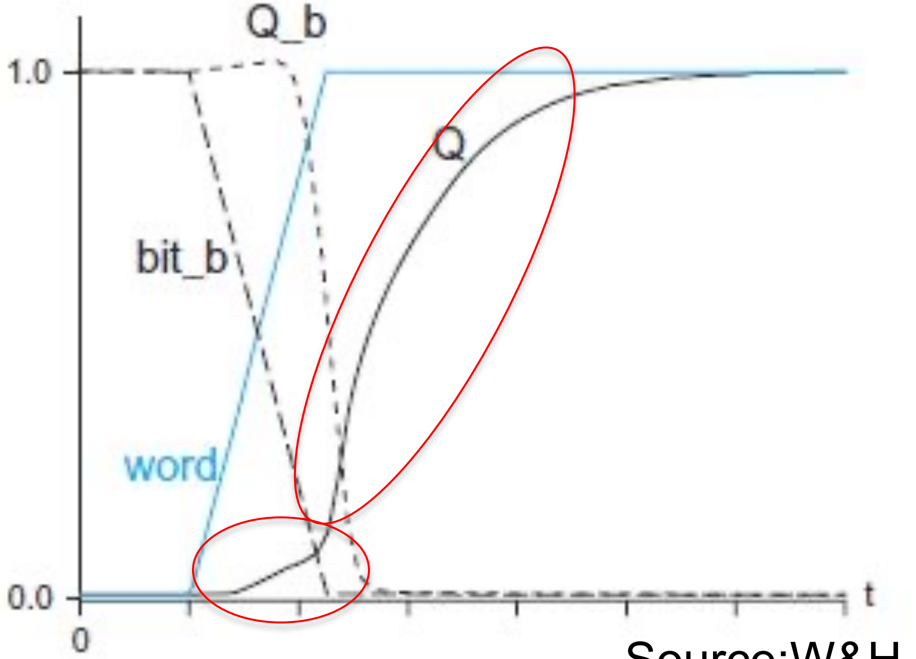


Source:W&H

- Successful Write
  - *bit, bit_b* driven as per write-data
  - Activate word-line (turn on access transistors)
  - If stored data disagrees with write-data, fight ensues
    - Voltage divider arrangement between A_2, U_2* and A_1,D_1*
    - *Q_b* pulled toward gnd. → U_1 and A_1 combine to overcome D_1 *Q* transitions upward
    - Positive feedback: Q Transition → D_2 assists in pulldown, U_1 asserts pull-up

Operation (Write)

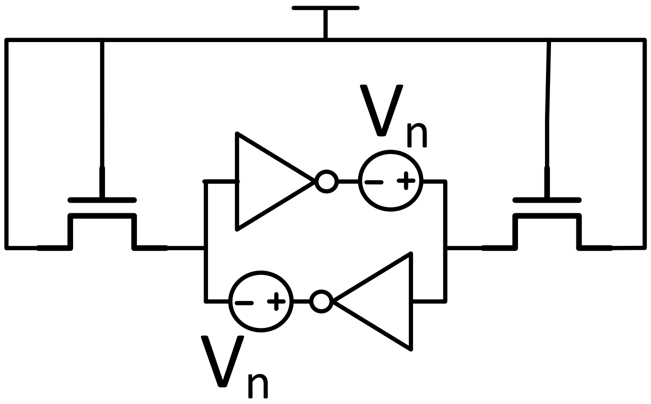




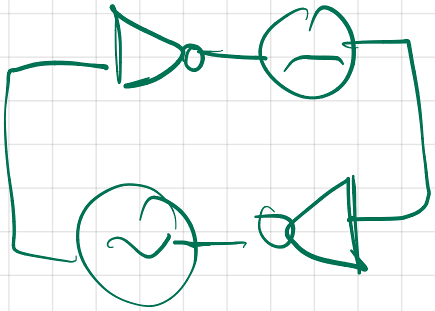
Source:W&H

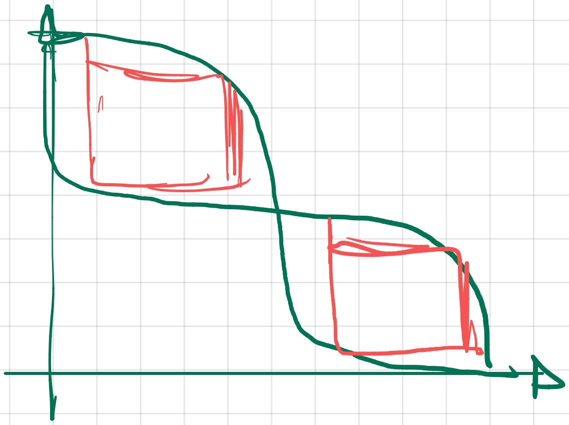
- Successful Write
  - *bit, bit_b* driven as per write-data
  - Activate word-line (turn on access transistors)
  - If stored data disagrees with write-data, fight ensues
    - Voltage divider arrangement between A_2, U_2* and A_1,D_1*
    - *Q_b* pulled toward gnd. → U_1 and A_1 combine to overcome D_1 *Q* transitions upward
    - Positive feedback: Q Transition → D_2 assists in pulldown, U_1 asserts pull-up

Stability

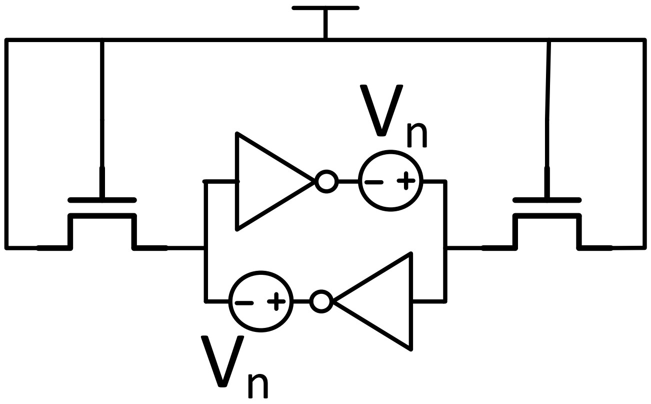


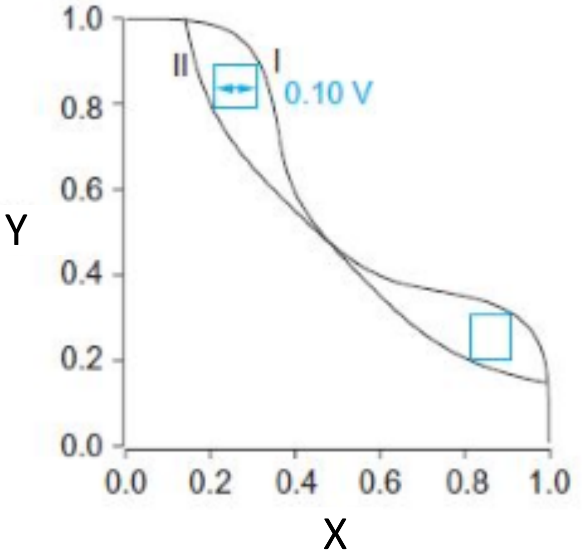
- Fight during read and write operation in a latch can result in destructive reads and incomplete write
  - Bitcell tradeoff between size and robustness (V_{min})
  - Bitcell relative sizing in descending order?:
- Static Noise Margins are popular for stability analysis **(Exercise)**
  - V_{th} variations are significant source of instability (→Static margin analysis)
  - DC representation and analysis of a transient phenomenon
  - Millions of cells, PVT variation over a broad area
  - Heavily statistical in nature





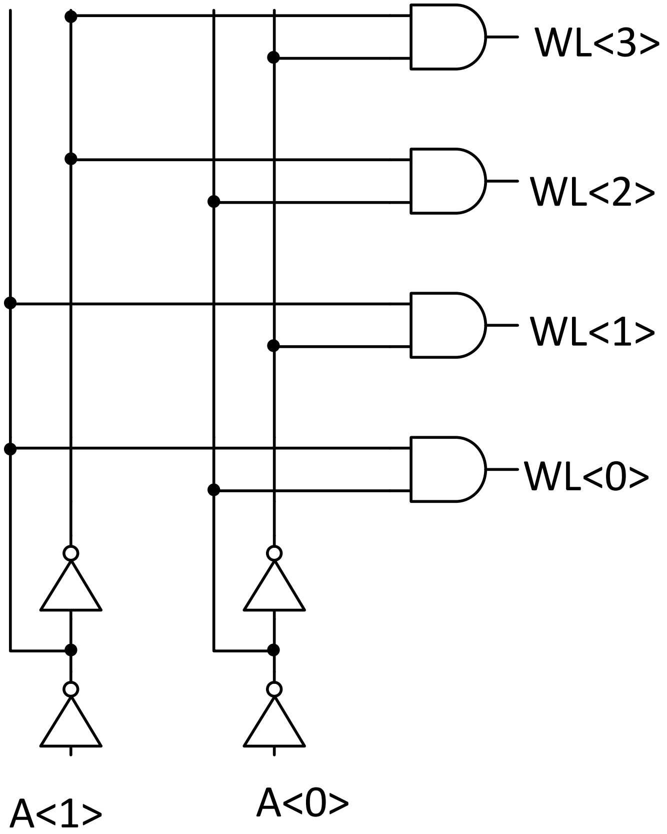
Stability





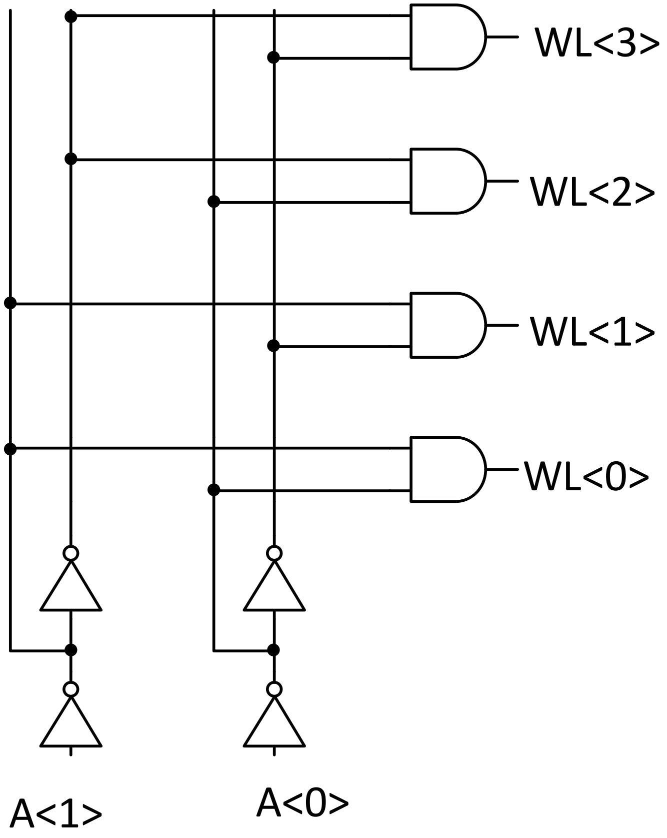
- Fight during read and write operation in a latch can result in destructive reads and incomplete write
    - Bitcell tradeoff between size and robustness (V_{min})
    - Bitcell relative sizing in descending order?:
- Static Noise Margins are popular for stability analysis **(Exercise)**
    - V_{th} variations are significant source of instability (→Static margin analysis)
    - DC representation and analysis of a transient phenomenon
    - Millions of cells, PVT variation over a broad area
    - Heavily statistical in nature

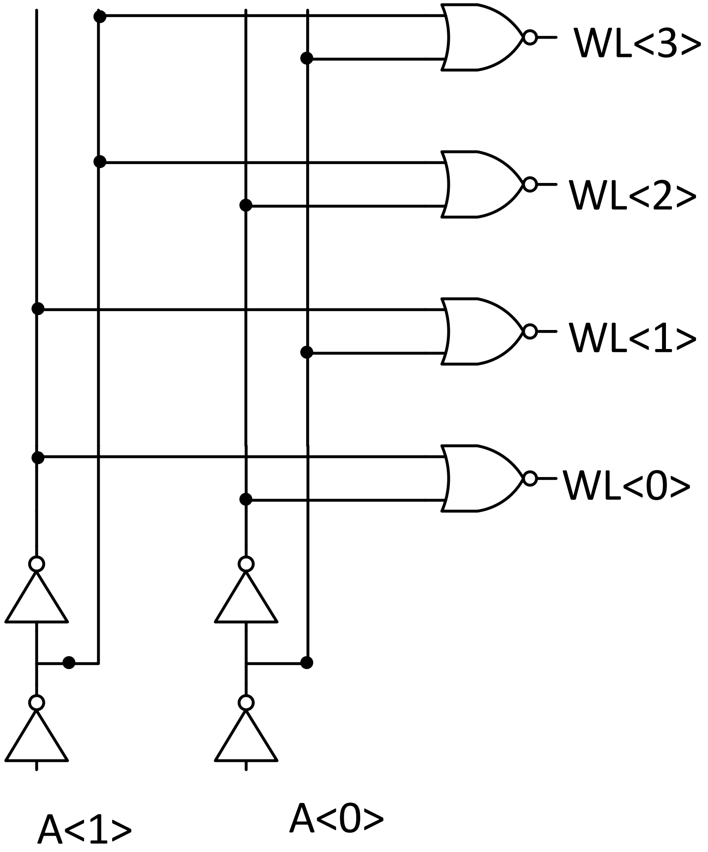
Decoder Design



- Selecting which row to access involves decoding the address
- Decoder design provides necessary functionality
- **Exercise**: Bubble-propagate to implement the decode efficiently
- Total Latency depends on decode, bitcell access, bitline read, column mux
- WL<n-1:0> lines run along memory rows. **Need to match height!! (pitch-matching)**

Decoder Design





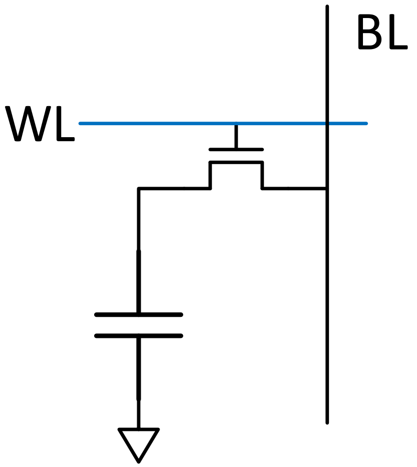
- Selecting which row to access involves decoding the address
- Decoder design provides necessary functionality
- **Exercise**: Bubble-propagate to implement the decode efficiently
- Total Latency depends on decode, bitcell access, bitline read, column mux
- WL<n-1:0> lines run along memory rows. **Need to match height!! (pitch-matching)**

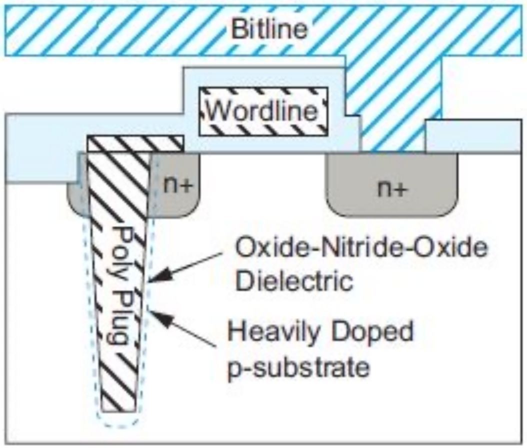
Tolerating Faults

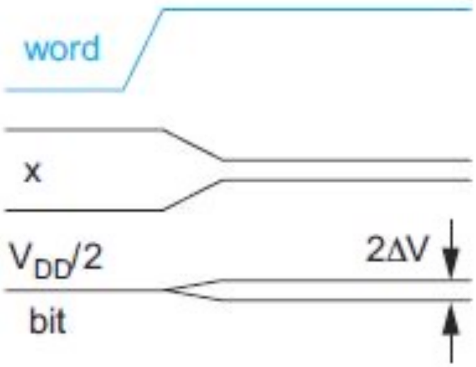
- Large number of memory cells
- Very difficult to guarantee that every single cell will work
- Structure and functionality allow for error-correction
- Error Detection and Correction
  - Maintain required hamming distance between valid symbols
  - If d is the minimum hamming distance between symbols
    - Possible to detect up to d-1 errors
    - Possible to correct for floor[(d-1)/2]

- Basic Block Code Example (7,4 Hamming Code):
  - 5-bits of data stored as 4b1011
  - Store a byte of data: 7b101$x_3$1x_1x_0
    - $X_3 = X_6$ XOR X_5 XOR X_4 = 1 XOR 0 XOR 1 = 0
    - $X_1 = X_6$ XOR X_5 XOR X_2 = 1 XOR 0 XOR 1 = 0
    - $X_0 = X_6$ XOR X_4 XOR X_2 = 1 XOR 1 XOR 1 = 1
    - Store : 7b1010101

DRAM





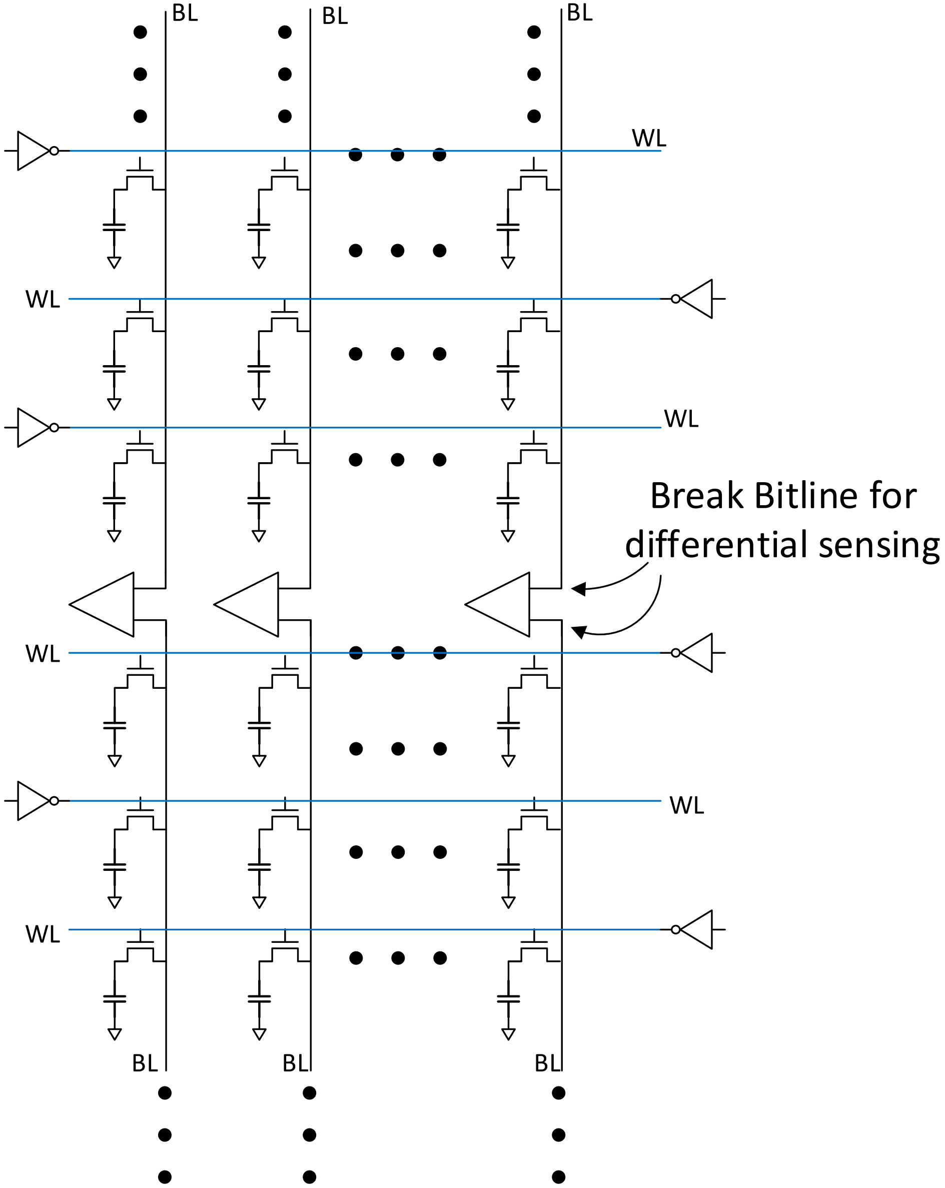


Source:W&H

- Dynamic RAM
  - Memory content stored as the charge on a capacitor (Refresh needed)
  - Very high density (order of magnitude higher than SRAM)
    - Special processing needed (Trench Capacitors). *Typically* incompatible with CMOS
  - Read Operation
    - Precharge bit-line to $V_{DD}/2$
    - Turn on access transistor
    - Detect small-signal transition on bitline, amplify **and regenerate** bitline to write-back data
  - Write Operation: Drive bitline, enable access transistor, charge/discharge node

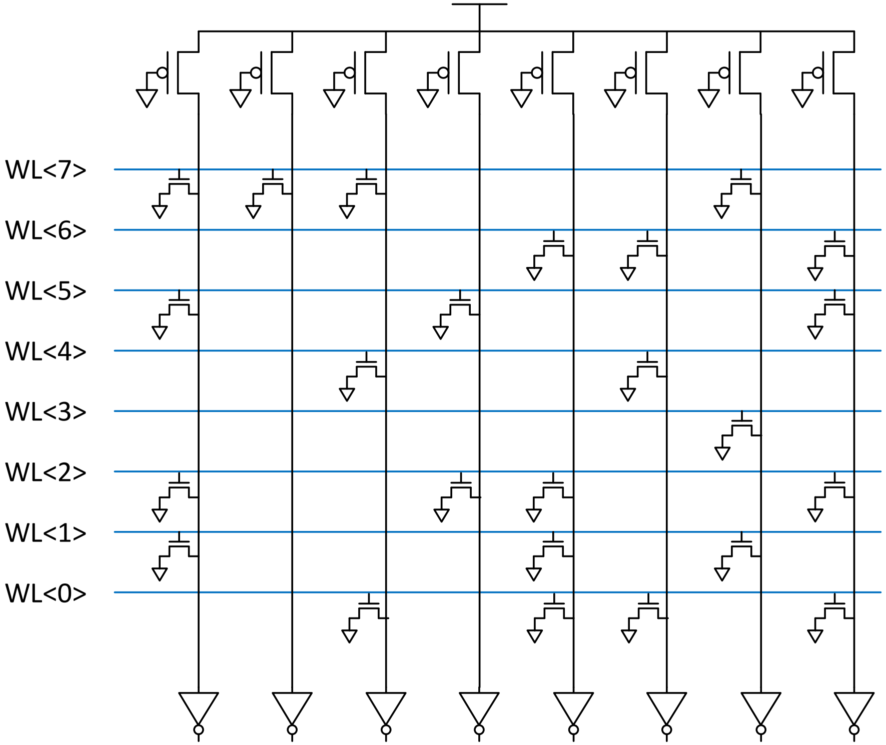
DRAM Array

- Many bitcells hanging off single bitline
  - Bitline capacitance >> bitcell capacitance
- Sense amplifier detects if bitline moves ↑ or ↓below $V_{dd}/2$
  - Use “control” bitline for differential sensing
  - Enables breaking up the bitline (mitigate bitline capacitance)



Break Bitline for differential sensing

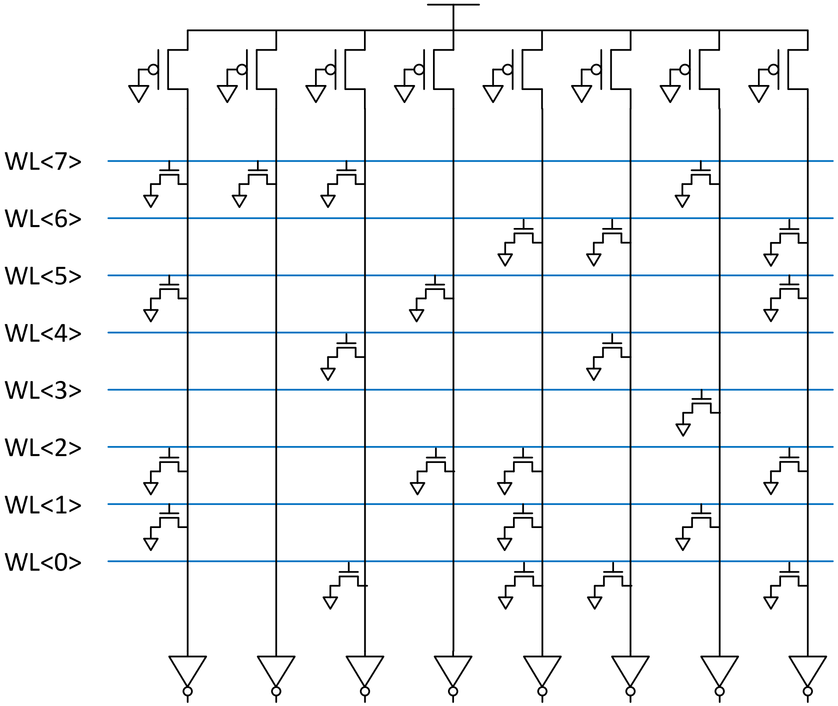
Mask ROMs

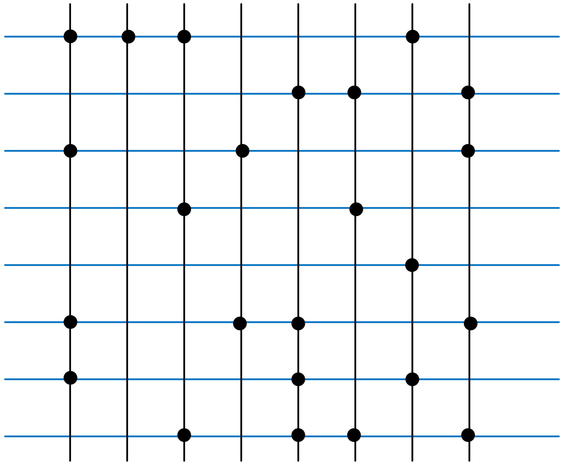


NOR-ROM

- Most compact on-chip memory storage
  - Microcode in processors
  - Signal processing (FIR Filters, FFT coefficients)
- 2-variants
  - Pseudo-nMOS logic based NOR gates
  - Pseudo-nMOS NAND ROM

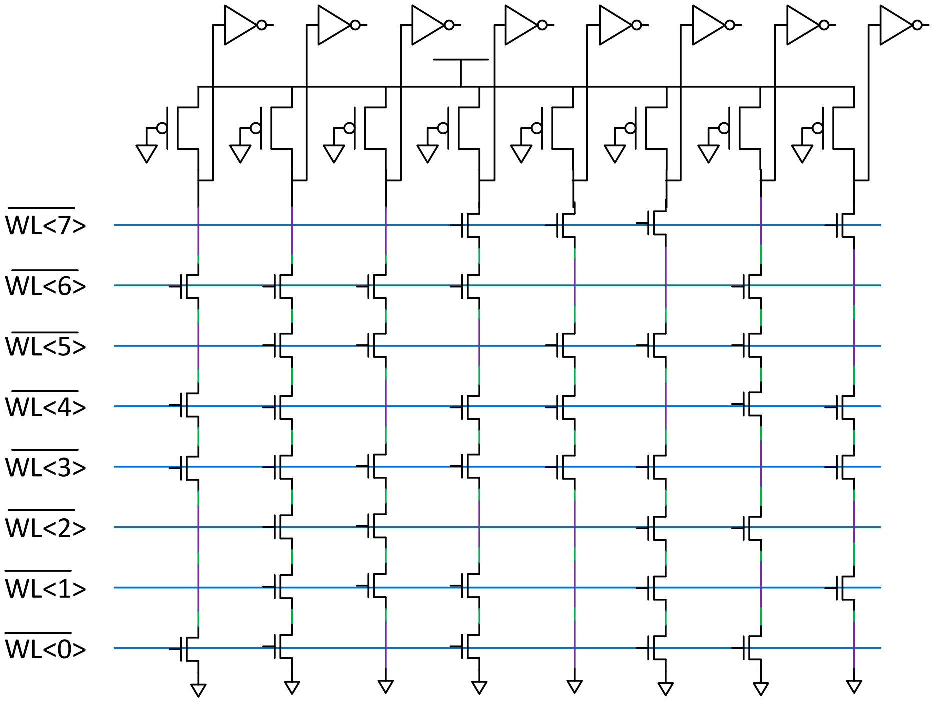
NOR ROMs

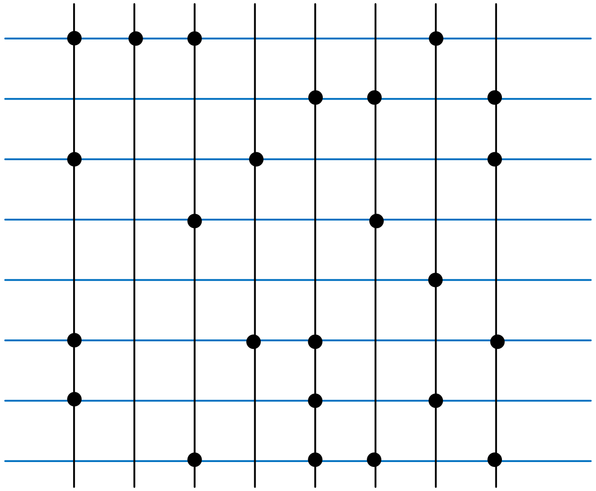




- Pseudo-nMOS NOR implementation (Avoid long, slow pullup)
  - Parallel NMOS connections
  - No full-rail swing
- For a given wordline, connect tx to bitline if corresponding bit is 1
  - Selection of wordline (active-high) pulls down bit-line, transitions output to 1

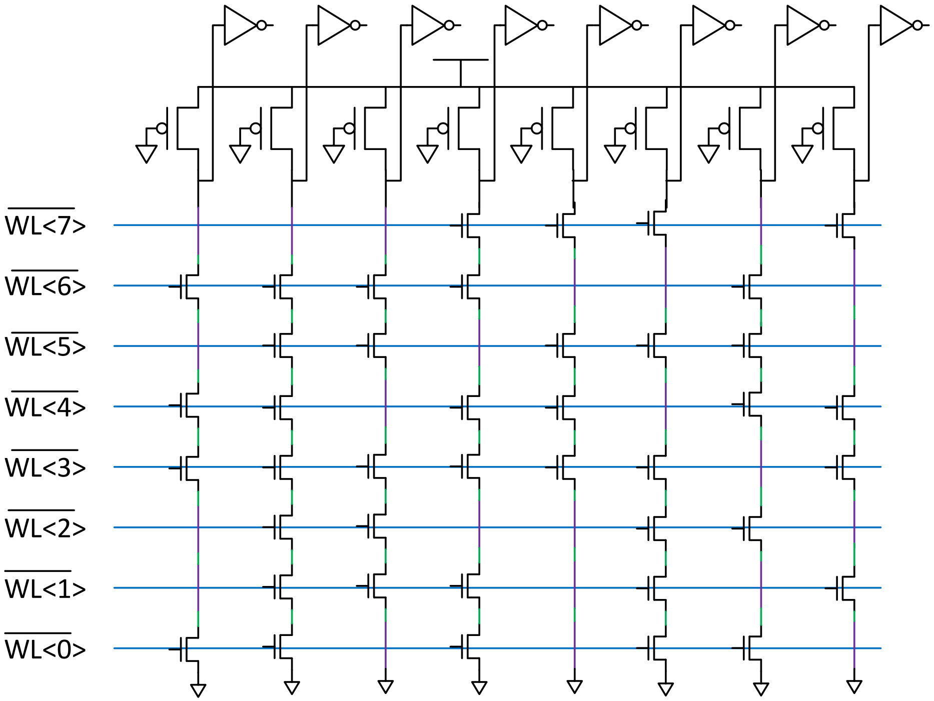
NAND ROMs


$\overline{\text{WL<7>}}$
$\overline{\text{WL<6>}}$
$\overline{\text{WL<5>}}$
$\overline{\text{WL<4>}}$
$\overline{\text{WL<3>}}$
$\overline{\text{WL<2>}}$
$\overline{\text{WL<1>}}$
$\overline{\text{WL<0>}}$

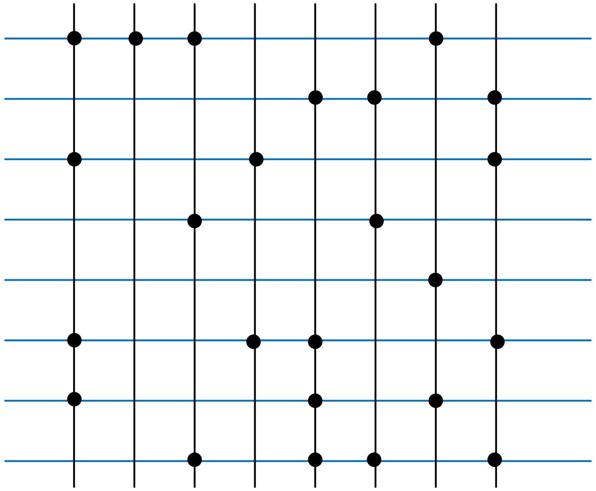


- Pseudo-nMOS NAND implementation
  - Series nMOS devices (Slower than NOR)
  - No full-rail swing
- Decode lines are active-low: All but selected WLs are 1
  - If transistor is present, WL=0 → BL remains high, output is low
  - If transistor not present, metal connection, WL=0 → BL pulldown, output 1
  - Insert transistor where you want to program a 0

NAND ROMs



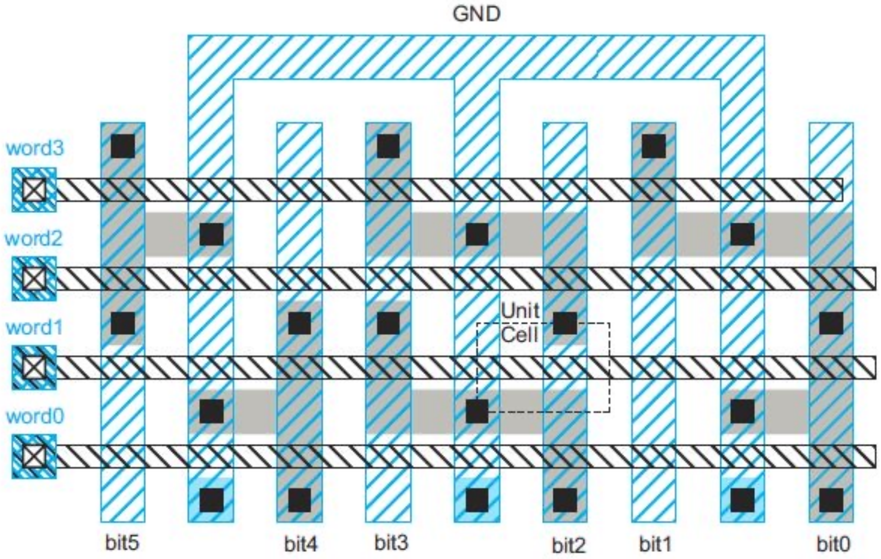
$\overline{\text{WL<7>}}$
$\overline{\text{WL<6>}}$
$\overline{\text{WL<5>}}$
$\overline{\text{WL<4>}}$
$\overline{\text{WL<3>}}$
$\overline{\text{WL<2>}}$
$\overline{\text{WL<1>}}$
$\overline{\text{WL<0>}}$



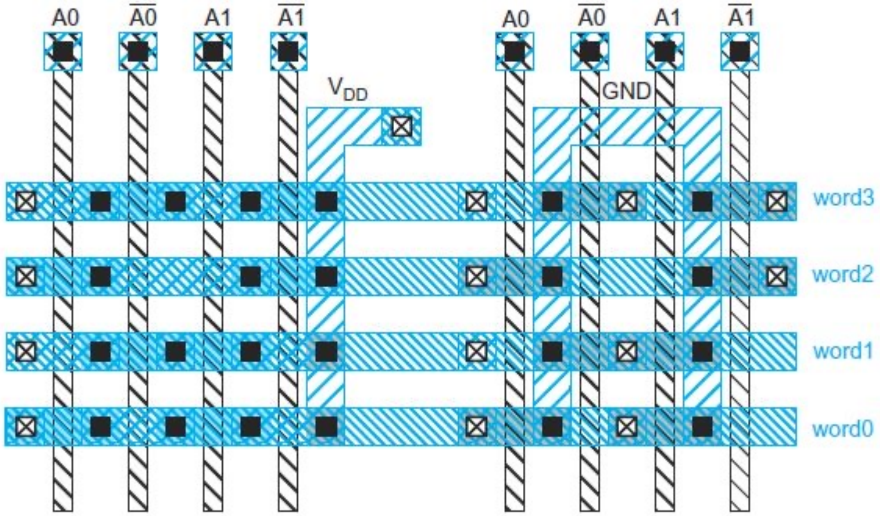
- Pseudo-nMOS NAND implementation
  - Series nMOS devices (Slower than NOR)
  - No full-rail swing
- Decode lines are active-low: All but selected WLs are 1
  - If transistor is present, WL=0 → BL remains high, output is low
  - If transistor not present, metal connection, WL=0 → BL pulldown, output 1
  - Insert transistor where you want to program a 0

So..what's the benefit?

Layout Considerations



ROM Array

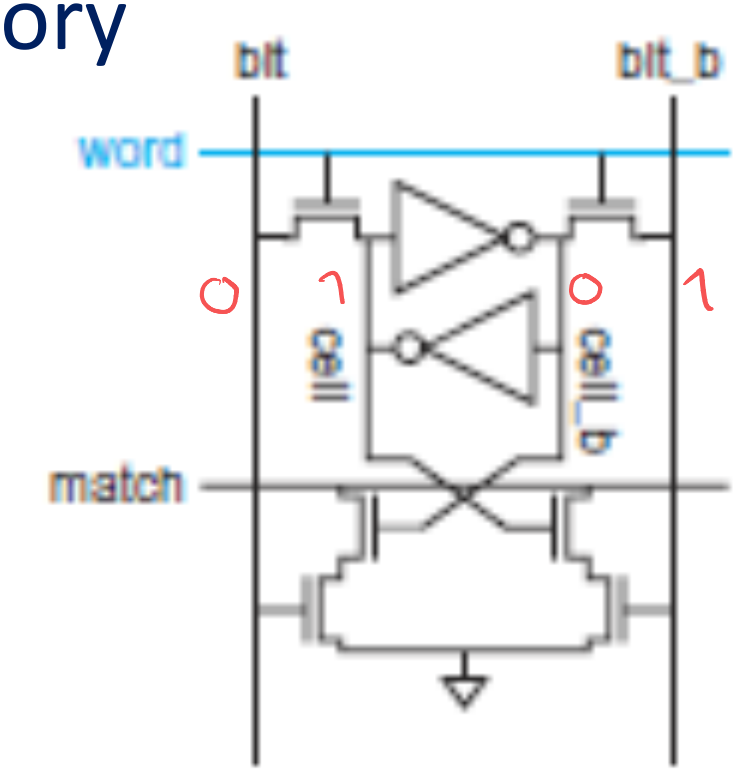


Array Decoder

Source:W&H

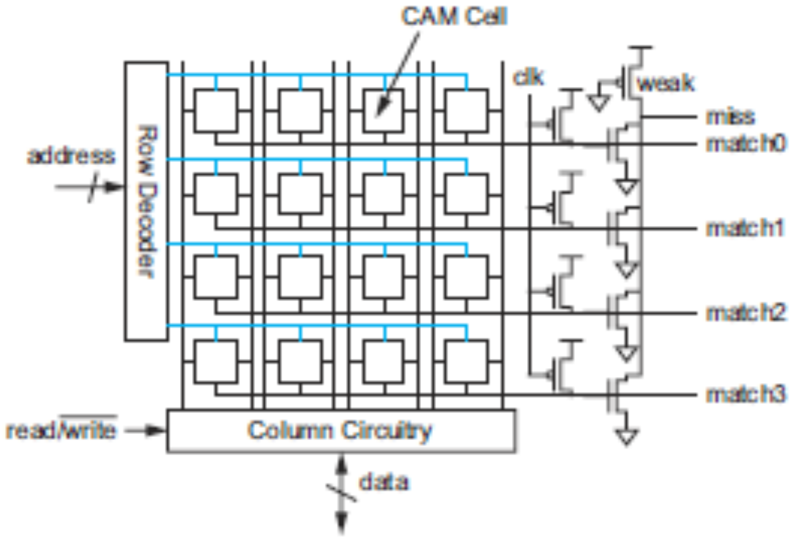
- ROM offers ~1 bit per transistor area, **very** compact
- Each bitcell is essentially 1 transistor high
  - Wordlines are spaced very close together
  - Decoder design needs some consideration (need to pitch match!)

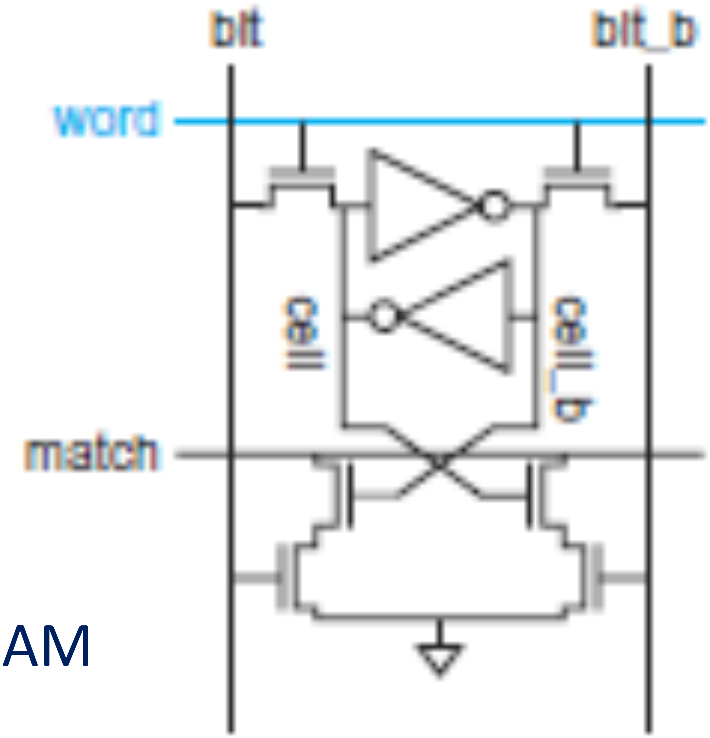
Content Addressable Memory



- Extended SRAM
  - Capable of random read/write access into the array
  - “Content Addressable” capability. Match *data* query with content in memory and return row index (indices) that match to it
- Extensive usage in
  - Accelerators (Hashing)
  - Processors (Associative Cache Memory, TLB)
  - Routers

CAM Operation





- Random Read/Write similar to SRAM
- Matching operation
  - Pre-charge matchline to V_{dd}
  - Broadcast query on bitlines across the array
  - If a cell entry mismatches with a query, discharge matchline
  - Most (even all) matchlines discharge for each query (dissipative)
  - Matching lines are either priority encoded or combined for a single match/miss result
  - Segmented matching, NAND CAMs trade off speed for efficiency

Questions/Assignments

Reading

- Required W&H
    - SRAM 12.2.1-12.2.1.4
    - DRAM 12.3-12.3.1
    - ROM 12.4-12.4.2, 12.4.3

- Optional
    - SRAM 12.2.1.4 – 12.2.3, 12.2.5