

EE-429

Fundamentals of VLSI Design

SRAM

Andreas Burg

Structure of Computing Systems

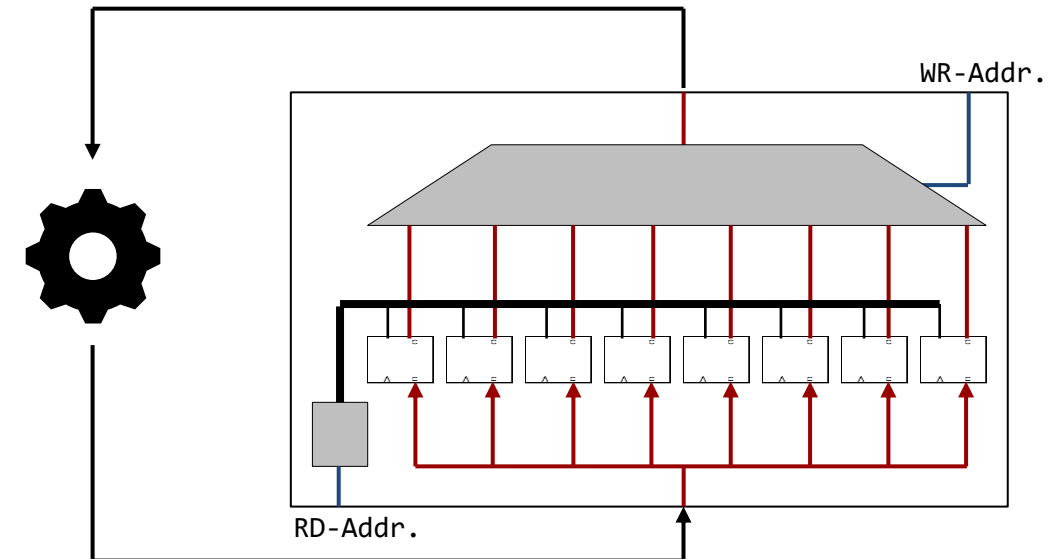
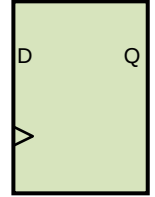
- **Computing systems are constructed from **combinatorial logic** and **memory****
 - Functional units fetch data from memory, process it and write back the result



- **Why do we need memory?**
 - Data that needs to be combined arrives one after another
 - Many computations are performed on a limited number of resources, one after another. Need to store intermediate results until they are processed further

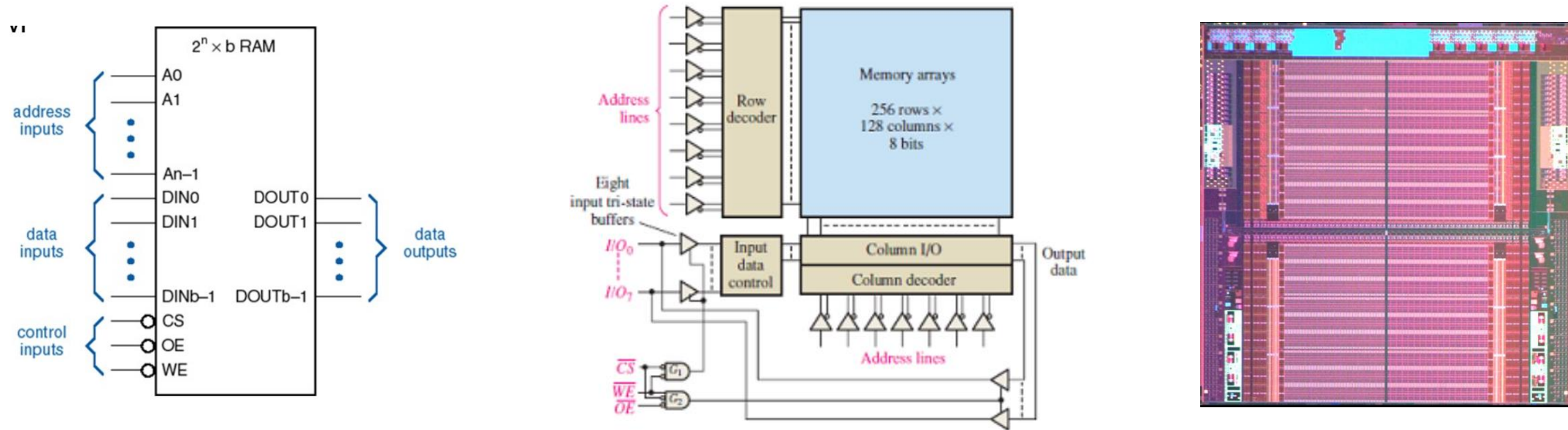
Registers are often Inefficient

- **Registers** are the most basic type of memory in a digital circuit
 - Parallel read access: data stored in a register is always available
 - Parallel write access: data can be stored in every register in each clock cycle
- **However**, when large amounts of data need to be stored, **consider that**
 - Each register is quite large (>20 transistors)
 - Selecting data from one of many registers requires significant overhead (MUX)
- **Further**, with few processing resources,
 - Parallel access to all memory locations is rarely required



Memory Arrays for Better Density

- To achieve better storage density, **memories group storage elements into a compact full-custom array** with custom access logic for R/W



- Full custom design enables a dense layout** and **allows for less conservative design** practices that save significant resources (e.g., ratioed-logic)

Many Different Types of Memories

- **Different types of memories** are optimize **for different objectives**

- **Capacity:** Bits, Bytes, Words, kBytes - TBytes

- **Integration:** On-chip vs. Off-chip

- **Persistence:**

- Read Only (ROM) – non-volatile
- Read-Write (RWM) – volatile
- NVRWM – Non-volatile

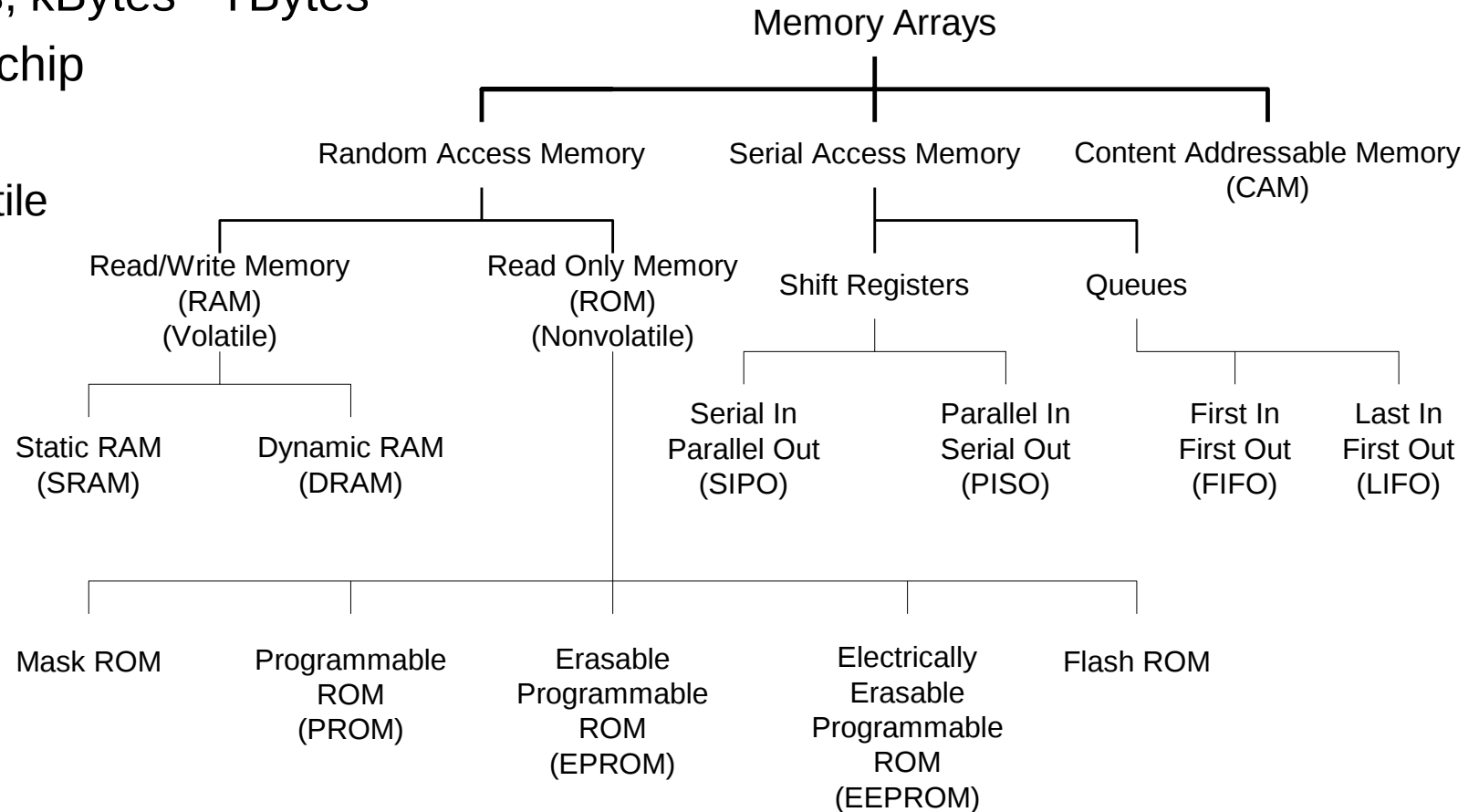
- **Timing Parameters:** read-, write-access, cycle time

- **I/O Architecture:**

- Single Port
- Multiport

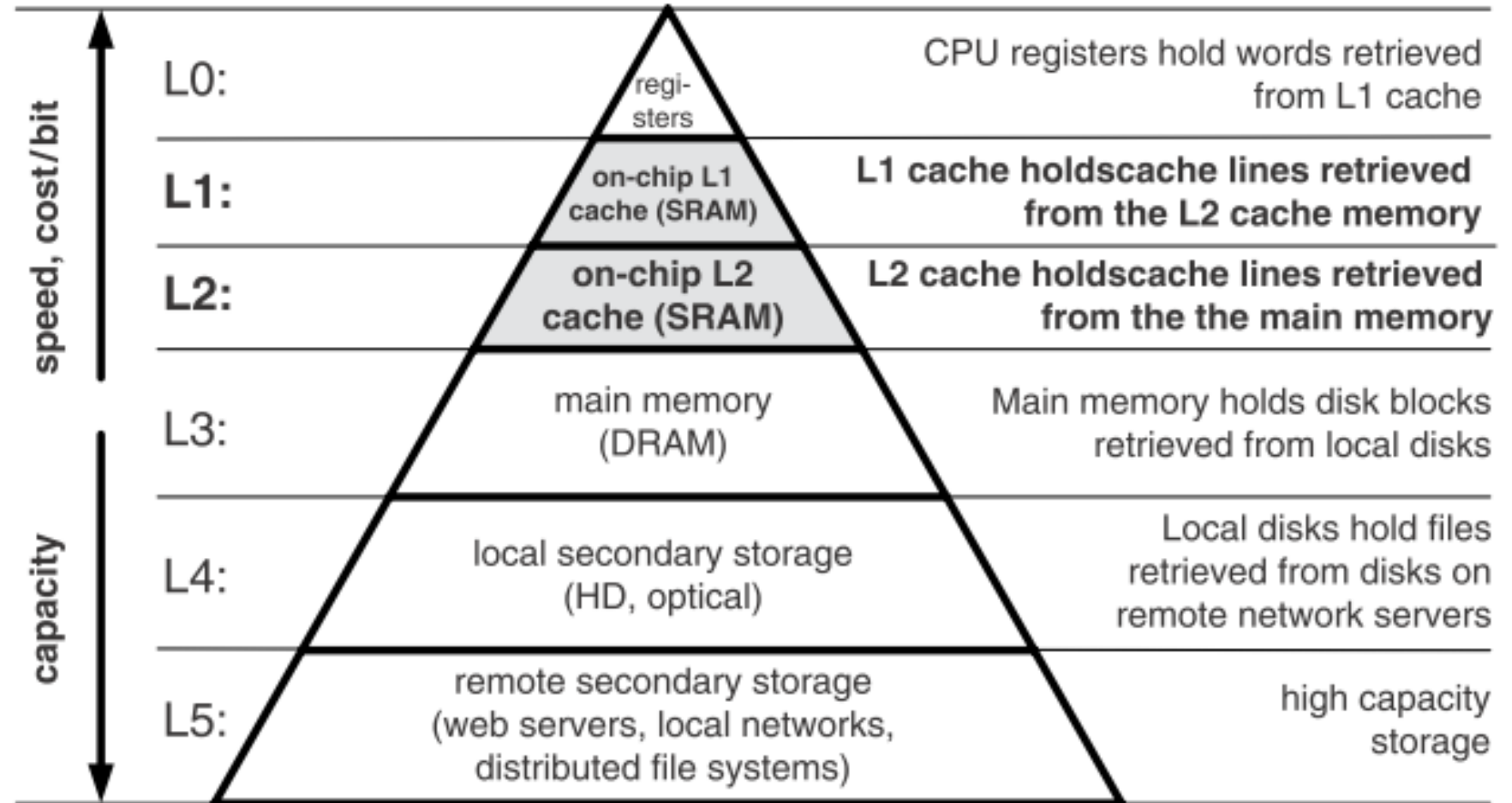
- **Access pattern/special fct.:**

- Random Access, FIFO, LIFO, Shift Register, CAM, IMC



Mixing and Matching in Memory Hierarchy

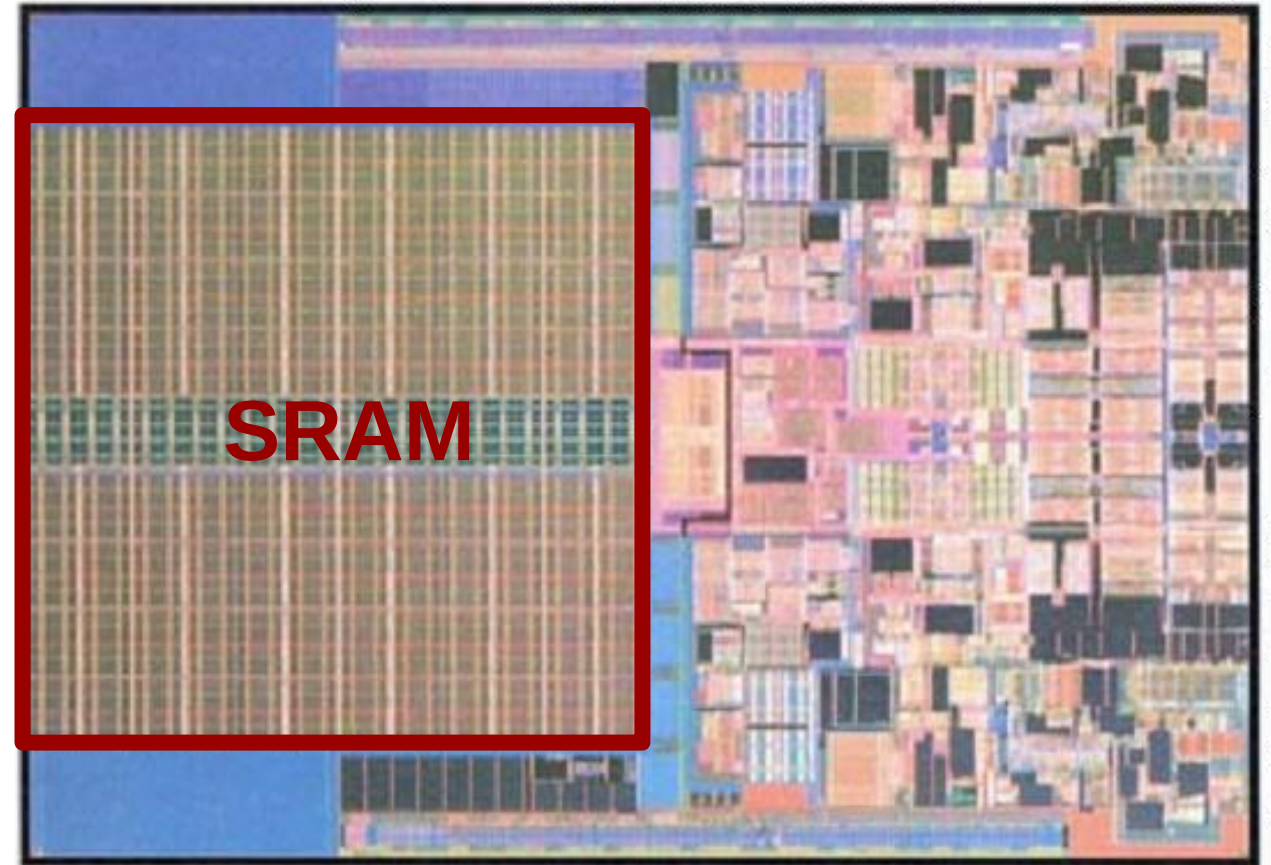
- Many systems have a mix of **conflicting requirements**: mostly **density & speed**
- memory hierarchy**:
Mix of different memory types



The Importance of SRAM in IC Design

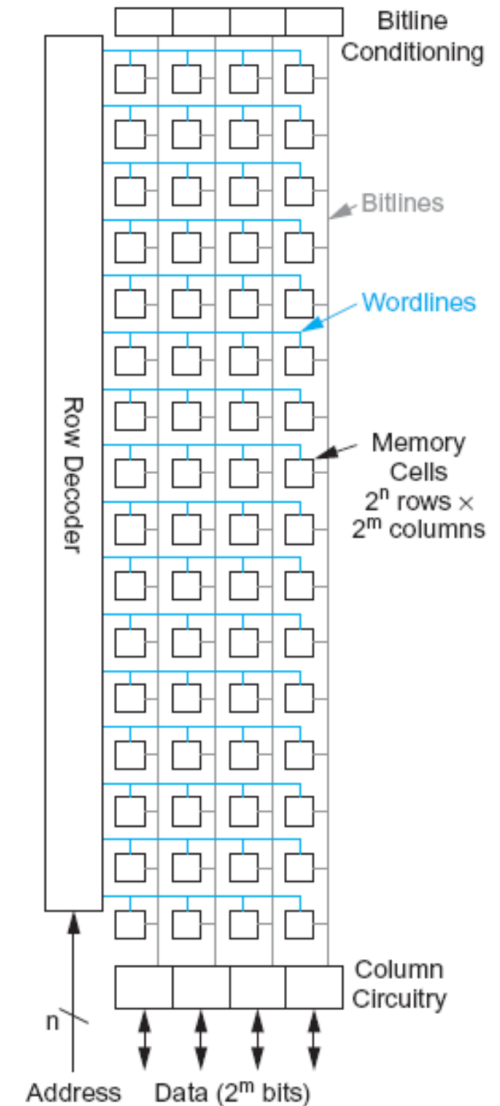
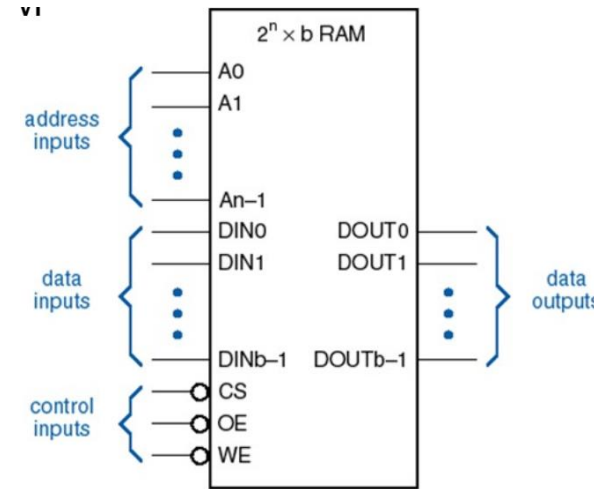
- **SRAM is the most common type of on-chip (embedded) memory**
 - Reasonably high density ~15x better than a standard FlipFlop!!
 - High speed ~1GHz in 28nm
 - Fully CMOS compatible (no special process steps required)
- **However, SRAMs often still occupy >50% of the chip area.**

Intel 45nm Core 2



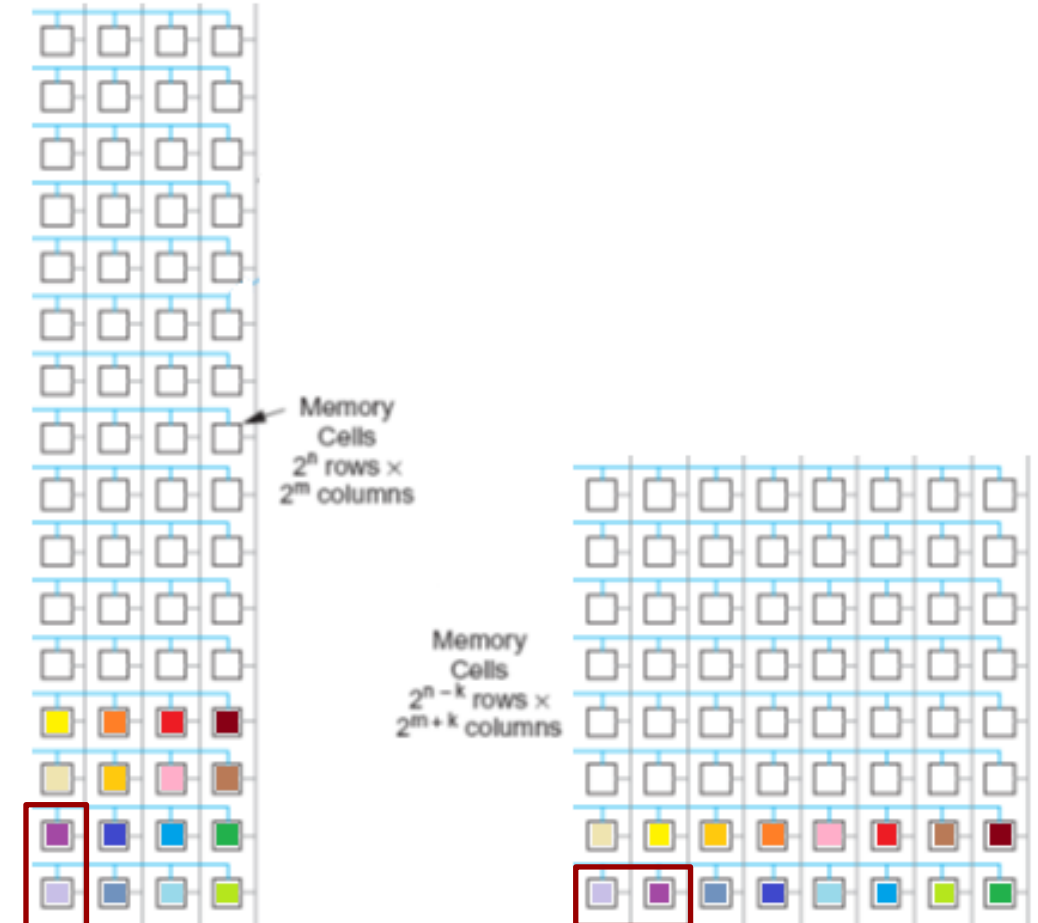
Logical View of a Memory

- **Logical organization** of a memory: 2^n words with 2^m bits/word
 - Total memory capacity: 2^{n+m} bits
- Each bit is stored in a bit-cell
- Memory macros organize bit-cells in a compact array
- Peripheral circuits provide access to one word at a time
 - n bit memory address selects the word to be accessed
 - All bits of a word are read/written at the same time
 - Most memories allow only either a read or a write at the same time (exception: two-port and dual-port memories)



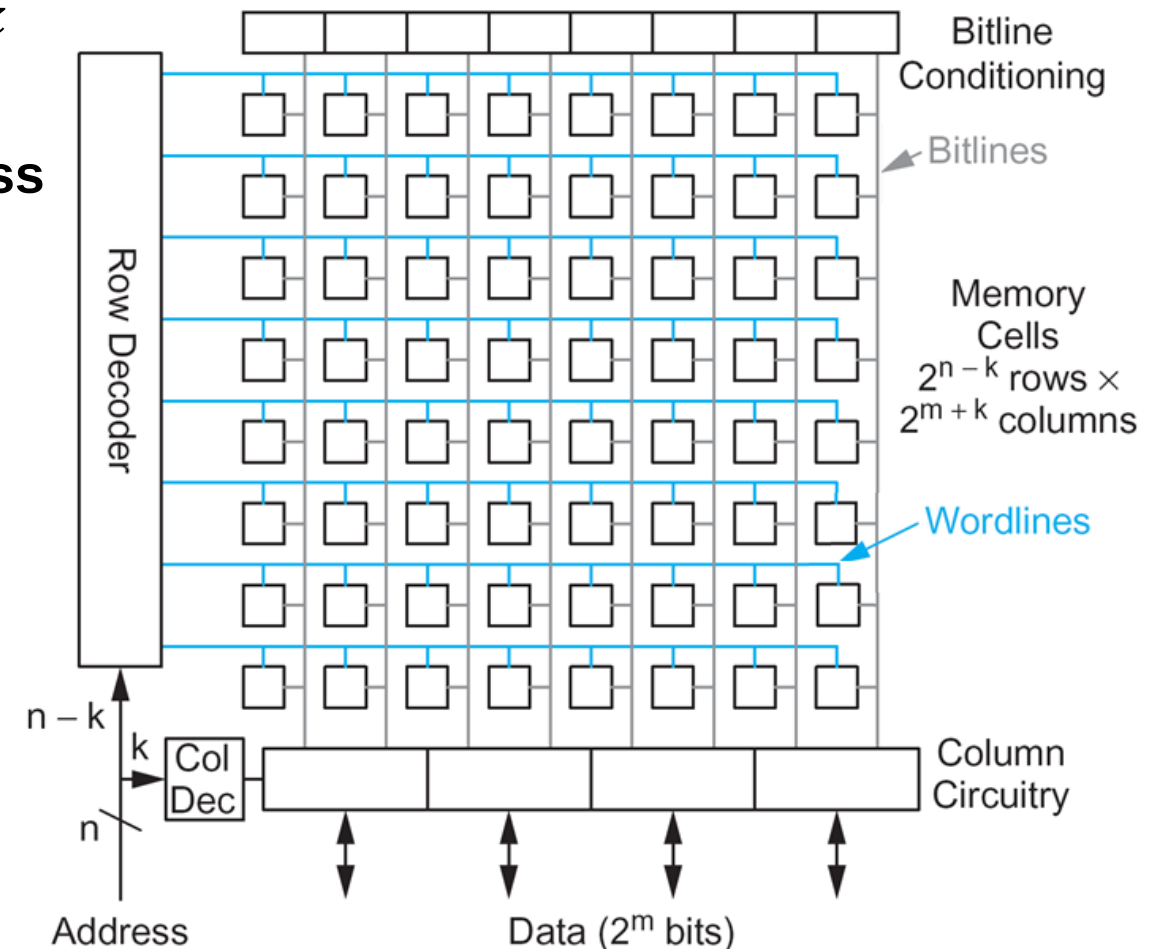
Physical Organization of the Bit-Cell Array

- For most memories, the **number of words 2^n is significantly larger than the number of bits/word 2^m** (e.g., 1024 words of 8 bit each)
 - Straightforward organization with 2^n rows and 2^m columns would be extremely tall
- Extremely tall memories cause issues**
 - Long bit lines connecting bit-cells to peripherals
 - Difficult to place on in a layout
 - Height may exceed the height of the chip
- Memory arrays are folded by 2^k to obtain an almost square shape**
 - Physical array: 2^{n-k} rows and 2^{m+k} columns
 - Bits of adjacent words are interleaved**



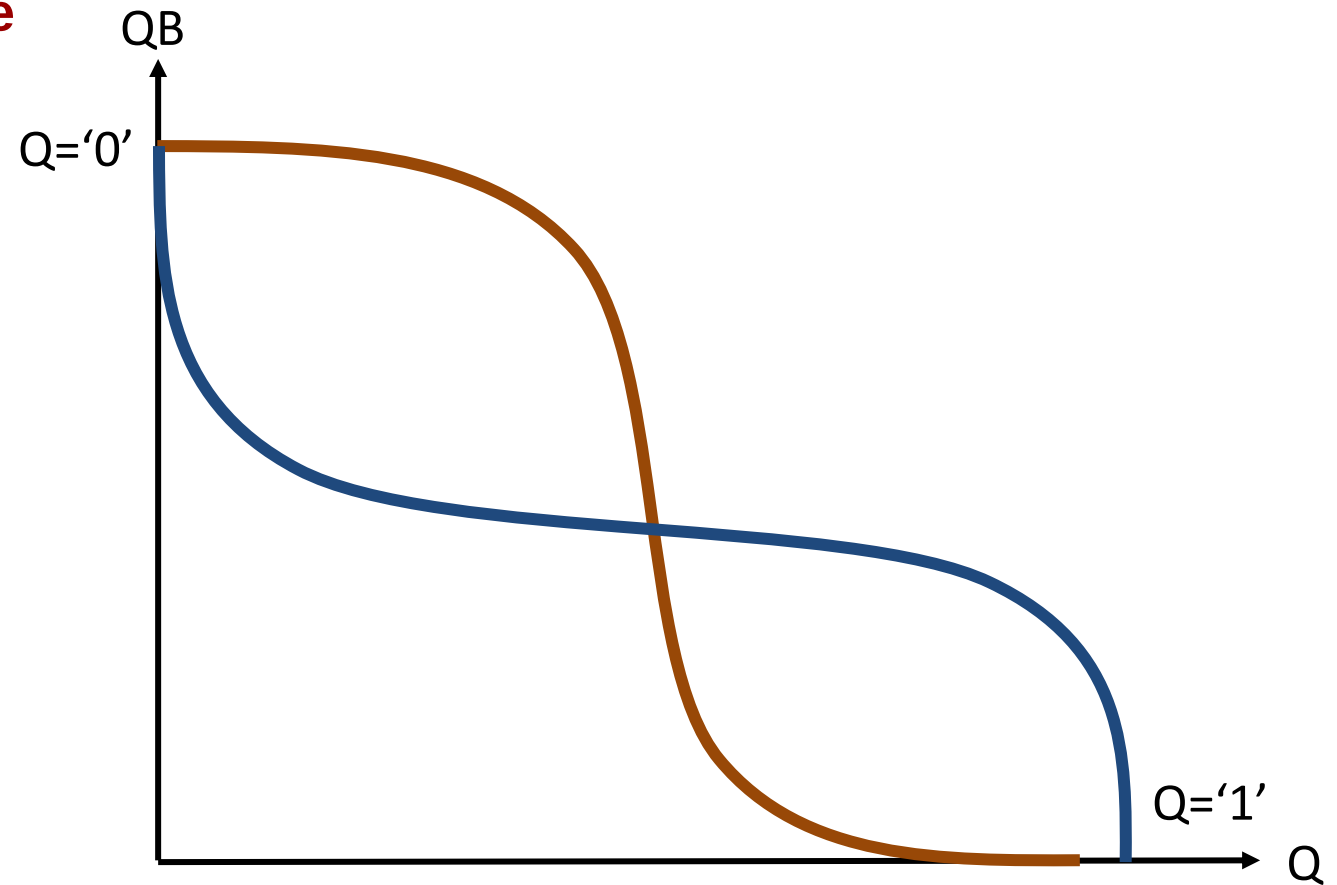
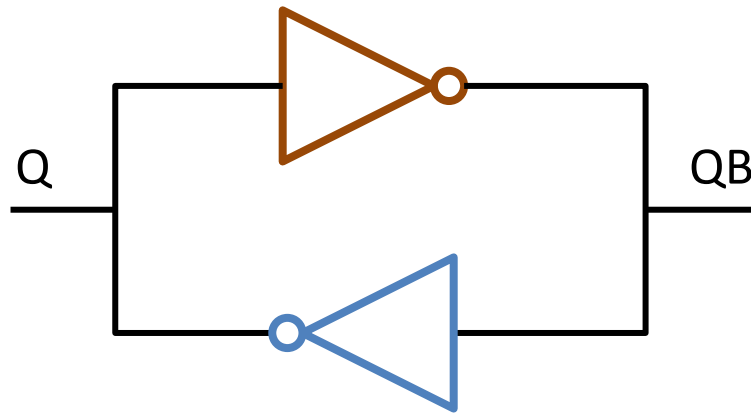
Array Access and Basic Peripherals

- Consider the **folded array** with its peripherals
 - Due to the folding the address is split into a $n - k$ bit row-address and a k bit column address
 - Row decoder decodes** the $n - k$ bit **row address** into 2^{n-k} one-hot encoded row-select signals
 - Only the selected row of bits is activated
 - Row-select signals are distributed through **horizontal word-lines**
 - Vertical bit lines** route the data of 2^k interleaved words from (read) and to (write) the bit cells
 - Column muxes** select the bits of the selected word based on the k bit column address
 - Interleaving reduces routing to the mux**



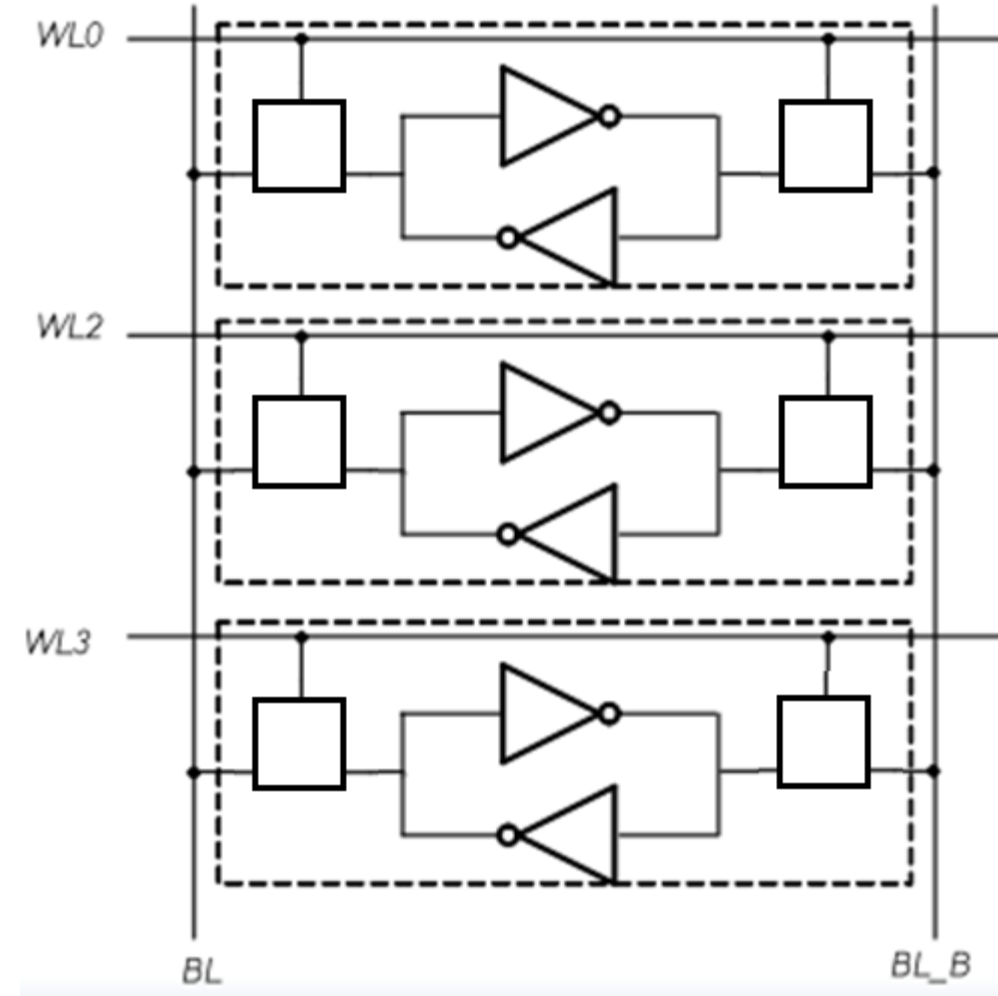
The 6T-SRAM Bit Cell

- **6T-SRAM bit cell: based on active feedback of two cross-coupled inverters**
 - Analysis by overlaying the VTCs of the two inverters: **butterfly curve**
 - Positive feedback and high gain correct disturbances from e.g., leakage, noise, or coupling



Accessing Bit Cells for Read and Write

- Bits in a column are connected to the one bit-line
- Read and write access by connecting them to the bit-line through access transistors
 - **During write access:** bit-line content is forced into the cross-coupled inverter pair
 - **During read access:** cross-coupled inverter pair forces its potential onto the connected bit-line
- Important considerations:
 - Minimize number of transistors per bit-cell for access
 - Bit-lines are long and have a high capacitance
 - Bit-cell content is mirrored i.e., Q and QB available

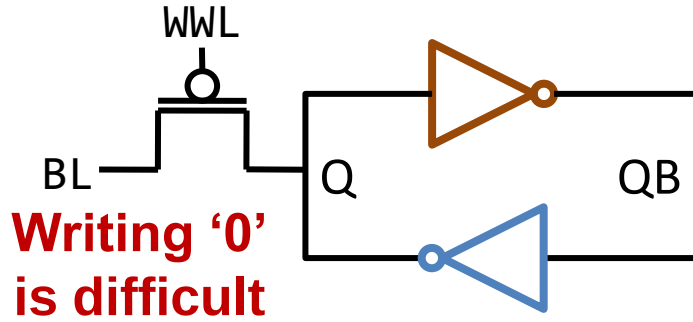


Options for Bit-Cell Access

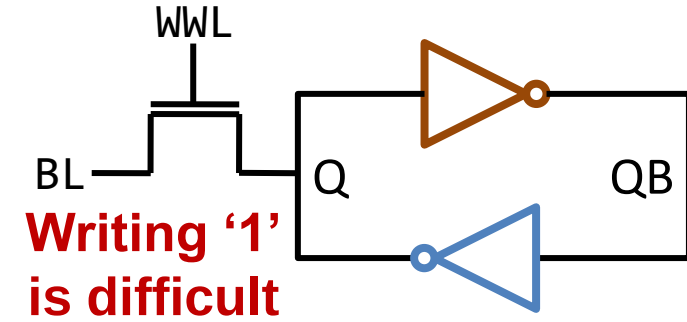
- **Write access:** write circuit needs to over-power the bit-cell feedback
- **Read access:** protect the cell content from accidentally flipping (read-disturb)

Option 1a: pMOS Access Transistor

Single access transistor



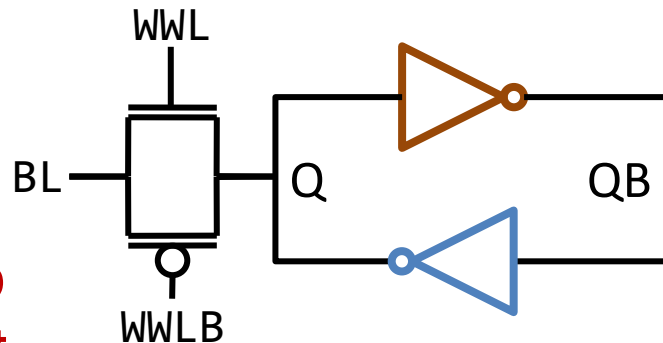
Option 1b: nMOS Access Transistor



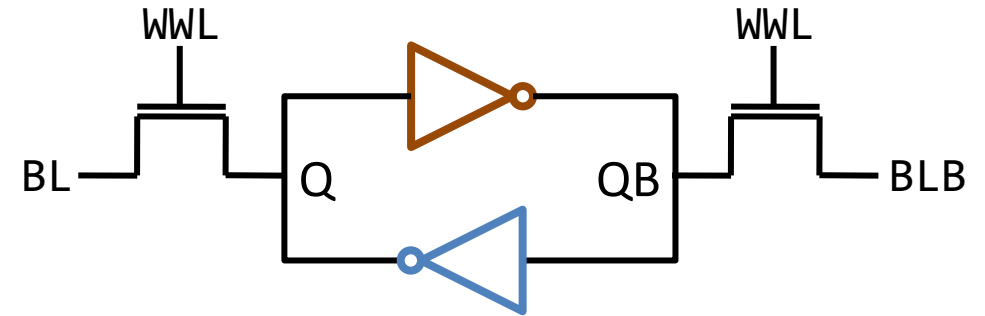
Option 2: Transmission Gate

Two access transistors

Reading without flipping Q is difficult

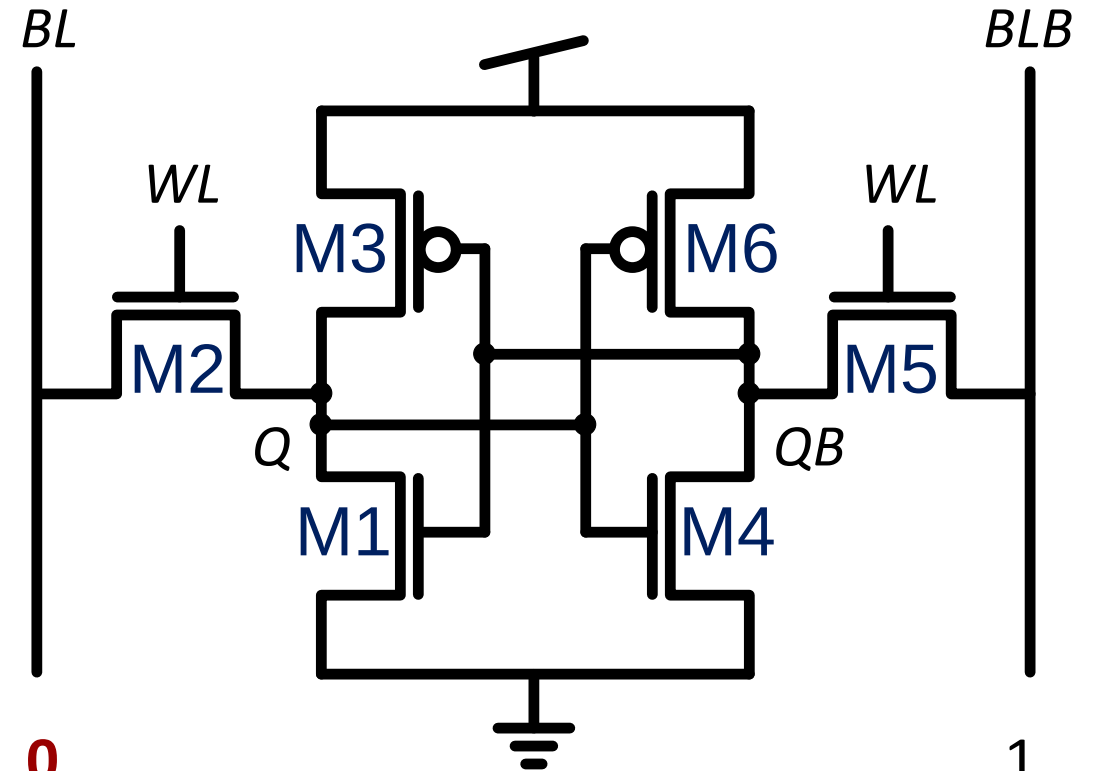


Solution: Two-Sided nMOS Write



6-Transistor CMOS SRAM Cell Operation

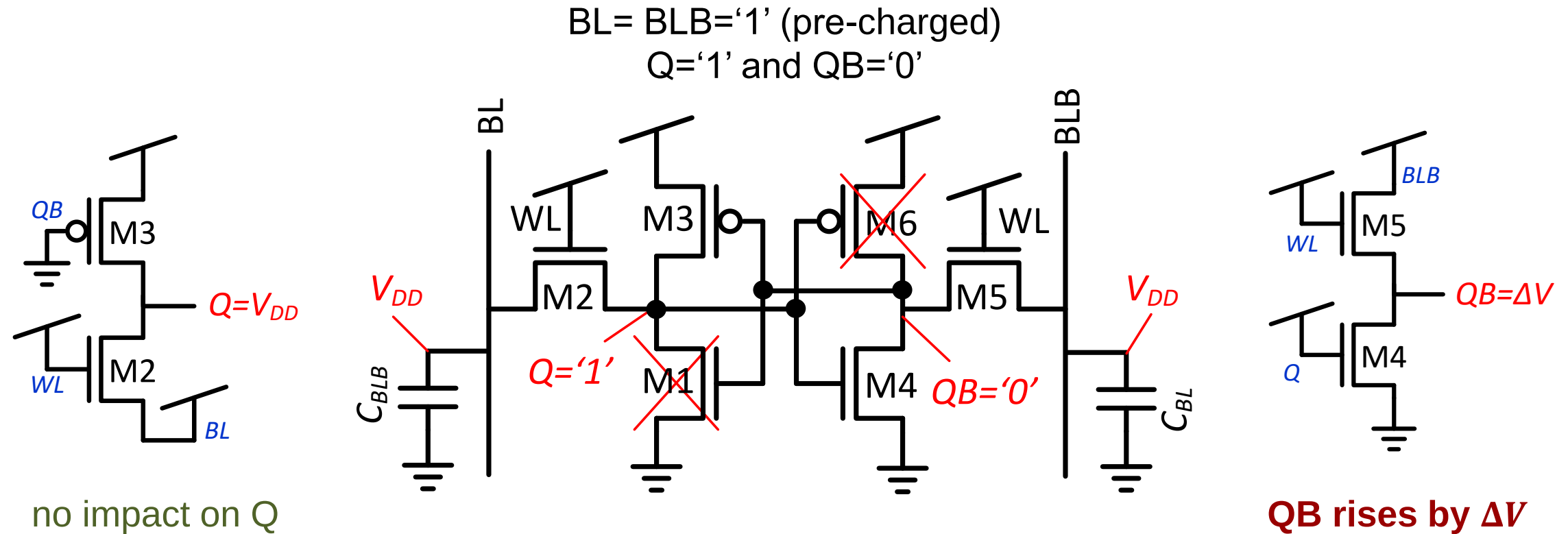
- **Write '0' operation:** $BL = '0', BLB = '1'$
 - M2 passes a strong '0' to Q
 - M5 leaves QB unaffected (nMOS passes a weak '1')
 - **Write a '0' to Q**
- **Write '1' operation:** $BL = '1', BLB = '0'$
 - M2 leaves Q unaffected (nMOS passes a weak '1')
 - **M5 passes a strong '0' to QB**
 - **Write a '0' to QB**
- **Read operation:**
 $BL = '1', BLB = '1'$
 - M2 and M5 leave Q and QB unaffected (both pass a weak '1')



Write '0'	0	1
Write '1'	1	0
READ	1	1

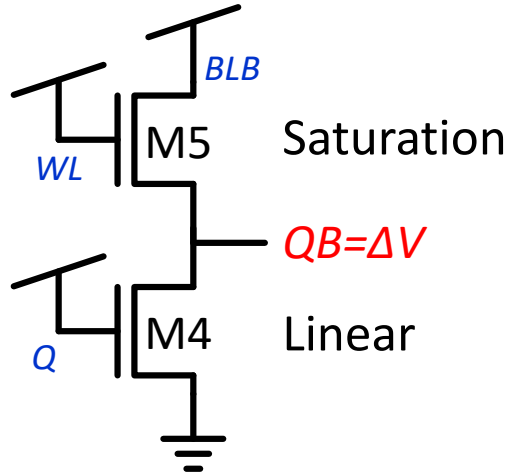
SRAM Operation Analysis: READ

- **Read** operation **should NOT** impact the **state**: Consider the change in the voltage on Q and QB during read (should be small to NOT flip the state)



SRAM Operation Optimization: READ

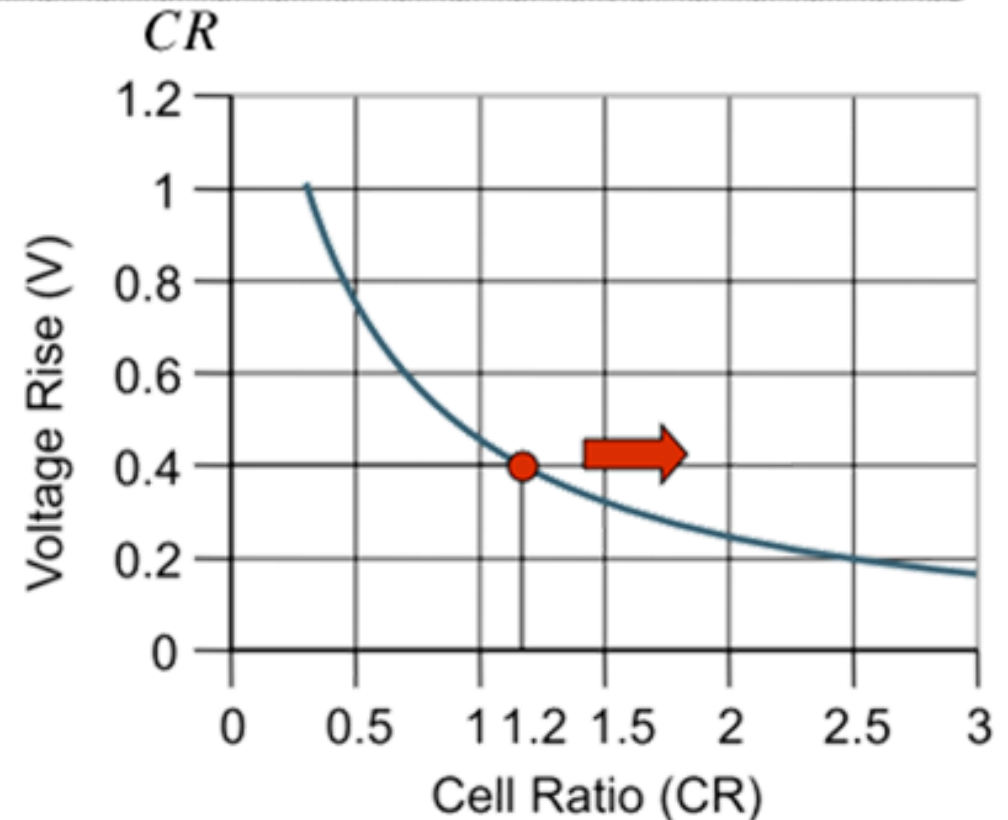
- Analysing ΔV yields: $k_{n,M5} \left((V_{DD} - \Delta V - V_{Tn}) V_{DSATn} - \frac{V_{DSATn}^2}{2} \right) = k_{n,M1} \left((V_{DD} - V_{Tn}) \Delta V - \frac{\Delta V^2}{2} \right)$



$$\Delta V = \frac{V_{DSATn} + CR(V_{DD} - V_{Tn}) - \sqrt{V_{DSATn}^2(1 + CR) + CR^2(V_{DD} - V_{Tn})^2}}{CR}$$

$$CR = \frac{W_4/L_4}{W_5/L_5}$$

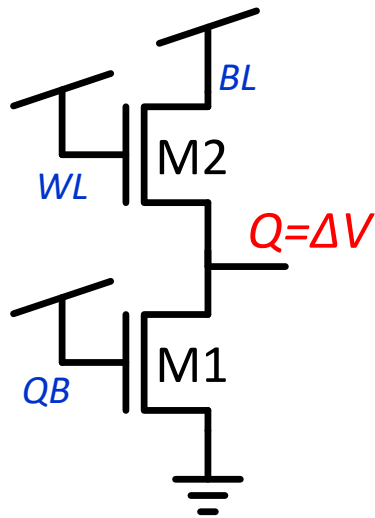
- Impact on QB depends on drive strength ratio of M4 and M5
- To keep ΔV low: choose a sufficiently strong pull-down keeper (not so difficult since competing nMOS can not drive a strong '1')



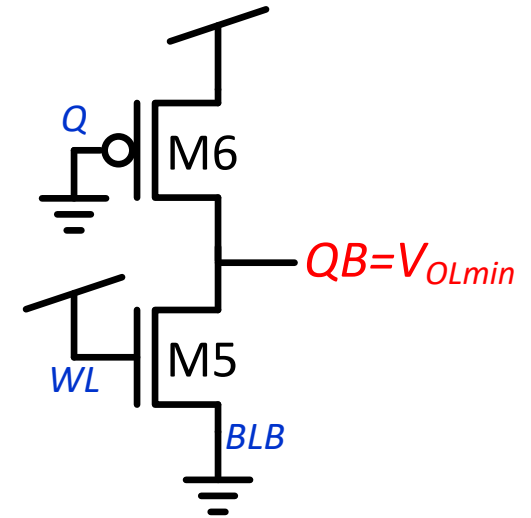
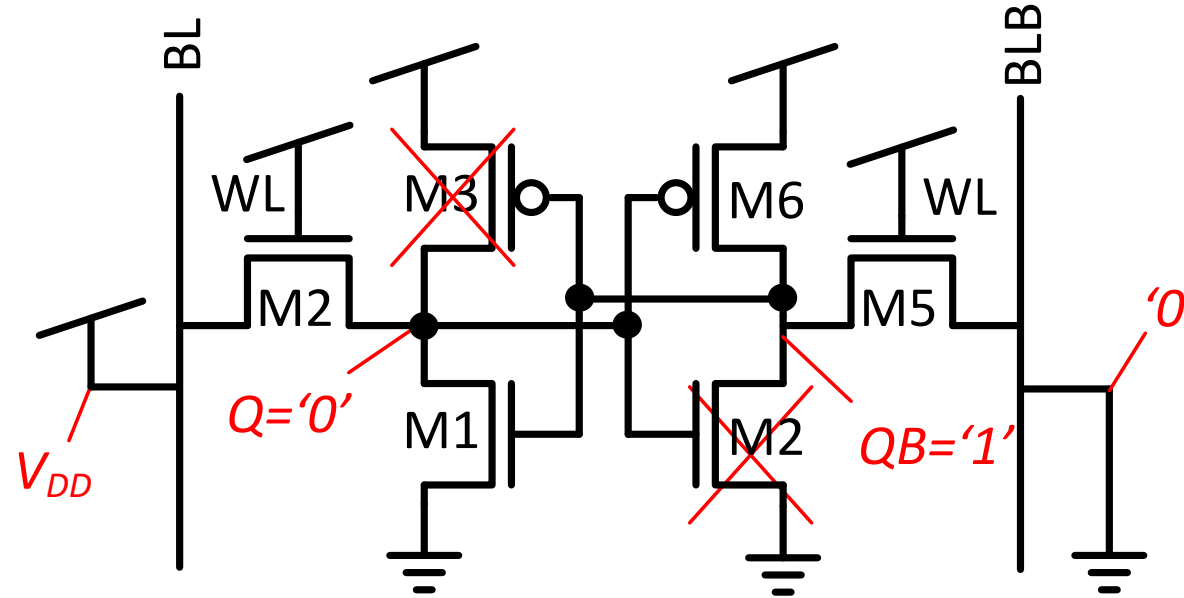
SRAM Operation Analysis: WRITE

- Write** operation **should** impact the **state**: Consider the change in the voltage on Q and QB during write (should be sufficient to flip the state)

BL='1', BLB='0' (driven)
Q='0' and QB='1'



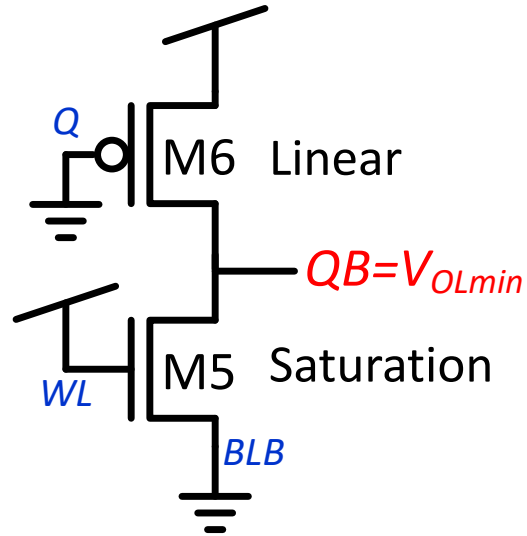
Same as for READ:
Impact on Q is small
by design



**Design to lower QB
below switching
threshold $V_{OLmin} < V_M$**

SRAM Operation Optimization: WRITE

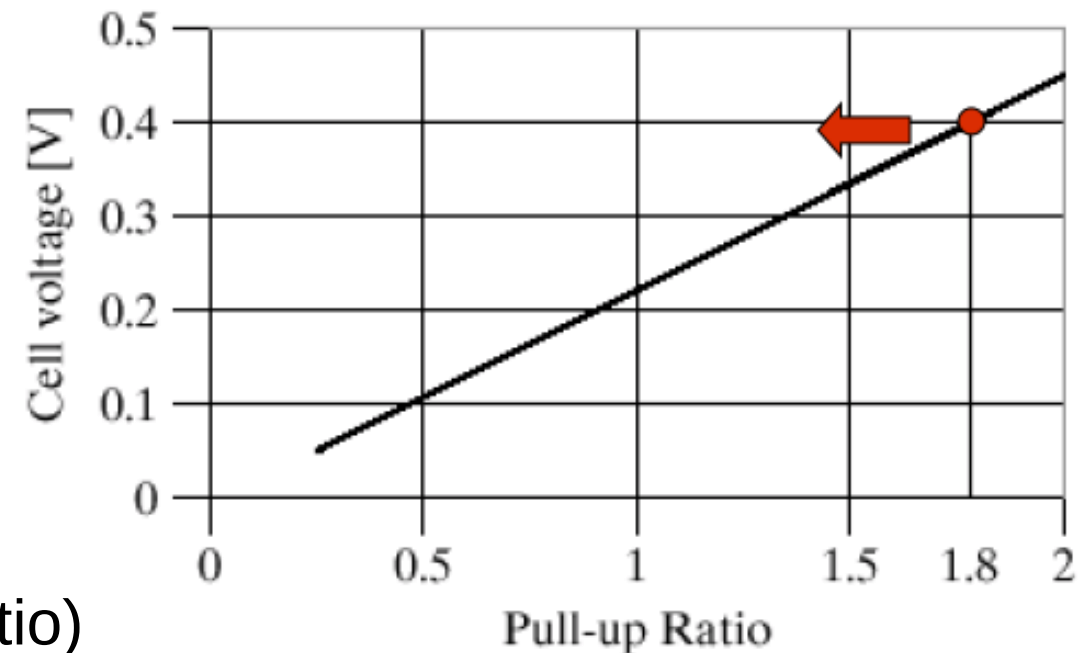
- Analysing V_{OLmin} : $k_{n,M6} \left((V_{DD} - V_{Tn}) V_Q - \frac{V_Q^2}{2} \right) = k_{p,M4} \left((V_{DD} - |V_{Tp}|) V_{DSATp} - \frac{V_{DSATp}^2}{2} \right)$



$$V_Q = V_{DD} - V_{Tn} - \sqrt{(V_{DD} - V_{Tn})^2 - 2 \frac{\mu_p}{\mu_n} PR \left((V_{DD} - |V_{Tp}|) V_{DSATp} - \frac{V_{DSATp}^2}{2} \right)}$$

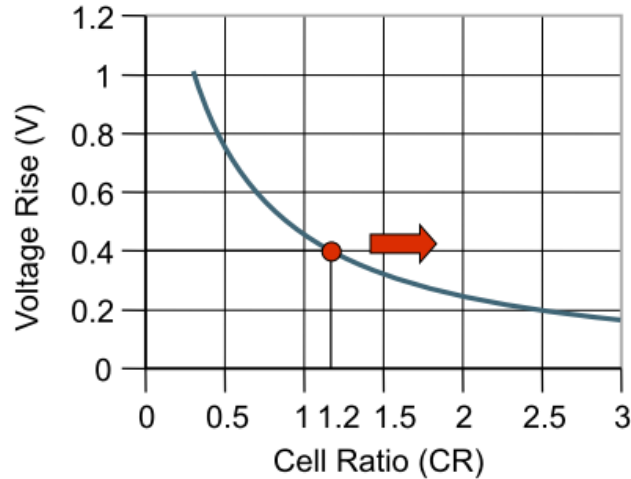
$$PR = \frac{W_6}{L_6} / \frac{W_5}{L_5}$$

- Impact on QB depends on drive strength ratio of M6 and M5 (pull-up ratio)
- To reach a sufficiently low V_{OLmin} : choose a sufficiently **strong access transistor M5** or a **weak pull-up transistor** (low pull-up ratio)

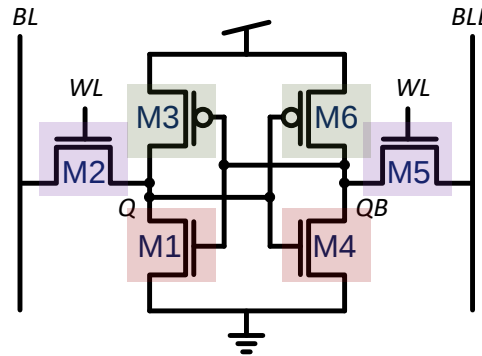


Summary – SRAM Sizing Constraints

Read Constraint

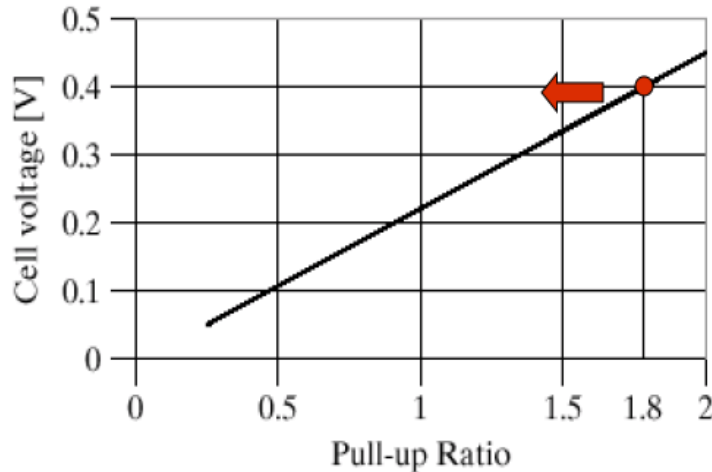


$$CR \equiv \frac{\frac{W_1}{L_1}}{\frac{W_2}{L_2}} = \frac{\frac{W_4}{L_4}}{\frac{W_5}{L_5}} = \frac{\text{Pull-Down}}{\text{Access}} \xrightarrow[\text{(keep during read)}]{\text{Strong '0'}} K_{\text{Pull-Down}} > K_{\text{Access}}$$



$$K_{\text{Pull-Down}} > K_{\text{Access}} > K_{\text{Pull-Up}}$$

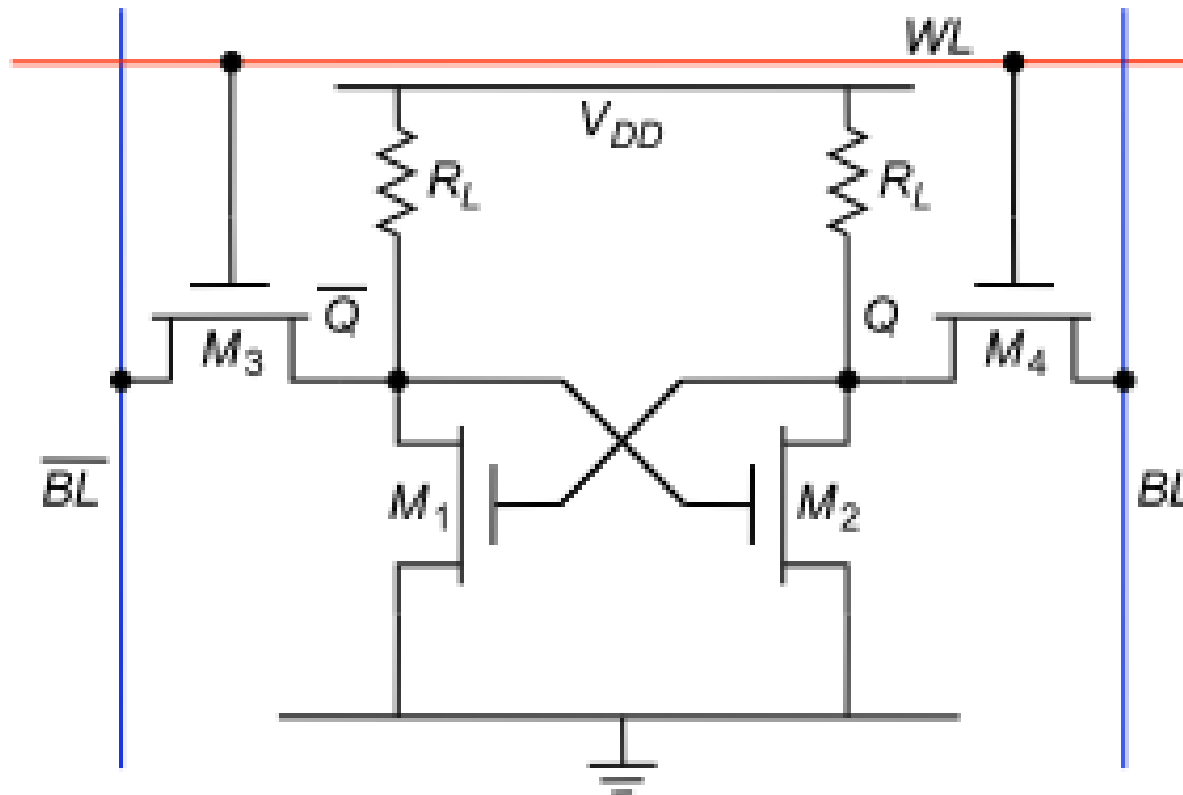
Write Constraint



$$PR \equiv \frac{\frac{W_3}{L_3}}{\frac{W_2}{L_2}} = \frac{\frac{W_6}{L_6}}{\frac{W_5}{L_5}} = \frac{\text{Pull-Up}}{\text{Access}} \xrightarrow[\text{(flip during write)}]{\text{Weak '1'}} K_{\text{Access}} > K_{\text{Pull-Up}}$$

4T Memory Cell

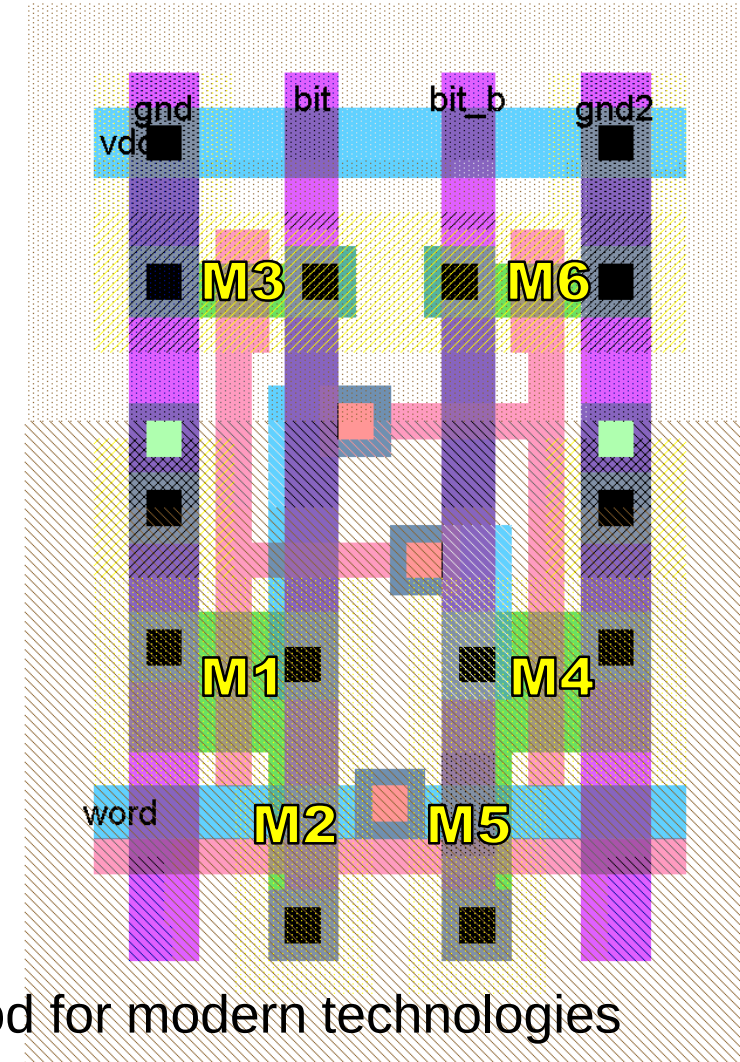
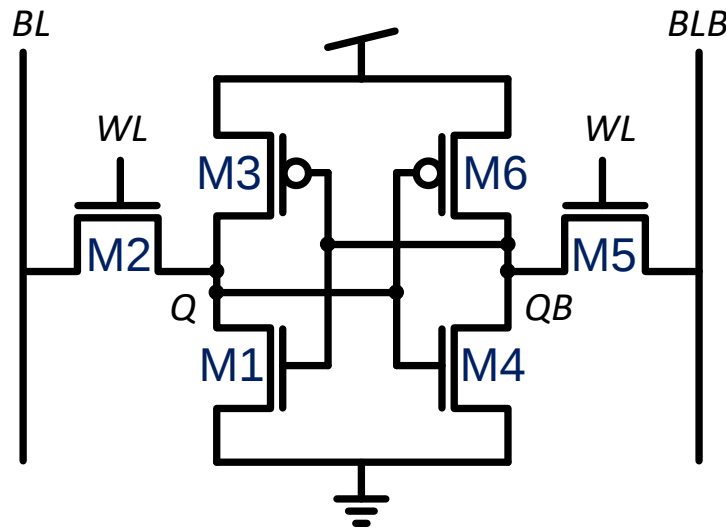
- **Achieve density by removing the PMOS pull-up**
 - Only used as a keeper (i.e., can be weak)
- **However, this results in static power dissipation.**



SRAM Layout - Traditional

- **Two sides of the bitcell symmetric**

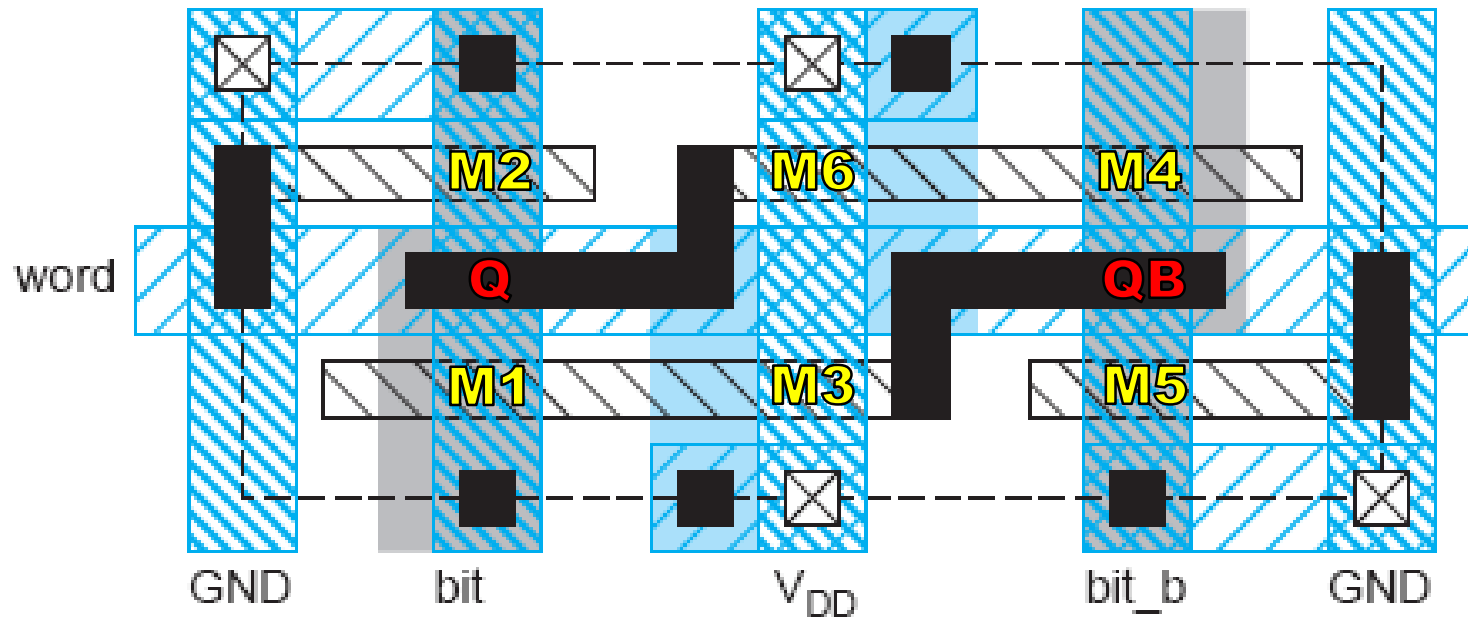
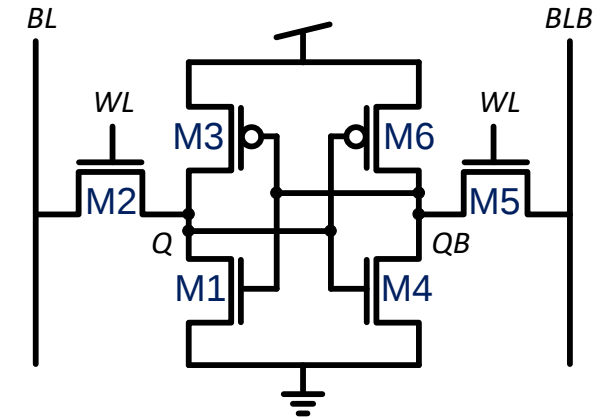
- Share horizontal routing (WWL).
- Share vertical routing (BL, BLB).
- Share power and ground.
- Word line routed double on Poly and Metal (reduce resistance)



- **Uses both horizontal and vertical poly:** not good for modern technologies

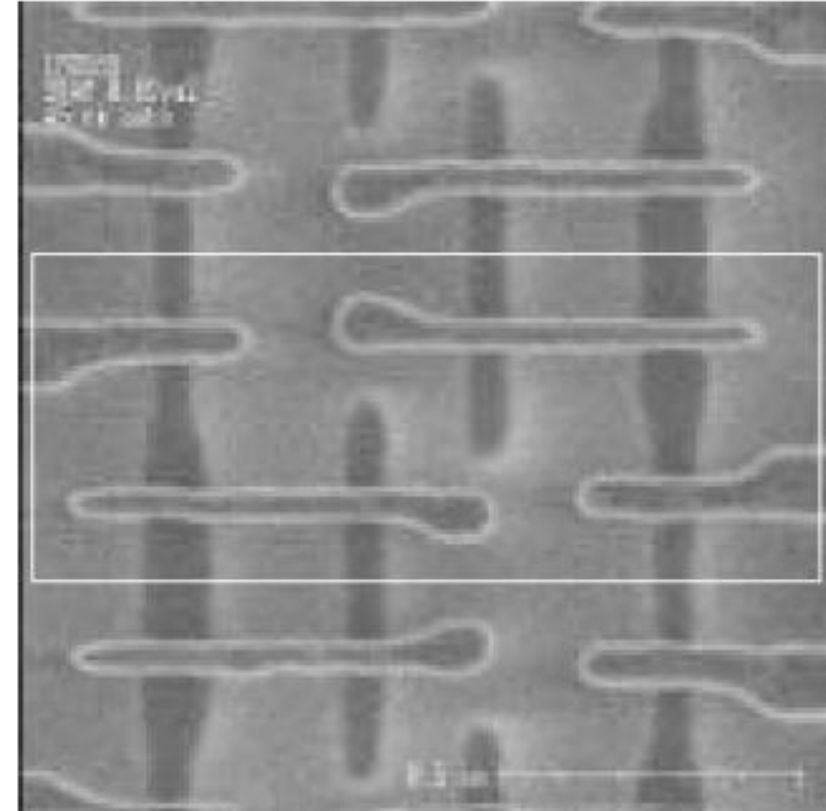
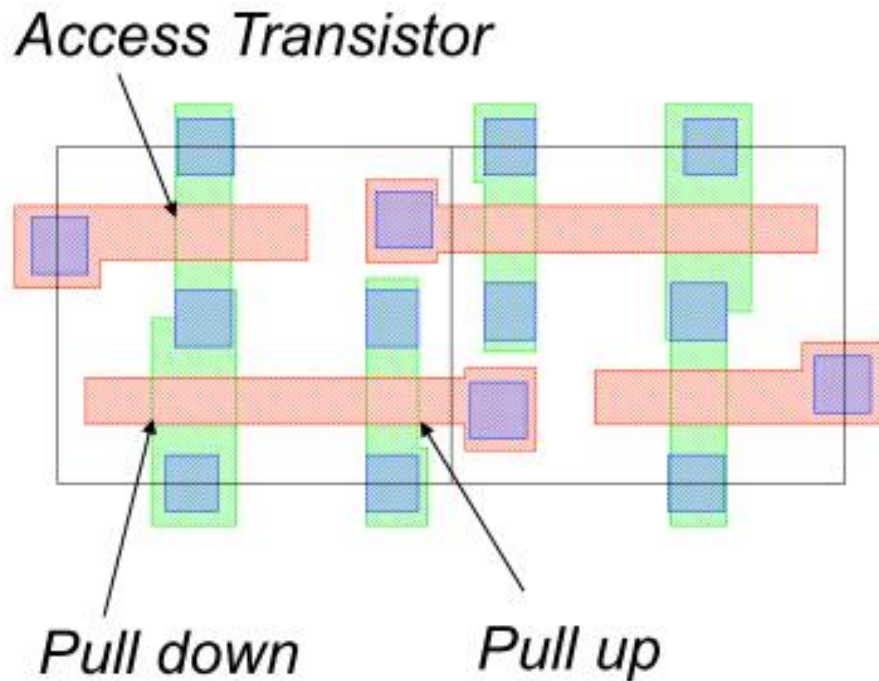
SRAM Layout – Thin Cell

- **Thin:** minimize bitline capacitance (length)
 - Avoid bends in polysilicon and diffusion (easier for lithography)
 - Orient all transistors in one direction.
 - **Metal word line:** reduced resistance



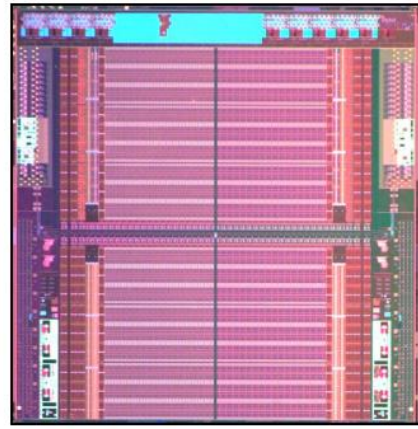
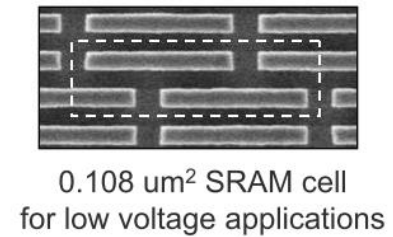
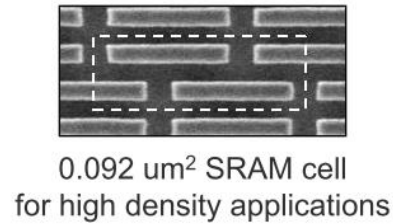
65nm SRAM

□ ST/Philips/Motorola

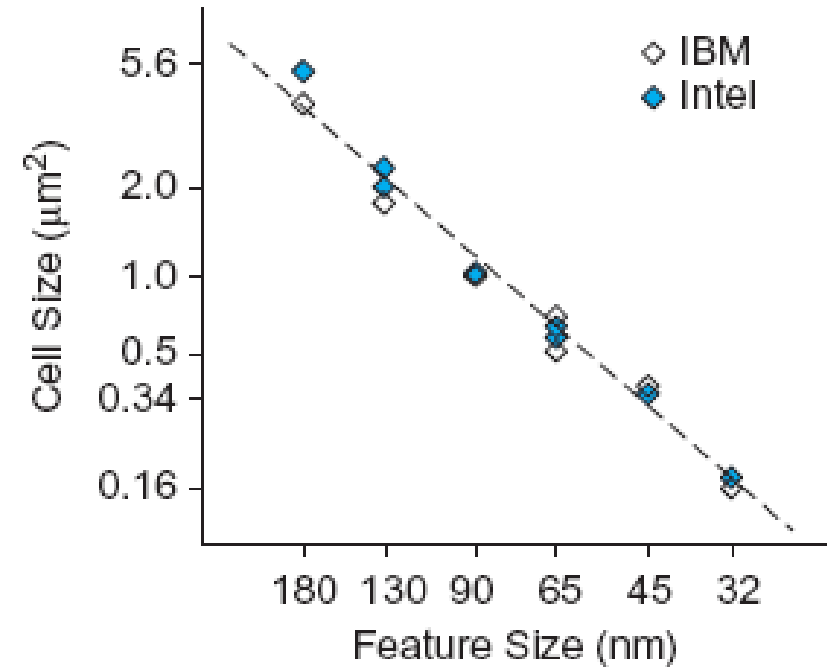


Sharing between neighbouring cells by flipping every other row/column
Bended design rules

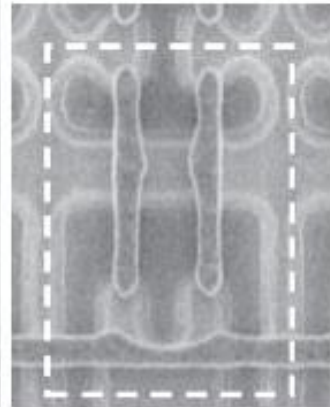
Commercial SRAMs as a Metric for Moore's Law



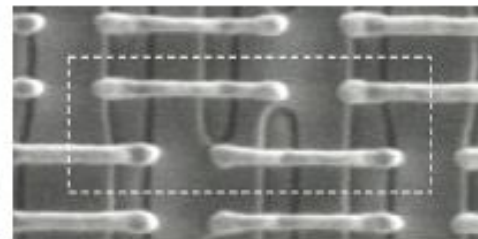
Intel Design Forum 2009



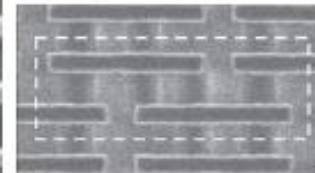
130 nm [Tyagi00]



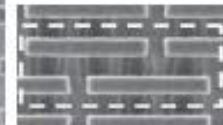
90 nm [Thompson02]



65 nm [Bai04]



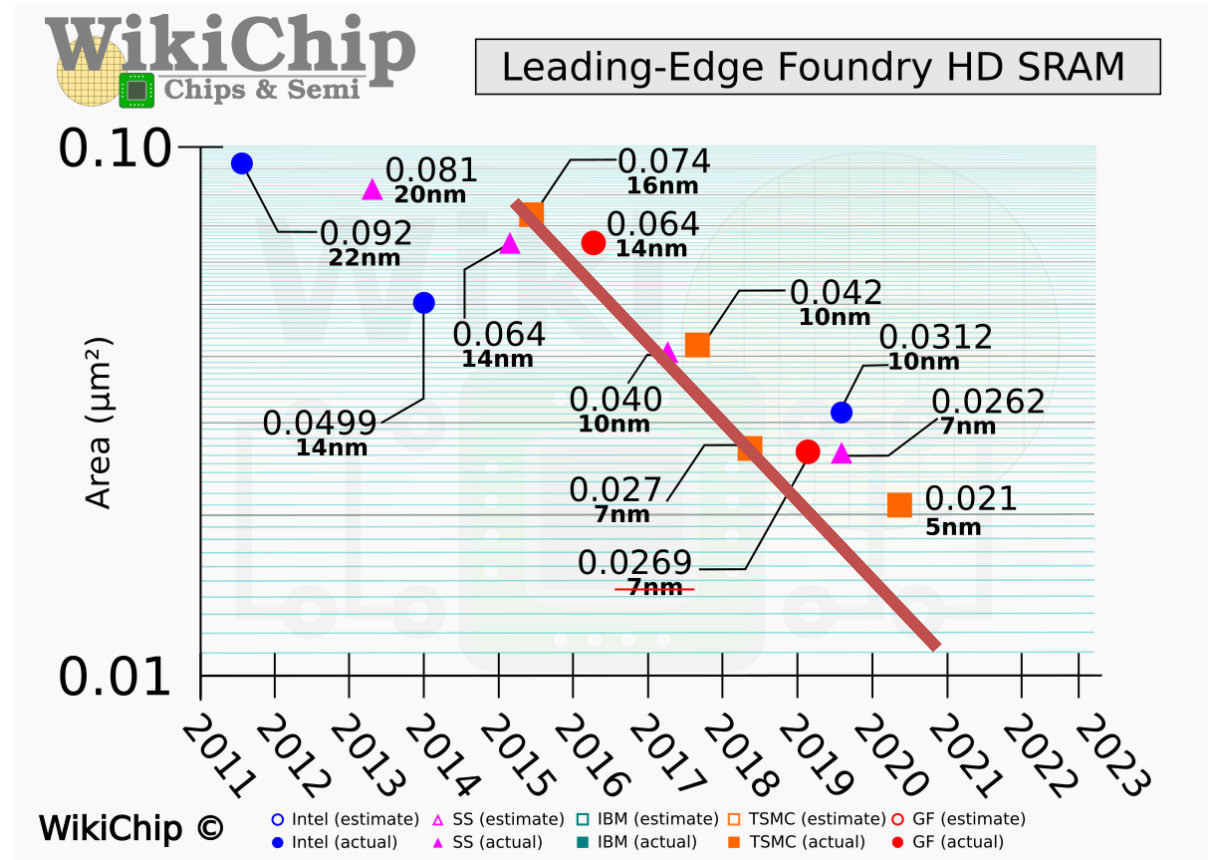
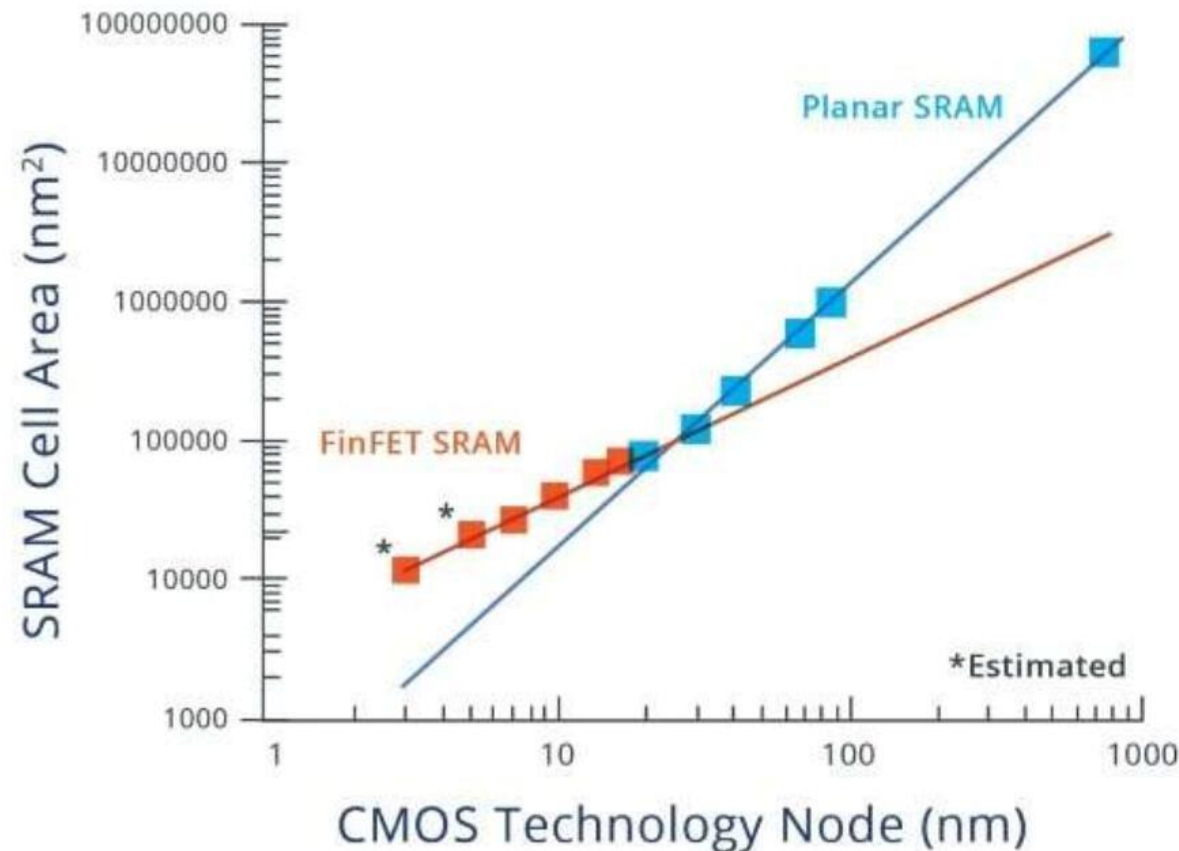
45 nm [Mistry07]



32 nm [Natarajan08]

SRAM and the End of Moore's Law

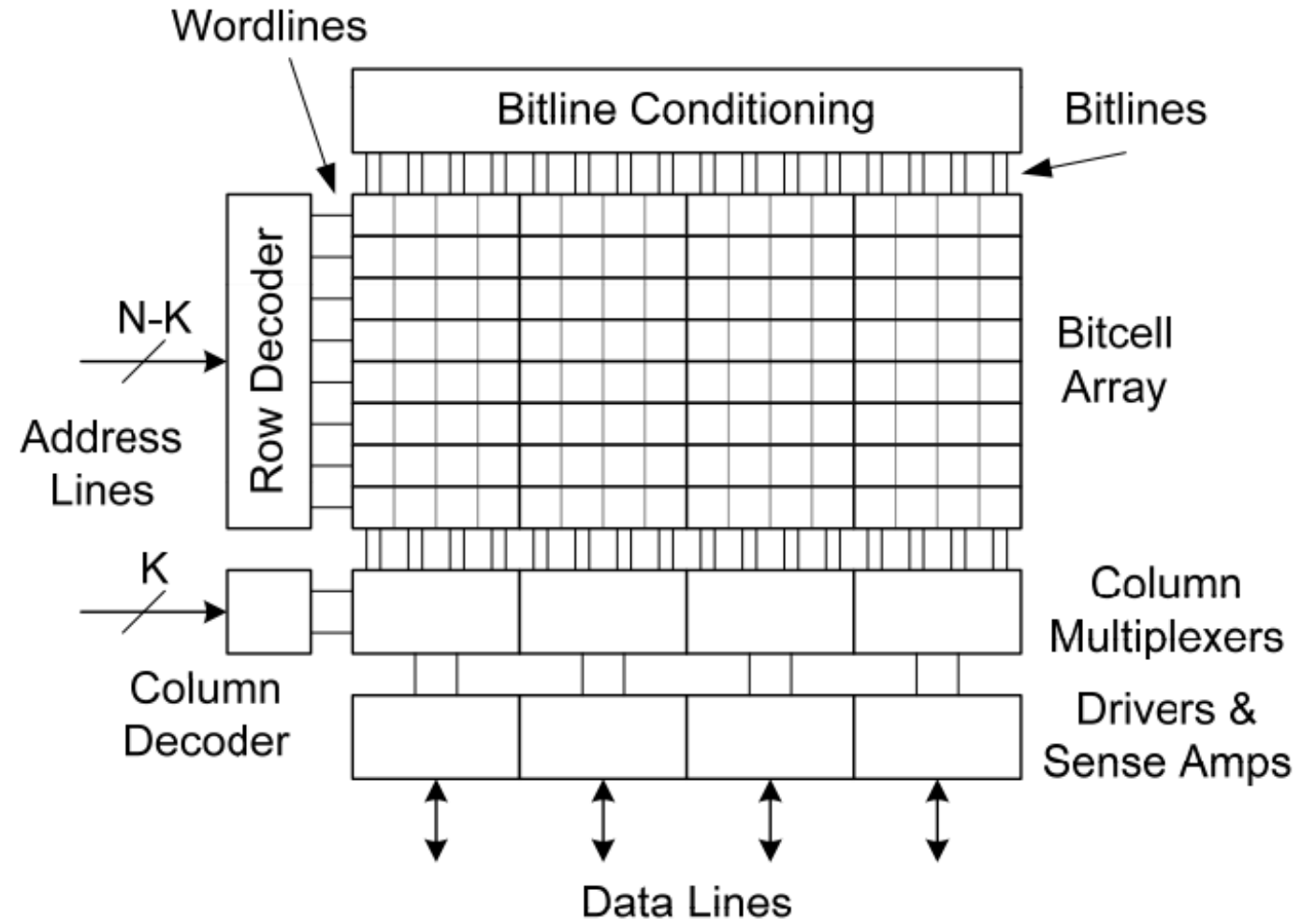
- Latest publications from TSMC confirm failure to track scaling with SRAM



SRAM Peripheral Circuits

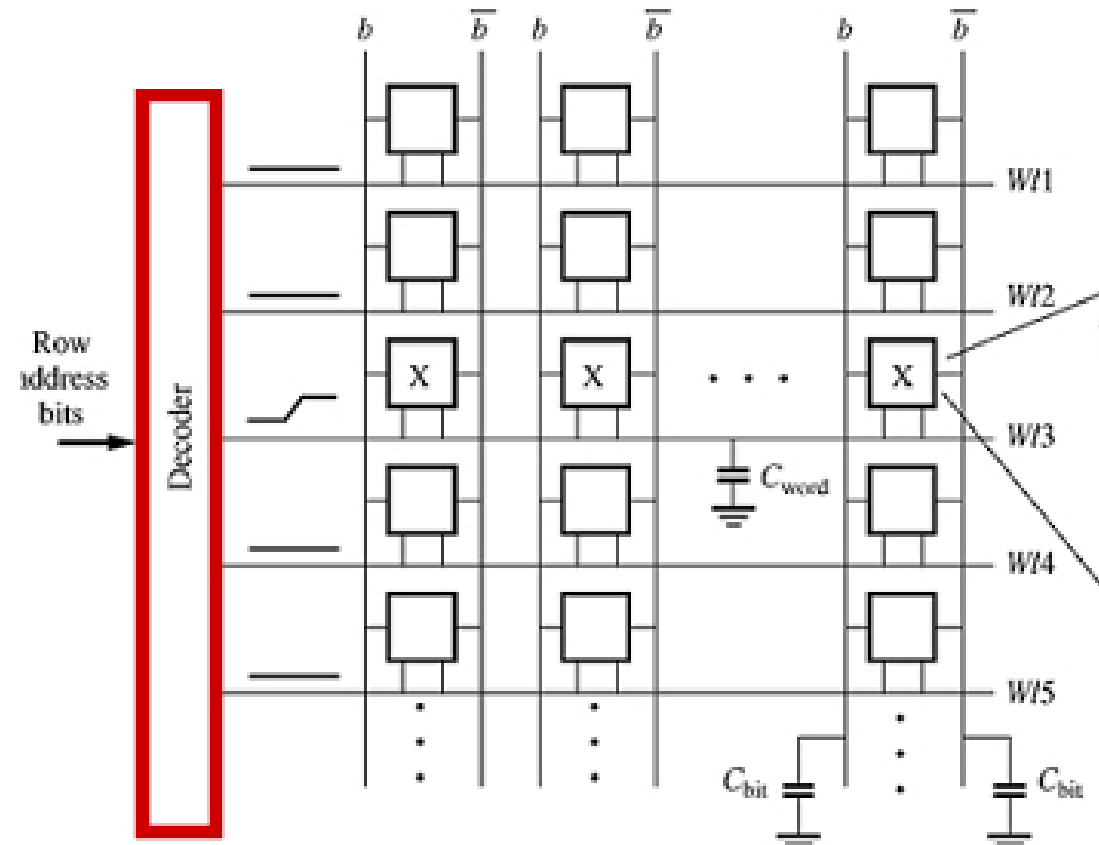
- **SRAM operation involves various peripherals** for read- and write-access

- Row decoder
- Precharge circuit
- Sense amplifier
- Column multiplexer
- Write driver



Row Decoders

- **Row decoder: activate the word-line of the selected row** during read and write
 - One hot decoder that decodes $n - k$ address bits into 2^{n-k} word lines
- **Important considerations**
 - Two word-lines should never be active at the same time (not even for a short period)
 - Word lines present a considerable load:
 - Parasitic wire capacitance
 - Fan-out for all 2^{m+k} bit cells in a row
 - Similar access timing should be ensured across the entire array
 - Area overhead must be minimized



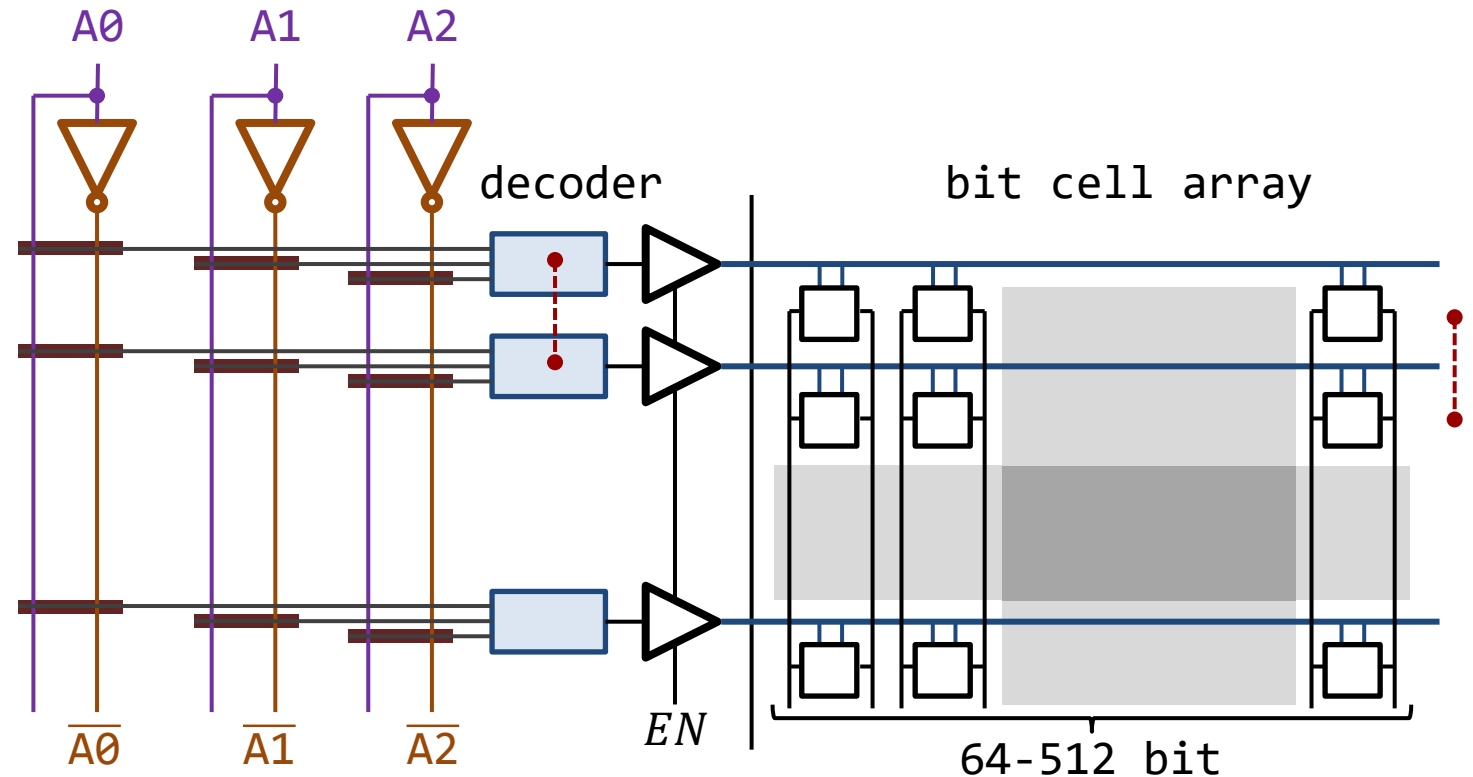
Folding: k times

Words: 2^n

Bits/word 2^m

Row Decoder: Single Stage Implementation

- **One-hot row-decoder** can be implemented **in a single stage**:
 - Independent row circuit (single stage) for each word-line
 - Each row circuit receives a unique combination of either the **true** or the **complemented** version of each address bit (i.e., $n - k$ bits)
- **Important considerations**:
 - Row circuits need to be **pitch matched to bit cell high (pitch)**
 - **Each row circuit has $n - k$ inputs** and a single output
- Row decoder is often followed by a driver to match the word line load

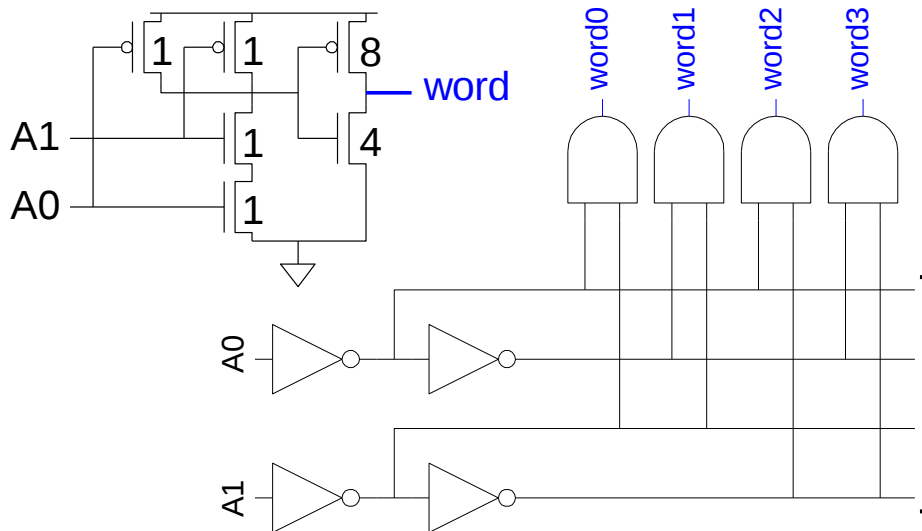


Row Decoder: Single Stage Implementation

- Two options to implement an active high row decoder (+driver)

NAND + INV

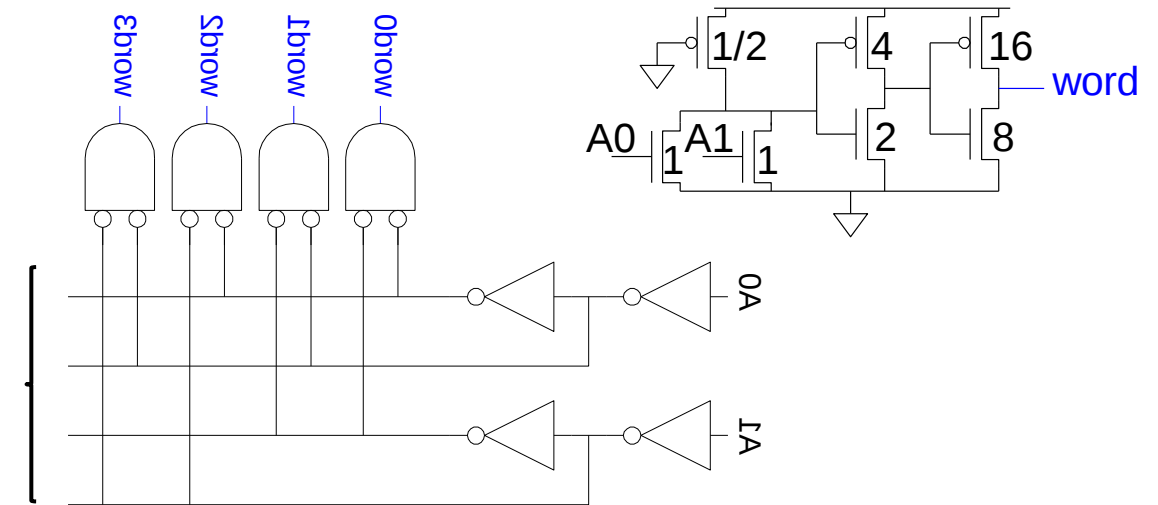
$$\left. \begin{aligned} WL_0 &= \bar{A}_7 \bar{A}_6 \bar{A}_5 \bar{A}_4 \bar{A}_3 \bar{A}_2 \bar{A}_1 \bar{A}_0 \\ &\vdots \\ WL_{255} &= A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0 \end{aligned} \right\} \begin{array}{l} \text{row circuit} \\ \text{fan-in : } n - k \end{array}$$



address line
load: 2^{n-k-1}

NOR + BUF

$$\left\{ \begin{aligned} WL_0 &= \overline{A_7 + A_6 + A_5 + A_4 + A_3 + A_2 + A_1 + A_0} \\ &\vdots \\ WL_{255} &= \overline{\bar{A}_7 + \bar{A}_6 + \bar{A}_5 + \bar{A}_4 + \bar{A}_3 + \bar{A}_2 + \bar{A}_1 + \bar{A}_0} \end{aligned} \right.$$



Limitations of the Single Stage Decoder

- **Single stage decoder is simple and regular**

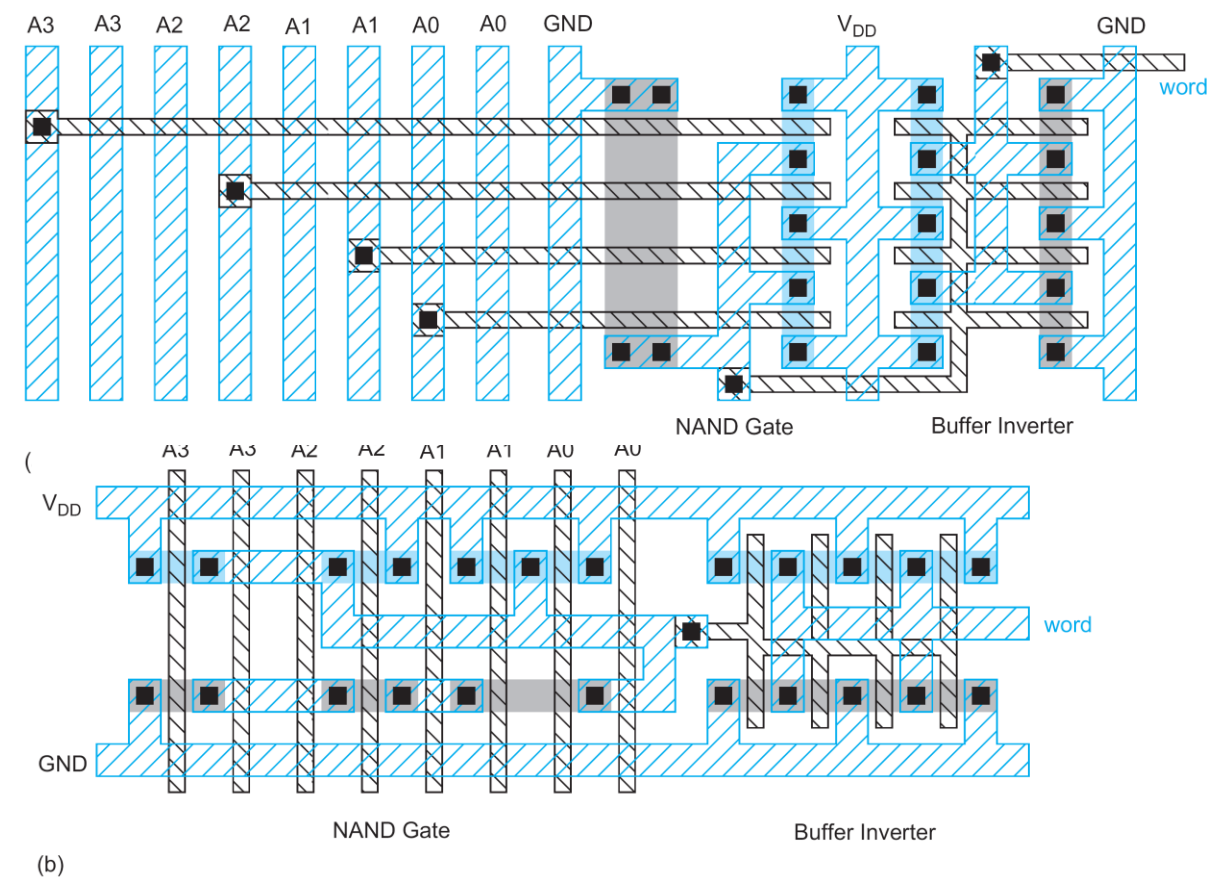
- Vertical routing distributes $2(n - k)$ wires for $A_0, \bar{A}_0, \dots, A_{n-k-1}, \bar{A}_{n-k-1}$

- Straightforward layout:

- Overhead for vertical routing
- Height grows with number of bits (rows) in the memory

- Optimized layout

- Vertical routing on Poly can overlap with row circuit to save area
- Need to strap Poly with Metal to keep resistance sufficiently low
- Height independent of the number of rows

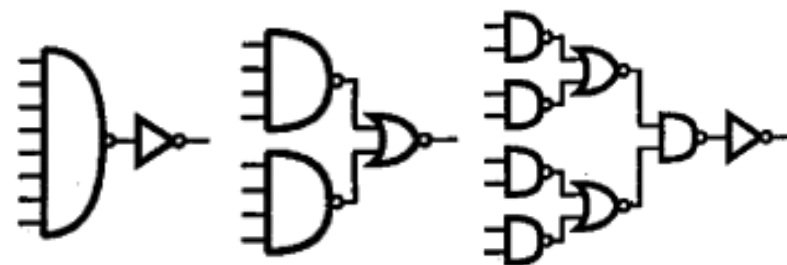


Limitations of the Single Stage Decoder

- **HOWEVER**, for memories with many (typically 128-1024) rows, some serious issues arise:

- **Growing (NAND/NOR) fan-in** of the row circuit

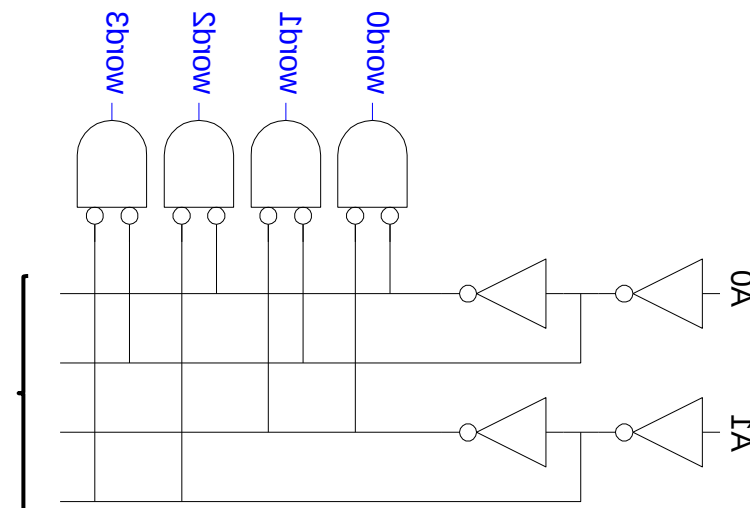
- Fan-in of each row circuit: $n - k$ (typical values ~7-10 bit)
- Difficult solution: **multi-stage row circuit** has high transistor
→ **difficult to pitch match** with bit cell



- **Growing fan-out** of the true and inverted address lines

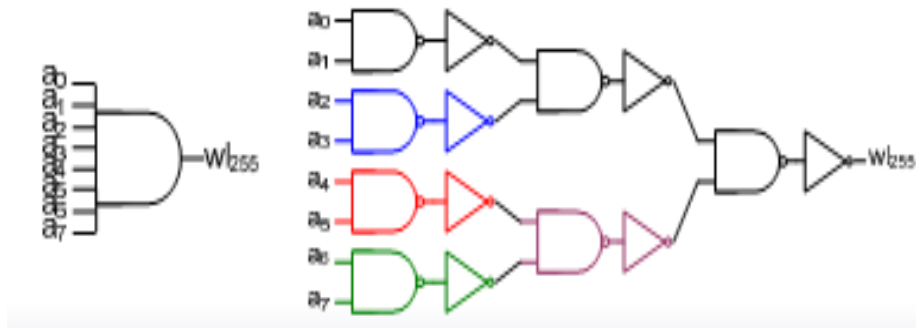
- $A_0, \bar{A}_0, \dots, A_{n-k-1}, \bar{A}_{n-k-1}$ are each tapped by $2^{n-k}/2$ gates
- More easy to solve by increasing driver strength

address line
load: 2^{n-k-1}

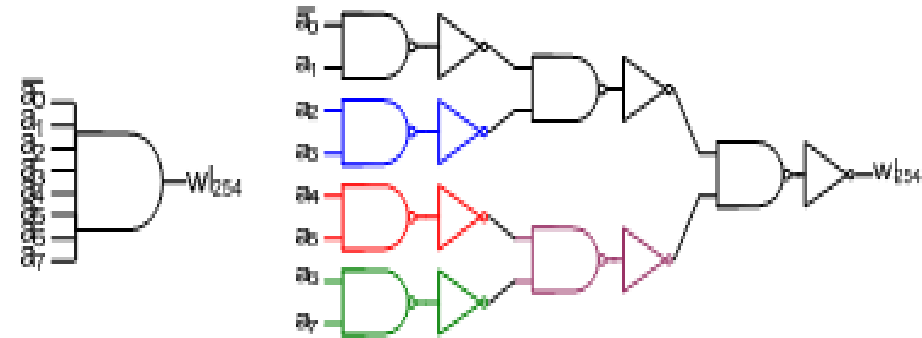


Multi-Stage Decoder with Pre-Decoding

- **Basic idea:** identify common sub-expressions between row decoders and share
 - Consider the row circuits of two adjacent rows (word-lines): W_{254} , W_{255}



$$WL_{255} = A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0$$



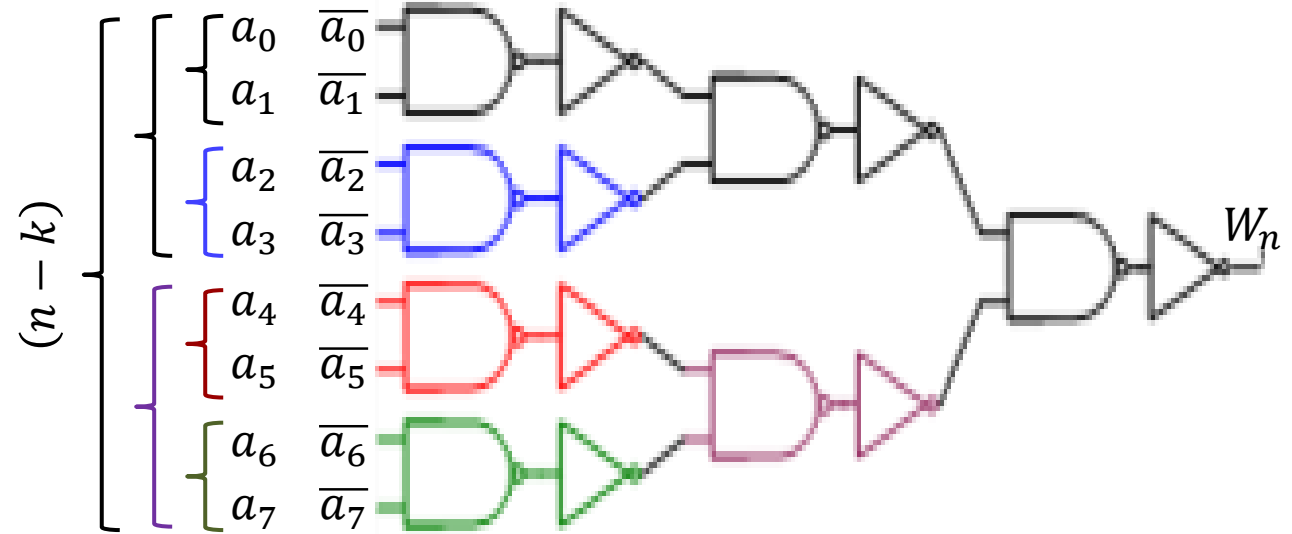
$$WL_{254} = A_7 A_6 A_5 A_4 A_3 A_2 A_1 \overline{A_0}$$

- **Many sub-expressions are common to different word-lines**
 - These sub-expressions can be computed only once

Multi-Stage Decoder with Pre-Decoding

- **Systematic computation of common sub-expressions:**

- Binary-tree decoder: sub-expressions concern groups of k subsequent bits, where g is a power-of-2 (2, 4, 8, 16, ...)
- For a group size of g bits, there are $(n - k)/g$ groups



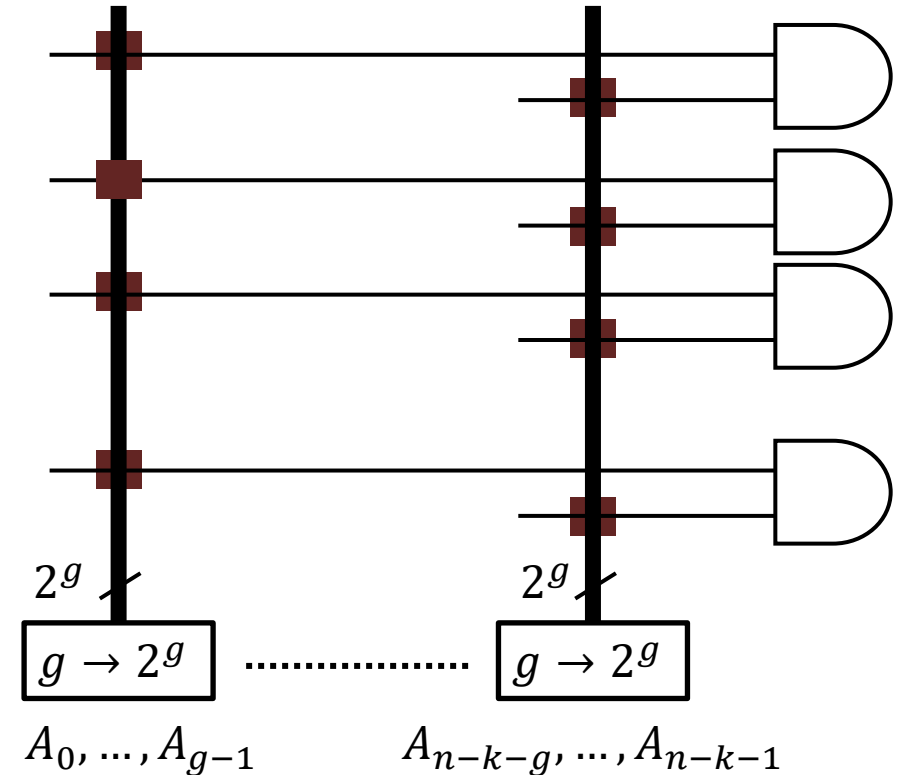
$$W = \text{dec}(A_0, A_1, A_2, A_3, A_4, A_5, A_6, A_7)$$

$\underbrace{\hspace{1.5cm}}_{(n-k)}$

Multi-Stage Decoder with Pre-Decoding

- **Systematic computation of common sub-expressions:**

- Each group of g bits is pre-decoded separately by a dedicated pre-decoder into 2^g one-hot signals
- Each row decoder now receives a unique combination of $n - k - g$ bits, one from each pre-decoder.
- Word-line is asserted of the right combination of pre-decoded signals is asserted.



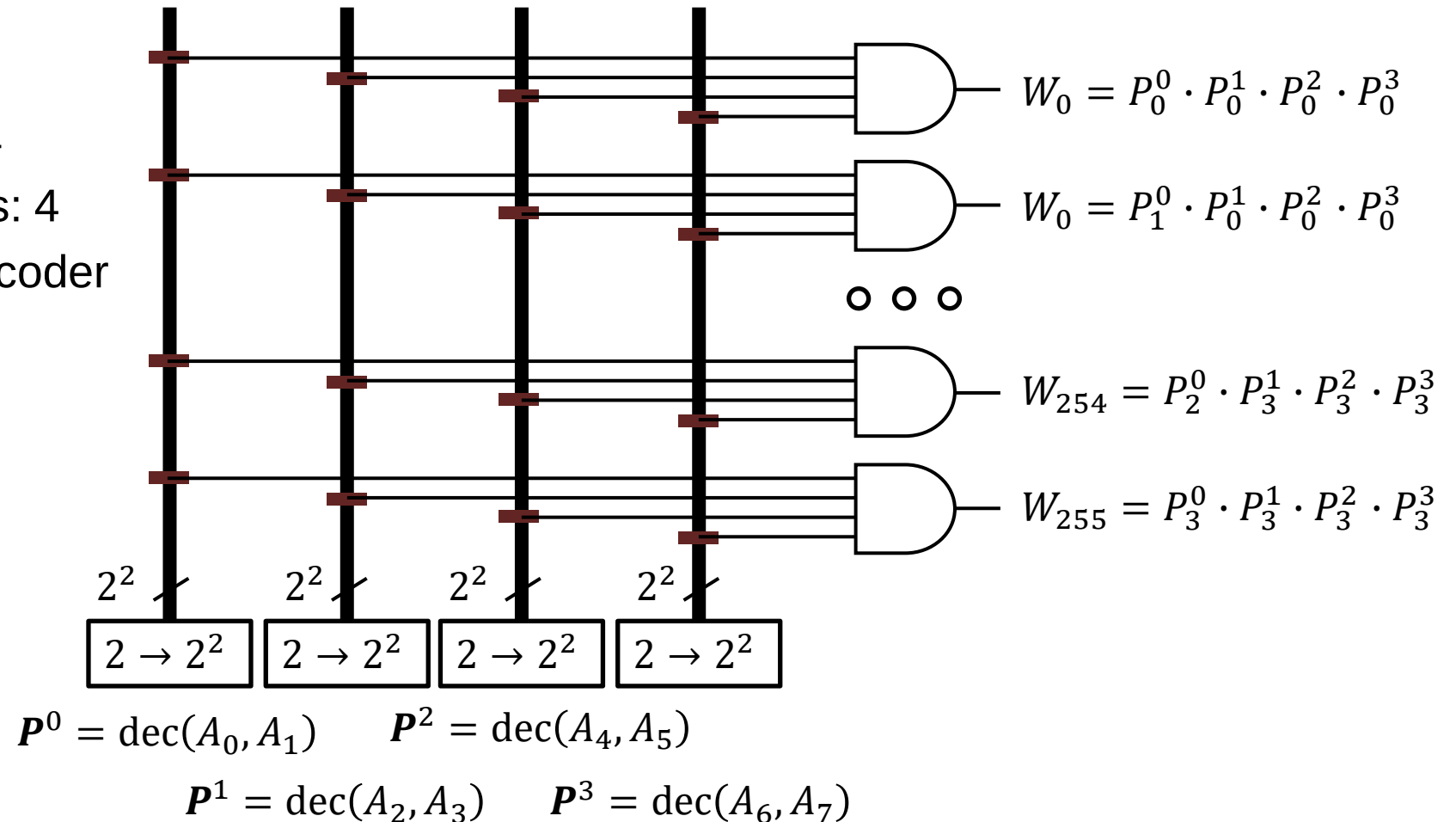
Pre-decoder: $P^i = \text{dec}(A_{i \cdot g}, \dots, A_{(i+1) \cdot g - 1})$

Post-decoder: $W_j = \prod P_{\emptyset(i,j)}^i$ where $\emptyset(i,j)$ selects the bit in the output of pre-decoder i that corresponds to the address j

Multi-Stage Decoder with Pre-Decoding: Example

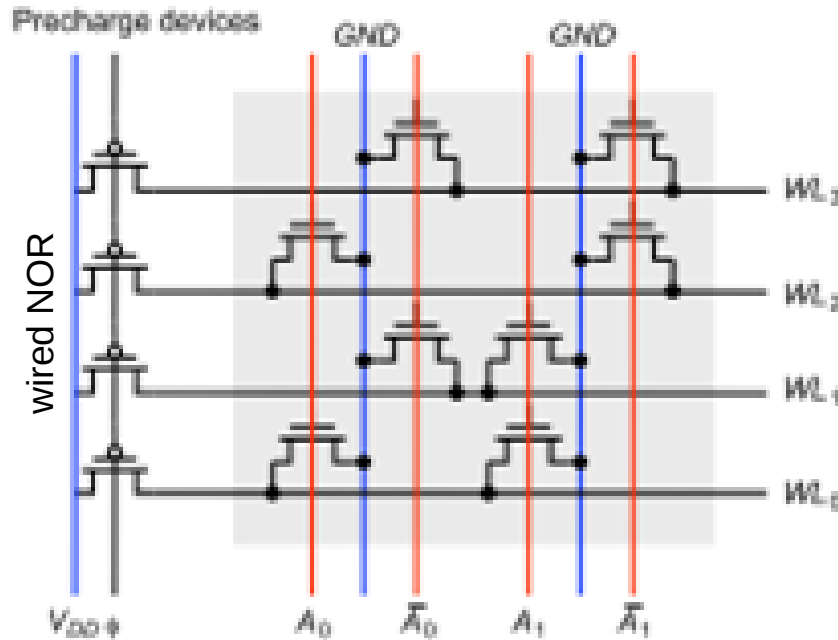
- Consider a memory with 256 rows and a 2-bit pre-decoder

- # of row address bits: 8
- # of pre decoder: $8/2=4$
- Fan-in of post decoders: 4
- Fan-out of each pre-decoder output: $256/4=64$



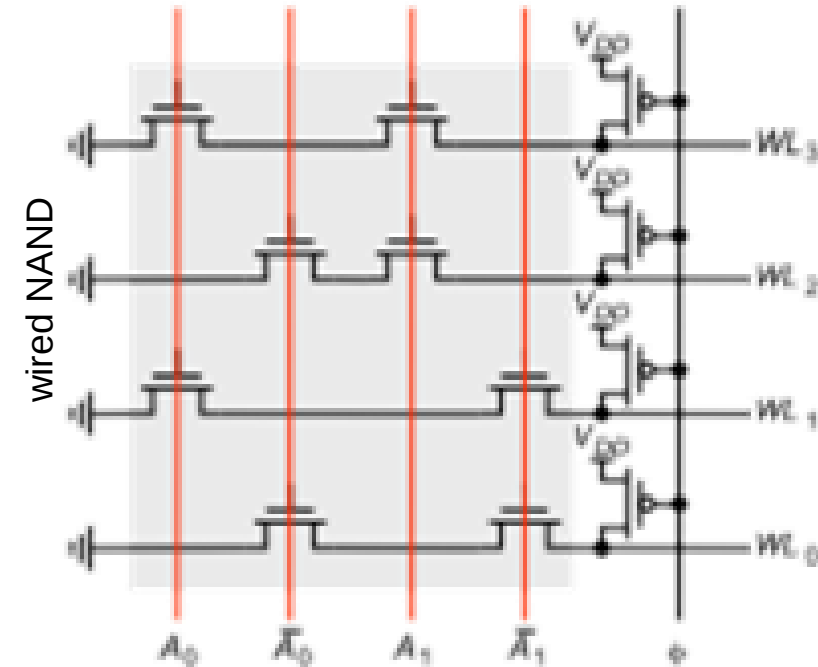
Alternative Solution: Dynamic Decoders

- **Even with extensive pre-decoding, row decoders are still large and slow**
 - Important reason: need of CMOS structures for a complementary PMOS in the post-decoder



All wordlines precharged to VDD,
all unselected lines pulled to GND

- Fast (single nMOS pull-down)
- High power consumption

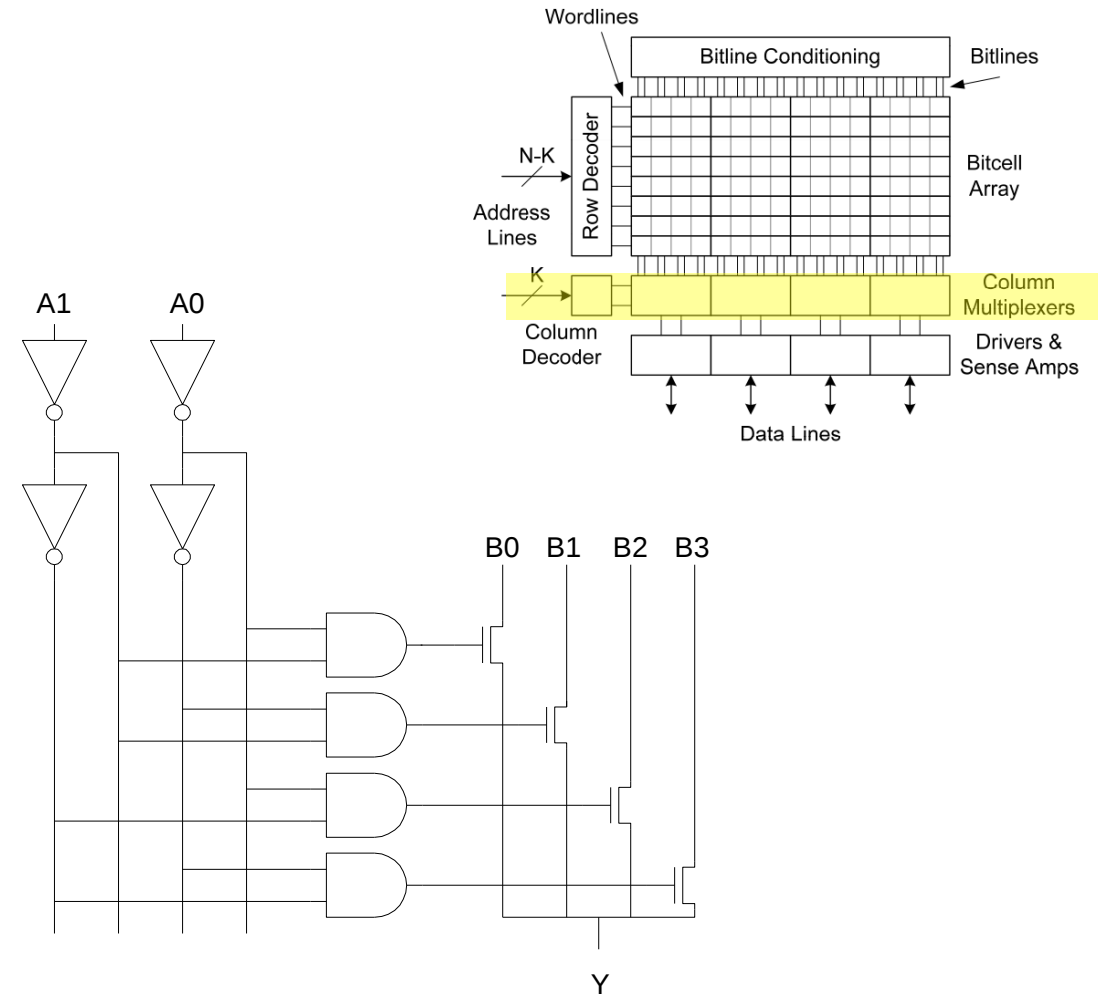
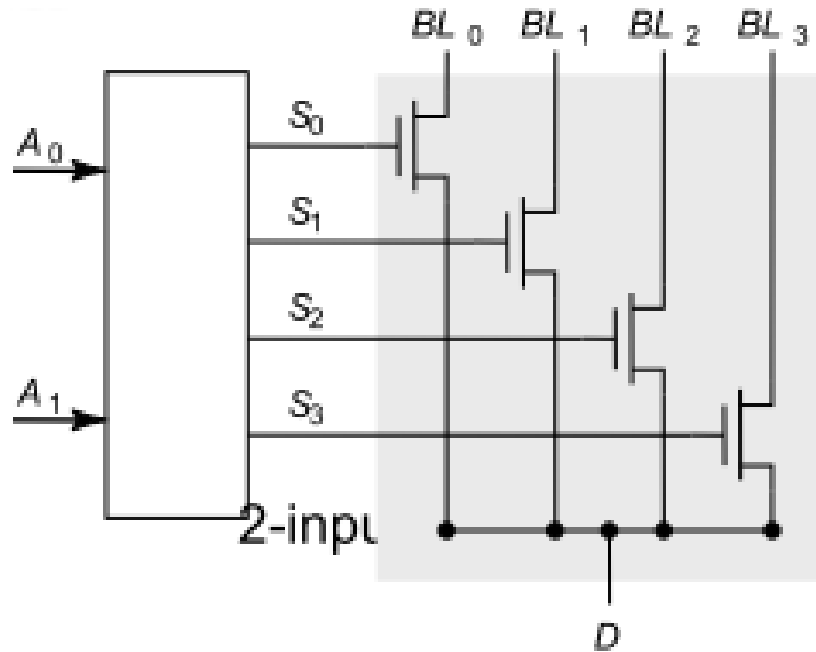


All wordlines precharged to VDD,
only selected line pulled to GND

- Slow (many nMOS in series)
- Low power consumption

Column Multiplexer

- **First option – PTL Mux with decoder**
 - Fast – only 1 transistor in signal path.
 - Large transistor count for one-hot decoder

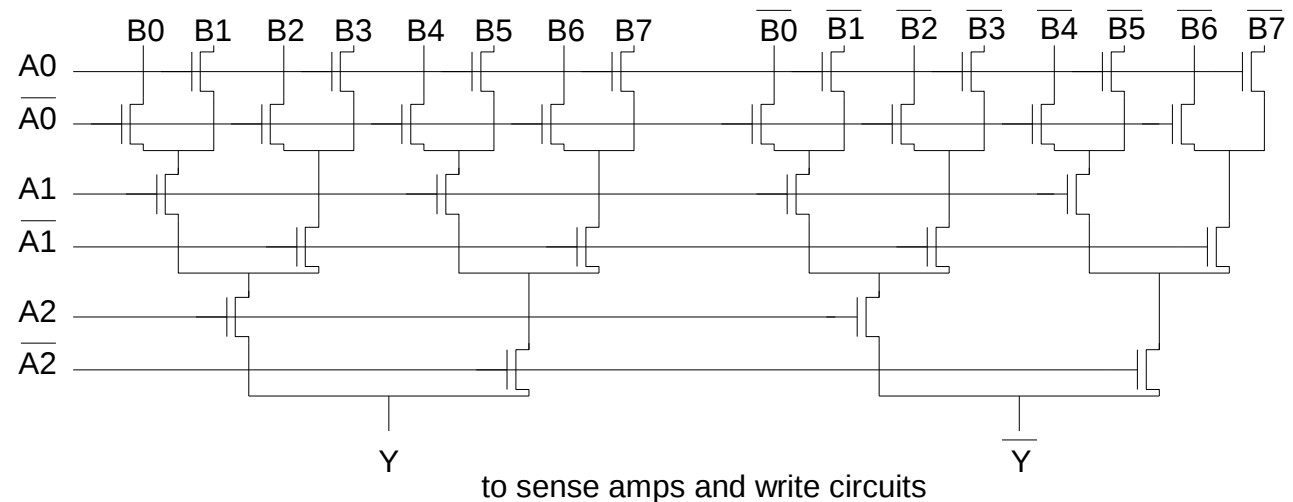
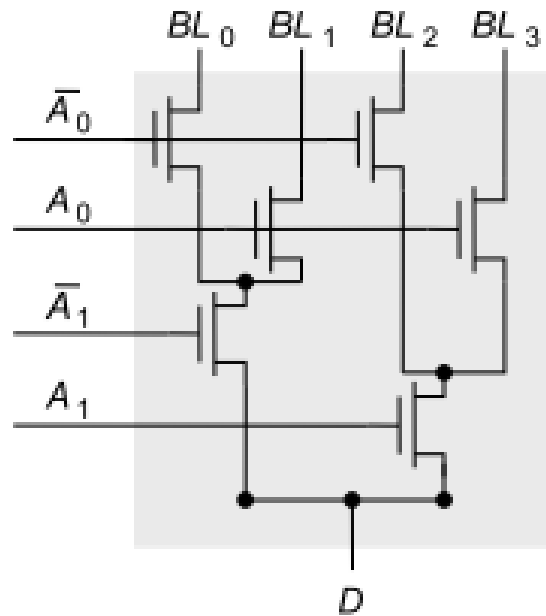


Fast, but many transistors in the pre-decoder

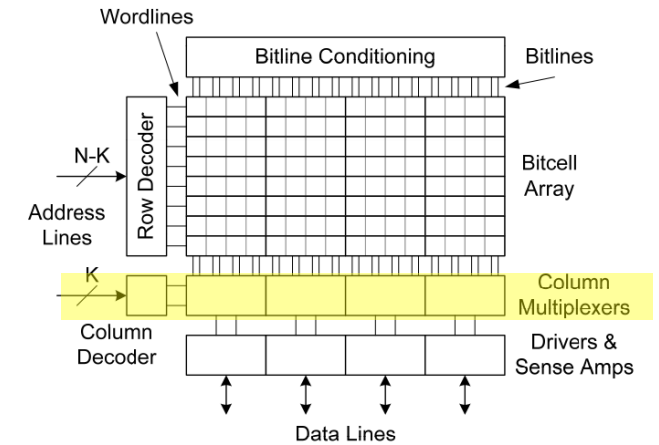
4 to 1 tree based column decoder

- **Second option – Tree Decoder**

- For $2k:1$ Mux, it uses k series transistors.
- Delay increases quadratically
- No external decode logic → big area reduction.

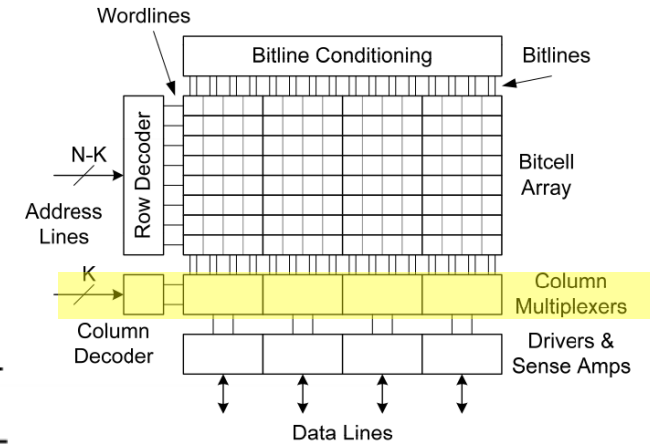
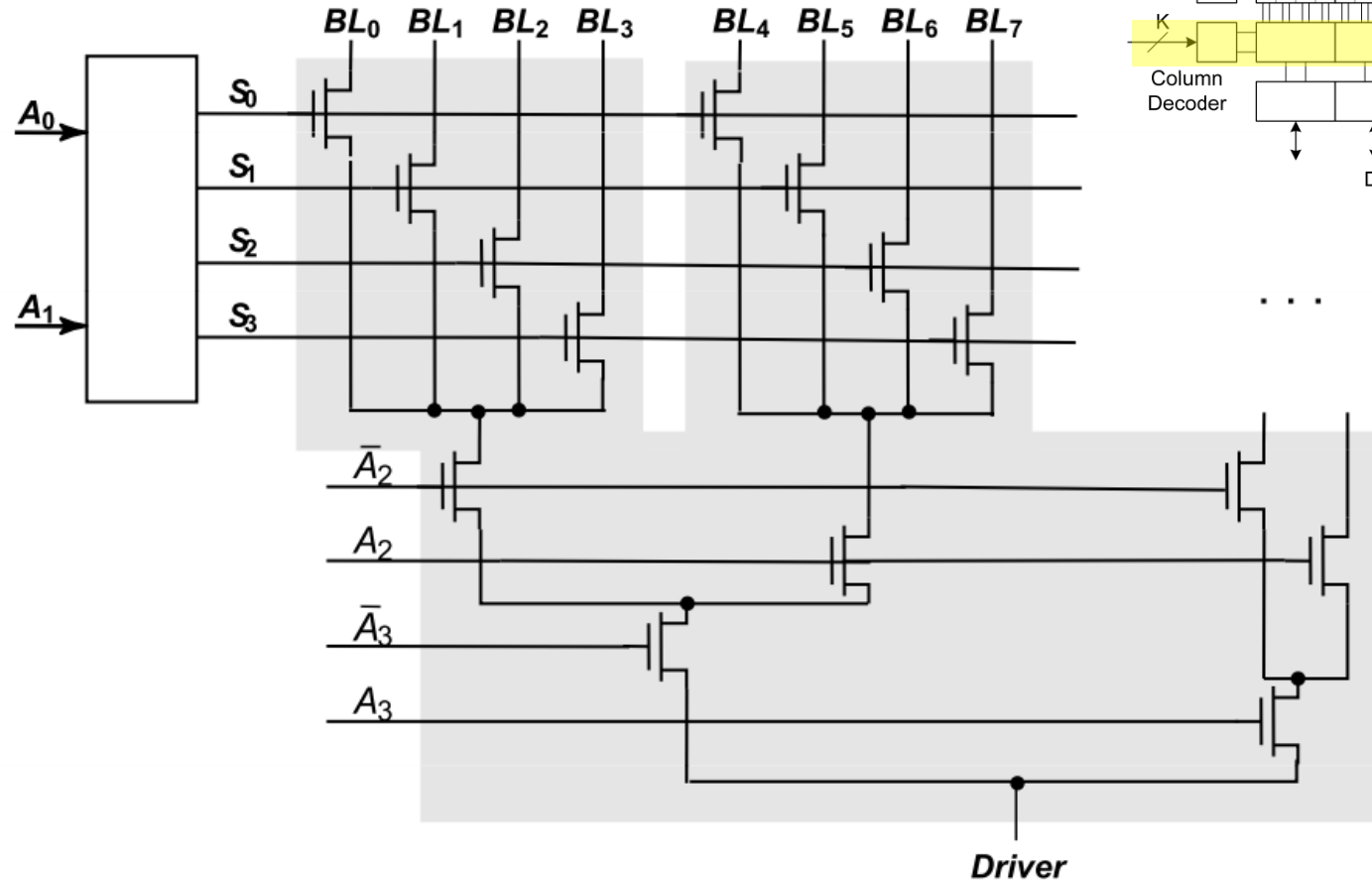


No pre-decoder, but slow due to many transistors in series



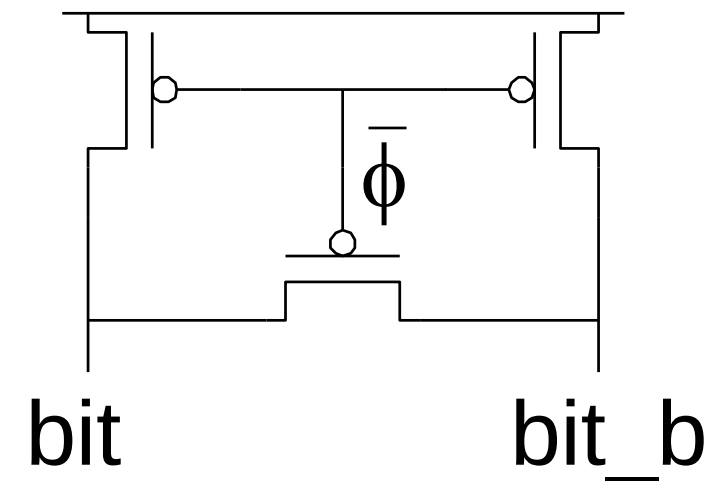
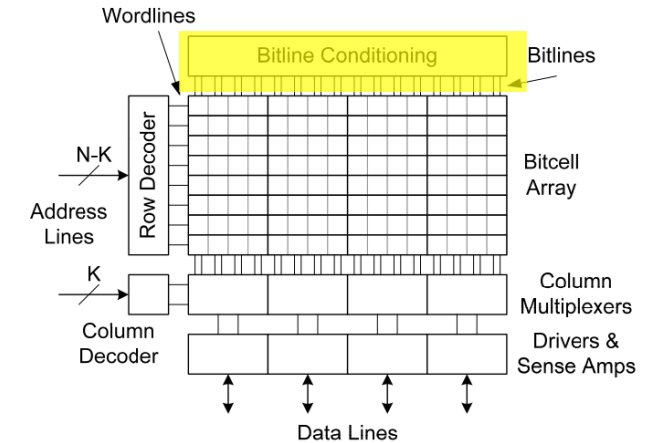
Combining the Two

- Combination of small PTL and larger tree-decoder based multiplexers are common for larger MUXes



Bit-Line Conditioning

- **Pre-charge bitlines high before reads**
 - Variations in transistor drive strength may lead to variations in the pre-charge level for a simple pre-charge driver
- **Equalize bitlines to minimize voltage difference when using sense amplifiers**
 - Important for differential sense amplifiers
 - Only minor overhead



Sense Amplifiers

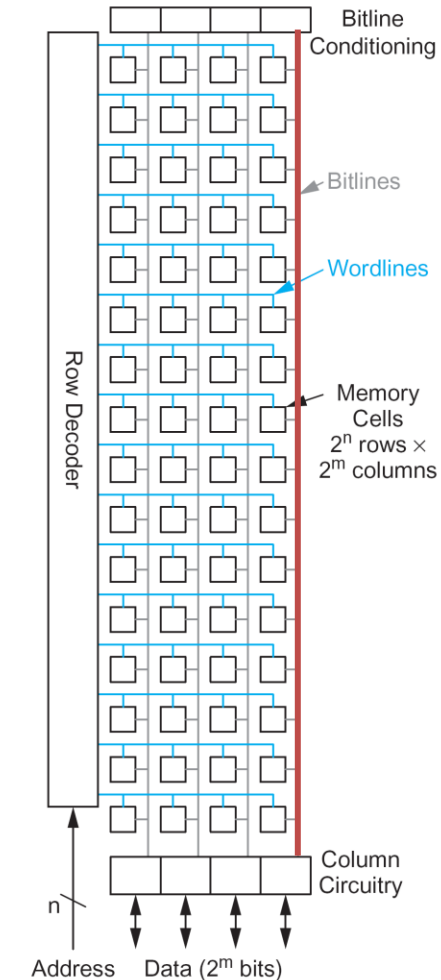
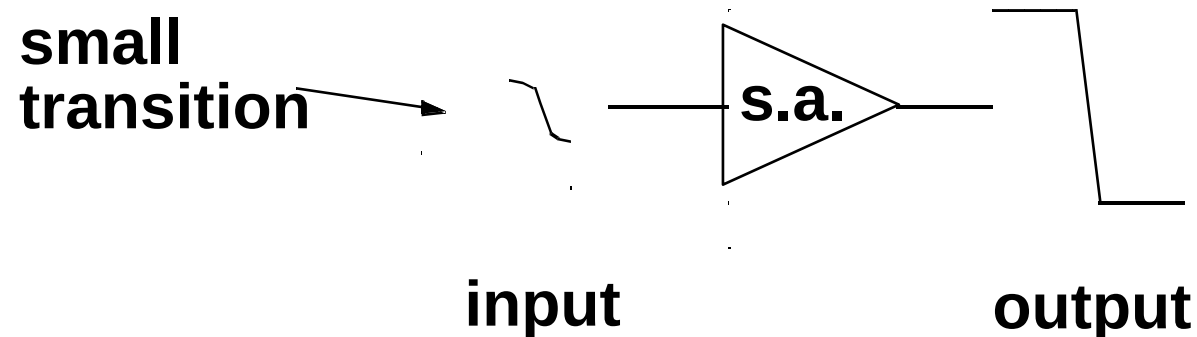
- During read, BL or BLB are discharged through the access transistor and the nMOS of the bit cell inverters

$$t_p = \frac{C \cdot \Delta V}{I_{av}}$$

Annotations for the equation:

- C : large
- ΔV : make ΔV as small as possible
- I_{av} : small

Idea: Use Sense Amplifier

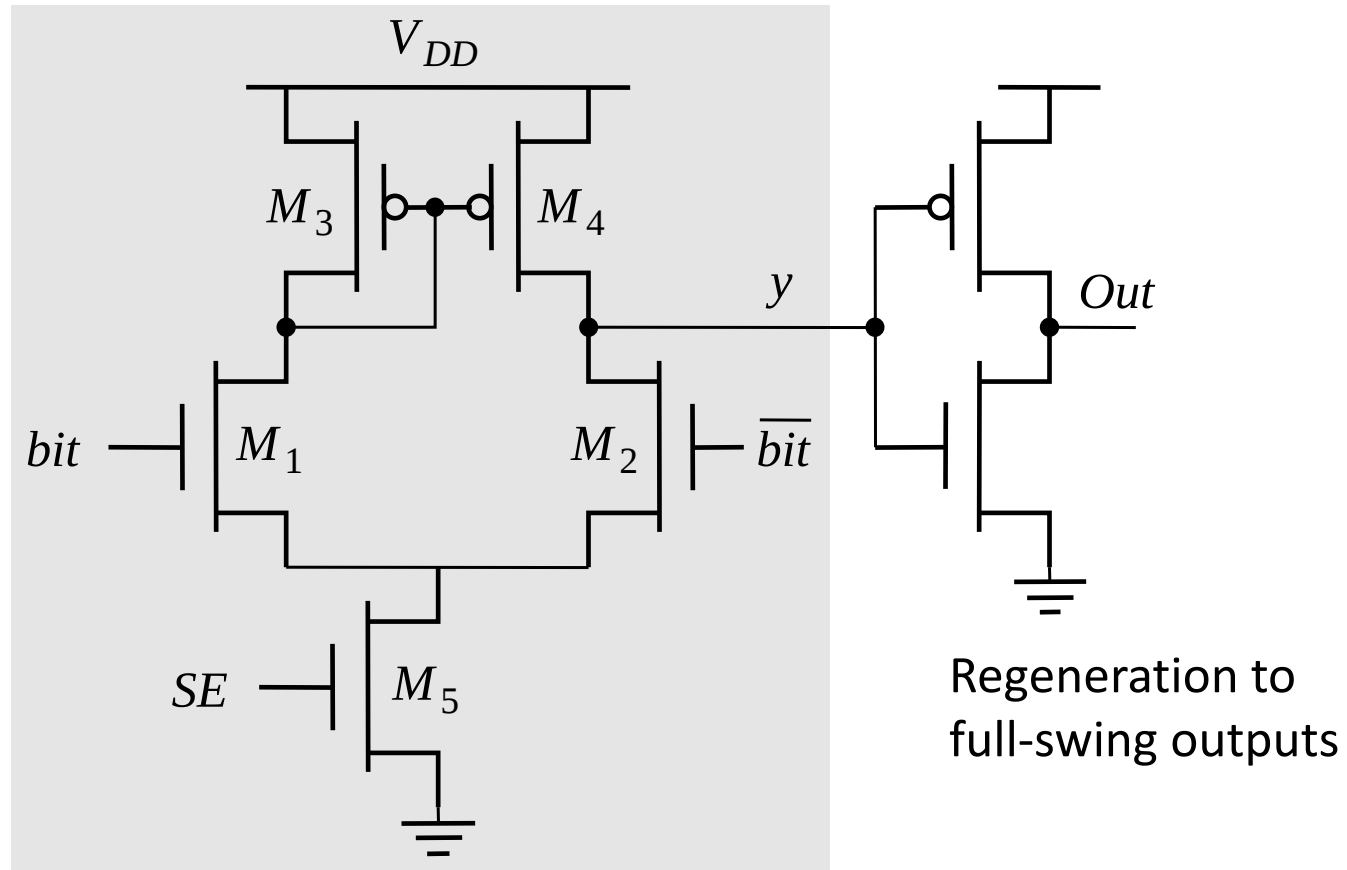


Differential Sense Amplifier

- **SRAM:** rapidly detecting already a small initial “swing” in either of the differential bit-lines

Basic differential amplifier: Current source with two differential transistors

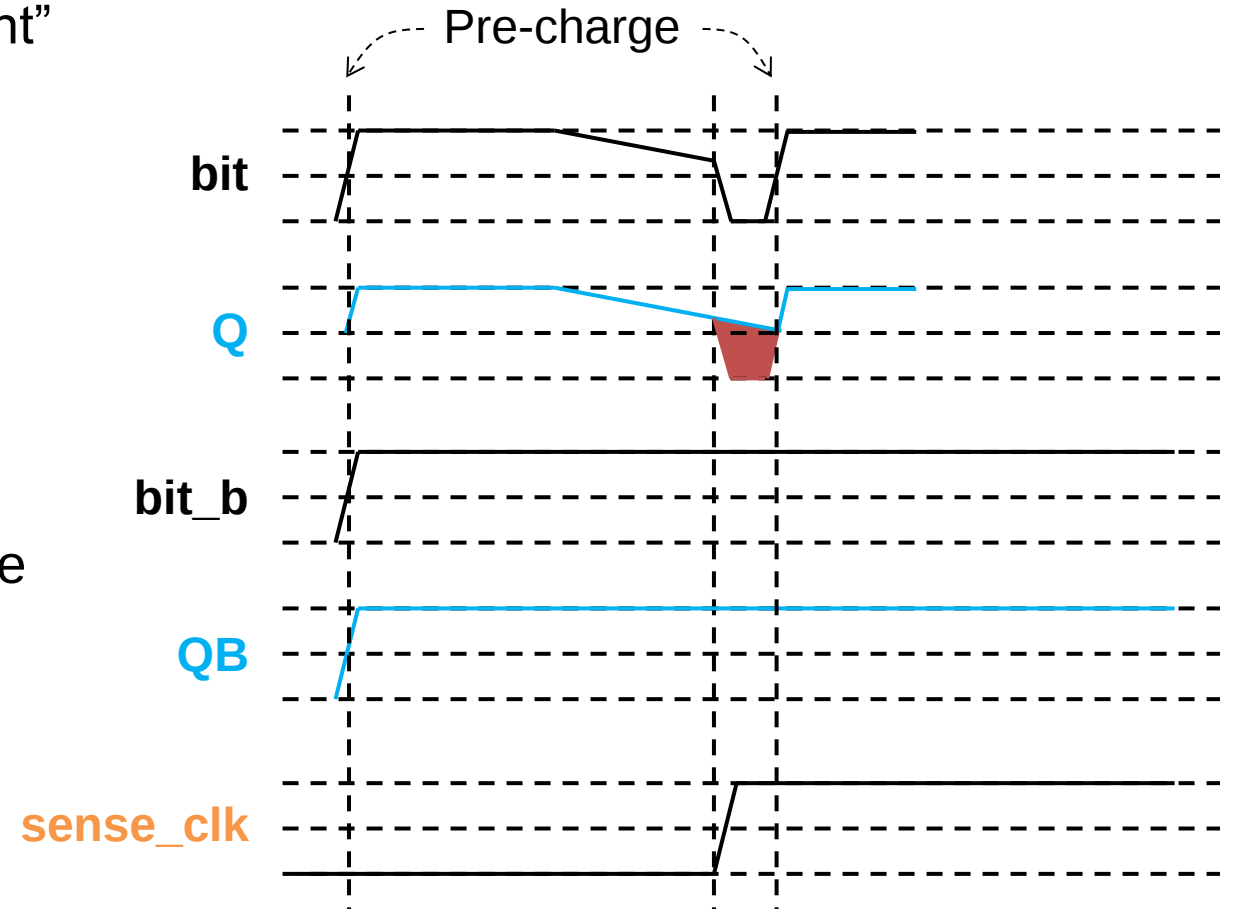
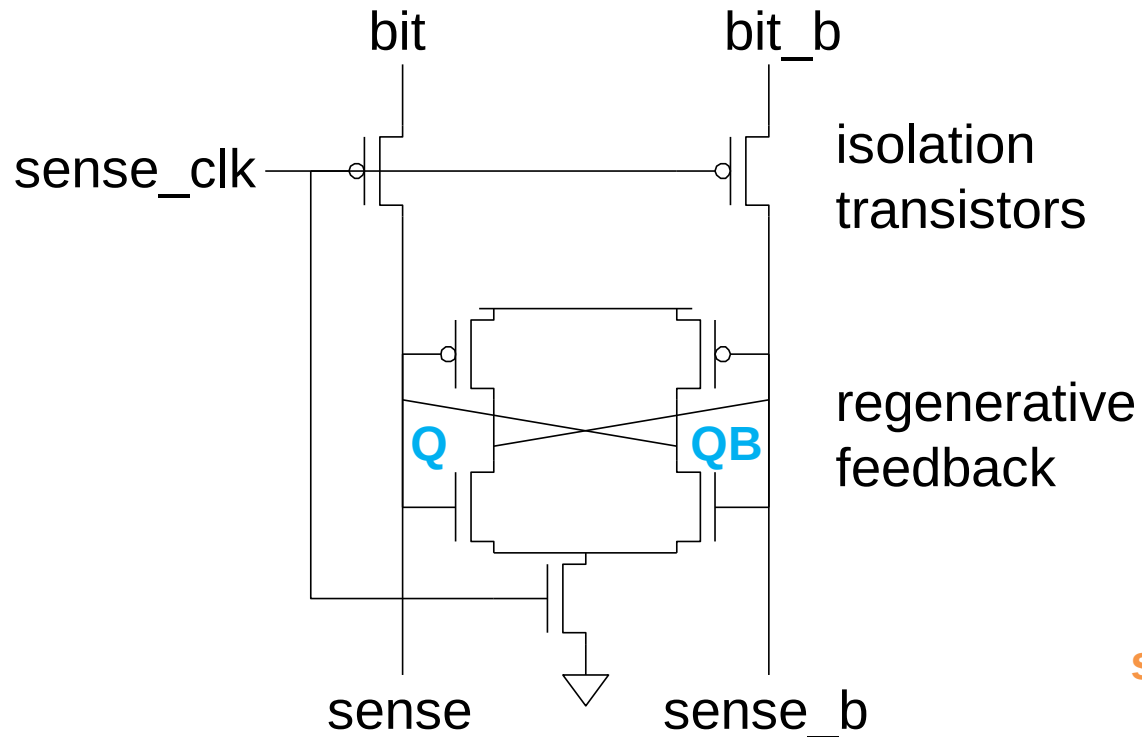
Disable when not needed to save power



Regeneration to full-swing outputs

Clocked Sense Amplifier

- **Clocked sense amplifier saves power**
 - Isolation transistors cut off large bitline capacitance
 - Requires sense_clk to arrive after “sufficient” bitline swing

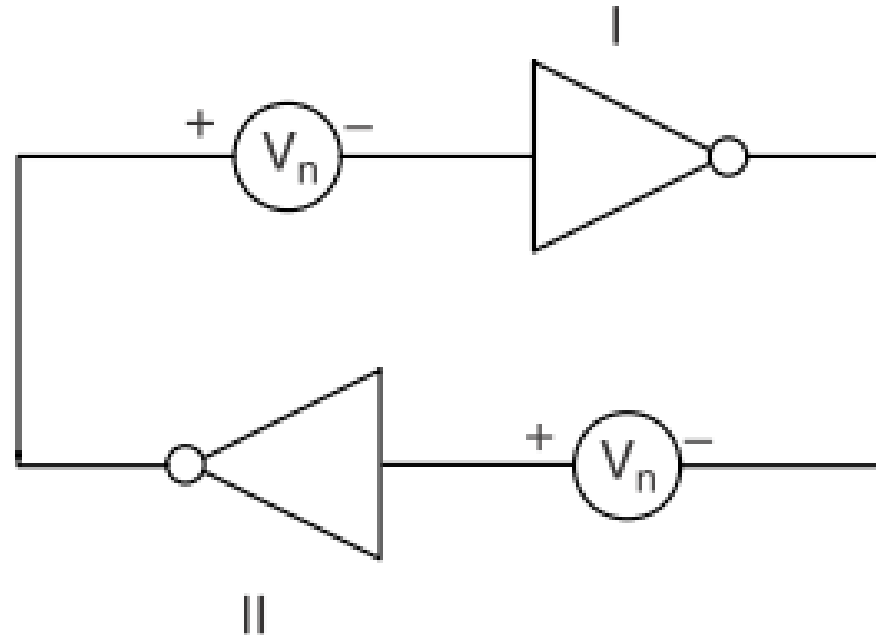


SRAM Reliability Concerns

- **Reliability is one of the main concerns in IC design**
- **SRAM is particularly prone to reliability issues for two main reasons:**
 - SRAM has the highest transistor density (and often most of the transistors) on the chip
 - SRAM relies on ratioed logic (calculated balance of transistor drive strengths) for operation, neglecting all best-practice rules of CMOS design
- **SRAM has multiple possible failures modes:**
 - Write failure: unable to write the correct value to a bit
 - Read failure: unable to correctly read a bit that is stored correctly
 - Read disturb: a read accidentally destroys the state of a stored bit
 - Hole failure: a bit cell fails to correctly keep the value of a bit

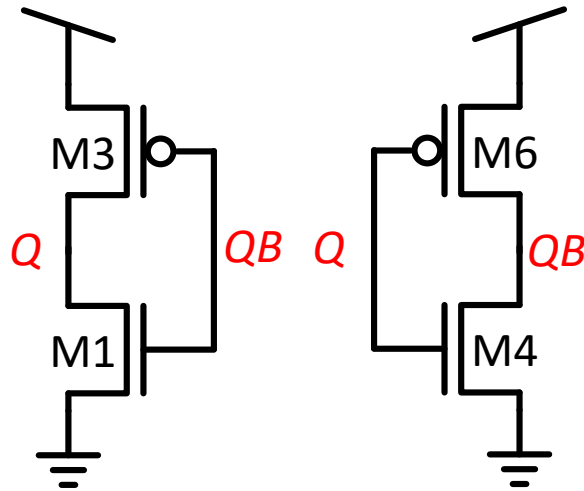
Static Noise Margin (SNM) - Hold

- **SNM is the voltage offset that can be tolerated without compromising functionality**
 - Static (pessimistic) metric determined by a DC analysis

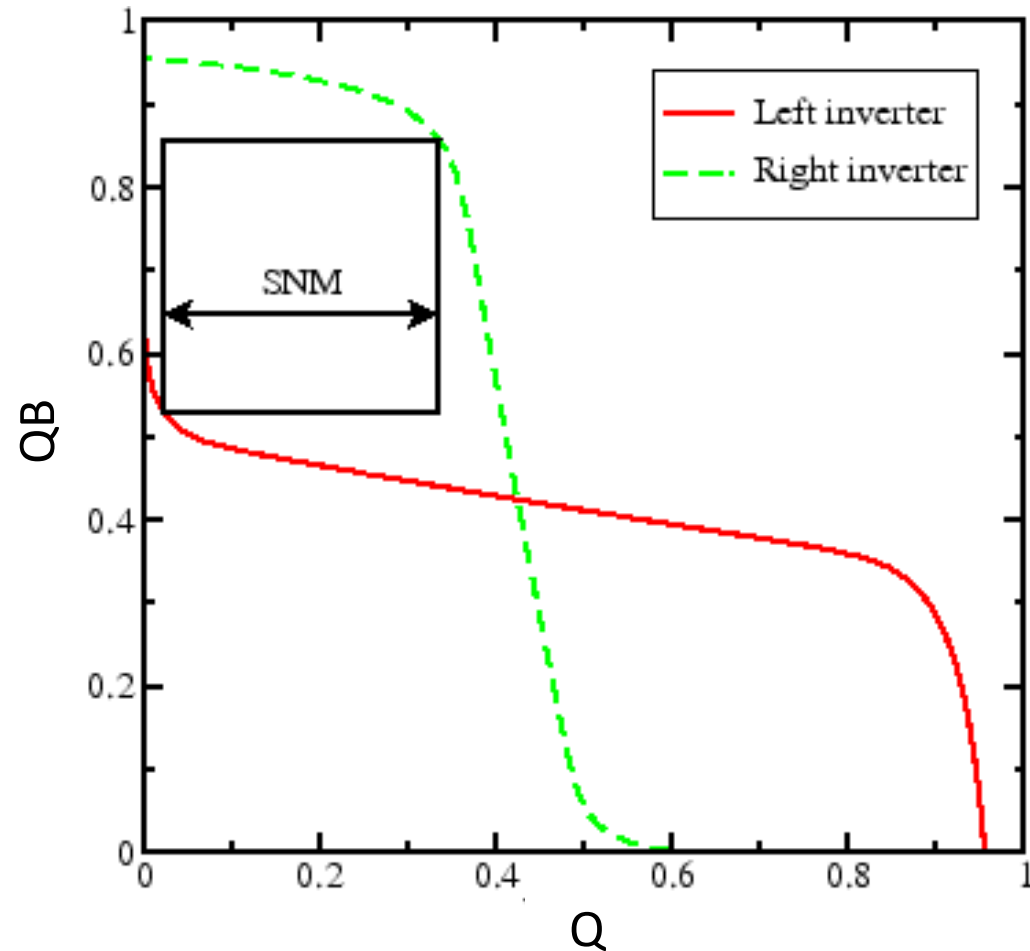


Static Noise Margin - Hold

- SNM is evaluated from the VTC of the separated (no feedback) bitcell inverters

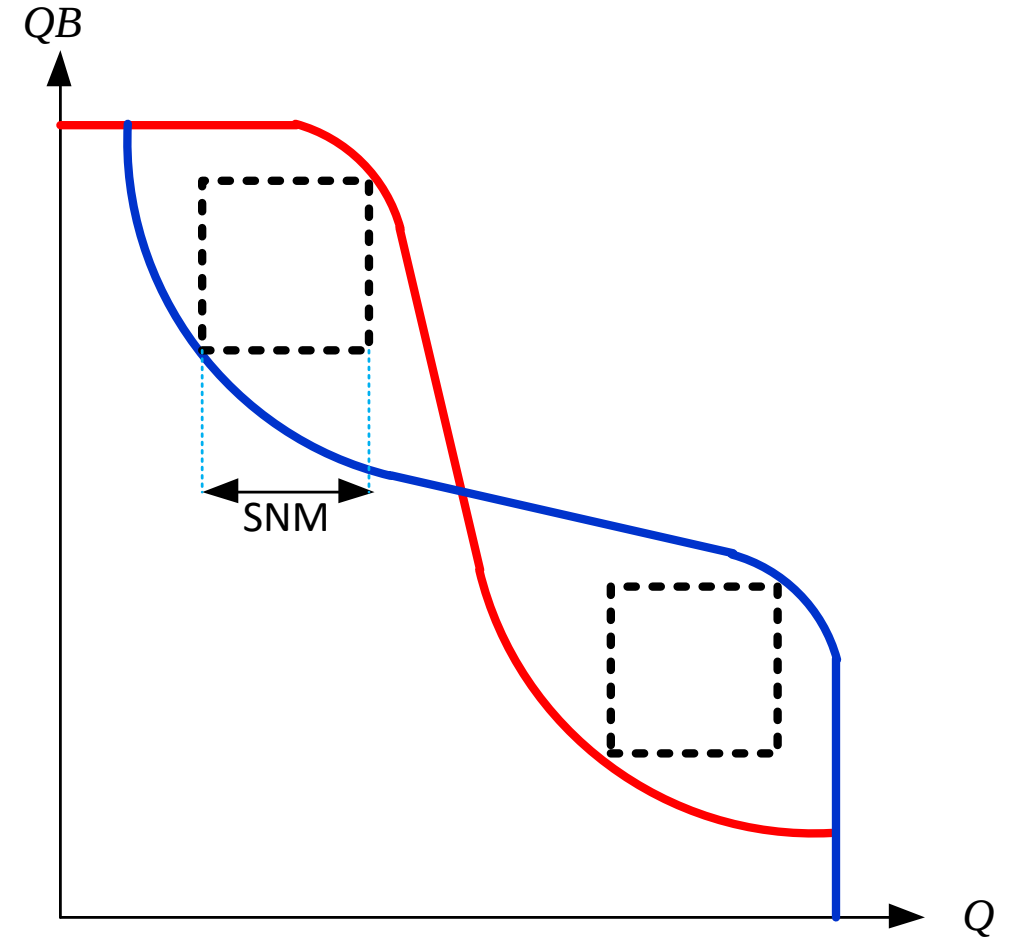
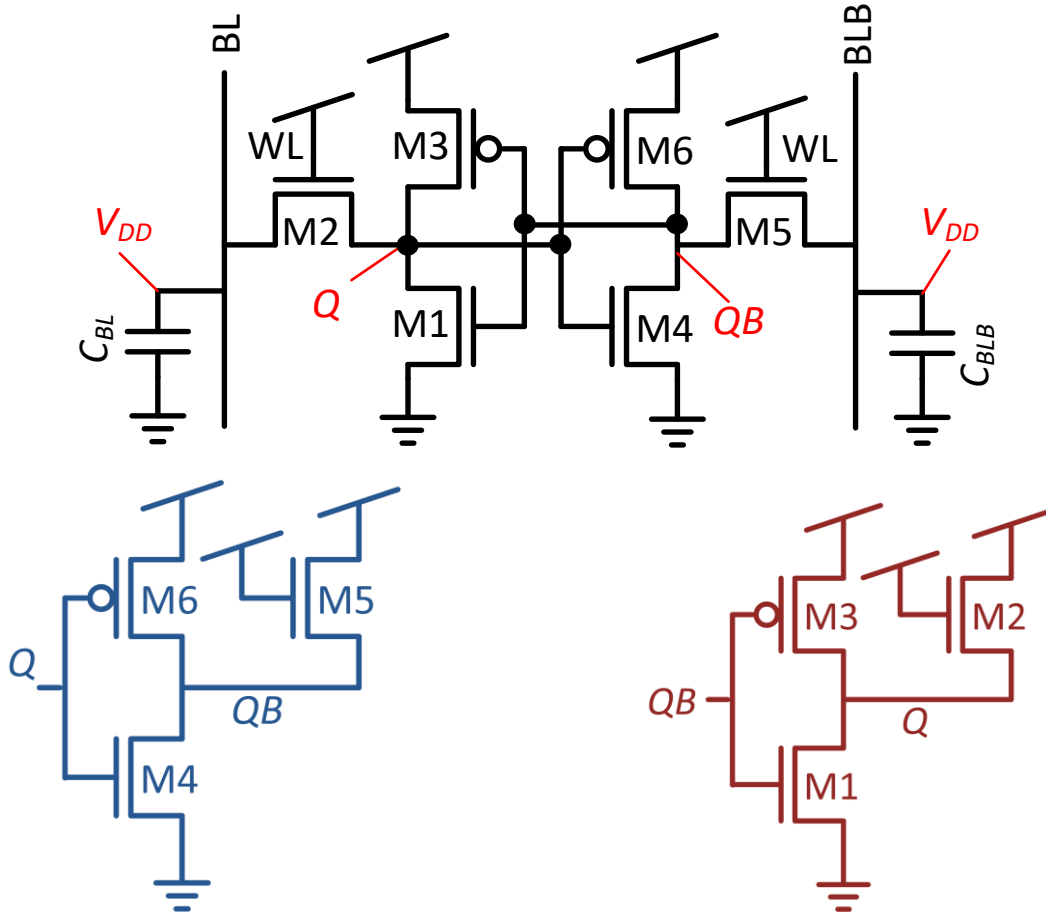


1. Plot both VTCs on the same graph
2. Find the maximum square that fits into the VTC (defined by the largest diagonal)
3. The SNM is defined as the side of the maximum square.



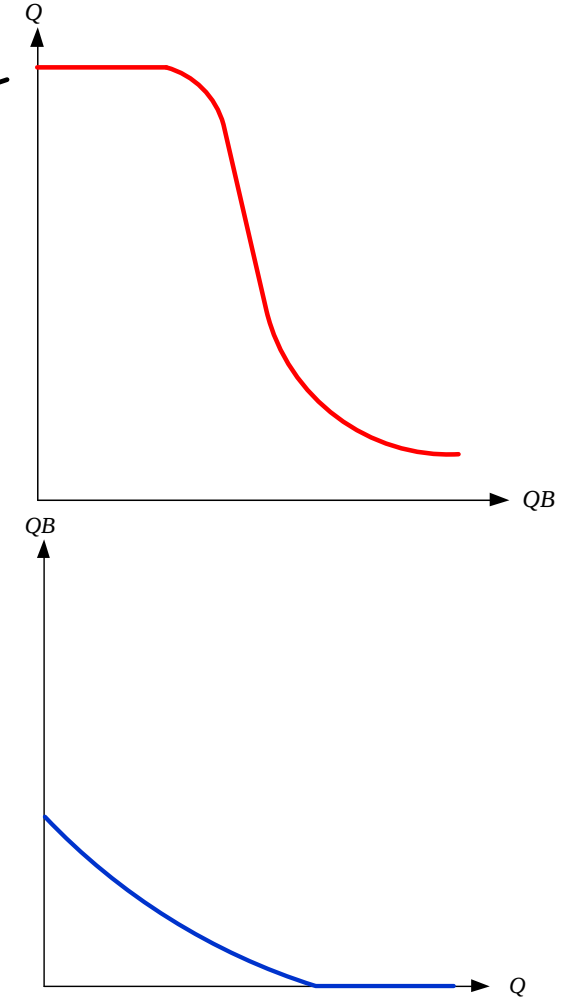
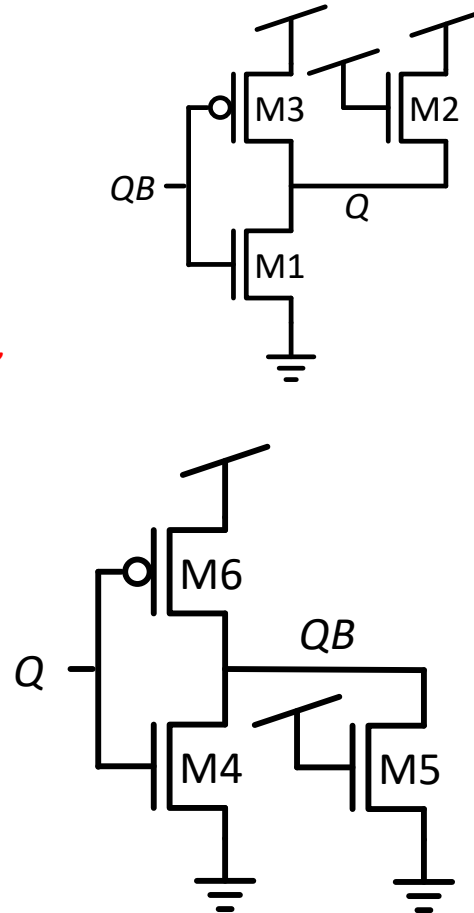
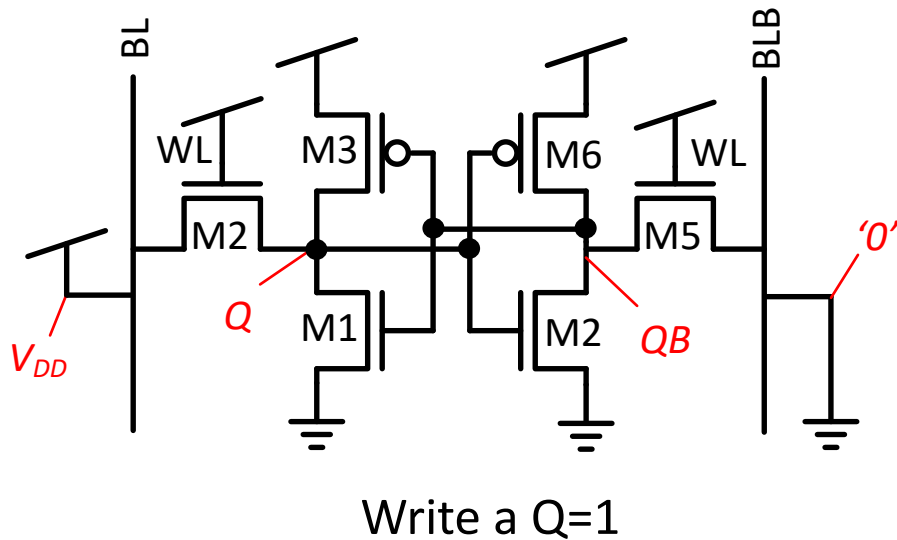
Static Noise Margin - Read

- **What happens during Read?**
 - Similar to HOLD, but include the access transistors

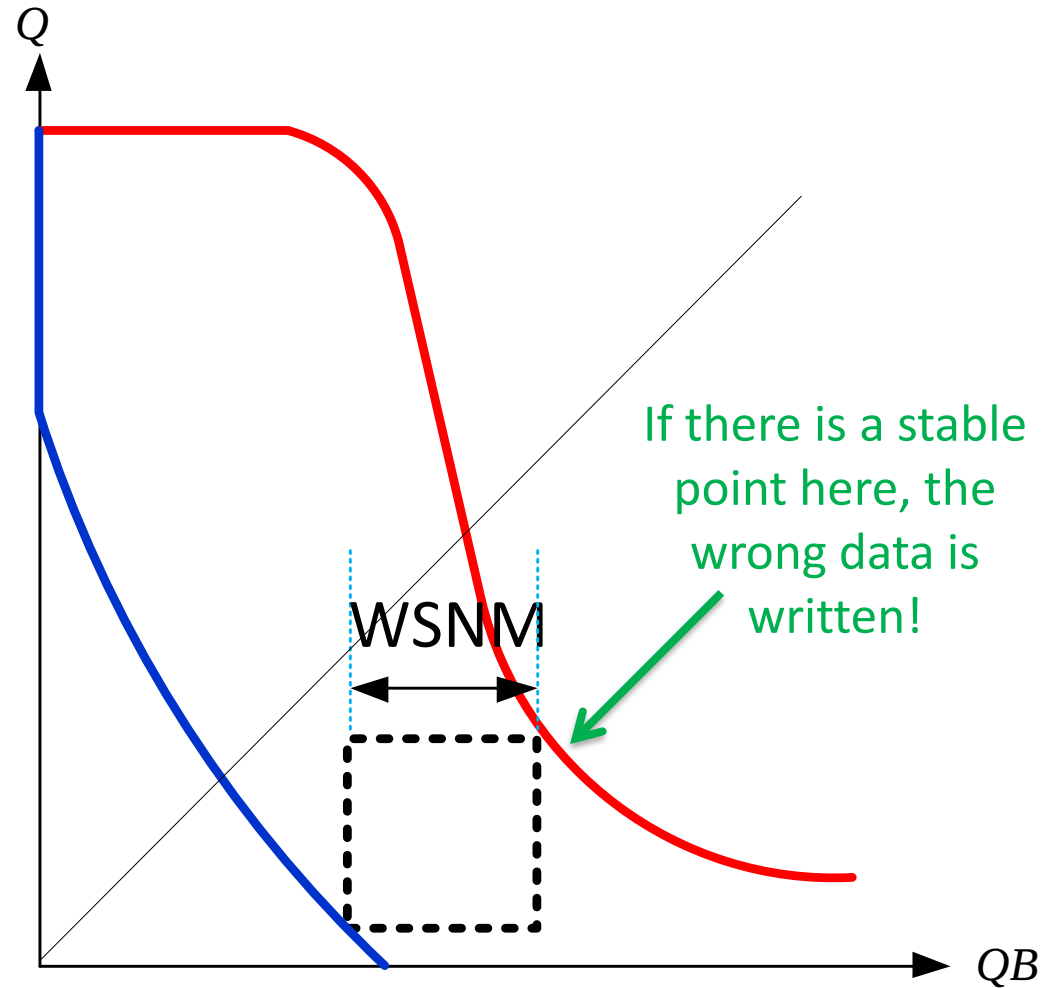
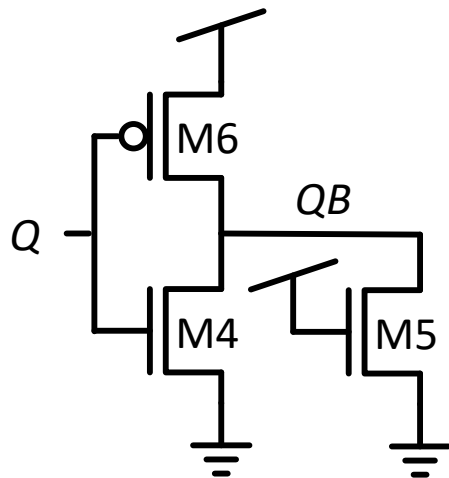
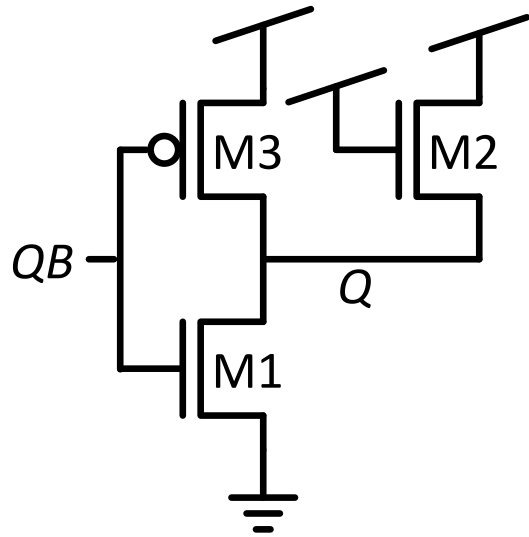


Static Noise Margin - Write

- **What happens during Write?**
 - The two sides are now different.

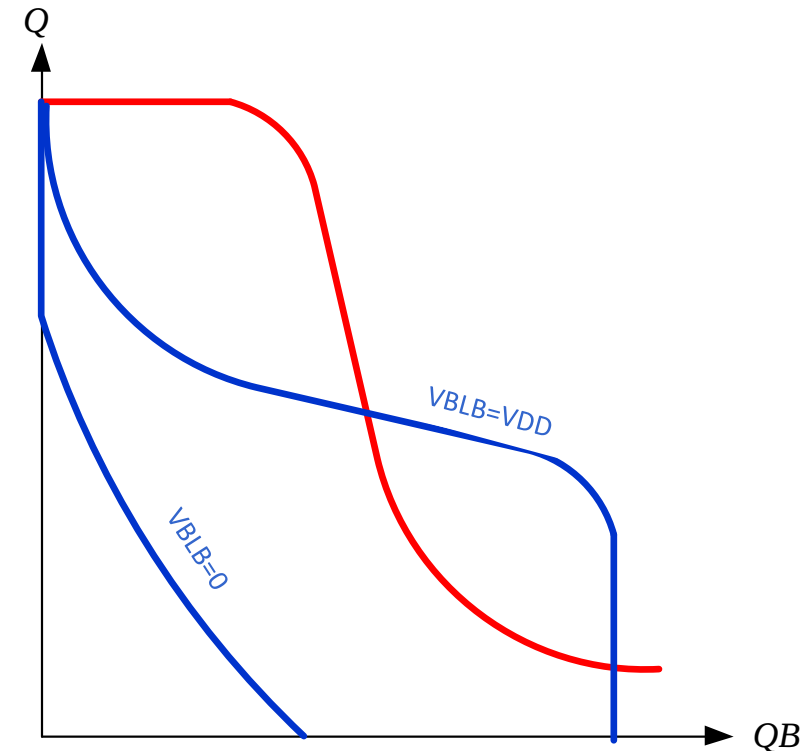
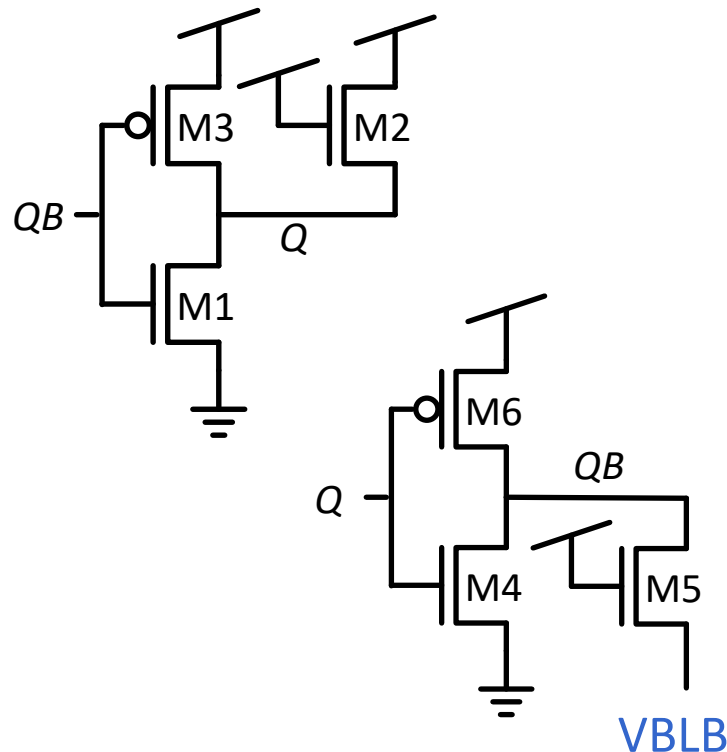


Static Noise Margin - Write



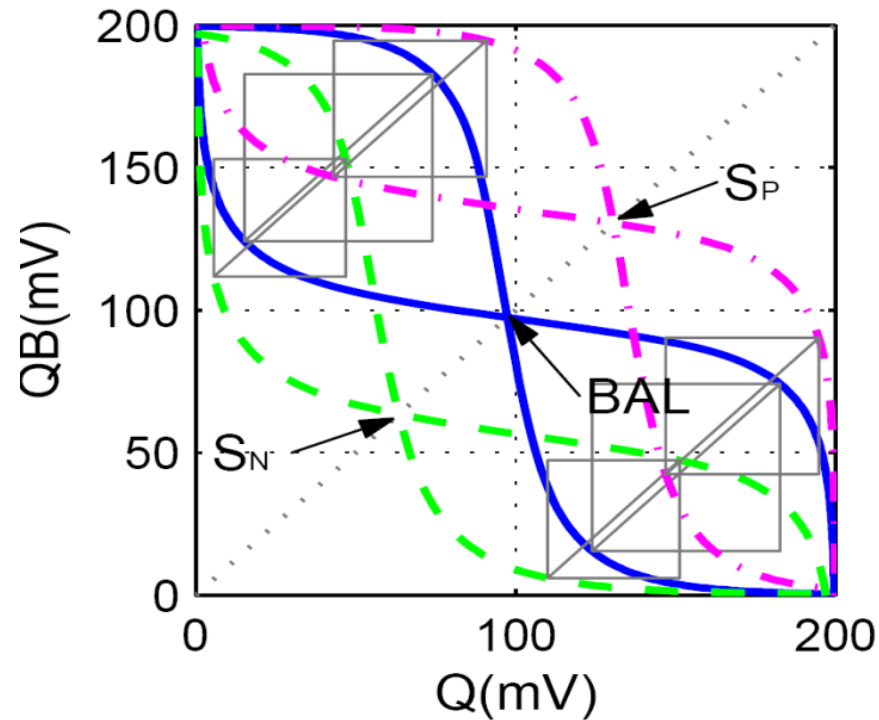
Alternative Write SNM Definition

- Write SNM depends on the cell's separatrix, therefore alternative definitions have been proposed.
- For example, add a DC Voltage (V_{BLB}) to the 0 bitline and see how high it can be and still flip the cell.



SNM for Variability

- Modern process technologies suffer from parameter uncertainties (i.e., transistor parameters can vary over a large range within a single chip)

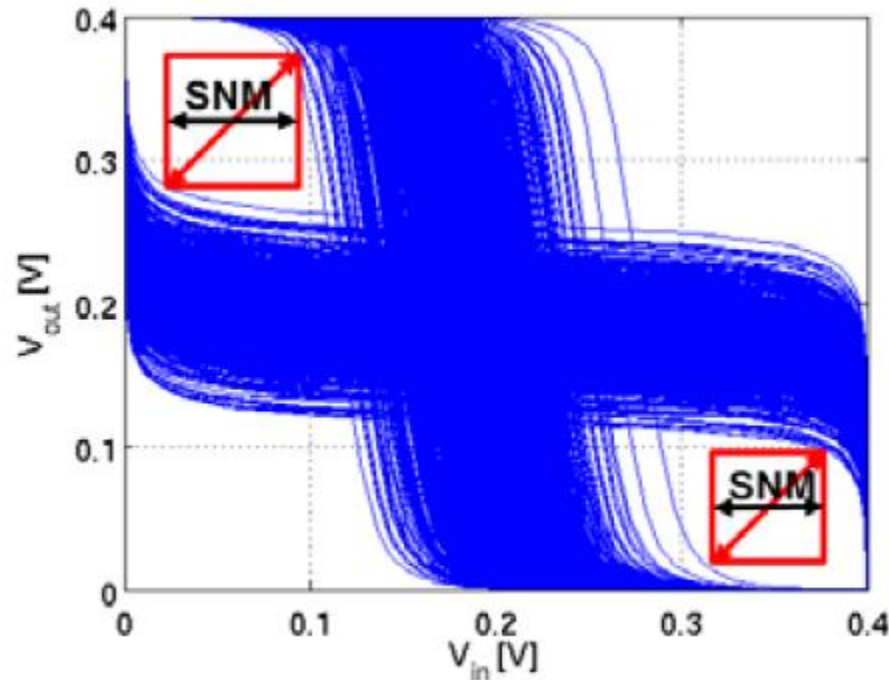


Stronger PMOS or NMOS (S_P, S_N)
SNM even for typical cell

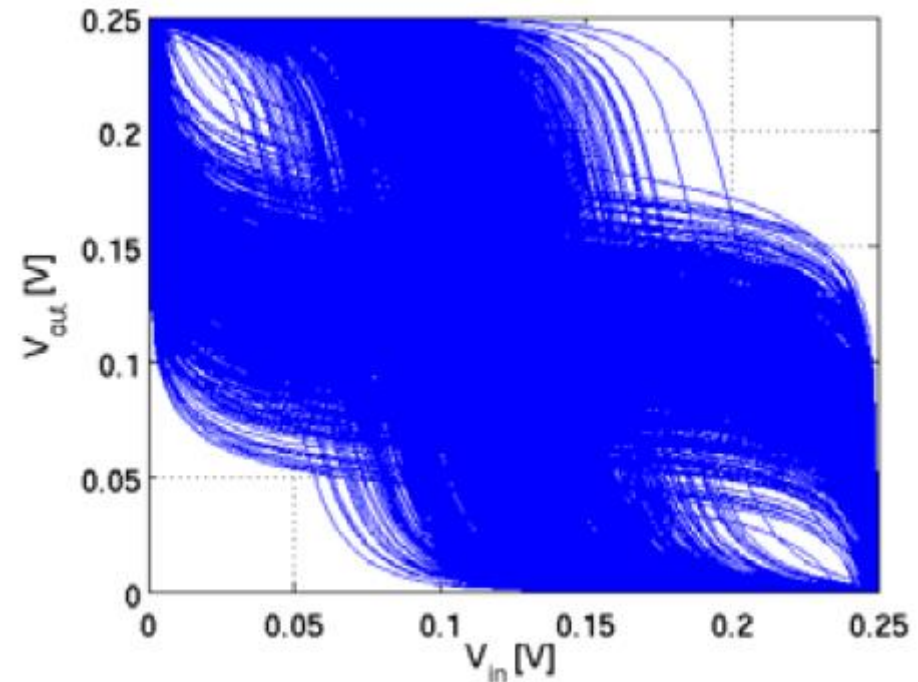
[Ref: J. Ryan, GLSVLSI'07]

Process Variations affect SNM of Memory Cells

- **Static Noise Margin (SNM):** Maximum amount of voltage noise on internal nodes of SRAM cells before data is lost
 - Simulation of 6T SRAM Cell in 65nm CMOS



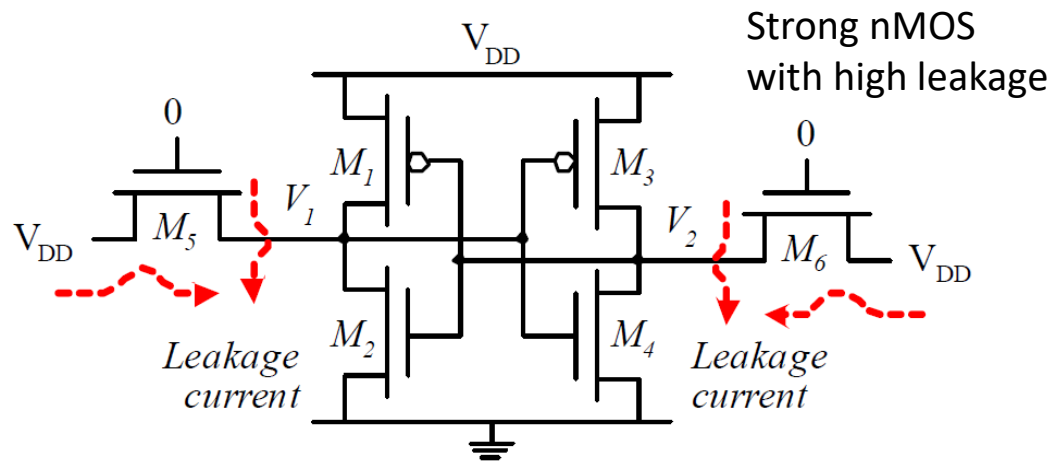
Good SNM: robust operation at **400mV**



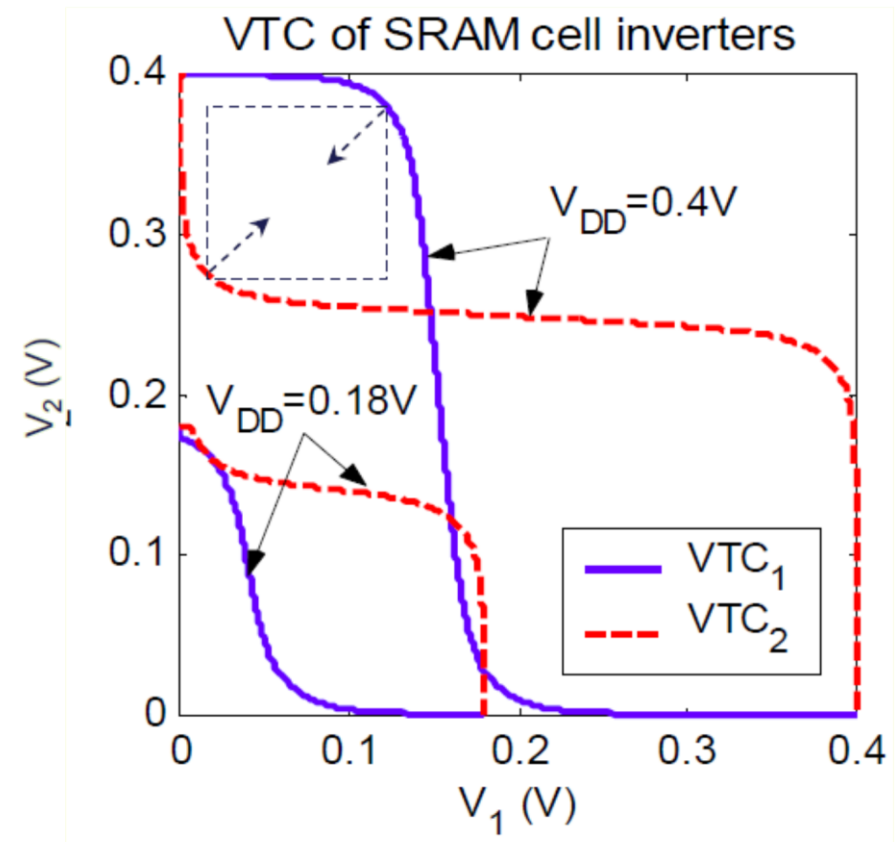
Operation at **250mV** is critical since **SNM diminishes**

SNM for Variability

- At low voltages, even 'off' transistors play a role, especially when variability causes large leakage
 - Degradation of Ion/Ioff ratio



Leakage and within die variability limit minimum operating voltage (e.g., data retention voltage)



EE-429

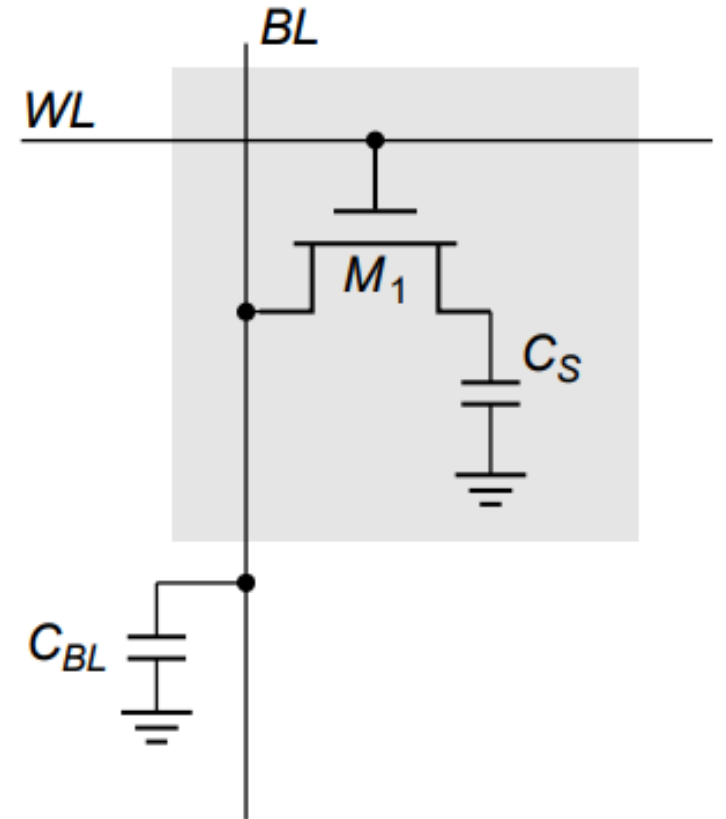
Fundamentals of VLSI Design

DRAM

Andreas Burg

Dynamic Random Access Memory (DRAM)

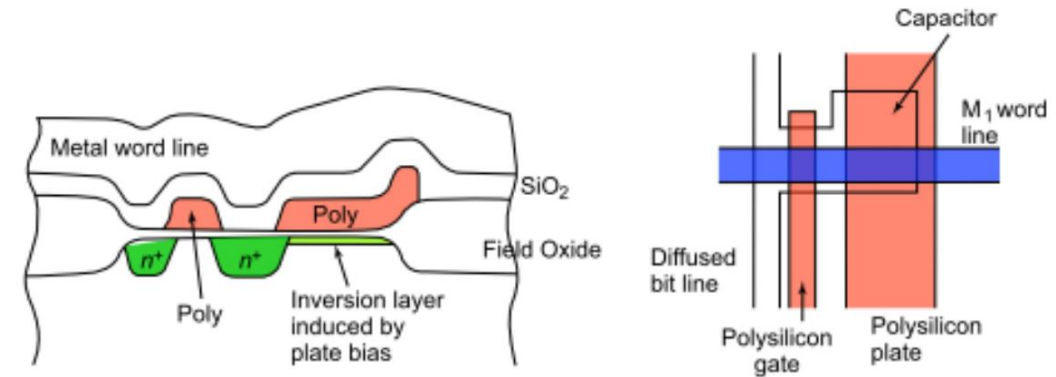
- **SRAM typically provides not enough density due to the large bit-cell**
 - Requires 6 transistors for each bit to implement a bi-stable storage and 2 access transistors
- **Dynamic storage: significantly more compact bit-cell**
 - Data is stored as charge on some form of capacitor C_S
 - A single transistor (M_1), controlled by the word-line (WL) is used to access the storage capacitor from the BL
- **Typical DRAM cell size in a special DRAM process is around $6-8 F^2$** (F: feature size of the process)
 - For comparison: **SRAM cell size** in standard CMOS is around $120-150 F^2$ ($>32\text{nm}$) and >200 below 16nm



1T-1C DRAM Fabrication

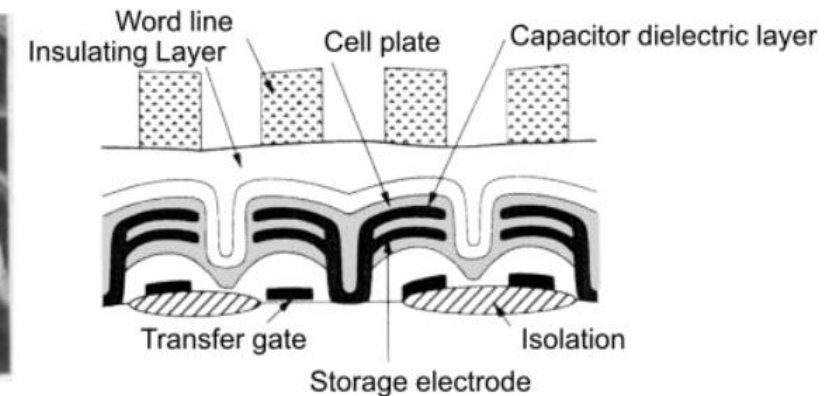
- **Standard process**

- Capacitor made from poly/diffusion
- Large area required for the capacitor
- Used mostly in 1970s and 1980s



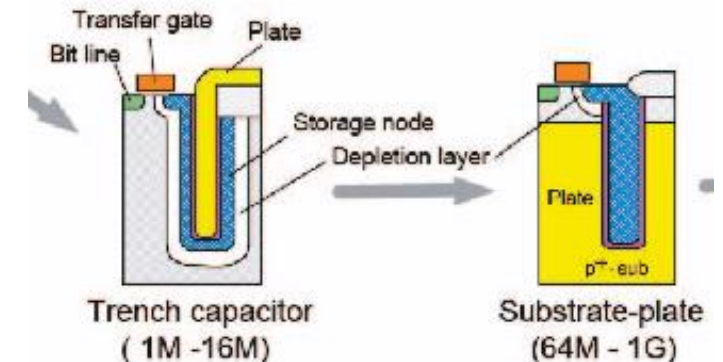
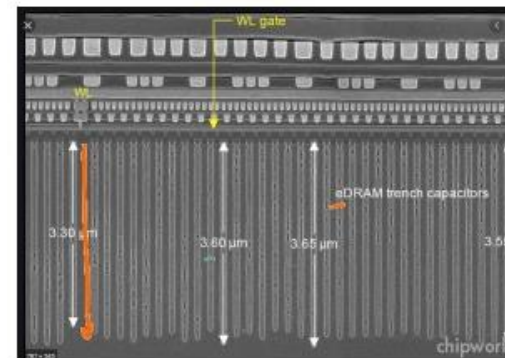
- **Stacked capacitor**

- Capacitor based on a special plate capacitor on top of the bit cell



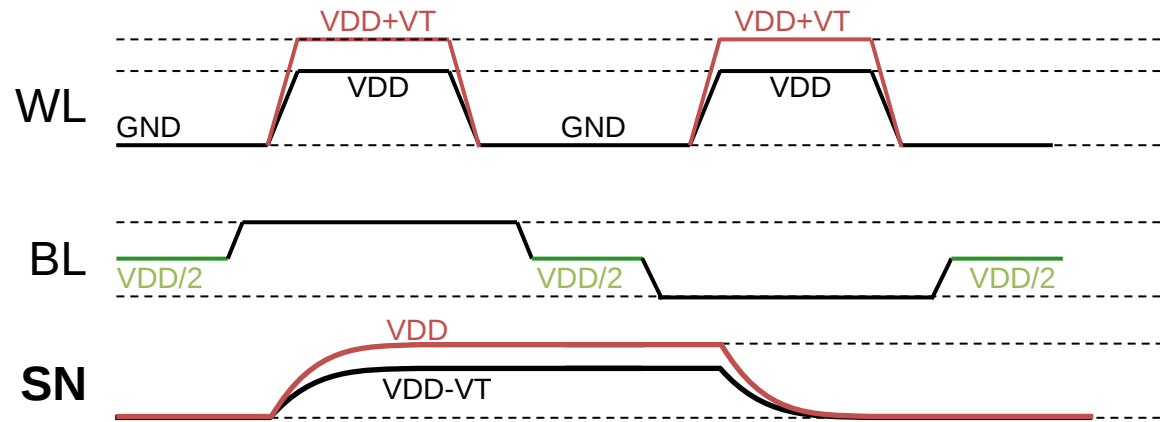
- **Trench capacitor**

- 3D capacitor located under the bit-cell
- Requires deep trenches which are tricky to fabricate
- Most often used today in dense DRAMs

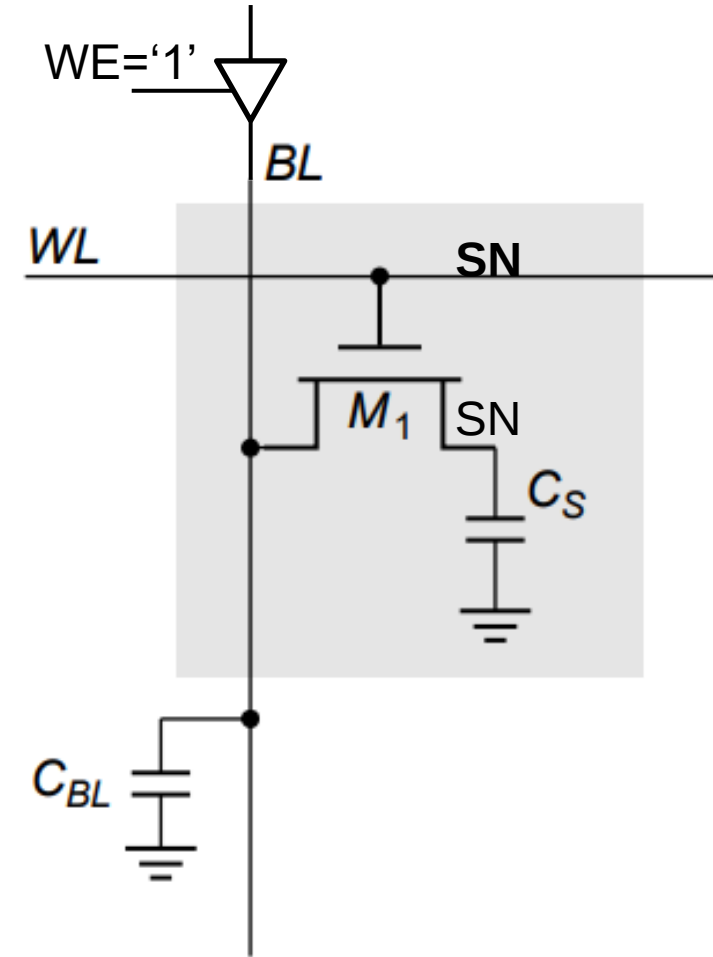


DRAM Read & Write Access

- **Write** and read through same access transistor
- **Write access:**



- Storage node (SN) capacitor C_S is charged/dis-charged according to BL when WL is asserted
- Writing a strong '1' can be achieved by over-driving the WL
- Note: between write access cycles, WL returns to an idle ($VDD/2$) state



DRAM Read & Write Access

- **Read** and write through same access transistor

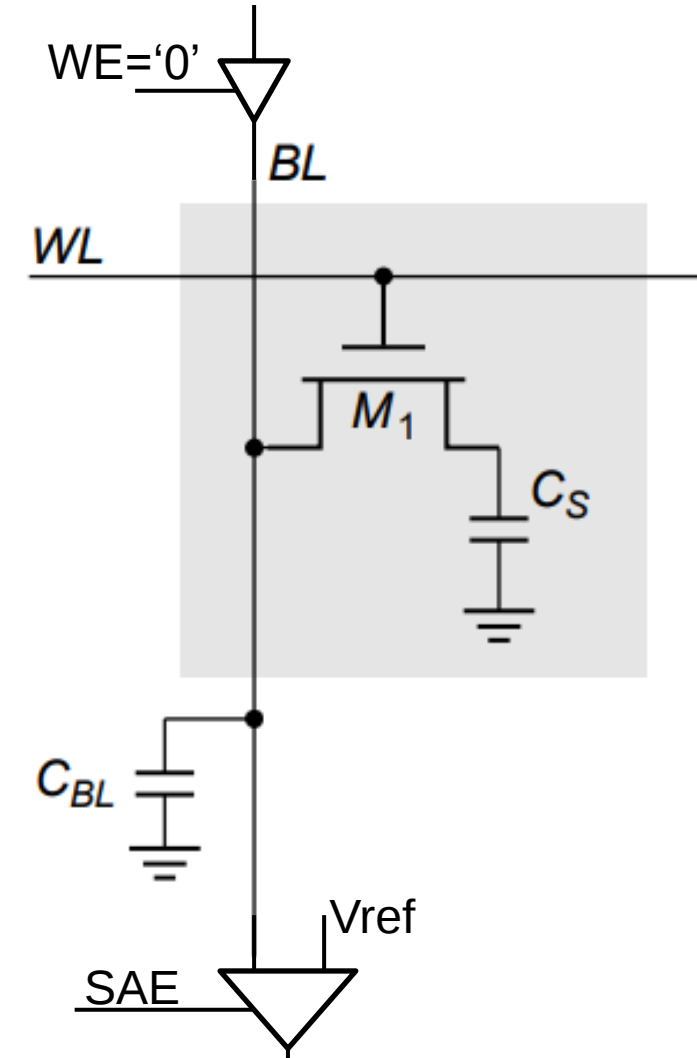
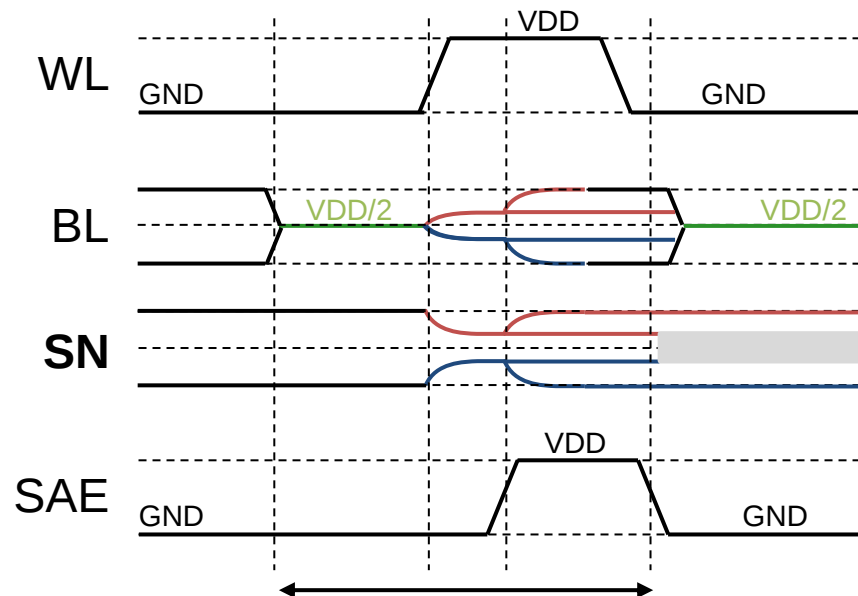
- **Read access:**

- Pre-charge BL to VDD/2
- WL is activated connecting the SN to the bit line BL
- Charge-sharing between C_{BL} and C_S

$$\Delta V_{BL} = (V_{SN} - V_{BL}) \frac{C_S}{(C_S + C_{BL})}$$

$$|\Delta V_{BL}| < \frac{(VDD/2) \cdot C_S}{(C_S + C_{BL})}$$

- Once BL develops a sufficient offset, the sense amplifier (SA) is enabled with SAE
 - Depending on the SA type, the SA feedback may pull the BL and the SN to '1' or '0' (restore the SN voltage)



DRAM Retention

- Between read and write accesses the DRAM is in **retention mode**
 - Leakage currents through **M1** degrade the SN level over time

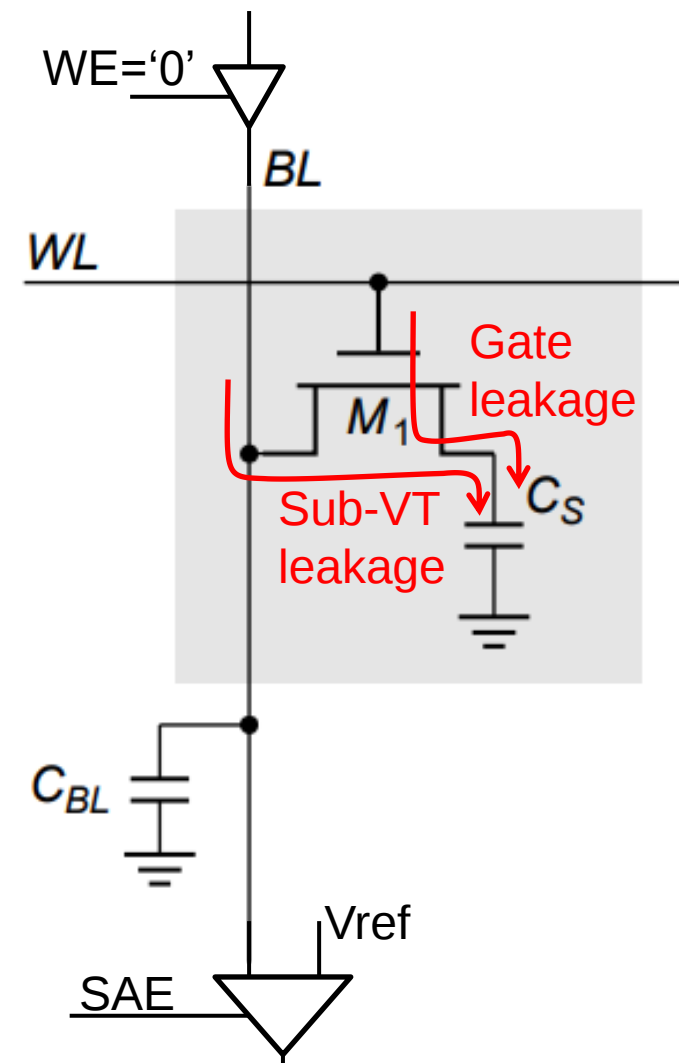
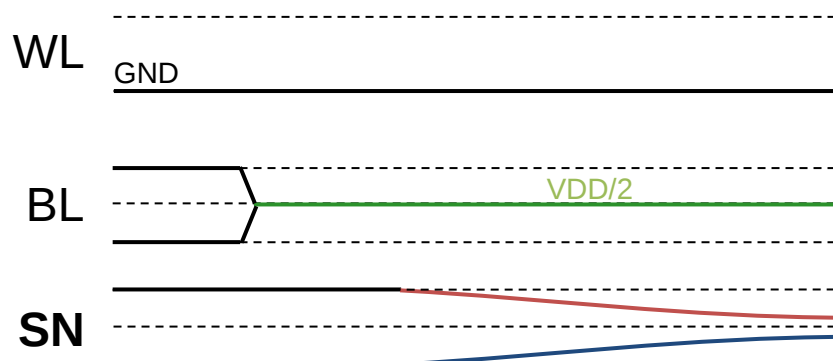
- Ideal retention:**
no read/no write

- BL is biased to balance both 0/1 retention leakage optimally
 - Not necessarily $V_{DD}/2$, depending on transistor characteristics

- Non-ideal retention:**

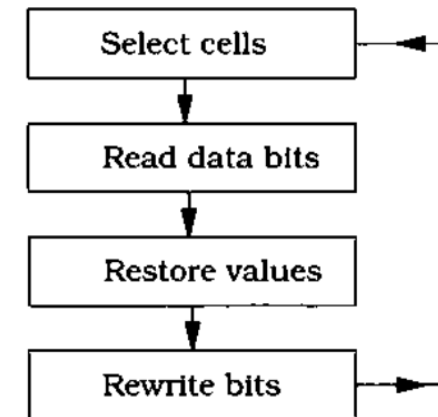
- BL is biased only between read- and write-access cycles
- During read and write: BL is temporarily at VDD or GND, leading to an overall slightly increased leakage

DRAM NEEDS REFRESH!!



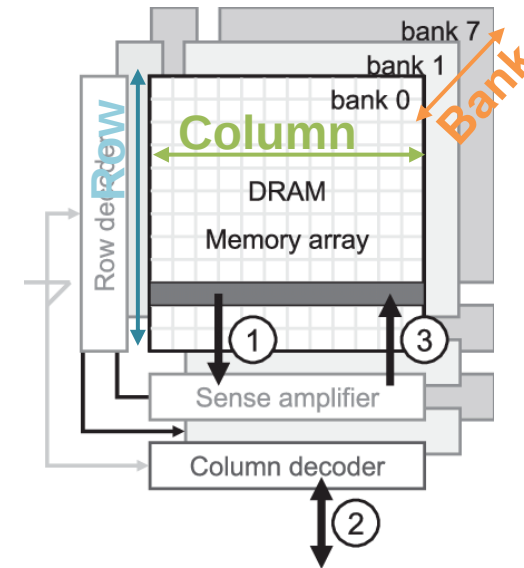
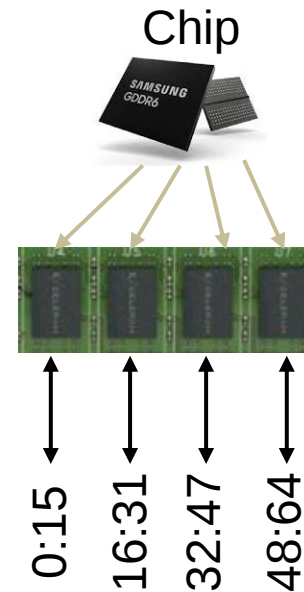
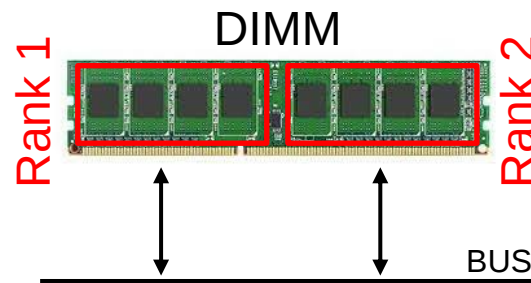
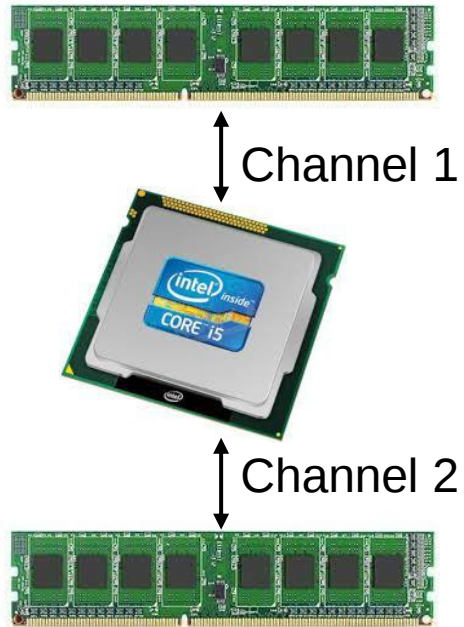
DRAM Refresh

- **Data retention time (DRT)** is determined by storage node capacitor and leakage currents
 - Depends on technology and temperature
 - Typical DRT for dedicated DRAM chips are in the order of tens of milliseconds
- **Dynamic memories (DRAM) need refresh**
 - Refresh = reading data, restoring logic levels and writing data back to the storage cells
 - Refresh rate depends on the DRT
 - Every refresh cycle refreshes one entire memory row
- **Refresh overhead depends on the retention time and the number of rows**



DRAM Organization

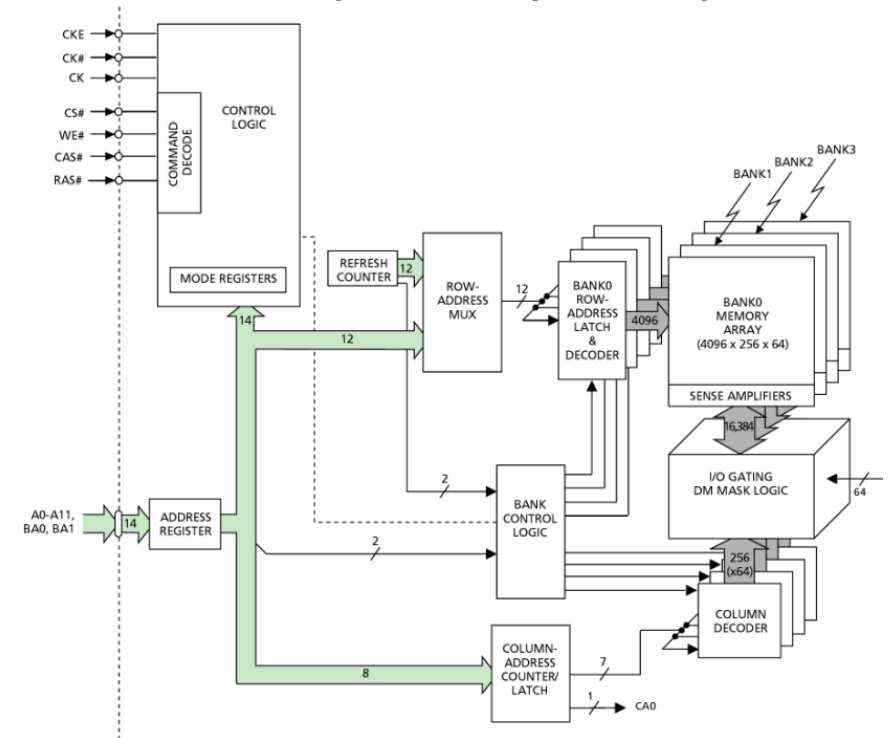
- Large DRAM memories are organized hierarchically



Channels
DIMMs
Ranks
Chips
Banks
Rows
Columns

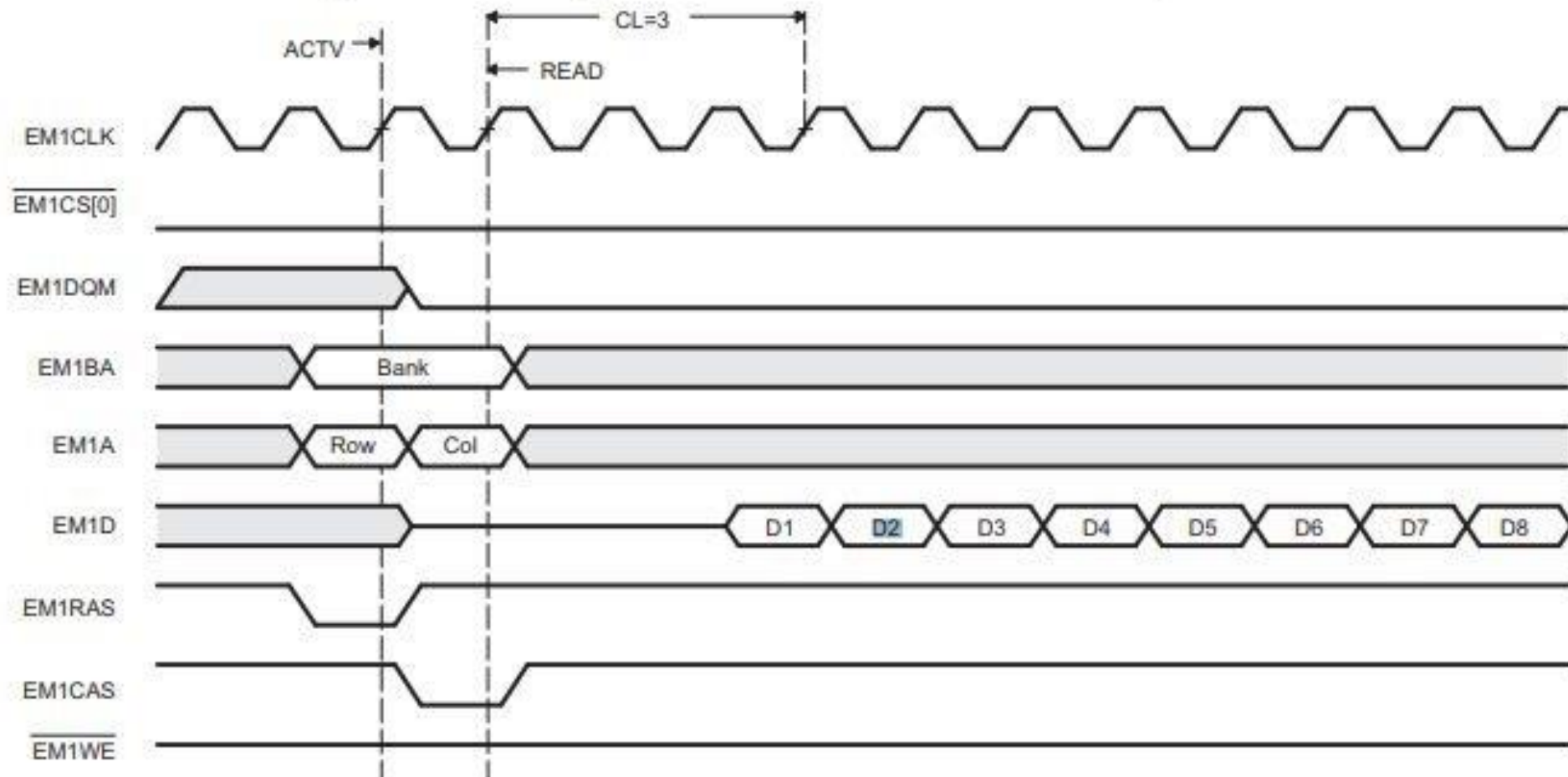
DRAM Access

- **DRAM access is broken into several steps, based on the structure**
 - Row access is slow due to the difficult sensing procedure
 - Each row contains a large number of bits (fewer rows allow to refresh many bits in parallel)
 - Destructive read-access: each row access requires restore
- **Access procedure:**
 - Row access: provide row address and read data from storage array into the row buffer (latches)
 - Column access: provide column address and read or write data from/to row buffer
 - Precharge: write row buffer back to storage array and prepare for next read by precharging the BLs
- **Row access is expensive: often followed by bursts from the row buffer**



DRAM Access Example (Burst Read)

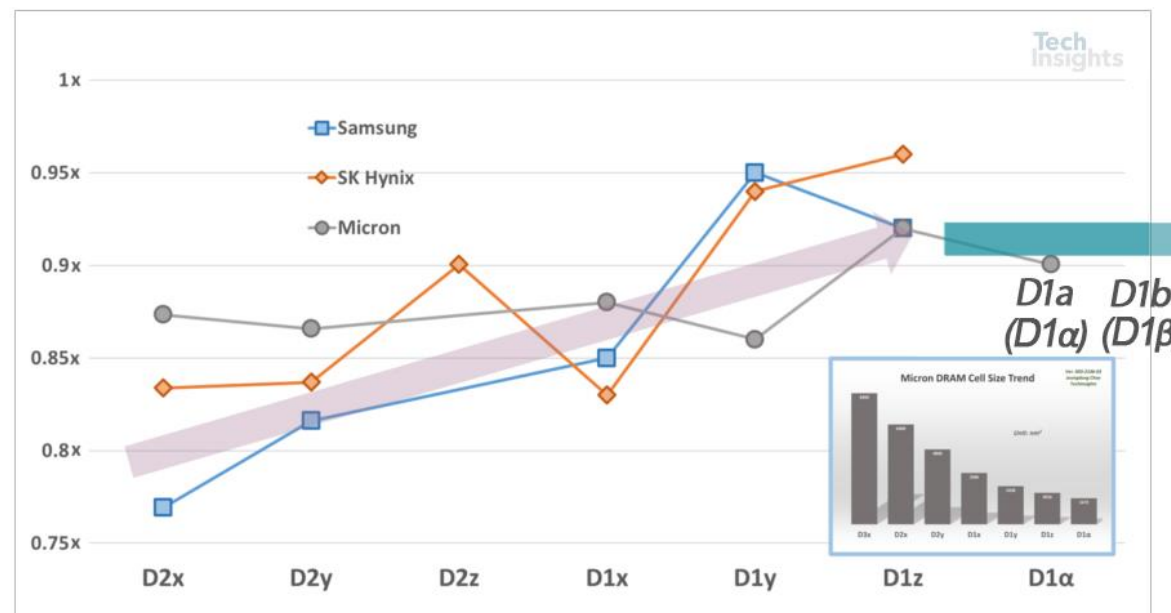
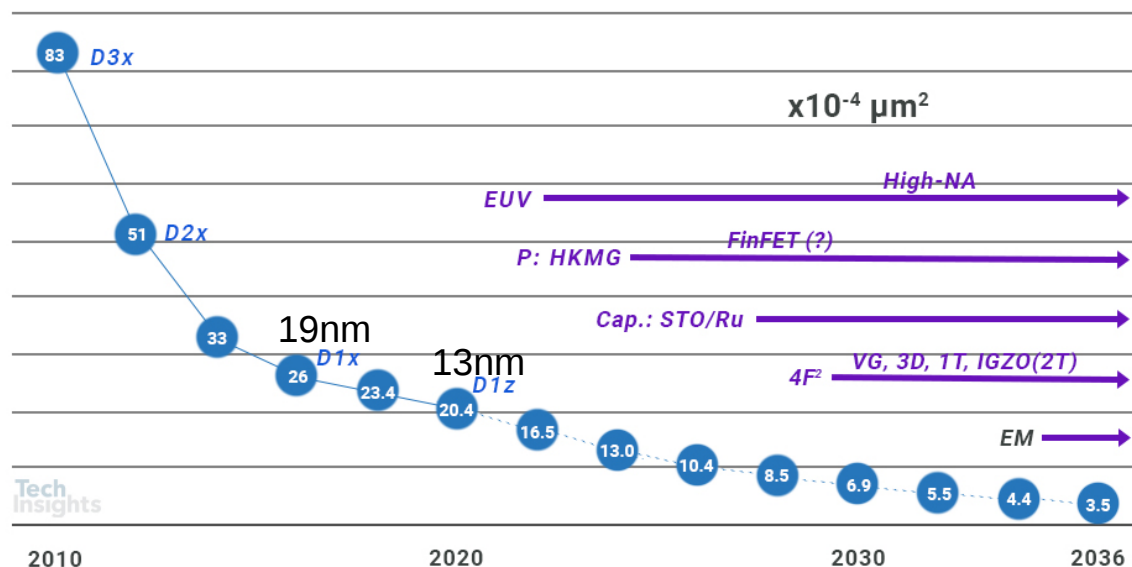
Figure 25-6. Timing Waveform for Basic SDRAM Read Operation



DRAM Scaling Trends

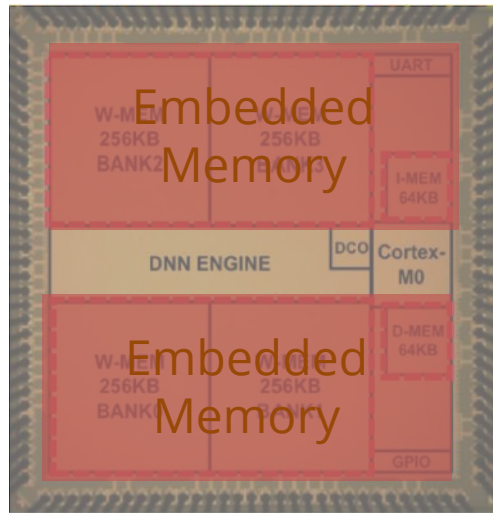
- **DRAM technologies behind CMOS** in feature size (today: ~10-12nm, BULK) & **scaling is slowing down significantly**
 - Main issue: shrinking cell capacitor area while maintaining its capacitance (~7f F)
 - Use of isolation materials with very high dielectric constants ($K > 50$)
 - Other concern: keeping access transistor leakage low and compatibility with trench capacitors

DRAM Cell Size Trend & Prediction

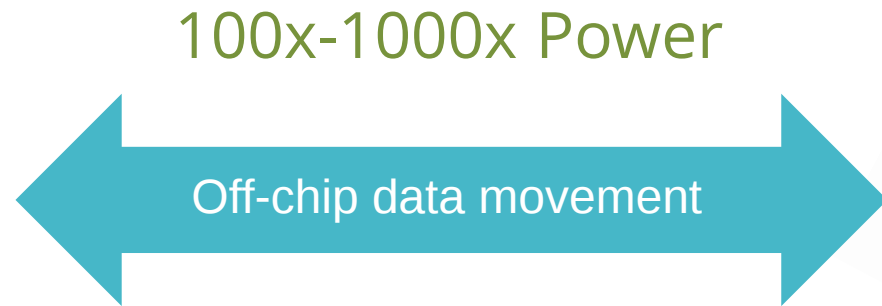


<https://www.techinsights.com/blog/dram-scaling-trend-and-beyond>

External DRAM is a Bottleneck



SoC



100x-1000x Lower Bandwidth

Significantly higher BOM and
3rd party dependencies



External DRAM

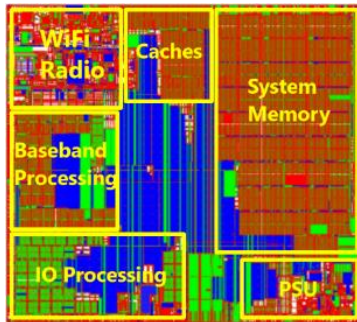
**External memory should be avoided at all costs and if needed,
access should be minimized**

Memory is the Limiting Factor

Memories are the limiting factor for cost and energy

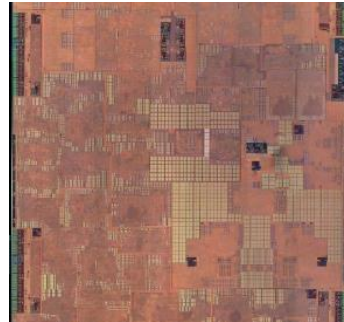
- On-chip memories have a poor area density and often dominate chip area and cost in many computing systems
- Memory often accounts for >50% of the active power and for 100% of the power during sleep/standby periods in low-power systems

IoT & MCU



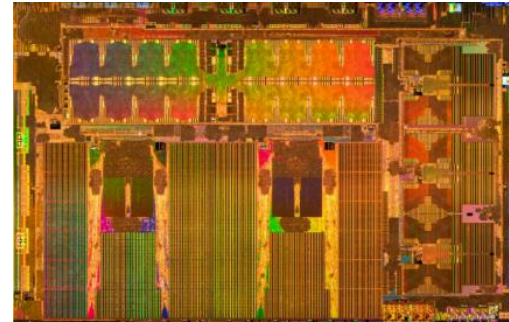
MediaTek MT3620, 40nm

Mobile



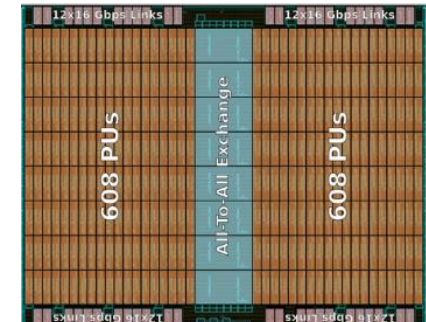
Apple A11, 10nm

Automotive



Tesla FSD, 14nm

ML/AI & Server



Graphcore IPU, 16nm

SRAM
Area [%]

35%

31%

36%

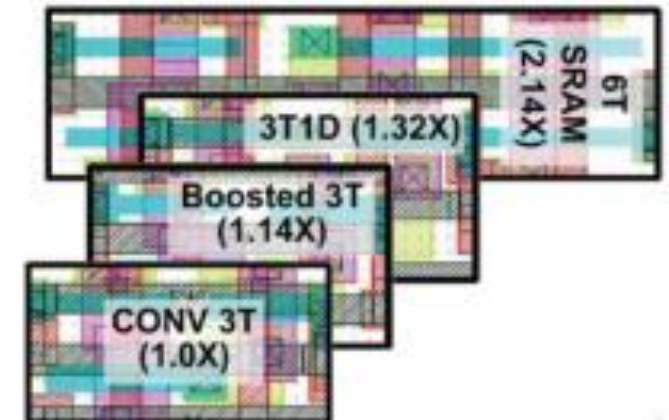
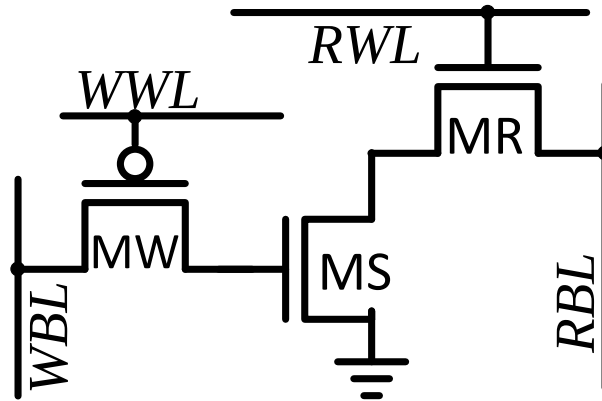
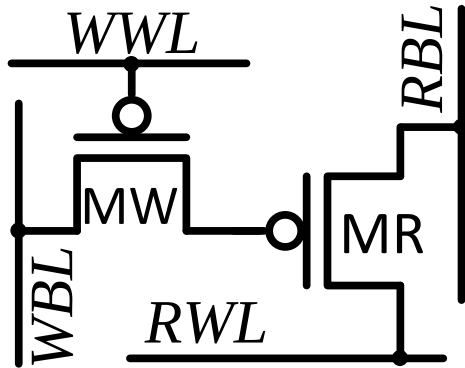
75%



Gain Cell Embedded DRAM (eDRAM)

- **DRAM on a standard CMOS process**

- Storage capacitor is a parasitic capacitor (gate capacitance + other parasitics)
- 1 access transistor for write
- 1-2 access transistors for read



2T and 3T Gain Cell with $\sim 70F^2$ per bit

2T Gain Cell eDRAM: Basic Operating Principle

- **Write port (WWL & WBL)**, storage cap, and **read port (RWL & RBL)**

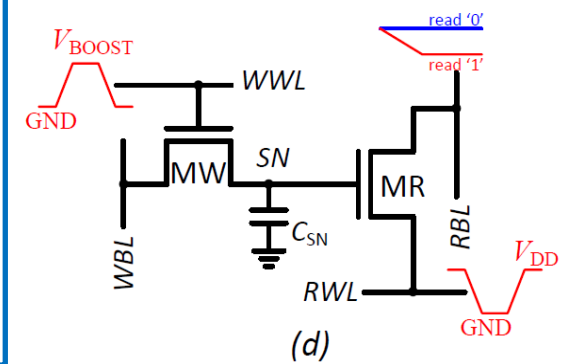
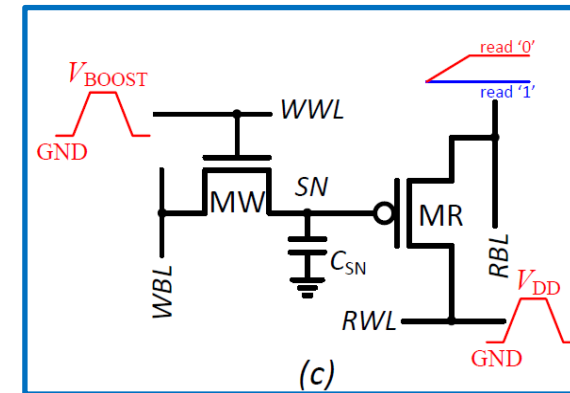
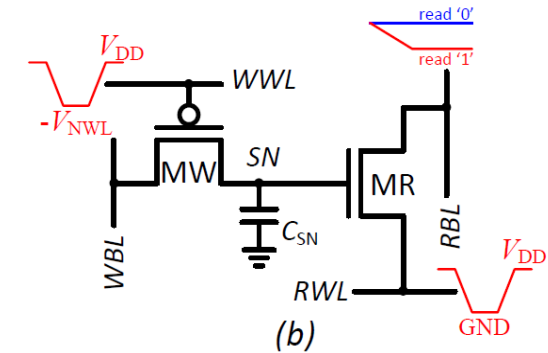
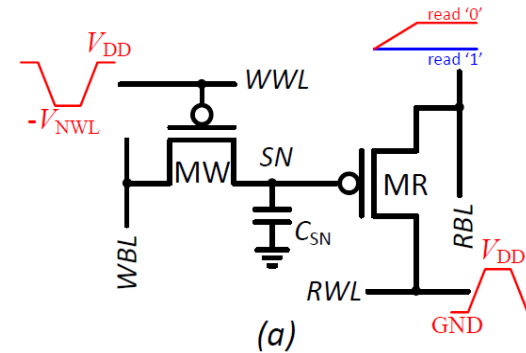
- Different combinations of PMOS and NMOS transistors
- Use of different threshold options

- **Write operation:**

- Boosted WWL, above V_{DD} for NMOS, below V_{SS} for PMOS

- **Read:**

- PMOS MR: Pre-discharge RBL, raise RWL $\rightarrow$ SN='0': RBL rises
- NMOS MR: Precharge RBL, lower RWL

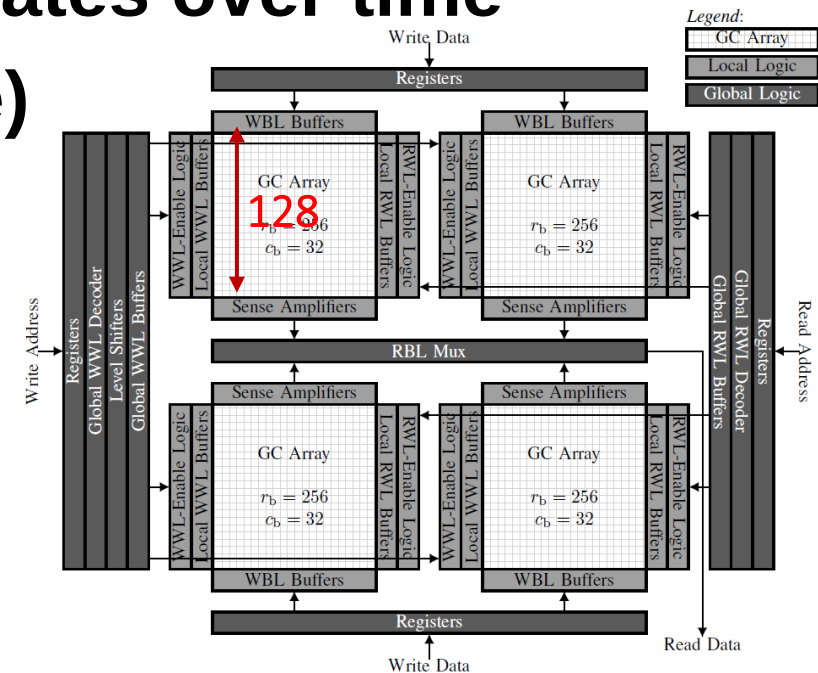


Gain Cell eDRAM Periodic Refresh

- **Dynamic storage mechanism: data deteriorates over time**
- **Need for periodic refresh cycles (read/write)**
 - Data arranged in sub-arrays
 - Parallel refresh in all sub-arrays
- **Array availability**

$$Availability [\%] = 1 - \frac{T_{clk}}{T_{ret}} N_r$$

- Typical retention times: $T_{ret} = 100\mu s - 1ms$
- Typical access/refresh cycle-time: $T_{clk} = 10ns$
- Typical sub-array size $N_r = 128-256$ rows



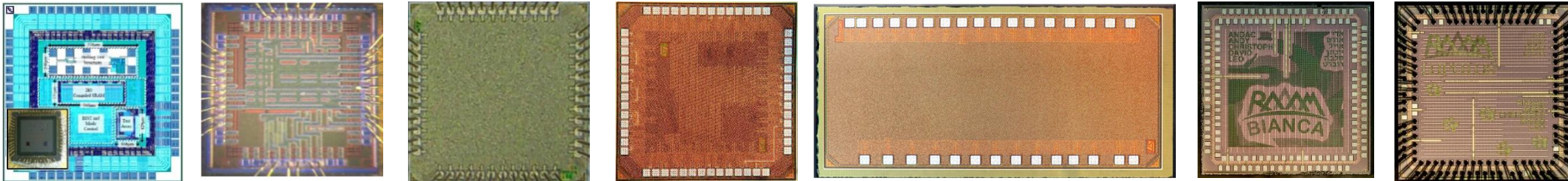
Typical array
availability: $\sim 98\%$

Delivering the Highest Density Volatile Embedded Memories in Standard CMOS

Reduced Cost | Longer Battery-Life | Better Performance



Looking for Engineers and Interns



Microphotographs of RAAAM's GCRAM Technology Implementations in 16nm – 180nm Processes



Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra
Swiss Confederation

Innosuisse – Swiss Innovation Agency

EPFL INNOGRANTS

Vice-Présidence pour l'innovation et la valorisation

BRIDGE

>>venture>>

Companies for tomorrow

Si

SILICON
CATALYST

PORTFOLIO
COMPANY