Digital Integrated Circuits

(83-313)

Lecture 7:

SRAM

Semester B, 2016-17

Lecturer: Dr. Adam Teman

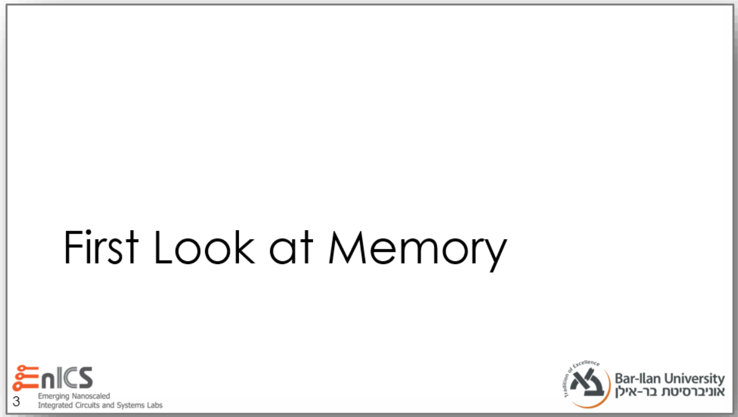
TAs: Itamar Levi, Robert Giterman

16 May 2017

EnICS Emerging Nanoscaled Integrated Circuits and Systems Labs

Tradition of Excellence Bar-Ilan University אוניברסיטת בר-אילן

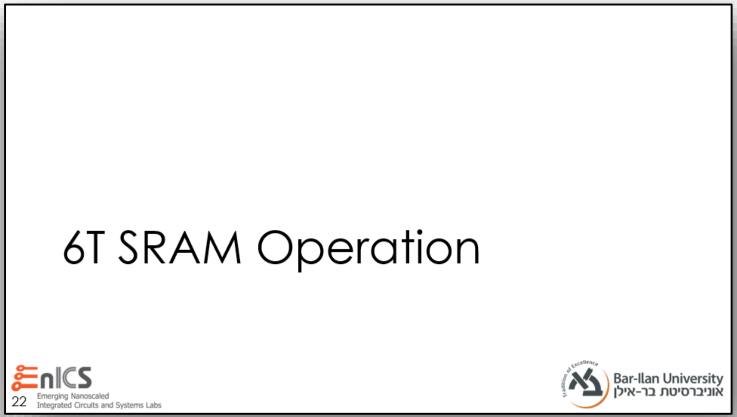
Lecture Content


First Look at Memory

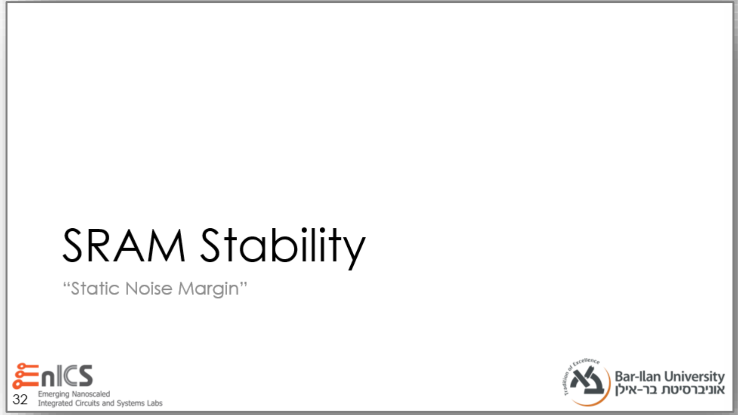
3

The 6T SRAM Bitcell

11

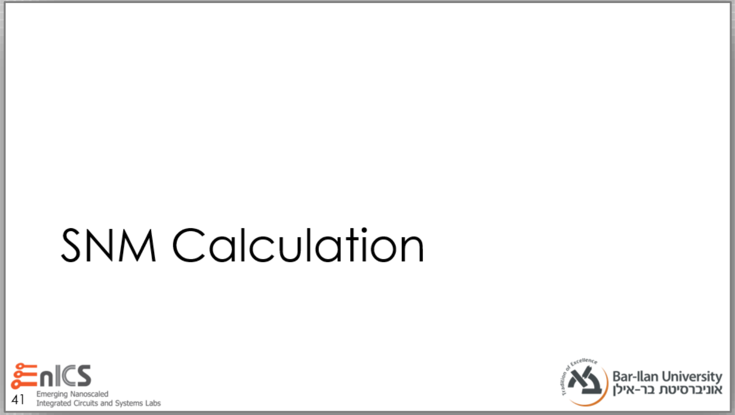

6T SRAM Operation

22


SRAM Stability

"Static Noise Margin"

32

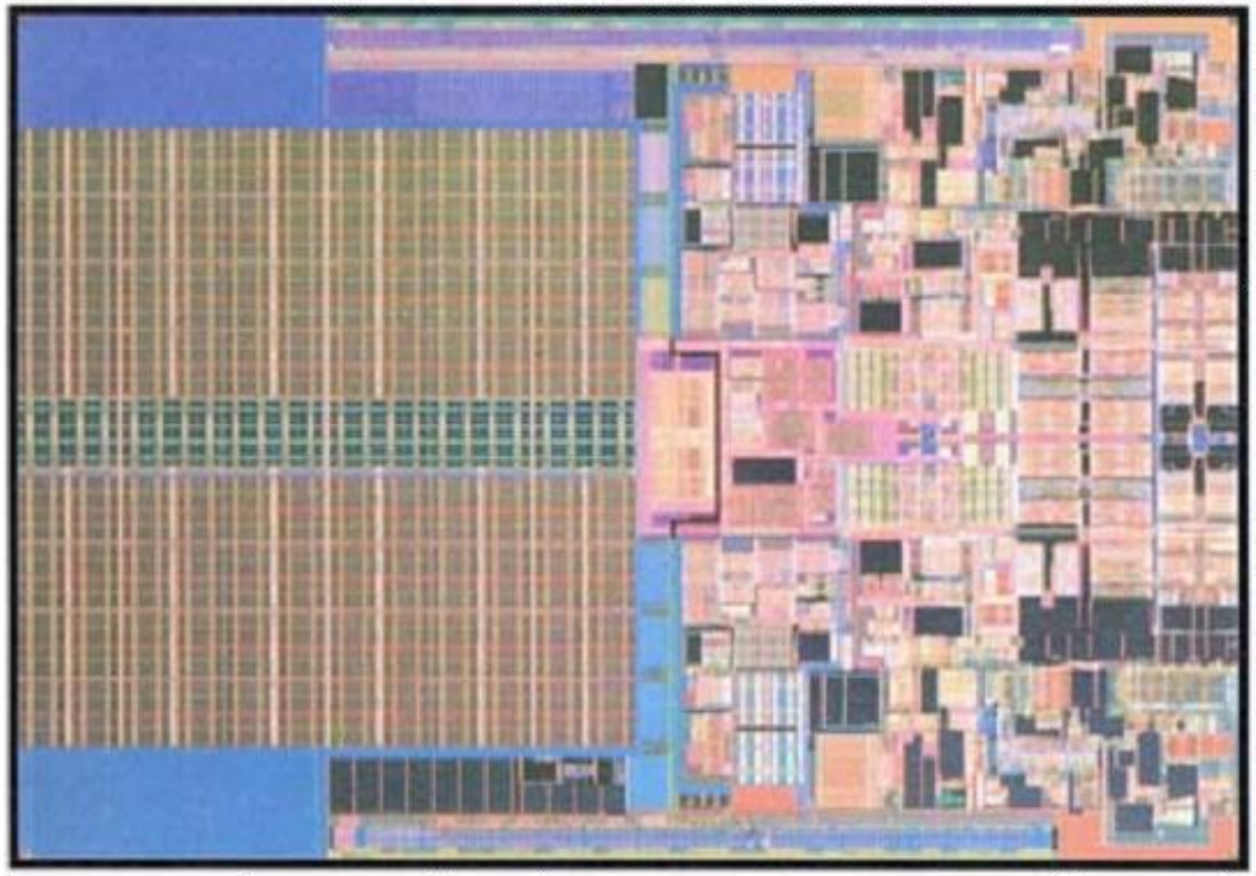

SNM Calculation

41

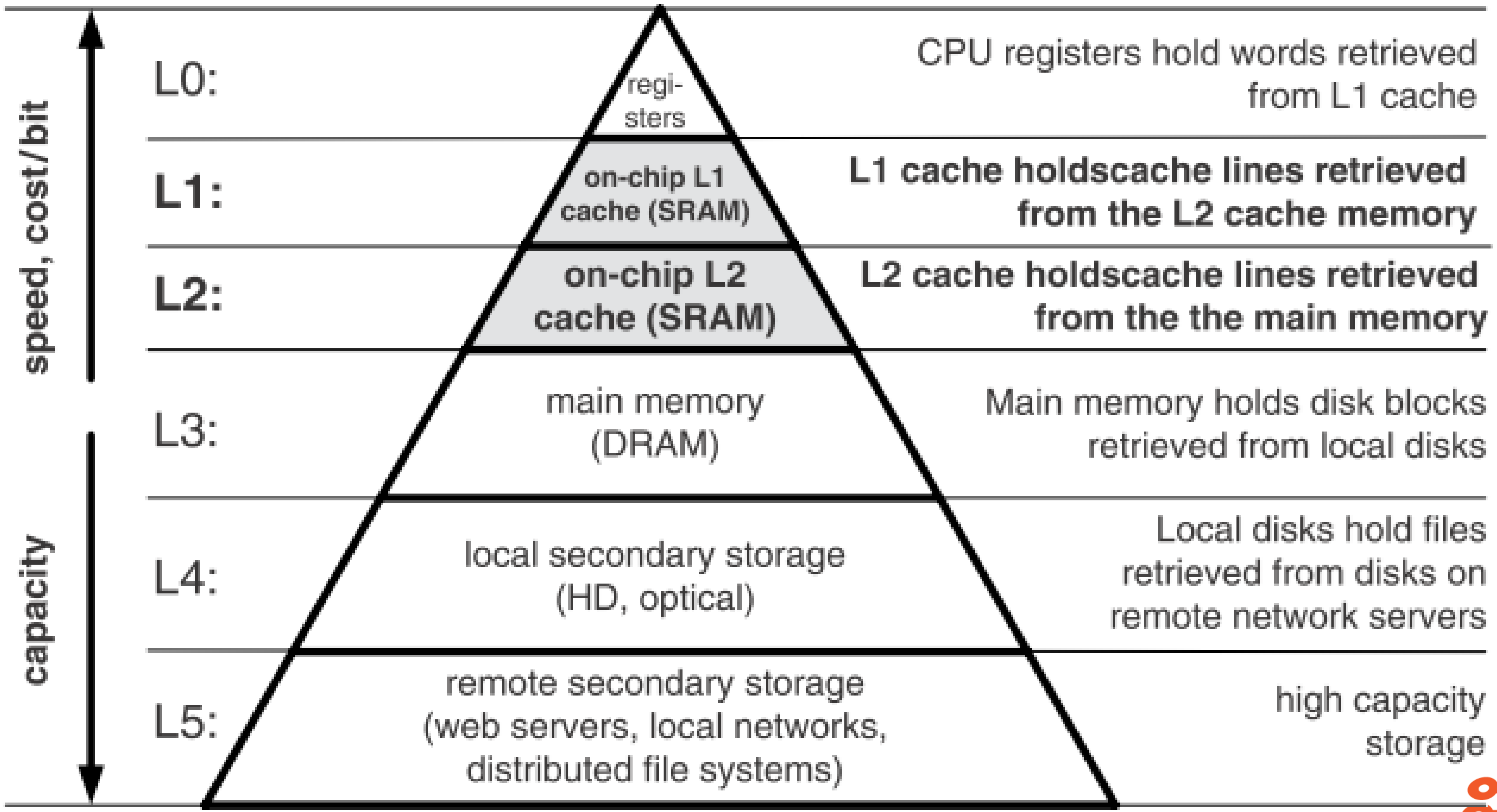
First Look at Memory

Why Memory?

Intel 45nm Core 2



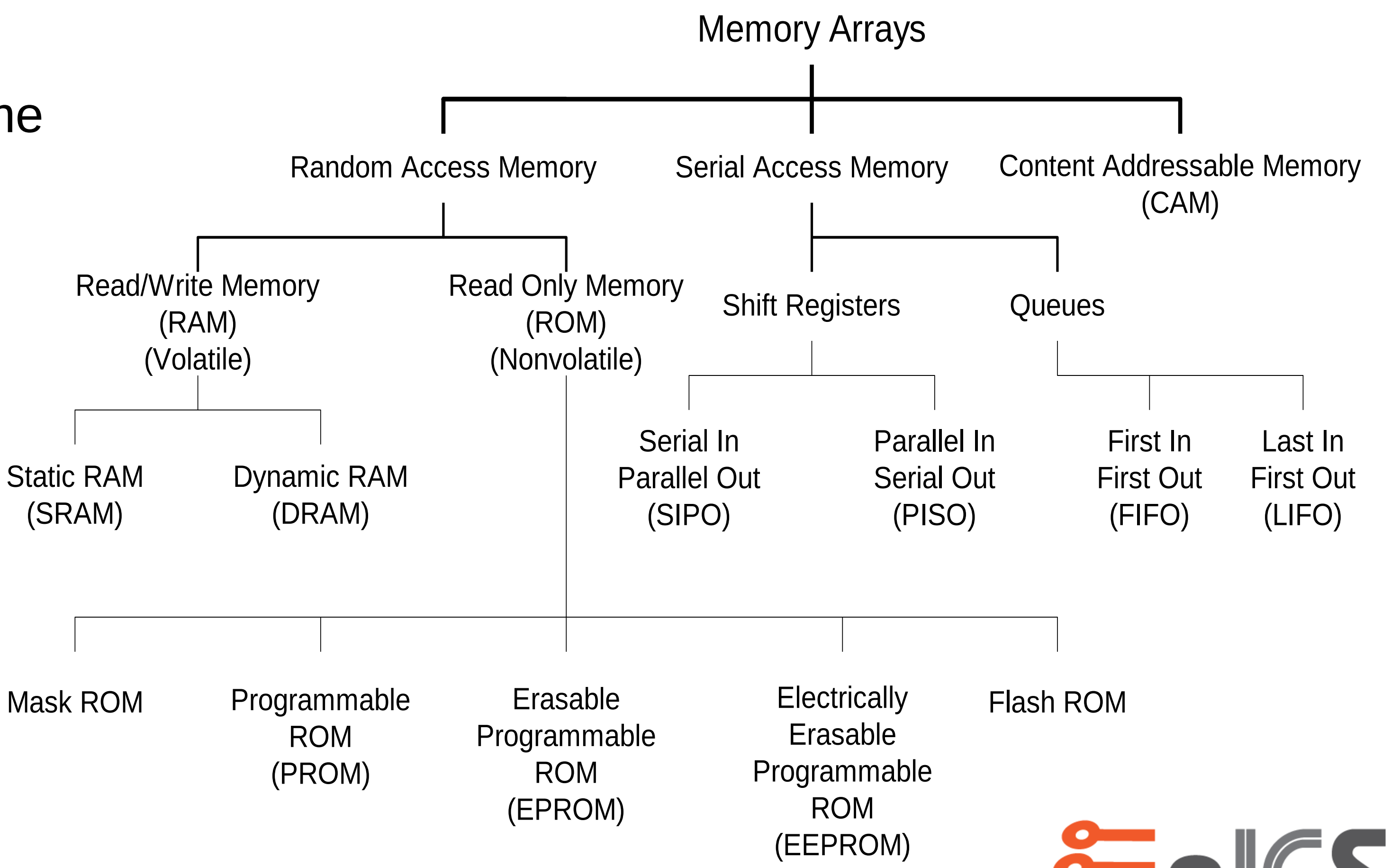
Memory Hierarchy of a Personal Computer


speed, cost/bit
capacity
L0:
regi-
sters
CPU registers hold words retrieved from L1 cache
L1:
on-chip L1 cache (SRAM)
L1 cache holdscache lines retrieved from the L2 cache memory
L2:
on-chip L2 cache (SRAM)
L2 cache holdscache lines retrieved from the the main memory
L3:
main memory (DRAM)
Main memory holds disk blocks retrieved from local disks
L4:
local secondary storage (HD, optical)
Local disks hold files retrieved from disks on remote network servers
L5:
remote secondary storage (web servers, local networks, distributed file systems)
high capacity storage

From Pavlov, Sachdev, 2008

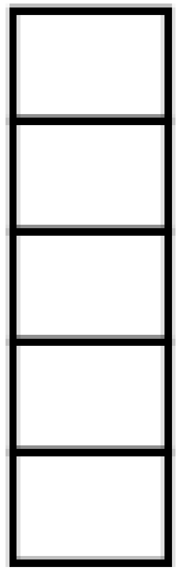
Semiconductor Memory Classification

- **Size:**
  - Bits, Bytes, Words
- **Timing Parameters:**
  - Read access, write access, cycle time
- **Function:**
  - Read Only (ROM) – non-volatile
  - Read-Write (RWM) – volatile
  - NVRWM – Non-volatile Read Write
- **Access Pattern:**
  - Random Access, FIFO, LIFO, Shift Register, CAM
  - I/O Architecture:
  - Single Port, Multi-port
- **Application:**
  - Embedded, External, Secondary


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

Random Access Chip Architecture

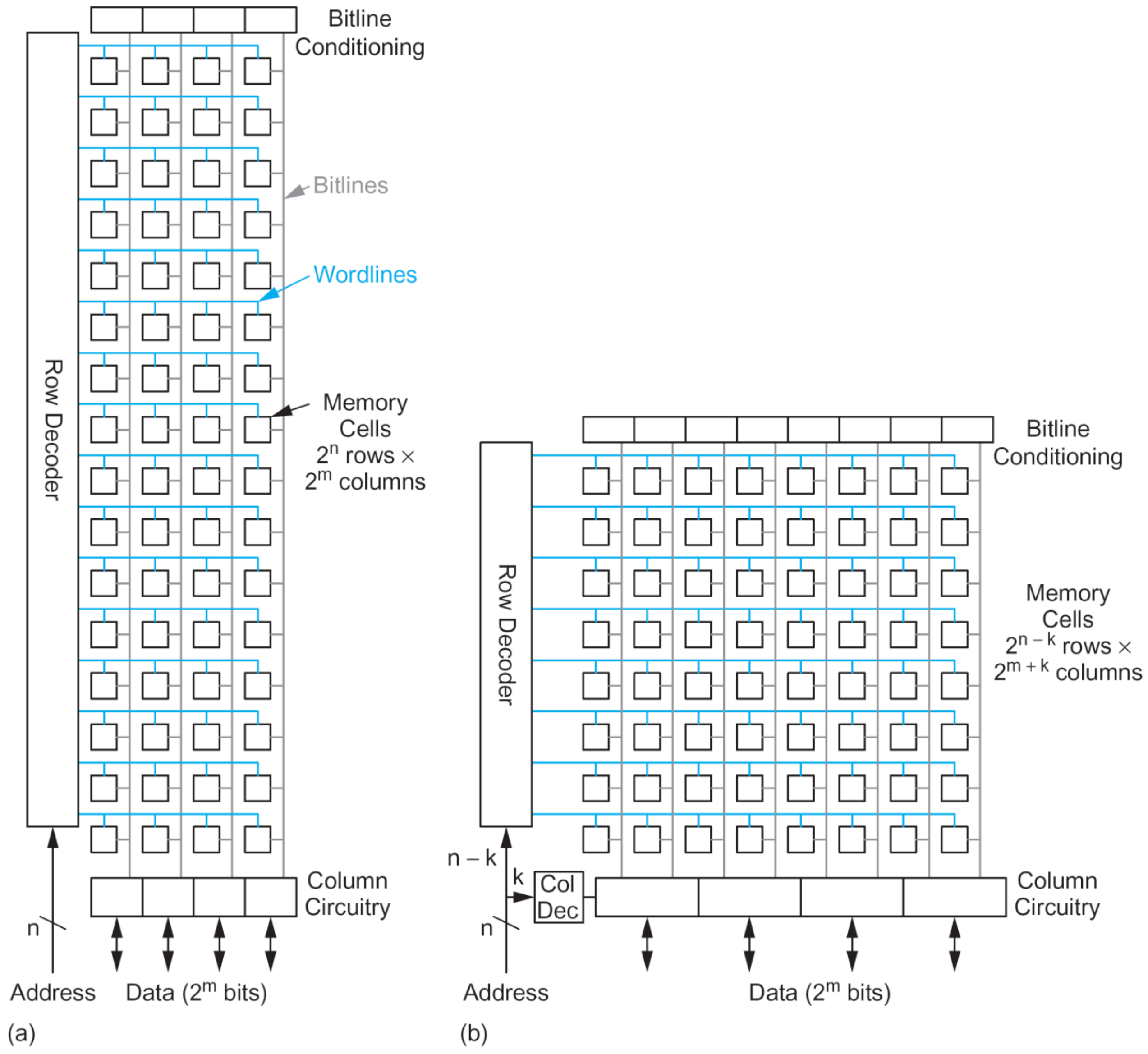
- **Conceptual: linear array**
  - Each box holds some data
  - But this leads to a long and skinny shape



- **Let's say we want to make a 1MB memory:**
  - 1MB=2^{20} words X 8 bits=2^{23} bits, each word in a separate row
  - A decoder would reduce the number of access pins from 2^{20} access pins to 20 *address* lines.
  - We'd fit the pitch of the decoder to the word cells, so we'd have *Word Lines* with no area overhead.
  - The output lines (=*bit lines*) would be extremely long, as would the delay of the huge decoder.
  - The array's height is about 128,000 times larger than its width ($2^{20}/2^{3}$).

Square Ratio

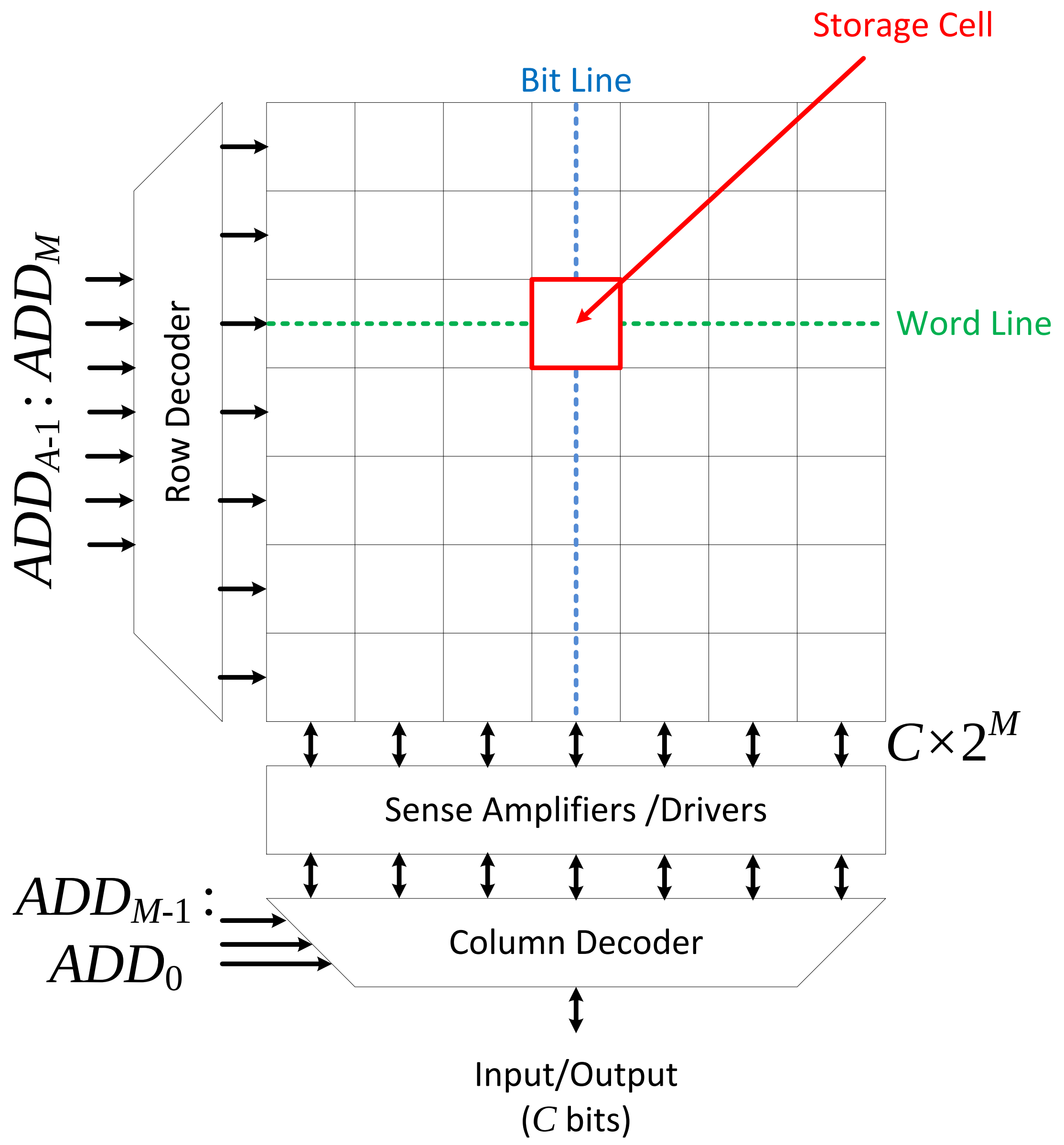
- **Instead, let's make the array square:**
  - 1MB=2^{23} bits=2^{12} rows X 2^{11} columns.
  - There are 4000 rows, so we need a 12-bit row address decoder (to select a single row)
  - There are 2000 columns, representing 256 8-bit words.
  - We need to select only one of the 256 words through a column address decoder (or multiplexer).
  - We call the row lines "*Word Lines*" and the column lines "*Bit Lines*".



Special Considerations

- **The "*core*" of the memory array is huge.**
  It can sometimes take up most of the chip area.
  - For this reason, we will try to make the "*bitcell*" as small as possible.
  - A standard Flip Flop uses at least 10 transistors per bit (usually more than 20). This is very area consuming.
- **We will trade-off area for other circuit properties:**
  - Noise Margins
  - Logic Swing
  - Speed
  - Design Rules
- **This requires special peripheral circuitry.**

Memory Architecture



Memory Size: W Words of C bits

$= W \times C$ bits

Address bus: A bits

$\rightarrow W = 2^A$

Number of Words in a Row: 2^M

Multiplexing Factor: M

Number of Rows: 2^{A-M}

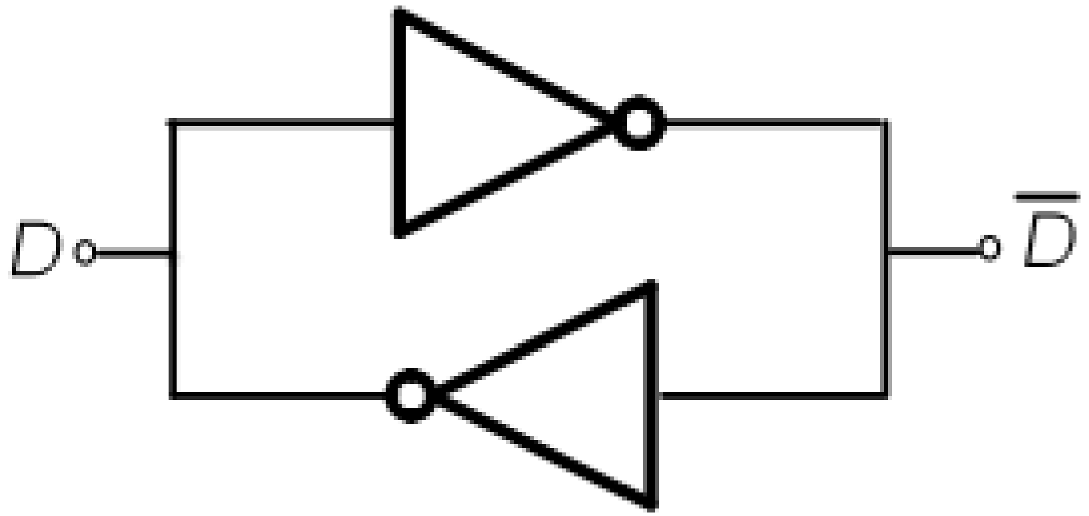
Number of Columns: $C \times 2^M$

Row Decoder: $A\text{-}M \rightarrow 2^{A-M}$

Column Decoder: $M \rightarrow 2^M$

The 6T SRAM Bitcell

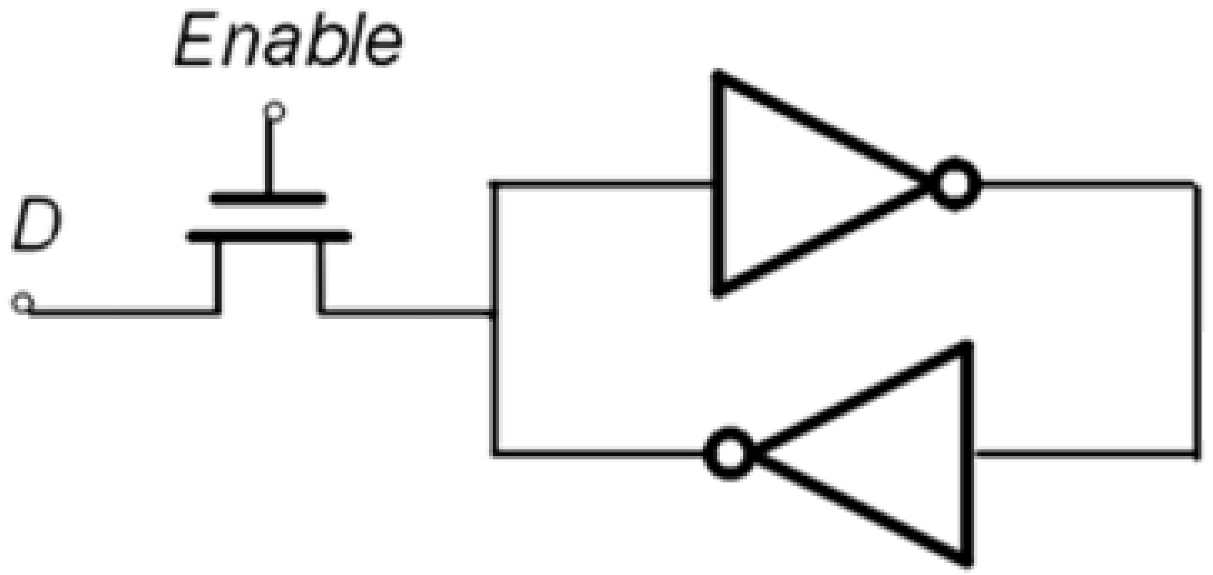
Basic Static Memory Element



Positive Feedback: Bi-Stability

Writing into a Cross-Coupled Pair

- **The write operation is ratioed**
  - The access transistor must overcome the feedback.



How should we write a ‘1’

Option 1: nMOS Access Transistor

Passes a “weak ‘1’”, bad at pulling up against the feedback

Option 2: pMOS Access Transistor

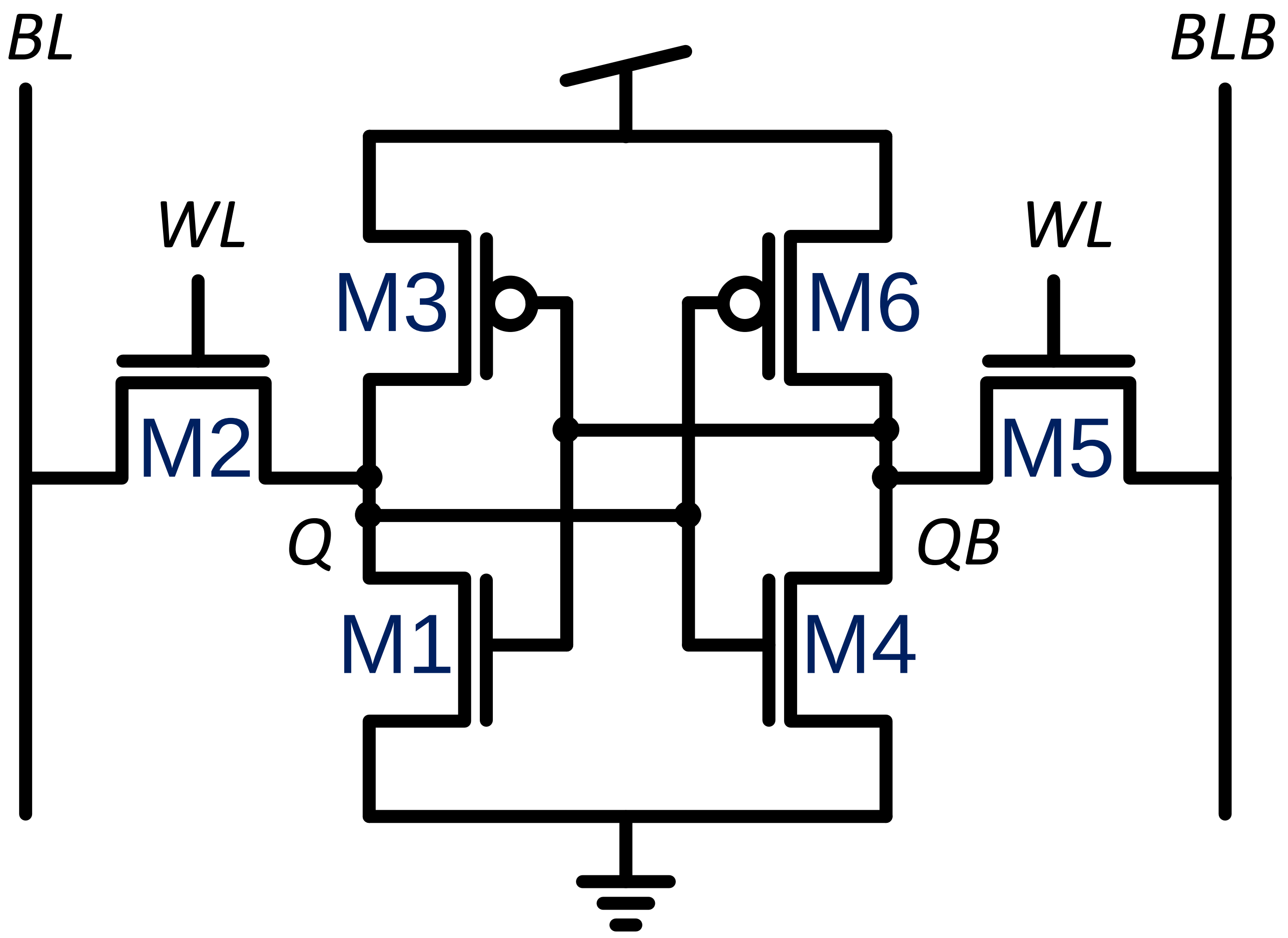
Passes a “weak ‘0’”, bad at pulling down against the feedback

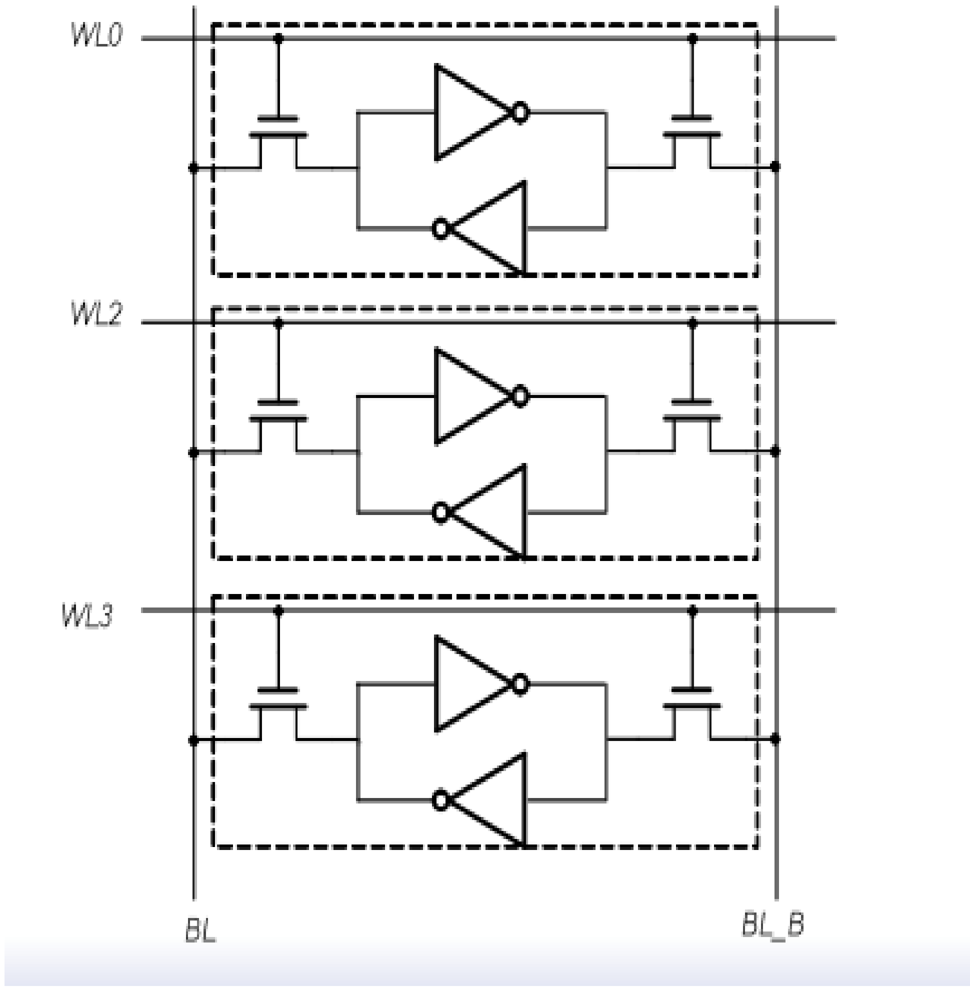
Option 3: Transmission Gate

Writes well, but how do we read?

Solution: Differential nMOS Write

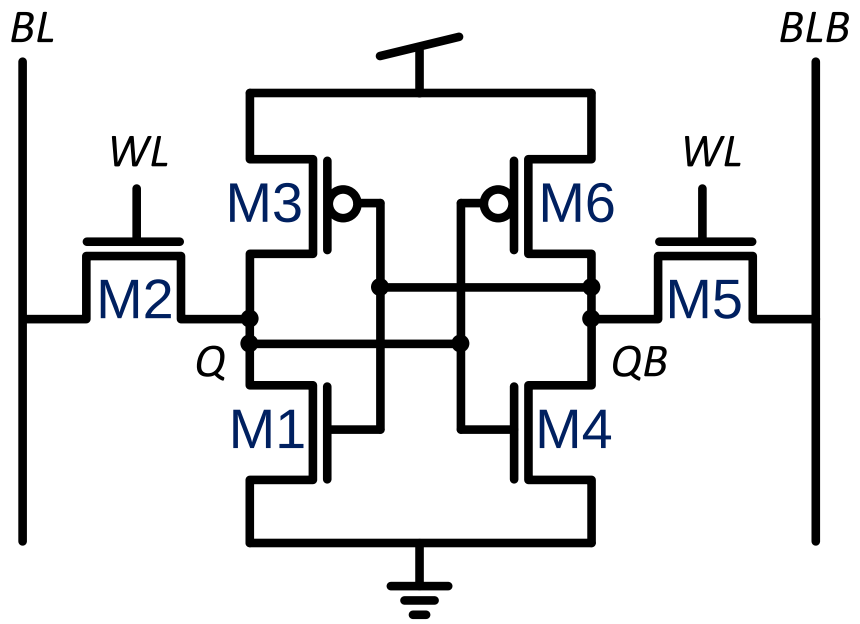
6-transistor CMOS SRAM Cell

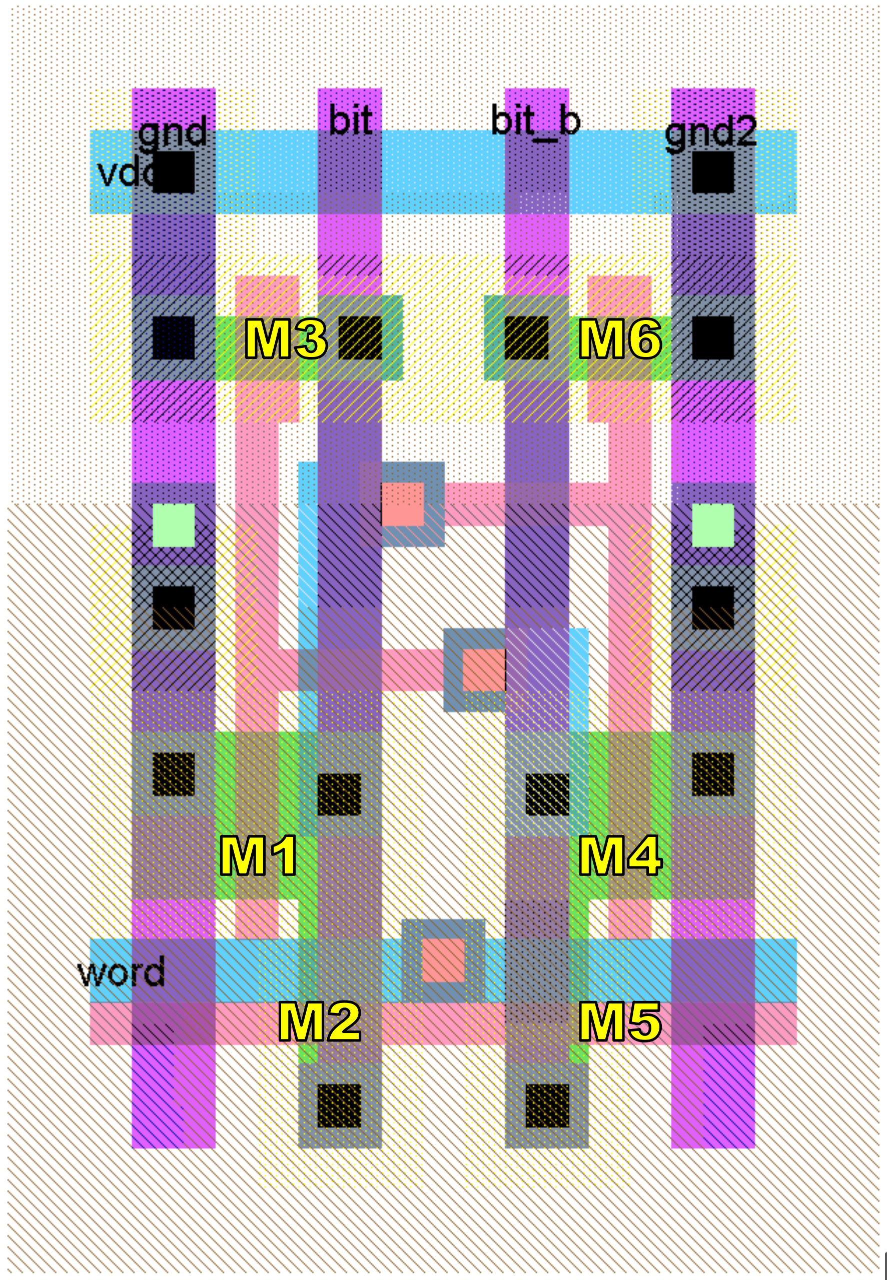




SRAM Layout - Traditional

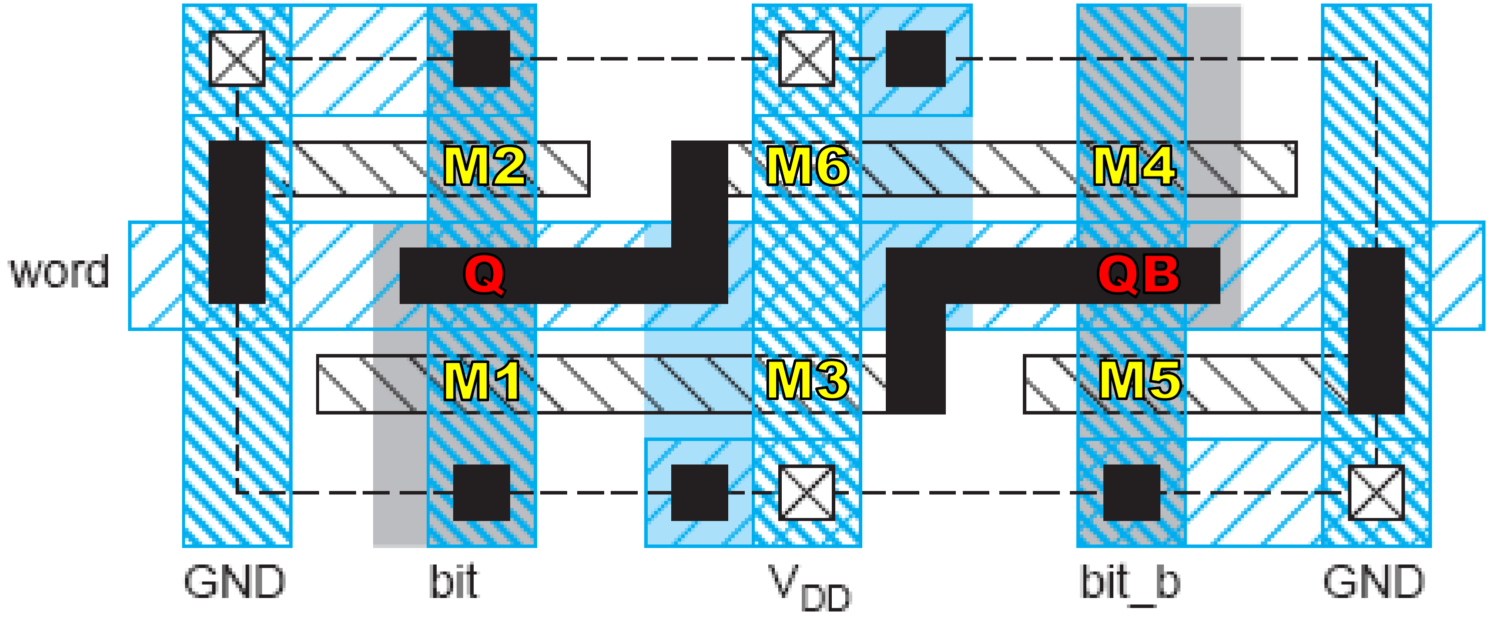
- Share Horizontal Routing (WWL).
- Share Vertical Routing (BL, BLB).
- Share Power and Ground.

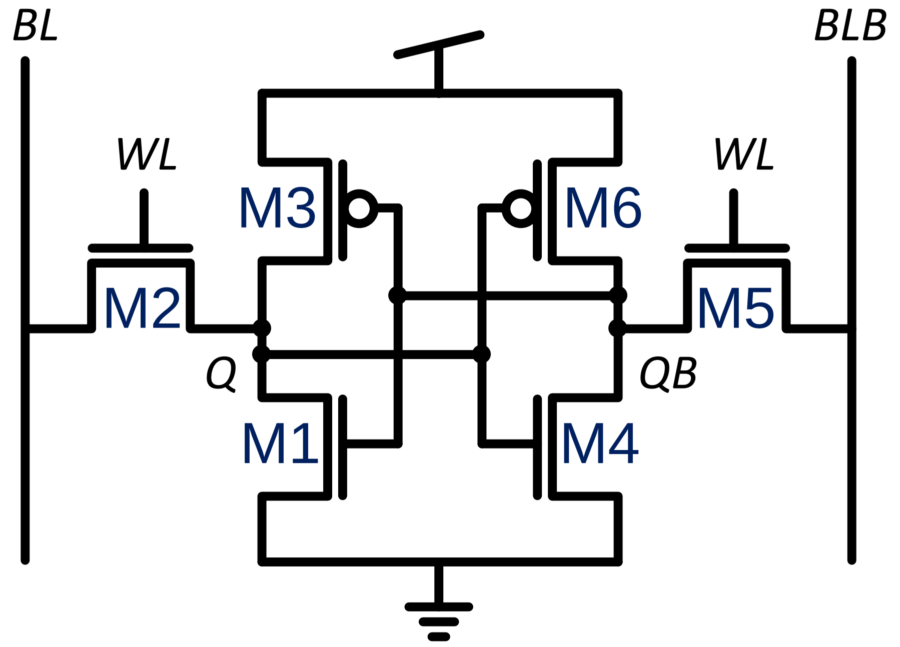




SRAM Layout – Thin Cell

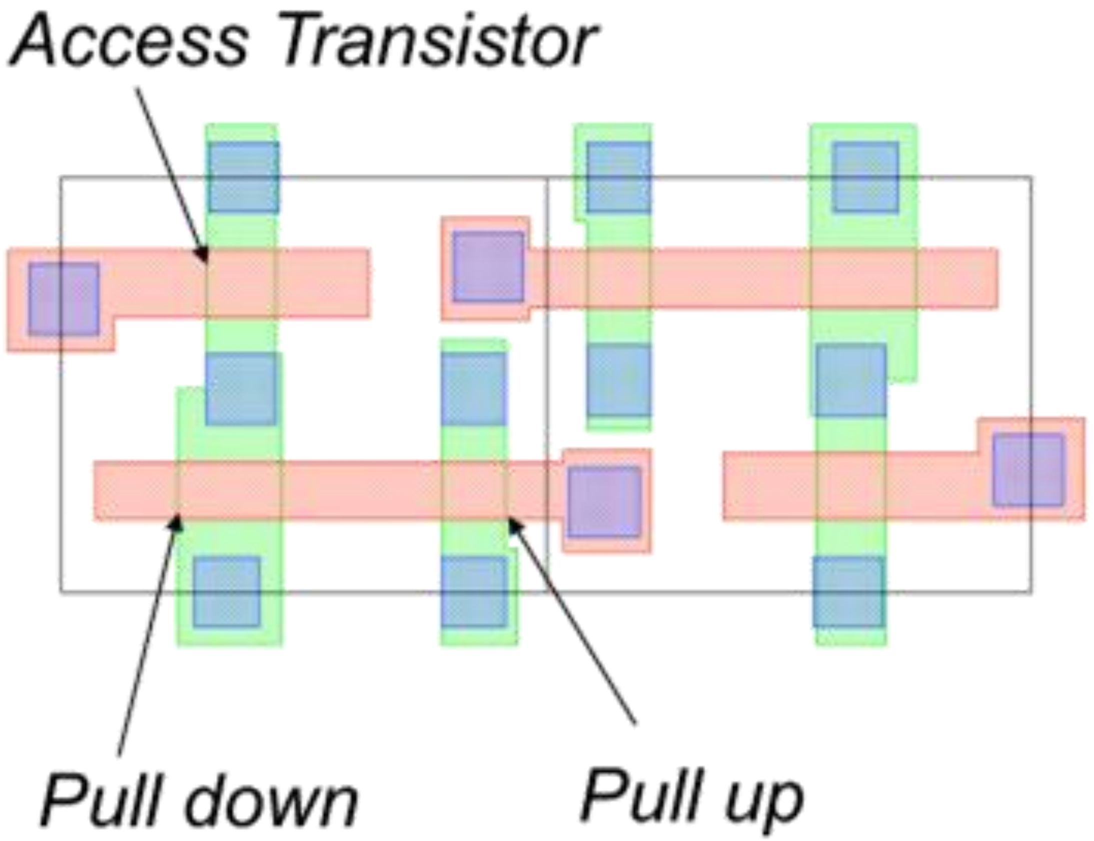
- **Avoid Bends in Polysilicon and Diffusion**
- **Orient all transistors in one direction.**
- **Minimize Bitline Capacitance.**

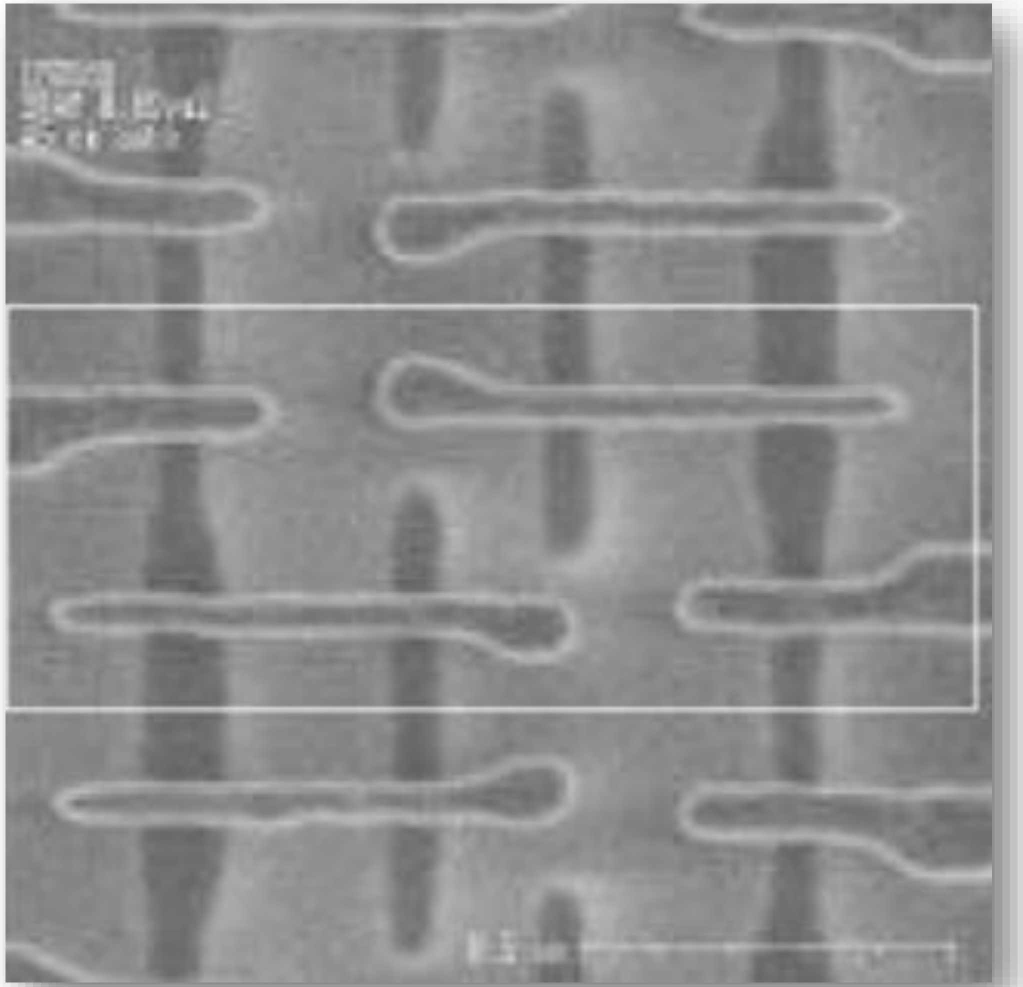




65nm SRAM

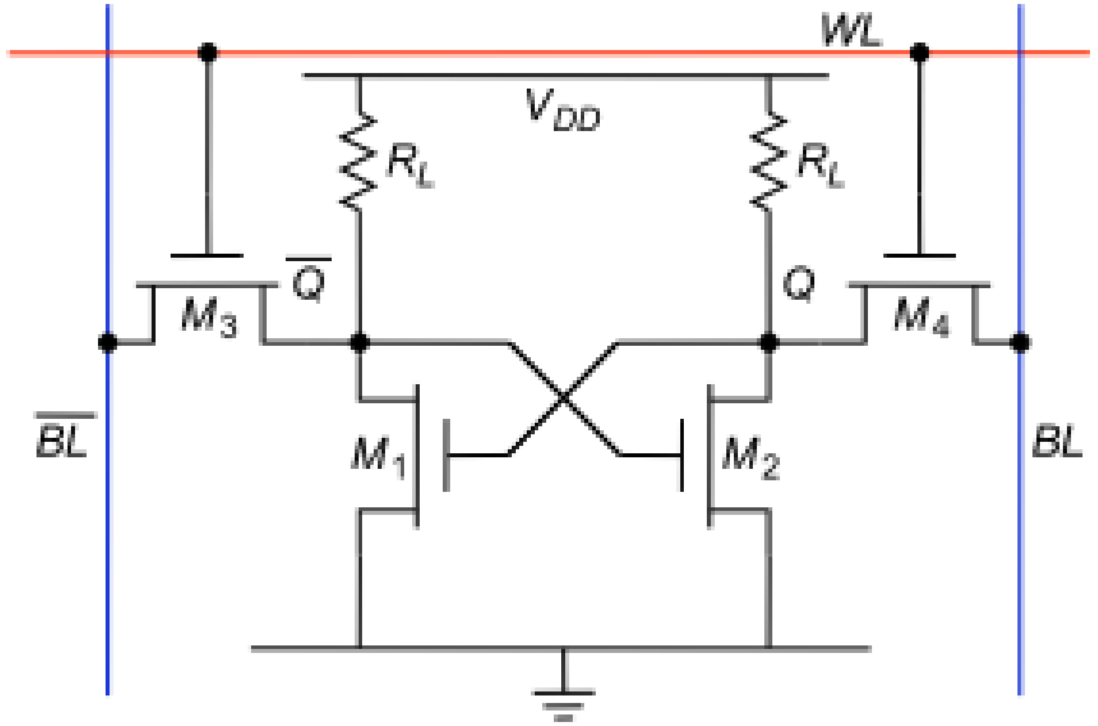
- **Industrial example from ST/Phillips**





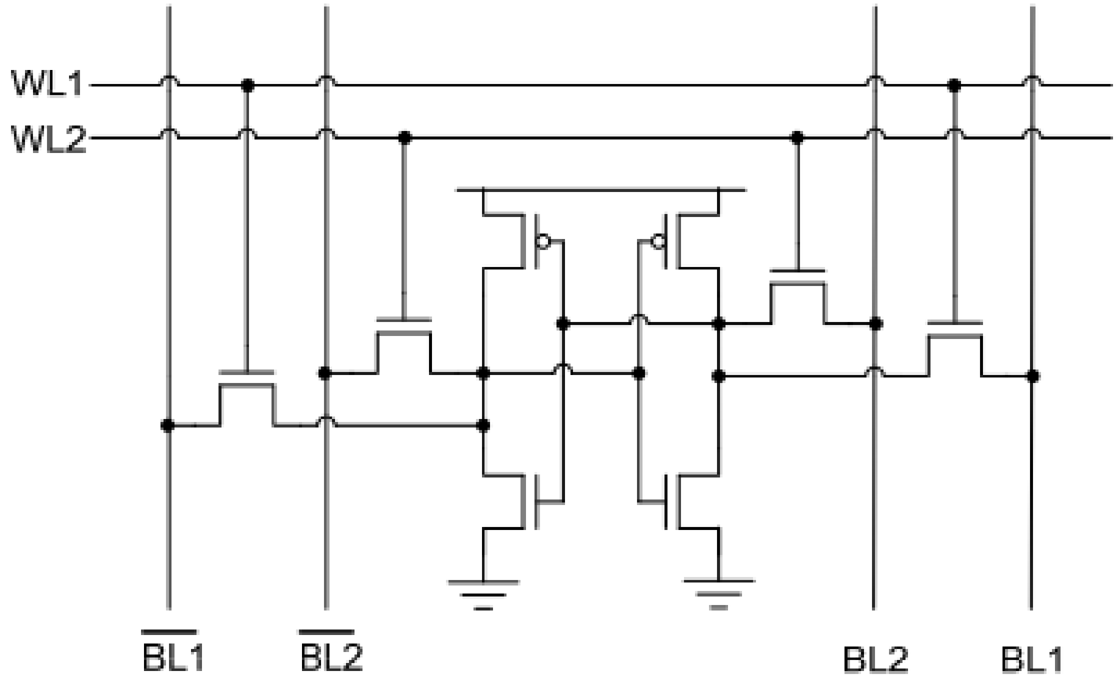
4T Memory Cell

- Achieve density by removing the PMOS pull-up.
- However, this results in static power dissipation.

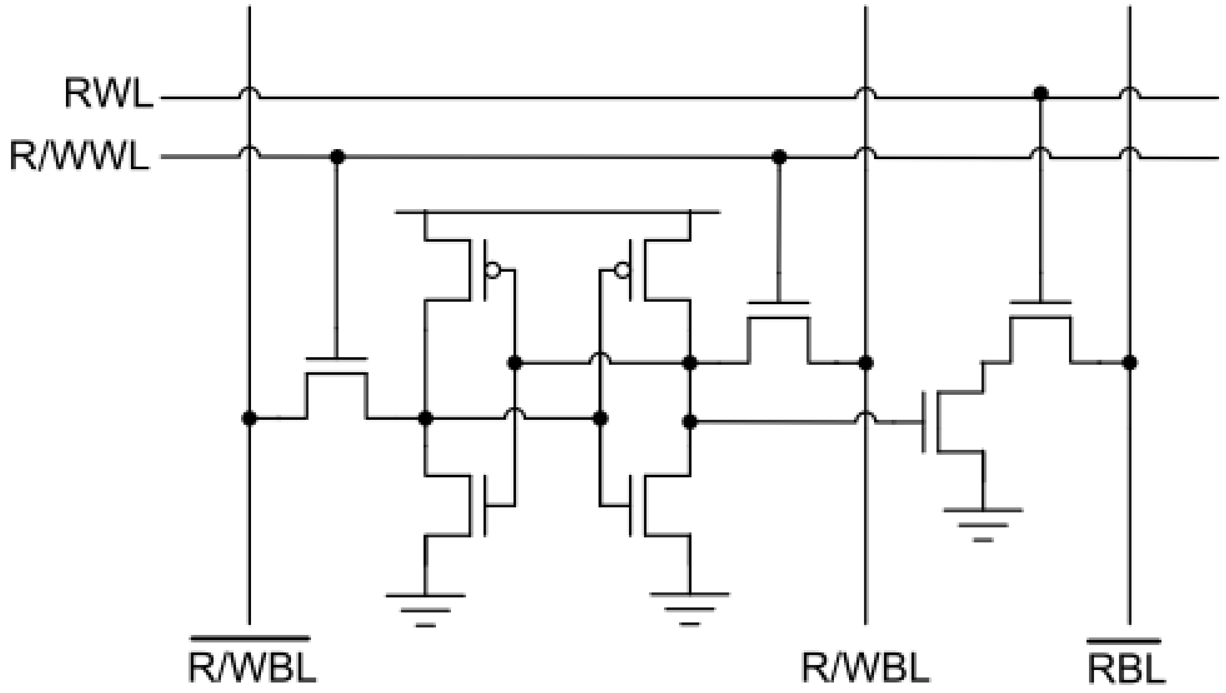


Multi-Port SRAM

Dual Port SRAM

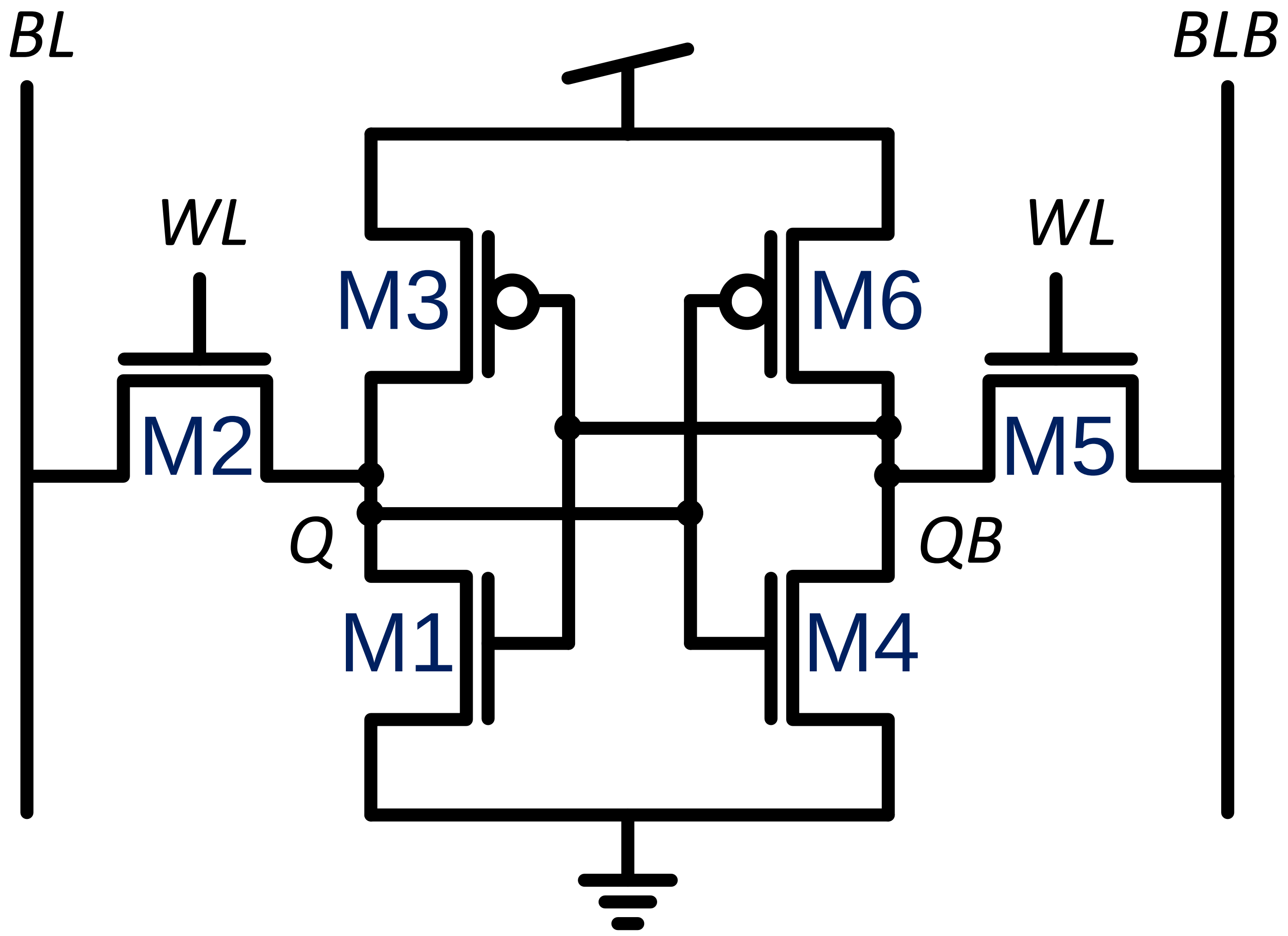


Two Port SRAM

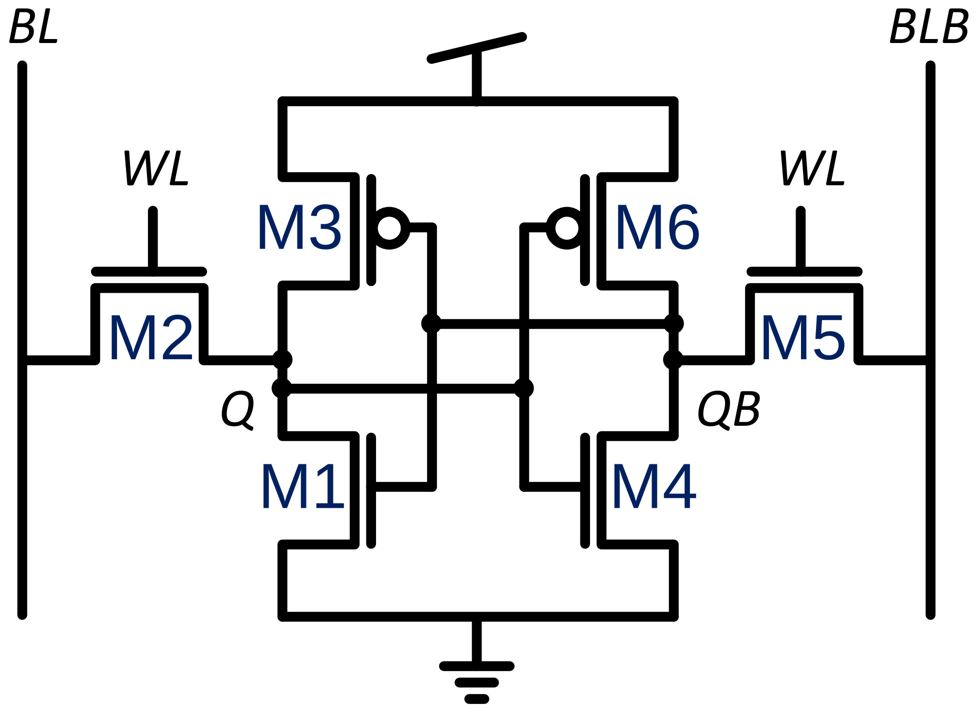


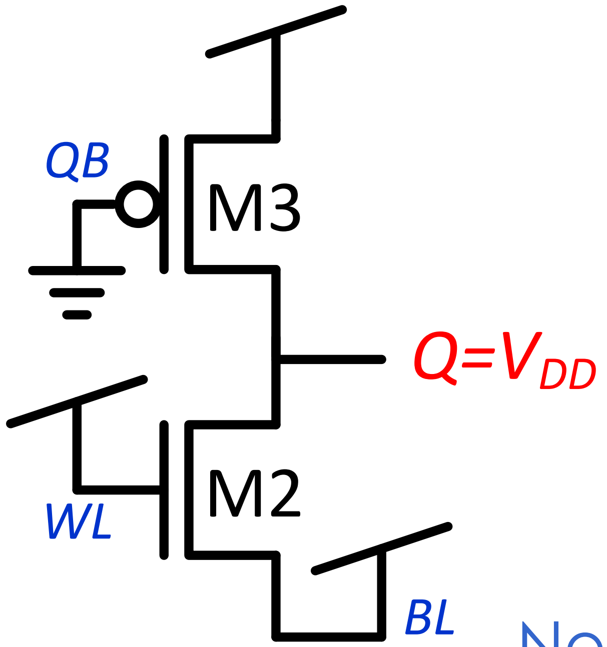
6T SRAM Operation

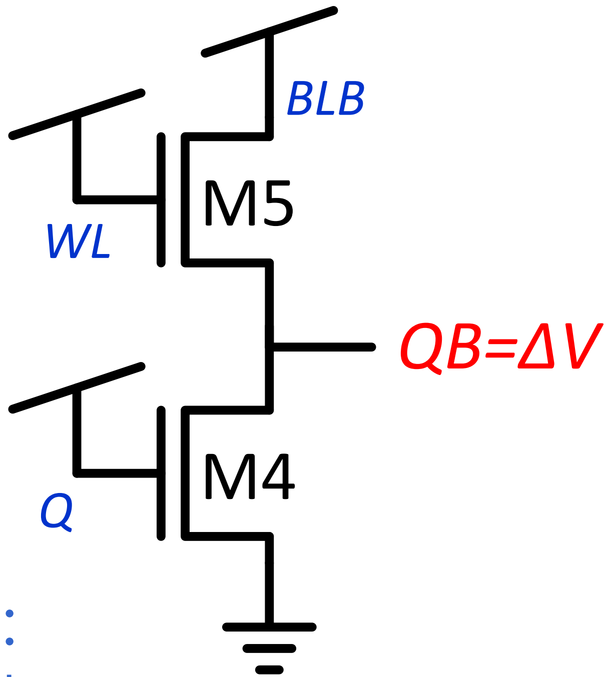
SRAM Operation: HOLD



SRAM Operation: READ



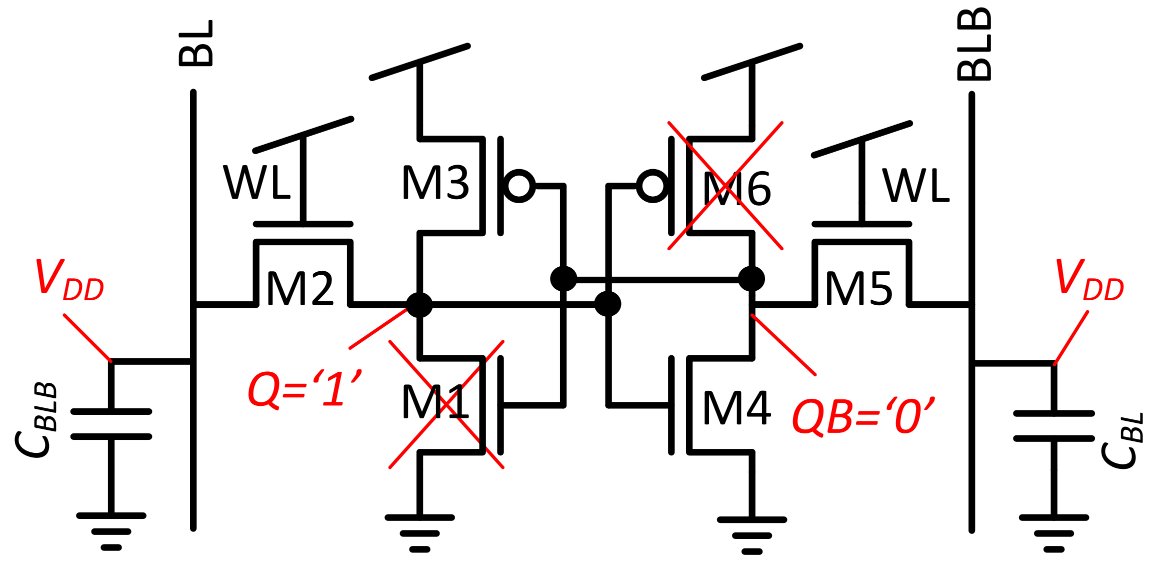


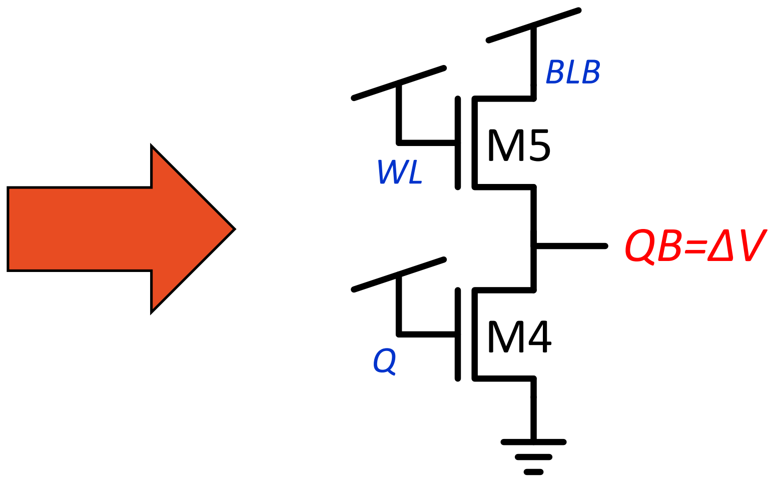


Left Side:
Nothing Changes…

Right Side:
"nMOS" inverter –
QB voltage rises

SRAM Operation - Read





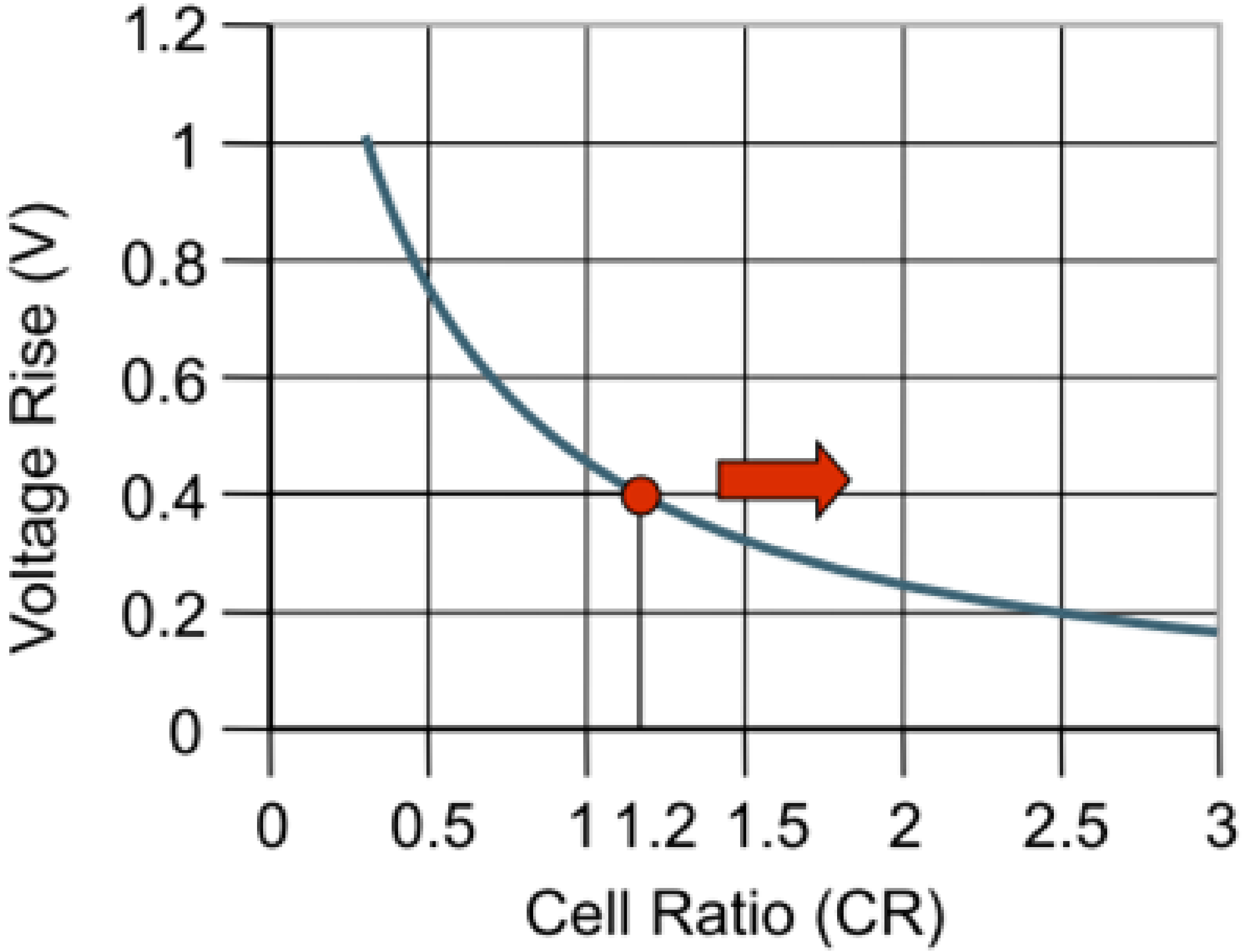
Cell Ratio:

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

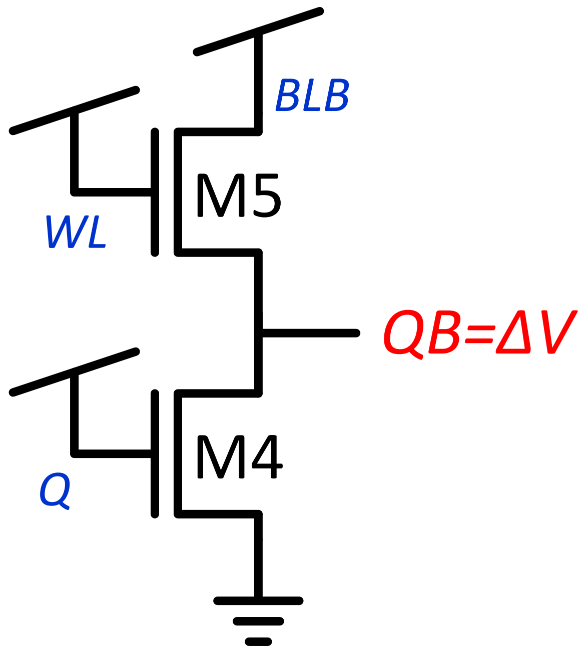
$$k_{M5}\left[\left(V_{DD}-\Delta V-V_{T,n}\right)V_{DSat,n}-\frac{V_{DSat,n}^2}{2}\right]=k_{M4}\left[\left(V_{DD}-V_{T,n}\right)\Delta V-\frac{\Delta V^2}{2}\right]$$

$$\Delta V=\frac{V_{DSat,n}+CR\left(V_{DD}-V_{T,n}\right)-\sqrt{V_{DSat,n}^2\left(1+CR\right)+CR^2\left(V_{DD}-V_{T,n}\right)^2}}{CR}$$

Cell Ratio (Read Constraint)

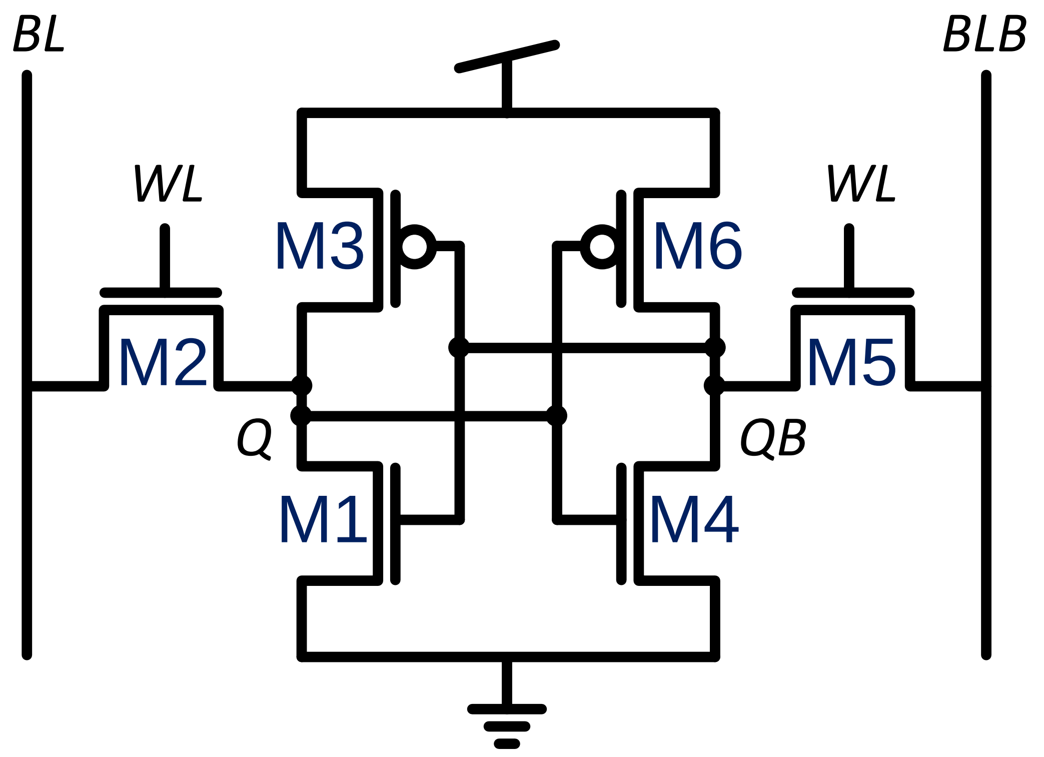


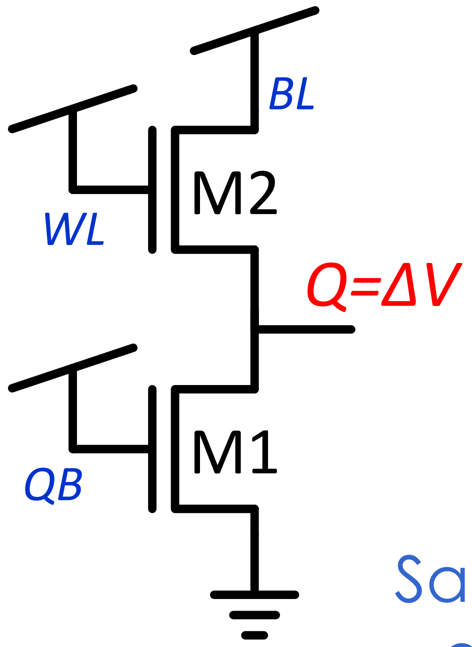
$$CR \equiv \frac{W_4/L_4}{W_5/L_5}$$

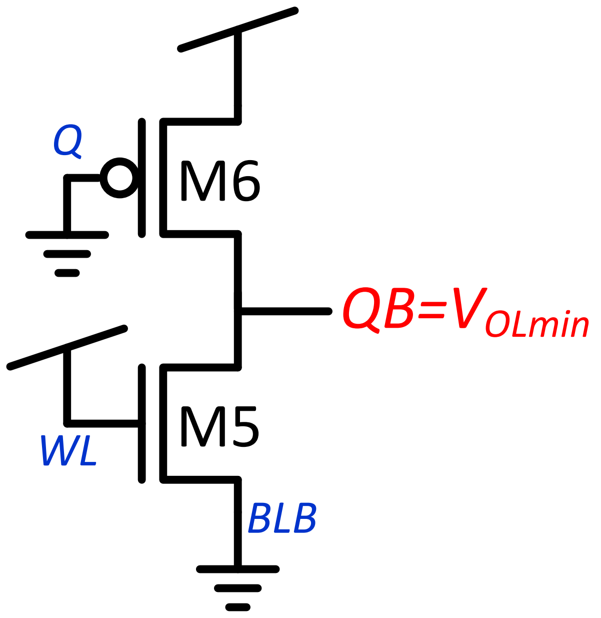


So we need the pull down transistor to be much stronger than the access transistor…

SRAM Operation: WRITE







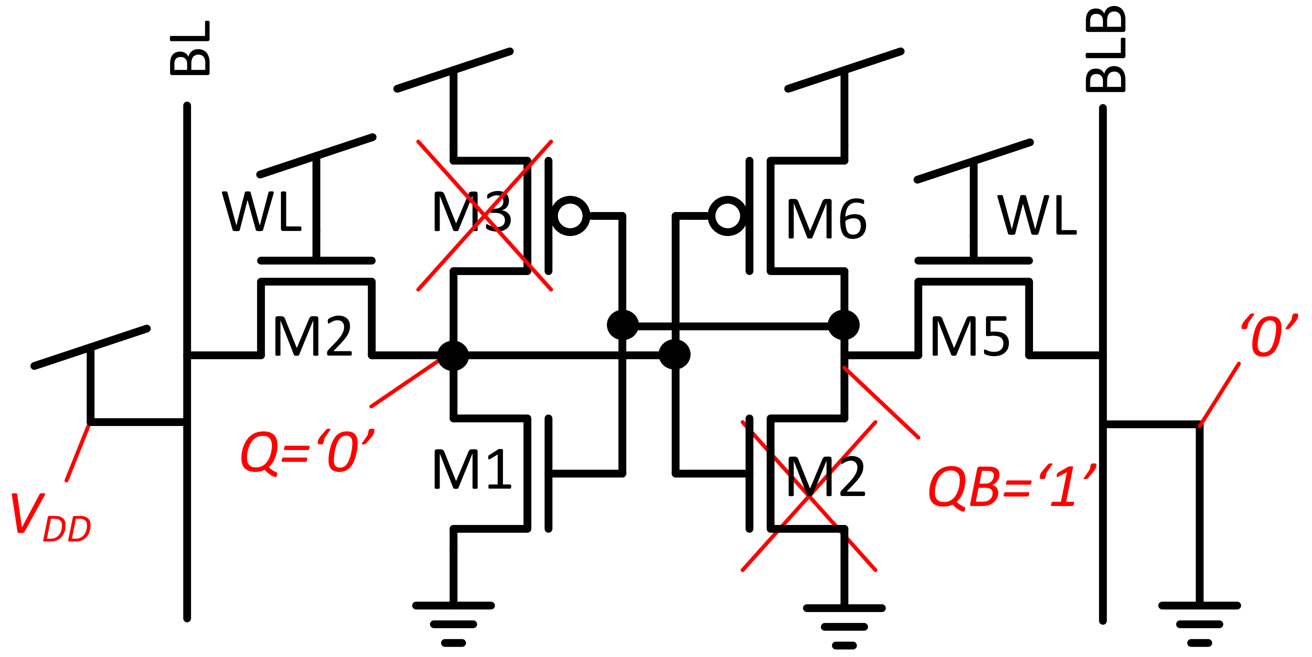
Left Side:
Same as during read – designed so $\Delta V < V_M$

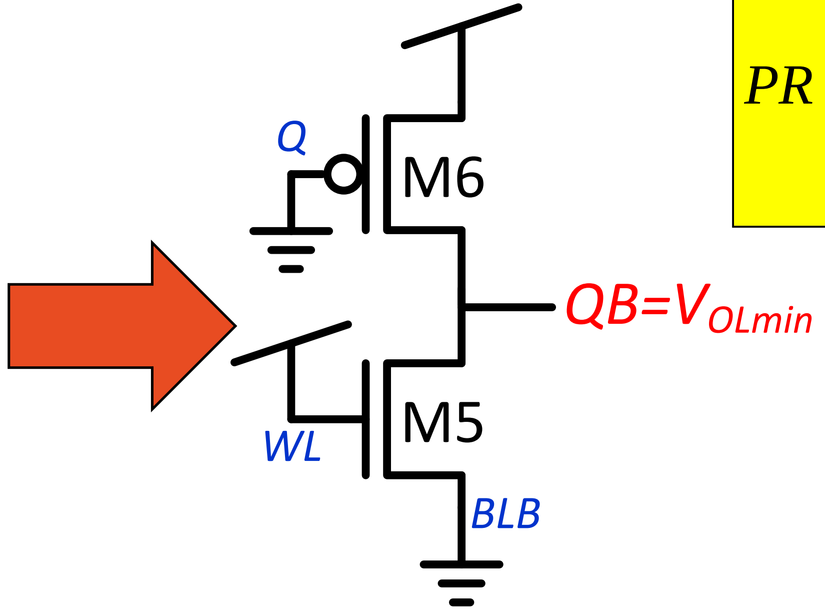
Right Side:
Pseudo nMOS inverter!

SRAM Operation - Write

Pull-Up Ratio

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

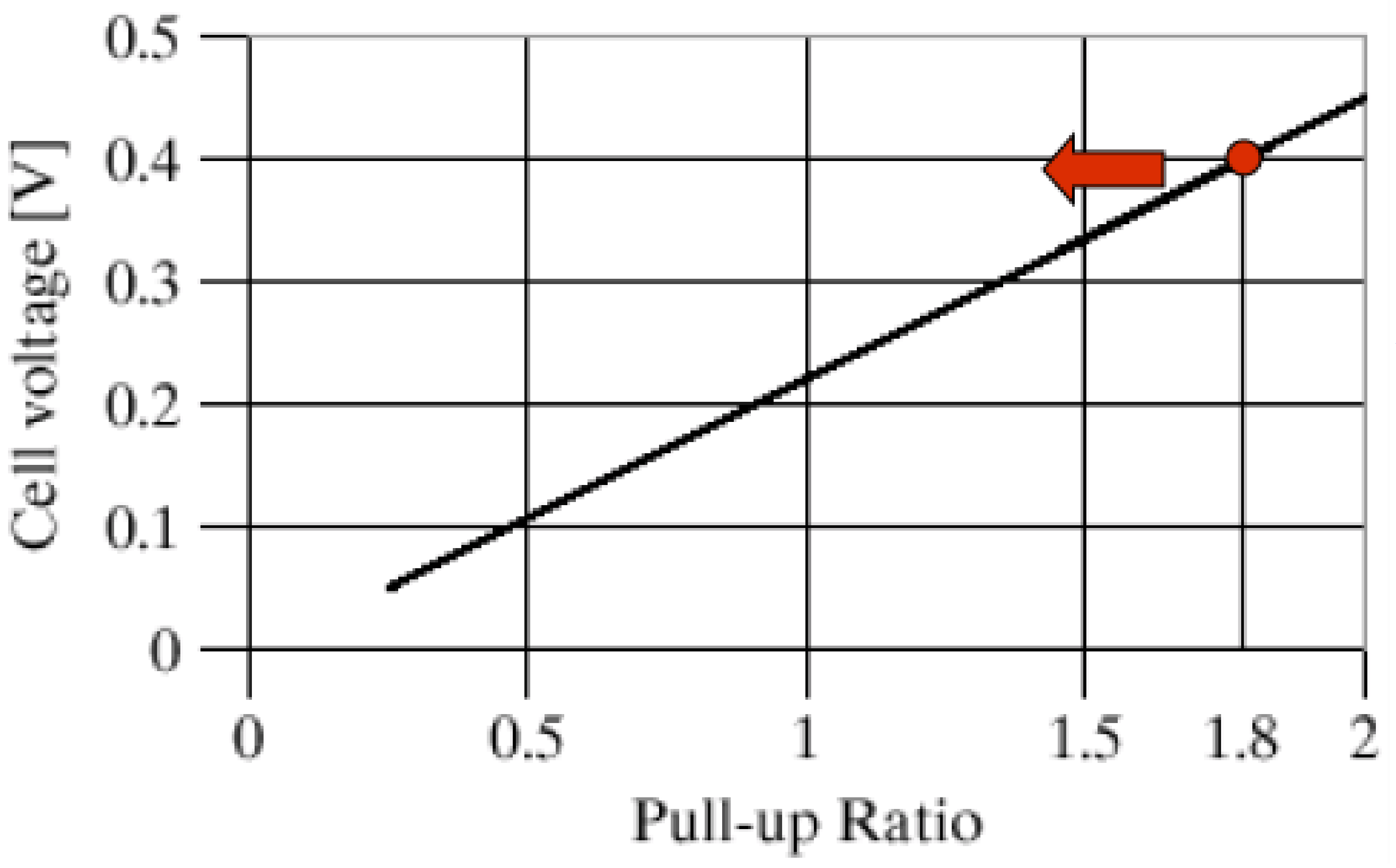


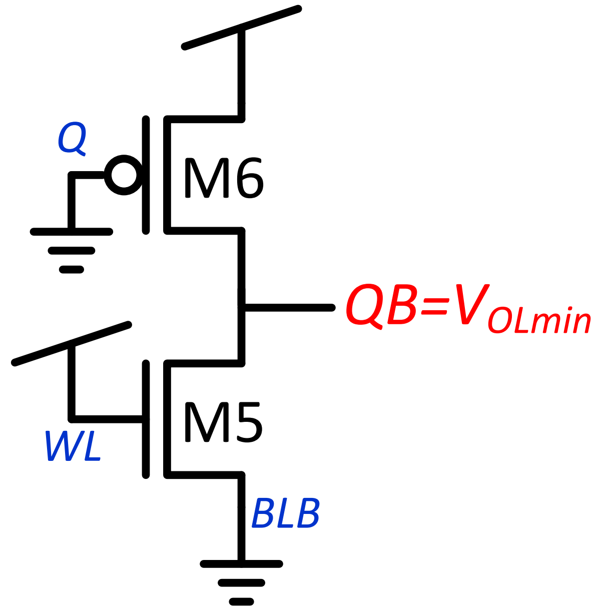


$$k_{M6}\left[\left(V_{DD}-\left|V_{T,p}\right|\right)V_{DSat,p}-\frac{V_{DSat,p}^2}{2}\right]=k_{M5}\left[\left(V_{DD}-V_{T,n}\right)V_{QB}-\frac{\Delta V_{QB}^2}{2}\right]$$

$$V_{QB}=V_{DD}-V_{T,n}-\sqrt{\left(V_{DD}-V_{T,n}\right)^2-2\frac{\mu_p}{\mu_n}PR\left[\left(V_{DD}-\left|V_{T,p}\right|\right)V_{DSat,p}-\frac{V_{DSat,p}^2}{2}\right]}$$

Pull Up Ratio – Write Constraint



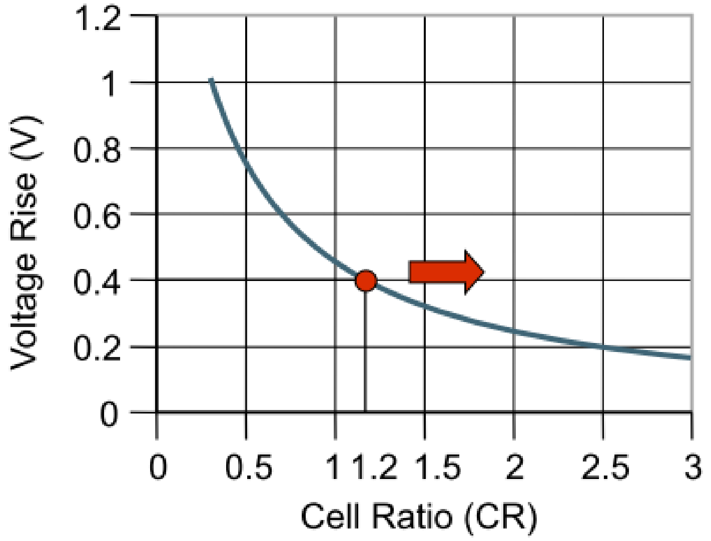


So we need the access transistor to be much stronger than the pull up transistor…

$$\boxed{PR \equiv \frac{W_6/L_6}{W_5/L_5}}$$

Summary – SRAM Sizing Constraints

Read Constraint



$$CR \equiv \frac{W_1/L_1}{W_2/L_2} = \frac{W_4/L_4}{W_5/L_5} = \frac{PDN}{access}$$

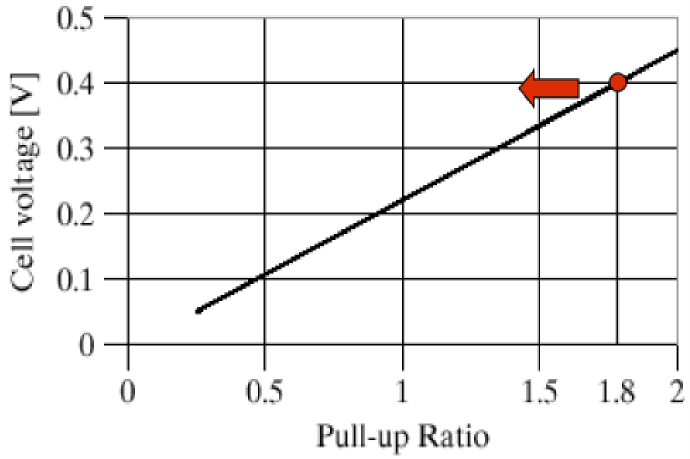
$$\Rightarrow K_{PDN} > K_{access}$$

$$\Downarrow$$

$$\boxed{K_{PDN} > K_{access} > K_{PUN}}$$

$$\Uparrow$$

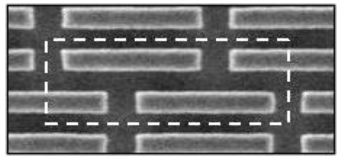
Write Constraint

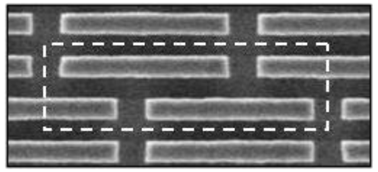


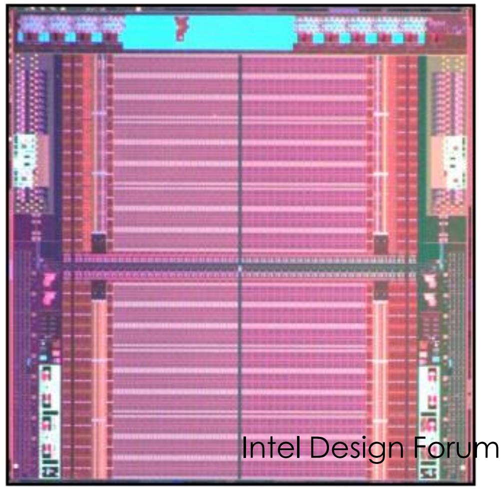
$$\Rightarrow K_{access} > K_{PUN}$$

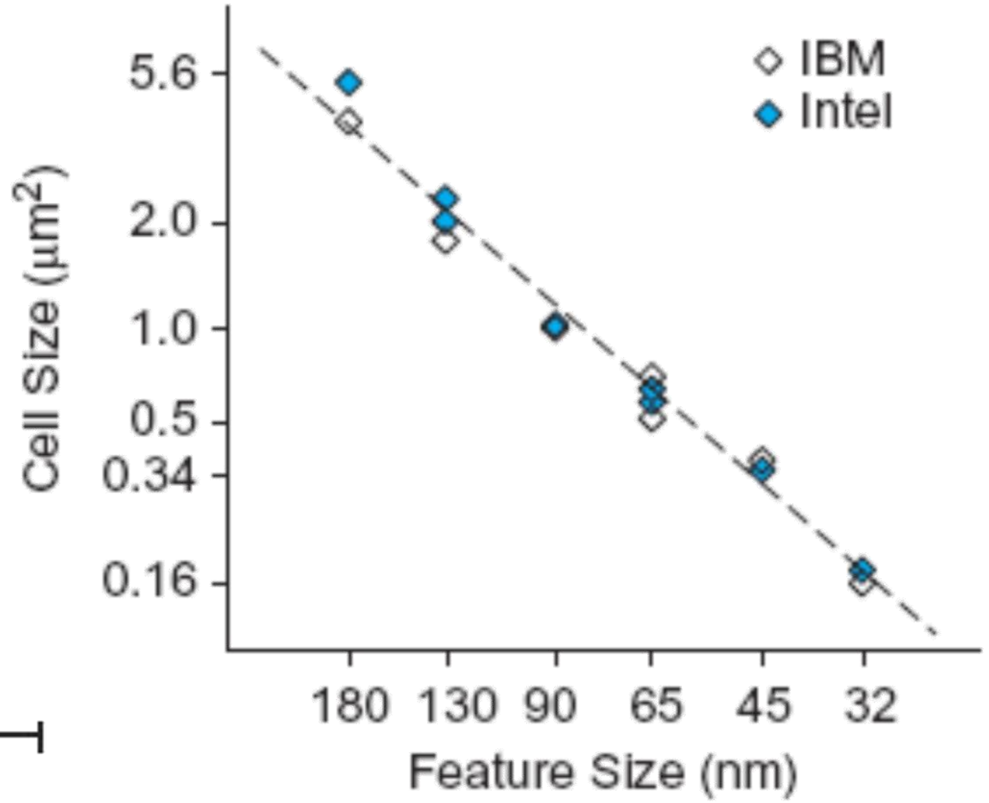
$$PR \equiv \frac{W_3/L_3}{W_2/L_2} = \frac{W_6/L_6}{W_5/L_5} = \frac{PUN}{access}$$

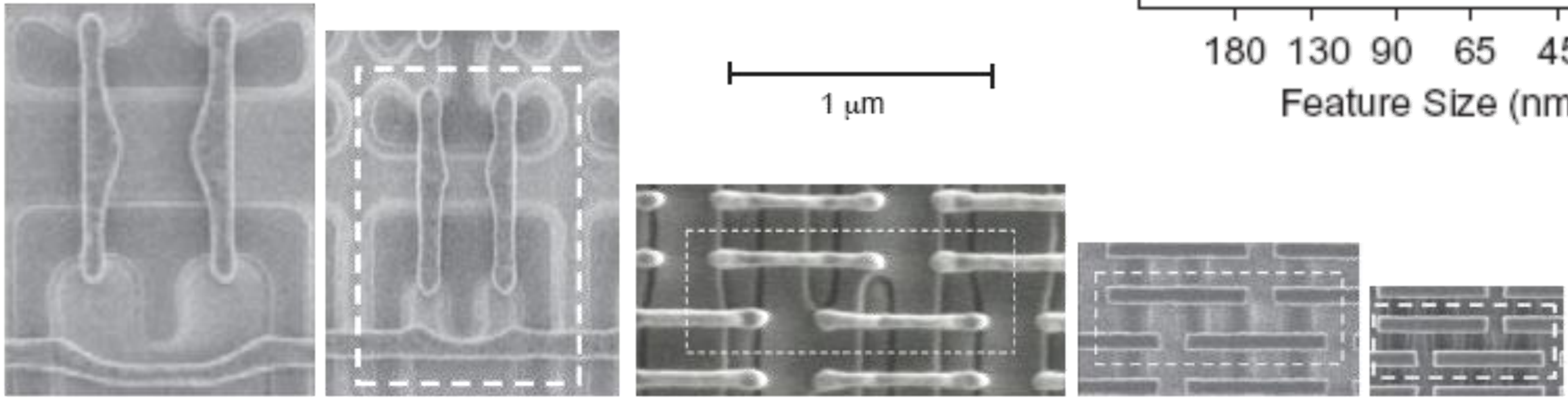
Commercial SRAMs


0.092 um² SRAM cell
for high density applications


0.108 um² SRAM cell
for low voltage applications


Intel Design Forum 2009



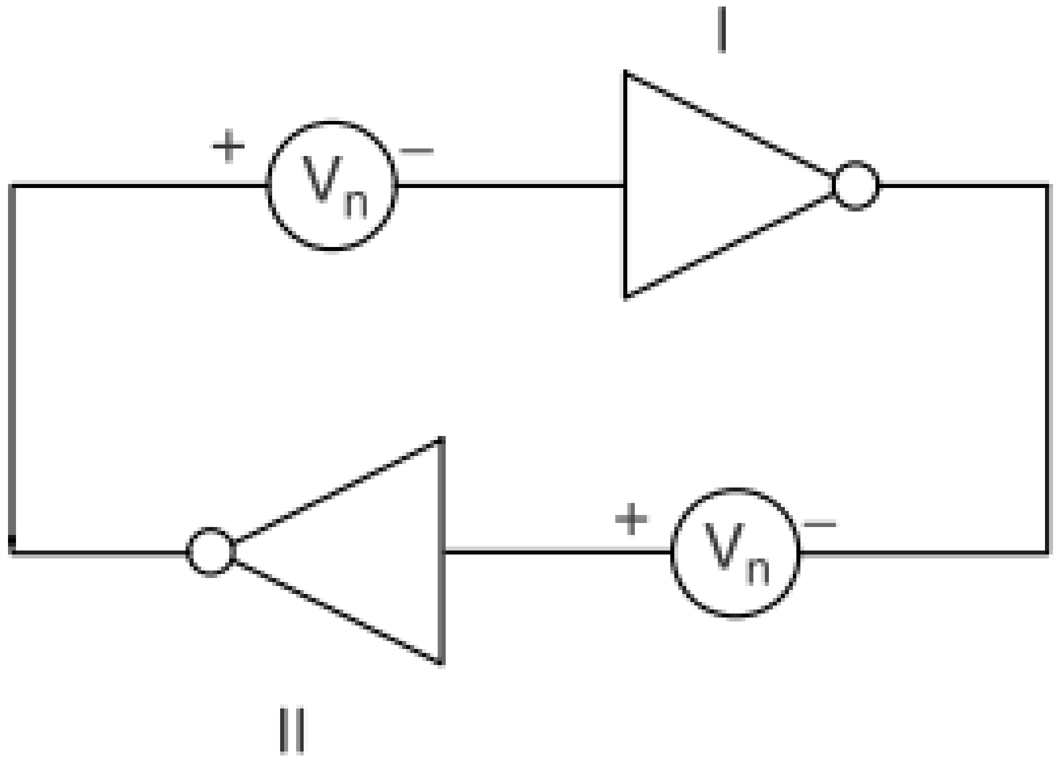


130 nm [Tyagi00] 90 nm [Thompson02] 65 nm [Bai04] 45 nm [Mistry07] 32 nm [Natarajan08]

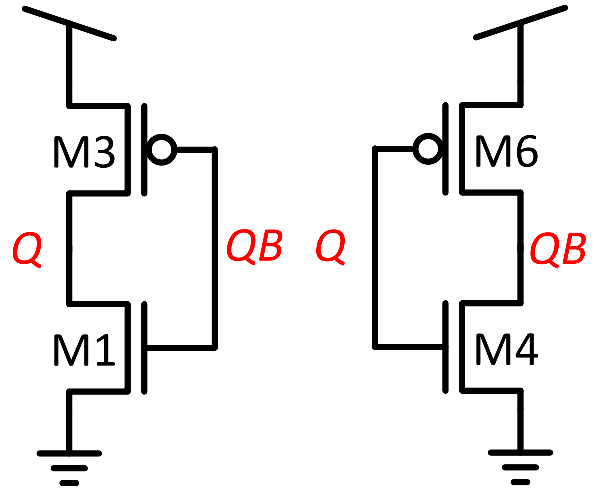
SRAM Stability

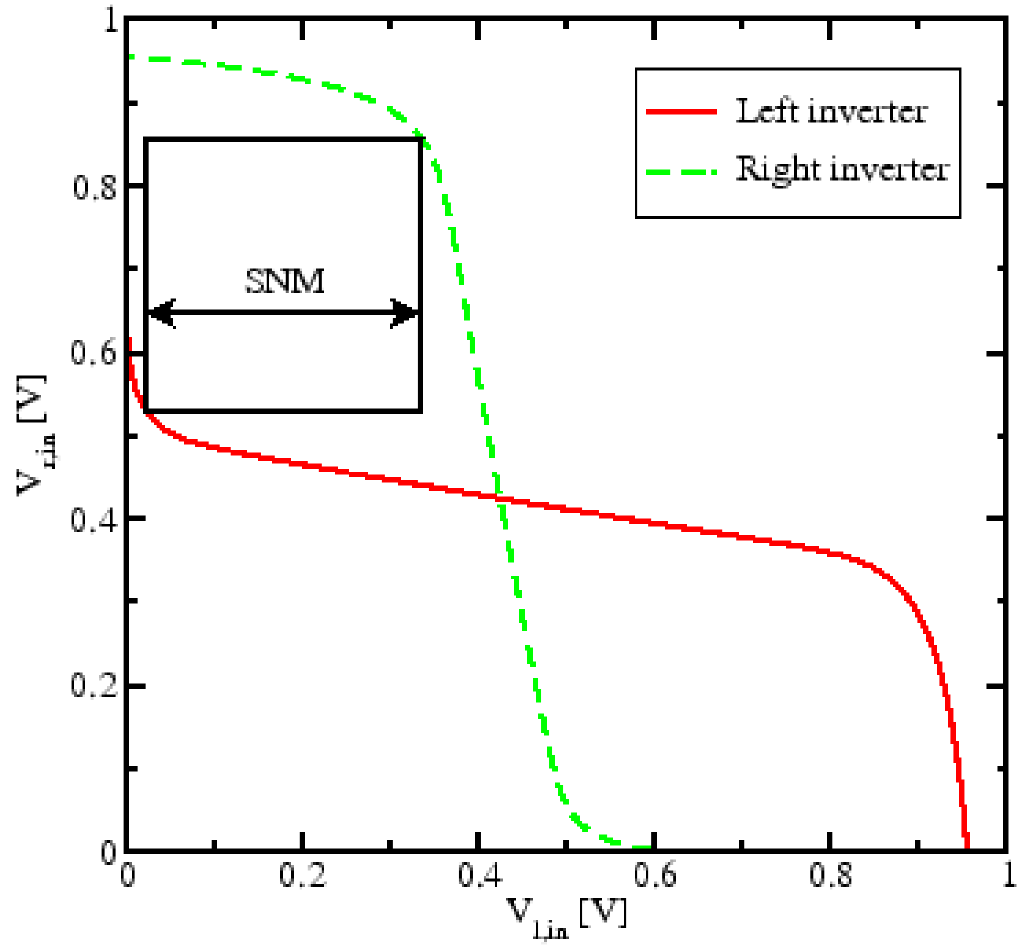
“Static Noise Margin”

Static Noise Margin - Hold



Static Noise Margin - Hold

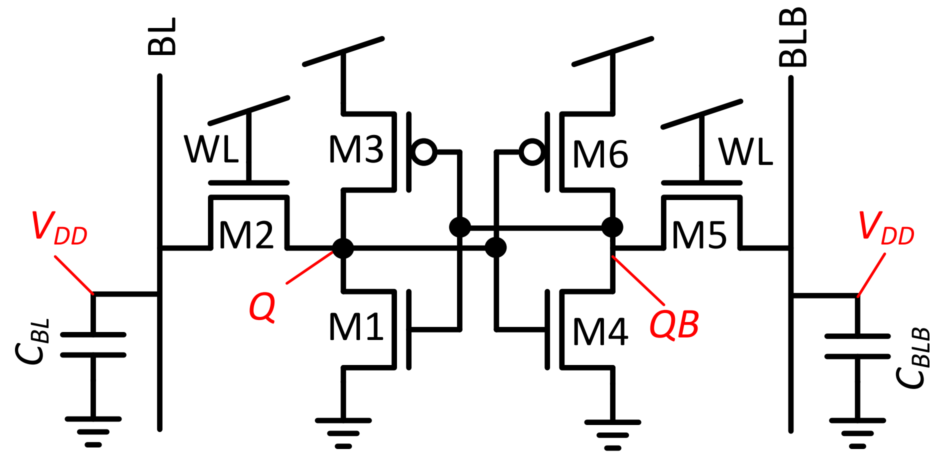


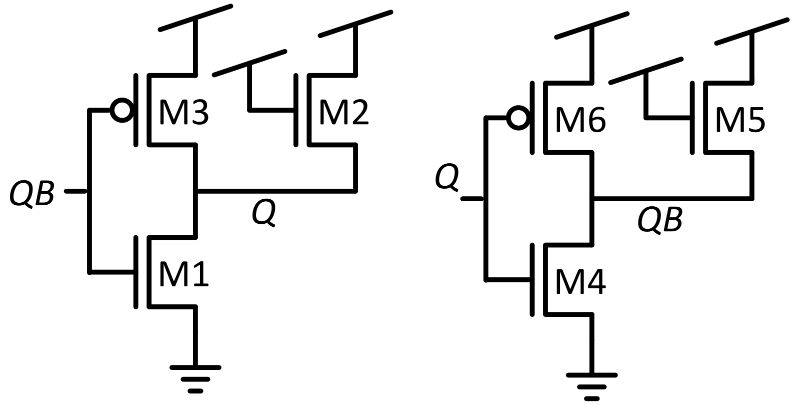


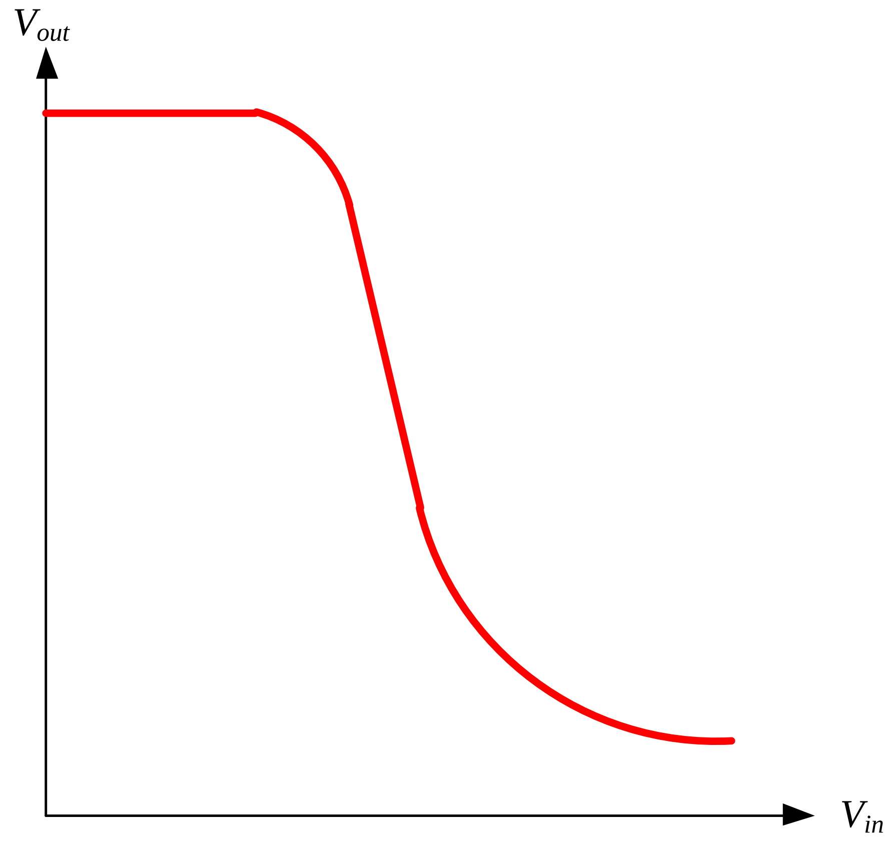
1. Plot both VTCs on the same graph
2. Find the maximum square that fits in the VTC.
3. The SNM is defined as the side of the maximum square.

Static Noise Margin - Read

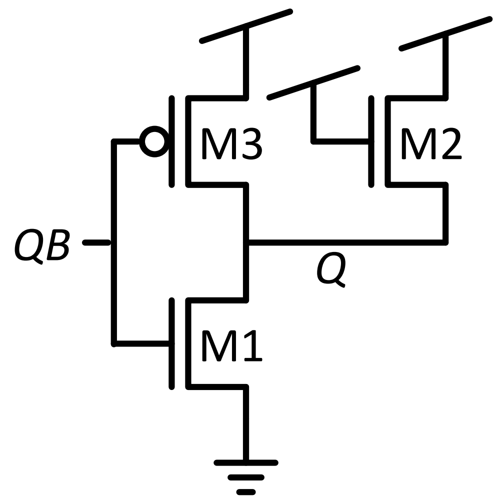
- **What happens during Read?**
  - We can't ignore the access transistors anymore…

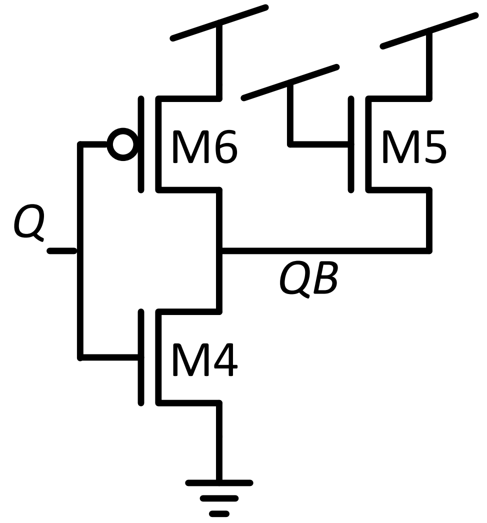


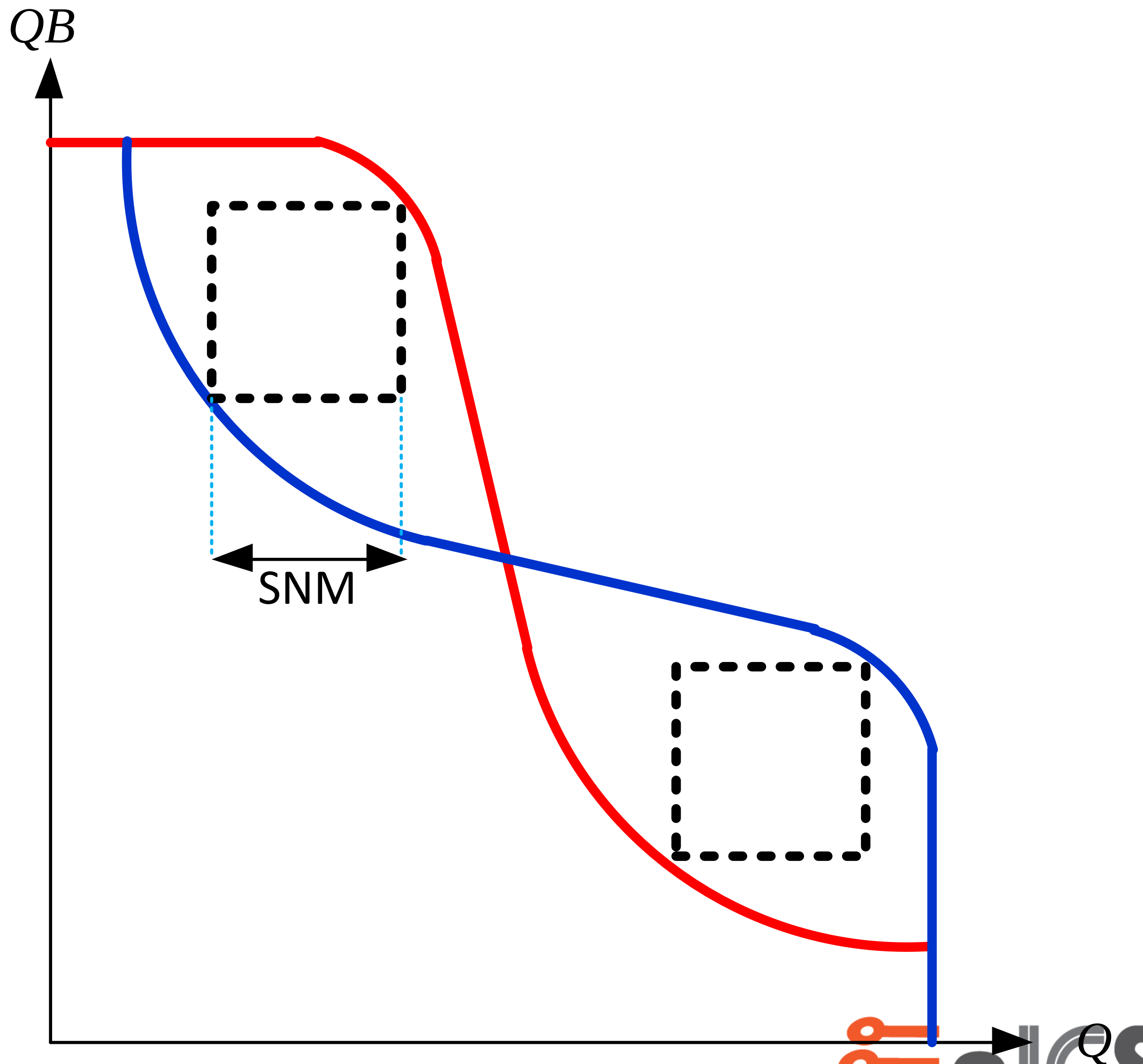




Static Noise Margin - Read

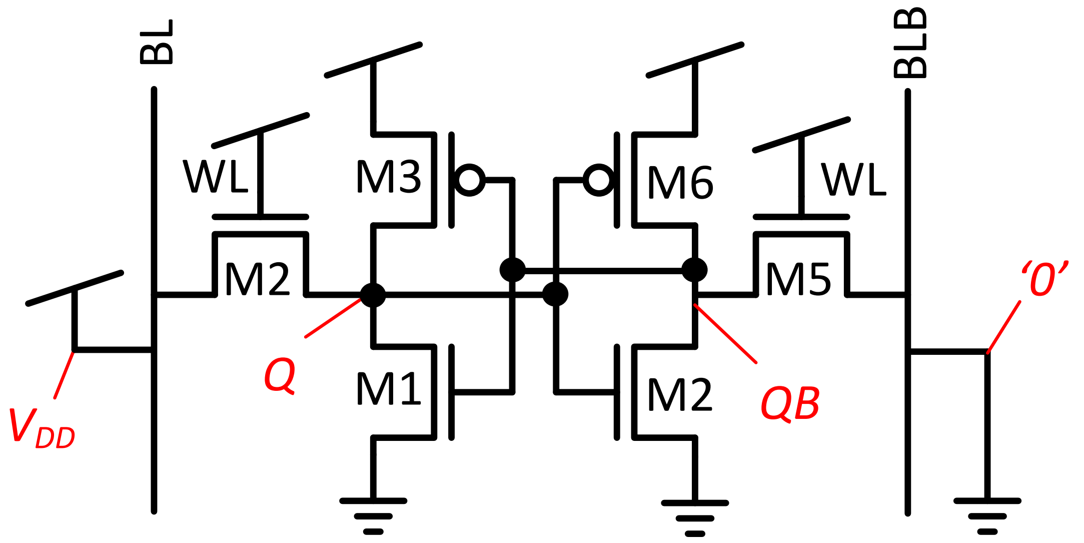


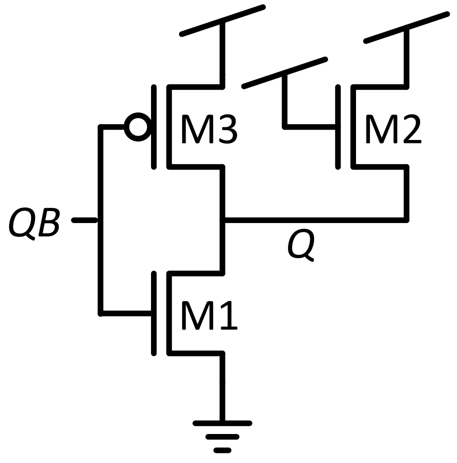


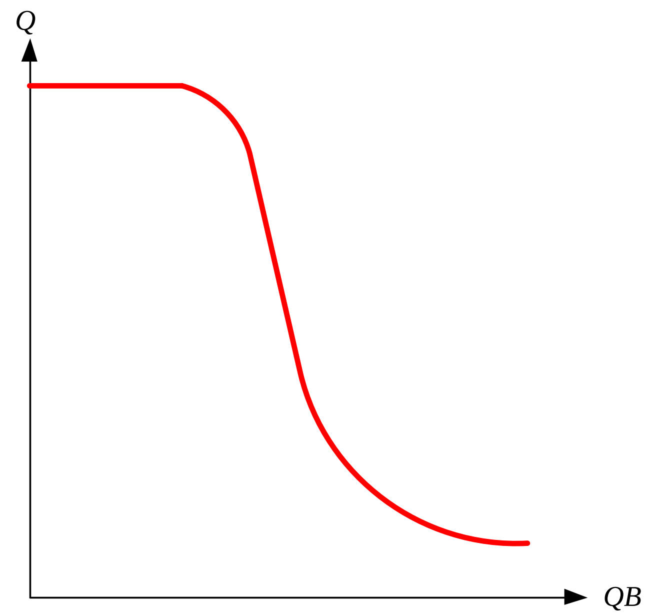


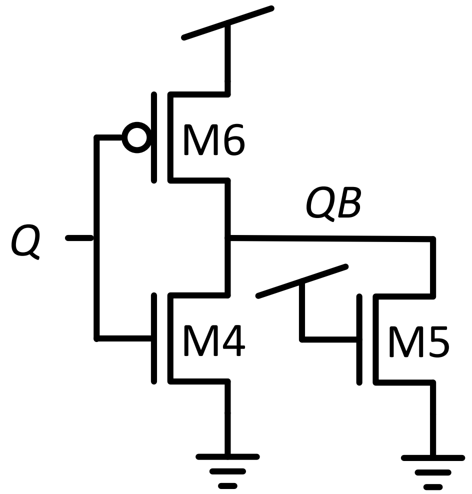
Static Noise Margin - Write

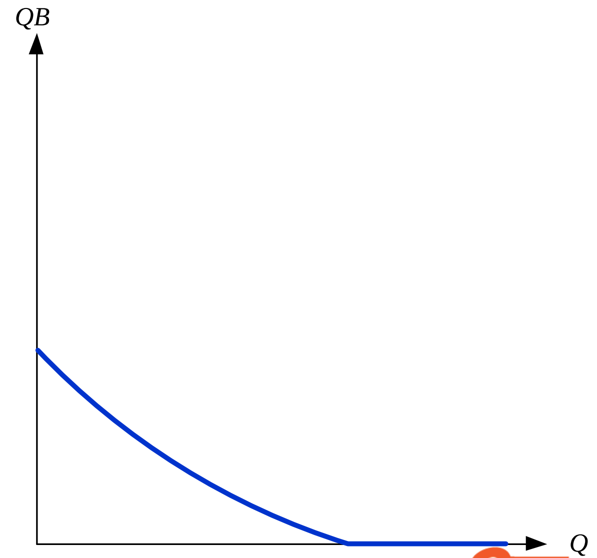
- **What happens during Write?**
  - The two sides are now different.



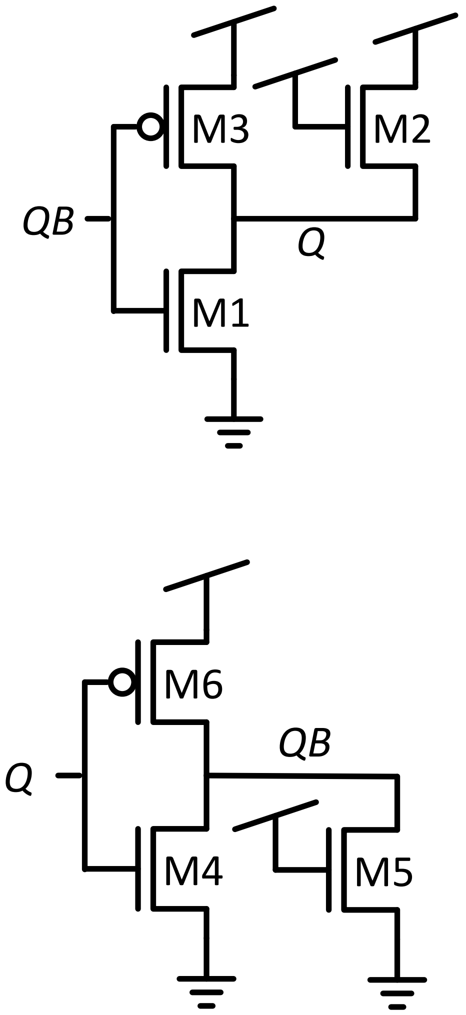


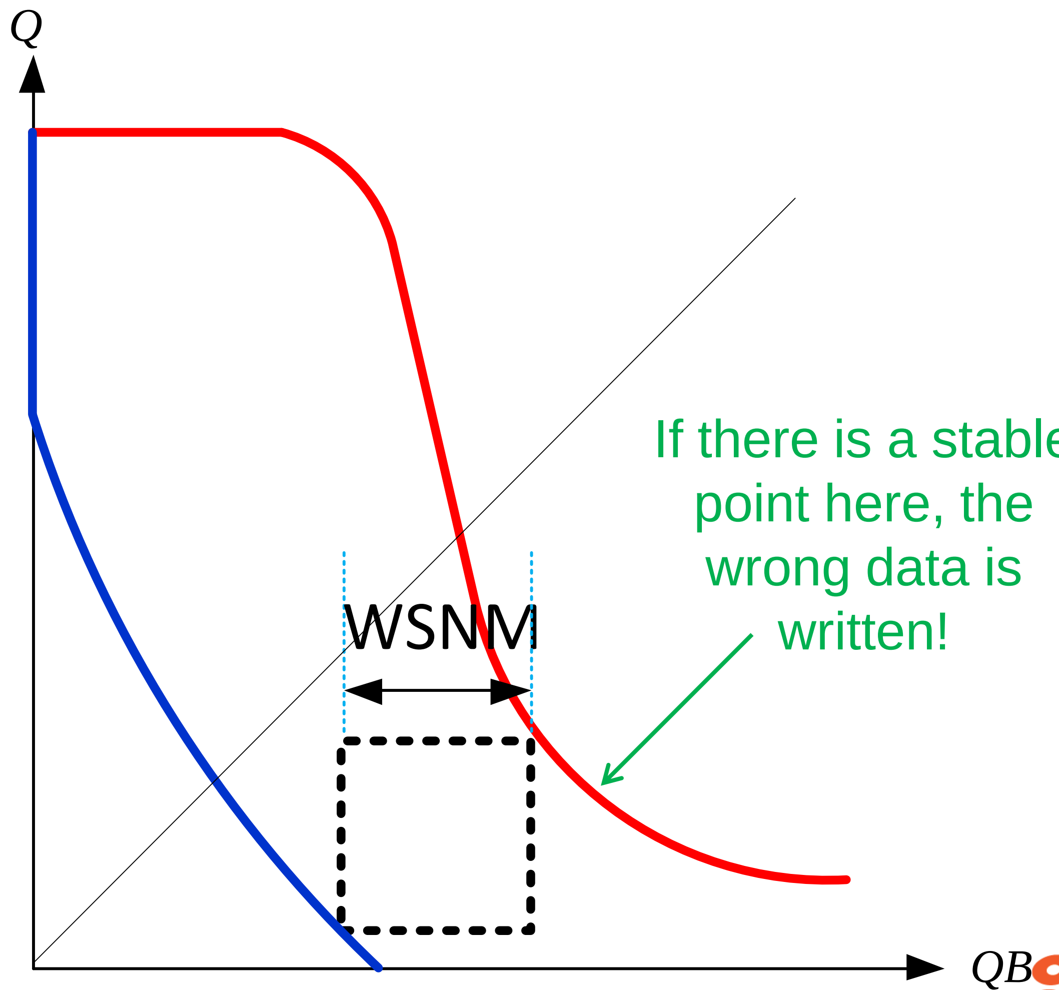






Static Noise Margin - Write

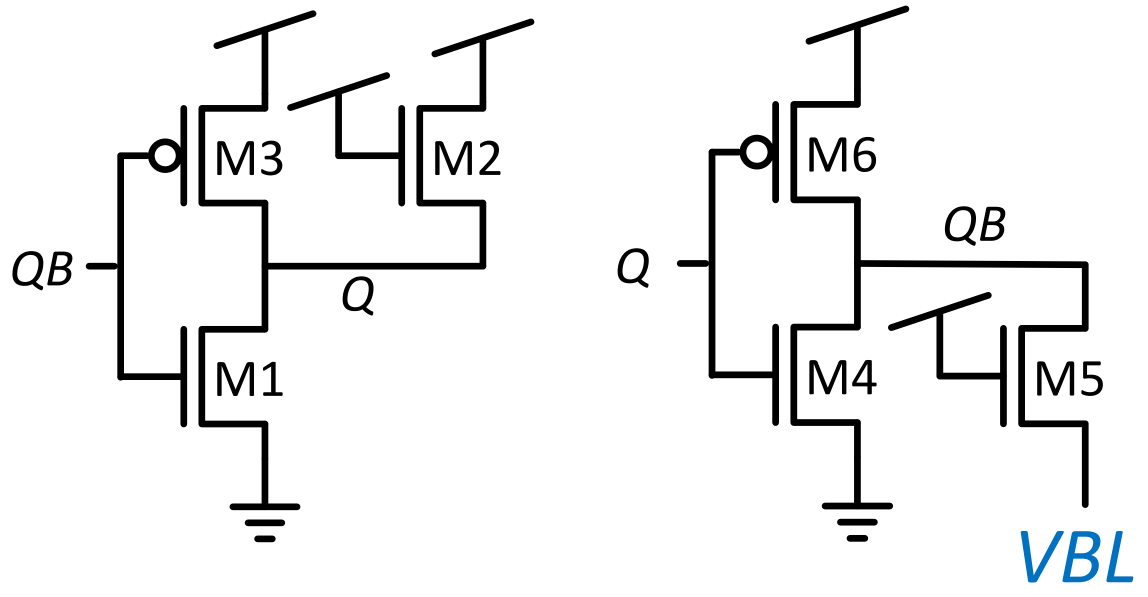


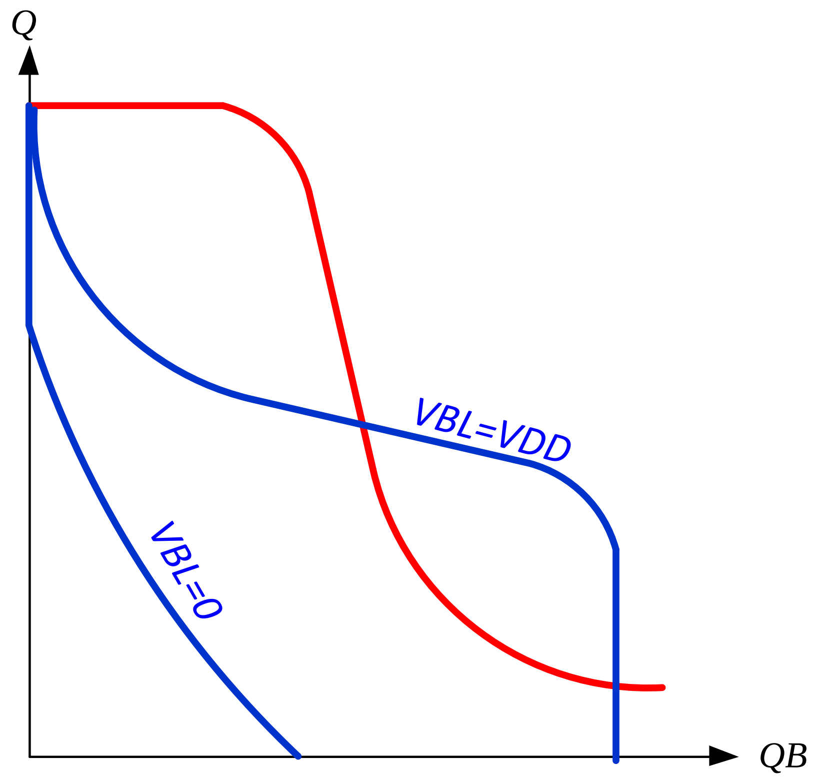


If there is a stable point here, the wrong data is written!

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_{BL}) to the 0 bitline and see how high it can be and still flip the cell.**





Dynamic Stability

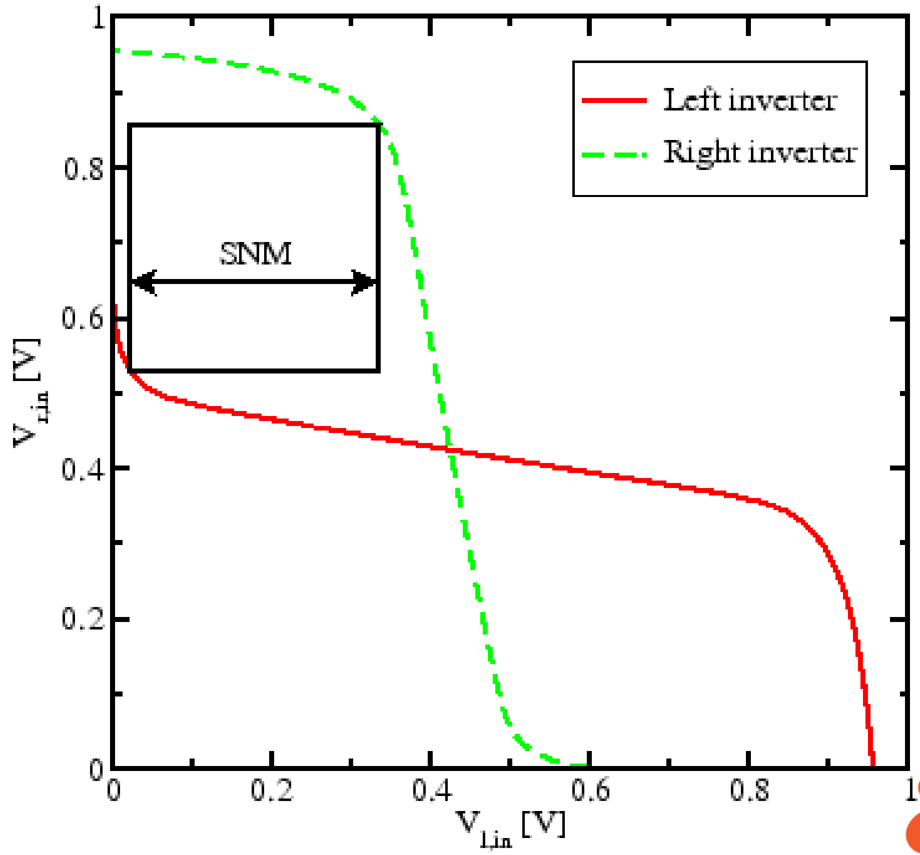
SNM Calculation

Simulating SNM

- **Problem:**
  - How can we calculate *SNM* with *SPICE*?
- **Some options:**
  - Insert *DC sources* at *Q* and *QB*
    - But where exactly do we connect them?
  - Draw *Butterfly Curves*
    - But how do we find the largest squares?
- **To run Monte Carlo Simulations we should have an easy way of calculation.**

Simulating SNM

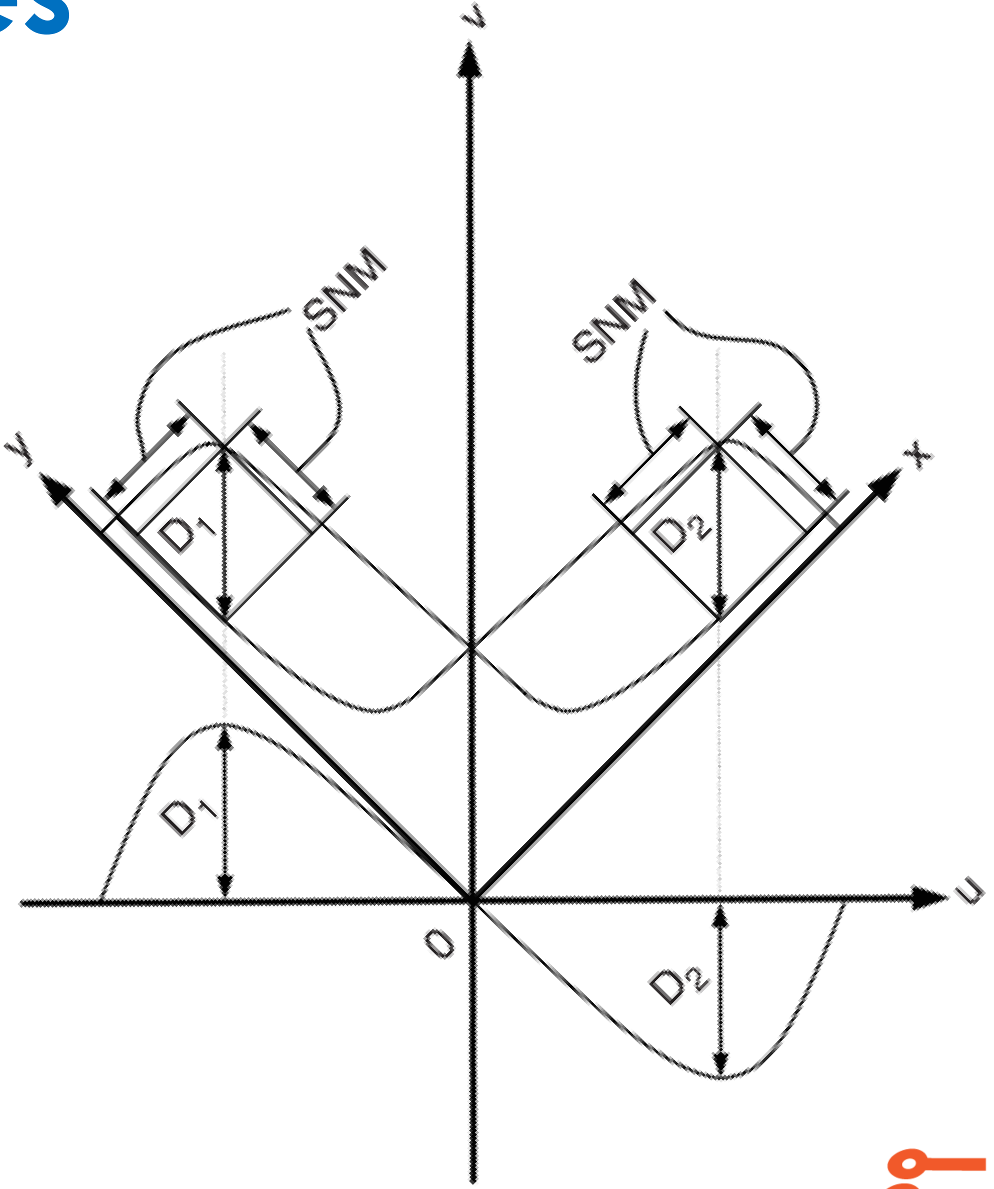
- **First let's define the graphical solution:**
  - The diagonals of all the squares are on lines parallel to $Q=QB$.
  - We need to find the distance between the points where these intersect the butterfly plot.
  - The largest of these distances is the diagonal of the maximum square in each lobe.
  - Multiply this by $cos45$ and we get the SNM.



- **Easy, right?**

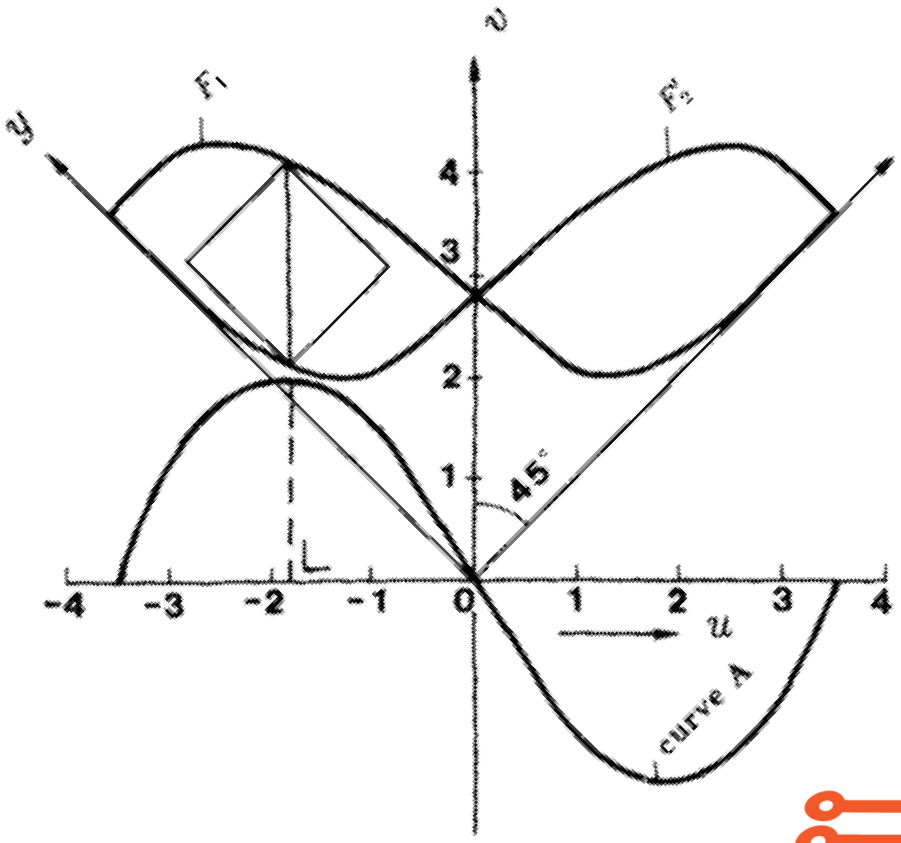
Changing Coordinates

- **What if we were to turn the graph?**



Changing Coordinates

- **If we were to use new axes, we could just subtract the graphs.**
  - This gives us the distances between the intersections with the $Q = QB$ parallels.
  - Now all we have to do is find the maximum of the subtraction.
  - (Don't forget to multiply by $\cos 45°$)



Changing Coordinates

- The required transformation is:

$$x = \frac{1}{\sqrt{2}}u + \frac{1}{\sqrt{2}}v$$

$$y = -\frac{1}{\sqrt{2}}u + \frac{1}{\sqrt{2}}v$$

- Now let's define some function as F_1

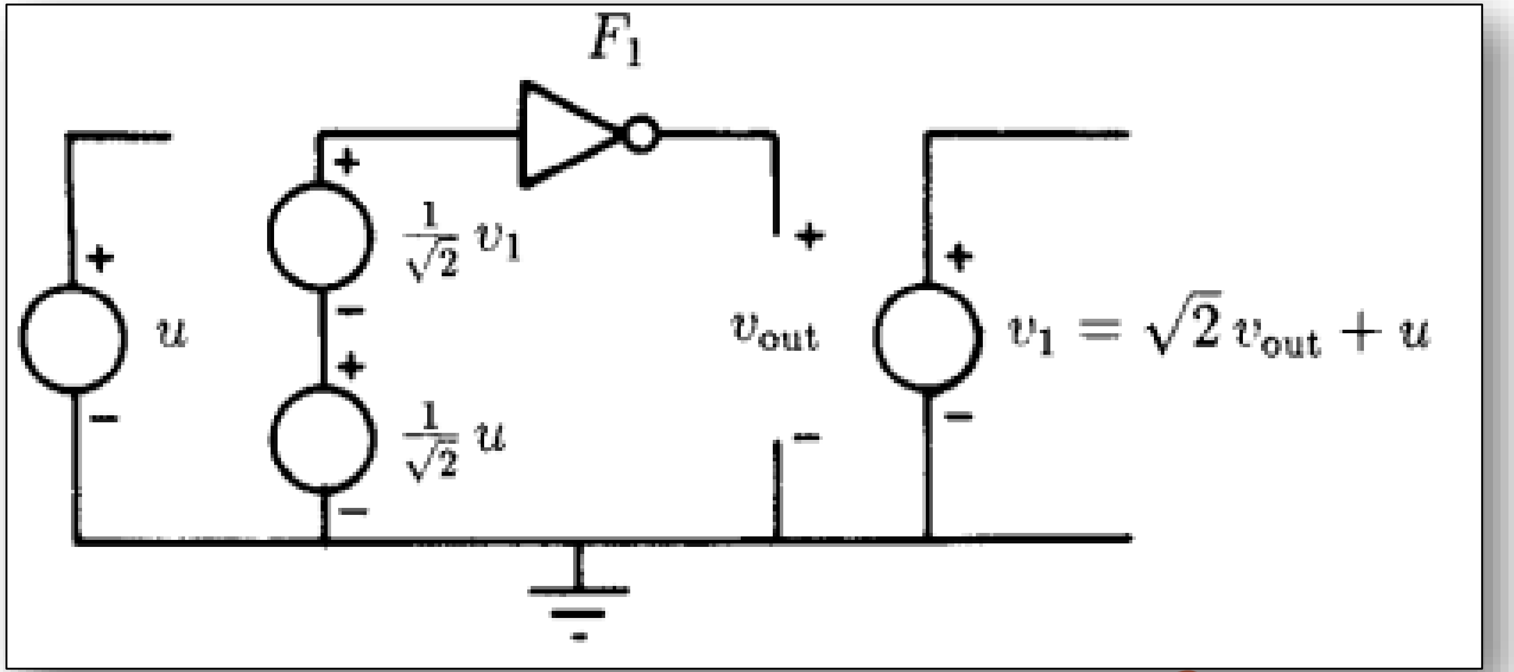
- Substituting $y=F_1(x)$ gives:

$$v = u + \sqrt{2}y =$$

$$= u + \sqrt{2}F_1\left(\frac{1}{\sqrt{2}}u + \frac{1}{\sqrt{2}}v\right)$$

Changing Coordinates

- What we did is turn some function (F_1) *45 degrees* counter clockwise.
- This can easily be implemented with the following circuit:



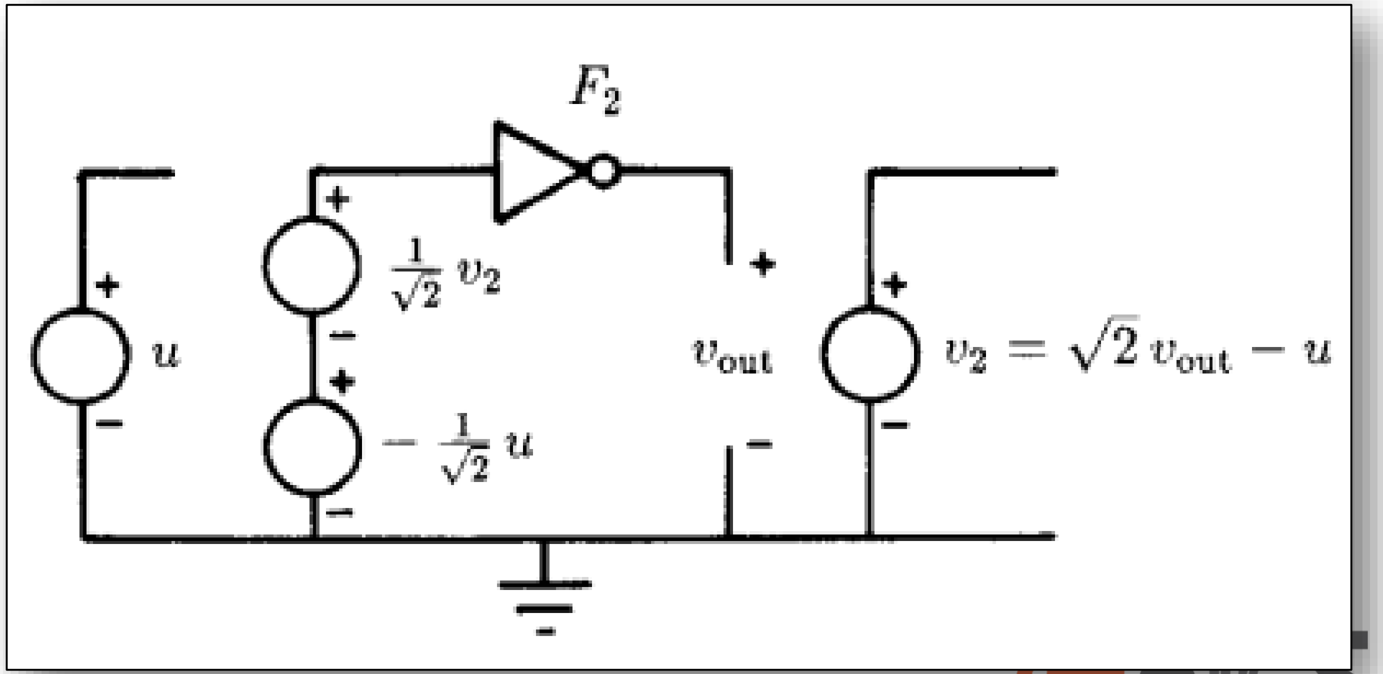
- What is F_1?
  - It could be the *VTC* of $V_{in}=Q$, $V_{out}=QB$…

Changing Coordinates

- **But what about the "mirrored" VTC?**
  - This needs to first be mirrored with respect to the v axis and then transformed to the (u,v) system.
  - If we call the second $VTC\ F_2$ with $\boldsymbol{x=F_2(y)}$ then the operation we need is:

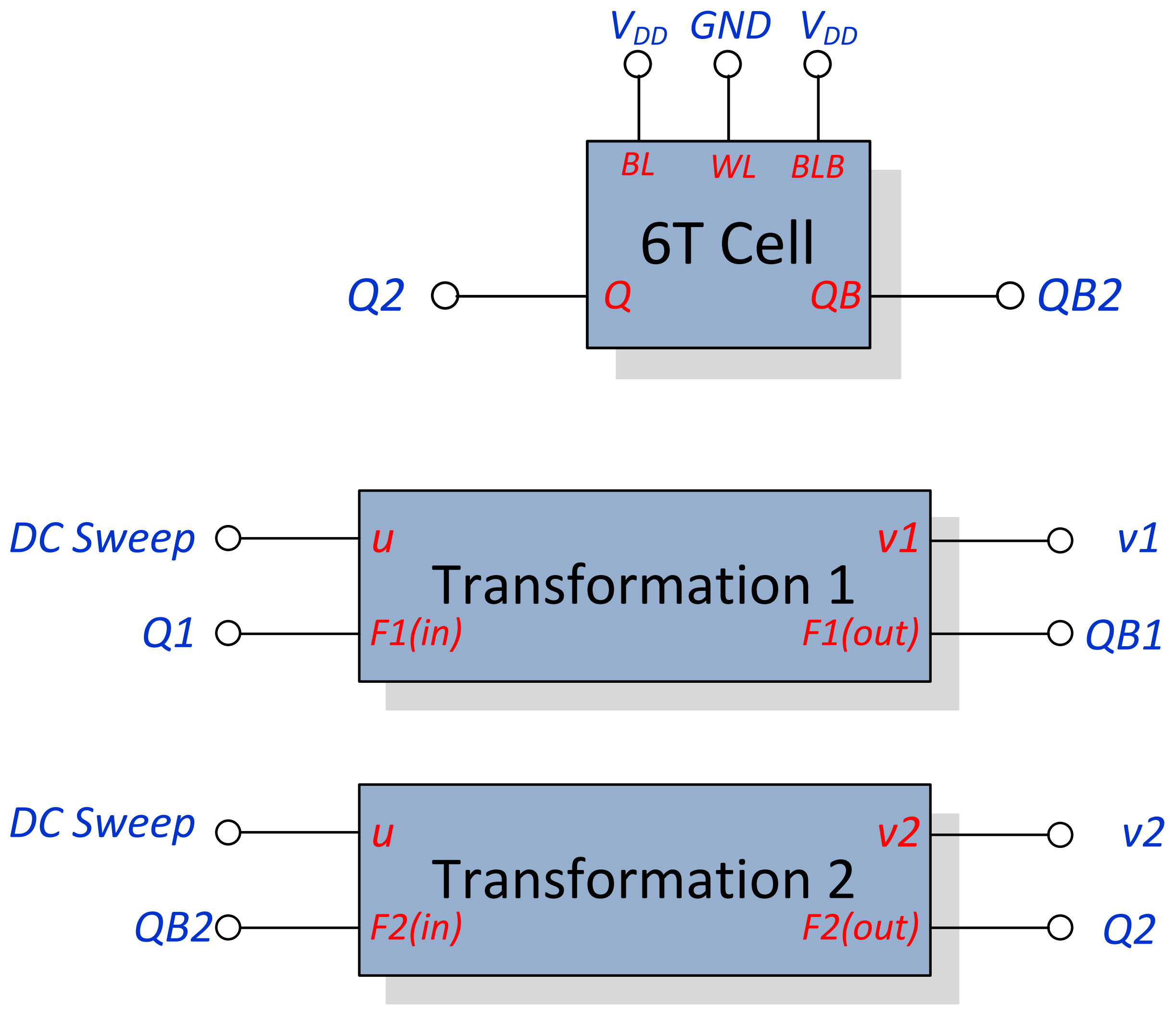
$$v = -u + \sqrt{2}x$$

$$= -u + \sqrt{2}F_2\left(\frac{v}{\sqrt{2}} - \frac{u}{\sqrt{2}}\right)$$



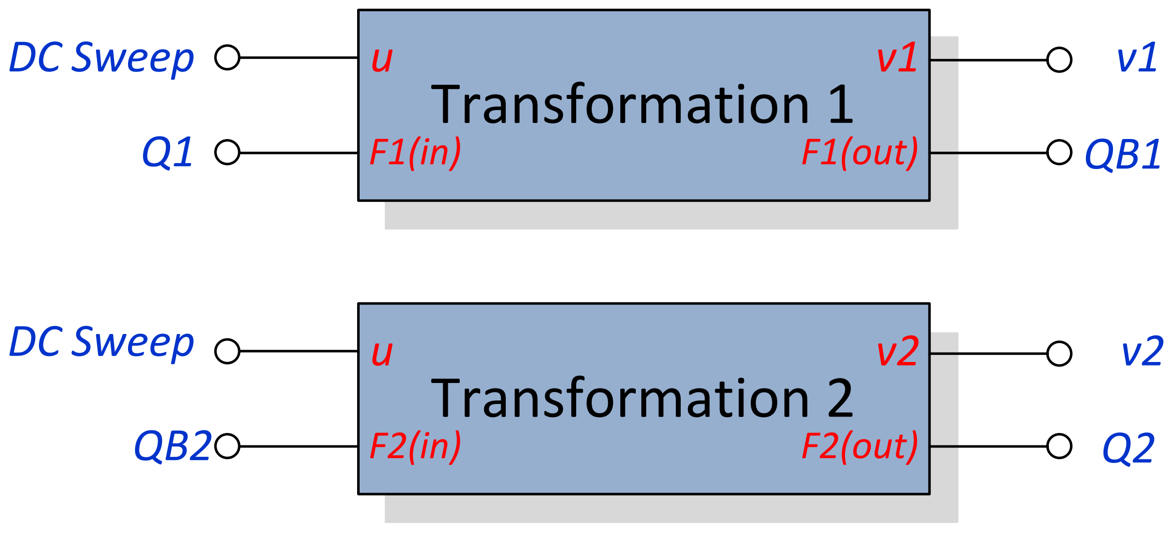
Final SNM Calculation

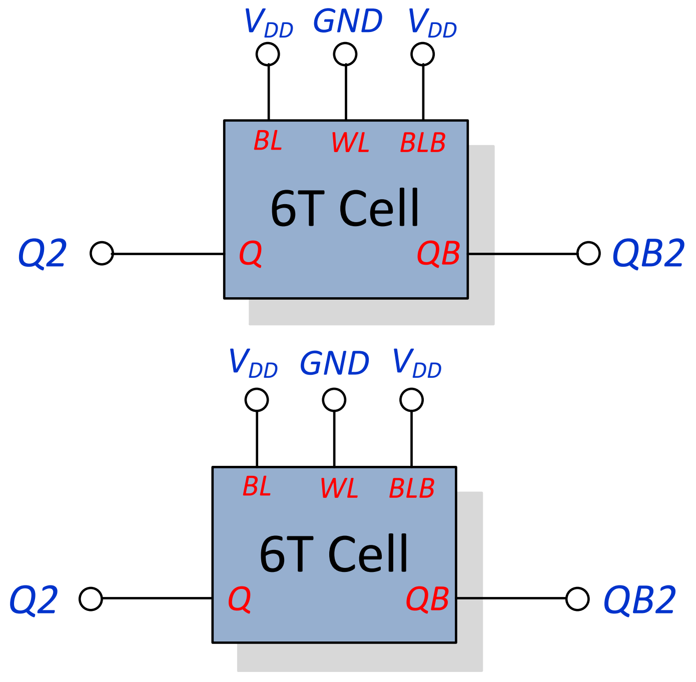
- **Now we need to:**
  - Make a schematic of our SRAM cell with two pins: Q and QB.
  - Create a coordinate changing circuit for each of the transformations.



Final SNM Calculation

- Now, connect F_1 to $Q \rightarrow QB$, and F_2 to $QB \rightarrow Q$.





- Run a *DC Sweep* on *u* from $-V_{DD}/\sqrt{2}$ to $V_{DD}/\sqrt{2}$
- This will present the butterfly curves turned 45 degrees.

Final SNM Calculation

- **Now just:**
  - Subtract the bottom graph from the top one.
  - Find the local maxima for each lobe.
  - The smaller of the local maxima
    is the diagonal of the largest square.
  - Multiply this by $cos45$ for the SNM

$$SNM = \frac{1}{\sqrt{2}} \cdot \min\left[\max\left(\left|v1 - v2\right|\right)\Big|_{-\sqrt{2}<u<0}, \max\left(\left|v1 - v2\right|\right)\Big|_{0<u<\sqrt{2}} \right]$$

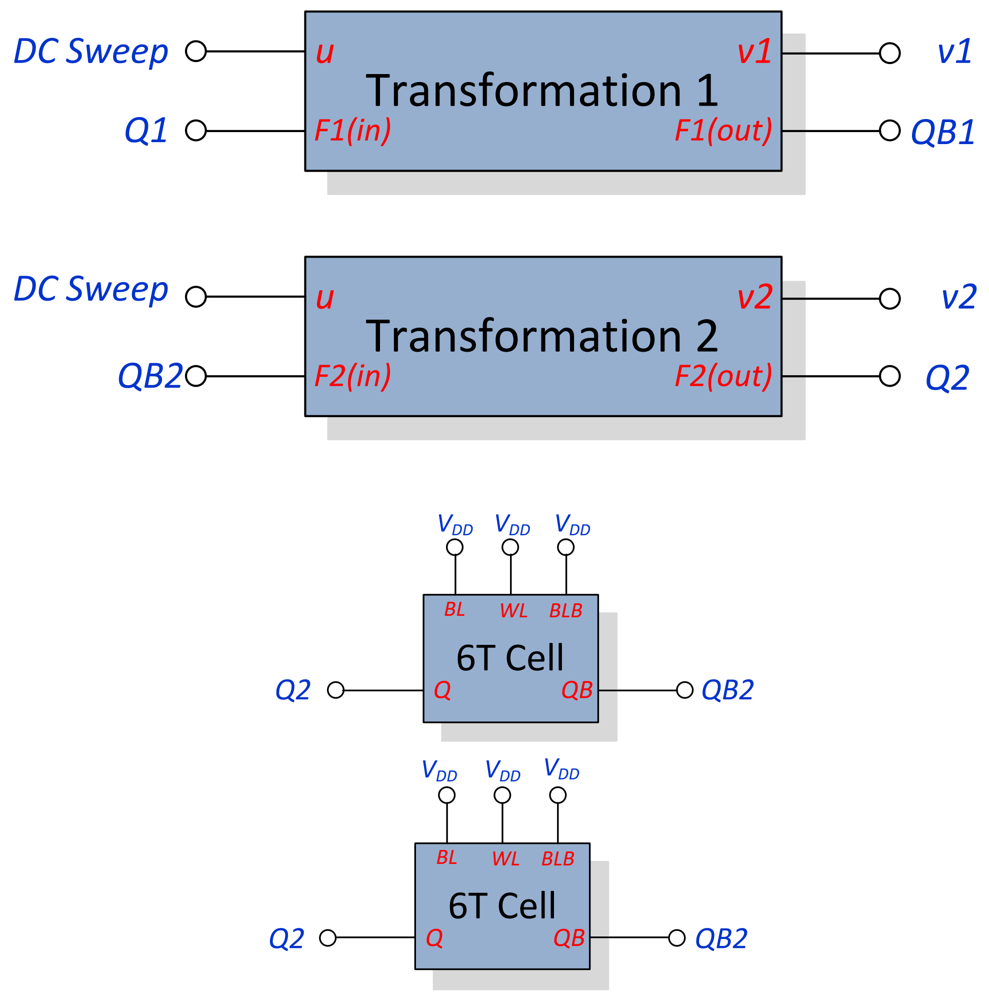
Read/Write SNM

- **How about Read SNM:**
  - Use the exact same setup.
  - Connect *BL* and *BLB* to *VDD*.
  - Connect *WL* to *VDD*.
  - Run the same calculation.

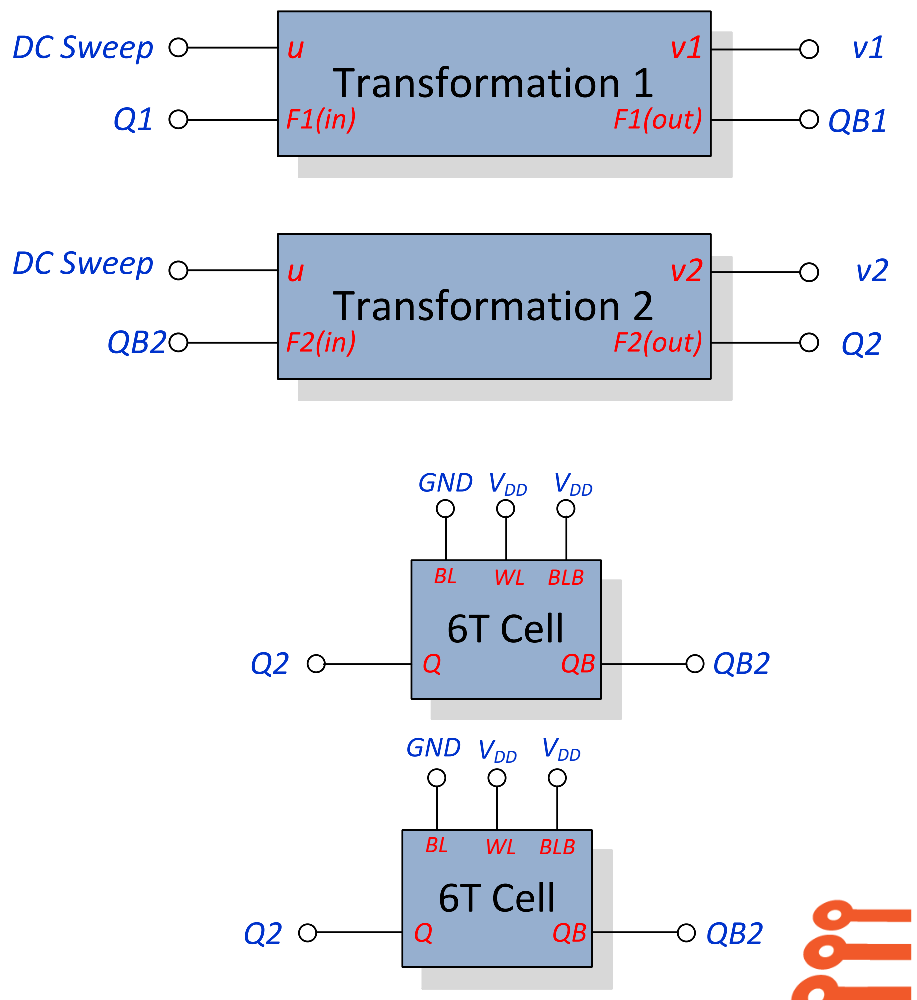
- **And Write SNM.**
  - Now connect one *BL* to *GND*.
  - This is trickier, so you’ll have to play around with the calculation.
  - There are other options for WM calculation.

Testbench Setup – Read/Write

Read Testbench:



Write Testbench:



SRAM Stability under process variations

Metastability Convergence in Spectre

- **Node Sets**
  - What solution does Virtuoso find with a standard OP?
  - To fix this, make sure you use the “**Node Set**” option.

Node Sets vs. Initial Conditions

- **SPICE supports two types of conversion aids :**
  - Node Sets:
    - Help SPICE converge by providing it with an *initial guess*.
    - Used only for DC convergence! Disregarded for Transient Analysis.
  - Initial Conditions:
    - Enforce a node voltage at time $t=0$.
    - Used only for Transient analysis! Disregarded for DC convergence.

Additional simulation tips

- **Work with Design Hierarchy**
  - Create transformation functions and DUTs as symbols.
- **Create multiple tests in single ADE-XL view.**
- **Use variables/parameters to define initial conditions/node sets.**
- **Create supply voltages in separate symbol.**
- **Use buffers to smooth transitions and reduce cross cap.**

Further Reading

- Rabaey, et al. “Digital Integrated Circuits” (2nd Edition)
- Elad Alon, Berkeley ee141 (online)
- Weste, Harris, “CMOS VLSI Design (4th Edition)”