

Memory Design

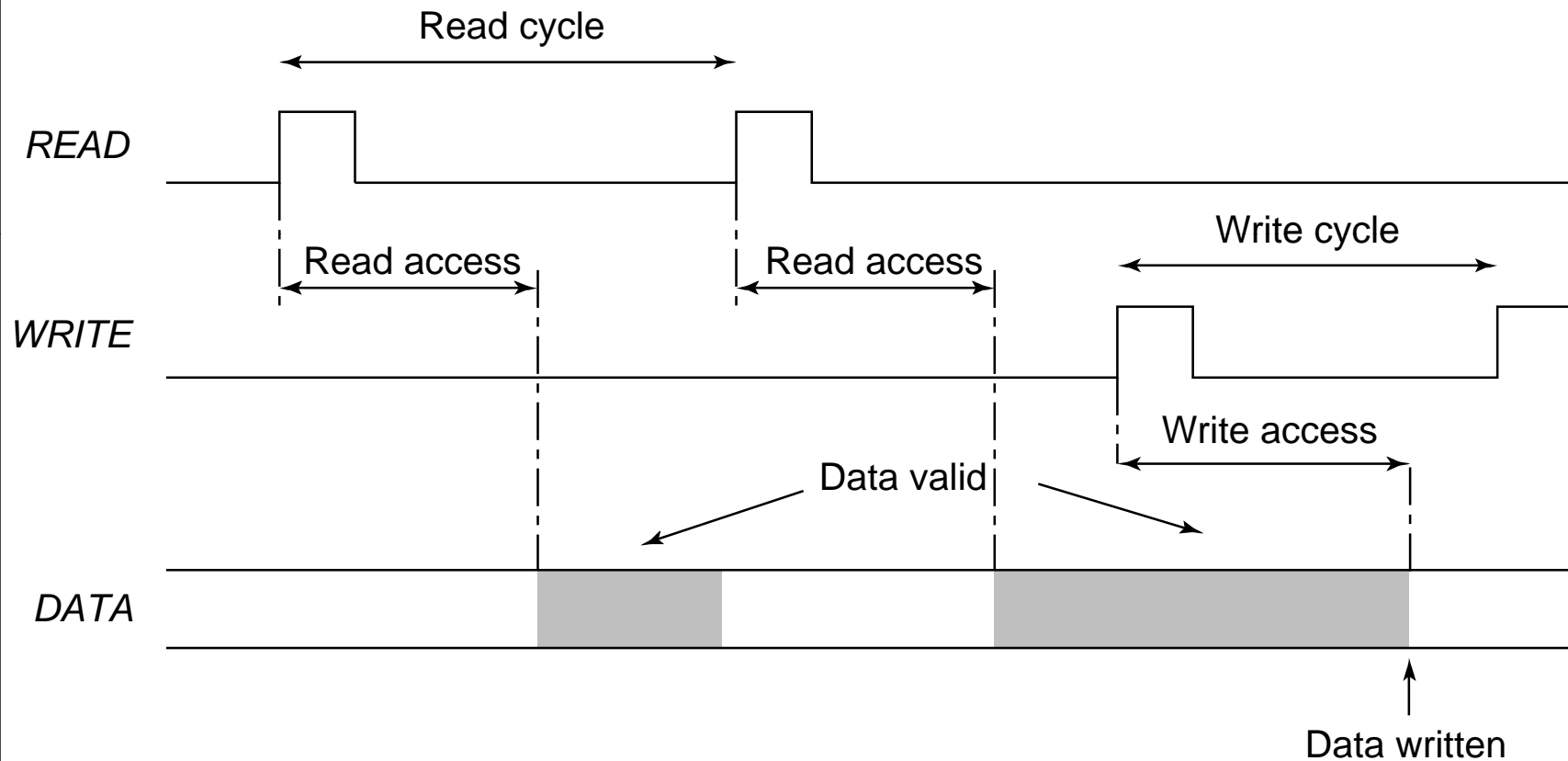
- Memory Types
- Memory Organization
- ROM design
- RAM design
- PLA design

Adapted from J. M. Rabaey, A. Chandrakasan and B. Nikolic, *Digital Integrated Circuits, 2nd ed.* Copyright 2003 Prentice Hall/Pearson.

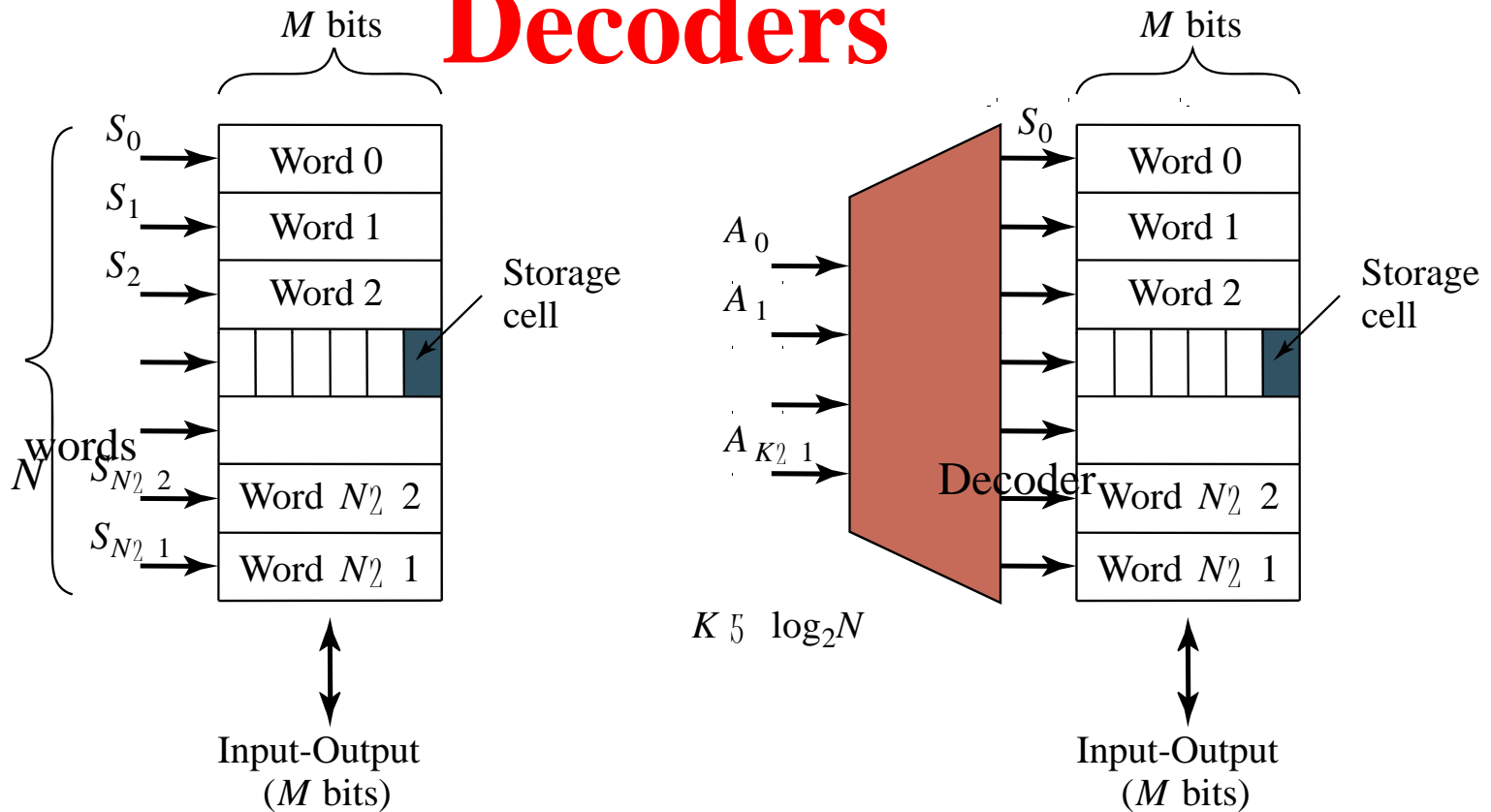
Semiconductor Memory Classification

Read-Write Memory		Non-Volatile Read-Write Memory	Read-Only Memory
Random Access	Non-Random Access	EPROM E ² PROM FLASH	Mask-Programmed Programmable (PROM)
SRAM DRAM	FIFO LIFO Shift Register CAM		

Memory Timing: Definitions



Memory Architecture: Decoders

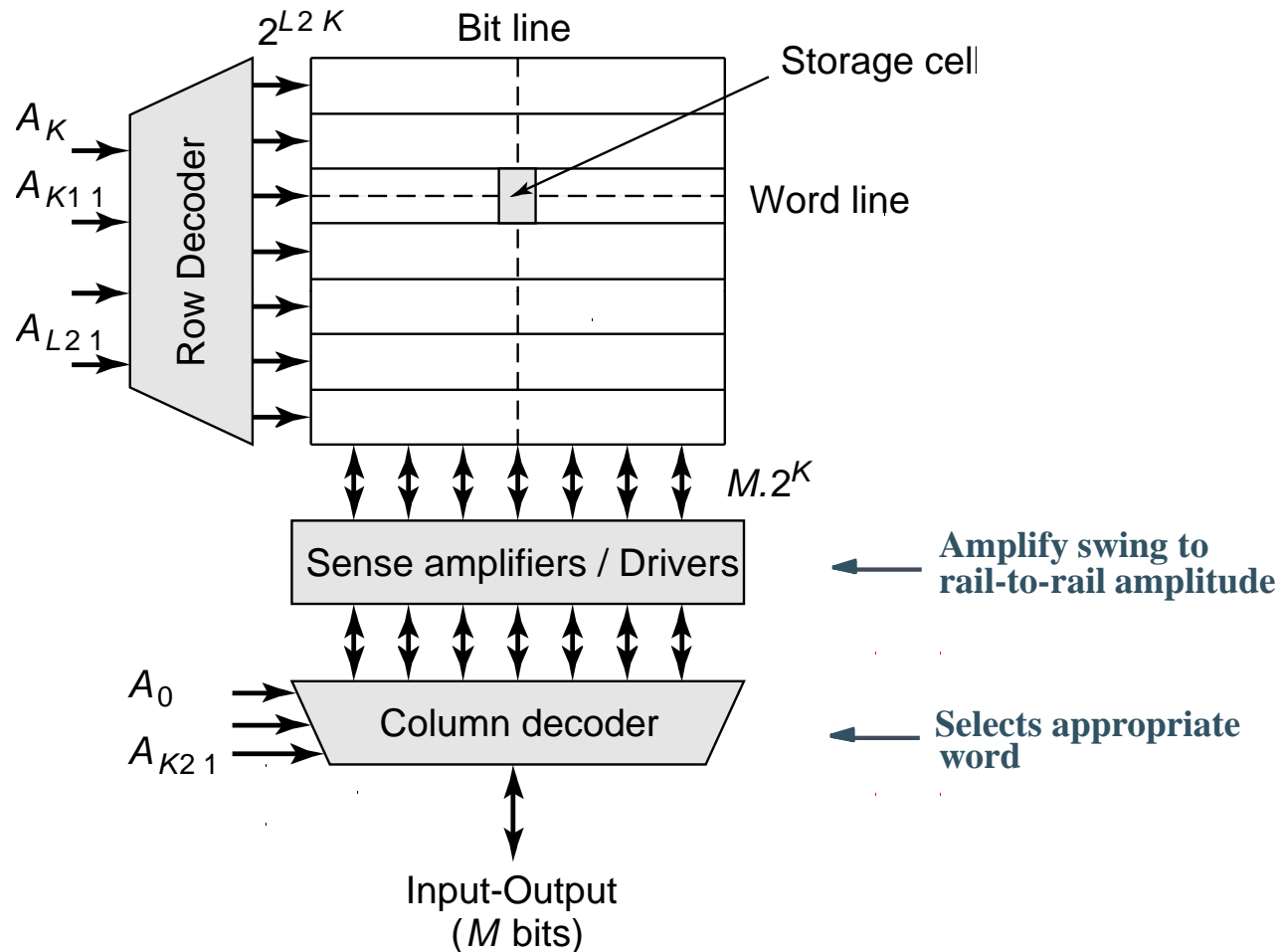


Intuitive architecture for $N \times M$ memory
 Too many select signals:
 N words == N select signals

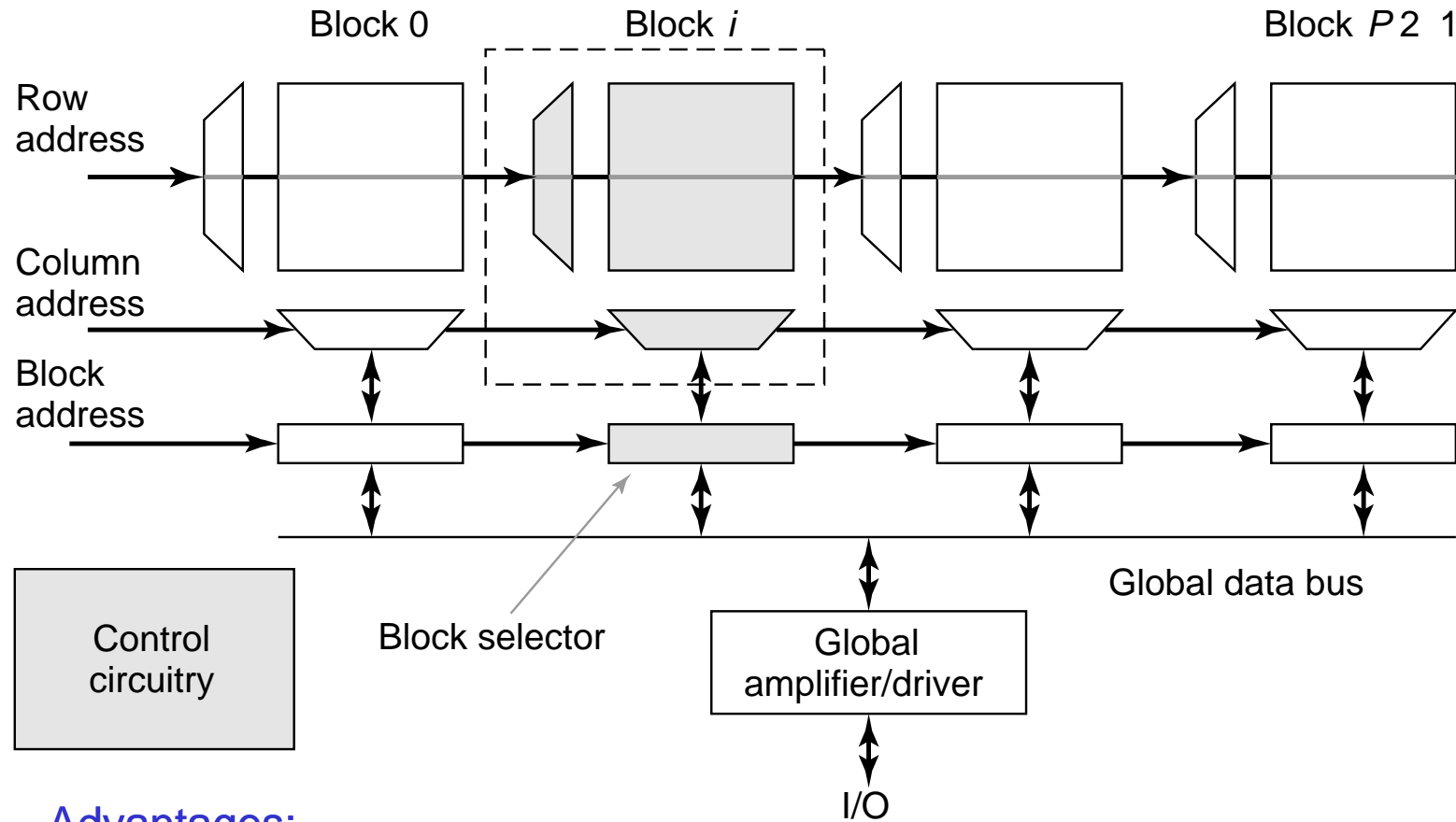
Decoder reduces the number of select signals
 $K = \log_2 N$

Array-Structured Memory Architecture

Problem: ASPECT RATIO or HEIGHT \gg WIDTH



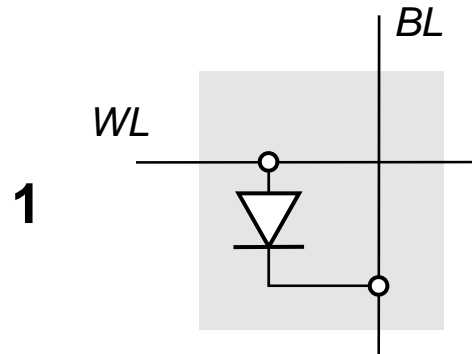
Hierarchical Memory Architecture



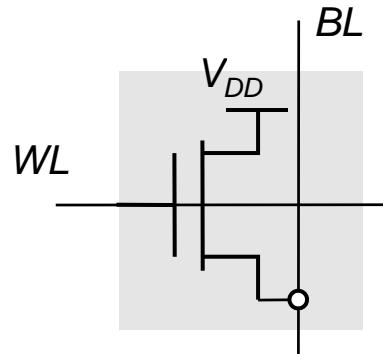
Advantages:

1. Shorter wires within blocks
2. Block address activates only 1 block => power savings

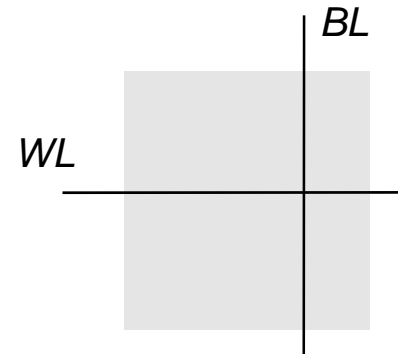
Read-Only Memory Cells



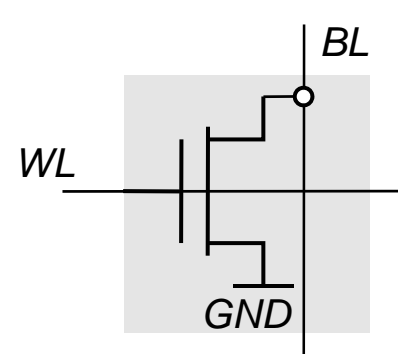
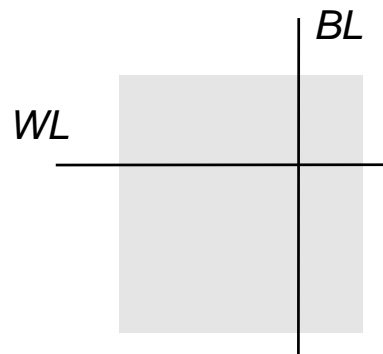
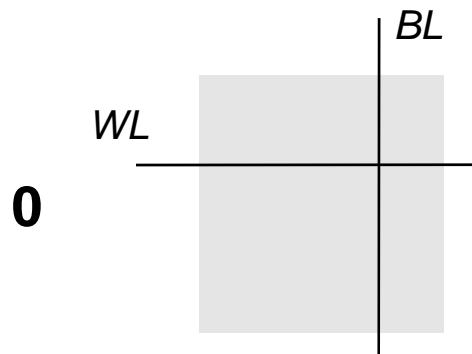
Diode ROM



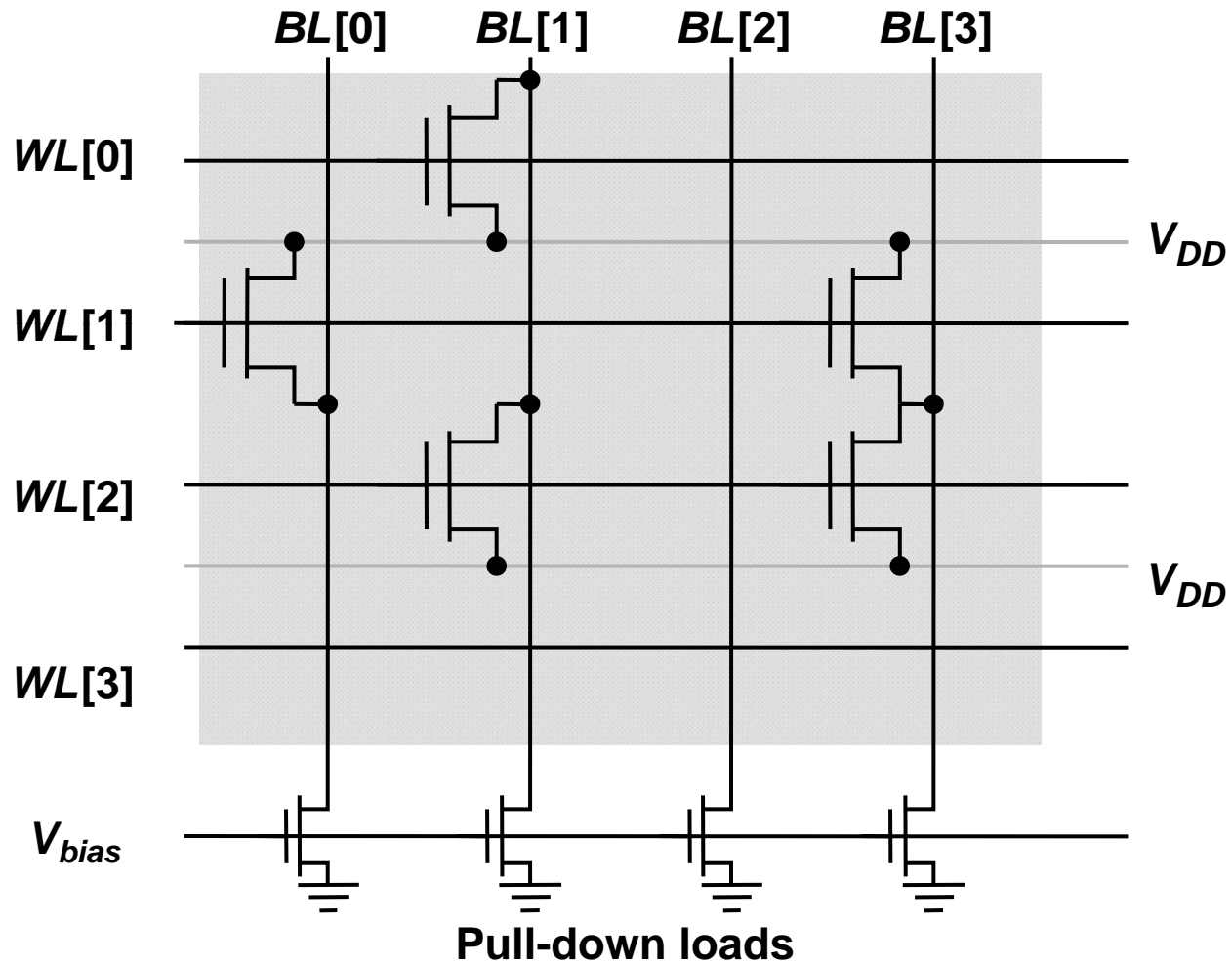
MOS ROM 1



MOS ROM 2



MOS OR ROM



ROM Example

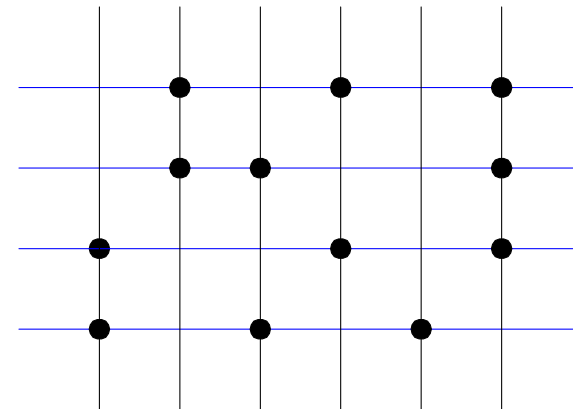
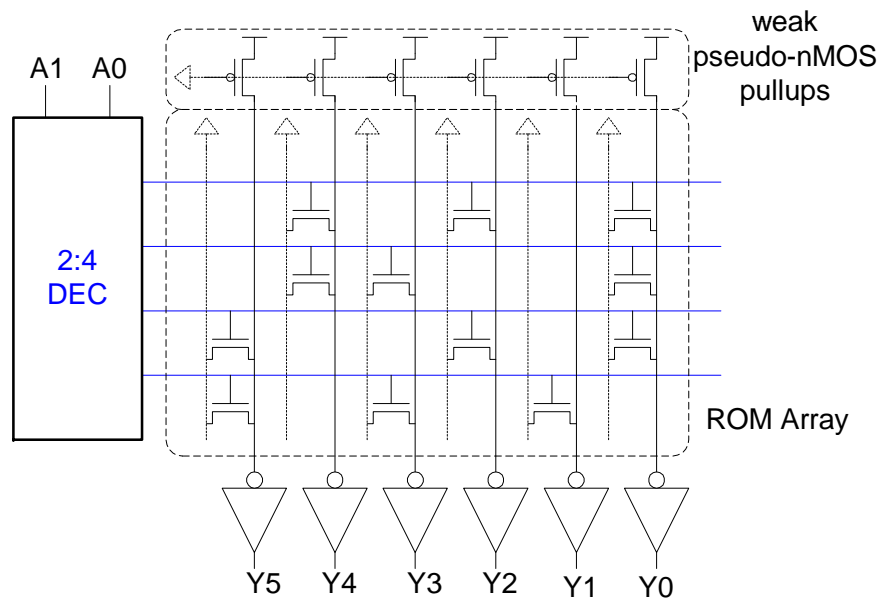
- 4-word x 6-bit ROM
 - Represented with dot diagram
 - Dots indicate 1's in ROM

Word 0: 010101

Word 1: 011001

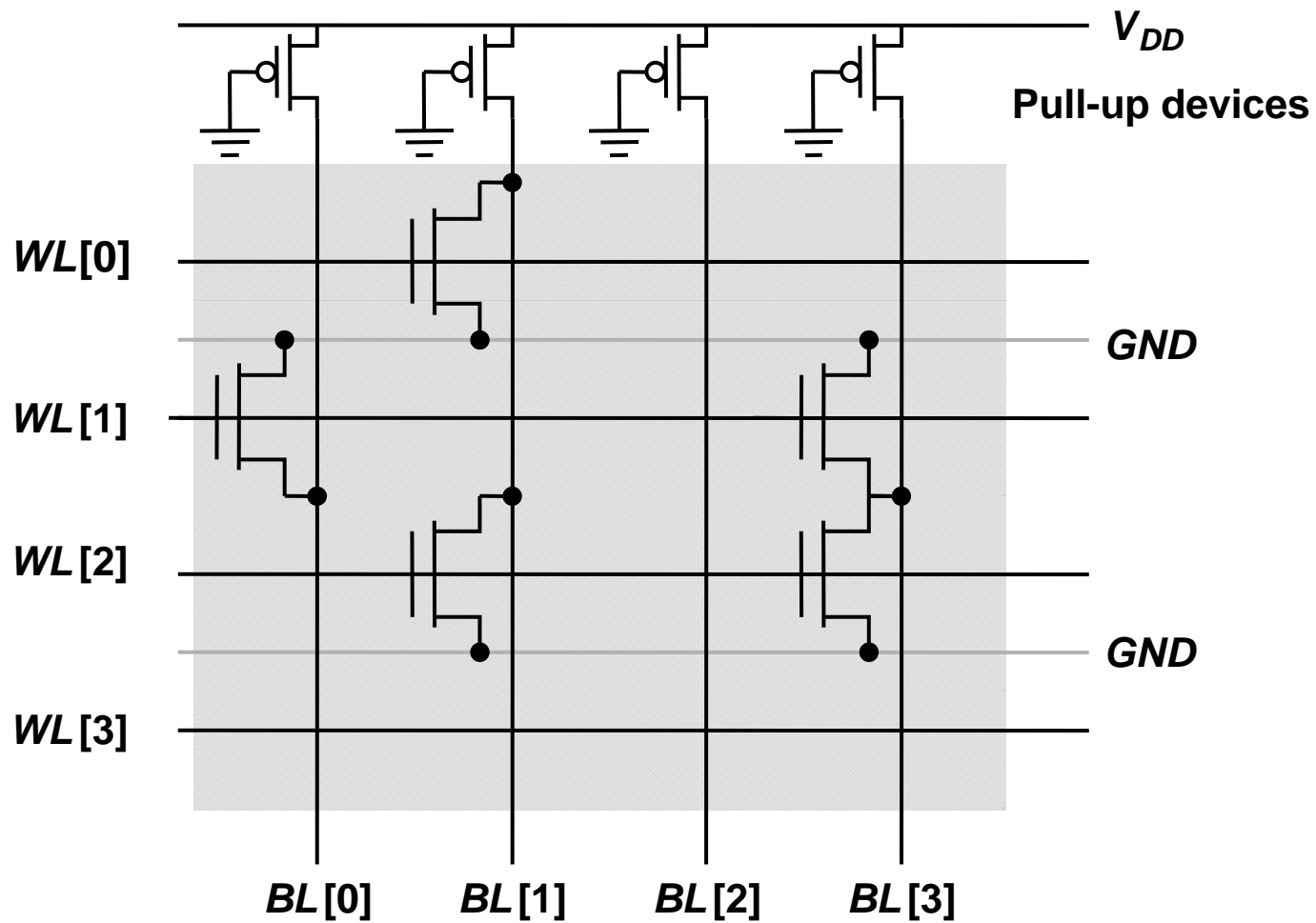
Word 2: 100101

Word 3: 101010

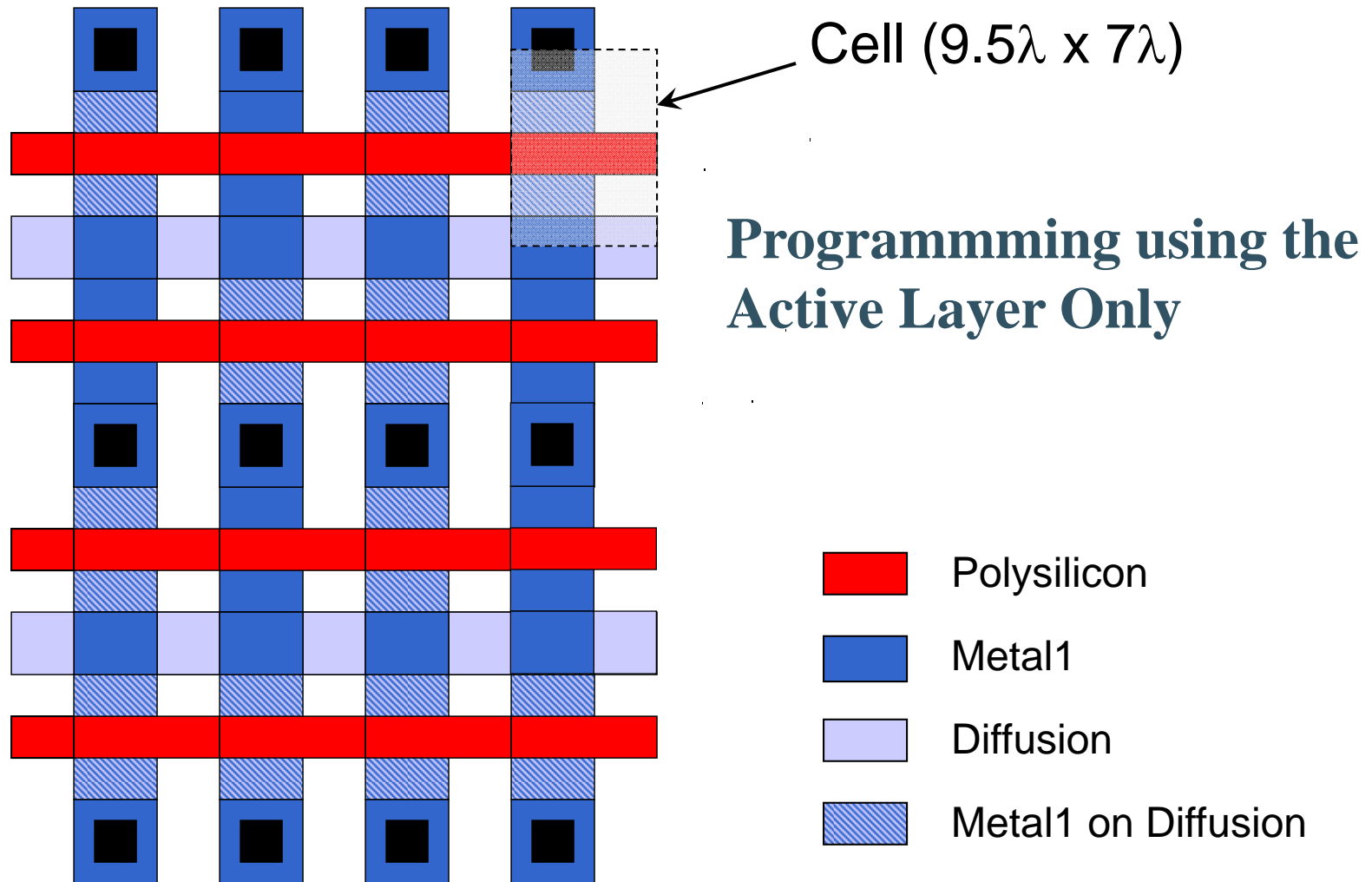


Looks like 6 4-input pseudo-nMOS NORs

MOS NOR ROM

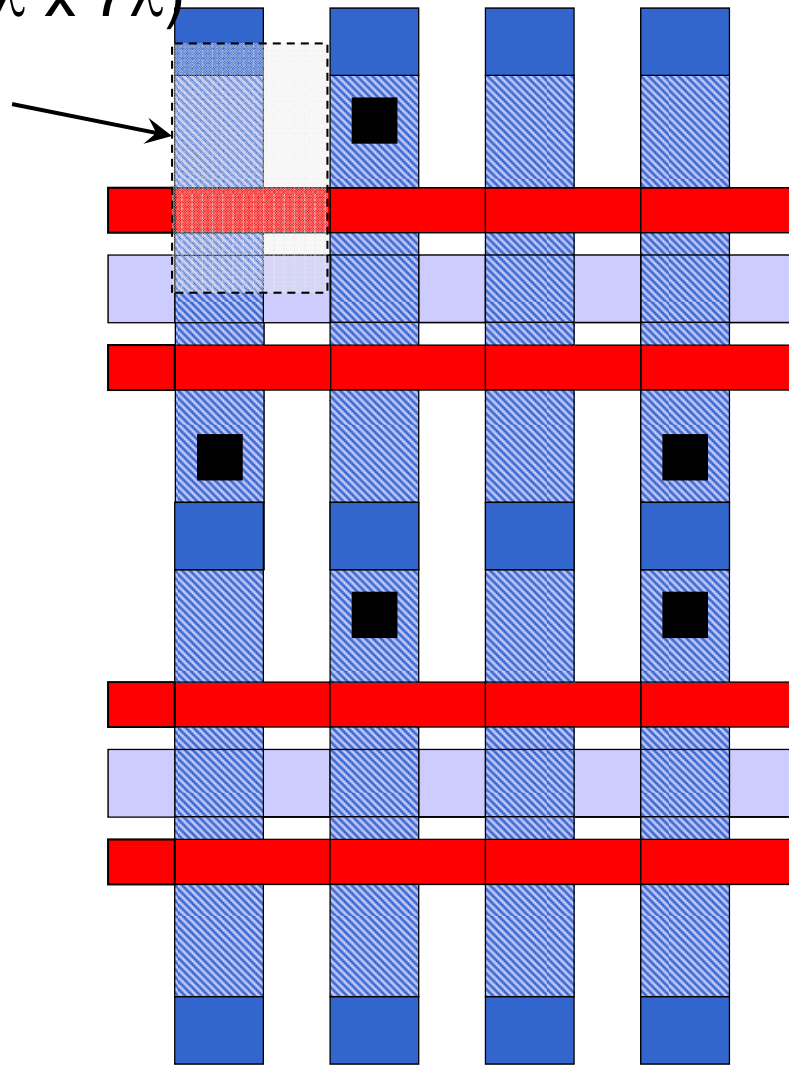


MOS NOR ROM Layout







MOS NOR ROM Layout

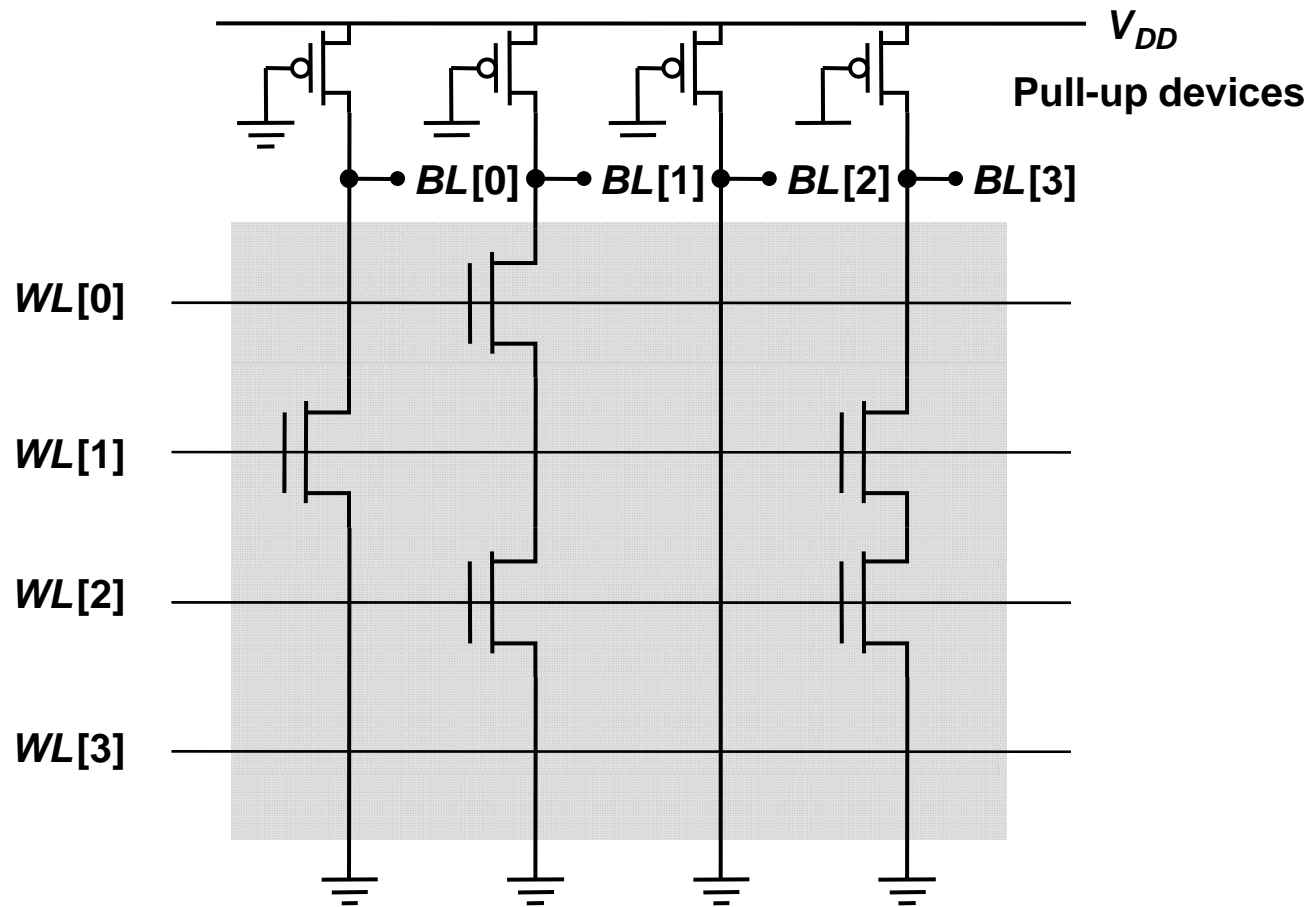
Cell ($11\lambda \times 7\lambda$)



Programming using
the Contact Layer Only

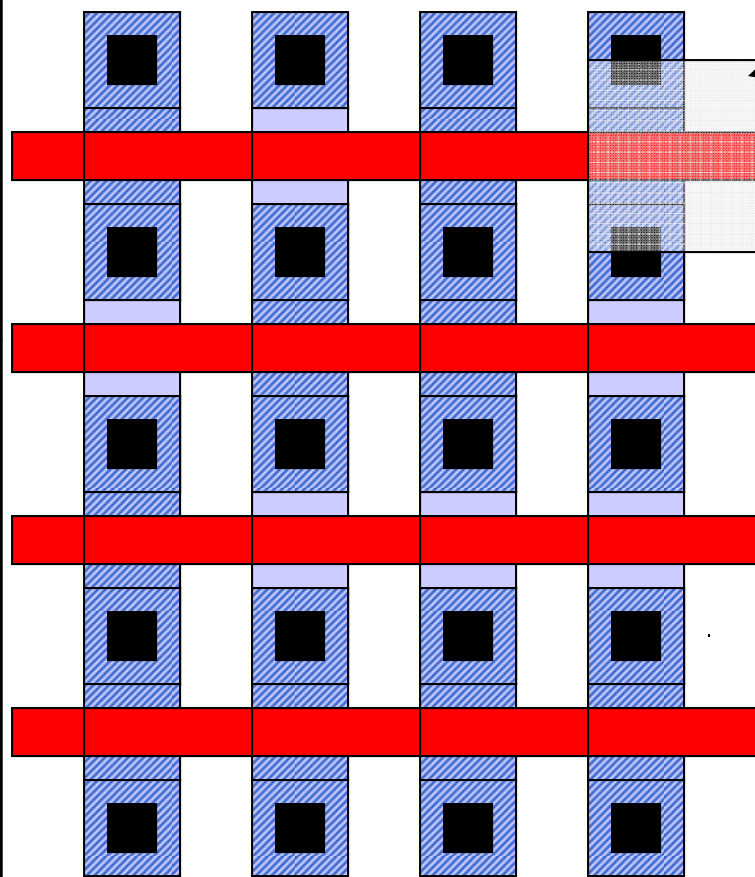
-  Polysilicon
-  Metal1
-  Diffusion
-  Metal1 on Diffusion

MOS NAND ROM

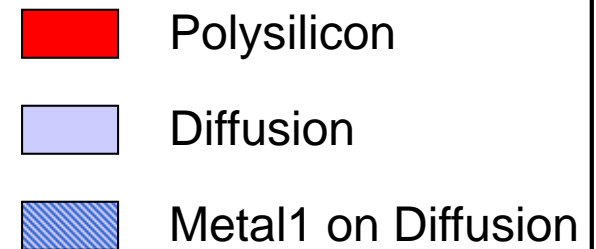


All word lines high by default with exception of selected row

MOS NAND ROM Layout

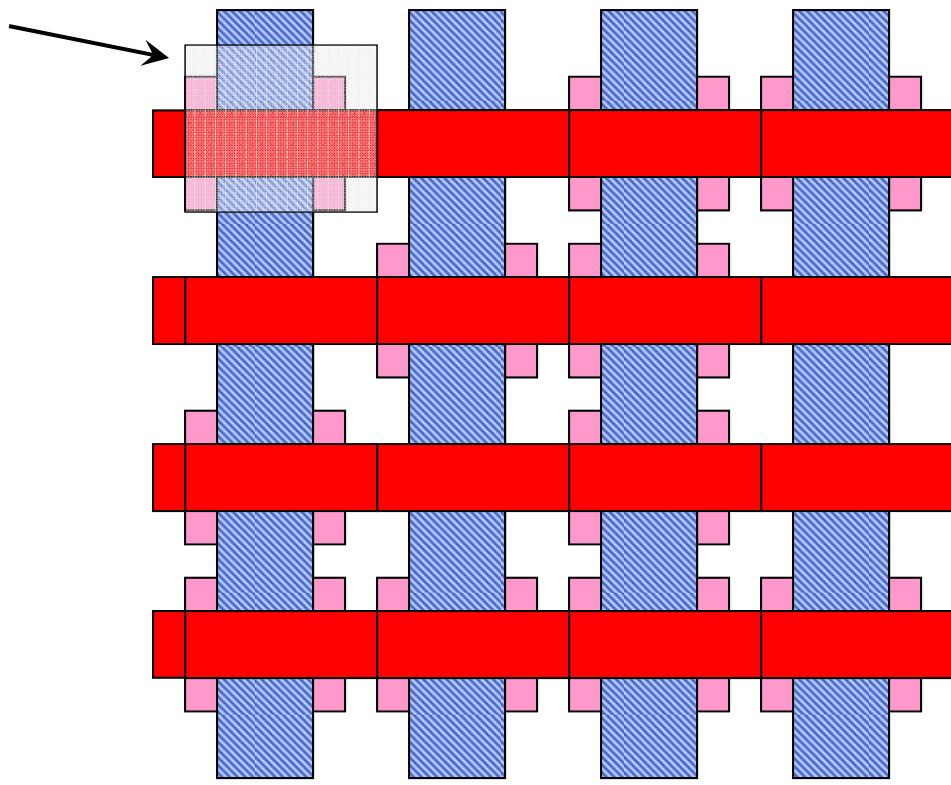


No contact to VDD or GND necessary;
drastically reduced cell size
Loss in performance compared to NOR ROM






NAND ROM Layout

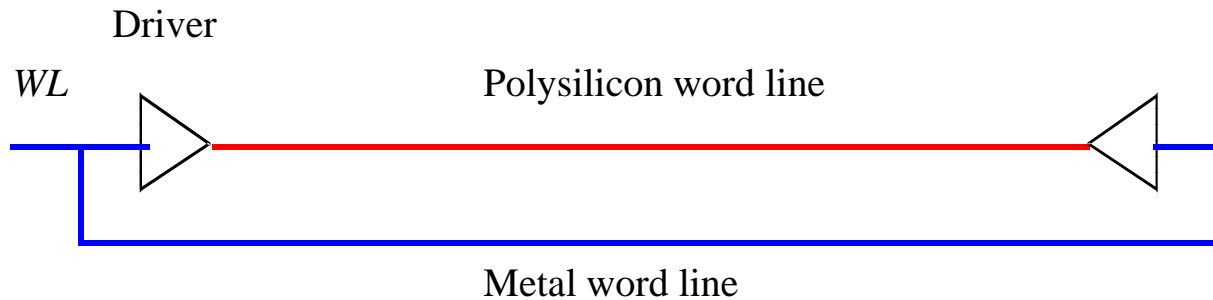
Cell ($5\lambda \times 6\lambda$)



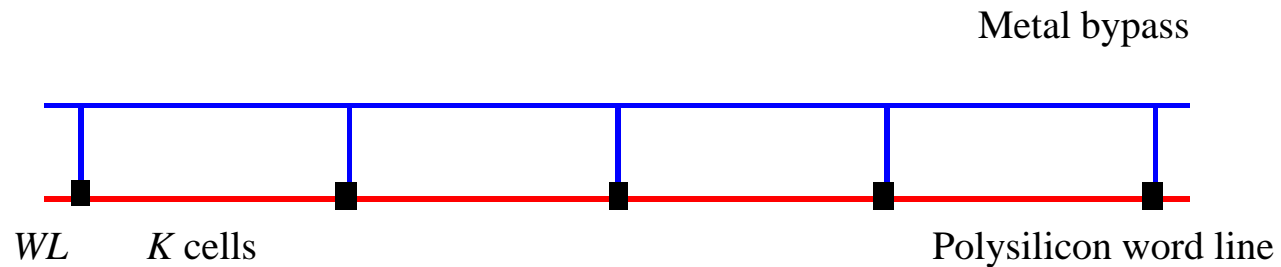
Programmming using
Implants Only

-  Polysilicon
-  Threshold-altering implant
-  Metal1 on Diffusion

Decreasing Word Line Delay

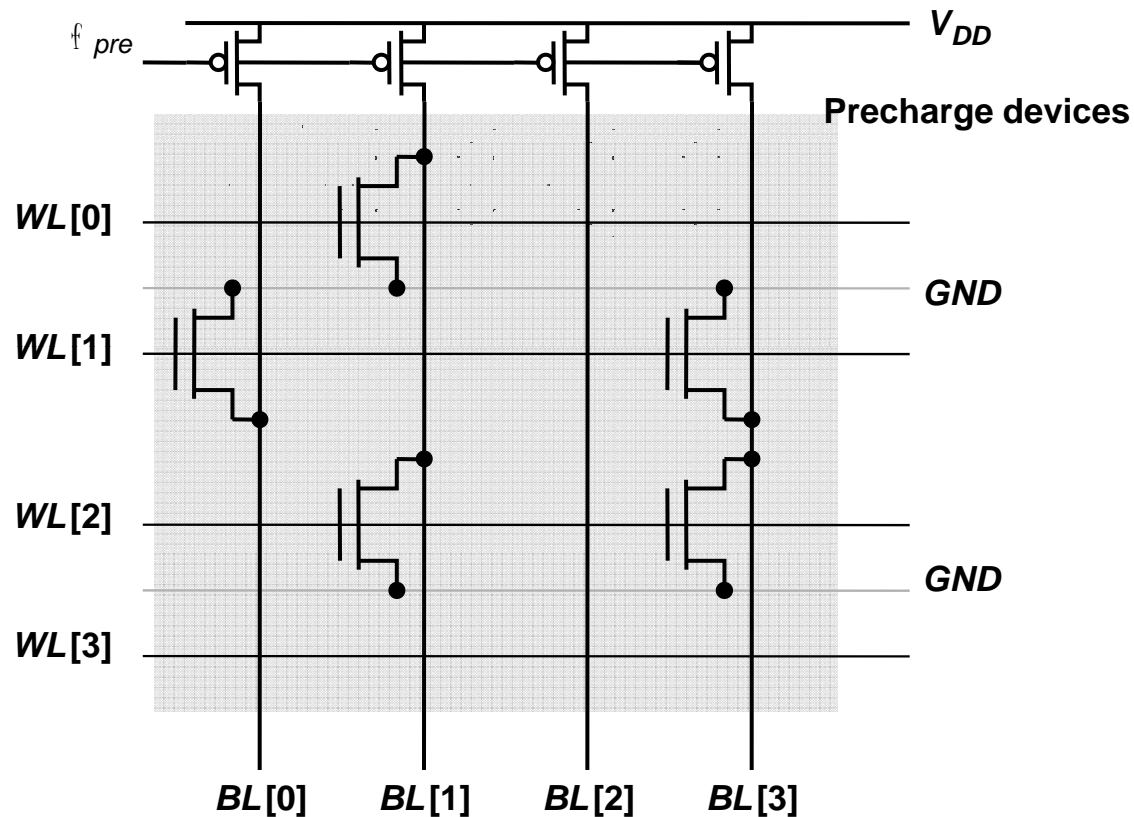


(a) Driving the word line from both sides



(b) Using a metal bypass

Precharged MOS NOR ROM



PMOS precharge device can be made as large as necessary, but clock driver becomes harder to design.

Read-Write Memories (RAM)

□ STATIC (SRAM)

Data stored as long as supply is applied

Large (6 transistors/cell)

Fast

Differential

□ DYNAMIC (DRAM)

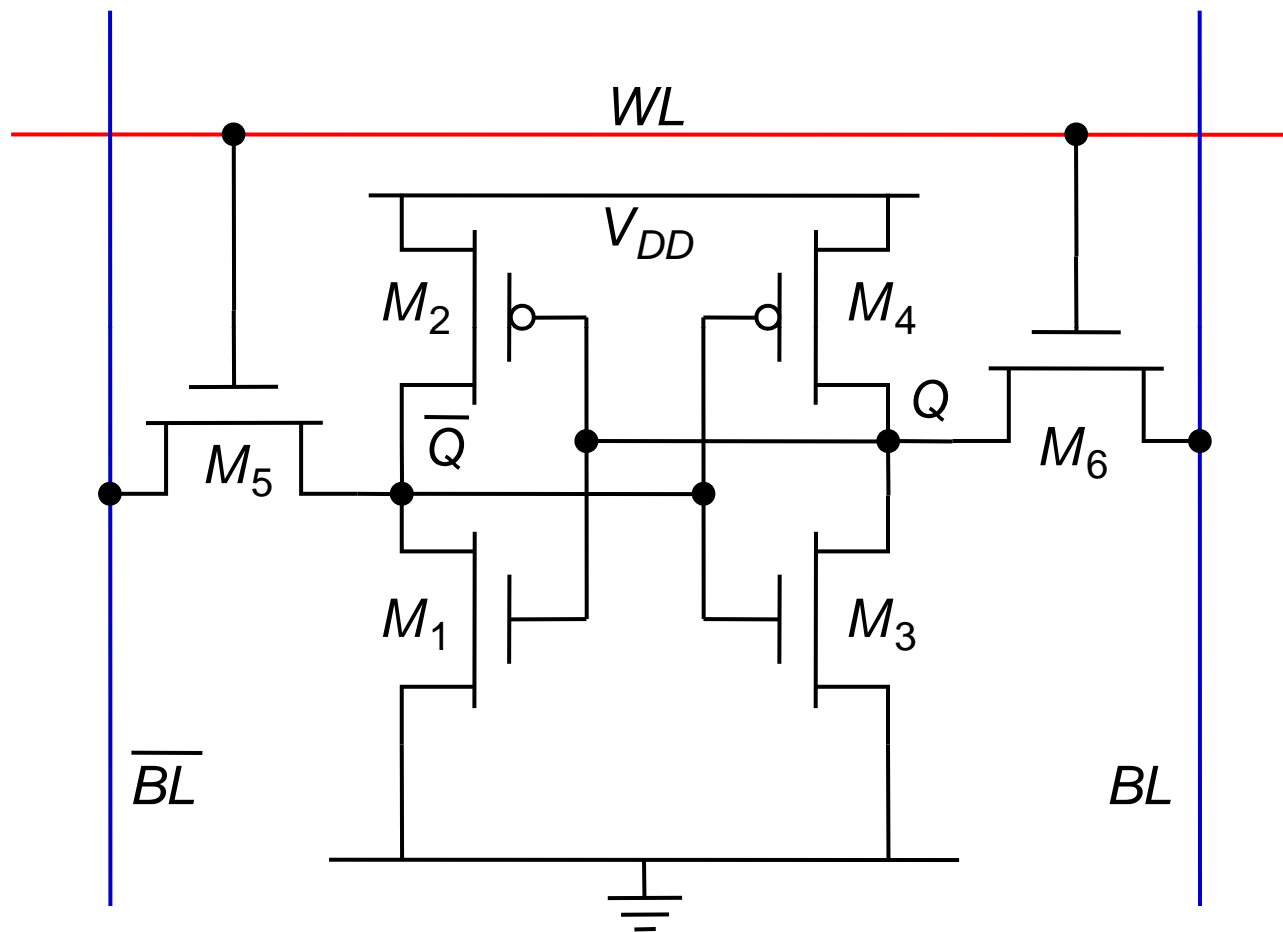
Periodic refresh required

Small (1-3 transistors/cell)

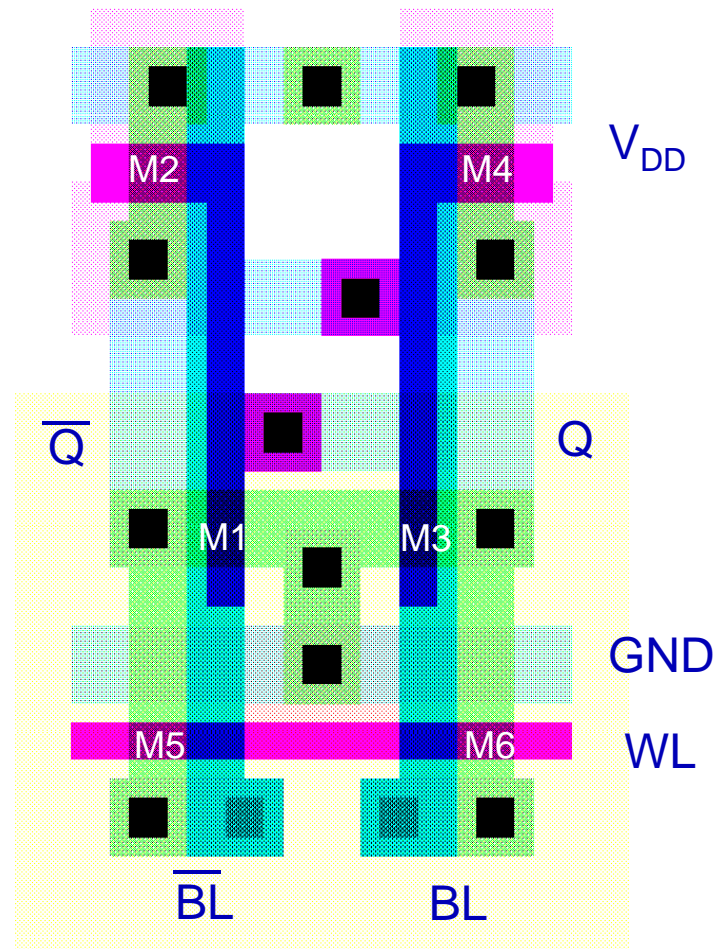
Slower

Single Ended

6-transistor CMOS SRAM Cell



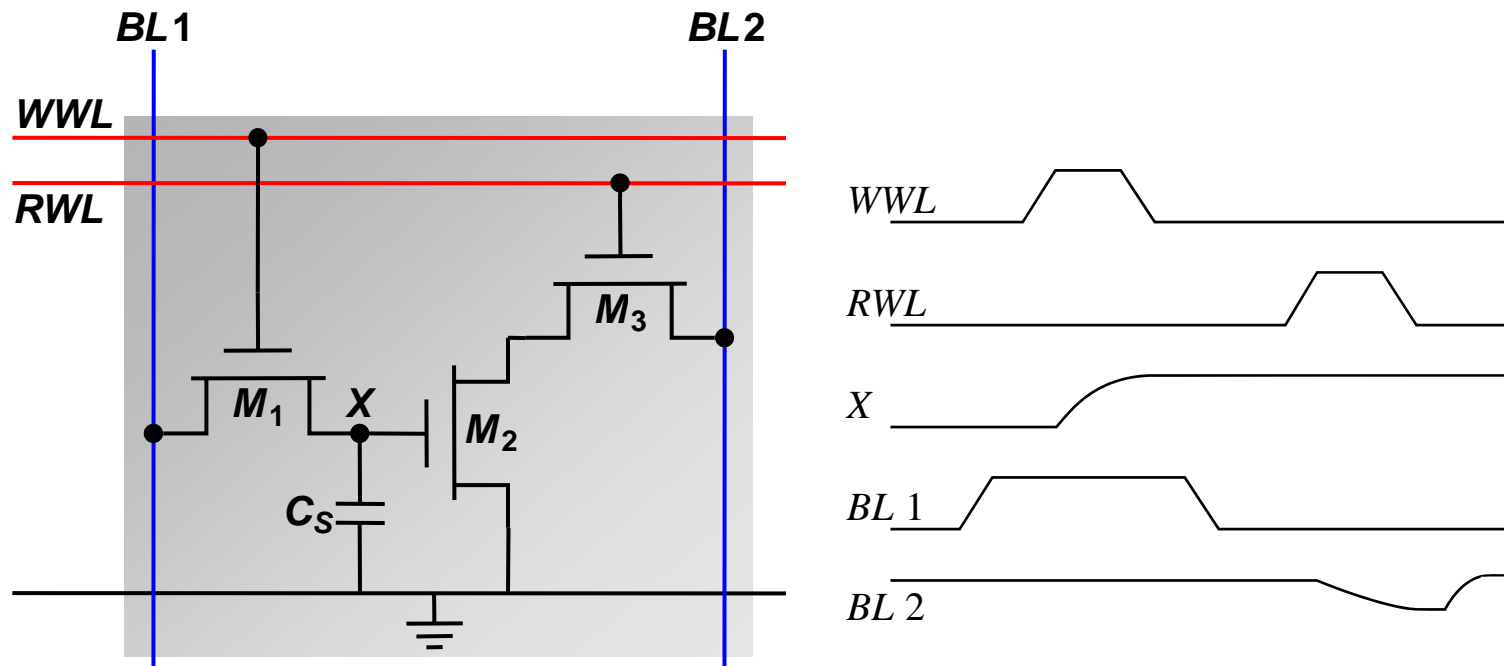
6T-SRAM — Layout



Statue of Goethe and Schiller: the German National Theater, Weimar



3-Transistor DRAM Cell

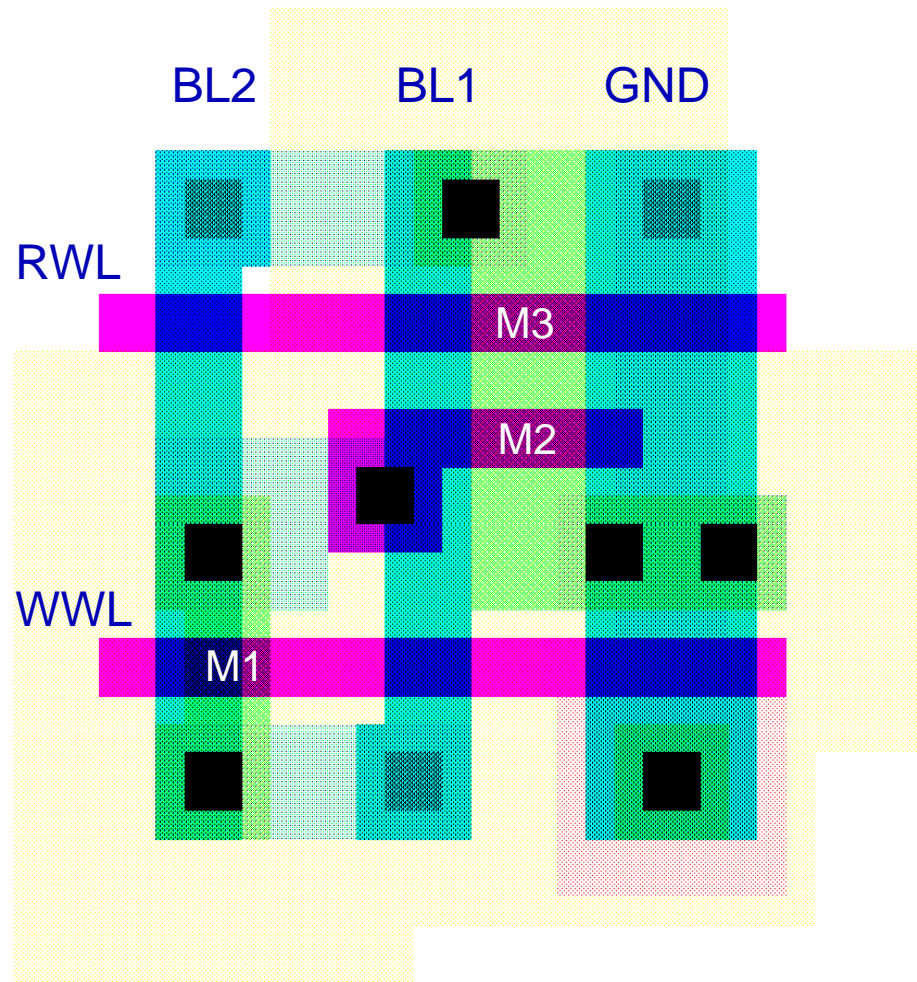


No constraints on device ratios

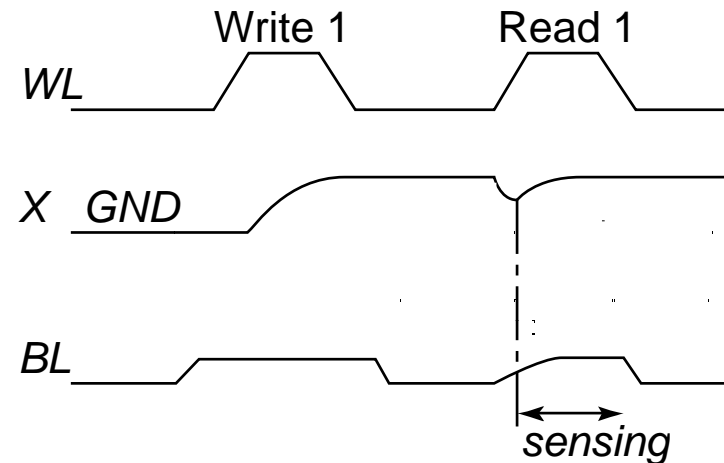
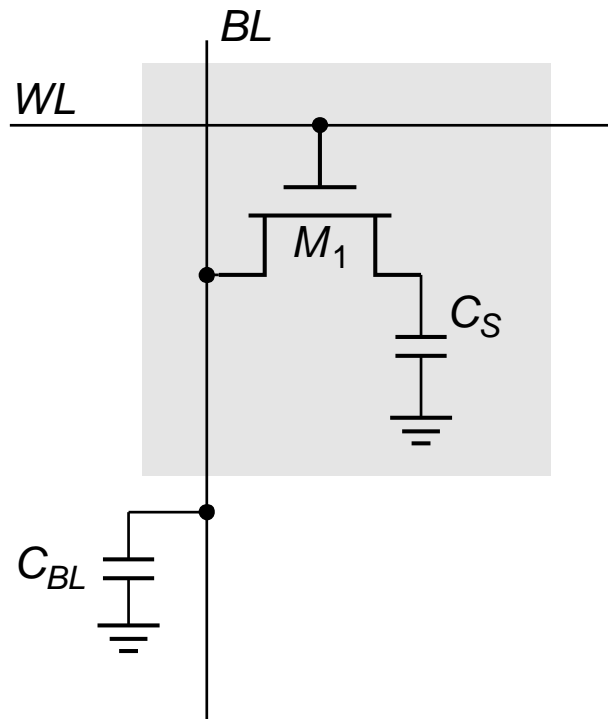
Reads are non-destructive

Value stored at node X when writing a "1" = $V_{WWL} - V_{Tn}$

3T-DRAM — Layout



1-Transistor DRAM Cell



Write: C_S is charged or discharged by asserting WL and BL.

Read: Charge redistribution takes place between bit line and storage capacitance

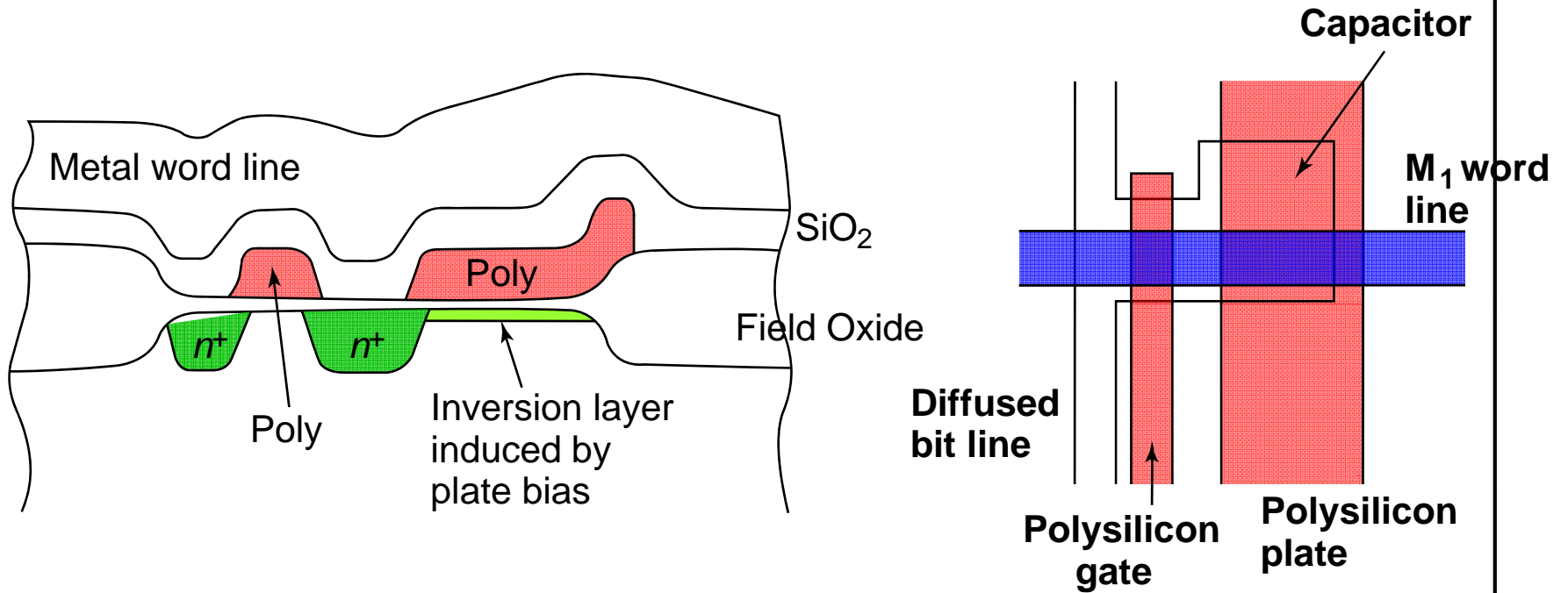
$$\Delta V = V_{BL} - V_{PRE} = V_{BIT} - V_{PRE} \frac{C_S}{C_S + C_{BL}}$$

Voltage swing is small; typically around 250 mV.

DRAM Cell Observations

- ❑ 1T DRAM requires a sense amplifier for each bit line, due to charge redistribution read-out.
- ❑ DRAM memory cells are single-ended in contrast to SRAM cells.
- ❑ The read-out of the 1T DRAM cell is destructive; read and refresh operations are necessary for correct operation.
- ❑ Unlike 3T cell, 1T cell requires presence of an extra capacitance that must be explicitly included in the design.
- ❑ When writing a “1” into a DRAM cell, a threshold voltage is lost. This charge loss can be circumvented by bootstrapping the word lines to a higher value than V_{DD}

1-T DRAM Cell



Cross-section

Layout

Uses Polysilicon-Diffusion Capacitance

Expensive in Area (trend now is to use trench capacitors)

Periphery

- Decoders
- Sense Amplifiers
- Input/Output Buffers
- Control / Timing Circuitry

Row Decoders

Collection of 2^M complex logic gates
Organized in regular and dense fashion

(N)AND Decoder

$$WL_0 = A_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9$$

$$WL_{511} = \bar{A}_0 \bar{A}_1 \bar{A}_2 \bar{A}_3 \bar{A}_4 \bar{A}_5 \bar{A}_6 \bar{A}_7 \bar{A}_8 \bar{A}_9$$

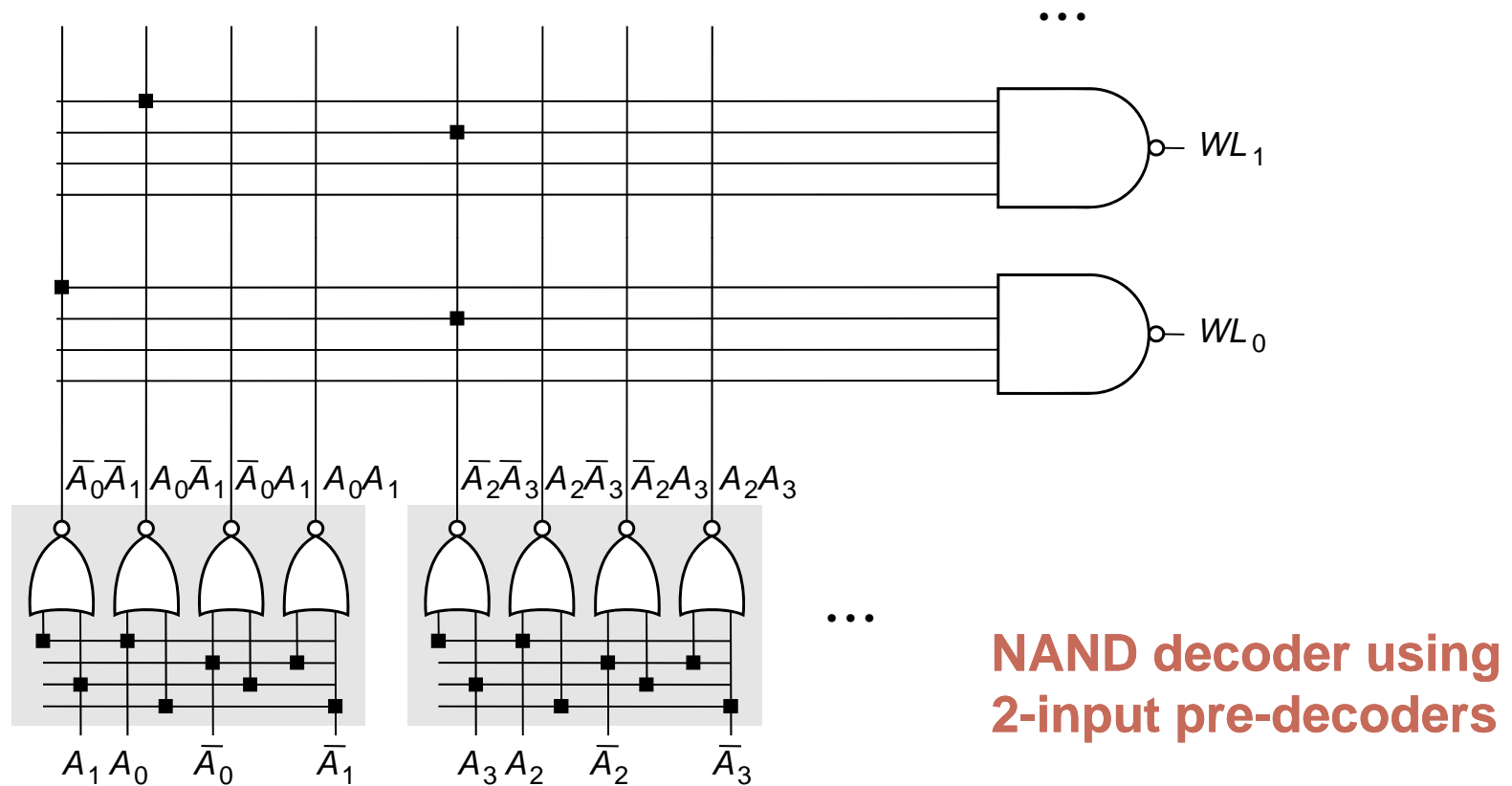
NOR Decoder

$$WL_0 = \overline{A_0 + A_1 + A_2 + A_3 + A_4 + A_5 + A_6 + A_7 + A_8 + A_9}$$

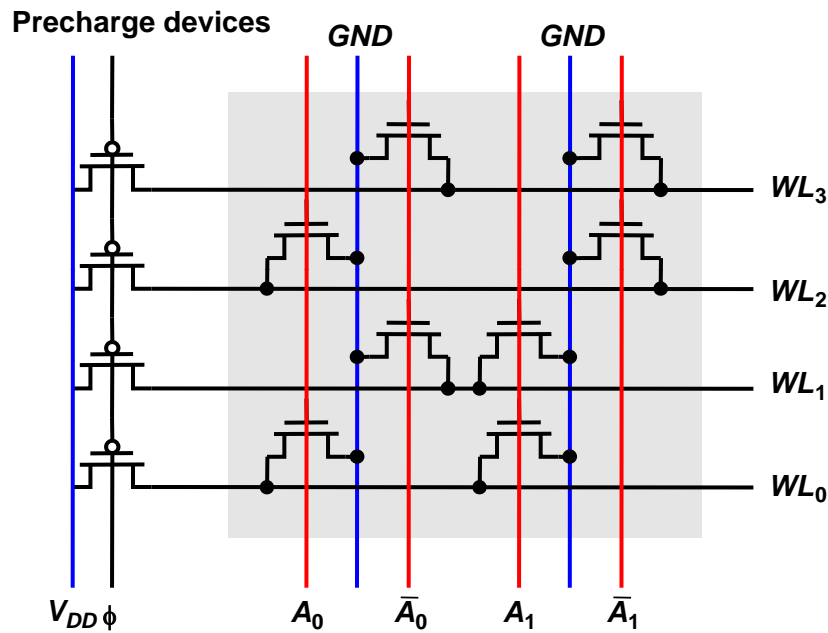
$$WL_{511} = \overline{A_0 + \bar{A}_1 + \bar{A}_2 + \bar{A}_3 + \bar{A}_4 + \bar{A}_5 + \bar{A}_6 + \bar{A}_7 + \bar{A}_8 + \bar{A}_9}$$

Hierarchical Decoders

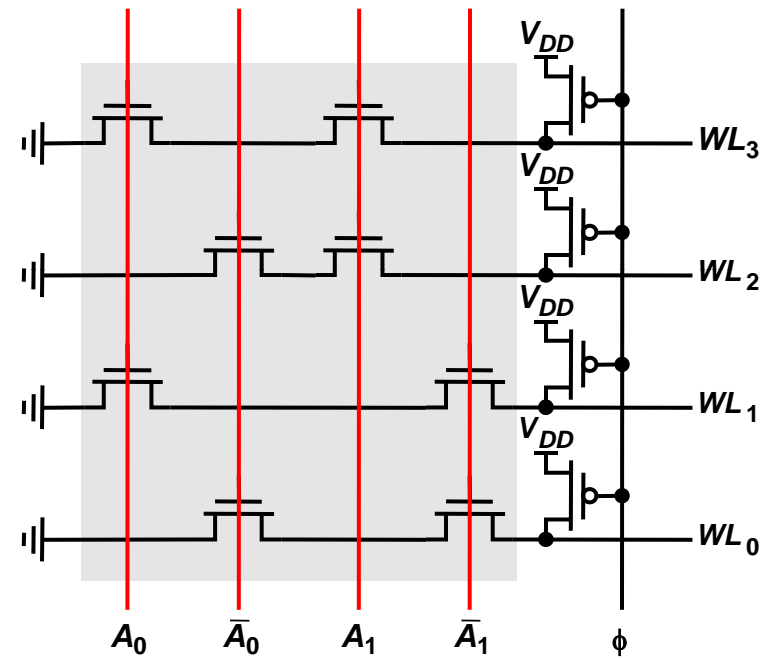
Multi-stage implementation improves performance



Dynamic Decoders

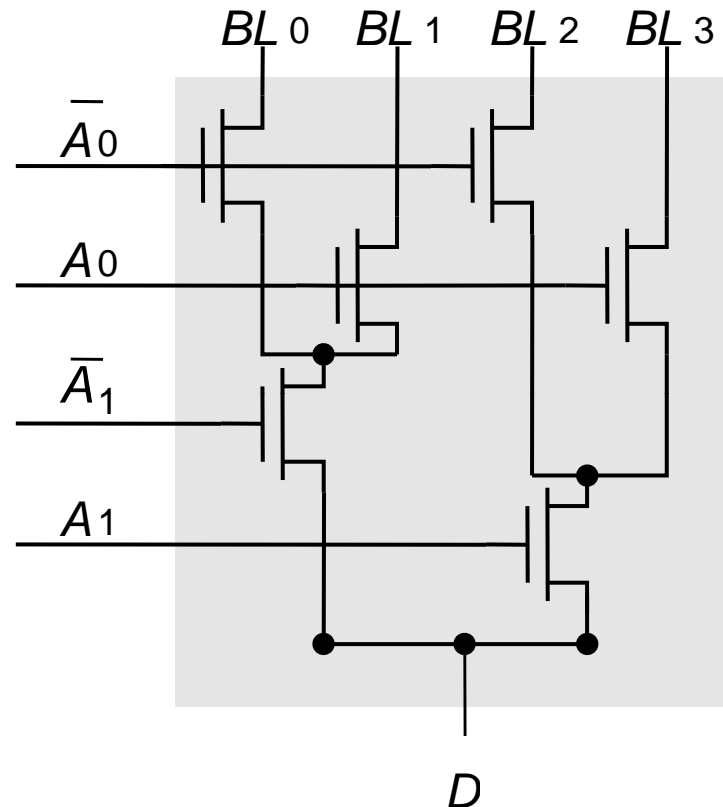


2-input NOR decoder



2-input NAND decoder

4-to-1 tree based column decoder



Number of devices drastically reduced

Delay increases quadratically with # of sections; prohibitive for large decoders

Solutions: buffers

progressive sizing

combination of tree and pass transistor approaches

PLA versus ROM

- **Programmable Logic Array**
structured approach to random logic
“two level logic implementation”
NOR-NOR (product of sums)
NAND-NAND (sum of products)

SIMILAR TO ROM

- **Main difference**
ROM: fully populated
PLA: one element per minterm

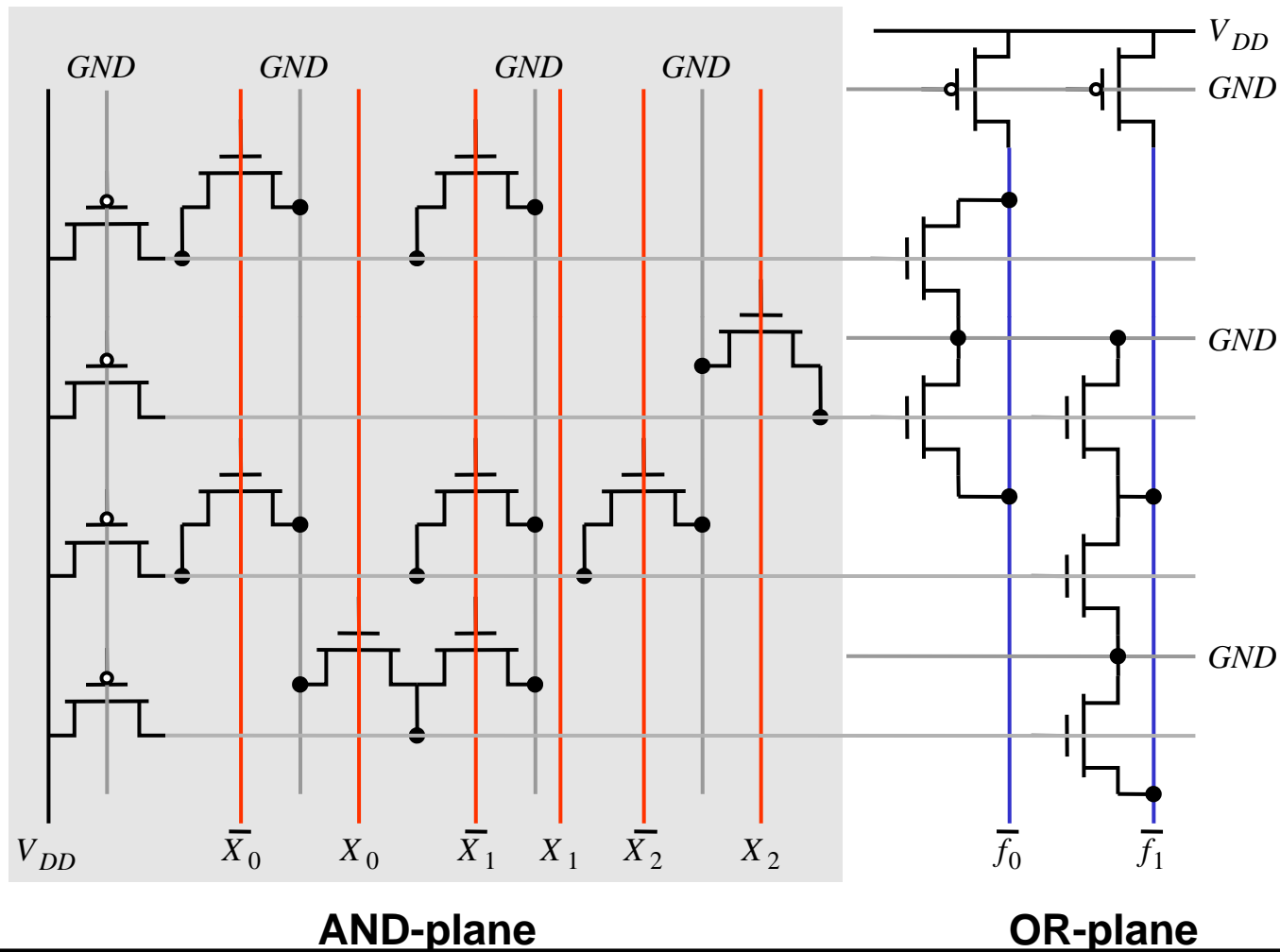
Note: Importance of PLA's has drastically reduced

1. **slow**
2. **better software techniques (multi-level logic synthesis)**

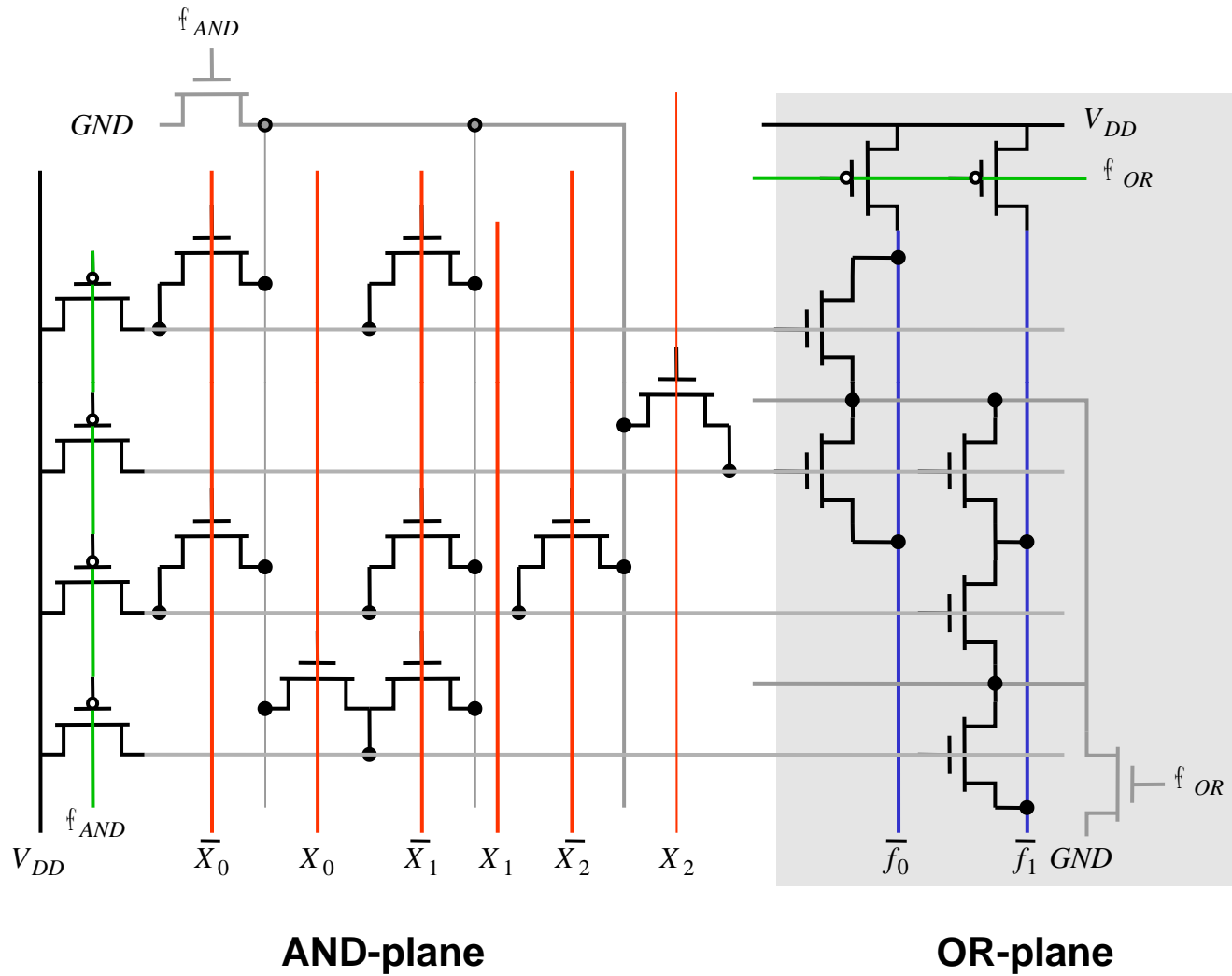
But ...

Programmable Logic Array

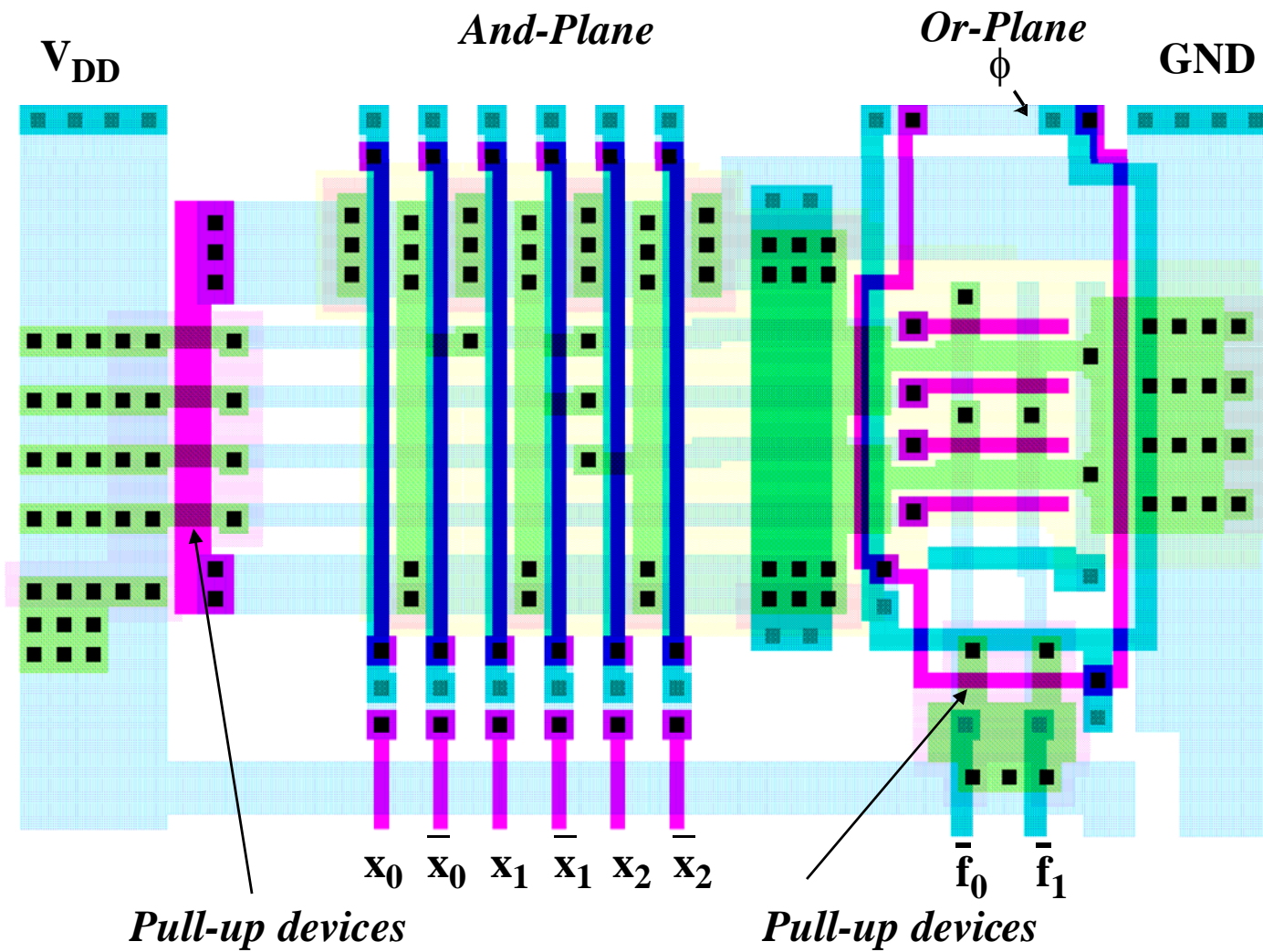
Pseudo-NMOS PLA



Dynamic PLA

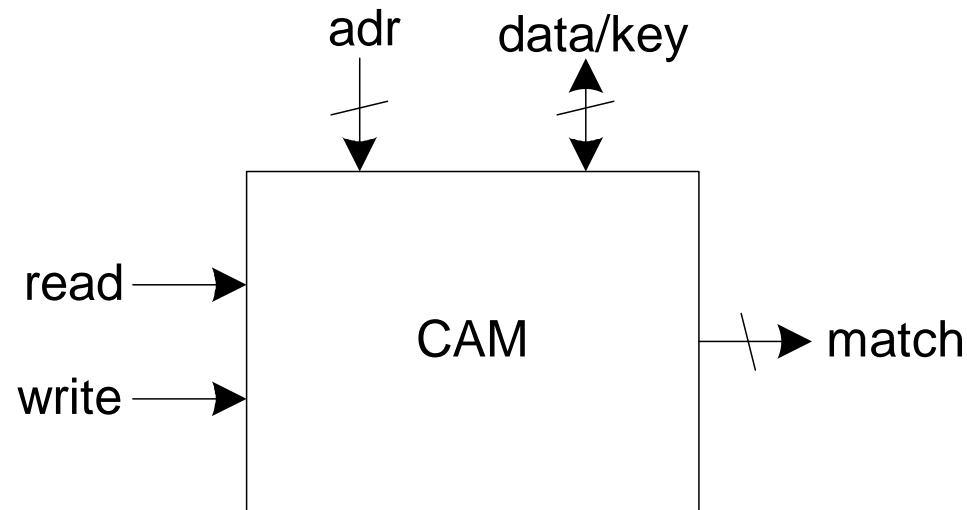


PLA Layout



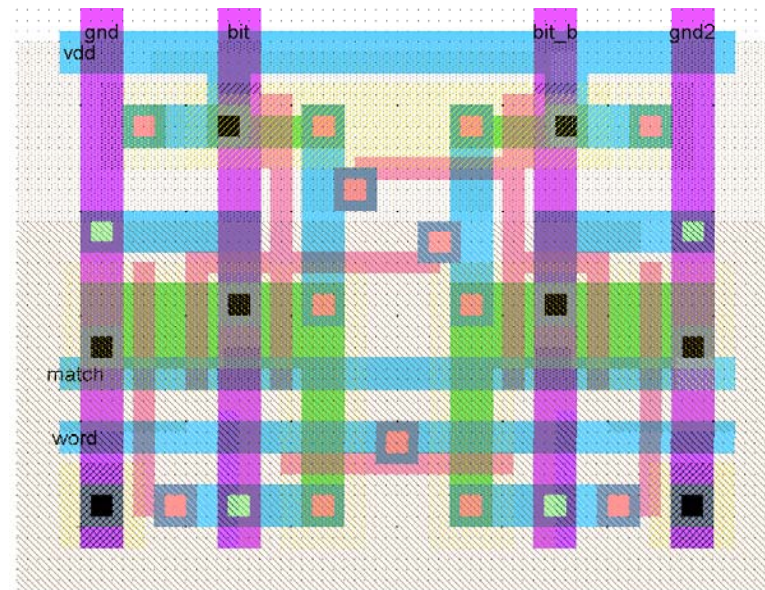
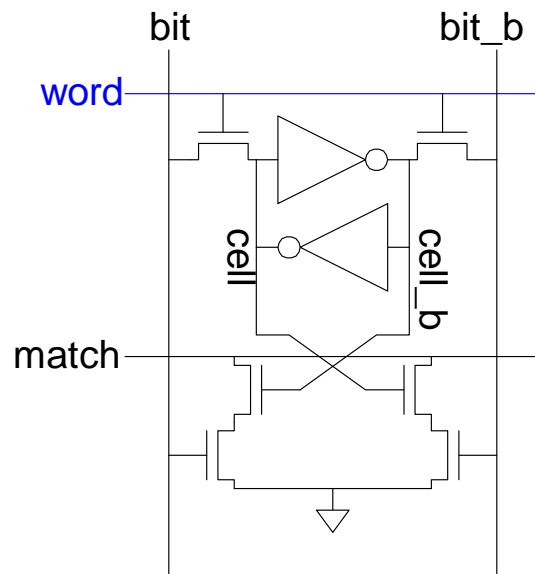
CAMs

- Extension of ordinary memory (e.g. SRAM)
 - Read and write memory as usual
 - Also *match* to see which words contain a *key*



10T CAM Cell

- Add four match transistors to 6T SRAM
 - 56 x 43 λ unit cell



CAM Cell Operation

- Read and write like ordinary SRAM

- For matching:

- Leave wordline low
- Precharge matchlines
- Place key on bitlines
- Matchlines evaluate

- Miss line

- Pseudo-nMOS NOR of match lines
- Goes high if no words match

