Electronics Systems (938II)

Lecture 3.2

Semiconductor Memories – ROM and PROM

Luca Crocetti

ROM

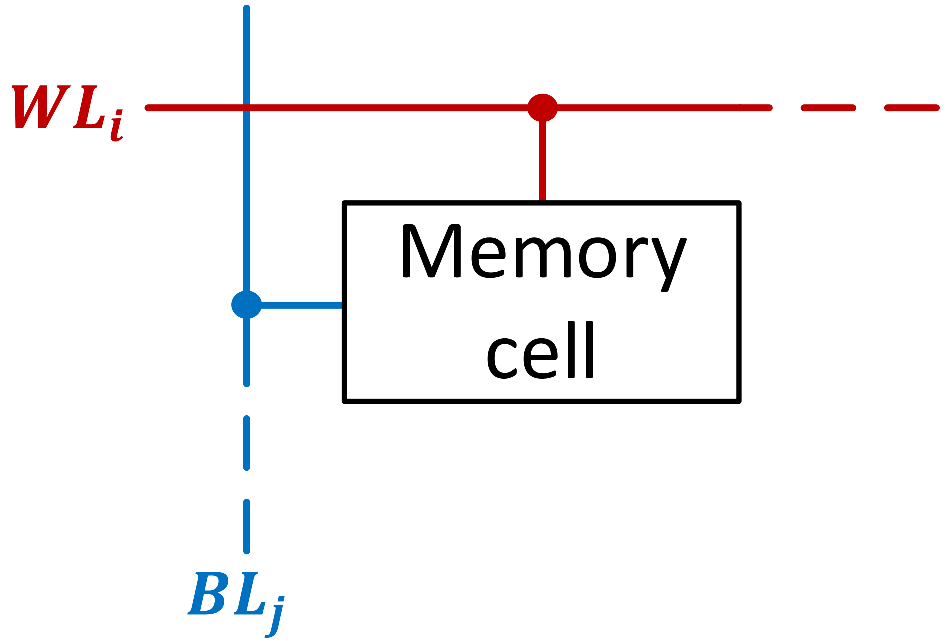
- **ROM** = **R**ead-**O**nly **M**emory

  - However, this gives information only on the supported operation(s)
    - I.e., read-only (not write)

  - It does not give information on the access mode (random or sequential)
    - Typically, this kind of memories supports random access
      - I.e., any arbitrary address can be accessed immediately
    - So, it is also a RAM
      - Please, do not confuse with the computer memory typically called RAM, which is a RAM (from the perspective of access mode), but it is also a RWM (Read-Write Memory)

ROM – Architecture

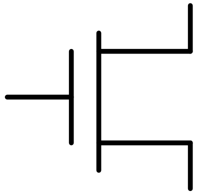
- Brief reminder of the structure of semiconductor memories
  - WL_i = Word Line
  - BL_j = Bit Line



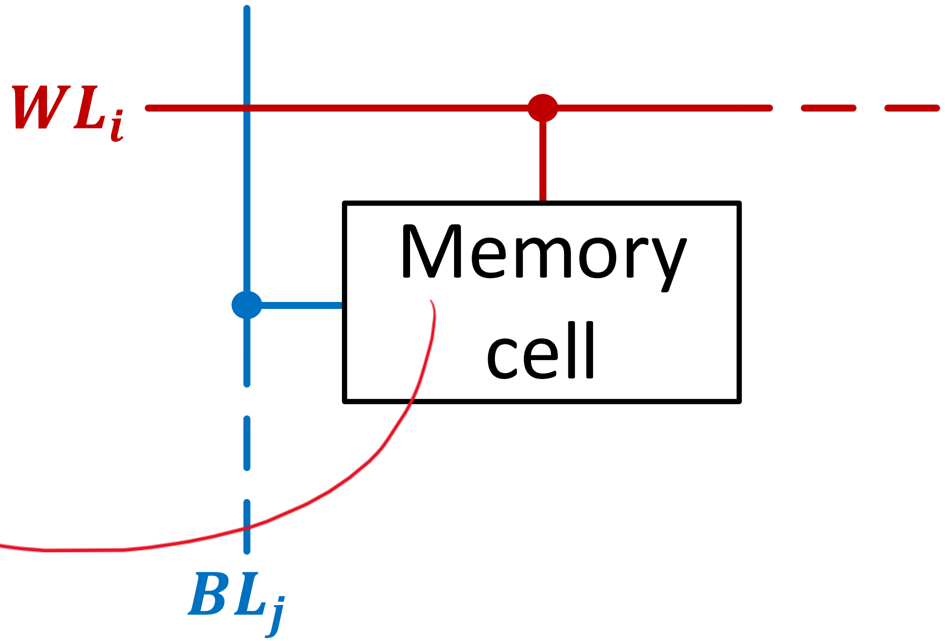
ROM – Architecture

- Brief reminder of the structure of semiconductor memories
  - $\boldsymbol{WL_i}$ = Word Line
  - $\boldsymbol{BL_j}$ = Bit Line

- In case of ROM, the memory cell can be
  - **1 n-MOS transistor**



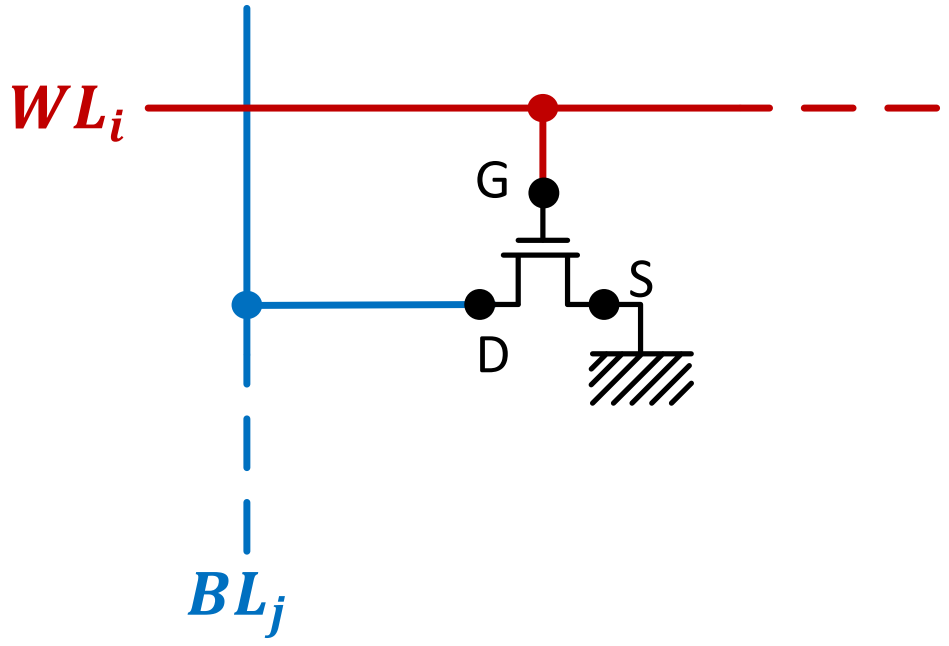
  - **Empty**



ROM – Architecture

- Memory cell – n-MOS transistor
  - $\boldsymbol{WL_i}$ = Word Line
  - $\boldsymbol{BL_j}$ = Bit Line

  - n-MOS
    - D = Drain
    - G = Gate
    - S = Source

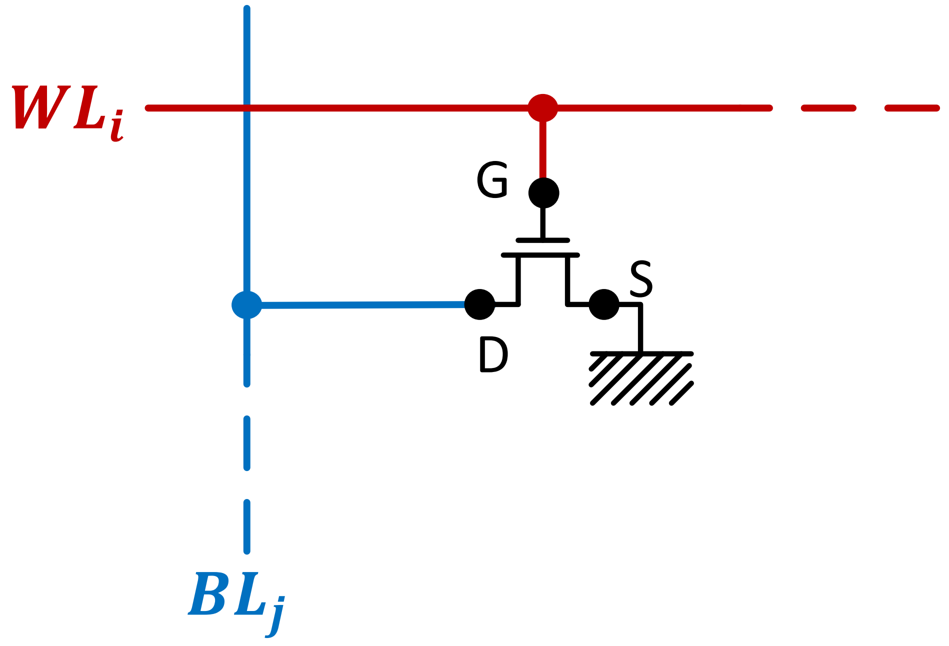


ROM – Architecture

- Memory cell – n-MOS transistor
  - WL_i = Word Line
  - BL_j = Bit Line

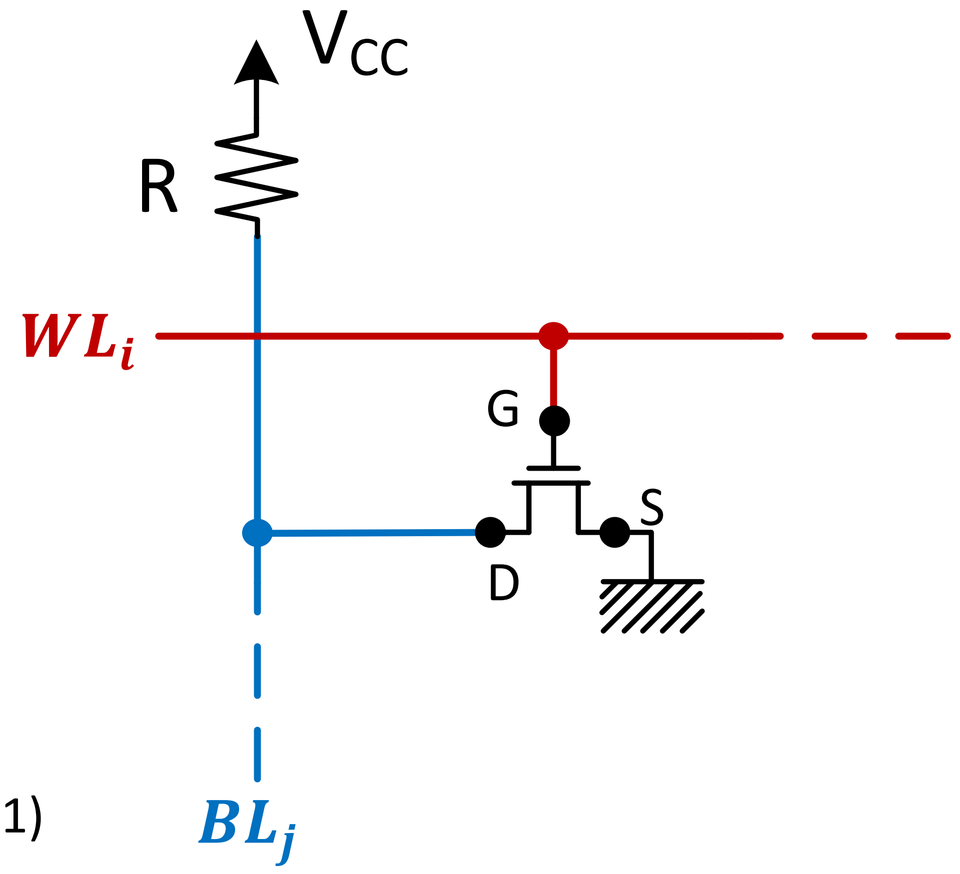
  - n-MOS
    - D = Drain
    - G = Gate
    - S = Source

  - To complete the architecture …



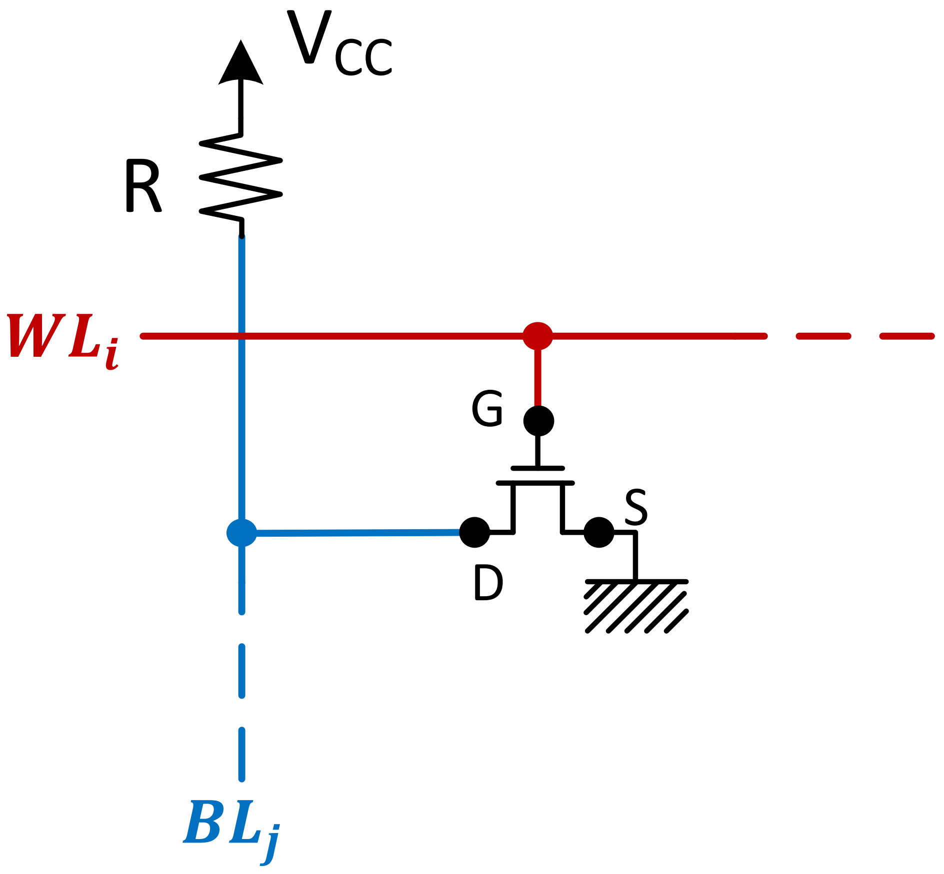
ROM – Architecture

- Memory cell – n-MOS transistor
  - $\boldsymbol{WL_i}$ = Word Line
  - $\boldsymbol{BL_j}$ = Bit Line
  - n-MOS
    - D = Drain
    - G = Gate
    - S = Source
  - R = Resistor
    - Pull-up resistor: connect BL_j to V_{CC} = high logic level (1)



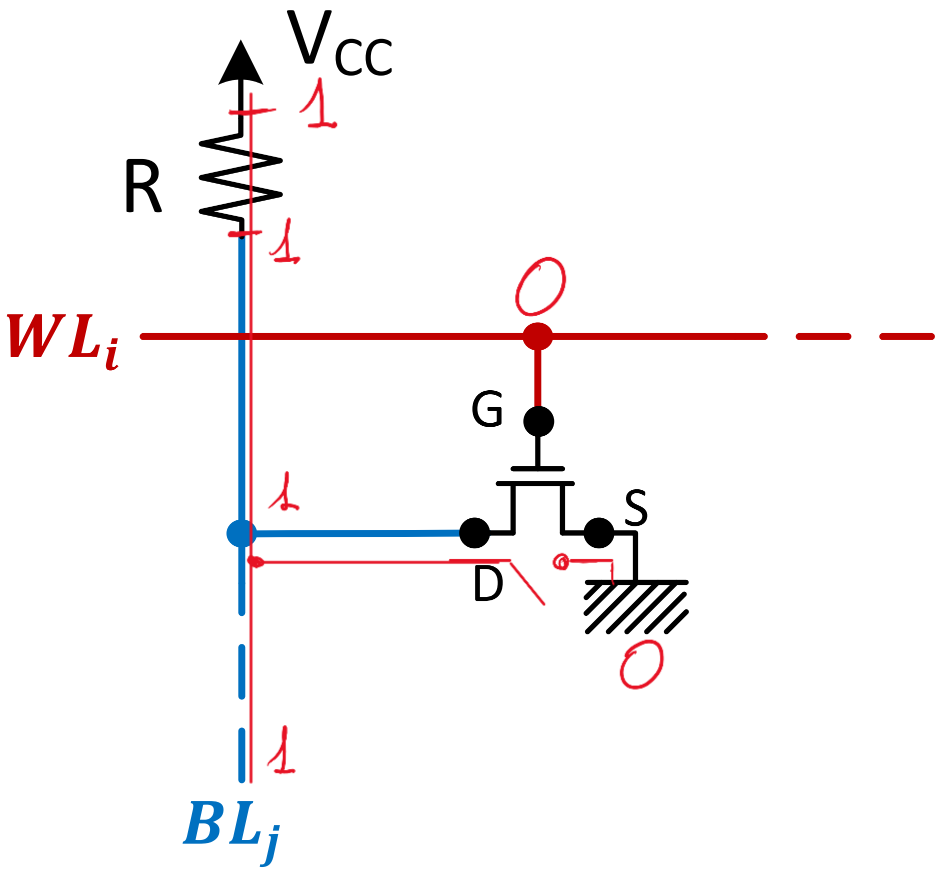
ROM – Architecture

- Memory cell – n-MOS transistor
  - Principle of operation



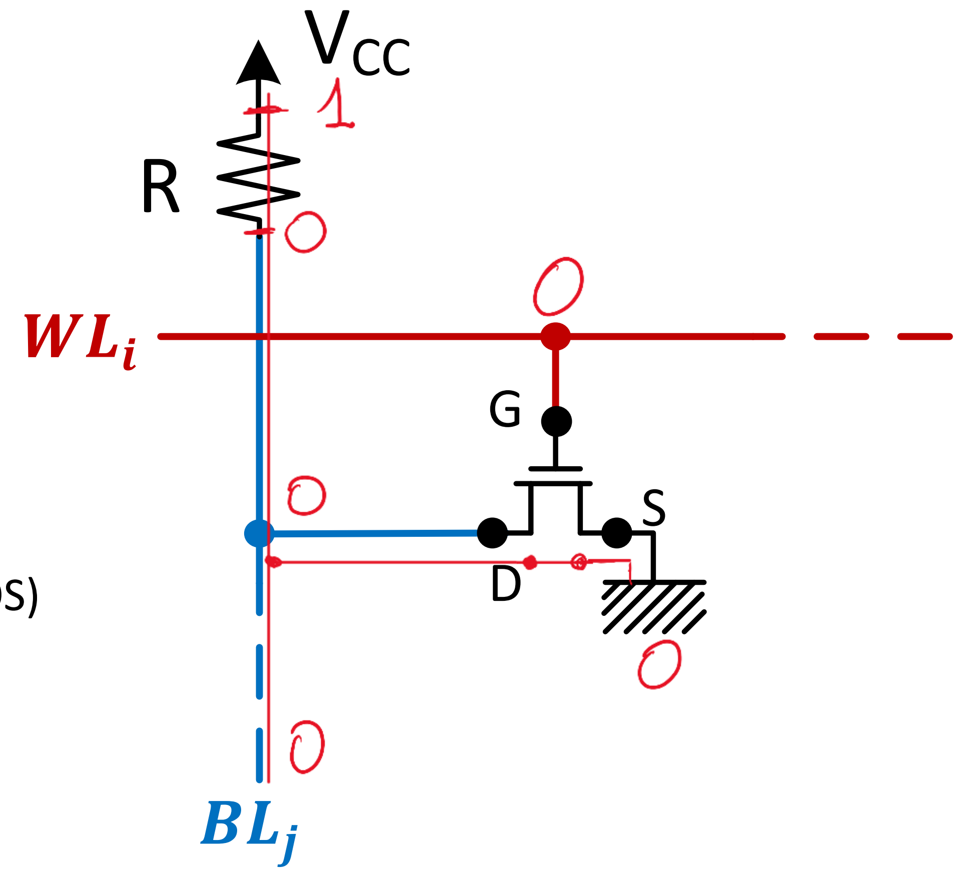
ROM – Architecture

- Memory cell – n-MOS transistor
  - Principle of operation
    - $\boldsymbol{WL_i}$ = 0
      - n-MOS = OFF (open circuit)
      - $\boldsymbol{BL_j}$ = 1 (because of pull-up resistor)



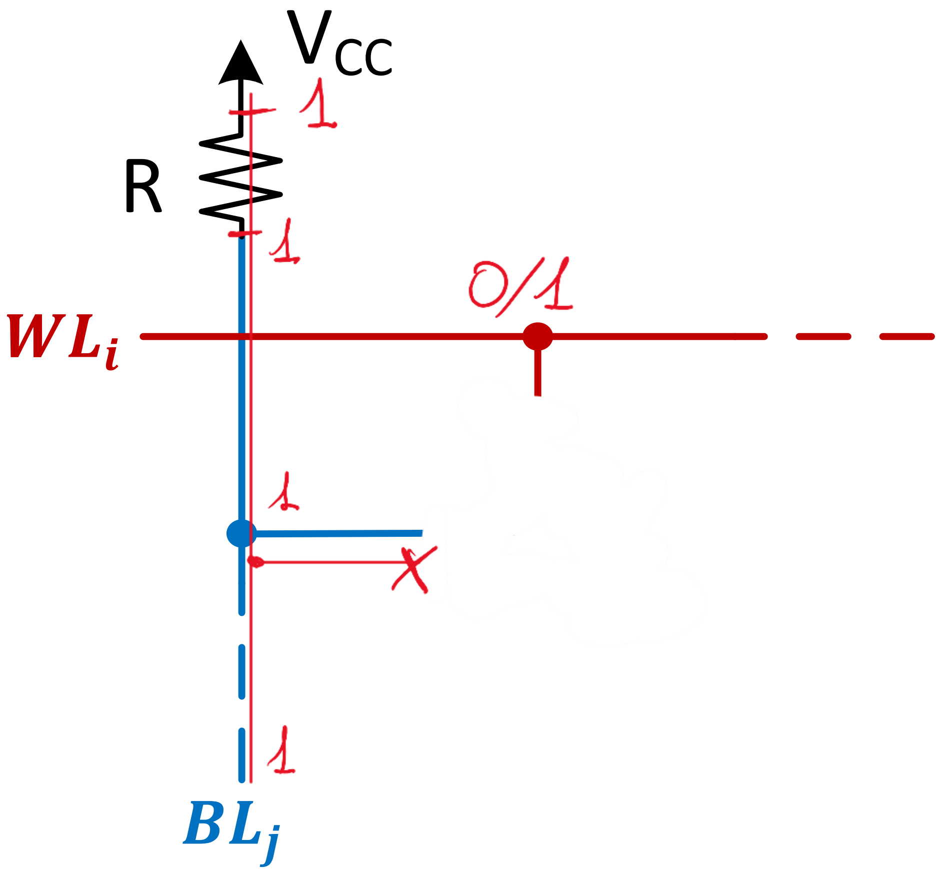
ROM – Architecture

- Memory cell – n-MOS transistor
  - Principle of operation
    - $\boldsymbol{WL_i}$ = 0
      - n-MOS = OFF (open circuit)
      - $\boldsymbol{BL_j}$ = 1 (because of pull-up resistor)
    - $\boldsymbol{WL_i}$ = 1
      - n-MOS = ON (short circuit)
      - $\boldsymbol{BL_j}$ = 0 (ground connection through the n-MOS)



ROM – Architecture

- Memory cell – Empty
  - Principle of operation
    - $\boldsymbol{WL_i}$ = 0
      - $\boldsymbol{BL_j}$ = 1 (because of pull-up resistor)
    - $\boldsymbol{WL_i}$ = 1
      - $\boldsymbol{BL_j}$ = 1 (because of pull-up resistor)



ROM – Architecture

- Memory cell – Summary
  - Principle of operation

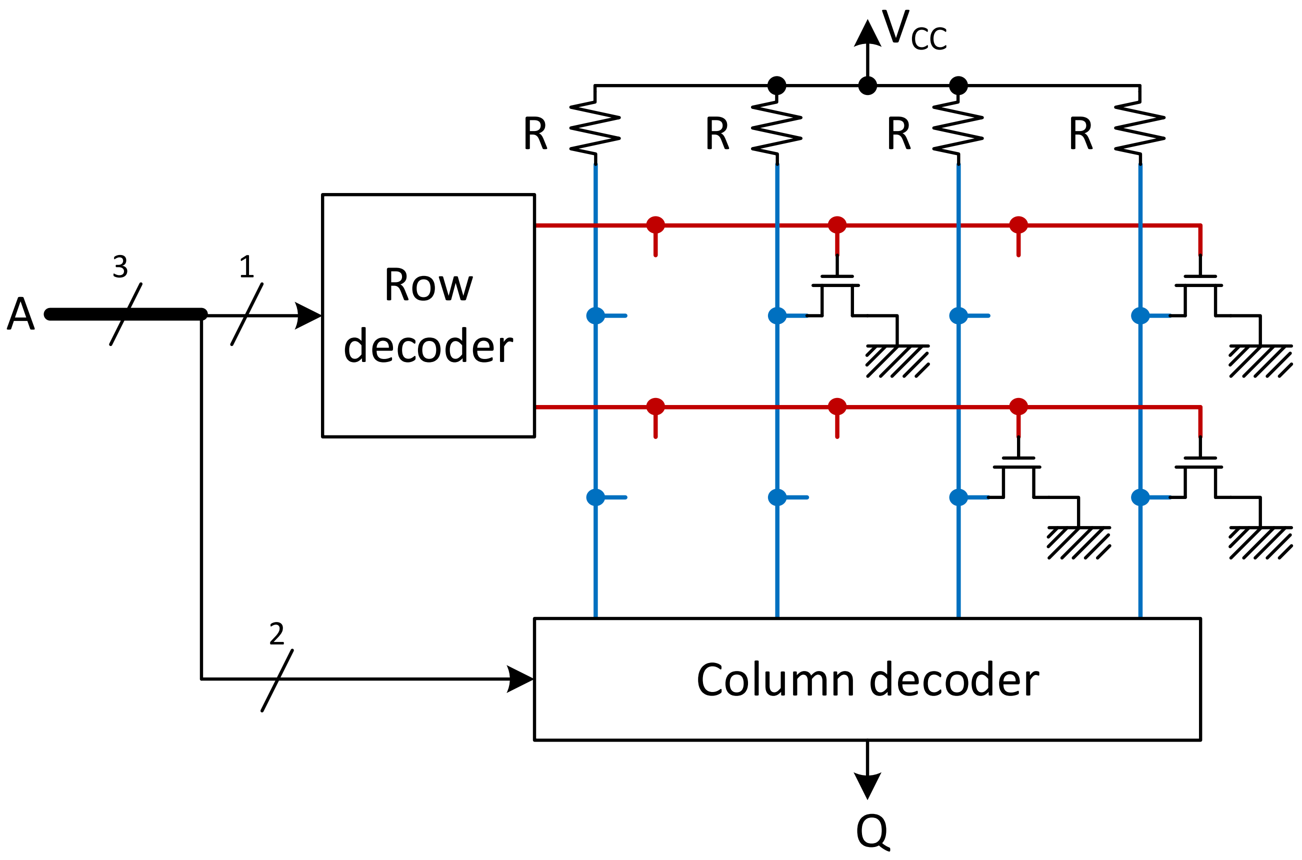
Memory cell	Word Line (WL_i)	Bit Line (BL_j)
With n-MOS	0 (disabled)	1
	1 (enabled)	0
Empty	0 (disabled)	1
	1 (enabled)	1

ROM – Architecture

- Integrating the row and column decoders to complete the architecture

ROM – Architecture

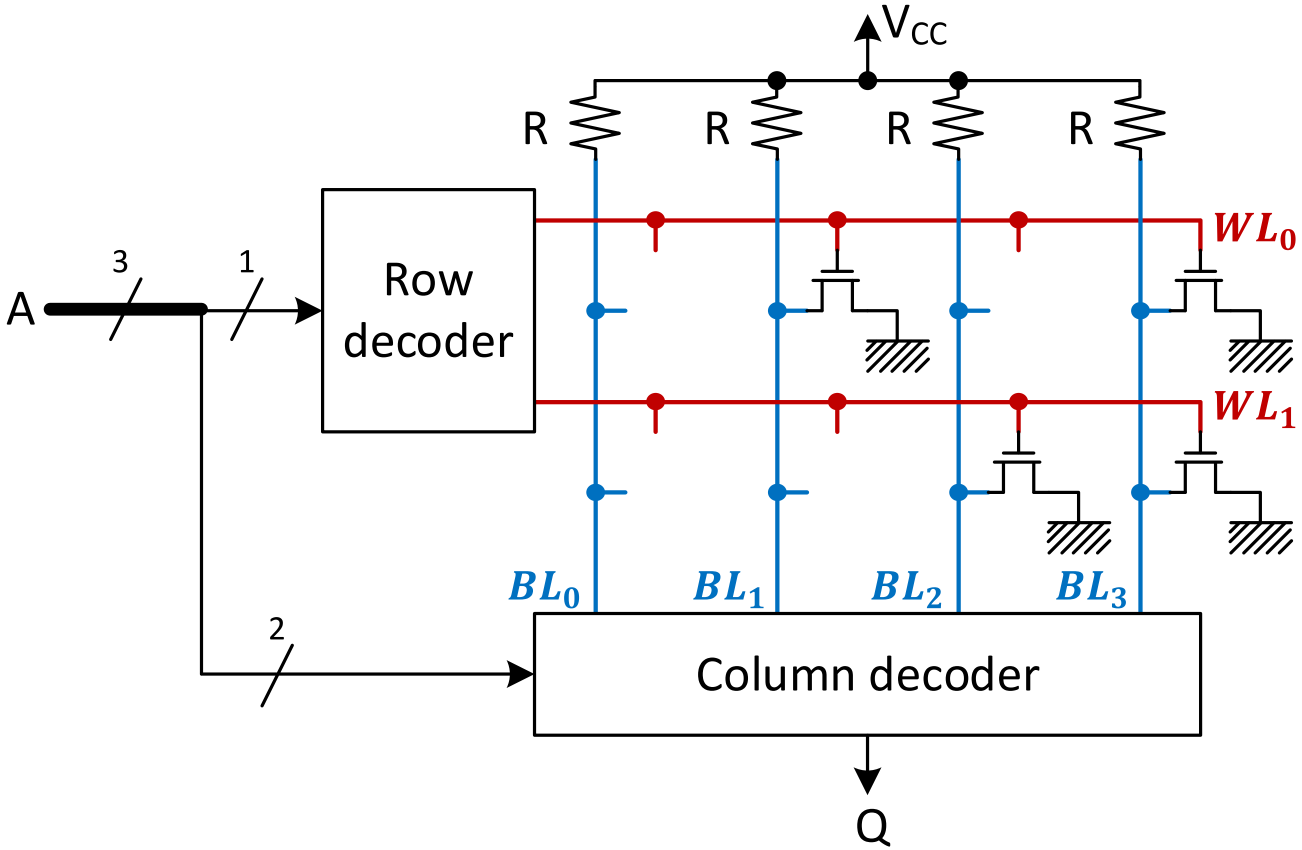
- Integrating the row and column decoders to complete the architecture
  - Example on a 2x4 ROM
    - A = Address (3-bit)
    - Q = single-bit output



ROM – Architecture

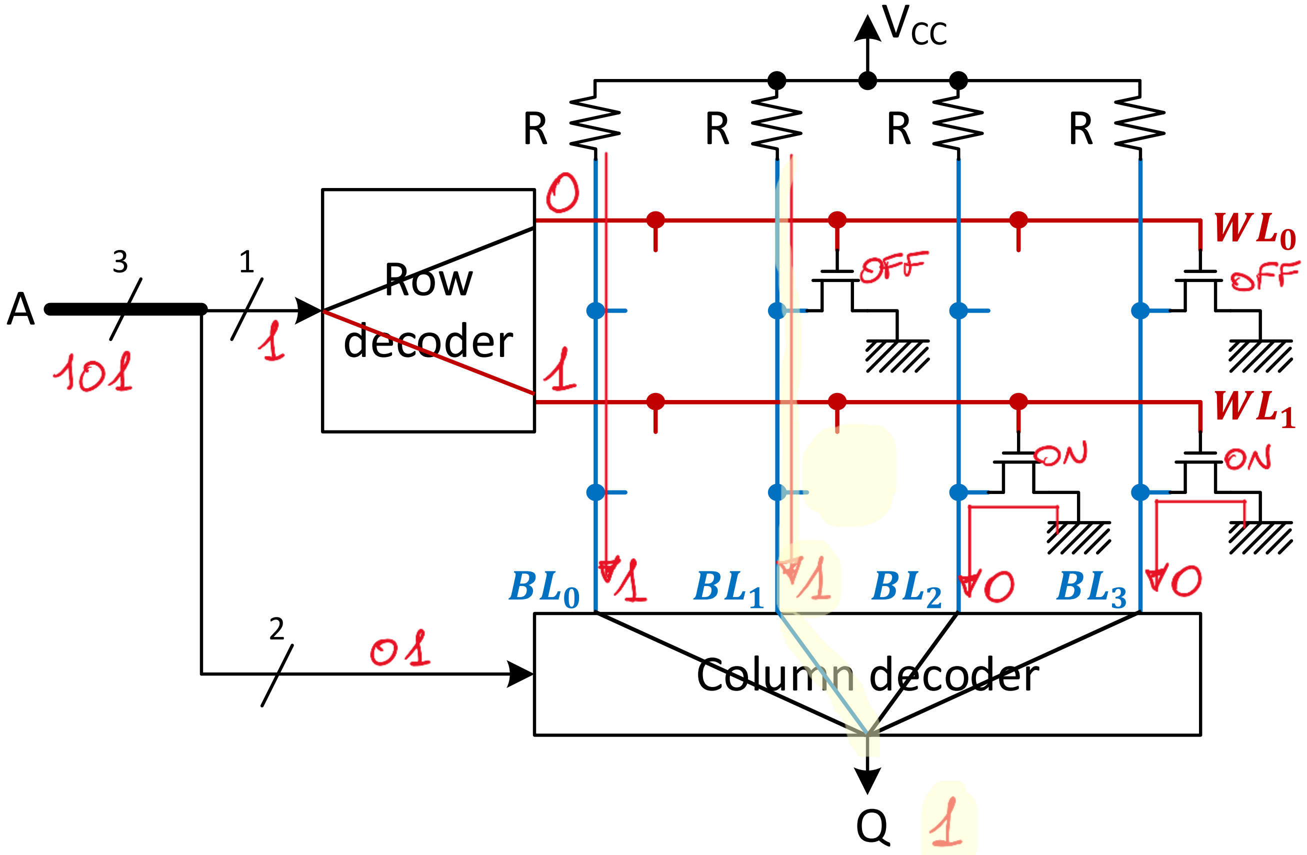
- Integrating the row and column decoders to complete the architecture
  - Example on a 2x4 ROM
    - A = Address (3-bit)
    - Q = single-bit output

    - Assume **cell (x,y)**
      - $\boldsymbol{WL_X}$
      - $\boldsymbol{BL_y}$



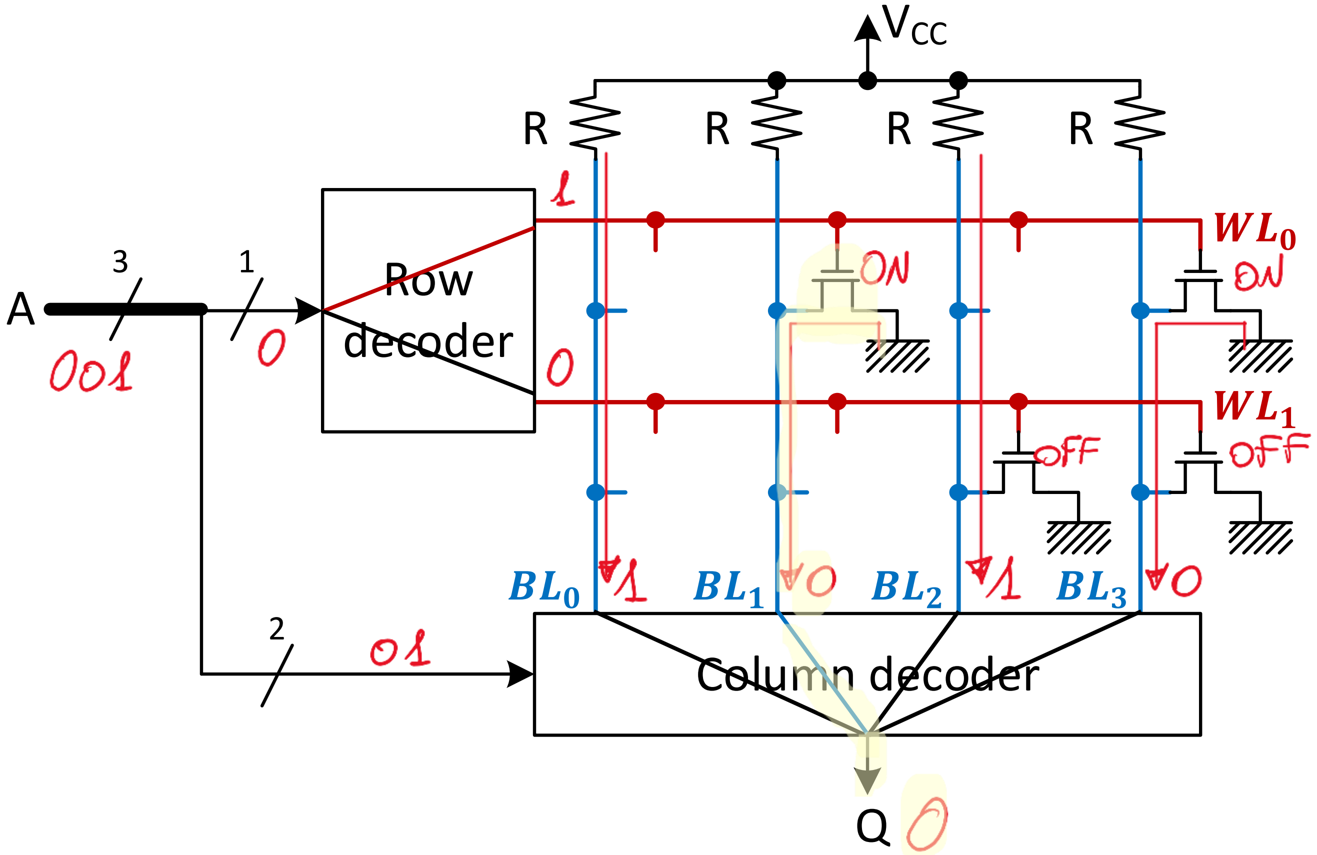
ROM – Architecture

- Integrating the row and column decoders to complete the architecture
  - Example on a 2x4 ROM
    - A = Address (3-bit)
    - Q = single-bit output
    - Assume **cell (x,y)**
      - WL_X
      - BL_y
    - Assume to read **cell (1,1)**



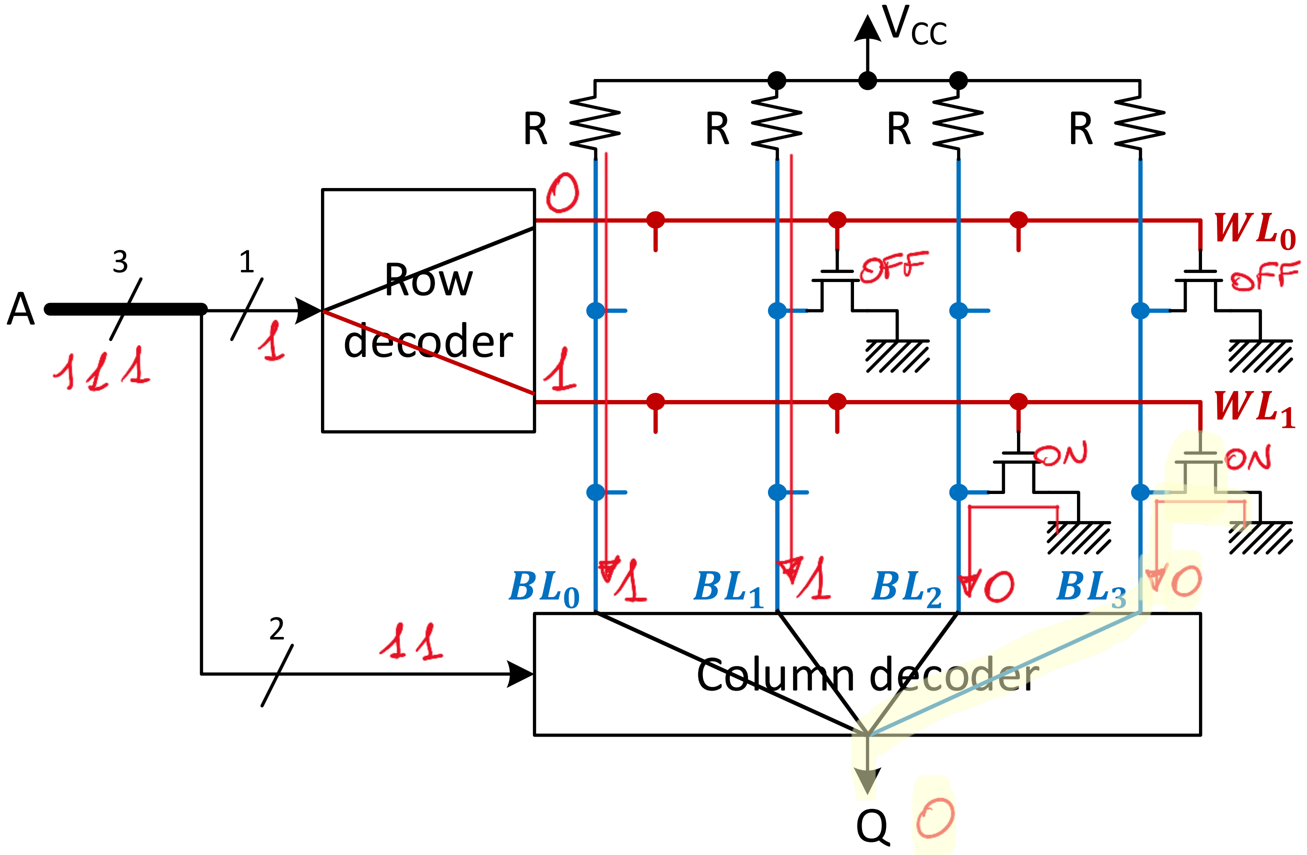
ROM – Architecture

- Integrating the row and column decoders to complete the architecture
  - Example on a 2x4 ROM
    - A = Address (3-bit)
    - Q = single-bit output
    - Assume **cell (x,y)**
      - WL_X
      - BL_y
    - Assume to read **cell (0,1)**



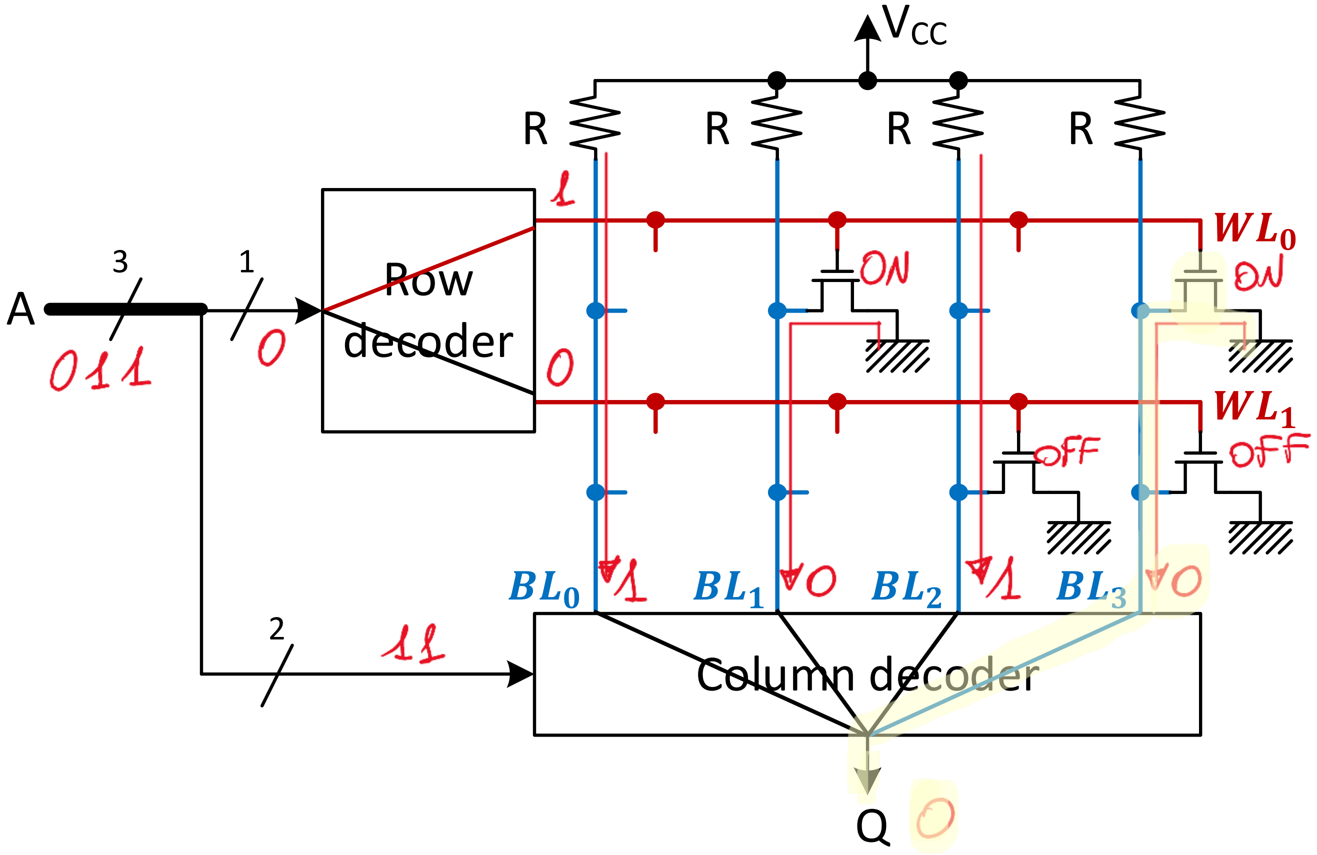
ROM – Architecture

- Integrating the row and column decoders to complete the architecture
  - Example on a 2x4 ROM
    - A = Address (3-bit)
    - Q = single-bit output
    - Assume **cell (x,y)**
      - WL_X
      - BL_y
    - Assume to read **cell (1,3)**



ROM – Architecture

- Integrating the row and column decoders to complete the architecture
  - Example on a 2x4 ROM
    - A = Address (3-bit)
    - Q = single-bit output
    - Assume **cell (x,y)**
      - WL_X
      - BL_y
    - Assume to read **cell (0,3)**

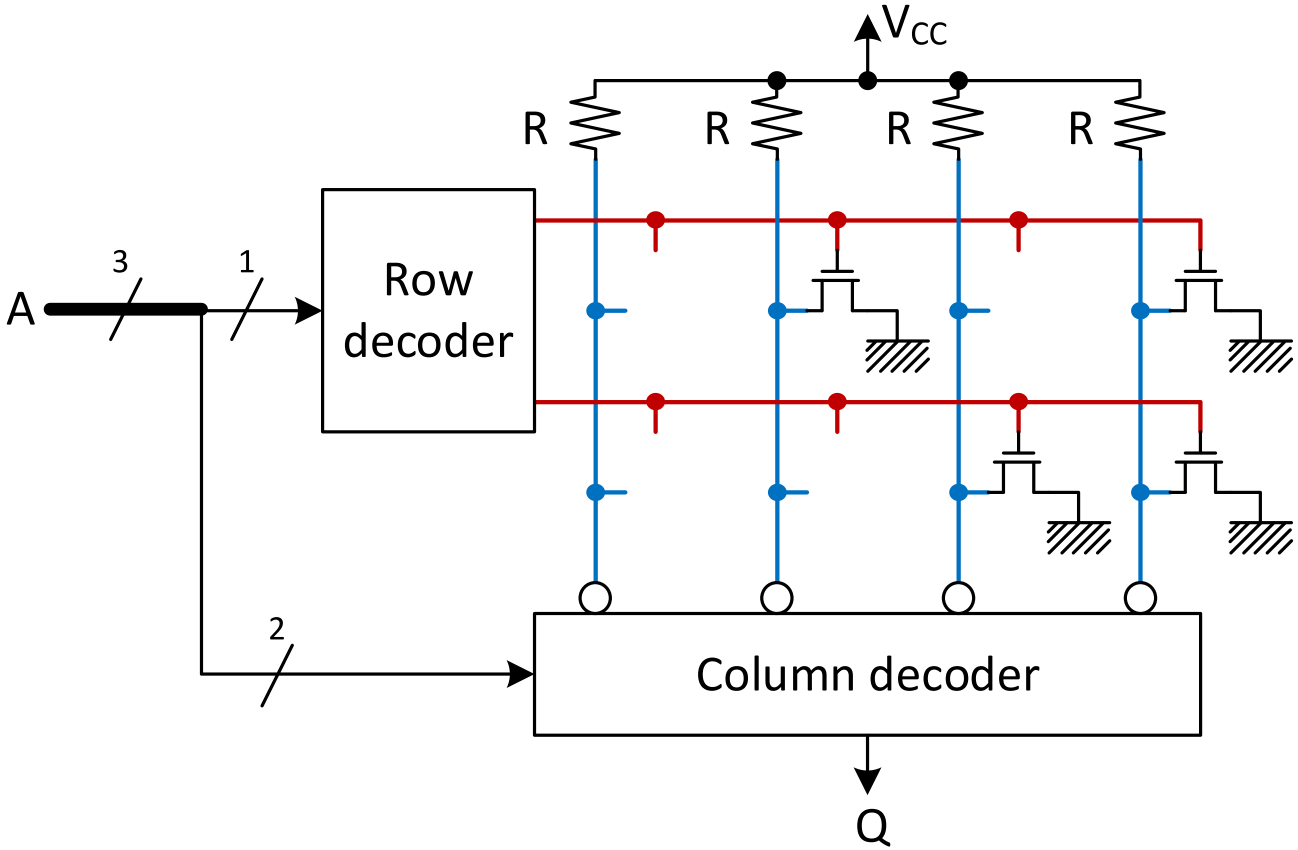


ROM – Architecture

- So, when reading (i.e. enabling corresponding word line and bit line) a memory cell
  - With (n-MOS) transistor → Output = 0
  - Empty (without transistor) → Output = 1
- However, also solutions with active-low decoding logic (for column decoder) exist

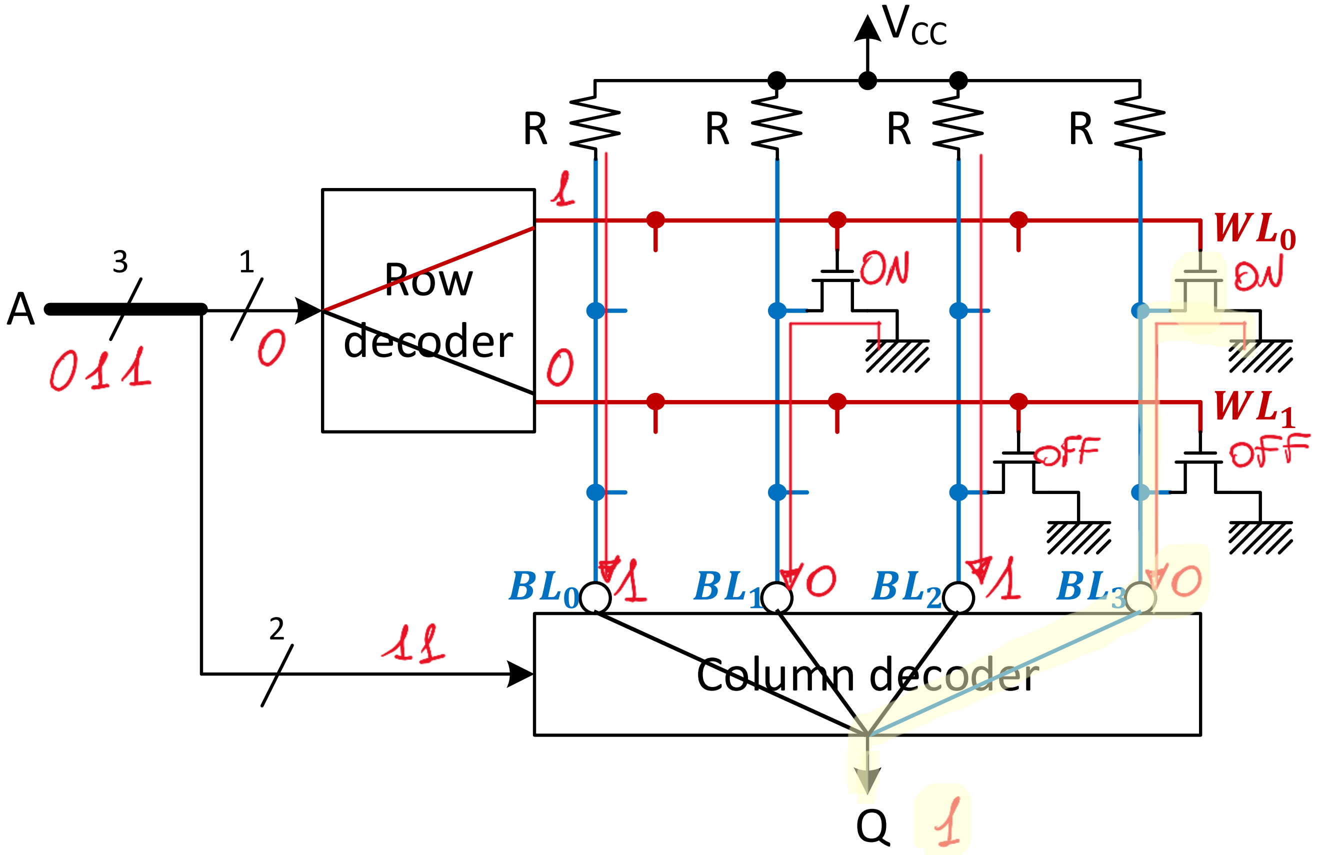
ROM – Architecture

- Active-low decoding logic (column decoder)
  - Example on a 2x4 ROM



ROM – Architecture

- Active-low decoding logic (column decoder)
  - Example on a 2x4 ROM
    - Assume to read **cell (0,3)**



ROM – Architecture

- When reading a memory cell when column decoder is active-low
  - With (n-MOS) transistor → Output = 1
  - Empty (without transistor) → Output = 0

  - The opposite of the previous case

ROM – Architecture

- Content of memory cells
  - The output (bit) corresponds to the content of memory cells, since that is what is obtained by reading that cell

Memory cell	Bit Line ($\mathrm{B}L_j$)	Output (bit) = Content	
		Active-high decoding logic	Active-low decoding logic
With n-MOS	0	0	1
Empty	1	1	0

ROM – Remarks and applications

- The one shown is called also **mask-ROM**
  - Remember about the photolithographic masks employed in the semiconductor manufacturing process
  - Each memory content corresponds to a different layout (of transistors), or, in other words, to a different integrated circuit
  - Thus, to a different mask to realize that layout (or circuit)

ROM – Remarks and applications

- Photolithographic masks have high costs
  - Only for applications that require high volumes of ROM

- Memory content cannot be modified
  - The modification of just one bit (due to updates or bugs) means wasting the entire chip (and all other chips with the same content)
  - Only for solid applications for which you are largely sure about the content of the memory

PROM

- In the course of technological progress, mask-ROM has evolved into **Programmable ROM (PROM)**
  - Still a ROM, Read-Only Memory
  - But whose content is defined after the manufacture (of the integrated circuit)
    - Not during the manufacture like the mask-ROM
  - This allowed for a reduction in production costs because …
    - For mask-ROMs: different memory content → different integrated circuit → different production line (masks, …)
    - For PROM: always the same integrated circuit → the same production line

PROM

- So, after production the PROM must be “written” to define its content
  - This operation is called Programming
    - From here the name
  - Just one time
    - OTP = One Time Programmable

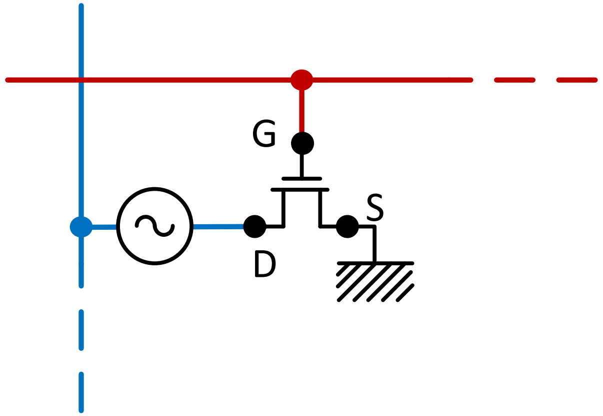
PROM

- So, after production the PROM must be "written" to define its content
  - This operation is called Programming
    - From here the name
  - Just one time
    - OTP = One Time Programmable

- For this purpose, each memory cell contains **a transistor + a fuse**
  - The **fuse** is an electronic component that …
    - … by default, is a connection (a short circuit)
    - … when subjected to a very strong current, it breaks down, breaking the connection (it becomes an open circuit)

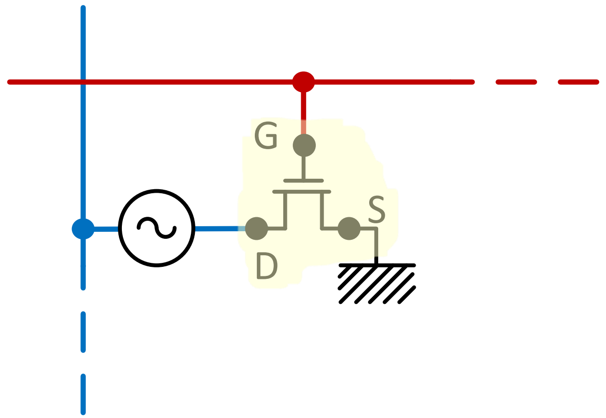
PROM – Architecture

- Memory cell



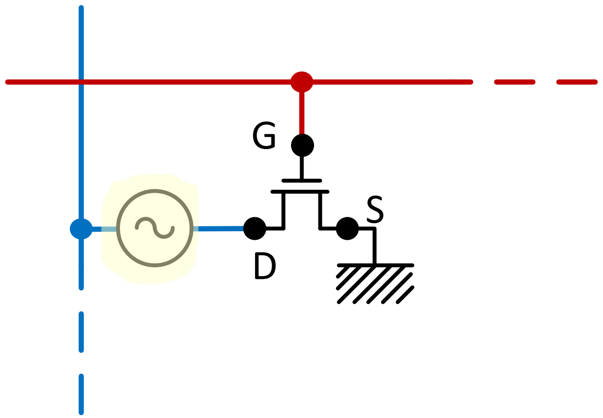
PROM – Architecture

- Memory cell
  - Transistor (n-MOS)



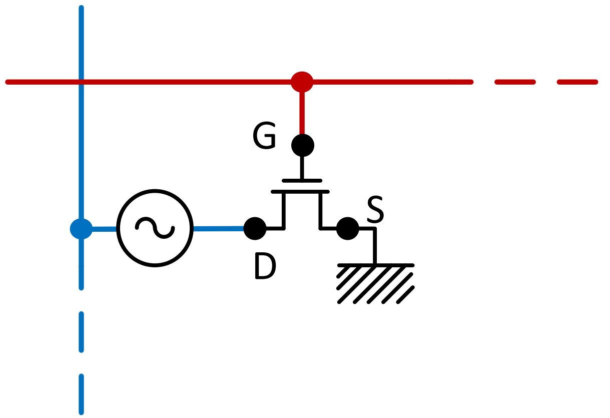
PROM – Architecture

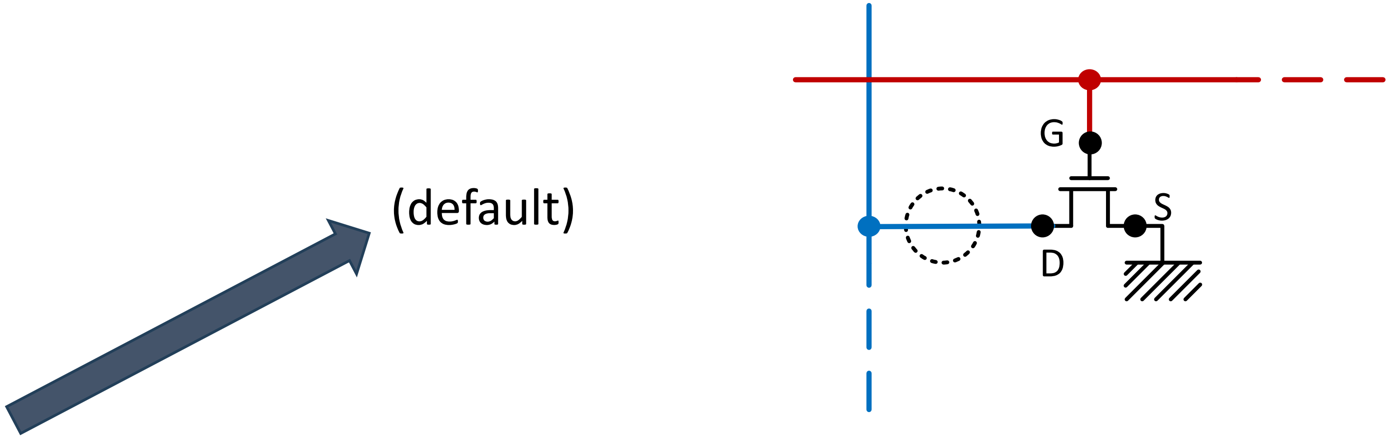
- Memory cell
  - Transistor (n-MOS)
  - Fuse



PROM – Architecture

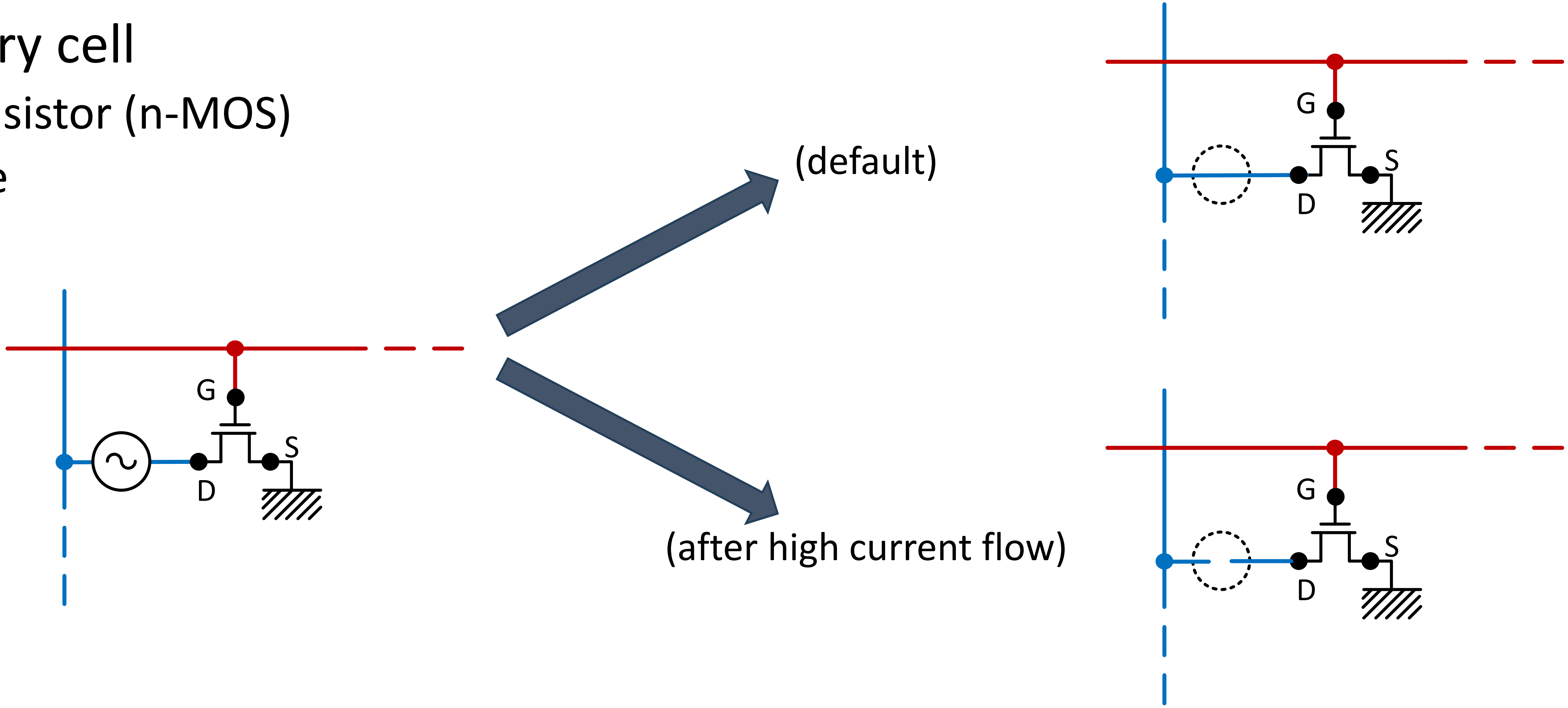
- Memory cell
  - Transistor (n-MOS)
  - Fuse





PROM – Architecture

- Memory cell
  - Transistor (n-MOS)
  - Fuse

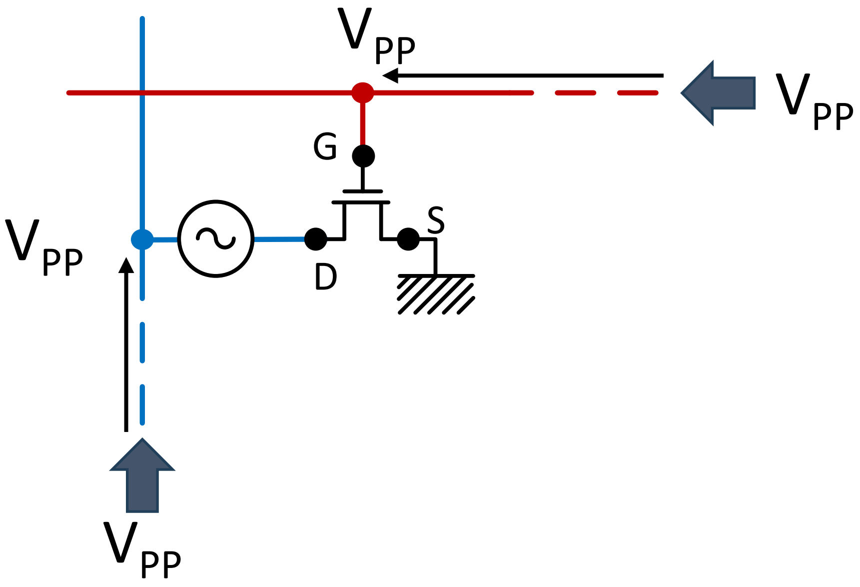


PROM – Principle of operation

- Thus, programming means selecting which fuses to break (by injecting a high current)
  - Programmed fuse = broken fuse = open circuit
  - Unprogrammed fuse = default-state fuse = short circuit

PROM – Principle of operation

- For this purpose, high voltage pulses (V_{PP} = 10/30 V) must be applied to
  - Gate terminal of n-MOS = Word Line
  - Drain terminal of n-MOS = Bit Line
  - n-MOS is in "high conduction" region = very high current



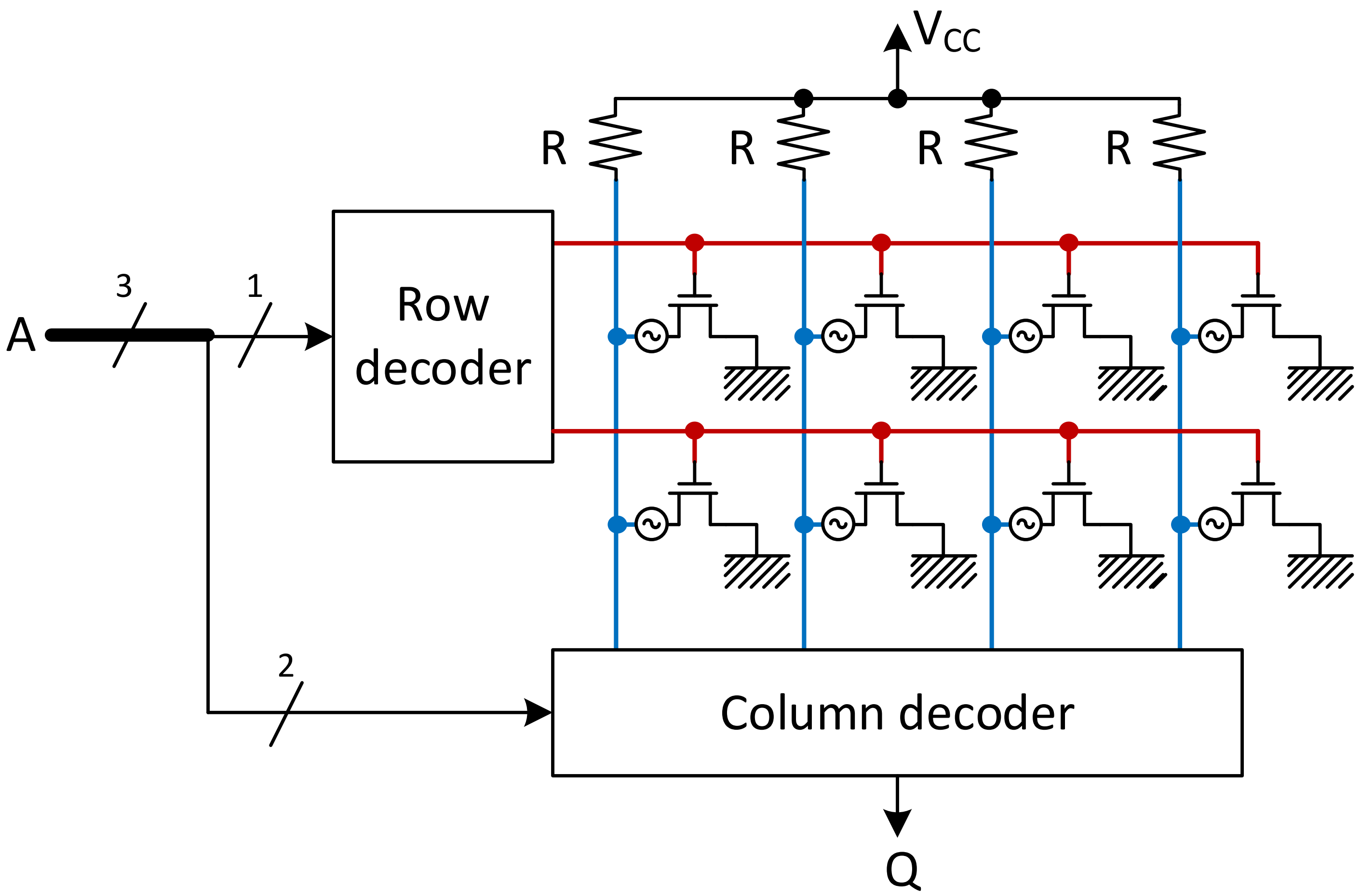
PROM – Principle of operation

- By selecting (only) the corresponding Word Line and Bit Line, each memory cell can be programmed (or not) without the risk of accidentally programming cells that should do not
  - A dedicated interface is required

- After programming, the PROM works exactly like the mask-ROM
  - Not programmed cell = cell with transistor → 0 on BL → 0/1 on output (Q)
  - Programmed cell ≅ empty cell → 1 on BL → 1/0 on output (Q)

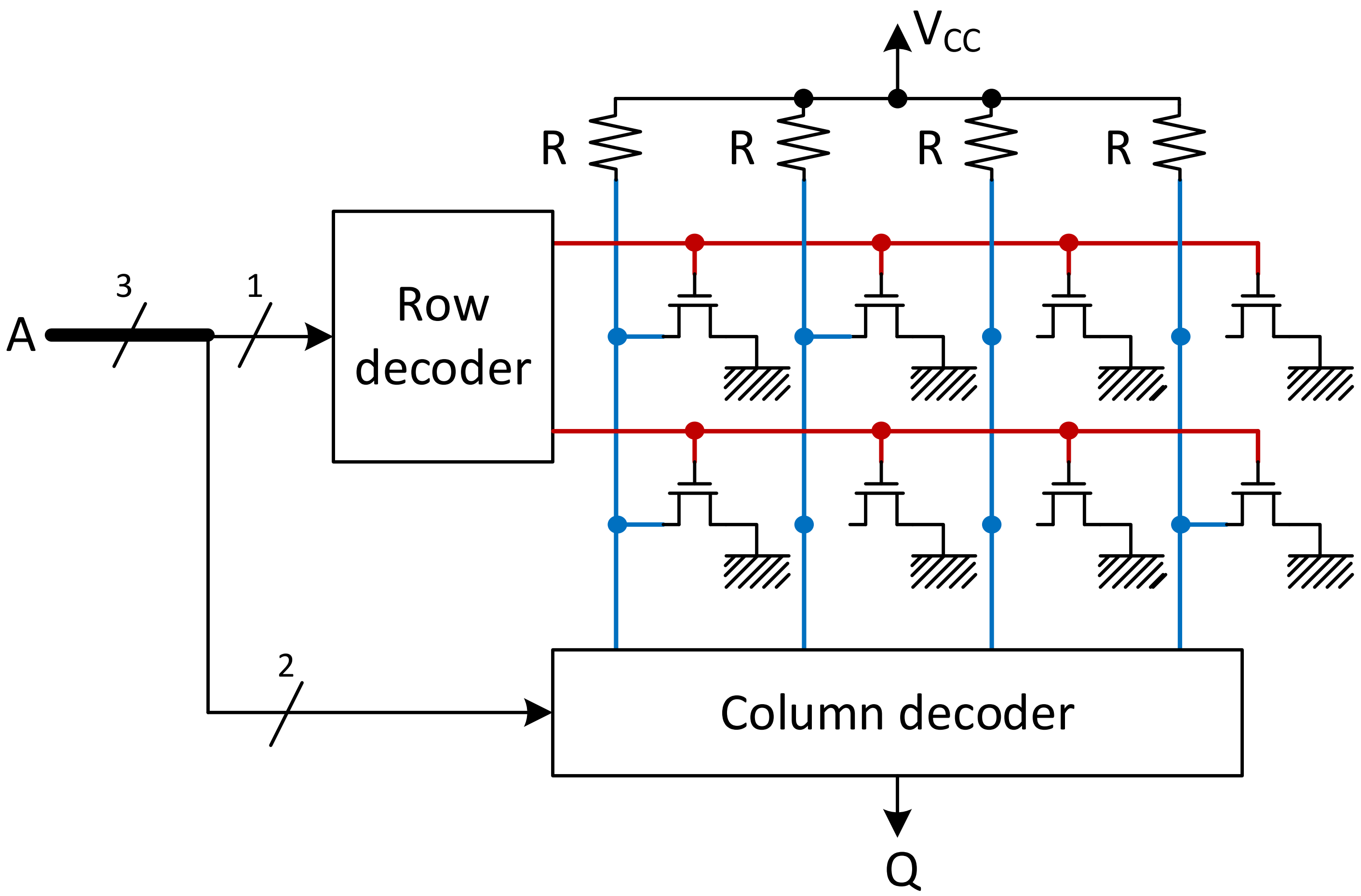
PROM – Principle of operation

- Example
  - Before programming



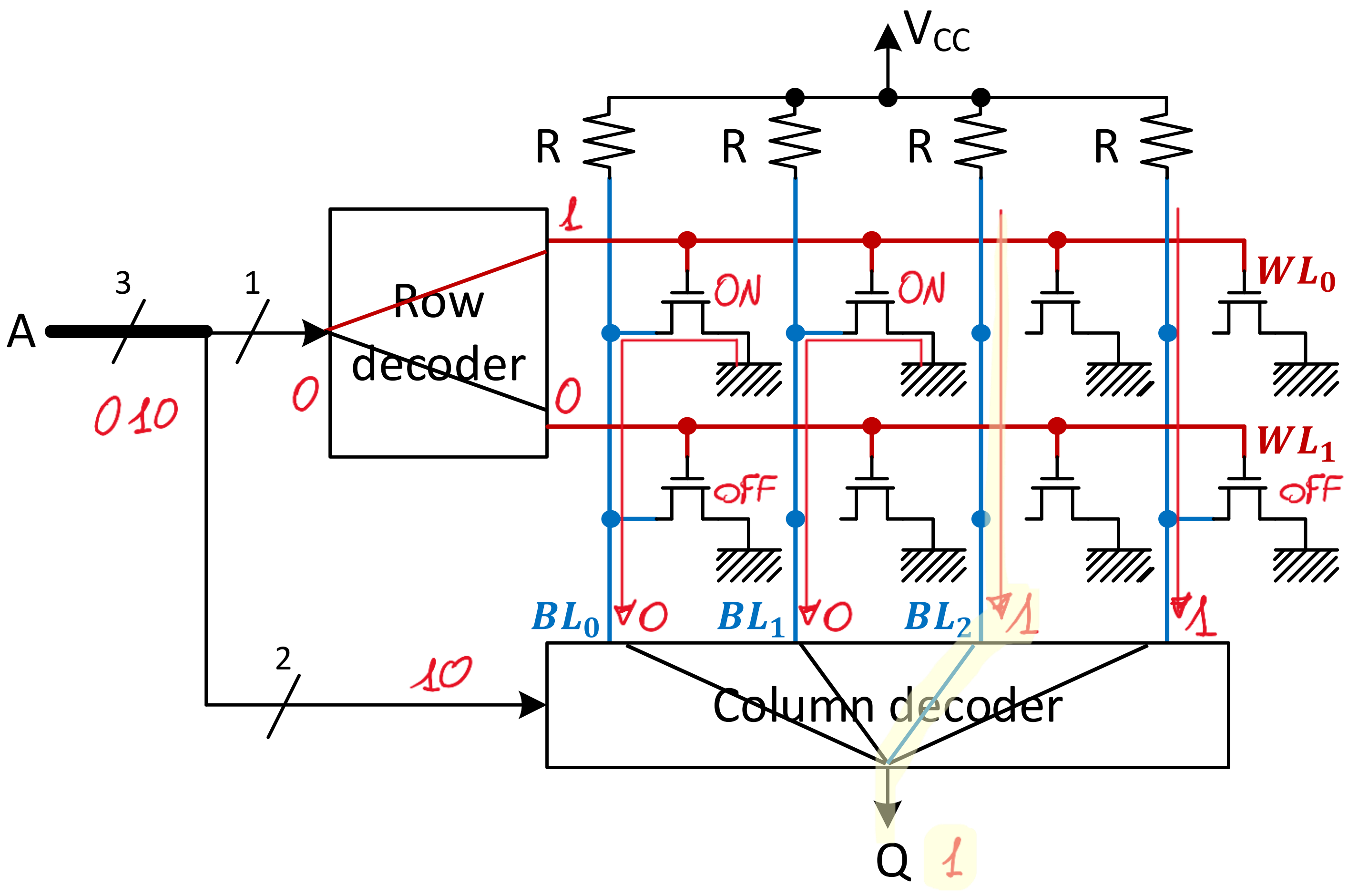
PROM – Principle of operation

- Example
  - After programming



PROM – Principle of operation

- Example
  - Reading **cell (0,2)**



PROM – Remarks and applications

- **Anti-fuses** can be used instead of fuses
  - Opposite of fuse
    - Default (not programmed) = open circuit
    - Programmed (with high current flow) = short circuit

- Early PROMs had reliability problems
  - Incompletely vaporized fuses
  - Floating shrapnel inside the IC package

- Target applications
  - The same as mask-ROMs

Thank you for your attention

Luca Crocetti
(luca.crocetti@unipi.it)