Electronics Systems (938II)

Lecture 3.4

Semiconductor Memories – Principles of Flash memory

Flash memory

- It is the latest evolution of ROM
  - It is an evolution of EEPROM (by a technological point of view)
    - Same principle
      - MOS with a floating gate
      - Fully electrical: programming and erasing
    - But with some advantages
    - We will see it in detail later

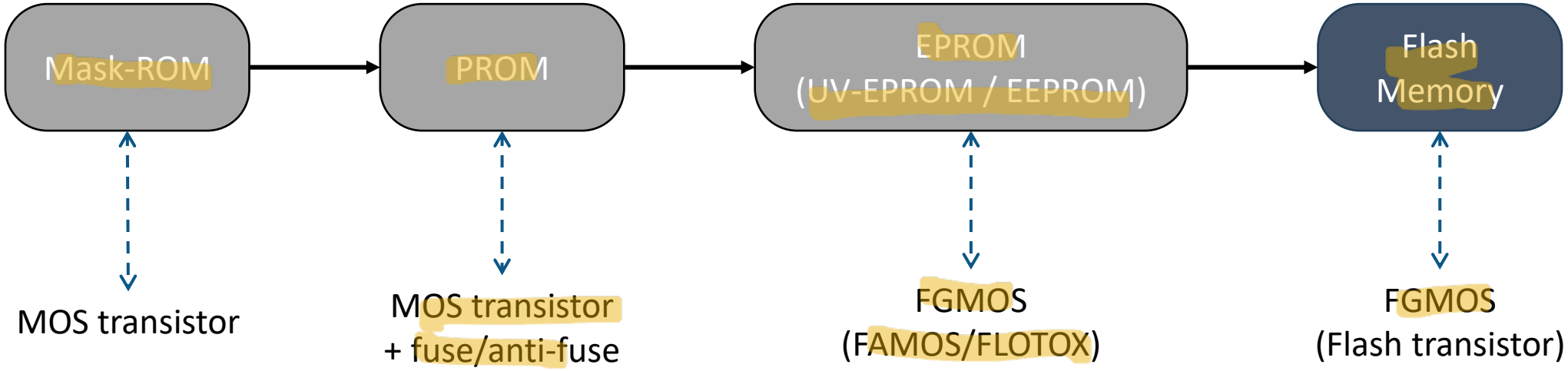
Flash memory

- It is the latest evolution of ROM
  - Main application = non-volatile (massive) storage
    - Like a ROM
    - Yes, it can be "written" and "rewritten" (erasing + programming), also several times
    - But it cannot properly be considered an RWM
      - Because "writing" process is disruptive: it damages the device
        - Ideally, a RWM should be able to be written an infinite number of times, instead …
        - Limited endurance (although very high)

↑ Cannot be seen as a RW memory, because cycles degrade oxide of transistors

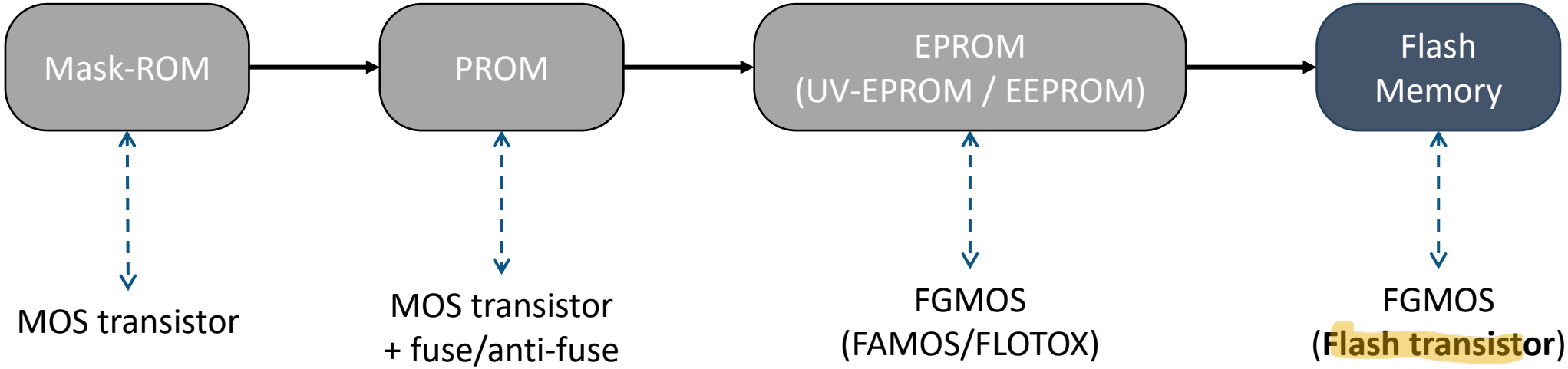
Flash memory

- It is the latest evolution of ROM



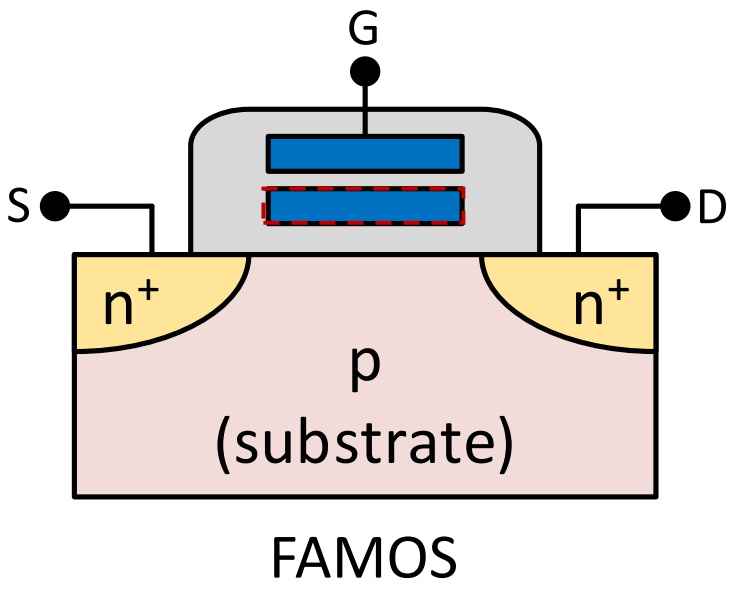
Flash memory

- It is the latest evolution of ROM



Flash transistor

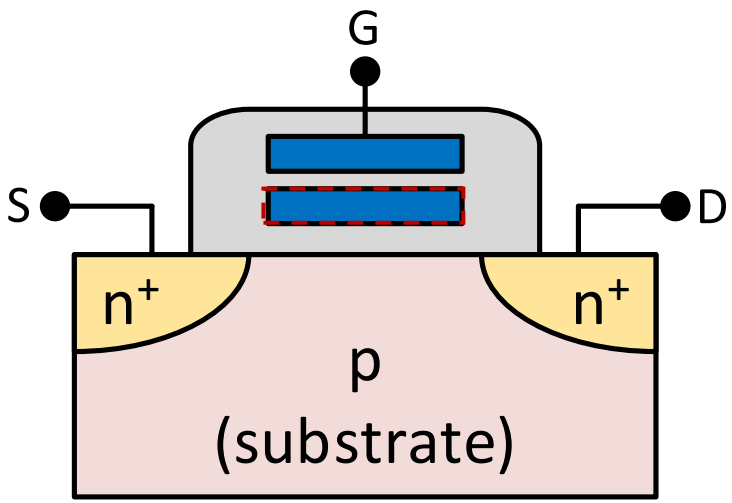
- Another FGMOS
  - Merging characteristics of FAMOS and FLOTOX
    1. Symmetric floating gate



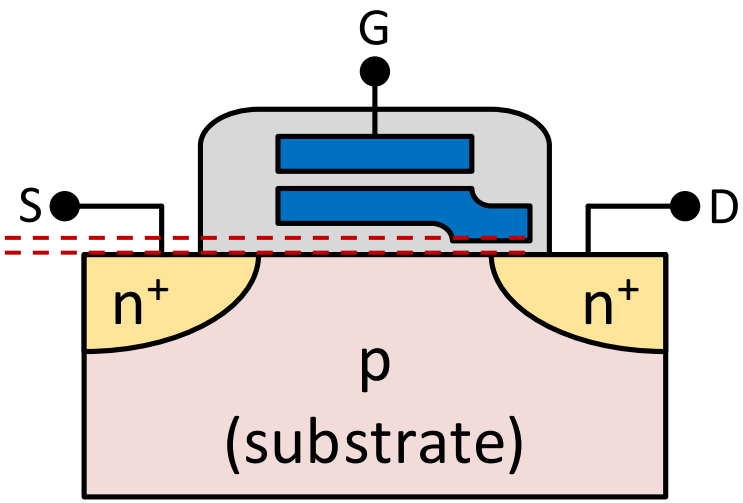
FAMOS

Flash transistor

- Another FGMOS
  - Merging characteristics of FAMOS and FLOTOX
    1. Symmetric floating gate
    2. Separated by a very thin oxide layer (from substrate)



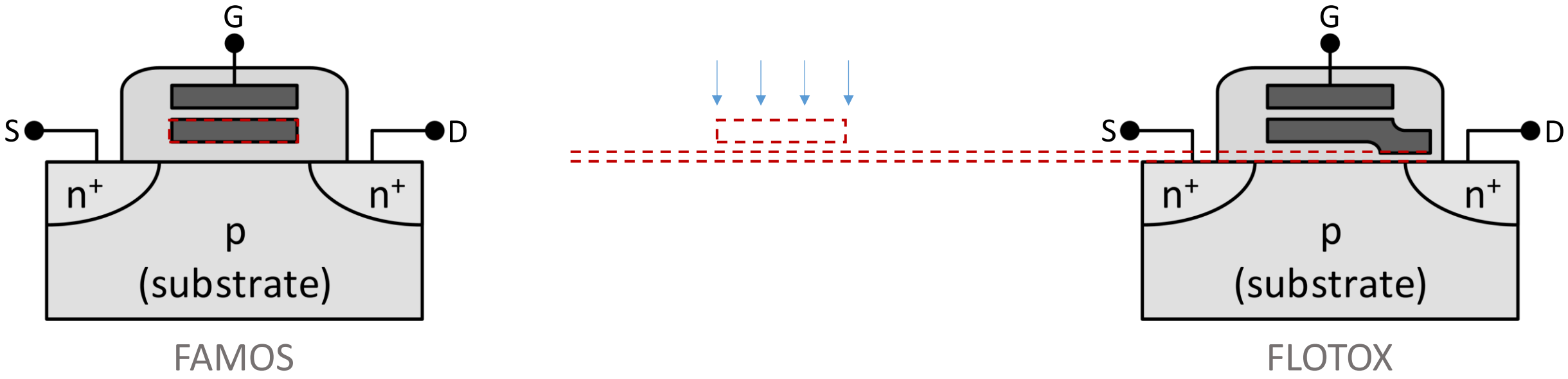
FAMOS



FLOTOX

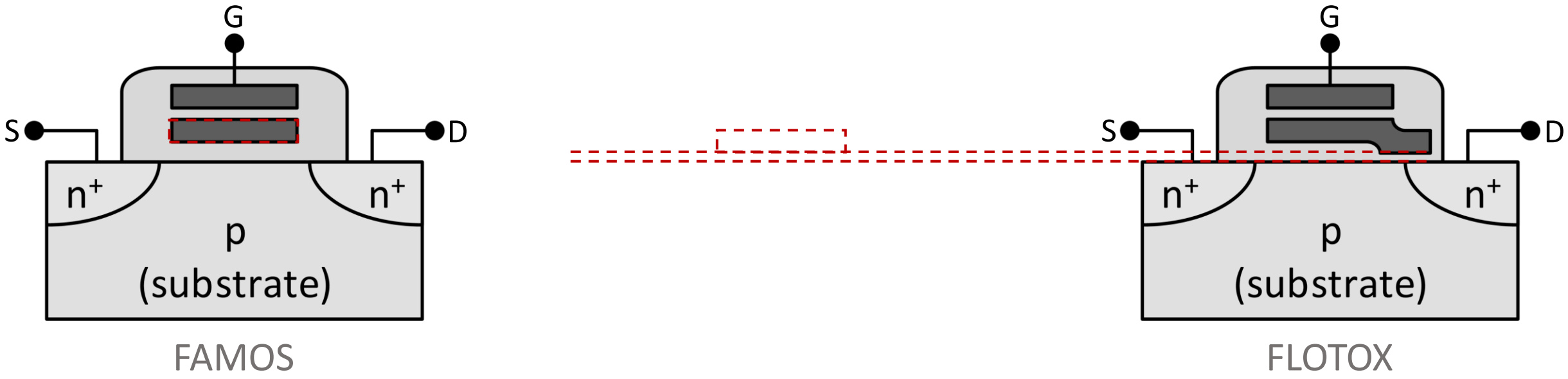
Flash transistor

- Another FGMOS
  - Merging characteristics of FAMOS and FLOTOX
    1. Symmetric floating gate
    2. Separated by a very thin oxide layer (from substrate)



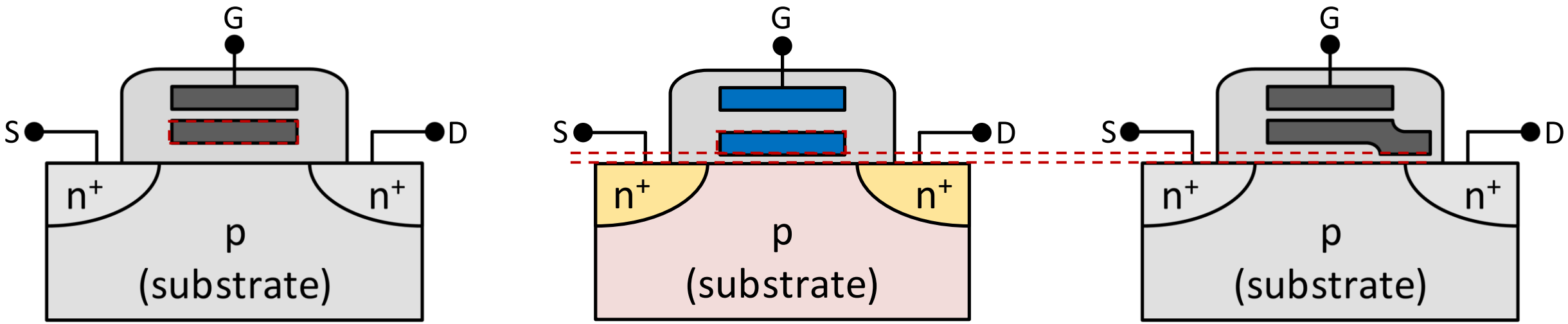
Flash transistor

- Another FGMOS
  - Merging characteristics of FAMOS and FLOTOX
    1. Symmetric floating gate
    2. Separated by a very thin oxide layer (from substrate)



Flash transistor

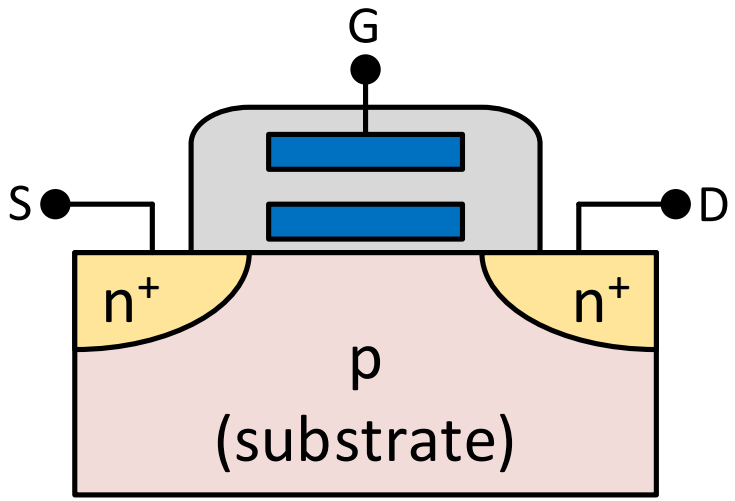
- Another FGMOS
  - Merging characteristics of FAMOS and FLOTOX
    1. Symmetric floating gate
    2. Separated by a very thin oxide layer (from substrate)



FAMOS **Flash** FLOTOX

Flash transistor

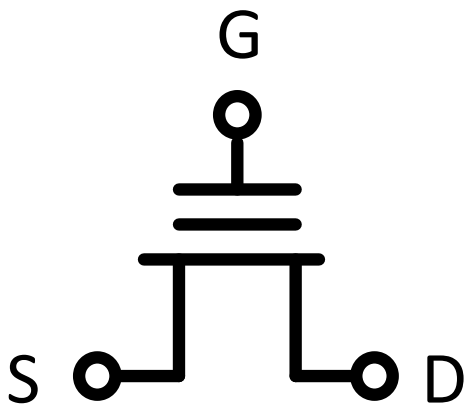
- Another FGMOS
  - Merging characteristics of FAMOS and FLOTOX
    1. Symmetric floating gate
    2. Separated by a very thin oxide layer (from substrate)



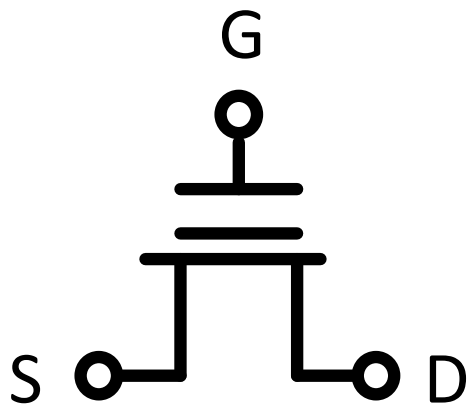
Flash

Flash transistor

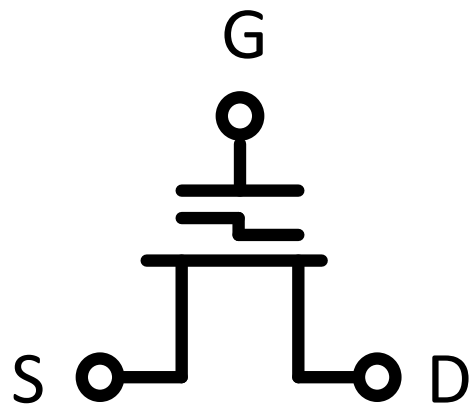
- Another FGMOS
  - Symbol



FAMOS



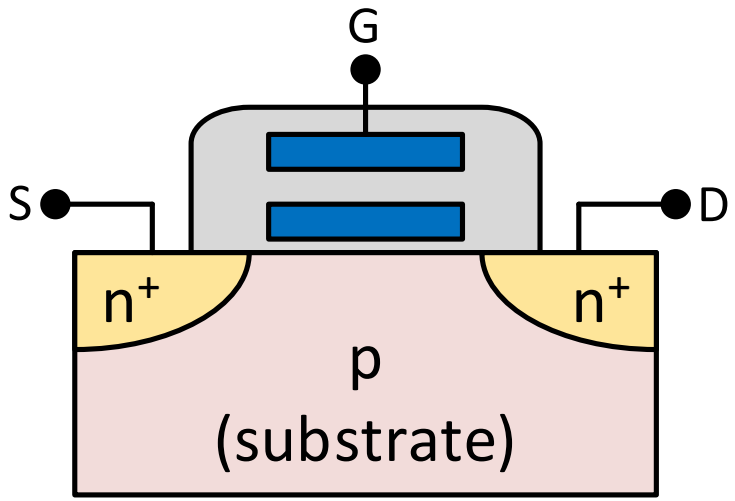
Flash



FLOTOX

Flash transistor

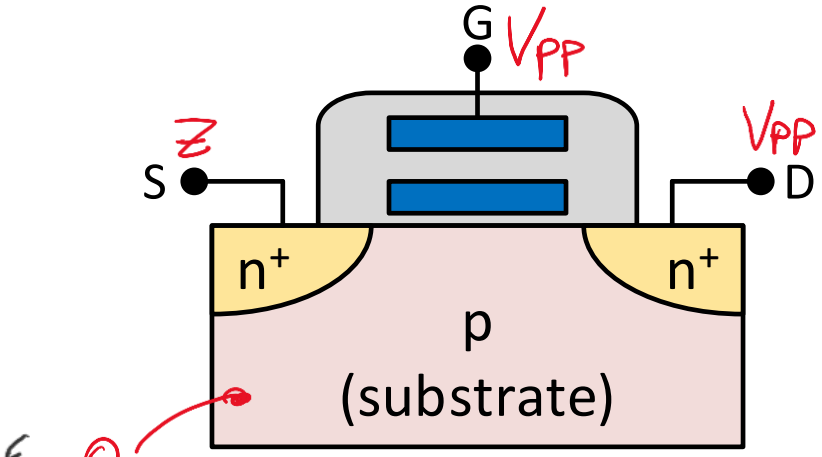
- Another FGMOS
  - Both programming and erasing depend on the tunnel effect (like FLOTOX), but
    - Using substrate: V_{sub}
    - Programming similar to FAMOS (using also Drain)!

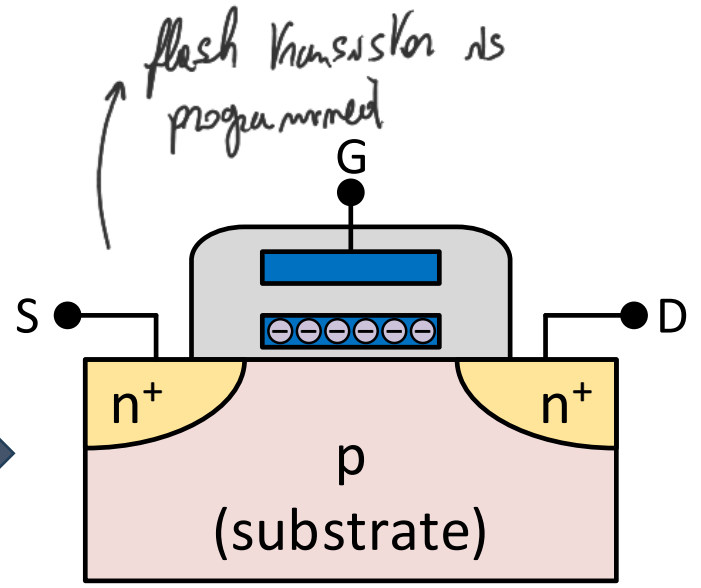


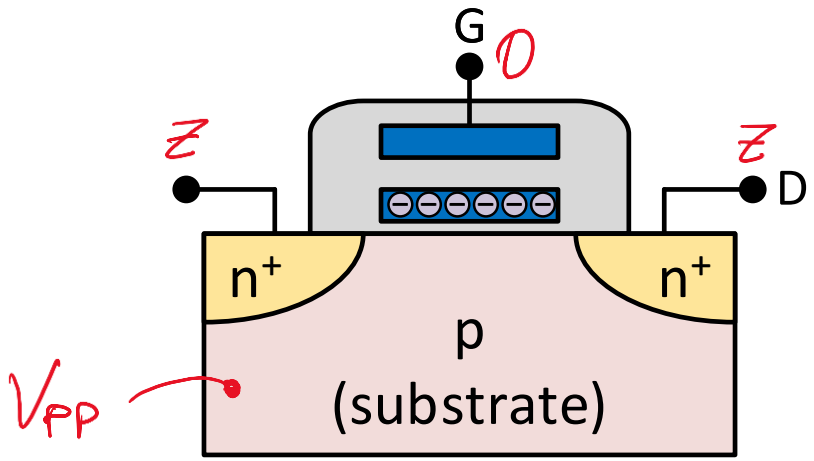
Flash transistor

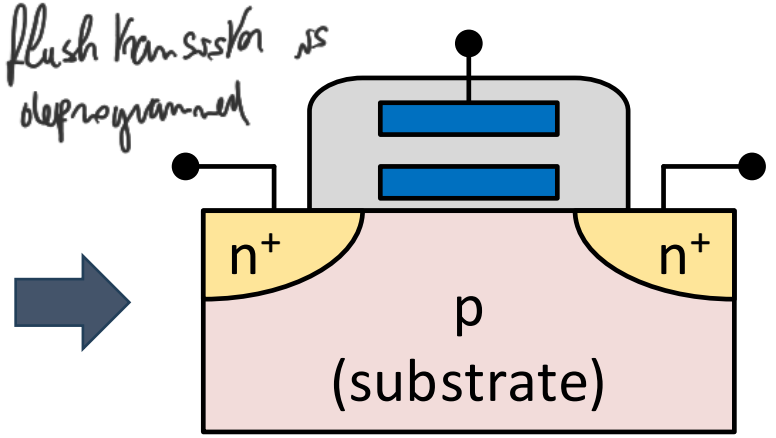
- **Programming**
  - $V_G = V_D = V_{PP}$
    - like FAMOS
  - $V_{sub} = 0$
  - $V_S = Z$
- **Erasing**
  - $V_G = 0$
  - $V_{sub} = V_{PP}$
  - $V_S = V_D = Z$

HIGH VOLTAGE



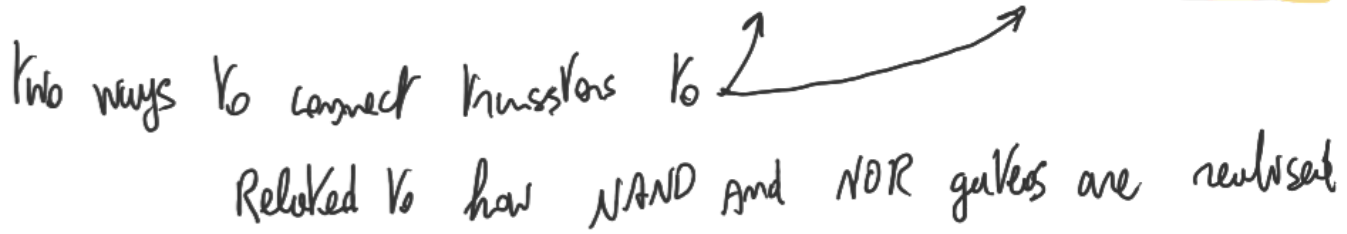






Flash memory

- Two main categories
  - Changing how (flash) transistors are connected to the Word Lines and Bit Lines



Flash memory

- Two main categories
  - Changing how (flash) transistors are connected to the Word Lines and Bit Lines

Type of Flash Memory	NAND
Cost per bit (density)	Low (High) density)
Erase speed	High
Write speed	High
Supporting random read	No (organized in blocks)
Data retention	Medium (10 years)
Endurance	High
Application	Data storage

Flash memory

- Two main categories
  - Changing how (flash) transistors are connected to the Word Lines and Bit Lines

Type of Flash Memory	NAND	NOR
Cost per bit (density)	Low (High)	High (Low)
Erase speed	High	Low (about 150 times slower)
Write speed	High	Low
Supporting random read	No (organized in blocks)	Yes
Data retention	Medium (10 years)	High (20 years)
Endurance	High	Low
Application	Data storage	Code storage and execution

Flash memory

- Two main categories
  - Changing how (flash) transistors are connected to the Word Lines and Bit Lines

amount of bits that you can realise on the same piece of silicon
Lower cost → more bits

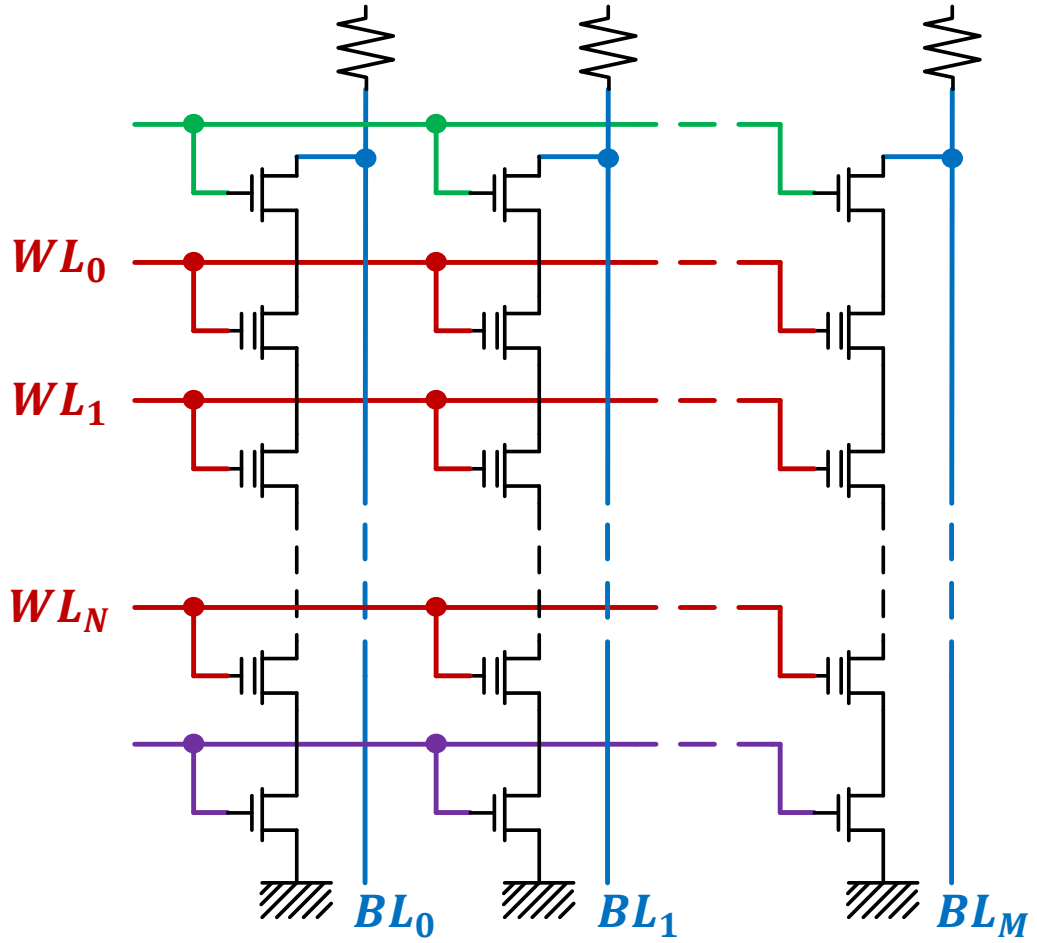
Type of Flash Memory	NAND	NOR
Cost per bit (density)	Low (High)	High (Low)
Erase speed	High	Low (about 150 times slower)
Write speed	High	Low
Supporting random read	No (organized in blocks)	Yes
Data retention	Medium (10 years)	High (20 years)
Endurance	High	Low
Application	**Data storage**	Code storage and execution

Mainly because of random access

USB drives and SSDs!!!

(NAND) Flash memory

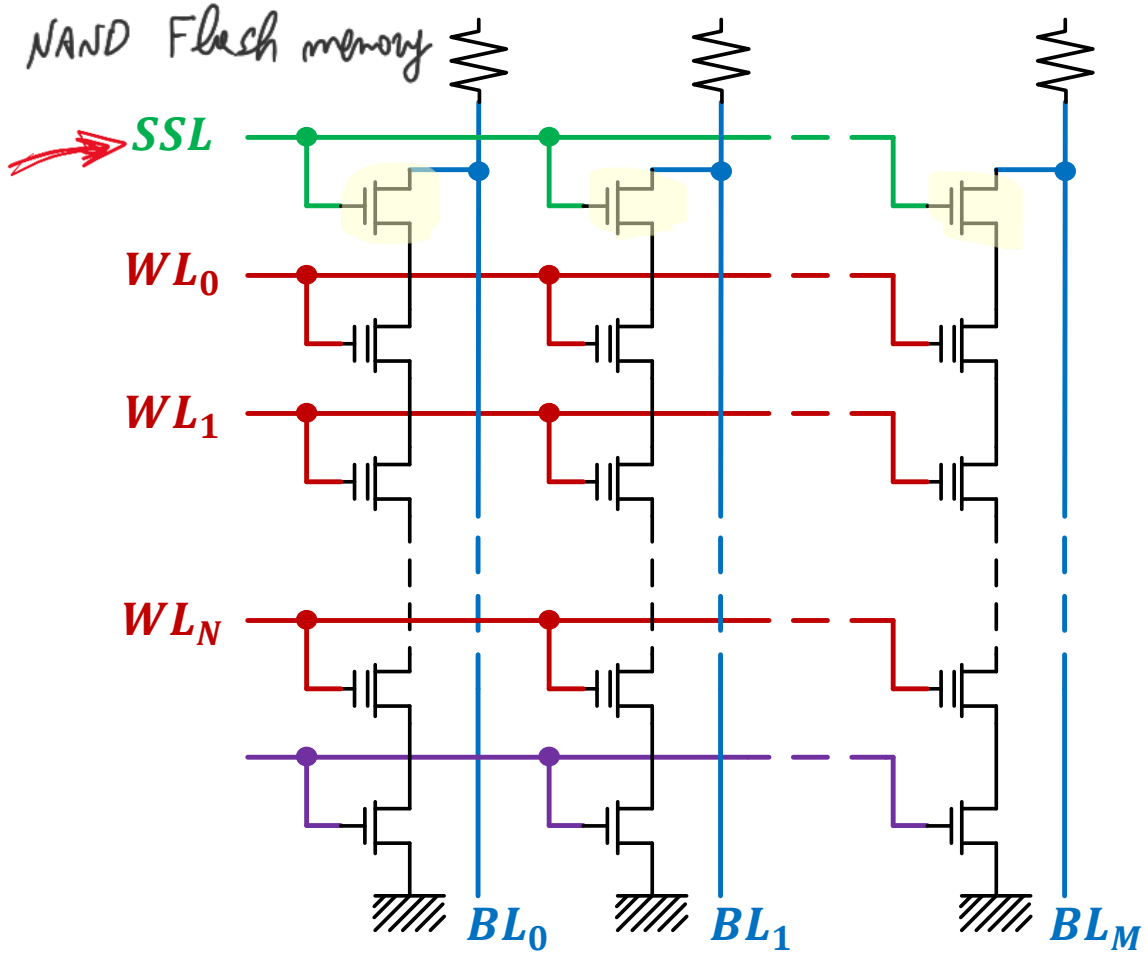
- Architecture outline
  - WL_i and BL_j well known



NOT Going to ask to show the functioning of a NAND Flash memory

(NAND) Flash memory

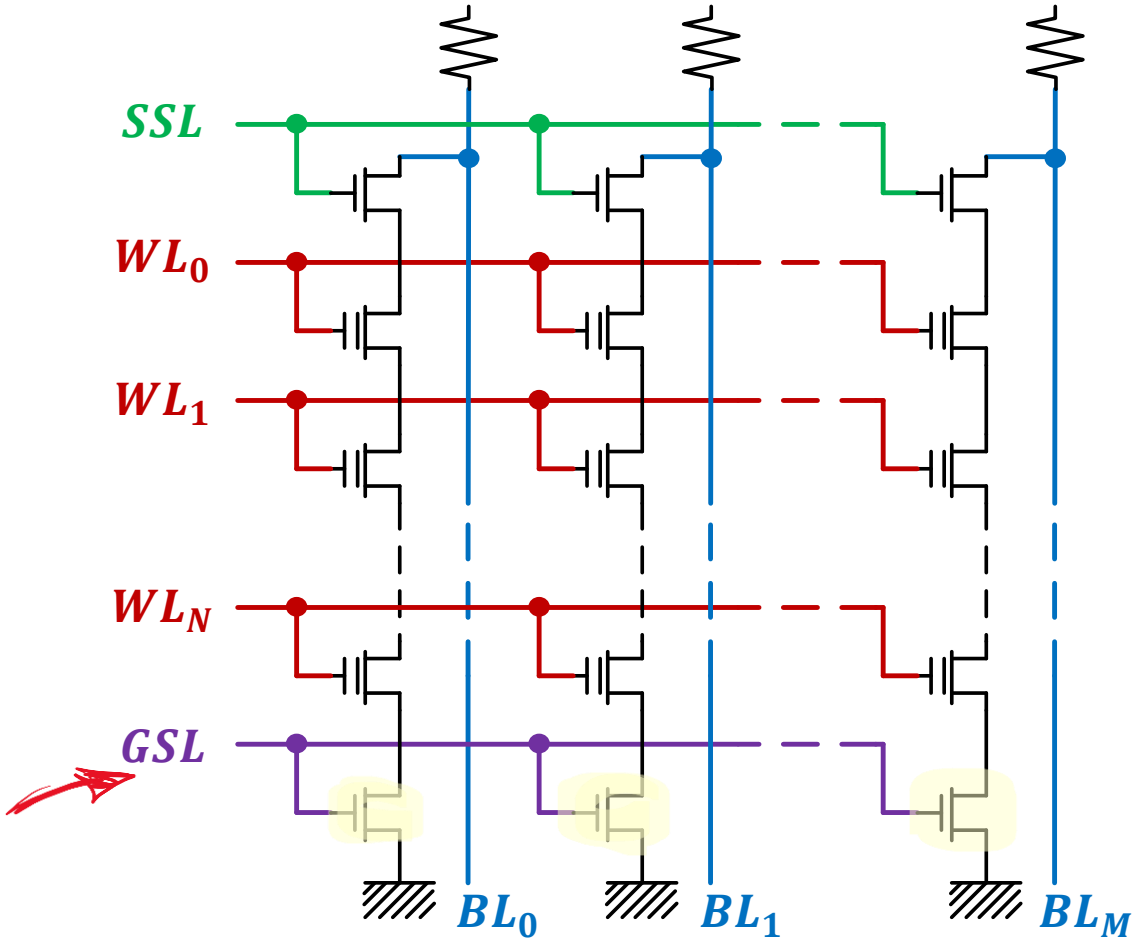
- Architecture outline
  - SSL = String Select Line
    - Driving pass transistors (n-MOS)



(NAND) Flash memory

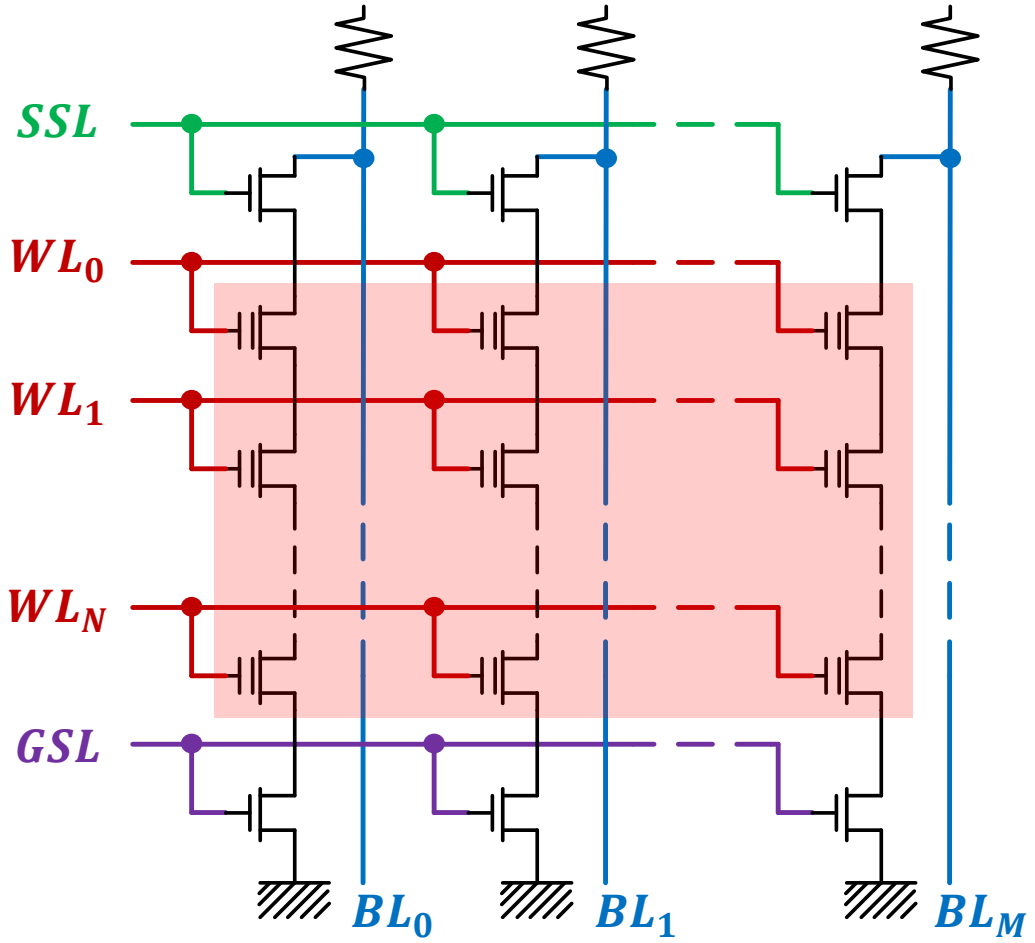
- Architecture outline
  - GSL = Ground Select Line
    - Driving pass transistors (n-MOS)

↑ Connect middle transistors (memory) to ground



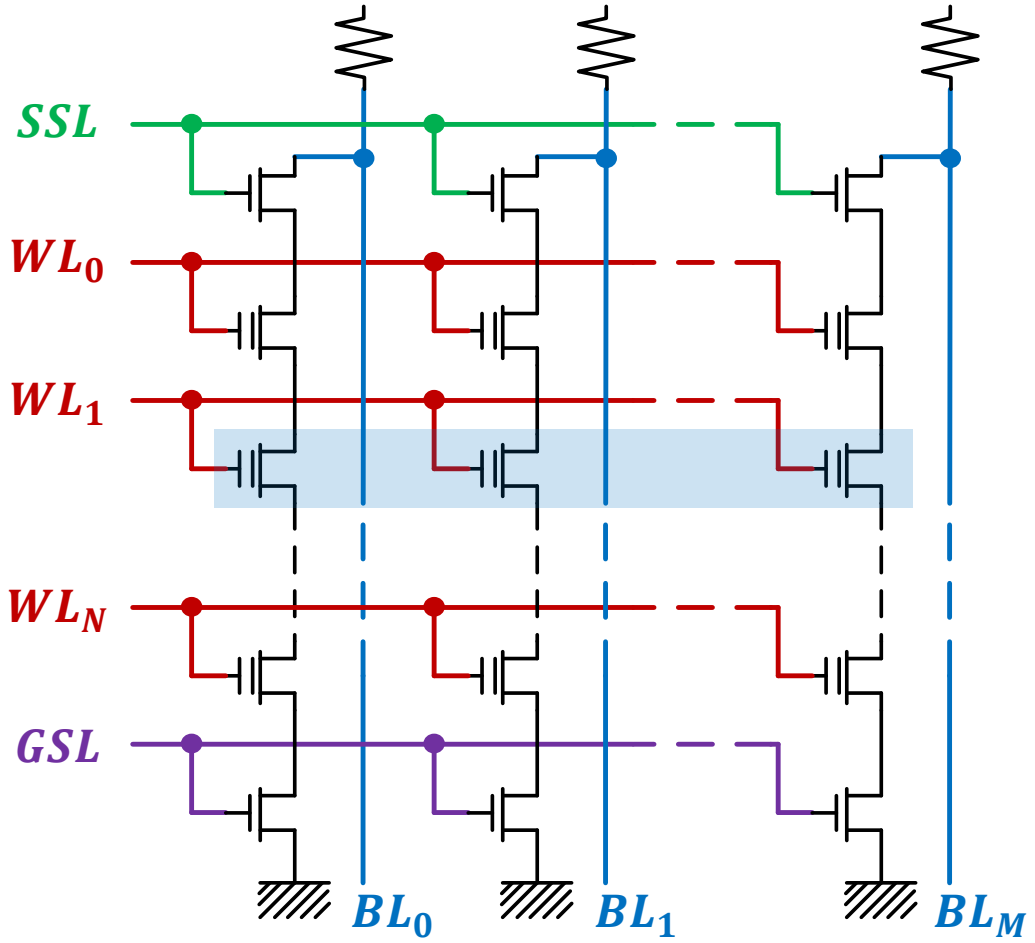
(NAND) Flash memory

- Architecture outline
  - Block
    - All the Flash transistors/memory cells
    - Sharing substrate!!! For programming



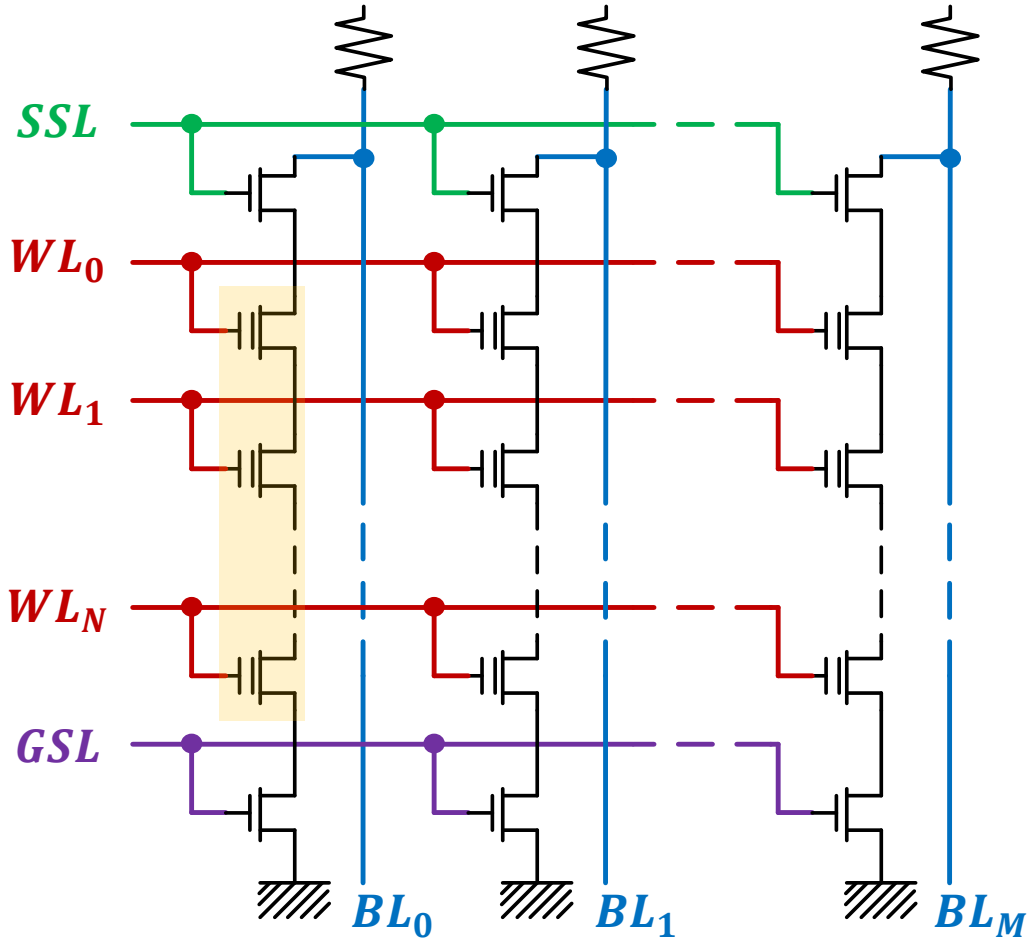
(NAND) Flash memory

- Architecture outline
  - Page



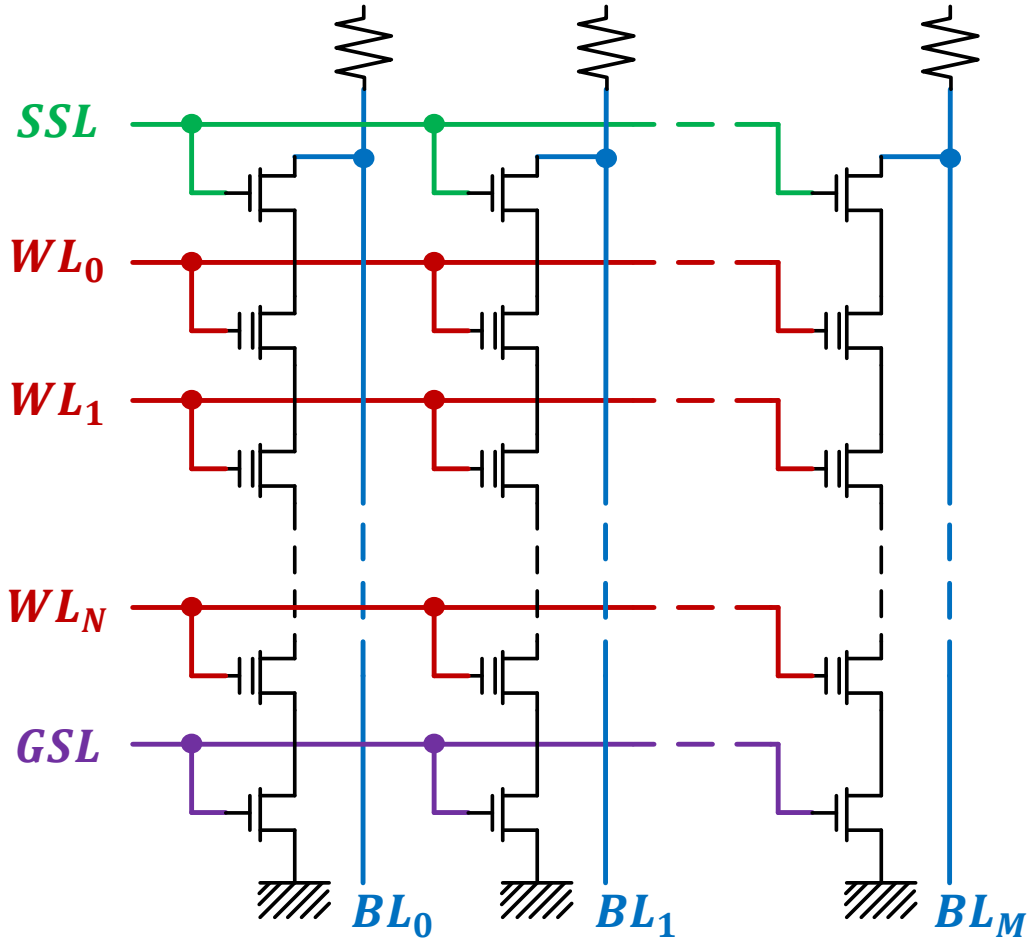
(NAND) Flash memory

- Architecture outline
  - String
    - [32 ÷ 128] transistors



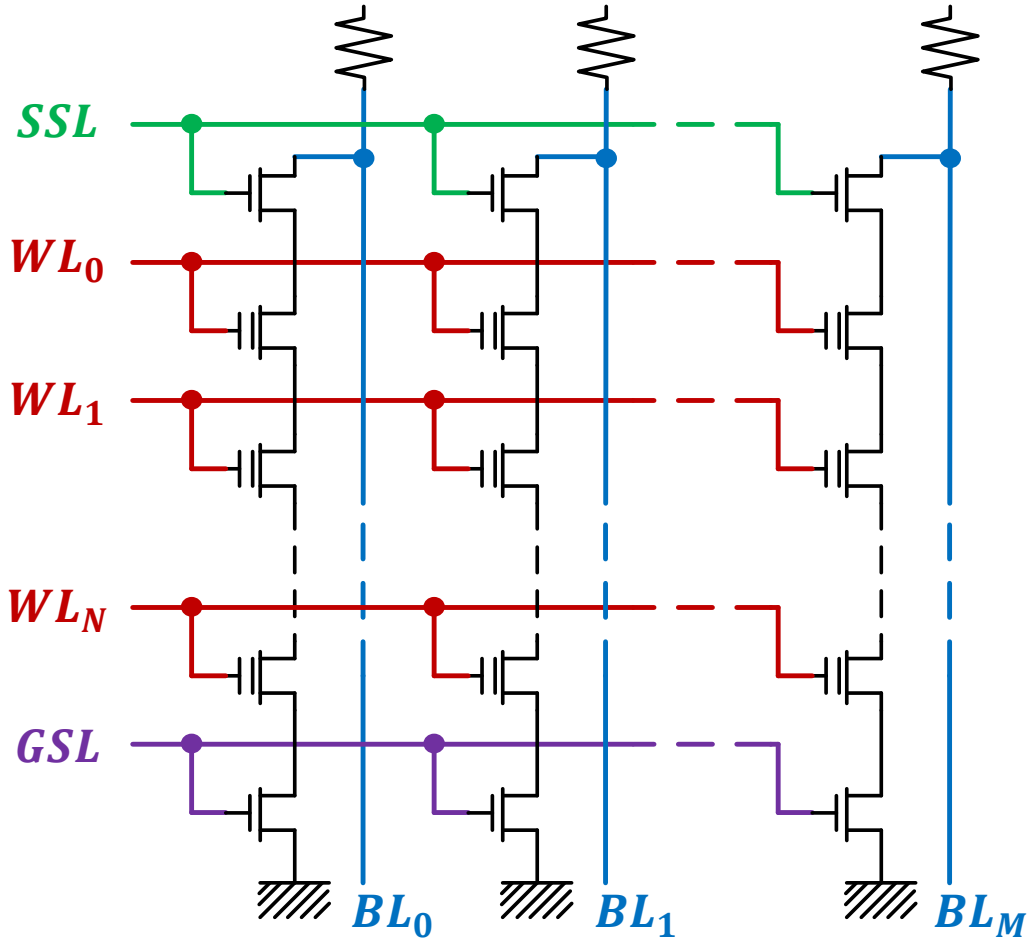
(NAND) Flash memory

- Working principle
  - Read by page
  - Program by page
  - Erase by block



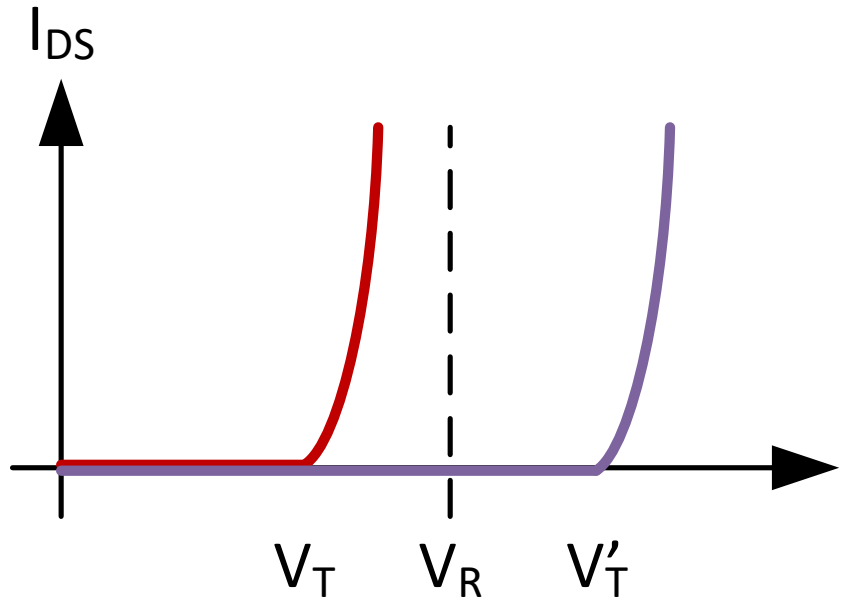
(NAND) Flash memory

- Working principle
  - Read by page
  - Program by page
  - Erase by block
- But before …



(NAND) Flash memory

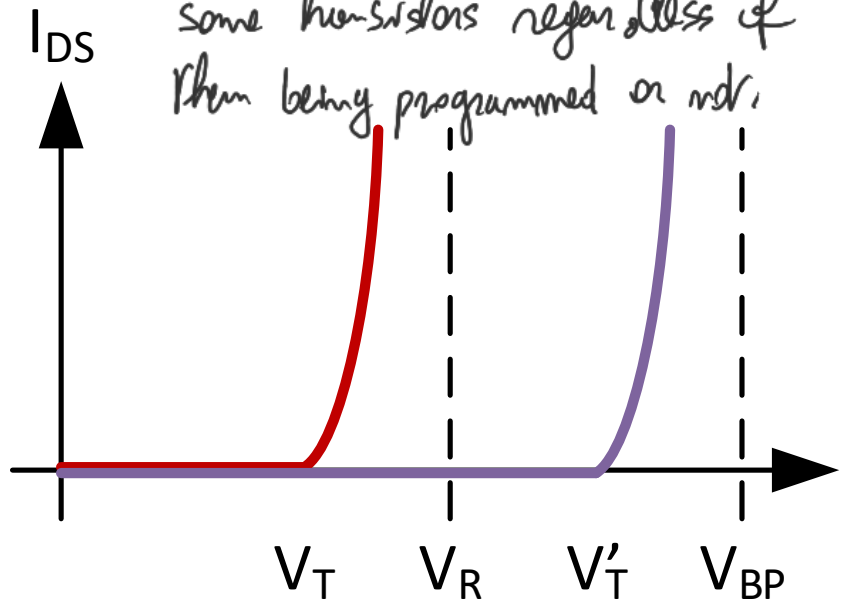
- Like any other FGMOS
  - $V_T < V_R < V_T'$
    - If flash transistor not programmed (erased)
      - Transistor = ON (short circuit), when applying V_R
    - If flash transistor programmed
      - Transistor = OFF (open circuit), when applying V_R



(NAND) Flash memory

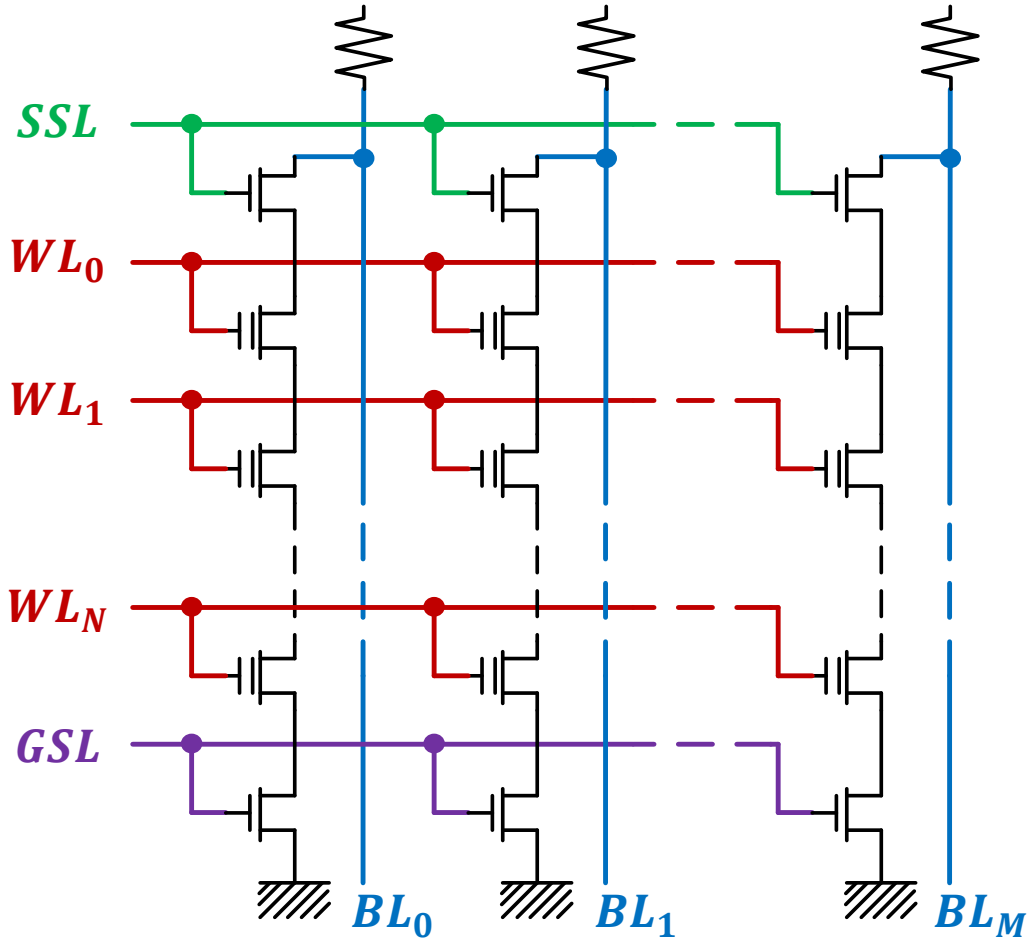
- Like any other FGMOS
  - $V_T < V_R < V_T'$
    - If flash transistor not programmed (erased)
      - Transistor = ON (short circuit), when applying V_R
    - If flash transistor programmed
      - Transistor = OFF (open circuit), when applying V_R
- In addition
  - $\boldsymbol{V_{BP}}$ = **bypass** voltage
    - $V_{BP} > V_T' \rightarrow$ **transistor always ON** (short circuit)

Need to remember that for erasing, recovery, programming you need to use V_{BP} to activate some transistors regardless of them being programmed or not.



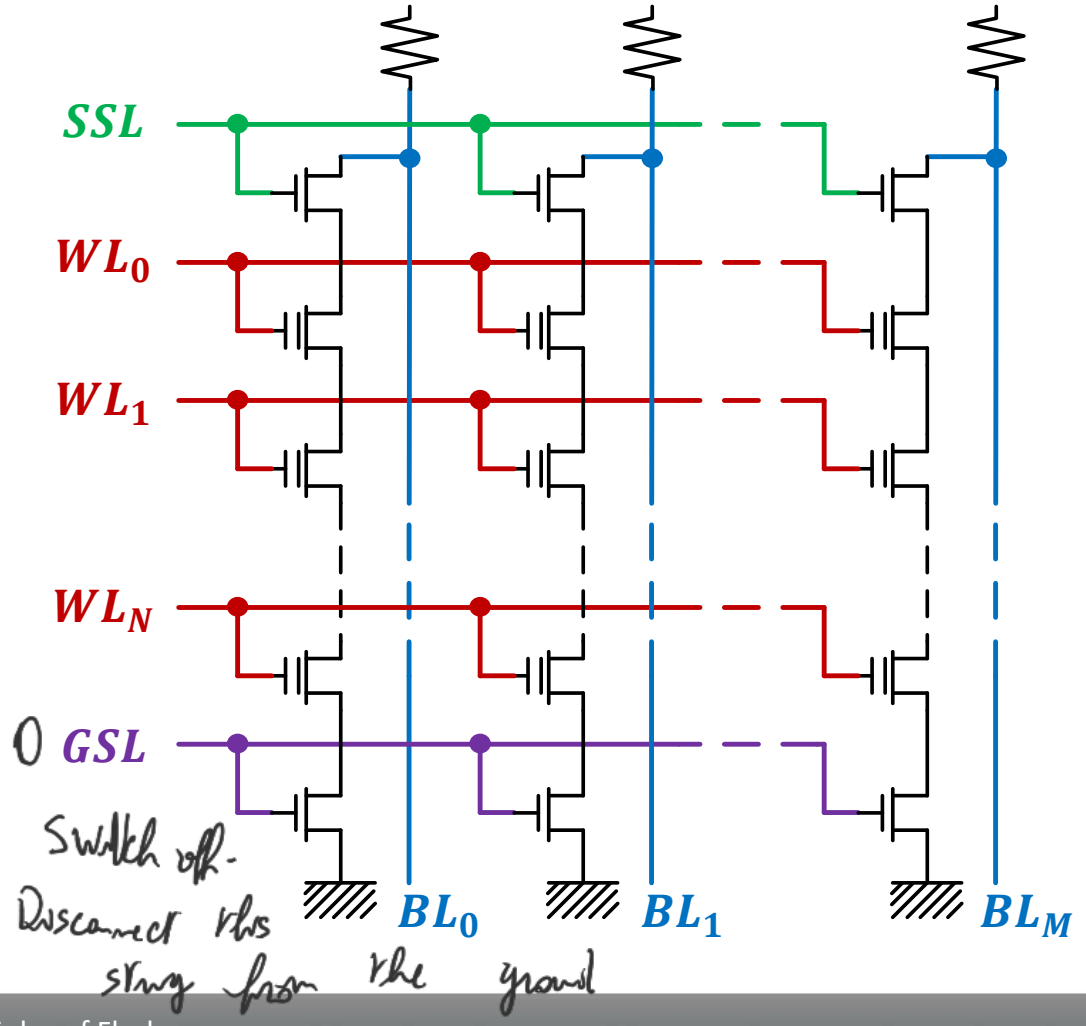
(NAND) Flash memory

- Working principle
  - Program by page – Example



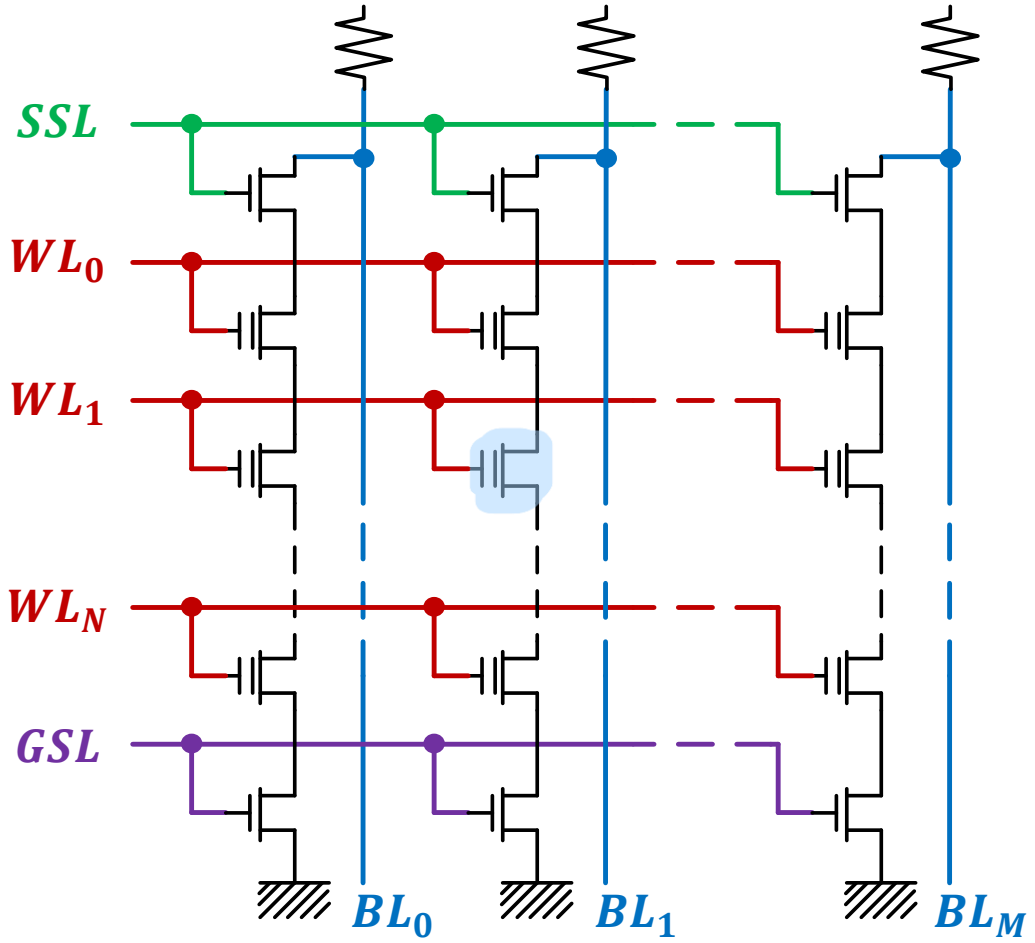
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0$ $(0\ V)$
    - $SSL = 1$ (V_{CC})
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs



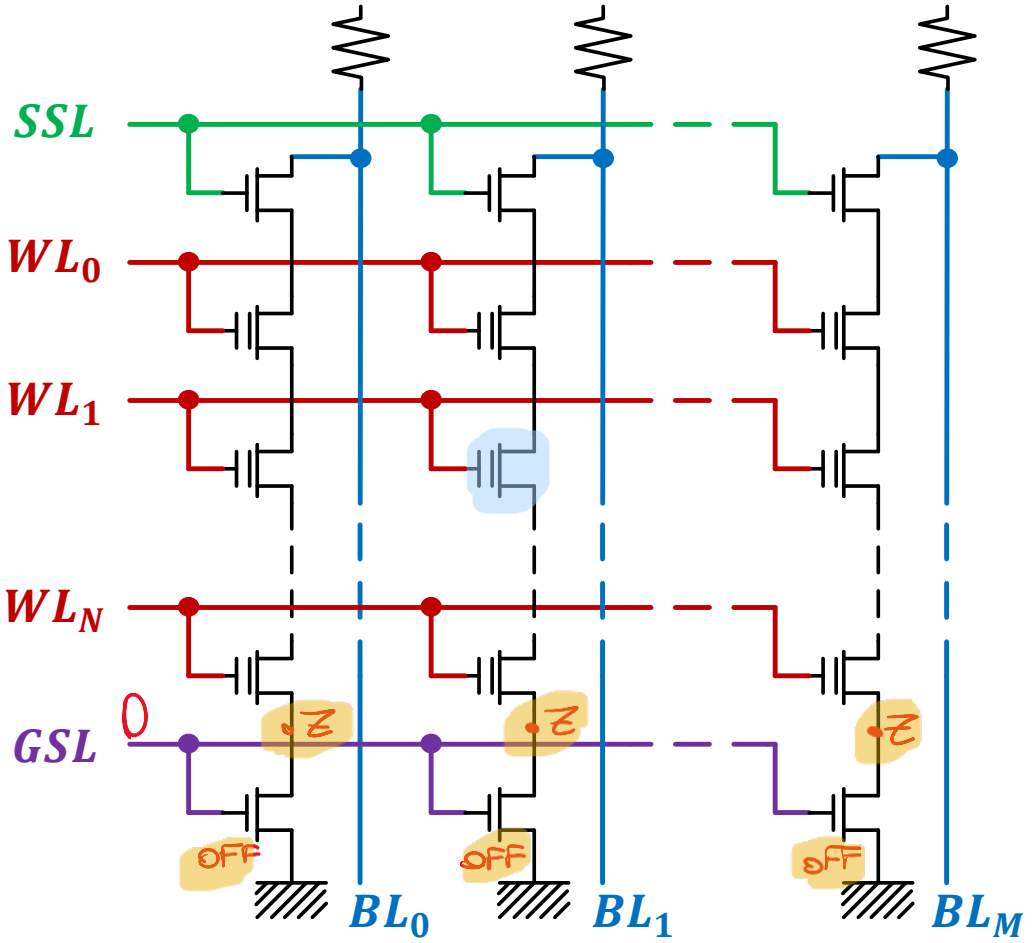
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0$ $(0\ V)$
    - $SSL = 1$ (V_{CC})
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs
    - Let's assume to program cell (0,1)



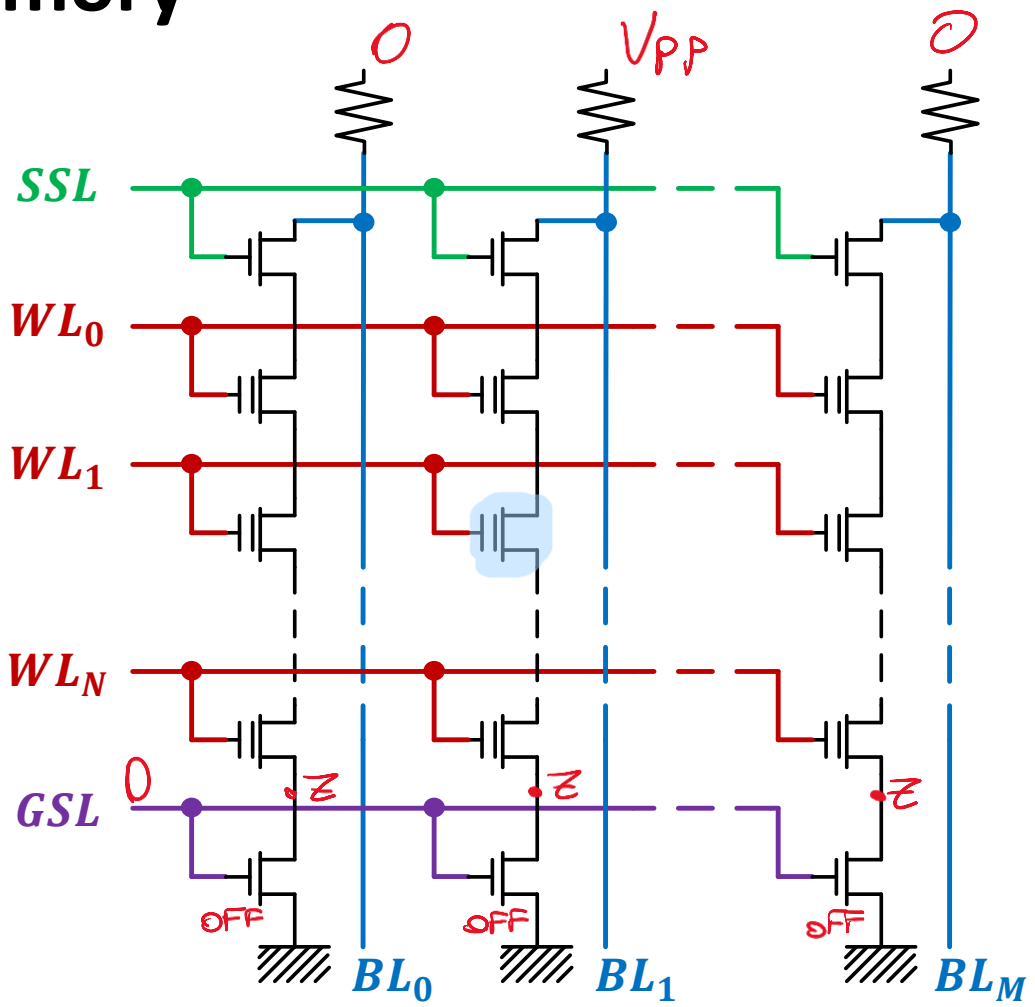
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0 \qquad (0\ V)$
    - $SSL = 1 \qquad (V_{CC})$
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs
    - Let's assume to program cell (0,1)



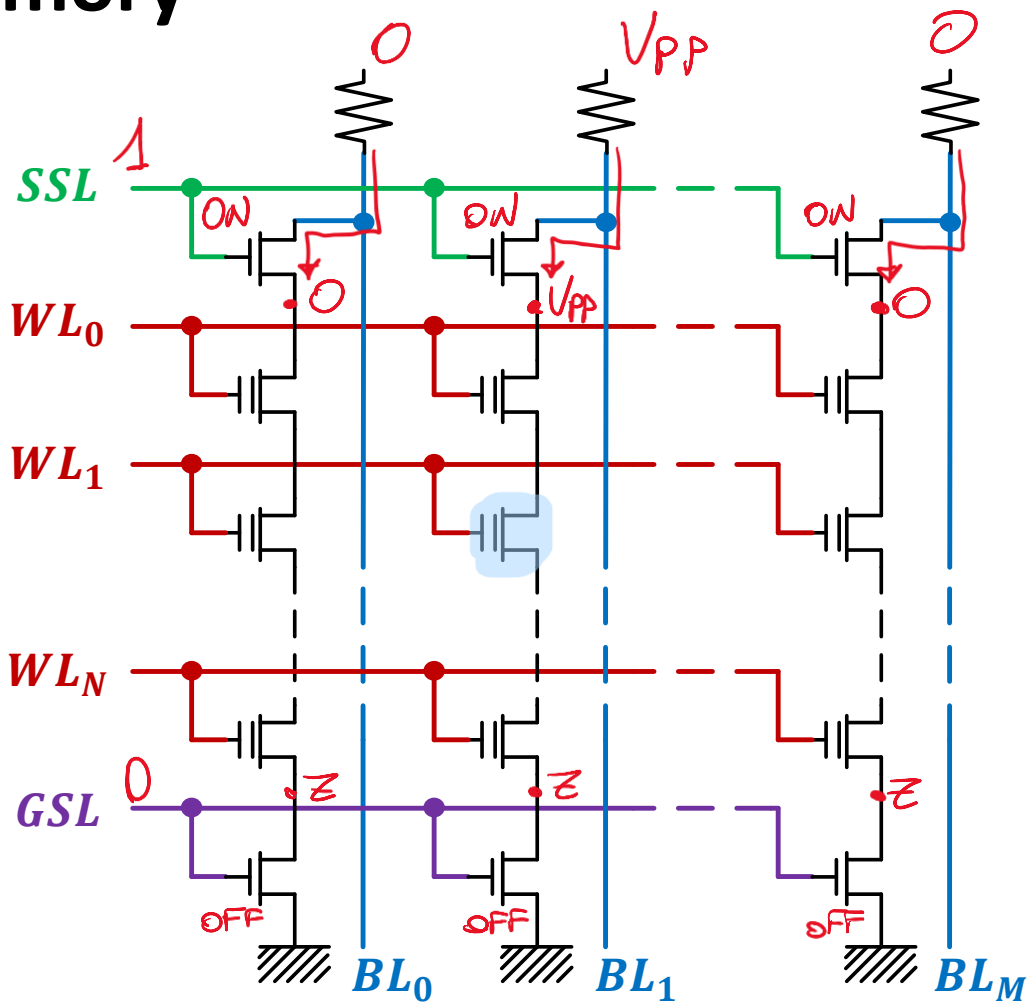
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0 \qquad (0\ V)$
    - $SSL = 1 \qquad (V_{CC})$
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs
    - Let's assume to program cell (0,1)



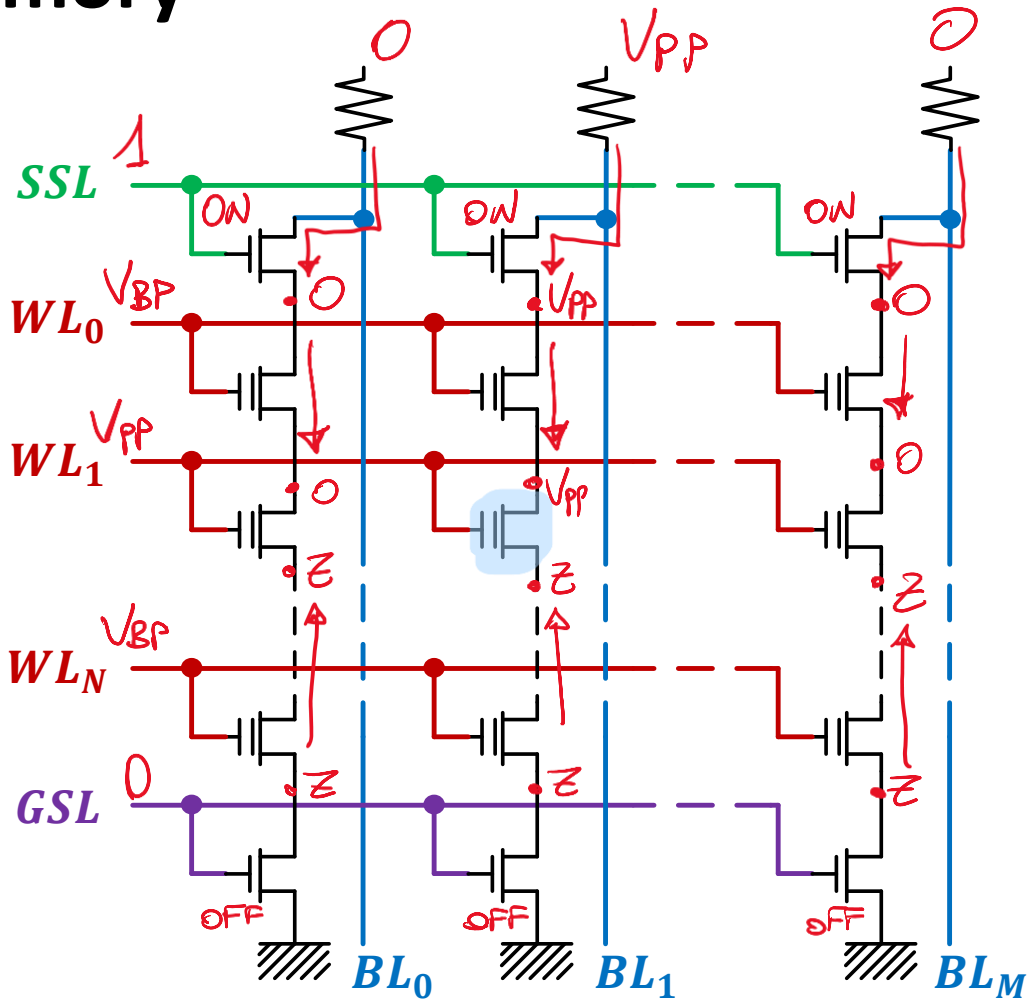
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0 \qquad (0\ V)$
    - $SSL = 1 \qquad (V_{CC})$
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs
    - Let's assume to program cell (0,1)



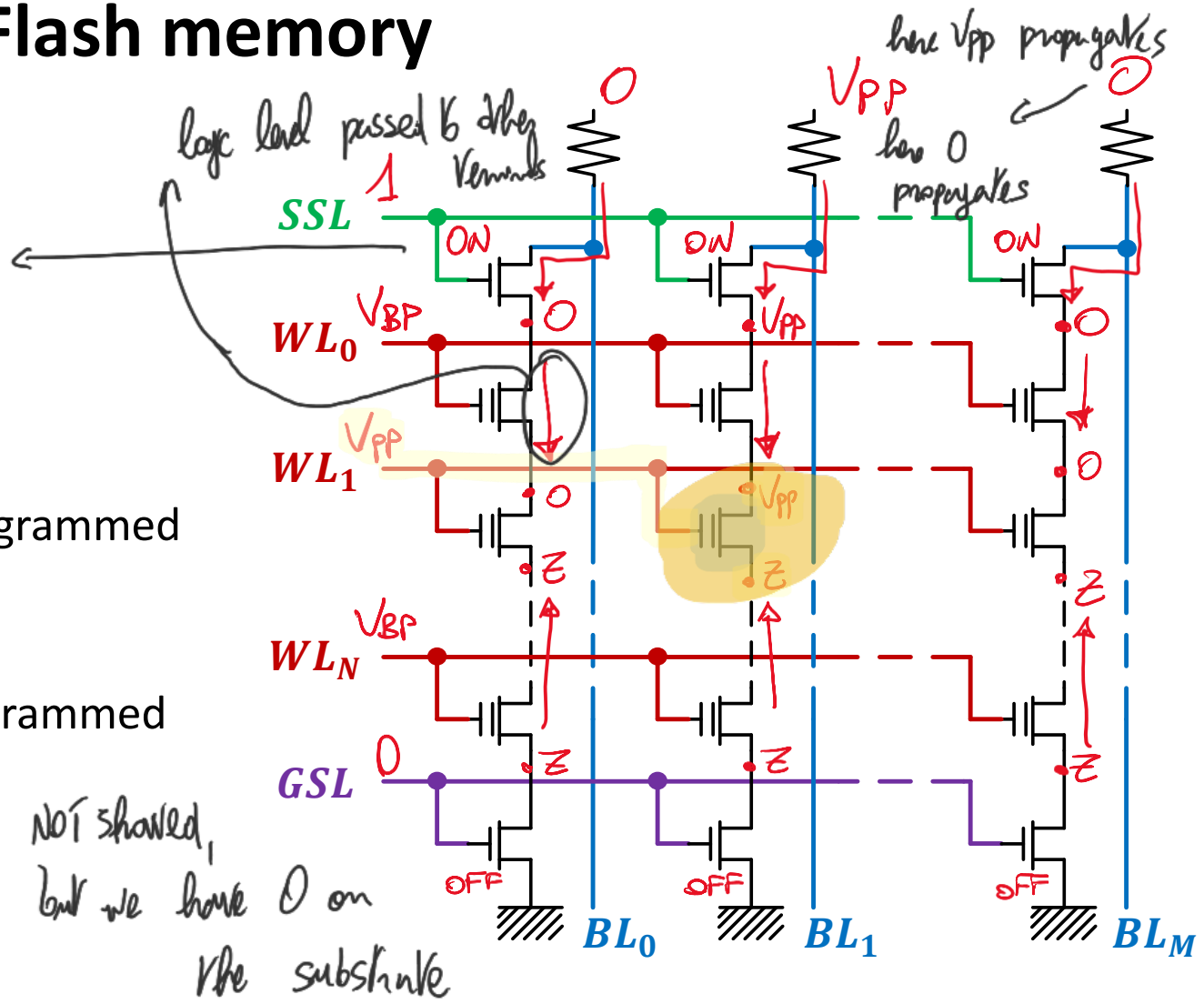
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0 \qquad (0\ V)$
    - $SSL = 1 \qquad (V_{CC})$
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs
    - Let's assume to program cell (0,1)



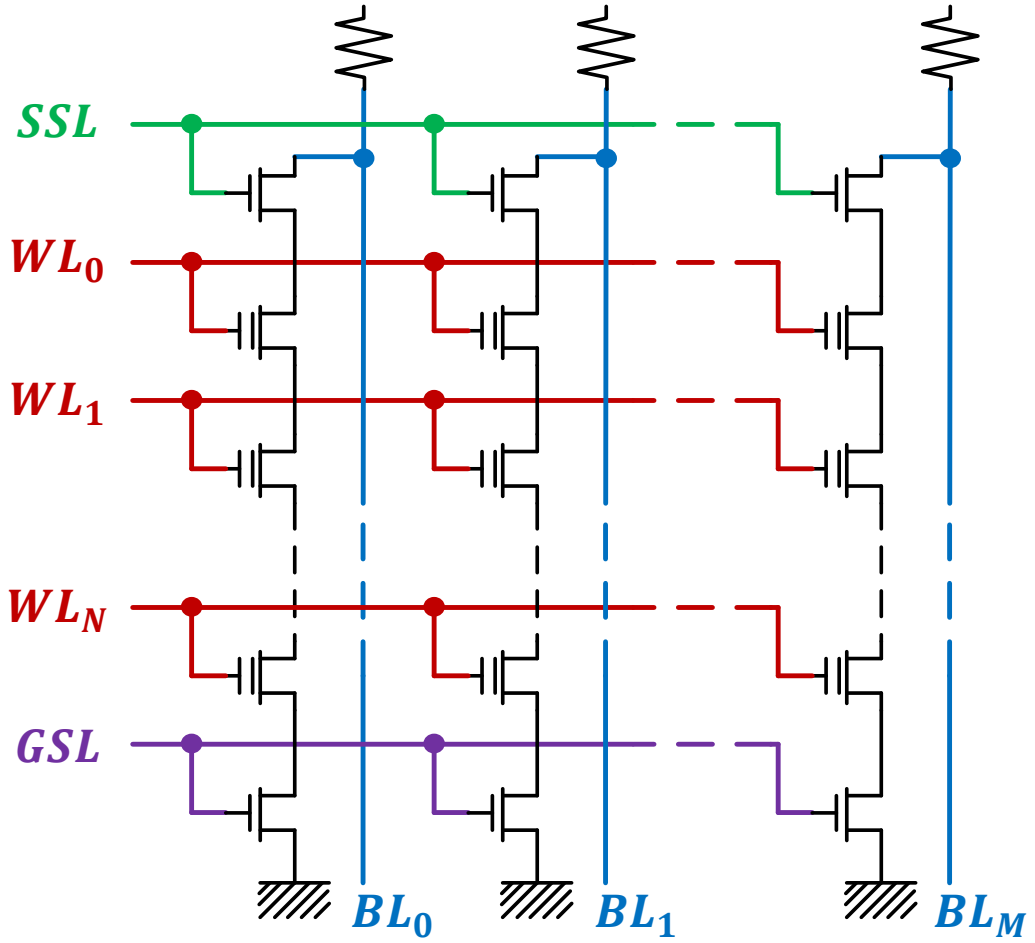
(NAND) Flash memory

- Working principle
  - Program by page – Example
    - $GSL = 0 \qquad (0\ V)$
    - $SSL = 1 \qquad (V_{CC})$
    - V_{PP} on WL of cells/transistors to be programmed
      - V_{BP} on all other WLs
    - V_{PP} on BL of cells/transistors to be programmed
      - 0 on all other BLs
    - Let's assume to program cell (0,1)



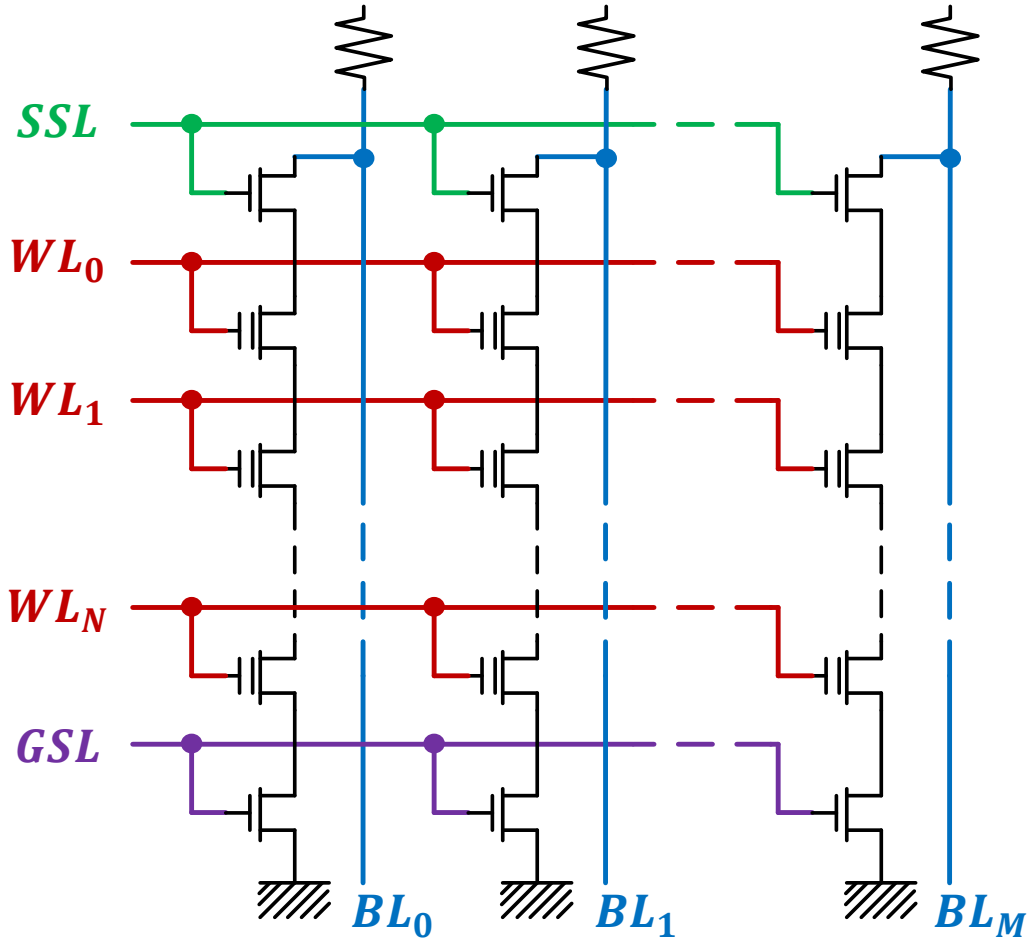
(NAND) Flash memory

- Working principle
  - Erase by block – Example



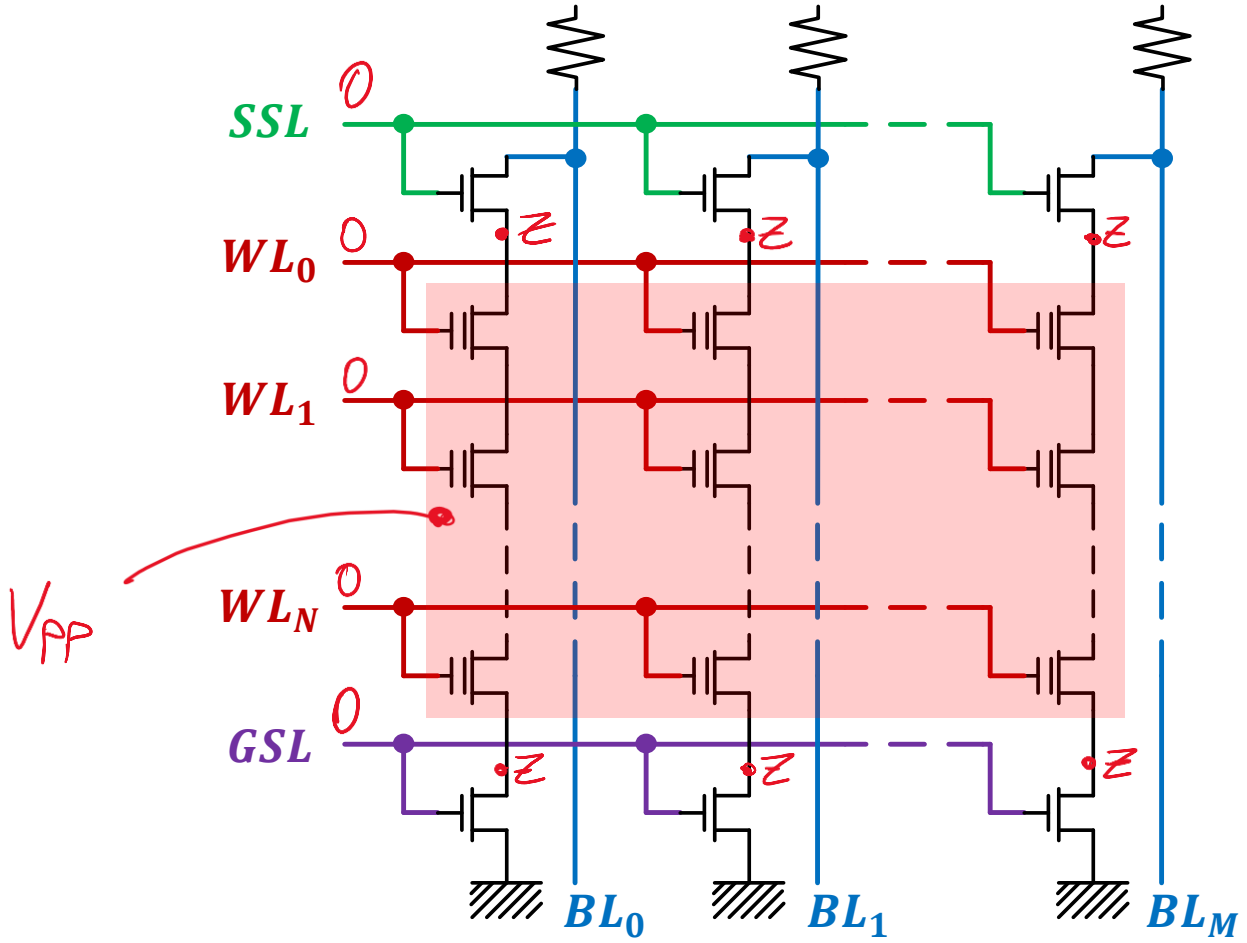
(NAND) Flash memory

- Working principle
  - Erase by block – Example
    - $GSL = 0$ $(0\ V)$
    - $SSL = 0$ $(0\ V)$
    - 0 on all WLs
    - V_{PP} on substrate



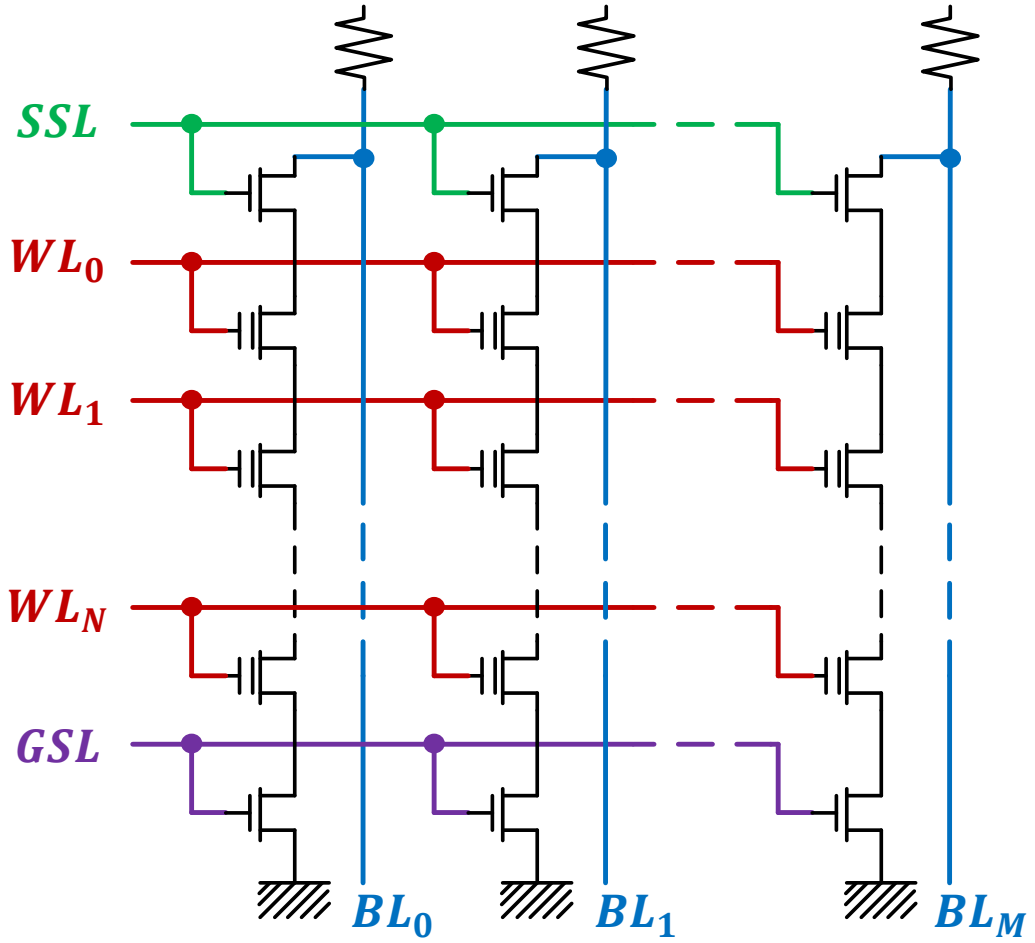
(NAND) Flash memory

- Working principle
  - Erase by block – Example
    - $GSL = 0 \qquad (0\ V)$
    - $SSL = 0 \qquad (0\ V)$
    - 0 on all WLs
    - V_{PP} on substrate



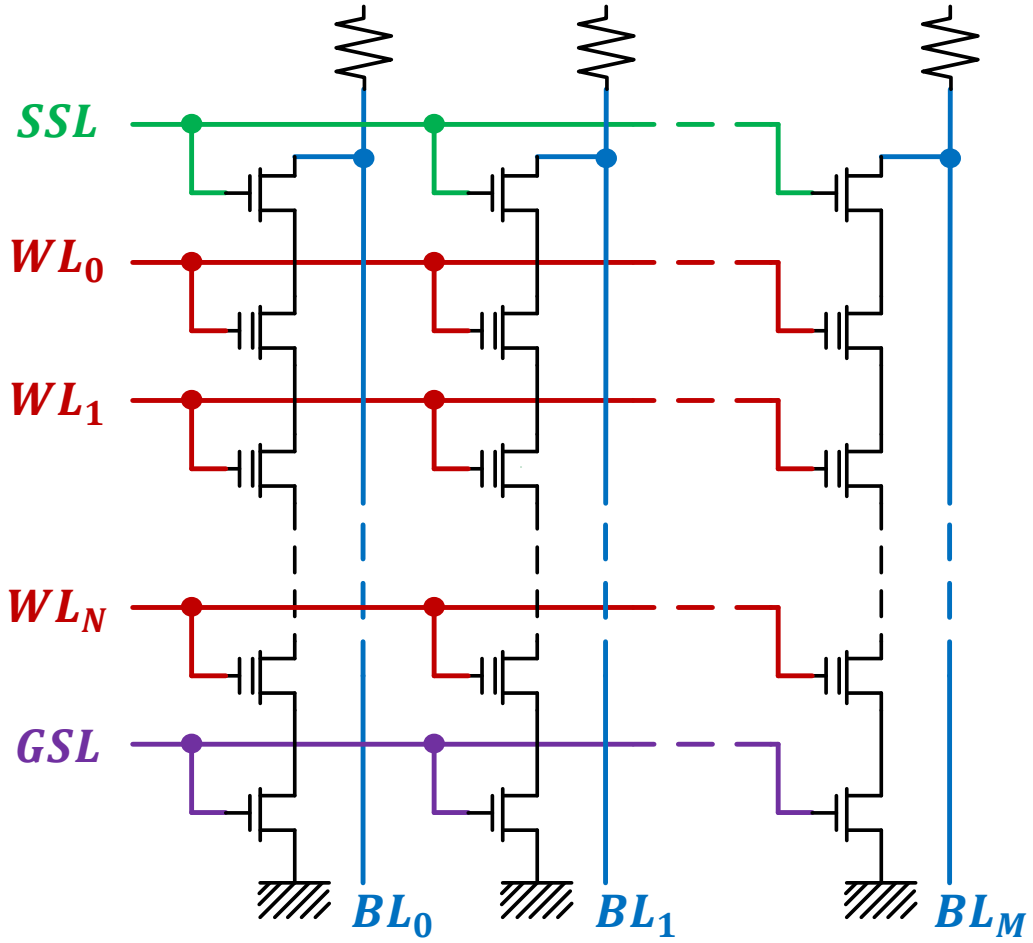
(NAND) Flash memory

- Working principle
  - Read by page – Example



(NAND) Flash memory

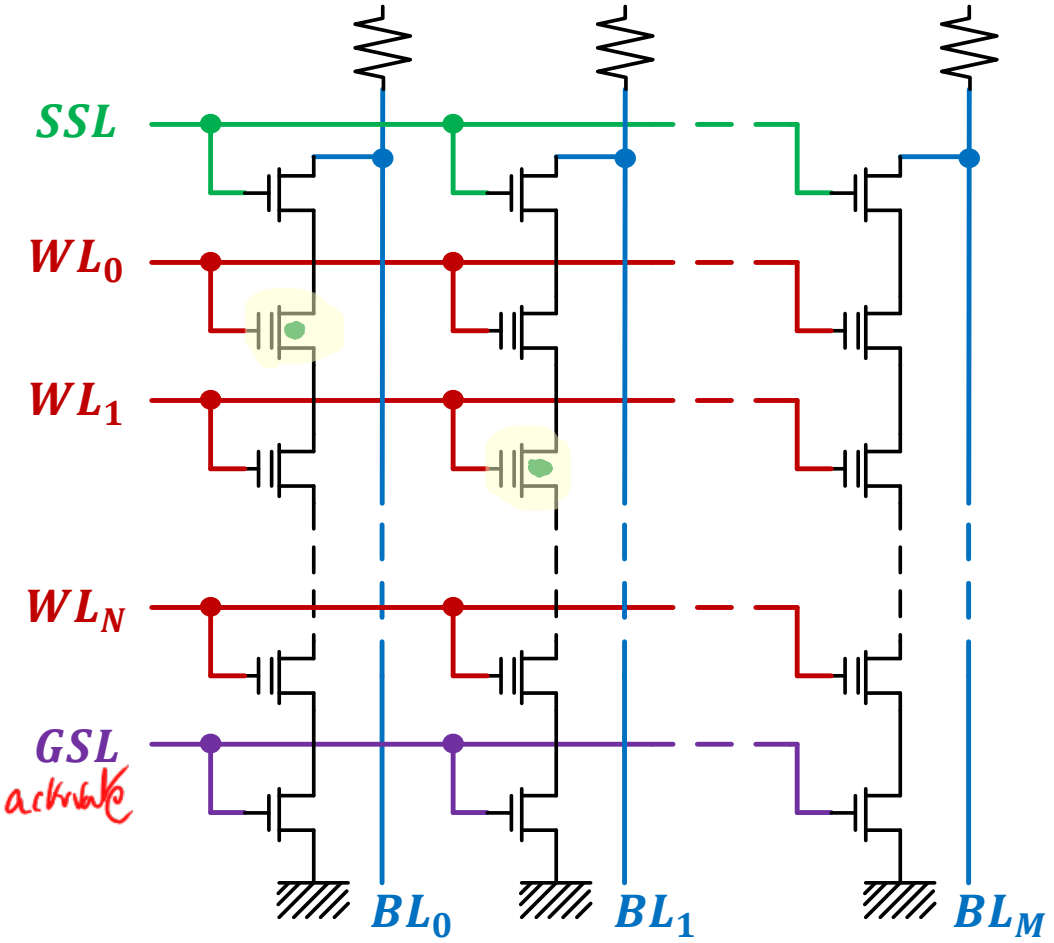
- Working principle
  - Read by page – Example
    - $GSL = 1$ (V_{CC})
    - $SSL = 1$ (V_{CC})
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs



(NAND) Flash memory

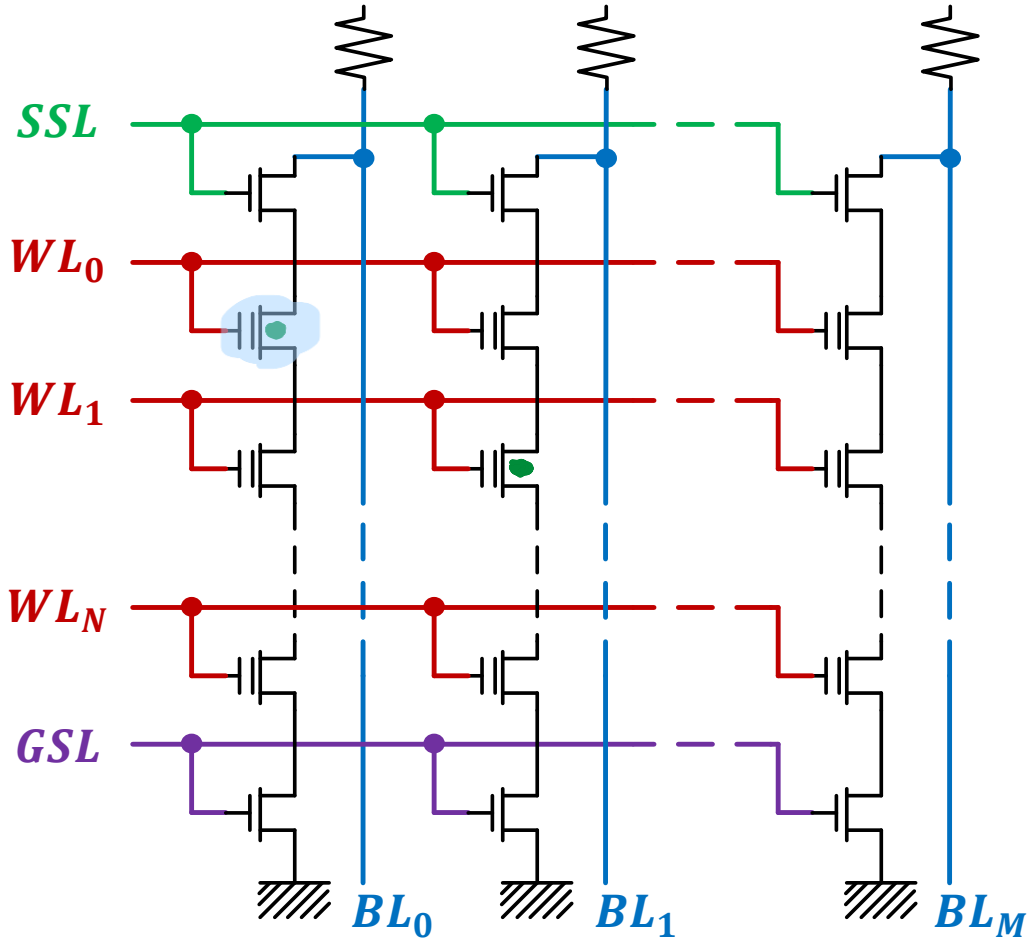
- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed

So they won't activate



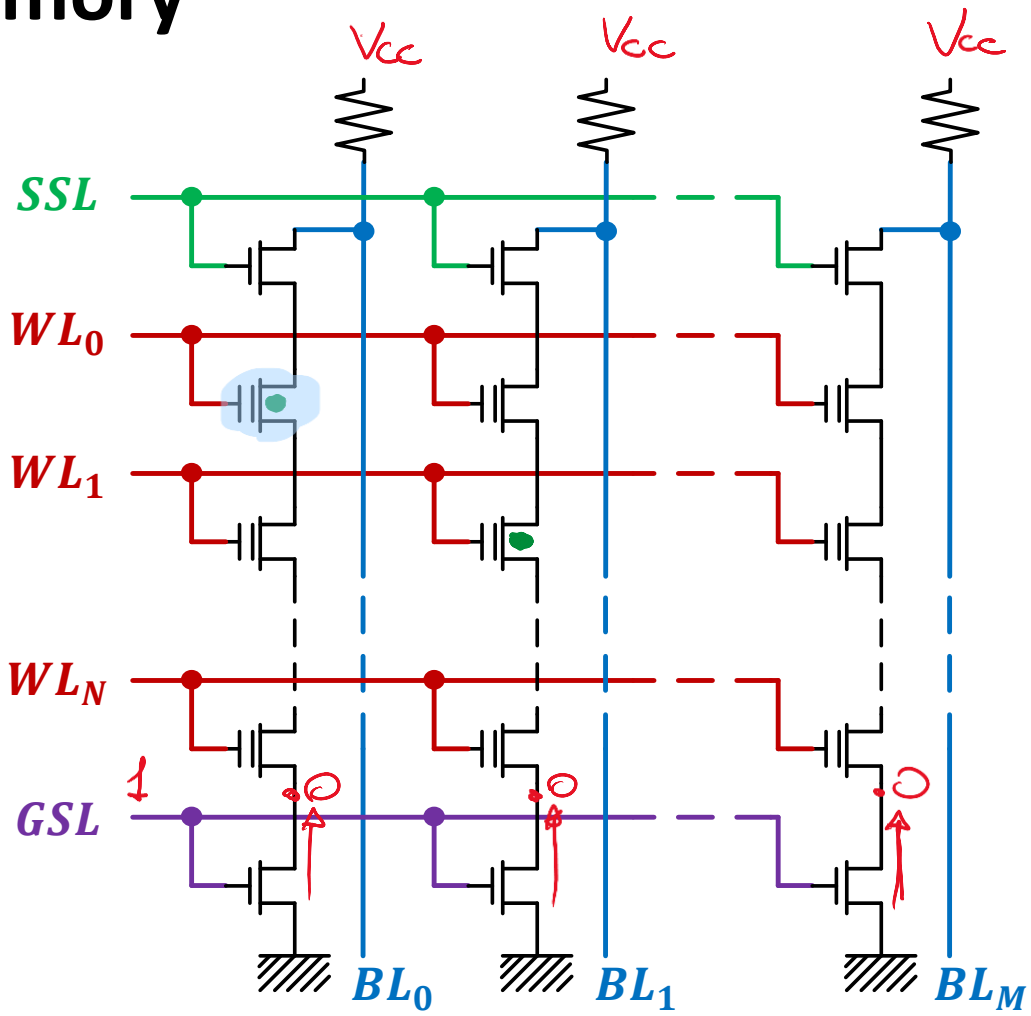
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (0,0)



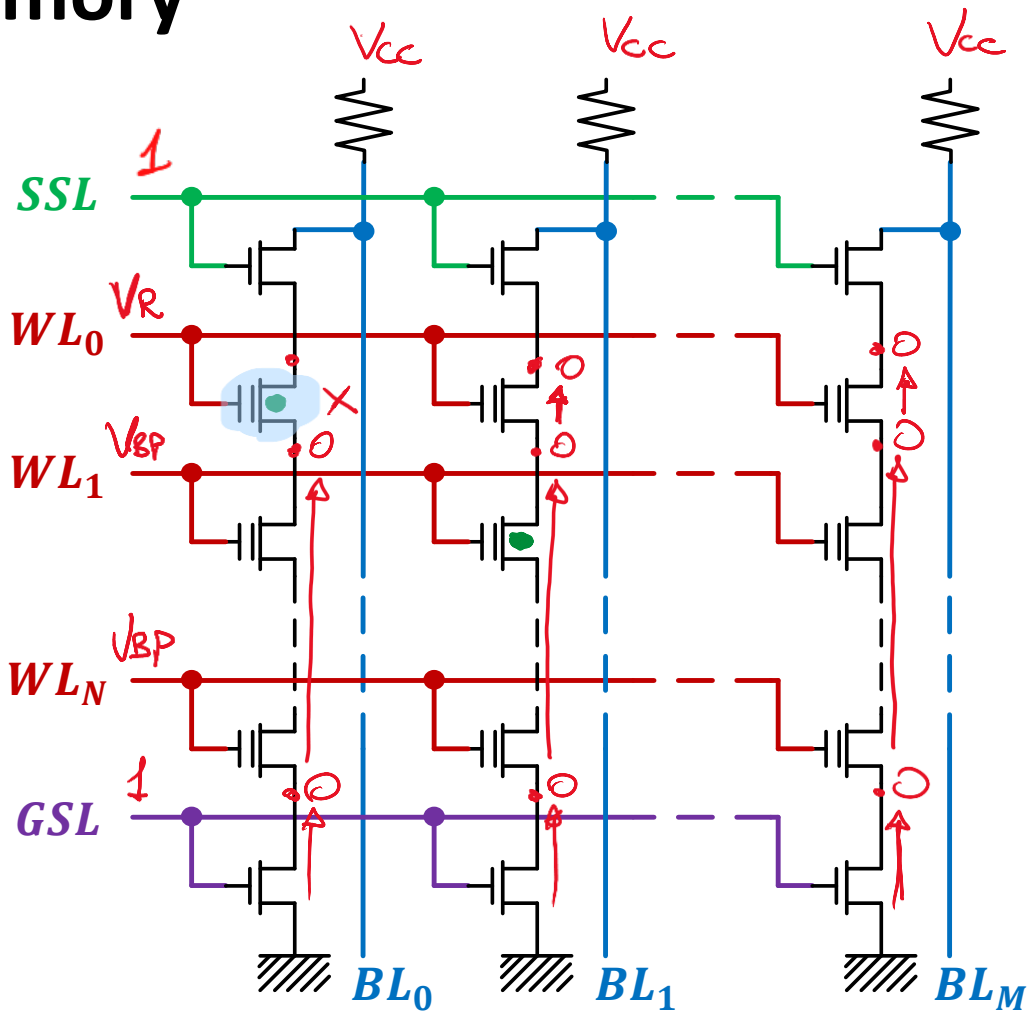
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (0,0)



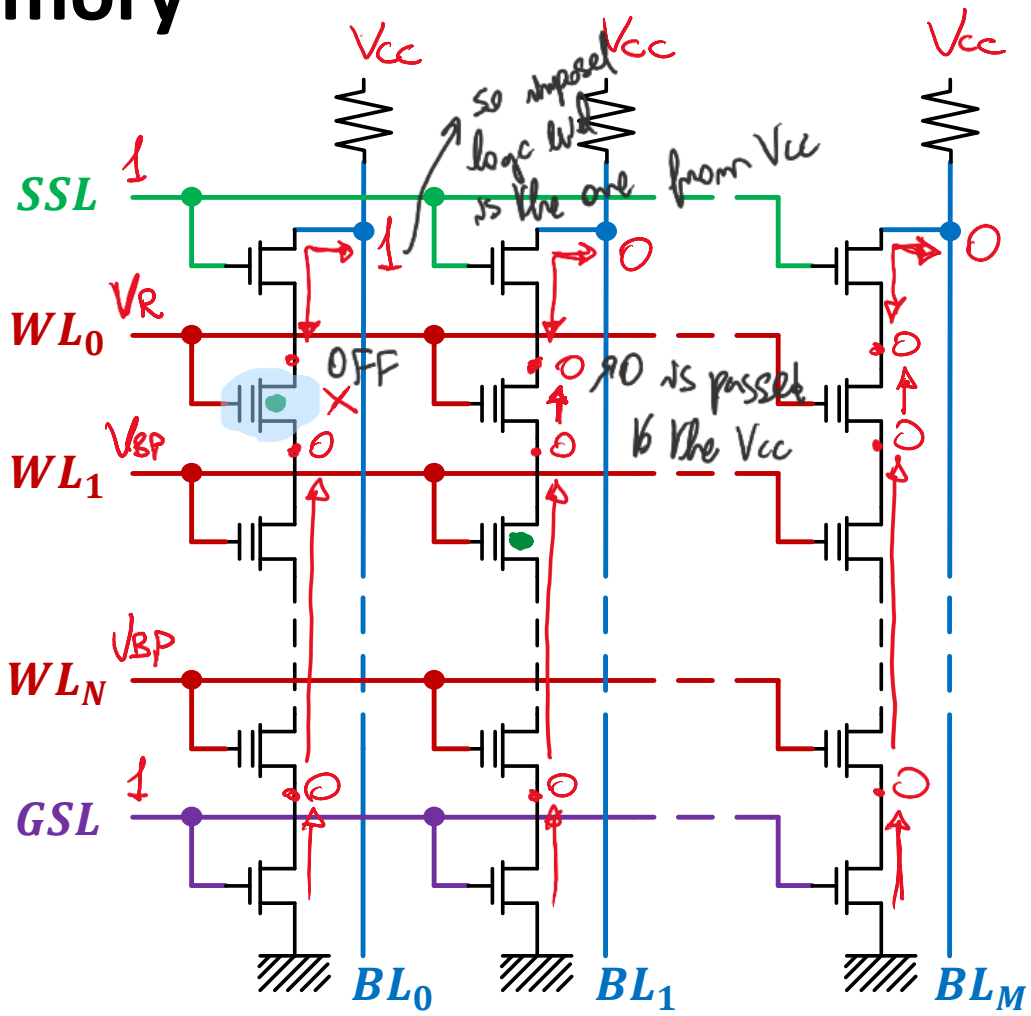
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs bypass voltage
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (0,0)



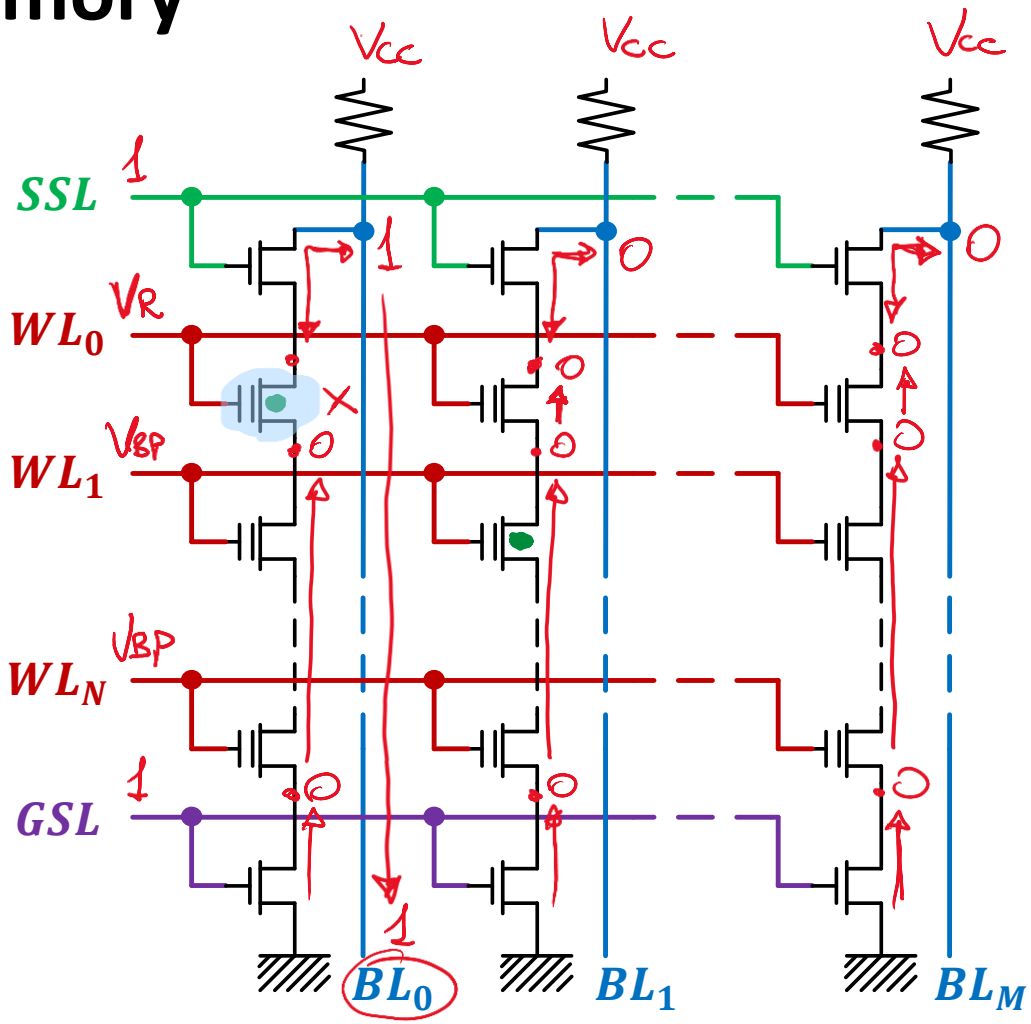
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (0,0)



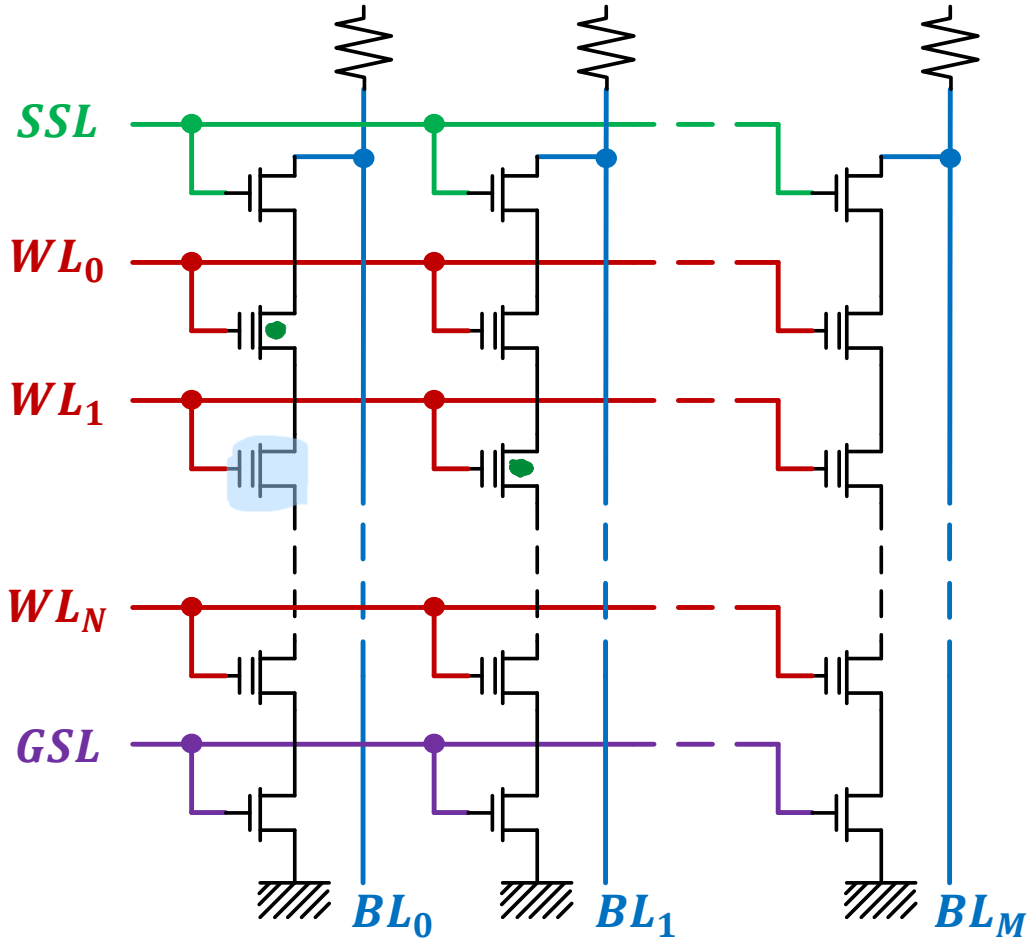
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (0,0)



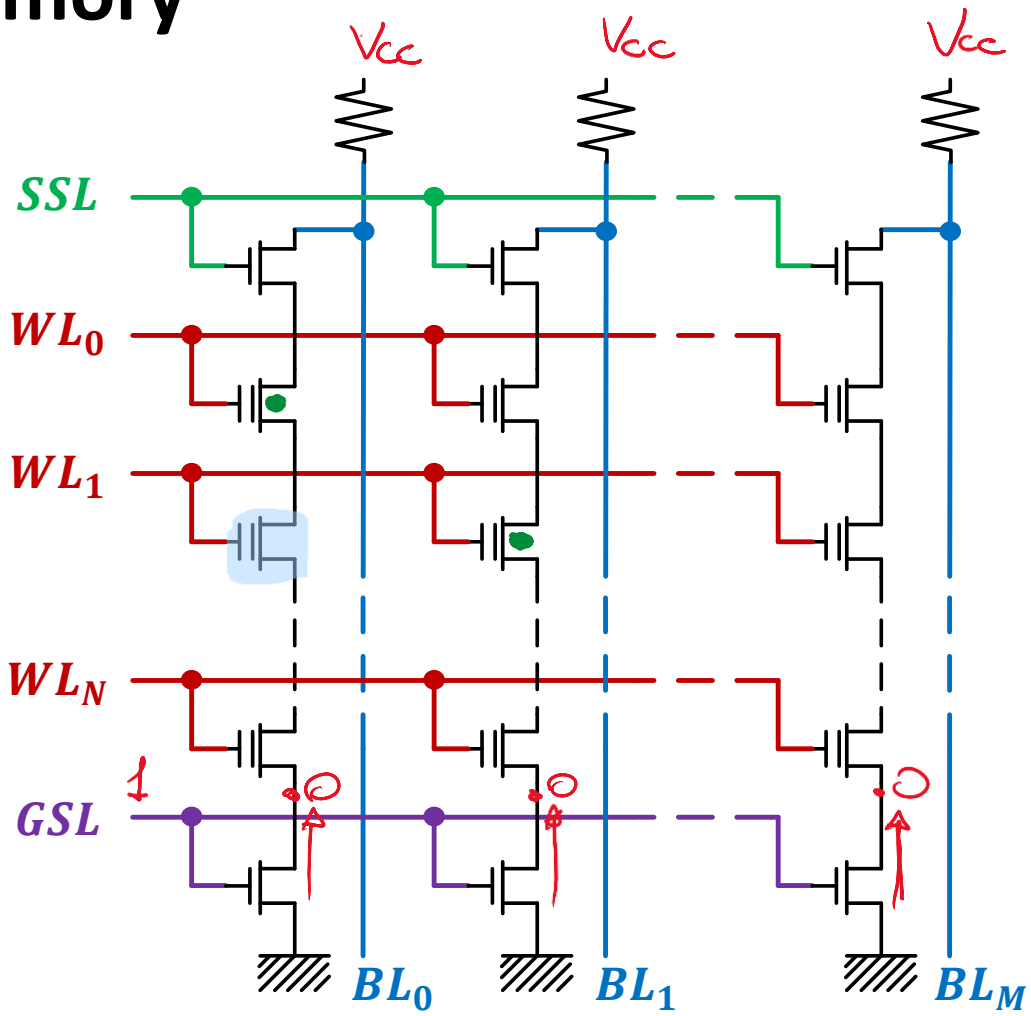
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (1,0)



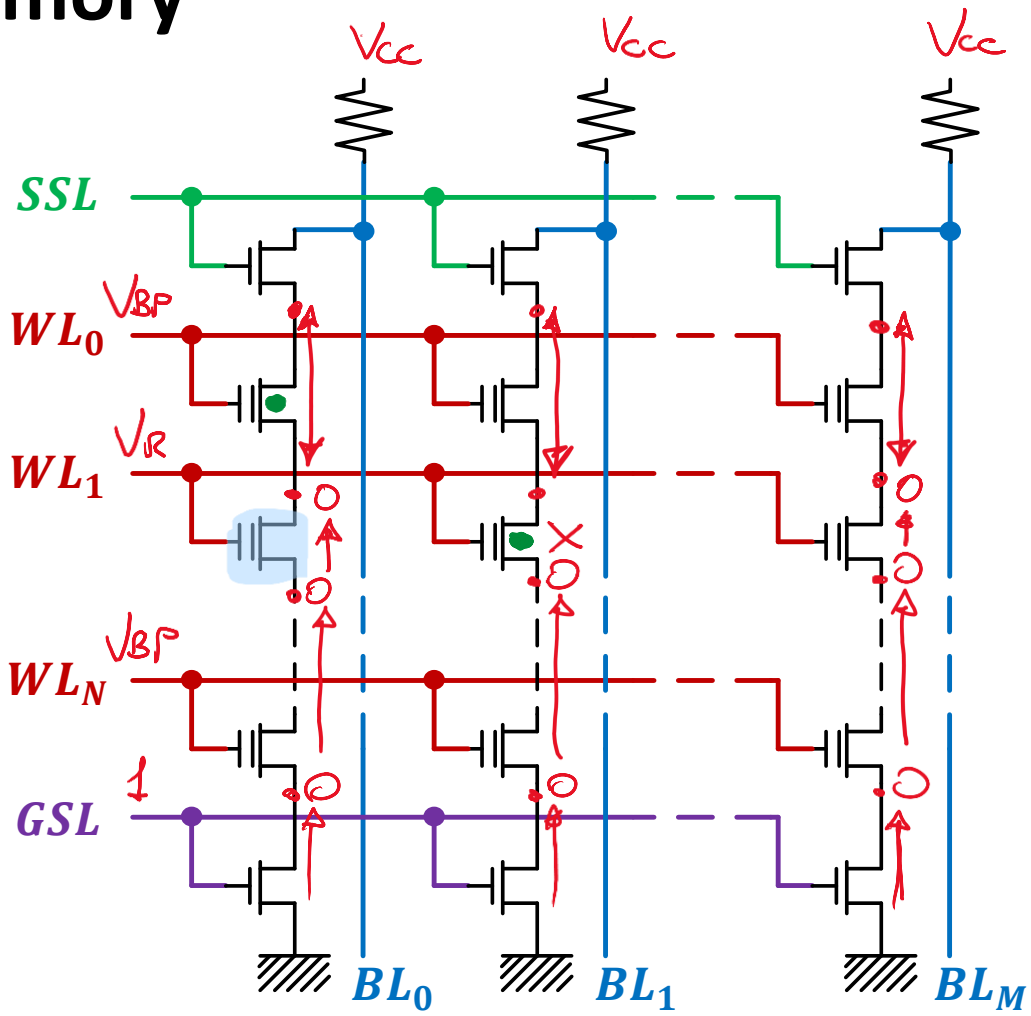
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (1,0)



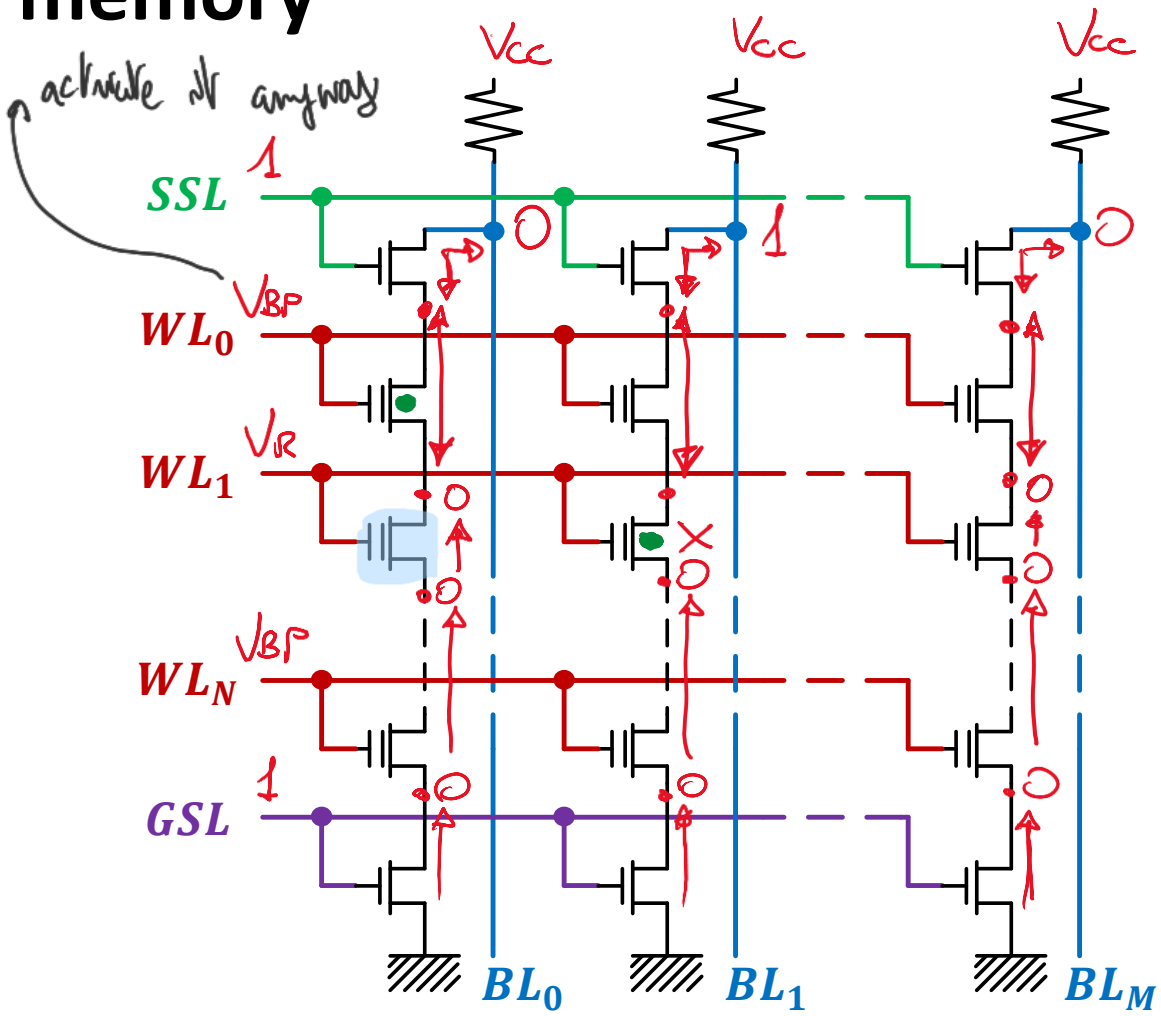
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (1,0)



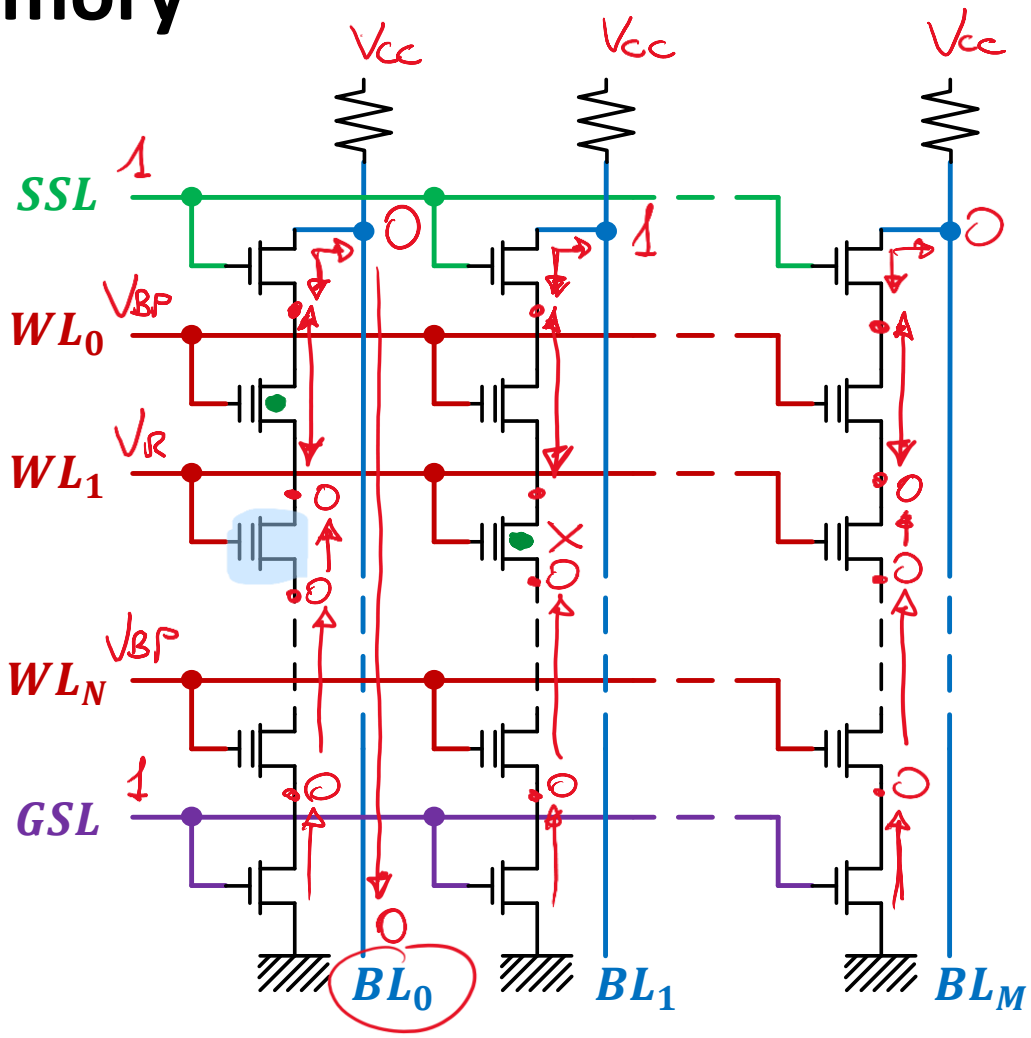
(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (1,0)



(NAND) Flash memory

- Working principle
  - Read by page – Example
    - $GSL = 1 \qquad (V_{CC})$
    - $SSL = 1 \qquad (V_{CC})$
    - V_R on WL of cells/transistors to be read
      - V_{BP} on all other WLs
    - V_{CC} on all BLs
    - Assume cells (0,0) and (1,1) are programmed
    - All other cells are not programmed
    - Let's assume to read cell (1,0)



(NAND) Flash memory – Final remarks

- More than one block in a memory
  - Not just one!

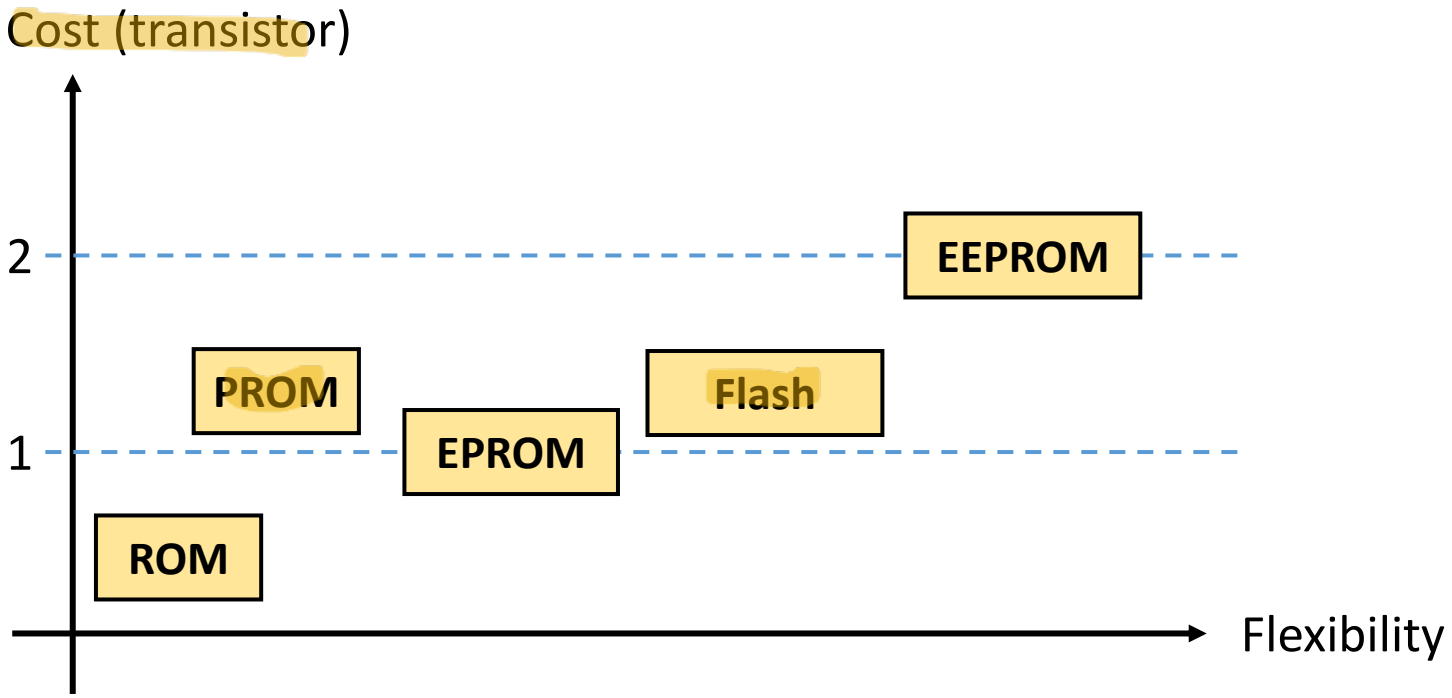
to overwrite, first erase then reprogram.

- Data overwriting is not possible

- For each data write (in a block)
  - Erasing (the whole block)
  - Programming the block by page (one word line at a time)

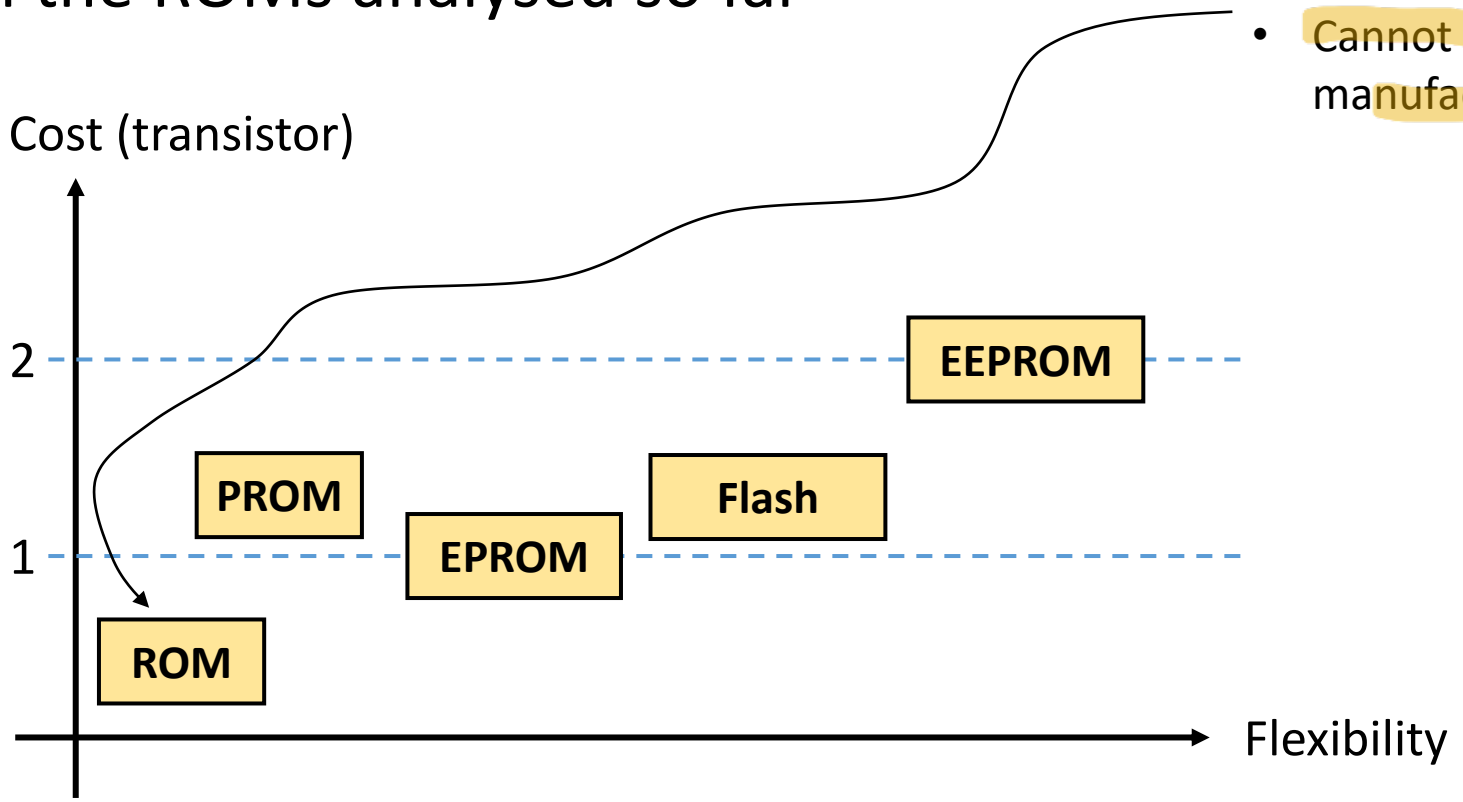
ROMs summary and comparison

- Comparing all the ROMs analysed so far



ROMs summary and comparison

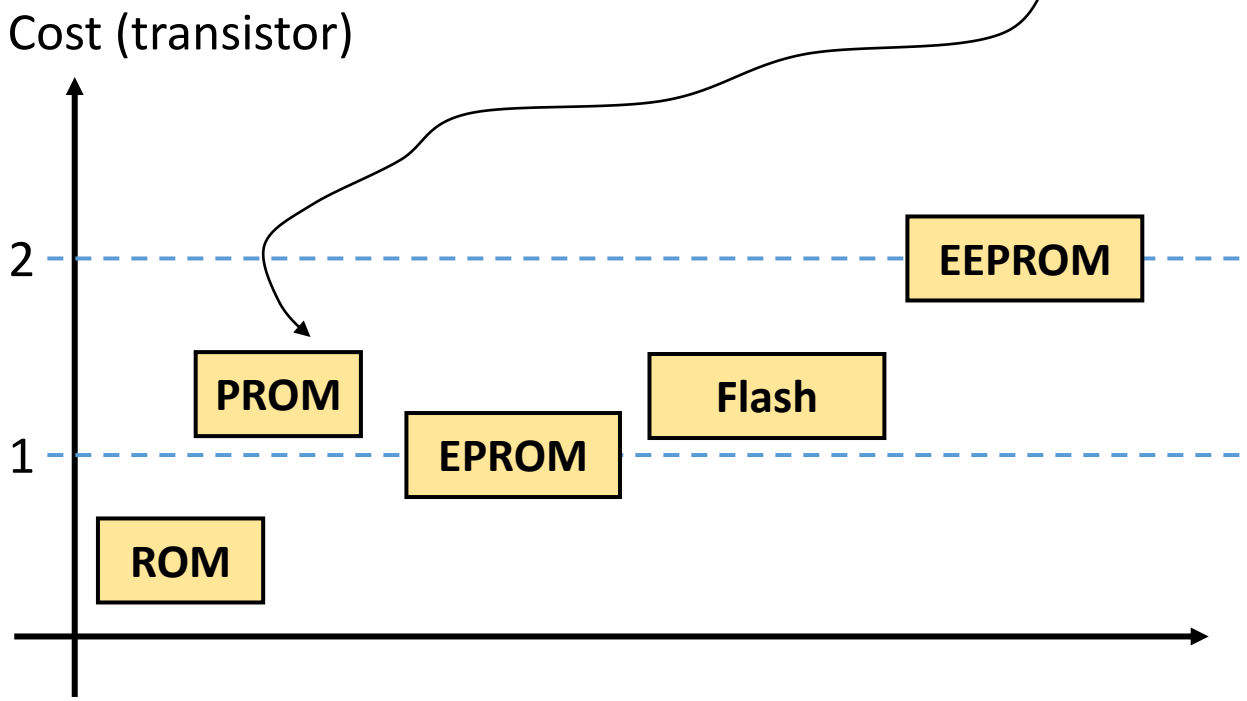
- Comparing all the ROMs analysed so far



- Transistor not in all cells
- Cannot be modified after manufacturing

ROMs summary and comparison

- Comparing all the ROMs analysed so far

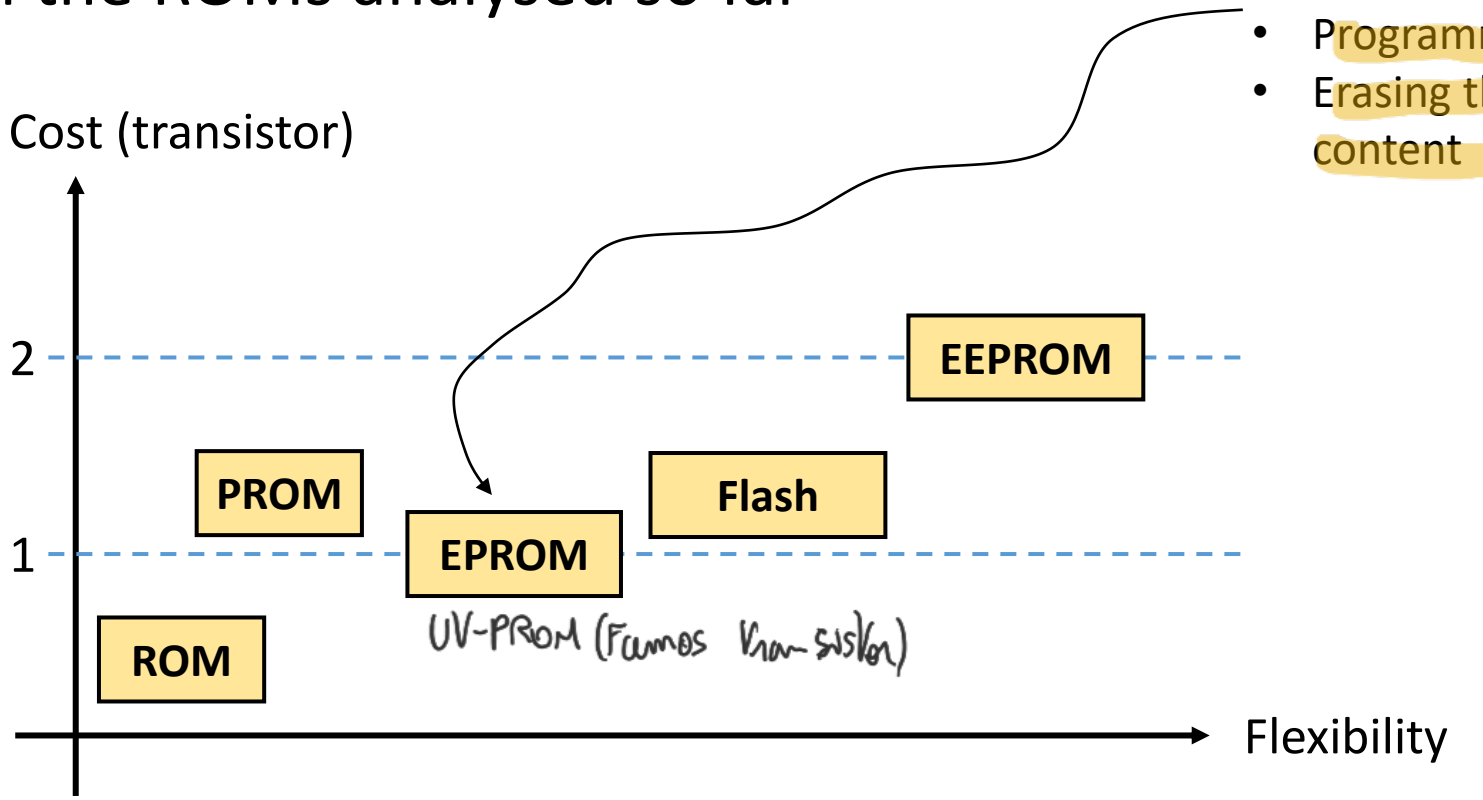


- Each cell: 1 transistor + 1 fuse
- Can be modified just one time after manufacturing

(or antifuse)
a bit more flexible

ROMs summary and comparison

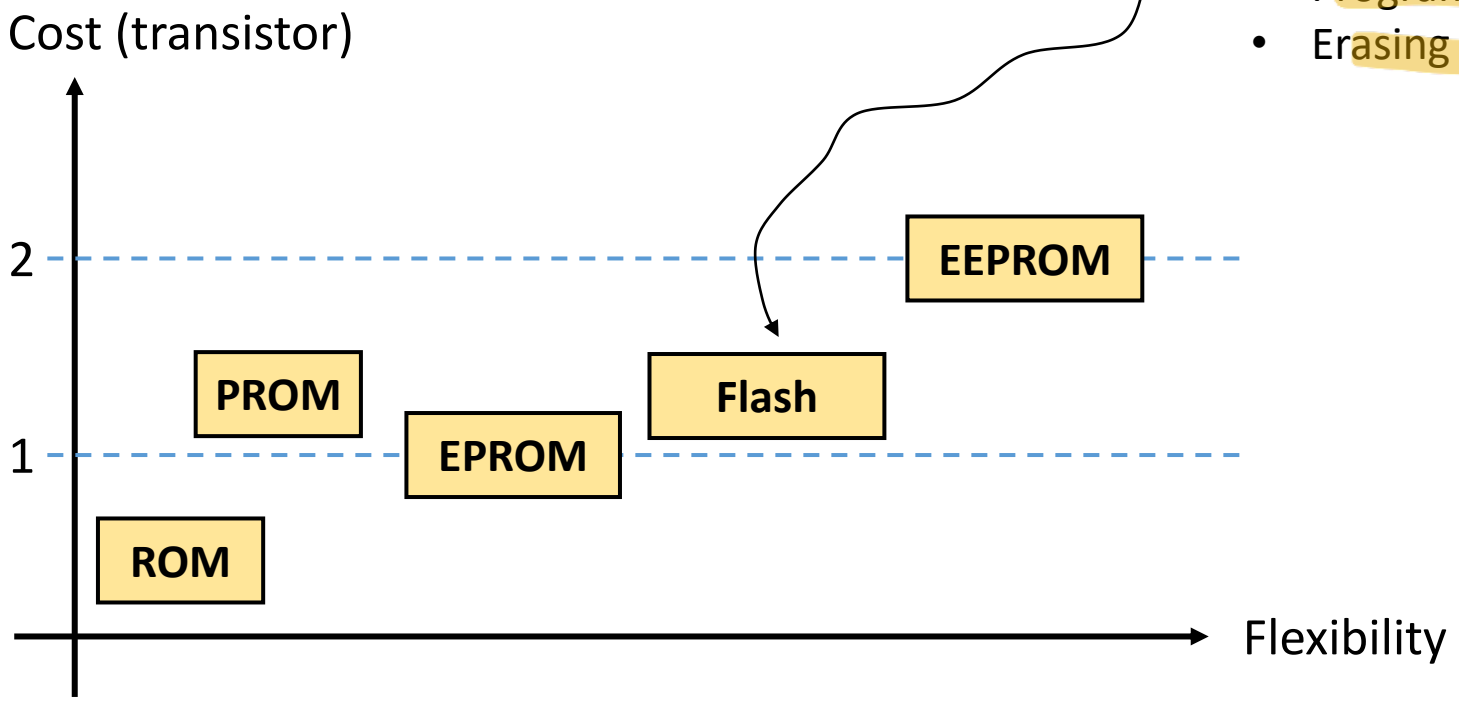
- Comparing all the ROMs analysed so far



- Each cell: 1 transistor
- Programming by cell
- Erasing the whole memory content

ROMs summary and comparison

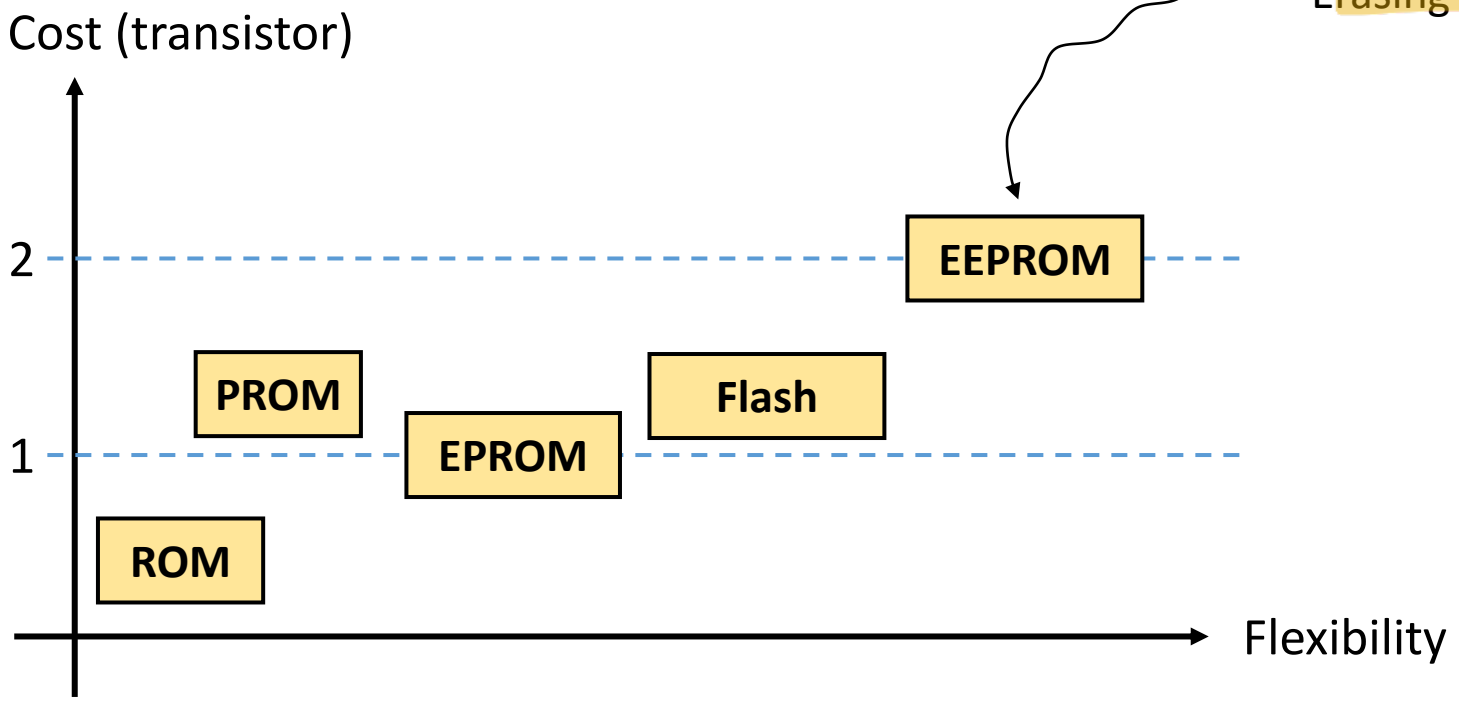
- Comparing all the ROMs analysed so far



- Each cell: 1 transistor
  - And pass transistor on strings
- Programming by cell
- Erasing by block

ROMs summary and comparison

- Comparing all the ROMs analysed so far



- Each cell: 2 transistors
- Programming by cell
- Erasing by cell

Thank you for your attention

Luca Crocetti
(luca.crocetti@unipi.it)