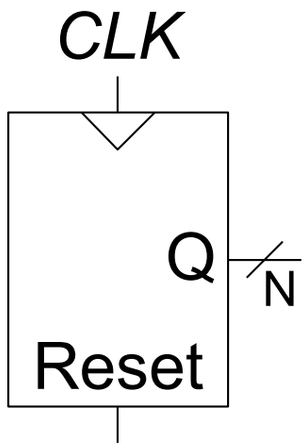
Counters, Shift Registers, Memory Arrays

Digital Computer Design

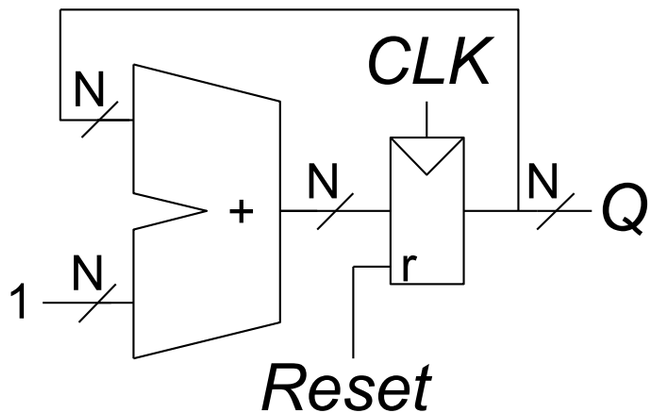
Counters

- Increments on each clock edge
- Used to cycle through numbers. For example,
  - 000, 001, 010, 011, 100, 101, 110, 111, 000, 001…
- Example uses:
  - Digital clock displays

Symbol



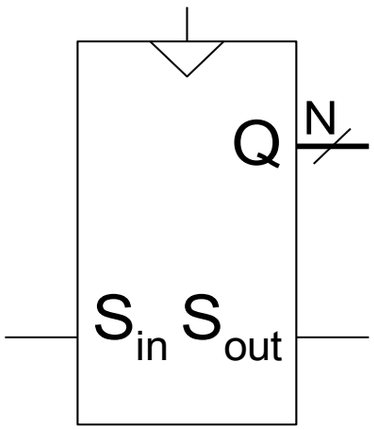
Implementation



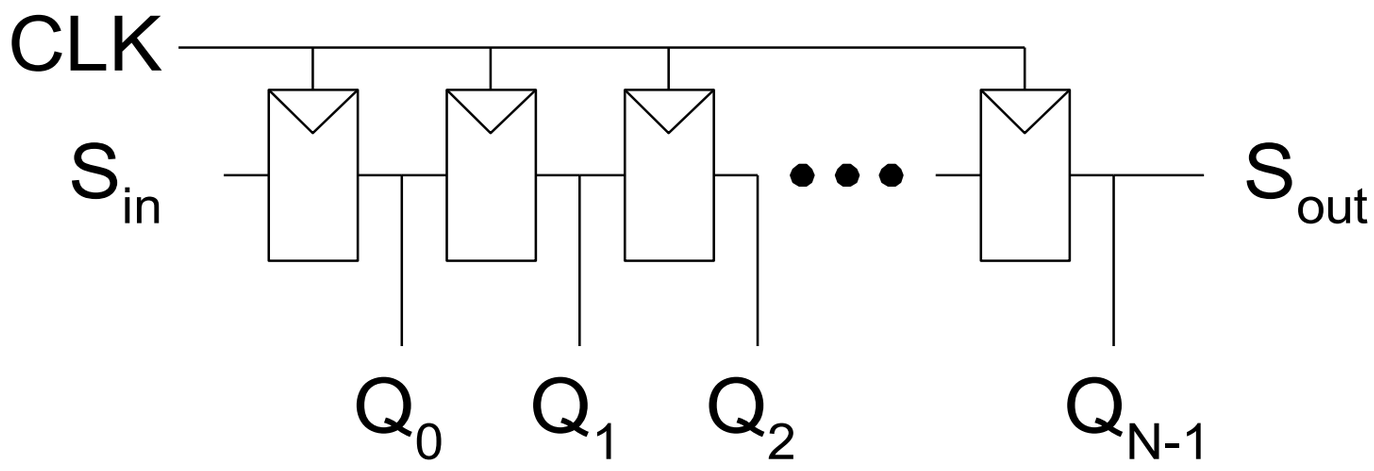
Shift Registers

- Shift a new bit in on each clock edge
- Shift a bit out on each clock edge
- *Serial-to-parallel converter*: converts serial input (S_{in}) to parallel output ($Q_{0:N-1}$)

Symbol:

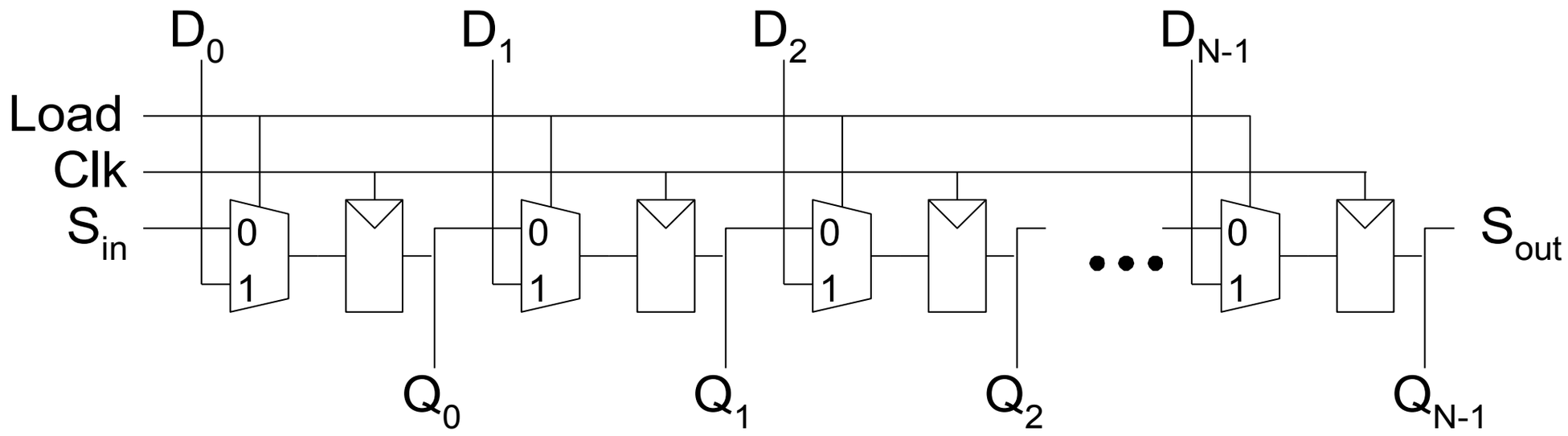


Implementation:



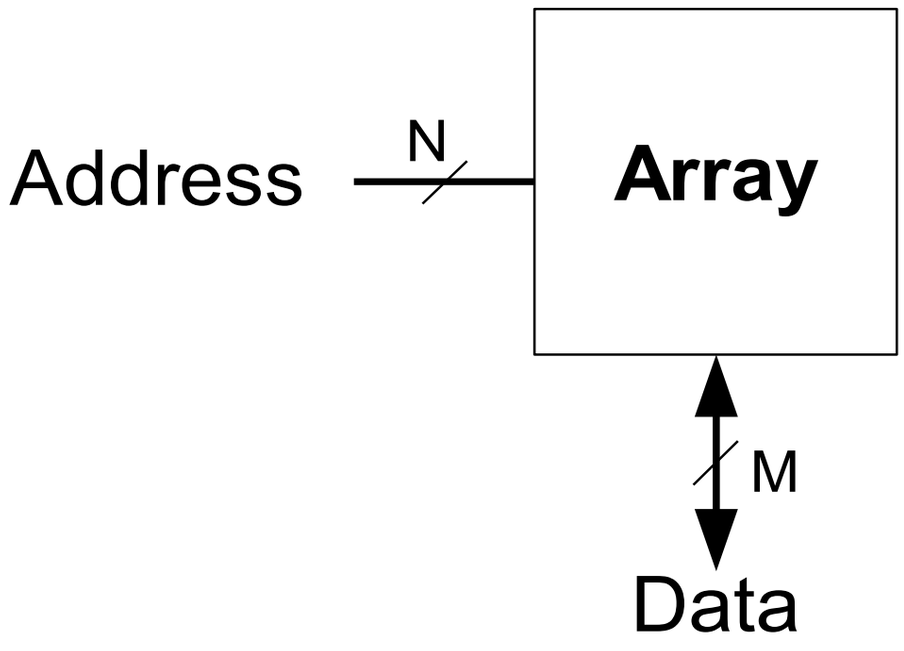
Shift Register with Parallel Load

- When *Load* = 1, acts as a normal *N*-bit register
- When *Load* = 0, acts as a shift register
- Now can act as a *serial-to-parallel converter* (S_{in} to $Q_{0:N-1}$) or a *parallel-to-serial converter* ($D_{0:N-1}$ to S_{out})



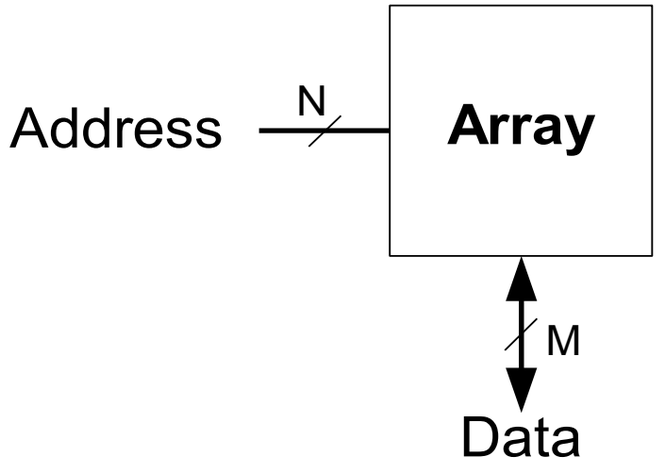
Memory Arrays

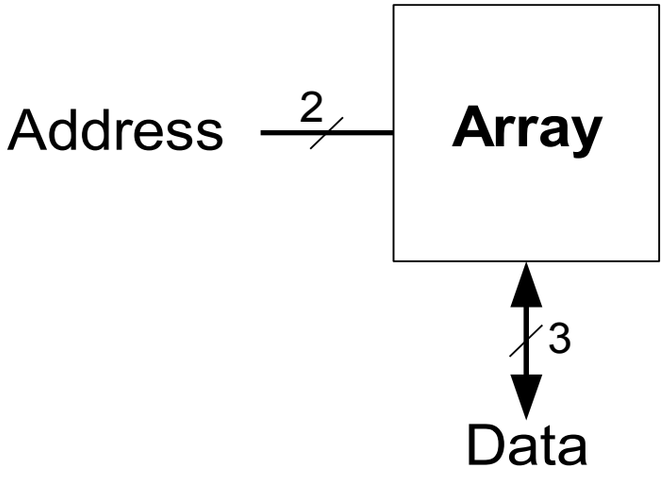
- Efficiently store large amounts of data
- 3 common types:
  - Dynamic random access memory (DRAM)
  - Static random access memory (SRAM)
  - Read only memory (ROM)
- *M*-bit data value **read/written** at each unique *N*-bit address

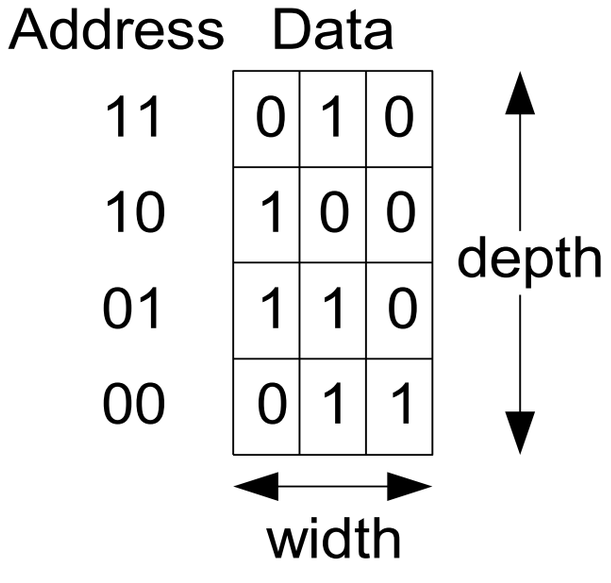


Memory Arrays

- 2-dimensional array of bit cells
- Each bit cell stores one bit
- *N* **address bits** and *M* **data bits**:
  - 2^N rows and *M* columns
  - **Depth:** number of rows (number of **words**)
  - **Width:** number of columns (size of **word**)
  - **Array size:** depth × width = $2^N \times M$

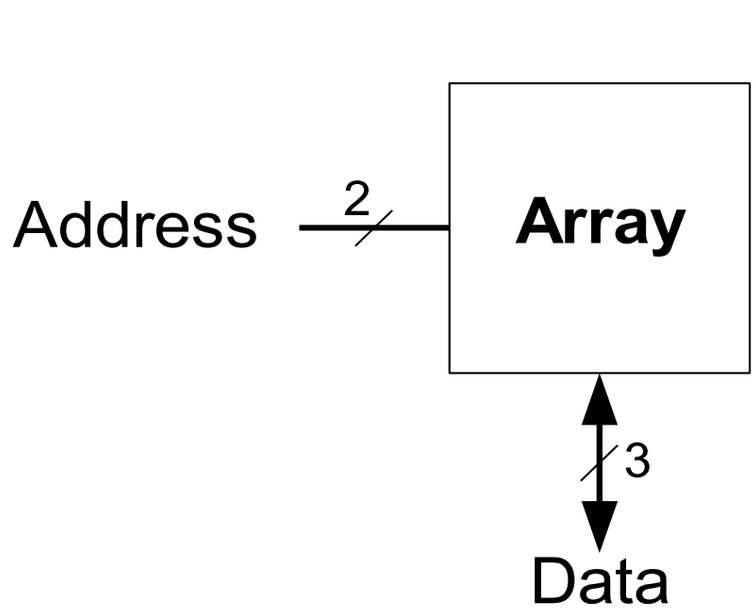


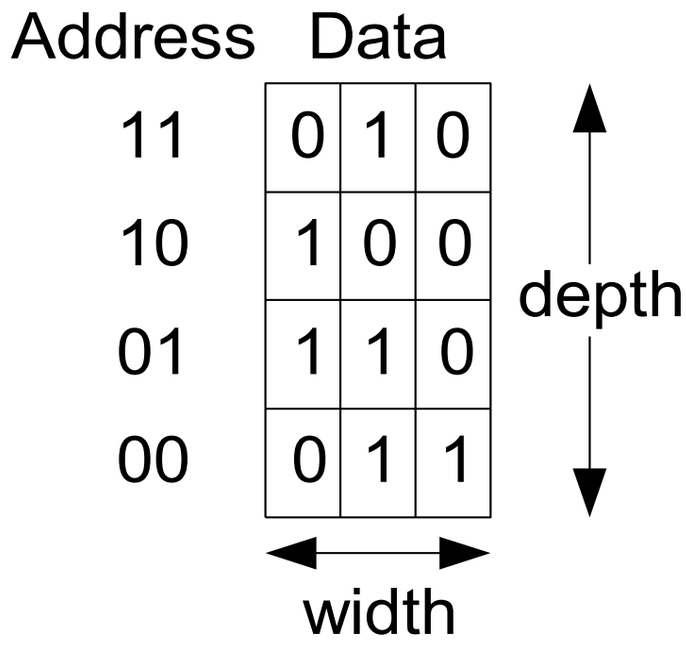




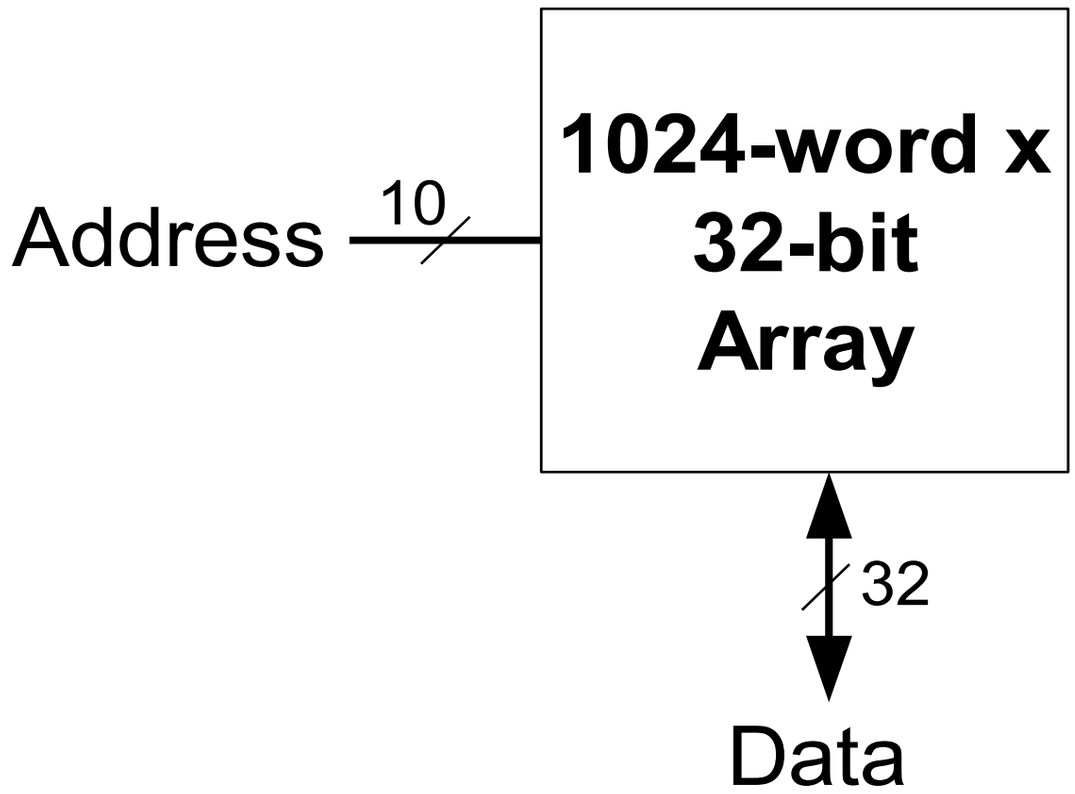
Memory Array Example

- 2^2 × 3-bit array
- Number of words: 4
- Word size: 3-bits
- For example, the 3-bit word stored at address 10 is 100





Memory Arrays



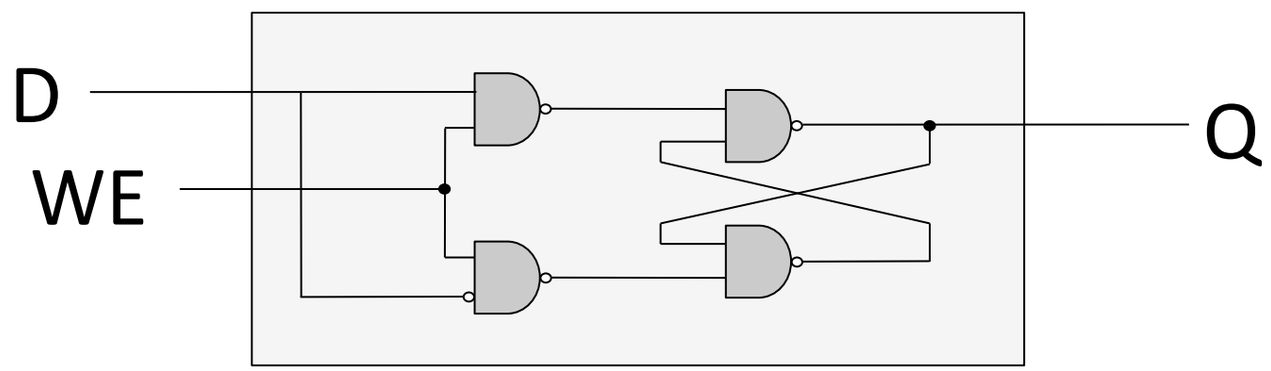
How many words? Word size?

Addressing Memory

Let's implement a simple memory array with:

- 3-bit addressability **&** address space size of 2 (total of 6 bits)

1 Bit



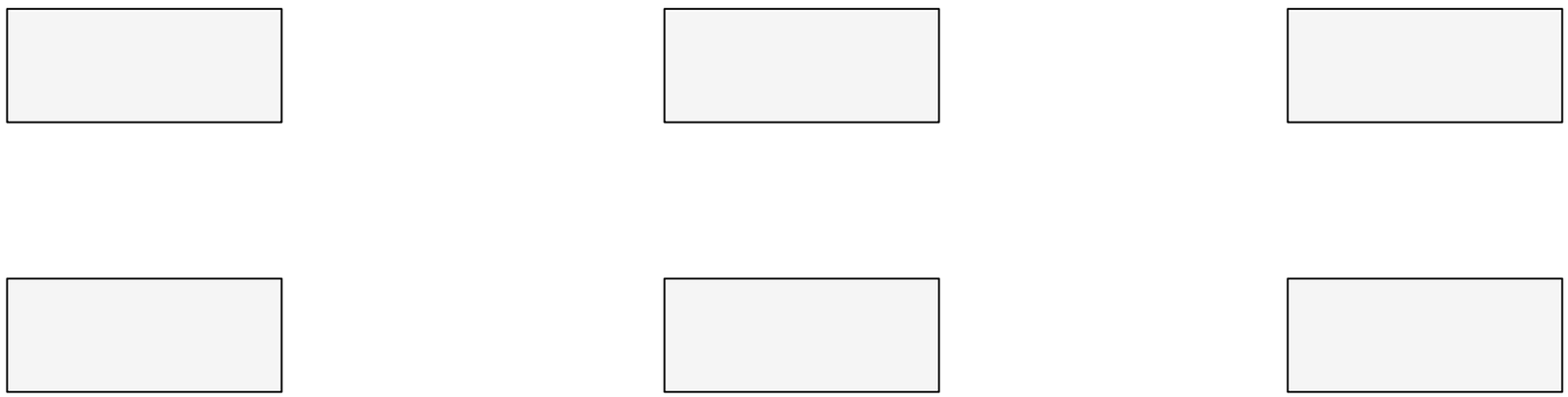
6-Bit Memory Array

Addr(0)	Bit_2	Bit_1	Bit_0
Addr(1)	Bit_2	Bit_1	Bit_0

Reading from Memory

How can we select the address to read?

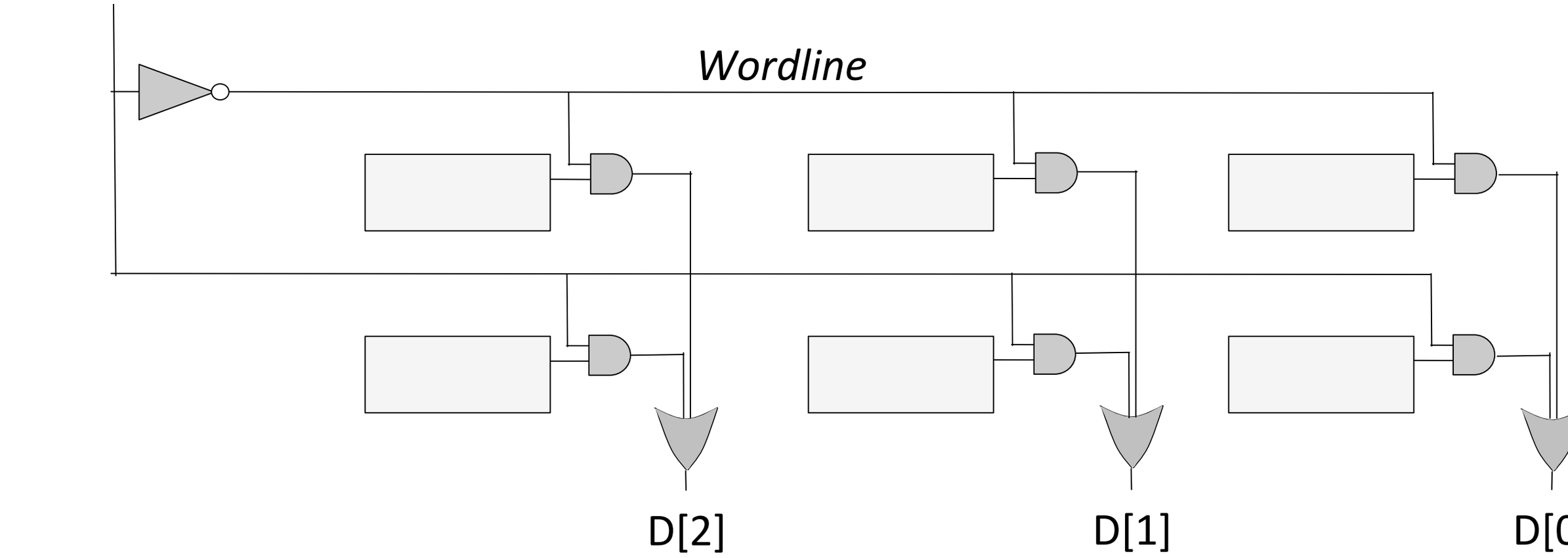
- Because there are 2 addresses, address size is log(2)=1 bit



Reading from Memory

How can we select the address to read?

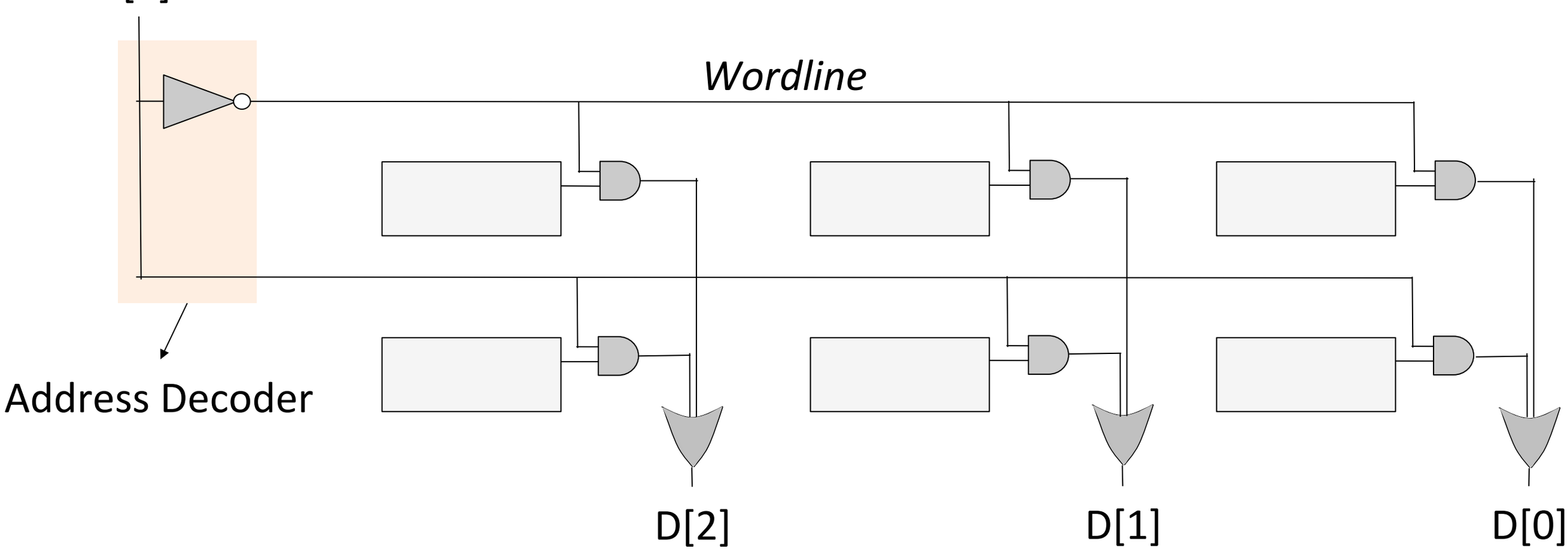
- Because there are 2 addresses, address size is log(2)=1 bit



Reading from Memory

How can we select the address to read?

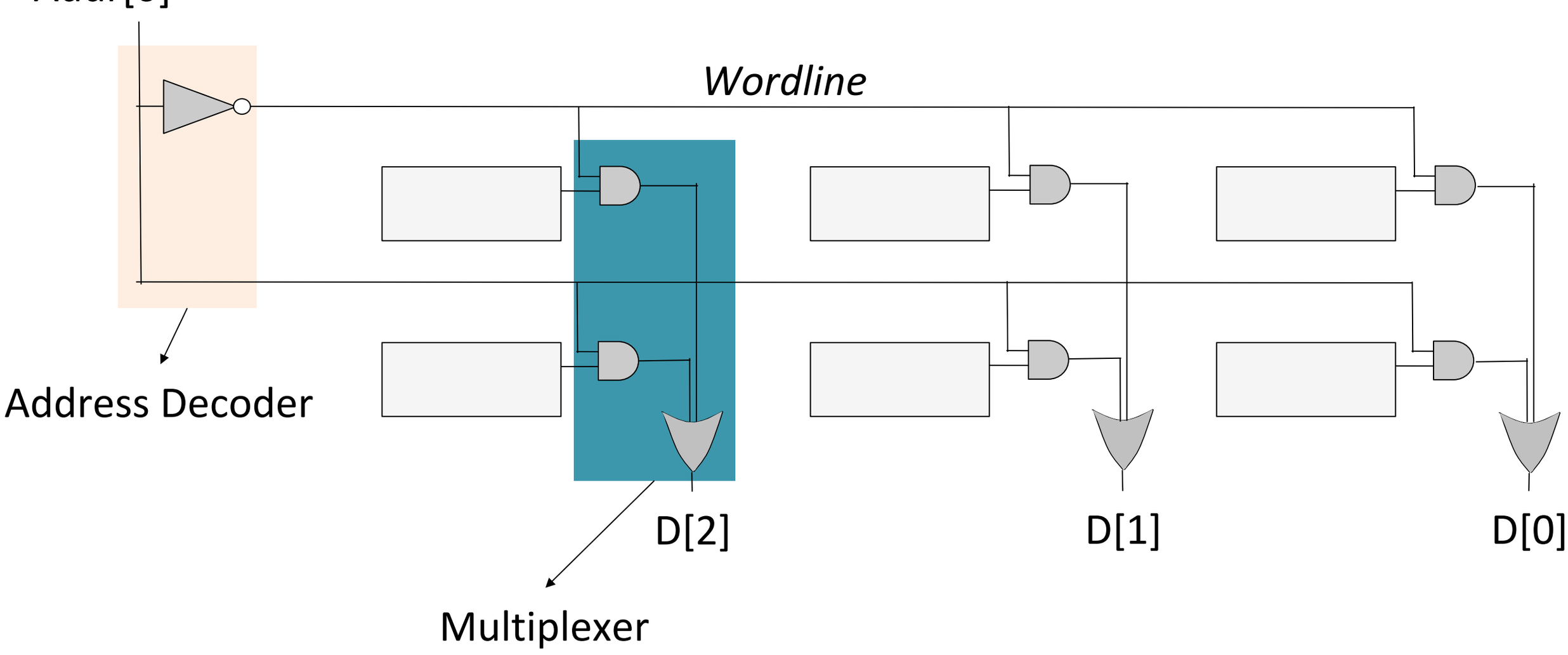
- Because there are 2 addresses, address size is log(2)=1 bit



Reading from Memory

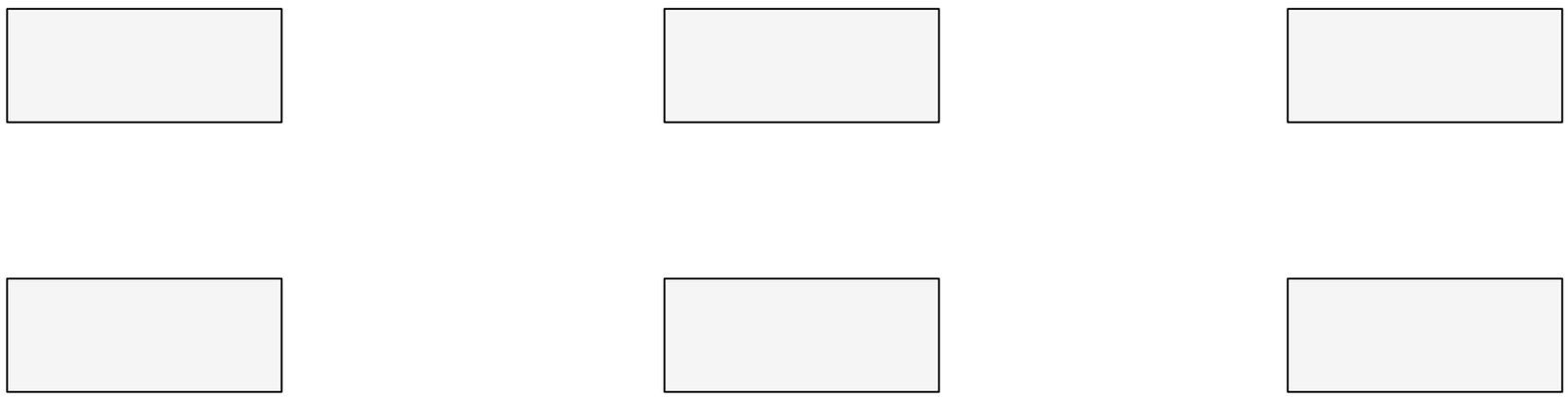
How can we select the address to read?

- Because there are 2 addresses, address size is log(2)=1 bit



Writing to Memory

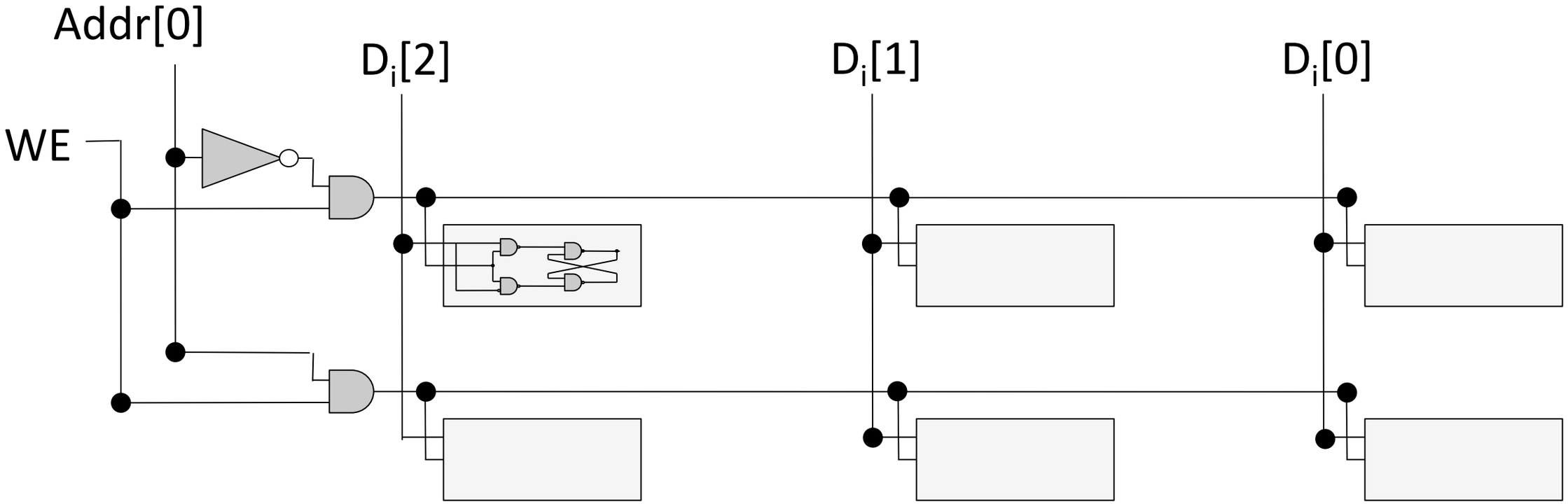
How can we select the address and write to it?



Writing to Memory

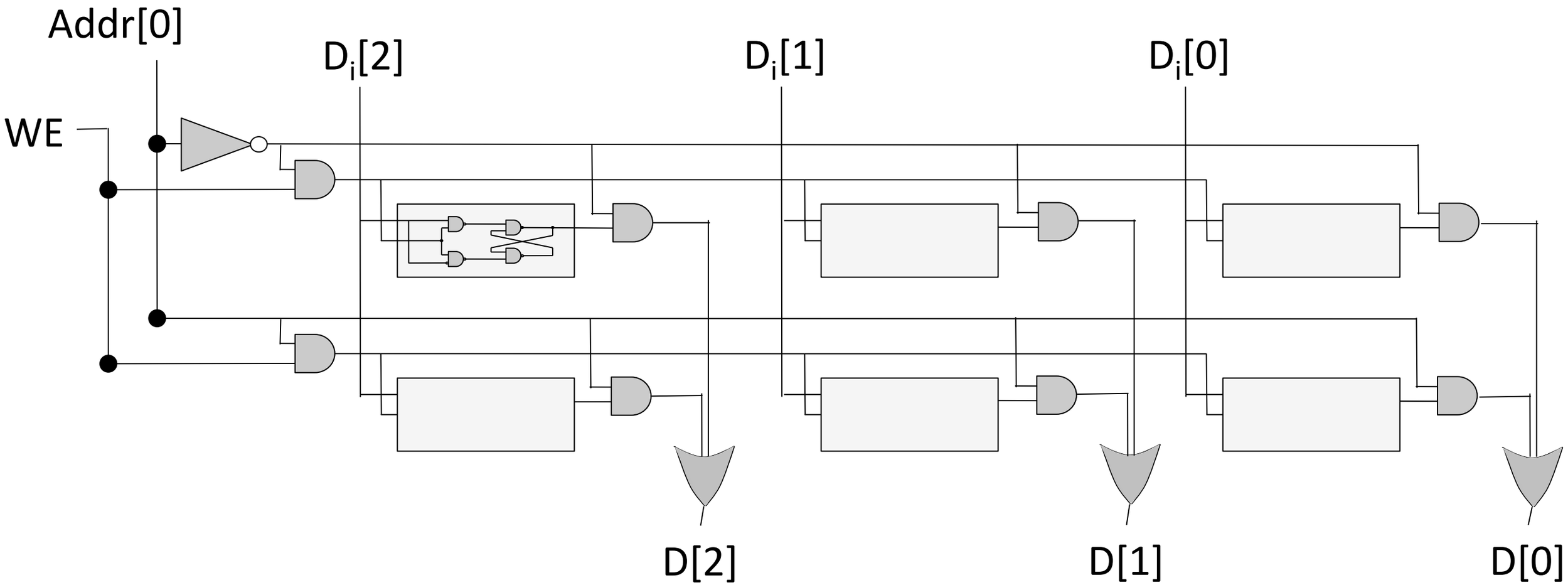
How can we select the address and write to it?

- Input is indicated with D_i

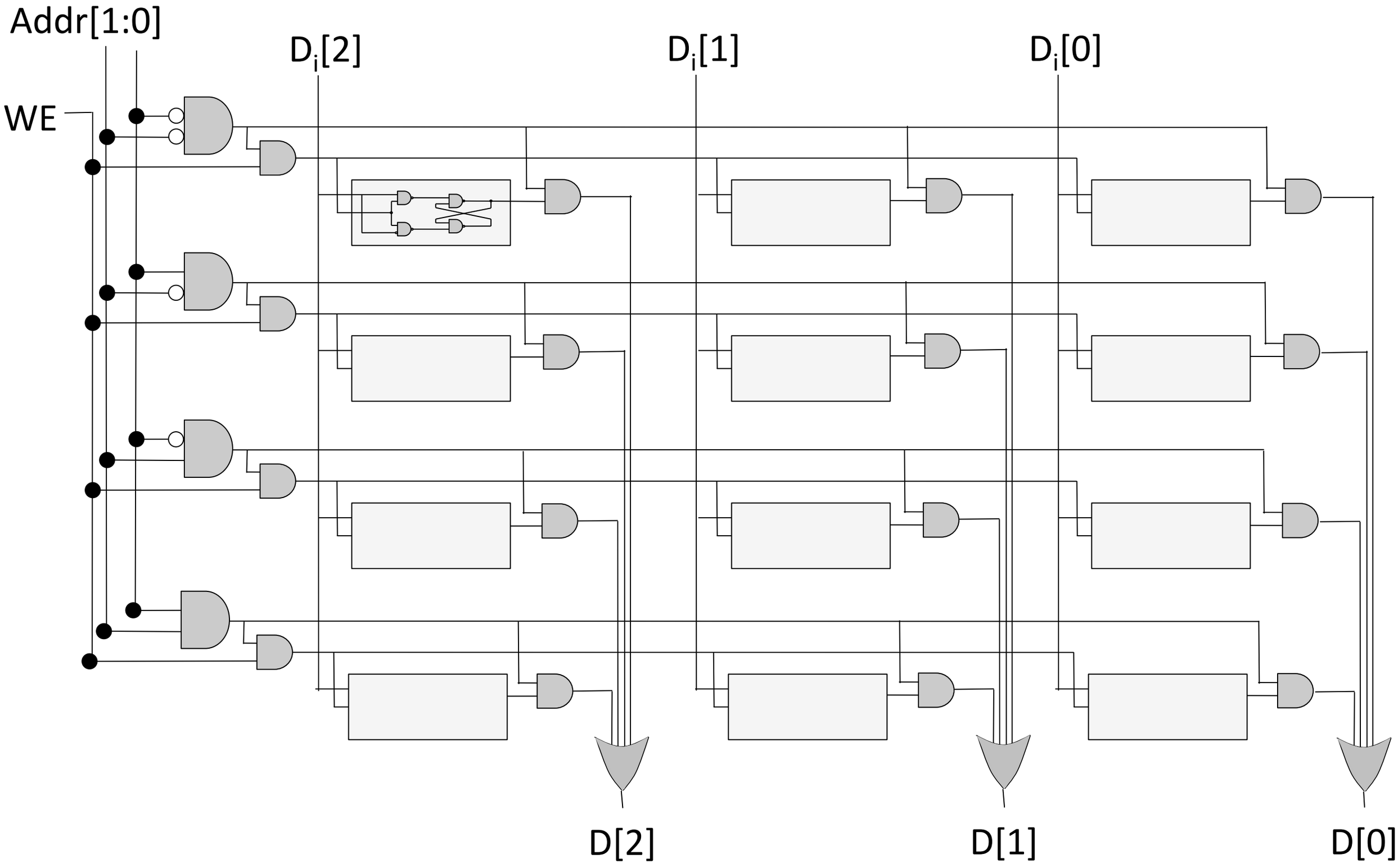


Putting it all Together

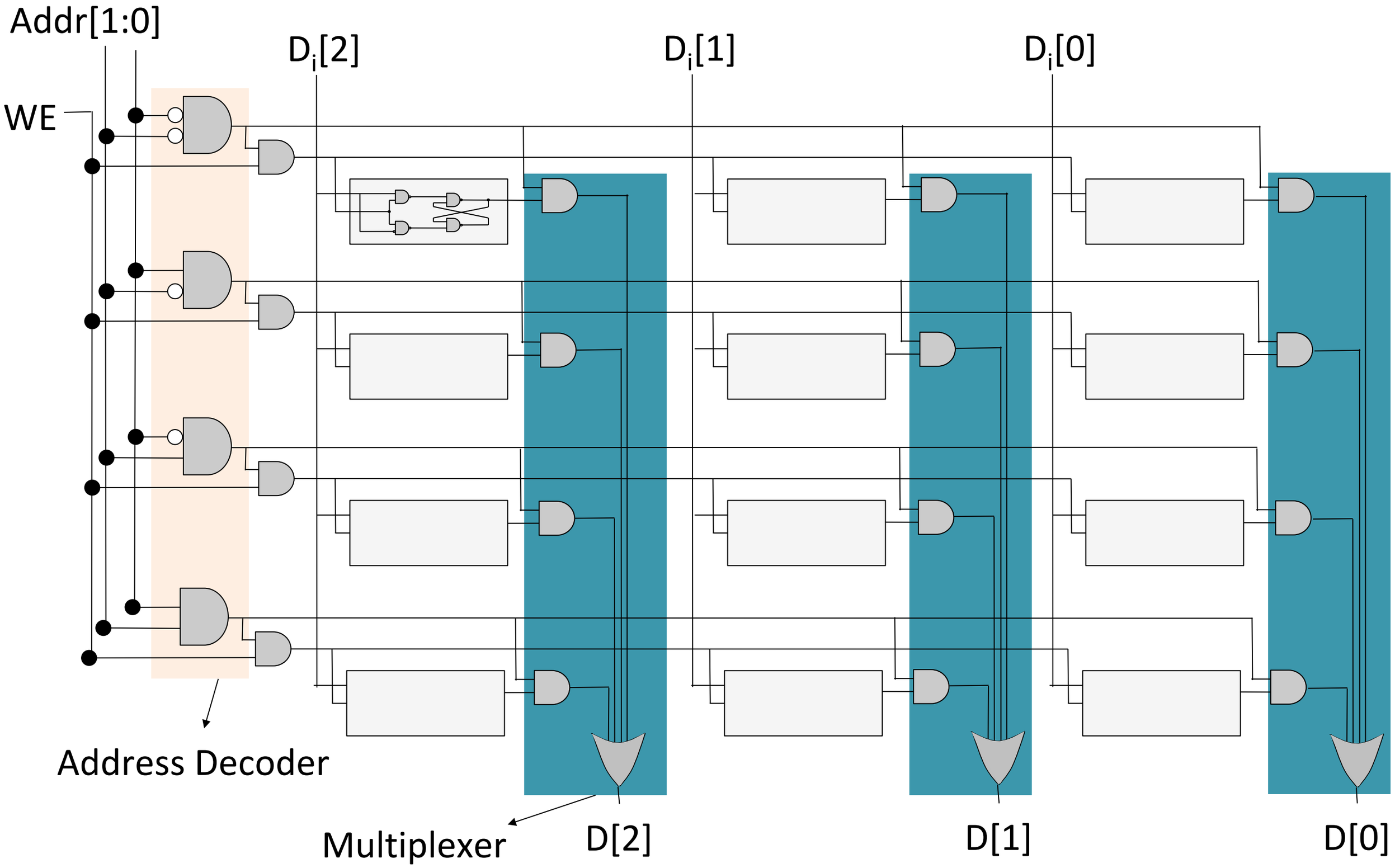
Enable reading and writing to a memory array



A Bigger Memory Array

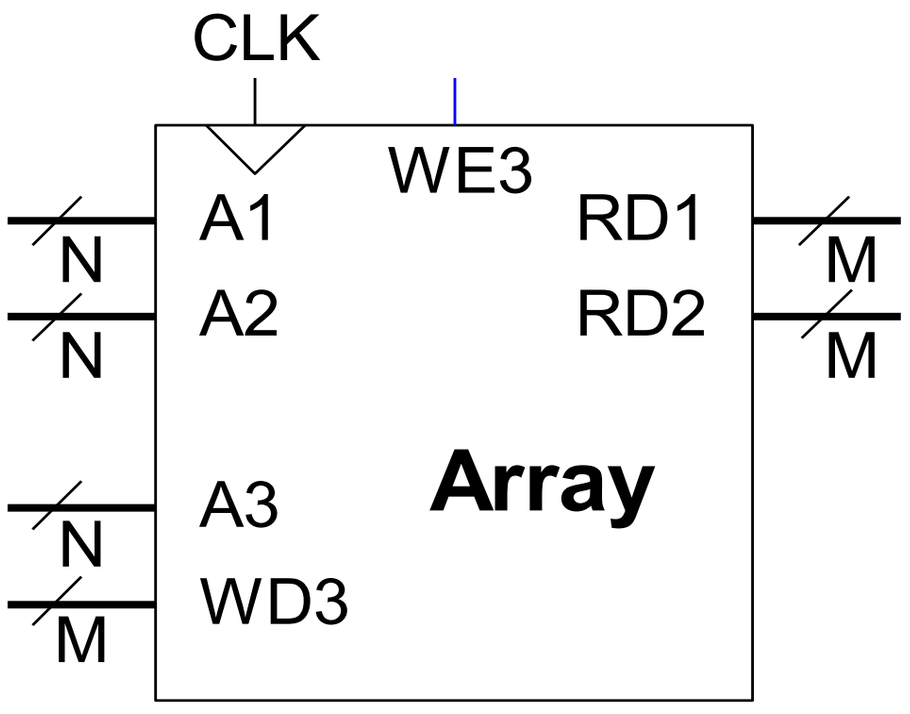


A Bigger Memory Array



Multi-ported Memories

- Multiported memories can access several addresses **simultaneously**
- Example: 3-ported memory
  - 2 read ports (A1/RD1, A2/RD2)
  - 1 write port (A3/WD3, WE3 enables writing)
- **Register file:** small multi-ported memory



Types of Memory

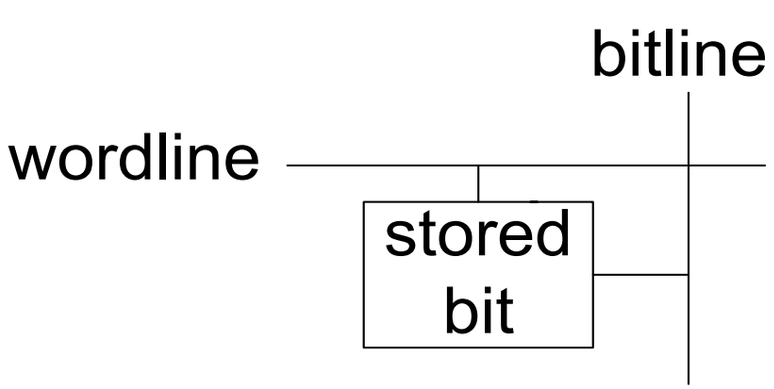
- Memories are classified based on how they store bits in the bit cell.
  - Random access memory (RAM):
    - **volatile:** loses its data when power off
    - called random access memory because any data word is accessed with the same delay as any other.
    - e.g. main memory in computers
  - Read only memory (ROM):
    - **nonvolatile:** retains data when power off
    - e.g. flash memory in cameras

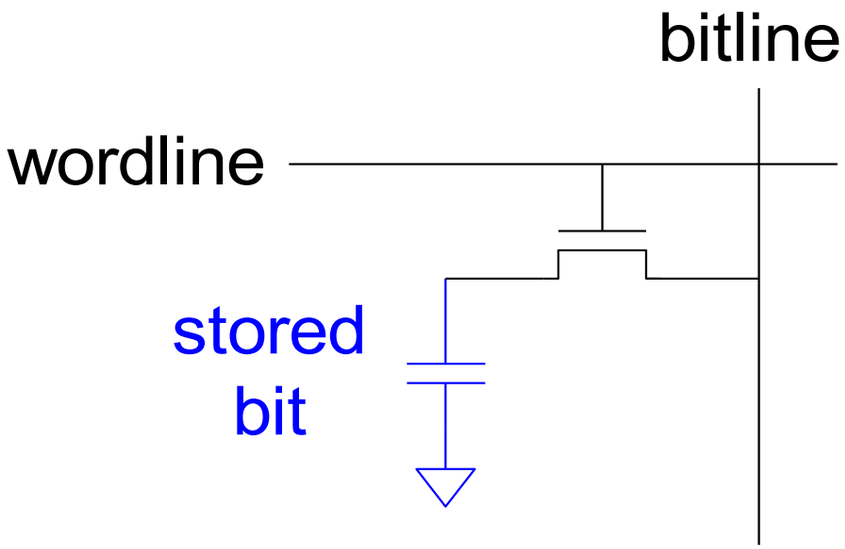
Types of RAM

- **DRAM** (Dynamic random access memory)
- **SRAM** (Static random access memory)
- Differ in how they store data:
  - DRAM uses a **capacitor**
  - SRAM uses **cross-coupled inverters**

Dynamic Random Access Memory (DRAM)

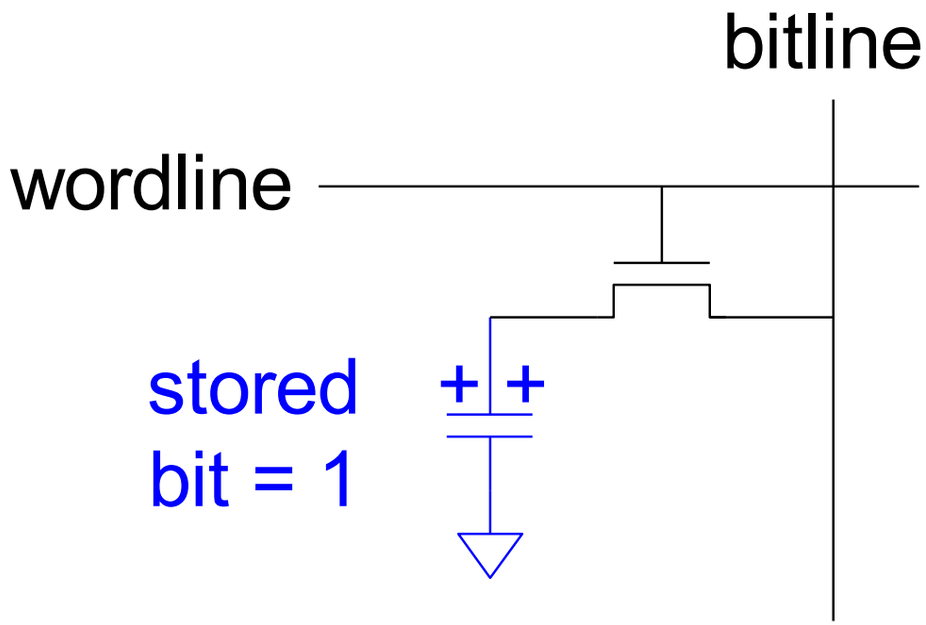
- Data bits stored on capacitor
  - When the capacitor is charged to V_{DD} the stored bit is 1
  - When it is discharged to GND the stored bit is 0
- The transistor behaves as a **switch**
  - either connects or disconnects the capacitor from the bitline.

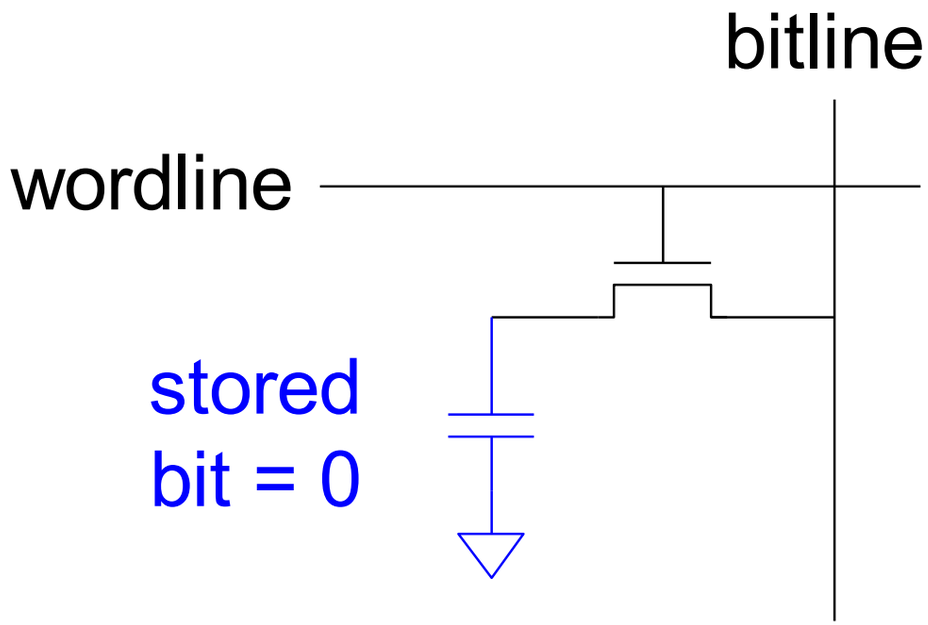




Dynamic Random Access Memory (DRAM)

- Dynamic because the value needs to be refreshed (rewritten) periodically and after read:
  - Charge leakage from the capacitor degrades the value
  - Reading destroys the stored value
    - Data word must be restored (rewritten) after each read.

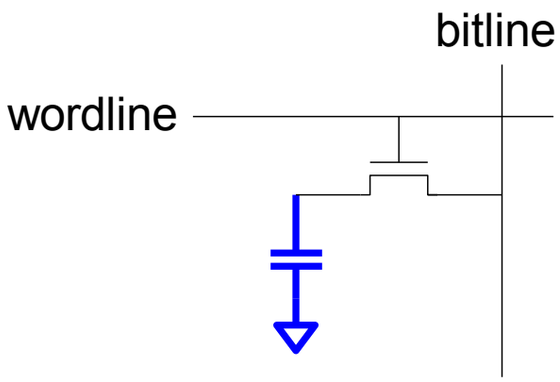




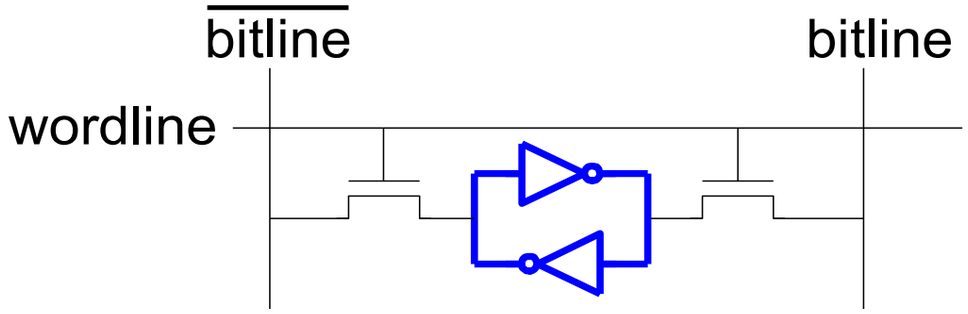
Static Random Access Memory (SRAM)

- Static because stored bits do not need to be refreshed.
  - Unlike DRAM, if noise degrades the value of the stored bit, the cross-coupled inverters restore the value.

DRAM bit cell:



SRAM bit cell:



Area and Delay

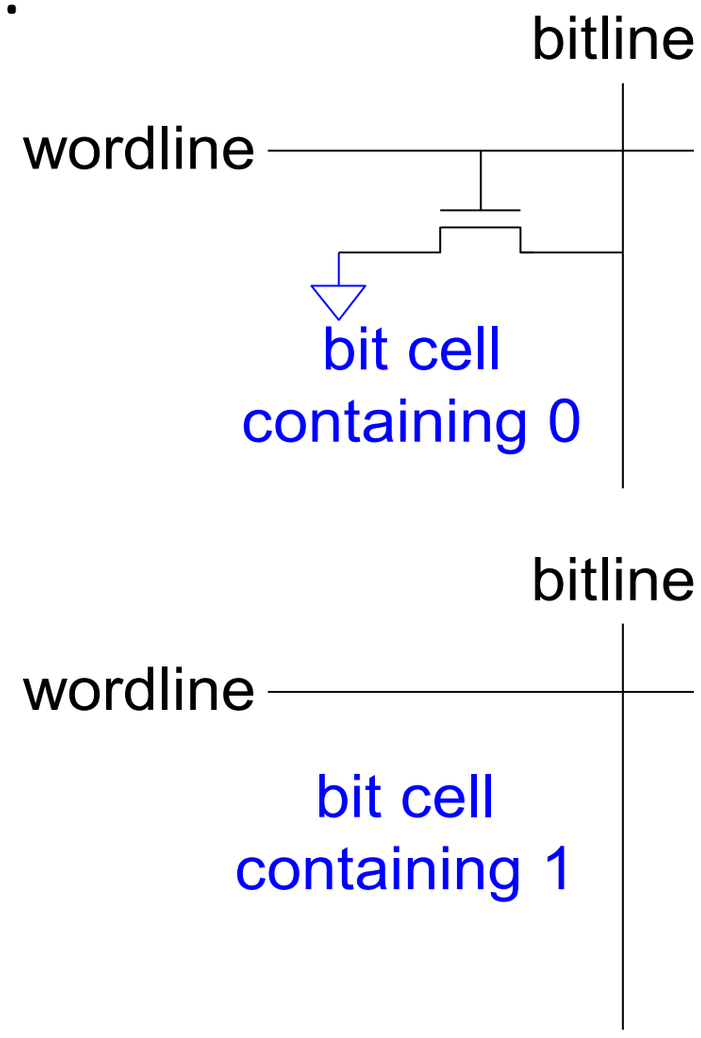
- The data bit stored in a flip-flop is available immediately at its output.
  - But flip-flops take at least 20 transistors to build.
- DRAM latency is longer than that of SRAM
  - Because its bitline is not actively driven by a transistor.
  - DRAM must wait for charge to move (relatively) slowly from the capacitor to the bitline.

Memory Type	Transistors per Bit Cell	Latency
flip-flop	~20	fast
SRAM	6	medium
DRAM	1	slow

Read Only Memory ROM

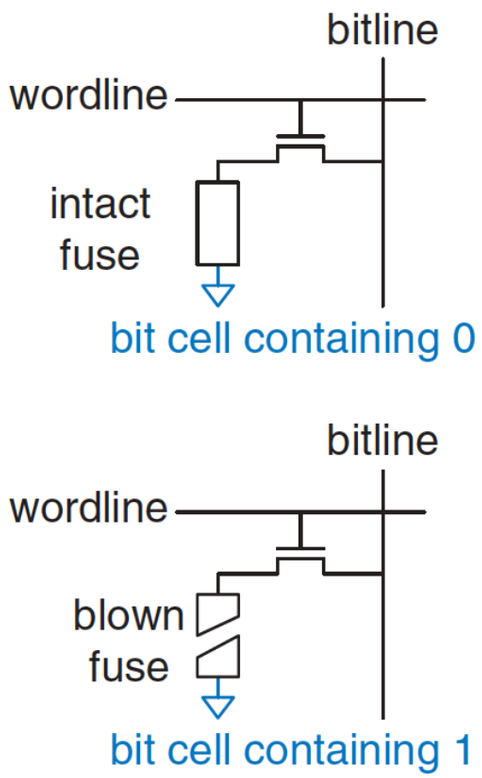
- Read only memory (ROM) stores a bit as the presence or absence of a transistor.
  - If the transistor is present, it pulls the bitline LOW.
  - If it is absent, the bitline remains HIGH.

The contents of the ROM bit are specified during **manufacturing** by the presence or absence of a transistor in each bit cell.



Read Only Memory ROM

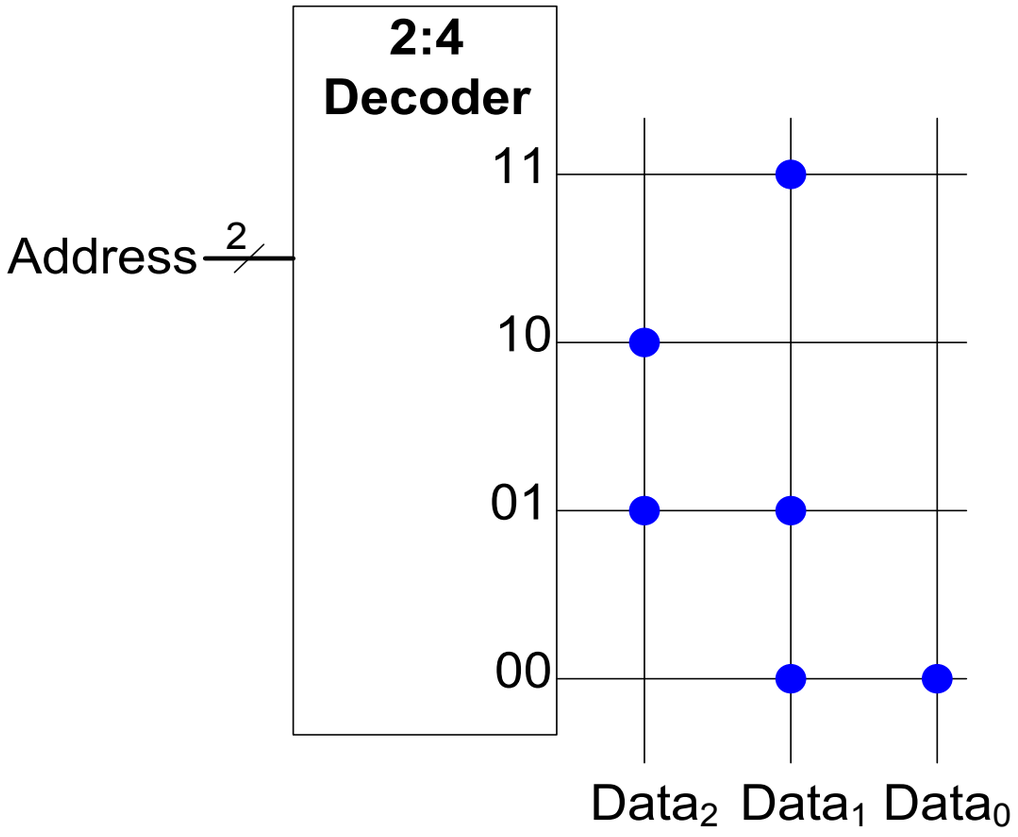
- A programmable ROM (PROM) places a transistor in every bit cell
  - but provides a way to connect or disconnect the transistor to ground.
- A fuse-programmable ROM allows user programs to selectively blow fuses by applying a high voltage.

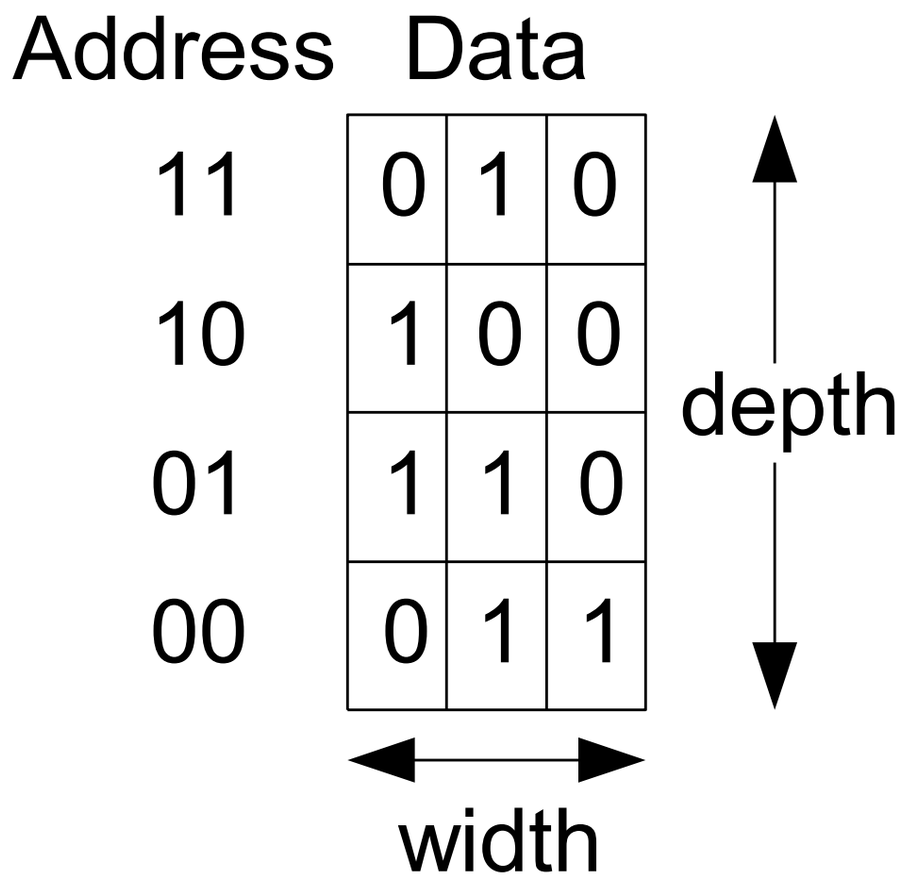


Read Only Memory ROM

- Erasable PROMs (EPROMs) replace the transistor and fuse with a *floating-gate transistor*.
  - –The floating gate is not physically attached to any other wires.
  - When suitable high voltages are applied, transistor is connects to the bitline to the wordline.
  - When the EPROM is exposed to intense ultraviolet (UV) light for about half an hour, the transistor is disconnected.
  - These actions are called **programming** and **erasing**, respectively.

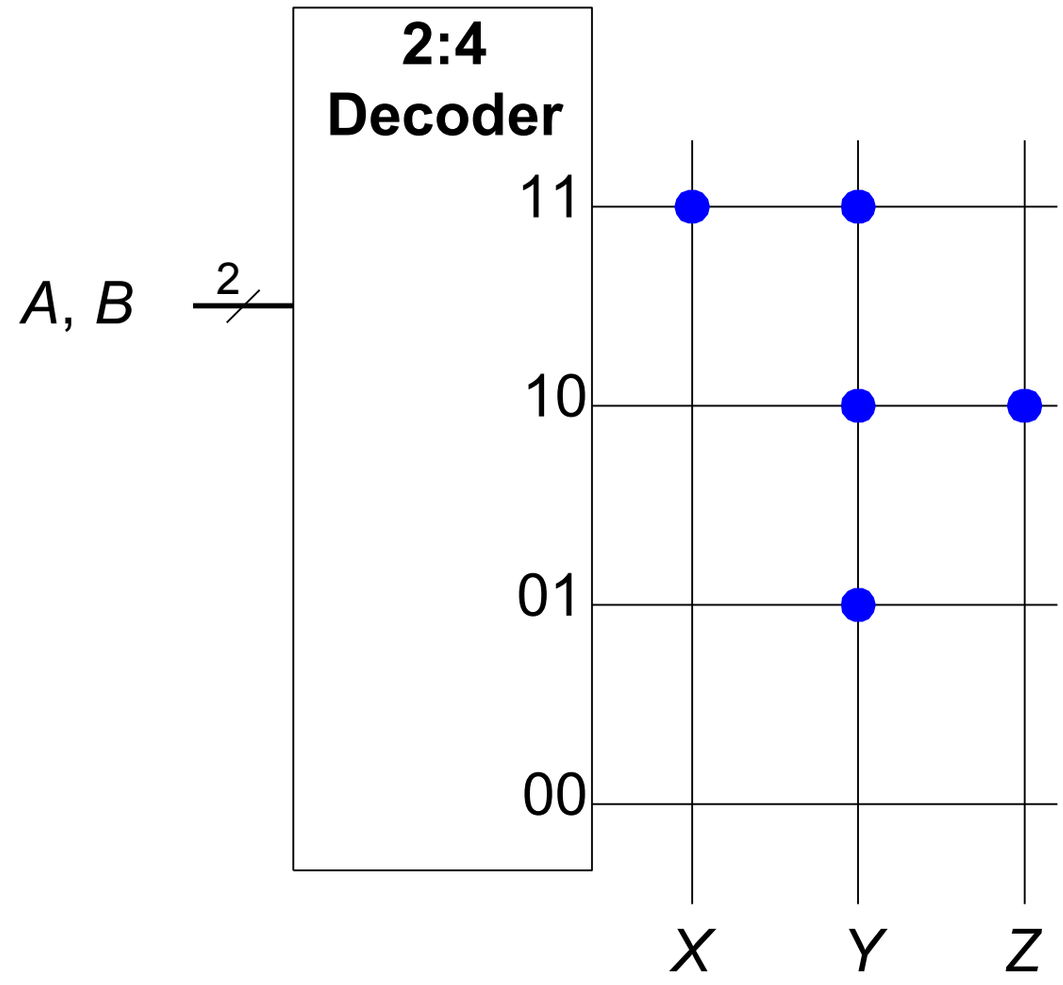
Read Only Memory ROM: Dot Notation





Logic with Memory Arrays

- Although they are used primarily for data storage, memory arrays can also perform **combinational logic functions**.
- A 2^N-word x M-bit memory can perform any combinational function of **N inputs** and **M outputs**.



X = AB

Y = A + B

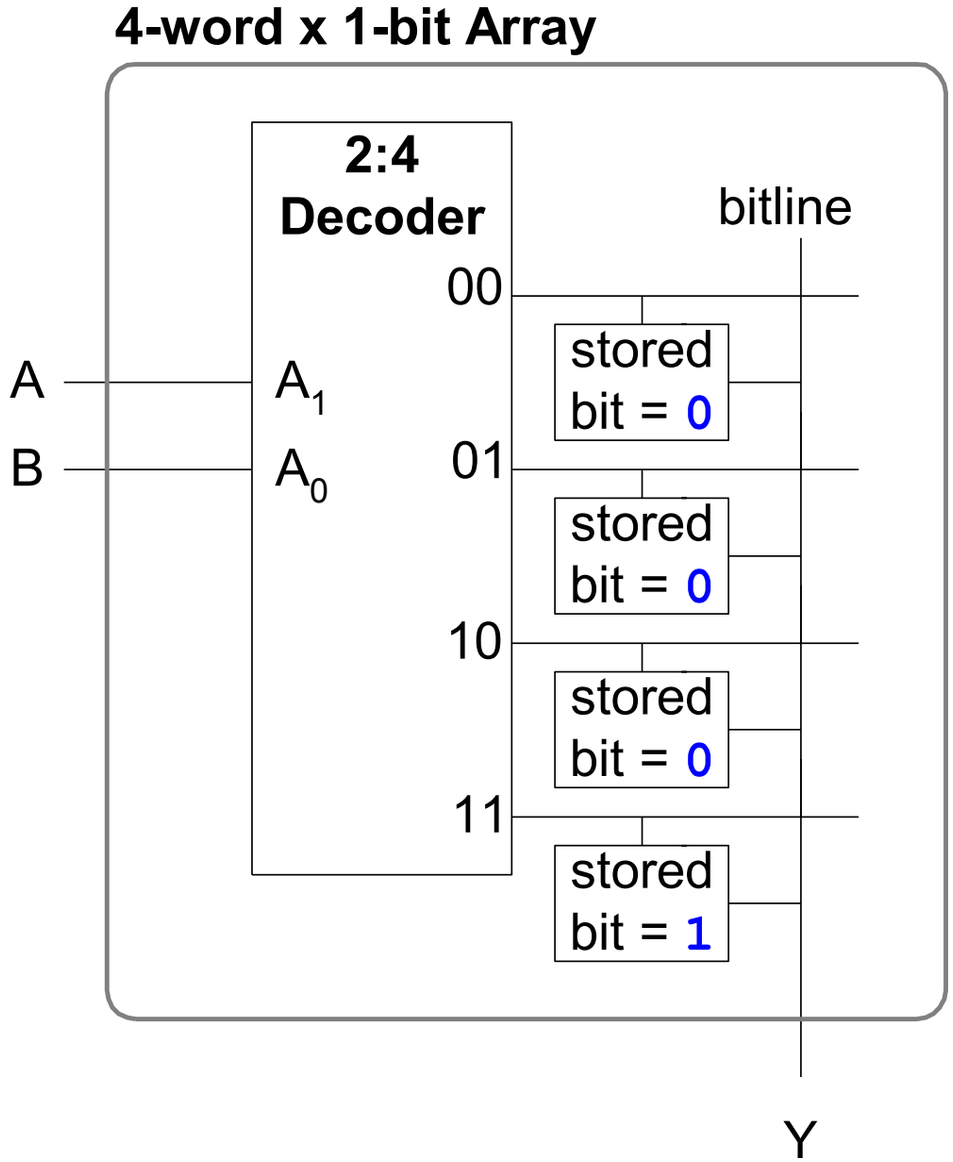
Z = A $\overline{B}$

Logic with Memory Arrays

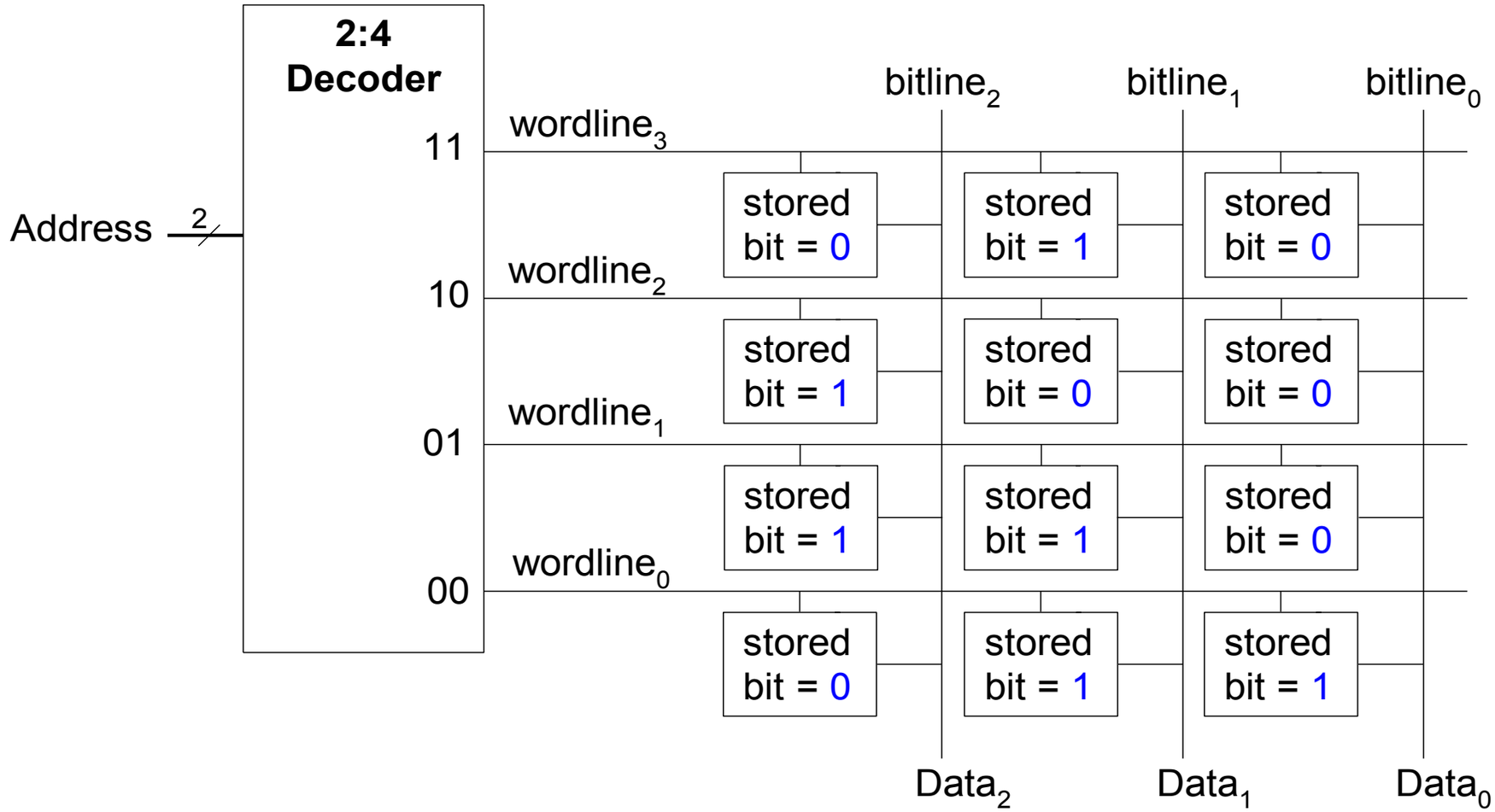
- Called *lookup tables* (LUTs): look up output at each input combination (address)

Truth Table

A	B	Y
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**



Logic with Memory Arrays



$$Data_2 = A_1 \text{ xor } A_0$$
$$Data_1 = \bar{A}_1 + A_0$$
$$Data_0 = \bar{A}_1\bar{A}_0$$

Further Reading

- You can read **Chapter 5** of your book
  - **Sections 5.4 and 5.5**