ECE 2300
Digital Logic & Computer Organization

Fall 2016

More Memories
Single Cycle Microprocessor

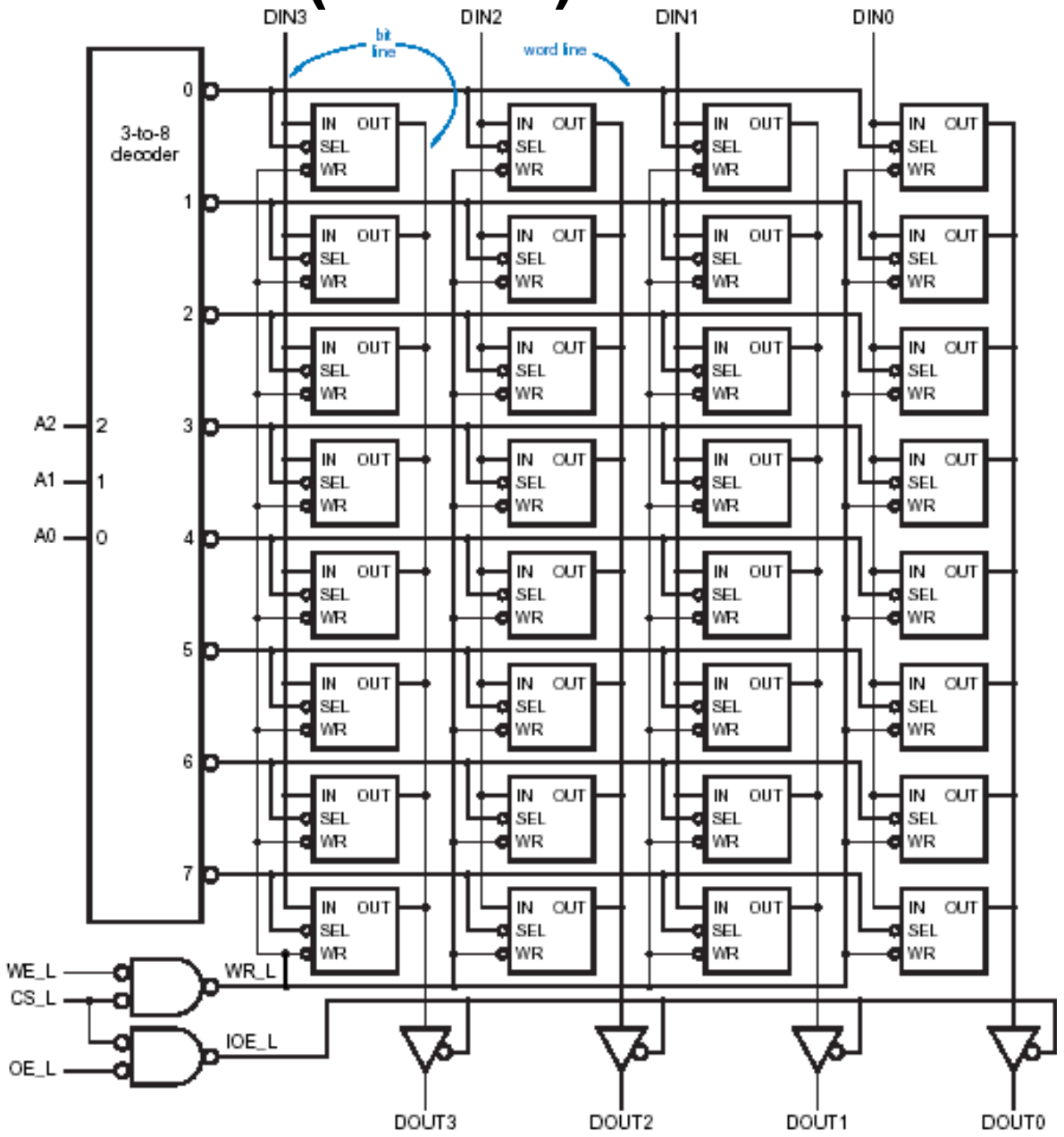
Cornell University

Types of Memories

- **Random Access Memory (RAM)**
  - Read and write any location at similar speeds
  - Volatile: loses contents when powered off
  - SRAM, DRAM

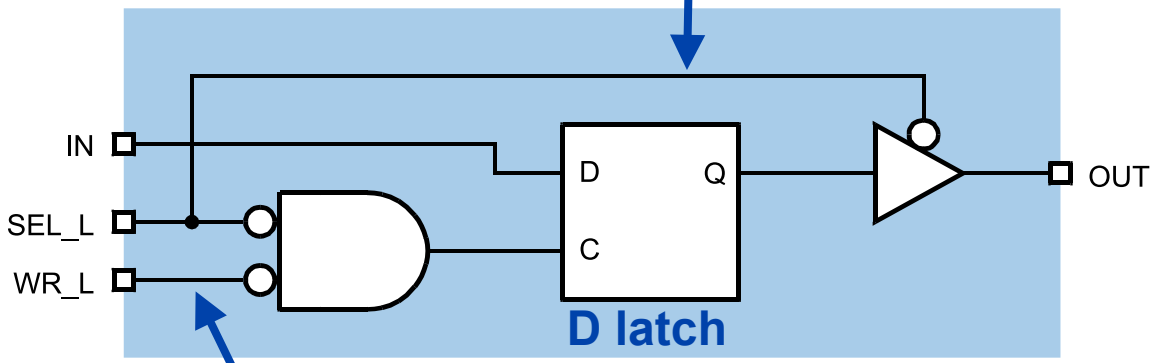
- **Read-Only Memory (ROM)**
  - Truly read-only
    - Written in the factory, and never written after installation
  - Mostly read and rarely written
    - Much faster to read than write
  - Non-volatile
  - ROM, PROM, EPROM, EEPROM, Flash memory

Static RAM (SRAM) Internal Structure



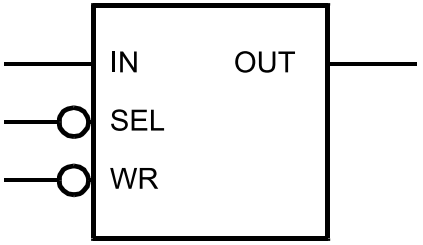
SRAM Cell

SEL_L asserted to access output (read)



SEL_L and WR_L asserted to enable latch (write)

≡



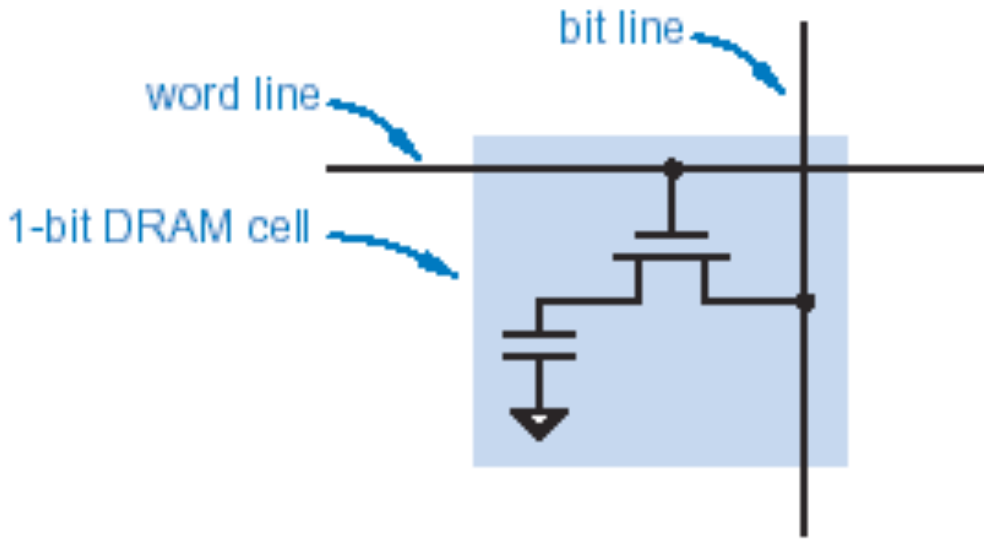
Dynamic RAM (DRAM)

- SRAM advantages and disadvantages
  - Very fast (+)
  - High power (−)
  - Relatively high area/bit (−)

- DRAM
  - Single transistor storage cell
  - Higher density ➔ lower cost/bit
  - Lower power/bit

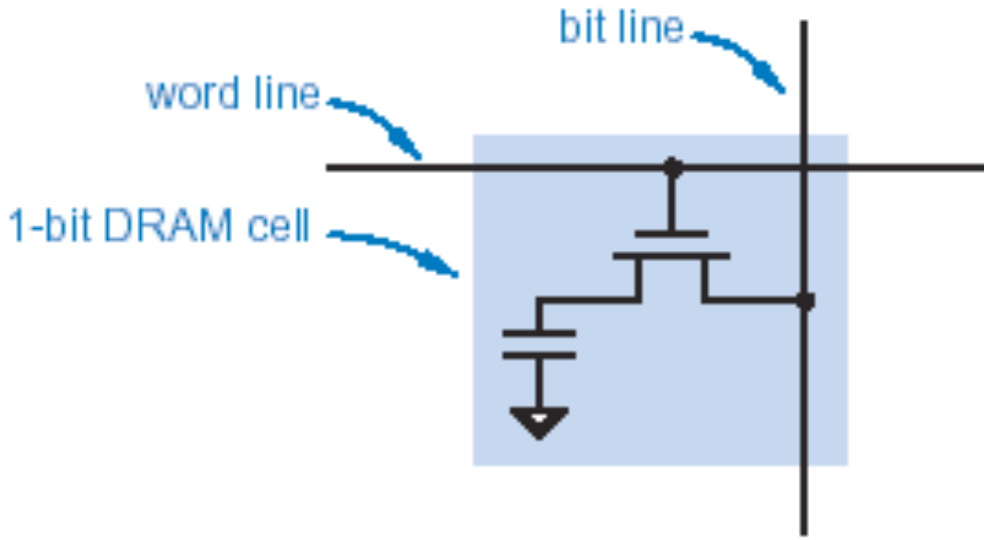
DRAM Bit Structure

- Capacitor accessed through a transistor
- Capacitor is charged through the bit line to store a 1
- Capacitor is discharged through the bit line to store a 0



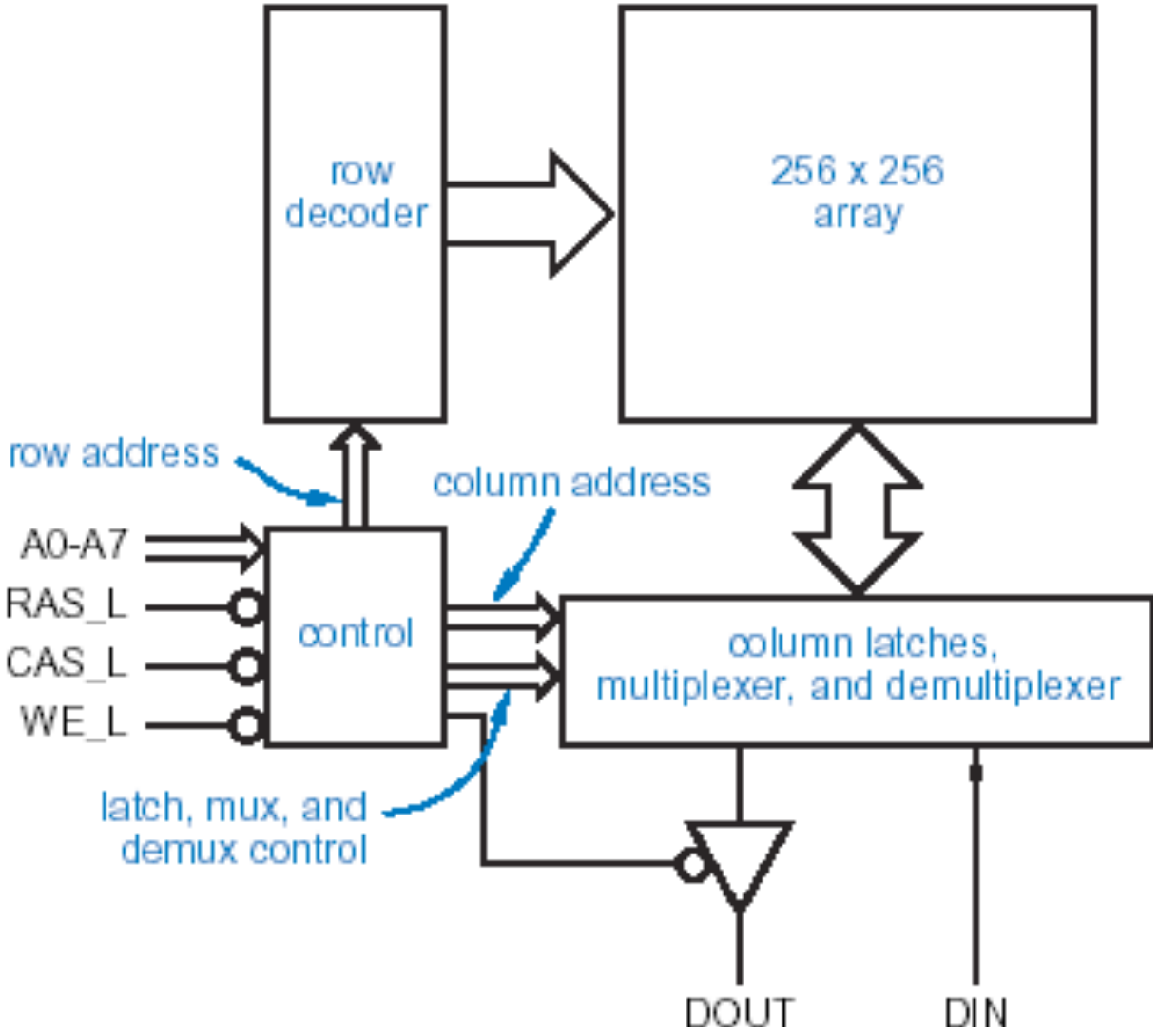
DRAM Read

- Bit line *precharged* halfway between 0 and 1
- Word line is asserted
- Capacitor voltage pulls the bit line slightly higher or lower
- *Sense amplifier* detects this small change (1 or 0)

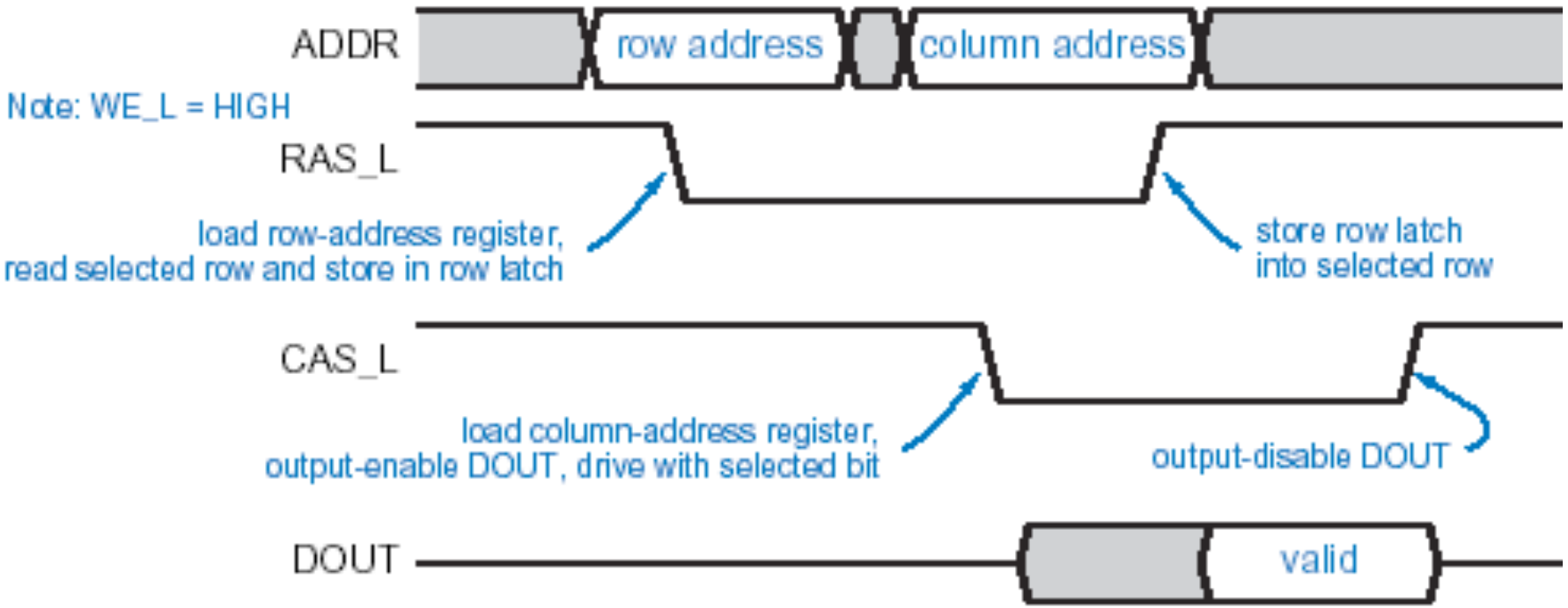


DRAM Organization (64K x 1)

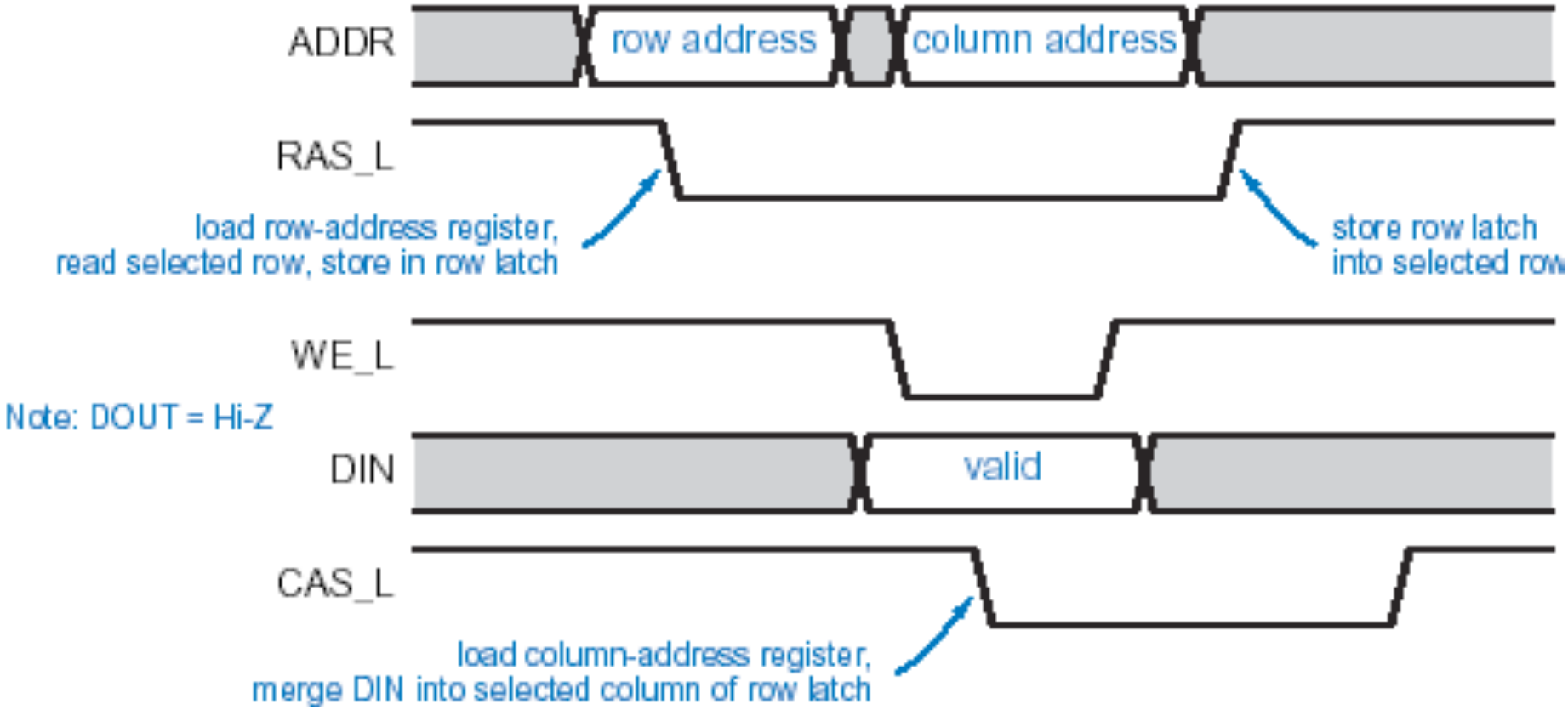
- **Multiplexed address inputs**
- **Row-address strobe (RAS)**
  - **Selects row of array**
- **Column-address strobe (CAS)**
  - **Selects subset of the row**



DRAM Read Cycle

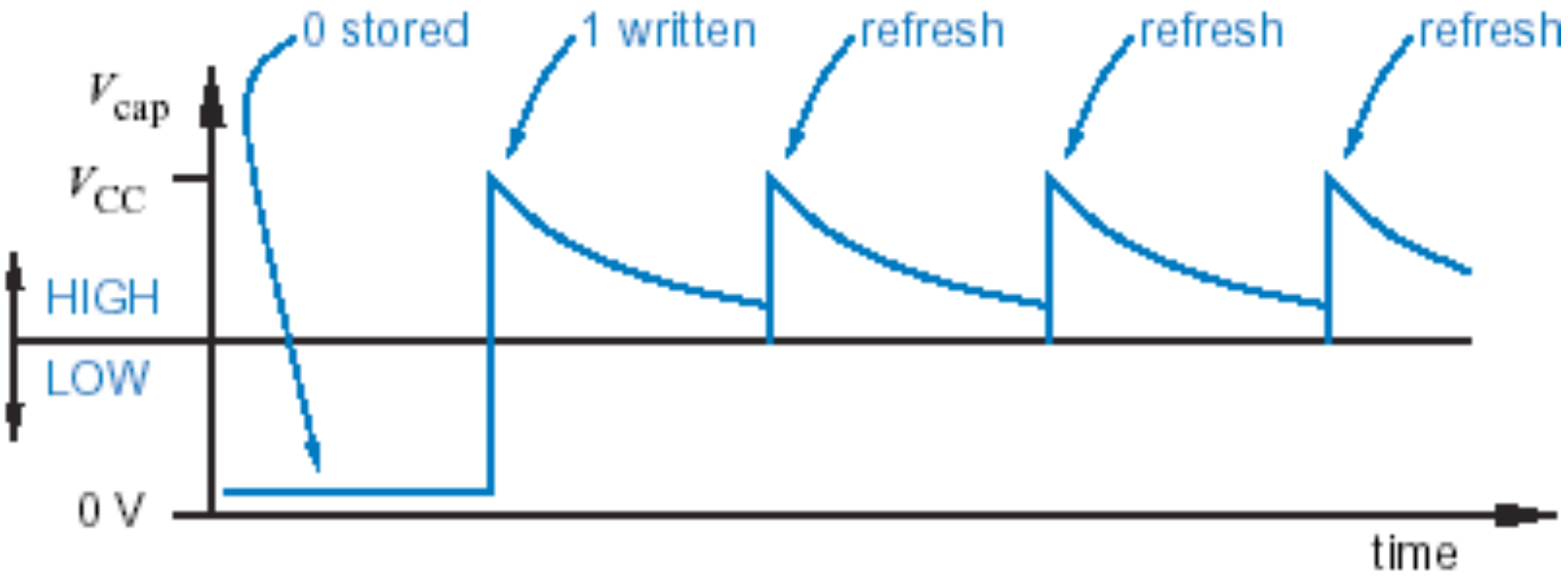


DRAM Write Cycle



DRAM Refresh

- Capacitors discharge over time
- Periodic *refresh* cycles recharge each memory bit
- Each row periodically accessed using RAS, which restores the charge



DRAM Refresh

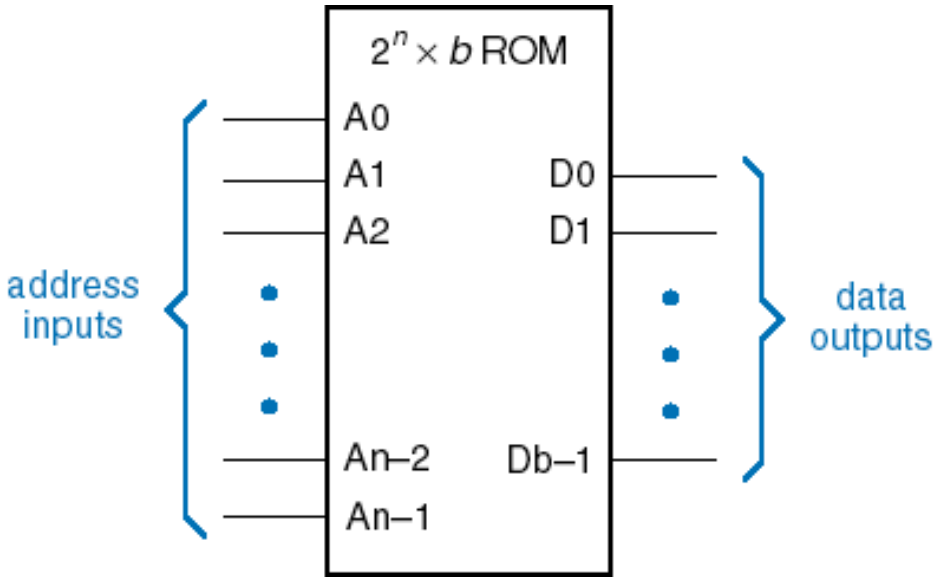
- Refresh an entire row at a time
- Typically, each bit must be refreshed every 64ms
- DRAM chip comprises multiple arrays (*banks*) that can be refreshed in parallel
- Assuming 4096 rows per bank, DRAM controller initiates a refresh cycle every 15.6 microseconds

Synchronous DRAM (SDRAM)

- D FFs on input and output signals
- Can “pipeline” multiple read and write operations
- Multiple banks can be accessed concurrently
- DDR: data transferred on both rising and falling clock edges

Read-Only Memory (ROM)

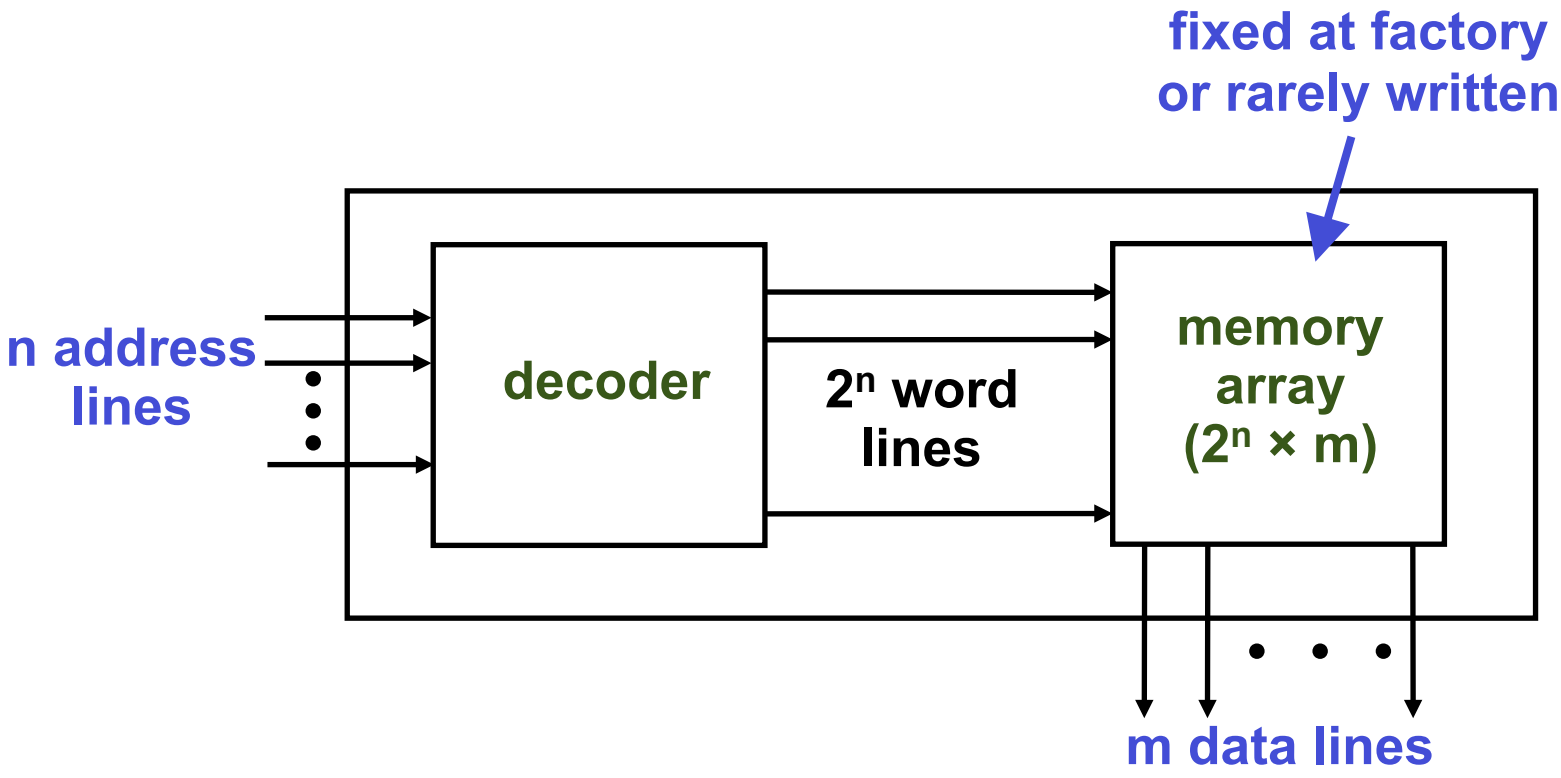
- **Truly read-only**
  - **Written in the factory, and never written after installation**
- **Mostly read and rarely written**
  - **Faster to read than write**
- **Non-volatile**
- **ROM, PROM, EPROM, EEPROM, Flash memory**



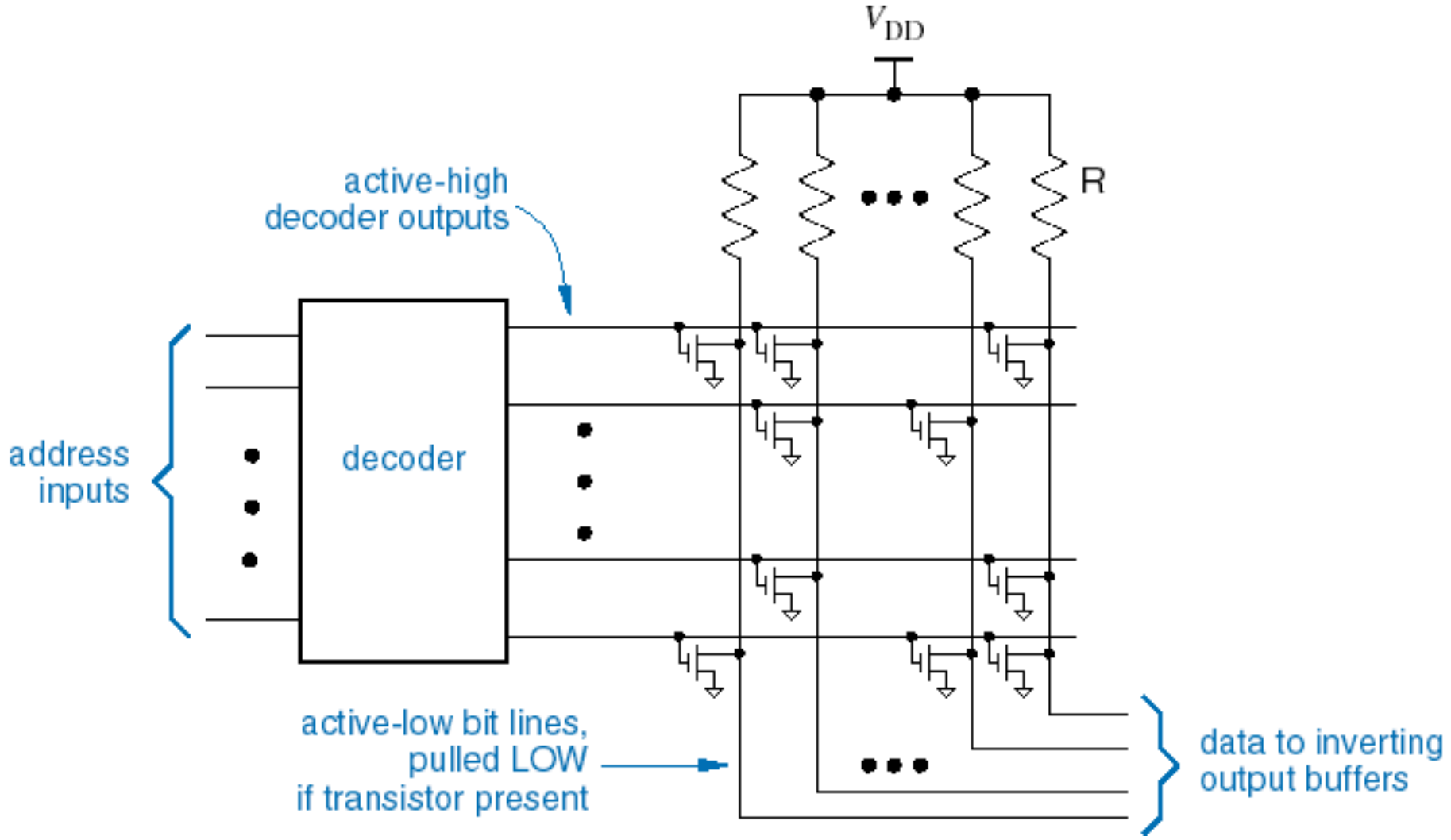
Applications of ROM

- **Program storage**
  - Boot code for personal computers
  - Complete application storage for embedded systems
- **Data storage**
  - Configuration information, music players, SSDs
- **Combinational logic functions**
  - Lookup table
    - Address inputs = function inputs
    - Data outputs = function outputs

ROM Structure



ROM Memory Array Organization

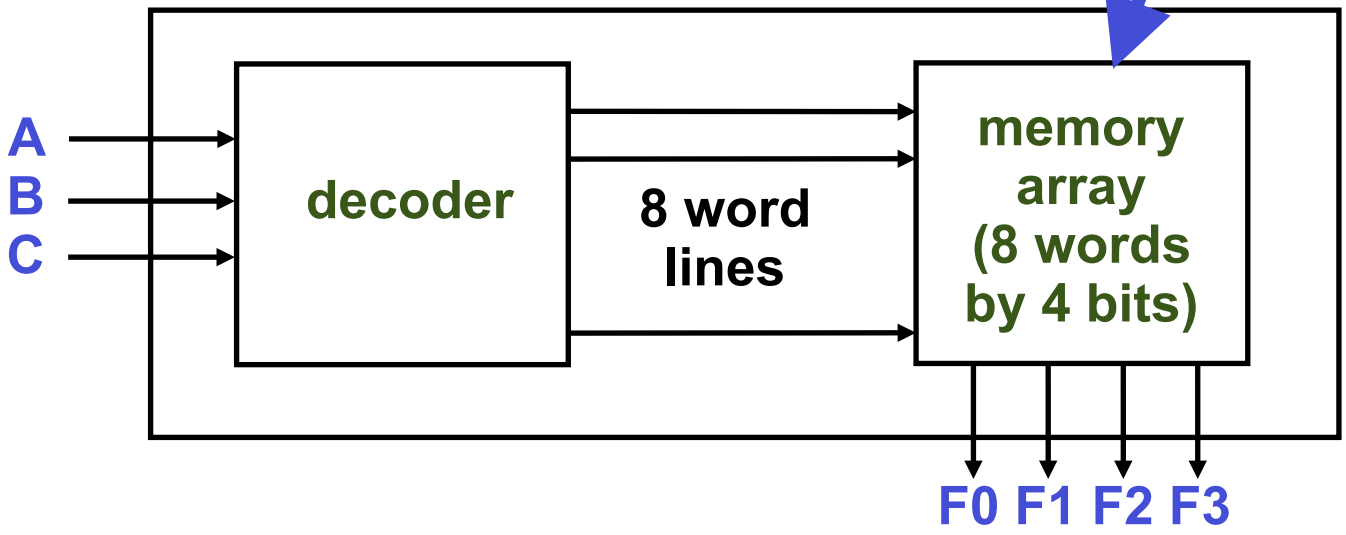


Using ROMs for Combinational Logic

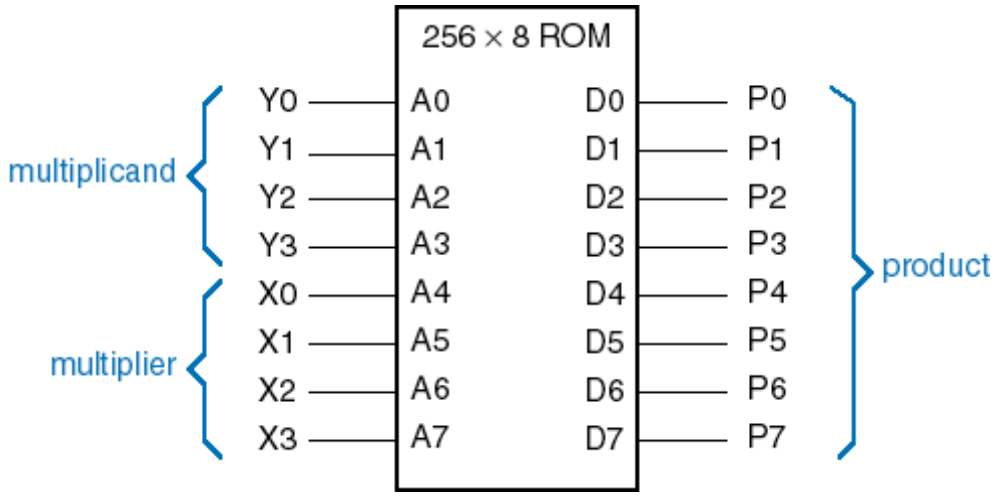
- Can implement sum-of-products using a ROM

F0 = A' B' C + A B' C' + A B' C
F1 = A' B' C + A' B C' + A B C
F2 = A' B' C' + A' B' C + A B' C'
F3 = A' B C + A B' C' + A B C'

A	B	C	F0	F1	F2	F3
0	0	0	0	0	1	0
0	0	1	1	1	1	0
0	1	0	0	1	0	0
0	1	1	0	0	0	1
1	0	0	1	0	1	1
1	0	1	1	0	0	0
1	1	0	0	0	0	1
1	1	1	0	1	0	0



Multiplier Using ROM



9x8 = 72 = 0x48

```
00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
20: 00 02 04 06 08 0A 0C 0E 10 12 14 16 18 1A 1C 1E
30: 00 03 06 09 0C 0F 12 15 18 1B 1E 21 24 27 2A 2D
40: 00 04 08 0C 10 14 18 1C 20 24 28 2C 30 34 38 3C
50: 00 05 0A 0F 14 19 1E 23 28 2D 32 37 3C 41 46 4B
60: 00 06 0C 12 18 1E 24 2A 30 36 3C 42 48 4E 54 5A
70: 00 07 0E 15 1C 23 2A 31 38 3F 46 4D 54 5B 62 69
80: 00 08 10 18 20 28 30 38 40 48 50 58 60 68 70 78
90: 00 09 12 1B 24 2D 36 3F 48 51 5A 63 6C 75 7E 87
A0: 00 0A 14 1E 28 32 3C 46 50 5A 64 6E 78 82 8C 96
B0: 00 0B 16 21 2C 37 42 4D 58 63 6E 79 84 8F 9A A5
C0: 00 0C 18 24 30 3C 48 54 60 6C 78 84 90 9C A8 B4
D0: 00 0D 1A 27 34 41 4E 5B 68 75 82 8F 9C A9 B6 C3
E0: 00 0E 1C 2A 38 46 54 62 70 7E 8C 9A A8 B6 C4 D2
F0: 00 0F 1E 2D 3C 4B 5A 69 78 87 96 A5 B4 C3 D2 E1
```

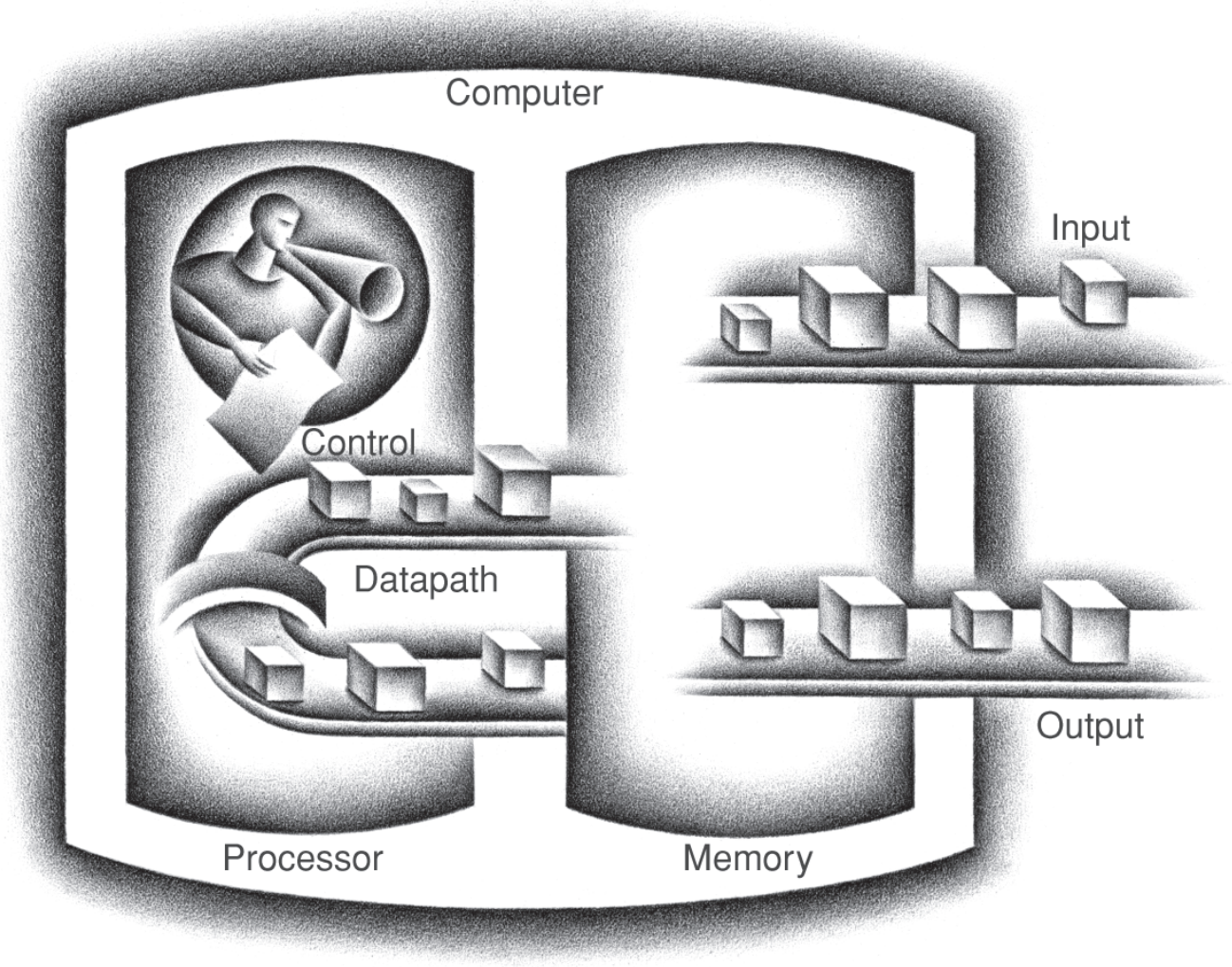
This Concludes Part 1

- Boolean algebra
- Combinational logic and minimization
- Logic functions
- CMOS gates
- Binary arithmetic and ALUs
- Latches and flip-flops
- Counters and shift registers
- Verilog
- Finite state machines
- Hazards, timing, clocking
- Memories

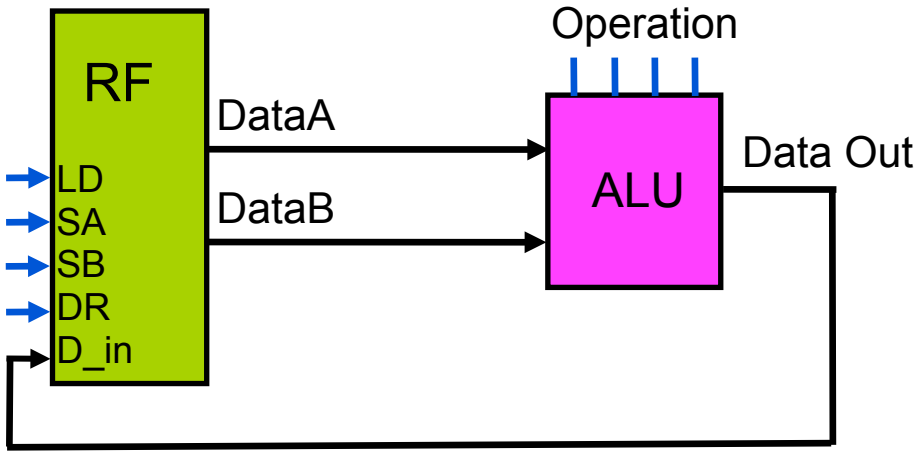
Part 2 Overview

- Single cycle microprocessor
- Instruction set architecture
- Pipelined microprocessor
- Caches and main memory
- Virtual memory
- Input/output
- Exceptions
- Case study

Organization of a Computer



The Basic Processing Cycle

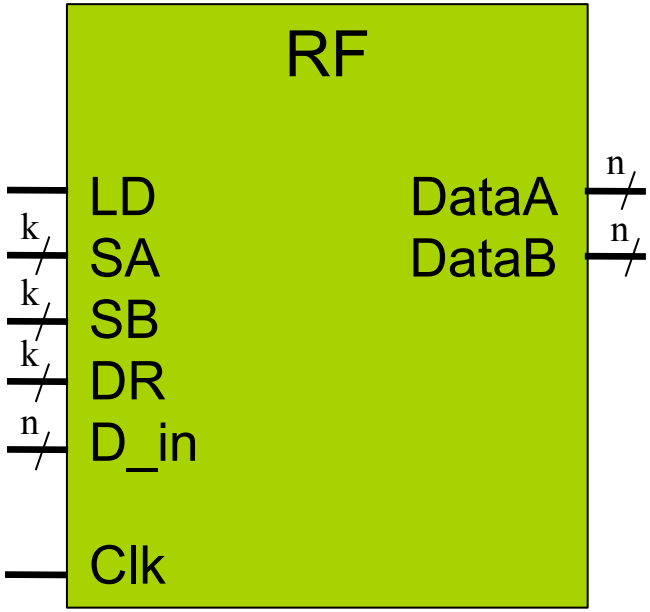


- Read data from two registers
- Perform an operation
- Place the result into a register

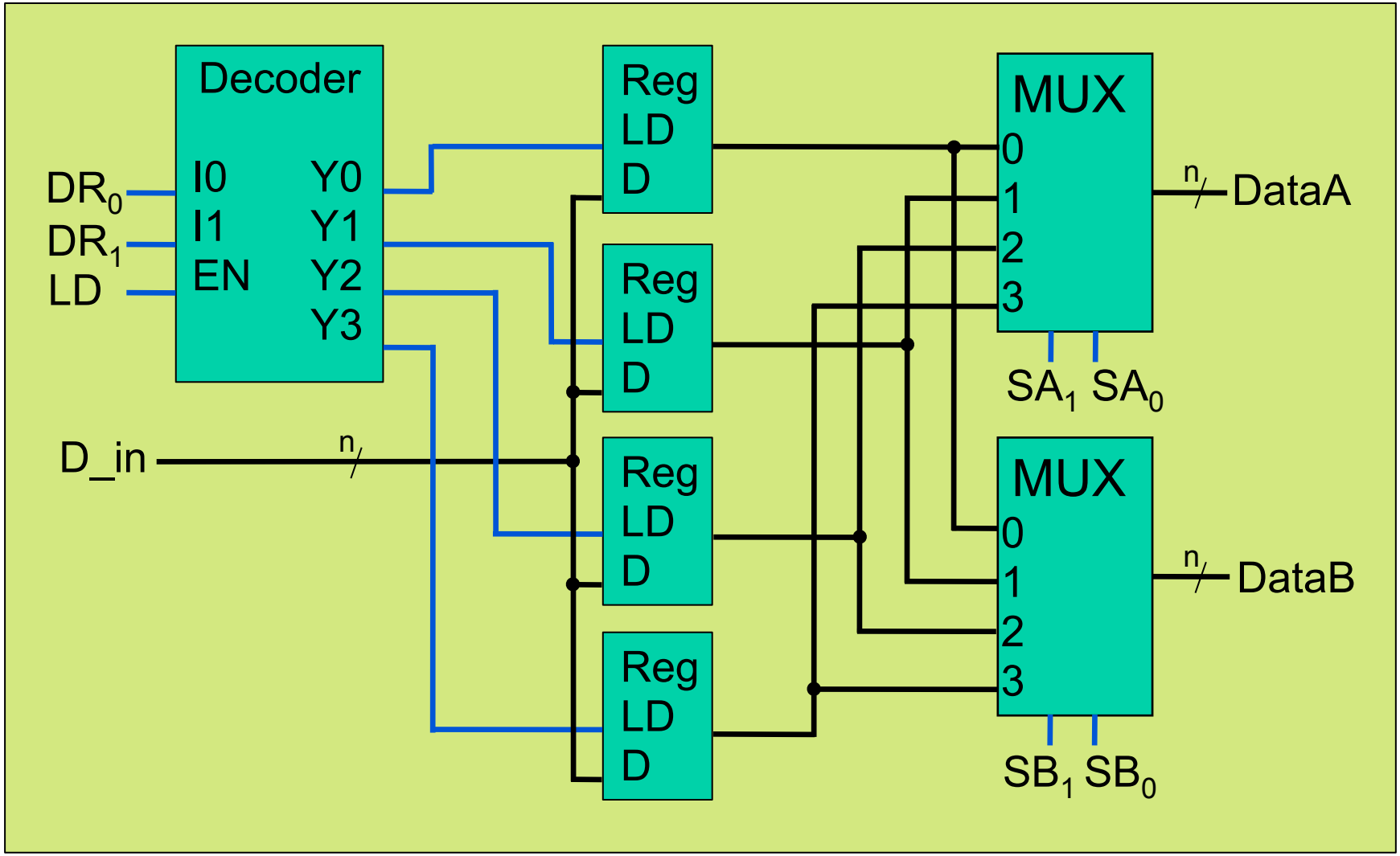
- All three steps performed in 1 clock cycle

Register File

- Collection of 2^k n-bit loadable registers
- Control inputs
  - SA – Source address A
  - SB – Source address B
  - DR – Destination address
  - LD – Load destination register with D_in
- Data inputs
  - D_in – Input data
- Data outputs
  - DataA – Output data A
  - DataB – Output data B

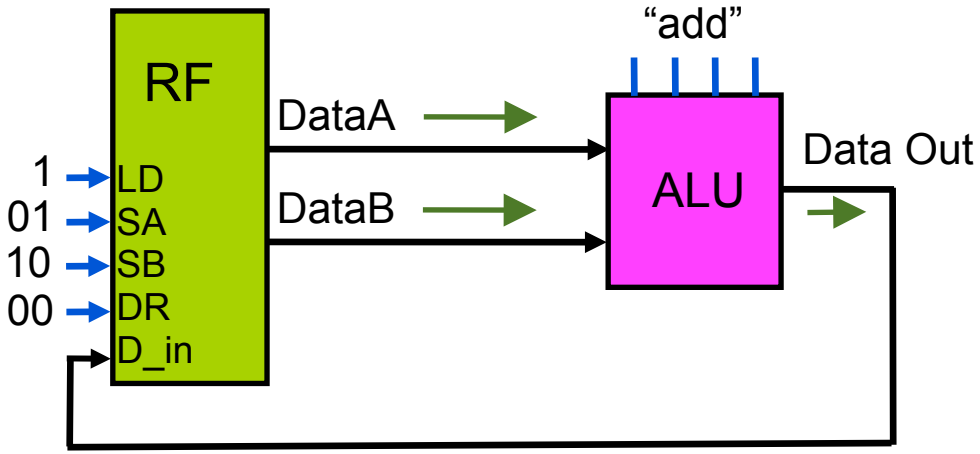


Register File



Example with 4 registers. Typically have 32 or more.

Instruction Execution



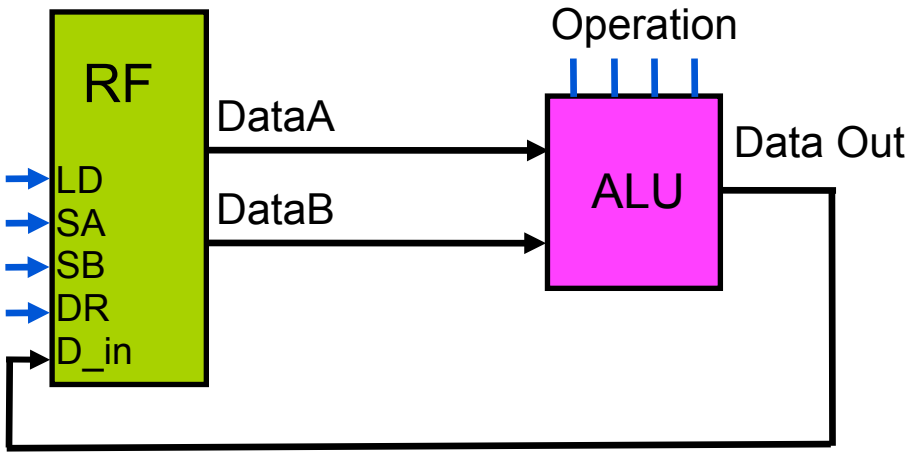
ADD R0, R1, R2

operation

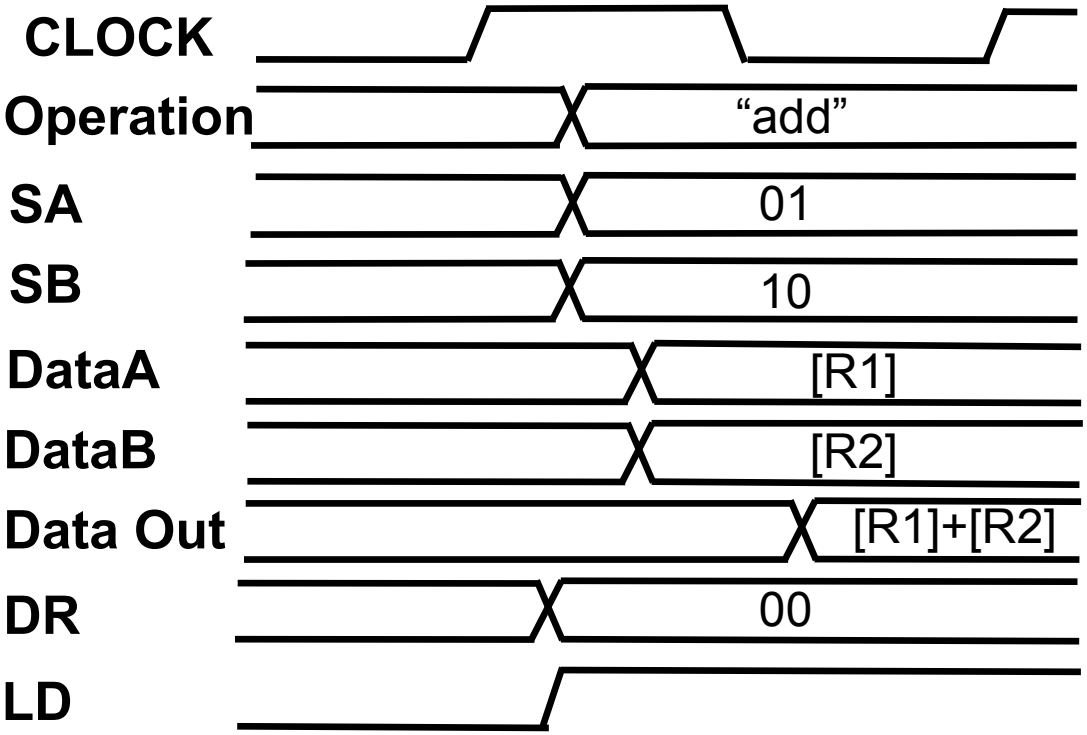
destination register

source registers

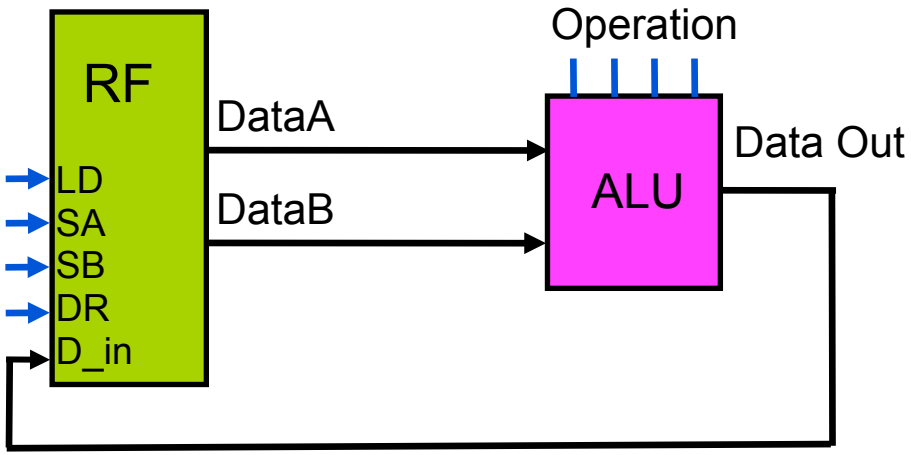
Instruction Execution



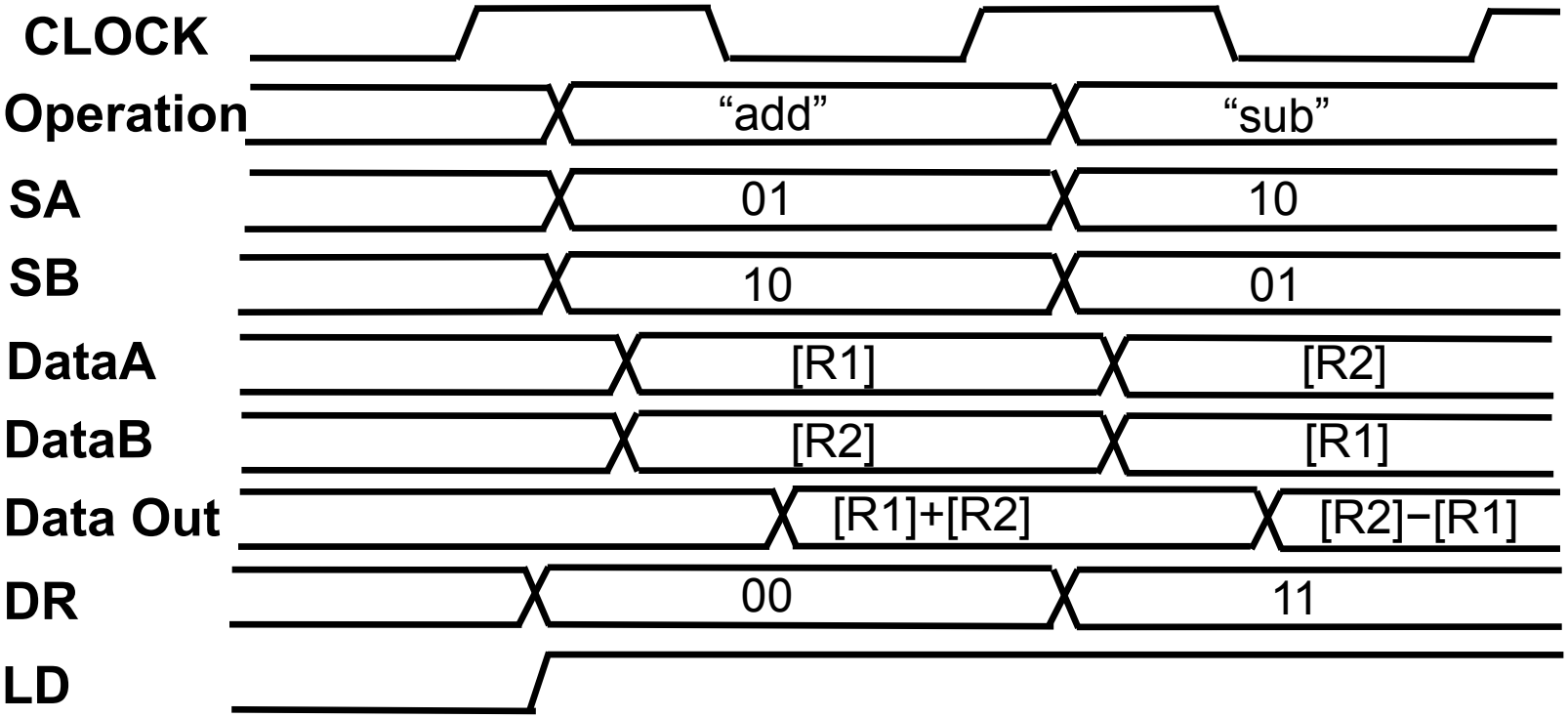
ADD R0, R1, R2



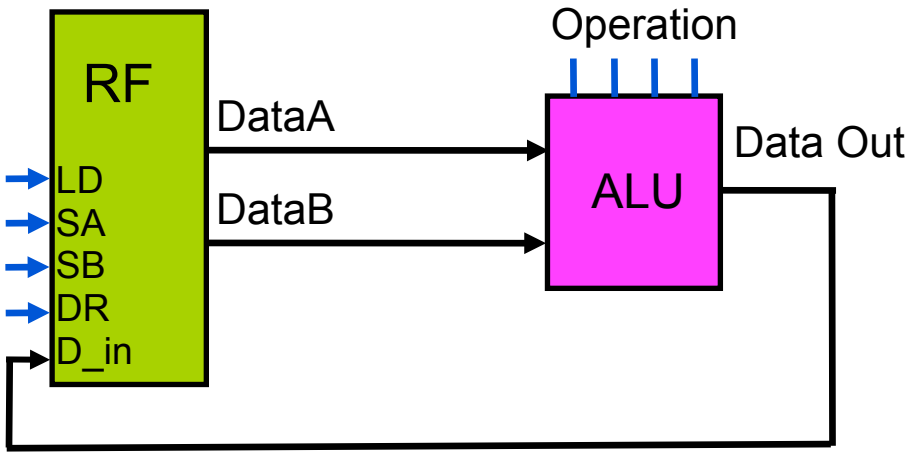
Instruction Execution



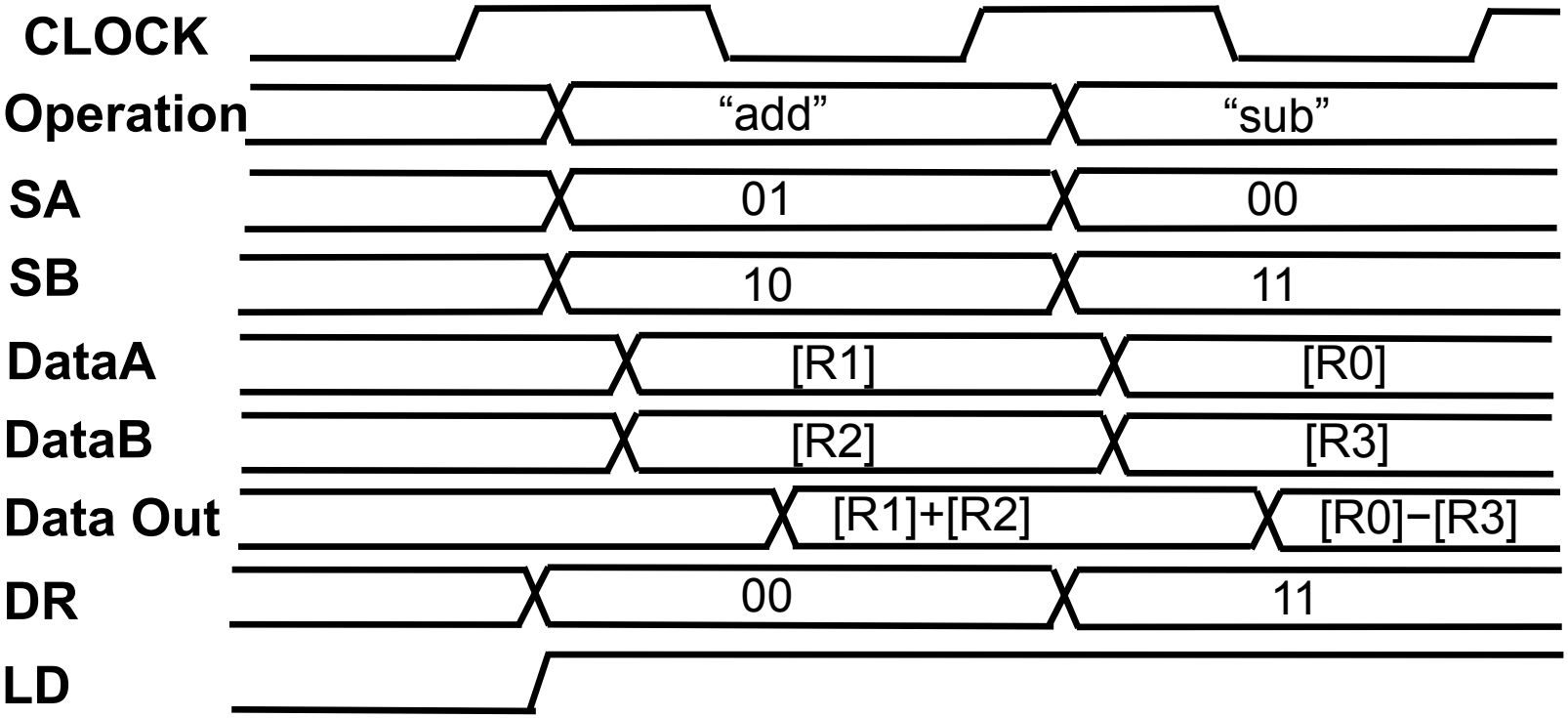
ADD R0, R1, R2
SUB R3, R2, R1



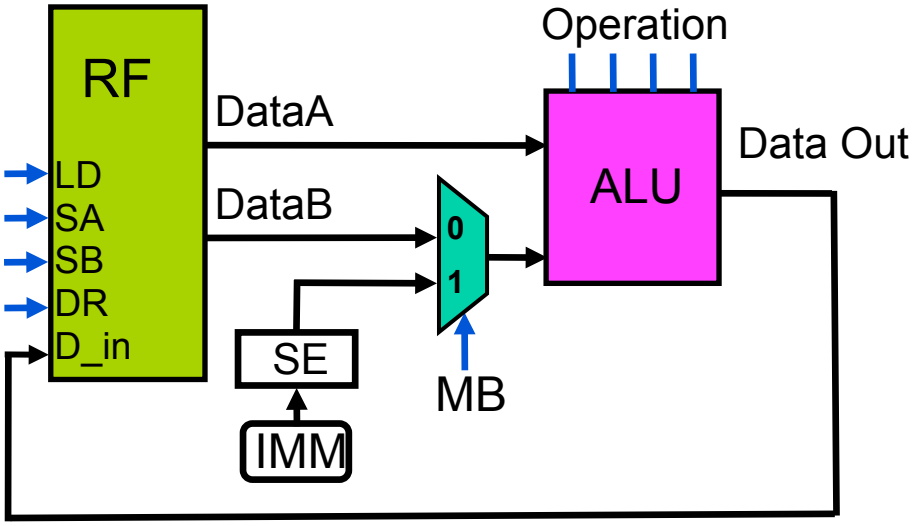
Instruction Execution



ADD R0, R1, R2
SUB R3, R0, R3



Operations With Constants

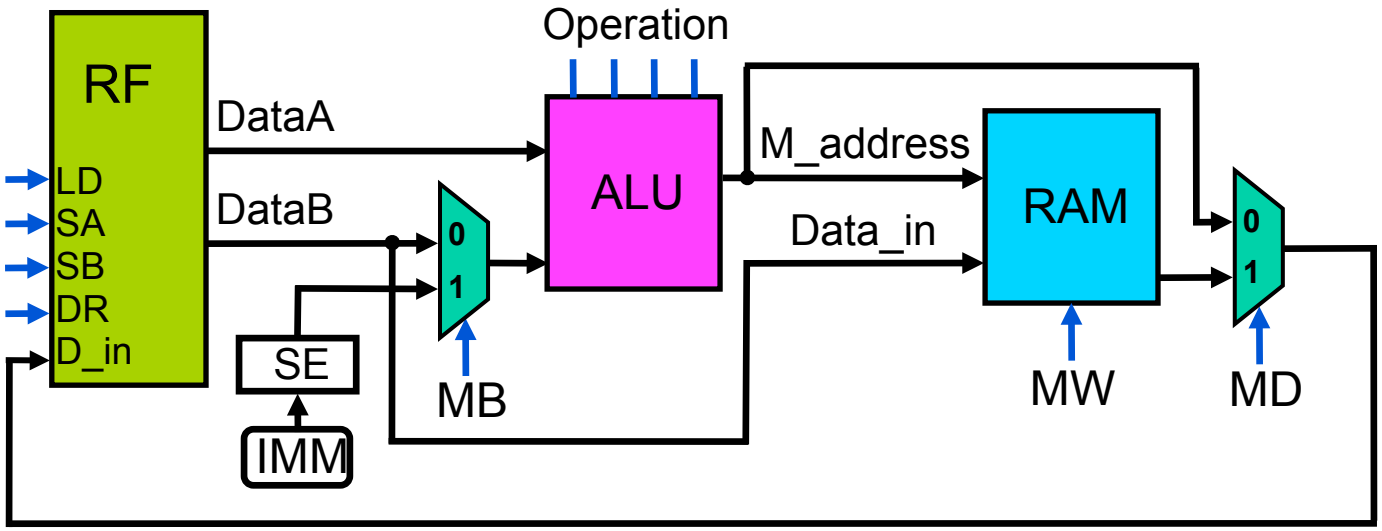


ADDI R1, R1, 1

- **Constants are called *immediate values***
- **Sign extend (SE) IMM to the width of DataA to perform correct two's complement operation**
  - **Why? May not have enough bits in instruction (later)**
  - **Assume IMM is 4 bits and DataA is 8 bits wide**

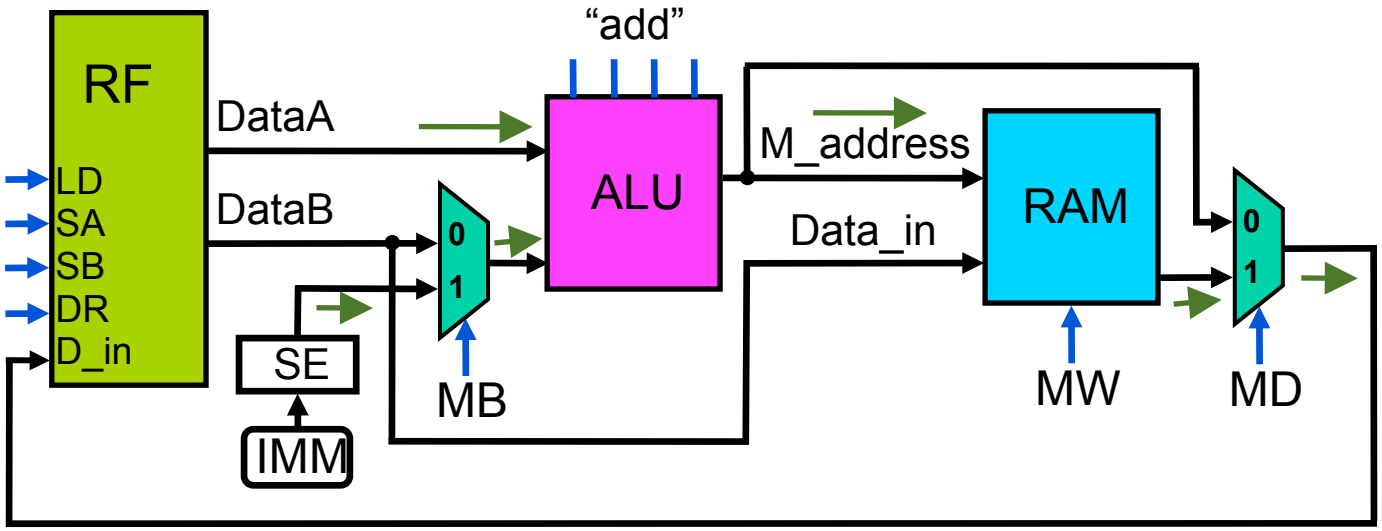
0101 → 00000101
1110 → 11111110

Reading and Writing Memory



- Most data is held in memory (RAM)
- Must be moved into a register in order to operate on it
- Data is also moved out of registers into memory
  - To make room for other data
  - To move it to permanent storage (e.g., disk)

Reading Memory ("Load")

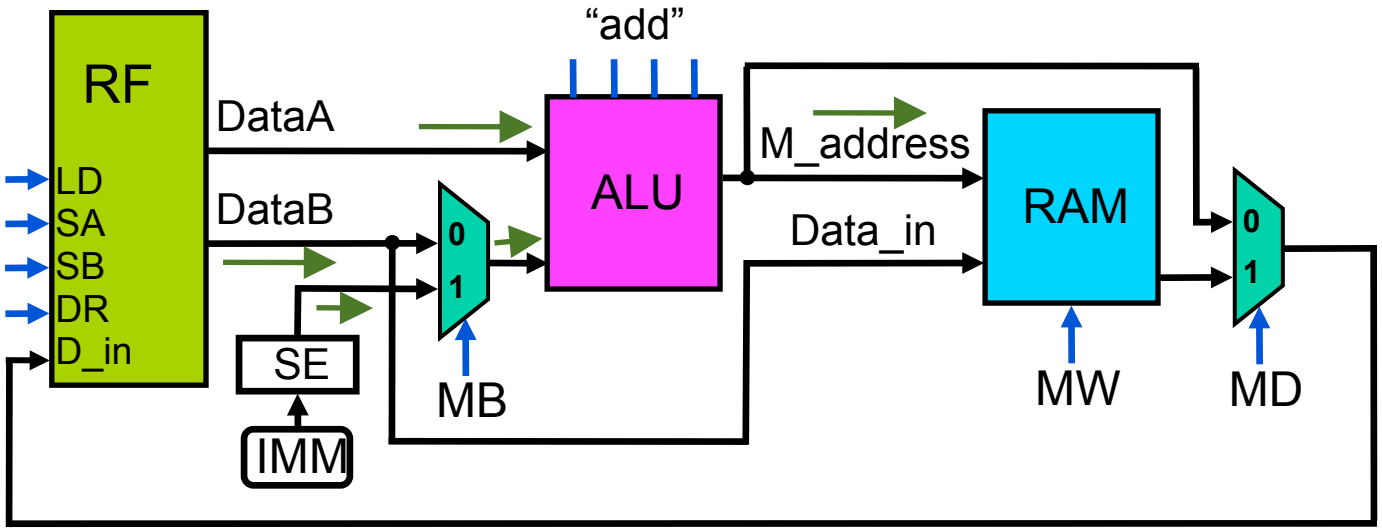


LOAD R3, 4(R1)

<u>Step 1</u>: Form the memory address by adding the value in R1 with the immediate (*offset*) 4

<u>Step 2</u>: Read the data at that address in RAM and place it in R3

Writing Memory ("Store")



STORE R2, 0(R0)

Step 1: Form the memory address by adding the value in R0 with the immediate 0

Step 2: Write the value in R2 into the RAM at that address

Before Next Class

- H&H 7.3.2-7.3.4

Next Time

More Single Cycle Microprocessor