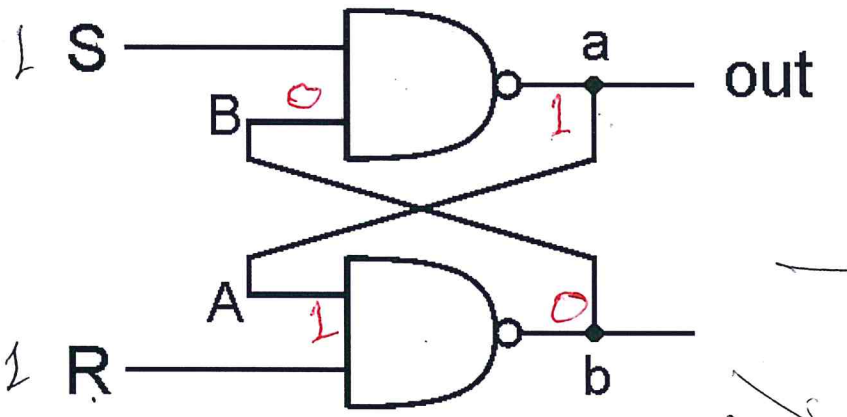
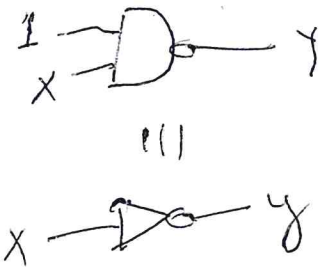
6. Memory, Finite-State Machines Sept. 19, 2018 (Chapter 3, 3.5, 3.6)

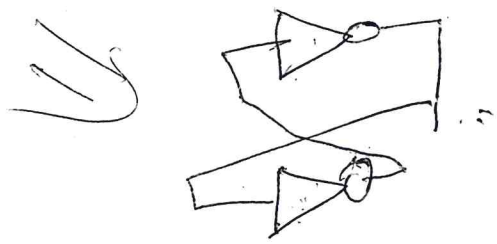
- Review
  - Storage Elements (latches)
- D-Latch, Registers
- Memories
  - Address space
  - Memory structure
- Finite-State Machines
  - State diagram
  - Implementation
- Example: warning sign

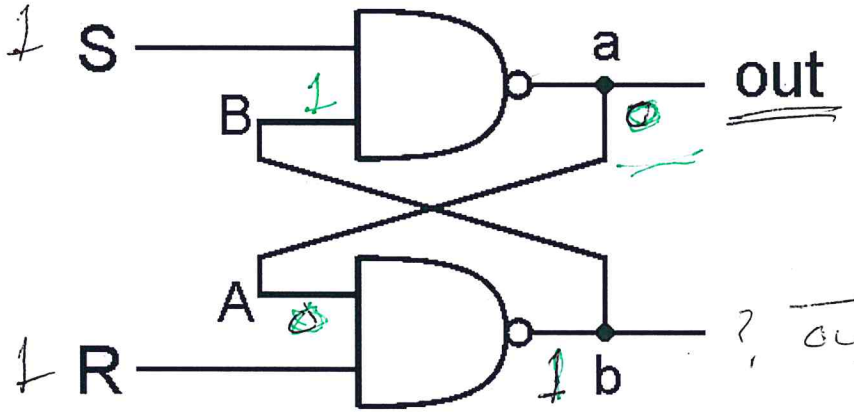
R-S Latch: Simple Storage Element





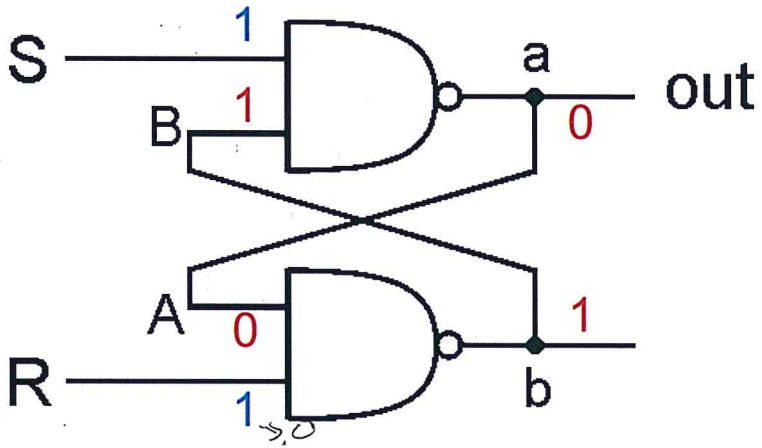
STORAGE

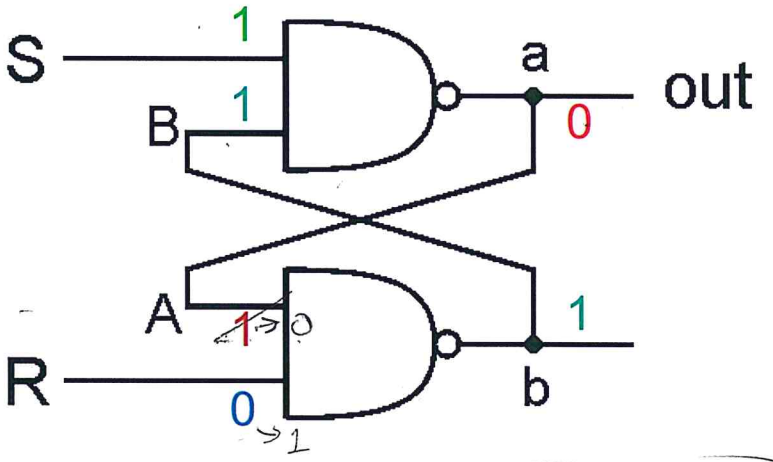


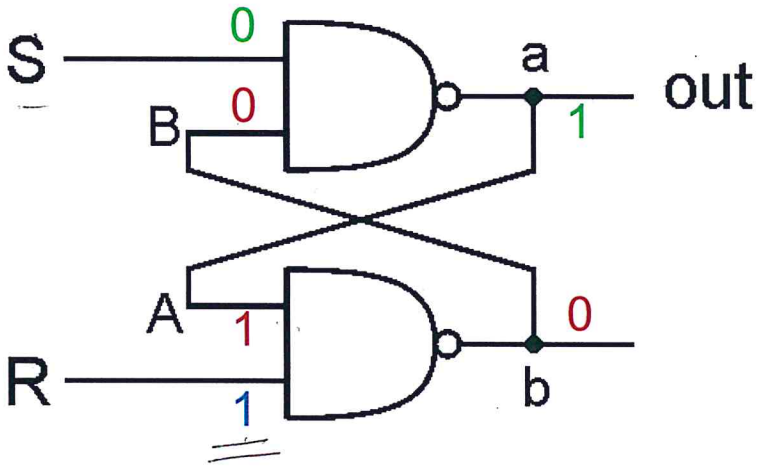


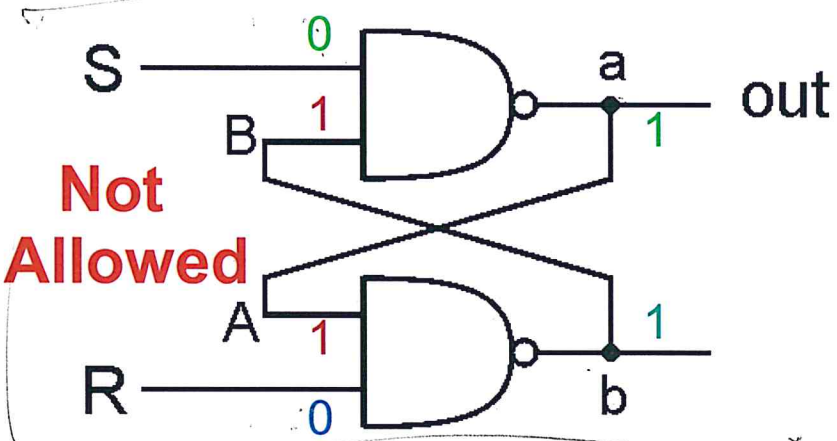
R-S Latch Behavior

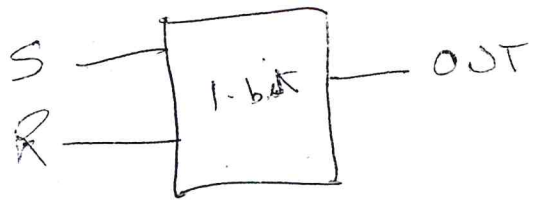
Start with output = 0

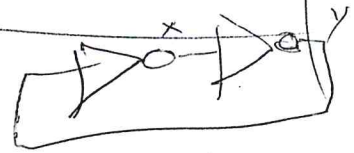












S-R LATCH

R = S = 1 : HOLD (PREVIOUS VALUE)

R = 0, S = 1 : OUT = 0

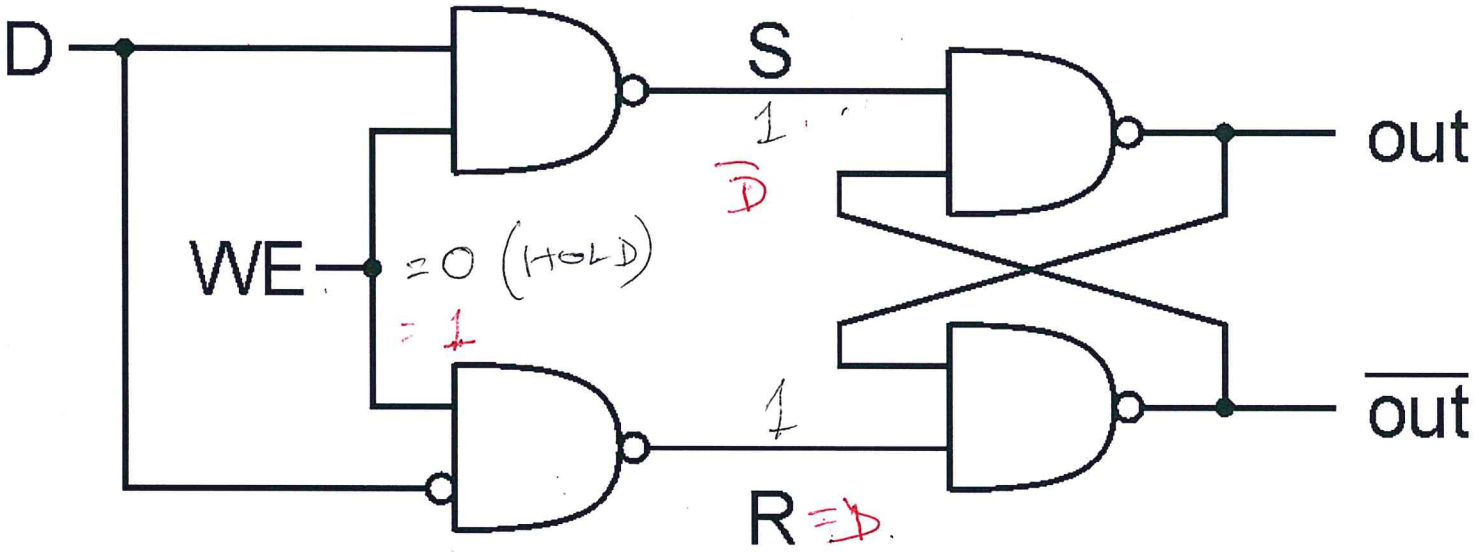
R = 1, S = 0 : OUT = 1

R = 0, S = 0 : NOT ALLOWED

Gated D-Latch

Two inputs: D (data) and WE (write enable)

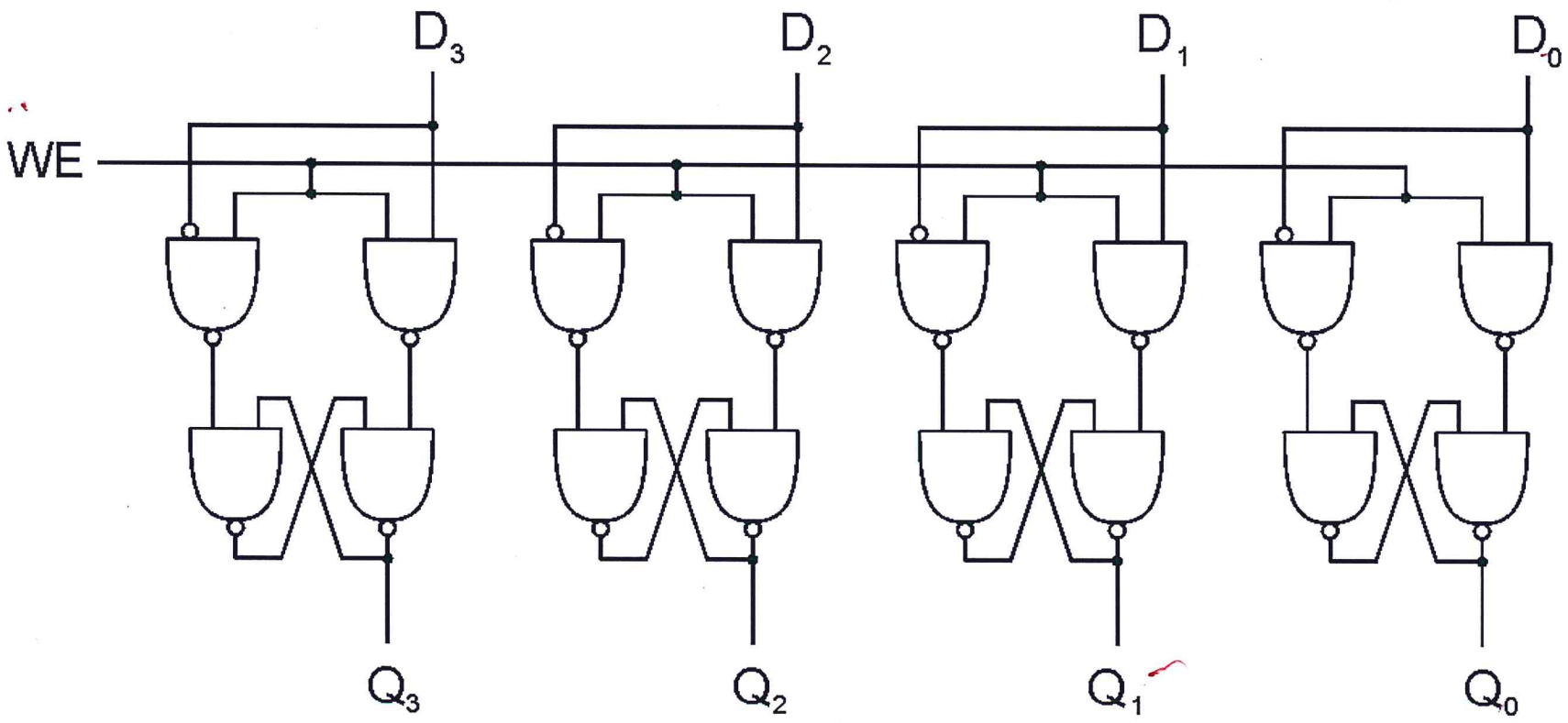
- when WE = 1, latch is set to value of D
  - S = NOT(D), R = D
- when WE = 0, latch holds previous value
  - S = R = 1



Register

A register stores a multi-bit value.

- We use a collection of D-latches, all controlled by a common WE.
- When WE=1, n-bit value D is written to register.

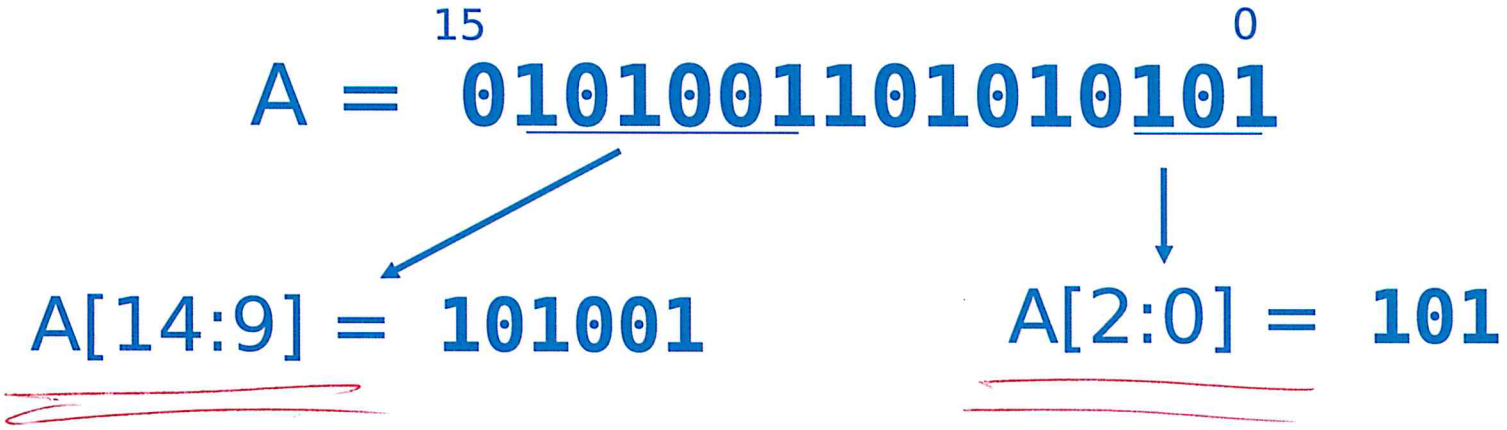


Representing Multi-bit Values

Number bits from right (0) to left (n-1)

- **just a convention -- could be left to right, but must be *consistent***

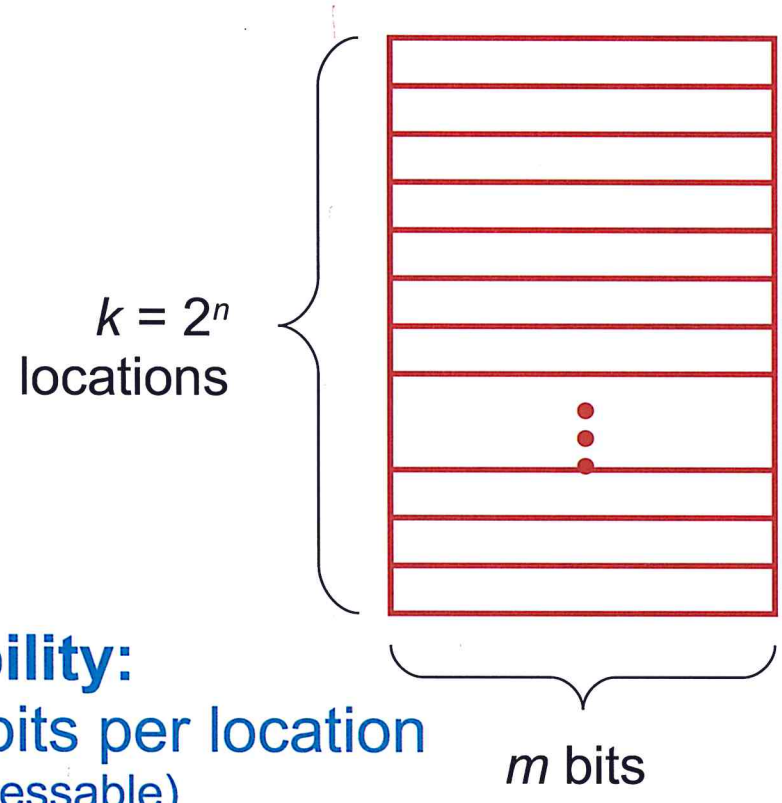
Use brackets to denote range:
D[l:r] **denotes bit** l **to bit** r**, from *left* to *right***



May also see A<14:9>**,**
especially in hardware block diagrams.

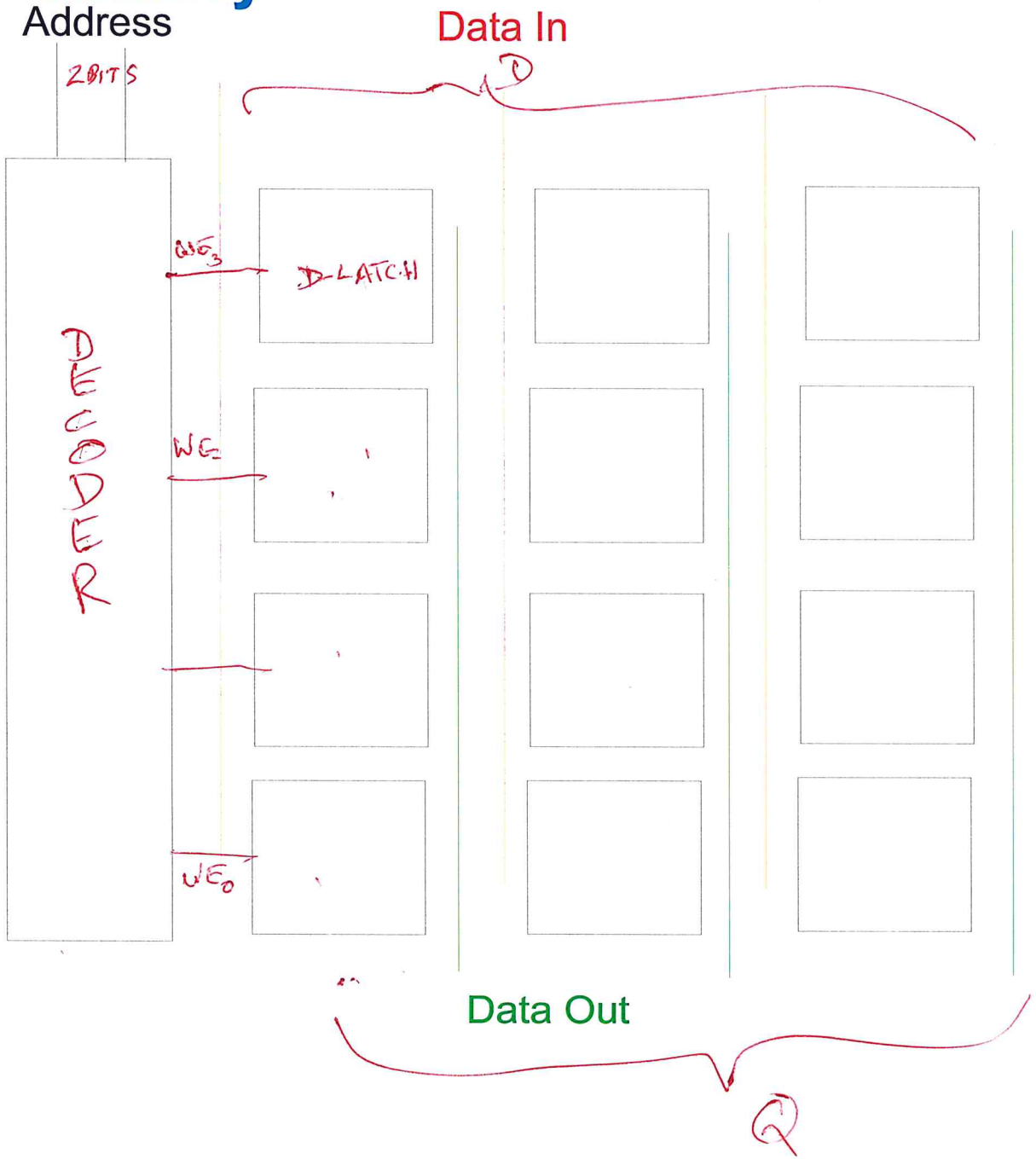
Memory

Address Space:
number of locations
(usually a power of 2)

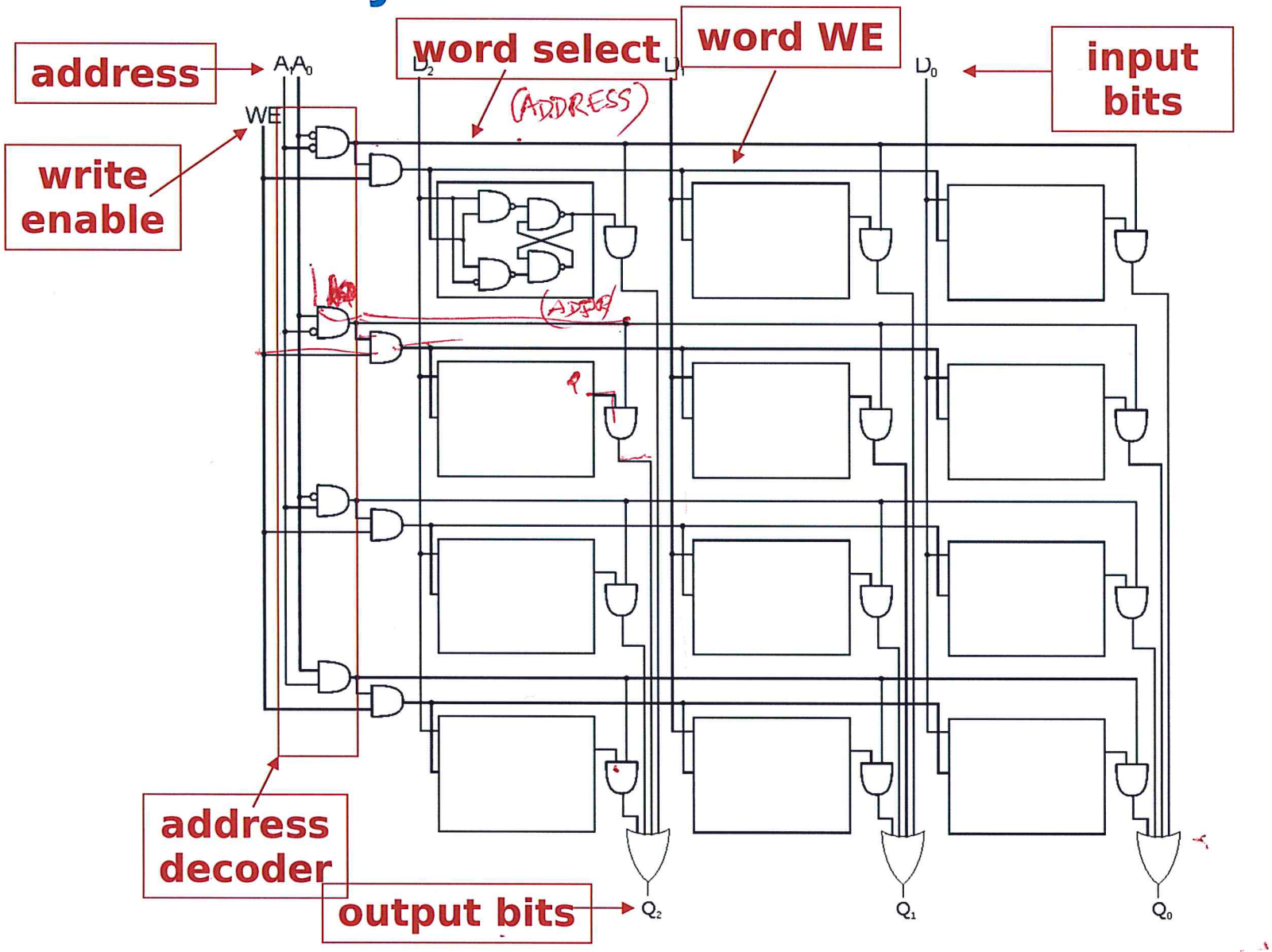


Addressability:
number of bits per location
(e.g., byte-addressable)

2^2 x 3 Memory

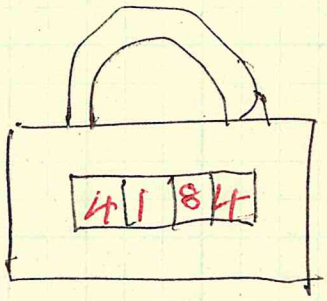


2^2 x 3 Memory

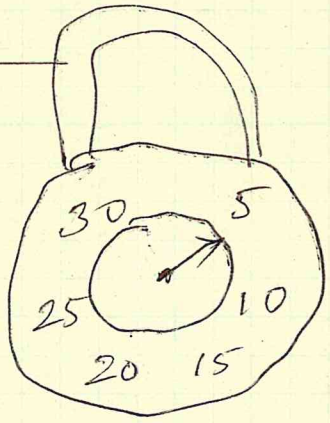

address
write enable
word select
word WE
input bits
address decoder
output bits
A
A0
WE
D2
D1
D0
Q2
Q1
Q0

SEQUENTIAL LOGIC

EXAMPLE: LOCK



COMBINATIONAL



SEQUENTIAL

EX: R-13, L-22, R-3

STATE:

"SNAPSHOT" OF THE RELEVANT ELEMENTS OF A SYSTEM

State of Sequential Lock

Our lock example has four different states, labelled A-D:

A: The lock is not open, and no relevant operations have been performed.

B: The lock is not open, and the user has completed the R-13 operation.

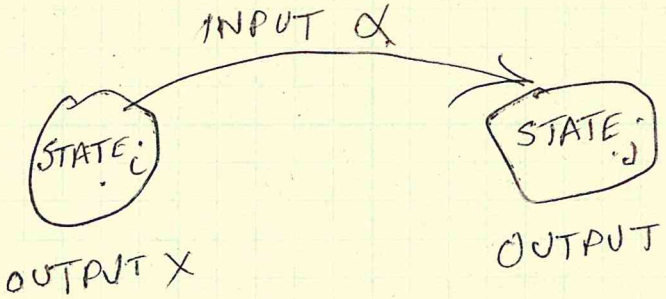
C: The lock is not open, and the user has completed R-13, followed by L-22.

D: User has completed R-13, followed by L-22, followed by R-3: the lock is open.

FINITE-STATE MACHINES

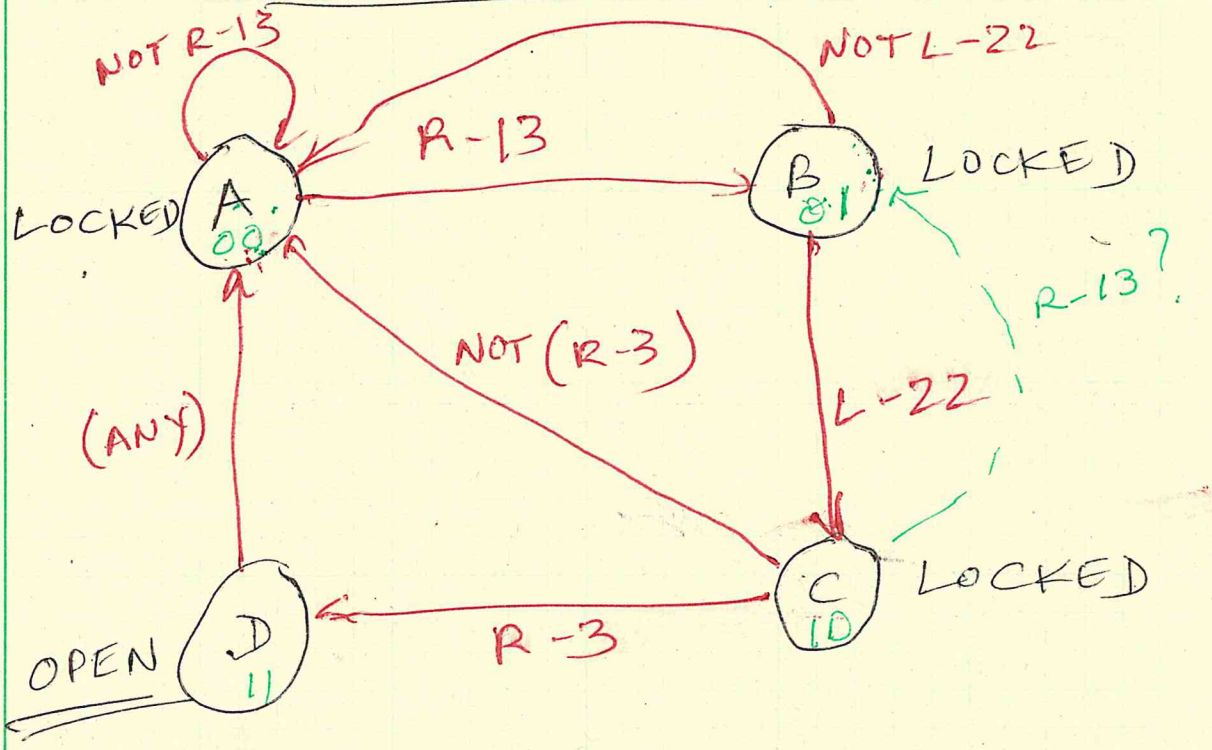
1. FINITE NUMBER OF STATES
2. ~~INPUTS~~ EXTERNAL INPUTS
3. OUTPUTS
4. SPECIFIED STATE TRANSITIONS
5. SPECIFIED OUTPUT VALUES

"STATE DIAGRAM"



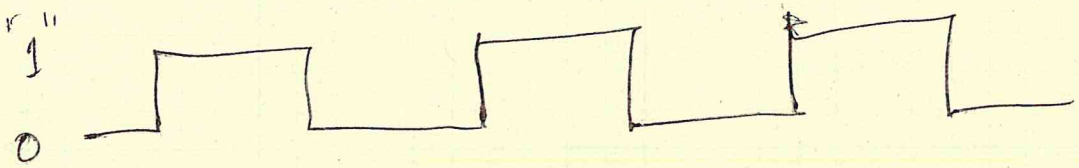
OR, "STATE TABLE"

STATE DIAGRAM FOR LOCK



TYPICALLY,

CLOCK TRIGGERS TRANSITIONS FROM ONE STATE TO THE NEXT



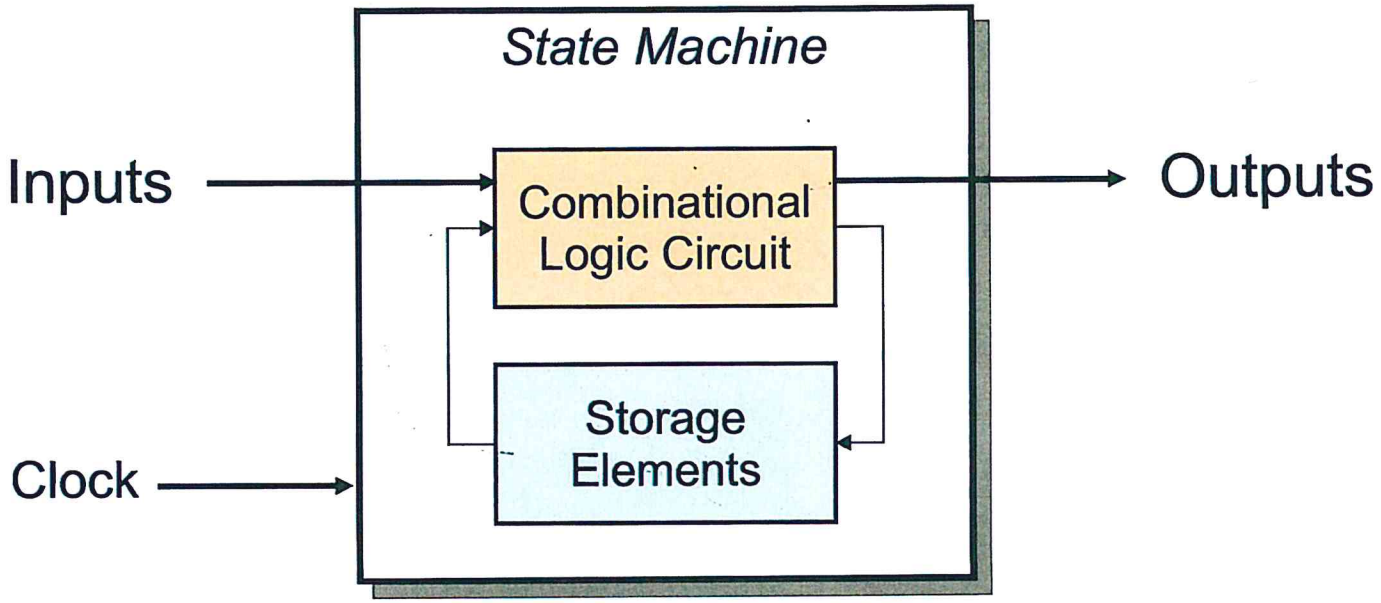
Implementing a Finite State Machine

Combinational logic

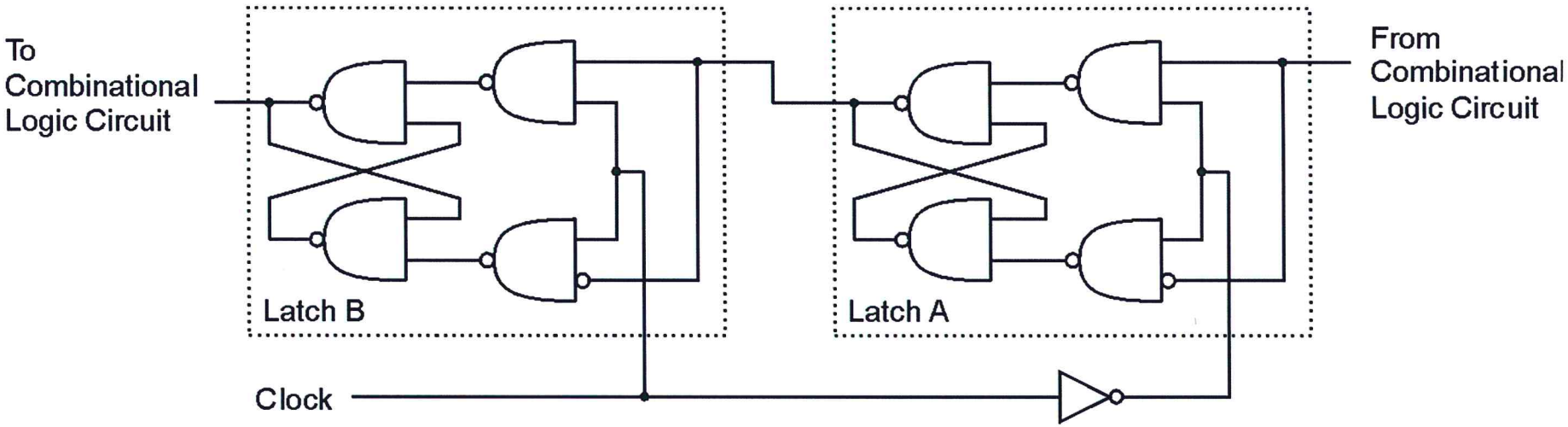
- **Determine outputs and next state.**

Storage elements

- **Maintain state representation.**



Storage: Master-Slave Flipflop



During 1st phase (clock=1), previously-computed state becomes *current* state and is sent to the logic circuit.

During 2nd phase (clock=0), *next* state, computed by logic circuit, is stored in Latch A.