7. The Stored-Program (von Neumann) Computer
Sept. 24, 2018
(Chapter 4)

- Review
  - Finite-State Machines
  - Example: design of control for a warning sign

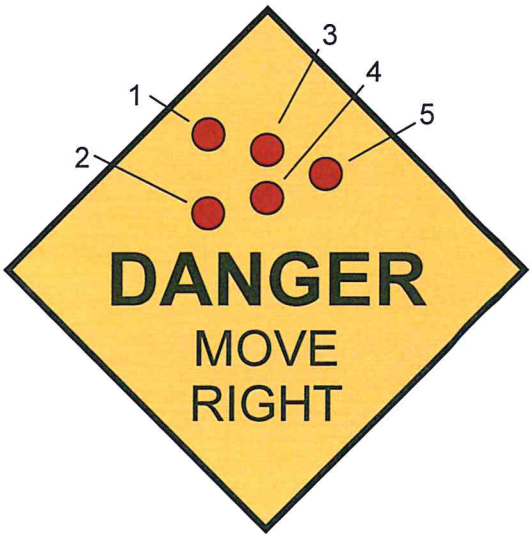
- LC-3 data path

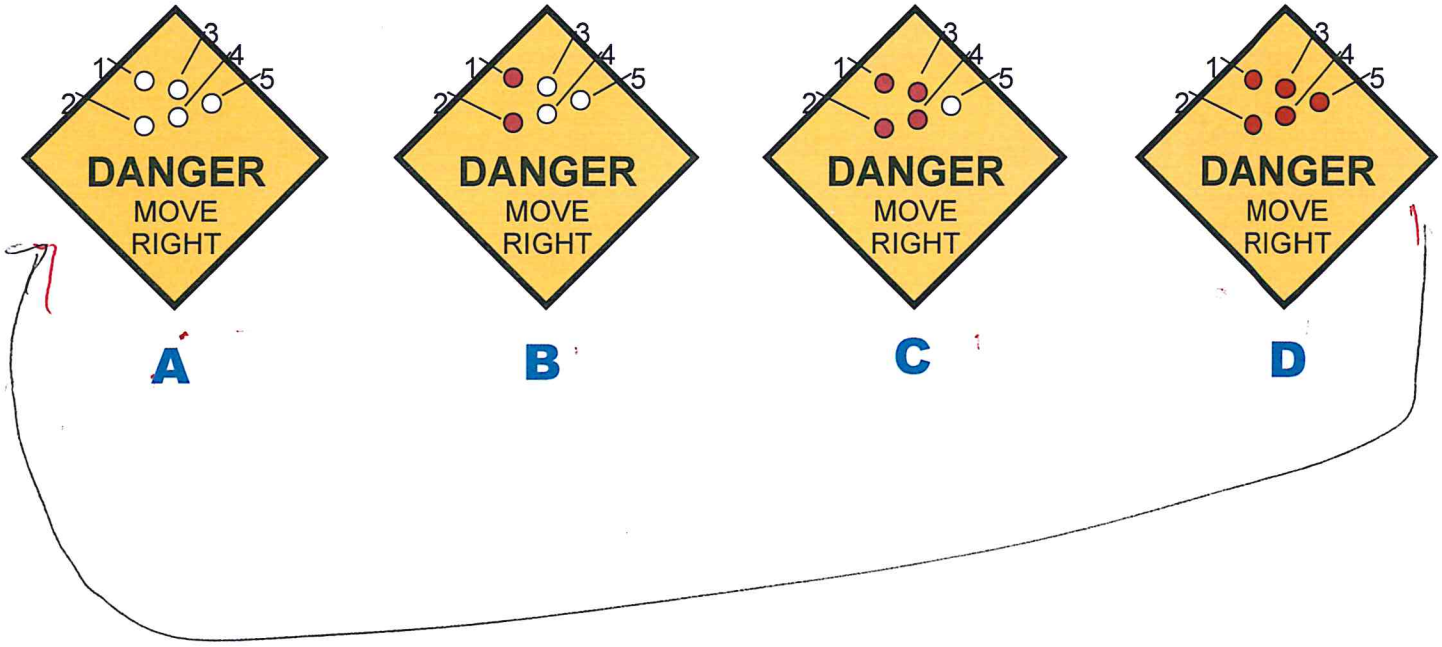
- The stored program computer
  - Interface to memory
  - Input/Output
  - Control
  - Example instructions

Review:
Complete Example

A blinking traffic sign

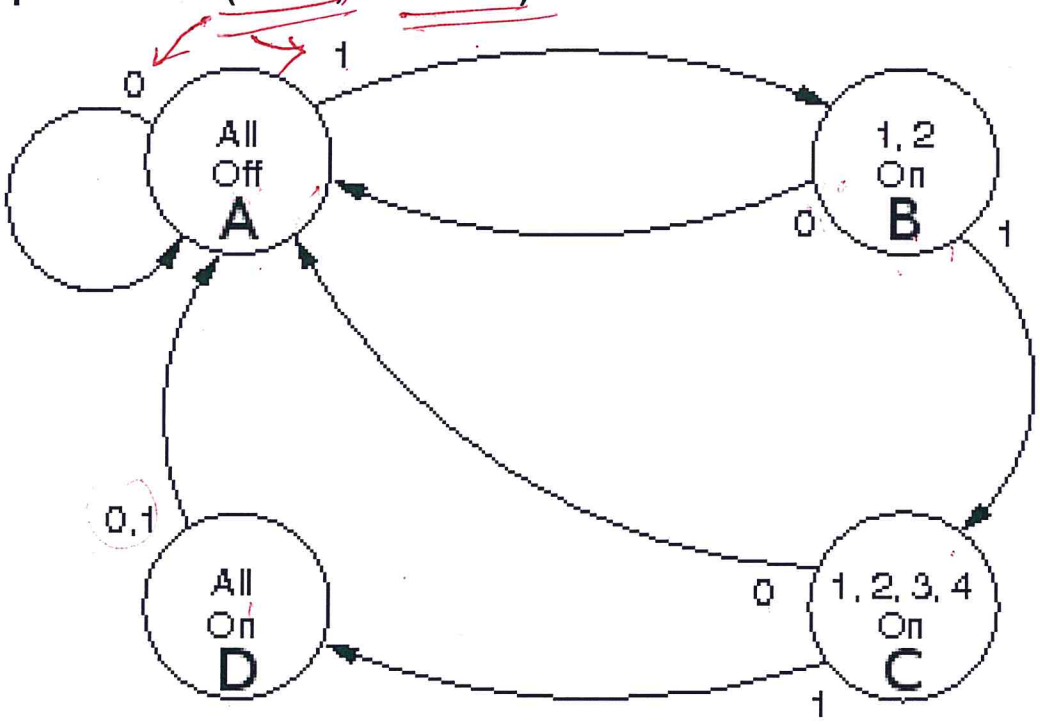
- A • No lights on
- B • 1 & 2 on
- C • 1, 2, 3, & 4 on
- D • 1, 2, 3, 4, & 5 on
- • (repeat as long as switch is turned on)





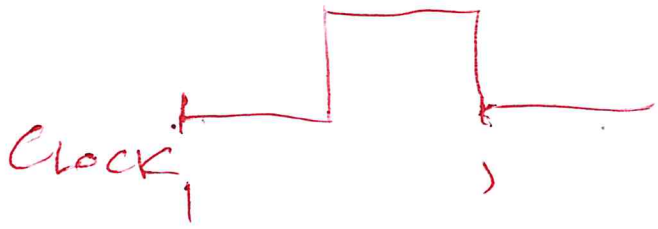
Finite-State Machine (FSM) for Controller

Input: Switch (ON=1, or OFF=0)

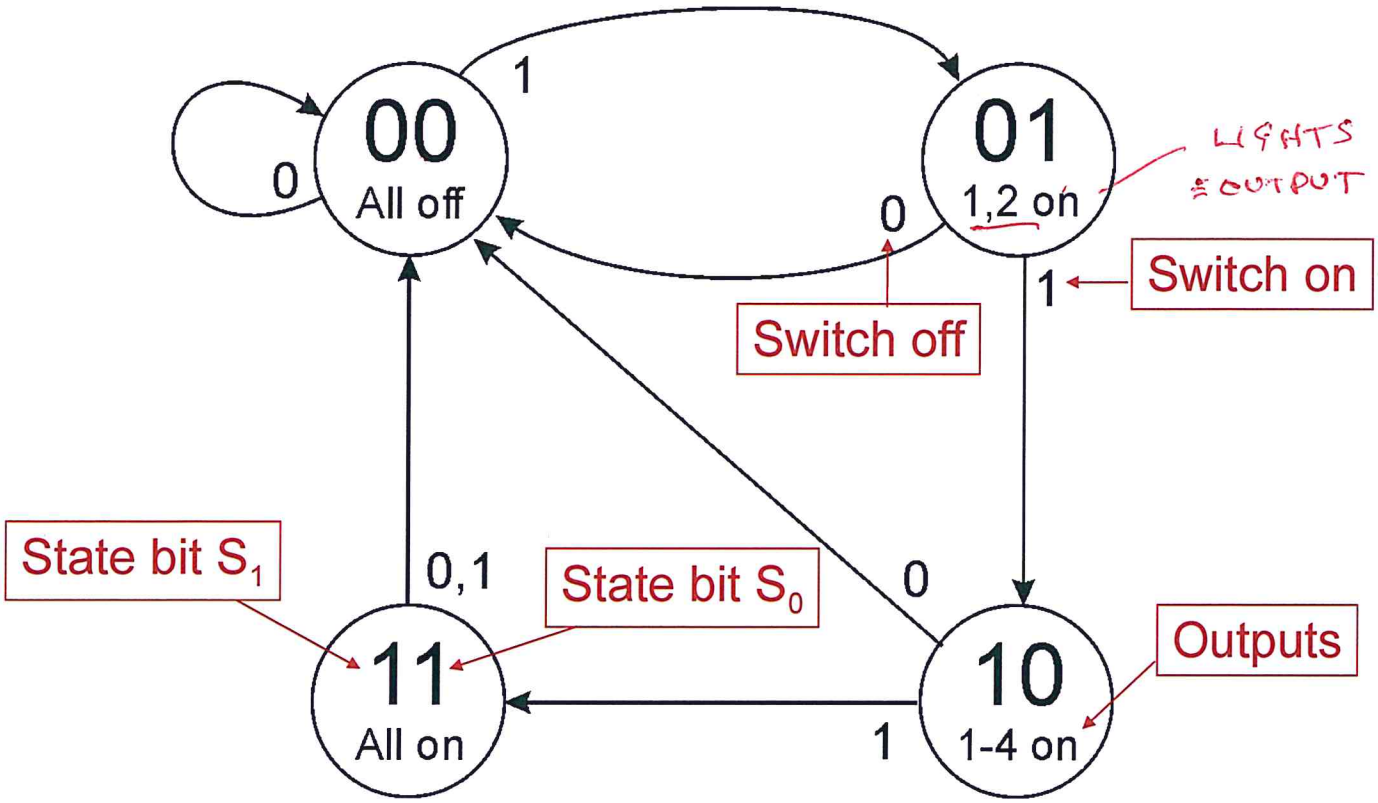


Transition from one state to next when Clock becomes 1

ON EACH CLOCK CYCLE



Traffic Sign State Diagram – with assignment of storage element values to states



Transition on each clock cycle.

ANOTHER ASSIGNMENT

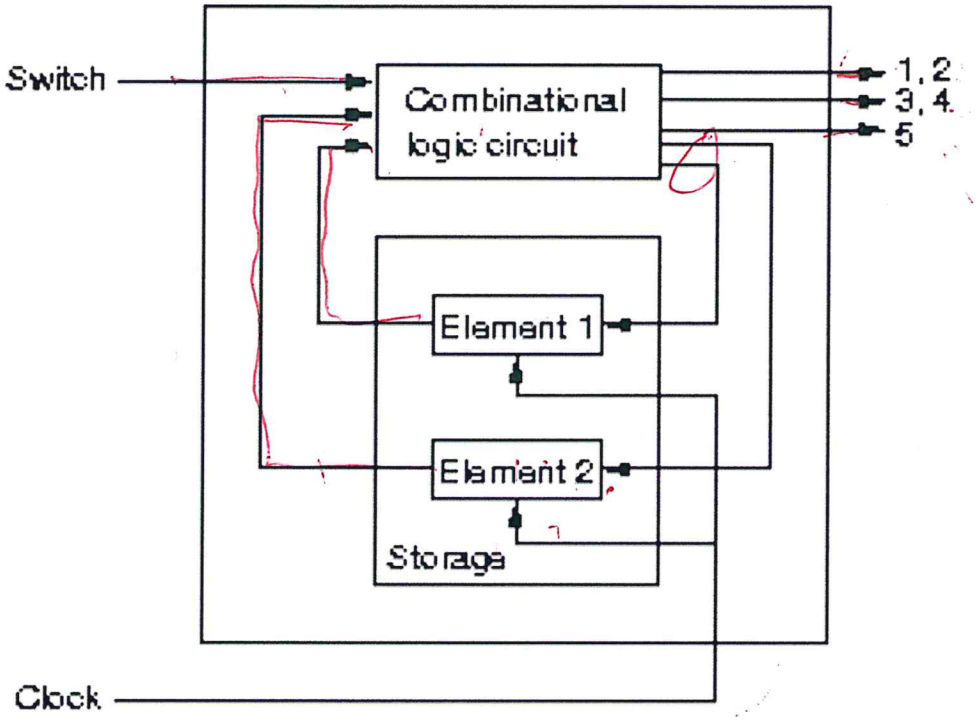
A: 0001

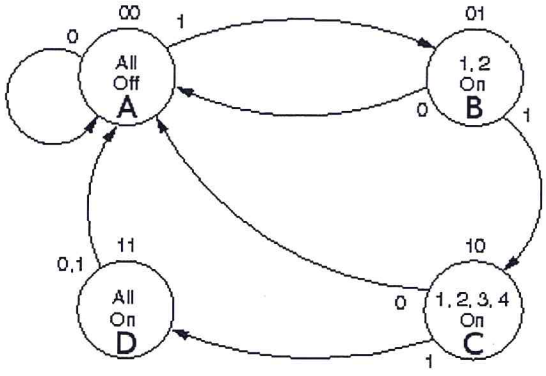
B: 0010

C: 0100

D: 1000

Block Diagram of Finite-State Controller





State Table for FSM

INPUTS

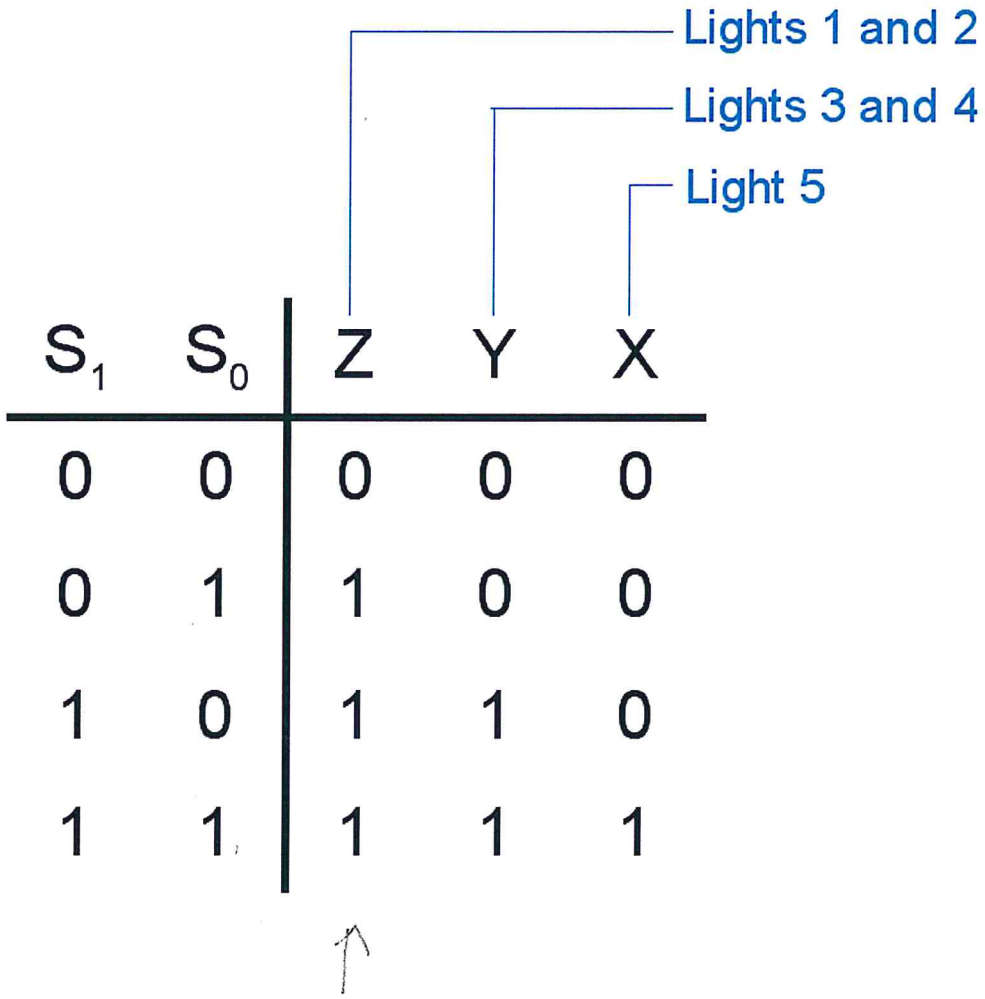
S_1S_0	SW OFF	SW ON	1,2	3,4	5
0 0	0 0	0 1	0	0	0
0 1	0 0	1 0	1	0	0
1 0	0 0	1 1	1	1	0
1 1	0 0	0 0	1	1	1

PRESENT STATE

NEXT STATE

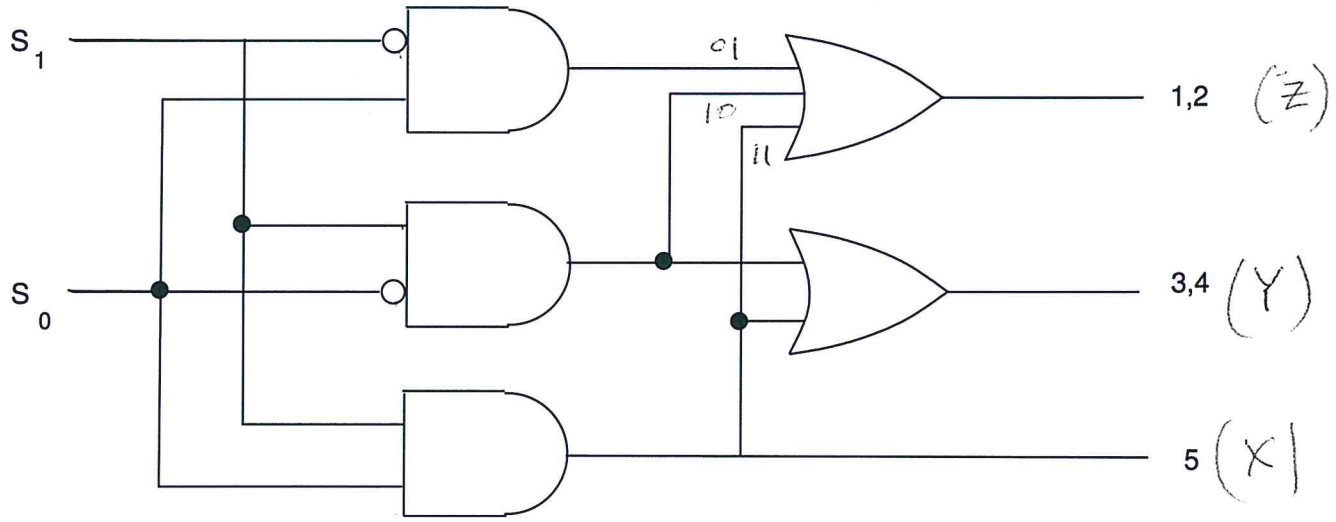
OUTPUTS

Truth Table for Lights



S_1	S_0	Z	Y	X
0	0	0	0	0
0	1	1	0	0
1	0	1	1	0
1	1	1	1	1

Logic Circuit for Lights



Truth Table for Next State Function

Next State: $S_1^+ S_0^+$
(depend on state and input)

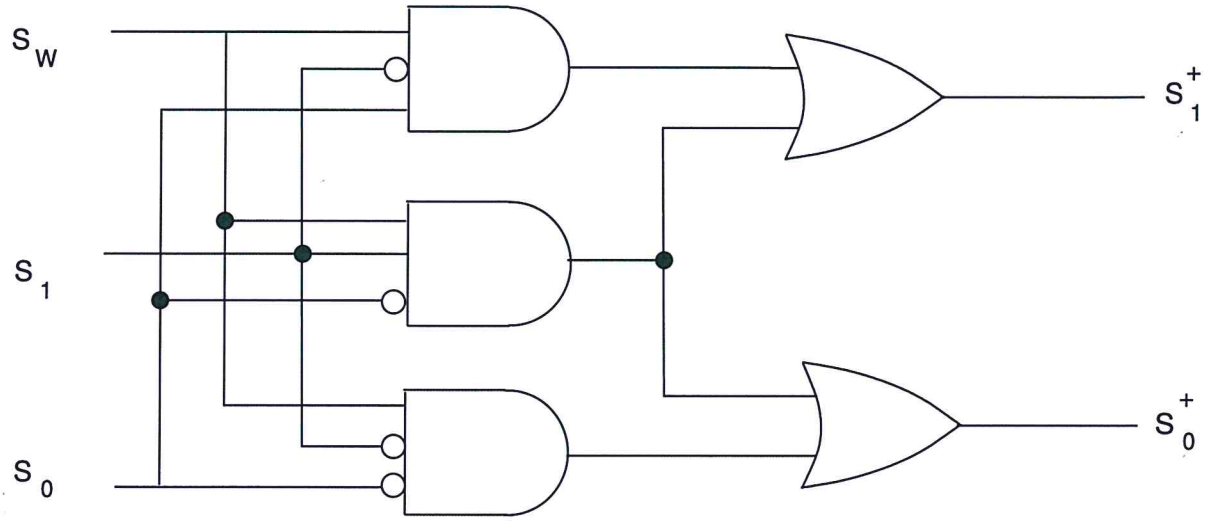
Switch

In	S_1	S_0	S_1^+	S_0^+
0	X	X	0	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	1
1	1	1	0	0

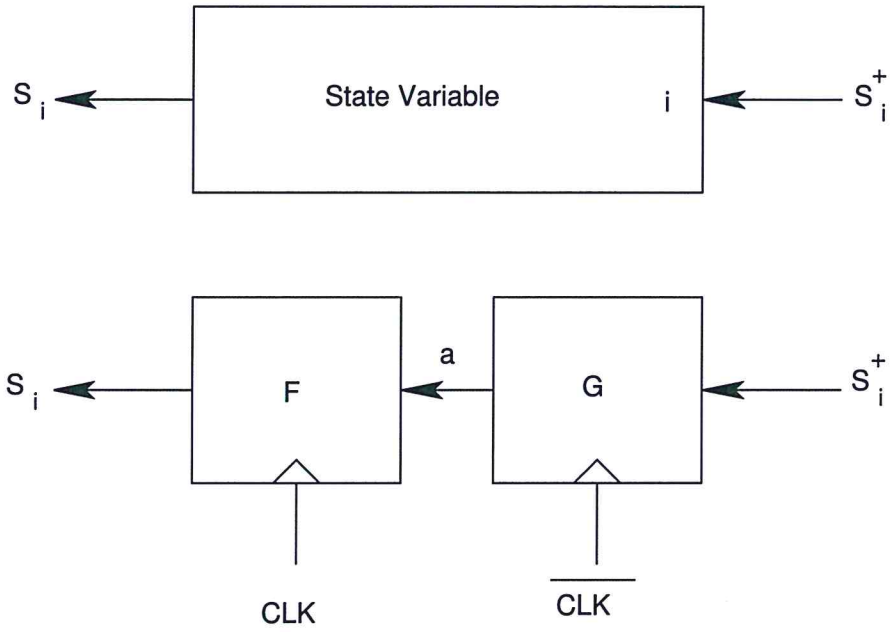
Whenever In=0, next state is 00.

X: "DON'T CARE"

Logic Circuit for Next-State Function

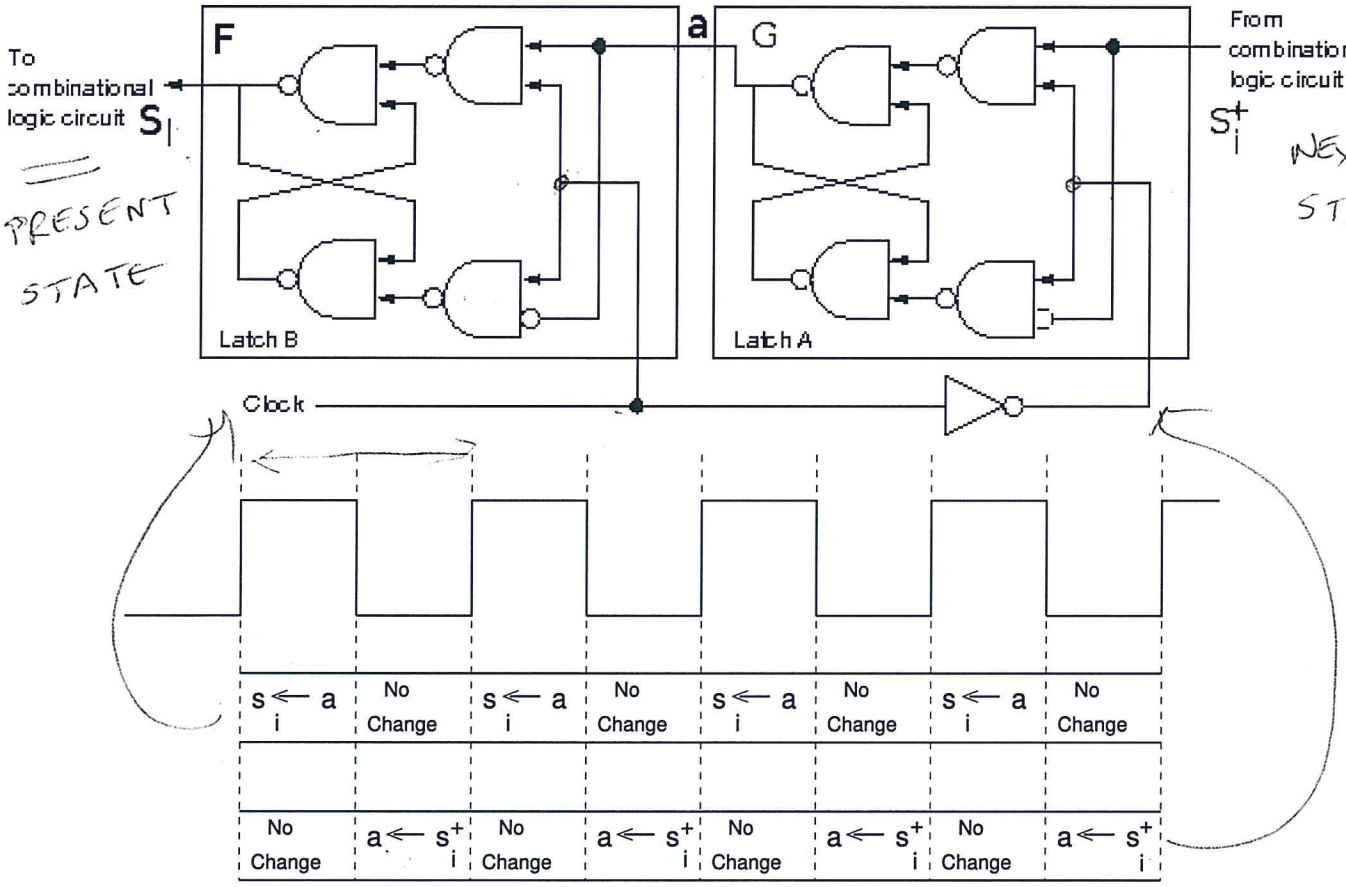


Sequencing the FSM

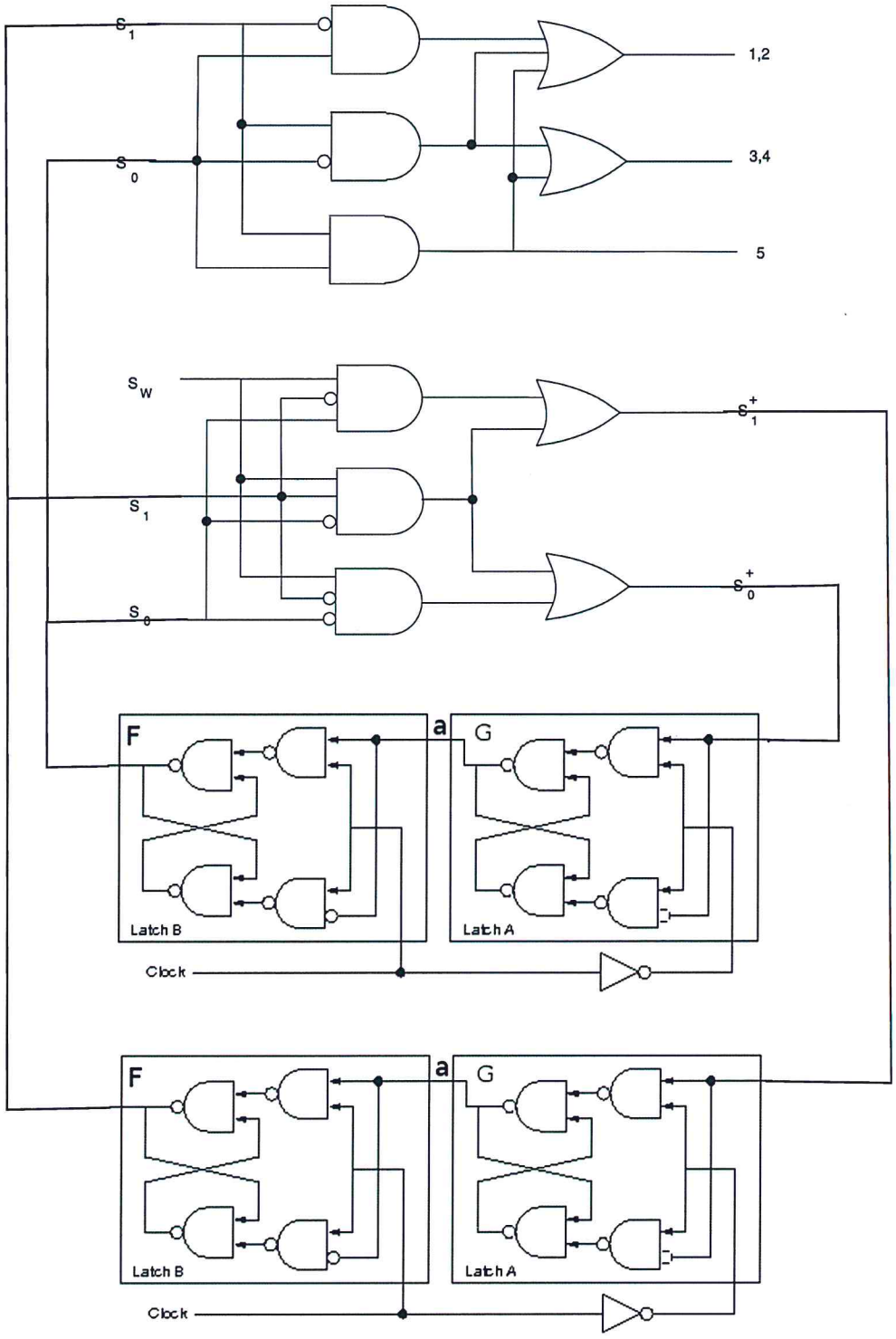


Clocking

MASTER-SLAVE FF (FLIP-FLOP)



"Simulating" the Traffic Sign Logic



The Stored Program Computer

1943: ENIAC

- **Presper Eckert and John Mauchly -- first general electronic computer.**
  (or was it John V. Atanasoff in 1939? – special-purpose electronic computer)
- **Hard-wired program -- settings of dials and switches.**

1944: Beginnings of EDVAC

- **among other improvements, includes program stored in memory**

1945: John von Neumann

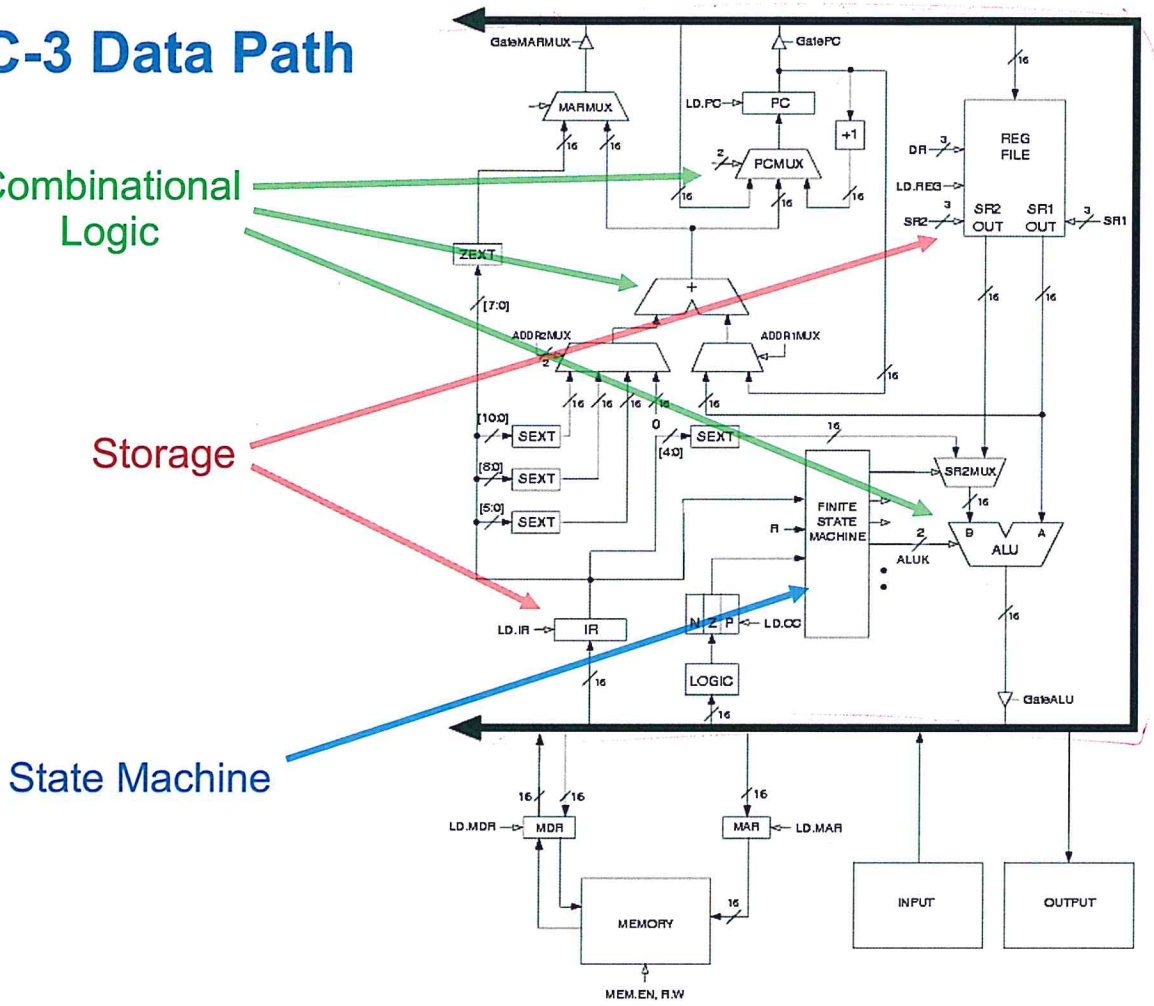
- **wrote a report on the stored program concept, known as the *First Draft of a Report on EDVAC***

The basic structure proposed in the draft became known as the "von Neumann machine" (or model).

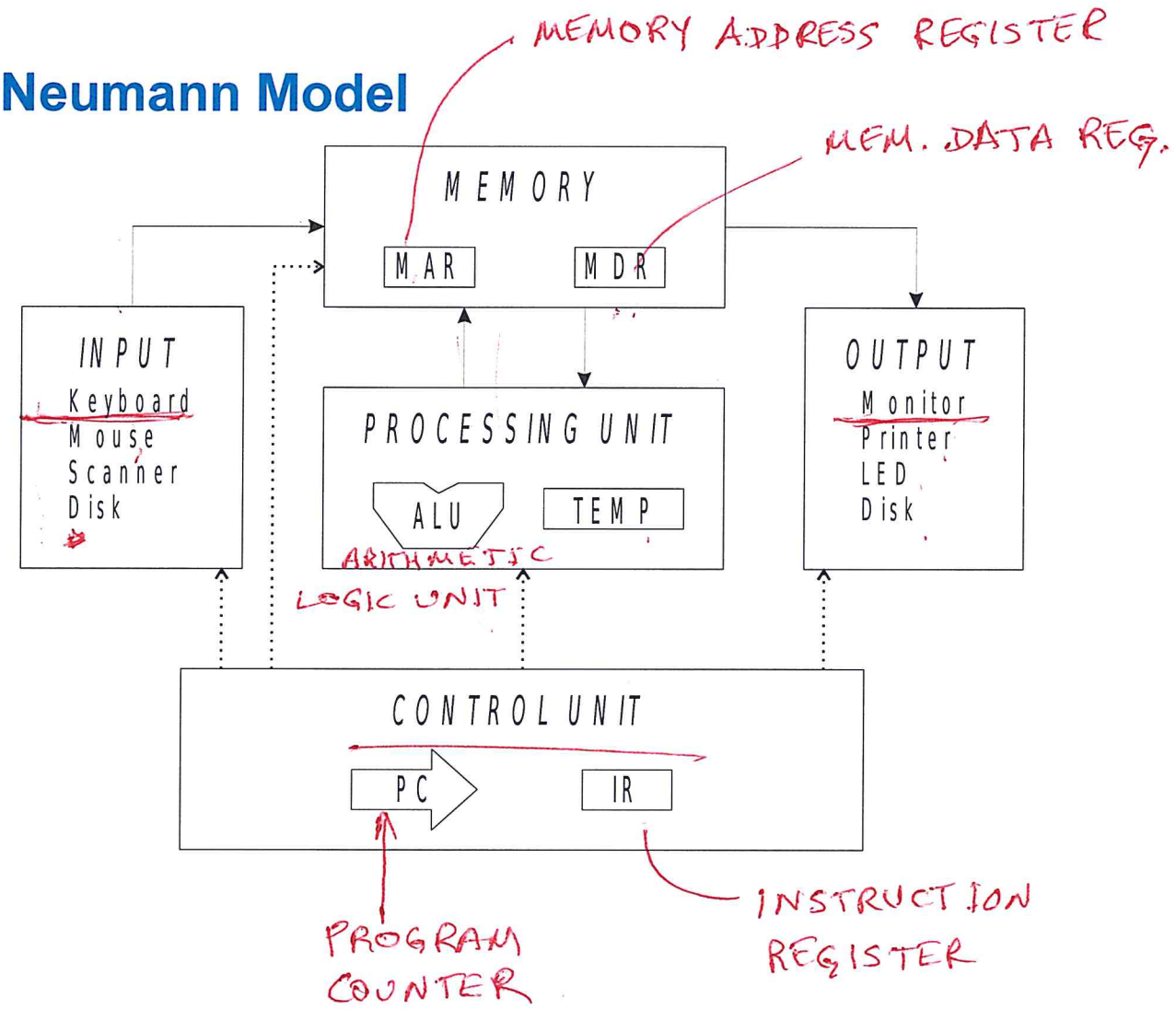
- **a *memory*, containing instructions and data**
- **a *processing unit*, for performing arithmetic and logical operations**
- **a *control unit*, for interpreting instructions**

For more history, see http://www.computerhistory.org/

LC-3 Data Path



Von Neumann Model



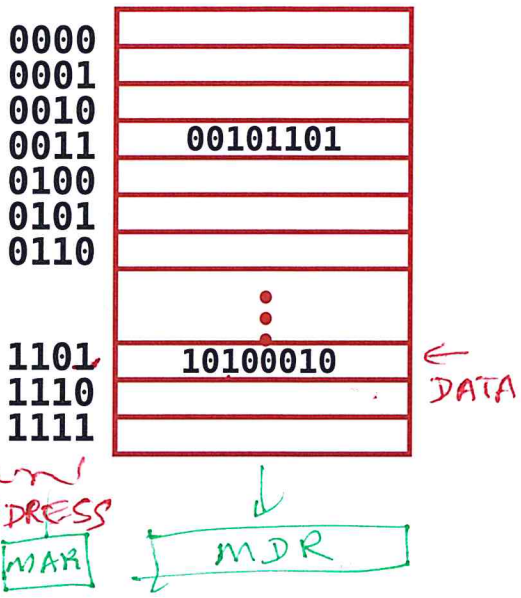
Memory

ARRAY

2^4 WORDS X 8 BITS

ADDRESS: 4 BITS

CONTENTS (DATA): 8 BITS



OPERATIONS

LOAD (READ): PUT ADDRESS IN MAR
SEND A "READ" COMMAND
GET DATA FROM MDR.

STORE (WRITE): WRITE DATA INTO MDR
PUT ADDRESS INTO MAR
SEND A "WRITE" SIGNAL TO MEMORY

PROCESSING UNIT

FUNCTION UNITS

↳ ALU

LC3: ADD
AND, NOT

REGISTERS

LC3: TEMP STORAGE (16 BIT)
8 REGISTERS, 16 BITS
$R_7, \ldots, R_0$

I/O

LC3: KEYBOARD INPUT

KBDR (DATA)

KBSR (STATUS)

MONITOR OUTPUT

DDR ("DISPLAY")

DSR

CONTROL

IR - INST. REG.
PC - PROG. COUNTER

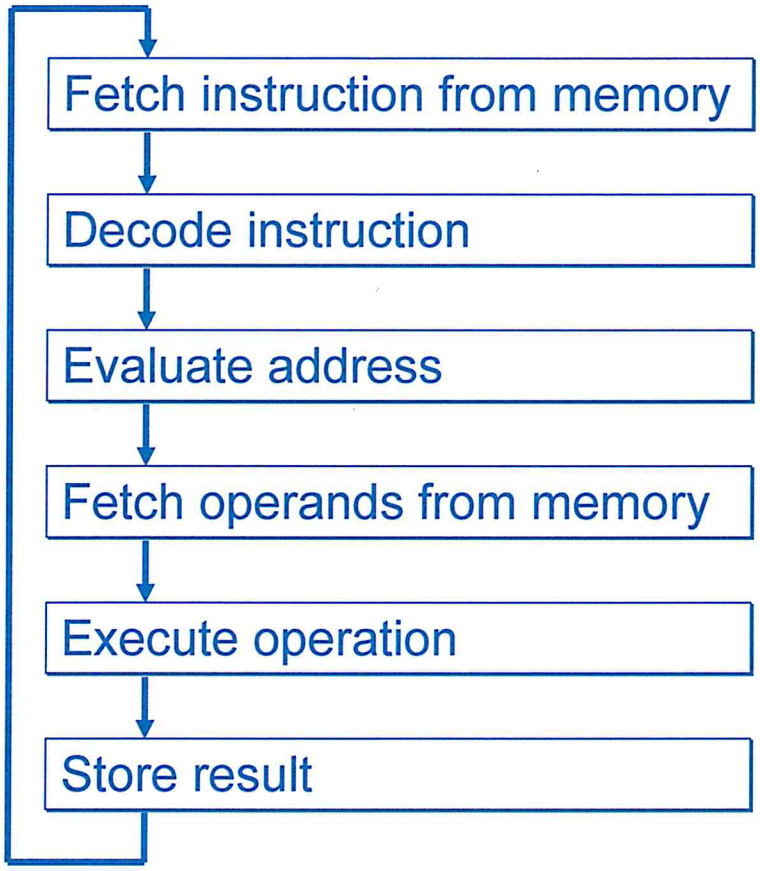
NEXT INSTRUCTION:

→ ADDRESS IS IN PC

→ EXECUTES INSTR. AT THAT ADDRESS

⟹ MANY CYCLES

Instruction Processing



ADDR : IN PC

Instruction Processing: FETCH

DATA IN

LOAD ^ ADDRESS POINTED TO BY PC.

INTO IR.

- MAR ← (PC)
- SEND "READ" TO MEMORY
- COPY CONTENTS INTO IR

  IR ← (MDR)
- PC ← PC + 1