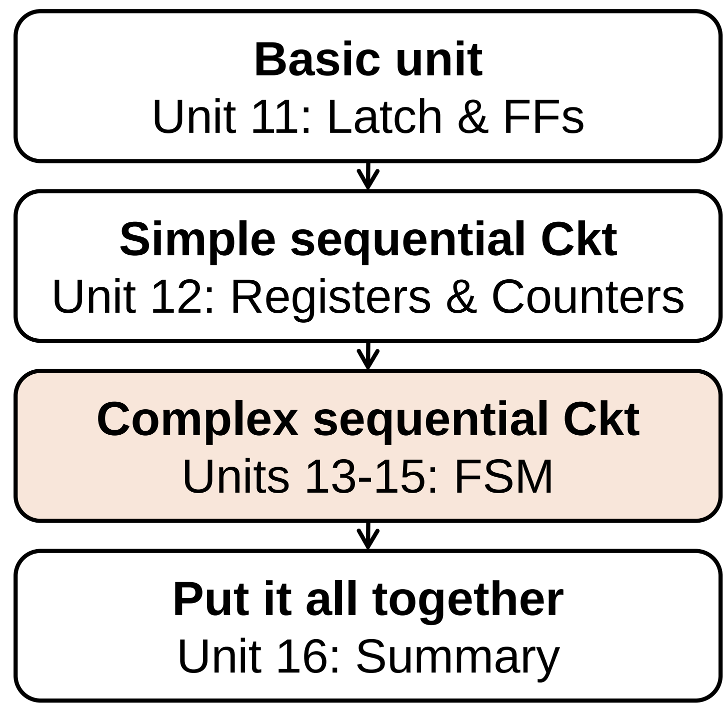
UNIT 14
DERIVATION OF STATE GRAPHS AND TABLES

Fall 2021

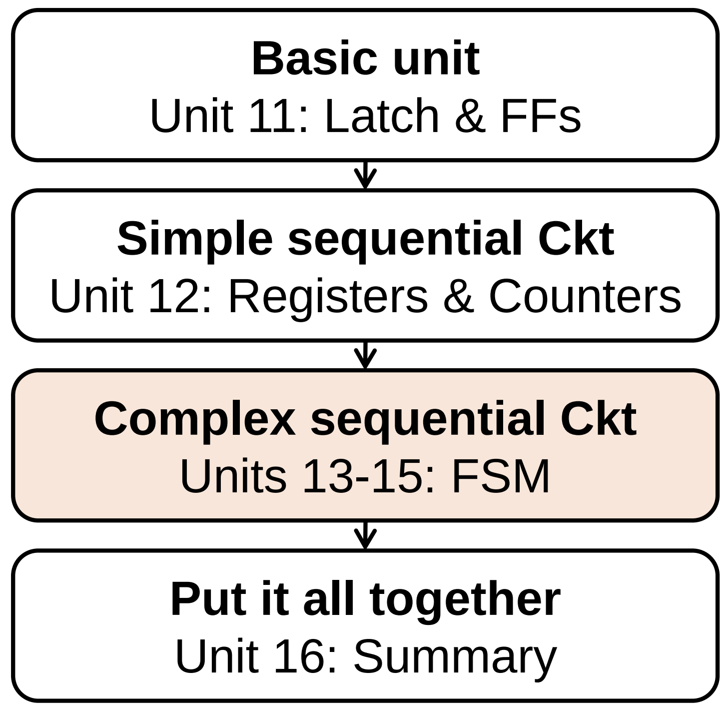
Derivation of State Graphs and Tables

- **Contents**
  - Case studies: sequence detectors
  - Guidelines for construction of state graphs
  - Serial data code conversion
  - Alphanumeric state graph notation
- **Reading**
  - Unit 14



Designing a Sequential Circuit

- **Given the specification of a sequential circuit**
- **Design procedure:**
  1. Construct a state table or state graph (Unit 14)
  2. Simplify (Unit 15)
  3. Derive FF input equations and output equations (Unit 12)



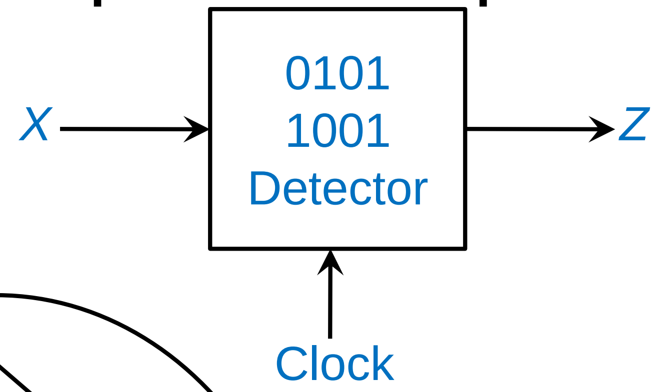
4

Sequence Detectors

Derivation of state graphs & tables

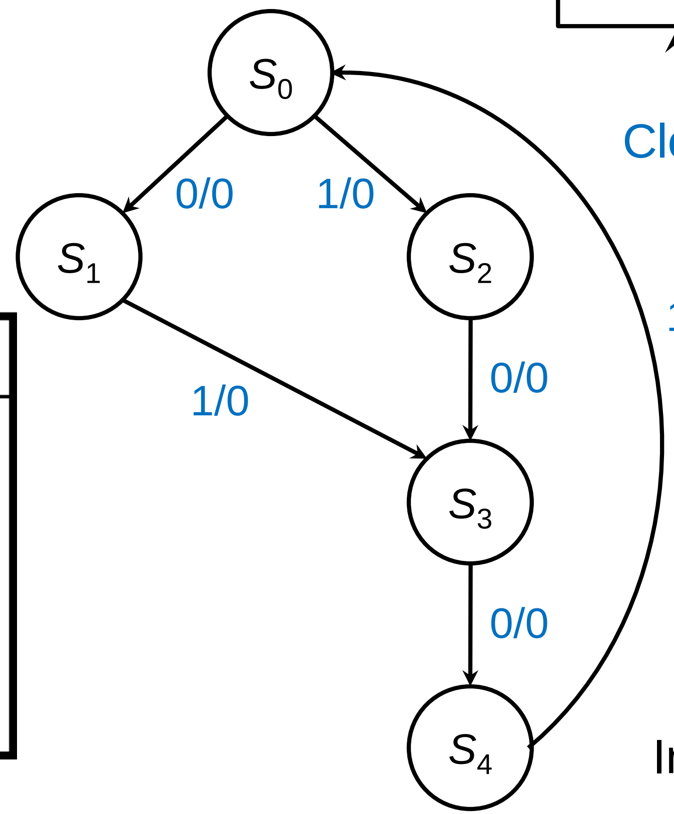
Case I (1/2)

- **Examine groups of 4 consecutive inputs & produce an output**
- **Reset after every 4 inputs**
  - e.g., X = 0101 0010 1001 0100
    Z = 0001 0000 0001 0000



- **Observation:** X = 0101
  X = 1001
- **Typical sequence**
  - Partial state graph

State	Sequence received
S_0	Reset
S_1	0
S_2	1
S_3	01 or 10
S_4	010 or 100

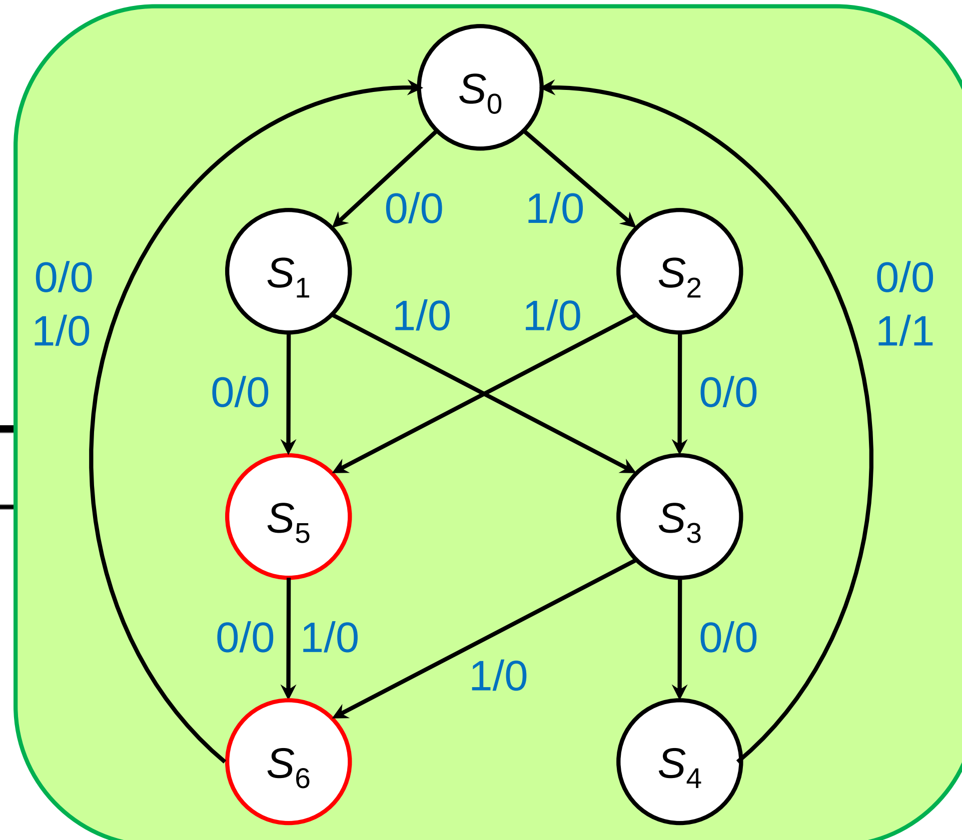


Input/output

Case I (2/2)

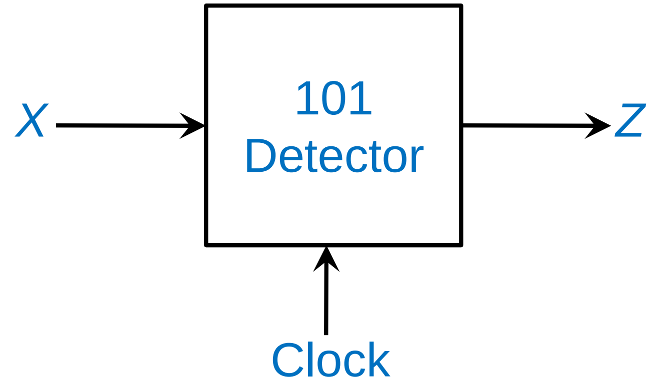
- Complete state graph

State	Sequence received
S_0	Reset
S_1	0
S_2	1
S_3	01 or 10
S_4	010 or 100
S_5	Two input received, no 1 output is possible
S_6	Three input received, no 1 output is possible

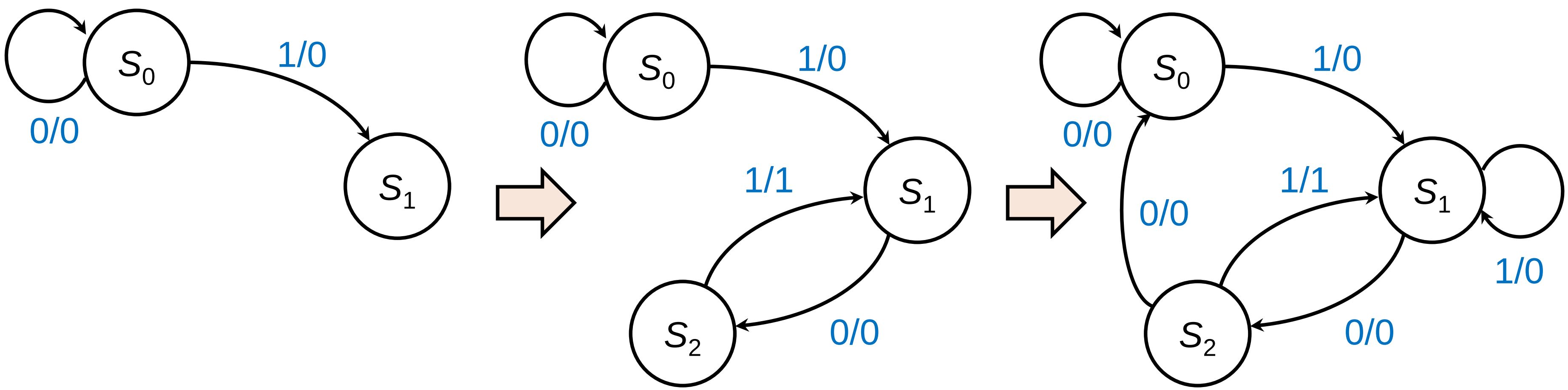


Case II (1/4)

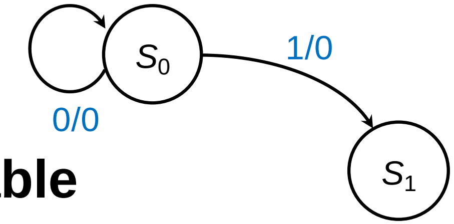
- **Examine groups of 3 consecutive inputs & produce an output**
- **No reset**
  - e.g., X = 0 0 1 1 0 1 1 0 0 1 0 1 0 1 0 0
    Z = 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0



- **State graph (Mealy)**
  - S_0: initial, S_1: get …1, S_2: get …10

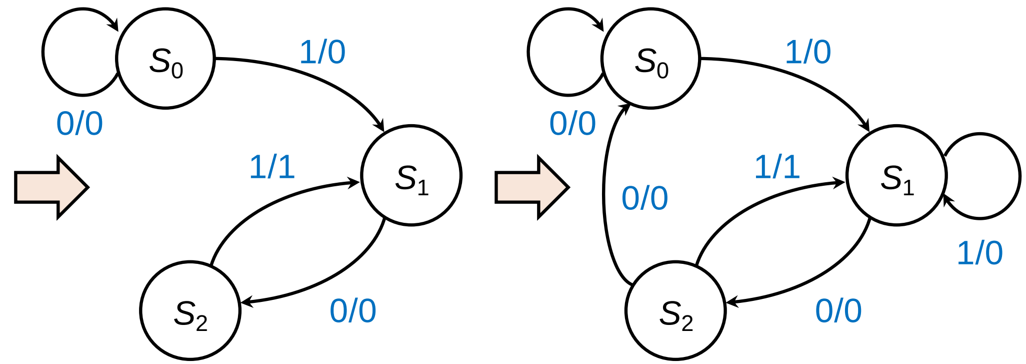


Case II (2/4)



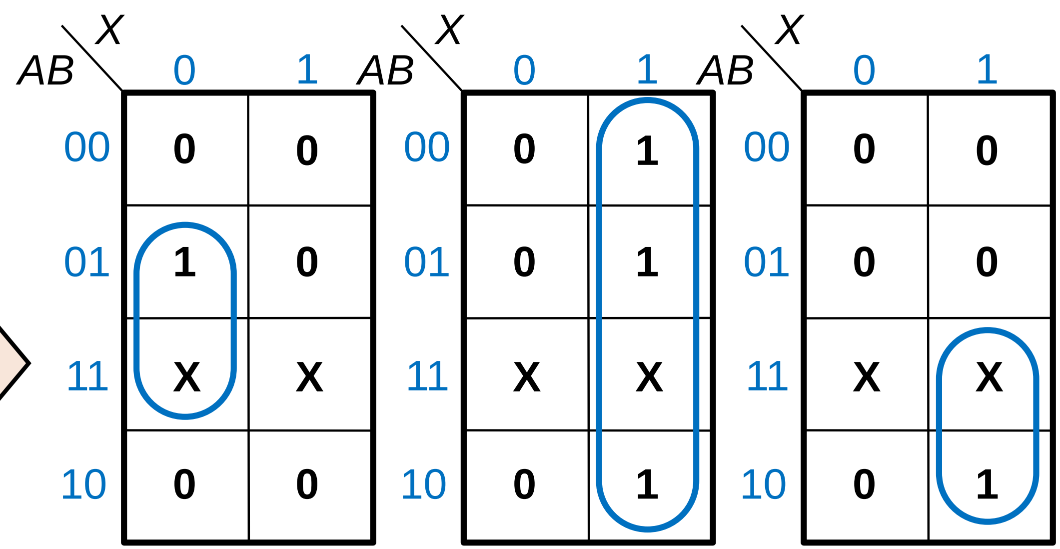
- **State table**

Present state	Next state $X=0$	Next state $X=1$	Present output $X=0$	Present output $X=1$
S_0	S_0	S_1	0	0
S_1	S_2	S_1	0	0
S_2	S_0	S_1	0	1



AB	A^+B^+ $X=0$	A^+B^+ $X=1$	Z $X=0$	Z $X=1$
00	00	01	0	0
01	10	01	0	0
11	XX	XX	X	X
10	00	01	0	1

- **State maps**



$A^+=X'B$ $B^+=X$ $Z=XA$

Case II (3/4)

- State maps

AB \ X	0	1
00	0	0
01	1	0
11	X	X
10	0	0

$A^+ = X'B$

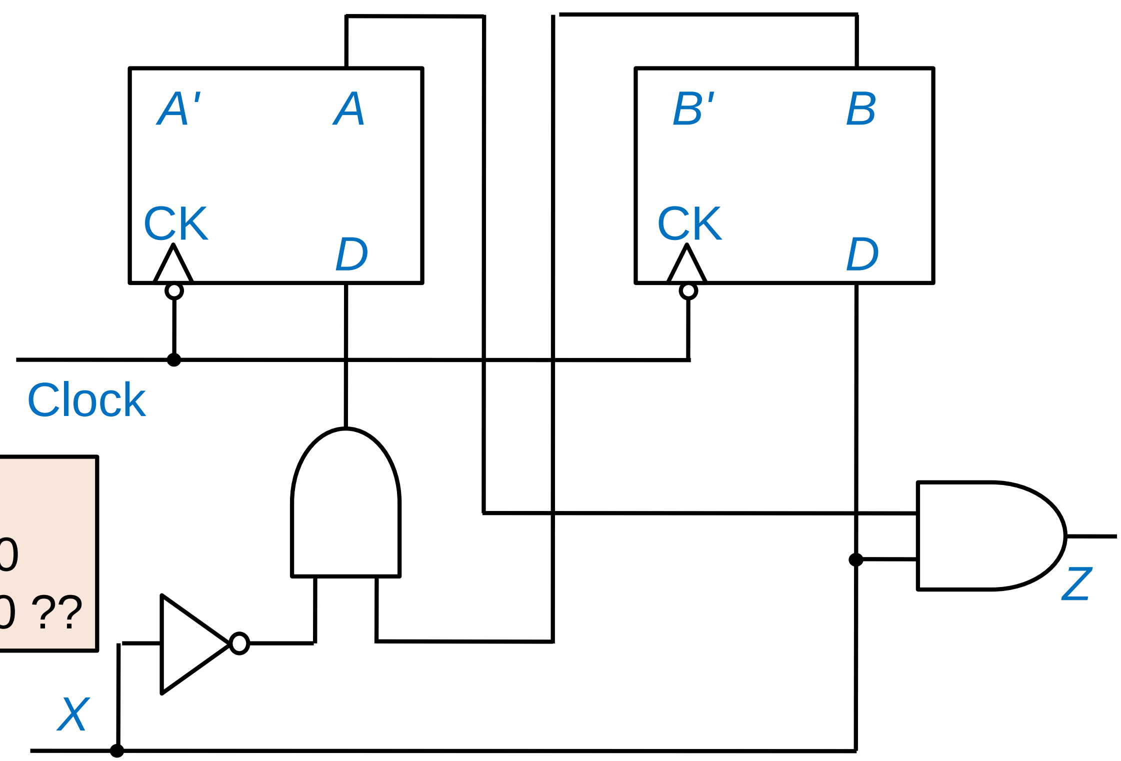
AB \ X	0	1
00	0	1
01	0	1
11	X	X
10	0	1

$B^+ = X$

AB \ X	0	1
00	0	0
01	0	0
11	X	X
10	0	1

$Z = XA$

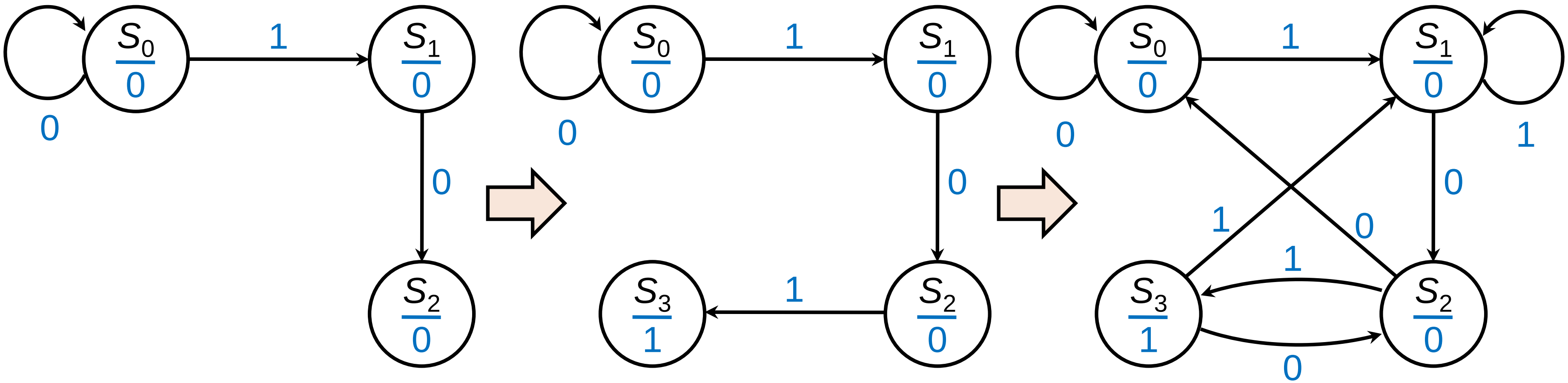
- Realize it



Q: Check by yourself
X = 0 0 1 1 0 1 1 0 0 1 0 1 0 1 0 0
Z = 0 0 0 0 0 1 0 0 0 0 0 1 0 1 0 0 ??

Case II (4/4)

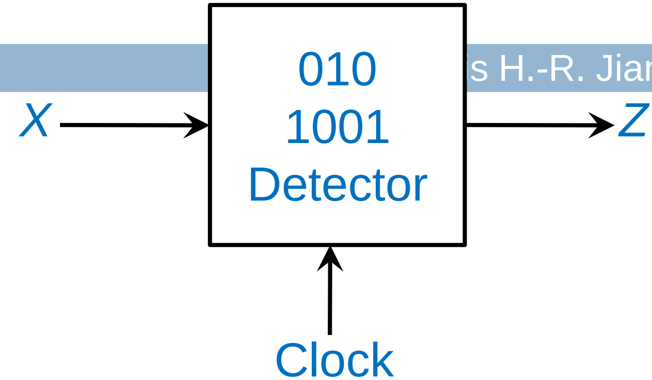
- **Moore?**
  - S_0: initial, S_1: get …1, S_2: get …10, S_3: get …101



Present state	Next state $X = 0$	Next state $X = 1$	Present output
S_0	S_0	S_1	0
S_1	S_2	S_1	0
S_2	S_0	S_3	0
S_3	S_2	S_1	1

AB	A^+B^+ $X = 0$	A^+B^+ $X = 1$	Z
00	00	01	0
01	10	01	0
10	00	11	0
11	10	01	1

Case III (1/2)



- **010 & 1001 detector**
  - e.g., X = 0 0 1 0 1 0 0 1 0 0 0 1 0 0 1 1

    a b c d e f

    Z = 0 0 0 1 0 1 0 1 1 0 0 0 1 0 1 0

- **State assignment**

State for "010"

State	Sequence received
S_0	Reset
S_1	0
S_2	01
S_3	010

+

State for "1001"

State	Sequence ends in
S_0	Reset
S_1	0(but not 10)
S_2	01
S_3	10
S_4	1(but not 01)
S_5	100

=

Complete

State	Sequence ends in
S_0	Reset
S_1	0(but not 10)
S_2	01
S_3	10
S_4	1(but not 01)
S_5	100

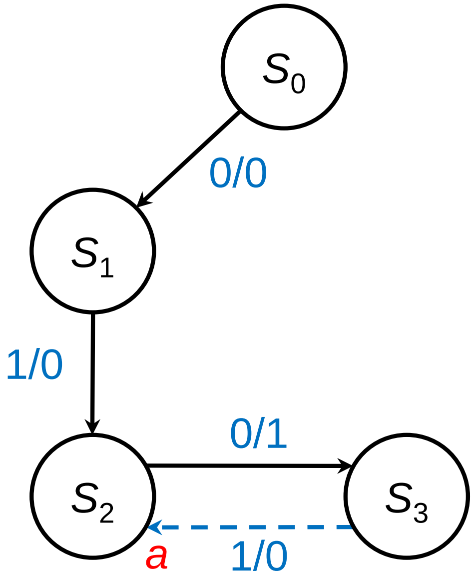
Case III (2/2)

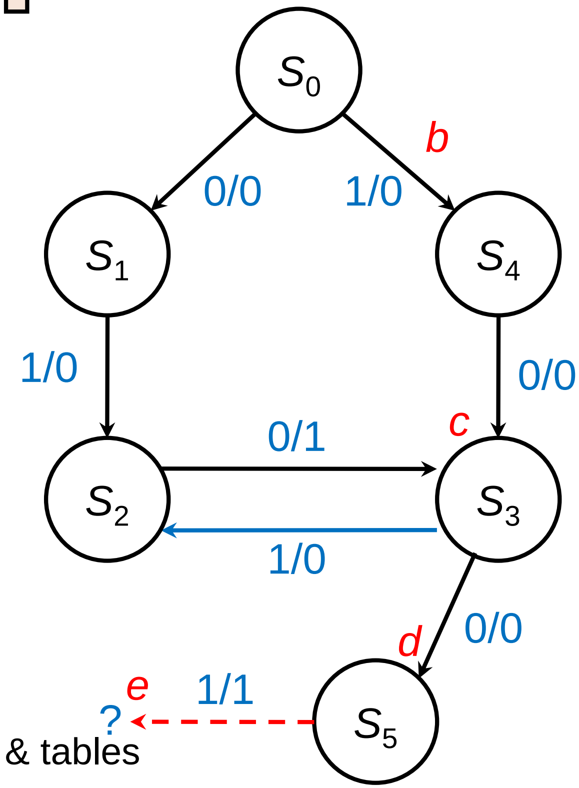
State	Sequence received
S_0	Reset
S_1	0
S_2	01
S_3	010

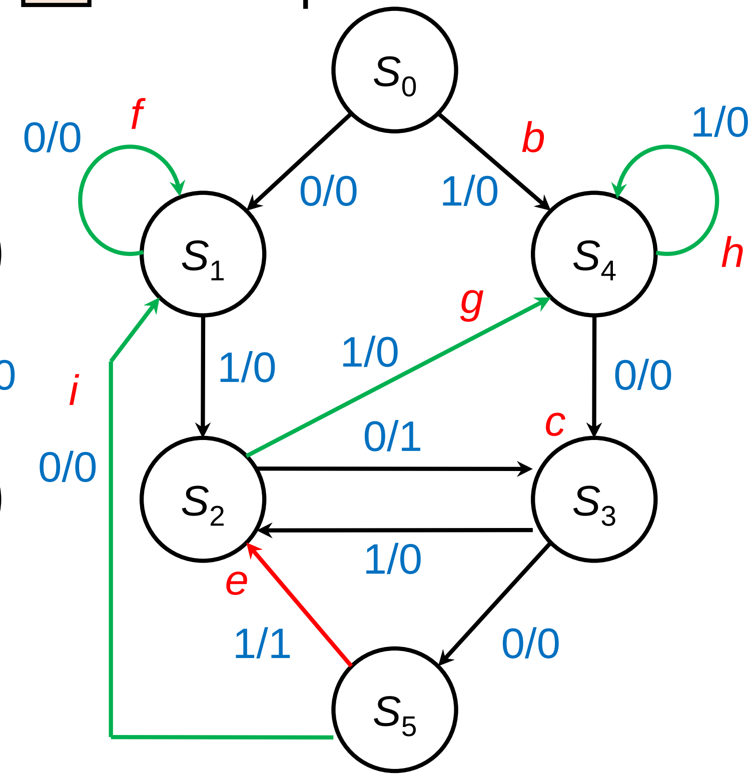
State	Sequence ends in
S_0	Reset
S_1	0(but not 10)
S_2	01
S_3	10
S_4	1(but not 01)
S_5	100

State	Sequence ends in
S_0	Reset
S_1	0(but not 10)
S_2	01
S_3	10
S_4	1(but not 01)
S_5	100

- State for "010" + State for "1001" = Complete

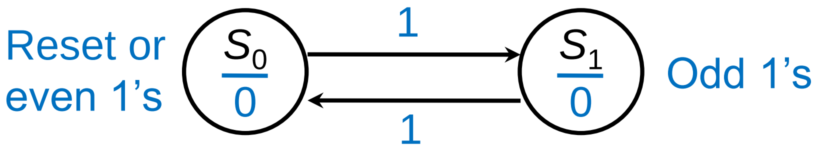






Case IV (1/2)

- **Specifications**
  - Z = 1 if total # of 1's is odd
    and
    at least two consecutive 0's have been received
  - e.g., X = 1 0 1 1 0 0 1 1
    Z =(0) 0 0 0 0 0 1 0 1
- **State assignment**
  - Initial state and state for 1's

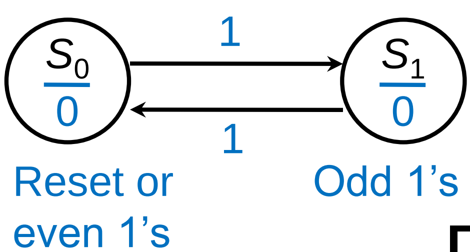


  - State for 0's

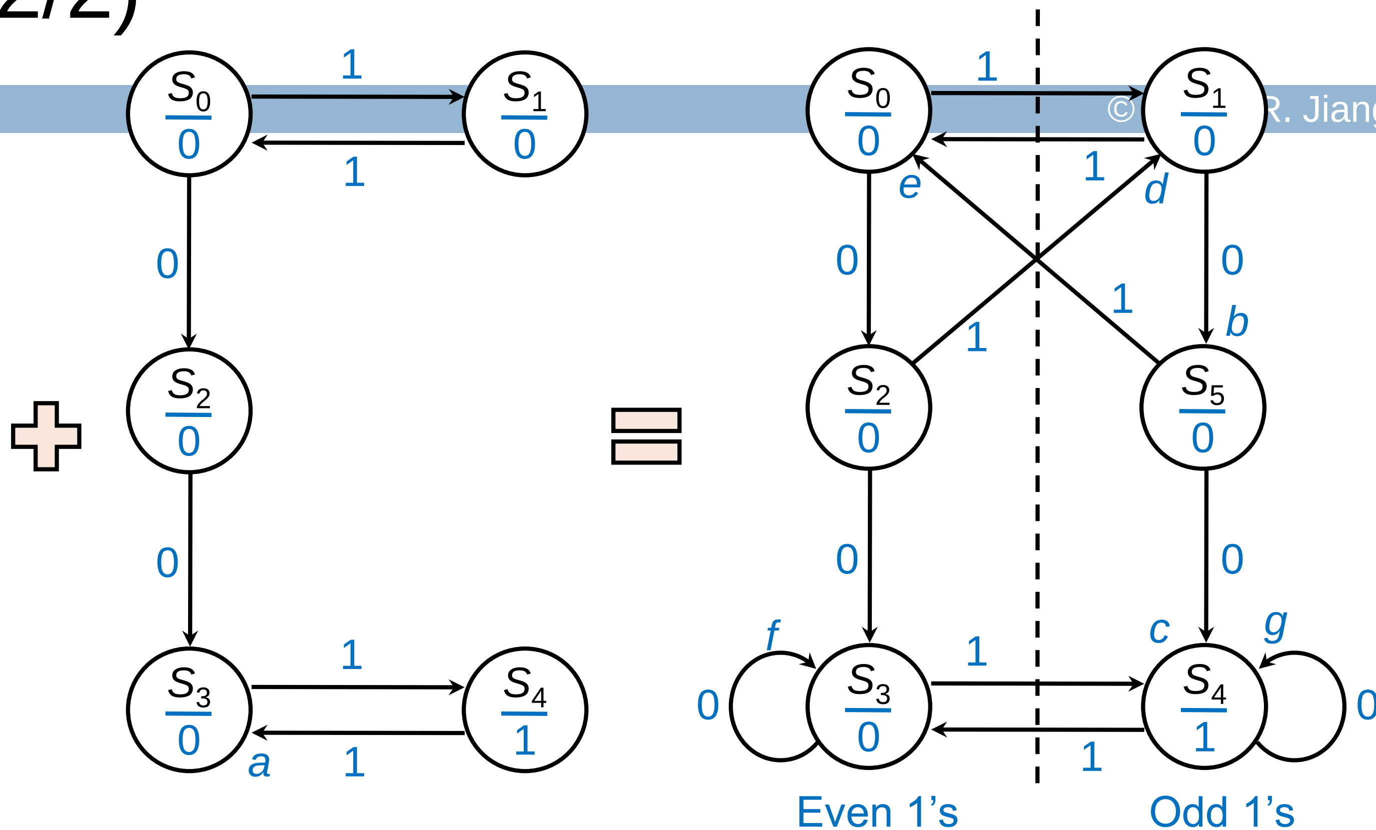
State	Sequence received
S_0	Reset or even 1's
S_1	Odd 1's
S_2	Even 1's and ends in 0
S_3	Even 1's and 00 has occurred
S_4	00 has occurred and odd 1's

Case IV (2/2)

- State graph



Initial state and state for 1's



State	Sequence received
S_0	Reset or even 1's
S_1	Odd 1's
S_2	Even 1's and ends in 0
S_3	Even 1's and 00 has occurred
S_4	00 has occurred and odd 1's

State	input sequences
S_0	Reset or even 1's
S_1	Odd 1's
S_2	Even 1's and ends in 0
S_3	Even 1's and 00 has occurred
S_4	00 has occurred and odd 1's
S_5	odd 1's and ends in 0

Guidelines for State Graph Construction

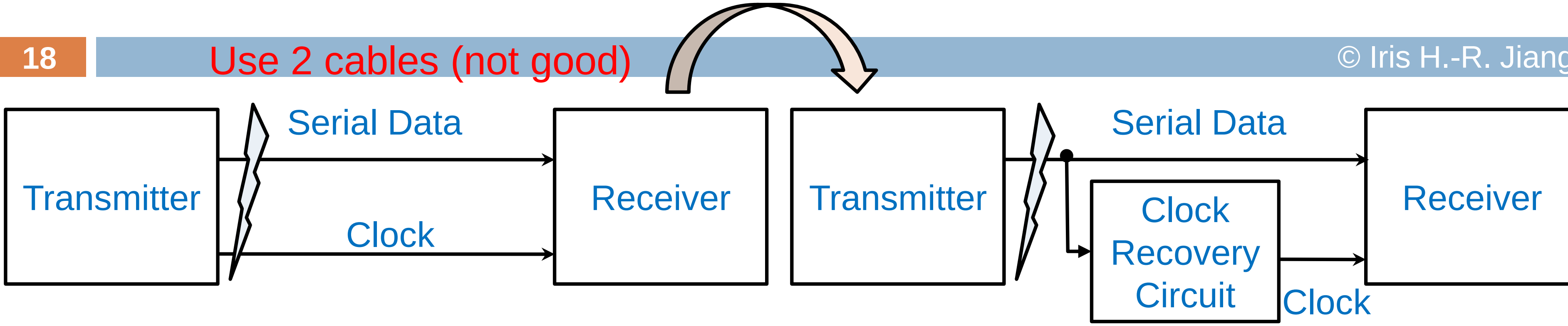
Guidelines for State Graphs Construction

- **Steps**
  1. Construct sample sequences to help you understand the problem
  2. Determine under what conditions it should reset
  3. If only one or two sequences leads to a nonzero output, construct a partial state graph
     - Another way, determine what sequences or groups of sequences must be remembered by the circuit and set up states accordingly
  4. Each time you add an arrow to the state graph, determine whether it can go to one of the previously defined states or whether a new state must added
  5. Check your graph to make sure there is one and only one path leaving each state for each combination of values of the input variables
  6. When your graph is complete, verify it by applying the input sequences formulated in step 1

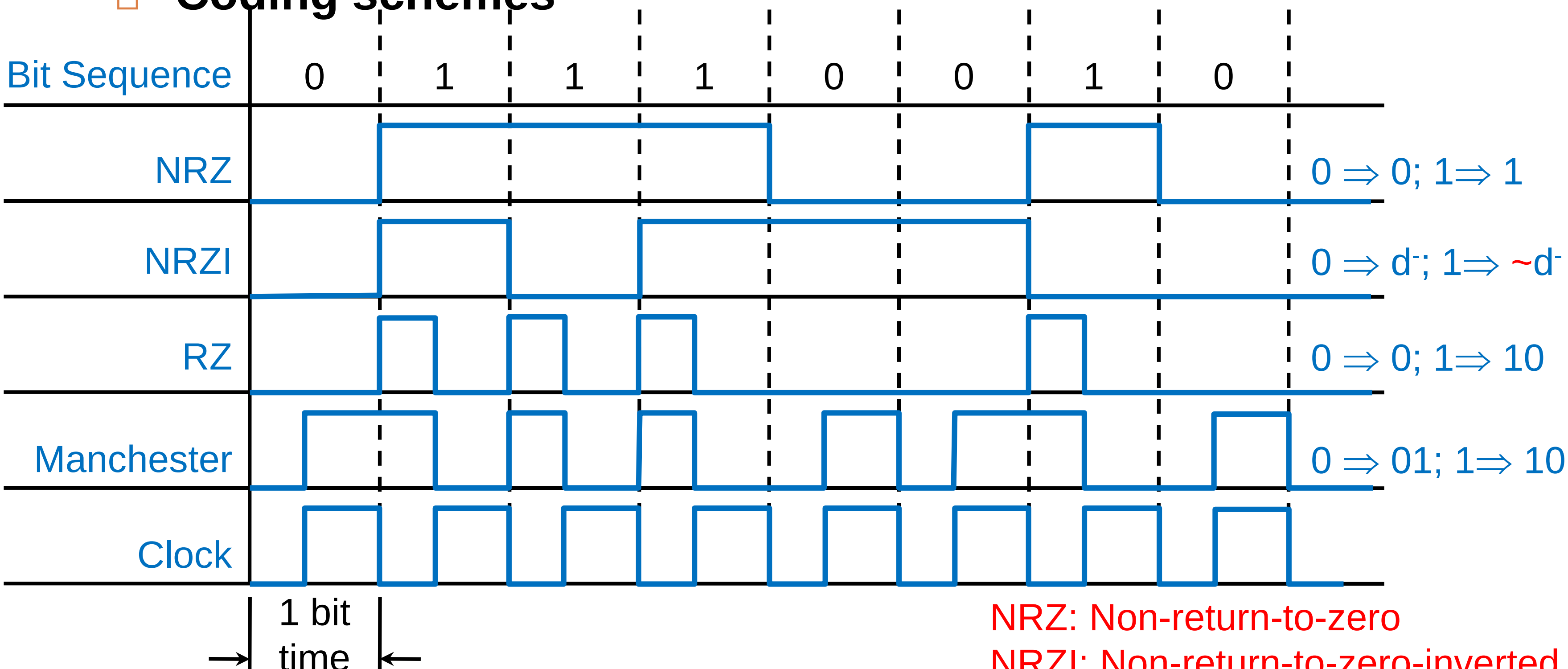
Serial Data Code Conversion

Serial Data Transmission

Use 2 cables (not good)

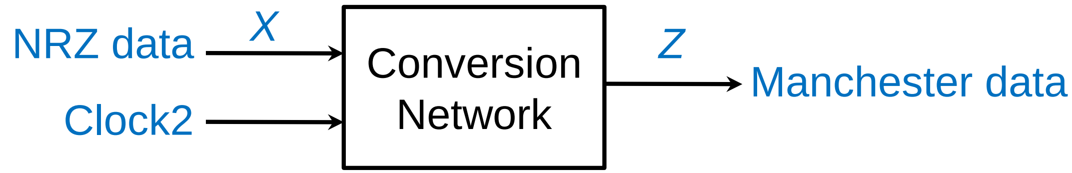


- **Coding schemes**



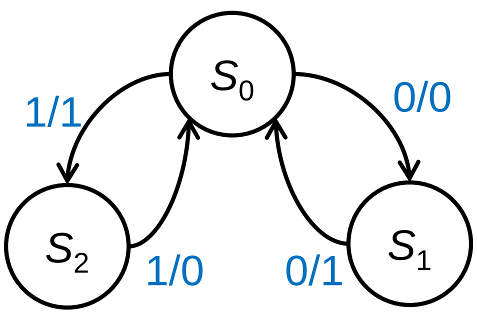
NRZ: Non-return-to-zero
NRZI: Non-return-to-zero-inverted
RZ: Return-to-zero

Mealy?



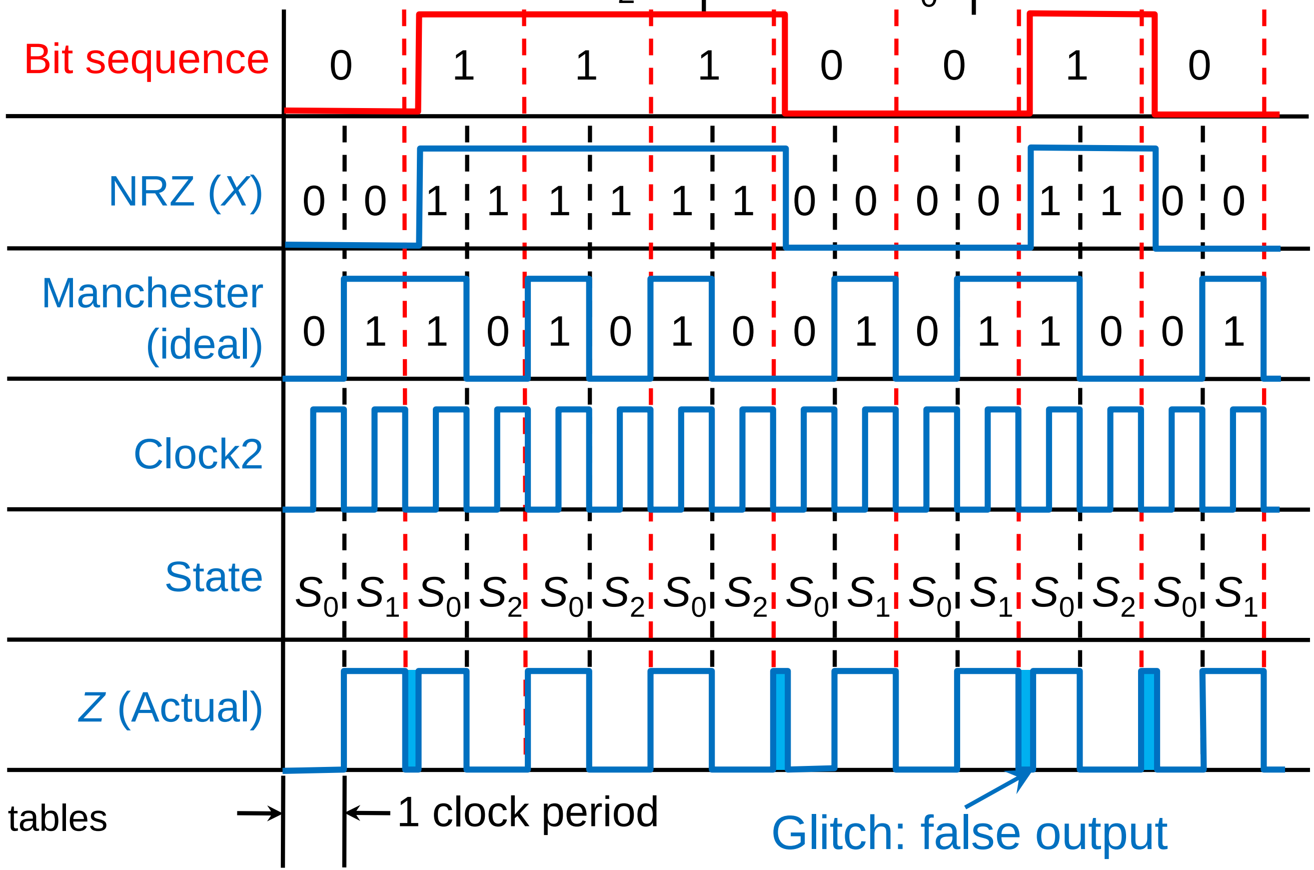
- **Mealy:**
  - Output depends on
    - Current state (synchronous)
    - Input (maybe asynchronous)
  - Fewer states

Present state	Next state $X = 0$	Next state $X = 1$	output (Z) $X = 0$	output (Z) $X = 1$
S_0	S_1	S_2	0	1
S_1	S_0	-	1	-
S_2	-	S_0	-	0

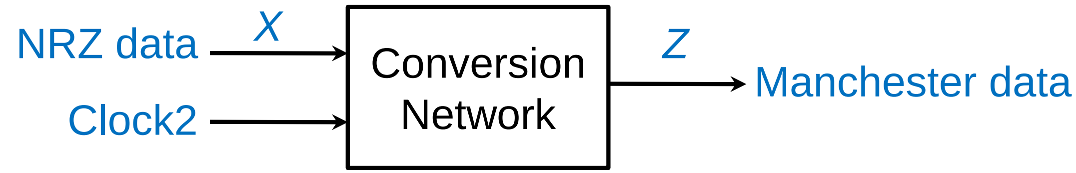


NRZ: combintaion of double 0's & double 1's

Starting state: S_0

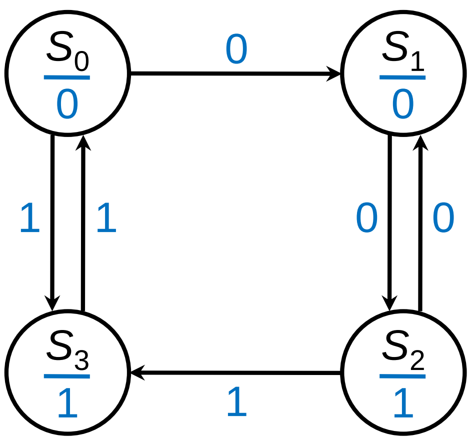


Moore?

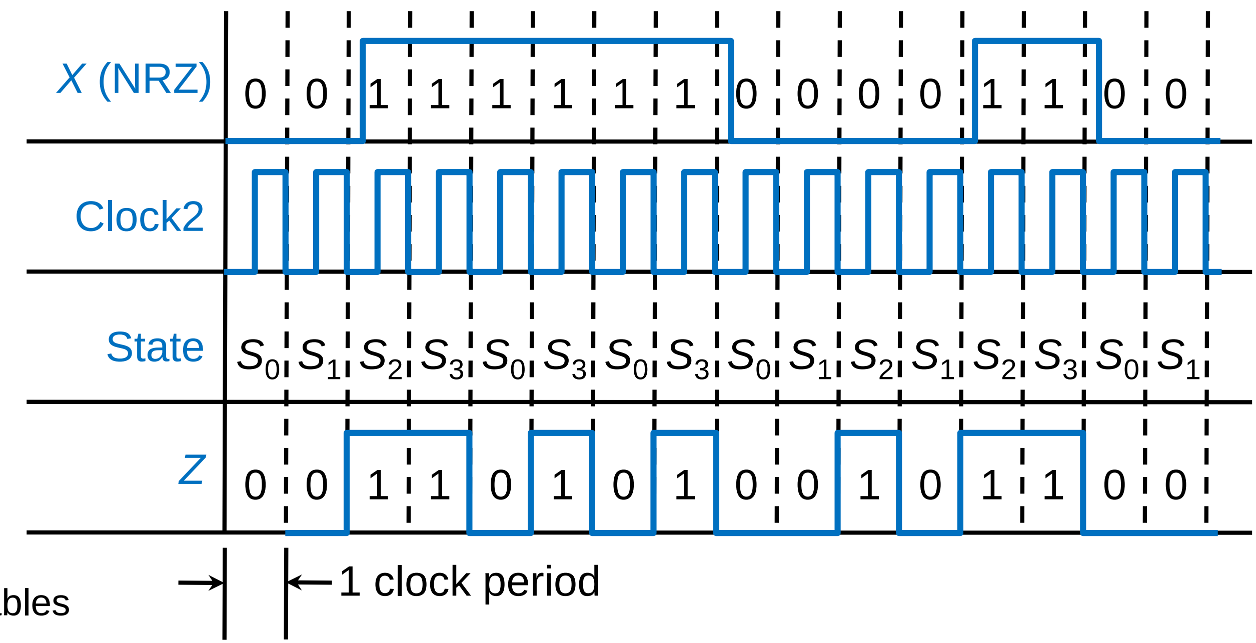


- **Moore:**
  - Output only depends on
    - Current state (synchronous)
  - More states (in general)
  - 1 clock period delay

Present state	Next state $X = 0$	Next state $X = 1$	Present output (Z)
S_0	S_1	S_3	0
S_1	S_2	-	0
S_2	S_1	S_3	1
S_3	-	S_0	1



Starting states: S_0, S_2

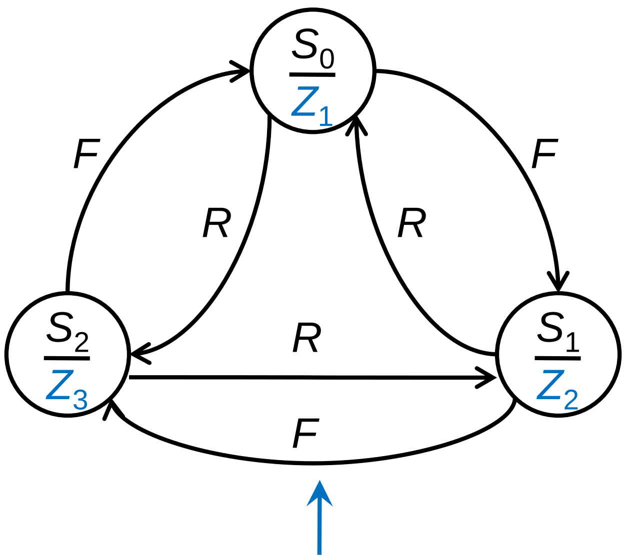


21

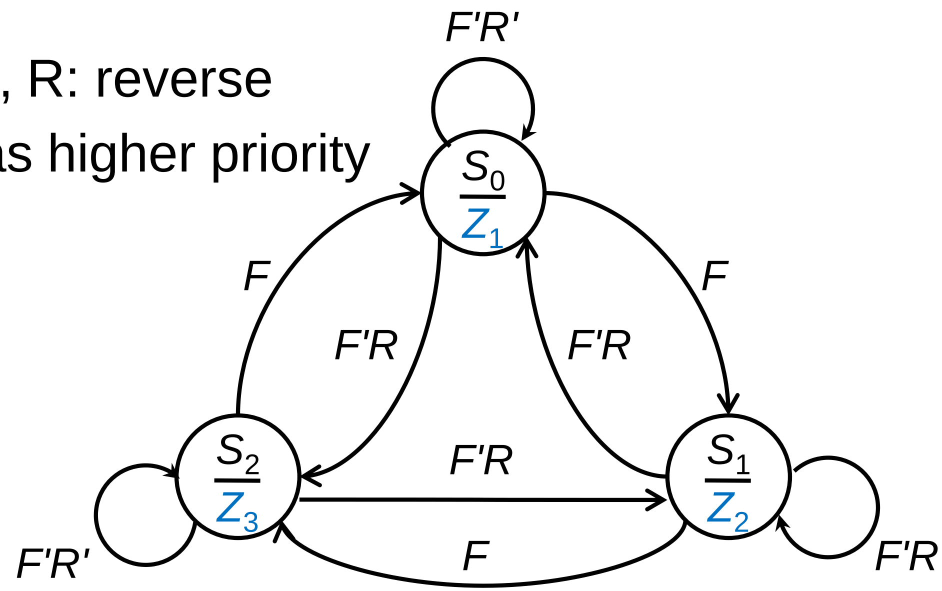
Alphanumeric State Graph Notation

Alphanumeric State Graph Notation

- **When a sequential circuit has several inputs, label the state graph arcs with alphanumeric input variable names instead of 0's and 1's**
  - e.g., 2 inputs: F: forward, R: reverse
  - Decide priority. E.g. F has higher priority than R



Incomplete



Present state	Next state FR = 00 01 10 11	Output $Z_1Z_2Z_3$
S_0	S_0 S_2 S_1 S_1	1 0 0
S_1	S_1 S_0 S_2 S_2	0 1 0
S_2	S_2 S_1 S_0 S_0	0 0 1

Complete?

- **Completely specified state graph**
  - OR together all input labels on arcs emanating from a state, the result can reduce to 1
    - Cover all conditions: $F + F'R + F'R' = F + F' = 1$
  - AND together any pair of input labels on arcs emanating from a state, the result can reduce to 0
    - Only one arc is valid: $F \cdot F'R = 0$, $F \cdot F'R' = 0$, $F'R \cdot F'R' = 0$
- **Notation in state graph**
  - $X_1X'_4/Z_2Z_3 \equiv$ 1--0/0110
  - -/$Z_1 \equiv$ ----/1000
    - For any combination of input values…