Digital Systems Design and Laboratory
[14. Derivation of State Graphs and Tables]

Chung-Wei Lin
cwlin@csie.ntu.edu.tw
CSIE Department
National Taiwan University
Spring 2019

We would like to thank Prof. Hui-Ru Jiang (NTU) for contributing the lecture material

Sequential Logic Design

- Unit 11: Latches and Flip-Flops
- Unit 12: Registers and Counters
- Units 13--15: Finite State Machines
- Unit 16: Summary

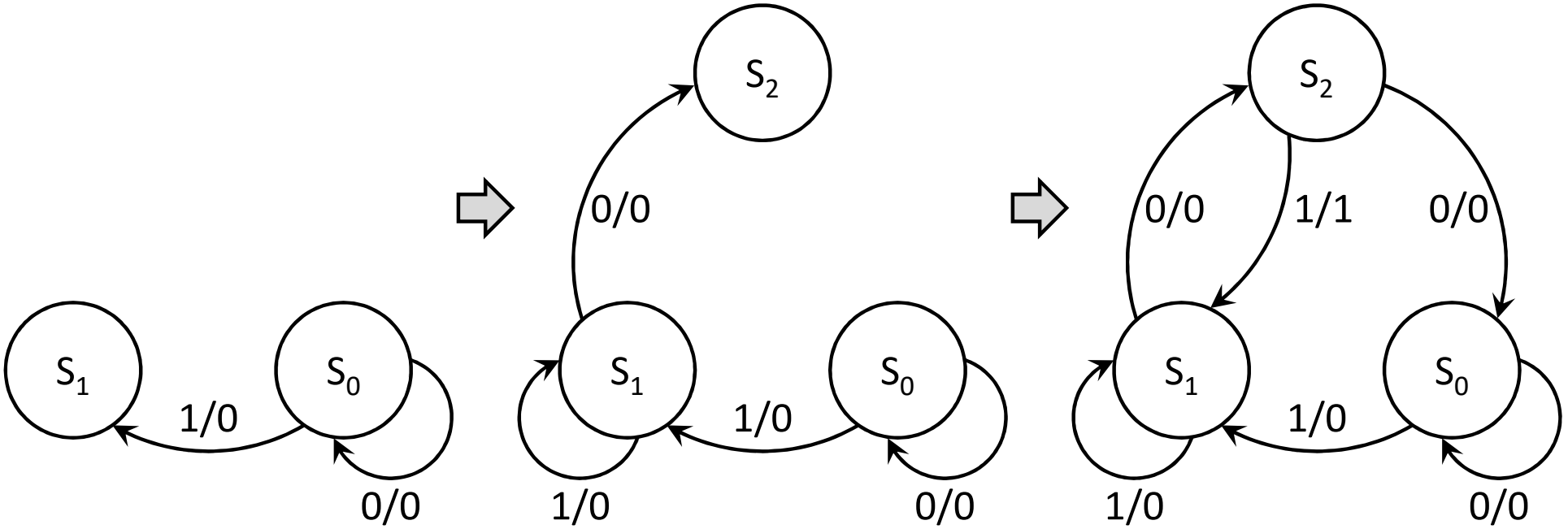
- Designing a sequential circuit
  - Construct a state graph or state table (Unit 14)
  - Simplify it (Unit 15)
  - Derive flip-flop input equations and output equations (Unit 12)

Outline

- **Design of a Sequence Detector**
- Guidelines for Construction of State Graphs
- Serial Data Code Conversion
- Alphanumeric State Graph Notation

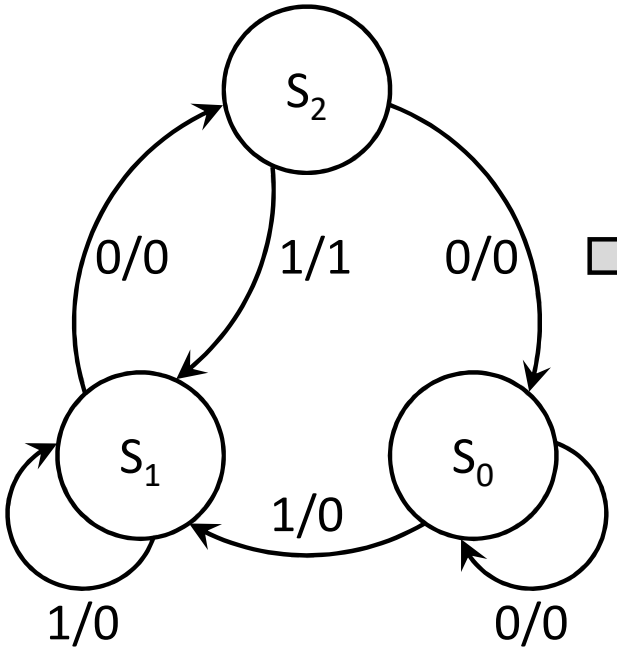
"101" Detector (1/5)

- Output "1" if detecting "101"
- Example
  - Input X 0 0 1 1 0 1 1 0 0 1 0 1 0 1 0 0
  - Output 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"



"101" Detector (2/5)

- State table



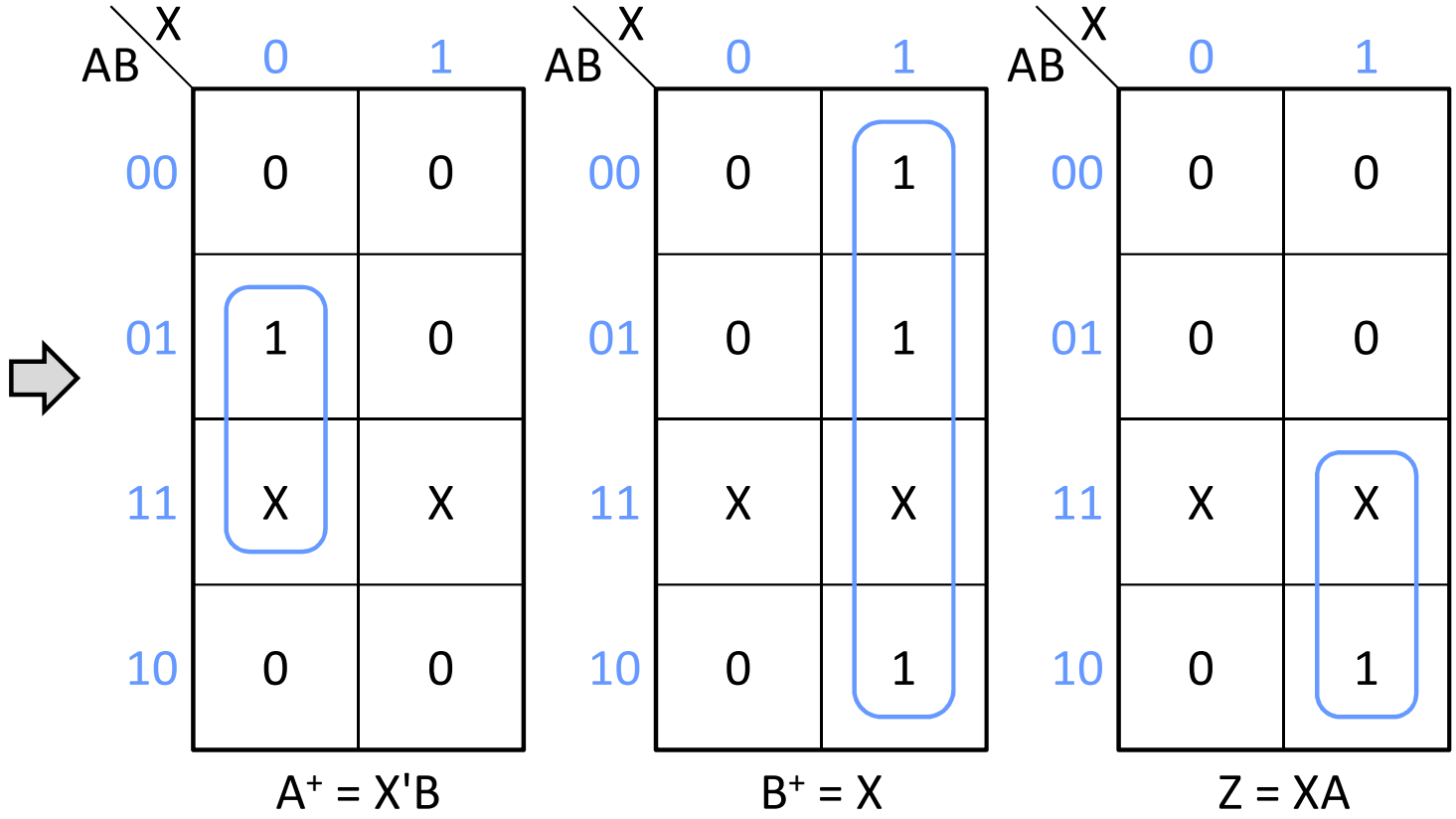
Present State	Next State		Present Output	
	X = 0	X = 1	X = 0	X = 1
S_0	S_0	S_1	0	0
S_1	S_2	S_1	0	0
S_2	S_0	S_1	0	1

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

"101" Detector (3/5)

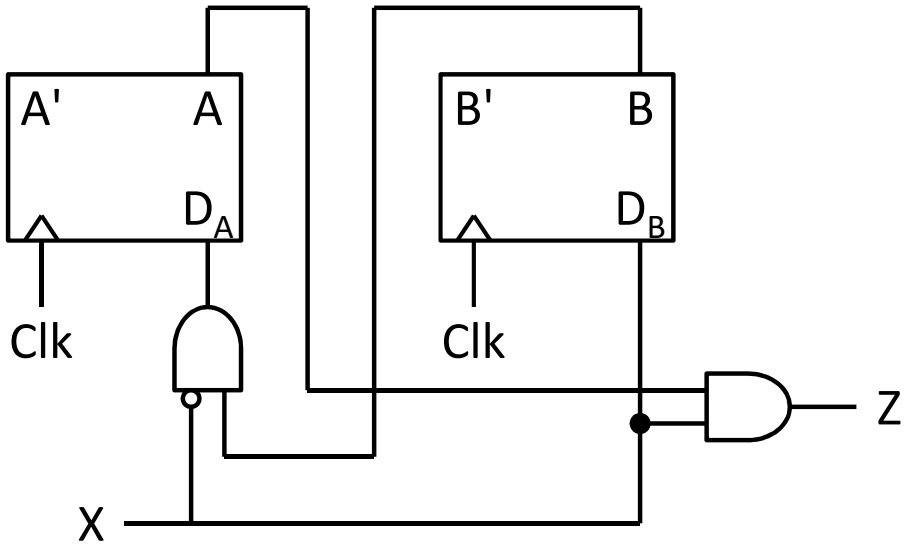
- State maps

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



"101" Detector (4/5)

- Realize it
  - $A^+ = X'B$
  - $B^+ = X$
  - $Z = XA$



"101" Detector (5/5)

- Some variants
  - Moore machine?
    - One more state
  - "010" and "1001" detector
    - A more complicated machine

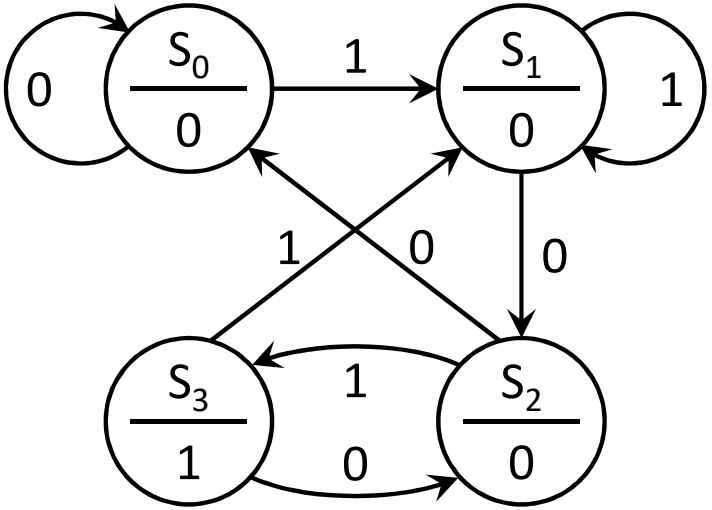
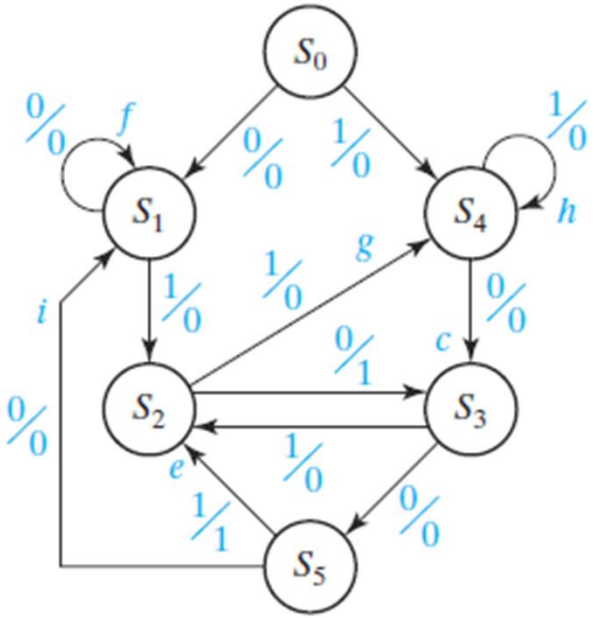


FIGURE 14-10



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

Outline

- Design of a Sequence Detector
- **Guidelines for Construction of State Graphs**
- Serial Data Code Conversion
- Alphanumeric State Graph Notation

Guidelines for Construction of State Graphs

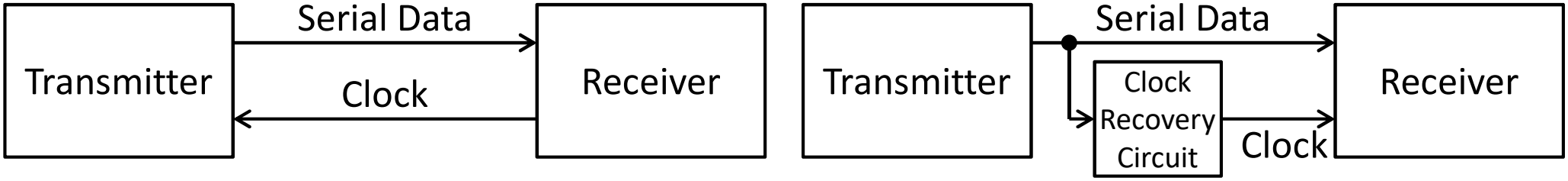
❑ Steps

- ➢ Construct sample sequences to help you understand the problem
- ➢ Determine under what conditions it should reset
- ➢ 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
- ➢ 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
- ➢ 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
- ➢ When your graph is complete, verify it by applying the input sequences formulated in step 1

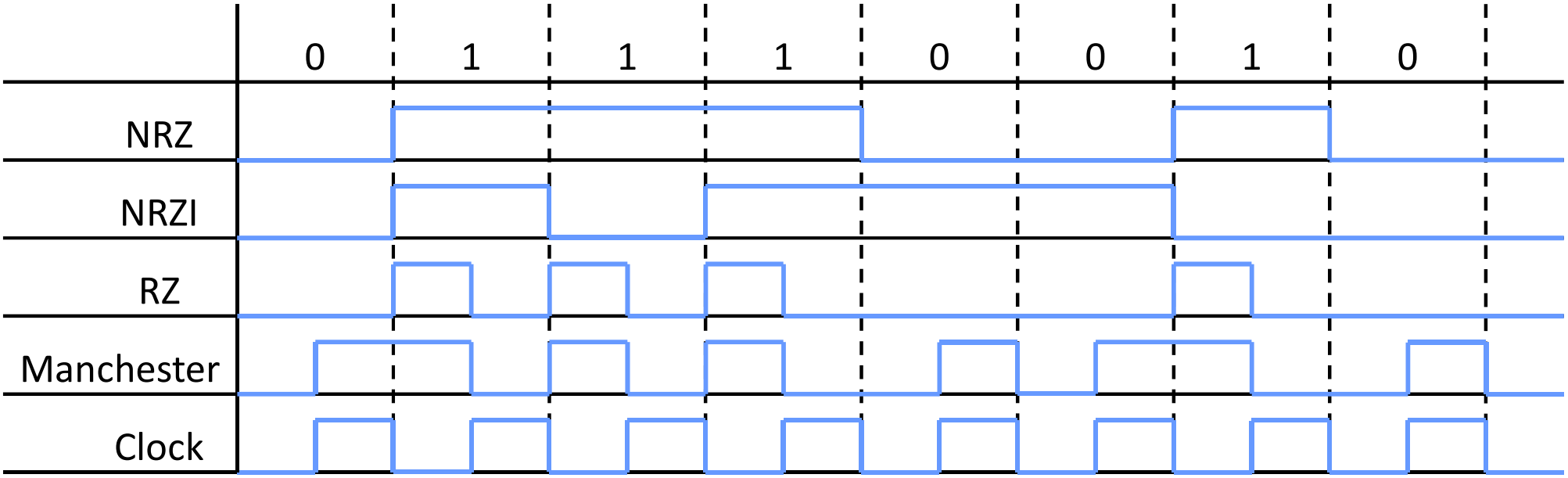
Outline

- Design of a Sequence Detector
- Guidelines for Construction of State Graphs
- **Serial Data Code Conversion**
- Alphanumeric State Graph Notation

Serial Data Transmission

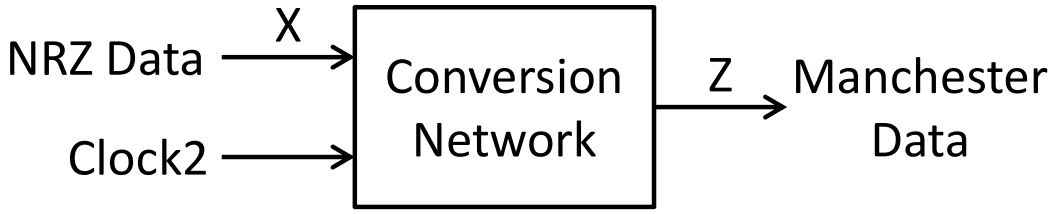


❑ Coding schemes

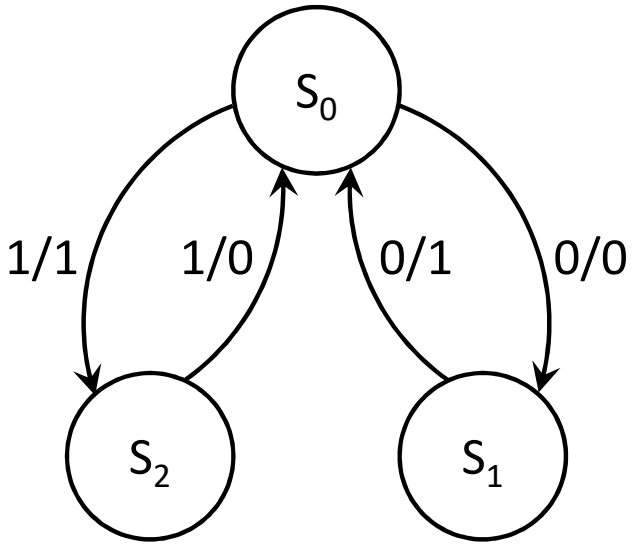


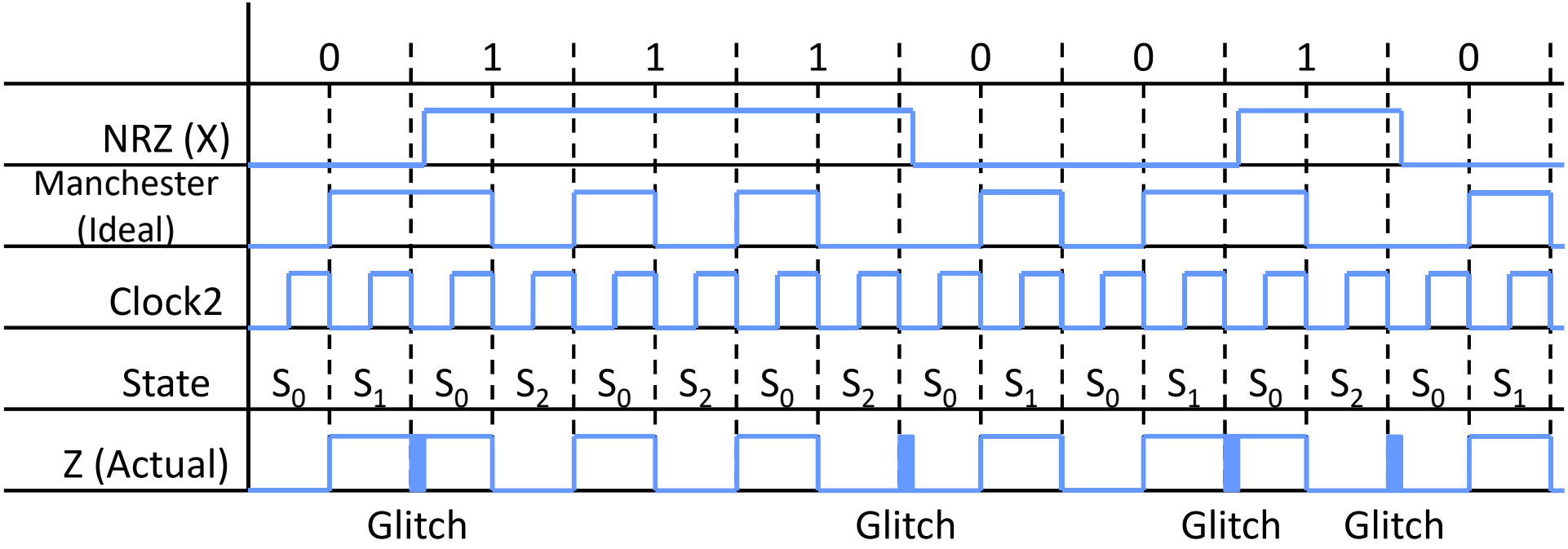
NRZ: Non-Return-to-Zero	0 → 0	1 → 1
NRZI: Non-Return-to-Zero-Inverted	0 → D	1 → D'
RZ: Return-to-Zero	0 → 0	1 → 10
Manchester	0 → 01	1 → 10

Mealy Machine

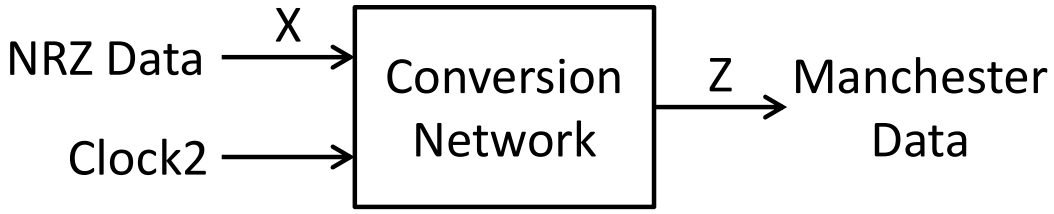


- Output depends on
  - Current state (synchronous)
  - Input (maybe asynchronous)
- State changes at a falling edge
- Fewer states

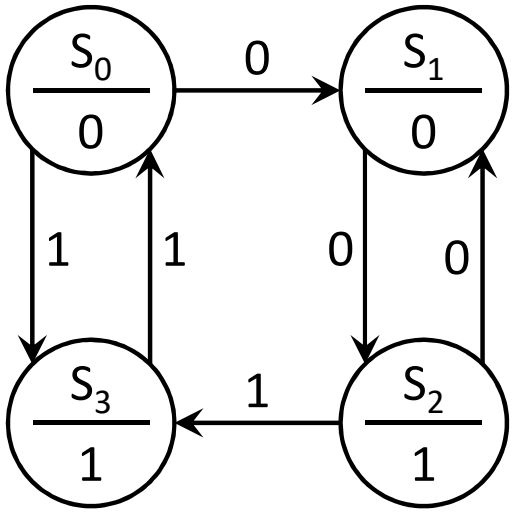


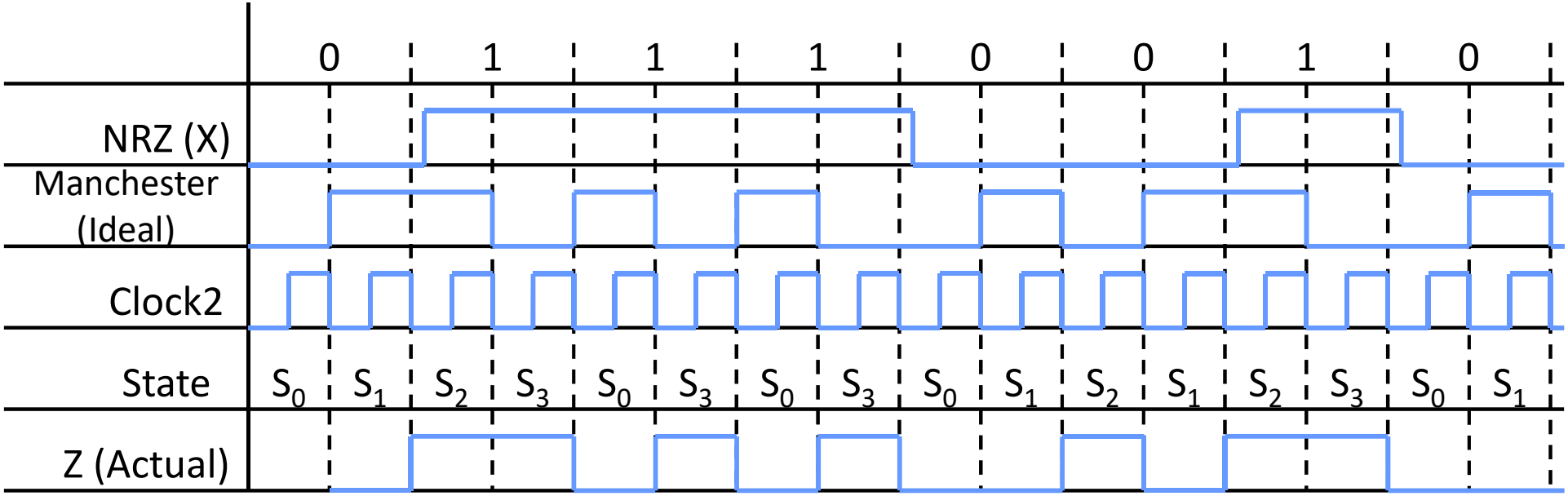


Moore Machine



- Output only depends on
  - Current state (synchronous)
- State changes at a falling edge
- More states (in general)
- 1 clock period delay



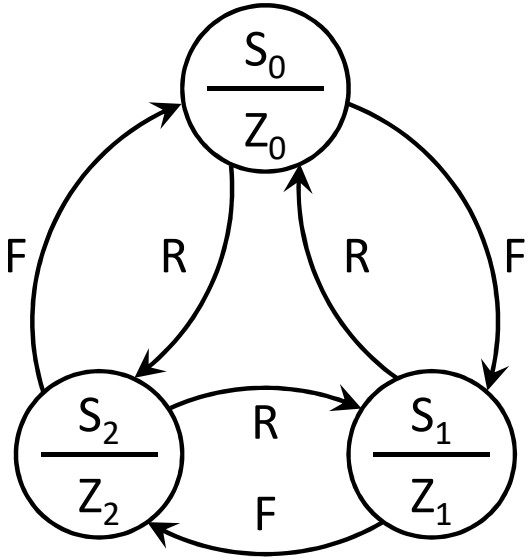


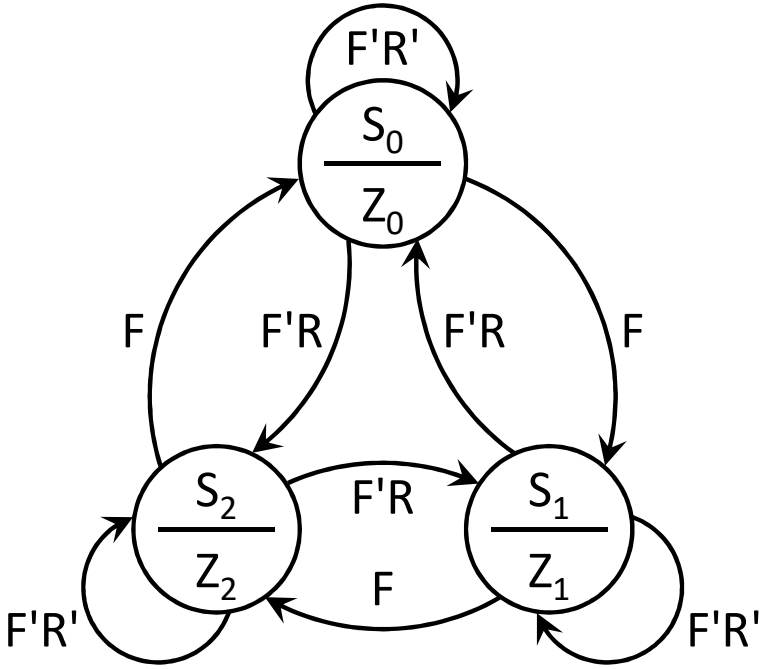
Outline

- Design of a Sequence Detector
- Guidelines for Construction of State Graphs
- Serial Data Code Conversion
- **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
  - Example
    - 2 inputs: F for "forward" and R for "reverse"

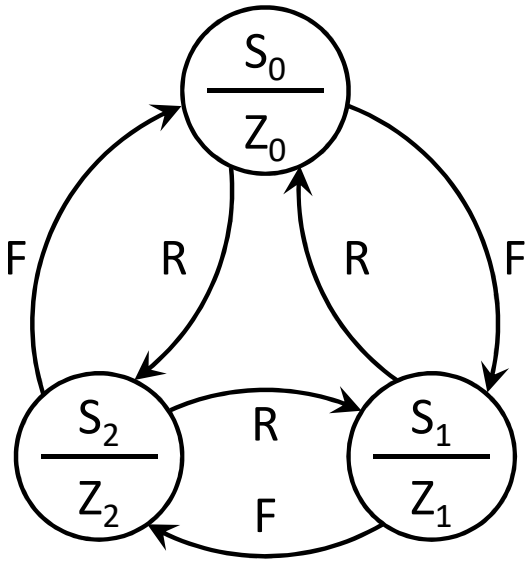




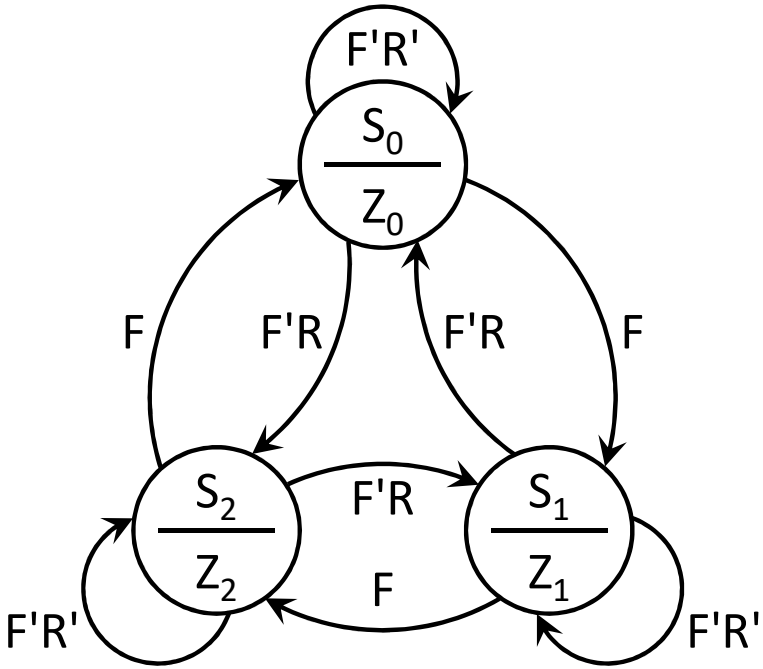
Completely Specified State Graph

Properties

- **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·F'R = 0, F·F'R' = 0, F'R·F'R' = 0



Incomplete



Q&A