



23.03.20 13:54

CSCI 150

Introduction to Digital and Computer System Design

Lecture 4: Sequential Circuit Review



Jetic Gū
2020 Winter Semester (S1)

Overview

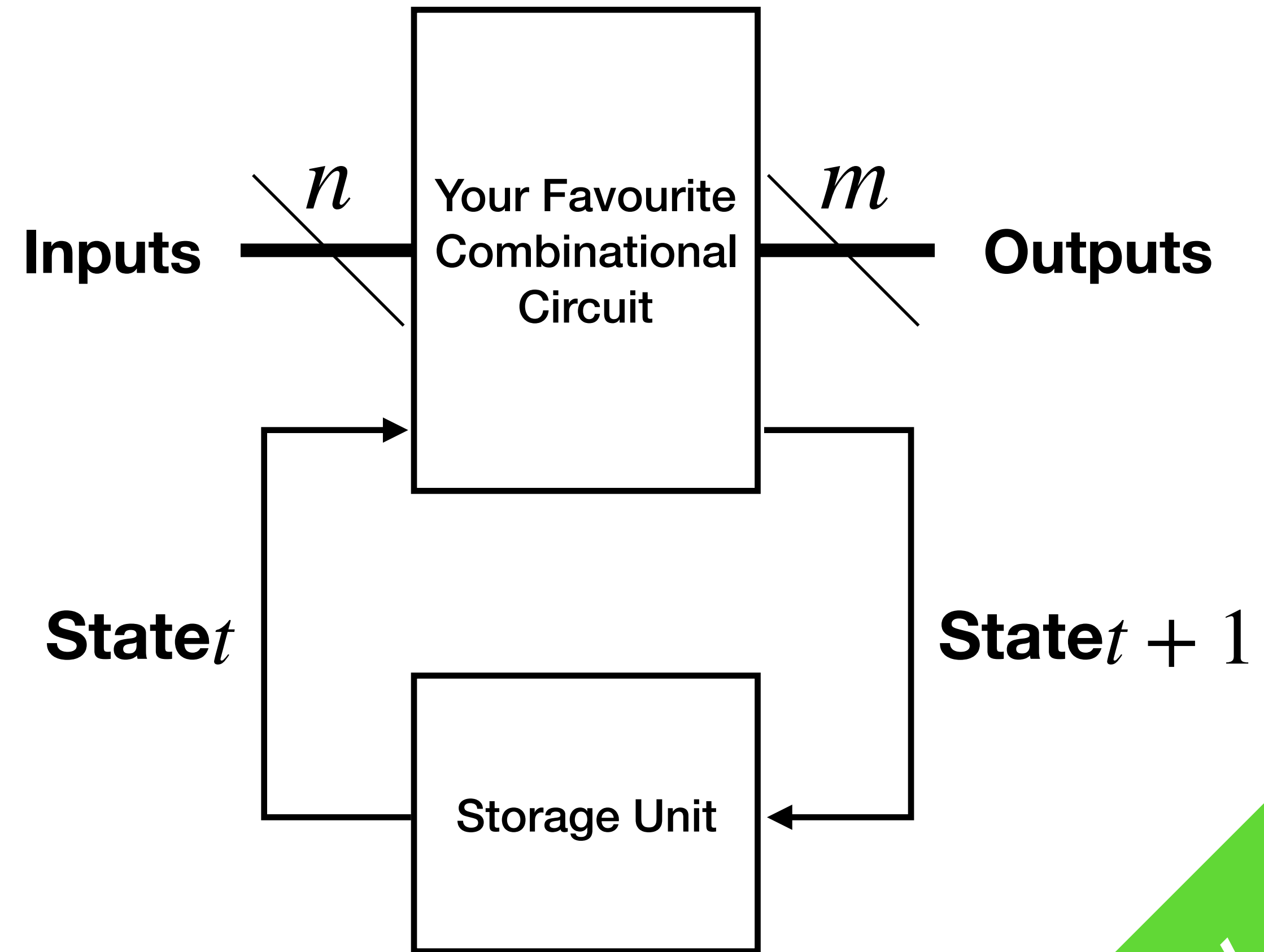
- Focus: Basic Information Retaining Blocks
- Architecture: Sequential Circuit
- Textbook v4: Ch5; v5: Ch4
- Core Ideas:
 1. What we've learned in Sequential Circuit
 2. How to apply sequential circuit design in real life?

Quick Review

Sequential circuits and How to Design 'em

Definitions

1. **Storage Elements**
circuits that can store binary information
2. **State**
partial results, instructions, etc.
3. **Synchronous Sequential Circuit**
Signals arrive at discrete instants of time,
outputs at next time step
4. **Asynchronous Sequential Circuit**
Signals arrive at any instant of time,
outputs when ready



Definitions

3. Synchronous Sequential Circuit

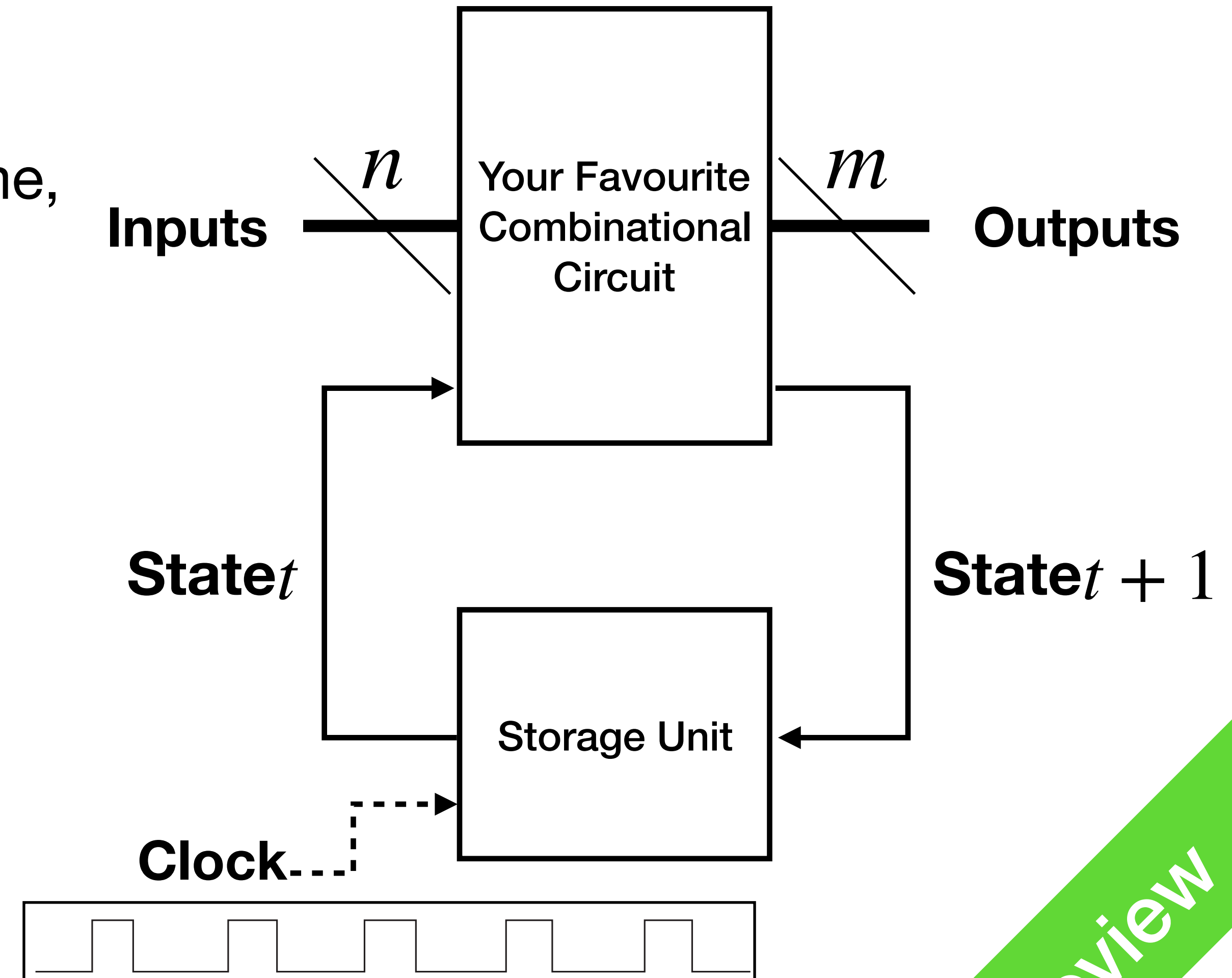
Signals arrive at discrete instants of time, outputs at next time step

- Has Clock

4. Asynchronous Sequential Circuit

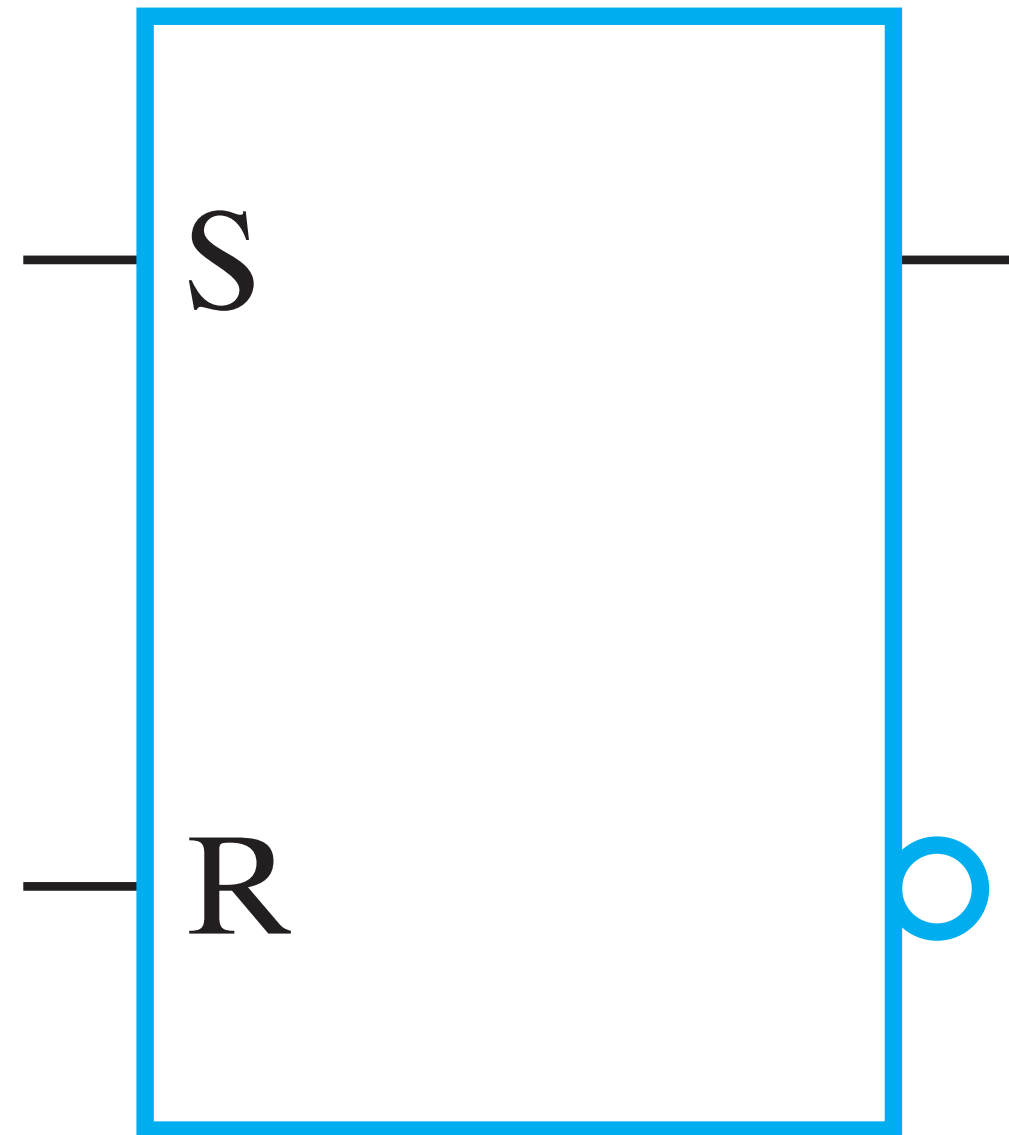
Signals arrive at any instant of time, outputs when ready

- May not have Clock

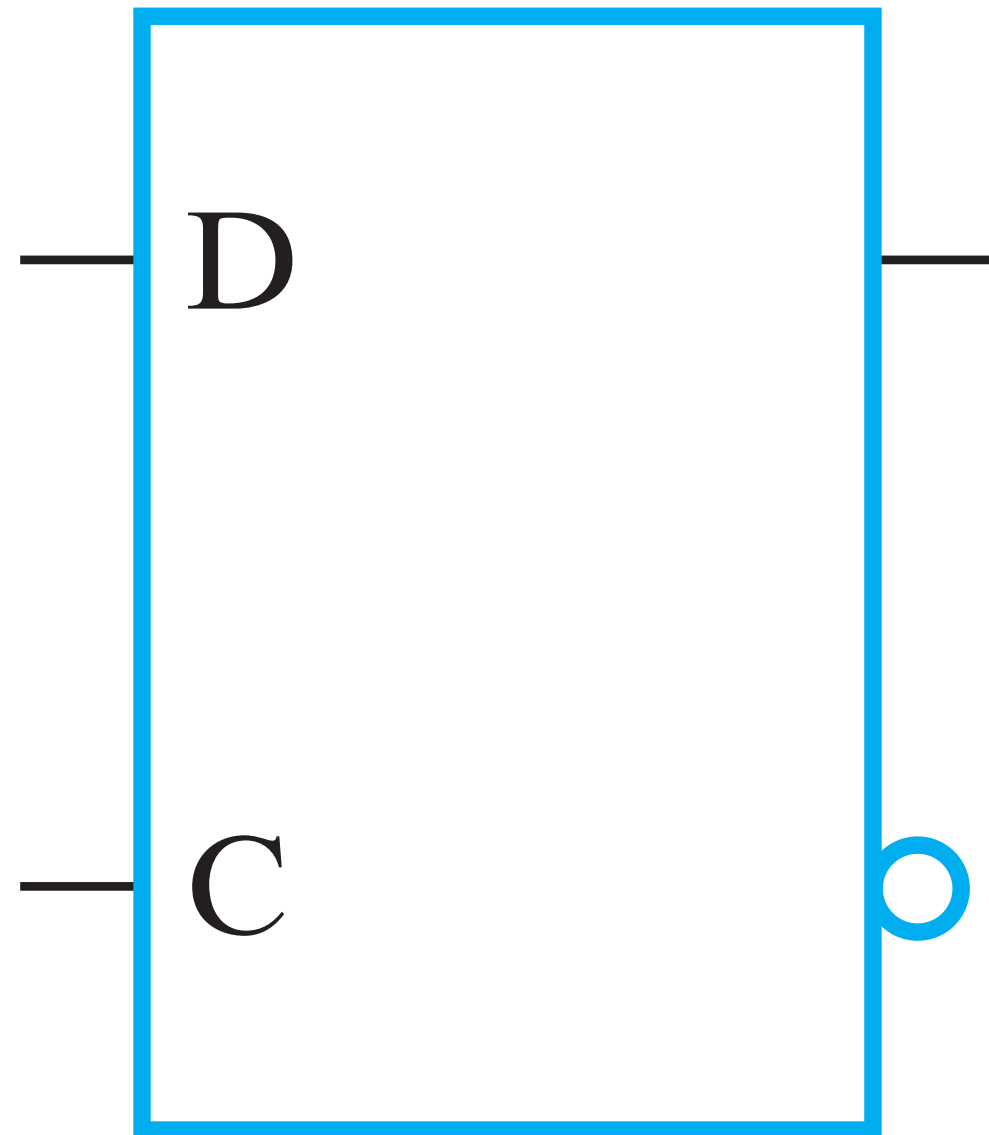


Summary

Latches

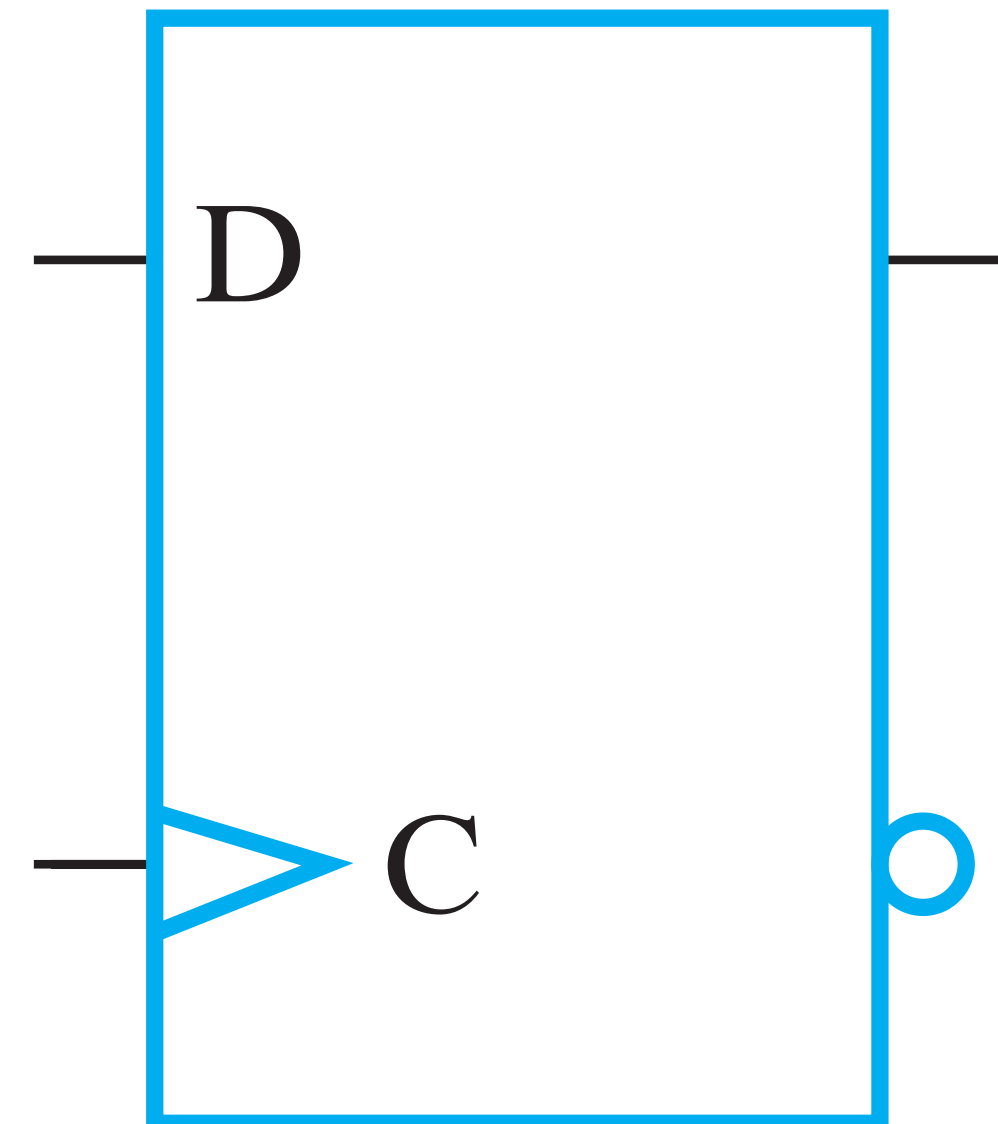


SR



D with 1 Control

Flip-Flops



Triggered D

Systematic Design Procedures Sequential Circuits

1. **Specification**
2. **Formulation**
e.g. using **state table** or **state diagram**
3. **State Assignment**: assign binary codes to states
4. **Flip-Flop Input Equation Determination**: Select flip-flop types, derive input equations from next-state entries
5. **Output Equation Determination**: Derive output equations from the output entries
6. **Optimisation**
7. **Technology Mapping**
8. **Verification**

Systematic Design Procedures

Sequential Circuits

2. **Formulation:** State Diagram (**Mealy**) & State Machine Diagram (**Moore**)
Modelling State Transitions; TC & OC
2. **Formulation:** State Table
Equivalent truth table: Input (Present State, Input); Output (Next State, Output)
3. **State Assignment**
Sequential indexing (0...0000, 0...0001, 0...0010, 0...0011, ...);
One-Hot states (0...0001, 0...0010, 0...0100, 0...1000, ...);

Systematic Design Procedures

Sequential Circuits

4. Flip-Flop Input Equation Determination

- Determine next state inputs by considering flip-flop types
e.g. D flip-flop easiest
- Derive input equations from next-state entries in **State Table**
e.g. using Sum-of-Minterms

5. Output Equation Determination

- Derive output equations from the output entries in **State Table**

Systematic Design Procedures

Sequential Circuits

4. Flip-Flop Input Equation Determination
5. Output Equation Determination

Present State	Next State		Output Z	
	X = 0	X = 1	X = 0	X = 1
A 00	00 A	B 01	0	0
B 01	00 A	C 10	0	0
C 10	11 D	C 10	0	0
D 11	00 A	B 01	0	1

Systematic Design Procedures

Sequential Circuits

4. Flip-Flop Input Equation Determination
5. Output Equation Determination

D_1D_0 for next state
 S_1S_0 for present

Present State	Next State		Output Z	
	X = 0	X = 1	X = 0	X = 1
A 00	00 A	B 01	0	0
B 01	00 A	C 10	0	0
C 10	11 D	C 10	0	0
D 11	00 A	B 01	0	1

Systematic Design Procedures

Sequential Circuits

4. Flip-Flop Input Equation Determination
5. Output Equation Determination

D_1D_0 for next state
 S_1S_0 for present

Present State S_1S_0	Next State D_1D_0		Output Z	
	X = 0	X = 1	X = 0	X = 1
00	00	01	0	0
01	00	10	0	0
10	11	10	0	0
11	00	01	0	1

Systematic Design Procedures

Sequential Circuits

4. Flip-Flop Input Equation Determination

5. Output Equation Determination

D_1D_0 for next state

S_1S_0 for present

$$D_1 = F_1(X, S_1, S_0) = \Sigma m(2, 5, 6)$$

$$D_0 = F_0(X, S_1, S_0) = \Sigma m(2, 4, 7)$$

$$Z = m_7$$

X	S_1S_0	D_1D_0	Z
0	00	00	0
0	01	00	0
0	10	11	0
0	11	00	0
1	00	01	0
1	01	10	0
1	10	10	0
1	11	01	1

Systematic Design Procedures

Sequential Circuits

6. Optimisation

Unused states can be implemented as don't care conditions

- In this example

$m_0, m_1, m_{12}, m_{13}, m_{14}, m_{15}$

are unused, and can all be **don't care conditions**

$$D_A = \Sigma m(5, 7, 8, 9, 11)$$

$$D_B = \Sigma m(3, 4)$$

$$D_C = \Sigma m(2, 4, 6, 8, 10)$$

$$d = \Sigma m(0, 1, 12, 13, 14, 15)$$

Present State			Input	Next State		
A	B	C	X	A	B	C
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	1	1
0	1	0	1	1	0	0
0	1	1	0	0	0	1
0	1	1	1	1	0	0
1	0	0	0	1	0	1
1	0	0	1	1	0	0
1	0	1	0	0	0	1
1	0	1	1	1	0	0

Systematic Design Procedures

Sequential Circuits

6. Optimisation

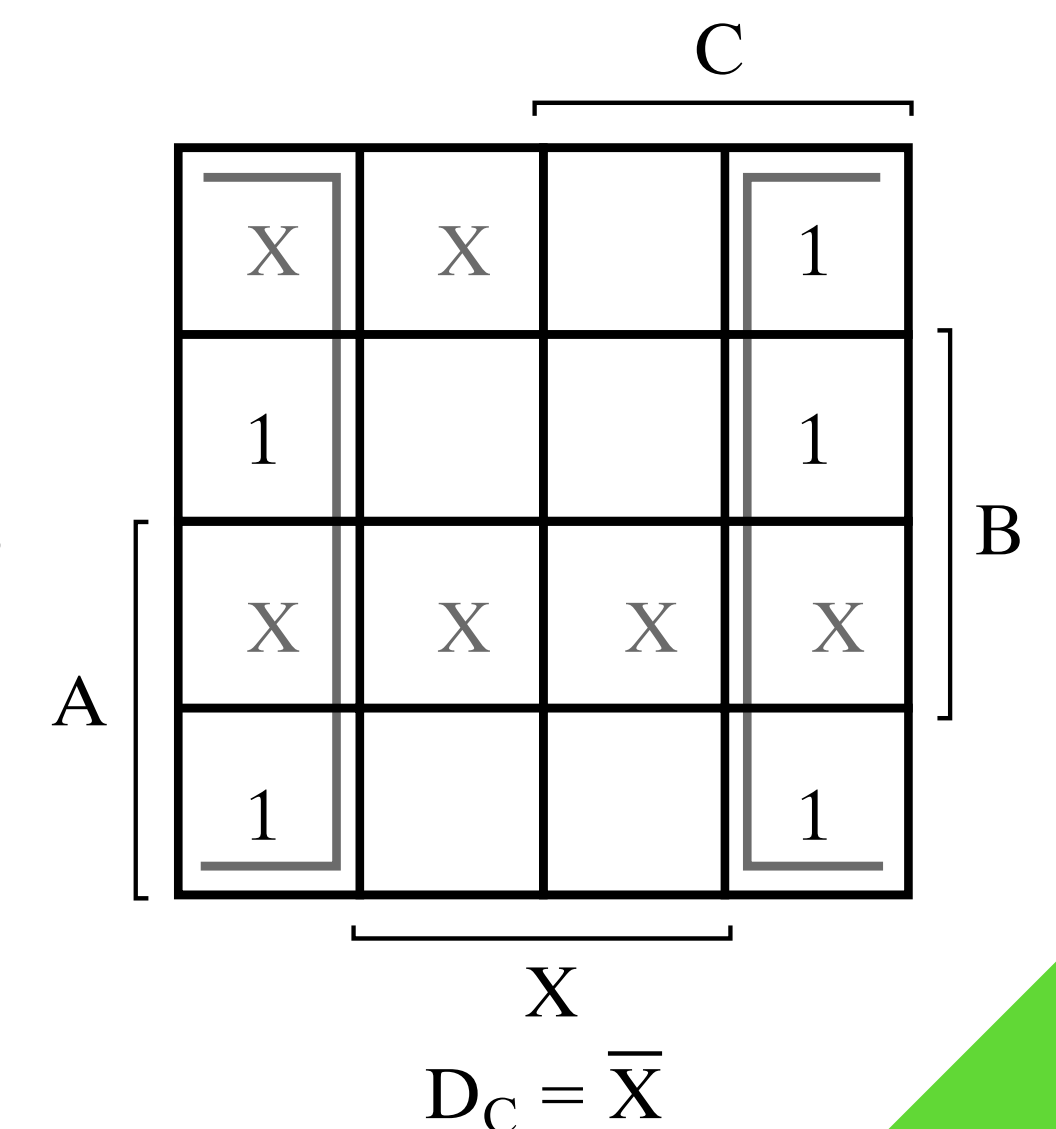
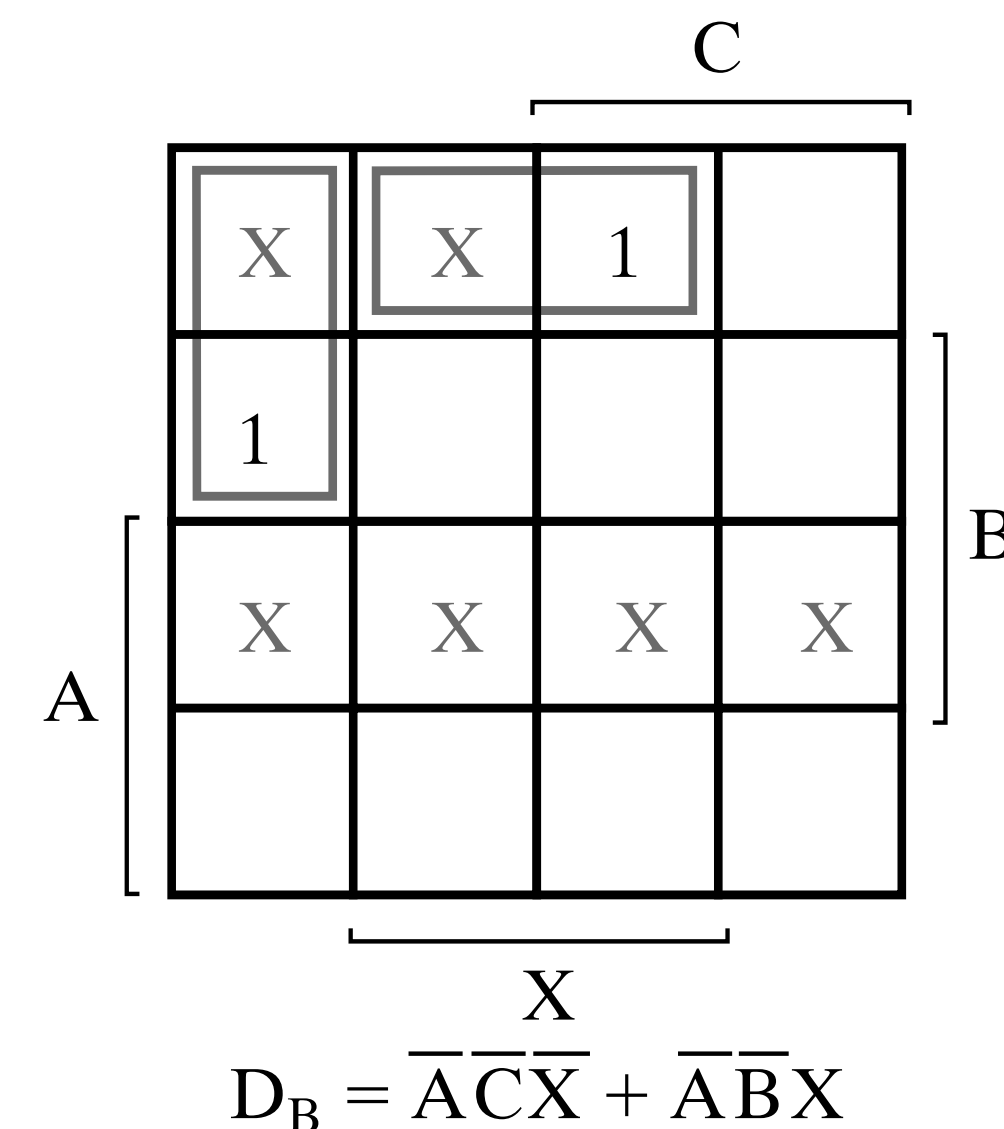
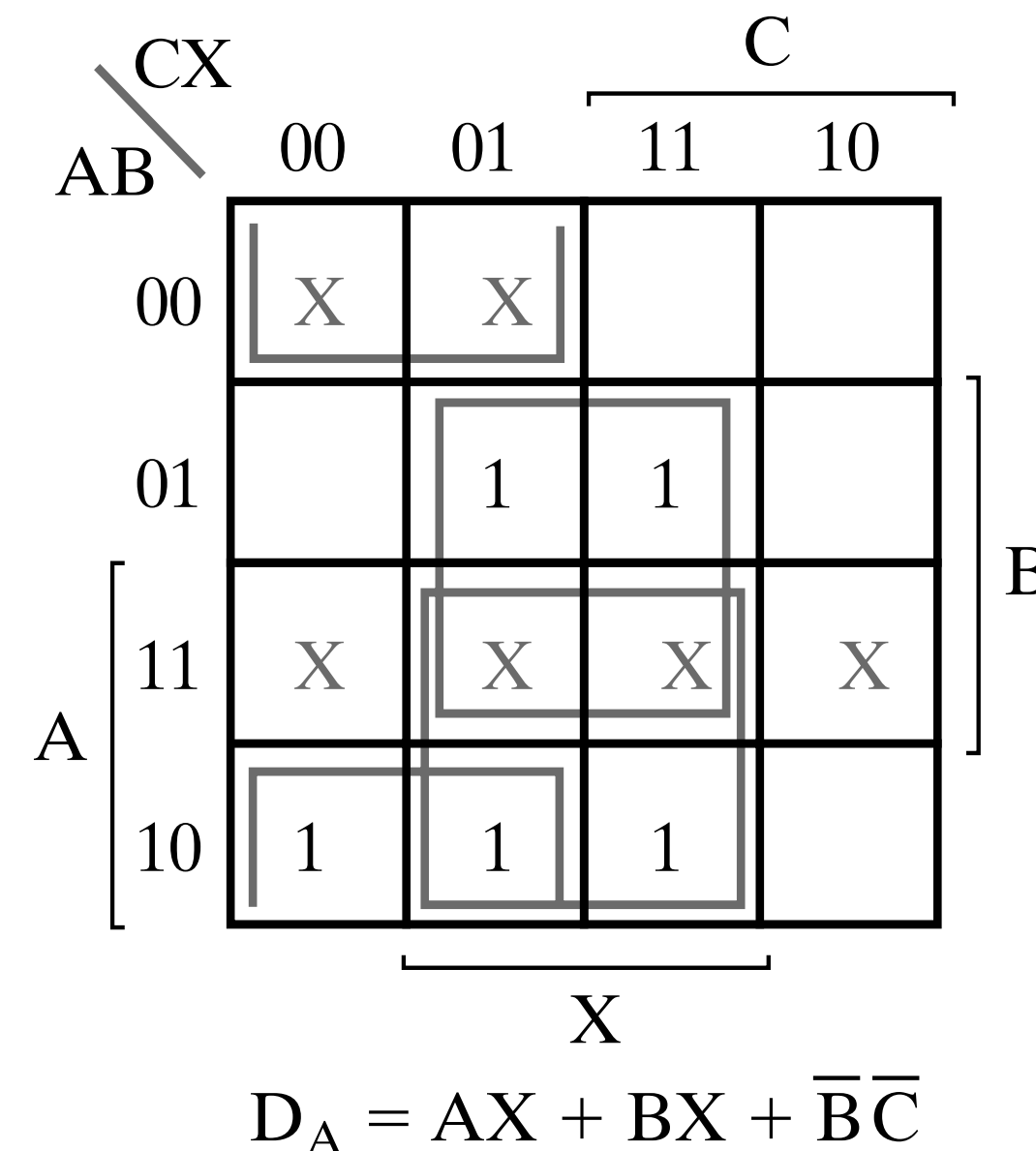
Unused states can be implemented as don't care conditions

$$D_A = \Sigma m(5,7,8,9,11)$$

$$D_B = \Sigma m(3,4)$$

$$D_C = \Sigma m(2,4,6,8,10)$$

$$d = \Sigma m(0,1,12,13,14,15)$$



BCD Decoder

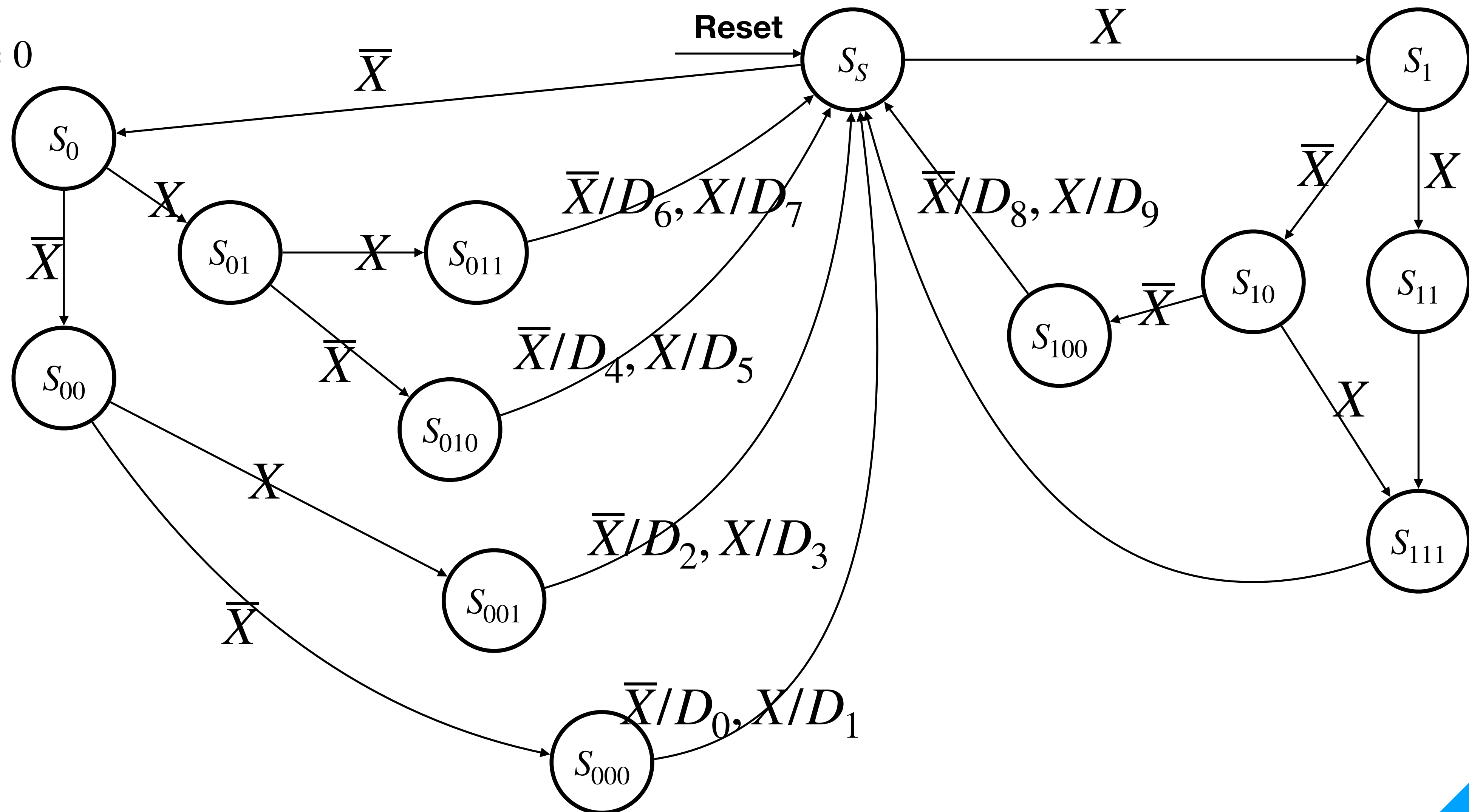
State Machine Diagram Exercise

Specification

- Input: 1-bit X
- Output: 10-bits, $D_0 \dots D_9$. D_i indicates current number is i .
All 0 for invalid or incomplete input. Outputs every 4 CLKs.
- e.g. Input sequence: 0; 0; 0; 1; 0; 1; 0; 0; 1; 1; 1; 1;
Output: 0; 0; 0; $D_1 = 1$; 0; 0; 0; 0; $D_4 = 1$; 0; 0; 0; 0;

Formulation

Default: $D_i = 0$



All 8 Steps examples are too long for in class tutorials

Textbook Examples in 5.7 (v4); 4.6 (v5)