YORK
UNIVERSITÉ
UNIVERSITY

EECS 3201: Digital Logic Design Lecture 11

Ihab Amer, PhD, SMIEEE, P.Eng.

II. Design of Sequential Circuits

- Given the word description and specs of the desired operation, obtain the state diagram/table of the circuit, and hence derive the circuit diagram

Example 1

- Using D-type FFs, design a circuit that detects three or more consecutive 1's in a stream of bits coming through an input line

Four States → Two FF's

Assuming binary state encoding

Required States

No cons. 1's detected so far → S_0

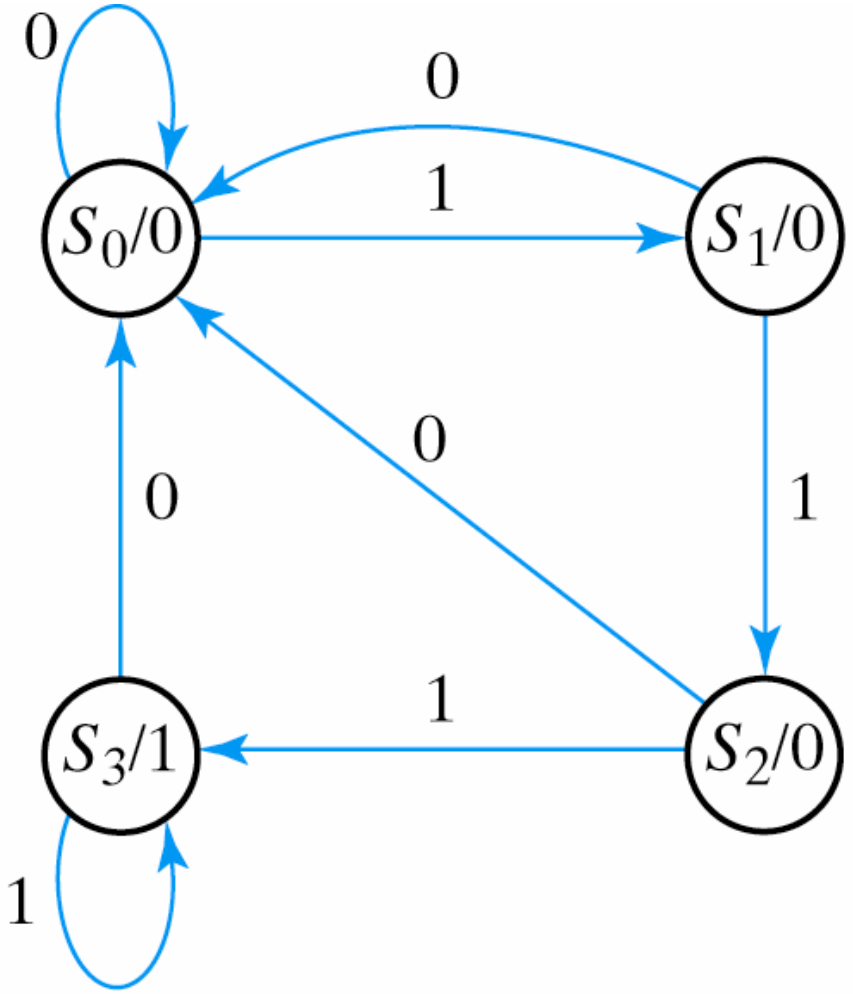
First cons. 1 detected so far → S_1

Two cons. 1's detected so far → S_2

Three (or more) cons. 1's detected so far → S_3

State	FF o/ps	
	A	B
S_0	0	0
S_1	0	1
S_2	1	0
S_3	1	1

State Diagram



Moore Model

State Table

From state diagram

From FF excitation table

Present State		Input	Next State		Output	FF Inputs	
A	B	x	A	B	y	D_A	D_B
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	1
0	1	0	0	0	0	0	0
0	1	1	1	0	0	1	0
1	0	0	0	0	0	0	0
1	0	1	1	1	0	1	1
1	1	0	0	0	1	0	0
1	1	1	1	1	1	1	1

D–FF Excitation Table

Q(t+1)	D
0	0
1	1

FF(s) Input/Output Equation(s):

$D_A(A, B, x) = \sum(3, 5, 7)$

$D_B(A, B, x) = \sum(1, 5, 7)$

$y(A, B, x) = \sum(6, 7)$

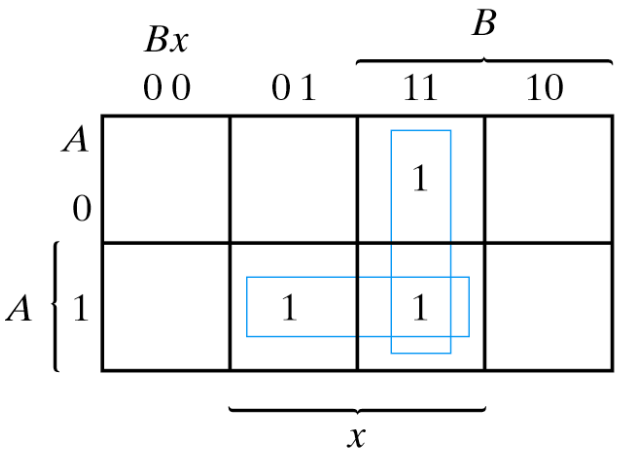
K-Maps for FF Input(s)

FF(s) Input/Output Equation(s):

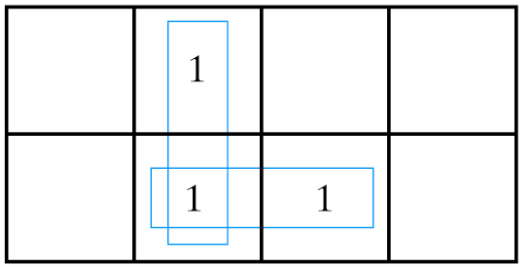
$D_A(A, B, x) = \sum(3, 5, 7)$

$D_B(A, B, x) = \sum(1, 5, 7)$

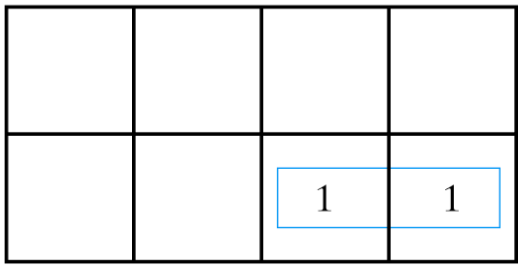
$y(A, B, x) = \sum(6, 7)$



$D_A = Ax + Bx$

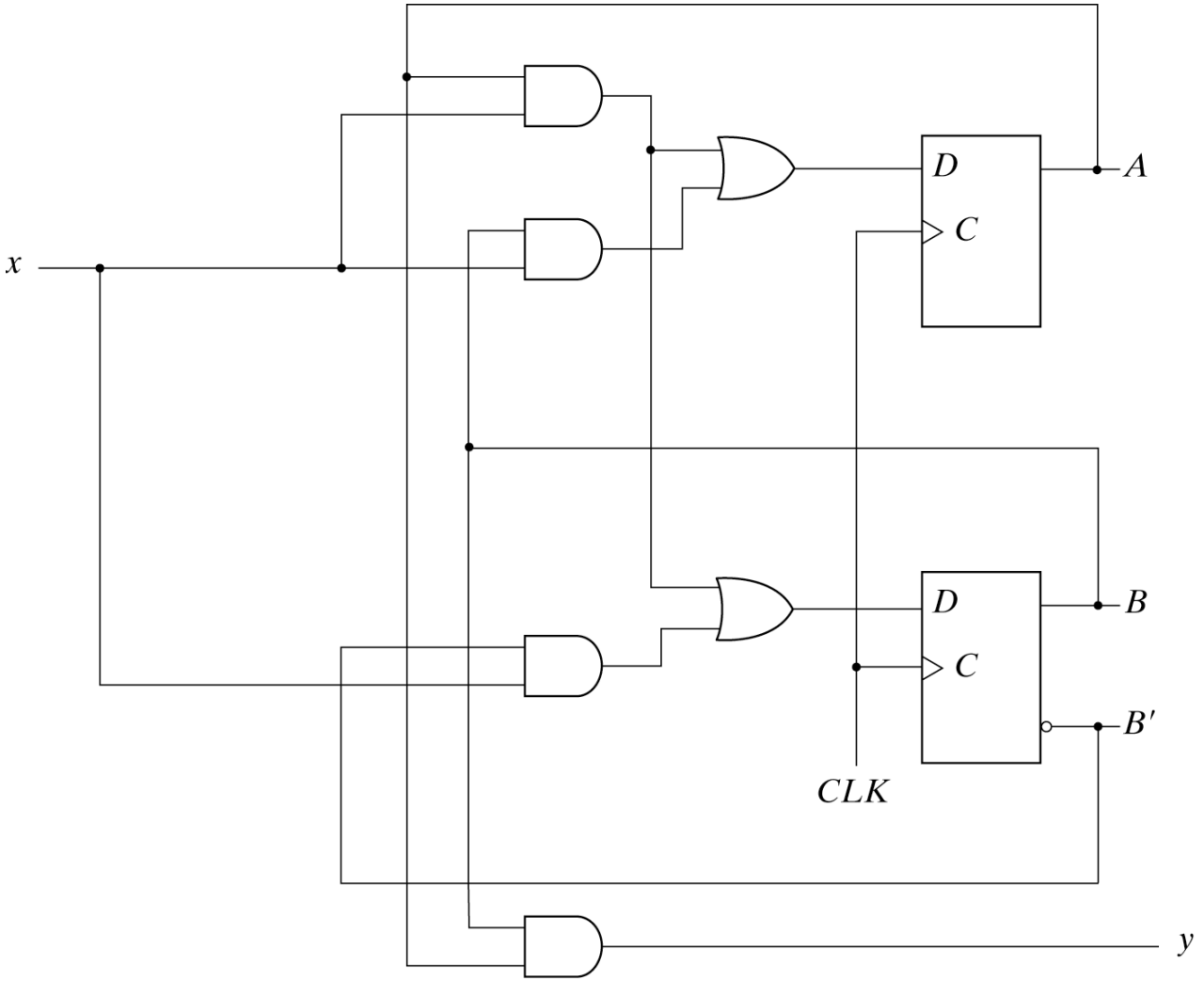


$D_B = Ax + B'x$



$y = AB$

Circuit (Logic) Diagram



Another State Encoding Technique

- Binary encoding uses the min possible number of FF's to store the state, however additional logic is required to encode or decode the state
- *One-hot encoding* is another state encoding technique, where only one bit of the "state vector" is set for any given state
- `n` states → `n` FF's

One-hot Vs Binary Encoding

One-hot

- Faster
- Speed independent of the # of states
- Easier to design/code the FSM
- Adding/deleting states is easier
- Easier debugging

Binary

- Slightly Cheaper
- Slower as the # of states gets larger
- More effort to design/code the FSM
- Adding/deleting states affects the rest of the machine
- More difficult debugging

Practice Assignment

- Solve the previous example using one-hot state encoding rather than binary state encoding

Example 2

- Using JK-FF(s), design the logic circuit that corresponds to the following state table

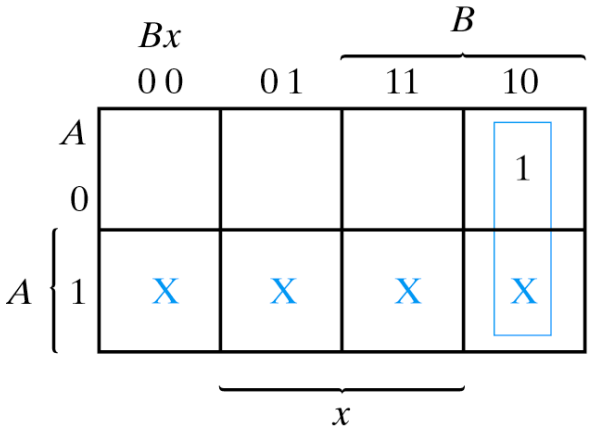
Present State		Input	Next State		FF Inputs			
A	B	x	A	B	J_A	K_A	J_B	K_B
0	0	0	0	0	0	X	0	X
0	0	1	0	1	0	X	1	X
0	1	0	1	0	1	X	X	1
0	1	1	0	1	0	X	X	0
1	0	0	1	0	X	0	0	X
1	0	1	1	1	X	0	1	X
1	1	0	1	1	X	0	X	0
1	1	1	0	0	X	1	X	1

From FF excitation table

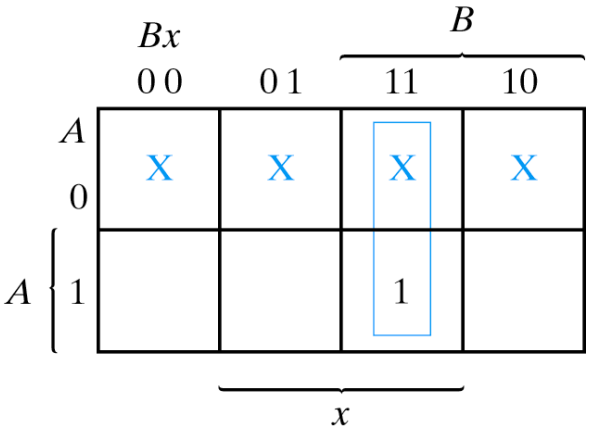
JK–FF Excitation Table

Q(t)	Q(t+1)	J	K
0	0	0	X
0	1	1	X
1	0	X	1
1	1	X	0

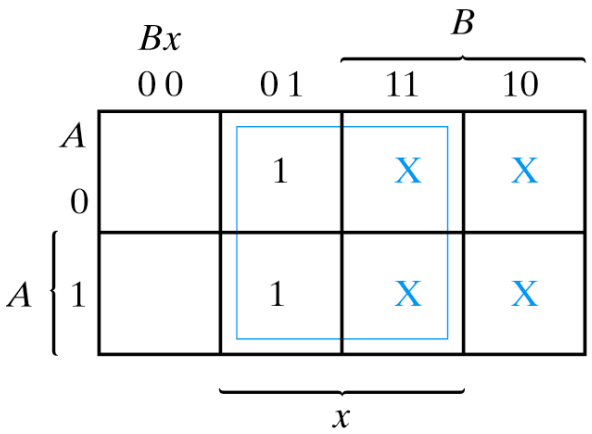
K-Maps for FF Input(s)



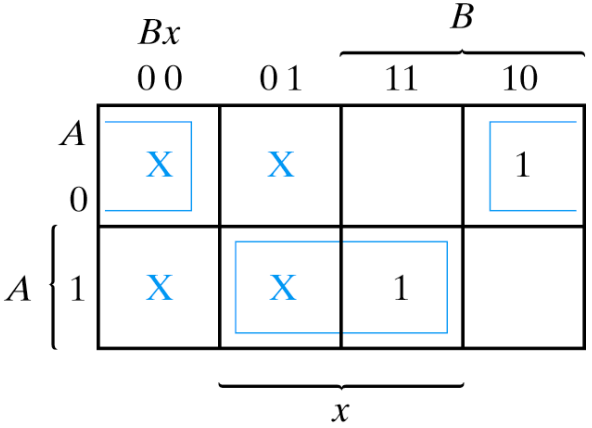
$J_A = Bx'$



$K_A = Bx$

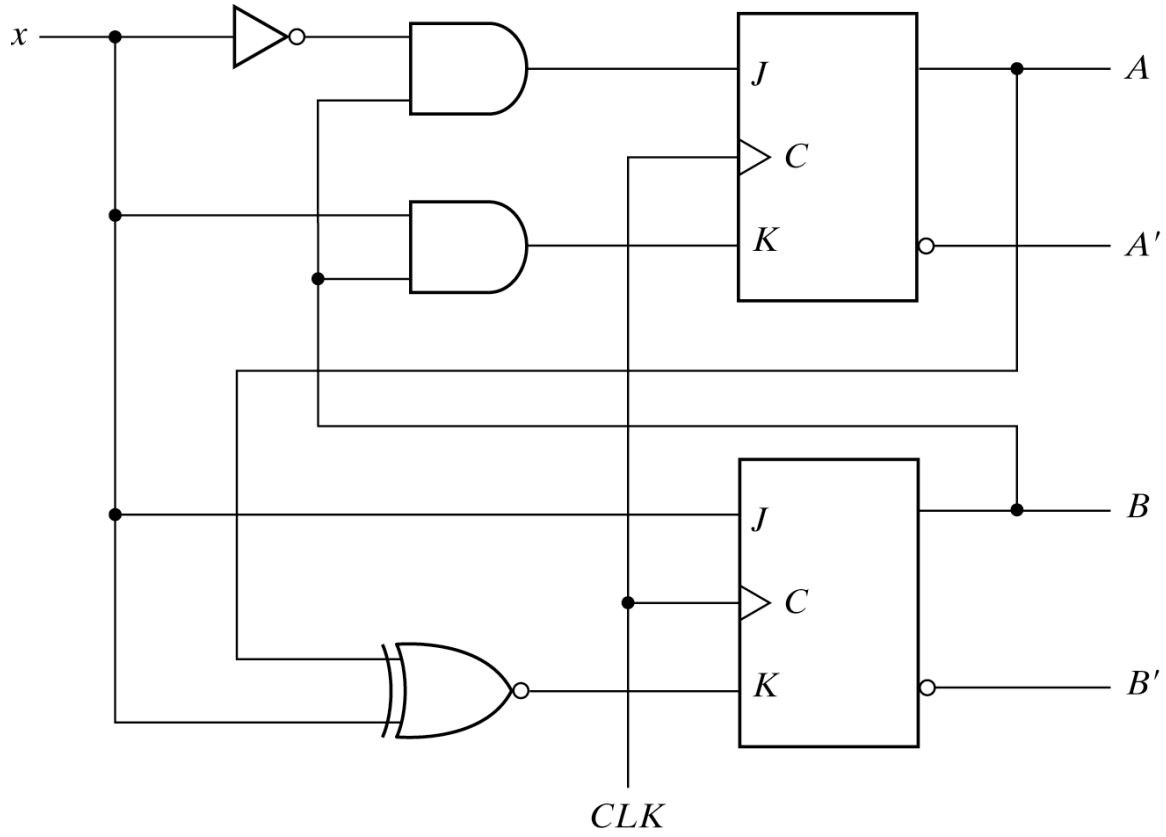


$J_B = x$



$K_B = (A \oplus x)'$

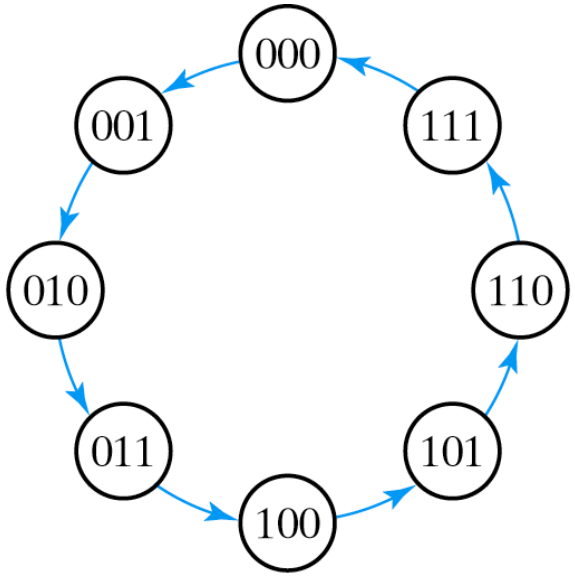
Circuit (Logic) Diagram



Example 3

- Using T-type FFs, design a 3-bits binary counter that can count in binary from 0 to 7

State Diagram



State Table

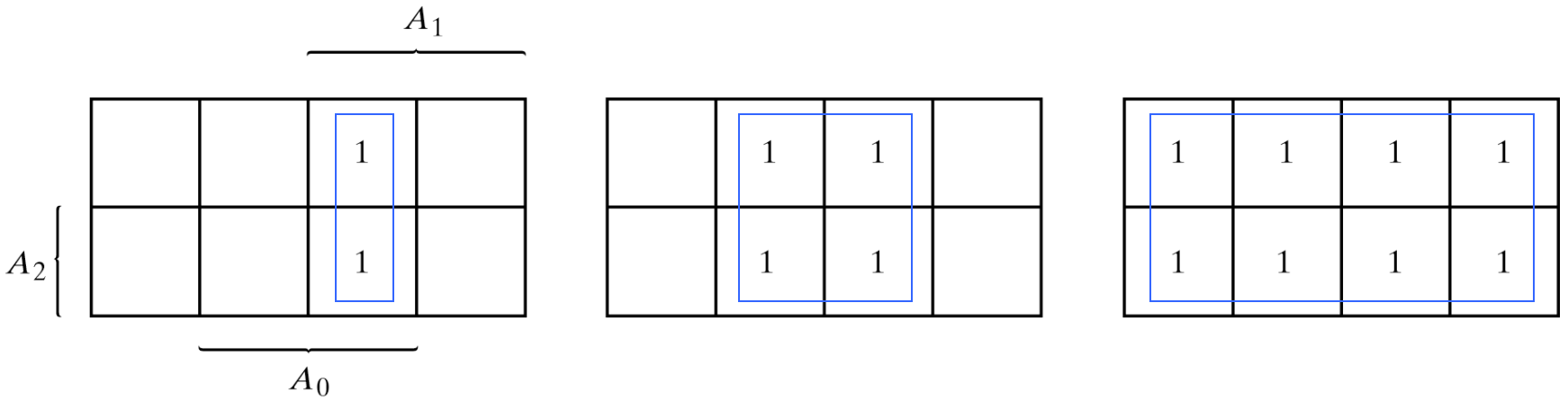
Present State			Next State			FF Inputs		
A_2	A_1	A_0	A_2	A_1	A_0	T_{A2}	T_{A1}	T_{A0}
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	1
0	1	0	0	1	1	0	0	1
0	1	1	1	0	0	1	1	1
1	0	0	1	0	1	0	0	1
1	0	1	1	1	0	0	1	1
1	1	0	1	1	1	0	0	1
1	1	1	0	0	0	1	1	1

From FF excitation table

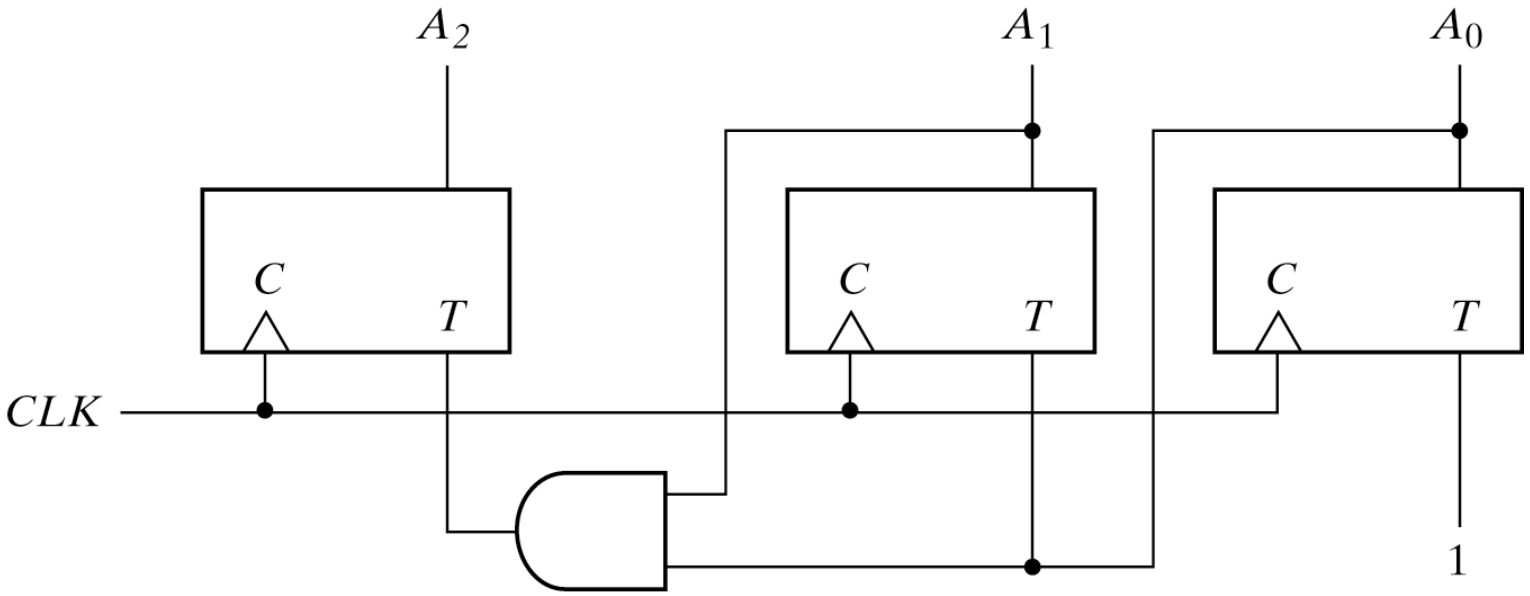
T–FF Excitation Table

Q(t)	Q(t+1)	T
0	0	0
0	1	1
1	0	1
1	1	0

K-Maps for FF Input(s)



Circuit (Logic) Diagram



Important Book Chapter

- Chapter 5 in textbook

Guidelines for the Midterm

Please read carefully before proceeding.

1. The duration of this exam is 90 minutes.
2. Point value of this exam is tentatively (30%).
3. Only un-programmable calculators are permitted for this exam.
4. You can use the back and/or the front of any exam sheet(s) (except pages 1– 3) for your scratch.
5. The exam consists of EIGHTEEN multiple choice questions, in SIXTEEN pages (including the cover page). Please inform your proctor if you have any missing page(s) or question(s). You will lose the grade of any question(s) that are located in missing page(s).

Exam Strategy

Please insert all your answers for the exam in the table(s) that are provided in the next page. Your grade in this exam will depend only on the answers that are indicated in the table(s). Tick (or cross) the table cell that corresponds to the answer that you believe is the most suitable.

More Guidelines

a. **If you solve any 15 out of the 18 questions correctly, you will get 100% of the total grade.**

b. Read all the *questions* **carefully**. For problems where Verilog HDL code is provided, pay attention to the comments inside the code and the modules' names.

c. Read all the *choices* **carefully**. There are choices that "*look*" alike, however they are obviously not!

d. This exam is designed so that the questions cover a broad range of difficulty-levels. Manage your time so that you do not lose the grades of the relatively-easy questions because of getting stuck at relatively-difficult questions.

e. Unless otherwise can be read from the waveforms, assume that shaded/patterned parts correspond to don't cares (X: unknowns).

f. Assume the existence of the following statement before all the given Verilog HDL modules: `` `timescale 1ns / 1ps ``

Exam Style!

- Broad selection of MCQ's based on topics delivered in lectures 1-8
- Various types of problems
  - Theory
  - Problems to assess understanding of theoretical principles provided
  - Lots of Verilog! Maybe in the question(s) or the answer(s)!
  - Lots of Timing Simulations! Maybe in the question(s) or the answer(s)!

Final Advice!

- Read question(s) well
- Read *all* choices. Pay attention to “None of the previous”, “All of the previous”, (i) and (ii), etc.
- Think first, don’t bang your head against the wall
- Good Luck!

References

- Digital Design, M. Morris, Mano
- http://bawankule.com/verilogfaq/files/jhld099401.pdf