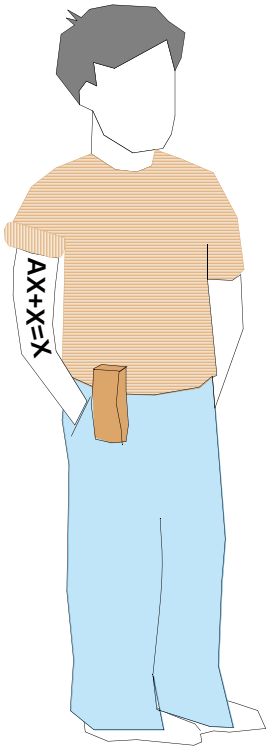
Exam Review



Writing Exams

State-Graph Construction

Small Boolean Problems

Multiple Outputs,

Synchronous Circuits

Asynchronous Circuits

Races, Cycles and State Assignment

1 hr 45 min

CARLETON UNIVERSITY

Final
EXAMINATION
April, 13 2009; 14:00

Name:
Number:
Signature:

DURATION: 3 HOURS **No. of Students: 250**

Department Name & Course Number: Electronics ELEC 2607 A,B and C

Course Instructor(s) J.W. Rogers and J. Knight

AUTHORIZED MEMORANDA **TURN OFF cell phones and personal communications equipment and LEAVE THEM AT THE FRONT. Notes, books, and non-communicating calculators are allowed.**

Students MUST count the number of pages in this examination question paper before beginning to write, and report any discrepancy immediately to a proctor. This question paper has 10 pages.

This examination paper May Not be taken from the examination room.

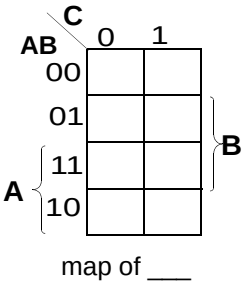
In addition to this question paper, students require: an examination booklet yes ☐ no ☒ *may request*
a Scantron sheet yes ☐ no ☒

Please answer on the examination paper. If your answer does not fit, check you are not using a long hard method. You may ask for a booklet if you need it.

1 Boolean

For ALL questions: If you use a map, indicate for which function it is.
If you use algebra, indicate the rules used at the right side of each line.

Especially for Absorption, D2, Swap, Reduction and other less obvious ones. Marks will be deducted if you don't!



2% a) Write the simplest

Writing Exams

Time yourself: 180 min, 100%
150 min + 1/2 hour for cleanup
15 min for 10%.
3 min for 2%.

Give Up: If you can't do it, move on.

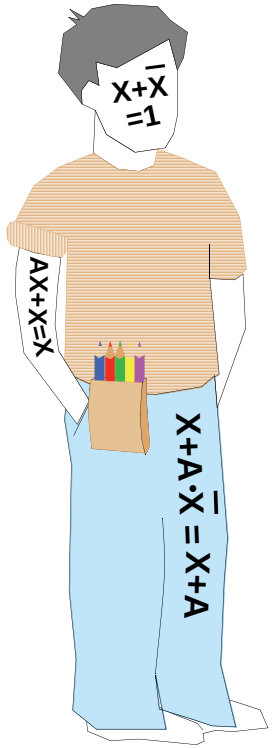
RTFQ: Read The Foolish Question;
read it again DAMN IT.

What was asked for?

- Just the state table?
  Don't make K-maps.
- Wants a circuit.
  Then draw it!
- Ask for P of S (Π of Σ)
  Don't give it $A + BCD + A\overline{D} + Q$ (Σ of Π)
  or $(A + BCD)(\overline{AD} + Q)$

Boolean Algebra

Exam Ready



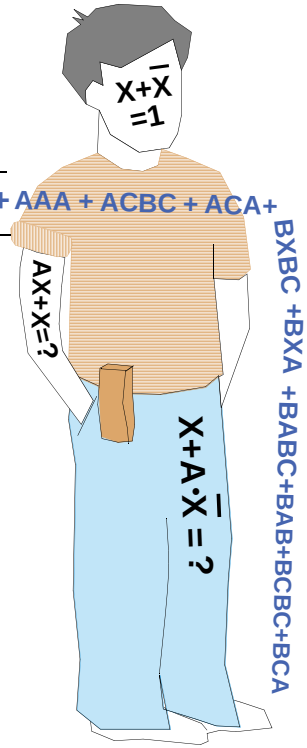
<u>Look for simple methods</u>:

Don't do all algebra questions by multiplying out.

(X+A+B)(X+A+C)(BC+A)=

Boolean Algebra

Exam Unready



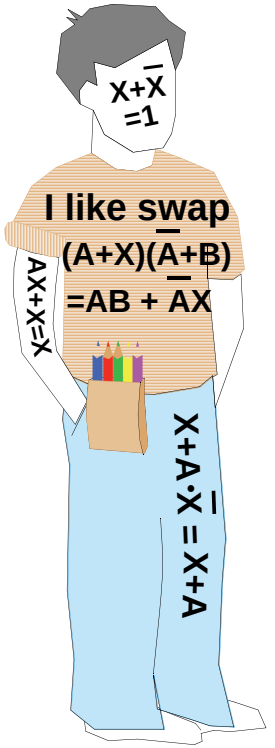
Look for simple methods:

Don't do all algebra questions by multiplying out.

(X+A+B)(X+A+C)(BC+A)=XXABC+XXA+XABC+XAA+XCBC+XCA+AXBC+AXA+AABC+ AAA + ACBC + ACA+

Boolean Algebra

Exam Ready



Look for simple methods:

Don't do all algebra questions by multiplying out.

(X+A+B)(X+A+C)(BC+A)=

Simplify at each step:

Right elbow: X+anything·X = X

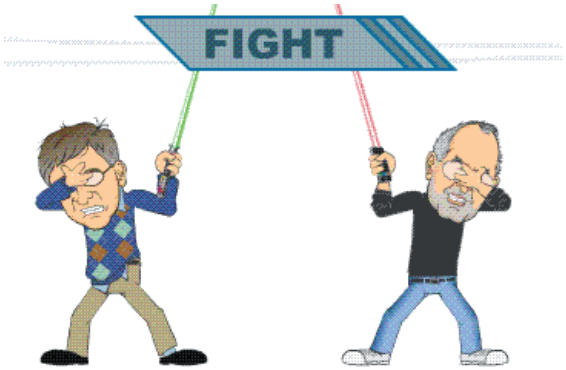
Left leg: X + anything·$\overline{X}$ = X + anything

Nose: X+$\overline{X}$ = 1

Shirt: Swap

Maybe Use Dual

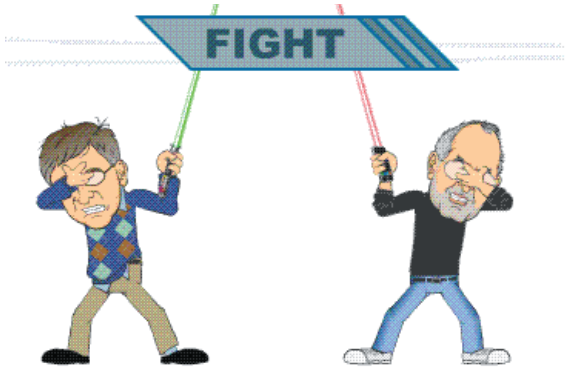
But don't be Duel Happy



Duels don't solve all problems

Boolean Algebra

Dual Usage



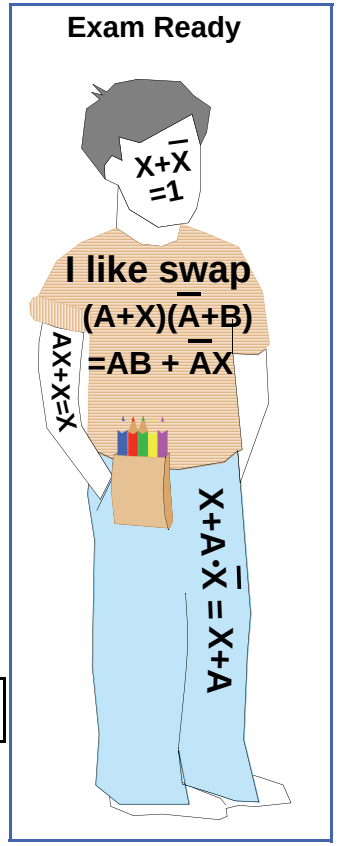
- Not for everything

- Use the dual for:.

(1) Simplifying things with many brackets
- but look for swap first

(2) Factoring,
- but simplify first

Lots of brackets



(X+A+B)(X+A+C)(BC+A) =

= (X+A+B)(X+A+C)((BC)+A) *Bracket ALL ANDs*

F_{dual} = XAB+XAC+(B+C)A *Take dual*

= XAB+XAC+BA+CA *Dist 1*

= BA+CA *BA+anything·BA = BA*
CA+anything·CA = CA

$(F_{dual})_{dual}$ = F = (B+A)(C+A)

Boolean Algebra

First look for simplifications.
Are there three cm space for the answer?
Do you need thirty? **No! see simplification =>**

.

Boolean (1% for stating rules and map names.)

If you use algebra, indicate the rules used at the right side of each line.

1.5% a) Simplify $f = BC + BC(\overline{A} + BD)$

$=$ *rule:*

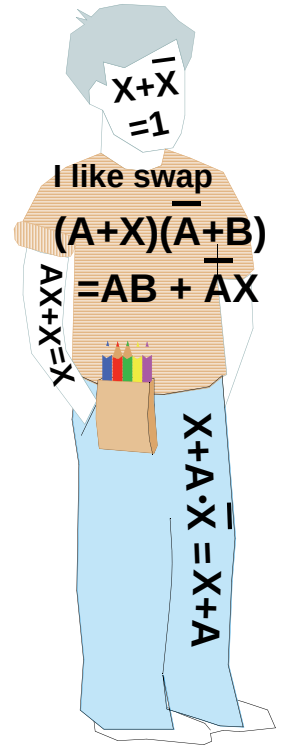
2% b) Simplify $g = BC + \overline{B}C(\overline{A+D})\overline{A}$

$=$

2% c) Simplify $h = (A + B) + \overline{A}\cdot\overline{B}\cdot E\cdot F\cdot G$

$=$

Exam Ready



Common Mistakes

1. Not using $X + XA = X$ to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as $\overline{a} \cdot \overline{b}$

$$A \cdot B + \overline{A} \cdot \overline{B} \cdot any = A \cdot B + any \quad (S)$$

Common Mistakes

1. Not using X + XA =XA to simplify expressions the FIRST THING.
2. Not reducing using X + $\overline{X}$R = X + R.
3. Saying $\overline{a \cdot b}$ is the same as ~~$\overline{a} \cdot \overline{b}$~~

A·B + $\overline{A}$·$\overline{B}$·any = ~~A·B + any~~ (BS)

Common Mistakes

1. Not using X + XA =XA to simplify expressions the FIRST THING.
2. Not reducing using X + $\overline{X}$R = X + R.
3. Saying $\overline{a \cdot b}$ is the same as $\overline{a} \cdot \overline{b}$

   a•b + $\overline{a} \cdot \overline{b}$•any = a•b + any (S)
4. Saying an expression is equal to its dual.

F= A•B•C + (A•C + D)•E

F= {A•B•C} + [({A•C} + D)•E]

F= {A + B + C}•[({A + C}•D) + E] *take dual*

Common Mistakes

1. Not using X + XA =XA to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as ~~$\overline{a} \cdot \overline{b}$~~

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + any$ (BS)
4. Saying an expression is equal to its dual.

$F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

$F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

$F_{dual} =$ ~~$F =$~~ $\{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*

Common Mistakes

1. Not using X + XA =XA to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as $\overline{a} \cdot \overline{b}$

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + any$ (S)
4. Saying an expression is equal to its dual.

   $F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

   $F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

   $F = \{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*
5. Saying ~~$X + 1 = X$~~

   Simplify $g = (1 + X)(A\overline{X})$

   Everybody knows better than this, but they still do it.

Common Mistakes

1. Not using $X + XA = XA$ to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as ~~$\overline{a} \cdot \overline{b}$~~

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + any$ (BS)
4. Saying an expression is equal to its dual.

   $F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

   $F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

   $F_{dual} =$ ~~$F =$~~ $\{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*
5. Saying ~~$X + 1 = X$~~

   Simplify $g = (1 + X)(A\overline{X}) = (1)(A\overline{X}) = A\overline{X}$

   Everybody knows better than this, but they still do it.

Common Mistakes

1. Not using $X + XA = XA$ to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as $\overline{a} \cdot \overline{b}$

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + any$ (S)
4. Saying an expression is equal to its dual.

   $F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

   $F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

   $F = \{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*
5. Saying ~~$X + 1 = X$~~

   Simplify $g = (1 + X)(A\overline{X})$

   Everybody knows better than this, but they still do it.
6. When you take dual, or general Demorgan, do not put in the brackets in your head.

   $F = (AB + C) + DE \Rightarrow F_{dual} = (A+BC)(D+E)$?

Common Mistakes

1. Not using X + XA =XA to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as ~~$\overline{a} \cdot \overline{b}$~~

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + any$ *(BS)*
4. Saying an expression is equal to its dual.

   $F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

   $F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

   $F_{dual} =$ ~~$F =$~~ $\{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*
5. Saying ~~$X + 1 = X$~~

   Simplify $g = (1 + X)(A\overline{X}) = (1)(A\overline{X})$

   Everybody knows better than this, but they still do it.
6. When you take dual, or general Demorgan, do not put in the brackets in your head.

   $F = (AB + C) + DE \Rightarrow F_{dual} = $ ~~$(A+BC)(D+E)$~~

   $F = (AB + C) + DE = ((AB) + C) + (DE) \Rightarrow F_{dual}\ ((A+B)C)(D+E)$ *(D2)*

Common Mistakes

1. Not using X + XA =XA to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as $\overline{a} \cdot \overline{b}$

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + any$ (S)
4. Saying an expression is equal to its dual.

   $F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

   $F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

   $F = \{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*
5. Saying ~~X + 1 = X~~

   Simplify $g = (1 + X)(A\overline{X})$

   Everybody knows better than this, but they still do it.
6. When you take dual, or general Demorgan, do not put in the brackets in your head.

   $F = (AB + C) + DE \Rightarrow F_{dual} = (A+BC)(D+E)$?
7. Not knowing D2.. $X + AB = (X + A)(X + B)$

   Factor $h = \overline{A} + BC\overline{E}$

Common Mistakes

1. Not using $X + XA = XA$ to simplify expressions the FIRST THING.
2. Not reducing using $X + \overline{X}R = X + R$.
3. Saying $\overline{a \cdot b}$ is the same as ~~$\overline{a} \cdot \overline{b}$~~

   $a \cdot b + \overline{a} \cdot \overline{b} \cdot any = a \cdot b + ~~any~~$ (BS)
4. Saying an expression is equal to its dual.

   $F = A \cdot B \cdot C + (A \cdot C + D) \cdot E$

   $F = \{A \cdot B \cdot C\} + [(\{A \cdot C\} + D) \cdot E]$

   $F_{dual} = ~~F~~ = \{A + B + C\} \cdot [(\{A + C\} \cdot D) + E]$ *take dual*
5. Saying ~~$X + 1 = X$~~

   Simplify $g = (1 + X)(A\overline{X}) = (1)(A\overline{X})$

   Everybody knows better than this, but they still do it.
6. When you take dual, or general Demorgan, do not put in the brackets in your head.

   $F = (AB + C) + DE \Rightarrow F_{dual} = ~~(A+BC)(D+E)~~$

   $F = (AB + C) + DE = ((AB) + C) + (DE) \Rightarrow F_{dual} ((A+B)C)(D+E)$
7. Not knowing D2. $X + AB = (X + A)(X + B)$

   Factor $h = \overline{A} + BC\overline{E}$ $h = (\overline{A} + B)(\overline{A} + C\overline{E})$

   $= (\overline{A} + B)(\overline{A} + C)(\overline{A} + \overline{E})$

Boolean Algebra Practice

1. Simplify $(A + C + \overline{C} \cdot \overline{A})$

2. Simplify $\overline{CD} + CE$

3. Take the dual of $F=(A + QB)(X + Y) + \overline{Z}$

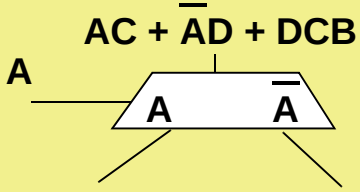
4. Factor $X + B\overline{C}$

5. Factor $X + AB\overline{C}$

6. Find the dual of Simply $G=(A + 1)(B + C) + \overline{D}$

7. Factor $AB + CD$

8. Construct with MUXs
   Extra hardware lowers mark



Boolean Algebra Practice

1. Simplify $(A + C + \overline{C}\cdot\overline{A})$

$(A + C + \overline{A}) = (A + \overline{A} + C) = 1$ rule $C+\overline{C}A=C+A$, rule $1+A=1$

2. Simplify $\overline{CD} + CE$

$\overline{C} + \overline{D} + CE = \overline{C}+\overline{D}+E$ rule DeMorg, rule $C+\overline{C}E=C+E$

3. Take the dual of $F=(A + QB)(X + Y) + \overline{Z}$

$F=\{(A + (QB))(X + Y)\} + \overline{Z}$ bracket ANDs

$F_{dual}=\{(A(Q+B))+(XY)\}\overline{Z}$

4. Factor $X + B\overline{C}$

$(X + B)(X + \overline{C})$ rule D2

5. Factor $X + AB\overline{C}$

$(X + AB)(X + \overline{C})$ rule D2

$=(X + A)(X + B)(X + \overline{C})$ rule D2

6. Find the dual of Simply $G=(A + 1)(B + C) + \overline{D}$

$\{(A + 1)(B + C)\} + \overline{D}$ bracket ANDs

$G_{dual}=\{(A\cancel{0})+(BC)\}\overline{D}$ dual $1\leftrightarrow0$

$=BC\overline{D}$ rules $A0=0$; $x+0=x$

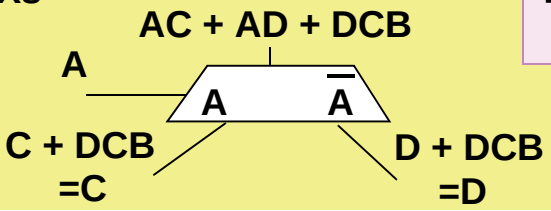
7. Factor $AB + CD$

$(AB + C)(AB + D)$ rule D2

$=(A + C)(B + C)(A + D)(B + D)$ rule D2

8. Construct with MUXs

Extra hardware lowers mark

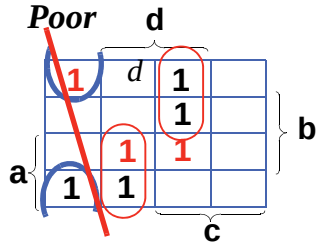


(S)

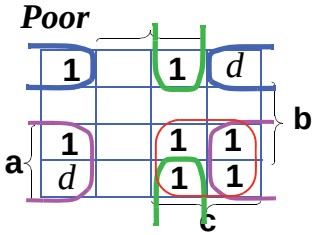
K-Maps; Common Errors

- **Check Your Map Entries**
  One variable in the wrong square, you're toast!

- **Check for Wrap Around,**
  and <u>Wider</u> Wrap Around



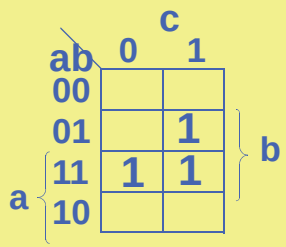
What is poor?



You find two better loops

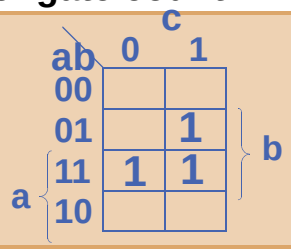
- **Using algebra after map simplification.**
  Usually no help:
  A K-map gives the simplest Σ of Π circuit.

Simplify $ab + \bar{a}bc$
to the simplest Σ of Π circuit



A bit of factoring might lower gate count.

Simplify $ab + \bar{a}bc$
to the simplest circuit
Minimum gates
and letters.

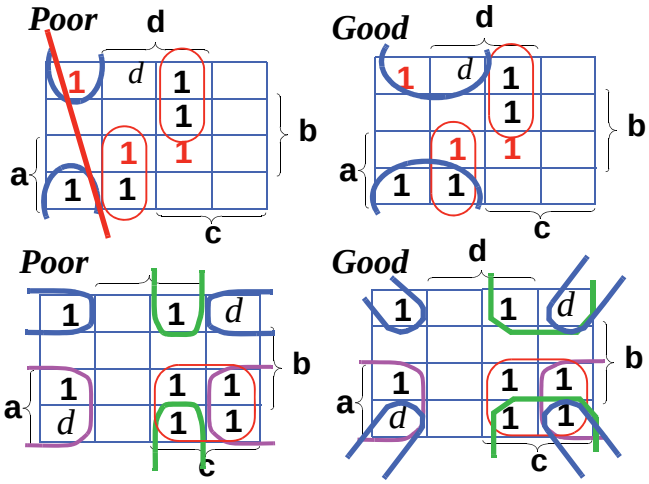


$ab + ac$

K-Maps; Common Errors

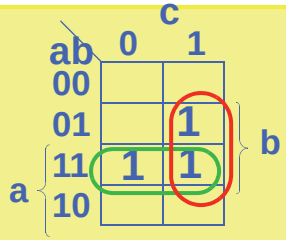
- **Check Your Map Entries**
  One variable in the wrong square, you're toast!

- **Check for Wrap Around,**
  and <u>Wider</u> Wrap Around



- **Using algebra after map simplification.**
  Usually no help:
  A K-map gives the simplest Σ of Π circuit. (If you loop correctly)

Simplify $ab + \bar{a}bc$
to the simplest Σ of Π circuit



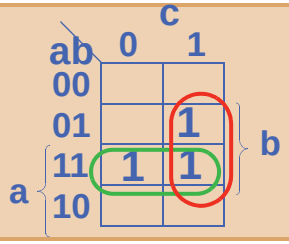
ab + bc

3 gates
4 letters

ab + bc = (ab +b)(ab+c) *(D2)*
= (b)(ab+c) *(S)*
= (ab+bc) *(D1)*
So what

A bit of factoring sometimes lowers gate count.

Simplify $ab + \bar{a}bc$
to the simplest circuit,
Minimum gates
and letters.



Use D1

ab + ac = a(b + c)

2 gates
3 letters

K-Maps; Common Errors

- **Do not confuse 5-variable maps and dual-output maps.**

5-variable **Bigger is always Better,**
Enlarge loops to use both maps.

Multiple output

Share loops, saves gates
Smaller is Sometimes Simpler.

- sharing lowers gate count.
- sharing increase may letter count. } **Balance these**

12 gates, none shared

28 letters

9 gates, 3 shared

25 letters

K-Maps; Common Errors

- **Do not confuse 5-variable maps and dual-output maps.**

5-variable **Bigger is always Better,**
Enlarge loops to use both maps.

Multiple output
Share loops, saves gates
Smaller is Sometimes Simpler.
- sharing lowers gate count.
- sharing increase may letter count. } **Balance these**

12 gates, none shared

28 letters

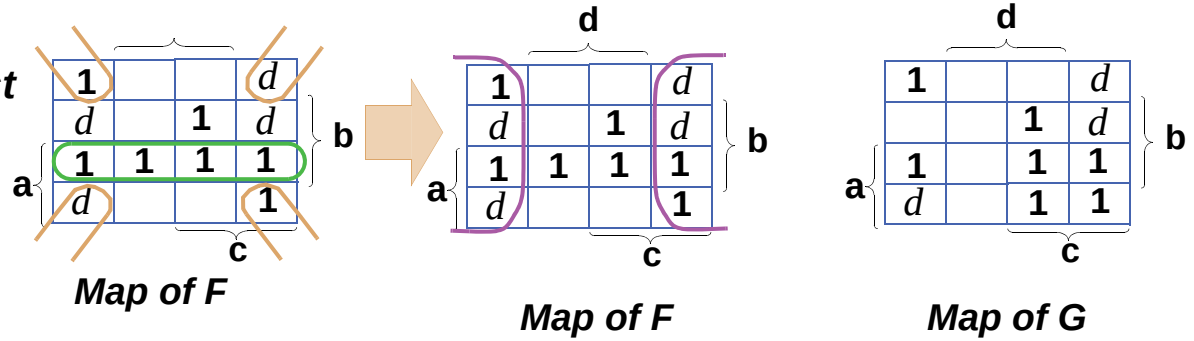
9 gates, 3 shared

25 letters

K-Maps; Common Errors

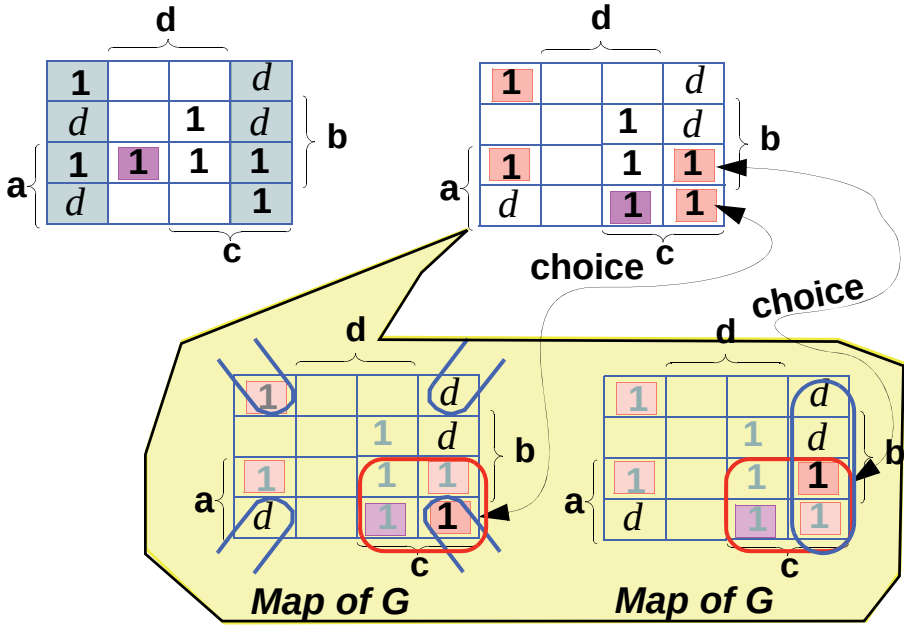
- **Multiple Output Mapping Rules**

Do Half-Maps First
(Except for PLAs)



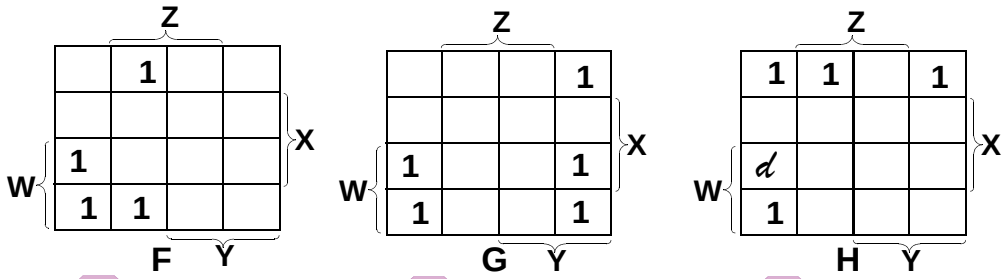
Then do lone "1"s

* ***No friend*** 1
* ***Only friends gone to dark side*** 1



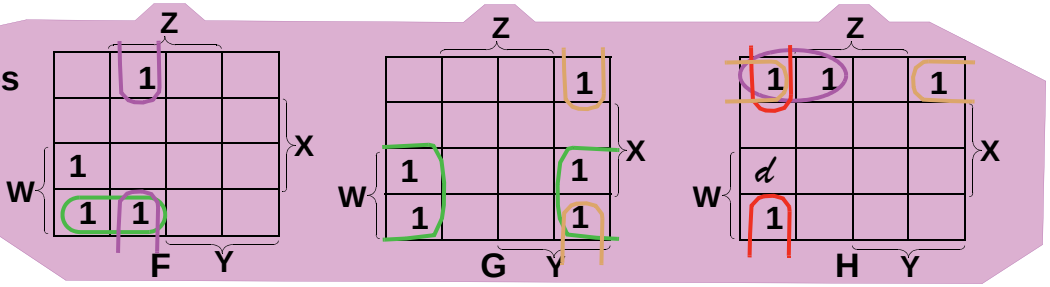
* ***No friends looping***
  May give a choice.
  Do squares with no choice first

Find the minimum Σ of Π



Poor Method

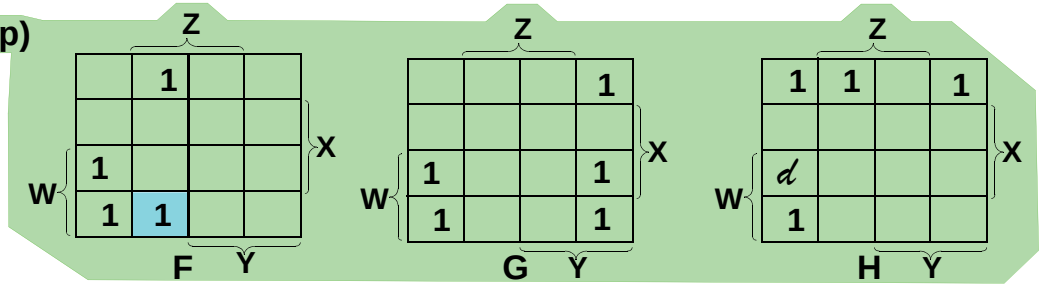
Largest circles
7 AND terms



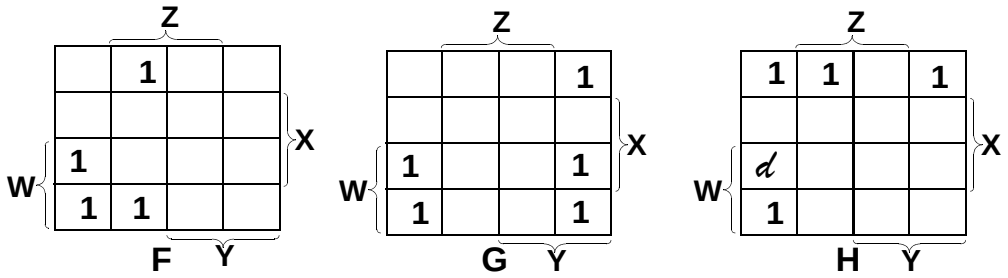
Better Method

(1) Circle half maps (none)

(2) No Friends, (“1”s on only one map)



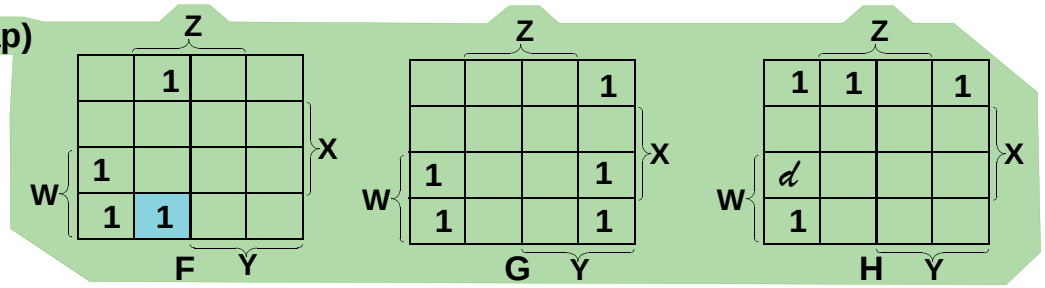
Find the minimum Σ of Π



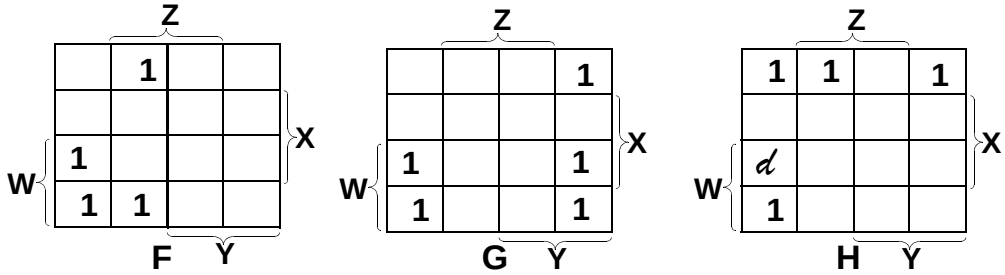
Better Method

(1) Circle half maps (none)

(2) No Friends, (“1”s on only one map)



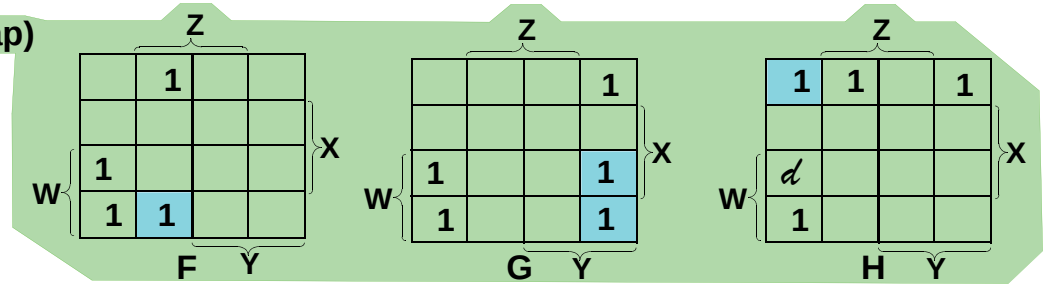
Find the minimum Σ of Π



Better Method

(1) Circle half maps (none)

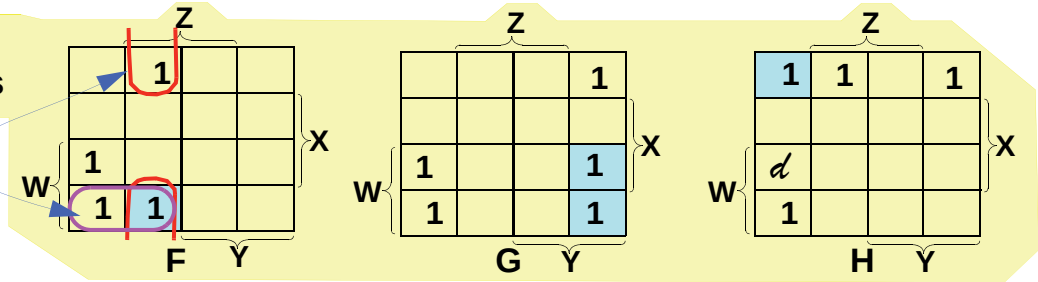
(2) No Friends, (“1”s on only one map)



Loop Friendless “1”s , they will never share

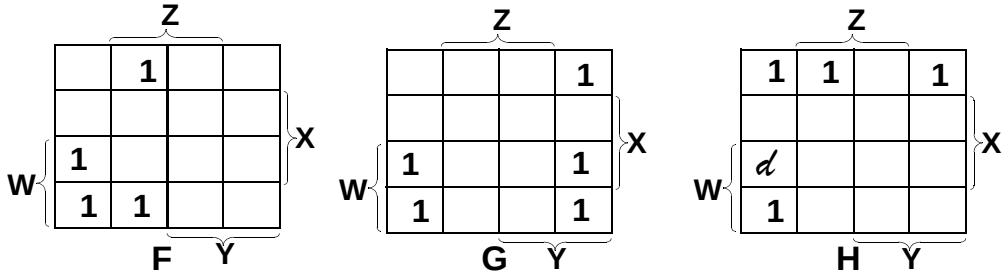
Unfortunately you have choices; several ways to loop some squares

Choices



Sketch t he Output

Find the minimum Σ of Π



Better Method

(1) Circle half maps (none)

(2) No Friends, (“1”s on only one map)

Loop Friendless “1”s , they will never share

Unfortunately you have choices; several ways to loop some squares

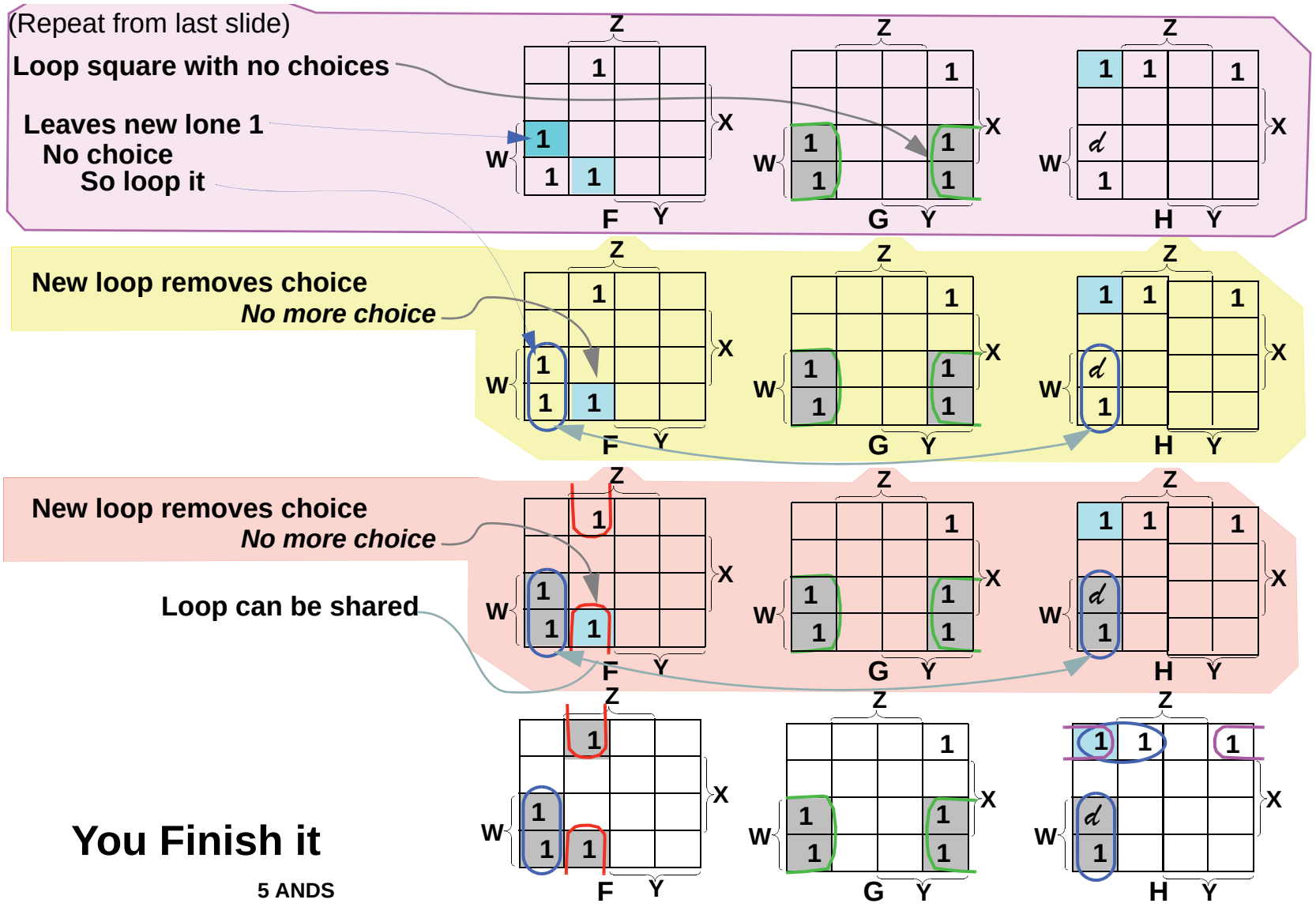
Choices

Choices

First

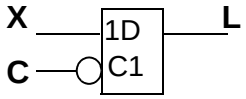
Loop square with no choices

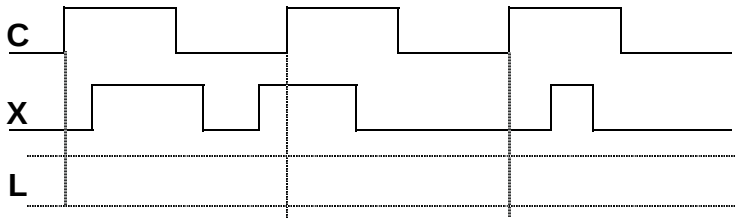
Leaves new lone 1 With one choice Loop it


(Repeat from last slide)
Loop square with no choices
Leaves new lone 1
No choice
So loop it
New loop removes choice
No more choice
New loop removes choice
No more choice
Loop can be shared
You Finish it
5 ANDS
Z
W
X
F
G
H
Y
d

Example.1-32a Sketch the Output Waveform

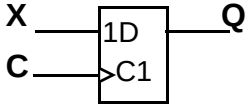
A

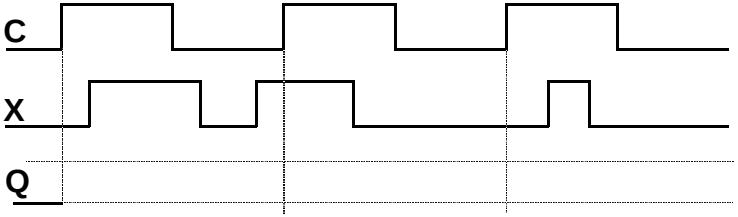




D Latch (inverted clock)
Transparent when?
Latched when?

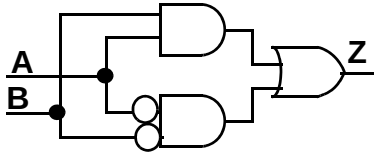
B

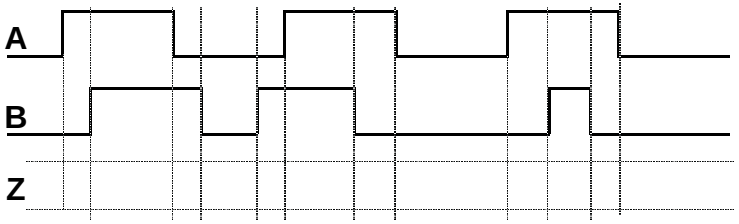




D Flip-Flop
_____ edge triggered

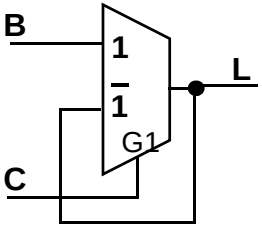
C

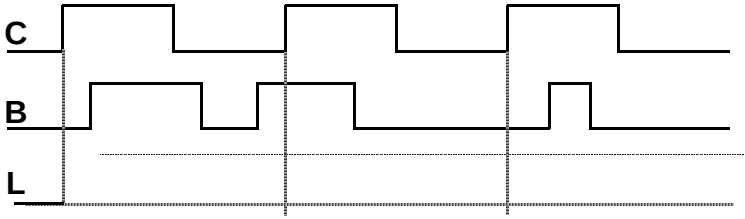




Equilivalent to
_____ gate

D



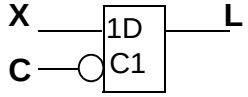


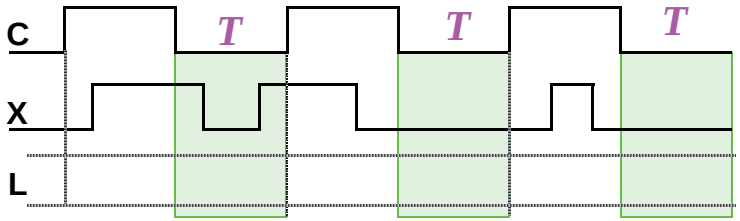
D Latch
Transparent when ___
Latched when____

Latches etc from Dec '96.

Example.1-32b Sketch the Output Waveforms

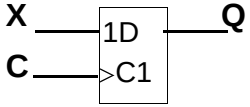
A

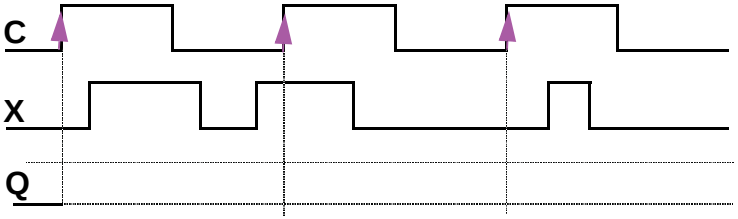




D Latch (inverted clock)
Transparent when C low
Latched when C high

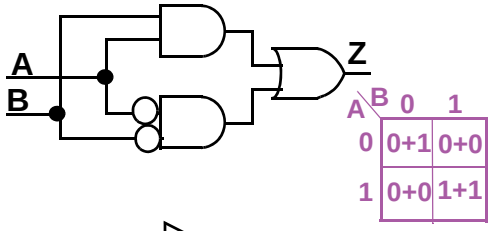
B



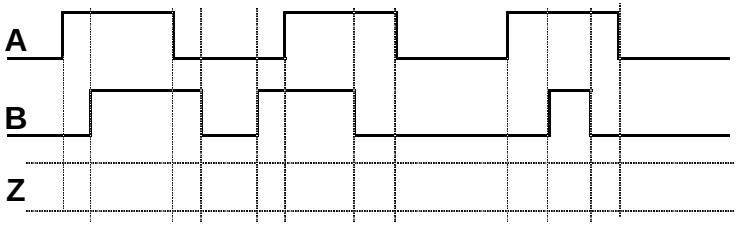


D Flip-Flop
Rising edge triggered

C

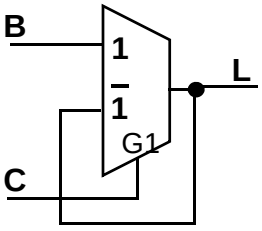


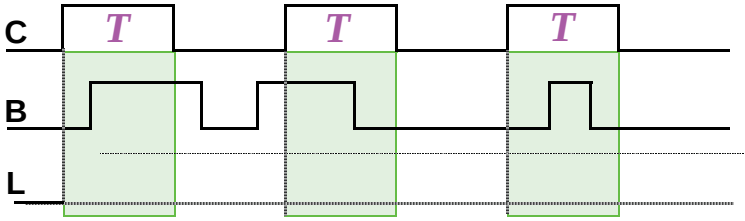
A \ B	0	1
0	0+1	0+0
1	0+0	1+1



XNOR gate

D

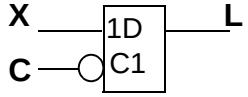


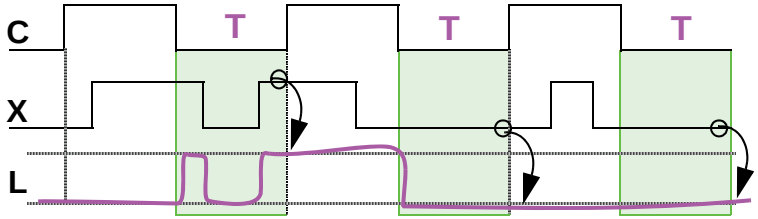


D Latch
Transparent when C=1
Latched when C=0

Example.1-32c Sketch the Output Waveforms

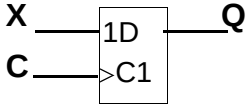
A

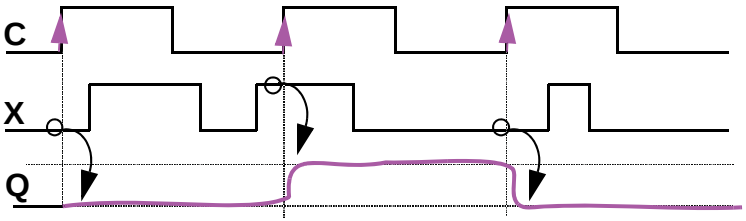




D Latch (inverted clock)
Transparent when clock low
Latched when high

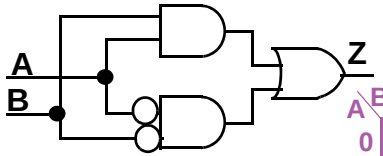
B



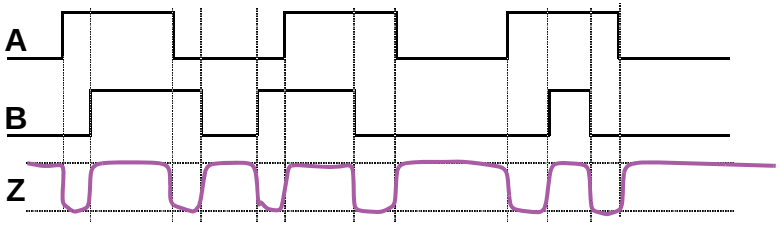


D Flip-Flop
Rising edge triggered

C

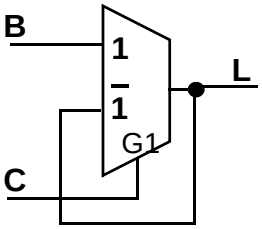


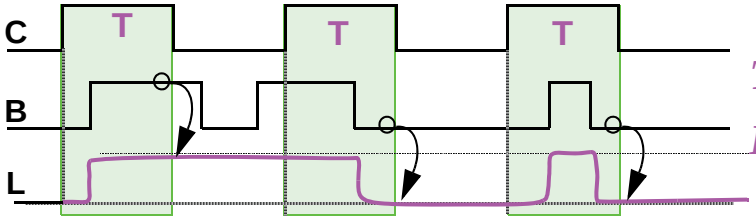
A \ B	0	1
0	1	0
1	0	1



XNOR gate

D

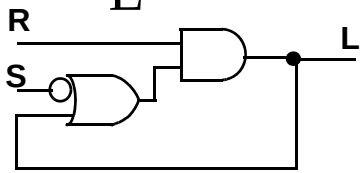


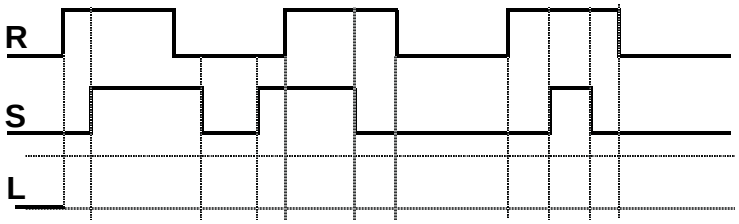


D Latch
Transparent when C=1
Latched whenC=0

Example.1-33a Sketch the Output Waveforms

E

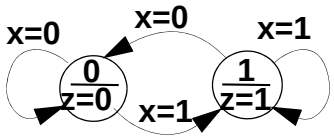


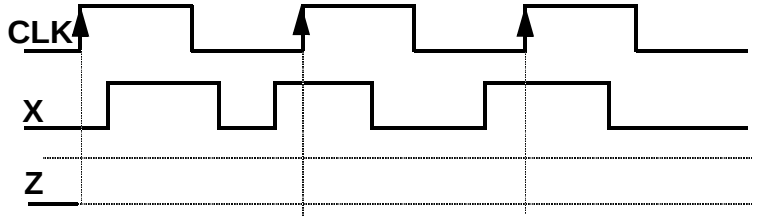


Reset(L)-Set(L) Latch
Reset L = 0 when R=0
Set L =1 when RS=10
Store output when SR=11

F

Synchronous State Graph

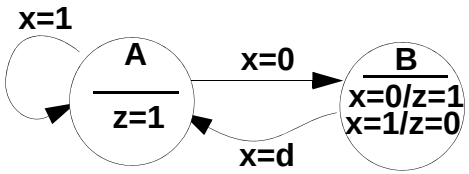


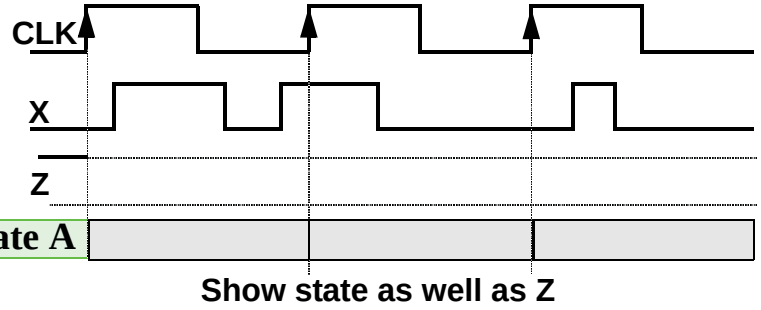


State=Z

Synchronous State Graph

G



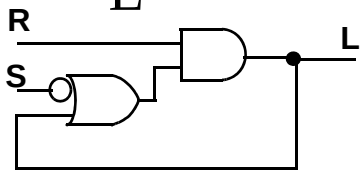


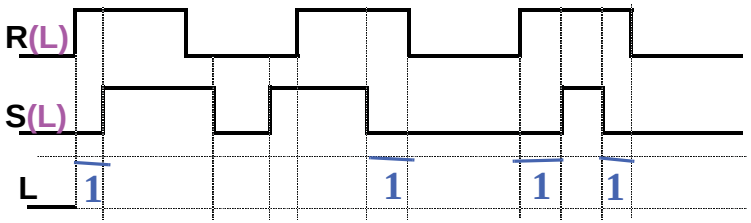
State A
State B

Show state as well as Z

Example.1-33b Sketch the Output Waveforms

E





Reset(L)-Set(L) Latch

Set L =1 when RS=10

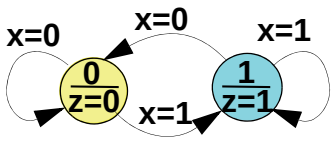
Reset L = 0 when R=0

Store output when SR=11

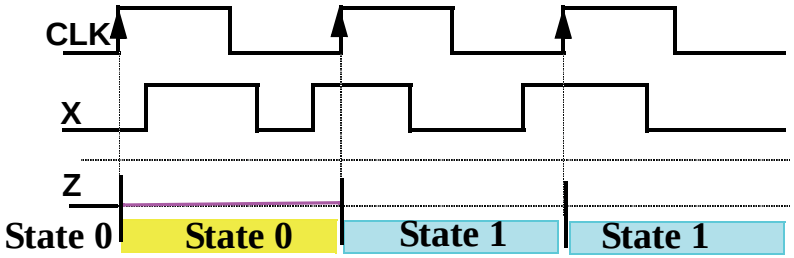
Set dominant

F

Synchronous State Graph



Sketch the circuit.

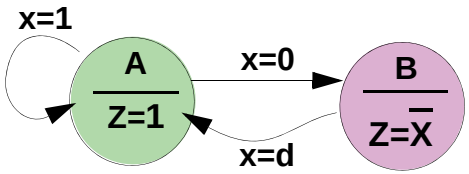


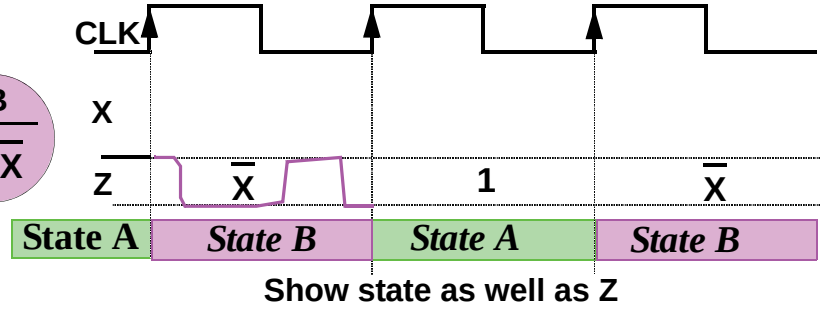
State=Z

State only changes on the clock edge

Synchronous State Graph

G





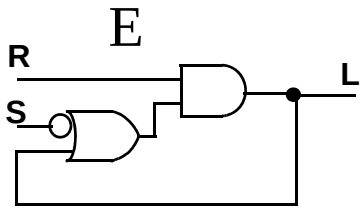
State A

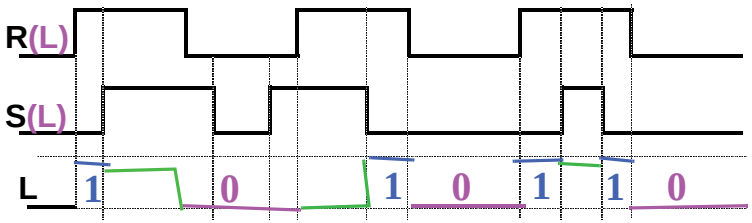
State B

Show state as well as Z

Example.1-33c Sketch the Output Waveforms

E

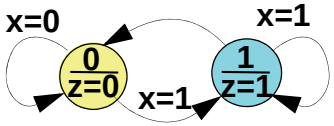




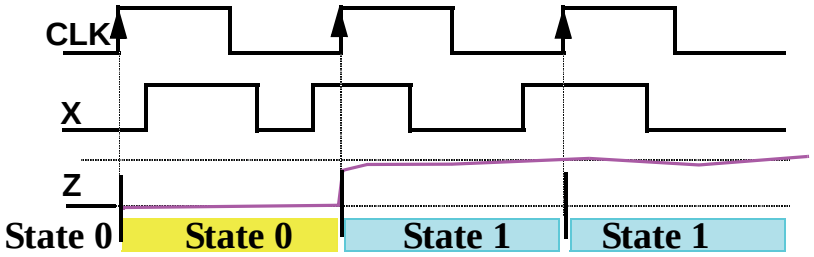
Reset(L)-Set(L) Latch
Set L =1 when RS=10
Reset L = 0 when R=0
Store output when SR=11
Set dominant

F

Synchronous State Graph



Sketch the circuit.

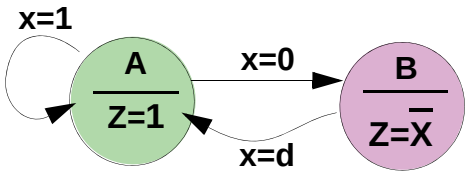


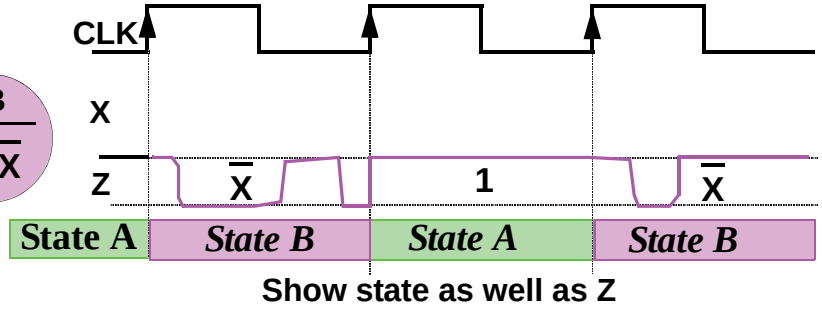
State=Z

State only changes on the clock edge

Synchronous State Graph

G





State A
State B

Show state as well as Z

Races and Cycles

Race

1 input change

changes 2 state variable (or more)

(00) → 11 (10) → 01

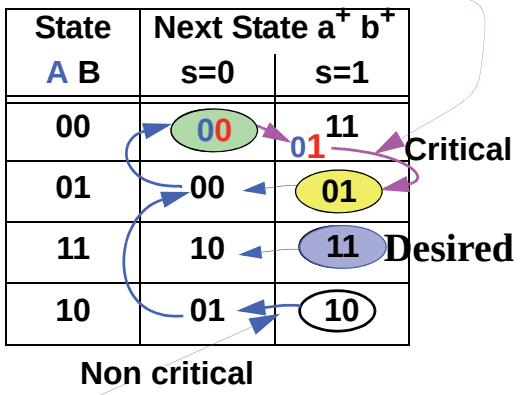
Stable State

Next state = State

Always circle stable states (10)

When looking for races

Start at stable states



Non critical

Critical Race

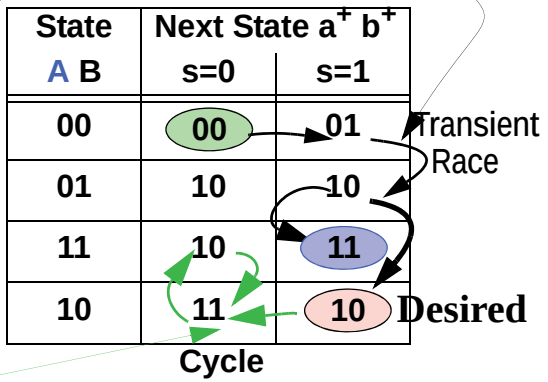
Race cause final state to be wrong

Non Critical Race

Race ends in correct final state

Race between transient states

Race cause final state to be wrong



Cycle

Who cares Race

Race on never used transition

Cycle

Oscillation

Races

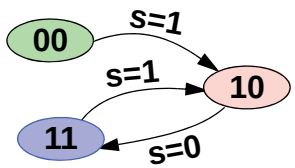
Who cares Race

Race on never used transition

Must start looking from a stable state

Never fine this race

State Graph



State A B	Next State a^+ b^+ s=0	s=1
00	00	10
01	11	00
11	11	10
10	11	10

never used transition

Non Critical Race

Race ends in correct final state

First change 00->11 or 00->01 or 00->10

Race ends in correct final state

State A B	Next State a^+ b^+ s=0	s=1
00	00	11
01	10	11
11	10	11
10	11	11

non critic

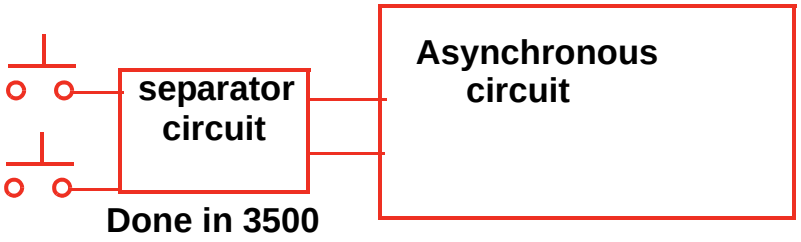
Desirec

Cycle

Races In Asynchronous State Tables

.Double Input Changes Are Not Allowed

If two inputs change at once,
build an input circuit
to separate them



Allowed input changes

starting from input 00

state	next state $g^+ h^+$			
GH	xy=00	xy=01	xy=11	xy=10
00	00			
01				
11				
10				

starting from input 01

state	next state $g^+ h^+$			
GH	xy=00	xy=01	xy=11	xy=10
00				
01		01		
11				
10				

starting from input 11

state	next state $g^+ h^+$			
GH	xy=00	xy=01	xy=11	xy=10
00				
01				
11				
10			10	

starting from input 10

state	next state $g^+ h^+$			
GH	xy=00	xy=01	xy=11	xy=10
00				
01				
11			11	
10				

When Checking For Races

1) Always start at a stable state
2) Check both sides
3) Remember wrap around

Find Races In Asynchronous State

Find Races In Asynchronous State Tables

Example.1-1 .Find Races and Cycles

a) State Tables to be checked

Present State FG	Next State F^+G^+ For inputs:			
	ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

Double input changes never happen.

Find Races In Asynchronous State

Find Races In Asynchronous State Tables

.Find Races and Cycles

a) State Tables to be checked

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

b) Circle stable states

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	(00)	(00)
FG=01	11	10	(01)	(01)
FG=11	(11)	(11)	(11)	01
FG=10	(10)	11	00	(10)

Double input changes never happen.

Find Races In Asynchronous State

Find Races In Asynchronous State Tables

Example.1-1 .Find Races

a) State Tables for poor design

Present State FG	Next State F^+G^+ For inputs:			
	ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

b) Circle stable states

Present State FG	Next State F^+G^+ For inputs:			
	ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	(00)	(00)
FG=01	11	10	(01)	(01)
FG=11	(11)	(11)	(11)	01
FG=10	(10)	11	00	(10)

c) Start at a stable state
Check next state on both sides
Do F^+ and G^+ both change?

Present State FG	Next State F^+G^+ For inputs:			
	ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01 ← ok	(00)	ok → (00)
FG=01	11	10	(01)	(01)
FG=11	(11)	(11)	(11)	01
FG=10	(10)	11	00	(10)

Find Races In Asynchronous State

Find Races In Asynchronous State Tables

Example.1-1 .Find Races

a) State Tables for poor design

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

b) Circle stable states

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	**00**	**00**
FG=01	11	10	**01**	**01**
FG=11	**11**	**11**	**11**	01
FG=10	**10**	11	00	**10**

c) Starting at stable states
Change <u>one</u> input, either a or b
Check for F^+ and G^+ both changing.

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	**00**
FG=01	11	10	**01**	**01**
FG=11	**11**	**11**	**11**	01
FG=10	**10**	11	00	**10**

Race 1

ok

d) Check Race 1

Present State FG	Next State F^+G^+ For inputs: ab = 00	01	11	ab = 10
FG=00	11	01	00	**00**
FG=01	11	10	**01**	**01**
FG=11	**11**	**11**	**11**	01
FG=10	**10**	11	00	**10**

Race 1

Race 1
Starts at
FG=00, ab=10

a changes 1->0

May end in
FG=11, ab=00
FG=10, ab=00

Find Races In Asynchronous State

Find Races In Asynchronous State Tables

Example.1-1 .Find Races

a) State Tables for poor design

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

b) Circle stable states

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

c) Starting at stable states
Change <u>one</u> input, either a or b
Check for F^+ and G^+ both changing.

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

Race 2

ok

Fixed thanks to Priti

e) Check Race 2

Present State FG	Next State F^+G^+ For inputs: 00	ab =01	ab = 11	10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

Race 2
Starts at FG=01, ab=11
a changes 1->0
Will end in FG=11, ab=01

Noncritical race

Races that don't start in a stable state are "who cares" unless part of a larger path.

Find Races In Asynchronous State

Find Races In Asynchronous State Tables (Cont)

Example.1-1 .Find Races Continued

b) Circle stable states

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

f) If the first step is not a race.
Check for both F^+ and G^+ changing as transitions go down the column.

Present State FG	Next State F^+G^+ For inputs: ab = 00	ab =01	ab = 11	ab = 10
FG=00	11 Race 3	01 ok	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

g) Check Race 3

Present State FG	Next State F^+G^+ For inputs: 00	ab =01	ab = 11	10
FG=00	11	01	00	00
FG=01	11	10	01	01
FG=11	11	11	11	01
FG=10	10	11	00	10

Race 3

Starts at FG=00, ab=11
a changes 1->0

Race invoves
FG=01, ab=01
FG=10, ab=01
FG=11, ab=01

Will end in FG=11, ab=01

Noncritical race between transient states

Find Races In Asynchronous State

Finding Races In Asynchronous State Tables

Example.1-2 Find Races

a) **Find races**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	00	01	10	00
01	01	11	00	10
11	00	00	01	11
10	00	10	11	10

b) **Circle stable states**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	(00)	01	10	(00)
01	(01)	11	00	10
11	00	00	01	(11)
10	00	(10)	11	(10)

c) **Starting from stable states look for double state changes**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	(00)	01	1 10	(00)
01	(01)	11	2 00	10
11	00	00	01	(11)
10	00	(10)	11	(10)

No race:
i) Not allowed input change
ii) Doesn't start in stable state

d) **Check 1st race**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	(00)	01	1 10	(00)
01	(01)	11	00	10
11	00	00	01	(11)
10	00	(10)	11	(10)

Critical.
Can end in three different different states

Start state 01 (ab=00)
End: one of
state 00 (ab=10)
state 11 (ab=10)
state 10 (ab=10)

e) **Check 2nd race**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	(00)	01	10	(00)
01	(01)	11	2 00	10
11	00	00	01	(11)
10	00	(10)	11	(10)

Critical.
Can end in two different states

Start state 11 (ab=10)
End: one of
state 00 (ab=00)
state 01 (ab=00)

Remember double input changes are not allowed.
Races that don't start in a stable state are "who cares" unless part of a larger path.

Find Races In Asynchronous State

Finding Cycles

Example.1-3 Find a Cycle

One which oscillates until an input changes..

a) **Find a true cycle**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	00	01	10	00
01	01	10	00	10
11	00	00	01	11
10	00	10	11	10

b) **Circle stable states**

XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	(00)	01	10	(00)
01	(01)	10	00	10
11	00	00	01	(11)
10	00	(10)	11	(10)

Starting at stable state check both sides

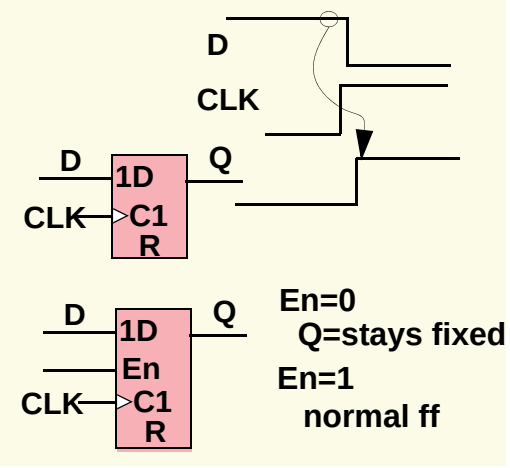
c) **From stable state**

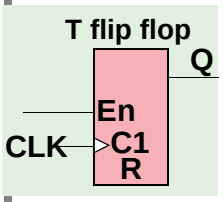
XY	$x^+ y^+$ for inputs ab: 00	01	11	10
00	(00)	01	10	(00)
01	(01)	10	00	10
11	00	00	01	(11)
10	00	(10)	11	(10)

Cycle
No stable state
Oscillates until input changes.

Synchronous Machines

- **D-Flip flops**
  - Sample D input just before the clock..
  - Transfer D to output just after the clock.
  - Q never changes except at clock edges



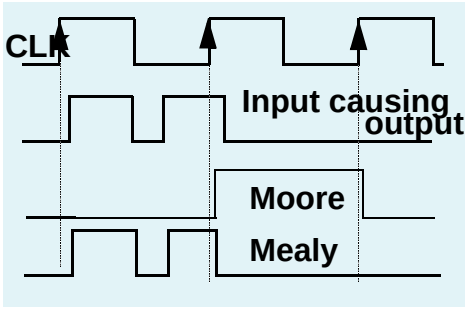


- **T-flip flops**
  - Toggle Q after every clock edge if enabled (provided it has an enable)

- **Put state tables in K-map order**

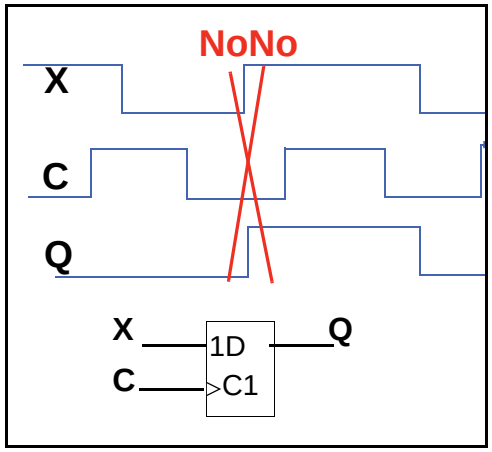
0 00
0 01
0 11
0 10
1 00
1 01
1 11
1 10

- **To tell Moore from Mealy in word problems**
  1. **Moore Outputs**
     - Outputs will appear after the next active clock edge.
     - Outputs will last a full clock cycle.
     - Glitch free outputs.
  2. **Mealy Outputs**
     - Outputs will appear after the input changes

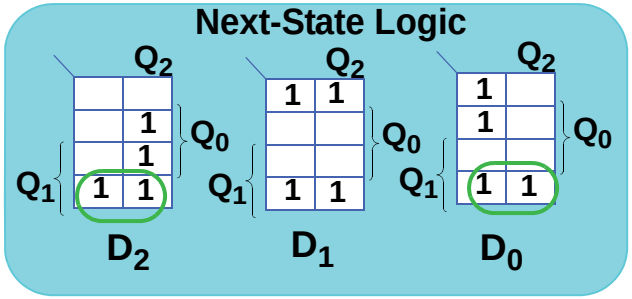


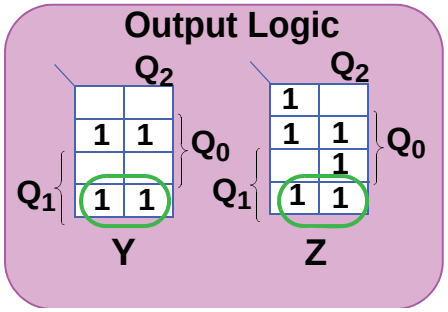
Synchronous Machines

- Change states only at an active clock edge.
  Watch on timing diagrams.



- FSMs are multiple output machines.
  Watch for shared gates when you loop maps.



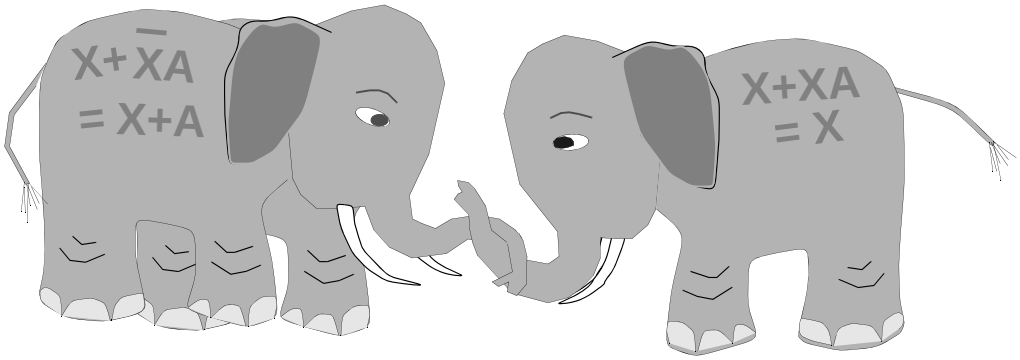


Key Points

- Read the question! All of it!
- Always check for $Ax + A = A$ Use this to simplify anywhere.
- Always reduce $A + \overline{A}x = A + x$ Use anywhere except for hazards

Too many people say ~~$x + 1 = x$~~

Good Luck on Your Exams



Have a Good Summer