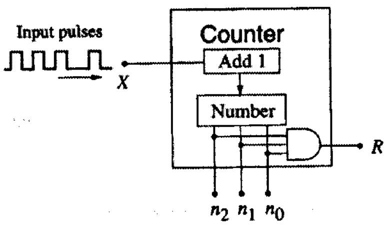
CpE2210 Introduction to Computer Engineering

Dr. Minsu Choi

CH 7&8: Sequential Logic Networks

Sequential Logic Networks?

- A sequential logic network is a digital system where the output is determined by both the present input and the result of a previous event.
- Ex) 3-bit counter



X= Input that causes transition
R= output

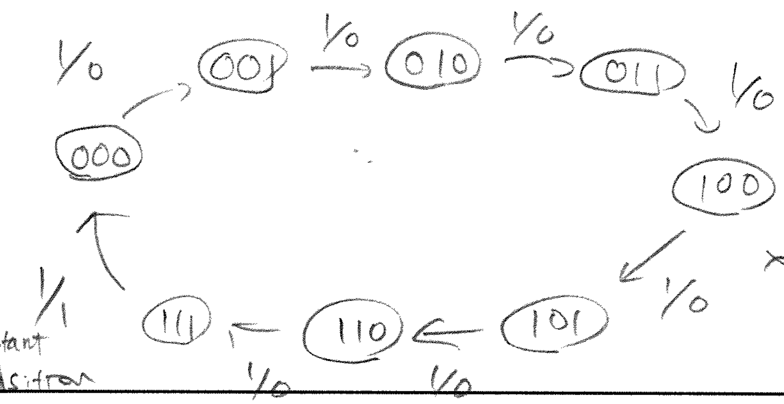
X/R to next state
State
from the previous state

=> symbolic representation of a state

Formal Description of the System: State Diagram

- We define each possible value of the word as a distinct state of the machine.

ex) 3-bit counter



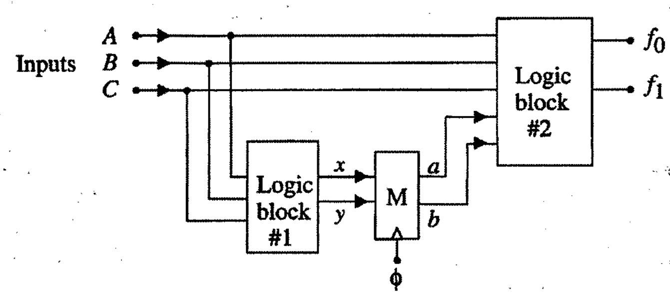
← state diagram.

Sequential circuit design ≈ Finite state machine (FSM)

Sequential Network Requirements

- A combinational logic network to perform Boolean operations.
- A memory element that stores the result of an earlier event.
- If we require a clock signal to synchronize the events, it is called "synchronous logic network", also.
- Clocking signal convention: T = 1 is assumed. → We now assume that 1 is the unit clock period
- (t-1) to t called previous cycle, t to (t+1) called present cycle and (t+1) to (t+2) called next clock cycle.

Ex) General Sequential Network #1

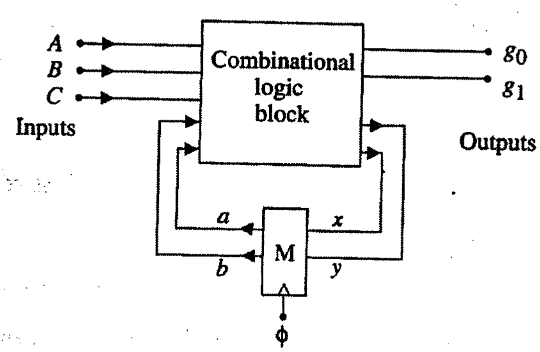


- Two combinational logic blocks #1 and #2 and a state element (a clocked memory) M.

$$a(t) = x(t-1)$$
$$b(t) = y(t-1)$$

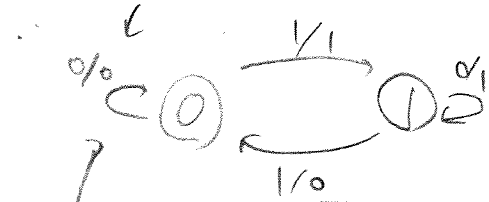
$$\Rightarrow f_0 = f_0(\overbrace{A, B, C}^{\text{current input}}, \overbrace{a, b}^{\text{previous state}})$$
$$f_1 = f_1(A, B, C, a, b)$$

Ex) General Sequential Network #2



- 1 combinational logic block & 1 memory element and feedback wires.

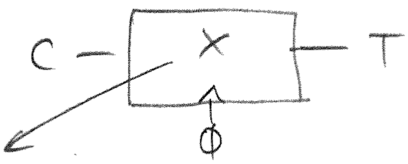
C/T notation



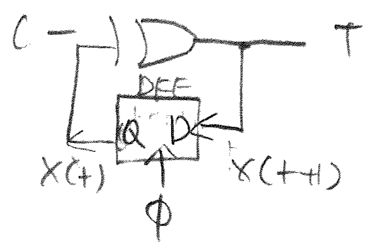
⇒ C acts as a control bit.
(if C=0 then no state change
(if C=1 " state change.

Single State Variable Networks

- **Ex1)**



$T(t) = C(t) \oplus X(t)$



⇒ Circuit diagram.

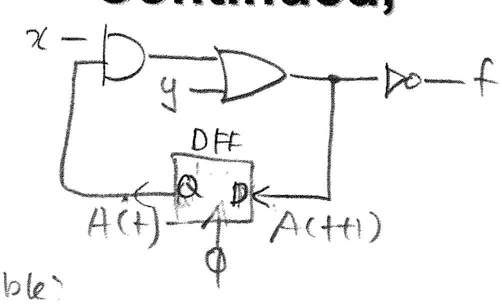
present value		output	Next value
C(t)	X(t)	T(t)	X(t+1)
0	0	0	0
0	1	1	1
1	0	1	1
1	1	0	0

$\boxed{X(t+1) = D_X(t) = T(t)}$
⇐ T(t) is used as X(t+1)

⇒ state table.

Continued,

- **Ex2)**

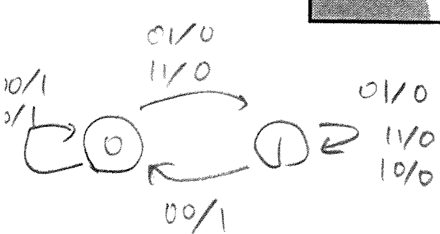


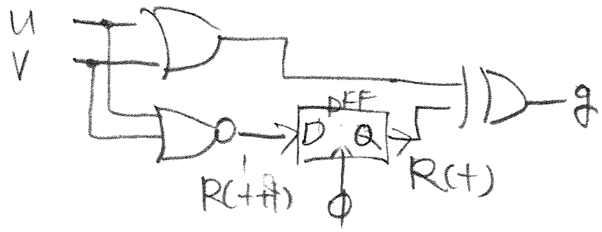
⇒ $f = \overline{x \cdot A + y}$
$D_A = \overline{f}$
$= x \cdot A + y$.
$A(t+1) = D_A(t)$

state table)

present state A(t)	present inputs x(t)	y(t)	present output f(t)	Next state A(t+1)
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	0	1

state diagram.





$\Rightarrow g = (u+v) \oplus R$

$D_R(t) = \overline{u(t) + v(t)}$

$= R(t+1)$

state table

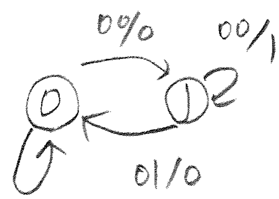
Continued,

- Ex3)

present values				next state
R(t)	u(t)	v(t)	g(t)	R(t+1)
0	0	0	0	1
0	0	1	1	0
0	1	0	1	0
0	1	1	1	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	0
1	1	1	0	0

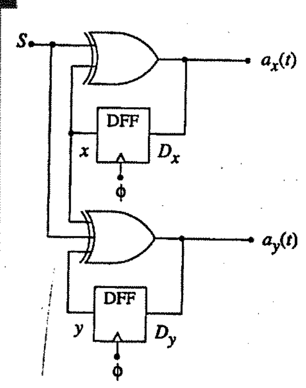
state diagram.

uv/g



Multi-State Variable Networks

- If we have n state variables, then 2^n states are needed when we draw a state diagram.
- Ex)



$a_x(t) = s(t) \oplus x(t)$

$a_y(t) = s(t) \oplus x(t) \oplus y(t)$

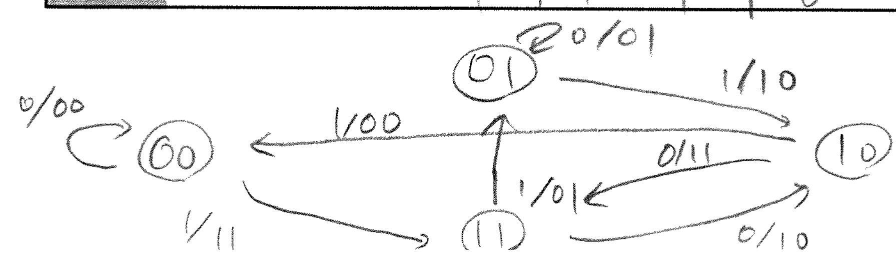
&

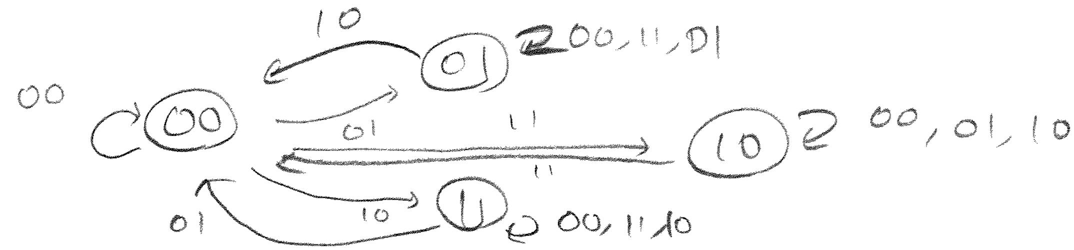
$x(t+1) = D_x(t) = a_x(t)$

$y(t+1) = D_y(t) = a_y(t)$

present values					next states	
s(t)	x(t)	y(t)	$a_x(t)$	$a_y(t)$	x(t+1)	y(t+1)
0	0	0	0	0	0	0
0	0	1	0	1	0	1
0	1	0	1	1	1	1
0	1	1	1	0	1	0
1	0	0	1	1	1	1
1	0	1	1	0	1	0
1	1	0	0	0	0	0
1	1	1	0	1	0	1

s/$a_x a_y$





State Diagram to State Table

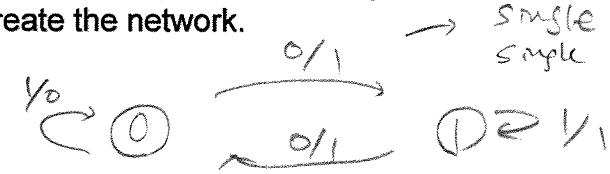
⇒ For each state, consider out going transitions.

Inputs X Y	present state A B	Next state A B
0 0	0 0	0 0
0 1	0 0	0 1
1 0	0 0	1 1
1 1	0 0	1 0
	⋮	

You need to consider 4 cases since you have 2 input bits.

Sequential Network Design

- We start with a listing of the desired transition characteristics (state table, etc) and use the information to create the network. → single input var $x(t)$, single output var $f(t)$, single state variable $A(t)$
- Ex)



present values A(t)	x(t)	f(t)	Next A(t+1)	FF in DA(t)
0	0	1	1	1
0	1	0	0	0
1	0	1	0	0
1	1	1	1	1

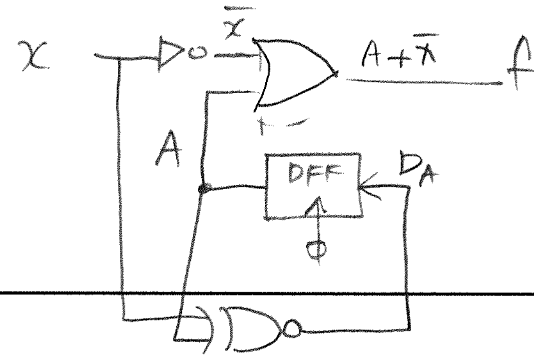
FF in → current FF input

Continued,

K-map to find f & D_A

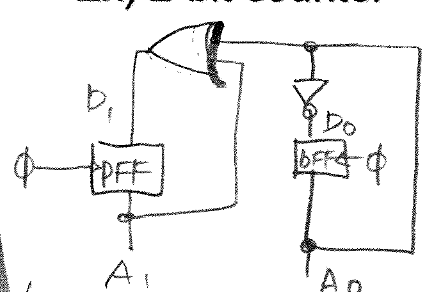
$D_A = \bar{A}\bar{x} + Ax = \overline{A \oplus x}$

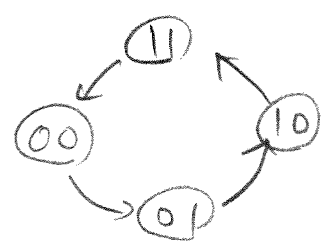
$f = A + \bar{x}$



Binary Counters

- Ex) 2-bit counter





$D_0 = \bar{A_0}$, $D_1 = A_1 \oplus A_0$

$A_1(t+1) = D_1(t)$, $A_0(t+1) = D_0(t)$

present		Next		Inputs to FFs	
A_1	A_0	A_1	A_0	D_1	D_0
0	0	0	1	0	1
0	1	1	0	1	0
1	0	1	1	1	1
1	1	0	0	0	0

(000) → (001) → … (111) → (000)

∈ 8 States.

⇒ we need three FFs

Three inputs : $D_2\ D_1\ D_0$

" outputs : $A_2\ A_1\ A_0$

Continued,

⇒ $A_2(t+1) = D_2(t)$
$A_1(t+1) = D_1(t)$
$A_0(t+1) = D_0(t)$

- Ex) 3-bit counter

present $A_2 A_1 A_0$	next $A_2 A_1 A_0$	inputs $D_2 D_1 D_0$
0 0 0	0 0 1	
0 0 1	0 1 0	
0 1 0	0 1 1	→ same
0 1 1	1 0 0	
1 0 0	1 0 1	
1 0 1	1 1 0	
1 1 0	1 1 1	
1 1 1	0 0 0	

K-maps

D_2 map

$A_2 \backslash A_1A_0$	00	01	11	10
0			1	
1	1	1		1

D_1 map

$A_2 \backslash A_1A_0$	00	01	11	10
0		1		1
1		1		1

D_0 map.

$A_2 \backslash A_1A_0$	00	01	11	10
0	1			1
1	1			1

Continued,

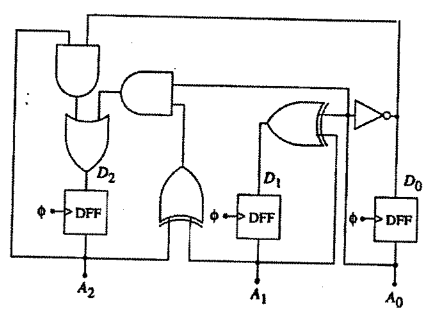
$D_2 = A_2\bar{A_0} + A_2\bar{A_1}A_0 + \bar{A_2}A_1A_0$

$\quad = A_2\bar{A_0} + A_0(A_2 \oplus A_1)$

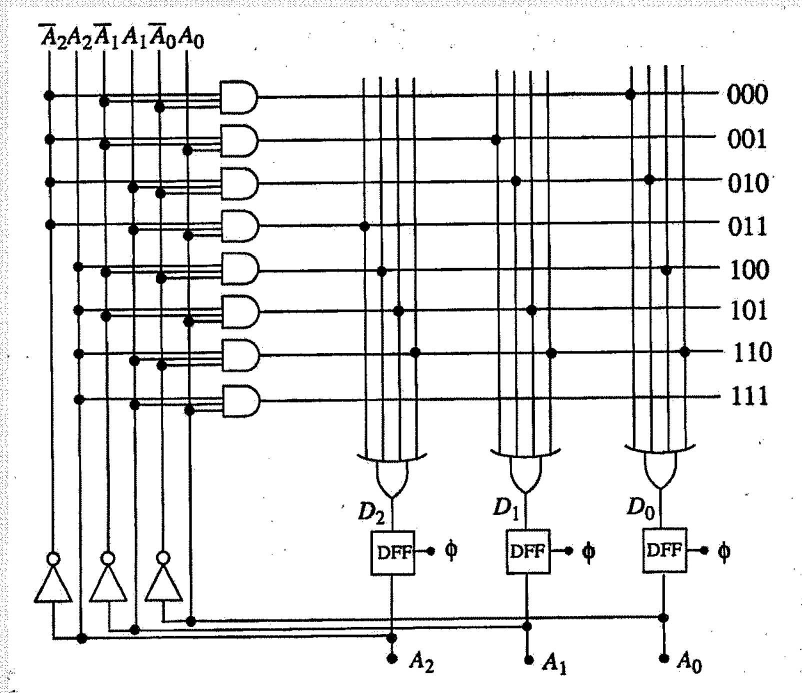
$D_1 = \bar{A_1}A_0 + A_1\bar{A_0} = A_1 \oplus A_0$.

$D_0 = \bar{A_0}$

Design



PLA Implementation of 3-Bit Counter (using SOP forms)



MISSOURI S&T

OLOGY

$D_2 = \Sigma m(3,4,5,6)$

$D_1 = \Sigma m(1,2,5,6)$

$D_0 = \Sigma m(0,2,4,6)$

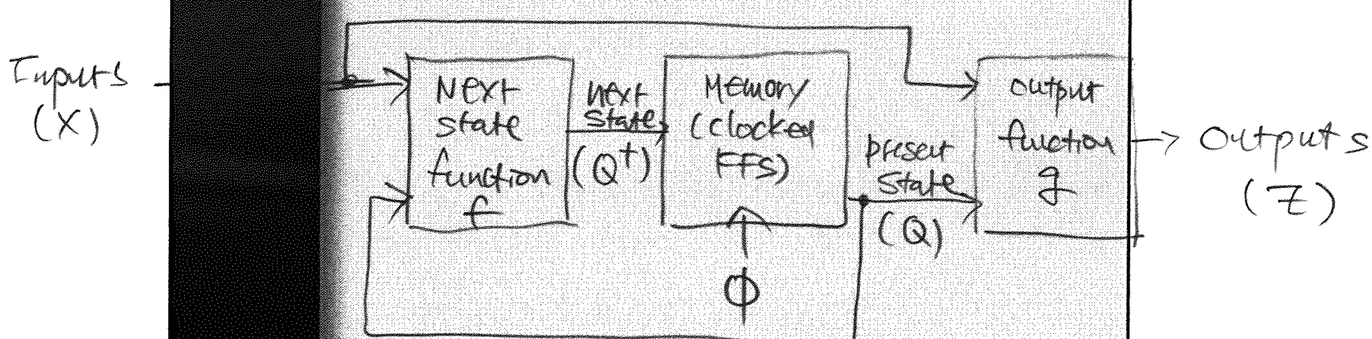
Two Different Types of State Machine

- Mealy model (machine)
  - Outputs are determined by both present inputs and present state.
  - State diagram: each transition edge labeled with the input triggering the transition and the output resulted from the transition.
- Moore model (machine)
  - Output values are determined solely by its current state.
  - State diagram: associates an output value with each state.

MISSOURI S&T

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

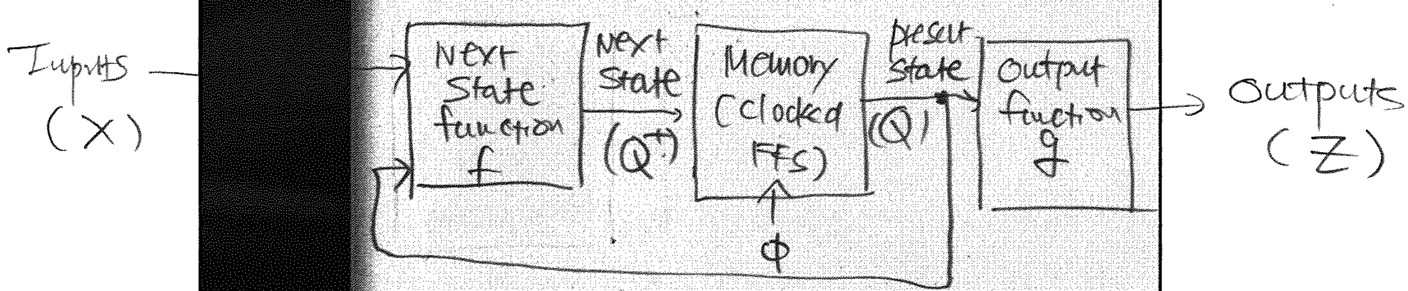
Mealy Machine Model



MISSOURI S&T

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Moore Machine Model



MISSOURI S&T

Ex) Non-resetting sequence detector

- Input: Random bit stream
- Output: = 1 if 0100 sequence is detected. Otherwise = 0.
- Non-resetting means state machine does not reset to the initial state when the desired sequence is detected (which means it keeps detecting the given sequence).

MISSOURI S&T

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Mealy Machine

See separate pages.

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Moore Machine

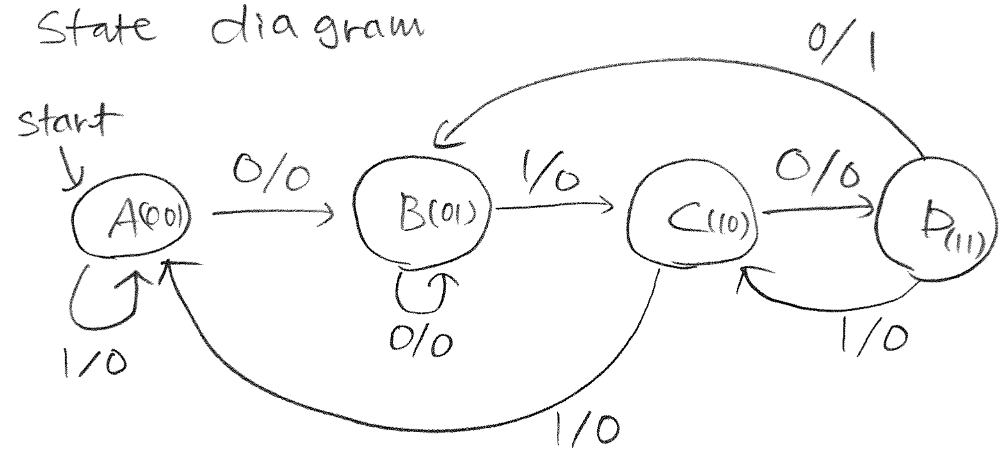
See separate pages

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Sequence detector (non-resetting, 0100)

★ Mealy machine

① State diagram



② Input, output, states encoding

Input: X
output: f
States

	D_1 D_0
A	00
B	01
C	10
D	11

③ State table.

Input $X(t)$	present state (t) D_1	D_0	next state $(t+1)$ D_1	D_0	output $f(t)$
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	1	1	0
0	1	1	0	1	1
1	0	0	0	0	0
1	0	1	1	0	0
1	1	0	0	0	0
1	1	1	1	0	0

④ K-maps for $D_1(t+1)$, $D_0(t+1)$, $f(t)$

$D_1(t+1)$

X \ $D_1 D_0$	00	01	11	10
0	0	0	0	1
1	0	1	1	0

$$D_1(t+1) = X \cdot D_0(t) + \bar{X} \cdot D_1(t) \cdot \overline{D_0(t)}$$

$D_0(t+1)$

X \ $D_1 D_0$	00	01	11	10
0	1	1	1	1
1	0	0	0	0

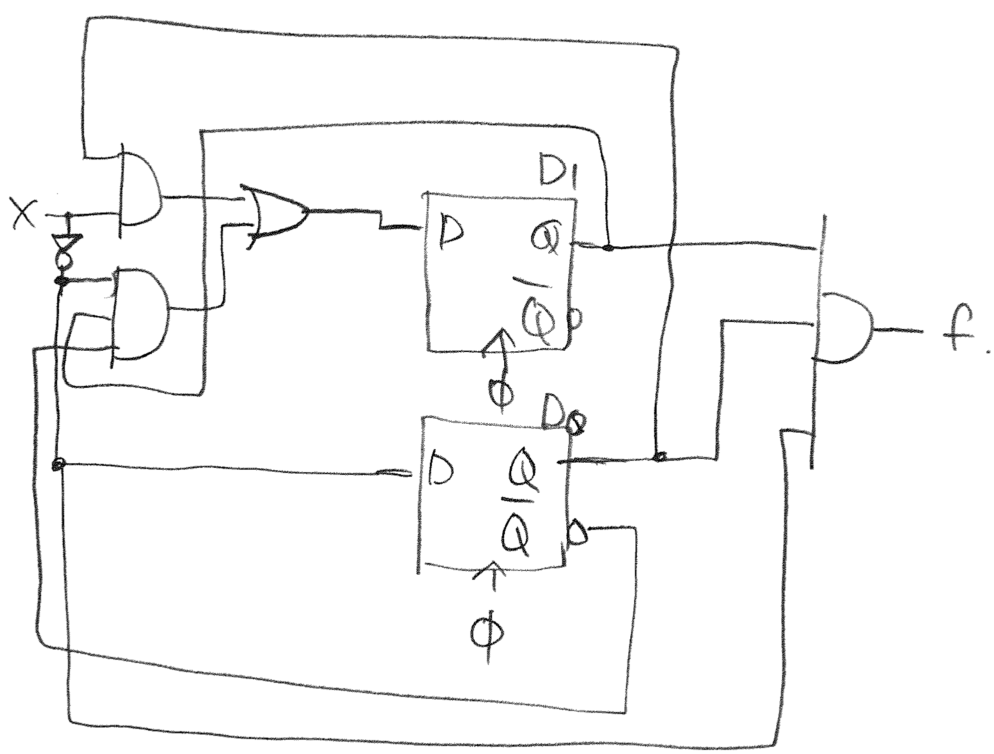
$$D_0(t+1) = \bar{X}$$

f

X \ $D_1 D_0$	00	01	11	10
0	0	0	1	0
1	0	0	0	0

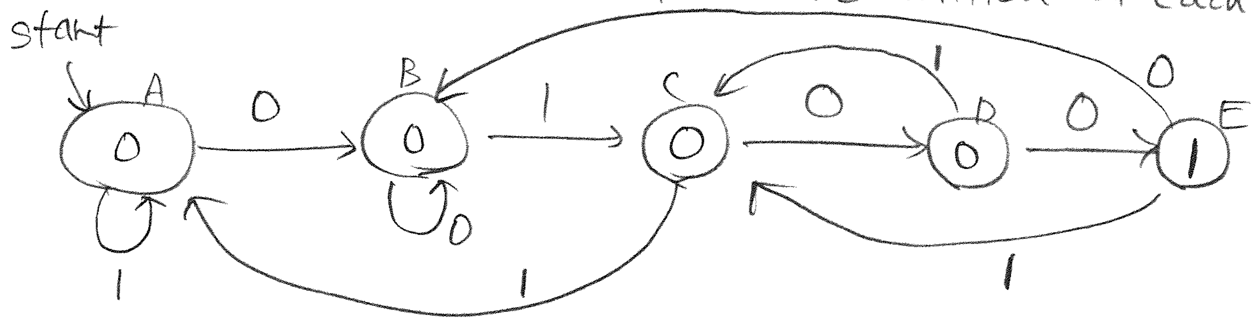
$$f = \bar{X} \cdot D_1(t) \cdot D_0(t)$$

⑤ Logic diagram (using DFF)



☆ Moore machine

① State diagram (note one more state needed & output value written at each state)



② Input, output / states encoding

Input: x, output: f, States: A 000, B 001, C 010, D 011, E 100

③ State table

Input	present State	/ output	next State			
x	D_2 D_1 D_0	f	D_2	D_1	D_0	
0	0 0 0	0	0	0	1	(A→B)
0	0 0 1	0	0	0	1	(B→B)
0	0 1 0	0	0	1	1	(C→D)
0	0 1 1	0	1	0	0	(D→E)
0	1 0 0	1	0	0	1	(E→B)
0	1 0 1	x	x	x	x	Invalid States so don't cares.
0	1 1 0	x	x	x	x	
0	1 1 1	x	x	x	x	
1	0 0 0	0	0	0	0	(A→A)
1	0 0 1	0	0	1	0	(B→C)
1	0 1 0	0	0	0	0	(C→A)
1	0 1 1	0	0	1	0	(D→C)
1	1 0 0	1	0	1	0	(E→C)
1	1 0 1	x	x	x	x	
1	1 1 0	x	x	x	x	
1	1 1 1	x	x	x	x	

4 K-maps.

D_2

$XD_2 \backslash D_1D_0$	00	01	11	10
00	0	0	1	0
01	0	X	X	X
11	0	X	X	X
10	0	0	0	0

$$D_2 = \bar{X} \cdot D_1 \cdot D_0$$

D_1

$XD_2 \backslash D_1D_0$	00	01	11	10
00	0	0	0	1
01	0	X	X	X
11	1	X	X	X
10	0	1	1	0

$$D_1 = X D_2 + X D_0 + \bar{X} D_1 \overline{D_0}$$

D_0

$XD_2 \backslash D_1D_0$	00	01	11	10
00	1	1	0	1
01	1	X	X	X
11	0	X	X	X
10	0	0	0	0

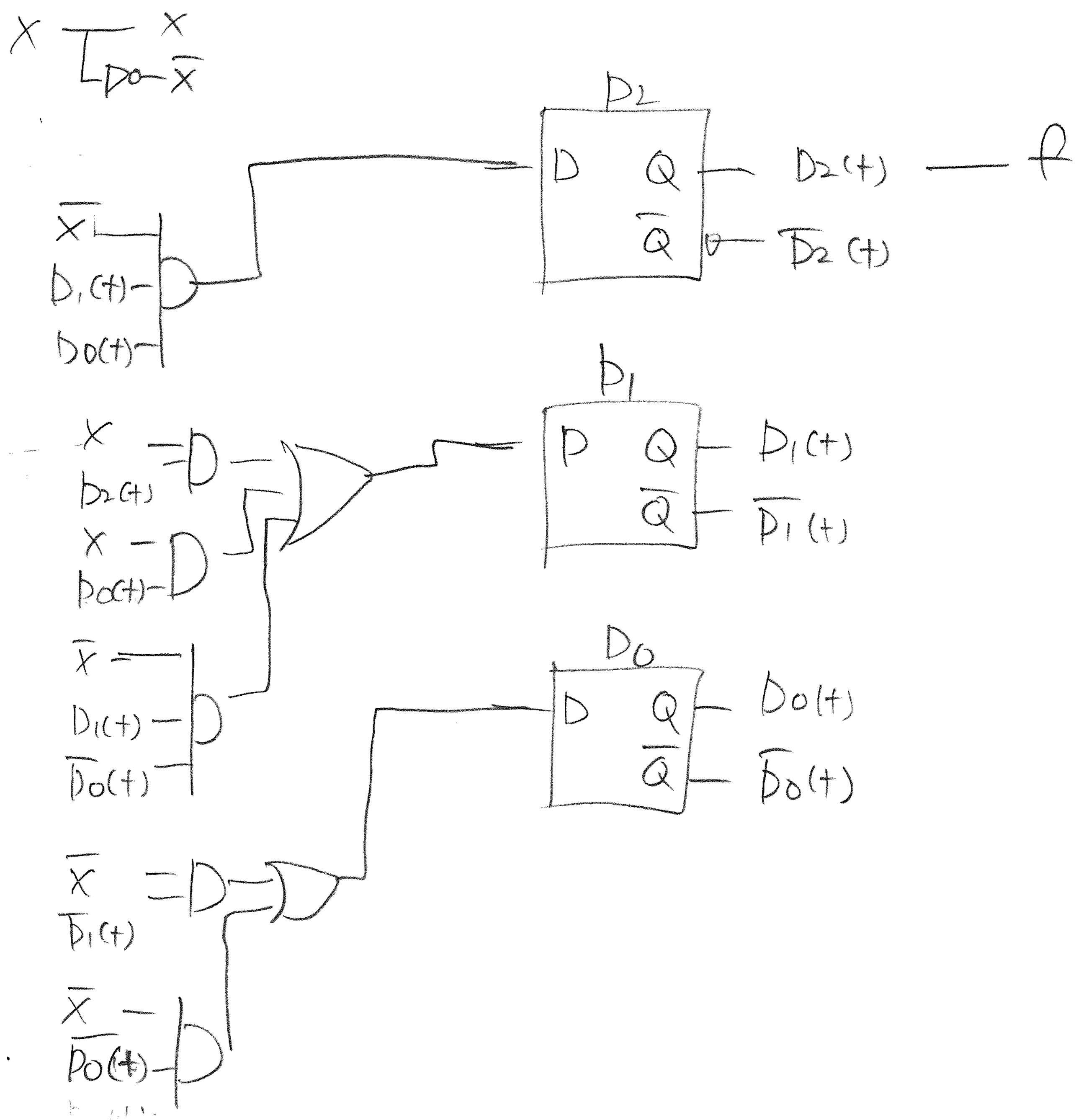
$$D_0 = \bar{X} \cdot \overline{D_1} + \bar{X} \overline{D_0}$$

f (note input X is not used to get f $\Rightarrow$ 3 variable K-maps)

$D_2 \backslash D_1D_0$	00	01	11	10
0	0	0	0	0
1	1	X	X	X

$$f = D_2$$

⑤ lofic diagram



⊛ FSM w/ FFs other than DFF.

ex)

present state $Q(t)$	Input $X(t)$	NextState $Q(t+1)$	Output $f(t)$	$J(t)$	$K(t)$
0	0	0	0	0	X
0	1	1	1	1	X
1	0	0	1	X	1
1	1	1	0	X	0

②

① JK implementation

J	K	Q(t+1)	
0	0	Q(t)	hold
0	1	0	reset
1	0	1	set
1	1	$\bar{Q}(t)$	Toggle.

⟹

Q(t)	Q(t+1)	J	K	
0	0	0	X	Hold ∅
0	1	1	X	set
1	0	X	1	Reset
1	1	X	0	Hold 1

③ Function for J?

0, 1, X, X → 0, 1, 1, 1 to make it OR2

$J(t) = Q(t) + X(t)$

K?

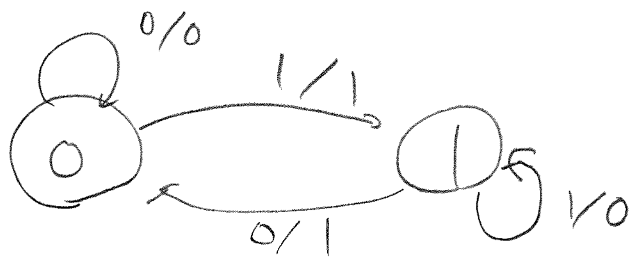
X, X, 1, 0 → 1, 1, 1, 0 to make it NAND2

$K(t) = \overline{Q(t) \cdot X(t)}$

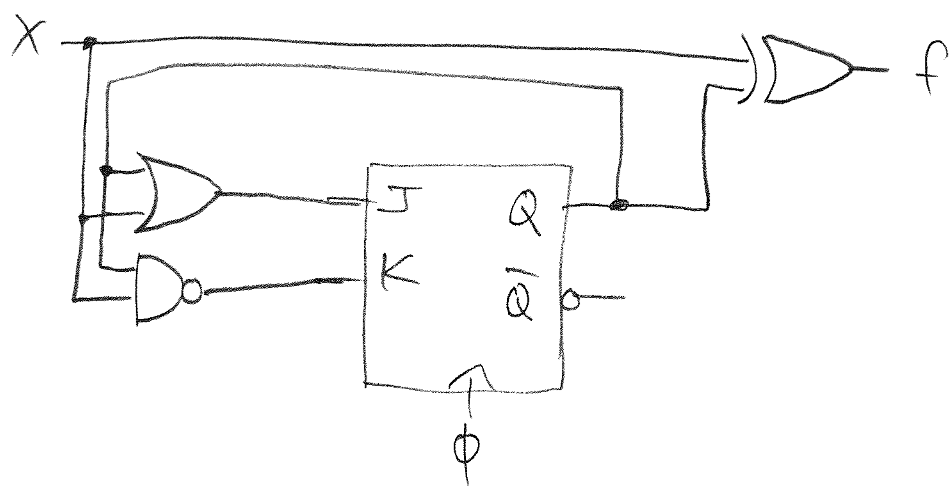
f? 0, 1, 1, 0 ⟹ XOR2

$f(t) = Q(t) \oplus X(t)$

④ state diagram



⑤ logic diagram.



ex)

T implementation

Q(t)	Q(t+1)	T	
0	0	0	hold
0	1	1	) Toggle.
1	0	1	
1	1	0	hold

①

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

② Function for T?

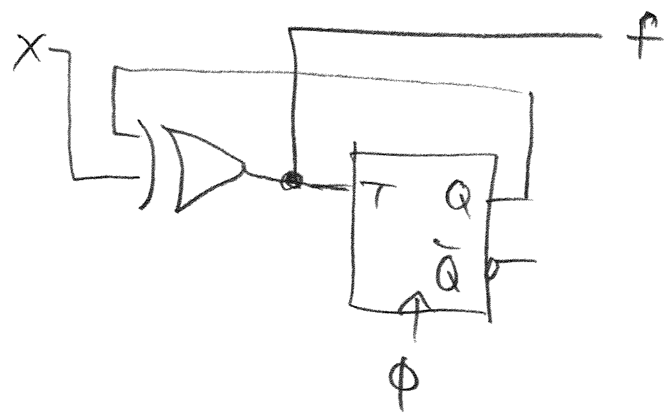
0
1 ⇒ XOR2
1
0

$T(t) = Q(t) \oplus x(t)$

Function for f?

$f(t) = Q(t) \oplus x(t)$

③ Logic diagram.



Program Completed

University of Missouri-Rolla

© 2003 Curators of University of Missouri

MISSOURI S&T

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY