Fault Tolerant Computing

CS 530

Lecture Notes 2

Digital Systems: overview

Yashwant K. Malaiya
Colorado State University

Digital Systems: overview

- **Building blocks**
- **Combinational design & analysis**
- **Untestable faults**
  - **some faults are untestable due to redundancy**
- **Storage elements**
- **Finite state machines: sequential circuits**
- **Testing a block**
  - **How can a combinational of sequancial block be exercised to test for potential faults?**

This is largely a review, but there are some new concepts in it.

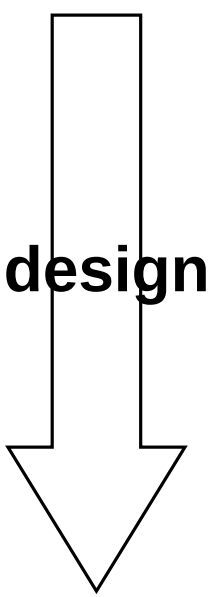
©Y.K. Malaiya

Digital Systems: Overview

- **Building block hierarchy**
  - **Subsystems (processors etc)**
  - **Combinational blocks, registers**
  - **Gates, storage elements**
  - **Switches (transistors)**
- **Blocks can be**
  - **Combinational (no memory)**
  - **Sequential (with states)**

Combinational Blocks

- **Boolean functions without storage capability**

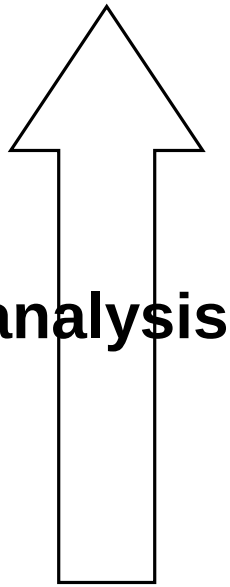


Truth table

Karnaugh maps (minimization)

Algebraic description

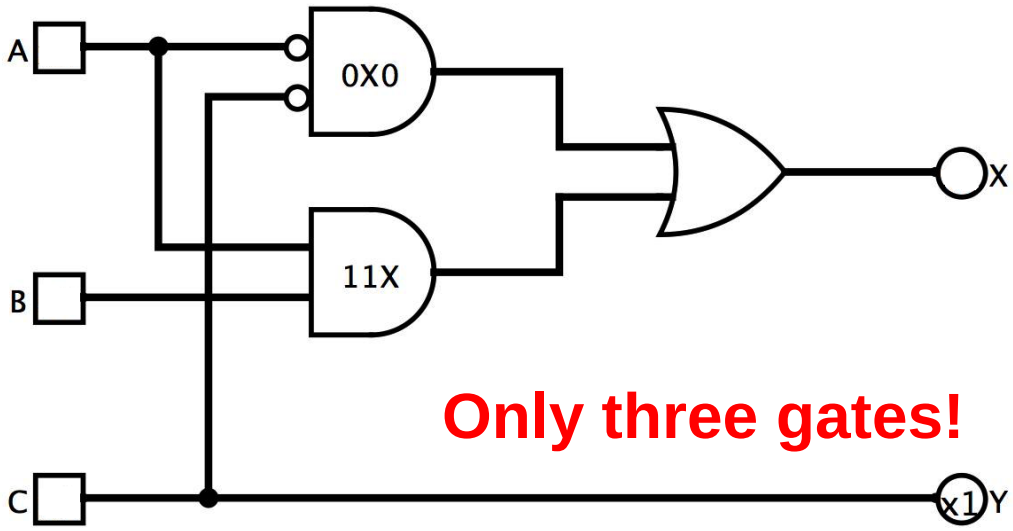
Circuit diagram (implementation)



If you have not seen Karnaugh maps before, you can find it in a textbook or on the web.

Circuit Minimization using Boolean Algebra

- **Boolean logic lets us reduce the circuit**
  - **X = A'B'C' + A'BC' + ABC' + ABC =**
    = A'C' + AB
  - **Y = A′B′C + A'BC + AB'C + ABC**
    = A'C+AC = C



Only three gates!

A	B	C	X	Y
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

Try with Logisim!

Karnaugh maps to minimize literals

Based on set-theory

- Visual representation of algebraic functions
- Allow algorithmic minimization of boolean functions in sum-of-products form
- "adjacent" terms can be combined.
  - Adjacent: differ in one variable, complemented in one, not complemented in the other.

Example:

- ABC+ABC' = AB(C+C')=AB
- Thus ABC and ABC' are two pieces of AB.

Combining Minterms

- For n-variables, there are 2^n minterms, corresponding to each row of truth table.
- Some of them can be combined into groups of 2, (or 4 or 8 ..) to simplify the function.

Karnaugh Maps

Visual representation of algebraic functions to make it easy to spot "adjacent" minterms"

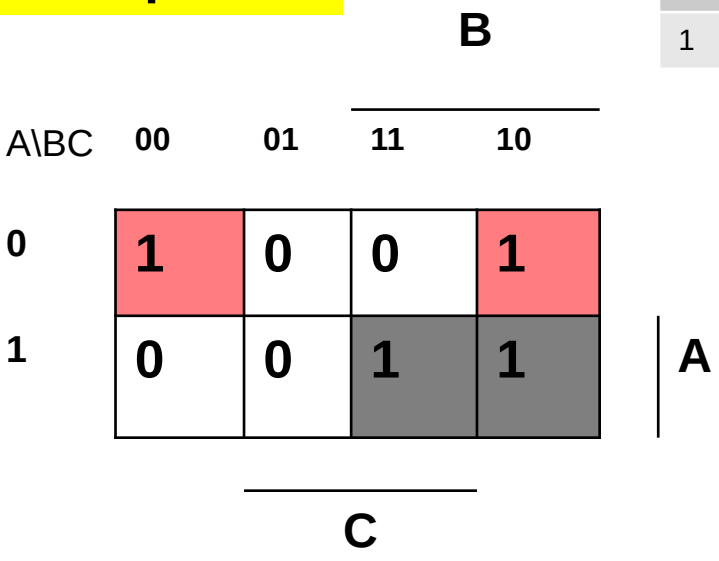
- Columns arranged so that *adjacent terms* are visually adjacent.
- Identify groups of 2, 4, 8 etc. terms that can be combined.
- All 1's must be covered.
- A 1 can be used more than once, if needed.
- Sometimes the solution is not unique
- Next: maps for X(A,B,C) and Y(A,B,C)

A	B	C	X	Y
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

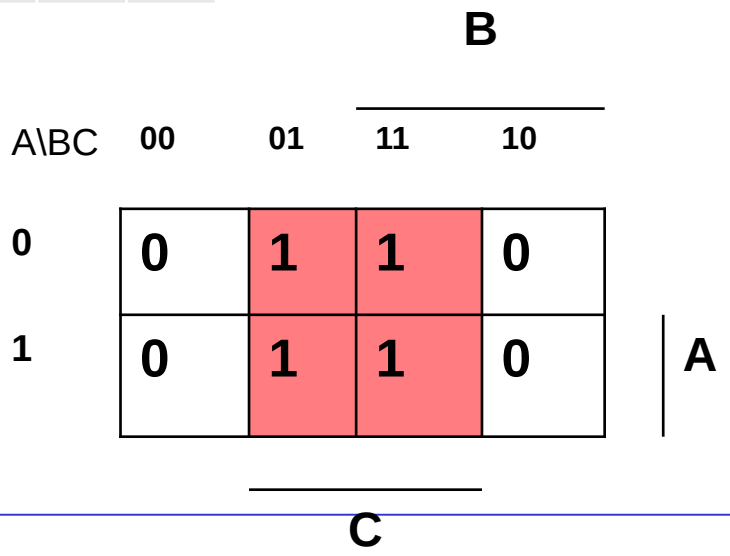
Karnaugh Maps: Visualization of algebra

A	B	C	X	Y
0	0	0	1	0
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1

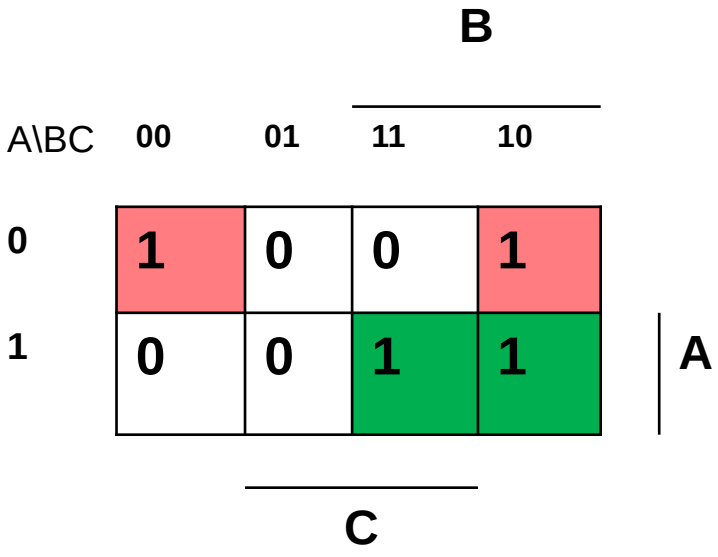
K Map for X

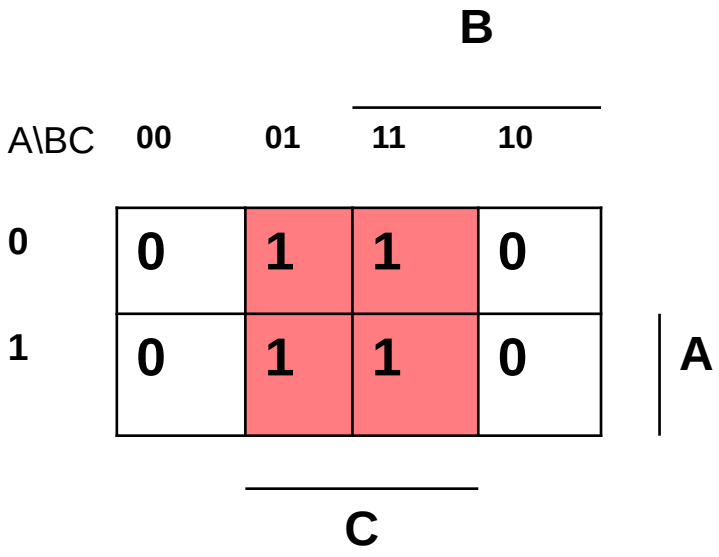


K Map for Y



Karnaugh Maps: Visualization of algebra





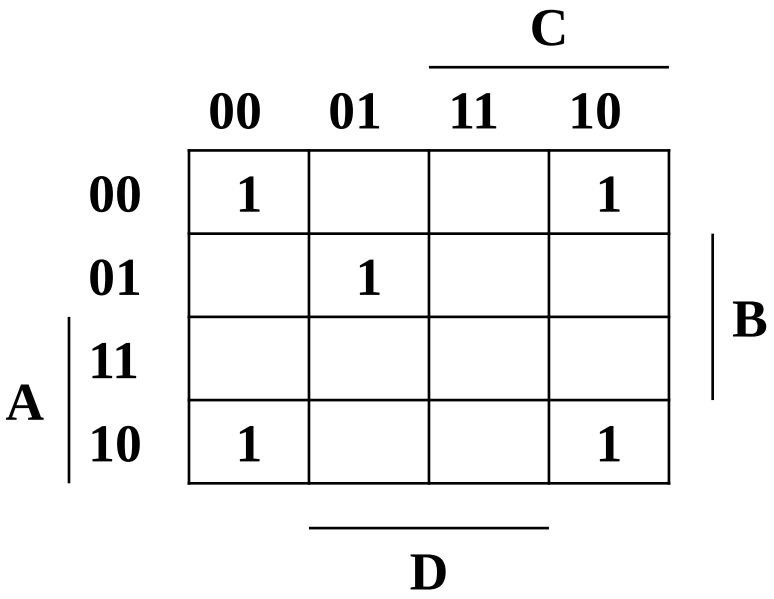
X: A'B'C'+A'BC' = A'C'; ABC+ABC' = AB

Y: A'B'C+A'BC+AB'C+ABC= A'C+AC = C

Thus minimized function is

X = A'C'+AB Y = C

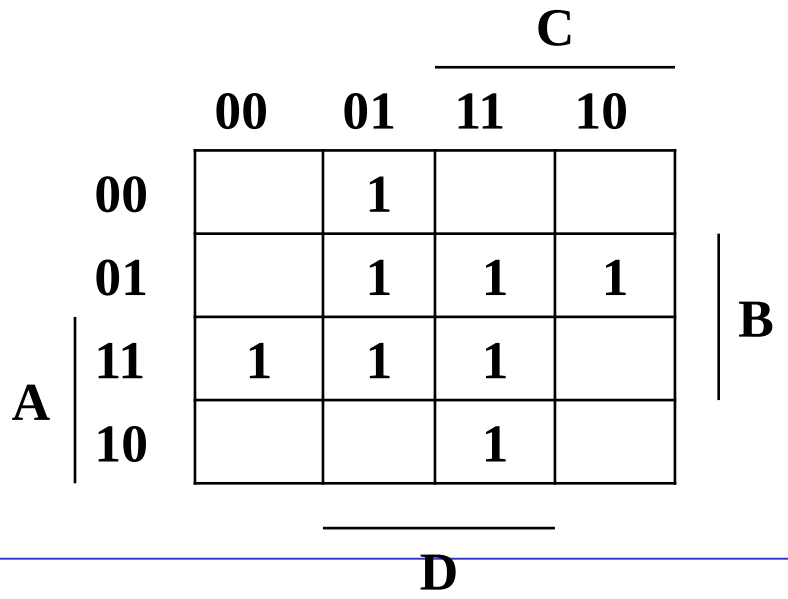
4-variable Kmaps / Design



F(A,B,C,D)=B’ D’ +_____

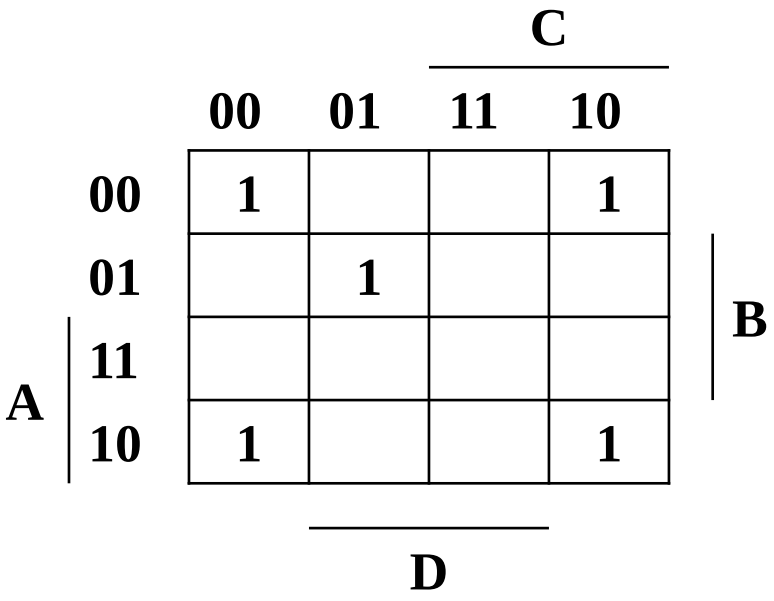
F(A,B,C,D)=ABC’ +A’ C’ D+

A’ BC+ACD+ ?



Try them with Logisim

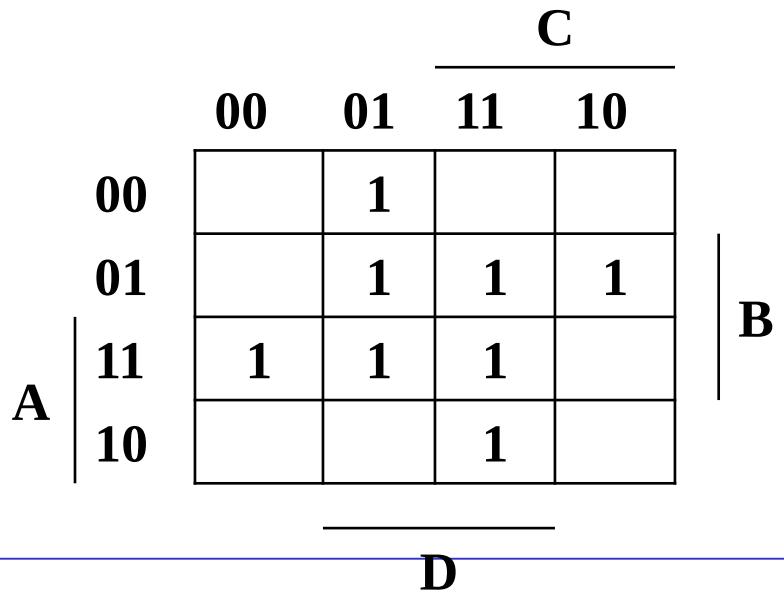
4-variable Kmaps / Design



F(A,B,C,D)=B’ D’ +A’BC’D

F(A,B,C,D)=ABC’ +A’ C’ D+

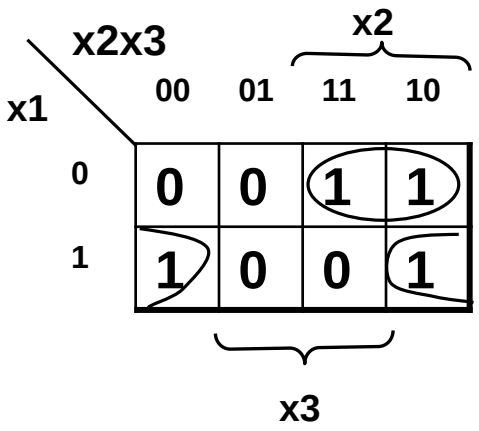
A’ BC+ACD ~~+ ?~~



Try them with Logisim

Combinational Example

x1	x2	x3	f
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

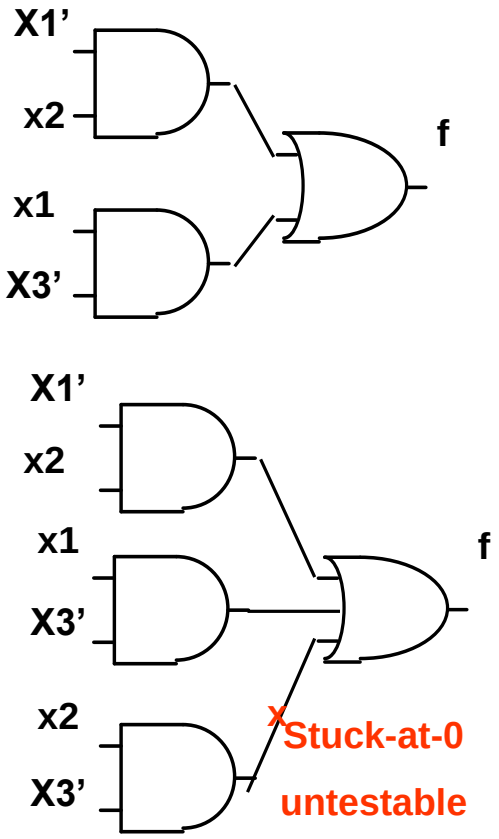


Implementation A

F=x1'x2 + x1x3'

Implementation B

F=x1'x2 + x1x3' + x2x3'



redundant

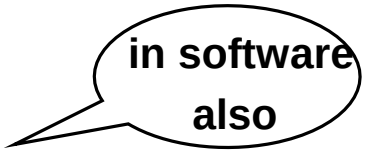
Note that the x2x3' term is redundant.

Here a prime indicates complement.

A stuck-at-0 fault makes the line always stay at 0 regardless of what it is supposed to be.

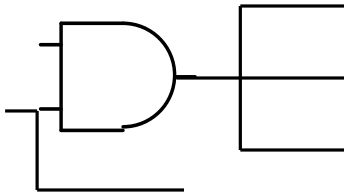
Special Considerations

- **In a redundant design, a fault may be undetectable.**
- **Undetectable faults are a major problem in testing.**

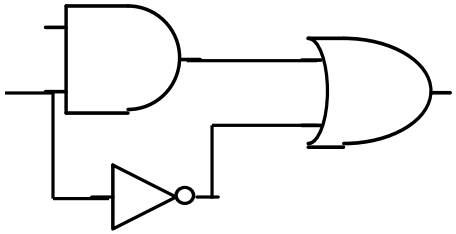


Fan-out: one output feeding multiple inputs.

Fan-out



Reconvergent fan-out

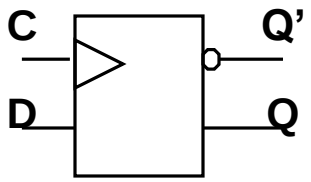


Reconvergent fan-out can lead to problen in testing, as we will see later.

Storage Elements

- **Synchronized by *clock* or *write* signals**
- **Construction: static (flip-flop), dynamic (capacitative, need refreshing)**
- **Functional: single input (D-type), double input**
- **Timing:** **rising or falling edge triggered**
  Pulse type etc
- **Significant instants:**
  - **when input is sampled**
  - **when outputs are influenced**

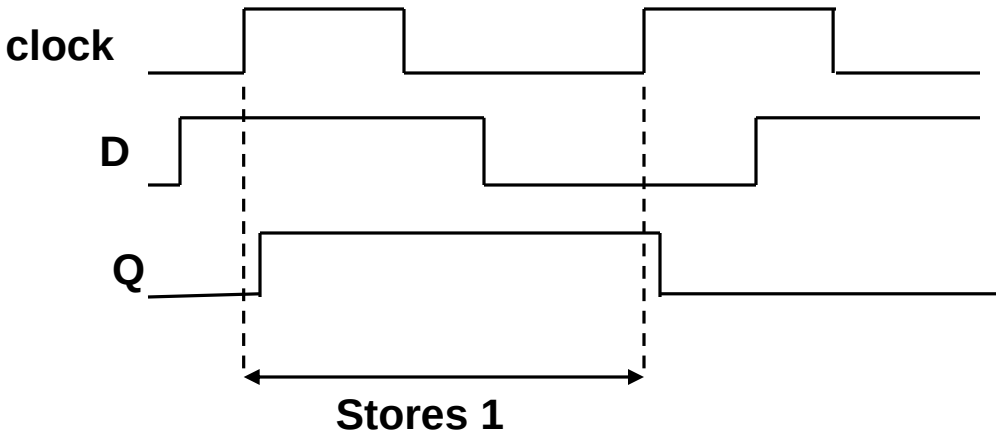
Ex: Rising Edge D flipflop



Samples on rising edge

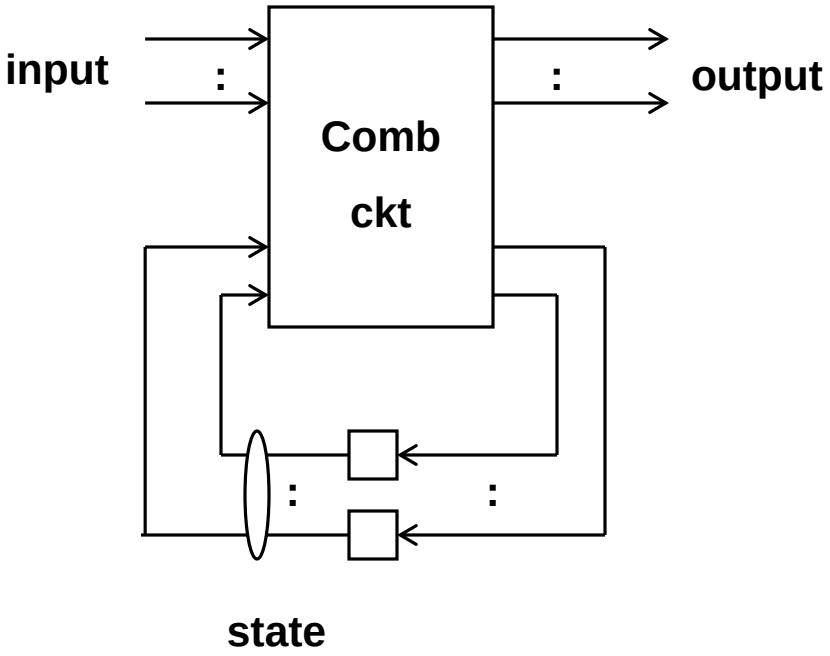
Output influenced soon after

Q_t	D	Q_{t+1}
0	0	0
0	1	1
1	0	0
1	1	1



During sampling input must remain stable

Sequential Circuits are Finite State Machines(FSMs)

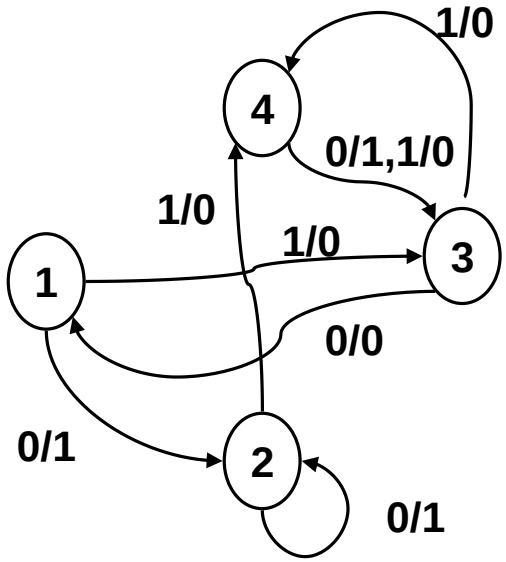


- Next state = N(state,input)
- Output = Z(state,input)

FSM Description

state \ Input x	0	1
1	2,1	3,0
2	2,1	4,0
3	1,0	4,0
4	3,1	3,0

Entries: N,Z



Format:
x/z

given

input	0	1	0
state	1	2	4
output	1	0	1

Based on FSM state table

(or state diagram)

Digital System with functional blocks

- **Ex: a microprocessor**

Blocks are

- **Paths**
  - **Signals, buses, nets**
- **Combinationals**
  - **Random logic, PLA, ROM**
- **Sequential**
  - **Flipflps, registers, memory arrays**
  - **Finite state machines (control part)**
- **System design objectives**
  - **functionality, reliability, peformence**

Exercising a block

- **Combinational blocks: a *test-vector* (or pattern) applied at a time for testing.**
  - **Exception: when testing for delay faults**
- **Sequential: a *test-sequence* (a sequence iof input vectors) at a time with proper initialization.**

References

- **See any Logic Design text-book for combinational and sequential circuits and Karnaugh Maps. For example: Computer System Architecture by Morris Mano.**
- **Karnaugh Maps:** http://web.archive.org/web/20080819230843/http://www.cs.usm.maine.edu/~welty/karnaugh.htm
- **Design for Testability in Digital Integrated circuits, Bob Strunz, Colin Flanagan, Tim Hall, Article, locally available at** http://www.cs.colostate.edu/~cs530/digital_testing.pdf