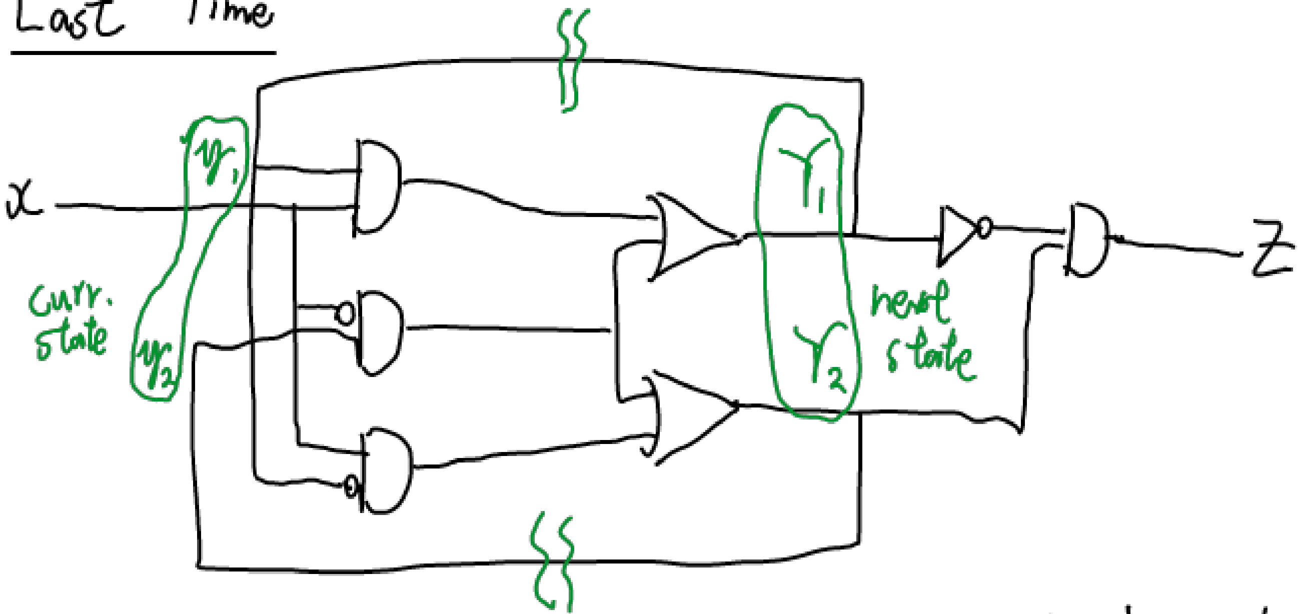
Last Time



Analysis – abstract away the actual implementation

steps: 1) Identify feedback paths, next state + curr. state.

2) Write eqns for next state + outputs in terms of inputs + curr. state.

3) Write down transition table & flow table; Identify stable states

has binary patterns for states

has symbolic names for states

like state table

Stability: Given an input, a circuit is stable when $Y_i = y_i \; \forall i$.

Eg.,

$$Y_1 = y_1 x + y_2 \bar{x}$$

$$Y_2 = \bar{x} y_2 + x \bar{y_1}$$

$$z = \bar{y_1} \, \bar{y_2}$$

Transition Table.

Curr. state $y_2 y_1$	next state, output $y_2 y_1, z$: $x=0$	$x=1$
0 0	00,0	10,0
0 1	00,0	01,0
1 0	11,1	10,1
1 1	11,0	01,0

stable situations

Flow Table

Let a=00, b=01, c=10, d=11

curr state	next state, output	
a	a,0	c,0
b	a,0	b,0
c	d,1	c,1
d	d,0	b,0

Design: make a circuit given a verbal description.

Defns: Fundamental mode assumption: You only change one input at a time, and only do this from a stable situation.

Primitive flow table: At most one stable state per row.

Steps:

1) Derive a state diagram + flow table. (generally, best to assume fundamental mode + try to get a primitive flow table)
2) Try to reduce the flowtable to get fewer states (impl. charts + merger diagrams).
3) Do state assignment (***avoid critical races)
4) Derive next state : output equations (***avoid hazards + glitches)
5) Draw circuit.

<u>Example:</u> Design a circuit with inputs D & G and output Q.
Q = D when G = 1, otherwise Q should not change.

Think of states as the measured values DG/Q

"hold 0"	10/0	"hold 1"	10/1
"hold 0"	00/0	"hold 1"	00/1
"follow 0"	01/0		~~01/1~~
"follow 1"	11/1		~~11/0~~