Sairam INSTITUTIONS

Sri SAI RAM ENGINEERING COLLEGE
INSTITUTE OF TECHNOLOGY
West Tambaram, Chennai - 44

YEAR	SEM
II	III

CS 8351

DIGITAL PRINCIPLES AND SYSTEM DESIGN

(Common to CSE & IT)

UNIT NO.4

4.1 ANALYSIS OF ASYNCHRONOUS SEQUENTIAL CIRCUITS

Version: 1.0

4 QUALITY EDUCATION

NBA NAAC (A+) nirf

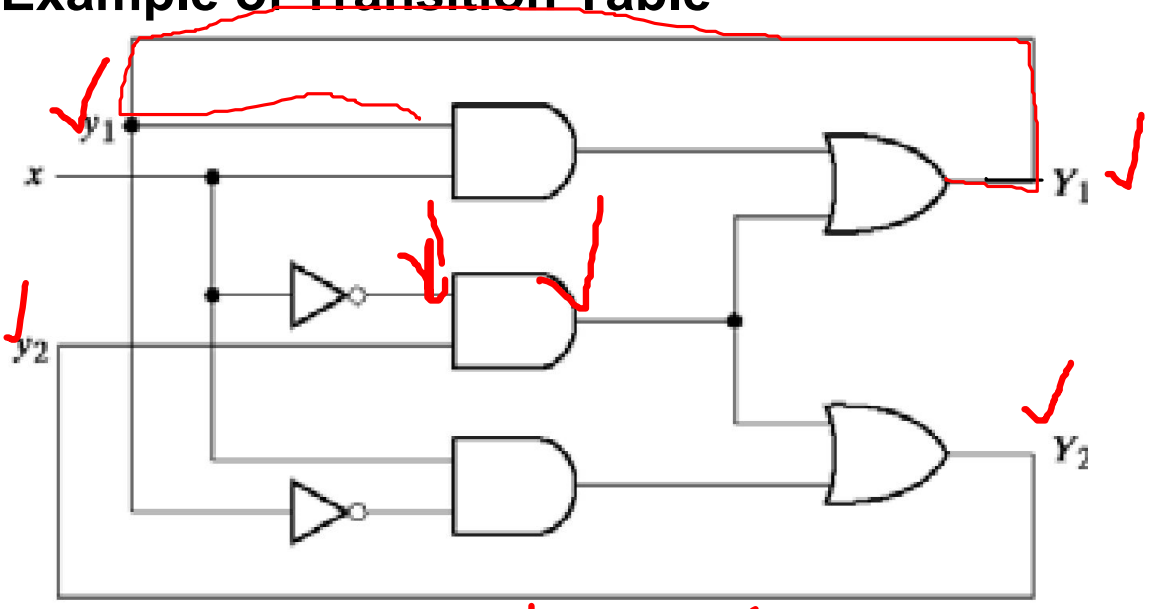
Transition Table

Transition table is useful to analyze an asynchronous circuit from the circuit diagram

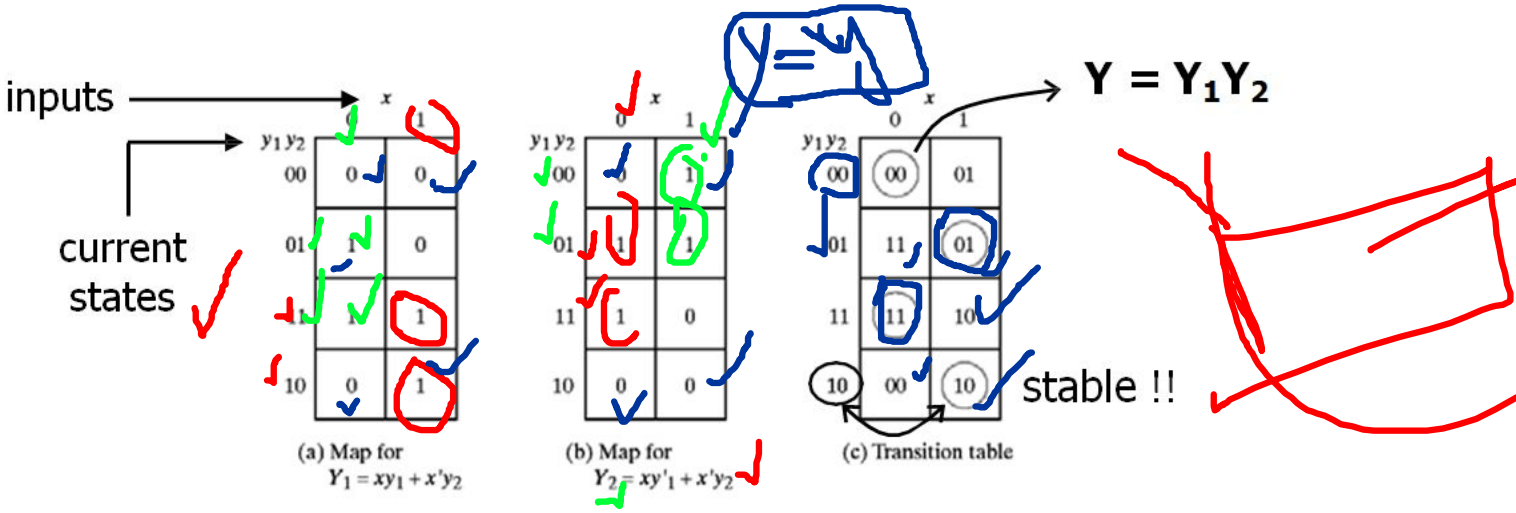
Procedure to obtain transition table:

1. Determine all feedback loops in the circuits
2. Mark the input (yi) and output (Yi) of each feedback loop .
3. Derive the Boolean functions of all Y's.
4. Plot each Y function in a map and combine all maps into one table.
5. Circle those values of Y in each square that are equal to the value of y in the same row.

An Example of Transition Table

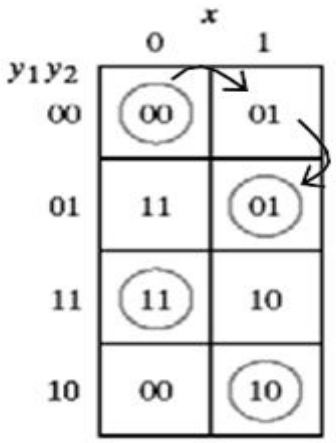


Y1 = xy1 + x'y2
Y2 = xy'1 + x'y2



State Table

- When input x changes from 0 to 1 while y=00:
  - Y changes to 01 € unstable
  - y becomes 01 after a short delay € stable at the second row
  - The next state is Y=01
- Each row must have ***at least one*** stable state
- Analyze each state in this way can obtain its state Table.



Present State		Next State			
		X=0		X=1	
0	0	0	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	1	1	1	1	0

y1y2x : total state

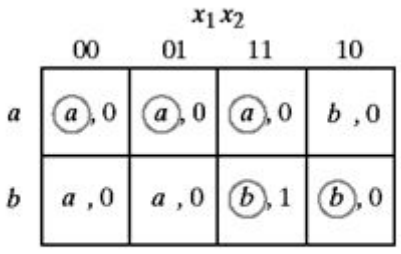
4 stable total states: 000,011,110,101

Flow Table

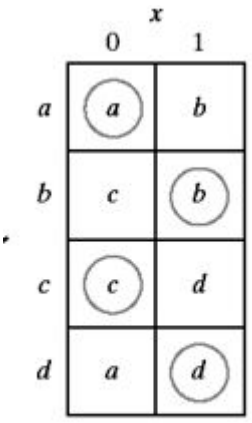
- Similar to a transition table except the states are represented by ***letter symbols***
- Can also include the output values
- Suitable to obtain the logic diagram from it

Primitive flow table:

only one stable state in each row



(b) Two states with two inputs and one output



(a) Four states with one input

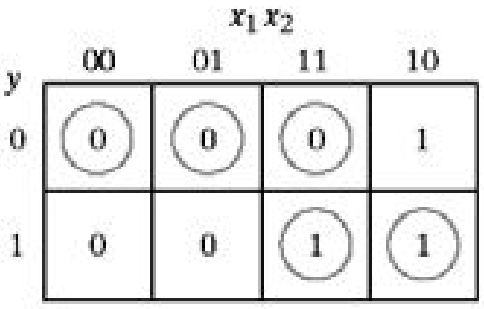
Flow Table to Circuits

Procedure to obtain circuits from flow table:

- Assign to each state a distinct binary value (convert to a transition table)
- Obtain circuits from the map

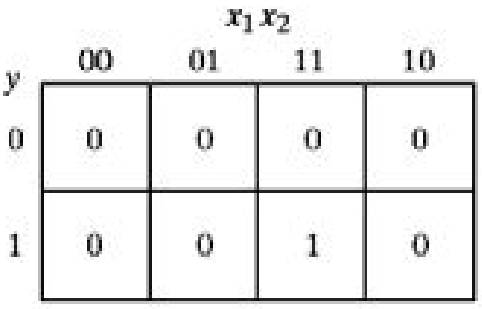
Two difficulties:

- The binary state assignment (to avoid race)
- The output assigned to the unstable states



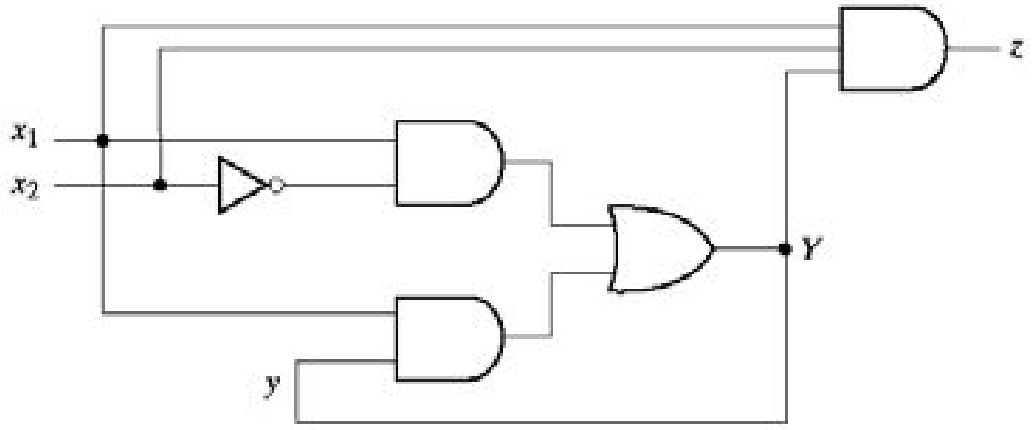
(a) Transition table

$Y = x_1x'_2 + x_1y$



(b) Map for output

$z = x_1x_2y$



(c) Logic diagram

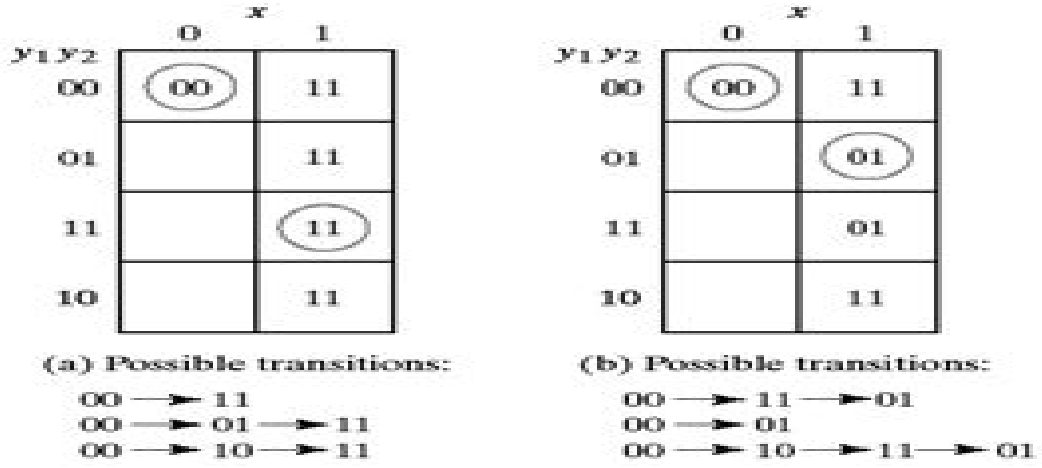
Race Conditions

Race condition:

- two or more binary state variables will change value when one input variable changes
- Cannot predict state sequence if unequal delay is encountered

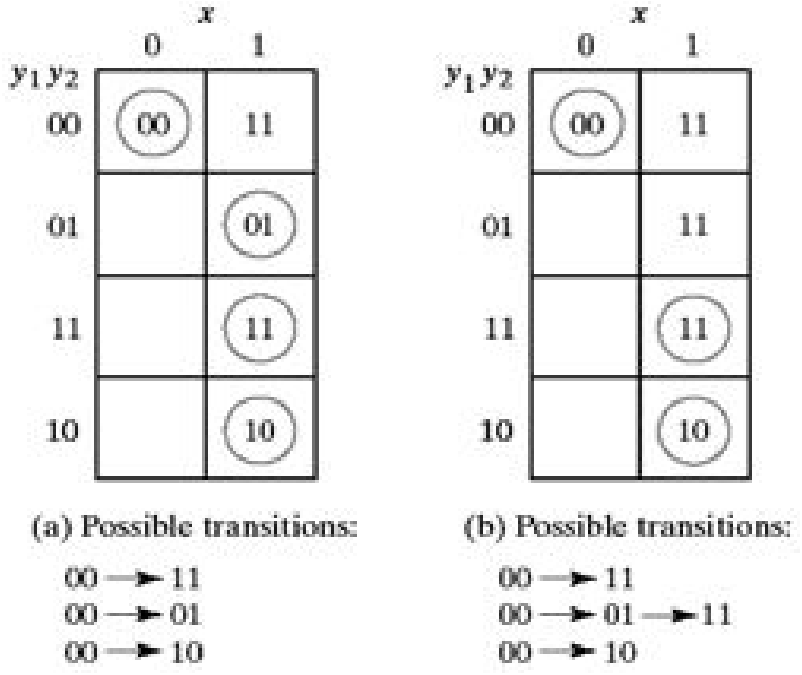
Non-critical race:

The final stable state does not depend on the change order of state variables.



Critical race:

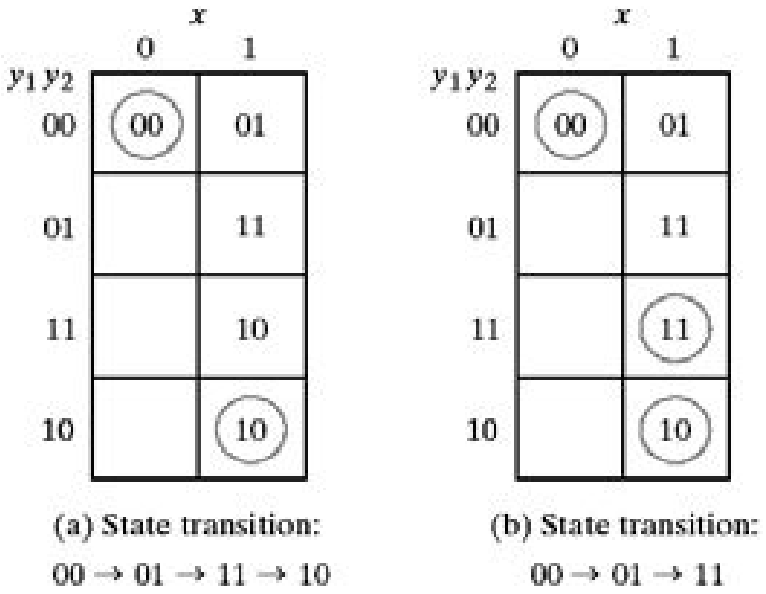
- The change order of state variables will result in different stable states.
- Should be avoided!!



Race-Free State Assignment

Race can be avoided by proper state assignment

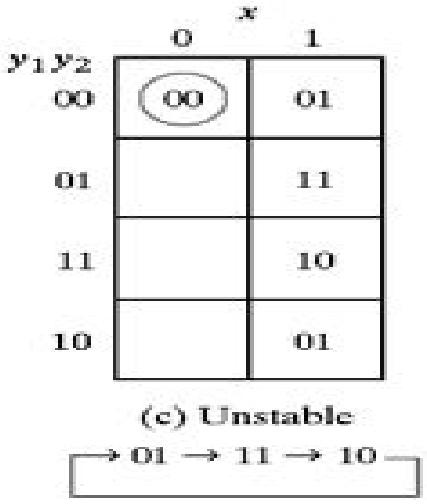
- Direct the circuit through intermediate unstable states with a unique state-variable change
- It is said to have a cycle



(a) State transition: 00 → 01 → 11 → 10

(b) State transition: 00 → 01 → 11

- Must ensure that a cycle will terminate with a stable state
- Otherwise, the circuit will keep going in unstable states

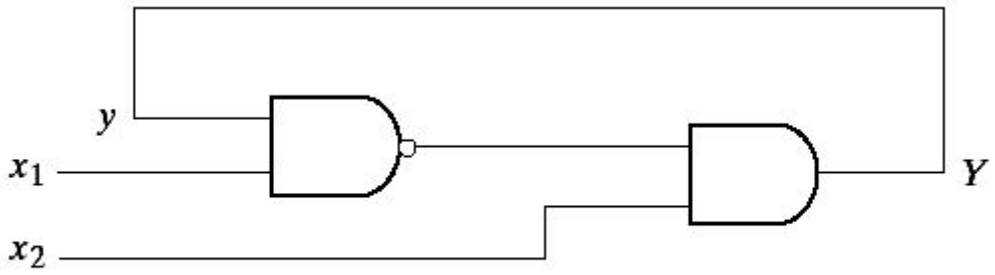


Stability Check

Asynchronous sequential circuits may oscillate between unstable states due to the feedback

- Must check for stability to ensure proper operations
  Can be easily checked from the transition table
- Any column has no stable states unstable
- Ex: when x1x2=11, Y and y are never the same

  Y = x′1x2 + x2y′



(a) Logic diagram

y \ $x_1 x_2$	00	01	11	10
0	0	1	1	0
1	0	1	0	0

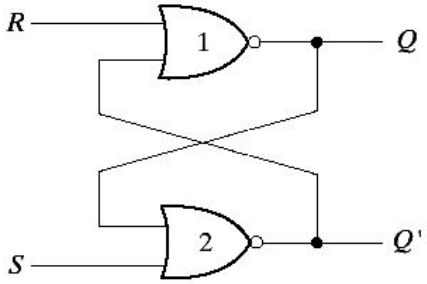
(b) Transition table

Latches in Asynchronous Circuits

- The traditional configuration of asynchronous circuits is using one or more feedback loops
  - No real delay elements
- It is more convenient to employ the SR latch as a memory element in asynchronous circuits

- Produce an orderly pattern in the logic diagram with the memory elements clearly visible
- SR latch is also an asynchronous circuit
  - Will be analyzed first using the method for asynchronous circuits.

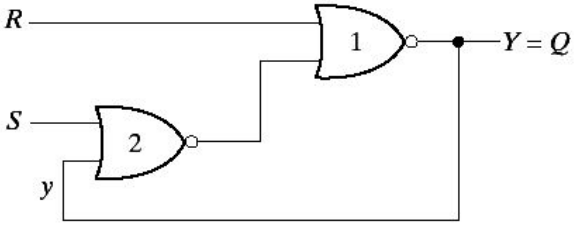
SR Latch with NOR Gates



(a) Crossed-coupled circuit

S	R	Q	Q'	
1	0	1	0	
0	0	1	0	(After $SR = 10$)
0	1	0	1	
0	0	0	1	(After $SR = 01$)
1	1	0	0	

(b) Truth table



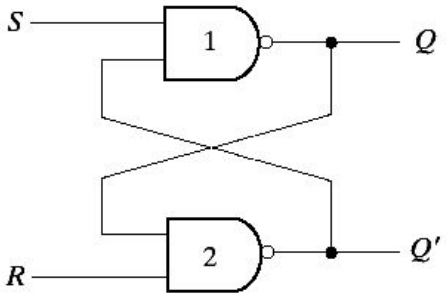
(c) Circuit showing feedback

y \ SR	00	01	11	10
0	(0)	(0)	(0)	1
1	(1)	0	0	(1)

$Y = SR' + R'y$
$Y = S + R'y$ when $SR = 0$

(d) Transition table

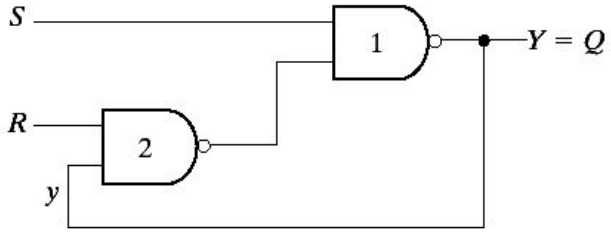
SR Latch with NAND Gates



(a) Crossed-coupled circuit

S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(After $SR = 10$)
0	1	1	0	
1	1	1	0	(After $SR = 01$)
0	0	1	1	

(b) Truth table



(c) Circuit showing feedback

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

SR

$Y = S' + Ry$ when $S'R' = 0$

(d) Transition table

Analysis Procedure

Procedure to analyze an asynchronous sequential circuits with SR latches:

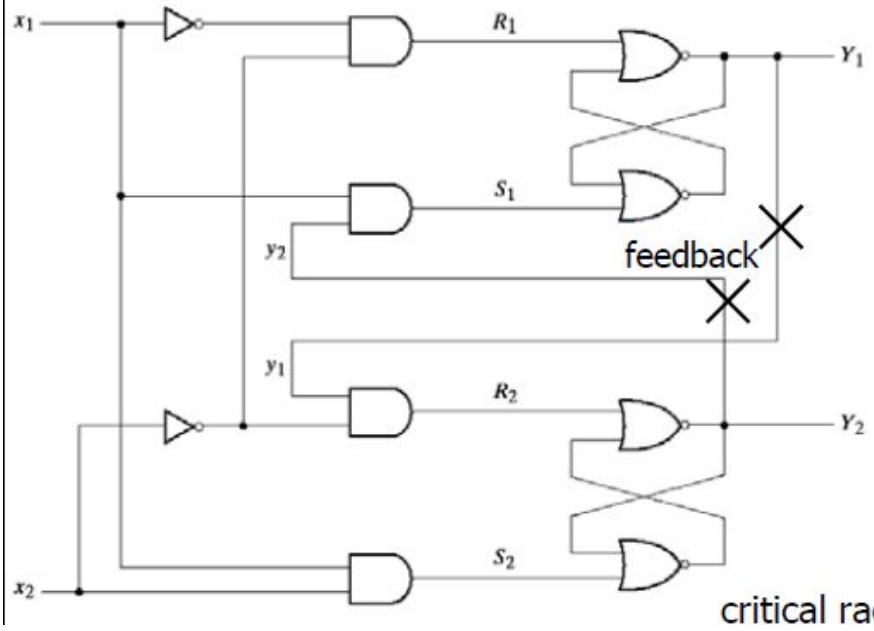
1. Label each latch output with Yi and its external feedback path (if any) with yi
2. Derive the Boolean functions for each Si and Ri
3. Check whether SR=0 (NOR latch) or S'R'=0 (NAND latch) is satisfied
4. Evaluate Y=S+R'y (NOR latch) or Y=S'+Ry (NAND latch)
5. Construct the transition table for Y=Y1Y2…Yk

6. Circle all stable states where Y=y

Analysis Example

$S_1=x_1y_2$ $R_1=x'_1x'_2 \Rightarrow S_1R_1 = x_1y_2x'_1x'_2 = 0$ (OK)
$S_2=x_1x_2$ $R_2=x'_2y_1 \Rightarrow S_2R_2 = x_1x_2x'_2y_1 = 0$ (OK)

$Y_1=S_1 + R'_1y_1$
$=x_1y_2 + (x_1+x_2)y_1$
$=x_1y_2+x_1y_1+x_2y_1$
$Y_2=S_2 + R'_2y_2$
$=x_1x_2 + (x_2+y'_1)y_2$
$=x_1x_2+x_2y_2+y'_1y_2$



y_1y_2 \ x_1x_2	00	01	11	10
00	(00)	(00)	01	(00)
01	(01)	(01)	11	11
11	00	(11)	(11)	10
10	00	(10)	11	(10)

critical race !!

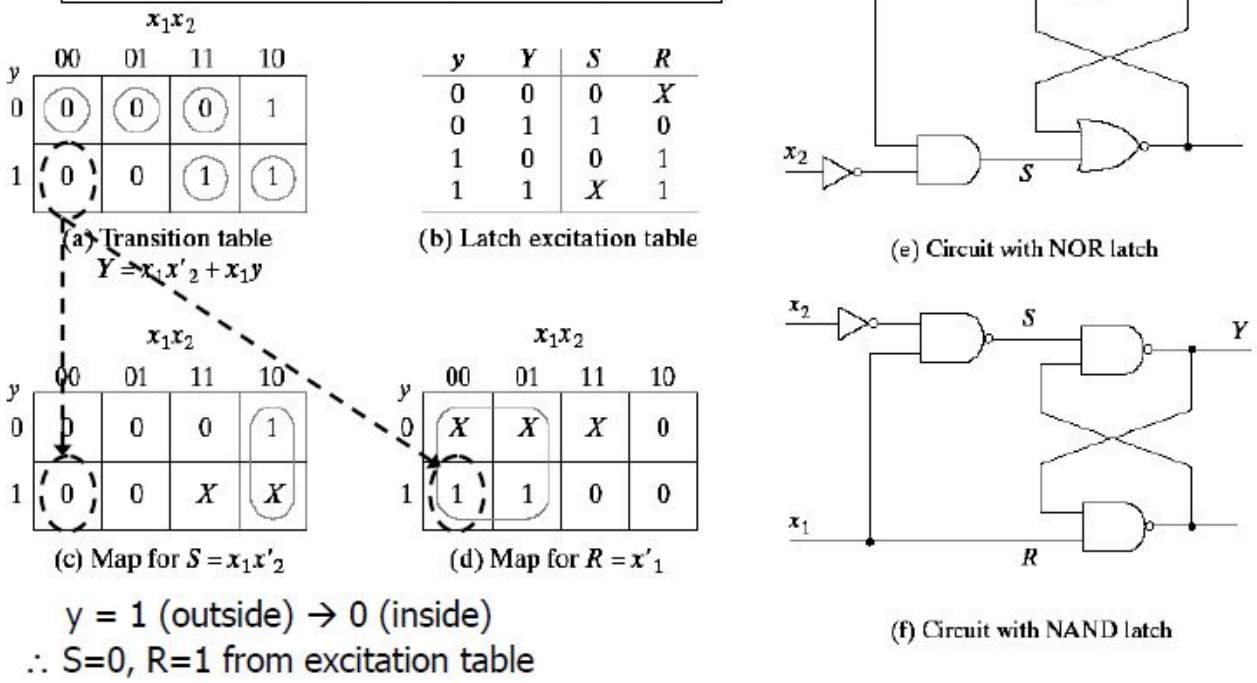
Implementation Procedure

Procedure to implement an asynchronous sequential circuits with SR latches:

1. Given a transition table that specifies the excitation function $Y = Y_1Y_2...Y_k$, derive a pair of maps for each S_i and R_i using the latch excitation table
2. Derive the Boolean functions for each S_i and R_i
   (do not to make Si and Ri equal to 1 in the same minterm square)
3. Draw the logic diagram using *k* latches together with the gates required to generate the S and R
   (for NAND latch, use the complemented values in step 2)

Implementation Example

Excitation table: list the required S and R for each possible transition from y to Y



y = 1 (outside) → 0 (inside)
∴ S=0, R=1 from excitation table

Debounce Circuit

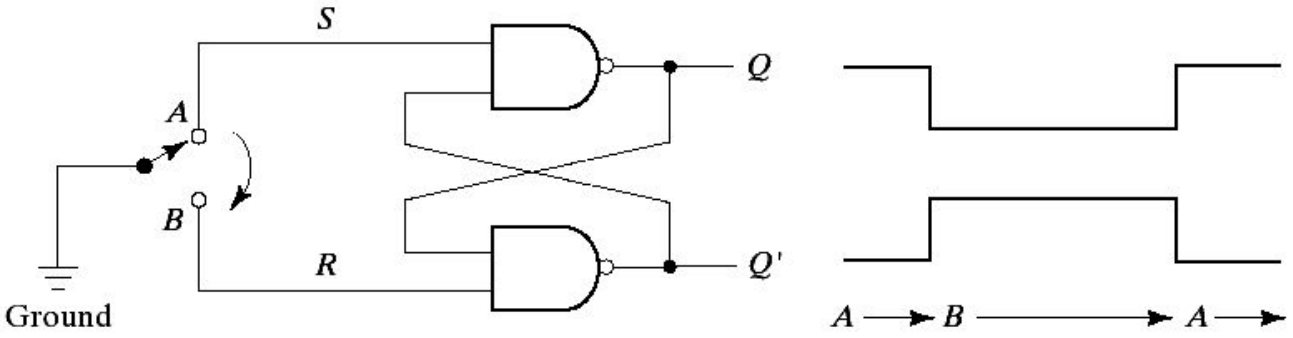
Mechanical switches are often used to generate binary signals to a digital circuit

- It may vibrate or bounce several times before going to a final rest
- Cause the signal to oscillate between 1 and 0

A debounce circuit can remove the series of pulses from a contact bounce and produce a single smooth transition

- Position A (SR=01) -> bouncing (SR=11) -> Position B (SR=10)

Q = 1 (set) □ Q = 1 (no change) □ Q = 0 (reset)


S
A
B
R
Q
Q'
Ground
A
B
A