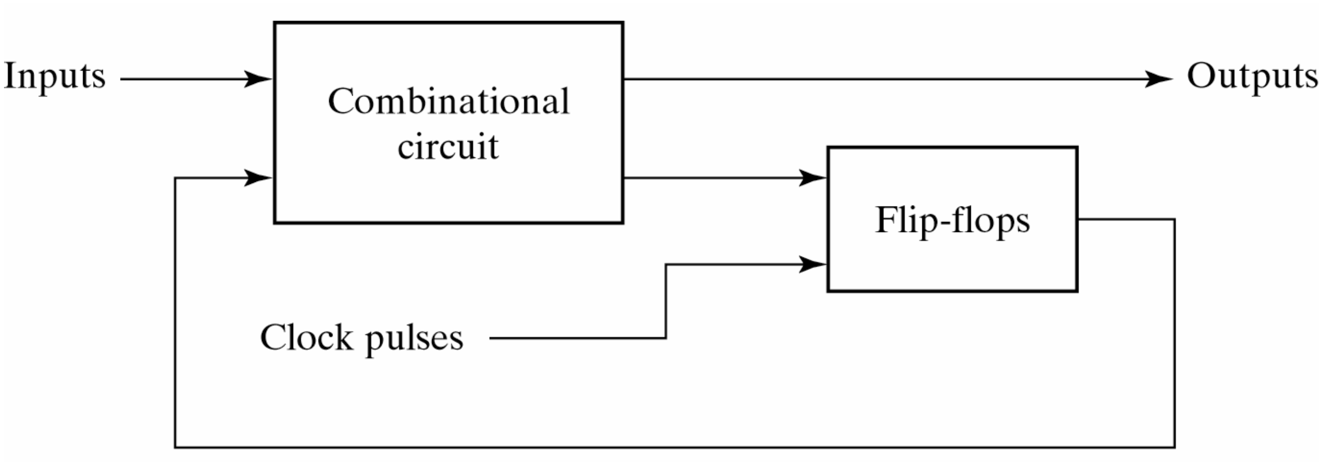
Asynchronous Logic I

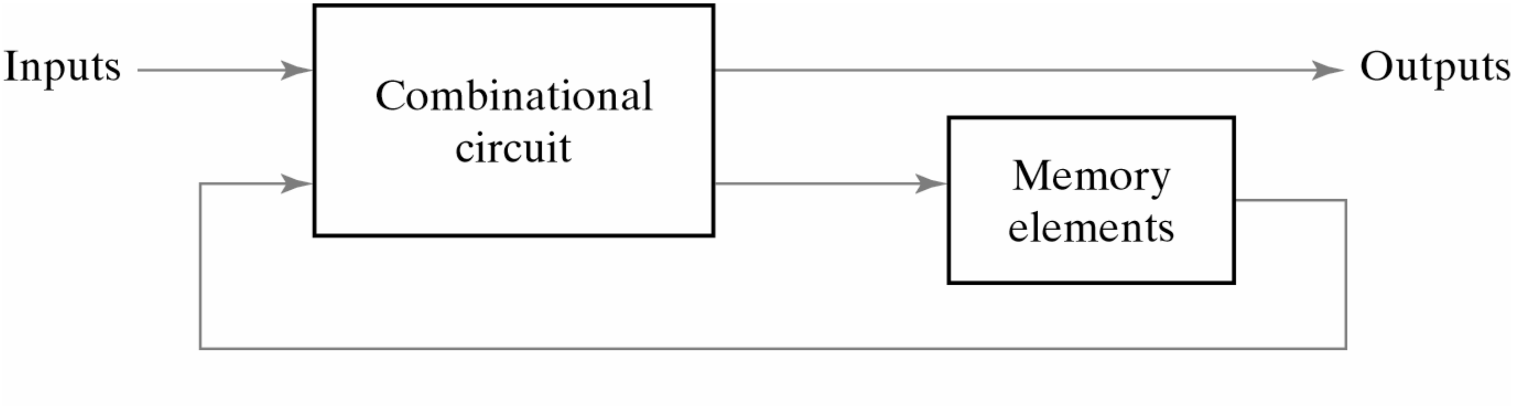
Synchronous Sequential Logic

- Flip-flops and registers are used to synchronise the operation sequential circuits to a train of clock pulses.
- Synchronisation allows the use of RTL and HDL logic descriptions.
- These techniques allow large scale implementation.



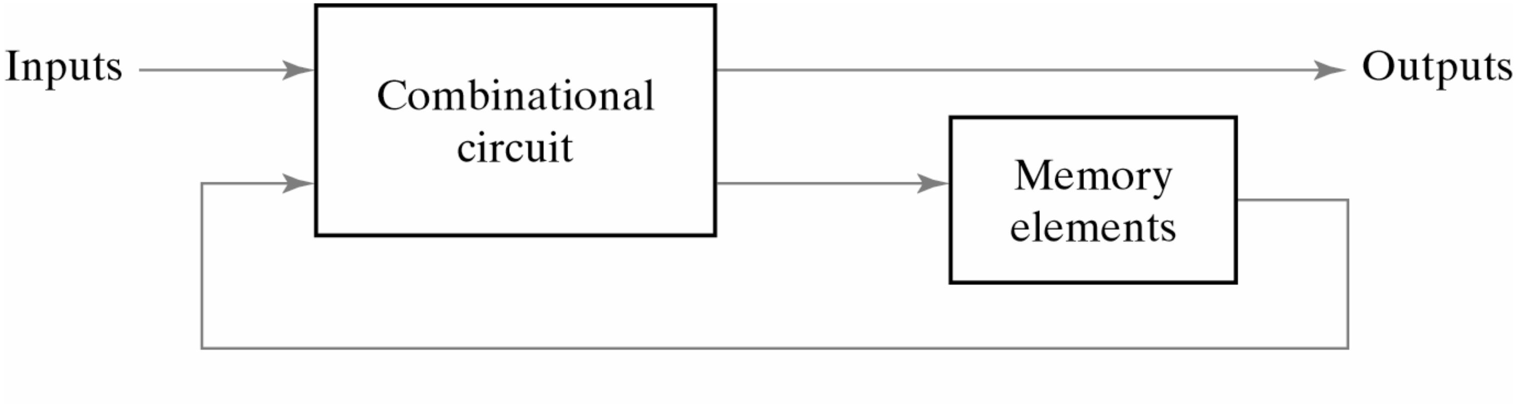
Asynchronous Sequential Logic

- Asynchronous sequential circuits do not use clock pulses.
- The state of the circuit changes immediately after a change of input variables.
- Memory elements are latches (without clock) or time-delay elements.



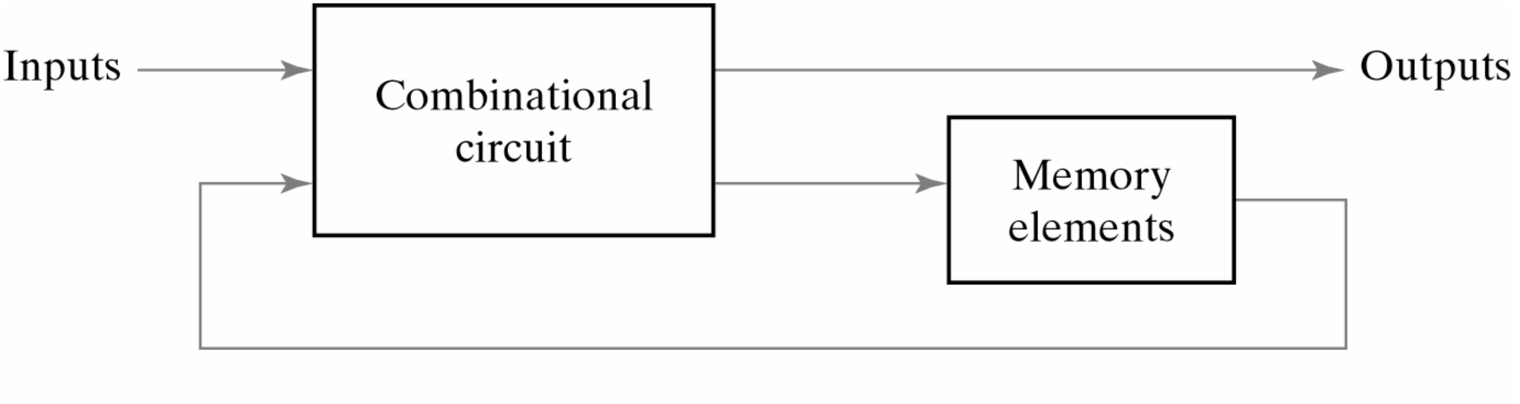
Applications

- High speed operation, e.g. when a circuit must respond quickly without waiting for a clock pulse.
- Interfacing between two circuits which have independent clocks.
- Understanding large synchronous circuits where the global synchronisation assumption breaks down.



Asynchronous Sequential Logic

- It should be understood that asynchronous sequential design is difficult.
- In this lecture we will examine some basic concepts involved.



Asynchronous Sequential Circuit

- Asynchronous sequential circuit:

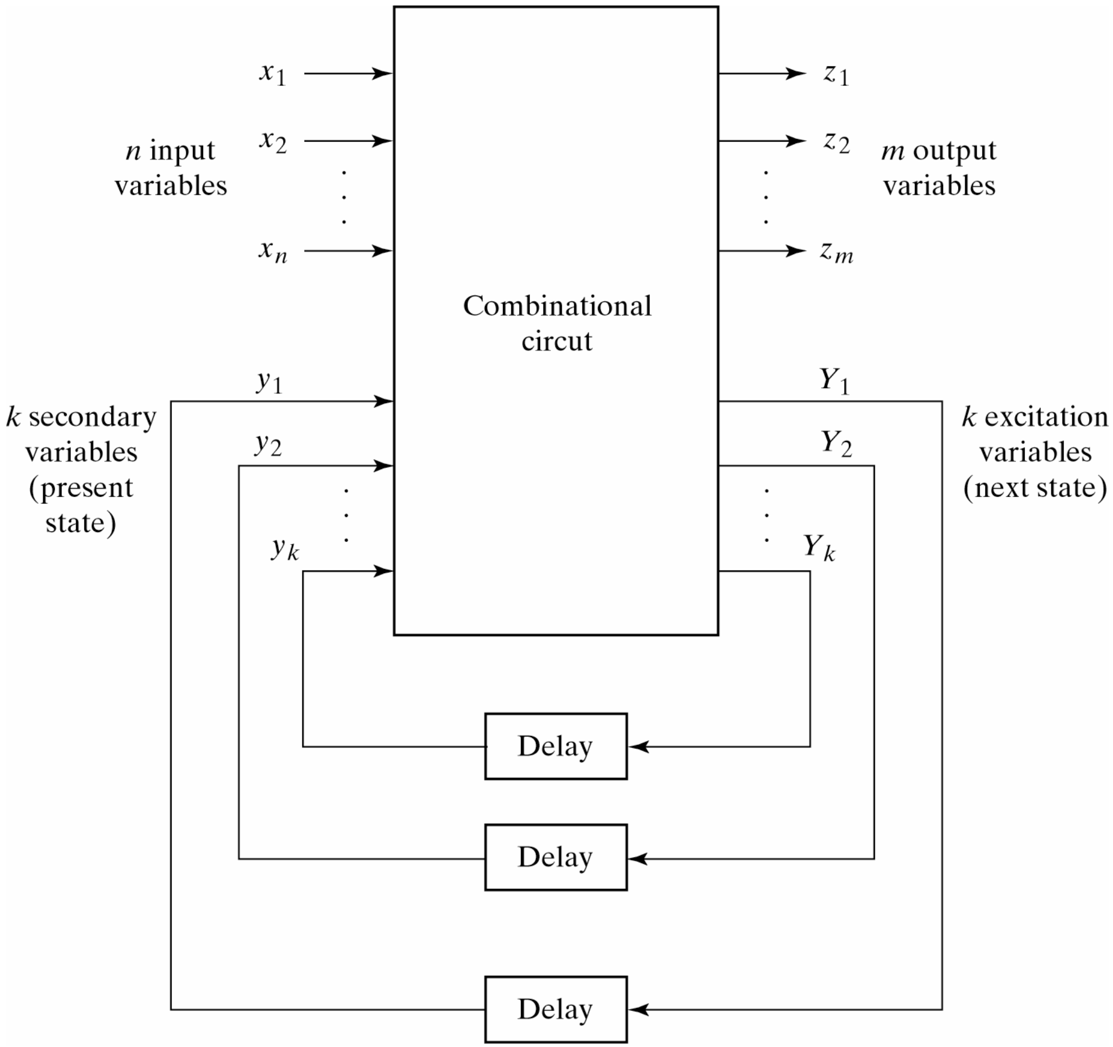
  n input variables

  m output variables

  k internal variables

  2^k internal **STATES**

- k delay elements provide short term memory.



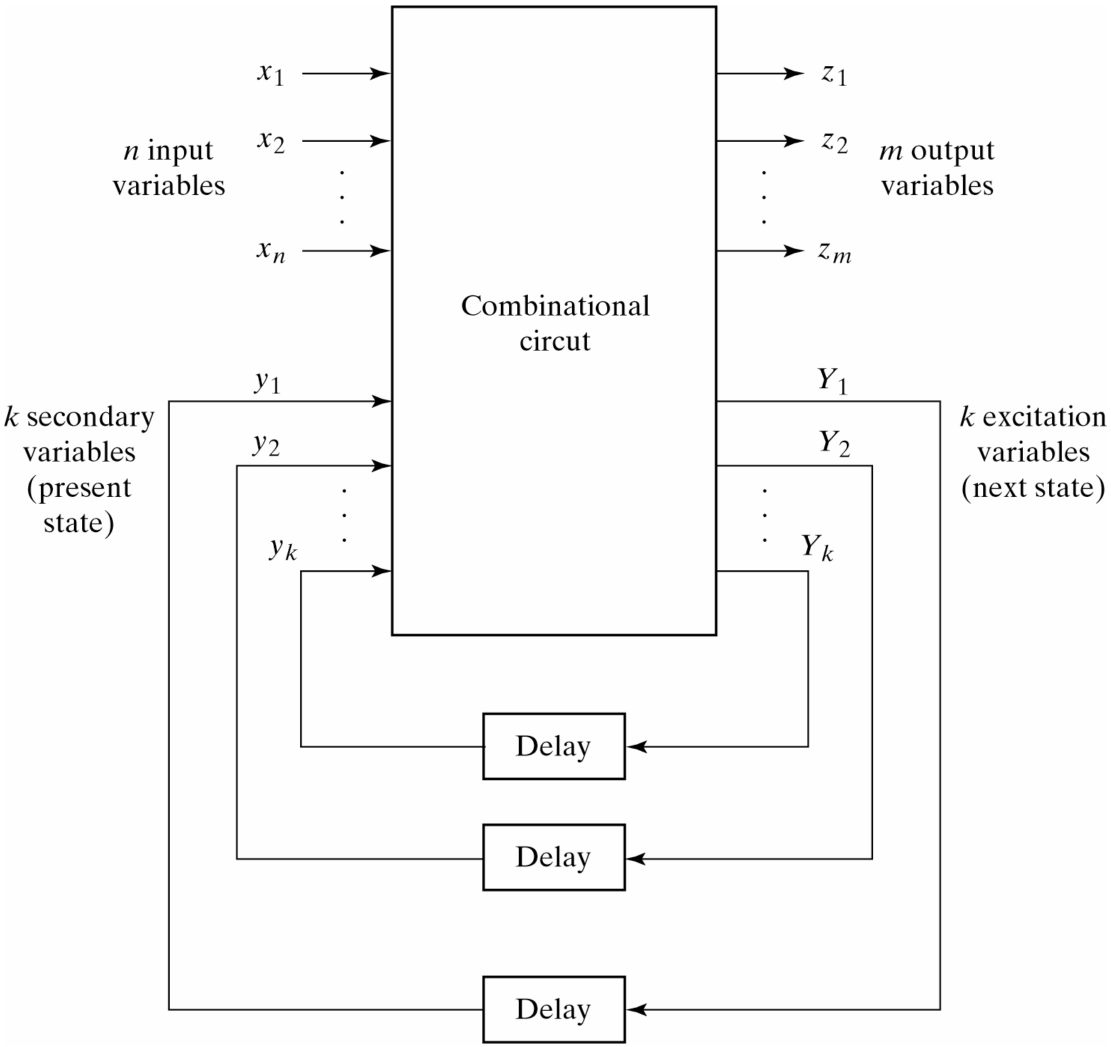
Asynchronous Sequential Circuit

- Present state variables (secondary variables)

$$\{y_1, y_2, \ldots, y_k\}$$

- Next state variables (excitation variables)

$$\{Y_1, Y_2, \ldots, Y_k\}$$



Asynchronous Sequential Circuit

- Delay elements model inherent propagation delays.

  $$y_i(t) = Y_i(t - \tau_i)$$

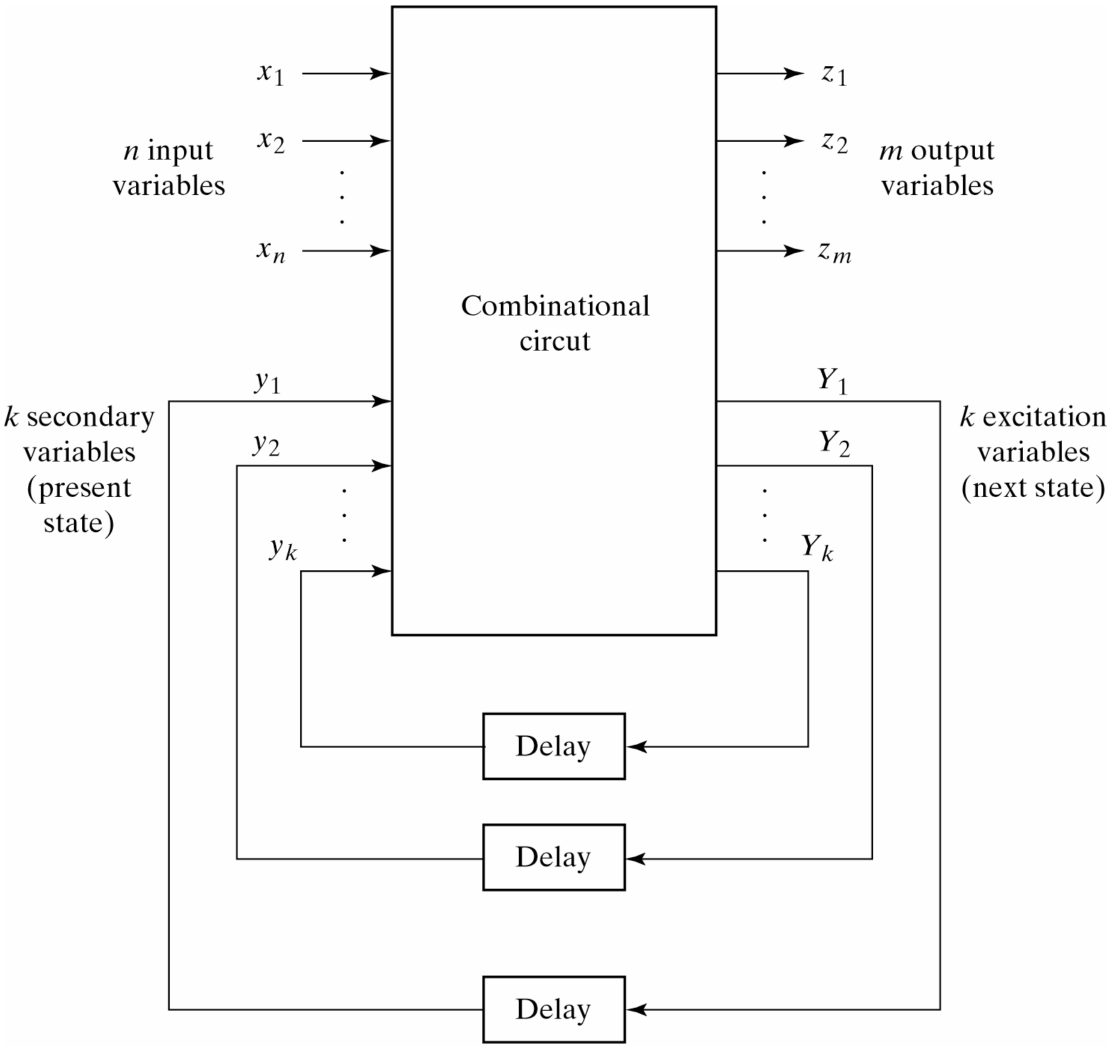
  τ_i is the propagation delay on the ith feedback path.

- Different from Synchronous circuit

  $$y_i(kT) = Y_i((k-1)T)$$

  T is the clock period

  k is the cycle index

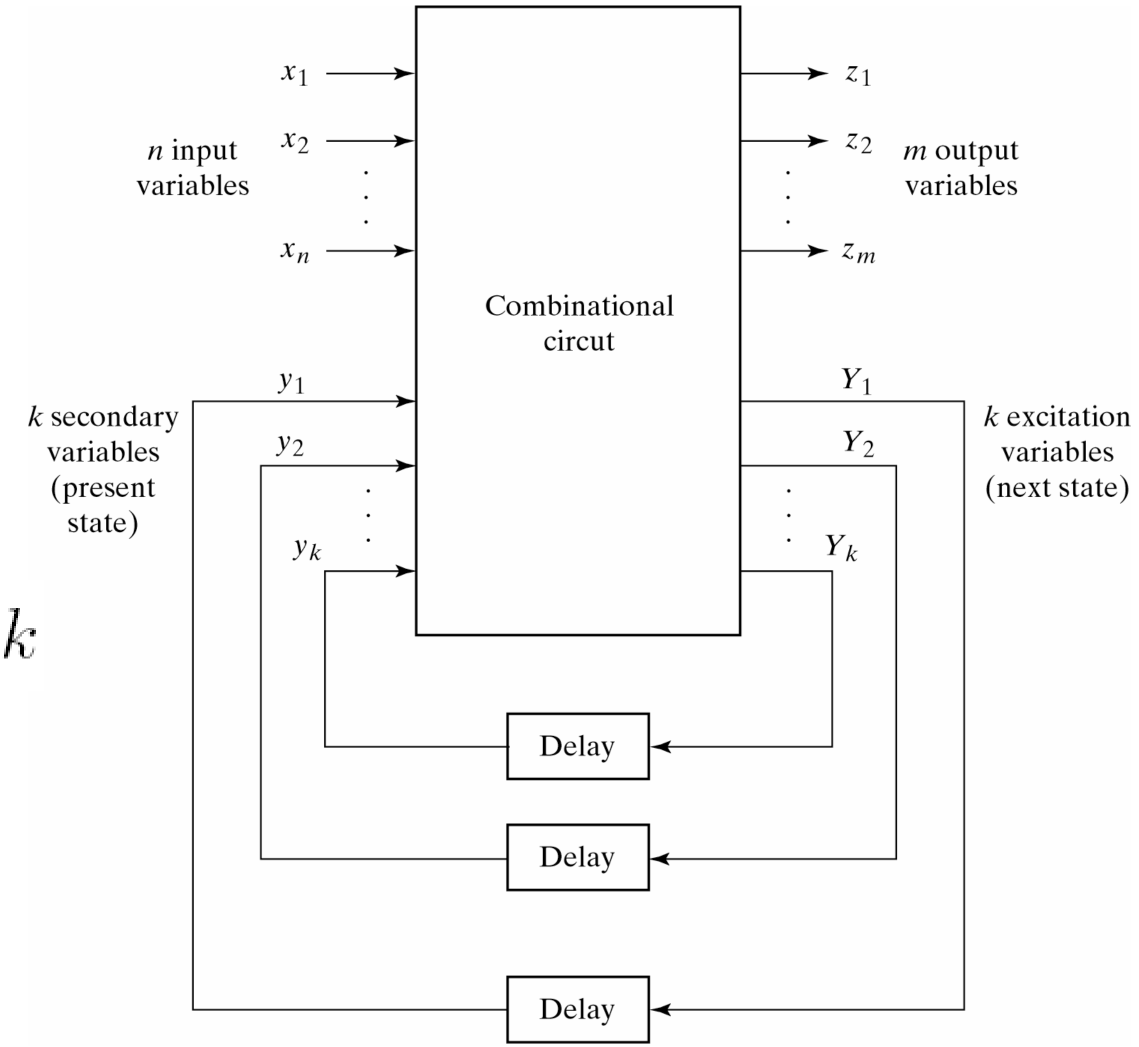


Asynchronous Sequential Circuit

- When the circuit reaches a steady state condition the secondary and excitation variables are equal.

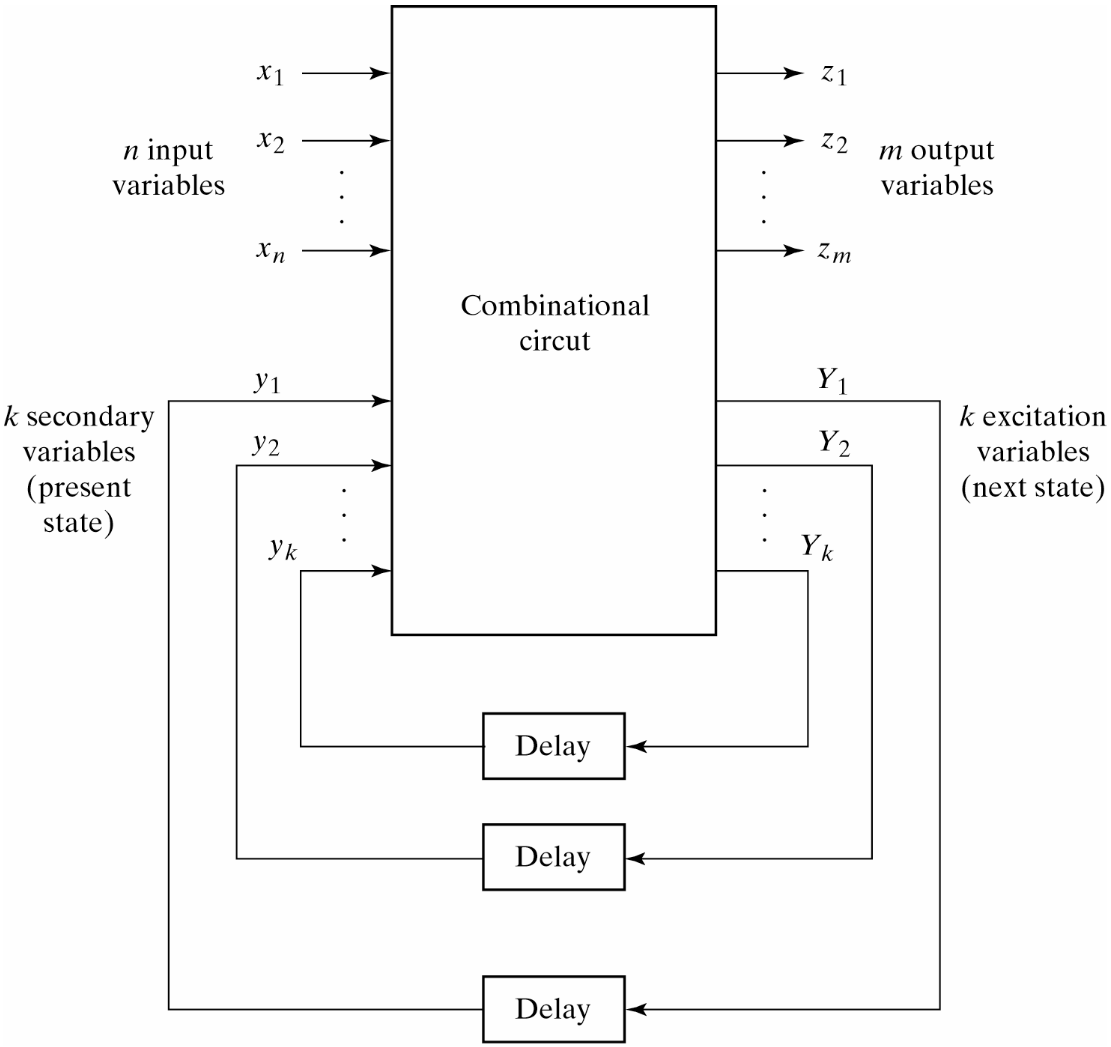
$$y_i = Y_i \, , \, \forall \, i = 1, 2, \ldots, k$$

- The circuit is **stable** if steady state is reached after a transition.



Asynchronous Sequential Circuit

- The circuit is **unstable** if the state variables continuously change without settling.
- In comparison, in a synchronous circuit state variables only change upon clock edges, this ensures stability.



Fundamental Mode

- To ensure stability in an asynchronous circuit it is required that the circuit reaches steady state before an input is allowed to change.

- When two or more inputs change it is impossible to tell which one changes first, due to propagation delay differences.

- *As a result inputs are only allowed to change one at a time and the time between input changes must be longer than the time it takes to reach steady state.* Such operation is defined as **Fundamental Mode**.

Fundamental Mode

- In other words, **Fundamental Mode** assumes that input signals change *one at a time* and only when the circuit has reached a *stable condition*.

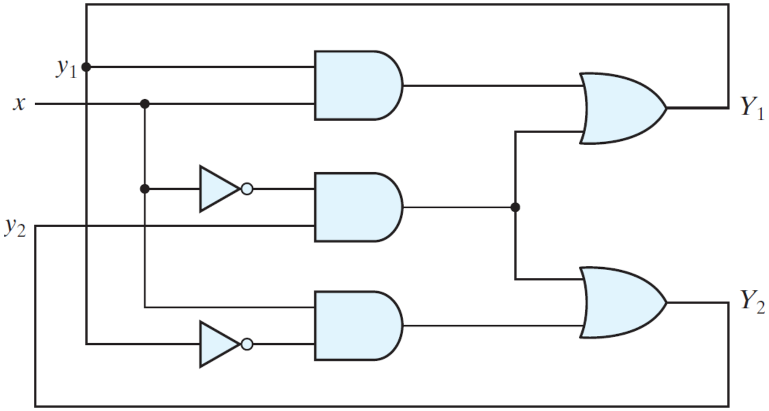
- Contrast this with a synchronous circuit where state changes occur only at clock edges and in response to *multiple input changes*.

Analyze Asynchronous Circuits

- Transition Table
  - Excitation variables as outputs and the secondary variables as internal inputs.
  - Secondary variables are in rows, and the input variables are in columns.
  - Combination of internal states with the input is called the **total state** of the circuit.
  - Stable states (Y = y) are circled to indicate a stable condition.

Procedure for Obtaining a Transition Table

- Determine all feedback loops in the circuit.
- Designate the output of each feedback loop with variable Y_i and its corresponding input with y_i.
- Derive the Boolean functions of all Y's as a function of the external inputs and the internal input y's.
- Plot each Y function in a map, using the y variables for the rows and the external inputs for the columns.
- Combine all the maps into one table.
- Circle those values of Y in each square that are equal to the value of y.



y_1y_2 \ x	0	1
00	0	0
01	1	0
11	1	1
10	0	1

(a) Map for $Y_1 = xy_1 + x'y_2$

y_1y_2 \ x	0	1
00	0	1
01	1	1
11	1	0
10	0	0

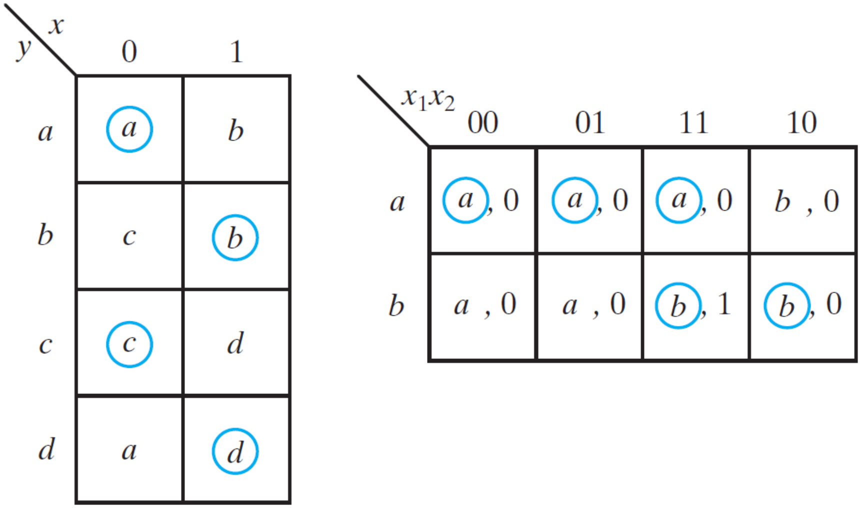
(b) Map for $Y_2 = xy'_1 + x'y_2$

y_1y_2 \ x	0	1
00	00	01
01	11	01
11	11	10
10	00	10

(c) Transition table

Analyze Asynchronous Circuits

- Flow Table
  - Name the states by letter symbols without making specific reference to their binary values.



y \ x	0	1
a	ⓐ	b
b	c	ⓑ
c	ⓒ	d
d	a	ⓓ

(a) Four states with one input

x_1x_2	00	01	11	10
a	ⓐ, 0	ⓐ, 0	ⓐ, 0	b , 0
b	a , 0	a , 0	ⓑ, 1	ⓑ, 0

(b) Two states with two inputs and one output

Derive Circuit from Flow Table

- In design process, if we know the flow table, we could derive a circuit from the flow table and draw the logic diagram.
- Assign binary numbers to the states to form the transition table.
- Simplified the circuit using Karnaugh Map, and draw the logic diagram.

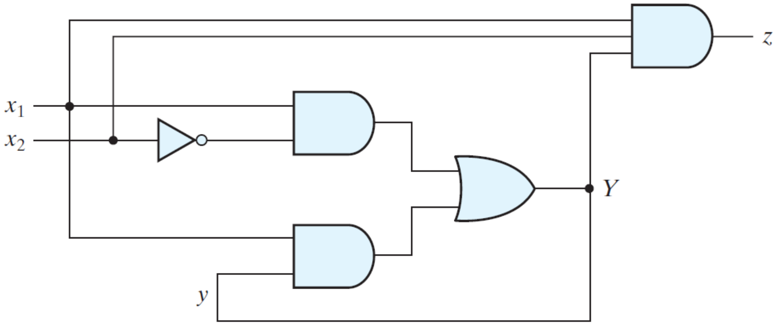
x_1x_2	00	01	11	10
a	a, 0	a, 0	a, 0	b, 0
b	a, 0	a, 0	b, 1	b, 0

y \ x_1x_2	00	01	11	10
0	0	0	0	1
1	0	0	1	1

(a) Transition table
$Y = x_1x'_2 + x_1y$

y \ x_1x_2	00	01	11	10
0	0	0	0	0
1	0	0	1	0

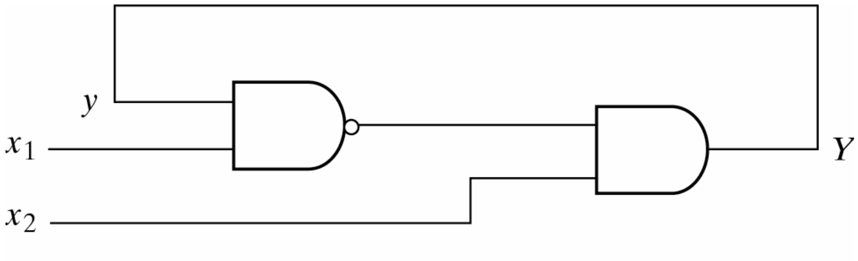
(b) Map for output
$z = x_1x_2\,y$



(c) Logic diagram

Stability

- Because of the feedback connection, the circuit may become unstable.
- This simple feedback circuit demonstrates instability.



(a) Logic diagram

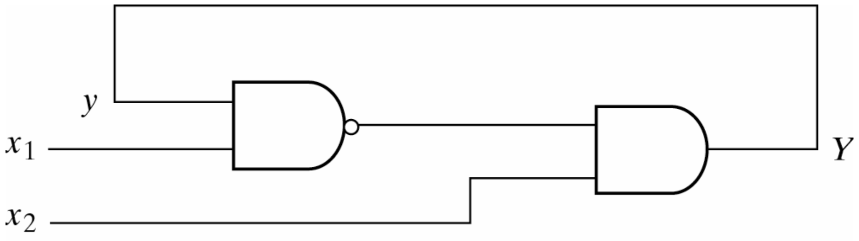
$$Y = \overline{x_1 . y} . x_2$$

Stability

- The Karnaugh Map is filled out using the sum of products expression.

$$Y = \overline{x_1}.x_2 + \overline{y}.x_2$$

- $\overline{x_1}.x_2$ corresponds to ones in the second column of the map.
- $\overline{y}.x_2$ corresponds to ones in the second and third boxes of the first row.



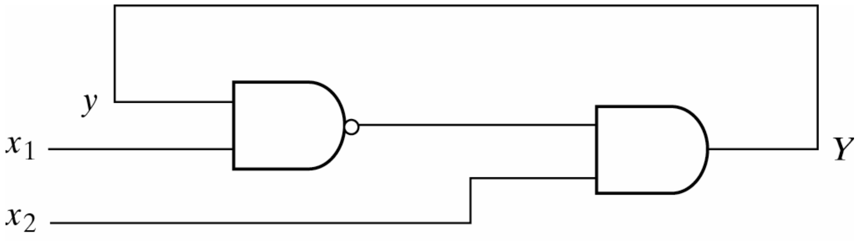
(a) Logic diagram

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

(b) Transition table

Stability

- The circled Y (excitation variable) values correspond to stable states.
- Because these Y values match the y (secondary variable) values.
- The non-circled values are unstable states. Because the Y values do not match the y values.



(a) Logic diagram

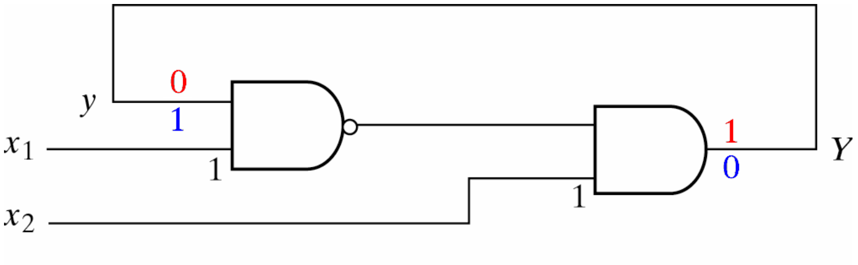
y \ $x_1 x_2$	00	01	11	10
0	0	1	1	0
1	0	1	0	0

(b) Transition table

Stability

- For this circuit the third column does not have any stable states.

- Assuming the inputs x_1 and x_2 are both one the circuit will oscillate between states.



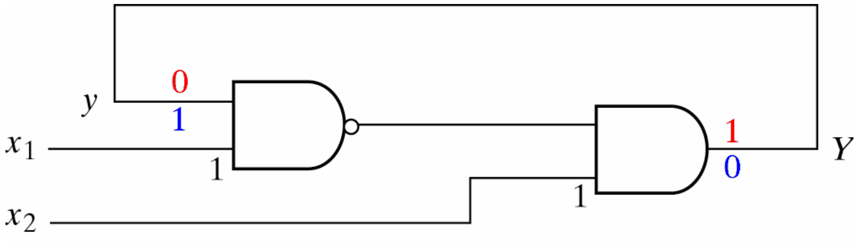
(a) Logic diagram

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

(b) Transition table

Stability

- When the secondary state variable y is 1 the excitation variable becomes 0.
- When the secondary state variable y is 0 the excitation variable becomes 1.
- Y oscillates between 0 and 1. The period of oscillation is twice the propagation time around the loop.



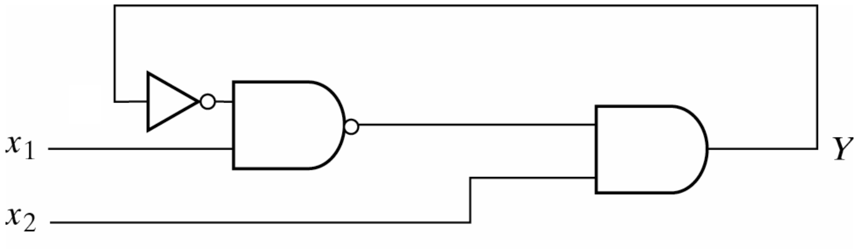
(a) Logic diagram

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

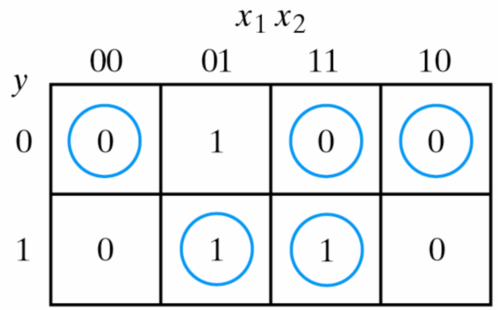
(b) Transition table

Stability

- The circuit can be made stable by inverting the secondary variable.
- Now both states in the third column are stable.
- In **fundamental mode** only transitions between adjacent states are permitted.



(a) Logic diagram



$x_1 x_2$

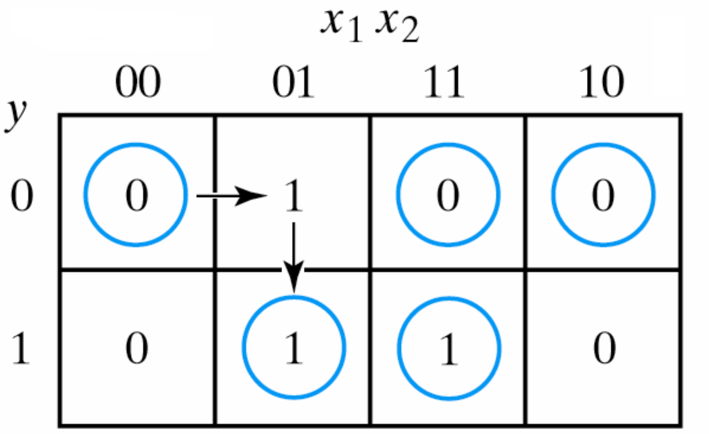
y	00	01	11	10
0	0	1	0	0
1	0	1	1	0

(b) Transition table

$$Y = \overline{x_1}.x_2 + y.x_2$$

Stability

- Initial state yx_1x_2 = 000
- x_2 transitions from 0 to 1. The circuit enters an unstable state yx_1x_2 = 001.
- Then the secondary variable y transitions from 0 to 1 and the circuit enters the stable state yx_1x_2 = 101.



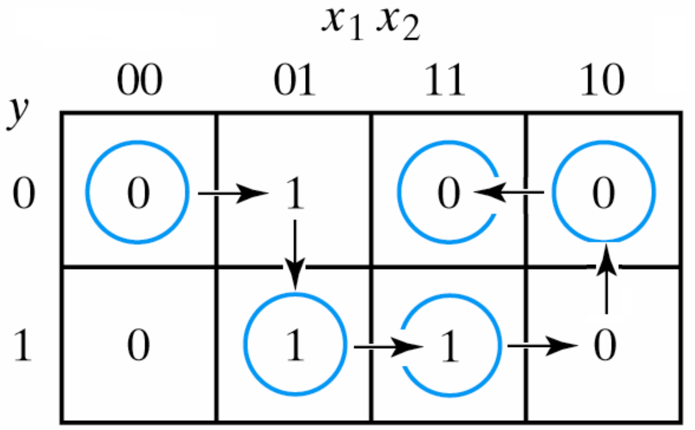
y \ $x_1 x_2$	00	01	11	10
0	0	1	0	0
1	0	1	1	0

Stability

- x_1 transition from 0 to 1
  yx_1x_2 : 101 → 111

- x_2 transition from 1 to 0
  yx_1x_2 : 111 → 110 → 010

- x_2 transition from 0 to 1
  yx_1x_2 : 010 → 011



Race Conditions

- A **race condition** exists in an asynchronous sequential circuit when two or more binary state variables change in response to a change in an input variable.

- When unequal delays are encountered, a race condition may cause the state variables to change in an unpredictable manner.

Race Conditions

- For example, if the state variables must change from 00 ➔ 11
- The difference in delays may cause the first variable to change faster than the second so the state variables change in the sequence

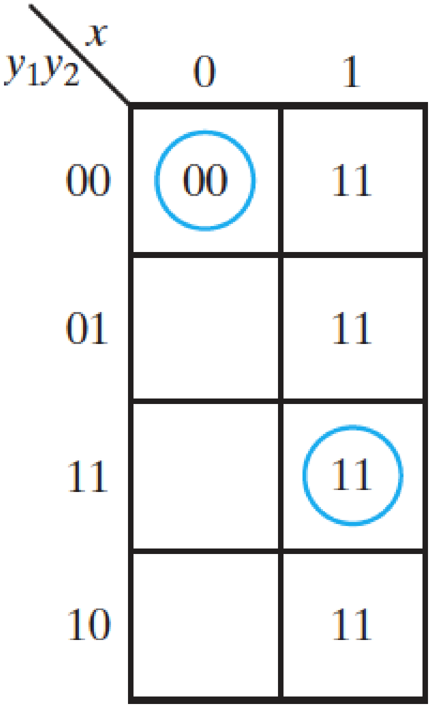
  00 ➔ 10 ➔ 11.
- If the second variable changes faster than the first, the state variables will change in the sequence

  00 ➔ 01 ➔ 11.

Race Conditions

- If the final stable state that the circuit reaches does not depend on the order in which the state variables changes, the race is called a **noncritical race**.
- If it is possible to end up in two or more different stable states, depending on the order in which the state variables change, then the race is a **critical race**. For proper operation, critical races must be avoided.

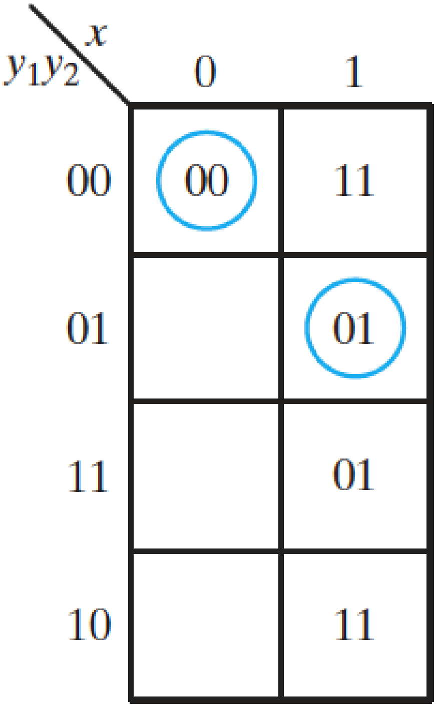
Noncritical Race



y_1y_2 \ x	0	1
00	00	11
01		11
11		11
10		11

(a) Possible transitions:

00 → 11
00 → 01 → 11
00 → 10 → 11



y_1y_2 \ x	0	1
00	00	11
01		01
11		01
10		11

(b) Possible transitions:

00 → 11 → 01
00 → 01
00 → 10 → 11 → 01

Critical Race

y_1y_2 \ x	0	1
00	00	11
01		01
11		11
10		10

(a) Possible transitions:

00 → 11
00 → 01
00 → 10

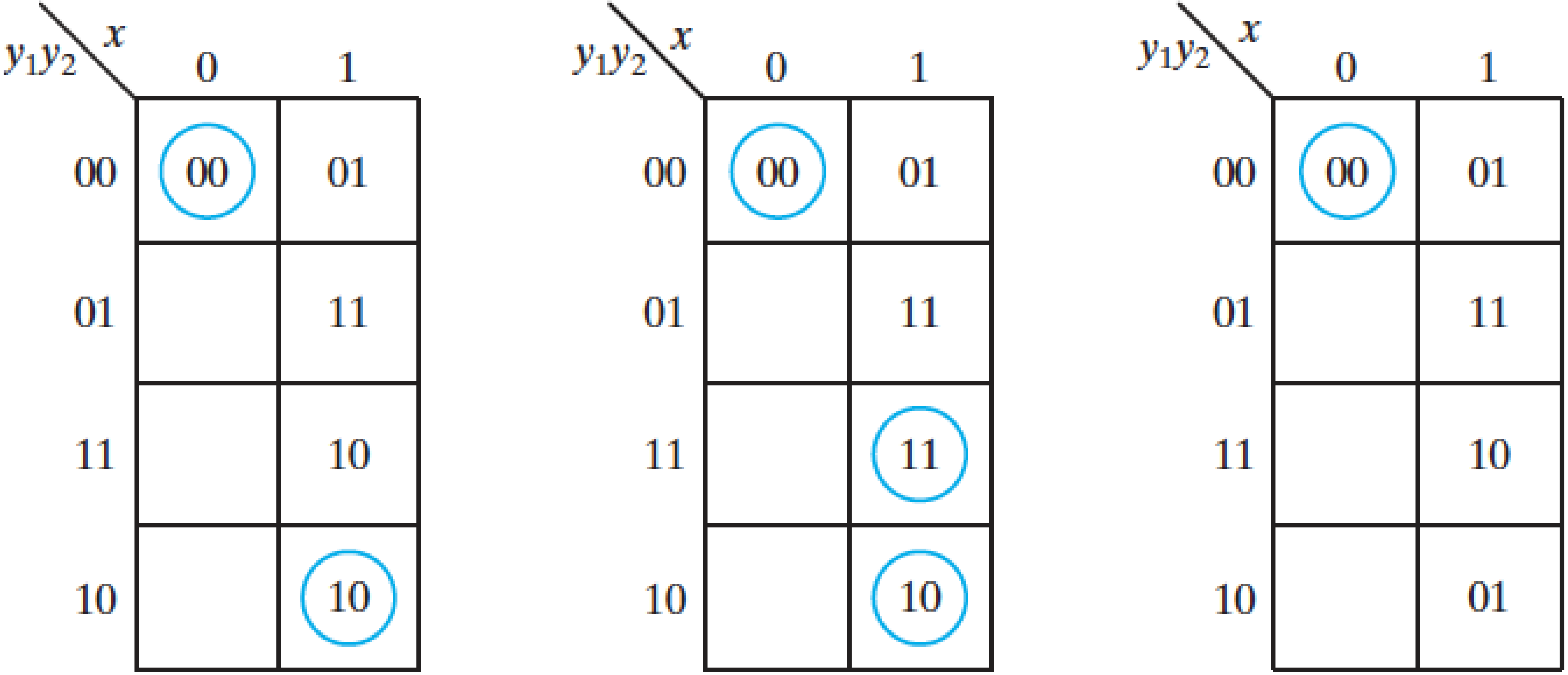
y_1y_2 \ x	0	1
00	00	11
01		11
11		11
10		10

(b) Possible transitions:

00 → 11
00 → 01 → 11
00 → 10

Avoid Races

- Critical races may be avoided by making a proper binary assignment to the state variables.
- The state variables must be assigned binary numbers in such a way that only one state variable can change at any one time when a state transition occurs in the transition or flow table.



y_1y_2 \ x	0	1
00	00	01
01		11
11		10
10		10

(a) State transition:

00 → 01 → 11 → 10

y_1y_2 \ x	0	1
00	00	01
01		11
11		11
10		10

(b) State transition:

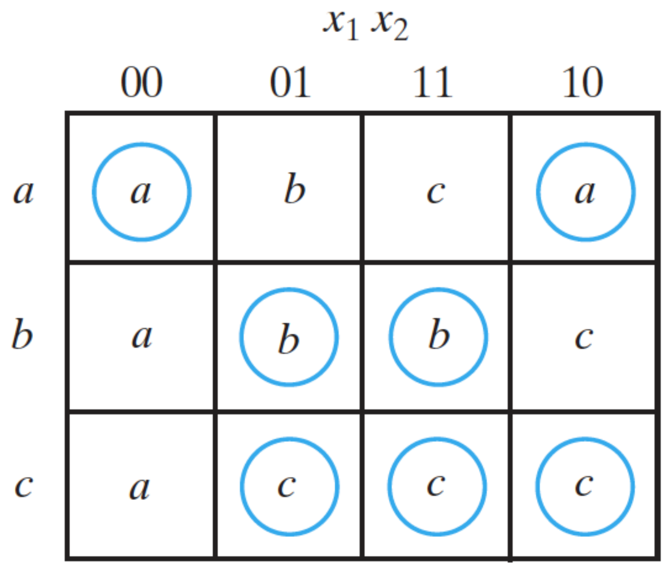
00 → 01 → 11

y_1y_2 \ x	0	1
00	00	01
01		11
11		10
10		01

(c) Unstable

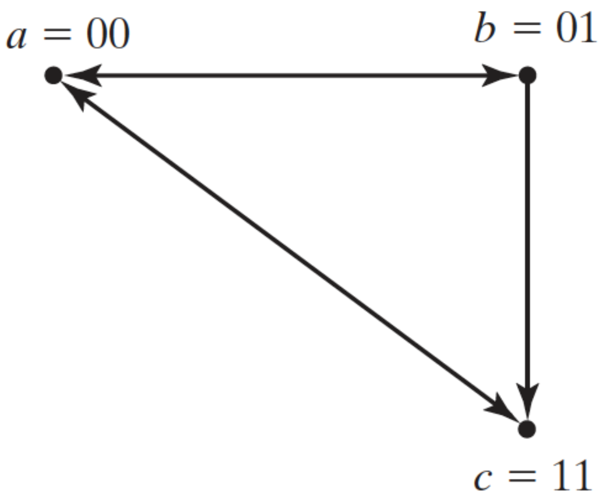
→ 01 → 11 → 10 →

Race-Free State Assignment



$x_1 x_2$	00	01	11	10
a	*a*	*b*	*c*	*a*
b	*a*	*b*	*b*	*c*
c	*a*	*c*	*c*	*c*

(a) Flow table

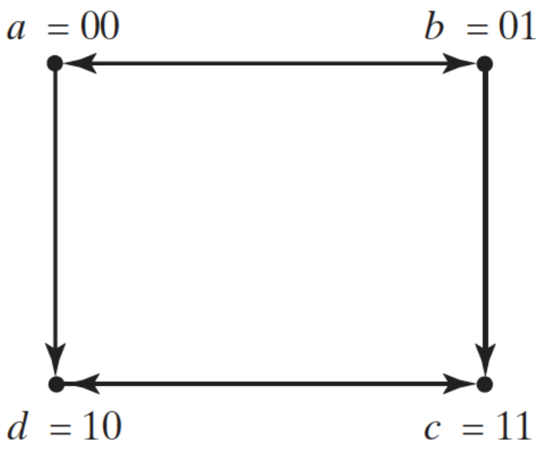


(b) Transition diagram

Race-Free State Assignment

	x_1x_2			
	00	01	11	10
a	*a*	*b*	*d*	*a*
b	*a*	*b*	*b*	*c*
c	*d*	*c*	*c*	*c*
d	*a*	–	*c*	–

(a) Flow table



(b) Transition diagram

Race-Free State Assignment

	x_1x_2			
	00	01	11	10
$a = 00$	00	01	10	00
$b = 01$	00	01	01	11
$c = 11$	10	11	11	11
$d = 10$	00	–	11	–