E&CE 223
Digital Circuits & Systems

Lecture Transparencies
(Asynchronous Sequential Circuits)

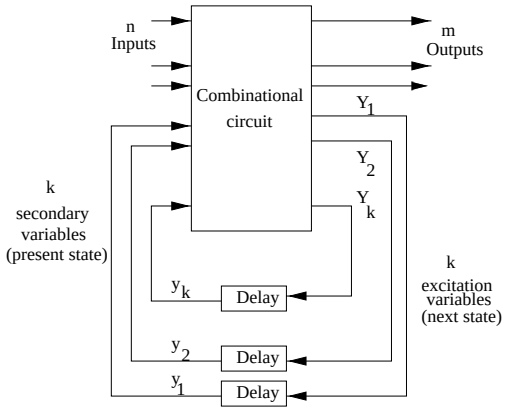
M. Sachdev

Section 5

- **Major topics**
  - Secondary variables
  - Excitation table
  - Transition table
  - Race hazard
  - Cycle and stability
  - Primitive flow table
  - Merged flow table
  - Hazards

Introduction

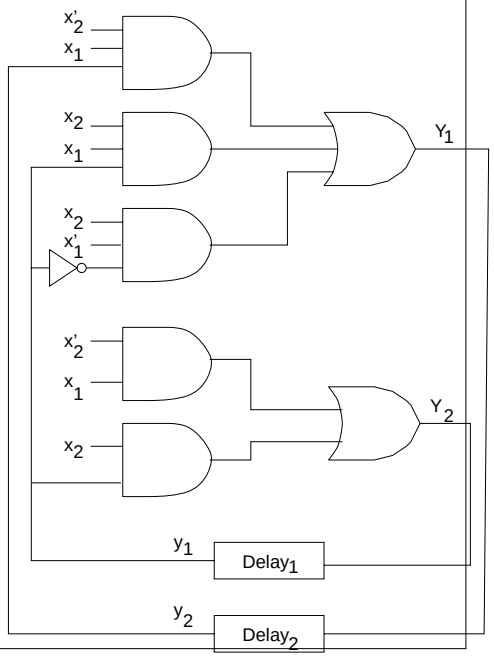
- **Synchronous, sequential circuit has a synchronizing signal (clock)**
  - Asynchronous sequential logic has no clock signal
  - Also known as fundamental mode sequential logic
- **Memory is achieved by**
  - Unclocked latches, or
  - Delay elements, or
  - Inherent delay in circuits



Definitions

- **The asynchronous circuit is in**
  - **Stable state** if $y_i = Y_i$ for all *i*
  - **Transition state** if $y_i \neq Y_i$ for all *i*
- **In analysis and design one normally assumes that the combinational circuit is ideal (0 delay) and considers the delays as lumped delay elements**
- **Design restrictions**
  - Normally requires that inputs **do not** change simultaneously (i.e., within the settling time of the circuit after an input change)
  - Hence, input changes only in stable states
- **Excitation table**
  - K map of Y_i and outputs in terms of y_i and inputs

- **Transition table**
  - Excitation table for Y_i with stable states marked
- **Flow table**
  - Same table as transition table, but in symbolic form
- **Example: analyze the circuit**
  - $Y_2 = x'_2x_1y_2 + x_2x_1y_1 + x_2x'_1y'_1$
  - $Y_1 = x'_2x_1 + x_2y_1$
  -



Excitation and Transition Table

- **Circle indicates a stable state**
  - Every row has a stable state, hence 4 stable states
- **When $y_2y_1 \neq Y_2Y_1$ (due to input change) a transition to new row will occur**
- **Example**

  x2x1= Y2y1 = y2y1 = 00 (ini. cond.)

  x2x1 =00 --> 01 --> 11 --> 10 --> 00

y2y1 \ x2x1	00	01	11	10
00	(00)	01	00	10
01	00	(01)	11	(01)
11	00	(11)	(11)	01
10	00	11	00	(10)

Y2Y1

y2y1	x2x1	Y2Y1	
00	00	00	(initial condition)
	01	01	
01	01	01	
	11	11	
11	11	11	
	10	01	
01	10	01	
	00	00	
00	00	00	

Flow Table

- **Let 00 = a; 01 = b; 11 = c; 01 = d**
  - 00 = a
  - 01 = b
  - 11 = c
  - 01 = d

  The resulting flow table is

present state	x2x1 = 00	01	11	10
a	(a)	b	a	d
b	a	(b)	c	(b)
c	a	(c)	(c)	b
d	a	c	a	(d)

next state

Race Conditions

- **A race condition is caused when two or more binary state variables change value due to change in an input variable**
  - Unequal delays may cause state variables to change in unpredictable manner
  - Race condition may be
    (i) non- critical, or
    (ii) critical

y1y2 \ x	0	1
00	(00)	11
01		11
11		(11)
10		11

Possible transistions
00 -> 11
00 -> 01 -> 11
00 -> 10 -> 11

non-critical race

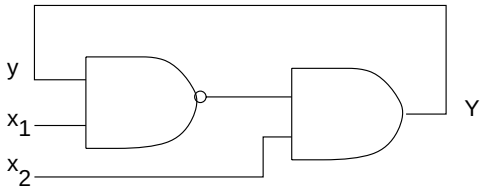
y1y2 \ x	0	1
00	(00)	11
01		(01)
11		(11)
10		(10)

Possible transistions
00 -> 11
00 -> 01
00 -> 10

critical race

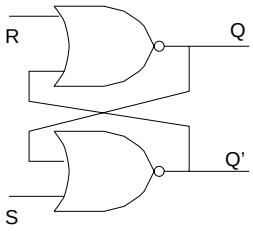
Stability Consideration

- **Circuit is unstable if it oscillates between two unstable states**
  - $Y = (x_1 y)' x_2 = x'_1 x_2 + x_2 y'$
  -

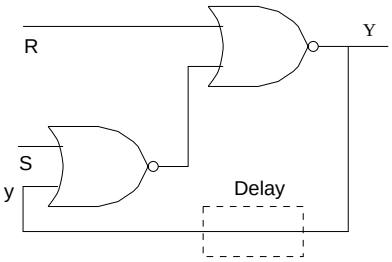


y \ x1x2	00	01	11	10
0	(0)	1	1	(0)
1	0	(1)	0	0

Circuits with Latches



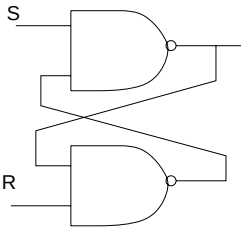
S	R	Q	Q'
1	0	1	0
0	0	1	0
0	1	0	1
0	0	0	1
1	1	0	0



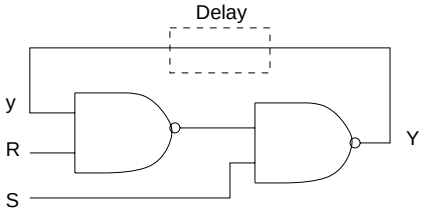
y \ SR	00	01	11	10
0	0	0	0	1
1	1	0	0	1

$Y = S + R'y$

Circuits with Latches



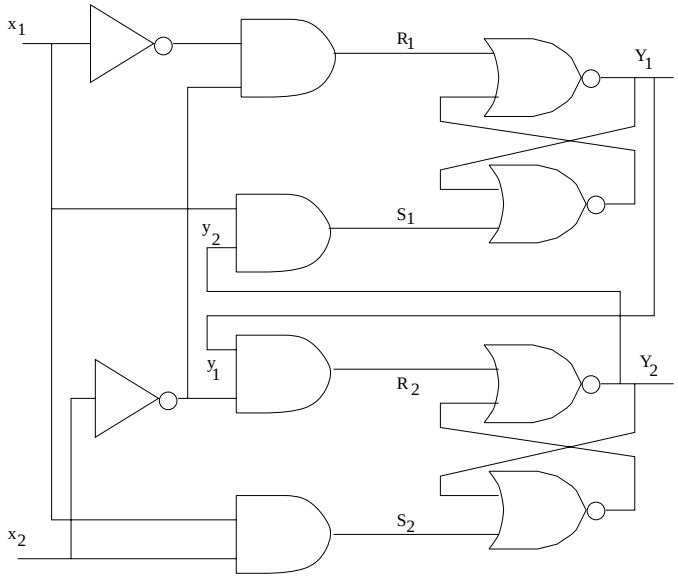
S	R	Q	Q'	
1	0	0	1	
1	1	0	1	(after SR = 10)
0	1	1	0	
1	1	1	0	(after SR = 01)
0	0	1	1	



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

$Y = S' + Ry$

Analysis Example



Analysis: Transition Table

- **Analysis Procedure for NOR latch based asynchronous circuit**
  - (i) Label each latch o/p with Y_i and feed back path with y_i
  - (ii) Derive Boolean functions for S_i and R_i
  - (iii) Check SR = 0 for each NOR latch
  - (iv) Evaluate Y = S + R'y for each latch
  - (v) Construct the transition table
  - (vi) Circle all stable states

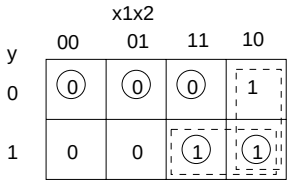
y1y2 \ x1x2	00	01	11	10
00	(00)	(00)	01	(00)
01	(01)	(01)	11	11
11	00	(11)	(11)	10
10	00	(10)	11	(10)

Y1 = S1 + R'1y1 = x1y2 + (x1 + x2)y1 = x1y2 + x1y1 + x2y1

Y2 = S2 + R'2y2 = x1x2 + (x2 +y'1)y2 = x1x2 + x2y2 + y'1y2

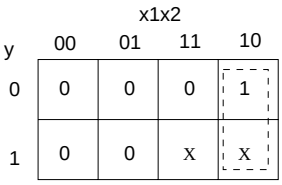
Implementation: Example

- **Given the transition table, obtain the sequential circuit with SR latches**

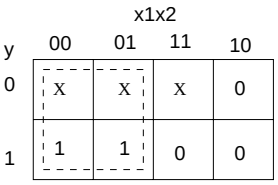


y	Y	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Y= x1x'2 + x1y



S = x1x'2



R = x'1

Implementation: Procedure

- **Given the transition table, the general procedure for implementing a circuit with SR latches is**

  (i) Derive a pair of maps for each S_i and R_i (i = 1,2, ...k)

  (ii) Derive the simplified Boolean function for each Si and Ri (remember $S_iR_i = 0$)

  (iii) Draw the logic diagram using k latches

Design Example

- **To design a gated latch**
  - D, data input; G, gating input; Q, the output
  - Q <-- D if G = 1 and Q retains its previous value if G = 0
- **Gated latch state table**

state	Inputs		Output	Comments
	D	G	Q	
a	0	1	0	G = 1
b	1	1	1	G = 1
c	0	0	0	after state a or d
d	1	0	0	after state c
e	1	0	1	after state b or f
f	0	0	1	after state e

Gated Latch: Flow Table

present state	DG = 00	DG = 01	DG = 11	DG = 10
a	c, -	(a), 0	b, -	-, -
b	-, -	a, -	(b), 1	e, -
c	(c), 0	a, -	-, -	d, -
d	c, -	-, -	b, -	(d), 0
e	f, -	-, -	b, -	(e), 1
f	(f), 1	a, -	-, -	e, -

- The primitive flow table has only one stable state in each row

Primitive Flow Table Reduction

	DG 00	01	11	10
a	c, -	(a,)0	b, -	-, -
c	(c,)0	a, -	-, -	d, -
d	c, -	-, -	b, -	(d,)0

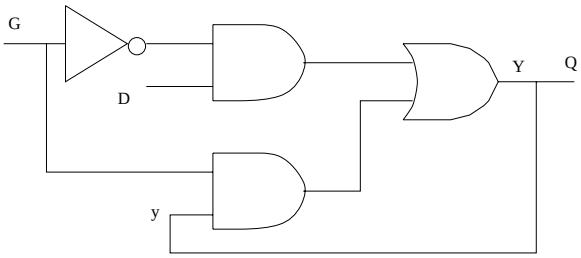
	DG 00	01	11	10
b	-, -	a, -	(b,)1	e, -
e	f, -	-, -	b, -	(e,)1
f	(f,)1	a, -	-, -	e, -

	DG 00	01	11	10
a, c, d	(c,)0	(a,)0	b, -	(d,)0
b, e, f	(f,)1	a, -	(b,)1	(e,)1

Transition Table and Logic Diagram

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

Y = DG +G'y



Unstable States

- **Unlike stable states, the unstable states have unspecified outputs**
  - Can cause momentary undesirable output

a	(a, 0)	b, -
b	c, -	(b, 0)
c	(c, 1)	d, -
d	a, -	(d, 1)

Flow table (given)

0	0
X	0
1	1
X	1

Output assignment

Design procedure for asynchronous circuits

(i) Obtain a primitive table from specifications

(ii) Reduce flow table by merging rows in the primitive flow table

(iii) Assign binary state variables to each row of reduced table

(iv) Assign output values to dashes associated with unstable states to obtain the output map

(v) Simplify Boolean functions for excitation and output variables;

(vi) Draw the logic diagram

Systematic reduction of Flow and State Tables

- **Two techniques:**
  - (i) Implications table, and
  - (ii) Merger diagram
- **Implication table**
  - Tabulation of possible equivalent states (rows)
  - Tick for equivalent, and X for not equivalent
  - Two states, a, b are equivalent iff

    (i) outputs are equivalent

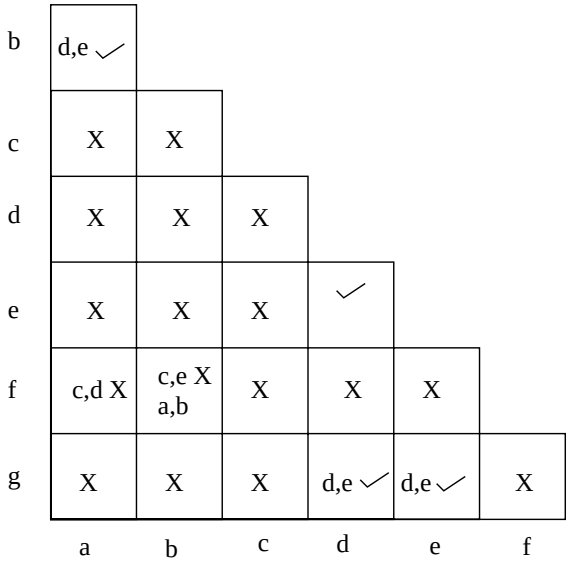
    (ii) transfers to the same (or equivalent state(s) for given input sequence

Example: Implication Table

- **State table to be reduced**

Present state	Next state		Output	
	X=0	X=1	X=0	X=1
a	d	b	0	0
b	e	a	0	0
c	g	f	0	1
d	a	d	1	0
e	a	d	1	0
f	c	b	0	0
g	a	e	1	0

Example: Implication Table



Example: Reduced State Table

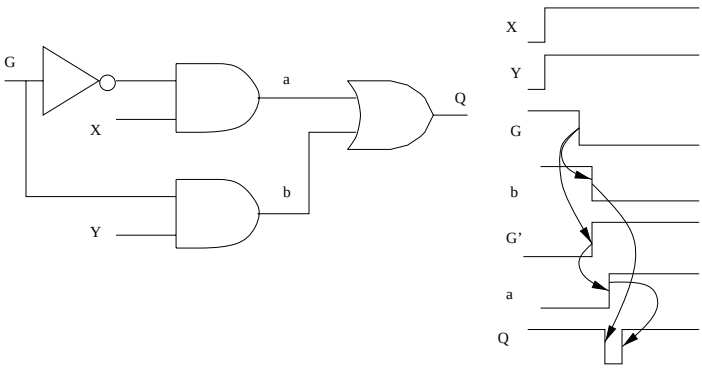
- **Reduced table**

Present state	Next state X=0	Next state X=1	Output X=0	Output X=1
a	d	a	0	0
c	d	f	0	1
d	a	d	1	0
f	c	a	0	0

- **The equivalent states are**
  - (a,b), (d,e), (d,g), (e,g)
- **Hence the reduced states are**
  - (a,b), c, (d,e,g), f

Hazards

- **Unwanted switching transients, “glitches”, that may cause circuit to malfunction**



  - Static 1 hazard

- **Glitches are inherent in a digital circuit and can not be eliminated totally**
  - Similarly static 0 hazard can be explained
- **Dynamic hazards**
  - the output changes multiple times instead of 0 -->1 or 1 --> 0 transition
- **Example of a dynamic hazard**