

CpE358/CS381

**Switching Theory and
Logical Design**

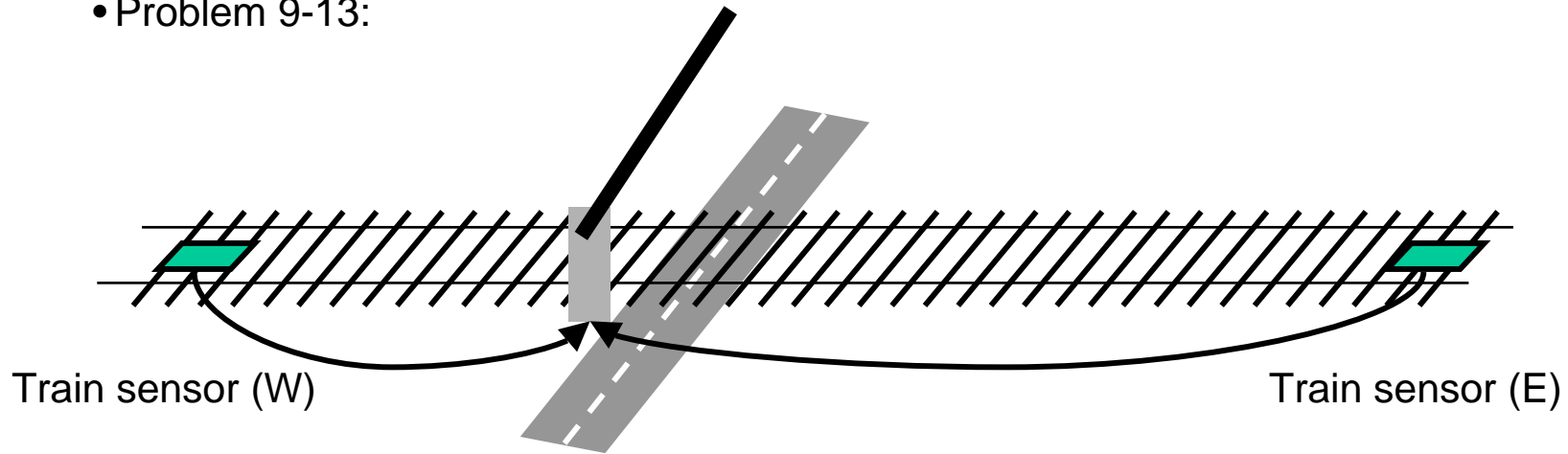
Class 14

Today

- Fundamental concepts of digital systems (Mano Chapter 1)
- Binary codes, number systems, and arithmetic (Ch 1)
- Boolean algebra (Ch 2)
- Simplification of switching equations (Ch 3)
- Digital device characteristics (e.g., TTL, CMOS)/design considerations (Ch 10)
- Combinatoric logical design including LSI implementation (Chapter 4)
- Flip-flops and state memory elements (Ch 5)
- Sequential logic analysis and design (Ch 5)
- Counters, shift register circuits (Ch 6)
- Hazards, Races, and time related issues in digital design (Ch 9)
- **Synchronous vs. asynchronous design (Ch 9)**
- Memory and Programmable logic (Ch 7)
- Minimization of sequential systems
- Introduction to Finite Automata

Asynchronous Design

- Problem 9-13:



Close gate when front of train passes first sensor,
Open gate when back of train passes second sensor.

Assume train length shorter than span between sensors
Assume only one train in region at a time
Assume single track

Asynchronous Design

- Define states and create a primitive flow table

State	State Name	Input E	Input W	Output	Comments
Idle	i	0	0	Open	No trains present – after state de2, dw2
EnterW	ew1	0	1	Closed	Eastbound train entering – after state i
EnteredW	ew2	0	0	Closed	Train present – after state ew1
DepartE	de1	1	0	Closed	Eastbound train departing – after state ew2
DepartedE	de2	0	0	Open	Eastbound train departed – after state de1
EnterE	ee1	1	0	Closed	Westbound train entering – after state i
EnteredE	ee2	0	0	Closed	Train present – after state ee1
DepartW	dw1	0	1	Closed	Westbound train departing – after state ee2
DepartedW	dw2	0	0	Open	Westbound train departed – after state dw1

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1		ee1,1
ew1	ew2,1	ew1,1		
ew2	ew2,1			de1,1
de1	de2,0			de1,1
de2	i,1			
ee1	ee2,1			ee1,1
ee2	ee2,1	dw1,1		
dw1	dw2,0	dw1,1		
dw2	i,1			

Output: 0=open
1=closed

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1		ee1,1
ew1	ew2,1	ew1,1		
ew2	ew2,1			de1,-
de1	de2,1			de1,1
de2	i,1			
ee1	ee2,1			ee1,1
ee2	ee2,1	dw1,1		
dw1	dw2,0	dw1,1		
dw2	i,1			

Output: 0=open
1=closed

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1		ee1,1
ew1	ew2,1	ew1,1		
ew2	ew2,1			de1,1
de1	i,1			de1,1
ee1	ee2,1			ee1,1
ee2	ee2,1	dw1,1		
dw1	i,1	dw1,1		

Output: 0=open
1=closed

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1	-, -	ee1,1
ew1	ew2,1	ew1,1	-, -	-, -
ew2	ew2,1	-, -	-, -	de1,1
de1	i,1	-, -	-, -	de1,1
ee1	ee2,1	-, -	-, -	ee1,1
ee2	ee2,1	dw1,1	-, -	-, -
dw1	i,1	dw1,1	-, -	-, -

Output: 0=open
1=closed

Mark unallowed transitions

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1	-, -	ee1,1
ew1	ew2,1	ew1,1	-, -	-, -
ew2	ew2,1	-, -	-, -	de1,1
de1	i,1	-, -	-, -	de1,1
ee1	ee2,1	-, -	-, -	ee1,1
ee2	ee2,1	dw1,1	-, -	-, -
dw1	i,1	dw1,1	-, -	-, -

Output: 0=open
1=closed

Identify candidates for merging

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1	-, -	ee1,1
ew1	ew2,1	ew1,1	-, -	-, -
ew2	ew2,1	-, -	-, -	de1,1
de1	i,1	-, -	-, -	de1,1
ee1	ee2,1	-, -	-, -	ee1,1
ee2	ee2,1	dw1,1	-, -	-, -
dw1	i,1	dw1,1	-, -	-, -

Output: 0=open
1=closed

Identify candidates for merging

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1	-, -	ee1,1
ew1	ew2,1	ew1,1	-, -	de1,1
ew2	ew2,1	,	,	de1,1
de1	i,1	dw1,1	-, -	de1,1
ee1	ee2,1	dw1,1	-, -	ee1,1
ee2	ee2,1	dw1,1	-, -	,
dw1	i,1	dw1,1	,	,

Output: 0=open
1=closed

Merge states

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1	-, -	ee1,1
ew1	ew1,1	ew1,1	-, -	de1,1
ew2	ew2,1	, ,	, ,	de1,1
de1	i,1	de1,1	-, -	de1,1
ee1	ee1,1	dw1,1	-, -	ee1,1
ee2	ee2,1	dw1,1	-, -	, ,
dw1	i,1	dw1,1	, ,	, ,

Output: 0=open
1=closed

Merge states,
Relabel transitions

Asynchronous Design

- Create primitive flow table

	E/W Sensor			
State	00	01	11	10
i	i,0	ew1,1	-, -	ee1,1
ew1	ew1,1	ew1,1	-, -	de1,1
de1	i,1	de1,1	-, -	de1,1
ee1	ee1,1	de,1	-, -	ee1,1

Output: 0=open
1=closed

Merge states,
Relabel transitions

Asynchronous Design

- Simplified flow table:

	E/W Sensor			
State	00	01	11	10
i	i,0	ew,1	-, -	ee,1
ew	ew,1	ew,1	-, -	d,1
d	i,1	d,1	-, -	d,1
ee	ee,1	d,1	-, -	ee,1

Output: 0=open
1=closed

Rename states

i=idle
ew=enter west
ee=enter east
d=depart

Asynchronous Design

- Stabilize all conditions and assign outputs:

	E/W Sensor			
State	00	01	11	10
i	i,0	ew,1	i,1	ee,1
ew	ew,1	ew,1	i,1	d,1
d	i,1	d,1	i,1	d,1
ee	ee,1	d,1	i,1	ee,1

Output: 0=open
1=closed

i=idle
ew=enter west
ee=enter east
d=depart

Asynchronous Design

- Assign state values:

	E/W Sensor			
State	00	01	11	10
00	00,0	ew,1	00,1	ee,1
ew	ew,1	ew,1	00,1	d,1
d	00,1	d,1	00,1	d,1
ee	ee,1	d,1	00,1	ee,1

Output: 0=open
1=closed

00=i=idle
ew=enter west
ee=enter east
d=depart

Asynchronous Design

- Assign state values:

	E/W Sensor			
State	00	01	11	10
00	00,0	01,1	00,1	ee,1
01	01,1	01,1	00,1	d,1
d	00,1	d,1	00,1	d,1
ee	ee,1	d,1	00,1	ee,1

Output: 0=open
1=closed

00=i=idle
01=ew=enter west
ee=enter east
d=depart

Asynchronous Design

- Assign state values:

	E/W Sensor			
State	00	01	11	10
00	00,0	01,1	00,1	10,1
01	01,1	01,1	00,1	d,1
d	00,1	d,1	00,1	d,1
10	10,1	d,1	00,1	10,1

Output: 0=open
1=closed

00=i=idle
01=ew=enter west
10=ee=enter east
d=depart

Asynchronous Design

- Assign state values:

	E/W Sensor			
State	00	01	11	10
00	00,0	01,1	00,1	10,1
01	01,1	01,1	00,1	11,1
11	00,1	11,1	00,1	11,1
10	10,1	11,1	00,1	10,1

Output: 0=open
1=closed

00=i=idle
01=ew=enter west
10=ee=enter east
11=d=depart

Asynchronous Design

- Create excitation and output maps:

	E/W Sensor			
State	00	01	11	10
00	00,0	01,1	00,1	10,1
01	01,1	01,1	00,1	11,1
11	00,1	11,1	00,1	11,1
10	10,1	11,1	00,1	10,1

	EW			
y_1y_2	00	01	11	10
00	0	0	0	1
01	0	0	0	1
11	0	1	0	1
10	1	1	0	1

Y_1

	EW			
y_1y_2	00	01	11	10
00	0	1	0	0
01	1	1	0	1
11	0	1	0	1
10	0	1	0	0

Y_2

	EW			
y_1y_2	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

out

Asynchronous Design

- Create excitation and output maps:

	EW			
y_1y_2	00	01	11	10
00	0	0	0	1
01	0	0	0	1
11	0	1	0	1
10	1	1	0	1

Y_1

	EW			
y_1y_2	00	01	11	10
00	0	1	0	0
01	1	1	0	1
11	0	1	0	1
10	0	1	0	0

Y_2

	EW			
y_1y_2	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

out

Asynchronous Design

- Create excitation and output maps:

	EW			
y_1y_2	00	01	11	10
00	0	0	0	1
01	0	0	0	1
11	0	1	0	1
10	1	1	0	1

Y_1

	EW			
y_1y_2	00	01	11	10
00	0	1	0	0
01	1	1	0	1
11	0	1	0	1
10	0	1	0	0

Y_2

	EW			
y_1y_2	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

out

$$\text{out} = (E'W'y_1'y_2)'$$

Asynchronous Design

- Create excitation and output maps:

	EW			
y_1y_2	00	01	11	10
00	0	0	0	1
01	0	0	0	1
11	0	1	0	1
10	1	1	0	1

Y_1

$$Y_1 = y_1y_2' + E'Wy_1 + E'y_1y_2'$$

	EW			
y_1y_2	00	01	11	10
00	0	1	0	0
01	1	1	0	1
11	0	1	0	1
10	0	1	0	0

Y_2

$$Y_2 = E'W + E'y_1'y_2 + EW'y_2$$

	EW			
y_1y_2	00	01	11	10
00	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

out

$$\text{out} = (E'W'y_1'y_2)'$$

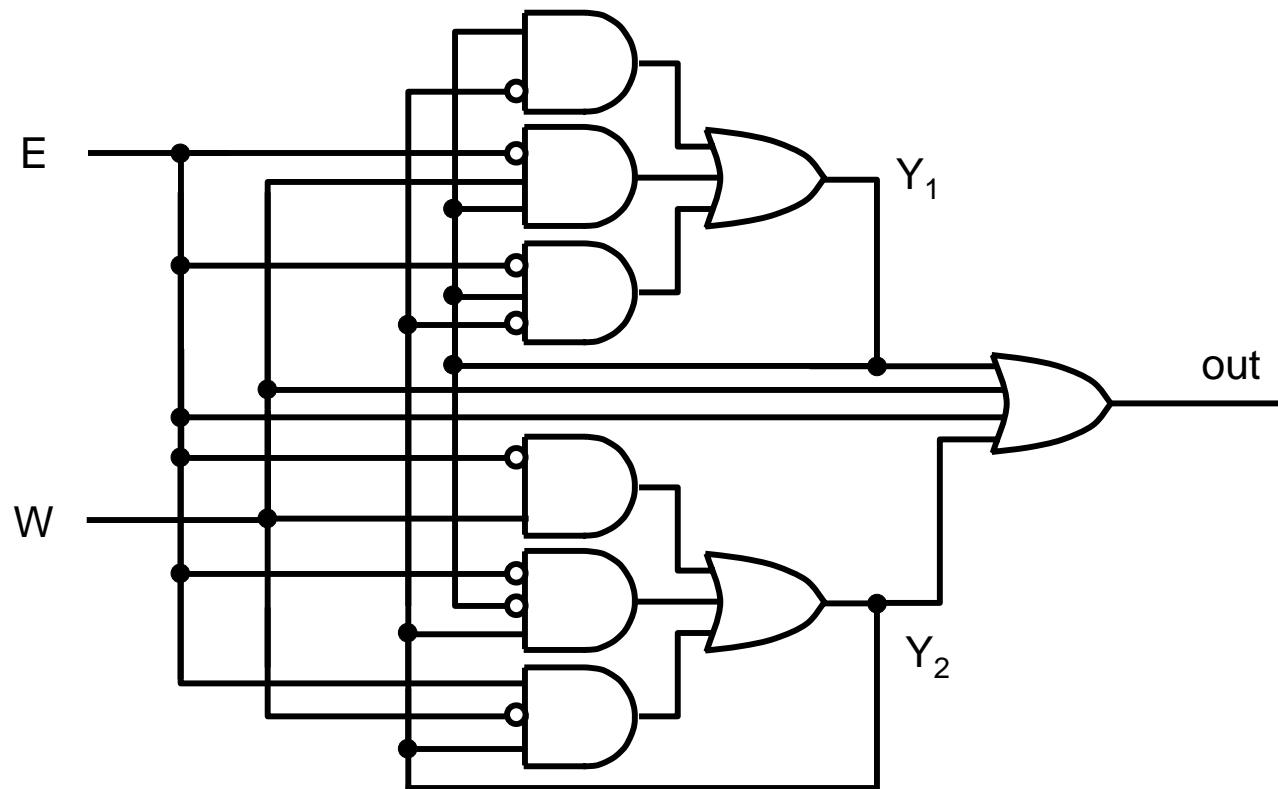
Asynchronous Design

- Create logic diagram:

$$Y_1 = y_1 y_2' + E' W y_1 + E' y_1 y_2'$$

$$Y_2 = E' W + E' y_1' y_2 + E W' y_2$$

$$\text{out} = E + W + y_1 + y_2$$



Summary

- Fundamental concepts of digital systems (Mano Chapter 1)
- Binary codes, number systems, and arithmetic (Ch 1)
- Boolean algebra (Ch 2)
- Simplification of switching equations (Ch 3)
- Digital device characteristics (e.g., TTL, CMOS)/design considerations (Ch 10)
- Combinatoric logical design including LSI implementation (Chapter 4)
- Flip-flops and state memory elements (Ch 5)
- Sequential logic analysis and design (Ch 5)
- Counters, shift register circuits (Ch 6)
- Hazards, Races, and time related issues in digital design (Ch 9)
- **Synchronous vs. asynchronous design (Ch 9)**
- Memory and Programmable logic (Ch 7)
- Minimization of sequential systems
- Introduction to Finite Automata

Homework 14 – due in Class 16

- Show all work
- Problems 9-14
- Verify the solution to problem 9-13 by analyzing the circuit and comparing resulting transition table to problem definition
- Redo problem 9-13 with a synchronous design. Assume that a clock exists with a 1 second period.