L08: Don't cares

ECE 2195/1170: Algorithms for Complex System Design and Modeling

Dr. Natasa Miskov-Zivanov
Department of Electrical and Computer Engineering
Swanson School of Engineering

Spring 2022

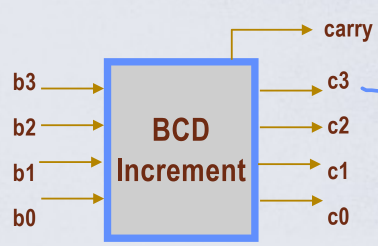
University of Pittsburgh

Some lecture material and figures courtesy of Rob Rutenbar

1

Don't Cares: 2-level

- In basic digital design...
  - We told you these were just input patterns that could **never** happen
  - This allowed you to do more simplifications, since you could add a 1 or 0 to the Kmap for that input depending on what was easier to simplify
  - Standard example: BCD incrementer circuit



b1b0 \ b3b2	00	01	11	10
00			d	1
01			d	
11		1	d	d
10			d	d

Patterns b3 b2 b1 b0 = 1010, 1011, 1100, 1101, 1110, 1111 cannot happen

University of Pittsburgh

3

Don’t Cares: Multi-level

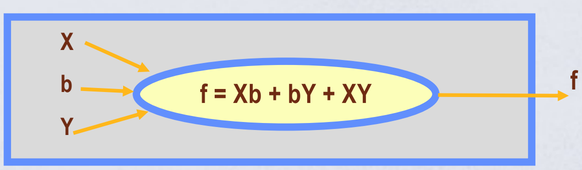
- To say this differently
  - In basic 2-level designs somebody told you what inputs wouldn’t happen...
  - ***...and you just believed them!***
- What’s different in multi-level?
  - Can still have these sorts of don’t cares at the primary inputs of the Boolean logic network....
  - ...but there can also be don’t cares **arising from structure** of the network
  - These latter kind are very useful for simplifying the individual vertices in the Boolean logic network
  - ***But, you have to go find these don’t cares explicitly***

University of Pittsburgh

4

Informal Tour of DCs in Multilevel Networks

- Suppose we have a Boolean network...
  - And we are looking at node **“f”** in that network
- Can we say anything about don’t cares for node f?
  - **NO**
  - We don’t know any “context” for surrounding parts of network
  - As far as we can tell, all patterns of inputs **(X,b,Y)** are possible



Y \ Xb	00	01	11	10
0			1	
1		1	1	1

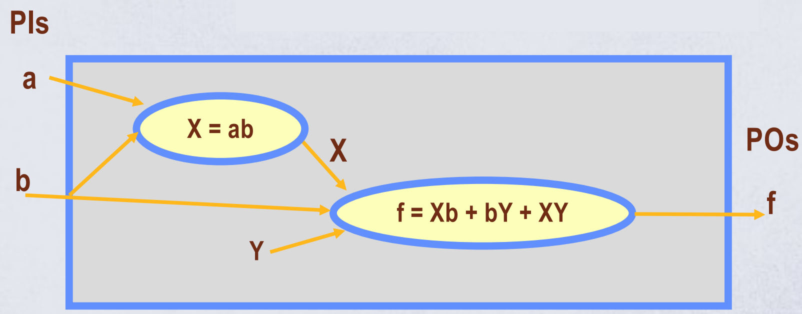
University of Pittsburgh

5

Informal Multilevel DC Tour

- OK, suppose we know this about input X to f
  - Node **X** is actually **a•b**
  - Now can we say something about **DCs** for node **f**...?
  - **YES**

B is input to both x and f



University of Pittsburgh

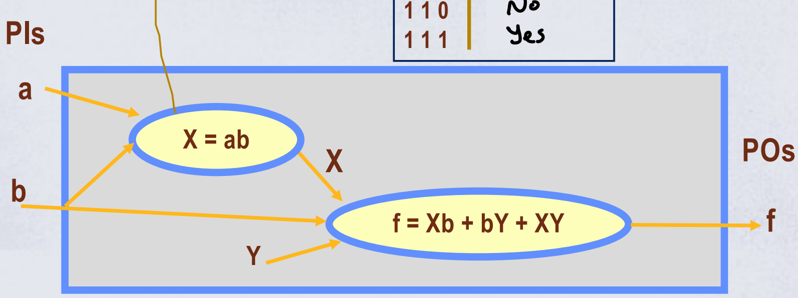
6

Are there any impossible patterns for X, b, and Y? → use DCs

Informal Multilevel DC Tour

- Go list all the input/output patterns for node X

a b X	can it occur?
0 0 0	yes
0 0 1	No
0 1 0	yes
0 1 1	No
1 0 0	yes
1 0 1	No
1 1 0	No
1 1 1	yes



University of Pittsburgh

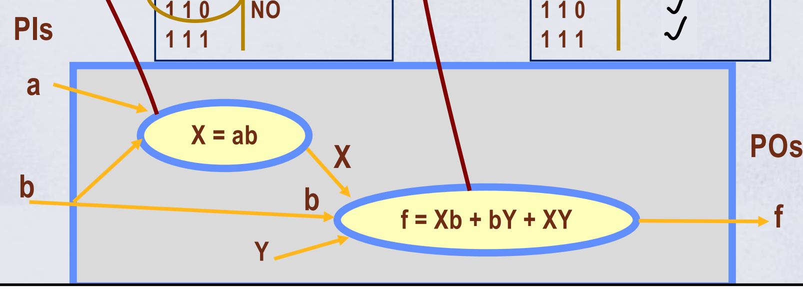
7

Informal Multilevel DC Tour

- Impossible a b X patterns → impossible X b Y patterns?

a b X	can it occur?
0 0 0	
0 0 1	NO
0 1 0	
0 1 1	NO
1 0 0	
1 0 1	NO
1 1 0	NO
1 1 1	

X bY	can it occur?
0 0 0	✓
0 0 1	✓
0 1 0	✓
0 1 1	✓
1 0 0	No
1 0 1	No
1 1 0	✓
1 1 1	✓



University of Pittsburgh

8

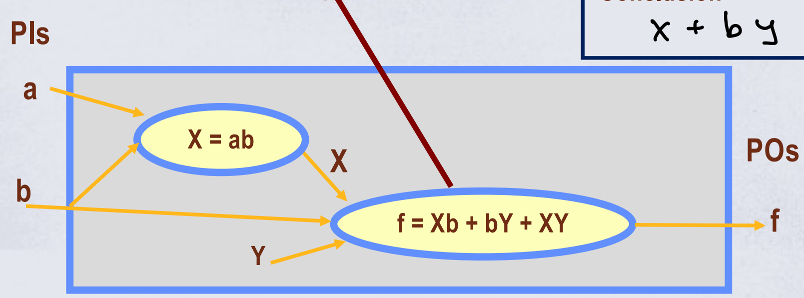
Informal Multilevel DC Tour

- Impossible X b Y patterns give us DCs for node f
  - Change how we would want to simplify node **f** (its Kmap)

Impossible patterns =
XbY = 100
101

Y \ Xb	00	01	11	10
0			1	d
1		1	1	d

Conclusion
X + bY

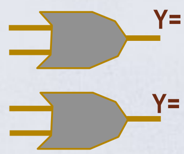


University of Pittsburgh

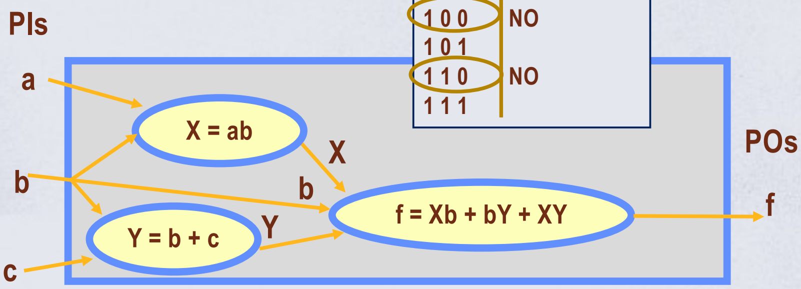
9

Informal Multilevel DC Tour

- OK, what if we now know Y = b+c as well?
  - Can do this again at **Y** ...are there impossible patterns of **b c Y** ?



b c Y	can it occur?
0 0 0	
0 0 1	NO
0 1 0	NO
0 1 1	
1 0 0	NO
1 0 1	
1 1 0	NO
1 1 1	



University of Pittsburgh

10

Informal Multilevel DC Tour

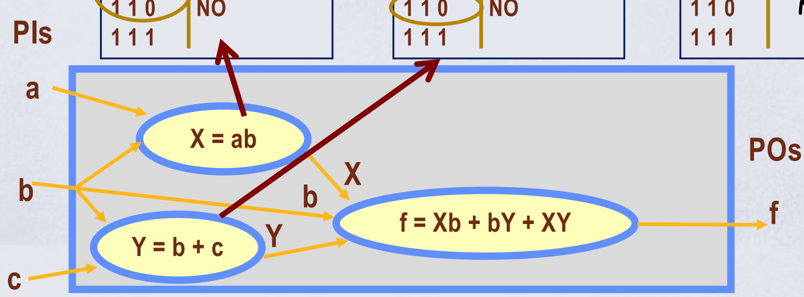
- OK, can we (again) get impossible patterns on X b Y?

y = b + c
x = a b

a b X	can it occur?
0 0 0	
0 0 1	NO
0 1 0	
0 1 1	NO
1 0 0	
1 0 1	NO
1 1 0	NO
1 1 1	

b c Y	can it occur?
0 0 0	
0 0 1	NO
0 1 0	NO
0 1 1	
1 0 0	NO
1 0 1	
1 1 0	NO
1 1 1	

X bY	can it occur?
0 0 0	
0 0 1	
0 1 0	No
0 1 1	
1 0 0	No
1 0 1	No
1 1 0	No
1 1 1	



University of Pittsburgh

11

Informal Multilevel DC Tour

- OK, do these change how we'd simplify inside f?

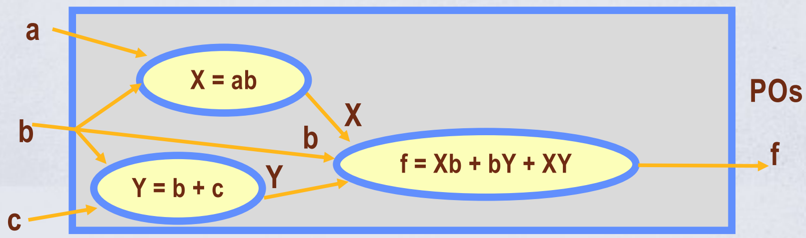
X bY	can it occur?
0 0 0	
0 0 1	
0 1 0	NO
0 1 1	
1 0 0	NO
1 0 1	NO
1 1 0	NO
1 1 1	

Y \ Xb	00	01	11	10
0		d	d	d
1		1	1	d

Conclusion
f = b

Primary inputs PIs

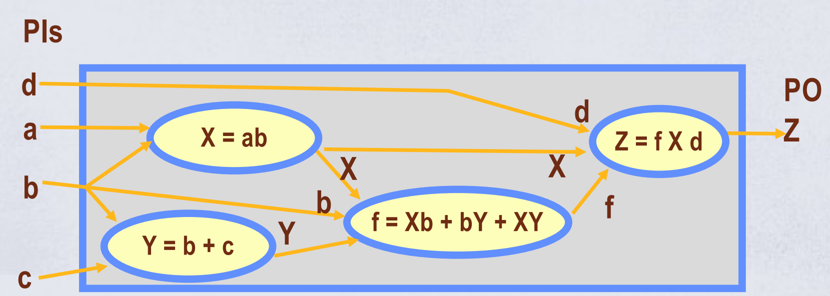
Primary outputs POs



12

Informal Multilevel DC Tour

- OK, now suppose f is *not* a primary output, Z is...
  - **Question**: when does a change in the output of node **f** actually **propagate** through to change the primary output **Z**, i.e., the output of the overall Boolean logic network
  - Or, reverse question: when does it ***not matter*** what **f** is...?
  - Let's go look at patterns of **f X d** at node **Z**...



13

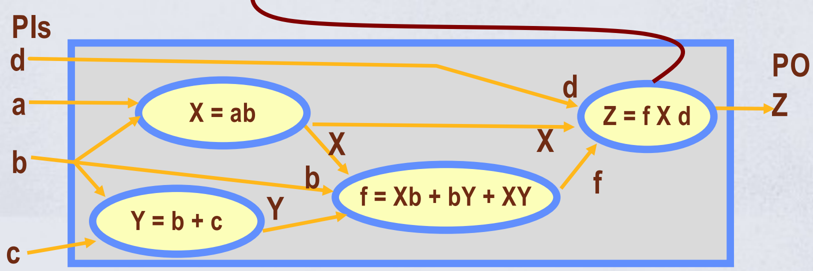
Informal Multilevel DC Tour

	f X d	Z does it change?
Δf	0 0 0 1 0 0	No, z = 0
Δf	0 0 1 1 0 1	No, z = 0
Δf	0 1 0 1 1 0	No, z = 0
Δf	0 1 1 1 1 1	Yes, z = f

Patterns at input to f node itself that are DCs just because those patterns make Z output *insensitive* to changes in f

X b Y

= 0 - -



University of Pittsburgh

14

Informal Multilevel DC Tour

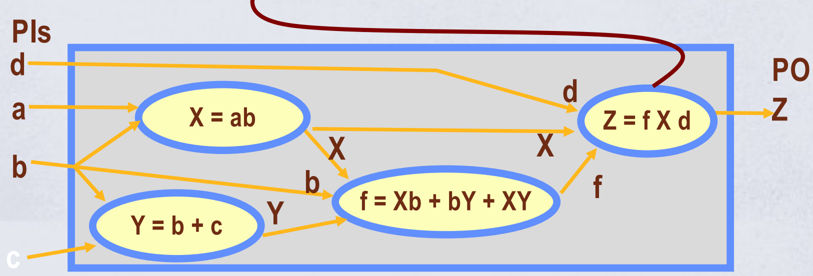
- OK, can we use this X=0 DC pattern to simplify f more?

Patterns at input to f node itself that are DCs just because those patterns make Z output *insensitive* to changes in f

=> XbY = 0 - -

Y \ Xb	00	01	11	10
0	d	d	d	d
1	d	d	1	d

Conclusion
f = 1

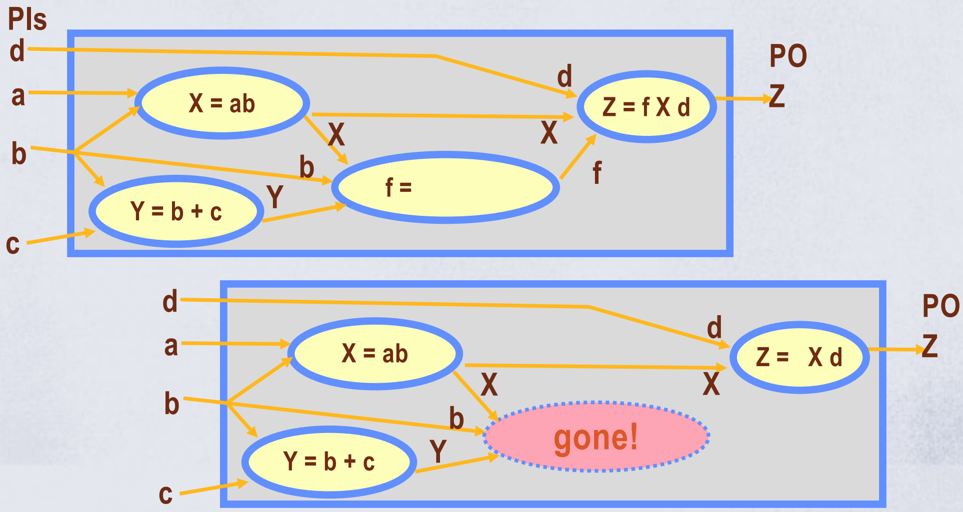


University of Pittsburgh

15

Informal Multilevel DC Tour

- Hey, look what happened to f node...
  - ***Due to context of surrounding nodes, it disappeared!***

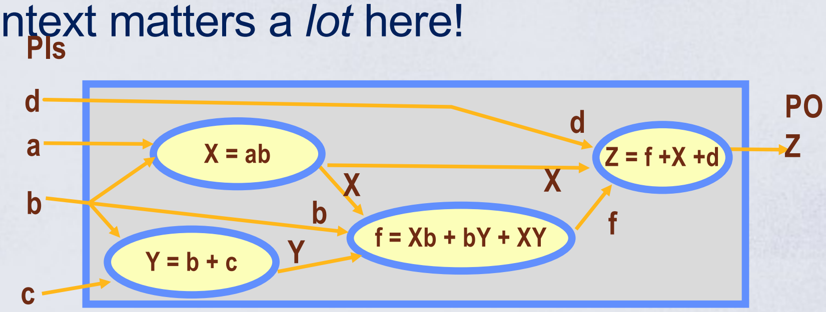


University of Pittsburgh

16

Informal Multilevel DC Tour

- OK, suppose instead that PO Z = f + X + d (OR not AND)
  - What changes?
  - **Answer**: patterns at **f** inputs that make **Z** insensitive to changes in **f**
  - There are still impossible patterns of **(f X d)** but you cannot specify any of them exactly only knowing the **(X b Y)** inputs to **f**
  - **f** doesn't disappear, it simplifies to **f = b** (***go check…***)
- Network context matters a *lot* here!



University of Pittsburgh

17

Formal View of These DCs

- Overall, there are 3 types of formal DCs...
  - **Satisfiability** don't cares
    - Patterns that can't occur on (inputs, output) of one network node...
    - ... because of internal structure of multi-level logic
  - **Controllability** don't cares
    - Global, external: patterns that can't happen at primary inputs to our overall Boolean network (these are the DCs you already knew about)
    - Local, internal: patterns that can't happen at inputs to a network vertex
  - **Observability** don't cares
    - Patterns at input of a vertex that prevent outputs of the network from being sensitive to changes in output of that vertex
    - Patterns that "mask" outputs
- Next lecture: See if we can clarify where these each come from...

University of Pittsburgh

18

ECE 2195/1170: Next lecture

- More on DCs

University of Pittsburgh