CSEE 3827: Fundamentals of Computer Systems, Spring 2021

Lecture 2

Prof. Dan Rubenstein (danr@cs.columbia.edu)

Course Announcements

- Please remember to fill in your P-credit office hour preferences by today (Tues 1/25) @ http://uribe.cs.columbia.edu/sched/table.php
- Please fill in the Google Form indicating attendance prefs:
  - https://forms.gle/aaYJ6paoTnnheHdr7
- See your assigned P-credit OH:
  - sorted by UNI: http://uribe.cs.columbia.edu/sched/assigned.html
  - sorted by name: http://uribe.cs.columbia.edu/sched/assigned-nameorder.html
  - sorted by OH date/time: http://uribe.cs.columbia.edu/sched/assigned-slotorder.html
  -

Agenda (M&K 2.1-2.2)

- Terminology
- Boolean algebra (incl. DeMorgan's Law)
- Logic gates
- Circuit fabrication
  - NAND, NOR
- DUAL
- XOR

Boolean Terminology

Terminology

- Recall: Digital / Binary / Boolean: 0 = False, 1 = True

- Binary or Boolean Variable: a symbolic representation of a value that might be 0 or 1, e.g., X, Y, A, B

- Complement (e.g., of a variable X): written $\overline{X}$: the opposite value of X

X	$\overline{X}$
0	1
1	0

I sometimes say “negate”: , e.g.,
if boolean X=1, negating X makes X=0.
Negating X again changes it back to 1

- Literal: a boolean variable or its complement (e.g., X, $\overline{X}$, $\overline{Y}$)

Boolean Logic

- All logical functions can be implemented in terms of three logical operations:

NOT

x	$\overline{x}$
0	1
1	0

AND

can omit the "·"

x	y	x·y
0	0	0
0	1	0
1	0	0
1	1	1

OR

Not addition anymore!

x	y	x + y
0	0	0
0	1	1
1	0	1
1	1	1

- Whether "+" represents addition or logical OR can often be gotten from context
- When not from context, we will specify

Boolean Logic 2

- Boolean logic precedence rules just like addition / multiplication
- Implied precedence: NOT before AND before OR
- Use parentheses as necessary

$A+BC$ same as $A+ (BC)$

$(\overline{A} + B)C$ same as $((\overline{A}) + B)C$

Terminology cont'd

- Expression: a set of literals (possibly with repeats) combined with logic operations (and possibly ordered by parentheses)
  - e.g., 4 expressions: $AB + C$, $(AB) + C$, $(\overline{A} + B)C$, $((\overline{A}) + B)C$
- Equation: expression1 = expression2
  - e.g., $(\overline{A} + B)C = ((\overline{A}) + B)C$
- Function of (possibly several) variables: an equation where the lefthand side is defined by the righthand side

$$F(A,B,C) = ((\overline{A}) + B)C$$

XOR: the parity operation

- $X \oplus Y = X\bar{Y} + \bar{X}Y$

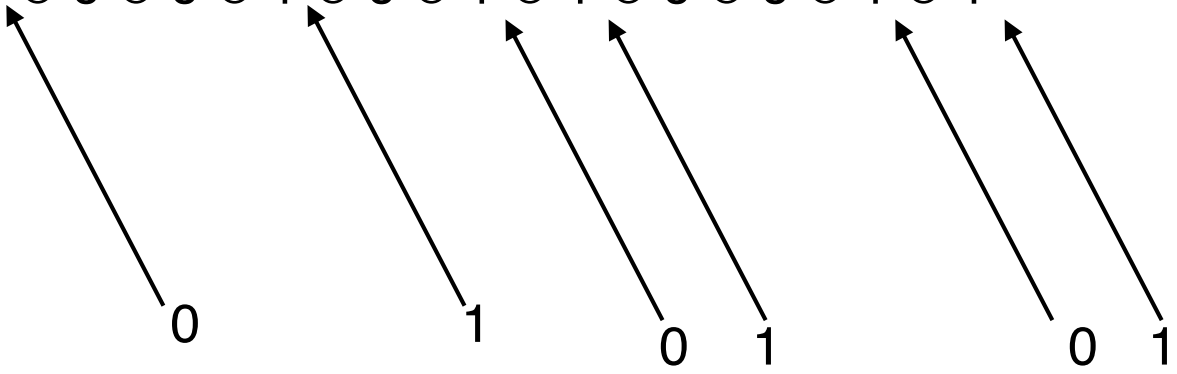
X	Y	X ⊕ Y
0	0	0
0	1	1
1	0	1
1	1	0

$X \oplus Y$ true when
X is true or Y is true,
but not both true

- In general, represents parity, i.e.,

- $X_1 \oplus X_2 \oplus X_3 \oplus \ldots \oplus X_k = 1$ when an odd number of $X_i = 1$

XOR as a “control”

- $Y \oplus 0 = Y$
- $Y \oplus 1 = \overline{Y}$
- As we XOR bits, the result “flips” when XOR’ing a 1, and stays the same when XOR’ing a 0
- $1 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 \oplus 1 \oplus 1 \oplus 0 \oplus 0 \oplus 1 \oplus 1 = 1$
- 

X	Y	X ⊕ Y
0	0	0
0	1	1
1	0	1
1	1	0

Boolean Logic: Truth Table

D	X	A	$L=D\overline{X} + A$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Truth Table: Function output for all combinations of input variables

k variables ➜ 2^k input combinations

e.g., $L = D\overline{X} + A$

Boolean Logic: Example

Intermediate computations to compute function:

D	X	A	$\overline{X}$	$\overline{DX}$	$L=\overline{DX} + A$
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	0	0
0	1	1	0	0	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	0	0	0
1	1	1	0	0	1

(M&K Table 2-2)

Boolean Logic: Example 2

Truth table for $XY + \overline{\overline{X}\,\overline{Y}}$

X	Y	$XY + \overline{\overline{X}\,\overline{Y}}$
0	0	
0	1	
1	0	
1	1	

Boolean Logic: Example 2

Truth table for $XY + \overline{X}\,\overline{Y}$: intermediate computations

X	Y	$\overline{X}$	$\overline{Y}$	XY	$\overline{XY}$	$XY + \overline{X}\,\overline{Y}$
0	0	1	1	0	1	1
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	1	0	0	1	0	1

Boolean Theorems

Boolean Algebra: Identities and Theorems

For any boolean X ($X \in \{0,1\}$):

OR	AND	NOT	
$X+0 = X$	$X1 = X$		(identity)
$X+1 = 1$	$X0 = 0$		(null)
$X+X = X$	$XX = X$		(idempotent)
$X+\overline{X} = 1$	$X\overline{X} = 0$		(complementarity)
		$\overline{\overline{X}} = X$	(involution)
$X+Y = Y+X$	$XY = YX$		(commutativity)
$X+(Y+Z) = (X+Y)+Z$	$X(YZ) = (XY)Z$		(associativity)
$X(Y+Z) = XY + XZ$	$X+YZ = (X+Y)(X+Z)$		(distributive)
$\overline{X+Y} = \overline{X}\,\overline{Y}$	$\overline{XY} = \overline{X} + \overline{Y}$		(DeMorgan's theorem)

Other important simplifications:

- $X + XY = X$:
  - If X is false, both sides clearly false
  - If X is true, LHS is true no matter what Y is

e.g.,
X = “You are male”
Y = “You are taller than 5’5”

- $X (X+Y) = X$:
  - If 1st term is false, doesn’t matter what second term is
  - If 1st term true, so is 2nd term

Boolean Algebra: Example

Simplify (reduce # of literals) this equation using algebraic manipulation.

$$F = \overline{X}YZ + \overline{X}Y\overline{Z} + XZ$$

Boolean Algebra: Example

Simplify this equation using algebraic manipulation.

$$F = \overline{X}YZ + \overline{X}Y\overline{Z} + XZ$$

$\overline{X}Y(Z + \overline{Z}) + XZ$ (by reverse distribution)

$\overline{X}Y(1) + XZ$ (by complementarity)

$\overline{X}Y + XZ$ (by identity)

Complementing (parts of) expressions (“the big bar”)

- Complement operations can be applied to expressions (big bars)

- Precedence: the complement is performed after all other boolean ops that lie beneath it

  - e.g., $\overline{A + B}$: First, OR A with B, then complement (negate)

A	B	A+B	$\overline{A+B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Complementing (parts of) expressions ("the big bar")

- Order of precedence example:

$$\overline{\overline{(A + BD)}}C$$

1. Complement A. (In parallel) AND B with D

2. OR $\overline{A}$ with BD

3. Complement $\overline{(\overline{A} + BD)}$

4. AND $\overline{\overline{(\overline{A} + BD)}}$ with C

5. Complement $\overline{\overline{(\overline{A} + BD)}\ C}$

DeMorgan's Theorem

Important!!!!!

DeMorgan's Theorem

- Procedure for simplifying complemented expressions (i.e., removal of "big bar"
- Remove the "big bar" over AND or OR of 2 (or more) functions or variables (e.g., F & G) and replace...
  - AND with OR, OR with AND
  - 1 with 0, 0 with 1
  - function F with $\overline{F}$, $\overline{F}$ with F

$$\overline{FG} = \overline{F} + \overline{G}$$

$$\overline{F + G} = \overline{F}\,\overline{G}$$

Understanding DeMorgan

- Example with boolean variables F & G:
  - F = It will snow (on Saturday)
  - G = Columbia will resume (in-person next week)

$\overline{FG}$: It's not the case that [It will snow AND Columbia will resume]

$\overline{F}+\overline{G}$: It won't snow OR Columbia won't resumse

$$\overline{FG} = \overline{F} + \overline{G}$$

$\overline{F+G}$: It's not the case that [It will snow OR Columbia will resume (or both)]

$\overline{F}\overline{G}$: It won't snow AND Columbia won't resume

$$\overline{F + G} = \overline{F}\,\overline{G}$$

DeMorgan's Theorem extended

- Can do for 3 functions (or variables), i.e.,

$$\overline{FGH} = \overline{F} + \overline{G} + \overline{H}$$

$$\overline{F + G + H} = \overline{F}\,\overline{G}\,\overline{H}$$

- Can apply to complementing the AND of any N functions (or variables), or complementing the OR of any N functions (or variables)

Boolean Algebra: Example 2

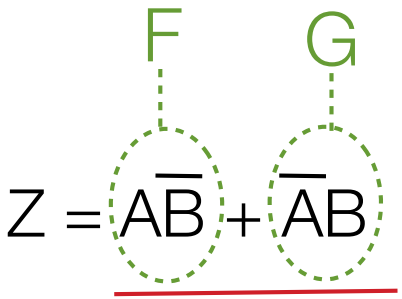
Find the complement of Z.

$$Z = A\overline{B} + \overline{A}B$$

$$\overline{Z} =$$

Boolean Algebra: Example 2

Find the complement of Z.



$$\overline{Z} = \overline{A\overline{B} + \overline{A}B}$$

$$\overline{(A\overline{B})}\ \overline{(\overline{A}B)} \qquad \text{(by DeMorgan's)}$$

$$(\overline{A} + \overline{\overline{B}})\ (\overline{\overline{A}} + \overline{B}) \qquad \text{(by DeMorgan's)}$$

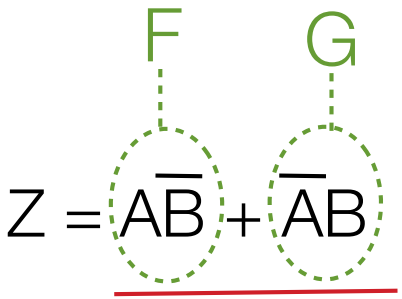
$$(\overline{A} + B)\ (A + \overline{B}) \qquad \text{(by involution)}$$

$$\overline{A}A + \overline{A}\,\overline{B} + BA + B\overline{B} \qquad \text{(by commutativity)}$$

$$\overline{A}\,\overline{B} + BA \qquad \text{(by complementarity and identity)}$$

Boolean Algebra: Example 2

Find the complement of Z.



$$\overline{Z} = \overline{A\overline{B} + \overline{A}B}$$

$$\overline{(A\overline{B})}\ \overline{(\overline{A}B)} \qquad \text{(by DeMorgan's)}$$

$$(\overline{A} + \overline{\overline{B}})\ (\overline{\overline{A}} + \overline{B}) \qquad \text{(by DeMorgan's)}$$

$$(\overline{A} + B)\ (A + \overline{B}) \qquad \text{(by involution)}$$

$$\overline{A}A + \overline{A}\,\overline{B} + BA + B\overline{B} \qquad \text{(by commutativity)}$$

$$\overline{A}\,\overline{B} + BA \qquad \text{(by complementarity and identity)}$$

DeMorgan's Practice

$$\overline{\overline{A\bar{B}C} + \overline{ACD} + B\bar{C}} \longrightarrow F_1 = \overline{A\bar{B}C},\ G_1 = \overline{ACD},\ H_1 = B\bar{C},\ \overline{F_1+G_1+H_1} = \bar{F}_1\,\bar{G}_1\,\bar{H}_1$$

$$= (A\bar{B}C)(ACD)(\overline{B\bar{C}}) \longrightarrow \text{For } \overline{B\bar{C}}: \quad F_2 = B,\ G_2 = \bar{C},\ \overline{F_2G_2} = \bar{F}_2+\bar{G}_2$$

Also, (ABC) (ACD) = ABCD

$$= (A\bar{B}CD)(\bar{B}+C)$$

$$= A\bar{B}CD + A\bar{B}CD$$

$$= A\bar{B}CD$$

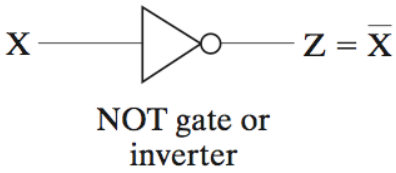
Consensus Theorem

- $XY + \overline{X}Z + YZ = XY + \overline{X}Z$ (The YZ term is redundant)
- Proof:
  - If YZ=0, it's obviously not needed (just have $XY+\overline{X}Z+0$ on LHS)
  - If YZ = 1
    - both Y and Z are 1. $XY + \overline{X}Z + 1 = 1$ so , LHS equation = 1
    - Either X or $\overline{X} = 1$. Since Y=Z=1, either XY=1 or $\overline{X}Z = 1$,
    - so RHS equation either 1 + 0 or 0 + 1 = 1

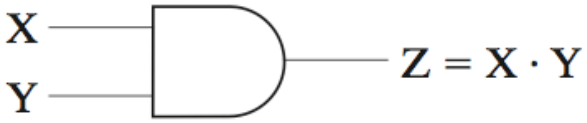
Circuit Perspective

Circuit Representation of Boolean Ops

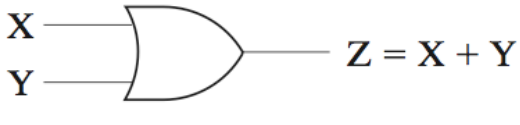
- Information flows from left to right
- Input(s) all the way on the left, output(s) on the right



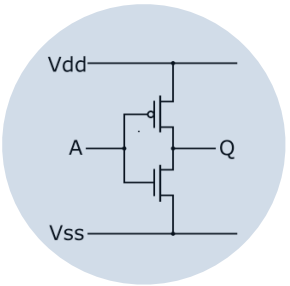
NOT gate or inverter

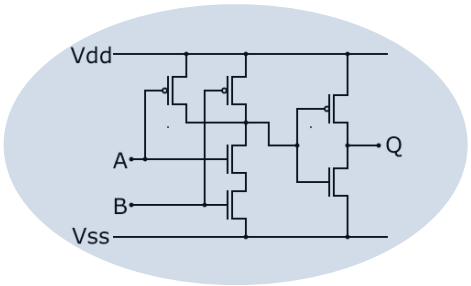


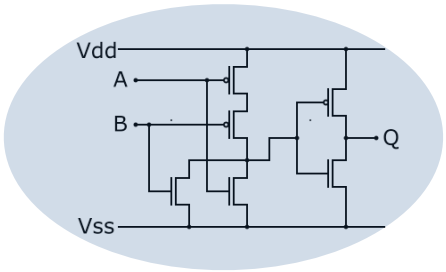
AND gate



OR gate





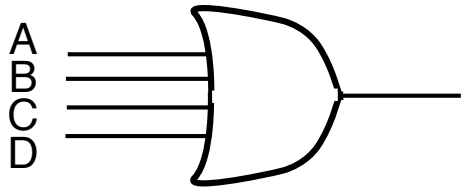


These circuits consume area, power, and time

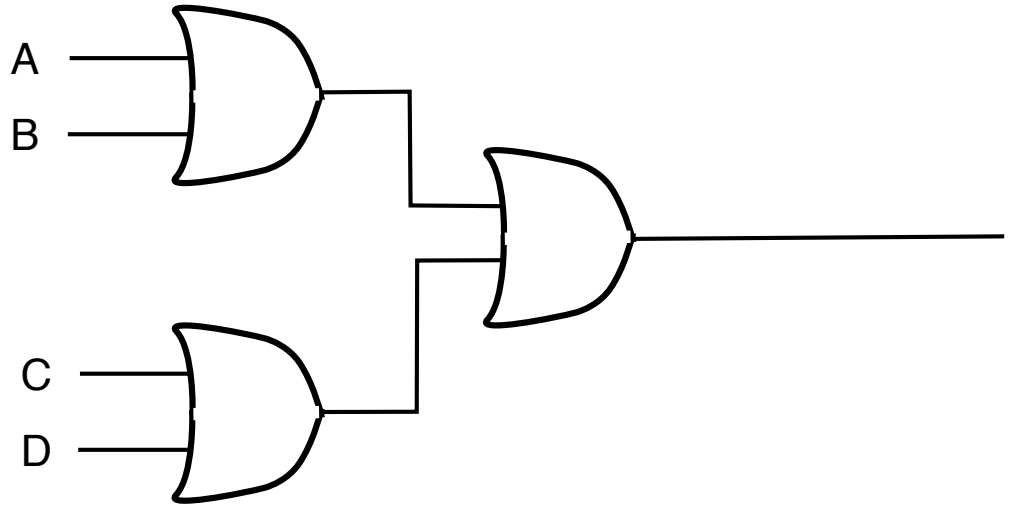
Goal: minimize the amount of circuitry to compute the desired function

more-than-2 input shorthand

- A+B+C+D can be written in shorthand as the gate:



- This is really a series of 2-input gates:



- i.e., a k-input gate is really a sequence of gates with depth $\log_2 k$

We simplify to reduce required circuitry...

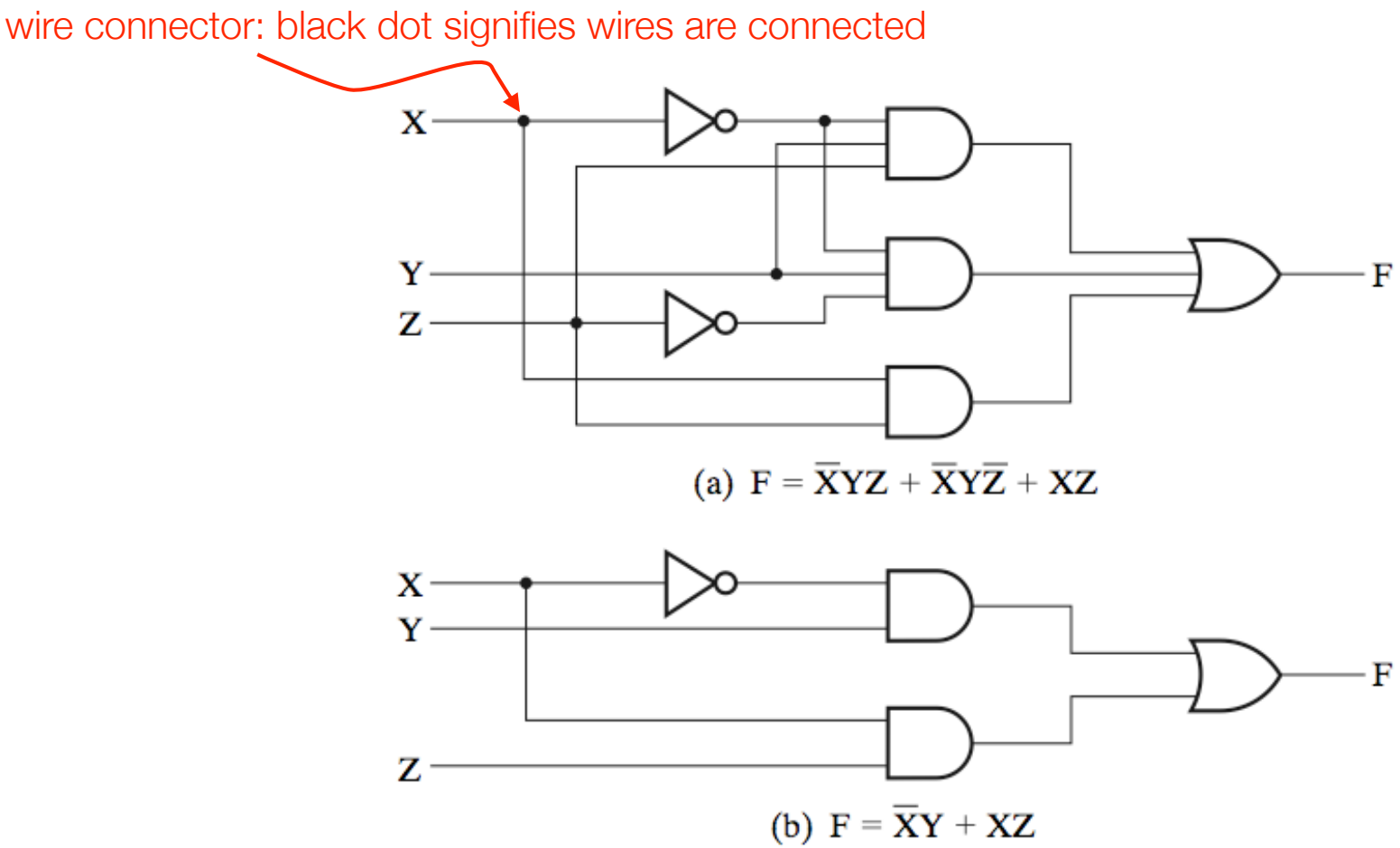
$F = \overline{X}YZ + \overline{X}Y\overline{Z} + XZ$

$\overline{X}Y(Z + \overline{Z}) + XZ$ (by reverse distribution)

$\overline{X}Y1 + XZ$ (by complementarity)

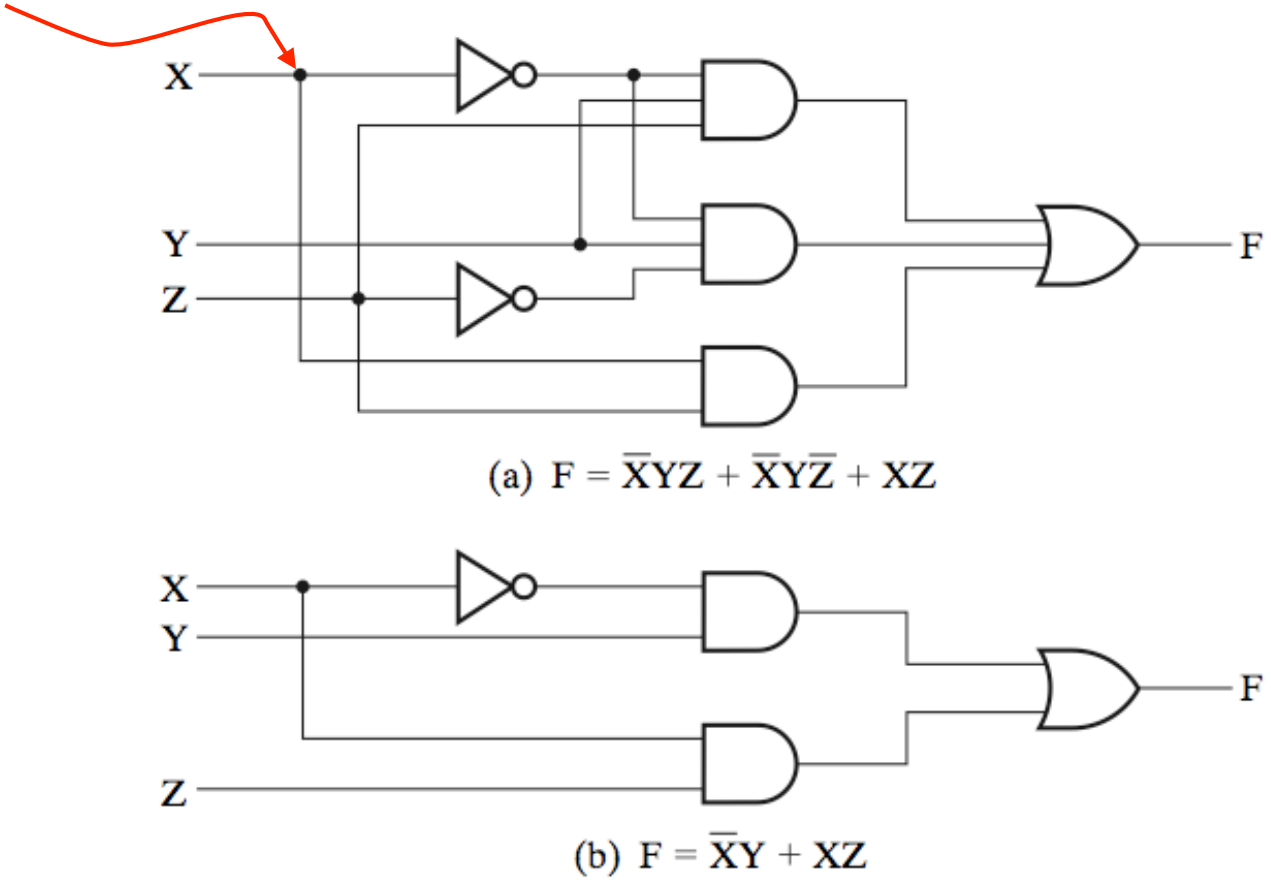
$\overline{X}Y + XZ$ (by identity)

Circuit view



Circuit view: why reduce?

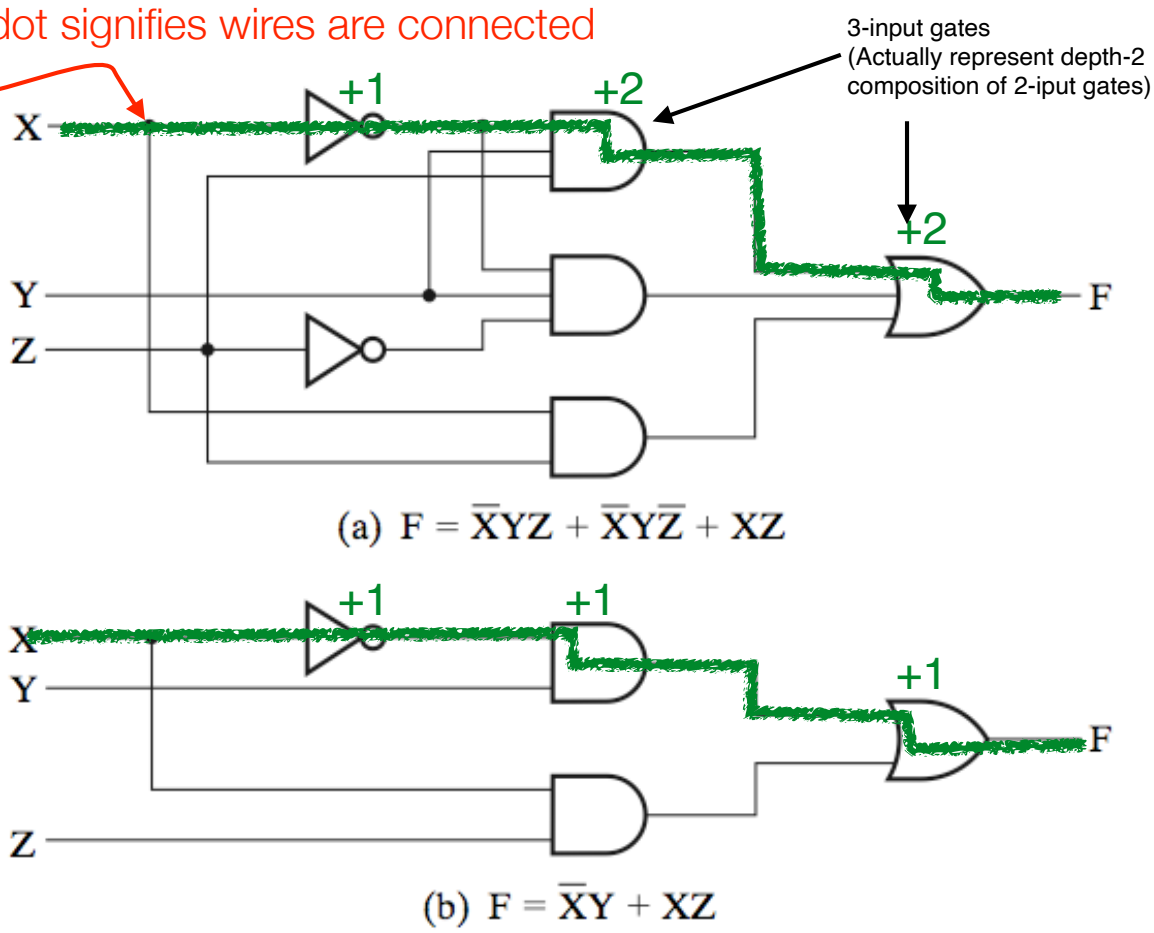
- Cost:
  - pay per gate: higher cost
  - more gates = more space on chip
- Wires are fast, gates are slow: speed of a circuit (time from setting inputs to getting output) roughly proportional to circuit depth: longest-gated path



Top circuit more costly (more gates), and faster when considering 3-input gates each have depth 2

Circuit view: why reduce?

- Cost:
  - pay per gate: higher cost
  - more gates = more space on chip
- Wires are fast, gates are slow: speed of a circuit (time from setting inputs to getting output) roughly proportional to circuit depth: longest-gated path

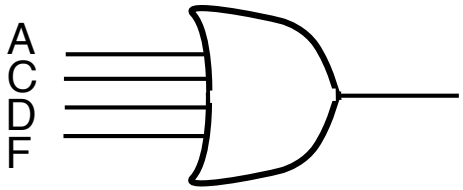


Top circuit more costly (more gates), and faster when considering 3-input gates each have depth 2

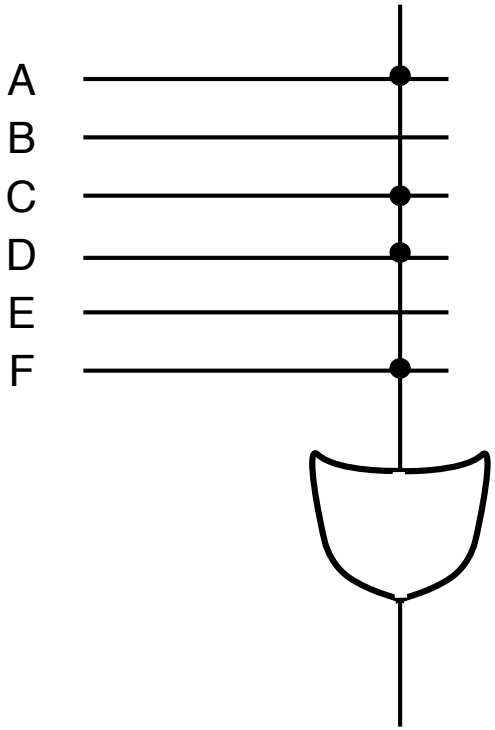
Top circuit depth 5 along top path, bottom circuit depth 3 along top path (longest paths)

Alternate shorthand for multiple inputs to Gate

- Given inputs A,B,C,D,E,F,



- Another representation for A+C+D+F This is really a series of 2-input gates:



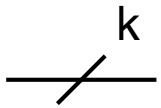
black dot signifies connection:
no dot means no connection

Bitwise parallel operations

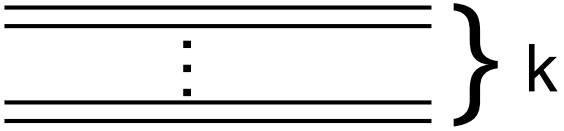
Bit-wise (in parallel) Logic Ops

- Up to now, we looked at single-bit logic ops, e.g., X + Y = Z where X,Y,Z are each one bit
- Will often need to perform logic op on set of bits in parallel, e.g.,
  - $X = X_{k-1}X_{k-2}\ldots X_1X_0$, $Y=Y_{k-1}Y_{k-2}\ldots Y_1Y_0$
  - Want to compute $Z=Z_{k-1}Z_{k-2}\ldots Z_1Z_0$ where for each i, $Z_i = X_i + Y_i$
  - Can just write Z = X+Y as a shorthand (here + is logical OR)
  - e.g., X = 0110, Y = 0011, Z = X+Y = 0110 + 0011 = 0111 (bitwise logical OR)
  - e.g., X=1010, Y = 1100, Z = XY = 0110 · 0011 = 0010 (bitwise logical AND)

Bitwise logical ops with circuitry

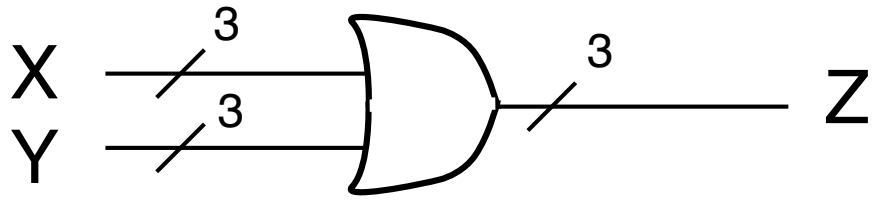


= k wires running in parallel, same as

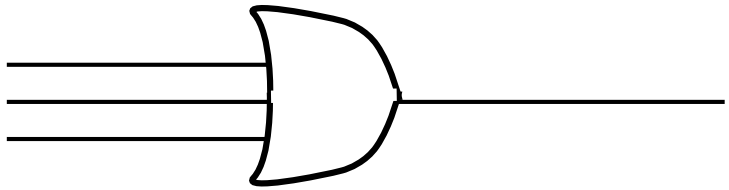


- extends to circuits as well, e.g., k=3

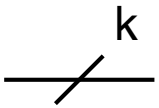
- X = 101, Y=110, X+Y = 111



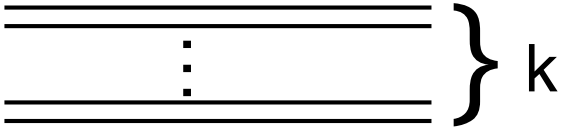
- Note the difference between above and which ORs 3 bits together



Bitwise logical ops with circuitry

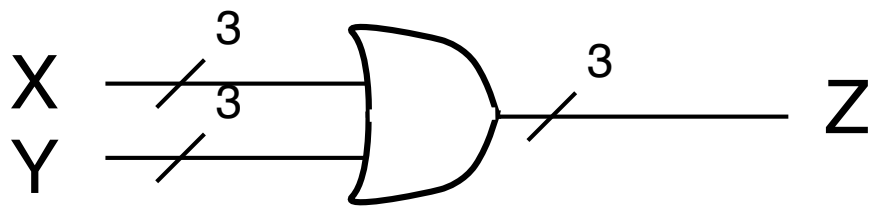


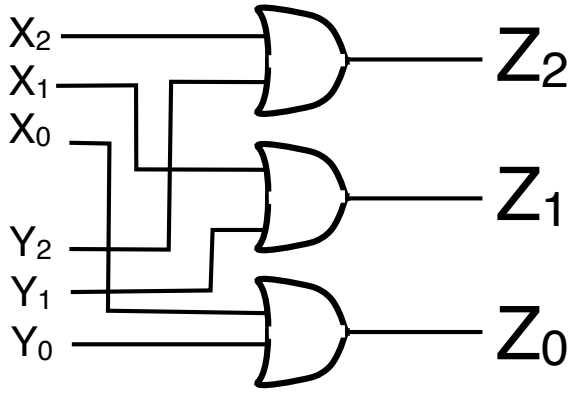
= k wires running in parallel, same as



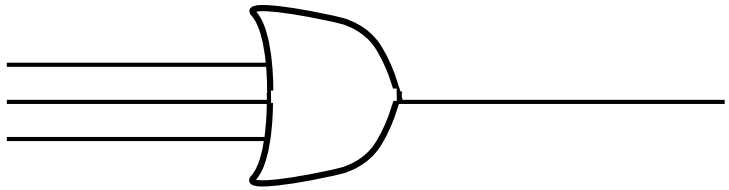
- extends to circuits as well, e.g., k=3

- X = 101, Y=110, X+Y = 111





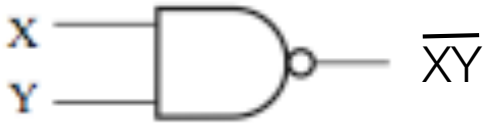
- Note the difference between above and which ORs 3 bits together



NAND and NOR: Universal gates

Universal gates: NAND, NOR

x	y	$z = \overline{xy}$
0	0	1
0	1	1
1	0	1
1	1	0



Note: the "o" in a circuit represents a NOT (inverter)

x	y	$z = \overline{x+y}$
0	0	1
0	1	0
1	0	0
1	1	0

Different from "●" which represents wire connector

NAND and NOR universal because...

- NOT, AND, OR can each be implemented using only NAND gates
- NOT, AND, OR can each be implemented using only NOR gates

$\overline{A} = A \text{ NAND } A$	$\overline{A} = A \text{ NOR } A$
$AB = \overline{A \text{ NAND } B}$	$A+B = \overline{A \text{ NOR } B}$
$A+B = \overline{A} \text{ NAND } \overline{B}$	$AB = \overline{A} \text{ NOR } \overline{B}$

Any circuit can be entirely implemented with just NAND or just NOR gates!!!

Duals

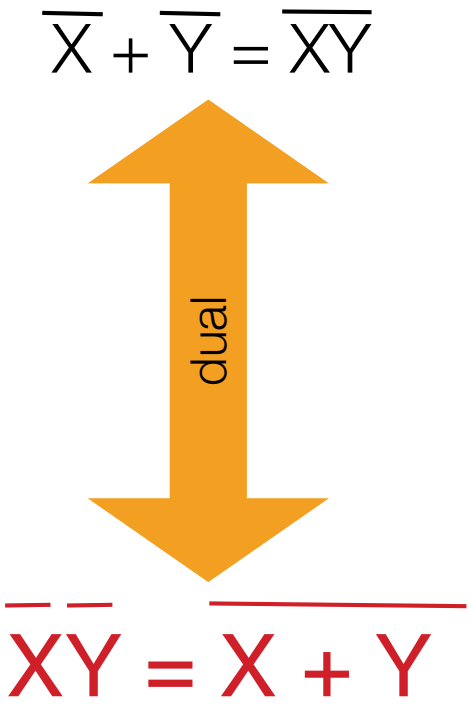
Duals

- All boolean expressions have duals
- Any theorem you can prove, you can also prove for its dual
- To form a dual...
  - replace AND with OR, OR with AND
  - replace 1 with 0, 0 with 1

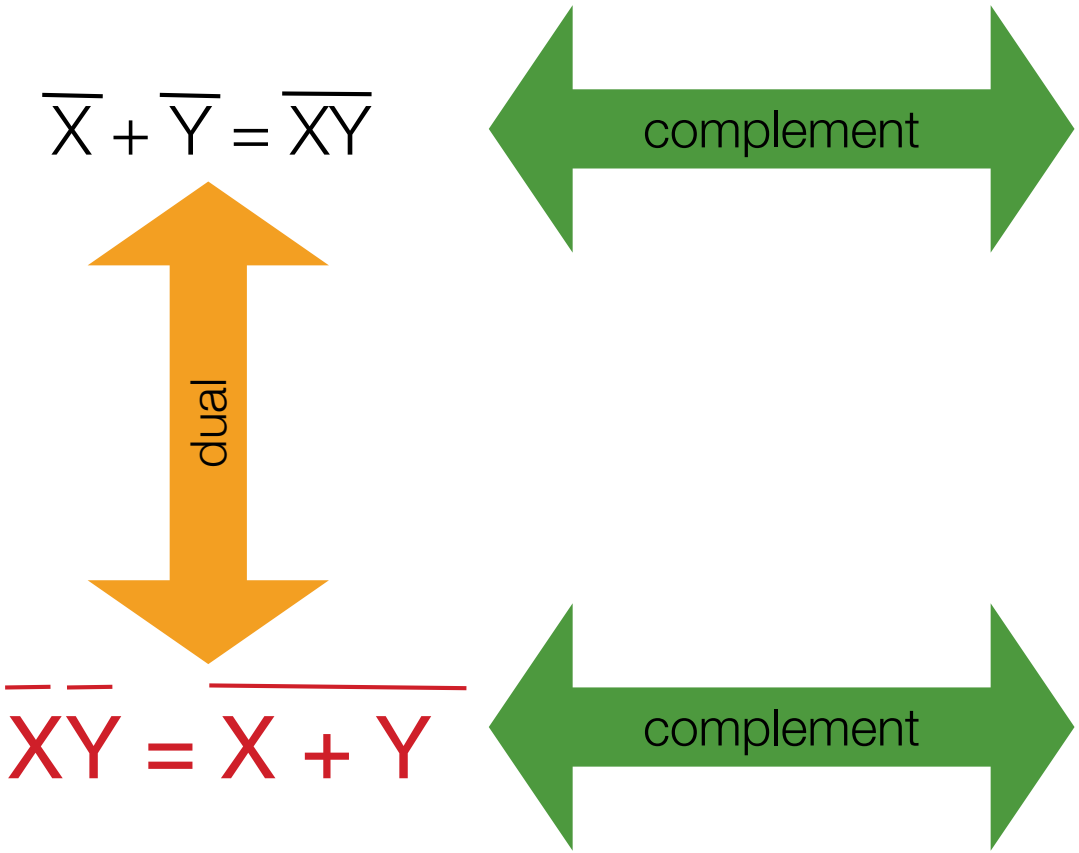
What is the dual of this expression?

$$\overline{X} + \overline{Y} = \overline{XY}$$

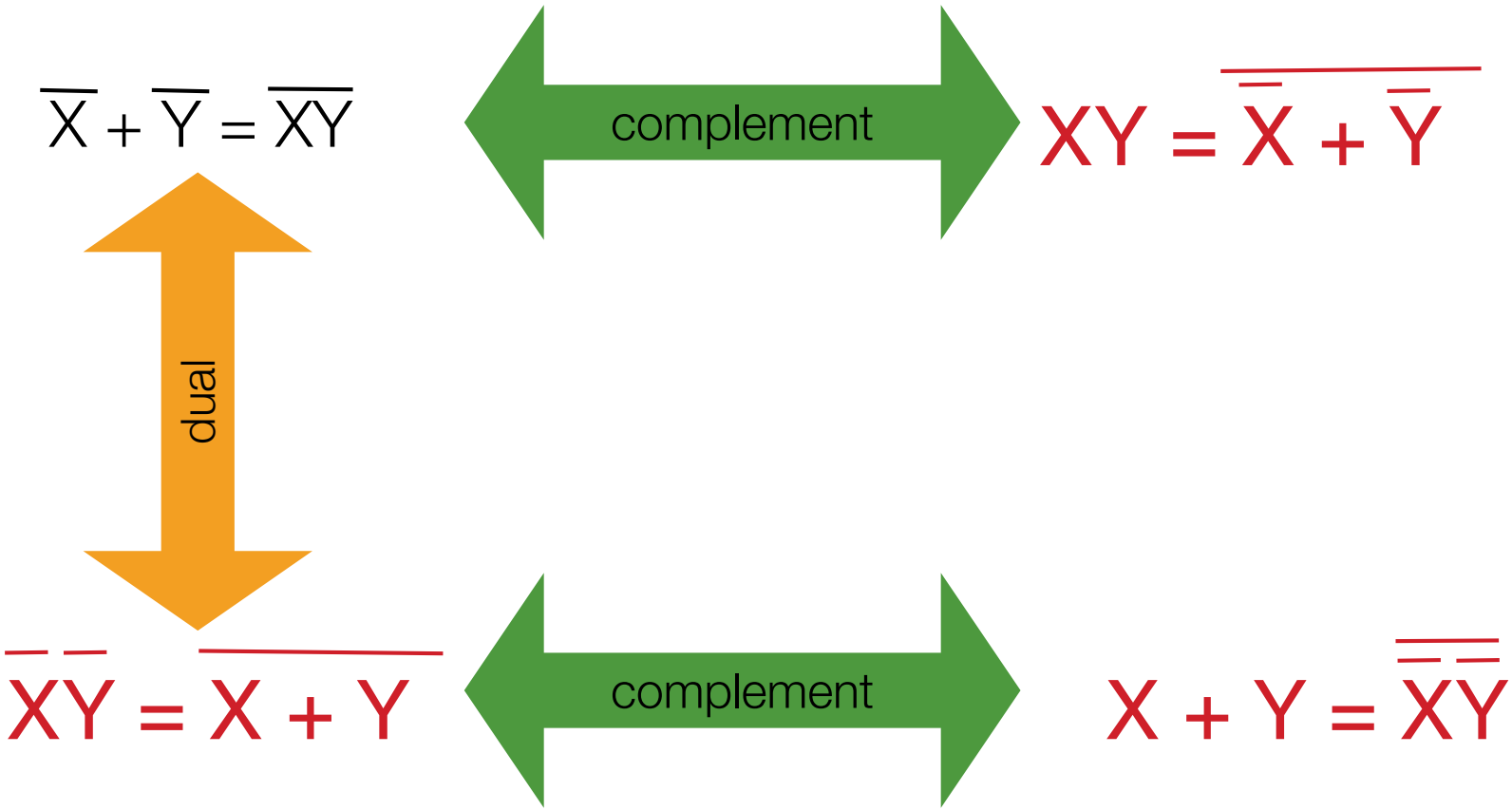
What is the dual of this expression?



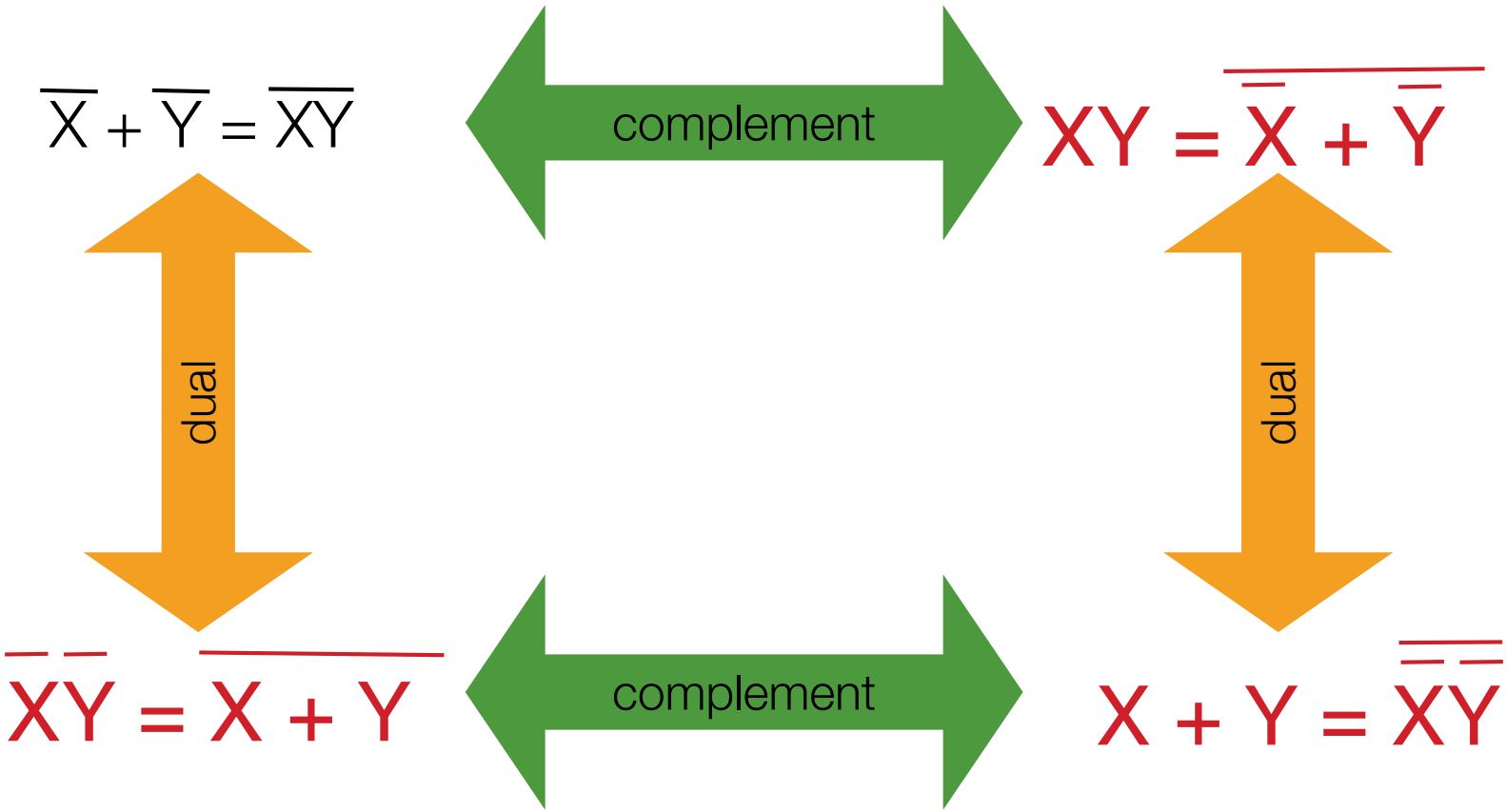
What are the complements of these expressions?



What are the complements of these expressions?



These are also the duals of one another.



Note: to complement a function, compute its dual, then complement literals

“Complement using Dual” example

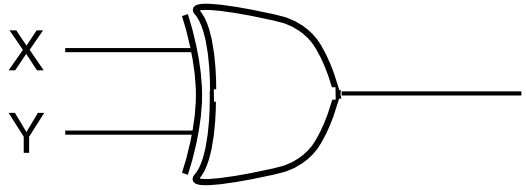
- $F = X + A\,(Z + \overline{X}\,(Y + W) + \overline{Y}\,(Z + W))$
- Dual: $F_{dual} = X\,(A + Z\,(\overline{X} + YW)(\overline{Y} + ZW))$ (swapped ANDs and ORs)
- $\overline{F} = \overline{X}\,(\overline{A} + \overline{Z}\,(X + \overline{Y}\,\overline{W})(Y + \overline{Z}\,\overline{W}))$ (“Flipped” literals)

Revisiting XOR

XOR: the parity operation

- $X \oplus Y = X\bar{Y} + \bar{X}Y$

X	Y	X ⊕ Y
0	0	0
0	1	1
1	0	1
1	1	0



- In general, represents parity, i.e.,

- $X_1 \oplus X_2 \oplus X_3 \oplus \ldots \oplus X_k = 1$ when an odd number of $X_i = 1$

Summary

- Subtle things that are very important to remember from this lecture:
  - Boolean order of precedence (slide 6)
  - XOR (slide 8)
  - Truth Tables (slide 10)
  - Boolean (basic) theorems (slide 15)
  - DeMorgan’s Law (slide 21) & using to simplify complemented expressions (slide 50)