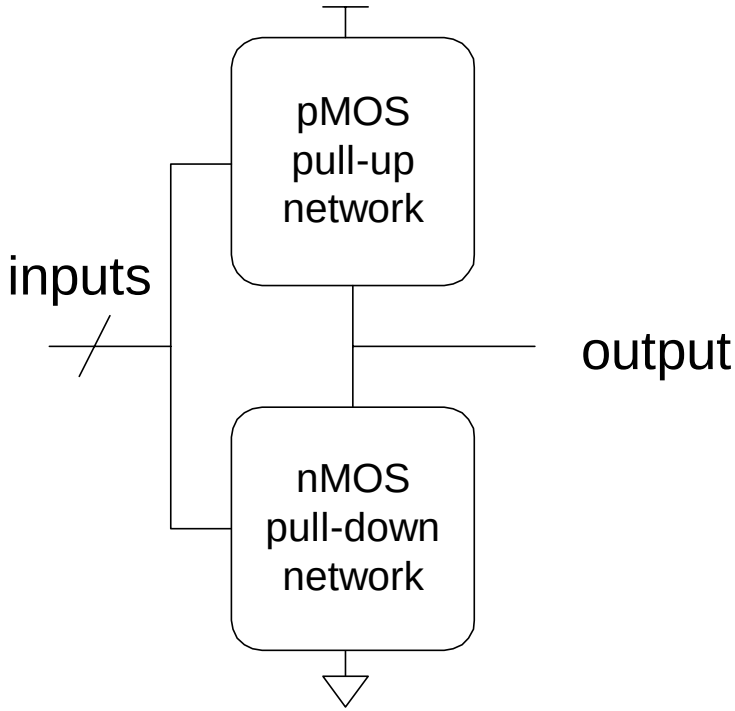
Complementary CMOS

Comlementary CMOS Logic Gates:

- nMOS pull-down network
- pMOS pull-up network
- Static CMOS



	Pull-up OFF	Pull-up ON
Pull-down OFF	**Z (float)**	**1**
Pull-down ON	**0**	**X(Crowbar)**

- Complementary CMOS gates always produce
  1 or 0
- Pull-up network is complement (dual) of pull-down network

Building CMOS Gates (n-side)

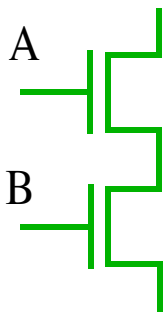
CMOS is inherently inverting.

- Gates with expression of the form $F = (\overline{expression})$ are easier to build.

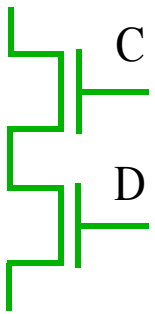
For making the n-side (pull-down network) use the un-inverted expression.

- For e.g.: Implement $F = \overline{((A \bullet B) + (C \bullet D))}$
- For n-side use $F = (A \bullet B) + (C \bullet D)$
- *AND* expressions are implemented using series connection of n transistors
- *OR* expressions are implemented using parallel connection of n transistors

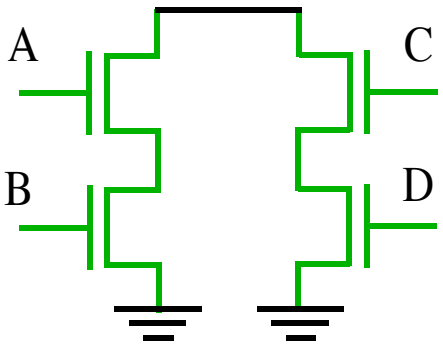
$(A \bullet B)$ — *AND*: Series



$(C \bullet D)$ — *AND*: Series



$(A \bullet B) + (C \bullet D)$ — *OR*: Parallel

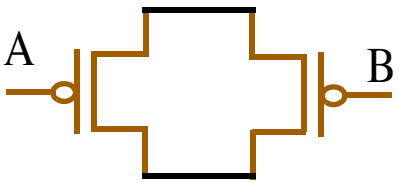


Building CMOS Gates (p-side)

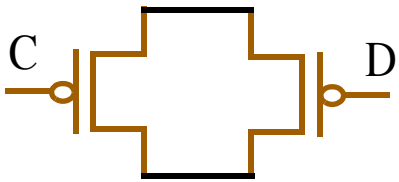
For making the p-side (pull-up network) invert the expression used for n-side.

- For e.g.: Implement $F = \overline{((A \bullet B) + (C \bullet D))}$
- For n-side use $F = (A \bullet B) + (C \bullet D)$ (previous slide)
- For p-side invert above expression: $F = (\overline{A} + \overline{B}) \bullet (\overline{C} + \overline{D})$
- ***AND*** expressions are implemented using series connection of p transistors
- ***OR*** expressions are implemented using parallel connection of p transistors

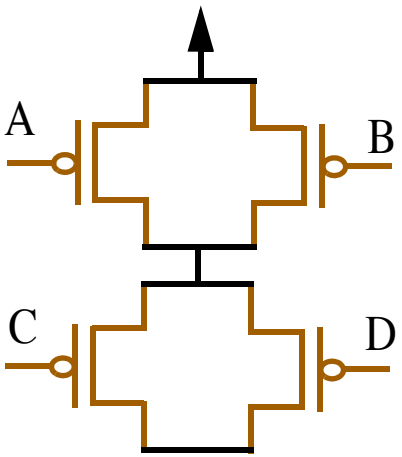
$(A + B)$ ***OR***: Parallel



$(C + D)$ ***OR***: Parallel



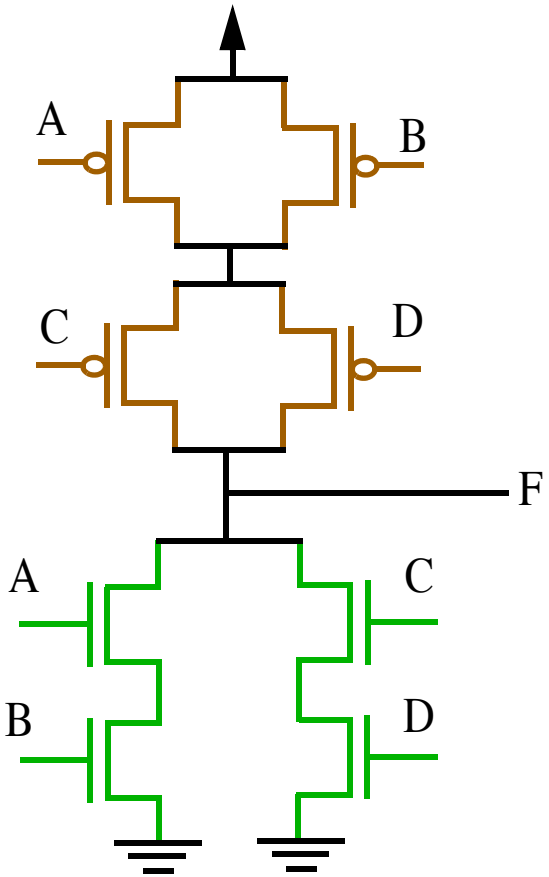
$(A + B) \bullet (C + D)$ ***AND***: Series



Building CMOS Gates (Final CMOS gate)

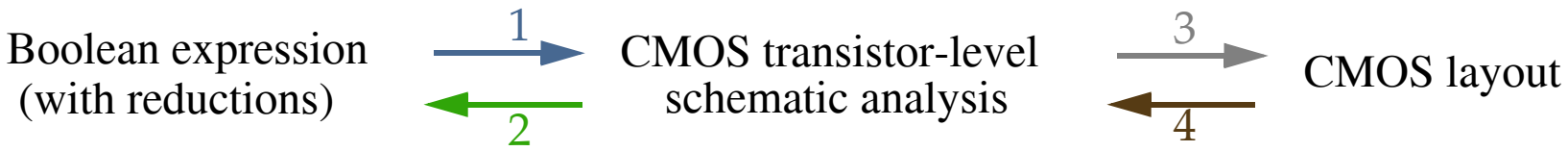
Combine the n-side (pull-down) and p-side (pull-up) to make the final gate.

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



Useful Transformations

You must know all the following transformations between levels of abstractions



(1) Boolean expression to CMOS transistor schematic

- Previous analysis shows how to do this step
- It assumes that the Boolean expression is already in the appropriate form, which may not always be ture.

Boolean Expression Reduction

- You should already know how to manipulate boolean expressions, e.g., using De Morgan's Laws, from exercises in other courses.
- The objective is to reduce a boolean expression so that it can be realized in full-complementary CMOS using the minimum number of transistors.
- You are not expected to realize CMOS gates using *pass structures* in which the inputs are used to drive the output of the gate. e.g. (Lecture 2: XOR, XNOR)

Useful Transformations

The following heuristics can be applied as target reductions that will help you to obtain minimum realizations:

- Since CMOS is naturally inverting, you'll want to target a final expression of the form:

$$F = \overline{(\text{expression})}$$

- Many times only **un**complemented literals are available as signals in your circuit.

  Therefore, the reductions should attempt to **remove** the complemented literals in the Boolean expression.

  Application of De Morgan's Laws can be used to transform complemented literals to NANDs and NORs.

- You should analyze each transformation to learn the trade-offs.

Transformation Examples

Let's try:

$$F = (\bar{A}B) + (C + D)E$$

The following reduction sequence can be applied that targets NANDs and removes the complemented literals:

$F = (\bar{A}B) + (C + D)E$ — *Build Inverse : 14 Transistors*

$\bar{F} = \overline{(\bar{A}B) + (C + D)E}$ — *Invert both sides.*

$\bar{F} = \overline{\bar{A}B} \bullet \overline{(C + D)E}$ — *How many transistors are needed here?*

$\bar{F} = (A + \bar{B}) \bullet \overline{(C + D)E}$ → **Build here?:**

$F = \overline{(A + \bar{B}) \bullet \overline{(C + D)E}}$

transistors: 6 for OAI, 2 for inverter for B, 6 for final OAI.

$\bar{F} = A\overline{(C + D)E} + \bar{B}\overline{(C + D)E}$ — *Multiply.*

$\bar{F} = A\overline{(C + D)E} + \overline{B + (C + D)E}$

$F = \overline{A\overline{(C + D)E} + \overline{B + (C + D)E}}$ — *Or Build here?: 6 for OAI, 8 for B AOI, 6 for final AOI.*

Transformation Examples

Note that further reductions to NANDs and NORs may not pay off in the previous case.

In the next case, it is possible to get rid of an uncomplemented literal without increasing the size of the OAI:

$$F = (\bar{A}B) + (C + D)\bar{E}$$

Build Inverse : 16 Transistors

$$\bar{F} = \overline{(\bar{A}B) + (C + D)\bar{E}}$$

Invert both sides.

$$\bar{F} = \overline{\bar{A}B} \bullet \overline{(C + D)\bar{E}}$$

Apply DeMorgan's Laws.

$$\bar{F} = (A + \bar{B}) \bullet \overline{(C + D)\bar{E}}$$

Build here?: $F = (A + \bar{B}) \bullet \overline{(C + D)\bar{E}}$

transistors: 6 for OAI, 4 for inverters, 6 for final OAI.

$$\bar{F} = (A + \bar{B})(\overline{C + D} + E)$$

Or Build here?: 4 for NOR, 2 for inverter, 8 for final OAI.

Further transformations are not useful -- convince yourself.

Transformation Examples

Expressions with repeated variables may be simplified to save a couple transistors

$$F = A\overline{BC} + \overline{A}CD$$

$$\overline{F} = \overline{A\overline{BC} + \overline{A}CD} \qquad 4 + 2 + 10 + 2$$

$$\overline{F} = (\overline{A} + BC)(A + \overline{CD}) \qquad 2 + 4 + 10$$

$$\overline{F} = \overline{A}A + \overline{A}\overline{CD} + ABC + BC\overline{CD}$$

$BC\overline{CD}$ is redundant (covered) by the other terms, e.g,

$$\overline{F} = \overline{A}\overline{CD} + BC(A + \overline{CD}) = \overline{A}\overline{CD} + BC(A + \cancel{\overline{A}\overline{CD}})$$

$$\overline{F} = \overline{A}\overline{CD} + ABC \qquad 2 + 4 + 10$$

$$\overline{F} = (\overline{A + CD}) + ABC$$

$$F = \overline{(\overline{A + CD}) + ABC} \qquad 6 + 8 \text{ (14 !)}$$

Transformation Examples

In contrast to:

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

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

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

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

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

$$\bar{F} = (A\bar{A} + AD + \bar{A}\bar{B} + \bar{B}D)(\bar{C} + D)$$

$$\bar{F} = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}D + AD\bar{C} + ADD + \bar{B}D\bar{C} + \bar{B}DD$$

$$\bar{F} = \bar{A}\bar{B}\bar{C} + \bar{A}\bar{B}D + AD + \bar{B}D$$

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

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

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

6 for AB NAND, 8 for OAI,
4 for final NAND.

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

8 for OAI, 2 for inverted B,
6 for final OAI.

6 for NOR, 2 for inverter,
8 for final OAI -- *no better than the earlier expression.*

Transformation Examples

Sometimes it is best to implement the *inverse* function and add an inverter.

For example, Carry, which has all uncomplemented inputs.

$$\text{Carry} = AB + C_{in}(A + B)$$

$$\overline{\text{Carry}} = \overline{AB + C_{in}(A + B)}$$

What about XOR and XNOR?

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

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

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

$$F = \overline{AB + \overline{A}\overline{B}}$$

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

How many transistors are needed here?

The best way to learn this is through practice.

Simply make up an expression of multiple variables and invert a couple of the literals and/or subexpressions.

Useful Transformations

(2) Translating from **transistor-level schematics** to **Boolean expressions** is straightforward.

Simply write the *n-tree* expression using the rules for series and parallel transistors given earlier. Invert the final expression.

(3) Translating from **transistor-level schematic** diagrams to **layout** is covered in the laboratories.

(4) Translating from **layout** to **transistor-level schematic** diagrams is also covered in the laboratories.

- In general, start by identifying the transistor sources connected to V_{DD} or GND nodes.
- Add series transistors in the schematic for transistors whose sources are connected to drains of the previously identified transistors.
- Add parallel transistors at fan-out points.
- Label the transistors so it possible to connect the gates properly by tracing the poly connections.