Disclaimer

The slides presented here are a combination of the CS251 course notes from previous terms, the work of Xiao-Bo Li, and material from the required textbook "Computer Organization and Design, ARM Edition," by David A. Patterson and John L. Hennessy. It is being used here with explicit permission from the authors.

CS251 course policy requires students to delete all course files after the term. Therefore, please do not post these slides to any website or share them.

CS251 - Computer Organization and Design

Intro to Digital Logic Design

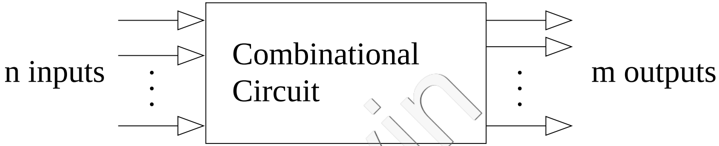
Instructors: Zille Huma Kamal and Richard Mann

University of Waterloo

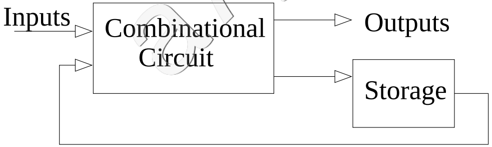
Winter 2023

Logic Blocks

- Combinational: without memory



- Sequential: with memory



- Inputs and outputs are 1/0
  (High/low voltage, true/false)

Compact alternative: Boolean algebra

- Variables (usually A, B, C or X, Y, Z) have values 0 or 1
- OR $(+)$ operator has result 1 iff either operand has value 1
- AND $(\cdot)$ operator has result 1 iff both operands have value 1
  $A \cdot B$ often written AB
- NOT $(\neg)$ operator has result 1 iff operand has value 0
  $\neg A$ usually written $\bar{A}$

OR		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

AND		
A	B	AB
0	0	0
0	1	0
1	0	0
1	1	1

NOT	
A	¬A
0	1
1	0

Specifying input/output behaviour

- Truth table: specifies outputs for each possible input combination

X	Y	Z	F	G
0	0	0	0	1
0	0	1	1	1
0	1	0	0	1
0	1	1	0	1
1	0	0	0	1
1	0	1	1	1
1	1	0	1	1
1	1	1	1	0

- Complete description, but big and hard to understand
- For truth table, $G = \overline{XYZ}$
- $F = \bar{X}\bar{Y}Z + X\bar{Y}Z + XY\bar{Z} + XYZ$ (not obvious)

Truth Table to Formula Using Minimal Terms

A	B	C	F	$\bar{A}\bar{B}C$	$A\bar{B}C$	ABC	$\bar{A}\bar{B}C + A\bar{B}C + ABC$
0	0	0	0	0	0	0	0
0	0	1	1	1	0	0	1
0	1	0	0	0	0	0	0
0	1	1	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	1	1	0	1	0	1
1	1	0	0	0	0	0	0
1	1	1	1	0	0	1	1

Two-Level Representations

- Any Boolean function can be represented as a sum of products (OR of ANDs) of literals
- Each term in sum corresponds to a single line in truth table with value 1
- This can be simplified by hand or by machine
- Product of sums representation may also be useful

Don't Cares in Truth Tables

- Represented as X instead of 0 or 1
- When used in output, indicates that we don't care what output is for that input
- When used in input, indicates outputs are valid for all inputs created by replacing X by 0 or 1 (useful in compressing truth tables)
- Example:

A	B	C	F
0	0	X	0
0	1	X	1
1	X	X	X

Compressed Truth Tables and Non-Minimal Terms

A	B	C	F	$\bar{A}\bar{B}C$	AC	$\bar{A}\bar{B}C + AC$
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	X	0	0	0	0
1	X	0	0	0	0	0
1	X	1	1	0	1	1

Using Overlapping Non-Minimal Terms

A	B	C	F	AB	AC	$AB + AC$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	0	0	0	0
1	0	1	1	0	1	1
1	1	0	1	1	0	1
1	1	1	1	1	1	1

Laws of Boolean Algebra

Rule	Dual Rule	
$\bar{\bar{X}} = X$		
$X + 0 = X$	$X \cdot 1 = X$	(identity)
$X + 1 = 1$	$X \cdot 0 = 0$	(zero/one)
$X + X = X$	$XX = X$	(absorption)
$X + \bar{X} = 1$	$X\bar{X} = 0$	(inverse)
$X + Y = Y + X$	$XY = YX$	(commutative)
$X + (Y + Z) = (X + Y) + Z$	$X(YZ) = (XY)Z$	(associative)
$X(Y + Z) = XY + XZ$	$X + YZ = (X + Y)(X + Z)$	(distributive)
$\overline{X + Y} = \bar{X} \cdot \bar{Y}$	$\overline{XY} = \bar{X} + \bar{Y}$	(DeMorgan)

Formula Simplification Using Laws

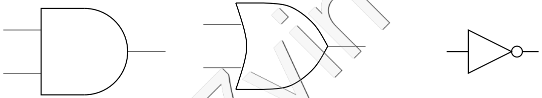
- We can use algebraic manipulation (based on laws) to simplify formulas
- An example using the previous truth table

$$\begin{aligned} F &= \bar{X}\bar{Y}Z + X\bar{Y}Z + XY\bar{Z} + XYZ \\ &= \bar{Y}Z(\bar{X} + X) + XY(\bar{Z} + Z) \\ &= \bar{Y}Z + XY \end{aligned}$$

- Difficult even for humans, tricky to automate
- Seems inherently hard to get “simplest” formula
- Is simplest formula the best for implementation?

Using Gates in Logic Design

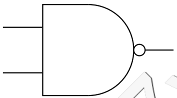
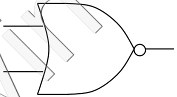
- Here are symbols for AND, OR, NOT gates



- NOT often drawn as “bubble” on input or output
- AND, OR can be generalized to many inputs (useful)
- We can design using AND, OR, NOT, and optimize afterwards

NAND and NOR

- In practice, logic minimization software works with NAND or NOR gates, or at transistor level
- Here are symbols for NAND, NOR:

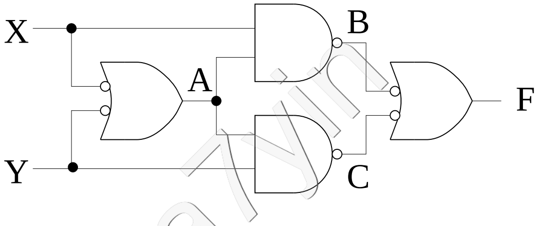
NAND		
A	B	$\overline{AB}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR		
A	B	$\overline{A+B}$
0	0	1
0	1	0
1	0	0
1	1	0

- Two level NAND circuit

Deriving Truth Table from Circuit

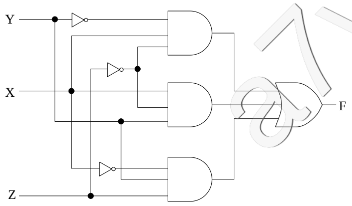
- Label intermediate gate outputs
- Fill in truth table in appropriate order



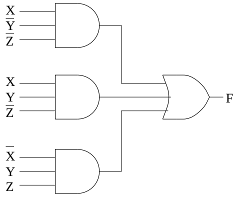
X	Y	A	B	C	F
0	0				
0	1				
1	0				
1	1				

Good Style in Circuit Drawing

- Assume all literals (variables and their negations) are available
- Rectilinear wires, dots when wires split
- Do not draw spaghetti wires for inputs; instead, write each literal as needed



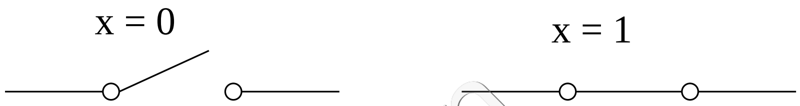
Bad



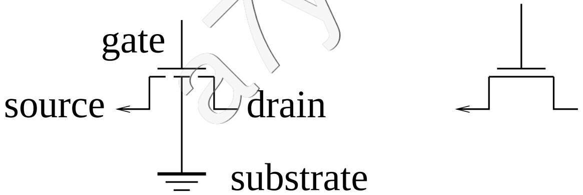
Good

Implementing Gates Using Transistors

- Transistor: an electrically-controlled switch

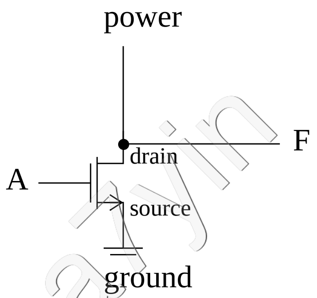


- An NMOS transistor ("n-transistor") and its symbol



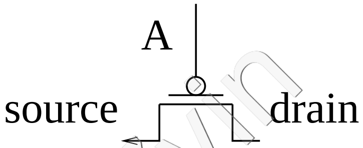
- This behaves like the switch above
- Problem: transmits strong 0 but weak 1

An NMOS NOT



- If $A = 1$, then low resistance between drain and source ($F = 0$)
- If $A = 0$, then very high resistance between drain and source ($F = 1$)
- Problem: in $A = 1$ case, lots of current flow

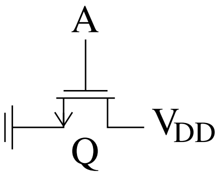
A PMOS transistor



- Opposite behaviour to NMOS:
  - If $A = 1$, high resistance between drain and source
  - If $A = 0$, low resistance between drain and source
  - Transmits strong 1 but weak 0
- Denote inversion with "bubble"

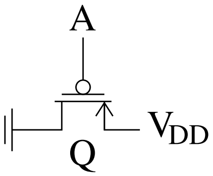
Transistor Summary

- Two types of transistors: nmos, pmos



NMOS

Input	$A = 0$	$A = 1$
Resistance	High	Low S0, W1



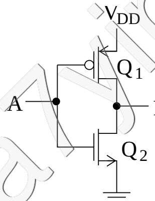
PMOS

Input	$A = 0$	$A = 1$
Resistance	Low W0, S1	High

- To analyze CMOS circuit:
  - Make table with inputs, transistors, and output(s)
  - For each row of table (setting of inputs),
    check whether transistor resistance is High,Low
  - For each row of table, check if output has clean path to
    power (1)
    ground (0)

CMOS

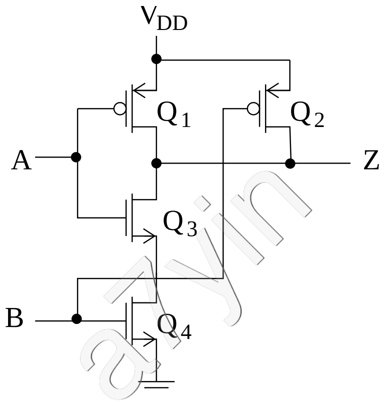
- CMOS circuits use both n-transistors and p-transistors
- Will build circuits with "clean" paths to exactly one of power and ground.
- CMOS NOT:



A	Q_1	Q_2	F
0	Low	High	1
1	High	Low	0

- No bad flow of current from power to ground
- No weak transmissions

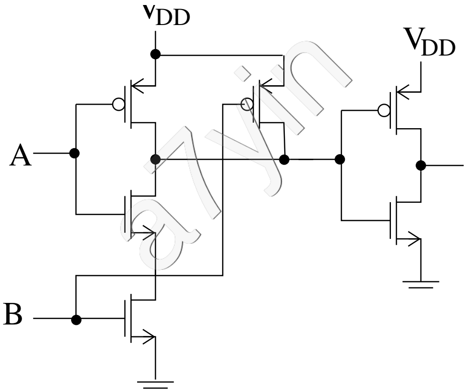
CMOS NAND



A	B	Q_1	Q_2	Q_3	Q_4	Z
0	0	Low	Low	High	High	1
0	1					
1	0					
1	1					

CMOS AND and OR

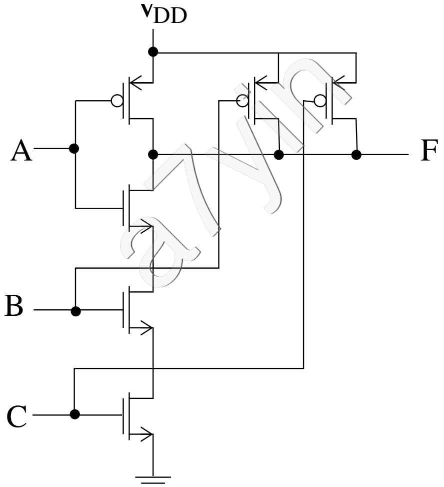
- To get AND and OR, add inverter at end
- Example:



- Thus, NAND is preferred to AND in actual circuits

CMOS 3 Input NAND and NOR

- *n*-input NAND: 2 transistors per input
- Example: 3 Input NAND

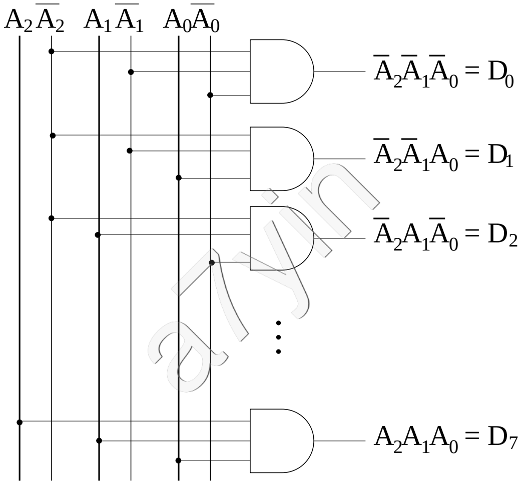


Useful Components: Decoders

- n inputs, 2^n outputs (converts binary to "unary")
- Example: 3-to-8 (or 3-bit) decoder

A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0	0	0
1	0	0	0	0	0	1	0	0	0	0
1	0	1	0	0	1	0	0	0	0	0
1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	0	0

- Circuit has regular structure



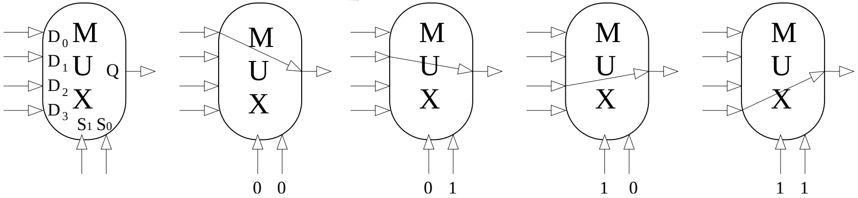
A_2 $\overline{A_2}$ A_1 $\overline{A_1}$ A_0 $\overline{A_0}$
$\overline{A}_2\overline{A}_1\overline{A}_0 = D_0$
$\overline{A}_2\overline{A}_1A_0 = D_1$
$\overline{A}_2A_1\overline{A}_0 = D_2$
$A_2A_1A_0 = D_7$

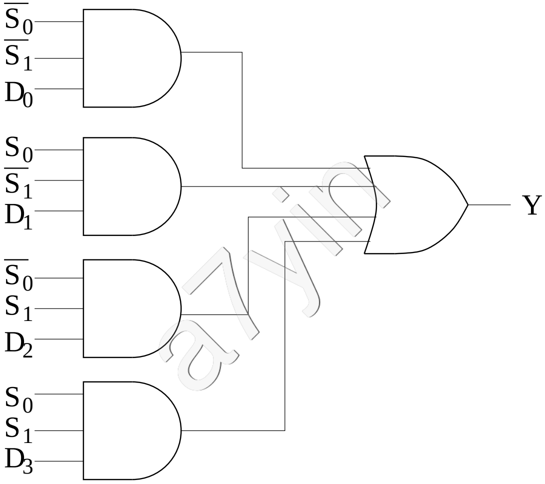
3-to-8 (or 3-bit) Decoder

Multiplexors

- Inputs: 2^n lines $(D_0, \ldots, D_{2^n-1})$
  n select lines $(S_{n-1}, \ldots, S_0)$
- Output: The value of the D_S line
- Example: 4-1 Multiplexer

S_1	S_0	Y
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

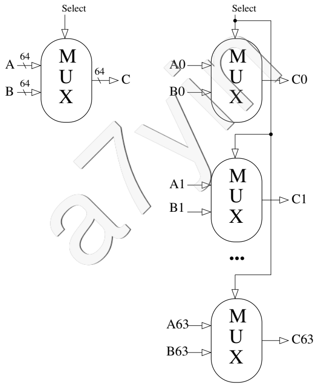




4-1 Multiplexor

Arrays of Logic Elements

- “Slash” notation is used to indicate lines carrying multiple bits, and to imply parallel constructions



64-bit wide, 2:1 multiplextor expands to 64, 1 bit, 2:1 multiplexors

Readings to accompany this lecture

- Appendix A, sections A.1–A-3

Disclaimer

The slides presented here are a combination of the CS251 course notes from previous terms, the work of Xiao-Bo Li, and material from the required textbook "Computer Organization and Design, ARM Edition," by David A. Patterson and John L. Hennessy. It is being used here with explicit permission from the authors.

CS251 course policy requires students to delete all course files after the term. Therefore, please do not post these slides to any website or share them.