18-100 Lecture 21: Boolean Logic

James C. Hoe
Dept of ECE, CMU
April 7, 2015

Today's Goal: Introduce Boolean logic

Announcements: Read Rizzoni 12.3 and 11.5
HW8 due Thursday
Office Hours: Wed 12:30~2:30

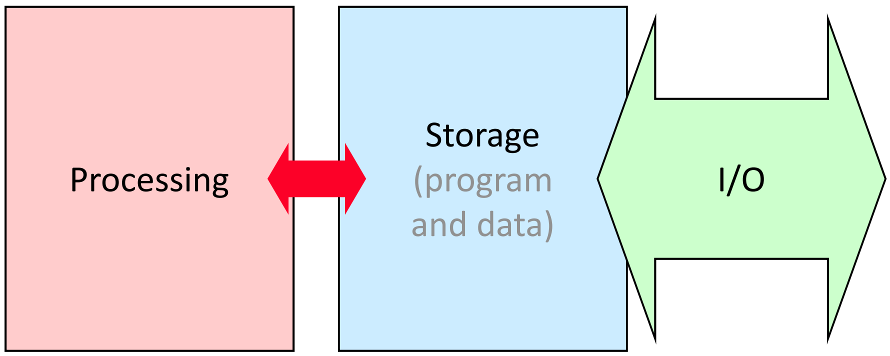
Handouts: Lab 11 (on Blackboard)

To Wrap up AVR

- To be a hacker, you also need to know
  - subroutine calls, in particular recursive calls
  - exception handling
  - I/O
  - how to hand optimize code

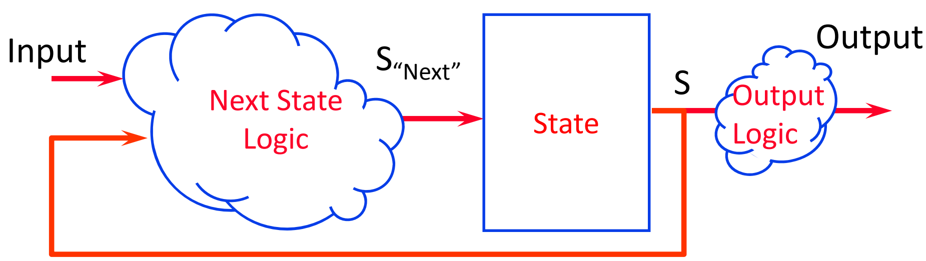
    Feel free to read the AVR documents on Blackboard
- Big picture to keep in mind
  - you will see assembly programming in much greater detail in 213/240/34{8,9}/447 etc.
  - most of you will not code in assembly for a living; this is more about understanding the underlying concepts
  - once you learn one ISA you can learn the rest
    - some ISAs are uglier than others
    - AVR is not a pretty one.

Recall the 3 aspects of a "computer"



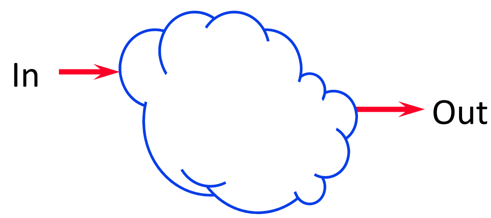
In General

- All "sequential" digital systems comprise



  - input and output
  - state stuff that remembers
  - "combinational" stuff that computes a function (has no memory)
- In an execution, state is updated to a "next state" based on a function of the current state

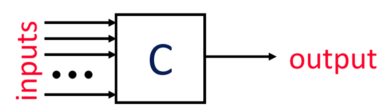
First things first: Combinational Logic



The same In *value always leads to the same* Out *value regardless of history*

Combinational Logic

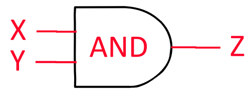
- A combinational logic circuit has an output that is a "function" of its inputs



- What do you mean by "function"?
  - unique mapping from input values to output values
  - in this context implies
    - the same input values produce the same output value every time
    - same output for different input values is okay
    - no memory (does not depend on the history of input values)
- Later on we will learn, a "sequential" circuit's output depends on the current and the history of its inputs

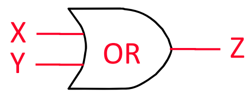
Examples: and / or / not

AND: Z is true iff both X and Y are true



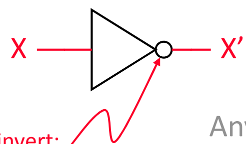
X	Y	Z
F	F	F
F	T	F
T	F	F
T	T	T

OR: Z is true iff X or Y or both are true



X	Y	Z
F	F	F
F	T	T
T	F	T
T	T	T

NOT: X' is the complement of X



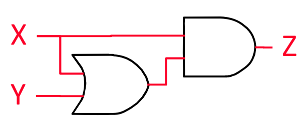
X	X'
F	T
T	F

bubble means invert;
triangle is "buffer"

Any logical function can be constructed using only AND and NOT gates, or only OR and NOT gates

Boolean Algebra: Big Picture

- An algebra on
  - two values (usually denoted as True vs False or 1 vs 0)
  - the operators AND (·), OR (+), NOT
- A useful formalism to represent and reason about combinational logic functions
- For example,



  - combinational logic above can be expressed as Z=X·(X+Y)
  - using rules of Boolean algebra, the above expression can be simplified to Z=X

Rules of Boolean Algebra (Table 12.11)

"·" means AND
"+" means OR

1. $0+X=X$
2. $1+X=1$
3. $X+X=X$
4. $X+X'=1$
9. $(X')'=X$
10. $X+Y=Y+X$
12. $X+(Y+Z)=(X+Y)+Z$
14. $X\cdot(Y+Z)=X\cdot Y+X\cdot Z$

6. $1\cdot X=X$
5. $0\cdot X=0$
7. $X\cdot X=X$
8. $X\cdot X'=0$
11. $X\cdot Y=Y\cdot X$ (commutativity)
13. $X\cdot(Y\cdot Z)=(X\cdot Y)\cdot Z$ (associativity)
(14). $X+(Y\cdot Z)=(X+Y)\cdot(X+Z)$ (distributive)

- The above is a standard set of rules that can be easily shown by "perfect induction"
- Notice the symmetry between the left and right columns with respect to substitutions of 0 by 1, + by · and vice versa

Does $X+(Y\cdot Z)=(X+Y)\cdot(X+Z)$?

X	Y	Z	(Y·Z)	**X+(Y·Z)**	(X+Y)	(X+Z)	**(X+Y) ·(X+Z)**
F	F	F	F	**F**	F	F	**F**
F	F	T	F	**F**	F	T	**F**
F	T	F	F	**F**	T	F	**F**
F	T	T	T	**T**	T	T	**T**
T	F	F	F	**T**	T	T	**T**
T	F	T	F	**T**	T	T	**T**
T	T	F	F	**T**	T	T	**T**
T	T	T	T	**T**	T	T	**T**

Enumerate all possible patterns; n variables have 2^n patterns

Evaluate the expressions in question for each case, possibly with the help of intermediate expressions like (Y·Z)

Electrical & Computer ENGINEERING

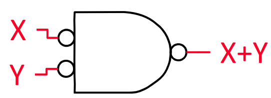
CMU 18-100
S'15 L21-11
© 2015
J. C. Hoe

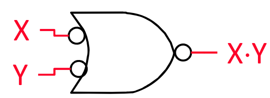
Duality

- Is that a truth table for AND or OR?
  - Ans. depends on ☜ means true or false
  - if ☜ means true then OR
  - if ☜ means false then AND

X	Y	Z
☜	☜	☜
☜	☞	☜
☞	☜	☜
☞	☞	☞

- AND and OR are duals





⇒ De Morgan's Theorem

$$X'\cdot Y' = (X+Y)'$$
$$X'+Y' = (X\cdot Y)'$$

⇒ For every rule, there is a "dual" rule where AND and OR are exchanged and 0 and 1 are exchanged

Electrical & Computer ENGINEERING

CMU 18-100
S'15 L21-12
© 2015
J. C. Hoe

Does X+XY=X ?

- Instead of perfect induction, we can also prove algebraically using proven rules, e.g.,

	$X+X\cdot Y$	
=	$1\cdot X+X\cdot Y$	rule 6
=	$X\cdot 1+X\cdot Y$	rule 11
=	$X\cdot(1+Y)$	rule 14
=	$X\cdot 1$	rule 2
=	$1\cdot X$	rule 11
=	X	rule 6

Does XY+YZ+X'Z=XY+X'Z ?

Rules of Boolean Algebra (Table 12.11)

15. $X+X\cdot Y=X$
16. $X\cdot(X+Y)=X$ (absorption)
17. $(X+Y)(X+Z)=X+YZ$

(17). $XY+XZ=X(Y+Z)$

18. $X+X'Y=X+Y$

(18). $X\cdot(X'+Y)=XY$

19. $XY+YZ+X'Z=XY+X'Z$

(19). $(X+Y)(Y+Z)(X'+Z)=(X+Y)(X'+Z)$

DM. $(A+B+C+\ldots)' = A'\cdot B'\cdot C'\ldots$

(DM). $(A\cdot B\cdot C\cdot\ldots)'=A'+B'+C'+\ldots$

- The above and other rules can be derived from the basic set, as we did for Rules 15 and 19

Application in Logic Minimization

- Given f=X'Y'Z'+X'Y'Z+X'YZ+XYZ simplify the expression by algebraic manipulation

	X'Y'Z'+X'Y'Z+X'YZ+XYZ	
=	X'Y'(Z'+Z)+(X'+X)YZ	rule 14
=	X'Y'·1+1·YZ	rule 4
=	X'Y'+YZ	rule 6

- Original expression required one 4-input OR gate and four 3-input AND gates
- Simplified expression requires one 2-input OR gate and two 2-input AND gates
- Can we do any better? If not, can we know for sure?

Minimize f=A'B'D+A'BD+BCD+ACD

(Example 12.3 from Rizzoni)

Quick Peek Below the Digital Abstraction

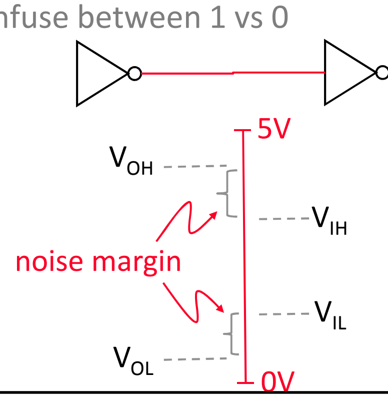
Physical Representations of 1s and 0s

- Many ways in electronics
  - voltage level (high vs low)
  - voltage change (delta up vs down)
  - relative current drive on a differential pair
  - on-off keying in various transmissions

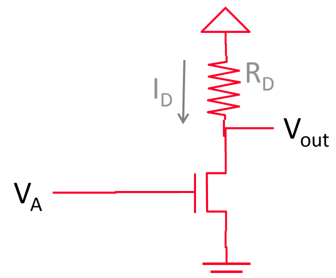
e.g., harddrive read back

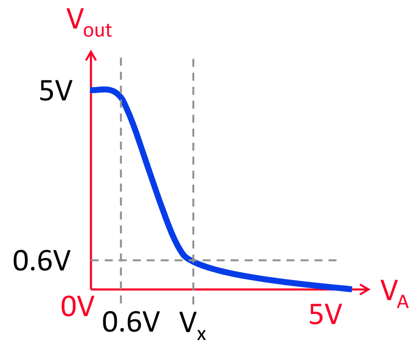
Objective: must be hard to confuse between 1 vs 0

- Voltage level noise margin
  - output devices guarantee min V_{OH} and max V_{OL}
  - input devices insist on a min V_{IH} and max V_{IL}; if violated does not register correctly as high or low



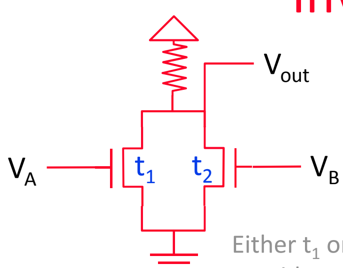
Example: NMOS Resistor-Transistor Logic





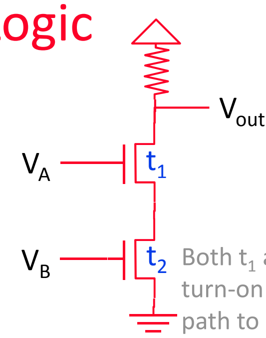
- I/O voltage levels
  - $V_{IL\text{-}max}$<turn-on≈0.6V, $V_{IH\text{-}min}$ >*(by how much?)*V_x
  - V_{OH}≈5V (without any load) , V_{OL} ≈ ??
- Noise margin: full-swing output even if input is not perfect
- But, in output-low state, I_D>0 ⇒ leaking power

Inverting Logic



Either t_1 or t_2 turn-on provides path to GND

V_A	V_B	V_{out}
0V	0V	5V
0V	5V	0V
5V	0V	0V
5V	5V	0V

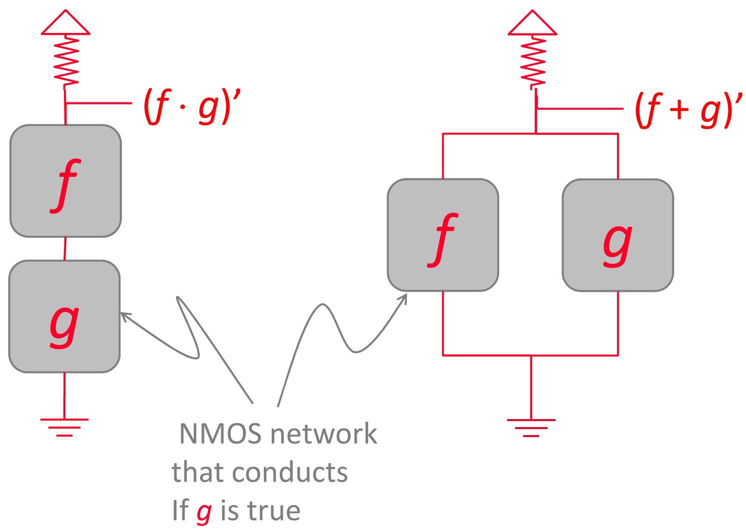


Both t_1 and t_2 must turn-on to have path to GND

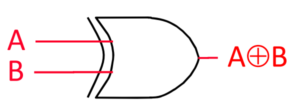
V_A	V_B	V_{out}
0V	0V	5V
0V	5V	5V
5V	0V	5V
5V	5V	0V

- Instead of AND and OR we have NAND and NOR!!
  - we can always stick an inverter on the output
  - fortunately either NAND or NOR is all we really need

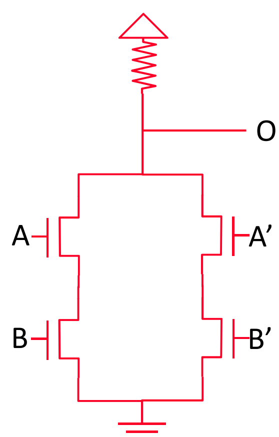
Complex Inverted Function Gates



Complex Gate Example: Exclusive OR (XOR)

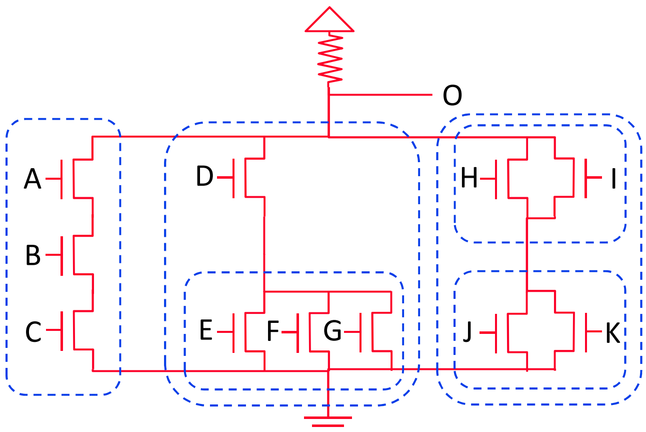


A	B	A⊕B
F	F	F
F	T	T
T	F	T
T	T	F



O=A⊕B=(AB+A'B')'

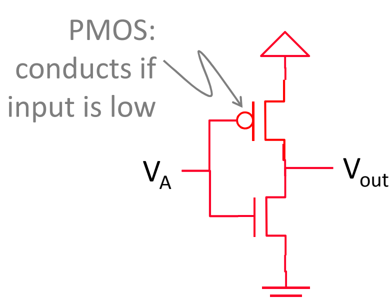
Complex Function Gate Example

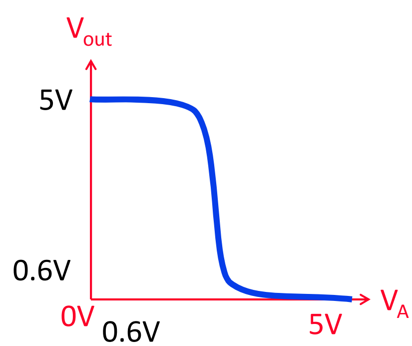


$$O=\{\ [A\cdot B\cdot C] + [D\cdot(E+F+G)] + [(H+I)\cdot(J+K)]\ \}'$$

Don't really build something this complex in one gate. Too clunky.

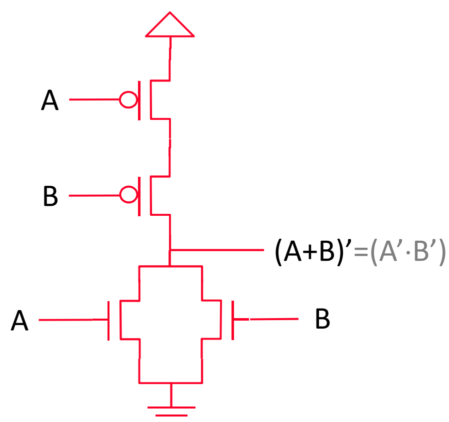
Complementary MOS (CMOS) Logic

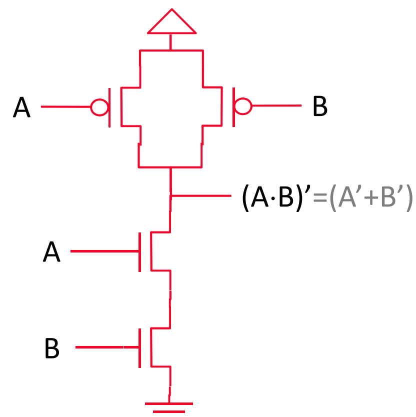




- Almost ideal inverter transfer curve
  - large, balanced noise margin; strong signal restorations
- Only 1 transistor (NMOS or PMOS) is conducting at a time
  - ideally no (very little) static leakage current/power

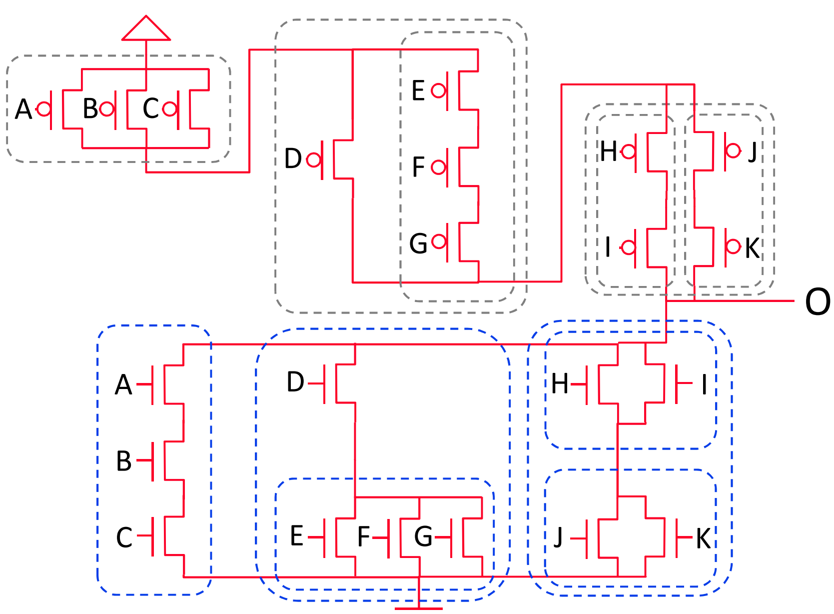
CMOS NANDs and NORs





- PMOS and NMOS networks are De Morgan duals
  ⇒ output either connected to VCC or GND, never both
- NANDs are better than NORs (take 320 to find out why)

CMOS Complex Gate Example



Minimize f=A'B'D+A'BD+BCD+ACD

(Example 12.3)

A'B'D+A'BD+BCD+ACD	
= A'D(B'+B)+BCD+ACD	rule 14
= A'D+BCD+ACD	rule 4 and 6
= (A'+AC)D+BCD	rule 14
= (A'+C)D+BCD	rule 18
= A'D+CD+BCD	rule 14
= A'D+CD(1+B)	rule 14
= A'D+CD	rule 2 and 6
= (A'+C)D	rule 14

Does XY+YZ+X'Z=XY+X'Z ?

XY+YZ+X'Z	
= XY·1+YZ·1+X'Z·1	rule 6
= XY(Z+Z')+YZ(X+X')+X'Z(Y+Y')	rule 4
= XYZ+XYZ'+YZX+YZX'+X'ZY+X'ZY'	rule 14
= XYZ+XYZ'+XYZ+X'YZ+X'YZ+X'Y'Z	rule 11
= XYZ+XYZ+XYZ'+X'YZ+X'YZ+X'Y'Z	rule 10
= XYZ+XYZ'+X'YZ+X'Y'Z	rule 3
= XY(Z+Z')+X'Z(Y+Y')	rule 14
= XY·1+X'Z·1	rule 4
= XY+X'Z	rule 6

This one is tricky

Does XY+YZ+X'Z=XY+X'Z ?

(done another way)

	XY+YZ+X'Z	
=	XY+YZ·1+X'Z	rule 6
=	XY+YZ(X+X')+X'Z	rule 4
=	XY+YZX+YZX'+X'Z	rule 14
=	XY+XYZ+X'ZY+X'Z	rule 11
=	XY+XYZ+X'Z+X'ZY	rule 10
=	XY·1+XYZ+X'Z·1+X'ZY	rule 6
=	XY(1+Z)+X'Z(1+Y)	rule 14
=	XY·1+X'Z·1	rule 2
=	XY+X'Z	rule 6

rule 15
absorption