



COMPUTER ORGANISATION (TỔ CHỨC MÁY TÍNH)

Logic gates and circuits

Acknowledgement

- The contents of these slides have origin from School of Computing, National University of Singapore.
- We greatly appreciate support from Mr. Aaron Tan Tuck Choy for kindly sharing these materials.

Policies for students

- These contents are only used for students PERSONALLY.
- Students are NOT allowed to modify or deliver these contents to anywhere or anyone for any purpose.

WHERE ARE WE NOW?

- Number systems and codes
 - Boolean algebra
 - **Logic gates and circuits**
 - Simplification
 - Combinational circuits
 - Sequential circuits
 - Performance
 - Assembly language
 - The processor: Datapath and control
 - Pipelining
 - Memory hierarchy: Cache
 - Input/output
-
- The diagram illustrates the progression of topics. A vertical blue bracket on the right side groups the topics into three main sections: 'Preparation: 2 weeks' (Number systems and codes, Boolean algebra), 'Logic Design: 3 weeks' (Logic gates and circuits, Simplification, Combinational circuits, Sequential circuits, Performance, Assembly language), and 'Computer organisation' (The processor: Datapath and control, Pipelining, Memory hierarchy: Cache, Input/output). A large red arrow points from the 'Logic Design' section towards the 'Computer organisation' section, indicating the flow of learning.
- Preparation: 2 weeks
- Logic Design: 3 weeks
- Computer organisation

LOGIC GATES AND CIRCUITS

- Gate Symbols
- Inverter/AND/OR/NAND/NOR/XOR/XNOR
- Drawing and Analysing Logic Circuits
- Universal Gates
- SOP and NAND Circuits
- POS and NOR Circuits
- Programmable Logic Array

Read up DLD for details!

LOGIC GATES

- Gate symbols

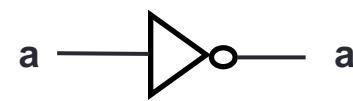
AND



OR



NOT



NAND



NOR

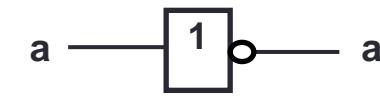


EXCLUSIVE OR



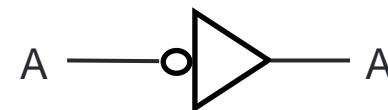
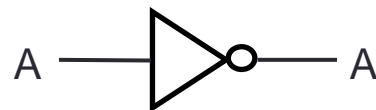
Symbol set 2

(ANSI/IEEE Standard 91-1984)



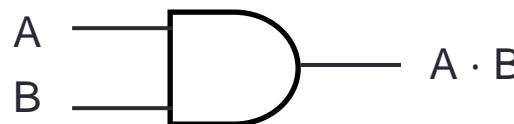
INVERTER/AND/OR GATES

- Inverter (NOT gate)



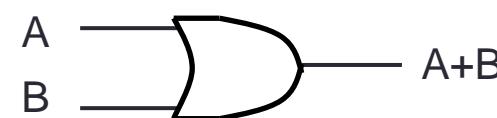
A	A'
0	1
1	0

- AND gate



A	B	A · B
0	0	0
0	1	0
1	0	0
1	1	1

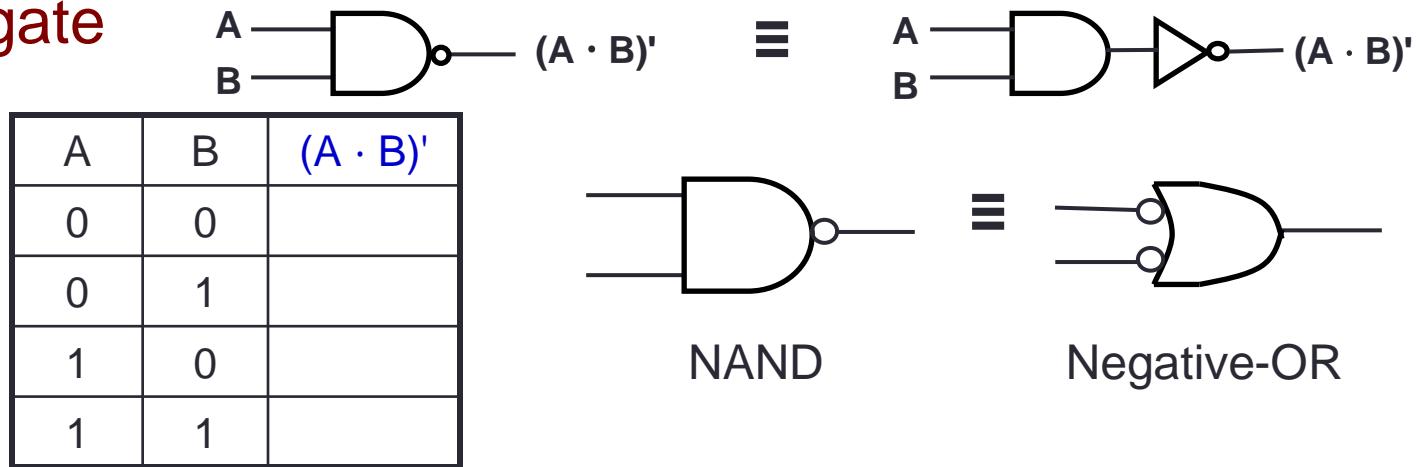
- OR gate



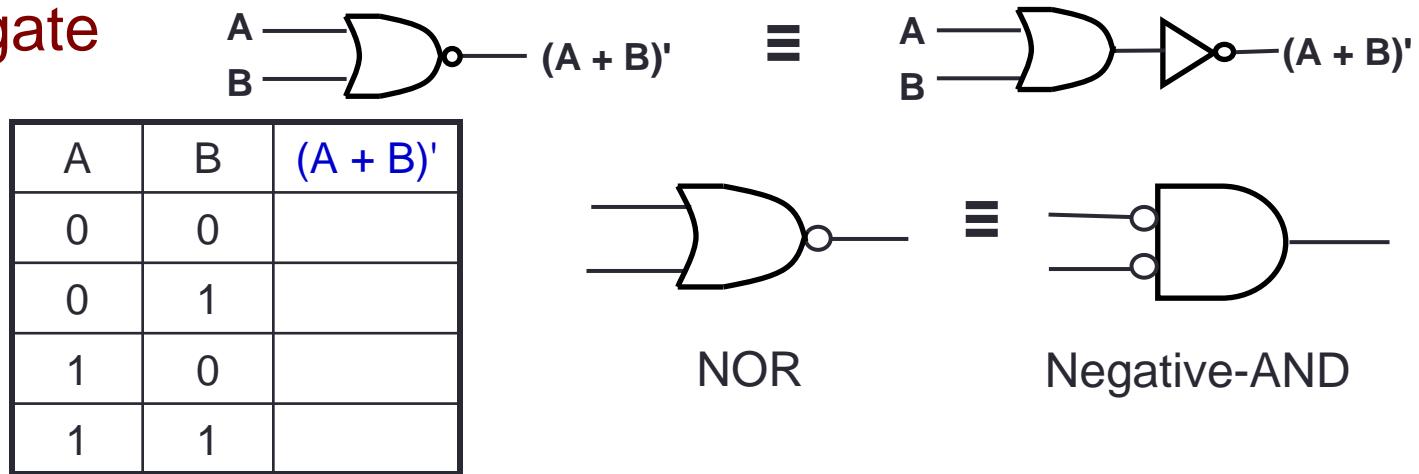
A	B	A + B
0	0	0
0	1	1
1	0	1
1	1	1

NAND/NOR GATES

- NAND gate

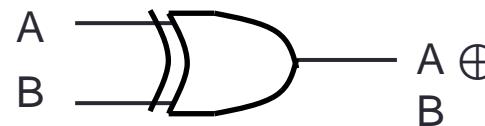


- NOR gate



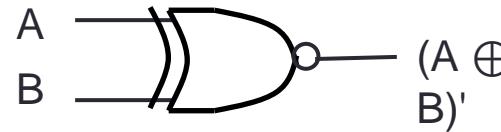
XOR/XNOR GATES

- XOR gate



A	B	$A \oplus B$
0	0	
0	1	
1	0	
1	1	

- XNOR gate

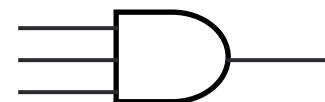


XNOR can be represented by \square
(Example: $A \square B$)

A	B	$(A \oplus B)'$
0	0	
0	1	
1	0	
1	1	

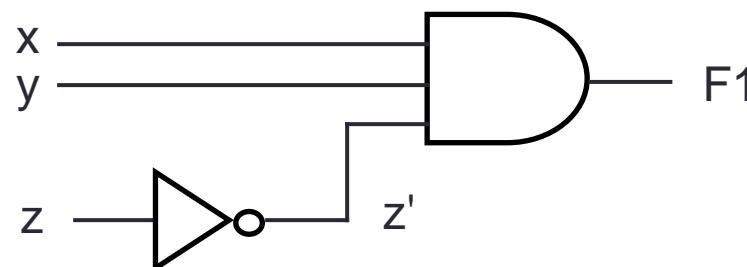
LOGIC CIRCUITS (1/2)

- Fan-in: the number of inputs of a gate.
- Gates may have fan-in more than 2.
 - Example: a 3-input AND gate



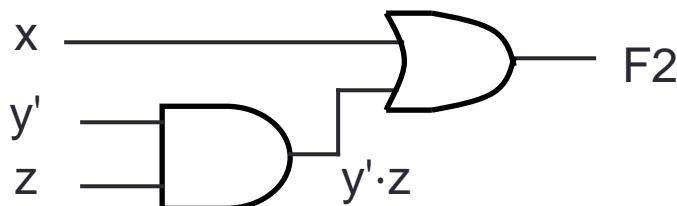
- Given a Boolean expression, we may implement it as a logic circuit.
- Example: $F1 = x \cdot y \cdot z'$ (note the use of a 3-input AND gate)

Every input must be connected in a working circuit.

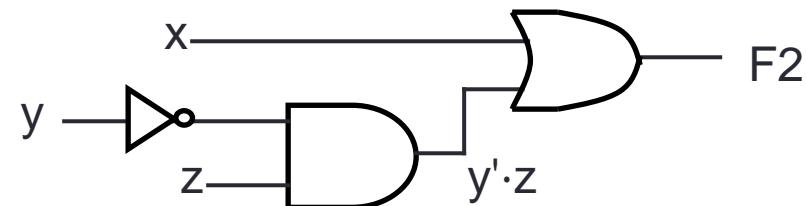


LOGIC CIRCUITS (2/2)

- Example: $F2 = x + y' \cdot z$

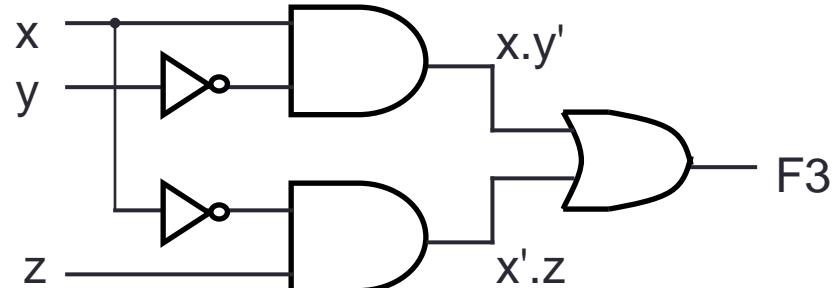
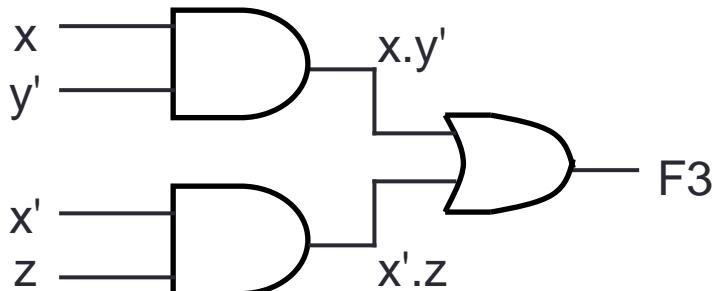


*If complemented literals
are available*



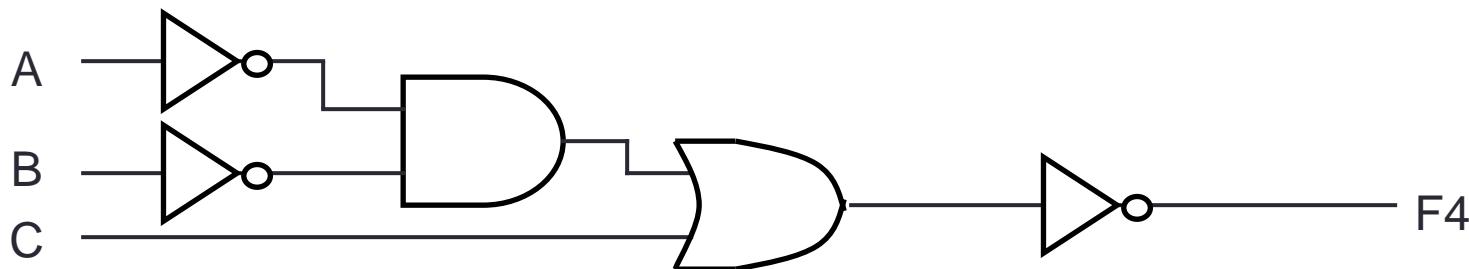
*If complemented literals
are not available*

- Example: $F3 = x \cdot y' + x' \cdot z$



ANALYSING LOGIC CIRCUITS

- Given a logic circuit, we can analyse it to obtain the logic expression.
- Example: Given the logic circuit below, what is the Boolean expression of F_4 ?



$$F_4 = ?$$

QUICK REVIEW QUESTIONS (1)

- DLD page 79
Questions 4-1 to 4-4.

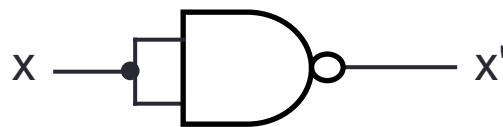


UNIVERSAL GATES

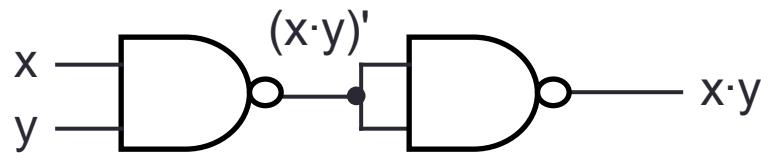
- AND/OR/NOT gates are sufficient for building any Boolean function.
- We call the set {AND, OR, NOT} a **complete set of logic**.
- However, other gates are also used:
 - Usefulness (eg: XOR gate for parity bit generation)
 - Economical
 - Self-sufficient (eg: NAND/NOR gates)

NAND GATE

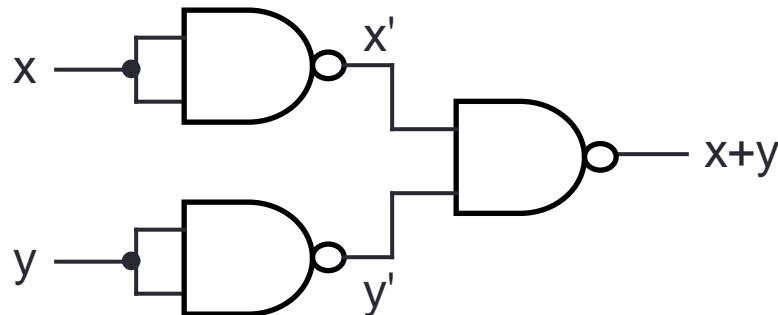
- {NAND} is a complete set of logic.
- Proof: Implement NOT/AND/OR using only NAND gates.



$$(x \cdot x)' = x' \quad (\text{idempotency})$$



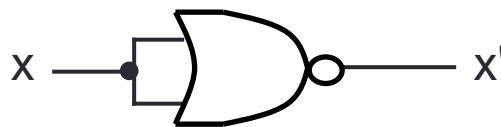
$$\begin{aligned} ((x \cdot y) \cdot (x \cdot y))' &= ((x \cdot y))' && (\text{idempotency}) \\ &= x \cdot y && (\text{involution}) \end{aligned}$$



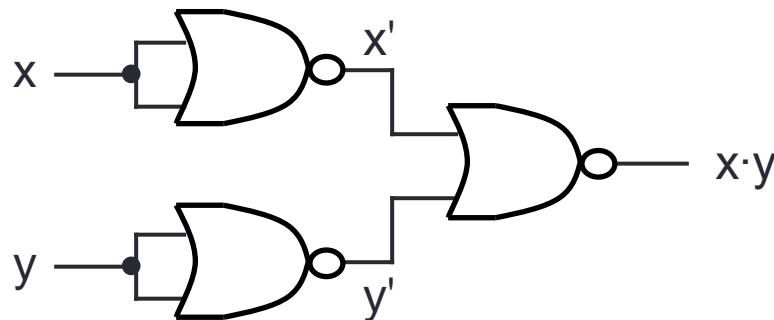
$$\begin{aligned} ((x \cdot x) \cdot (y \cdot y))' &= (x' \cdot y')' && (\text{idempotency}) \\ &= (x')' + (y')' && (\text{DeMorgan}) \\ &= x + y && (\text{involution}) \end{aligned}$$

NOR GATE

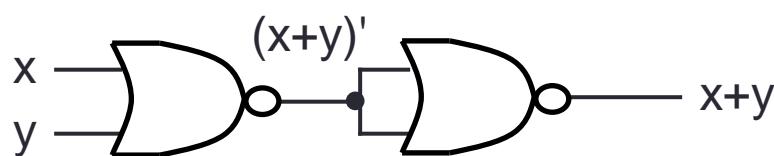
- {NOR} is a complete set of logic.
- Proof: Implement NOT/AND/OR using only NOR gates.



$$(x+x)' = x' \quad (\text{idempotency})$$



$$\begin{aligned} ((x+x)+(y+y))' &= (x'+y')' \quad (\text{idempotency}) \\ &= (x') \cdot (y')' \quad (\text{DeMorgan}) \\ &= x \cdot y \quad (\text{involution}) \end{aligned}$$



$$\begin{aligned} ((x+y)+(x+y))' &= ((x+y)')' \quad (\text{idempotency}) \\ &= x+y \quad (\text{involution}) \end{aligned}$$

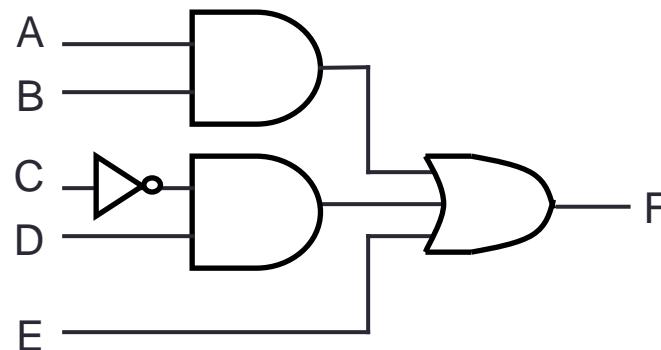
QUICK REVIEW QUESTIONS (2)

- DLD page 79
Questions 4-6 to 4-8.



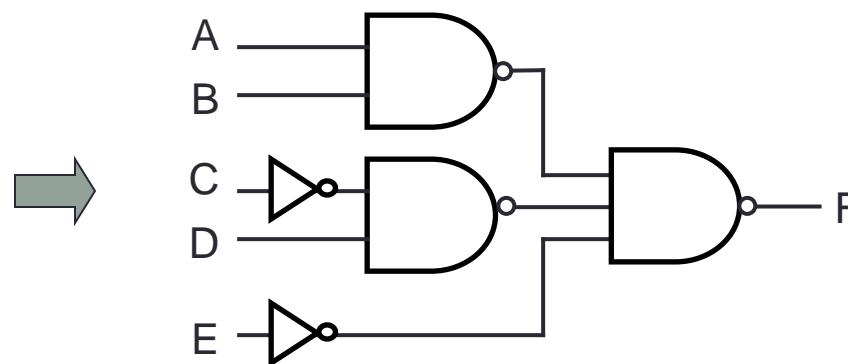
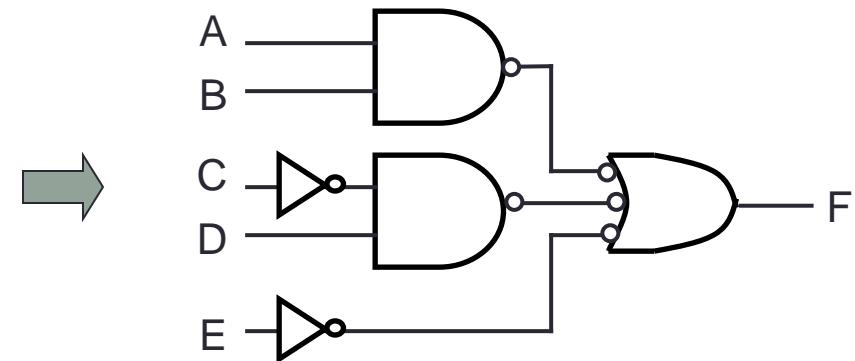
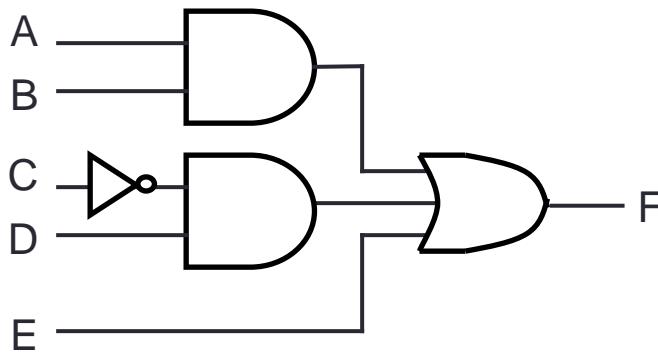
SOP AND NAND CIRCUITS (1/2)

- An SOP expression can be easily implemented using
 - 2-level AND-OR circuit
 - 2-level NAND circuit
- Example: $F = A \cdot B + C' \cdot D + E$
 - Using 2-level AND-OR circuit



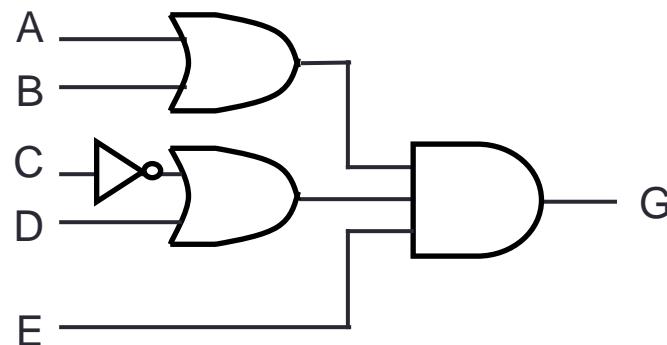
SOP AND NAND CIRCUITS (2/2)

- Example: $F = A \cdot B + C' \cdot D + E$
 - Using 2-level NAND circuit



POS AND NOR CIRCUITS (1/2)

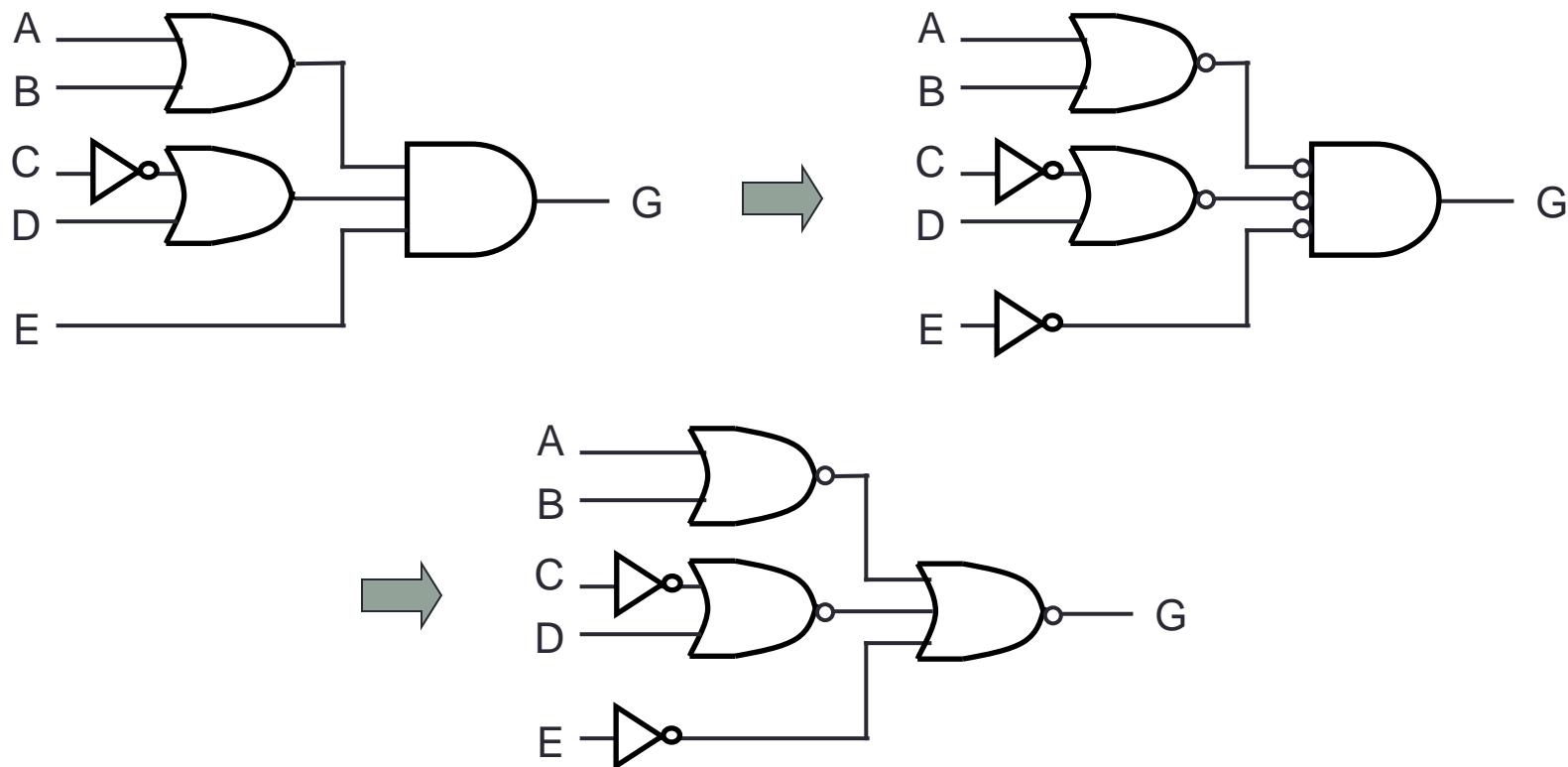
- A POS expression can be easily implemented using
 - 2-level OR-AND circuit
 - 2-level NOR circuit
- Example: $G = (A+B) \cdot (C'+D) \cdot E$
 - Using 2-level OR-AND circuit



POS AND NOR CIRCUITS (2/2)

- Example: $G = (A+B) \cdot (C'+D) \cdot E$

- Using 2-level NOR circuit

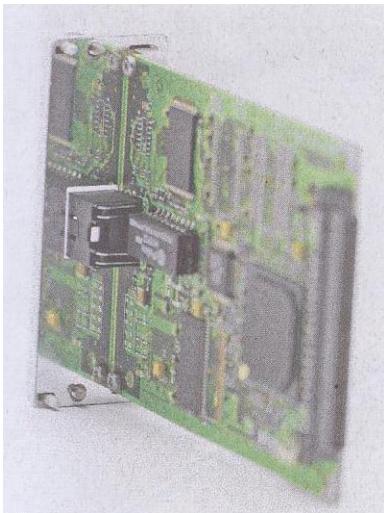


READING ASSIGNMENT

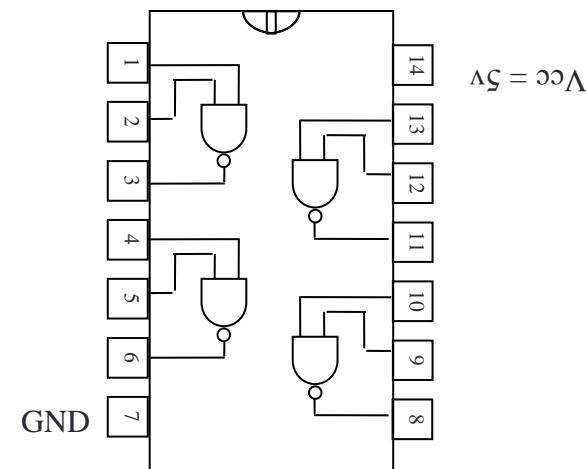
- Propagation Delay
 - Read up DLD section 4.5, pg 75 – 77.
- Integrated Circuit Logic Families
 - Read up DLD section 4.6, pg 77 – 78.



INTEGRATED CIRCUIT (IC) CHIP

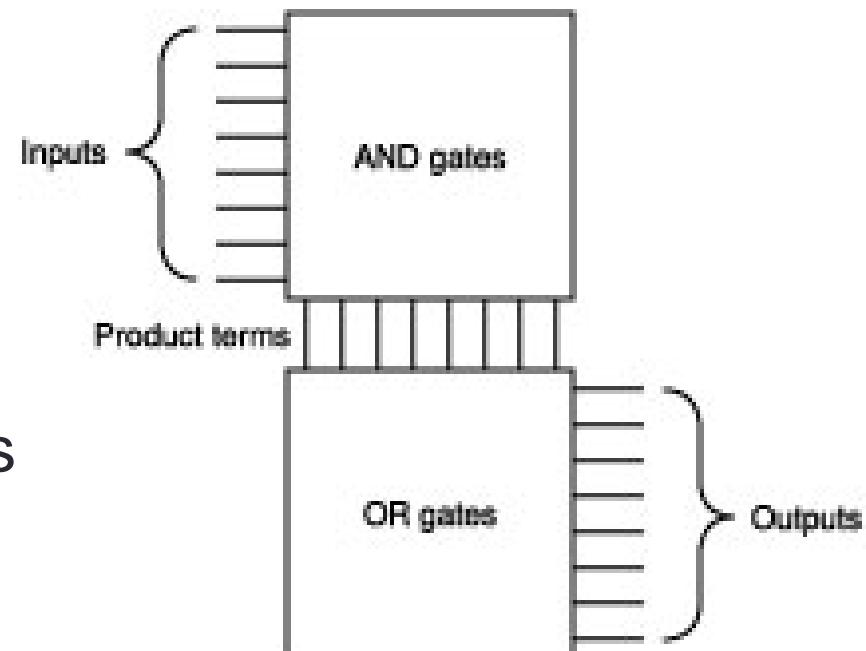


- Example of a **74LS00** chip:
Quad NAND gates.



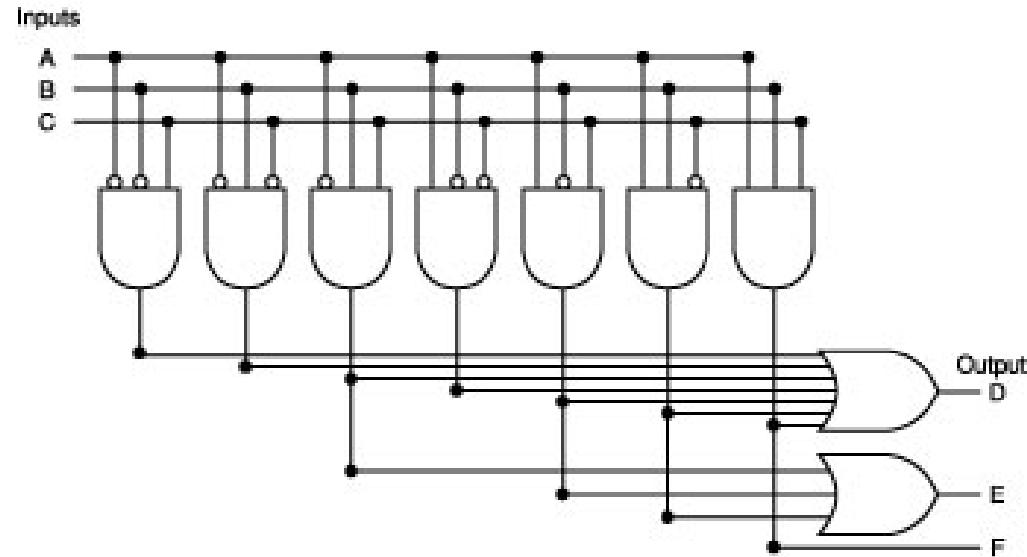
PROGRAMMABLE LOGIC ARRAY

- A programmable integrated circuit – implements sum-of-products circuits (allow multiple outputs).
- 2 stages
 - AND gates = product terms
 - OR gates = outputs
- Connections between inputs and the planes can be ‘burned’.



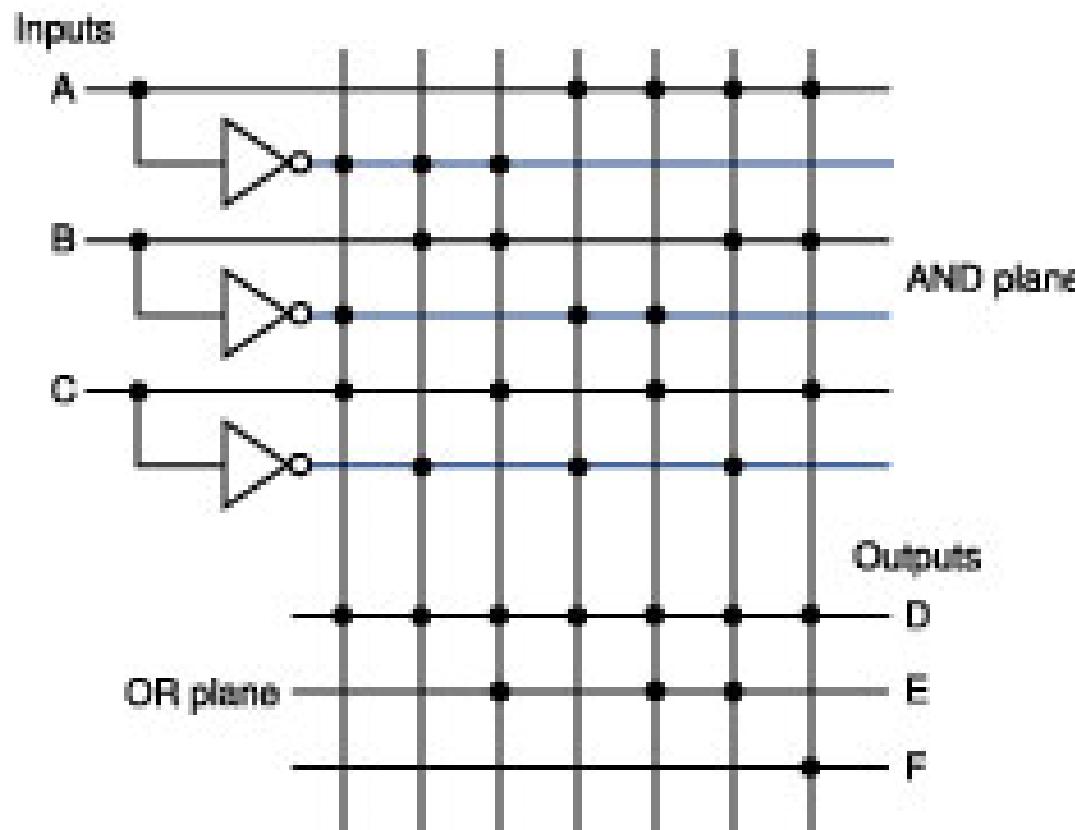
PLA CYLINDIQUE (1/2)

Inputs			Outputs		
A	B	C	D	E	F
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	1	0	0
0	1	1	1	1	0
1	0	0	1	0	0
1	0	1	1	1	0
1	1	0	1	1	0
1	1	1	1	0	1



PLA EXAMPLE (2/2)

- Simplified representation of previous PLA.



READ ONLY MEMORY (ROM)

- Similar to PLA
 - Set of inputs (called addresses)
 - Set of outputs
 - Programmable mapping between inputs and outputs
- Fully decoded: able to implement any mapping.
- In contrast, PLAs may not be able to implement a given mapping due to not having enough minterms.

LAB ASSIGNMENTS (1/3)

- For the first few labs, you will implement simple circuits using the Logic Trainer



LAB ASSIGNMENTS (2/3)

- Lab sheets will be given out in lectures.
- Remember to read the **Lab Guidelines** and **Lab #1 Introductory Lab** before you come for your first lab session.
- For subsequent labs, please read the lab sheet and **fill up as much as you can** before the lab, or you may not have enough time to complete your lab experiment.
- Aim to finish your experiment as quickly as possible. Vacate the room 10 minutes before the hour. If not, just submit your lab report.



LAB ASSIGNMENTS (3/3)



- No make-up lab if you miss a lab. If you have valid reason, you will be given an “EX” (exempt) mark and your CA marks will not suffer. You may submit your medical certificate or other relevant document to your labTA.
- To attend a different group only for that week with valid reason, please seek consent from the lecturer, preferably at least two days in advance.

Q&A