CS2100 COMPUTER ORGANISATION

http://www.comp.nus.edu.sg/~cs2100/

Lecture #14

# Logic Circuits

NUS National University of Singapore | School of Computing

# Lecture #14: Logic Circuits

# 1. Logic Gates

## Gate symbols

Symbol set 1

Symbol set 2
(ANSI/IEEE Standard 91-1984)

AND

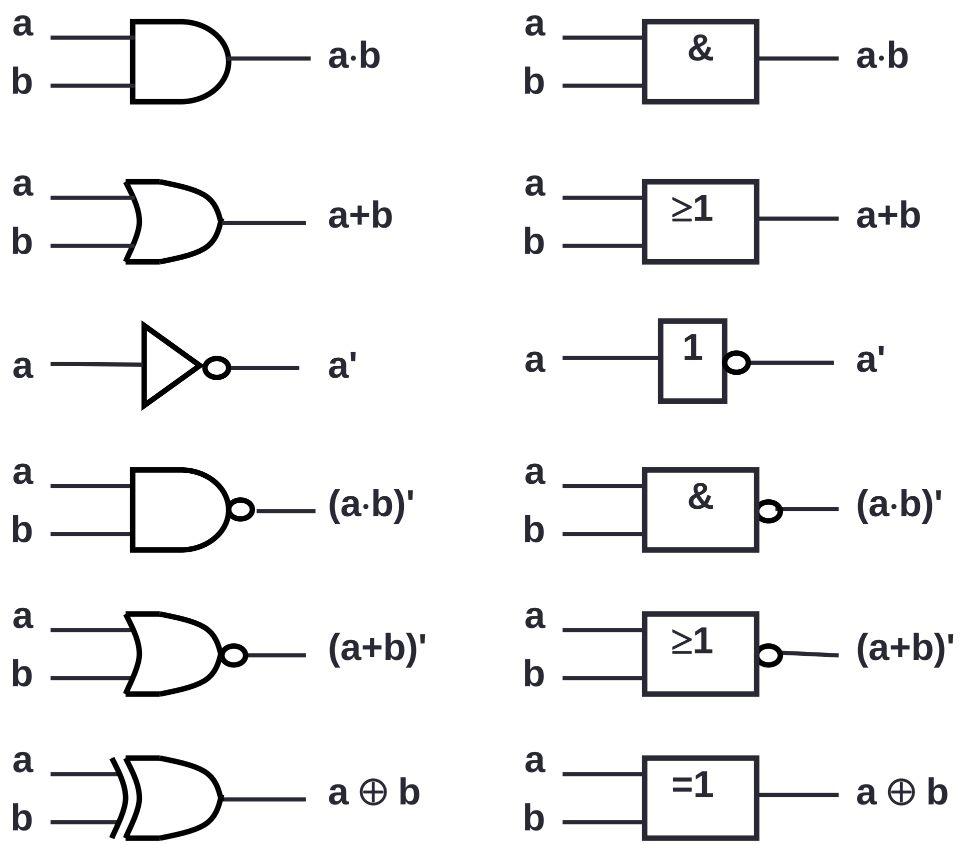
OR

NOT

NAND

NOR

EXCLUSIVE OR

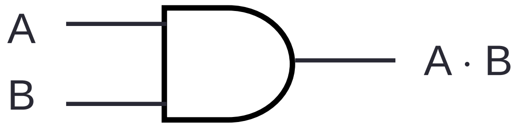


# 1.1 Inverter/AND/OR Gates

- Inverter (NOT gate)

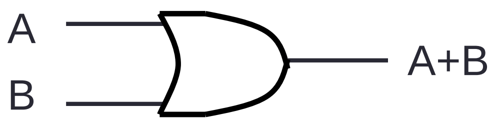

| A | A' |
|---|---|
| 0 | 1 |
| 1 | 0 |

- 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 |

# 1.2 NAND/NOR Gates

- NAND gate


| A | B | (A · B)' |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |


NAND Negative-OR

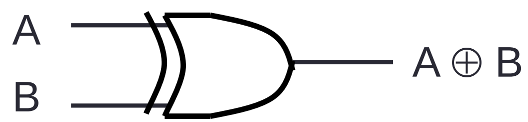
- NOR gate


| A | B | (A + B)' |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |


NOR Negative-AND

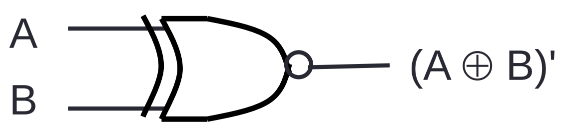
# 1.3 XOR/XNOR Gates

- XOR gate




| A | B | A ⊕ B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

- XNOR gate




| A | B | (A ⊕ B)' |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

XNOR can be represented by ⊙
(Example: A ⊙ B)

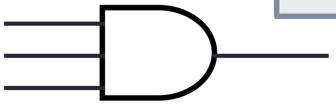
aka equivalent gate

# 2. Logic Circuits (1/2)

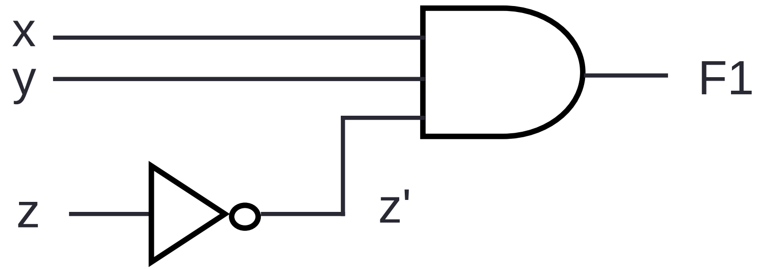
Fan-out = does not mean the gate has more outputs since can only have 1 output

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

Every input must be connected in a working circuit!



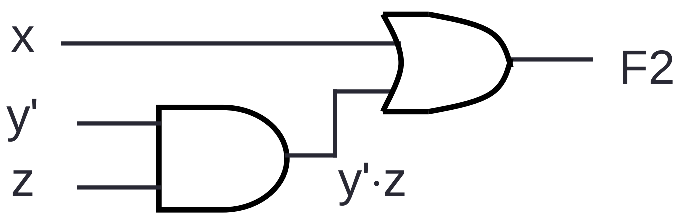
- Given a Boolean expression, we may implement it as a logic circuit.
- Example: F1 = x·y·z' (note the use of a 3-input AND gate)



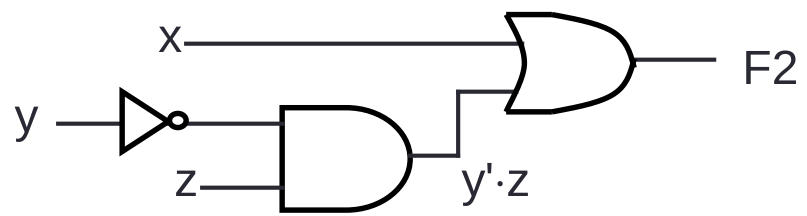
# 2. Logic Circuits (2/2)

is XOR commutative?
check the fan in truth table

- Example: F2 = x + y'·z

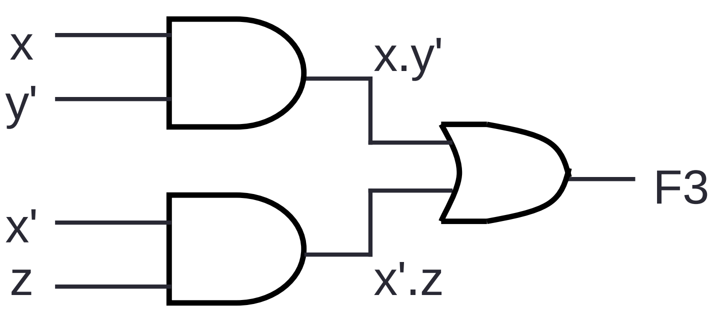



*If complemented literals are available*

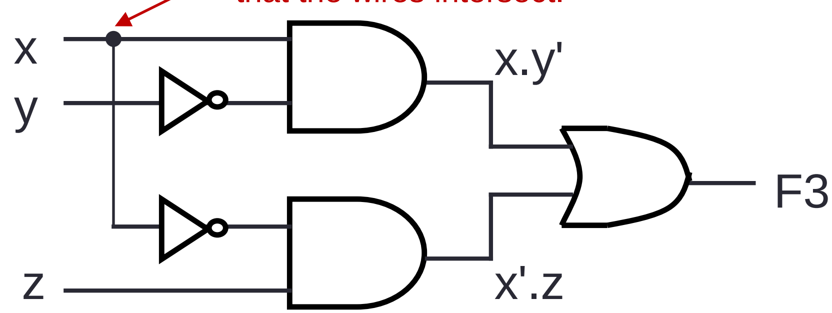



*If complemented literals are not available*

- Example: F3 = x·y' + x'·z

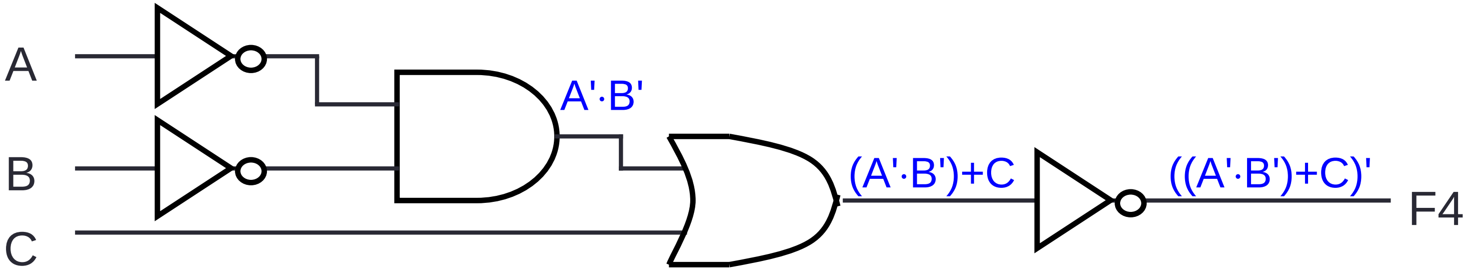



Draw a solid circle to denote that the wires intersect.



# 2.1 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 F4?




F4 = ? $((A'.B')+C)' = (A+B).C'$

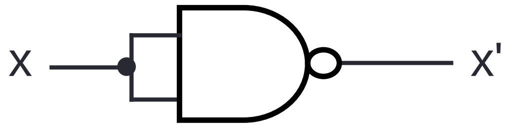
- DLD page79 Quick Review Questions Questions 4-1 to 4-4.

# 3. Universal Gates

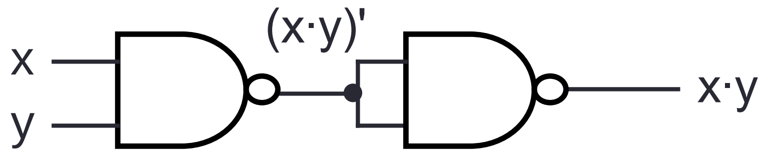
- AND/OR/NOT gates are sufficient for building any Boolean function.
- We call the set {AND, OR, NOT} a complete set of logic.
- However, other gates are also used:
  - Usefulness (eg: XOR gate for parity bit generation)
  - Economical
  - Self-sufficient (eg: NAND/NOR gates)

# 3.1 Universal Gates: NAND Gate

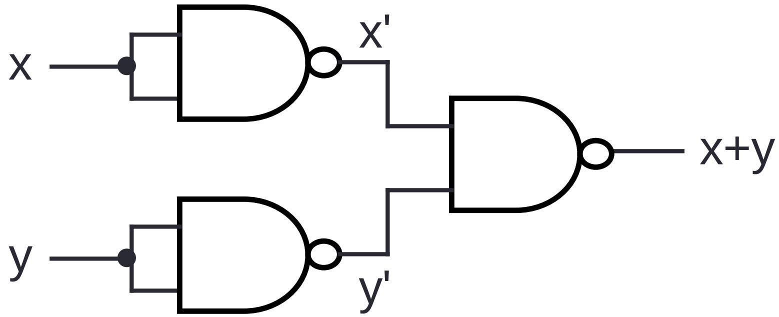
- {NAND} is a complete set of logic.
- Proof: Implement NOT/AND/OR using only NAND gates.




$(x \cdot x)' = x'$ (idempotency)



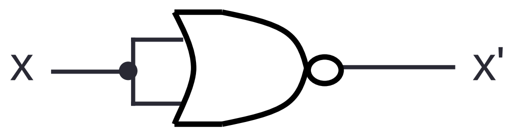

$((x \cdot y)' \cdot (x \cdot y)')' = ((x \cdot y)')'$ (idempotency)
$= x \cdot y$ (involution)



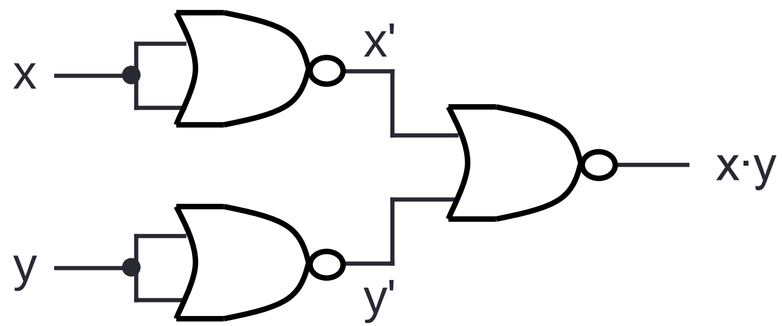

$((x \cdot x)' \cdot (y \cdot y)')' = (x' \cdot y')'$ (idempotency)
$= (x')' + (y')'$ (DeMorgan)
$= x + y$ (involution)

# 3.2 Universal Gates: NOR Gate

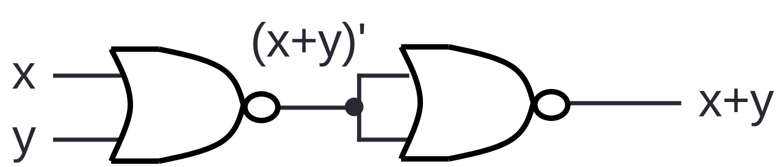
- {NOR} is a complete set of logic.
- Proof: Implement NOT/AND/OR using only NOR gates.




$(x+x)' = x'$ (idempotency)




$((x+x)'+(y+y)')' = (x'+y')'$ (idempotency)
$= (x')'\cdot(y')'$ (DeMorgan)
$= x\cdot y$ (involution)

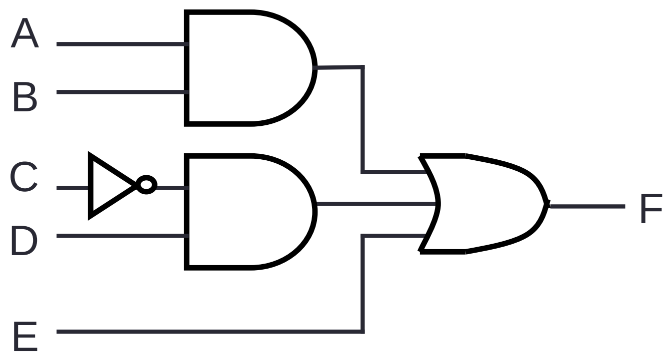



$((x+y)'+(x+y)')' = ((x+y)')'$ (idempotency)
$= x+y$ (involution)

- DLD page79 Quick Review Questions Questions 4-6 to 4-8.

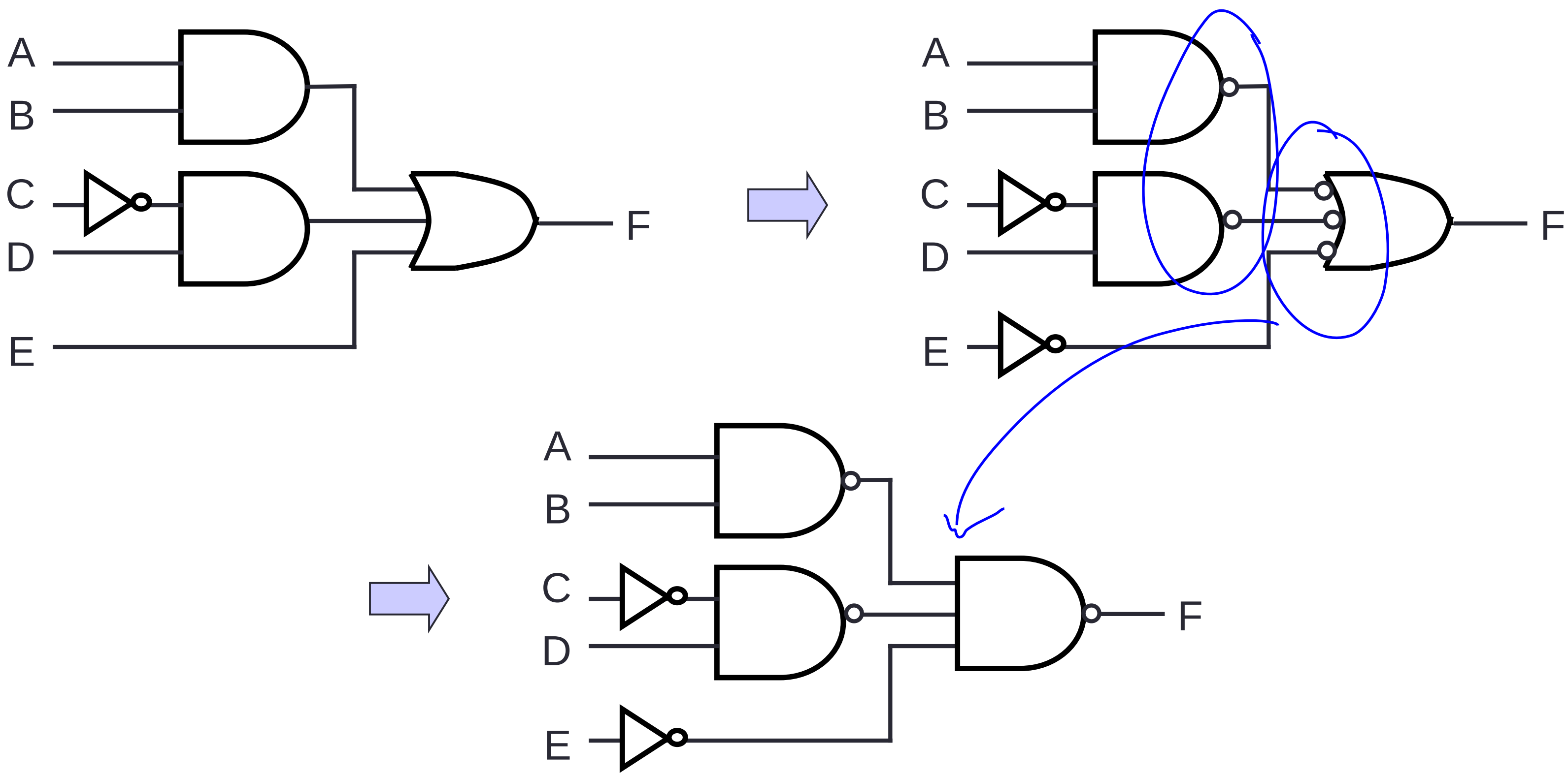
# 3.3 SOP and NAND Circuits (1/2)

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



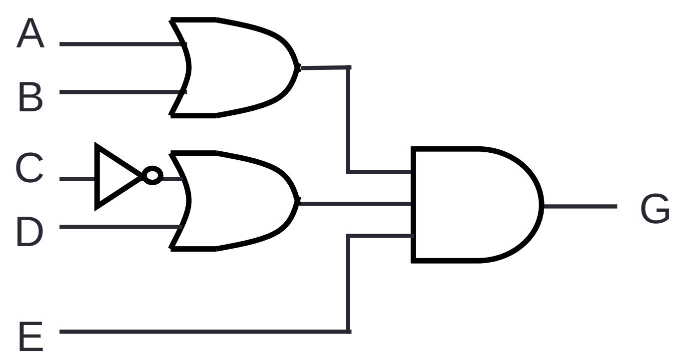
# 3.3 SOP and NAND Circuits (2/2)

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



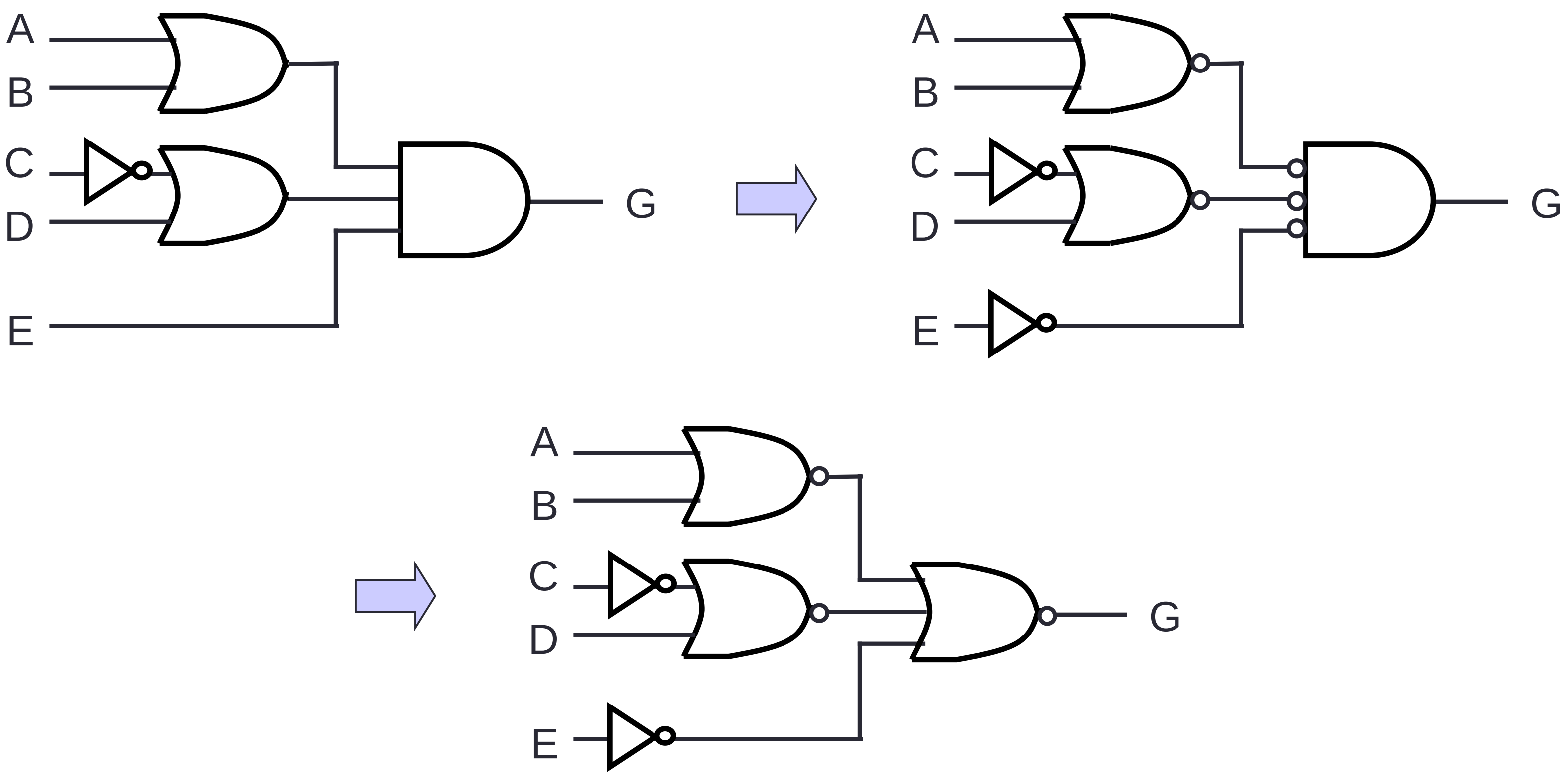
# 3.4 POS and NOR Circuits (1/2)

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



# 3.4 POS and NOR Circuits (2/2)

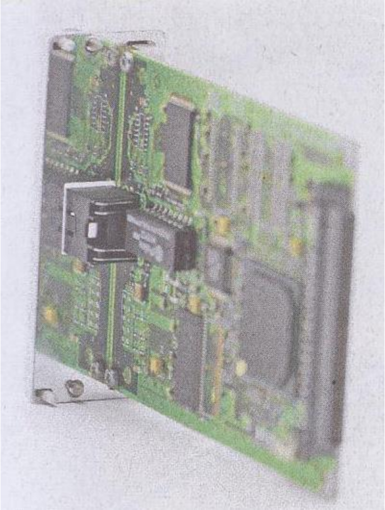
- Example: G = (A+B) · (C'+D) · E
  - Using 2-level NOR circuit



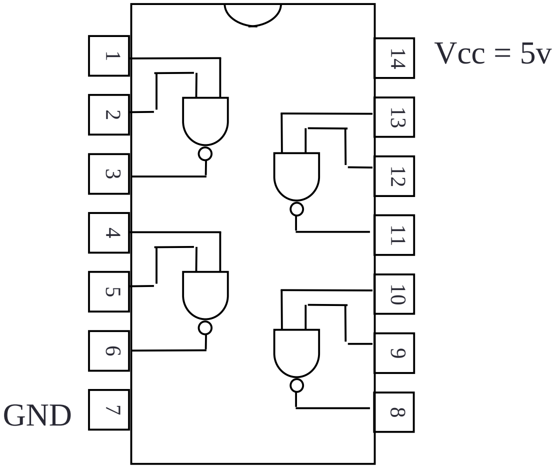
# Reading

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

# 4. Integrated Circuit (IC) Chip

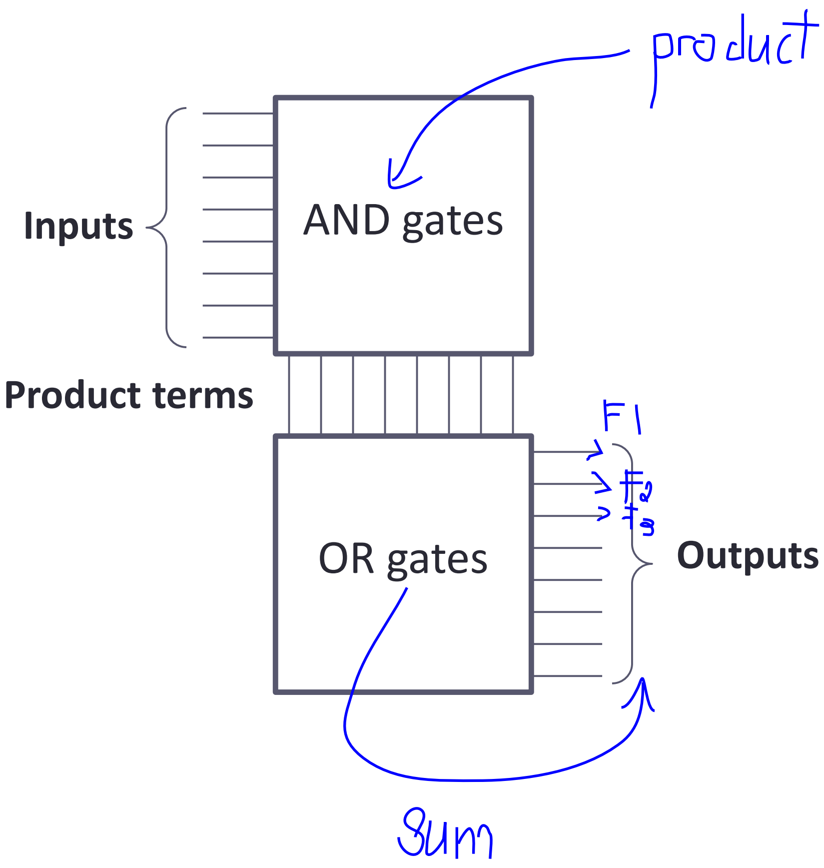


- Example of a 74LS00 chip: Quad NAND gates.



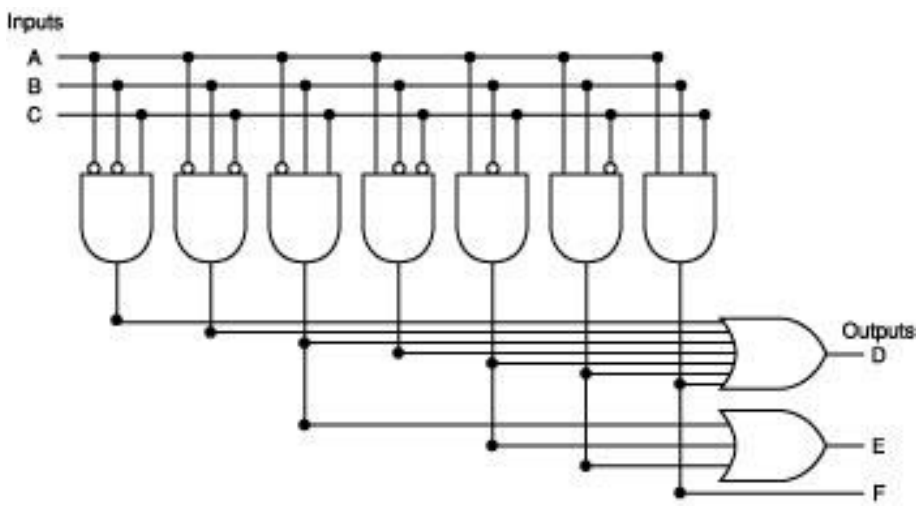
# 5. Programming Logic Array (PLA) (1/3)

- 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'.



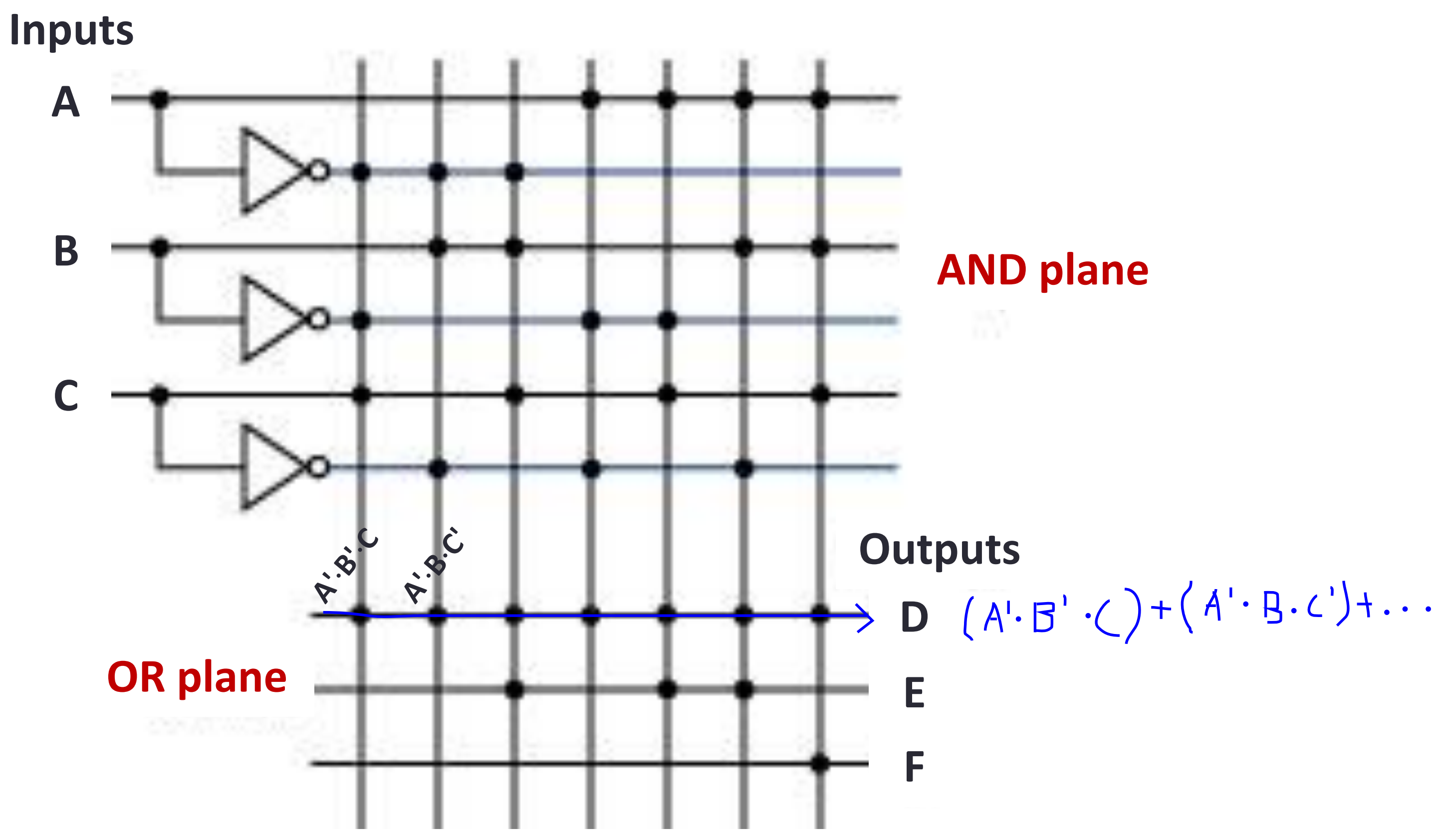
# 5. PLA Example (2/3)

| 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 |



# 5. PLA Example (3/3)

- Simplified representation of previous PLA.



# 6. 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 (1/2)

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

# Lab Assignments (2/2)


- Lab sheets will be given out in lectures.
- Remember to read the Logic Lab Guidelines <u>before</u> you come for your first lab session.
- Please read the lab sheet and fill up as much as you can <u>before the lab</u>, or you may not have enough time to complete your lab experiment.
- Aim to finish your experiment as quickly as possible. Vacate the room <u>10 minutes before the hour</u>. If not, just submit your lab report.

# End of File