# CS2100 - L17 - Logic Gates & Circuits

Week 10

17.1 - Logic gates

17.2 - Universal gates

17.3 - Standard forms

- Terminology
- SOP and POS

17.4 - Minterms & maxterms

- Sum of minterms
- Product of maxterms

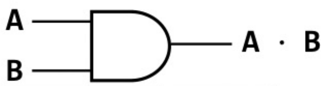
17.1 - Logic gates

# Inverter / AND / OR Gates

## Inverter (NOT gate)

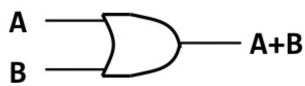

| A | A' |
|---|---|
| 0 | 1 |
| 1 | 0 |

## AND gate




| A | B | A · B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

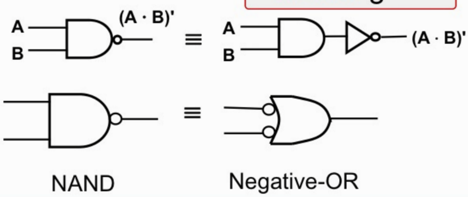
## OR gate




| A | B | A + B |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

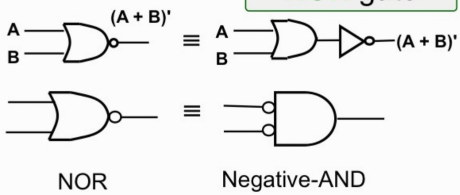
# NAND/NOR Gates

## NAND gate




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

## NOR gate

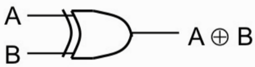



| A | B | (A + B)' |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |



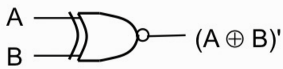
# XOR/XNOR Gates

## XOR gate




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

## XNOR gate



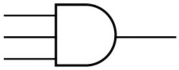

XNOR can be represented by ⊙ (Example: A ⊙ B)

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

[ L17 - AY2021S1 ]

17.2 - Universal gates

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

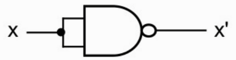


## Universal Gates

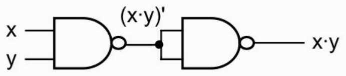
- { NOT, AND, OR } gates are sufficient for building **any** Boolean function
  - Also known as **set of universal gates**
- There are other sets of universal gate:
  - { NAND } (i.e. 1 gate is suffice)
  - { NOR }

# { NAND } is a Universal Gate

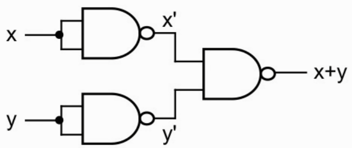
- **Proof:**
  Implement **NOT** / **AND** / **OR** using only **NAND** gates




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




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



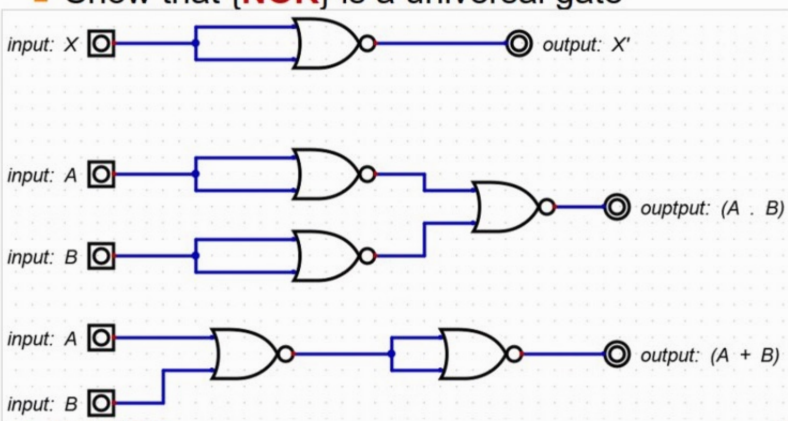

$((x\cdot x)'\cdot(y\cdot y)')' = (x'\cdot y')'$ (idempotency)
$= (x')'+(y')'$ (DeMorgan)
$= x+y$ (involution)



***idea: duality: NOR and NAND gate are dual forms of each other!?***
***todo: do this direct mirror image of the NAND gate***

# EX: NOR Gate

- Show that {**NOR**} is a universal gate



17.3 - Standard forms

# Standard Forms

- Types of Boolean expressions that lead to simple circuit translation
- **Terminology**:

| | |
|---|---|
| Literals | • A Boolean variable on its own or in its complemented form<br>• Examples: **x, x', y, y'** |
| Product term | • A single literal or a logical product (AND) of several literals<br>• Examples: **x, x·y·z', A'·B, A·B, d·g'·v·w** |
| Sum term | • A single literal or a logical sum (OR) of several literals<br>• Examples: **x, x+y+z', A'+B, A+B, c+d+h'+j** |



# SOP and POS Standard Forms

| | |
|---|---|
| Sum-of-Products (SOP) | • A product term or a logical sum (OR) of several product terms<br>• Examples: **x, x + y·z', x·y' + x'·y·z, A·B + A'·B', A + B'·C + A·C' + C·D** |
| Product-of-Sums (POS) | • A sum term or a logical product (AND) of several sum terms<br>• Examples: **x, x·(y+z'), (A+B)·(A'+B'), (A+B+C)·D'·(B'+D+E')** |

- All Boolean expression can be expressed in **SOP** or **POS**

## SOP → 2-Level AND-OR Circuit

- An **SOP** expression can be easily implemented using
  1. 2-level AND-OR circuit
  2. 2-level NAND circuit

1st level: construct each product term
2nd level: pluck all into OR gate

## POS → 2-Level OR-AND Circuit

Mirror image of SOP

- A POS expression can be easily implemented using
  1. 2-level OR-AND circuit
  2. 2-level NOR circuit

1st level: construct each sum term
2nd level: pluck all into AND gate

17.4 - Minterms & maxterms

# Canonical Forms

- Canonical / normal form: a **unique form** of representation

Specialized form of SOP

1. **Sum-of-minterms** = Canonical sum-of-products

2. **Product-of-maxterms** = Canonical product-of-sums

Specialized form of POS

# Minterm & Maxterm: Definition

**Minterm**

- A **minterm** of *n* variables is a product term that contains ***n*** **literals** from all the variables.
- Example: On 2 variables **x** and **y**, the minterms are: **x'·y'**, **x'·y**, **x·y'** and **x·y**

**Maxterm**

- A maxterm of *n* variables is a sum term that contains ***n*** **literals** from all the variables.
- Example: On 2 variables **x** and **y**, the maxterms are: **x'+y'**, **x'+y**, **x+y'** and **x+y**

- In general, with ***n*** **variables** we have $\mathbf{2^n}$ **minterms** and $\mathbf{2^n}$ **maxterms**

# Minterm & Maxterm: Terminology

| x | y | Minterms | | Maxterms | |
|---|---|---|---|---|---|
| | | Term | Notation | Term | Notation |
| 0 | 0 | $x' \cdot y'$ | m0 | $x+y$ | M0 |
| 0 | 1 | $x' \cdot y$ | m1 | $x+y'$ | M1 |
| 1 | 0 | $x \cdot y'$ | m2 | $x'+y$ | M2 |
| 1 | 1 | $x \cdot y$ | m3 | $x'+y'$ | M3 |

- Each minterm is the complement of the corresponding maxterm
  - E.g.: $m2 = x \cdot y'$
    $m2' = (x \cdot y')' = x' + (y')' = x' + y = M2$



# Conversion

- $F2 = \Sigma m(1, 4, 5, 6, 7)$
  $= \Pi M(0, 2, 3)$
  (Why? How?)

| x | y | z | F2 |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 |
| 1 | 1 | 1 | 1 |

- $F2' = m0 + m2 + m3$

- Therefore,
  $F2 = (m0 + m2 + m3)'$
  $= m0' \cdot m2' \cdot m3'$ (*DeMorgan's*)
  $= M0 \cdot M2 \cdot M3$ (*mx' =Mx*)