Introducción al Diseño Lógico (E0301)

Ingeniería en Computación

Gerardo E. Sager

Clase 6 curso 2021

Introducción al Diseño Lógico

- *Temas que se tratan*
  - Uso de compuertas para implementar circuitos lógicos
  - Compuertas universales (NAND / NOR)
  - Ejemplos

Formas canónicas

- Se definen dos formas canónicas de expresión booleana
  - Conjunción de maxitérminos o “Producto de Sumas”
    - Ejemplo (A+\B)(\A+ B)
  - Disyunción de minitérminos o “Suma de Productos"
    - Ejemplo A\B + \AB
- Podemos llevar cualquier expresión booleana a una forma canónica, operando sobre ella mediantes los teoremas de Boole y De Morgan.

Expresiones Canónicas

Implementación de funciones lógicas

- Vimos que las expresiones lógicas se pueden expresar en las formas canónicas que llamamos “suma de productos” o “producto de sumas”.
- En el caso de “suma de productos” esto significa que tenemos una serie de “productos” (minitérminos) que se agrupan mediante la operación OR (“suma”) de todos ellos.

  Un minitérmino esta formado por la operación AND sobre distintas variables, que pueden aparecer en forma directa o negada.
- **Ejemplos:** $x = A.B + \overline{A}.\overline{B}$ $z = \overline{A}.B.C + A.\overline{B}.C + A.B.\overline{C}$

Expresiones Canónicas

Implementación de funciones lógicas

- En el caso de “producto de sumas” esto significa que tenemos una serie de “sumas” (maxitérminos) que se agrupan mediante la operación AND (“producto”) de todos ellos.

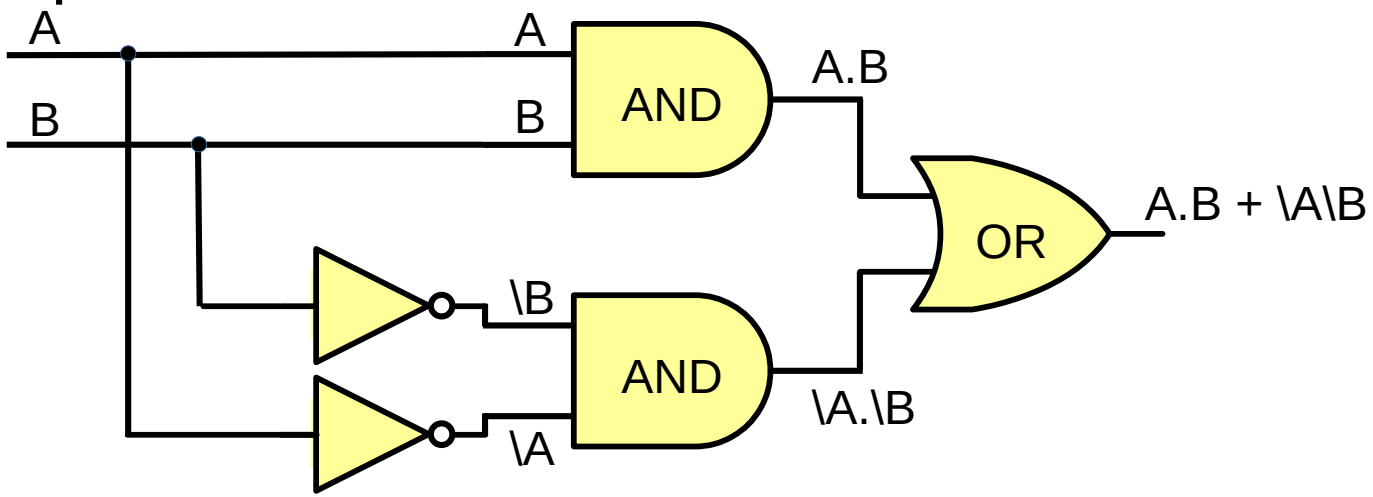
  Un maxitérmino esta formado por la operación OR sobre distintas variables, que pueden aparecer en forma directa o negada.

- Ejemplos: $y = (A + B).(\overline{A} + \overline{B})$

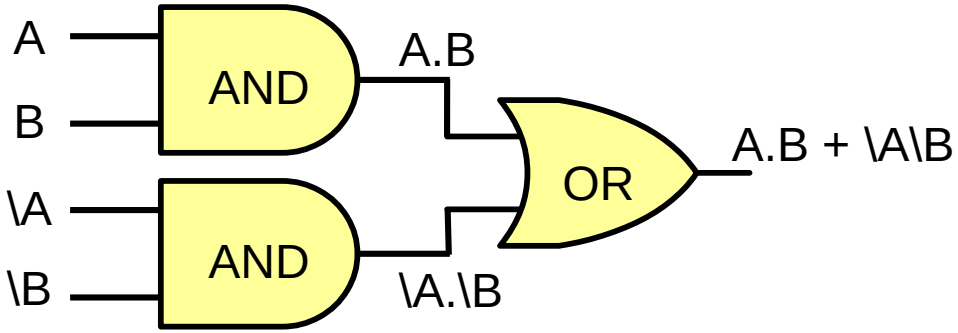
$$w = (\overline{A} + B + C).(A + \overline{B} + C).(A + B + \overline{C})$$

Implementación con compuertas

- Implementemos $x = A.B + \overline{A}.\overline{B}$

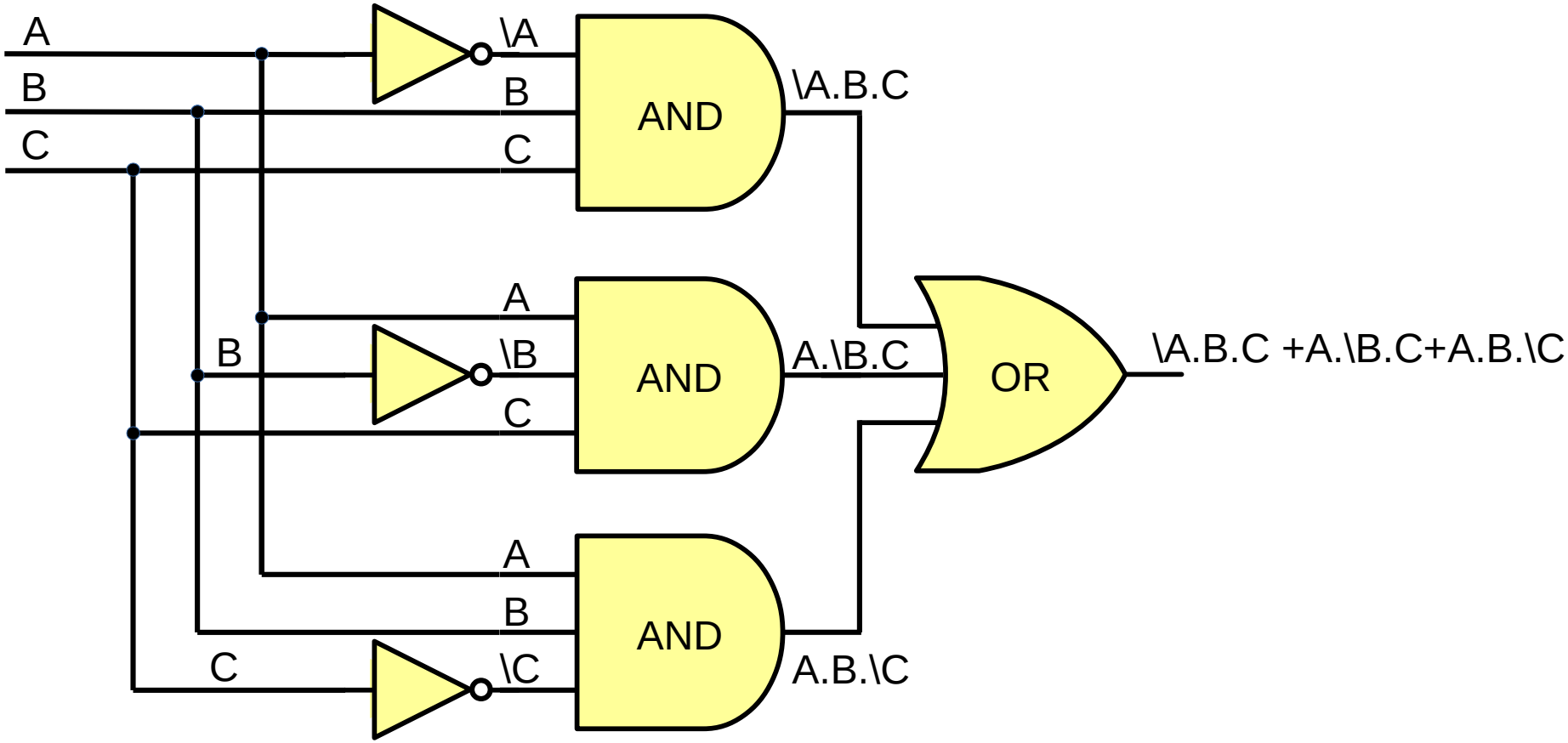


- Podemos ver que se utilizan los tres niveles mencionados.
- A veces se consideran disponibles tanto las variables como sus versiones negadas, en este caso se utilizan sólo dos niveles



Implementación con compuertas

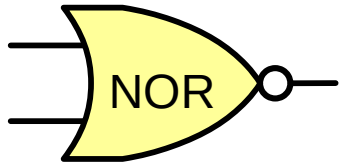
- Ahora, implementemos $z = \overline{A}.B.C + A.\overline{B}.C + A.B.\overline{C}$



- Podemos ver que también se utilizan los tres niveles mencionados.

Compuertas universales

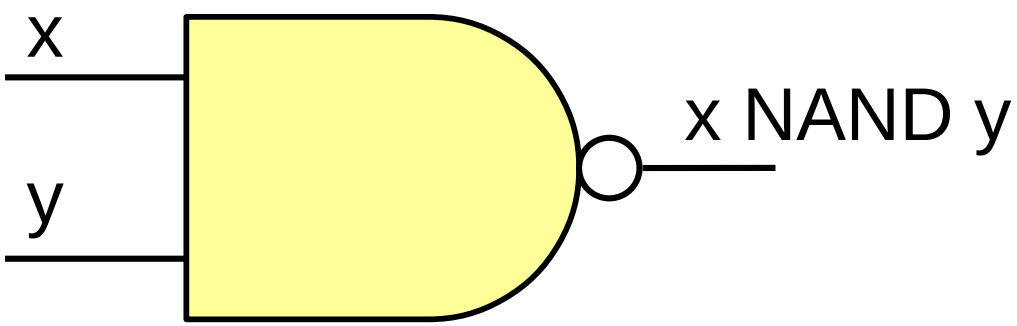
- Las compuertas NAND y NOR permiten implementar cualquier expresión lógica al conectarlas adecuadamente. Por eso se llaman Compuertas Universales.
- Durante la década del 70 y principios de los 80 se utilizaron masivamente para la implementación de sistemas digitales.
- Se distinguen facilmente de las compuertas OR y AND porque a la salida tienen un círculo que simboliza la inversión de la salida



NAND

- **La compuerta NAND es una compuerta AND con la salida invertida.**

$$x\ NAND\ y = \overline{x\ .\ y}$$

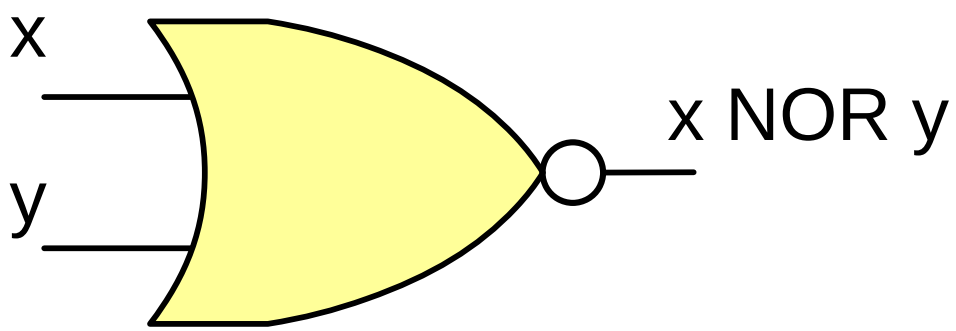


x	y	NAND
0	0	1
0	1	1
1	0	1
1	1	0

NOR

- **La compuerta NOR es una compuerta OR con la salida invertida.**

$$x\ NOR\ y = \overline{x + y}$$



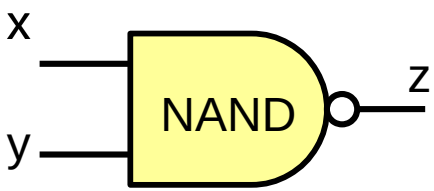
x	y	NOR
0	0	1
0	1	0
1	0	0
1	1	0

Universalidad de NAND y NOR

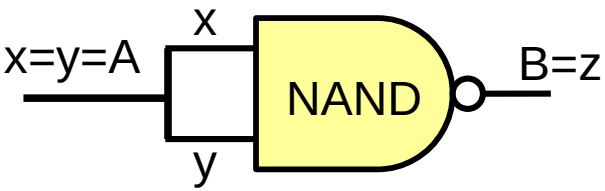
- ¿Por qué decimos que NAND y NOR son compuertas universales?
- Porque se puede implementar cualquier expresión logica utilizando exclusivamente ya sea compuertas NOR, o bien compuertas NAND.
- Para ello debo verificar que puedo construir las tres expresiones lógicas básicas (NOT, OR, AND) a partir de ellas

Universalidad de NAND

- NOT a partir de NAND: Dos maneras distintas

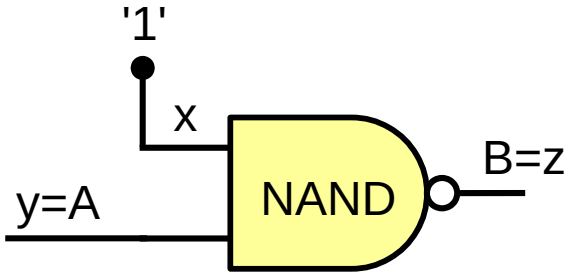


x	y	z
0	0	1
0	1	1
1	0	1
1	1	0



A	x	y	z	B
0	0	0	1	1
	0	1	1	
	1	0	1	
1	1	1	0	0

Como x e y no pueden ser distintos las filas están en rojo

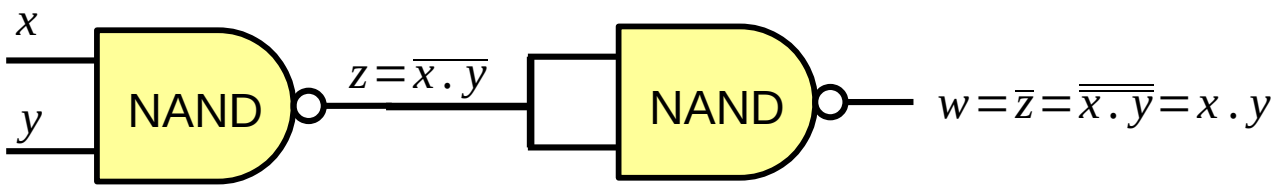


A	x	y	z	B
	0	0	1	
	0	1	1	
0	1	0	1	1
1	1	1	0	0

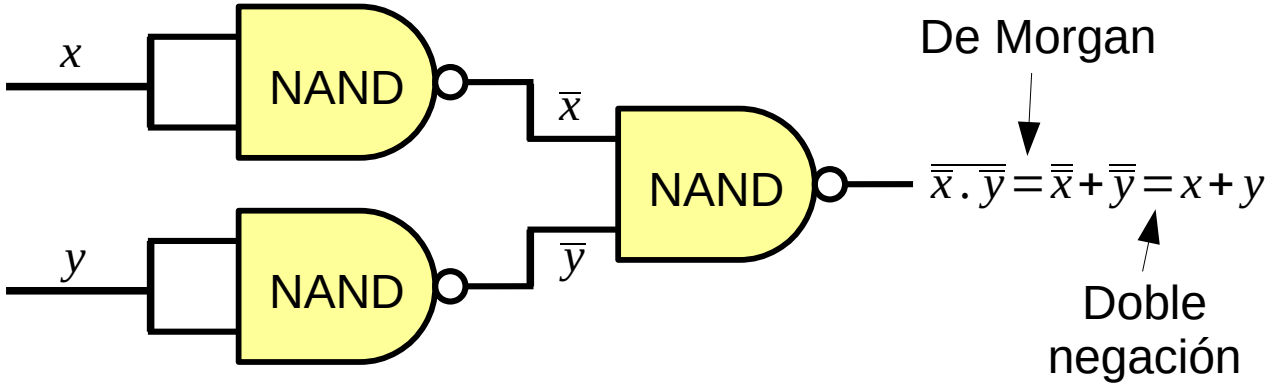
Como x no puede ser '0' las filas están en rojo

Universalidad de NAND

- AND a partir de NAND



- OR a partir de NAND

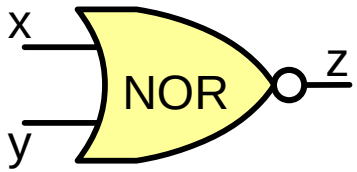


- Podemos implementar las operaciones básicas NOT, AND y OR con la compuerta NAND.

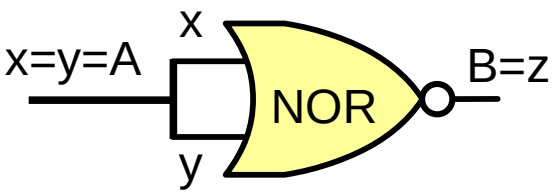
NAND es una compuerta universal

Universalidad de NOR

- NOT a partir de NOR: Dos maneras distintas

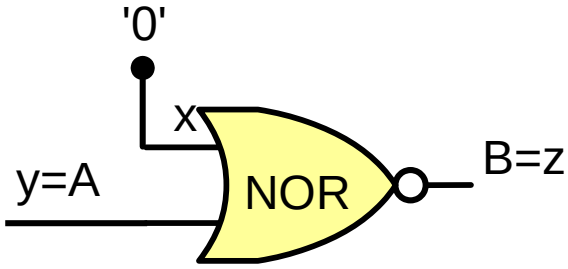


x	y	z
0	0	1
0	1	0
1	0	0
1	1	0



A	x	y	z	B
0	0	0	1	1
	0	1	1	
	1	0	1	
1	1	1	0	0

Como x e y no pueden ser distintos las filas están en rojo

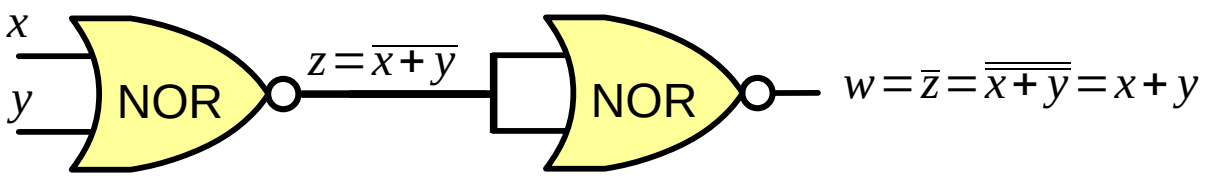


A	x	y	z	B
0	0	0	1	1
0	0	1	1	1
	1	0	1	
	1	1	0	

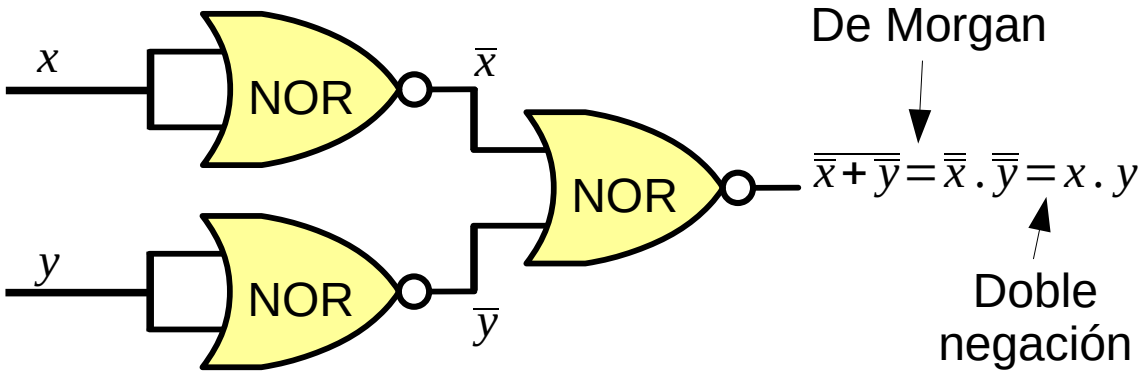
Como x no puede ser '1' las filas están en rojo

Universalidad de NAND

- OR a partir de NOR



- AND a partir de NOR

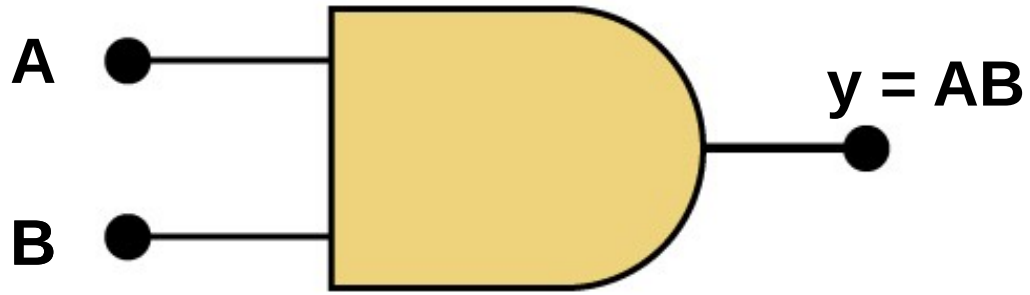


- Podemos implementar las operaciones básicas NOT, AND y OR con la compuerta NOR.

NOR es una compuerta universal

Computadora vs Compuerta

Ej. Comparar la operación de una computadora y un circuito lógico para realizar *y = AB.*

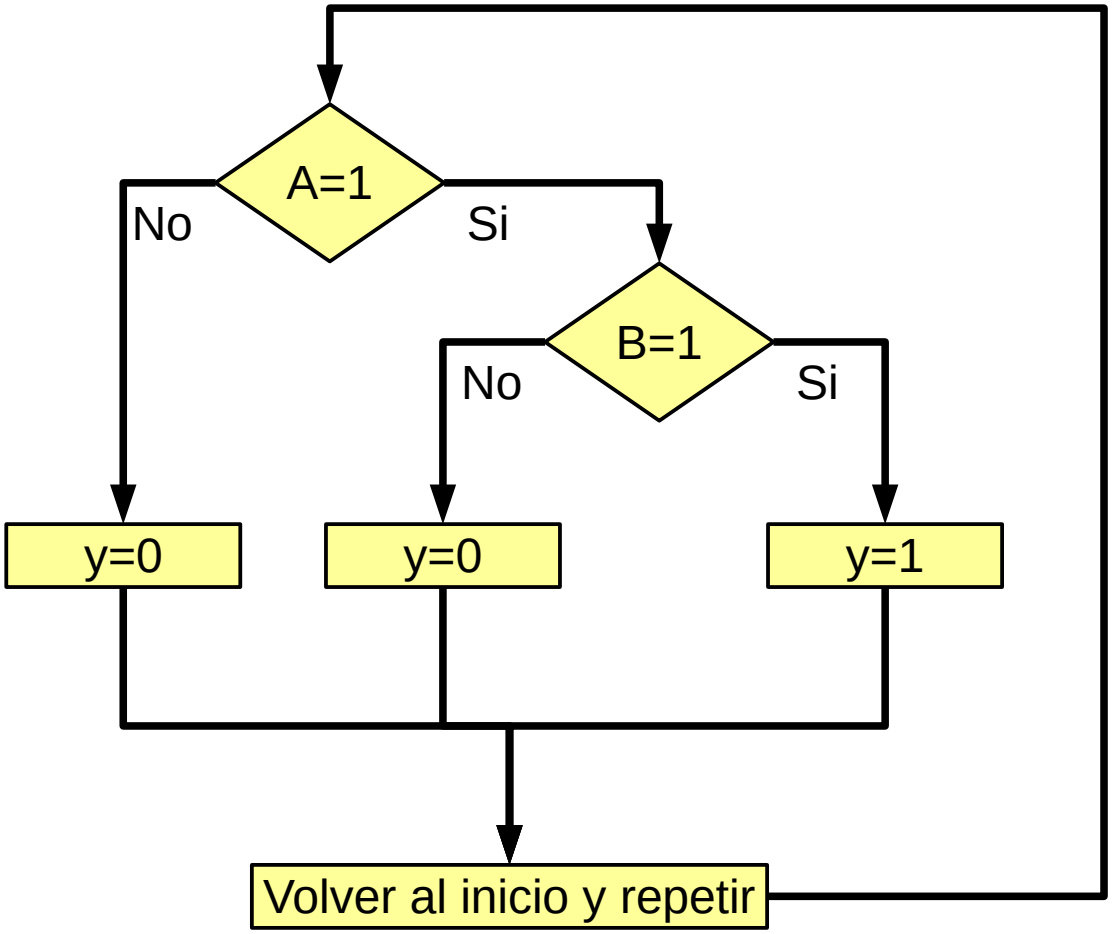


El circuito lógico es una compuerta AND. La salida y estará en HIGH unos 10 ns después que A y B estén en HIGH simultáneamente.

Luego de aproximadamente 10 ns después que cualquiera de las entradas vaya a LOW, la salida y irá a LOW.

Computadora vs Compuerta

Comparación de la operación de una computadora y un circuito lógico al realizar la operación lógica *y = AB.*



Cada forma en el diagrama de flujo representa una operación

Si cada una lleva 20 ns, va a llevar un mínim de dos o tres instrucciones (40-60 ns) responder a los cambios en las entradas.

Introducción al Diseño Lógico (E0301)

Ingeniería en Computación

Gerardo E. Sager

Clase 6 curso 2021

Introducción al Diseño Lógico

- *Temas que se tratan*
  - Uso de compuertas para implementar circuitos lógicos
  - Compuertas universales (NAND / NOR)
  - Ejemplos

Formas canónicas

- Se definen dos formas canónicas de expresión booleana
  - Conjunción de maxitérminos o “Producto de Sumas”
    - Ejemplo (A+\B)(\A+ B)
  - Disyunción de minitérminos o “Suma de Productos"
    - Ejemplo A\B + \AB
- Podemos llevar cualquier expresión booleana a una forma canónica, operando sobre ella mediantes los teoremas de Boole y De Morgan.

Expresiones Canónicas

Implementación de funciones lógicas

- Vimos que las expresiones lógicas se pueden expresar en las formas canónicas que llamamos “suma de productos” o “producto de sumas”.
- En el caso de “suma de productos” esto significa que tenemos una serie de “productos” (minitérminos) que se agrupan mediante la operación OR (“suma”) de todos ellos.

  Un minitérmino esta formado por la operación AND sobre distintas variables, que pueden aparecer en forma directa o negada.
- **Ejemplos:** $x = A.B + \overline{A}.\overline{B} \quad z = \overline{A}.B.C + A.\overline{B}.C + A.B.\overline{C}$

Expresiones Canónicas

Implementación de funciones lógicas

- En el caso de “producto de sumas” esto significa que tenemos una serie de “sumas” (maxitérminos) que se agrupan mediante la operación AND (“producto”) de todos ellos.

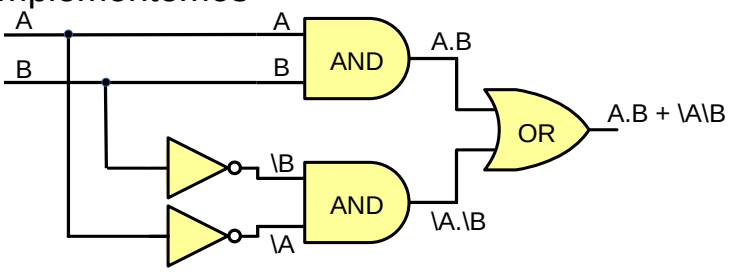
  Un maxitérmino esta formado por la operación OR sobre distintas variables, que pueden aparecer en forma directa o negada.

- Ejemplos: $y = (A + B).(\overline{A} + \overline{B})$

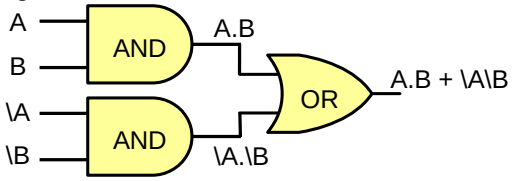
  $w = (\overline{A} + B + C).(A + \overline{B} + C).(A + B + \overline{C})$

Implementación con compuertas

- Implementemos $x = A.B + \overline{A}.\overline{B}$

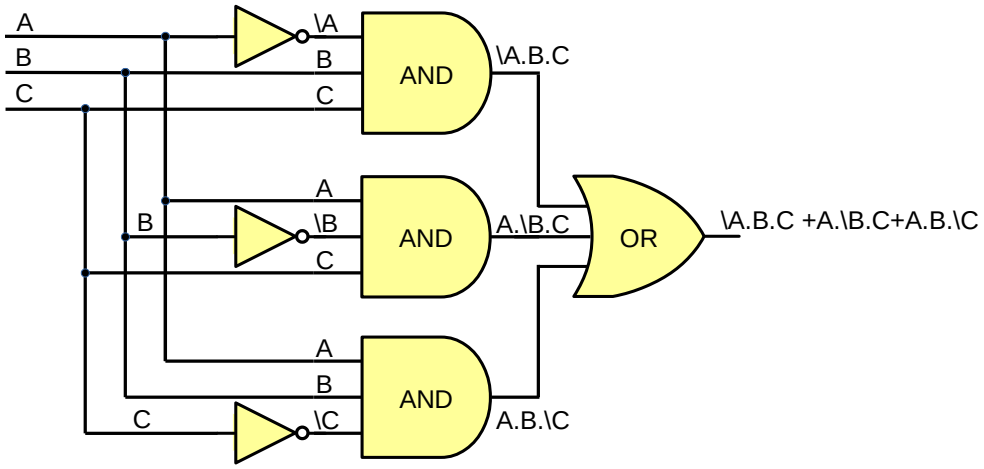


- Podemos ver que se utilizan los tres niveles mencionados.
- A veces se consideran disponibles tanto las variables como sus versiones negadas, en este caso se utilizan sólo dos niveles



Implementación con compuertas

- Ahora, implementemos $z = \overline{A}.B.C + A.\overline{B}.C + A.B.\overline{C}$



  - Podemos ver que también se utilizan los tres niveles mencionados.

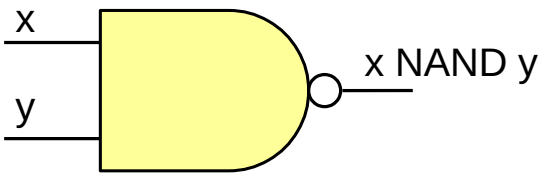
Compuertas universales

- Las compuertas NAND y NOR permiten implementar cualquier expresión lógica al conectarlas adecuadamente. Por eso se llaman Compuertas Universales.
- Durante la década del 70 y principios de los 80 se utilizaron masivamente para la implementación de sistemas digitales.
- Se distinguen facilmente de las compuertas OR y AND porque a la salida tienen un círculo que simboliza la inversión de la salida

NAND

- **La compuerta NAND es una compuerta AND con la salida invertida.**

$$x\ NAND\ y = \overline{x\,.\,y}$$

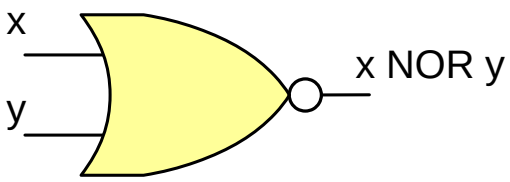


x	y	NAND
0	0	1
0	1	1
1	0	1
1	1	0

NOR

- **La compuerta NOR es una compuerta OR con la salida invertida.**

$$x\ NOR\ y = \overline{x + y}$$



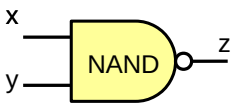
x	y	NOR
0	0	1
0	1	0
1	0	0
1	1	0

Universalidad de NAND y NOR

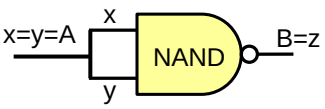
- ¿Por qué decimos que NAND y NOR son compuertas universales?
- Porque se puede implementar cualquier expresión logica utilizando exclusivamente ya sea compuertas NOR, o bien compuertas NAND.
- Para ello debo verificar que puedo construir las tres expresiones lógicas básicas (NOT, OR, AND) a partir de ellas

Universalidad de NAND

- NOT a partir de NAND: Dos maneras distintas

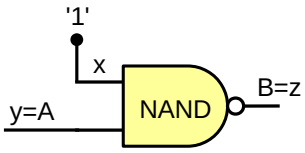


x	y	z
0	0	1
0	1	1
1	0	1
1	1	0



A	x	y	z	B
0	0	0	1	1
	0	1	1	
	1	0	1	
1	1	1	0	0

Como x e y no pueden ser distintos las filas están en rojo

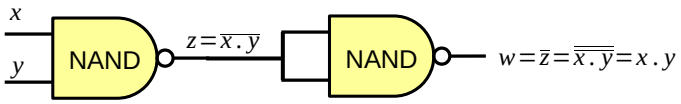


A	x	y	z	B
	0	0	1	
	0	1	1	
0	1	0	1	1
1	1	1	0	0

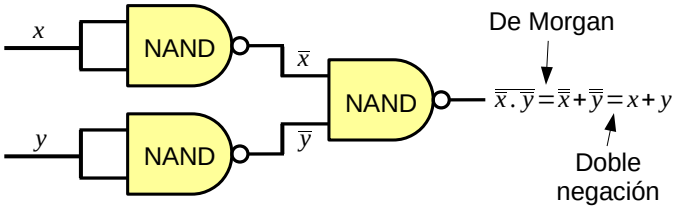
Como x no puede ser '0' las filas están en rojo

Universalidad de NAND

- AND a partir de NAND



- OR a partir de NAND

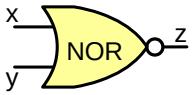


- Podemos implementar las operaciones básicas NOT, AND y OR con la compuerta NAND.

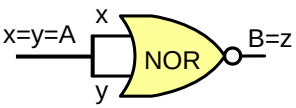
NAND es una compuerta universal

Universalidad de NOR

- NOT a partir de NOR: Dos maneras distintas

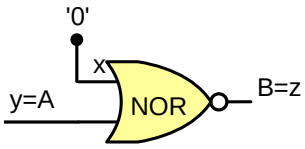


x	y	z
0	0	1
0	1	0
1	0	0
1	1	0



A	x	y	z	B
0	0	0	1	1
	0	1	1	
	1	0	1	
1	1	1	0	0

Como x e y no pueden ser distintos las filas están en rojo

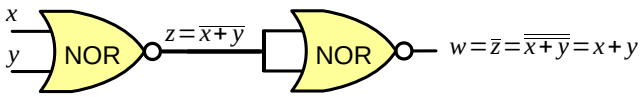


A	x	y	z	B
0	0	0	1	1
0	0	1	1	1
	1	0	1	
	1	1	0	

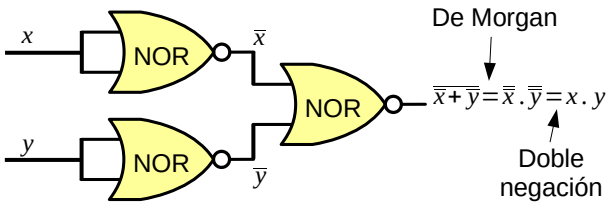
Como x no puede ser '1' las filas están en rojo

Universalidad de NAND

- OR a partir de NOR



- AND a partir de NOR

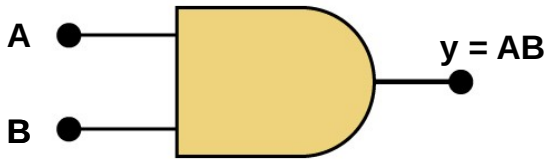


- Podemos implementar las operaciones básicas NOT, AND y OR con la compuerta NOR.

NOR es una compuerta universal

Computadora vs Compuerta

Ej. Comparar la operación de una computadora y un circuito lógico para realizar *y = AB.*

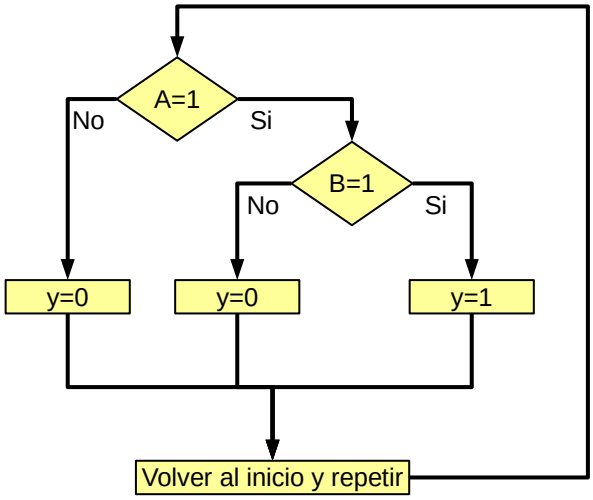


El circuito lógico es una compuerta AND. La salida y estará en HIGH unos 10 ns después que A y B estén en HIGH simultáneamente.

Luego de aproximadamente 10 ns después que cualquiera de las entradas vaya a LOW, la salida y irá a LOW.

Computadora vs Compuerta

Comparación de la operación de una computadora y un circuito lógico al realizar la operación lógica *y = AB.*



Cada forma en el diagrama de flujo representa una operación

Si cada una lleva 20 ns, va a llevar un mínim de dos o tres instrucciones (40-60 ns) responder a los cambios en las entradas.