**Arquitectura de Computadoras**

**Trabajo Práctico**

**Tema: Circuitos Combinatorios**

1. *Construya la tabla de verdad para las compuertas: AND, OR, NAND, NOR, XOR, NXOR, INVERSOR.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **AND** | | | **OR** | | |
| **A** | B | f | A | B | f |
| **0** | 0 | 0 | 0 | 0 | 0 |
| **0** | 1 | 0 | 0 | 1 | 1 |
| **1** | 0 | 0 | 1 | 0 | 1 |
| **1** | 1 | 1 | 1 | 1 | 1 |

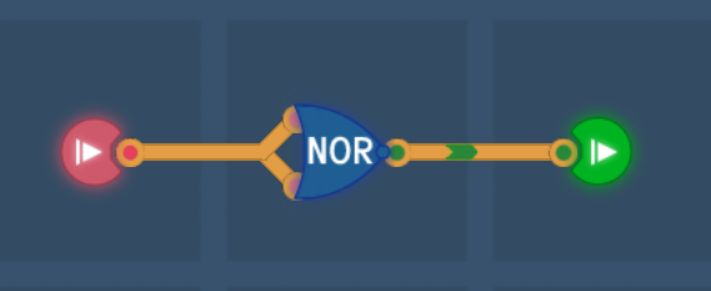
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NAND** | | | **NOR** | | |
| **A** | B | f | A | B | f |
| **0** | 0 | 1 | 0 | 0 | 1 |
| **0** | 1 | 1 | 0 | 1 | 0 |
| **1** | 0 | 1 | 1 | 0 | 0 |
| **1** | 1 | 0 | 1 | 1 | 0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **XOR** | | | **NXOR** | | |
| A | B | **f** | A | B | **f** |
| 0 | 0 | **0** | 0 | 0 | **1** |
| 0 | 1 | **1** | 0 | 1 | **0** |
| 1 | 0 | **1** | 1 | 0 | **0** |
| 1 | 1 | **0** | 1 | 1 | **1** |

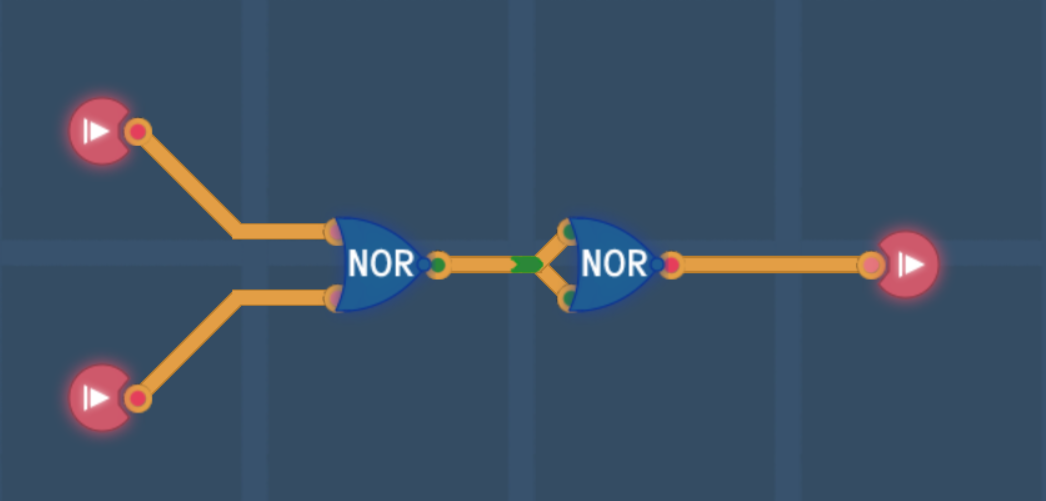
|  |  |
| --- | --- |
| **INVERSOR** | |
| A | **f** |
| 0 | **1** |
| 1 | **0** |

*2. Implemente las funciones lógicas NOT (una entrada), AND y OR (dos entradas), utilizando compuertas NOR.*

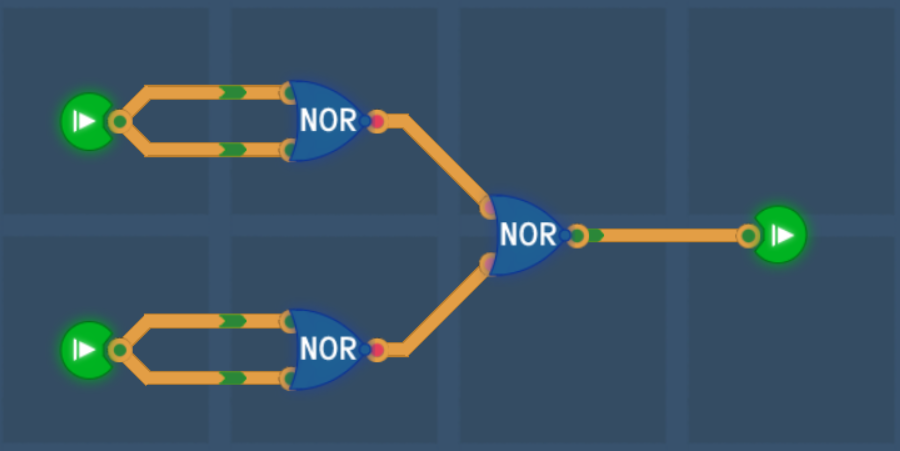
*NOT*

**

*OR*

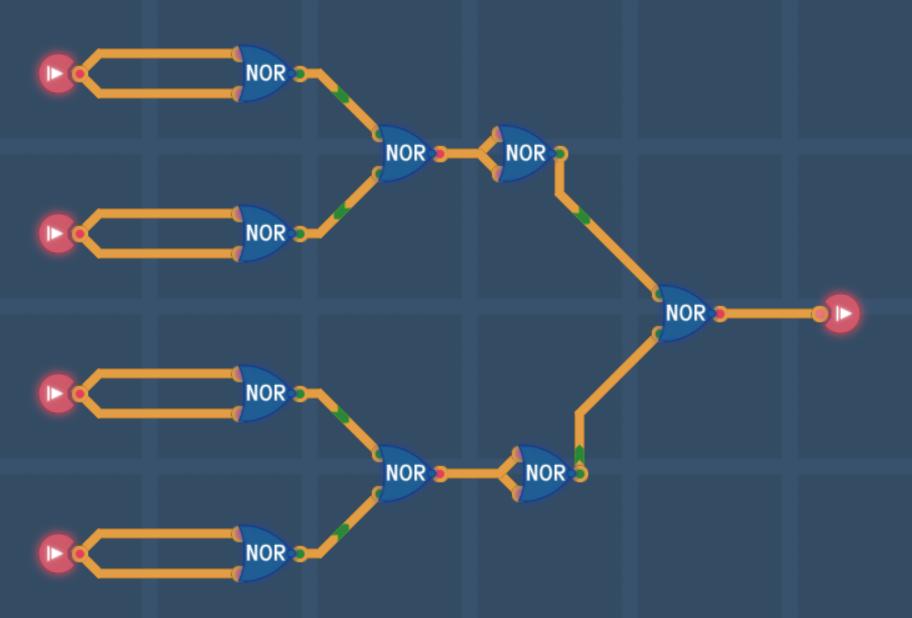
**

*AND*

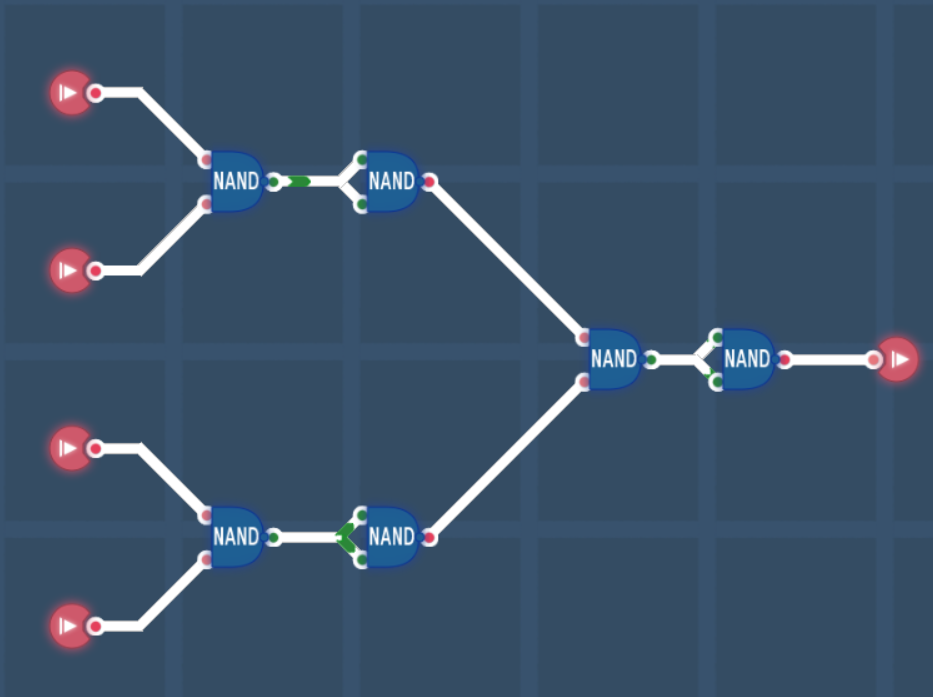
**

*3. Implemente la función lógica AND de cuatro entradas utilizando exclusivamente:*

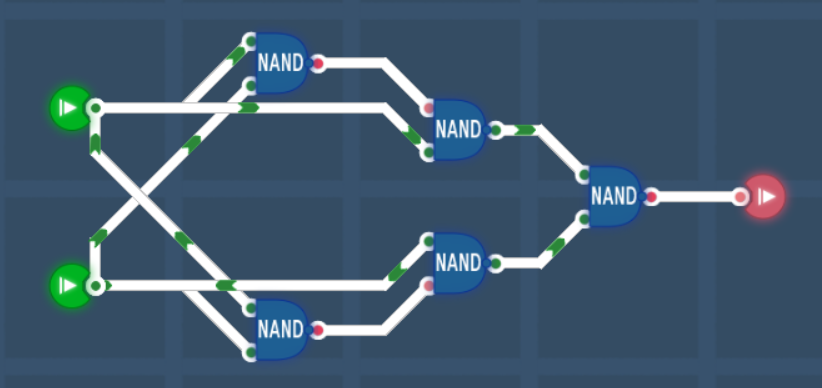
*a) Compuertas NOR de dos entradas.*

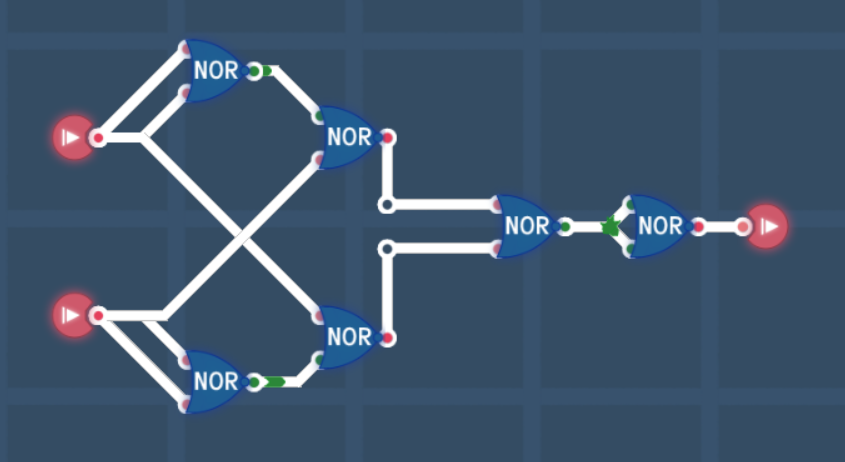
**

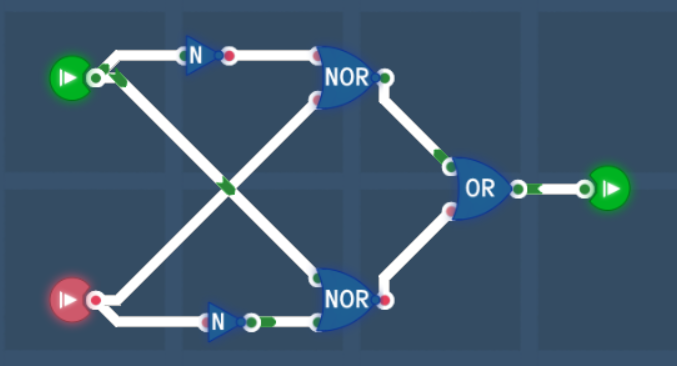
*b) Compuertas NAND de dos entradas.*

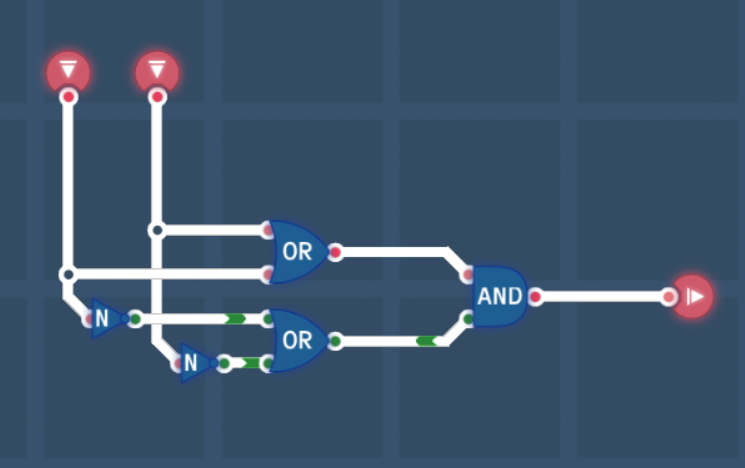
**

*4. Diseñe al menos dos circuitos para calcular la función lógica XOR (OR exclusivo)*

**

**

**

**

*5. Exprese la función de la tabla de verdad que se muestra en el Anexo I como suma de productos y como productos de suma. Simplifique por medio del álgebra de Boole.*

Tabla

Descripción generada automáticamente

Tabla 1: F = A’B’C’+A’B’C+A’BC+ABC’

F = A’B’(C’+C)+A’BC+ABC’

F = A’B’1+A’BC+ABC’

F = A’B’+A’BC+ABC’

F = A’(B’+BC)+ABC’

F = A’(B’+C)+ABC’

F = A’B’+A’C+ABC’

Comprobación:

|  |  |  |
| --- | --- | --- |
| AB\C | 0 | 1 |
| 00 | 1 | 1 |
| 01 | 0 | 1 |
| 11 | 1 | 0 |
| 10 | 0 | 0 |

F’= (A+B’+C)(A’+B+C)(A’+B+C’)(A’+B’+C’)

F’= (A+B’+C)((A’+B)+(CC’))(A’+B’+C’)

F’= (A+B’+C)((A’+B)+0)(A’+B’+C’)

F’= (A+B’+C)(A’+B)(A’+B’+C’)

F’= (A+B’+C)(A’+B(B’+C’))

F’= (A+B’+C)(A’+BC’)

F’= (A+B’+C)(A’+B)(A’+C’)

Comprobación:

|  |  |  |
| --- | --- | --- |
| AB\C | 0 | 1 |
| 00 | 1 | 1 |
| 01 | 0 | 1 |
| 11 | 1 | 0 |
| 10 | 0 | 0 |

Tabla 2:

S1 = A’BC’D’+AB’C’D’

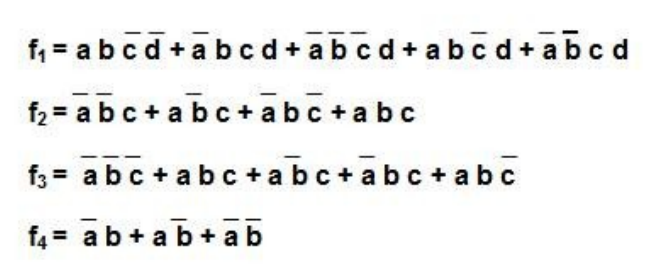
S2 = A’B’CD’+AB’C’D’

S3 = A’B’C’D+A’B’CD’+AB’C’D’

Tabla

Descripción generada automáticamente

1. *Simplifique las funciones f1 a f4 del Anexo II, utilizando mapas de Karnaugh, álgebra de Boole y el método de Quine-McKluskey. Compare los resultados y elabore una conclusión.*



F1 = ABC’D’ + A’BCD + A’B’C’D + ABC’D + A’B’CD

Mapa de Karnaugh.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AB\CD | 00 | 01 | 11 | 10 |
| 00 | 0 | 1 | 1 | 0 |
| 01 | 0 | 0 | 1 | 0 |
| 11 | 1 | 1 | 0 | 0 |
| 10 | 0 | 0 | 0 | 0 |

F1 = ABC’ + A’B’D + A’CD

Algebra de Boole.

F1 = ABC’D’ + A’BCD + A’B’C’D + ABC’D + A’B’CD

= ABC’D’ + ABC’D + A’BCD + A’B’C’D + A’B’CD

= ABC’(D’ + D) + A’BCD + A’B’D(C’ + C)

= ABC’ + A’BCD + A’B’D

= ABC’ + A’D(BC + B’)

= ABC’ + A’D (C+B’)

= ABC’ + A’B’D + A’CD

Metodo de Quine-Mcklauskey.

F1 = ABC’D’ + A’BCD + A’B’C’D + ABC’D + A’B’CD

F1 = M1 + M3 + M0 + M4 + M2

|  |  |  |
| --- | --- | --- |
| M0 | 0001 | 1 |
| M1 | 1100 | 2 |
| M2 | 0011 | 2 |
| M3 | 0111 | 3 |
| M4 | 1101 | 3 |

|  |  |  |
| --- | --- | --- |
| M1-m4 | 110- | 2 |
| M0-m2 | 00-1 | 1 |
| M2-m3 | 0-11 | 2 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | M0 | M1 | M2 | M3 | M4 |
| 110- |  | x |  |  | x |
| 00-1 | x |  | x |  |  |
| 0-11 |  |  | x | x |  |

Esenciales: ABC’+A’B’D+A’CD

F2 = A’B’C + AB’C + A’BC’ +ABC

Mapa de Karnaugh.

|  |  |  |
| --- | --- | --- |
| AB\C | 0 | 1 |
| 00 | 0 | 1 |
| 01 | 1 | 0 |
| 11 | 0 | 1 |
| 10 | 0 | 1 |

F2 = A’BC’ + AC + B’C

Albebra de Boole.

F2 = A’B’C + AB’C + A’BC’ + ABC

= B’C(A’ + A) + A’BC’ + ABC

= B’C + A’BC’ + ABC

= A’BC’ + (B’ + AB)C

= A’BC’ + B’C + AC

Metodo de Quine-Mcklauskey.

F2 = A’B’C + AB’C + A’BC’ + ABC

= M0 + M2 + M1 + M3

|  |  |  |
| --- | --- | --- |
| M0 | 001 | 1 |
| M1 | 010 | 1 |
| M2 | 101 | 2 |
| M3 | 111 | 3 |

|  |  |  |
| --- | --- | --- |
| M0-M2 | -01 | 1 |
| M1 | 010 | 1 |
| M2-M3 | 1-1 | 2 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | M0 | M1 | M2 | M3 |
| -01 | X |  | X |  |
| 010 |  | X |  |  |
| 1-1 |  |  | X | X |

F2 = B’C + A’BC’ + AC

*F3 = A’B’C’+ABC+AB’C+A’BC+ABC’*

Mapa de Karnaugh.

|  |  |  |
| --- | --- | --- |
| AB\C | 0 | 1 |
| 00 | 1 | 0 |
| 01 | 0 | 1 |
| 11 | 1 | 1 |
| 10 | 0 | 1 |

*F3 = A’B’C’ + AB + BC + AC*

*Algebra de Boole.*

*F3 = A’B’C’ + ABC + AB’C + A’BC + ABC’*

*= A’B’C’ + AC(B + B’) + A’BC + ABC’*

*= A’B’C’ + AC + A’BC + ABC’*

*= A’B’C’ + (A + A’B)C + ABC’*

*= A’B’C’ + ABC’ + AC + BC*

*= A’B’C’ + A(BC’ + C) + BC*

*= A’B’C’ + AB + AC + BC*

Metodo de Quine-Mcklauskey.

*F3 = A’B’C’ + ABC + AB’C + A’BC + ABC’*

*= M0 + M4 + M1 + M2 + M3*

|  |  |  |
| --- | --- | --- |
| *M0* | *000* | *0* |
| *M1* | *101* | *2* |
| *M2* | *011* | *2* |
| *M3* | *110* | *2* |
| *M4* | *111* | *3* |

|  |  |  |
| --- | --- | --- |
| *M0* | *000* | *0* |
| *M1-M4* | *1-1* | *2* |
| *M2-M4* | *-11* | *2* |
| *M3-M4* | *11-* | *2* |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | M0 | M1 | M2 | M3 | M4 |
| *000* | X |  |  |  |  |
| *1-1* |  | X |  |  | X |
| *-11* |  |  | X |  | X |
| *11-* |  |  |  | X | X |

*F3 = A’B’C’ + AC + BC + AB*

Mapa de Karnaugh.

*F4 = A’B + AB’ + A’B’*

|  |  |
| --- | --- |
| AB | f |
| 00 | 1 |
| 01 | 1 |
| 11 | 0 |
| 10 | 1 |

*F4 = A’ + B’*

*Algebra de Boole.*

*F4 = A’B + AB’ + A’B’*

*= A’B + (A + A’)B’*

*= A’B + B’*

*F4 = A’ + B’*

*Método de Quine-Mcklauskey.*

*F4 = A’B + AB’ + A’B’*

*= M1 + M2 + M0*

|  |  |  |
| --- | --- | --- |
| *M0* | *00* | *0* |
| *M1* | *01* | *1* |
| *M2* | *10* | *1* |

|  |  |  |
| --- | --- | --- |
| *M0-M1* | *0-* | *0* |
| *M0-M2* | *-0* | *0* |

|  |  |  |  |
| --- | --- | --- | --- |
|  | M0 | M1 | M2 |
| *0-* | X | X |  |
| -0 | X |  | X |

*F4 = A’ + B’*

1. *Dibuje el circuito lógico de las funciones simplificadas en el ejercicio anterior.*

*8. Diseñe un circuito que acepte un carácter de 4 bits como entrada y genere una salida de 7 bits que corresponden al código de Hamming de paridad impar del carácter ingresado.*

*9. Resolver utilizando exclusivamente compuertas NOR la función f5 del Anexo II.*

*10. Diseñe los siguientes circuitos utilizando un multiplexor.*

*a) La función paridad de 5 (sale 1 si y sólo si hay un número par de unos en las entradas)*

*b) La función Mayoría de 5 (sale 1 si y sólo si hay más unos que ceros en las entradas)*

*c) La función minoría de 5 (sale 1 si y sólo si hay más ceros que unos en las entradas)*

*11. Dos números de dos bits A=(a1,a0) y B=(b1,b0) deben compararse por medio de una función de 4 variables f(a0.a1.b0.b1). La función f debe tener el valor 1 si v(A) <= v(B), donde v(x) = 2\*x1 + x0. Diseñe el circuito correspondiente para implementar la misma*

*12. Diseñe un circuito que acepte 4 bits en la entrada y que representen un dígito decimal en formato BCD y determine los valores de las siete salidas que corresponden a su representación en un display de siete segmentos. En el Anexo III, la figura 1 indica cómo se identifican cada uno de los siete segmentos en el display y la figura 2 muestra cómo debe construirse la tabla de verdad dando como ejemplo el resultado para los valores de entrada 0000 y 1001 respectivamente. 13. Diseñe un circuito con cuatro entradas y tres salidas que funcione de acuerdo a la tabla 2 del Anexo I, la salida s3 indica que solo hay una entrada con valor 1 y s1 con s2 indican cual es la entrada que tiene dicho valor. 14. Implemente un circuito con dos entradas y cuatro salidas, donde el valor presente en las entradas definirán qué salida estará en 1 (no debe haber mas de una salida con 1 simultáneamente).*