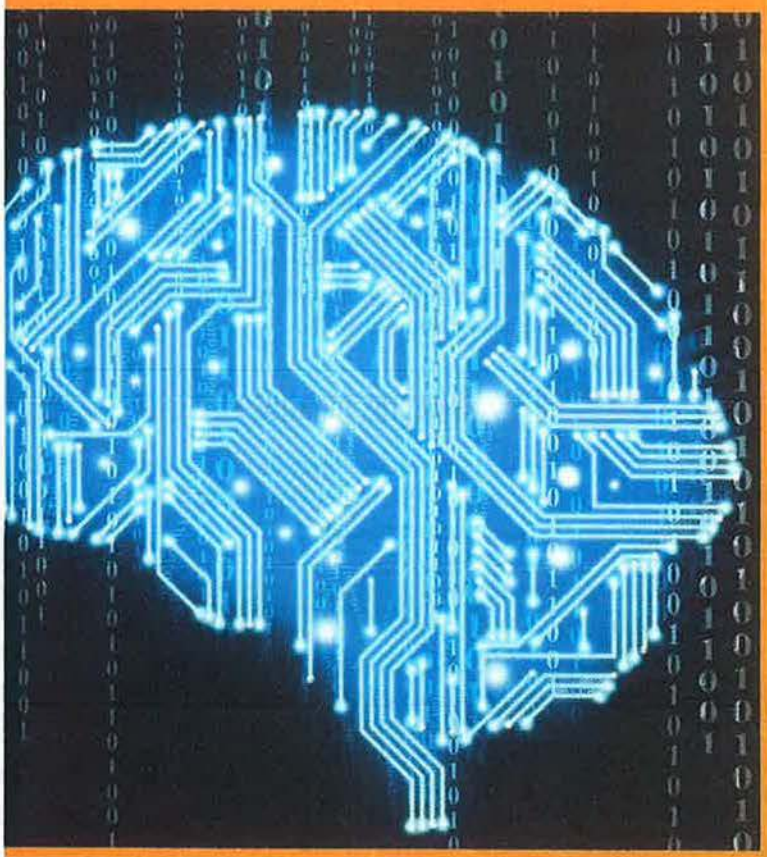
Fundamentos de electrónica digital

9



Los circuitos digitales se encuentran en innumerables productos de equipos domésticos, industriales y de telecomunicación, entre otros, ya que el desarrollo de circuitos integrados a bajo coste ha motivado la evolución de la tecnología y su gran expansión.

La electrónica digital es la parte de la electrónica que ocupa los sistemas que trabajan en dos únicos estados. A ambos estados se les asigna los valores 1 o 0, o también se denominan valor "verdadero" o valor "falso". Así pues, en un circuito electrónico digital hay dos niveles de tensión. Típicamente para el valor "verdadero" o 1 se utilizan los rangos de voltaje: 1,5, 3, 5, 9 y 18 V en función de la aplicación a la que se destine. El valor "falso" o 0 corresponde al valor de potencial de referencia o masa.

Contenidos

Objetivos

- Diferenciar los distintos sistemas de numeración.
- Definir la lógica de contactos y de funciones digitales.
- Dar a conocer el álgebra de Boole, teoremas y axiomas.
- Analizar las formas de obtener circuitos lógicos y tablas de verdad.
- Describir los procedimientos de simplificación de funciones.

9.1. Sistemas de numeración

Un sistema de numeración se puede definir como el conjunto de símbolos (dígitos, caracteres o ambos) y reglas que se utilizan para la representación de datos numéricos o cantidades.

Los sistemas de numeración tienen diferentes reglas, según:

- El sistema de numeración considerado (por ejemplo, decimal, binario, octal, hexadecimal, etc.).
- El conjunto de símbolos permitidos en el sistema. En el caso del sistema decimal son {0,1,...9}; en el binario son {0,1}; en el octal son {0,1,...7}; en el hexadecimal son {0,1,...9, A, B, C, D, E, F}.
- Los números que son válidos en el sistema y cuáles no. En un sistema de numeración posicional las reglas son bastante simples, mientras que la numeración no posicional, por ejemplo la romana, requiere reglas algo más elaboradas.

Estas reglas son diferentes para cada sistema de numeración considerado, pero una regla común a todos es que para construir números válidos solo se pueden utilizar los símbolos permitidos en ese sistema. Para indicar en qué sistema de numeración se representa una cantidad se añade como subíndice a la derecha el número de símbolos que se pueden representar en dicho sistema (base).

SABÍAS QUE

La civilización maya utilizaba un sistema de numeración de raíz mixta de base 20 (vigesimal) y ya desarrollaron el concepto de cero alrededor del año 36 a.C. Las inscripciones muestran sumas de hasta cientos de millones y fechas tan extensas que se empleaban varias líneas para poder representarlas.

9.1.1. Números decimales

El sistema de numeración decimal es el más usado en todo el mundo. Se utilizan los dígitos 0, 1, 2, 3, 4, 5, 6, 7, 8 y 9. Dado que tiene diez dígitos, se denomina **sistema de base 10**, de ahí su nombre "decimal". Es un **sistema que tiene la característica de valor por posición**. Para descomponerlo, el valor de las unidades se multiplica por la base 10 elevándola a 0, el valor de las decenas se multiplica por la base 10 elevándola a 1, el valor de las centenas se multiplica por la base 10 elevándola a 2, el valor de los millares se multiplica por la base 10 elevándola a 3 y así sucesivamente, sumando los anteriores resultados se obtiene el número compuesto. Por ejemplo: si tomamos el número 924, se puede descomponer en las sumas: $9 \times 10^2 + 2 \times 10^1 + 4 \times 10^0$, dándonos como resultado el número buscado.

9	2	4
↓	↓	↓
9×10^2	2×10^1	4×10^0
$= 900$	$= 20$	$= 4$

Veamos otro ejemplo, esta vez para el número 8.351.

8	3	5	1
↓	↓	↓	↓
8×10^3	3×10^2	5×10^1	1×10^0
$= 8.000$	$= 300$	$= 50$	$= 1$

Como será estudiado más adelante, el sistema binario (base 2), el sistema octal (base 8) y el hexadecimal (base 16), tienen la misma característica de valor por posición. Estos sistemas se utilizan en electrónica digital, computadoras y microcomputadoras.

9.1.2. Números binarios

El sistema de numeración binario utiliza únicamente dos dígitos: el 0 y el 1. Por tanto, tiene base 2. A cada uno de los dígitos se le denomina **bit**.

SABÍAS QUE

Bivalente es sinónimo de tener dos posibilidades de combinación. En lógica binaria significa tener dos estados, 0 o 1.

Así, por ejemplo, tenemos el número binario: 110, que se lee "uno, uno, cero".

Conversión de numeración binaria a decimal

Haciendo lo mismo que en el sistema decimal, pero en vez de con base 10 con base 2, tenemos lo siguiente:

$1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 6$. Por tanto, el número binario 110 equivale al número 6 en sistema decimal.

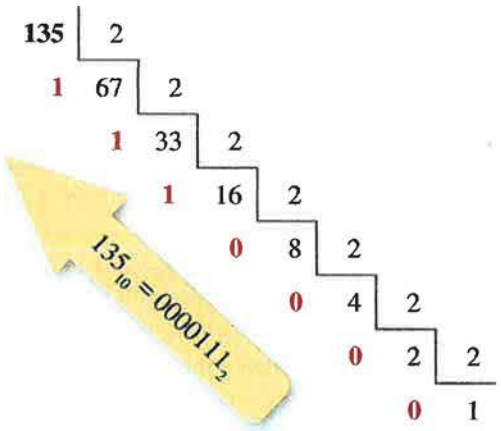
El bit más significativo es el que está más a la izquierda y el menos significativo es el que está situado más a la derecha.

RECUERDA

Los sistemas de control que utilizan contactores y relés funcionan con componentes que tienen dos estados muy bien diferenciados: abierto (conduce) o cerrado (no conduce). A estos componentes se los denomina lógicos o de **"todo o nada"**.

Conversión de numeración decimal a binario

Para realizar la conversión de decimal a binario se procede de la siguiente manera: se divide el número del sistema decimal entre 2, este último resultado entero se vuelve a dividir entre 2, y así sucesivamente hasta que el dividendo sea menor que el divisor (2). Cuando el número a dividir sea 1 se termina. A continuación se ordenan los restos de las divisiones empezando desde el último hacia el primero, y se colocan en dicho orden inverso. Este será el número binario que buscamos.



Otro método de conversión decimal a binario consiste hacerlo de forma similar a la factorización de números primos. Se pone el número y una línea vertical para separar dos zonas. En la parte de la izquierda se pondrán los resultados de dividir entre 2 el número inmediatamente superior. **En la parte de la derecha se pone 0 si el resultado obtenido ha sido par o 1 si el resultado ha sido impar. En caso de salir impar, nos quedamos siempre con la parte entera de dicha división. La parte decimal no la tenemos en cuenta,** nunca ponemos decimales. Para obtener el número binario, ordenamos la columna de la derecha empezando de abajo hacia arriba.

125	**1**
62	0
31	1
15	1
7	1
3	1
1	1

$125_{10} = 1111101_2$

RECUERDA

Un sistema de base 2 (binario) tiene dos posibles valores (0 y 1), de base 8 (octal) ocho posibles valores (0 al 7), de base 10 (decimal) diez posibles valores (0 al 9), de base 16 (hexadecimal) dieciséis valores (0 al 9 y de la A a la F).

Para convertir un número de decimal a binario hay **otro método denominado "de distribución".** Se trata de distribuir los unos entre las potencias de dos, de modo que la suma de las potencias dé el número decimal para ser convertido.

Ejemplo

Queremos convertir el número 241 a binario. Se colocan a la izquierda todas las potencias de 2 hasta encontrar el límite inmediatamente inferior a 241 e inmediatamente superior. En este caso, el límite inferior es $2^7 = 128$ ya que $2^8 = 256$ es superior al número a convertir y por tanto es el límite superior. El límite superior no se utiliza. Ahora se trata de buscar la suma para obtener el número 241. En 128 se pone un 1 a la derecha y como tenemos que $241 - 128 = 113$, 113 es el número que tenemos que obtener a base de sumar desde 2^0 a 2^6, por tanto ponemos 1 en: 64, 32, 16 y 1. Como en casos anteriores, ordenamos los números binarios de abajo hacia arriba obteniendo el número binario 11110001.

$2^0 = 1$	**1**
$2^1 = 2$	**0**
$2^2 = 4$	**0**
$2^3 = 8$	**0**
$2^4 = 16$	**1**
$2^5 = 32$	**1**
$2^6 = 64$	**1**
$2^7 = 128$	**1**
$2^8 = 256$	-

$241_{10} = 11110001_2$

Actividad propuesta 9.1

Convierte los siguientes números decimales al sistema binario, utilizando el método que prefieras de los estudiados.

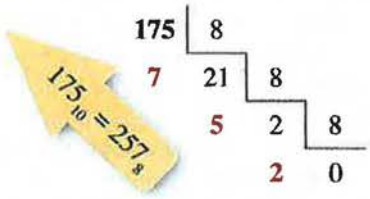
a) 345.
b) 1.124.
c) 76.
d) 92.
e) 154.
f) 3.257.

9.1.3. Números octales

Es el sistema numérico de base 8 y utiliza los dígitos 0, 1, 2, 3, 4, 5, 6 y 7. Estos tienen el mismo valor que en el sistema de numeración decimal. Tiene la característica que la base 8 es potencia exacta de 2 ($8 = 2^3$) que es la base de la numeración binaria, por lo que la conversión octal a binario o viceversa resulta muy simple.

Conversión de decimal a octal

Para convertir un número en base decimal a base octal se divide por 8 sucesivamente hasta llegar a cociente 0, y los restos de las divisiones en orden inverso indican el número en octal. Para pasar de base 8 a base decimal, solo hay que multiplicar cada cifra por 8 elevado a la posición de la cifra, y sumar el resultado.



Conversión de binario a octal

La ventaja principal del sistema de numeración octal es la **facilidad** con que puede realizarse la **conversión entre un número binario y octal.**

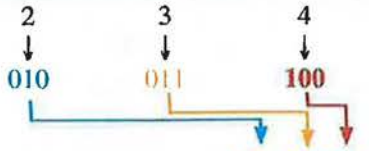
Para convertir de binario a octal solo hay que agrupar de 3 en 3 los dígitos binarios, así, el número 107 (en decimal) es 1101011 (en binario). Para convertirlo a octal, el número binario lo agrupamos de tres en tres empezando por la derecha: 1|101|011, después obtenemos el número en decimal de cada uno de los números en binario obtenidos: 1 = 1, 101 = 5 y 011 = 3. De modo que el número decimal 107, en binario es 1101011 y en octal es 153.

Conversión de octal a binario

Para convertir de octal a binario se procede de forma similar a la anterior. **Se convierte cada dígito octal en un número binario de tres dígitos** y seguidamente se juntan los grupos de dígitos obtenidos.

Ejemplo

Convertimos el número 234 (octal) a binario.



El número binario resultante es: 010011100

9.1.4. Números hexadecimales

El sistema de numeración hexadecimal tiene base 16. En este sistema, se utilizan los símbolos alfanuméricos: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F. La letra A equivale al 10 decimal, la B al 11, la C al 12, la D al 13, la E al 14 y la F al 15.

Este sistema **es muy útil y práctico para transformar directamente números binarios de 4 bits**, es decir, del 0000 al 1111 pudiéndose representar únicamente por un símbolo hexadecimal.

En informática y en electrónica para circuitos digitales industriales, a veces se utiliza la numeración octal en lugar de la hexadecimal. Tiene la ventaja que no requiere utilizar otros símbolos diferentes de los dígitos.

Sin embargo, para trabajar con bytes o conjuntos de ellos, y teniendo en cuenta que un byte es un conjunto de 8 bits, suele ser más cómodo el sistema hexadecimal, ya que todo byte así definido es completamente representable por dos dígitos hexadecimales.

Al igual que los sistemas anteriores (decimal, binario y octal), utiliza el mismo concepto de valor por posición. Así pues, tenemos el número 23 decimal, que equivale al 10111 en binario y al 17 en hexadecimal ($1 \times 16^1 + 7 \times 16^0 = 23$).

RECUERDA

La A en hexadecimal corresponde al 10 en decimal, B al 11, C al 12, D al 13, E al 14 y F al 15.

Conversión de hexadecimal a decimal

Para convertir de hexadecimal a decimal se aplica el mismo método que los anteriores de multiplicar el valor de la posición de cada uno de los caracteres del número hexadecimal, multiplicándola por la base (16) y elevando dicha base a la potencia que indica la posición, siendo 0 para el valor de la derecha (menos significativo), 1 para el segundo empezando por la derecha, 2 para el tercero empezando por la derecha, etc., hasta llegar al carácter más significativo. Los caracteres con letras se sustituyen por su equivalente en decimal (A = 10, B = 11, C = 12, D = 13, E = 14, F = 15).

SABÍAS QUE

Las calculadoras científicas hacen todo tipo de conversiones con números decimales, binarios, octales, hexadecimales... Una herramienta informática muy útil para este tipo de conversones es la hoja de cálculo Microsoft Excel.

Ejemplos

$$1D8_{16} = 1 \times 16^2 + \mathbf{13} \times 16^1 + 8 \times 16^0 = 472_{10}$$

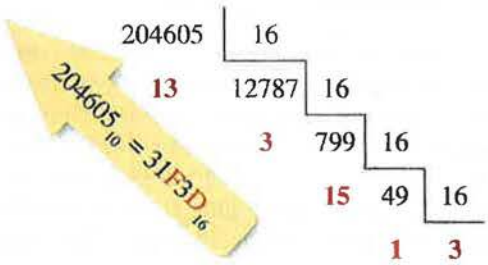
$$A5C98F_{16} = \mathbf{10} \times 16^5 + 5 \times 16^4 + \mathbf{12} \times 16^3 +$$

$$+ 9 \times 16^2 + 8 \times 16^1 + \mathbf{15} \times 16^0 = 10865039_{10}$$

Conversión de decimal a hexadecimal

Para convertir de decimal a hexadecimal se hace de la misma manera que convertir un número de decimal a binario, solo que en vez de dividir por 2 dividimos por 16 (método de divisiones sucesivas) y cogemos el último cociente y los restos ordenados de forma inversa, teniendo en cuenta que si en el último cociente o restos da como resultado del 10 al 15, se convierte automáticamente la A por 10, la B por 11, la C por 12, la D por 13, la E por 14 y la F por 15.

Ejemplo



Conversión de hexadecimal a binario

Para convertir un número hexadecimal en código binario, se obtiene el número binario de cada una de sus cifras con cuatro bits, es decir, un byte. Posteriormente a la conversión, **si hay alguno o varios ceros a la izquierda**, se eliminan hasta el primer uno.

$5D = 01011101 = 1011101$

$1A = 00011010 = 11010$

Actividad propuesta 9.2

Convierte a binario y a decimal los siguientes números hexadecimales:

a) $2B_{16}$ b) $1F3_{16}$

c) $23E_{16}$ d) $49B_{16}$

9.1.5. Sistemas para representar números negativos binarios

Para representar números binarios negativos, el bit más significativo o MSB es la cifra del número que representa el bit de signo.

Si el bit más significativo es 0, el número binario es positivo (+). Si por el contrario, si es 1, el número es negativo (–).

Este tipo de números se emplea en microprocesadores que son utilizados en PC, autómatas programables para automatismos y todos los circuitos electrónicos que utilizan sistema digital para procesamiento de datos.

Para realizar la conversión de números negativos, se distinguen dos complementos:

Complemento a-1

El complemento a-1 se obtiene al cambiar cada una de sus cifras 0 por 1 y viceversa, esto es, permutar cada una de sus cifras por su complementario.

Ejemplo

Obtención del complemento a-1 del número binario 11011010.

Número binario	1	1	0	1	1	0	1	0
Complemento a-1	0	0	1	0	0	1	0	1

Complemento a-2

Para obtener el complemento a-2 de un número binario, se hace el complemento a-1 de él y **se suma 1 al bit menos significativo** (cifra que corresponde a la derecha del número).

Para convertirlo a negativo, se añade un bit a la izquierda del bit más significativo del complemento a-2. Si este bit es 0, el número es positivo y si es 1 es negativo.

Ejemplo

Obtención del complemento a-2 del número decimal 13.

Número decimal	1	3			
Número binario	1	1	0	1	Se convierte 13_{10} a binario
Complemento a-1	0	0	1	0	Se cambia 1 por 0 y viceversa
Complemento a-2	0	0	1	1	Se suma 1 al bit menos significativo
Número negativo (-13_{10})	1	0	0	1	1	Se añade la cifra 1 a la izquierda del número para indicar que es negativo

Actividad propuesta 9.3

Obtén el complemento a-2, de los siguientes números:

a) -12_{10}

b) -45_{10}

c) 68_{10}

d) 36_{10}

9.1.6. Código Gray

El código Gray o binario reflejado es un sistema de numeración en el que los números sucesivos se diferencian solo en una cifra.

Para formarlo, se suma el número a sí mismo desplazado una posición a la derecha, y descartando el bit menos significativo.

Se emplea para comprobar señales con ruidos en telecomunicaciones, verificación de posiciones correctas en *swiches* de tarjetas electrónicas y similares.

Ejemplo

Obtención del número binario 10010101 en código Gray.

Número decimal	1	0	0	1	0	1	0	1
Desplazamiento		1	0	0	1	0	1	0 +
Número en código Gray	1	1	0	1	1	1	1	1

Actividad propuesta 9.4

Obtén el código Gray de los números decimales del 0 al 16.

9.2. Lógica de contactos

Consiste en representar los esquemas de automatismos industriales compuestos de relés, contactores, pulsadores e interruptores.

Entradas

Las entradas son variables y se clasifican en:

- **Directa:** es aquella que entrega valor alto, valor verdadero o "1" al activarse.
- **Inversa:** al contrario que la entrada inversa, esta entrega un valor bajo, valor falso o "0" al activarse. Es una variable negada.

Desde el punto de vista de cómo son introducidas las variables de entrada al sistema, pueden ser:

- **Entradas puras:** son aquellas que son manejadas por el proceso o por el operador del automatismo.
- **Entradas con salida realimentada:** son las entradas que vienen previamente de una variable de salida que se introduce como entrada. Es típico en automatismos industriales secuenciales.

Salidas

Van asociadas a las variables de salida de una función lógica. Esta función es implementada por los relés y contactores del circuito de control que actúa sobre el circuito de potencia.

Todos los elementos anteriormente indicados **se interconectan creando estructuras o redes de contactos elementales que, al combinar estas, se obtienen los grandes circuitos de mando** o de maniobra de los cuadros para automatismos.

9.2.1. Función lógica AND o "Y"

La función Y en lógica de contactos se trata de elementos de entrada directa como pulsadores e interruptores, que están conectados en serie, por tanto, cualquier pulsador o **contacto que no esté cerrado la salida será un 0** lógico o nivel bajo, mientras que si todos los **contactos están cerrados, será un 1** lógico o nivel alto. Equivale por tanto a la operación del producto. La función lógica del esquema siguiente viene dada por *y*.

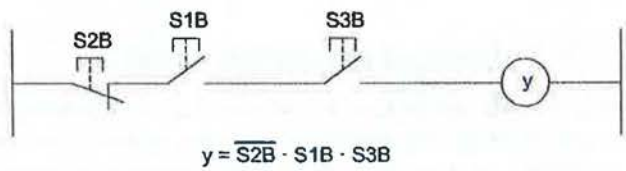


Figura 9.1. *Esquema de contactos de la función lógica AND o "Y".*

Se verifica el estado encendido o apagado por la siguiente tabla de verdad:

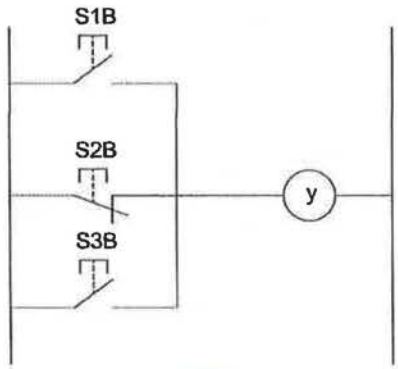
Tabla 9.1. Tabla de verdad de la función lógica AND o "Y".

S1	S2	S3	Y
OFF	OFF	OFF	OFF
OFF	OFF	ON	OFF
OFF	ON	OFF	OFF
OFF	ON	ON	ON

S1	S2	S3	Y
ON	OFF	OFF	OFF
ON	OFF	ON	OFF
ON	ON	OFF	OFF
ON	ON	ON	OFF

9.2.2. Función lógica OR u "O"

La función O en lógica de contactos son elementos de entrada directa como los pulsadores e interruptores, que están conectados en paralelo, por tanto, si todos los contactos están abiertos, la salida será un 0 lógico o nivel bajo, mientras que si cualquiera de los contactos se cierra, la salida será un 1 lógico o nivel alto. Por tanto, equivale a la suma algebraica. La **función lógica** del esquema siguiente viene dada por *y*.



$y = S1B + \overline{S2B} + S3B$

Figura 9.2. *Esquema de contactos de la función OR u "O".*

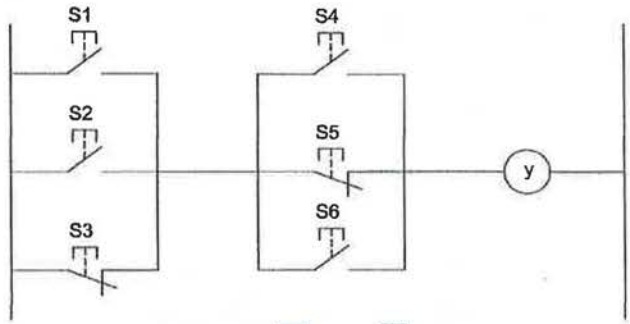
Se verifica el estado encendido o apagado por la siguiente tabla de verdad:

Tabla 9.2. Tabla de verdad de la función OR u "O".

S1	S2	S3	Y
OFF	OFF	OFF	ON
OFF	OFF	ON	ON
OFF	ON	OFF	OFF
OFF	ON	ON	ON
ON	OFF	OFF	ON
ON	OFF	ON	ON
ON	ON	OFF	ON
ON	ON	ON	ON

9.2.3. Función AND de funciones OR

La función AND de funciones OR consiste en la unión en serie de conjuntos en paralelo. Para que la salida de la función *y* tenga valor lógico 1 o nivel alto, se debe cumplir que los contactos de una o varias de las ramas del primer bloque tengan los contactos cerrados, del segundo bloque una o varias ramas tengan los contactos cerrados, del tercero si existiera ídem y así sucesivamente. En caso contrario, el circuito quedaría abierto y la función lógica *y* toma el valor 0 lógico o nivel bajo.



$y = (S1 + S2 + \overline{S3}) \cdot (S4 + \overline{S5} + S6)$

Figura 9.3. *Función AND de funciones OR.*

Se verifica el estado encendido o apagado por la siguiente tabla de verdad:

Tabla 9.3. Tabla de verdad de la función anterior AND de funciones OR.

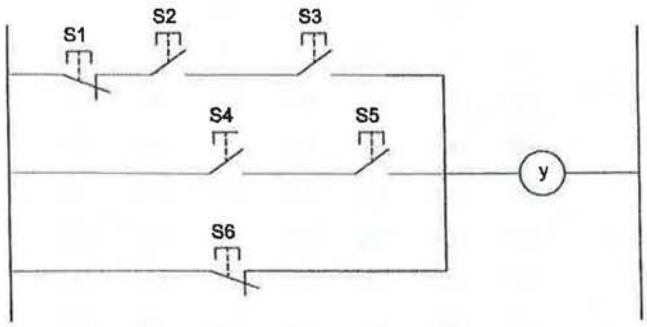
S1	S2	S3	S4	S5	S6	Y
OFF	OFF	OFF	OFF	OFF	OFF	ON
OFF	OFF	OFF	OFF	OFF	ON	ON
OFF	OFF	OFF	OFF	ON	OFF	OFF
OFF	OFF	OFF	OFF	ON	ON	ON
OFF	OFF	OFF	ON	OFF	OFF	ON
OFF	OFF	OFF	ON	OFF	ON	ON
OFF	OFF	OFF	ON	ON	OFF	ON
OFF	OFF	OFF	ON	ON	ON	ON
OFF	OFF	ON	OFF	OFF	OFF	OFF
OFF	OFF	ON	OFF	OFF	ON	OFF
OFF	OFF	ON	OFF	ON	OFF	OFF
OFF	OFF	ON	OFF	ON	ON	OFF
OFF	OFF	ON	ON	OFF	OFF	OFF
OFF	OFF	ON	ON	OFF	ON	OFF
OFF	OFF	ON	ON	ON	OFF	OFF
OFF	OFF	ON	ON	ON	ON	OFF
OFF	ON	OFF	OFF	OFF	OFF	ON
OFF	ON	OFF	OFF	OFF	ON	ON
OFF	ON	OFF	OFF	ON	OFF	OFF

Tabla 9.3. Tabla de verdad de la función anterior AND de funciones OR *(continuación).*

S1	S2	S3	S4	S5	S6	Y
OFF	ON	OFF	OFF	ON	ON	ON
OFF	ON	OFF	ON	OFF	OFF	ON
OFF	ON	OFF	ON	OFF	ON	ON
OFF	ON	OFF	ON	ON	OFF	ON
OFF	ON	OFF	ON	ON	ON	ON
OFF	ON	ON	OFF	OFF	OFF	ON
OFF	ON	ON	OFF	OFF	ON	ON
OFF	ON	ON	OFF	ON	OFF	OFF
OFF	ON	ON	OFF	ON	ON	ON
OFF	ON	ON	ON	OFF	OFF	ON
OFF	ON	ON	ON	OFF	ON	ON
OFF	ON	ON	ON	ON	OFF	ON
OFF	ON	ON	ON	ON	ON	ON

9.2.4. Función OR de funciones AND

Son combinaciones de la conexión en serie y paralelo. Para que la función *y* tenga un valor lógico 1 o nivel alto, cualquiera de sus ramas ha de tener valor 1 lógico y por tanto cualquiera de las ramas serie, dos o las tres, ha de tener todos sus contactos cerrados. En caso contrario, la función *y* toma valor 0 lógico o nivel bajo.



$$y = (\overline{S1} \cdot S2 \cdot S3) + (S4 \cdot S5) + \overline{S6}$$

Figura 9.4. *Función OR de funciones AND.*

Se verifica el estado encendido o apagado por la siguiente tabla de verdad:

Tabla 9.4. Tabla de verdad de la función anterior OR de funciones AND.

S1	S2	S3	S4	S5	S6	Y
OFF	OFF	OFF	OFF	OFF	OFF	ON
OFF	OFF	OFF	OFF	OFF	ON	OFF
OFF	OFF	OFF	OFF	ON	OFF	ON
OFF	OFF	OFF	OFF	ON	ON	OFF
OFF	OFF	OFF	ON	OFF	OFF	ON
OFF	OFF	OFF	ON	OFF	ON	OFF
OFF	OFF	OFF	ON	ON	OFF	ON
OFF	OFF	OFF	ON	ON	ON	ON
OFF	OFF	ON	OFF	OFF	OFF	ON
OFF	OFF	ON	OFF	OFF	ON	OFF
OFF	OFF	ON	OFF	ON	OFF	ON
OFF	OFF	ON	OFF	ON	ON	OFF
OFF	OFF	ON	ON	OFF	OFF	ON
OFF	OFF	ON	ON	OFF	ON	OFF
OFF	OFF	ON	ON	ON	OFF	ON
OFF	OFF	ON	ON	ON	ON	ON
OFF	ON	OFF	OFF	OFF	OFF	ON
OFF	ON	OFF	OFF	OFF	ON	OFF
OFF	ON	OFF	OFF	ON	OFF	ON
OFF	ON	OFF	OFF	ON	ON	OFF
OFF	ON	OFF	ON	OFF	OFF	ON
OFF	ON	OFF	ON	OFF	ON	OFF
OFF	ON	OFF	ON	ON	OFF	ON
OFF	ON	OFF	ON	ON	ON	ON
OFF	ON	ON	OFF	OFF	OFF	ON
OFF	ON	ON	OFF	OFF	ON	ON
OFF	ON	ON	OFF	ON	OFF	ON
OFF	ON	ON	OFF	ON	ON	ON
OFF	ON	ON	ON	OFF	OFF	ON
OFF	ON	ON	ON	OFF	ON	ON
OFF	ON	ON	ON	ON	OFF	ON
OFF	ON	ON	ON	ON	ON	ON

9.3. Lógica de funciones

Un sistema digital es aquel que solo puede manejar dos valores, 0 y 1. Esto es, por ejemplo un transistor, que es un componente electrónico analógico, puede hacer tres funciones: contacto cerrado, contacto abierto o saturación. Dado que el estado de saturación es utilizado para amplificar señales, al tener estados intermedios es analógico y en lógica digital no lo tendremos en cuenta, haciéndolos trabajar únicamente en los dos únicos estados de conmutación o de contacto, abierto o cerrado, es decir: "todo" o "nada".

Lo que antes en lógica de contactos eran circuitos eléctricos formados por contactos en serie y paralelo, en lógica de funciones, o lo que se denomina lógica digital, se utilizan funciones o puertas digitales representadas por bloques gráficos, que combinándolos dan lugar a circuitos lógicos todo lo complejos que sea necesario.

Cuando se asocian estas puertas formando un circuito lógico, se reduce a un número de entradas y en general una salida.

A partir de una tabla de verdad, se obtendrá la función algebraica que rige el sistema o proceso. Para implementarlo físicamente se utilizan las puertas o funciones lógicas. Se introducen en las entradas del sistema las variables que solo pueden tomar dos estados, 0 o 1 (nivel alto o nivel bajo, verdadero o falso), realizando la suma, el producto, la negación, la igualdad o las operaciones exclusivas según la función algebraica. Por supuesto, sabiendo el esquema lógico, podemos llegar a la función algebraica y a su tabla de verdad.

Una de las grandes diferencias, entre otras muchas, de la **electrónica digital** frente a la lógica de contactos o lógica cableada, es que las **conmutaciones son internas**, es decir, no hay conmutaciones físicas que producen desgaste en las superficies de los contactos eléctricos al estar formados por circuitos integrados en cuyo interior hay cientos, miles o millones de transistores.

Es muy importante saber que **por convenio** hay **dos tipos de lógica:**

- **Lógica positiva:** se trata de aquella notación que establece que el uno lógico (1) es el nivel más alto de tensión o valor verdadero, y cero lógico (0) el nivel más bajo o valor falso. Es el que utilizaremos en todo momento en la unidad.
- **Lógica negativa:** se trata de aquella notación que establece que un uno lógico (1) es el nivel más bajo de tensión o valor falso, y un cero lógico (0) es el nivel más alto o valor verdadero, por tanto opuesta a la anterior y que no usaremos en la unidad.

A continuación se muestran los operadores básicos de la electrónica digital, llamados funciones o puertas lógicas. Se presentan las tablas de verdad de cada función, que son tablas en las que se indica qué valor toma la salida de la puerta en función de las posibles combinaciones de las variables que se introducen en las entradas. Cada fila corresponde a una combinación diferente, que es igual a 2^n, donde n es el número de variables.

La composición y características de una tabla de verdad se especifican a continuación:

- Una columna por variable (n columnas).
- Una fila por cada combinación posible de variables (total: 2^n filas).
- Una columna adicional para registrar el valor que toma la función o salida para cada combinación de variables.
- El total es de $n + 1$ columnas · 2^n filas.
- La tabla de verdad de una función es única.

Para formarla se procede de la siguiente manera:

1. Se escriben en la primera fila las variables que haya y en la última la salida obtenida.
2. Se empieza desde la última variable y en su columna, en filas, se escribe 0, 1, 0, 1, 0, 1...
3. En la siguiente variable anterior a la última, en su columna y en filas se escribe el doble de ceros y de unos, 0, 0, 1, 1, 0, 0, 1, 1...
4. En la siguiente, el doble a la anterior y así hasta completar la tabla.
5. Se escribe en la salida qué combinaciones de las variables hacen uno lógico la función. El resto serán ceros lógicos.

9.3.1. Puerta lógica OR u "O"

Es la función que corresponde a la suma lógica de las entradas, es decir, su **salida** siempre es **uno lógico o nivel alto si por lo menos una de las variables de entrada es un uno lógico.** Pueden ser con dos entradas o más.

Su representación gráfica se puede hacer por los siguientes símbolos:

Figura 9.5. *Representación gráfica de la función OR u "O".*

Tabla 9.5. Tabla de verdad de la función OR u "O".

A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

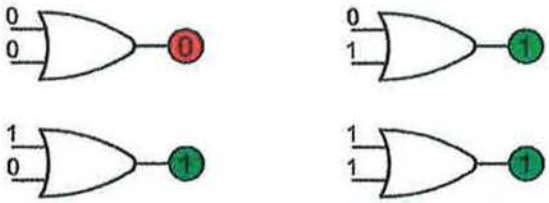


Figura 9.6. *Posibles estados de las entradas y salida de la función OR u "O" de dos variables.*

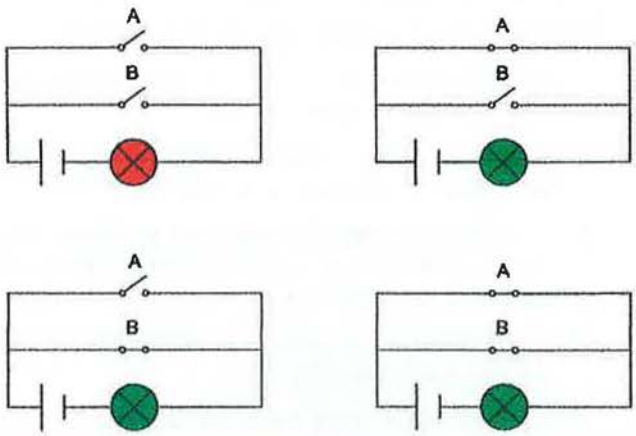


Figura 9.7. *Esquema eléctrico equivalente a la función OR u "O".*

9.3.2. Puerta lógica AND o "Y"

Es la función que corresponde al producto lógico de las entradas, es decir, su salida solo será uno lógico o nivel alto cuando todas las variables sean uno lógico.

Pueden ser con dos entradas o más.

Su representación gráfica puede ser dada por los siguientes símbolos:

Figura 9.8. *Representación gráfica de la función AND o "Y".*

Tabla 9.6. Tabla de verdad de la función AND o "Y".

A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

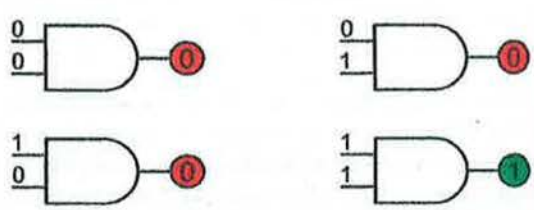


Figura 9.9. *Posibles estados de las entradas y salida de la función AND o "Y" de dos variables.*

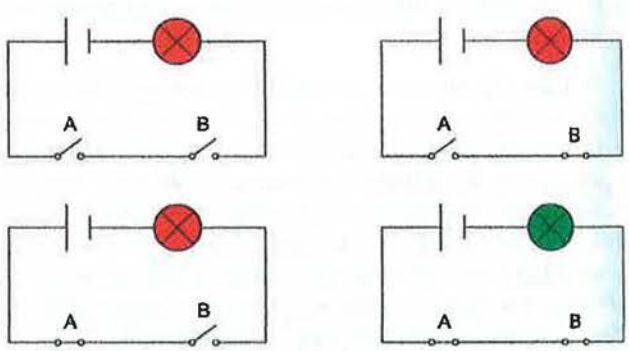


Figura 9.10. *Esquema eléctrico equivalente a la función AND o "Y".*

9.3.3. Puerta lógica NOT o "NEGACIÓN"

Es la función que corresponde a la negación de una variable, es decir, su salida siempre es el valor inverso de la entrada. La negación se representa con una línea encima de la variable o función, indicando que es el valor inverso o negado. También se dice que la variable negada es el complemento de la variable sin negar.

Solo tiene una entrada y una salida.

A veces, para negar las entradas de las puertas lógicas, y hacer más sencilla la visualización de los circuitos lógicos, se pone un **círculo de negación en la variable de entrada a la puerta** y así indicar que la **variable entra a la función lógica negada**.

Su representación gráfica puede ser dada por los siguientes símbolos:

Figura 9.11. *Representación gráfica de la función NOT o "NEGACIÓN".*

Tabla 9.7. **Tabla de verdad de la función NOT.**

A	S
0	1
1	0

Figura 9.12. *Posibles estados de las entradas y salida de la función NOT o "NEGACIÓN" de una variable.*

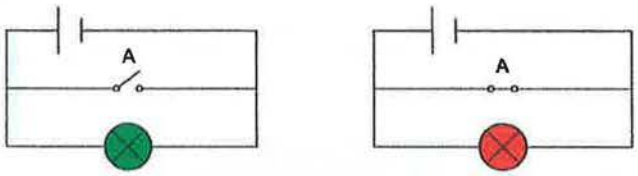


Figura 9.13. *Esquema eléctrico equivalente a la función NOT.*

SABÍAS QUE

La función lógica AND en lógica positiva actúa como una puerta lógica OR en lógica negativa, y una puerta OR en lógica positiva actúa como una puerta AND en lógica negativa.

Un inversor será siempre un inversor, aunque se use lógica positiva o negativa.

9.3.4. Puerta lógica NOR

Es la función que corresponde a la suma lógica de las entradas invertidas, es decir, su salida siempre es uno lógico o nivel alto si las dos variables de entrada son cero lógico o nivel bajo. En caso contrario, las salidas son cero lógico. Por tanto, se realiza la negación a la salida de la función OR.

Pueden ser con dos o más entradas.

Su representación gráfica puede ser por los siguientes símbolos:

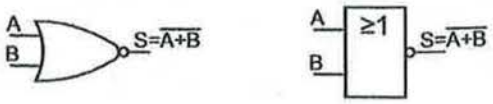


Figura 9.14. *Representación gráfica de la función NOR.*

Tabla 9.8. **Tabla de verdad de la función NOR.**

A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

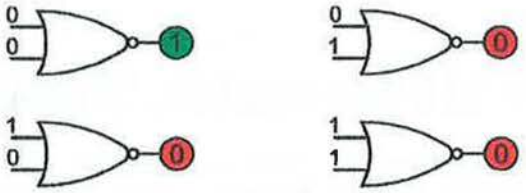


Figura 9.15. *Posibles estados de las entradas y salida de la función NOR de dos variables.*

9.3.5. Puerta lógica NAND

Es la función que corresponde a la negación del producto lógico de las entradas, es decir, su salida es siempre uno lógico si las entradas son simultáneamente distintas a uno lógico, en caso contrario la salida es cero lógico.

Por tanto, se realiza la negación a la salida de la función AND. Como veremos, el producto de dos variables se puede descomponer en la suma de las variables negada cada una de ellas independientemente.

Pueden ser con dos entradas o más.

Su representación gráfica puede ser por los siguientes símbolos:

Figura 9.16. *Representación gráfica de la función NAND.*

Tabla 9.9. **Tabla de verdad de la función NAND.**

A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

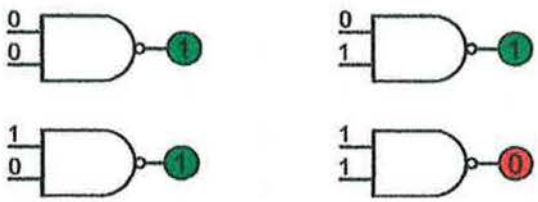


Figura 9.17. *Posibles estados de las entradas y salida de la función NAND.*

9.3.6. Puerta lógica X-OR u "OR EXCLUSIVA"

Es la función que corresponde a la suma directa de dos o más variables, es decir, **la salida es siempre uno lógico o nivel alto, si el número de unos de las entradas es impar,** en caso contrario, la salida es cero lógico o nivel bajo.

La suma directa de dos variables corresponde a la primera sin negar por la segunda variable negada más la primera negada por la segunda sin negar.

$$S = A \oplus B = \overline{A} \cdot B + A \cdot \overline{B}$$

Pueden ser con dos entradas o más.

Su representación gráfica puede ser por los siguientes símbolos:

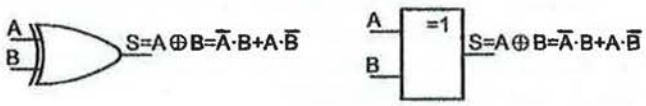


Figura 9.18. *Representación gráfica de la función X-OR u "OR EXCLUSIVA".*

Tabla 9.10. **Tabla de verdad de la función X-OR u "OR EXCLUSIVA".**

A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

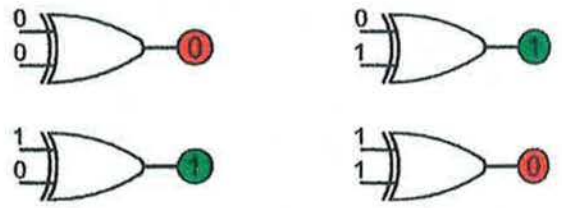


Figura 9.19. *Posibles estados de las entradas y salida de la función X-OR u "OR EXCLUSIVA" de dos variables.*

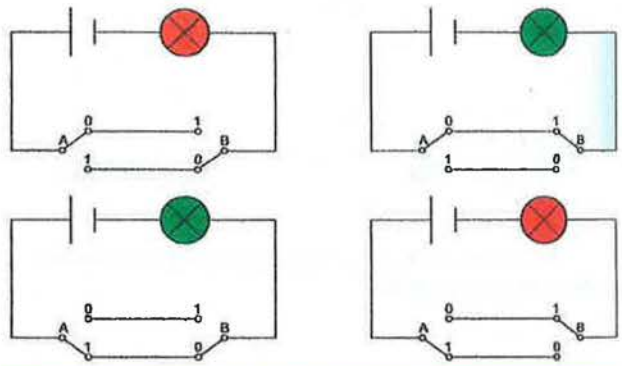


Figura 9.20. *Esquema eléctrico equivalente a la función X-OR u "OR EXCLUSIVA".*

9.3.7. Puerta lógica XNOR o "NOR EXCLUSIVA"

Es la función que corresponde a la **inversa de X-OR** (inversa de la suma directa de dos o más variables), es decir, **la salida es uno lógico o nivel alto, si el número de unos lógicos de las entradas es par.** En caso contrario, la salida es cero lógico o nivel bajo.

$$S = \overline{A \oplus B} = \overline{A} \cdot \overline{B} + A \cdot B$$

Figura 9.21. *Representación gráfica de la función XNOR o "NOR EXCLUSIVA".*

Tabla 9.11. **Tabla de verdad de la función XNOR o "NOR EXCLUSIVA".**

A	B	S
0	0	1
0	1	0
1	0	0
1	1	1

Observar, que cuando llegan dos ceros en la entrada, se entiende que el número de unos lógicos en la entrada es par al no haber ningún uno.

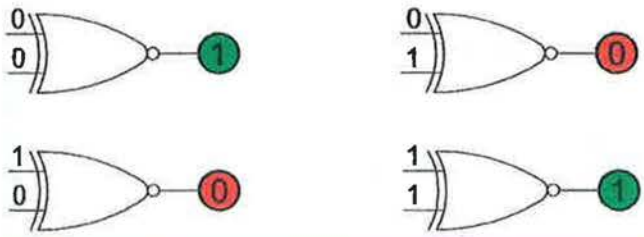


Figura 9.22. *Posibles estados de las entradas y salida de la función XNOR o "NOR EXCLUSIVA" de dos variables.*

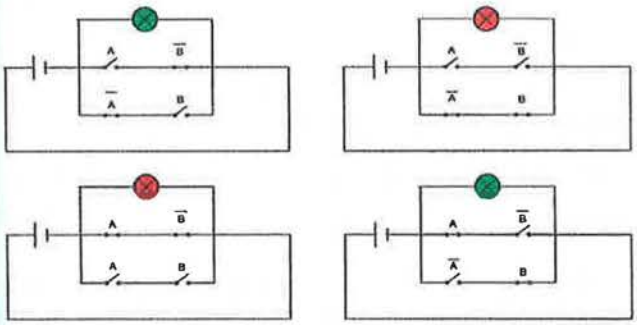


Figura 9.23. *Esquema eléctrico equivalente a la función XNOR o "NOR EXCLUSIVA".*

9.3.8. Puerta lógica IGUALDAD

Es la función que corresponde a dar en su salida el mismo valor lógico que tiene en la entrada. Se utiliza fundamentalmente para amplificadores digitales.

Figura 9.24. *Representación gráfica de la función IGUALDAD.*

Tabla 9.12. **Tabla de verdad de la función IGUALDAD.**

A	S
0	0
1	1

Actividad resuelta 9.1

Genera la tabla de verdad de la función X-NOR con tres entradas, e indica una aplicación donde se podría utilizar.

Solución:

Dado que existen tres variables, la puerta X-NOR es la suma directa de las tres variables, esto es, que su salida será un uno lógico cuando el número de unos en las entradas es par. Por tanto:

A	B	C	S
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Un uso práctico de esta puerta de dos o más variables es como comparador lógico, indicando que hay bits de entrada pares idénticos.

Estas puertas lógicas vistas anteriormente se venden en forma de circuitos integrados, denominándose en el argot electrónico "pastillas", en las que internamente tienen puertas lógicas internas (cuatro o más). Tienen los contactos o "patas" para conectar la alimentación, entradas y salidas a circuitos impresos, zócalos o placas de prueba (*protoboard* o *breadboard*).

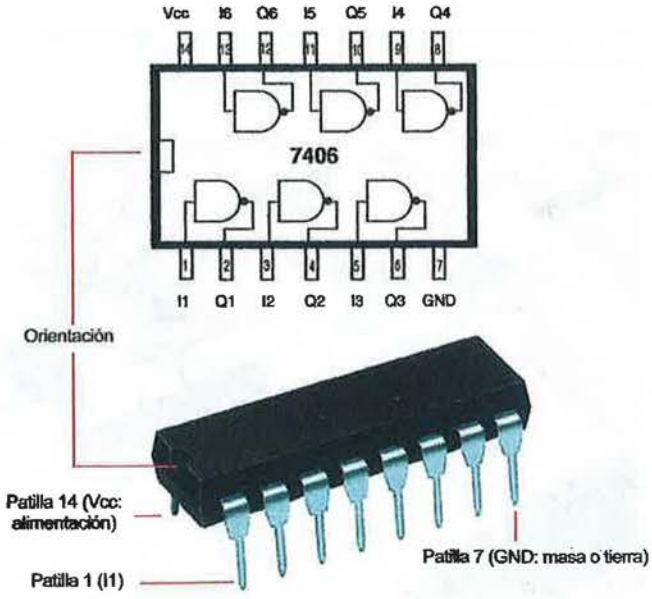


Figura 9.25. *Orientación e interpretación de uno de los circuitos integrados de tecnología TTL (Transistor-Transistor Logic) para implementación de puertas lógicas.*

SABÍAS QUE

Para introducir un uno lógico, nivel alto o valor verdadero, hay que aplicar una tensión de +5 V respecto a masa o tierra, mientras que para introducir un cero lógico, nivel bajo o valor falso hay que introducir 0 V respecto masa o tierra.

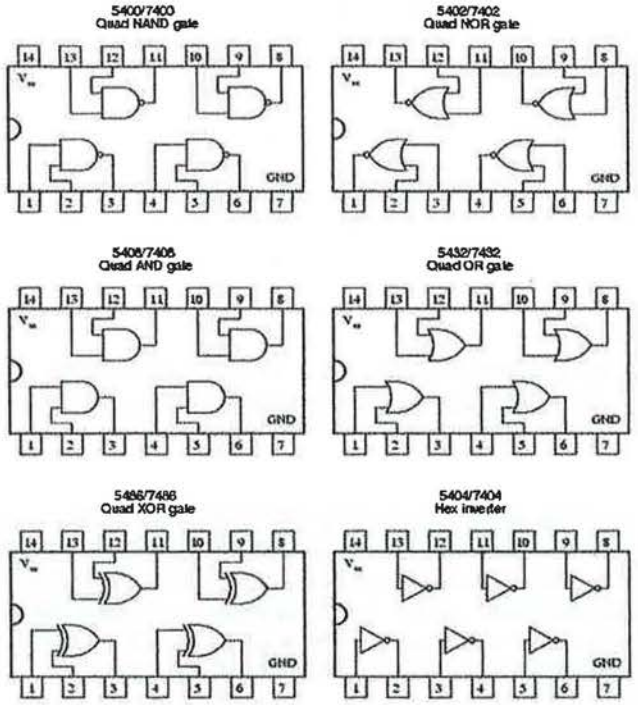


Figura 9.26. *Composición interna de algunos circuitos integrados de la familia TTL (Transistor-Transistor Logic).*

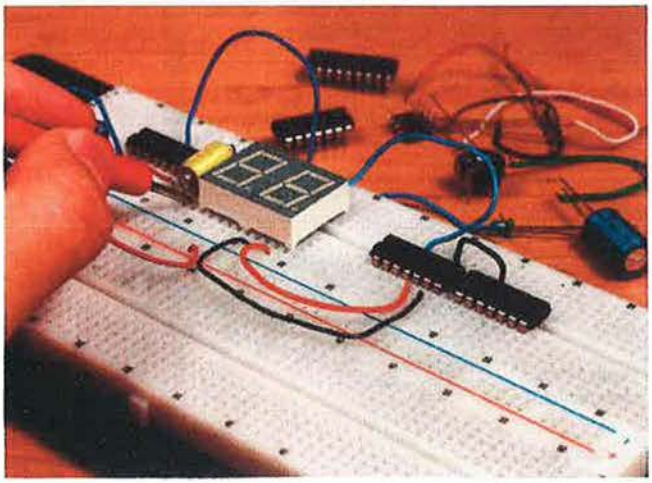


Figura 9.27. *Protoboard para realizar ensayos con componentes electrónicos, como los circuitos integrados con puertas lógicas para automatismos.*

9.4. Álgebra de Boole y teoremas de Morgan

George Boole fue un filósofo y matemático que fundamentó un sistema matemático, llamado álgebra de Boole, que mediante postulados, operaciones y relaciones lógicas fue capaz de solucionar problemas de automatismos y procesos, ya que con la manipulación y simplificación de la función inicial, realiza la misma tarea que con la ecuación primitiva.

SABÍAS QUE

El padre de George Boole, John Boole (1779-1848), era un comerciante de pocos recursos pero que se interesó por las matemáticas y la lógica y después enseñó a su hijo conocimientos básicos. Pero George Boole al principio mostró mayor interés por las humanidades demostrando más adelante su gran capacidad en las mismas disciplinas que su padre tuvo curiosidad.

El álgebra de Boole se distingue del álgebra convencional en que los elementos o variables que lo forman solo pueden **tomar dos estados, 0 y 1**, siendo números binarios o digitales.

9.4.1. Axiomas del álgebra de Boole

En álgebra, los **axiomas** son proposiciones que **no necesitan demostración**, por tanto sirven para establecer unas bases evidentes para poder demostrar otras ecuaciones más complejas. Los **axiomas de Boole son** por tanto **premisas que permiten demostrar los postulados que se verán a continuación.**

1. **Ley de clausura:** sus operadores son únicamente la suma y el producto lógicos, y se definen de la siguiente forma:

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

2. **Propiedad asociativa:**

$$A + (B + C) = (A + B) + C$$

$$A \cdot (B \cdot C) = (A \cdot B) \cdot C$$

3. **Propiedad conmutativa:**

$$A + B = B + A$$

$$A \cdot B = B \cdot A$$

4. **Propiedad distributiva:**

$$A \cdot (B + C) = A \cdot B + A \cdot C$$

$$A + B \cdot C = (A + B) \cdot (A + C)$$

5. **Elementos neutros:** existe un elemento neutro para la suma (0) y un neutro para el producto (1).

$$A + 0 = A$$

$$A \cdot 1 = A$$

6. **Complemento:** todo elemento o variable en álgebra de Boole tiene su **complemento o negado y es único**. El complemento de A se simboliza con una línea horizontal encima de la variable $\overline{A}$.

$$A + \overline{A} = 1$$

$$A \cdot \overline{A} = 0$$

Su tabla de verdad es:

A	$\overline{A}$
0	1
1	0

7. **Dualidad:** en todas las expresiones booleanas, se cumple el principio de dualidad, que significa que cualquier identidad algebraica se puede transformar en otra identidad válida intercambiado (+) por (·) y 1 por 0.

RECUERDA

El producto de variables equivale a la conexión en serie, mientras que la suma equivale a circuitos conectados en paralelo. Un uno lógico o nivel alto significa lo mismo, que eléctricamente es un contacto cerrado. Un cero lógico o nivel bajo equivale a un contacto abierto.

- El operador de adición o unión (+) equivale a la función OR.
- El operador de producto o intersección (·) equivale a la función AND.
- El operador de inversión o complementación ($\overline{\ }$) equivale a la función NOT.

De estos axiomas, que no tienen demostración, se deducen los siguientes **teoremas del álgebra de Boole** y por tanto pudiendo demostrar cada uno de ellos.

Teorema 1. Idempotencia. Para cada variable, se cumple:

$$A + A = A$$

$$A \cdot A = A$$

Teorema 2. Elementos nulos. Para cualquier elemento, se cumple:

$$A + 1 = 1$$

$$A \cdot 0 = 0$$

Teorema 3. Absorción. Para cada par de variables, se verifica:

$$A + A \cdot B = A$$

$$A \cdot (A + B) = A$$

Teorema 5. Involución. En el álgebra booleana, el complemento de A es $\overline{A}$. Por tanto:

$$\overline{\overline{A}} = A$$

Su tabla de verdad es:

$\overline{\overline{A}}$	A
0	0
1	1

Teorema 6. Cada elemento neutro es el complemento del otro.

$$\overline{0} = 1$$

$$\overline{1} = 0$$

Teorema 7. Para cada par de variables, se verifica:

$$A + \overline{A} \cdot B = A + B$$

$$A \cdot (\overline{A} + B) = A \cdot B$$

9.4.2. Teoremas de Morgan

Con los **teoremas de Morgan** se puede **convertir rápidamente una expresión booleana en *maxterms* y *minterms***. Además, permite la **eliminación de las barras de negación**.

La ley de Morgan generalizada dice que la inversa de una función se obtiene complementando cada una de las variables por separado que aparecen en ella y cambiando los operadores de suma (+) en (·) y viceversa.

$$\overline{A+B+C+\ldots+N} = \overline{A}\cdot\overline{B}\cdot\overline{C}\cdot\ldots\cdot\overline{N}$$

$$\overline{A\cdot B\cdot C\cdot\ldots\cdot N} = \overline{A}+\overline{B}+\overline{C}+\ldots+\overline{N}$$

Para cada par de variables, se verifica:

$$\overline{(A+B)} = \overline{A}\cdot\overline{B}$$

$$\overline{(A\cdot B)} = \overline{A}+\overline{B}$$

9.4.3. Expresiones y funciones booleanas

Una **expresión booleana** es una combinación de variables, constantes y operadores. Por ejemplo: $A\cdot B+\overline{B}\cdot C+\overline{A}\cdot 1$.

Las **funciones booleanas** son expresiones con variables: $f(A,B,C)=A\cdot B+\overline{B}\cdot C+\overline{A}\cdot 1$, de tal forma que en función de los valores que se den a las variables de la función, se obtiene un resultado, siendo en sistemas lógicos la variable de salida. Si se recogen todas las posibles combinaciones que pueden tomar las variables de la expresión y sus salidas, se obtiene la tabla de verdad de la función, para posteriormente obtener el circuito lógico que verifica la función.

Un ***minterm*** **es un término** de una función **formado por el producto de las variables** que componen la función, de tal forma que **cada variable aparece una sola vez** bien en forma normal o con su complemento o negación, es decir, **son sumas de productos.**

Ejemplo

$$S=f(A,B,C)=A\cdot B\cdot\overline{C}+\overline{A}\cdot B\cdot\overline{C}+A\cdot\overline{B}\cdot C$$

RECUERDA

El símbolo · se utiliza para indicar producto entre **letras o variables.**

El símbolo × es el operador que representa el producto de **números.**

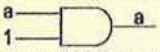
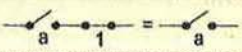
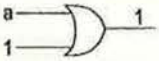
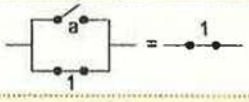
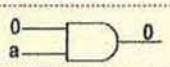
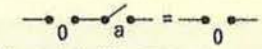
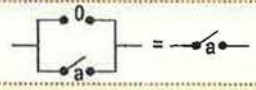
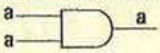
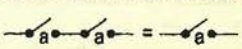
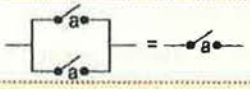
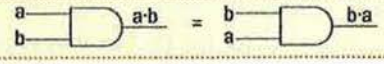
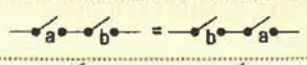
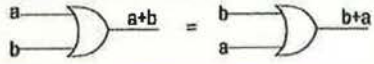
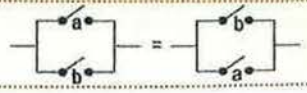
Un ***maxterm*** **es un término** de una función **formado por la suma de las variables** que componen la función, de tal forma que **cada variable aparece una sola vez** bien en forma normal o complementada, es decir, **son productos de sumas.**

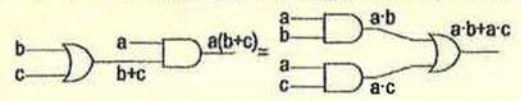
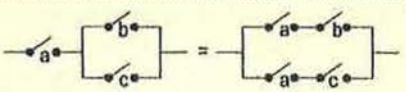
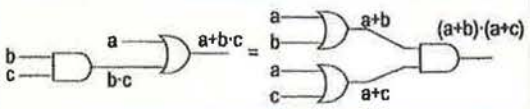
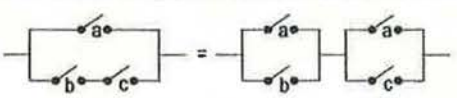
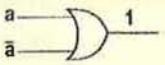
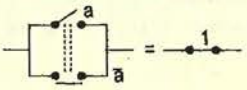
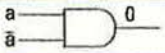
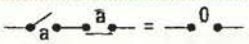
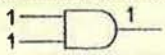
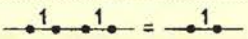
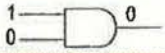
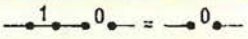
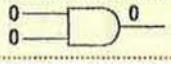
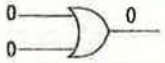
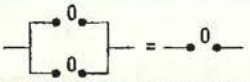
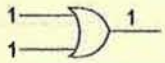
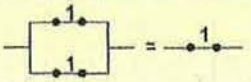
Ejemplo

$$S=f(A,B,C)=(A+B+\overline{C})\cdot(\overline{A}+B+\overline{C})\cdot(A+\overline{B}+C)$$

Para comprender adecuadamente los principios del álgebra o lógica de Boole, se indica la representación mediante puertas lógicas y contactos eléctricos, ya habituados a ello. Los axiomas, los teoremas y las leyes son prácticos para resolver las ecuaciones lógicas.

Tabla 9.13. Lógica de Boole con sus equivalentes lógicos y de contactos.

Álgebra de Boole	Representación con puertas lógicas	Representación con contactos
$a\cdot 1=a$		
$a+1=1$		
$0\cdot A=0$		
$0+a=a$		
$a\cdot a=a$		
$a+a=a$		
$a\cdot b=b\cdot a$		
$a+b=b+a$		

Álgebra de Boole	Representación con puertas lógicas	Representación con contactos
$a+b+c=(a+b)+c=$ $=a+(b+c)=(a+c)+b$		
$a\cdot b\cdot c=(a\cdot b)\cdot c=$ $=a\cdot(b\cdot c)=(a\cdot c)\cdot b$		
$a(b+c)=a\cdot b+a\cdot c$		
$a+b\cdot c=(a+b)\cdot(a+c)$		
$a+\bar{a}=1$		
$a\cdot\bar{a}=0$		
$\bar{\bar{a}}=a$		
$a=b \quad \bar{a}=\bar{b}$		
$1\cdot 1=1$		
$1\cdot 0=0$		
$0\cdot 0=0$		
$0+0=0$		
$1+1=1$		
$\bar{0}=1$		
$\bar{1}=0$		

Actividad resuelta 9.2

Simplifica la siguiente función booleana:

$$S=\bar{A}\cdot\bar{B}\cdot\bar{C}+\bar{A}\cdot B\cdot\bar{C}+A\cdot\bar{B}\cdot\bar{C}+A\cdot\bar{B}\cdot C$$

Solución:

Sacando factor común dos a dos sumandos ($\bar{A}\cdot\bar{C}$ y $A\cdot\bar{B}$) tenemos:

$$\bar{S}=\bar{A}\cdot\bar{C}\cdot(\bar{B}+B)+A\cdot\bar{B}\cdot(\bar{C}+C)$$

$$S=\bar{A}\cdot\bar{C}+A\cdot\bar{B}$$

Actividad propuesta 9.5

Simplifica las siguientes funciones:

a) $S=A\cdot\bar{B}\cdot C+\bar{A}\cdot B\cdot C+A\cdot B\cdot C$

b) $S=(\bar{A}+A\cdot B+A\cdot B\cdot\bar{C})\cdot(A+B+C)$

c) $S=A\cdot\bar{B}+B\cdot\bar{A}+A\cdot\bar{C}+AC+\bar{A}$

d) $S=\overline{\bar{B}\cdot(A+C)+B\cdot(\bar{A}\cdot C+A\cdot\bar{C})}$

9.5. Obtención del circuito lógico a partir de una tabla de verdad y viceversa

La electrónica digital es la tecnología en la que está fundamentada la automatización mediante automatismos programables industriales (API). Una vez tenemos los valores que se deben obtener en función de unas entradas determinadas, el objetivo es obtener el circuito lógico en el que se verifican esos valores de salida para todas las combinaciones posibles.

También podemos llegar a la tabla de verdad conociendo el circuito lógico. A continuación de muestran algunos ejemplos:

Obtención de un circuito lógico a partir de la tabla de verdad

A partir de la siguiente tabla de verdad:

Entradas			Salidas
A	B	C	S
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

En primer lugar, obtenemos la función lógica a partir de la tabla de verdad. Para ello, se siguen los siguientes pasos:

1. Nos fijamos en las salidas en las que se verifica (valor 1 lógico) la función.
2. En estas filas donde tenemos uno lógico, escribimos la expresión en *minterms* (sumas de productos) en las que cada término es el producto del estado de las variables. Sumando todos los términos de cada fila, obtenemos la función booleana de salida.

$$S = \overline{A} \cdot \overline{B} \cdot \overline{C} + \overline{A} \cdot B \cdot \overline{C} + A \cdot B \cdot \overline{C}$$

Una vez obtenida esta expresión, dibujamos el esquema lógico correspondiente, sabiendo que las variables con la línea encima es el complemento y por tanto la negación o una puerta NOT, los productos puertas AND, y las sumas puertas OR.

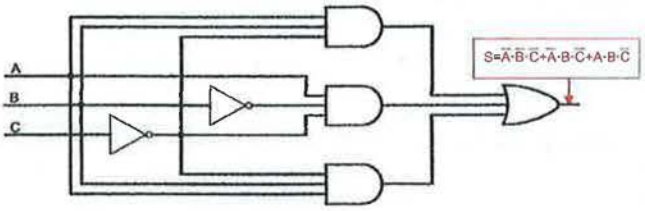


Figura 9.28. *Circuito lógico de la expresión.*

Obtención de la tabla de verdad a partir del circuito lógico

Para obtener la tabla de verdad del circuito lógico, primeramente se marcan las salidas de cada puerta con X1, X2 y X3. La salida será la función booleana que verifica el circuito, denominada S. En cada salida de cada puerta lógica, escribimos la operación que realiza, y la última puerta, al ser AND, sumamos todos los términos que se obtengan (S = X1 + X2 + X3).

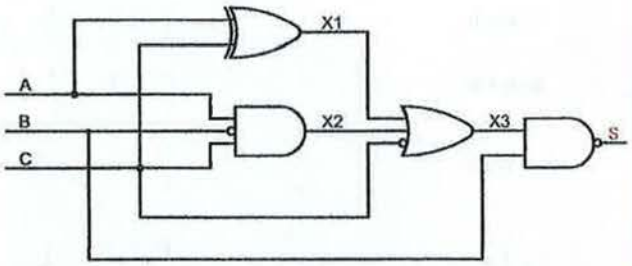


Figura 9.29. *Circuito lógico del que se desea obtener su tabla de verdad.*

RECUERDA

La negación a una entrada se puede representar sin necesidad de utilizar la función NOT en la entrada de la puerta lógica con un círculo.

$$X1 = \overline{A} \cdot C + A \cdot \overline{C}$$

$$X2 = A \cdot \overline{B} \cdot C$$

$$X3 = \overline{A} \cdot C + A \cdot \overline{C} + A \cdot \overline{B} \cdot C + \overline{C}$$

Siendo, por tanto, la función algebraica que rige el circuito lógico:

$$S = (\overline{A} \cdot C + A \cdot \overline{C} + A \cdot \overline{B} \cdot C + \overline{C}) \cdot B$$

Obtenida la expresión algebraica, dando valores a las variables A, B y C, para todas las combinaciones posibles, obtenemos las filas de la columna de salida, quedando completa la tabla de verdad.

Entradas			Salidas
A	B	C	S
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Actividad propuesta 9.6

Realiza la tabla de verdad de las siguientes funciones:

a) $\overline{A} \cdot \overline{B} \cdot \overline{C} \cdot D + \overline{A} \cdot \overline{B}$

b) $\overline{A} \cdot B \cdot C \cdot D + A \cdot \overline{B} \cdot \overline{C} \cdot D + A \cdot C \cdot B + A \cdot B \cdot C \cdot \overline{D}$

c) $(\overline{A} + \overline{B} + C) \cdot (\overline{A} + B + C) \cdot (A + B + C)$

9.6. Simplificación de funciones booleanas

Mediante una tabla de verdad se puede obtener una expresión algebraica, y como se ha visto, podemos obtener directamente el circuito lógico que verifica esa función. Además, con las propiedades y teoremas de Boole y de Morgan, se puede transformar la función booleana y simplificar en gran medida la expresión y esto implica utilizar menos puertas lógicas.

Tabla 9.14. Diferentes formas de expresar funciones digitales.

Tabla de verdad	↔	Expresión algebraica (función booleana)
Expresión algebraica (función booleana)	↔ Simplificar la función aplicando axiomas, postulados de Boole y teoremas de Morgan ↔	Circuito o diagrama lógico (puertas lógicas) (mínimo número de puertas lógicas posible para su implementación física del circuito)

En electrónica digital, el objetivo del diseñador es encontrar el circuito lógico que, realizando la misma función, cumpla idénticamente la tabla de verdad, ya que esta, sí es única para todas las expresiones algebraicas que verifiquen la tabla de verdad

9.6.1. Métodos de simplificación de funciones

A veces, se conoce el circuito lógico y lo que interesa es extraer de él la función lógica que cumpla todas las combinaciones posibles como se ha visto anteriormente, y simplificarla lo máximo posible para que la implementación sea más sencilla.

Para ello, existen varios métodos que dependen de la complejidad de la tabla de verdad o función algebraica. Estas técnicas pueden ser:

1. Observando la tabla de verdad, puede simplificarse el circuito lógico directamente si él o parte de él realiza la función equivalente a alguna de las puertas lógicas conocidas. Este método requiere práctica y es el menos sistemático.
2. Sacar factor común y aplicar los teoremas de la función algebraica deducida directamente de la tabla de verdad. Este método no siempre es lo suficientemente potente para obtener la función óptima.
3. Simplificación mediante procedimientos sistemáticos que son sencillos de aplicar, pero requieren que la función esté en forma de *minterms* o *maxterms*, siendo sencillas de obtener a partir de la tabla de verdad. Existen dos métodos, que son:
   - Diagramas o mapas de **Karnaugh**, utilizados para funciones de hasta seis variables, obteniendo una forma de la expresión con menor número de términos y variables posibles.
   - Método de **Quine-McClustey**, el cual no trataremos, ya que es utilizado para más de seis variables y está basado en programación por ordenador.

A continuación se explica la manera de simplificar las **funciones lógicas booleanas en forma de *minterms* mediante el diagrama de Karnaugh de hasta seis variables, partiendo de la tabla de verdad.**

Además de todo esto e independientemente del método utilizado, todo circuito lógico se puede implementar mediante puertas NAND o NOR exclusivamente, para así emplear el menor número de circuitos integrados y además por ser los más baratos.

9.6.2. Simplificación de funciones lógicas mediante diagramas o mapas de Karnaugh

El diagrama o mapa de Karnaugh consiste en simplificar mediante teoremas, pero de un modo visual, elaborando un diagrama, que no es otra cosa que una transcripción de la tabla de verdad con un formato determinado para poder aplicar una sistemática generalizada que simplifique la función gráfica y su máxima expresión.

Aunque los diagramas de Karnaugh son prácticos hasta seis variables, se verá únicamente la simplificación de funciones hasta cuatro variables. El objetivo es familiarizarse y conocer el proceso de simplificación mediante esta metodología.

El formato del diagrama de Karnaugh y la posición de las variables en las casillas es la siguiente, no pudiendo ser de otra manera: para una función lógica con dos variables, se realiza una tabla o cuadrícula con la siguiente configuración, de tal forma que en la primera columna se pone la variable A en sus dos estados posibles, y en la primera fila, la variable B con los dos estados que puede tomar, ambas variables, 0 o 1.

Tabla 9.15. Formato del diagrama de Karnaugh para funciones lógicas de dos variables.

A \ B	$\bar{B}$	B
	0	1
$\bar{A}$	0	
A	1	

En el caso de tener funciones booleanas con tres variables, en la primera columna se pone el estado de las dos primeras variables, realizando las combinaciones posibles tal cual se muestran a continuación, y en la primera fila, los valores posibles de la tercera variable.

RECUERDA

Las tablas de Karnaugh están basadas en teoremas matemáticos y no se puede alterar el orden de la colocación de las variables.

Tabla 9.16. Formato del diagrama de Karnaugh para funciones lógicas de tres variables.

A B \ C	$\bar{C}$	C
	0	1
$\bar{A}\ \bar{B}$	00	
$\bar{A}$ B	01	
A B	11	
A $\bar{B}$	10	

En el caso de tener cuatro variables en la función lógica, se pone en la primera columna los valores que pueden adoptar las dos primeras variables y en la primera fila las otras dos, con las combinaciones que se muestran en la siguiente tabla.

Tabla 9.17. Formato del diagrama de Karnaugh para funciones lógicas de cuatro variables.

A B \ C D	$\bar{C}\ \bar{D}$	$\bar{C}$ D	C D	C $\bar{D}$
	00	01	11	10
$\bar{A}\ \bar{B}$	00				
$\bar{A}$ B	01				
A B	11				
A $\bar{B}$	10				

SABÍAS QUE

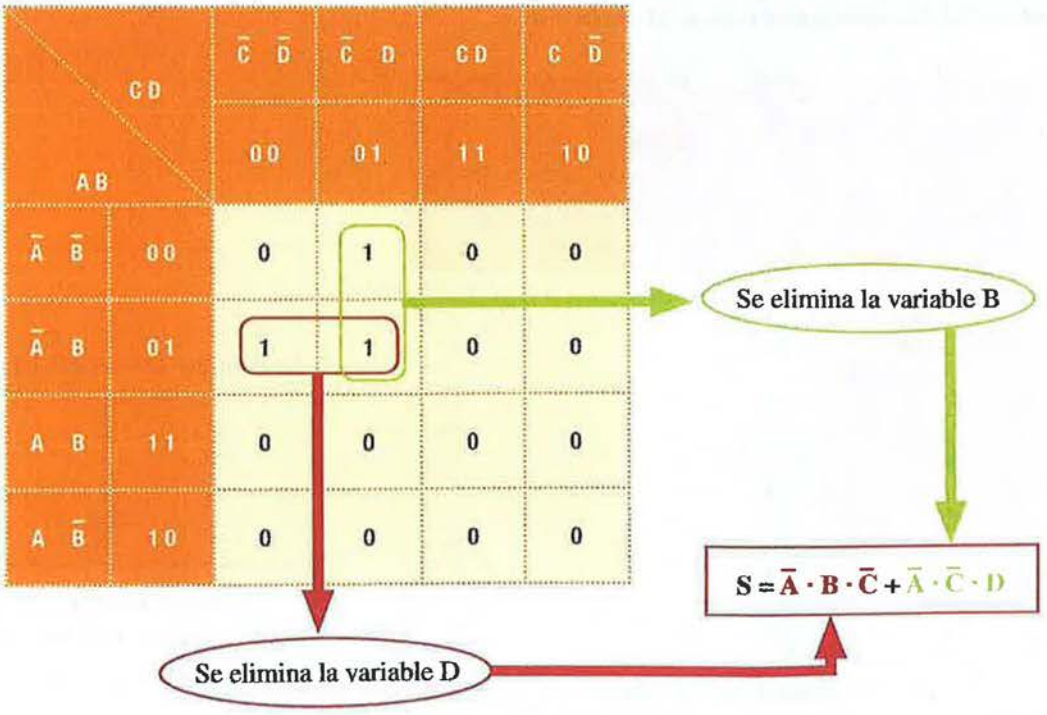
La posición de las variables en el diagrama de Karnaugh es diferente a la colocación de la tabla de verdad, ya que este corresponde al código de Gray, el cual es similar al binario pero de un número al siguiente solo varía un bit.

Una vez se distinguen los diferentes formatos y respetando el orden de las combinaciones, se muestra el proceso a seguir:

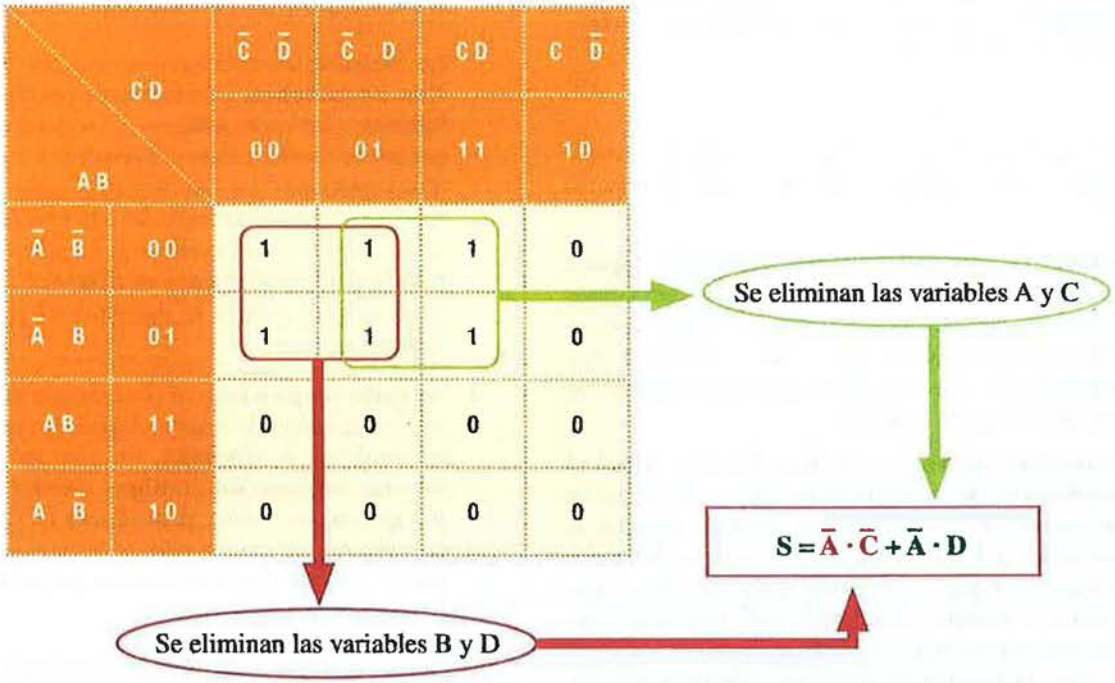
1. **Obtener la tabla de verdad de la función lógica.** Esta puede venir dada directamente, o si conocemos la función que queremos simplificar, si es posible, se obtienen los valores en que la salida es uno lógico para todas las combinaciones posibles como se ha realizado en casos anteriores.
2. **Transcribir los valores de la tabla de verdad al diagrama de Karnaugh.** Para ello, observamos de qué combinaciones de las variables se obtiene el uno lógico y los ponemos en las casillas del mapa de Karnaugh que corresponda según los valores que tomen las variables, de acuerdo con el diagrama que corresponda según el número de variables. En caso de tener la función en *minterms*, por cada término se pone un uno lógico en el diagrama. El resto de casillas, obviamente, serán cero.
3. **Agrupar los unos lógicos en grupos de 2, 4, 8, 16... formando un lazo**, es decir, los grupos a formar serán de 2^n unos lógicos, siendo $n = 1, 2, 3, 4...$ Para 2 variables, **el grupo máximo de unos lógicos** que se puede agrupar si la posición de unos lo permite, es de 4, para 3 variables, 8, para 4 variables, 16, y para 5 y 6 variables, 32 y 64 respectivamente.

   La forma de agrupar los unos lógicos puede ser **en filas, en columnas y en forma de matriz (filas y columnas)** pudiendo **utilizar unos comunes a otros grupos** e incluso **entre laterales y esquinas del diagrama opuestos**. Es decir, es como si el diagrama fuera una proyección de una esfera, donde los laterales y esquinas coinciden. Lo que **no está permitido es formar grupos en diagonal**. Ejemplos de formación de grupos se muestran en las siguientes figuras.
4. **De cada grupo o lazo** se observa qué variable o variables cambian de estado. **Aquella o aquellas que se cambien, se eliminan**. En caso de quedar **más de una variable sin eliminar dentro de un mismo grupo, se realiza el producto de las variables** de cada grupo, obteniendo un término que se suma con el resto de términos de otros grupos formando la función simplificada.

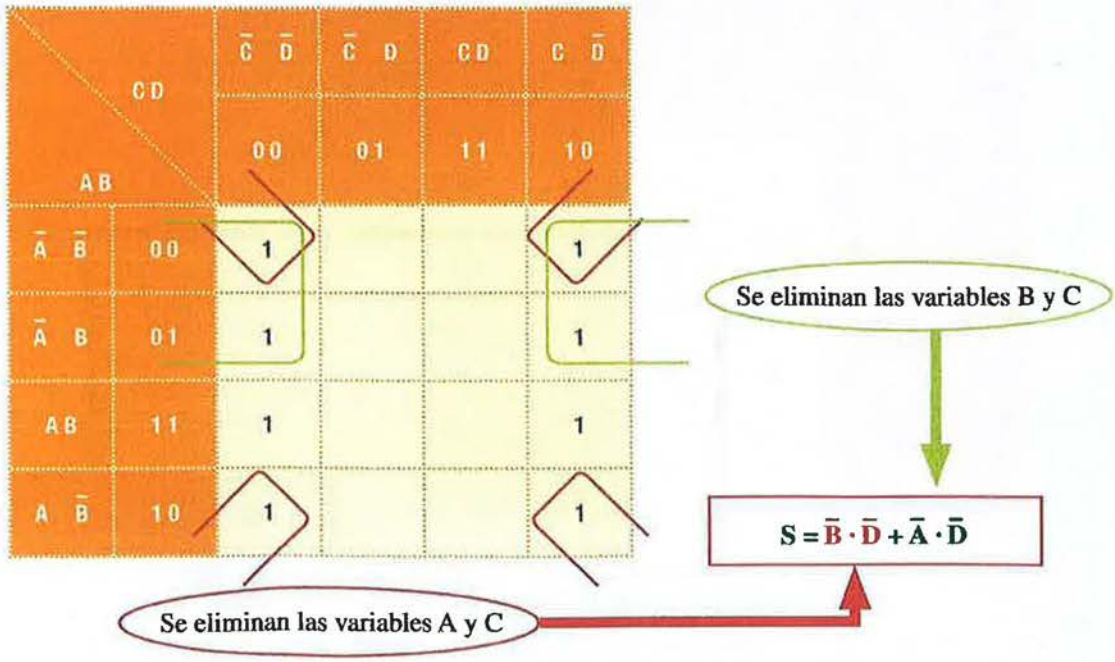
   En el siguiente diagrama de Karnaugh se muestra un ejemplo de agrupación de dos lazos con dos unos lógicos cada uno. En el grupo rojo, al cambiar de estado la variable D, se elimina, quedando $\overline{A} \cdot B \cdot \overline{C}$, y en el lazo verde formado por dos unos, se elimina la variable B al cambiar de estado, quedando $\overline{A} \cdot \overline{C} \cdot D$.



En el siguiente mapa de Karnaugh se muestra un ejemplo de agrupación de dos lazos en forma de matriz, formados por cuatro unos lógicos cada uno. En el grupo rojo, al cambiar de estado la variable B y D, se eliminan, quedando $\bar{A} \cdot \bar{C}$, y en el lazo verde formado por cuatro unos, se elimina la variable B y C al cambiar de estado, quedando $\bar{A} \cdot D$.



En el siguiente diagrama de Karnaugh se muestra un ejemplo de agrupación de dos lazos en forma de matriz, formados por cuatro unos lógicos cada uno. **El grupo rojo, está formado por las esquinas del diagrama, como si las esquinas se juntaran en un único punto habiendo continuidad entre ellas formando una esfera.** Al cambiar de estado, las variables A y C se eliminan, quedando $\bar{B} \cdot \bar{D}$. **El lazo verde está formado por cuatro unos, formados entre ambos laterales derecho e izquierdo, ya que están comunicados entre sí**, eliminándose las variables B y C al cambiar de estado, quedando $\bar{A} \cdot \bar{D}$.



Actividad resuelta 9.3

Simplifica la siguiente función lógica:

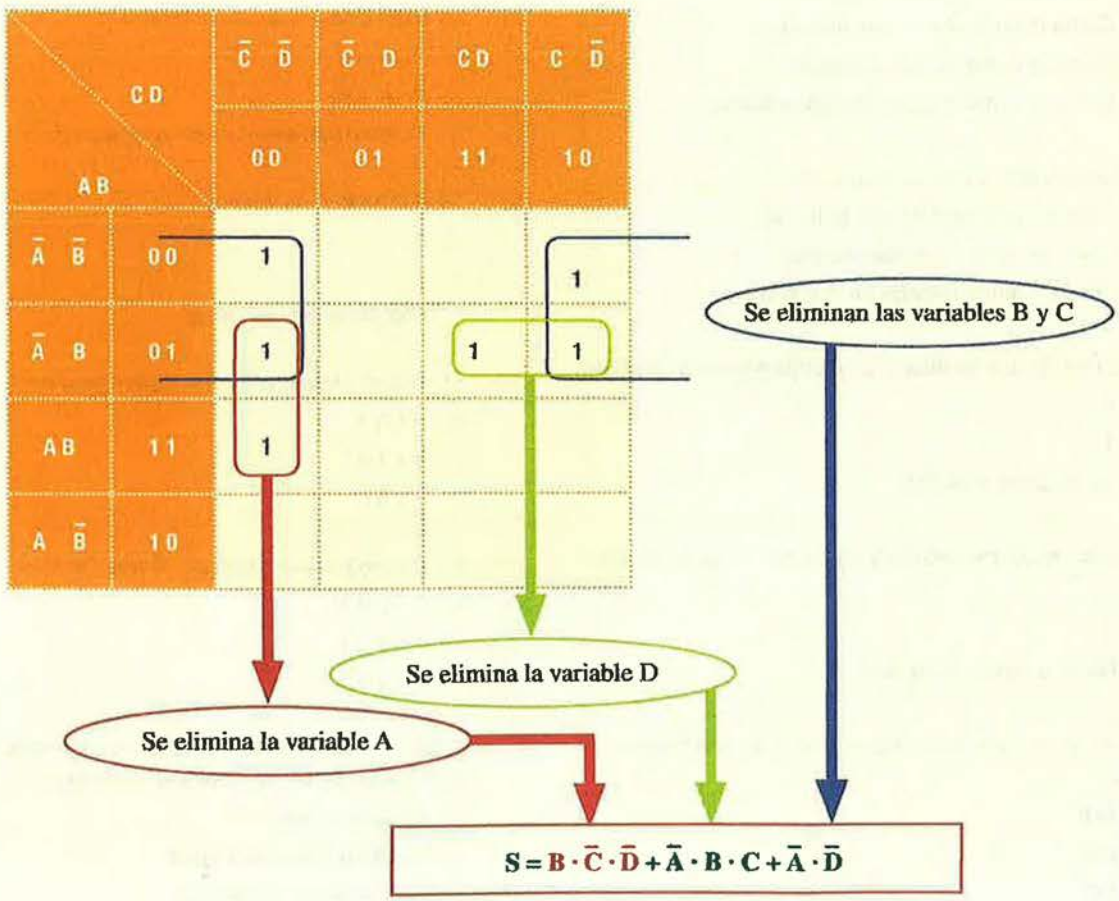
$$S=\bar{A}\cdot\bar{B}\cdot\bar{C}\cdot\bar{D}+\bar{A}\cdot\bar{B}\cdot C\cdot\bar{D}+\bar{A}\cdot B\cdot\bar{C}\cdot\bar{D}+\bar{A}\cdot B\cdot C\cdot\bar{D}+\bar{A}\cdot B\cdot C\cdot D+A\cdot B\cdot\bar{C}\cdot\bar{D}$$

Solución:

La tabla de verdad será la siguiente, siendo la salida 1 para cada uno de los términos de la expresión, teniendo en cuenta que una variable es 1 cuando está escrita normal y 0 cuando tiene el complemento encima de la variable.

Entradas				Salidas
A	B	C	D	S
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1

Entradas				Salidas
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	1
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0



Actividades de comprobación

9.1. ¿A qué número en hexadecimal corresponde el número decimal 13?

a) 13.
b) A.
c) D.

9.2. ¿El complemento a-1 del número 5 es?

a) 1 0 1.
b) 0 1 0.
c) 1 1 0.

9.3. ¿Cómo se representa el número negativo -6?

a) 0 1 1 0.
b) 1 1 1 0.
c) 1 0 1 0.

9.4. La función lógica AND se representa:

a) Como el producto de sus entradas.
b) Como la suma de sus entradas.
c) Como la suma negada de sus entradas.

9.5. La función lógica OR se representa:

a) Como el producto de sus entradas.
b) Como la suma de sus entradas.
c) Como la suma negada de sus entradas.

9.6. La suma de una variable y su complemento da siempre:

a) 0.
b) 1.
c) No se pueden sumar.

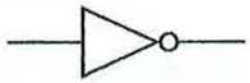
9.7. El producto de una variable y su complemento da siempre:

a) 0.
b) 1.
c) No se pueden multiplicar.

9.8. ¿Cuál de las siguientes opciones no es una función lógica?

a) AND.
b) RO.
c) NOT.

9.9. ¿Qué representa el símbolo?



a) Una función AND.
b) Una función OR.
c) Una función Inversor

9.10. ¿Cuál de las siguientes expresiones es correcta?

a) $A + B = B \cdot A$.
b) $A + 1 = \overline{A} \cdot 0$.
c) $A + B = B + A$.

9.11. La expresión $\overline{A \oplus B}$ es igual a:

a) $\overline{A} \cdot \overline{B} + A \cdot B$.
b) $\overline{A} \cdot B + A \cdot \overline{B}$.
c) $\overline{A} \cdot \overline{B}$.

9.12. La expresión $A + A \cdot B$ es igual a:

a) A.
b) $\overline{A} \cdot A + \overline{B}$.
c) El resultado siempre toma el valor 1.

9.13. La elemento $\overline{0}$ es igual a:

a) 0.
b) 1.
c) Ese elemento no existe.

9.14. El número binario 0101 en código Gray es:

a) 1 1 0 1.
b) 0 1 1 1.
c) 0 1 0 0.

9.15. El número binario 10110 en código Gray es:

a) 0 1 0 0 1.
b) 1 0 1 1 1.
c) 1 1 1 0 1.

9.16. ¿Qué método permite convertir rápidamente una expresión booleana en *maxterms* o *minterms*?

a) El código Gray.
b) Los diagramas de Kernel.
c) Los teoremas de Morgan.

Actividades de aplicación

9.1. ¿Cuáles son los números que integran el sistema octal?, ¿y el sistema hexadecimal?

9.2. En la conversión de un sistema de numeración a otro diferente, ¿qué significa el bit más significativo y el menos significativo?

9.3. Explica cómo realizarías la conversión de un número decimal a octal.

9.4. ¿Cómo se representa un número negativo?

9.5. ¿Cómo se obtiene el complemento a 2 de un número decimal? ¿Para qué se emplea el complemento a 2 de un número?

9.6. ¿En qué se diferencia el código Gray del código binario?

9.7. ¿Cuál es la diferencia entre la lógica positiva y la lógica negativa?

9.8. Indica la ecuaciones lógicas de las puertas lógicas, con dos entradas: OR, AND, NOR, NAND, XOR, XNOR; y con una entrada: NOT

9.9. Según la Ley de Morgan, ¿cuáles son los elementos neutros? Pon un ejemplo.

9.10. ¿En qué se basa el principio de dualidad según el álgebra de Boole?

9.11. ¿Qué es un *minterms* y un *maxterms* según las leyes de Morgan?

9.12. ¿Para qué se emplean los diagramas de Karnaugh?

9.13. Convierte a binario los números:

a) 123_{10}

b) 84_{10}

c) 285_{10}

d) 526_{10}

9.14. Convierte a octal los números:

a) 261_{10}

b) 462_{10}

c) 4037_{10}

d) 6654_{10}

9.15. Convierte de hexadecimal a decimal:

a) $2A4_{16}$

b) $B8C2_{16}$

c) 920_{16}

d) $T31AF_{16}$

9.16. Convierte de octal a binario:

a) 125_{8}

b) 701_{8}

c) 436_{8}

d) 260_{8}

9.17. Representa en un diagrama de Karnaugh las siguientes funciones:

a) $f(A,B) = \overline{A} + AB$

b) $f(A,B,C) = \overline{A}\overline{B}\overline{C} + BC + A\overline{B}\overline{C}$

c) $f(A,B,C,D) = \overline{A}\overline{B} + CD + AB\overline{C}\overline{D}$

d) $f(A,B,C,D) = \overline{B}\overline{C}\overline{D} + \overline{A}B\overline{C}D + BC + \overline{A}C\overline{D}$

9.18. Simplifica por Karnaugh las funciones:

a) $f(A,B,C,D) = AB\overline{C}\overline{D} + AB\overline{C}D + A\overline{B}\overline{C}\overline{D} + A\overline{B}\overline{C}D + A\overline{B}C\overline{D} + \overline{A}\overline{B}C\overline{D} + \overline{A}BC\overline{D} + \overline{A}BCD + \overline{A}\overline{B}C\overline{D} + \overline{A}B\overline{C}\overline{D}$

b) $f(A,B,C,D) = \overline{A}\overline{B}CD + \overline{A}B\overline{C}\overline{D} + \overline{A}B\overline{C}D + AB\overline{C}D + \overline{A}BCD + \overline{A}BC\overline{D} + A\overline{B}\overline{C}\overline{D} + A\overline{B}CD$

9.19. Realiza el esquema lógico de las siguientes funciones booleanas:

a) $S = \overline{A} \cdot B \cdot C + \overline{B}(A \cdot \overline{C} + C)$

b) $S = (\overline{A} \cdot \overline{B} + A \cdot C) \cdot (\overline{A} + \overline{B} \cdot C)$

c) $S = (A + \overline{B} + \overline{C}) \cdot (\overline{A} + B \cdot C)$

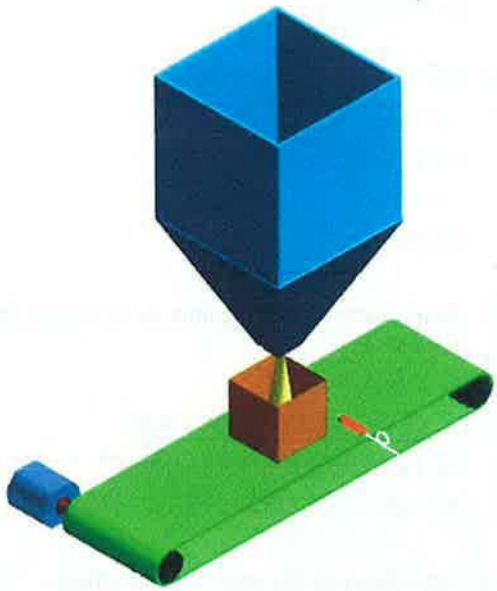
9.20. Representa el esquema lógico y de contactos de la expresión: $S = (A + C) \cdot B + B \cdot \overline{C}$

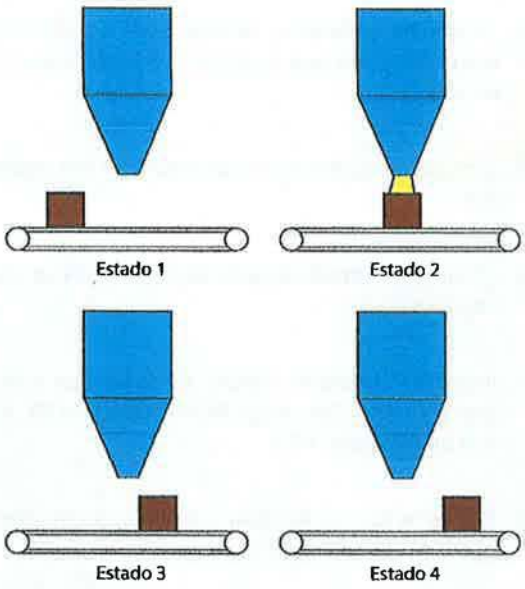
Casos prácticos

9.1. A continuación se representa un proceso industrial en el que el funcionamiento de todos sus componentes se expresa en sistema binario (1 en marcha/activado – 0 paro).

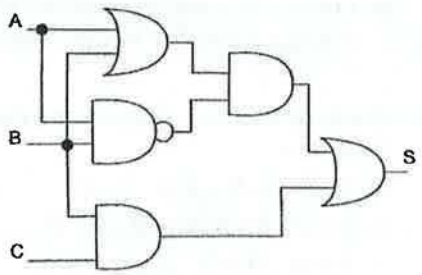
Analiza el proceso, explica su funcionamiento y realiza los esquemas de potencia y de maniobra completos asociados al mismo.

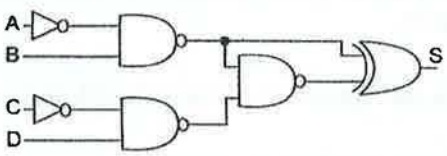
Tolva	0	0	0	0	0	1	1	1	0	0	0	0
Motor	0	0	1	1	1	0	0	0	1	1	1	1
Detector	0	0	0	0	0	1	1	1	1	1	0	0
Temporizador	0	0	0	0	0	1	1	1	0	0	0	0
	Estado 1					Estado 2			Estado 3		Estado 4	





9.2. De los circuitos lógicos representados, obtén:

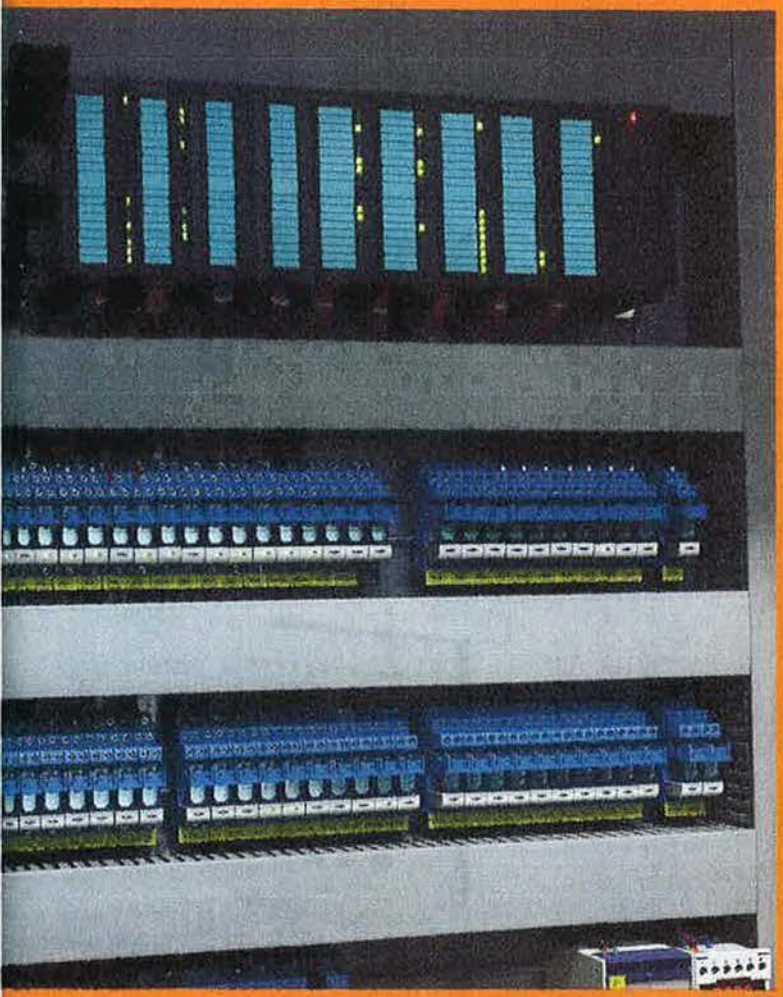




a) La ecuación de la salida en *minterms*.
b) La tabla de verdad que rige el sistema.
c) El diagrama o mapa de Karnaugh.
d) Obtén a partir del apartado anterior las ecuaciónes simplificadas en suma de productos y productos de sumas.
e) Dibuja el esquema lógico de las ecuaciones en *minterms* y *maxterms* simplificadas.

El autómata programable

10



Los autómatas programables han supuesto una revolución en aquellas tareas donde se requiere automatizar un proceso. No importa si el proceso es simple o complejo, existe una diversidad de versiones de autómatas con lo que es posible encontrar el adecuado.

La siguiente fase de la instalación del autómata programable es su programación. Existen varias maneras de realizar dicha programación. Con la programación se consigue que el sistema responda adecuadamente a las instrucciones marcadas por el técnico.

Contenidos

Objetivos

- Aprender qué es un autómata programable y qué se puede conseguir con él.
- Conocer las diversas partes que lo componen.
- Saber realizar el montaje de un circuito con lógica programada.
- Conocer los diversos lenguajes de programación orientados a los autómatas.
- Aprender el lenguaje de programación de bloques funcionales.
- Aprender el lenguaje de programación de contactos.
- Conocer el lenguaje de programación de lista de instrucciones.

10.1. Los automatismos programados

A la hora de diseñar un automatismo hay dos caminos: realizar el automatismo mediante **tecnología cableada**, en la cual la manera de funcionar depende de los elementos empleados y la forma de interactuar entre ellos (tarea que se realiza mediante el cableado), o bien emplear la **tecnología programada**, en la cual mediante un programa se establece cómo debe responder el sistema ante estímulos en las entradas.

Los sistemas cableados son útiles en sistemas fijos donde no se necesita modificar la forma de funcionamiento.

Los sistemas programados se emplean en sistemas complejos donde un cambio en el programa o forma de actuar no implica un cambio en los elementos que lo integran.

Los autómatas programables también reciben el nombre de **PLC** (*Programmable Logic Controller*, controlador lógico programable).

SABÍAS QUE

En el argot técnico se denomina simplemente autómata a los **autómatas** programables o PLC.

10.2. Las ventajas y los inconvenientes

El emplear una tecnología u otra supone una serie de ventajas e inconvenientes. Entre las ventajas están:

- Posibilidad de introducir **cambios** sin realizar modificaciones del cableado.
- **Reducción de tiempo** en la puesta en marcha al reducir el cableado.
- **Reducción de costes de mano de obra** de la instalación.
- **Reducción de costes asociados al mantenimiento**. Fiabilidad de los autómatas junto con la detección de averías.
- Posibilidad de **recogida de datos de históricos**, debido a la memoria de almacenamiento.
- Posibilidad de **obtener datos de funcionamiento** en tiempo real.
- Posibilidad de **comunicarse** con otros autómatas.

Pero entre sus inconvenientes están:

- **Alta cualificación** del personal técnico, por la tarea de programación de los autómatas.
- **Alto coste material** de la instalación. Pero puede ser compensado con sus ventajas.

10.3. La estructura del autómata programable

Atendiendo a su estructura externa, los autómatas se clasifican en:

- **Compactos.** Todos los elementos necesarios están agrupados y se integran un único dispositivo. Son ideales para pequeñas aplicaciones con pocas entradas y salidas.

Figura 10.1. *Logo. (Cortesía de Siemens.)*

Figura 10.2. *Zelio. (Cortesía de Schneider.)*

Figura 10.3. *Easy. (Cortesía de Moeller.)*

Figura 10.4. *S7-1200. (Cortesía de Siemens.)*

- **Modulares.** Se componen de varios módulos, donde cada módulo cumple con una función específica, por ejemplo: entradas analógicas, entradas digitales, salidas, comunicaciones, etc. Permiten fácilmente la ampliación del sistema hasta cubrir las necesidades.
- **Semicompactos.** Casi todos los módulos están agrupados salvo algunos de ellos, como por ejemplo la fuente de alimentación, comunicaciones, etc.

A nivel interno, un autómata se compone de las siguientes partes:

- **CPU.** Es la parte más importante del autómata. Se encarga de leer las entradas y activar las salidas en función de un programa.
- **Memorias.** Las hay de dos tipos: las tipo ROM (ROM, EPROM y EEPROM) que son memorias de solo lectura, estas no pierden su información ante un corte de energía; y las memorias RAM, que son unas memorias de lectura/escritura y ante un corte de energía pierden su información.

  Las memorias se emplean de tres formas:

  — Memoria de usuario: es donde se almacena el programa del usuario. Es una memoria RAM que cuenta con el respaldo de una pequeña batería para evitar la pérdida de su información ante un corte de energía. Algunos autómatas cuentan con una memoria tipo EPROM y EEPROM en las que el fabricante ha almacenado previamente el programa.

  — Memoria de tabla de datos: es una memoria tipo RAM. Aquí se almacenan la imagen de las tablas de los estados de entrada y salida junto con las variables de los programas y datos internos (contadores, temporizadores, etc.).

  — Memoria y programa del sistema: se encuentra dividida en dos áreas: por un lado está el programa del sistema o *firmware* (que es grabado por el fabricante en la memoria tipo ROM) y la memoria que junto con el procesador componen la CPU.
- **Interface de entrada.** Adapta las señales de entrada para que las entienda la CPU.
- **Interface de salida.** Es el encargado de preparar las órdenes de la CPU en valores de salida.
- **Interface de periféricos.** Son el resto de elementos que pueden conectarse al autómata, como por ejemplo: una consola de programación, un panel de operación, otros PLC, impresoras, etc.

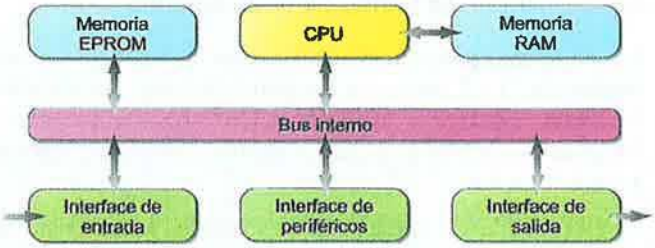


Figura 10.5. *Estructura interna de un autómata programable.*

10.3.1. La CPU

Es la parte más importante del PLC, cumpliendo varias funciones:

- Procesar las instrucciones del programa. Se encarga de ejecutar las instrucciones del programa de manera secuencial y de forma cíclica. La velocidad de procesamiento es muy alta.
- Leer el estado de las entradas.
- Activar las salidas.
- Comprobar todo el sistema. Se cuenta con una serie de alarmas que el autómata vigila.

Figura 10.6. *Varias CPU de la serie S7-300. (Cortesía de Siemens.)*

SABÍAS QUE

Hay una función en los autómatas que se denomina *watchdog* (perro guardián). Se encarga de comprobar que la ejecución del programa no excede de un tiempo.

Los diversos fabricantes agrupan sus modelos en función de la CPU.

10.3.2. La fuente de alimentación

Los autómatas programables funcionan internamente a una tensión de 24 y 5 voltios en corriente continua, por ello necesitan de una fuente de alimentación. Algunos autómatas integran internamente dicha fuente pero otros emplean un módulo externo.

Si la fuente de alimentación es interna, el autómata se conecta directamente a la red eléctrica entre fase y neutro. Por el contrario, si el autómata necesita de una fuente externa, su alimentación es a 24 V_{cc}.

La fuente de alimentación, además de suministrar energía al propio autómata se emplea para alimentar a los sensores o dispositivos que lo necesiten. Algunos autómatas con fuente interna cuentan con unos terminales para esta función.

Figura 10.7. *Fuente de alimentación para Logo. (Cortesía de Siemens).*

Figura 10.8. *Fuente de alimentación para S7-300. (Cortesía de Siemens.)*

Figura 10.9. *Fuente de alimentación genérica. (Cortesía de Siemens.)*

10.3.3. Los módulos de entradas y salidas

Los módulos de entradas y salidas se clasifican en función del tipo de datos que emplean, así se tienen:

- **Módulos digitales o binarios.** Utilizan datos a nivel de bit, es decir todo o nada. Detectan tensión en una entrada o no la detectan, activan un bit de salida o no lo activan.
- **Módulos analógicos.** Poseen cualquier valor dentro de un rango. Utilizan datos a nivel de *byte* (8 bits) o *word* (16 bits). Los módulos de entradas analógicas se emplean para leer magnitudes que no se pueden expresar en valores todo o nada como por ejemplo: temperatura, presión, distancia, etc.

Estos módulos o son de entradas o son de salidas, aunque también los hay mixtos, donde combinan tanto las entradas como las salidas, sin embargo no mezclan el tipo de datos (analógico o discreto).

Figura 10.10. *Entradas analógicas.*

Figura 10.11. *Entradas digitales.*

Figura 10.12. *Entradas y salidas analógicas.*

Figura 10.13. *Salidas digitales.*

A las entradas del autómata se conectan los **captadores**. Estos captadores pueden ser de dos tipos: que no necesiten conexión eléctrica (por ejemplo, un pulsador o un final de carrera), o que necesiten conexión eléctrica (por ejemplo, un sensor inductivo o una barrera fotoeléctrica). Es conveniente que estos dispositivos sean de la misma tensión que el autómata.

La conexión de un captador sin tensión, como un pulsador, se realiza conectando un extremo a la entrada y el otro extremo al positivo de la fuente de tensión. Además, es necesario conectar el negativo de la entrada con el negativo de la fuente de alimentación. Con el fin de reducir el espacio, estos negativos se agrupan en un terminal común. A veces, se suele dividir las entradas en grupos, en los cuales cada grupo tiene su propio común.

En la Figura 10.14, se tiene un autómata con dos grupos de entradas, los comunes son 1M y 2M respectivamente.

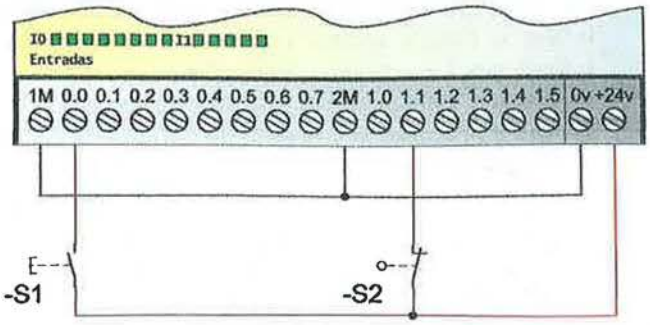


Figura 10.14. *Entradas pasivas.*

SABÍAS QUE

A los captadores que no necesitan ser conectados a una fuente de alimentación eléctrica se les denomina captadores pasivos.

Para los captadores que necesitan conexión eléctrica, esta se puede obtener de una fuente de alimentación externa o bien del propio autómata, el cual suele contar con unos terminales destinados a este fin.

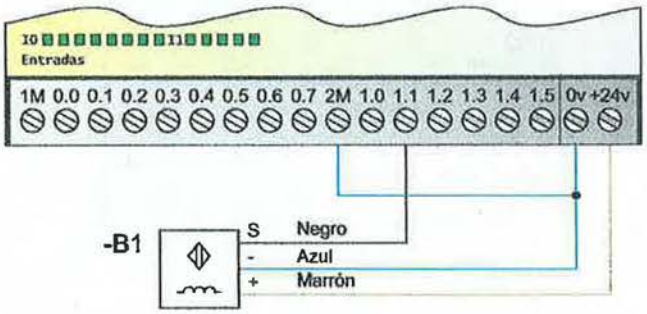


Figura 10.15. *Entradas activas.*

SABÍAS QUE

A los captadores que necesitan ser conectados a una fuente de alimentación eléctrica se les denomina captadores activos.

A las salidas del autómata se conectan los **actuadores**. Existen varios tipos de salidas digitales:

- **Salidas a relé.** Tiene dos terminales y actúan como un contacto. En sus bornes no existe tensión eléctrica, por ello se llaman **libres de potencial**. Pueden trabajar con cualquier tipo de tensión, alterna o continua y de cualquier valor.

Las conexiones de salida pueden tener dos bornes o bien se pueden agrupar por bloques (al igual que ocurría con las entradas). Cada bloque contará con un contacto común. El tener varios bloques facilita la conexión de actuadores de diferente tensión eléctrica.

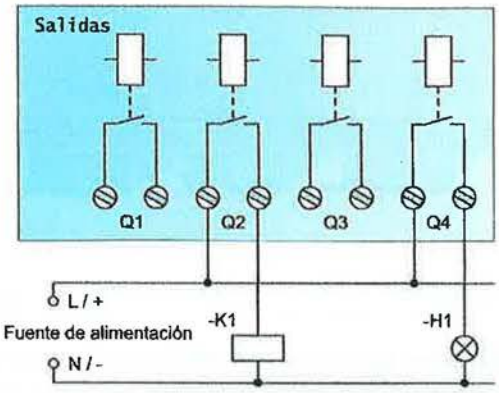


Figura 10.16. *Salidas a relé con dos bornes.*

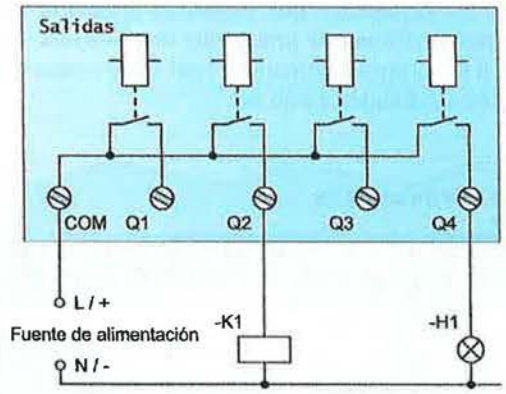


Figura 10.17. *Salidas a relé con borne común.*

- **Salidas a transistor.** La salida es con tensión, por ello los actuadores deben ser del mismo valor de tensión eléctrica. Son adecuados para actuadores de corriente continua. Pueden ser de dos tipos: PNP o NPN.

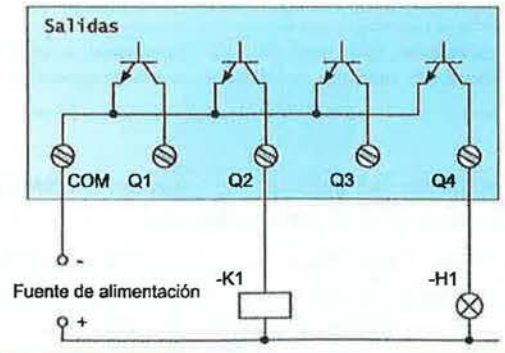


Figura 10.18. *Salidas a transistor.*

SABÍAS QUE

Aunque el módulo de salida a relé no tiene polaridad, se suele emplear de tal forma que la salida del autómata proporcione positivo o fase al actuador.

- **Salida a TRIAC.** Se emplean donde se requieran altas capacidades de conmutación.

Las salidas a relé pueden trabajar para cualquier tipo de voltaje y permiten trabajar con mayores corrientes que con las salidas a transistor. Sin embargo, la velocidad de conmutación del transistor es superior a las de relé. Por otro lado, la vida útil de los relés es inferior a la de los transistores. A la hora de seleccionar un tipo de módulo de salida dependerá de la aplicación a controlar.

A la hora de trabajar con los módulos de salidas digitales o binarias, se deben tener en cuenta una serie de consideraciones:

- Los actuadores conectados en un mismo grupo de salida deben ser de la misma tensión.
- Para cada grupo de contactos de salida, se debe calcular las intensidades demandadas por cada actuador y la suma de ambas no debe sobrepasar la intensidad indicada por el fabricante.

En la mayoría de los casos, como actuadores, se conectarán contactores. Los contactores son elementos inductivos y este tipo de carga genera picos de tensión en el proceso de desconexión. Los fabricantes de autómatas incorporan en los módulos un circuito de protección, pero a veces estos no son suficientes y se debe de complementar.

Cuando la carga es de corriente continua, hay tres circuitos de protección (Figura 10.19):

- **Protección mediante diodo.** Se emplea en cargas inductivas con bajo número de maniobras.
- **Protección mediante diodo y resistencia.** Es más completo que el anterior y se emplea para el mismo caso, cargas inductivas y con bajo número de maniobras.
- **Protección mediante diodo y VDR (varistor).** Se emplean en cargas inductivas con alto número de maniobras.

En el caso de cargas en corriente alterna, se tienen dos situaciones:

- Carga de alta inductancia.
- Carga de alta impedancia.

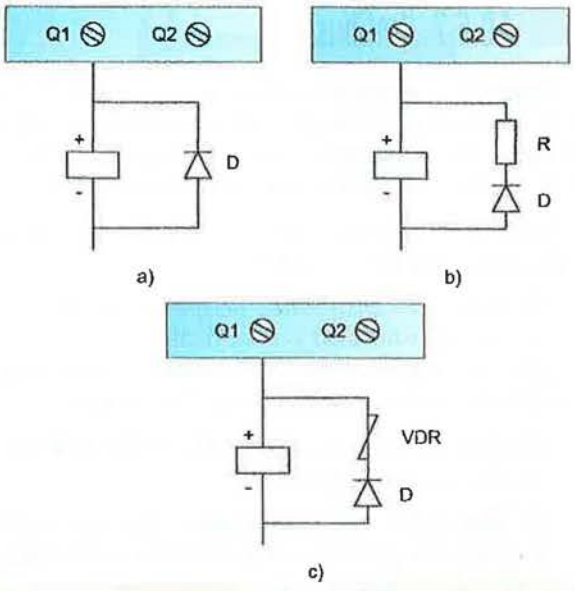


Figura 10.19. *Protección del autómata en corriente continua.*

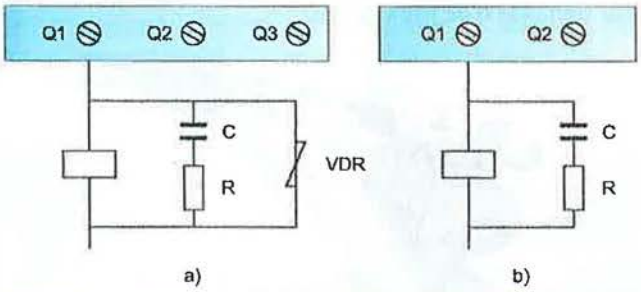


Figura 10.20. *Protección del autómata en corriente alterna.*

10.4. Los paneles de operación

Los paneles de operación son unos periféricos destinados a interactuar con el operario. A este tipo de dispositivos se les denomina HMI (*Human Machine Interface*).

Estos paneles o consolas se componen de dos partes: una parte es la encargada de visualizar información (tales como la situación de una máquina o proceso, alarmas, etc.) y otra parte es la encargada de recoger información que proporciona el operario (dar alguna orden concreta) mediante un conjunto de teclas.

Existen dos tipos de pantallas: las **pantallas alfanuméricas**, que proporcionan información en formato texto; y las **pantallas gráficas**, que proporcionan información en formato gráfico.

Estas pantallas se combinan con una serie de teclas, aunque actualmente, los paneles más modernos integran ambas funciones mediante las pantallas táctiles.

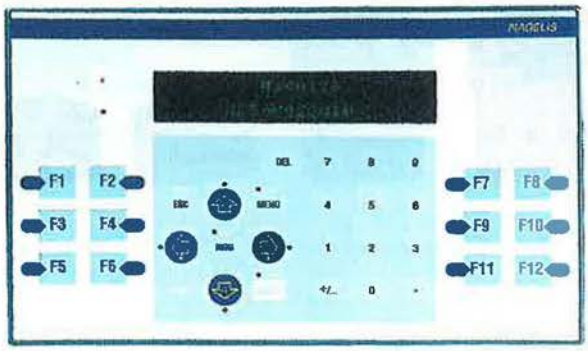


Figura 10.21. *Consola Magelis. (Cortesía de Schneider.)*

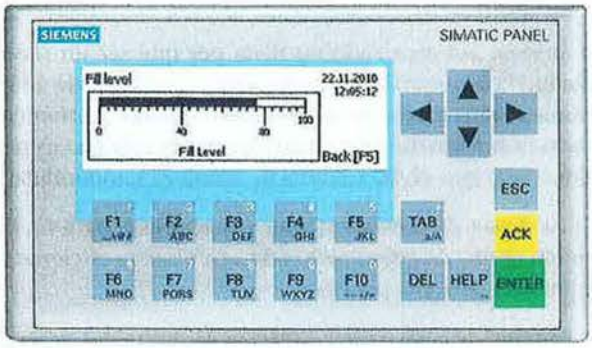


Figura 10.22. *Consola. (Cortesía de Siemens.)*

Estos paneles requieren de una programación que se realiza mediante un *software* específico.

10.5. Las tarjetas de memoria

Una vez realizado un programa para el autómata, se puede copiar en una tarjeta de memoria o *memory card* y luego insertarla en el autómata para su transferencia. También se utiliza para el paso inverso, copiar el programa del logo a la tarjeta de memoria.

Estas tarjetas de memoria se encuentran en dos formatos:

- Formato tarjeta.
- Formato cartucho.

Figura 10.23. *Memoria tipo tarjeta.*

Figura 10.24. *Memoria CF.*

Figura 10.25. *Memoria tipo cartucho.*

10.6. Las comunicaciones industriales

Un sistema automatizado no tiene por qué ser un sistema aislado. Ni un sistema automatizado se compone de un solo autómata. Hoy en día los autómatas requieren comunicarse con otros autómatas y dispositivos, es por ello que es necesario contar con algún sistema de comunicación industrial.

Los buses de comunicaciones se emplean para facilitar el intercambio de información de una manera eficiente con el mínimo cableado posible.

Aunque existen varias soluciones, entre las más empleadas actualmente están las siguientes:

- Modbus.
- Profibus.
- AS-i.
- Ethernet.

10.6.1. Modbus

Es un bus veterano que se emplea para enviar y recibir datos de control entre los sensores y los controladores a través del puerto RS-232 y con una comunicación punto a punto. No ha sido estandarizado. Tiene una filosofía de maestro-esclavo y existen dos modos: uno en modo ASCII en el cual se transmiten dos caracteres (2 bytes) por cada mensaje y el modo RTU en el que se transmiten cuatro caracteres por mensaje. Existe una versión que emplea comunicación mediante RS-485 con la que se aumentan sus prestaciones.

El medio físico está compuesto por un cableado de par de hilos trenzados con alimentación independiente para cada dispositivo.

Es un protocolo de comunicación con muchas limitaciones y hoy en día solo se emplea en instalaciones ya existentes. Existen intentos de relanzarlo empleando Modbus sobre TCP/IP.

10.6.2. Profibus

El sistema de comunicación Profibus (*Process Field Bus*) fue creado en un principio por empresas alemanas a las que posteriormente se añadieron otras europeas. Actualmente es líder en Europa. Es estándar mediante la norma EN-50170.

Es un bus de tipo maestro-esclavo de altas prestaciones el cual cuenta con tres versiones:

- **Profibus-DP** (*Distributed Peripherals*). Optimizado para la alta velocidad y coste reducido. Está indicado para la comunicación entre sistemas automáticos de control y entradas/salidas a nivel de campo.
- **Profibus-PA** (*Process Automation*). Diseñado para la automatización de procesos.
- **Profibus-FMS** (*Fieldbus Message Specification*). Es de propósito general, con preferencia de la funcionalidad a la velocidad.

Profibus emplea una topología de bus con terminación de impedancia en los extremos. El cableado puede ser de par trenzado o de fibra óptica.

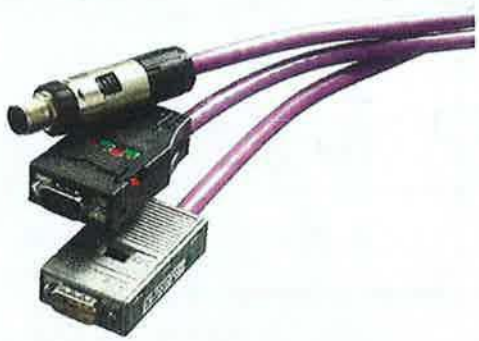


Figura 10.26. *Cables Profibus.*

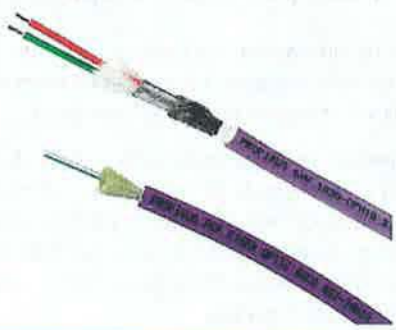


Figura 10.27. *Cables Profibus par trenzado y óptico.*

Figura 10.28. *Terminador.*

Figura 10.29. *Módulo Profibus para S7-200. (Cortesía de Siemens.)*

10.6.3. AS-i

El bus AS-i (*Actuador Sensor interface*) nació con la idea de eliminar el cableado entre los actuadores y sensores a la vez que proporciona alimentación eléctrica a los dispositivos, todo ello por un mismo sistema de cableado de dos hilos.

Figura 10.30. *Cable AS-i.*

Así, en vez de llevar a cada dispositivo los cables de control (entradas o salidas) más sus cables de alimentación, solo se lleva un único cable de dos hilos al cual se conectan todos los dispositivos: sensores, contactores, pilotos de señalización, etc.

Figura 10.31. *Módulo AS-i para S7-1200. (Cortesía de Siemens).*

Figura 10.32. *Final de carrera con AS-i.*

Figura 10.33. *Arrancador con AS-i. (Cortesía de Siemens.)*

El bus AS-i se emplea en el escalafón más bajo de la pirámide de la automatización, enlazando captadores y actuadores. Es un sistema de comunicación abierto y reconocido por el estándar EN-50295 y IEC 62026-2.

Sus características principales son:

- Es un sistema maestro-esclavo, en el cual se realiza un sondeo con un tiempo máximo de 5 ms (es decir, en ese tiempo se deben reconocer todos los elementos del bus).
- Un maestro controla hasta 32 esclavos (62 en la versión 2.1).
- Es posible la comunicación con módulos analógicos.
- La longitud máxima del bus es de 100 m sin repetidores.

Un sistema de bus AS-i lo forman los siguientes elementos:

- **Un maestro del bus AS-i.** Suele estar conectado a un autómata programable.
- **Una fuente de alimentación.** Proporciona una tensión de 30 V_{CC} y hasta 8 A para alimentar a los elementos esclavos del bus (Figura 10.34). Es posible conectar otras fuentes cuando se necesitan tensiones diferentes a través de cables adicionales: 24 V_{CC} (cable negro) y 230 V_{CA} (cable rojo).
- **Los esclavos del bus AS-i.** Existen dos tipos principales: las que cuentan con la electrónica integrada dentro del dispositivo y los módulos AS-i genéricos (Figura 10.35), los cuales disponen de varias entradas/salidas y es posible conectar cualquier sensor/actuador. La primera opción se emplea en instalaciones nuevas y la segunda opción se emplea en instalaciones existentes y de esta manera se aprovecha la infraestructura reduciendo los costes. Para indicar cuáles son los esclavos dentro del bus, se emplea la programadora de direcciones (Figura 10.36).

- **El cable del bus AS-i.** Es un cable plano de color amarillo de dos hilos el cual cuenta con una muesca para su colocación en una única posición.

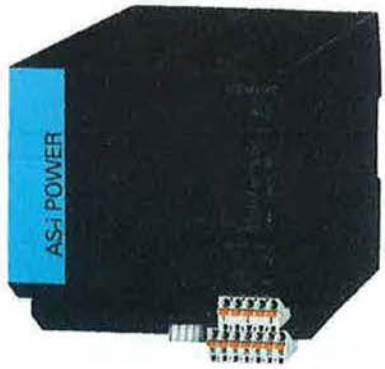


Figura 10.34. *Fuente de alimentación AS-i.*

Figura 10.35. *Módulo esclavo AS-i.*

Figura 10.36. *Programadora de direcciones.*

10.6.4. Ethernet industrial

El sistema de Ethernet industrial se emplea especialmente en aquellos casos con las siguientes características:

- Intercambio de grandes cantidades de datos.
- Gran distancia entre dispositivos.

Para la comunicación se empleaba originalmente cable coaxial, sin embargo el sistema ha evolucionado hacia otros soportes físicos tales como el par de hilos trenzado o la fibra óptica, consiguiendo mayores velocidades de transmisión y mediante otras mejoras se ha aumentado el ancho de banda. Los conectores de los cables son de tipo RJ-45. Actualmente, también es posible la comunicación inalámbrica.

Las velocidades se fijan en 10, 100 y 1.000 Mbit/s. Y las ventajas que ofrece son:

- Altas prestaciones aun existiendo muchos participantes y a grandes distancias.
- Transferencia de datos segura. Para entornos industriales con muchas perturbaciones electromagnéticas.
- Ahorro en costes con la simplificación del cableado.
- Es la red industrial que más se está implantando.

Las topologías de montajes pueden ser: lineal, en árbol, en estrella, en anillo. La topología en anillo es la más confiable debido a la redundancia en la conexión.

Aunque hay varios protocolos de comunicación sobre Ethernet, el empleado con preferencia es el TCP/IP.

Figura 10.37. *Logo con Ethernet. (Cortesía de Siemens.)*

Figura 10.38. *Módulo de comunicación para S7-200. (Cortesía de Siemens.)*

El elemento principal en una red Ethernet es el *switch* (conmutador), a él se conectan los participantes de la red.

Figura 10.39. Switch. *(Cortesía de Siemens.)*

Figura 10.40. *Módulo para ET200. (Cortesía de Siemens.)*

Figura 10.41. *Módulo de Ethernet para Twido. (Cortesía de Schneider).*

10.7. Los sistemas SCADA

Un sistema SCADA (*Supervisory Control And Data Adquisition*) es un sistema de *software* que, desde la pantalla de un ordenador, se comunica con los dispositivos de campo supervisando todo el proceso industrial.

Esta información que proporciona es útil para el trabajo diario de cada usuario o departamento: control de producción, mantenimiento, operadores, control de calidad, etc.

Las tareas de automatización las realizan los autómatas del sistema y estos proporcionan la información al ordenador, pero también desde el propio ordenador se puede dar órdenes a los autómatas.

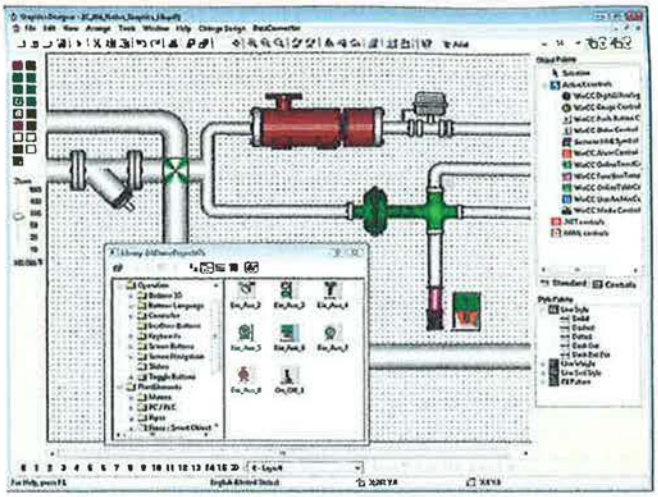


Figura 10.42. *Diseño de la pantalla de un sistema SCADA.*

Las funciones principales que realiza son:

- De **adquisición de datos**. Recoge, procesa y almacena la información recibida.
- De **supervisión**. Observación en la pantalla de la evolución de las variables que intervienen en el proceso.
- De **control**. Modificación de la evolución del proceso mediante órdenes a los dispositivos que intervienen en él.

Las ventajas principales del empleo de un sistema SCADA son:

- **Gestión de alarmas.** Información sobre las alarmas producidas con registro de las incidencias.
- **Generación de históricos.** Se recoge información sobre la evolución del proceso controlado a lo largo del tiempo, lo que permite entre otras cosas, detectar desviaciones.
- **Descarga de trabajo a los autómatas.** Ciertas tareas requieren una alta capacidad de cálculo. Se puede tomar esos datos en el propio ordenador, realizar el cálculo y con los resultados obtenidos pasárselos al autómata.

10.8. Los circuitos eléctricos en los autómatas

Los circuitos eléctricos en los cuales se emplean autómatas reducen su cableado. Así, por ejemplo, en la Figura 10.43 se muestra un arranque de un motor mediante lógica cableada y lógica programada. En el primer caso, se le indica la forma de conectar los diferentes elementos y cómo debe

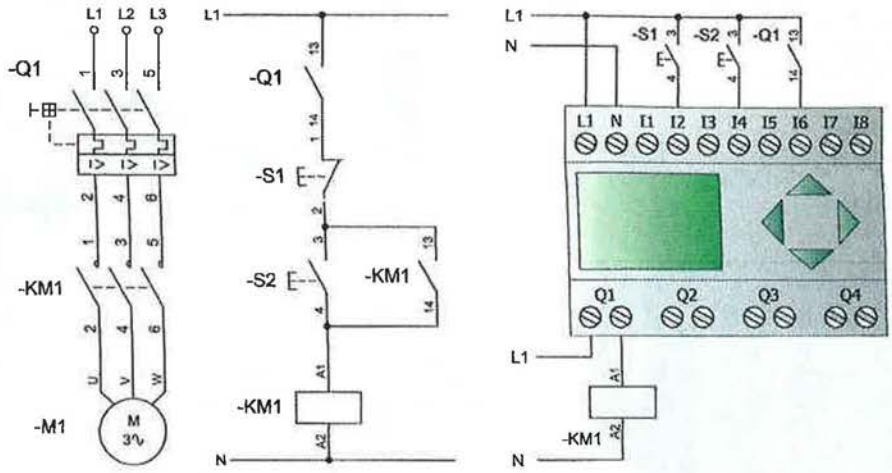


Figura 10.43. *Comparación de circuito en lógica cableada y lógica programada.*

responder ante la intervención de los pulsadores. En cambio, en la lógica programada, al autómata se le conectan los captadores a la entrada y los actuadores se conectan a la salida. La forma de cómo debe responder ante la intervención en las entradas se realiza mediante el programa.

Aunque cada marca comercial tiene sus modelos de autómatas, es fácil identificar cada parte y en caso de duda se debe consultar su manual. En las figuras siguientes se muestran tres modelos muy conocidos de una gama baja (Logo y S7-200 de Siemens y Zelio de Schneider), identificando cada parte.

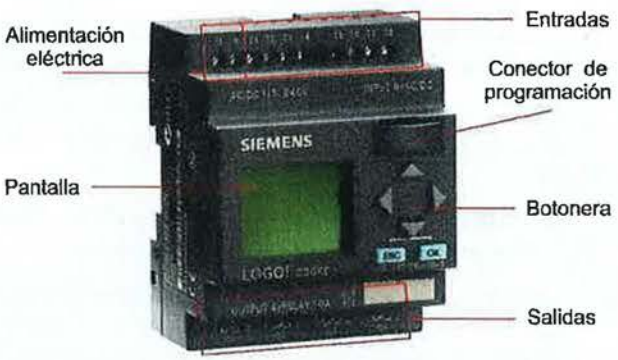


Figura 10.44. *Logo. (Cortesía de Siemens.)*

Figura 10.45. *Zelio. (Cortesía de Schneider.)*

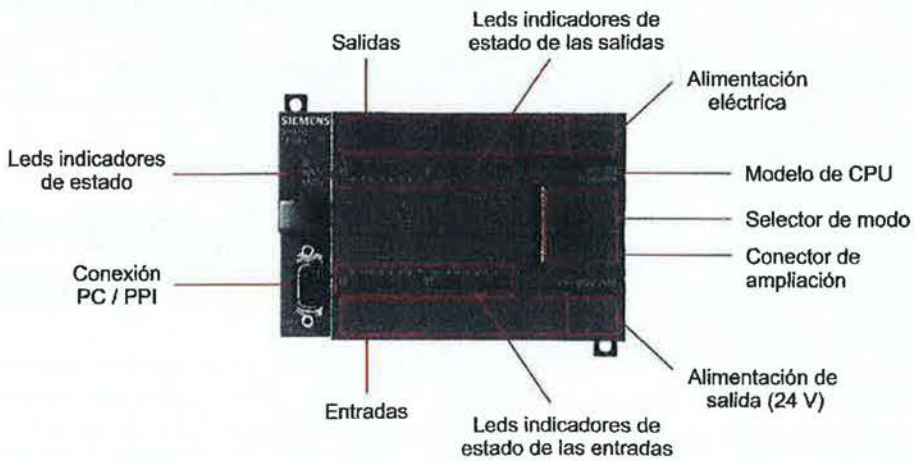


Figura 10.46. *S7-200. (Cortesía de Siemens.)*

10.8.1. Los contactos de seguridad

En un autómata, el tipo de contacto (normalmente abierto o normalmente cerrado) de un pulsador o bien de un captador no importa, puesto que por medio del programa se trata según su condición. Sin embargo, en el caso de elementos de seguridad (por ejemplo, relés térmicos, paros de emergencia, etc.) conviene que sean del tipo normalmente cerrados (NC) por una cuestión de seguridad. El autómata estaría constantemente detectando la señal mandada por el sensor o elemento de seguridad, si ocurre un fallo se dejaría de recibir esa señal y el autómata lo detectaría inmediatamente.

Los relés térmicos, así como otros elementos de seguridad, se pueden tratar de dos maneras:

- Conectado como entrada (Figura 10.47). El contacto del relé se lleva a una entrada del autómata.
- Conectado como salida (Figura 10.48). El contacto normalmente cerrado del relé se conecta en serie con el contactor del motor a proteger.

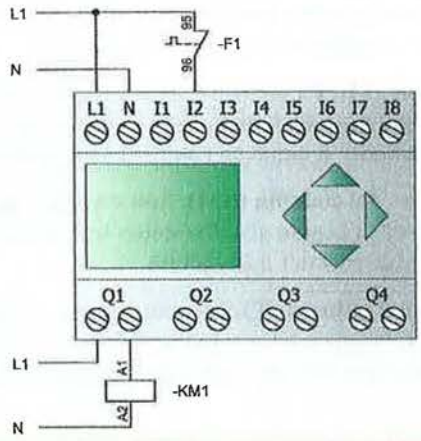


Figura 10.47. *Contacto del relé térmico conectado como entrada.*

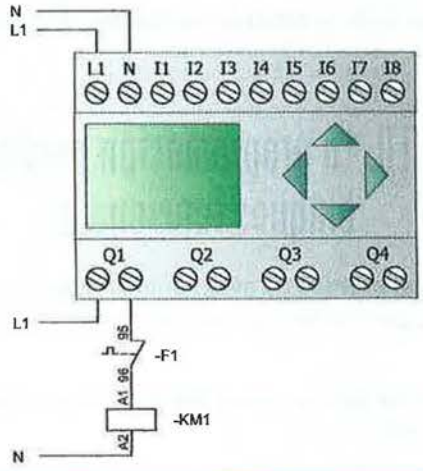


Figura 10.48. *Contacto del relé térmico conectado como salida.*

La conexión del contacto del relé térmico como entrada es la más apropiada desde el punto de vista del control, ya que un disparo del relé térmico sería detectado por el autómata y se podría generar una señal de alarma. Si se conecta como salida, el relé térmico seguiría protegiendo el circuito pero el autómata no detectaría su disparo y por tanto no podría generar ninguna señal de alarma o paro del proceso por motivos de seguridad.

Existen máquinas en las cuales se emplean varios motores y por tanto varios relés térmicos u otros sistemas de seguridad. Lo ideal es conectar cada térmico a una entrada, pero en muchas ocasiones no se dispone de entradas libres. Se puede aumentar el número de entradas pero el coste de la instalación aumentaría. Una solución muy empleada es enlazar en serie los relés térmicos y conectarla a una entrada. El autómata estaría en condiciones de detectar el fallo por sobrecarga pero no indicaría cuál térmico se disparó.

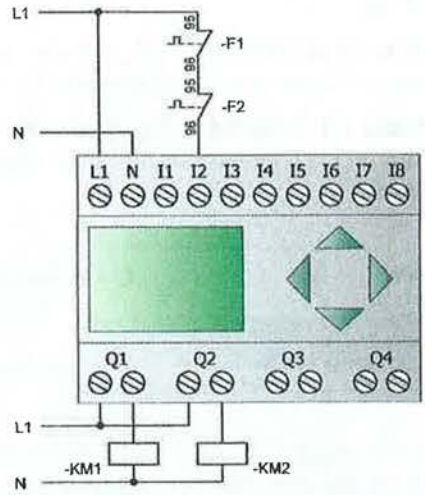


Figura 10.49. *Contactos de varios relés térmicos conectados en serie.*

10.9. La programación de autómatas

La manera de cómo el autómata procesa los datos de los captadores conectados a las entradas para activar o desactivar las salidas del autómata se realiza mediante una serie de órdenes. Estas órdenes se indican mediante algún lenguaje de programación.

10.9.1. Los lenguajes de programación

Los lenguajes de programación se han ido estandarizando desde que se publicó la norma IEC 1131-3. Con esta norma, se ha conseguido simplificar y facilitar el aprendizaje de los lenguajes de programación, ya que al unificar los criterios se facilita el poder programar de la misma manera cualquier autómata acogido a esta norma.

Existen cuatro lenguajes de programación normalizados. Dos de ellos son de tipo gráfico y los otros dos son de tipo texto.

- Tipo gráfico:
  - — **Diagrama de bloques funcionales** (FBD, *Function Block Diagram*). Se basa en el empleo de las funciones lógicas.
  - — **Diagrama de contactos** (LD, *Ladder Diagram*). Se basa en el empleo de la lógica de relés.

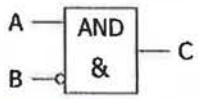


Figura 10.50. *Ejemplo de diagrama de bloques funcionales.*

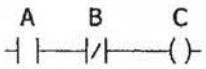


Figura 10.51. *Ejemplo de diagrama de contactos.*

- Tipo texto:
  - — **Lista de instrucciones** (IL, *Instruction List*). Se basa en los principios del lenguaje ensamblador.
  - — **Texto estructurado** (ST, *Structured Text*). Se basa en los lenguajes de alto nivel (lenguaje C y otros).

```
LD      A
ANDN    B
ST      C
```

Figura 10.52. *Ejemplo de lista de instrucciones.*

```
C = A AND NOT B
```

Figura 10.53. *Ejemplo de texto estructurado.*

La elección de un lenguaje u otro va a depender, principalmente, de las preferencias y conocimientos del programador, de la dificultad del problema a resolver y del autómata a emplear.

10.9.2. Las áreas o mapa de memoria

El autómata, a la hora de trabajar, necesita almacenar temporalmente los datos. Por ejemplo, a la hora de utilizar un contador, debe guardar en una parte de la memoria la cuenta que lleva, además del valor de la cuenta a la cual se activará. Estos datos se guardan en el área de memoria reservada a los contadores.

Las áreas de memoria se identifican mediante una letra que hace referencia a la función que realizan.

Para referirse a un elemento concreto del área de memoria, se indica el identificador de área más un número que hace referencia a la posición de ese elemento dentro de esa área. Por ejemplo, el elemento T3 indica que es un temporizador (T) y el número 3 hace referencia a que es el tercer temporizador.

Tabla 10.1. Mapa de memoria.

Área de memoria	Letra
Entradas	I
Salidas	Q
Marcas	M
Marcas de sistema	SM
Temporizadores	T
Contadores	C

Cada área de memoria indica:

- **Entradas (I).** El autómata lee las entradas y vuelca sus estados en el área de memoria reservada para las entradas. En estas posiciones de memoria se coloca el valor leído por los captadores conectados a las entradas físicas del autómata.
- **Salidas (Q).** El autómata vuelca el valor de esa área de memoria sobre las salidas, activando o desactivando los actuadores conectados en ella.
- **Marcas (M).** Es un espacio de almacenamiento temporal para las variables. Está a libre disposición del programador aunque su cantidad está limitada.
- **Marcas del sistema (SM).** Son espacios de memoria donde es el propio sistema quien lo emplea, pero que es posible acceder a su lectura.
- **Temporizadores (T).** En esta área se almacenan los datos referentes a los temporizadores: tiempo de umbral, tiempo parcial, cuenta ascendente o descendente, etc.
- **Contadores (C).** En esta área se almacenan los datos referentes a los contadores.

Cada modelo de autómata está dotado con una cantidad limitada de esa área de memoria.

10.10. La programación mediante bloques funcionales

Es una programación de modo gráfico, donde cada bloque representa una función. Existen dos grupos de bloques funcionales:

- **Bloques de funciones básicas.** Siendo las más importantes:
  - — Entrada. Se emplea para indicar la lectura de una entrada (captador).

— Salida. Se emplea para indicar una salida (actuador).

— Función OR. Realiza la suma lógica de las señales.

— Función NOR. Realiza la suma lógica de las señales y luego la invierte.

— Función AND. Realiza el producto lógico de las señales.

— Función NAND. Realiza el producto lógico de las señales y luego lo invierte.

— Función NOT. Invierte el valor de la señal.

- **Bloques de funciones especiales.** Siendo las más importantes:

— Temporizadores. A la conexión, desconexión, semanales, anuales, etc.

— Contadores.

— Otros.

10.10.1. Las entradas

Se identifica con la letra I seguida del número de entrada al autómata. En ella se almacena el valor de dicha entrada.

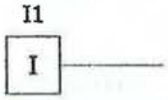


Figura 10.54. *Entradas.*

Figura 10.55. *Contacto abierto.*

Su equivalente eléctrico sería un contacto abierto o cerrado.

10.10.2. Las salidas

Se identifica con la letra Q seguida del número de salida del autómata.

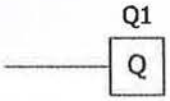


Figura 10.56. *Salidas.*

10.10.3. La función OR

La función OR realiza la suma lógica de dos o más señales y el resultado lo coloca en la salida del bloque.

Tabla 10.2. Función OR.

Entradas		Salida
I1	I2	Q
0	0	0
0	1	1
1	0	1
1	1	1

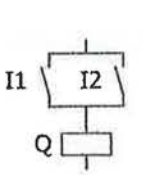


Figura 10.57. *Función OR (circuito).*

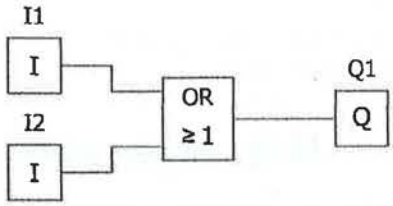


Figura 10.58. *Función OR (bloques).*

La ecuación lógica es: $Q = I1 + I2$

La función OR representa la conexión de dos o más contactos en paralelo. La salida vale 1 cuando alguna de sus entradas tenga un estado lógico de 1.

RECUERDA

La función OR, al representar un circuito paralelo, con estar cerrado (1) alguno de sus contactos ya es suficiente para que se cierre el circuito, obteniendo un estado lógico de 1.

10.10.4. La función NOR

La función NOR es la función OR a la cual se invierte su resultado.

Tabla 10.3. Función NOR.

Entradas		Salida
I1	I2	Q
0	0	1
0	1	0
1	0	0
1	1	0

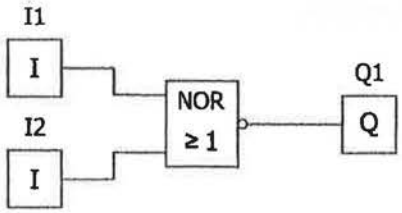


Figura 10.59. *Función NOR (bloques).*

La ecuación lógica es:

$$Q = \overline{I1 + I2}$$

La salida vale 1 cuando todas sus entradas tengan un estado lógico de 0.

10.10.5. La función AND

La función AND realiza el producto lógico de dos o más señales y el resultado lo coloca en la salida del bloque.

Tabla 10.4. Función AND.

Entradas		Salida
I1	I2	Q
0	0	0
0	1	0
1	0	0
1	1	1

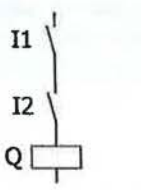


Figura 10.60. *Función AND (circuito).*

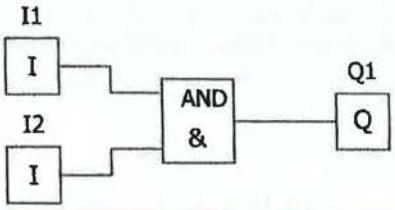


Figura 10.61. *Función AND (bloques).*

La ecuación lógica es: $Q = I1 \times I2$

La función AND representa la conexión de dos o más contactos en serie. La salida vale 1 cuando todas sus entradas tengan un estado lógico de 1.

RECUERDA

La función AND, al representar un circuito serie, todos sus contactos deben estar cerrados (1) para que circule la corriente y se tenga un 1 a su salida. Si se abre cualquier contacto ya se impide el paso de la corriente.

10.10.6. La función NAND

La función NAND es la función AND a la cual se invierte su resultado.

Tabla 10.5. Función NAND.

Entradas		Salida
I1	I2	Q
0	0	1
0	1	1
1	0	1
1	1	0

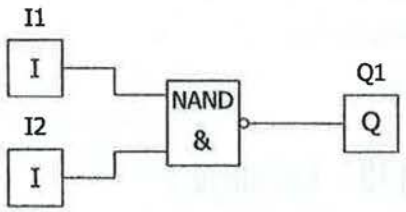


Figura 10.62. *Función NAND (bloques).*

La ecuación lógica es: $Q = \overline{I1 \times I2}$

La salida vale 0 cuando todas sus entradas tengan un estado lógico de 1.

10.10.7. La función NOT

La función NOT se encarga de realizar la inversión del valor de su entrada.

Tabla 10.6. Función NOT.

Entradas	Salida
I1	Q
0	1
1	0

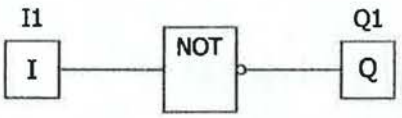


Figura 10.63. *Función NOT (bloques).*

La ecuación lógica es:

$$Q = \overline{I1}$$

La salida vale 1 cuando su entrada tenga un estado lógico de 0.

Se puede aplicar la función NOT tanto a resultados de bloques como a las entradas. En este caso un contacto abierto negado es lo mismo que un contacto cerrado.

10.10.8. La resolución de problemas

La resolución de problemas mediante diagrama de bloques de funciones se realiza obteniendo la función lógica y posteriormente aplicando el diagrama de bloques.

Actividad resuelta 10.1

Realiza el programa mediante diagrama de bloques funcionales del circuito dado:

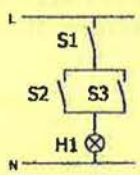


Figura 10.64. *Circuito.*

Solución:

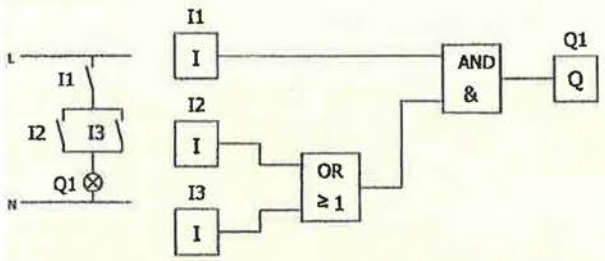


Figura 10.65. *Solución.*

Se asignan los contactos a las entradas y salidas.

El circuito se resuelve obteniendo la ecuación lógica. Se observa que está compuesto por dos contactos en paralelo (I2 y I3) que se representa por la suma; y su resultado está en serie con el contacto I1 que se representa por el producto, por tanto se obtiene:

$$Q1 = I1 \times (I2 + I3)$$

En la ecuación se tienen dos funciones: una OR (paralelo) y una AND (serie). Por ello, se resuelve en paralelo (función OR) y su salida se multiplica (función AND) por la entrada I1, obteniendo el resultado final.

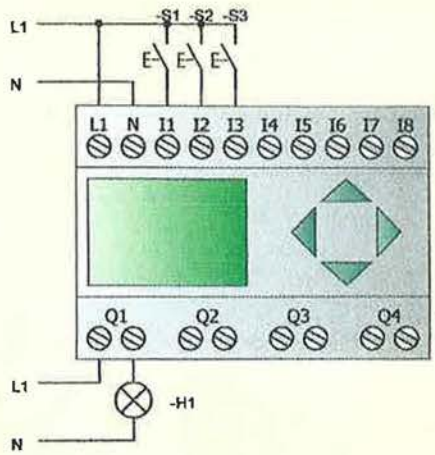


Figura 10.66. *Montaje.*

Actividad resuelta 10.2

Realiza el programa mediante diagrama de bloques funcionales del circuito dado:

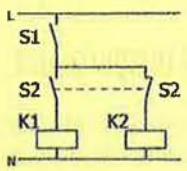


Figura 10.67. *Circuito.*

Solución:

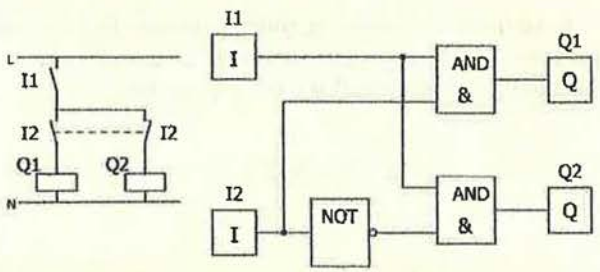


Figura 10.68. *Solución.*

Se asignan los contactos a las entradas y salidas. Como solo hay dos contactos (S1 y S2) se asignan las dos primeras entradas del autómata. El contacto cerrado de S2 se obtiene negando S2.

Se obtienen las ecuaciones lógicas para cada actuador, en este caso dos ecuaciones (para K1 y K2).

$$Q1 = I1 \times I2$$

$$Q2 = I1 \times \overline{I2}$$

Se obtiene una función AND por cada salida. Como se necesita negar I2, se necesita además, una función NOT.

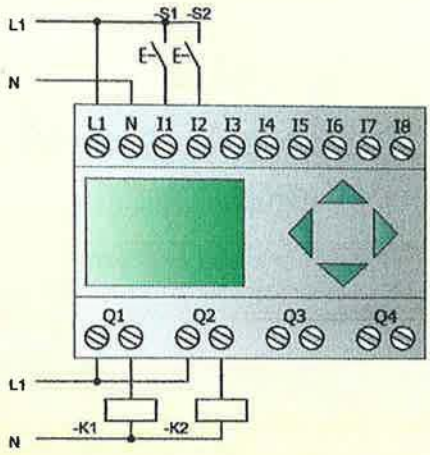


Figura 10.69. *Montaje.*

10.10.9. El temporizador

Para la gestión del tiempo se emplean los temporizadores. Los temporizadores pertenecen al grupo de las funciones especiales. Estos bloques de función se representan con la letra T seguida del número de orden de temporizador de entre todos los que dispone el autómata. Esta cantidad está limitada.

Existen varios modos de funcionamiento de los temporizadores, siendo los más empleados el temporizador a la conexión y el temporizador a la desconexión.

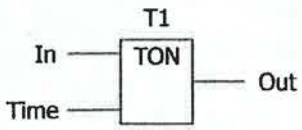


Figura 10.70. *Temporizador a la conexión.*

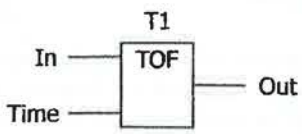


Figura 10.71. *Temporizador a la desconexión.*

En un temporizador a la conexión (TON), el proceso de la cuenta del tiempo empieza nada más se detecta la señal en la entrada.

En un temporizador a la desconexión (TOF), el proceso de la cuenta del tiempo empieza cuando el temporizador deja de detectar la señal a la entrada.

En ambos casos se debe fijar el tiempo de operación.

También se dispone de los programadores semanales y anuales, los cuales activan una salida del autómata en función del día y de la hora programada.

10.10.10. El contador

El contador es un bloque de función el cual cada vez que recibe un impulso en su entrada, lo registra llevando una cuenta. Cuando esta cuenta llega a un valor prefijado en la programación, activa su salida.

Existen los contadores con cuenta en modo ascendente (CTU), en modo descendente (CTD) y en ambos (ascendente y descendente) (CTUD).

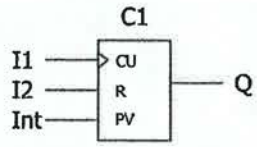


Figura 10.72. *Contador ascendente.*

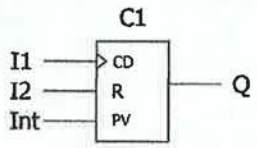


Figura 10.73. *Contador descendente.*

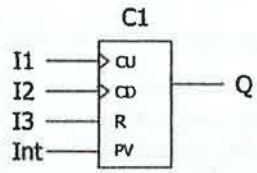


Figura 10.74. *Contador ascendente/descendente.*

Una de las entradas se encarga de recibir los pulsos a contar. Desde otra de las entradas se pone el contador a cero. El punto de umbral se debe fijar, para que al llegar la cuenta a ese valor se active la salida.

10.10.11. Otras funciones

Los autómatas, en función del modelo empleado, incorporan otras funciones especiales. Las más usuales son los comparadores y los biestables o básculas:

- **Comparadores.** Comparan el valor de dos entradas y en función de este, activan o no la salida. Se emplean con señales analógicas, por ejemplo para controlar una salida en función de un valor de temperatura leída en una entrada.
- **Básculas o biestables.** Memorizan un valor de una entrada hasta que se le da la orden de borrado. También se les llama básculas SR (Set-Reset).

10.11. La programación mediante diagrama de contactos

La programación mediante diagramas de contactos es un lenguaje parecido a un esquema eléctrico de lógica cableada.

Está compuesta por contactos que actúan sobre una salida o bobina. Existen dos tipos de entradas (normalmente abierta y normalmente cerrada, que representan a los pulsadores, finales de carrera, sensores, etc.) y varios tipos de salida.

En los esquemas eléctricos, estos se leen de arriba a abajo en cambio en los diagramas de contactos, estos se leen de izquierda a derecha

La simbología respecto a la empleada en los circuitos eléctricos cambia, teniendo la siguiente representación:

Tabla 10.7. Símbolos básicos en lenguaje de contactos.

Contacto normalmente abierto	⊣ ⊢	Contacto normalmente cerrado	⊣/⊢
Salida (bobina)	( )	Salida invertida (bobina)	(/)

Existen diversas variaciones sobre las salidas:

Tabla 10.8. Variaciones sobre las salidas en lenguaje de contactos.

(S)	Al cerrarse el circuito, la bobina se activa y permanece activa aunque el circuito se abra. Recibe el nombre de bobina de activación (Set).
(R)	Al cerrase el circuito, la bobina se desactiva, permaneciendo en este estado aunque se varíe su entrada. Recibe el nombre de bobina de desactivación (Reset).
(M)	En función de su entrada, se activa o desactiva una marca.
(SM)	Al cerrarse el circuito, la marca se activa y permanece activa aunque el circuito se abra. Es similar a la bobina de activación pero memoriza su estado.
(RM)	Al cerrase el circuito, la marca se desactiva, permaneciendo en este estado aunque se varíe su entrada. Es similar a la bobina de desactivación (Reset).

Las marcas actúan como relés auxiliares internos. Al activar una marca, se está poniendo a 1 un bit interno, y cuando se desactiva se pone a 0 ese mismo bit.

Respecto a las entradas, pueden responder ante variaciones de dos maneras: por niveles o por flancos.

Tabla 10.9. Variaciones sobre las entradas en lenguaje de contactos.

⊣ ⊢	Activación por niveles
⊣ P ⊢	Activación por flanco ascendente
⊣ N ⊢	Activación por flanco descendente

Las funciones especiales, tales como por ejemplo temporizadores, contadores, etc., se emplean en dos líneas de instrucciones. En una de ellas, el temporizador se representa en la línea de las bobinas, en esta se dan las órdenes (activación para el temporizador, pulsos para el contador, etc.). En la segunda línea es donde se emplean estos elementos como contactos.

Veamos el ejemplo de la Figura 10.75:

- En la línea 1, cuando se active la entrada I1 se activará la salida Q1. Si se desactiva la entrada, se desactiva la salida.
- En la línea 2, cuando se active la entrada I2 y no esté activa la I3 se activará la salida Q2 y permanecerá en este estado aunque se varíen sus entradas (I2 o I3).
- En la línea 3, al activar la entrada I3 se desactiva la salida Q2, que se había activado en la línea anterior.
- En las líneas 4 y 5 se emplea un temporizador (T1) el cual se configura una vez colocado (modo de trabajo, tiempo, etc.). Cuando se active la salida Q1 se activará el temporizador (T1) y este actuará sobre la salida Q3.

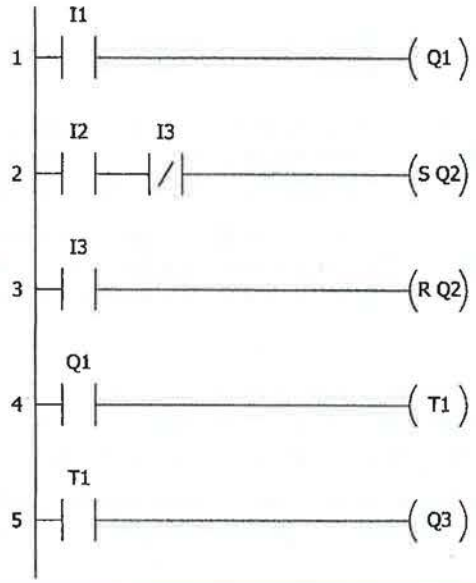


Figura 10.75. *Ejemplo de programación mediante lenguaje de contactos.*

10.12. La programación mediante lista de instrucciones

La programación mediante lista de instrucciones (IL, AWL) consiste en un listado de órdenes que el autómata ejecuta de manera secuencial. Estas órdenes se corresponden con el lenguaje máquina del autómata y es un lenguaje de bajo nivel.

En las siguientes tablas se muestra un conjunto de instrucciones.

Tabla 10.10. Instrucciones básicas.

Variables definidas	
Entradas	I, IX, IB, IW
Salidas	Q, QX, QB, QW, QD
Marcas	M, MX, MB, MW, MD
Operaciones lógicas	
Carga inicial	LD
Y	AND
NO-Y	ANDN
O	OR
NO-O	ORN
O exclusiva	XOR
NO-O exclusiva	XORN
Terminar una cadena	
Asignar	ST
Activar	S
Desactivar	R

Tabla 10.11. Instrucciones de bloques funcionales.

Bloques funcionales		
Bloque	**Operadores**	**Descripción**
S R	S1 R	Báscula SR (prioridad S)
R S	S R1	Báscula SR (prioridad R)
CTU	CU, R, PV	Contador ascendente
CTD	CD, R, PV	Contador descendente
CTUD	CU, CD, R, PV	Contador reversible
TP	IN, PT	Temporizador de impulso
TON	IN, PT	Temporizador a la conexión
TOFF	IN, PT	Temporizador a la desconexión

10.13. Los diagramas de Grafcet

El Grafcet (gráfico de función de etapas de transición) es un diagrama en el cual se representa el funcionamiento de un sistema automático y secuencial. El Grafcet también se denomina SFC (*Sequential Function Chart*, diagrama de función secuencial) y es una ayuda a la hora de diseñar y mantener un programa, puesto que se descompone este en partes más simples y por tanto más fáciles de abordar.

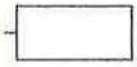
El resultado final del Grafcet es la obtención de un programa en lenguaje de contactos.

10.13.1. Los elementos del diagrama

Los elementos gráficos que intervienen en un diagrama Grafcet son:

Tabla 10.12. Símbolos Grafcet.

Símbolo	Nombre	Descripción
	Etapa inicial	Indica el estado inicial o punto de partida.

Símbolo	Nombre	Descripción
	Etapa	Son las etapas o estados intermedios.
	Transición	Indica la condición para pasar de una etapa a otra.
	Direccionamiento	Indica la dirección de la evolución de las etapas.
	Proceso simultáneo	Indica un proceso simultáneo de varias etapas.
	Etiquetas	Las etiquetas muestran las acciones que se llevan a cabo.

10.13.2. Las etapas

Al ser un programa cíclico, este se desarrolla por etapas o estados. El punto de partida se denomina **etapa inicial**. Esta etapa inicial se representa simbólicamente de manera diferente a las siguientes etapas.

Una etapa se compone de su símbolo con un número para poder identificarlo. Toda etapa está comprendida por una **transición** de entrada y una transición de salida. Al cumplir con las condiciones marcadas, se provoca el paso por esa transición, es decir la salida de una etapa y la entrada en la siguiente.

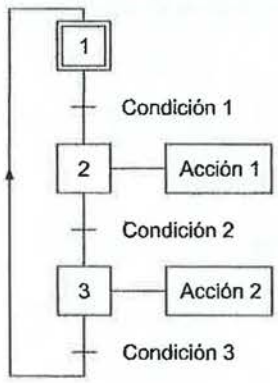


Figura 10.76. *Ejemplo genérico de Grafcet.*

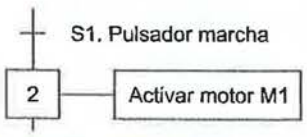


Figura 10.77. *Ejemplo de transición y acción.*

A la condición asociada a una transición se le llama **receptividad**. En el ejemplo de la Figura 10.77, la receptividad de la transición es "S1. Pulsador marcha".

Las **líneas de evolución** se entiende que van de arriba hacia abajo, en caso contrario se añade una flecha para indicar la dirección.

Las etapas llevan asociadas unas **etiquetas** para indicar las acciones que se llevan a cabo.

Las transiciones pueden ser de varios tipos:

- Por nivel. Por ejemplo, al activarse un final de carrera o un pulsador.
- Por ecuación booleana. Se cumple una condición marcada por una ecuación.
- Temporizada. Es un temporizador el que provoca la transición.
- Por variable interna. Se emplean variables internas del programa.
- Automática. Se realiza la transición de manera automática al cumplir una condición.

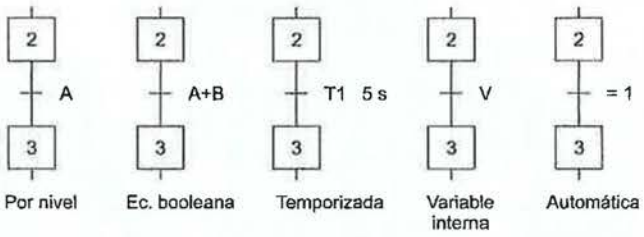


Figura 10.78. *Tipos de transiciones.*

10.13.3. Las reglas

Los diagramas de Grafcet se rigen por unas reglas:

- **Reglas de síntesis:**
  - — La alternativa etapa-transición debe ser respetada.
  - — Dos etapas deben estar separadas por una transición.
  - — Dos transiciones deben estar siempre separadas por una etapa.

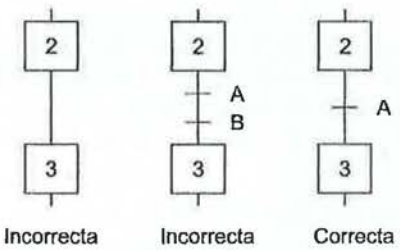


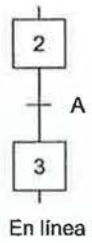
Figura 10.79. *Reglas de síntesis.*

- **Reglas de evolución:**
  - — Etapa inicial. Debe haber al menos una etapa inicial, aunque en casos concretos puede haber varias etapas iniciales.
  - — Activación de etapas. La evolución de una transición implica la desactivación de la etapa anterior y la activación de la etapa siguiente.

10.13.4. Las estructuras

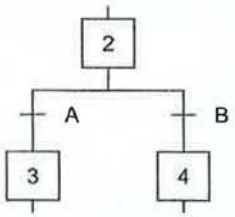
Un diagrama de Grafcet puede adoptar varias estructuras, siendo estas:

- **En línea.** Dos etapas están separadas por una transición.
- **Divergencia en O.** Una etapa puede evolucionar hacia varias etapas.
- **Convergencia en O.** Varias etapas pueden evolucionar convergiendo hacia una única etapa.



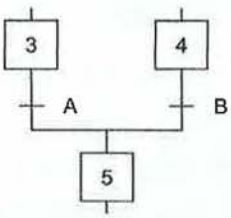
En línea

Figura 10.80. *Grafcet en línea.*



Divergencia en O

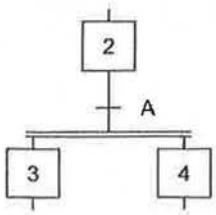
Figura 10.81. *Grafcet en divergencia en O.*



Convergencia en O

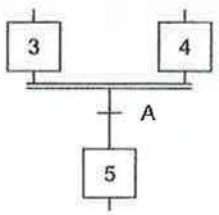
Figura 10.82. *Grafcet en convergencia en O.*

- **Divergencia en Y.** Se emplea cuando se debe representar secuencias que se desarrollan en paralelo. En la Figura 10.83, a partir del estado 2 y al cumplirse la condición A, se pasa a los estados 3 y 4 de manera simultánea.
- **Convergencia en Y.** Normalmente, al terminar las secuencias en paralelo, se suele pasar a un estado único o de convergencia.



Divergencia en Y

Figura 10.83. *Grafcet en divergencia en Y.*



Convergencia en Y

Figura 10.84. *Grafcet en convergencia en Y.*

10.13.5. Implementación del Grafcet

La implementación de un Grafcet se desarrolla en varias etapas o niveles. En un primer nivel se realiza una descripción general del problema y en siguientes niveles se va detallando. Con dos o tres niveles es suficiente para detallar un proceso. En el último nivel se detallan los elementos eléctricos que intervienen en el proceso de automatización.

Actividad resuelta 10.3

Se desea automatizar el siguiente proceso. Un operario acciona el pulsador de una máquina. Esta máquina dispone de una pieza móvil que avanza y al llegar al final de recorrido, se detiene 3 segundos y retrocede hasta situarse como al principio. Solo se pondrá en marcha si se pulsa y la pieza móvil esta replegada. Una señalización indicará cuándo la máquina está en estado de avance.

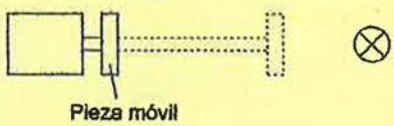


Figura 10.85. *Enunciado.*

Solución:

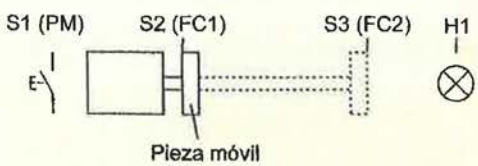


Figura 10.86. *Elementos que intervienen.*

Se asignan las entradas y las salidas del sistema.

Tabla 10.13. Asignación de entradas y salidas.

Elemento	Autómata
S1 – Pulsador de marcha.	I1
S2 – Final de carrera 1. Pieza al final.	I2
S3 – Final de carrera 2. Pieza al principio.	I3
KM1 – Motor en giro directo (avance).	Q1
KM2 – Motor en giro inverso (retroceso).	Q2
H1 – Piloto de señalización.	Q3

Se plantea el Grafcet de primer nivel (descriptivo).

Figura 10.87. *Grafcet de primer nivel.*

A cada etapa se le asigna una marca (M). Y con estos datos se genera el Grafcet de segundo nivel (Figura 10.88).

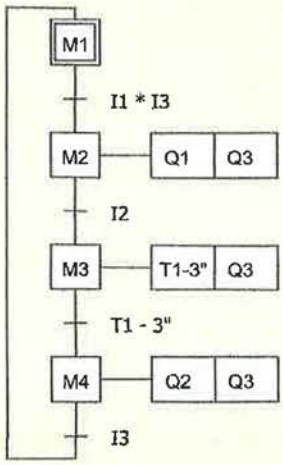


Figura 10.88. *Grafcet de segundo nivel.*

La etapa inicial (M1) se activa cuando las demás etapas están inactivas.

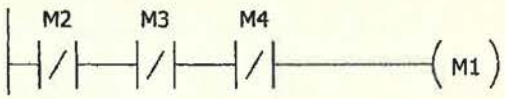


Figura 10.89. *Etapa inicial.*

La condición de transición implica activar la etapa siguiente y desactivar la anterior. La activación se realiza poniéndolo a Set y la desactivación poniéndolo a Reset. Estas instrucciones de Set y Reset se emplean con marcas (M).

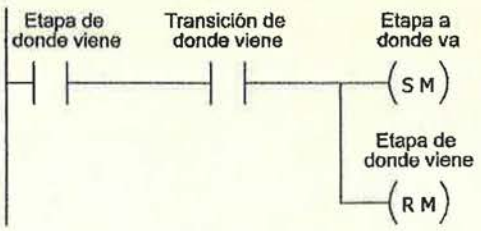


Figura 10.90. *Evolución.*

Así, el paso de la etapa M1 a la etapa M2:

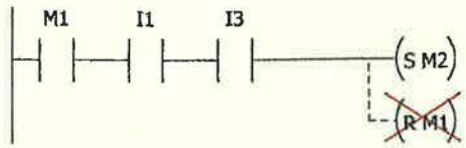


Figura 10.91. *Evolución de la etapa M1 a la M2.*

Todos los Set y Reset del estado inicial (S M1 y R M1) no se ponen puesto que esa opción ya está contemplada con la primera línea (Figura 10.89)

Y para el resto de pasos se procede de la misma manera:

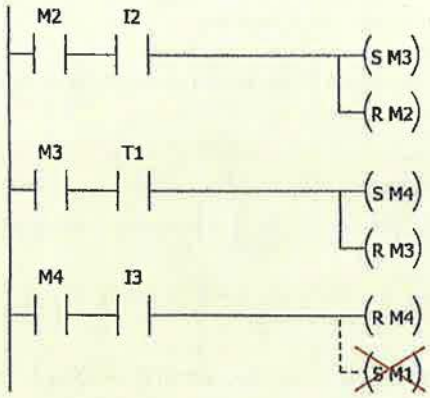


Figura 10.92. *Evolución del resto de etapas.*

Una vez terminadas las secuencias, cada marca realiza sus acciones (activación de salidas, activación de temporizadores, etc.).

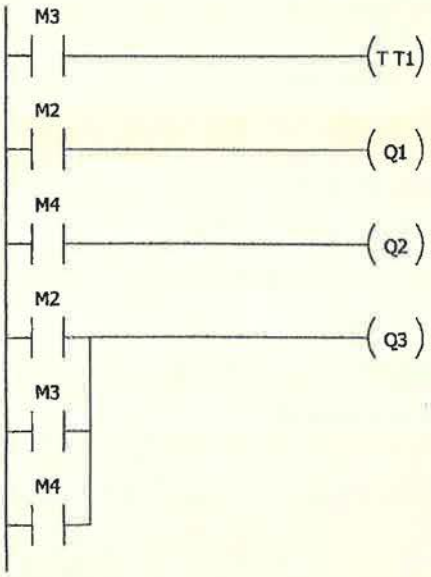


Figura 10.93. *Activación de las salidas.*

Y su conexión eléctrica de las entradas y salidas al autómata será:

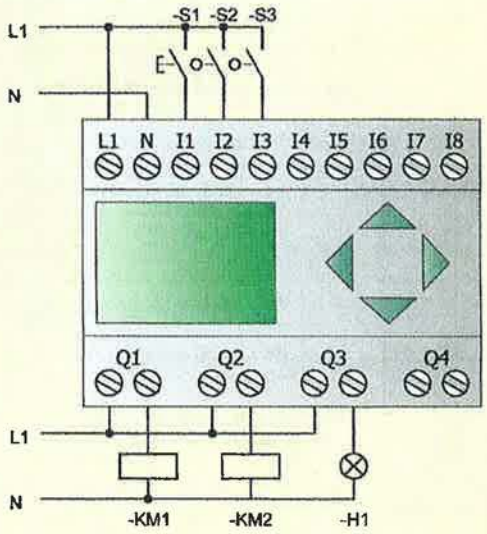


Figura 10.94. *Conexión.*

Actividades de comprobación

10.1. A un módulo de entradas analógicos se puede conectar:

a) Una sonda de temperatura.

b) Un pulsador de paro de emergencia.

c) Un contactor con bobina a una tensión de 24 V_{DC}.

10.2. ¿Qué característica poseen los módulos de salidas digitales a transistor?

a) Se puede conectar cualquier receptor sin importar la tensión.

b) Permiten controlar corrientes mayores que los otros tipos de salidas.

c) Poseen una alta frecuencia de conmutación.

10.3. ¿Qué protección extra se debe emplear cuando un autómata programable maneja cargas resistivas?

a) No necesitan protección extra.

b) Protección mediante diodo y resistencia.

c) Protección mediante diodo y varistor.

10.4. Los paneles de operación se emplean para:

a) Como dispositivo de salida de información hacia el operario.

b) Como dispositivo de entrada de información para el operario.

c) Como dispositivo de entrada y salida de información.

10.5. El bus de comunicación AS-i se emplea para:

a) Que el autómata dé las ordenes a los salidas.

b) En una red de enlace entre los sensores y los actuadores.

c) Comunicarse entre sí una red de varios autómatas programables.

10.6. En la comunicación mediante Ethernet industrial, una de sus características es:

a) Apropiada para grandes distancias entre dispositivos.

b) Sensible a las perturbaciones electromagnéticas generadas principalmente por la maniobra de motores.

c) Permite un gran intercambio de información a costa de bajar su velocidad de transmisión.

10.7. En el lenguaje del diagrama de bloques funcionales, dos dispositivos en serie se representan por:

a) Una función AND.

b) Una función OR.

c) Una función NOT.

10.8. En el lenguaje del diagrama de bloques funcionales, dos dispositivos en paralelo se representan por:

a) Una función AND.

b) Una función OR.

c) Una función NOT.

10.9. En el lenguaje de contactos, la instrucción Set:

a) Activa su salida, solo si su entrada está activada.

b) Activa su salida, y permanece en ese estado aunque se desactive su entrada.

c) Alterna su estado a cada variación de su entrada, en el primer pulso se activa y en el segundo pulso se desactiva.

10.10. En un diagrama Grafcet:

a) Las etapas están separadas por dos transiciones.

b) Puede haber etapas sin transiciones.

c) Solo puede haber una única transición entre dos etapas.

10.11. En un diagrama de Grafcet, la divergencia en O:

a) Representa el paso de una etapa a varias etapas en paralelo de las cuales solo puede ir hacia una única rama.

b) Representa el paso de una etapa a varias etapas en paralelo de las cuales todas se desarrollan en paralelo.

c) Representa el paso de varias etapas a una única etapa de las cuales las anteriores etapas se desarrollaron en paralelo.

10.12. En un diagrama de Grafcet, la convergencia en Y:

a) Representa el paso de varias etapas en paralelo a una única etapa

b) Representa el paso de una etapa a varias etapas que se desarrollan en paralelo.

c) Representa el paso de una etapa a otra etapa que se desarrolla en serie.

Actividades de aplicación

10.1. Clasifica los autómatas en función de estructura externa.

10.2. Indica las partes de las que se compone un autómata a nivel interno.

10.3. ¿Para qué se emplea una fuente de alimentación en los autómatas?

10.4. ¿Qué diferencia hay entre una entrada analógica y una digital o discreta?

10.5. ¿Qué diferencia hay entre un módulo de salida a relé y un módulo a salida a transistor?

10.6. ¿Qué diferencias existen entre un display o pantalla alfanumérica y una gráfica?

10.7. ¿Qué ventajas tienen los sistemas de conexión mediante bus frente a las conexiones punto a punto? Cita al menos dos sistemas de bus.

10.8. ¿Qué es un sistema *SCADA*?

10.9. Indica los tipos de lenguajes de programación que conozcas.

10.10. Cita las áreas o mapa de memoria que se emplean en la programación de autómatas. Indica cuál es la letra identificativa que les representa.

10.11. ¿Para qué se emplean los diagramas de Grafcet?, ¿en qué se basan?

Casos prácticos

10.1. Realiza el esquema de maniobra en lógica cableada de un arranque de un motor trifásico el cual se puede poner en marcha desde dos pulsadores y parar desde uno. El circuito dispone de protección mediante relé térmico. Dota al circuito de señalización de disparo del relé térmico y de funcionamiento del motor.

Pasa el esquema a lógica programada empleando un autómata programable.

10.2. Realiza el esquema de maniobra en lógica cableada de un arranque de un motor trifásico con inversión de giro. El motor dispone de protección mediante relé térmico. Dota al circuito con señalización del sentido de giro.

Pasa el esquema a lógica programada empleando un autómata programable.

10.3. Un automatismo, consistente en un motor trifásico con arranque estrella-triángulo, se pone en marcha al ser activado un final de carrera. Se puede parar en cualquier momento con un pulsador o bien al activarse un sensor de proximidad (a 3 hilos). La alimentación eléctrica del autómata es a 24 V_{cc}. Realiza cómo sería la conexión eléctrica entre el autómata y todos los captadores y actuadores.

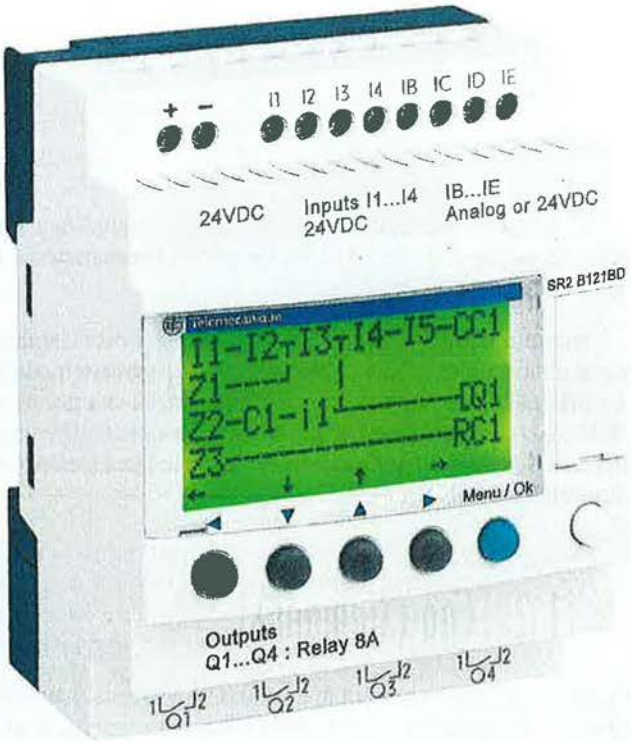
10.4. Realiza el diagrama de Grafcet de primer nivel para el siguiente automatismo:

- Estando el circuito en reposo, si se pulsa sobre el pulsador S1, se activa KM1 que pone en funcionamiento un motor trifásico (M1).
- Transcurridos 5 segundos, se activa un contactor (KM2), que pone en funcionamiento un segundo motor.
- Se mantiene en esta situación hasta que se acciona el pulsador de paro (S2), volviendo el circuito al estado de reposo.

10.5. Realiza el diagrama de Grafcet de segundo nivel para el automatismo del caso anterior.

Relés programables

11



Contenidos

Objetivos

- Tener conocimiento de los principales autómatas del mercado.
- Aprender a programar en el lenguaje de diagramas de bloques de funciones.
- Aprender a programar en el lenguaje de contactos.
- Aprender a programar con el Logo.
- Aprender a programar con el Zelio.
- Conocer cómo conectar las entradas a un autómata.
- Conocer cómo conectar las salidas a un autómata.

Los relés programables son unos autómatas compactos muy empleados en aplicaciones donde no se requiere una gran capacidad de procesamiento. Su bajo coste y facilidad de programación hace que su uso se esté generalizando.

Todas las marcas comerciales que tienen en su catálogo autómatas, tienen algún modelo de relé programable. De entre ellas, un alto porcentaje de instalaciones se llevan a cabo con el modelo Logo de Siemens y Zelio de Schneider Electric. Estas mismas marcas proveen de un *software* para la realización de la programación.

11.1. Los relés programables

A la hora de realizar una instalación eléctrica con un automatismo programable, el primer paso es la selección de dicho autómata. Su elección dependerá de la complejidad a tratar. Existen una serie de aplicaciones donde no es necesario contar con una elevada capacidad de procesamiento ni de un gran número de entradas o salidas. En estos casos se emplean relés programables, que si bien tienen ciertas limitaciones, son idóneos para estas aplicaciones.

RECUERDA

A mayor prestación de un modelo de autómata, su precio se incrementa notablemente. Por ello el autómata ideal para una aplicación será aquel que a menor coste es capaz de resolver un problema de automatización.

Para estos casos surgieron los relés programables. Los relés programables se sitúan en el escalafón más bajo de la automatización, lo cual los hace apropiados para tareas simples de control, con pocas entradas o salidas.

Todas las firmas comerciales de autómatas disponen de algún modelo de tales requerimientos, por ejemplo están: Siemens con el Logo, Schneider con el Zelio, Omron con el Zen, Moeller con el Easy, etc.

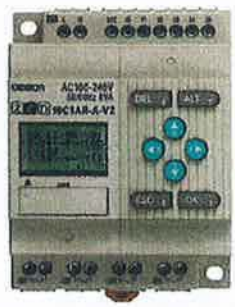


Figura 11.1. *Zen. (Cortesía de Omron.)*

Figura 11.2. *Zelio. (Cortesía de Schneider Electric.)*

Figura 11.3. *Logo (Cortesía de Siemens.)*

Figura 11.4. *Easy (Cortesía de Moeller.)*

Son modelos de tipo compacto, lo que significa que cada dispositivo dispone de todas las partes necesarias: entradas, salidas, CPU, etc.

Aunque debido a la gran aceptación que están teniendo algunos modelos han evolucionado y permiten poder acoplarles algún módulo extra que le aporta una gran flexibilidad, como por ejemplo: entradas para sondas de temperaturas de tipo Pt-100, módulos de comunicaciones para domótica (EIB-KNX), etc.

11.2. Logo (Siemens)

El Logo es el relé programable básico de Siemens. Inició su andadura comercial en el año 1996. Desde entonces está en continua evolución. Sus primeras versiones eran muy simples, su programación se realizaba solo mediante la programación por diagrama de bloques funcionales. Hoy en día se puede realizar en bloques de funciones o en lenguaje de contactos.

11.2.1. Tipos de Logo

Los módulos base de Logo están disponibles para dos clases de tensión:

Clase 1 ≤ 24 V. Comprende los modelos de 12 V_{CC}, 24 V_{CC} y 24 V_{CA}.

Clase 2 > 24 V. Comprende los modelos de 115 a 240 V CA/CC.

También existen dos versiones:

Logo Basic. Versión con display.

Logo Pure. Versión sin display.

Figura 11.5. *Logo Pure.*

Figura 11.6. *Logo Basic.*

El Logo dispone de modelos de 8 y 12 entradas, pudiendo ser estas analógicas o digitales. El número de salidas es de 4 o de 8, de tipo salida a relé o a transistor. Asimismo, hay versiones que ya cuentan internamente con algún sistema de comunicación, aunque se le puede dotar con módulos externos para adquirir dichas capacidades.

11.2.2. Partes del Logo

El logo consta de las siguientes partes:

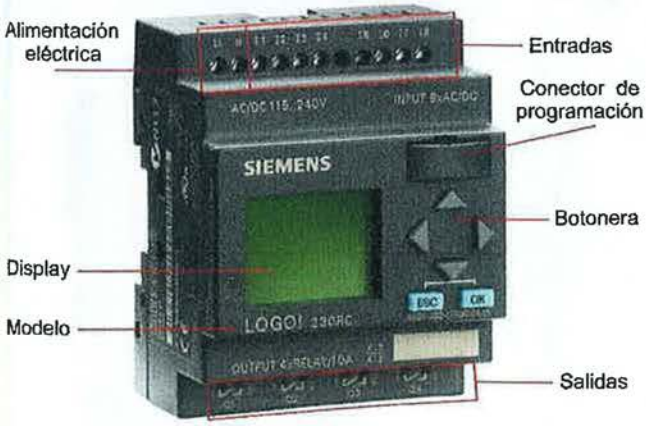


Figura 11.7. *Logo (Cortesía de Siemens.)*

- **Alimentación eléctrica.** Consta de dos bornes para la alimentación eléctrica. Si es un modelo de 230 V_{CA}, se debe proteger mediante un interruptor magnetotérmico. Si es un modelo de 24 V o de 12 V, es necesario contar con una fuente de alimentación de dicho valor.
- **Entradas.** El modelo básico cuenta con 8 entradas digitales, numeradas desde la I1 a I8.
- **Salidas.** El modelo básico cuenta con 4 salidas digitales, numeradas desde la Q1 a Q4.
- **Conector de programación.** Es el conector mediante el cual se traspasa el programa al Logo. Se puede conectar a un ordenador o a una tarjeta de memoria. Este conector va protegido mediante una tapa.
- **Botonera.** Dispone de un panel de control con 4 teclas de cursor más 2 teclas de función. Las teclas de cursor se pueden utilizar a nivel interno mediante programa.
- **Display o pantalla LCD.** Permite visualizar mensajes, así como la gestión de menús de programación.
- **Conector TD.** En un lateral cuenta con un conector para un panel de operación el cual dispone de un display junto con una botonera.
- **Interfaz de ampliación.** Dispone en el otro lateral de un conector mediante el cual es posible su ampliación a través de módulos.

11.2.3. Conexión a la fuente de alimentación

Para proteger al Logo frente a picos de tensión, basta con emplear un varistor conectado a la entrada de tensión. La tensión de servicio de este varistor debe ser como mínimo un 20 % superior a la tensión de servicio.

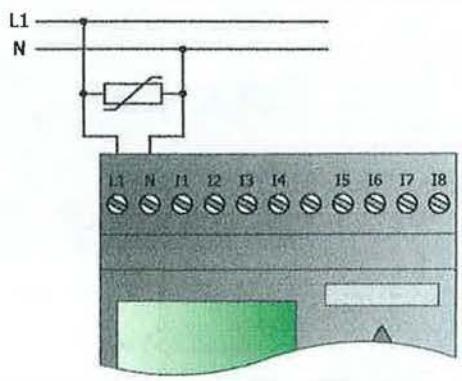


Figura 11.8. *Conexión de un varistor al Logo.*

11.2.4. Conexión de las entradas

Las entradas de señal se conectarán a la parte de las entradas, de tal manera que reciban fase o positivo (en función del modelo). Esta tensión la detectará un optoacoplador de entrada y la interpretará como un "1" lógico.

El bloque de las entradas está dividido en grupos. Por ejemplo, los modelos de ocho entradas cuentan con dos grupos: de I1 a I4 y de I5 a I8. Cada grupo solo puede recibir una fase, aunque puede haber fases diferentes en distintos grupos (Figura 11.9).

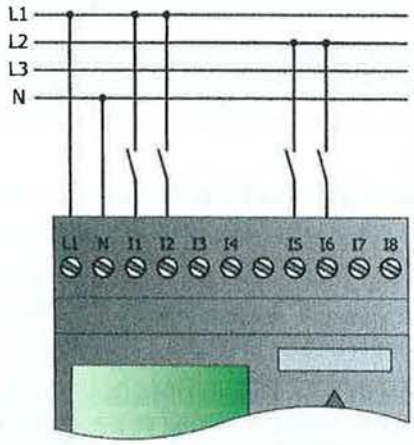


Figura 11.9. *Conexión de las entradas.*

11.2.5. Conexión de las salidas

Se puede conectar cualquier carga a la salida siempre y cuando se respete la corriente máxima en las salidas que marca el fabricante según el modelo. Estas salidas deben contar con algún sistema de protección, como por ejemplo con un interruptor magnetotérmico o con fusible.

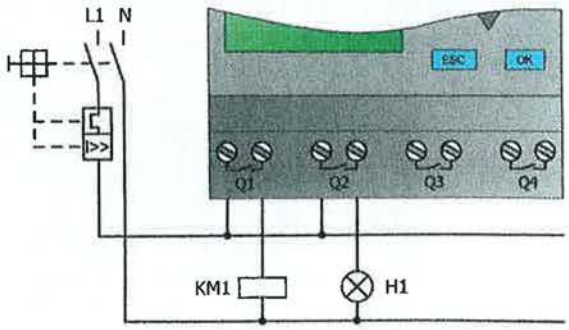


Figura 11.10. *Conexión de las salidas.*

11.2.6. Los módulos de expansión

El Logo de Siemens dispone de una serie de módulos de expansión que mejoran sus prestaciones:

- Los módulos de ampliación de entradas, tanto digitales como analógicas.
- Los módulos de ampliación de salidas, tanto digitales como analógicas.
- El display de texto externo. Con visualización y cuatro teclas de función.
- Los módulos de comunicación KNX. El estándar EIB-KNX se emplea en el área de domótica.
- Los módulos de comunicación AS-i. Se facilita la conexión eléctrica entre los actuadores y sensores mediante un bus de comunicación con la consiguiente reducción del cableado.
- Los módulos de comunicación Ethernet. La red Ethernet es de mayor aceptación actualmente en el área de comunicación industrial.

Figura 11.11. *Módulo de entradas analógicas.*

Figura 11.12. *Módulo bus AS-i.*

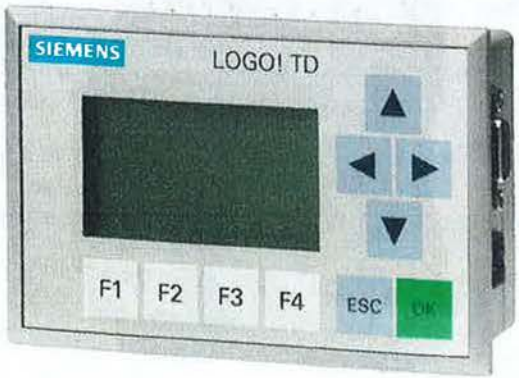


Figura 11.13. *Display externo.*

También en el área de las memorias externas ha evolucionado, desde los cartuchos de memoria hasta las memorias de tipo SD.

Entre la innovaciones más recientes se encuentran aplicaciones para *smartphones* y *tablets* con la posibilidad de conectarse al Logo.

11.2.7. La programación del Logo

Para realizar la programación del Logo se pueden emplear dos métodos: mediante el propio Logo utilizando la botonera, o mediante un ordenador y su posterior transferencia. En este caso nos centraremos sobre el ordenador y su *software*.

El *software* para ordenador que proporciona Siemens se llama LogoSoft Comfort.

La interfaz de usuario del *software* de programación de LogoSoft consta de las siguientes partes:

- **Barra de menús.** Está situada en la parte superior. En ella se encuentran los comandos para la elaboración, configuración y transferencia de programas.
- **Barra de símbolos estándar.** Contiene los botones de comandos de uso general, tales como: crear, cargar y guardar el programa; cortar y copiar; transferencia de programas, etc.
- **Barra de herramientas.** Se encuentra a la izquierda. Contiene una serie de botones que facilitan y agilizan la creación de los programas.
- **Barra de estado.** Proporciona información adicional, tal como el nivel de zoom, la situación de un programa o la actividad.
- **Plataforma de programación.** Sobre esta superficie se desarrolla el programa.
- **Ventana de información.** Se encuentra en la parte inferior y proporciona indicaciones e información.

11.2.8. Los bloques de funciones

En la programación mediante bloques funcionales, cada bloque representa una función. Existen varios tipos de funciones que se agrupan entre sí. Para acceder a ellas, se puede ir a la barra de herramientas donde se encuentran estos.

El LogoSoft adapta el entorno de programación en función del modelo empleado. Así, por ejemplo, un modelo con solo 8 entradas, solo muestra esas 8 y un modelo con 12 entradas mostrará 12.

Al situar el cursor sobre cada una de ellas, proporciona información acerca del elemento.

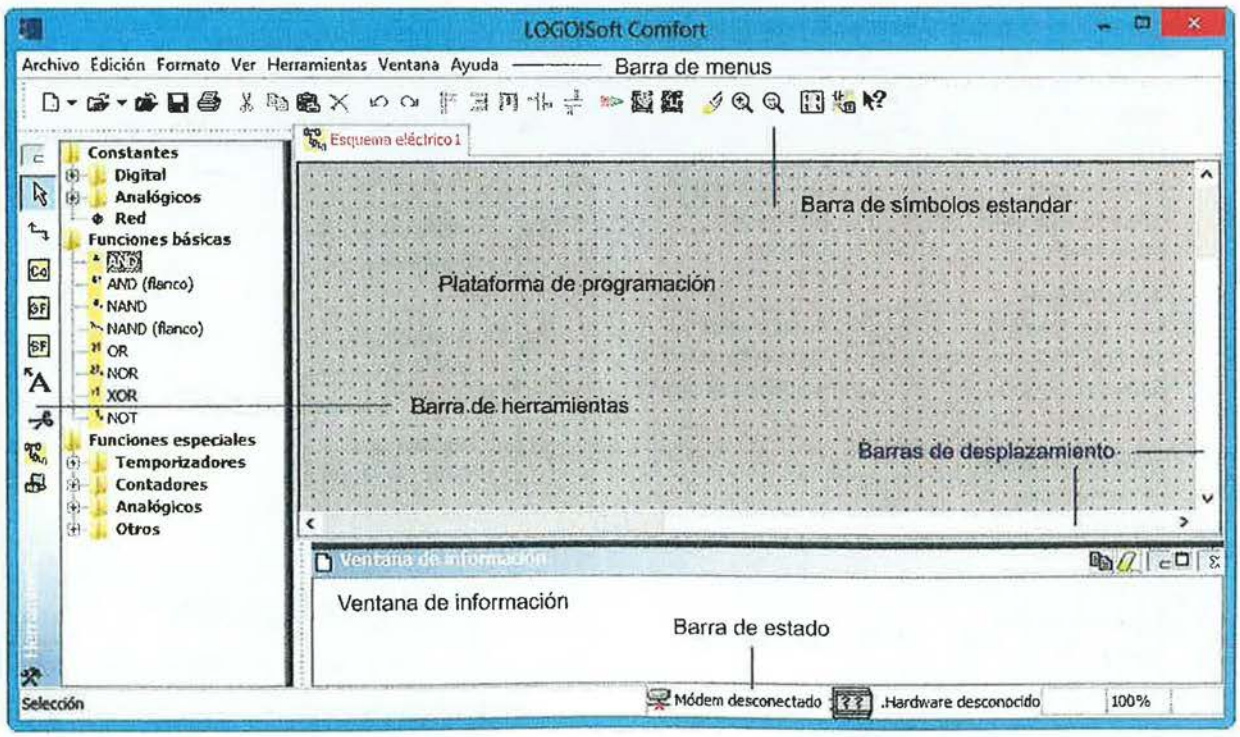


Figura 11.14. *Software de programación LogoSoft Confort.*

Los conectores

Representan básicamente a los diferentes tipos de entradas y salidas.

Figura 11.15. *Conectores.*

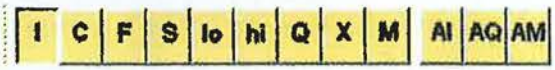


Figura 11.16. *Tipos de conectores.*

Tabla 11.1. Tipos de conectores.

I	Son las entradas. Dependiendo del modelo varía su número.	Q	Son las salidas.
C	Son las teclas del cursor. Se programan como entradas.	X	Son conectores abiertos o no utilizados.
F	Son las teclas de función que posee el panel de operaciones externo. Dispone de cuatro teclas.	M	Son las marcas. Representan a relés auxiliares internos.
S	Son los bits de registro de desplazamiento. Su cantidad depende del modelo.	AI	Entrada analógica.
Io	Representa un nivel lógico de "0".	AQ	Salida analógica
hi	Representa un nivel lógico de "1".	AM	Marcas analógicas.

Las funciones básicas

Representan a las funciones realizadas por las puertas lógicas. Las funciones AND, NAND, OR y NOR cuentan con cuatro entradas por bloque.

Figura 11.17. *Funciones básicas.*

Figura 11.18. *Tipos de funciones básicas.*

Tabla 11.2. Tipos de funciones básicas.

&	Función lógica AND. Por niveles.	≥1	Función lógica OR.
&↑	Función lógica AND. Por flancos.	≥1	Función lógica NOR.
&	Función lógica NAND. Por niveles.	=1	Función lógica XOR (OR exclusiva).
&↓	Función lógica NAND. Por flancos.	1	Función lógica NOT. Inversor.

Las funciones especiales

Las funciones especiales se dividen en cuatro grupos:

- Temporizadores. Son funciones en las cuales interviene el parámetro tiempo.
- Contadores. Realizan operaciones de cuenta.

- Analógicos. Son funciones relacionadas con parámetros de tipo analógico.
- Otros. Incluyen funciones diversas tales como: los textos o algunos tipos de relés especiales.

Figura 11.19. *Funciones especiales.*

Figura 11.20. *Tipos de funciones especiales.*

Tabla 11.3. Tipos de funciones especiales de temporizadores.

	Retardo a la conexión. La salida se activa trascurrido un tiempo. Entrada **Trg** (*trigger*). Dispara el temporizador.
	Retardo a la desconexión. La salida se desactiva tras haber transcurrido el tiempo configurado. Entrada **Trg**. Dispara el temporizador. Entrada **R**. Una señal en la entrada reinicia el temporizador.
	Retardo a la conexión/desconexión. La salida se activa trascurrido un tiempo (TH) y se desactiva transcurrido otro tiempo (TL), ambos configurables.
	Retardo a la conexión con memoria. Se activa con una señal de impulso. Entrada **Trg** y entrada **R**.
	Relé de barrido (salida de impulso). Una señal de entrada genera una señal de duración configurable. Entrada **Trg**.
	Relé de barrido (activado por flanco). Un impulso de entrada genera un número predeterminado de impulsos de salida con una relación impulso/pausa definida. Entrada **Trg** y **R**, y se configura el número de ciclos.
	Generador de impulsos asíncrono. Genera una onda de impulsos configurable. Entrada **En**. Activa y desactiva el generador de impulsos. Entrada **INV**. Invierte la onda de salida.
	Generador aleatorio. La salida del generador se activa y desactiva dentro de un tiempo configurable.
	Interruptor de alumbrado de escalera. Un flanco de entrada inicia un tiempo configurable que transcurrido desactiva la salida. Entrada **Trg**.
	Interruptor bifuncional. Cuenta con dos funciones: interruptor de impulsos con retardo a la desconexión y como pulsador (alumbrado permanente). Entrada **Trg** y **R**.
	Temporizador semanal. La salida se controla en función de un día de la semana. Dispone de tres programaciones.
MM DD	**Temporizador anual.** La salida se controla en función de la fecha.
AC	**Reloj astronómico.** La salida se controla en función de la salida y puesta del sol según unas coordenadas geográficas.
SW	**Cronómetro.**

Tabla 11.4. Tipos de funciones especiales de contadores.

+/-	**Contador adelante/atrás.** Realiza una cuenta en modo ascendente o descendente. Entrada **Cnt**. Cuenta los pulsos (flancos de subida). Entrada **Dir**. Indica la dirección de la cuenta (0 ascendente y 1 descendente). Entrada **R**. Reinicia el contador.
	Contador de las horas de funcionamiento.
	Selector de umbral.

Tabla 11.5. Tipos de funciones especiales de tipo analógico.

+= A→	**Instrucción aritmética.**	A→	**Rampa analógica.**
ΔA	**Comparador analógico.**	A→	**Regulador PI.**
/A	**Conmutador analógico de valor umbral.**		**Modulación por ancho de impulsos (PWM).**
A→	**Amplificador analógico.**	Ax	**Filtro analógico.**
A ±Δ	**Vigilancia de valor analógico.**	Ax	**Máximo/mínimo.**
/A Δ	**Conmutador analógico de valor diferencial.**	Ax	**Valor medio.**
A→	**Multiplexor analógico.**		

Tabla 11.6. Tipos de funciones especiales de otros tipos.

RS	**Relé autoenclavado.** Entrada **S.** Con una señal de entrada, la salida se activa y permanece en ese estado. Entrada **R.** Con una señal de entrada, la salida se desactiva y permanece en ese estado.		
RS	**Relé de impulsos.** Funciona igual que un telerruptor. Cambia el estado de cada salida por cada pulso de entrada.		
	Texto de aviso.	>>	**Registro de desplazamiento.**
	Interruptor de *software*.	+= E→	**Detección de error de la instrucción aritmética.**

Con las funciones especiales es posible desarrollar aplicaciones complejas de una manera muy sencilla.

Debido a que el Logo está en continua evolución, en cada nueva versión se incluyen nuevas funciones. Así, es posible que al emplear un Logo de las primeras versiones, algunas de las funciones no estén disponibles.

Actividad resuelta 11.1

Se desea realizar el proceso para encender y apagar una lámpara desde dos puntos cualesquiera (función conmutador).

Solución:

Para resolver el problema hay que emplear un bloque de función especial llamado interruptor bidireccional. Este elemento se comporta de igual manera que un telerruptor. Como se activa desde dos puntos, se debe emplear una puerta OR con los pulsadores. Con cualquier pulsación se enciende y con otra pulsación se apaga.

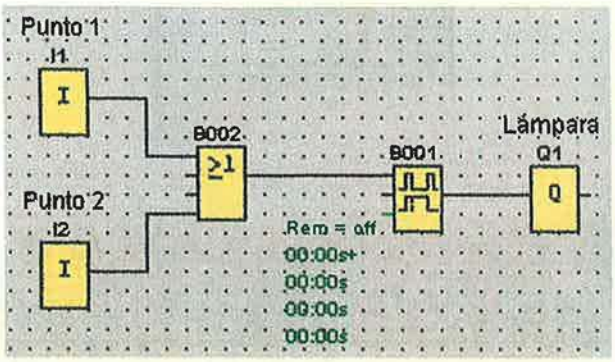


Figura 11.21. *Programa.*

Actividad resuelta 11.2

Se desea realizar el proceso para automatizar un sistema de riego para que se active los sábados y realice ese riego desde las 10:00 hasta las 10:15 h. El sistema contará con un interruptor, de tal manera que para realizar el riego el interruptor debe estar cerrado.

Solución:

Como salida se va a emplear una electroválvula, la cual al estar activada, se abre permitiendo la circulación del agua.

Como debe cumplir con dos condiciones (interruptor y día/hora de la semana) se debe colocar una puerta AND.

Para saber el día y la hora se empleará un temporizador semanal. Una vez colocado el temporizador en la plataforma de programación, este se debe configurar. Para ello nos dirigimos al menú *Edición* y entramos en *Propiedades del bloque*. Cada temporizador semanal permite hasta tres programaciones.

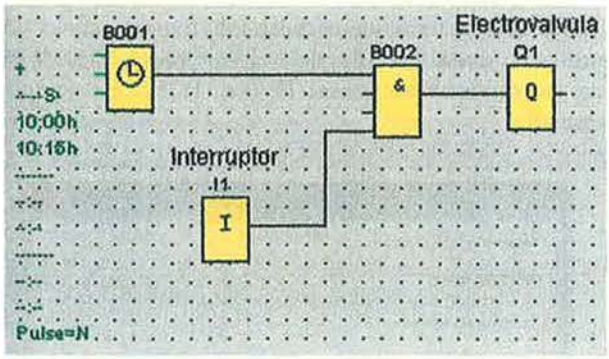


Figura 11.22. *Programa.*

Actividad resuelta 11.3

Se desea automatizar el control de una luz, de tal manera que cuando un sensor de presencia detecte una persona provoque el encendido de una lámpara. Cuando ya no detecte la presencia de nadie y pasen 3 segundos la lámpara se apagará.

Solución:

Un sensor de presencia se comporta como un interruptor. En este caso, si detecta a alguien el interruptor se cierra y se abre cuando deja de detectar.

Para controlar el tiempo se emplea un temporizador con retardo a la desconexión calibrado a un tiempo de 3 segundos.

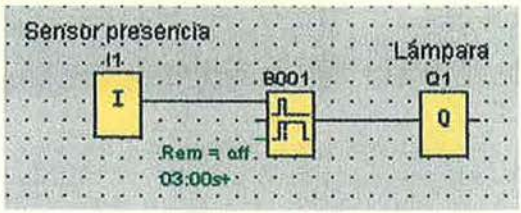


Figura 11.23. *Programa.*

11.2.9. La realización de un programa

Partimos del supuesto de que se desea pasar una automatización mediante lógica cableada a lógica programada. El circuito de ejemplo de partida consiste en el arranque de un motor mediante una botonera marcha-paro y el arranque de un segundo motor transcurrido un tiempo, tal y como se muestra en el esquema de la Figura 11.24.

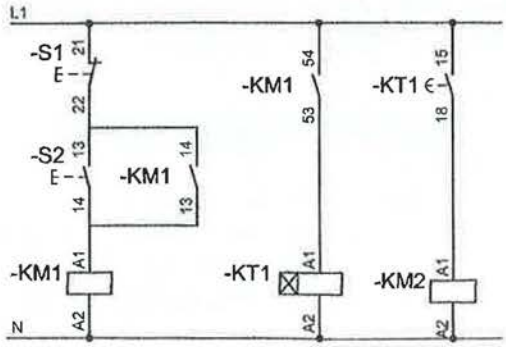


Figura 11.24. *Esquema eléctrico.*

Tabla 11.7. Lista de referencias.

Elemento	Desig. Logo	Descripción
S1	I1	Pulsador de paro
S2	I2	Pulsador de marcha
KM1	Q1	Motor 1
KM2	Q2	Motor 2
KT1		Temporizador

Con los datos de los elementos de entradas y salidas se elabora la tabla de referencias donde se asigna cada elemento a las entradas y salidas del Logo (Tabla 11.7).

Con estos datos, se obtienen las funciones lógicas para cada receptor:

$$Q1 = \overline{I1} \times (I2 + Q1)$$

$$KT1 = \overline{Q1}$$

$$Q2 = KT1$$

SABÍAS QUE

El *software* de programación LogoSoft Comfort está disponible para su descarga gratuita desde la web del fabricante, www.siemens.es.

Una vez lanzado el *software* LogoSoft Comfort, un programa se empieza con la opción *Nuevo* del menú *Archivo* o desde el icono de la barra de menús.

El primer paso es indicar al programa qué sistema de programación se va a emplear (Figura 11.25). En esta unidad didáctica se va a utilizar el sistema de diagramas de

funciones (FUP) con el Logo, aunque también se puede programar con el lenguaje de contactos.

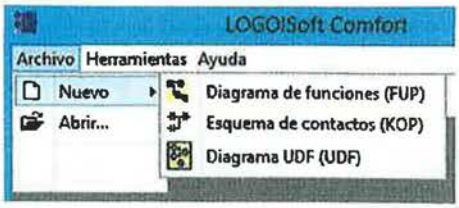


Figura 11.25. *Menú archivo nuevo.*

También es necesario indicar el *hardware* empleado, opción que se encuentra en el menú *Herramientas* (Figura 11.26). Aquí indicaremos qué tipo de Logo se va a emplear.

Figura 11.26. *Selección de dispositivos.*

Partiendo de las funciones lógicas, se colocan los diferentes bloques funcionales, empezando por los conectores (entradas y salidas) y continuando por las funciones (funciones básicas, funciones lógicas).

De los conectores, se selecciona la entrada (I) y se lleva a la plataforma de programación. Con el botón derecho, se entra en las propiedades y se configura el pulsador de paro como normalmente cerrado (Figura 11.27).

Con el pulsador de marcha (S2-I2) se procede igual pero se configura como contacto normalmente abierto. En la pestaña de comentario se puede añadir algún texto que ayude a identificar la entrada (Figura 11.27).

El siguiente paso es colocar las funciones generales (AND, OR) y funciones especiales (temporizador con retardo a la conexión). Entrando a sus propiedades de bloque, se configura su tiempo (Figura 11.28), por ejemplo 3 segundos.

Figura 11.28. *Configuración del temporizador.*

Una vez colocados todos los bloques funcionales tan solo falta realizar el conexionado entre ellos. Para ello se debe seleccionar la función *Conectar* de la barra de herramientas. Se pincha en una conexión del bloque a conectar y se suelta en el otro punto.

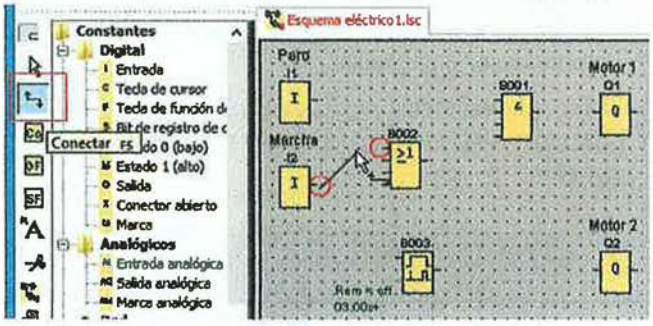


Figura 11.29. *Conexionado de bloques funcionales.*

Figura 11.27. *Configuración de las entradas.*

Una vez conectados todos los bloques funcionales (Figura 11.29), el programa estaría finalizado (Figura 11.30) y pasaría a la fase de simulación.

RECUERDA

Para realizar la conexión eléctrica entre dos bloques funcionales se debe seleccionar *Conectar* y pinchar con el ratón en el punto de inicio de la conexión y sin soltar se arrastra hasta el destino donde se soltará.

Una vez conectados, se puede mover las conexiones eléctricas para situarlas a gusto.

11.2.10. La simulación

Una de las grandes facilidades que ofrece el *software* LogoSoft Comfort es la posibilidad de simular la programación realizada, aspecto que facilita la depuración del programa.

En la barra de herramientas se encuentra el modo de simulación. Al entrar en ella el entorno cambia pasando a disponer de los útiles de simulación. Además, en la parte inferior de la plataforma de programación, aparecen las entradas y salidas utilizadas en el programa, en las cuales es posible ver su estado (activado o desactivado) y actuar sobre ellas. Junto a estas, aparecen los botones de iniciar, pausar y cancelar la simulación.

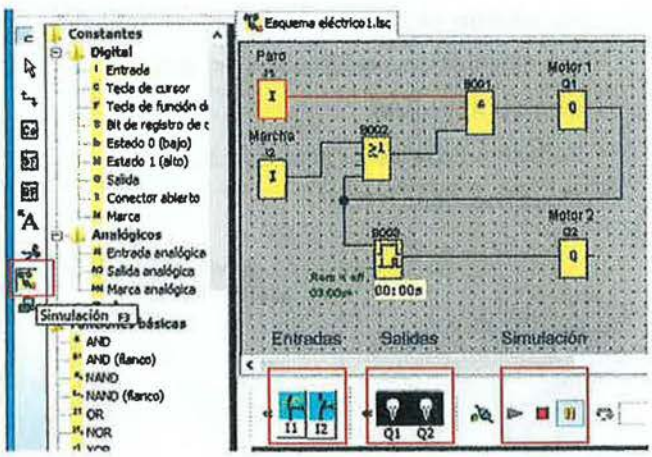


Figura 11.30. *Programa terminado y en fase de simulación.*

En esta barra de simulación aparecerán tantas entradas y salidas como se hayan empleado en la programación.

Se dispone de tres botones para el control de la simulación: *Iniciar simulación*, *Cancelar simulación* y *Realizar pausa de la simulación*. Por ello, el primer paso es iniciar la simulación activándola.

El conexionado ha cambiado de color. Las líneas en color rojo son líneas a nivel lógico 1, y en color azul a nivel lógico 0.

Al activarse una salida, esta se marca cambiando de color en la barra de simulación.

11.3. Zelio Logic (Schneider)

El relé programable Zelio pertenece a la gama más baja de autómatas de la marca Schneider. El Zelio se puede programar tanto en lenguaje de contactos como en diagrama de bloques funcionales, de similar manera que el Logo.

SABÍAS QUE

Telemecanique es la marca que en un principio comercializaba el Zelio, por ello muchos modelos van etiquetados así. Telemecanique pertenece al grupo Schneider Electric y la tendencia de este grupo es utilizar el nombre del grupo como marca comercial.

11.3.1. Tipos de Zelio

La gama Zelio se divide en dos grupos:

- SR2. Modelos de tipo compactos.
- SR3. Modelos de tipo modular.

El rango de tensiones es el típico: 12 V_{CC}, 24 $V_{CA/CC}$ y 100-240 V_{CA}. El número de entradas varía entre modelos, habiendo de 6, 8 y 12 entradas y de 4 y 8 salidas. Existen modelos que manejan tanto entradas como salidas de tipo digital con otros modelos que las combinan.

Figura 11.31. *Zelio (con pantalla).*

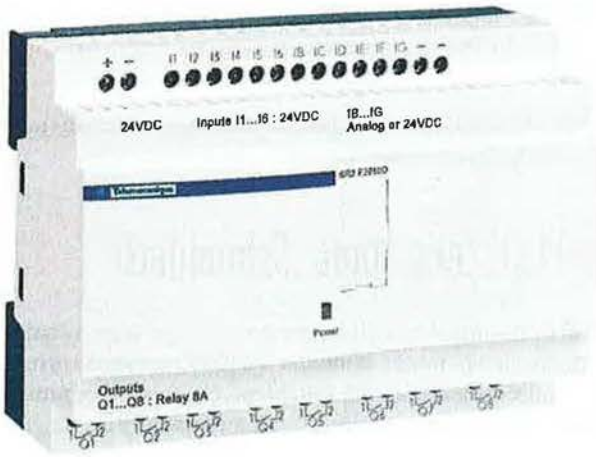


Figura 11.32. *Zelio (sin pantalla).*

11.3.2. Partes del Zelio

El Zelio consta de las siguientes partes:

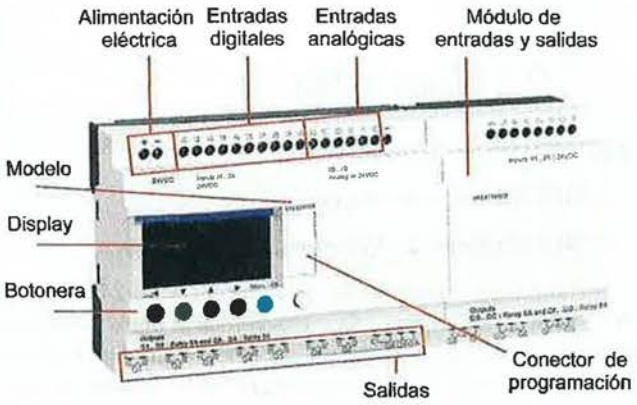


Figura 11.33. *Partes del autómata Zelio (Cortesía de Schneider).*

- **Alimentación eléctrica.** Consta de dos bornes para la alimentación eléctrica. Si es un modelo de 24 V o de 12 V, es necesario contar con una fuente de alimentación de dicho valor.
- **Entradas.** Están situadas en la parte superior del Zelio. Su número y naturaleza depende del modelo. En las entradas de tipo analógico su valor oscila entre 0-10 V. Van etiquetadas con la letra I.
- **Salidas.** Están situadas en la parte inferior del Zelio. Su número depende del modelo. Van etiquetadas con la letra Q. El tipo de salida puede ser a relé (8 A) o a transistor.
- **Botonera.** Dispone de un panel de control con cuatro teclas de cursor más dos teclas de función. Estas teclas permiten configurar, programar, controlar y supervisar su desarrollo.

  La tecla *Mayus* corresponde a la de color blanco. Al pulsarla aparece un menú contextual encima de las teclas de cursor (de color gris).

  La tecla *Menú/Aceptar* corresponde a la de color verde. Se emplea para realizar todas las validaciones: menú, submenú, programa, parámetros, etc.

  Las teclas de cursor (de color gris) se pueden utilizar en la programación como entradas.
- **Display o pantalla LCD.** Permite visualizar mensajes, así como la gestión de menús de programación. Se compone de cuatro líneas de 18 caracteres.
- **Conector de programación.** Es el conector mediante el cual se traspasa el programa al Logo. Se puede conectar a un ordenador o a una tarjeta de memoria. Este conector va protegido mediante una tapa.

11.3.3. Los módulos de expansión

El Zelio de Schneider dispone de una serie de módulos de expansión que mejoran sus prestaciones:

- Los módulos de ampliación de entradas, tanto digitales como analógicas.
- Los módulos de ampliación de salidas, tanto digitales como analógicas.
- Los módulos de comunicación para Modbus.
- Los módulos de comunicación para Ethernet.
- Interface Bluetooth.
- Cartuchos de memoria, para copias de seguridad.
- Modem telefónico GSM.

Figura 11.34. *Cartucho de memoria.*

Figura 11.35. *Módulo de 4 entradas y 2 salidas.*

Figura 11.36. *Módulo de 8 entradas y 6 salidas.*

SABÍAS QUE

Existen varios módulos de entradas y salidas en los cuales varía el número de entradas y salidas, de esta manera es muy fácil adaptar el *hardware* a las necesidades físicas y contener el coste de la instalación.

11.3.4. Las conexiones

El conexionado tanto de las entradas como de las salidas, se realiza de idéntica manera que con el Logo.

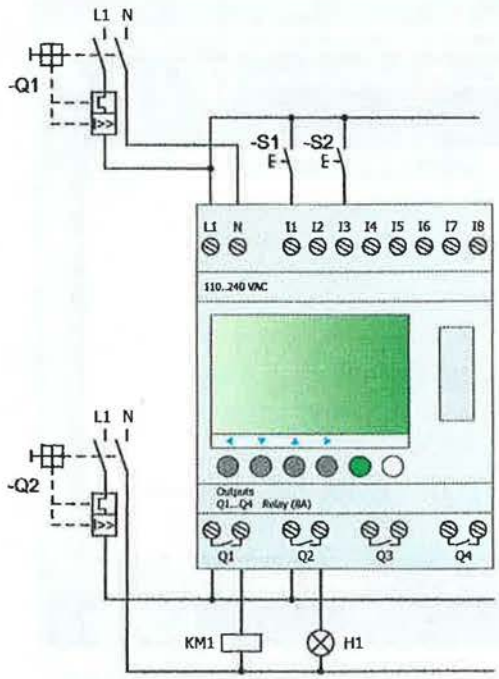


Figura 11.37. *Las conexiones en el Zelio.*

Cuando las entradas reciben una señal de fase o de positivo se interpretará como un 1 lógico, en caso contrario se interpretará como un 0 lógico.

Se debe proteger con un interruptor automático magnetotérmico o con fusible al propio autómata y a las entradas.

Las salidas también deben estar protegidas. A la hora de diseñar la parte de las salidas se debe tener en cuenta que las corrientes de cada receptor no deben sobrepasar a la máxima admisible por el Zelio, normalmente de 8 A.

RECUERDA

La corriente máxima admisible por cada salida del autómata está limitada. Si se necesita controlar receptores mayores se puede intercalar un contactor como elemento de conmutación de potencia.

11.3.5. La programación del Zelio

El Zelio se puede programar desde la botonera situada en el frontal y debajo del display, pero es mucho más cómodo realizarla desde un ordenador y posteriormente se vuelca al Zelio.

Schneider dispone del *software* de programación específico para el Zelio, llamado *Zelio Soft*. Nada más iniciarlo nos muestra una pantalla (Figura 11.38) con las operaciones más frecuentes.

SABÍAS QUE

El *software* de programación Zelio Soft está disponible para su descarga gratuita desde la web del fabricante.

Figura 11.38. *Pantalla de inicio.*

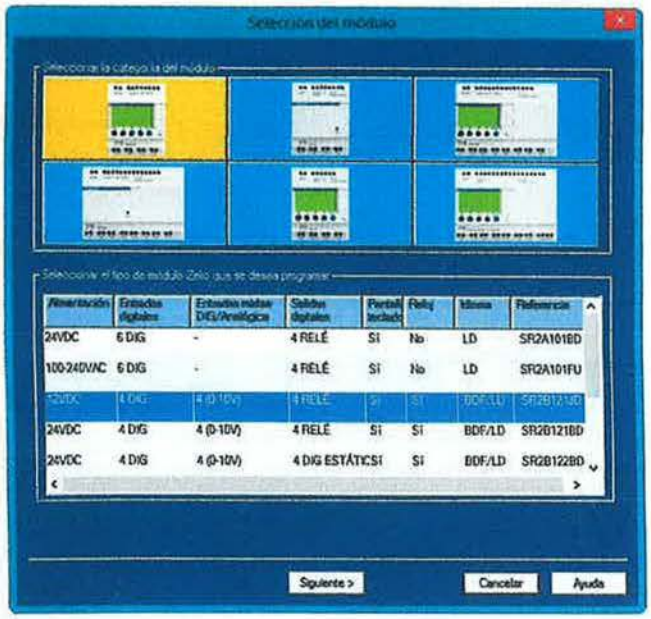


Figura 11.39. *Pantalla de selección de módulo.*

Al crear un programa nuevo, lo primero es seleccionar el módulo. Para ello, mediante un menú visual se muestran los diferentes módulos junto con sus características según versiones (Figura 11.39). En caso de ser un Zelio modular se pueden seleccionar los módulos externos.

La forma de realizar la programación del Zelio depende del modelo empleado. Hay modelos que admiten tanto la programación en lenguaje de contactos (Ladder) como en diagrama de bloques de funciones (BDF). Cuando se elige modelo y versión, aparece una columna llamada *Idioma*, en la que se muestra con cuál se puede llevar a cabo esta tarea. Si el modelo en cuestión admite ambas formas aparecerá a continuación una pantalla para su selección (Figura 11.40).

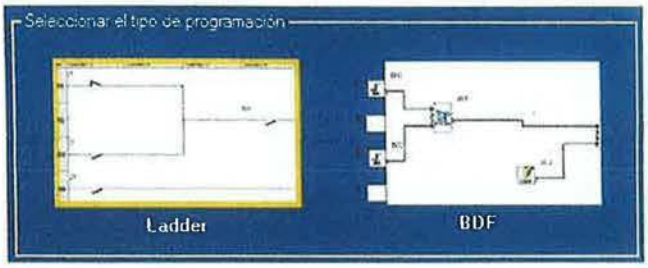


Figura 11.40. *Pantalla de selección del lenguaje de programación.*

La interfaz de usuario del *software* de programación de Zelio Soft consta de las siguientes partes:

- **Barra de menús.** Está situada en la parte superior. En ella se encuentran los comandos para la elaboración, configuración y transferencia de programas.

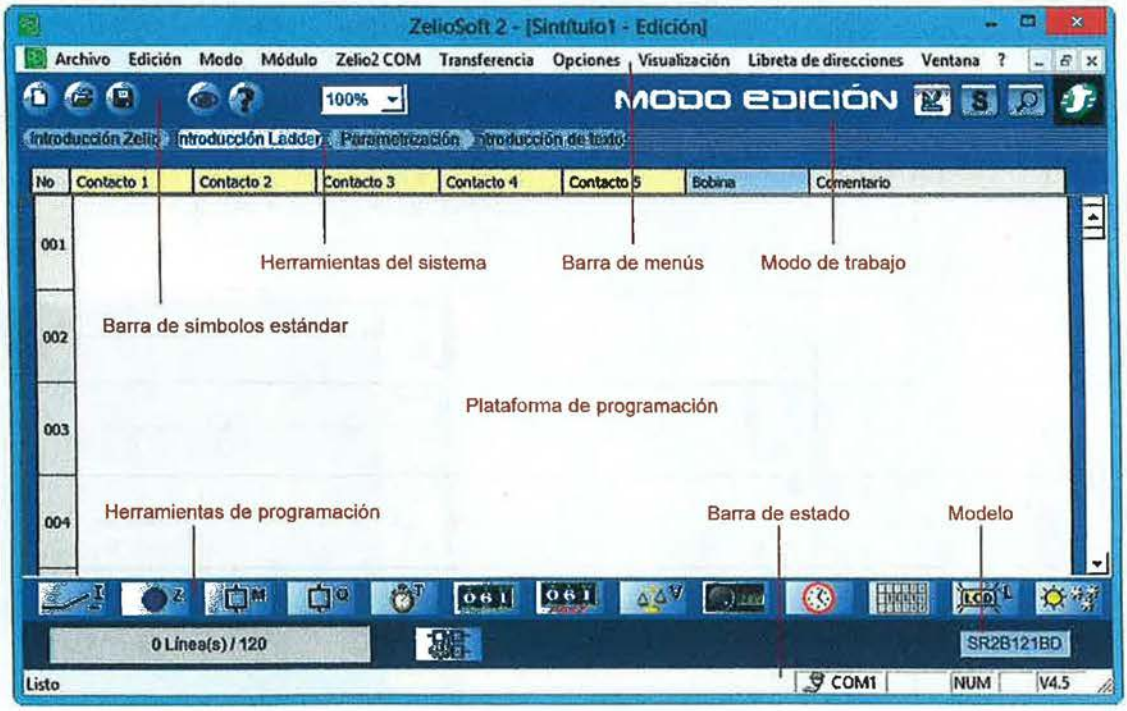


Figura 11.41. *Pantalla de programación.*

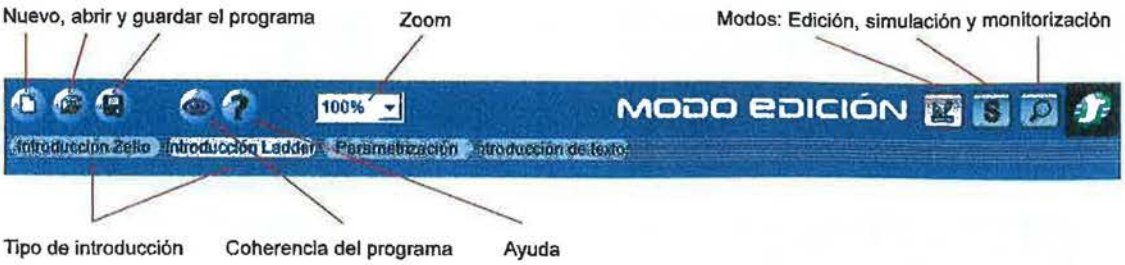


Figura 11.42. *Barra de símbolos estándar.*

- **Barra de símbolos estándar.** Contiene los botones de comandos de uso general, tales como: crear, cargar y guardar el programa.
- **Herramientas del sistema.** Por medio de ellos se indica el modo de introducción del programa.
- **Modos de trabajo.** El Zelio dispone de tres modos de trabajo: edición, simulación y monitorización.
- **Plataforma de programación.** Sobre esta superficie se desarrolla el programa. Está dividida en filas y columnas.
- **Herramientas de programación.** Contiene los elementos mediante los cuales se lleva a cabo la programación: entradas, salidas, temporizadores, etc.
- **Barra de estado.** Proporciona información adicional, tal como el nivel de zoom, la situación de un programa o la actividad.

SABÍAS QUE

El *software* de programación es capaz de detectar la coherencia del programa y si tiene algún error, tal como algún elemento sin conectar, etc., lo indicará mediante un icono en la barra de símbolos.

11.3.6. Las herramientas de programación

La barra de herramientas de programación es la más útil y empleada durante la fase de realización del programa. Está situada en la parte inferior y debajo de la plataforma de programación. Ella contiene los diversos recursos que se emplean. Esta barra se adapta al modelo de Zelio que se emplea, así no aparecerán aquellos recursos de los que el Zelio carezca, por ejemplo: cierto número de entradas o salidas, temporizadores rápidos, display, etc.

SABÍAS QUE

El tamaño máximo de un programa con el Zelio es de 120 líneas. En la barra de herramientas de la parte inferior puedes saber cuántas llevas utilizadas.

Las entradas

Las entradas digitales se utilizan exclusivamente como un contacto en el programa. Este contacto representa el estado de la entrada del módulo lógico conectado a un captador (botón pulsador, interruptor, detector, etc.).

Figura 11.43. *Herramientas de programación.*

El contacto puede ser abierto o cerrado. Para los contactos abiertos, su símbolo será una "I" con el número correspondiente al orden del módulo. Para el caso de los contactos cerrados, su símbolo será "i" con el número correspondiente.

Figura 11.44. *Entradas.*

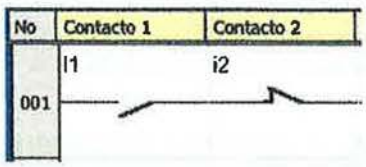


Figura 11.45. *Símbolo eléctrico.*

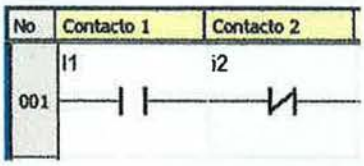


Figura 11.46. *Símbolo* ladder.

La visualización de los contactos se puede llevar a cabo de dos maneras: símbolo eléctrico (se asemeja a la simbología eléctrica empleada en lógica cableada) y símbolo *ladder* (es la más empleada por la mayoría de los autómatas). Para cambiar su modo de representación hay que ir al menú de visualización y ahí escoger la deseada.

Las teclas de navegación

Las teclas de navegación o cursores son aquellas que están situadas en el frontal del Zelio. Se dispone de cuatro teclas o botones y su comportamiento es idéntico a las entradas digitales.

Se identifican con la letra Z y si sus contactos son normalmente abiertos estos se nombran en mayúscula (Z) y si son normalmente cerrados se nombran en minúscula (z).

Figura 11.47. *Teclas de navegación.*

Las salidas

Las salidas digitales corresponden a los relés de salida del propio Zelio y sobre estas se conectan los diferentes actuadores que se empleen en la instalación.

Van etiquetados con la letra Q seguida del número de orden.

Estas salidas se pueden utilizar en la parte de la bobina o en la parte de los contactos. Si son contactos normalmente abiertos, estos se nombrarán en mayúscula (Q) y si son cerrados se nombrarán en minúscula (q).

Figura 11.48. *Salidas.*

Existen cuatro tipos de bobinas para las salidas:

- Tipo **conector**. Si reciben señal se activan y se desactivan cuando esta desaparece. Se comportan igual que un relé o contactor.
- Tipo **telerruptor**. Funcionan a impulsos. Con un impulso se activa y con otro se desactiva.
- Tipo **Set**. Con un impulso se activa la bobina y permanece en ese estado aunque desaparezca la señal.
- Tipo **Reset**. Con un impulso de desactiva la bobina. La bobina Reset tiene prioridad sobre la Set.

RECUERDA

Si se emplea una bobina de tipo Set, se debe emplear otra bobina tipo Reset para poder desactivarla.

Los relés auxiliares o marcas

Son idénticos a las salidas digitales pero no tienen una salida física del Zelio. Se identifican con la letra M. Constan de dos partes: la bobina (que también puede ser de tipo conector, telerruptor, Set o Reset) y los contactos.

Los temporizadores

Los temporizadores se encargan de realizar tareas de control respecto a la variable tiempo. Permiten retardar, prolongar y activar acciones con un tiempo configurable.

Se denominan por la letra T seguida del número de orden. El número de temporizadores depende del modelo. En un módulo con 16 temporizadores, estos van desde el T1 hasta el TG (el número 10 corresponde a la letra A, así el décimo será TA).

Figura 11.49. *Temporizadores.*

Constan de dos partes: la bobina y los contactos. La bobina se compone de dos entradas:

- **Activación.** Se identifica por la letra T. Permite activar el temporizador.
- **Reset.** Se identifica por la letra R. Cuando se aplica una señal a esta entrada el temporizador se reinicia.

Existen varios modos de funcionamiento junto con una serie de parámetros para configurar los temporizadores. Para ello, una vez colocado el temporizador sobre la plataforma de programación, o se hace doble *clic* o con el botón derecho del ratón se entra en la ventana *Parámetros* (Figura 11.50).

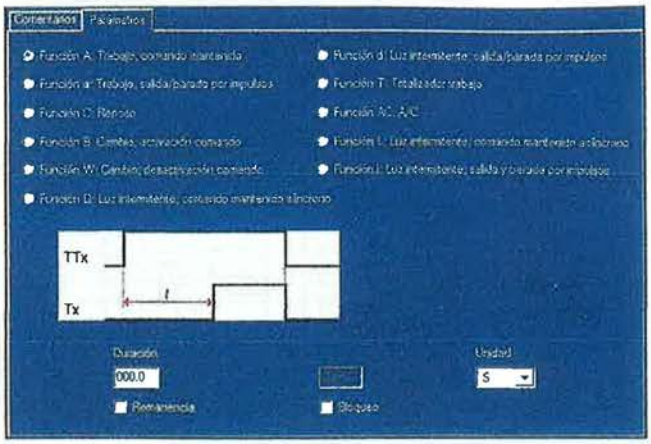


Figura 11.50. *Tipos de temporizadores.*

Los parámetros necesarios son:

- Tipo o función de temporizador. Indica cuál es el modo de funcionamiento.
- Duración.
- Unidad de medida o base de tiempos.
- Remanencia.
- Bloqueo.

El tiempo de funcionamiento depende de la duración marcada y de la unidad de medida.

La remanencia permite conservar el valor de tiempo transcurrido como protección ante un corte de corriente. Así, si está activada esta casilla, el tiempo transcurrido se memoriza y no se pierde.

El bloqueo, si está activado, impide que se puedan modificar los parámetros.

Tabla 11.8. Tipos de funciones de los temporizadores.

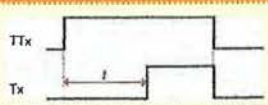
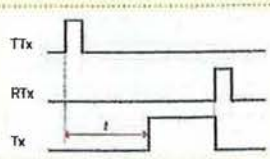
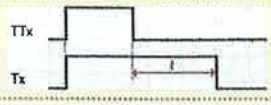
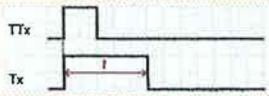
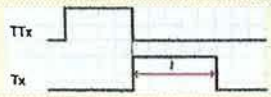
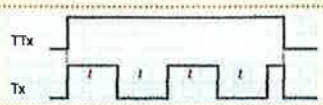
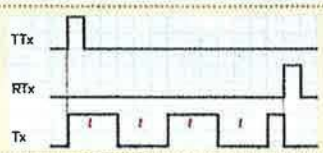
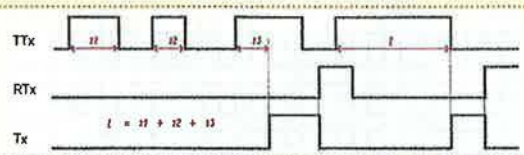
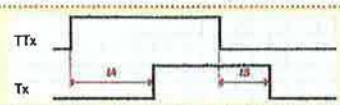
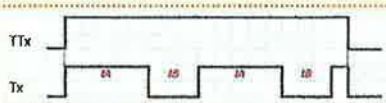
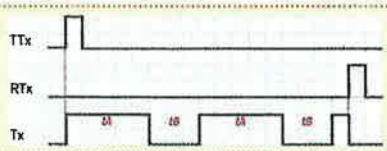
Función	Cronograma	Descripción
A		Trabajo, comando mantenido. Retardo a la conexión.
a		Trabajo, salida/parada por impulsos. Retardo a la conexión.

Tabla 11.8. Tipos de funciones de los temporizadores *(continuación)*.

Función	Cronograma	Descripción
C		Reposo. Retardo a la desconexión.
B		Cambio, activación comando. Activación temporizada.
W		Cambio, desactivación comando.
D		Luz intermitente, comando mantenido síncrono.
d		Luz intermitente, salida/parada por impulsos.
T		Totalizador trabajo.
AC		A/C. Temporización a la conexión y desconexión.
L		Luz intermitente, comando mantenido asíncrono.
I		Luz intermitente, salida y parada por impulsos.

Actividad resuelta 11.4

Se desea, que al accionar un pulsador (I1) se encienda una lámpara (conectada a la salida Q1) durante 5 segundos.

La función del temporizador a emplear es la función B, con una duración de 5 segundos.

Solución:

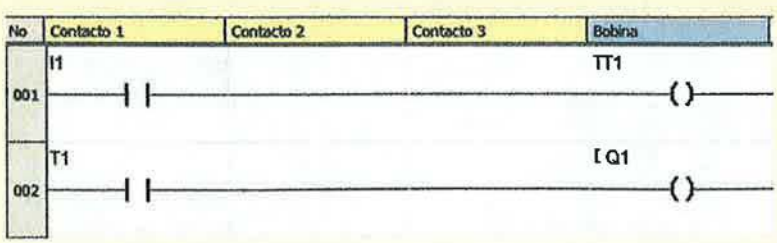


Figura 11.51. *Ejemplo de empleo de temporizadores.*

Contadores

La función de los contadores es la de contar los impulsos que le llegan a una de las entradas.

El modo en el cual realiza la cuenta puede ser de modo ascendente (cuenta progresiva) o descendente (cuenta regresiva).

Los contadores se identifican por la letra C seguida del número de orden.

Los contadores disponen de dos partes: los contactos y la parte de la bobina.

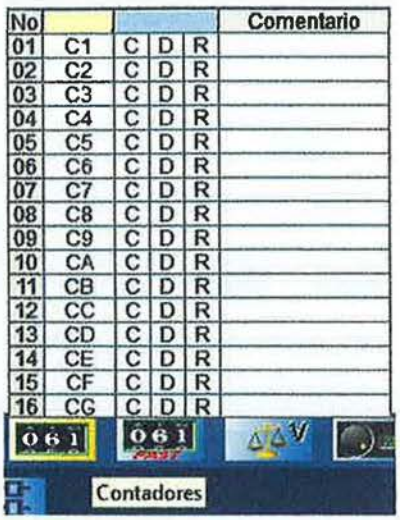


Figura 11.52. *Contadores.*

La parte de la bobina dispone de tres entradas:

- **Entrada de impulsos a contar (C).**
- **Modo de cuenta (D).** Si a la entrada tiene un "0" la cuenta se realiza en modo progresivo y si tiene un "1" la cuenta se realiza en modo regresivo. Si no se especifica nada, es decir si esa entrada no se emplea, el Zelio por defecto realiza la cuenta de modo progresivo o ascendente.
- **Reset (R).** Al activar esa entrada el contador vuelve a su estado inicial.

Mediante el contacto asociado al contador es posible:

- En cuenta progresiva o ascendente, conocer cuándo la cuenta ha llegado al valor determinado (es configurable).
- En cuenta regresiva o descendente, conocer cuándo la cuenta ha llegado a cero desde un valor determinado.

La pantalla de configuración de los parámetros contiene:

- **Impulsos.** Es el valor de preselección.
- **Salida ON.** Según sea cuenta progresiva o regresiva, es decir que se llegue al valor de preselección o al valor cero.
- **Remanencia.** Memoriza la cuenta en caso de desconexión eléctrica.
- **Bloqueo.** Impide la modificación.

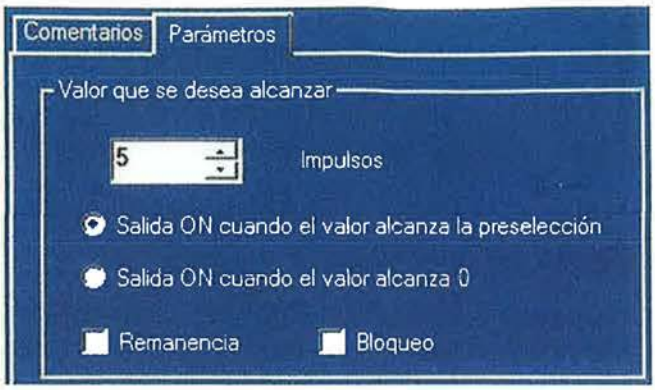


Figura 11.53. *Configuración de los contadores.*

Según el modelo de Zelio, este puede disponer de dos tipos de contadores: contadores y contadores rápidos. Los contadores rápidos tienen una mayor capacidad de poder realizar cuenta de entrada de pulsos de alta frecuencia (hasta 1 kHz). Este tipo de contador de define por la letra K.

Actividad resuelta 11.5

Se desea, que al accionar cinco veces un pulsador (I1) se active una salida (Q1). Se debe poder realizar la cuenta tanto en modo ascendente como en modo descendente (I2) y en cualquier momento se puede reiniciar el contador (I3).

Solución:

La configuración de parámetros es:

- Número de impulsos = 5.
- Salida On: al alcanzar la preselección.

Al contador C1 se le conectan las tres entradas:

- I1: entrada de impulsos a contar.
- I2: modo de cuenta, si es abierto la cuenta se realiza de modo ascendente hasta la preselección. Si es cerrado la cuenta se realiza en modo descendente.
- I3: al activarse inicializa el contador poniéndolo a cero.

Con el contacto de C1 al llegar a la cuenta determinada se cerrará activando la salida Q1.

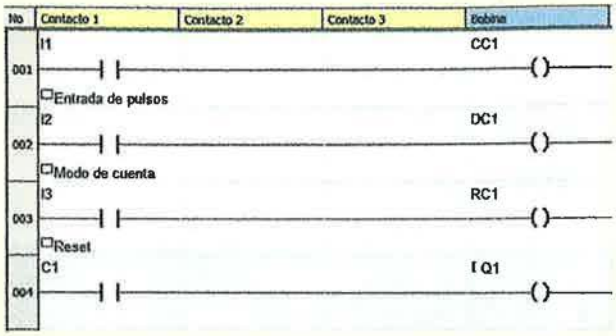


Figura 11.54. *Ejemplo de empleo de contadores.*

Los comparadores de contadores

La función de comparador de contadores es la de permitir comparar la cuenta de un contador con otro contador o con una constante.

Se denominan por la letra V seguida del número de orden.

El comparador de contadores se emplea como un contacto y al cumplir con las condiciones de comparación prefijadas se activa. Este contacto puede ser normalmente abierto (V) o normalmente cerrado (v).

Figura 11.55. *Comparadores.*

Figura 11.56. *Configuración del comparador.*

En la pantalla de parámetros se fijan los dos parámetros a comparar: Cx y Cy. Estos pueden hacer referencia a contadores o a una constante. Además, se puede añadir un *offset* o desplazamiento al contador a comparar o bien se fija un valor si es una constante.

También se debe fijar el criterio de comparación: mayor, mayor o igual, igual, distinto, menor o igual y menor.

En el ejemplo de la Figura 11.56, se parametriza el comparador de tal manera que el contacto asociado a este comparador se activará cuando la cuenta del contador C1 sea igual o superior a 5.

Se pueden combinar varios comparadores de contadores en una misma línea de instrucciones (Figura 11.57).

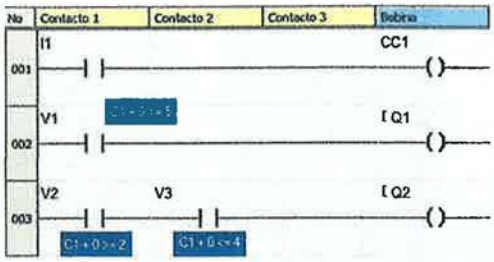


Figura 11.57. *Ejemplo de empleo de contadores.*

Existe una función similar para trabajar con datos analógicos, el **comparador analógico**. Se identifica con la letra A seguida del número de orden.

El reloj

El reloj o programador horario se emplea para realizar acciones en función de la hora y del día de la semana. Se emplean como contacto. Disponen de cuatro canales que se pueden programar de manera independiente. En cada canal se marca el día y la hora de conexión (*on*) y desconexión (*off*).

En la parte inferior del configurador de los parámetros del reloj, se muestra una vista general donde aparecen todos los días de la semana junto con las horas diarias. Por medio de unas franjas de color se visualizan los rangos de trabajo.

Figura 11.58. *Reloj.*

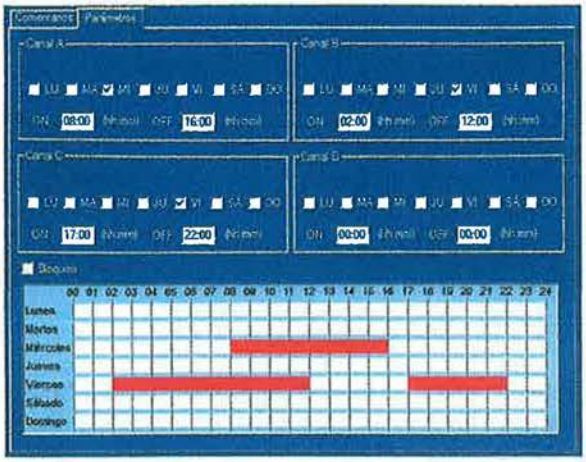


Figura 11.59. *Configuración del reloj.*

Bloques de texto

La función de bloques de textos solo está disponible en los módulos que disponen de display. Permite mostrar textos y valores sobre la pantalla.

Los bloques de texto se emplean como bobinas. Se identifican por las letras TX.

En cada bloque de texto se pueden mostrar hasta cuatro líneas de información o datos. Y se pueden disponer hasta 16 bloques de texto, de TX1 a TXG.

Figura 11.60. *Textos.*

Retroiluminación LCD

Esta función de retroiluminación de la pantalla LCD solo está disponible en los modelos con display. Permite asegurar la iluminación.

Se representa por las letras TL y solo se necesita una función.

Figura 11.61. *Retroiluminación.*

Cambio de horario verano/invierno

El cambio de horario se emplea como contacto y sólo está disponible una única función.

En la franja horaria de invierno está en estado de paro, pasando a estado de marcha durante el verano.

Figura 11.62. *Verano/invierno.*

11.3.7. La realización de un programa

Para comparar la forma de realizar un programa con el lenguaje de diagrama de bloques de funciones respecto al lenguaje de contactos se va a volver a resolver el mismo ejemplo, tal y como se muestra en la Figura 11.63.

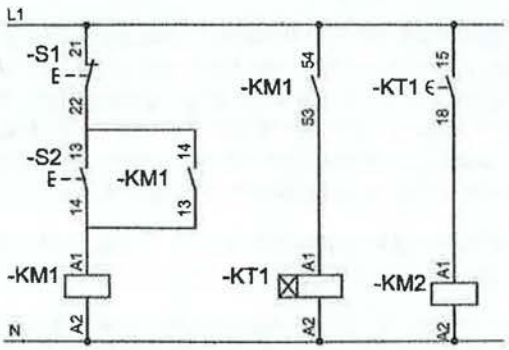


Figura 11.63. *Esquema eléctrico.*

El circuito de este ejemplo consiste en el control de un contactor KM1 que es gobernado mediante una botonera marcha-paro. Este contactor activa un temporizador a la conexión, el cual transcurrido el tiempo prefijado, activa un segundo contactor (KM2).

Tabla 11.9. Lista de referencias.

Elemento	Desig. Zelio	Descripción
S1	I1	Pulsador de paro
S2	I2	Pulsador de marcha
KM1	Q1	Motor 1
KM2	Q2	Motor 2
KT1	T1	Temporizador

Con los datos de los elementos de entradas y salidas se elabora la tabla de referencias donde se asigna cada elemento a las entradas y salidas físicas del Zelio (Tabla 11.9).

Con estos datos, se obtienen las funciones lógicas para cada receptor:

$$Q1 = (\overline{I1}) \times (I2 + Q1)$$

$$T1 = Q1$$

$$Q2 = T1$$

Ahora ya se puede empezar a realizar el programa, para ello arrancamos el *software* Zelio Soft. Comenzamos con un nuevo programa y se elige el módulo de Zelio que cumpla con los requisitos mínimos, que en este caso se va a necesitar que tenga más de dos entradas digitales y más de dos salidas digitales. Se elige, por ejemplo, el módulo SR2B121FU (ocho entradas y cuatro salidas, con display, alimentación a 230 V, sin expansión). Y se escoge la programación en lenguaje *ladder*.

Empezamos con las entradas. Se accede a ellas y se comentan. Los comentarios son importantes puesto que ayudan a comprender y mantener el programa. Para ello nos situamos encima de las entradas digitales y se desplegará una ventana, a continuación nos situamos sobre la zona de los comentarios de cada entrada y se escribe.

Con las salidas se procede igual y se añaden los comentarios para poderlas identificar correctamente.

Para insertar la entrada, nos situados sobre ella en la parte del contacto (en el menú desplegable de las entradas aparecen tres columnas: el número de orden, el contacto de la entrada y el comentario) y la marcamos y arrastramos con el ratón, soltándolas sobre la casilla de destino (Figura 11.64).

Al poner cualquier contacto sobre la plataforma de programación siempre la coloca en formato de normalmente abierto y si se desea un contacto normalmente cerrado se pincha con el botón derecho del ratón y se cambia.

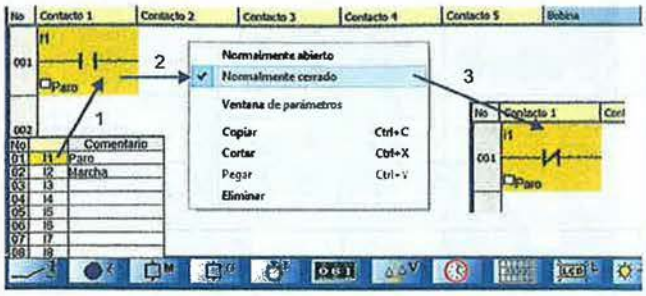


Figura 11.64. *Colocación y configuración de las entradas.*

Para las salidas se procede de similar manera. Se ponen los comentarios pero para la parte de la bobina hay que tener en cuenta que tiene cuatro formatos: **conector, telerruptor, Set y Reset**. Para este caso se desea que sea de tipo conector. Se marca y se arrastra hasta la columna de las bobinas (Figura 11.65).

RECUERDA

Si se intenta poner un contacto en la zona de la bobina y viceversa, el programa no lo permitirá, apareciendo el símbolo de prohibido.

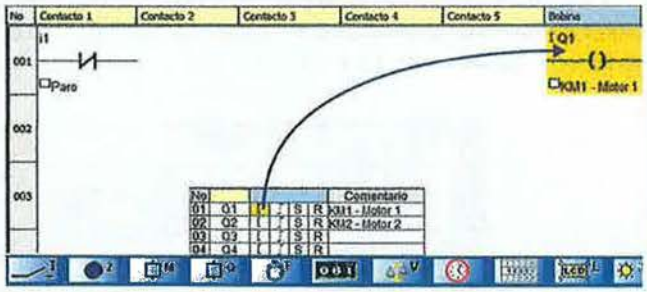


Figura 11.65. *Colocación de las salidas.*

Para realizar el conexionado eléctrico, simplemente se hace clic con el ratón sobre la línea discontinua (Figura 11.66, círculos rojos).

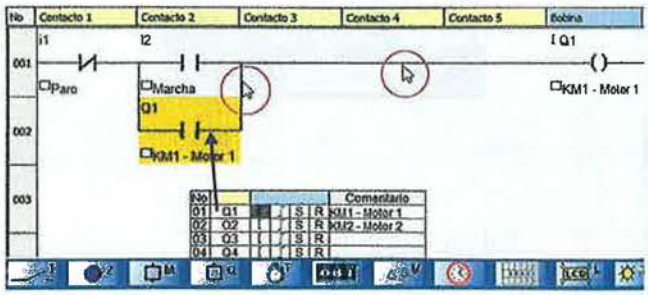


Figura 11.66. *Colocación de las conexiones eléctricas y del contacto de una bobina.*

Las salidas tienen dos partes: contacto y bobina. Para realizar la realimentación de Q1 se emplean ambas partes. El contacto de la salida (Q1) se coloca en paralelo con el pulsador normalmente abierto (Figura 11.66).

Para colocar el temporizador se procede de igual manera. Como es un temporizador a la conexión, el tipo a emplear es el de *función A* con un tiempo de 3 segundos. Este temporizador activará al segundo contactor, por ello se emplea un contacto abierto de este.

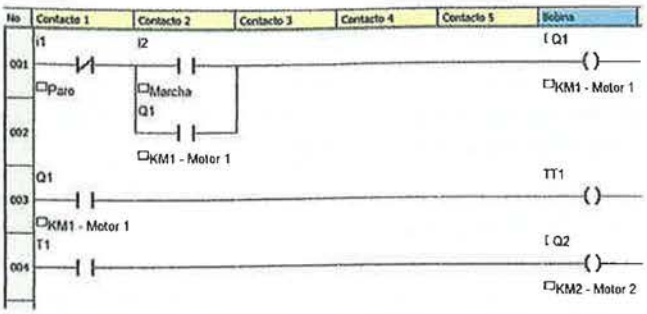


Figura 11.67. *Programa.*

Actividad resuelta 11.6

Un sistema de extracción y renovación de aire compuesto por un ventilador se pone en marcha de forma manual mediante un interruptor (I1) o bien de forma automática cuando un sensor (I2) detecte la mala calidad del aire y una ventana de la sala no esté abierta (final de carrera I3). Resuélvelo mediante bobinas tipo Conector.

Solución:

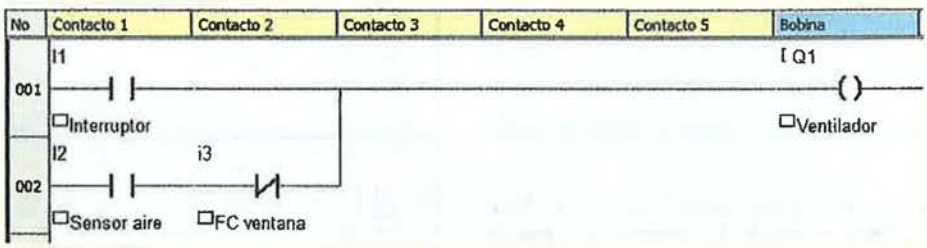


Figura 11.68. *Programa.*

Actividad resuelta 11.7

Se desea controlar un motor mediante un pulsador de marcha y uno de paro. Resuélvelo mediante bobinas de tipo Set y Reset.

Solución:

En este caso tanto el pulsador de marcha como el de paro son de contacto normalmente abierto. Mediante el pulsador de marcha se enclava la salida Q1 y con el de paro se desenclava. En caso de estar ambos pulsadores activados tiene prioridad el de paro sobre el de marcha.

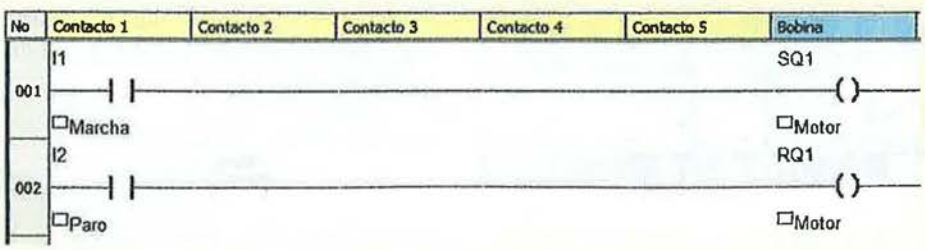


Figura 11.69. *Programa.*

11.3.8. La simulación

Una vez terminado el programa, es la hora de simular su funcionamiento para poder comprobar que responde correctamente según las necesidades.

El Zelio Soft cuenta con los modos de edición, simulación y monitorización.

Figura 11.70. *Modo edición.*

Figura 11.71. *Modo simulación.*

Para acceder a la simulación, se debe cambiar de modo de trabajo.

El modo de simulación posee cuatro acciones: **Run** (puesta en marcha), **Stop** (paro de la simulación), **Pausa** (realiza una pausa y permite continuar con la simulación) y **Corte de alimentación** (se simula un corte en el suministro eléctrico al autómata).

Cada vez que se accede al modo de simulación, se parte de la situación de *Parada* o *Stop*. Por ello lo primero es poner en marcha la simulación pulsando sobre el icono *Run* (Figura 11.71).

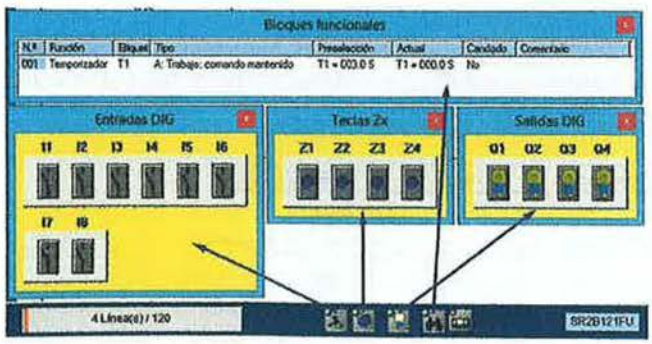


Figura 11.72. *Herramientas de supervisión.*

El modo de simulación cuenta con unas ventanas para poder mostrar y manipular los siguientes elementos: entradas, salidas y teclas de navegación. También cuenta con un panel para mostrar cómo están funcionando los diversos bloques de funciones (temporizadores, contadores, etc.).

Una vez arrancada la simulación, el cableado eléctrico situado sobre el panel de operación cambia de color. Aquellas partes sometidas a tensión aparecen en color rojo y en ausencia de tensión aparecen en color azul. Además, en los paneles de visualización de las entradas y salidas, estas cambian de color en función de su estado. En el panel de las entradas, estas se pueden manipular para interactuar en la simulación. Así, en la línea 001 del programa al accionar I2 (marcha) se activa la bobina del contactor KM1 que está en la salida Q1, por eso está coloreada en rojo. Sin embargo, la salida Q2 está en ese momento desactivada (línea 004) y por ello aparece coloreada de azul.

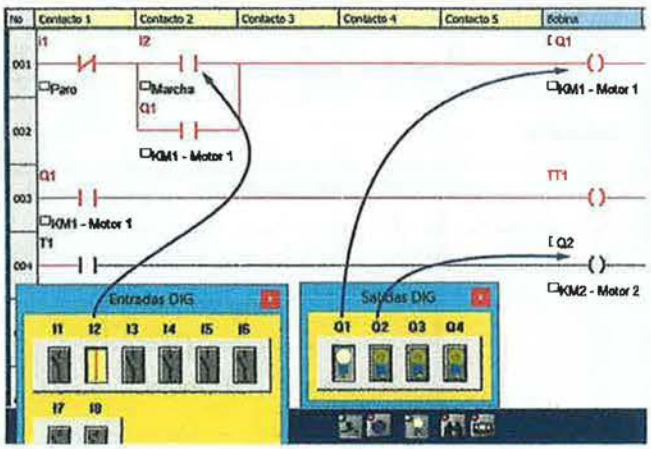


Figura 11.73. *Simulación del programa.*

En caso de tener que modificar alguna parte del programa, se puede pasar al modo de edición en cualquier momento.

11.3.9. La transferencia del programa

Una vez terminado y verificado el programa con el simulador, es el momento de realizar la transferencia al Zelio. Lo primero es conectar el cable de transferencia entre el ordenador y el propio Zelio. El conector en el autómata se encuentra en la parte frontal protegido por una tapa, que habrá de retirar previamente.

Figura 11.74. *Cable USB de transferencia.*

Para comenzar con la orden de transferencia, se debe volver al modo de edición, para así tener accesible el menú *Transferencia*. Desde allí se accede a *Transferir programa* y *PC > Módulo*.

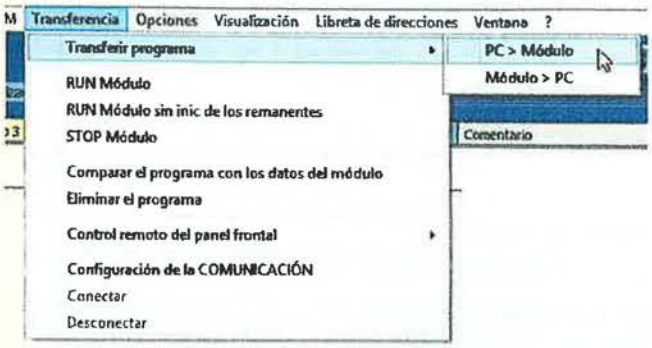


Figura 11.75. *Menú de transferencia.*

En el caso de realizar la programación para un módulo y llegado el momento de realizar la transferencia, se comprueba que el módulo a emplear es otro y se puede realizar el cambio.

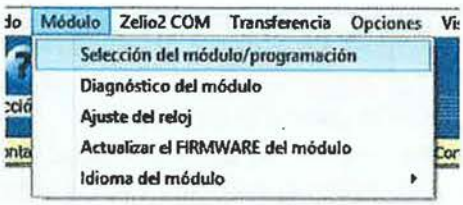


Figura 11.76. *Menú de selección de módulo.*

Si durante el cambio hay alguna incompatibilidad, se mostrará un mensaje. Por ejemplo, si se pasa de un modelo de 8 entradas a uno de 6 entradas, se avisará de la pérdida de esas 2 entradas de diferencia.

SABÍAS QUE

A la hora de realizar la transferencia del programa del ordenador hacia el autómata, es importante fijarse en el tipo de puerto de salida del ordenador. Los puertos de comunicaciones pueden ser del tipo serie (COM) que actualmente ya están obsoletos o del tipo USB. Existen cables de comunicaciones para ambos tipos de puertos.

11.3.10. El modo de monitorización

El modo de monitorización permite una comunicación entre el Zelio y el ordenador, mientras el Zelio está ejecutando un programa. De esta manera se permite su control en tiempo real, manipulando las entradas.

Para poder acceder a este modo de monitorización, el ordenador con el ZelioSoft debe estar ejecutándose y además ambos dispositivos deben estar conectados mediante el cable de programación.

Otro requisito es que tanto el Zelio físico como el módulo del *software* sean idénticos. En caso de que sean diferentes se debe cambiar y adaptar el módulo del *software*.

Una vez comunicados y en ejecución, es posible acceder a las entradas para poder forzarlas del mismo modo que se interactuaba en la fase de simulación.

Actividades de comprobación

11.1. Un relé programable es:

a) Un autómata de menores prestaciones.

b) Un autómata de altas prestaciones el cual lleva incluida su propia fuente de alimentación y un conjunto de entradas y salidas digitales.

c) Un conjunto de relés bajo una misma carcasa, cada uno de ellos con su propia bobina.

11.2. ¿En qué lenguaje se puede programar el Logo?

a) En diagrama de bloques de funciones.

b) En lenguaje de contactos.

c) En diagrama de bloques de funciones y lenguaje de contactos.

11.3. El Logo es un relé programable que:

a) Solo permite la ampliación de entradas y salidas.

b) Solo permite la ampliación de aquellas partes de las cuales no consta como por ejemplo un módulo de comunicación.

c) Permite la ampliación de varias partes incluso de aquellas de las que ya posee.

11.4. La alimentación eléctrica del Logo es de:

a) 230 V en corriente alterna.

b) 24 V en corriente continua.

c) Existen Logos de varios niveles de tensión.

11.5. Si un Logo tiene una alimentación de 230 V_{CA}, las entradas:

a) Deben ser de 230 V y de la misma fase cada grupo de entradas.

b) Deben ser de 230 V y no importa la fase, solo el nivel de tensión.

c) Se pueden utilizar entradas de 24 V por ser una tensión inferior.

11.6. ¿Cuál de los siguientes bloques de funciones representa a un contador?

			
a)	b)	c)	d)

11.7. Si un relé programable Zelio tiene una alimentación a 24 V_{CC} y sus entradas son sensores también a la misma tensión, en sus salidas a relé, se podrán conectar:

a) Receptores de la misma tensión a 24 V_{CC}.

b) Receptores a una tensión de 230 V_{CA}.

c) Es independiente y por tanto se podrá conectar cualquier tipo de receptor.

11.8. Los temporizadores para el Zelio:

a) Hay tantos temporizadores como modos de funcionamiento.

b) Hay solo dos temporizadores: a la conexión y a la desconexión. Y se configuran para responder a niveles o por flancos.

c) Solo hay un temporizador y se configura para adaptar su modo de funcionamiento.

11.9. Un contador para el Zelio, se compone de las siguientes entradas:

a) Una entrada para los pulsos a contar más otra para indicarle el sentido de las cuentas.

b) Una entrada para las cuentas progresivas y otra entrada para las cuentas regresivas.

c) Solo una entrada para las cuentas. Hay un contador de modo progresivo y otro contador para el modo regresivo.

11.10. El modo de monitorización del Zelio se emplea para:

a) Realizar el programa.

b) Realizar la transferencia del programa desde el ordenador al propio Zelio.

c) Manipular y supervisar al Zelio cuando está en ejecución del programa.

11.11. Se necesita automatizar un proceso industrial el cual cuenta con los siguientes elementos: un pulsador de marcha y uno de paro, un motor trifásico que gira en ambos sentidos protegido mediante relé térmico, un final de carrera para cada sentido de giro, un piloto de señalización para cada sentido de giro y otro para el disparo por sobrecarga del motor. Con estos datos, selecciona un relé programable el cual disponga de, al menos, las siguientes características:

a) 6 entradas y 4 salidas.

b) 5 entradas y 4 salidas.

c) 5 entradas y 5 salidas.

ACTIVIDADES FINALES

Actividades de aplicación

11.1. Explica en qué casos es preferible el empleo de un relé programable frente a un autómata de alto nivel.

11.2. Comenta los diferentes tipos de Logos que hay comercialmente.

11.3. Indica las precauciones que se deben adoptar a la hora de realizar las conexiones de la alimentación eléctrica al Logo.

11.4. Indica las precauciones que se deben adoptar a la hora de realizar las conexiones respecto a las entradas en el Logo.

11.5. Indica las precauciones que se deben adoptar a la hora de realizar las conexiones respecto a las salidas en el Logo.

11.6. Describe brevemente de qué están compuestas las funciones básicas del LogoSoft.

11.7. Describe brevemente de qué están compuestas las funciones especiales del LogoSoft.

11.8. Comenta los diferentes tipos de Zelio que hay comercialmente.

11.9. Describe los modos de trabajo con los que cuenta el ZelioSoft.

11.10. ¿Qué función realizan las marcas?

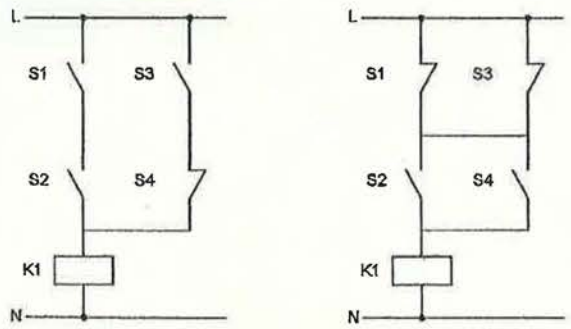
11.11. Describe el comportamiento que tienen las salidas en el ZelioSoft.

11.12. En el ZelioSoft existen dos temporizadores a la conexión o trabajo, la función A (mayúscula) y la función a (minúscula). ¿Qué diferencia hay entre ambas?

11.13. ¿Qué significa cada entrada de un contador?

Casos prácticos

11.1. Realiza mediante bloques funcionales el programa para los circuitos dados:



11.2. Realiza el automatismo para un arranque de un motor en el cual se emplee la realimentación. Resuélvelo mediante bloques funcionales.

11.3. Realiza el automatismo para el control de dos motores de tal manera que cumpla lo siguiente:

- Con un pulsador (I1) se pondrá en marcha un motor 1 (Q1).
- Con otro pulsador (I2) se pondrá en marcha el motor 2 (Q2) siempre que el M1 esté en marcha previamente.
- Con un pulsador (I4) se podrá parar el motor 2 (Q2).
- Con otro pulsador (I3) se podrá parar el motor 1 (Q1).

Resuélvelo mediante:

a) Bloques funcionales.

b) Lenguaje de contactos.

11.4. Un equipo de extracción de aire está compuesto por dos motores y tres sensores de detección de la calidad del aire (I1, I2, I3). Cuando se activa alguno de esos sensores, pone en marcha el primer motor (Q1), y cuando al menos dos de los sensores se activan ponen en marcha el segundo motor (Q2). Resuélvelo por:

a) Diagrama de bloques funcionales.

b) Lenguaje de contactos.

11.5. Un sistema de alarma está compuesto por:

- Un interruptor (I1) para activar la alarma.
- Dos sensores (I2, I3) de detección de aperturas de puertas/ventanas.
- Un sensor de movimiento (I4) en el interior de la casa.
- Una señalización luminosa (Q1).
- Una señalización acústica (sirena) (Q2).
- Cuando se detecte (estando activada la alarma) una apertura de puertas o ventanas, se activará la señalización luminosa.
- Cuando además de la apertura se detecte movimiento se activará la sirena.

Resuélvelo mediante:

a) Bloques funcionales.

b) Lenguaje de contactos.

11.6. Se desea automatizar una puerta corredera de un garaje. Se cuenta con un pulsador de apertura (I1) y dos finales de carrera para puerta cerrada (I2) y puerta abierta (I3). Una vez la puerta está abierta, se cierra automáticamente. Si se está cerrando la puerta y se pulsa su apertura, tendrá prioridad la apertura.

11.7. Modifica el ejercicio anterior de la puerta del garaje para que:

- Una vez la puerta está abierta, permanezca así durante 5 segundos antes de empezar a cerrarse.
- Se añada una fotocélula (I4) para abrir la puerta en caso de detectar un obstáculo.