

10

Programación de autómatas

vamos a conocer...

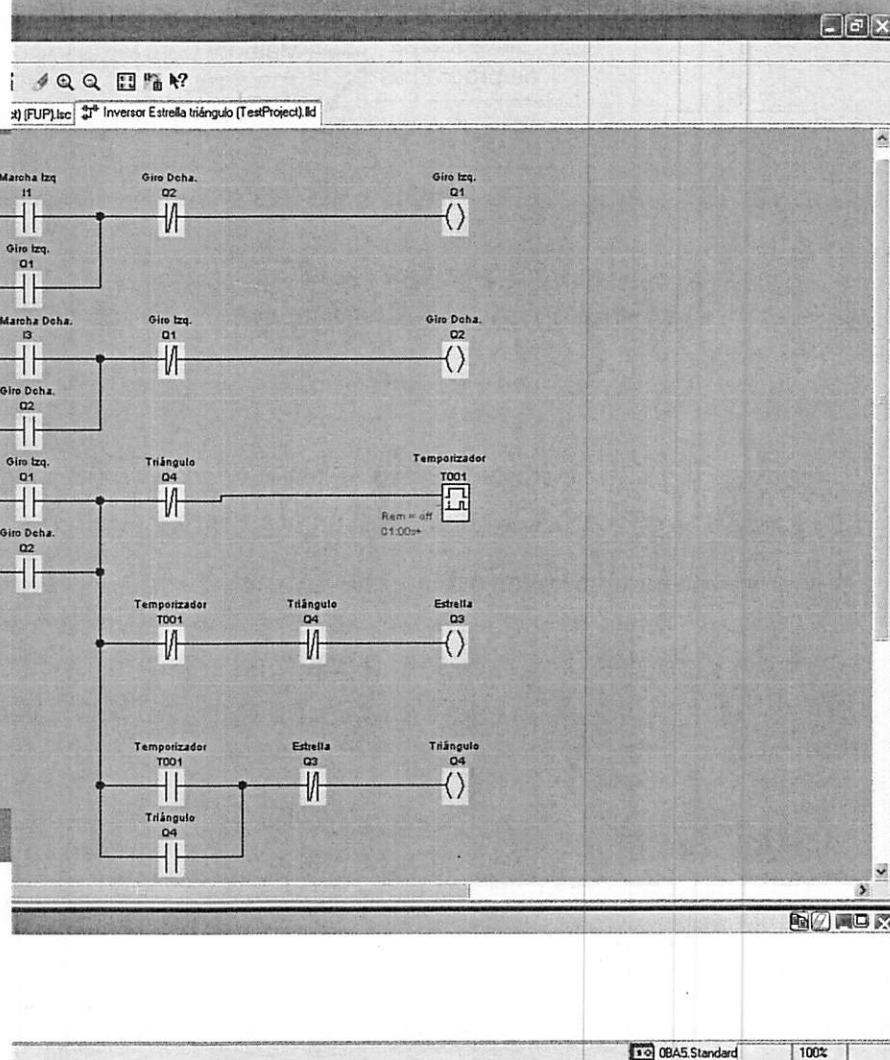
1. Zonas de memoria del autómata programable
2. Lenguajes de programación
3. Representación en lenguaje de contactos
4. GRAFCET

PRÁCTICA PROFESIONAL

Automatización de un taladro

MUNDO TÉCNICO

Programación gráfica del GRAFCET



y al finalizar...

- Conocerás los diferentes lenguajes de programación que utilizan los autómatas programables.
- Direccionarás adecuadamente las variables de un autómata programable a las diferentes zonas de memoria.
- Sabrás programar de forma básica en lenguaje de contactos.
- Implementarás circuitos secuenciales mediante GRAFCET en autómatas programables.

CASO PRÁCTICO INICIAL

situación de partida

Los técnicos de mantenimiento de una cadena de mecanizado de piezas están pensando adaptar las máquinas antiguas de taladrado, a los nuevos sistemas de fabricación. La maquinaria antigua dispone de circuitos de automatismos eléctricos cableados, que no son demasiado flexibles para su integración en los nuevos procesos.

Después de realizar un estudio de los diferentes sistemas automáticos, que se pueden aplicar a las nuevas tareas de automa-

tización de máquina-herramienta, parece claro que el uso de autómatas programables es la mejor opción.

Sin embargo, y viendo la diversidad de lenguajes de programación que disponen algunos PLCs, se debe tomar la decisión de utilizar el lenguaje que mejor se adapte al perfil electromecánico de los técnicos de mantenimiento de empresa y que permita desarrollar procesos secuenciales con facilidad.

estudio del caso

Antes de empezar a leer esta unidad de trabajo, puedes contestar las tres primeras preguntas. Después, analiza cada punto del tema, con el objetivo de contestar al resto de las preguntas de este caso práctico.

1. Seguramente que alguna vez has oído hablar algo relacionado con lenguajes de programación ¿Qué entiendes por este concepto?
2. ¿Qué necesitarían los técnicos de la empresa nombrada para transferir los programas a los autómatas programables?
3. Busca en internet algún documento relacionado con los lenguajes de programación para autómatas y nombra los que has localizado.
4. Si los técnicos de mantenimiento de la empresa de mecanizado tienen conocimientos de automatismos eléctricos cableados ¿Cuál de los lenguajes de programación crees que se adapta mejor a su perfil profesional? ¿Por qué?
5. Las máquinas de taladrado que se van a automatizar, disponen de un funcionamiento secuencial ¿crees que es adecuado el GRAFCET como método para dar solución a los problemas de automatización planteados?
6. Si se utiliza el GRAFCET ¿Qué elementos gráficos habrá que utilizar para implementar paradas de emergencia y rearmes?
7. ¿Qué parte del GRAFCET se encarga de activar las salidas físicas del autómata para gestionar el funcionamiento de los motores de los taladros?

1. Zonas de memoria de un autómata programable

Los autómatas programables almacenan los datos que procesan en diferentes zonas de memoria. El desarrollo de programas requiere conocer adecuadamente el acceso a estas zonas, tanto para escribir como para leer datos.

Las más comunes en la mayoría de los autómatas programables son:

Zonas de memoria	Identificador	Procesan:
Entradas	I	Entradas físicas
Salidas	Q	Salidas físicas
Marcas	M	Variables de procesamiento interno
Temporizadores	T	Valores de tiempo
Contadores	C	Valores de cómputo
Variables de sistema	SM	Operaciones propias del sistema

saber más

A los módulos E/S de un autómata programable también se les denomina **canales**.

- **Entradas (I).** Detectan el estado de los captadores conectados a los bornes de las entradas del autómata.
- **Salidas (Q).** Se encarga de activar los actuadores y preactuadores desde el módulo de salidas del autómata.
- **Marcas (M).** También llamadas memorias o bits internos.

Tienen un comportamiento similar al de los relés auxiliares en la lógica cableada.

- **Temporizadores (T).** Los temporizadores generan eventos cuando alcanzan un valor de tiempo predeterminado.

Su comportamiento es similar a los utilizados en la lógica cableada. Existen temporizadores a la conexión y a la desconexión.

- **Contadores (C).** Los contadores originan eventos cuando alcanzan un determinado número de sucesos.
- **Variables o bits de sistema (SM).** También denominadas marcas de sistema, se encargan de realizar tareas prefijadas por el fabricante en el sistema operativo del autómata.

Cada modelo tiene sus propios bits de sistema y para conocerlos se hace necesario consultar el manual de usuario.

Direccionamiento

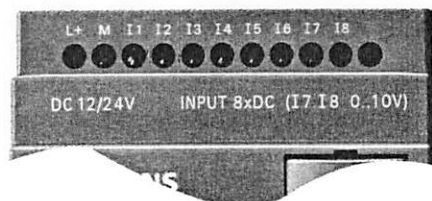
Para que un autómata programable pueda procesar datos de las zonas de memoria, el técnico programador debe hacer referencia a dichas zonas adecuadamente en el momento de la elaboración del programa.

El direccionamiento es la forma de «apuntar» a una variable en el programa de usuario y puede ser diferente según el modelo o fabricante del autómata programable. En esta unidad se utiliza una forma genérica de direccionamiento, que debes adaptar al dispositivo que tú emplees en el aula-taller.

El direccionamiento a una variable del autómata, se realiza escribiendo el **operando** en formato textual. En él se representa, con letras, la zona de memoria a la que se tiene acceso y, con números, el orden de la variable en dicha zona de memoria.

Varios ejemplos de direccionamiento son:

Operando		Operando	
I 10	Entrada número 10	M 20	Marca interna número 20
I 4	Entrada número 4	T 10	Temporizador número 10
Q 1	Salida número 1	T 25	Temporizador número 25
Q 7	Salida número 7	C 4	Temporizador número 4



↑ Figura 10.1. Ejemplo de direccionamiento de entradas (Zona I).



↑ Figura 10.2. Ejemplo de direccionamiento de salidas (Zona Q).

2. Lenguajes de programación

El lenguaje de programación es el encargado de manejar el juego de instrucciones del autómata para realizar las funciones lógicas y de cálculo de la CPU.

El programa de usuario se procesa en la memoria del autómata y se introduce a través de la consola o terminal de programación.

Para realizar la programación de forma adecuada, el técnico debe conocer con detalle las diferentes zonas de memoria del autómata programable y su direccionamiento.

Según la norma UNE-EN 61131-3 (que concuerda con la IEC 1131-3) cuatro pueden ser los lenguajes de programación para autómatas. Dos en formato texto y otros dos en formato gráfico, pudiendo ser combinables y complementarios entre sí.

saber más

El direccionamiento de entradas/salidas pueden ser diferentes según el tipo y la firma comercial del autómata.

Por tanto, siempre que se instale un nuevo equipo, es necesario consultar los manuales de usuario.

vocabulario

Bit

Es la unidad básica de información, 8 bits forma un **Byte**, 2 bytes (16bits) forman una palabra (**Word**). La palabra de 32 bits se denomina doble palabra (**DW**).

vocabulario

Nemotecnia

Es un conjunto de símbolos que generan una memoria artificial.

caso práctico inicial

El lenguaje LD es el que más similitudes tiene con los esquemas eléctricos de automatismos industriales.

Así, este lenguaje es el que mejor se adapta a los técnicos de mantenimiento del caso práctico.

2.1. Lenguajes gráficos

Permiten la elaboración de programas de forma gráfica, dibujando los esquemas mediante el dispositivo de programación.

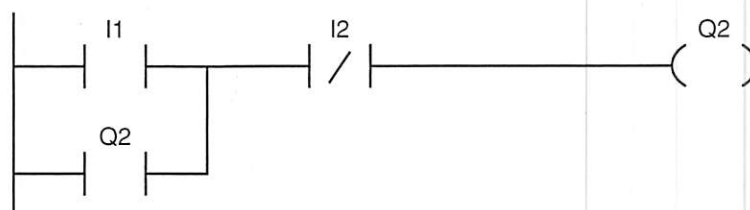
Son de dos tipos:

- Lenguajes gráficos de contactos (LD) (KOP).
- Lenguajes gráficos de funciones lógicas (FBD) (FUP).

Lenguaje gráfico de contactos (LD) (KOP)

Los símbolos empleados son similares a los de los esquemas de circuitos eléctricos a relés, por lo tanto, la transcripción para un técnico automatista es mucho más inmediata que otro tipo de lenguaje.

El abaratamiento de los equipos informáticos está favoreciendo el empleo de la programación gráfica.



↑ Figura 10.3. Ejemplo de programa con lenguaje gráfico de contactos.

Nemotecnia gráfica del lenguaje LD

Se aplica tanto en la transcripción de los esquemas eléctricos cableados a lista de instrucciones, como en la elaboración de programas en lenguaje gráfico.

Su utilización es similar a la de los circuitos de automatismos con relés.

Los símbolos básicos son:

	Contacto abierto		Bobina directa
	Contacto cerrado		Bobina inversa
	Flanco positivo		Bobina de activación
	Flanco negativo		Bobina de desactivación
	Negación		

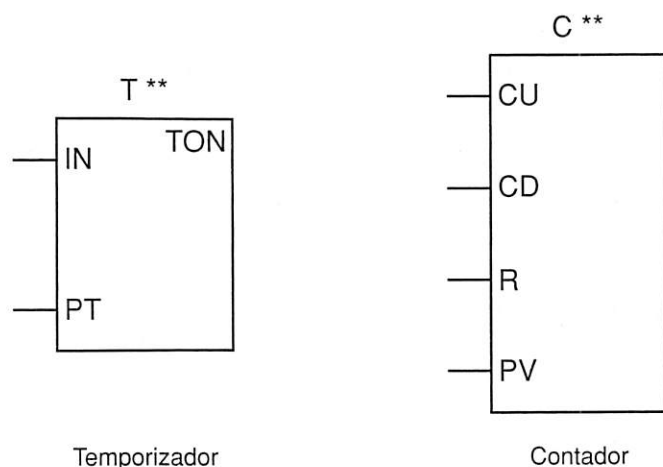
↑ Figura 10.4. Símbolos especiales.

saber más

Simatic S7 denomina a los lenguajes **AWL** al lenguaje por lista de instrucciones, **KOP** al lenguaje de contactos y **FUP** al lenguaje de funciones.

Representan operadores con funciones específicas. Algunas de estas pueden presentar grandes diferencias entre diferentes modelos de autómatas.

Los bloques para funciones especiales se representan con un rectángulo en cuyo interior se indican con símbolos alfanuméricos las diferentes operaciones que pueden realizar.



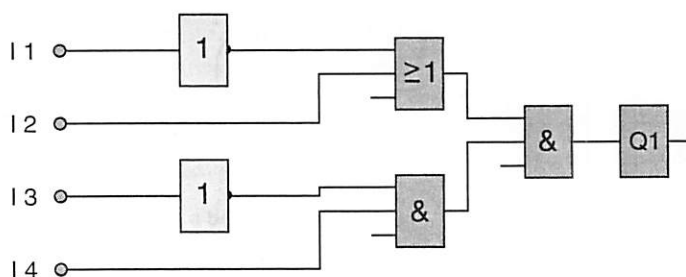
↑ Figura 10.5. Ejemplo de símbolos especiales.

El lenguaje de contactos es el empleado en este libro, por su similitud a los esquemas eléctricos con contactores y relés.

Lenguaje gráfico de funciones lógicas (FBD) (FUP)

Este lenguaje utiliza bloques lógicos similares a los utilizados en electrónica digital.

Las variables de entrada se representan a la izquierda de los bloques y las de salida a la derecha.



↑ Figura 10.6. Ejemplo de esquema de funciones lógicas.

2.2. Lenguajes textuales

La norma define dos tipos:

- Lenguaje por Lista de Instrucciones (IL) (AWL).
- Lenguaje Estructurado (ST).

Lenguaje por lista de instrucciones (IL) (AWL)

También llamado *booleano*, está basado en un listado de símbolos nemotécnicos, cercanos al lenguaje máquina. Se escribe en forma de texto utilizando caracteres alfanuméricos para definir las líneas de operaciones lógicas.

Es el lenguaje más próximo al juego de instrucciones de la CPU por lo que generalmente suele ser el más potente y rápido de escribir, aunque también el más complejo.

Esta es la única forma de programación en los PLCs más antiguos.

saber más

Ejemplo de un programa en lista de instrucciones:

LD	I1
O	Q1
AN	I2
=	Q1

vocabulario

Booleano

Referente al álgebra lógica elaborada por el matemático George Boole.

Lenguaje estructurado (ST)

Tiene su origen en los lenguajes de alto nivel como el Basic, C o Pascal, siendo su programación similar a ellos.

Solamente los autómatas de alta gama permiten este tipo de programación.

La utilización del ST se sale de los objetivos de este módulo, por lo tanto no será tratada aquí.

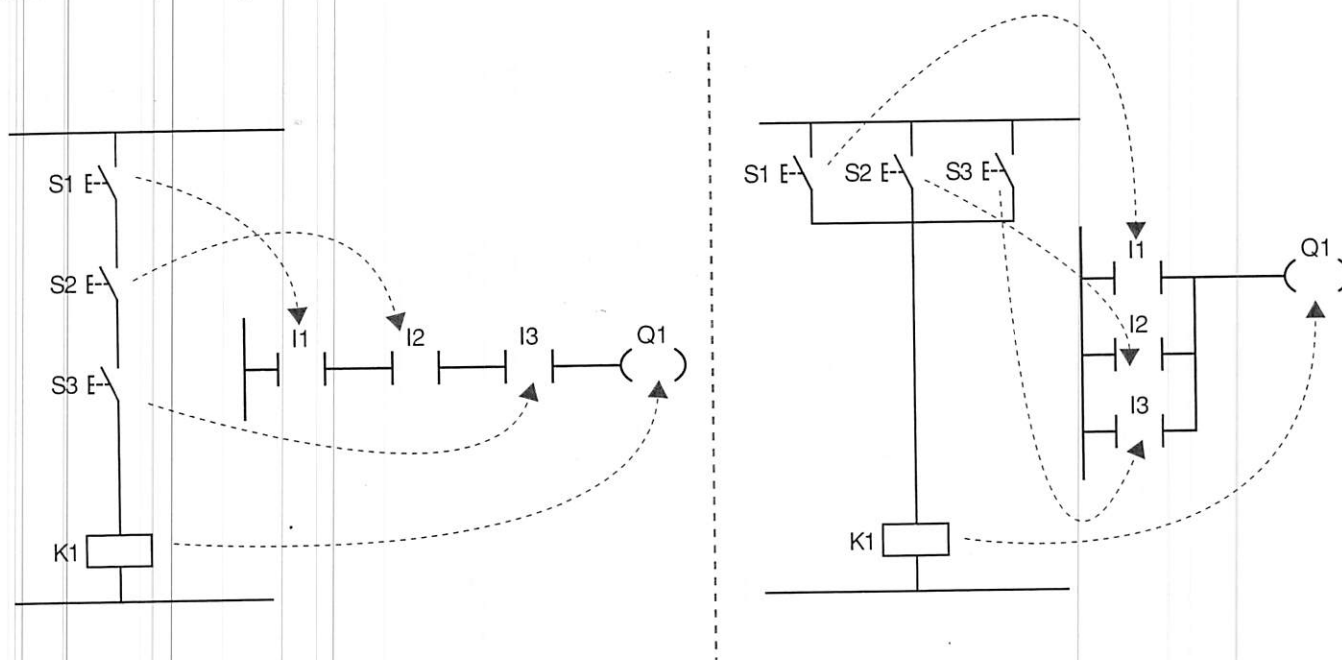
3. Representación en lenguaje de contactos

caso práctico inicial

Por su similitud con los esquemas eléctricos de automatismos cableados, el lenguaje de programación a contactos LD o KOP es muy popular entre los técnicos de automatización que recibieron su formación inicial basada en ese tipo de esquemas.

Los esquemas empleados en la elaboración de programas para autómatas son similares, en lo básico, a los utilizados en los circuitos eléctricos con relés y contactores. Por lo tanto, antes de comenzar a programar es necesario realizar el esquema sobre el papel con la nemotecnia normalizada para posteriormente escribir el programa en cualquiera de los lenguajes vistos anteriormente.

Aquí se muestra cómo transcribir dos esquemas eléctricos de automatismos a lenguaje de contactos (LD).



↑ **Figura 10.7.** Dos ejemplos de esquemas eléctricos transcritos en esquemas para autómatas.

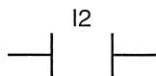
3.1. Elementos básicos de programación en lenguaje LD

Contactos

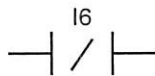
Son elementos de tipo bit que detectan los estados todo o nada de una variable. Se representan como abiertos y cerrados y su uso es idéntico al de los esquemas a relés y contactores.

Pueden estar asociados tanto entradas y salidas físicas (I/Q), como a elementos de uso interno del autómata como temporizadores, contadores, marcas, etc.

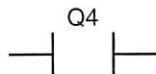
Contacto normalmente abierto asociado a la entrada I2 del autómata



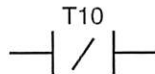
Contacto normalmente cerrado asociado a la entrada I6 del autómata



Contacto normalmente abierto asociado a la salida Q4



Contacto normalmente cerrado asociado a un temporizador

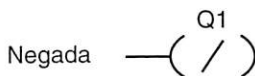
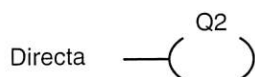


↑ Figura 10.8. Variables asociadas a contactos.

Bobinas

Reciben este nombre por herencia de los circuitos a relés. En ellas se escribe el resultado lógico de una red de contactos.

Pueden ser de diferentes tipos:



saber más

No todos los autómatas permiten la programación de bobinas negadas.

↑ Figura 10.9. Salidas asociadas a bobinas.

La de tipo directo, se activa cuando el valor lógico de la red de contactos es verdadero, sin embargo, la de tipo negado hace lo contrario.



↑ Figura 10.10. Uso de bobina directa y bobina negada.

Las bobinas de activación y desactivación son denominadas como SET y RESET respectivamente.

En una bobina SET, cuando el valor de la red de contactos es verdadero se activa y se mantiene dicho valor aunque cese la acción que lo activó. En este caso, se puede decir que es una bobina con memoria.

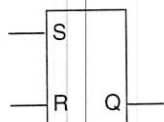
Para desactivar el valor del operando asociado a la bobina SET, se debe utilizar una nueva red de contactos que escriba sobre la bobina RESET asociada a la misma variable que la del SET.

saber más

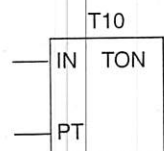
A la acción de activar mediante una bobina SET se le conoce vulgarmente como «seteo» y la acción de desactivar mediante RESET como «reseteo».

recuerda

En lenguaje FBD las bobinas de enganche/ desenganche se representan en un solo bloque denominado biestable:



↑ Figura 10.11. Biestable.

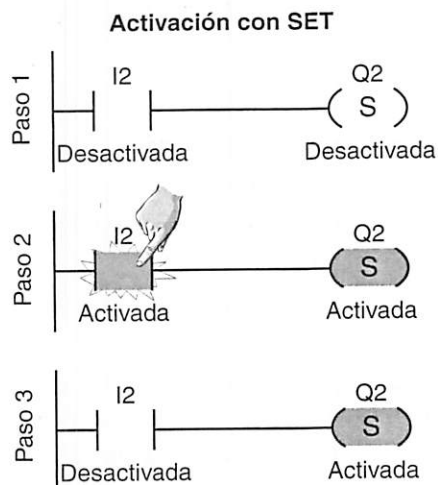


↑ Figura 10.14. Aspecto de un bloque temporizador básico.

saber más

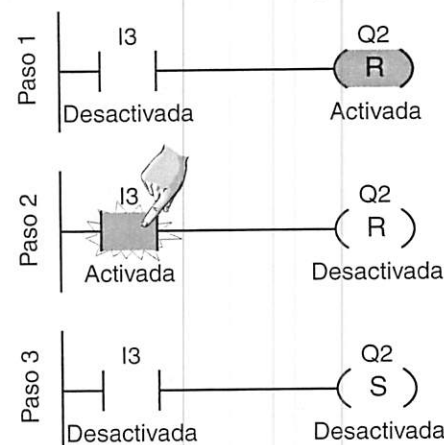
Algunos modelos de autómatas disponen de diversos tipos de temporizadores, aunque los más habituales son los temporizadores **a la conexión y a la desconexión**. La norma denomina **TON** a los primeros y **TOF** a los segundos.

En este ejemplo, la salida Q2 se activa mediante I2 y se desactiva con I3.



↑ Figura 10.12. Activación con salida con SET.

Desactivación con RESET previo SET



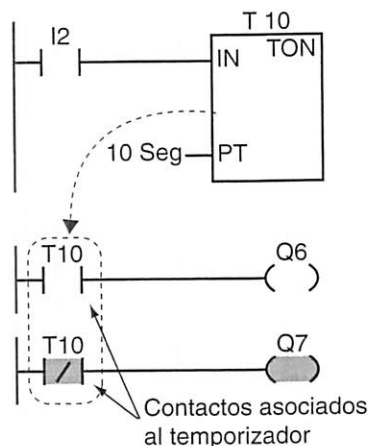
↑ Figura 10.13. Desactivación de salida con RESET.

Temporizadores

Los temporizadores generan eventos cuando alcanzan un valor de tiempo predefinido (PT). Su comportamiento es similar a los utilizados en la lógica cableada.

El temporizador se representa gráficamente como un rectángulo con varias líneas de entrada para conectar la red de contactos de activación (In) y configurar el tiempo de temporización (PV).

A los temporizadores se les asocian contactos abiertos y cerrados que cambian de estado cuando alcanza el valor de tiempo máximo.



En este ejemplo, si se mantiene activa la I2, el T10 Temporiza. Una vez transcurridos los 10 segundos configurados en la entrada PT, los contactos asociados al temporizador se cierran y se abren, activando y desactivando las salidas Q6 y Q7 respectivamente.

Si antes de que se cumpla el tiempo de preselección, se desactiva la entrada IN, el temporizador se inicializa.

← Figura 10.15. Uso de temporizador.

ACTIVIDADES

1. Programa y prueba en un autómata:

- La activación y desactivación de una salida mediante dos entradas utilizando bobinas Set y Reset.
- El ejemplo del temporizador de la figura 10.15.

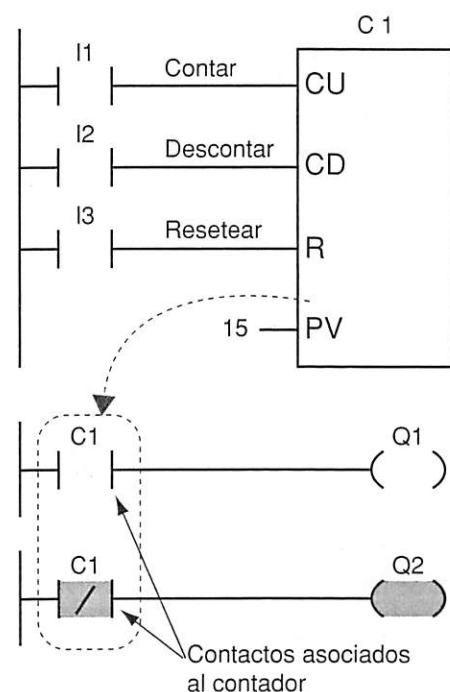
Contadores

Los contadores originan eventos cuando alcanzan un determinado número de sucesos.

Son ideales para utilizar en circuitos automáticos que requieren contabilizar objetos y, en función del número almacenado, actuar sobre el programa.

Se representan en forma de rectángulo y disponen de varias líneas de entrada para conectar las redes de contactos que permiten contar (CU), descontar (CD), poner a RESET (R) y configurar el valor de preselección (PV).

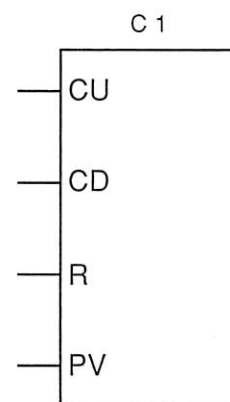
A los contadores se les asocian contactos abiertos y cerrados que cambian de estado cuando alcanza el valor preseleccionado.



En este ejemplo la entrada I1 incrementa en 1 el valor almacenado en el contador, la I2 decrementa dicho valor y la I3 lo pone a 0.

Si se consigue el valor de preselección, que en este caso es 15, los contactos del contador se cierran y abren activando la salida Q1 y desactivando la salida Q2 respectivamente.

← Figura 10.17. Uso de un contador.



↑ Figura 10.16. Aspecto de un bloque contador/descontador.

saber más

Dos ejemplos del uso de contadores pueden ser:

- Controlar el número de coches de un aparcamiento, contando los que entran y descontando los que salen.
- Contabilizar el número de pizzas que pasan por una cinta transportadora.

3.2. Operaciones lógicas en lenguaje de contactos

En este capítulo estudiarás las operaciones básicas que se pueden realizar en la programación de autómatas. Todos los ejemplos están implementados de forma genérica en Lenguaje de Contactos, sin estar vinculados a ningún modelo de autómata en particular. Corre de tu parte la adaptación de los programas al dispositivo que utilices en el aula-taller.

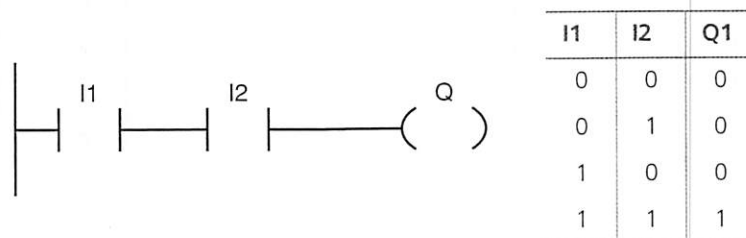
Conexión Serie (AND)

Tiene su correspondencia con el circuito eléctrico combinacional de contactos en serie.

ACTIVIDADES

2. Programa y prueba el ejemplo del contador mostrado en la figura 10.17.

El resultado de la operación solamente se escribe en la bobina cuando los operandos de los contactos «no negados» se encuentran a «1» y los negados a «0».

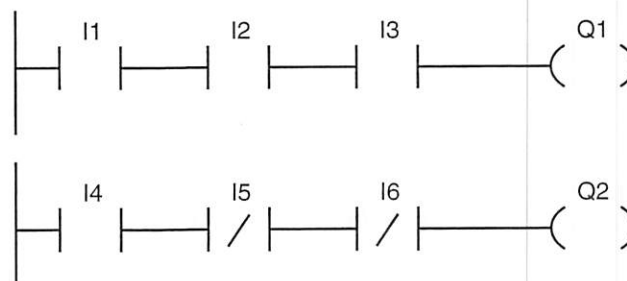


$$Q1 = I1 \cdot I2$$

↑ Figura 10.18. Asociación de contactos en serie. Tabla de la verdad.

Ecuación lógica.

Diferentes ejemplos de programación de contactos en serie.

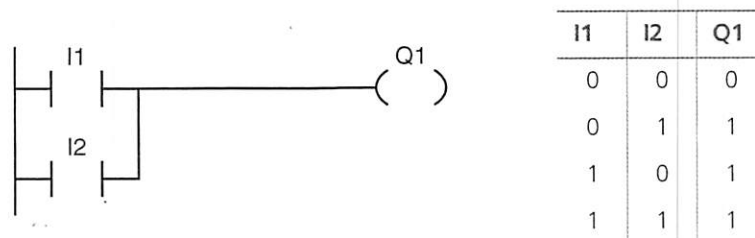


↑ Figura 10.19. Contactos en serie negados y sin negar.

Conexión paralelo (OR)

Tiene su correspondencia con el circuito eléctrico combinacional de contactos en paralelo.

El resultado de la operación se escribe en la bobina cuando cualquiera de los operandos «no negados» está a «1» o cualquiera de los negados está a «0».

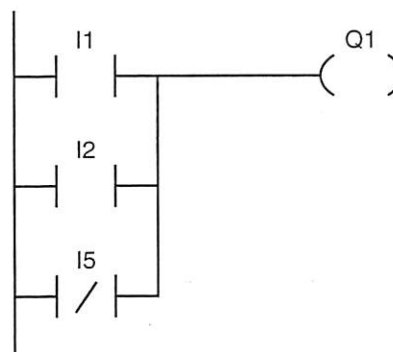


$$Q1 = I1 + I2$$

↑ Figura 10.20. Asociación de contactos en paralelo. Tabla de verdad.

Ecuación lógica

En paralelo también se pueden asociar contactos cerrados:

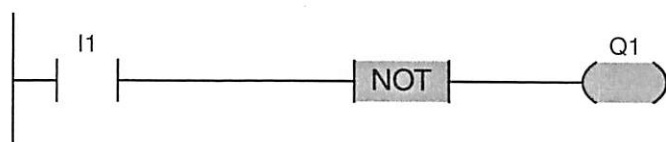


↑ Figura 10.21. Otro ejemplo de asociación en paralelo.

Operación negación (NOT)

La operación NOT invierte el valor de la operación lógica escrita previamente a ella.

Se representa como un contacto con la palabra NOT en su interior y no tiene asociado ningún operando.



↑ Figura 10.22. Operación NOT.

I1	Q1
0	1
1	0

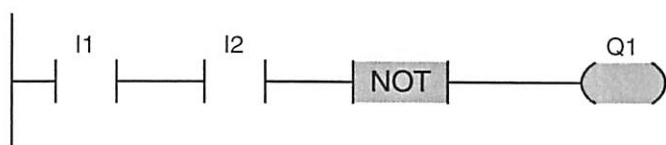
Tabla de la verdad.

$$Q1 = \overline{I1}$$

Ecuación lógica.

Operación serie negada (NAND)

El resultado de esta operación es inverso al de la operación serie (AND). Se consigue colocando una negación NOT después de la asociación de contactos.



↑ Figura 10.23. Operación NAND.

I1	I2	Q1
0	0	1
0	1	1
1	0	1
1	1	0

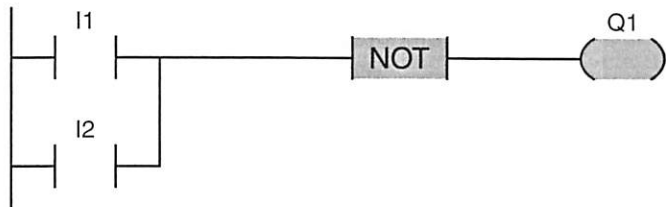
Tabla de la verdad.

$$Q1 = \overline{I1 \cdot I2}$$

Ecuación lógica.

Operación paralelo negada (NOR)

El resultado de esta operación es inverso al de la operación paralelo (OR). Se consigue colocando una negación NOT después de la asociación de contactos.



↑ Figura 10.24. Operación NOR.

I1	I2	Q1
0	0	1
0	1	0
1	0	0
1	1	0

Tabla de la verdad

$$Q1 = \overline{I1 + I2}$$

Ecuación lógica

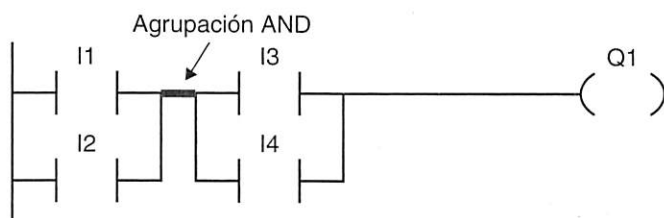
Agrupaciones de contactos

Los contactos pueden operar entre sí creando bloques, que a su vez operan con otros bloques en serie o en paralelo originando agrupaciones de contactos.

Agrupación AND

Conecta grupos de contactos con la función serie (AND).

En la siguiente figura se muestra una agrupación serie de dos bloques de contactos en paralelo.

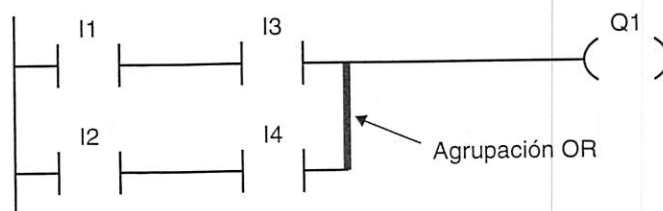


← Figura 10.25. Agrupación AND.

Agrupación OR

Conecta grupos de contactos con la función paralelo (OR).

La figura siguiente muestra la agrupación de dos bloques de contactos en serie conectados entre sí en paralelo.



↑ Figura 10.26. Agrupación OR.

Operaciones con marcas internas

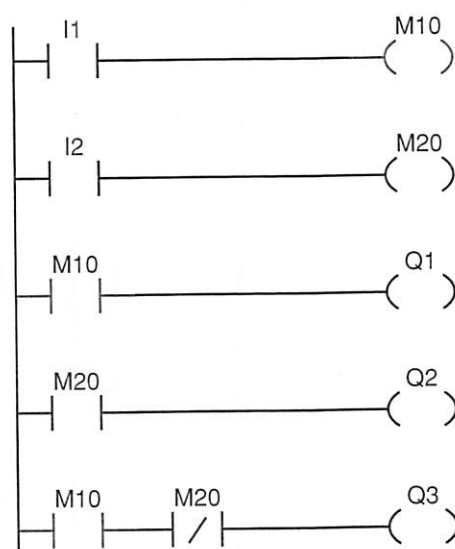
También llamadas memorias o bits internos. Tienen un comportamiento similar al de los relés auxiliares en la lógica cableada.

Su representación gráfica se realiza con el símbolo de la bobina, tanto en forma directa como negada.



↑ Figura 10.27. Marcas asociadas a bobinas.

A las memorias internas se les asocian contactos, abiertos y cerrados, que cambian de estado cuando se activan.



En este ejemplo las entradas I1 e I2 activan las marcas M10 y M20 respectivamente.

Cuando la M10 está a «1», se cierran sus dos contactos, activando las salidas Q1 y Q3.

Cuando M20 está a «1» se cierra el contacto abierto de la cuarta rama, activando Q2, y se abre el cerrado de la quinta, impidiendo que Q3 se active aunque M10 esté a valor verdadero.

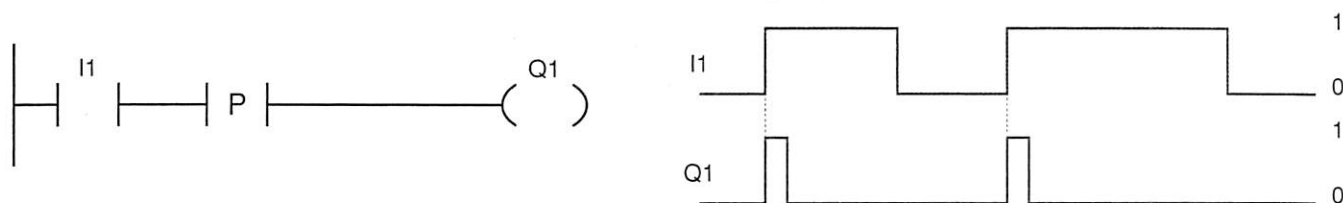
→ Figura 10.28. Uso de las marcas internas.

Flanco positivo

También llamado frente ascendente, detecta el paso de 0 a 1 del operando asociado.

El símbolo es: —| P |—

Por ejemplo, una entrada asociada a una operación de flanco positivo solamente generará la señal «1» en el instante de ser pulsado, pasando a la situación «0» inmediatamente aunque continúe la acción sobre el pulsador.



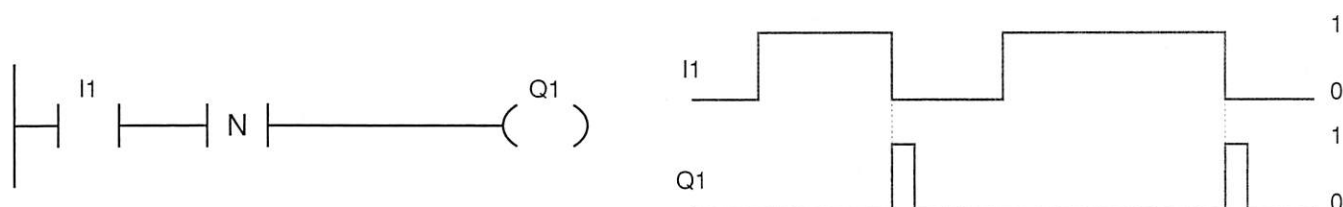
↑ Figura 10.29. Asociación de un flanco positivo a una entrada y cronograma del flanco positivo.

Flanco negativo

También llamado frente descendente, detecta el paso de 1 a 0 del operador asociado.

El símbolo es: —| N |—

Utilizando el ejemplo del punto anterior, solamente se generará la señal «1» en el instante de «soltar» el pulsador.



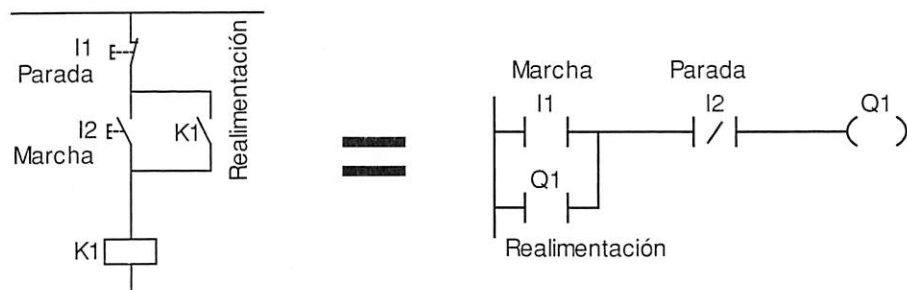
↑ Figura 10.30. Asociación de un flanco negativo a una entrada y cronograma del flanco negativo.

Operaciones de memoria

Una función de memoria es aquella que genera un evento y lo mantiene aunque cese la acción inicial que lo originó.

Realimentación por contacto

En la lógica cableada esta operación la realiza el circuito llamado «Arranque con pulsadores de marcha y paro», donde el pulsador de marcha es el que origina el evento que alimenta la bobina, el contacto en paralelo el que la realimenta y el pulsador de parada el que la desactiva. Por tanto, si se transcribe dicho circuito al lenguaje de contactos se obtiene el programa para activar y desactivar una bobina mediante realimentación.



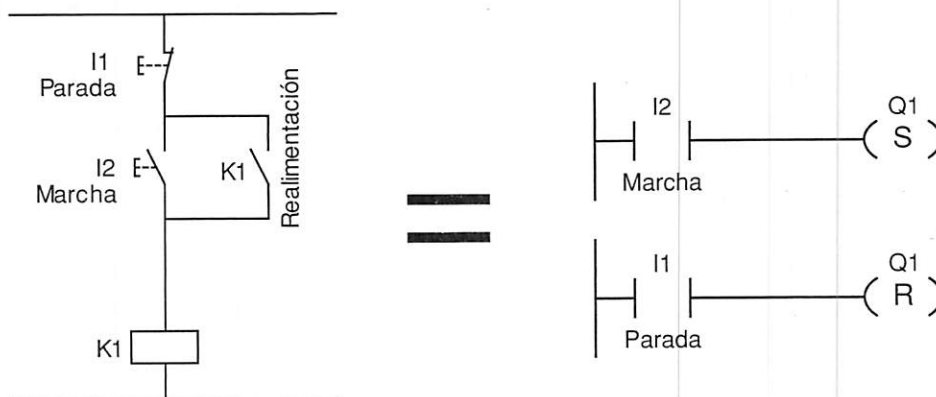
↑ Figura 10.31. Realimentación por contacto.

Como el contacto en paralelo a la entrada de Marcha I1 pertenece a la bobina, su direccionamiento debe ser el mismo, es decir Q1.

Operación memoria con bobinas SET y RESET

La misma operación de memoria se puede programar utilizando las bobinas de enganche —(S) y desenganche —(R) descritas anteriormente. Al escribir sobre la bobina S el operando asociado se pone a «1» y se mantiene hasta que se realiza la escritura sobre la bobina R que lo pone a «0».

El uso de estos tipos de bobinas facilita la elaboración de programas en detrimento de la programación por realimentación.



↑ Figura 10.32. Operación memoria con SET y RESET.

4. GRAFCET

El GRAFCET, también denominado SFC, es una forma gráfica de representar el funcionamiento de un sistema secuencial.

El GRAFCET representa la secuencia de funcionamiento de la máquina y facilita la implementación a cualquier lenguaje de programación de autómatas.

El GRAFCET está formado por un conjunto de símbolos denominados etapas, transiciones, etiquetas y líneas de dirección.

4.1. Etapas

Representan los diferentes estados del proceso secuencial.

Su símbolo es un cuadrado con un número en su interior que debe ser único en la secuencia.

Pueden ser de dos tipos: de Paso e Iniciales.



↑ Figura 10.33. Etapas

La etapa inicial es el estado en el que comienza el proceso. Aunque no es habitual, puede haber más de una etapa inicial en un mismo GRAFCET.

vocabulario

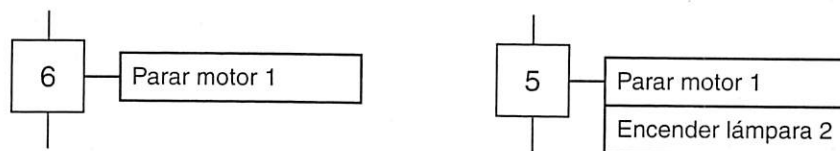
GRAFCET

Es el acrónimo de Gráfico de Control Etapas Transición.

SFC

Es el acrónimo de Sequential Function Chart (Gráfico de Función Secuencial).

De las etapas cuelgan las denominadas **etiquetas** en las que se indican las acciones a realizar.



↑ Figura 10.34. Formas de representar acciones en etapas.

Cuando la secuencia llega a una etapa determinada, se ejecutan las acciones que en ella se indican.

En un GRAFCET solamente puede estar una etapa activa, excepto si la secuencia se ha diseñado con más de una etapa inicial o existen secuencias paralelas.

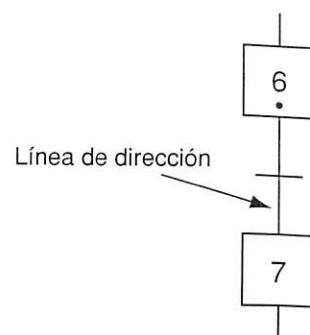
4.2. Transiciones

Una transición es una condición que permite el paso de una etapa a otra.

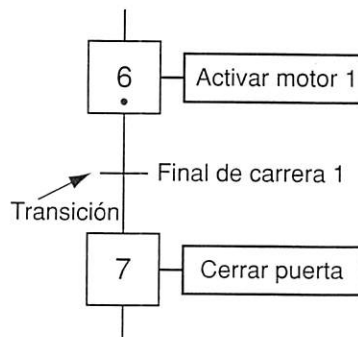
Se representa con una línea horizontal en forma de cruz sobre la línea de dirección.

4.3. Tipos de GRAFCET

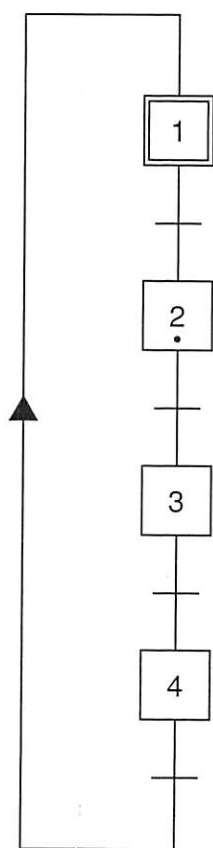
Según la evolución de la secuencia, tres son los tipos de GRAFCET que se pueden representar:



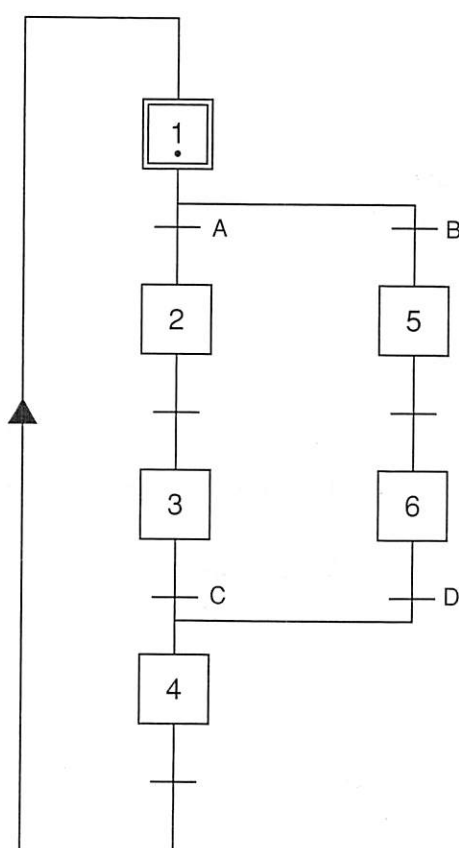
↑ Figura 10.35. Línea de dirección en un GRAFCET.



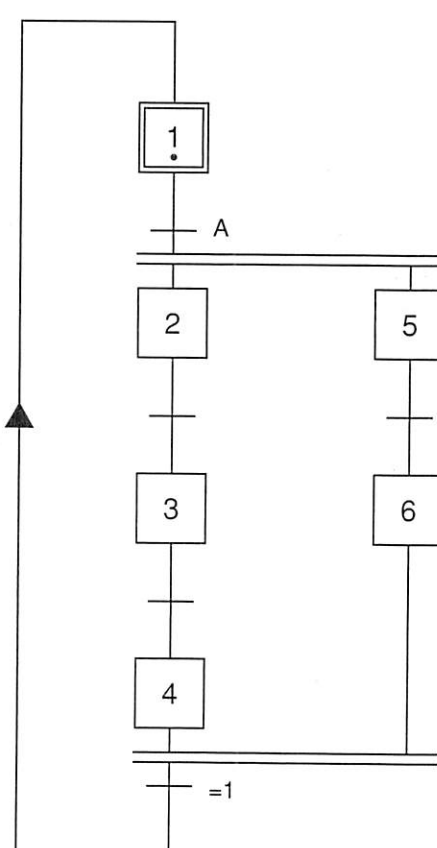
↑ Figura 10.36. Transición.



↑ Figura 10.37. GRAFCET de secuencia única.



↑ Figura 10.38. GRAFCET de secuencias opcionales.

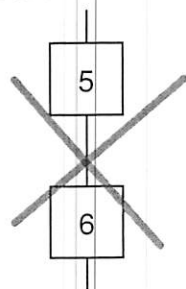


↑ Figura 10.39. GRAFCET de secuencias simultáneas.

saber más

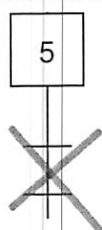
Reglas del GRAFCET

- No pueden representarse dos etapas consecutivas.



↑ Figura 10.40.

- No pueden representarse dos transiciones consecutivas.



↑ Figura 10.41.

- El orden de evolución debe ser siempre etapa-transición-etapa-transición...

caso práctico inicial

El GRAFCET es la herramienta ideal para resolver los problemas de automatización de los taladros del caso práctico inicial.

GRAFCET de secuencia única

Es un conjunto de etapas y transiciones conectadas en cascada. En este caso, la evolución solamente sigue un camino, ya que no existen convergencias y divergencias en la línea de dirección.

Es la forma más sencilla de representar la secuencia de un proceso, pero la más incompleta.

GRAFCET de secuencias opcionales

En este tipo de GRAFCET la secuencia puede optar por seguir entre dos o más caminos, pero solamente se ejecuta uno a la vez.

En la figura 10.38 se muestra cómo estando en la etapa inicial, el proceso puede evolucionar por la 2, a través de la transición A, o por la etapa 5, a través de la transición B.

La convergencia y divergencia de las vías opcionales se representa con una línea de dirección dibujada en horizontal.

GRAFCET de secuencias simultáneas

Durante la evolución de este tipo de GRAFCET, se ejecutan dos o más caminos a la vez.

En la figura 10.39 se muestra cómo al cumplirse la transición A, se ejecutan simultáneamente los caminos que comienzan por las etapas 2 y 5. Ambos deben terminarse, con las etapas 4 y 6 respectivamente, para volver a la secuencia principal.

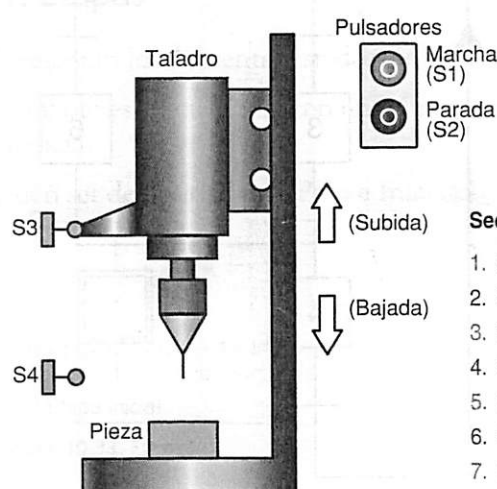
Hasta que todas estas vías secundarias no se han realizado completamente, la secuencia no evoluciona por la del camino principal.

La convergencia y divergencia de las vías simultáneas se representa con una línea de dirección doble dibujada en horizontal.

EJEMPLO

Ejemplo de aplicación de un GRAFCET

La secuencia de trabajo del taladro de la figura escrita en lenguaje natural es la siguiente:



↑ Figura 10.42. Taladro.

Secuencia del proceso:

1. Estado de espera.
2. Evento sobre el pulsador de marcha.
3. Giro de la broca y bajada del taladro.
4. Evento sobre el final de carrera inferior (S4).
5. Subida del taladro con la broca girando.
6. Evento sobre el final de carrera superior (S3).
7. Paro total y vuelta al estado de espera.

En ella se puede ver cómo ocurren eventos sobre captadores y se ejecutan acciones sobre los motores.

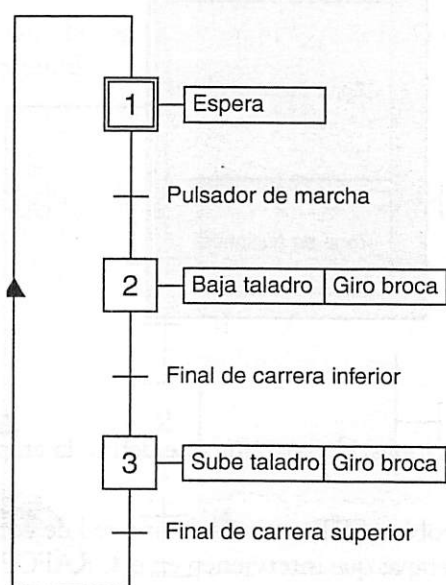
Los eventos (de color azul) son las transiciones del GRAFCET y las acciones (de color verde) son las etapas.

Esto permite crear el denominado GRAFCET Descriptivo o de Primer Nivel.

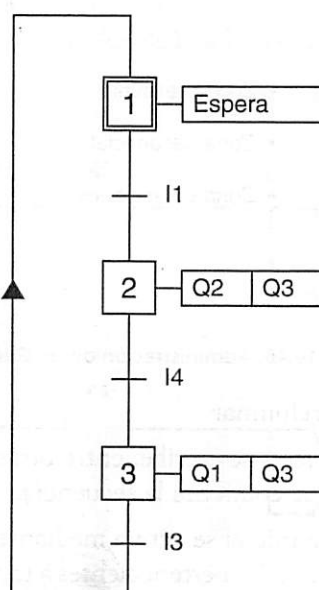
En la segunda fase de diseño, se aplica al GRAFCET la tecnología a utilizar. En nuestro caso el control se realizará mediante un autómata programable. Por tanto, es necesario asociar los captadores y actuadores que intervienen en la máquina a las transiciones y acciones de la secuencia.

Captadores y actuadores	Variables del autómata
Pulsador de marcha (S1)	I1
Final de carrera superior (S3)	I3
Final de carrera inferior (S4)	I4
Contactador de subida de taladro	Q1
Contactador de bajada de taladro	Q2
Contactador giro de broca	Q3

Así, el GRAFCET de segundo nivel o tecnológico del taladro utilizado como ejemplo es el siguiente:



↑ Figura 10.43. GRAFCET de primer nivel del taladro.



↑ Figura 10.44. GRAFCET de segundo nivel o tecnológico.

vocabulario

Español-Inglés

O: *or*

Y: *and*

Marca: *mark*

Lista de instrucciones:

list of instructions

Escalera: *ladder*

Flanco: *flank*

Función lógica: *logic function*

Salida: *output*

Entrada: *input*

Contador: *counter*

Etapas: *stage*

Paso: *step*

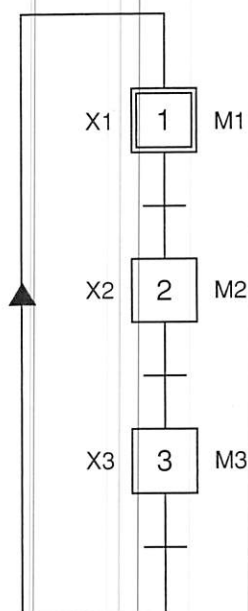
Transición: *transition*

saber más

Actualmente la mayoría de los autómatas permiten trabajar con los datos en formato simbólico. Esto significa que el usuario puede definir una serie de símbolos asociados a los operandos del autómata, de forma que en el momento de la programación, se utilizan dichos símbolos en lugar de la nemotecnia propia del equipo.

Por ejemplo:

Se puede definir que cada una de las salidas utilizadas en el programa (Q0.0, Q0.1, Q0.2...), se les denomine literalmente con nombres simbólicos (Motor1, Lámpara4, Válvula2...) que faciliten su identificación. Así en el momento de la programación, se utilizarán estos símbolos en lugar de las denominaciones nemotécnicas del autómata.



↑ Figura 10.45. Asociación de Marcas y nombres simbólicos a las etapas del GRAFCET.

4.4. Programación de un GRAFCET en lenguaje de contactos

Aunque algunos autómatas permiten su programación directa, bien gráficamente o por lista de instrucciones, se hace necesario conocer el algoritmo de implementación en alguno de los lenguajes anteriormente nombrados.

Aquí se describe cómo pasar la secuencia de un GRAFCET de secuencia única a lenguaje de contactos (LD).

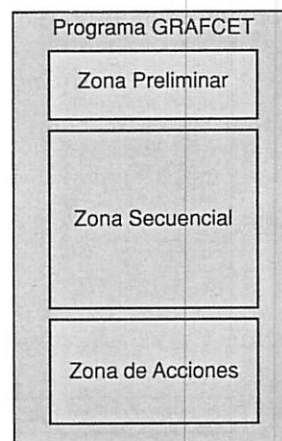
A cada etapa del GRAFCET se le asocia una marca interna de la zona de memoria **M**. Aquí, de forma genérica, se utilizan nombres simbólicos para identificar las etapas. Por norma, estos nombres se identifican con **X** seguido del número de la etapa.

Etapa	Marca interna	Nombre simbólico
Etapa 1 (Inicial)	M1	X1
Etapa 2	M2	X2
Etapa 3	M3	X3

Administración de un GRAFCET

La programación de un GRAFCET se divide en tres zonas de administración:

- Zona preliminar
- Zona secuencial
- Zona de acciones

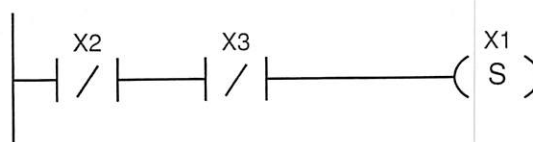


↑ Figura 10.46. Administración de un GRAFCET.

Zona preliminar

En esta zona se escribe, entre otras operaciones, el programa que define la etapa por la que comienza la secuencia.

La etapa inicial se activa mediante una bobina SET, a través de una red de contactos cerrados pertenecientes a todas las etapas que intervienen en el GRAFCET excepto la inicial.



↑ Figura 10.47. Activación de la etapa inicial.

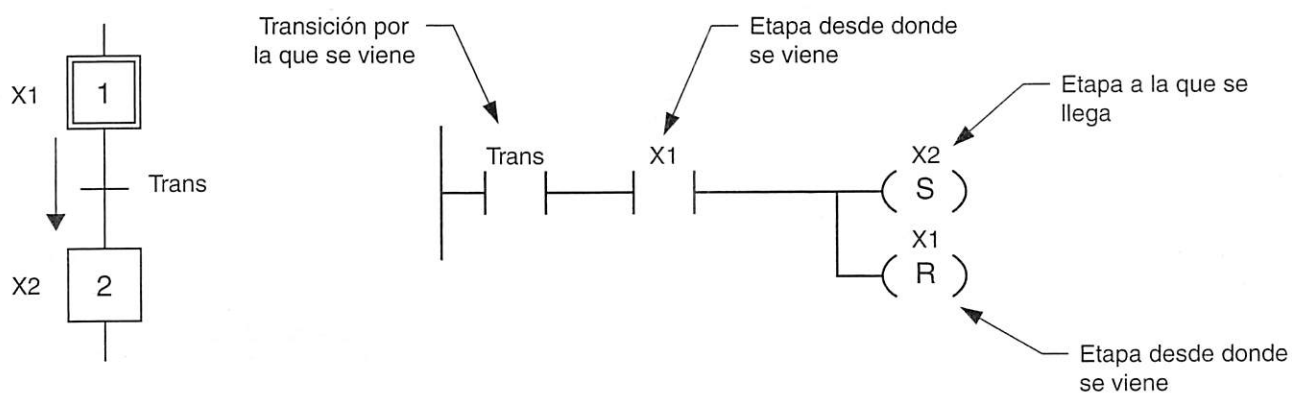
Zona secuencial

Define el gráfico de secuencia que representa el encadenamiento de etapas y las transiciones asociadas.

Cada etapa se activa con una bobina SET, que se programa en una red de contactos en el que se debe «decir» lo siguiente:

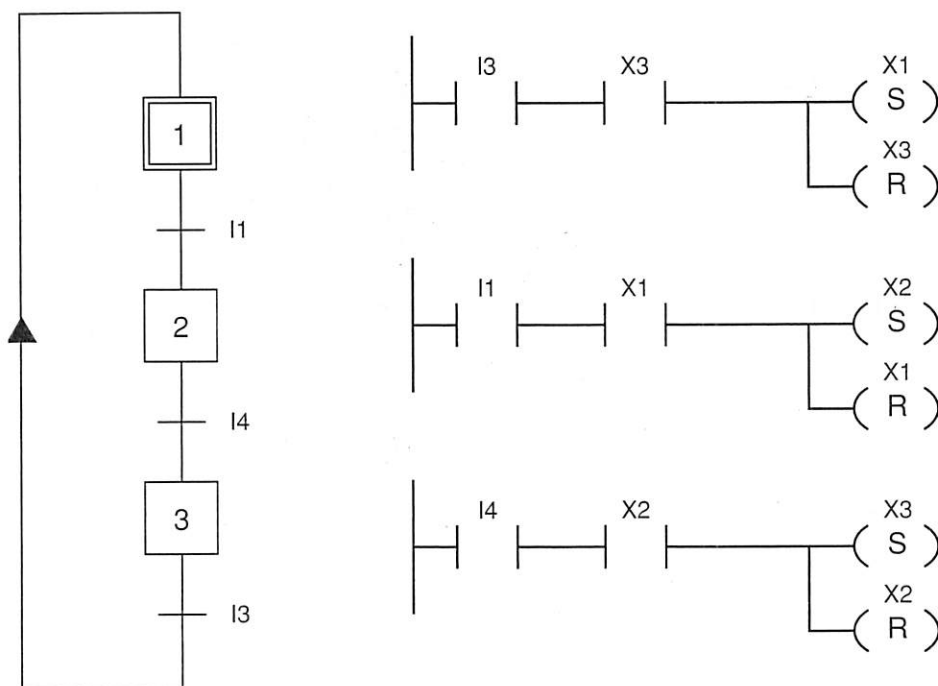
- Etapa a la que se llega.
- Etapa desde la que se viene.
- Transición por la que se viene.

Así, cada segmento de contactos correspondientes a una etapa de la zona secuencial debe tener este aspecto:



↑ Figura 10.48. Zona secuencial.

Así, el programa completo para la secuencia del GRAFCET del taladro es el siguiente:

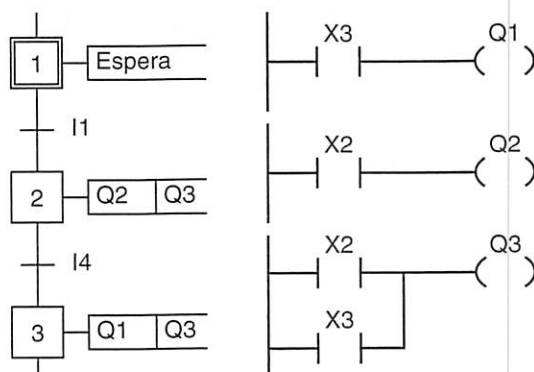


↑ Figura 10.49. Zona secuencial del GRAFCET del taladro en lenguaje de contactos.

Zona de acciones

Ejecuta las acciones originadas en las zonas anteriores activando las salidas del autómata.

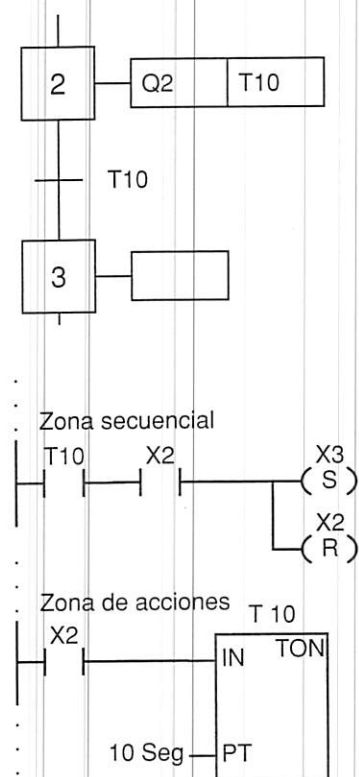
En el ejemplo del taladro, se producen tres acciones sobre las salidas Q1, Q2 y Q3. Estas salidas solamente se deben escribir una vez en el programa. Para indicar que una salida se activa en más de dos etapas, como ocurre con la Q3 del ejemplo, se deben poner en paralelo contactos de estas etapas en la red de contactos dicha acción.



→ **Figura 10.50.** Zona secuencial del GRAFCET del taladro en lenguaje de contactos.

saber más

En una secuencia GRAFCET, también se pueden utilizar temporizadores. En este caso, la activación del temporizador se realiza en la zona de acciones y el uso de los contactos asociados, en la zona secuencial.

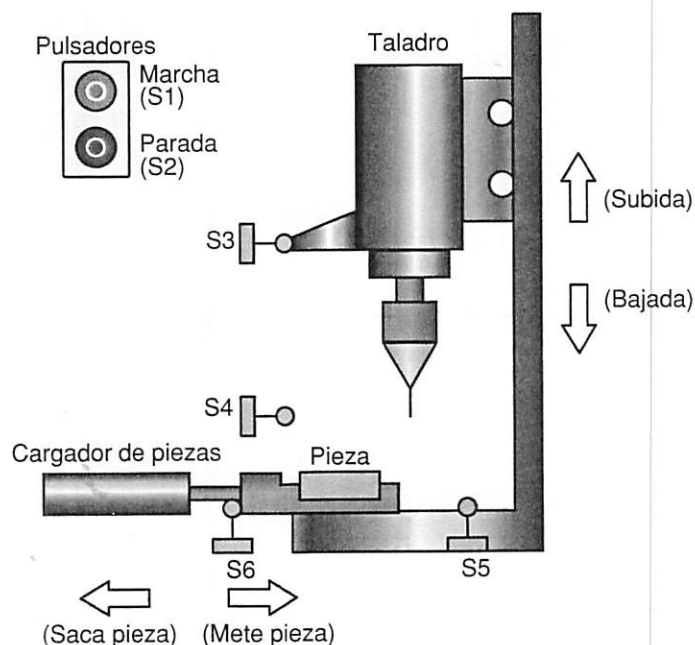


↑ **Figura 10.51.** Temporizador en GRAFCET.

La unión de las tres zonas, completan el programa del GRAFCET del taladro.

ACTIVIDADES

- Elabora los GRAFCET de primer y segundo nivel de una variante del taladro con cargador de piezas.
- Programa y comprueba en un autómata el funcionamiento del GRAFCET del taladro.



↑ **Figura 10.52.**

4.5. Programación de los diferentes tipos de GRAFCET

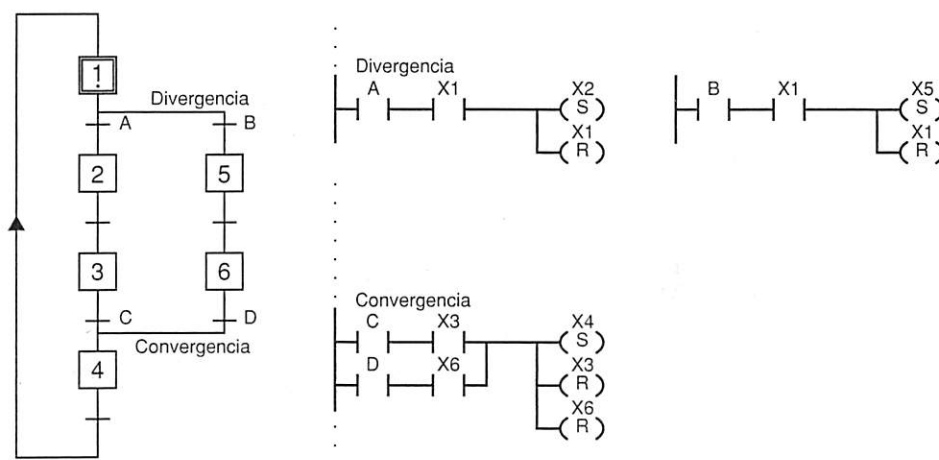
GRAFCET de secuencias opcionales

La zona preliminar y la zona de acciones se programan de la misma forma que en el GRAFCET de secuencia única. La diferencia se encuentra en la zona secuencial.

Las secuencias opcionales se abren con una divergencia y se cierran con una convergencia y precisamente en estos puntos de la secuencia es donde existen diferencias con la programación del GRAFCET de secuencia única.

caso práctico inicial

Las secuencias opcionales permiten crear saltos y retornos que resultan perfectos para implementar paradas de emergencia y rearmes en procesos secuenciales como los del caso práctico.

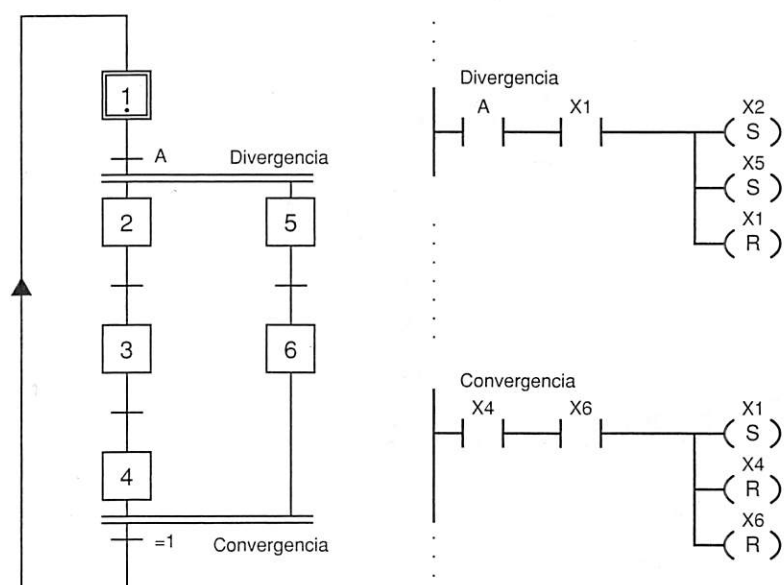


↑ Figura 10.53. Detalles de programación de la zona secuencial de GRAFCET de secuencias opcionales

GRAFCET de secuencias simultáneas

La programación de la zona preliminar y de acciones es idéntica a las vistas anteriormente.

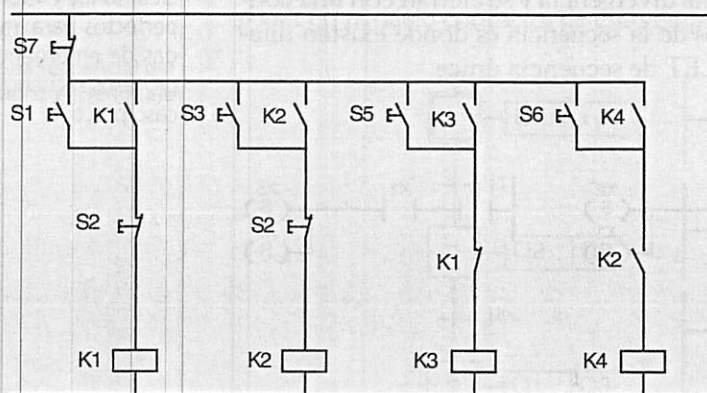
De la misma forma que en el GRAFCET de secuencias opcionales, las secuencias simultáneas se abren con una divergencia y se cierran con una convergencia.



↑ Figura 10.54. Detalles de programación de la zona secuencial de GRAFCET de secuencias simultáneas.

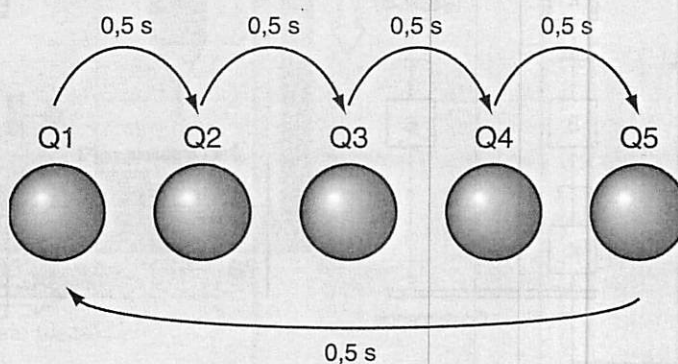
ACTIVIDADES FINALES

- 1. Transcribe al lenguaje de contactos el siguiente circuito eléctrico. Prueba su funcionamiento en el panel de pruebas que montaste en la Práctica Profesional de la unidad anterior.



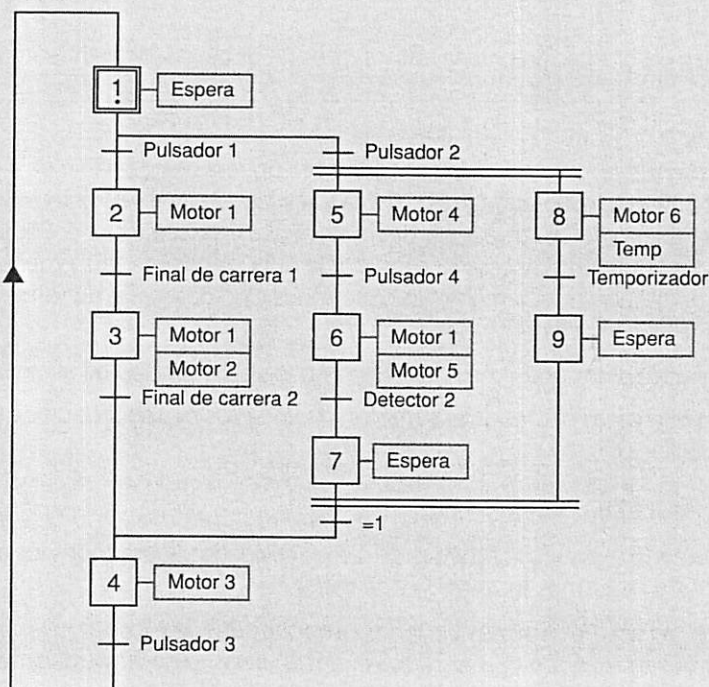
← Figura 10.55. Circuito de automatismos.

- 2. Transcribe a lenguaje de contactos (LD) los circuitos de mando de los siguientes automatismos cableados vistos en unidades anteriores y pruébalos sobre el panel que montaste en la Práctica Profesional de la unidad anterior.
- Arranque de un motor con pulsadores de marcha y paro con realimentación.
 - Arranque de un motor con 3 pulsadores de marcha y 3 de parada con bobinas SET y RESET.
 - Inversión del sentido de giro de un motor trifásico con pulsadores pasando por paro.
 - Inversión del sentido de giro de un motor trifásico con pulsadores. En este caso la inversión de sentido de giro, se debe realizar sin pasar previamente por paro.
 - Arranque estrella-triángulo de un motor trifásico con pulsadores. La conmutación de estrella a triángulo se realizará de forma manual mediante un pulsador.
 - Arranque estrella-triángulo de un motor trifásico con pulsadores. La conmutación de estrella a triángulo se realizará de forma automática con un temporizador.
 - Arranque de un motor trifásico de rotor bobinado con tres grupos de resistencias rotóricas.
- 3. Dibuja los GRAFCET de primer nivel de los circuitos de la actividad anterior.
- 4. Transcribe a lenguaje de contactos los GRAFCET de los circuitos C y F de la actividad 2.
- 5. Realiza un programa, en lenguaje de contactos, que active de forma automática y temporizada cinco salidas del autómata, con una cadencia de 1 segundo entre una y otra. Las salidas deben ir quedando activadas y al concluir el ciclo, después de 1 segundo, deben desactivarse todas y comenzar de nuevo. La actuación sobre una entrada permitirá detener o parar el funcionamiento de la secuencia.



→ Figura 10.56.

- 6. Asocia los elementos de este GRAFCET a las variables de E/S de un autómata programable, transcríbelo a lenguaje de contactos y prueba su funcionamiento sobre el panel de prueba utilizado en las actividades anteriores.



↑ Figura 10.57.

- 7. Dibuja el GRAFCET del taladro con cargador de piezas que aparece en una de las actividades de la unidad anterior. Implementa el programa en lenguaje de contactos en un autómata programable y comprueba su funcionamiento. Para ello puedes ayudarte del panel de pruebas que montaste en la Práctica Profesional de la unidad anterior.

entra en internet

- 8. Busca en internet referencias a la norma IEC 1131-3 que define los lenguajes de autómatas.
- 9. Investiga sobre los lenguajes de programación utilizados por modelos de autómatas de Siemens, Omron, Telemecanique y Allen Bradley. ¿Todos utilizan los mismos lenguajes de programación? ¿Es común la nomenclatura utilizada en ellos para las operaciones lógicas en direccionamientos de variables? ¿Cuántos de ellos se rigen por la norma?
- 10. Localiza cuáles son los orígenes del GRAFCET y cuál es el motivo de su importancia en la actualidad para el desarrollo de sistemas de control secuencial.
- 11. Busca si existen autómatas que permiten la programación gráfica del GRAFCET. Si la respuesta es afirmativa, nombra algunos de ellos.

PRÁCTICA PROFESIONAL

HERRAMIENTAS

- Montaje y programación para la automatización de un taladro semiautomático

MATERIAL

- Panel de pruebas utilizado en las prácticas de automatismos cableados
- Bornes para rail
- Un magnetotérmico bipolar
- Un magnetotérmico tripolar
- Cable de línea de 1,5mm²
- Manguera de 5x2,5mm²
- Una botonera de superficie para dos pulsadores
- Dos pulsadores normalmente abiertos
- Dos finales de carrera de rodana
- Tres contactores
- Dos relés térmicos
- Un autómata programable o relé programable
- Un cable para la programación del autómata desde PC
- Software de programación del PC

Automatización de un taladro

OBJETIVO

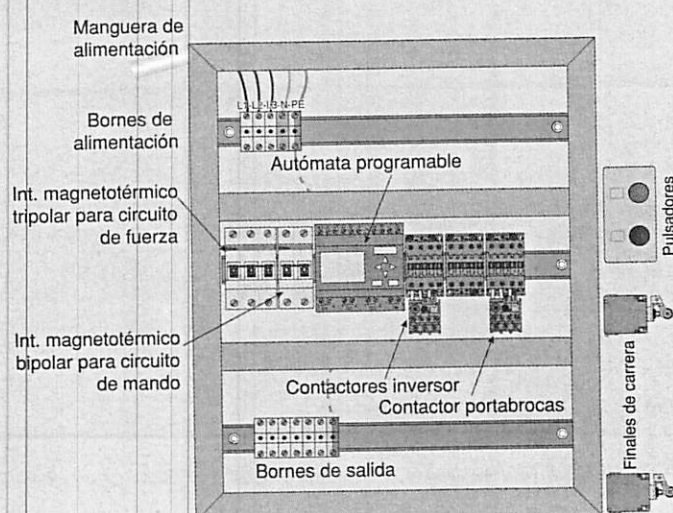
Montaje y programación para la automatización de un taladro semiautomático.

PRECAUCIONES

- **Importante:** en esta práctica se ha utilizado un relé programable de alimentación a 230 Vca, con entradas a 230 Vca y salidas a relés libres de tensión. El autómata que tú utilices puede ser diferente al que aquí se propone. Por tanto, se hace absolutamente necesario consultar la hoja de características del fabricante, para realizar correctamente las conexiones y no deteriorar de forma irremediable la electrónica del autómata.
- No manipules las conexiones con el panel conectado a la red de alimentación.
- Ten en cuenta las pautas que se marcan en las fichas de seguridad que tienes al final del libro.

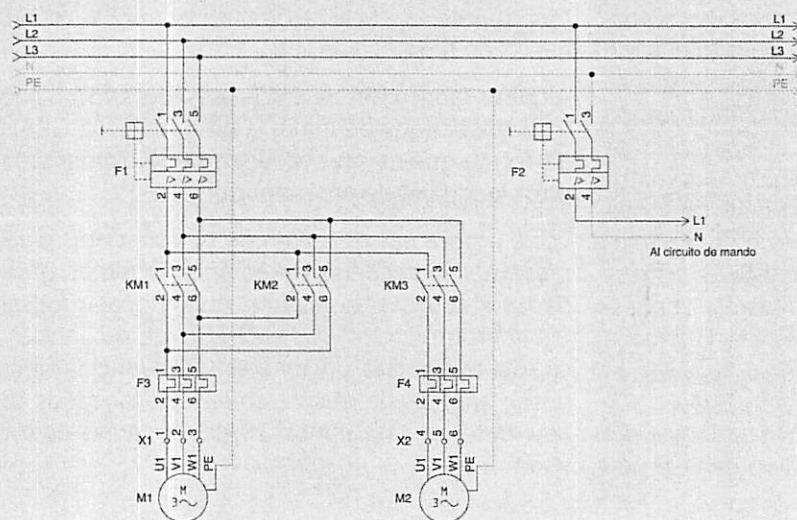
DESARROLLO

1. Utilizando el panel de pruebas de otras actividades, retira todos los aparatos excepto los bornes de entrada y de salida.
2. En el rail central deberás ubicar 3 contactores, dos de ellos con relés térmicos, comprueba que caben en el hueco entre canaletas. Si no es así, deberás desplazar dicho rail para que estos elementos se puedan fijar en el panel de pruebas.
3. Fija los interruptores magnetotérmicos a la izquierda del rail central. El tripolar más a la izquierda y a su derecha el bipolar.
4. Fija el autómata sobre el rail central y a su derecha los tres contactores con sus relés térmicos.
5. En la parte exterior de la canaleta vertical derecha, fija la botonera con sus pulsadores y los dos finales de carrera.



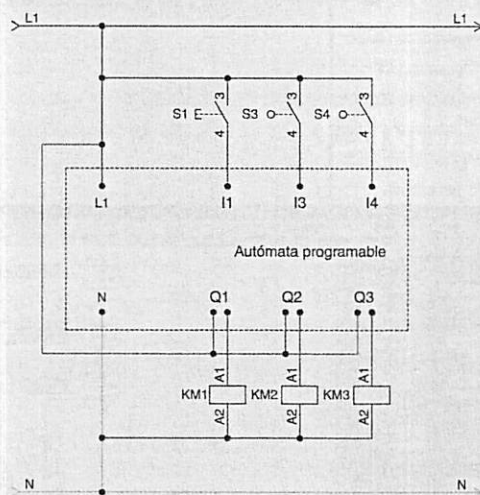
← Figura 10.58. Aparamenta en el panel de pruebas.

6. Desde los bornes de alimentación, cablea las entradas de los magnetotérmicos.
7. Dibuja el esquema de fuerza para la inversión del sentido de giro del motor que sube y baja el taladro y para el arranque directo del motor que gira el portabrocas. Ten en cuenta que ambos circuitos están protegidos con el mismo magnetotérmico tripolar y, además, cada motor debe disponer de protección por relé térmico.



← Figura 10.59. Esquema de fuerza.

8. Realiza el cableado de fuerza, con cable de 2,5 mm², sobre el panel de pruebas.
9. Dibuja el esquema de mando con la conexión del autómata a los actuadores y captadores.



← Figura 10.60. Esquema de mando.

10. Realiza el cableado de mando, con cable de 1,5 mm², según el esquema que has dibujado anteriormente. Deja la entrada I2 del autómata reservada para una posible conexión del pulsador de parada.
11. Utilizando el ejemplo en lenguaje de contactos desarrollado en esta unidad para el taladro semiautomático, prográmalo y transfíreselo al autómata.
12. Conecta el panel de pruebas a la red eléctrica y acciona ambos magnetotérmicos.
13. Acciona el pulsador de marcha y simula la secuencia de funcionamiento del taladro, accionando manualmente los finales de carrera.

MUNDO TÉCNICO

Programación gráfica del GRAFCET

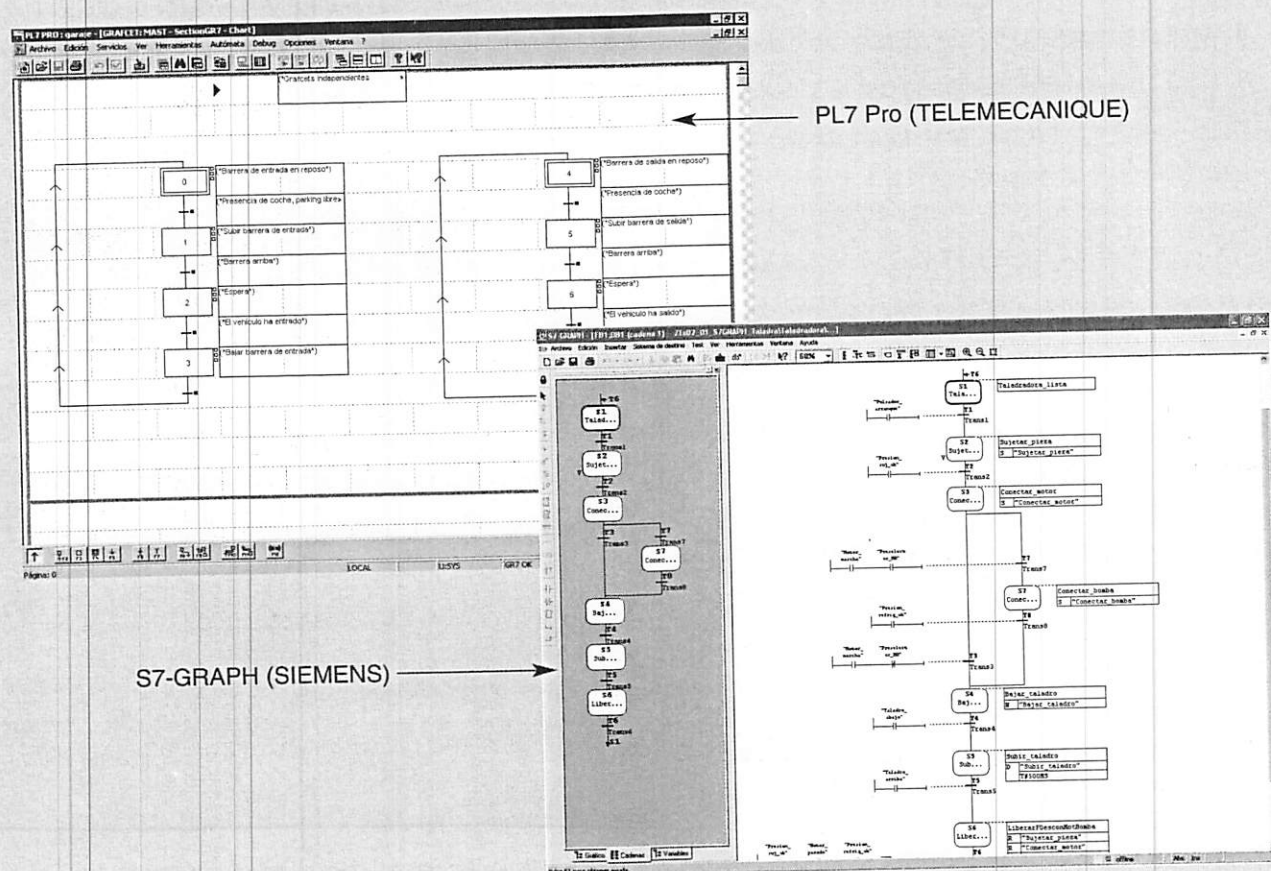
El GRAFCET es una excelente herramienta para el desarrollo de sistemas de control secuencial. Su implementación en sistemas cableados (basados en contactores y relés), neumáticos y/o programados (mediante PLCs o relés programables), facilita en gran medida las operaciones de automatización. El uso como método de desarrollo está muy difundido en el ámbito de los autómatas programables. Muchos técnicos automatistas lo utilizan para elaborar sus programas de programación estandarizados (de contactos, de lista de instrucciones o de funciones lógicas), agilizando así la puesta a punto de los sistemas secuenciales y facilitando la posterior ampliación de los mismos.

Sin embargo, a medida que la secuencia lógica del programa aumenta, mayor es el número de líneas de

código que el técnico tiene que desarrollar y por tanto, mayor es el tiempo empleado en escribirlas y, posteriormente, en localizarlas para la ampliación o búsqueda de fallos.

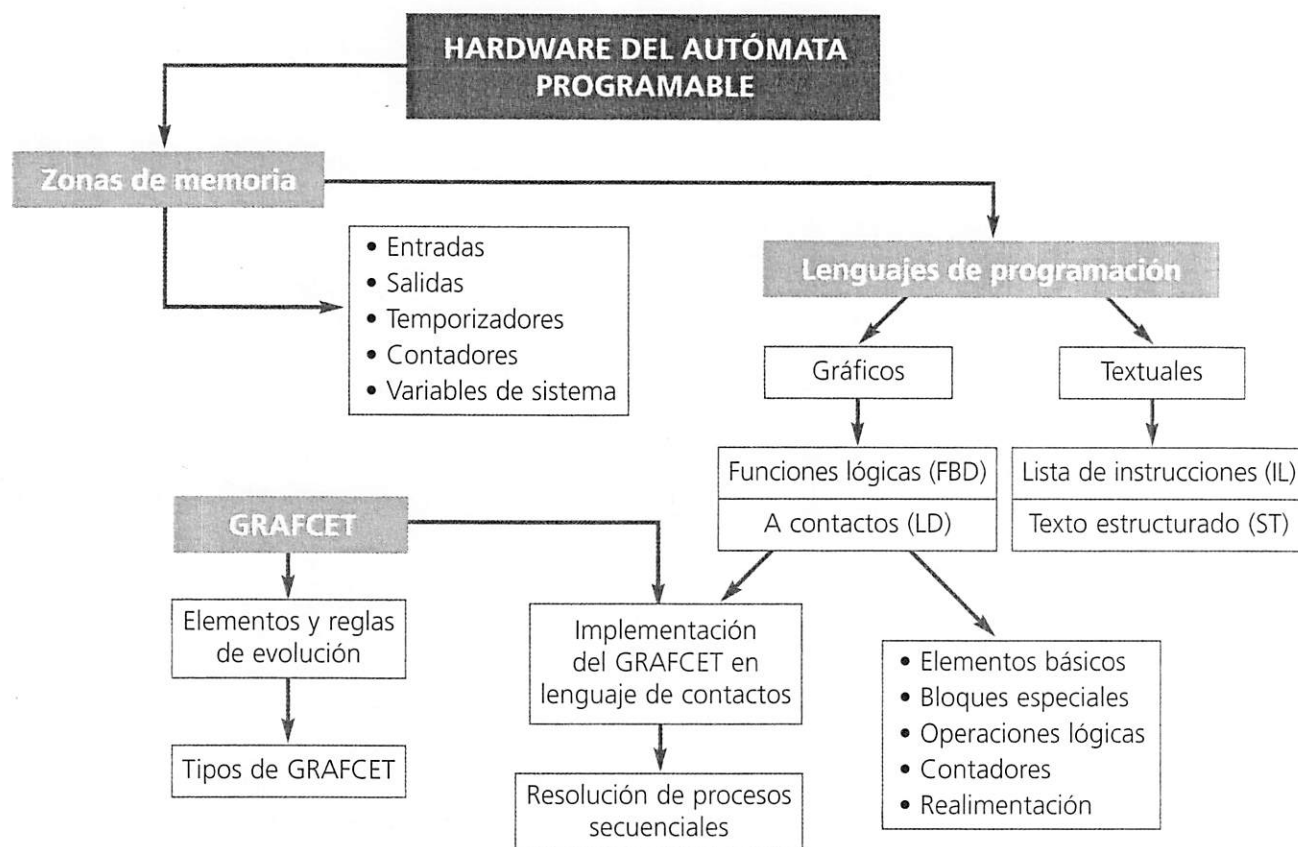
Algunos fabricantes de autómatas, desarrollan paquetes *software* que permiten la programación GRAFCET de forma gráfica sin necesidad de implementarlo en otros lenguajes de programación.

Estos programas disponen de herramientas para representar la secuencia mediante símbolos gráficos de las etapas y transiciones, pudiéndose programar los diferentes tipos de GRAFCET (de secuencia única, de secuencias opcionales y secuencias simultáneas), además de otros elementos avanzados como macroetapas, subrutinas GRAFCET, tareas de supervisión, etc.



↑ Figura 10.61. GRAFCET.

EN RESUMEN



EVALÚA TUS CONOCIMIENTOS

- La norma que regula los lenguajes de autómatas es:
 - La DIN 2345.
 - No hay norma para esto.
 - La UNE- EN 61131-3.
 - Simatic S7.
- La norma define los siguientes tipos de lenguajes:
 - 2 textuales.
 - 2 textuales y 1 gráfico.
 - 4 gráficos.
 - 2 textuales y 2 gráficos.
- El lenguaje por lista de instrucciones se denomina según norma: ____ y el de contactos: ____
- Una marca interna sirve:
 - Para realizar operaciones internas de la misma forma que los relés auxiliares en la lógica cableada.
 - No se utilizan, ya que hace la misma función que una salida.
 - Realiza operaciones de tiempo.
 - Almacena el cómputo de un contador.
- En la zona de acciones de un GRAFCET se define:
 - La secuencia que debe seguir el proceso.
 - Como se inicializa el GRAFCET.
 - Las salidas, temporizadores, contadores, etc que se deben activar en la secuencia.
- Cada etapa GRAFCET está asociada a:
 - Una salida.
 - Una marca interna.
 - Una entrada.
- La bobina SET se utiliza para ____ y la RESET para ____