CAPITULO

5

Instrucciones y programas en los Autómatas Programables

5.1. INTRODUCCION

En este capítulo vamos a tratar del *software* que, como sabemos, se refiere a los *programas* o partes no tangibles físicamente del Autómata. Si bien el software en su amplio término trata tanto de los programas creados por el usuario como los propios creados para el funcionamiento interno del Autómata, nosotros aquí nos vamos a referir a los primeros, ya que en el Capítulo 3 se ha hecho referencia a los segundos.

5.2. INSTRUCCIONES Y PROGRAMAS

Un programa es una sucesión o lista en un determinado orden de distintas *órdenes de trabajo* también llamadas *instrucciones* y capaz de hacer ejecutar al Autómata la secuencia de trabajo pretendida. La pregunta que vamos a tratar de responder, por tanto, va a ser: *¿Qué es una instrucción?*

Una instrucción u orden de trabajo es la parte más pequeña de un programa y consta de dos partes principales: **operación** y **operando;** a su vez el operando está dividido en *símbolo* y *parámetro.*

Instrucción		
Operación ¿Qué?	Operando ¿Dónde?	
	Símbolo	Parámetro

La **operación** es el *código (CODE) de la instrucción.* Puede venir dado como código numérico o cifrado (08) o código nemónico (AND).

El **operando** es el *complemento al código u operación.* Mediante el operando indicamos la dirección del elemento de que se trate (contadores, temporizadores, E/S, marcas internas, etc.), así como las contadas, temporizaciones, bits de registro de desplazamiento, etcétera.

En los autómatas más sencillos el *símbolo* no aparece, ya que al ser fijas las entradas/salidas y tener asignados números distintos unas y otras, al indicarle a la CPU el número, ya queda determinada, en el caso de las entradas la distinción con las marcas internas, y las salidas con el código de la operación.

La operación le indica a la CPU *qué tiene que hacer,* o lo que es lo mismo, la clase de instrucción que ha de ejecutar.

Ejemplo:

AND (Y): Formar una concatenación serie.
OR (O): Formar una concatenación paralelo.
OUT (=): Asignar una salida a lo precedente.

El operando le indica a la CPU *dónde debe de hacerlo,* o lo que es lo mismo, dónde debe realizarse esa instrucción.

Ejemplo: **E.2.1.** Realiza la *operación* anterior en la entrada (E) número 1 del módulo 2.

En este caso E representa el *símbolo* y 2.1 el *parámetro.*

En otros autómatas: **12.** Si este número corresponde a una entrada, y, por ejemplo, la *operación* ha sido AND, la *instrucción* queda determinada como asignación de contacto normalmente abierto a la entrada número 12.

Cuando se programa, cada instrucción del programa se aloja en una celda o plaza de memoria que están numeradas desde la dirección 0000 hasta el último número, en función de la capacidad de memoria; en el caso de una memoria de usuario de 1 K palabras, las direcciones disponibles serían de la 0000 a la 1023. Se ha supuesto que cada instrucción ocupa una palabra que, en general, es de 16 bits o 2 bytes, si la instrucción ocupa más de 2 bytes como ocurre en algunos casos, el número de direcciones disponibles se reduce.

Otro concepto a tener en cuenta es el de *línea o línea de programa.* Una línea contiene dirección o paso, operación y operando, por tanto, se puede decir que una línea de programa consta de una instrucción, salvo algunos casos en el que son necesarias dos líneas para alojar una sola instrucción.

Para poder elaborar un programa no es suficiente con las instrucciones de mando o de programa, son necesarias otro tipo de instrucciones que reciben el nombre de *instrucciones de servicio u órdenes de manejo* y por medio de las cuales se consigue la elaboración, análisis y puesta a punto del programa, así como otras posibilidades que en los ejemplos prácticos se verán.

En la Figura 5.1 se recoge de forma gráfica la estructura en que se ordenan ambos tipos de instrucciones.

5.3. EJECUCION DE PROGRAMAS

Como ya quedó dicho en el Capítulo 3, cuando el Autómata se sitúa en *ciclo de ejecución o ejecución cíclica,* la CPU realiza, entre otras funciones, el barrido del programa contenido

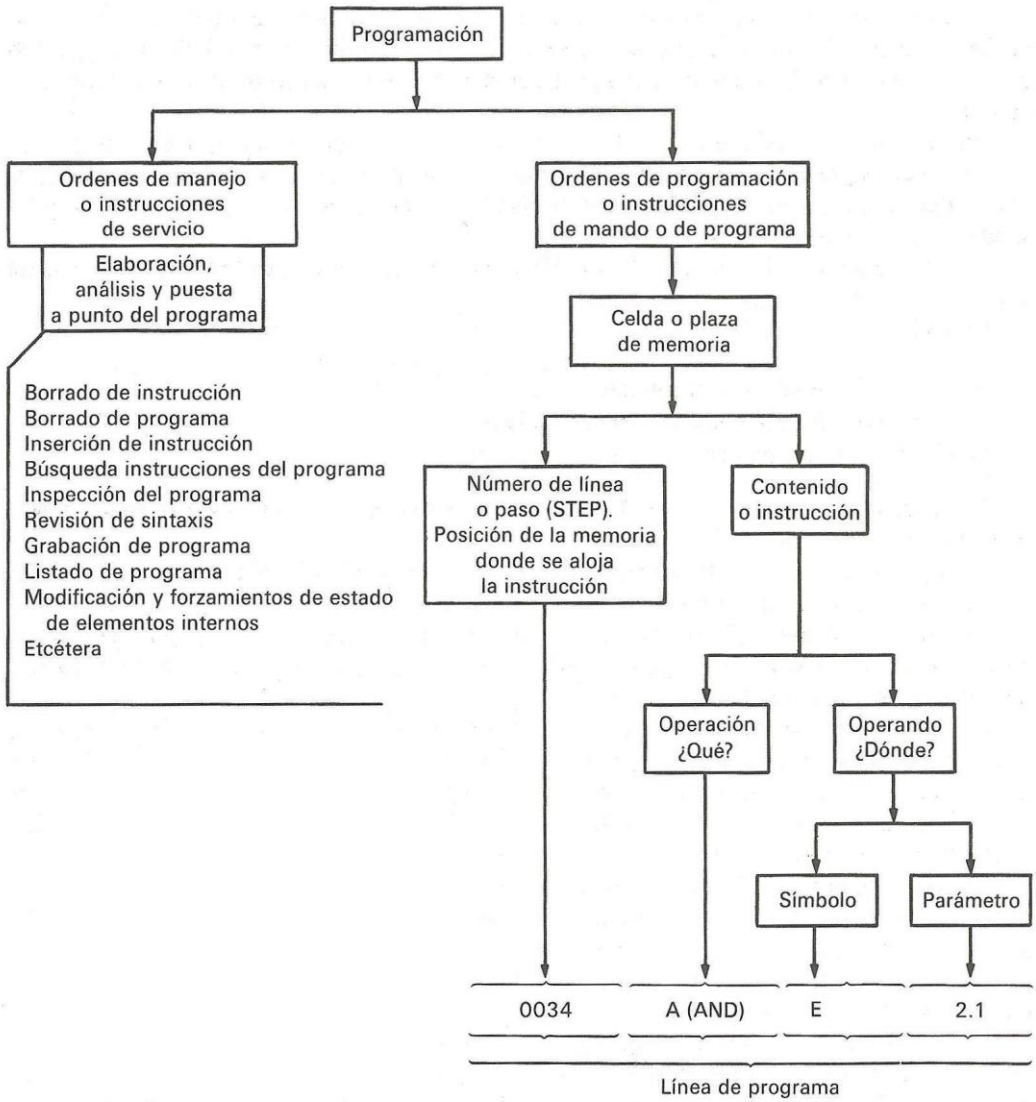


Figura 5.1. Estructura de las instrucciones u órdenes de manejo y programación de un Autómata Programable.

en la memoria de usuario, desde la casilla, dirección o línea 0000 hasta la última posible, según la capacidad de la misma, esto es, efectúa lo que se denomina *ciclo de Scanning.*

En función de cómo se efectúe la ejecución o barrido del programa, se distinguen los siguientes sistemas, modos o estructuras de programación:

- Ejecución cíclica lineal.
- Ejecución con salto condicional.

- Ejecución con salto a subrutinas.
- Ejecución con programas paralelos.

El principio de ejecución de cada uno de los sistemas indicados se describe a continuación.

5.3.1. Ejecución lineal

Cuando el ciclo de barrido de la memoria de usuario se realiza línea tras línea sin solución de continuidad, se dice que la *programación es lineal,* y la CPU consulta las instrucciones contenidas en la memoria secuencialmente, una a continuación de la otra, sin alterar este orden, tal y como se aprecia en la Figura 5.2.

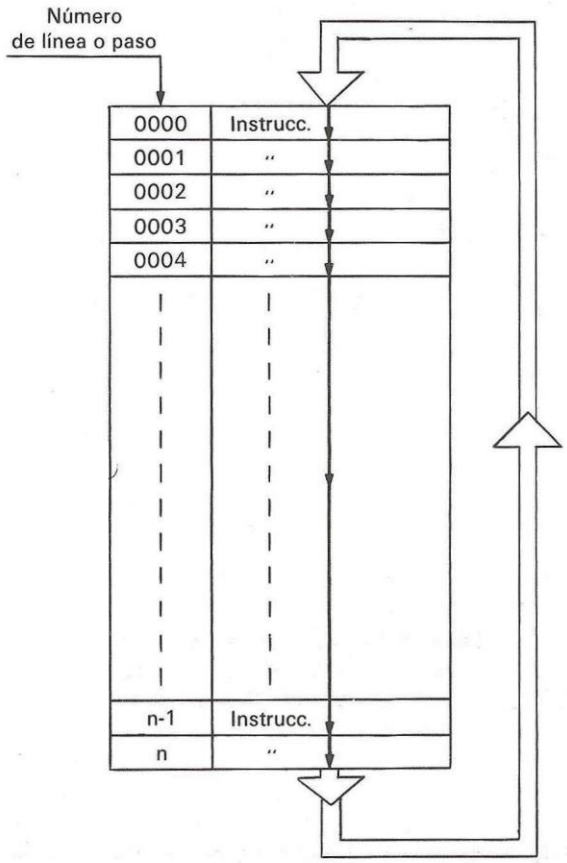


Figura 5.2. Ejecución cíclica lineal del programa de usuario en un Autómata Programable.

5.3.2. Salto condicional

Cuando el ciclo de ejecución de un programa tiene la posibilidad, previa condición establecida, de alterar la secuencia línea a línea del mismo y dar un salto a otras líneas de programa, dejando x líneas sin ejecutar, se dice que se ha realizado un salto condicional.

Su funcionamiento es el siguiente:

Si al llegar en el proceso de ejecución del programa a la instrucción *U*, tal y como queda descrito en la Figura 5.3, se cumple la condición en ella indicada, se salta a *V* continuando el barrido en *V*+1 hasta *n*. Si por el contrario al llegar a *U* no se cumple la condición, el programa se ejecuta linealmente continuando en *U*+1.

Esta posibilidad que poseen muchos Autómatas permite reducir el tiempo de ejecución del ciclo y es aplicable en aquellos casos en que las instrucciones contenidas en el salto sólo son necesarias cuando se dan ciertas condiciones impuestas por el programa.

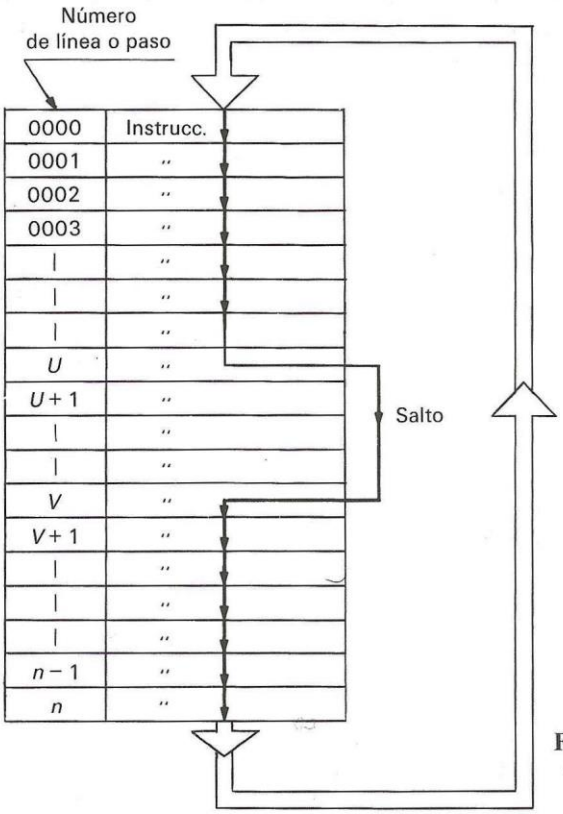


Figura 5.3. Salto condicional en la ejecución cíclica lineal del programa de usuario en un Autómata Programable.

5.3.3. Salto a subrutina

En algunas ocasiones ocurre que en un programa hay uno o más grupos de secuencias de instrucciones idénticas que se repiten y que habrá que reescribir tantas veces como éstas se repitan en dicho programa principal. En estos casos, es muy útil escribir una sola vez esta *secuencia o subrutina,* e ir a ella cuando se requiera. Los autómatas de la gama baja no suelen incorporar esta posibilidad. En la Figura 5.4 se puede observar gráficamente este tipo de salto. La posibilidad de distintos tipos de subrutinas y de distintos niveles de las mismas también quedan reflejados en la citada figura.

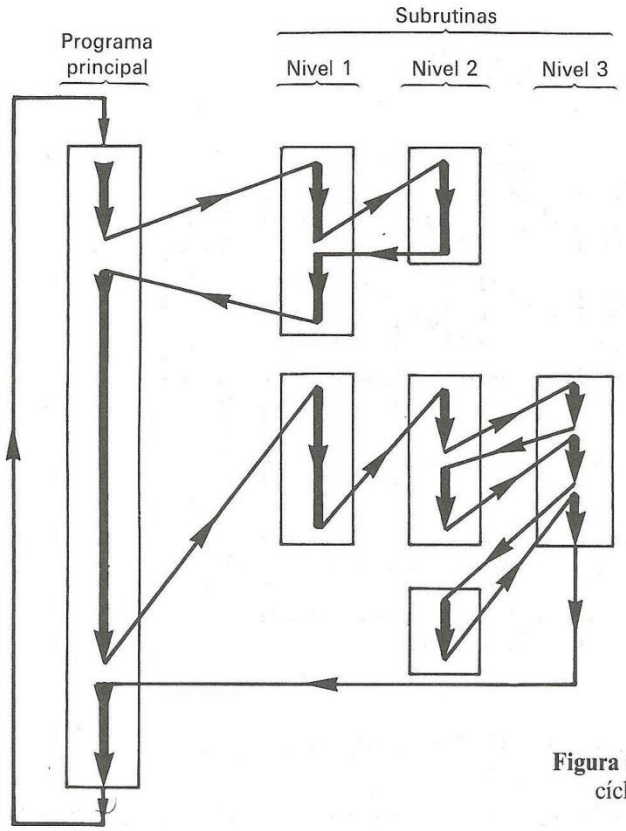


Figura 5.4. Salto a subrutina en la ejecución cíclica del programa de usuario en un Autómata Programable.

5.3.4. Programas paralelos

En este sistema, utilizado por algunos fabricantes y denominado de programas paralelos, el procesamiento se realiza paralelamente y de forma asíncrona. En aquellos casos en que con un único Autómata queremos controlar varios procesos totalmente independientes, este sistema es muy útil, aunque también se utiliza controlando funciones de un proceso único. En este tipo de ejecución es posible el uso de subrutinas en cada programa paralelo.

La ejecución de este tipo de programas se realiza de la siguiente forma:

Cada uno de los tramos en línea gruesa de la Figura 5.5 contiene sólo algunas líneas de programa, de tal forma que la secuencia consiste en el procesamiento de, por ejemplo, diez líneas del programa M0, o lo que es lo mismo, una pequeña parte de un programa; a continuación el barrido salta al programa M1 para procesar sus diez primeras líneas, pasando a continuación al M2 realizando el mismo proceso, etc. Cuando ha barrido todos los programas paralelos, incluso las subrutinas, si se encuentran adscritas a esas líneas, vuelve al programa M0 para repetir el ciclo en el siguiente grupo de diez líneas.

Para que la secuencia deseada en el barrido de programas se realice, ha de indicarse mediante la elaboración de un programa de asignación, tal y como se indica en la Figura 5.5.

Los Autómatas de la gama baja no suelen incorporar esta posibilidad.

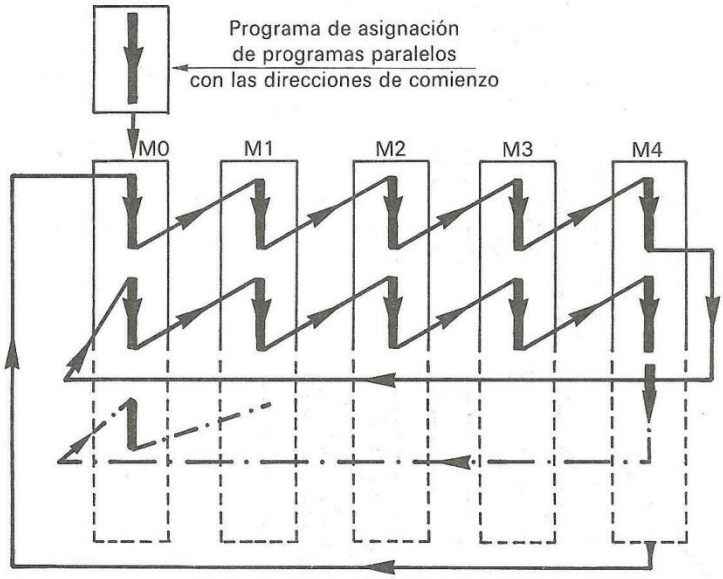


Figura 5.5. Ejecución de programas paralelos de usuario en un Autómata Programable.

5.4. SISTEMAS O LENGUAJES DE PROGRAMACION

Varios son los lenguajes o sistemas de programación posibles en los Autómatas Programables, aunque su utilización no se puede dar en todos los Autómatas; es por esto que cada fabricante indica en las características generales de su equipo el lenguaje o los lenguajes con los que puede operar. En general, se podría decir que los lenguajes de programación más usuales son *aquellos que transfieren directamente el esquema de contactos y las ecuaciones lógicas o los logigramas,* pero éstos no son los únicos.

A continuación figura una relación de los lenguajes y métodos gráficos más utilizados:

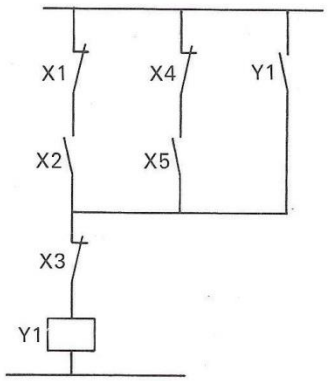
- **Nemónico,** también conocido como lista de instrucciones, booleano, abreviaturas nemotécnicas, AWL.
- **Diagrama de contactos** (Ladder diagram), plano de contactos, esquema de contactos, KOP.
- **Plano de funciones,** o bloques funcionales, logigrama, FUP.
- **Grafcet,** o diagrama funcional, diagrama de etapas o fases.
- **Organigrama,** u ordinograma, diagrama de flujo.

Excepto el *nemónico,* los demás tienen como base su representación gráfica, pero todos ellos deben ir acompañados del correspondiente *cuadro o lista de programación,* esto es, la relación de líneas de programa que configuran el mismo.

Para mejor entender estos lenguajes, a continuación se realiza una explicación de ellos. En el caso de los tres primeros, por otra parte los más utilizados, se ha puesto un ejemplo

de cada uno tomando como base un mismo circuito y partiendo de la ecuación lógica del mismo, de su esquema de relés y del circuito con puertas lógicas.

Definido un proceso simple, la ecuación lógica del mismo que a continuación figura, obtenida, por ejemplo, a partir de la tabla de verdad y previa minimización por medio de *Boole, Karnaught,* etc., nos es válida para dibujar el esquema de relés de la Figura 5.6, así como su implementación con puertas lógicas, según Figura 5.7.



Ecuación lógica:

$$[(\overline{X1}\cdot X2)+(\overline{X4}\cdot X5)+Y1]\,\overline{X3}=Y1$$

Ecuación:

$$[(\overline{X1}\cdot X2)+(\overline{X4}\cdot X5)+Y1]\,\overline{X3}=Y1$$

Figura 5.6. Esquema de relés o lógica cableada de la ecuación propuesta.

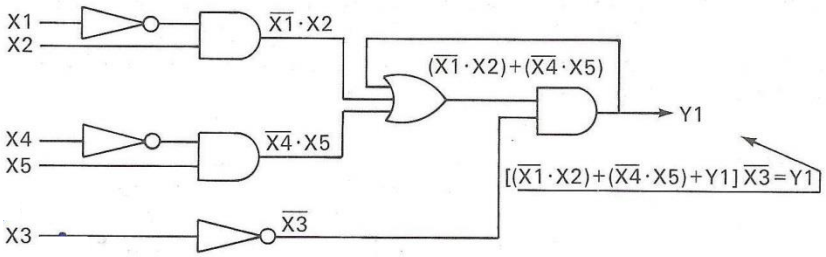


Figura 5.7. Implementación de la ecuación lógica propuesta.

5.4.1. Nemómicos o booleanos

Es un lenguaje en el cual cada instrucción se basa en las definiciones del *álgebra de Boole o álgebra lógica.* A continuación figura una relación de *nemónicos,* con indicación de lo que representan:

STR: Operación inicio contacto abierto.
STR NOT: Operación inicio contacto cerrado.
AND (Y): Contacto serie abierto.
OR (O): Contacto paralelo abierto.
AND NOT: Contacto serie cerrado.
OR NOT: Contacto paralelo cerrado.
OUT: Bobina de relé de salida.
TMR: Temporizador.

CNT: Contador.
MCS: Conexión de una función a un grupo de salidas.
MCR: Fin de la conexión del grupo de salidas.
SFR: Registro de desplazamiento.
Etcétera.

Los *nemónicos* de la ecuación propuesta serían los siguientes:

STR NOT.....X1
AND.....X2
STR NOT.....X4
AND.....X5
OR STR.....
OR.....Y1
AND NOT X3
OUT.....Y1

5.4.2. Diagrama de contactos

La mayoría de los fabricantes incorporan este lenguaje, ello es debido a la semejanza con los esquemas de relés utilizados en los automatismos eléctricos de lógica cableada, lo que facilita la labor a los técnicos habituados a trabajar con dichos automatismos. Si comparamos el *diagrama de contactos* de la Figura 5.8 *con el de relés* de la Figura 5.6, nos daremos cuenta de la similitud a la que aludíamos.

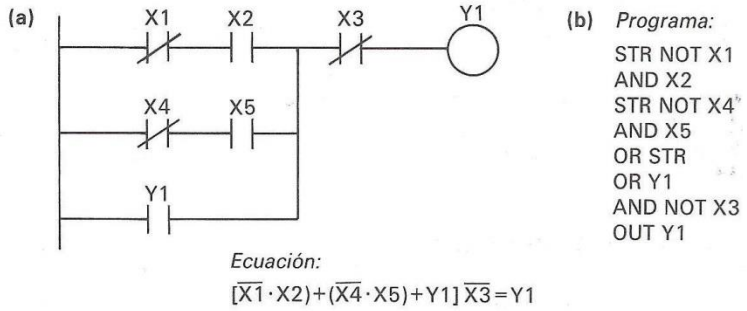


Figura 5.8. Esquema o diagrama de contactos de la ecuación propuesta (a). Programa con Autómata Programable (b).

5.4.3. Plano de funciones

Su semejanza con los *símbolos lógicos o puertas lógicas* hace también interesante este lenguaje por la facilidad en su representación para los conocedores de la electrónica lógica. En la Figura 5.9 aparece el esquema para la ecuación propuesta. Podemos observar su similitud con el esquema lógico de la Figura 5.7, si comparamos ambos.

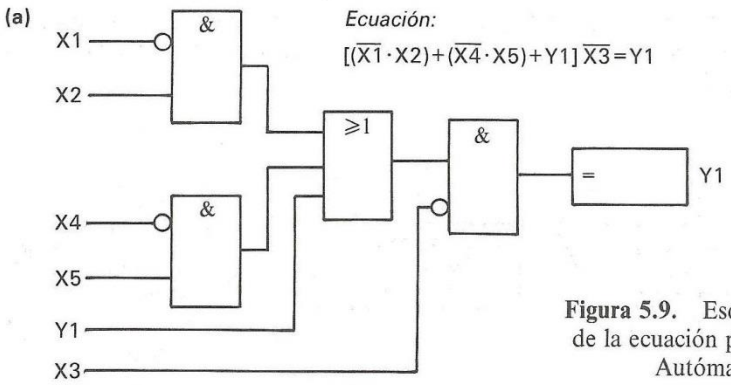


Figura 5.9. Esquema en plano de funciones de la ecuación propuesta (a). Programa con Autómata Programable (b).

5.4.4. GRAFCET

El GRAFCET, o *Graphe de Comande Etape Transition,* esto es, *Gráfico de Orden Etapa Transición,* es un método por el cual se describen en una forma gráfica perfectamente inteligible las especificaciones de cualquier automatismo. La Figura 5.10, nos da una idea simplificada de este sistema.

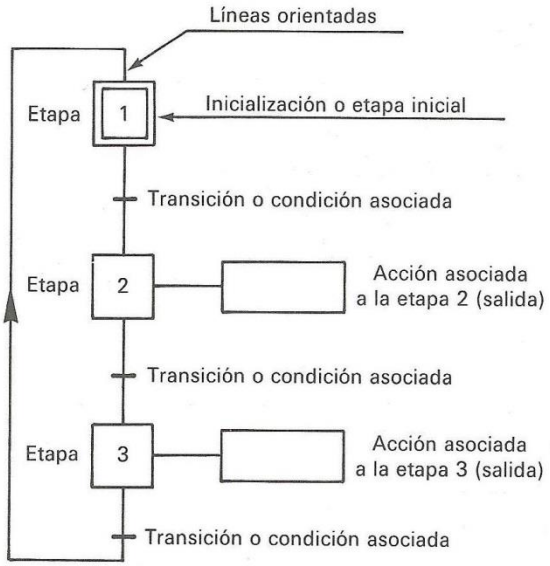
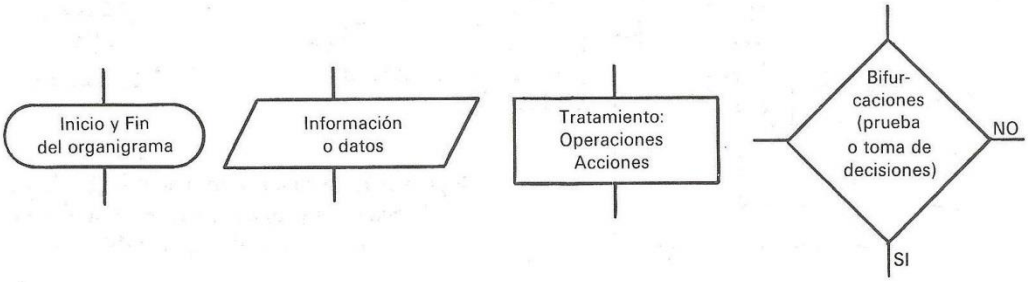


Figura 5.10. Gráfico correspondiente a la estructura del GRAFCET.

5.4.5. Organigrama

También llamado ordinograma, diagrama de flujo y flujograma, es un sistema de representación que se basa en una serie de figuras geométricas, utilizadas como símbolos, unidas

por líneas y que tiene como misión mostrar gráficamente, en nuestro caso, un proceso o problema, así como informar del mismo, esto es, analizar las partes y darles solución. Los símbolos utilizados son los siguientes:



Podríamos establecer dos tipos de organigramas: *organigrama de nivel 1,* en el cual se representan las acciones a desarrollar por el proceso acción por acción, y *organigrama de nivel 2 u organigrama de programación,* en el cual se sustituirán las designaciones del nivel 1 por las instrucciones correspondientes, para así poder realizar el programa en el autómata. La Figura 5.11 representa el primer caso. Dada su escasa utilización en programas de

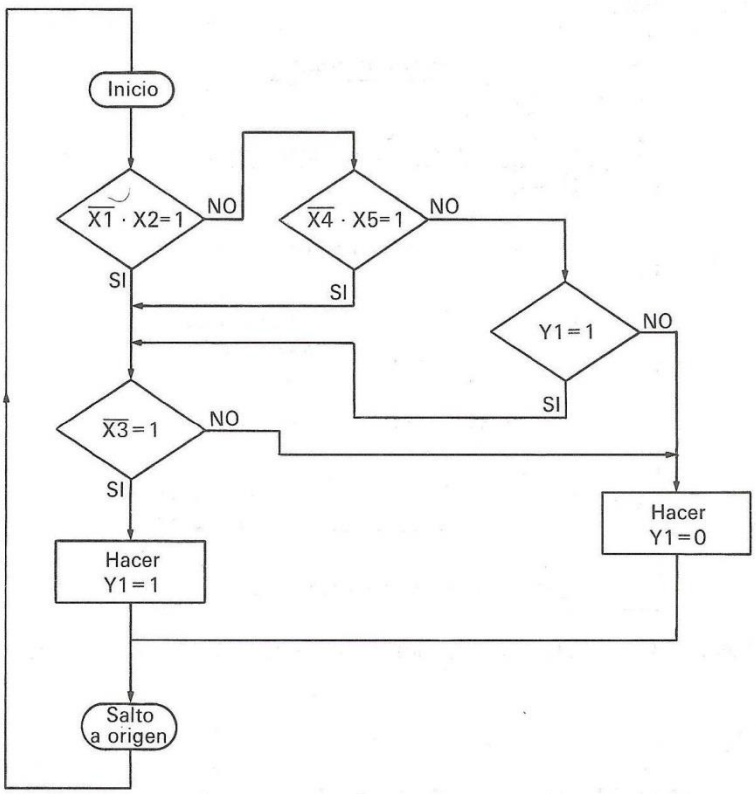


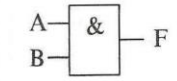
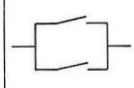
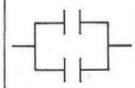
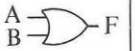
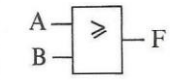
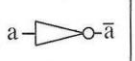
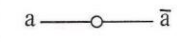
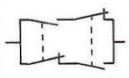
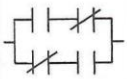
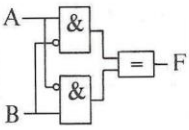
Figura 5.11. Organigrama de nivel 1 representativo de la ecuación lógica propuesta.

Autómatas Programables debido a que su complejidad es superior a la de los sistemas anteriormente descritos, se ha omitido el organigrama de nivel 2.

5.5. SIMBOLOGIA Y EQUIVALENCIAS

En el Cuadro 5.1 se representan los símbolos utilizados en los ejemplos del Apartado 5.4, su correspondencia entre sí y otros conocidos tales como los *nemónicos o de Boole.* Su estudio es interesante dada su normal utilización cuando se trabaja con Autómatas Programables.

Cuadro 5.1. Símbolos utilizados en esquemas de trabajo con Autómatas Programables y sus equivalencias

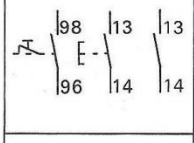
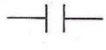
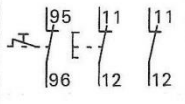
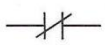
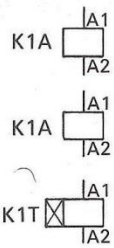
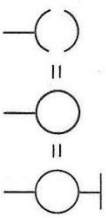
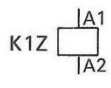
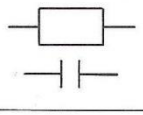
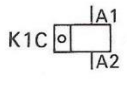
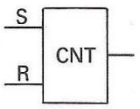
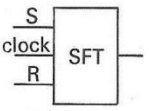
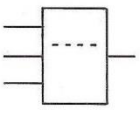
Norma / Función	Nemónicos	Boole	DIN-40713-6 (relés)	NEMA (contactos)	Símbolos lógicos	Operadores lógicos UNE-20-004-75 (XVI)
Y (Serie)	AND	•				
O (Paralelo)	OR	+				
Complementaria	NOT	$\overline{a}$				
Exclusiva	XOR	⊕				

En el Cuadro 5.2 se han representado las equivalencias específicas entre símbolos de lógica cableada y los normalmente utilizados en diagrama de contactos.

5.6. ASIGNACIONES AL PROGRAMA

La realización de una buena programación pasa, entre otras cosas, por diseñar correctamente el diagrama correspondiente. Para no incurrir en errores tales como asignaciones repetidas o asignaciones de elementos inexistentes, es conveniente confeccionar un cuadro

Cuadro 5.2. Equivalencias entre símbolos utilizados en lógica cableada y los normalmente utilizados como lógica programada en diagrama de contactos

Lógica cableada	Autómatas. Diagramas de contactos	Designación	Observaciones
		Contactos en general: • De entrada al autómata • De relés de salida • De relés auxiliares • De relés especiales • De temporizadores • De contadores • De registros de desplaz. • De pulsadores • De relés térmicos • Etcétera	El reconocimiento de la pertenencia del contacto se realiza tanto por el número que lo acompaña como por su designación: • Sólo núm.: ver núms. asignados a relés y entradas • Temporizadores: Designación del mismo y número • Contadores: Idem • Registros de desplaz.: Idem
			
		Bobinas: Bobinas de relés en general, de salida, auxiliares, con temporizadores, etc.	Los tres símbolos de la columna de diagramas de contactos se utilizan indistintamente como equivalentes a cualquiera de las bobinas de relés.
		Relés especiales: Tales como generadores de impulsos, osciladores, etc.	
		Contadores: BCD, UP/DOWN, etc.	S = Set = actuación R = Reset = puesta a cero
		Registros de desplazamiento: De cualquier tipo	Clock = reloj = impulsos o secuencia de impulsos
		Otras funciones: • Comparadores • Biestables • Etcétera	

similar al representado como Cuadro 5.3, que se irá completando conforme vayamos desarrollando el diagrama, o mejor aún, si es posible antes de confeccionar.

Es interesante desde un punto de vista práctico tener delante la relación de elementos del Autómata, tal y como aparece en el Cuadro 5.4.

Cuadro 5.3. Ejemplo de asignaciones al programa

Tipo	Terminal asignado	Descripción
Entradas (captadores)	0000	Pulsador de marcha
	0001	Pulsador de paro
	0002	Relé térmico F1
	⋮	⋮
	n	⋮
Salidas (actuadores)	0050	Contactor de línea
	0051	Contactor estrella (λ)
	0052	Contactor triángulo (Δ)
	⋮	⋮
	n	⋮

Tipo	Número	Descripción
Relés auxiliares	205	Marcha contador 1
Relés especiales	316	Pues. 0 reg. 1 en 1.^{er} ciclo
Tempo-rizadores	00	Esp. 5 seg. mar. motor M1
Contadores	01	Contaje cajas 1.ª cadena
Registros desplaz.	00	Contaje piez. y señal cont.
Otros		

Cuadro 5.4. Elementos del Autómata Sestep 364 de Sprecher+Schuh

Elementos	Números	
Entradas (40)	00 a 07; 10 a 17; 20 a 27; 30 a 37 y 40 a 47	
Relés de salida (24)	50 a 57; 60 a 67 y 70 a 77	
Relés auxiliares o Marcas (64 + 13)	sin protec.	200-207; 210-217; 220-227; 230-237; 240-247;
		250-257; 260-267 y 270-277.
	c/protec.	300-307 y 310-314.
Relés especiales (3)	315, 316 y 317.	
Otros relés		
Temporizadores (16)	TMR 00 a 07 y 10 a 17	
Contadores (16)	CNT 00 a 07 y 10 a 17	
Registros de desplaz. (2)	SFR 00 (00-07); SFR 10 (10-17)	
Otros		

RESUMEN

- Dos son los tipos de instrucciones necesarios para la grabación de programa en el Autómata: instrucciones de servicio e instrucciones de mando o de programa.
- Generalmente la longitud de la palabra que define una instrucción es de 16 bits.
- Las instrucciones de servicio son necesarias para la elaboración, análisis y puesta a punto del programa.
- Se entiende por *salto condicional,* la posibilidad, previa condición establecida, de alterar la secuencia del programa dando un salto en el barrido de un determinado número de líneas del programa.
- Los esquemas en diagrama de contactos son similares a los esquemas de relés en la lógica cableada.
- Los planos de funciones son equivalentes a los representados mediante el álgebra lógica o de Boole.
- El diagrama de contactos es el lenguaje más universalmente utilizado y lo incorporan prácticamente todos los fabricantes de autómatas.
- El GRAFCET es un método por medio del cual se describen de una forma gráfica las especificaciones de cualquier automatismo.
- Es necesario al realizar un programa asignar los elementos mediante la anotación de los mismos teniendo en cuenta su numeración.

CUESTIONES

1. Dar una explicación satisfactoria de los conceptos de hardware y software.
2. ¿Qué es una instrucción y en qué partes se divide?
3. ¿En qué formas puede venir dado el código de una instrucción?
4. ¿Qué se entiende por programa?
5. Indicar las partes de que consta una línea de programa.
6. ¿Qué tipos de instrucciones existen y cuál es su misión?
7. Establecer la diferencia en la ejecución de programas entre salto condicional y salto a subrutina.
8. ¿En qué orden se procesan los programas paralelos?
9. Explicar brevemente con un ejemplo los lenguajes de programación: diagrama de contactos y plano de funciones.
10. ¿Qué se entiende por asignaciones al programa?