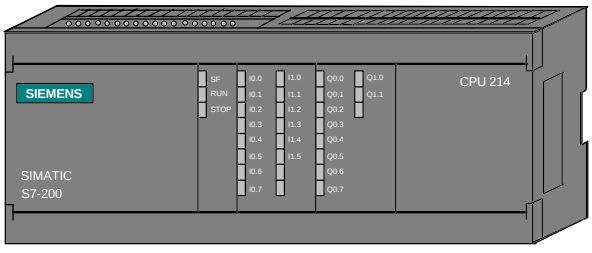
PLC Simatic S7



Programar en lugar de cablear

Lógica cableada

Tarea de mando

Un mando con lógica cableada es un automatismo con contactores y relés que solo se configura una vez conocida la tarea que debe resolver. Para ello, hasta ahora la tarea de mando se representaba con ayuda de un esquema eléctrico.

Seguidamente ha ido montando en un armario o cofre los diferentes elementos tales como contactores, relés, relés de tiempo, etc.; estos se enlazaban con cable siguiendo una lista de cableado fija.

Al interconectarlos ha fijado la función de los elementos de conmutación en el mando; por ejemplo ha conectado en serie o paralelo los contactos normalmente cerrados o normalmente abiertos, respectivamente.

La lógica de su función de mando está fijada en el cableado y en la combinación de los elementos de combinación. Para probar el mando es necesario verificar la corrección del cableado.

¿Error,
conexión errónea?

¡Qué contrariedad! Esto significa soltar el cableado y volver a inerconectar los elementos.

Nuevo mando,
misma tarea

¿Precisa nuevamente el mismo mando?. En este caso deberá comenzar completamente desde el principio; es decir, montar los aparatos en el armario, cablearlos de acuerdo a la lista correspondiente y comprobar la configuración.

Ampliación

¿Desea modificar más adelante la función del mando?. Esto significa añadir nuevos componentes, cambiar cableados y trabajos de montaje. Esto le llevará gran cantidad de tiempo y material.

Lógica programable (P.L.C.)

Tarea de mando

Para un mando con lógica programable (P.L.C.) se utiliza un autómata programable (AG.). Este está compuesto por:

- Fuente de alimentación.
- Entradas y salidas digitales en las que se conectan los emisores y actuadores.
- Una memoria en la que se escribe el programa a ejecutar.
- Un procesador que organiza la ejecución del programa.

Los emisores y actuadores se conectan con independencia de la tarea planteada, a las entradas y salidas de su AG., ¡esto es todo el cableado!.

Qué actuadores deberán ser activados por qué emisores se fija en el programa. En él se especifica la función del circuito de mando. El programa se entra usando un aparato de programación (PG) desde el que se transmite al AG; es decir, se escribe en él. El procesador en el AG ejecuta el programa paso a paso. Así pues, en un PLC, la lógica de la tarea de mando queda fijada en el programa. Con él se especifica cuándo deben conectarse o desconectarse los actuadores.

Así pues, la tarea de mando se programa en lugar de cablearla.

¿Ha cometido un error?

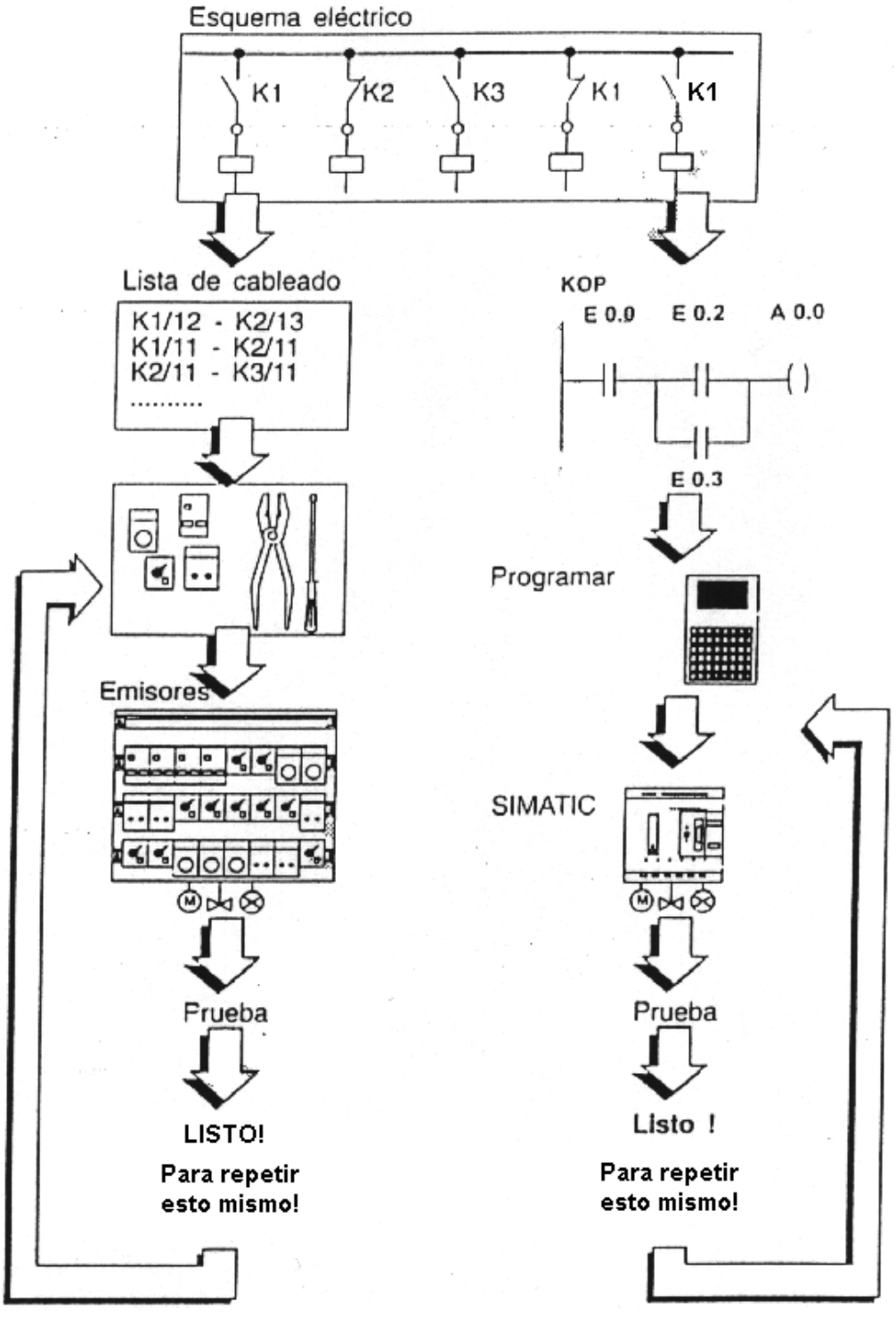
Antes de arrancar el autómata es posible comprobar - con ayuda del simulador y del PG - la ausencia de errores en el programa. Si aparece un error, basta con modificar la instrucción correspondiente dentro del programa. Esto es todo.

Nuevo mando, misma tarea

Un programa ya confeccionado puede usarse todas las veces que se desee; esto proporciona un ahorro considerable de gastos y tiempo.

Ampliación

¿Desea modificar a posteriori la tarea del mando? Para ello basta con modificar el programa. No es necesario desembornar las entradas y salidas ya conectadas, es decir, el cableado. Tampoco el programa deberá rehacerse totalmente, porque siempre es posible cambiar, borrar o insertar determinadas partes del mismo o solo instrucciones individuales. Es decir, cualquier cambio o ampliación se realiza de forma rápida y simple.


Esquema eléctrico
K1
K2
K3
K1
K1
Lista de cableado
K1/12 - K2/13
K1/11 - K2/11
K2/11 - K3/11
.....
KOP
E 0.0
E 0.2
A 0.0
E 0.3
Programar
Emisores
SIMATIC
Prueba
Prueba
LISTO!
Para repetir
esto mismo!
Listo !
Para repetir
esto mismo!

¿Qué significa programar?

En un esquema eléctrico las combinaciones lógicas de las entradas y salidas se materializan usando contactos NA y NC. En cambio, un autómata consulta las entradas para ver qué estado de señal tienen; es decir, si hay tensión aplicada o no en ellas. Para poder decir al P.L.C. lo que debe hacer es preciso aprender el lenguaje de programación adecuado. Nada más fácil que ello.

¿Qué es un lenguaje de programación?

Con un lenguaje de programación ocurre lo mismo que con cualquier idioma, en él se especifican las palabras (en este caso se denominan instrucciones), la ortografía y la gramática. Usando instrucciones se escribe un programa que se deposita en la memoria del P.L.C. Este va ejecutando el programa paso a paso: al llegar a su fin comienza nuevamente desde el principio. Así, el P.L.C. sabe lo que tiene que hacer.
Dependiendo del programa el P.L.C. conecta y desconecta los actuadores. Los estados <<CON>> y <<DES>> son unívocamente diferenciables y se describen con los conceptos siguientes

Estado “0” = tensión no presente = DES
Estado “1” = tensión presente = CON

Una señal cuyo estado queda definido exclusivamente por dos valores constituye una señal binaria y se designa como bit (bit = Binary Digit).

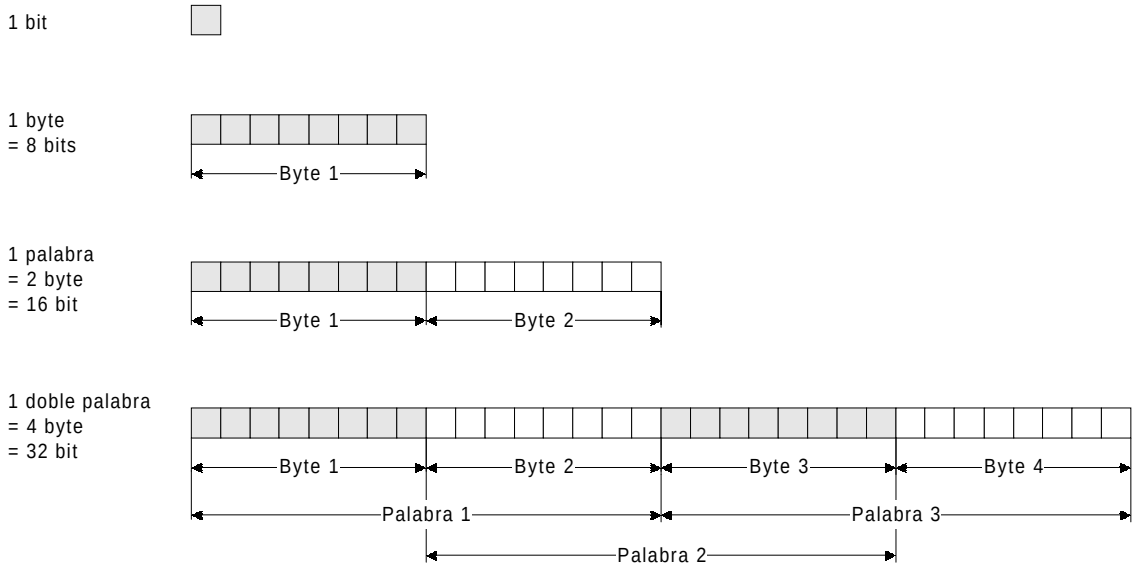
Bit, Byte, palabra, doble palabra.

El Bit es la unidad de una señal binaria. 1 bit es la menor unidad de información y puede adoptar los estados “1” ó “0”.

Un Byte está formado por 8 caracteres binarios sucesivos. Así pues, un byte tiene una longitud de 8 bits. En un P.L.C. esto permite agrupar en un byte de entrada (EB), un byte de salida (AB) los estados de señal de 8 entradas o 8 salidas. De la misma manera que para las entradas/salidas, se hablará de byte de marca interna (MB) ó de byte de memoria especial (VB).

Si se agrupan 2 byte - es decir, 16 bit - formando una unidad, entonces las 16 posiciones binarias forman una palabra. En el P.L.C. los estados de señal de 16 entradas o 16 salidas se agrupan en una palabra de entrada (EW), una palabra de salida (AW), una palabra de marca interna (MW), ó en una palabra de memoria variable (VW).

Si finalmente agrupamos 2 palabras, obtenemos una doble palabra que estará formada por 32 bits. Los P.L.C. ´s mas potentes permiten trabajar con dobles palabras de entradas (ED), dobles palabras de salidas (AD), dobles palabras de marcas internas (MW), ó dobles palabras de memoria de variables (VW).



Sistemas numéricos

Los sistemas digitales actúan bajo el control de variables discretas, entendiéndose por éstas, las variables que pueden tomar un número finito de valores. Por ser de fácil realización los componentes físicos con dos estados diferenciados, es éste el número de valores utilizado usualmente para dichas variables que, por tanto, son binarias.

Tanto si se utilizan en proceso de datos como en control industrial, los sistemas digitales han de realizar operaciones con números discretos. Los números pueden representarse en diversos sistemas de numeración, que se diferencian por su base.

La base de un sistema de numeración es el número de símbolos distintos utilizados para la representación de las cantidades en el mismo. El sistema de numeración utilizado en los cálculos habituales es el de base diez, en el cual existen diez símbolos distintos, del 0 al 9.

Sistemas de numeración decimal

Necesita para la representación de valores numéricos (números) 10 cifras, es decir, las cifras comprendidas entre 0 y 9. Con estas cifras se forman los números en el sistema de numeración decimal. Las cifras se encuentran en los números una tras otra, con la misma disposición que las letras en las palabras.

Las cifras solas no son suficientes. En la disposición de las cifras, para formar los números, es importante la situación que cada una ocupa. Según la posición dentro del número, cada cifra tiene diferente valor, el llamado **valor de posición.** Estos valores de posición son potencias de 10 en el sistema decimal. Al 10 se le llama, por tanto, la base del sistema de numeración decimal. La posición de la cifra dentro del número nos indica el valor de posición. La suma de todos los productos de cifras y valores de posición nos dá el valor numérico.

Este sistema para valora los números se llama sistema de valoración por posición. Se pueden formar sistemas de numeración con cualquier base. Los sistemas de numeración más conocidos son el sistema binario (base 2), el sistema octal (base 8) y el sistema hexadecimal (base 16).

Sistema de numeración binario

El sistema de numeración binario es especialmente apropiado para la representación de números con la ayuda de aparatos electrónicos (por ejemplo autómatas). La base de este sistema de numeración es 2. Tiene, por tanto, solamente dos cifras, 0 y 1.

Estas cifras se pueden representar de forma simple, por medio de estados, por ejemplo "tensión disponible".

Si se tratan estos estados de forma individual y sin valoración de sus magnitudes, se hablará de *valores binarios*, por ejemplo estado de señal "0" y estado de señal "1".

Conteo binario y decimal

	conteo binario						***conteo binario***				
conteo decimal	**16**	**8**	**4**	**2**	**1**	***conteo decimal***	**16**	**8**	**4**	**2**	**1**
0	0	0	0	0	0	10	0	1	0	1	0
1	0	0	0	0	1	11	0	1	0	1	1
2	0	0	0	1	0	12	0	1	1	0	0
3	0	0	0	1	1	13	0	1	1	0	1
4	0	0	1	0	0	14	0	1	1	1	0
5	0	0	1	0	1	15	0	1	1	1	1
6	0	0	1	1	0	16	1	0	0	0	0
7	0	0	1	1	1	17	1	0	0	0	1
8	0	1	0	0	0	18	1	0	0	1	0
9	0	1	0	0	1	19	1	0	0	1	1

- **Ejemplo: sistema de valoración de posición.**

  El valor de posición del número 1024 se determina de acuerdo con los siguientes sumandos:

  1 0 2 4

  $1 \times 10^3 + 0 \times 10^2 + 2 \times 10^1 + 4 \times 10^0$

- **Ejemplo: valor decimal de un número binario.**

  El valor decimal del número binario 1011, se calcula de la siguiente forma:

  1 0 1 1

  $1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

Se habla de “valores digitales” cuando hay que valorar las cifras 0 y 1. El sistema de numeración binario está constituido de forma semejante al sistema de numeración decimal. El “valor de posición” de las cifras está determinado por las potencias de la base del sistema de numeración, es decir, las potencias de 2. Las cifras indican si el “valor de posición” existe (cifra 1), o si no existe (cifra 0).

Sistema de numeración hexadecimal

Cuando se trate de valores binarios grandes, con sólo las cifras 0 y 1, su escritura es muy engorrosa. Por ello, cuando se trate de valores binarios grandes, se emplean los signos del sistema de representación hexadecimal.

El sistema de numeración hexadecimal es un sistema de numeración con base 16.
En la siguiente figura se encuentra una tabla de correspondencias donde se encuentras las cifras que utiliza el sistema hexadecimal y los valores decimales y binarios correspondientes.

Hexadecimal	Decimal	Binario
0	0	0 0 0 0
1	1	0 0 0 1
2	2	0 0 1 0
3	3	0 0 1 1
4	4	0 1 0 0
5	5	0 1 0 1
6	6	0 1 1 0
7	7	0 1 1 1
8	8	1 0 0 0
9	9	1 0 0 1
A	10	1 0 1 0
B	11	1 0 1 1
C	12	1 1 0 0
D	13	1 1 0 1
E	14	1 1 1 0
F	15	1 1 1 1
10	16	1 0 0 0 0
11	17	1 0 0 0 1
.	.	.
.	.	.
.	.	.

- **Ejemplo de representación del número 1FA3H en binario**.

1	F	A	3	hexadecimal
0001	1111	1010	1001	binario

- **Ejemplo de representación del número binario 1001 0010 1111 en hexadecimal.**

1001	0010	1111	binario
9	2	F	Hexadecimal

Código BCD. Conversión a decimal.

Los números hexadecimales son, sin embargo, sólo una ayuda para la representación de números binarios. Para extraer de un valor (decimal) el correspondiente número binario, hay que referirse, en particular para números grandes, a tablas de conversión. El camino contrario, la construcción de un número binario, de tal manera que se pueda leer en él directamente un valor decimal, es, sin embargo, más sencillo. Para ello se utiliza el llamado código BCD (Binary Coded decimal Code = código binario para cifras decimales).

En un número binario, codificado en BCD, se mantiene el valor de posición de los números decimales (potencias de base 10). Aunque, las cifras del número decimal se representan en binario.

Para los números codificados en BCD no se necesitan todas las posibilidades de la tétrada del número binario. Los valores (decimales) 10 a 15 (= cifras hexadecimales A a la F) no aparecen en esta representación. Por esto se llama a las tétradas que representan estos valores en código BCD “pseudotétradas”.

Conteo decimal y conteo binario codificado en BCD

Decimal	BCD	Decimal	BCD
0	0 0 0 0 0 0 0 0	10	0 0 0 1 0 0 0 0
1	0 0 0 0 0 0 0 1	11	0 0 0 1 0 0 0 1
2	0 0 0 0 0 0 1 0	12	0 0 0 1 0 0 1 0
3	0 0 0 0 0 0 1 1	13	0 0 0 1 0 0 1 1
4	0 0 0 0 0 1 0 0	14	0 0 0 1 0 1 0 0
5	0 0 0 0 0 1 0 1	.	
6	0 0 0 0 0 1 1 0	.	
7	0 0 0 0 0 1 1 1	20	0 0 1 0 0 0 0 0
8	0 0 0 0 1 0 0 0	30	0 0 1 1 0 0 0 0
9	0 0 0 0 1 0 0 1	40	0 1 0 0 0 0 0 0

Códigos alfanuméricos ASCII

Se han usado 1 y 0 binarios para representar diferentes números. Los bit pueden ser codificados también para representar letras del alfabeto, números y signos de puntuación. Uno de estos códigos, de 7 bit, es el *American Standard Code for Information Interchange* (ASCII), se muestra en la figura siguiente. Note que la letra A se representa como 1000001 en tanto que B es 1000010. El código ASCII se usa ampliamente en las computadoras pequeñas para traducir de los caracteres del teclado al lenguaje de la computadora.

Los códigos que pueden representar letras y números son llamados *códigos alfanuméricos*.

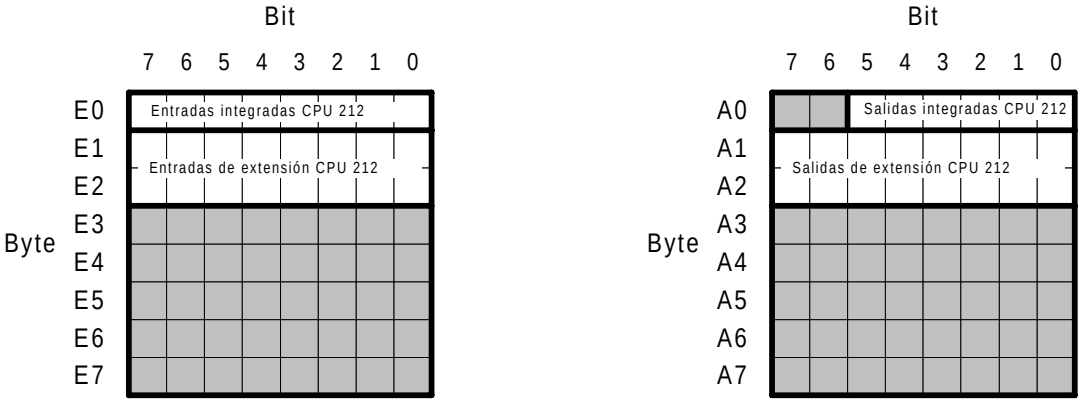
Carácter	ASCII	Carácter	ASCII
Espacio	0 1 0 0 0 0 0	A	1 0 0 0 0 0 1
!	0 1 0 0 0 0 1	B	1 0 0 0 0 1 0
“	0 1 0 0 0 1 0	C	1 0 0 0 0 1 1
#	0 1 0 0 0 1 1	D	1 0 0 0 1 0 0
$	0 1 0 0 1 0 0	E	1 0 0 0 1 0 1
%	0 1 0 0 1 0 1	F	1 0 0 0 1 1 0
&	0 1 0 0 1 1 0	G	1 0 0 0 1 1 1
´	0 1 0 0 1 1 1	H	1 0 0 1 0 0 0
(	0 1 0 1 0 0 0	I	1 0 0 1 0 0 1
)	0 1 0 1 0 0 1	J	1 0 0 1 0 1 0
*	0 1 0 1 0 1 0	K	1 0 0 1 0 1 1
+	0 1 0 1 0 1 1	L	1 0 0 1 1 0 0
,	0 1 0 1 1 0 0	M	1 0 0 1 1 0 1
-	0 1 0 1 1 0 1	N	1 0 0 1 1 1 0
.	0 1 0 1 1 1 0	O	1 0 0 1 1 1 1
/	0 1 0 1 1 1 1	P	1 0 1 0 0 0 0
0	0 1 1 0 0 0 0	Q	1 0 1 0 0 0 1
1	0 1 1 0 0 0 1	R	1 0 1 0 0 1 0
2	0 1 1 0 0 1 0	S	1 0 1 0 0 1 1
3	0 1 1 0 0 1 1	T	1 0 1 0 1 0 0
4	0 1 1 0 1 0 0	U	1 0 1 0 1 0 1
5	0 1 1 0 1 0 1	V	1 0 1 0 1 1 0
6	0 1 1 0 1 1 0	W	1 0 1 0 1 1 1
7	0 1 1 0 1 1 1	X	1 0 1 1 0 0 0
8	0 1 1 1 0 0 0	Y	1 0 1 1 0 0 1
9	0 1 1 1 0 0 1	Z	1 0 1 1 0 1 0

Áreas de E/S

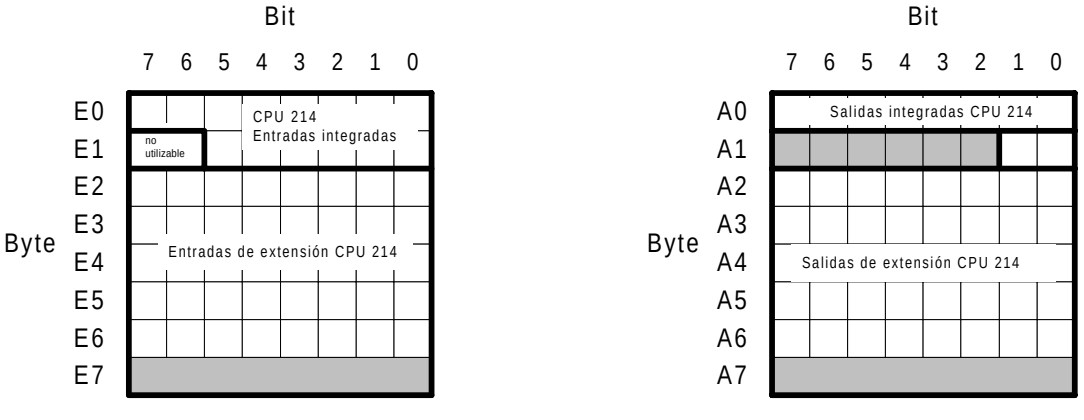
Al principio del ciclo se leen los valores actuales de las entradas y luego se escriben en la imagen de proceso (parte de la memoria de datos). Puesto que la versión CPU 214 dispone de 14 entradas no se leen las entradas E1.6 y E1.7. Estas dos entradas se ponen a cero en la imagen de proceso cada vez que se actualizan las entradas. En la siguiente figura se muestra el área de E/S de las CPU 212 Y CPU 214.

La CPU 212 dispone de 8 entradas (E0.0 a E0.7) y 6 salidas (A0.0 a A0.5). Sólo se utilizan seis de los ocho bits del primer byte de salida. Los bits no utilizados se almacenan en la imagen de proceso. Algunos de ellos se requieren solamente en caso de conectar módulos de extensión a la CPU 212. Los bytes E3 a E7 y A3 a A7 no se pueden emplear como entradas y salidas físicas, pero pueden servir de marcas internas. Del mismo modo se puede usar como marca interna cualquier bit de los módulos de extensión que no se utilice.

La versión CPU 214 es muy parecida al CPU 212, a diferencia de que la CPU 214 dispone de más entradas y salidas integradas (en el aparato central). Además puede conectársele un mayor número de módulos de extensión.



Nota: El área sombreado y el área extendida pueden utilizarse como marcas internas



Nota: El área sombreado y el área extendida pueden utilizarse como marcas internas

Direccionamiento en la CPU 212

CPU 212 *8ED/6AD*	EM 221 *8 ED*	EM 222 *8AD*
EB_ AB_	EB_	AB_

Direccionamiento en la CPU 214

CPU 214 *14ED/10AD*	EM 221 *8 ED*	EM 222 *8AD*	EM 231 *3EA*	EM 235 *3EA/1AA*	EM 223 *4ED/4AD*	EM 235 *3EA/1AA*
EB_ EB_ AB_ AB_	EB_	AB_	AEW_ AEW_ AEW_	AEW_ AEW_ AEW_ AAW_	EB_ AB_	AEW_ AEW_ AEW_ AAW_

EJERCICIO: ***Asigne a cada módulo de expansión las direcciones adecuadas.***

Balance de corriente

Introducción

Las unidades básicas S7-200 tienen una alimentación interna que suministra corriente para diversas funciones. Dicha alimentación abastece con corriente al aparato central, a los módulos de expansión, a una unidad de programación PG 702 de mano y a otros consumidores de corriente de 24VDC. En base a este capítulo podrá determinar cuánta energía (o corriente) puede suministrar el aparato central para su configuración.

Demanda de corriente

Cada aparato central S7-200 suministra corriente continua de 5V y de 24V. La fuente de alimentación de 5 VDC abastece con corriente a los módulos de extensión a través del bus de extensión, en tanto que la fuente de alimentación para sensores de corriente continua abastece los puntos de entradas de 24VDC y las bobinas de relé de los módulos de extensión. Los 24VDC pueden ser suministrados bien sea por el aparato central o por otra fuente adicional de 24VDC.

Ejemplo

El ejemplo de la siguiente página indica cómo calcular el balance de corriente de un sistema. Comparar dicho cálculo con el balance de corriente del aparato central para determinar si éste puede abastecer el sistema por sí solo.

El sistema se compone de:

- CPU 214 DC/DC/DC
- Tres EM221. 8 entradas digitales x DC 24V.
- Dos EM 222. 8 salidas digitales x RELE.

Tabla de cálculo: balance de corriente de la CPU 214 (ejemplo).

Modelo	***Nº de referencia***	***Consumo de corriente (mA)***		***Cant.***	***Balance de corriente (mA)***	
		5V	**24V**	**X**	**5V**	**24V**
CPU 214						
DC/DC/DC	6ES7 214-1AC00-0XB0	340	105[1]	1	**340**	**105**
AC/DC/Relé	6ES7 214-1BC00-0XB0	340	105[1]			
AC/AC/AC	6ES7 214-1CC00-0XB0	440				
EM 221 entradas digitales						
8 x 24 V DC	6ES7 221-1BF00-0XA0	60	60[1]	3	**180**	**180**
8 x 120 V DC	6ES7 221-1EF00-0XA0	70				
EM 222 salidas digitales						
8 x 24 V DC	6ES7 222-1BF00-0XA0	80				
8 x Relé	6ES7 222-1HF00-0XA0	80	85[2]	2	**160**	**170**
8 x 120/230 V AC	6ES7 222-1EF00-0XA0	120				
EM 223 Combinaciones de entradas/salidas digitales						
4 x 24V DC ent./sal.	6ES7 223-1BF00-0XA0	80	30[1]			
4 x 24V DC ent./ 4 x salidas relé.	6ES7 223-1HF00-0XA0	80	65[1,2]			
4 x 120V AC ent./ 4 x120-230 V sal.	6ES7 223-1EF00-0XA0	100	___			
EM Módulos analógicos						
AE 3 x 12 bits entradas analógicas	6ES7 231-0HC00-0XA0	15	60[1]			
AE 3 / AA 1x12 bits ent./sal. analógicas	6ES7 235-0KD00-0XA0	15	60[1]			
Suma = corriente necesaria[3]					***680***	***455***
Corriente máxima (suministrada por la CPU 214)					***1000***	***280***

[1] Corriente de entrada para entradas 24VDC.

[2] Alimentación de bobinas de relé.

[3] La corriente necesaria calculada en la columna de 5 V debe ser inferior a la corriente máxima. En caso contrario, será necesario desmontar un módulo de extensión. Si la corriente necesaria calculada en la columna de 24 V es superior a la corriente máxima, se le deberá agregar al sistema una fuente de alimentación externa de 24 VDC.

Ejercicio Realizar el balance de corriente para la configuración de el ejercicio de la página 7 (CPU 212).

Balance de corriente para la CPU 212

Modelo	***Nº de referencia***	***Consumo de corriente (mA)***		***Cant.***	***Balance de corriente (mA)***	
		5V	**24V**	**X**	**5V**	**24V**
CPU 212						
DC/DC/DC	6ES7 212-1AC00-0XB0	260	60[1]			
AC/DC/Relé	6ES7 212-1BC00-0XB0	260	60[1]			
AC/AC/AC	6ES7 212-1CC00-0XB0	320				
EM 221 entradas digitales						
8 x 24 V DC	6ES7 221-1BF00-0XA0	60	60[1]			
8 x 120 V DC	6ES7 221-1EF00-0XA0	70				
EM 222 salidas digitales						
8 x 24 V DC	6ES7 222-1BF00-0XA0	80				
8 x Relé	6ES7 222-1HF00-0XA0	80	85[2]			
8 x 120/230 V AC	6ES7 222-1EF00-0XA0	120				
EM 223 Combinaciones de entradas/salidas digitales						
4 x 24V DC ent./sal.	6ES7 223-1BF00-0XA0	80	30[1]			
4 x 24V DC ent./ 4 x salidas relé.	6ES7 223-1HF00-0XA0	80	65[1,2]			
4 x 120V AC ent./ 4 x120-230 V sal.	6ES7 223-1EF00-0XA0	100	___			
EM Módulos analógicos						
AE 3 x 12 bits entradas analógicas	6ES7 231-0HC00-0XA0	15	60[1]			
AE 3 / AA 1x12 bits ent./sal. analógicas	6ES7 235-0KD00-0XA0	15	60[1]			
Suma = corriente necesaria[3]						
Corriente máxima (suministrada por la CPU 212)					***600***	***180***

[1] Corriente de entrada para entradas 24VDC.
[2] Alimentación de bobinas de relé.
[3] La corriente necesaria calculada en la columna de 5 V debe ser inferior a la corriente máxima. En caso contrario, será necesario desmontar un módulo de extensión. Si la corriente necesaria calculada en la columna de 24 V es superior a la corriente máxima, se le deberá agregar al sistema una fuente de alimentación externa de 24 VDC.

Ejercicio Realizar el balance de corriente para la configuración de el ejercicio de la página 7 (CPU 214).

Balance de corriente para la CPU 214

Modelo	***Nº de referencia***	***Consumo de corriente (mA)***		***Cant.***	***Balance de corriente (mA)***	
		5V	**24V**	**X**	**5V**	**24V**
CPU 214						
DC/DC/DC	6ES7 214-1AC00-0XB0	340	105[1]			
AC/DC/Relé	6ES7 214-1BC00-0XB0	340	105[1]			
AC/AC/AC	6ES7 214-1CC00-0XB0	440				
EM 221 entradas digitales						
8 x 24 V DC	6ES7 221-1BF00-0XA0	60	60[1]			
8 x 120 V DC	6ES7 221-1EF00-0XA0	70				
EM 222 salidas digitales						
8 x 24 V DC	6ES7 222-1BF00-0XA0	80				
8 x Relé	6ES7 222-1HF00-0XA0	80	85[2]			
8 x 120/230 V AC	6ES7 222-1EF00-0XA0	120				
EM 223 Combinaciones de entradas/salidas digitales						
4 x 24V DC ent./sal.	6ES7 223-1BF00-0XA0	80	30[1]			
4 x 24V DC ent./ 4 x salidas relé.	6ES7 223-1HF00-0XA0	80	65[1,2]			
4 x 120V AC ent./ 4 x120-230 V sal.	6ES7 223-1EF00-0XA0	100	___			
EM Módulos analógicos						
AE 3 x 12 bits entradas analógicas	6ES7 231-0HC00-0XA0	15	60[1]			
AE 3 / AA 1x12 bits ent./sal. analógicas	6ES7 235-0KD00-0XA0	15	60[1]			
Suma = corriente necesaria[3]						
Corriente máxima (suministrada por la CPU 214)					***1000***	***280***

[1] Corriente de entrada para entradas 24VDC.

[2] Alimentación de bobinas de relé.

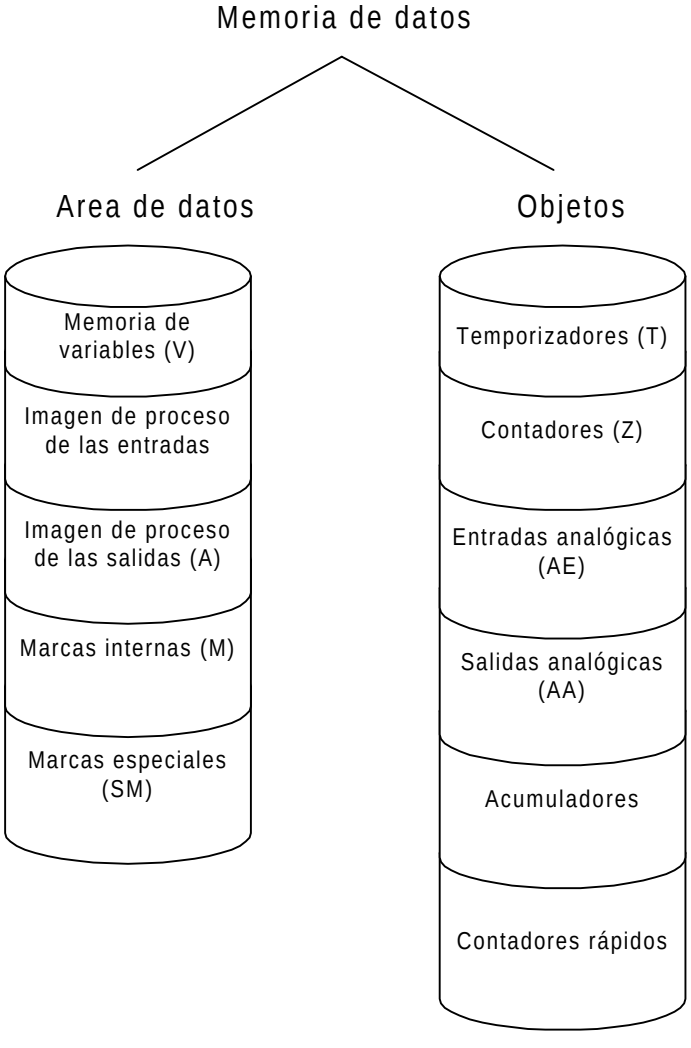
[3] La corriente necesaria calculada en la columna de 5 V debe ser inferior a la corriente máxima. En caso contrario, será necesario desmontar un módulo de extensión. Si la corriente necesaria calculada en la columna de 24 V es superior a la corriente máxima, se le deberá agregar al sistema una fuente de alimentación externa de 24 VDC.

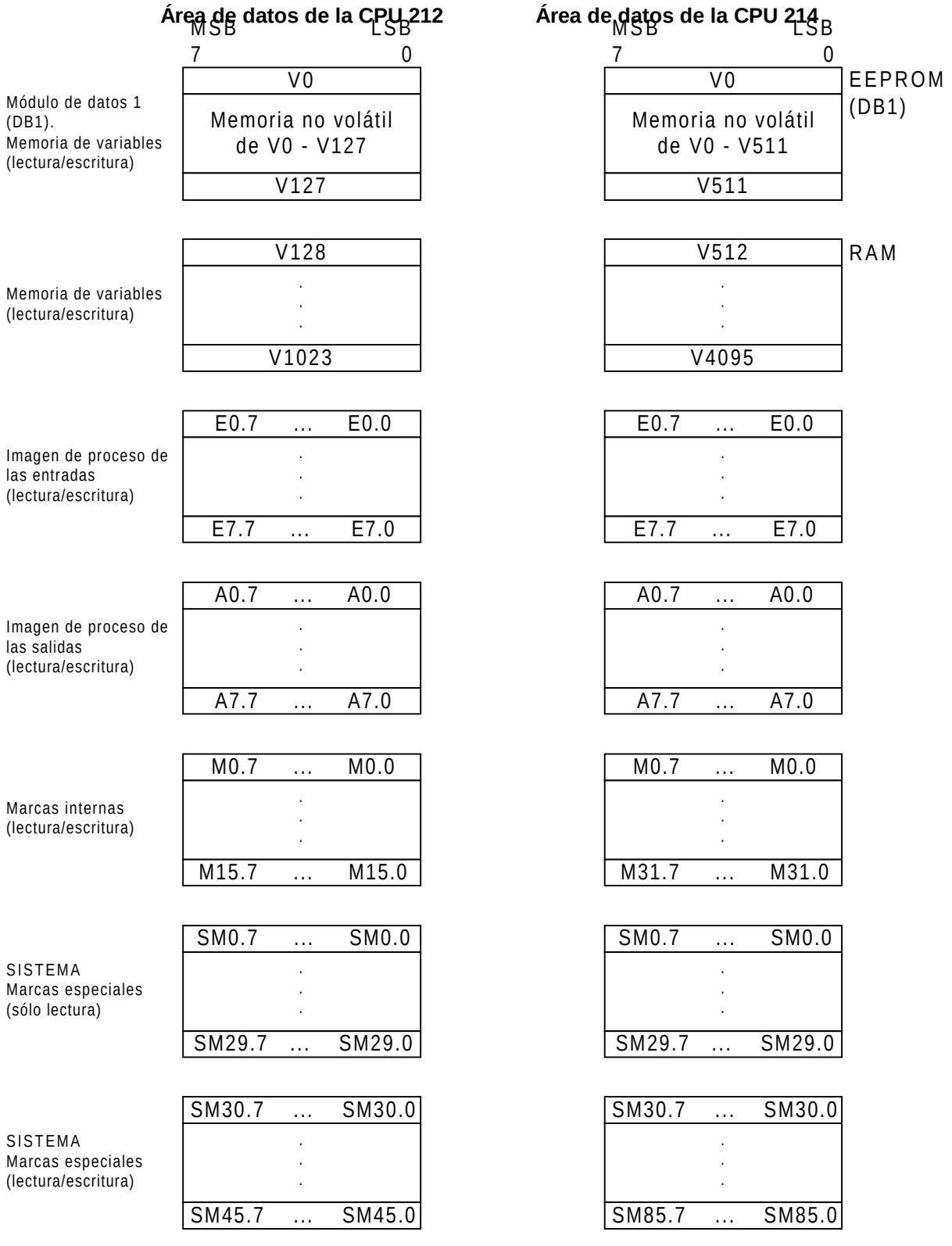
Memoria de datos

La memoria de datos del S7-200 se compone del área de datos y de objetos.

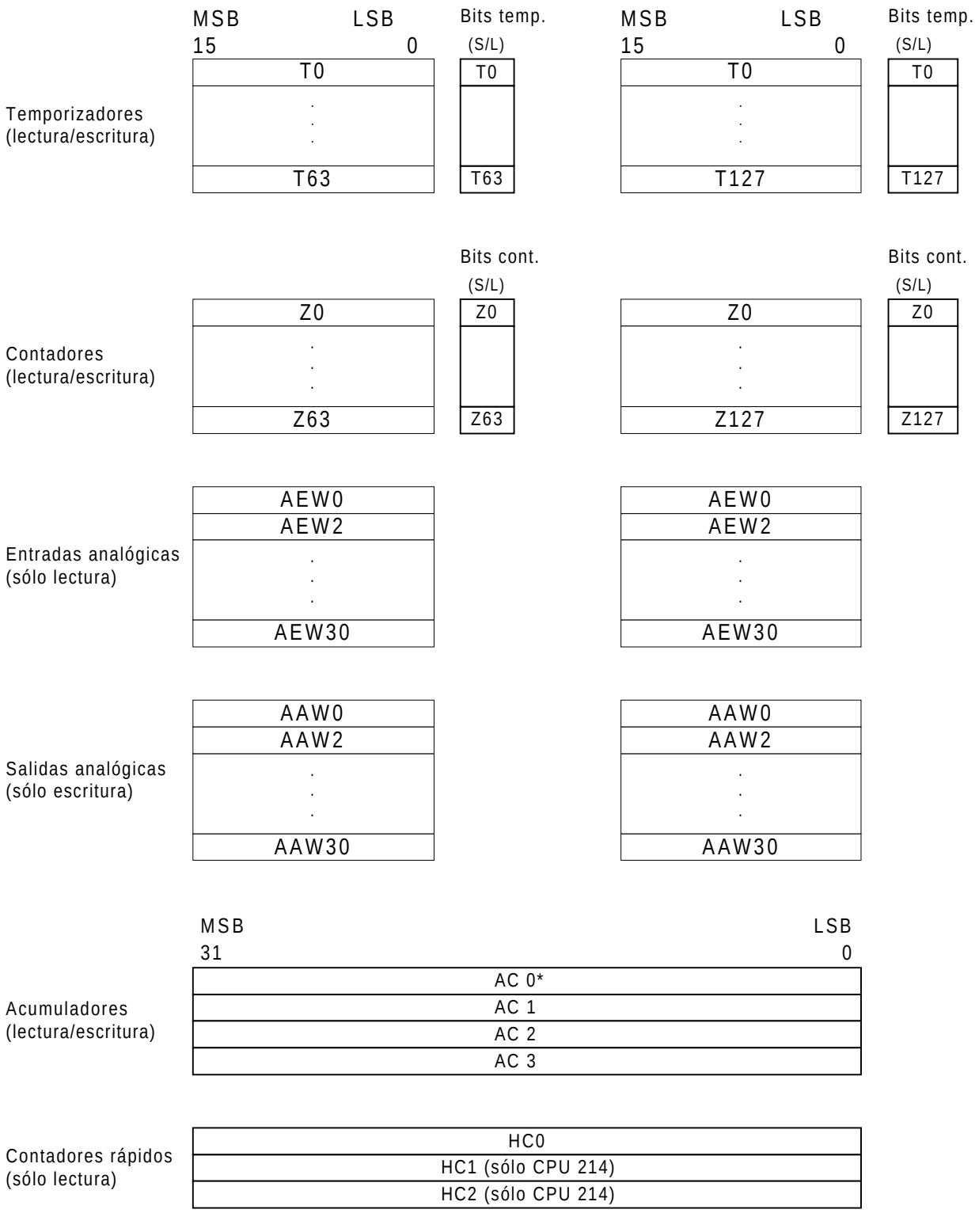
El área de datos se divide en una memoria de variables, una imagen de proceso de las entradas, una imagen de proceso de las salidas, marcas internas y marcas especiales. El área de datos es muy flexible, permitiendo accesos de lectura/escritura a todas las áreas de memoria, a excepción de algunas marcas especiales que sólo pueden leerse. El acceso a la memoria de datos completa se realiza en forma de bits, bytes, palabras o palabras dobles.

Los objetos son direcciones asignadas a elementos, como puede ser p.ej. el valor de un temporizador. Los objetos abarcan temporizadores, contadores, entradas y salidas analógicas, acumuladores y valores actuales de los contadores rápidos. El acceso a los objetos está más limitado, puesto que solamente se puede acceder a ellos en función del uso que se les haya previsto.





MSB = bit más significativo
LSB = bit menos significativo

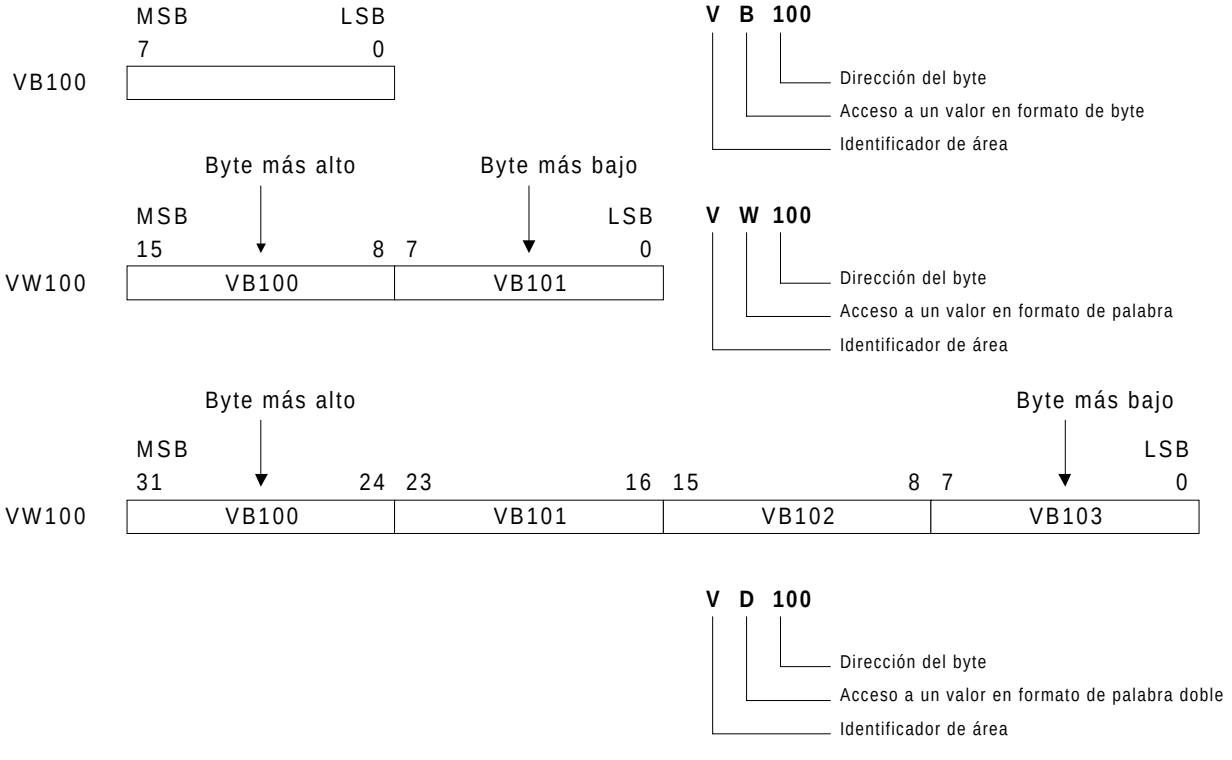


MSB = bit más significativo
LSB = bit menos significativo
*El AC 0 no puede utilizarse como puntero para el direccionamiento indirecto

El área de datos y los indicadores de tamaño van seguidos de la dirección de byte del byte, de la palabra o de la palabra doble. Independientemente del tamaño que tengan los datos puede accederse a ellos con direcciones pares o impares. La dirección se compone de un número decimal cuyo área depende de la versión de CPU utilizada (CPU 212 o CPU 214), así como del área de datos y del tamaño del valor al que se accede. La siguiente tabla indica las áreas de direcciones que corresponden a las diferentes áreas de datos.

Área de datos	CPU 212			CPU 214		
	Byte	Palabra	Palabra doble	Byte	Palabra	Palabra doble
Entradas	E0 a E7	E0 a E6	E0 a E4	E0 a E7	E0 a E6	E0 a E4
Salidas	A0 a A7	A0 a A6	A0 a A4	A0 a A7	A0 a A6	A0 a A4
Marcas internas	M0 a M15	M0 a M14	M0 a M12	M0 a M31	M0 a M30	M0 a M28
Marcas especiales	SM0 a SM45	SM0 a SM44	SM0 a SM42	SM0 a SM85	SM0 a SM84	SM0 a SM82
Memoria de variables	V0 a V1023	V0 a V1022	V0 a V1020	V0 a V4095	V0 a V4094	V0 a V4092

En la siguiente figura se puede comparar el acceso a una misma dirección utilizando los formatos byte, palabra y palabra doble



Respaldo de datos si falla la alimentación

El área de datos contiene una memoria de variables (V), la imagen de proceso de las entradas (E), la imagen de proceso de las salidas (A), marcas internas (M) y marcas especiales (SM). Los objetos pueden ser temporizadores (T), contadores (Z), entradas analógicas (AE), salidas analógicas (AA), acumuladores (AC) y los valores actuales de los contadores rápidos (HC).

Las entradas y salidas analógicas así como los valores de los contadores rápidos (HC) se almacenan por lo general en elementos (módulos analógicos o contador rápido) más que en la memoria RAM. La memoria RAM provee espacio para las demás áreas de datos y objetos.

Un condensador de alta potencia que alimenta la memoria RAM se encarga de respaldar los datos por un tiempo determinado después de desconectar el autómata programable y sin necesidad de ningún tipo de mantenimiento adicional. En el caso de la CPU 212, después de poner el autómata en marcha, el condensador respalda la memoria unas 50 horas, mientras que en la CPU 214 se respalda unas 190 horas.

El usuario puede definir hasta seis áreas remanentes para elegir las áreas de memoria que deberán ser respaldadas cuando se interrumpa la alimentación. No todas las áreas de datos almacenadas en la memoria RAM pueden ser remanentes: las áreas de datos que pueden ser remanentes son V, M, T(T0 a T31 y T64 a T95) y C.

Para definir un área remanente hay que indicar en la memoria un área de datos “de ... a ...”. Este área no se borrará al poner el S7-200 en marcha, a condición de que el condensador de alta potencia haya podido respaldar el contenido de la memoria RAM. En otro caso, se activa la marca Datos remanentes perdidos (SM0.2), borrándose las restantes áreas remanentes junto con los datos de usuario no remanentes.

En la siguiente tabla se muestra el ajuste por defecto de las áreas remanentes.

Área remanente	CPU 212	CPU 214
Área remanente 0	V0 - V1023	V0 - V4095
Área remanente 1	no utilizada	no utilizada
Área remanente 2	T0 - T31	T0 - T31
Área remanente 3	no utilizada	T64 - T95
Área remanente 4	Z0 - Z63	Z0 -Z127
Área remanente 5	M0 - M15	M0 - M31

Marcas especiales

Las marcas especiales ponen a disposición una serie de funciones de estado y control y también sirven para intercambiar informaciones entre el autómata y el programa.

Las marcas especiales disponen de áreas de sólo lectura y de lectura/escritura. El área de sólo lectura comienza a partir de SM0 y termina en SM29. El autómata actualiza solamente las direcciones de sólo lectura que proporcionan diversas informaciones de estado. Con las marcas SM30 a SM45 en el caso de la CPU 212, y SM30 a SM85 en el caso de la CPU 214 se pueden seleccionar y controlar funciones especiales (contadores rápidos, modo freeport (comunicación Freeport) así como salidas de impulsos) y acceder a los valores de los dos potenciómetros yntegrados en la CPU 214.

Ver capítulo Nº 9, Curso SIMATIC S7-200 , Nivel I
Ver capítulo Nº 4, Curso SIMATIC S7-200, Nivel II

Temporizadores

Los temporizadores (TON ó TONR) son elementos que cuentan intervalos de tiempo. Los temporizadores del S7-200 tienen resoluciones (intervalos) de 1, 10 y 100 milisegundos. La CPU 212 dispone de 64 temporizadores. La CPU 214 dispone de 128 temporizadores.

Ver capítulo Nº 11, Curso SIMATIC S7-200 , Nivel I

Contadores

Los contadores (ZV ó ZVR) son elementos que cuentan los cambios de negativo a positivo en las entradas de contaje. La CPU 212 dispone de 64 contadores. La CPU 214 dispone de 128 contadores.

Ver capítulo Nº 12, Curso SIMATIC S7-200 , Nivel I

Entradas y salidas analógicas

Los módulos analógicos convierten valores reales (tensión, temperatura, etc.) en valores digitales en formato de palabra y viceversa. Los módulos analógicos pueden ser módulos de entradas, módulos de salidas, o bien módulos de entradas y salidas.

Ver capítulo Nº 11, Curso SIMATIC S7-200, Nivel II

Acumuladores

Los acumuladores son elementos de lectura/escritura que se utilizan igual que una memoria. Los acumuladores se pueden utilizar por ejemplo para transferir parámetros no sólo a subrutinas sino también a cualquier operación o cuadro (box) parametrizable. Cuando un evento de interrupción provoca un salto a una rutina de interrupción, el autómata programable almacena los valores que se encuentran en el acumulador justo antes de la rutina de interrupción. Los valores se restablecen al finalizar la ejecución de la rutina de interrupción. Los acumuladores se pueden utilizar mientras se ejecuta dicha rutina sin el riesgo de que se modifiquen datos del programa principal. Sin embargo, los acumuladores no permiten transferir parámetros entre el programa principal y una rutina de interrupción.

Ver capítulo Nº 3, Curso SIMATIC S7-200, Nivel II

Contadores rápidos

Los contadores rápidos (HSC) cuentan eventos más deprisa de lo que puede explorarlos el autómata. Los contadores rápidos disponen de un valor de contaje entero de 32 bits con signo (también denominado valor actual). En caso de acceder directamente al valor actual de un contador rápido, dicho valor permite una acceso de sólo lectura.

Para poder escribir en los valores actuales de los contadores rápidos existen funciones especiales.

Ver capítulo Nº 9, Curso SIMATIC S7-200, Nivel II

Resumen de las áreas de datos y métodos de direccionamiento

La memoria de datos del S7-200 se compone de un área de datos y de objetos. La siguiente tabla describe las áreas de memoria y limitaciones de acceso. También indica las áreas de datos que pueden definirse como áreas remanentes así como las áreas que pueden forzarse (su valor no puede ser modificado ni por el programa ni por el operador).

Área	Descripción	Bit	Byte	Palabra	Palabra doble	Puede ser remanente	Puede forzarse
E	Entradas digitales e imagen de proceso	lectura/ escritura	lectura/ escritura	lectura/ escritura	lectura/ escritura	no	sí
A	Salidas digitales e imagen de proceso	lectura/ escritura	lectura/ escritura	lectura/ escritura	lectura/ escritura	no	sí
M	Marcas internas	lectura/ escritura	lectura/ escritura	lectura/ escritura	lectura/ escritura	sí	sí
SM	Marcas especiales SM0 a 29, sólo lectura	lectura/ escritura	lectura/ escritura	lectura/ escritura	lectura/ escritura	no	no
V	Memoria de variables	lectura/ escritura	lectura/ escritura	lectura/ escritura	lectura/ escritura	sí	sí
T	Valores actuales del temporizador y bits de temporizador	Bit T lectura/ escritura	no	Valor T actual lecutra/ escritura	no	T0 -31, T64-95	no
Z	Valores actuales del temporizador y bits de contador	Bit Z lectura/ escritura	no	Valor Z acutal lectura/ escritura	no	sí	no
HC	Valores actuales del contador rápido	no	no	no	sólo lectura	no	no
AE	Entradas analógicas	no	no	sólo lectura	no	no	sí
AA	Entradas analógicas	no	no	sólo lectura	no	no	sí
AC	Acumuladores	no	lectura/ escritura	lectura/ escritura	lectura/ escritura	no	no

Operaciones lógicas con contactos

Objetivo

Las operaciones lógicas con contactos sirven para crear y conectar circuitos lógicos.

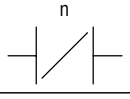
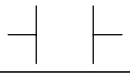
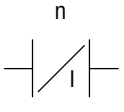
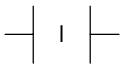
Descripción

En el lenguaje de programación KOP, los contactos pueden ser contactos normalmente abiertos y contactos normalmente cerrados.

En KOP se usa una I (immediata) para indicar que la operación se va ejecutar directamente. Esta operación directa, o contacto, lee el valor direccionado en la entrada física al ejecutarse la operación o contacto. Sin embargo, la imagen de proceso no se actualiza. La operación no directa, o contacto, lee el valor direccionado en la imagen de proceso, que es actualizada por el autómata antes de cada ciclo.

Representación

A continuación se representan las operaciones lógicas con contactos en el lenguaje KOP

Elemento KOP	Símbolo KOP	Descripción	Operandos
Consulta: si no hay tensión.		Consulta si en el elemento n no hay tensión (n =0)	n: E, A, M, SM, T, (Bit) Z, V
Consulta: si hay tensión.		Consulta si en el elemento n hay tensión (n=1)	n: E, A, M, SM, T, (Bit) Z, V
Consulta directa (si no hay tensión)		Consulta de forma directa si en el elemento n no hay tensión (n=0)	n: E (Bit)
Consulta directa (si hay tensión)		Consulta de forma directa si en el elemento n hay tensión (n=1)	n: E (Bit)

Operaciones lógicas con salidas

Objetivo

Las operaciones lógicas con salidas activan o desactivan salidas digitales

Descripción

En el esquema de contactos (KOP), la lógica que controla el flujo de corriente activa o desactiva la bobina de salida. Cuando la corriente excita la bobina se activa la salida (o salidas) de la operación Poner a 1 (activar) y se desactiva la salida (o salidas) de la operación Poner a 0 (desactivar).

Se utiliza una I para indicar que la operación se va a ejecutar directamente. El valor de la salida (o salidas) direccionada se escribe en la imagen de proceso y en la salida física mientras se ejecuta el programa. A diferencia de éstas, las operaciones no directas escriben el valor de la salida (o salidas) direccionada solamente en la imagen de proceso.

Representación

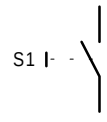
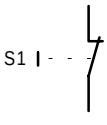
A continuación se explican las operaciones lógicas con salidas en representación KOP.

Elemento KOP	Símbolo KOP	Descripción	Operandos
Bobina	n —( )—	Asignar	n: (Bit) E, A, M, SM, T, Z, V
Bobina Poner a 1 directamente	S_Bit N —(S)—	La bobina poner a 1 (S) activa el área de de salidas a partir de S_BIT tantas salidas como indique N	S_BIT: (Bit) E, A, M, SM,T,Z,V N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Poner a 0 directamente	S_Bit N —(R)—	La bobina poner a 0 (R) activa el área de de salidas a partir de S_BIT tantas salidas como indique N	S_BIT: (Bit) E, A, M, SM,T,Z,V N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Poner a 0 directamente	S_Bit N —(R_I)—	Poner a 0 directamente	S_BIT: (Bit) A N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Poner a 1 directamente	S_Bit N —(S_I)—	Poner a 1 directamente	S_BIT: (Bit) A N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Asignar directamente	n —(I)—	Asignar directamente	n: (Bit) A

Consulta del estado de señal, contactos NA y NC

Contactos NA y NC

Las entradas del AG se consultan cíclicamente para apreciar su estado de señal; para el AG es indiferente si el emisor de señal es un contacto normalmente abierto o normalmente cerrado.

El emisor de señal es un	El emisor está	Tensión en la entrada	Estado de la señal en la entrada
Contacto NA 	accionado	existente	**1**
	no accionado	no existente	**0**
Contacto NC 	accionado	no existente	**0**
	no accionado	existente	**1**

Contacto NA

Si conecta a una entrada un contacto NA, dicha entrada tiene estado de señal “1” cuando se acciona el contacto.

Contacto NC

En cambio, si a una entrada se conecta un contacto NC, ésta tiene estado de señal “0” cuando se acciona el contacto.

Así pues, para que el AG pueda distingue entre contactos NA y NC, el programa deberá incluir instrucciones de consulta del estado de señal en las entradas.

Consulta sobre estado de señal “1”

Una entrada tiene estado de señal “1” cuando:

- está cerrado el **contacto NA**, es decir, **accionado**, o
- está cerrado el **contacto NC**, es decir, **no accionado**.

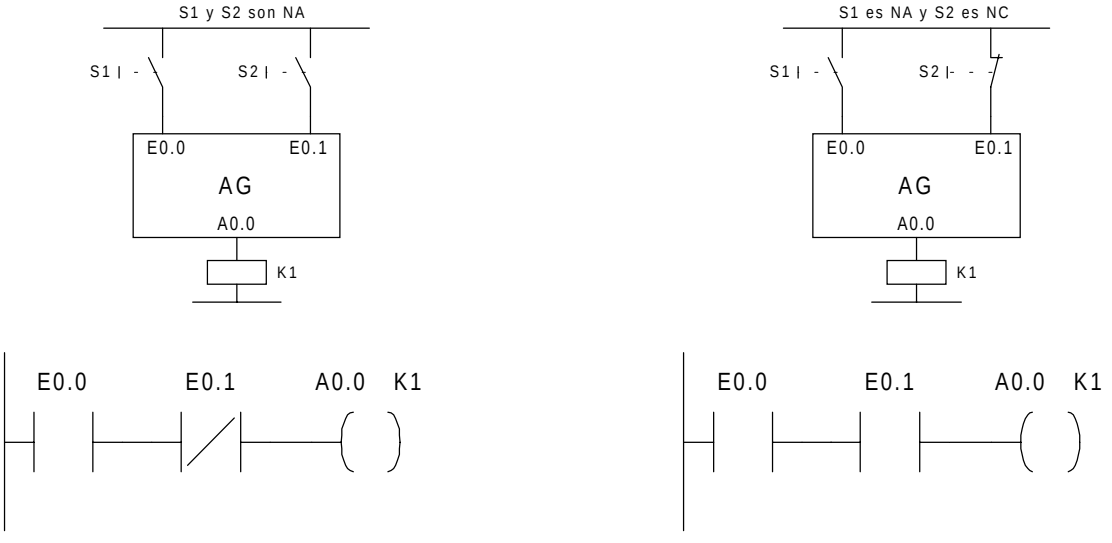
Consulta sobre estado de señal “0”

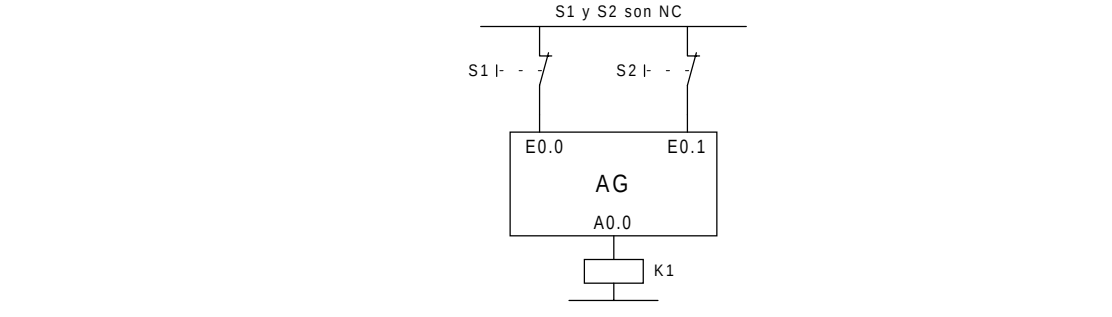
Una entrada tiene estado de señal “1” cuando:

- está abierto el **contacto NA**, es decir, **no accionado**, o
- está abierto el **contacto NC**, es decir, **accionado**.

Veamos un ejercicio sobre este tema.

El contacto K1 deberá excitarse cuando se accione el emisor S1 y no esté accionado el emisor S2.

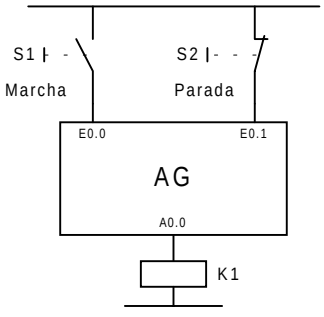




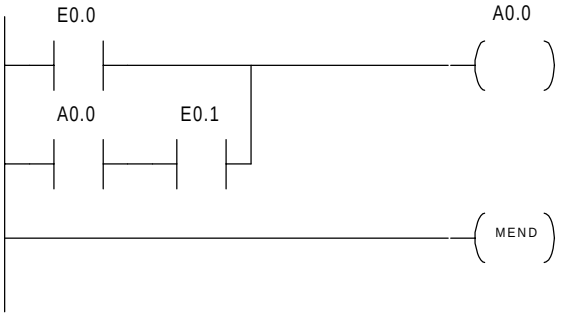
Ejercicio

Se trata de realizar, mediante contactos una función de memoria o autoretención.

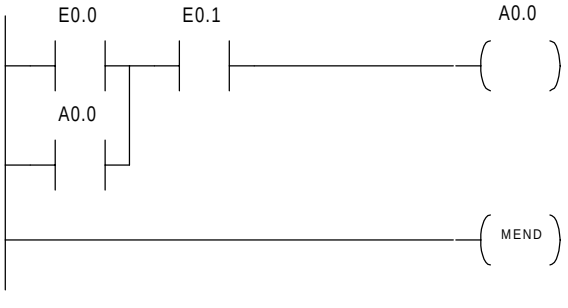
Se dispone de un contacto NA (E0.0) para la marcha y un contacto NC (E0.1) para la parada.



Función de memoria con preferencia a la activación.



Función de memoria con preferencia a la puesta en 0.



Operaciones de transferencia

Objetivo

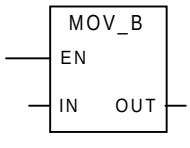
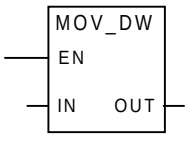
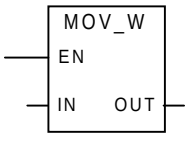
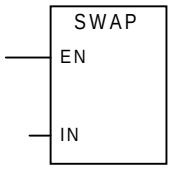
Las operaciones de transferencia se utilizan para transferir datos de una dirección a otra.

Descripción

Estas operaciones transfieren un valor de byte, palabra o palabra doble de una dirección a otra. La operación Intercambiar bytes de uan palabra (SWAP) intercambia el byte más significativo y el byte menos significativo de una palabra.

Representación

A continuación se explican las operaciones de transferencia en representación KOP.

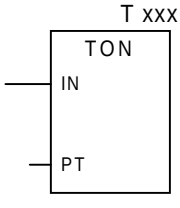
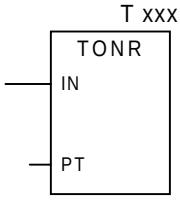
Elemento KOP	Símbolo KOP	Descripción	Operandos
MOV_B		Transferir byte	IN: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (Byte) VB, EB, AB, MB, SMB, AC, *VD, *AC
MOV_DW		Transferir doble palabra	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC, &VB, &EB, &AB, &MB, &T, &Z OUT: (Dpal:) VD, ED, AD, MD, SMD, AC, *AC
MOV_W		Tranferir palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal:) VW, T, Z, EW, AW, MW, SMW, AC, AAW, *VD, *AC
SWAP		Intercambiar bytes en palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC

En consecuencia, las operaciones con temporizadores de 1ms, 10 ms, y 100 ms presentan algunas diferencias. Así, el valor actual y el bit T de un temporizador de 1 ms pueden actualizarse varias veces durante un solo ciclo. Si se utiliza un temporizador de 100 ms en una subrutina o en una subrutina o en una rutina de interrupción que no se ejecuta en todos los ciclos no se actualizarán correctamente el valor actual ni el bit T. Al valor actual se suman solamente los valores de contaje de la base de tiempo que pertenecen al ciclo actual. De forma similar, cuando un temporizador de 100 ms habilitado aparece varias veces en un mismo ciclo hace que el valor de contaje de dicho ciclo se sume repetidas veces. Por tanto, los temporizadores de 100 ms deberán utilizarse solamente cuando se vayan a ejecutar exactamente una vez por ciclo.

La base de tiempo del sistema y el ciclo trabajan de forma asíncrona, produciendo una imprecisión de cuantificación. Esta imprecisión suele ser insignificante, pero podría tener importancia en caso de utilizar valores de preselección muy bajos. En tal caso habrá que elegir si es posible una resolución más baja, o bien utilizar alternativamente una interrupción temporizada.

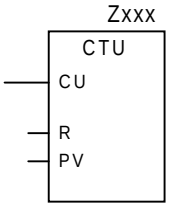
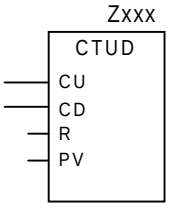
Representación

A continuación se explican las operaciones de temporización en representación KOP.

Elemento KOP	Símbolo KOP	Descripción	Operandos
TON		Temporizador de retardo a la conexión	T xxx: (pal.) CPU 212: 32-63 CPU 214: 32-63 96-127 PT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
TONR		Temporizador de retardo a la conexión con memoria	T xxx: (pal.) CPU 212: 0-31 CPU 214: 0-31, 64-95 PT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC

Representación

A continuación se explican las operaciones de contaje en representación KOP.

Elemento KOP	Símbolo KOP	Descripción	Operandos
CTU		Contar adelante.	Zxxx: (pal.) CPU 212: 0-47; CPU 214: 0-47, 80-127 PV: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
CTUD		Contar adelante/atrás	Zxxx: (pal.) CPU 212: 48-63 CPU 214: 48-79 PV: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC

- El cuadro **Contar adelante** (**CTU**) empieza a contar hasta el valor máximo cuando se produce un flanco creciente en la entrada de contaje adelante. Si el valor actual del contador es mayor o igual al valor de preselección (PV) se activa el bit de contaje. Este bit se pone a 0 cuando se activa la entrada de desactivación (R) y para de contar al alcanzar el valor máximo (32 767).

- El cuadro **Contar adelante/atrás** (**CTUD**) empieza a contar adelante hasta el valor máximo cuando se produce un flanco creciente en la entrada de contaje adelante (CU). Por el contrario, empieza a contar atrás cuando se produce un flanco creciente en la entrada de contaje atrás. Si el valor actual del contador es mayor o igual al valor de preselección (PV) se activa el bit de contaje. Este bit para de contar adelante en cuanto se alcanza el valor máximo (32 767), y para de contar atrás en cuanto se alcanza el valor mínimo (-32 768). El bit de contaje se borra en cuanto se activa la entrada de desactivación (R).

Operaciones de comparación

Objetivo

El resultado de las operaciones de comparación permite tomar decisiones de control.

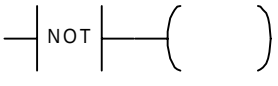
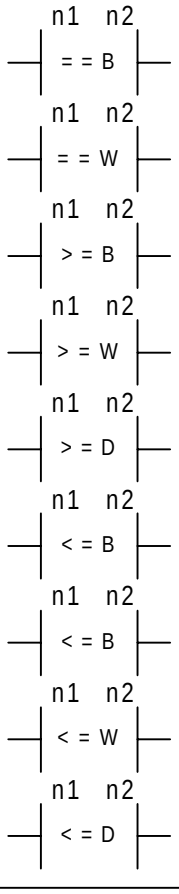
Descripción

El esquema de contactos (KOP) utiliza contactos de comparación para comparar valores de bytes, palabras y palabras dobles. Las comparaciones posibles son las siguientes: menor o igual (<=), igual (=), mayor o igual (>=).

En KOP, las operaciones de comparación comparan bytes, palabras y palabras dobles. El resultado binario se carga y se combina mediante una operación Y u O dependiendo del tipo de operación (<=, =, >=). Para realizar las comparaciones <>, >, y < hay que utilizar la operación NOT juntos con las operaciones <=, =, o>=.

Las comparaciones de bytes no llevan signo. Las comparaciones de palabras y palabras dobles llevan signo (7FFF<8000, y 7FFFFFFF>80000000).

A continuación se explican las operaciones de comparación en KOP

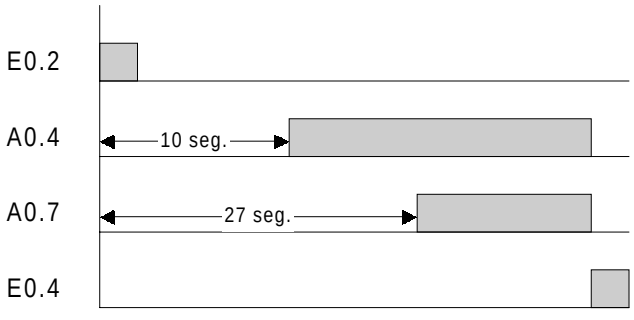
Elemento KOP	Símbolo KOP	Descripción	Operandos
Contacto (NOT)		Contacto negado	ninguno
Contacto (comparación)		Comparar contactos B = byte W = entero (16 bits) D = entero doble (32 bits)	n1, n2: (Byte) VB, IB, AB, MB, SMB, AC, constante, *VD, *AC n1, n2: (pal.) VW, T, Z, EW, AW, MW, SWMW, AC, AEW, constante, *VD, *AC n1, n2: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC

Ejemplo:

Mediante el uso de un sólo temporizador y con la ayuda de las funciones de comparación activar la salidas A0.4 y A0.7 una vez hayan transcurridos 10 y 27 segundos respectivamente de la activación de E0.2.

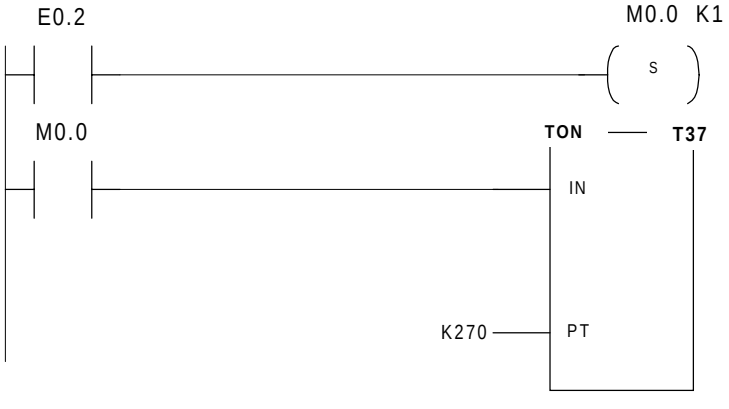
Para la puesta a 0 de ambas salidas se utilizará la entrada E0.4

Diagrama de impulsos



Edición en KOP

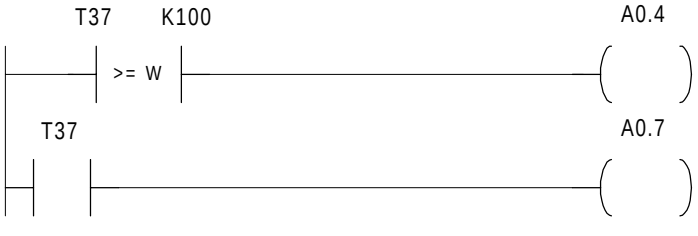
ARRANQUE DE LA TEMPORIZACIÓN



Con la entrada E0.2 seteamos una marca interna M0.0 que servirá para lanzar el temporizador T37.

El tiempo de carga del temporizador T0N es 27 segundos

ASIGNACIÓN DE SALIDAS



Cuando el temporizador T37 llegue a contar 10 segundos

= salida A0.4

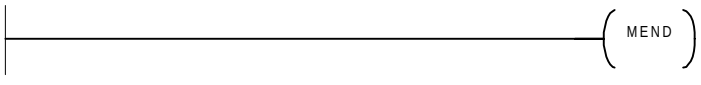
Cuando el temporizador finalice el tiempo de contaje (27 seg.)

= salida A0.7

PUESTA A 0

Con la acción de E0.3 volvemos a las condiciones iniciales, es decir, reseteo de la cuenta del temporizador y reseteo de la marca interna M0.0 que lanza a T37

FIN DE PROGRAMA



Ejemplo: Señalización de un garaje

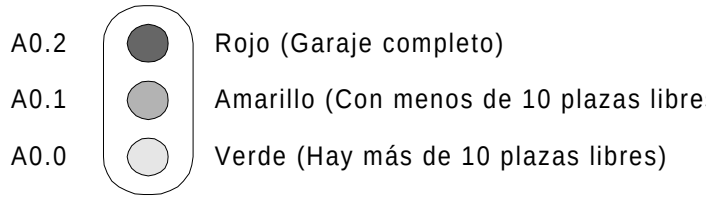
Objetivo

Un garaje dispone de 250 plazas libres. Se desea que:

- Mientras el número de plazas libres sea mayor o igual a 10 deberá lucir en la entrada del garaje una luz verde (A0.0).
- Cuando el número de plazas libres sea inferior a 10 se deberá activar una señal de aviso (A0.1).
- Por último, una vez que el garaje este totalmente ocupado se deberá iluminar una luz roja y apagar la señal de aviso.

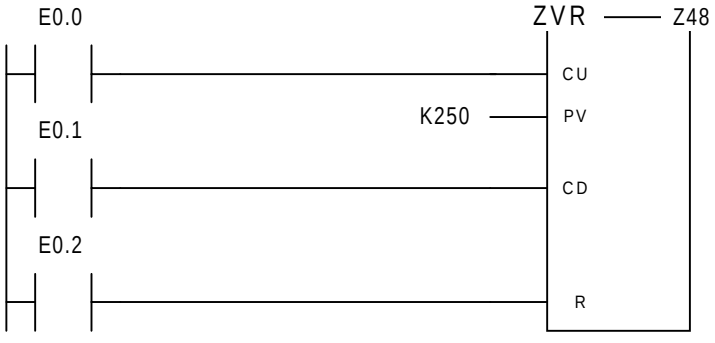
Entradas/Salidas

Símbolo	Descripción
E0.0	Sensor en la entrada del garaje.
E0.1	Sensor en la salida del garaje.
E0.2	Pulsador para iniciar el estado del contaje
A0.0	Luz verde
A0.1	Señal de aviso
A0.2	Luz roja
ZVR48	Contador ascendente/descendente



Edición en
KOP

CUENTA / DESCUENTA

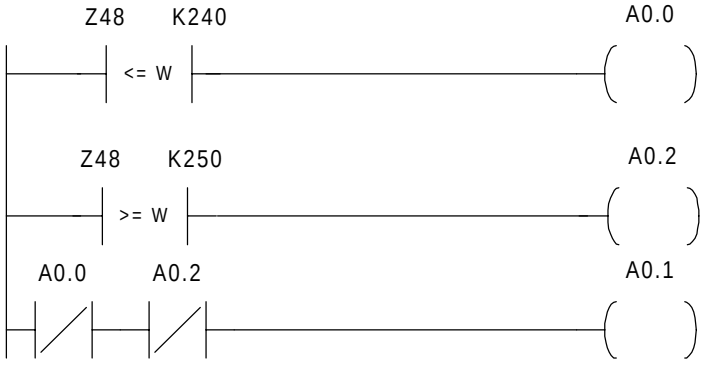


Si el sensor de la entrada esta activo, el contador cuenta hacia adelante

Si el sensor de la salida esta activo, el contador cuenta hacia detrás.

Cuando la cuenta llegue al valor preseleccionado (250), Z48 activa el bit del contador.

ASIGNACIÓN DE SALIDAS

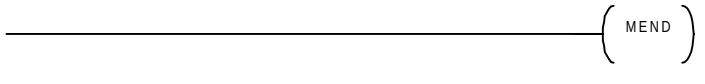


Mientras que el contador registre un número inferior a 240 (10 plazas libres), se ilumina la lámpara verde.

Cuando el garaje esta completo se indica por medio de la salida A0.2

Si ninguna de las dos salidas anteriores estan activas
= A0.1 (señal de aviso)

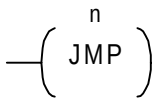
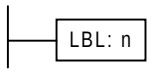
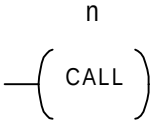
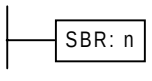
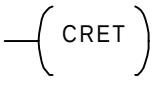
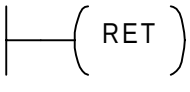
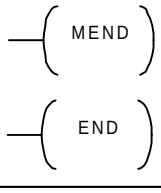
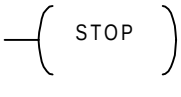
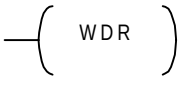
MEND



Instrucción fin de programa principal.

Operaciones de salto y operaciones de subrutinas (continuación)

En la siguiente tabla se recogen las distintas operaciones de salto y subrutinas en KOP

Elemento KOP	Símbolo KOP	Descripción	Operandos
Bobina Saltar a meta		Saltar a meta (JMP)	n: CPU 212: M0-M63 CPU 214: M0-M255
LBL		Definir meta	n: CPU 212: M0-M63 CPU 214:
Bobina Llamar subrutina		Llamar subrutina	n: CPU 212: 0-15 CPU 214: 0-63
SBR		Comenzar subrutina	n: (pal.) CPU 212: 0-15 CPU 214: 0-63
Bobina Retorno condicional desde subrutina		Retorno condicional desde subrutina	ninguno
Bobina Retorno absoluto desde subrutina		Retorno absoluto desde subrutina	ninguno
Bobina Fin ejecución		Fin absoluto Fin condicional	ninguno
Bobina STOP		Pasar a modo STOP	ninguno
Bobina Poner a 0 temporizador de vigilancia		Poner a 0 temporizador de vigilancia	ninguno

Ejercicio: Señal de periodo preseleccionable

Objetivo:

Mediante la preselección de tres interruptores se desea conseguir una señal de periodo variable (A0.0).

Los tiempos deseados son los siguientes:

1. Si está activo el interruptor 0 (E0.0=1) ==> T = 0,6 sg.
2. Si está activo el interruptor 1 (E0.1=1) ==> T = 1 sg.
3. Si está activo el interruptor 2 (E0.2=1) ==> T = 2 sg.

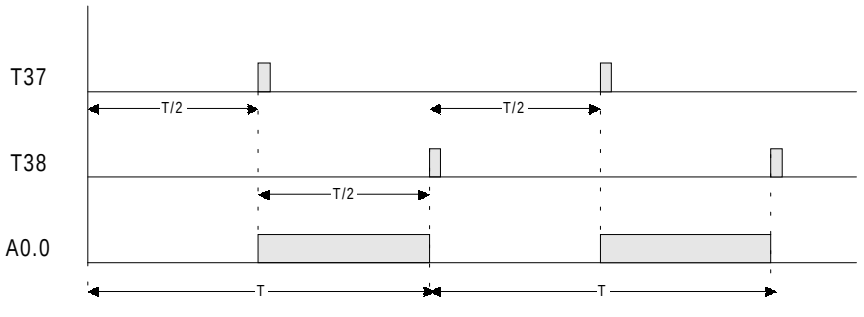
En el caso de que ninguno de los interruptores este activo la salida debe de anularse.

Descripción:

Para la solución de este ejercicio se han utilizado 3 subrutinas y un salto a meta. Cuando ninguno de los interruptores este activo se produce el salto a meta con el fin de resetear la salida A0.0.

Una vez sean activados cualquiera de los interruptores E0.0, E0.1 ó E0.3 se origina un salto a las subrutinas 0, 1 y 2 respectivamente. En estas subrutinas se carga el valor de tiempo correspondiente en una marca de variable (VW0). Al finalizar dichas subrutinas se inicia la temporización.

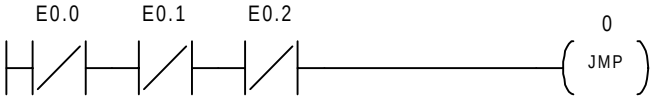
Para conseguir una señal periódica se utilizan dos temporizadores TON, T37 y T38. El primero de ellos comienza la cuenta cuando la salida A0.0 está desactivada. Al alcanzar el valor de preselección el temporizador T37 pone a 1 su bit T. Este bit lanza un segundo temporizador T38 y activa la salida A0.0 que queda autoretenida hasta que T38 finaliza la el tiempo de contaje. Una vez que A0.0 vuelve a caer se inicia de nuevo el ciclo anterior.



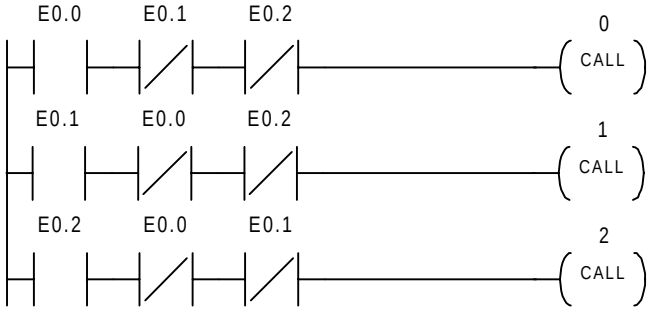
Entradas/Salidas en el programa

Símbolo	Descripción
E0.0	Interruptor 0. Selecciona un tiempo T/2 = 0,3 seg.
E0.1	Interruptor 1. Selecciona un tiempo T/2 = 0,5 seg.
E0.2	Interruptor 2. Selecciona un tiempo T/2 = 1 seg.
A0.0	Señal periódica
T37	Temporizador TON
T38	Temporizador TON

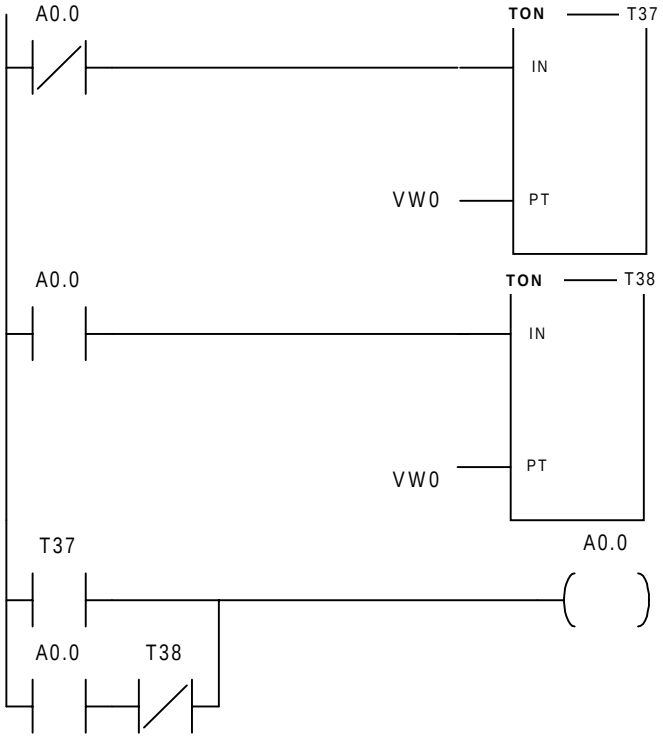
Edición en KOP



Mientras no se accione ningún interruptor, salto a la meta 0.

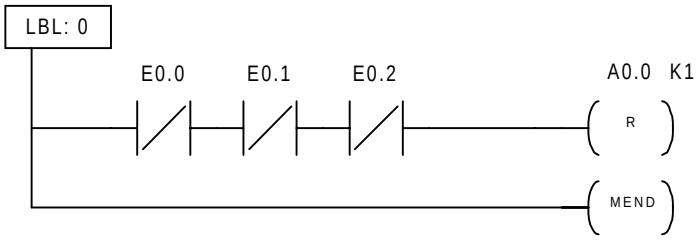


Cuando E0.0 = 1 llamo a la subrutina 0.
Cuando E0.1 = 1, llamo a la subrutina 1.
Cuando E0.2 =1, llamo a la subrutina 2.



Para conseguir una señal periódica se utilizan dos temporizadores TON, T37 y T38. El primero de ellos comienza la cuenta cuando la salida A0.0 está desactivada. Al alcanzar el valor de preselección el temporizador T37 pone a 1 su bit T. Este bit lanza un segundo temporizador T38 y activa la salida A0.0 que queda autoretenida hasta que T38 finaliza la el tiempo de contaje. Una vez que A0.0 vuelve a caer se inicia de nuevo el ciclo anterior.

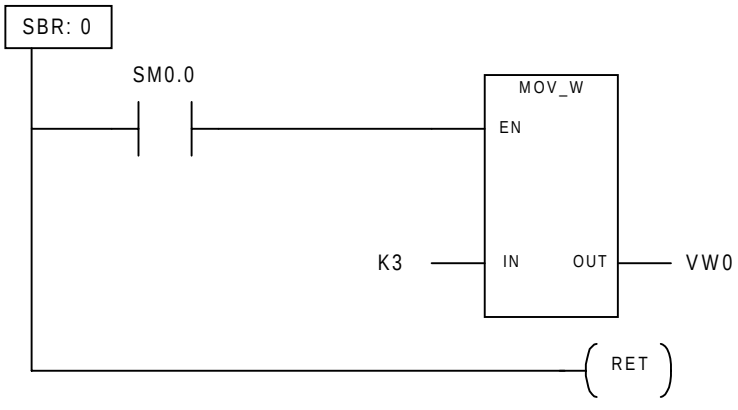
Ver diagrama de impulsos de la página 6.



Meta 0:

Si no se ha accionado ningún interruptor se resetea la salida

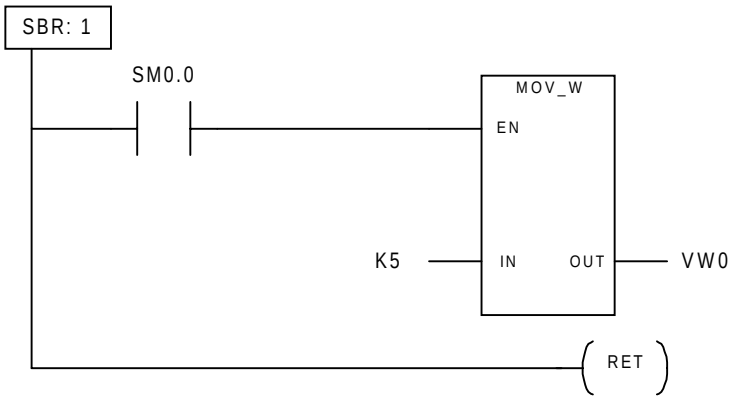
Con la instrucción MEND se consigue volver a la primera línea del programa



Subrutina 0:

Carga en la marca de variable VW0 los 0,3 segundos (T/2) para obtener una señal de periodo 0,6 segundos.

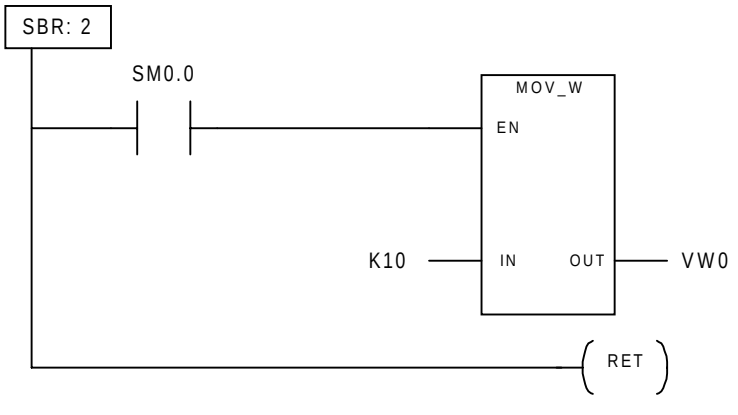
Con la instrucción RET (retorno incondicional) se vuelve a la línea inmediatamente posterior a la llamada de la subrutina 0.



Subrutina 1:

Carga en la marca de variable VW0 los 0,5 segundos (T/2) para obtener una señal de periodo 1 segundo.

Con la instrucción RET (retorno incondicional) se vuelve a la línea inmediatamente posterior a la llamada de la subrutina 1.



Subrutina 2:

Carga en la marca de variable VW0 l segundo (T/2) para obtener una señal de periodo 2 segundos.

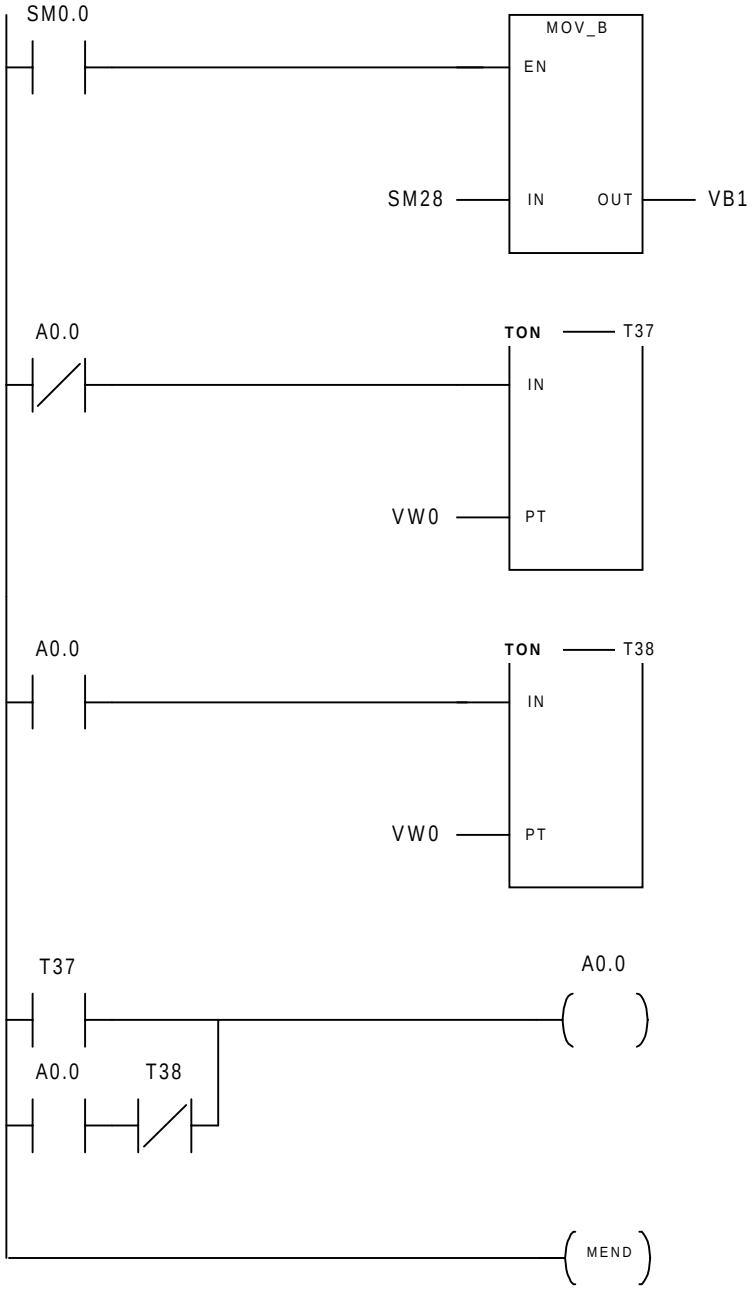
Con la instrucción RET (retorno incondicional) se vuelve a la línea inmediatamente posterior a la llamada de la subrutina 2.

Ejemplo: Generación de pulsos de frecuencia variable

Se desea diseñar un programa en el que el tiempo de activación/ desactivación de una señal sea posible regularlo mediante el potenciómetro integrado en el S7-200.

Se tomará a A0.0 como la señal de intermitencia variable

Edición en KOP



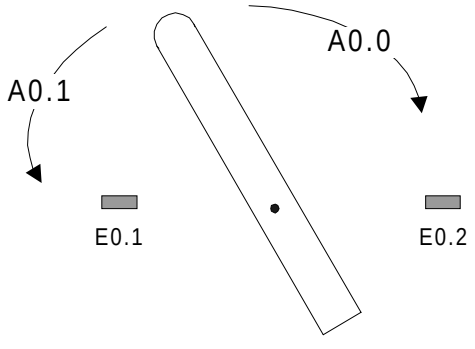
Carga del valor actual del potenciómetro en el byte de la marca de variable VB1 (byte mas bajo de la palabra VW0)

Cargo el tiempo seleccionado por el potenciómetro en dos temporizadores T0N que se activan con los distintos estados de A0.0

Ejercicio 1

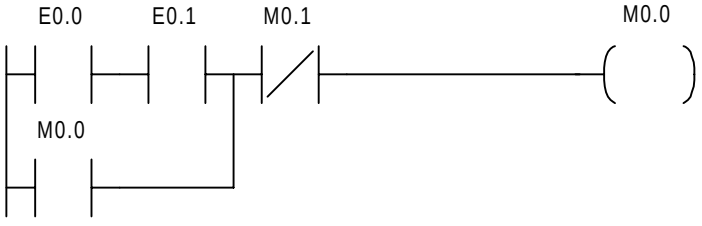
Control de la acción de un brazo motorizado

La figura siguiente representa una máquina que tiene un brazo motorizado. Cuando se pulsa el botón de arranque con el brazo en la posición 1, el brazo gira en sentido horario y detiene su rotación una vez que ha llegado a la posición 2. Transcurridos 5 segundos, el brazo gira en sentido anti-horario hasta la posición 1 y se para. El ciclo se puede repitir de nuevo, cuando se pulse el botón de arranque.

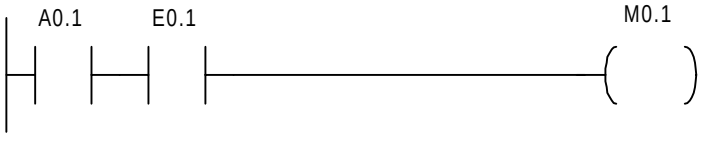


E0.0	Botón de arranque
E0.1	Posición 1
E0.2	Posición 2
A0.0	Giro en sentido horario
A0.1	Giro en sentido antihorario

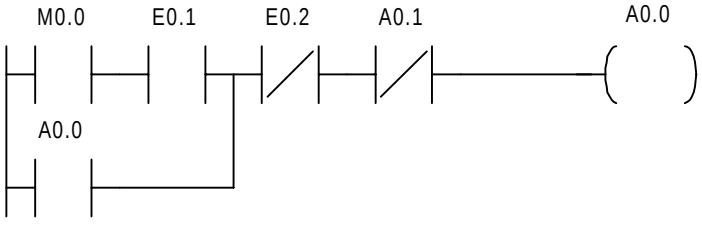
Solución al ejercicio 1



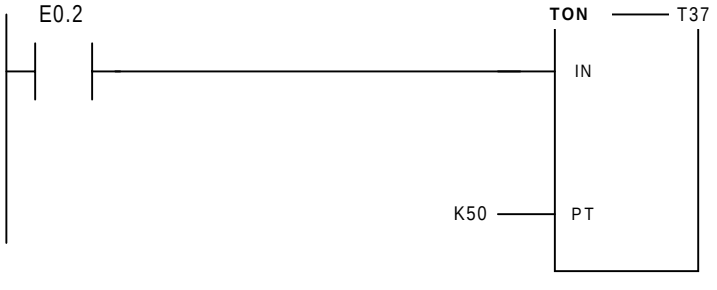
Enclavamiento del relé secuencia de arranque (M0.0) cuando el operador pulsa el botón de arranque (E0.0) y el brazo se encuentra en la posición 1



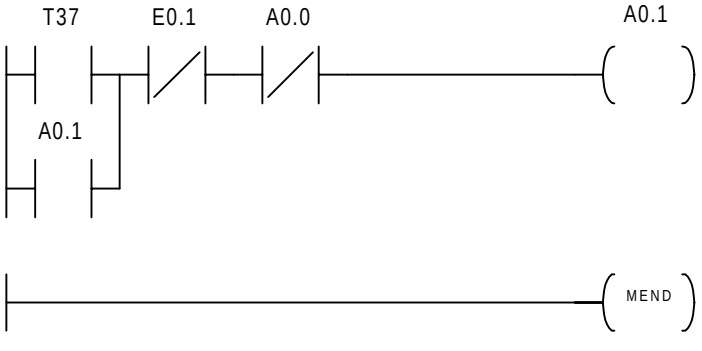
La secuencia termina cuando el brazo, girando en sentido anti-horario (A0.1) alcanza la pos. 2



Enclavamiento sobre el relé de rotación del motor en sentido horario A0.0 una vez que la secuencia ha comenzado (M0.0) y el brazo está en posición 1



Una vez que el brazo alcanza la posición 2 (E0.2), esperar 5 segundos hasta que el brazo empiece a volver a la posición 1.



Enclavamiento en la rotación del brazo en sentido anti-horario (A0.1) una vez que el brazo está en la pos. 2 durante el tiempo especificado (T37). Desenclavar una vez que el brazo llega a la posición 1 (E0.1)

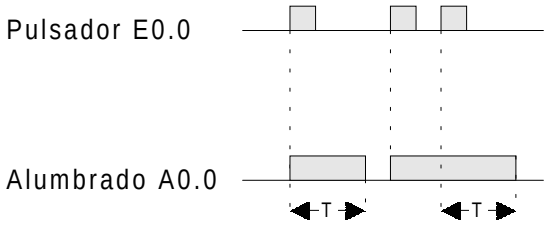
Ejercicio 2:

Alumbrado de una escalera temporizada

Descripción:

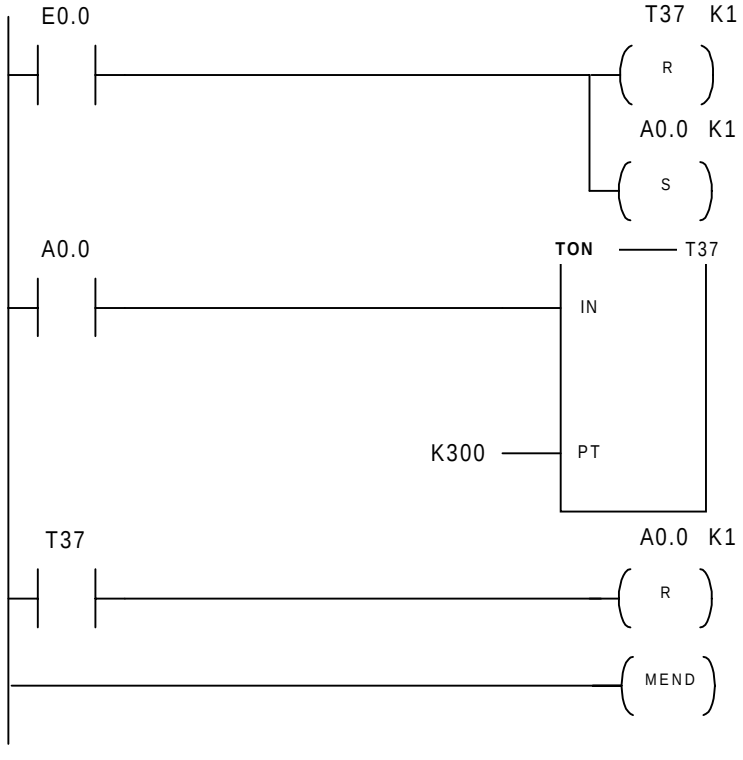
Este ejemplo de programa sirve para el alumbrado de una escalera. Los pulsadores de conexión en las distintas plantas están todos conectados sobre la entrada E0.0 del control. Tras accionar un pulasdor de conesión, se conecta la luz durante un tiempo de 30 segundos, y en ese momento se activa la salida A0.0 prevista. Si durante este tiempo se acciona de nuevo un pulsador de conexión, el intervalo de tiempo comienza de nuevo desde el principio. De esta forma se asegura que la luz se apaga transcurridos 30 segundos después de la última pulsación del pulsador.

Diagrama de impulsos



Solución al ejercicio 2:

Edición en KOP



Activa la entrada , se resetea el temporizador para que comience a contar desde el principio. Al mismo tiempo se activa la salida.

Cuando A0.0 = 1 comienza la temporización de 30 segundos

Una vez finalizado el tiempo de alumbrado reseteamos la salida.

Fin de programa

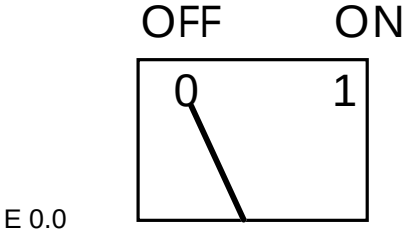
Ejercicio 3

Automatización de una escalera mecánica

El control del motor de una escalera automática consta de un interruptor de encendido y apagado (ON/OFF), un sensor de temperatura para detectar sobrecalentamientos y una célula fotoeléctrica a la entrada de la misma para detectar el paso de personas.

Se desea diseñar el control de funcionamiento de la misma teniendo en cuenta que el tiempo estimado en recorrer todo el trayecto es 5 seg.

Señales de la instalación :



E 0.1 → Sensor térmico.

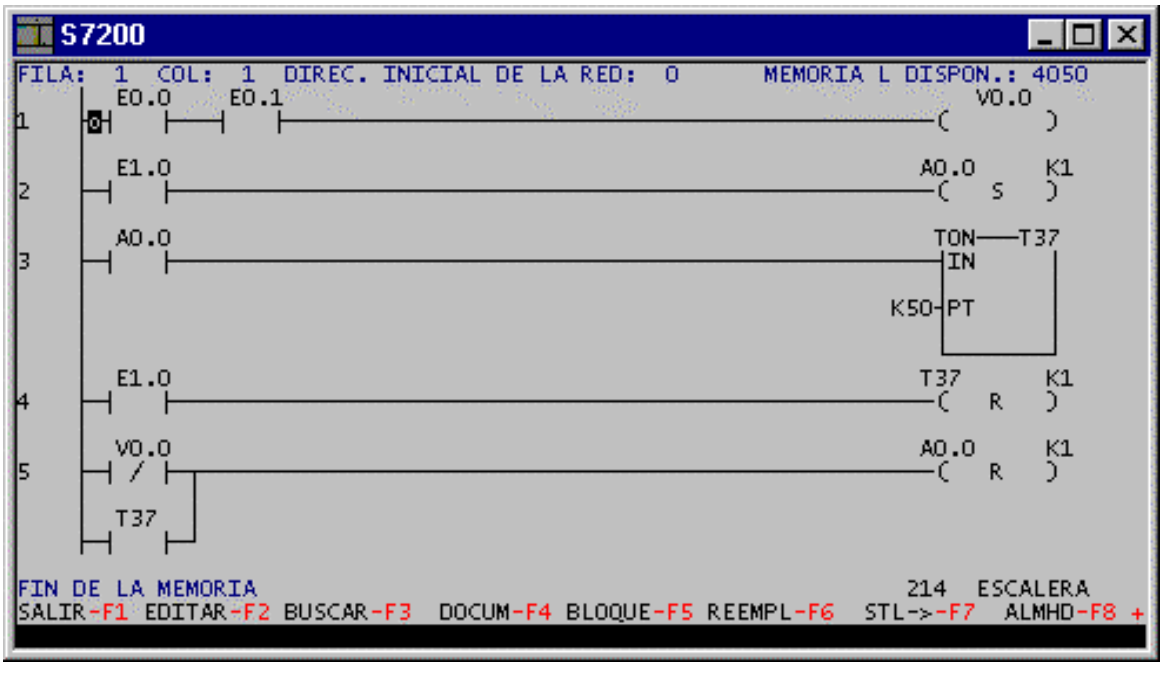
E 1.0 → Fotocélula.

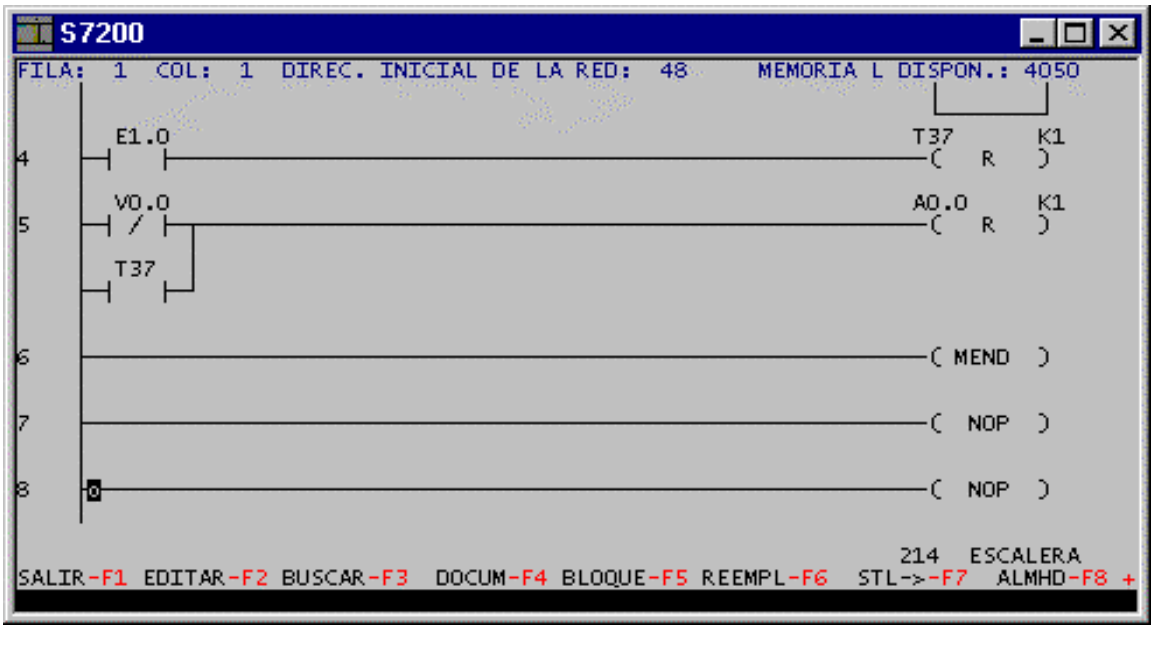
Solución al ejercicio 3

Usamos la memoria auxiliar V 0.0 para establecer las condiciones de funcionamiento (Interruptor = ON, Térmico = 1).

La fotocélula E 1.0 es la encargada de setear la salida del motor A 0.0.

Esta salida se reseteará bien cuando caigan las condiciones de funcionamiento (V 0.0) o bien cuando el temporizador (TON 37) termine de contar los 50 seg. El temporizador se cargará cada vez que la fotocélula detecte presencia para evitar que nadie se quede a mitad de recorrido.





Ejercicio 4

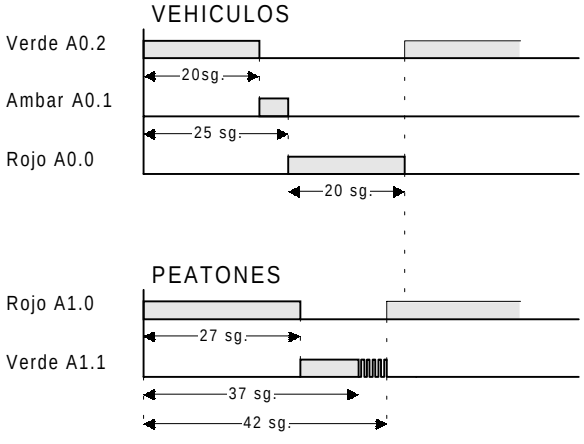
Automatización del control de un cruce con semáforo para peatones

Se desea diseñar el programa de control de un semáforo, el cual dispone de un pulsador de arranque (E0.0) para iniciar la secuencia semafórica y otro (E0.1) para volver a las condiciones iniciales.

El funcionamiento es el siguiente:

Estado normal, los semáforos de vehículos están verdes y los de peatones están rojos.

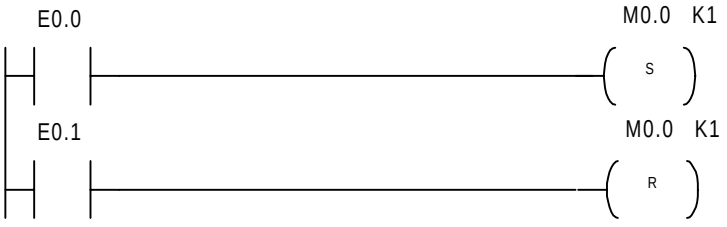
La secuencia es la siguiente:



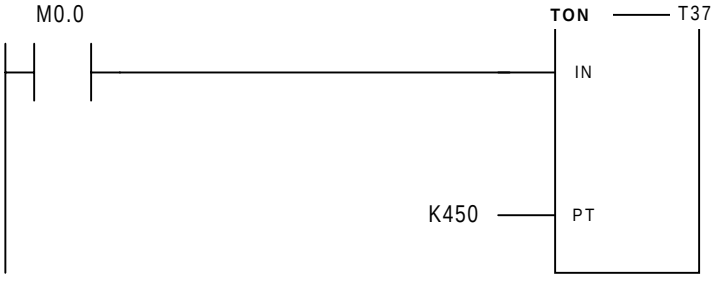
1. La luz verde de el semáforo de los vehículos estará activa durante 20 segundos.
2. La luz ambar se encenderá durante 5 segundos, inmediatamente que se apage la luz verde.
3. La luz roja de los semáforos de vehículos se encenderá durante 20 segundos.
4. La luz roja de los semáforos de peatones estará encendida 7 segundos más que la verde de los vehículos.
5. Una vez que se apaga la luz roja de los peatones se encendará la luz verde de forma continua durante 10 segundos para despues parpadear con una frecuencia de 1 segundo durante 5 segundos.

E0.0	Pulsador de arranque
E0.1	Pulsador de desconexión
A0.0	Luz verde para vehículos
A0.1	Luz amarilla para vehículos
A0.2	Luz roja para vehículos
A1.1	Luz verde para peatones
A1.0	Luz roja para peatones

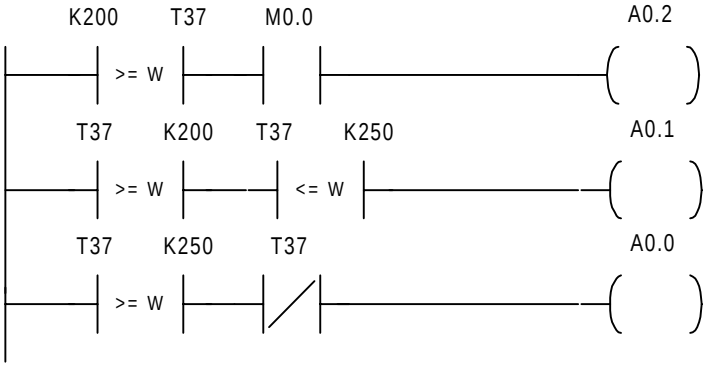
Solución al ejercicio 4



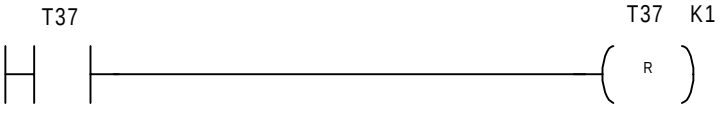
Con los pulsadores E0.0 y E0.1 se logra el arranque de la secuencia semafórica ó la desconexión y vuelta a las condiciones iniciales



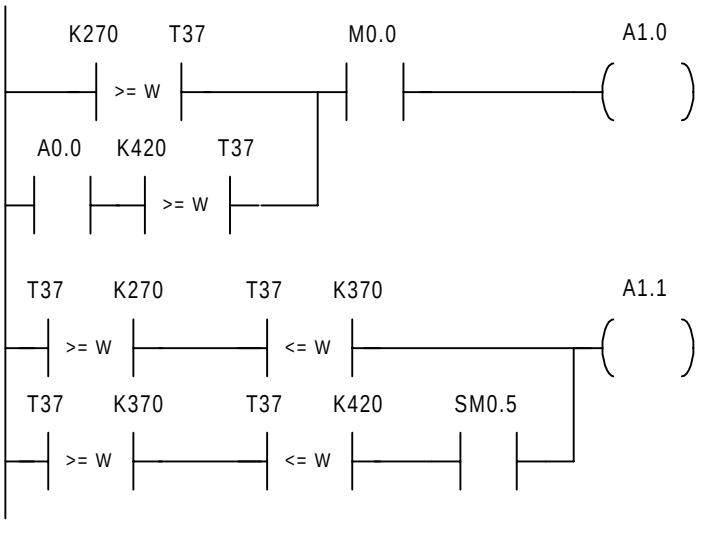
Una vez seteada la marca M0.0, esta activa el temporizador TON, con un tiempo de contaje de 45 segundos (tiempo a partir del cual se vuelve a repetir toda la secuencia)



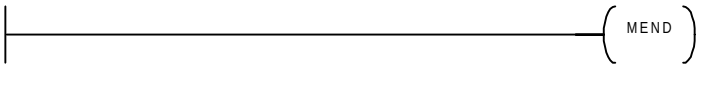
Mediante la comparación de el tiempo de contaje de nuestro temporizador con los tiempos constantes deseados se logran activar las diferentes luces de el semáforo de vehículos



Una vez finalizado el tiempo de contaje, se vuelve a resetear le temporizador para establecer condiciones inicales



De la misma forma que para el semáforo de vehículos, establecemos mediante diversas comparaciones las salidas correspondientes a el semáforo de peatones.



Ejercicio 5

Arranque estrella - triángulo con acuse de recibo de los contactores

Objetivo

Este ejemplo de programa controla el arranque estrella - triángulo de un motor asíncrono de corriente trifásica. Después de accionar el pulsador de arranque conectado sobre la entrada E0.0, el motor arranca en estrella. Tras la finalización del tiempo ajustado de 5 s, el motor conmuta a triángulo. En el caso de que el contactor estrella este defectuoso, se detecta dicha avería mediante un retroacuse del contactor de estrella, y por tanto transcurridos los 5 segundos el SIMATIC S7-200 no pasa a la fase de triángulo, evitando así averías mayores.

Descripción

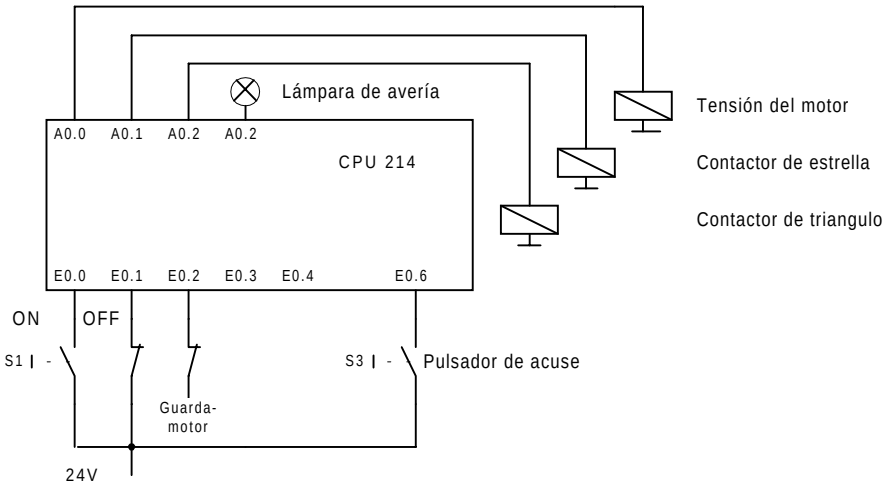
Tras accionar el pulsador de conexión cableados sobre la entrada E0.0, el motor arranca en estrella. Después de que el tiempo de 5 seg. Ajustado haya transcurrido se conmuta el motor a triángulo, siempre y cuando no se encuentre un aviso de avería del contactor. Un aviso de avería se indica mediante una lámpara conectada en la salida A0.3. Con el pulsador de acuse en la entrada E0.6 se puede resetear el aviso de avería, después de haber subsanado la misma interferencia. El acuse de recibo del contactor tiene lugar en las entradas desde E0.3 hasta E0.5.

Si se acciona el pulsador de desconexión o el guardamotor, los cuales están ubicados en las entradas E0.1 y E0.2, se desconecta el motor. En el caso de que sean accionados los pulsadores de conexión y desconexión al mismo tiempo, el motor permanece desconectado.

Para la evaluación del acuse de recibo se comparan los estados de las señales de sallida con los de las entradas, sobre las cuales están realmente depositados los estados de los contactores.

Los estados de estas salidas se comparan con los estados de las entradas de retroaviso E0.3 para el contactor de red, E0.4 para el contactor de estrella, y E0.5 para el contactor de triángulo. En caso de desviación se pone en marcha un tiempo de retardo de 2 seg. En el temporizador T38, el cual corresponde al tiempo de conexión máximo del contactor.

Si transcurrido ese tiempo los estados son todavía diferentes, se activa la salida de error A0.3. Dicha salida de error puede ser desconectada mediante un pulsador de acuse cableado sobre la entrada E0.6.



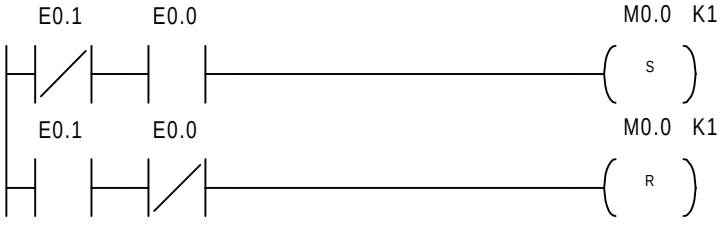
Solución al ejercicio 5:

Entradas/Salidas
en el programa

Símbolo	Descripción
E0.0	Pulsador de conexión
E0.1	Pulsador de desconexión de apertura
E0.2	Protector de motor de apertura (Guardamotor)
E0.3	Acuse de recibo del contactor de red
E0.4	Acuse de recibo del contactor de estrella
E0.5	Acuse de recibo del contactor de triángulo
E0.6	Pulsador de acuse
A0.0	Contactor de red
A0.1	Contactor de estrella
A0.2	Contactor de triángulo
A0.3	Lámpara indicadora de avería
T37	Temporizador de 5 seg. para la conmutación
T38	Temporizador de 2 seg. para aviso de error

Edición en KOP

BLOQUEO



Accionado pulsador de desconexión y accionado pulsador de conexión seteamos marca auxiliar de bloqueo

Si el pulsador de desconexión no es accionado y el pulsador de conexion tampoco liberamos el bloqueo

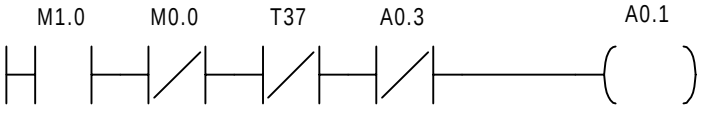
CONEXIÓN

Accionado pulsador de conexión
Sin contactor de red
Contactor de red
Contactor de estrella

Guardamotor OK
Pulsador de conexión no accionado
Sin contactor de triángulo

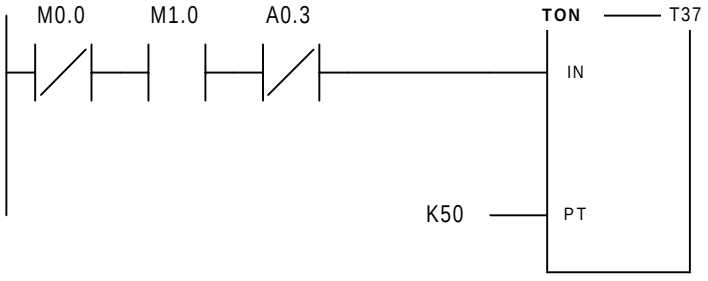
Marca auxiliar

CONECTAR EL CONTACTOR ESTRELLA



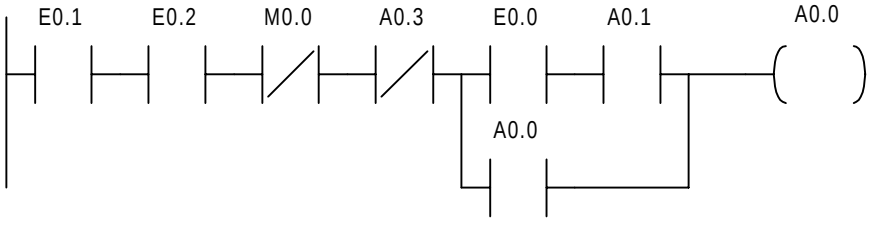
Marca auxiliar
Sin bloquear
Tiempo de conmutación sin transcurrir
Sin aviso de error
Contactor de estrella

ARRANQUE DEL TIEMPO DE CONMUTACIÓN



Sin bloquear
Marca auxiliar
Sin aviso de avería
Arranque del tiempo de conmutación (5seg.)

CONECTAR EL CONTACTOR DE RED



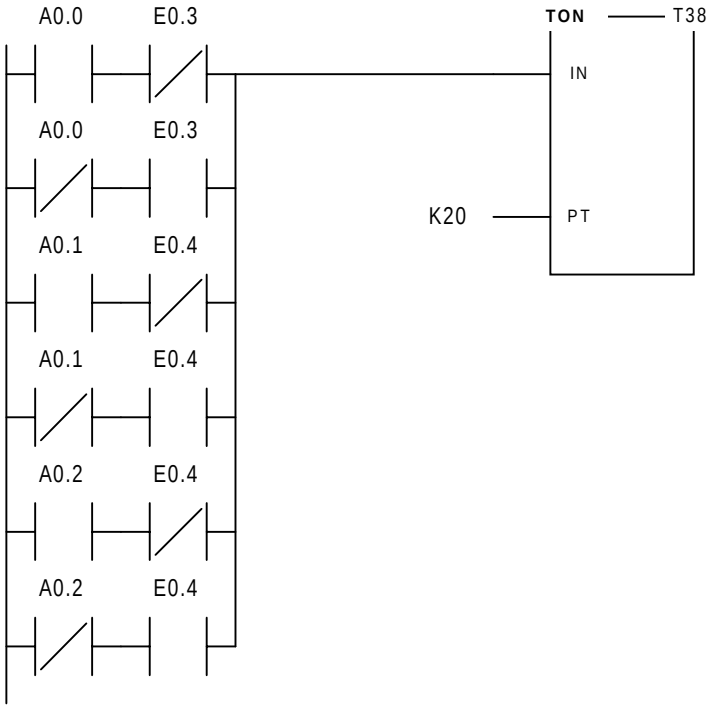
Pulsador de desconexión sin accionar.
Guardamotor OK
Sin bloquear
Sin aviso de avería
Accionado pulsador de conexión
Contactor de estrella
Contactor de red

Contactor de red

CONECTAR EL CONTACTOR DE TRIANGULO

Contactor de red
Contactor de estrella
Contactor de triángulo

ACUSE DE RECIBO DEL CONTACTOR



Contactor de red
Acuse de recibo del contactor de red
Sin contactor de red
Acuse de recibo del contactor de red

Contactor de estrella
Sin acuse de recibo del contactor de estrella

Sin protector de estrella
Acuse de recibo del contactor de triángulo

Contactor de triángulo
Sin acuse de recibo del contactor de triángulo

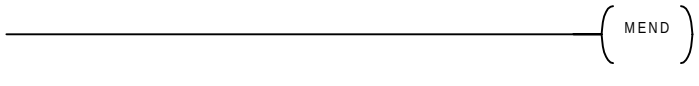
Sin protector triangular
Acuse de recibo del protector triangular

Tiempo de retardo para aviso de avería (2 seg.)

PERTURBACIÓN

PULSADOR DE ACUSE

MEND



Ejercicio 6

Arranque de un motor asíncrono por medio de resistencias rotóricas

Objetivo

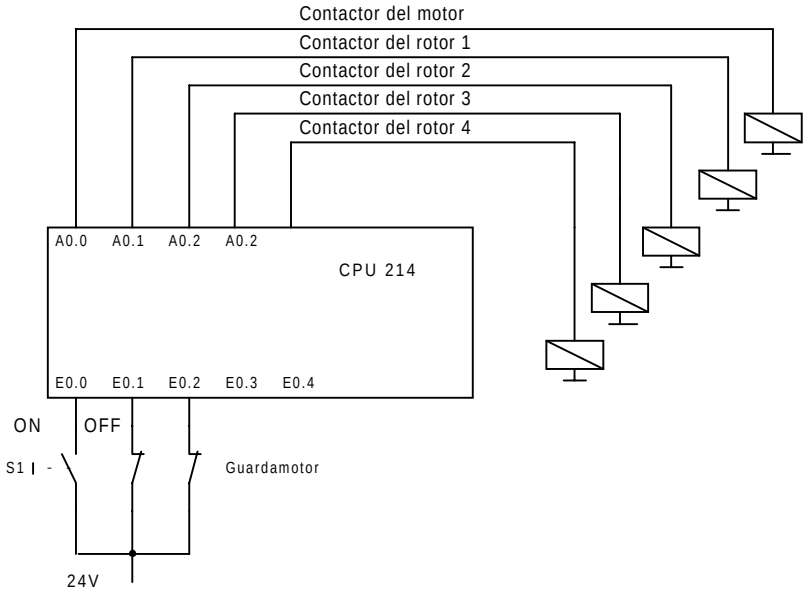
Se desea realizar el arranque de un motor asíncrono de corriente trifásica con un rotor de anillo colector, a través de 4 niveles. El motor arranca aquí a plena resistencia del rotor. Tras el transcurso de un tiempo concreto se cierra el primer contactor del rotor y puentea una parte de la resistencia del rotor. Tras varios intervalos de tiempo se seleccionan los siguientes contactores uno tras otro, en cuyo caso se reduce cada vez mas la resistencia del rotor, hasta que se puentea completamente, y el motor esta en funcionamiento con un número de revoluciones nominal.

Con el pulsador de conexión en la entrada E0.0 tiene lugar este arranque suave del motor, y con el pulsador de desconexión E0.1 se desconecta el motor. En la entrada E0.2 esta conectado el guardamotor, el cual se abre en caso de sobrecarga del motor y le desconecta.

Descripción

Si se acciona el pulsador de conexión en la entrada E0.0 y no se encuentra activo ningún contactor del rotor, o si la autorretención del guardamotor ya ha sido seteada (A0.0=1), se activa la marca intermedia M0.1. Esta marca intermedia se emplea, para activar el contactor de red (A0.0), mientras que no se haya abierto el interruptor de desconexión, o bien el interruptor del guardamotor. La marca de bloqueo M0.0 se activa si se accionan al mismo tiempo los pulsadores de conexión y desconexión; a partir del instante en que se encuentren los pulsadores de nuevo en su posición inicial, esta se desactiva.

Después de haber activado la salida del contactor del motor A0.0, se arranca el temporizador T37. Realizando una comparación entre el tiempo de contaje de T37 y unos valores constantes (en nuestro ejercicio van a ser dos segundos), se consigue accionar cada uno de los contactores que cortocirucitan la resistencia rotórica de los cuatro niveles de arranque. De esta manera la intensidad durante el arranque del motor disminuye eficazmente. En el caso de que durante esta operación se active el pulsador de desconexión todos los contactores deben de desexcitarse hasta que haya una nueva orden de conexión. Ello se consigue mediante el reseteo del temporizador T37.



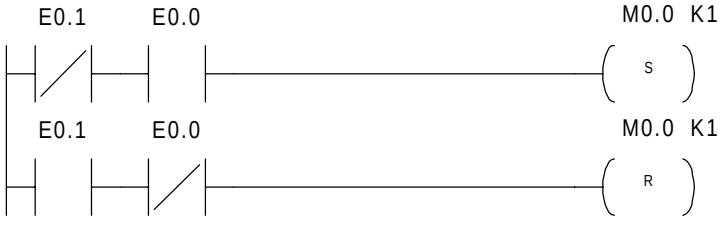
Solución al ejercicio 6

Entradas/Salidas
en el programa

Símbolo	Descripción
E0.0	Pulsador de conexión
E0.1	Pulsador de desconexión contacto de apertura
E0.2	Guardamotor contacto de apertura
A0.0	Contactor de red
A0.1	Contactor del rotor 1
A0.2	Contactor del rotor 2
A0.3	Contactor del rotor 3
A0.4	Contactor del rotor 4
T37	Temporizador de 2 seg. para activar cada uno de los contactores de rotor

Edición en KOP

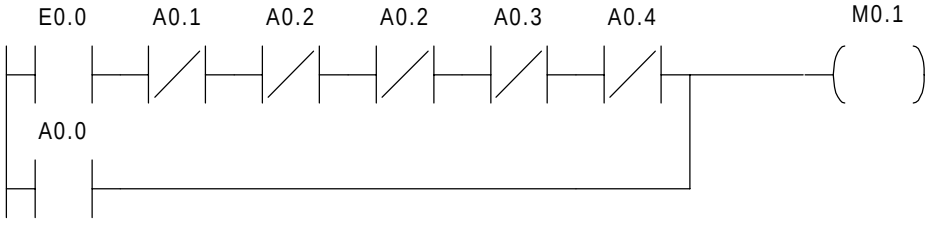
BLOQUEO



Prioridad a la desconexión

El bloqueo se desactiva al volver a las condiciones iniciales

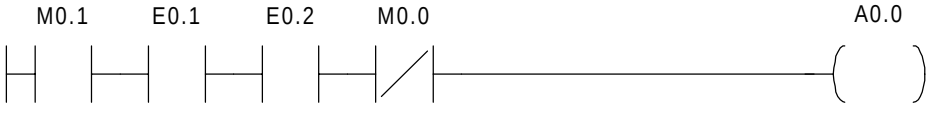
ARRANQUE



Si:
activamos conexión y los contactos de red y de las resistencias rotóricas estan desactividos

= Marca de arranque

CONTACTOR DE RED



Si la marca de arranque esta activa, no se dan las condiciones de bloqueo, no se activa la desconexión y el guardamotor esta OK
= Contactor de red

ARRANQUE DEL TEMPORIZADOR

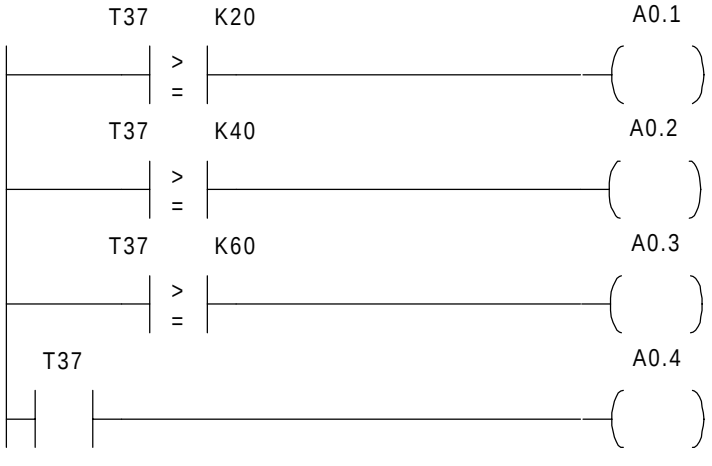
Una vez activa la salida del contactor de red, ésta dispara el temporizador T38 con un tiempo preseleccionado de 8 sg.

RESETEO DEL TEMPORIZADOR

El temporizador se reseteará cuando se active el pulsador de desconexión o bien si el interruptor del guardamotor se encuentra en estado abierto

---

CONEXIÓN DEL ROTOR 1, 2, 3 Y 4



Mediante la comparación del contaje del temporizador con unos tiempos fijos se iran produciendo la activación de cada uno de los contactores de rotor.
Una vez arrancado el temporizador T37 la secuencia es la siguiente:
A los 2 segundos se activa A0.1
A los 4 segundos se activa A0.2
A los 6 segundos se activa A0.3

Cuando el temporizador alcanze el valor de contaje preseleccionado (8 segundos), se activa el último contactor de rotor A0.4.

MEND

Fin de programa
Instrucción MEND

Ejercicio 7

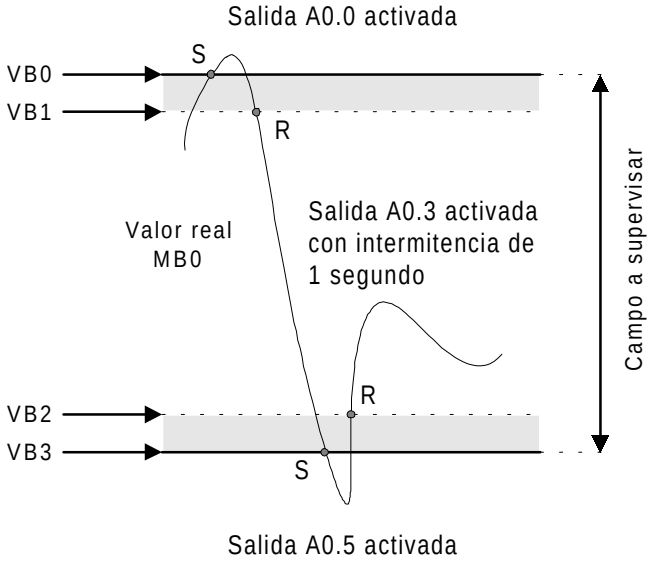
Valoración de límite con histéresis

Descripción:

En una máquina, que se desea controlar, se capta un valor real (por ejemplo una indicación de temperatura), cuyo valor se encuentra en el byte de la marca MB0. Este valor real se ha de supervisar dentro de un campo. Los límites del campo deben de ser modificables. El byte de la marca de variable VB0 contiene el límite superior, el byte de la marca de variable VB3 contiene el límite inferior.

La simulación del valor real se realizará con el potenciómetro 0 del S7-214 .
Si el valor real, que se encuentra en el byte de la marca M0.0, está fuera del campo delimitado por VB0 y VB3, hay que activar las salidas:

- A0.0, cuando el valor real es mayor que el campo permitido y la salida A0.5 cuando el valor real es menor.
- En el caso de que esté comprendido entre los dos valores límites se desea que la salida A0.3 esté activa y desactiva durante 0,5 segundos (Periodo = 1 segundo).



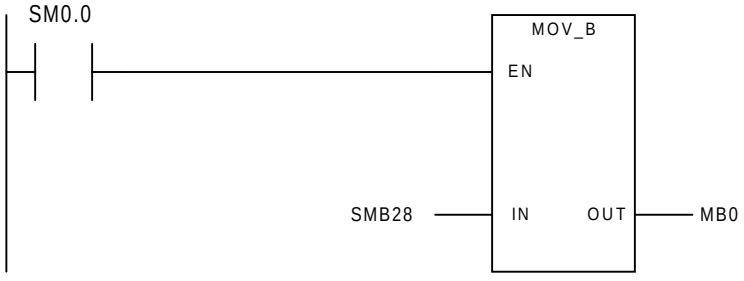
Si el valor real se encuentra dentro de los valores límite, y éste se modifica de forma insignificante, (es decir, varía el valor real dentro de los valores límite), la marca de límite estará permanentemente activando y poniendo a cero (la salida oscila), dado que la precisión de el potenciómetro es de ± 3 uds.

Para evitar esta oscilación de la señal de límite, se crea una histéresis (una banda de tolerancia). La marca límite se activará o pondrá a cero, solamente al alcanzarse el límite de la banda de tolerancia. La banda de tolerancia se crea, de tal forma, que la salida se active cuando el valor real abandone el área permitida; la salida se pondrá de nuevo a cero, cuando haya sobrepasado la histéresis y sobrepasado el campo permitido. Para cada salida existen, por tanto, dos límites de conmutación (ver la figura). El valor de la histéresis se almacena en los bytes : VB1 y en VB2.

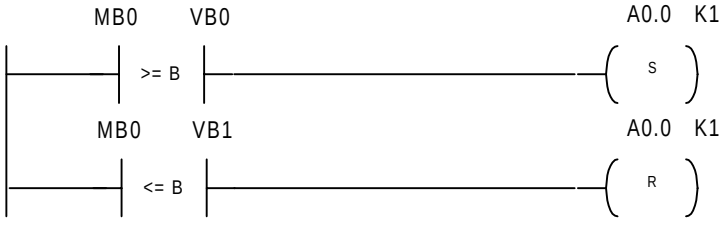
Para el correcto funcionamiento del programa se recomienda la carga de los siguientes valores en los bytes de marcas de variable. Para la introducción de los valores pulse CTRL + V.

VB0	200	Entero
VB1	190	Entero
VB2	110	Entero
VB3	100	Entero

CARGA DEL VALOR REAL EN MB0

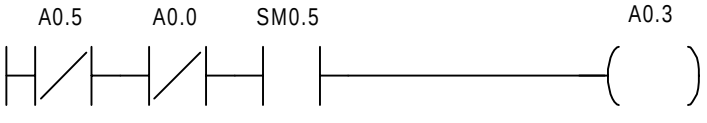


ACTIVACIÓN / DESACTIVACIÓN DE A0.0

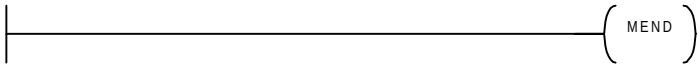


ACTIVACIÓN / DESACTIVACIÓN DE A0.5

VALOR REAL ENTRE LOS VALORES LIMITE SUPERIOR E INFERIOR



FIN DE PROGRAMA PRINCIPAL



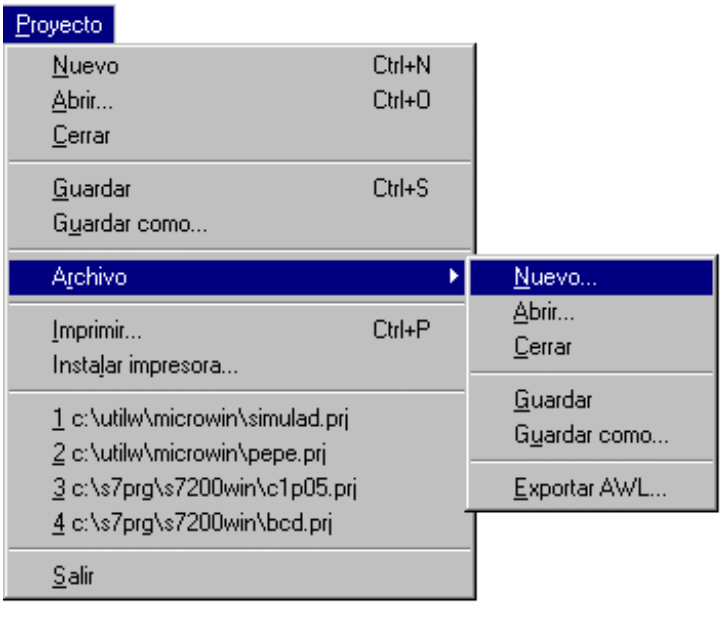
MAPAS DE MENUS

Barra de herramientas:

- Crea un nuevo proyecto
- Abre un proyecto ya existente
- Guarda los cambios del proyecto
- Imprime la ventana activa
- Corta los objetos seleccionados
- Copia los objetos seleccionados en el portapapeles
- Inserta el contenido del portapapeles
- Compila la ventana activa del proyecto
- Carga el proyecto en la PG
- Carga el proyecto activo en la CPU
- Conmuta la CPU a modo RUN
- Conmuta la CPU a modo STOP
- Accede a la Ayuda

MENU PROYECTO:

Los programas del S7-200 y las informaciones correspondientes se guardan en archivos que se crean y se abren con los comandos del menú Proyecto. Este comprende los comandos para crear, abrir y guardar archivos de programa, para imprimir programas y la documentación de los mismos y para salir de Micro/WIN.



Nuevo 

Crea un nuevo proyecto. Eligiendo el comando Nuevo en el menú Proyecto se crean automáticamente los archivos para un bloque de organización, un bloque de datos, una tabla de estado, así como para los comentarios y la configuración. Dichos archivos componen el nuevo proyecto.

El ajuste predeterminado cuando se elige el comando Nuevo es el editor KOP. Allí puede introducir directamente su programa con el lenguaje Esquema de contactos (KOP). Si desea programar mediante la Lista de instrucciones (AWL), conmute al editor AWL con el comando AWL en el menú Ver.

Si desea que el editor AWL sea el ajuste predeterminado, elija el comando Preferencias en el menú Instalar.

Abrir 
Abre el proyecto que se haya seleccionado. Sólo es posible abrir un proyecto por sesión.

Cerrar
Cierra el proyecto actual. La aplicación Micro/WIN permanece activada.

Guardar 
Guarda el proyecto activado con el nombre y en la ruta que se haya indicado en el cuadro de diálogo "Guardar como" al crear el proyecto. Si desea guardar el proyecto por primera vez, Micro/WIN visualiza el cuadro de diálogo "Guardar como".

Para cambiar el nombre o la ruta de un archivo, utilice el comando Guardar como.

Guardar como
Guarda el proyecto activado con el nombre y en la ruta que se haya indicado en el cuadro de diálogo "Guardar como". Utilice el comando "Guardar como" también para cambiar el nombre o la ruta de un proyecto.

Archivo
Los comandos Nuevo, Abrir, Cerrar, Guardar y Guardar como del menú Archivo permiten acceder a cada uno de los archivos de un proyecto. En el cuadro de diálogo "Abrir archivo" es posible elegir la ruta del proyecto y seleccionar allí luego los archivos que desea abrir, o bien, crear un archivo nuevo en dicho proyecto. Si está trabajando en un archivo de proyecto puede guardar o cerrar ese archivo mediante este menú.

Imprimir 
Imprime bien sea el archivo de programa (.ob) o el archivo de bloque de datos (.db) o de tabla de símbolos (.sym) de un proyecto.

Instalar impresora
Visualiza el cuadro de diálogo "Instalar impresora" que contiene opciones para elegir una impresora, cambiar las propiedades de impresión y ajustar la orientación y el tamaño del papel.

Lista de archivos
Muestra una lista de los últimos seis proyectos que se han abierto antes. Puede acceder rápidamente a cualquiera de ellos haciendo clic en el que desee abrir.

Salir
Finaliza la sesión actual, cierra Micro/WIN y retorna al Escritorio. Se le pregunta si desea almacenar los cambios que no haya guardado todavía.

MENÚ EDICIÓN

Cortar

Elimina el área de texto y los gráficos que se hayan seleccionado. Esta función sólo se puede activar si se ha seleccionado previamente un área.

Copiar

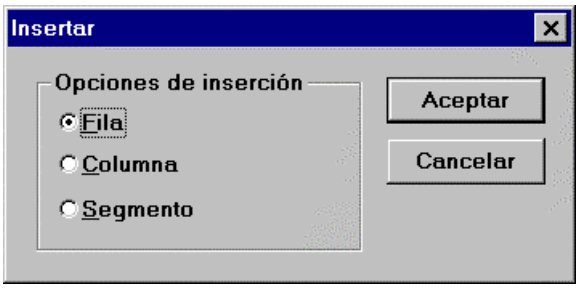
Copia al portapapeles el área de texto y los gráficos que se hayan seleccionado. Esta función sólo se puede activar si se ha seleccionado previamente un área.

Pegar

Inserta una copia del contenido del portapapeles en la posición del cursor.

Insertar

Inserta una fila, una columna o un segmento conforme a lo que se haya seleccionado en el cuadro de diálogo "Insertar"



Borrar

Borra un elemento, una línea vertical, una fila, una columna o un segmento conforme a lo que haya seleccionado en el cuadro de diálogo "Borrar" .

Seleccionar todo

(disponible en los editores de bloques de datos y AWL)

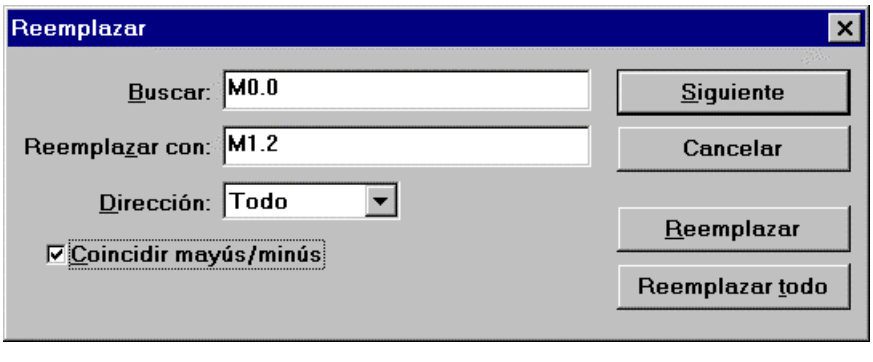
Selecciona todo el texto en la ventana que esté abierta. Luego puede llevar a cabo cualquier otra operación como p.ej. cortar o copiar que será aplicada a todo el texto.

Buscar

Busca el texto que haya introducido en el cuadro de diálogo "Buscar".

Reemplazar

Reemplaza el texto que se haya introducido en el cuadro de diálogo "Reemplazar" .



Insertar fila

(disponible en los editores de tablas de estado y de tablas de símbolos)

Agrega una fila debajo de la posición del cursor en la tabla.

Borrar fila

(disponible en los editores de tablas de estado y de tablas de símbolos)

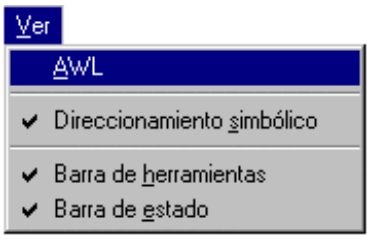
Borra la fila de la tabla en la que esté situado el cursor.

Borrar tabla

(disponible en los editores de tablas de estado y de tablas de símbolos)

Borra todas las filas y todos los datos de la tabla.

MENÚ VER

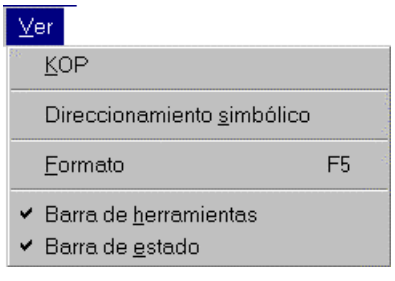


Con los comandos del menú Ver se ajusta qué informaciones se han de visualizar en la ventana activa.

KOP
Muestra el editor KOP. Si está ajustado este modo de visualización, utilice el lenguaje Esquema de contactos para introducir su programa.

AWL
(Lista de instrucciones)
Muestra el editor AWL. Si está ajustado este modo de visualización, utilice el lenguaje Lista de instrucciones para introducir su programa.

Direccionamiento simbólico
Conmuta la visualización del programa de direccionamiento absoluto a direccionamiento simbólico.



Formato
(disponible en el editor AWL)

Cuando la visualización de un programa se conmuta de KOP a AWL es posible elegir la opción Formato (F5) para optimizar la visualización en AWL.

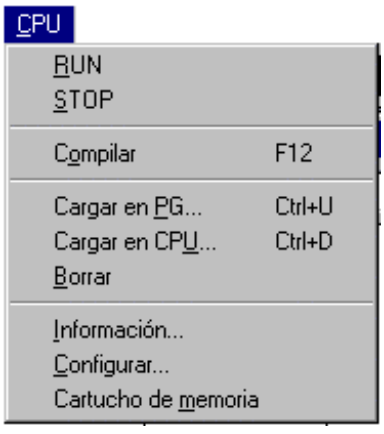
Barra de herramientas
Si está marcada, se visualiza la barra de herramientas de Micro/WIN . Para ocultar la barra de herramientas, retire la marca de verificación seleccionándola nuevamente en el menú Ver.

Barra de estado
Visualiza los mensajes de la aplicación. Además, indica si está activado el modo de inserción o de sobreescritura, así como el número de la línea en la que está situado el cursor (en el caso de los editores AWL y de bloques de datos).

Organizar por nombres
(disponible en la tabla de símbolos)
Organiza en orden alfabético los nombres de los símbolos contenidos en la tabla actual.

Organizar por direcciones
(disponible en la tabla de símbolos)
Organiza en un orden numérico predefinido las direcciones contenidas en la tabla de símbolos actual.

CPU



Este menú le ofrece funciones de comunicación para su CPU. Los comandos del menú CPU permiten cambiar el modo de operación de la CPU, así como compilar, cargar programas de y en la CPU, borrar la memoria de la CPU, leer el modelo de CPU y la información respecto a la misma, así como programar el cartucho de memoria.

RUN

Conmuta la CPU a modo RUN. Antes de intentar cambiar el modo de operación, coloque el selector de la CPU en posición TERM.

STOP

Conmuta la CPU a modo STOP. Antes de intentar cambiar el modo de operación, coloque el selector de la CPU en posición TERM.

Compilar

Compila el bloque lógico y el bloque de datos del programa en lenguaje máquina para que sea ejecutado por la CPU. Para poder cargar un programa en la CPU es necesario compilarlo previamente.

Cargar en PG

Copia el programa de la CPU en el proyecto que esté abierto actualmente. Entonces es posible guardar el programa en forma de archivo. Los archivos a ser cargados que seleccione son los del proyecto actual de la CPU.

Si un programa se carga de la CPU sólo se podrá visualizar y editar en AWL.

Cargar en CPU

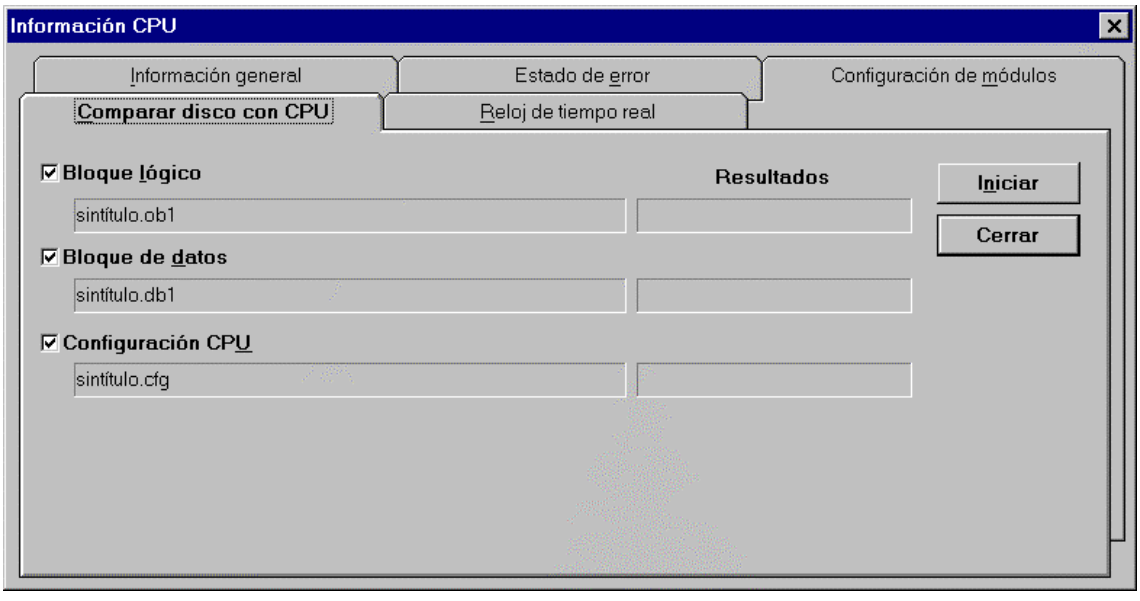
Copia en la CPU el proyecto de los archivos que Vd. haya indicado. Los archivos que se cargan forman parte del proyecto que está abierto actualmente. Tenga en cuenta que el selector de modos de operación de la CPU debe estar en posición TERM y que la CPU se debe conmutar a modo STOP antes de iniciar el proceso de carga. Si el selector de la CPU se encuentra en posición TERM, es posible conmutar la CPU a modo STOP mediante Micro/WIN.

Borrar

Borra toda la información sobre el proyecto que se haya cargado en la CPU y conmuta la CPU a modo STOP. Borra todos los parámetros de configuración excepto la dirección de estación.

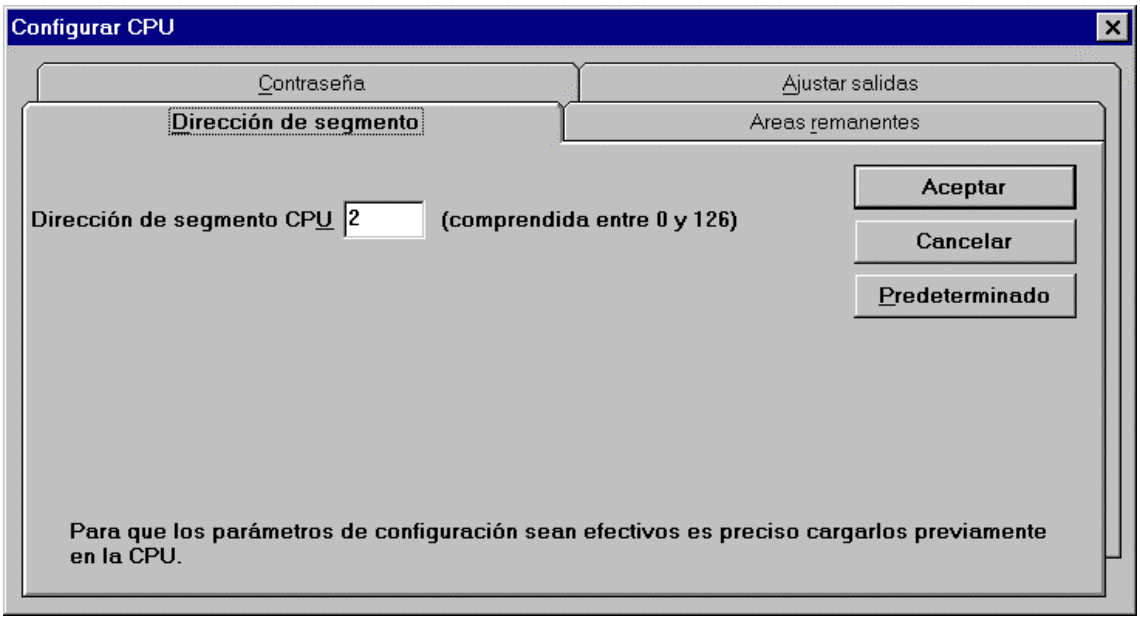
Información

Muestra el modelo y la versión de la CPU, el modo de operación, la velocidad de ciclos, el reloj de tiempo real y la configuración de los módulos E/S, así como una lista de errores de la CPU y los módulos E/S. Asimismo es posible comparar el proyecto abierto actualmente con el proyecto contenido en la CPU.



Configurar

Muestra la configuración actual de la CPU por lo que respecta a la dirección de red, a los ajustes de las salidas, a las áreas remanentes y a la contraseña. Cada uno de dichos ajustes se puede configurar en la pantalla del ajuste actual.

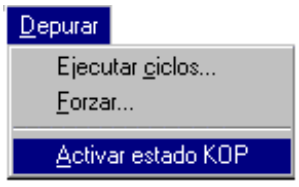


Cartucho de memoria

(sólo en la CPU 214)

Copia el proyecto contenido en la CPU en el cartucho de memoria. Dicha copia incluye el programa de usuario, el bloque de datos, la configuración de la CPU y todas las E/S que se hayan forzado.

DEPURAR



Ejecutar ciclos

Visualiza un cuadro de diálogo en el que es posible indicar el número de ciclos a ejecutar. La CPU ejecuta el número de ciclos que se haya introducido y pasa luego a modo STOP.

Forzar tabla

Permite crear una tabla de direcciones e introducir los valores a escribir en las correspondientes direcciones en la CPU. Los valores contenidos en dichas direcciones se forzan al valor que haya introducido hasta que se desfuercen de nuevo. Las direcciones adoptan luego los valores conformes a la ejecución del programa.

Activar estado

KOP

Muestra el estado del segmento visualizado. Desplácese a otra posición para apreciar el estado de otro segmento.

Tabla

Si se encuentra en un archivo de tabla, se lee el valor actual de las direcciones indicadas. Entonces puede introducir valores en la columna "Cambiar valor en" para las direcciones que desee ajustar con un valor determinado.

Lectura sencilla

Si elige este comando de menú, la CPU lee una sola vez las direcciones que haya introducido en su tabla de estado y visualiza los valores correspondientes.

Escribir

Si elige este comando de menú, la CPU escribe los valores que Vd. haya introducido para las direcciones que aparecen en la tabla de estado.

MENÚ INSTALAR

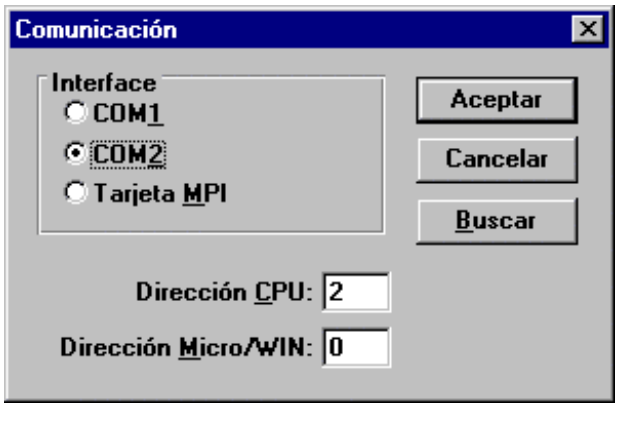
El menú Instalar comprende funciones para seleccionar los ajustes predeterminados (preferencias) y para configurar la comunicación de la CPU. Es posible cambiar los siguientes ajustes predeterminados: lenguaje de programación, nemotecnia, idioma y parámetros de los bloques de datos. El interface de comunicación de la CPU y la dirección de Micro/WIN se configuran mediante el comando Comunicación.

Preferencias

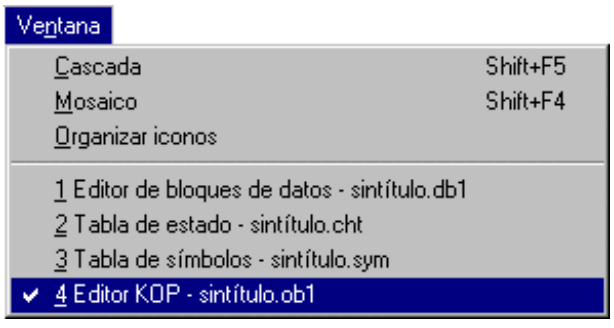
Permite seleccionar los ajustes predeterminados en lo relativo al editor de programación, al interface de comunicación y al formato de visualización de los datos cargados.

Comunicación

Permite acceder al cuadro de diálogo para ajustar la comunicación con la CPU. Con este comando puede configurar el número de interface y la dirección de Micro/WIN.



VENTANA



Este menú le permite organizar las ventanas y los iconos de Micro/WIN y visualizar una lista de las ventanas de Micro/WIN que estén abiertas o minimizadas.

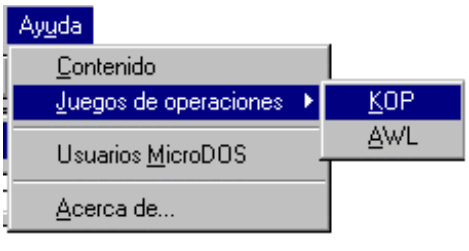
Cascada
Organiza todas ventanas abiertas de forma superpuesta, dejando a la vista las barras de títulos de todas ellas. Haga clic en cualquier barra de título para activar la correspondiente ventana.

Mosaico
Organiza las ventanas abiertas de manera que todas queden visibles. Puede maximizar cualquier ventana para ver todo su contenido.

Organizar iconos
Sitúa los iconos de ventanas minimizadas en una línea horizontal a lo largo del borde inferior de la pantalla.

Lista de ventanas abiertas
Visualiza una lista numerada de todas las ventanas abiertas. Seleccione en dicha lista una de las ventanas para activarla.

AYUDA



Visualiza los comandos para acceder a la Ayuda de Micro/WIN. Al desplazarse por la Ayuda puede retornar a la pantalla inicial haciendo clic en el botón "Contenido" de la ventana de Ayuda.

Contenido
Contiene una lista de los temas básicos de la Ayuda de Micro/WIN.

Usuarios MicroDOS
La ayuda para usuarios de MicroDOS se indica conforme a las etiquetas de las teclas de función de MicroDOS. Puesto que para programar con MicroDOS se utilizan las teclas de función, esta ayuda incluye las funciones básicas conforme a las etiquetas utilizadas en dichas teclas.
Funciones de MicroDOS

SALIR, DOCUM, Ayuda online de MicroDOS, DESHCR, BLOQUE, ACTCOM, ENTRAR, REEMPL, ACTSIN, CONFIG, AWL, ESCRHD, ONLINE/OFFLINE, KOP, ESCPLC, COLOR, IMPRIM, CHGVAL, PROGMS, MEMRIA, ST/RUN, UTILES, EDITAR, ESTADO, BUSCAR, TABLA .

Juegos de operaciones
Permite acceder a la Ayuda para las operaciones de KOP y AWL.
Lista alfabética de operaciones KOP

Activar contador rápido
Ajustar reloj de tiempo real
Asignar bobina de salida
Asignar directamente bobina de salida
Asociar interrupción
Borrar primer registro de la tabla
Borrar temporizador de vigilancia
Borrar último registro de la tabla

Buscar valor en tabla
Codificar
Combinación O con palabras dobles

Combinación O con palabras
Combinación O-exclusiva con palabras dobles
Combinación O-exclusiva con palabras
Combinación Y con palabras dobles
Combinación Y con palabras
Comenzar rutina de interrupción
Comenzar subrutina
Comparación <= byte
Comparación <= entero
Comparación <= palabra doble
Comparación <= real

Comparación == byte
Comparación == entero
Comparación == palabra doble
Comparación == real
Comparación >= byte
Comparación >= entero
Comparación >= palabra doble
Comparación >= real
Contacto abierto directo
Contacto abierto
Contacto cerrado directo
Contacto cerrado
Contar adelante/atrás

Contar adelante
Convertir de ASCII a hexadecimal
Convertir de BCD a entero
Convertir de entero a BCD
Convertir de entero de palabra doble a real
Convertir de hexadecimal a ASCII
Decodificar
Decrementar palabra doble
Decrementar palabra
Definir meta
Definir modo para contador rápido
Desasociar interrupción

Desplazar palabra a la derecha
Desplazar palabra a la izquierda
Desplazar palabra doble a la derecha
Desplazar palabra doble a la izquierda
Detectar flanco negativo
Detectar flanco positivo
Dividir enteros de 16 bits
Dividir reales
END (finalizar programa principal)
Escribir en la red
FOR
Habilitar todos los eventos de interrupción

Incrementar palabra doble
Incrementar palabra
Inhibir todos los eventos de interrupción
Inicializar memoria
Invertir bytes de una palabra
Invertir palabra doble
Invertir palabra
Leer de la red
Leer reloj de tiempo real
Líneas horizontales
Líneas verticales
Llamar subrutina
Multiplicar enteros de 16 bits

Multiplicar reales
NEXT
NOT
Operación nula
Poner a 0 directamente
Poner a 0
Poner a 1 directamente
Poner a 1
Raíz cuadrada de números reales
Registrar valor en tabla
Registro de desplazamiento
Restar enteros de 16 bits
Restar enteros de 32 bits
Restar reales

Retorno desde rutina de interrupción
Retorno
Rotar palabra a la derecha
Rotar palabra a la izquierda
Rotar palabra doble a la derecha
Rotar palabra doble a la izquierda
Salida de impulsos
Saltar a meta
Segmento
STOP
Sumar enteros de 16 bits
Sumar enteros de 32 bits
Sumar reales

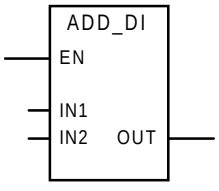
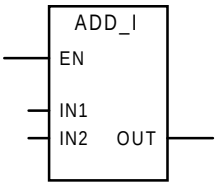
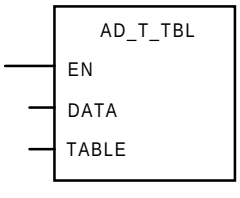
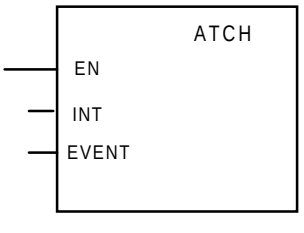
Temporizador de retardo a la conexión memorizado
Temporizador de retardo a la conexión
Transferir byte
Transferir bytes en bloque
Transferir palabra doble
Transferir palabra
Transferir palabras en bloque
Transferir real
Transmitir mensaje
Truncar

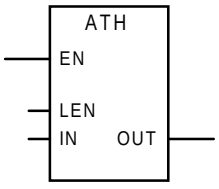
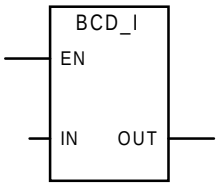
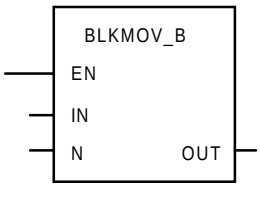
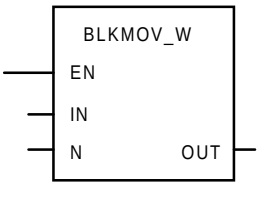
Acerca de

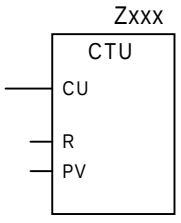
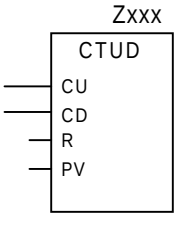
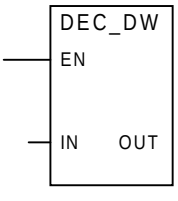
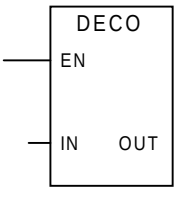
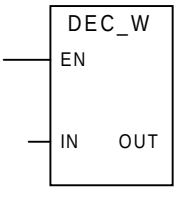
Muestra el número de versión de su copia de Micro/WIN, el copyright e informaciones de carácter legal y relativas a la licencia, así como los datos del sistema de su ordenador.

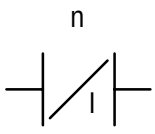
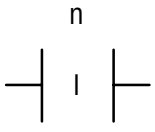
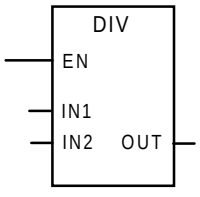
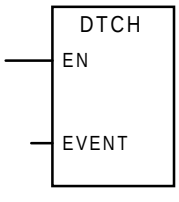
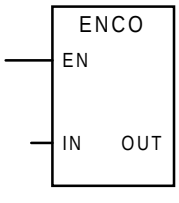
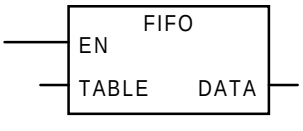
Elementos de programación en KOP

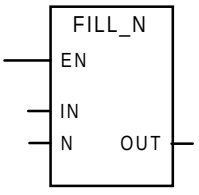
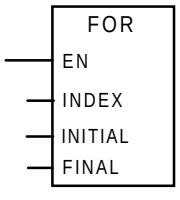
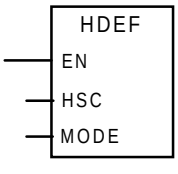
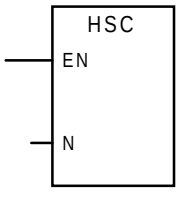
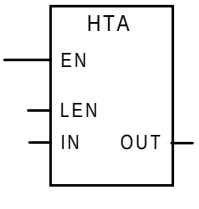
La siguiente tabla muestra los elementos del esquema de contactos así como los símbolos KOP y sus operandos.

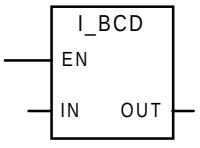
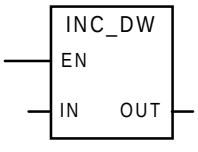
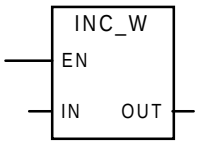
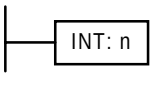
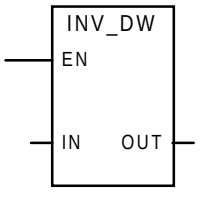
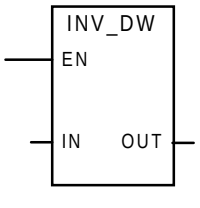
Elemento KOP	Símbolo KOP	Descripción	Operandos
ADD_DI		Sumar enteros dobles (32 bits)	IN1, IN2: (Dpal.) VD, ED, AD MD, SMD, AC, HC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD MD, SMD, AC *VD, *AC
ADD_I		Sumar enteros (16 bits)	IN1, IN2: (Pal.) VW, T, Z, EW, AW MW, SMW, AC, AEW constante, *VD, *AC OUT: (Pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
AD_T_TBL		Registrar valores en una tabla	DATA: (Pal.) VW, T,Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC TABLE: (Pal.) VW, T,Z,EW, AW, MW, SMW,*VD, *AC
ATCH		Asociar interrupción	INT: (Byte) CPU 212: 0-31 CPU 214: 0-127 EVENT: (Byte) CPU 212: 0,1,8-10,12 CPU 214: 0-2

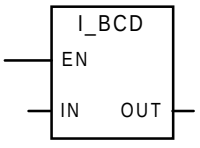
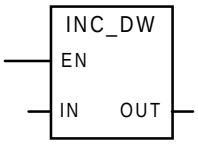
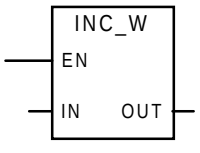
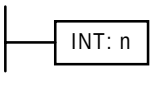
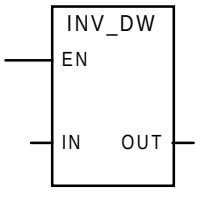
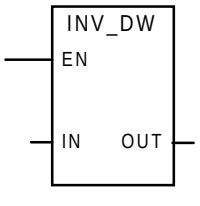
Elemento KOP	Símbolo KOP	Descripción	Operandos
ATH		Convertir de ASCII a hexadecimal	LEN: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC IN: (Byte) VB,EB, AB, MB, SMB, *VD, *AC OUT: (Byte) VB, EB, AB, MB, SMB, *VD, *AC
BCD_I		Convertir de BCD a entero	IN: (Pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW constante, *VD, *AC OUT: (Pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
BLKMOV_B		Transferir bytes en bloque	IN: (Byte) VB, EB, AB, MB, SMB, *VD, *AC OUT: (Byte) VB, EB, AB, MB, SMB, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, constante *VD, *AC
BLKMOV_W		Transferir palabras en bloque	IN: (Byte) VW, T, Z, EW, AW, MW, SMW, AEW, *VD, *AC OUT: (Byte) VW, T, Z, EW, AW, MW, SMW, AAW, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, constante *VD, *AC

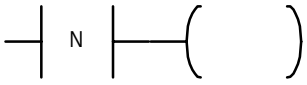
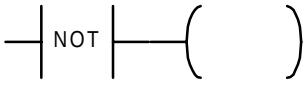
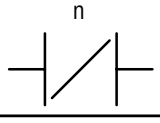
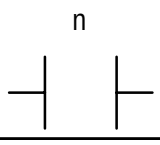
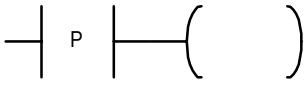
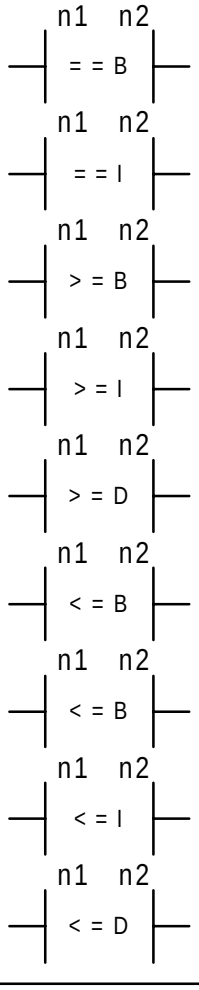
Elemento KOP	Símbolo KOP	Descripción	Operandos
CTU		Contar adelante	Zxxx: (pal.) CPU 212: 0-47; CPU 214: 0-47, 80-127 PV: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
CTUD		Contar adelante/atrás	Zxxx: (pal.) CPU 212: 48-63 CPU 214: 48-79 PV: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
DEC_DW		Decrementar palabra doble	IN: (pal.) VD, ED, AD, MD, SMD, AC, constante, *VD, *AC OUT: (pal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
DECO		Convertir un bit en un número hexadecimal	IN: (byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
DEC_W		Decrementar palabra doble	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, *VD,*AC

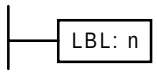
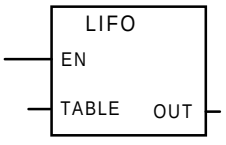
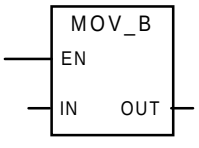
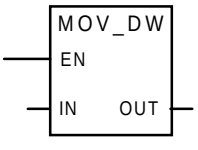
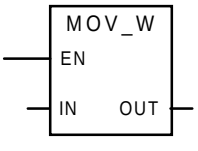
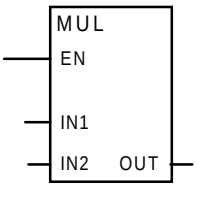
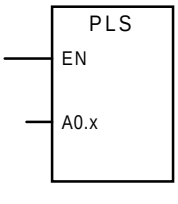
Elemento KOP	Símbolo KOP	Descripción	Operandos
Contacto directo (normalmente cerrado)		Contacto directo normalmente cerrado	n: E (Bit)
Contacto directo (normalmente abierto)		Contacto directo normalmente abierto	n: E (Bit)
DIV		Dividir enteros	IN1: VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC (pal.) IN2: VW, T, Z, EW, AW, MW, SMW, AC, constante, *VD, *AC (pal.) OUT: VD, ED, AD, MD, SMD, AC, *VD, *AC (Dpal.)
DTCH		Desasociar interrupción	EVENT: (Byte) CPU 212: 0,1,8-10,12 CPU 214: 0-20
ENCO		Convertir un número hexadecimal en un bit	IN: VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC (pal.) OUT: VB, EB, AB, MB, SMB, AC, *VD, *AC (Byte)
FIFO		Borrar primer valor de la tabla (FIFO)	TABLE: VW, T, Z, EW, AW, MW, SMW, *VD, *AC (pal.) DATA: VW, T, Z, EW, AW, MW, SMW, AC, AAW, *VD, *AC (pal.)

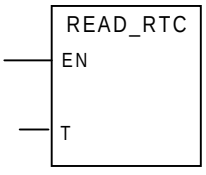
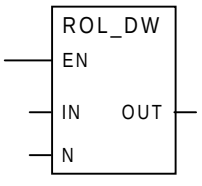
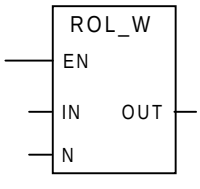
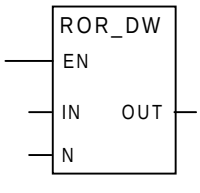
Elemento KOP	Símbolo KOP	Descripción	Operandos
FILL_N		Inicializar memoria	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AEW, constante, *VD, * AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AAW, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante *VD, *AC
FOR		Cuadro FOR	INDEX (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC INITIAL (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC FINAL (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
HDEF		Definir modo para contador	HSC: (Byte) CPU 212: 0 CPU 214: 0-2 MODE: (Byte) CPU 212: 0 CPU 214: 0 (HSC0) 0-11 (HSC1-2)
HSCN		Activar contador rápido	N: (pal.) CPU 212: 0 CPU 214: 0-2
HTA		Convertir de hexadecimal a ASCII	LEN: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC IN (Byte) VB, EB, AB, MB, SMB, *VD, *AC OUT (Byte) VB, EB, AB, MB, SMB, *VD, *AC

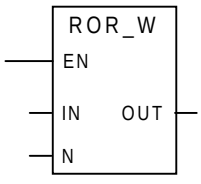
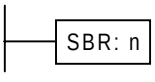
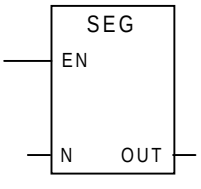
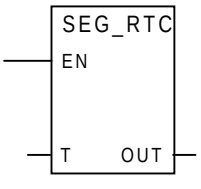
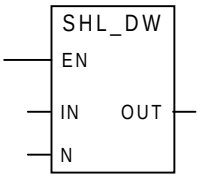
Elemento KOP	Símbolo KOP	Descripción	Operandos
I_BCD		Convertir de entero a BCD	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, * AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
INC_DW		Incrementar palabra doble	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, constante, *VD, *AC OUT: (Dpal:) VD, ED, AD, MD, SMD, AC, *VD, *AC
INC_W		Incrementar palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal:) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
INT		Comenzar rutina de interrupción	n: (pal.) CPU 212: 0-31 CPU 214: 0-127
INV_DW		Complemento a 1 de un entero doble (32 bits)	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, constante, *VD, *AC OUT (Dpal.) VD, ED ,AD, MD, SMD, AC *VD, *AC
INV_W		Complemento a 1 de un entero (16 bits)	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC

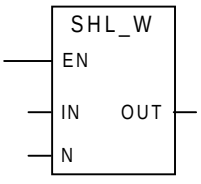
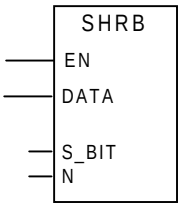
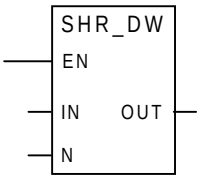
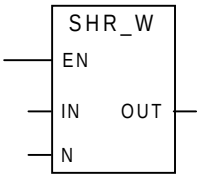
Elemento KOP	Símbolo KOP	Descripción	Operandos
I_BCD		Convertir de entero a BCD	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, * AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
INC_DW		Incrementar palabra doble	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, constante, *VD, *AC OUT: (Dpal:) VD, ED, AD, MD, SMD, AC, *VD, *AC
INC_W		Incrementar palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal:) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
INT		Comenzar rutina de interrupción	n: (pal.) CPU 212: 0-31 CPU 214: 0-127
INV_DW		Complemento a 1 de un entero doble (32 bits)	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, constante, *VD, *AC OUT (Dpal.) VD, ED ,AD, MD, SMD, AC *VD, *AC
INV_W		Complemento a 1 de un entero (16 bits)	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC

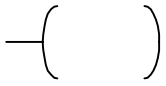
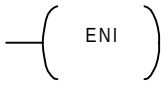
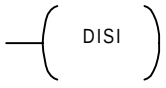
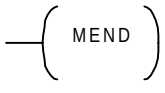
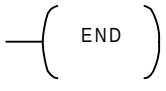
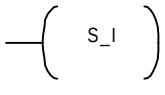
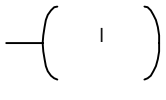
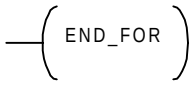
Elemento KOP	Símbolo KOP	Descripción	Operandos
Contacto (flanco decreciente)		Flanco decreciente	ninguno
Contacto (NOT)		Contacto negado	ninguno
Contacto (normalmente cerrado)		Contacto normalmente cerrado se cierra (activa) si n = 0	n: E, A, M, SM, T, (Bit) Z, V
Contacto (normalmente abierto)		Contacto normalmente cerrado secierra (activa) si n = 1	n: E, A, M, SM, T, (Bit) Z, V
Contacto (flanco creciente)		Flanco creciente	ninguno
Contacto (comparación)		Comparar contactos B = byte I = entero (16 bits) D = entero doble (32 bits)	n1, n2: (Byte) VB, IB, AB, MB, SMB, AC, constante, *VD, *AC n1, n2: (pal.) VW, T, Z, EW, AW, MW, SWMW, AC, AEW, constante, *VD, *AC n1, n2: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC

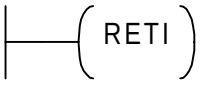
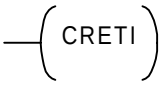
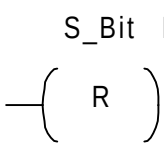
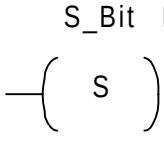
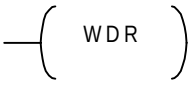
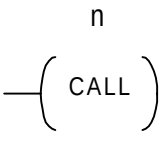
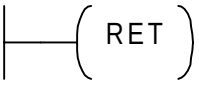
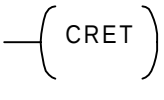
Elemento KOP	Símbolo KOP	Descripción	Operandos
LBL		Definir meta	n: CPU 212: M0-M63 CPU 214: M0-M255
LIFO		Borrar último valor de la tabla (LIFO)	TABLE: (pal.) VW, T, Z, EW, AW, MW, SMW, *VD, * AC DATA: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AAW, *VD, *AC
MOV_B		Transferir byte	IN: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (Byte) VB, EB, AB, MB, SMB, AC, *VD, *AC
MOV_DW		Transferir doble palabra	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC, &VB, &EB, &AB, &MB, &T, &Z OUT: (Dpal:) VD, ED, AD, MD, SMD, AC, *AC
MOV_W		Tranferir palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal:) VW, T, Z, EW, AW, MW, SMW, AC, AAW, *VD, *AC
MUL		Multiplicar enteros	IN1, IN2: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, VD, *AC OUT: (pal:) VD, ED, AD, MD, SMD, AC, *VD, *AC
PLS		Salida de impulsos	A0.x: (pal.) CPU 214 (0-1)

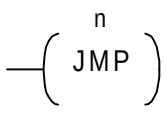
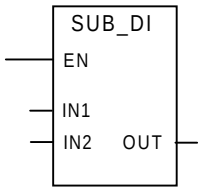
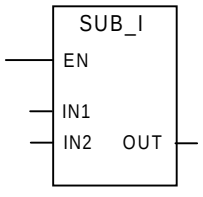
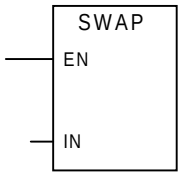
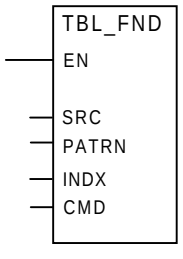
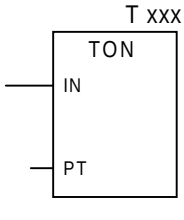
Elemento KOP	Símbolo KOP	Descripción	Operandos
READ_RTC		Leer reloj de tiempo real	T: (Byte) VB, EB, AB, MB, SMB, *VD, *AC
ROL_DW		Rotar a la izquierda palabra doble	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
ROL_W		Rotar a la izquierda palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
ROR_DW		Rotar a la derecha palabra doble	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC

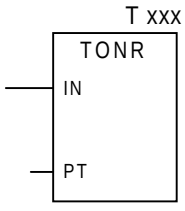
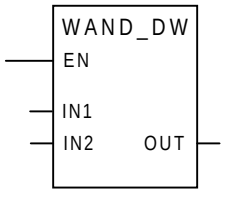
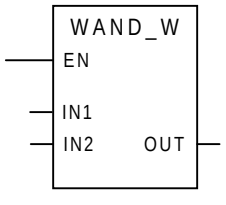
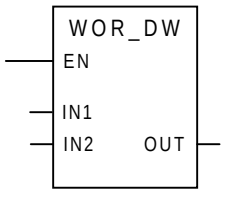
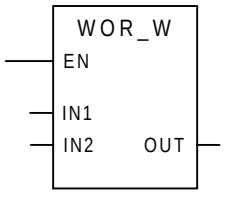
Elemento KOP	Símbolo KOP	Descripción	Operandos
ROR_W		Rotar a la derecha palabra	IN: (pal.) VW, T, Z, EW, AW, AW, MW, SMW, AC, AEW, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
SBR		Comenzar subrutina	n: (pal.) CPU 212: 0-15 CPU 214: 0-63
SEG		Generar configuración binaria para indicador de 7 segmentos	IN: (Byte) VB, EB, AB, MB, SMB, AC, constante *VD, *AC OUT: (Byte) VB, EB, AB, MB, SMB, AC, *VD, *AC
SET_RTC		Escribir en reloj de tiempo real	T: (Byte) VB, EB, AB, MB, SMB, *VD, *AC
SHL_DW		Desplazar a la izquierda palabra doble	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC

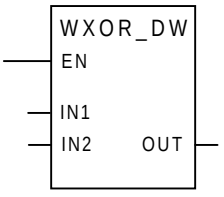
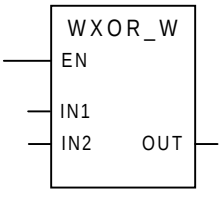
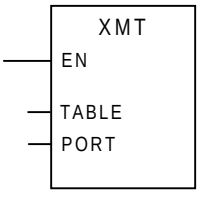
Elemento KOP	Símbolo KOP	Descripción	Operandos
SHL_W		Desplazar a la izquierda palabra	IN: (pal.) VW, T, Z ,EW, AWW, AW, MW, SMW, AC, AEW, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
SHRB		Desplazar bit en registro de desplazamiento	S_BIT: (Bit) E, A, M, SM, T, Z, V N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC
SHR_DW		Desplazar a la derecha palabra doble	IN: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
SHR_W		Desplazar a la derecha palabra	IN: (pal.) VW, T, Z ,EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC N: (Byte) VB, EB, AB, MB, SMB, AC, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC

Elemento KOP	Símbolo KOP	Descripción	Operandos
Bobina	n 	Asignar	n: (Bit) E, A, M, SM, T, Z, V
Bobina Habilitar todos los eventos de interrupción		Habilitar todos los eventos de interrupción	ninguno
Bobina Bloquear todos los eventos de interrupción		Bloquear todos los eventos de interrupción	ninguno
Bobina Fin ejecución	 	Fin absoluto Fin condicional	ninguno
Bobina Poner a 0 directamente	S_Bit N 	Poner a 0 directamente	S_BIT: (Bit) A N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Poner a 1 directamente	S_Bit N 	Poner a 1 directamente	S_BIT: (Bit) A N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Asignar directamente	n 	Asignar directamente	n: (Bit) A
Bobina END_FOR		Bucle con FOR	ninguno

Elemento KOP	Símbolo KOP	Descripción	Operandos
Bobina Retorno absoluto desde rutina de interrupción		Retorno absoluto desde rutina de interrupción	ninguno
Bobina Retorno condicional desde rutina de interrupción		Retorno condicional desde rutina de interrupción	ninguno
Bobina Poner a 0		Poner a 0 (desactivar)	S_BIT: (Bit) E, A, M, SM, T, Z, V N: (Byte) AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina Poner a 1		Poner a 1 (activar)	S_BIT: (Bit) E, A, M, SM, T, Z, V N: (Byte) EB, AB, MB, SMB, VB, AC, constante, *VD, *AC
Bobina STOP		Pasar a modo STOP	ninguno
Bobina Poner a 0 temporizador de vigilancia		Poner a 0 temporizador de vigilancia	ninguno
Bobina Llamar subrutina		Llamar subrutina	n: CPU 212: 0-15 CPU 214: 0-63
Bobina Retorno absoluto desde subrutina		Retorno absoluto desde subrutina	ninguno
Bobina Retorno condicional desde subrutina		Retorno condicional desde subrutina	ninguno

Elemento KOP	Símbolo KOP	Descripción	Operandos
Bobina Saltar a meta		Saltar a meta (JMP)	n: CPU 212: M0-M63 CPU 214: M0-M255
SUB_DI		Restar enteros dobles (32 bits)	IN1, IN2: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
SUB_I		Restar enteros dobles (16 bits)	IN1, IN2: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
SWAP		Intercambiar bytes en palabra	IN: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
TBL_FND		Buscar valor en tabla	SRC: (pal.) VW, T, Z, EW, AW, MW, SMW, *VD, *AC PATRN: (pal.) VW, T, Z, EW, AW, MW, SMW, AEW, constante, *VD, *AC INDX: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC CMD: 0-4
TON		Temporizador de retardo a la conexión	T xxx: (pal.) CPU 212: 32-63 CPU 214: 32-63 96-127 PT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC

Elemento KOP	Símbolo KOP	Descripción	Operandos
TONR		Temporizador de retardo a la conexión con memoria	T xxx: (pal.) CPU 212: 0-31 CPU 214: 0-31, 64-95 PT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC
WAND_DW		Combinación Y con palabras dobles	IN1, IN2: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
WAND_W		Combinación Y con palabras	IN1, IN2: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
WOR_DW		Combinación O con palabras dobles	IN1, IN2: (pal.) VD, ED, AC, MD, SMD, AC, HC, constante, *VD, *AC OUT: (pal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
WOR_W		Combinación O con palabras	IN1, IN2: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC

Elemento KOP	Símbolo KOP	Descripción	Operandos
WXOR_DW		Combinación O-exclusiva con palabras dobles	IN1, IN2: (Dpal.) VD, ED, AD, MD, SMD, AC, HC, constante, *VD, *AC OUT: (Dpal.) VD, ED, AD, MD, SMD, AC, *VD, *AC
WXOR_W		Combinación O-exclusiva con palabras	IN1, IN2: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, AEW, constante, *VD, *AC OUT: (pal.) VW, T, Z, EW, AW, MW, SMW, AC, *VD, *AC
XMT		Transmitir mensaje depositado en búfer	TABLE: (Byte) VB, EB, AB, MB, SMB, *VD, *AC PORT: (Byte) 0

Glosario

A

AC AC (alternating current) es la abreviatura de corriente alterna

Acumulador Un acumulador es uno de los cuatro registros universales que se denominan AC0, AC1, AC2 y AC3.

AE Una entrada analógica es una palabra digital que representa un valor analógico y que está protegida contra escritura.

Analizar La función analizar registra el valor de una dirección definida por el usuario al final de cada ciclo (máximo 124 ciclos). En la CPU 212 se puede activar la función Analizar una vez mientras que en la CPU 214 se puede activar hasta ocho veces.

Aparato central El aparato central pertenece al hardware y abarca la CPU y las entradas y salidas.

Área de datos El área de datos de la S7-200 está dividida en cinco áreas de datos diferentes: imagen de procesos de las entradas y salidas, marcas internas, marcas especiales y memoria de variables.

Área remanente Un área de memoria puede ser programada como área remanente definiendo un área de .. a... Los valores almacenados en el área remanente no se borran al poner en marcha el S7-200, siempre que el condensador de alta potencia haya respaldado el contenido de la memoria RAM.

ASCII El código ASCII (American Standard Code for Information Interchange) sirve para representar información alfanumérica.

Asíncrono No sincronizado ni referido a un flanco de reloj u otros eventos cíclicos.

Autómata programable (PLC)

Un autómata programable es un controlador de lógica programable concebido inicialmente para sustituir los sistemas de control de relés fijamente cableados.

Actualmente, los autómatas programables incorporan una gran variedad de funciones de control. Estas funciones son ejecutadas cíclicamente por el programa creado por el usuario (programa de aplicación).

B

Balance de corriente

En el balance de corriente se averigua la corriente que necesitan los distintos componentes del sistema de control que alimenta el autómata programable S7-200.

BCD

BCD es la abreviatura del formato decimal codificado en binario. Se trata de un código que representa con cuatro bits las cifras decimales 0 a 9, respectivamente.

Bit

Un bit es un dígito binario que puede tener uno de los dos valores siguientes: 0 ó 1 (activado/desactivado, verdadero/falso).

Bobina

Una bobina es un símbolo del lenguaje de programación Esquema de contactos ( ) que representa una bobina de relé que se excita al pasar la corriente.

Byte

Un byte se compone de ocho bits.

C

Cable PC/PPI

El cable PC/PPI permite conectar el puerto RS-485 del S7-200 al puerto RS-232 de un PC estándar.

Cartucho de memoria

El cartucho de memoria (opcional, sólo en la CPU 214) provee espacio de memoria no volátil para el programa.

Ciclo

En la ejecución cíclica, el programa de aplicación se ejecuta en un bucle que va repitiéndose continuamente y que se denomina ciclo. Un ciclo abarca las siguientes operaciones: leer las entradas, ejecutar el programa de aplicación, procesar mensajes, autodiagnóstico, escribir en las salidas. Estas operaciones se ejecutan periódicamente y en un orden determinado.

CLEARPLC La contraseña maestra CLEARPLC permite acceder al autómata programable en caso de desconocerse la contraseña. CLEARPLC tiene que utilizarse con la máxima cautela.

Complemento a 1 El complemento a 1 es la negación lógica de cada uno de los bits de un valor binario.

Comunicación Freeport En la comunicación Freeport (comunicación de programación libre), el programa KOP controla el funcionamiento del puerto de programación mediante interrupciones de recepción (o interrupciones de transmisión) y mediante la operación Transmitir mensaje (XMT).

Condensador de alta potencia El condensador de alta potencia se encarga de alimentar la memoria RAM en caso de que se corte la alimentación del autómata programable, manteniendo los datos por un tiempo determinado sin necesidad de mantenimiento adicional (p. ej. pila). El condensador respalda la memoria tras un corte de alimentación unas 50 horas en la CPU 212 y unas 190 horas en la CPU 214.

Conector de bus El conector de bus permite empalmar la CPU o un módulo de extensión al siguiente módulo enganchado en el soporte. Los conectores se suministran con los módulos de extensión.

Contacto Un contacto es un símbolo del lenguaje de programación Esquema de contactos (KOP). Representa un contacto por el que pasa la corriente cuando se cierra. Los contactos pueden ser contactos normalmente cerrados | / | o contactos normalmente abiertos | |.

Contador Un contador es un elemento que cuenta los flancos crecientes en las entradas de contaje. La CPU 212 dispone de 64 contadores y la CPU 214 hasta un total de 128 contadores.

Contador rápido Un contador rápido puede contar más deprisa de lo que tarda el autómata programable en consultar los eventos. Los contadores rápidos tienen un valor de contaje entero de 32 bits (o valor actual) con signo.

Contraseña La contraseña es un código especial que limita el acceso a las funciones y a la memoria del autómata programable.

Contraseña maestra La contraseña maestra CLEARPLC permite acceder al autómata programable en caso de desconocerse la contraseña. CLEARPLC tiene que utilizarse con la máxima cautela.

Cuadro Un cuadro es un símbolo del lenguaje de programación Esquema de contactos. Representa distintas funciones que se ejecutan cuando la corriente fluye hasta el cuadro. Las funciones representas generalmente por los cuadros suelen ser temporizadores, contadores y funciones aritméticas.

D

DC DC (direct current) es la abreviatura de corriente continua.

Direccionamiento directo El direccionamiento directo cosiste en indicar una dirección de la memoria o la dirección de un elemento. La dirección contiene el valor al que se desea acceder.

Direccionamiento indirecto El direccionamiento indirecto consiste en acceder a una dirección que indica la dirección a la que se desea acceder.

E

EEPROM EEPROM es la abreviatura inglesa de Electrically Erasable Programmable Read Only Memory. Se trata de una memoria no volátil que sirve para almacenar datos.

Entero Un número entero es un número que consta exclusivamente de una o más unidades, a diferencia de los quebrados.

Entrada Una entrada es una información que lee el autómata programable y que utiliza como condición para el control o estado (status).

Entradas y salidas digitales Las entradas y salidas digitales pueden tener uno de los estados de señal siguientes: 0 ó 1 (activado/desactivado; verdadero/falso).

E/S directas Las E/S directas son los valores reales de las entradas y salidas en los módulos E/S. Es decir, son entradas o salidas cuyo valor es leído o escrito al ejecutarse la operación, contrariamente a las entradas y salidas de la imagen de proceso, es decir las E/S que se actualizan al finalizar el ciclo.

Error grave Cuando se produce un error grave, el autómata programable ya no puede ejecutar el programa de aplicación. Según la gravedad del error pueden fallar algunas o bien todas las funciones del autómata.

Error leve Un error leve puede mermar el funcionamiento del autómata programable, pero no lo incapacita para ejecutar el programa de aplicación y actualizar las entradas y salidas.

Esquema de El Esquema de contactos (KOP) es un lenguaje de programación que sirve para programar el autómata programable S7-200. KOP utiliza

contactos (KOP) símbolos del esquema de circuitos tales como contactos, bobinas de relé, elementos o cuadros para representar la lógica de control o programa.

F

Fijador de soporte El fijador se atornilla en le soporte para impedir que resbalen los módulos en el soporte.

Forzar La función Forzar permite forzar los valores de las entradas y salidas o variables independientemente del estado del proceso o del programa.

H

Hexadecimal La representación hexadecimal se basa en un sistema numérico de 16 dígitos.

I

Imagen de proceso de las entradas La imagen de proceso de las entradas es una memoria en la que se deposita el estado de las entradas durante el ciclo. Al principio de cada ciclo se leen los valores de las entradas.

Imagen de proceso de las salidas La imagen de proceso de las salidas es una memoria en la que se deposita el estado de las entradas durante el ciclo. Al principio de cada ciclo se transfieren los valores a las salidas.

Instantánea Una instantánea recoge los valores de hasta ocho direcciones de datos de usuario después de que el autómata haya ejecutado una operación determinada. La CPU 212 puede tomar una instantánea y la CPU 214 hasta un total de 8 instantáneas.

L

LED

Un LED es un diodo luminoso. Los LED sirven de indicadores de estado, es decir, inician el estado actual del aparato central así como el estado de las entradas y salidas.

Línea de corriente del esquema de contactos

Una línea de corriente del esquema de contactos se compone de varios elementos que forman un circuito completo junto con los raíles izquierdo y derecho. El raíl izquierdo representa el conductor excitado. El raíl derecho representa el conductor neutro (en el software de programación STEP 7-Micro no se representa el raíl derecho). La corriente fluye desde el raíl izquierdo a través de los contactos hasta alcanzar las bobinas o cuadros conectados al raíl derecho.

Lista de instrucciones (AWL)

La Lista de instrucciones (AWL) es un lenguaje de programación textual (a diferencia del lenguaje de programación Esquema de contactos (KOP) que imita un esquema de circuitos). AWL se utiliza para programar el autómata programable S7-200. Cada línea del programa AWL contiene una instrucción determinada que contiene uno o más operandos según la operación.

M

Marca interna

La marca interna, también denominada relé de control, ofrece espacio de memoria para informaciones de estado y control temporales.

Marcas especiales

Las marcas especiales proveen funciones de estado y control. Una marca especial permite intercambiar informaciones entre el autómata y el programa.

Memoria de datos

La memoria de datos sirve de área de trabajo y contiene direcciones para cálculos, memoria temporal para resultados intermedios así como constantes utilizadas en comandos o en otros parámetros de control fijos. La memoria de datos contiene además elementos especiales y objetos tales como temporizadores, contadores, contadores rápidos y entradas y salidas analógicas. Una parte de la memoria de datos está almacenada en la memoria no volátil.

Memoria de parámetros

La memoria de parámetros provee espacio de memoria para parámetros configurables como contraseñas, direcciones de estaciones y áreas remanentes. El contenido de la memoria de parámetros se almacena en la memoria no volátil.

Memoria de programa

La memoria de programa contiene la lista de operaciones que ejecuta el autómata programable para implementar la función de control deseada. La memoria de programa abarca 512 palabras en la CPU 212 y hasta un total de 2048 palabras en la CPU 214.

Memoria de sólo escritura

La memoria de sólo escritura puede ser una memoria o elementos cuyo valor puede modificarse pero no leerse.

Memoria de variables

La memoria de variables es una memoria de lectura/escritura que se encuentra en la memoria RAM. Se divide en dos áreas:

- El módulo de datos 1 (DB1) contiene los primeros 128 bytes de la memoria de variables de la CPU 212, o bien los primeros 512 bytes de la memoria de variables de la CPU 214. Los datos del DB1 se guardan en memoria RAM y, cada vez que se carga el DB1, se copian en una memoria interna no volátil.
- El segundo área de la memoria de variables es prácticamente idéntico al DB1. La única diferencia radica en que la memoria interna no volátil no es lo suficientemente grande, por lo que no se puede guardar en la memoria no volátil.

Memoria no volátil

La memoria no volátil no pierde su contenido aun sin aplicar corriente.

Modo de operación

El autómata programable S7-200 dispone de dos modos de operación: STOP y RUN.

Modulación en ancho de impulsos

La función modulación en ancho de impulsos provee un tiempo de ciclo fijo con un factor de trabajo variable relativo.

Módulo de datos 1 (DB1)

El módulo de datos 1 (DB1) representa en la CPU 212 los primeros 128 bytes de la memoria de variables. En la CPU 214 representa los primeros 512 bytes de la memoria de variables.

Módulo de E/S analógicas

Los módulos analógicos convierten dimensiones reales (analógicas) tales como tensión, temperatura etc. en una palabra digital y viceversa. El módulo analógico puede ser un módulo de entradas analógicas, de salidas analógicas o de entradas y salidas analógicas.

Módulo de extensión Un módulo de extensión dispone de entradas y salidas adicionales por lo que permite aumentar el número de entradas y salidas del aparato central (CPU).

O

OB1 El módulo de organización (OB1) contiene la memoria de programa y se encuentra en una memoria interna no volátil.

Objeto Un objeto es la dirección de memoria asignada a un elemento. Los objetos pueden ser temporizadores, contadores, entradas y salidas analógicas, acumuladores y valores actuales de contadores rápidos.

Operando Un operando es el parámetro de una instrucción.

P

Palabra Una palabra consta de 16 bits.

Palabra doble Una palabra consta de 32 bits.

Puerto de extensión del bus El puerto de extensión de bus permite conectar módulos de extensión adicionales.

Pipeline PTO La pipeline PTO es una cadena de definiciones de salidas de impulsos. Una vez creado el primer eslabón de la cadena puede añadírsele la segunda definición.

Primer valor de la pila El primer valor de la pila representa el nivel superior de la pila: La pila tiene nueve niveles de un bit, respectivamente. La pila sirve al autómata para ejecutar la lógica programada.

R

Random Access Memory (RAM) La memoria de trabajo (memoria de lectura/escritura) de la CPU se denomina memoria RAM. Contiene el programa y los datos a los que accede el programa durante su ejecución.

Read Only Memory (ROM) La memoria ROM (memoria de sólo lectura) es una memoria permanente cuyo contenido no puede ser modificado.

Reloj de tiempo real El reloj de tiempo real indica segundos, minutos, horas, así como el día de la semana, el mes y el año.

Rutina de interrupción Una rutina de interrupción es una parte opcional de programa que no se ejecuta en cada ciclo sino solamente cuando se cumple una condición de interrupción.

S

Segmento Un segmento consta de varias operaciones KOP que forman juntas una línea de corriente.

Selector de modo El selector de modo tiene tres posiciones y sirve para elegir el modo de operación del autómata programable.

Soporte de los módulos El soporte para enganchar los módulos cumple con la norma DIN (DIN EN 50 022).

Subrutina Una subrutina es una parte de programa que ha de ser llamada para su ejecución. En el autómata programable S7-200 pueden utilizarse subrutinas aunque no es necesario. Las subrutinas se agregan al final del programa principal.

T

Tasa de baudios La tasa de baudios es una unidad variable que indica la velocidad de transmisión en bits/s.

Temporizadores Un temporizador es un elemento que cuenta incrementos de tiempo. En el S7-200, los temporizadores tienen resoluciones (incrementos)de 1, 10 ó 100 milisegundos. La CPU 212 ofrece 64 temporizadores y la CPU 214 hasta un total de 128.

Temporizador de vigilancia El temporizador de vigilancia sirve para detectar errores. El temporizador funciona de forma continua y es puesto a 0 periódicamente por el autómata programable o por una operación del programa de aplicación. Una vez transcurrido el tiempo programado (porque no fue puesto a 0) se produce un error grave, con lo cual el autómata pasa a modo STOP.

Tierra Se denomina tierra a la masa conductiva cuyo potencial eléctrico se puede desactivar (poner a 0) en cualquier punto.

Tipos de direccionamiento El autómata S7-200 asiste dos tipos de direccionamiento para acceder a los operandos de una instrucción: se pueden direccionar directamente todos los elementos y registros de memoria indicando el área y la dirección. Se pueden direccionar directamente las siguientes áreas de datos: E, A, M, T (valores actuales de temporizadores), Z (valores actuales de contadores) y V.

Tren de impulsos La función Tren de impulsos provee una salida en cuadratura (50% factor de trabajo) para un número de impulsos y un tiempo de ciclo determinados.

U

Unidad de programación PG 702 La PG 702 es una unidad de programación portátil con la que se puede programar en Lista de instrucciones (AWL)

V

Valor de contaje de impulsos En la función PTO, el valor de contaje de impulsos representa el número de ciclos o impulsos de salida.

Varistor metalóxido (MOV) Un varistor metalóxido es un semiconductor que se emplea para proteger otros aparatos electrónicos en caso de sobretensión.

Verificación del programa La verificación o prueba del programa consiste en buscar y corregir errores en el programa y sistema.