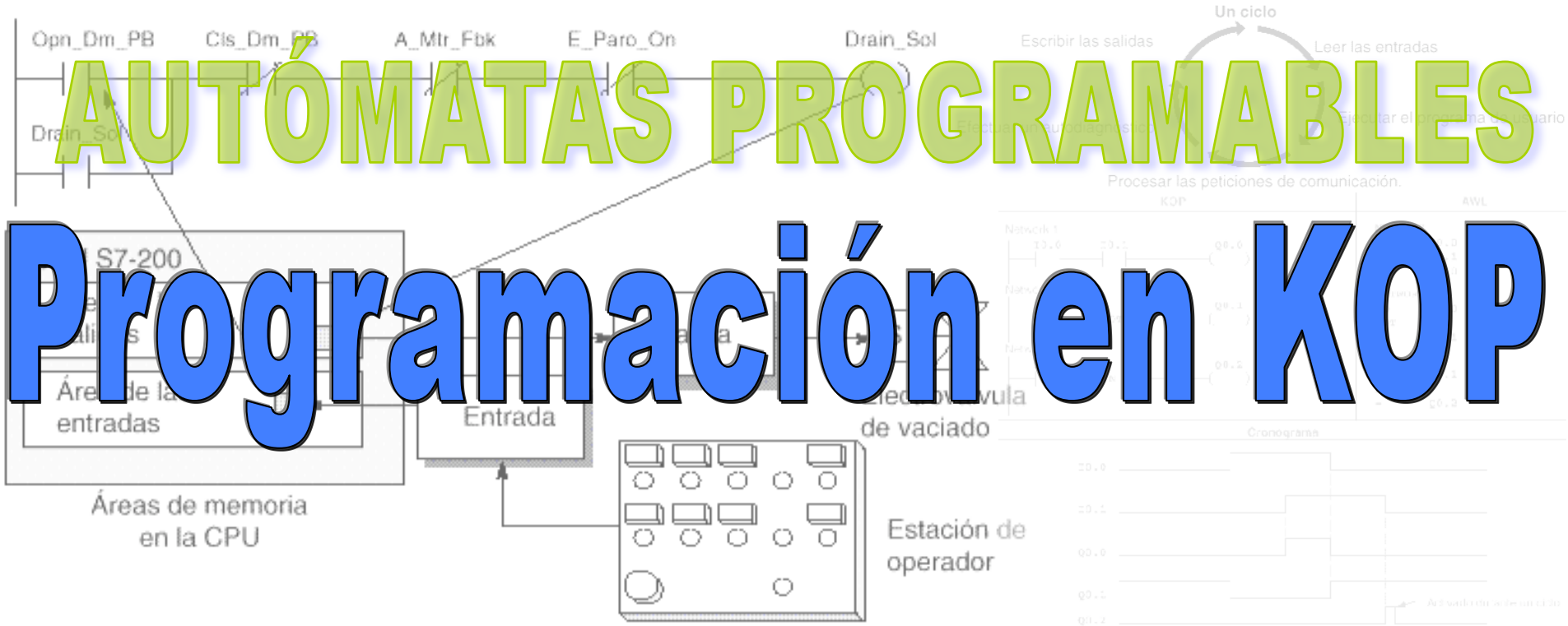


AUTÓMATAS PROGRAMABLES

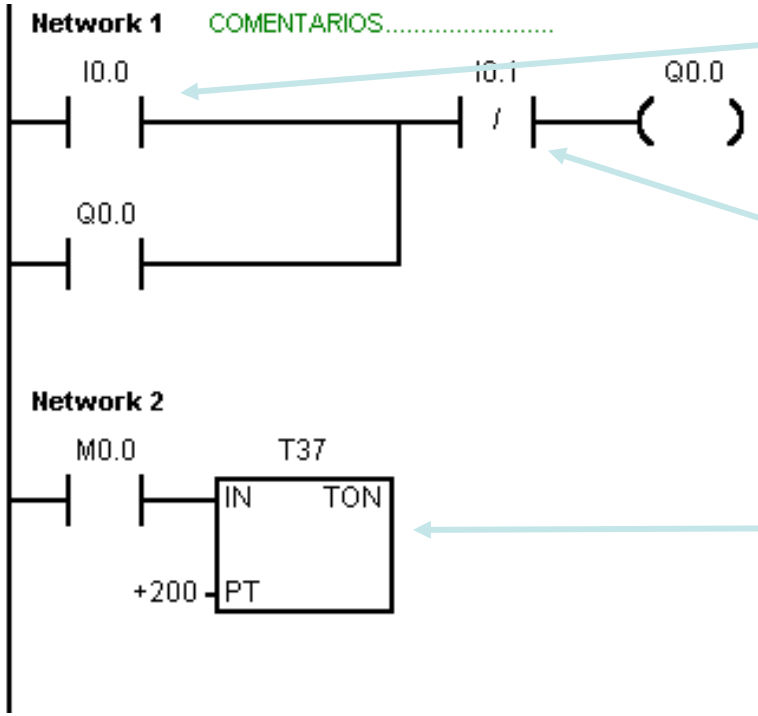
Programación en KOP

ÍNDICE

- Introducción a SETP 7: KOP
  - Lenguaje KOP: Diagrama de contactos
    - Operaciones básicas: contactos y salidas
    - Operaciones con temporizadores
    - Operaciones con contadores
    - Operaciones de comparación
    - Operaciones de transferencia
  - Distribución de la memoria
  - Ejemplos

Lenguaje KOP

- Esquema de Contactos KOP
  - la lógica se divide en unidades pequeñas y de fácil comprensión llamadas “segmentos" o “networks”
  - El programa se ejecuta segmento por segmento, de izquierda a derecha y luego de arriba a abajo.
  - Tras alcanzar la CPU el final del programa, comienza nuevamente en la primera operación del mismo

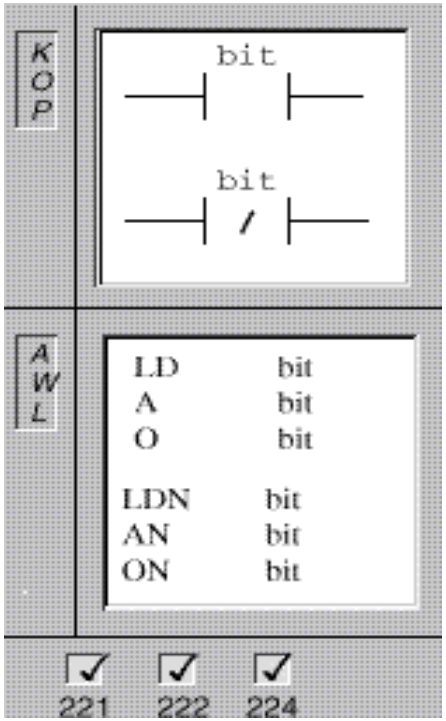


Contactos representan condiciones lógicas de "entrada" similares a interruptores, botones, condiciones internas, etc.

Bobinas: representan condiciones lógicas de "salida" similares a lámparas, arrancadores de motor, relés interpuestos, condiciones internas de salida, etc.

Cuadros representan operaciones adicionales tales como temporizadores, contadores u operaciones aritméticas.

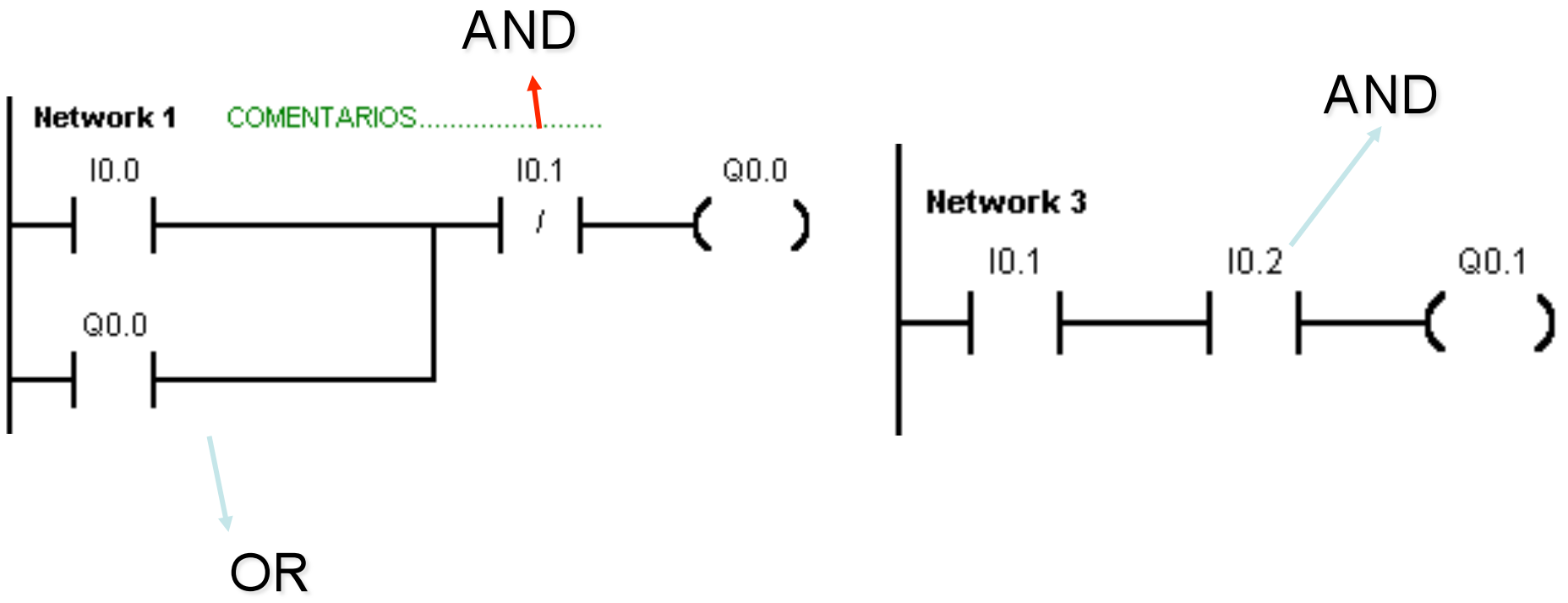
Operaciones con Contactos



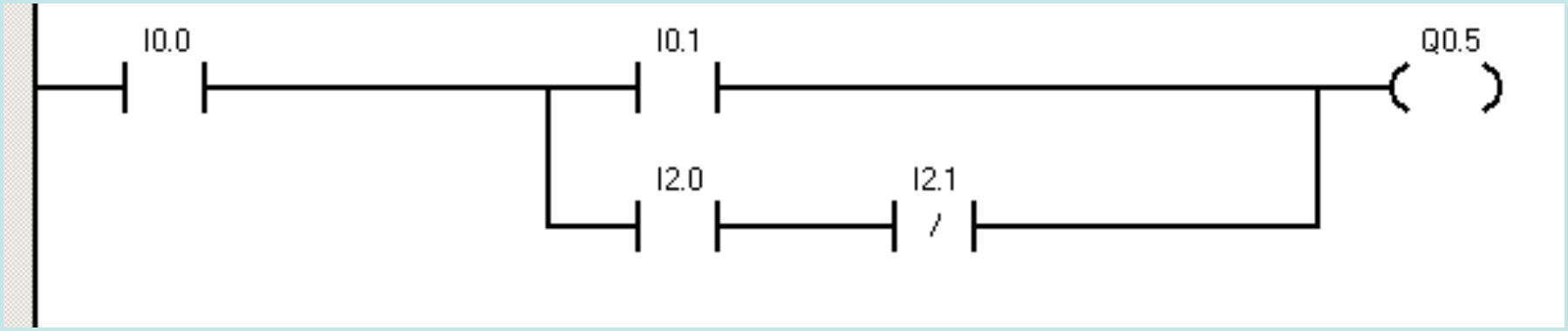
- Contactos estándar
  - El **contacto abierto (-| |-)** se cierra (se activa) si el valor binario de la dirección n = 1.
  - El **contacto cerrado (-| / |-)** se cierra (se activa) si el valor binario de la dirección n = 0
  - Tipos de operandos:
    - n: I, Q, M, SM, T, C, V

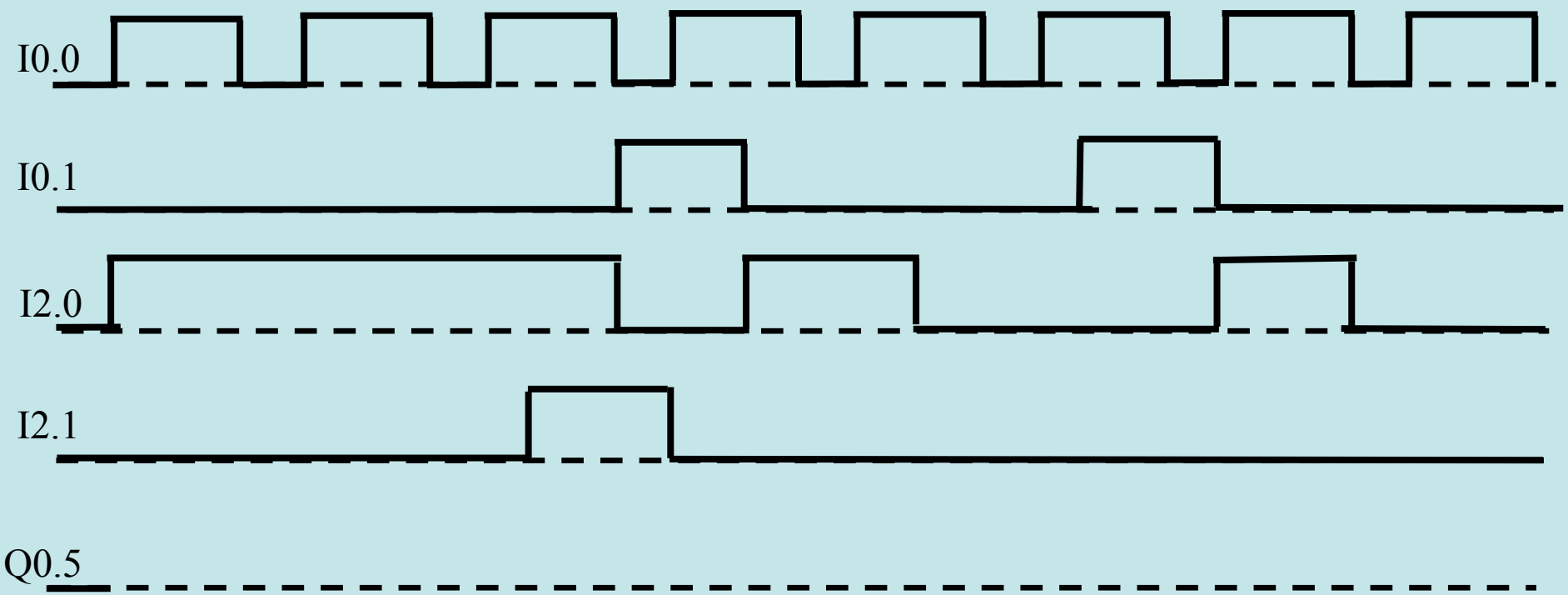
Operaciones con Contactos

- La operación AND se implementa mediante contactos en serie
- La operación OR se implementa mediante contactos en paralelo

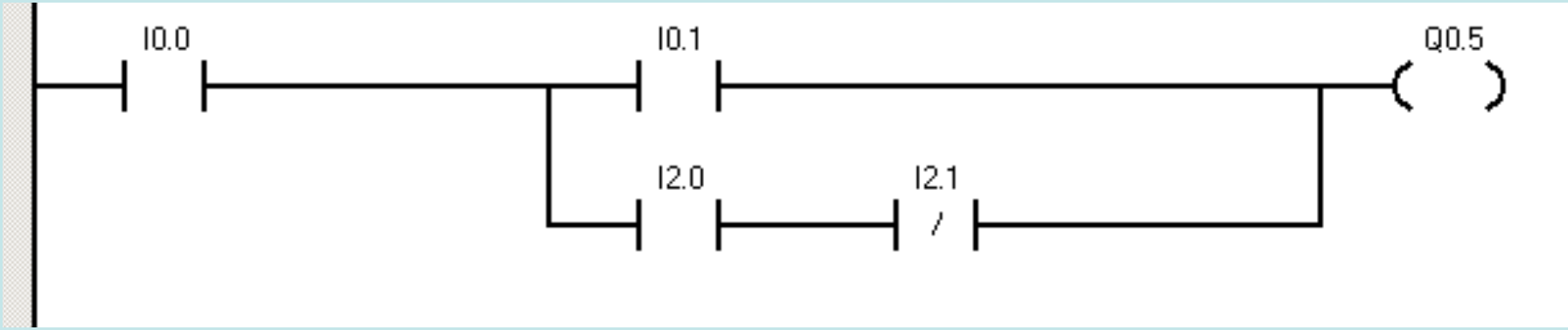


- Hacer el cronograma





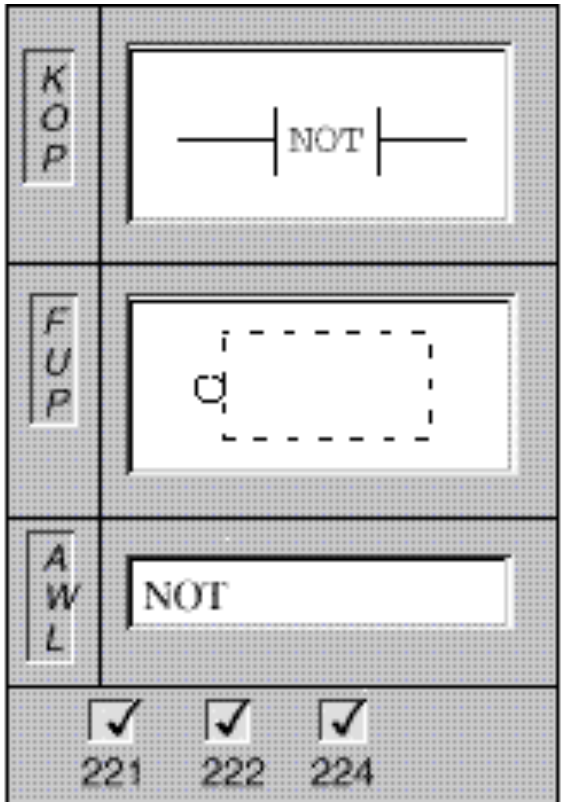
- Vamos a hacer primero primero la tabla de verdad:



I0.0	I0.1	I2.0	I2.1	Q0.5
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0

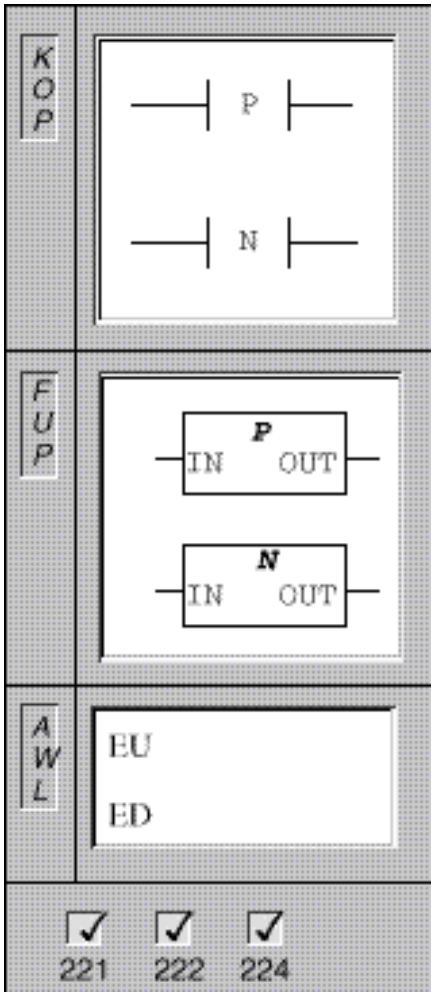
I0.0	I0.1	I2.0	I2.1	Q0.5
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

Operaciones con Contactos



- NOT
  - El contacto **NOT** invierte el valor lógico existente en ese punto del circuito.
  - En otras palabras, si al contacto **NOT** llega un “0” entonces sale un “1”, y si llega un “1” sale un “0”.
  - Operandos:
    - ninguno

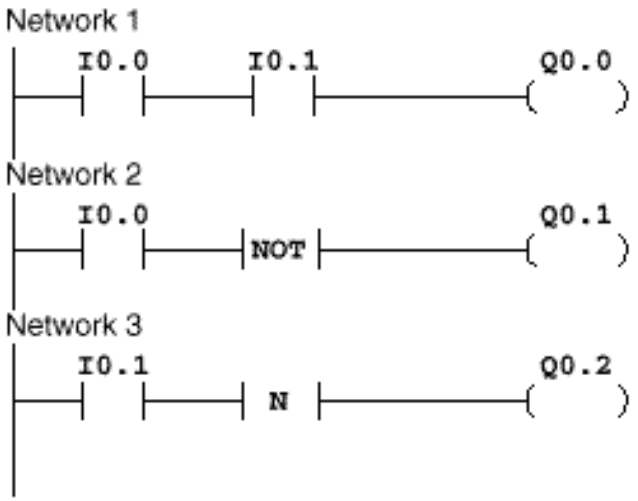
Operaciones con Contactos



- Detectar flanco positivo y negativo
  - El contacto **Detectar flanco positivo** permite que fluya la corriente durante un ciclo cada vez que se produce un cambio de 0 a 1 (de ”off” a ”on”).
  - El contacto **Detectar flanco negativo** permite que fluya la corriente durante un ciclo cada vez que se produce un cambio de 1 a 0 (de ”on” a ”off”)
  - Operandos:
    - ninguno

Operaciones con Contactos

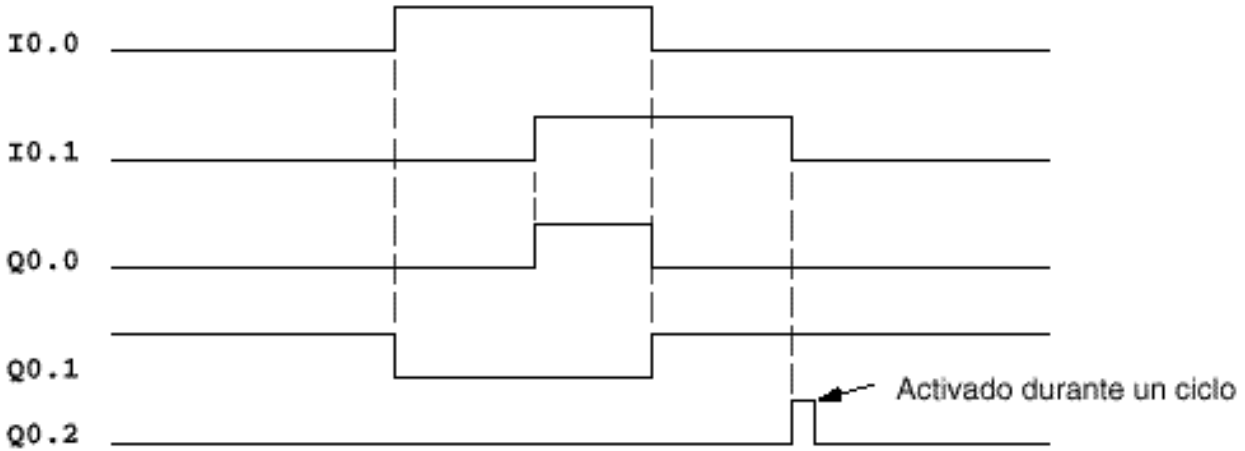
KOP



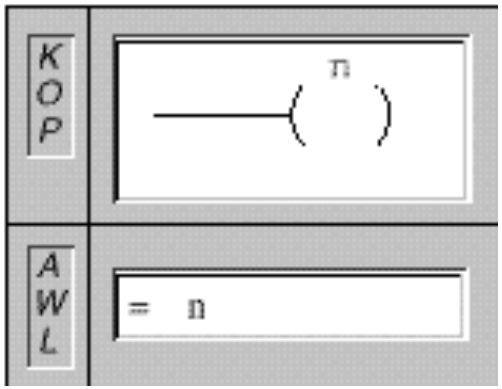
AWL

```
NETWORK
LD      I0.0
A       I0.1
=       Q0.0

NETWORK
LD      I0.0
NOT
=       Q0.1

NETWORK
LD      I0.1
ED
=       Q0.2
```

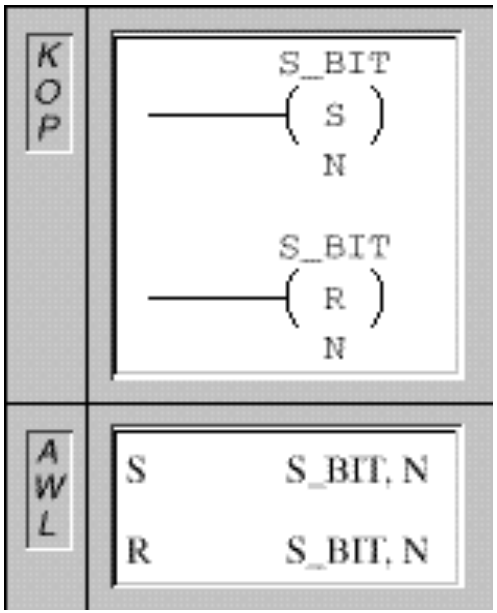
Cronograma



Operaciones con salidas



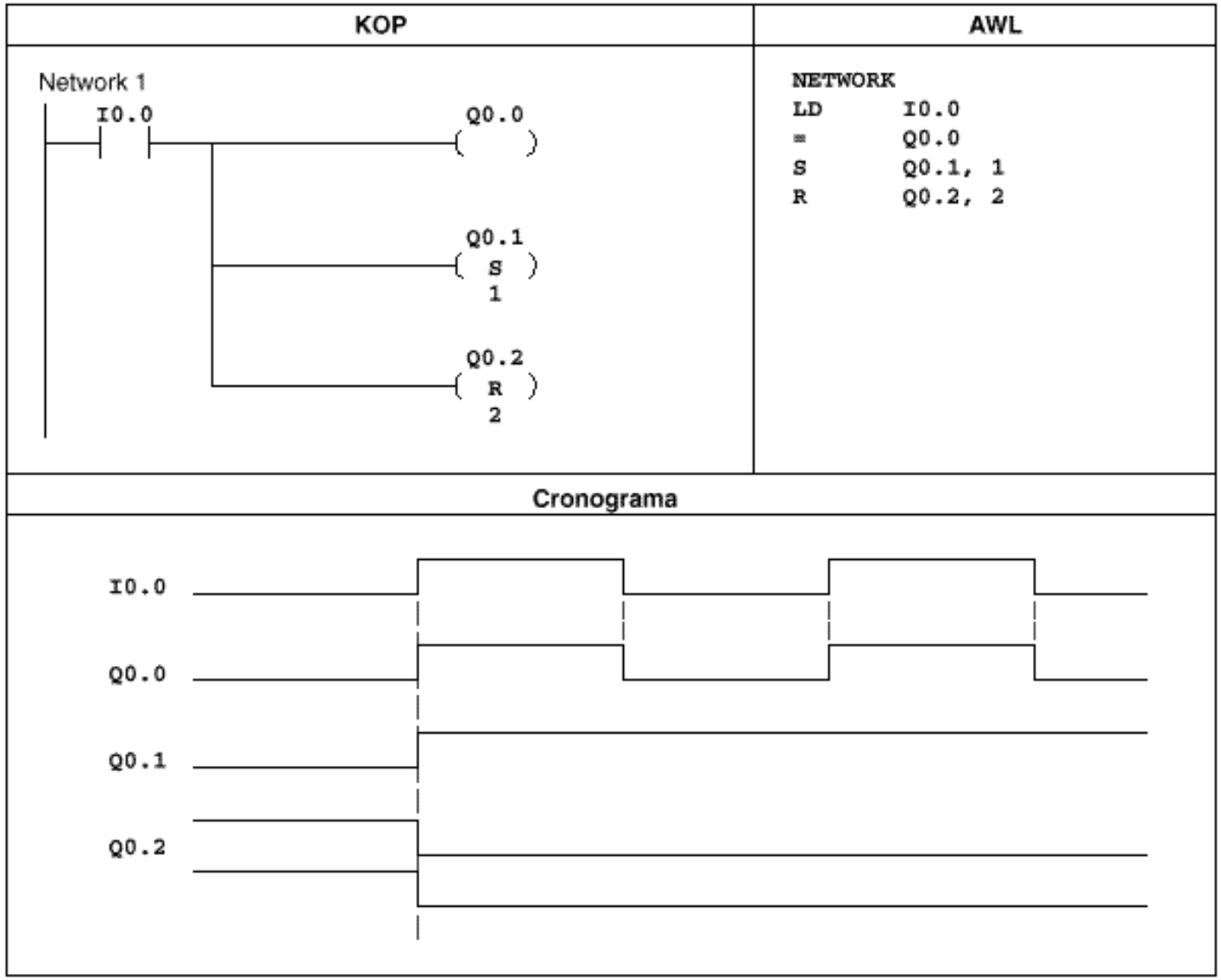
- Asignar
  - Al ejecutar la operación Asignar (bobina) se activa/desactiva el parámetro indicado (n) en función del valor lógico.
  - Operandos:
    - n: I, Q, M, SM, T, C, V



- Poner a 1 (SET), poner a cero (RESET)
  - Si el valor lógico es “1”, entonces se ejecutará la operación SET (S) o RESET (R).
  - SET pone a 1 el bit especificado, mientras que RESET lo pone a cero.
  - Además, se activan/desactivan N bits consecutivos a partir de la dirección especificada.
  - Operandos:
    - S_BIT: I, Q, M, SM, T, C, V
    - N: IB, QB, MB, SMB, VB, AC, constante (1-255)
- Resultan especialmente útiles para almacenar una condición o estado (p.e. si se acciona un pulsador).

Operaciones con Salidas

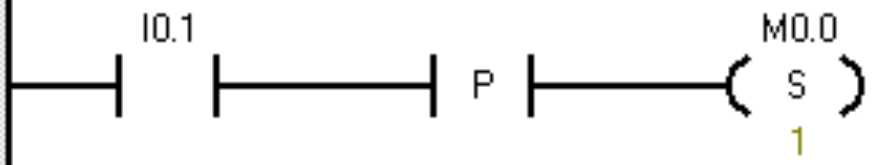
- Ejemplo:



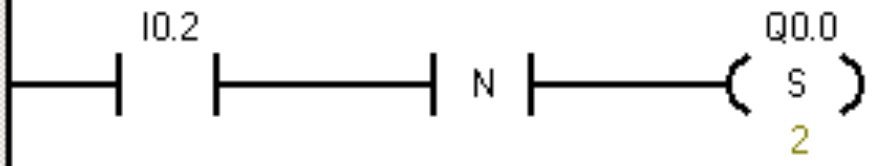
COMENTARIOS DEL PROGRAMA

Network 1 Título de segmento

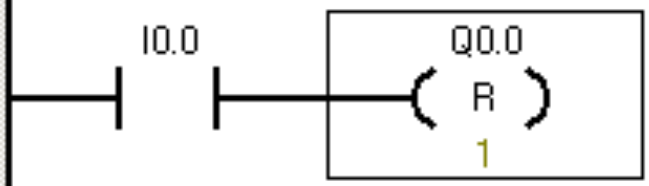
Comentario de segmento



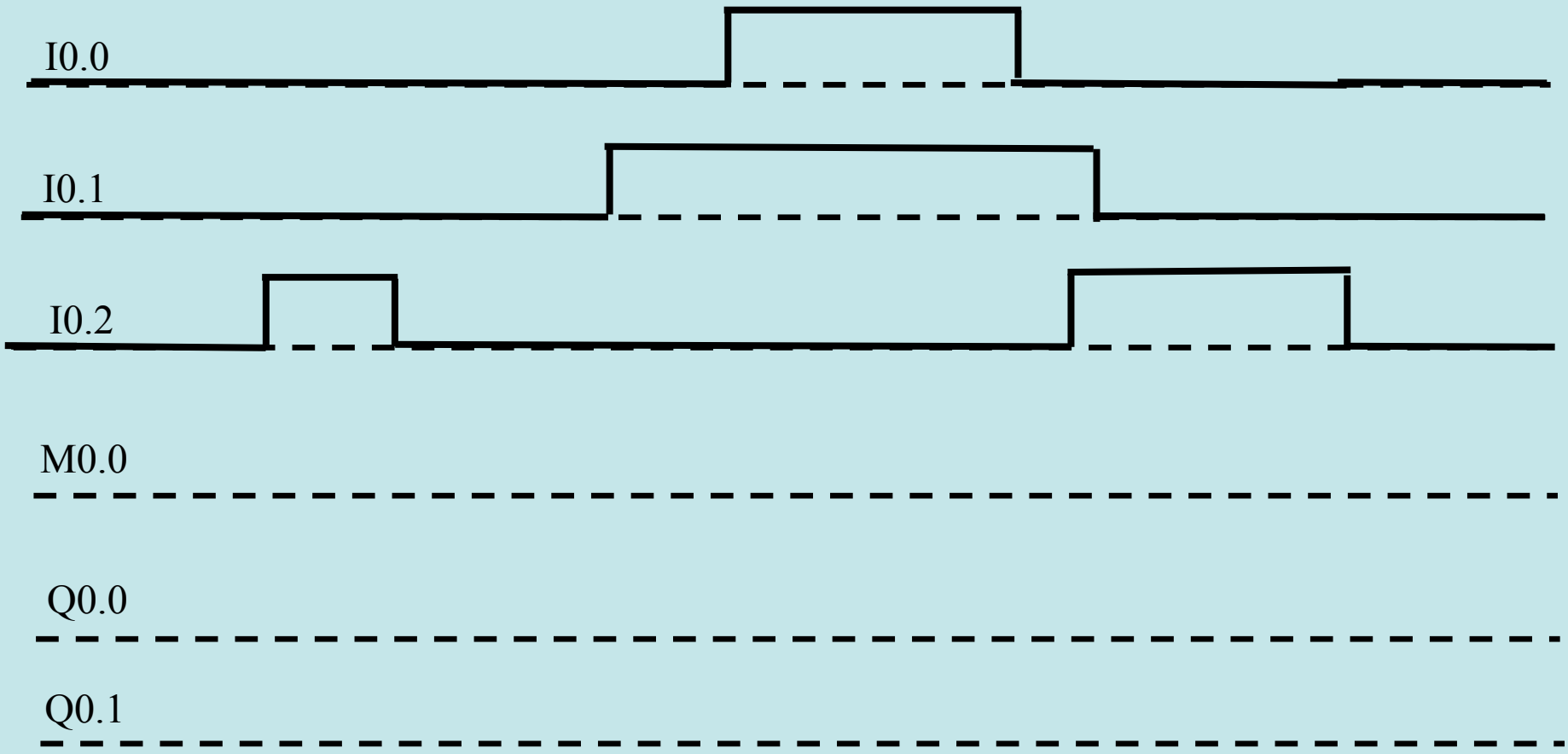
Network 2



Network 3

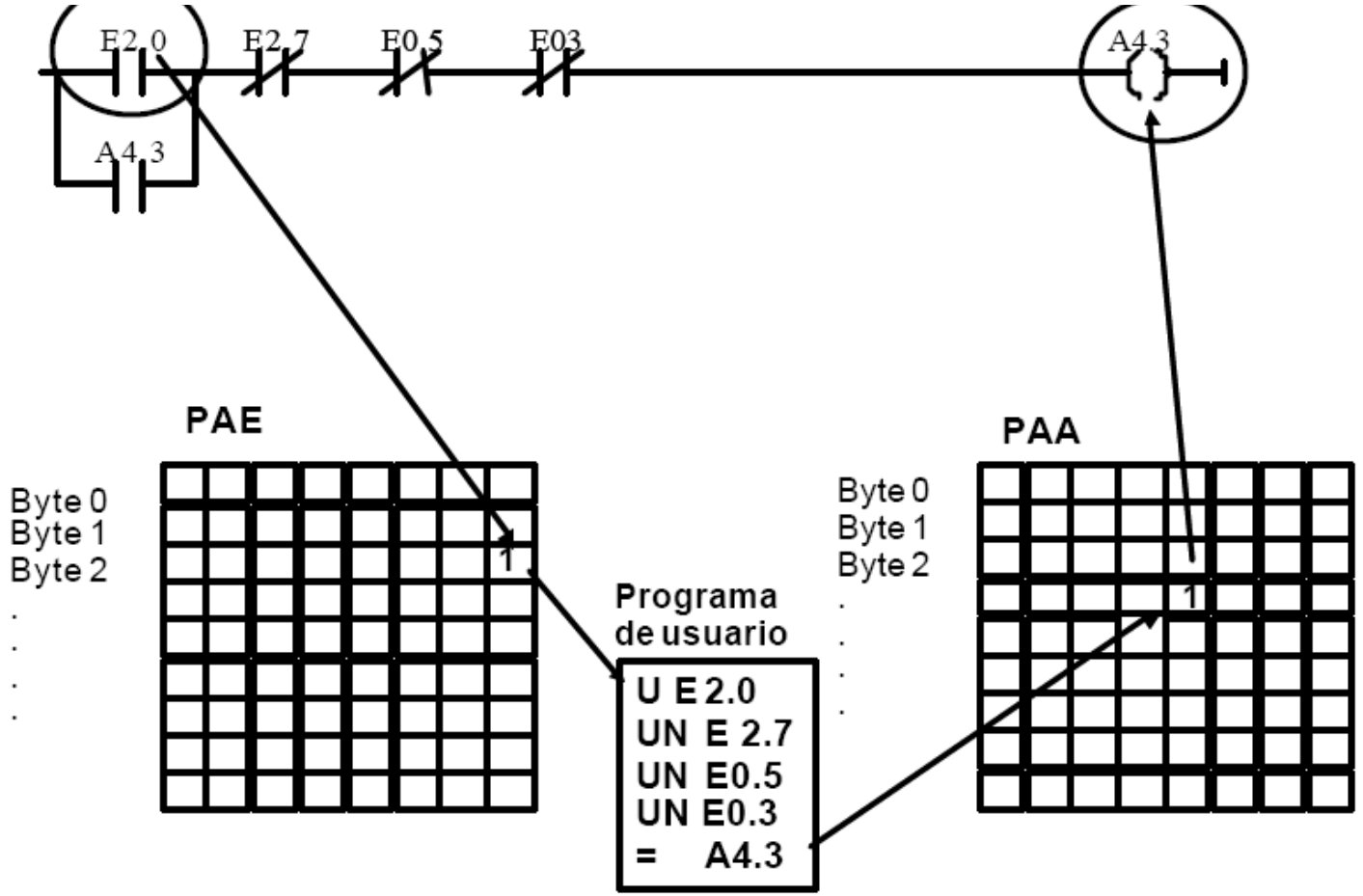


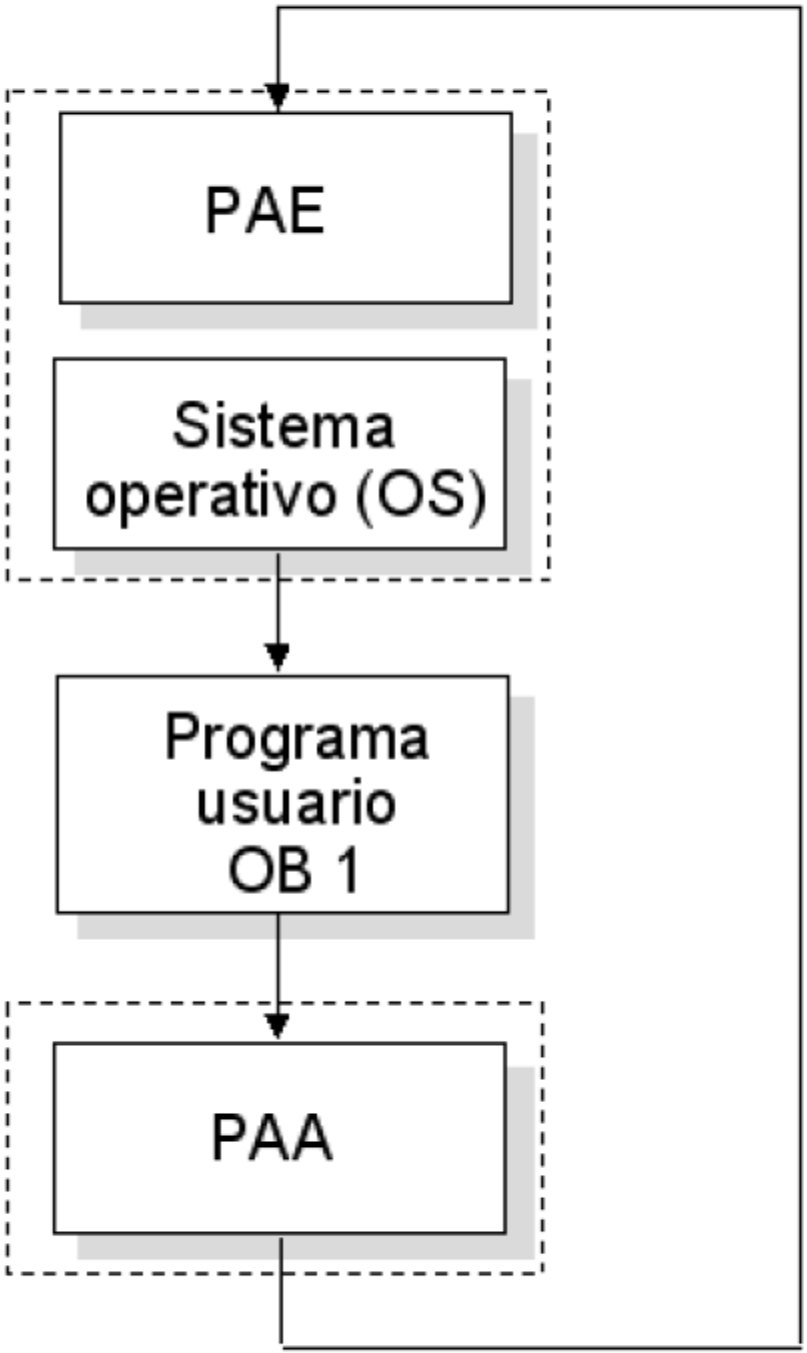
Operaciones con Salidas



Operaciones con Salidas

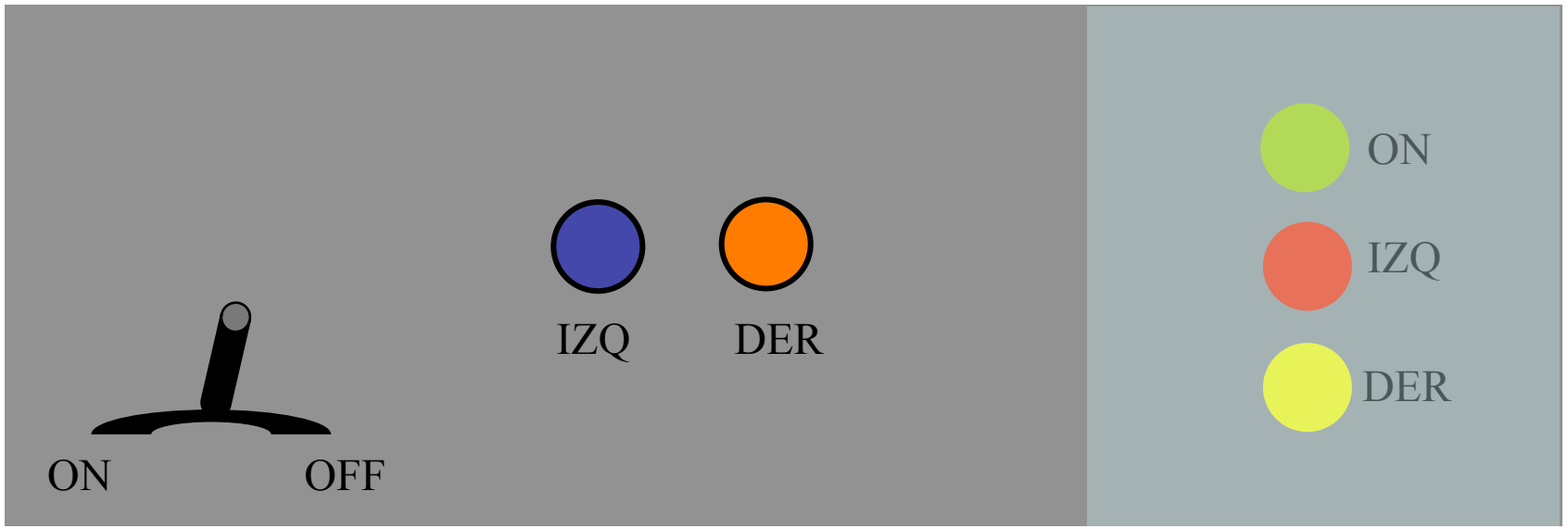
- Al ejecutarse el programa de usuario, se lee el valor de las entradas de la memoria imagen de entradas y, a continuación, se escriben los resultados en la memoria imagen de salidas.




PAE
Sistema operativo (OS)
Programa usuario OB 1
PAA

Ejemplo

- Ejemplo:
  - Panel de mando de un motor.



Ejemplo

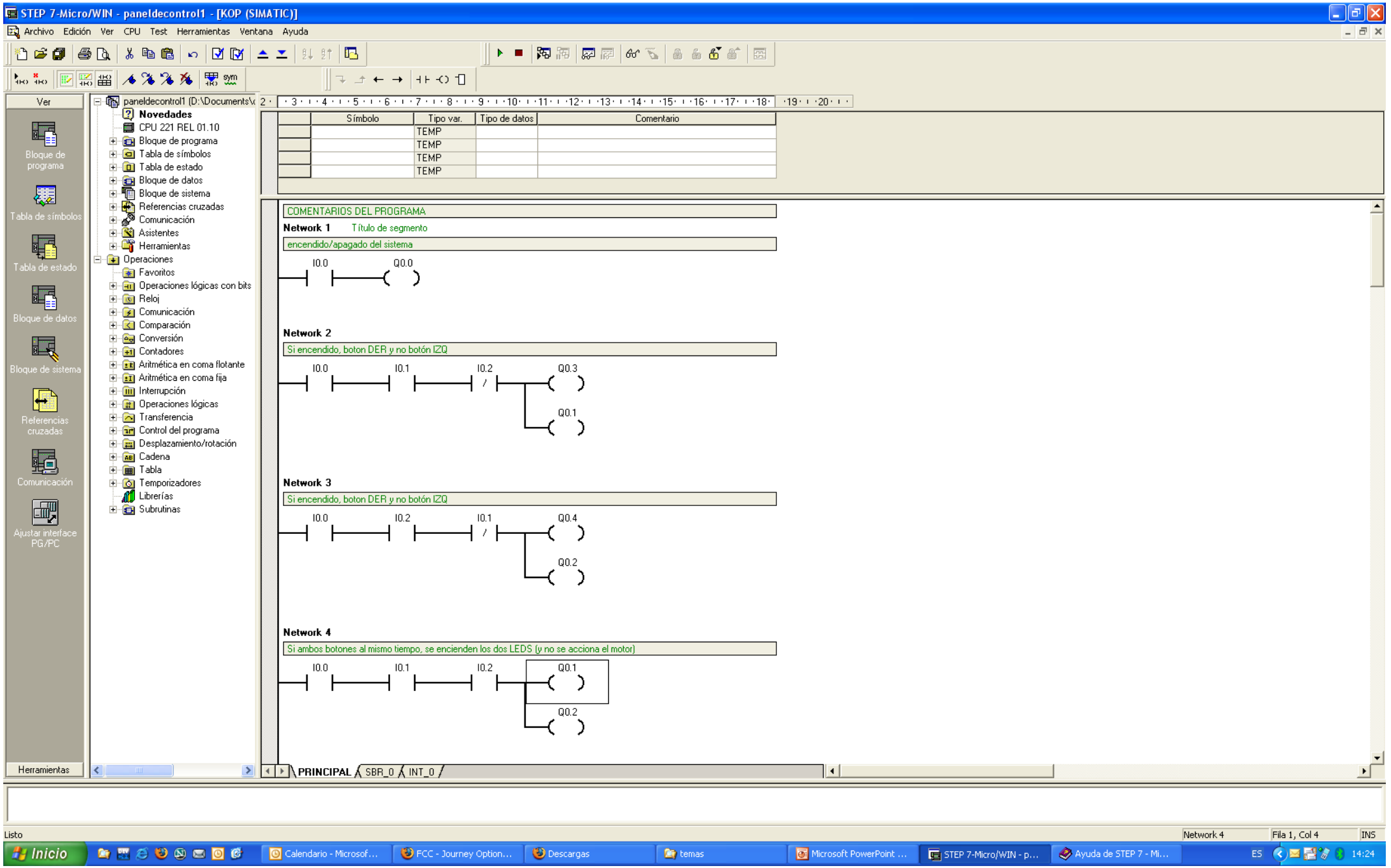
- El panel de mando cuenta con los siguientes elementos:
  - Interruptor on/off general.
  - Botón giro positivo motor.
  - Botón giro negativo motor.
  - Lámpara sistema encendido.
  - Lámpara sentido giro positivo.
  - Lámpara sentido negativo giro.
- El motor se acciona mediante dos contactores:
  - Contactor giro positivo
  - Contactor giro negativo.

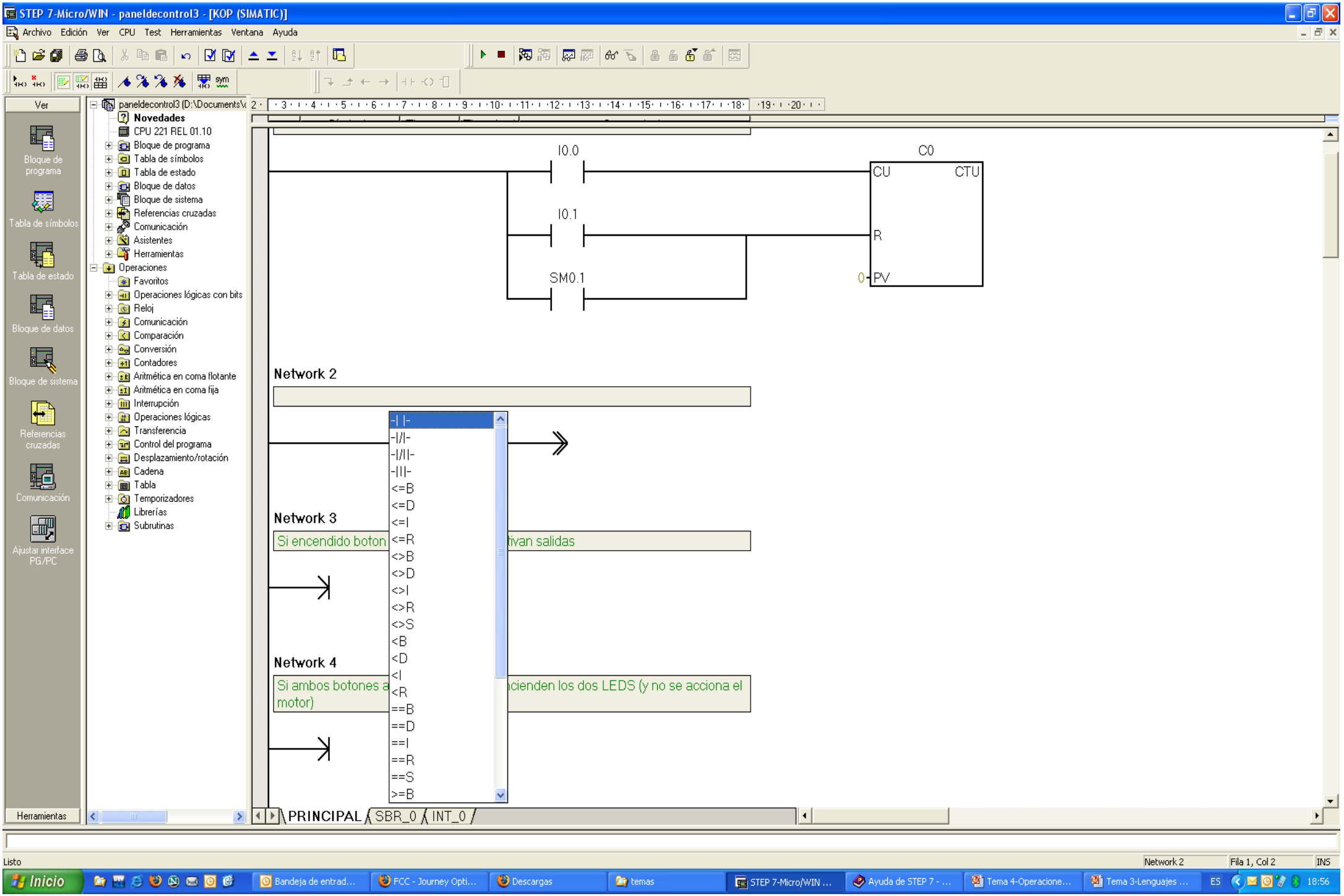
Operaciones con Salidas

- Funcionamiento:
  - El interruptor on/off arranca/detiene el sistema. Cuando el sistema está encendido la lámpara luce.
  - Cuando el sistema está encendido, si se presiona el botón DER el motor gira en ese sentido y se enciende la lámpara correspondiente.
  - Cuando el sistema está encendido, el botón IZQ hace girar al motor en ese sentido y enciende la lámpara correspondiente.
  - Apretar los dos botones simultáneamente detiene el motor y se encienden ambas lámparas.

Operaciones con Salidas

- Asignamos entradas y salidas del autómata:
- El panel de mando cuenta con los siguientes elementos:
  - Interruptor on/off general (I0.0, on=24V, off=0V).
  - Botón giro positivo motor (I0.1, giro=24V, off=0V).
  - Botón giro negativo motor (I0.2, giro=24V, off=0V).
  - Lámpara sistema encendido (Q0.0, encendida=24V, off=0V).
  - Lámpara sentido giro positivo (Q0.1, encendida=24V, off=0V).
  - Lámpara sentido negativo giro (Q0.2, encendida=24V, off=0V).
- El motor se acciona mediante dos contactores:
  - Contactor giro positivo (Q0.3, giro=24V, off=0V).
  - Contactor giro negativo (Q0.4, giro=24V, off=0V).
- Con esto vamos a desarrollar el programa en KOP.
- GRAFCET… más adelante.


STEP 7-Micro/WIN - paneldecontrol1 - [KOP (SIMATIC)]
Archivo Edición Ver CPU Test Herramientas Ventana Ayuda
Ver
Bloque de programa
Tabla de símbolos
Tabla de estado
Bloque de datos
Bloque de sistema
Referencias cruzadas
Comunicación
Ajustar interface PG/PC
Herramientas
paneldecontrol1 (D:\Documents\
Novedades
CPU 221 REL 01.10
Bloque de programa
Tabla de símbolos
Tabla de estado
Bloque de datos
Bloque de sistema
Referencias cruzadas
Comunicación
Asistentes
Herramientas
Operaciones
Favoritos
Operaciones lógicas con bits
Reloj
Comunicación
Comparación
Conversión
Contadores
Aritmética en coma flotante
Aritmética en coma fija
Interrupción
Operaciones lógicas
Transferencia
Control del programa
Desplazamiento/rotación
Cadena
Tabla
Temporizadores
Librerías
Subrutinas
Símbolo
Tipo var.
Tipo de datos
Comentario
TEMP
TEMP
TEMP
TEMP
COMENTARIOS DEL PROGRAMA
Network 1
Título de segmento
encendido/apagado del sistema
I0.0
Q0.0
Network 2
Si encendido, boton DER y no botón IZQ
I0.0
I0.1
I0.2
Q0.3
Q0.1
Network 3
Si encendido, boton DER y no botón IZQ
I0.0
I0.2
I0.1
Q0.4
Q0.2
Network 4
Si ambos botones al mismo tiempo, se encienden los dos LEDS (y no se acciona el motor)
I0.0
I0.1
I0.2
Q0.1
Q0.2
PRINCIPAL
SBR_0
INT_0
Listo
Network 4
Fila 1, Col 4
INS
Inicio
Calendario - Microsof...
FCC - Journey Option...
Descargas
temas
Microsoft PowerPoint ...
STEP 7-Micro/WIN - p...
Ayuda de STEP 7 - Mi...
ES
14:24


STEP 7-Micro/WIN - paneldecontrol3 - [KOP (SIMATIC)]
Archivo Edición Ver CPU Test Herramientas Ventana Ayuda
Ver
Bloque de programa
Tabla de símbolos
Tabla de estado
Bloque de datos
Bloque de sistema
Referencias cruzadas
Comunicación
Ajustar interface PG/PC
Herramientas
paneldecontrol3 (D:\Documents\
Novedades
CPU 221 REL 01.10
Bloque de programa
Tabla de símbolos
Tabla de estado
Bloque de datos
Bloque de sistema
Referencias cruzadas
Comunicación
Asistentes
Herramientas
Operaciones
Favoritos
Operaciones lógicas con bits
Reloj
Comunicación
Comparación
Conversión
Contadores
Aritmética en coma flotante
Aritmética en coma fija
Interrupción
Operaciones lógicas
Transferencia
Control del programa
Desplazamiento/rotación
Cadena
Tabla
Temporizadores
Librerías
Subrutinas
I0.0
I0.1
SM0.1
C0
CU
CTU
R
0
PV
Network 2
Network 3
Si encendido boton
tivan salidas
Network 4
Si ambos botones a
cienden los dos LEDS (y no se acciona el motor)
-| |-
-|/|-
-|/||-
-|||-
<=B
<=D
<=I
<=R
<>B
<>D
<>I
<>R
<>S
<B
<D
<I
<R
==B
==D
==I
==R
==S
>=B
PRINCIPAL
SBR_0
INT_0
Listo
Network 2
Fila 1, Col 2
INS
Inicio
Bandeja de entrad...
FCC - Journey Opti...
Descargas
temas
STEP 7-Micro/WIN ...
Ayuda de STEP 7 - ...
Tema 4-Operacione...
Tema 3-Lenguajes ...
ES
18:56

PANEL DE CONTROL DEL MOTOR

Network 1

encendido/apagado del sistema

Network 2

Si encendido, boton DER y no botón IZQ

Network 3

Si encendido, boton DER y no botón IZQ

Ejemplo

- Falta algo… al presionar ambos botones al tiempo: no se mueve y se encienden las dos luces.
- Añadid esta condición al programa

PANEL DE CONTROL DEL MOTOR

Network 1

encendido/apagado del sistema

Network 2

Si encendido, boton DER y no botón IZQ

Network 3

Si encendido, boton DER y no botón IZQ

Network 4

Si ambos botones al mismo tiempo, se encienden los dos LEDS (y no se acciona el motor)

Ejemplo

- PREGUNTA: Al cargar el programa, observamos que no hace lo que pensábamos… No se encienden las luces de giro IZQ y DER… ¿Por qué?

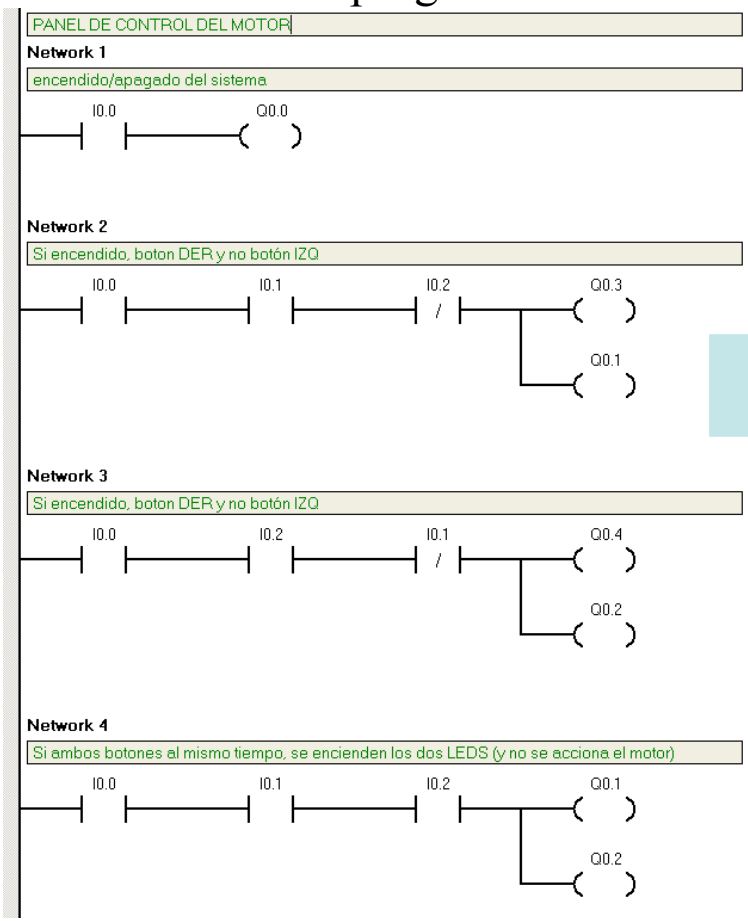
Ejemplo

- Hemos dicho… el programa en KOP se ejecuta de izquierda a derecha y de arriba abajo… ¿qué quiere decir eso?
- Las entradas se vuelcan en la memoria imagen de entradas antes de la ejecución.
- Las salidas se vuelcan en la memoria imagen de salidas después de la ejecución.
- Vamos a ver esto sobre el ejemplo

1.- **Memoria imagen de entradas**

I0.0	**I0.1**	**I0.2**	**…**
I1.0	**I1.1**	…	

2.- Ejecución del programa



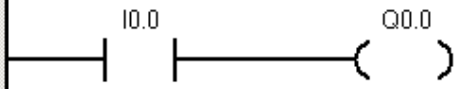
3.- Volcado de los **últimos** valores de las variables en la **memoria imagen de salidas**

Q0.0	**Q0.1**	**Q0.2**	**…**
Q1.0	**Q1.1**	…	

PANEL DE CONTROL DEL MOTOR

Network 1

encendido/apagado del sistema



Inicialmente, Sup:

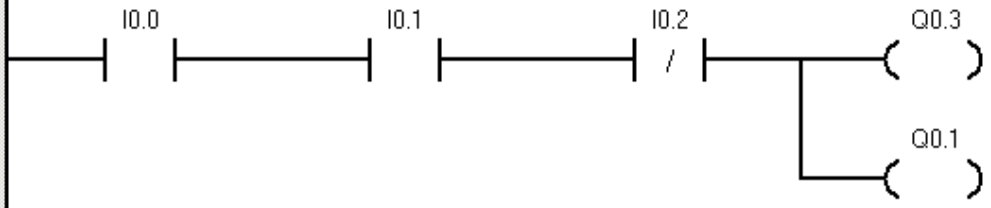
Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

I0.0	I0.1	I0.2
1	0	0

Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

Network 2

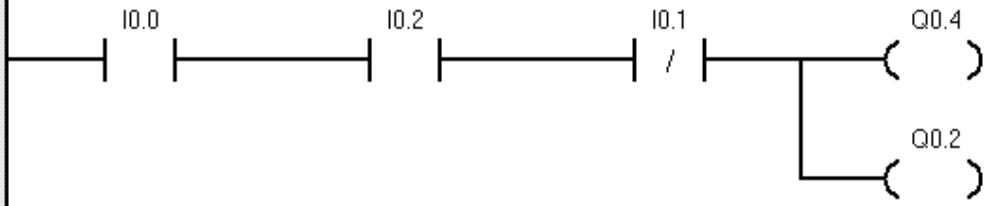
Si encendido, boton DER y no botón IZQ



Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

Network 3

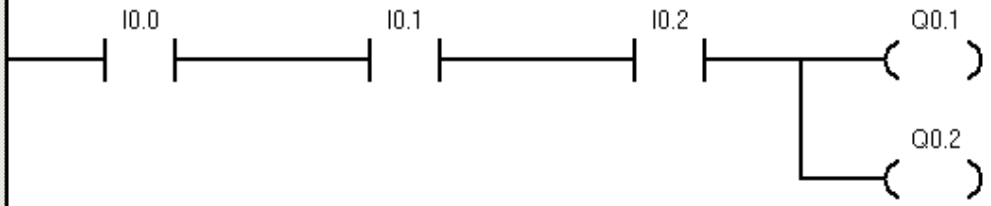
Si encendido, boton DER y no botón IZQ



Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

Network 4

Si ambos botones al mismo tiempo, se encienden los dos LEDS (y no se acciona el motor)

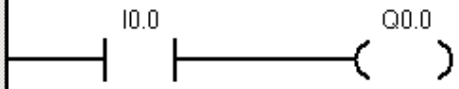


Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

PANEL DE CONTROL DEL MOTOR

Network 1

encendido/apagado del sistema



Inicialmente, Sup:

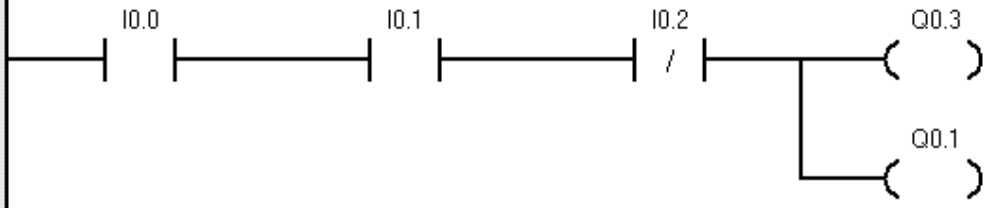
Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

I0.0	I0.1	I0.2
1	1	0

Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

Network 2

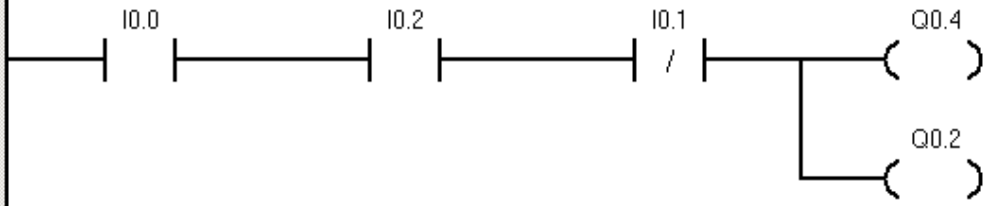
Si encendido, boton DER y no botón IZQ



Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	1	0	1	0

Network 3

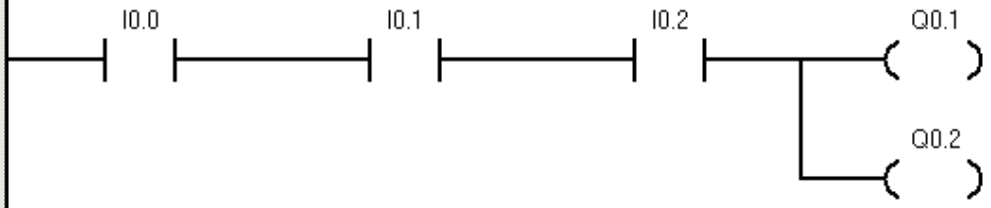
Si encendido, boton DER y no botón IZQ



Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	1	0	1	0

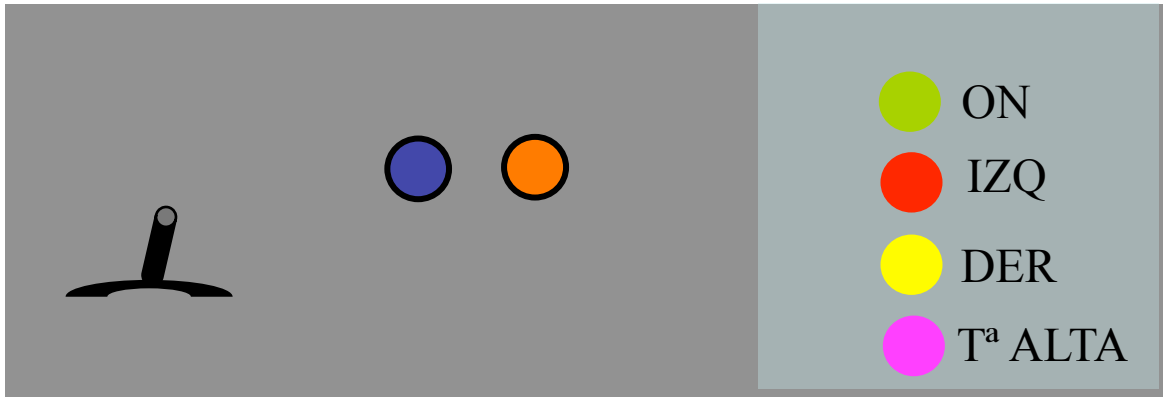
Network 4

Si ambos botones al mismo tiempo, se encienden los dos LEDS (y no se acciona el motor)

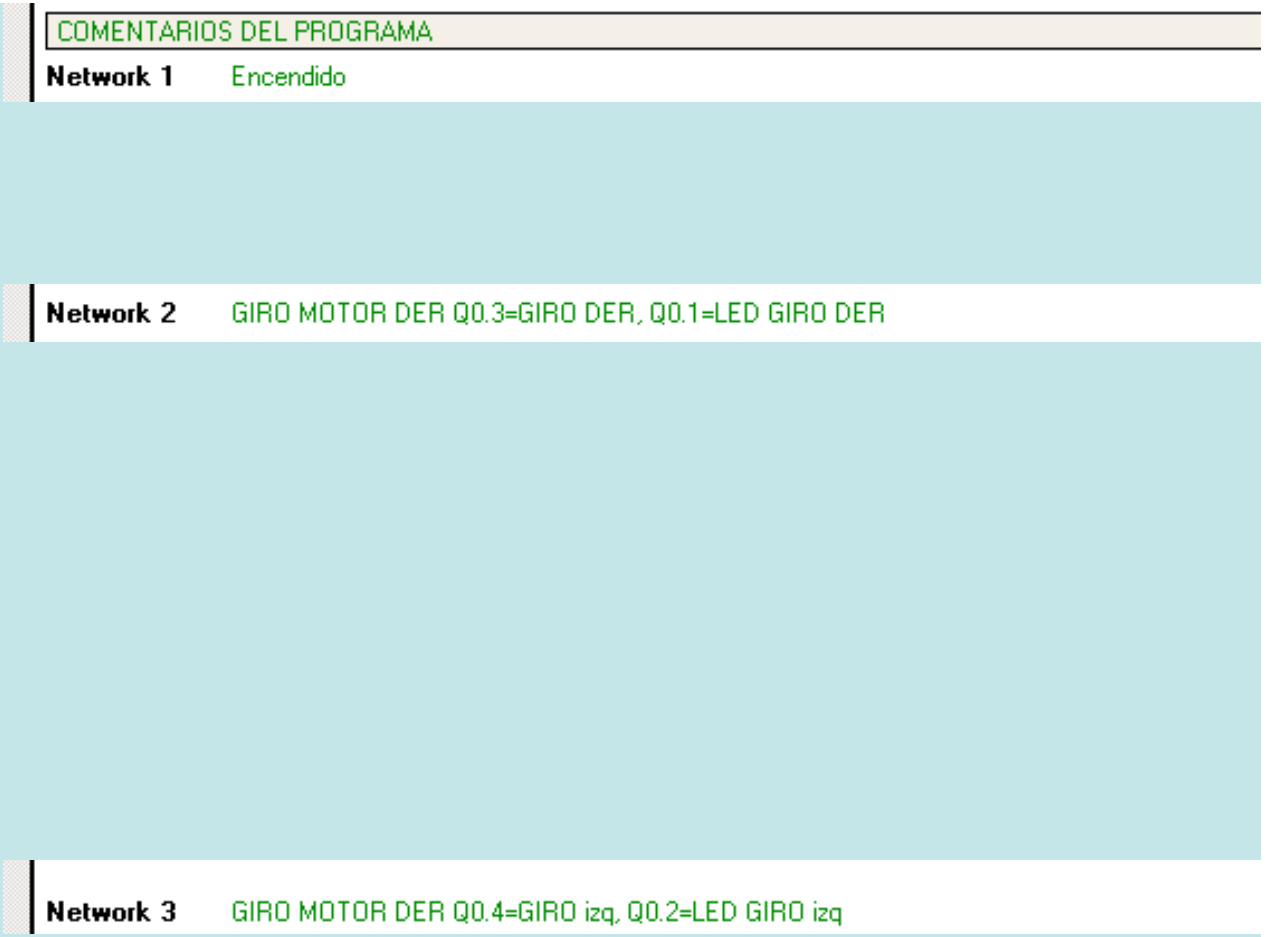


Q0.0	Q0.1	Q0.2	Q0.3	Q0.4
1	0	0	0	0

- Ahora cambian ligeramente las especificaciones.
- Mantener el botón apretado es algo pesado. Con apretar los botones y soltar debería bastar.
- Además, se añade un sensor de temperatura del motor y una lámpara correspondiente. I0.3 (temperatura ok=24V, temperatura alta=0V). Lámpara, salida Q0.5.
- Funcionamiento:
  - El interruptor on/off arranca/para el sistema. Cuando el sistema está encendido la lámpara se enciende. Al desactivar ON, se debe parar el motor
  - Cuando el sistema está encendido, si se presiona el botón DER el motor gira en ese sentido y se enciende la lámpara correspondiente.
  - Cuando el sistema está encendido, el botón IZQ hace girar al motor en ese sentido y enciende la lámpara correspondiente.
  - Apretar los dos botones simultáneamente detiene el motor y se APAGAN ambas lámparas.
  - Si, en cualquier momento, la temperatura es excesiva, el motor se debe parar. En este caso, se debe iluminar la lámpara de exceso de temperatura



- Hay varias opciones para resolver el problema…
- En este caso se utilizan las operaciones de SET y RESET.
- Una solución es, por ejemplo:
- …



Network 4 APRETAR AMBOS BOTONES DETIENE EL MOTOR Y ENCIENDE AMBOS LEDS

Network 5 TEMPERATURA

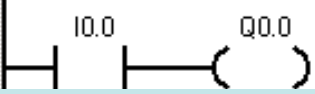
Network 6 AL DESACTIVAR EL SISTEMA--> AHORA HAY QUE DETENER TODO EXPLÍCITAMENTE

- Solución:
  - Nótese que si Q0.4 y Q0.5 no pueden estar activas simultáneamente.
  - Además, si I0.3=0, se resetean todas las salidas y se enciende el LED correspondiente. Y ESTÁ AL FINAL DEL PROGRAMA… DA IGUAL LO QUE HAYAMOS HECHO ANTES…
  - Para entenderlo hay que pensar que el programa se ejecuta de izquierda a derecha y de arriba abajo.
  - El programa se evalúa completamente y después se trasladan las salidas a la memoria imagen de Salidas.
  - Y si queremos que parpadee la luz cuando Tª excesiva…

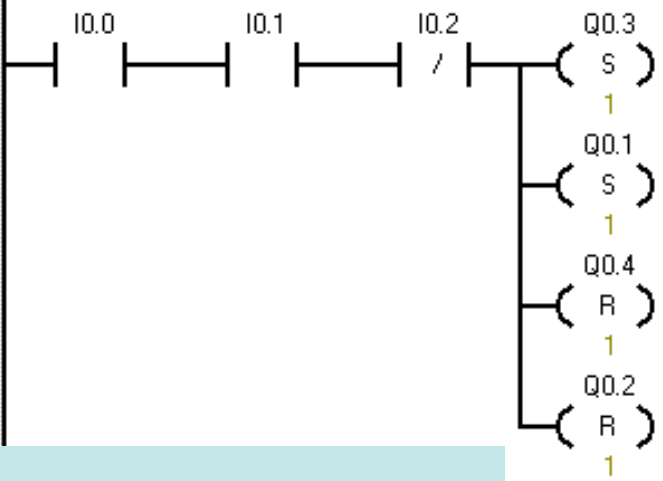
COMENTARIOS DEL PROGRAMA

I0.0	I0.1	I0.2	I0.3
1	1	0	0

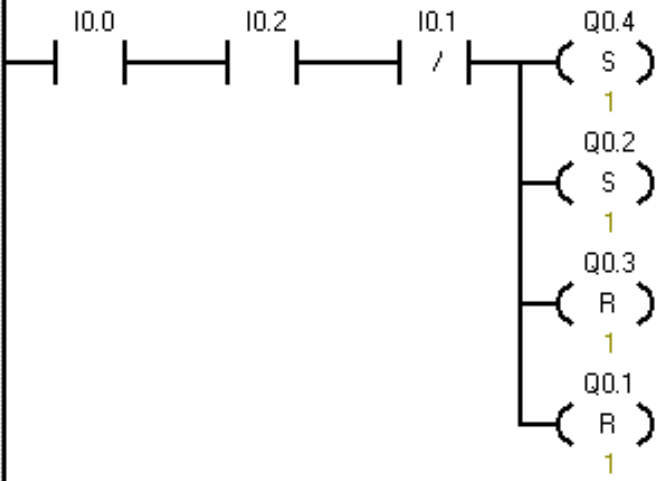
Network 1 Encendido



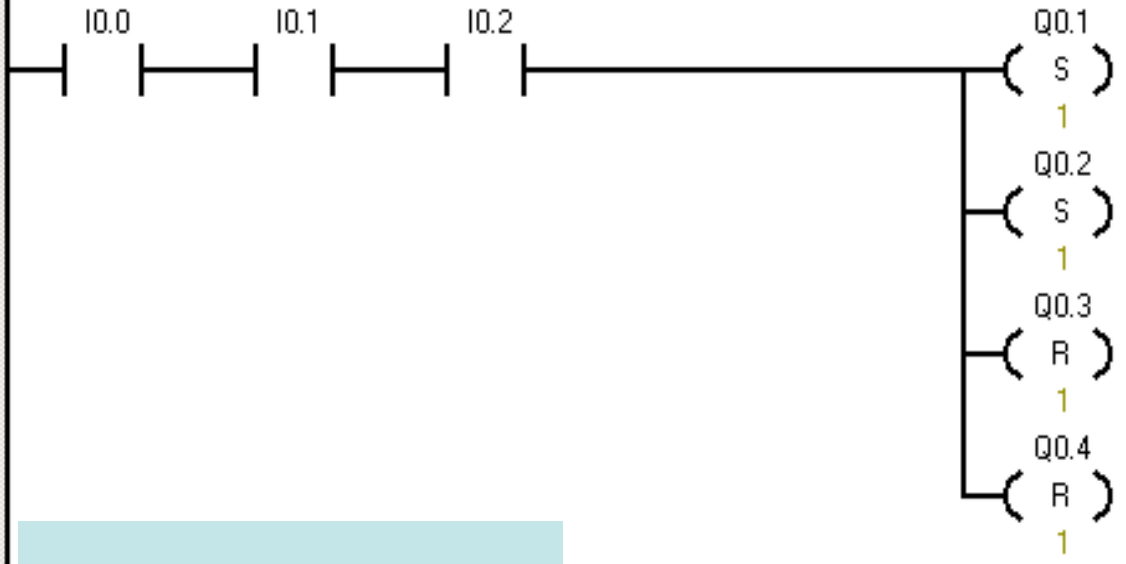
Network 2 GIRO MOTOR DER Q0.3=GIRO DER, Q0.1=LED GIRO DER



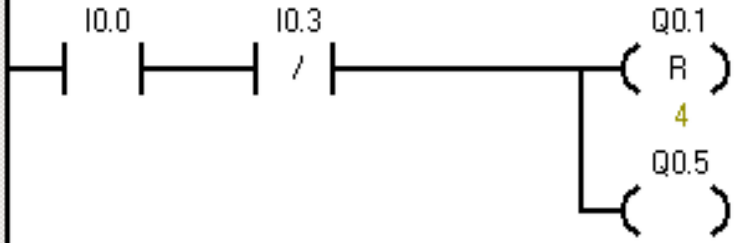
Network 3 GIRO MOTOR DER Q0.4=GIRO izq, Q0.2=LED GIRO izq



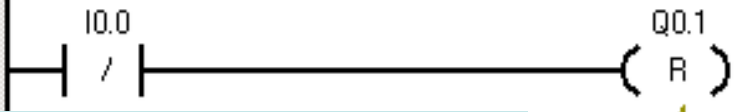
Network 4 APRETAR AMBOS BOTONES DETIENE EL MOTOR Y ENCIENDE AMBOS LEDS



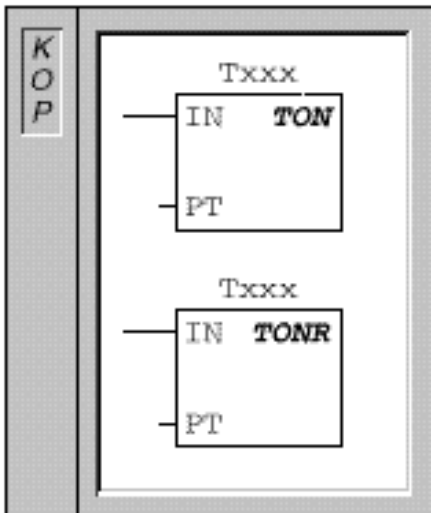
Network 5 TEMPERATURA



Network 6 AL DESACTIVAR EL SISTEMA--> AHORA HAY QUE DETENER TODO EXPLÍCITAMENTE

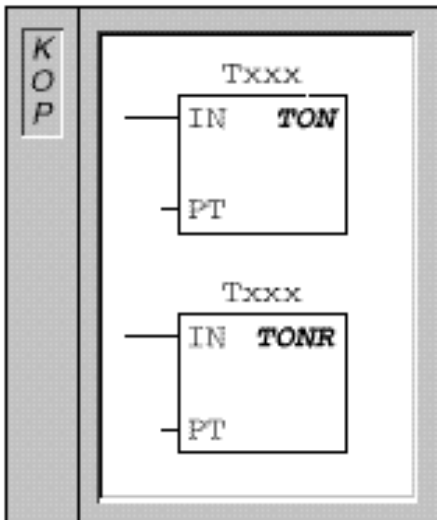


Operaciones con Temporizadores



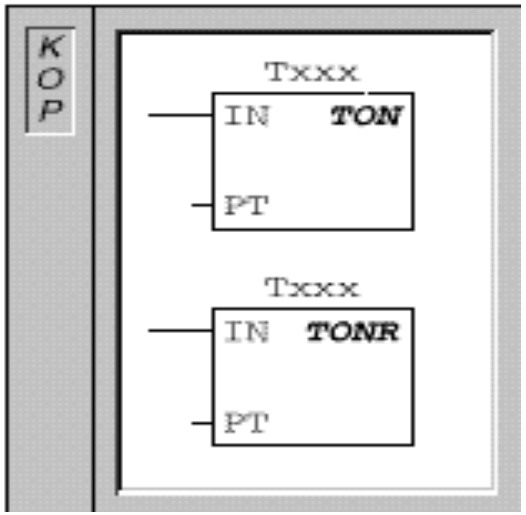
- Especialmente pensados para medir periodos de tiempo.
- En los autómatas se distinguen diferentes tipos de temporizadores, en función de la resolución (mínimo intervalo de tiempo que pueden medir).
  - P.e. temporizadores de 1ms, 10ms y 100ms.
- Además, existen dos tipos:
  - Temporizador de retardo a la conexión (TON).
  - Temporizador de retardo a la conexión memorizado (TONR).

Operaciones con Temporizadores



- Tenemos que pensar en un temporizador como en una variable con varios campos (p.e. una estructura en C). Distinguiremos dos partes en el temporizador:
  - Un valor lógico del temporizador (1 ó 0)→ bit de temporización.
  - Un valor entero (2 bytes) → número de ciclos que han transcurrido
- Distinguimos una entrada de activación (IN).
- Distinguimos una entrada de preselección (PT).
- Distinguimos el número de temporizador (Txxx, predefinido)
- TON:
  - Cuenta el nº de ciclos de tiempo cuando la entrada IN está activa. El bit de temporización se pone a 1 al alcanzar el valor de preselección (PT)
  - Cuando IN=0, el nº de ciclos se pone a 0 y también el bit de temporización.
- TONR:
  - Igual que el anterior, pero cuando IN=0, se mantiene el nº de ciclos, así como el bit de temporización.
- Ambos tipos (TON, TONR) se detienen al alcanzar el valor máximo de conteo.

Operaciones con Temporizadores



– Operandos:

Txxx:

	TON	TONR
1 ms	T32, T96	T0, T64
10 ms	T33 a T36	T1 a T4
	T97 a T100	T65 a T68
100 ms	T37 a T63	T5 a T31
	T101 a T255	T69 a T95

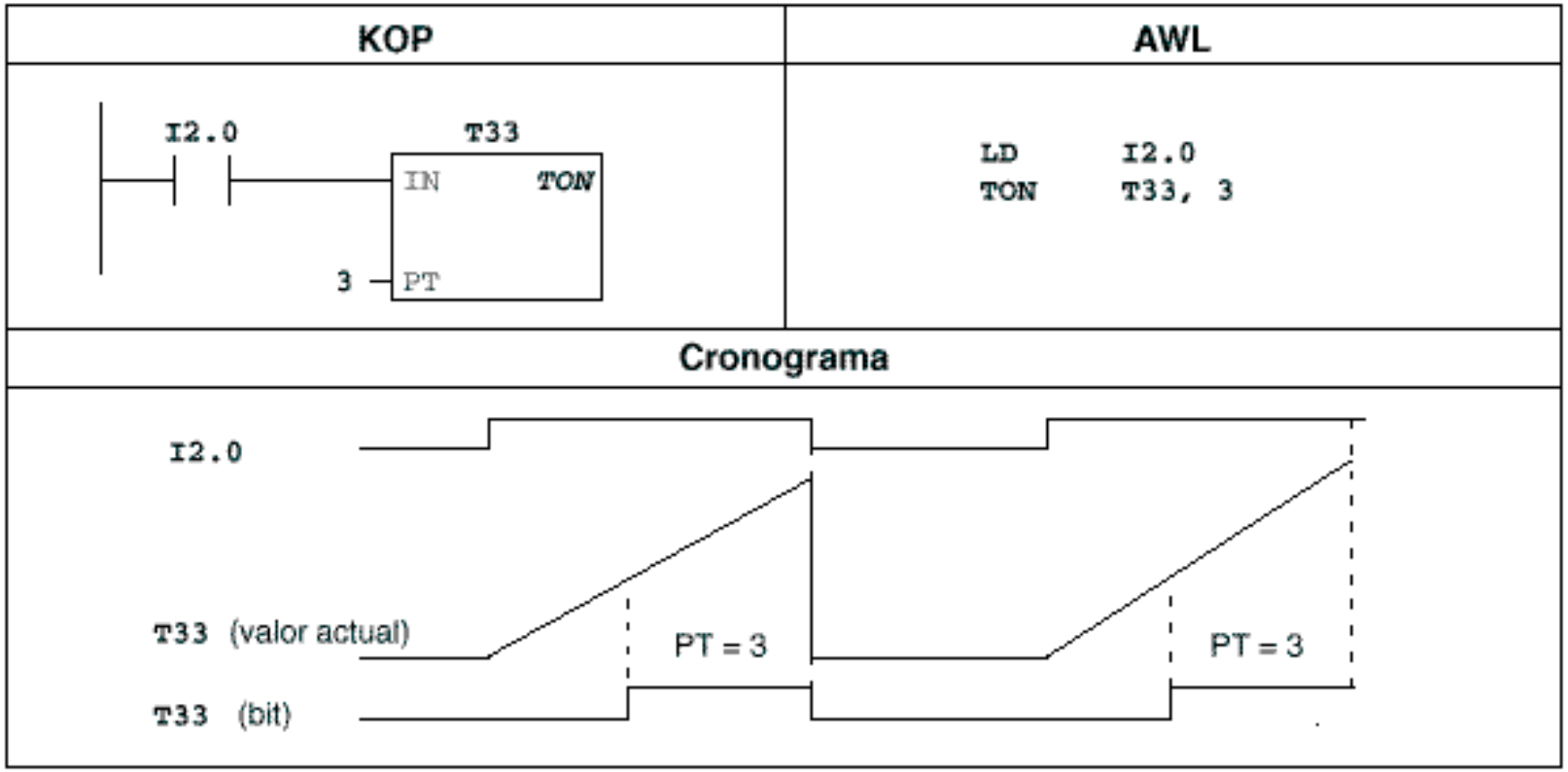
PT: VW, T, C, IW, QW, MW, SMW, AIW, constante

– Por ejemplo, el valor de contaje 50 en un temporizador de 100 milisegundos (ms) equivale a 5000 ms = 5 seg.

38

Operaciones con Temporizadores

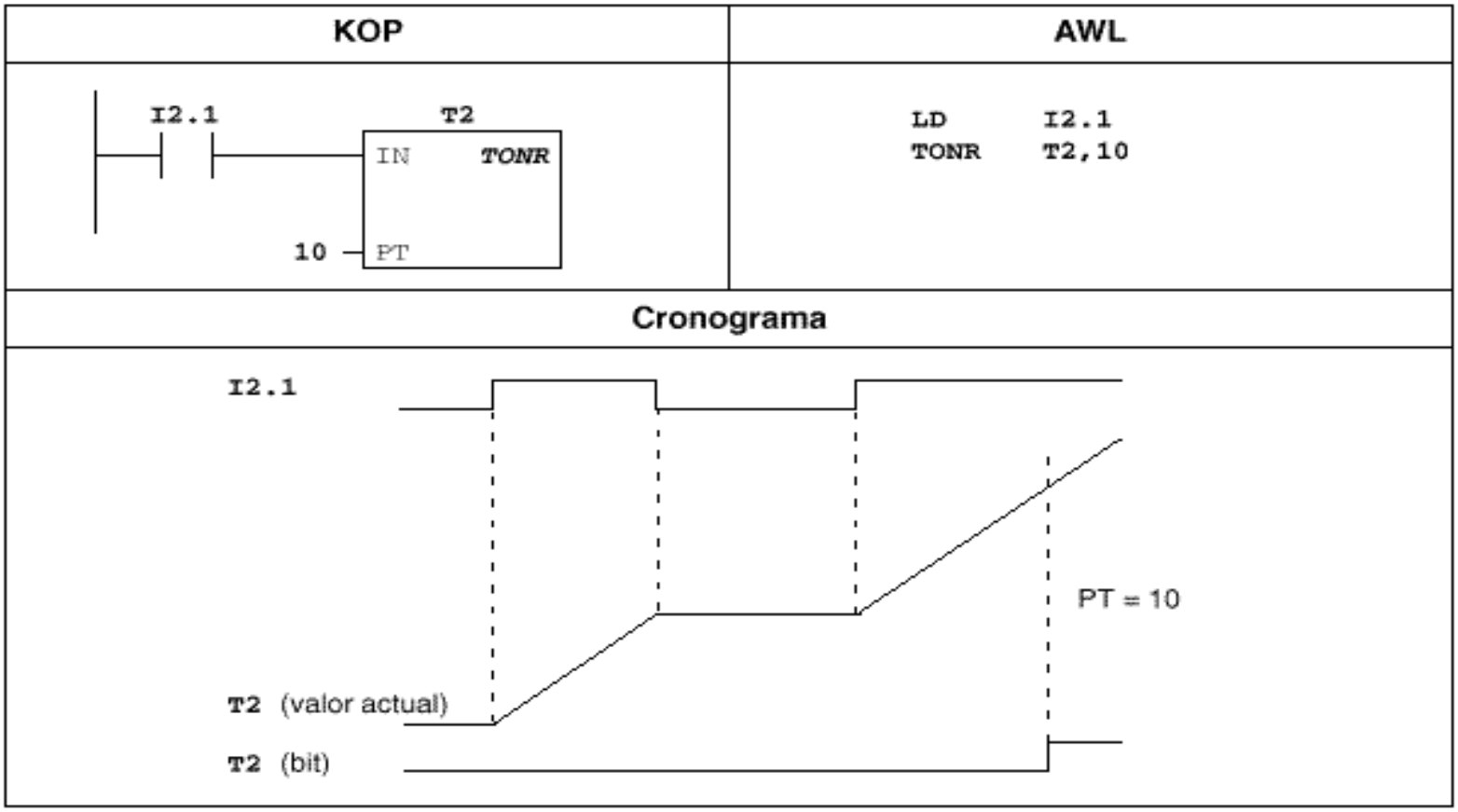
- TON



39

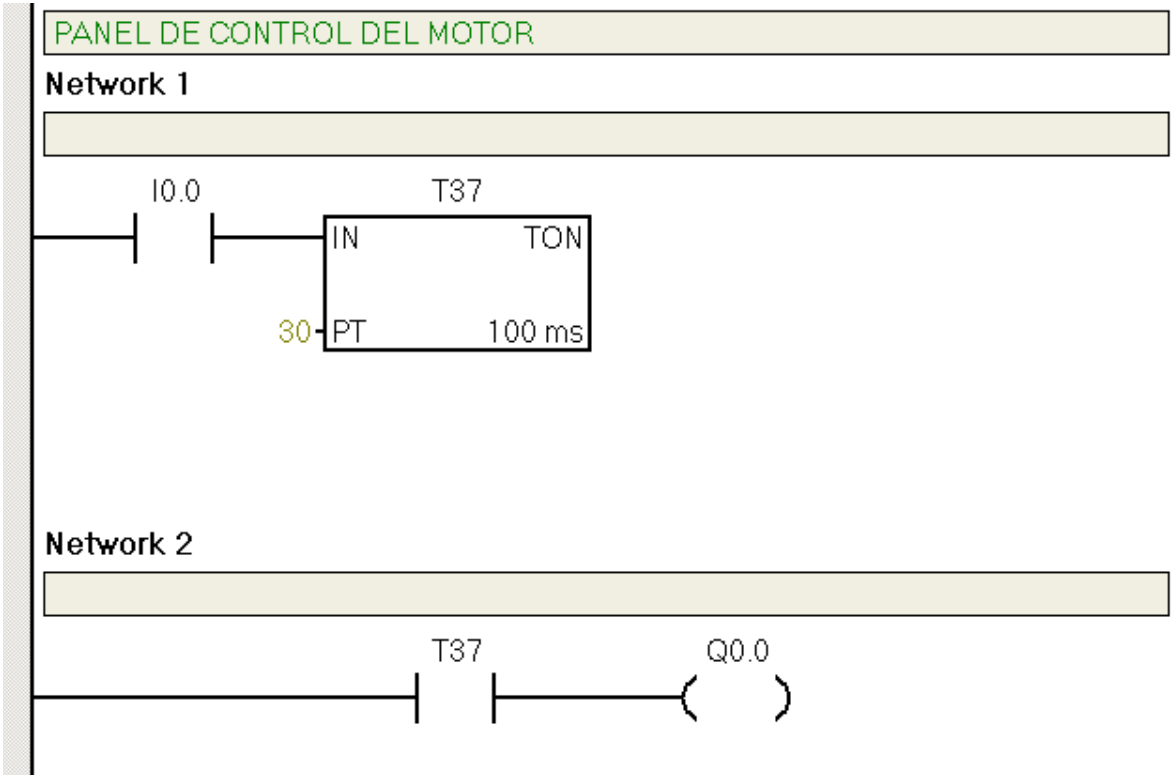
Operaciones con Temporizadores

- TONR



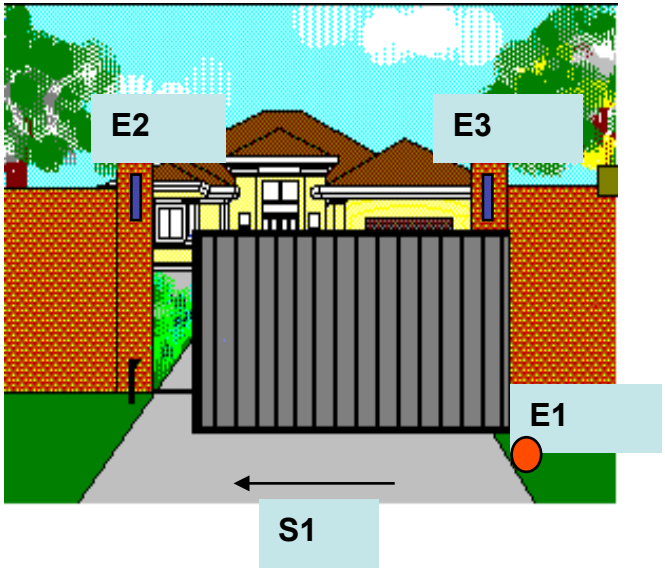
Operaciones con temporizadores

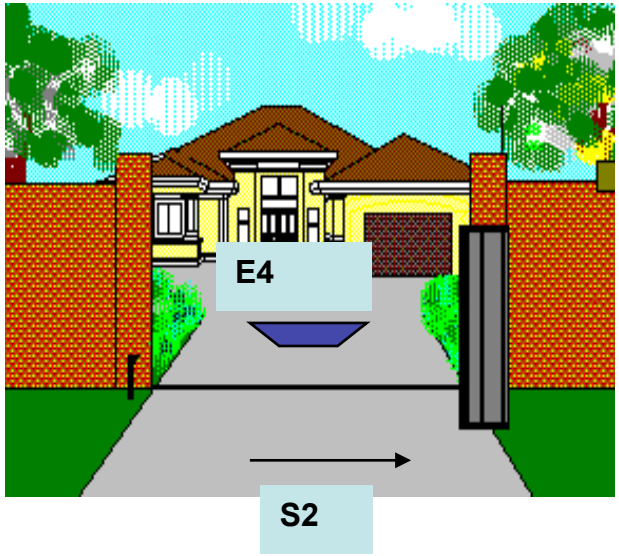
- Direccionamiento del área de temporizadores (T)
- Se accede al bit de temporización o al valor dependiendo de la operación que se esté realizando
  - Formato: T [número del temporizador] Ej. T24
  - P.e. siempre que la entrada I0.0 se encuentre activa más de 3 s, activar la entrada Q0.0.



Ejemplo

- Ejemplo: *"Control de una puerta corredera accionada por medio de un motor"*
  - *La puerta se abre al presionar el botón E1 situado enfrente de la puerta.*
    - *Si se activa E1, se cierra el contactor S2 (activar motor sentido apertura) y se mantiene cerrado hasta que se active el interruptor E3 de final de carrera.*
    - *Una vez abierta la puerta, se activa el temporizador T1, y transcurridos 10 segundos, la puerta se cierra mediante el contactor S1 (activa motor en sentido de cierre).*
    - *La acción de cerrar se produce hasta que se detecta fin de carrera E2 y si no se detecta un vehículo con el sensor de paso E4.*
  - *Las lámparas LED1 y LED2 indican cuándo se está cerrando o abriendo la puerta respectivamente.*



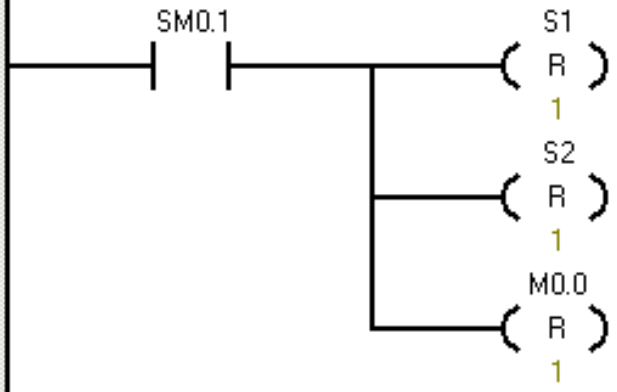


Ejemplo

Símbolo	Dirección	Comentario
S1	Q0.0	contactor cerrar puerta
S2	Q0.1	contactor abrir puerta
E1	I0.0	pulsador
E2	I0.1	final de carrera puerta cerrada
E3	I0.2	final de carrera puerta abierta
E4	I0.3	detector paso vehículos

Network 1 Título de segmento

Inicialización



Network 2

pulsador

Network 3

abrir puerta hasta activar final de carrera E3

Network 4

Network 5

Pasada la temporización, cerrar puerta

Network 6

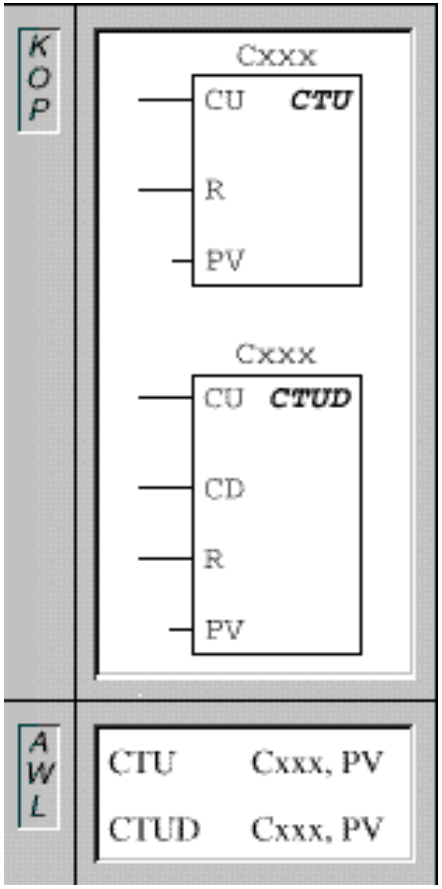
cerrar puerta hasta llegar a final de carrera E2

Network 7

Si de detecta el paso de un vehículo cuando se cierra la puerta, entonces abrir e ir a segmento 3

- ¿Qué ocurre si, mientras se está cerrando la puerta (S1), alguien presiona E1?
  - Modificar el programa para que funcione en ese caso.

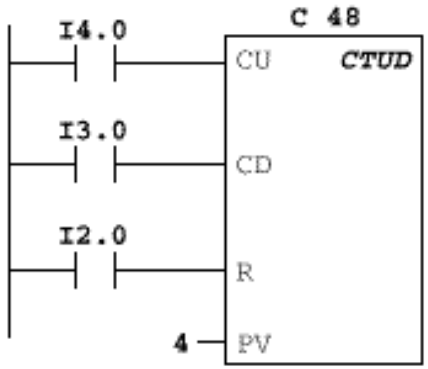
Operaciones con Contadores



- Contar adelante, Contar adelante/atrás
  - La operación **Contar adelante**
    - empieza a contar hasta el valor máximo cuando se produce un flanco positivo en la entrada de contaje adelante (CU).
    - Si el valor actual (Cxxx) es mayor o igual al valor de preselección (PV), se activa el bit de contaje (Cxxx).
    - El contador se inicializa al activarse la entrada de desactivación (R).
  - La operación **Contar adelante/atrás**
    - Cuenta hacia delante cuando se produce un flanco positivo en CU.
    - empieza a contar atrás cuando se produce un flanco positivo en la entrada de contaje atrás (CD).
  - Operandos:
    - Cxxx: 0 a 255
    - PV: VW, T, C, IW, QW, MW, SMW, AC, AIW, constante

Operaciones con Contadores

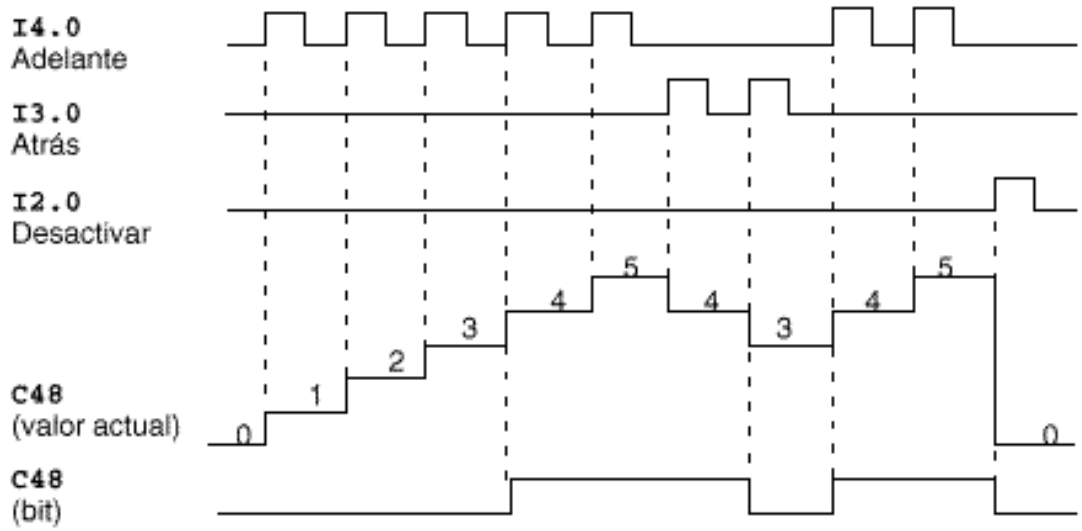
KOP



AWL

```
LD      I4.0    //Contar adelante
LD      I3.0    //Contar atrás
LD      I2.0    //Poner a 0
CTUD    C48, 4
```

Cronograma



Operaciones con Contadores

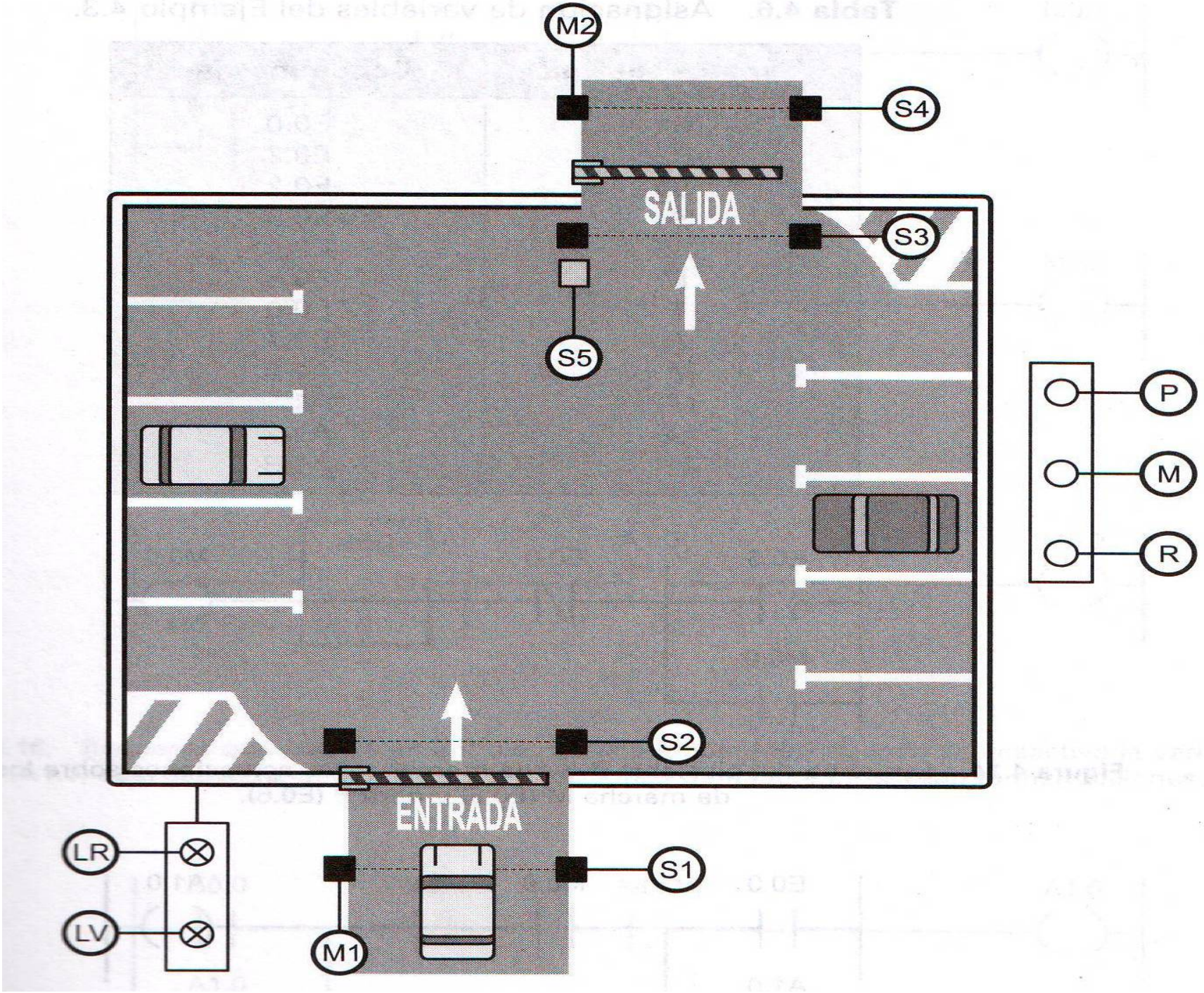
- Direccionamiento de los contadores (C)
  - Utilizados p.e. para llevar la cuenta de piezas producidas, ciclos completados, nº de coches en un parking.
  - Hay dos variables asociadas a los contadores:
    - Valor actual: En este número entero de 16 bits con signo se deposita el valor de contaje acumulado.
    - Bit del contador (bit C): Este bit se activa (se pone a 1) cuando el valor actual del contador es mayor o igual al valor predeterminado. (Éste último se introduce como parte de la operación).
  - A estas dos variables se accede utilizando la dirección del contador (C + número del contador).
  - Dependiendo de la operación utilizada, se accede al bit del contador o al valor actual.
  - Formato: C [número del contador] Ej. C20

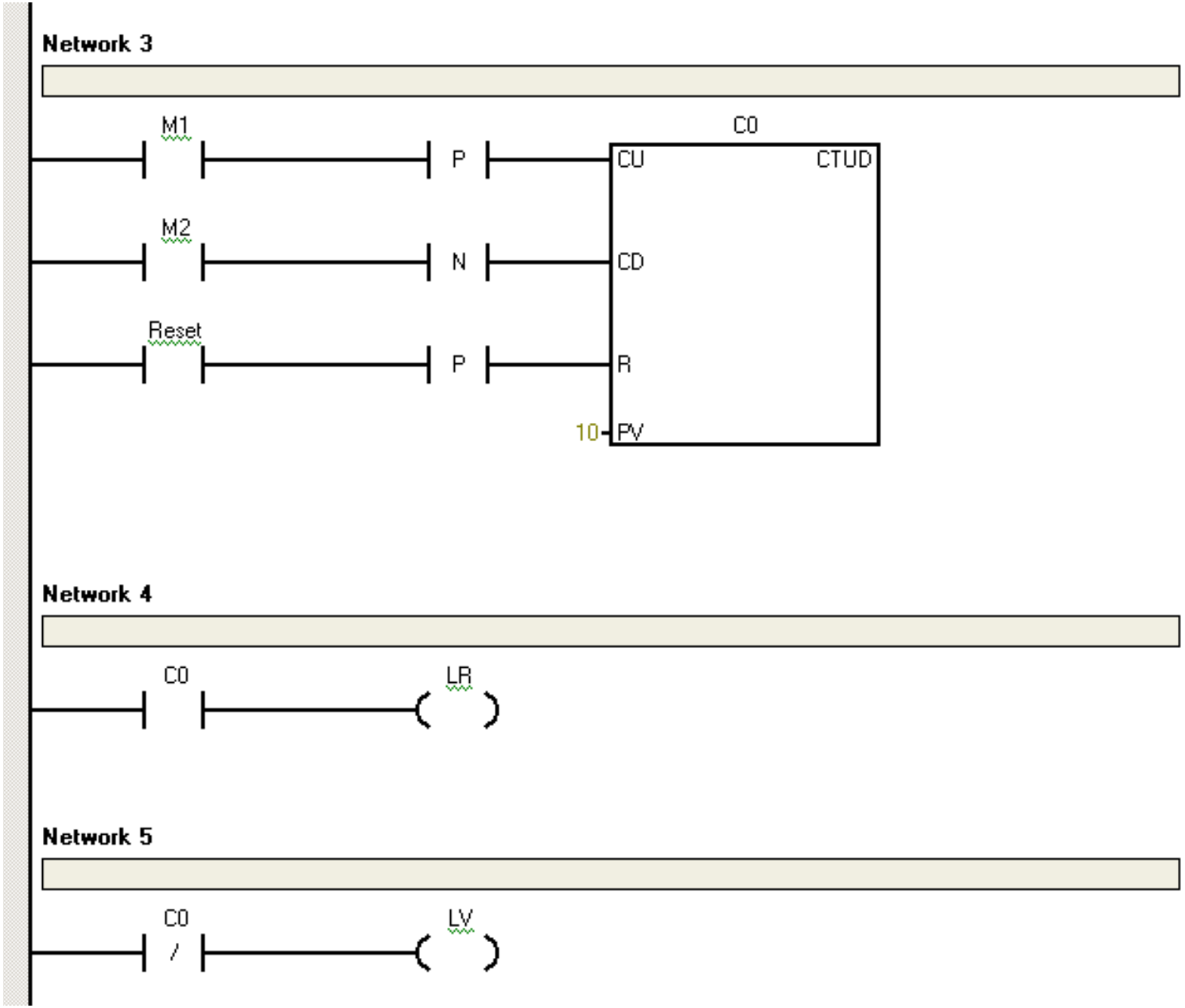
Contadores: Ejemplo

- Se presenta un aparcamiento.
- Se cuenta con los siguientes sensores y actuadores
  - **Dos motores M1 y M2 que controlan el acceso mediante una barrera.**
  - **A ambos lados de las barreras S1, S2, S3 y S4 son sensores de presencia.**
  - **S5 comprueba la ficha a la salida.**
  - **LR y LV son luces verde y roja respectivamente que informan al conductor.**
  - **M, P y R son pulsadores de Marcha, Paro y Reset.**

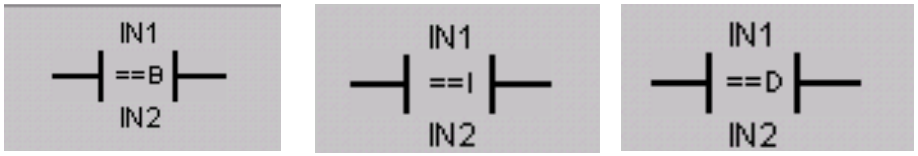
Contadores: Ejemplo

- **Funcionamiento:**
  - **M: enciende el sistema.**
  - **P: si se presiona detiene el sistema. No se permite la entrada de más vehículos.**
  - **R: reinicia el sistema.**
  - **La barrera de entrada debe abrirse si en el interior del garaje hay menos de 10 vehículos y se produce un flanco de subida del sensor S1. La barrera se baja con un flanco de bajada en S2.**
  - **La barrera de salida debe abrirse si se detecta una ficha en S5 y se producen flancos de subida y bajada en S3 y S4 respectivamente.**
  - **LV si hay plazas libres. LR en caso contrario.**


M2
S4
SALIDA
S3
S5
P
M
R
S2
ENTRADA
LR
LV
S1
M1


Network 3
M1
P
C0
CU
CTUD
M2
N
CD
Reset
P
R
10
PV
Network 4
C0
LR
Network 5
C0
/
LV

Operaciones de Comparación



– Las operaciones disponibles permiten comparar bytes, palabras y palabras dobles.

- Operaciones: IN1 == IN2, IN1 >= IN2, IN1 <= IN2, IN1 > IN2, IN1 < IN2, o IN1 <> IN2 (!=).
- Las comparaciones de bytes no llevan signo.
- Mientras que las comparaciones de palabras y palabras dobles sí que llevan signo (el bit más significativo indica el signo: 0 = + y 1 = -)

  [Hex: 7FFF > 8000 / Bin: 0111111111111111 > 1000000000000000]

  Dec: + 32767 > - 0

Operaciones de Comparación: ejemplo

– En un proceso se cuenta el nº de piezas defectuosas fabricadas, y se indica el estado de la máquina mediante tres luces.
  - Luz verde: menos de 10 piezas defectuosas producidas.
  - Luz amarilla, 10 o más piezas defectuosas.
  - Luz roja: 20 o más piezas defectuosas.
– Al producirse 20 o más piezas defectuosas se hace sonar una alarma→ se debe ajustar la máquina de nuevo.
– Conexiones con el autómata:
  - I0.0→ sensor pieza defectuosa. (0V pieza OK, 24V pieza defectuosa)
  - I0.1 botón de reset de la máquina (24V pulsado)
  - Q0.0→luz verde
  - Q0.1→luz naranja
  - Q0.2 → luz roja
  - Q0.3→ Alarma

MAQUINA DE PRODUCCIÓN DE PIEZAS

Network 1

Network 2

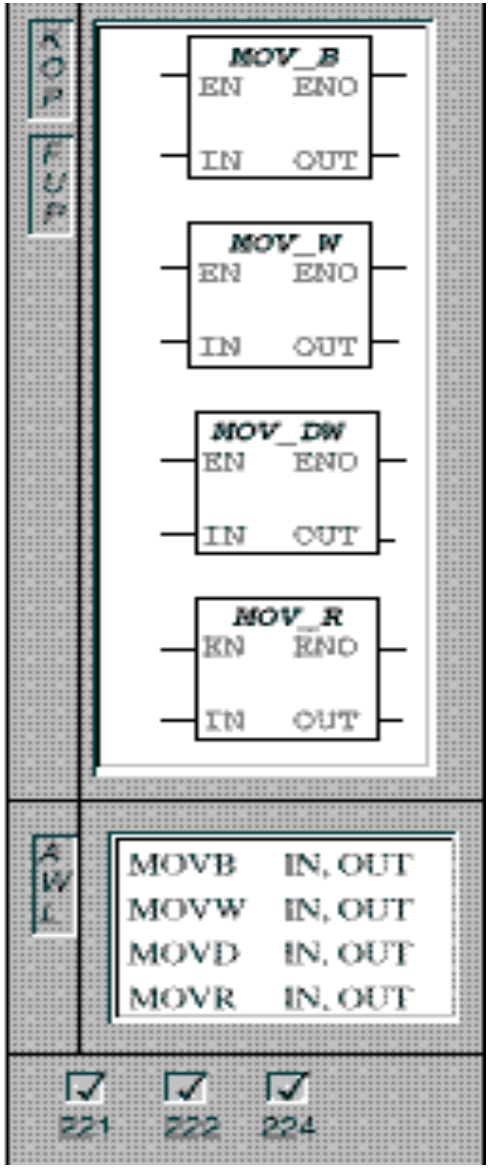
Network 3

Network 4

Temporizadores y comparaciones

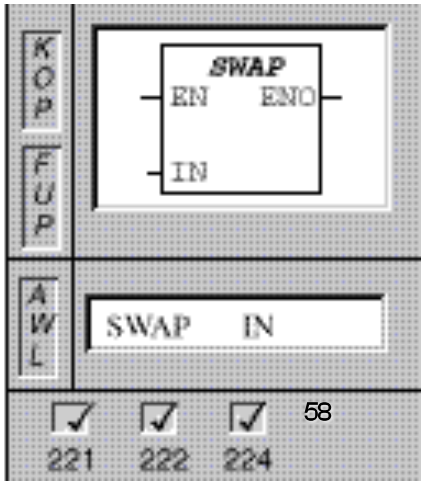
- Encender tres luces, en esta secuencia:
  - Q0.0 verde (durante 3s)
  - Q0.1 naranja (se enciende transcurridos 3s, luce durante 6s)
  - Q0.2 roja (se enciende transcurridos 9s, luce 2s)

Operaciones de Transferencia

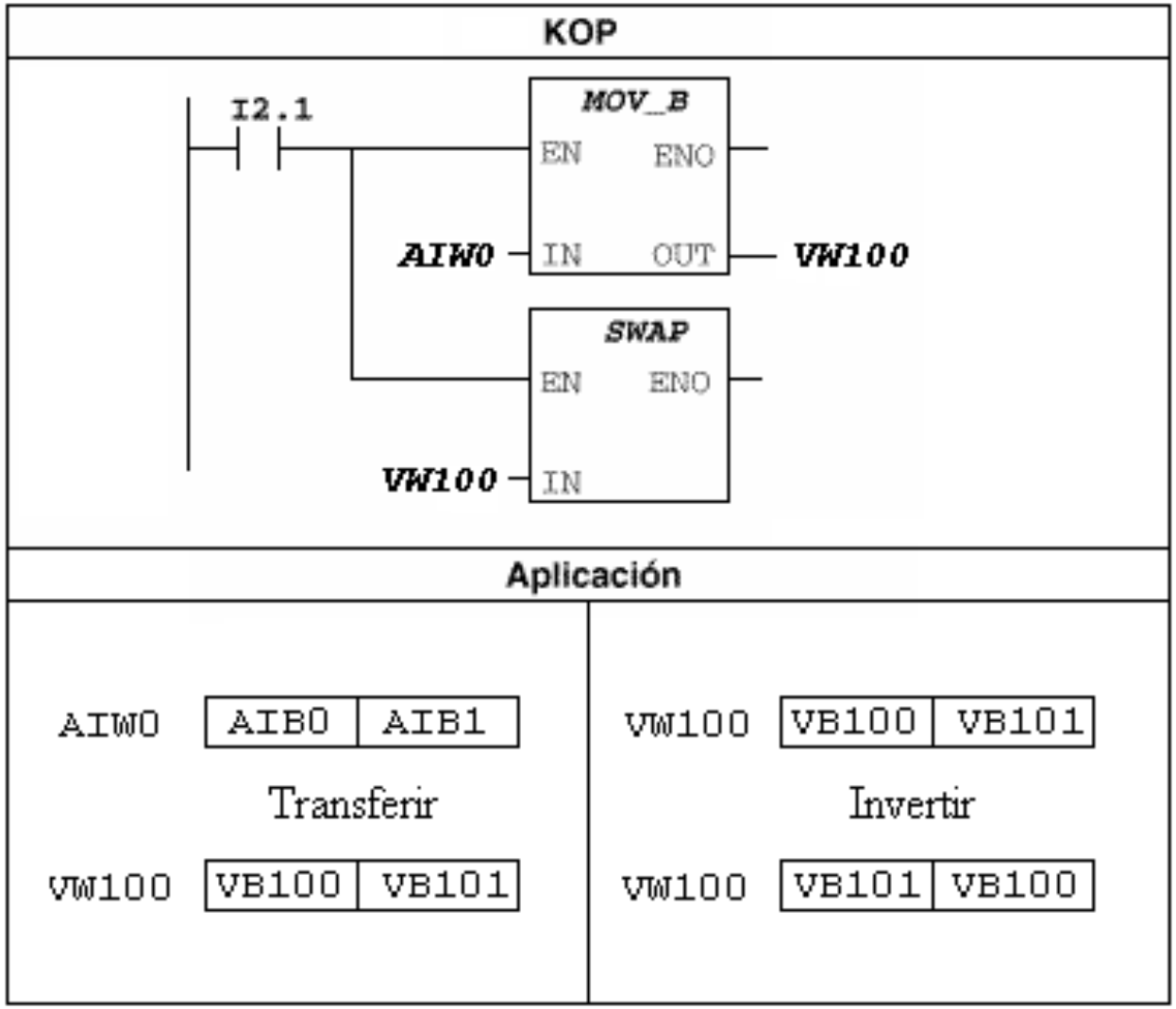


- Transferir byte, Transferir palabra, Transferir palabra doble y Transferir real
  - Las operaciones de transferencia se utilizan para transferir datos de una dirección a otra.

- La operación Invertir bytes de una palabra intercambia el byte más significativo y el byte menos significativo de una palabra (IN).



Operaciones de Transferencia



59

ÍNDICE

- Introducción a SETP 7: KOP
  - Lenguaje KOP: Diagrama de contactos
    - Operaciones básicas: contactos y salidas
    - Operaciones con temporizadores
    - Operaciones con contadores
    - Operaciones de comparación
    - Operaciones de transferencia
  - Distribución de la memoria
  - Ejemplos

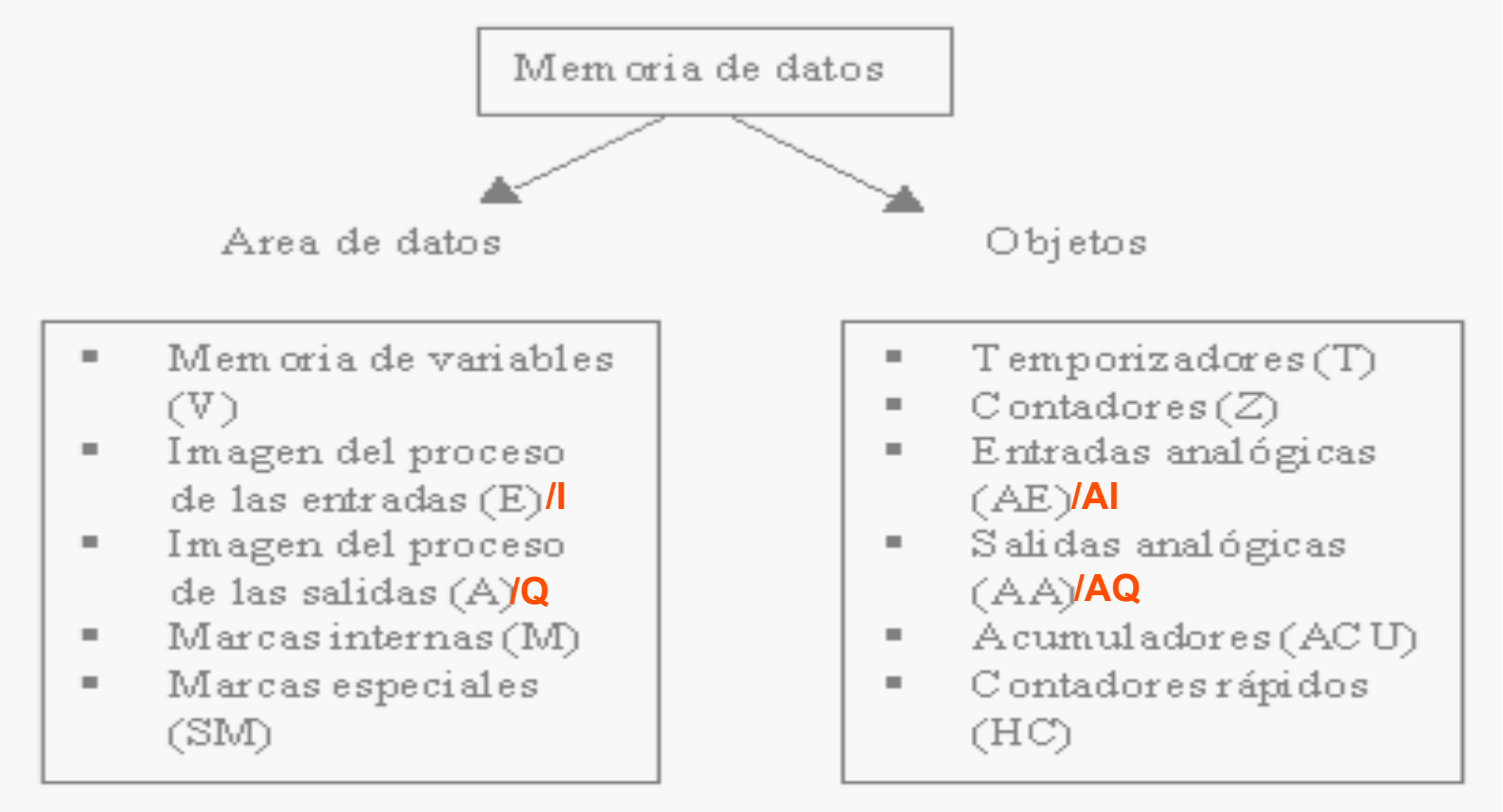
Distribución de la memoria

- Memoria de programa
  - La memoria de programa contiene las operaciones de esquema de contactos (KOP) o de lista de instrucciones (AWL), que ejecuta el autómata programable para la aplicación deseada.

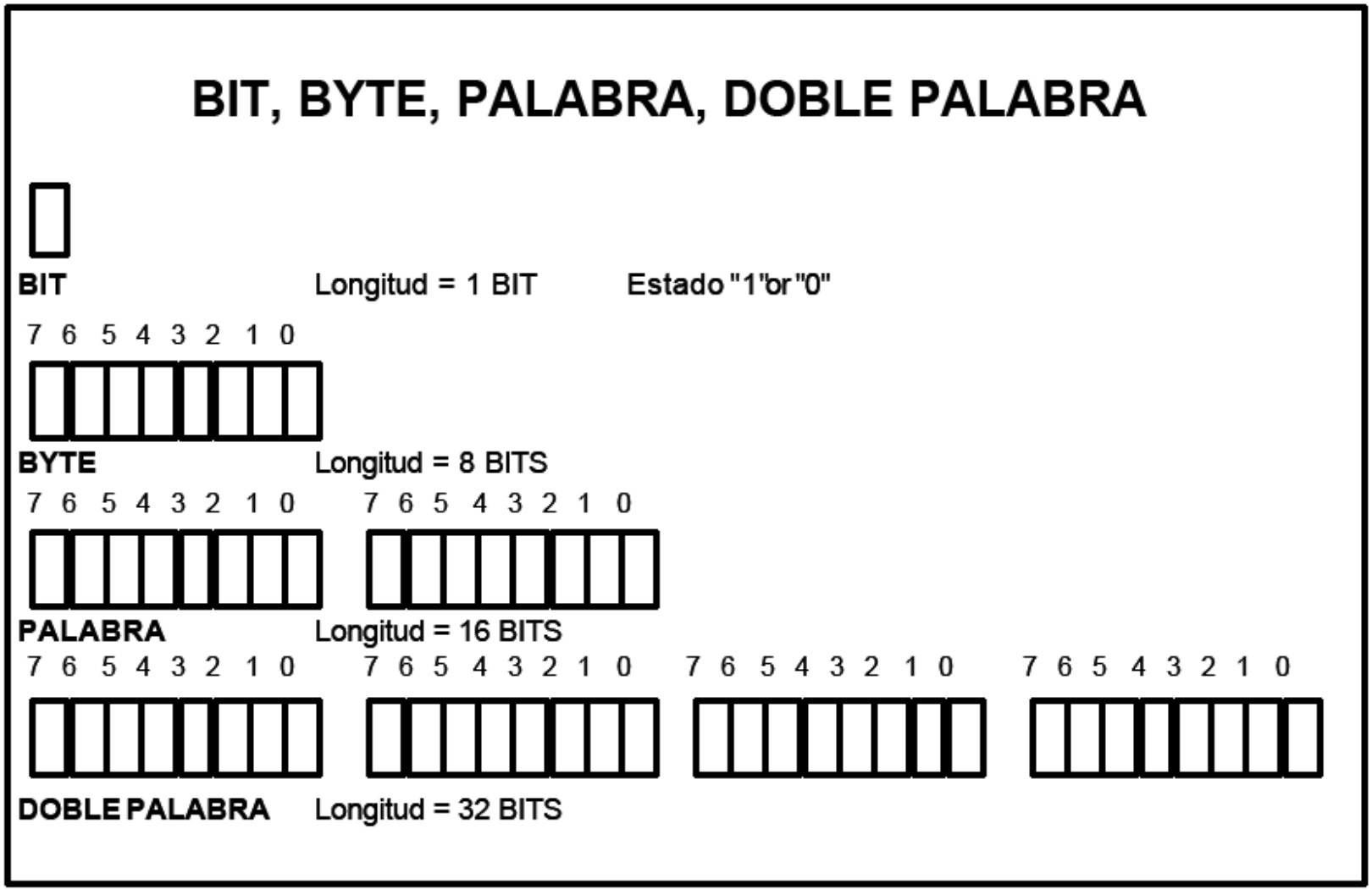
- Memoria de parámetros
  - La memoria de parámetros permite almacenar determinados parámetros configurables, tales como contraseñas, direcciones de módulos

Distribución de la memoria

- Memoria de datos
  - La memoria de datos es el área de trabajo a la que accede el programa de aplicación (también denominado programa de usuario).



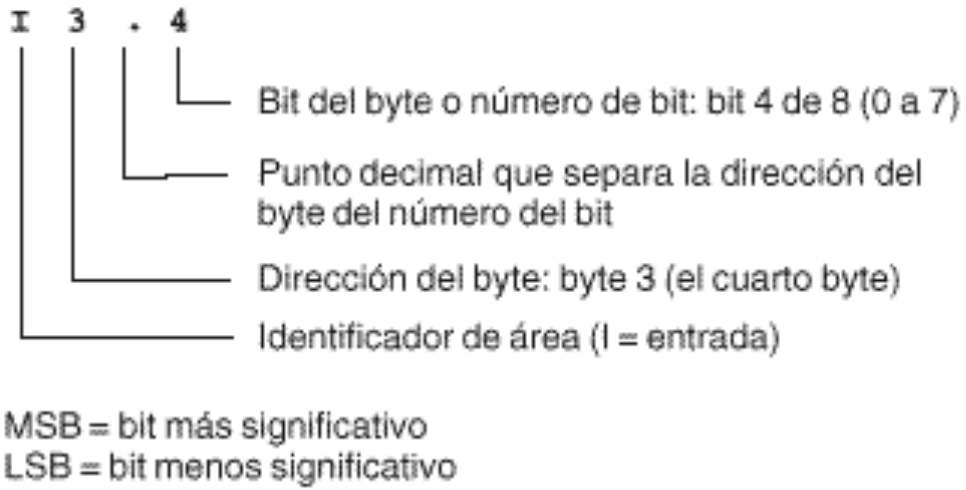
Distribución de la memoria

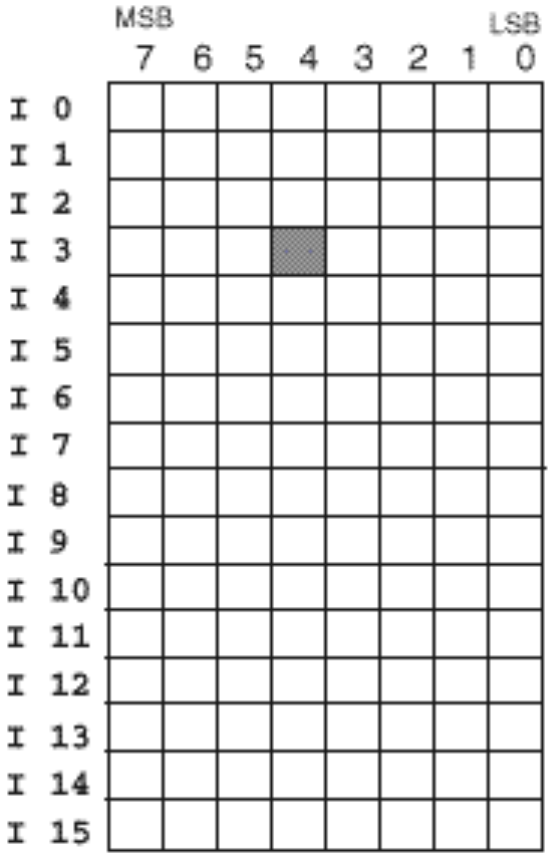


Direccionamiento de la Memoria

- Acceso a un bit
  - “Identificador de área” “dirección del byte” . “nº del bit”

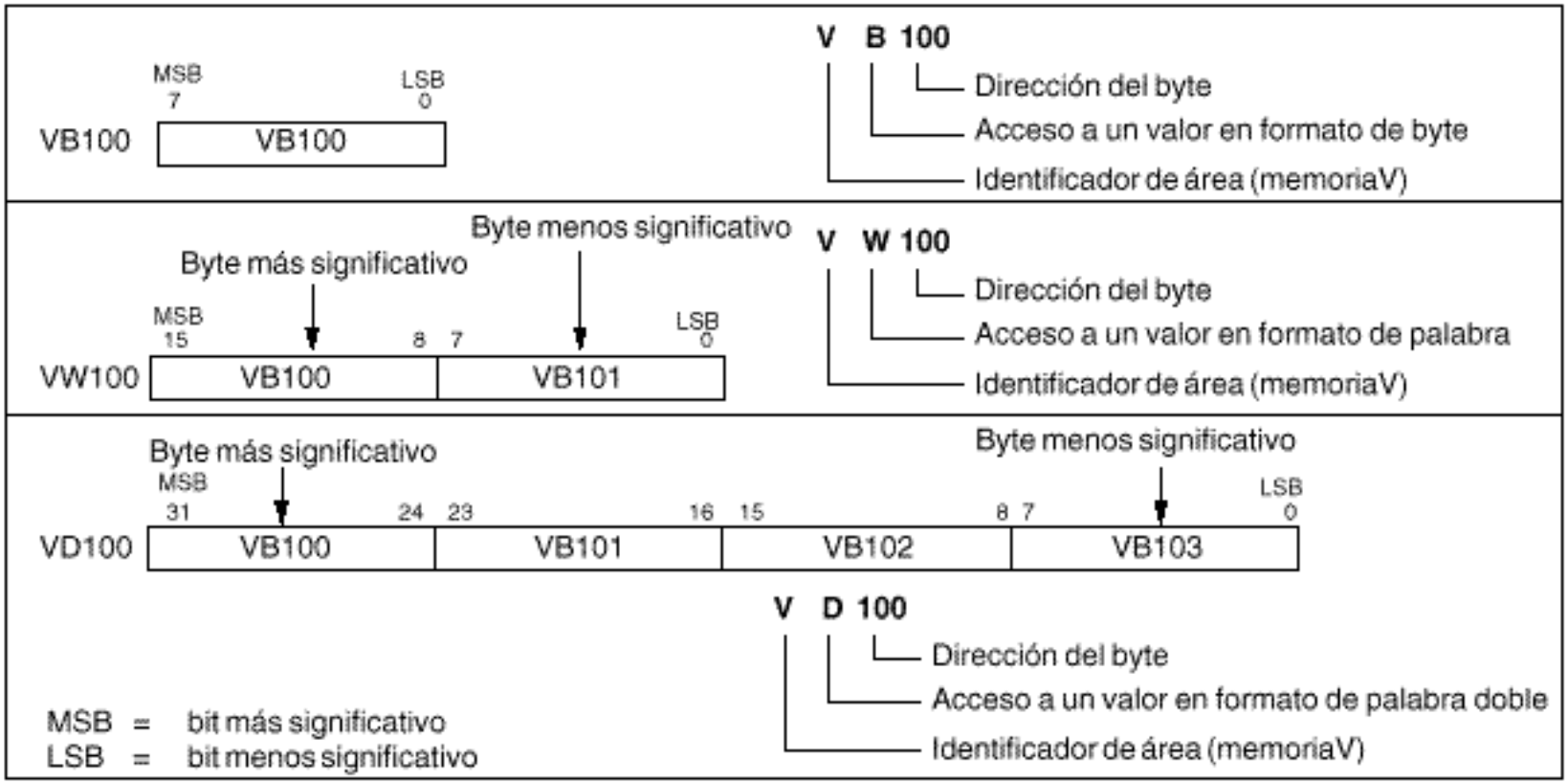
    Ejemplo I 0.0 el bit 0 del byte 0 de las entradas





Direccionamiento de la Memoria

- Se puede acceder a diversas áreas de la memoria de la CPU (V, I, Q, M, SM) en formato byte, palabra y palabra doble



Direccionamiento de la Memoria

- Direccionamiento de la imagen del proceso de las entradas (I/E)
  - Formato:
    - Bit — I [módulo].[direcc. del bit] — I0.1
    - Byte/word/double — I [tamaño][direcc. del byte inicial] — IB4
- Direccionamiento de la imagen del proceso de las salidas (**Q/A**)
  - Formato:
    - Bit — Q [módulo].[direcc. del bit] — Q1.1
    - Byte/word/double — Q [tamaño][direcc. del byte inicial] — QB5
- Direccionamiento del área de marcas (**M**)
  - Las marcas internas (área de marcas M) se pueden utilizar como relés de control para almacenar el estado intermedio de una operación u otras informaciones de control
  - Formato:
    - Bit — M [direcc. del byte].[direcc. del bit] — M26.7
    - Byte/word/double — M [tamaño][direcc. del byte inicial] — MD20

Direccionamiento de la Memoria

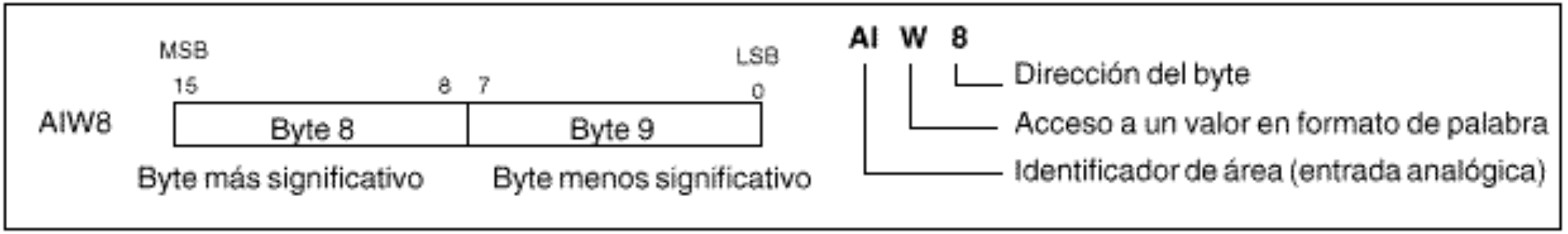
- Direccionamiento de la memoria de variables (**V**)
  - Usada para almacenar datos generales.
  - Formato:
    - Bit — V [direcc. del byte].[direcc. del bit] — V10.2
    - Byte, palabra, p. Doble — V [tamaño][direcc. del byte inicial] — VW100
- Direccionamiento de las marcas especiales (**SM**)
  - Estas marcas se pueden utilizar para seleccionar y controlar algunas funciones especiales de la CPU S7-200, tales como:
    - Un bit que se activa sólo en el primer ciclo.
    - Bits que se activan y se desactivan en determinados intervalos (SM0.5, activado y desactivado cada 0.5s).
    - Bits que muestran el estado de operaciones matemáticas y de otras operaciones.
  - Formato:
    - Bit — **SM [direcc. del byte].[direcc. del bit]** — **SM0.1**
    - Byte,palabra,p. Doble — **SM [tamaño][direcc. del byte inicial]** — **SMB86**

Direccionamiento de la Memoria

- Marcas especiales (primer byte, sólo lectura):
  - ***SM0.0:*** ***Este bit siempre está activado.***
  - ***SM0.1:*** ***Este bit se activa sólo en el primer ciclo. Se utiliza, por ejemplo, para inicializar las variables que se necesiten.***
  - ***SM0.4:*** ***Este bit ofrece un reloj que está desactivado durante 30 segundos y activado durante 30 segundos. Ofrece un retardo fácil de utilizar.***
  - ***SM0.5:*** ***Este bit ofrece un reloj que está desactivado durante 0,5 segundos y activado durante 0,5 segundos.***
  - ***SM0.6:*** ***Este bit es un reloj de ciclo que está activado en un ciclo y desactivado en el ciclo siguiente. Se puede utilizar como entrada de contaje de ciclos.***

Direccionamiento de la Memoria

- Direccionamiento de las entradas analógicas (**AI**)
  - Se utilizarán para leer las entradas analógicas (p.ej. temperatura, tensión, etc).
  - La CPU S7-200 convierte valores reales analógicos en valores digitales en formato de palabra (de 16 bits).
    - Comienzan siempre en bytes pares (p.ej. 0, 2, 4, etc)., es preciso utilizar direcciones con bytes pares (p.ej. AIW0, AIW2, AIW4, etc)
  - Formato:
    - AIW [dirección del byte inicial] AIW4



Direccionamiento de la Memoria

- Direccionamiento de las salidas analógicas (AQ)
  - La CPU S7-200 convierte valores digitales en formato de palabra (de 16 bits) a valores reales analógicos (p.ej. corriente o voltaje), proporcionales al valor digital.
    - Comienzan siempre en bytes pares (p.ej. 0, 2, 4, etc)., es preciso utilizar direcciones con bytes pares (p.ej. AQW0, AQW2, AQW4, etc.) para acceder a las mismas.
  - Formato:
    - AQW [dirección del byte inicial] AQW4

Entradas y salidas integradas y ampliadas mediante módulos de expansión.

CPU 221	
I0.0	Q0.0
I0.1	Q0.1
I0.2	Q0.2
I0.3	Q0.3
I0.4	
I0.5	

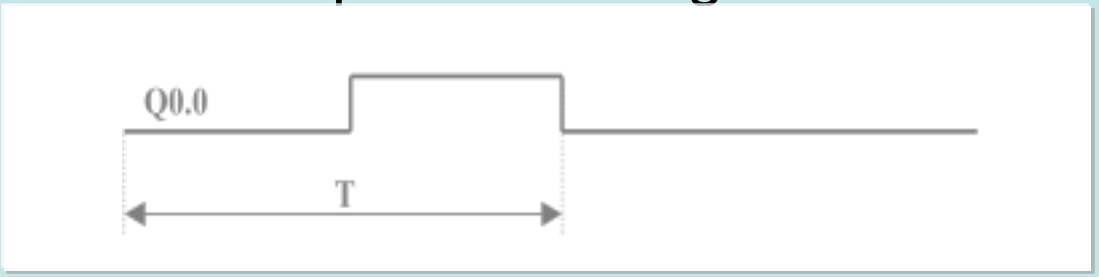
CPU 224		4 En / 4 Sal		8 Entradas	8 Salidas
I0.0	Q0.0	I2.0	Q2.0	I3.0	Q3.0
I0.1	Q0.1	I2.1	Q2.1	I3.1	Q3.1
I0.2	Q0.2	I2.2	Q2.2	I3.2	Q3.2
I0.3	Q0.3	I2.3	Q2.3	I3.3	Q3.3
I0.4	Q0.4			I3.4	Q3.4
I0.5	Q0.5			I3.5	Q3.5
I0.6	Q0.6			I3.6	Q3.6
I0.7	Q0.7			I3.7	Q3.7
I1.0	Q1.0				
I1.1	Q1.1				
I1.2					
I1.3					
I1.4					
I1.5					

Área	Descripción	Acceso a bits	Acceso a bytes	Acceso a palabras	Acceso a palabras dobles	Puede ser remanente	Se puede forzar
I	Entradas digitales e imagen del proceso de las entradas	lectura / escritura	lectura / escritura	lectura / escritura	lectura / escritura	no	sí
Q	Salidas digitales e imagen del proceso de las salidas	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 SM29 son de 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 y bits de temporizadores	Bit T lectura / escritura	no	Valor actual T lectura / escritura	no	Valor actual T - sí Bit T - no	no
C	Valores actuales y bits de contadores	Bit C lectura / escritura	no	Valor actual C lectura / escritura	no	Valor actual C - sí Bit C - no	no
HC	Valores actuales de contadores rápidos	no	no	no	sólo lectura	no	no
AI	Entradas analógicas	no	no	sólo lectura	no	no	sí
AQ	Salidas analógicas	no	no	sólo escritura	no	no	sí
AC	Acumuladores	no	lectura / escritura	lectura / escritura	lectura / escritura	no	no
L	Memoria de variables locales	lectura / escritura	lectura / escritura	lectura / escritura	lectura / escritura	no	no
S	SCR	lectura / escritura	lectura / escritura	lectura / escritura	lectura / escritura	no	no

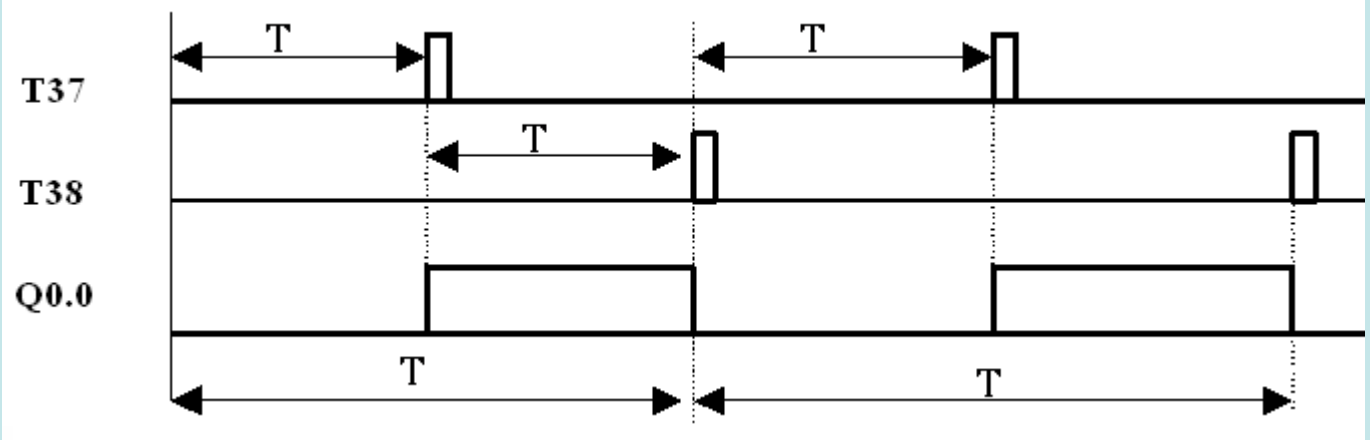
Acceso en formato de:	**Área de memoria**	**CPU 221**	**CPU 222**	**CPU 224**	**CPU 226**
Bit (*Byte.bit*)	V	0.0 - 2047.7	0.0 - 2047.7	0.0 - 5119.7 V 1.22 0.0 - 8191.7 V 2.00 0.0 - 10239.7 XP	0.0 - 5119.7 V 1.23 0.0 - 10239.7 V 2.00
	I	0.0 - 15.7	0.0 - 15.7	0.0 - 15.7	0.0 - 15.7
	Q	0.0 - 15.7	0.0 - 15.7	0.0 - 15.7	0.0 - 15.7
	M	0.0 - 31.7	0.0 - 31.7	0.0 - 31.7	0.0 - 31.7
	SM	0.0 - 179.7	0.0 - 299.7	0.0 - 549.7	0.0 - 549.7
	S	0.0 - 31.7	0.0 - 31.7	0.0 - 31.7	0.0 - 31.7
	T	0 - 255	0 - 255	0 - 255	0 - 255
	C	0 - 255	0 - 255	0 - 255	0 - 255
	L	0.0 - 59.7	0.0 - 59.7	0.0 - 59.7	0.0 - 59.7
Byte	VB	0 - 2047	0 - 2047	0 - 5119 V 1.22 0 - 8191 V 2.00 0 - 10239 XP	0 - 5119 V 1.23 0 - 10239 V 2.00
	IB	0 - 15	0 - 15	0 - 15	0 - 15
	QB	0 - 15	0 - 15	0 - 15	0 - 15
	MB	0 - 31	0 - 31	0 - 31	0 - 31
	SMB	0 - 179	0 - 299	0 - 549	0 - 549
	SB	0 - 31	0 - 31	0 - 31	0 - 31
	LB	0 - 59	0 - 59	0 - 59	0 - 59
	AC	0 - 3	0 - 3	0 - 3	0 - 3

Ejemplos

- Realizar el programa de control que obtenga en la salida Q0.0 una señal periódica de período 6 segundos.



- Para conseguir una señal periódica se utilizan dos temporizadores con retardo a la conexión TON, T37 y T38.



74

Ejemplo

- Example: *Control a sliding door by means of electric motor*
  - *The door opens when E1 button is pressed.*
    - *If E1 is active, the S2 driver is on (move motor to open door). The S2 switch is on untill the E3 switch is activated ().*
    - *Once the door is open, the system waits 10 seconds and then the door is closed by means of S1.*
    - *The door is closed untill E2 is detected and if the car is not detected with the E4 sensor (presence sensor).*
  - *LED1 and LED2 show the actions of opening and closing the door respectively.*