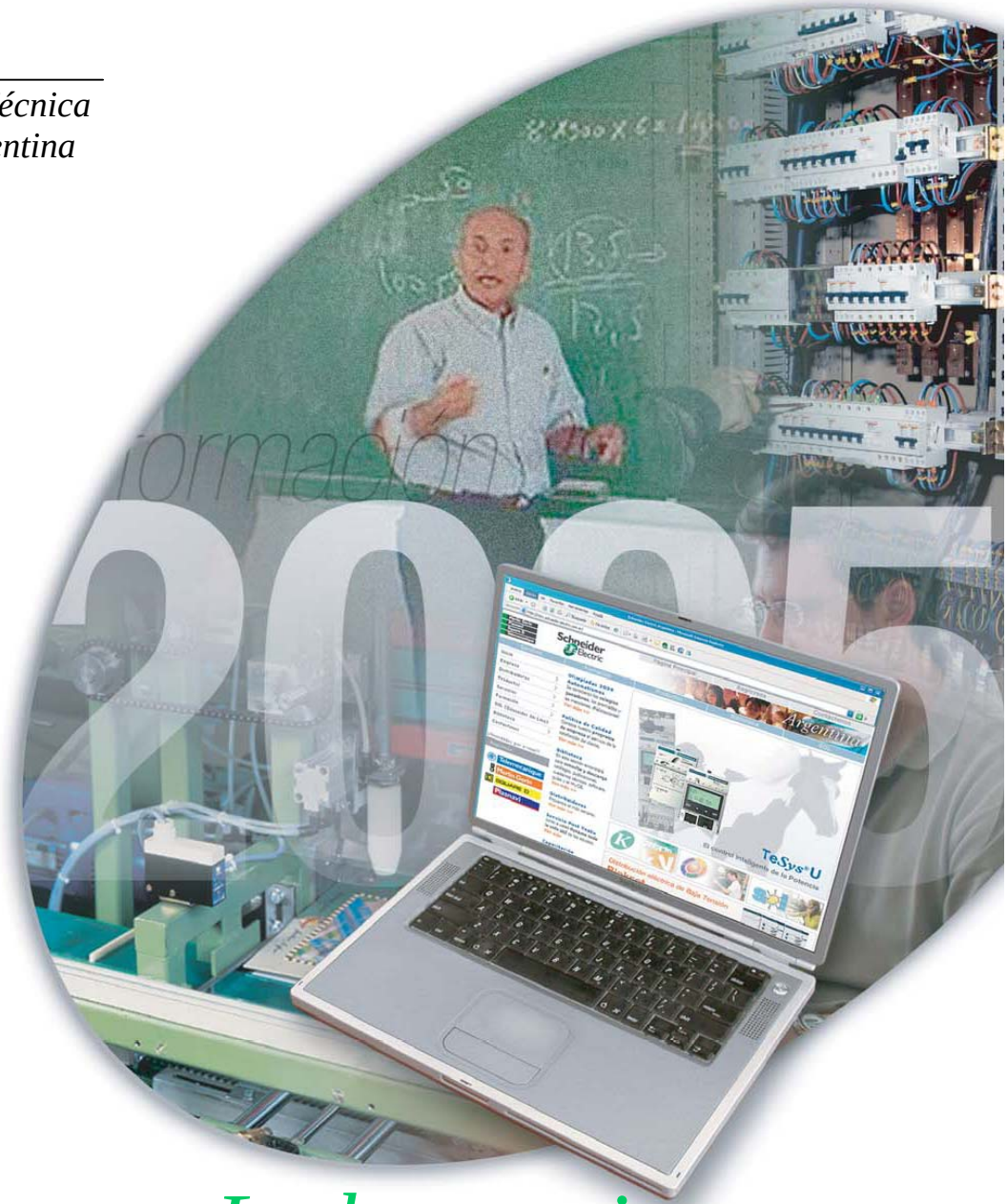
PLC

Introducción a la programación de autómatas según Norma IEC 61131-3

Manual de Curso

Centro de Formación Técnica
Schneider Electric Argentina



La herramienta para perfeccionar sus conocimientos

Merlin Gerin
Modicon
Plasnavi
Square D
Telemecanique

Schneider Electric

Índice

Sección 1 - Prestaciones

Sección 2 - Ejercitación

1.- CONTROLADOR LOGICO PROGRAMABLE.

1.1 Definición.

Es un dispositivo electrónico programable por el usuario y que esta destinado a gobernar, dentro de un entorno industrial, máquinas o procesos lógicos y/o secuenciales.

1.2 Funciones del PLC.

· Reemplazar la lógica de relés para el comando de motores, máquinas, cilindros neumáticos e hidráulicos, etc...

· Reemplazar temporizadores y contadores electromecánicos.

· Efectuar procesos de control de lazo abierto y/o cerrado.

· Actuar como interfase computador - proceso de fabricación.

· Efectuar diagnóstico de falla y alarma.

· Control y comando de tareas repetitivas, que pueden ser peligrosas para los operarios.

· Regulación de los aparatos que estén situados en ambientes peligrosos.

· Regulación de aparatos remotos desde un punto de la fábrica

1.3 Descripción.

Un autómata programable se compone de cuatro subgrupos principales:

· Procesador (CPU).

· Entradas.

· Salidas.

· Memoria.

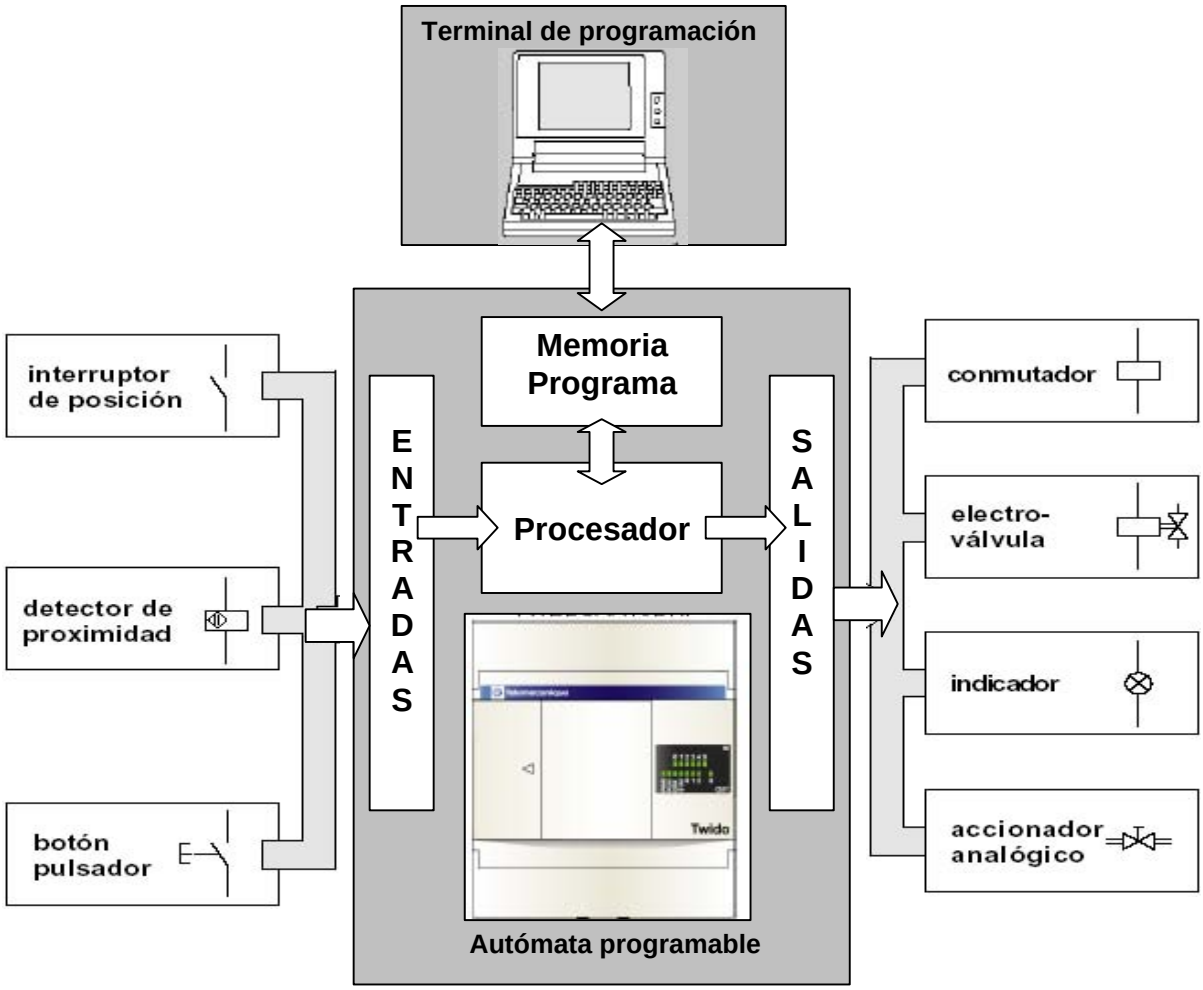
1.3.1 Procesador (CPU : Unidad Central de Proceso)

Recibe, interpreta y ejecuta las instrucciones del programa en curso

1.3.2 Módulos de entradas / salidas.

Cumplen la función de conectar el equipo con el mundo exterior. Todas las señales provenientes del campo son informadas a la CPU luego de ser captadas por los

módulos de entradas. A su vez las ordenes generadas por la CPU son comunicadas a los elementos del proceso bajo control a través de los módulos de salidas.



1.3.3 Memoria.

La memoria es la que contiene tanto el programa a ejecutar, como los datos generados por el programa en curso. Existen diferentes tipos de memoria, a continuación detallamos las más usadas:

- **RAM (Random Access Memory):** Es posible direccionar cualquier punto o dato almacenado en la memoria, por eso se la llama de acceso aleatorio. Son accesibles en lectura y escritura. Este tipo de memoria es volátil, es decir, la información almacenada en esta se pierde al quitarle la alimentación.

- **ROM (Read Only Memory):** Esta memoria es accesible únicamente en lectura, los datos guardados en ellas son grabados por el fabricante. Es también de acceso aleatorio y no es volátil.

· **PROM (Programmable Read Only Memory):** Esta memoria tiene las mismas características de la ROM, pero es grabada por el usuario. Esta programación puede efectuarse solamente una vez.

· **EPROM (Erasable Programmable Read Only Memory):** Esta memoria suma a las características de la PROM, la posibilidad de ser borrada por el usuario, exponiéndola a una fuente de luz ultravioleta.

· **EEPROM (Electrically Erasable Programmable Read Only Memory):** Estas memorias se diferencian de las EPROM en que el borrado se realiza con impulsos eléctricos.

Los PLC en general cuentan con dos memorias, una RAM y una EEPROM. La RAM contiene el programa a ejecutar y los datos generados por éste. La EEPROM contiene una copia de seguridad, que sirve de respaldo al programa guardado en la RAM. Si el PLC detecta que se borró el programa en RAM, automáticamente recupera la copia, y empieza a ejecutarlo.

1.3.4 Accesorios de Automatismos.

Existen otros elementos dentro y fuera del PLC que debemos mencionar:

· Módulo de alimentación: puede ser en AC o DC.
· Módulos de extensión: permite ampliar la capacidad de E/S del PLC.
· Terminal de Programación: Permite crear, configurar transferir el programa de usuario, poner en marcha el automatismo, realizar la depuración del programa, etc.

2.- HARDWARE TWIDO.

2.1 Presentación del producto.

El autómata Twido surge del desarrollo conjunto entre Modicon y Telemecanique, marcas de Schneider Electric y especialistas en autómatas programables industriales (PLC).

Dedicado a la automatización de instalaciones industriales simples y de máquinas pequeñas, Twido se encuentra disponible en dos versiones: **Compacto** y **Modular**, que comparten opcionales, extensiones de E/S y el software de programación, otorgándole máxima flexibilidad y simplicidad de uso.

De dimensiones reducidas y con una puesta en marcha muy sencilla, dispone de dos formas de programación:

a) el lenguaje lista de instrucciones «list»

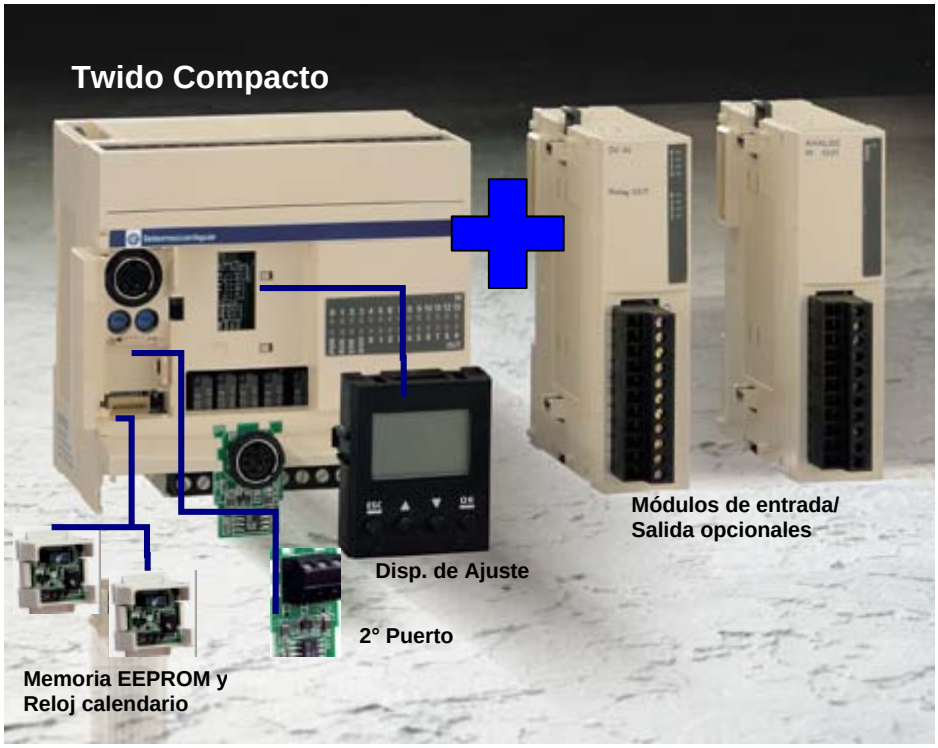
b) lenguaje a contactos «ladder».

Twido permite además la creación de páginas GRAFCET, facilitando la programación de procesos secuenciales.

La programación se efectúa con la ayuda de una PC, con el software TwidoSoft.

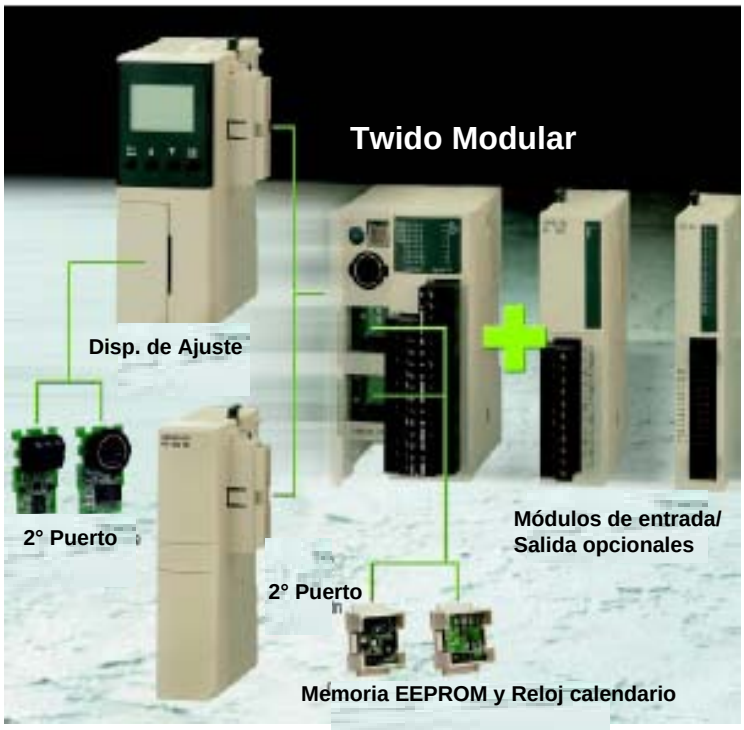
2.1.1 Twido Compacto.

Para optimizar tiempos costos en la instalación, el Twido compacto está disponible en tres tallas: 10, 16 y 24 E/S. Este último, con la posibilidad de ser ampliado incorporándole módulos de entrada o salidas digitales o analógicas. La alimentación del modelo compacto es en corriente alterna (100 – 240 Vca), y posee entradas de 24 Vcc, y salidas a relé.



2.1.2 Twido Modular.

Para soluciones hechas a medida, maximizando la eficiencia de las máquinas, el Twido modular está disponible en dos tallas: 20 y 40 E/S. La alimentación del modelo modular es en 24 Vcc, y posee entradas de 24 Vcc y salidas transistores, a relés o mixtas (transistores + relé). Además, cada Twido modular trae de base una entrada analógica de 0 a 10 Vcc.



2.2 Principales características.

Más flexibilidad para componer un autómata programable acorde a su necesidad:

- Con sus 6 modelos de CPU compactas y modulares, Twido le ofrece múltiples posibilidades para resolver su automatismo.
- Gracias a una gran variedad de módulos usted puede encontrar exactamente lo que necesita en aplicaciones estándar de 10 a 100 E/S.
- Ya sea si necesita un reloj calendario o un 2do puerto serie, etc. Twido le ofrece un amplio abanico de opciones. Evalúe su necesidad y utilice lo estrictamente necesario.

Más comunicación.

- Posibilidad de un 2° puerto serie opcional para los Twido Compactos y Modulares (en éstos últimos a través de los módulos de comunicación).
- Cada CPU Twido: Compacta o Modular puede extenderse con otras como:
  - E/S descentralizadas, en este caso en las bases no pueden adicionarse módulos de extensión de E/S.
  - Twidos conectados como CPU´s: en este caso en las base pueden adicionarse módulos de extensión de E/S. Cada Twido tiene su propio programa de aplicación y tiene reservadas cuatro palabras de Entradas (%INW) y cuatro de Salida (%QNW) para intercambiar datos entre los Twidos.

  Hasta 7 Twidos pueden conectarse a un Twido Compacto o Modular. La distancia máxima del Bus RS485 es 200 m. Pueden utilizarse tanto los puertos integrados como los opcionales.
- Twido comunicado en Modbus. Puede integrarse fácilmente a los equipos existentes en campo como ser: otros autómatas programables, variadores de velocidad, monitores de circuito, arrancadores suaves, etc.

Más posibilidades de ajuste de parámetros

El visualizador de 4 botones puede ser utilizado para realizar los ajustes básicos directamente sobre el controlador.

Más simplicidad para ganar tiempo y disponibilidad

- **Fácil de cablear:** Twido le propone una gran variedad de conexiones:
  - soluciones con borneras a tornillo (extraíbles o fijas),
  - soluciones pre-cableadas para una conexión rápida y confiable (conectores HE10, Twido Fast),
  - soluciones de E/S remotas u otras CPU´s remotas (hasta 50 m),
  - Nuevas borneras a resorte, asociando un cableado rápido y una conexión segura.
- **Fácil de ensamblar**, con un simple click, podrá agregar las extensiones y/o los opcionales que necesite.
- **Fácil de instalar**, su pequeño tamaño facilita la integración en los tableros.
- **Fácil de aprender.**

Mayor capacidad

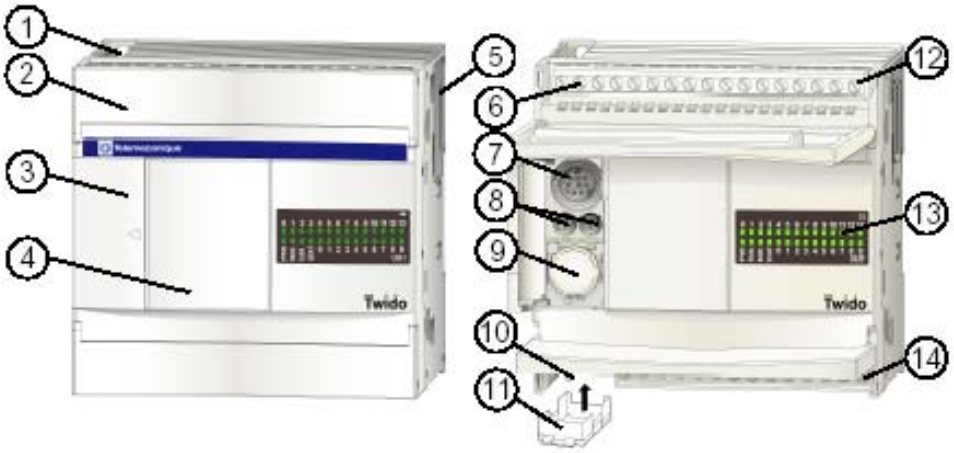
- **Con el opcional**: Reloj calendario.
- **Con memoria suplementaria de 32 y 64 k**, permitiendo una rápida puesta en marcha a distancia de su aplicación.
- **Con las siguientes funciones integradas:**
  - Contadores rápidos (5 y 20 kHz)
  - Posicionamiento con funciones PLS (generador de pulsos) y PWM (modulación de ancho de pulso) en los Twido Modulares (2 salidas configurables).
  - 1E analógica integrada, en tensión (0-10VCC) en todas las CPU´s Twido Modular.
  - Potenciómetros analógicos.

Mayor compatibilidad para garantizar funcionamiento sin costos extra.
Twido: sinergia total con los productos Schneider Electric.

El más pequeño y poderoso entre sus pares.
Imagine un Autómata Programable de 40 E/S y numerosas funciones integradas, todo en un tamaño no mayor a una tarjeta personal. **Twido**, supera todo lo imaginado.

2.3 Descripción.

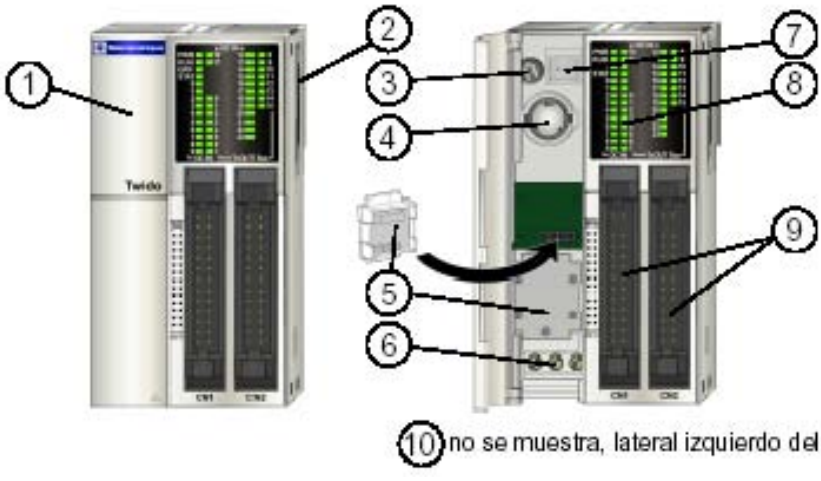
Twido compacto.



Referencias:

1. Orificio de montaje.
2. Cubierta de terminales.
3. Tapa con bisagra.
4. Cubierta extraíble del conector de visualización del operador.
5. Conector de ampliación - sólo en el controlador TWDLCAA24DRF.
6. Terminales de alimentación de sensores.
7. Puerto serie 1.
8. Potenciómetros analógicos - TWDLCAA10DRF y TWDLCAA16DRF tienen uno.
9. Conector de puerto serie 2 - TWDLCAA10DRF no tiene ninguno.
10. Terminales de fuentes de alimentación de 100 a 240 V CA.
11. Conector de cartuchos - ubicado en la parte inferior del controlador.
12. Terminales de entrada.
13. LED.
14. Terminales de salida.

Twido Modular.



Referencias.

1. Tapa con bisagra.
2. Conector de ampliación.
3. Potenciómetro analógico.
4. Puerto serie 1.
5. Cubiertas de los cartuchos.
6. Terminales de fuente de alimentación de 24 V CC.
7. Conector de entrada de tensión analógica.
8. LED.
9. Terminales de E/S.
10. Conector de comunicaciones.

2.4. Referencias de productos.

Descripción de las referencias y sus características a partir del código.

TWDL	TIPO	A	Cant. E/S	Caract. E/S

Tipo

☑ CA: Modelo compacto, alimentación en 100/240 Vca.
☑ MD: Modelo modular, alimentación en 24 Vcc.

Cantidad de Entradas / Salidas

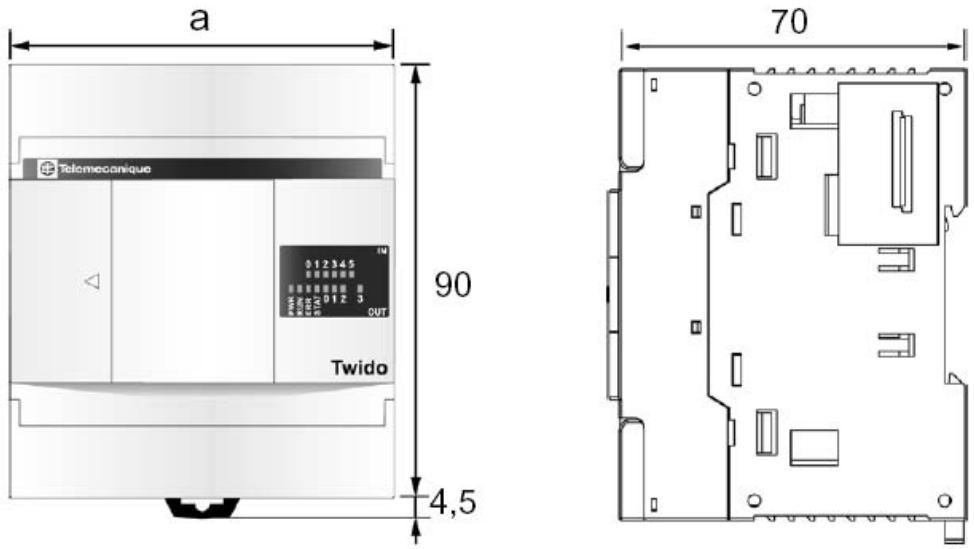
☑ 10: 6 entradas + 4 salidas.
☑ 16: 9 entradas + 7 salidas.
☑ 20: 12 entradas + 8 salidas.
☑ 24: 14 entradas + 10 salidas.
☑ 40: 24 entradas + 16 salidas.

Características de Entradas / Salidas

☑ Dxx: Entradas 24 Vcc NPN/PNP
☑ DFR: Salidas a Relé.
☑ DUK: Salidas a transistor NPN
☑ DTK: Salidas a transistor PNP
☑ DRT: Salidas a relé + salidas a transistor PNP

2.5 Dimensiones.

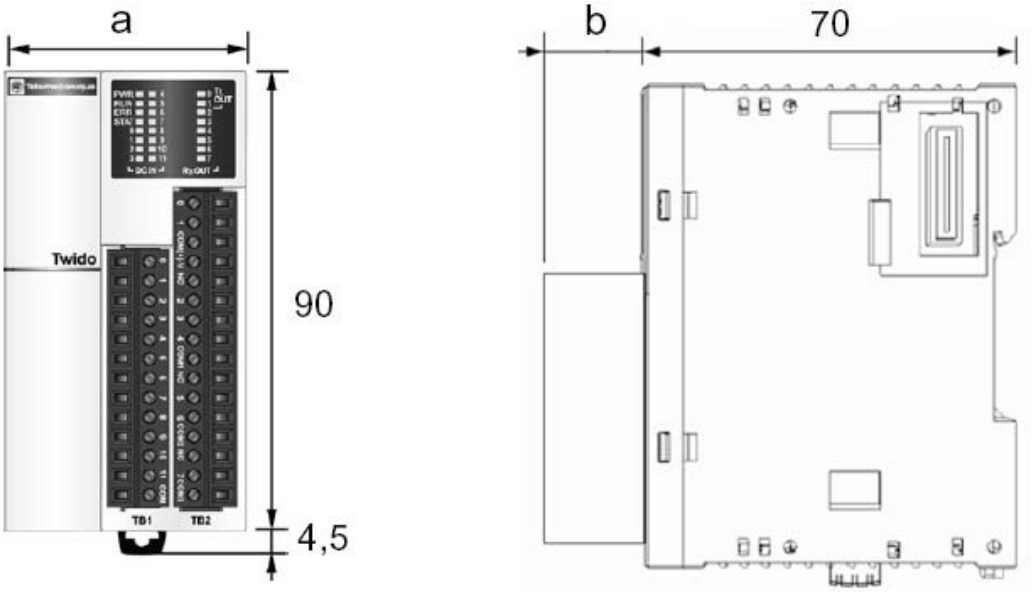
2.5.1 Modelos compactos



	a
TWDLCAA 10DRF	80
TWDLCAA 16DRF	80
TWDLCAA 24DRF	95

Nota: dimensiones en milímetros

2.5.2 Modelos Modulares



	a	b
TWDLMDA 20DTK/DUK	35,4	0 *
TWDLMDA 20DRT	47,5	14,6
TWDLMDA 40DTK/DUK	47,5	0 *

Nota: Dimensiones en milímetros
* Sin el conector

2.6 Conexionado.

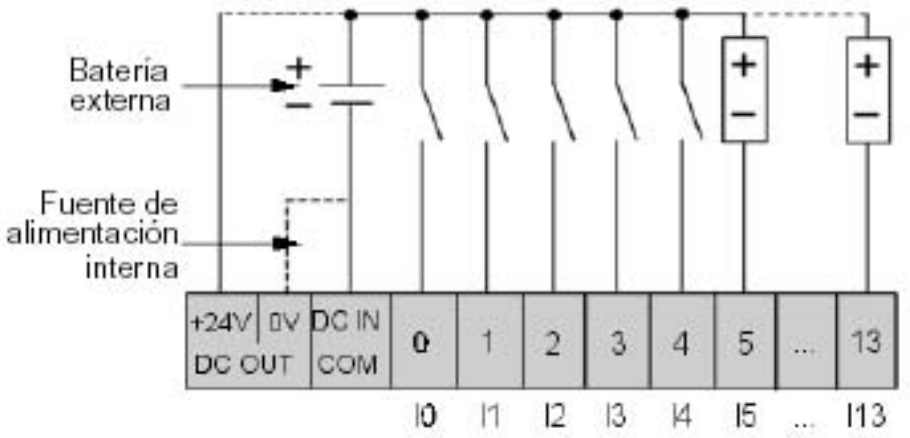
En esta sección se muestra un resumen del conexionado de las entradas y salidas digitales del autómata Twido, para mayor información sobre conexionado de los distintos módulos (E/S digitales ó analógicas, módulos de comunicación, etc.) remitirse a la guía de referencia de Hardware **TWD USE 10AS**.

2.6.1 Conexionado de las entradas digitales.

A continuación se descrie la forma de conexión de las entradas del TWIDO

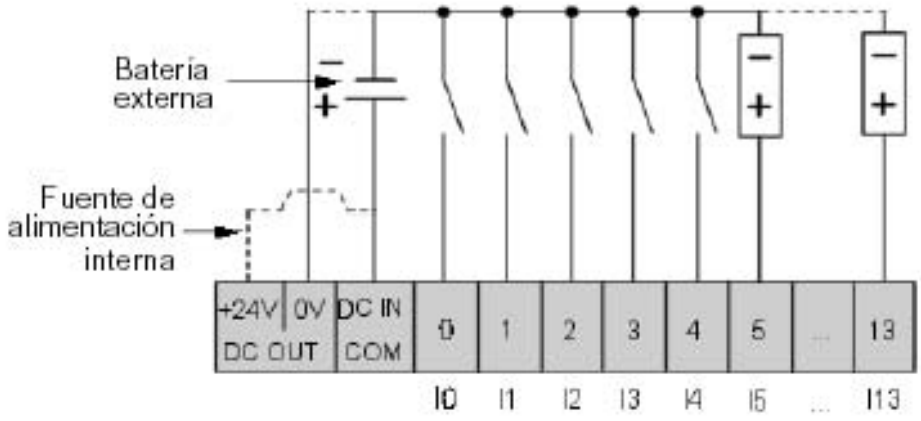
2.6.1.1 Entradas con lógica positiva.

Conexión de detectores **PNP.**



2.6.1.2 Entradas con lógica negativa.

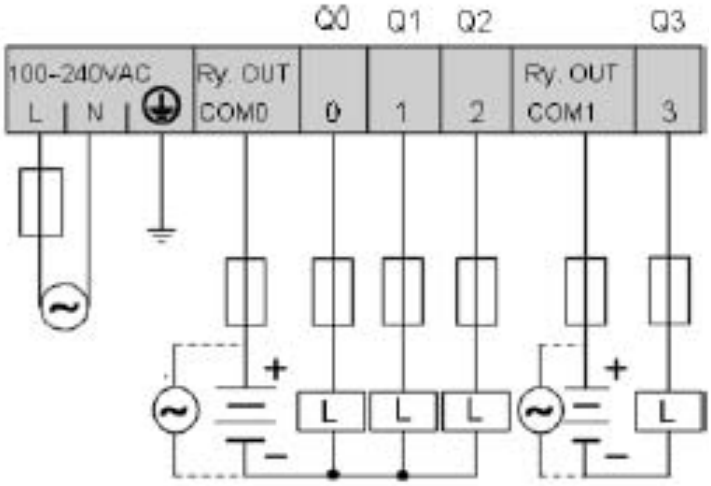
Conexión de detectores **NPN.**



2.6.2 Conexionado de las salidas digitales.

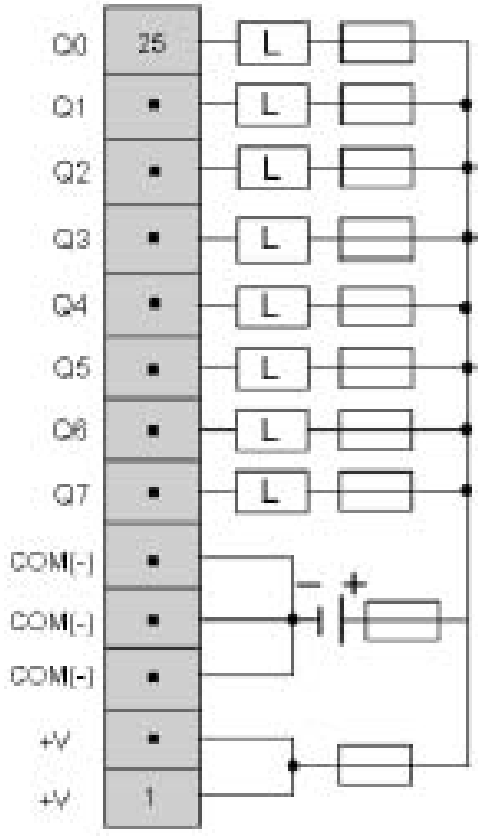
A continuación se descrie la forma de conexión de las salidas del TWIDO

2.6.2.1 Salidas con relé.

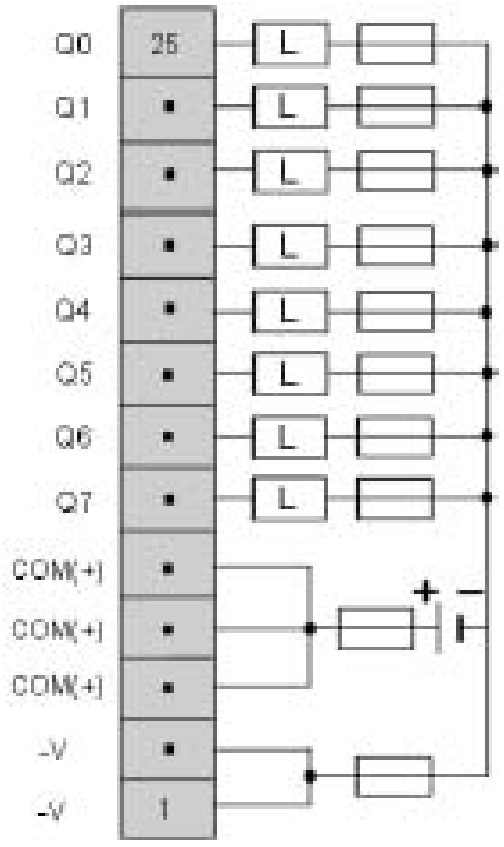


2.6.2.2 Salidas a transistor.

Con lógica negativa



Con lógica Positiva



2.7 Estructura de la memoria de usuario.

2.7.1 Introducción.

La memoria del controlador accesible a través de una aplicación de usuario está dividida en dos partes diferentes:

- Valores de bit
- Valores de palabra (valores con signo de 16 bits)

Memoria de bits La memoria de bits se almacena en la memoria RAM interna que está integrada en el controlador. Contiene el mapa de 1280 objetos de bit.

Función de la memoria de palabras
La memoria de palabras (16 bits) admite:

- **Datos**: datos de sistema y datos de aplicación dinámicos.
- **Programa**: descriptores y código ejecutable para tareas.
- **Constantes**: palabras constantes, valores iniciales y configuración de entrada/salida.

2.7.2 Tipos de memoria

A continuación se enumeran los distintos tipos de memoria para los controladores Twido.

- RAM interna (integrada)

  Esta es la memoria RAM integrada del controlador. Los 10 primeros KB de la memoria RAM interna constituyen la RAM rápida. Los 32 KB siguientes constituyen la RAM estándar. La RAM interna contiene el programa, constantes y datos.

- EEPROM interna

  EEPROM integrada de 32 KB que proporciona una copia de seguridad interna en el controlador de una aplicación. Protege la aplicación contra los daños provocados por fallos de batería o cortes de corriente superiores a 30 días. Contiene el programa y constantes.

- Cartucho de copia de seguridad de memoria externa

  Cartucho de EEPROM externa opcional para realizar copias de seguridad de una aplicación o para dar cabida a una aplicación más grande. Se puede utilizar para actualizar la aplicación en la RAM del controlador. Contiene el programa y constantes, pero ningún dato.

2.7.2.1 Estructura sin cartucho de memoria externa.

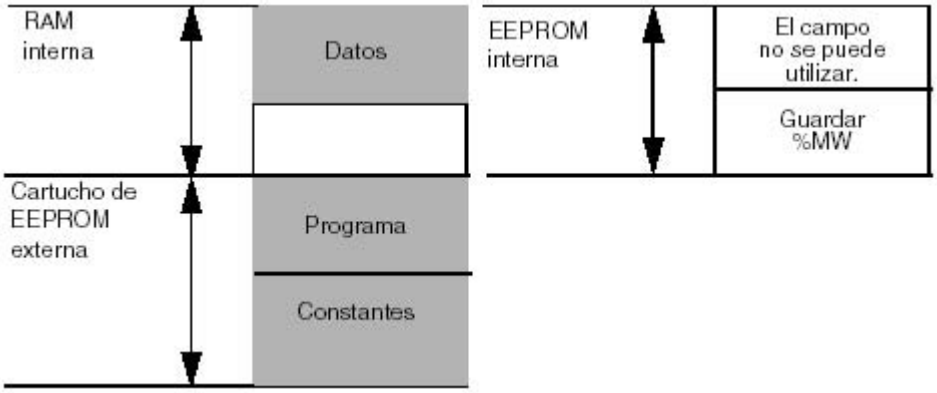
En el diagrama que aparece a continuación se describe la estructura de memoria sin cartucho de memoria externa.

La EEPROM interna está integrada en el controlador y proporciona 32 KB de memoria para lo siguiente:

- El programa de aplicación (32 KB)
- 512 palabras internas (%MWi)

2.7.2.2 Estructura con cartucho de memoria externa.

El cartucho de memoria externa opcional proporciona una copia de seguridad de los programas y constantes, al mismo tiempo que ofrece memoria ampliada para aplicaciones de mayor tamaño.
En el diagrama siguiente se describe la estructura de memoria con cartucho de memoria externa.



La EEPROM interna de 32 KB puede almacenar 512 palabras internas (%MWi).

2.7.3 Almacenamiento de la memoria

La memoria RAM interna del controlador se puede almacenar mediante:

- Batería interna (hasta 30 días)
- EEPROM interna (32 KB como máximo)
- Cartucho de memoria externa opcional (64 KB como máximo)

La transferencia de la aplicación desde la memoria EEPROM interna hasta la memoria RAM se realiza automáticamente cuando la aplicación se pierde en la RAM (si no se ha guardado o si no hay batería). También se puede realizar una transferencia manual mediante TwidoSoft.

3.- MODOS DE FUNCIONAMIENTO DEL CONTROLADOR.

3.1 Ciclo autómata.

El ciclo de ejecución del autómata se puede realizar de dos maneras:

- ☑Ejecución Normal (cíclica), configurada por defecto.
- ☑Ejecución periódica.

Cualquiera sea el modo de exploración elegido, el autómata realiza los siguientes pasos:

- ☑Tratamiento interno.
- ☑Lectura de las entradas.
- ☑Tratamiento del programa.
- ☑Actualización de las salidas.

Las funciones que realiza el autómata en cada uno de estos pasos son:

Tratamiento interno:

El sistema asegura implícitamente:
☑El sistema supervisa el controlador de forma implícita (gestionando las palabras y los bits de sistema, actualizando los valores de temporizador actuales, actualizando las luces de estado, detectando los cambios entre ejecución / detención, etc.) y procesa las solicitudes de TwidoSoft (modificaciones y animación).

Lectura de las entradas:

☑ Se escribe en la memoria el estado de la información relativa a las entradas binarias (%I) y del módulo específico de la aplicación asociados a la tarea.

Tratamiento del programa:

☑ Ejecución del programa de aplicación escrito por el usuario.

Actualización de las salidas:

☑ Se escriben los bits de salida (%Q)o las palabras asociadas a los módulos discretos específicos de la aplicación asociados a la tarea según el estado definido por el programa de aplicación.

Ciclo de funcionamiento

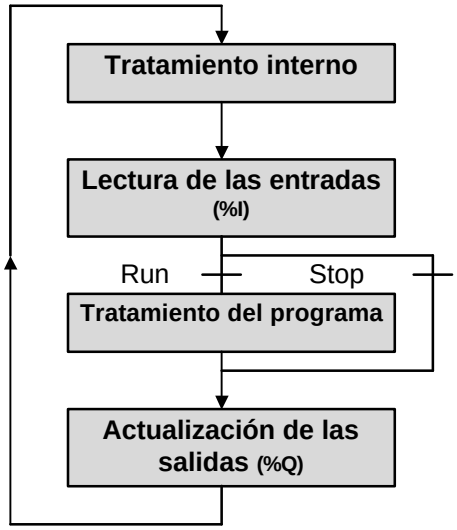
Existen dos posibilidades en cuanto al ciclo de ejecución, que el autómata este RUN o STOP. En cada uno de estos casos el autómata se comporta de la siguiente manera:

☑Autómata en **RUN**: El procesador ejecuta el tratamiento interno, la confirmación de entradas, el tratamiento del programa y la actualización de las salidas.

☑Autómata en **STOP**: En este caso no se ejecuta el tratamiento del programa.

3.1.1 Ejecución Normal (cíclica).

Por defecto, el ciclo autómata se ejecuta en forma cíclica de la siguiente manera:



Terminado el ciclo de ejecución actual, el autómata comienza inmediatamente con uno nuevo.

Desbordamiento del tiempo de ejecución

El temporizador watchdog del controlador supervisa el tiempo de ciclo del programa del usuario. Éste no debe exceder los 150 ms, ya que de lo contrario se producirá un fallo que provoque la detención inmediata del controlador en modo de parada. Las salidas en este modo se fuerzan a su estado de retorno predeterminado.

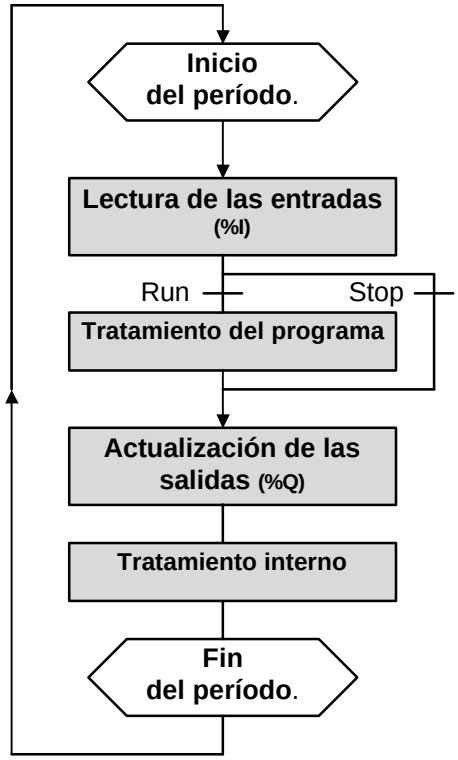
3.1.1.1 Casos posibles de funcionamiento:

☑Tiempo de ciclo $\leq$ watch dog: Funcionamiento normal, una vez finalizado el ciclo, se inicia el siguiente.

☑Tiempo de ciclo $\geq$ watch dog: El autómata pasa a STOP, los indicadores RUN y ERR parpadean y el bit de sistema %S11 pasa a 1.

3.1.2 Ejecución periódica.

En este caso, la lectura de las entradas, el tratamiento del programa y la actualización de las salidas se realiza de forma periódica según un tiempo definido por el usuario durante la configuración (2 a 150ms), tal como se indica en la figura siguiente:



En el inicio del ciclo del autómata, un temporizador de programa se ajusta al valor definido en configuración. El ciclo del autómata debe finalizar antes de que expire este temporizador. Al final del ciclo del temporizador, se inicia el siguiente.

Si el tiempo del ciclo supera al tiempo programado, el bit de sistema (%S19) pasará a 1. La comprobación y reinicialización a 0 correrán a cargo del programa usuario.

Desbordamiento del tiempo de ejecución

La duración del tiempo de ejecución del programa usuario es controlada por el autómata (watch dog) y no debe superar los 150ms En caso contrario, aparecerá un fallo que provocará la parada inmediata del autómata (indicadores RUN y ERR intermitentes)

3.1.2.1 Casos posibles de funcionamiento:

☑Tiempo de ciclo ≤ período: Funcionamiento normal, el ciclo siguiente se inicia una vez alcanzado el final del período programado.

☑Período ≤ tiempo de ciclo ≤ watch dog: El sistema pone el bit de sistema %S19 en estado 1 y el ajuste al estado 0 depende del programa usuario. El autómata permanece en RUN.

☑Tiempo de ciclo ≥ watch dog: El autómata pasa a STOP, los indicadores RUN y ERR parpadean y el bit de sistema %S11 pasa a 1.

3.2 Comprobación del tiempo de ciclo.

3.2.1 Generalidades.

El ciclo de tarea master se controla mediante un temporizador watchdog llamado Tmax (duración máxima del ciclo de tarea master). Permite mostrar errores de aplicación (bucles infinitos, etc.) y garantiza una duración máxima para actualizar las salidas.

3.2.2 WatchDog del software (operación periódica o cíclica).

En una operación periódica o cíclica, la activación del watchdog provoca un error del software. La aplicación pasa a estado de pausa y establece el bit %S11 a 1. La nueva ejecución de la tarea necesita una conexión a Twido Soft con el fin de analizar la causa del error, la modificación de la aplicación para corregir el error y la nueva ejecución de las solicitudes de inicio y ejecución.

Nota: El estado de pausa se produce cuando la aplicación se detiene inmediatamente debido a un error del software de la aplicación, como un desborde de ciclo. Los datos conservan los valores actuales que permiten un análisis de la causa del error. Todas las tareas se detienen en la instrucción actual. Está disponible la comunicación con el controlador.

3.2.3 Comprobación de la operación periódica.

En una operación periódica, se utiliza una comprobación adicional para detectar el periodo que se está excediendo.

- **%S19** indica que se ha superado el periodo. Se establece a:
  - 1 por el sistema cuando el tiempo de ciclo es mayor que el periodo de la tarea.
  - 0 por el usuario.
- **%SW0** contiene el valor del periodo (0-150 ms). Es decir:
  - Se inicializa cuando se inicia a partir de un inicio en frío mediante el valor establecido en la configuración.
  - El usuario puede modificarlo

3.2.4 Uso del tiempo de ejecución de la tarea master.

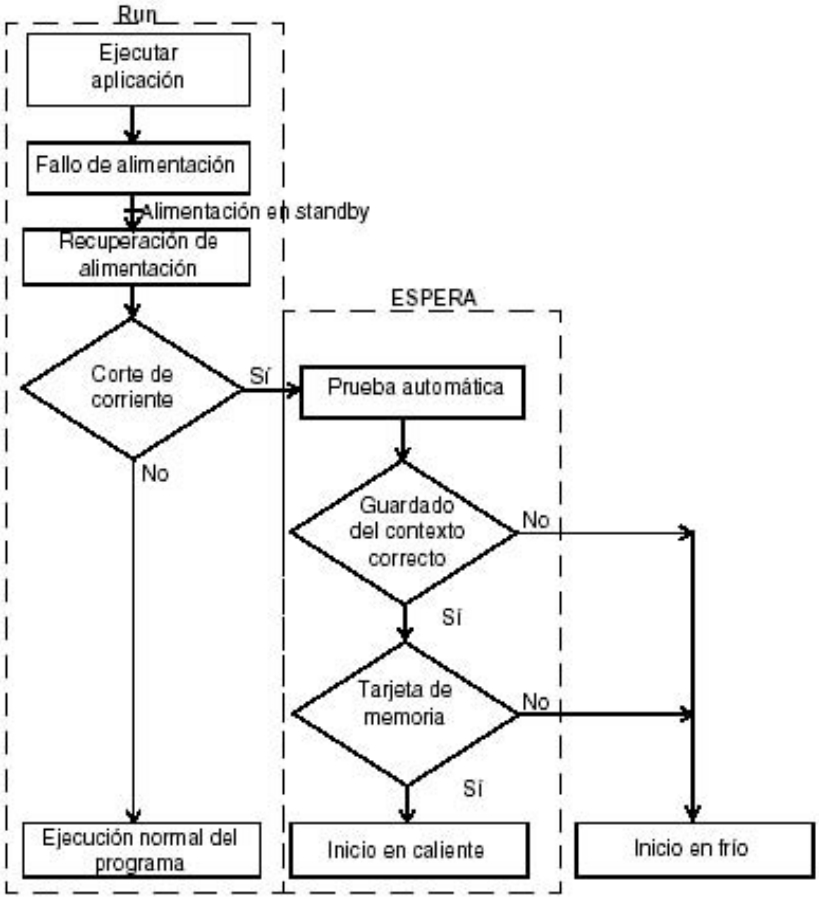
Las siguientes palabras del sistema se utilizan para ofrecer información sobre el tiempo de ciclo de exploración del controlador:

- **%SW11** Se inicializa con el tiempo de vigilancia máximo de watchdog (10 a 500 ms).
- **%SW30** contiene el tiempo de ejecución para el último ciclo de exploración del controlador.
- **%SW31** contiene el tiempo de ejecución para el ciclo de exploración del controlador más largo.

- **%SW32** contiene el tiempo de ejecución para el ciclo de exploración del controlador más corto.

3.3 Comportamiento ante cortes de corriente y recuperación de alimentación.

La ilustración que aparece a continuación muestra los distintos tipos de reinicio de alimentación detectados por el sistema. Si la duración del corte de corriente es inferior al tiempo de filtrado de suministro de alimentación (unos 10 ms para el suministro de corriente alterna o 1 ms para el suministro de corriente continua), el programa no lo advierte y sigue funcionando con normalidad.

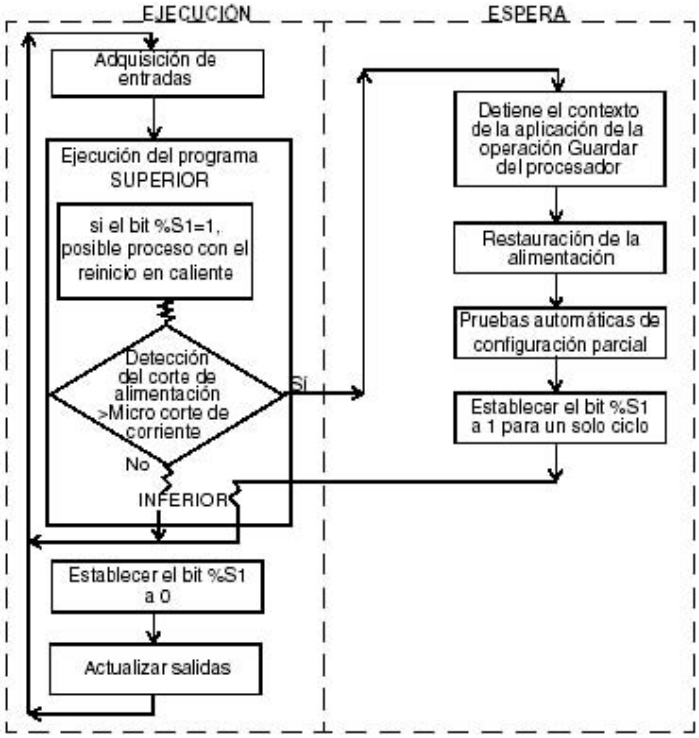


3.3.1 Comportamiento ante un inicio en caliente.

Causa de un reinicio en caliente.
Un inicio en caliente puede producirse:

- Cuando se restaura la alimentación sin pérdida de contexto de las aplicaciones.
- Cuando el programa establece el bit **%S1** a estado.
- Desde la visualización del operador, cuando el controlador está en modo de detención.

El dibujo que aparece a continuación describe una operación de reinicio en caliente, en modo de ejecución.



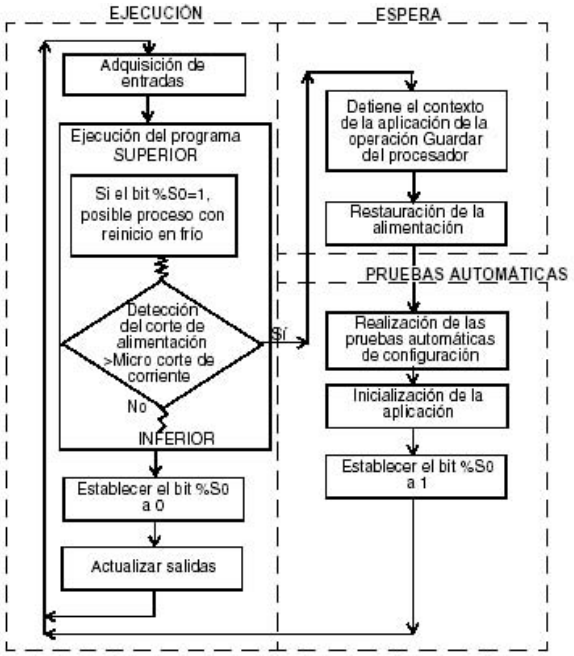
3.3.2 Comportamiento ante un inicio en frío.

Causas de un inicio en frío

Un inicio en frío puede producirse:

- Al cargar una aplicación nueva en la RAM.
- Cuando se restaura la alimentación con pérdida de contexto de las aplicaciones.
- Cuando el programa ajusta el bit **%S0** a estado 1.
- Desde el monitor de operación, cuando el controlador está en modo de detención.

El dibujo de abajo describe una operación de reinicio en frío en modo de ejecución.



4.- LENGUAJES DE PROGRAMACIÓN.

4.1 Introducción al TwidoSoft.

El desarrollo de una aplicación destinada al autómata TWIDO debe realizarse mediante el software de desarrollo TwidoSoft.
TwidoSoft es un entorno de desarrollo gráfico para crear, configurar y mantener aplicaciones para controladores programables Twido. TwidoSoft permite introducir programas de control utilizando los editores de programa de lista o Ladder Logic y, a continuación, transferir el programa para ejecutarlo en un controlador.

TwidoSoft es un programa basado en Windows de 32 bits para PC, que se ejecute bajo los sistemas operativos Microsoft Windows 98 segunda edición o Microsoft Windows 2000 Profesional.
Las principales funciones del software TwidoSoft son:

- Interfase de usuario estándar de Windows
- Programar y configurar controladores Twido
- Control y comunicaciones del controlador

4.2 Lenguajes de programación de Twido

Para crear programas de control Twido se pueden utilizar los siguientes lenguajes de programación:

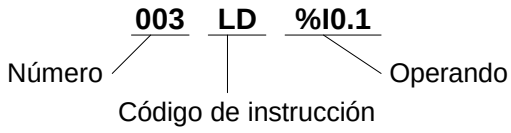
- Lenguaje de lista de instrucciones
  Un programa de lista de instrucciones se compone de una serie de expresiones lógicas escritas como una secuencia de instrucciones booleanas.
- Diagramas Ladder Logic
  Un diagrama Ladder Logic es una forma gráfica de mostrar una expresión lógica.
- Grafcet
  Twido admite las instrucciones de lista Grafcet, pero no Grafcet gráfico.

TwidoSoft ofrece una función de reversibilidad de Lista/Ladder Logic, la cual permite pasar un programa de Lista a Ladder Logic y viceversa, según convenga.

4.2.1 Lenguaje Lista de instrucciones (Lista o IL)

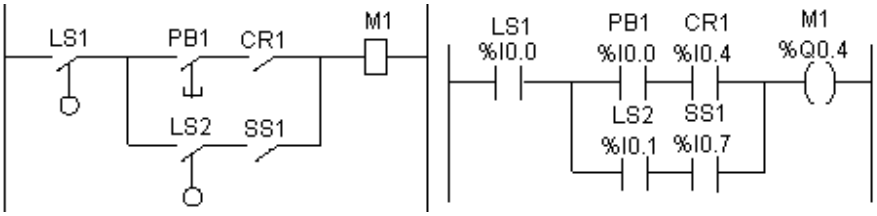
Un programa en lenguaje Lista consta de una serie de instrucciones de diversos tipos. Cada fila de programa tiene un número generado en forma automática, un código de instrucción y un operando tipo bit o palabra.

Ejemplo de instrucción:



4.2.2 Lenguaje de Contactos (Ladder o LD).

Un programa escrito en lenguaje de contactos se compone de una serie de circuitos ejecutados secuencialmente por el autómata. La representación de un circuito se asemeja a la de un esquema eléctrico de relés. Elementos gráficos de test simbolizan los contactos (botón pulsador, contactos fin de recorrido, etc). así como elementos gráficos de acciones simbolizan las bobinas.

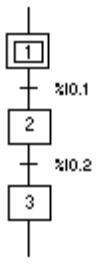


En la figura anterior se ilustra el esquema de cableado simplificado de un circuito de lógica de relés y su equivalente en esquema de contactos. Las referencias que aparecen encima de cada símbolo indican la ubicación de las conexiones tanto de entradas como de salidas.
Un circuito de contactos se compone de una serie de instrucciones gráficas específicas, relacionadas entre sí, y situadas entre las dos barras que representan el potencial.
Estas instrucciones gráficas se asocian entre sí mediante conexiones horizontales y verticales que conducen a una o varias salidas o acciones.
Un circuito no puede soportar mas de un grupo de instrucciones asociadas.

4.2.3 Grafcet.

Grafcet es un método de análisis que consiste en descomponer un automatismo secuencial en una sucesión de etapas, a las que se asocian acciones, transiciones y condiciones.
El TwidoSoft al no soportar el Grafcet gráfico, posee instrucciones específicas para programación en grafcet.

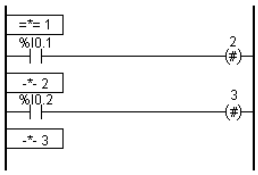
Grafcet gráfico



Grafcet Lista

```
=*=   01
LD    %I0.1
#     02
-*-   02
LD    %I0.2
#     03
-*-   03
```

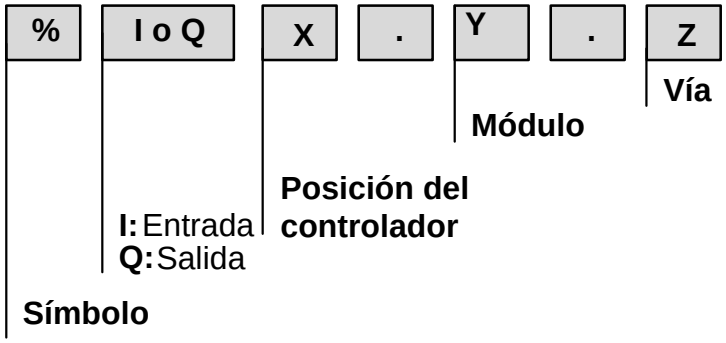
Grafcet Ladder



5.- TRATAMIENTO BOOLEANO.

5.1 Definición de los principales objetos de bits.

☑ Bits de entradas / salidas: Estos bits son las "imágenes lógicas" de los estados eléctricos de las entradas / salidas. Están almacenados en la memoria de datos y se actualizan en cada explotación del programa. El direccionamiento de estos bits es el siguiente:



Símbolo: IEC61131
Tipo de objeto: %I: Entradas, %Q: Salidas.
X. Posición del controlador: 0 Controlador master, 1 a 7 controlador remoto.
Y. Módulo: 0 unidad de E/S local, 1 a 7 módulos de ampliación.
Z. Vía, número de la entrada o salida.

☑ Bits internos: Los bits internos **(%Mi)** memorizan los estados intermedios durante la ejecución del programa.
☑ Bits de sistema: Los bits de sistema **(%Si)** controlan el buen funcionamiento del autómata así como el desarrollo del programa de aplicación. El detalle de los mismos se encuentra en el capítulo Bits y Palabras Sistema.
☑ Existen otros bits que pueden usarse en el tratamiento booleano, como son los bits de los bloques de función y los bits extraídos de palabras, los cuales explicaremos en los capítulos Programación de Bloques Función y Tratamiento Numérico respectivamente.

Resumen: La siguiente tabla agrupa a todos los bits:

Tipo	Dirección	Cantidad máxima	Escritura
Bits Entrada	**%IX.Y.Z**	**Depende Twido**	**No**
Salida	**%QX.Y.Z**	**Depende Twido**	**Sí**
Bits Internos	**%Mi**	**128 ó 256 según modelo**	**Sí**
Bits Sistema	**%Si**	**128**	**Según i**

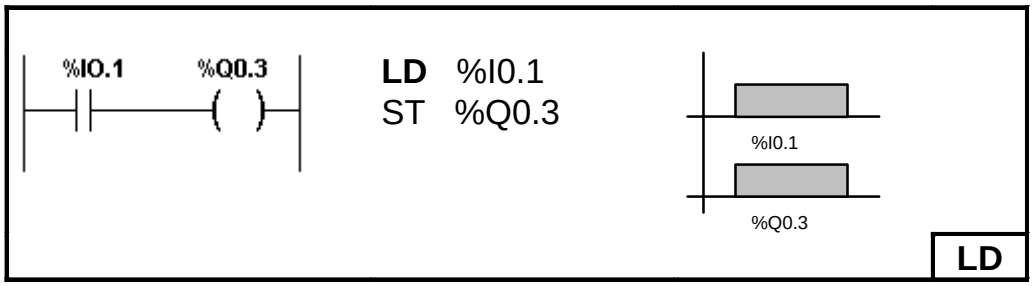
5.2 Descripción de instrucciones.

En el punto 5.1 se describe el direccionamiento completo de las entradas y salidas de un autómata Twido. Para mayor simplicidad en la lectura, en los ejemplos mostrados a continuación, no se tendrá en cuenta el parámetro **X** (posición del controlador), pues solo se aplica para entradas y salidas remotas. Esta opción se analizará en el capítulo referido a comunicaciones.

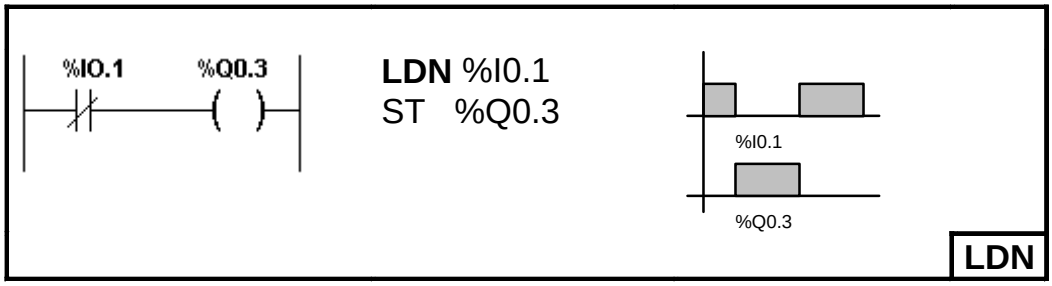
5.2.1 Instrucciones de carga LD, LDN, LDR y LDF.

Las instrucciones LD, LDN, LDR y LDF corresponden respectivamente a contactos normal abierto, normal cerrado, de flanco ascendente y descendente.

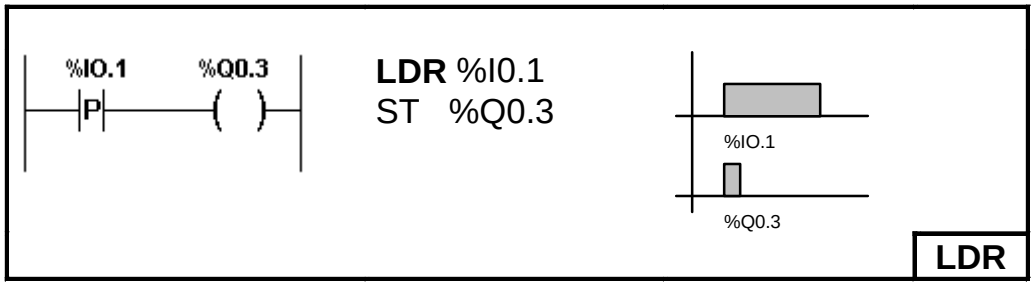
5.2.1.1 Contacto normal abierto



5.2.1.2 Contacto normal cerrado.

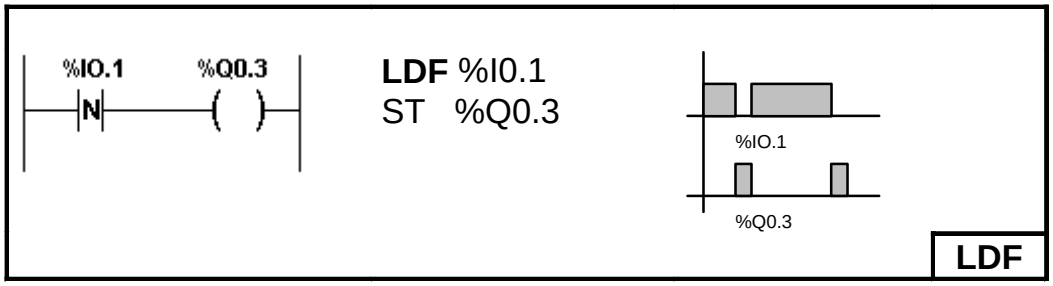


5.2.1.3 Contacto flanco ascendente.



El tiempo que permanece activa la salida equivale a un ciclo del autómata.

5.2.1.4 Contacto flanco descendente.

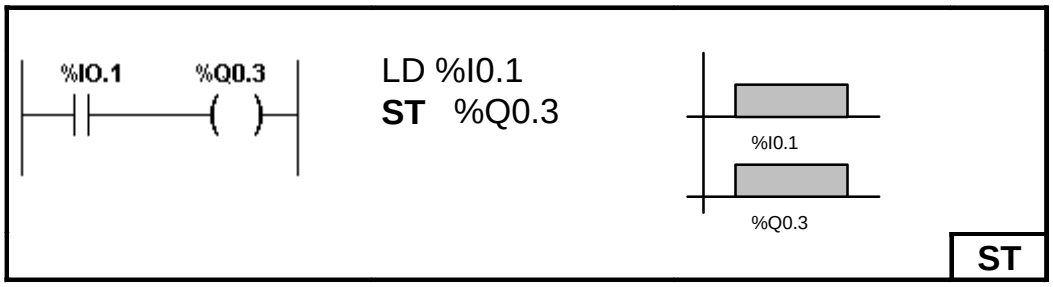


El tiempo que permanece activa la salida equivale a un ciclo del autómata.

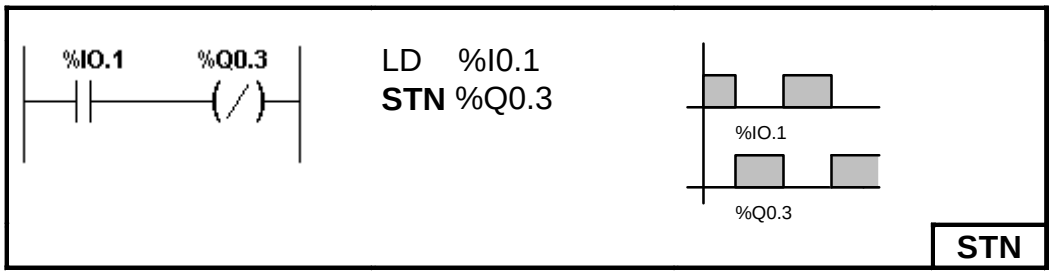
5.2.2 Instrucciones de asignación ST, STN, S y R.

Las instrucciones ST, STN, S y R corresponden respectivamente a: bobina directa, inversa, set y reset.

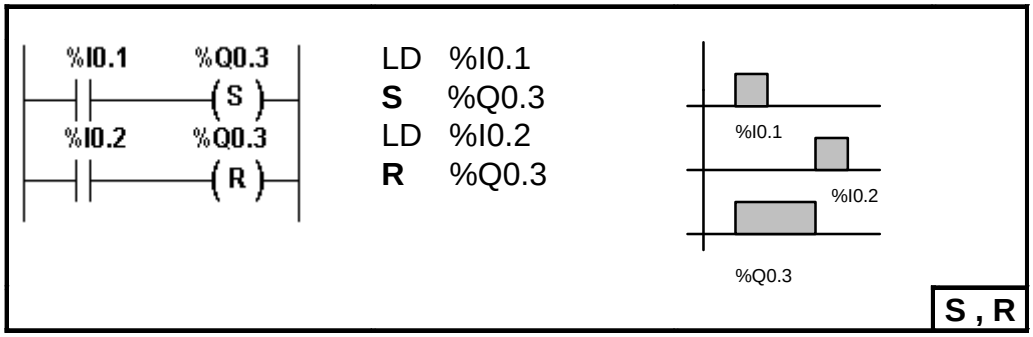
5.2.2.1 Bobina directa.



5.2.2.2 Bobina inversa.



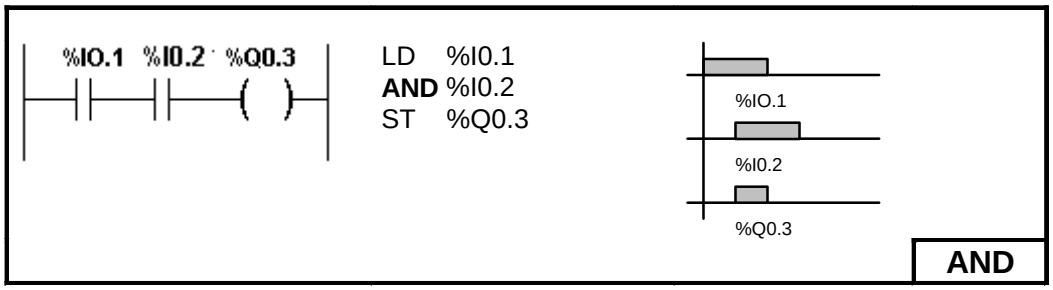
5.2.2.3.-Bobinas set y reset.



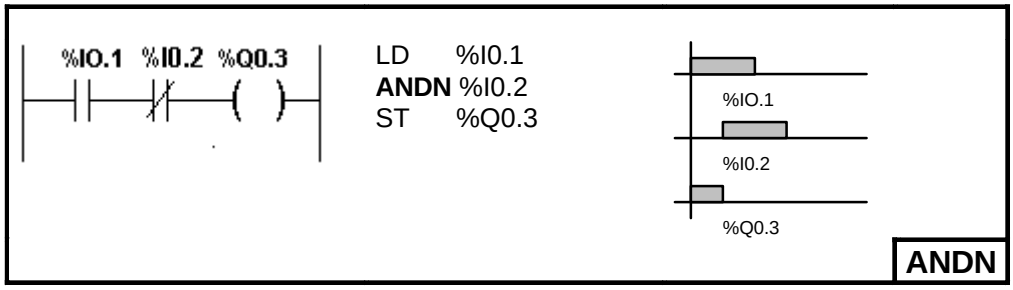
5.2.3 Instrucción lógica Y, AND, ANDN, ANDR y ANDF.

Estas instrucciones realizan el producto lógico o asociación en serie de contactos entre el operando (la inversa de un operando, un flanco ascendente o un flanco descendente) y el resultado booleano de la instrucción precedente.

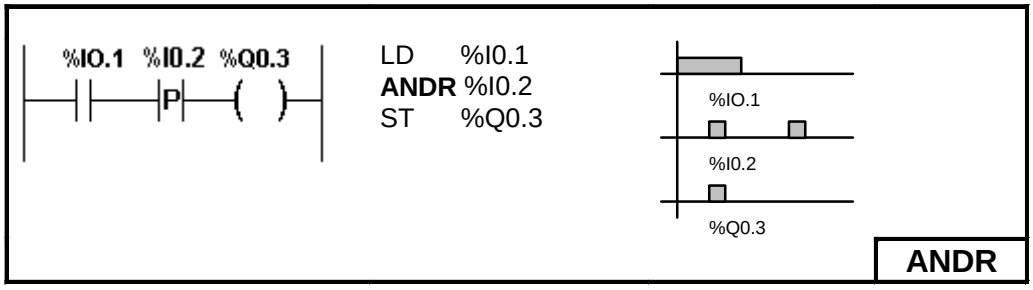
5.2.3.1 Producto lógico.



5.2.3.2 Producto lógico negado.

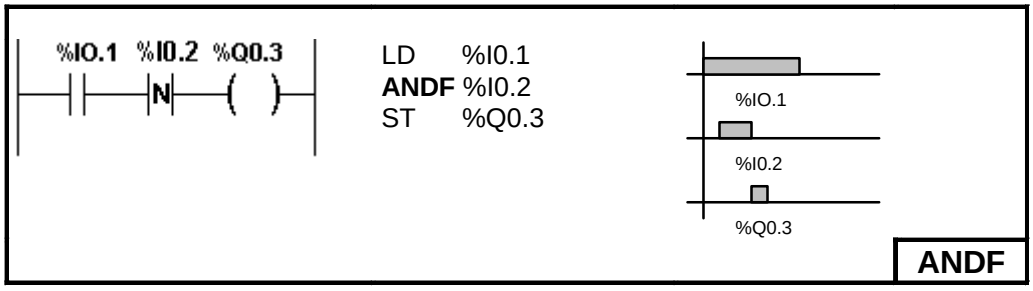


5.2.3.3 Producto lógico flanco ascendente.



El tiempo que permanece activa la salida equivale a un ciclo del autómata.

5.2.3.4 Producto lógico flanco descendente

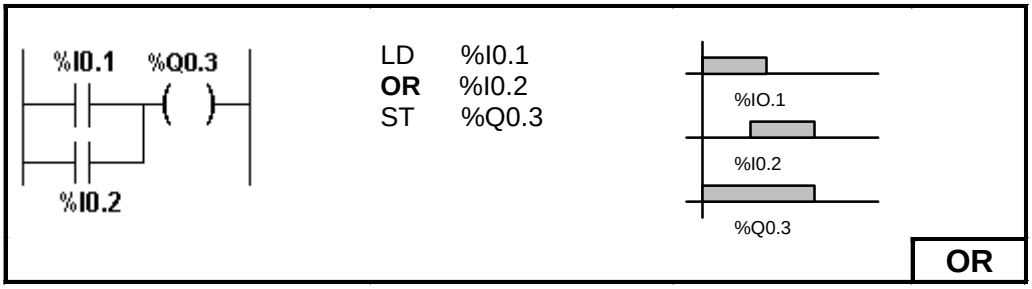


El tiempo que permanece activa la salida equivale a un ciclo del autómata.

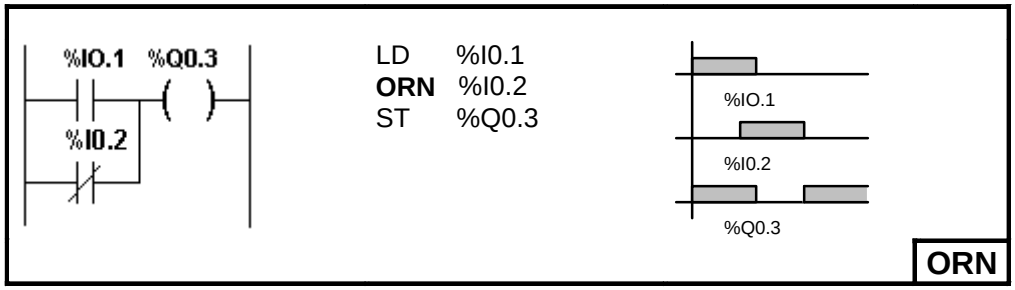
5.2.4 Instrucciones lógicas O, OR, ORN, ORR y ORF

Esta instrucción efectúa la suma lógica o asociación de contactos en paralelo entre el operando (la inversa del operando, un flanco ascendente o un flanco descendente) y el resultado booleano de la instrucción precedente.

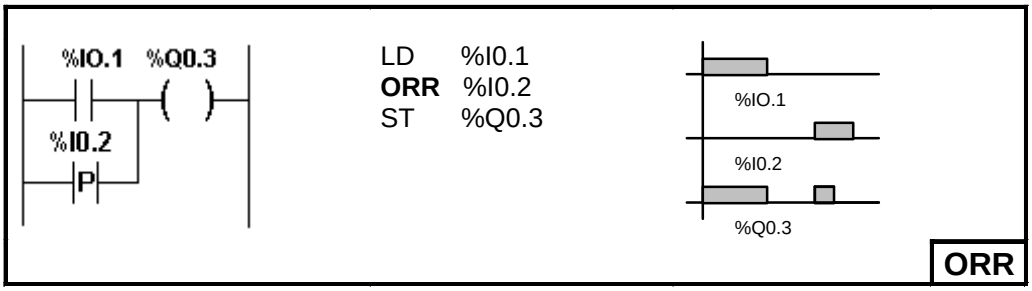
5.2.4.1 Suma Lógica



5.2.4.2 Suma Lógica Negada

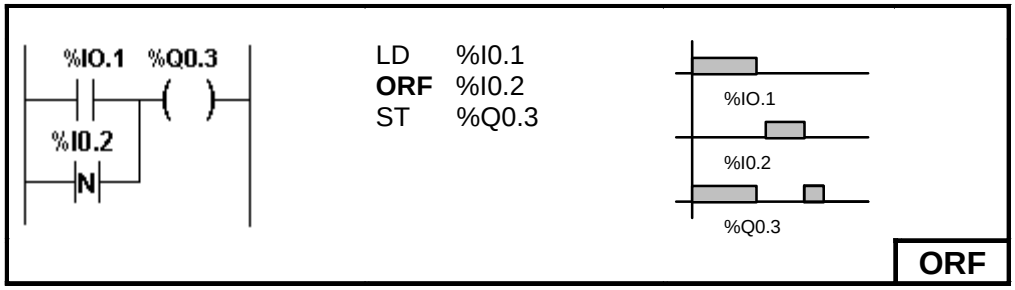


5.2.4.3 Suma lógica con flanco ascendente.



El tiempo que permanece activa la salida equivale a un ciclo del autómata.

5.2.4.4 Suma lógica con flanco descendente.



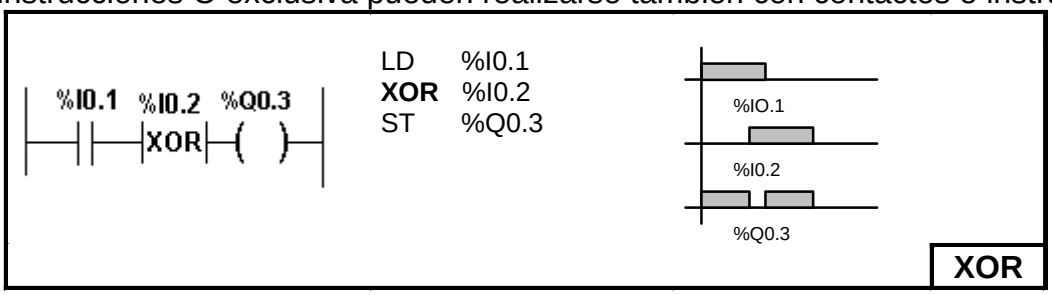
El tiempo que permanece activa la salida equivale a un ciclo del autómata.

5.2.5 Instrucción O exclusiva: XOR, XORN, XORR y XORF.

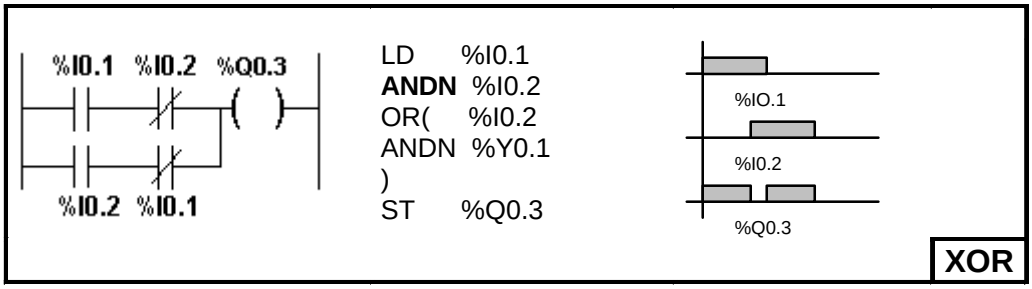
Estas instrucciones realizan un O exclusivo entre el operando (o su inverso, o flanco ascendente o descendente) y el resultado booleano de la instrucción anterior. Esta operación es conocida también como comparador de desigualdad, puesto que es resultado es 1 (ON), cuando los operandos involucrados son distintos.

5.2.5.1 Suma Lógica Exclusiva.

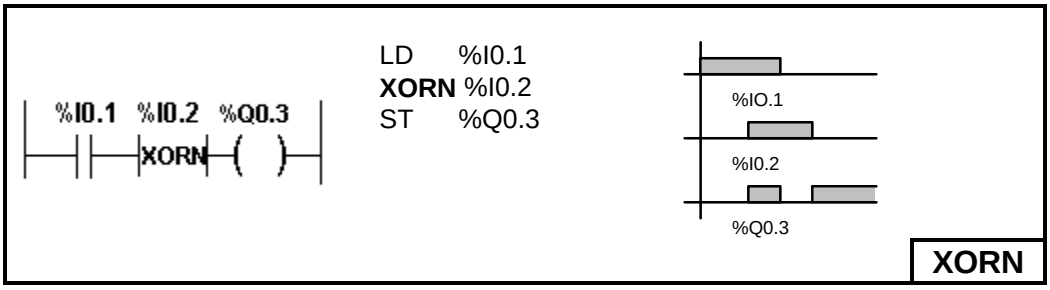
Las instrucciones O exclusiva pueden realizarse también con contactos e instrucciones



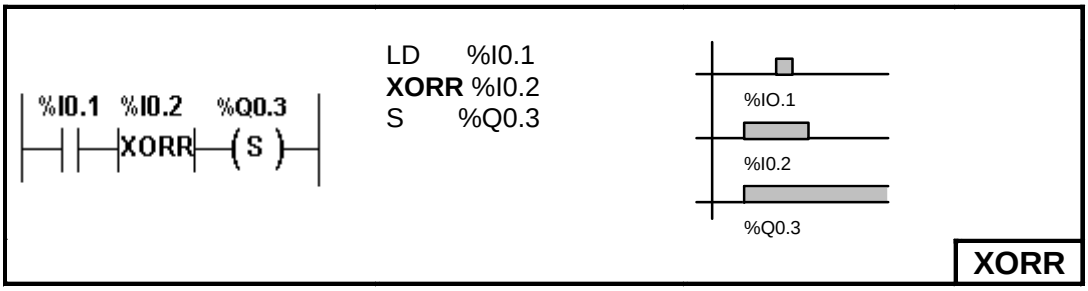
comunes, a continuación detallamos la forma de realizarlo para ilustrar la lógica de la instrucción.



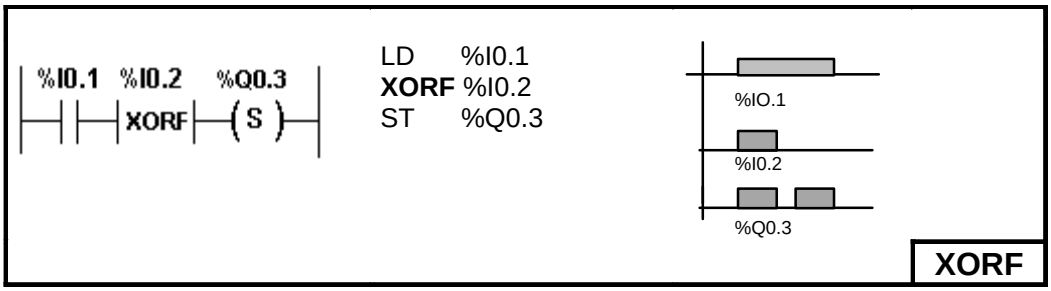
5.2.5.2 Suma Lógica Exclusiva Negada.



5.2.5.3 Suma Lógica Exclusiva Flanco Ascendente.



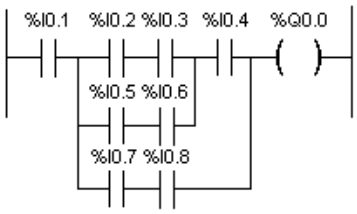
5.2.5.4 Suma Lógica Exclusiva Flanco Descendente.



5.3 OTRAS INSTRUCCIONES.

5.3.1 Utilización de paréntesis.

Las instrucciones AND y OR pueden utilizar paréntesis. Estos paréntesis permiten realizar esquemas de contactos de forma sencilla. El signo de apertura se asocia a la instrucción AND u OR. El paréntesis de cierre es una instrucción obligatoria para cada paréntesis abierta. Es posible anidar hasta 8 niveles de paréntesis. No se deben programar instrucciones de almacenamiento ST, STN, S o R entre paréntesis. Tampoco se pueden utilizar instrucciones de stack MPS, MRD o MPP.



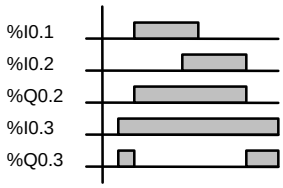

```
LD    %I0.1
AND(  %I0.2
AND   %I0.3
OR(   %I0.5
AND   %I0.6
)
AND   %I0.4
OR(   %I0.7
AND   %I0.8
)
)
ST    %Q0.0
```

( )

5.3.2 Instrucción NOT.

Esta instrucción realiza la negación del resultado booleano precedente.

```
LD
OR   %I0.
ST
N
AND
ST
```

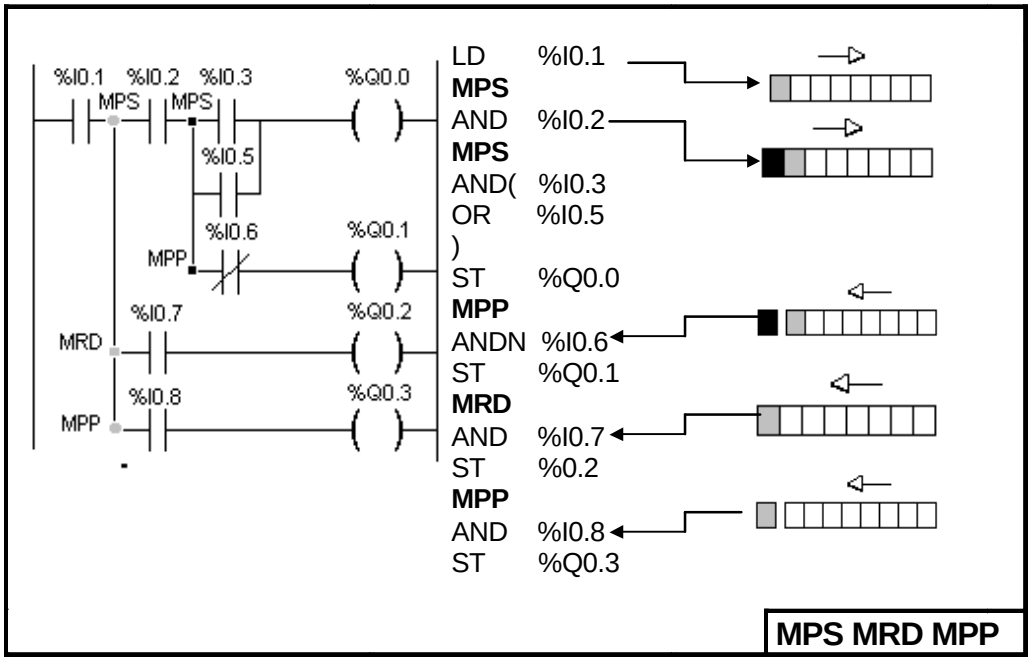


N

Esta instrucción no es reversible pues está disponible solo en el lenguaje list.

5.3.3 Instrucciones MPS, MRD y MPP.

Los tres tipos de instrucciones permiten tratar la derivación hacia las bobinas.
Estas instrucciones utilizan una memoria intermedia llamada pila que puede almacenar hasta 8 informaciones booleanas.
La instrucción MPS almacena el resultado de la última instrucción de comprobación en la parte superior de la pila y desplaza los otros valores hacia el fondo de la pila.
La instrucción MRD lee el inicio de la pila.
La instrucción MPP lee, desocupa el inicio de la pila y desplaza los otros valores hacia el inicio de la pila.


%I0.1
%I0.2
%I0.3
%Q0.0
MPS
MPS
%I0.5
%I0.6
%Q0.1
MPP
%I0.7
%Q0.2
MRD
%I0.8
%Q0.3
MPP
LD %I0.1
MPS
AND %I0.2
MPS
AND(%I0.3
OR %I0.5
)
ST %Q0.0
MPP
ANDN %I0.6
ST %Q0.1
MRD
AND %I0.7
ST %0.2
MPP
AND %I0.8
ST %Q0.3
MPS MRD MPP

6.- PROGRAMACIÓN DE BLOQUES FUNCIÓN.

6.1 Objetos bits y palabras asociadas a bloques de función.

Los bloques de función emplean objetos bits y palabras específicos.

☑Objetos bits: Corresponden a salidas de los bloques. Se accede a estos bits mediante las instrucciones booleanas de comprobación.

☑Objetos palabras: Corresponden a parámetros de configuración de bloque (bases de tiempo, valor preseleccionado, etc), y a valores actuales (valor de contaje en curso, etc).

Lista de objetos bits y palabras de bloques de función accesibles por programa:

Bloque	Palabras y bits asociados		Dirección	Escritura
Temporizador %TMi i = 0 a 127	Palabras	Valor actual	%TMi.V	no
		Valor de preselección	%TMi.P	si
	Bits	Salida temporizador	%TMi.Q	no
Contador / Descontador %Ci I=0 a 31	Palabras	Valor actual	%Ci.V	no
		Valor de preselección	%Ci.P	si
	Bits	Salida desbordam. (vacío)	%Ci.E	no
		Salida preselec. alcanzada	%CI.D	no
		Salida desbordam. (lleno)	%Ci.F	no
Registro %Ri i=0 a 3	Palabras	Acceso al registro	%Ri.I	si
		Salida del registro	%Ri.O	si
	Bits	Registro lleno	%Ri.F	no
		Registro vacío	%Ri.E	no

6.2 Principios de programación

Los bloques función pueden ser programados de dos formas diferentes:

☑ Con instrucciones de bloque función (Ej: BLK%TM2), que es la forma reversible en lenguaje de contactos. Autoriza a efectuar las operaciones sobre el bloque en un solo lugar del programa.

☑ Con instrucciones específicas (Ej: CU%Ci), que es la forma no reversible. Permite efectuar las operaciones sobre las entradas de los bloques en diversos lugares del programa (Ej: línea 100 CU %C1, línea 174 CD %C1, línea 209 LD %C1.D).
Nuestra recomendación es realizar una programación tal que permita la reversibilidad de lenguajes.

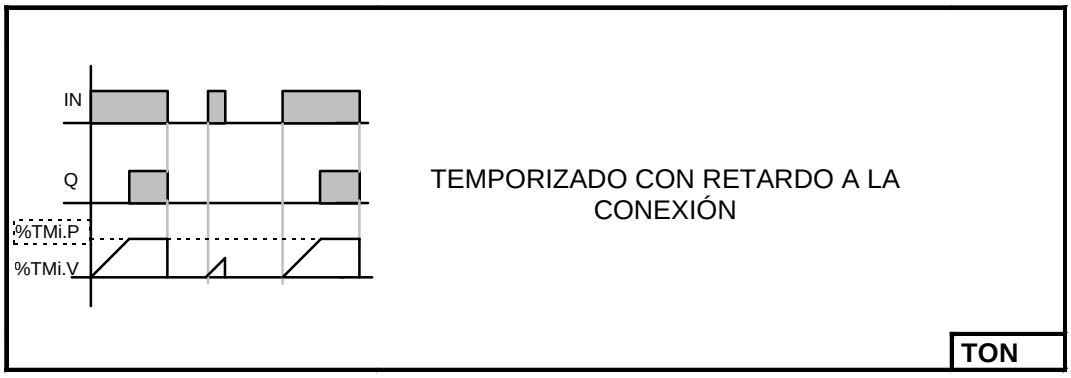
Principios de la programación reversible de los bloques función:
Este tipo de programación utiliza las instrucciones de bloque OUT y END_BLK.
BLK indica el inicio de un bloque función.
OUT_BLK opcional, permite «cablear» directamente las salidas del bloque.
END_BLK indica el fin del bloque.

6.3 Bloque de función temporizador.

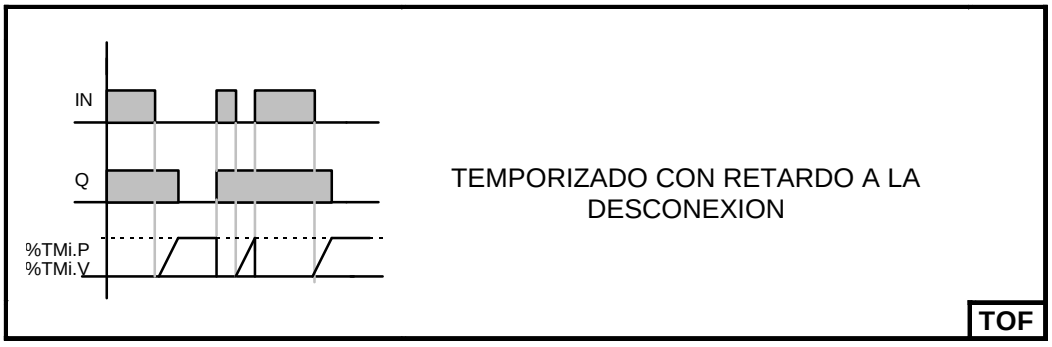
Cada uno de los temporizadores pueden configurarse de una de las tres formas propuestas por la normativa IEC61131. Los 3 tipos propuestos son:

1. **TON: Retardo a la conexión.**
2. **TOF: Retardo a la desconexión.**
3. **TP: Monoestable.**

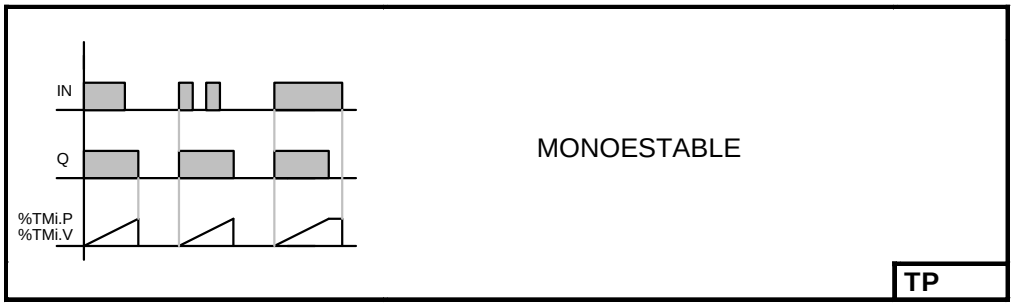
1) TON este tipo de temporizador permite generar retardos a la conexión. Dicho retardo es programable y puede ser modificado o no a través de la terminal.



2) TOF este tipo de temporizador permite generar retardos a la desconexión, dicho retardo es programable y puede ser modificado o no a través de la terminal.



3) TP este tipo de temporizador permite elaborar un pulso de duración precisa. Esta duración es programable y puede ser modificado o no a través de la terminal.



Características: (En **negrita** se detallan los valores por defecto)

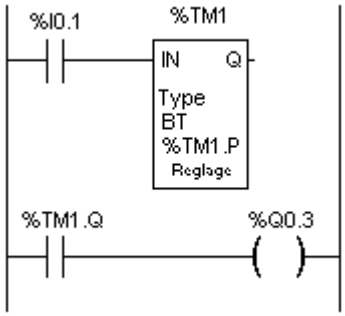
La selección del modo de funcionamiento del temporizador **(TON, TOF, TP)** se efectúa desde la configuración.

Número de temporizador	%TMi	0 a 127
Tipo	**TON** TOF TP	
Base de tiempo	BT	**1 mn**, 1 s, 100 ms, 10 ms y 1 ms. Cuanto menor es la base de tiempo, mayor es la precisión del temporizador
Valor actual	%TMi.V	Palabra que crece desde 0 hasta %TMi.P durante la temporización. El programa no puede escribirlo.
Valor de preselección	%TMi.P	0 ≤ %TMi.P ≤ **9999**. Puede ser leída y escrita por el programa.
Ajuste	S/N	**S:** permite la modificación de %TMi.P N: no accesible %TMi.P
Entrada de activación	IN	Arranca el temporizador
Salida Temporizador	Q	Bit asociado a la salida %TMi.Q

El tiempo t de temporización se calcula de la siguiente forma:

t = BT x %TMi.P

Ejemplo de programación.




```
BLK   %TM1
LD    %I0.1
IN
END_BLK
LD    %TM1.Q
ST    %Q0.3
```

%TMi

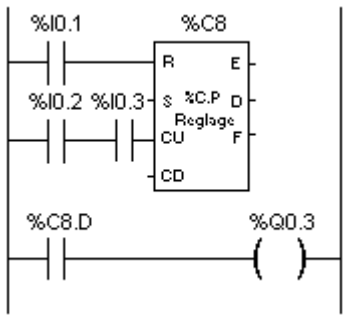
6.4 Bloque de función contador.

El bloque función contador permite efectuar un conteo en forma ascendente y descendente, a dicho bloque se lo designa %Ci

Características: (En **negrita** se detallan los valores por defecto).

Número de contador	%Ci	0 a 31
Valor actual	%Ci.V	Palabra aumentada o disminuida en función de las entradas. El programa no puede escribirlo.
Valor de preselección	%Ci.P	0 ≤ %Ci.P ≤ **9999**. Puede ser leída y esrita por el programa.
Ajuste	S/N	**S:** permite la modificación de %Ci.P N: no accesible %Ci.P
Entrada de reset	R	En estado 1 pone %Ci.V=0
Entrada set	S	En estado 1 pone %Ci.V=%CI.P
Entrada de contaje	CU	Con la aparición de un flanco ascendente en la entrada CU, el valor actual de %Ci incrementa en una unidad.
Entrada de descontaje	CD	Con la aparición de un flanco ascendente en la entrada CD, el valor actual de %Ci decrementa en una unidad.
Salida desbordamiento	E (empty)	El bit %Ci.E pasa a estado 1 cuando en descontaje el valor actual pasa de 0 a 9999..
Salida preselección alcanzada	D (done)	El bit %Ci.D se mantiene en estado 1 mientras %Ci.V = %Ci.P
Salida desbordamiento	F (full)	El bit %Ci.F pasa a estado 1 cuando en contaje el valor actual pasa de 9999 a 0.

Ejemplo de programación del contador.




```
BLK   %C8
LD    %I0.1
R
LD    %I0.2
AND   %I0.3
CU
END_BLK
LD    %C8.D
ST    %Q0.3
```

%Ci

6.5 Bloque de función registro.

Un registro es un bloque de memoria que permite almacenar hasta 16 palabras de 16 bits de dos maneras diferentes:

- **FIFO** (First In ,First Out) Primero en entrar, primero en salir
- **LIFO** (Last In ,First Out) Ultimo en entrar, primero en salir

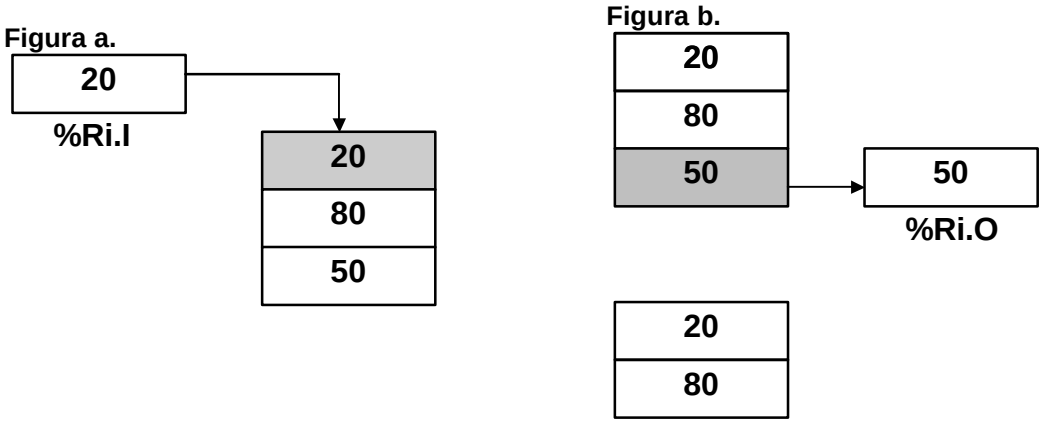
Características: (En **negrita** se detallan los valores por defecto).

Número de registro	%Ri	0 a 3
Tipo	**FIFO** LIFO	
Palabra de entrada	%Ri.I	Palabra de entrada al registro. Puede leerse y escribirse.
Palabra de salida	%Ri.O	Palabra de salida del registro. Puede leerse y escribirse.
Entrada “Ingreso” de dato	I (in)	En un flanco ascendente almacena el contenido de la palabra %Ri.I en el registro.
Entrada “Egreso” de dato	O (out)	En un flanco ascendente coloca una palabra del registro en la palabra %Ri.O.
Entrada de reset	R	En estado 1 inicializa el registro.
Salida vacío	E (empty)	El bit %Ri.E puesto a 1 indica que el registro está vacío.
Salida lleno.	F (full)	El bit %Ri.F puesto a 1 indica que el registro está lleno.

Funcionamiento:

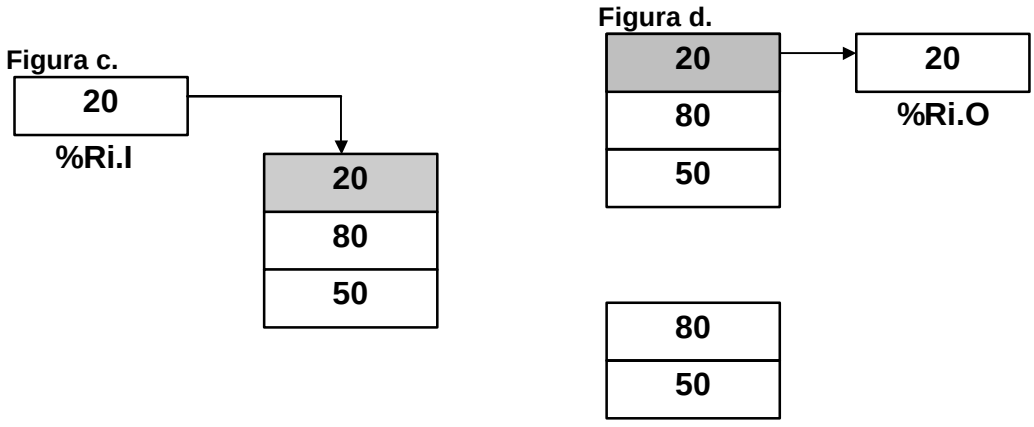
FIFO (First In, First Out).

En este registro el primer dato ingresado es el primero en salir. Cuando aparece un flanco ascendente en la entrada **I**, el contenido de la palabra de entrada (%Ri.I) se almacena en el punto mas alto de la pila (Fig. a). Cuando la pila está llena (%Ri.F=1) es imposible almacenar otro dato. Cuando aparece un flanco ascendente en la entrada **O**¸ el dato contenido en la parte mas baja de la pila se transfiere a la palabra de salida (%Ri.O), y el contenido del registro se desplaza un paso hacia abajo (Fig. b). Cuando el registro está vacío (%Ri.E=1) es imposible sacar datos, la palabra de salida %Ri.O no cambia.



LIFO (Last In, First Out).

En este registro el último dato ingresado es el primero en salir. Cuando aparece un flanco ascendente en la entrada **I**, el contenido de la palabra de entrada (%Ri.I) se almacena en el lugar mas alto de la pila (Fig. c). Cuando la pila está llena (%Ri.F=1) es imposible almacenar otro dato. Cuando aparece un flanco ascendente en la entrada **O**¸ el dato contenido en la posición mas alta de la pila se transfiere a la palabra de salida (%Ri.O), y el contenido del registro se desplaza un paso hacia arriba (Fig. d). Cuando el registro está vacío (%Ri.E=1) es imposible sacar datos, la palabra de salida %Ri.O no cambia.

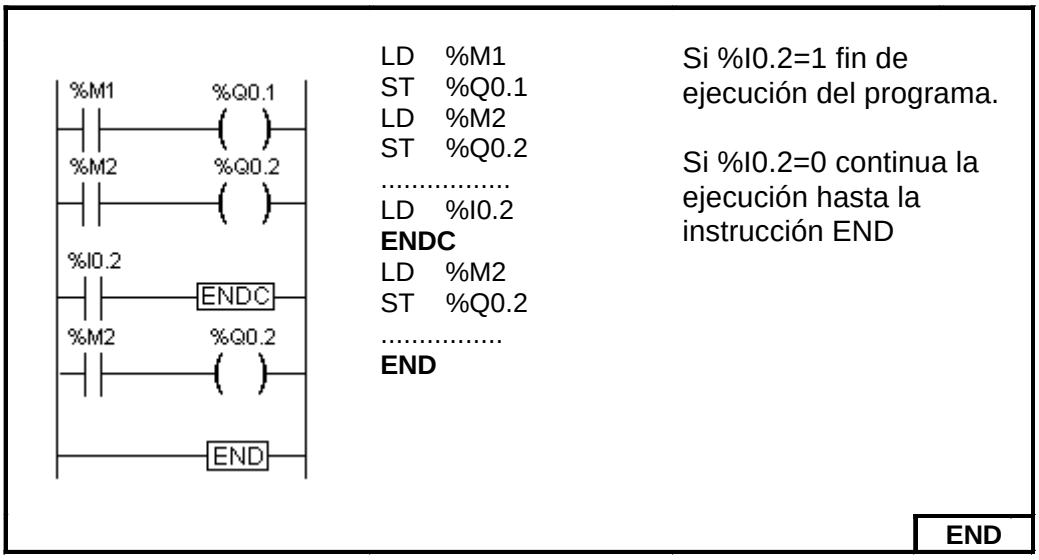


7.- INSTRUCCIONES DE PROGRAMA.

7.1 Instrucciones de fin de programa.

El fin de la ejecución de un ciclo de scan se define usando las instrucciones: END, ENDC Y ENDCN:
END: Fin de programa incondicional.
ENDC: Finaliza el programa si el resultado de la instrucción precedente es 1.
ENDCN: Finaliza el programa si el resultado de la instrucción precedente es 0.
Por defecto (modo normal), cuando se activa el fin del programa, se actualizan las salidas y se inicia el próximo ciclo de scan.
Si el ciclo del autómata es periódico, primero espera el fin del período, y luego actualiza las salidas e inicia el nuevo ciclo periódico.

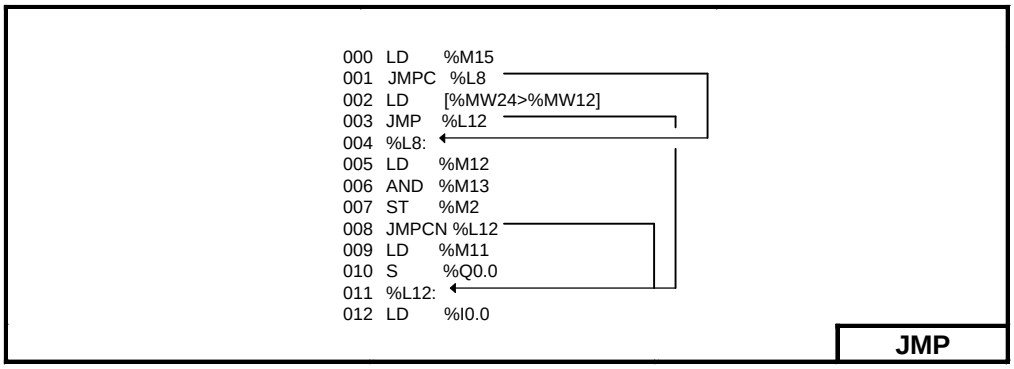
Ejemplo:



7.2 Instrucciones de salto JMP, JMPC y JMPCN.

Las instrucciones JMP, JMPC Y JMPCN provocan la interrupción inmediata de la ejecución y la continuación del programa a partir de la línea de programa indicada con la etiqueta **%Li:** (i = 0 a 15).
JMP: Salto de programa incondicional.
JMPC: Salto de programa si el resultado de la instrucción precedente es 1.
JMPCN: Salto de programa si el resultado de la instrucción precedente es 0.
%Ln: Etiqueta de destino del salto.

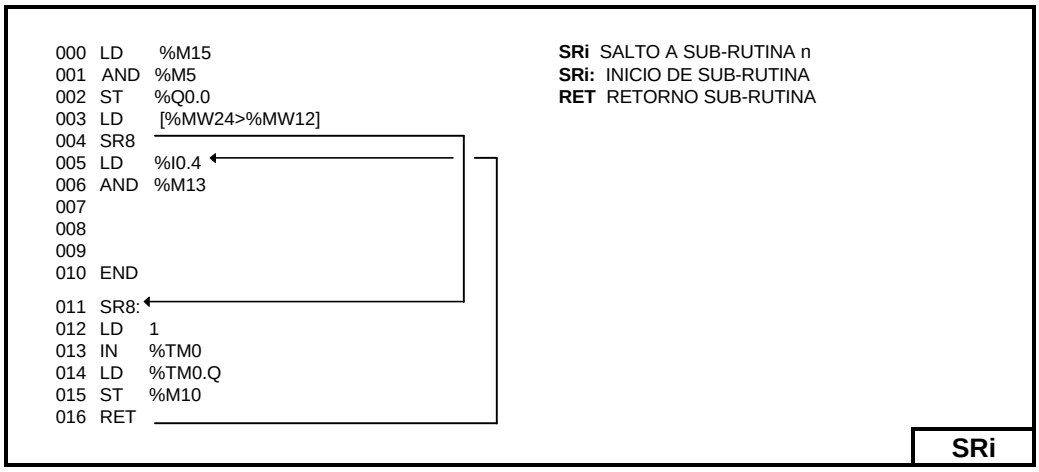
Ejemplo:



7.3 Instrucciones de sub-rutinas SRi, SRi: y RET.

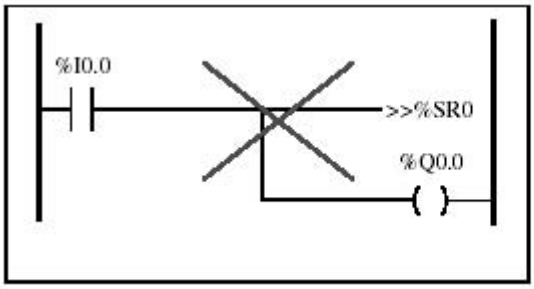
La instrucción **SRi** efectúa el llamado a una sub-rutina identificado por la etiqueta **SRi:** si el resultado de la instrucción booleana precedente es igual a 1.
La instrucción **RET** se coloca al final de la sub-rutina para indicar el retorno al programa principal. La etiqueta **SRi:** hace referencia a la subrutina con i= 0 a 15 en los controladores Twido TWDLCAA10DRF y TWDLCAA16DRF (compactos de 10 y 16 E/S) , siendo i= 0 a 63 para los otros controladores.

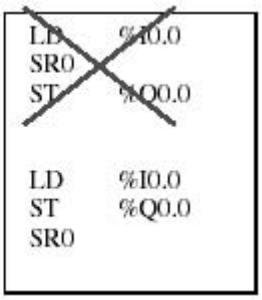
Ejemplo



Notas:

- Una subrutina no debe llamar a otra subrutina.
- Las instrucciones de subrutina no están permitidas entre paréntesis y no deben situarse entre las instrucciones AND(, OR(, y una instrucción de cierre de paréntesis ")".
- La llamada a la subrutina no debe ir seguida por una instrucción de asignación. Esto se debe a que es posible que la subrutina modifique el contenido del acumulador booleano. Por lo tanto, es posible que, durante la respuesta, tenga un valor diferente al que tenía antes de la llamada (consulte el siguiente ejemplo).


%I0.0
>>%SR0
%Q0.0
( )

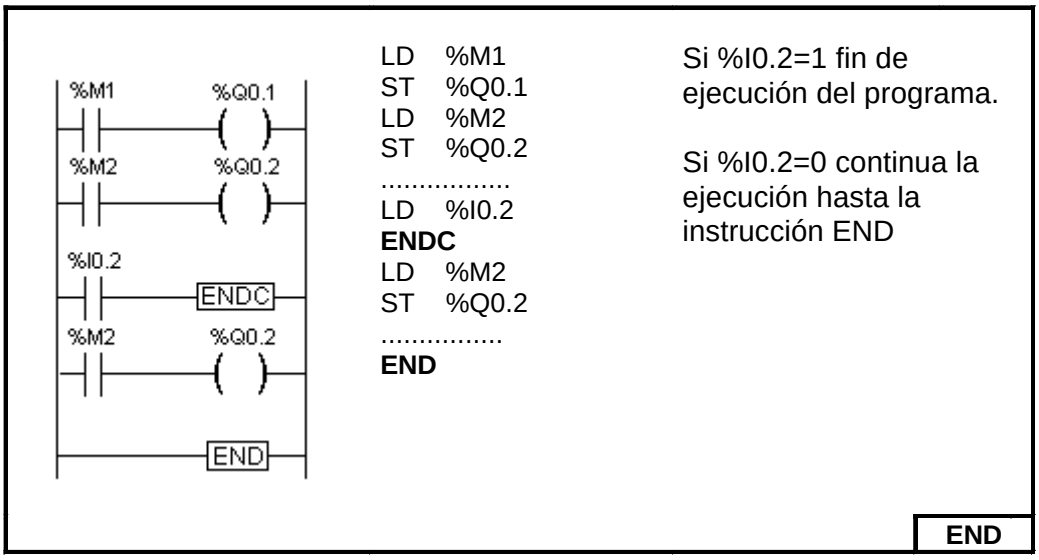

LD %I0.0
SR0
ST %Q0.0
LD %I0.0
ST %Q0.0
SR0

7.- INSTRUCCIONES DE PROGRAMA.

7.1 Instrucciones de fin de programa.

El fin de la ejecución de un ciclo de scan se define usando las instrucciones: END, ENDC Y ENDCN:
END: Fin de programa incondicional.
ENDC: Finaliza el programa si el resultado de la instrucción precedente es 1.
ENDCN: Finaliza el programa si el resultado de la instrucción precedente es 0.
Por defecto (modo normal), cuando se activa el fin del programa, se actualizan las salidas y se inicia el próximo ciclo de scan.
Si el ciclo del autómata es periódico, primero espera el fin del período, y luego actualiza las salidas e inicia el nuevo ciclo periódico.

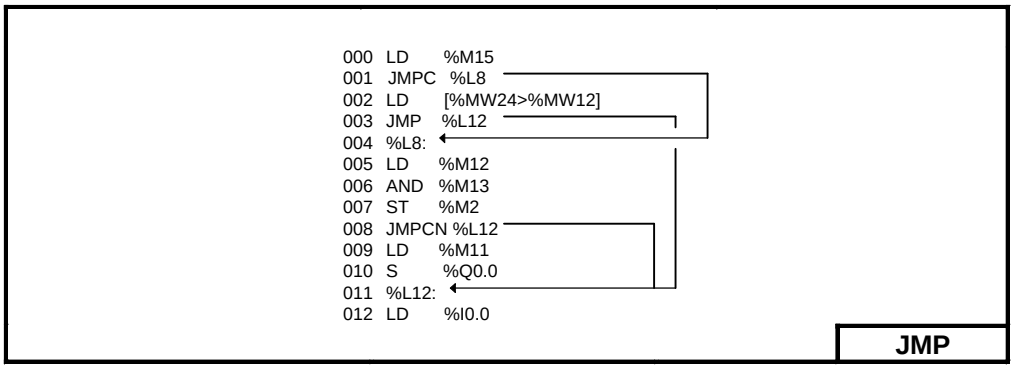
Ejemplo:



7.2 Instrucciones de salto JMP, JMPC y JMPCN.

Las instrucciones JMP, JMPC Y JMPCN provocan la interrupción inmediata de la ejecución y la continuación del programa a partir de la línea de programa indicada con la etiqueta **%Li:** (i = 0 a 15).
JMP: Salto de programa incondicional.
JMPC: Salto de programa si el resultado de la instrucción precedente es 1.
JMPCN: Salto de programa si el resultado de la instrucción precedente es 0.
%Ln: Etiqueta de destino del salto.

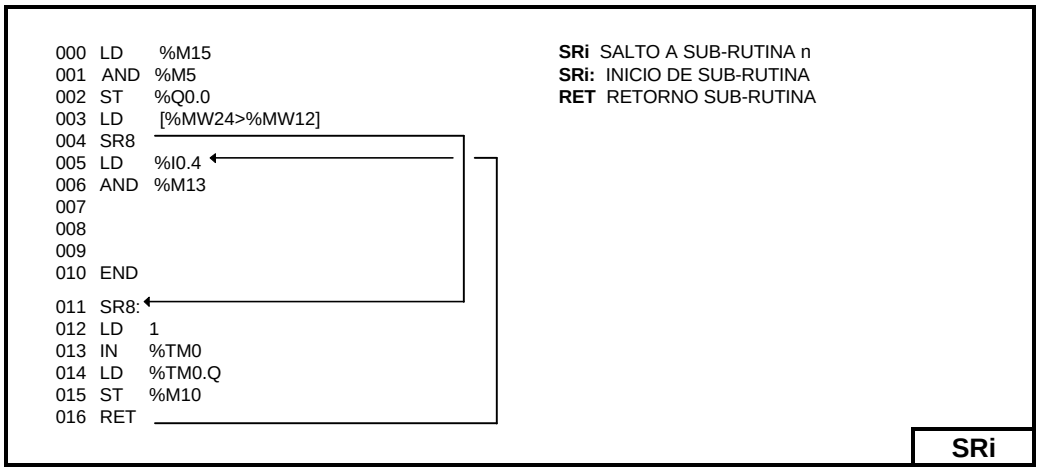
Ejemplo:



7.3 Instrucciones de sub-rutinas SRi, SRi: y RET.

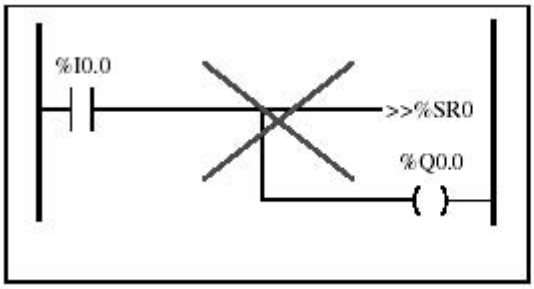
La instrucción **SRi** efectúa el llamado a una sub-rutina identificado por la etiqueta **SRi:** si el resultado de la instrucción booleana precedente es igual a 1.
La instrucción **RET** se coloca al final de la sub-rutina para indicar el retorno al programa principal. La etiqueta **SRi:** hace referencia a la subrutina con i= 0 a 15 en los controladores Twido TWDLCAA10DRF y TWDLCAA16DRF (compactos de 10 y 16 E/S) , siendo i= 0 a 63 para los otros controladores.

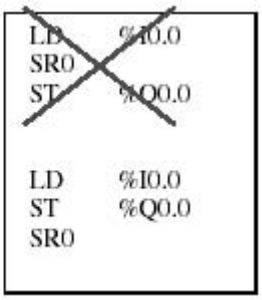
Ejemplo



Notas:

- Una subrutina no debe llamar a otra subrutina.
- Las instrucciones de subrutina no están permitidas entre paréntesis y no deben situarse entre las instrucciones AND(, OR(, y una instrucción de cierre de paréntesis ")".
- La llamada a la subrutina no debe ir seguida por una instrucción de asignación. Esto se debe a que es posible que la subrutina modifique el contenido del acumulador booleano. Por lo tanto, es posible que, durante la respuesta, tenga un valor diferente al que tenía antes de la llamada (consulte el siguiente ejemplo).


%I0.0
>>%SR0
%Q0.0
( )


LD %I0.0
SR0
ST %Q0.0
LD %I0.0
ST %Q0.0
SR0

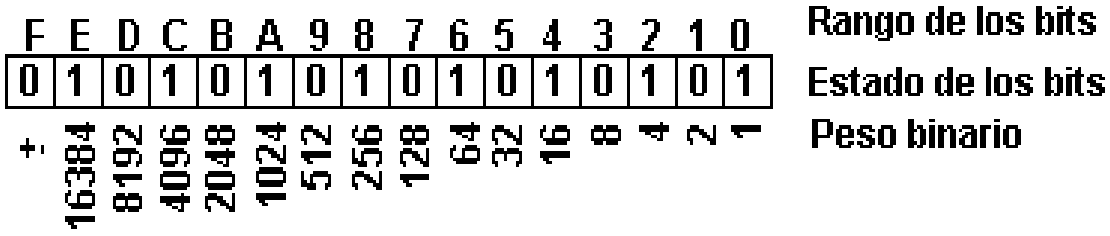
8.- TRATAMIENTO NUMÉRICO.

8.1 Definición de los principales objetos de palabra.

Los objetos de palabra, situados en la memoria de datos, se direccionan bajo el formato palabra de 16 bits de longitud. Contienen un valor algebraico comprendido entre -32768 y 32767 (excepto el contador rápido que evoluciona entre 0 y 65535).

Valores inmediatos: Son los valores algebraicos de formato homogéneo al de las palabras de 16 bits, quienes permiten la afectación de valores a estas palabras. Ellos son almacenados en la memoria del programa y están comprendidos entre - 32768 y 32767.

Formato de las palabras: El contenido de las palabras , valores numéricos o códigos de caracteres está registrado en memoria en código binario, sobre 16 bits, con la convención ilustrada a continuación.



En sistema binario con signo, el bit de rango “F” se atribuye según la convención al signo del valor codificado.
· Bit F a 0: El contenido de la palabra es un valor positivo.
· Bit F a 1: El contenido de la palabra es un valor negativo.
Las palabras y valores inmediatos pueden ser introducidos, visualizados o restituidos bajo la forma:
· decimal: 1579 (máximo 32767, mínimo -32768)
· hexadecimal: 16#A536 (máximo 16#FFFF, mínimo 16#0000)

Palabras internas (%MW)

Las palabras internas están destinadas al almacenamiento de los valores en curso de explotación por el programa. Ellas se ubican en la zona de memoria de datos.
Las palabras %MW0 a %MW255 son accesibles directamente por programa en lectura / escritura. Se utilizan como palabras de trabajo. El número máximo de %MW es de 1500 (con chip de ampliación de memoria).

Entradas y salidas analógicas (%IW / %QW)

Las entradas y salidas analógicas se leen y se escriben como palabras de 16 bits. El formato es el mismo que se emplea para las de tipo bit, con la diferencia que se agrega la letra W luego del identificados %I o %Q.

Palabras constantes (%KW)

Almacenan mensajes alfanuméricos o constantes. Su contenido sólo se puede escribir o modificar utilizando TwidoSoft durante la configuración. Las palabras constantes %KW0 a %KW63 tienen acceso de sólo lectura para el programa.

Palabras de intercambio (%INW / %QNW)

Asignadas a controladores conectados como conexiones remotas. Estas palabras se utilizan para la comunicación entre controladores.

Palabras sistema (%SW)

Estas palabras de 16 bit son internas y controladas por la CPU y aseguran varias funciones: dan acceso a las informaciones que provienen directamente del autómata mediante la lectura de las palabras %SWi, y permiten actuar sobre la aplicación (Ej: ajuste del reloj calendario). En el capítulo Bits y Palabras Sistema, se detallan algunas de las palabras sistema.

Extracción de bits de palabras

Es posible extraer de una palabra uno de sus 16 bits. La referencia de la palabra se completa entonces con el rango del bit extraído, separado por dos puntos.

Sintaxis: % Palabra: Xk con k=0 a 15 rango del bit de la palabra.
Ejemplo: %MW5:X6 - Bit de rango 6 de la palabra interna %MW5.

8.2 Objetos estructurados.

8.2.1 Cadenas de bits.

Las cadenas de bits son una serie de objetos bits adyacentes del mismo tipo y de longitud definida «:L».

Ejemplo de cadena de 6 bits a partir de %M10: %M10:6

%M10	%M11	%M12	%M13	%M14	%M15

Tipo	Dirección	Máximo	escritura
Bits de entrada	%I0:L o %I1:L	0 < L < 17	no
Bits de salida	%Q0:L o %Q1:L	0 < L < 17	si
Bits sistema	%Si:L	0 < L < 17 e i+L ≤ 128	según i
Bits internos	%Mi:L	0 < L < 17 e i+L<128	si

La instrucción asignación (:=) permite explotar las cadenas de bits.

8.2.2 Tablas de palabras.

Las tablas de palabras son series de palabras adyacentes del mismo tipo y de longitud definida «:L»

Ejemplo de tabla de palabras: %KW10:7

%KW10 16 bits
%KW16

Tipo	Sintaxis	Máximo	Escritura
Palabras internas	%MWi:L	0 < L < 256 e i+L ≤ 256	si
Palabras constantes	%KWi:L	0 < L < 64 e i+L ≤ 64	no
Palabras sistema	%SWi:L	0 < L < 128 e i+L ≤ 128	según i

8.2.3 Palabras indexadas.

8.2.3.1 Direccionamiento directo

Llamamos direccionamiento directo de objetos, cuando la dirección de los mismos está fijada y definida en la escritura del programa.
Ejemplo: %MW26 (Palabra interna de dirección 26)

8.2.3.2 Direccionamiento indexado.

En este tipo de direccionamiento, un índice completa la dirección directa del objeto: a la dirección del objeto se le suma el contenido del índice. El índice se define por una palabra interna %MWi.
Ejemplo: %MW108[%MW2]: palabra de dirección directa 108 + contenido de la palabra %MW2. Si la palabra %MW2 contiene el valor 12, la dirección efectiva es 120 (Equivale a la palabra %MW120).

8.3 Instrucciones numéricas.

8.3.1 Instrucción de asignación.

Realizan la carga de un operando Op2 en un operando Op1

Sintaxis [Op1:=Op2] <=> Op2 -> Op1

Las operaciones de asignación pueden realizarse en:

☑Cadenas de bits.
☑Palabras.
☑Tablas de palabras.

8.3.1.1 Asignación de cadenas de bits.

En cadenas de bits se pueden realizar las siguientes operaciones :

☑Cadena de bits -> Cadena de bits.
☑Cadena de bits -> Palabra
☑Palabra -> Cadena de bits
☑Valor inmediato -> Cadena de bits.

8.3.1.2 Asignación de palabra.

Pueden realizarse las siguientes operaciones:

☑Palabra -> Palabra
☑Palabra indexada -> Palabra
☑Valor inmediato -> Palabra
☑Cadena de bits -> Palabra
☑Palabra -> Palabra
☑Palabra indexada -> Palabra
☑Valor inmediato -> Palabra
☑Palabra -> Cadena de

8.3.1.3 Asignación de tablas de palabras.

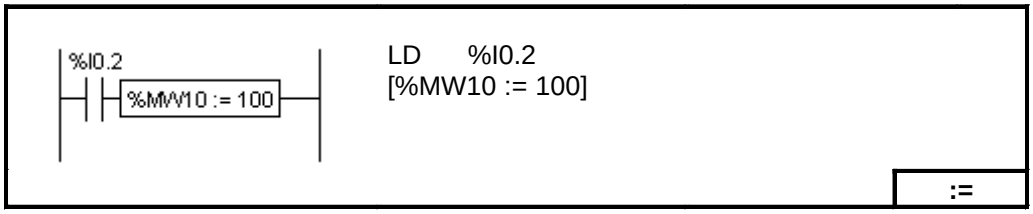
A continuación se detallan las operaciones posibles:

☑ Valor inmediato -> Tabla de palabras
☑ Palabra -> Tabla de palabras
☑ Tabla de palabras -> Tabla de palabras

8.3.1.4 Ejemplos de asignaciones.

%MWi := MWi	palabra a palabra
%MWi(%MWi):= %Mwi	palabra a palabra indexada
%MWi := %MWi(%MWi)	palabra indexada a palabra
%MWi:= 100	valor inmediato a palabra
%MWi(%MWi):= 50	valor inmediato a palabra indexada
%MWi :L := 700	valor inmediato a tabla de palabras
%MWi :L := %MWi :L	tabla de palabra a tabla de palabras
%M:L := %M:L	cadena de bits a cadena de bits
%MWi := %M:L	cadena de bits a palabras.

Las instrucciones de transferencia se realizan de la siguiente manera:

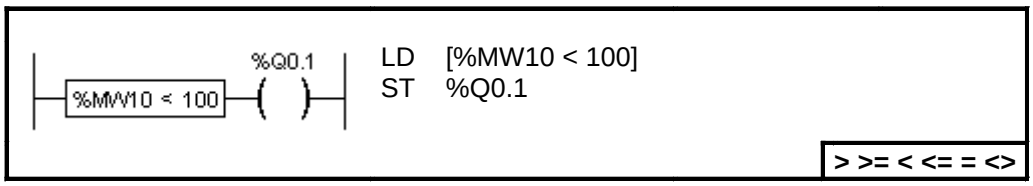


8.3.2 Instrucciones de comparación.

Estas instrucciones permiten realizar la comparación entre dos operandos
> Prueba si el operando 1 es superior al operando 2
> = Prueba si el operando 1 es superior o igual al operando 2
< Prueba si el operando 1 es inferior al operando 2
< = Prueba si el operando 1 es inferior o igual al operando 2
= Prueba si el operando 1 es igual al operando 2
< > Prueba si el operando 1 es diferente al operando 2

La comparación se realiza entre corchetes detrás de las instrucciones LD, AND, y OR , cuando la comparación es verdadera su resultado es 1 y hay continuidad lógica.

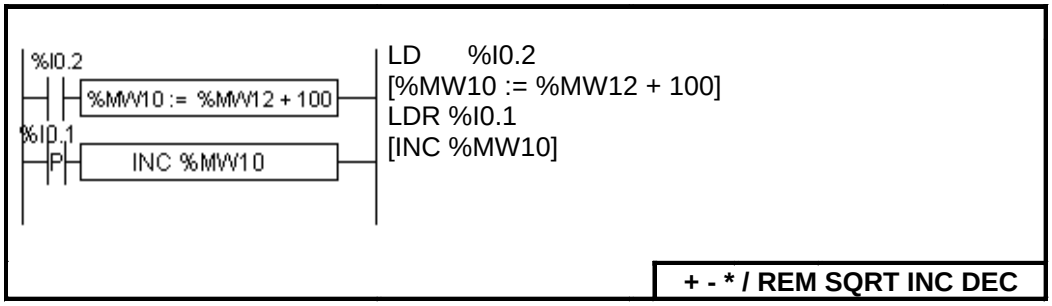
Ejemplo:



8.3.3 Instrucciones aritméticas.

Permiten realizar una operación aritmética entre dos operandos o en un operando.
+ suma de dos operandos
- resta de dos operandos
* producto de dos operandos
/ división de dos operandos
REM resto de la división de dos operandos
SQRT raíz cuadrada de dos operandos
INC incremento de un operando
DEC decremento de un operando

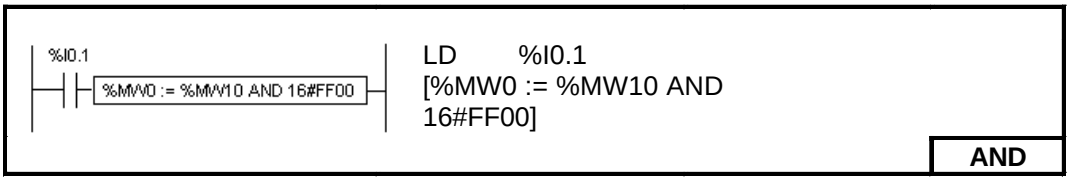
Las operaciones aritméticas se realizan de la siguiente forma:



8.3.4 Instrucciones lógicas.

Estas instrucciones permiten realizar una operación lógica entre dos operandos o en un operando.
AND : producto lógico entre dos operandos (bit a bit),
OR : suma lógica entre dos operandos (bit a bit),
XOR : suma lógica exclusiva entre dos operandos (bit a bit),
NOT : negación de un operando (bit a bit),

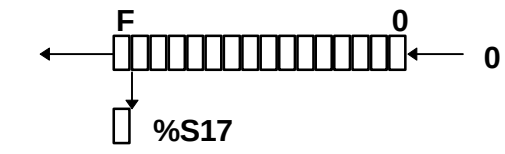
Las operaciones lógicas se realizan de la siguiente manera:



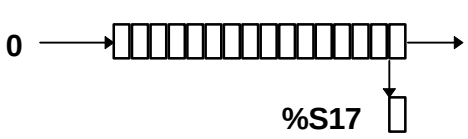
8.3.5 Instrucciones de rotación.

Estas instrucciones consisten en desplazar los bits de un operando, un cierto número de posiciones hacia la derecha o hacia la izquierda.
Existen dos tipos de desplazamiento:

8.3.5.1 Desplazamiento lógico:

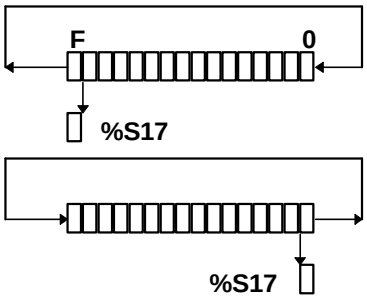


☑ SHL(Op,i): Desplazamiento lógico hacia la izquierda



☑ SHR(Op,i): Desplazamiento lógico hacia la derecha.

8.3.5.2 Desplazamiento circular:

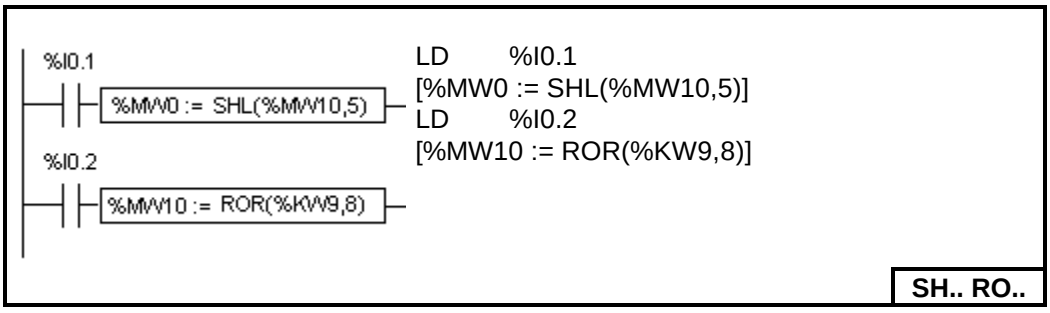


☑ ROL(Op,i): Desplazamiento circular hacia la izquierda

☑ ROR(Op,i): Desplazamiento circular hacia la derecha.

8.3.5.3 Estructura.

Como el operando a desplazar tiene una longitud normal (16 bits), la variable i estará comprendida necesariamente entre 1 y 16. El estado del último bit desplazado o rotado queda memorizado en el bit %S17.
Las instrucciones de desplazamiento se realizan de la siguiente manera:



8.3.6 Instrucciones de conversión.

Existen dos instrucciones de conversión:

☑ BTI : conversión BCD -> Binario

☑ ITB : conversión Binario -> BCD

El código BCD (Binary Coded Decimal) que significa Decimal Codificado en Binario, permite representar una cifra decimal de 0 a 9 mediante un conjunto de 4 bit. Una palabra de 16 bit puede así contener un número expresado en 4 cifras. ($0 \leq N \leq 9999$).

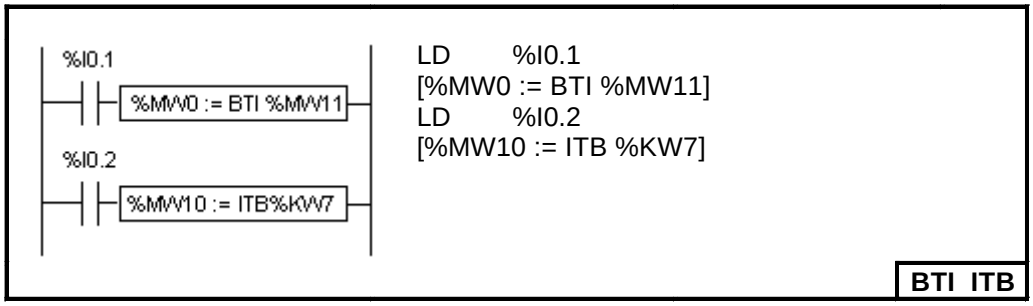
Decimal	0	1	2	3	4	5	6	7	8	9
BCD	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Ejemplo:
La palabra %MW5 expresa el valor BCD «2450»; corresponde al valor binario
0010 0100 0101 0000.

La palabra %MW12 expresa el valor DECIMAL «2450»; corresponde al valor binario **0000 1001 1001 0010**

El pasaje de la palabra %MW5 a la palabra %MW12 se realiza a través de la instrucción BTI.

El pasaje de la palabra %MW12 a la palabra %MW5 se realiza a través de la instrucción ITB.

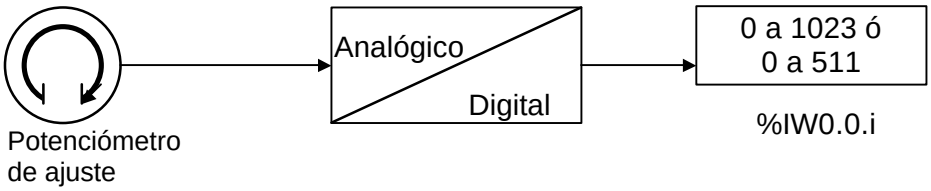


9.- FUNCIÓN ANALÓGICA.

9.1 Puntos de reglaje analógico.

9.1.1 Principio

Un conversor analógico / digital convierte la tensión a los bornes de un potenciómetro en un valor numérico, el cual es colocado en una palabra.



Los controladores Twido tienen:

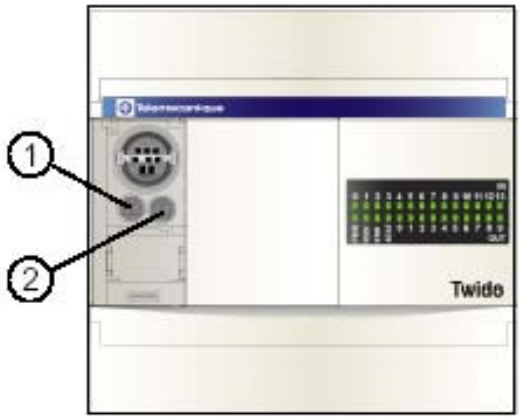
- Un potenciómetro en los controladores Compactos de 10 y 16 E/S, y uno en los controladores modulares.
- Dos potenciómetros en el controlador compacto de 24 E/S.

9.1.2 Programación.

Los valores numéricos, de 0 a 1023 para el potenciómetro 1 y de 0 a 511 para el potenciómetro 2, correspondientes a los valores analógicos que indican estos potenciómetros, se direccionan como Input Word de la siguiente manera:

- %IW0.0.0 para el potenciómetro 1 (más a la izquierda)
- %IW0.0.1 para el potenciómetro 2 (más a la derecha)

Estas palabras se pueden utilizar en operaciones aritméticas y para cualquier ajuste, por ejemplo, preestablecer un retardo o un contador, ajustar la frecuencia del generador de pulsos o el precalentamiento de una máquina, sin recurrir a un terminal de diálogo electrónica para hacerlo.



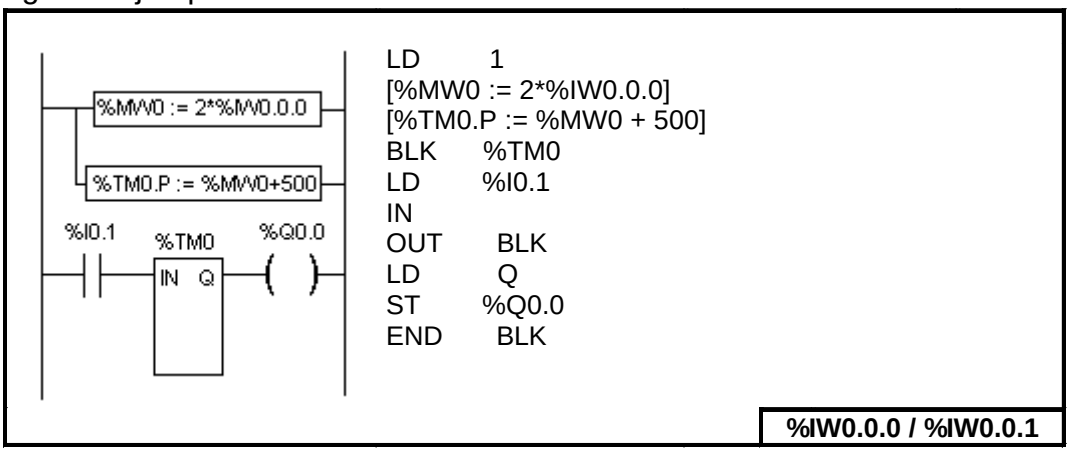
9.1.3 Ejemplo de programación.

Ajuste de un retardo de 5 a 25 segundos utilizando el potenciómetro 1:

Los siguientes parámetros están seleccionados en la configuración del bloque de retardo %TM0:

- Tipo TON
- Base de tiempo TB: 10 ms

El valor predeterminado del retardo se calcula a partir del valor de ajuste del potenciómetro utilizando la siguiente ecuación %TM0.P := 2 * %IW0.0.0 + 500.
Código del ejemplo anterior:



9.2 Entrada Analógica Integrada.

9.2.1 Principio.

Todos los controladores modulares disponen de un canal analógico incorporado. La entrada de tensión varía entre 0 y 10 V y la señal digitalizada entre 0 y 511. El canal analógico aprovecha un esquema de promedio simple que se aplica a ocho muestras.

Un convertidor de digital a analógico muestrea una tensión de entrada de 0 a 10 V con un valor digital de 0 a 511. Este valor se almacena en la Input Word %IW0.0.1. El valor es lineal en todo el rango, de modo que cada conteo es aproximadamente de 20 mV (10 V/512). Una lectura de 511 se utiliza para detectar si se ha superado el valor máximo de la señal de entrada.

9.2.2 Programación.

Los valores numéricos de 0 a 511, que corresponden al valor analógico proporcionado por la entrada, está disponible en la Input Word %IW0.0.1, la misma que es usada el Potenciómetro 2 en los compactos de 24 E/S.

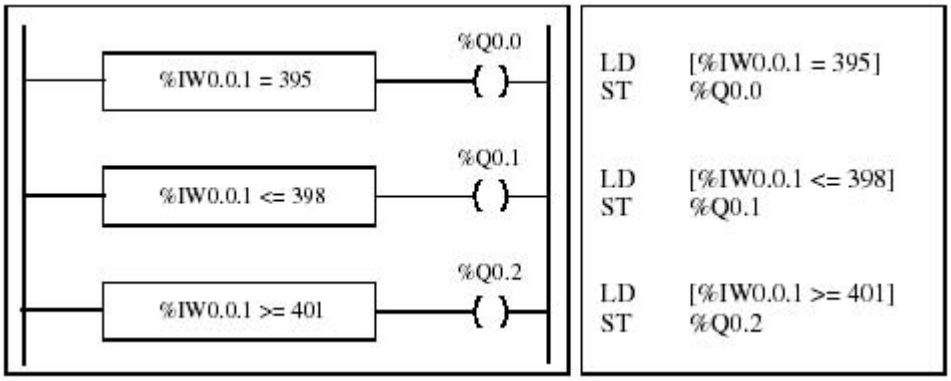
Esta palabra puede utilizarse, mediante operaciones aritméticas, para cualquier tipo de ajuste (preselección de un temporizador, del contador, etc) o como lectura de una variable física externa (Temperatura, Presión, Caudal, etc).

9.2.3 Ejemplo de programación.

Control de la temperatura de un horno: La temperatura del horno se fija en 350 °C. Una variación de +/- 2,5 °C supone la interrupción de las salidas %Q0.1 y %Q0.2. En este ejemplo se utilizan prácticamente todos los rangos de configuración posibles del canal analógico de 0 a 511. La configuración analógica de los valores teóricos de temperatura es la siguiente.

Temperatura (°C)	Tensión	%IW0.0.1
0	0	0
347,5	7,72	395
350	7,77	398
352,5	7,83	401
450	10	511

Código del ejemplo anterior:



9.3 Módulos analógicos de gestión

9.3.1 Introducción

Además del potenciómetro integrado de 10 bits y el canal analógico de 9 bits, todos los controladores Twido Modulares pueden incorporar módulos de E/S analógicas.
Los módulos son los siguientes:

Nombre	Canales	Rango de señal	Codificado
TWDAMI2HT	2 entradas	0 a 10 V o 4 a 20 mA	12 Bit
TWDAM01HT	1 salida	0 a 10 V o 4 a 20 mA	12 Bit
TWDAMM3HT	2 entradas, 1 salida	0 a 10 V o 4 a 20 mA	12 Bit
TWDALM3LT	2 entradas, 1 salida	0 a 10 V, entradas Th o RTD, salidas 4 a 20 mA	12 Bit

9.3.2 Funcionamiento de módulos analógicos.

Las palabras de entrada y de salida (%IW y %QW) se utilizan para intercambiar datos entre la aplicación del usuario y cualquier canal analógico. La actualización de estas palabras se lleva a cabo de manera sincronizada con la ejecución del controlador con el modo de ejecución.

Cuando el controlador pasa a STOP, la salida analógica se establece en su posición anterior. **Si no se respetan estas precauciones pueden producirse daños corporales y/o materiales.**

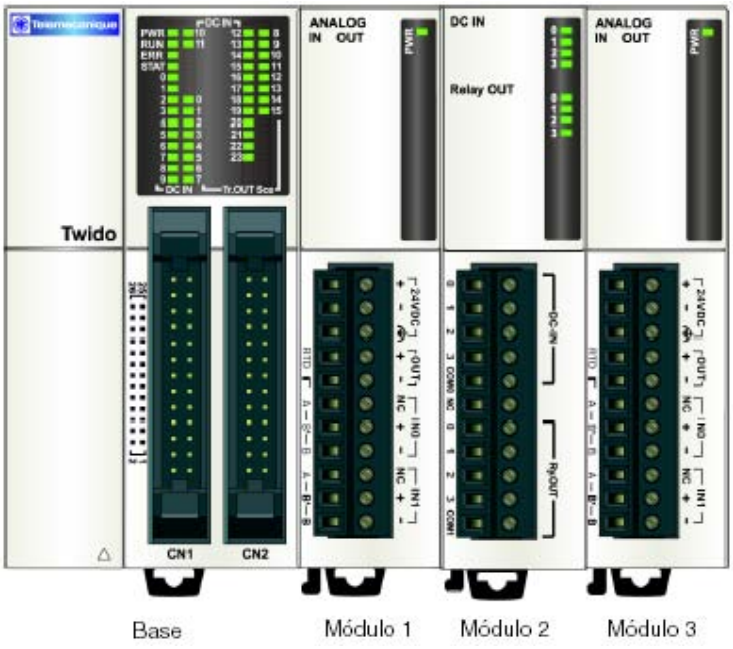
9.3.3 Direccionamiento de entradas y salidas analógicas

Se asignan direcciones a los canales analógicos según su ubicación en el bus de ampliación. La dirección de la E/S analógica es similar a la explicada en el capítulo 5, la única diferencia es el formato de los datos. En el Cap. 5 se ejemplifican las E/S **digitáles**, en el caso de las **analógicas**, se requiere una palabra de entrada o una palabra de salida respectivamente. Esta diferencia se manifiesta en la letra **W (WORD),** que acompaña a la letra **I** o **Q** indicando el formato palabra, y el direccionamiento será:

- **%IWCont.Mod. Vía (Entrada analógica)**
- **%QWCont.Mod.Vía (Salida Analógica)**

Cont: Posición del Controlador, 0 = Base
Mod: Módulo donde está colocada la E/S analógica
Vía: Numero de la E/S.

A continuación se completa la explicación del direccionamiento de las entradas y salidas analógicas con un ejemplo: El controlador TWDLMDA40DUK tiene el potenciómetro integrado de 10 bit y un canal analógico integrado de 9 bits. En el bus de ampliación, se configuran un módulo TWDAMM3HT, un modelo de relé digital de entrada / salida TWDDMM8DRT y un segundo módulo analógico TWDAMM3HT.



La tabla que aparece a continuación proporciona información acerca del direccionamiento de cada entrada / salida analógica o digital.

Descripción	Base	Módulo 1	Módulo 2	Módulo 3
Potenciómetro 1	%IW0.0.0			
Canal analógico integrado	%IW0.0.1			
Canal 1 de entrada analógica		%IW0.1.0		%IW0.3.0
Canal 2 de entrada analógica		%IW0.1.1		%IW0.3.1
Canal 1 de salida analógica		%QW0.1.0		%QW0.3.0
Canales de entrada digital			%I0.2.0 %I0.2.3	
Canales de salida digital			%Q0.2.0 %Q0.2.3	

9.3.4 Configuración de E/S analógicas

El cuadro de diálogo Configurar módulo se utiliza para administrar los parámetros de los módulos analógicos. Las direcciones se asignan a los canales analógicos según su ubicación en el bus de ampliación. Para facilitar la programación, también puede asignar símbolos previamente definidos para gestionar los datos en la aplicación.
Los modelos **TWDAM01HT, TWDAMM3HT y TWDALM3LT** poseen un único canal de salida, el cual se puede configurar como:

- No utilizado
- 0 - 10 V
- 4 - 20 mA

Los modelos **TWDAMI2HT y TWDAMM3HT** poseen dos canales de entrada, los cuales se pueden configurar como:

- No utilizado
- 0 - 10 V
- 4 - 20 mA

Nota: Los parámetros sólo se pueden modificar en estado offline, cuando no esté conectado al controlador.

Los dos canales de entrada del modelo **TWDALM3LT** se pueden configurar como:

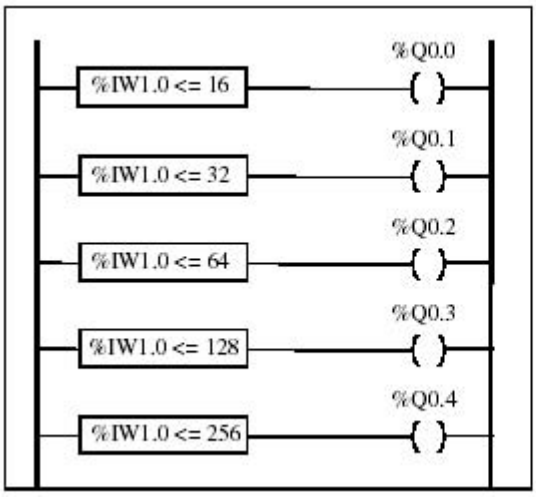
- No utilizado
- Termopar K
- Termopar J
- Termopar T
- PT 100

Cuando se configura un canal, puede elegir entre asignar unidades de medición de temperatura y asignar el rango de entradas según la tabla que aparece a continuación.

Rango	Unidades	Descripción
Normal	Ninguna	Rango establecido desde un mínimo de 0 hasta 4.095.
Personalizado	Ninguna	Definido por el usuario, con un mínimo no inferior a -32.768 y un máximo no superior a 32.767.
Centígrados	0,1 °C	Escala termométrica internacional. Sólo disponible para los canales de entrada del modelo **TWDALM3LT**.
Fahrenheit	0,1 °F	Escala termométrica en la que el punto de ebullición del agua es 212 °F (100 °C) y el de congelación es 32 °F (0 °C). Sólo disponible para los canales de entrada del modelo **TWDALM3LT**.

9.3.5 Ejemplo de programación.

En este ejemplo, la señal de entrada analógica se compara con cinco valores de umbral independientes. Se realiza una comparación de la entrada analógica y se ajusta un bit en el controlador base si la entrada es menor que el umbral.




```
LD [%IW1.0 <= 16]
ST  %Q0.0

LD [%IW1.0 <= 32]
ST %Q0.1

LD [%IW1.0 <= 64]
ST %Q0.2

LD [%IW1.0 <= 128]
ST %Q0.3

LD [%IW1.0 <= 256]
ST %Q0.4
```

10.- FUNCIONES ESPECIALES.

10.1 Programador y consignador temporal.

En todos los modelos de los autómatas TWIDO, es posible incorporarles como accesorio un reloj de tiempo real, a partir del cual se pueden elaborar dos funciones.

☑ Programador Temporal.

☑ Consignador Temporal.

La actualización del reloj interno del autómata se efectúa en modo configuración e incluso se puede realizar por programa. Su funcionamiento está asegurado aún cuando el autómata está sin tensión.
Los datos del reloj están disponibles en 5 palabras sistema (%SW49 a %SW53). EL formato del reloj es de 24 Horas y tiene en cuenta los años bisiestos.

10.1.1 Programador temporal. (Fechadores)

Esta función permite comandar acciones en horarios preestablecidos.
Cada Twido, con el opcional de reloj incorporado (TWDXCPRTC), posee 16 (0 a 15) bloques de programación temporal. A cada bloque se le asigna como salida un objeto bit que es puesto a 1 solamente durante los horarios definidos (%Qx.i ó %Mi). Estos bloques no forman parte del programa usuario, y se definen por configuración **(Fechadores)**. La palabra sistema %SW114 permite habilitar o deshabilitar a través de cada uno de sus bits el funcionamiento de cada bloque. El bit 0 corresponde al bloque 0, el bit 1 al bloque 1, y así sucesivamente.
Estos bits puestos a 1 significa que el bloque esta validado, si está en 0 se inhibe su funcionamiento
A modo de ejemplo, a continuación presentamos la pantalla de configuración de un bloque:

Programador temporal: **Bloque 0.** Se activa la salida **%Q0.0**, desde el **3 de Julio** hasta el **1 de Septiembre**, solo los días **Lunes, Martes y Sábados**, entre las **08:00** y las **19:00** horas.

10.1.2 Consignador temporal.

Permite memorizar el día y la hora de la aparición de un evento.
Las palabras sistema %SW50 a la %SW53 contienen el día y la hora en curso, en formato BCD.
Para fechar un evento es suficiente utilizar las operaciones de asignación, para copiar el contenido de las palabras sistema correspondientes en palabras internas (%MWi) y luego tratar dichas palabras por programa.

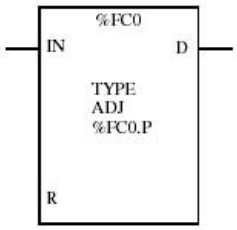
11.- CONTAJE.

11.1 Introducción.

Los autómatas Twido incluyen contadores capaces de contar pulsos de alta velocidad, ingresando los mismos por entradas específicas del controlador. Para esta Función, los autómatas Twido poseen contadores llamados **RÁPIDOS (%FCi)** y **MUY RÁPIDOS (%VFCi).** A continuación se describe el funcionamiento de los mismos.

11.2 Contador Rápido (%FCi).

El bloque de función de contador rápido (%FC) se puede utilizar como contador progresivo o regresivo. Puede contar el flanco ascendente de las entradas digitales con una frecuencia de hasta 5 kHz.
Los controladores compactos se pueden configurar para utilizar un máximo de tres contadores rápidos, mientras que los controladores modulares sólo pueden usar un máximo de dos. Los bloques de función de contador rápido %FC0, %FC1 y %FC2 utilizan las entradas especializadas %I0.0.2, %I0.0.3 y %I0.0.4 respectivamente.
Estos bits no están reservados para su uso exclusivo. Para su ubicación se debe tener en cuenta el uso de otros bloques de función en cuanto a estos recursos especializados.
A continuación se muestra un ejemplo de un bloque de función de contador rápido.



En la tabla siguiente se enumeran los parámetros del bloque de función de contador rápido.

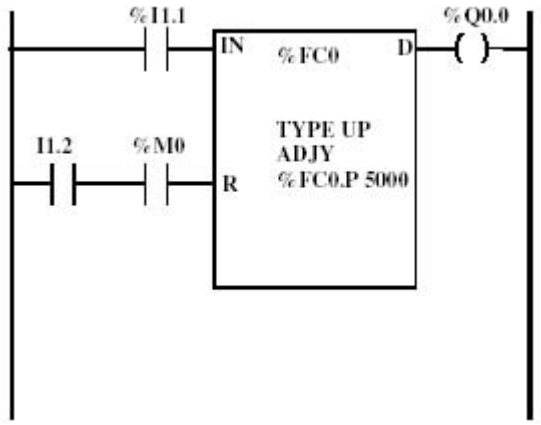
Parámetro	Etiqueta	Descripción
Tipo	TYPE	Contador regresivo o progresivo.
Valor Preestablecido	%FCi.P	Valor inicial ajustado entre 1 y 65535
Ajustable	Y/N	Puesto en Y es factible modificar el valor preestablecido %FCi.P
Valor Actual	%FCi.V	Valor de la cuenta.
Entrada de habilitación	IN	En estado 1, el valor actual se actualiza de acuerdo a los pulsos recibidos. En estado 0, el valor actual mantiene el último valor.
Restablecer	%FCi.R	Usado para inicializar el bloque. En estado 1 se copia en el valor actual, el valor preestablecido.
Finalización	%FCi.D	Este bit se pone a 1: a) cuando el valor actual es igual al valor preestablecido (contador progresivo), b) cuando el valor actual es igual a 0 (contador regresivo).

11.2.1 Operación

Si está configurado como contador progresivo, el valor actual se incrementa en 1 con cada flanco ascendente que aparezca en la entrada especializada. Si el valor es igual que el valor preestablecido %FCi.P, el bit de salida Finalización %FCi.D se pone a 1 y en el valor actual %FCi.V se carga cero. Si está configurado como contador regresivo, el valor actual se reduce en 1 con cada flanco ascendente que aparezca en la entrada especializada. Si el valor es igual a cero, el bit de salida Finalización %FCi.D se pone a 1 y en el valor actual %FCi.P se carga el valor preestablecido.

11.2.2 Configuración y programación

En este ejemplo, la aplicación cuenta un número de elementos hasta 5000 mientras %I1.1 se pone a 1. La entrada para %FC0 es la entrada especializada %I0.0.2. Cuando se alcanza el valor preestablecido, %FC0.D se activa y permanece así hasta que se restablece %FC0.R mediante el resultado de agregar %I1.2 y %M0 con un operador AND lógico.



11.3 Contador Muy Rápido (%VFCi)

El bloque de función de contador muy rápido (%VFC) se puede configurar mediante TwidoSoft y realiza cualquiera de las siguientes funciones:

- Contador progresivo/regresivo
- Contador progresivo/regresivo bifásico
- Contador progresivo
- Contador regresivo
- Frecuencímetro

El %VFC proporciona el conteo de entradas digitales con una frecuencia de hasta 20 kHz. Los controladores compactos pueden configurar un contador muy rápido; los controladores modulares pueden configurar hasta dos contadores muy rápidos.

Los bloques de función de contador muy rápido utilizan entradas especializadas y entradas y salidas auxiliares. Estas entradas y salidas no están reservadas para su uso exclusivo. Para su ubicación se debe tener en cuenta el uso de otros bloques

de función en cuanto a estos recursos especializados. La siguiente tabla resume estas asignaciones.

Asignación de las entradas especializadas.

%VFC	Uso seleccionado	Entradas			
		Principales		Auxiliares	
		Primera entrada de pulsos	Segunda entrada de pulsos	Entrada de Preselección	Entrada Rápida
0	Progresivo / Regresivo	%I0.1 pulsos	%I0.0 Sentido Prog = 1 Re = 0	%I0.2 opcional	%I0.3 Opcional
1		%I0.7 pulsos	%I0.6 Sentido Prog = 1 Re = 0	%I0.5 opcional	%I0.4 Opcional
0	Progresivo / Regresivo Bifásico	%I0.1 pulsos	%I0.0 Pulsos fase B	%I0.2 opcional	%I0.3 Opcional
1		%I0.7 pulsos	%I0.6 Pulsos fase B	%I0.5 opcional	%I0.4 Opcional
0	Progresivo	%I0.1 pulsos	Sin uso	%I0.2 opcional	%I0.3 Opcional
1		%I0.7 pulsos	Sin uso	%I0.5 opcional	%I0.4 Opcional
0	Regresivo	%I0.1 pulsos	Sin uso	%I0.2 opcional	%I0.3 Opcional
1		%I0.7 pulsos	Sin uso	%I0.5 opcional	%I0.4 Opcional
0	Frecuencímetro	%I0.1 pulsos	Sin uso	Sin uso	Sin uso
1		%I0.7 pulsos	Sin uso	Sin uso	Sin uso

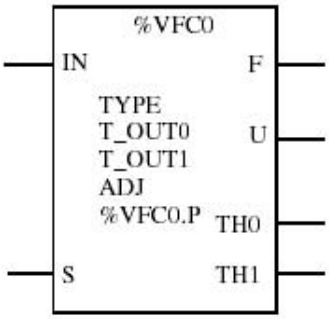
Asignación de las salidas especializadas.

%VFC	Uso seleccionado	Salidas Reflejas	
		Primera	Segunda
0	Conteo (en todos los casos)	%Q0.2 Opcional	%Q0.3 Opcional
1		%Q0.4 Opcional	%Q0.5 Opcional
0	Frecuencímetro	Sin uso	Sin uso
1		Sin uso	Sin uso

Comentarios:

- Si se utiliza %I0.2, no está disponible %FC0.
- Si se utiliza %I0.3, no está disponible %FC2.
- Si se utiliza %I0.4, no está disponible %FC3.

En la figura siguiente, se muestra el formato de un contador muy rápido.



La función de las entradas del bloque son:

IN: Habilita al bloque, en estado 0, los pulsos no son tomados en cuenta.
S: Si está configurado como contador **progresivo**, pone al valor actual (%VFCi.V) en cero, en tanto que si está configurado como contador **Regresivo ó Progresivo / Regresivo**, copia en el valor actual (%VFCi.V) el valor de preselección (%VFCi.P)

Los parámetros del contador muy rápido son los que se muestras en la próxima tabla:

Objeto	Descripción	Valores	Modo	Acc.
%VFCi.V	Valor actual	0 -> 65535	CM o FM	L
%VFCi.P	Valor de preselección, se copia este valor en %VFCi.V cuando se activa la entrada de preselección	0 -> 65535	CM	L/ E (1)
%VFCi.U	Sentido de cuenta. Solo se emplea cuando el contador está configurado como PROG/REG	0: REG 1: PROG	CM	L
%VFCi.R y %VFCI.S	Habilitar salida reflejas .R: salida refleja 0 .S: Salida refleja 1	0: bloqueada 1: habilitada	CM	L/E (2)
%VFCi.S0 y %VFCi.S1	Umbral S0 y S1. con S0 < S1	0 -> 65535	CM	L/E (2)
%VFCi.M	Medida de frecuencia Válida	0: no valida 1: Valida	FM	L
%VFCi.T	Base de tiempo para medición de frecuencia	1000 o 100 mS	FM	L/E
%VFCi.F	Desborde, se activa cuando el contador contó más de 65535 pulsos, se borra cuando se activa la entrada de preselección	0 o 1	CM	L
%VFCi.TH0 y %VFCi.TH1	Umbral alcanzado. TH0: se activa si %VFCi.V >= %VFCi.S0 TH1 se activa si %VFCi.V >= %VFCi.S1	0 ó 1	CM	L

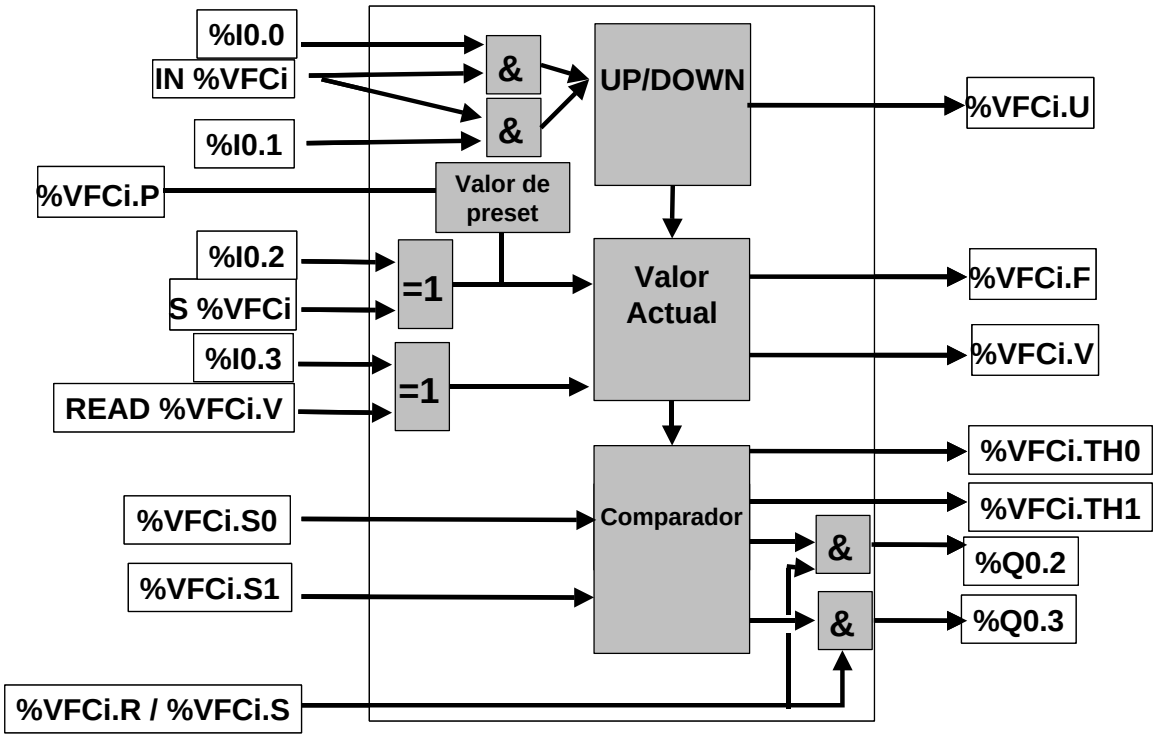
REFERENCIAS:
PROG: Progresivo.
REG: Regresivo.
CM: Modo de contador.
FM: Modo frecuencímetro
L: Lectura.
E: Escritura.

NOTAS:
(1): Se tiene acceso en escritura solo si se configuró el ajuste en SI.
(2): Solo se pueden acceder a estos datos si están configurados (si la función elegida los emplea).

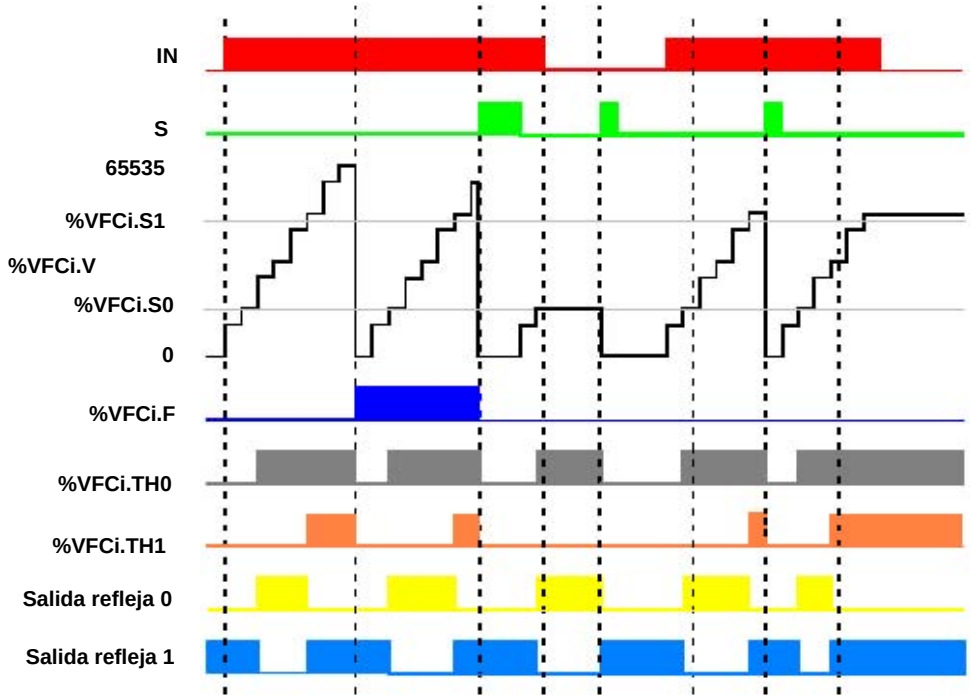
11.3.1 Conteo.

El contador muy rápido puede trabajar de 5 formas distintas (Conteo progresivo, regresivo, Progresivo / Regresivo, Progresivo / Regresivo bifásico y Fecuencímetro.). En todas la funciones de conteo, el contador trabaja de manera similar, por lo que a continuación se mostrará un diagrama en bloques y un diagrama temporal único para los 4 modos, y a continuación se analizarán las particularidades de cada función. Más adelante se analizará el funcionamiento del contador trabajando en modo frecuencímetro.

11.3.1.1 Diagrama en bloques.



11.3.1.2 Diagrama temporal.



11.3.1.3 Contador Muy Rápido, Función Conteo Progresivo.

Cuando el contador muy rápido se configura como contador progresivo, los pulsos a contar ingresan por la entrada específica IA (%I0.1 para el contador 0 y %I0.7 para el contador 1). Los pulsos de contaje son tomados en cuenta solo si la entrada IN se encuentra en estado “1”. Al activarse la entrada S, el valor actual del contador (%VFCi.V) regresa a cero, esta entrada se activa en estado 1.
De modo opcional, se pueden emplear dos entradas específicas auxiliares, y las funciones que tienen asignadas son:

- Entrada de preselección, copia el valor de preselección (%VFCi.P) en el valor actual (%VFCi.V).
- Entrada rápida: Obliga al controlador a actualizar el valor actual (%VFCi.V).

La entrada IB no se emplea con esta configuración.

11.3.1.4 Contador Muy Rápido, Función Conteo Regresivo.

Con esta configuración, la única diferencia con el caso anterior, es el funcionamiento de la entrada S. En este caso, cuando se activa la entrada S, el valor actual del contador (%VFCi.V) copia el valor de preselección (%VFCi.P), esta entrada se activa en estado 1.

11.3.1.5 Contador Muy Rápido, Función Conteo Progresivo / Regresivo.

Puede configurarse al Twido de dos formas distintas para realizar esta función. Una es llamada Progresivo / Regresivo, y la restante Progresivo / Regresivo bifásico.
Lo que tienen en común ambas son las siguientes funciones:

a) Los pulsos de contaje / descontaje son tomados en cuenta solo si la entrada IN se encuentra en estado “1”.
b) Al activarse la entrada S, en el valor actual del contador (%VFCi.V) se copia el valor guardado en %VFCi.P
c) De modo opcional, se pueden emplear dos entradas específicas auxiliares, y las funciones que tienen asignadas son:
   - Entrada de preselección, copia el valor de preselección (%VFCi.P) en el valor actual (%VFCi.V).
   - Entrada rápida: Obliga al controlador a actualizar el valor actual (%VFCi.V).

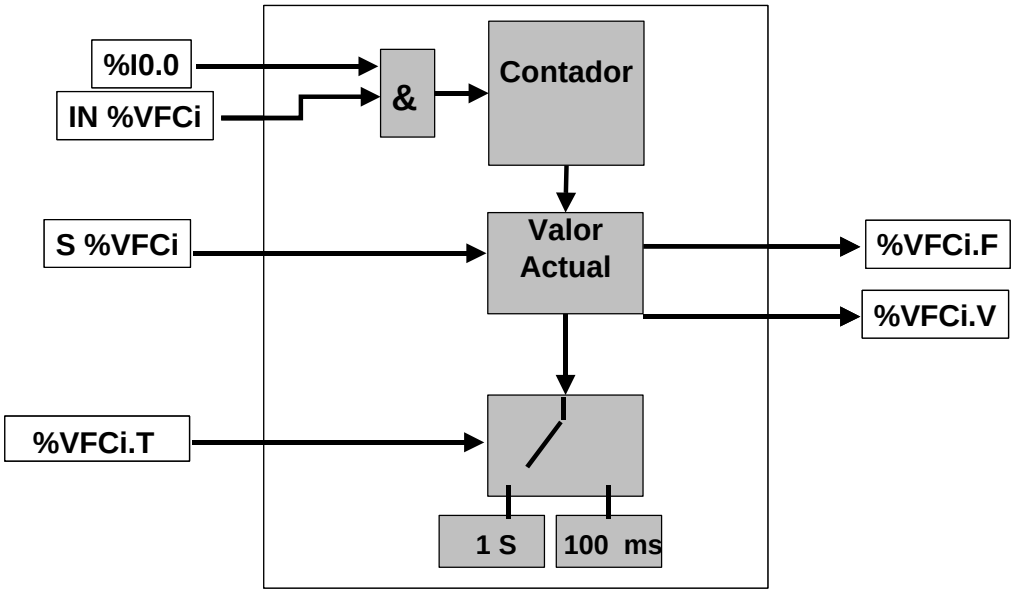
La diferencia está dada por el funcionamiento de las dos entradas específicas principales:

a) Contador Progresivo / Regresivo:
   La entrada IA (%I0.1 para el contador 0 y %I0.7 para el contador 1), recibe los pulsos a contar o descontar, en tanto que la entrada IB (%i0.0 para el 0 y %I0.6 para el 1) define el sentido de la cuenta, siendo 1 = Progresivo, 0 = Regresivo.
b) Contador Progresivo / Regresivo Bifásico:
   La entrada IA (%I0.1 para el contador 0 y %I0.7 para el contador 1), recibe los pulsos a contar, en tanto que la entrada IB (%i0.0 para el 0 y %I0.6), recibe los pulsos a descontar.

11.3.2 Frecuencímetro.

Cuando a un contador muy rápido se lo configura en modo frecuencímetro, el valor actual del mismo (%VFCi.V), devuelve el valor de la frecuencia con que ingresan los pulsos en la entrada IA (%I0.1 para el contador 0 y %I0.7 para el contador 1) y las entradas específicas restantes no se emplean.
En cuanto a las entradas del bloque, solo se emplea la entrada IN, y su función es habilitar el funcionamiento del frecuencímetro.

11.3.2.1 Diagrama en bloques.



12.- REGULACIÓN.

12.1 Introducción.

Los autómatas Twido modulares ofrecen dos módulos de regulación, los cuales pueden configurarse como PLS (Tren de pulsos), o PWM (Modulación por ancho de pulsos). Estos pueden servir por ejemplo para: Control de un motor paso a paso, y hacer control adaptado con una salida TON respectivamente. Estos bloque tienen asociados los siguientes objetos bit y word.

Bloque	Tipo	Descripción	Dirección	Escritura
PWM	Palabra	% del impulso a 1 en relación al período total.	%PWMi.R	Si
		Valor de preselección del período	%PWMi.P	No
PLS	Palabra	Valor de preselección	%PLSi.P	Si
		Nº de impulsos a generar	%PLSi.N	Si
	Bit	Salida en curso	%PLSi.Q	No
		Salida generación terminada	%PLSi.D	No

Cave aclarar que estas funciones no existen en las CPU Twido compactas.
Estos bloques emplean salidas específicas del PLC, siendo las mismas:

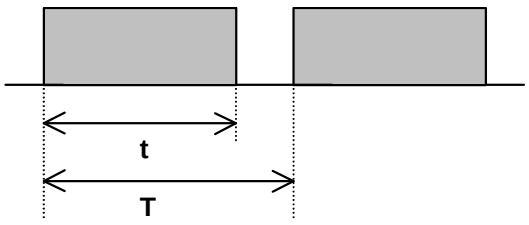
- %Q0.0 para el bloque cero
- %Q0.1 para el bloque 1.

Los bloques PLS y PWM emplean las mismas salidas específicas, por lo tanto, se deberá elegir una de las dos funciones.

12.2 Salida de modulación de amplitud %PWM.

12.2.1 Descripción.

El bloque de función %PWMi permite generar en la salida específica (%Q0.0 para el bloque cero y %Q0.1 para el bloque 1), una señal rectangular, cuyo período (T) es constante, con la posibilidad de variar el tiempo en que permanece activa la misma (t). Los pulsos son generados si la entrada IN del bloque está en 1, en estado 0 queda inhibida la función PWM.



El valor del período T y el porcentaje de tiempo en que la señal está en 1 para cada período son definidos por configuración del bloque función.

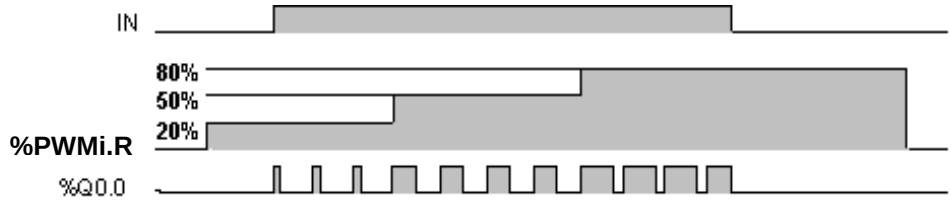
12.2.2 Parámetros configurables:

- **Definición del período**: T = BT x %PWMi.P
  BT = Base de tiempo
  0,142 ms (se aconseja usar solo en autómatas con salidas a transistor), 0,57 ms, 10ms ó 1 s (valor por defecto).
  %PWM.P = valor de preselección
  0 ≤ %PWMi.P ≤ 32767 con BT = 10ms o 1s
  0 ≤ %PWMi.P ≤ 255 con BT = 0,142 ms ó 0,57 ms.

- **Definición del tiempo t**: t = T x (%PWMi.R/100)
  %PWMi.R da el porcentaje de tiempo en que la señal está en 1 para cada período (0≤%PWM.R≤100)

12.2.3 Funcionamiento.

La frecuencia de la señal se fija en configuración mediante la selección de la base de tiempo BT y la preselección %PWMi.P. La modulación del ancho del pulso se obtiene modificando por programa el parámetro %PWMi.R. En la figura siguiente se puede observar como evoluciona la salida %Q0.0 (Bloque cero), en función del parámetro %PWMi.R.

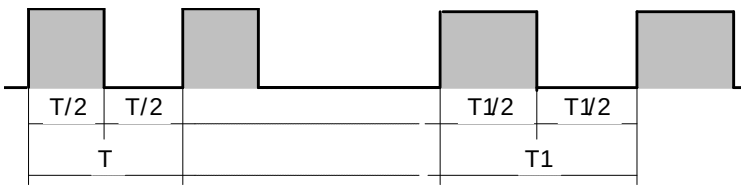


Un ejemplo de aplicación podría ser regulación de intensidad lumínica o regulación de temperatura.

12.3 Salida del generador de impulsos %PLS.

12.3.1 Descripción.

El bloque de función %PLSi permite generar en la salida específica (%Q0.0 para el bloque cero y %Q0.1 para el bloque 1), de una señal de periodo variable, manteniendo la relación del 50% en el tiempo de actividad de la señal.



12.3.2 Parámetros configurables:

- **Definición del período**: T = BT x %PLSi.P
  BT = Base de tiempo
  0,142 ms (se aconseja usar solo en autómatas con salidas a transistor), 0,57 ms, 10ms ó 1 s (valor por defecto).
  %PLSi.P = valor de preselección
  0 ≤ %PLSi.P ≤ 32767 con BT = 10ms o 1s
  0 ≤ %PLSi.P ≤ 255 con BT = 0,142 ms ó 0,57 ms.

- **Definición del número de impulsos: %PLSi.N**
  El número de impulsos de período T a generar (%PLSi.N) puede ser limitado o ilimitado según como se lo defina por configuración:
  0 < %PLS.N ≤ 32767
  %PLS.N = 0 generación ilimitada

Un ejemplo de aplicación podría ser el comando de un driver para motores pasa a paso.

13.- COMUNICACIÓN.

13.1 Introducción.

Cada vez son más las aplicaciones en las que se requiere que el autómata se comunique, ya sea para enviar datos a una estación central, para realizar automatismo distribuido, etc. Esta necesidad de "comunicación" es esencial para conseguir automatizar procesos. El autómata Twido ofrece varias formas de comunicación, lo cual le permite conectarse con otros equipos similares, con autómatas de mayor capacidad y con otros dispositivos, como por ejemplo terminales de diálogo, sistemas de supervisión, impresoras, etc.
Las distintas posibilidades de comunicación que ofrece el Twido son:

☑ Conexión remota.

☑ Comunicación MODBUS.

☑ Comunicación ASCII.

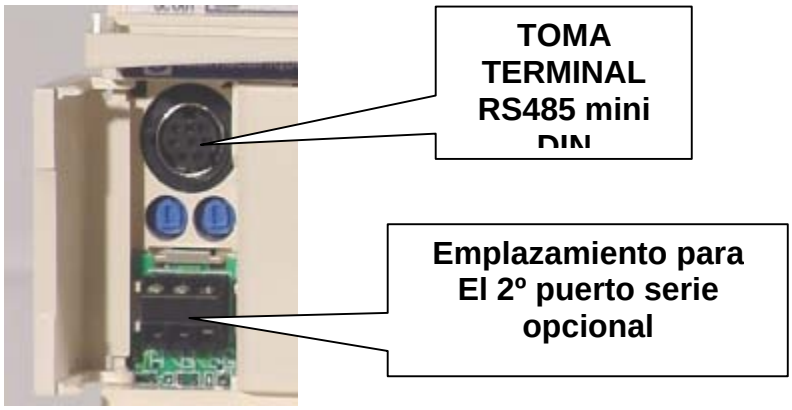
13.2 Puertos de comunicación.

Todos los autómatas Twido ofrecen de base un puerto de comunicaciones con conexión mini DIN y con soporte físico en RS485. Este puerto puede configurarse en cualquiera de los 3 protocolos mencionados en el punto anterior, y además es el que se emplea para programar al autómata.
En los modelos compactos de 16 o 24 entradas / salidas, y en los modelos modulares, puede agregarse un segundo puerto de comunicación opcional. A continuación se detallaran los accesorios que se pueden emplear en cada caso. En todos los casos, se debe tener presente que este segundo puerto **NO** es apto para programar al autómata.

13.2.1 Twido Compacto.

El segundo puerto solo puede colocarse en los modelos compactos de 16 o 24 entradas / salidas. Existen tres posibles accesorios a colocar:

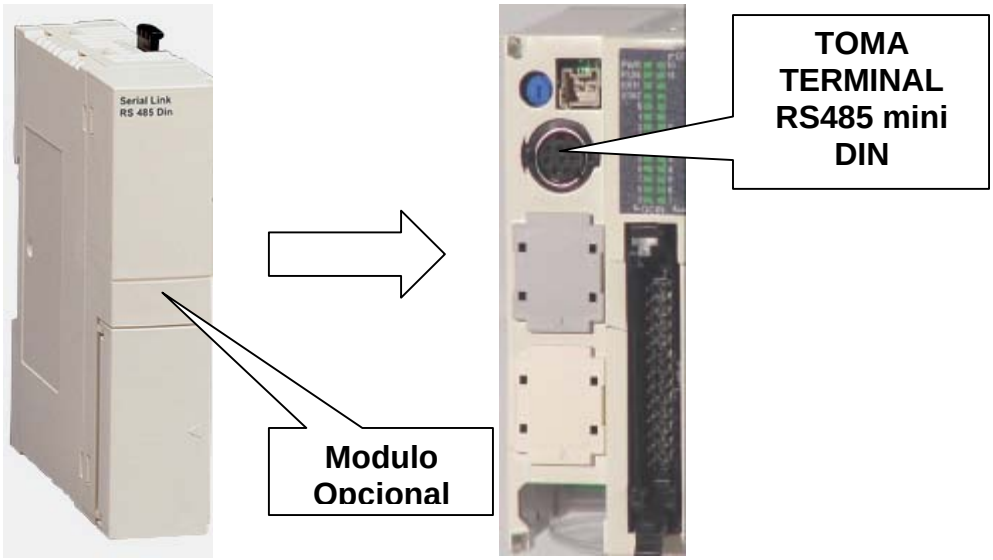
- TWDNAC232D: RS232, formato mini DIN
- TWDNAC485D: RS485, formato mini DIN
- TWDNAC485T: RS485, formato bornera.



13.2.2 Twido Modular.

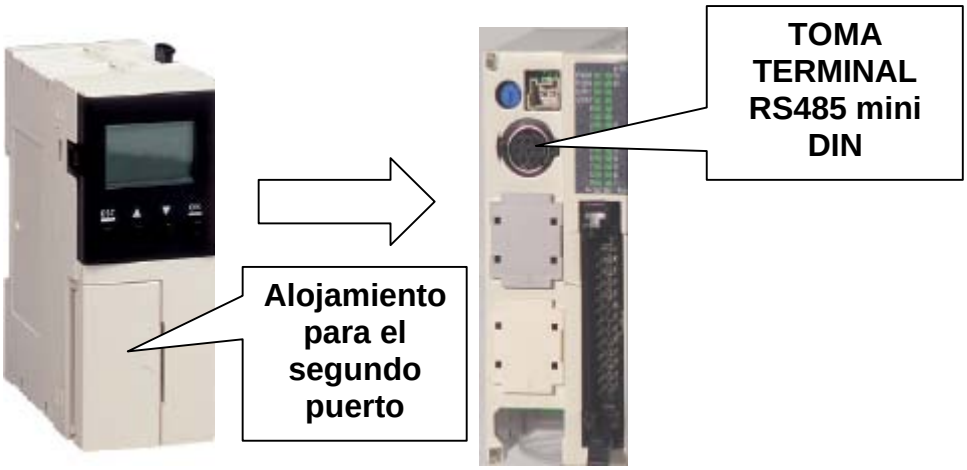
El segundo puerto puede colocarse en todos los modelos modulares, y requiere de un módulo que se coloca a la izquierda de la CPU. Los módulos que pueden emplearse son:

- TWDNOZ232D: RS232, formato mini DIN.
- TWDNOZ485D: RS485, formato mini DIN.
- TWDNOZ485T: RS485, formato bornera.



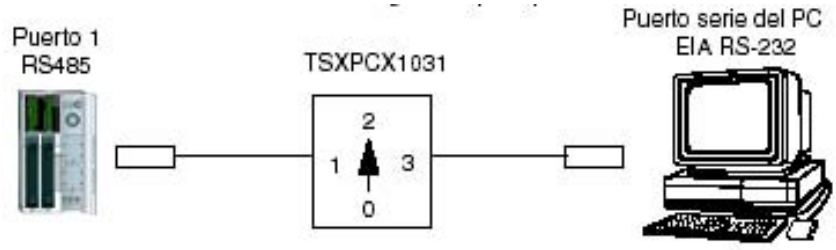
En los autómatas modulares existe además, una segunda alternativa para incorporar el segundo puerto serie. Esta alternativa permite incorporar como puerto serie los mismos accesorios empleados en los autómatas compactos, pero requiere del agregado de un módulo de visualización **(TWDXCPODM**), y dentro del mismo se incorpora uno de los accesorios:

- TWDNAC232D: RS232, formato mini DIN
- TWDNAC485D: RS485, formato mini DIN
- TWDNAC485T: RS485, formato bornera.



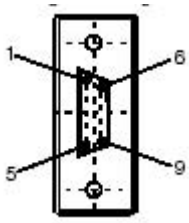
13.3 Comunicación con TwidoSoft.

Cada controlador Twido tiene en su puerto 1 un puerto mini DIN RS-485 incorporado con fuente de alimentación interna. Debe utilizar el puerto 1 para comunicarse con el paquete de programación de TwidoSoft. No puede utilizarse ningún cartucho opcional o módulo de comunicaciones para esta conexión.
El puerto RS-232C de su PC está conectado al puerto 1 del controlador tilizando el cable de comunicaciones con varias funciones TSXPCX1031. Este cable convierte las señales de RS-232 a RS-485 y viceversa. Este cable está equipado con un conmutador giratorio de 4 posiciones para seleccionar diferentes modos de funcionamiento. El conmutador designa las cuatro posiciones como "0-3" y el ajuste apropiado para conectarse con TwidoSoft es la ubicación 2. Esta conexión se ilustra en el diagrama que aparece a continuación.



La descripción de las funciones de cada pin de los conectores del TSXPCX1031 son los siguientes.

Conector Mini din macho de 8 pines:
1: A(+), 2: B(-), 3: NC, 4: /DE, 5: DPT, 6: NC, 7: 0V, 8: 5V.



Conector subD hembra de 9 pines
1: DCD, 2: RX, 3: TX, 4: DTR, 5: SG, 6: NC, 7: RTS, 8: CTS, 9: NC

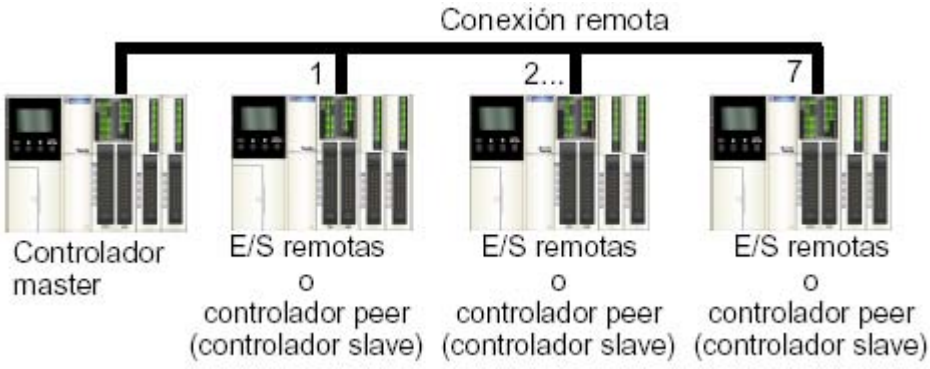
13.4 Conexión remota.

El protocolo de conexión remota es un bus master / slave de alta velocidad diseñado para transferir una pequeña cantidad de datos entre el controlador master y hasta siete controladores remotos (slave). Se transfieren datos de E/S o de aplicación dependiendo de la configuración de los controladores remotos. Es posible realizar una mezcla de varios tipos de controladores remotos, donde unos pueden ser E/S remotas y otros, controladores peer. El protocolo y el bus de E/S utilizados están patentados y

no se permite utilizar dispositivos de otros fabricantes en la red, solo es válido para autómatas Twido.
Para emplear este tipo de conexión se deben tener en cuenta las siguientes limitaciones:

- Asegúrese de que sólo exista un controlador master en una conexión remota.
- Asegúrese de que todos los slaves tengan direcciones exclusivas.
- La conexión remota requiere una conexión EIA RS-485 y sólo puede ejecutarse en un puerto de comunicaciones cada vez.

Como se dijo anteriormente, la conexión remota le permite al maestro de la red tener hasta 7 esclavos. Cada uno de ellos puede funcionar como **E/S Remotas** o como **controlador Peer**.



NOTA: Es importante tener presente que todos los modelos de CPU´s Twido pueden ser configurados como maestro o como esclavo.

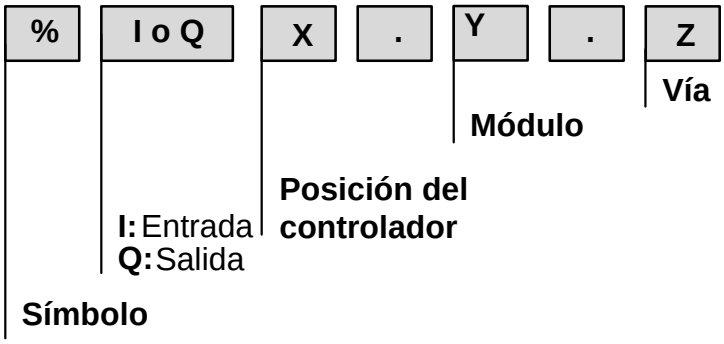
El cableado a realizar en los autómatas es el mostrado en la figura siguiente.

Conectar los cables de señal D(+) y D(-) juntos. La conexión de la señal DPT a tierra es necesaria solo si la conexión es por el puerto 1 del Twido (Toma integrada), pero, aunque no es necesario, es recomendable poner a tierra esta señal para utilizarla con una conexión remota en el puerto 2 (el cartucho opcional o el módulo de comunicación).

La información referida al estado de la comunicación se encuentra en los bits de sistema **%S100,** y **%S110 a %S113**, y en las palabras de sistema **%SW111 a %SW113** Para obtener información sobre estos objetos de sistema rogamos consultar en la ayuda de TwidoSoft, o en la guía de referencia de software.

13.4.1 Esclavos funcionando como E/S Remotas.

Cuando un autómata Twido se emplea como E/S remotas, el mismo no ejecuta ningún programa, y solamente se usan sus entradas y salidas. Las entradas y salidas de este autómata, son usadas por el maestro como propias, pero remotas.
Para direccionar las entradas y las salidas del autómata remoto, en el maestro se debe emplear el direccionamiento con tres dígitos explicado en el capítulo 5, y que se repite a continuación.

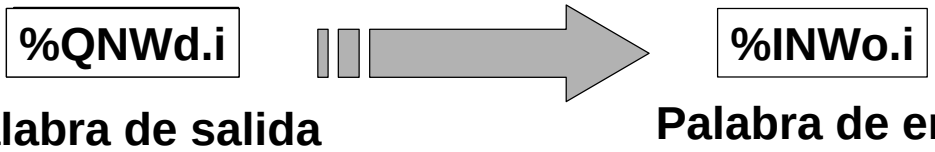


Símbolo: IEC61131
Tipo de objeto: %I: Entradas, %Q: Salidas.
X. Posición del controlador: 0 Controlador master, 1 a 7 controlador remoto.
Y. Módulo: 0 unidad de E/S local, 1 a 7 módulos de ampliación.
Z. Vía, número de la entrada o salida

Donde X es el número de esclavo asignado, Y solo puede tomar el valor 0 (solo se reportan las entradas y salidas de la base, no es posible hacerlo con las extensiones) y z es el número de la entrada o la salida.
Este funcionamiento es utilizable solamente para las entradas y salidas de tipo **digitales**.

13.4.2 Esclavos funcionando como Controlador Peer.

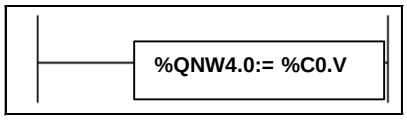
Cuando un esclavo se define como Controlador Peer, puede ejecutar su propio programa, y compartir datos con el maestro. La cantidad de datos de tipo palabra que se pueden emplear son 4 desde el maestro hacia cada uno de los esclavos y otras cuatro entre cada uno de los esclavos y el maestro. Solo es posible enviar datos entre el maestro y un esclavo y, si fuese necesario enviar un dato desde un esclavo a otro, el mismo debe ser enviado por el esclavo hacia el maestro, y este lo envía hacia el otro esclavo.
Para comunicarse con los controladores peer, el master utiliza las palabras de intercambio **%INW y %QNW**. Debe accederse a cada peer de la red mediante su dirección remota "j" utilizando las palabras %INWj.k y %QNWj.k. Cada controlador peer de la red utiliza %INW0.0 a %INW0.3 y %QNW0.0 a %QNW0.3
para acceder a los datos del master. Las palabras de red se actualizan de forma automática cuando el controlador está en modo de ejecución o detenido.
La figura siguiente ilustra el método de intercambio.



d: Destino, 0 a 7, identifica la dirección del controlador a donde va dirigido el dato, 0 para el maestro.
o: Origen, 0 a 7, identifica la dirección desde donde viene el mensaje, 0 para el maestro.
i: Numero de palabra, 0 a 3, identifica la palabra usada para el intercambio, por ejemplo, un dato guardado por el controlador **3** en la palabra **%QNW0.2**, está dirigida al maestro (dirección de destino **0**). El maestro recibe ese mismo dato en la palabra **%INW3.2**, pues tiene origen en el esclavo 3.

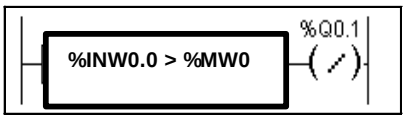
Ejemplo:
El autómata maestro debe transmitir al controlador peer nº 4 el valor actual del contador 0. Cuando este valor actual supera al umbral contenido en la palabra %MW0 el controlador peer debe detener una máquina, desactivando la salida %Q0.1

Programación del maestro.




```
LD 1
[%QNW4.0 := %C0.V]
```

Programación del controlador 4.




```
LD [%INW0.0 > %MW0]
STN %Q0.1
```

13.5 Comunicaciones ASCII.

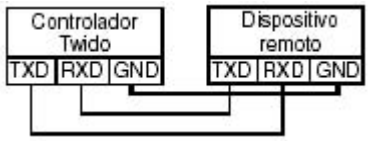
El protocolo ASCII proporciona a los controladores Twido un protocolo simple de modo de caracteres semi-dúplex que se utiliza para transmitir y/o recibir una cadena de caracteres hacia/desde un dispositivo simple (impresora o terminal). Este protocolo sólo se admite a través de la instrucción "EXCHx" y se controla mediante el bloque de función %MSGx.
Hay tres tipos de comunicaciones posibles utilizando el protocolo ASCII:

- Sólo transmisión
- Transmisión/Recepción
- Sólo recepción

El tamaño máximo de las tramas transmitidas o recibidas mediante la instrucción EXCHx es de 128 bytes.
Una conexión ASCII se puede establecer en el puerto RS-232 o RS-485 y se puede ejecutar hasta en dos puertos de comunicaciones al mismo tiempo. Este protocolo se puede configurar en todos los puerto de Twido (integrado o accesorio).

A continuación, se ilustran las conexiones de cableado nominal para los tipos RS-232 y RS-485.

Cable RS 232



Cable RS 485

Nota: Si se utiliza el puerto 1 en el controlador Twido, la señal DPT deberá estar conectada a tierra. Esto indica al controlador Twido que la comunicación a través del puerto 1 es ASCII (previa configuración del software) y no el protocolo utilizado para comunicarse con el Twido Soft.

13.5.1 Configuración del búfer de transmisión/recepción para ASCII.

El tamaño máximo de las tramas transmitidas o recibidas es 128 bytes, y la tabla de palabras asociada a la instrucción EXCHx está compuesta por tablas de transmisión y de recepción. El formato de dicha tabla será el siguiente:

	Byte de mayor valor	**Byte de menor valor**
Palabras de control	Comando	Longitud (Tx/Rx)
	Reservado (0)	Reservado (0)
Tabla de transmisión	Byte 1 transmitido	Byte 2 transmitido
	... ...	... ...
	... ...	Byte n transmitido
	Byte n+1 transmitido	
Tabla de recepción	Byte 1 recibido	Byte 2 recibido
	... ...	... ...
	... ...	Byte p recibido
	Byte p+1 recibido	

Palabras de control
El byte de **longitud** contiene la longitud que se va a transmitir, sobrescrita por el número de caracteres recibidos al final de la recepción, en caso de que ésta se solicite.
El byte de **comando** debe contener uno de los valores siguientes:

- 0: Sólo transmisión
- 1: Transmisión/Recepción
- 2: Sólo recepción

Tablas de transmisión/recepción

Cuando está activo el modo sólo transmisión, las tablas de transmisión y control se completan antes de ejecutar la instrucción EXCHx, y pueden ser del tipo %KW o %MW. No se requiere ningún espacio para la recepción de caracteres en el modo sólo

transmisión. Una vez transmitidos todos los bytes, el estado de %MSGx.D se pone a 1 y se puede ejecutar una instrucción EXCHx nueva.
Cuando está activo el modo Transmisión/Recepción, las tablas de transmisión y control se completan antes de ejecutar la instrucción EXCHx, y deben ser del tipo %MW. Se requiere espacio para hasta 128 bytes de recepción al final de la tabla de transmisión. Una vez transmitidos todos los bytes, el controlador Twido cambia a modo de recepción y espera a recibir los bytes.
Cuando está activo el modo sólo recepción, la tabla de control se completa antes de ejecutar la instrucción EXCHx y debe ser del tipo %MW. Se requiere espacio para hasta 128 bytes de recepción al final de la tabla de control. El controlador Twido cambia inmediatamente a modo de recepción y espera a recibir los bytes. La recepción concluye cuando se recibe el byte de final de trama o cuando la tabla de recepción está llena. Si se configura un timeout que no sea cero, la recepción concluye cuando el timeout se completa. Si se selecciona un valor de timeout cero, no hay timeout de recepción. Por lo tanto, para detener la recepción hay que activar la entrada %MSGx.R.
No hay direccionamiento inherente asociado con el protocolo ASCII a menos que el dispositivo simple lo tenga incorporado en el protocolo. No obstante, el controlador Twido no lo admite.

Intercambio de mensajes.

El controlador Twido puede configurarse para enviar o recibir mensajes en modo carácter. El lenguaje ofrece dos servicios para esto:

- **Instrucción EXCHx:** para transmitir/recibir mensajes
- **Bloque de función %MSGx:** para controlar los intercambios de mensajes.

Cuando se procesa una instrucción EXCHx, el controlador Twido utiliza el protocolo configurado para dicho puerto.

Nota: Cada puerto de comunicaciones puede configurarse para protocolos diferentes o para el mismo. El modo de acceder a la instrucción EXCHx o al bloque de función %MSGx para cada puerto de comunicaciones es agregando el número de puerto (1 ó 2).

13.6 Comunicaciones Modbus

El protocolo Modbus es un protocolo maestro/esclavo que permite a un maestro, y sólo a uno, pedir respuestas de los esclavos o realizar acciones dependiendo de las peticiones. El maestro puede dirigirse a los esclavos individualmente o iniciar una difusión de mensajes para todos. Los esclavos devuelven un mensaje (respuesta) a las peticiones que se les envían individualmente. No se devuelven respuestas a las peticiones de difusión.
Una conexión Modbus puede establecerse en el puerto RS-232 o RS-485 y puede ejecutarse hasta en dos puertos de comunicaciones al mismo tiempo.
El cableado necesario es el mismo que se indicó en el punto 13.5 para el protocolo ASCII.

13.6.1 Maestro Modbus

El modo maestro de Modbus permite al controlador iniciar una transmisión de peticiones Modbus, esperando una respuesta desde un esclavo Modbus. El modo maestro de Modbus sólo se admite a través de la instrucción EXCHx y admite los formatos Modbus ASCII y Modbus RTU.
El tamaño máximo de las tramas transmitidas o recibidas es 128 bytes y la tabla de palabras asociada a la instrucción EXCHx está compuesta por tablas de transmisión y recepción.

	Byte de mayor valor	**Byte de menor valor**
Palabras de control	Comando	Longitud (Tx/Rx)
	RX Offset	Tx FOCET
Tabla de transmisión	Byte 1 transmitido	Byte 2 transmitido
	... ...	... ...
	... ...	Byte n transmitido
	Byte n+1 transmitido	
Tabla de recepción	Byte 1 recibido	Byte 2 recibido
	... ...	... ...
	... ...	Byte p recibido
	Byte p+1 recibido	

Palabras de control

El byte de longitud contiene la longitud que se va a transmitir, sobrescrita por el número de caracteres recibidos al final de la recepción, en caso de que ésta se solicite. Este parámetro es la longitud en bytes de la tabla de transmisión. Si el parámetro Tx Offset es igual a 0, este parámetro será igual que la propia longitud de trama menos 2 bytes CRC. Si el parámetro Tx Offset no es igual a 0, no se transmitirá un byte del búfer (indicado por el valor de offset) y este parámetro será igual a la propia longitud de trama más 1.
El byte de comando, en caso de que se produzca una solicitud Modbus RTU (excepto para la difusión), debe ser siempre igual a 1 (Tx y Rx). El byte Tx Offset contiene el offset (1 para el primer byte, 2 para el segundo byte, etc.) dentro de la tabla de transmisión que se ignorará cuando se transmita el paquete. Esto se utiliza para gestionar los problemas asociados a los valores de bytes/palabras del protocolo Modbus. Por ejemplo, si este byte contiene 3, el tercer byte se ignorará, haciendo que el cuarto byte de la tabla sea el tercero en transmitirse.
El byte Rx Offset contiene el offset (1 para el primer byte, 2 para el segundo byte, etc.) dentro de la tabla de recepción que se agregará cuando se transmita el paquete. Esto se utiliza para gestionar los problemas asociados a los valores de bytes/palabras del protocolo Modbus. Por ejemplo, si este byte contiene 3, el tercer byte de la tabla se completará con un cero y el tercer byte recibido se introducirá en la cuarta ubicación de la tabla.

Tablas de transmisión/recepción.

Cuando se utiliza cualquier modo (Modbus ASCII ó Modbus RTU Modbus), la tabla de

transmisión se completará con la solicitud previa a la ejecución de la instrucción EXCHx. En el momento de la ejecución, el controlador determina qué es la capa de enlace de datos y realiza todas las conversiones necesarias para procesar la transmisión y la respuesta. Los caracteres de inicio, fin y comprobación no se almacenan en las tablas de transmisión/recepción.
Una vez transmitidos todos los bytes, el controlador cambia a modo de recepción y espera a recibir los bytes. La recepción se completa de una de estas formas:

- el carácter de final de trama se recibe en modo ASCII
- se detecta el timeout de un carácter o trama
- la tabla de recepción está llena.

Las entradas de **byte X transmitido** contienen los datos del protocolo Modbus (codificación RTU) que se va a transmitir. Si el puerto de comunicaciones está configurado para ASCII Modbus, los caracteres de trama correctos se agregan a la transmisión. El primer byte contiene la dirección de dispositivo (específica o difusión), el segundo byte contiene el código de función y el resto contienen información asociada al código de función.

Nota: Ésta es una aplicación típica, pero no define todas las posibilidades. No se realizará ninguna validación de los datos que se están transmitiendo.

Las entradas de **byte X recibido** contienen los datos del protocolo Modbus (codificación RTU) que se va a recibir. Si el puerto de comunicaciones está configurado para ASCII Modbus, los caracteres de trama correctos se eliminan de la respuesta. El primer byte contiene la dirección de dispositivo, el segundo byte contiene el código de función (o código de respuesta) y el resto contienen información asociada al código de función.

Nota: Ésta es una aplicación típica, pero no define todas las posibilidades. No se realizará ninguna validación de los datos que se están recibiendo, excepto para la verificación de la suma de control.

13.6.2 Esclavo Modbus

El modo esclavo Modbus permite al controlador responder a las solicitudes de Modbus del maestro. El controlador admite los datos Modbus estándar y las funciones de control, así como las ampliaciones UMAS para el acceso a objetos y la configuración.
El protocolo Modbus admite dos formatos de capa de enlace de datos: ASCII y RTU. Cada uno está definido por la implementación de la capa física: ASCII utiliza 7 bits de datos y RTU utiliza 8 bits de datos.
Cuando se utiliza el modo Modbus ASCII, cada byte del mensaje se envía como dos caracteres ASCII. La trama Modbus ASCII comienza con un carácter inicial (':') y finaliza con dos caracteres finales (CR y LF). El carácter de final de trama se establece de forma predeterminada como 0x0A (avance de línea) y el usuario puede modificar el valor de este byte durante la configuración. El valor de comprobación para la trama Modbus ASCII es un complemento de dos de la trama, excluyendo los caracteres inicial y final.

El modo Modbus RTU no vuelve a formatear el mensaje antes de transmitirlo; sin embargo, utiliza un modo de cálculo de suma de verificación diferente, especificado como CRC.
La capa de enlace de datos de Modbus tiene las siguientes limitaciones:

- Dirección 1-247
- Bits: 128 bits al realizar la solicitud utilizando solicitudes abiertas de Modbus
- Palabras: 64 palabras de 16 bits al realizar la solicitud utilizando solicitudes abiertas de Modbus.

Para información sobre los códigos de peticiones estándar de Modbus, sirvase consultar la ayuda de TwidoSoft o la Guía de referencia de Software.

13.7 Instrucción EXCHx

La instrucción EXCHx permite al controlador Twido enviar o recibir información dirigida a, o procedente de, dispositivos ASCII. El usuario define una tabla de palabras %MWi:L o %KWi:L) que contiene información de control y los datos que se van a enviar o recibir (hasta 64 palabras en la transmisión o recepción). El formato de la tabla se describe en la sección anterior.
Un intercambio de mensajes se realiza utilizando la instrucción EXCHx. El controlador Twido debe finalizar el intercambio desde la primera instrucción EXCHx antes de que se ejecute una segunda. El bloque de función %MSGx debe utilizarse cuando se nvíen varios mensajes. El procesamiento de la instrucción de lista EXCHx se produce inmediatamente, con cualquier transmisión iniciada bajo el control de interrupción (la recepción de datos también se encuentra bajo el control de interrupción), que se considera procesamiento de fondo.

La Sintaxis a emplear es: [EXCHx %MWi:L] o [EXCHx %KWi:L]

donde: x = número de puerto (1 ó 2).
L = número de palabras en la tabla de palabras.

Si se produce un error durante el uso de una instrucción EXCHx, los bits %MSGx.D y %MSGx.E se ponen a 1 y la palabra de sistema %SW63 contiene el código de error para el puerto 1, y %SW64 contiene el código de error para el puerto 2. Para obtener información sobre estos objetos de sistema rogamos consultar en la ayuda de TwidoSoft, o en la guía de referencia de software.

13.8 Bloque de función %MSGx.

El uso del bloque de función %MSGx es opcional; puede utilizarse para gestionar los intercambios de datos. El bloque de función %MSGx tiene tres propósitos.

- **Comprobación de errores de comunicación.**
  La comprobación de errores verifica que la longitud del bloque (tabla de palabras) programada con la instrucción EXCHx es lo suficientemente grande para contener la longitud del mensaje que se va a enviar. Esto se compara con la longitud programada en el byte de menor valor de la primera palabra de la tabla de palabras.
- **Coordinación de varios mensajes**

Para asegurar la coordinación cuando se envíen varios mensajes, el bloque de función %MSGx proporciona la información requerida para determinar cuándo está completo un mensaje anterior.

- **Transmisión de mensajes prioritarios**

El bloque de función %MSGx permite la detención de la transmisión del mensaje actual para permitir el envío inmediato de un mensaje urgente.

El bloque de función %MSGx tiene una entrada y dos salidas asociadas.

Entrada/salida	Definición	Descripción
R	Restablecer entrada	Poner a 1: reinicializa la comunicación o restablece el bloque (%MSGx.E = 0 y %MSGx.D = 1).
%MSGx.D	Comunicación completa	0: solicitud en curso. 1: comunicación realizada si se produce el final de la transmisión, se recibe el carácter final, se produce un error o se restablece el bloque
%MSGx.E	Error	0: longitud del mensaje y enlace correctos. 1: si hay un comando inválido, la tabla se configura de forma incorrecta, se recibe un carácter incorrecto (velocidad, paridad, etc.) o la tabla de recepción está llena.

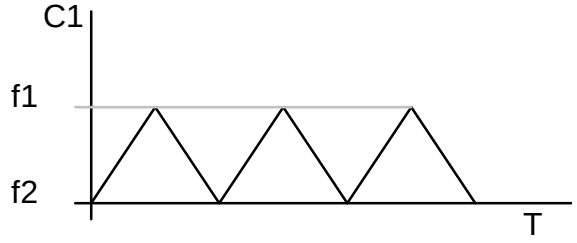
14.1 EJERCICIOS NIVEL 1

Ejercicio nº: 1

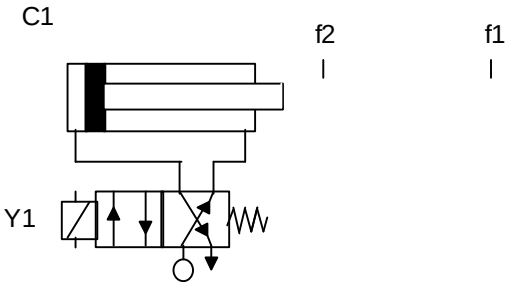
Realice el esquema de contactos y listado de instrucciones para comandar la marcha parada de un motor trifásico.
Posteriormente realice el esquema de contactos y listado de instrucciones para comandar la marcha parada de un motor trifásico utilizando las instrucciones SET y RESET.

Ejercicio nº: 2

Realice el esquema de contactos y listado de instrucciones para que el cilindro C1 describa el siguiente diagrama Espacio - Fase:



C1	Cilindro 1	
f1	Final de carrera de indicación de vástago del cilindro afuera	%I0.1
f2	Final de carrera de indicación de vástago del cilindro adentro	%I0.2
Pm	Pulsador de marcha	%I0.3
Pp	Pulsador de parada	%I0.4
Y1	electroimán de la electroválvula	%Q0.1



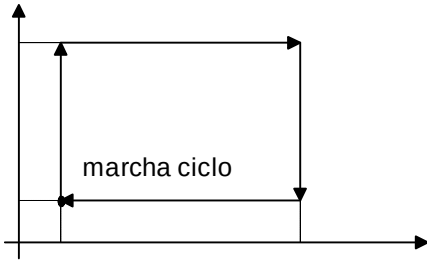
Ejercicio nº: 3

Realice el esquema de contactos y listado de instrucciones para comandar un montacarga con las siguientes especificaciones:

Pulsador ARRIBA	%I0.1
Pulsador ABAJO	%I0.2
Final de carrera CARRO ARRIBA	%I0.3
Final de carrera CARRO ABAJO	%I0.4
Salida SUBIR	%Q0.3
Salida BAJAR	%Q0.2

Ejercicio nº: 4

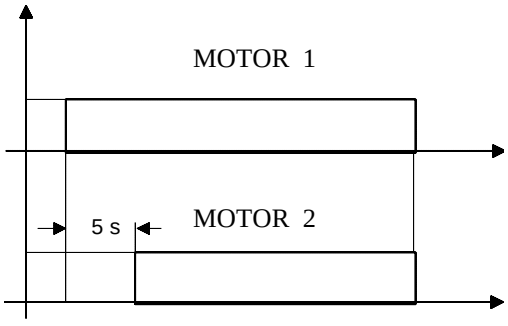
Realice el esquema de contactos y listado de instrucciones para comandar un carro que cumpla con el siguiente ciclo.



Pulsador MARCHA CICLO	%I0.1
Marcha IZQUIERDA	%Q0.0
Marcha DERECHA	%Q0.1
Marcha ABAJO	%Q0.2
Marcha ARRIBA	%Q0.3
Posición CARRO ABAJO	%I0.2
Posición CARRO ARRIBA	%I0.3
Posición CARRO IZQUIERDA	%I0.4
Posición CARRO DERECHA	%I0.5

Ejercicio nº: 5

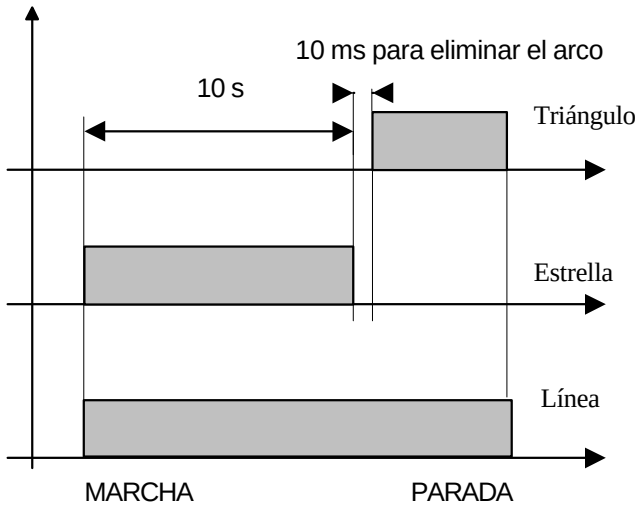
Realice el esquema de contactos y listado de instrucciones para comandar la MARCHA PARADA de dos motores en distinto tiempo.



Pulsador de MARCHA	%I0.1
Pulsador de PARADA	%I0.2
MOTOR 1	%Q0.0
MOTOR 2	%Q0.1

Ejercicio nº: 6

Realice el esquema de contactos y listado de instrucciones para comandar el siguiente arranque ESTRELLA-TRIÁNGULO.



Pulsador de marcha	%I0.1
Pulsador de parada	%I0.2
Línea	%Q0.0
Estrella	%Q0.1
Triángulo	%Q0.2

Ejercicio nº: 7

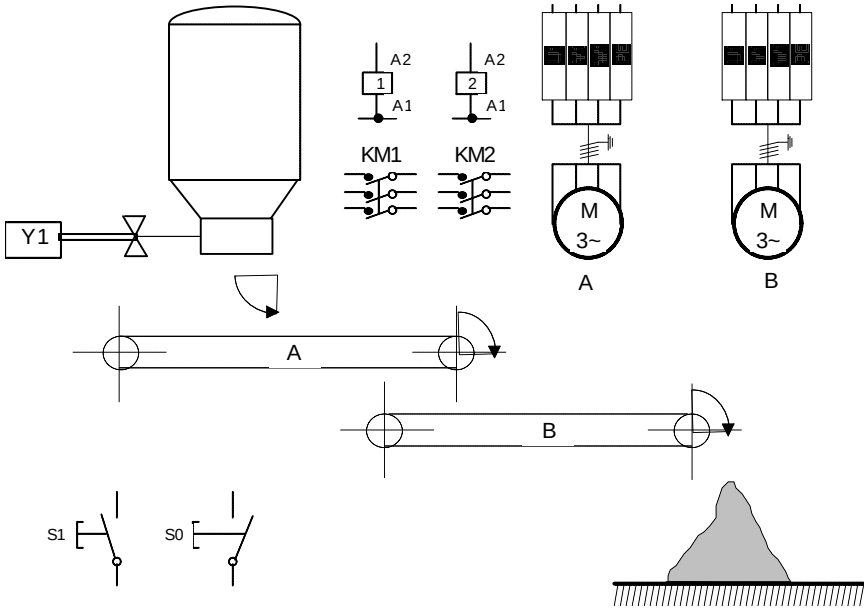
Realice el esquema de contactos y listado de instrucciones para controlar una playa de estacionamiento cuya capacidad máxima es de 12 automóviles, cuenta con 2 carteles indicadores de “HAY LUGAR” y “NO HAY LUGAR” que se activarán mediante las salidas %Q0.2 y %Q0.1 respectivamente. Además se cuenta con una barrera que se alzará durante un tiempo de 10 segundos ante la presencia de un automóvil. La salida es independiente de la entrada y no tiene barrera.

Detector de entrada	%I0.1
Detector de salida	%I0.2
Barrera	%Q0.3

Ejercicio nº: 8

Realice el esquema de contactos y listado de instrucciones para comandar la siguiente Tolva y Cintas Transportadoras.
Al pulsar S1 se inicia el movimiento de la cinta B. Una temporización de tres segundos permite que se active seguidamente la cinta A y la electroválvula Y1.
La detención del sistema se realiza de la siguiente manera:

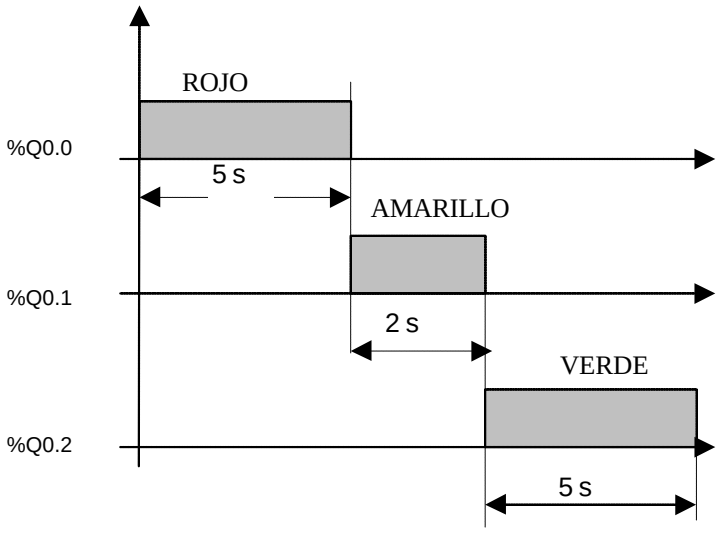
La electroválvula Y1 se corta por la activación de la parada S0.
Tras un tiempo de transporte de 10 seg. las cintas A y B se detienen simultáneamente.



Pulsador de MARCHA %IO.1
Pulsador de PARADA %I0.2
MOTOR B %Q0.0
MOTOR A %Q0.1
ELECTROVÁLVULA %Q0.2

Ejercicio nº: 9

Realice la siguiente secuencia de Semáforo:

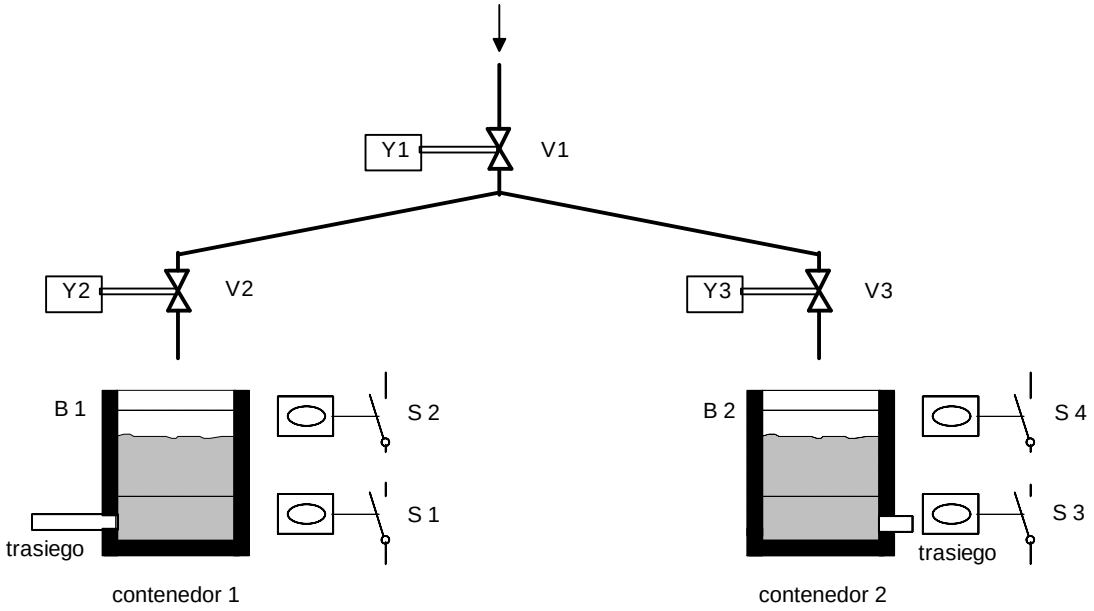


Ejercicio nº: 10

Se trata de realizar un control de contenedores. Los recipientes B1 y B2 (para líquidos) se llenan alternativamente. La operación de llenado se inicia por la señalización de vacío y se termina con la de lleno.
Ambas señales son proporcionadas por interruptor a flotador. Si al comienzo de la operación los dos recipientes están vacíos se llenará B1 en primer lugar. El sistema descrito se ilustra en la figura. Implementar el esquema de contactos y listado de instrucciones correspondiente si la operación de llenado se realiza de la siguiente forma:

La compuerta del contenedor V2 o V3 se abre instantáneamente en respuesta a la señal de llenado emitida.
Después de un tiempo to=2seg. se abre la compuerta principal V1.La terminación de la operación de llenado se realiza:

La válvula principal V1 se cierra instantáneamente
La válvula del contenedor correspondiente V2 o V3 se cierra después de un tiempo t1=4seg. ó t2=5seg.



Y1 %Q0.0
Y2 %Q0.1
Y3 %Q0.2
S1 %I0.1
S2 %I0.2
S3 %I0.3
S4 %I0.4

14.2 EJERCICIOS NIVEL 2.

Ejercicio nº: 11

Variar el porcentaje (de 0 a 100) del período de la salida PWM (%PWM0.R) utilizando un punto de reglaje analógico.
Utilizar una BT=10ms y %PWM0.P=100

Ejercicio nº: 12

Se requiere medir la velocidad angular a la que gira un motor en R.P.M., utilizando la entrada de contaje muy rápido en modo frecuencímetro. Además se desea visualizar dicha medición en el display de una XBT.
El motor lleva adosado un plato con 32 dientes, detectados por un sensor foto-eléctrico y tomados por el autómata a través de la entrada %I0.1.
Implementar el listado de instrucciones y diagrama de contactos correspondiente.
Nota: Para poder visualizar el dato en la terminal, se debe guardar en la palabra interna %MW10.

Ejercicio nº: 13

Se necesita enviar el valor de un punto de reglaje analógico y el segundero de un Twido que funciona como controlador peer número 4 al autómata base y visualizar ambos valores en una XBT (%MW0 y %MW2).
Implementar el listado de instrucciones y diagrama de contactos correspondiente.

Ejercicio nº: 14

Configurar el fechador número 6 de manera que accione la salida %Q0.2, cuando la fecha está comprendida entre el 21 de Julio y el 21 de Septiembre, solo los Lunes, Miércoles y Viernes, de 22:00 a 23:00 horas.
Luego de realizar esta configuración, probar el funcionamiento con la fecha y la hora actual.

Ejercicio nº: 15

Gobernar con la salida de generación de tren de impulsos un motor paso a paso. Los desplazamientos solicitados son los siguientes:

☑ Avanzar 10 puntos a una frecuencia de 1Hz. Entrada asignada: %I0.0.

☑ Avanzar 15 puntos a una frecuencia de 1Hz. Entrada asignada: %I0.1.

☑ Retroceder 25 puntos a una frecuencia de 50 Hz. Entrada asignada: %I0.2.

El avance se indica con la salida %Q0.3 puesta en 0.

14.3 RESOLUCIÓN DE EJERCICIOS.

Ejercicio nº: 1

Programa 1

```
0000    LD      %I0.1
0001    OR      %Q0.0
0002    ANDN    %I0.2
0003    ST      %Q0.0
0004    END
```

Programa 2

```
0000    LD      %I0.1
0001    S       %Q0.0
0002    LD      %I0.2
0003    R       %Q0.0
0004    END
```

Ejercicio nº: 2

```
0000    LD      %I0.2
0001    AND     %M0
0002    OR      %Q0.1
0003    OR      %I0.3
0004    ANDN    %I0.1
0005    ANDN    %I0.4
0006    ST      %Q0.1
0007    LD      %I0.3
0008    OR      %M0
0009    ANDN    %I0.4
0010    ST      %M0
0011    END
```

Ejercicio nº: 3

```
0000    LD      %I0.1
0001    AND     %I0.4
0002    OR      %Q0.3
0003    ANDN    %I0.3
0004    ST      %Q0.3
0005    LD      %I0.2
0006    AND     %I0.3
0007    OR      %Q0.2
0008    ANDN    %I0.4
0009    ST      %Q0.2
0010    END
```

Ejercicio nº: 4

```
0000    LD      %I0.1
0001    AND     %I0.4
0002    AND     %I0.2
0003    S       %Q0.3
0004    R       %Q0.0
0005    LD      %I0.3
0006    AND     %I0.4
0007    S       %Q0.1
0008    R       %Q0.3
0009    LD      %I0.3
0010    AND     %I0.5
0011    S       %Q0.2
0012    R       %Q0.1
0013    LD      %I0.5
0014    AND     %I0.2
0015    S       %Q0.0
0016    R       %Q0.2
0017    LD      %I0.4
0018    AND     %I0.2
0019    R       %Q0.0
0020    END
```

Ejercicio nº: 5

```
0000    LD      %I0.1
0001    S       %Q0.0
0002    BLK     %TM0
0003    LD      %Q0.0
0004    IN
0005    END_BLK
0006    LD      %TM0.Q
0007    S       %Q0.1
0008    LD      %I0.2
0009    R       %Q0.0
0010    R       %Q0.1
0011    END
```

Configuración

```
%TM0 :
BT      1s
PRESET  5
```

Ejercicio nº: 6

```
0000    LD      %I0.1
0001    ANDN    %Q0.2
0002    S       %Q0.0
0003    S       %Q0.1
0004    BLK     %TM0
0005    LD      %Q0.0
0006    IN
0007    END_BLK
0008    LD      %TM0.Q
0009    R       %Q0.1
0010    BLK     %TM1
0011    LD      %Q0.0
0012    ANDN    %Q0.1
0013    IN
0014    END_BLK
0015    LD      %TM1.Q
0016    ANDN    %Q0.1
0017    S       %Q0.2
0018    LD      %I0.2
0019    R       %Q0.0
0020    R       %Q0.2
0021    R       %Q0.1
0022    END
```

Configuración

```
%TM0 :
BT      1s
PRESET  10
```

```
%TM1 :
BT      10ms
```

Ejercicio nº: 7

```
0000    LDN     %C0.D
0001    AND     %I0.1
0002    S       %Q0.3
0003    BLK  %C0
0004    LDN     %C0.D
0005    AND     %I0.1
0006    CU
0007    LD      %I0.2
0008    CD
0009    END_BLK
0010    BLK     %TM0
0011    LD      %Q0.3
0012    IN
0013    END_BLK
0014    LD      TM0.Q
0015    R       %Q0.3
0016    LD      %C0.D
0017    ST      %Q0.1
0018    STN     %Q0.2
0019    END
```

Configuración

```
%TM0 :                          %C0 :
BT      1s
PRESET  10                      PRESET  12
```

Ejercicio nº: 8

```
0000    LD      %I0.1
0001    S       %Q0.0
0002    BLK     %TM0
0003    LD      %Q0.0
0004    IN
0005    END_BLK
0006    LD      %TM0.Q
0007    ST      %Q0.1
0008    LD      %TM0.Q
0009    ANDN    %M1
0010    ST      %Q0.2
0011    LD      %I0.2
0012    S       %M1
0013    R       %Q0.2
0014    BLK     %TM1
0015    LDN     %Q0.2
```

```
0016        AND         %Q0.0
0017        AND         %Q0.1
0018        IN
0019        END_BLK
0020        LD          %TM1.Q
0021        R           %Q0.0
0022        R           %Q0.1
0023        END
```

Configuración

%TM0 :		%TM1 :	
BT	1s	BT	1s
PRESET	3	PRESET	10

Ejercicio nº: 9

```
0000        LD          %I0.1
0001        S           %Q0.0
0002        BLK         %TM0
0003        LD          %Q0.0
0004        IN
0005        END_BLK
0006        LD          %TM0.Q
0007        S           %Q0.1
0008        R           %Q0.0
0009        BLK         %TM1
0010        LD          %Q0.1
0011        IN
0012        END_BLK
0013        LD          %TM1.Q
0014        S           %Q0.2
0015        R           %Q0.1
0016        BLK         %TM2
0017        LD          %Q0.2
0018        IN
0019        END_BLK
0020        LD          %TM2.Q
0021        R           %Q0.2
0022        S           %Q0.0
0023        LD          %I0.2
0024        R           %Q0.0
0025        R           %Q0.1
0026        R           %Q0.2
0027        END
```

Ejercicio nº: 10

```
0000    LD      %I0.1
0001    OR      %I0.2
0002    N
0003    S       %Q0.1
0004    BLK     %TM1
0005    LD      %I0.1
0006    AND     %I0.2
0007    IN
0008    END_BLK
0009    LD      %TM1.Q
0010    R       %Q0.1
0011    LD      %I0.3
0012    OR      %I0.4
0013    N
0014    ANDN    %Q0.1
0015    S       %Q0.2
0016    BLK     %TM2
0017    LD      %I0.3
0018    AND     %I0.4
0019    IN
0020    END_BLK
0021    LD      %TM2.Q
0022    R       %Q0.2
0023    BLK     %TM0
0024    LD      %Q0.1
0025    OR      %Q0.2
0026    IN
0027    END_BLK
0028    LD      %TM0.Q
0029    S       %Q0.0
0030    LD      %I0.1
0031    AND     %I0.2
0032    AND     %I0.3
0033    AND     %I0.4
0034    R       %Q0.0
0035    END
```

Configuración

%TM0 :		%TM1 :		%TM2 :	
BT	1s	BT	1s	BT	1s
PRESET	2	PRESET	4	PRESET	5

Ejercicio nº: 11

```
0000        LD          1
0001        [%MW1:=%iW0.0.0 * 25]
0002        LD          1
0003        [%MW0:=%MW1 / 1023]
0004        LD          1
0005        [%PWM0.R:= %MW0 * 4]
0006        BLK         %PWM0
0007        LD          %I0.1
0008        IN
0009        END_BLK
0010        END
```

Configuración

```
%Q0.0 :
BT          10 ms
%PWM0.P     10
```

Ejercicio nº: 12

```
0000        LD          1
0001        [%MW7:= %FC.V * 60]
0002        LD          1
0003        [%MW10:= %MW7 / 32]
0004        END
```

CONFIGURACIÓN CONTADOR

Modo: FRECUENCÍMETRO

Ejercicio nº: 13

Programa Autómata Base

```
0000        LD          1
0001        [ %MW0 := %INW4.0 ]
0002        LD          1
0003        [ %MW1 := %INW4.1]
0004        LD          1
0005        END
```

Programa Controlador Peer

```
0000        LD          1
0001        LD          1
0002        [ %QNW0.0 := %IW0.0.0 ]
0003        LD          1
0004        [ %QNW0.1 := %SW50 ]
0005        END
```

Ejercicio n°: 14

Ejercicio nº: 15

```
0000        LD          %I0.0
0001        [%PLS1.N := 10]
0002        [%PLS1.P := 100]
0003        S           %M0
0004        LD          %I0.1
0005        [%PLS1.N := 15]
0006        [%PLS1.P := 100]
0007        S           %M0
0008        LD          %I0.2
0009        [%PLS1.N := 25]
0010        [%PLS1.P := 2]
0011        S           %M0
0012        S           %Q0.3
0013        BLK         %PLS
0014        LD          %M0
0015        IN
```

```
0016        OUT_BLK
0017        LD          D
0018        R           %M0
0019        R           %Q0.3
0020        END_BLK
0021        END
```

Schneider Electric Argentina
http://www.schneider-electric.com.ar

0810-444-7246
Schneider On Line (SOL) Costo de llamada local.

Sede Central y Agencia Buenos Aires
Viamonte 2850
(B1678DWF) Caseros – Pcia de Buenos Aires
Tel. (54-11) 4716-8888 Fax (54-11) 4716-8822

En razón de la evolución de las normativas y del material, las características indicadas por el texto y las imágenes de este documento no nos comprometen hasta después de una confirmación por parte de nuestros servicios.

Planta Industrial San Martín
Av. 101 / Ricardo Balbín 3102/34
B1650NBN San Martín - Bs. As
Tel. (54 11) 4724-4444
Fax (54 11) 4724-4411

Planta Industrial Plasnavi
Héroes de Malvinas 2071/73
B1824CCE Lanús - Bs. As.
Tel. (54-11) 4246-7545_
Fax (54-11) 4246-5200

Agencia Córdoba
Bv. Los Granaderos 3163
5009 Córdoba,- Córdoba
Teléfono: (54 351) 481-6407
Fax: (54 351) 481-9235

Agencia Mendoza
San Martín 198 2°
M5501AAO Godoy Cruz - Mendoza
Tel. (54-261) 422-1110/4
Fax (54-261) 422-1119

Agencia Rosario
Cafferata 1130
S2002QXH Rosario - Santa Fé
Tel. (54-341) 430-0202
Fax (54-341) 430-0660

Delegación Bahía Blanca
Zelarrayán 151 1º 'E'
(8000) Bahía Blanca - Bs. As.
Tel/Fax: (54 291) 452-1567

Delegación Comodoro. Rivadavia
Ciudad Universitaria -Km 4
(9000) Comodoro Rivadavia, - Chubut
Tel/Fax: (54 297) 455-0836

Delegación Mar del Plata
Juan B. Justo 4302
(7608) Mar del Plata – Bs. As.
Tel.: (54 223) 481-6600 int.250
Fax: (54 223) 481-0046

Delegación Neuquén
Entre Ríos 531
8300 Neuquén - Neuquén
Telefax: (54 299) 448-8087

Delegación Paraná
Santiago del Estero 637 7ºD (3100)
Paraná, Pcia. de Entre Ríos
Tel. (0343) 15-611-0383

Delegación Pergamino
El Socorro 1475
2700 Pergamino - Bs. As.
Tel/Fax: (54 2477) 42-1239

Delegación Posadas
Av.Trincheras de San José 313
3300 Posadas - Misiones
Telefax: (54 3752) 43-8220

Delegación Salta
Sarmiento 213
4403 Cerrillos - Salta
Tel. (54 387) 490-2850

Delegación Tucumán
Ituzaingó 560
4107 Yerba Buena - Tucumán
Tel.: (54 381) 435-1103

Schneider Electric Uruguay
Ramón Masini 3190
11300 Montevideo - Uruguay
Tel.: (00 598 2) 707-2392 / 708-8237
Fax: (00 598 2) 707-2184