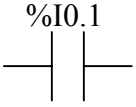
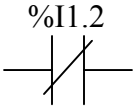
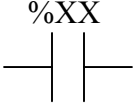
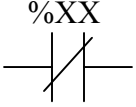
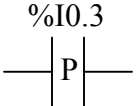
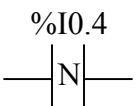
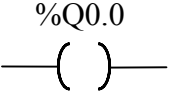
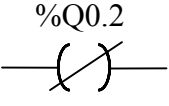
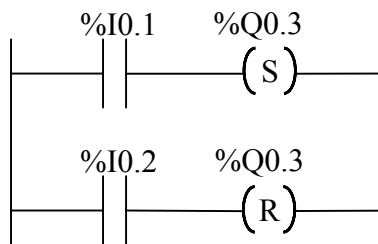


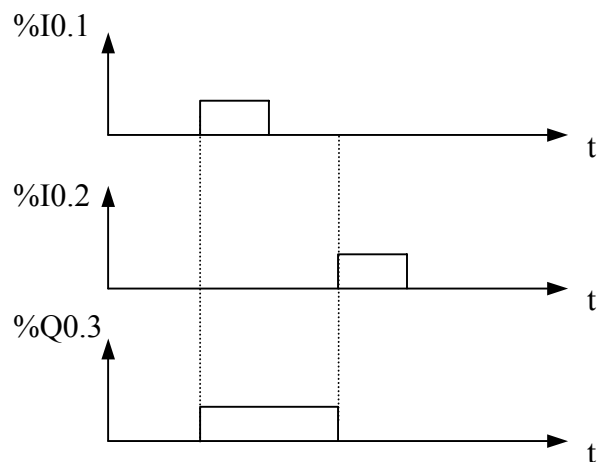
## INSTRUCCIONES BÁSICAS DEL PLC

- |  |  |
|--|--|
|   | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a un contacto NA que conducirá "corriente" (estará accionado) cuando la entrada 1 del PLC N° 0 se encuentra en estado 1.</li> </ul>   |
|   | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a un contacto NC que conducirá "corriente" (no estará accionado) cuando la entrada 2 del PLC N° 1 se encuentre en estado 0.</li> </ul>  |
|   | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a un contacto auxiliar NA.</li> </ul>   |
|   | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a un contacto auxiliar NC.</li> </ul>   |
|   | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a un contacto NA que estará accionado cuando en la entrada 3 del PLC N° 0, se produzca un flanco ascendente.</li> </ul>   |
|  | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a un contacto NA que estará accionado cuando en la entrada 4 del PLC N° 0, se produzca un flanco descendente.</li> </ul>  |
|  | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a una bobina directa cuyo estado lógico es el mismo que el resultado de las operaciones booleanas que le anteceden en el mismo circuito corresponde a la salida 0 del PLC N° 0.</li> </ul>    |
|  | <ul style="list-style-type: none"> <li>▪ Este símbolo equivale a una bobina inversa cuyo estado lógico es contrario que el resultado de las operaciones booleanas que le anteceden en el mismo circuito y corresponde a la salida 2 del PLC N° 0.</li> </ul> |

### Bobinas Set y Reset

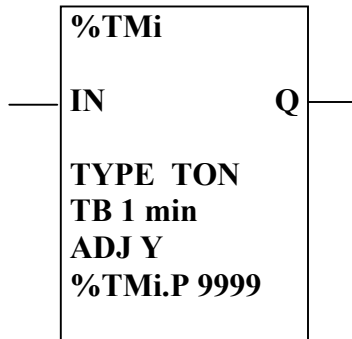


Nota: La instrucción Set tiene prioridad sobre el Reset



## Bloques de función de temporizador % Tmi

### Se proponen 3 tipos de temporizadores:

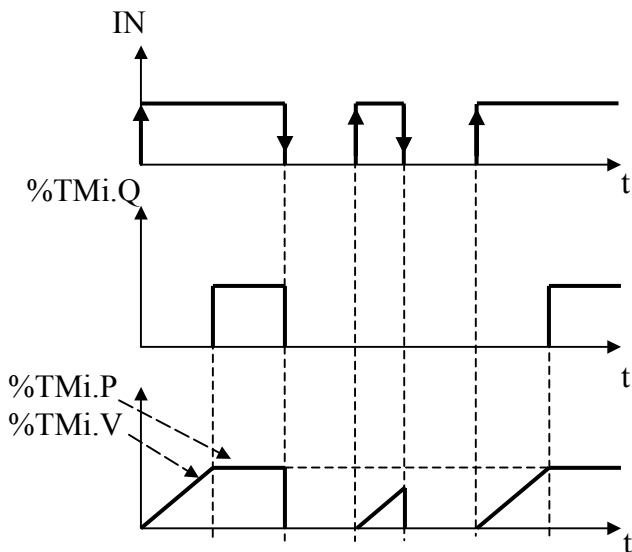


Bloque temporizador

- TON:** Este tipo de temporizador permite gestionar retardos en la conexión. Este retardo es programable.
- TOF:** Este tipo de temporizador permite gestionar retardos en la desconexión. Este retardo es programable.
- TP:** Este tipo de temporizador permite elaborar un impulso de duración precisa. Esta duración es programable.

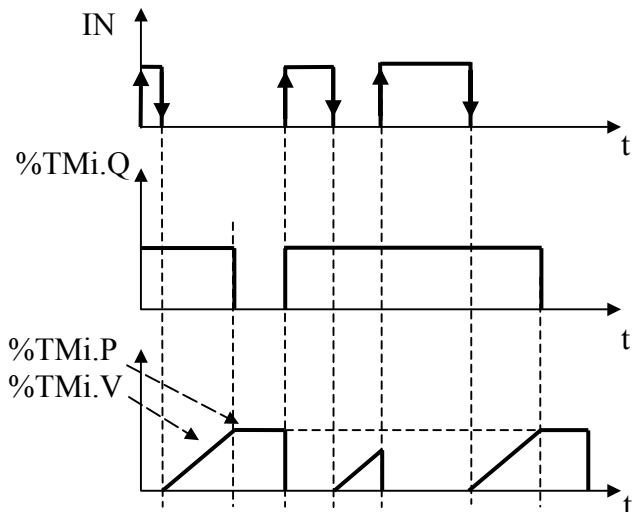
Características del bloque temporizador		
Nro. Temporizador	<b>% Tmi</b>	de 0 a 31
Tipo	<b>TON</b> <b>TOF</b> <b>TP</b>	Retardo en la conexión (por defecto) Retardo en la desconexión Monoestable (pasillo de edificio)
Base de Tiempo:	<b>BT</b>	1 min. (por defecto) 1 s, 100 ms., 10 ms, 1 ms (para TM0 y TM1). Cuanto más corta es la base de tiempo, mayor es la precisión del temporizador.
Valor actual:	<b>%Tmi.V</b>	Palabra que crece de 0 a %Tmi.P en el transcurso del temporizador. El programa puede leer y compararlo pero no escribirlo ( el valor no lo puede modificar el usuario).
Valor de preselección:	<b>%Tmi.P</b>	$0 \leq \%Tmi.P \leq 9999$ . Palabra que el programa puede leer, compararlo y escribir. Por defecto su valor es 9999. La duración o retardo elaborado es igual a % Tmi.P x BT
Entrada:	<b>IN</b>	En flanco ascendente (tipo TON o TP) o flanco descendente (tipo TOF), arranca el temporizador.
Salida Temporizador:	<b>Q</b>	Bit asociado %Tmi.Q, su puesta a 1 depende de la función realizada TON, TOF, TP.

### Utilización como temporizador de retardo en la conexión (tipo TON)



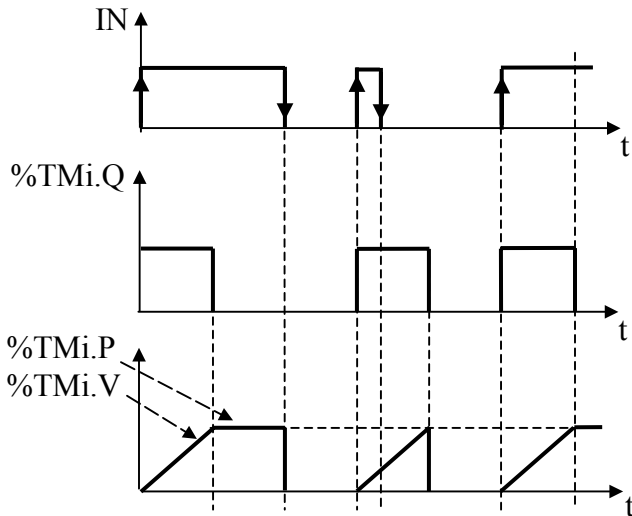
Inicialmente la variable %TMi.P (palabra) tendrá un valor mayor a cero, mientras que el valor de la variable %TMi.V (palabra) será cero. Después de aplicar un flanco ascendente en la entrada IN, se activa el temporizador, el valor de la variable %TMi.V crece en una unidad a cada pulso de la base de tiempo. Solo cuando el valor de %TMi.V iguala al valor de la variable %TMi.P, el valor de %TMi.V se mantendrá constante y el bit de salida %TMi.Q pasa a 1. Después de producirse esa igualdad, el bit %TMi.Q se mantiene en 1 mientras no se aplica un flanco descendente en la entrada IN. Cada vez que se aplica un flanco descendente en la entrada IN, el valor de %TMi.V se vuelve a cero y como es distinto al de %TMi.P, el bit %TMi.Q se pone a 0.

### Utilización como temporizador de retardo en la desconexión: (tipo TOF)



Inicialmente la variable %TMi.P (palabra) tendrá un valor mayor a cero, mientras que el valor de la variable %TMi.V (palabra) será cero. Al aplicar un flanco ascendente en la entrada IN, el bit de salida %TMi.Q pasa a 1. Después de aplicar un flanco descendente de la entrada IN se activa el temporizador, el valor de la variable %TMi.V crece en una unidad a cada pulso de la base de tiempo. Solo cuando el valor de %TMi.V iguala al valor de la variable %TMi.P, el valor de %TMi.V se mantendrá constante y el bit de salida %TMi.Q pasa a 0. Cada vez que se aplique un flanco ascendente en la entrada IN, la variable %TMi.V se pone a 0, mientras que el bit %TMi.Q se pone en 1 volviéndose a repetir la secuencia.

**Utilización como monoestable: tipo TP.**

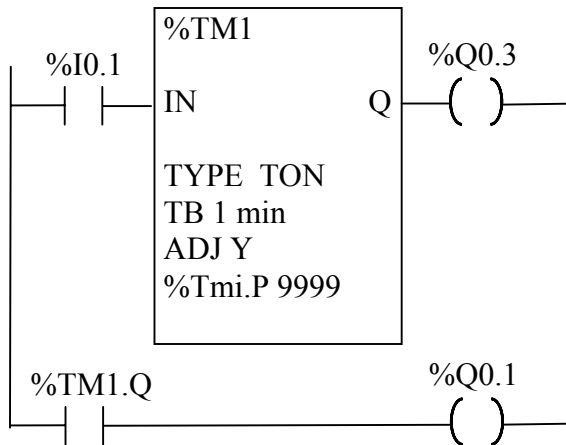


Tras un flanco ascendente de la entrada IN, se activa el temporizador (sí el temporizador **no** se encontraba activado). A partir de la activación, el bit de salida %TMi.Q pasa a 1 y el valor de %TMi.V crece de 0 en una unidad a cada pulso de la base de tiempo BT. Cuando el valor de %TMi.V alcanza al valor de % TMi.P, el bit de salida %TMi.Q pasa a 0.

Si se aplica un flanco descendente en IN el valor de %TMi.V tomará el valor 0 **si** en el momento de aplicar dicho flanco se verifica que: %TMi.V = % TMi.P.

**Programación y configuración**

La programación de los bloques de función del temporizador es idéntica sea cual sea su tipo de utilización. La selección del funcionamiento TON, TOF o TP se efectúa en la configuración.

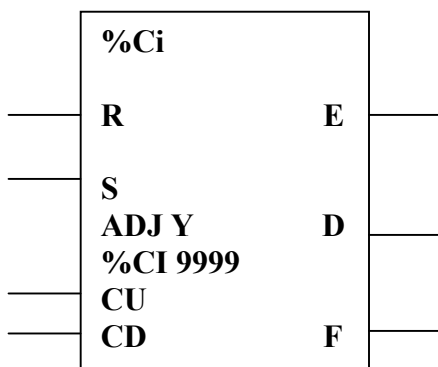


**Configuración**

Los parámetros siguientes deben completarse en la configuración:

- Tipo TON, TOF o TP
- BT 1 min, 1 s, 100 ms, 10 ms o 1 ms
- %TMi.P 0 a 9999
- Ajuste S o N

**Bloques de función de contador/ descontador %Ci**



Bloque contador/descontador

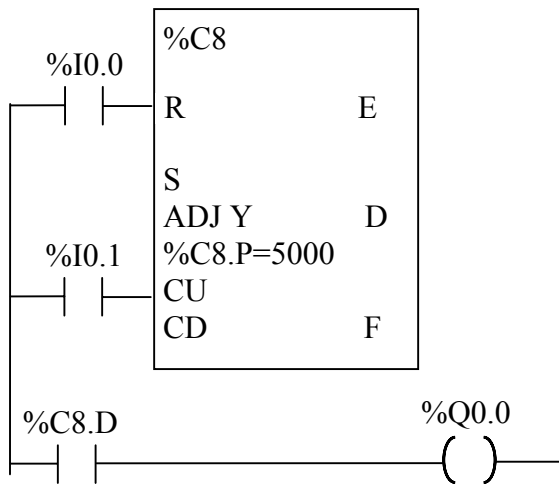
El bloque de función de contador/ descontador realiza el conteo o desconteo de eventos, estas dos operaciones pueden ser simultáneas.

## Funcionamiento.

- **Contaje:** con la aparición de un flanco ascendente en la entrada de contaje CU el valor actual (%Ci.V) aumenta en una unidad. **Solo** cuando el valor de %Ci.V es igual al valor de preselección %Ci.P, el bit de salida %Ci. D pasa al estado 1. El bit de salida %Ci.F (salida por "llenado") pasa al estado 1 cuando %Ci.V pasa de 9999 a 0, y %Ci.F volverá a cero si el contador sigue contando.
- **Descontaje:** con la aparición de un flanco ascendente en la entrada de "descontaje" CD, el valor actual (%Ci.V) disminuye en una unidad. El bit de salida %Ci.E (salida por "vaciado") pasa al estado 1 cuando %Ci.V pasa de 0 a 9999, y %Ci.E volverá a 0 si el contador sigue descontando.
- **Contaje/ descontaje:** para contar y descontar en forma simultánea, es necesario controlar las dos entradas correspondientes a CU y CD. Estas dos entradas se exploran sucesivamente. Si las dos entradas están a 1 simultáneamente, el valor actual no cambia.
- **Puesta a cero:** cuando se pone a 1 la entrada R, el valor actual %Ci.V se fuerza a 0, y las salidas %Ci.E, %Ci.D y %Ci.F tomarán el valor 0. La entrada "puesta a cero" es prioritaria sobre las otras.
- **Puesta a 1:** si la entrada S se pone a 1 y la entrada R "puesta a cero" en 0, el valor actual %Ci.V toma el valor de %Ci.P y la salida %Ci.D tomará el valor 1.

Características del bloque contador		
Nº de contador	%Ci	0 a 15
Valor actual	%Ci.V	Palabra aumentada o disminuida en función de las entradas CU y CD. El programa puede leerla, compararla pero no escribirla (no puedo variar su valor).
Valor de Preselección	%Ci.P	$0 \leq \%Ci.P \leq 9999$ . La palabra puede leerse, compararla y escribirse (valor 9999 por definición)
Entrada de puesta a cero (Reset)	R	En estado 1 %Ci.V=0
Entrada de puesta a 1 (Set)	S	En estado 1 %Ci.V= %Ci.P
Entrada de contaje (Counter Up)	CU	Aumenta %Ci.V en flanco ascendente
Entrada de descontaje (Counter Down)	CD	Dismuye %Ci.V en flanco ascendente
Salida por vaciado (Empty).	E	El bit asociado %Ci.E= 1, cuando el descontaje %Ci.V pasa de 0 a 9999 (puesta a 1 cuando %Ci.V es igual a 9999 y de nuevo a 0 si el contador sigue descontando).
Salida preselección alcanzada (Done)	D	El bit asociado %Ci.D=1 cuando %Ci.V= %Ci.P
Salida por llenado ( Full)	F	El bit asociado % Ci. F = 1 cuando %Ci.V pasa a 9999 a 0 ( puesta a 1 cuando %Ci.V es igual a 0 y de nuevo a 0, si el contador continua contando)

**Configuración y programación.**



Ejemplo:

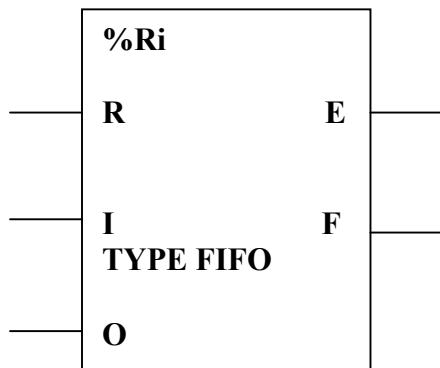
Contaje de un número de piezas = 5.000.

Cada impulso en la entrada %I0.1 provoca el aumento en la palabra %C8.V.

Cuando %C8.V iguala el valor de la palabra %C8.P, el bit %C8.D se pone en 1.

La entrada %I0.0 pone a %C8.V a 0.

**Bloques de función de registro %Ri.**



Bloque de registro

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

- FIFO (primero en entrar, primero en salir).
- LIFO (último en entrar, primero en salir) (Last In, First Out).

**Funcionamiento**

**FIFO (First In First Out).** El primer dato introducido es el primero en salir. Es similar a los viejos monederos usados en los colectivos: la primera moneda que ingresa ( por arriba) es la primera en salir (por debajo).

Cuando se tiene en cuenta una petición de entrada (flanco ascendente en la entrada I) el contenido de la palabra de entrada %Ri.I previamente cargada se almacena en el punto más alto de la pila (figura A).

Cuando la pila está llena (salida F=1), es imposible almacenar.

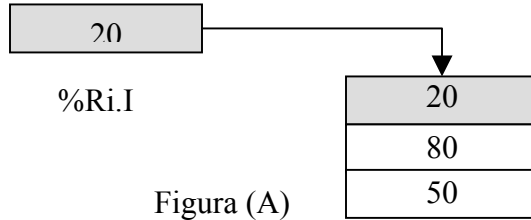
Cuando se tiene en cuenta una petición de salida (flanco ascendente en la entrada O), la palabra información más baja de la pila se coloca en la palabra de salida %Ri.O y el contenido del registro se desplaza un paso hacia abajo (figura B).

Cuando el registro está vacío (salida E=1), es imposible desalmacenar, la palabra de salida %Ri.O ya no cambia y conserva su valor.

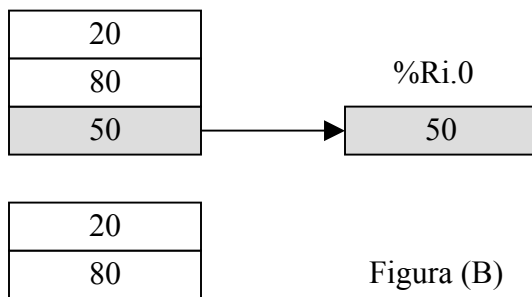
La pila puede reinicializarse en todo momento (estado 1 en la entrada R)

**Ejemplos:**

a) Almacenar el contenido de %Ri.I en el punto mas alto de la pila:



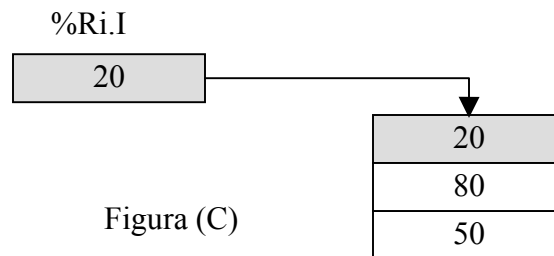
b) Desalmacenar la primera información y ubicarla en %Ri.O



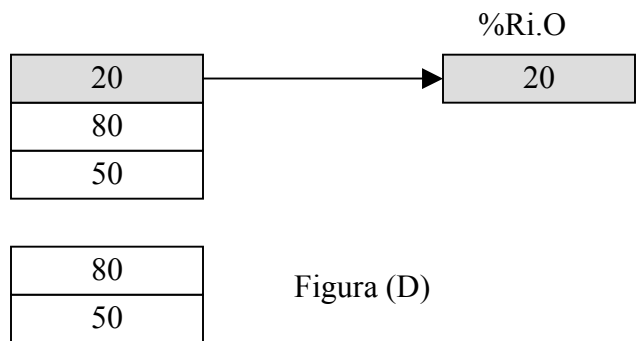
**LIFO (Last In First Out).** El último dato introducido es el primero en salir. Es similar a cuando se estiban cajas en una columna: la última caja en almacenarse (arriba) es la primera en sacarse (de arriba). Cuando se tiene en cuenta una petición de entrada (flanco ascendente en la entrada I), el contenido de la palabra de entrada %Ri.I previamente cargada se almacena en el lugar más alto de la pila (figura C). Cuando la pila está llena (salida F a 1) es imposible almacenar. Cuando se tiene en cuenta una petición de salida (flanco descendente en la entrada O) la palabra de información más alta (última información ingresada) se coloca en la palabra %Ri.O (figura D). Cuando el registro está vacío (salida E=1), es imposible desalmacenar. La palabra de salida %Ri.O ya no cambia y conserva su último valor. La pila puede ser reinicializada en cualquier momento (estado 1 en la entrada R). El elemento indicado es el más alto de la pila.

**Ejemplo:**

d) Almacenar el contenido de %Ri.I en el punto mas alto de la pila



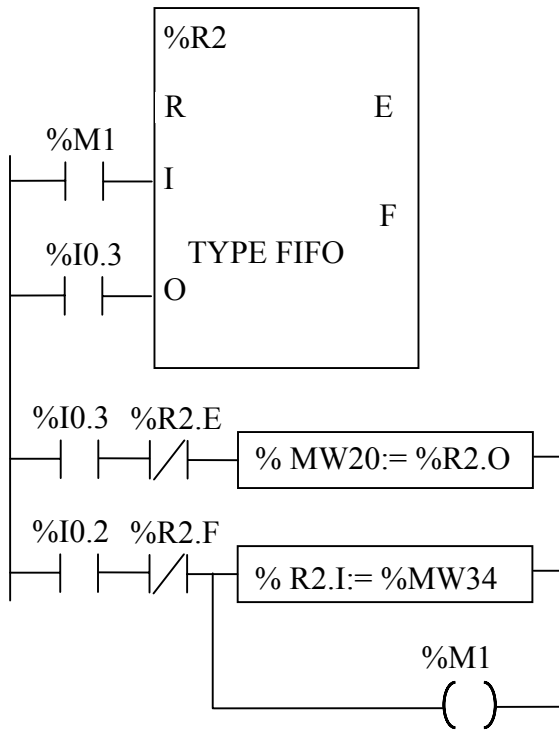
e) Desalmacenar la palabra de información del lugar más alto de la pila.



**Características:**

Nº Registro	% Ri	De 0 a 3
<b>LIFO</b>		
Palabra de entrada	<b>%Ri.I</b>	Palabra de entrada al registro. Puede leerse, compararla y escribirse.
Palabra de salida	<b>%Ri.O</b>	Palabra de salida del registro. Puede leerse, compararla y escribirse
Entrada de "Almacenamiento" (In)	<b>I</b>	En un flanco ascendente almacena el contenido de la palabra %Ri.I en el registro
Entrada de "Desalmacenamiento" (Out)	<b>O</b>	En un flanco ascendente, coloca una palabra de información en la palabra %Ri. 0
Entrada de Puesta a cero (Reset)	<b>R</b>	En el estado 1, inicializa el registro.
Salida Registro "vacío" (Empty)	<b>E</b>	El bit %Ri.E asociado indica que el registro está vacío. Puede comprobarse.
Salida Registro "lleno" (Full)	<b>F</b>	El bit %Ri.F asociado indica que el registro está lleno. Puede comprobarse

**Programación y Configuración**

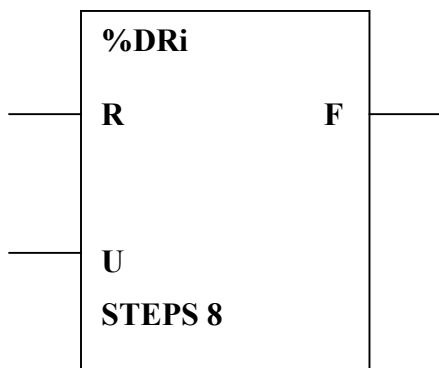


El ejemplo de programa muestra la carga de %R2.1 para la palabra %MW34 en petición de la entrada %I0.2, si el registro R2 no está lleno (%R2.F=0).

%M1 efectúa La petición de entrada en el registro. La petición de salida se realiza por la entrada %I0.3 y la ubicación de %R2.0 en %MW20 se efectúa si el registro no está vacío (%R2.E=0).

**Configuración** El único parámetro que se debe introducir en la configuración es el tipo de registro FIFO (por defecto) o LIFO.

**Bloques de función de programación cíclico %Dri**



Bloque del programador cíclico

Con un principio de funcionamiento similar al programador de levas, el programador cíclico cambia de paso en función de eventos exteriores.

A cada paso, el punto alto de una leva da una orden ejecutada por el automatismo.

En el caso de un programador cíclico, estos puntos altos se simbolizan por un estado 1 al nivel de cada paso y son asignados a bits de salida %Qi.j o a bits internos %Mi. llamados bits de orden.

**Características**

Número	<b>%DRi</b>	0 a 3
Número de Paso en curso	<b>%DRi.S</b>	$0 \leq \%DRi.S \leq 7$ . Palabra que puede leerse y compararla. Sólo puede escribirse en el programa a partir de un valor decimal inmediato
Número de pasos configurados		1 a 8 (por defecto)
Entrada regreso al paso 0 (RESET)	<b>R</b>	En el estado 1, inicializa el programador al paso 0
Entrada "avance" (UP)	<b>U</b>	En un flanco ascendente, avanza en un paso el programador y actualiza los bits de orden
Salida (FULL)	<b>F</b>	Indica que el último paso definido está en curso. El bit %DRi.F asociado puede ser comprobado (%DRi.F=1 si %DRi.S= número de pasos configurados - 1).
Bits de orden		Salidas o bits internos asociados al paso (16 bits de orden).

**Funcionamiento**

El programador cíclico se compone de:

- Una matriz de datos constantes (levas) organizada en 8 pasos (de 0 a 7) y de 16 informaciones binarias (estados de paso) ordenadas en columnas y referenciadas de 0 a F.
- Una lista de bits de orden (1 por columna) que corresponden a salidas %Q0.i o %Q1.i o bits internos %Mi. En el paso en curso, los bits de orden toman los estados binarios definidos para este paso.

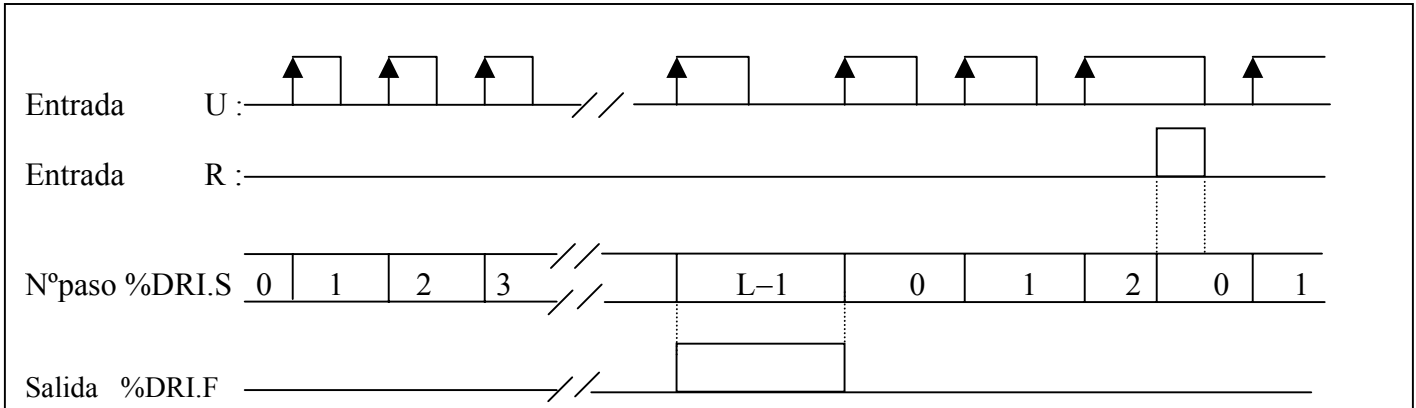
La tabla siguiente resume las características principales del programador cíclico.

Bits de orden →	0	1	2	13	14	15
	↓	↓	↓	↓	↓	↓
	%Q0.1	%Q0.3	%Q1.5	%Q0.6	%M1	%M2
Paso 0	0	0	1	1	1	0
Paso 1	1	0	1	1	0	0
Paso 6	0	1	1	0	1	0
Paso 7	1	1	1	1	0	0

En el ejemplo anterior, como está en curso el paso 5, los bits de orden %Q0.1, %Q0.3 y %Q1.5 se ponen a 1; mientras que los bits de orden %Q0.6, %M1 y %M2 se ponen en 0.

El número del paso en curso aumenta en cada flanco ascendente de la entrada U. Este número puede ser modificado por el programa.

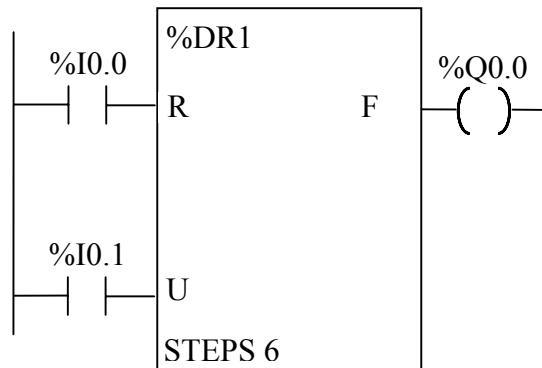
**Diagrama de funcionamiento**



(L = Número de pasos configurados)

**Programación y configuración:**

En este ejemplo, las 5 primeras salidas %Q0.0 a %Q0.4 se activan una tras otra cada vez que la entrada %I0.1 se pone a 1. La entrada I0.0 reinicializa las salidas a 0.



**Configuración:**

La información siguiente se define en la configuración:

- 1) Número de pasos: 6 (es decir que irá del paso N° 0 al paso N°5).
- 2) Asignación de los bits de orden:
  - Paso N° 1: %Q0.0 ; Paso N° 2: %Q0.1 ; Paso N°3: %Q0.2
  - Paso N° 4 : %Q0.3 ; Paso N°5: %Q0.4
- 3) Los estados de las salidas (bits de orden) para cada paso del programador.

Salida (Q)	0	1	2	3	4
	↓	↓	↓	↓	↓
Paso 0:	0	0	0	0	0
Paso 1:	1	0	0	0	0
Paso 2:	0	1	0	0	0
Paso 3:	0	0	1	0	0
Paso 4:	0	0	0	1	0
Paso 5:	0	0	0	0	1

## Punto de ajuste analógico

El PLC Nano dispone en la parte delantera de un potenciómetro de ajuste analógico (de color gris).

### Programación

Los valores numéricos de 0 a 255 que corresponden a los valores analógicos proporcionados por dichos potenciómetros, están disponibles en las palabras sistema %SW112 para el potenciómetro de color gris.

Esta palabra se puede utilizar para los siguientes tipos de ajustes:

- Ajuste de temporizadores.
- Ajuste de contadores.
- Ajuste de la frecuencia del generador de pulsos.
- Etc.

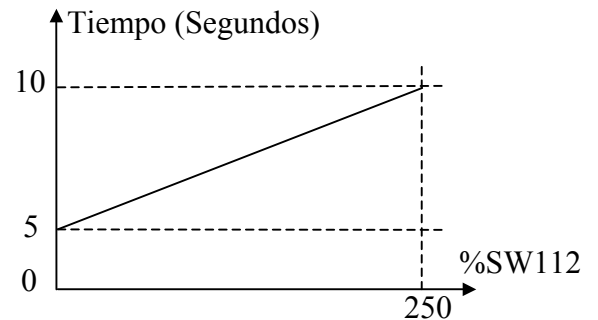
### Ejemplo

Ajuste de la duración de una temporización de 5 a 10 seg con el potenciómetro gris.

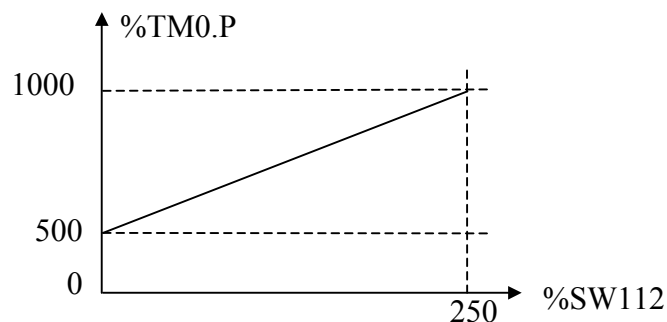
Para este ajuste, se utiliza casi toda (255) la extensión de ajuste del potenciómetro de 0 a 250. La relación entre el tiempo de temporización y la palabra %SW112 es lineal.

En la configuración, se seleccionan los parámetros siguientes para el bloque del temporizador %TM0:

- Tipo de temporizador: On- Delay (**TON**).
- Temporizador número: 0 (**%TM0**)
- Base de tiempo BT: 0,01 segundos.



Adoptando una relación lineal entre %TM0.P y %SW 112, se obtiene el siguiente gráfico:



La ecuación de %TM0.P en función de %SW112 será:

$\%TM0.P = 500 + m * \%SW112$  (1); siendo m la pendiente de la recta y cuyo valor es:

$$m = \frac{1000 - 500}{250 - 0} = 2 \quad (2)$$

Reemplazando (2) en (1), se obtiene:

$$\%TM0.P = 500 + 2 * \%SW112 \quad (3)$$

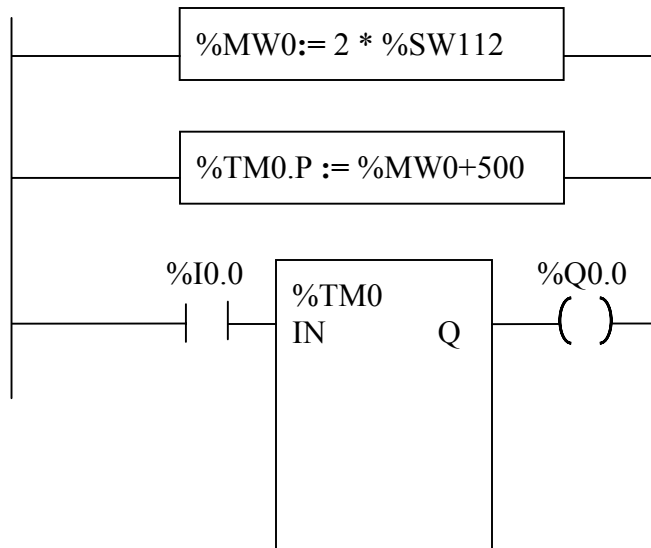
Si quisiéramos introducir esta operación, el programa del PLC no lo permitirá ( se sugiere al alumno intentarlo). Para solucionar este problema, se recurre a una variable intermedia ( del tipo palabra), del siguiente modo:

$$\%MW0 = 2 * \%SW112 \quad (4)$$

Reemplazando (4) en (3), se obtiene:

$$\%TM0.P = 500 + MW0 \quad (5)$$

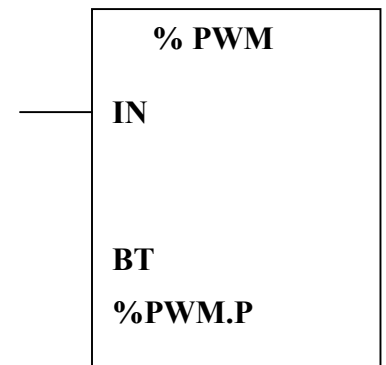
La (4) y la (5) se “escribe “ en el PLC con el bloque **Funcionar**, quedándonos:



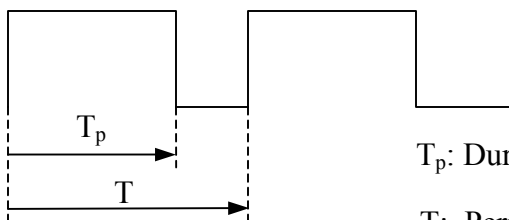
### Salida de modulación de amplitud %PWM

El bloque de función %PWM permite generar en la salida %Q0.0 una señal rectangular, cuya duración en estado alto puede cambiarse (relación cíclica) por programa.

Además esta función permite controlar un módulo de salida analógica conectado a la salida %Q0.0.



Bloque % PWM



$T_p$ : Duración en estado alto (ajustable)

T: Periodo fijo (ajustable)

## Funcionamiento

El período fijo (T) esta dado por :  $T = \%PWM.P \times BT$ ; y se puede fijar en la etapa de configuración mediante la selección de la base de tiempo (BT) y de la preselección del periodo (%PWM.P).

La duración del estado alto (Tp) está dada por:  $Tp = T \times (\%PWM.R / 100)$ .

Puedo variar Tp modificando %PWM.R por el programa

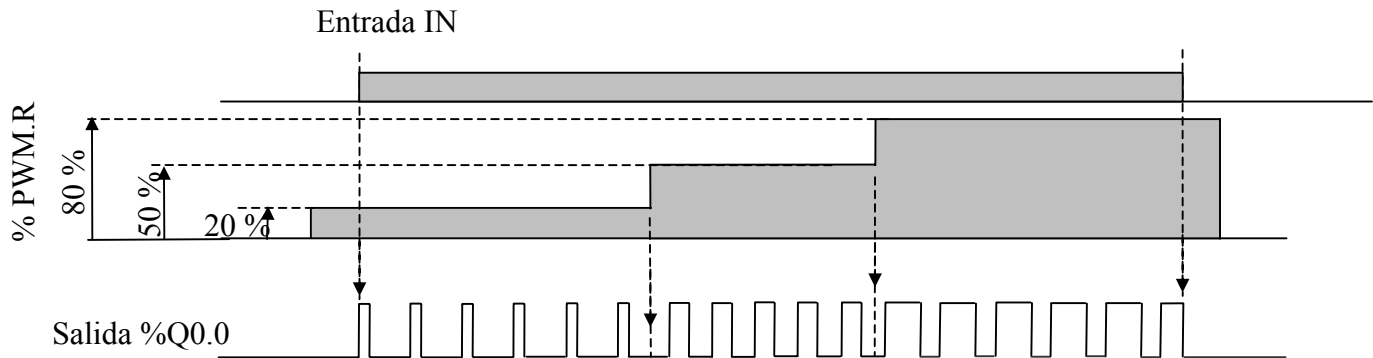


Figura A

## Características

Base de tiempo	<b>BT</b>	0,1 ms <sup>(1)</sup> , 10 ms, 1 s (valor por defecto)
Preselección del período	<b>%PWM.P</b>	0 < %PMW.P < 32767 si la base de tiempo es de 0,01 s ó 1s. 0 < %PMW.P < 255 si base de tiempo es de 0,1 milisegundos. (0= función inactiva). En configuración, se accede para fijar el valor de: $T = \%PWM.P \times BT$ .
Intervalo del periodo	<b>%PWM.R</b>	0 ≤ %PMW.R ≤ 100 <sup>(2)</sup> esta palabra da el porcentaje de la señal en el estado 1 sobre el período (0= valor por defecto). El valor de Tp está dado por: $Tp = T \times \%PWM.R / 100$ . La palabra %PWM.R se escribe mediante el programa de usuario, esta palabra es la que permite variar Tp.
Entrada generación de pulsos	<b>IN</b>	En estado 1, genera la señal modulada en amplitud en la salida %Q0.0 En estado 0, pone la salida % Q0.0 a 0.

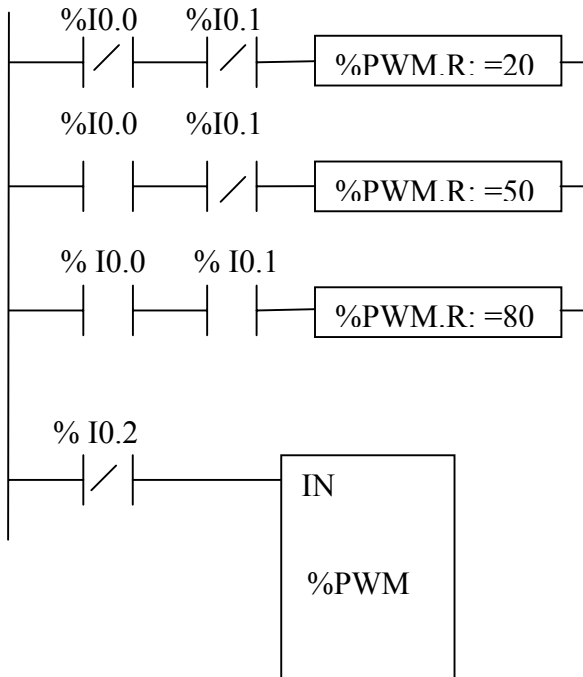
(1) Se desaconseja utilizar esta base de tiempo para los TSX Nano con salidas relés.

(2) Los valores superiores a 100 se considerará como iguales a 100.

**Programación y Configuración**

Queremos implementar con el PLC un ladder que cumpla con lo indicado en la Figura A.

En esa figura, la duración en estado alto de la señal se modifica por programa en función del estado de las entradas %I0.0 y %I0.1 del PLC



El período de la señal se fija en 500 milisegundos en configuración

Si %I0.0 y %I0.1 están a 0, el intervalo %PWM.R se fija en 20%. La duración de la señal en el estado 1 es entonces del 20% x 500 ms= 100 ms.

Si %I0.0 está en 1 e %I0.1 están a 0, el intervalo %PWM.R se fija en 50% (duración 250 ms).

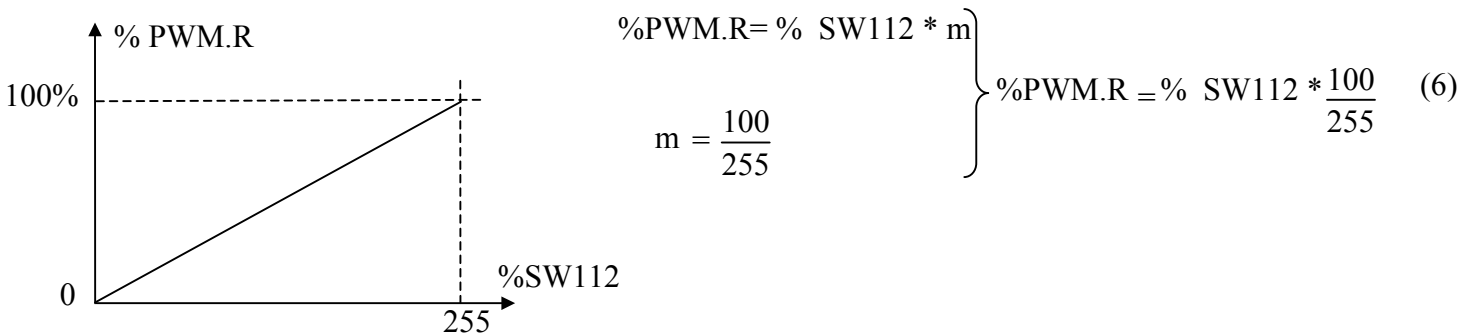
Si %I0.0 e %I0.1 están es estado 1, el intervalo %PWM.R se fija en 80% (duración 400 ms).

Otro ejemplo:

Queremos variar Tp mediante el potenciómetro gris que comanda la palabra %SW112.

Como  $T_p = T \cdot (\% \text{ PWM.R} / 100)$ , variando % PWM.R puedo variar Tp.

Para ello, se adopta una relación lineal entre % PWM.R y %SW112

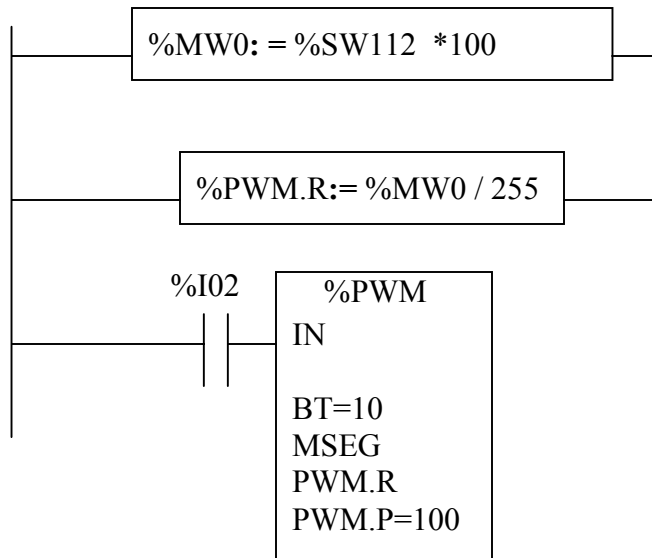


La (6) no la podemos implementar con el PLC pues al hacer el cociente (100 / 255), como el resultado es 0,3921.. el PLC al suprimir la parte decimal, el cociente anterior valdrá 0 produciendo un error. Para evitar esto se hace lo siguiente:

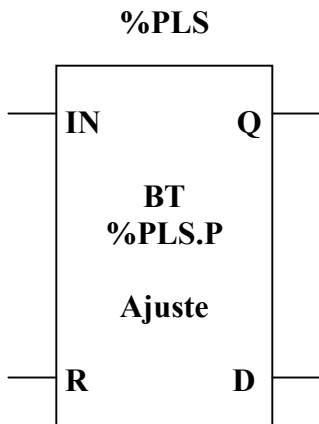
Llamando a  $\% MW0 = \% SW112 \cdot 100$ , reemplazando en (6) obtenemos:

$$\%PWM.R = \frac{\% MW0}{255}$$

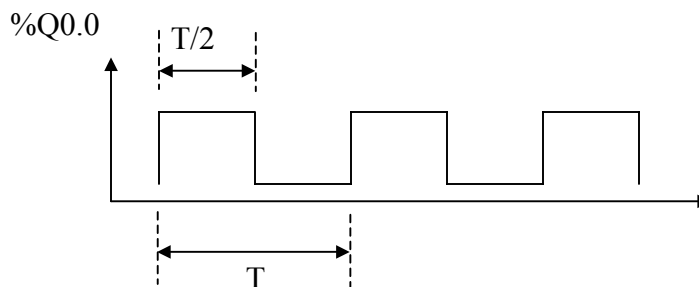
El ladder resultante, será:



### Salida del generador de pulsos %PLS



**Bloque %PLS**



El bloque de función %PLS permite generar una señal cuadrada (relación cíclica de 50% garantizada si %PLS tiene un valor par) en la salida autómeta %Q0.0

Esta señal puede ser:

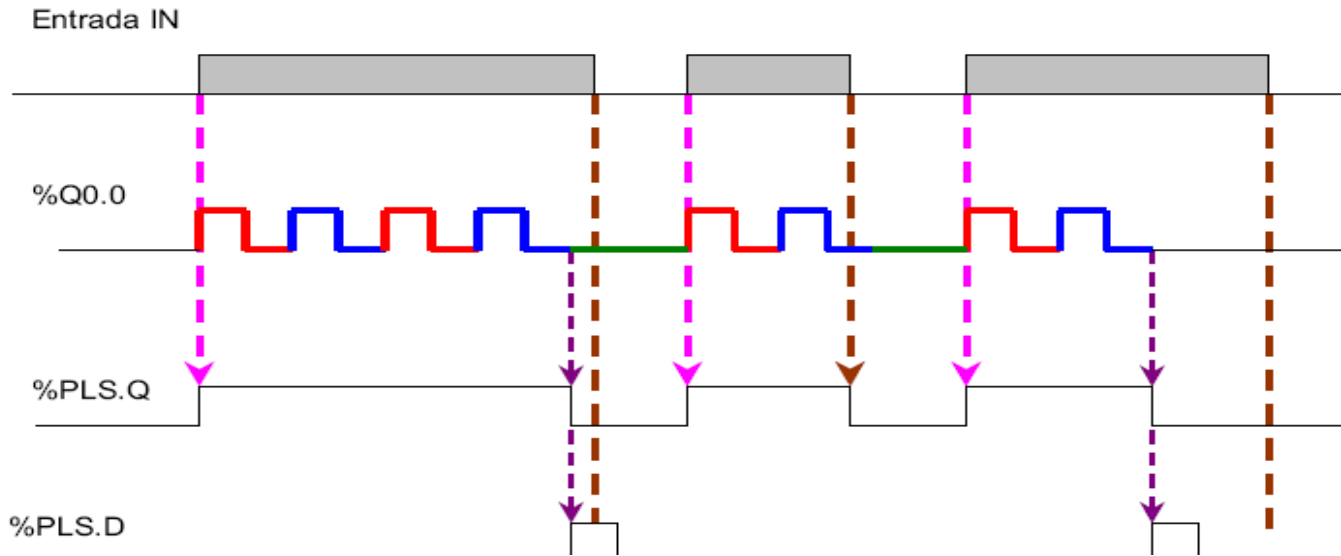
- de duración limitada: el número de pulsos y el período se escriben por el programa (o en configuración).
- de duración ilimitada: el período se escribe por el programa o en la configuración.

### Características

Base de tiempo	<b>BT</b>	0.1 ms <sup>(1)</sup> ; 10 ms ; 1 s (valor por defecto).
Preselección (T)	<b>%PLS.P</b>	0 < %PLS.P < 32767 si BT=10 ms o 1 s. 0 < %PLS.P < 255 si BT=0.1 ms <sup>(1)</sup> . La preselección permite modular el período de la señal $T = \%PLS.P * BT$ Nota: % PLS.P debe ser un número par.
Número de pulsos	<b>%PLS.N</b>	0 ≤ %PLS.N ≤ 32767, esta palabra da el número de pulsos del tren de pulsos a generar. 0 = señal cuadrada de duración ilimitada (por defecto). El programa verifica y escribe %PLS.N
Ajuste por Terminal	<b>O/S</b>	O: Posibilidad de modificar el valor de preselección %PLS.P en modo de ajuste. N: Ningún acceso en modo de ajuste.
Entrada generación de pulsos	<b>IN</b>	En estado 1, genera la señal sobre la salida %Q0.0. En estado 0 pone la salida %Q0.0 a 0
Entrada de reinicialización (reset)	<b>R</b>	En estado 1, pone a 0 el número de pulsos de las salidas %PLS.Q y % PLS.D
Salida generación de pulsos en curso	<b>%PLS.Q</b>	En estado 1, se está generando la señal en las salida %Q0.0
Salida generación de pulsos terminada	<b>%PLS.D</b>	En estado 1, se termina la generación de la señal en la salida %Q0.0.

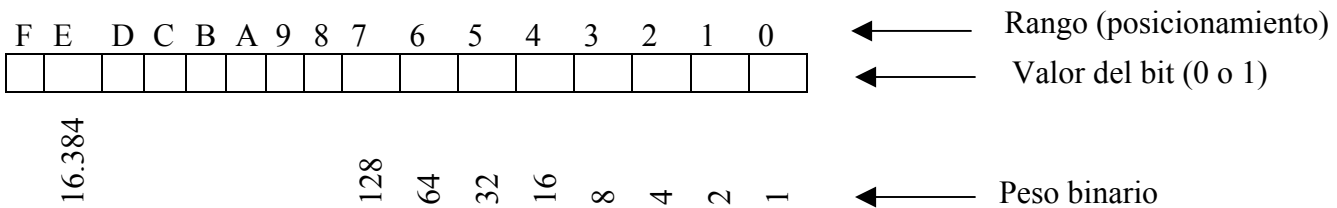
(1) Se aconseja usar esta base de tiempo para PLC con salida para relé

## Funcionamiento



## Tratamiento de palabras

Hasta ahora hemos estado trabajando con valores binarios (0 y 1). Si se desea trabajar con valores analógicos, para eso tendré que engañar al PLC. Para ello el número analógico lo voy a presentar mediante una PALABRA, la cual a través de un peso binario, representará al número analógico en cuestión. La longitud de la palabra en el nano es de 16 bits.



El bits de rango F se atribuye por convención al signo del valor almacenado en la palabra.

F=0, El contenido de la palabra tiene valor positivo (Máximo valor positivo = 32767).

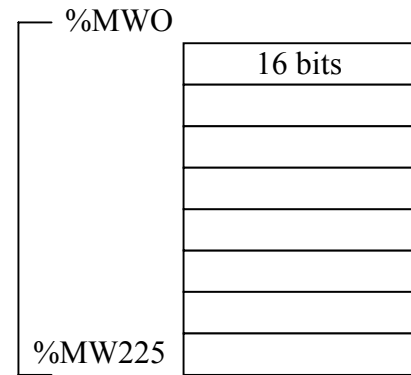
F=1, El contenido de la palabra tiene valor negativo (Mínimo valor negativo = -32768) ( los valores negativos se expresan en lógica complemento a 2).

## Tipos de palabras

- Palabras internas
- Palabras constantes.
- Palabras de intercambio entre autómatas.
- Palabras sistemas.
- Palabras bloque de función.
- Objetos de bits extraídos de palabras.

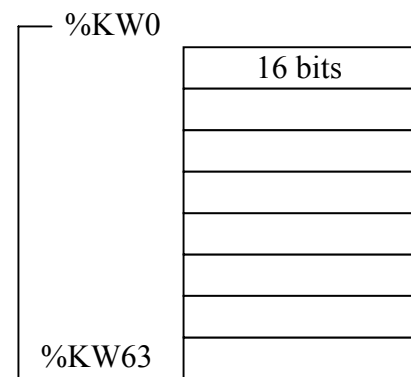
**a) Palabras Internas**

Las palabras internas están destinadas a almacenar valores. Se accede a las palabras % MW0 a %MW255, directamente desde el programa (en lectura/ escritura). Se utilizan como palabras de trabajo.



**b) Palabras constantes**

Las palabras constantes memorizan valores constantes o mensajes alfanuméricos. Estas palabras se almacenan en la memoria de programa. Se accede a las palabras constantes %KW0 a %KW63 directamente desde el programa (sólo en lectura).



**c) Palabras de intercambio de entradas / salidas**

Las palabras de intercambio %IW/QW están asociadas a los autómatas conectados al cable de extensión. Permiten los intercambios entre los autómatas.

**d) Palabras de Sistema**

Estas palabras de 16 bits aseguran varias funciones: dan acceso a informaciones que provienen directamente del autómata mediante la lectura de las palabras %SWi (Ej: valores de los puntos de ajuste analógico) y permiten actuar sobre la aplicación (Ej: ajuste del reloj calendario).

**e) Palabras bloque de función:**

Ejemplo: % Tmi.P; %Ci.P

**f) Objetos de bits extraídos de palabras**

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

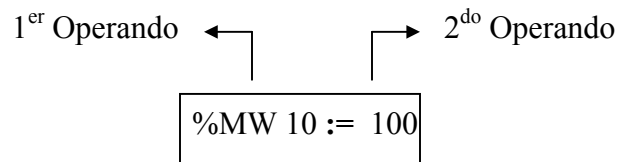
Sintaxis: % Objeto Palabra : Xk con k= 0 a 15 rango del bit del objeto palabra.

Ejemplo: %MW5: X6, esto significa que se trata del bit de rango 6 de la palabra interna %MW5.

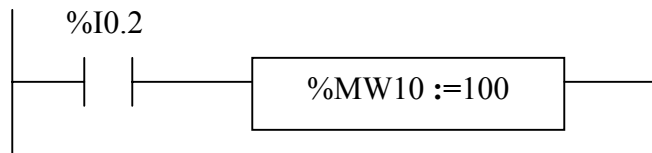
## UTILIZACIÓN DE LAS PALABRAS

¿Qué cosas puedo hacer con las palabras?.

### a) Instrucciones de Transferencia o asignación (Bloque Funcionar)



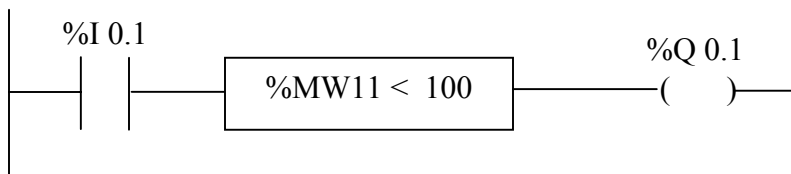
Con esto guardo el N° 100 en la palabra interna N° 11, ( la primera palabra es la 0) es decir que el 2<sup>do</sup> operando lo guardo en el 1<sup>er</sup> operando. En el diagrama ladder esto se visualizaría del siguiente modo:



Si se acciona la entrada `%I0.2`, el valor 100 se le asigna a la palabra `%MW10`.

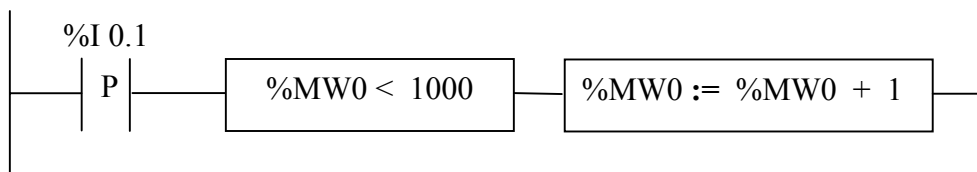
### b) Instrucciones de comparación

Comparo el valor analógico que está alojado en una palabra interna con otro número analógico. En el diagrama ladder esto se visualizaría del siguiente modo:

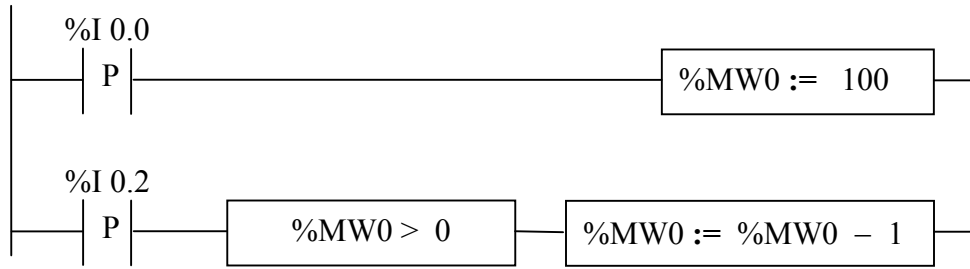


Si la entrada `%I0.1` está accionada y si el valor analógico que está almacenado en la palabra interna `%MW11` es menor que 100, entonces está activa la salida `%Q0.1`.

### Otro ejemplo: Contador ascendente de 0 a 1000



**Otro ejemplo : Contador descendente hasta cero.**



**Otro ejemplo: Arranque Secuencial de Motores**

%I0.1: Pulsador de Marcha.

%I0.2: Pulsador de Parada.

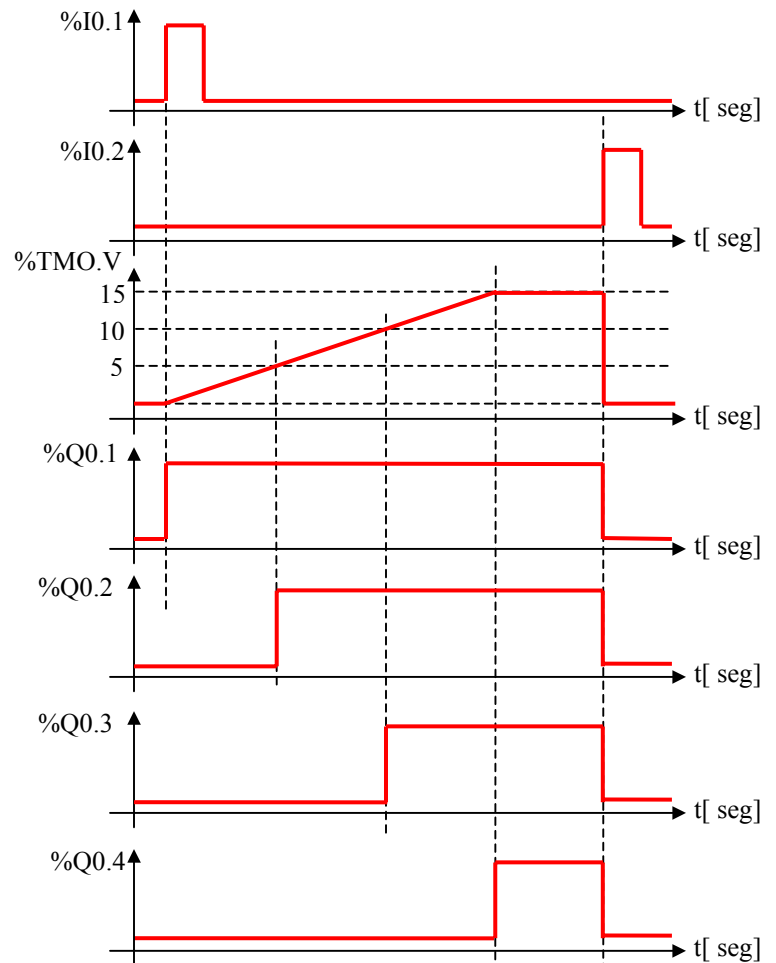
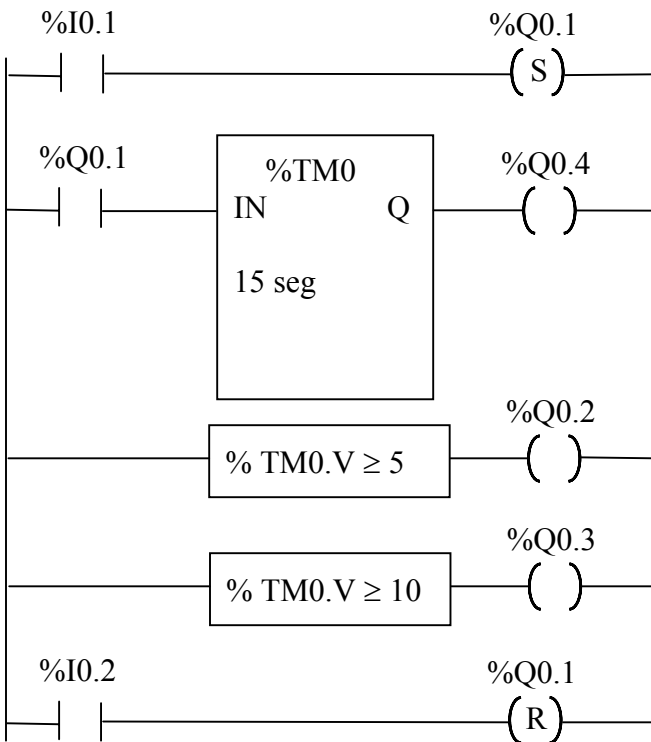
%Q0.1: Motor 1, arranca al accionar el pulsador de marcha.

%Q0.2: Motor 2, arranca 5 segundos después del motor 1.

%Q0.3: Motor 3, arranca 10 segundos después del motor 1.

%Q0.4: Motor 4, arranca 15 segundos después del motor 1.

Al apretar el pulsador de parada se detienen simultáneamente los cuatro motores.

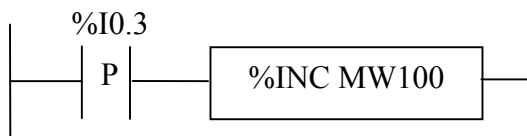


### **Otras Instrucciones de Comparación son:**

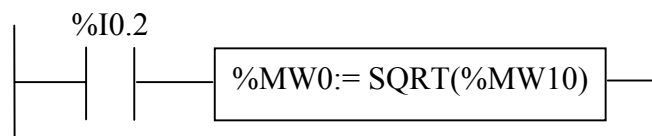
Op1 > Op2	Prueba si el operando 1 es superior al operando 2.
Op1 ≥ Op2	Prueba si el operando 1 es superior o igual al operando 2
Op1 < Op2	Prueba si el operando 1 es menor al operando 2
Op1 ≤ Op2	Prueba si el operando 1 es menor o igual al operando 2
Op1 = Op2	Prueba si el operando 1 es igual al operando 2
Op1 <> Op2	Prueba si el operando 1 es diferente del operando 2

### **c) Instrucciones aritméticas**

Op1 + Op2	Suma de dos operandos
Op1 - Op2	Resta de dos operandos
Op1 * Op2	Producto de dos operandos
Op1 / Op2	División de dos operandos
Op1 REM Op2	Resto de dos operandos
SQRT Op2	Raíz Cuadrada de un operando
INC Op2	Incremento de un operando
DEC Op2	Decremento de un operando



INC: Incremento de un operando



SQRT: Raíz Cuadrada de un operando

### **d) Instrucciones lógicas (and, or, not, or excluida)**

### **e) Otras Instrucciones**