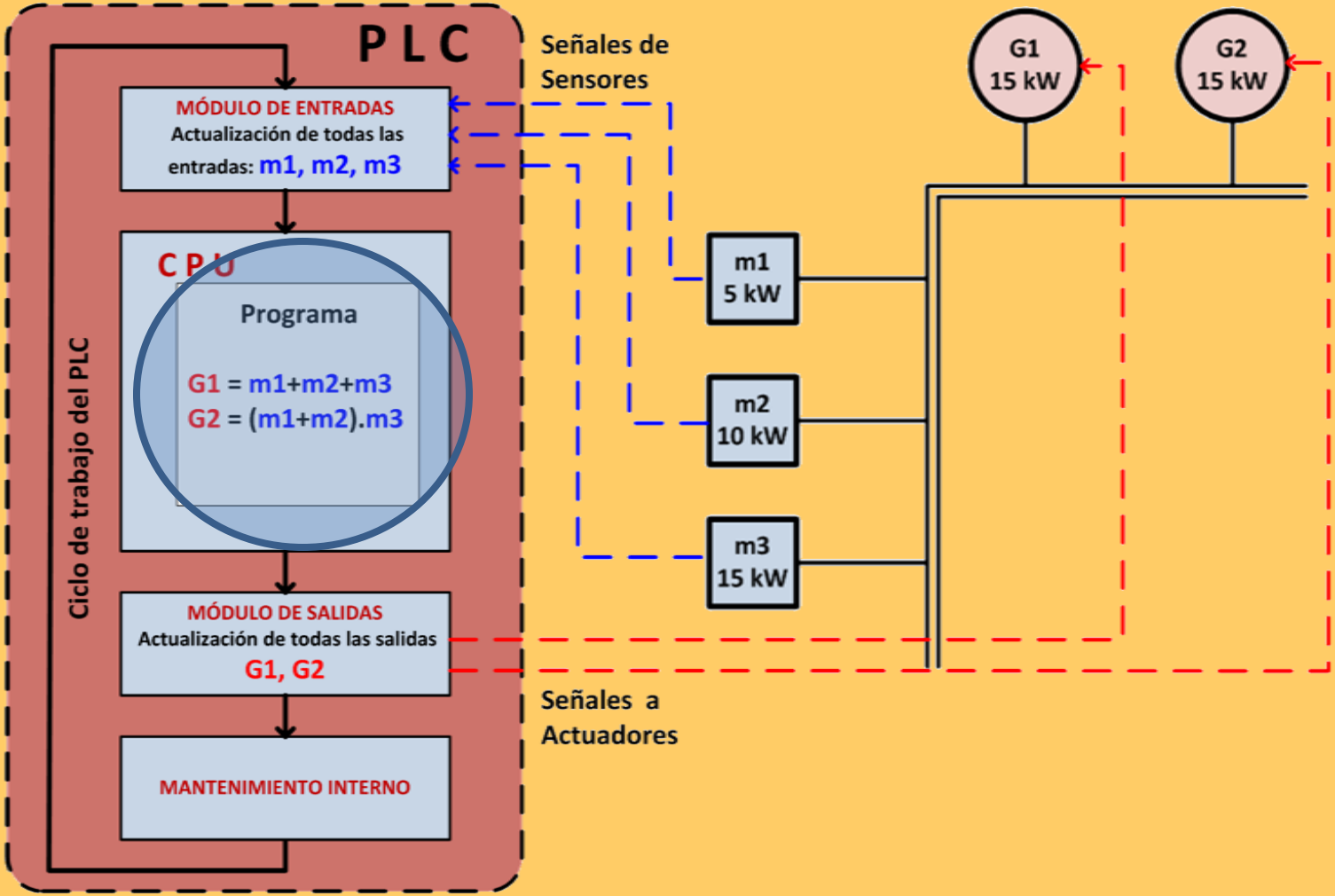
PROGRAMACIÓN DE PLC

CÓMO FUNCIONA UN PLC

Control Combinacional – Programación del PLC



PROGRAMACIÓN DE PLC

Procedimiento para programar y cargar:

- **Determinar los requisitos del sistema al cual se aplica el PLC.**
- **Identificar los dispositivos de E/S y asociarlos a las direcciones físicas mediante una tabla de asignación.**
- **Preparar tablas que indiquen: bits de trabajo, temporizadores, contadores.**
- **Generar el programa en el lenguaje seleccionado.**
- **Transferir el programa a la CPU. Si se realiza mediante consola habrá que traducir el programa a mnemónico.**
- **Verificar, vía simulación, el correcto funcionamiento del programa.**
- **Memorizar el programa definitivo.**

LENGUAJES DE PROGRAMACIÓN

Los lenguajes para la programación de los PLCs han sido considerados en el estándar IEC 61131-3.

Se definieron dos lenguajes literales:

- Lista de instrucciones (IL) – lenguaje de tipo ensamblador con uso de acumuladores.
- Texto Estructurado (ST) – un lenguaje de alto nivel similar a C y, sobre todo a Pascal.

Se establecieron tres lenguajes gráficos:

- Diagrama de Bloques de Funciones (FBD) - Basado en esquemas de compuertas lógicas
- Diagramas de Tipo Escalera (LD) o lenguaje de contactos
- Diagrama de Funciones Secuenciales (SFC) o GRAFCET. Especificado para sistemas secuenciales.

LISTA DE INSTRUCCIONES (IL)

Lenguaje de texto (mnemónico), similar al assembler.

Cada línea de programa contiene una sola instrucción y su ejecución es secuencial.

Todos los operadores trabajan con un registro especial, denominado acumulador (LD, ST).

Es conveniente para los programas pequeños. Pero cualquier programa en otro lenguaje puede traducirse a IL.

Ideal para programar con dispositivo manual.

Para el sistema

G1 = m1+m2+m3

G2 = (m1+m2).m3

```
// Caso de Estudio
LD %I.1   // Inicio
OR  %I.2
OR %I.3
ST  %Q.1
LD %I.1
OR %I.2
AND %I.3
ST %Q.2
END
```

TEXTO ESTRUCTURADO (ST)

Lenguaje de alto nivel estructurado en bloques. Sintaxis similar al Pascal. Posibilidad de utilizar expresiones complejas e instrucciones anidadas.

```
// Caso de Estudio
Q 4.0 := I 0.0 AND I 1.1 OR NOT I 0.1
IF Q 4.0 == 1 THEN GOTO M001
ELSE Q 1.0 = NOT Q 4.0;
END_IF;
M001 MW 2= 1+MW 2;
```

Soporte para:

- Lazos (Repeat-Until, While-Do)
- Ejecución Condicional (If-Then-Else, Case)
- Funciones (SQRT(), SIN(), etc.)

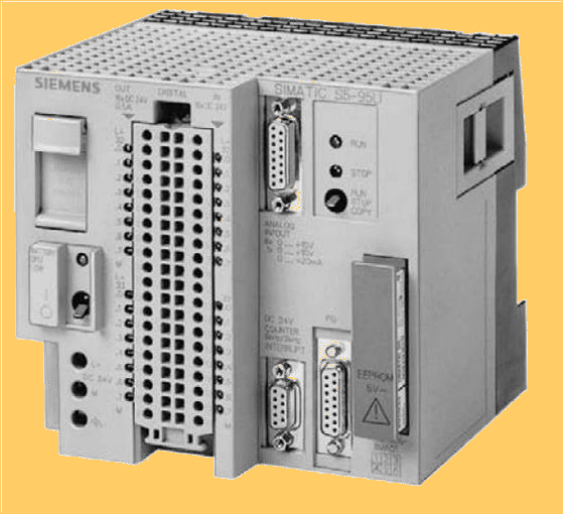


DIAGRAMA DE BLOQUES DE FUNCIONES (FBD)

Lenguaje gráfico. Proviene del campo del procesamiento de señales. Permite elementos de programa que se unen en forma análoga a compuertas lógicas en un circuito electrónico.

Cada función lógica tiene asociado un bloque funcional que realiza la operación que corresponde con grafos estándar. Para el mismo ejemplo:

G1 = m1+m2+m3
G2 = (m1+m2).m3

Se ejecuta de arriba hacia abajo

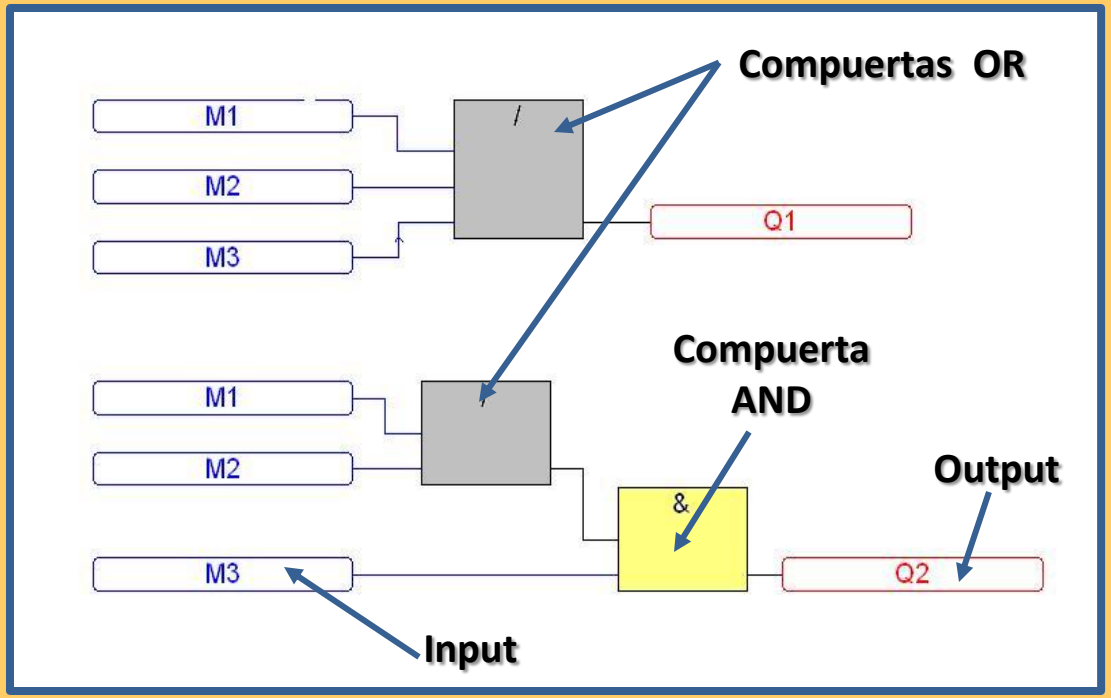
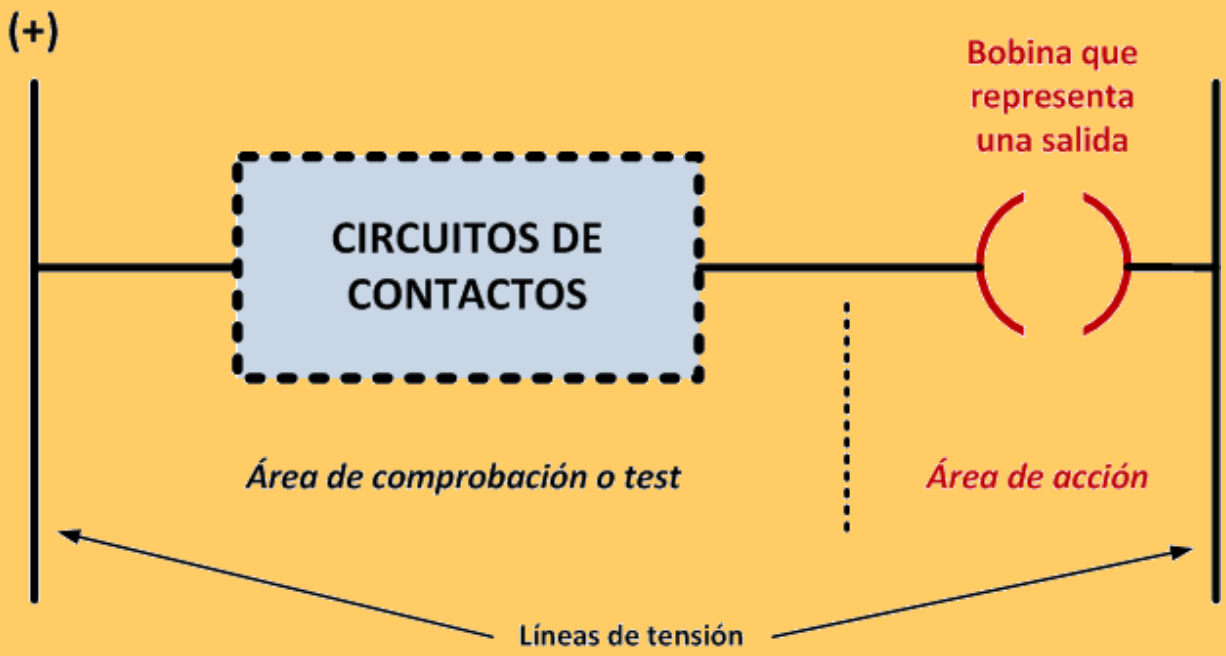


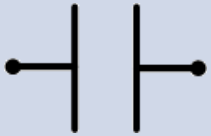
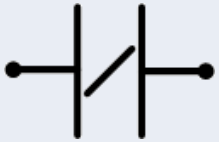
DIAGRAMA DE TIPO ESCALERA O DE CONTACTOS (LD)

Un circuito de contactos se compone de una serie de instrucciones gráficas específicas, relacionadas entre sí mediante conexiones horizontales y verticales que conducen a una o varias salidas y/o acciones, situadas entre las dos barras verticales que representan la diferencia de potencial. Se siguen las reglas del álgebra de contactos.

Las funciones lógicas se representan mediante un circuito de contactos conectado en serie con la variable de salida. El cierre de dicho circuito de contactos activa la variable de salida. La línea vertical de la izquierda representa el terminal de alimentación, mientras que la línea vertical de la derecha representa el terminal de masa.



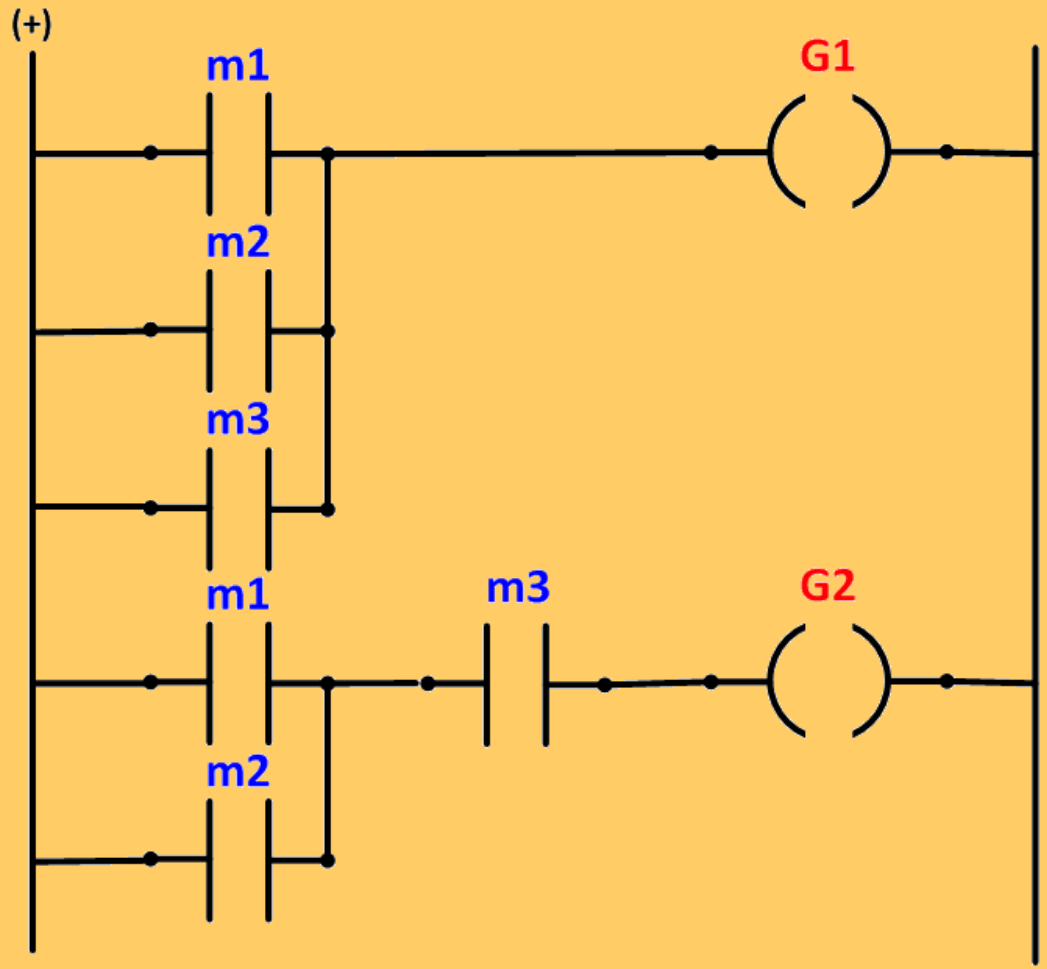
LENGUAJE DE CONTACTOS (LD)

ELEMENTO	SÍMBOLO	ACCIÓN
Contacto normalmente abierto		Conduce cuando la variable asociada está en 1 (activada)
Contacto normalmente cerrado		Conduce cuando la variable asociada está en 0 (inactiva)
Bobina Directa		La variable asociada toma el valor del resultado de la zona de test
Bobina inversa (negada)		La variable asociada toma el valor inverso del resultado de la zona de test
Bobina de Set		La variable asociada se pone en 1 cuando el resultado de la zona de test es 1 y se mantiene activa aunque el circuito de contactos se abra. Pasará a 0 por acción de una bobina Reset.
Bobina de Reset		La variable asociada se pone en 0 cuando el resultado de la zona de test es 1 y permanece así aunque el circuito se abra. Pasará a 1 por acción de una bobina Set.

LENGUAJE DE CONTACTOS (LD)

La conexión de contactos en serie equivale a la función de operación lógica **AND** y la conexión de contactos en paralelo equivale a la función de operación lógica **OR**.

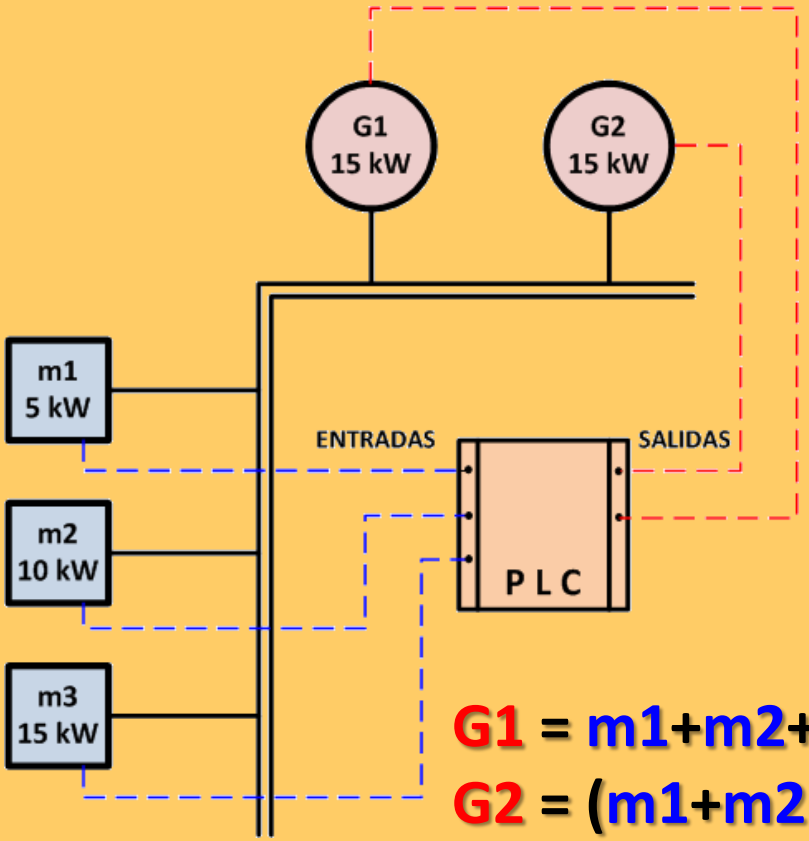
Para que se produzca la activación de la variable de salida (que corresponde a la bobina de un relé) es preciso que el circuito de contactos se cierre al menos a través de uno de los caminos alternativos que conducen a la variable de salida.



El **orden de ejecución** es de **arriba a bajo** y de **izquierda a la derecha**, primero los contactos y luego las bobinas, de manera que al llegar a éstas ya se conoce el valor de los contactos y se activan si corresponde.

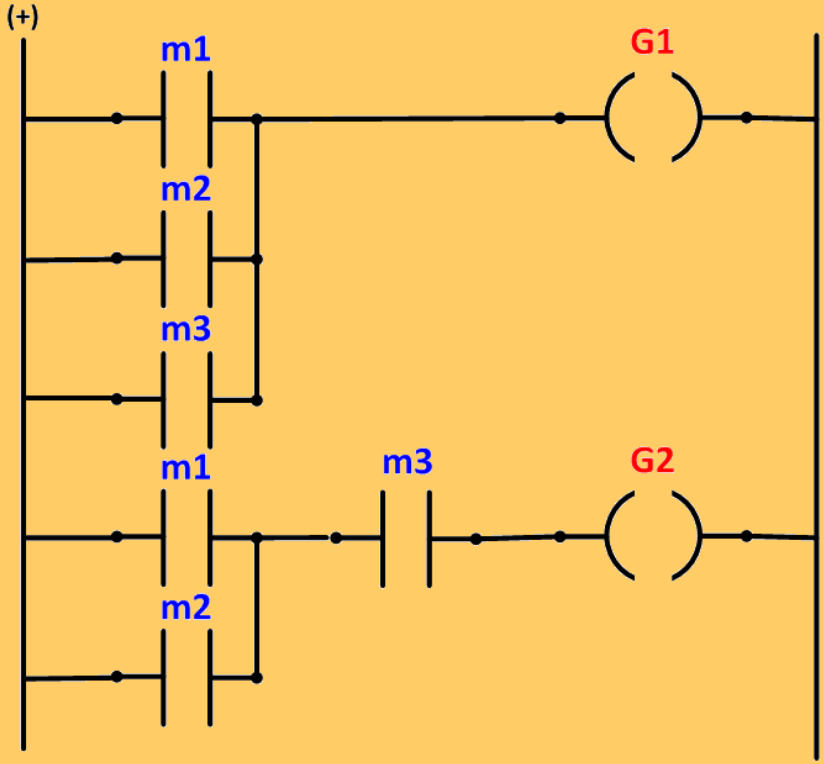
LÓGICA COMBINACIONAL

Un sistema es combinacional si las salidas sólo dependen del valor de las entradas en ese momento. El lenguaje de contacto es ideal para la programación de estos sistemas.



G1 = m1+m2+m3

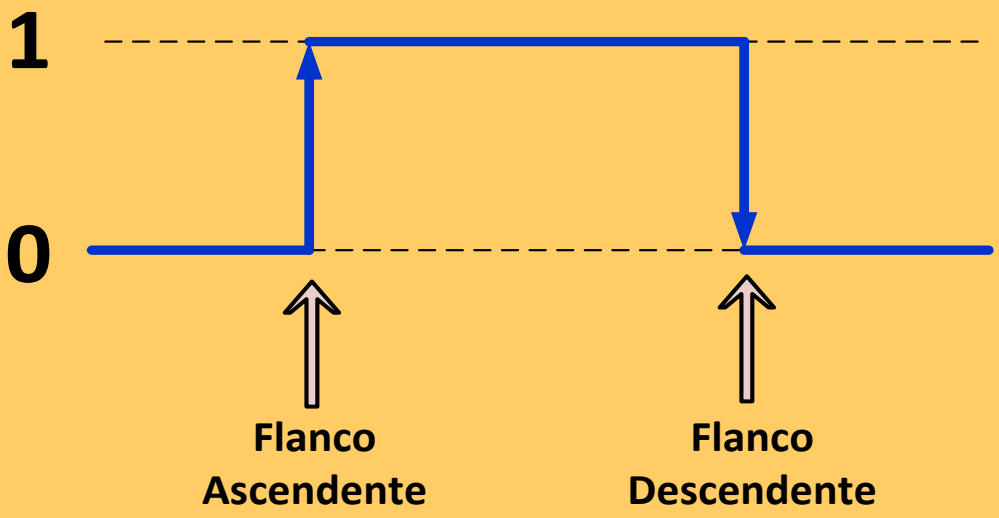
G2 = (m1+m2).m3



FLANCOS

Flanco ascendente.
Cuando una variable lógica pasa de 0 a 1.

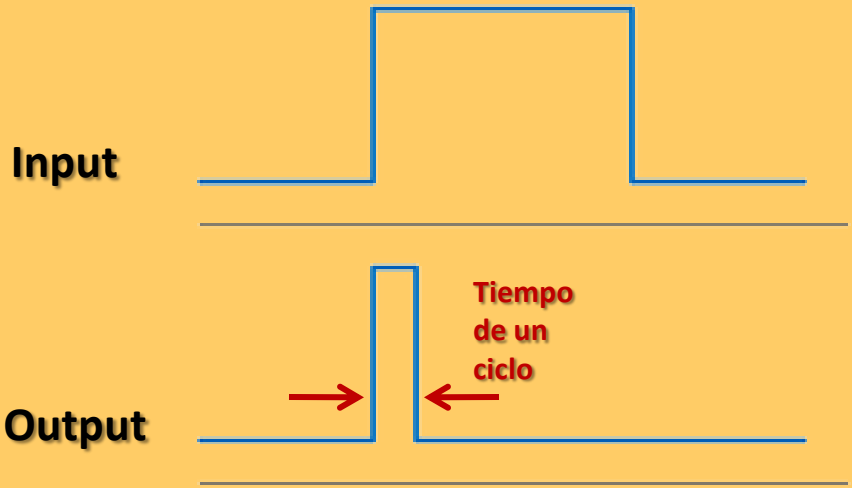
Flanco descendente.
Cuando una variable lógica pasa de 1 a 0.



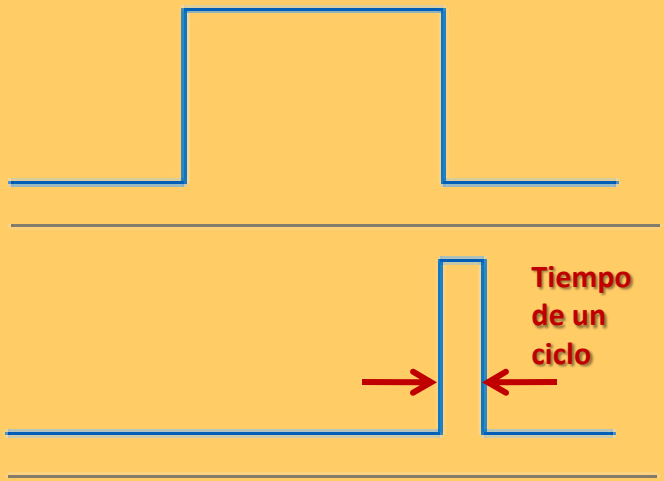
Estos elementos son útiles para detectar cambios de variables cuyo estado o evolución interesa controlar.

DETECTORES DE FLANCOS

Detector de Flanco ascendente.
La salida pasa de 0 a 1 durante un ciclo del PLC (pulso) cuando detecta un flanco ascendente.

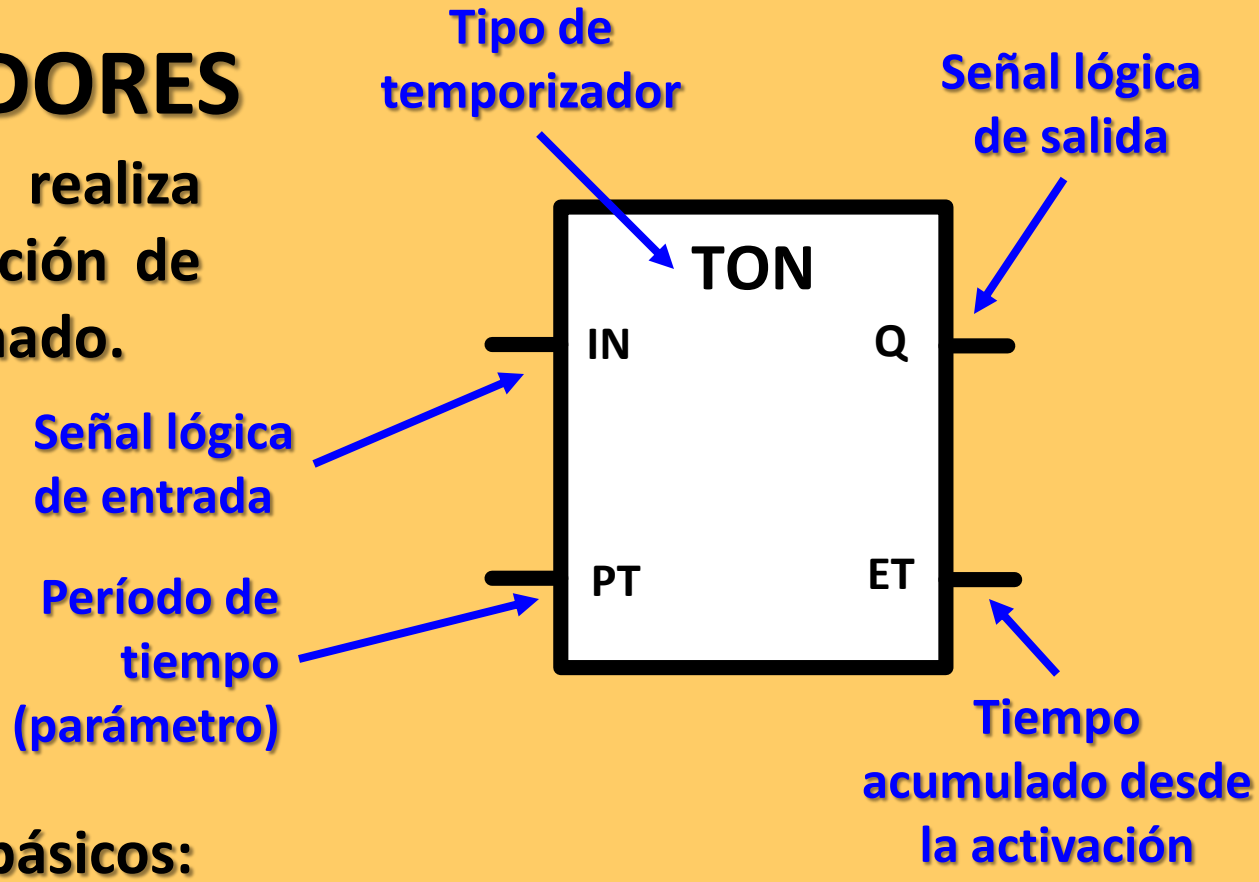


Detector de Flanco descendente.
La salida pasa de 0 a 1 durante un ciclo del PLC (pulso) cuando detecta un flanco descendente.



TEMPORIZADORES

Un temporizador realiza una acción en función de un tiempo programado.

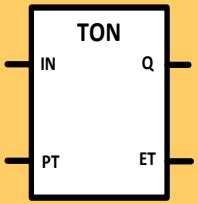


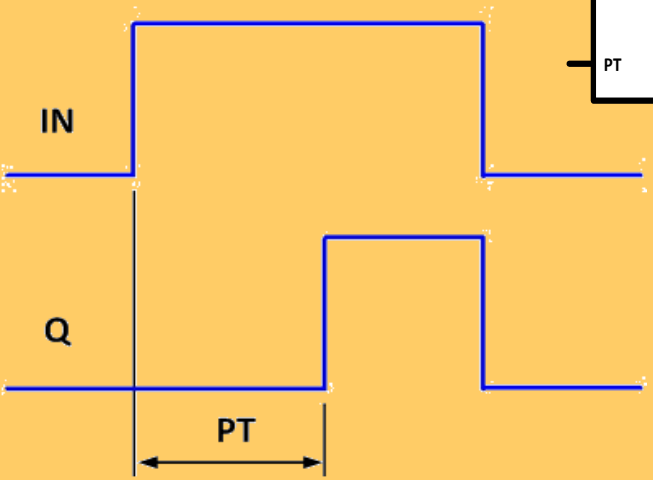
Hay tres modos básicos:

- **TON** – Temporizador con retardo en la activación
- **TOF** – Temporizador con retardo en la desactivación
- **RTO** – Temporizador activado por un pulso

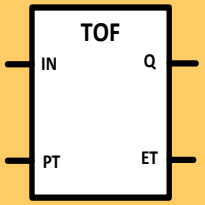
TEMPORIZADORES

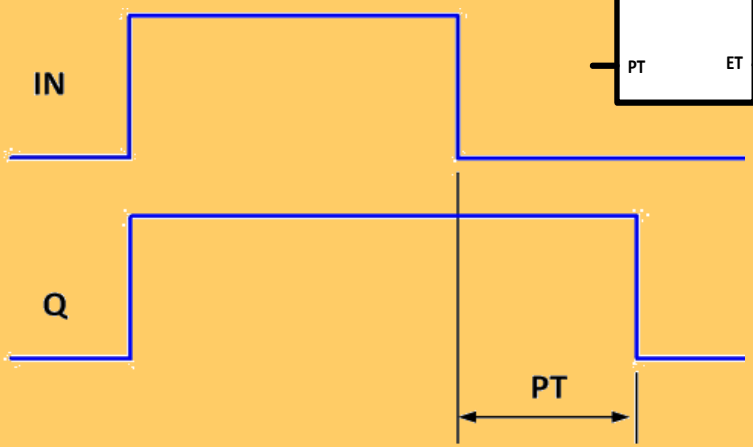
TON – Temporizador con retardo en la activación



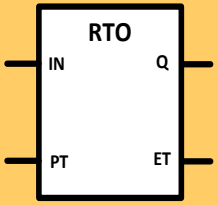


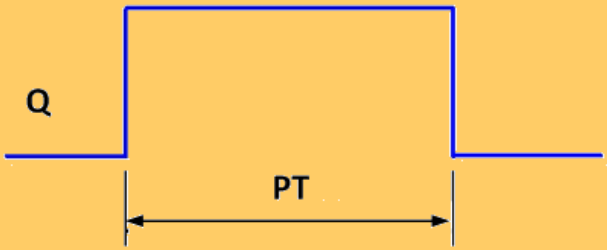
TOF – Temporizador con retardo en la desactivación





RTO – Temporizador activado por un pulso





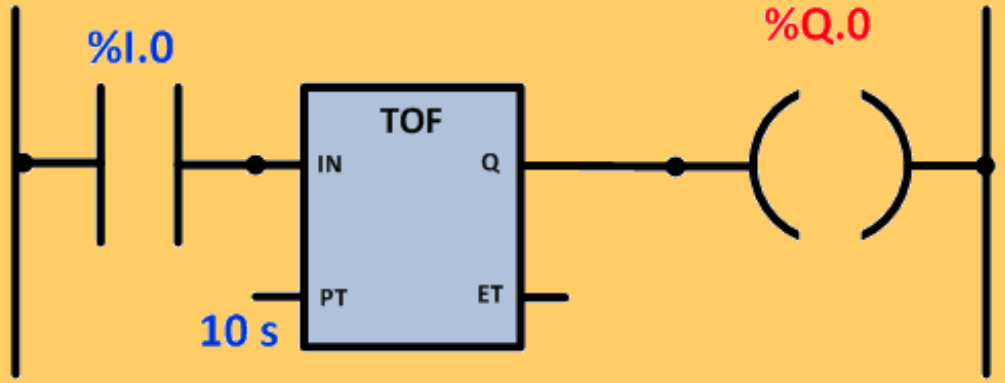
TEMPORIZADORES

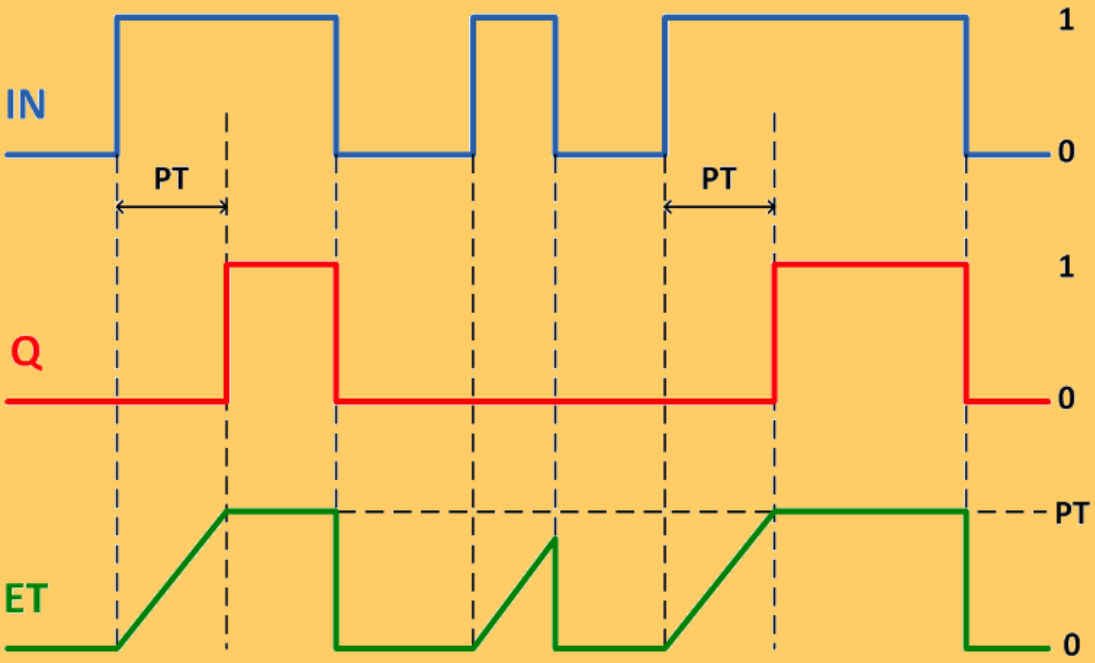
APLICACIÓN A UN LADDER

Un detector de proximidad activa una salida binaria (%I.0) cuando un objeto se encuentra a una distancia menor a 30 cm.

Se requiere que se encienda una luz (cierre un circuito con un foco %Q.0) después que hayan transcurrido 10 s de la detección del objeto.

Se usa un temporizador con retardo en la activación TON.





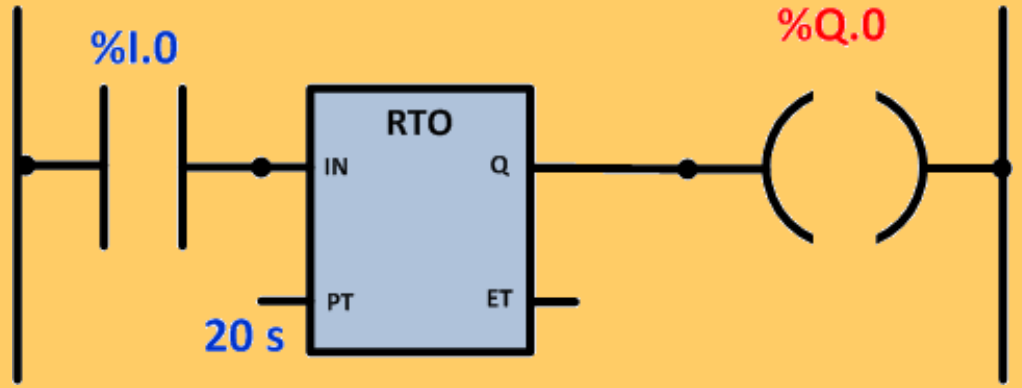
TEMPORIZADORES

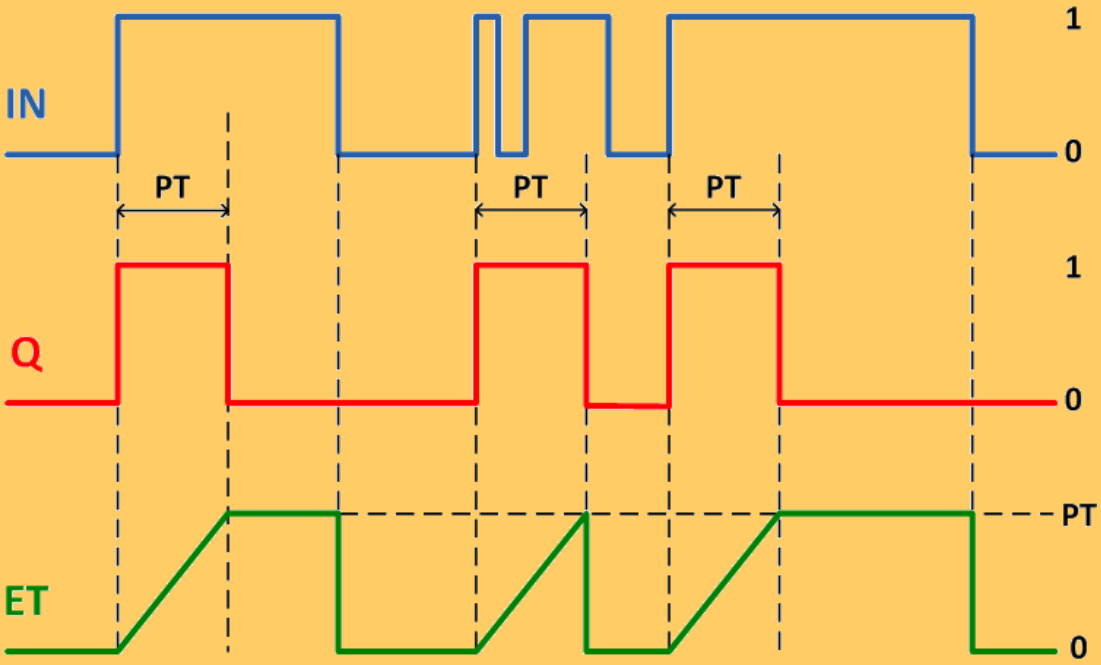
APLICACIÓN A UN LADDER

Un detector de proximidad activa una salida binaria (%I.0) cuando un objeto se encuentra a una distancia menor a 30 cm.

Se requiere que se encienda una luz (cierre un circuito con un foco %Q.0) y permanezca encendido por 20 s.

Se usa un temporizador activado por un pulso RTO.





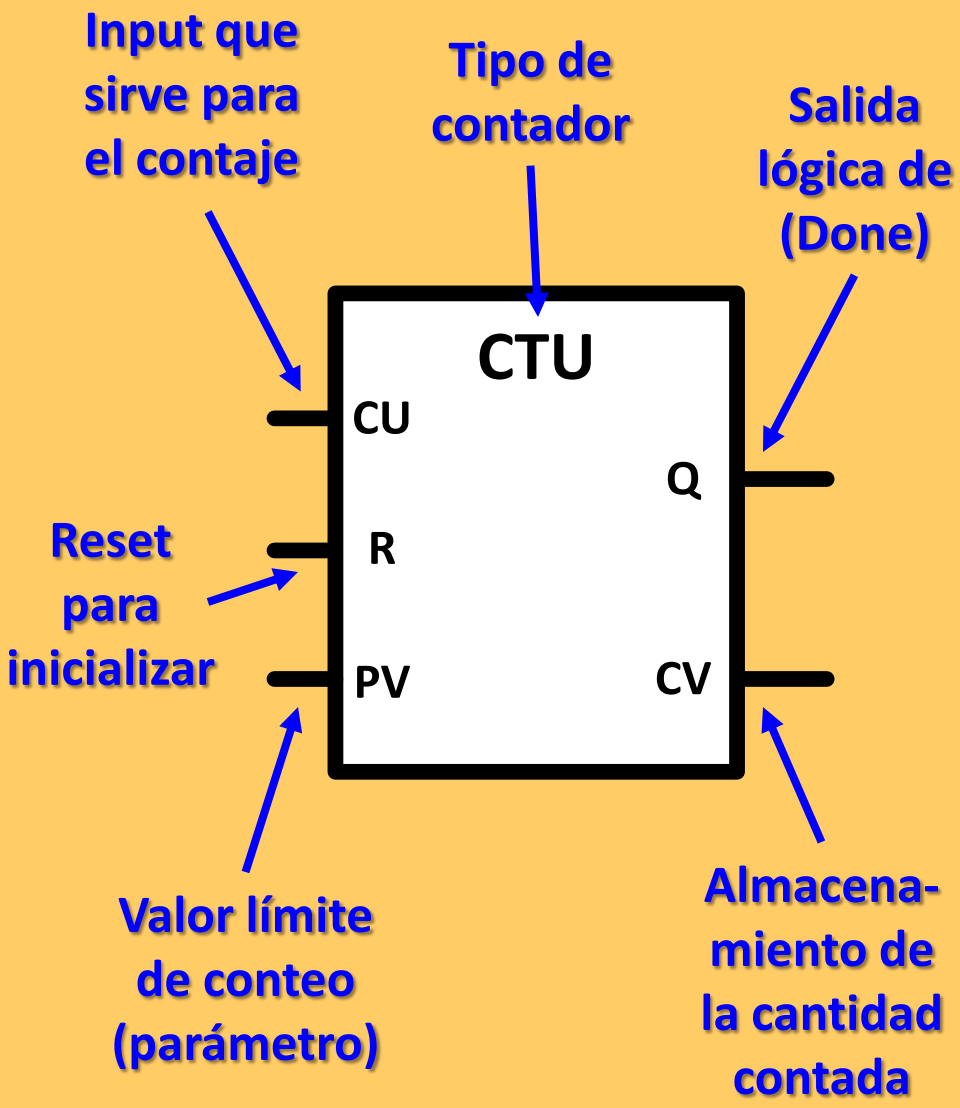
CONTADORES CTU

Se usan para contar eventos en orden ascendente.

Tiene tres entradas:

- PV: Valor de contaje (entero)
- CU: Detector de flanco ascendente que incrementa en una unidad el contador (que se almacena en CV).
- R: entrada booleana que sirve para resetear el contador, poniendo en cero CV.

La salida Q pasa de 0 a 1 cuando la cuenta, almacenada en CV, alcanza el límite indicado (PV). Q tomará el valor 1 cuando CV ≥ PV.



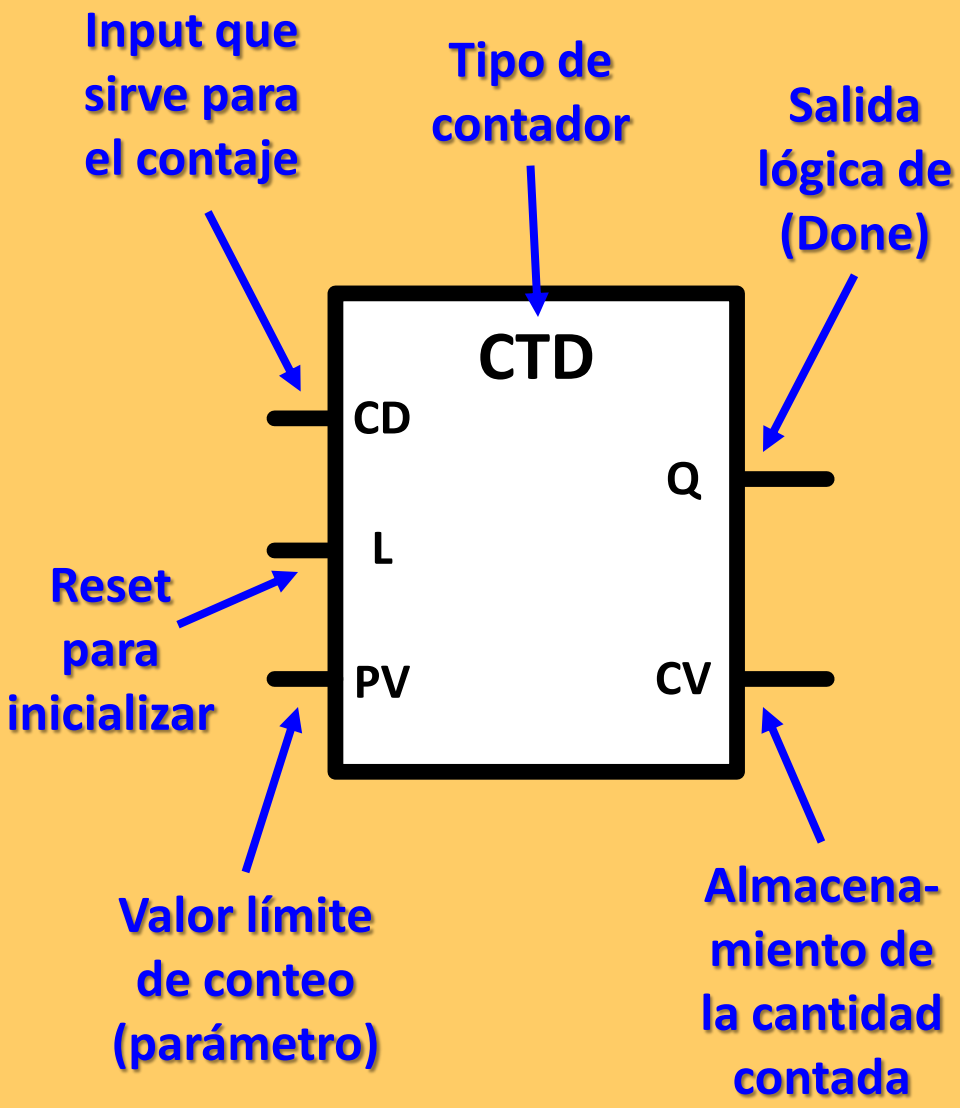
CONTADORES CTD

Se usan para contar eventos en orden descendente.

Tiene tres entradas:

- **PV: Valor de contaje (entero)**
- **CD: Detector de flanco ascendente que disminuye en una unidad el contador (que se almacena en CV).**
- **L: entrada booleana para resetear el contador, poniendo el valor de PV la salida CV.**

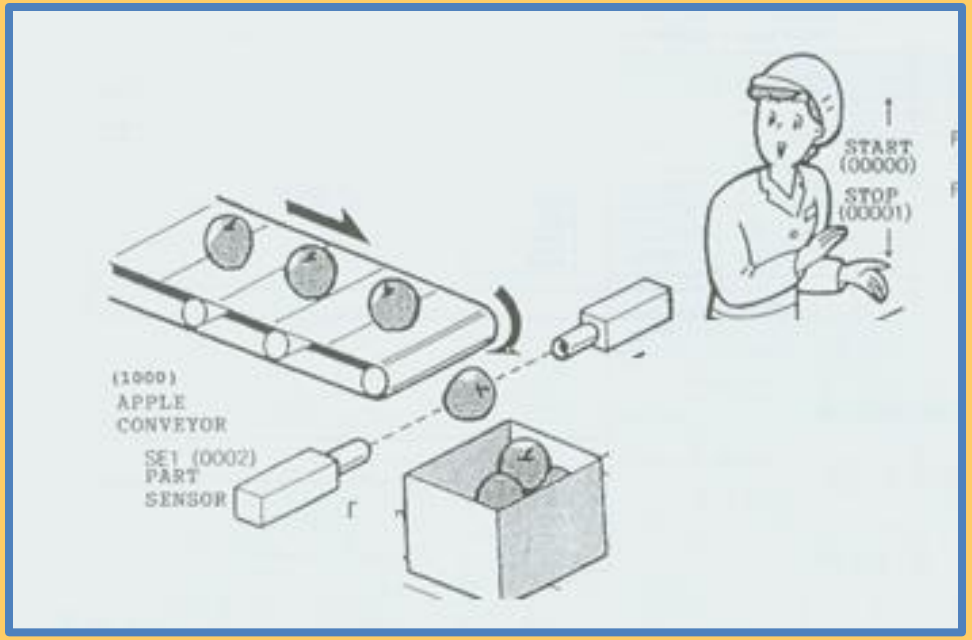
La salida Q pasa de 0 a 1 cuando la cuenta, almacenada en CV, alcanza el valor cero. Q tomará el valor 1 cuando CV sea igual a 0.

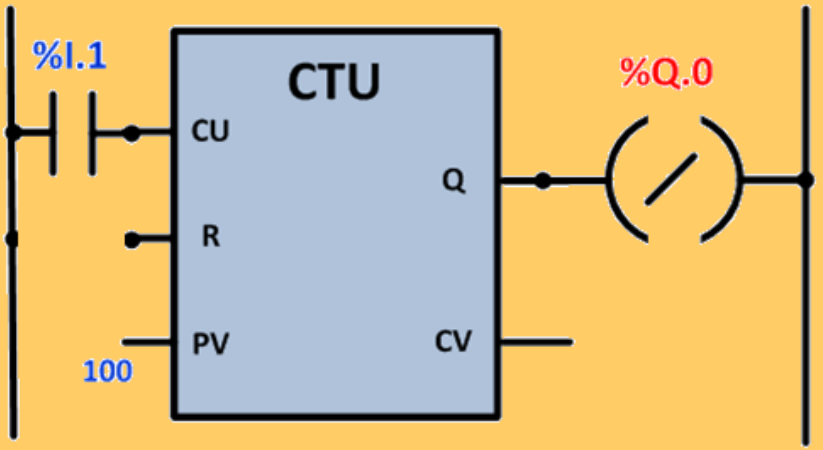


CONTADORES

APLICACIÓN A UN LADDER

Un autómata programable controlará las manzanas de una cinta transportadora de manera que cuando cuente 100 manzanas (usando un detector de proximidad %I.1), la cinta se detendrá (motor %Q.0 en off) hasta que un operario las almacene.



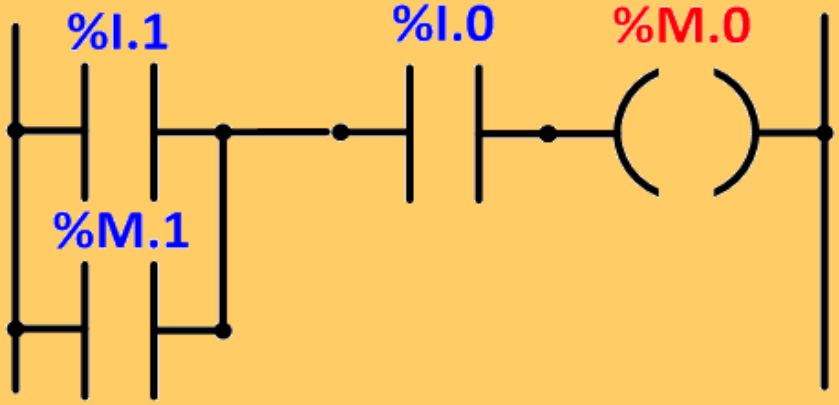


MARCAS O BITS INTERNOS

Son como relés auxiliares dentro del autómata. El resultado de una operación que termina en activar o desactivar una salida interna (M), que se almacena en la memoria.

Por lo tanto una marca es una salida "no visible", juega un papel muy importante en la programación.

Tiene un comportamiento parecido a las salidas aunque físicamente no se ve "nada activado". Una marca es una variable interna, que no tiene terminales exteriores.



En este caso, se almacena en %M.0 lo que resulta de la operación lógica:

%M.0 = (%I.1+%M.1).%I.0

Donde %M.1 corresponde a un valor almacenado en la memoria.

También puede almacenarse en memoria bytes (MB), palabras (MW) y palabras dobles (MD).