Norma IEC-61131

La Norma IEC-61131 se refiere a:

• Los autómatas programables (AP ó PLC´s) y a sus periféricos correspondientes, tales como:

- Los equipos de programación y depuración (PADT´s)
- Los equipos de ensayo (TE´s)
- Los interfaces hombre-máquina (MMI´s)

Esta norma no trata del sistema automatizado, del cual el autómata programable es un componente básico.

PADT: *Programming And Debugging Tool*
TE: *Test Equipment*
MMI: *Man-Machine Interface*

La finalidad de esta Norma IEC - 61131 es:

• Definir e identificar las *características principales* que se refieren a la selección y aplicación de los PLC´s y sus periféricos.

• Especificar los *requisitos mínimos* para las características funcionales, las condiciones de servicio, los aspectos constructivos, la seguridad general y los ensayos aplicables a los PLC´s y sus periféricos.

• Definir los *lenguajes de programación* de uso más corriente, las reglas sintácticas y semánticas, el juego de instrucciones fundamental, los ensayos y los medios de ampliación y adaptación de los equipos.

• Dar a los usuarios una *información* de carácter general y unas directrices de aplicación.

• Definir las *comunicaciones* entre los PLC´s y otros sistemas.

Partes de la Norma IEC 61131:

Parte 1: *Información general*

Parte 2: *Especificaciones y ensayos de los equipos*

Parte 3: *Lenguajes de programación*

Parte 4: *Guías de usuario*

Parte 5: *Comunicaciones*

ESTANDAR INTERNACIONAL

Parte 3: Lenguajes de Programación

Objeto y campo de aplicación

• Definir los *lenguajes de programación* de uso más corriente, las reglas sintácticas y semánticas, el juego de instrucciones fundamental, los ensayos y los medios de ampliación y adaptación de los equipos.

Es la interface entre el programador y el sistema de control

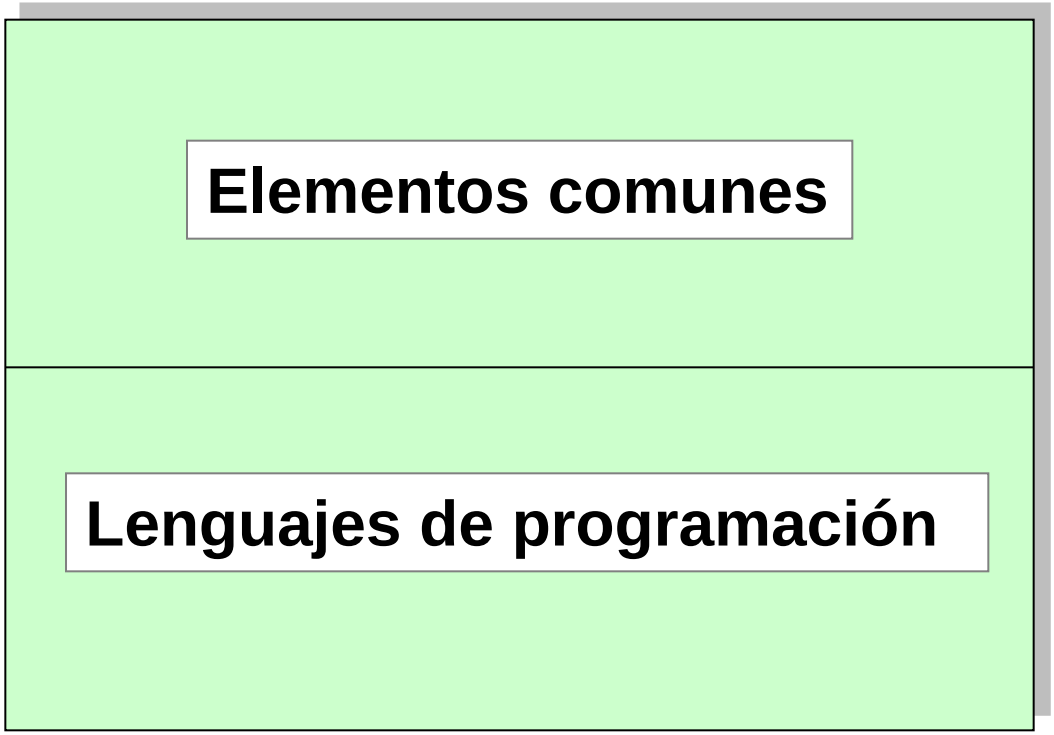
Parte 3: Lenguajes de Programación

Definiciones

Tiempo absoluto, vía de acceso, acción, argumento, matriz, asignación, bloque funcional biestable, cadena de bits, cuerpo, llamada, cadena de caracteres, comentario, compilar, tipo de datos, declaración, delimitador, doble palabra, flanco ascendente/descendente, función, diagrama de bloques funcionales, direccionamiento, valor, parámetro de entrada, instancia, literal entero, palabra clave, etiqueta, real largo, temporizador con retardo de conexión/desconexión, parámetro de salida, sentido de corriente, unidad de organización de programa, recurso, tarea, retorno, etapa, secuencia, transición, representación simbólica, etc...

Parte 3: Lenguajes de Programación

El estándar IEC-61131



Parte 3: Lenguajes de Programación

Elementos comunes

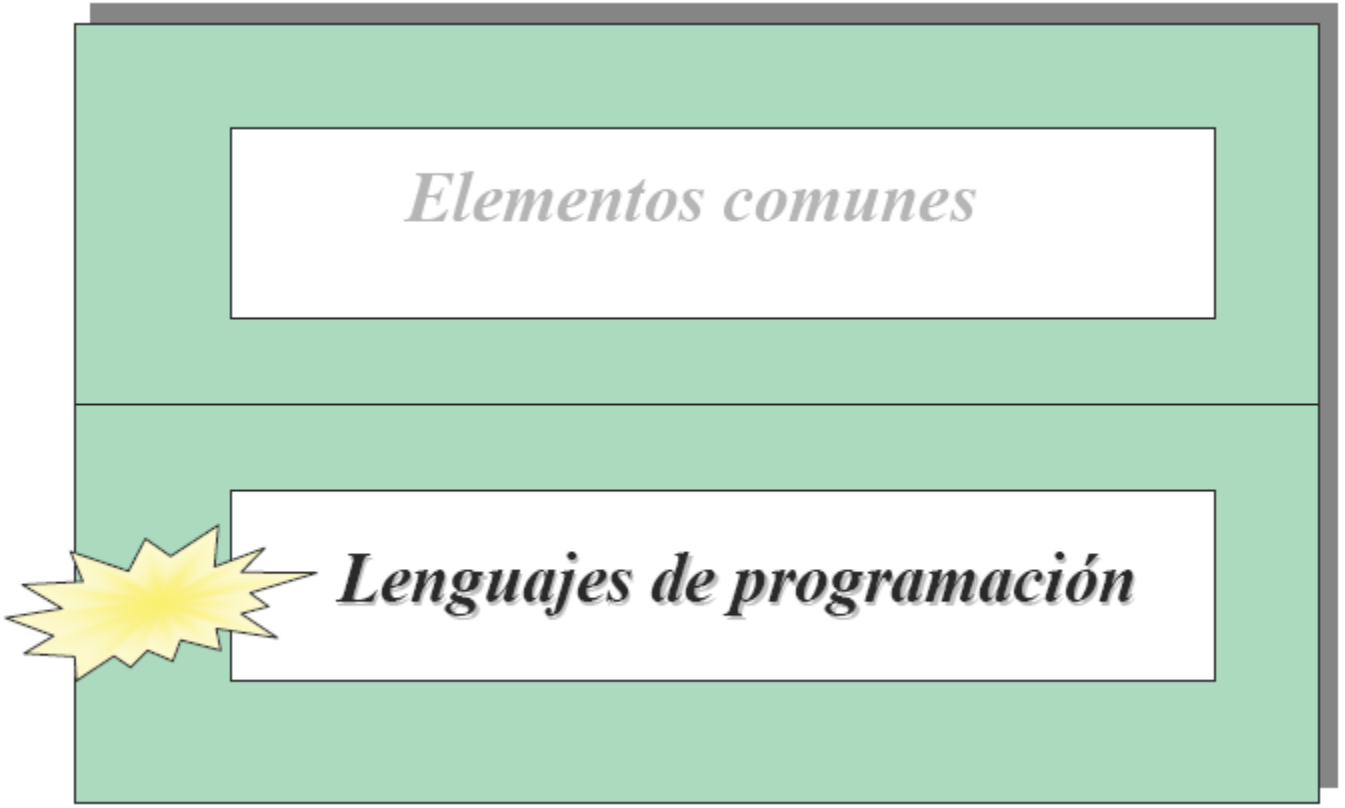
- Tipos de datos y variables
- Modelo de software
- Modelo de comunicación de datos
- Modelo de programación
- Unidades de organización del programa
- Gráfico Funcional Secuencial (**SFC**)
- Elementos de configuración

Lenguajes de programación

- Lista de instrucciones (**IL**)
- Texto estructurado (**ST**)
- Diagrama de bloques funcionales (**FBD**)
- Diagrama de contactos (**LD**)

Parte 3: Lenguajes de Programación

El estándar IEC-61131



Parte 3: Lenguajes de Programación

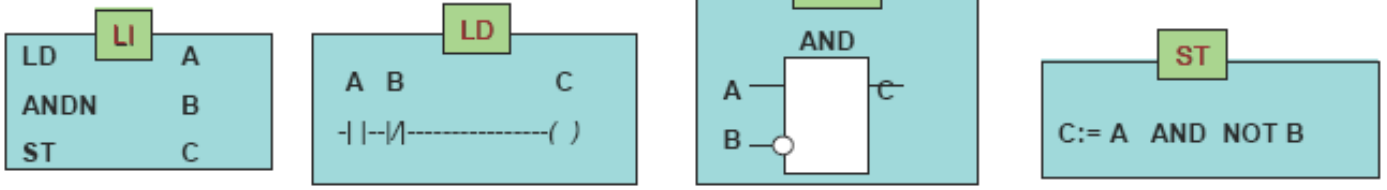
IEC 1131-3

Elementos Comunes

- Variables, tipos de datos y declaraciones
- Diseño, proyectos y tareas
- Funciones, bloques de funciones y programas
- *Sequential Function Charts*

Diseño
Proyectos
Tareas
Variables Globales
Caminos de Acceso

Lenguajes de Programación



Parte 3: Lenguajes de Programación

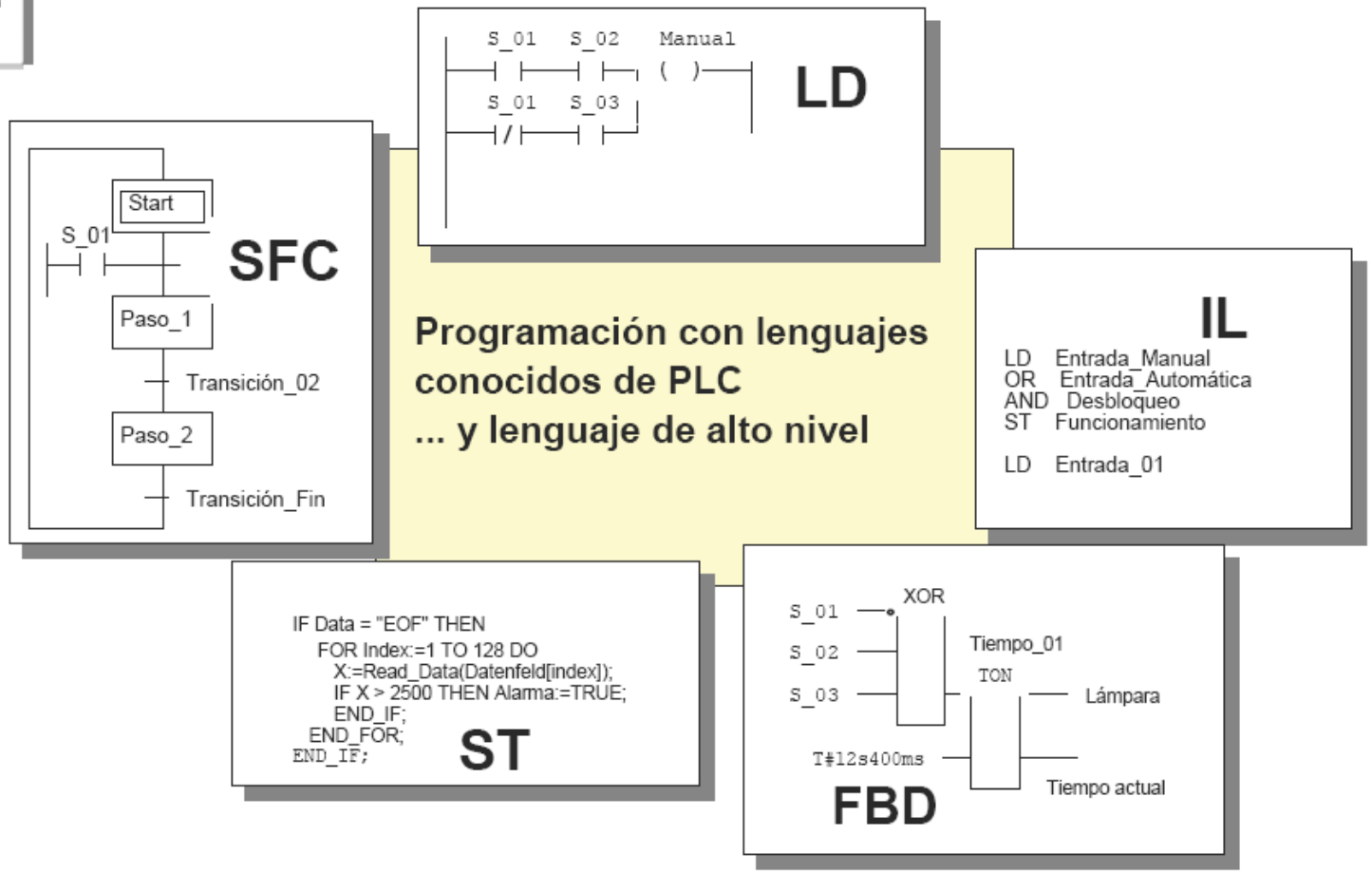
4 Lenguajes de programación

- **Lenguajes gráficos**
  Diagrama de escalera (“Ladder Diagram”, **LD**)
  Diagrama de Bloques Funcionales (”Function Block Diagram, **FBD)**

- **Lenguajes literales**
  Lista de instrucciones (“Instruction List”, **IL**)
  Texto estructurado (“Structured Text”, Texto estructurado (“Structured Text”, **ST ST**)

La ***selección del lenguaje de programación*** depende de la experiencia del programador, de la aplicación concreta, del nivel de definición de la aplicación, de la estructura del sistema de control y del grado de comunicación con otros departamentos de la empresa...

Parte 3: Lenguajes de Programación



Parte 3: Lenguajes de Programación

- **LD**: Diagrama a contactos o de escalera (Ladder Diagram).
- **IL**: Lista de Instruccion
- **FBD**: Diagrama de F (Function Block D
- **ST**: Texto Estructu
- **SFC**: Carta de Func (Sequential Function Char

Es una representación gráfica de expresiones booleanas, combinando contactos (condiciones) con bobinas (resultados) similar a un diagrama de contactos eléctricos.

Parte 3: Lenguajes de Programación

• **LD**: Diagrama a contactos o de escalera (Ladder Diagram).

• **IL**: Lis[illegible] Instrucciones (Instruc[illegible] List)

• **FBD**: Di[illegible] (Function

• **ST**:

• **SFC**: (Sequential F[illegible]

Su estructura principal es una lista de instrucciones, donde cada instrucción debe ocupar una nueva línea. Cada línea contiene un operador, que es completado por modificadores opcionales y uno o más operandos, si la operación específica lo requiere.

Parte 3: Lenguajes d

Consiste en una representación gráfica de diferentes tipos de ecuaciones. Los operadores son representados por cajas rectangulares de funciones y los operandos se conectan a sus lados izquierdo (entradas) y derecho (salidas).

• **LD**: L
(Ladd

• **IL**: Li

• **FBD**: Diagrama de Bloque s
(Function Block Diagram).

• **ST**: Texto Estructurado (Structured Text).

• **SFC**: Carta de Funciones Secuenciales (Sequential Function Chart).

Parte 3: Lengua...

Un programa en ST es una lista de sentencias ST. Cada sentencia termina en un separador ";" y se incluye dentro de uno de los tipos básicos de: asignación, selección, iteración, control o especiales. Los nombres usados en el código fuente (identificadores de variables, constantes, palabras reservadas del lenguaje, ...) se desagrupan usando separadores inactivos o activos.

- **LD**: ...
(Lad...

- **IL**: ...

- **FBD**: D...
(Function B...

- **ST**: Texto Estructurado (Structured Text).

- **SFC**: Carta de Funciones Secuenciales (Sequential Function Chart).

Parte 3: Lenguajes de Programa

Es un conjunto gráfico de pasos y transiciones enlazados por conexiones orientadas. Cada transición es atada a una condición booleana. Las acciones de los pasos son detalladas usando otros lenguajes (ST, IL, LD, FBD).

- **LD**: Dia
(Ladder
- **IL**: L
- **FBD**:
(Function
- **ST**: Texto E urado (ed Text).
- **SFC**: Carta de Funciones Secuenciales (Sequential Function Chart).

Parte 3: Lenguajes de Programación

Tipos de operandos de uso común en PLCs

La IEC 1131-3 recoge todos los tipos de operandos de uso común en PLCs. En su apartado 2.2 (Representación exterior de los datos) se establece que dicha representación deberá consistir en *literales numéricos*, *literales de cadenas de caracteres* y *literales de tiempo*.

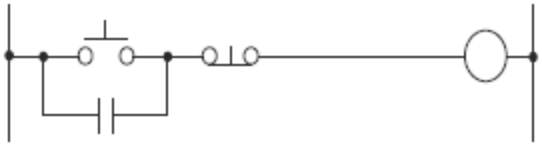
A partir de ello en el sistema ISaGRAF (IEC 1131-3 compatible) de CJ International se agrupan en cuatro tipos básicos: ***Booleano, Analógico, Temporizado y Mensaje***.

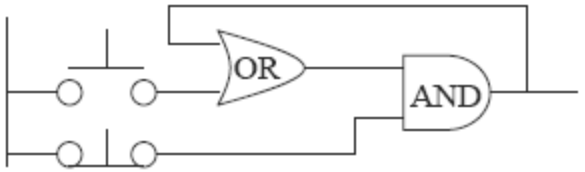
Parte 3: Lenguajes de Programación

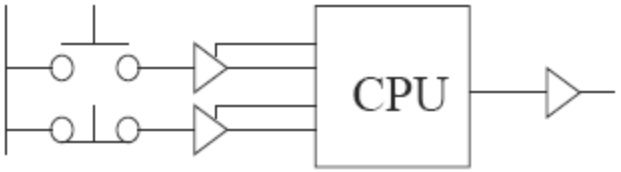
Lenguaje LD

Origenes del LD

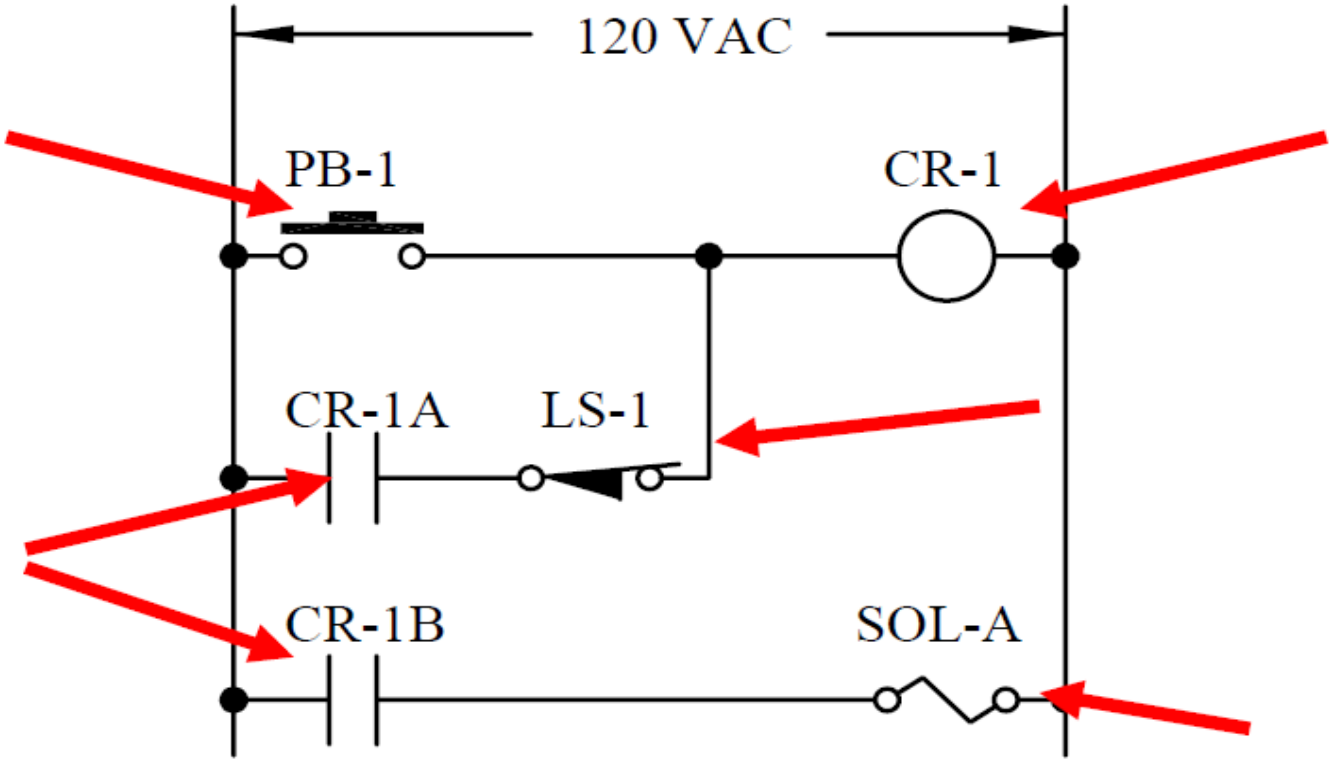
- Su origen es la representación gráfica utilizada en el diseño de sistemas de control eléctricos
  - Las decisiones de control se hacen efectivas activando relés
- Despues los relés se sustituyeron por circuitos lógicos
  - Las decisiones de control se hacen efectivas en función de las salidas de la puertas lógicas
- Finalmente las CPUs sustiteyen los complejos y amplios circuitos lógicos
  - Las E/S se cablean con *buffers*
  - Las decisiones de control son programas en ejecución
- La representación de la lógica de relés evolucionó para una creación y comprensión más sencilla de los programas
  - Reduce el tiempo de formación de los programadores







Componentes de Control Lógico.



Lógica Booleana Básica

Reglas de simbolización

Regla I.

Cada uno de los enunciados simples del lenguaje natural se sustituirá por variables proposicionales simbolizadas por letras minúsculas, p, q, r, s, t…

Regla II.

Las expresiones del lenguaje natural tales como "no", "no es cierto", "no es el caso que" "es falso", "es imposible" y todas aquellas que sean equivalentes, se sustituirán por el símbolo ¬

Llueve, p; No llueve: ¬ p

Regla III.

Las expresiones del lenguaje natural tales como "y", "ni" "pero", "que", "mas", y todas las que sean equivalentes, se sustituyen por el símbolo ∧

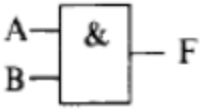
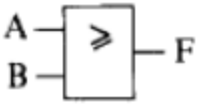
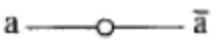
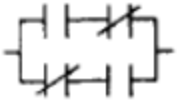
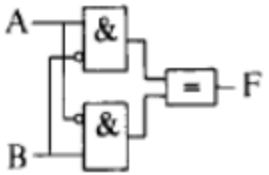
Llueve: p; Hace frío: q; Llueve y hace frío: p ∧ q;

Regla IV.

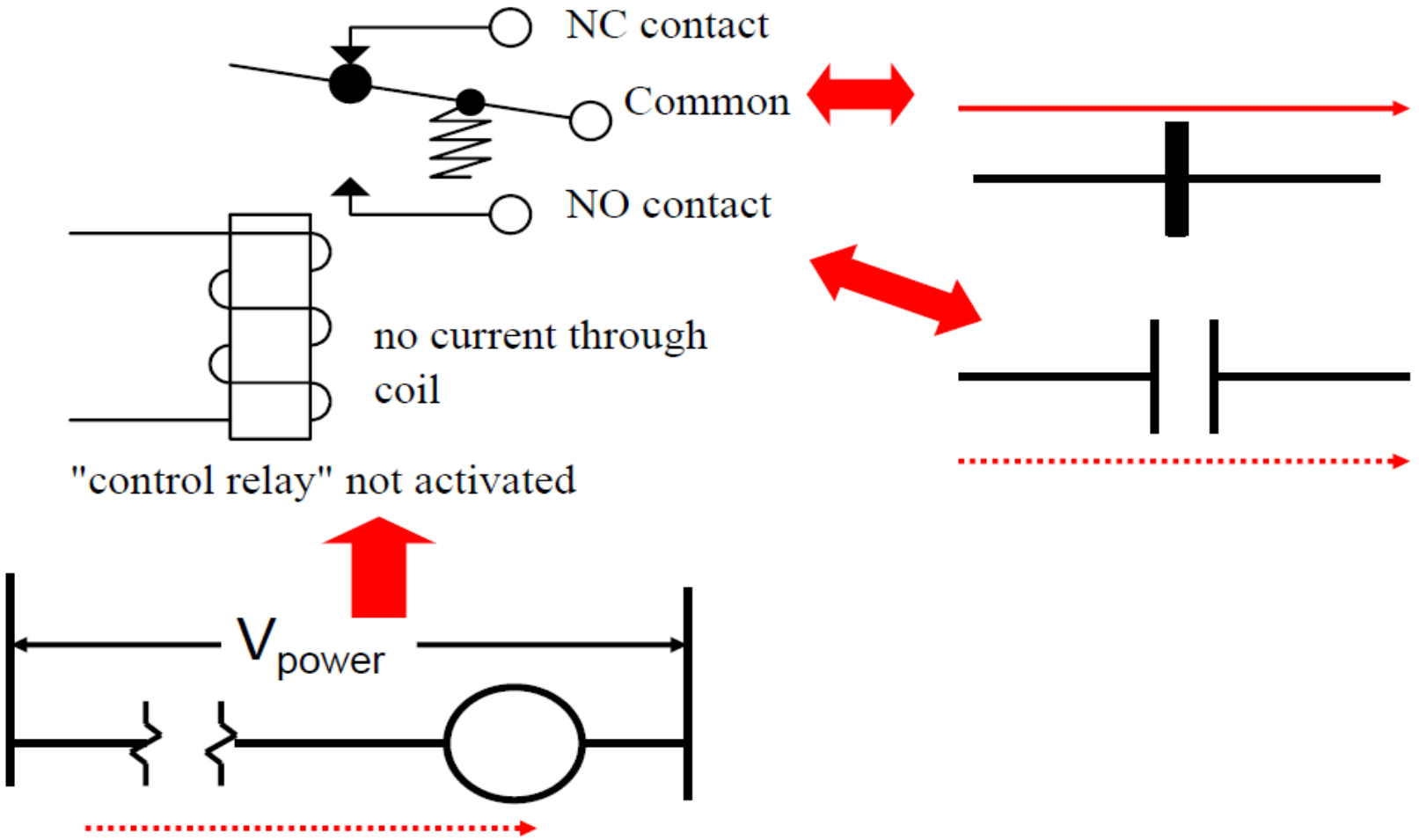
Las expresiones del lenguaje natural tales como "o", "o...o", "bien...bien", "ya...ya", y sus equivalentes, se sustituyen por el símbolo ∨

Llueve: p; Hace frío: q; O llueve o hace frío: p ∨ q

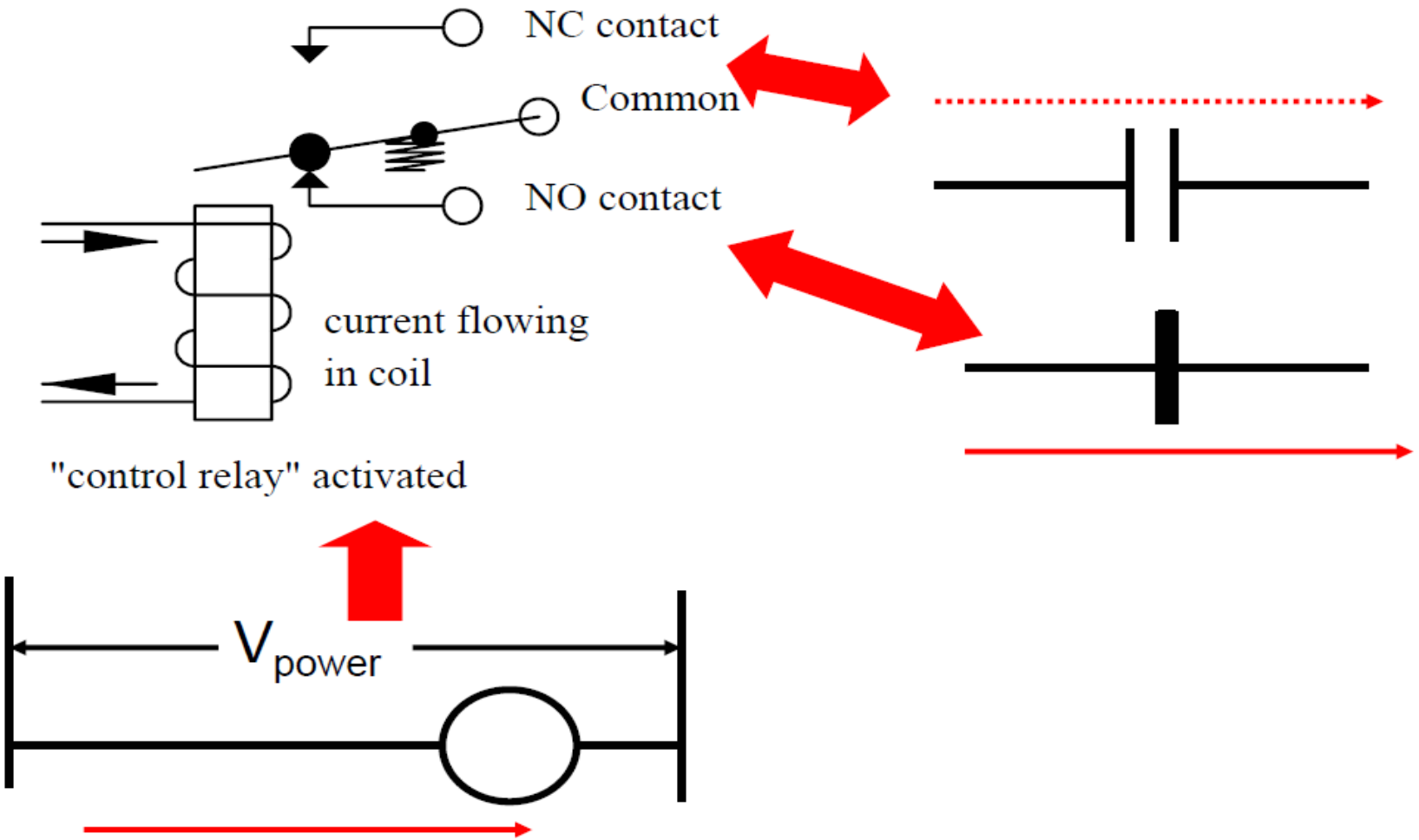
Lógica Booleana.

Norma / Función	Nemónicos	Boole	DIN-40713-6 (relés)	NEMA (contactos)	Símbolos lógicos	Operadores lógicos UNE-20-004-75 (XVI)
Y (Serie)	AND	•				
O (Paralelo)	OR	+				
Complemen-taria	NOT	$\overline{a}$				
Exclusiva	XOR	⊕				

Relé de Control No Activado:



Relé de Control Activado:



Componentes Normalmente Abiertos:

Limit Switch

Momentary Contact Pushbutton

Pressure Switch

Manual Switch

Contacts

Componentes Normalmente Cerrados:

Limit Switch

Momentary Contact Pushbutton

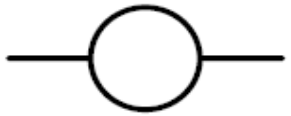
Pressure Switch

Manual Switch

Contacts

Componentes de Salida:

Solenoid Coil



Control Relay Coil

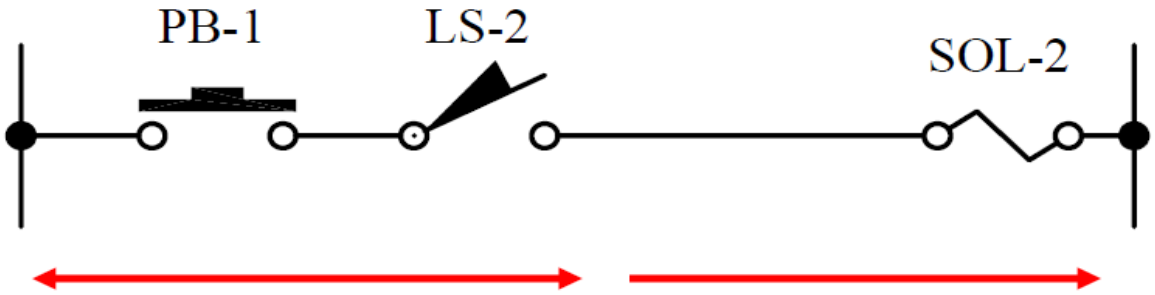
Lamp

Annunciator (Horn)

¿ Por qué es llamado Control Lógico ?

IF _____ **AND** _____

THEN _____

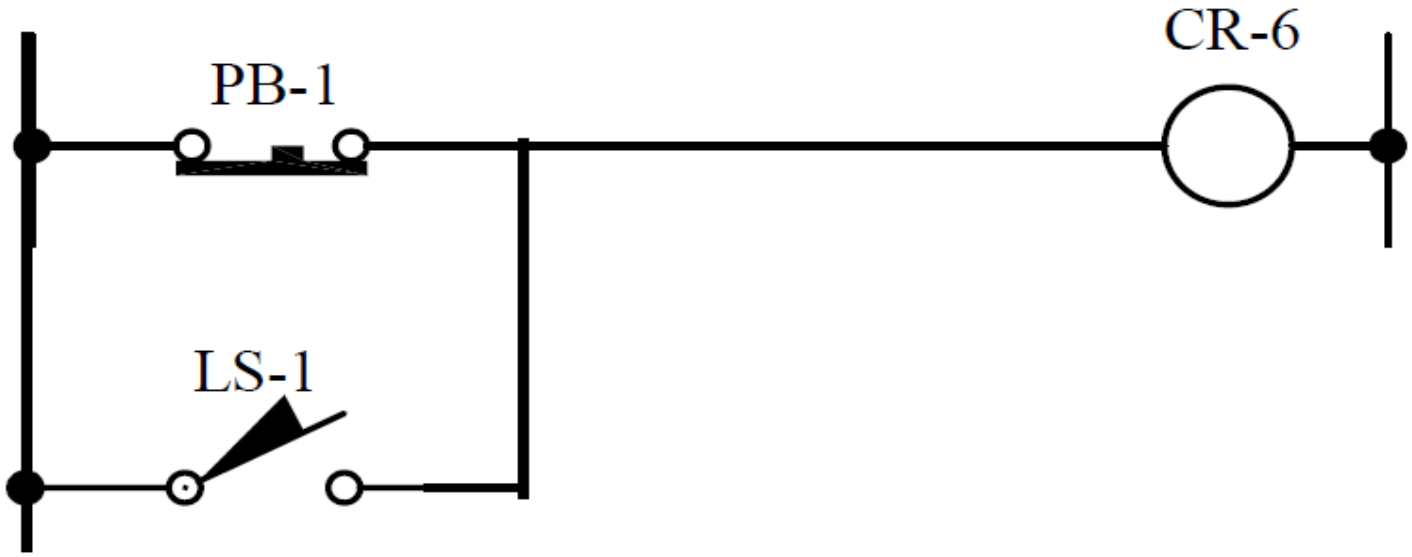


IF there is continuity across the inputs

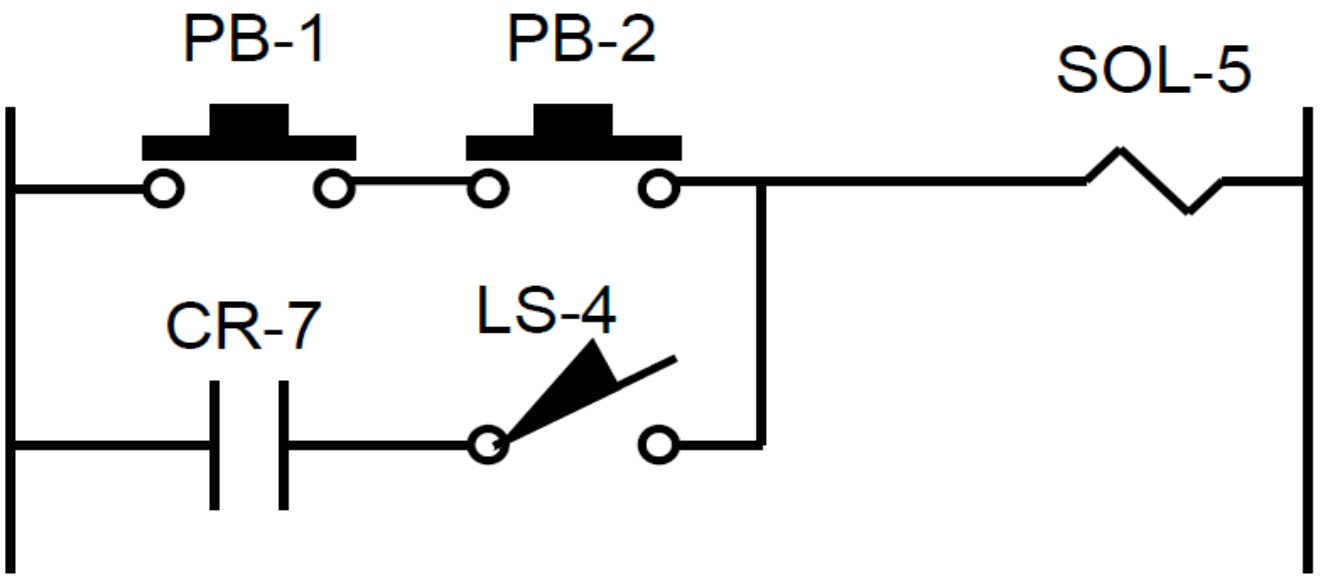
Ejemplo OR:

IF _____ **OR** _____

THEN _____

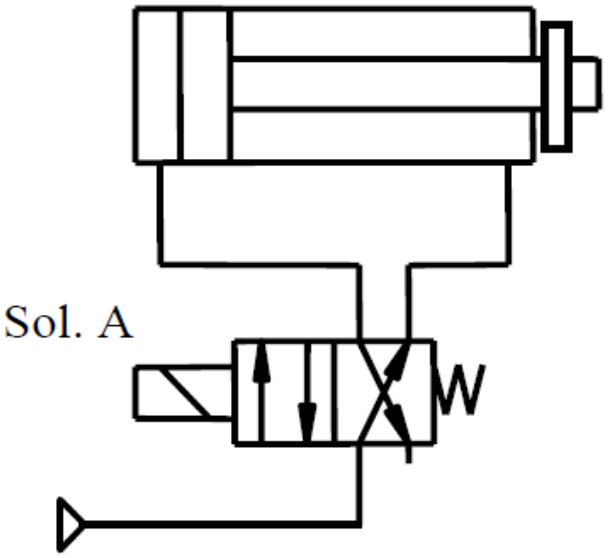


Completa la Lógica:

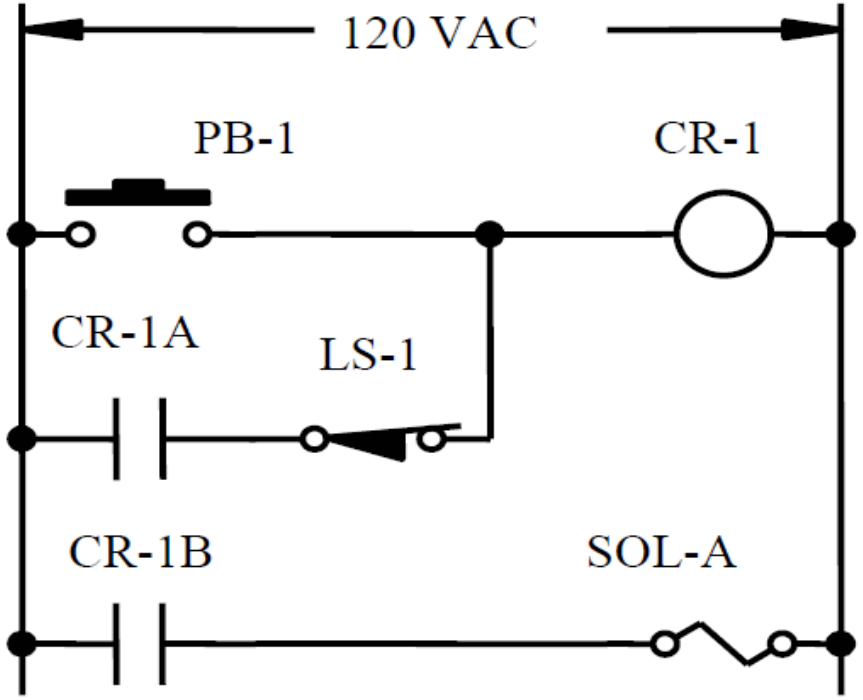


IF { _____ **AND** _____ }
OR { _____ **AND** _____ }
THEN _____

- Pressing the pushbutton PB-1 will cause the cylinder to extend and retract one time

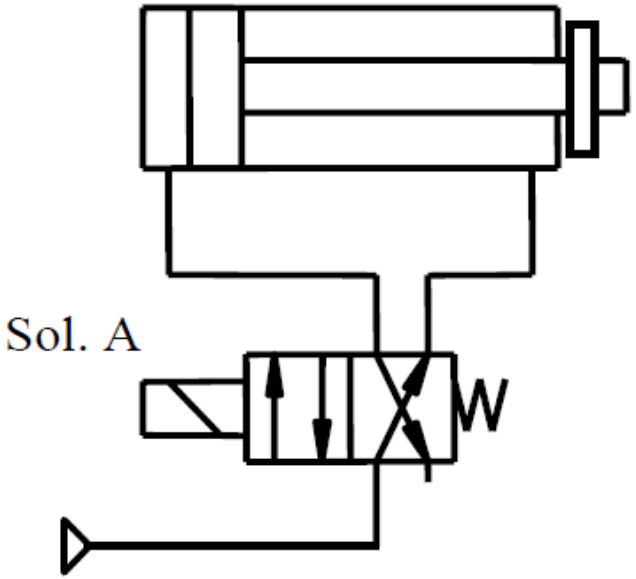


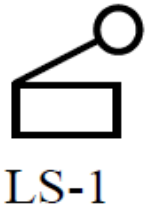
Pneumatic System

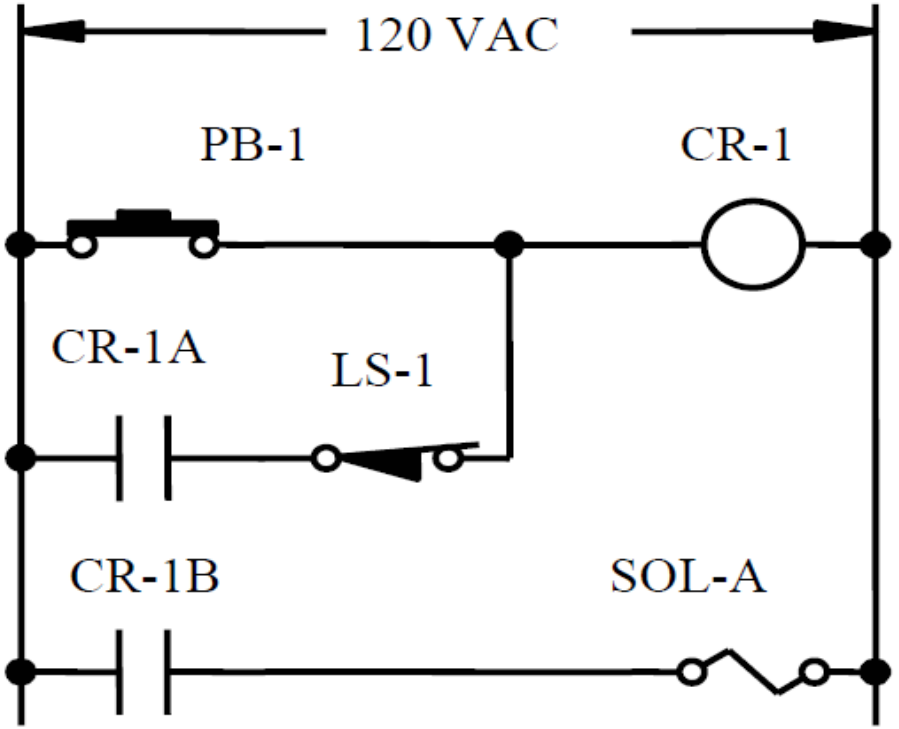


"Ladder" Logic Control

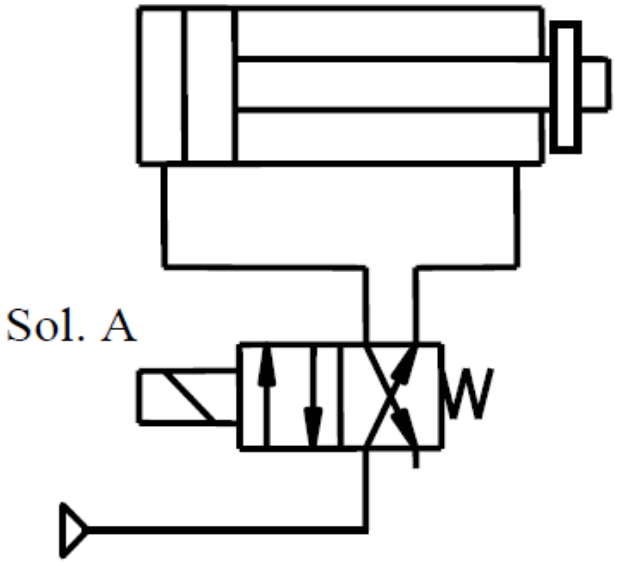
- Pressing the momentary contact pushbutton PB-1 energizes the control relay CR-1

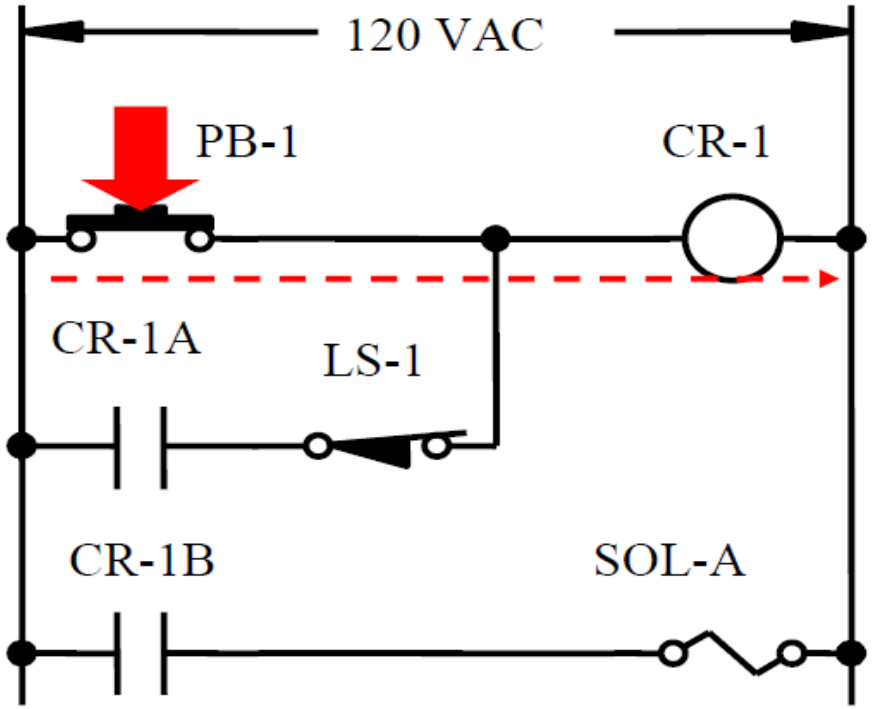




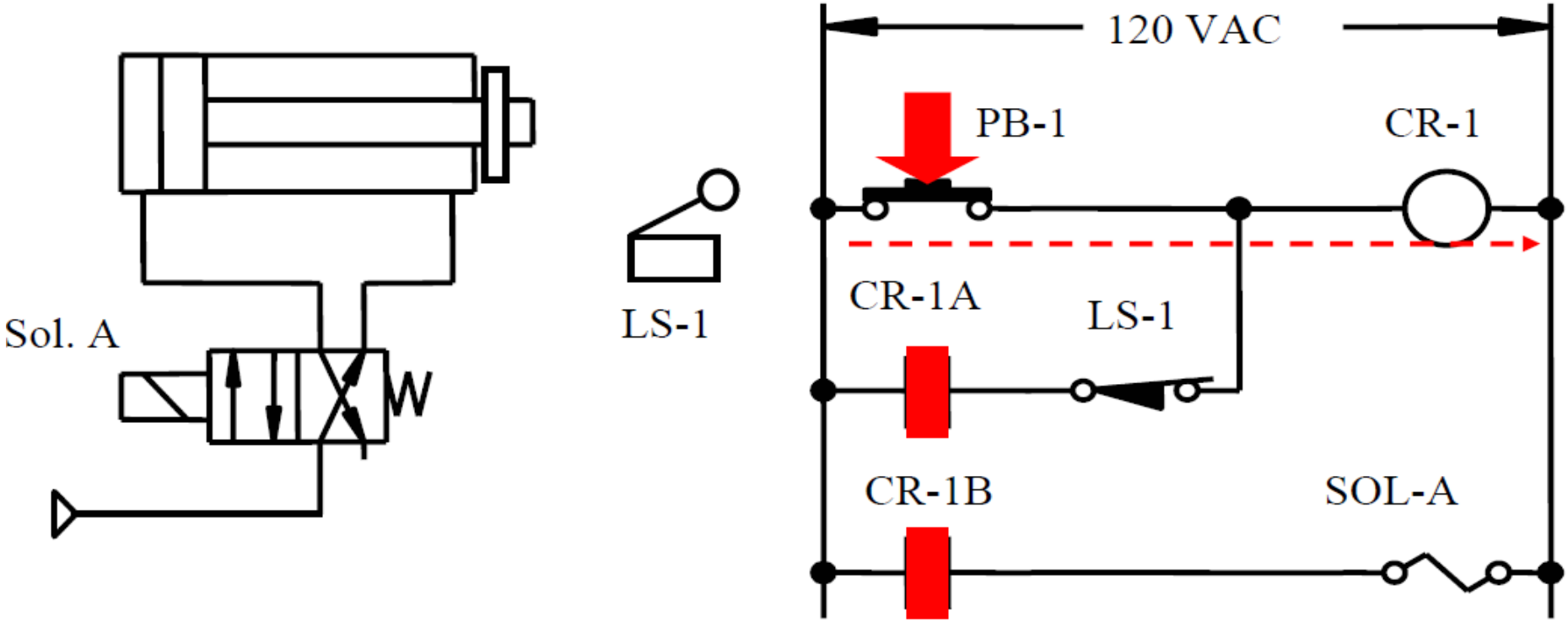


- After control relay CR-1 energizes, normally open contacts CR-1A and CR-1B activate

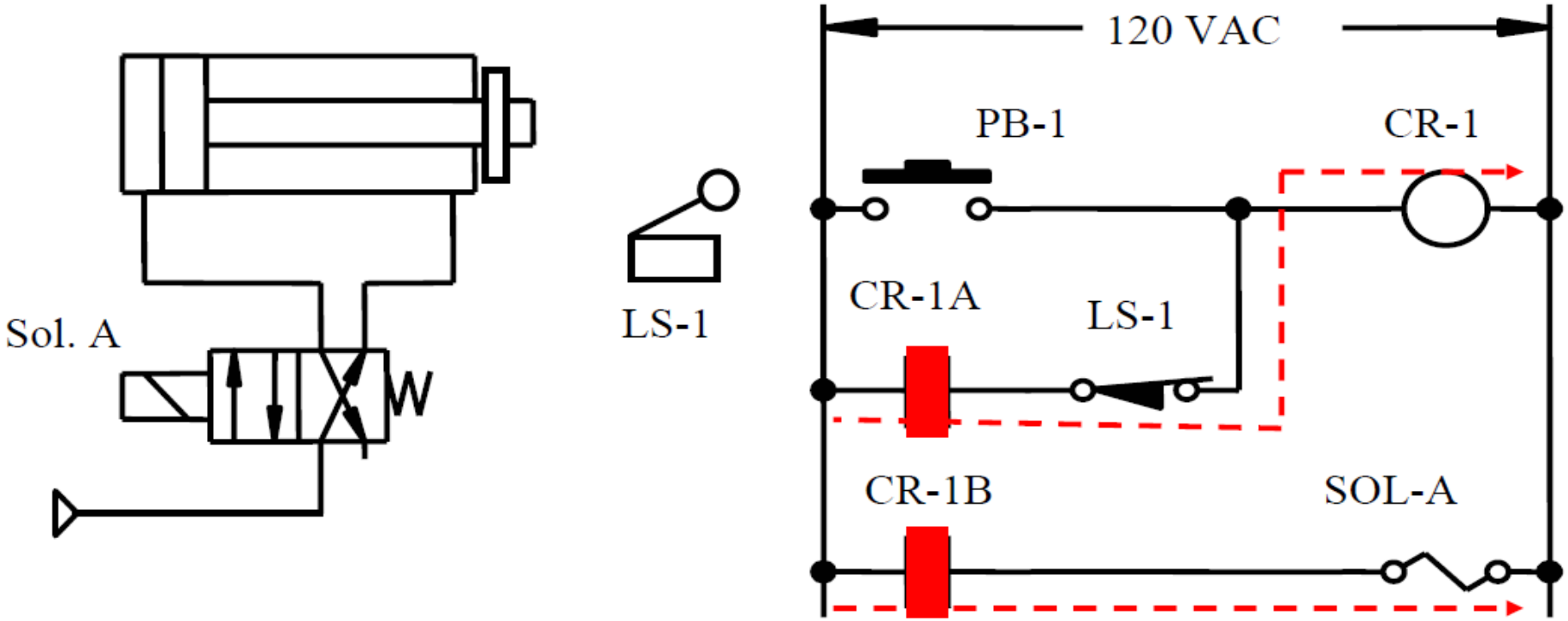




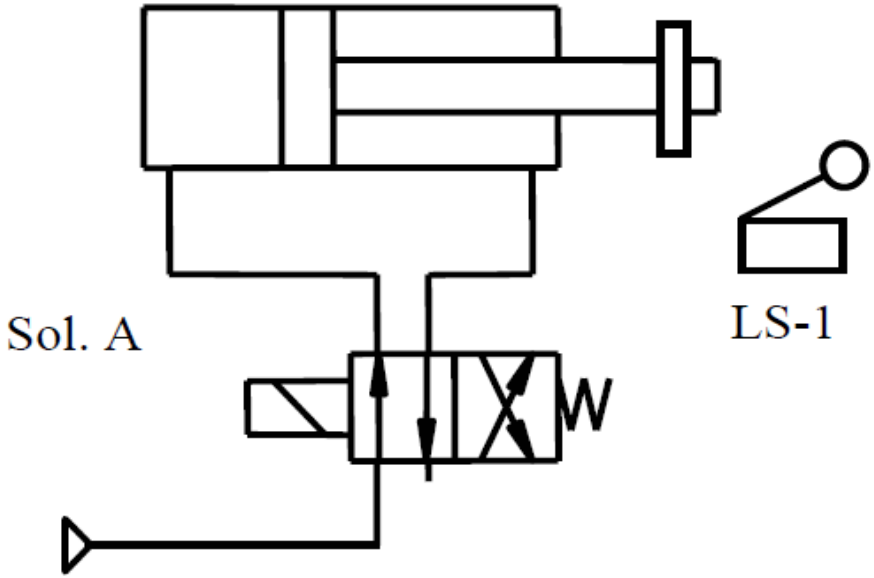
- Control relay CR-1 is now energized by a 2nd path, solenoid SOL-A also activates

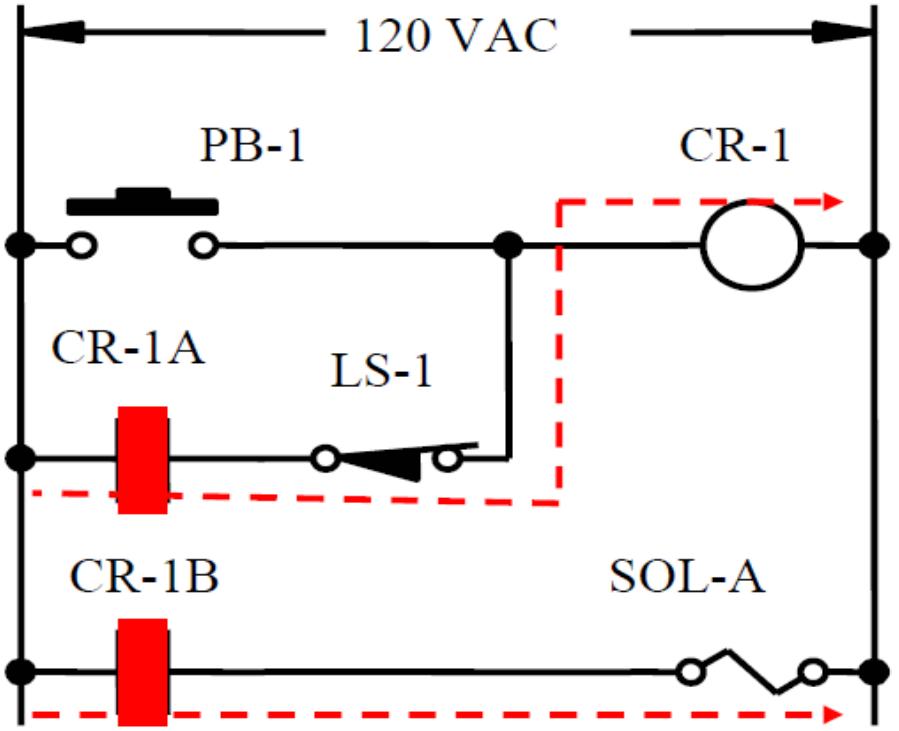


- PB-1 is released, but control relay CR-1 is still energized by the 2nd path ("hold" circuit)

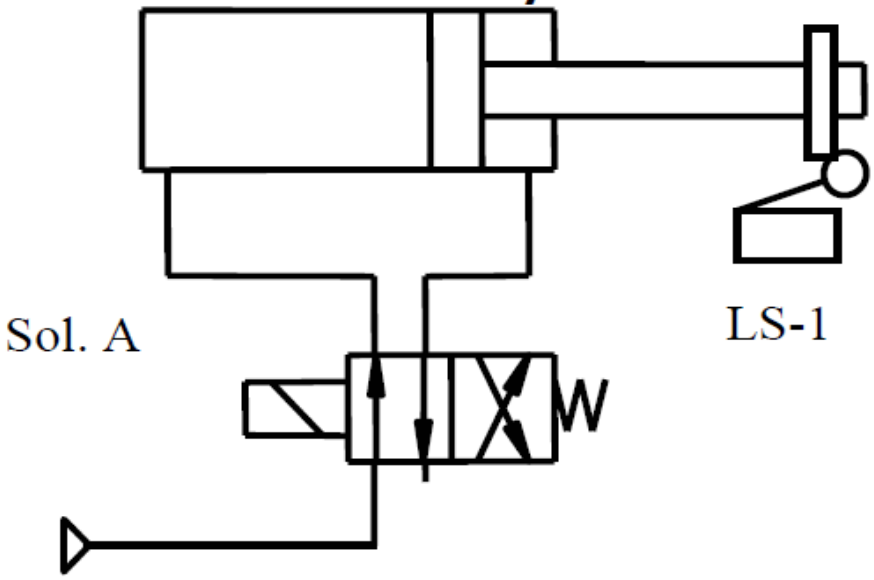


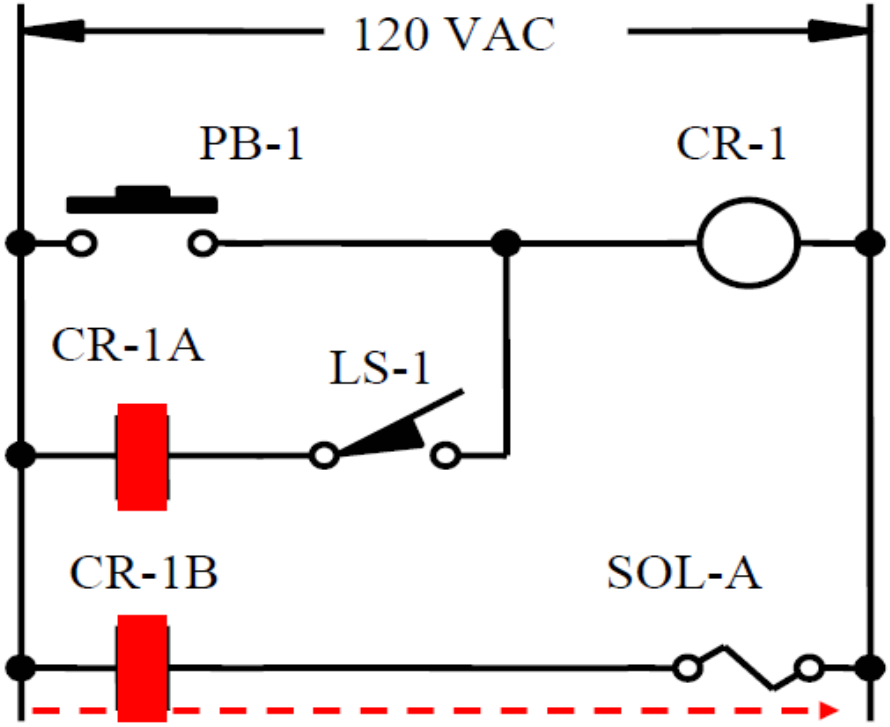
- Solenoid A shifts the valve spool to the right, and the cylinder begins to extend



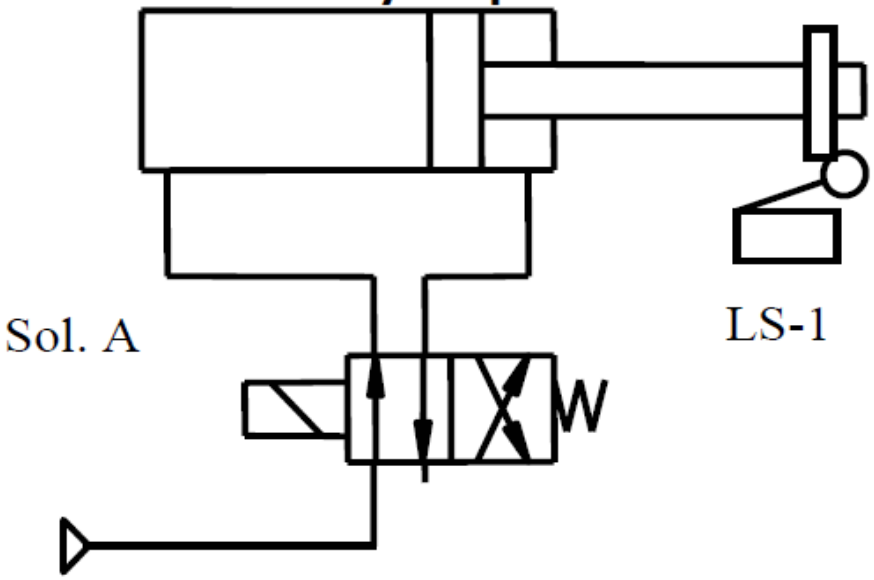


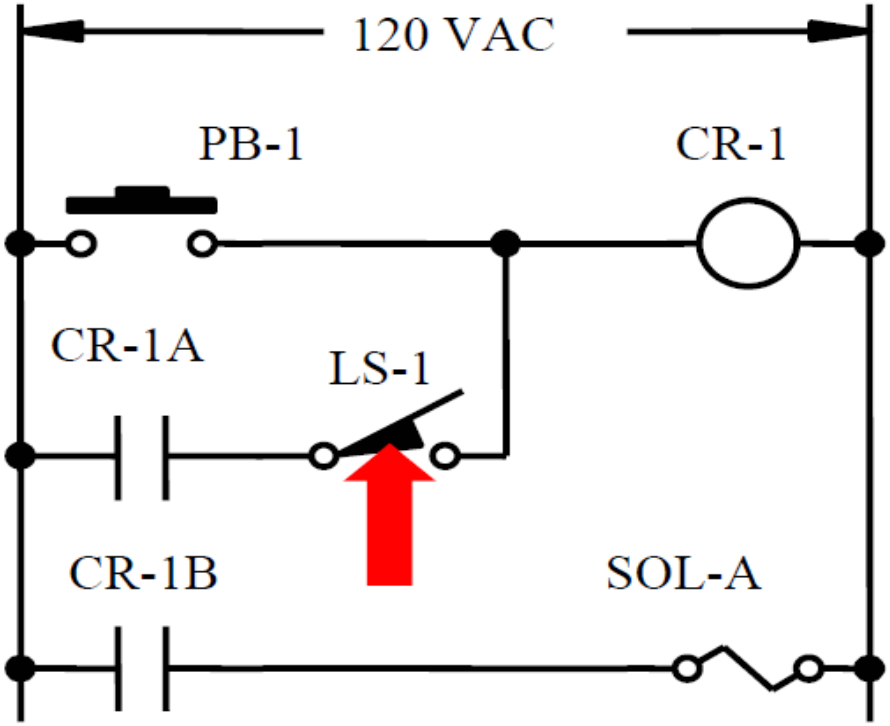
- Cylinder activates the normally closed limit switch LS-1, which "kills" the hold circuit for control relay CR-1



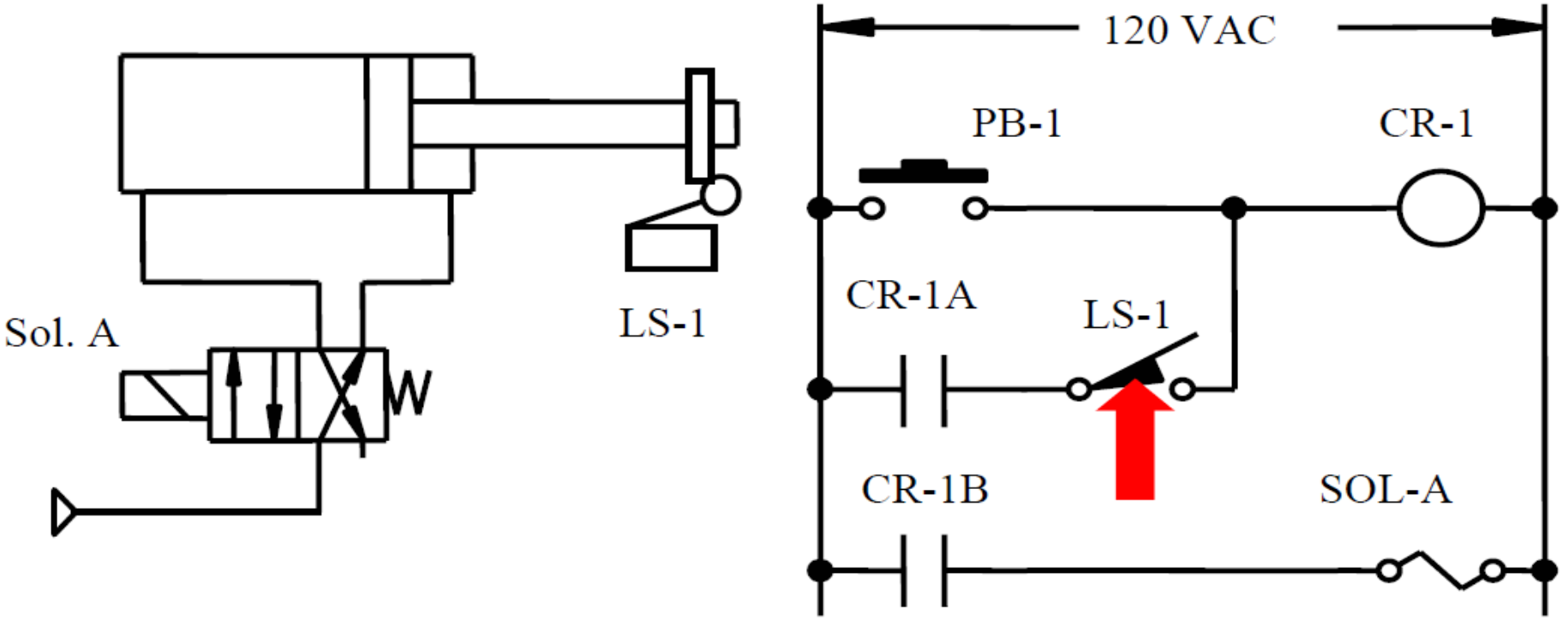


- With control relay CR-1 de-activated, the contacts CR-1A and CR-1B return to their normally open state

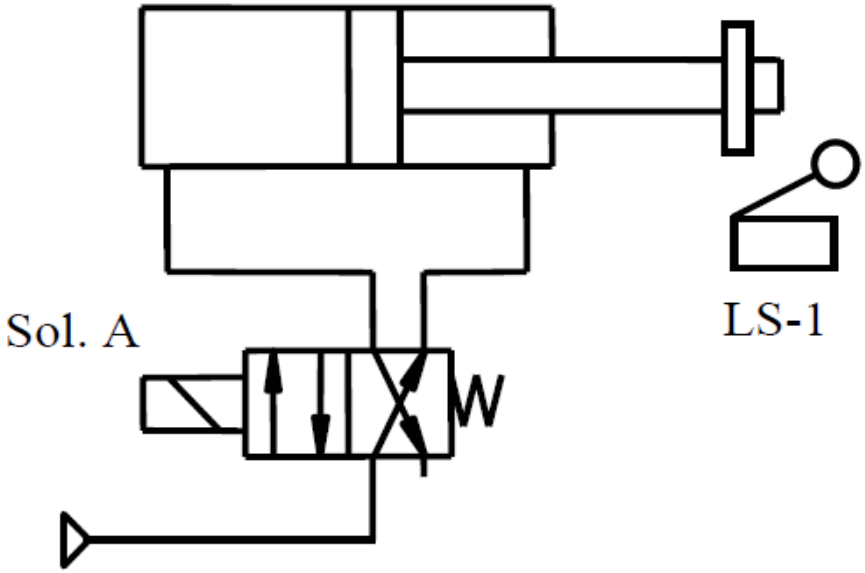


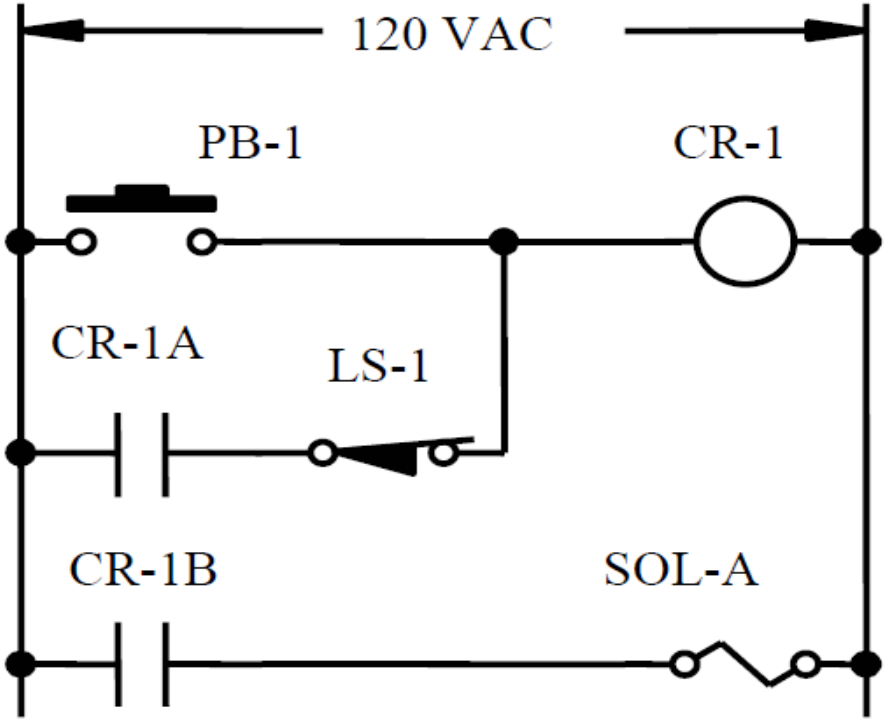


- CR-1B is now open, SOL-A is de-activated, spring returns valve to default state

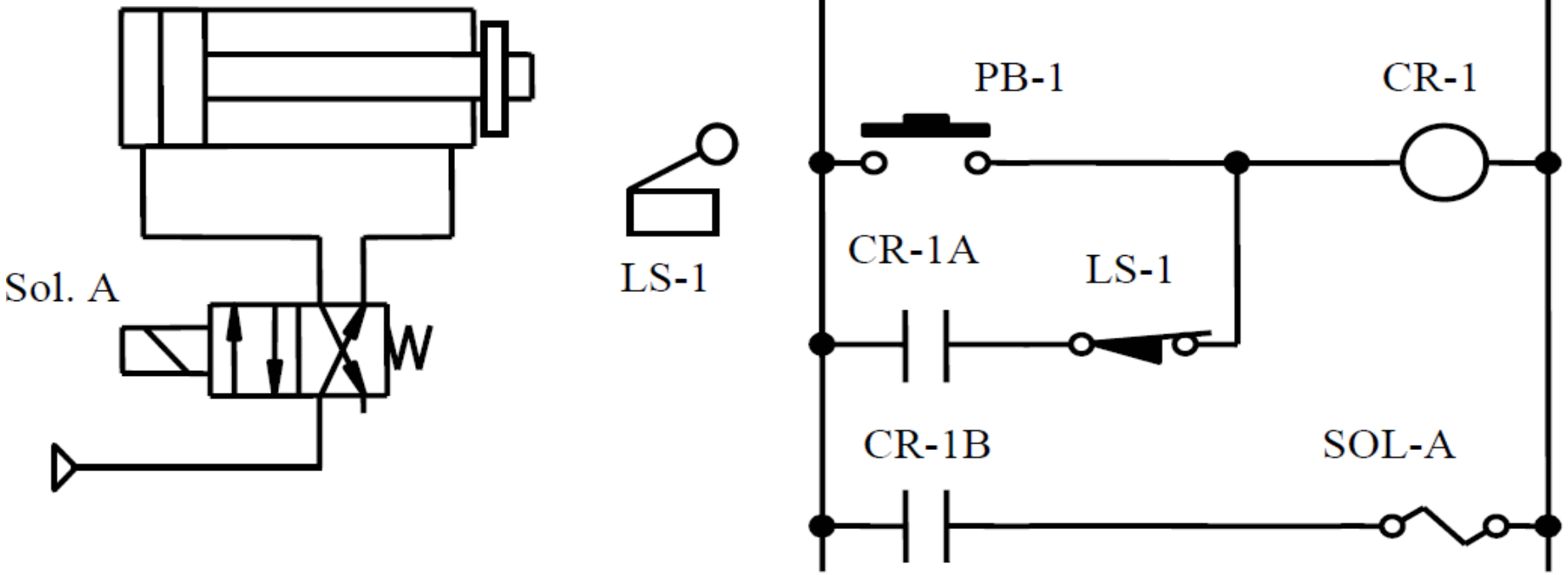


- Cylinder begins to retract, and "rolls off" of LS-1, which returns to its N.C. state





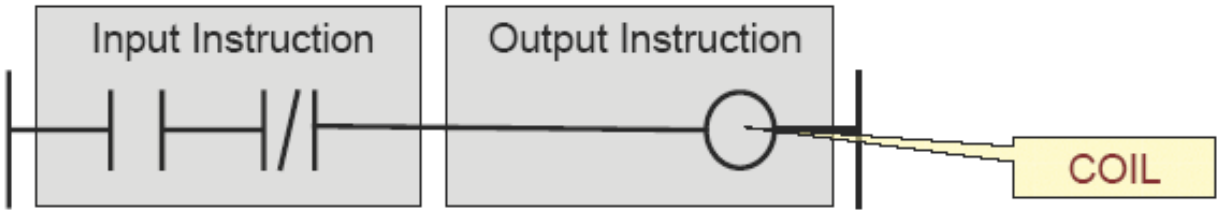
- Cylinder fully retracts and system has returned to the start-up configuration



Parte 3: Lenguajes de Programación **Lenguaje LD**

¿Qué es un *Rung*?

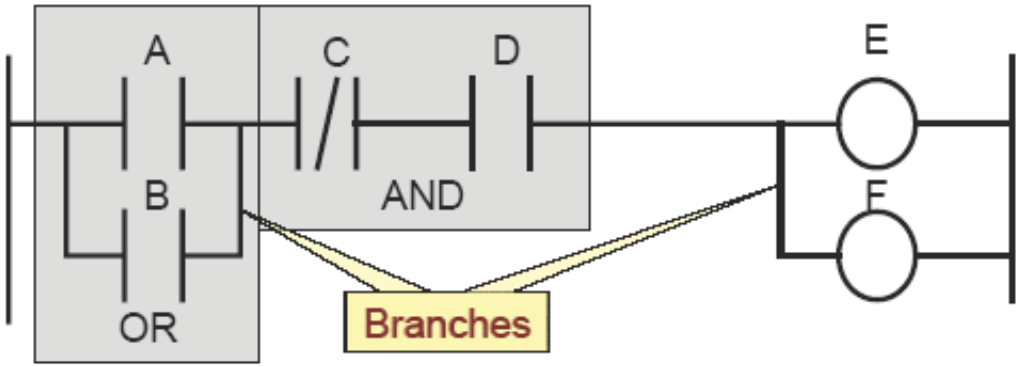
- Es una línea de programa
- Contiene las instrucciones de entrada y salida
  - Entrada: permiten una comparación o test de las condiciones y se obtiene el resultado de la evaluación.
    - Habitualmente aparecen el la parte izquierda del *rung*
  - Salida (*Coil*): examinan el resultado de la evaluación y si es *true* ejecutan alguna operación o función
    - En algunos casos pueden ser el estado del *rung*
    - Habitualmente aparecen el la parte derecha del *rung*



Parte 3: Lenguajes de Programación **Lenguaje LD**

Operaciones en Serie y Paralelo

- Las instrucciones de entrada pueden ejecutarse mediante relaciones lógicas AND y OR en un sencillo formato
  - Si las instrucciones están en serie se evalua una relación AND
  - Si las instrucciones están en paralelo se evalua una relación OR
- Salidas en paralelo permite activar varias òperacoines o funciones con el mismo resutado de la evaluación

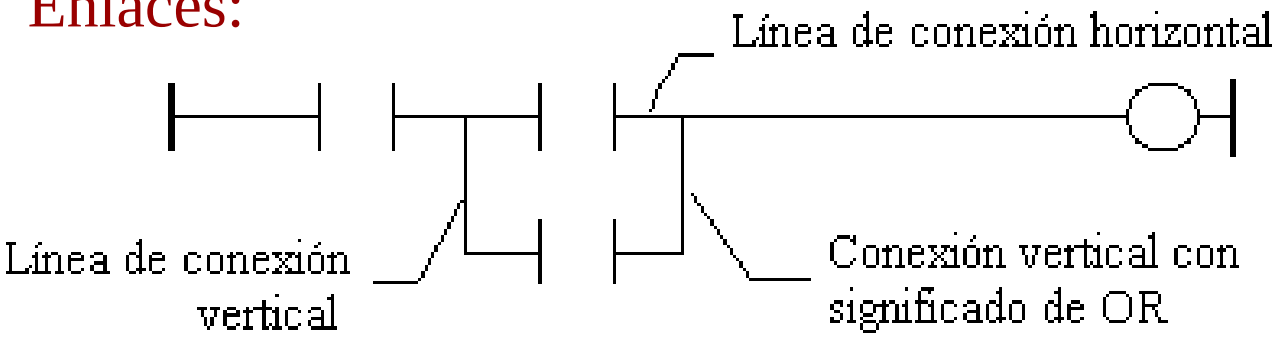


IF ((A OR B) AND (NOT C) AND D) THEN E=1; F=1 END_IF

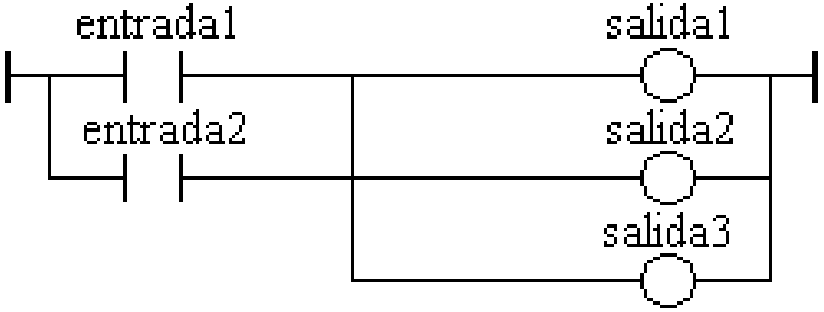
Parte 3: Lenguajes de Programación

Lenguaje LD

Enlaces:



(*Ejemplo de conexiones multiples*)



(*Equivalencia en ST*)

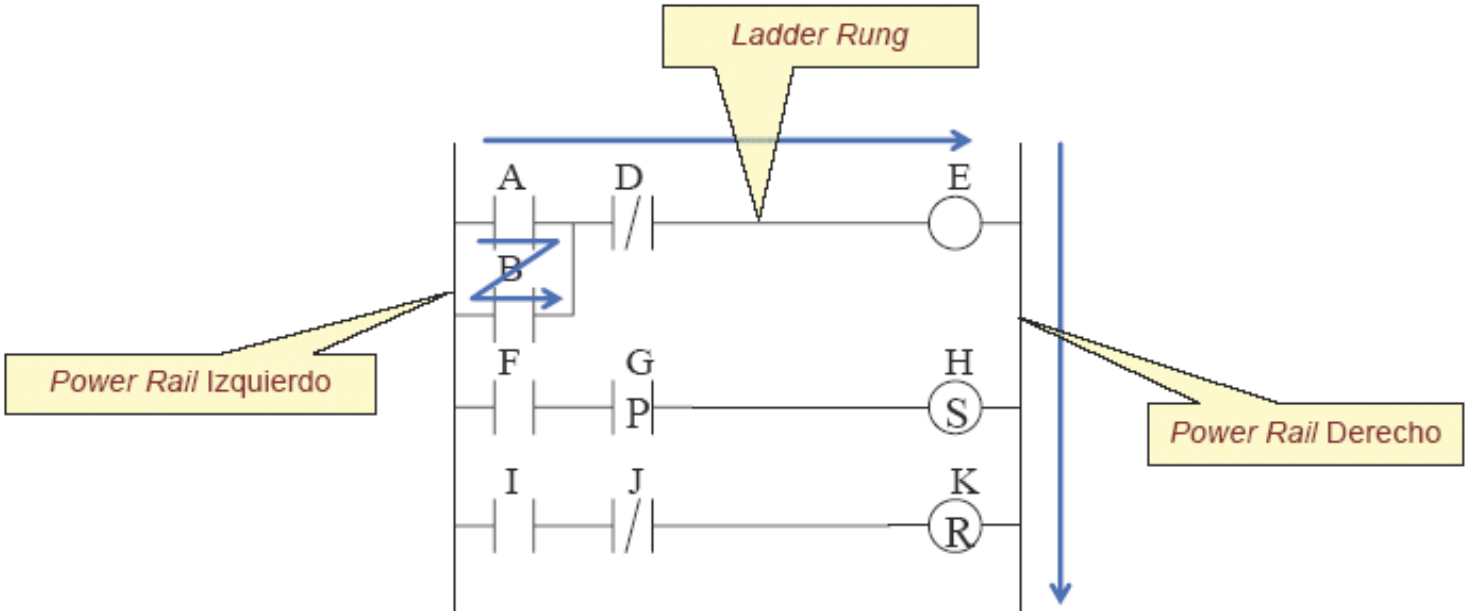
```
salida1 := entrada1 OR entrada2;
salida2 := entrada1 OR entrada2;
salida3 := entrada1 OR entrada2;
```

Parte 3: Lenguajes de Programación

Lenguaje LD

Ejecución Lógica en *Ladder*

- Los *Rungs* se ejecutan de izquierda a derecha y de arriba a abajo
- Los *Rungs* con bifurcaciones se ejecutan de arriba izquierda a abajo derecha



Parte 3: Lenguajes de Programación

Lenguaje LD

Contactos

- Normalmente Abierto **--| |--**
  - Activa el *rung* hacia la derecha de la instrucción cuando el contacto se activa
- Normalmente Cerrado **--|/|--**
  - Activa el *rung* hacia la derecha de la instrucción cuando el contacto se desactiva
- Transición positiva **--|P|--**
  - Activa el *rung* hacia la derecha de la instrucción cuando el contacto está desactivo en el *scan* anterior y activo en el *scan* actual
  - P.e.: Allen Bradley PLC5 utiliza --[ONS]--
- Transición Negativa **--|N|--**
  - Activa el *rung* hacia la derecha de la instrucción cuando el contacto está activo en el *scan* anterior y desactivo en el *scan* actual

Parte 3: Lenguajes de Programación **Lenguaje LD**

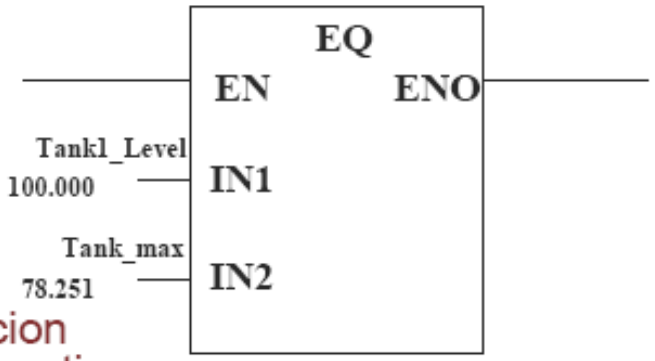
Acciones (*Coils*)

- Acción **--( )--**
  - Activa un bit cuando el *rung* es *true* y lo desactiva cuando es *false*
- Acción negada **--(/)--**
  - Activa un bit cuando el *rung* es *false* y lo desactiva cuando es *true*
- Enclavamiento (Latch) **--(S)--**
  - Activa un bit cuando el *rung* es *true* y no hace nada cuando es *false*
- Desenclavamiento (Unlatch) **--(R)--**
  - Desactiva un bit cuando el *rung* es *true* y no hace nada cuando es *false*
- Acción activa por flanco de subida **--(P)--**
  - Activa un bit cuando la instrucción de entrada transiciona de *false* a *true*
- Acción activa por flanco de bajada **--(N)--**
  - Activa un bit cuando la instrucción de entrada transiciona de *true a false*

Parte 3: Lenguajes de Programación **Lenguaje LD**

Instrucciones IEC de Comparación

- Si el *rung* de entrada está activo (EN), la instrucción ejecuta la operación y activa el *rung* de salida (ENO) basado en la comparación
  - Ejemplo
    - Cuando EN es *true*, EQ (=) la funcion compara In1 y In2 y si son iguales activa ENO

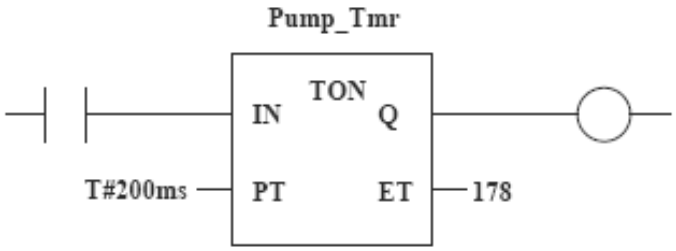


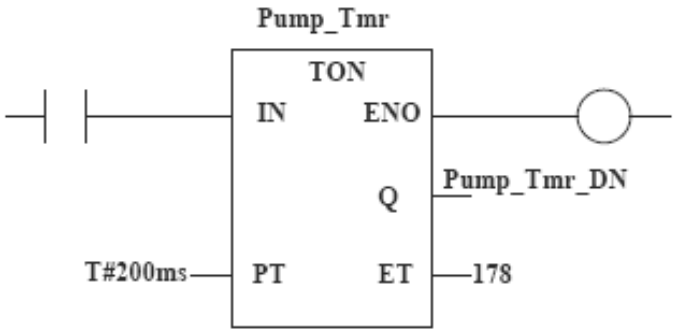
- Conjunto de instrucciones de comparación
  - EQ(=), GT (>), GE (>=), LT (<), LE (<=), NE (<>)

Parte 3: Lenguajes de Programación **Lenguaje LD**

Instrucciones IEC de Temporización

- Tres insctrucciones básicas
  - TP - *Pulse timer*
  - TON - *Timer On Delay*
  - TOF - *Timer Off Delay*
- Valores temporales enteros
  - Base de tiempos de 1msec
- Dos posibles formas de uso
  - 1ª necesita programación extra en otro *rung* para interaccionar sobre el estado del *timer*
  - 2ª activa un bit que puede ser utlilizado en otras funciones lógicas



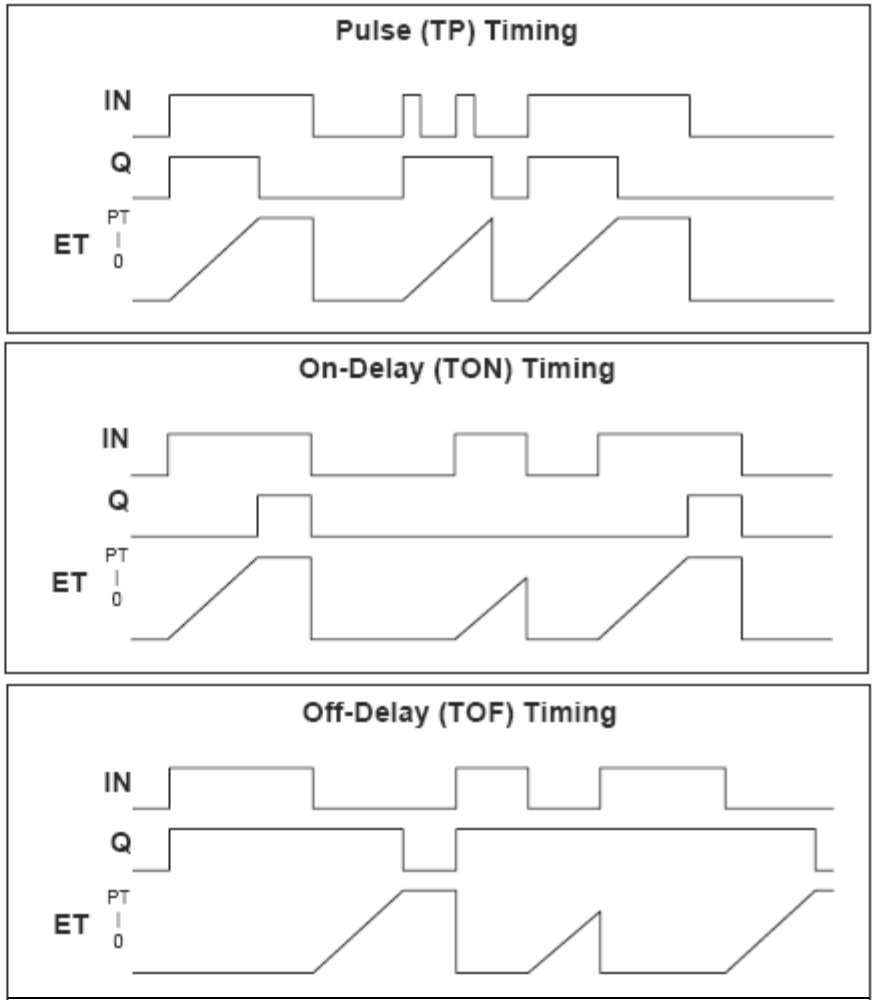


Parte 3: Lenguajes de Programación

Lenguaje LD

Temporizador

- IN = instrucción de entrada del *Rung*
- Q = Resultado de la comparación
  - Varía con el tipo de *timer*
- PT = *Preset Time*
- ET = *Elapse Time*

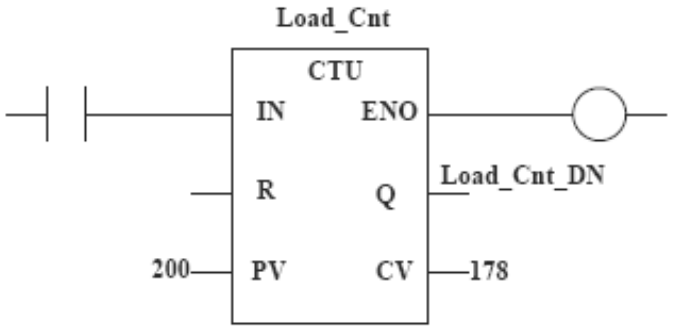


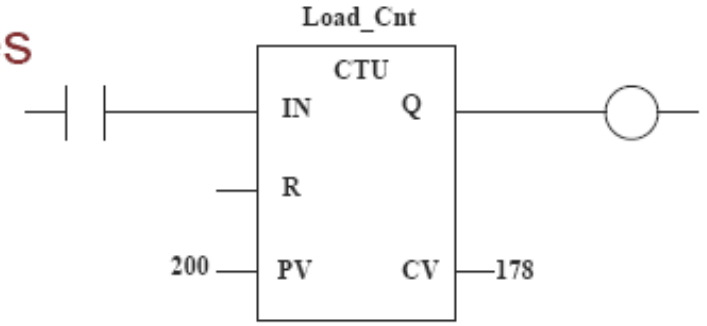
Parte 3: Lenguajes de Programación

Lenguaje LD

Instrucciones IEC de Contadores

- Tres instrucciones básicas
  - CTU - *Count Up Counter*
  - CTD - *Count Down Counter*
  - CTUD - *Count Up/Down Counter*
- Todos cuentan transiciones
- Dos formas de uso, igual que los temporizadores



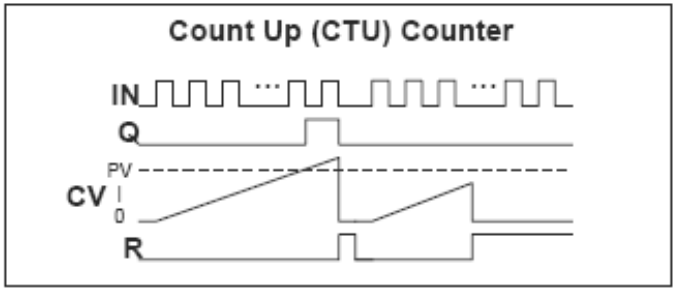


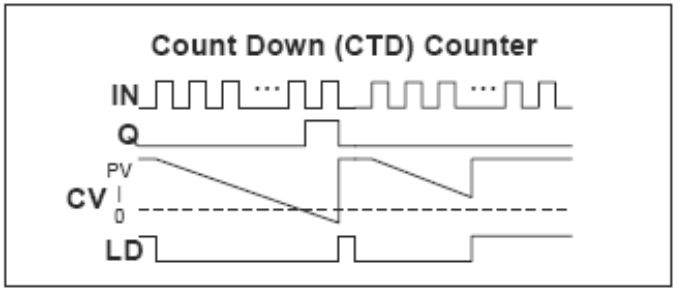
Parte 3: Lenguajes de Programación

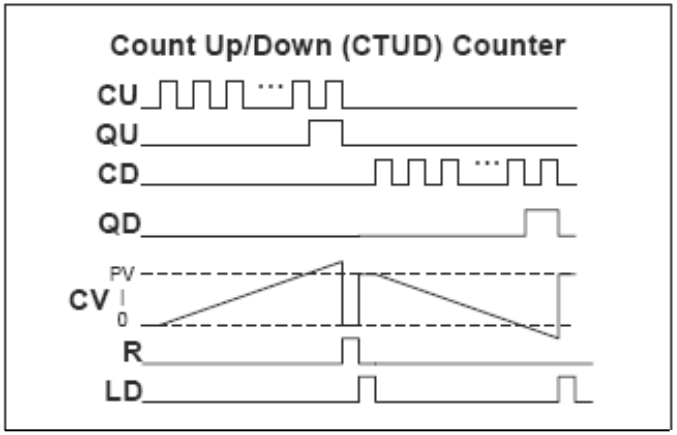
Lenguaje LD

Contadores

- CU/CD = *Count up/Down*
- Q/QU/QD = Comparación de salida
- R = Puesta a cero
- LD = Carga CV con PV
- PV = *Preset Value*
- CV = *Count Value*



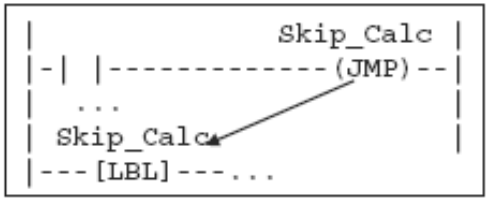


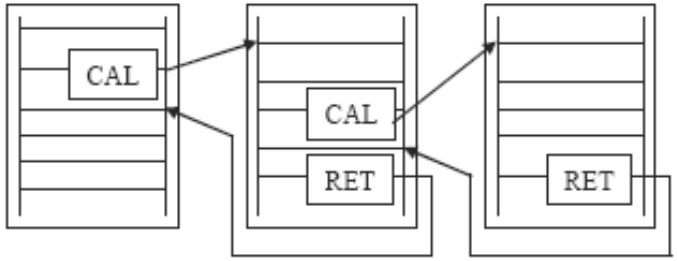


Parte 3: Lenguajes de Programación **Lenguaje LD**

Ruptura de la secuencia de ejecución

- Instrucciones de salto a etiquetas
  - Salta a un bloque de código del programa
  - LBL – nombre de la etiqueta para la operación de salto
  - JMP – ejecución de un salto cuando se activa la instrucción de entrada
- Instrucciones de salto a subrutinas
  - Salta a un bloque de código encapsulado como una subrutina
  - CALL – pasa el control a otra función
  - RET – retorno al punto siguiente desde donde fue llamada la subrutina





Parte 3: Lenguajes de Programación

Lenguaje FBD

Diagrama de Bloques funcionales (FBD)

Características:

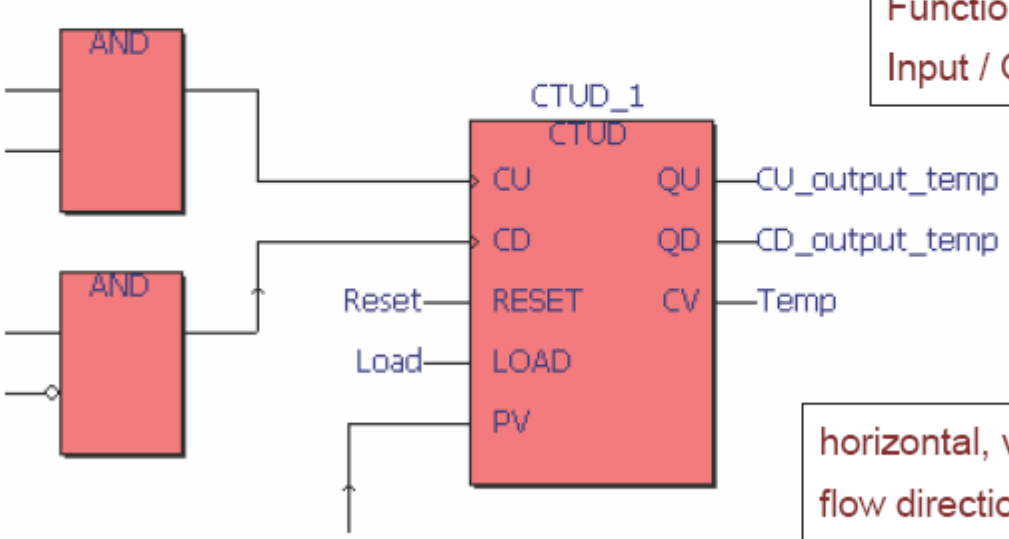
- La representación es coherente con la Norma CEI 617-12.
- Las salidas de los bloques funcionales no se conectarán entre sí (se precisa bloque “OR”).
- La evaluación de una red estará terminada antes de la siguiente.

Parte 3: Lenguajes de Programación

Lenguaje FBD

Elementos: Funciones, Bloques y Variables

Functions
Functions Blocks
Input / Output Variables

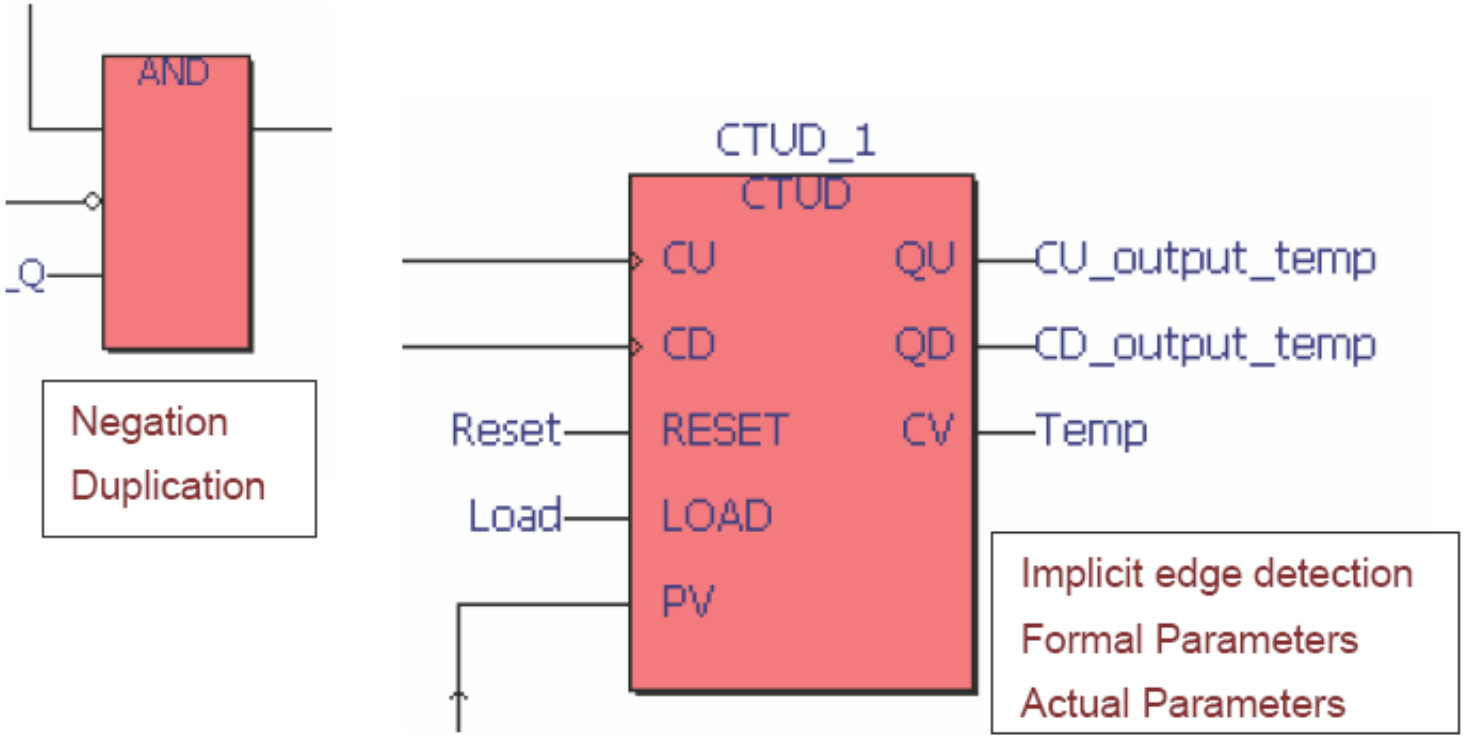


horizontal, vertical
flow direction
crossing with / without contact

Parte 3: Lenguajes de Programación

Lenguaje FBD

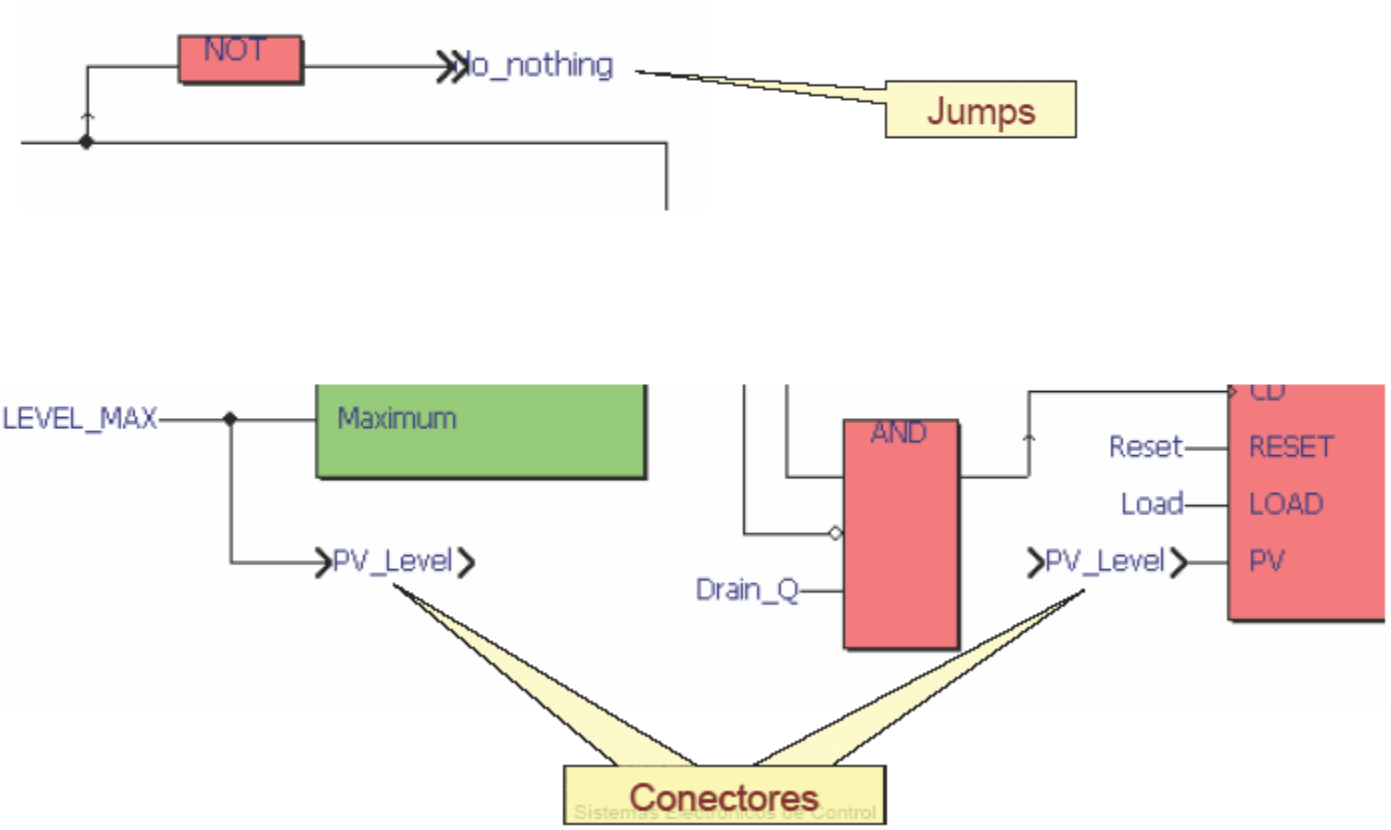
Elementos - Parametrización



Parte 3: Lenguajes de Programación

Lenguaje FBD

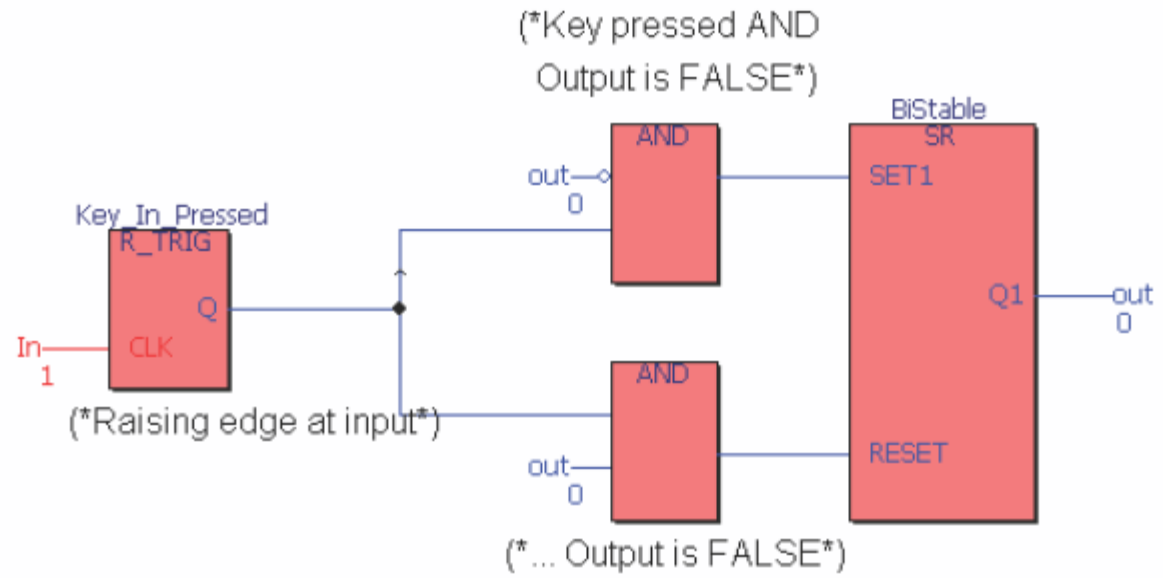
Elementos adicionales



Parte 3: Lenguajes de Programación

Lenguaje FBD

Reglas de Ejecución

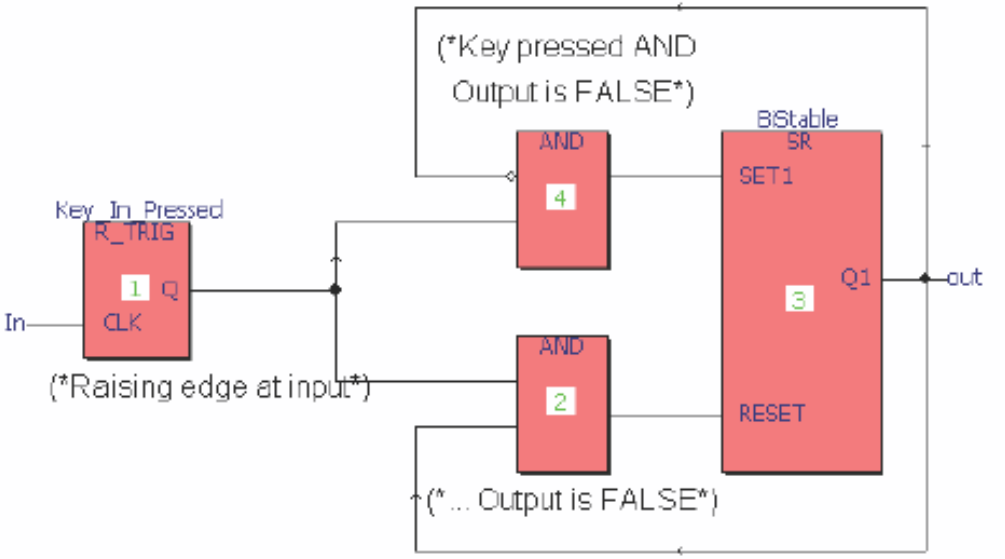


- 1. El bloque se ejecuta cuando todas sus entradas han sido evaluadas
- 2. El bloque se evalúa por completo cuando se has calculado todas sus salidas
- 3. La evaluzación de un conjunto de bloques termina cuando se calculan todas y cada una de las salidas

Parte 3: Lenguajes de Programación

Lenguaje FBD

Realimentación

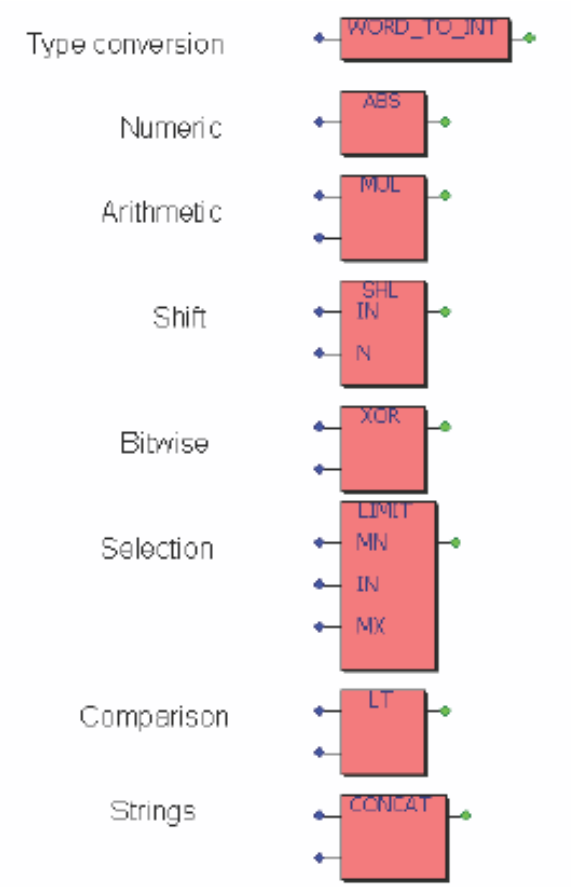


- No se puede valorar el orden de la ejecución
- Existen formas de resolverlo como la asignación de un orden de ejecución

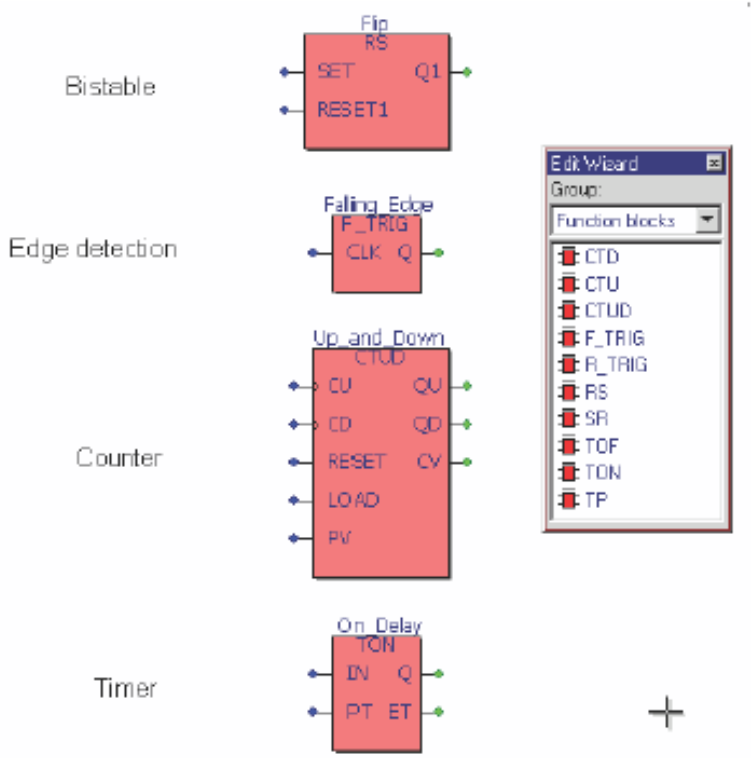
Parte 3: Lenguajes de Programación

Lenguaje FBD

Funciones Estándar



Bloques Estándar

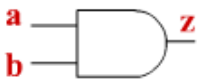
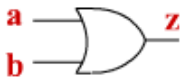
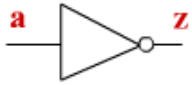
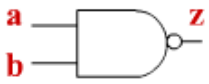
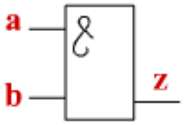
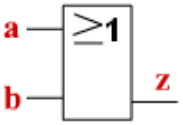
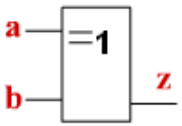
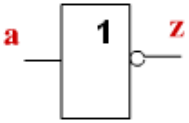
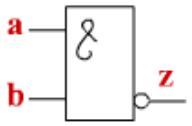
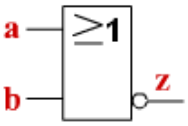
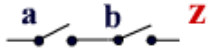
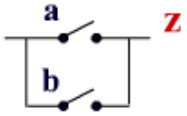
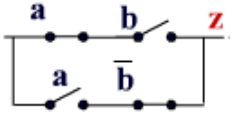
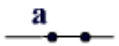
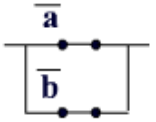


- Norma IEC 61131: “Si se conece el estándar, se conoce todo"

Parte 3: Lenguajes de Programación

Lenguaje FBD

FUNCIONES LÓGICAS BÁSICAS

NOMRE	AND - Y	OR - O	XOR O-exclusiva	NOT Inversor	NAND	NOR
SÍMBOLO						
SÍMBOLO						
TABLA DE VERDAD	a b z 0 0 0 0 1 0 1 0 0 1 1 1	a b z 0 0 0 0 1 1 1 0 1 1 1 1	a b z 0 0 0 0 1 1 1 0 1 1 1 0	a z 0 1 1 0	a b z 0 0 1 0 1 1 1 0 1 1 1 0	a b z 0 0 1 0 1 0 1 0 0 1 1 0
EQUIVALENTE EN CONTACTOS						
AXIOMA	$z = a \cdot b$	$z = a + b$	$z = \bar{a} \cdot b + a \cdot \bar{b}$	$z = \bar{a}$	$z = \overline{a \cdot b}$	$z = \overline{a + b}$

Parte 3: Lenguajes de Programación

Lenguaje SFC

¿De dónde proviene su nombre?

Es el acronismo para:

GRAfico
Funcional de
Control
de Etapas y
Transiciones

Parte 3: Lenguajes de Programación

Lenguaje SFC

1977 — La AFCET (Association Française pour la Cybernétique Economique et Technique) define

GRAFCET = GRAfico Funcional de Control Etapa-Transción

1982 — NF C03-1904. Estándar Francés

1988 — IEC-848. Preparation of function charts for control systems.

1993 — IEC-61131-3. Sequential Function Chart (SFC).

GRAFCET = Lenguaje Gráfico
de Descripción de Modelos de Automatismos Secuenciales
Independiente de la Tecnología de las Partes de Mando y Operativa

GRAFCET NO ES un Lenguaje de Programación

Parte 3: Lenguajes de Programación

Lenguaje SFC

El lenguaje SFC es usado para describir operaciones de procesos secuenciales. Utiliza una simple representación gráfica de diferentes pasos de un proceso, y de las condiciones que habilitan el cambio (transición) de los pasos activos. Sus principales elementos son:


Paso inicial


Paso


Transición


Salto a un paso


Macro paso


Inicio del macro paso

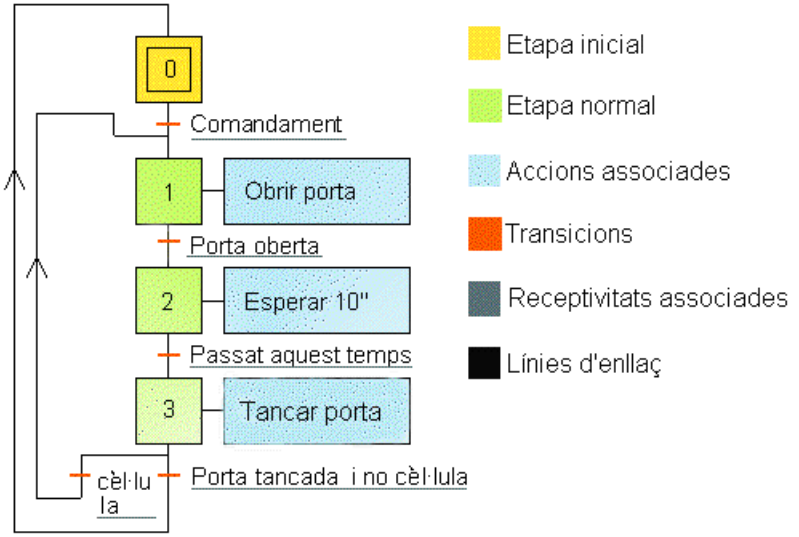

Fin del macro paso

Parte 3: Lenguajes de Programación

Lenguaje SFC

Elementos del Grafcet

- Etapas iniciales
- Etapas normales
- Acciones asociadas
  Acciones asociadas condicionadas
- Transiciones
- Líneas de enlace



Parte 3: Lenguajes de Programación **Lenguaje SFC**

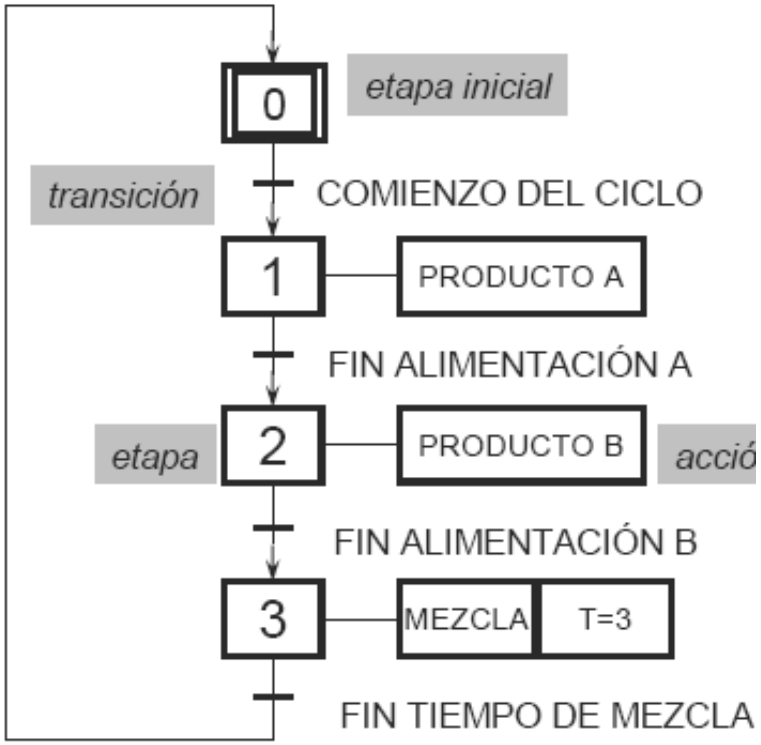
IEC-848

- En 1988, el GRAFCET es reconocido por una norma internacional, la IEC-848 (*Preparation of function charts for control systems*, Preparación de diagramas funcionales para sistemas de control) con los nombres *Function Chart*, *Diagramme fonctionnel* o Diagrama funcional. La norma IEC no reconoce el nombre GRAFCET porque las traducciones pueden dar lugar a ambigüedades.

Parte 3: Lenguajes de Programación

Lenguaje SFC

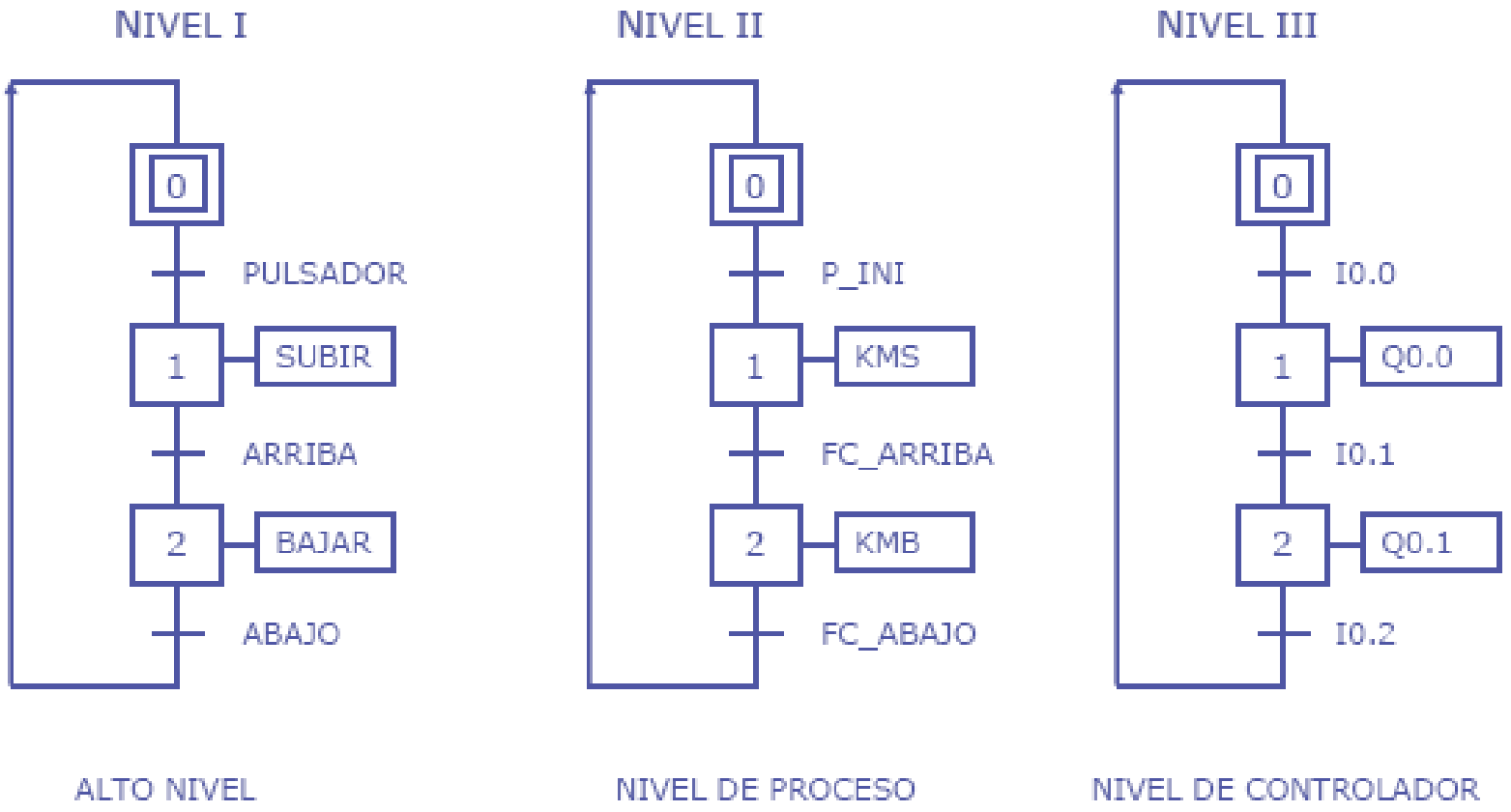
- Las etapas o estados implican acciones asociadas
- Las transiciones gobiernan los cambios de estado
- Las flechas indican la dirección del cambio
- Pueden darse esquemas menos lineales



Parte 3: Lenguajes de Programación

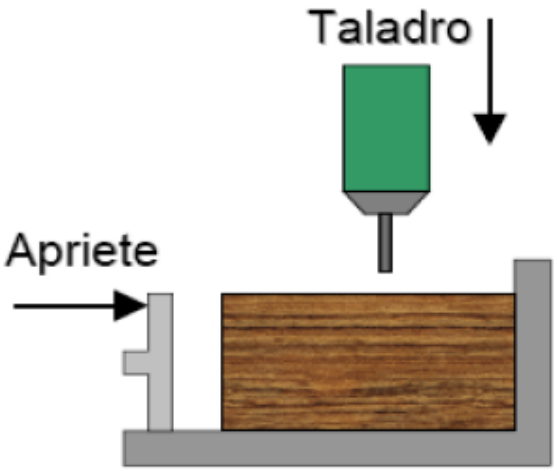
Lenguaje SFC

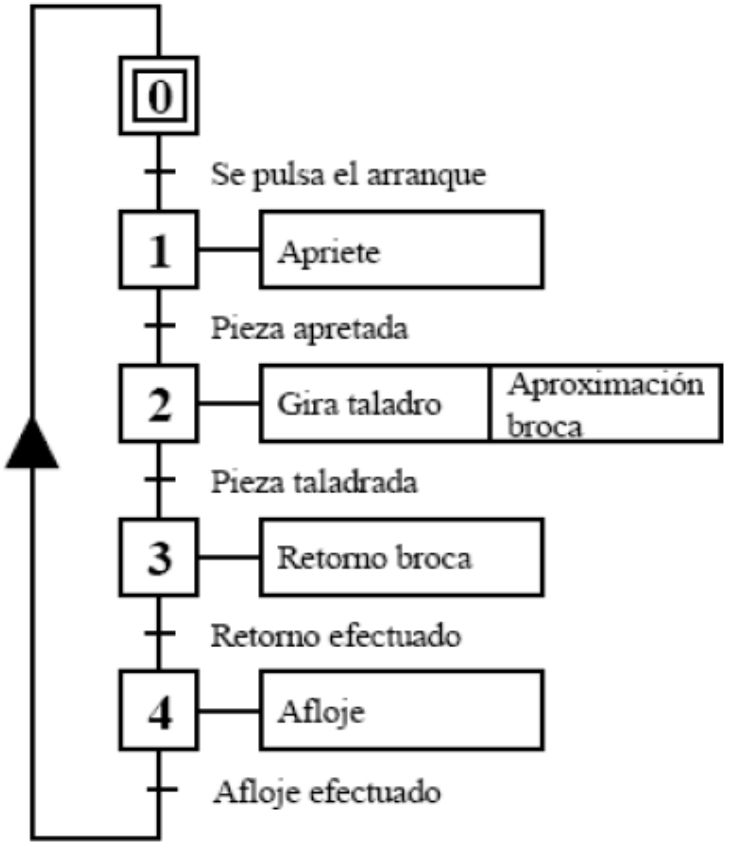
NIVELES DE REPRESENTACIÓN DE GRAFCET



Nivel 1. Descripción Funcional

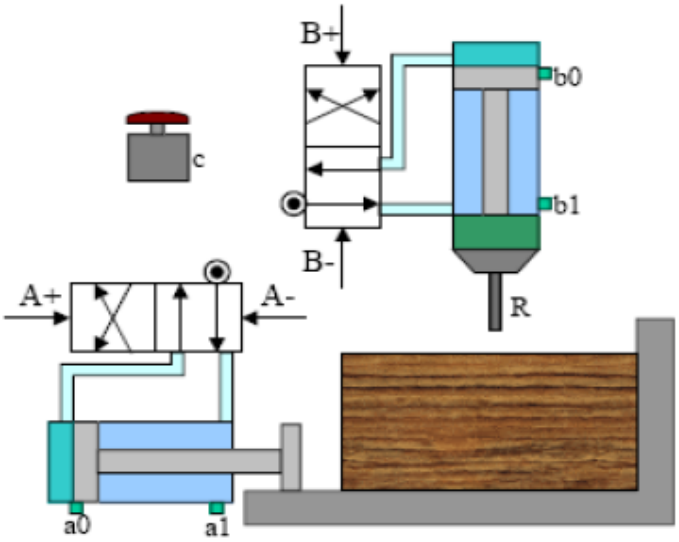
En este nivel no se tiene en cuenta la tecnología empleada para desarrollar el sistema. Simplemente se describe qué es lo que tiene que hacer.

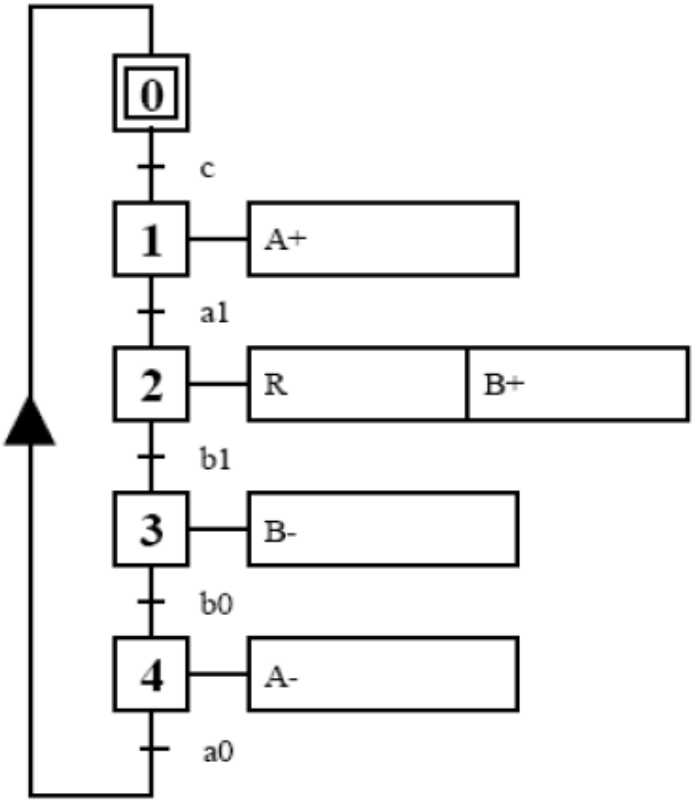




Nivel 2. Descripción Tecnológica

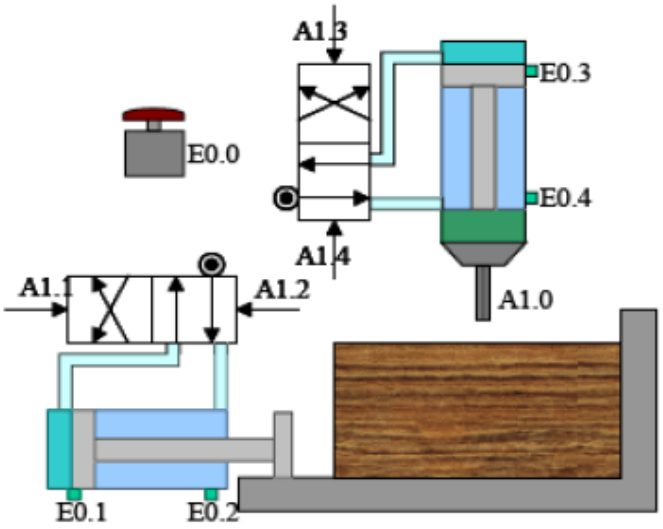
Una vez decidida la tecnología a emplear, y definidos los sensores y actuadores, se puede realizar un GRAFCET con un menor grado de abstracción, en el que se muestran claramente las especificaciones técnicas y operativas.
En este caso ya sabemos que se va utilizar un sistema de cilindros hidráulicos o neumáticos de doble efecto accionados por válvulas, que el taladro tiene accionamiento eléctrico, unos finales de carrera y un pulsador

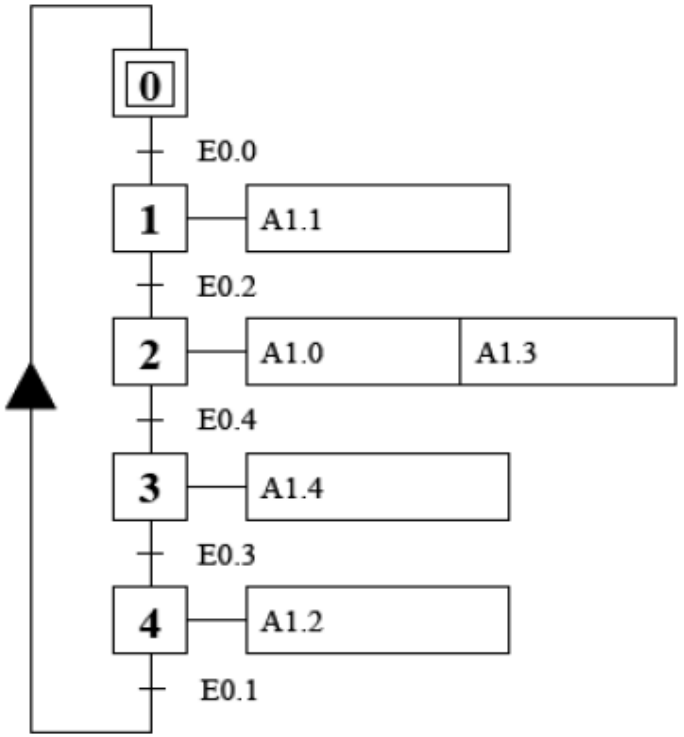




Nivel 3. Descripción Operativa

Ahora ya no sólo hacemos referencia a los accionamientos y sensores que utilizaremos, sino a la propia tecnología del control y de la automatización.

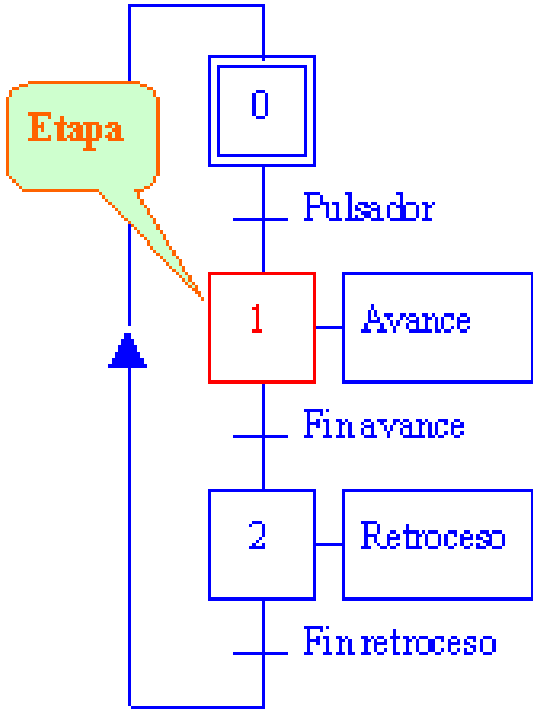




Parte 3: Lenguajes de Programación **Lenguaje SFC**

Definiciones básicas

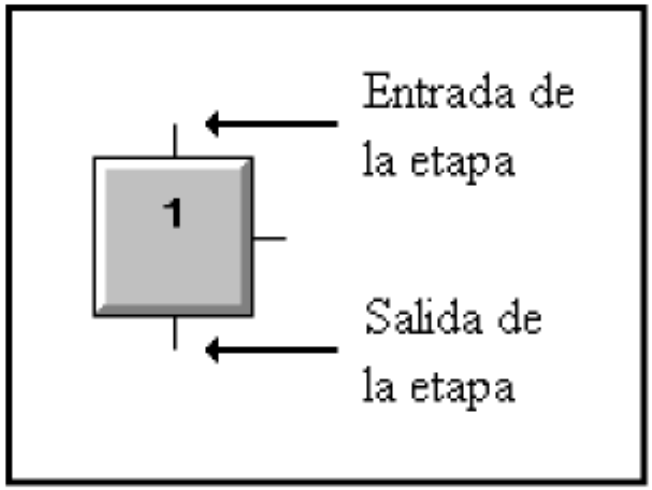
– Etapa: es cada uno de los estados o partes en los que se divide el GRAFCET. Se representan con cuadros. A la etapa se le asocian acciones. La etapa cero (0) se activa en la puesta en marcha.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Etapas

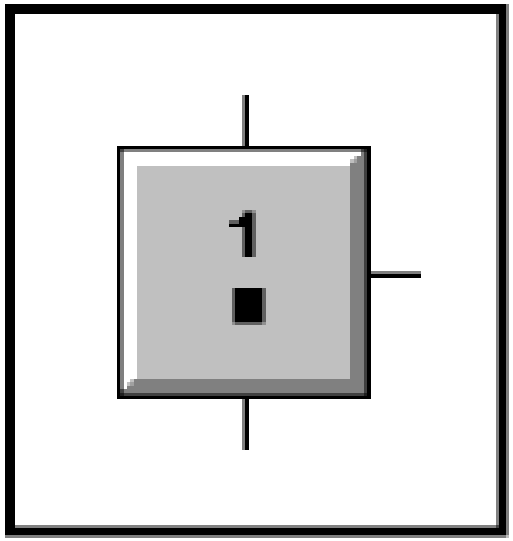
- Una etapa se caracteriza por un comportamiento invariable en una parte o en la totalidad de la parte de mando.
- Una **etapa puede estar activa o inactiva.**
- La etapa inicial se dibuja con doble recuadro, una línea de entrada y una línea de salida, y se identifica con un número.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Etapas

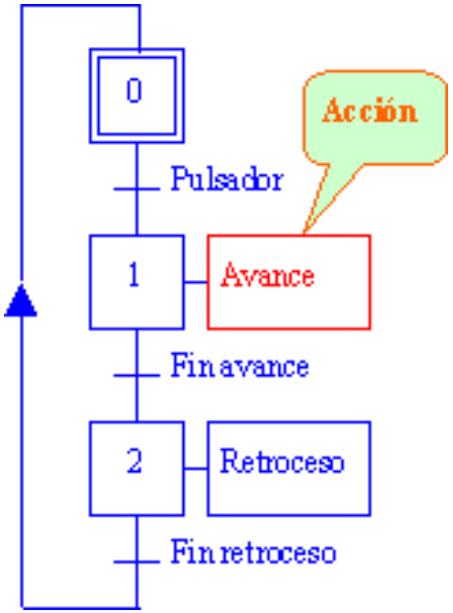
- Cuando es necesario determinar la situación del Grafcet en un momento determinado, es útil identificar todas las etapas activas en ese momento, mediante un **punto** en la parte inferior de los símbolos de las etapas activas.



Parte 3: Lenguajes de Programación

Lenguaje SFC

- Acción: la función o funciones que debe realizar el sistema descrito en una etapa.
  - Cada etapa tiene sus **acciones** asociadas de forma que cuando una etapa está activa se realizan las correspondientes acciones; pero estas acciones no podrán ejecutarse nunca si la etapa no está activa.
  - Se dibujan con rectángulo.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

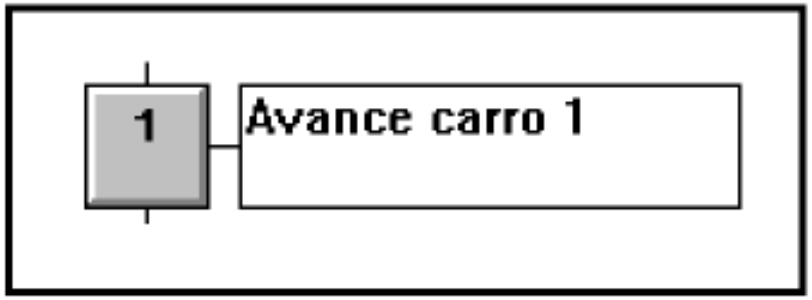
Acciones

- Las acciones están descritas, literal o simbólicamente, en el interior de uno o varios rectángulos unidos al símbolo de la etapa a la que van asociados.
- Las acciones asociadas a las etapas y las receptividades asociadas a las transiciones se pueden describir a dos niveles:
  - Nivel 1
  - Nivel 2

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Niveles de las Acciones

- Nivel 1: no tienen en cuenta los elementos tecnológicos que implementan. **Se describen las funciones del sistema.**



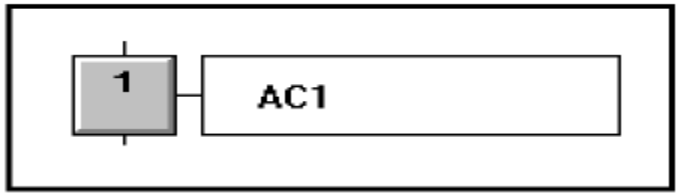
Acción asociada a la etapa 1 (nivel 1):

Avance del carro 1

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Niveles de las Acciones

- Nivel 2: **se especifican los aspectos tecnológicos de los dispositivos a utilizar,** y de las funciones del equipo de control. Por ejemplo: movimiento de actuadores, activación de pre-actuadores.



Acción asociada a la etapa 1 (nivel 2):

AC1

AC1: Avance del Carro 1

Parte 3: Lenguajes de Programación

Lenguaje SFC

- Transición:
  - Entre dos etapas hay una **transición**
  - Una transición indica la posibilidad de evolución entre etapas. Es la condición booleana que se prueba para pasar de una etapa activa a una etapa inactiva.
  - La evolución entre dos etapas ocurre cuando se produce el **franqueo de la transición.**

- Franqueo de una transición: provoca el paso en la parte de mando de una situación a otra situación.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Transición

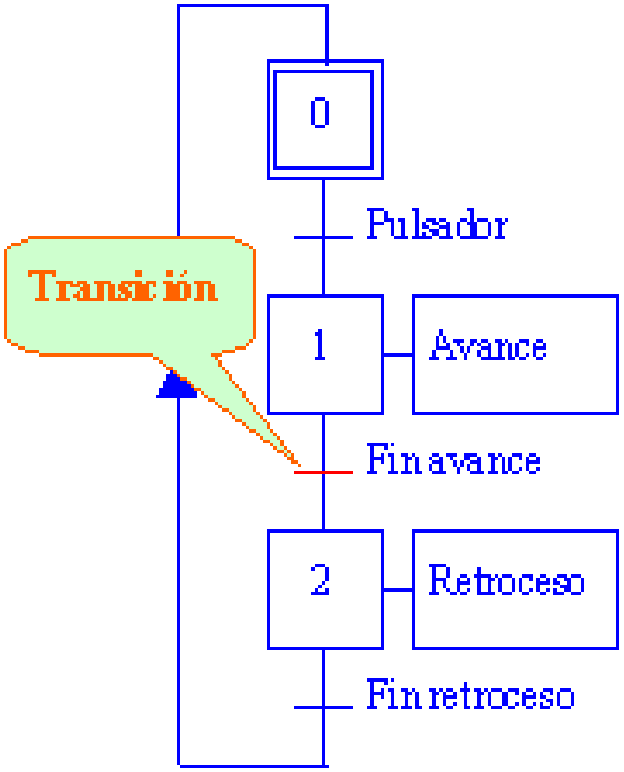
- Una transición entre dos etapas se representa mediante una línea perpendicular a las uniones orientadas, también puede llevar una línea paralela a las uniones orientadas.

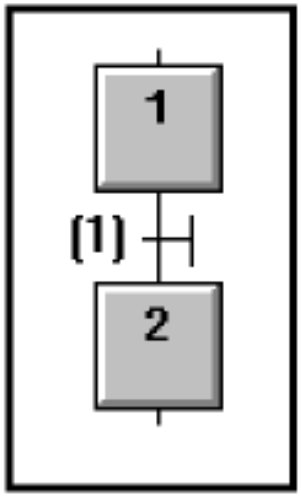
- Para facilitar la comprensión del Grafcet cada transición puede ir numerada a la izquierda de la línea perpendicular.

Parte 3: Lenguajes de Programación

Lenguaje SFC

Transición





Transición que une la etapa 1 con la etapa 2

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Transición

- Una **transición puede estar validada o no validada.** Se dice que está validada cuando todas las etapas inmediatamente unidas a esta transición están activas.

- A cada transición le corresponde una **receptividad**, es decir una condición que se ha de cumplir para poder pasar la transición. Es la condición booleana que se prueba para pasar de una etapa activa a una etapa inactiva

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Transición

- Una transición es **válida** cuando la etapa inmediatamente anterior a ella está activa. Cuando una transición es válida y su receptividad asociada se cumple se dice que la transición es **franqueable**.

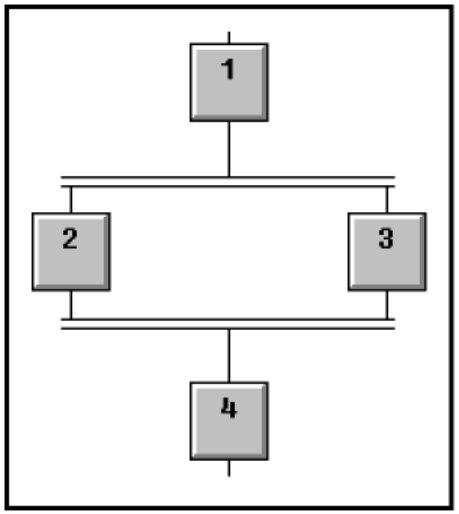
- La condición de franqueable es sólo temporal: toda transición franqueable será de inmediato franqueada.

- Transición franqueada: significa que se ha(n) activado la(s) etapa(s) posterior(es) a la transición y se ha(n) desactivado la(s) etapa(s) inmediatamente anterior(res) a ella.

Parte 3: Lenguajes de Programación

Lenguaje SFC

- **Uniones Orientadas:**
  - Son las líneas que unen las etapas a las transiciones y las transiciones a las etapas.
  - Cuando varias transiciones van unidas a una misma etapa, las uniones orientadas correspondientes se reagrupan antes o después de la etapa:



Parte 3: Lenguajes de Programación

Lenguaje SFC

Estructuras básicas

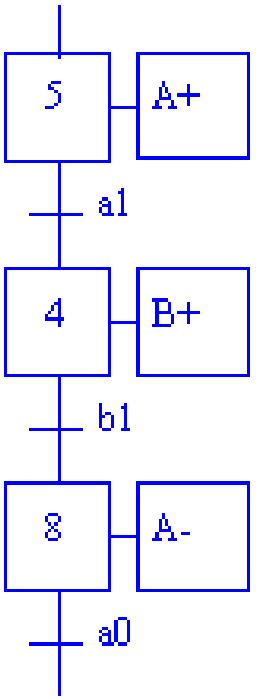
- Secuencia
- Selección de secuencia
- Salto de etapas
- Repetición de secuencia
- Paralelismo estructural
- Paralelismo interpretado

Parte 3: Lenguajes de Programación

Lenguaje SFC

Secuencia

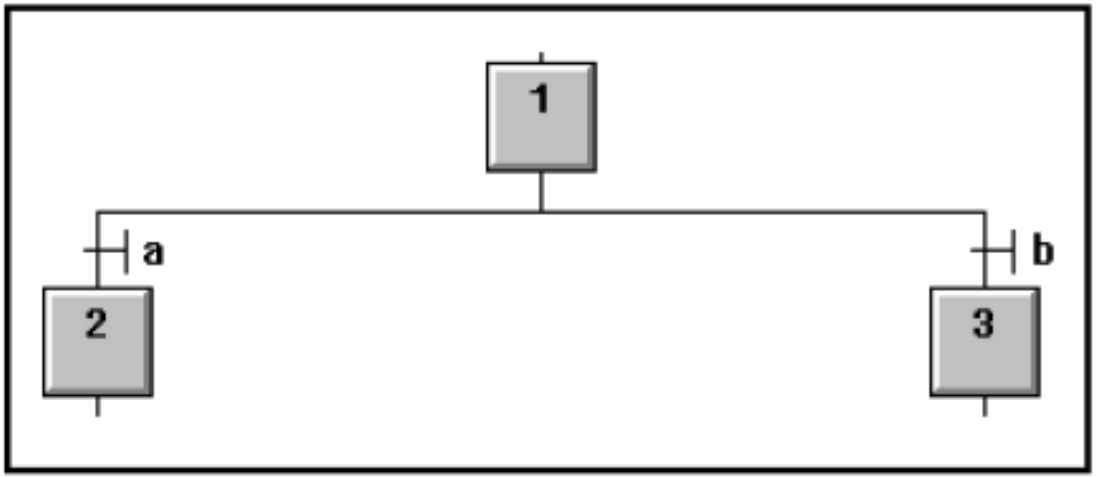
- Una secuencia es una sucesión alternada de etapas y transiciones en la que las etapas se van activando una detrás de otra. Una secuencia está activa cuando, como mínimo, una de sus etapas está activa. Una secuencia está inactiva cuando todas sus etapas están inactivas.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Divergencias y convergencias.

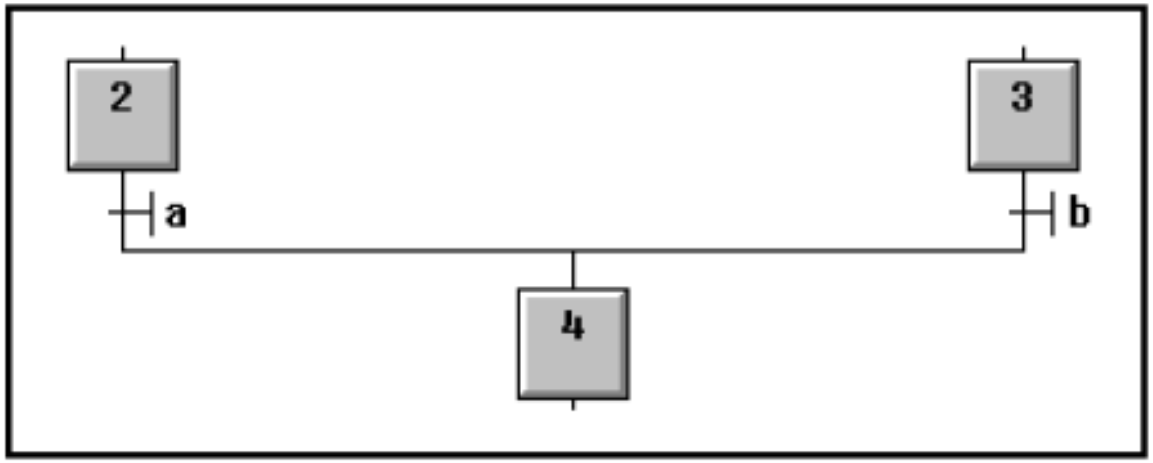
- **Divergencia en O.**
  - Cuando la etapa 1 está activa, según se cumpla la receptividad asociada a la transición **a o la receptividad asociada a la transición b, pasará** a ser activa la etapa 2 o bien la etapa 3 respectivamente. Se trata de secuencias excluyentes.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Divergencias y convergencias.

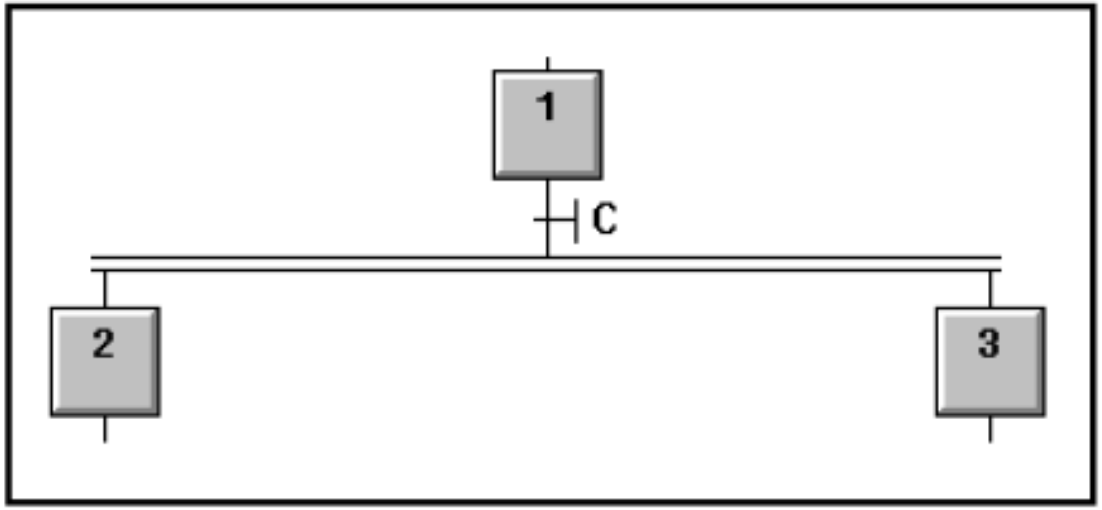
- **Convergencia en O.**
  - Si la etapa activa es la 2 debe cumplirse la receptividad asociada a la transición **a para pasar a la etapa 4 a activa. Si la etapa activa es la 3 debe** cumplirse la receptividad asociada a la transición **b, para que la etapa 4 pase** a estar activa.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Divergencias y convergencias.

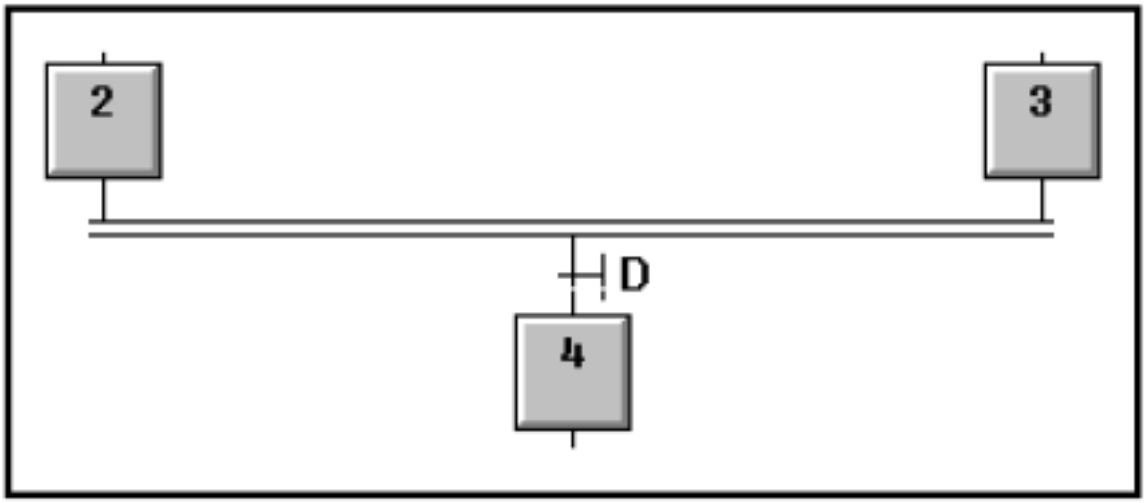
- **Divergencia en Y.**
  - Estando activa la etapa 1 y si se cumple la receptividad asociada a la transición C, pasan a estar activas las etapas 2 y 3.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Divergencias y convergencias.

- **Convergencia en Y.**
  - Para que se activa la etapa 4 deben estar activas las etapas 2 y 3 y cumplirse la receptividad asociada a la transición D.



Parte 3: Lenguajes de Programación **Lenguaje SFC**

Reglas de evolución

- **Regla 1: Inicialización**
  - En la inicialización del sistema se han de activar todas las etapas iniciales y sólo las iniciales.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Reglas de evolución

- **Regla 2: Evolución de las transiciones**
  - Una transición está validada cuando todas las etapas inmediatamente anteriores a ella están activas. Una transición es franqueable cuando está validada y su receptividad asociada es cierta. Toda transición franqueable debe ser obligatoriamente e inmediatamente franqueada.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Reglas de evolución

- **Regla 3: Evolución de las etapas activas**
  - Al franquear una transición se deben activar todas las etapas inmediatamente posteriores y desactivar simultáneamente todas las inmediatamente anteriores.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Reglas de evolución

- **Regla 4: Simultaneidad en el franqueamiento de las transiciones**
  - Las transiciones simultáneamente franqueables han de ser simultáneamente franqueadas.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Reglas de evolución

- **Regla 5: Prioridad de la activación**
  - Si al evolucionar un GRAFCET, una etapa ha de ser activada y desactivada al mismo tiempo, deberá permanecer activa.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Implementación del Grafcet sobre PLC

- El Grafcet representa el funcionamiento del proceso
  - Establece de forma más clara cuáles son las salidas y entradas en cada etapa.
  - Pero **no** es un lenguaje de programación.

- Nos interesa ahora ver la manera de implementar el Diagrama Grafcet en un PLC.

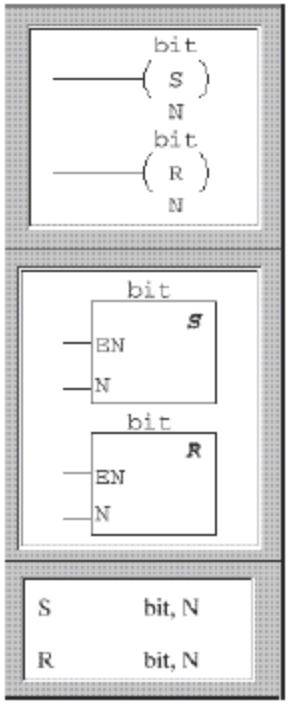
- Para ello a cada una de las etapas en las que se divide el Grafcet se le asocia una variable interna.

- La condición de transición es la encargada de activar la etapa siguiente(s) y desactivar la anterior(es); para ello se utilizan las instrucciones Set y Reset que poseen todos los autómatas programables.

Parte 3: Lenguajes de Programación

Lenguaje SFC

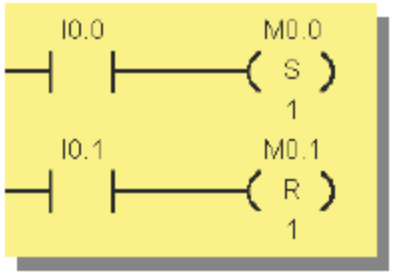
En el Siemens S7 las instrucciones SET y RESET son:



- Cuando se ejecutan las operaciones “Set” y “Reset”, se (se pone a 1) o se desactiva (se pone a 0) un núme- ro N tivos a partir de la dirección especificada.

- Las instrucciones Set y Reset se utilizan para activar o desactivar las marcas internas (M0.0, M0.1,...., etc.).

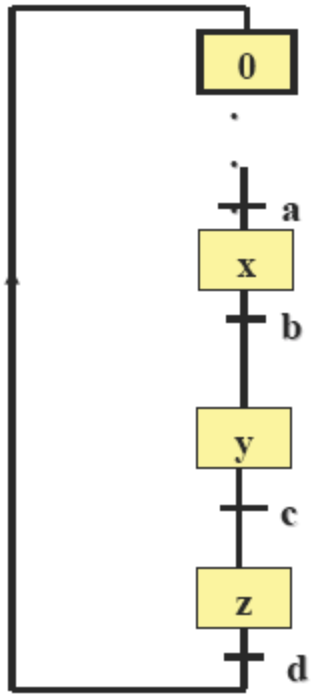
- Típicamente asociaremos cada etapa a una marca:
  - Etapa 0: M0.0. Etapa 1: M0.1 …
  - Cuando la etapa 0 esté activa, M0.0 valdrá 1…



Parte 3: Lenguajes de Programación

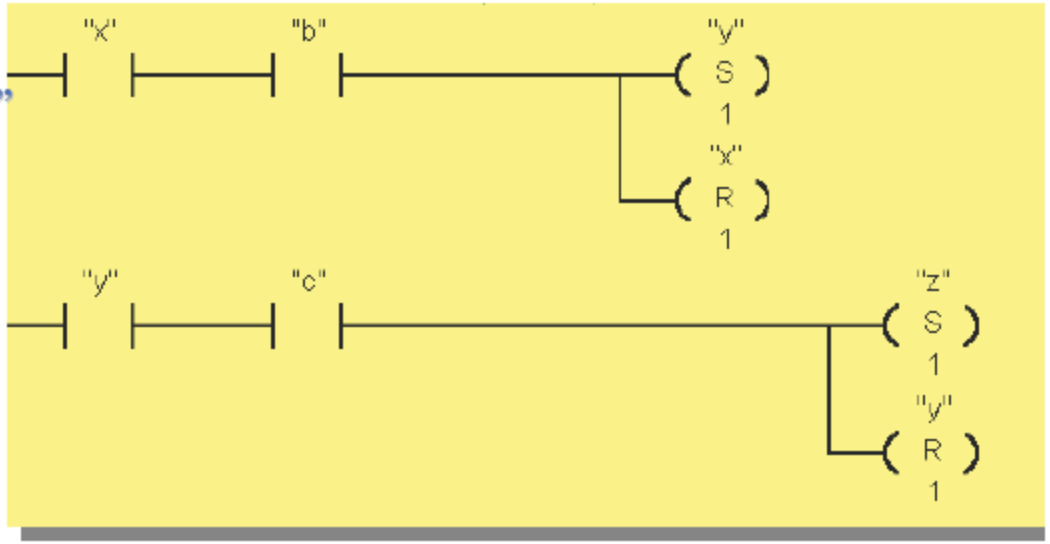
Lenguaje SFC

Implementación en KOP de transiciones sencillas:



Activa la etapa “y” y desactivar la etapa “x”

Activa la etapa “z” y desactivar la etapa “y”



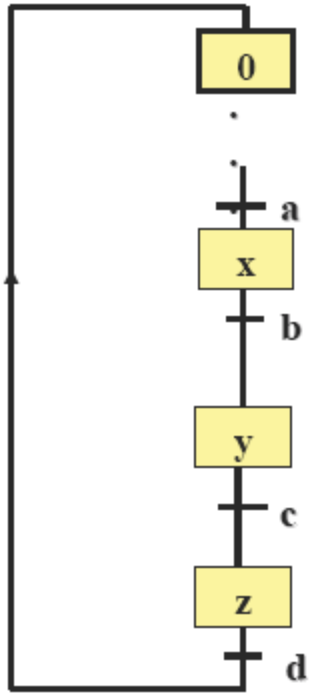
Símbolo	Dirección	Comentario
x	M0.1	Etapa x
y	M0.2	Etapa y

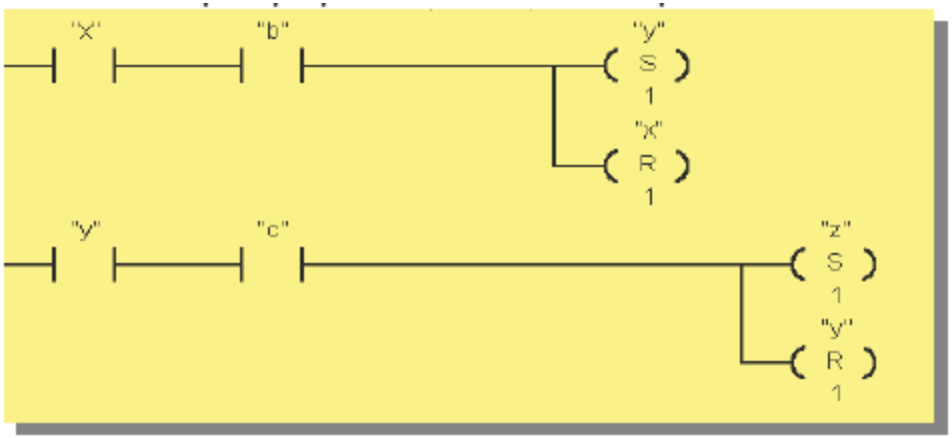
Parte 3: Lenguajes de Programación

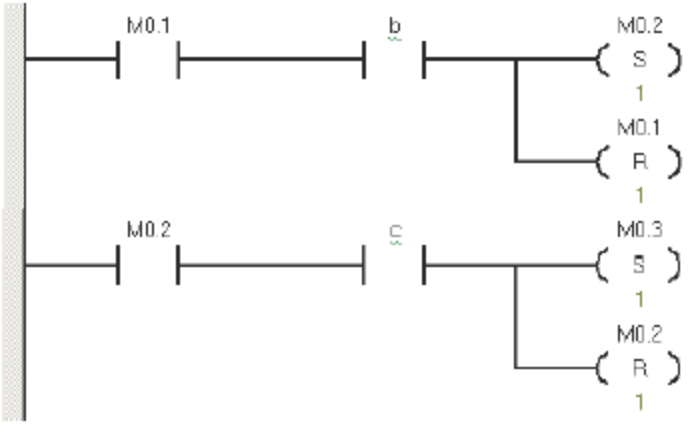
Lenguaje SFC

Implementación en KOP de transiciones sencillas:

- Que se puede leer:… si la etapa “x” está activada y se cumple la transición “b”, entonces activar la etapa “y” y desactivar la etapa “x”







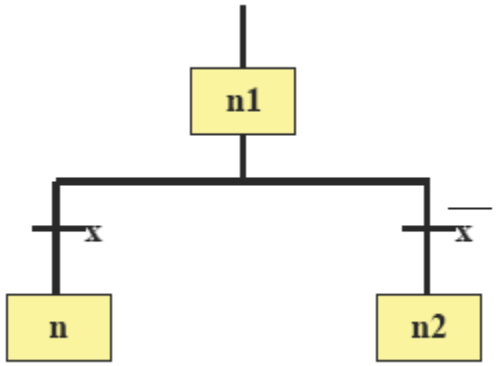
Símbolo	Dirección	
x	M0.1	Etapa x
y	M0.2	Etapa y

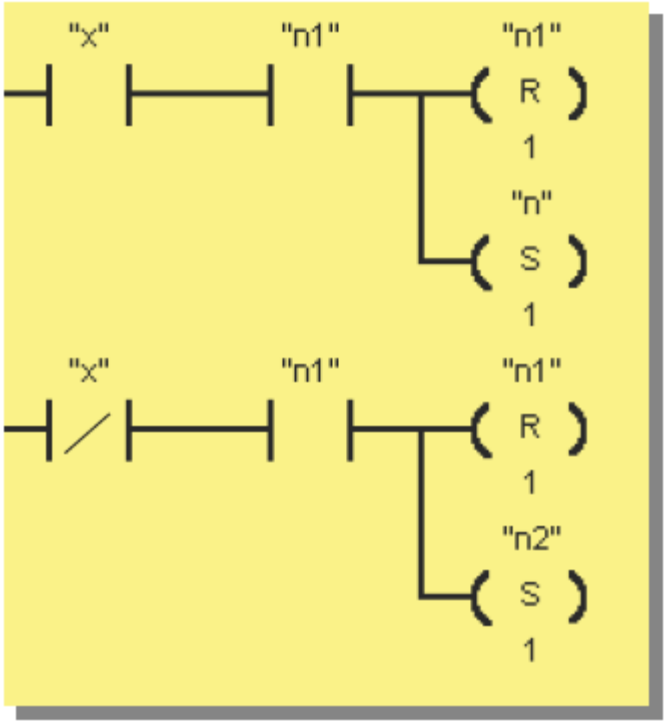
Parte 3: Lenguajes de Programación

Lenguaje SFC

Varios ejemplos de cómo codificar en lenguajes de contactos algunos casos que se pueden dar en diagramas Grafcet

- Divergencia OR

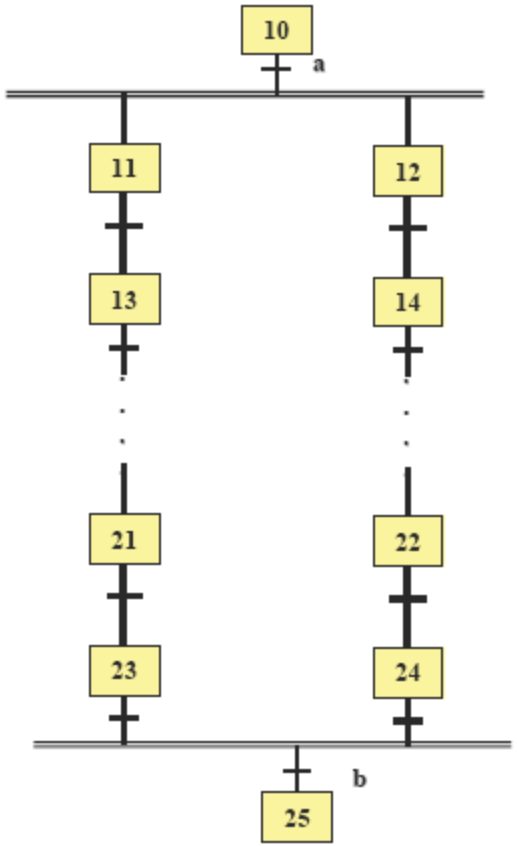


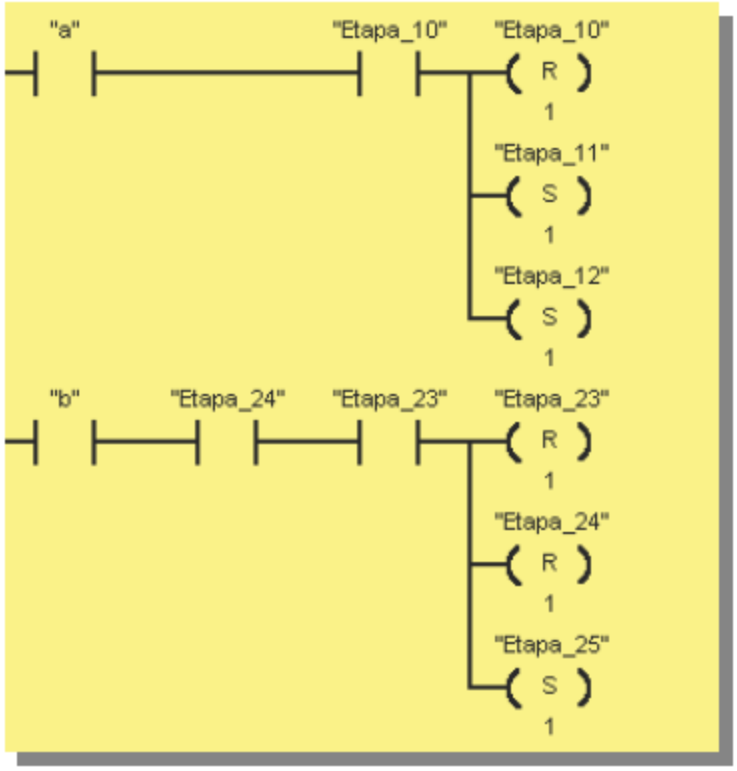


Parte 3: Lenguajes de Programación

Lenguaje SFC

Caso de secuencias paralelas: divergencia y convergencia AND

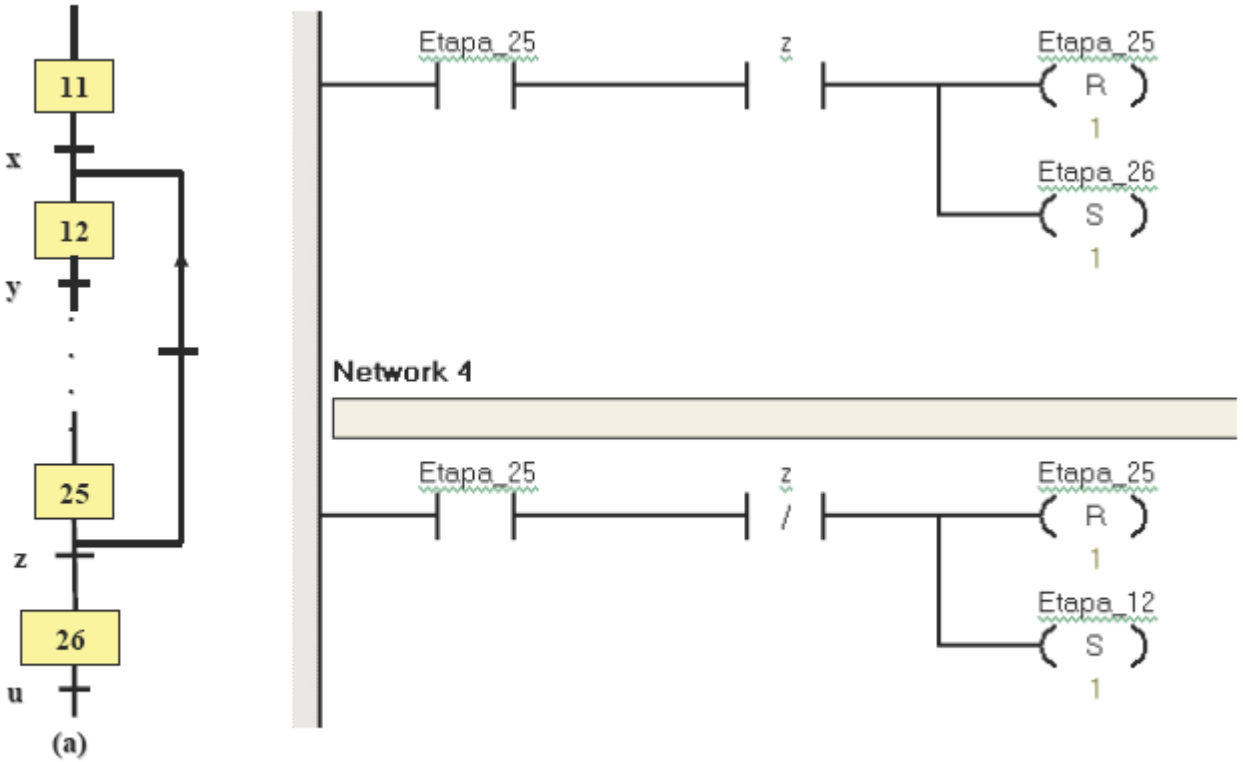




Parte 3: Lenguajes de Programación

Lenguaje SFC

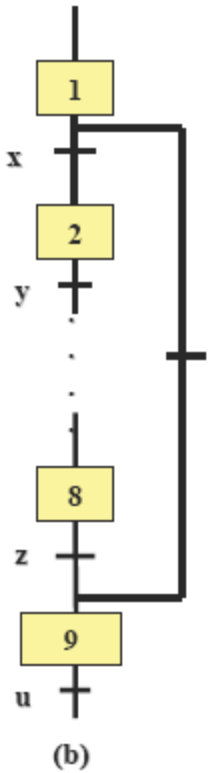
Saltos condicionales a otras etapas



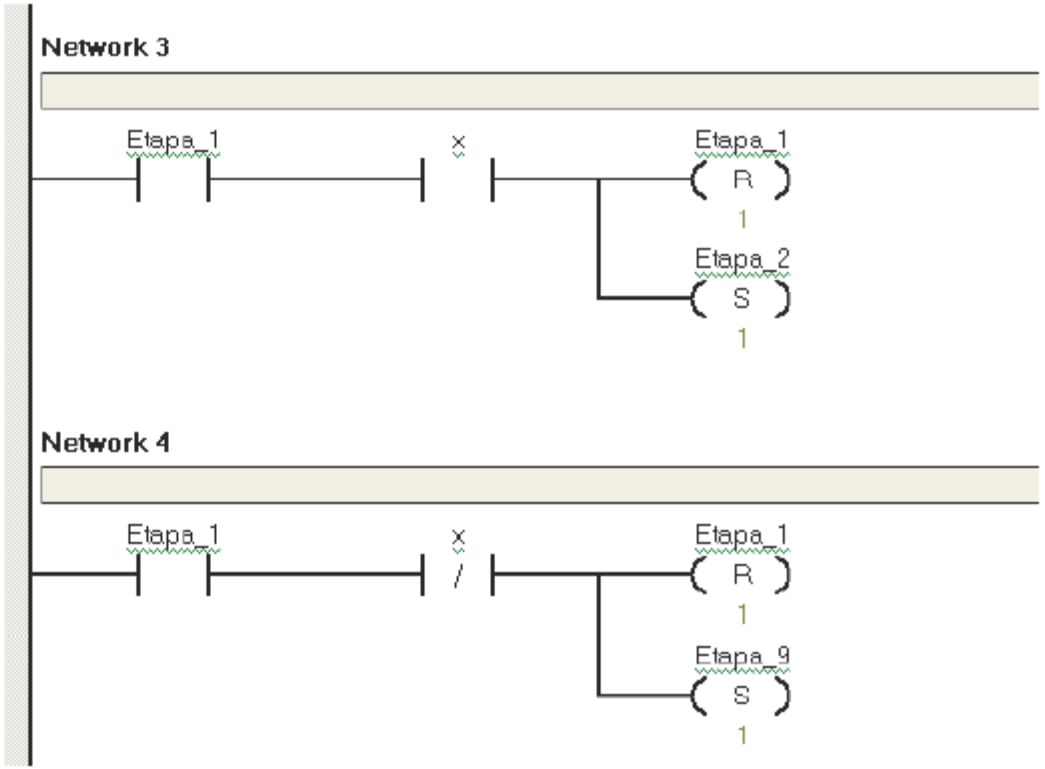
Parte 3: Lenguajes de Programación

Lenguaje SFC

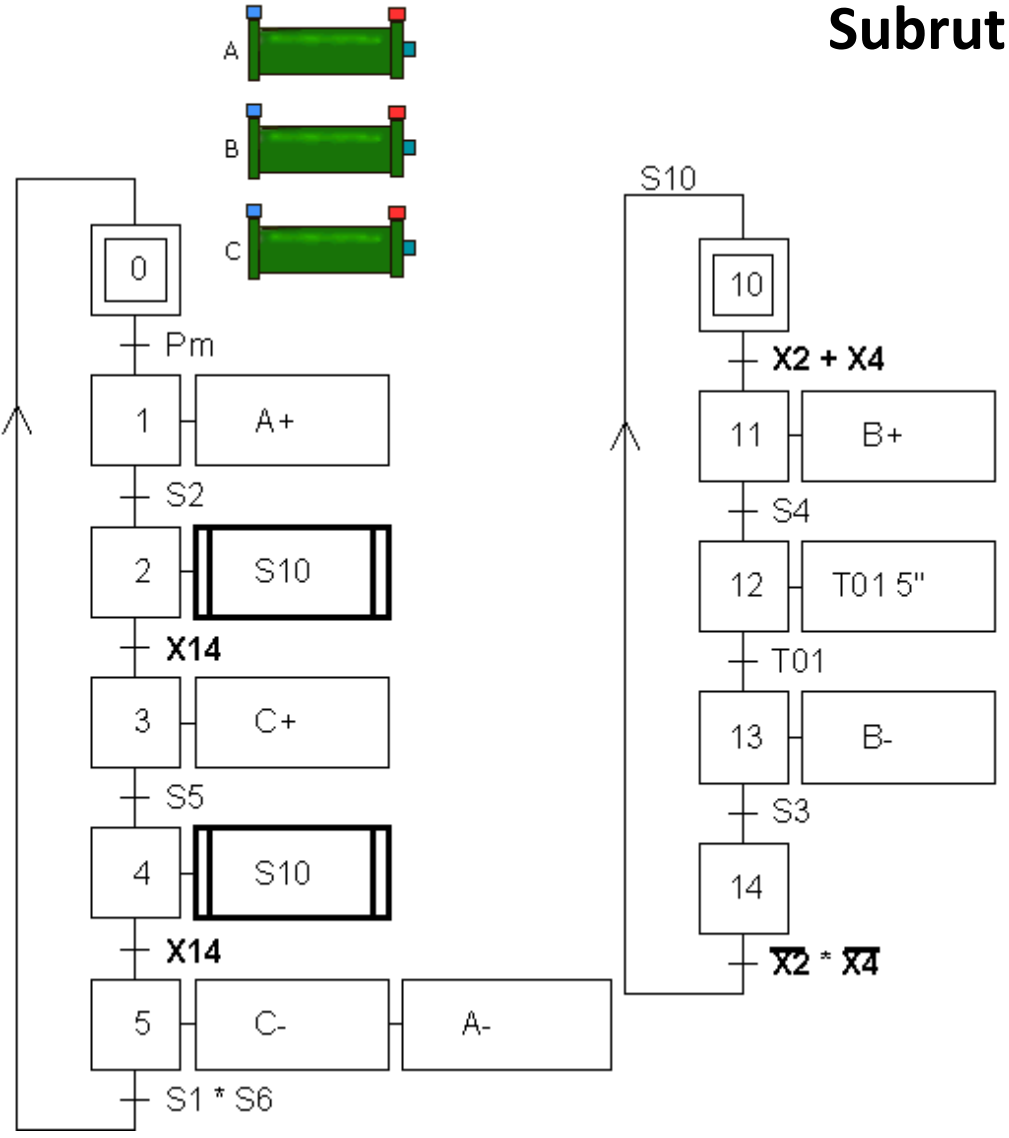
Saltos condicionales a otras etapas



(b)



Subrutinas



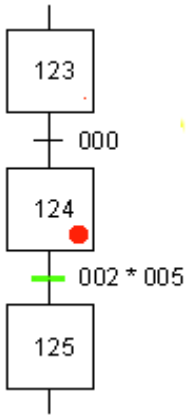
Los trabajos a desarrollar en un automatismo se pueden dividir entre diferentes diagramas. Puede haber un diagrama principal (0-5) y otros de secundarios (10-14) que hacen determinadas funciones que una vez realizadas devuelven el control al diagrama principal.

Al llegar a la etapa 2 o 4 del primer diagrama se valida la transición X2+X4 y empieza la subrutina. Al llegar a la etapa 14 se valida la transición X14 y continua la evolución del diagrama principal a las etapas 3 o 5 respectivamente.

Parte 3: Lenguajes de Programación

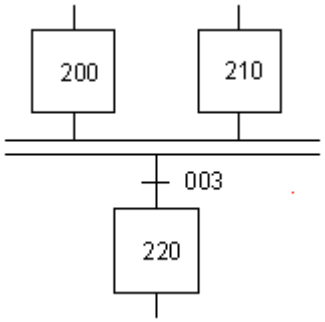
Lenguaje SFC

Evolución del Sistema



La primera transición se podrá validar, si la etapa 123 esta activa, y ademas se cumple la condición 000. En este momento deja de estar activa la etapa 123, y le toma el relevo la 124.

El grafcet evolucionara a la etapa 125, si estando activa la etapa 124 se cumple la condición 002 y también la 005



Las etapas 200 y 210 son etapas de entrada a la transición.

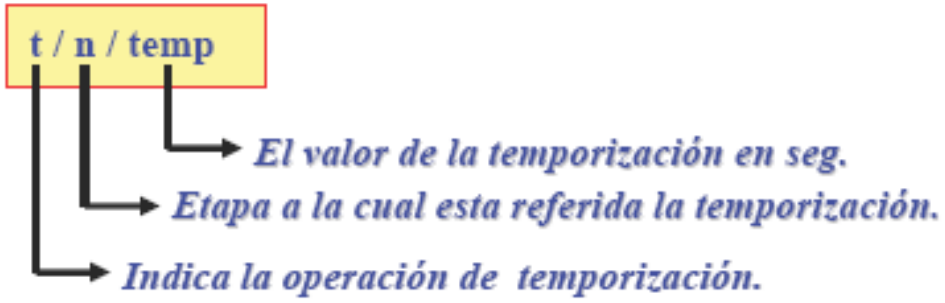
Para validar la transición, deben estar activas las dos etapas.

Para poder entrar a la etapa 220, la transición tiene que estar validada y se debe de cumplir la receptividad asociada (003) a la transición.

Parte 3: Lenguajes de Programación **Lenguaje SFC**

Temporizadores y contadores

• Función temporización en Grafcet se implementa a través del operador de temporización

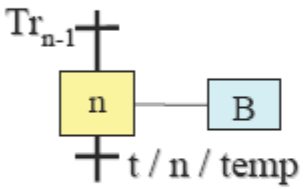


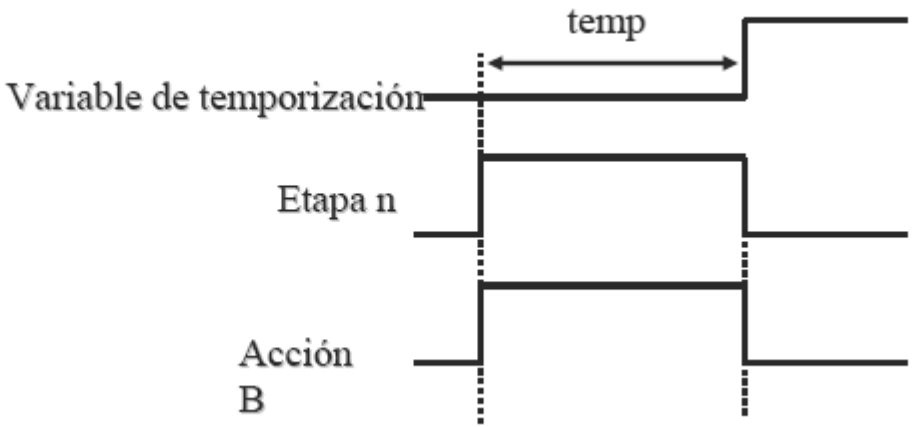
Parte 3: Lenguajes de Programación

Lenguaje SFC

Temporización de la transición de una etapa:

Es el caso en el cual la receptividad asociada a una transición depende de que la variable de temporización sea activada. Si se trata de una temporización con retardo esta transición no será superada hasta que transcurra un cierto instante de tiempo.



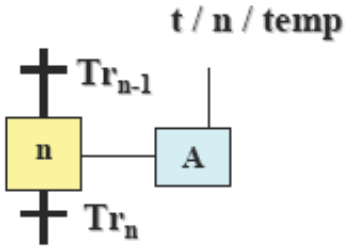


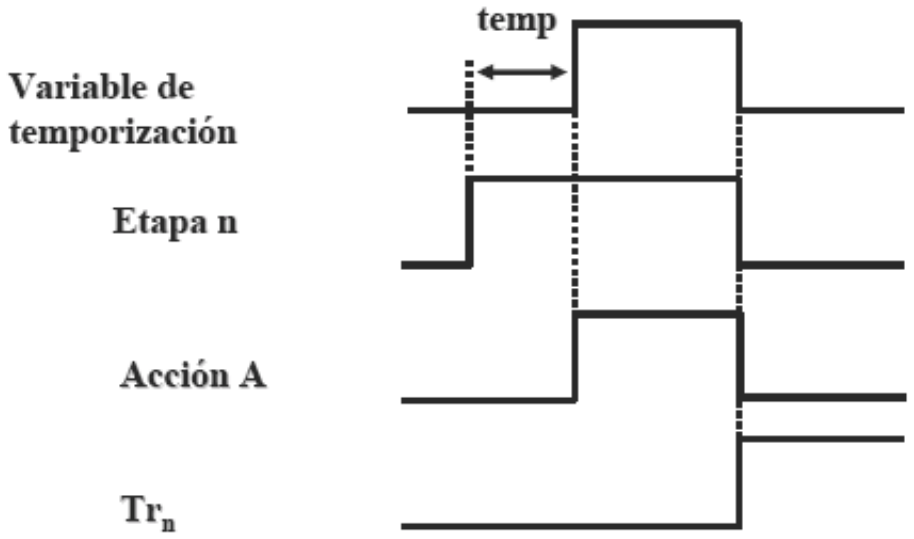
Parte 3: Lenguajes de Programación

Lenguaje SFC

Incorporación de las temporizaciones al Grafcet:

• Temporización de acciones: Es el caso en el cual se pretende temporizar la ejecución de la acción asociada a una etapa, de forma que no se ejecute la acción hasta que transcurra un cierto instante de tiempo.

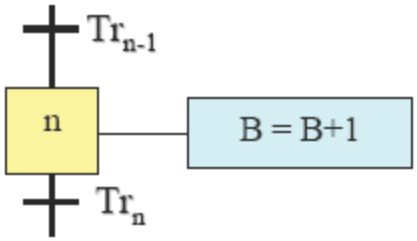


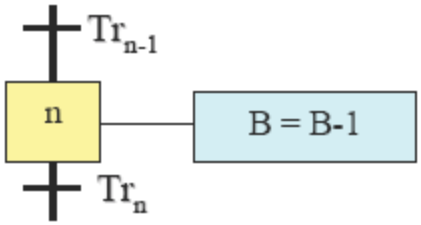


Parte 3: Lenguajes de Programación

Lenguaje SFC

Contadores

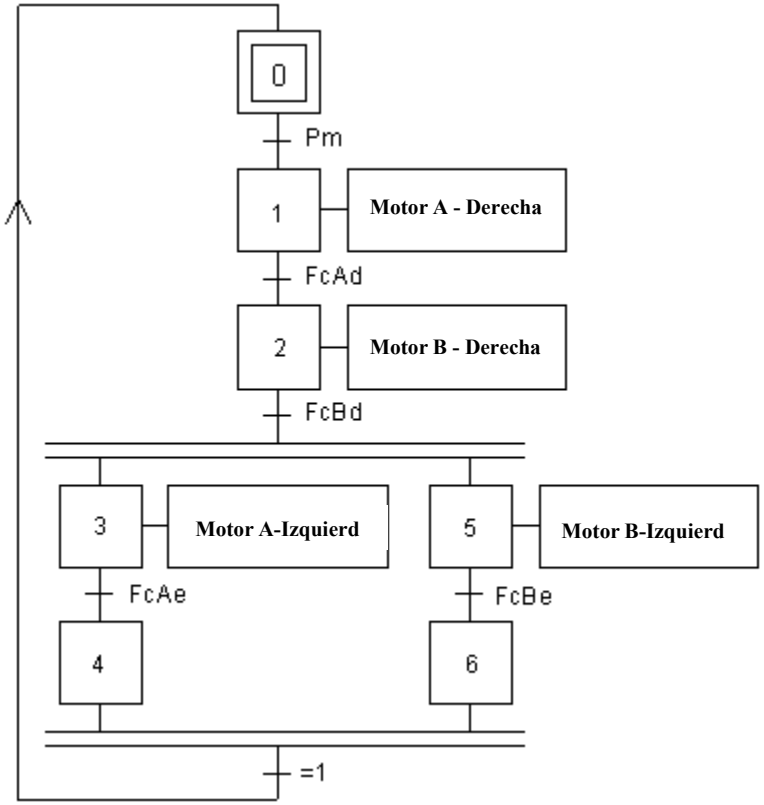




Parte 3: Lenguajes de Programación

Ejemplos

Motores con trabajos simultáneos

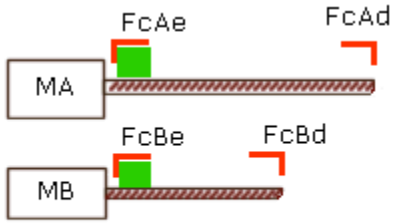


Dos motores MA y MB, desplazan unas piezas.

Primero el motor MA va desde FcAe a FcAd, entonces es el MB quien lo hace desde FcBe hasta FcBd.

Después los dos vuelven a las posiciones iniciales FcAe y FcBe.

El ciclo se re inicia cuando los dos están de nuevo en las posiciones iniciales.



Ejemplo Transfer Circular

Queremos preparar un automatismo para una máquina transfer:
Consta de un plato giratorio, con cuatro estaciones de trabajo: Alimentación, Preparación, Mecanizado y Salida.

Alimentación: Las piezas descienden por una rampa, dos detectores (uno inductivo y uno capacitativo) perciben la presencia de pieza.
Las piezas **metálicas, son detectadas por los dos detectores; mientras que las sintéticas solo las detecta el capacitativo.**
Un cilindro actúa como barrera dosificadora, controlando el acceso de las piezas al soporte giratorio.

Preparación: Las piezas metálicas se someten a un baño de color y a un acabado, mientras que las sintéticas solo al proceso de acabado.

Mecanizado: El taladro actúa durante 5'' sobre las piezas metálicas, en las piezas sintéticas se reduce a 2''

La estación de **salida desplaza las piezas a la rampa de salida.**

Cada estación de trabajo, debe actuar conforme al tipo de pieza que tenga que tratar. La información del tipo de piezas se guarda en un registro de desplazamiento, que está controlado por él movimiento del plato.

Ejemplo Transfer Circular

Niveles de trabajo.

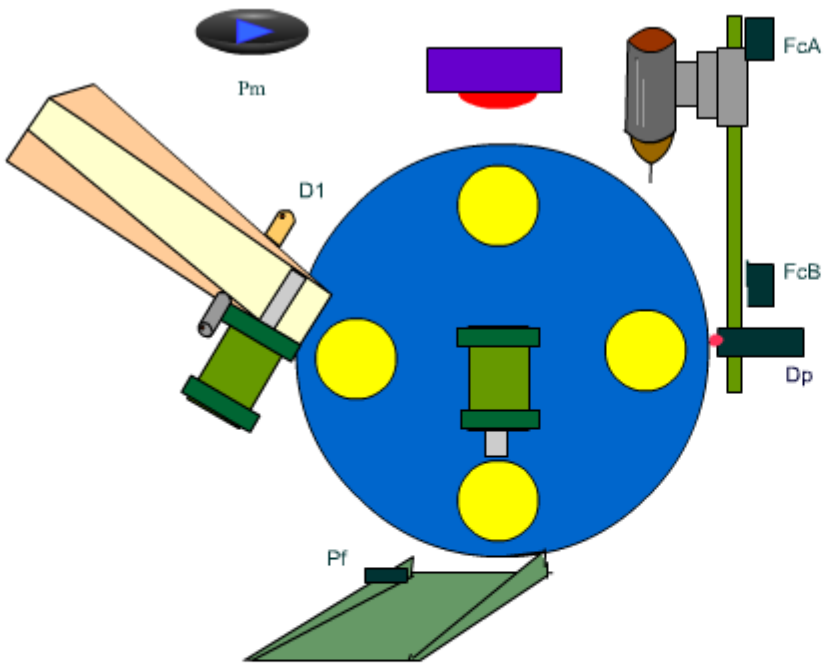
Nivel 1.
Hay un solo tipo de piezas, y entran de una en una a la estación de trabajo. Se trata de un proceso puramente secuencial, la segunda pieza no entra hasta que la primera pieza no este mecanizada y fuera del plato.

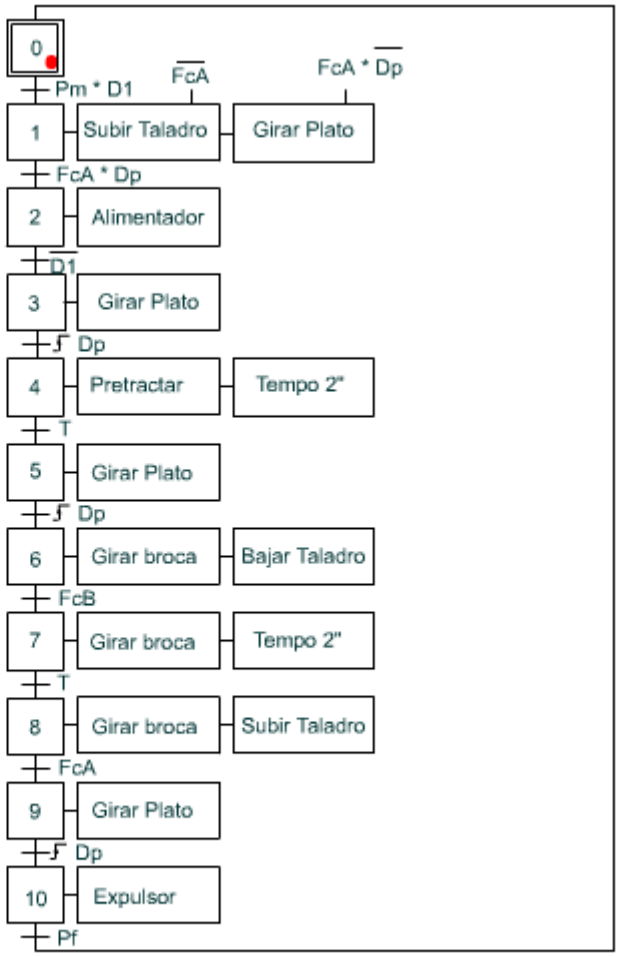
Nivel 2.
Hay dos tipos de piezas, pero no puede entrar la segunda pieza hasta que esté terminada la primera.

Nivel 3.
Cada estación de trabajo actuara conforme la pieza a tratar, conforme el enunciado principal.

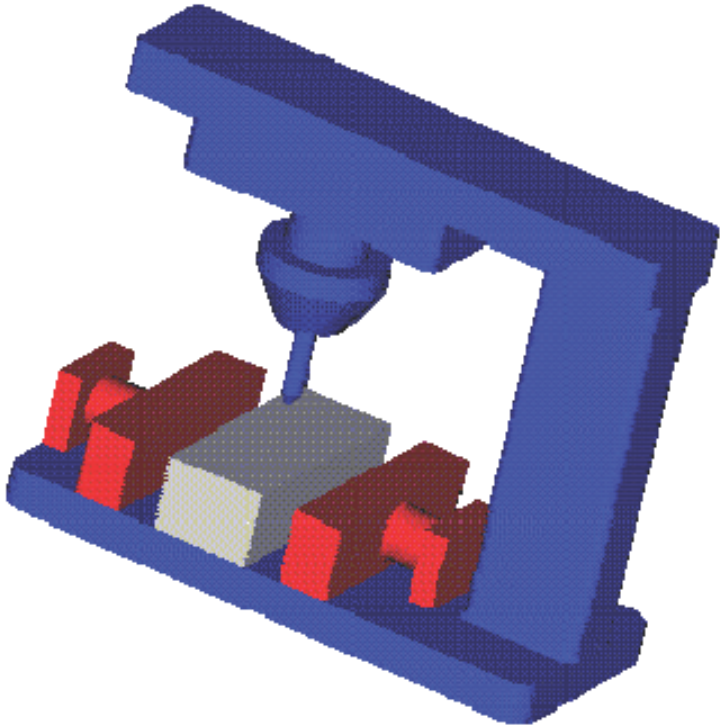
Nivel 4.
En una fase final se pueden incorporar, un pulsador de paro, un paro a final de ciclo y uno de emergencia.

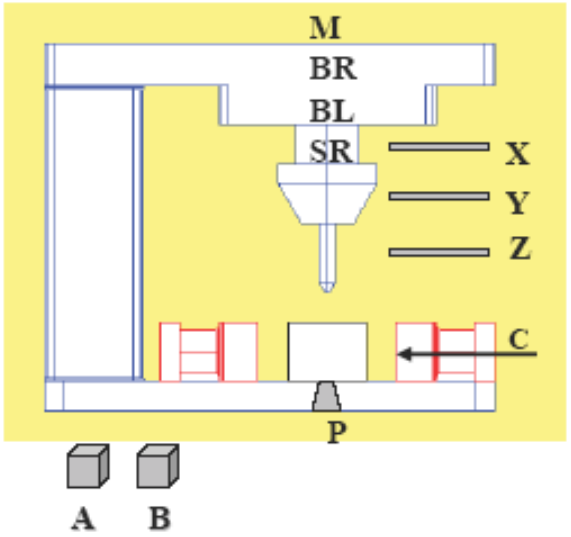
Ejemplo Transfer Circular





Sistema de control de una máquina taladradora





Sistema de control de una máquina taladradora

Descripción del sistema:

- Existe un pulsador "B", de encendido del sistema. Una vez presionado, se acciona el motor (M) hasta que alcanza una velocidad de giro de régimen permanente.

- Existe un botón A que inicia la operación de taladrado.

- El taladro posee varias velocidades en el sentido longitudinal del eje,
  - bajada lenta del utensilio del taladro BL
  - bajada rápida BR y
  - subida rápida SR.

- Existen detectores de presencia del tornillo a diferentes alturas:
  - X
  - Y
  - Z

- Existe un detector de presencia de pieza en la presa (detector inductivo P).

- Existe un cilindro neumático que sujeta la pieza (accionado mediante C).

Sistema de control de una máquina taladradora

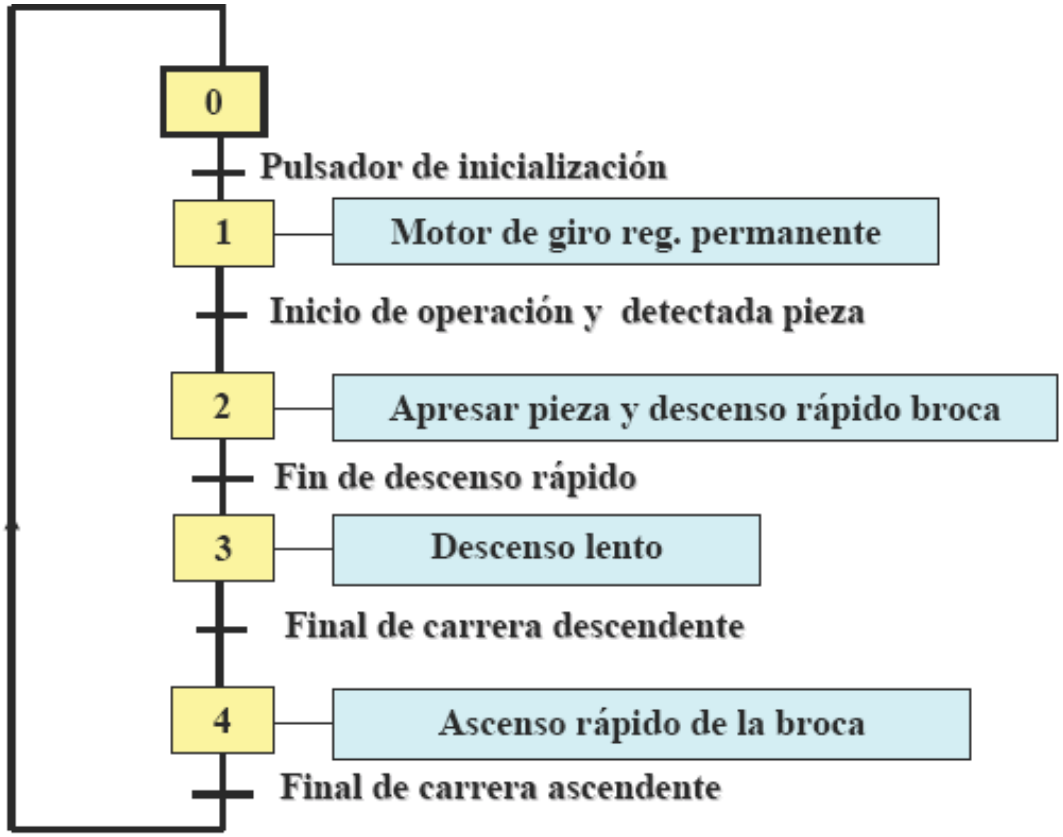
Funcionamiento de la taladradora:

• La pieza en la que se va a realizar el taladro se detecta mediante un detector inductivo P, y se sujeta mediante dos sujeciones accionadas por C. La tarea de realizar un taladro sigue la siguiente secuencia: primero se detecta la pieza mediante el detector inductivo, posteriormente se pulsa el botón "A" de inicio de operación con lo que actúan las sujeciones de la pieza y al mismo tiempo se inicia el descenso rápido de la broca "BR".

• Antes de empezar a realizar el taladro propiamente dicho a la pieza, el detector "Y" provoca el paso de descenso rápido de la broca a descenso lento "BL", el cual se interrumpe cuando se detecta el final de carrera "Z". Inmediatamente se produce la subida rápida de la broca hasta alcanzar la posición de reposo "X" y se libera la pieza.

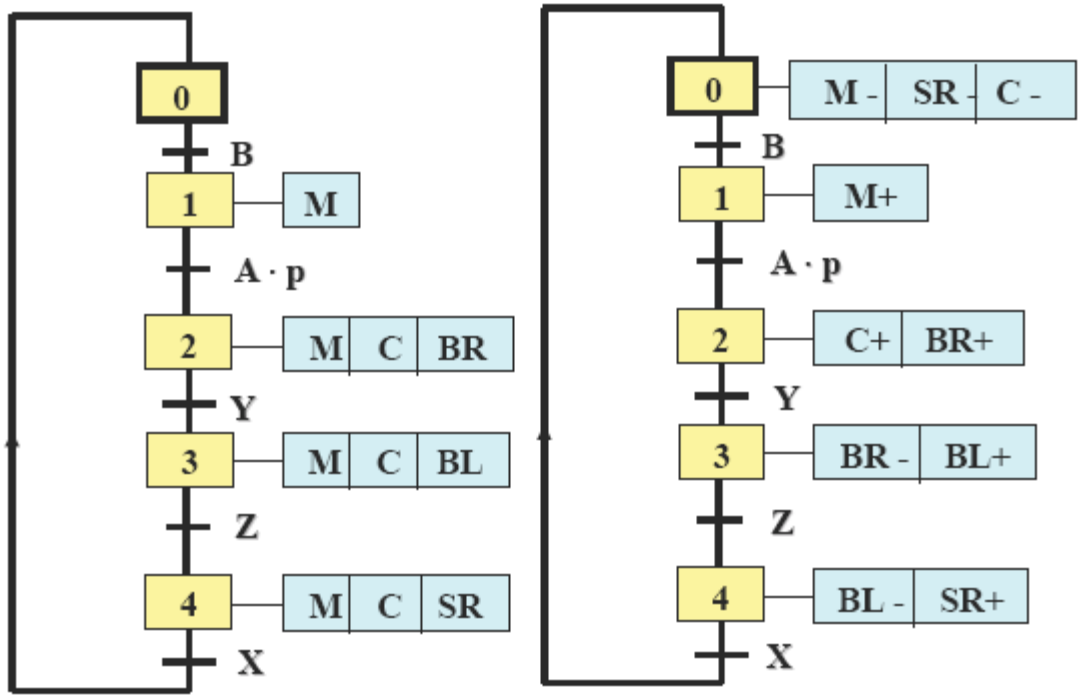
Sistema de control de una máquina taladradora

Grafcet Nivel 1



Sistema de control de una máquina taladradora

Grafcet Nivel 2



(a) Pr. monoestable

(b) Pr. biestable

Sistema de control de una máquina taladradora

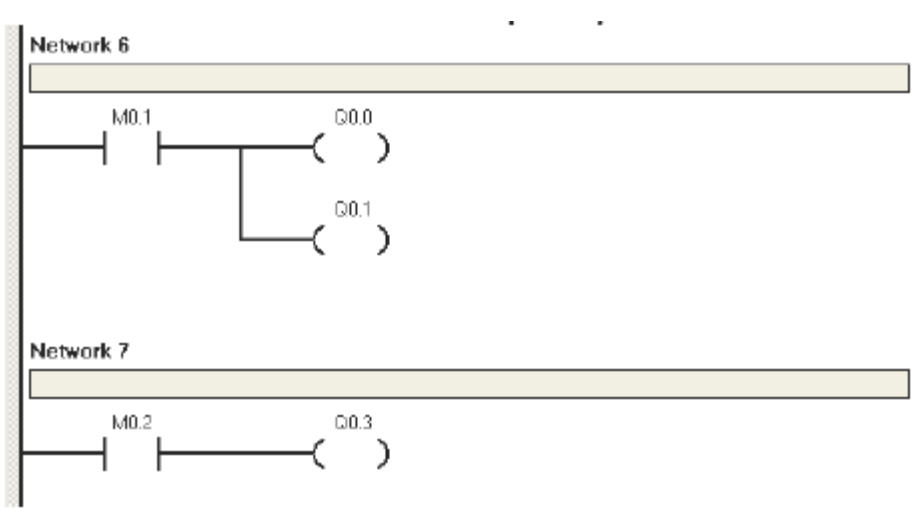
La representación es ligeramente diferente:

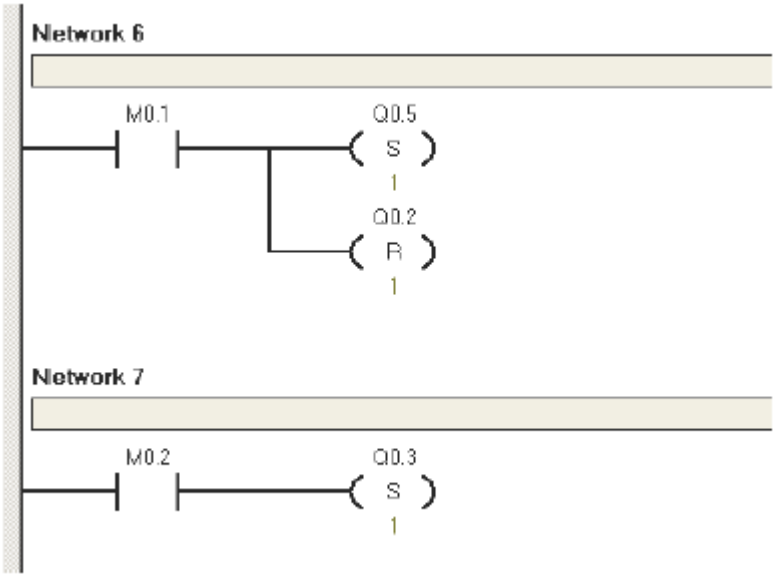
- En el caso biestable, se representa la activación y desactivación de ciertas salidas.

P.e. se puede realizar con funciones Set y Reset. Una etapa puede activar una salida, mientras que otra etapa diferente la desactiva.

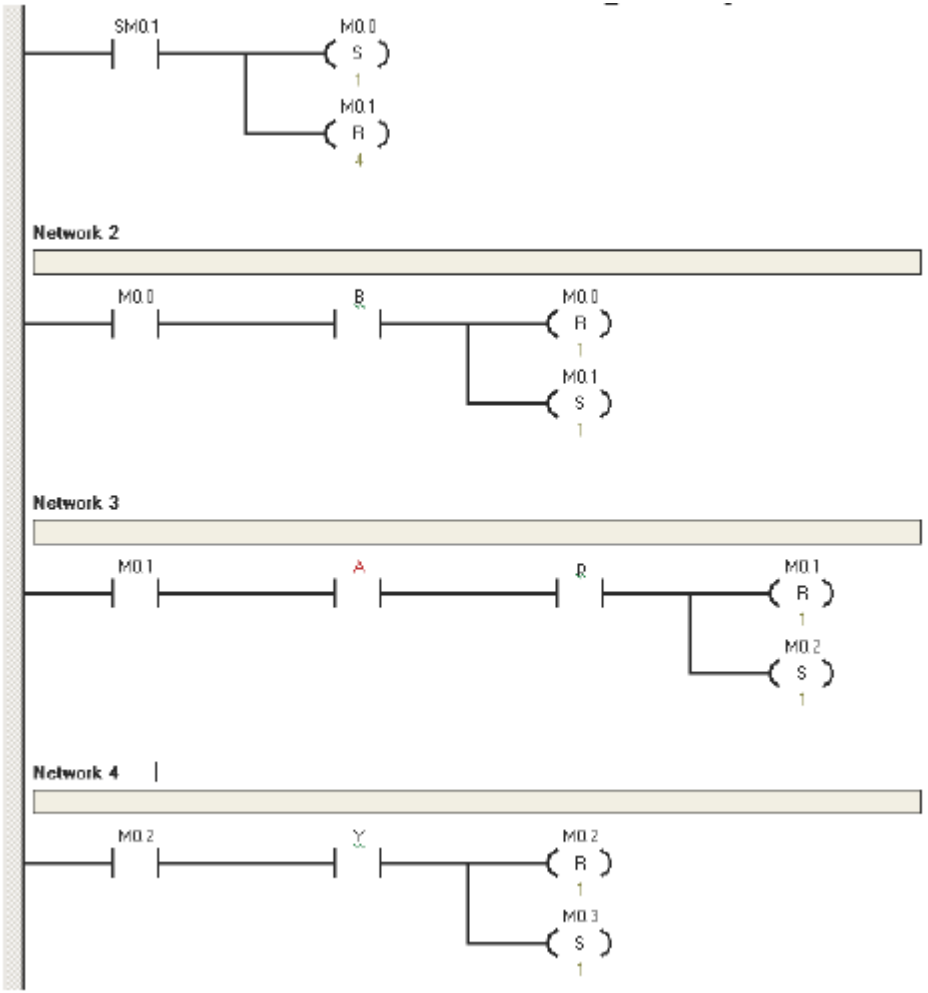
- En el caso monoestable, las salidas están directamente asociadas a las entradas.

- A la hora de programar:

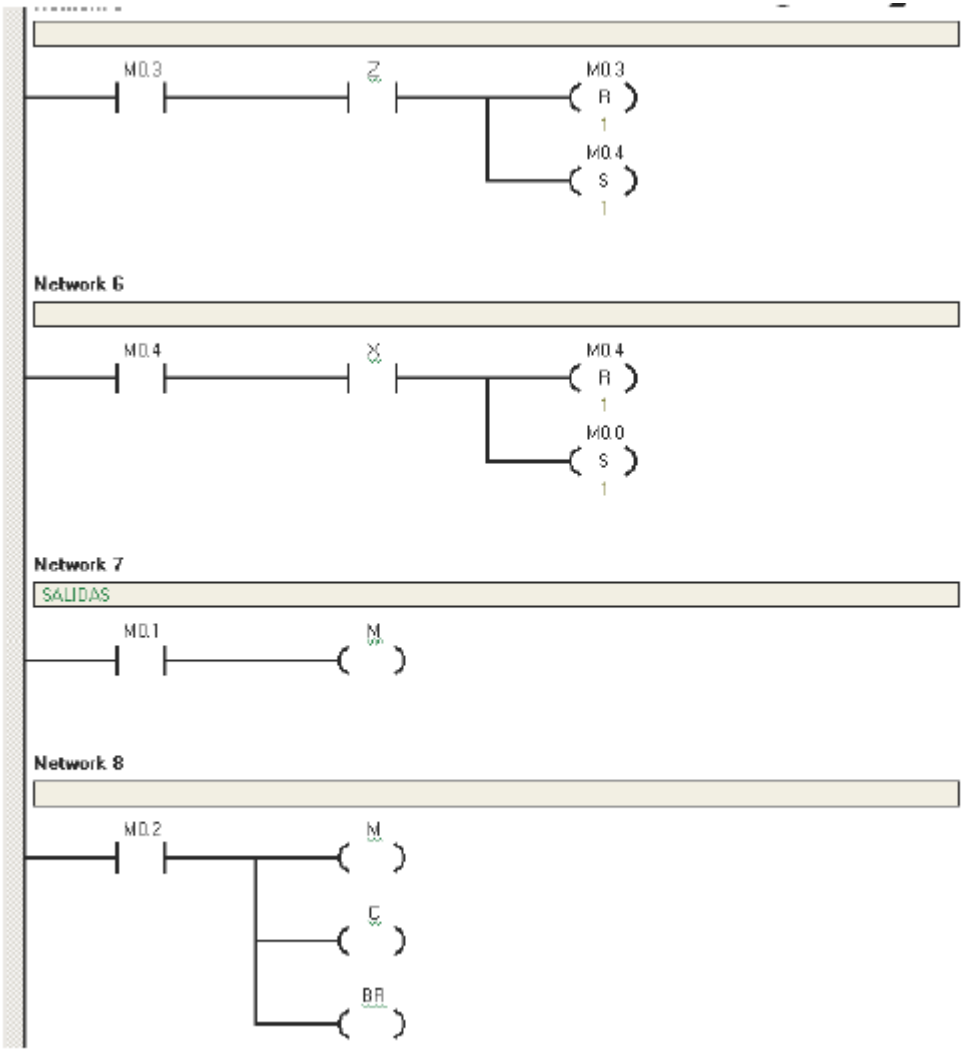




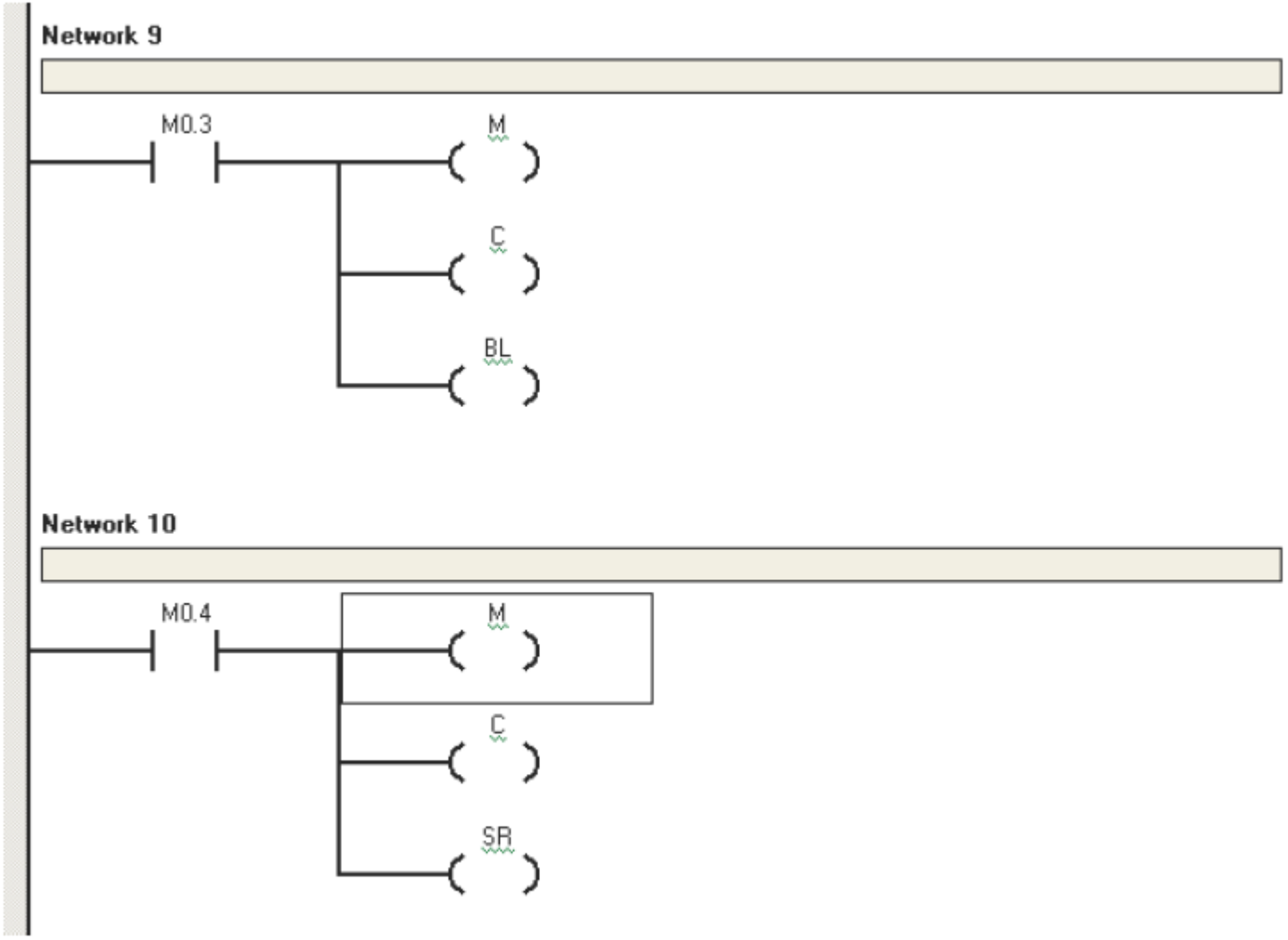
Sistema de control de una máquina taladradora



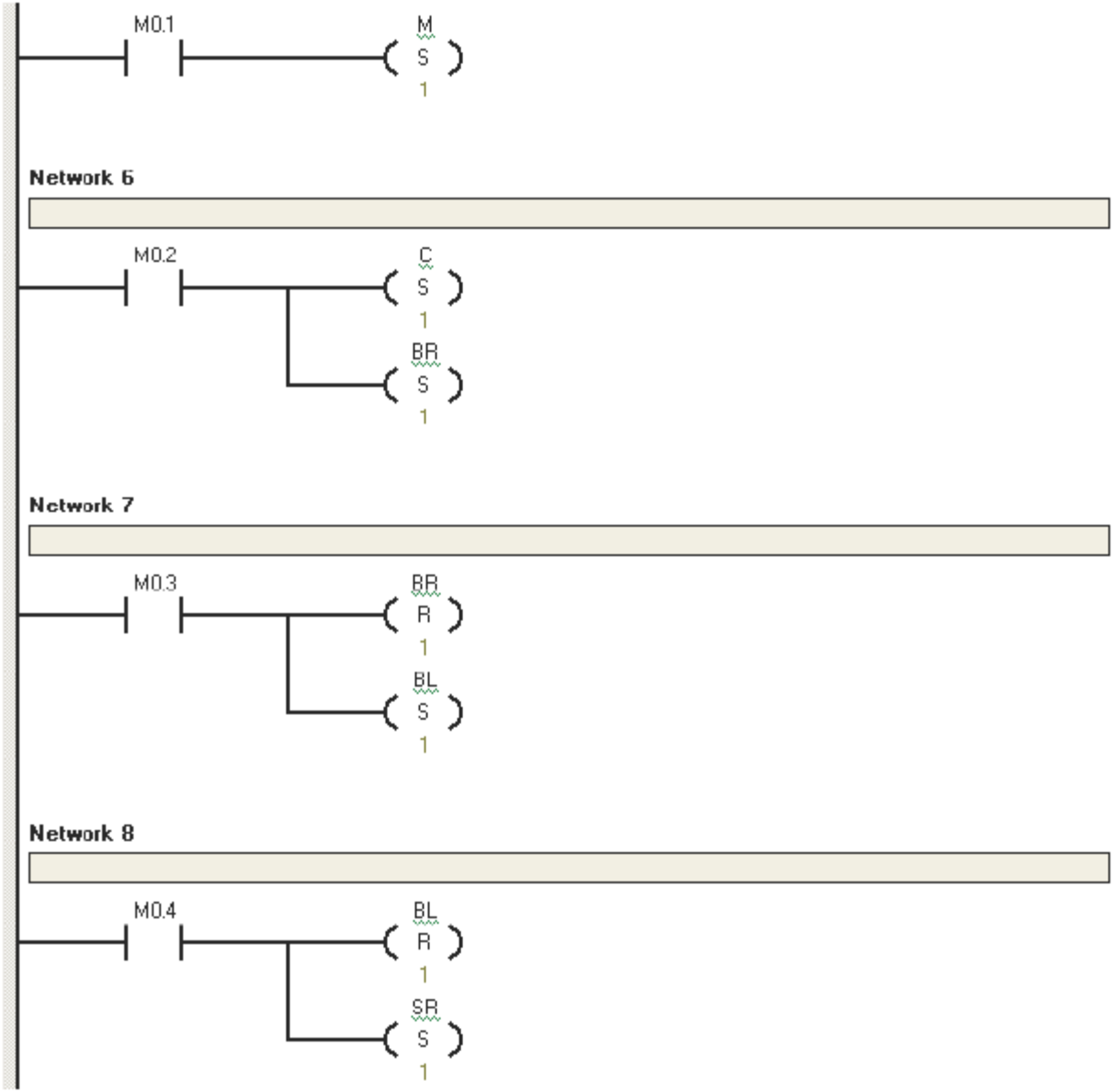
Sistema de control de una máquina taladradora



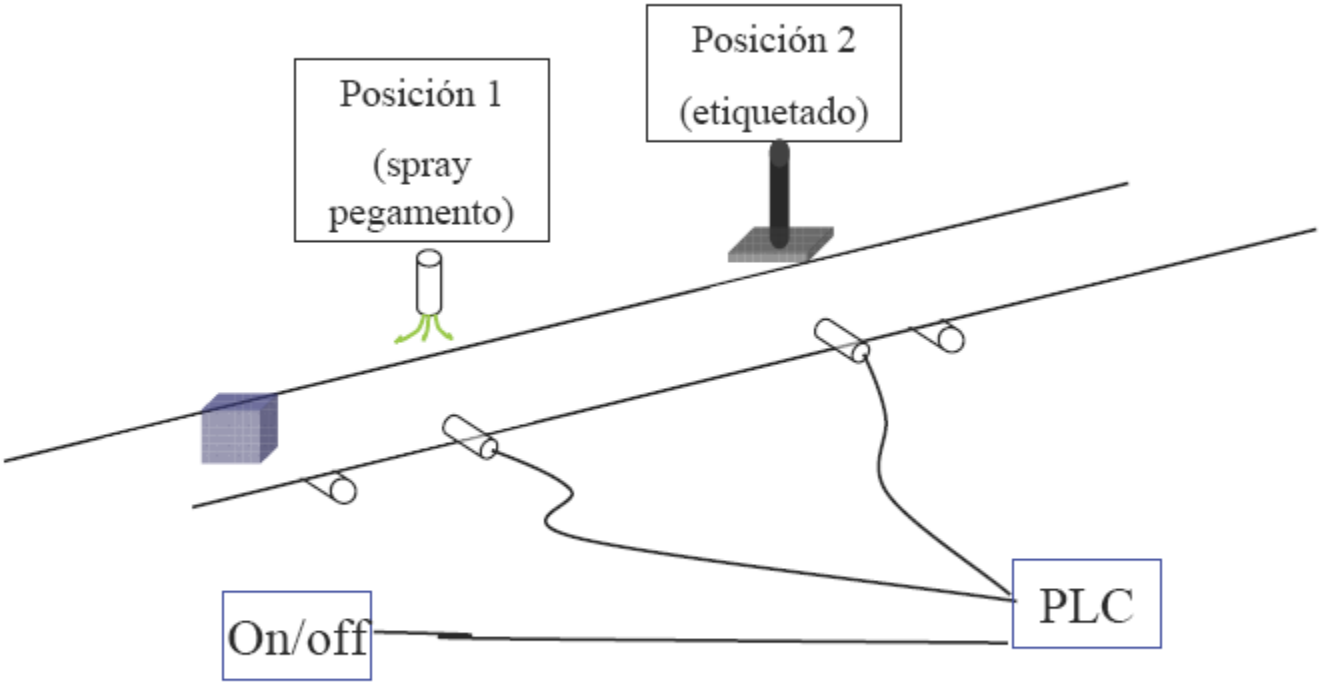
Sistema de control de una máquina taladradora



Sistema de control de una máquina taladradora



Máquina de etiquetado



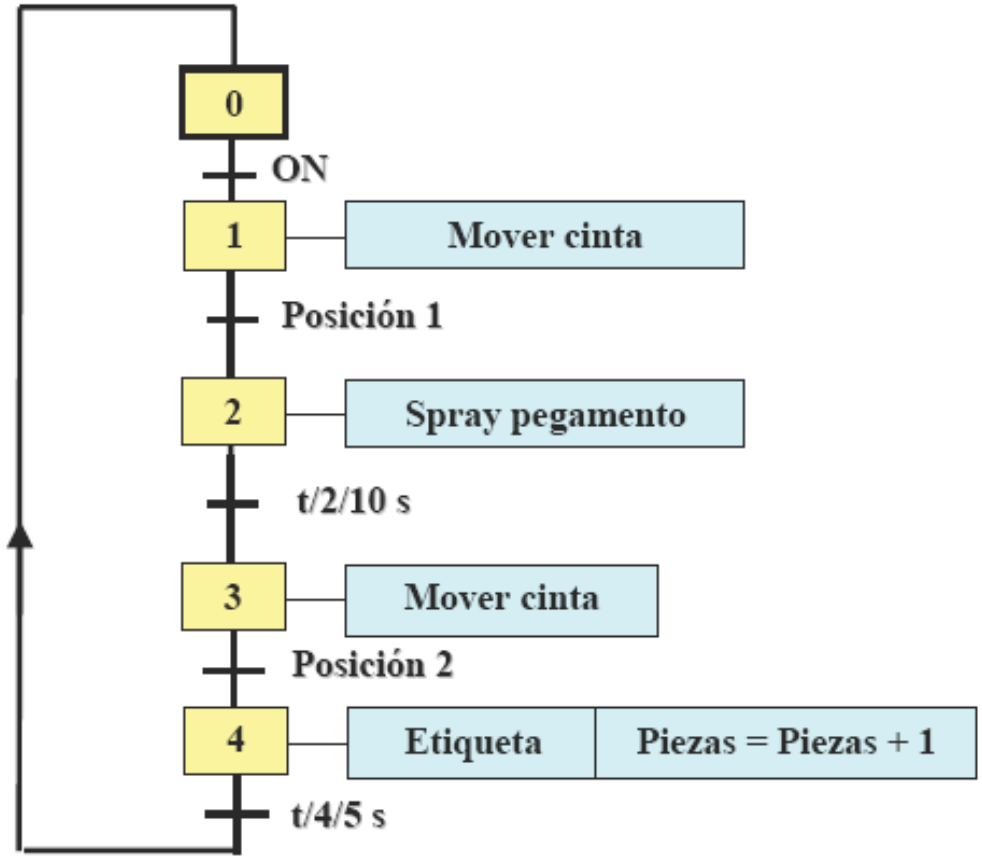
Máquina de etiquetado

Funcionamiento:

- El botón on/off pone en marcha o para el sistema.
- La cinta avanza hasta la posición 1.
- Se esperan 10 segundos.
- A continuación, la cinta avanza hasta la posición 2.
- Se pega una etiqueta y se cuenta una pieza fabricada.
- Se esperan cinco segundos (se mantiene la etiqueta presionada)
- La cinta vuelve a avanzar hasta que la nueva pieza llegue a la posición 1.
- Si On se encuentra activo, se continúa con el proceso.
- Si Off, se termina la última pieza y se para

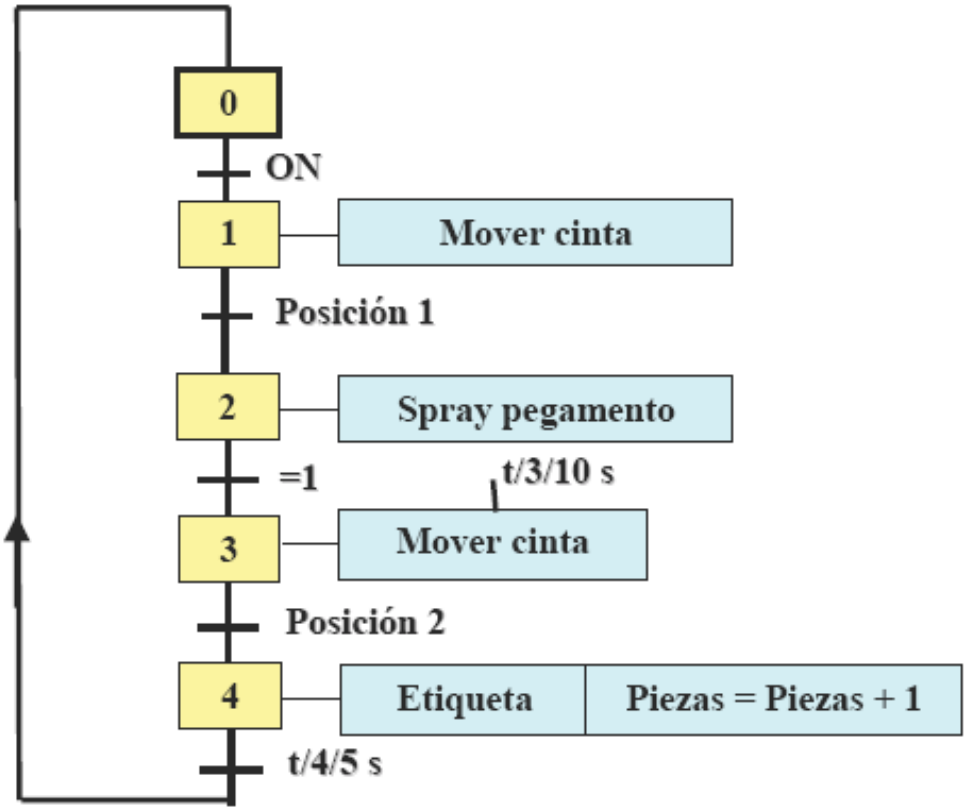
Máquina de etiquetado

Grafcet Nivel 1



Máquina de etiquetado

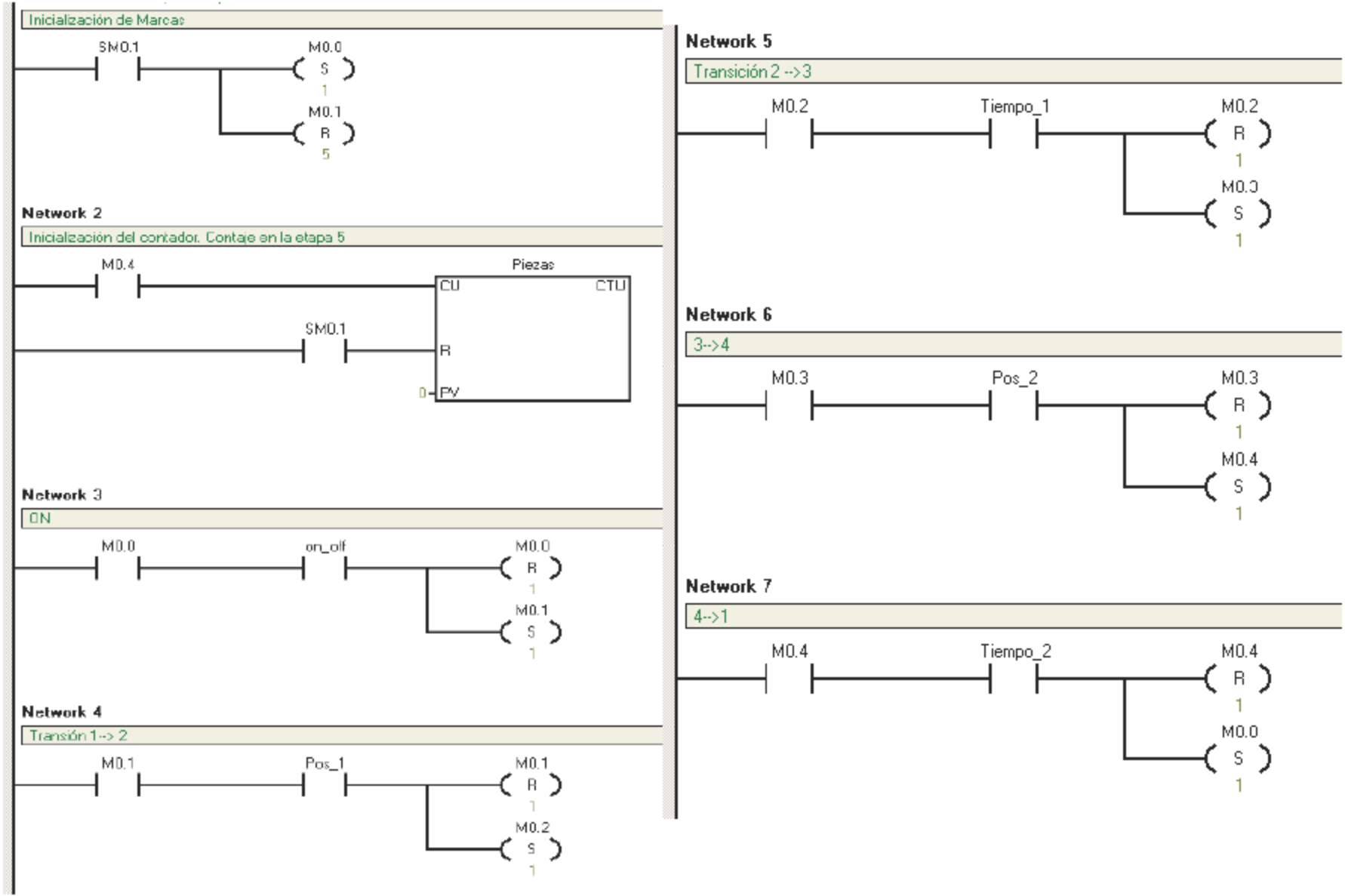
Grafcet Nivel 1



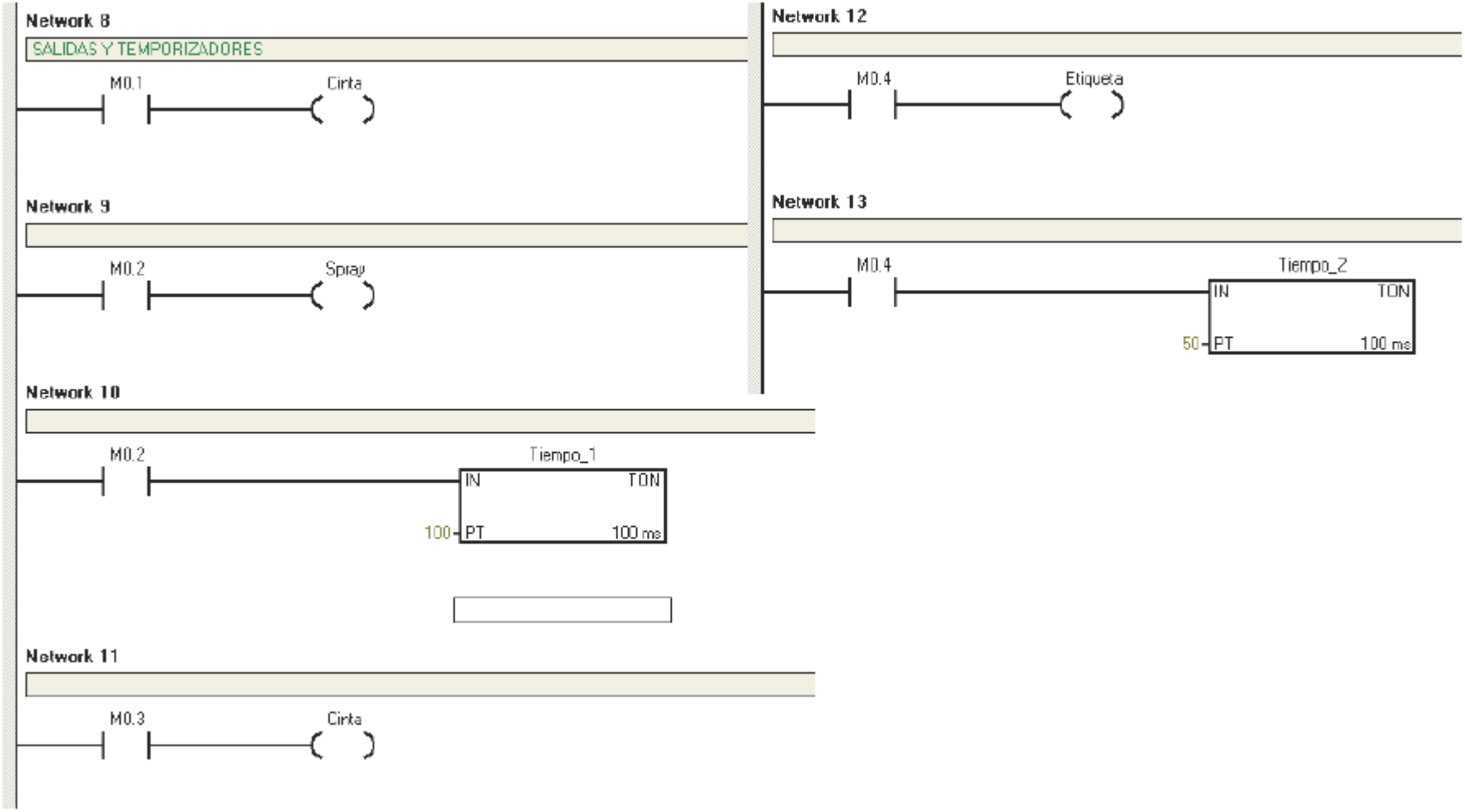
Máquina de etiquetado

Símbolo	Dirección	Comentario
Cinta	Q0.0	Mover cinta
Spray	Q0.1	Activar spray pegamento
Etiqueta	Q0.2	Poner etiqueta
Pos_1	I0.1	Posición 1
Pos_2	I0.2	Posición 2
Piezas	C0	contador de piezas
Tiempo_1	T37	Temporizador 1
on_off	I0.3	Interruptor general
Tiempo_2	T38	Temporizador 2

Máquina de etiquetado



Máquina de etiquetado



Ejercicio: Control de una escalera mecánica

Sensores:

- Interruptor general on_off I0.3 (24V on, 0V off)
- Interruptor (seta) de emergencia, con enclavamiento PE I0.0 (24V off, 0V on)
- Sensor de temperatura del motor TM I0.1 (24v ok, 0v, exceso ttura)
- Sensor óptico a la entrada de la escalera SO I0.2 (24v detección, 0v no detección)

Actuadores:

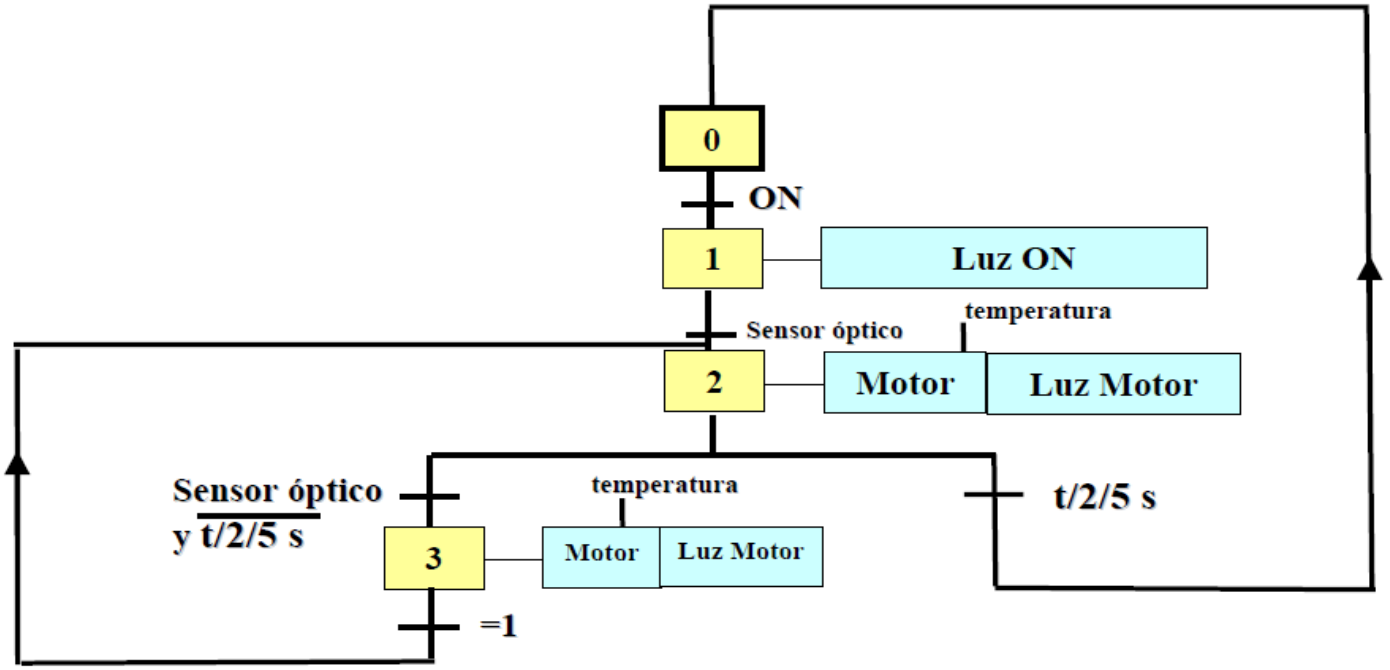
- Motor escalera Q0.0 (24v marcha)
- Luz sistema on Q0.1
- Luz motor on Q0.2
- Luz emergencia Q0.3

Funcionamiento:

- El sistema se pone en marcha al apretar on.
- Normalmente, la escalera está parada.
- Cuando se detecta la persona se pone en marcha durante 5s (duración del trayecto)
- Si durante el trayecto se detecta otra persona, se la debe llevar hasta el final también.
- Si en cualquier momento se presiona la seta de emergencia (con enclavamiento) el motor se debe detener. Al desactivarse la seta, el sistema deberá volver al estado de reposo.

Ejercicio: Control de una escalera mecánica

Grafcet de nivel 1

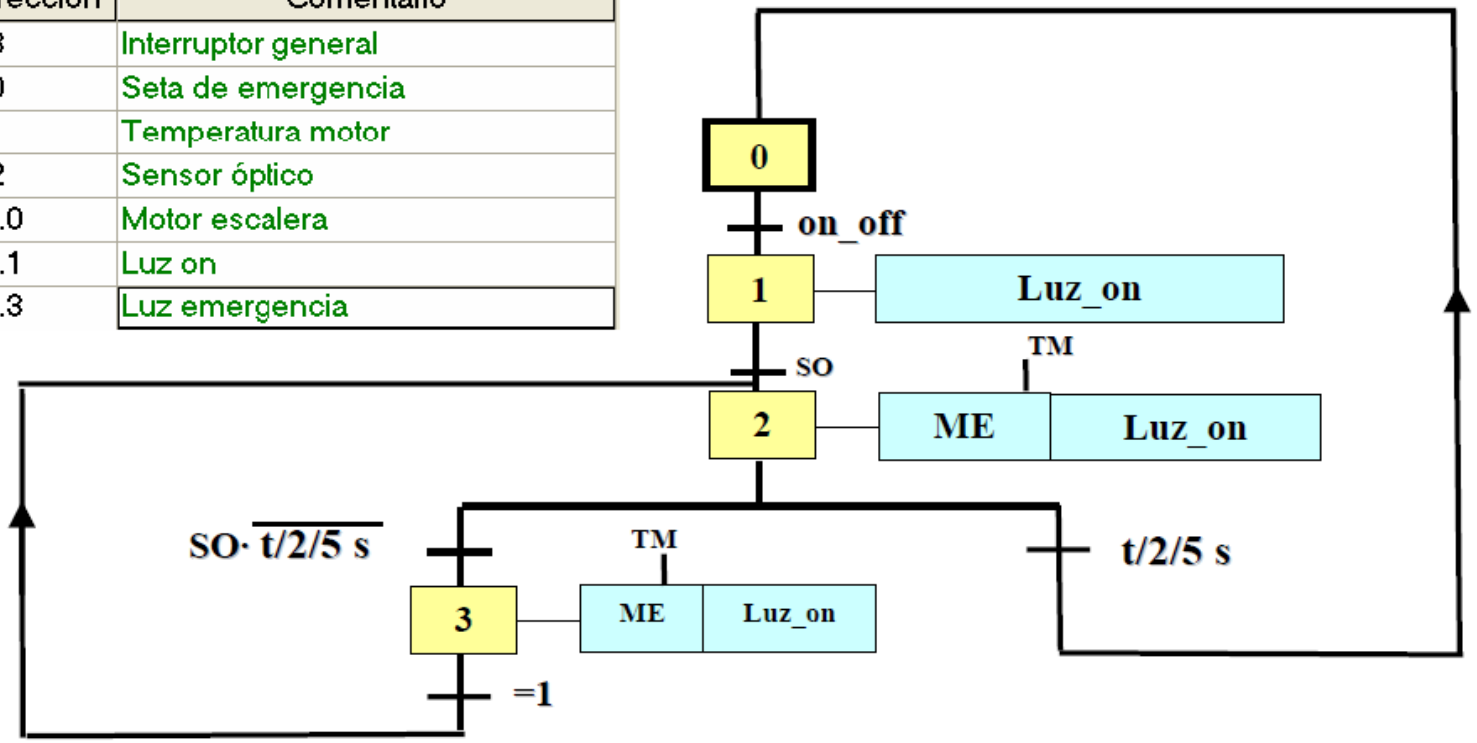


Nota: Etapa 3 necesaria para desactivar el temporizador y reiniciarlo

Ejercicio: Control de una escalera mecánica

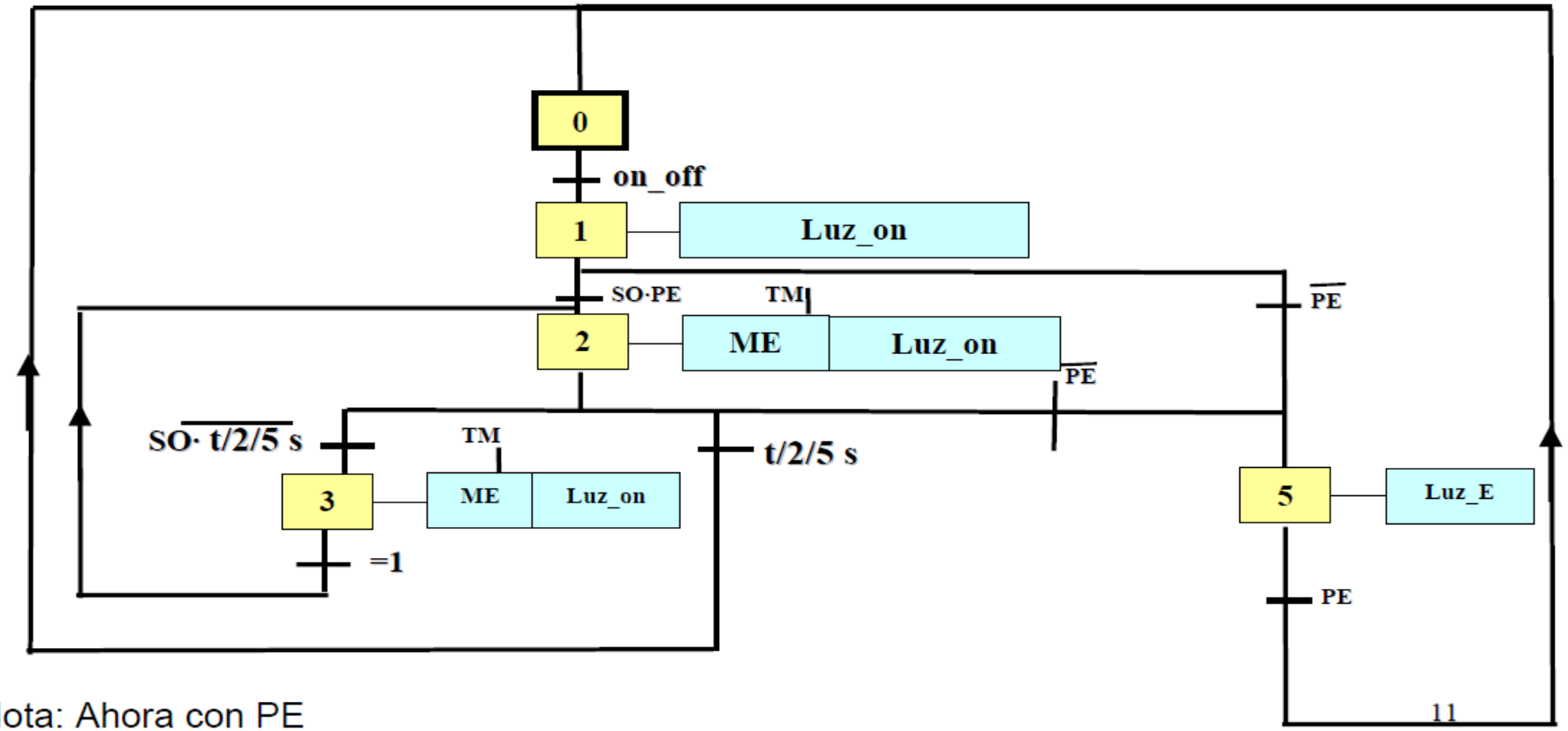
Grafcet de nivel 2

Símbolo	Dirección	Comentario
on_off	I0.3	Interruptor general
PE	I0.0	Seta de emergencia
TM	I0.1	Temperatura motor
SO	I0.2	Sensor óptico
ME	Q0.0	Motor escalera
Luz_on	Q0.1	Luz on
Luz_E	Q0.3	Luz emergencia



Ejercicio: Control de una escalera mecánica

Grafcet de nivel 2

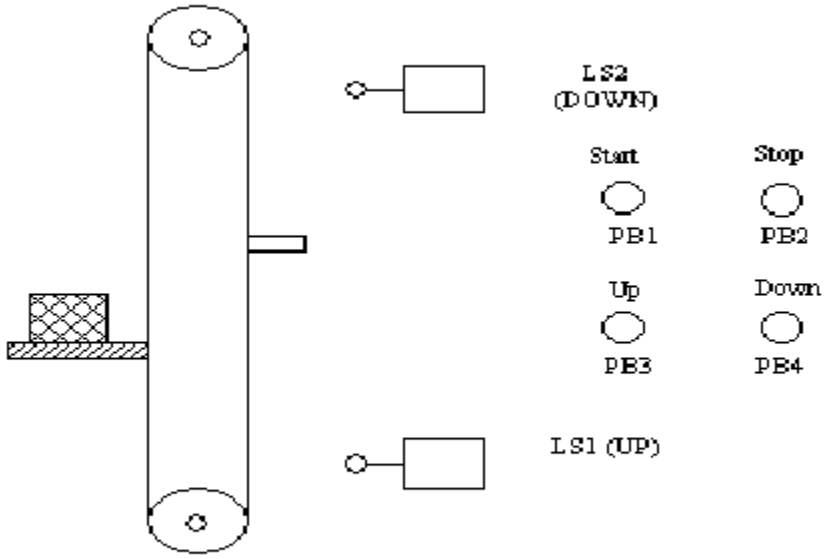


Nota: Ahora con PE

Elevador

El sistema elevador mostrado en la figura, emplea una plataforma para mover objetos DOWN (abajo) y UP (arriba). El objetivo es que cuando el botón UP (arriba) es pulsado la plataforma lleva algo a la posición DOWN (abajo).

M1 = Motor para hacer subir la plataforma.



M2 = Motor para hacer bajar la plataforma.

Los datos específicos de hardware siguientes definen el equipo usado en el elevador.

Elementos de salida.

M1 = Motor para hacer subir la plataforma.

M2 = Motor para hacer bajar la plataforma.

Elementos de Entrada.

LS1= Limit Switch NC (Normalmente Cerrado) para indicar la posición UP (arriba).
LS2= Limit Switch NC (Normalmente Cerrado) para indicar la posición DOWN (abajo).
START= push-button NA (Normalmente Abierto) para iniciar.
STOP= push-button NA (Normalmente Abierto) para detener.
UP = push-button NA (Normalmente Abierto) para orden arriba.
DOWN= push-button NA (Normalmente Abierto) para orden abajo.

La descripción narrativa siguiente indica la secuencia requerida de acontecimientos para el sistema de elevador.

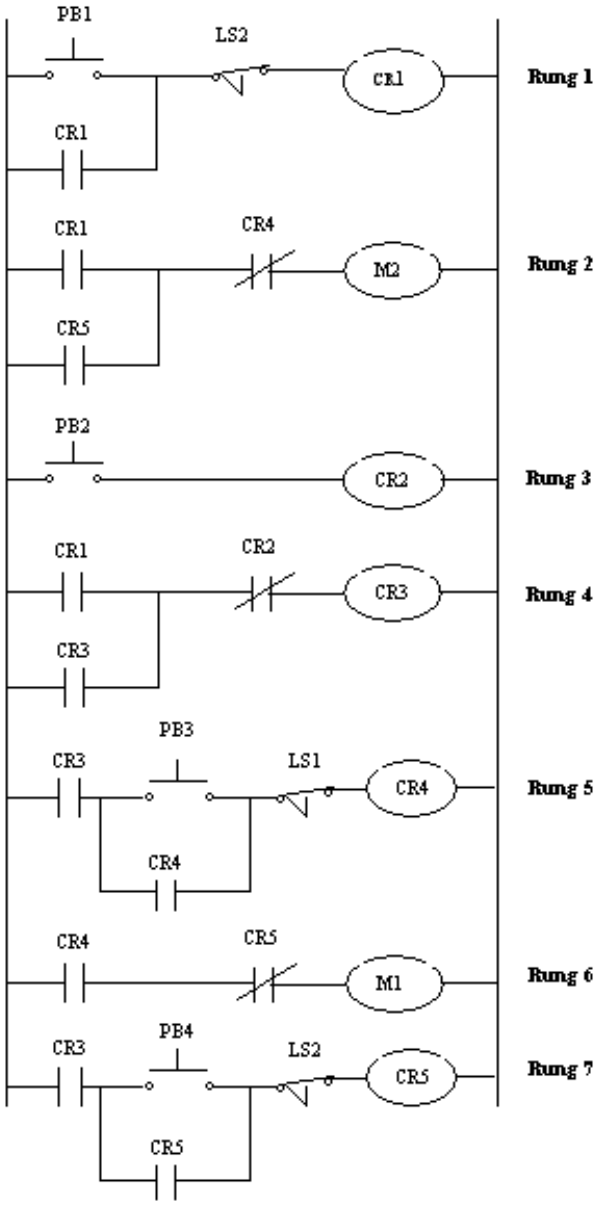
• Cuando el botón START es pulsado, la plataforma es conducida para la posición arriba.

• Cuando el botón STOP es pulsado, la plataforma es parada en cualquier posición que esta ocupa en ese tiempo o momento.

• Cuando el boton UP es pulsado, la plataforma, si no esta moviéndose hacia abajo (DOWN), es conducido a la posición UP (arriba).

• Cuando el botón DOWN es pulsado, la plataforma, si no esta moviéndose hacia arriba, es conducida a la posición DOWN (abajo).

Diagrama de Escalera completo para el Elevador.



Prepare el diagrama de escalera programado para el problema de control mostrado en la figura 8.31. El objetivo global es calentar un líquido a una temperatura especificada y mantenerlo allí durante 30 minutos.

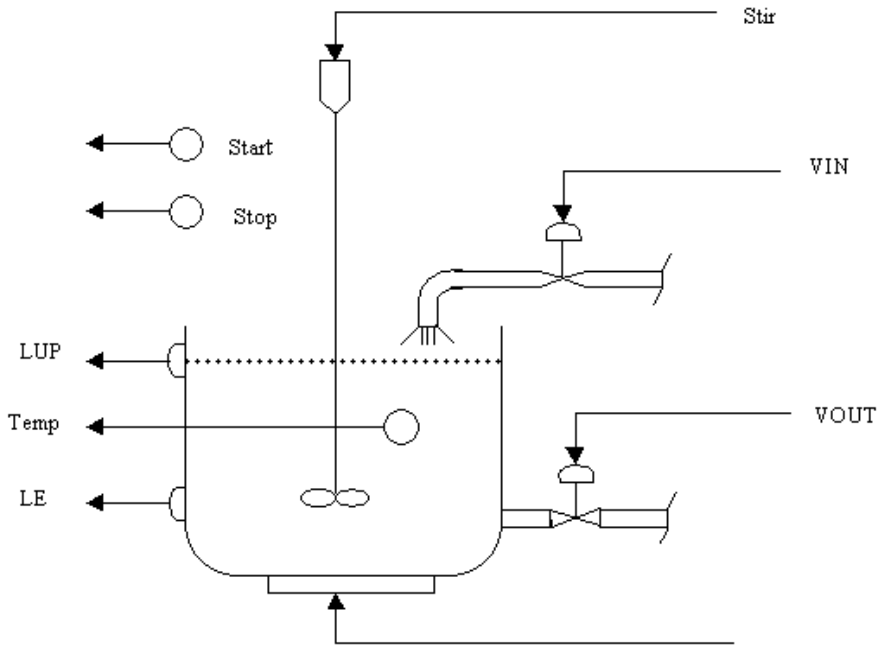


Figura 8.31

El botón START es normalmente abierto (NO), STOP es normalmente cerrado (NC).

NO y NC están disponible para los limit switches.

La secuencia de acontecimientos es:

1. Llene el tanque.

2. Caliéntese y mueva el líquido a la temperatura setpoint y asimiento durante 30 minutos.

3. Vacíe el tanque.

4. Repita del paso 1.

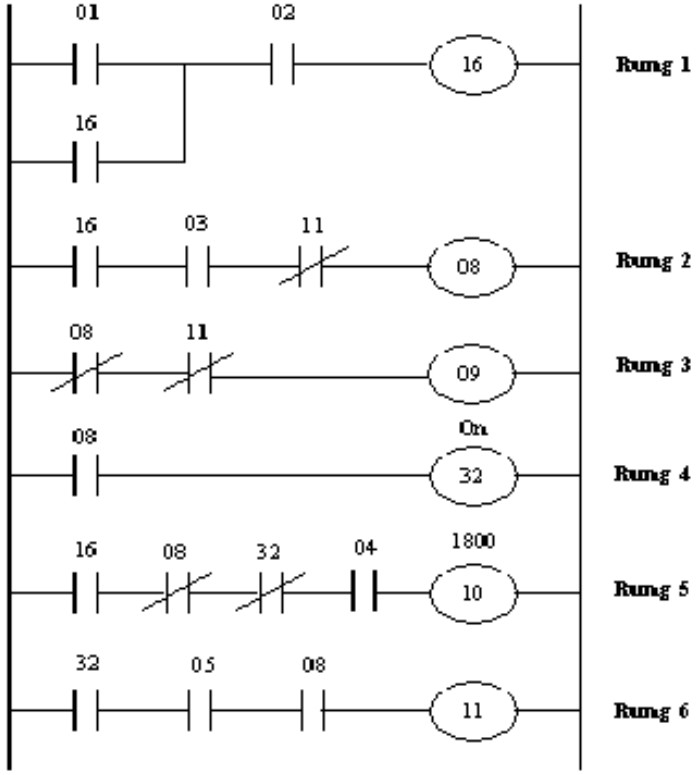
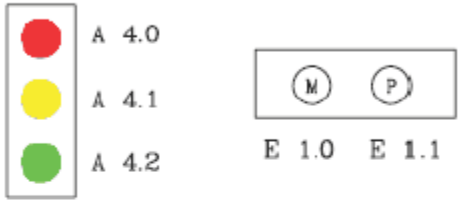


DIAGRAMA DE ESCALERA PROGRAMADO

Semáforo

Tenemos un semáforo con las tres luces verde, amarillo y rojo. Tenemos dos pulsadores de mando: un pulsador de marcha y un pulsador de paro.



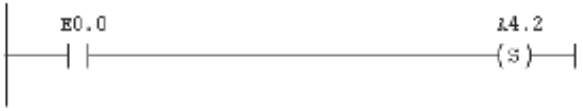
Con el pulsador de marcha quiero que comience el ciclo. El ciclo de funcionamiento es el siguiente:

1º/ Verde durante 5 seg.
2º/ Verde + Amarillo durante 2 seg.
3º/ Rojo durante 6 seg.

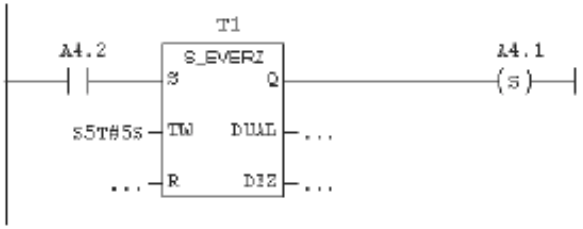
El ciclo es repetitivo hasta que se pulse el pulsador de paro. En ese momento se apaga todo. Siempre que le dé al pulsador de marcha quiero que empiece por el verde.

OB1 : SEMAFORO

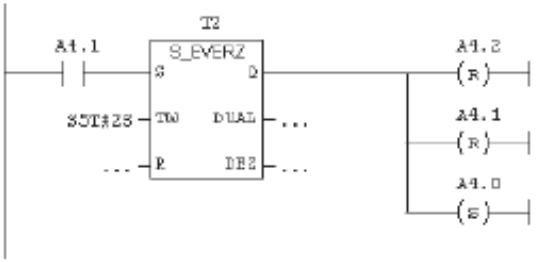
Segm. 1 : ENCENDER EL VERDE



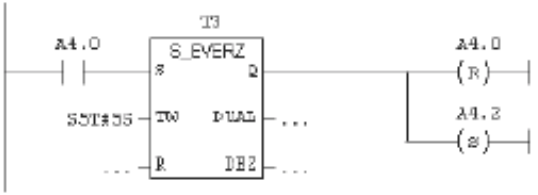
Segm. 2 : A LOS 5 SEGUNDOS ENCENDER EL AMARILLO



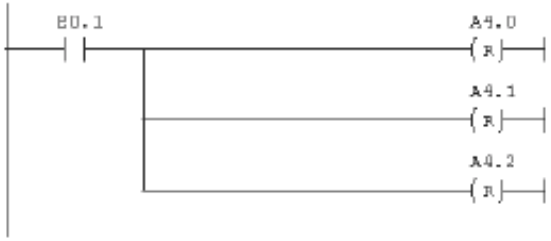
Segm. 3 : A LOS 2 S. APAGAR AMARILLO Y VERDE Y ENCENDER ROJO

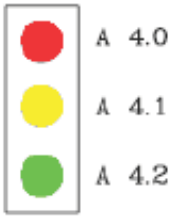


Segm. 4 : A LOS 5 SEG. APAGAR ROJO Y ENCENDER VERDE

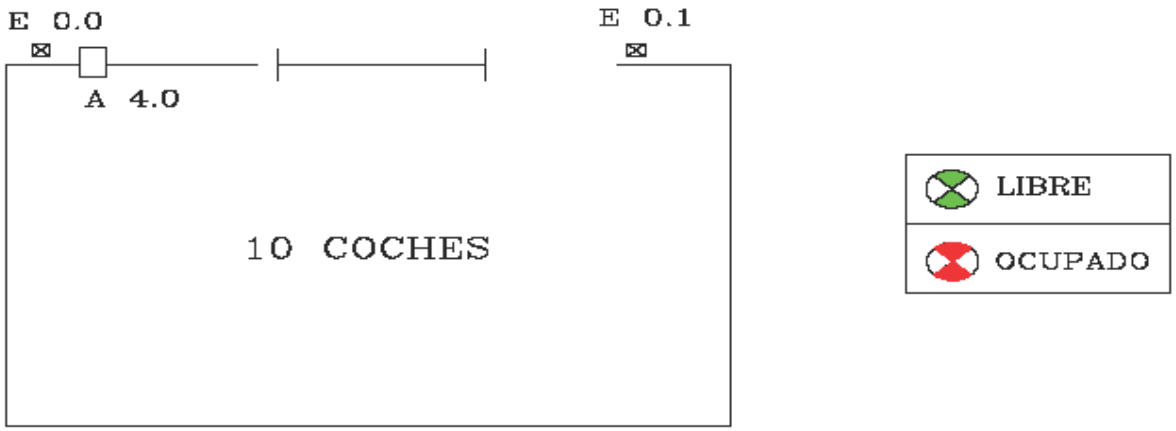


Segm. 5 : SI SE PULSA PARA APAGAR TODO





Tenemos el siguiente parking de coches:



El funcionamiento que queremos es el siguiente:

Cuando llega un coche y el parking esté libre, queremos que se abra la barrera. A la salida no tenemos barrera. Cuando sale un coche simplemente sabemos que ha salido.

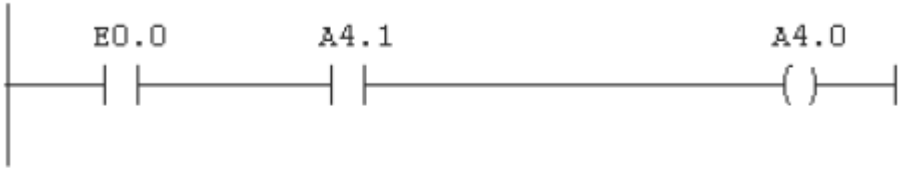
En el parking caben 10 coches. Cuando el parking tenga menos de 10 coches queremos que esté encendida la luz de libre. Cuando en el parking haya 10 coches queremos que esté encendida la luz de ocupado.

Además queremos que si el parking está ocupado y llega un coche que no se le abra la barrera.

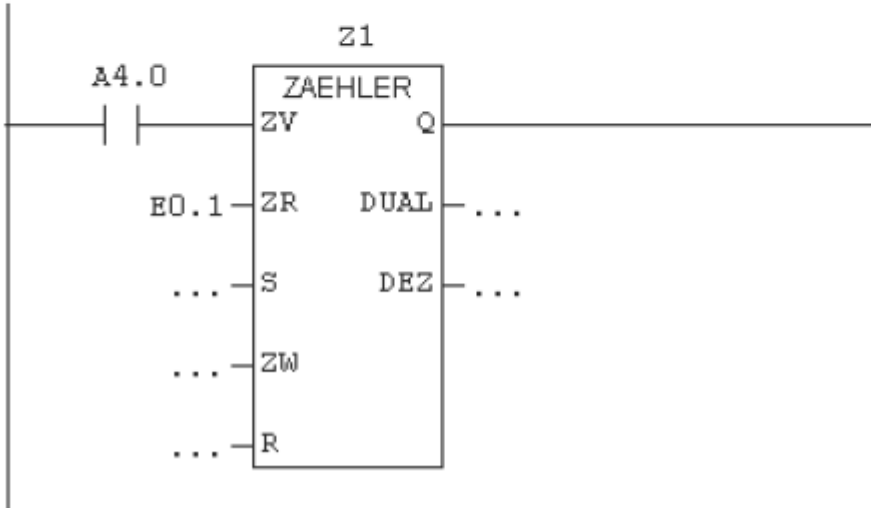
Diagrama de Escaleras

OB1 : PARKING

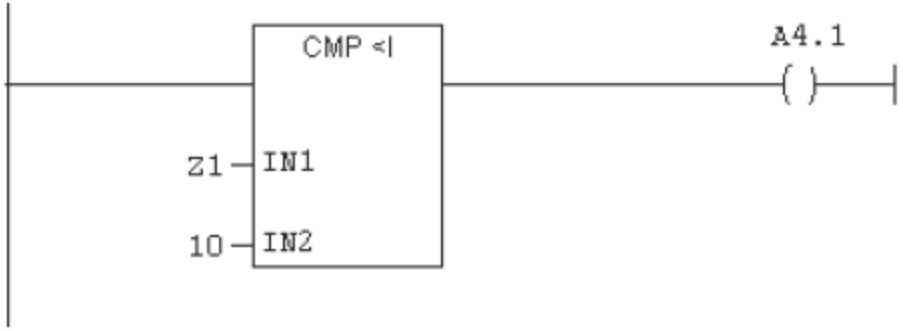
Segm. 1 : Si llega un coche le abro.



Segm. 2 : Contaje de coches.



Segm. 3 : Con menos de 10 coches está libre.

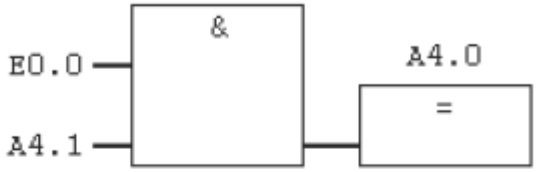


Segm. 4 : Si no está libre, está ocupado.

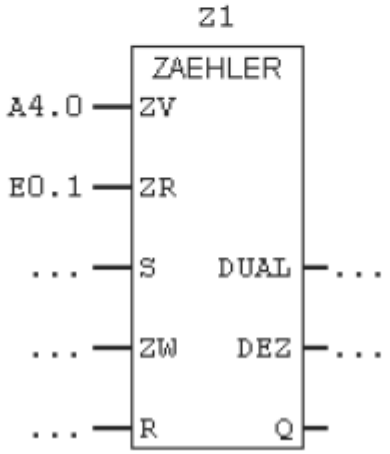
Diagrama de Bloques de Funciones

OB1 : PARKING

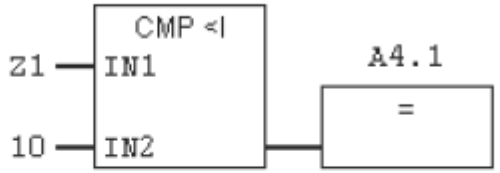
Segm. 1 : Si llega un coche le abro.



Segm. 2 : Contaje de coches.



Segm. 3 : Con menos de 10 coches está libre.



Segm. 4 : Si no está libre, está ocupado.