Dpto. de Sistemas Electrónicos y de Control

PLCopen
Standardization in Industrial Control Programming

T3. Lenguajes de Programación

Objetivos

1. Comprender la necesidad de estandarización del software de programación de PLC
2. Conocer la norma IEC 1131
3. Conocer los lenguajes más utilizados para la programación de PLC
4. Conocer la forma de ejecución de programas en un AP

PLCopen
Standardization in Industrial Control Programming

Introducción

Programación

Modos de funcionamiento

Lenguajes

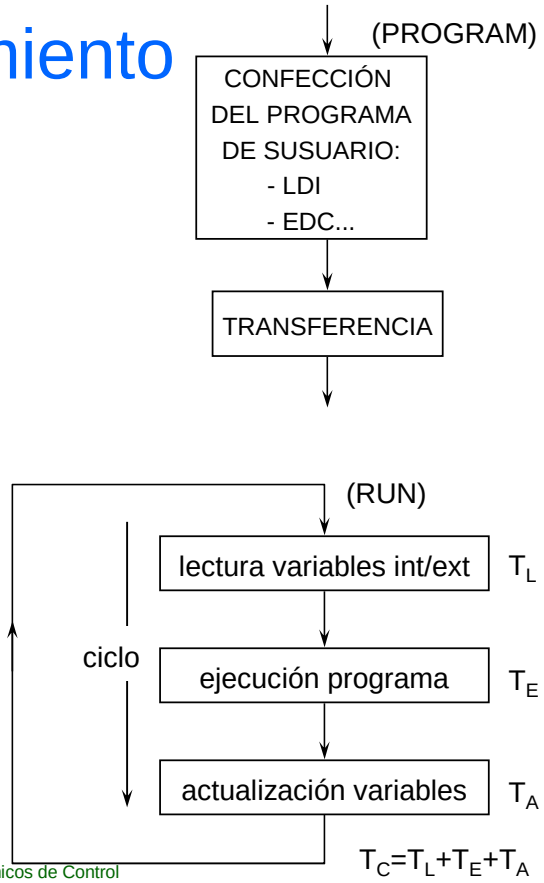
Programación

- Resolución del problema de automatización
- Confeccionar los programas
  - Programas = conjunto de instrucciones elementales
  - Instrucciones = conjunto de operandos y operadores
  - Operadores: particulares de cada lenguaje
  - Operandos: siempre los mismos:
    - entradas y salidas digitales y analógicas
    - Módulos o estructuras complejas de datos: contadores, temporizadores, desplazadores,...
    - Memoria, marcas o TAGS

- forma de ejecución de los programas
  - Cíclica es la más habitual
  - Eventos cada vez que se cambie de estado en una línea
  - Periódica: por periodos definidos
  - Periódica y por eventos de forma simultánea

Modos de funcionamiento

- PROGRAM
  - Permite programar y transferir el programa de usuario desde el sistema de programación hasta el AP
- RUN:
  - Permite al AP controlar el proceso
- Ejecución cíclica
  $T_{proceso} > T_{ciclo}$
- Ejecución periódica
  $T_{proceso} > T_{periodo}$



Lenguajes

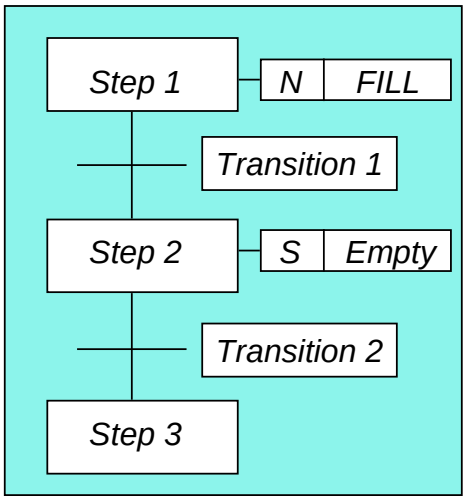
LI

```
LD    A
ANDN  B
ST    C
```

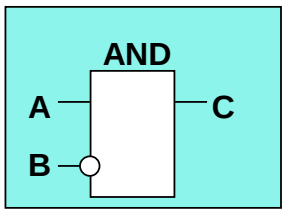
ST

```
C:= A  AND  NOT B
```

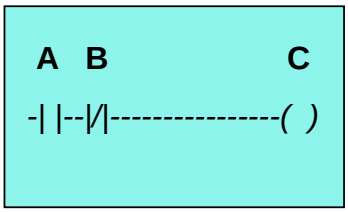
SFC



FBD



LD



- Todos ellos tienen la finalidad de generar el código objeto para que sea ejecutado en la CPU del PLC

PLCopen
Standardization in Industrial Control Programming

Norma IEC 1131

Estandarización

Norma IEC 1131-3

¿Estandarización?

Problema planteado a los ingenieros de Sw

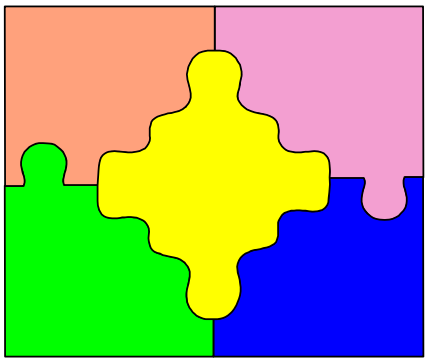
* Cómo resolver la automatización de una industria
* Trabajando con diferentes PLC de fabricantes distintos
* Utilizando diferentes lenguajes de programación
* Y que sea comprendido por ingenieros eléctricos o personal de mantenimiento de planta
- Y que el fabricante del producto sea mejor que su más directo competidor

SOLUCIÓN NORMA IEC1131

Las 5 partes de la norma IEC 1131

- 1 Visión general, definiciones IS
- 2 Hardware IS
- 3 Lenguajes de programación IS
- 4 Manuales de usuarios
- 5 Especificaciones

IS = International Standard

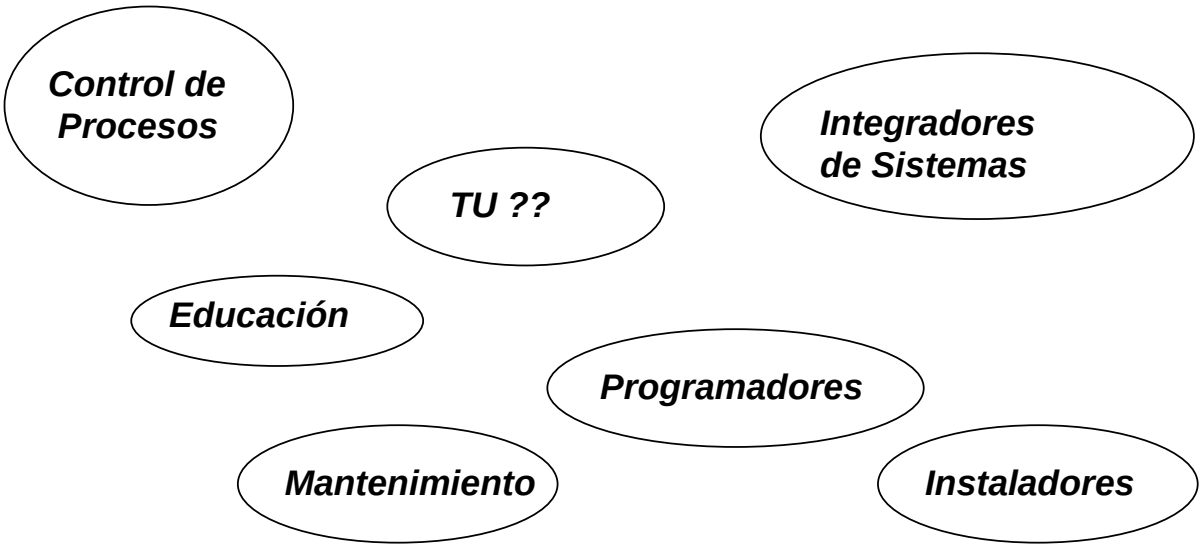


Lenguajes de Programación Vs Programación de Control Industrial

El interface entre el programador y el Sistema de control ...

...con soporte para personal de diferente cualificación profesional

Usuarios de la norma



Ventajas de la norma IEC 1131

- Reduce el esfuerzo humano en entrenamiento, depuración, mantenimiento y consultoría
  - Una vez que se aprende se puede utilzar en todos los sistemas
- Posibilidad de crear Sw reutilizable, minimiza
  - el tiempo de desarrollo
  - el esfuerzo de codificación
  - los errores de compilación y ejecución
- Técnicas de programación usados en otros entornos no industriales
- Coordina eficazmente diferentes componentes desde distintas localizaciones, compañías o proyectos
  - Amplio campo de aplicación
- Aumenta la conectividad facilita la distribución del control

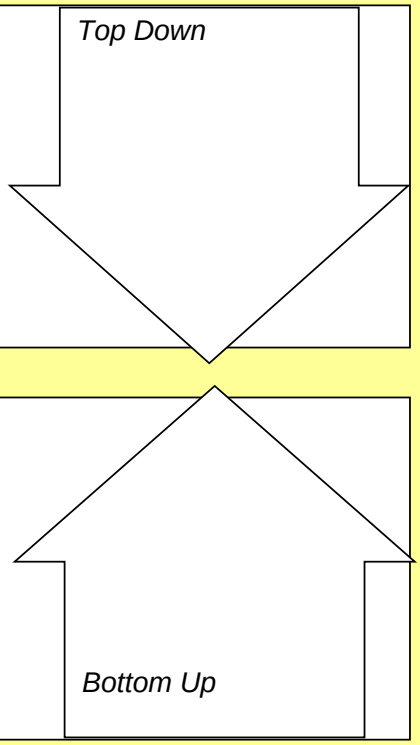
Claves de éxito de la norma IEC 1131-3

- Sw estructurado a través de Diseño, Proyectos, Tareas, Programas y Bloques
  - Unidades de Organización de Programas (Program Organization Units (POUs))
- Tipado fuerte de datos a través de lenguajes que poseen operaciones a las que sólo se le puede aplicar un tipo apropiado de datos
- Control de la ejecución a través de tareas
- Descripción del Comportamiento secuencial complejo de un proceso a través de SFC
- Encapsulación del Sw a través de POUs, estructuras y tipos complejos de datos

Norma IEC 1131-3

Elementos Comunes

- La norma permite dos caminos de desarrollo de un programa
  - Hacia abajo: configuración de los datos pensando en el proceso y después elegir el Sw de programación
  - Hacia arriba: seleccionando un Sw de programación más adecuado al proceso y posteriormente definir los tipos de datos



Lenguajes de Programación

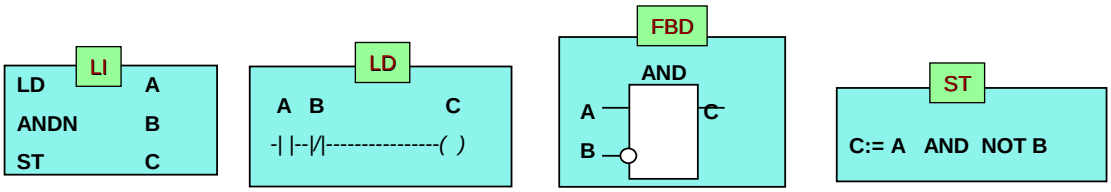
IEC 1131-3

Elementos Comunes

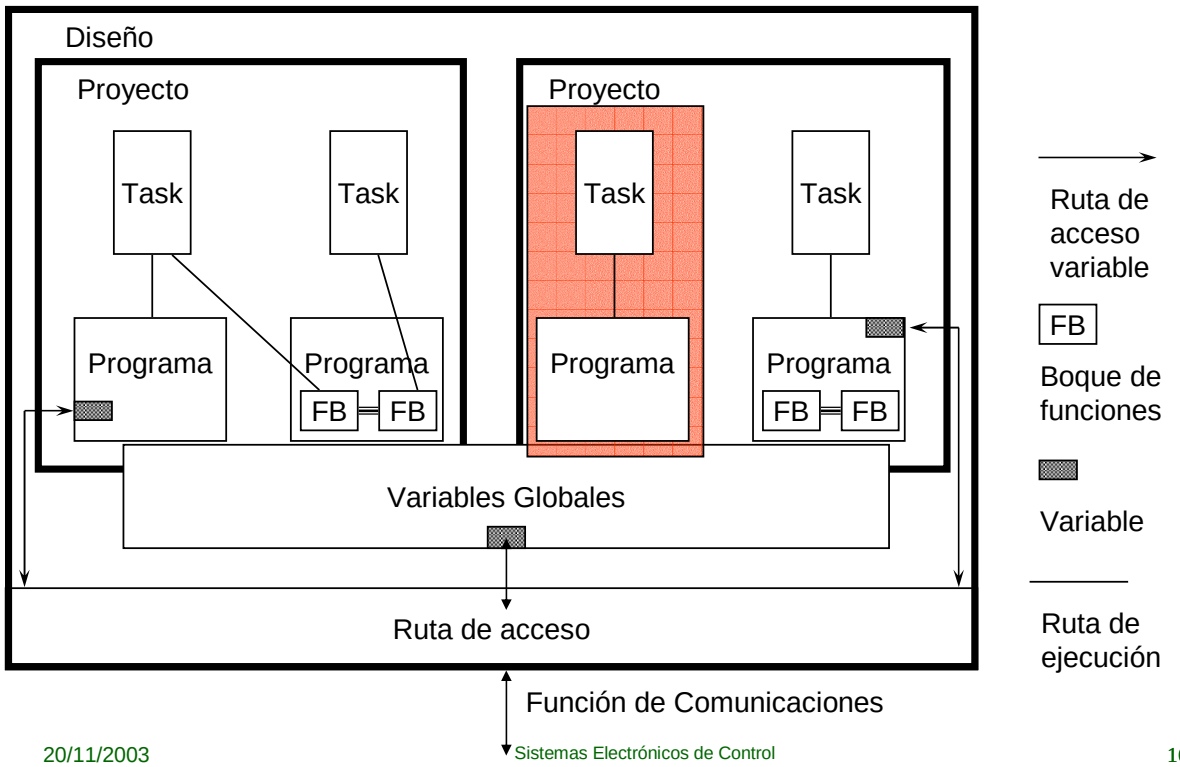
- **Variables, tipos de datos y declaraciones**
- **Diseño, proyectos y tareas**
- **Funciones, bloques de funciones y programas**
- ***Sequential Function Charts***

Diseño
Proyectos
Tareas
Variables Globales
Caminos de Acceso

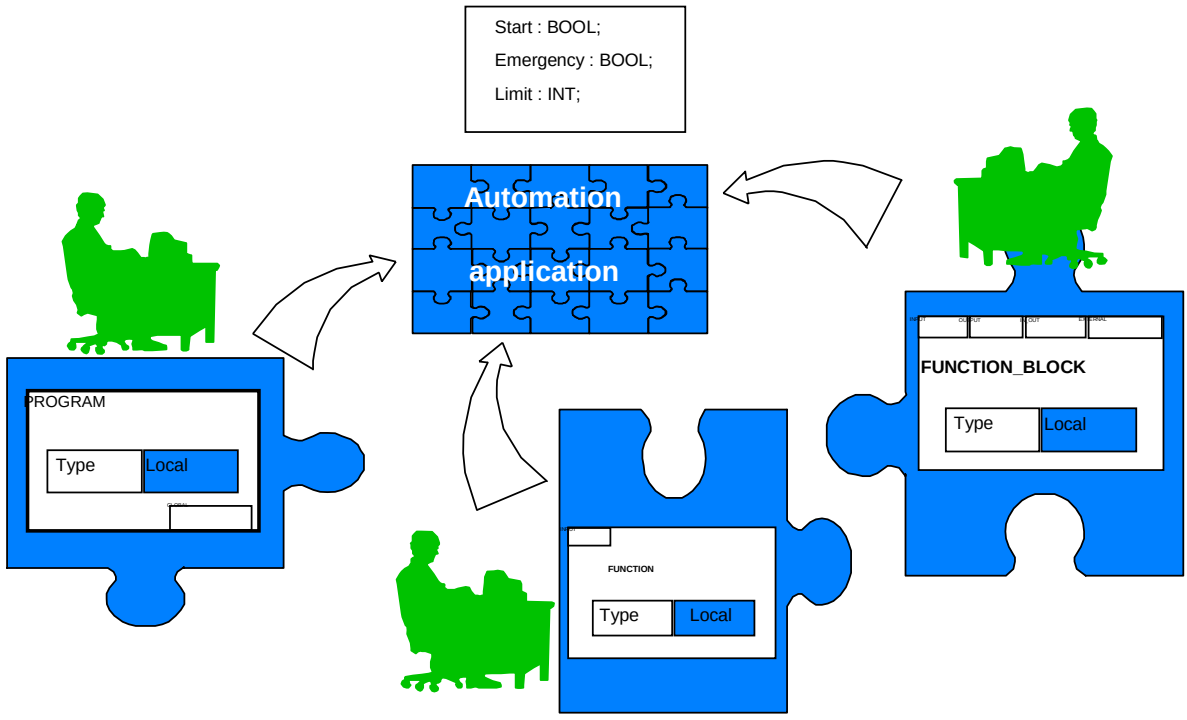
Lenguajes de Programación



IEC 1131-3 vs convencional PLC



Programas: diseño Jerárquico



PLCopen
Standardization in Industrial Control Programming

Lista de Instrucciones
Instruction List (IL)

Listado de Instrucciones

LI

```
LD    A
ANDN  B
ST    C
```

- Es un tipo de lenguaje ensamblador con un repertorio muy reducido de instrucciones
- Los programas utilizan un estilo muy similar al empleado por los lenguajes de ensamblador
- Este tipo de lenguaje es una trascripción elemental e inmediata de las instrucciones del lenguaje máquina
  - que están representadas por expresiones nemotécnicas
- Se suele aplicar para pequeñas aplicaciones y para optimizar partes de una aplicación

Semántica y Operadores

LD	Set current result equal to operand
ST	Store current result to operand location
S	Set Boolean operand to 1
R	Reset Boolean operand to 0

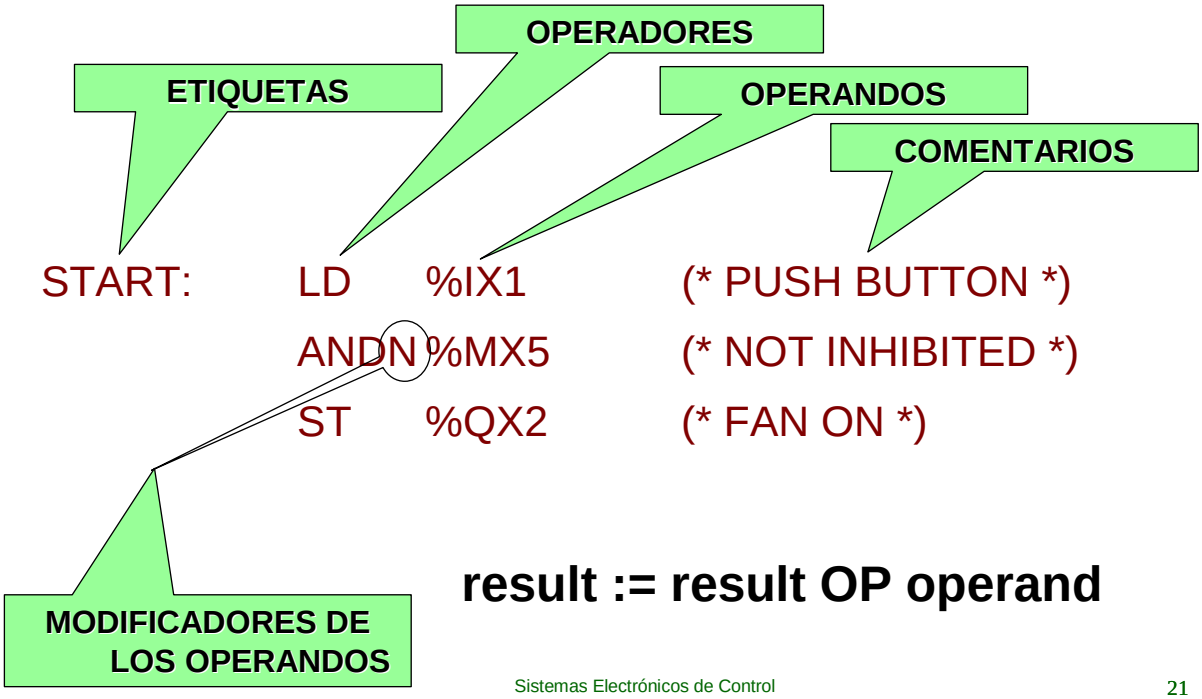
ADD	Addition
SUB	Subtraction
MUL	Multiplication
DIV	Division

JMP	Jump to label
CAL	Call function block
RET	Return from called function or function block

GT	Comparison: >
GE	Comparison: >=
EQ	Comparison: =
NE	Comparison: <>
LE	Comparison: <=
LT	Comparison: <

&, AND	Boolean AND
OR	Boolean OR
XOR	Boolean xclusive OR

Ejemplos de instrucciones



result := result OP operand

PLCopen
Standardization in Industrial Control Programming

Texto Estructurado
Structured Text (ST)

Texto Estructurado

ST

```
C:= A   AND  NOT B
```

- Los lenguajes basados en *texto estructurado* facilitan la programación de procesos que requieren instrucciones complejas y cálculos muy grandes
- Se trata de lenguajes de alto nivel

Operadores

Symbol	Operation
(expression)	Parenthesization
identifier(argument list)	Function evaluation
Examples:	
LN(A), MAX(X,Y), etc.	
**	Exponentiation
-	Negation
NOT	Complement

Operadores

*	Multiply
/	Divide
MOD	Modulo
+	Add
-	Subtract
< , > , <= , >=	Comparison
=	Equality
<>	Inequality

&, AND	Boolean AND
OR	Boolean OR
XOR	Boolean xclusive OR

A := B; asignación

Datatype to Datatype

INT to INT

or

Analog_Channel_Configuration to Analog_Channel_Configuration

CV := CV+1;

C := SIN(X);

IF .. THEN .. ELSE

CASE

FOR

WHILE …

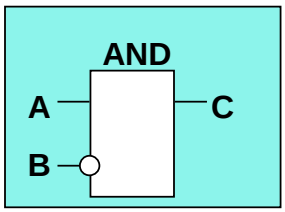
REPEAT UNTIL

PLCopen
Standardization in Industrial Control Programming

Esquema Básico de Funciones
Function Block Diagram (FBD)

Esquema Básico de Funciones

FBD



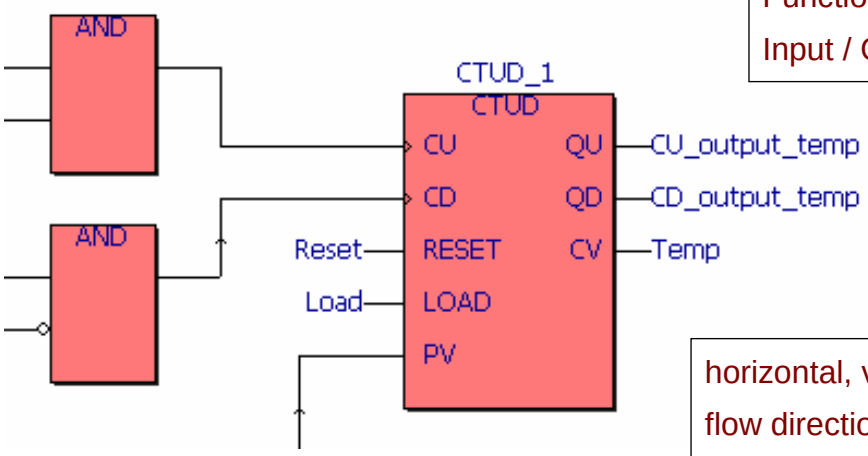
- El diagrama de funciones (también conocido como esquema básico de funciones EBF o function block diagram FBD) es un **lenguaje gráfico**
- Los programas son bloques cableados entre sí de forma análoga al esquema de un circuito
- Tiene una interface de E/S bien definida, y además poseen un código interno oculto

Ventajas

- Documentación y programación en un mismo elemento del programa
  - Informes generales, comentarios, flujo de datos...
- Aplicación universal, enteros, punto flotante...
- Programación estructurada
  - Definición y llamada a subrutinas
- Conjunto de funciones y de bloques estandarizados
  - Se pueden mezclar bloques de distintos fabricantes
  - Se pueden definir nuevos bloques
- Los FBs son altmaente reutilizables
  - En un mismo programa
  - En programas diferentes
  - En diferentes proyectos

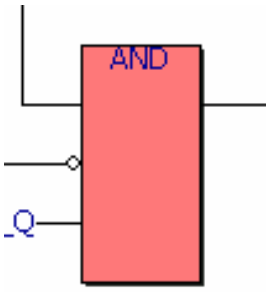
Elementos: Funciones, Bloques y Variables

- Functions
- Functions Blocks
- Input / Output Variables

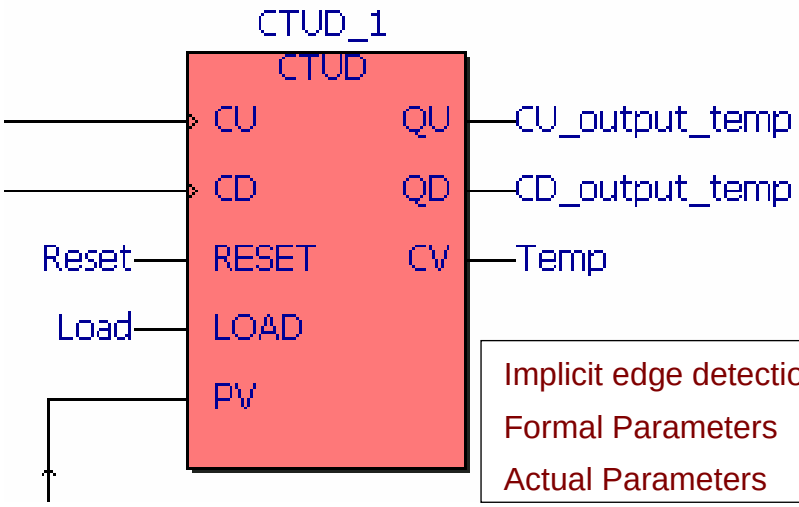


- horizontal, vertical
- flow direction
- crossing with / without contact

Elementos - Parametrización

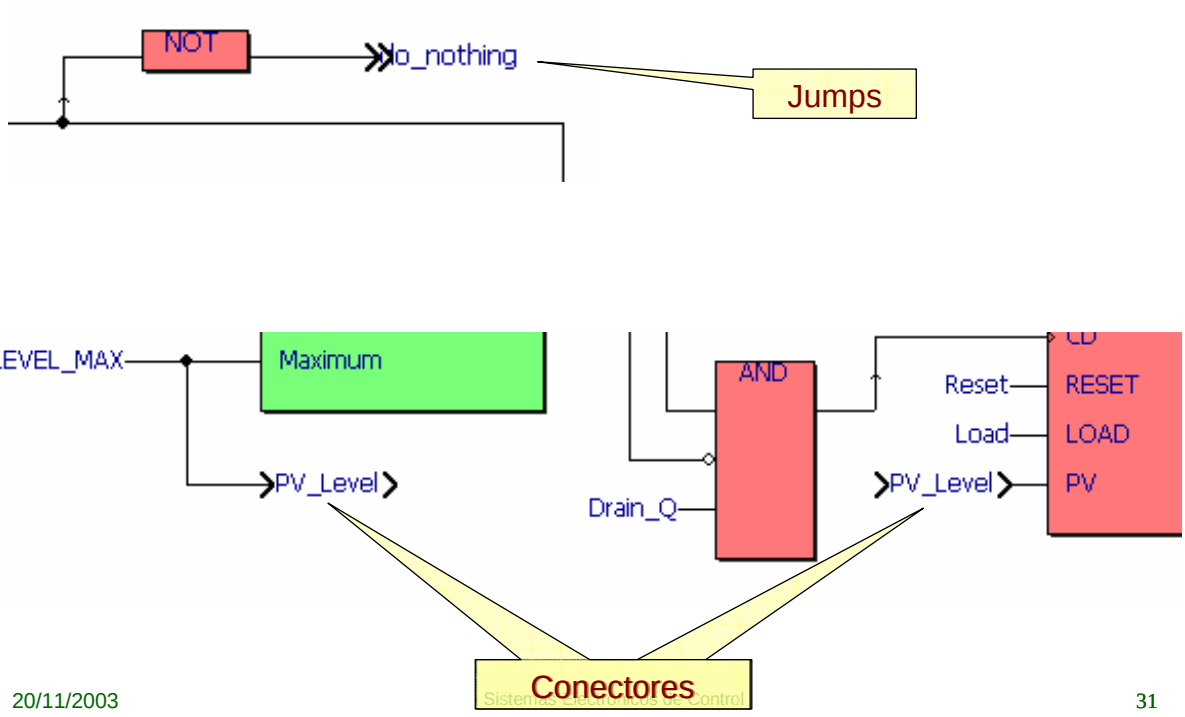


Negation
Duplication

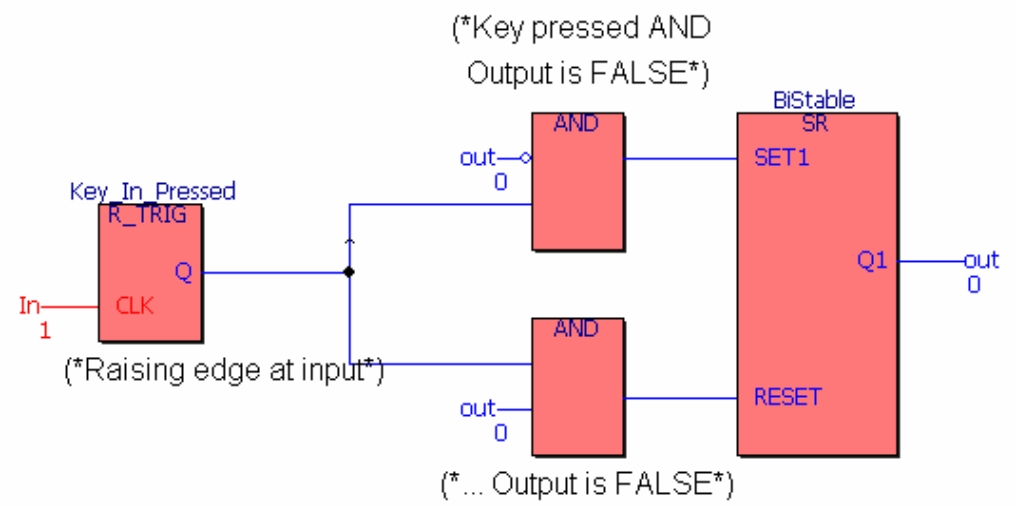


Implicit edge detection
Formal Parameters
Actual Parameters

Elementos adicionales

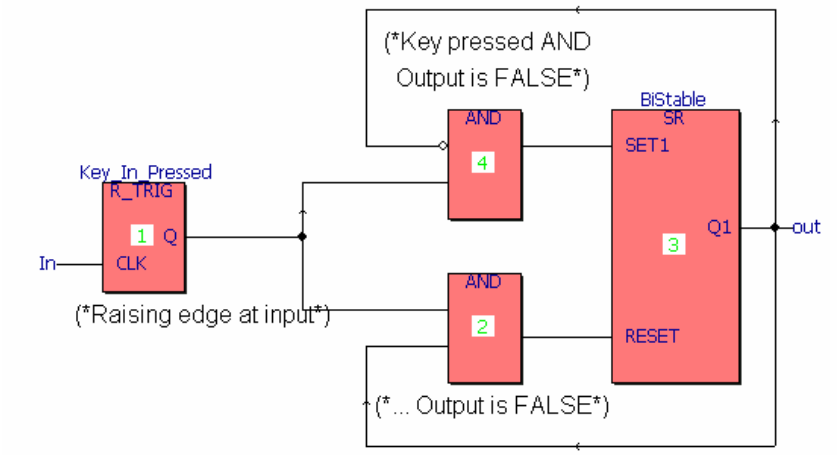


Reglas de Ejecución



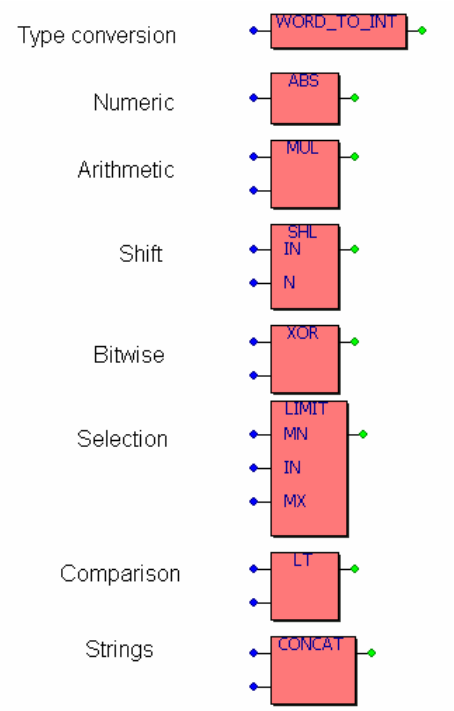
- 1. El bloque se ejecuta cuando todas sus entradas han sido evaludas
- 2. El bloque se evalúa por completo cuando se has calculado todas sus salidas
- 3. La evaluzación de un conjunto de bloques termina cuando se calculan todas y cada una de las salidas

Realimentación

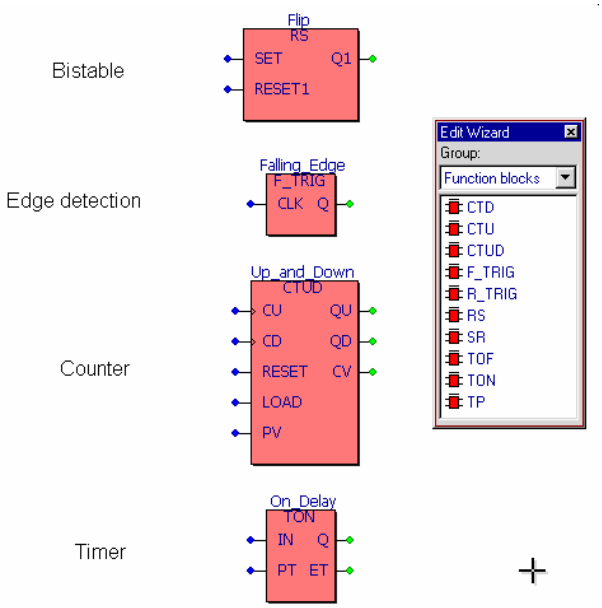


- No se puede valorar el orden de la ejecución
- Existen formas de resolverlo como la asignación de un orden de ejecución

Funciones Estándar



Bloques Estándar



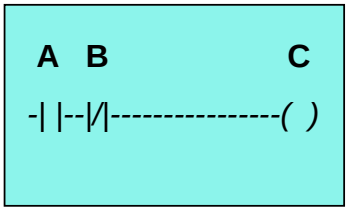
- Norma IEC 61131: “Si se conece el estándar, se conoce todo"

PLCopen
Standardization in Industrial Control Programming

Esquema de Contactos
Ladder Diagram (LD)

Ladder

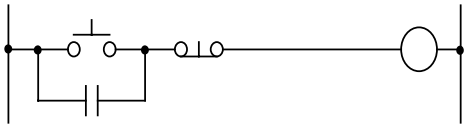
LD

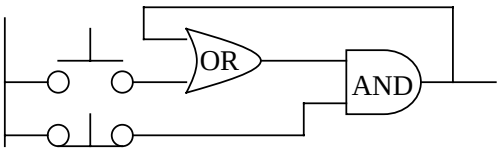


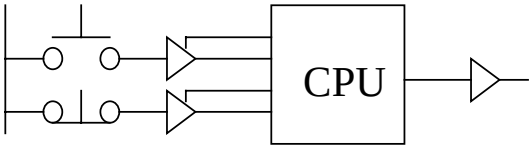
- La lógica de escalera o ladder es el lenguaje de programación más usado para la programación de PLCs
- Fue el primero con el que se comenzó a programar, de ahí que presente grandes semejanzas con los diagramas eléctricos de escalera utilizados por los técnicos anteriormente a la aparición del autómata
- Este lenguaje está especialmente indicado para facilitar el cambio de un sistema de control realizado con relés por un PLC

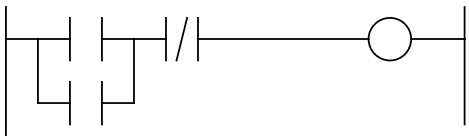
Origenes del LD

- 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



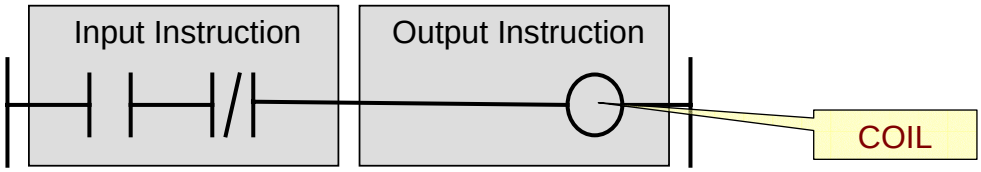






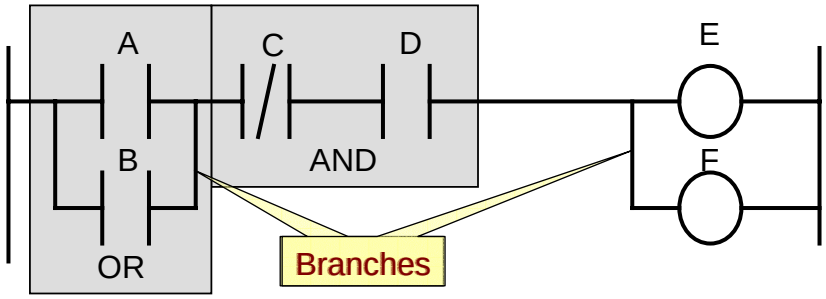
¿Qué es un *Rung*?

- 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*



Operaciones en Serie y Paralelo

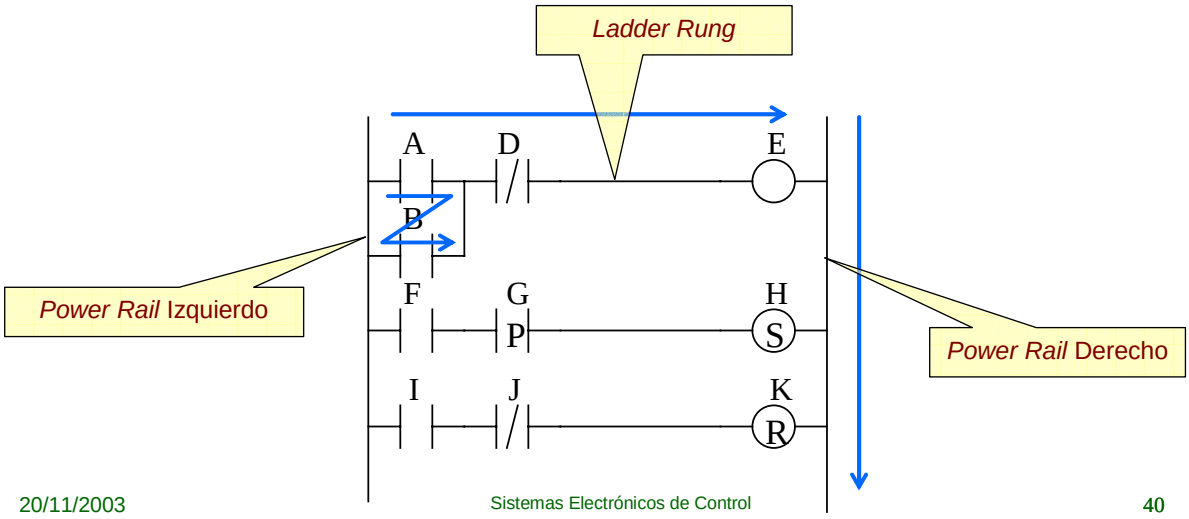
- Las instrucciones de entrada pueden ejecutarse mediante relaciones lógicas AND y OR en un sencillo formato
  - Si las instrucciones están en serie se evalua una relación AND
  - Si las instrucciones están en paralelo se evalua una relación OR
- Salidas en paralelo permite activar varias òperacoines 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

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



Contactos

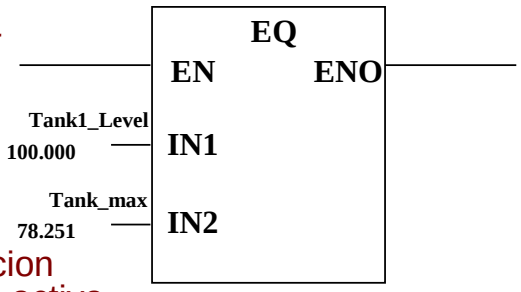
- Normalmente Abierto **--| |--**
  - Activa el *rung* hacia la derecha de la instrucción cuando el contacto se activa
- Normalmente Cerrado **--|/|--**
  - Activa el *rung* hacia la derecha de la instrucción cuando el contacto se desactiva
- Transición positiva **--|P|--**
  - Activa el *rung* hacia 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

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*

Instrucciones IEC de Comparación

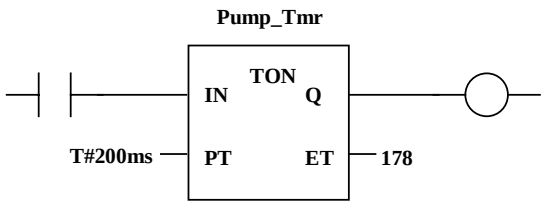
- Si el *rung* de entrada está activo (EN), la instrucción ejecuta la operación y activa el *rung* de salida (ENO) basado en la comparación
  - Ejemplo
    - Cuando EN es *true*, EQ (=) la funcion compara In1 y In2 y si son iguales activa ENO

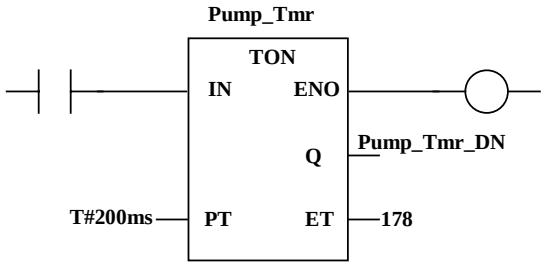


- Conjunto de instrucciones de comparación
  - EQ(=), GT (>), GE (>=), LT (<), LE (<=), NE (<>)

Instrucciones IEC de Temporización

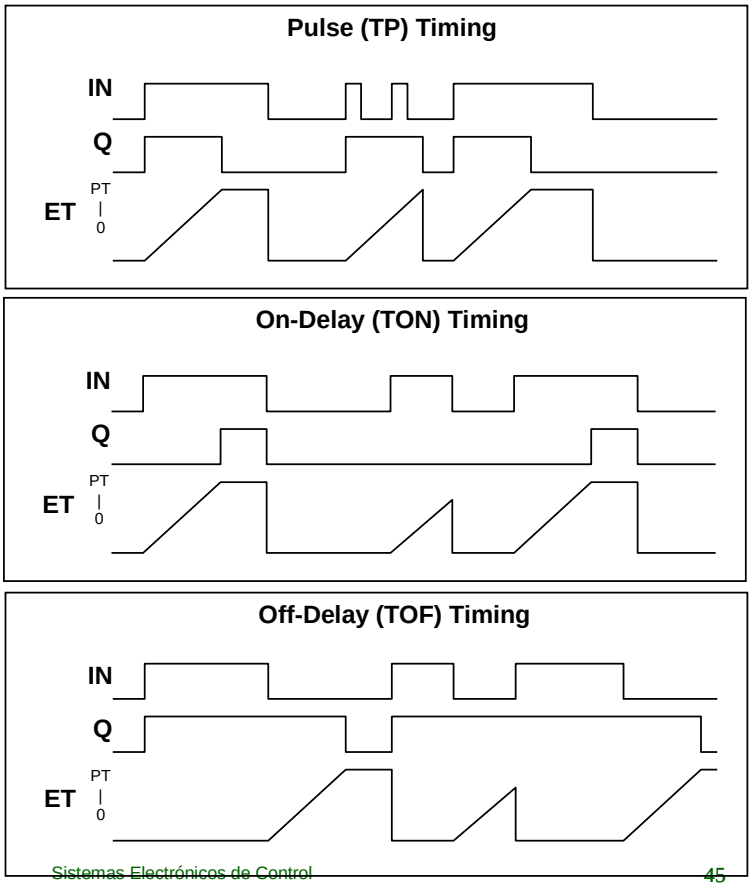
- Tres insctrucciones básicas
  - TP - *Pulse timer*
  - TON - *Timer On Delay*
  - TOF - *Timer Off Delay*
- Valores temporales enteros
  - Base de tiempos de 1msec
- Dos posibles formas de uso
  - 1ª necesita programación extra en otro *rung* para interaccionar sobre el estado del *timer*
  - 2ª activa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





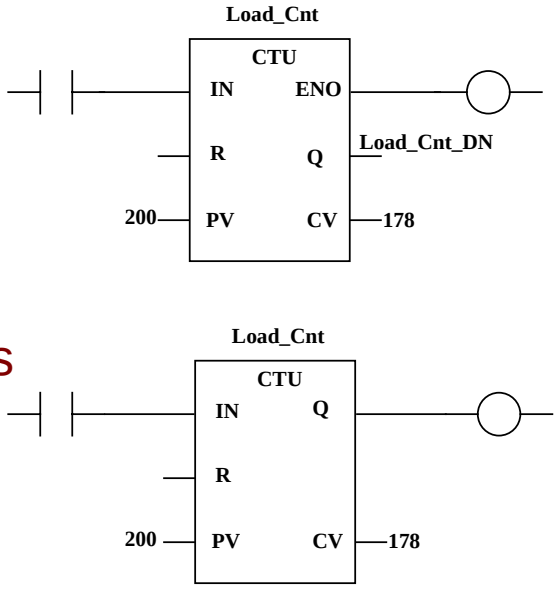
Temporizador

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



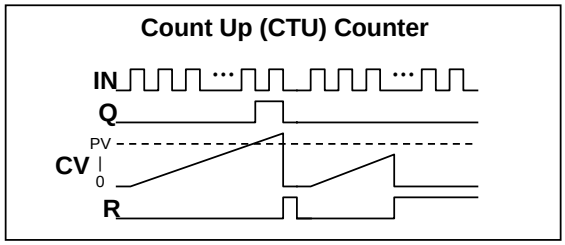
Instrucciones IEC de Contadores

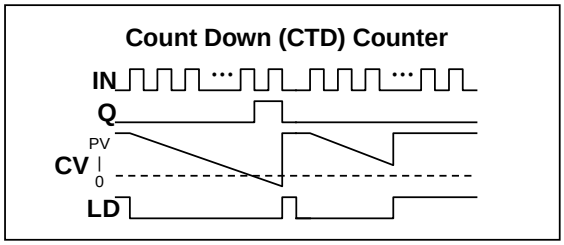
- Tres instrucciones básicas
  - CTU - *Count Up Counter*
  - CTD - *Count Down Counter*
  - CTUD - *Count Up/Down Counter*
- Todos cuentan transiciones
- Dos formas de uso, igual que los temporizadores

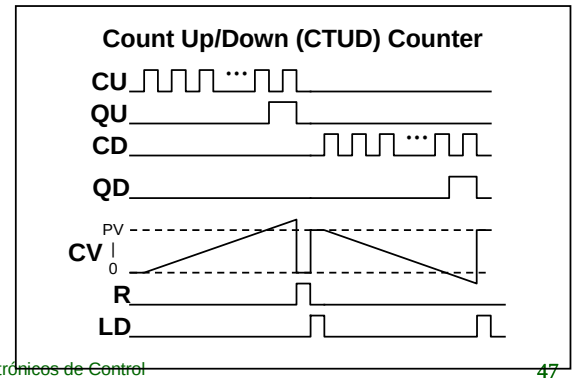


Contadores

- 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*

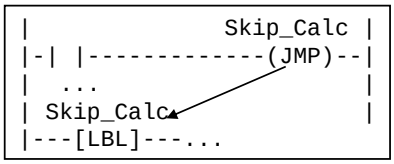


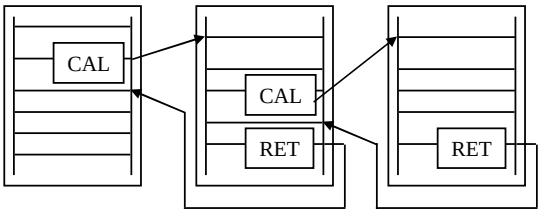




Ruptura de la secuencia de ejecución

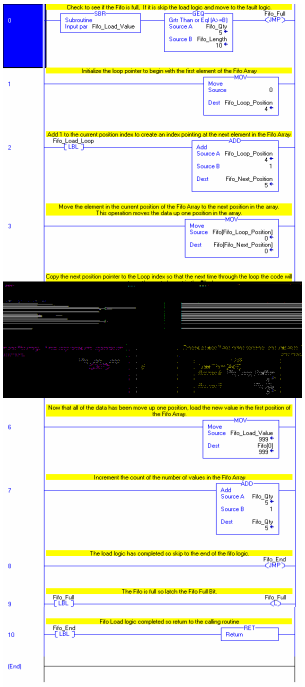
- Instrucciones de salto a etiquetas
  - Salta a un bloque de código del programa
  - LBL – nombre de la etiqueta para la operación de salto
  - JMP – ejecución de un salto cuando se activa la instrucción de entrada
- Instrucciones de salto a subrutinas
  - Salta a un bloque de código encapsulado como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



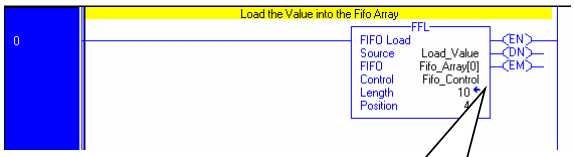


Extensiones de IEC optimiza el código y facilita su uso

IEC1131-3 Load FIFO Logic



Rockwell Automation FIFO Load Instruction



=

11 Rungs of Logic
17 Instructions
Hours to code and debug

1 Rung of Logic
1 Instruction
Minutes to code and debug

PLCopen
Standardization in Industrial Control Programming

Esquema Secuencial de Funciones
Sequential Function Chart (SFC)

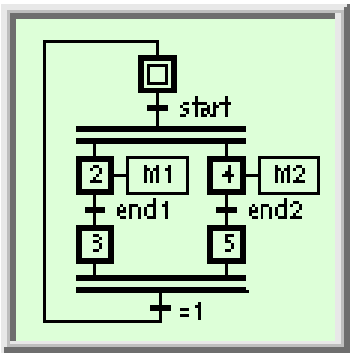
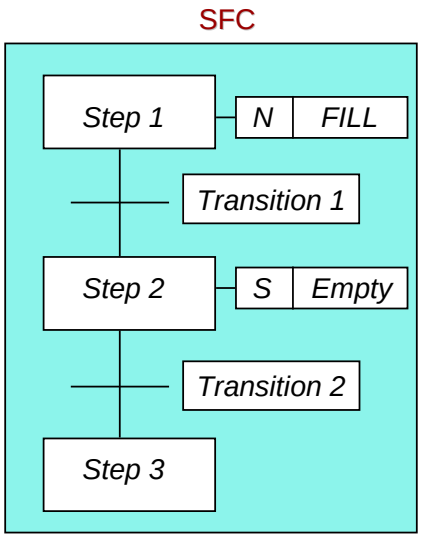


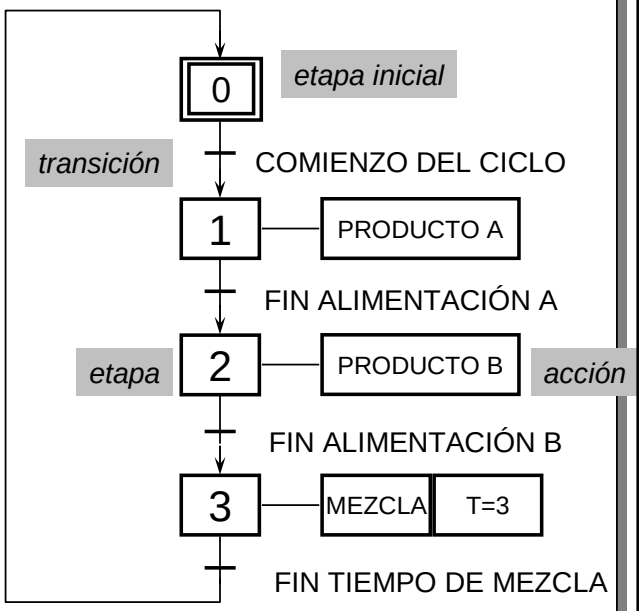
Diagrama Funcional Secuencial

- En sus orígenes fue GRAFCET (GRAFico Funcional de Control Etapa Transición)
  - surge a mediados de los 70, TELEMECANIQUE, APER, AFCET, ADEPA.
- Eficaz técnica para describir el comportamiento secuencial de un proceso y de un programa
- Se usa para distribuir un problema de control
- Permite un rápido diagnóstico



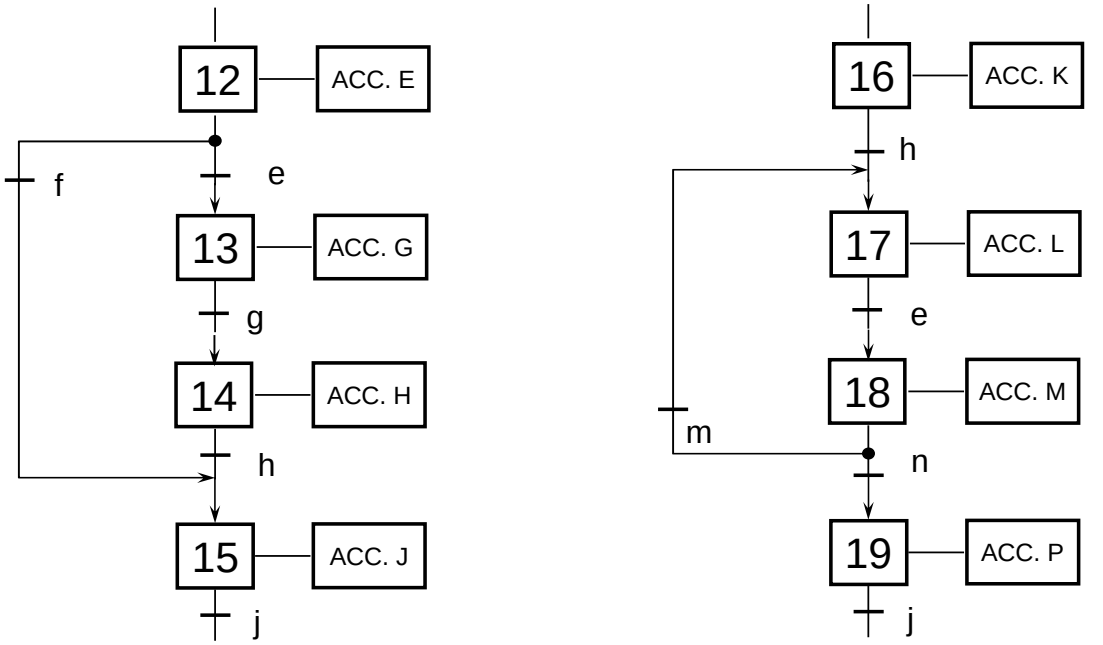
Grafcet

- 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
- The basic elements are STEPS with ACTION BLOCKS and TRANSITIONS
- Support for alternative and parallel sequences



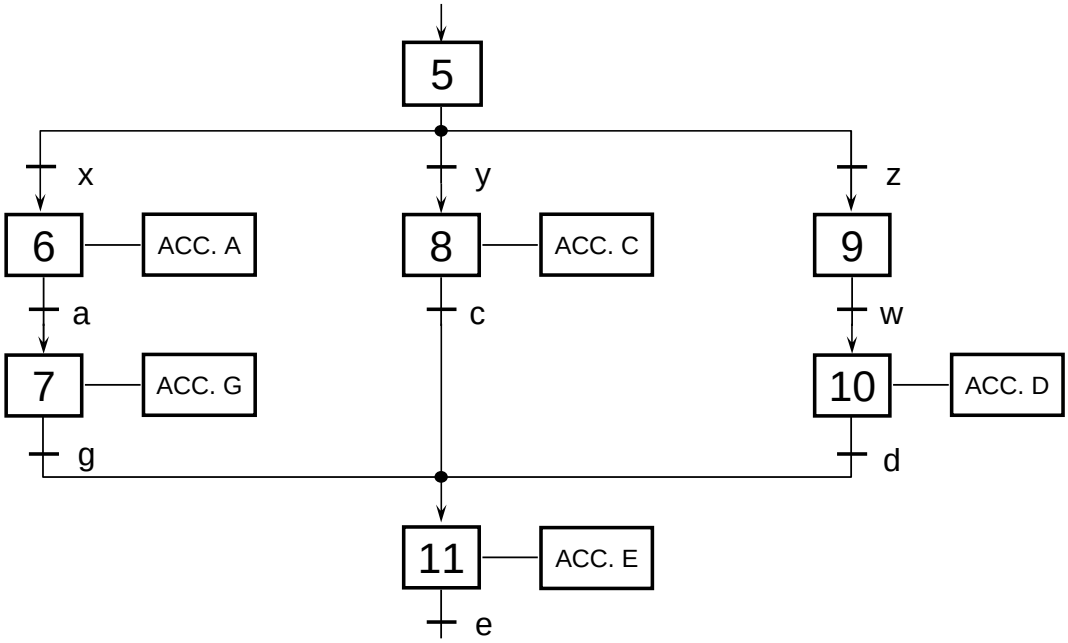
- SALTO CONDICIONAL DE ETAPA:

Direccionamiento específico hacia atrás y adelante



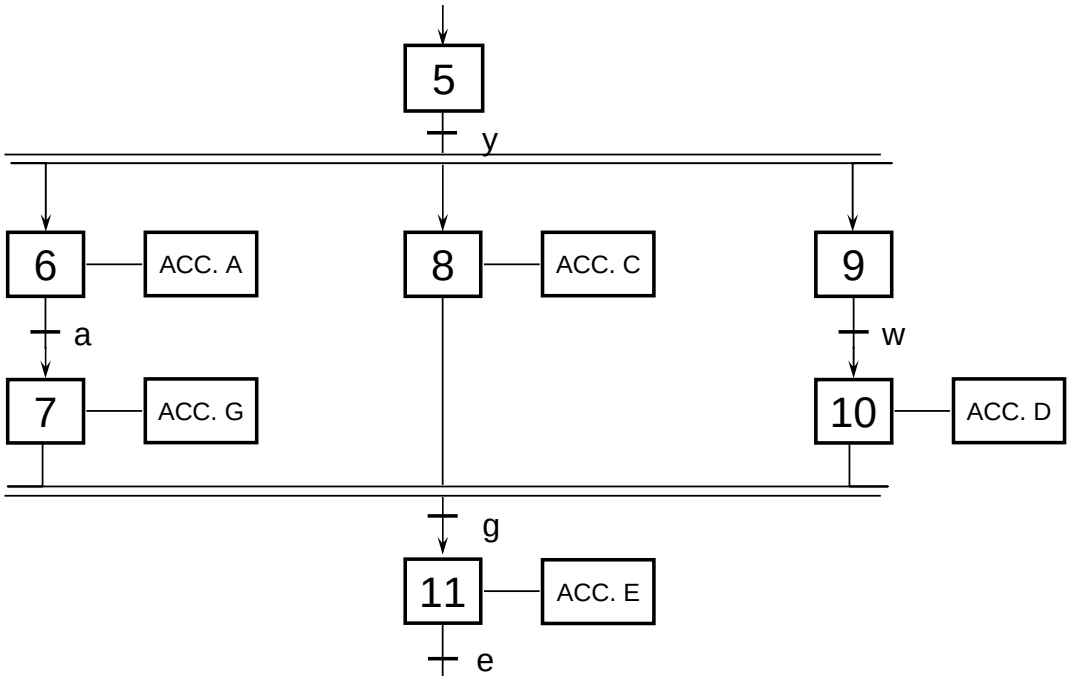
- DIRECCIONAMIENTO CONDICIONAL:

Elección condicional entre varias secuencias posibles



- SECUENCIAS SIMULTÁNEAS:

Varios estados activos a la vez



PLCopen
Standardization in Industrial Control Programming

Fuentes de Información

Fuentes de Información

- http://www.pclopen.org: organización que vela por la estandarización del Sw aplicado a los PLC
- http://olmo.pntic.mec.es/~jmarti50/enlaces/grafcet.html: todo sobre Grafcet
- http://isa.uniovi.es/genia/spanish/app/prog/mediss_5.htm: demo de Grafcet para PLC de Siemens