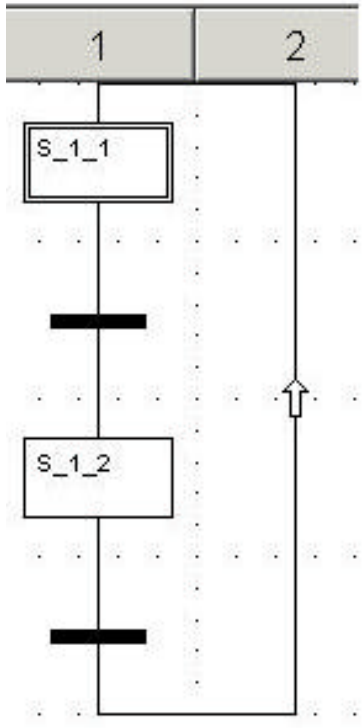
Modulo 3.A

Lenguaje IEC: SFC

¿Qué es SFC?



SFC : **S**equential **F**unction **C**hart, diagrama de funciones secuencial

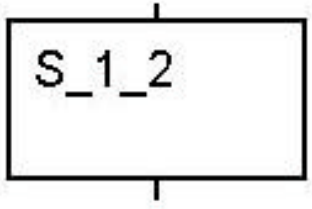
- Metodo gráfico para representar un sistema de control secuencial utilizando una secuencia de etapas y transiciones
- Cada etapa es una acción que puede estar activa o inactiva
- El control de flujo pasa de una etapa a la siguiente por medio de una trasición condicionada ya sea verdades o falsa.
- Estandar IEC 1131-3

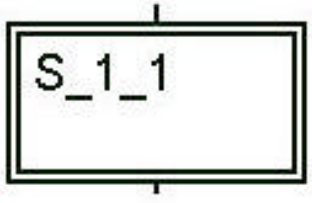
Reglas SFC

- El programa secuencial se compone de :
  - secciones SFC (nivel alto)
  - secciones de acciones
  - Secciones de Transiciones
- Las secciones SFC unicamente son permitidas en la tarea maestra
- Cada sección SFC contiene una o varias gráficas SFC
- Una sección SFC tiene 200 lineas y 32 columnas
- Una sección SFC puede contener 1024 pasos

Etapas

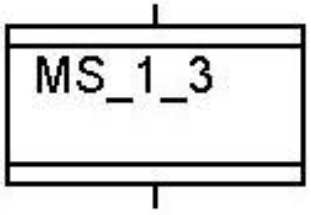
Una sección del SFC es un estado de la máquina ; Los estados se definen por etapas activas (100 etapas máximo se pueden definir en múltiple activación"token")



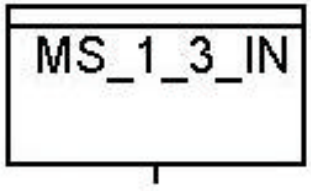


- **Etapa normal**
  - Se convierte en activa cuando la transición anterior es verdadera
  - Se convierte en inactiva cuando la transición que sigue es verdadera
  - Cada etapa puede contenir ningunam una o varias acciones
- **Etapa inicial**
  - Arranca la secuencia cuando la aplicación es inicializada.
  - No contiene acción
  - Un paso inicial debe ser definido en cada cadena secuencial (gráfica)
- Un calificador debe ser declarado en cada paso
- Una función de supervisión y temporización pueden ser definidas en

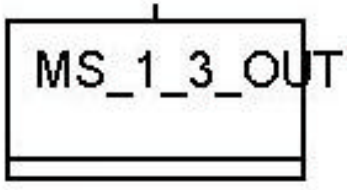
Macro etapa



- **Macro etapa**
  - Utilizado para llamar la sección de la macro etapa
  - La seccion de macro etapas debe ser activasa en las herraminetas de configuración de proyectos



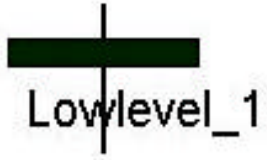
- **Etapa de entrada**
  - Primera etapa de una sección macro etapa
  - La etapa de entrada es automáticamente creada
  - No se puede borrar o insertar



- **Etapa de salida**
  - Ultima etapa de una sección de macro etepa
  - La etapa de salida es automáticamente creada
  - No se puede borrar ni insertar

Transición

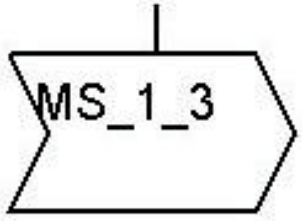
Para cada transición la condición puede ser :



- Una variable booleana

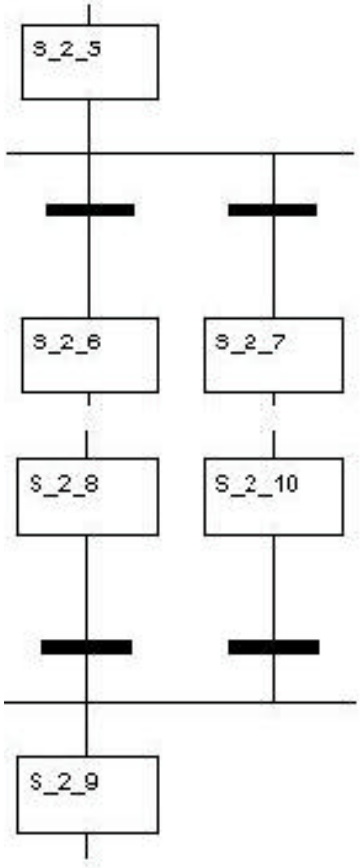
- Una sección transición
  - La transición condicionada es editada utilizando el lenguaje LD, FBD, ST o IL
  - El resultado de la evaluación de la transición condicionada es asignada a una variable booleana con el mismo nombre de la transición misma.

Salto



- Permite representar un enlace directo que cual no se dibuja de manera completa
- Puede utilizarse en cualquier ubicación de la gráfica, pero despues de una transición
- Varios saltos son posibles a el mismo punto

Ramificación alternativa (Alternative branch / joint)

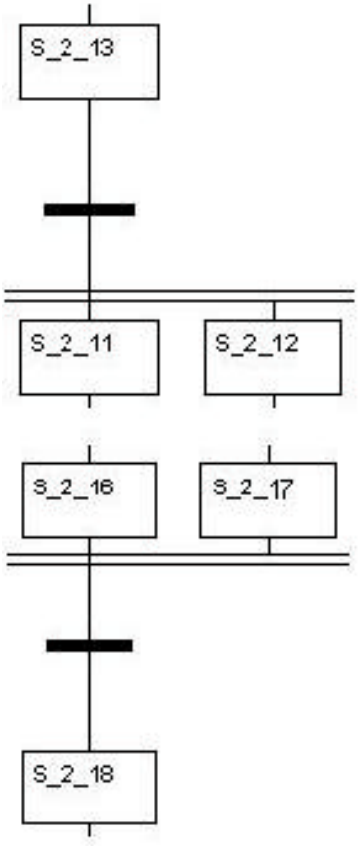


- **Ramificación Alternativa** (Divergencia O "OR")
  - La ejecución de la ramificación es determinada por el resultado de cada condición de transición, siguiendo la ramificación alternativa.
  - Las condiciones de transiciones son procesadas de izquierda a derecha.

- **Enlace alternativo** (convergencia O "OR")
  - Las diferentes secuencias, se unen nuevamente, en una ramificación en la cual el procesamiento continúa.
  - Normalmente un paso es permitido despues de un enlace arternativo
  - Una ramificación paralela es también permitida despues de un

Telemecanique

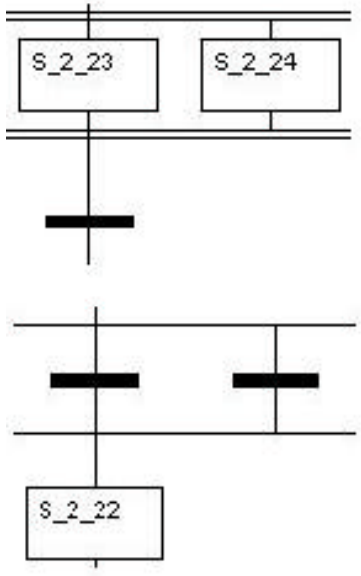
Ramas paralelas / unión



- **Ramas paralelas** (divergencia AND)
  - El Procesamiento es dividido en varias secuencias (32 máximo)
  - Las secuencias paralelas son procesadas de manera independiente de izquierda a derecha
  - Normalmente una transición es permitida antes de una ramificación paralela
  - Una unión alternativa es permitida despues de la ramificación paralela
- **Enlace paralelo** (convergencia AND)
  - Combina las secuencias paralelas en una ramificación
  - La transición seguida de un enlace paralelo es evaluada cuando todas las etapas prescedentes han sido resueltas (sincronización de secuencias paralelas)
  - Normalmente una transición es permitida despues de una ramificación paralela
  - Una ramificación alternativa tambien es permitida despues de un

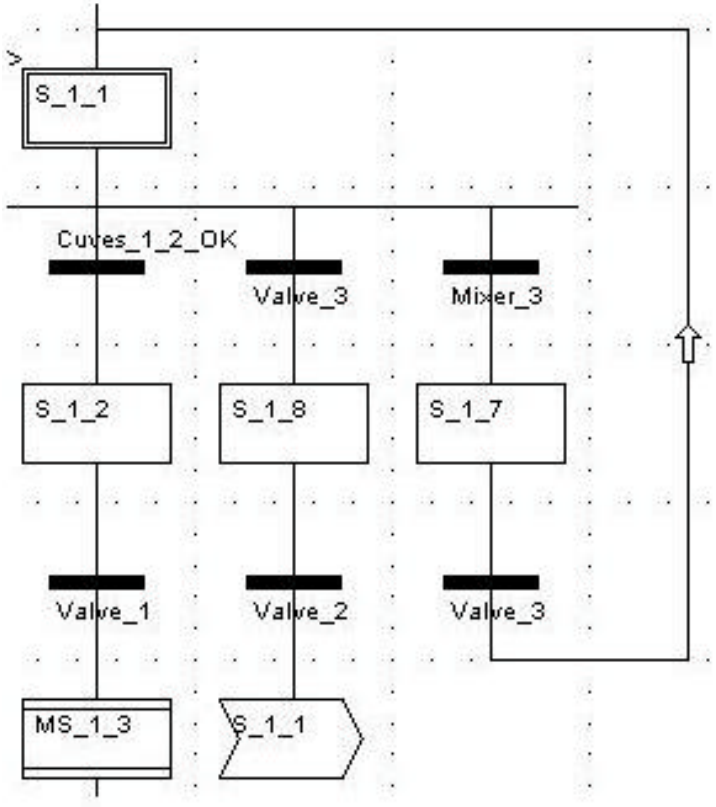
Secuencias

Para minimizar la creación de objetos una estructuración de secuencias es propuesta



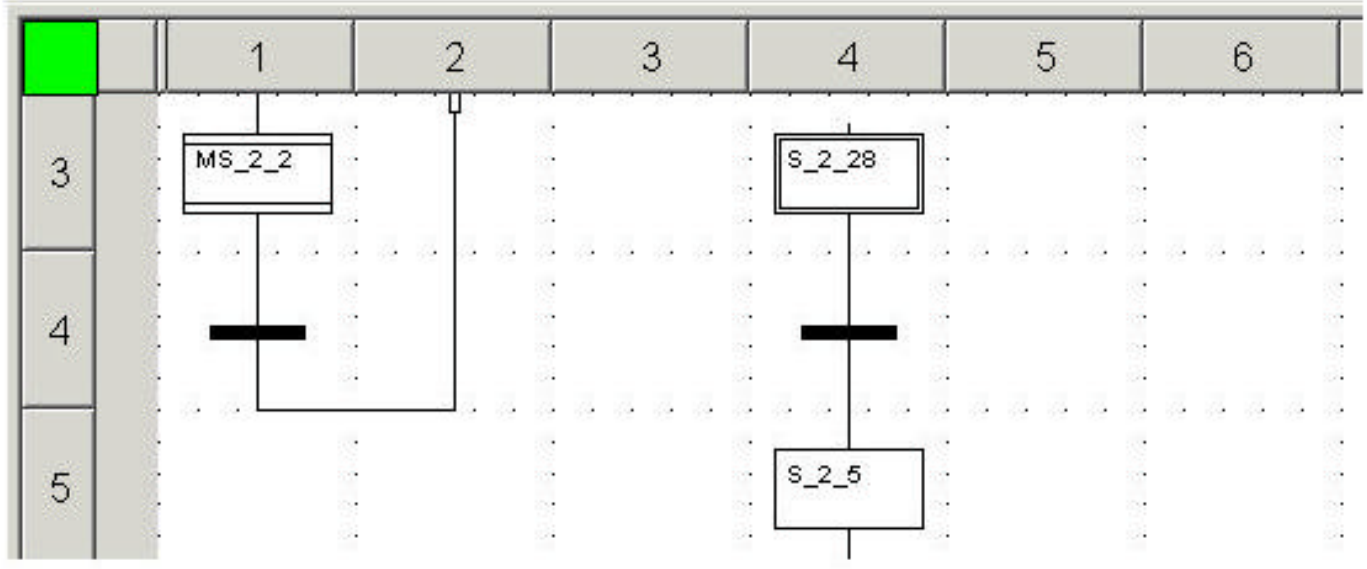
- **Secuencia simple**
  - Iniciando con una etapa / con una transición
- **Secuencia paralela conteniendo etapas**
  - divergencia AND
  - Ramificaciones
  - Convergencia AND
  - Transición Final
- **Secuencia de selección conteniendo transiciones**
  - Divergencia OR
  - Ramificaciones
  - convergencia OR
  - Etapa final

Enlace



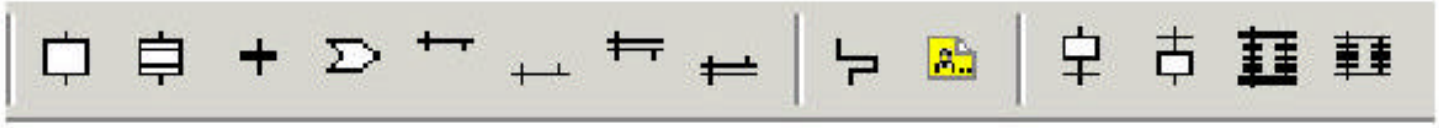
- El Enlace directo conecta una etapa a una transición o una transición a una etapa
- El enlace existe solo si la fuente y el destino existen
- El enlace nunca conecta una etapa a otra etapa o una transición a una transición
- El enlace nuca se eobrpone a otro objeto (etapa, transición, salto)
- Los enlaces nunca se sobreponen o cruzan con otros similares
- Un enlace está compuesto de segmentos ortogonales
- Una flecha indica el sentido del salto

Editor SFC



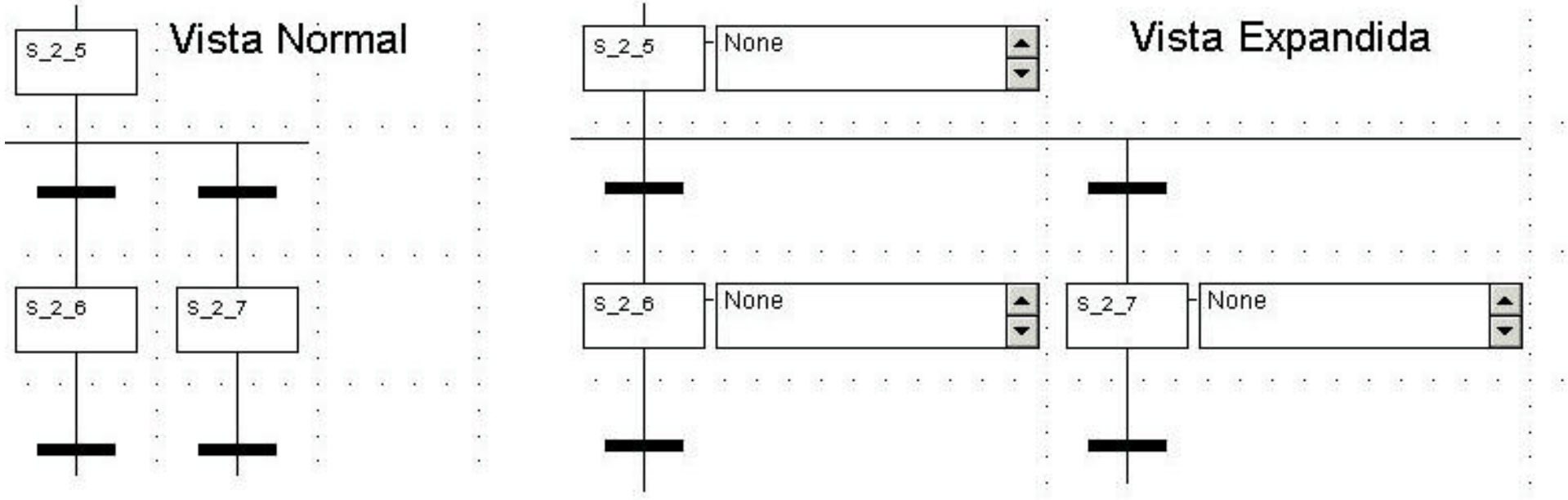
- Editor basado en una ventana que permite múltiples gráficas
- Al abrir una sección de una gráfica o una macro expansión se abre una ventana de edición
- Una celda puede contener diferentes tipos de objetos
- Una sección puede contener 200 filas y 32 columnas

Objetos de un programa SFC



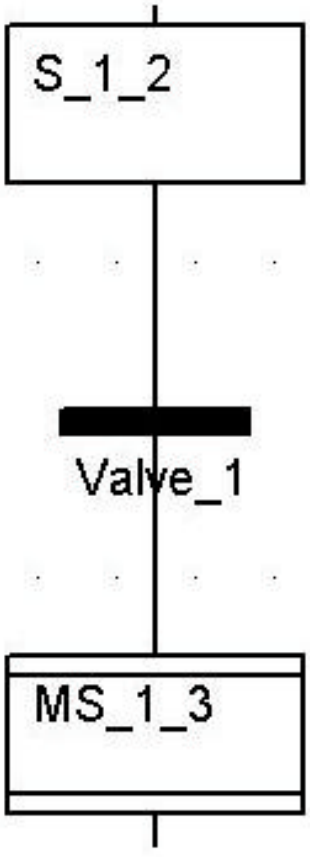
- Etapa
- Macro etapa
- Transición
- Salto
- Rama alternativa / Enlace Alternativo (divergencia OR / convergencia)
- rama paralela / enlace paralelo (divergencia AND / convergencia)
- Enlace
- Comentario
- Secuencia simple iniciando con un paso / transición
- Secuencia Paralela / selección de secuencia

Vista Normal / Expandida



- Vista expandida de las acciones asociadas a las etapas
- se pueden aplicar la función de acercamiento al editor

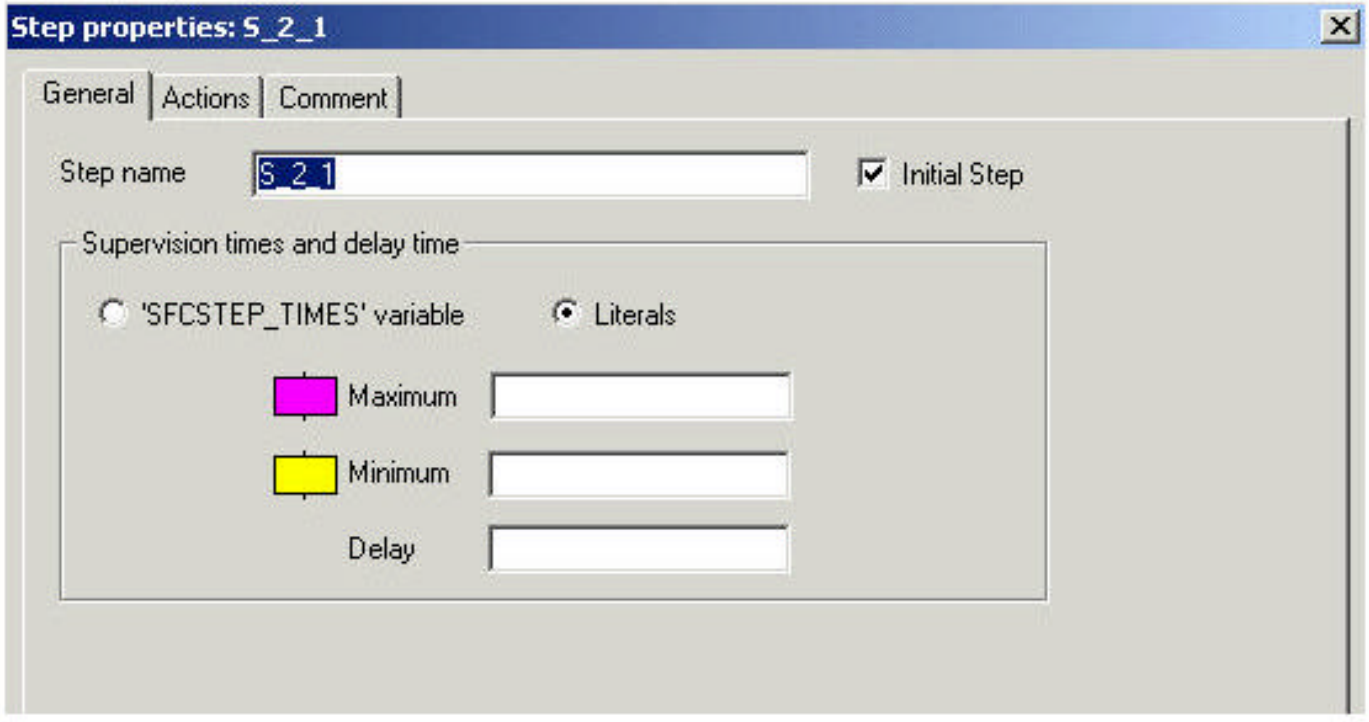
Nombre de Etapas



los nombres de etapas son como en IEC

- El nombre de la etapa es automáticamente generado y puede ser editado (cualquier nombre)
- El nombre de la etapa debe ser único en todo el proyecto
- El nombre de la etapa puede contener hasta 32 caracteres máximum
- Automaticamante se genera un nombre con la siguiente estructura : S_n_m (para la etapa) o MS_n_m (para una macroetapa)
  - S = Etapa o MS = Macro Etapa
  - n = número de sección (número consecutivo)
  - m = número de etapa en la sección (número consecutivo)

Propiedades de la etapa



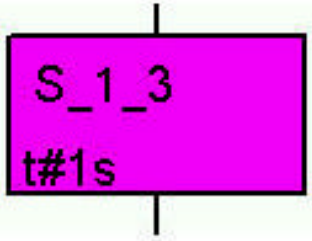
Propiedades de la etapa (continuación)

3 páginas a definir las propiedades de la etapa

- Nombre de la etapa
- Atributo de la etapa inicial
- Tiempo de Supervisión t temporizador
  - Forma literal (ejemplo 10 ms)
  - Variable SFCSTEP_TIMES
- Acciónes
  - Calificador
  - Tiempo deAcción
  - Tipo de Acción (variable, direccionamineto directo, sección)
- Comentario

Tiempo de etapas

Tiempo de retraso< Mínimo tiempo de supervisión< Máximo tiempo de Supervisión



Máximo Tiempo de Supervisión

- Máximo tiempo en el cual la etapa debe estar activa
- Si la etapa continua activa despues de este tiempo :
  - aparece un mensaje de error
  - La etapa toma el color magenta (modo animación)

Mínimo Tiempo de Supervisión

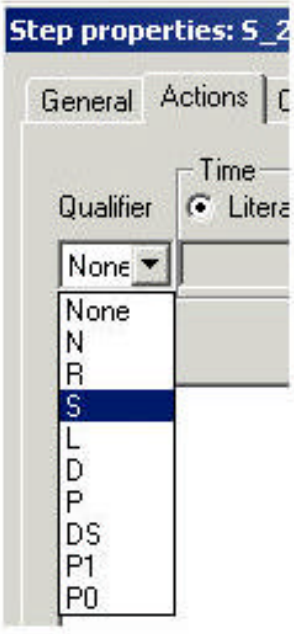
- Minimo tiempo en el cual la etapa debe estar activa
- Si la etapa pasa al estado inactivo antes de este tiempo:
  - Aparece un mensaje de error
  - La etapa toma el color amarillo (modo animación)

Tiempo de retardo (La etapa dwell time)

- Mínimo tiempo en el cual la etapa debe estar activa

Calificadores

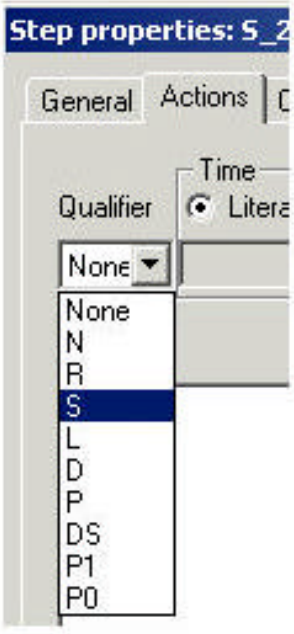
Definen el control para la acción ligada a un paso



- Ninguno / N : acción es activada a (1) cuando la etapa es activa
- R (Reset) : la acción inactiva (0) cuando el paso es activo
- S (Set) : la acción es activa y se mantiene aún si la etapa vuelve a ser inactiva.
- L (time Limited) : La acción es activa surante un periodo de tiempo de la etapa.
- D (Delayed) : la acción sera activa despues de un tiempo y se mantendrá activa en el periodo de la etapa
- P (Pulse) : La acción es activa cuando el paso es activo y se mantiene activa durante un ciclo de programa
- DS (Delayed and Set) : la acción es activa despues de un tiempo y se mantiene activa aún si la etapa es inactiva, si la etapa toma el estado inactivo antes del tiempo la acción no se activa

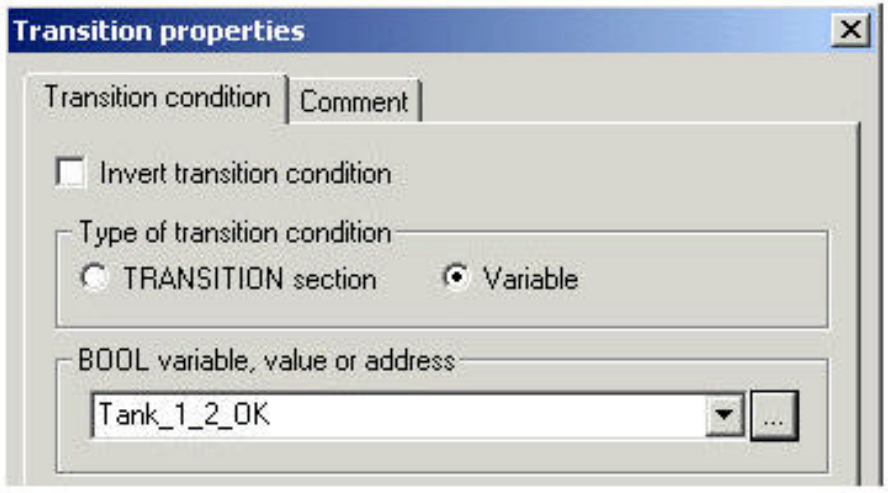
Calificadores (cont)

Define el control de la acción ligada a la etapa



- P1 (Plus, frente positivo) : La acción es activada en un ciclo de programa cuando la etapa está activa (0 -> 1 frente ascendente)
- P0 (Plus, frente negativo) : La acción es activada en un ciclo de programa cuando la etapa está activa (1 -> 0 frente descendente)

Propiedades de la transición

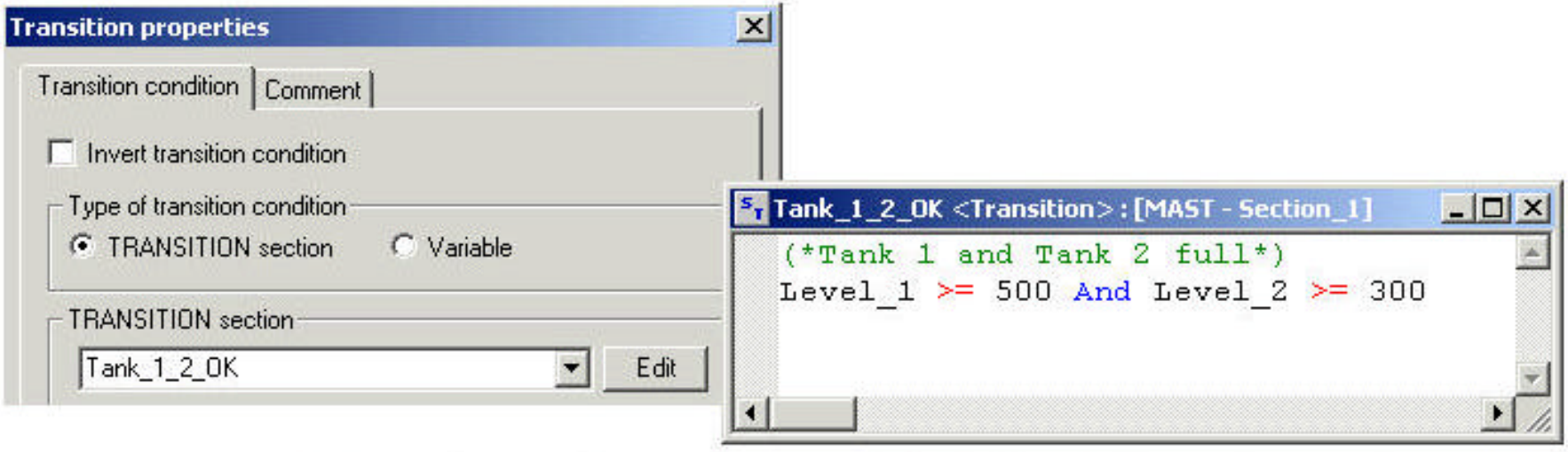


variable Booleana (Valvula_1)
Valor booleano (0 o 1)
Dirección Booleana (%M, I, ...)

2 páginas definen las propiedades de la transición

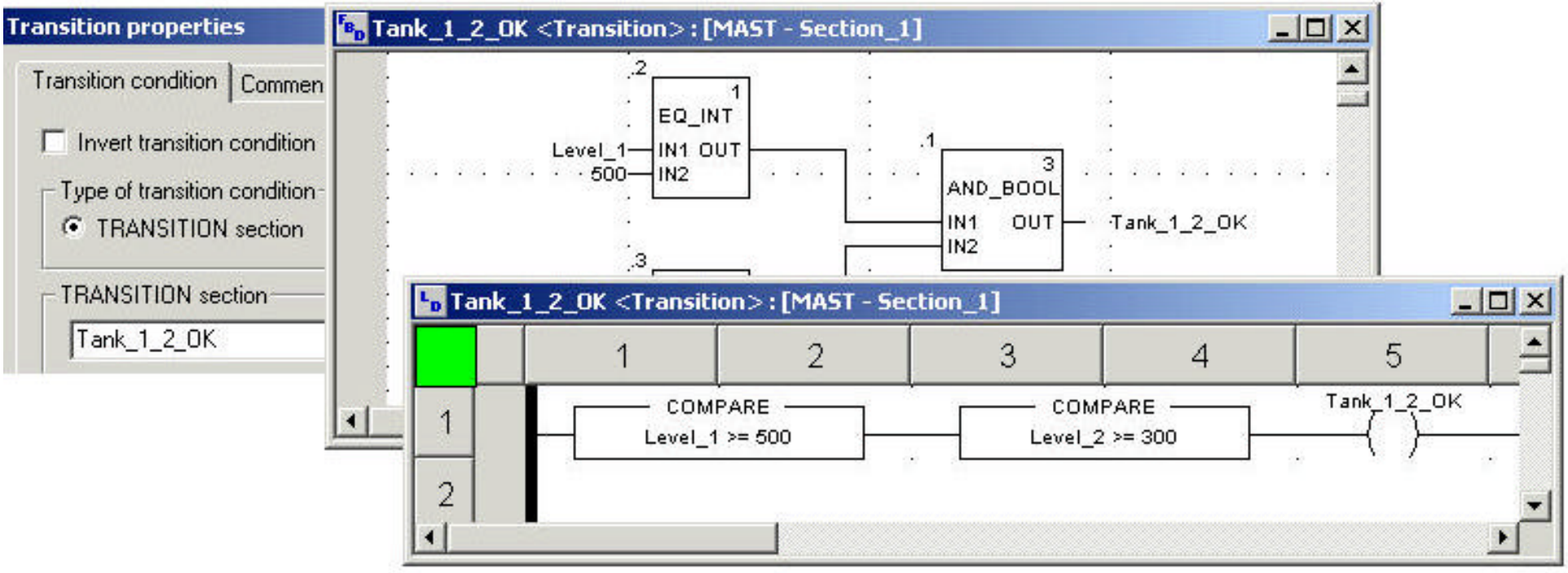
- Condición Invertida de transición (activa a 0)
- Tipo de transición condicionada (sección transición o variable)
- Comentario

Sección transición



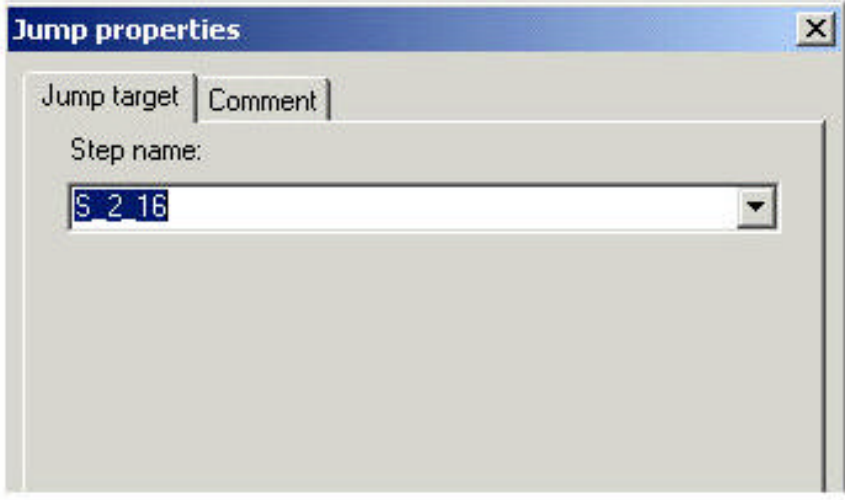
- En IL y ST, contiene una expresión lógica de la cual el resultado es asignado a la variable de la transición (ejemplo; la asignación de memoria *ST* no está permitida en IL, la instrucción := no está permitida en ST)

Sección transición(cont)



- En FBD, este contiene un bloque AND con una salida ligada a una variable de transición
- En LD, este contiene una red con una bobina ligada a una variable de transición

Propiedades del salto

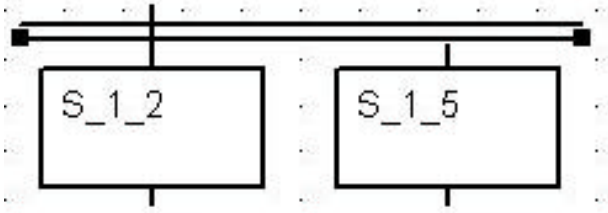


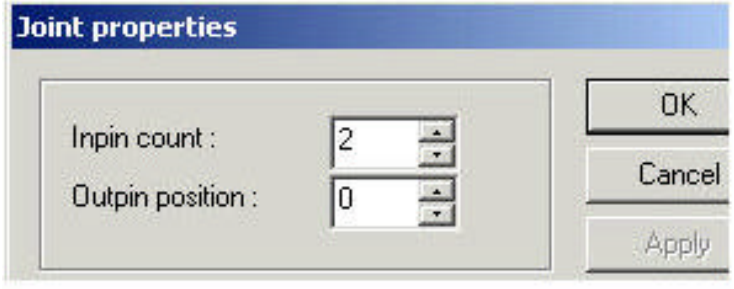
2 pasos para definir las propiedades del salto

- objetivo del salto (nombre dela etapa o macroetapa)
- Comentario

Telemecanique

Propiedades Ramificación/ unida.





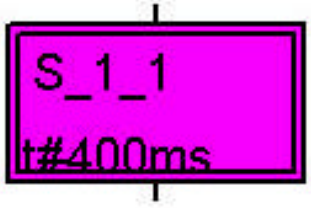
- "Herramienta de ajuste de tamaño" Permiten el ajuste de la longitud deseada

- Ventana de configuración de la ramificación.
  - Número de salidas ó conectores (tamaño del objeto)
  - Posición de los conectores

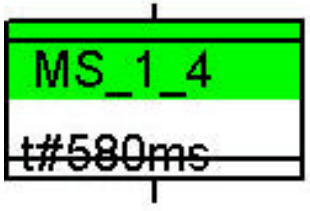
- Ventana de configuración propiedades de Enlace Unión
  - Número de conectores (tamaño del objeto)
  - Posición de los conectores outpin

Colores de los objetos SFC

Los colores son utilizados cuando la sección está animada

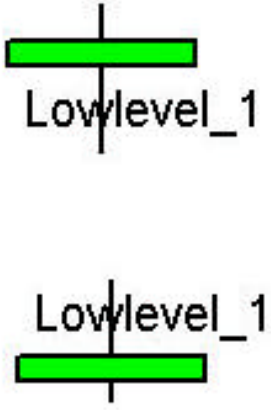


- Paso o etapa
  - Verde : Etapa activa
  - Blanco: Etapa inactiva
  - Amarillo: El tiempo de la etapa es más pequeño que el mínimo tiempo de supervisión.
  - Magenta : El tiempo de la etapa es más grande que el máximo tiempo de supervisión.



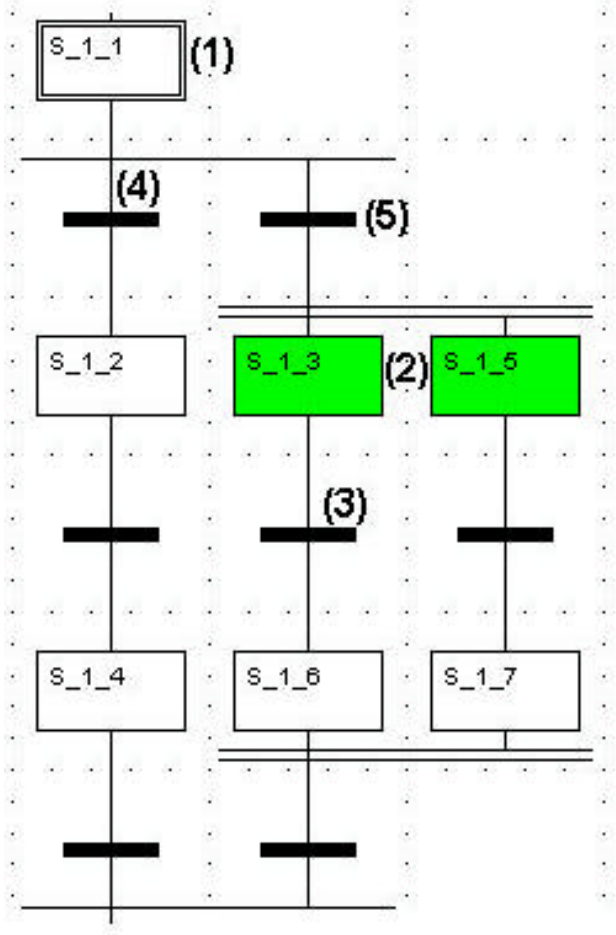
- Macro etapas
  - Verde (parte superior): macro etapa es activa
  - Verde (parte superior e inferior) : Salida de la macro etapa.
  - Blanco: La macro etapa es nactiva

Colores de objetos SFC (continuación)



Los colores son utilizados cuando las secciones son animadas.

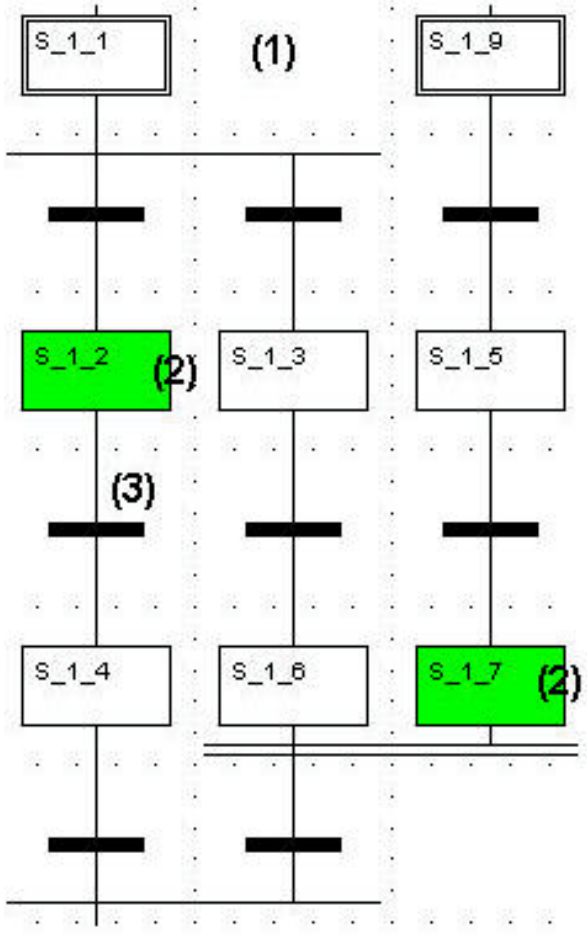
- La transición está enlazada a una variable boleana o una simple expresión Boleana.
  - Verde: La expresión boleana es verdadera
  - Rojo : Variable boleana o expresión Falsa
- La Transicion enlazada a una sección
  - Negro: etapa INACTIVA
  - Verde : las condiciones de la sección es VERDADERA
  - Rojo : las condiciones de la sección es FALSA



Ejecución de una etapa

- Solamente hay una etapa inicial (1)
- Una sola etapa es activa excepto en ramas paralelas (una etapa por rama) (2)
- Una transición es evaluada si la etapa previa es activa (3)
- Las ramas con transiciones son procesadas de izquierda a derecha
- Sola una rama puede ser activa en ramas alternativas (4)
- Con ramas paralelas una transición puede activar varias etapas (32 max) (5)
- Saltos hacia una rama o dentro de una rama paralela no son posibles.

Telemecanique



Múltiple ejecución

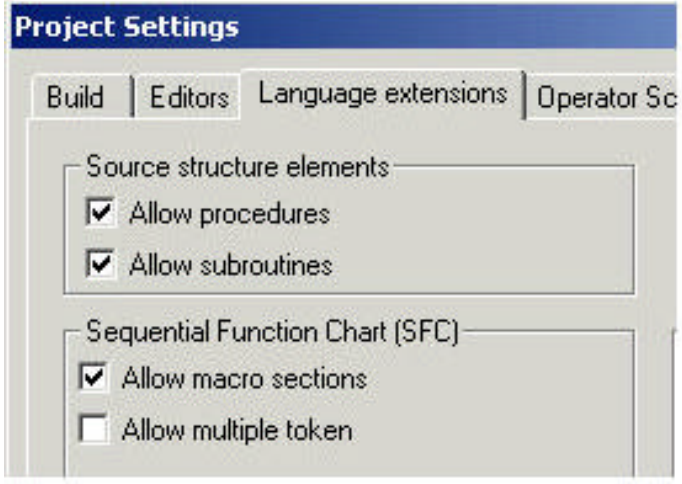
- Vários pasos o etapas iniciales (0 a 100) (1)
- Una macro etapa puede contener etapas iniciales
- Varias etapas pueden ser activas al mismo tiempo (2)
- Una transición es evaluada si la etapa previa es activa (3)
- Más de una ramificación puede ser activa en ramas alternativas
- En una rama paralela una trasición puede activar varias etapas
- Los saltos dentro de o fuera de una rama paralela es posible
- Las etapas son activadas desde una sección que no sea SFC (SETSTEP block)

Objetos del sistema en la programación

Los objetos del sistema para controlar la gráfica son remplazados por EF / EFB

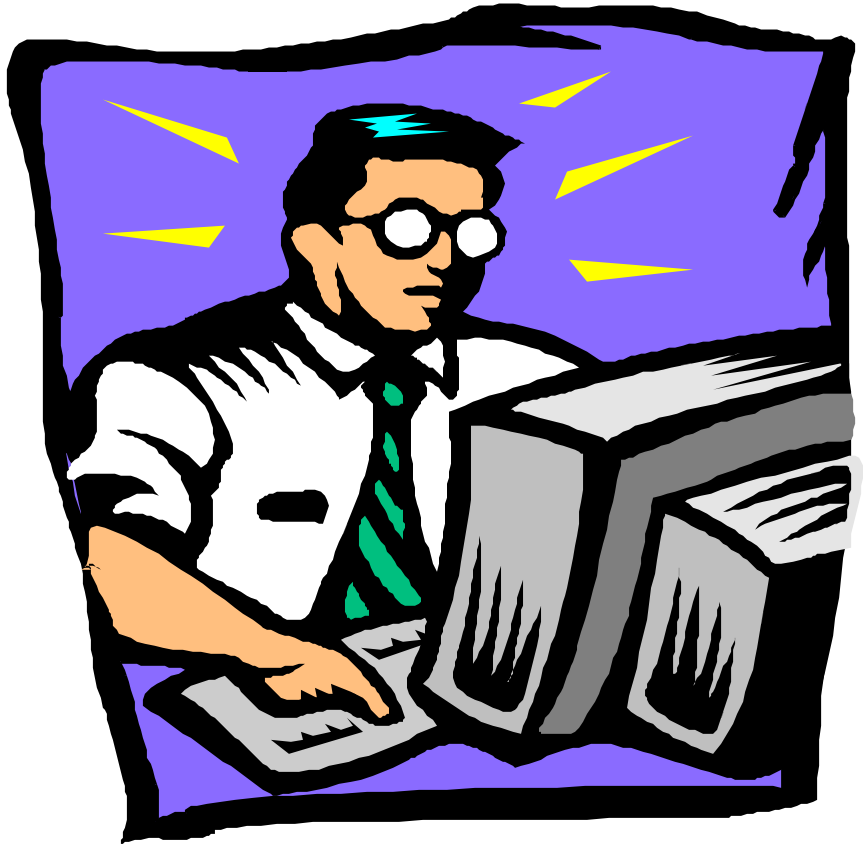
Objetos PL7	Función	Remplazados por
%S21	Borra las etapas + activa todas las etapas iniciales	INIT (SFCCNTRL EFB) Función INITCHART
%S22	borra todas las etapas	CLEAR (SFCCNTRL EFB) Función CLEARCHART
%S23	Congela todas las etapas	DISTRANS (SFCCNTRL EFB) Función FREEZECHART
%S24 %SW22 - 25	Borrar todas las macro etapas borrado de la tabla Macro etapa	INIT (SFCCNTRL EFB) Función INITCHART
%S26 %SW125 - 127	error de secuencia	Visualizador de enventos
%SW20	Número de etapas activas	Sin equivalencia
%SW21	Número de transiciones válidas	Sin equivalencia

Nuevas características del Grafcet



- Es posible tener varias secciones SFC
- Las secuencias PRL/GRAPH/POSTno existen. PRL y POST son remplazadas por sección que no son SFC
- Varios pasos o etapas activas al mismo tiempo "Multiple token"
- Etapa inicial dentro de una macro sección
- Ajuste de las etapas es posible (0 o 1)en todas las secciónes (excepto en la sección SFC y secciones de subrutina) (solamente en el PRL con el Grafcet)

Ejercicio 9



Módulo 3.B

Languaje IEC : ST

Panorama

- Los Lenguajes en computación (como VBA, Pascal, ...) utilizan una gama de instrucciones para asignar valores a variables, llamados FFBs, creando expresiones para su evaluación condicional, iteración o repeticion del código selecciónado de la sección
- utilizado para escribir lógica estructurada y digital procesando programas
- Cumpliendo con el estandar IEC 61131
- Fácil de aprender y a utilizar
- Particularmente adecuado para programación compleja y funciones aritméticas

Referencia del lenguaje

Un programa de texto estructurado es una secuencia de líneas ,cada una está compuesta de uno o más :

test_st : [MAST]

```
LOOP:
SUM := 0 ;
FOR I := 1 TO 3 DO
 FOR J := 1 TO 2 DO
  IF FLAG=1 THEN EXIT; (*Exit of the loop*)
  END_IF ;
 SUM := SUM + J ;
 END_FOR ;
SUM := SUM + I ;
END_FOR;
```

- Etiquetas : definen un bloque de programa
- Instrucciones: terminadas por un punto y coma ";"
- Comentarios: información adicional

LOOP :

IF FLAG=1 THEN EXIT ;

(*Exit of the loop*)

Referencia del Lenguaje (cont.)

test_st : [MAST]

```
LOOP:
SUM := 0 ;
FOR I := 1 TO 3 DO
 FOR J := 1 TO 2 DO
  IF FLAG=1 THEN EXIT; (*Exit of the loop*)
  END_IF ;
 SUM := SUM + J ;
 END_FOR ;
SUM := SUM + I ;
```

+

SUM

IF FLAG=1 THEN EXIT

- Operador: símbolo que define la operación que se debe realizar
- Operando: El objeto en el cual la operación es efectuada

Operadores

Estos son las siguientes instrucciones básicas:

- Asignar (:=)
- Operadores Lógicos (AND, OR, XOR, NOT)
- Operadores Aritméticos (+, -, *, /, MOD, **)
- Comparación (>, <, =, >=, <=, <>)
- Llamado de funciones (FUNCTIONSNAME)

Operadores

Asignación

Operador	Orden de prioridad	Significado/ Operando
()	1(la mas alta)	Los parentesis son utilizados para alterar la ejecución de las expresiones del operador
:=	-	Afecta el valor del operando Variable, direccionamineto directo o cualquier tipo de datos

Call

Operador	Orden de prioridad	Significado/ Operando
FUNCTIONSNAME	2	Executing a function Value, variable, direct address (data type is dependent on function)

Arithmetic

Operator	Orden de prioridad	Significado/ Operando
-	3	Negación : con esta negación el operando toma considera el valor inverso de la variable - Expresiones asiciadas , Literal, Variable, Dirección o datos tipo INT, DINT, UINT, UDINT o REAL
**	4	Exponencial: para exponenciació ** el valor del primer operando es exponenciado con el valor del segundo operando (exponente). Expresion, Literal, Variable, Dirección o tipo de dato tipo REAL e INT, DINT, UINT, UDINT o REAL (Exponente)

Operandos

Un operando puede ser:

- una dirección directa:
- un valor
- una variable simbólica
- Un dato estructurado
- Un elemento de un dato estructurado
- un EFB/DFB salida (para un solo tipo)
- un EFB/DFB llamada

Comentario

Instrucciones de control

Estas son las siguientes instrucciones:

- IF...THEN...END_IF
- ELSE
- ELSIF...THEN
- CASE...OF...END_CASE
- FOR...TO...BY...DO...END_FOR
- WHILE...DO...END_WHILE
- REPEAT...UNTIL...END_REPEAT
- EXIT
- RETURN
- JMP

Estructura

Instrucción	Significado
IF...THEN...END_IF	Comandos de ejecución condicionada
ELSE	Ejecución de comandos si las expresiones o comandos anteriores son falsos
ELSIF...THEN	Comando de ejecución condicionada, si los elementos o expresiones anteriores son falsos
CASE...OF...END_CASE	Lista de comandos con etiquetas se ejecutan si las expresiones son verdaderas
FOR...TO...BY...DO...END_FOR	Repite el comando un número de veces indicado
WHILE...DO...END_WHILE	Repite la ejecución de comandos hasta la condición precedida = 0
REPEAT...UNTIL...END_REPEAT	Repite la ejecución de comandos hsta lograr la condición = 0.
EXIT	Termina la repetición de comandos (FOR, WHILE, REPEAT) antes del finalizar la condición es verdadera
RETURN	Para salir del programa en curso y regresar
JMP	Para acccesar etiquetas (unicamente permitido en la misma sección)

Comentarios

En el editor ST, los comentarios siempre inician con los caracteres, "(*" y terminan con "*)".

Entre estos caracteres cualquier comentario puede ser escrito.

Los comentarios integrados no están permitidos de acuerdo a la norma IEC 61131-3. Pero es posible si la opción es seleccionada en la opción de proyectos.

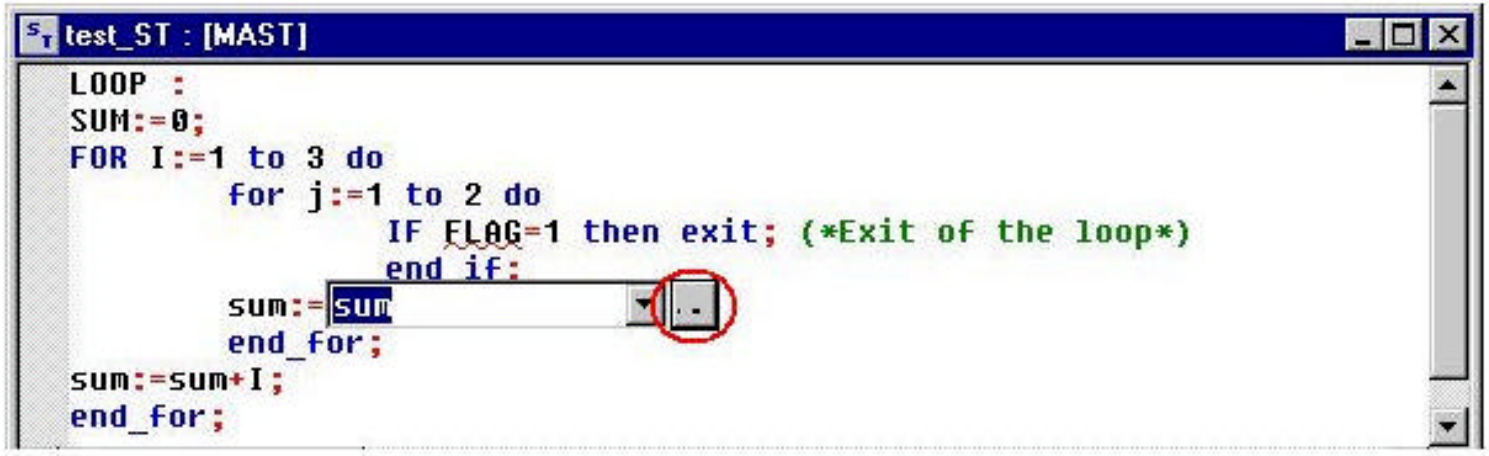
Etiquetas

Las etiquetas sirven como destino de los saltos.

- Deben estra siempre como primer elemento de una línea
- Debe ser única en una sección
- No es sensible al tamaño del caracter
- puede contener hasta 32 carcateres de largo (max.)
- debe estar separado por dos puntos ": "
- Solamente se permiten al inicio de la "Expresions"

```
IF var1 THEN JMP START;
:
:
START: ...
```

Editor del Lenguaje



- Mismo aspecto de un editor estandar (Wordpad) con funciones de copiar y pegar etc...
- una o más instrucciones por linea separadas un punto y coma ";"
- para definir diferentes objetos se utilizan diferentes colores
- Análisis automático de la sintaxis del programa y semántica durante la creación del programa .
- Menú de ayuda

Editor del lenguaje

Características

- Instruction deben estar entre un punto y coma ";"
- En linea se puede contener varias instrucciones, separadas por ";"
- Una línea contiene un máximo de 300 caracteres (instrucción, comentario, etiqueta)
- La longitud de una sección no está limitada
- La posición de las etiquetas y comentarios no es fija
- Buscar, remplazar puro texto ó variables
- La línea y columna es deplegada en la barra de estatus
- Copiar, pegar, cortar...
- Las secciones que contienen errores se pueden guardar (ejemplo error en sintaxis, variable no declarada, ...)

Marcado de colores

Azul	Operadores como AND, IF...THEN, FOR...DO, ...
Rojo	Operdores como "(", ":", ":=", "+", ...
Verde	Comentarios
Negro	Texto normal
Subrrallado rojo	Texto inválido (error detectado de sintaxis o semántica)
Video inverso	texto seleccionado

Tool tip

Si el cursor es puesto encima de una texto de falla , una información (Tool tip) aparecera con una breve descripción .

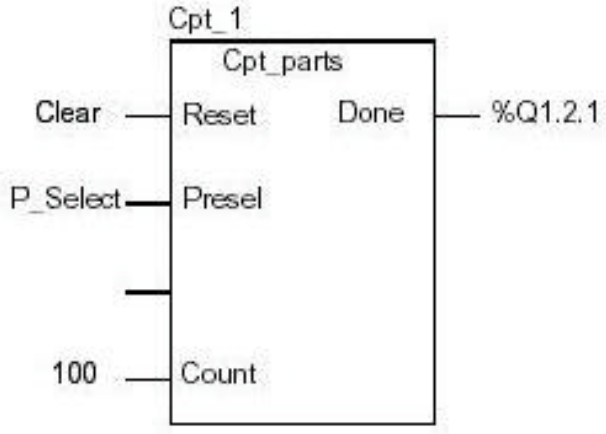
Barra de herramientas específica

Para Insertar :

- IF
- FOR
- WHILE
- Repeat
- CASE
- asistente de configuraciónFFB
- Commentario
- ventana de Inspección
- Subroutina
- Fuente todas en Mayusculas o minúsculas

Comentario

Reglas de Programación




```
Cpt_1 (Reset:=Clear, Presel:=P_Select,
Count:=100);
...
%Q1.2.1:=Cpt_1.Done;

Cpt_1 (Clear, %MD10, , 100);
...
%Q1.2.1:=Cpt_1.Done;
```

- Variable
  - definida directamente en la ventana de datos
- Funciones elementales
  - no requiren declaración
  - EF utilizado directamente
  - Ingresado directamente o con la función de ayuda de FFB
- Bloque de función elemental o bloque de función derivado
  - no requiere declaración (automaticamente définido con la ayuda de FFB)
  - instancia EFB/DFB-utilizado directamente
  - llamado Formal o informal
  - Ingresado directamente o con la ayuda de FFB

Reglas de programación

Las funciones elementales (EF) tienen disponibles operadores disponibles en texto estructurado y no necesita ser llamado (función CAL no utilizada).

No hay declaraciones con una sección , VAR y END_VAR no pueden ser utilizados. Si una declaración de variables es necesaria, esta debe ser hecha utilizando un Editor de datos (creando una referencia de una variable, EFB, ...)

Una referencia EFB/DFB-es automáticamente creada si el FFB Wizard ee utilizado para crear una sección EFB.

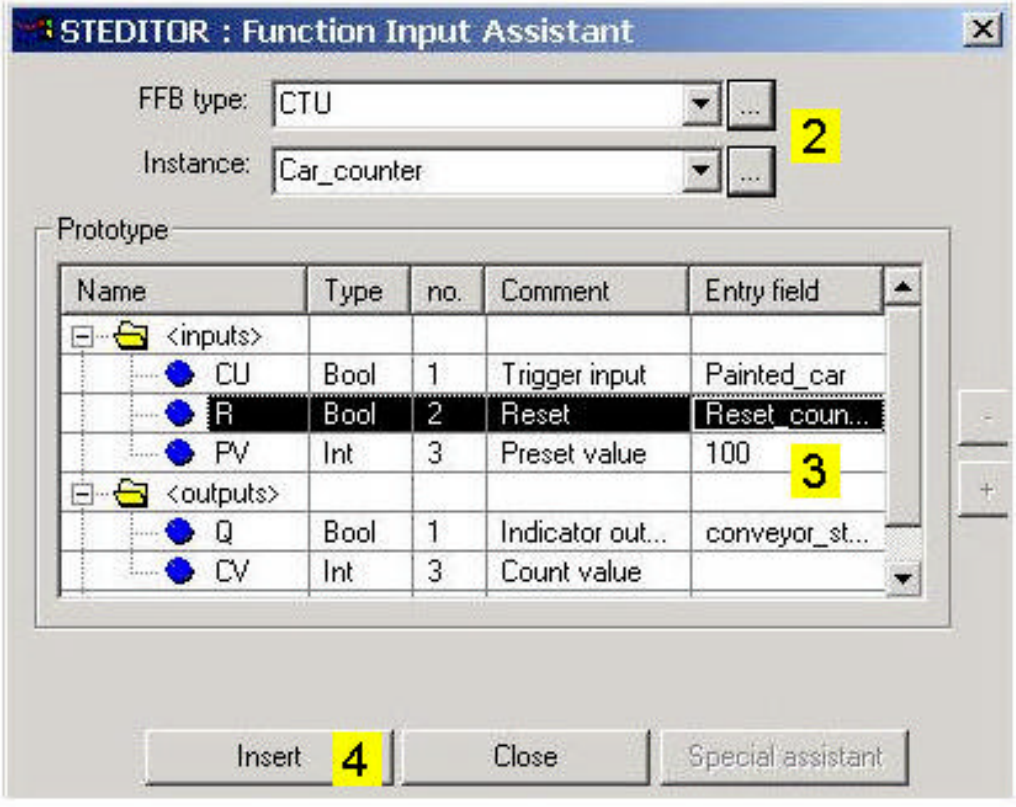
EL EFB/DFB llamado puede ser uno de dos tipos :

- una llamada formal , cuando los argumentos están asignados (parametro = valor). en este caso ,el orden en que los argumentos son definidos en la lista no son importantes. El pará metro de entrada EN y el de salida ENO puede ser utilizado para controlar la ejecución del bloque función.
  Ejemplo :
  Cpt_1 (Reset:=Clear, Presel:=P_Select, Count:=100, Done=>%Q1.2.1);
  or
  Cpt_1 (Reset:=Clear, Presel:=P_Select, Count:=100);
  ...
  %Q1.2.1:=Cpt_1.Done;
- Una llamada informal, cuando los argumentos son valores (expresiones, objectos o un valor inmediato).En este caso. el ordern el el cual los argumentos son definidos en la lista deben seguir un orden del los parámetros de entrada del DFB, incluyendo las entradas no asignadas
- Ejemplo :
  Cpt_1 (Clear, %MD10, , 100);
  ...
  %Q1.2.1:=Cpt_1.Done;

Desarrollo del Programa

- Crear o abrir una sección en texto estructurado ST
- Ecribir el programa y editar las variables utilizadas
  - Oprimir Enter o seleccionar el operador, control construct, ...
  - Oprimir Enter or seleccionar operando (variable)
  - Oprimir Enter o seleccióna la instancia EFB/DFB
- Analisar el programa editándolo frecuentemente para verificar la sintaxis
  - Cualquier error encontrado será desplegado en la ventana de mensajes
  - Hacer un doble click en la linea para ir a la linea que contiene el error en ST
- Guardar el programa
- Generar el proyecto y descargarlo al PLC
- Transferir y conectar el proyecto al PLC
- Probar el programa en modo conectado

Asistente de FFB



- Presione el icono Asistente
- Seleccione el FFB por
  - El tipo
  - La instancia

Nota : la Instancia es creada automaticamente si se selecciona el tipo de FFB

- En el campo de entrada seleccione para cada entrada o salida la variable.
- Oprima el boton **Insert** para validar

Telemecanique

Ejemplos de Programación

main : [MAST]

```
(* Simple example *)
Delay:=t#5s;

(* Start the motor after delay (in seconds) *)
IF not fault_motor THEN
                MOTOR_TIMER (PT := Delay,
                             IN := start_motor,
                             Q => motor_run,
                             ET => timer_value);
END_IF;

(* Stop the motor after ten seconds in running *)
STOP_TIMER (motor_run, t#10s);
IF stop_timer.q THEN
                start_motor := false ;
END_IF;

(* fault motor *)
fault_motor:= overspeed or hot_temp;
```

- Comentarios y asignación
- LLamado Formal EFB con asignación
- Llamado Informal EFB
- Función OR con asignación

Probando la aplicación

```
(* Simple example *)
Delay:=t#5s;

(* Start the motor after delay (in seconds) *)
IF not fault_motor THEN
                    OR_TIMER (PT := Delay,
                              IN := start_motor,
                              Q => motor_run,
                              ET => timer_value);
END_IF;
```

fault_motor
TRUE

timer_value : t#0s

Variables window - main : [MAST]

Name	Value	Type	Comment
Delay	5s	Time	
motor_timer		TON	
<inputs>			
IN	0	Bool	Start delay
PT	0s	Time	Preset delay time
<outputs>			
<inputs/outputs>			
<public>			

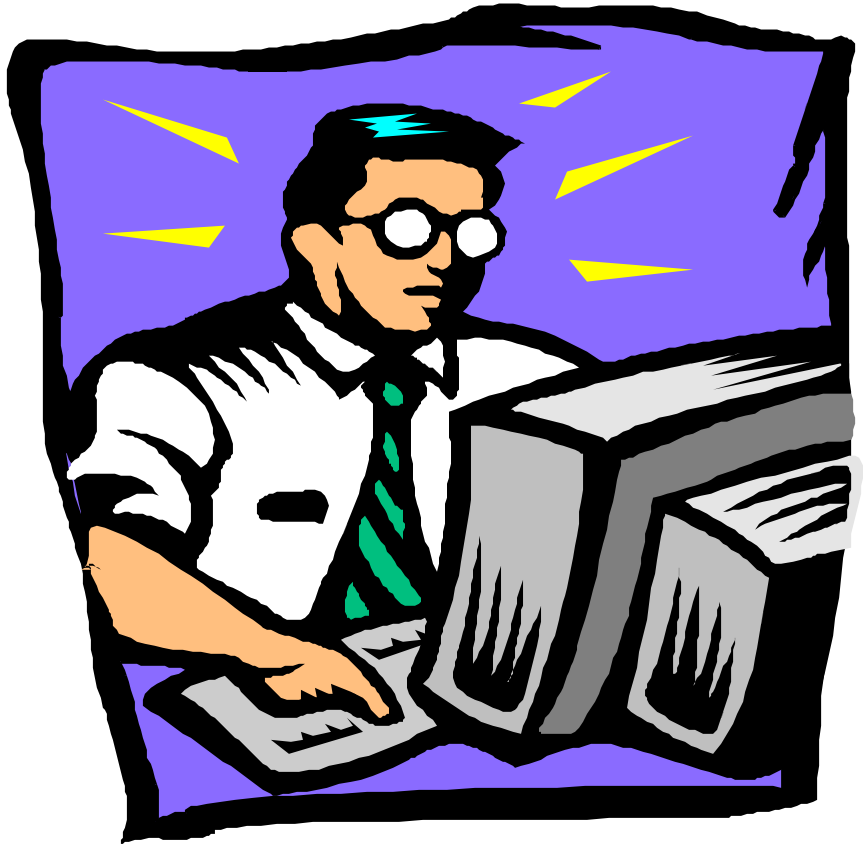
Animación de sección ST :

- Variable Boleana es verdadera: Verde
- Variable Boleana es falsa: Rojo
- Variables de otro tipo: Ambar

Las variables no boleanas no son animadas en el editor pero lo son en :

- Ventanas de tips
- Ventanas de inspección
- Ventanas de visualización de variables
- Tablas de animación

Ejercicio 10



Notas