



---

## ÍNDICE DEL MANUAL

### PRESENTACIÓN

Presentación del manual de prácticas página 4

### BREVE INTRODUCCIÓN TEÓRICA

El autómata TSX Micro página 6  
Tipos de variables bajo PL7 página 8  
Tipos de instrucciones página 10  
Estructura de una red página 11

### PRÁCTICAS

1. Configuración básica de una aplicación página 12  
2. Control de un extractor página 16  
3. Arranque directo de motor trifásico de inducción página 20  
4. Selección de cajas página 24  
5. Mando de una escalera mecánica página 30  
6. Control de acceso a una instalación página 36  
7. Señales luminosas página 40  
8. Control de un cruce con semáforos página 44  
9. Automatismo de un garaje página 52  
10. Regulación de temperatura página 56  
11. Control de posición con codificador página 62  
12. Diálogo Hombre-Máquina con pantallas de explotación página 72

### PREGUNTAS TÉCNICAS MÁS FRECUENTES

Cuestiones técnicas página 82

### DOCUMENTACIÓN

Webgrafía y bibliografía página 84



## **PRESENTACIÓN DEL MANUAL DE PRÁCTICAS**

Con el afán de ofrecer un mejor servicio, el Centro de Formación de Schneider Electric España edita el manual de prácticas para el autómata TSX Micro. La finalidad de este, es la de proporcionar al alumno y al profesor una herramienta que le permita introducirse y ampliar sus conocimientos de programación en autómatas MODICON TSX, mediante la práctica y resolución de diferentes ejercicios.

Dado el amplio campo de aplicaciones existente y la flexibilidad que permite la programación en sus diferentes lenguajes, las soluciones que se exponen en cada práctica serán sólo un ejemplo que permitirán alcanzar una metodología de programación estándar para resolver las cuestiones más diversas que se plantean en la industria actual.

La realización de estas prácticas se llevarán a cabo con la maqueta didáctica TSX Micro que el centro de formación pone a disposición de sus clientes. Esta maqueta permite el desarrollo de diversas aplicaciones gracias a los componentes que incorpora de serie: autómata de amplias prestaciones, simulador de E/S digitales y analógicas, contaje rápido para codificadores, terminal táctil Magelís para el diálogo hombre-máquina, etc.

Respecto a la programación de las aplicaciones se utilizará la última versión del software PL7PRO, que en el momento de editar este manual corresponde a la versión 4.4. Sin embargo, y dado el tipo de ejercicios que se llevarán a cabo, los software PL7MICRO y PL7JUNIOR en sus versiones más recientes serán totalmente válidos.



## BREVE INTRODUCCIÓN TEÓRICA

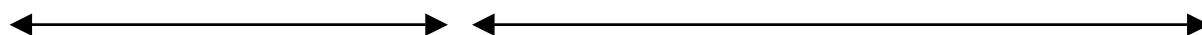
### El autómatas TSX Micro

La gama de autómatas TSX Micro se compone de varios tipos de autómatas con el fin de dar una mejor respuesta a los diferentes tipos de aplicaciones. Dentro de esta gama se proponen los autómatas compactos y los modulares.

Los autómatas compactos se identifican con la referencia TSX-3705 y TSX-3708 siendo adecuados para el desarrollo de aplicaciones sencillas. Mientras que el 05 incorpora un módulo de 16E / 12S discretas, el 08 posee dos módulos de este tipo que le permiten un mayor tratamiento de E/S de serie. Por otro lado, mientras que el número máximo de E/S para el 05 es de 92, para el 08 es de 120.

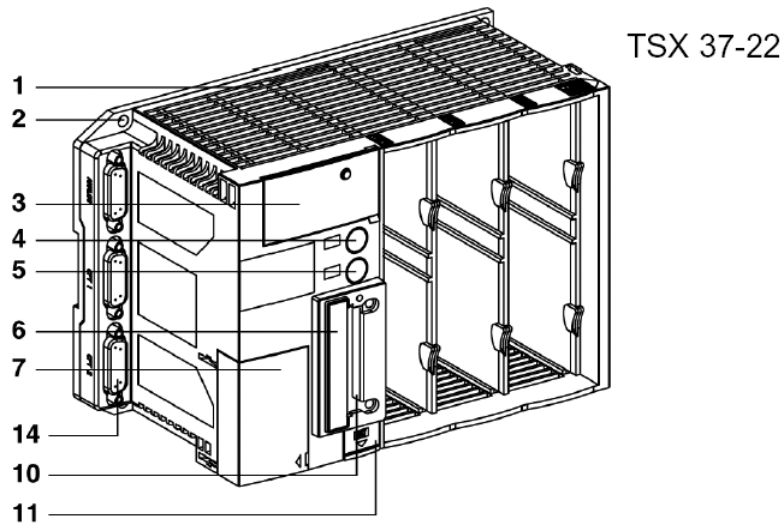
Para los autómatas modulares existen las referencias TSX-3710, TSX-3721 y TSX-3722. Dado que no incorporan módulos de E/S de serie, permiten una configuración libre al usuario multiplicando sus posibilidades mediante el rack extensible opcional. Las versiones 21 y 22 incorporan dos puertos de comunicaciones para diálogo operador ( modbus, unitelway ó ASCII ) y dos ranuras para la extensión de memoria y/o una tarjeta de comunicaciones ( modbus plus, fipway, MODEM, etc ). Además, la versión 22 posee tres conectores integrados: dos para contaje rápido y otro para 8 entradas y 1 salida analógicas. El número máximo de entradas y salidas que puede manejar este autómatas asciende a 368.

Otra de las características que hacen del TSX37 un autómatas versátil para la gran mayoría de aplicaciones, es la amplia gama de módulos de entradas y salidas que maneja. Es posible clasificar esta gama, en un primer acercamiento, atendiendo al tipo de salida que poseen. De esta forma existen módulos con salidas a rele que proporcionan una corriente de hasta 3 amperios por vía. El otro tipo de módulos posee salidas a transistor, las cuales pueden entregar una corriente del orden de 500 mA. Otro de los parámetros que permite clasificar la gama de entradas y salidas del TSX Micro, es el tipo de conexión de estas vías. Atendiendo a este parámetro es posible encontrar las conexiones por bornero y las conexiones por conector tipo Telefast.

**TSX-3705****TSX-3708****TSX-3710****TSX-3721****TSX-3722***no extensible**extensible*

## BREVE INTRODUCCIÓN TEÓRICA

El autómata TSX Micro



- 1 Caja que contiene la alimentación, el procesador y la memoria base.
- 2 Orificio de fijación.
- 3 Bloque de visualización.
- 4 Toma terminal TER para programación con PL7 mediante puerto serie.
- 5 Toma auxiliar AUX para diálogo hombre máquina.
- 6 Alojamiento para tarjeta de memoria supletoria.
- 7 Tapa para la alimentación del autómata.
- 10 Alojamiento para una tarjeta de comunicaciones tipo PCMCIA.
- 11 Tapa de acceso a la pila de seguridad.
- 14 Conectores para las funciones analógicas y de contaje integradas.

El bloque de visualización central está formado a su vez por tres bloques de 32 leds cada uno. Estos pueden utilizarse para visualizar el estado de las entradas y salidas o datos y variables del autómata mediante el pulsador de diagnóstico situado en el mismo bloque de visualización.

La columna de leds situada a la derecha permite conocer el estado del procesador:

- RUN:** Run/Stop (intermitente = Stop)
- TER:** Tráfico de comunicaciones en la toma terminal
- I/O:** Error de E/S
- ERR:** Error de la aplicación (intermitente / fijo)
- BAT:** Fallo Batería

Por último remarcar el pulsador de RESET situado debajo de la toma terminal del autómata. Presionando este pulsador, es posible reiniciar la aplicación del autómata.



## BREVE INTRODUCCIÓN TEÓRICA

### Tipos de variables bajo PL7

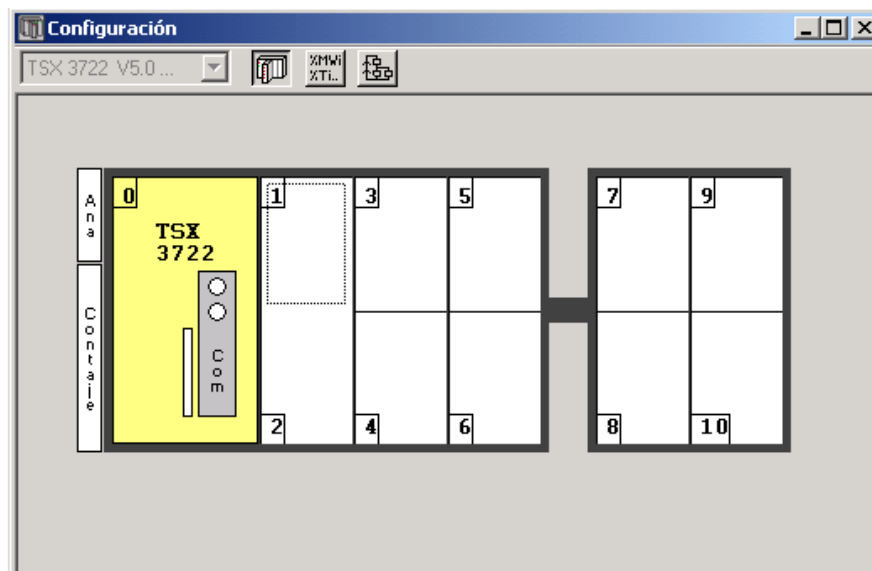
Bajo PL7, los objetos de un programa se estructuran en diferentes zonas atendiendo a su naturaleza u origen. Estas son:

- ✓ zona de **memoria** (%M)
- ✓ zona de **entradas** (%I)
- ✓ zona de **salidas** (%Q)
- ✓ zona de **constantes** (%K)
- ✓ zona de **sistema** (%S)
- ✓ zona de **bloques función** (Temporizador %TM, Contador %C, etc...)

Así mismo estos objetos definidos por zonas pueden adoptar diversas formas para obtener las variables de un programa. Las formas disponibles son:

- ✓ bit (X)
- ✓ byte (B) - 8 bits
- ✓ palabra (W) - 16 bits
- ✓ doble palabra ó word (D) - 32 bits
- ✓ palabra ó word real con coma flotante (F) - 32 bits

Finalmente, y para implementar las variables en PL7, la nomenclatura que se utiliza queda avalada por la normativa IEC-1131. En esta normativa las variables de E/S se identifican en función de la posición del módulo que ocupen físicamente dentro del autómata.



## BREVE INTRODUCCIÓN TEÓRICA

### Tipos de variables bajo PL7

- ✓ Objeto bit simple

**% M, X ó S i**

Ejemplos:

%M0: Primer bit de memoria.

%X0: Primer bit de etapa grafcet.

%S0: Bit sistema cero.

- ✓ Objetos varios

**% M,K ó S B, W, D ó F i**

Ejemplos:

%MW3: Tercera palabra de memoria.

%KW10: Décima constante.

%MD0: Doble palabra real cero.

- ✓ Objetos de entradas y salidas

**% I ó Q B, W ó D x . i**

x: Posición del módulo donde se encuentra la vía físicamente.

I: Número de vía.

Ejemplos:

%I1.0: Entrada 0 del módulo situado en la posición 1.

%Q2.7: Salida número 7 del módulo situado en la posición 2.

%IW0.1: Entrada analógica 1 integrada en el módulo 0.

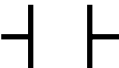






%QW0.9: Salida analógica 9 integrada en el módulo 0.

Uno de los aspectos importantes a remarcar sobre la utilización de variables consiste en que los bits de memoria %Mx, no comparten el mismo área de memoria que los bytes %MBx, las palabras %MWx ó las dobles palabras %MDx y %MFx. De esta forma, el bit %M0 no corresponde con el bit menos significativo de la palabra %MW0.

## BREVE INTRODUCCIÓN TEÓRICA

### Tipos de instrucciones

PL7Pro permite programar las aplicaciones en varios lenguajes de programación: texto estructurado, lista de instrucciones, contactos y grafcet. De entre ellos, el lenguaje que se utilizará en este manual para programar las aplicaciones será el de contactos ó LADDER (LD) debido a su facilidad de aprendizaje y a su alto grado de integración en el mundo industrial. Este tipo de lenguaje utiliza elementos gráficos y entre ellos encontramos instrucciones para bits del tipo:

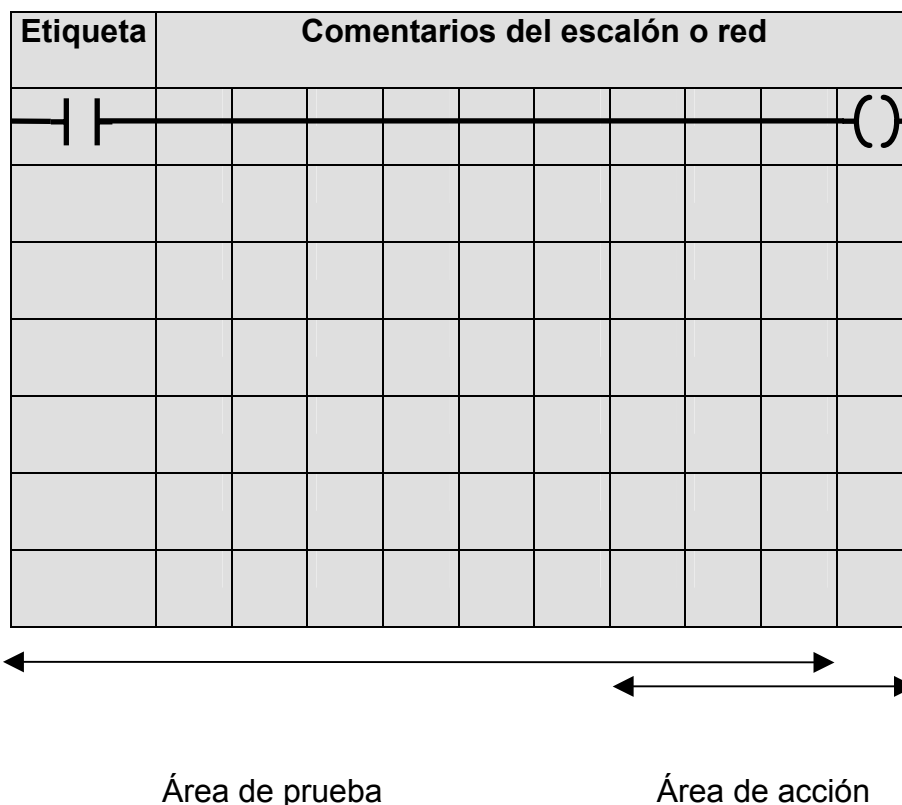
SÍMBOLO	FUNCIONES	COMENTARIO
	Contacto establecido cuando el objeto bit que lo controla está en el estado 1.	Contacto normalmente abierto.
	Contacto establecido cuando el objeto bit que lo controla está en el estado 0.	Contacto normalmente cerrado.
	Detección del paso de 0 a 1 del objeto bit que lo controla.	Activo únicamente durante un ciclo del autómata.
	Detección del paso de 1 a 0 del objeto bit que lo controla.	Activo únicamente durante un ciclo del autómata.
	El objeto bit asociado toma el resultado del área de prueba.	Activación de una bobina directa.
	El objeto bit asociado toma el resultado inverso del área de prueba.	Activación de una bobina inversa.
	El objeto bit se coloca a 1 cuando el resultado del área de prueba es 1.	Después de realizar un SET, la variable solo se desactivará realizando un RESET.
	El objeto bit se coloca a 0 cuando el resultado del área de prueba es 1.	Después de realizar un RESET, la variable solo se desactivará realizando un SET.
	Propuesta en lenguaje grafcet, utilizada para programar las receptividades asociadas a transiciones.	Permite pasar a la etapa siguiente.

## BREVE INTRODUCCIÓN TEÓRICA

## Estructura de una red

En la realización de un programa, PL7 permite subdividir este en bloques o secciones para obtener una mayor modularidad. A su vez, cada sección esta estructurada en escalones de redes de contactos formadas por 7 líneas y 11 columnas donde insertaremos todas las instrucciones de nuestro programa y los comentarios necesarios.

### SECCIÓN 1



Mientras que el área de prueba contendrá todas las instrucciones de consulta a una variable, en el área de acción situaremos las acciones como SET, RESET, etc.

Las etiquetas de una red se utilizan para identificar un conjunto de instrucciones en el caso de realizar saltos dentro de un mismo programa. Estas etiquetas se identifican como %Ln, siendo n un número determinado. La tecla Esc (escape) permite salir de la edición de una etiqueta.

## PRÁCTICA 1

### CONFIGURACIÓN BÁSICA DE UNA APLICACIÓN

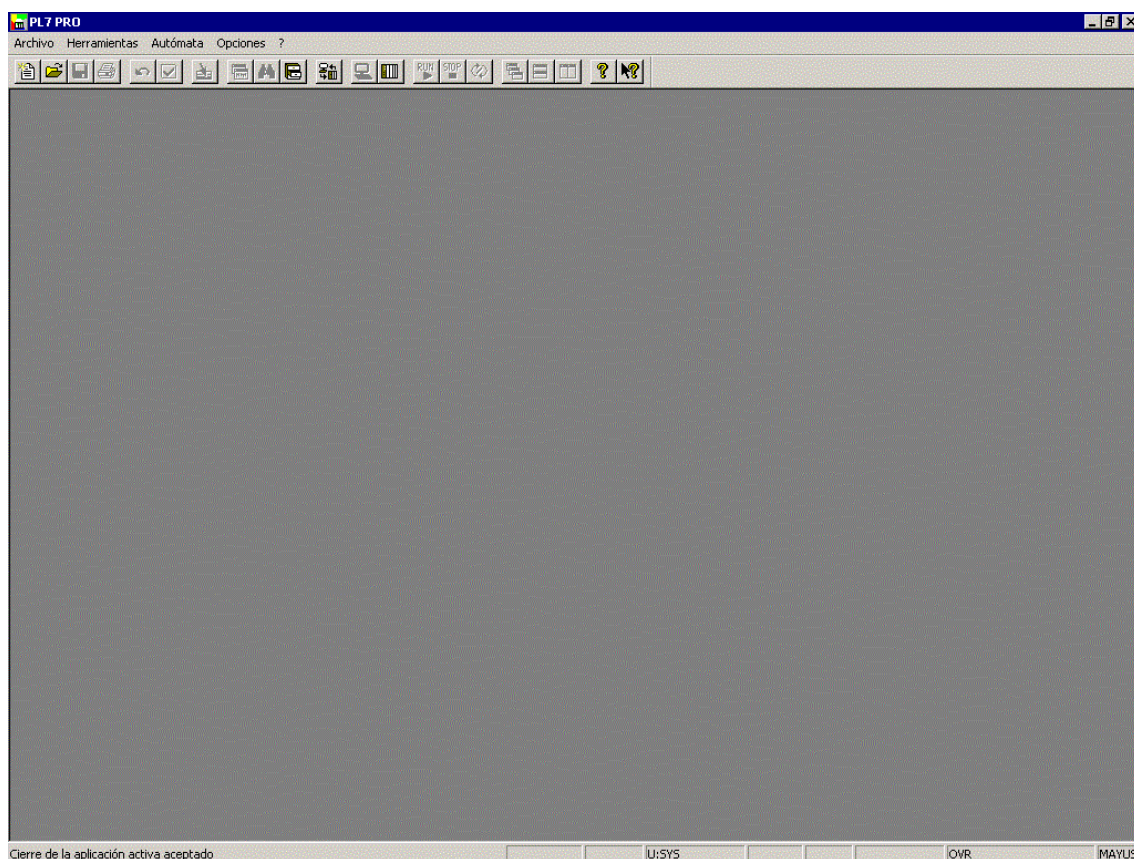
#### OBJETIVO

En esta primera práctica se realizará una configuración básica del autómata TSX3722 con las opciones instaladas. Esta, nos permitirá obtener un fichero origen a partir del cual se realizarán los programas de cada práctica.

#### PRESENTACIÓN

Antes de realizar un programa es necesario configurar una serie de parámetros relacionados con las características hardware del autómata a utilizar. El tipo de autómata, el número de tarjetas de entradas y salidas, así como la configuración de las diferentes variables y bloques función, serán parámetros a controlar en la aplicación.

En esta ocasión configuraremos el autómata TSX3722001, con un módulo de entradas y salidas TSXDMZ28DTK, sin tarjeta de memoria y sin graficet. Respecto a la configuración del software (temporizadores, contadores, monoestables, etc.) se utilizarán las que PL7 marca por defecto.



## DESARROLLO PRÁCTICA 1

Para entrar en el programa PL7 buscaremos el acceso directo situado en el escritorio de windows o, en su defecto, la opción PL7 Pro V 4.x desde el menú **Inicio->Programas->Modicon Telemecanique->PL7 Pro V4.3**.

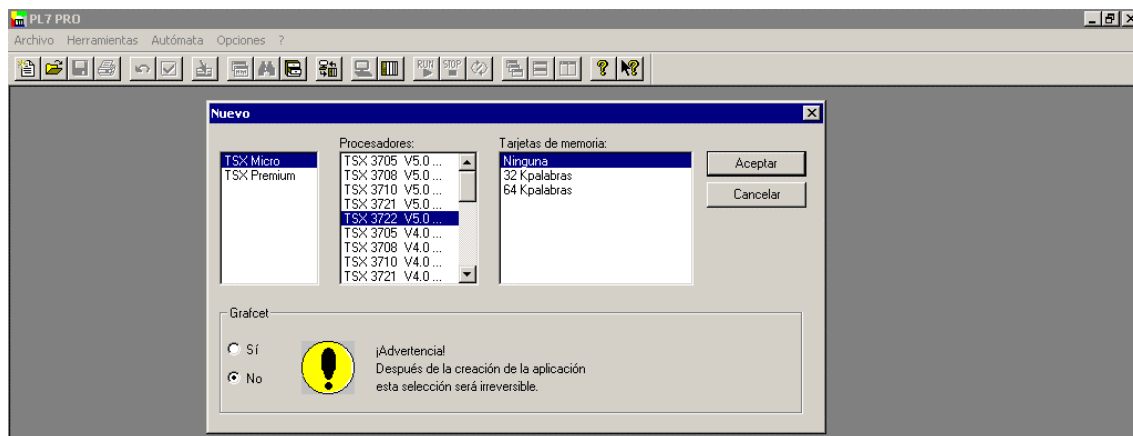
Una vez se accede a la pantalla principal y desde el menú **Archivo**, hacer “clic” en **Nuevo** y seleccionar las opciones:

Autómata: TSX Micro

Procesador : TSX 3722 V5.0...(ver tapa de alimentación del autómata)

Memoria: Ninguna

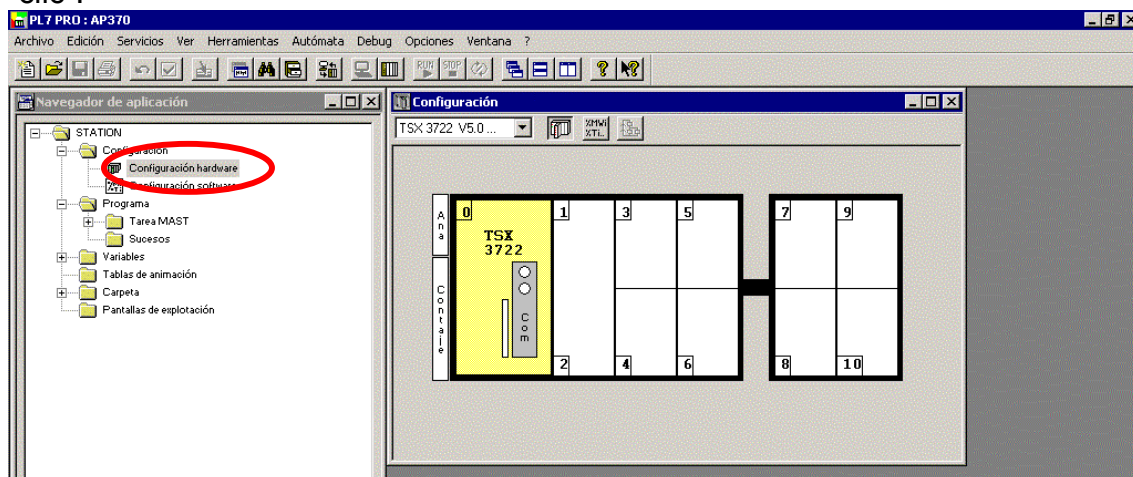
Grafcet: No



Aceptar la configuración para que el programa genere automáticamente la nueva aplicación.

En la parte izquierda de la pantalla aparece el navegador de la aplicación. Esta pantalla contendrá, estructuradas en carpetas, todas las opciones ó parámetros necesarios para configurar nuestra aplicación.

El paso siguiente en la resolución de esta práctica consiste en configurar las opciones hardware de nuestro autómata. Para ello, entraremos en la opción **Configuración -> Configuración hardware** del navegador mediante un doble “clic”.

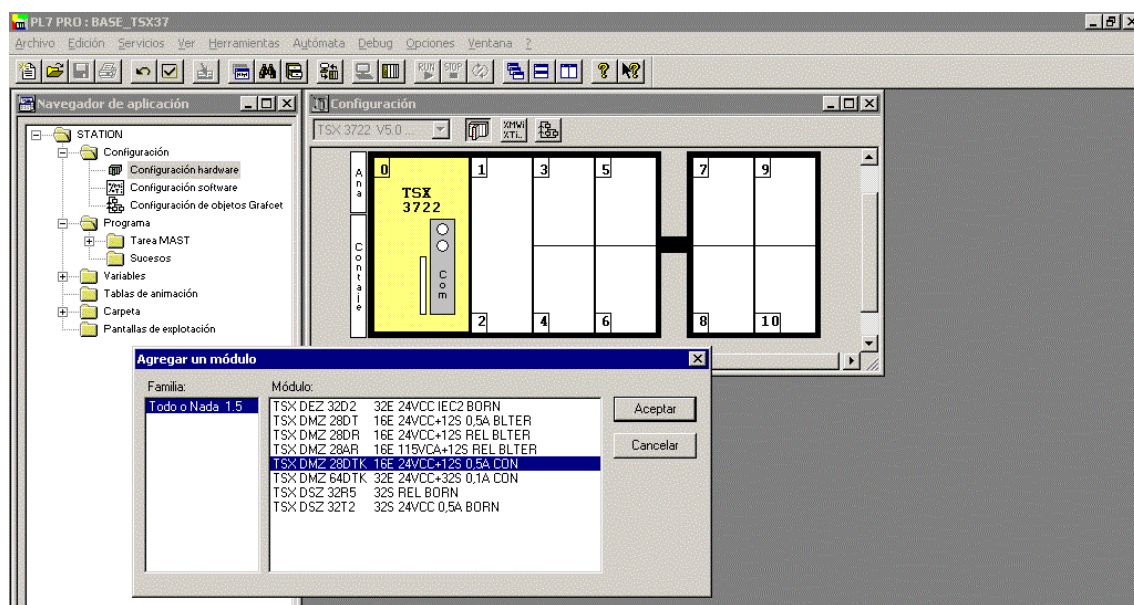




## DESARROLLO PRÁCTICA 1


La pantalla que aparece representa la disposición física de los diferentes elementos que conforman nuestro autómatas: conectores, CPU y posiciones donde colocar las diversas opciones. Mientras que en la posición 1 y 2 podremos configurar un módulo de entradas y salidas digitales, en las posiciones 3,4,5 y 6 podremos configurar cualquiera de las tarjetas de entradas y salidas, de conteo, de comunicación, etc. Las posiciones 7,8,9 y 10 no se utilizarán en nuestro caso dado que no disponemos del rack de ampliación.

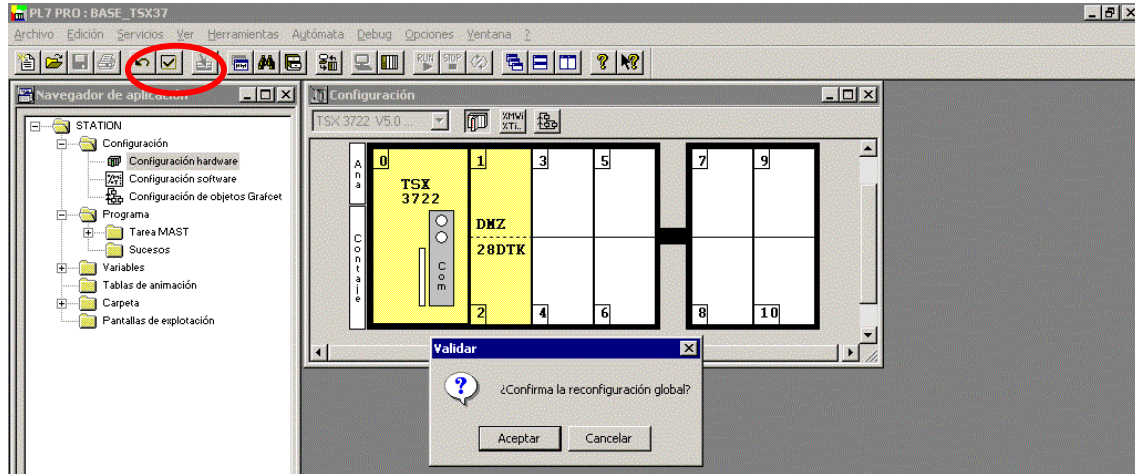
Para configurar el hardware deberemos hacer doble “clic” en la posición correspondiente y seleccionar el módulo de entradas y salidas del que disponemos. Podremos ver la referencia de esta tarjeta en la parte frontal de la misma. En nuestro caso, esta referencia será TSXDMZ28DTK.



Una vez hemos configurado el archivo base, solo queda validar la configuración y guardar el archivo con un nombre.

## DESARROLLO PRÁCTICA 1

Para validar la configuración pulsaremos el icono de **Validación** , situado en la barra del menú superior. La opción **Guardar como** del menú **Archivo**, nos permitirá dar un nombre a la aplicación y salvarla.



Configurados los parámetros básicos y, siguiendo esta metodología de trabajo, es posible editar las propiedades de la CPU, los módulos de entradas y salidas, las vías analógicas, las vías de conteo ó los parámetros del software mediante esta misma ventana de configuración. En nuestro caso, todos estos parámetros no han de modificarse en esta práctica ya que utilizaremos los mismos que PL7 configura por defecto.



## PRÁCTICA 2

### CONTROL DE UN EXTRACTOR

#### OBJETIVO

Para esta segunda práctica se desea controlar el accionamiento de un extractor de humos que permite la ventilación de unas instalaciones. En su desarrollo se utilizaran instrucciones de enclavamiento (set y reset).

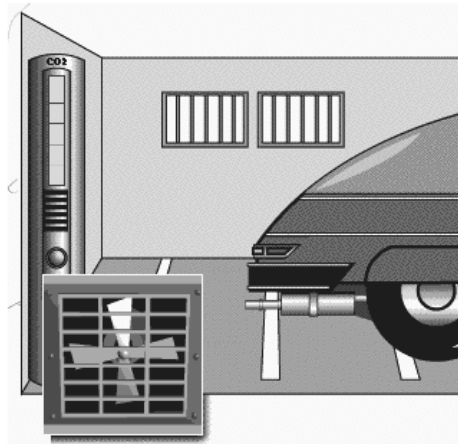
#### PRESENTACIÓN

A partir de la configuración básica de nuestro autómatas, se desea controlar un extractor de humos mediante un pulsador de marcha y otro de paro normalmente cerrado (NC). Para garantizar la seguridad, el paro ha de ser prioritario frente a la marcha. Esto implica que en el caso de pulsar ambos, el motor no ha de ponerse en marcha. Además, será necesario una confirmación de marcha para volver a funcionar.

#### Lista de variables

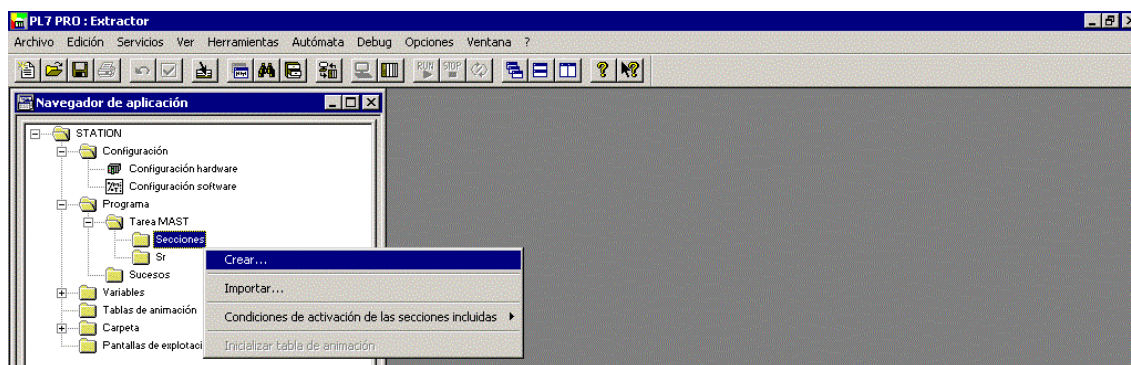
Pulsador de marcha	%I1.0
Pulsador de paro (NC)	%I1.1
Contactador del motor	%Q2.0

#### Esquema de la instalación

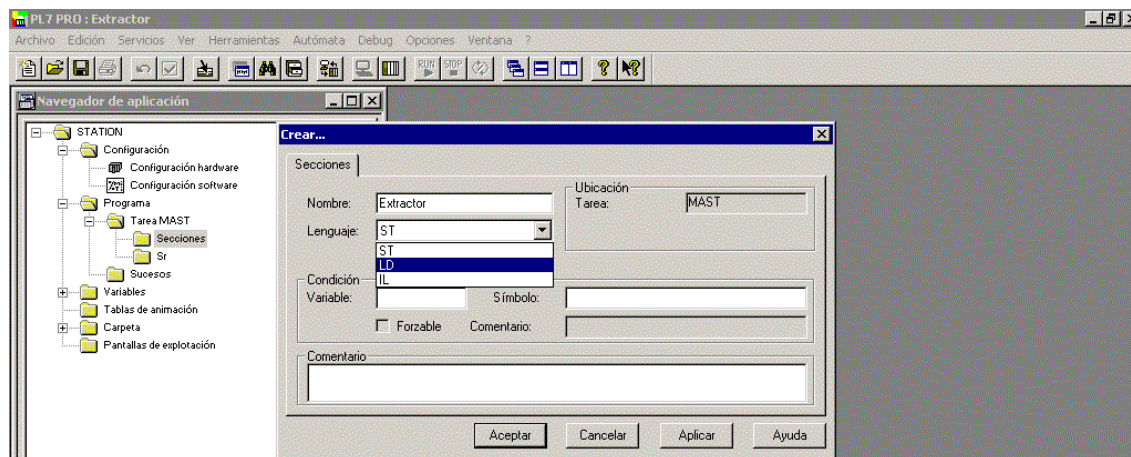


## DESARROLLO PRÁCTICA 2

A partir de la configuración básica creada en la primera práctica, añadiremos una sección nueva donde se va a escribir el programa. Podemos encontrar esta opción dentro del navegador de la aplicación y concretamente en la carpeta **Programa->Tarea Mast->Secciones**. Con el botón derecho del ratón hacer "clic" en la opción **Crear**.



En la ventana que aparece, asignaremos un nombre a la nueva sección o página en blanco que se va a crear y se definirá el lenguaje que más convenga para la aplicación. En este caso, se indicará la opción LD o LADDER dado que se programará en contactos.

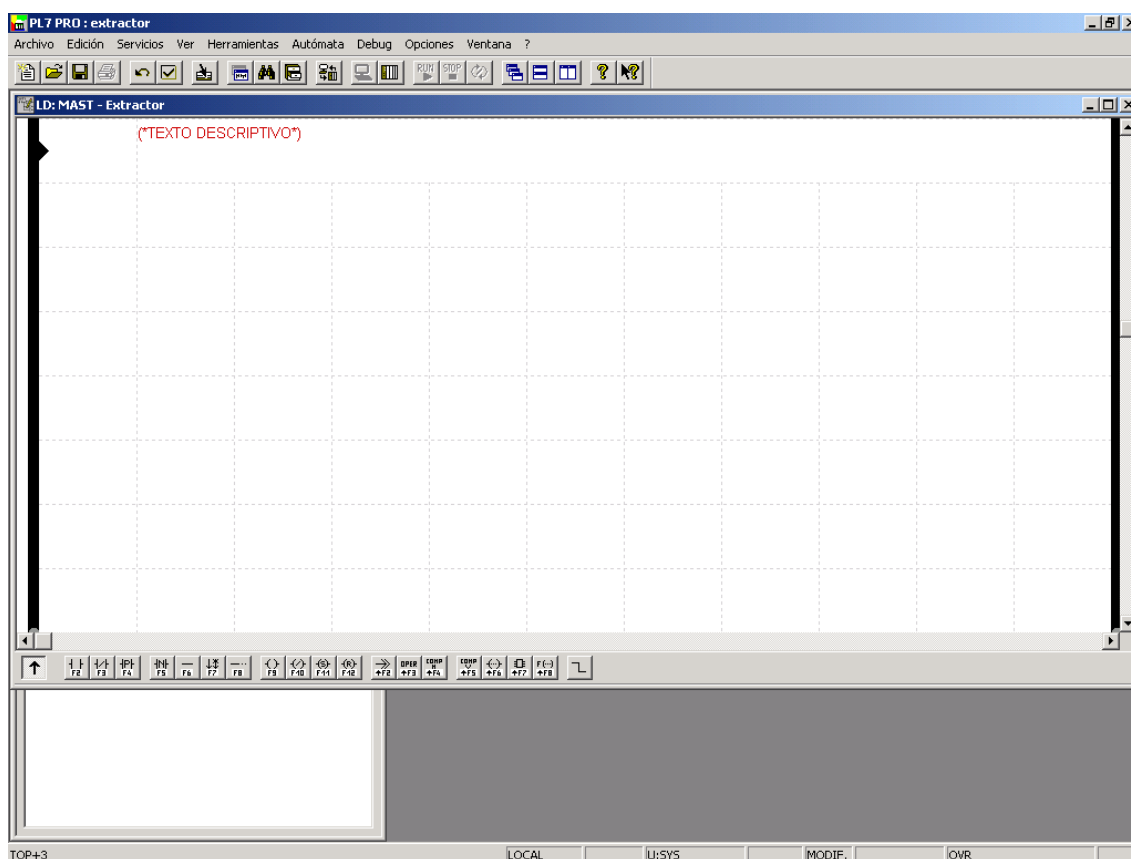


El espacio para **Condición variable** permite escribir un bit que habilitará o deshabilitará esta sección en función de su valor. Si su valor es uno la sección se ejecutará normalmente, si es cero, esta sección no se ejecutará y todas las instrucciones que se hayan escrito dentro no se realizarán. Por el momento, no se utilizará ninguna condición variable.

## DESARROLLO PRÁCTICA 2

La pantalla que aparece es la nueva sección donde se escribirá el programa del autómatas. Tal y como se había comentado en la introducción teórica, podemos ver en la parte superior un espacio para la introducción de un texto descriptivo. A su izquierda, existe una casilla que permite identificar el escalón mediante una etiqueta del tipo %L. Los espacios siguientes permiten colocar todas las instrucciones programadas en contactos, con un total de 7 filas y 11 columnas.

Uno de los aspectos a tener en cuenta al programar un escalón es que mientras se esta editando, las instrucciones colocadas quedan resaltadas en color rojo. También es posible identificar que un escalón se esta editando mediante el color oscuro que adoptan los bordes laterales del mismo escalón. Es importante tener este punto en cuenta dado que solo es posible editar un escalón a la vez. Para salir de la edición de un escalón pulsaremos la tecla enter.

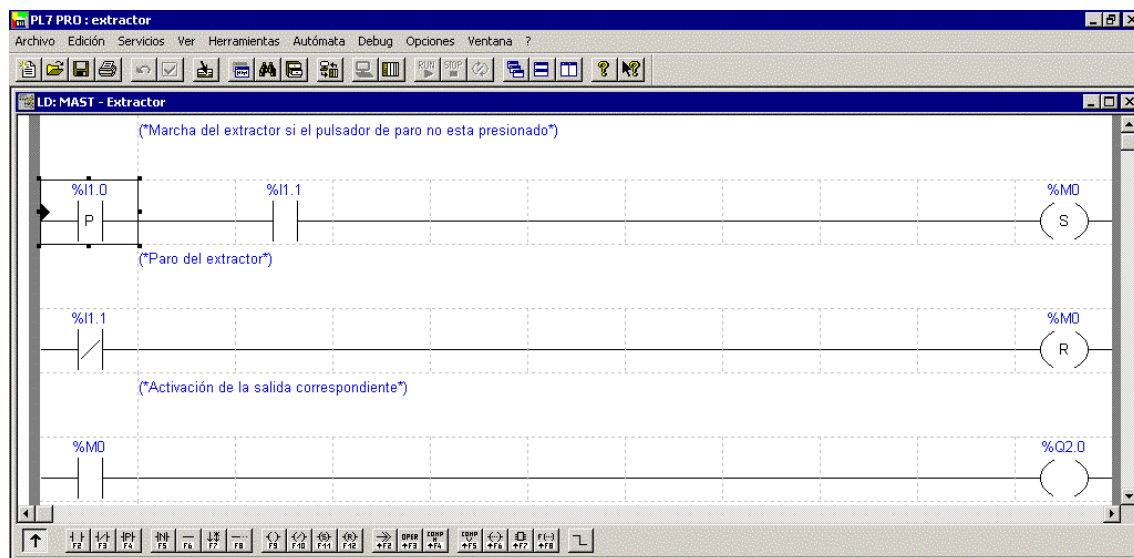


Las instrucciones necesarias para el desarrollo de nuestro programa, podemos encontrarlas en la parte inferior de la sección. Instrucciones de consulta (contacto normalmente abierto y cerrado y flancos), instrucciones de asignación (bobinas directas, set, reset, etc) y enlaces entre elementos pueden utilizarse mediante un “clic” sobre el icono correspondiente. Esta misma forma de actuar se utilizará para la colocación de bloques función del tipo temporizador o contador.

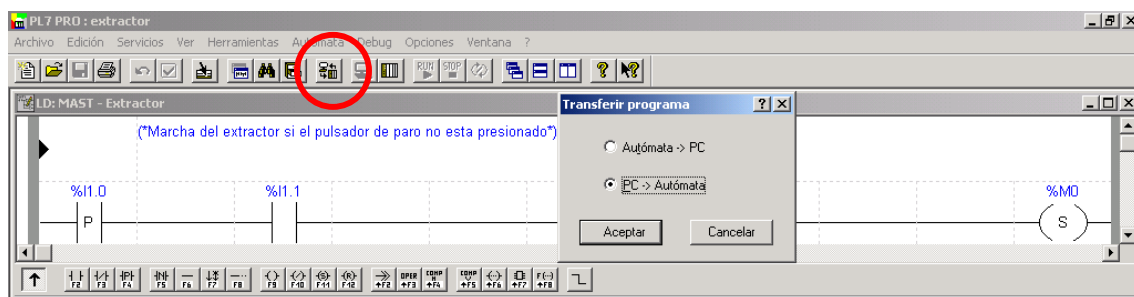
## DESARROLLO PRÁCTICA 2

En la siguiente figura se ha realizado una de las soluciones posibles al problema. Ya que disponemos de pulsadores, se han utilizado bobinas de enclavamiento (set y reset) que actúan sobre un bit manteniendo la salida.


Para dar prioridad al paro, que es normalmente cerrado físicamente, se ha programado la marcha con un flanco positivo.



Una vez definido nuestro programa, podremos transferirlo al autómata mediante la opción **Transferir programa** del menú **Autómata**, indicando la opción **PC->Autómata** a la pregunta que nos realiza el sistema.



Cuando el programa haya sido transferido (podemos ver el proceso en la parte inferior izquierda de la pantalla), podremos conectar el PC y ejecutar la aplicación en línea mediante la opción **Conectar**, del menú **Autómata**.

Para probar nuestra práctica, sólo queda ejecutar el programa enviando una instrucción de RUN mediante la opción **RUN...** del menú **Autómata** o el icono correspondiente .

## PRÁCTICA 3

### ARRANQUE DIRECTO DE UN MOTOR TRIFÁSICO DE INDUCCIÓN

#### OBJETIVO

Realizar el automatismo para el arranque directo de un motor trifásico de inducción mediante pulsadores de paro y marcha. Realizar además, la señalización del estado de marcha.

#### PRESENTACIÓN

Al accionar el pulsador de marcha se activará la salida que excita el contactor de línea, quedando el motor en funcionamiento. Si se acciona el pulsador de paro, la maniobra se abortará. Un relé de protección térmica del motor, asociado a una entrada del autómatas, debe interrumpir también la maniobra al activarse. Se señalizarán mediante pilotos luminosos los estados de marcha, paro y activación del relé térmico.

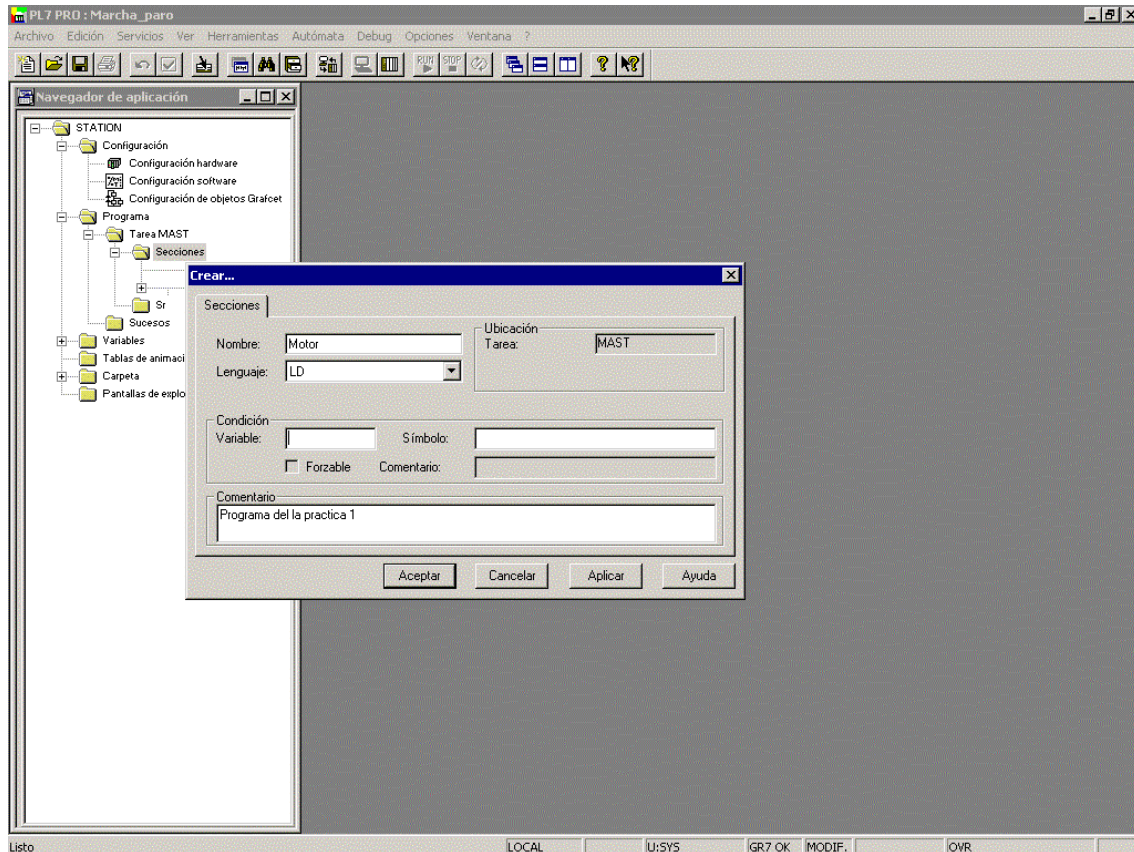
#### Lista de variables:

Pulsador de paro (NC)	%I1.0
Contacto del relé térmico (NC)	%I1.1
Pulsador de marcha	%I1.2
Contactor motor	%Q2.0
Piloto motor STOP	%Q2.1
Piloto motor en marcha	%Q2.2
Piloto de defecto térmico	%Q2.3

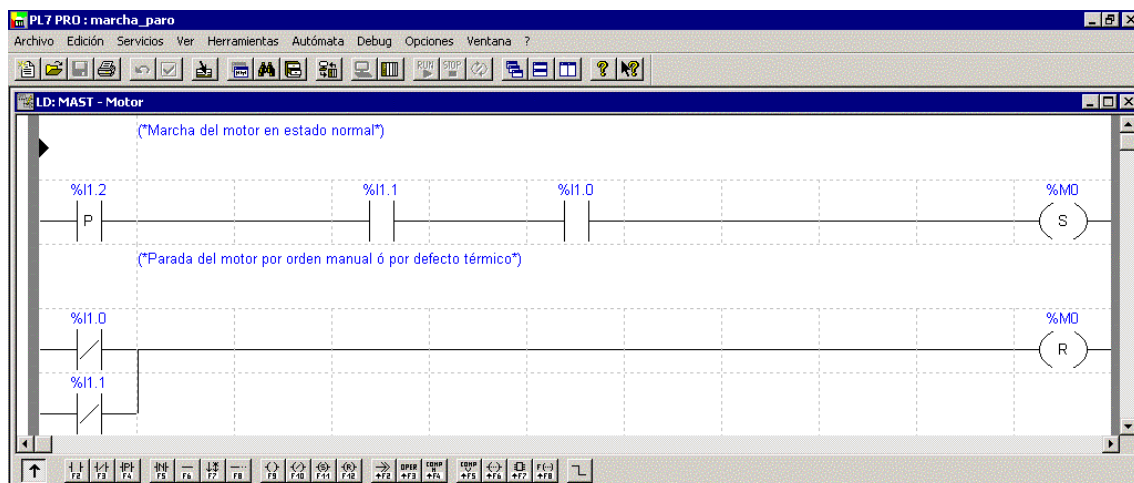


## DESARROLLO PRÁCTICA 3

A partir de la configuración básica realizada en la práctica 1 crearemos una sección o página en blanco donde escribir nuestro programa. Para ello, debemos situarnos en la carpeta **Programa -> Tarea MAST -> Secciones**, dentro del navegador de la aplicación. Con el botón derecho del ratón haremos “clic” sobre esta misma carpeta y seleccionaremos la opción **Crear**.



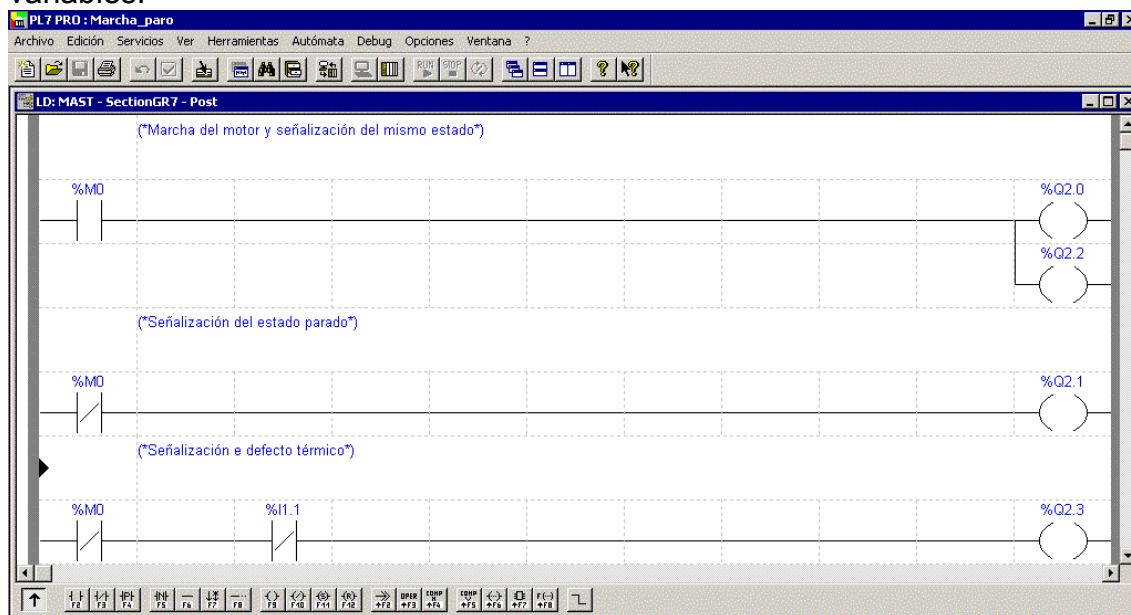
Dado que el funcionamiento del sistema es muy similar al de la práctica anterior, se ha utilizado la misma estructura añadiendo la señal de emergencia. El bit %M0, permite mantener la salida al motor y no forzarla directamente.



## DESARROLLO PRÁCTICA 3

Dado que la activación de las salidas son mantenidas y no por SET ó RESET, es importante no repetir instrucciones con la misma variable dentro de un programa. En caso de repetir instrucciones de asignación sin realizar un SET ó un RESET, cabe la posibilidad de que el programa no se ejecute adecuadamente.

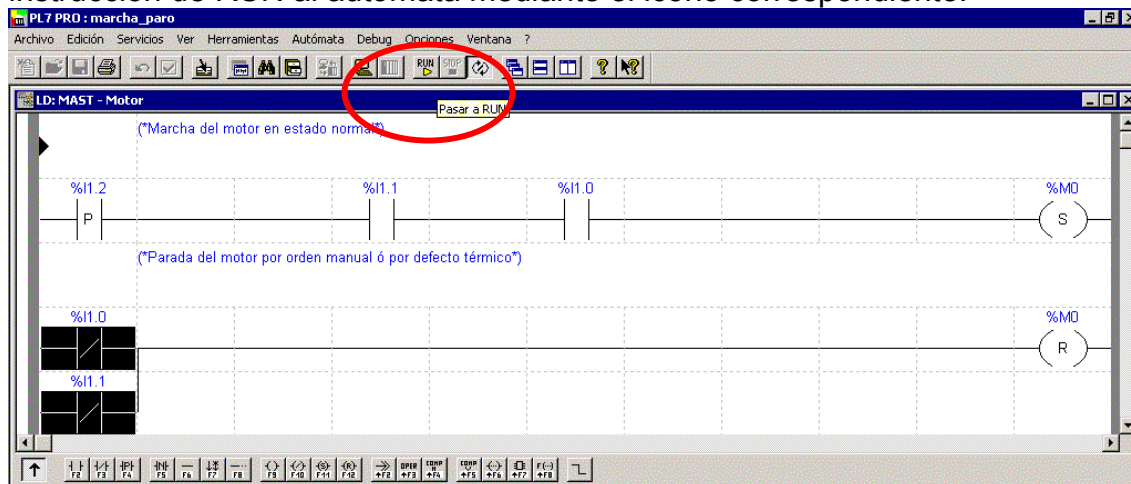
En esta sección vemos que cuando el bit de marcha está activo, se activa el motor y su correspondiente señalización. Lo mismo ocurre con el resto de variables.



Una vez definido nuestro programa, podremos transferirlo al autómata mediante la opción **Transferir programa** del menú **Autómata**, indicando la opción **PC->Autómata** a la pregunta que nos realiza el sistema.

Cuando el programa haya sido transferido (podemos ver el proceso en la parte inferior izquierda de la pantalla), podremos conectar el PC para ver la aplicación en línea mediante la opción **Conectar**, del menú **Autómata**.

Para probar nuestra práctica, sólo queda ejecutar el programa enviando una instrucción de RUN al autómata mediante el icono correspondiente.







## PRÁCTICA 4

### SELECCIÓN DE CAJAS

#### OBJETIVO

Realizar el automatismo que permite clasificar unas cajas de medidas diferentes en una línea de producción.

#### PRESENTACIÓN

Una vez se active el clasificador mediante la orden de marcha, las cajas llegarán por la cinta transportadora. Al final del primer tramo existen dos células fotoeléctricas que permiten conocer la altura de las cajas dado que están situadas a distancias diferentes. Si se activan los sensores A y B, el sistema nos estará indicando que la caja es grande. Si únicamente se activa el sensor B, la caja a clasificar será pequeña. El orden de activación será A, si se presenta la ocasión, y después B. Además, el sistema nos asegura que sólo llegará una caja cuando la última haya sido clasificada.

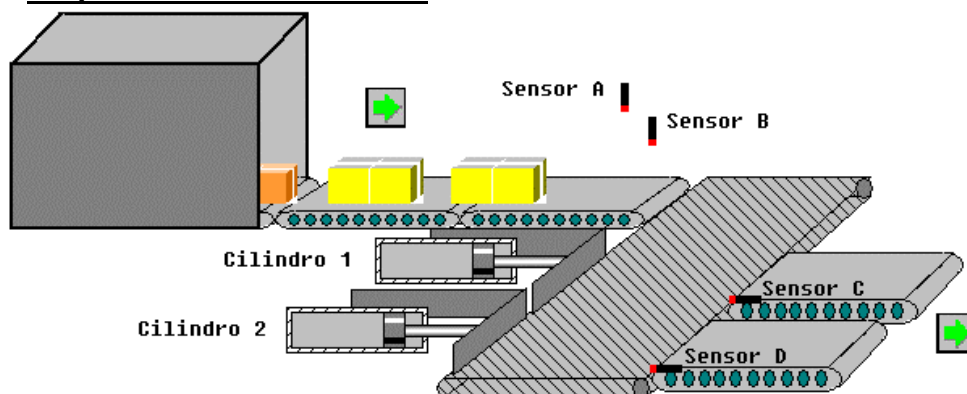
Al llegar a la zona de clasificación se activará el cilindro empujador 1 ó 2, ambos con válvulas monoestables, mediante los sensores C ó D respectivamente. Si la caja es grande se activará el empujador 1 y si es pequeña el 2.

Se pide, además, la utilización de símbolos para la identificación de variables.

#### Lista de variables

Marcha	%I1.0
Paro (NC)	%I1.1
Emergencia (NC)	%I1.2
Sensor A	%I1.3
Sensor B	%I1.4
Sensor C	%I1.5
Sensor D	%I1.6
Cilindro 1 fuera	%I1.7
Cilindro 2 fuera	%I1.8
Motor cinta	%Q2.0
Cilindro 1	%Q2.1
Cilindro 2	%Q2.2

#### Esquema de la instalación



## DESARROLLO PRÁCTICA 4

Dado que el número de variables que intervienen en el proceso es sensiblemente superior a la práctica anterior, se ha pensado en utilizar símbolos para describir los objetos que intervienen en este ejercicio. Estos símbolos permiten al programador identificar más rápidamente las variables para realizar los cambios oportunos en el mantenimiento de los programas. Las reglas para la creación de estos símbolos son:

### Tipo de caracteres

Alfabéticos en mayúsculas

alfabéticos en minúsculas

letras acentuadas

digitales

Especial

### Descripción

de la A a la Z y letras siguientes "ÀÁÊÏÎÏDÑÒÓÔØÙÚÿp"

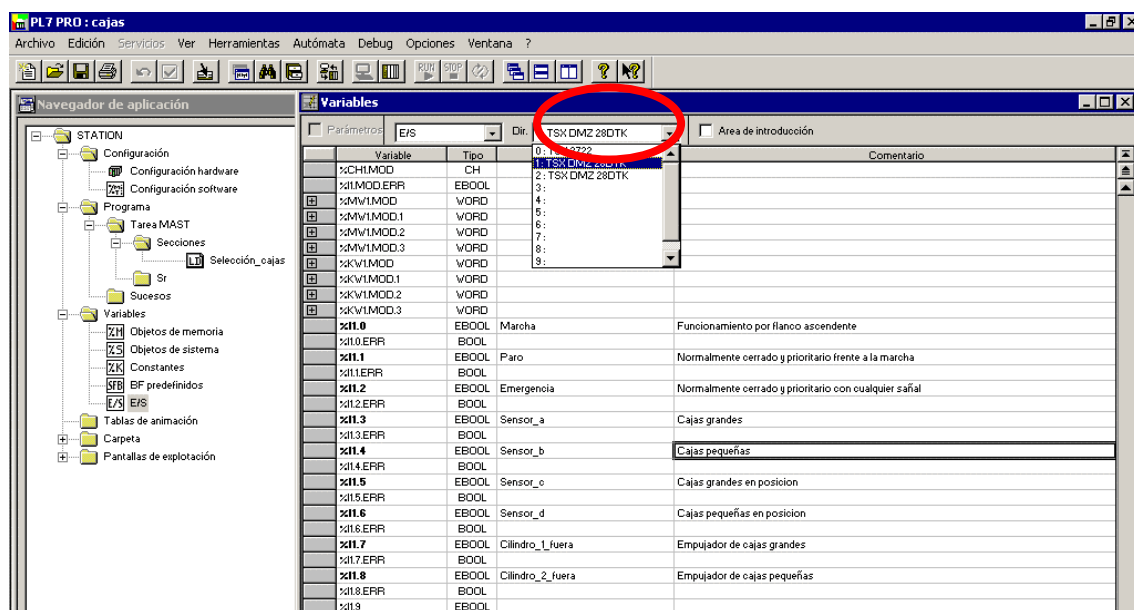
de la a a la z

àáâäæçèéêëìíîïñðóôõöøùúüÿþÿ

cifras de 0 a 9 (no pueden colocarse al principio del símbolo).

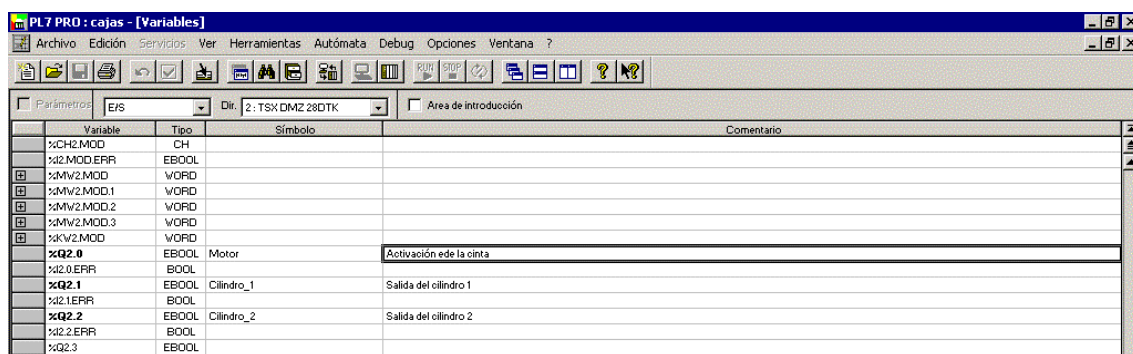
El lenguaje se reserva algunas palabras ( exit, stop, etc.) que no pueden usarse como símbolos.

Para definir esta serie de símbolos entraremos en la carpeta **Variables**, dentro del navegador de la aplicación. Seleccionando la lista donde se encuentran las variables a simbolizar aparecerá la siguiente pantalla. En nuestro caso se escogerá la opción de **E/S**.

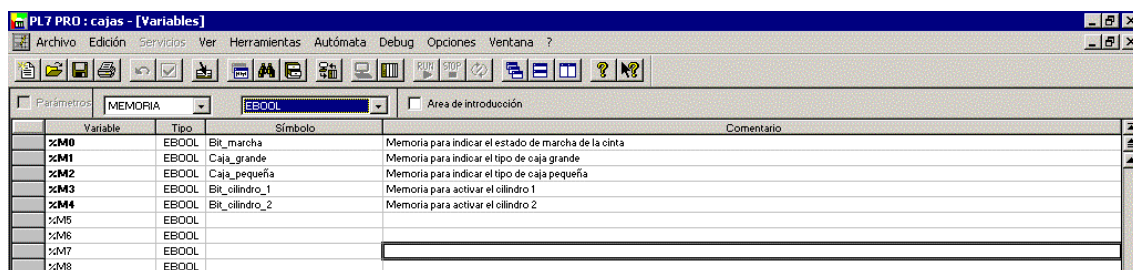


El menú desplegable situado a la izquierda de la ventana **Variables** contiene la lista con todos los tipos de objetos (E/S, memoria, constantes, etc) y a su derecha otro con el tipo de variable (bit, word, etc). Mientras que para el caso de las entradas y salidas tendremos que seleccionar la ubicación física de la variable dentro del autómata (número situado a la izquierda de la referencia), para el caso de las memorias deberemos seleccionar el tipo de variable (bit, palabra, doble palabra, etc.).

## DESARROLLO PRÁCTICA 4

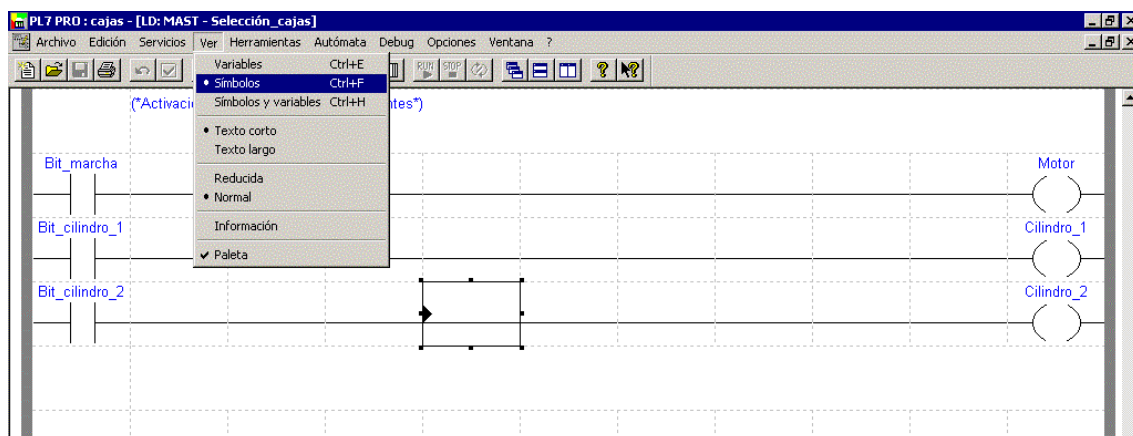


Una vez definidos los símbolos para las entradas y salidas, y dado que se ha planteado la utilización de bits para mantener estas últimas, se definen los símbolos para estas variables.



Un detalle sobre estas tablas consiste en que sólo las variables utilizadas en el programa quedan resaltadas en negrita.

Una vez definidos todos los símbolos que se utilizarán en nuestro programa, podremos alternar su visualización con la opción **Símbolos**, del menú **Ver**.



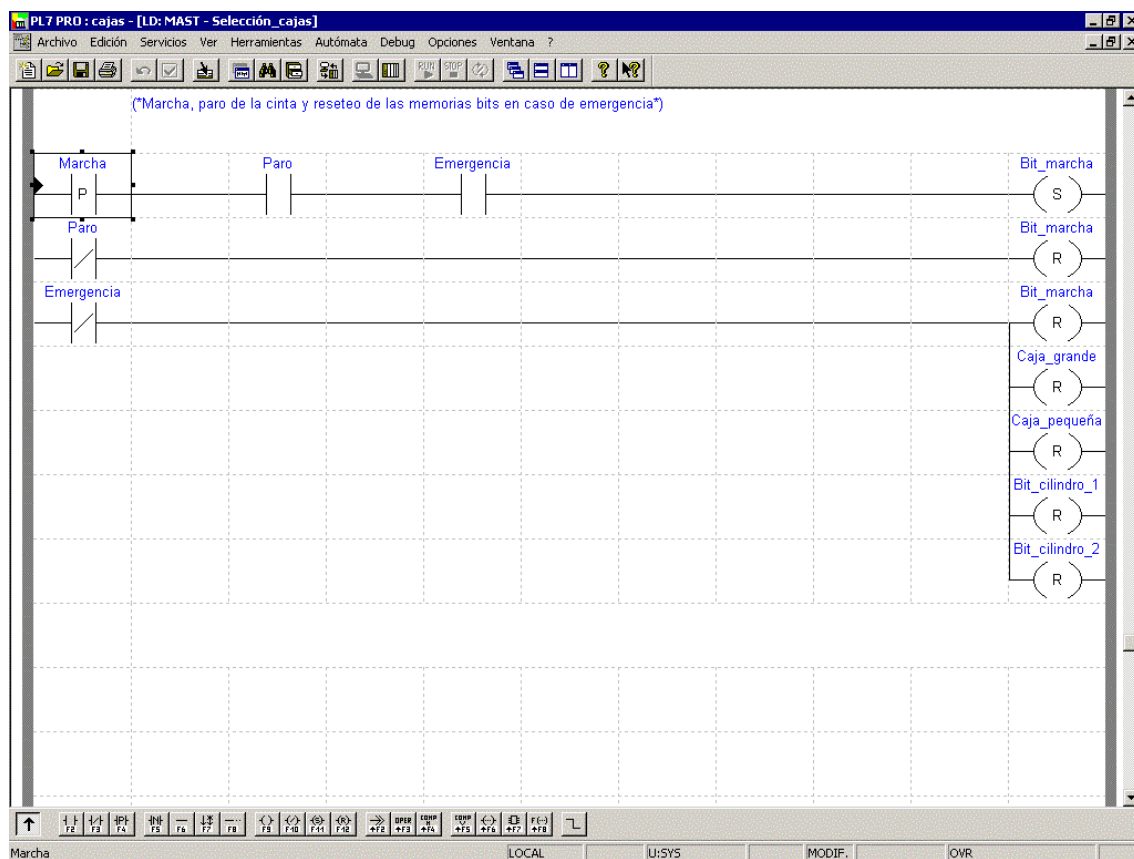
## DESARROLLO PRÁCTICA 4

Para la resolución de este ejercicio, se a estructurado el programa en tres partes:

1. Control de cinta y reset de variables intermedias
2. Cuerpo del programa
3. Activación de salidas

Tal y como se observa en la figura, la condición de marcha se activa por un flanco del mismo pulsador estando las señales de paro y emergencia activas. Programado de esta forma, se evita que la cinta entre en marcha directamente al volver de una emergencia en caso de que la marcha quedase enclavada.

La señal de emergencia desactiva todos los bits intermedios colocando el sistema en condiciones iniciales.



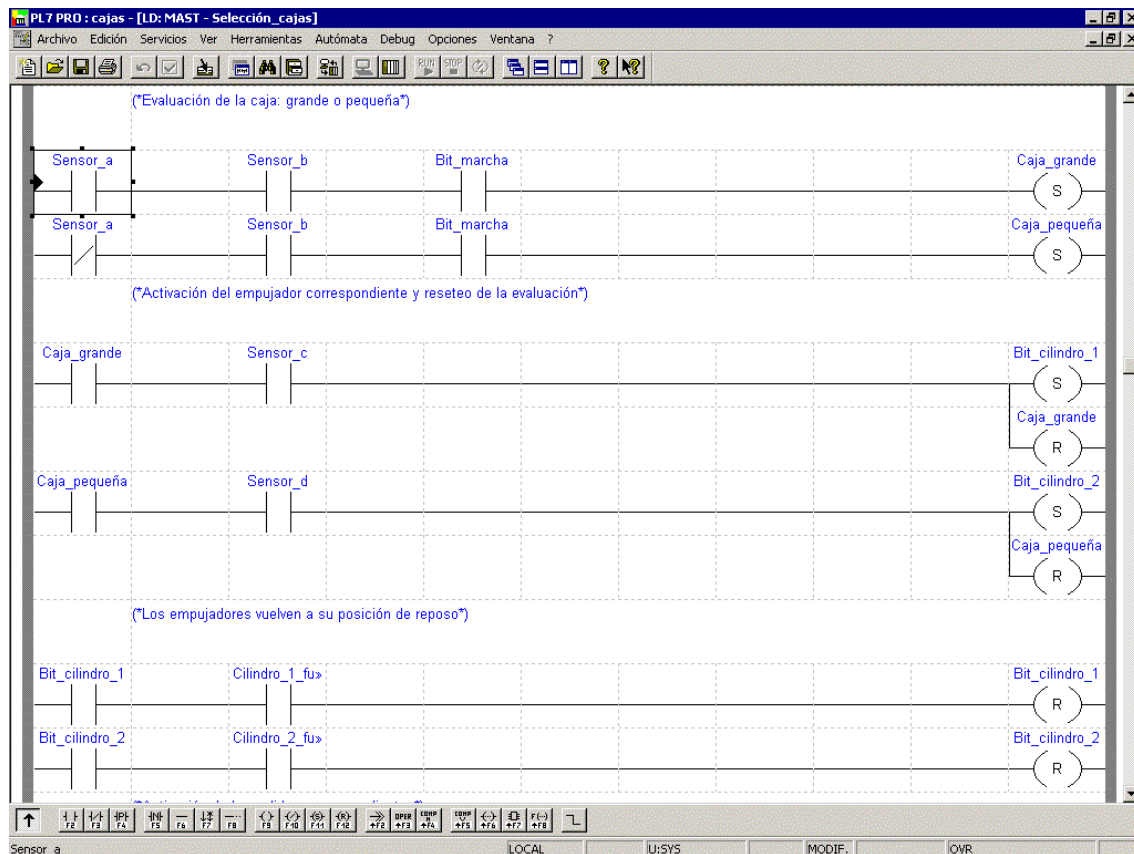
## DESARROLLO PRÁCTICA 4

El cuerpo del programa permite activar o desactivar los bits intermedios (salidas indirectas en su mayoría) en función de los sensores de entrada.

En el primer escalón vemos que, estando la cinta en marcha, el sistema puede interpretar si la caja es pequeña ó grande en función de los sensores que se activen. Es importante recordar aquí el orden de activación de estos sensores mencionado en el enunciado. En caso de que B se activase primero, el sistema consideraría que la caja es siempre pequeña.

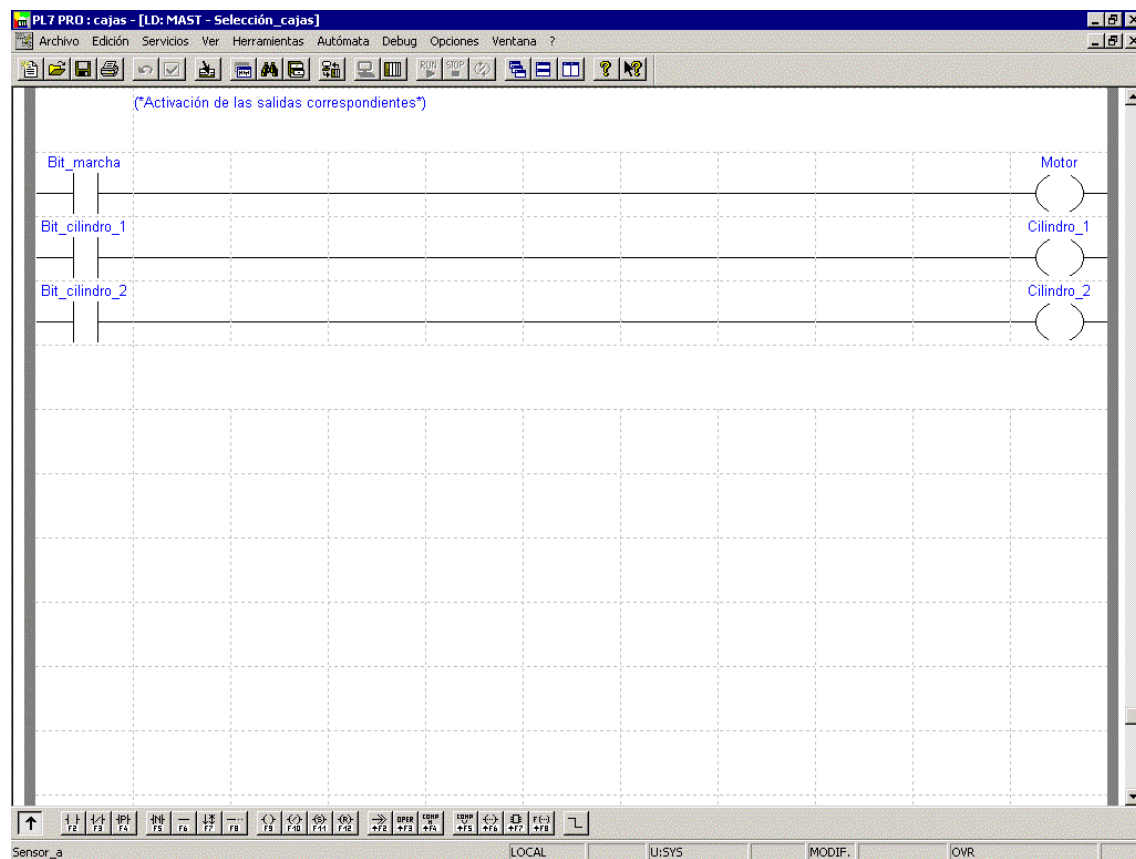
El segundo escalón permite activar el cilindro correspondiente una vez la caja llegue a la posición deseada. Es en ésta etapa del programa en la que se resetean los bits que indican el tamaño de la caja.

Finalmente desactivaremos cualquiera de los dos cilindros, con válvulas monoestables, cuando éstos lleguen al final de su carrera.



## DESARROLLO PRÁCTICA 4

La activación de las salidas corresponde a la última parte del programa. Con la finalidad de no hacer un reset ó set sobre las salidas directamente, se utilizan bit intermedios que mantienen la salida al igual que en las prácticas anteriores.



## PRÁCTICA 5

### MANDO DE UNA ESCALERA MECÁNICA

#### OBJETIVO

Realizar el automatismo para el control de una escalera mecánica mediante pulsadores de servicio, células fotoeléctricas y temporizadores.

#### PRESENTACIÓN

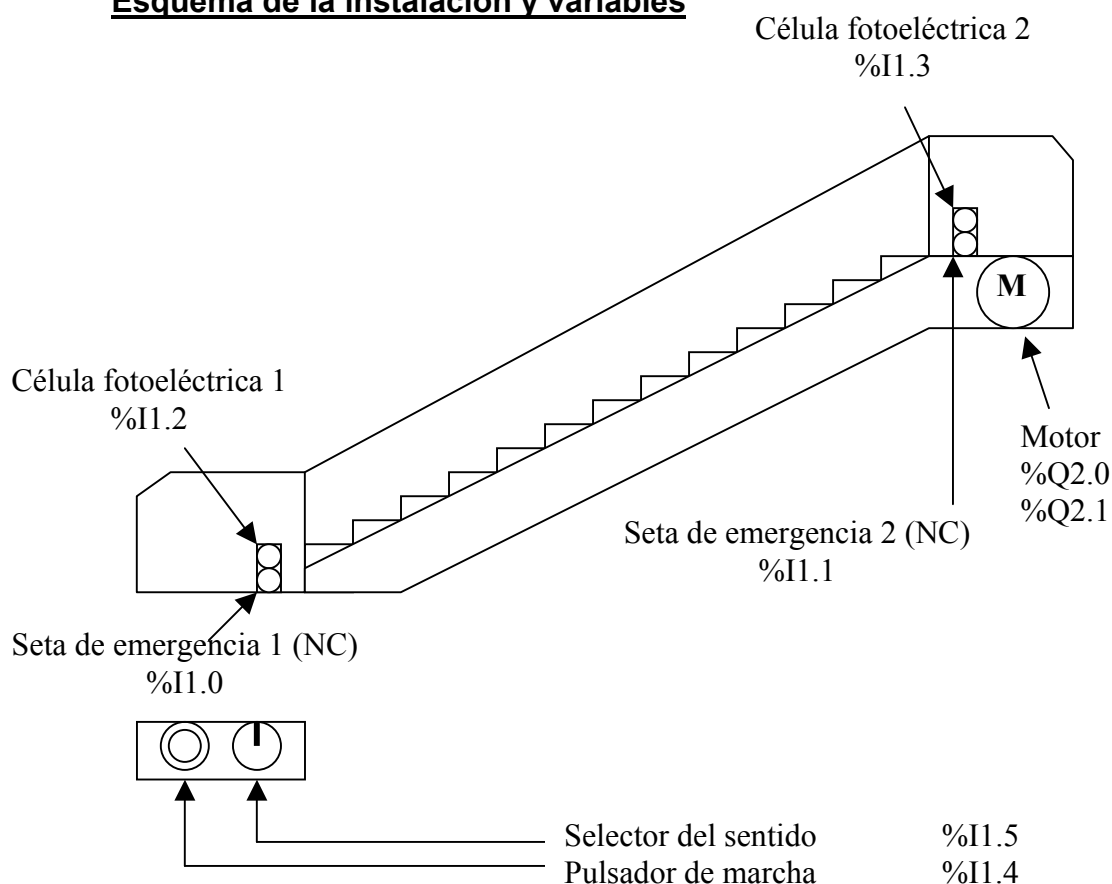
Para el control de esta escalera contaremos con un panel de mando compuesto por un pulsador de marcha y un interruptor que determinará el sentido de la escalera. También dispondremos de un pulsador de emergencia situado en cada extremo de la escalera.

Estando el automatismo en marcha, la cinta se pondrá en funcionamiento si la célula fotoeléctrica correspondiente al sentido seleccionado detecta la presencia de personas. Cada vez que se detecte una persona la cinta estará en movimiento durante 20 segundos en el sentido correspondiente.

El sentido ascendente de la escalera vendrá determinado por el valor uno del selector y actuará sobre la primera salida. El sentido descendente lo determinará el valor cero del selector y actuará sobre la segunda salida.

La seta de emergencia parará el motor de la escalera y se requerirá una confirmación de la marcha para continuar. El cambio de sentido de la escalera se realizará estando parada y con una confirmación de marcha.

#### Esquema de la instalación y variables



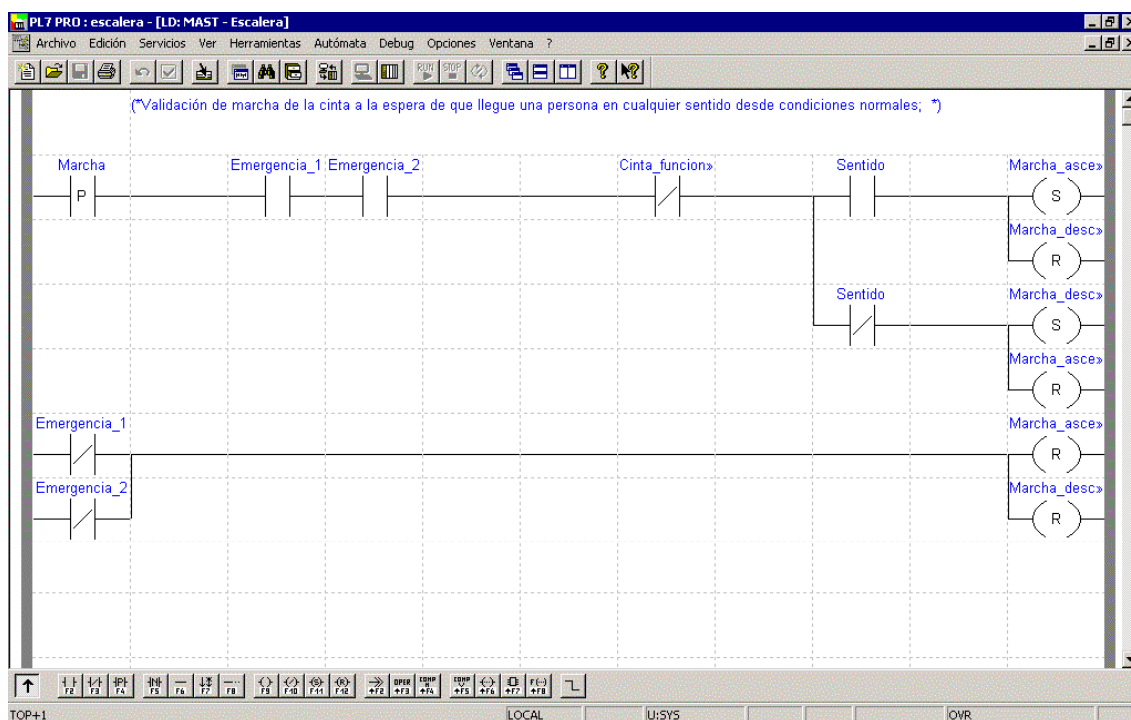


## DESARROLLO PRÁCTICA 5

Tal y como se ha descrito en las especificaciones, la escalera mecánica ha de ser capaz de transportar personas en un sentido y otro. Para ello, disponemos de dos setas de emergencias y dos fotocélulas situadas en los extremos de la escalera además de los mandos de servicio.

En la realización de este programa se han utilizado dos bits de memoria los cuales indicarán el sentido activo en cada momento. Estos bits no podrán cambiar su valor en cualquier momento y, de hecho, para cambiar el sentido de la escalera la cinta deberá estar parada y las emergencias no estar pulsadas. Finalmente, con una confirmación de marcha podremos cambiar el sentido a la espera de una persona para activar la escalera en un sentido u otro. Para evitar la simultaneidad de sentidos, resetearemos además el bit opuesto.

En el último escalón vemos que cualquiera de las dos emergencias podrán parar la cinta en el momento que se desactiven.



Una vez validada la variable de marcha (en cualquier sentido), podemos esperar la llegada de una persona para poner la escalera en funcionamiento. A la llegada de una persona, activaremos la escalera durante un tiempo calculado en función de la velocidad del motor; en este caso se han utilizado 20 segundos.



## DESARROLLO PRÁCTICA 5

La sintaxis para un temporizador es %Tmx, donde x determina el número del temporizador. Este temporizador puede ser de tres tipos ó atender a tres funcionamientos distintos: TON, TP y TOF.

TON: Temporizador a la conexión.

TP: Temporizador pulso.

TOF: Temporizador a la desconexión.

Las variables para la gestión de un temporizador son:

%Tmi: i de 0 a 63 (número del temporizador)

Modo: TON,TP,TOF.

TB: Base de tiempo: 1ms,10ms,100ms,1s,1m

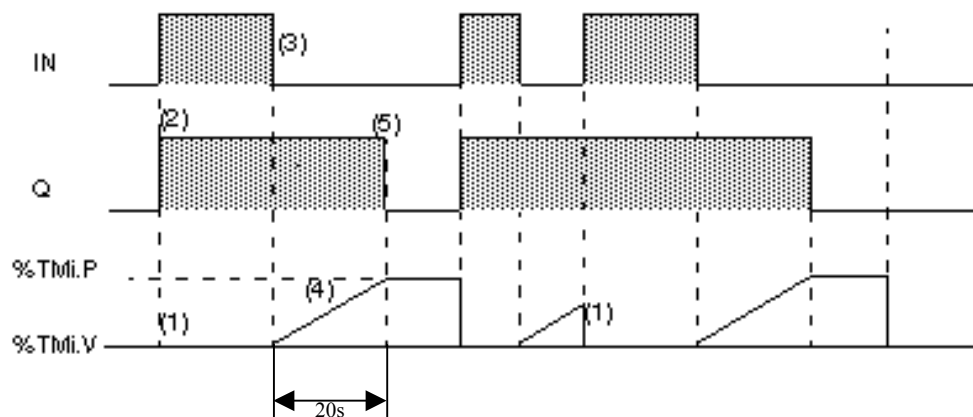
%Tmi.V: Palabra que crece de 0 al valor configurado %Tmi.P

%Tmi.P: Valor de preselección configurado.

IN: Entada de activación del temporizador.

Q: Salida del temporizador

Para seleccionar el tipo de temporizador es necesario tener en cuenta las posibles situaciones que pueden darse. Si tenemos en cuenta que una persona puede llegar, activar la fotocélula y quedarse parada, el tipo de temporizador ha utilizar será un TOF. Esto es porque el tiempo ha de empezar a contar a partir de que la persona pisa los escalones y deja la fotocélula.



En la tabla siguiente se describe el funcionamiento del temporizador TOF.

### Fase Descripción

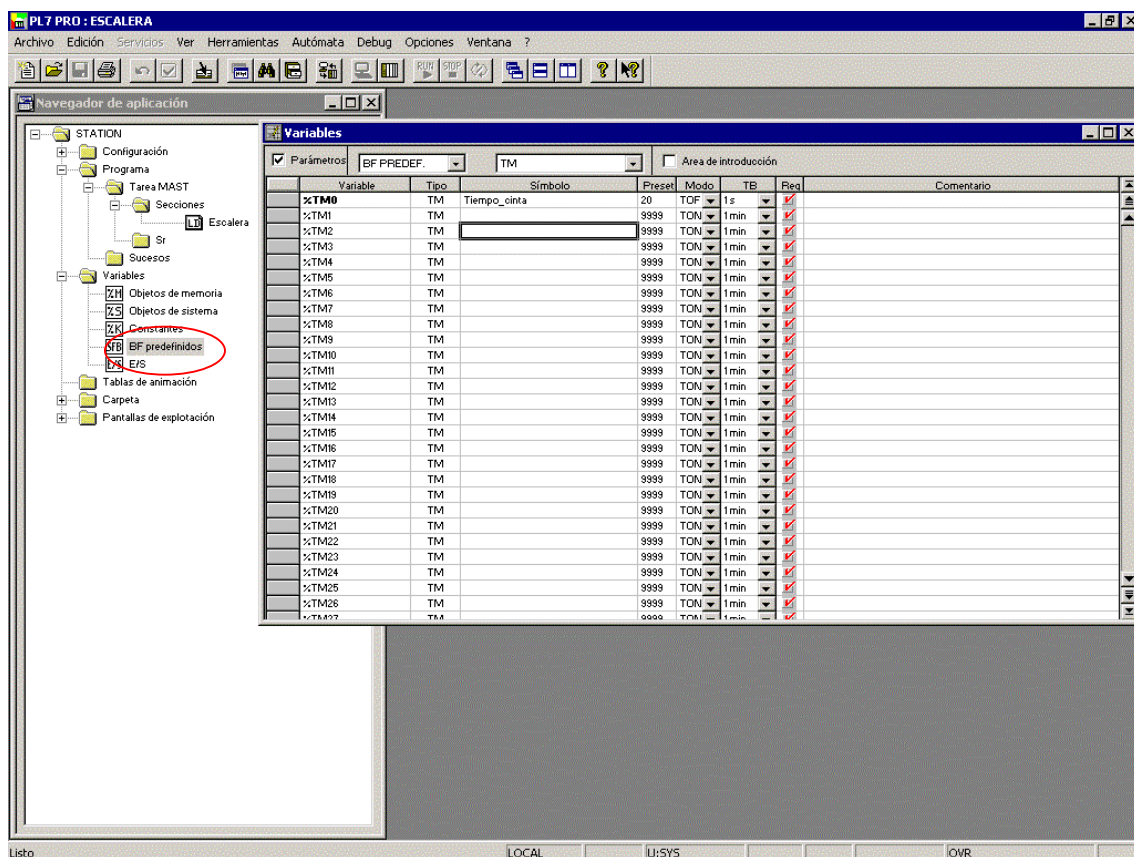
- 1 El valor actual %Tmi.V toma el valor 0 en un flanco ascendente de la entrada IN (aunque el temporizador esté en curso de evolución)
- 2 El bit de salida %Tmi.Q pasa a 1.
- 3 En un flanco descendente en la entrada IN, el temporizador se inicia.
- 4 El valor actual aumenta hacia %Tmi.P de una unidad en cada impulso de la base de tiempo TB.
- 5 El bit de salida %Tmi.Q vuelve a 0 si el valor actual alcanza %Tmi.P

## DESARROLLO PRÁCTICA 5

Para la parametrización del temporizador, seleccionaremos la opción **Variables->BF predefinidos** desde el navegador de la aplicación.

En la pantalla que aparece disponemos de todos los parámetros necesarios para configurar el temporizador.

Con la opción **Parámetros** indicada, podremos dar un símbolo al temporizador. El tiempo total de este bloque, será el resultado de multiplicar el valor de preselección (**Preset**) por la base de tiempo (**TB**). De esta forma, si necesitamos un temporizador de 20 segundos indicaremos un 20 en **Preset** y 1s como base de tiempo ó **TB**. Por último seleccionaremos el tipo de temporizador mediante la opción **Modo**.

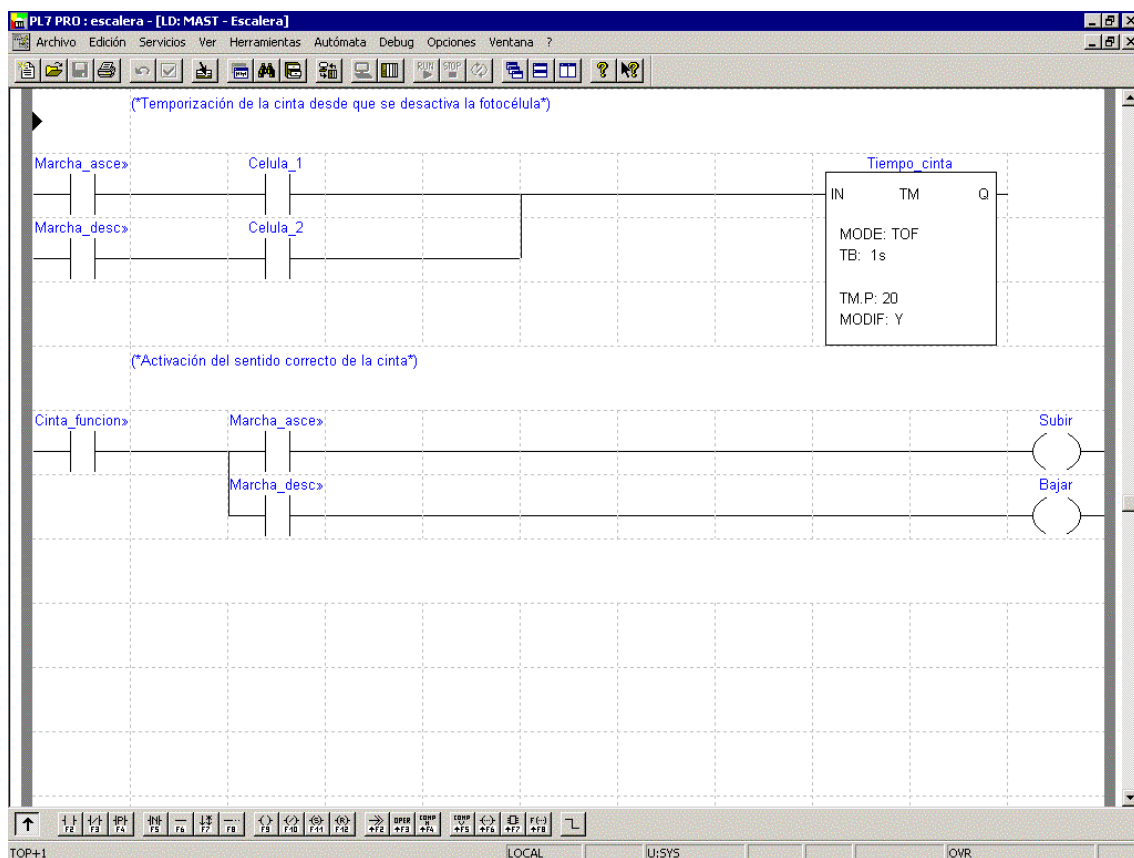


## DESARROLLO PRÁCTICA 5

Una vez configurado nuestro temporizador, podemos utilizarlo en el programa tal y como se haya planificado.

En el caso que nos ocupa cualquiera de los bits que indican el sentido con su correspondiente célula, podrán activar el temporizador. Cuando la variable Celula\_x se desactive, el temporizador comenzará a contar.

En el siguiente escalón se ha utilizado el bit %TM0.Q (cinta\_funcionando) que representa la salida del temporizador cero. Finalmente activaremos la salida correspondiente en función del sentido seleccionado.

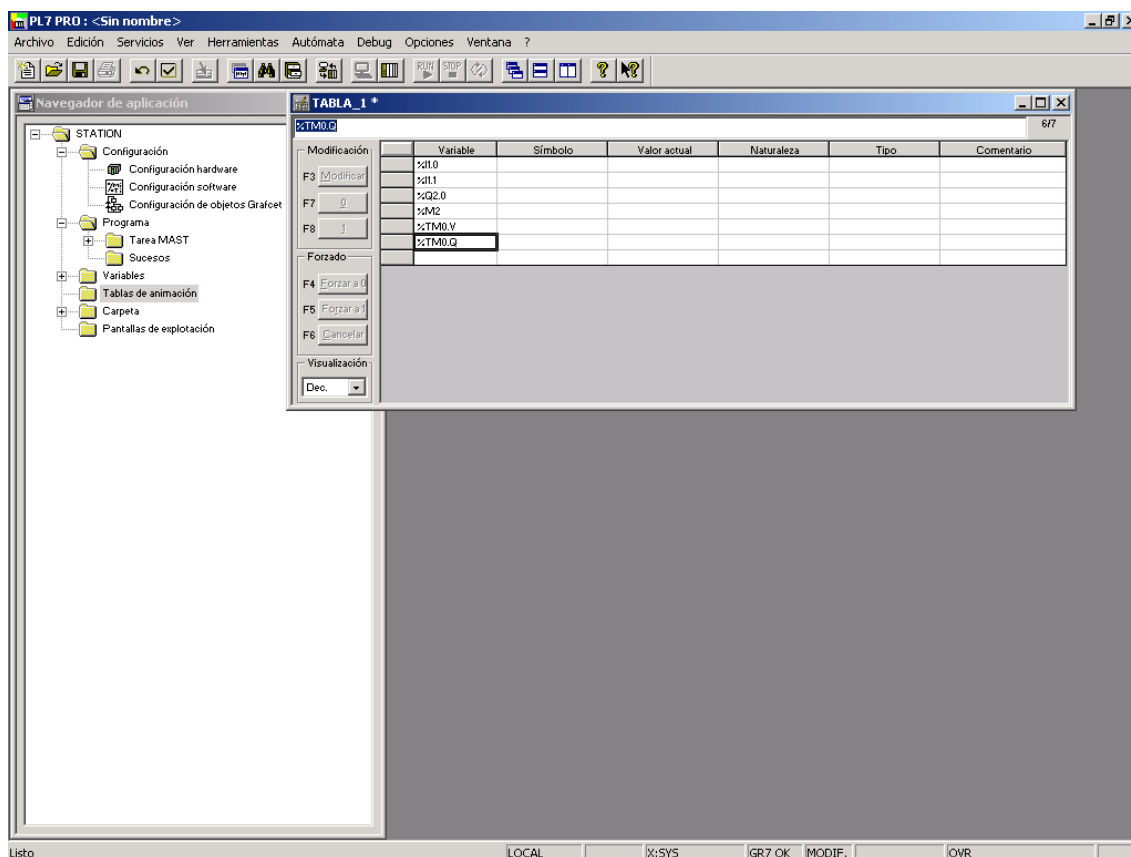


## DESARROLLO PRÁCTICA 5

Dado que en un programa pueden existir numerosas variables a controlar, es interesante disponer de una herramienta que permite la visualización simultánea de un conjunto de variables (valores de temporizadores, entradas, salidas, etc).

Para realizar esta función se dispone en PL7 de unas TABLAS ANIMADAS. Estas tablas, tipo hoja de cálculo, permiten ver y modificar en línea el estado de la variable que se indique en cada fila. Además ofrecen información sobre el símbolo y comentario que se haya definido previamente.

Para crear una tabla animada es necesario que desde la carpeta **Tablas animadas** del navegador de la aplicación y con el botón derecho del ratón, indiquemos la opción crear.



Escribiendo la variable que se quiere visualizar en la columna de la izquierda, podremos ver, modificar o forzar su valor. Mientras que la columna de **Modificación** permite activar un bit durante un ciclo de scan del autómat, el **Forzado** de los bits activa o desactiva estos independientemente de las instrucciones programadas.

Finalmente, es importante saber que estas tablas animadas pueden ser guardadas para su visualización posterior en el PC asignándoles un nombre. Además estas tablas son exportables a otra aplicación para su posterior utilización desde la opción exportar con el botón derecho de ratón sobre la tabla. La extensión de este tipo de archivos, es nombre.LD.

## PRÁCTICA 6

### CONTROL DE ACCESO A UNA INSTALACIÓN

#### OBJETIVO

Realizar el automatismo que controla el acceso de las personas a una instalación mediante dos puertas automáticas, una de salida y otra de entrada.

#### PRESENTACIÓN

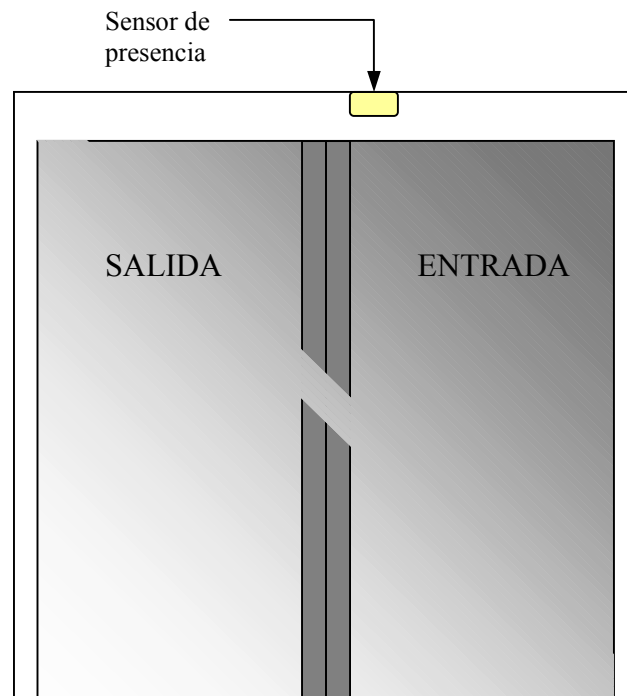
En esta ocasión necesitamos conocer el número de personas que acceden a una instalación. Cuando el detector de proximidad detecte una persona, la puerta correspondiente se abrirá inmediatamente permitiendo el paso de esta, ya sea de salida o de entrada.

Dada la construcción física del dispositivo, al desactivar la orden de apertura las puertas se cerrarán lentamente.

#### Lista de variables

Sensor de entrada	%I1.0
Sensor de salida	%I1.1
Puesta a cero	%I1.2
Apertura entrada	%Q2.0
Apertura salida	%Q2.1

#### Esquema de la instalación



## DESARROLLO PRÁCTICA 6

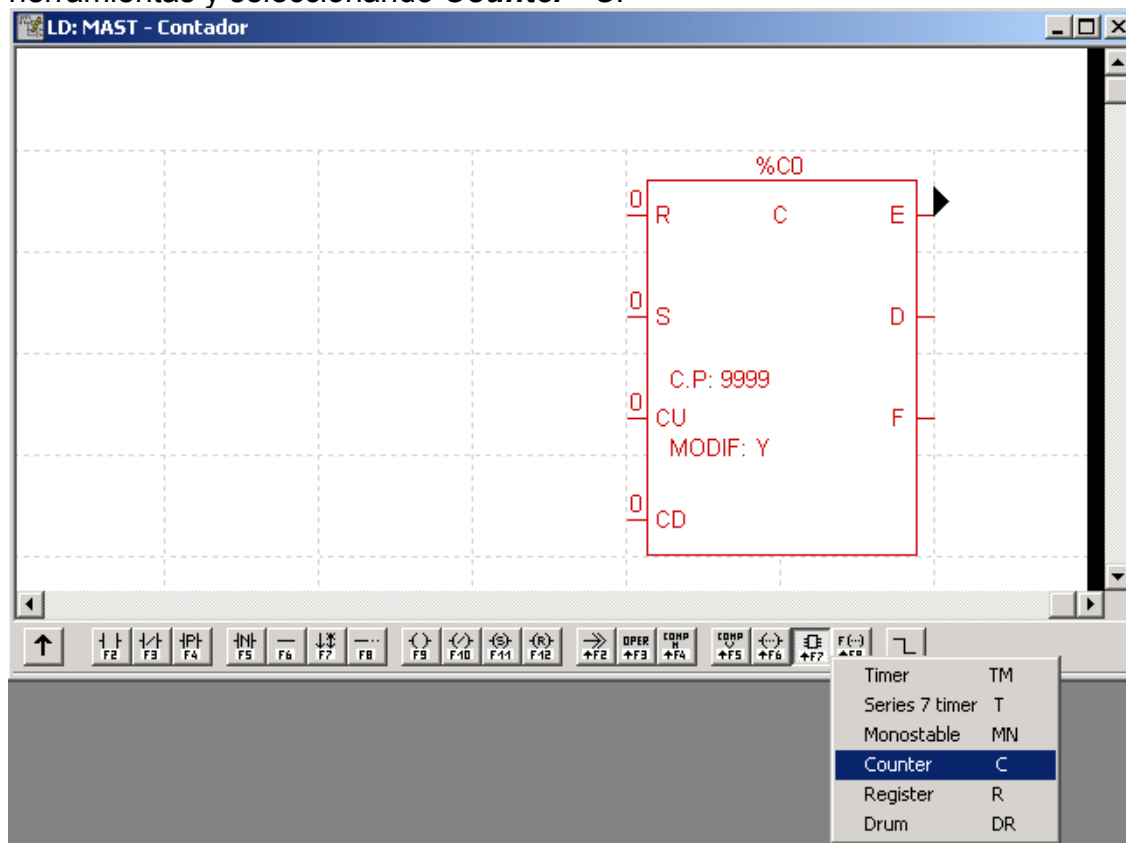
En el desarrollo de esta práctica se utilizará un contador que permitirá controlar el número de personas que acceden a la instalación.

La sintaxis para un contador es %Cx, donde x determina el número del contador. Una de sus propiedades es el valor de preselección. Este valor únicamente permite que el contador active una de sus salidas al alcanzar dicho número y concretamente es la salida %Cx.P. Para indicar este valor, entraremos en la opción **BF Predefinidos** situada en la carpeta **Variables** del navegador de la aplicación.

Indicando la opción **Parámetros** podremos programar el número máximo de personas, que en este caso estableceremos en 10 dentro de la casilla **Preset** de la fila correspondiente al contador a utilizar. En este caso se ha utilizado el contador cero.

Variable	Tipo	Símbolo	Preset	Req	Comentario
%C0	C		10	✓	
%C1	C		9999	✓	
%C2	C		9999	✓	
%C3	C		9999	✓	
%C4	C		9999	✓	
%C5	C		9999	✓	

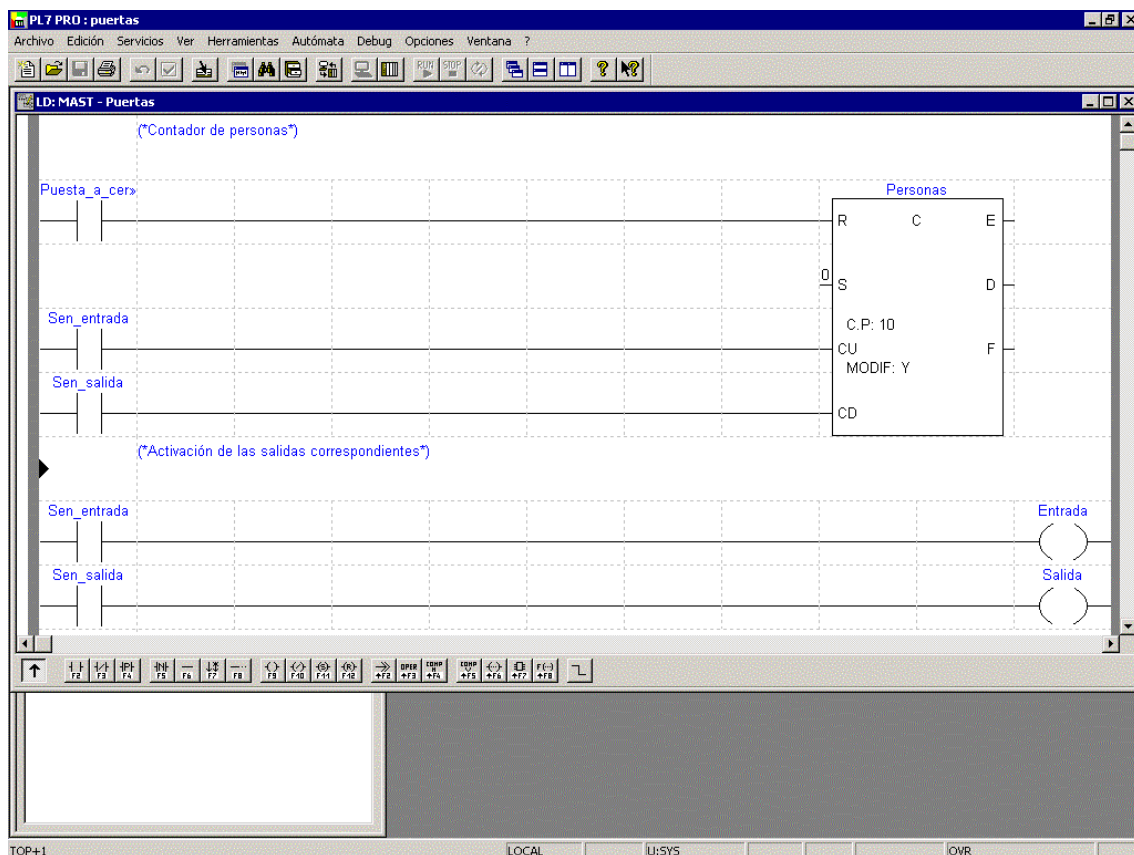
Una vez configurado el contador correspondiente, podremos utilizarlo en nuestro programa mediante el icono **Función Gráfica** situado en la barra de herramientas y seleccionando **Counter C**.



## DESARROLLO PRÁCTICA 6

Al disponer de una puesta a cero mediante la entrada %I1.2, conectaremos un contacto normalmente abierto a la entrada R de nuestro contador ( Reset ). La entrada que incrementa el contador en una unidad es CU ( Count Up ) y por ello se utilizará el bit de entrada %I1.0. Finalmente conectaremos el bit %I1.1 a la entrada CD ( Count Down ) para poder decrementar el valor del contador en uno cada vez que una persona salga de la instalación.

El abrir o cerrar las puertas dependerá directamente de los sensores de entrada o salida respectivamente. Si el sensor de entrada se activa, activaremos la puerta de entrada para que pueda abrirse. Lo mismo ocurrirá cuando el sensor de salida se active. Dada la construcción de esta instalación el cierre de las puertas será progresivo, por lo que las puertas no se cerrarán inmediatamente después de que el sensor se desactive.







## PRÁCTICA 7

### SEÑALES LUMINOSAS

#### OBJETIVO

Realizar el automatismo que controla el ciclo de encendido automático para un indicador de farmacia mediante bloques función.

#### PRESENTACIÓN

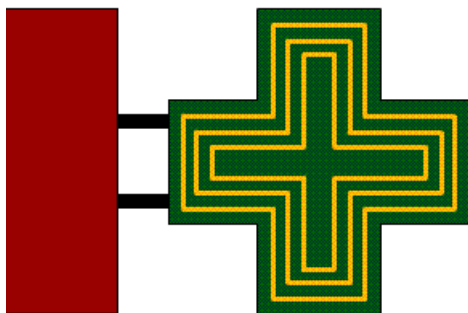
En esta ocasión necesitamos programar el encendido y apagado de 3 fluorescentes en un indicador. La cadencia de estos será de 1 segundo y su ciclo de funcionamiento es el siguiente:

Paso	Fluorescente 1	Fluorescente 2	Fluorescente 3
1	0	0	0
2	1	0	0
3	0	1	0
4	0	0	1
5	1	1	1

#### Lista de variables

Interruptor del indicador	%I1.0
Fluorescente 1	%Q2.0
Fluorescente 2	%Q2.1
Fluorescente 3	%Q2.2

#### Esquema de la instalación

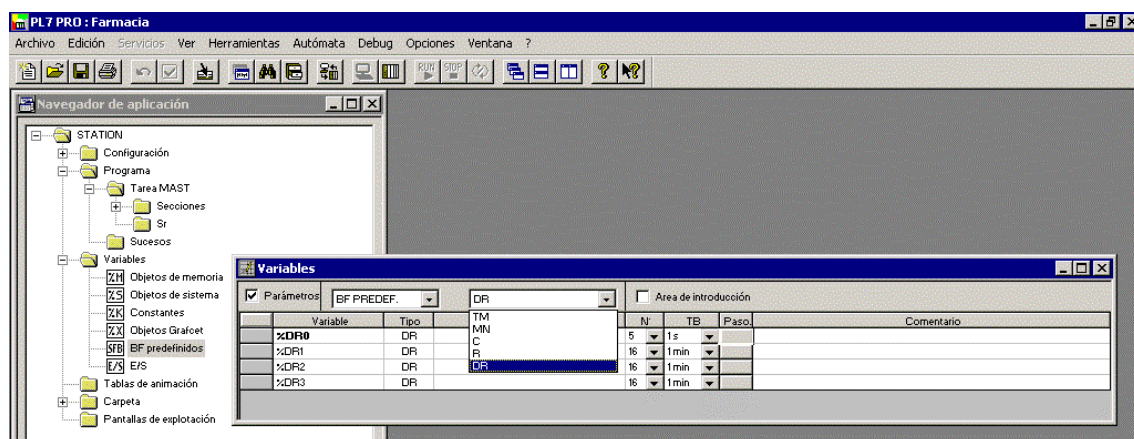


## DESARROLLO PRÁCTICA 7

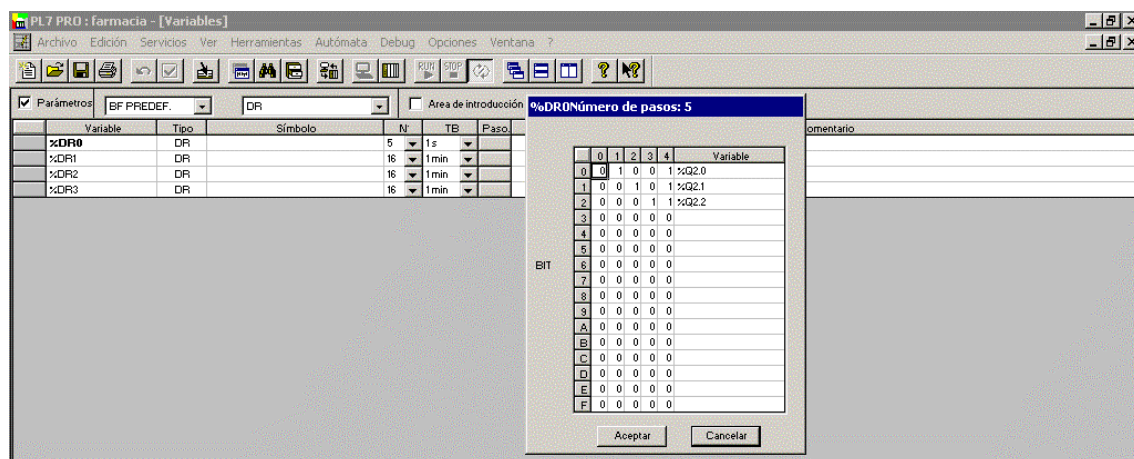
En esta ocasión se presenta la posibilidad de utilizar uno de los bloques predefinidos más prácticos de los que dispone PL7.

Un Drum o programador cíclico es un bloque que permite definir una serie de pasos o estado y unas variables que activar en cada estado. Un entrada de avance permite cambiar de estado. Puesto que la secuencia a seguir ha de ser automática utilizaremos un programador cíclico con 5 pasos ó estados y una base de tiempo de 1 segundo.

Para configurar el programador cíclico, entraremos en la opción **BF predefinidos** dentro de la carpeta variables situada en el navegador de la aplicación. Indicando sobre las opciones **Parámetros** y **DR** podremos introducir el número de pasos y la base de tiempo, que en este caso son 5 pasos y 1 segundo respectivamente.



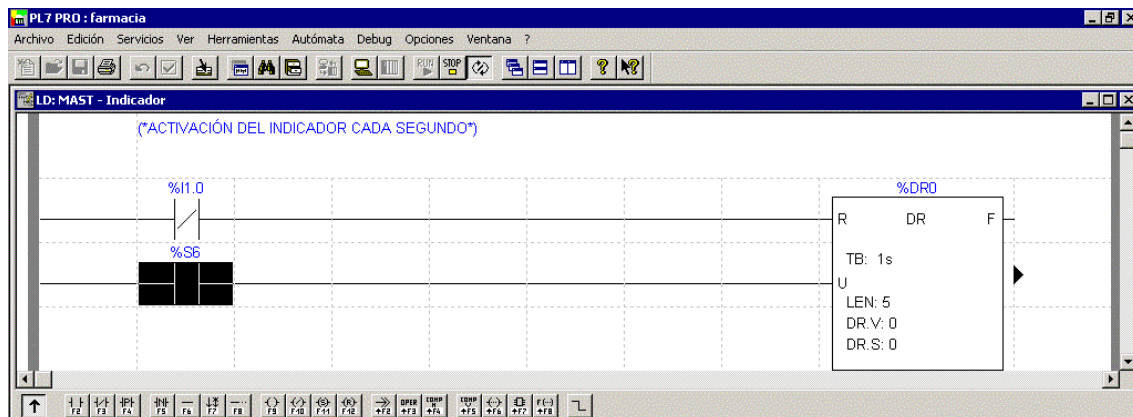
Mediante un “clic” sobre la casilla correspondiente de la columna **paso**, podremos configurar el programador para que active las salidas deseadas en cada paso. En esta ventana vemos que para el paso 0, no se activa ninguna salida. Para el paso 1, que corresponde con la segunda columna es la variable %Q2.0 la que queda activada. Para el siguiente paso (columna 2), la variable que se activa es la %Q2.1. Para el último paso (columna 4), se activan todas las variables.



## DESARROLLO PRÁCTICA 7

Una vez configurado el programador cíclico sólo queda utilizarlo en nuestro programa con el resto de variables.

La orden de marcha para el programador cíclico es un interruptor NA que, cuando esté desactivado, realizará un reset sobre el programador y este no se ejecutará. Para permitir que el programador cíclico avance en sus pasos, necesitamos una señal que alterne su valor automáticamente con un período de 1 segundo. En este caso se ha utilizado el bit sistema %S6.



Observar que mientras el parámetro %DR0.S indica el número de paso actual, %DR0.V indica el tiempo que permanecemos en cada paso medido en su base de tiempo.



## PRÁCTICA 8

### CONTROL DE UN CRUCE CON SEMÁFOROS

#### OBJETIVO

Realizar el automatismo que controla un cruce de vehículos programado en grafset.

#### PRESENTACIÓN

En esta ocasión necesitamos programar un cruce con dos direcciones y cuatro sentidos de circulación. El orden de activación y los tiempos para todos los semáforos son los siguientes:

PASO	SEMÁFORO A	SEMÁFORO B	SEMÁFORO C	SEMÁFORO D
1	VERDE	VERDE	ROJO	ROJO
2	AMBAR	AMBAR	ROJO	ROJO
3	ROJO	ROJO	VERDE	VERDE
4	ROJO	ROJO	AMBAR	AMBAR

ROJO: 35 segundos

AMBAR: 5 segundos

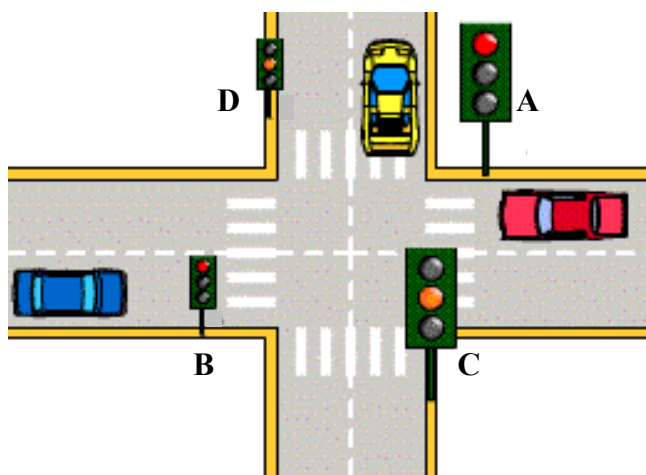
VERDE: 30 segundos

El automatismo estará controlado por un interruptor de marcha que, al estar desactivado y con el autómatas en RUN, pondrá todos los semáforos en ámbar intermitente.

#### Lista de variables

Interruptor de marcha	%I1.0
Rojo semáforo A	%Q2.0
Ámbar semáforo A	%Q2.1
Verde semáforo A	%Q2.2
Rojo semáforo B	%Q2.3
Ámbar semáforo B	%Q2.4
Verde semáforo B	%Q2.5
Rojo semáforo C	%Q2.6
Ámbar semáforo C	%Q2.7
Verde semáforo C	%Q2.8
Rojo semáforo D	%Q2.9
Ámbar semáforo D	%Q2.10
Verde semáforo D	%Q2.11

#### Esquema de la instalación





## DESARROLLO PRÁCTICA 8

El problema que se presenta en esta ocasión es un sistema secuencial, por lo que realizaremos un programa en grafcet que nos permitirá introducirnos en este lenguaje de una forma sencilla.

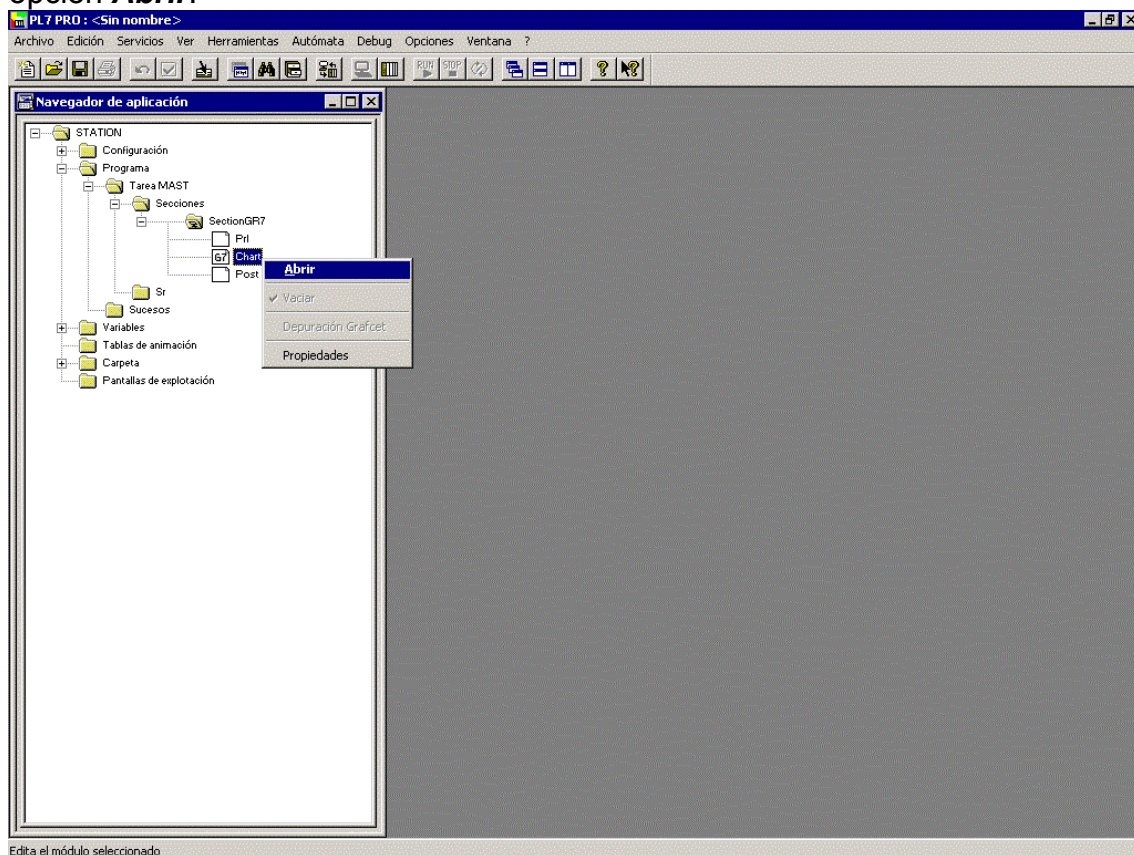
Como recordatorio a la teoría de la programación en grafcet podemos afirmar que este consiste en un diagrama funcional, cuyo objetivo es describir de forma gráfica el comportamiento de un automatismo secuencial. Además, este modelo queda definido por:

- unos elementos gráficos
- unas reglas de evolución

Teniendo presente estos principios, el siguiente paso consiste en crear una aplicación nueva indicando la opción Grafcet en la creación de la misma. En este momento PL7 a creado tres secciones contenidas en la carpeta **Programa->Tarea Mast->Secciones->SectionGR7**.

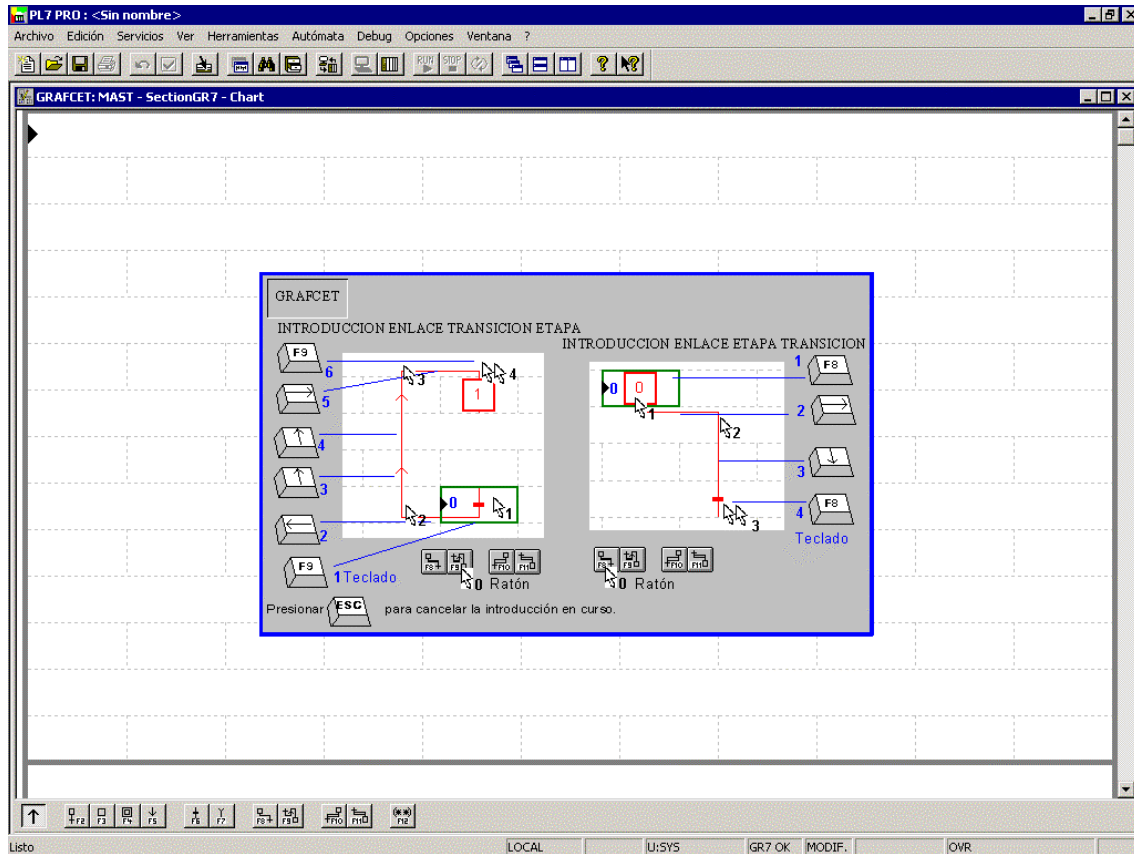
La sección **Pri** ó preeliminar, se utiliza para realizar los posibles posicionamientos del grafcet (emergencias, rearranques, etc). La sección **Chart** ó gráfico contiene la programación gráfica de nuestro autómat. Y por último, la sección **Post** ó posterior, que utilizaremos para la activación de salidas. Mientras que la sección **Chart** contiene el gráfico del Grafcet, las secciones preeliminar y posterior pueden ser programadas en cualquier lenguaje.

Situándose encima de esta sección y con el botón derecho seleccionaremos la opción **Abrir**.



## DESARROLLO PRÁCTICA 8

La página inicial del gráfico nos da un resumen de las teclas función para insertar los diferentes componentes del graficet.



Ratón en modo selección



Colocar una etapa con transición



Colocar una etapa



Colocar una etapa inicial



Envío a una etapa de destino



Colocar una transición



Indicar procedencia de envío



Enlace de etapa a transición



Enlace de transición a etapa



Final de secuencias simultáneas



Comienzo de secuencias simultáneas



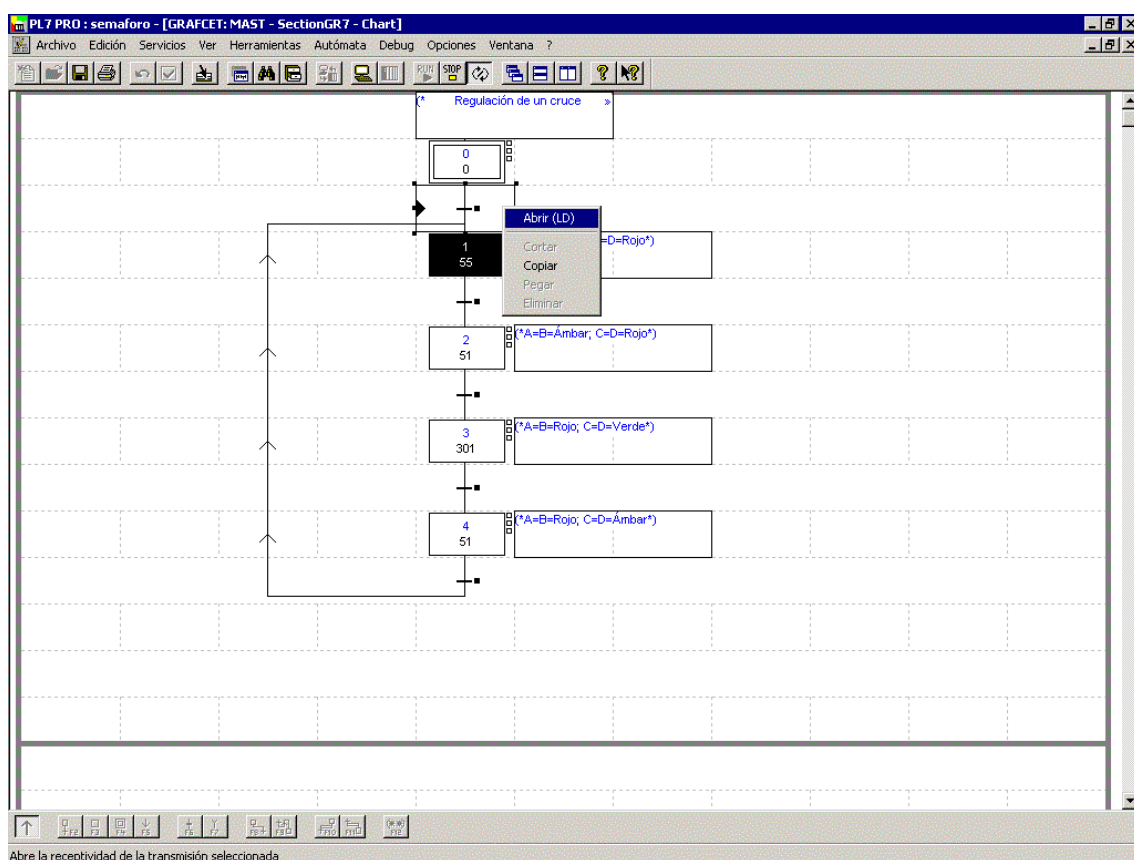
Colocar un comentario



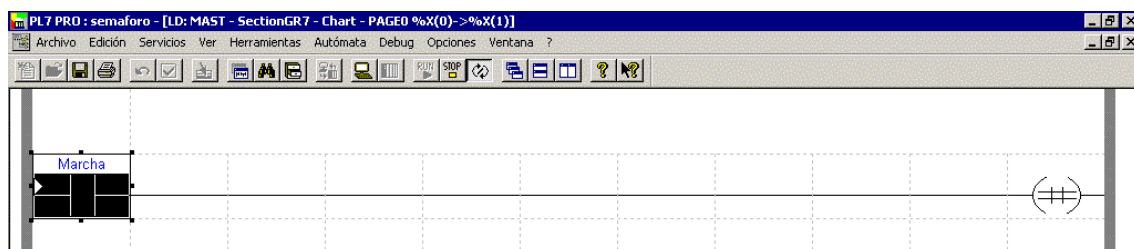
## DESARROLLO PRÁCTICA 8

Una de las posibles soluciones a implementar puede ser la que figura más abajo. Al existir cinco estados del automatismo (paro con ámbar intermitente y cuatro estados diferentes), se han utilizado cinco etapas grafcet en total.

Se ha programado un estado inicial (identificado por un doble recuadro y que se activa automáticamente al reinicializar el sistema), en el que activaremos las señales ámbar de cada semáforo de forma intermitente. La condición que activa el automatismo es Marcha, por lo que cuando esta se active pasaremos a un nuevo estado definido como paso 1 en el enunciado. Para programar la condición de cambio de estado deberemos entrar dentro de esta misma transición mediante el botón derecho del ratón y escribir las condiciones.



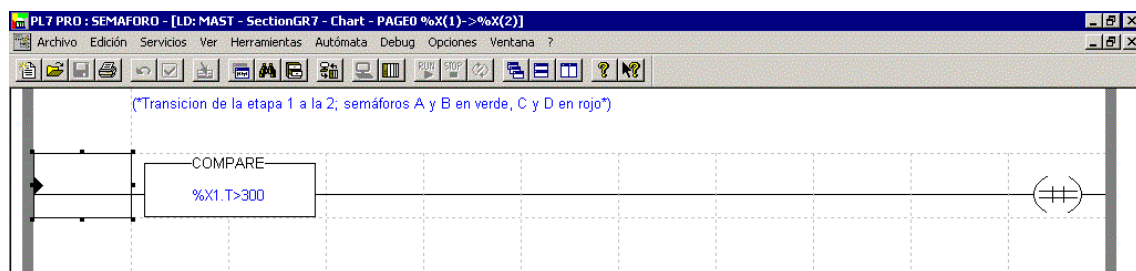
Una vez dentro de esta deberemos utilizar una bobina de transición, la cual nos permite programar esta parte del grafcet y realizar el salto a otra etapa.



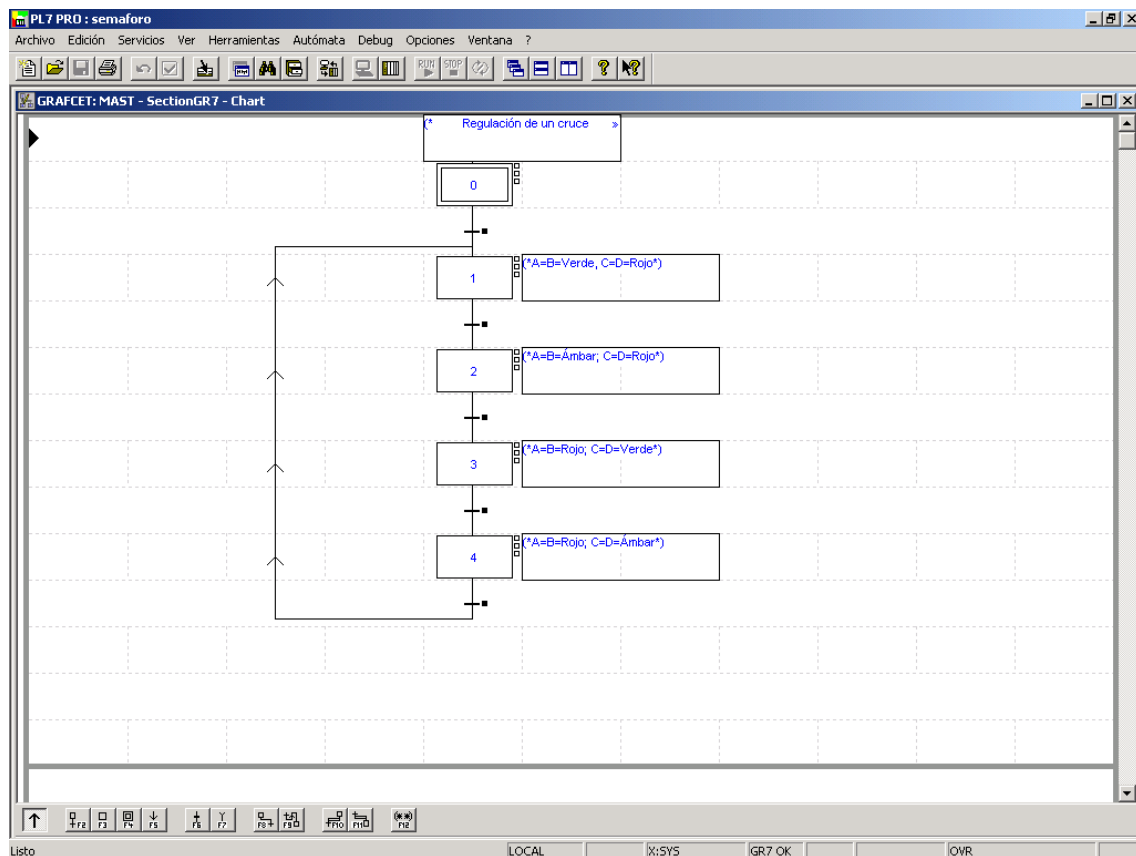
## DESARROLLO PRÁCTICA 8

La transición desde la etapa 1 a la 2 corresponde al tiempo en que los semáforos A y C están en verde. Si tenemos presente que esta condición consiste en un tiempo (30 segundos) y que podemos conocer el tiempo en que la etapa 1 está activa, podemos utilizar la variable %X1.T dentro de un bloque de comparación. Esta variable nos da el tiempo en que la etapa 1 esta activa en décimas de segundo.

El bloque utilizado corresponde a un comparador horizontal que utiliza dos variables tipo palabra. Este se encuentra en la barra de herramientas o mediante la tecla F4 y las instrucciones posibles son: <,>,<=,>=,<>.



De igual forma y siguiendo esta filosofía, podemos programar el resto del grafcet hasta terminar el dibujo. Puesto que la cuarta etapa representa el último estado del automatismo, deberemos realizar el envío al primer estado, que en este caso queda representado por la etapa 1.

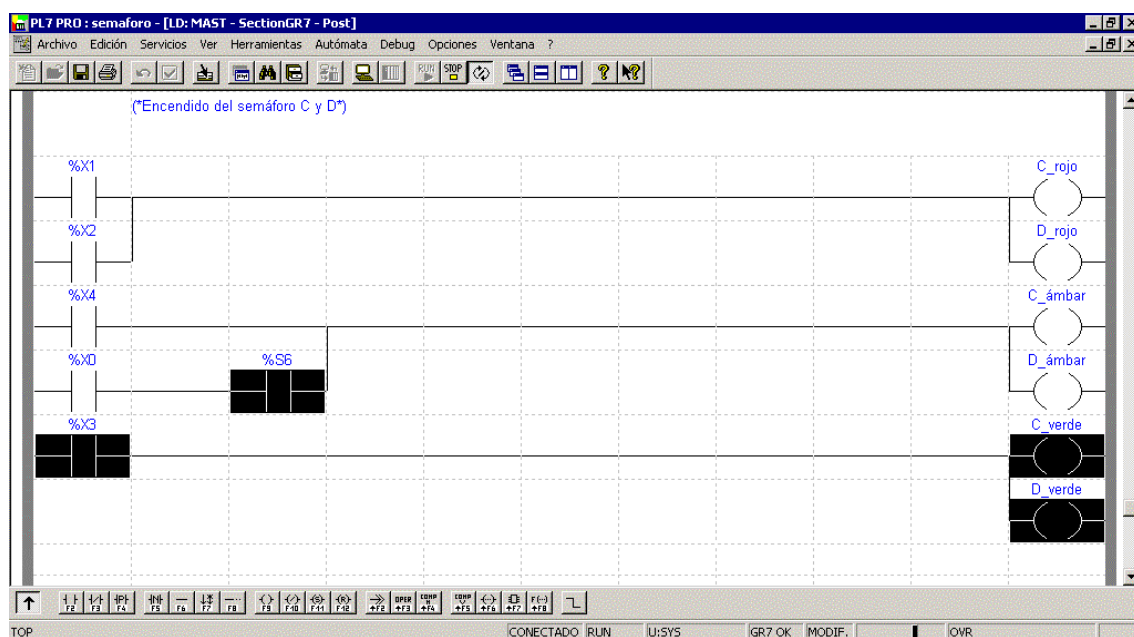
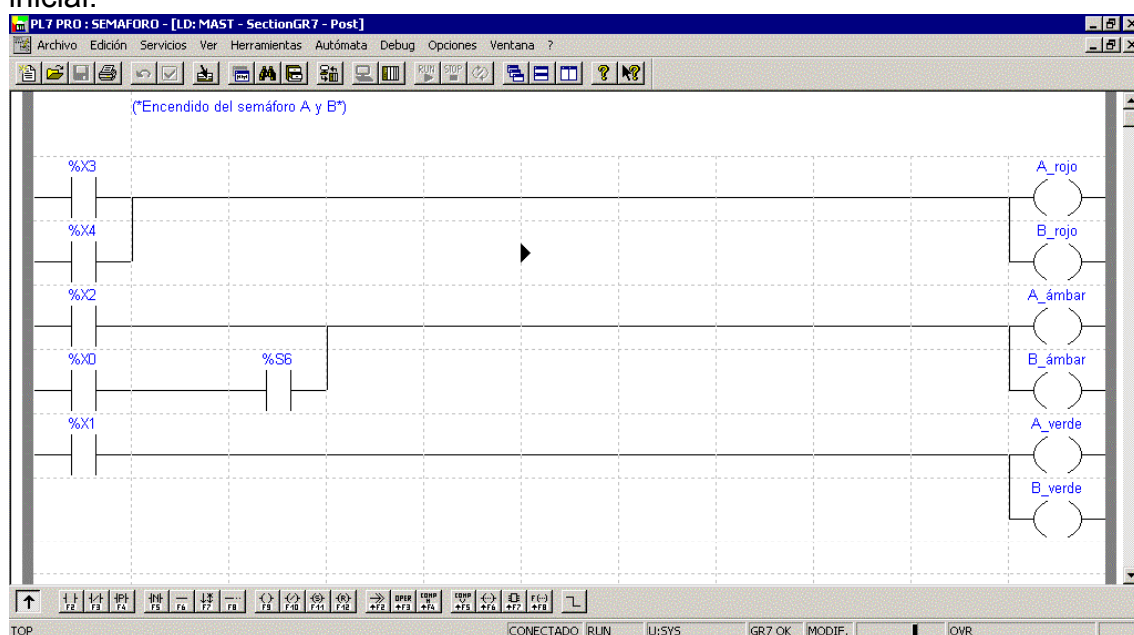


## DESARROLLO PRÁCTICA 8

Una vez dibujado el gráfico de nuestro programa, es necesario indicar qué salidas se activarán en cada etapa del grafcet. Para ello entraremos dentro de la sección posterior del grafcet desde el navegador de la aplicación y, teniendo en cuenta que una etapa del grafcet se identifica por **%Xn** (siendo n el número de etapa), podremos activar las salidas en cada estado ó etapa del grafcet.

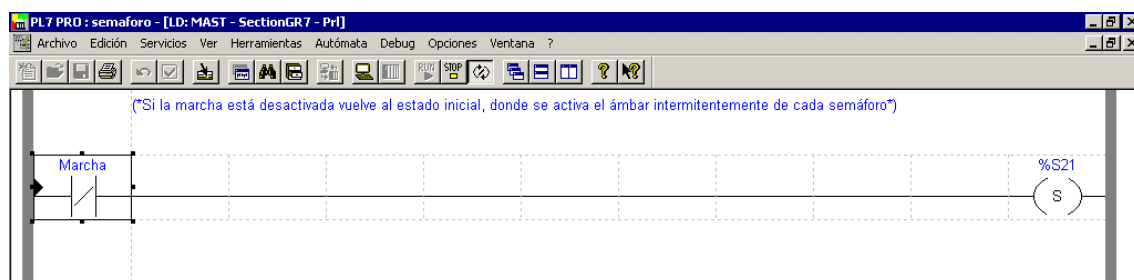
Tal y como puede apreciarse en la figura, el indicador verde de los semáforos A y B se activan en el estado 1. De la misma forma se activarán el resto de salidas en función del estado del grafcet.

Por otro lado, podemos ver la intermitencia del ámbar de cada semáforo con el bit sistema **%S6** (activo cada segundo) siempre y cuando estemos en el estado inicial.



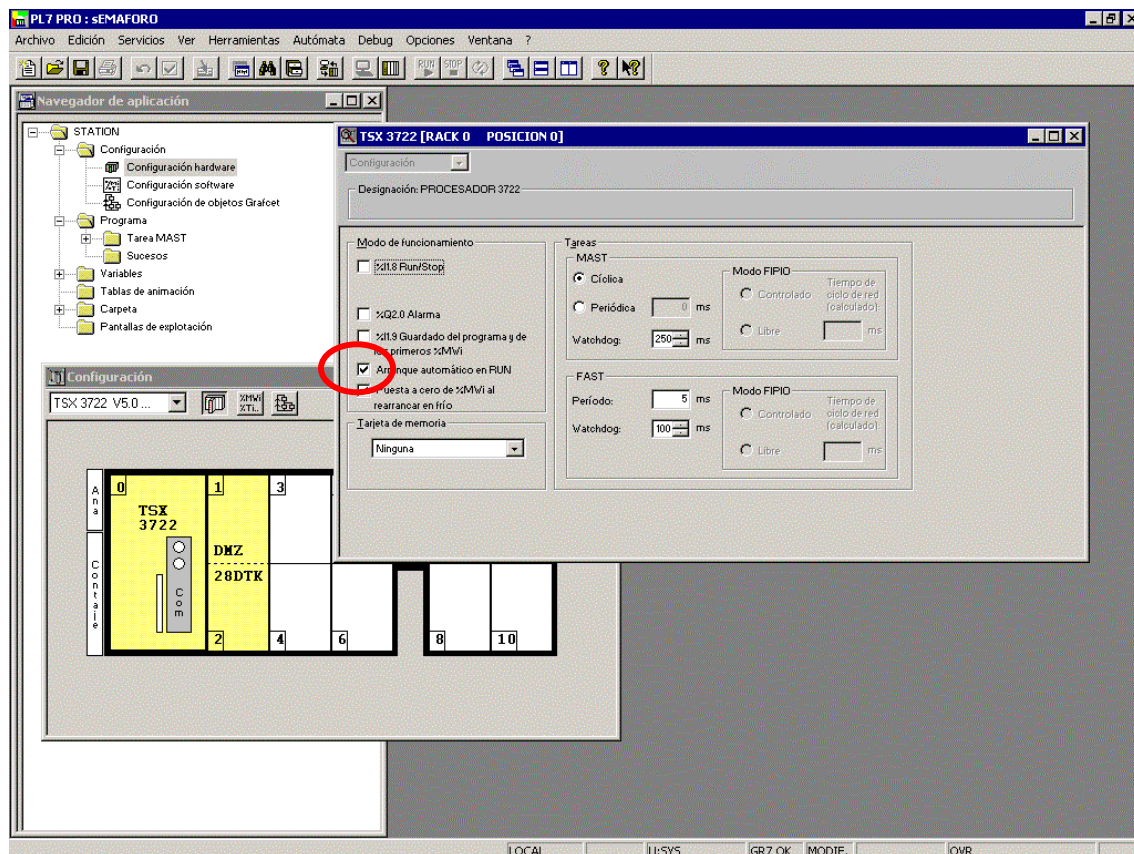
## DESARROLLO PRÁCTICA 8

Por último, es necesario programar la condición que lleva el grafcet al estado inicial y permite colocar todos los semáforos en ámbar intermitente. Podemos considerar esta acción como un posicionamiento del grafcet, lo cual invita a programar esta acción en el módulo preeliminar. Si entramos en esta sección programada en ladder desde el navegador de la aplicación, podremos programar la activación del bit %S21 cuando la marcha pase a cero. Este bit permite la inicialización del grafcet.



Otra de las cuestiones a tener en cuenta es que este tipo de instalaciones pueden estar situadas lejos del centro de control ó en un lugar de difícil acceso, y puede ser conveniente seleccionar la opción de **“Rearranque automático en RUN”**. Esta opción permitirá al automático ejecutar el programa en caso de una caída de tensión y retorno de la misma.

Podremos encontrarla dentro de la carpeta **configuración hardware** y haciendo doble “clic” sobre el esquema de la CPU.







## PRÁCTICA 9

### AUTOMATISMO DE UN GARAJE

#### OBJETIVO

Realizar el automatismo que controla la entrada y salida de vehículos en un garaje programado en Grafset.

#### PRESENTACIÓN

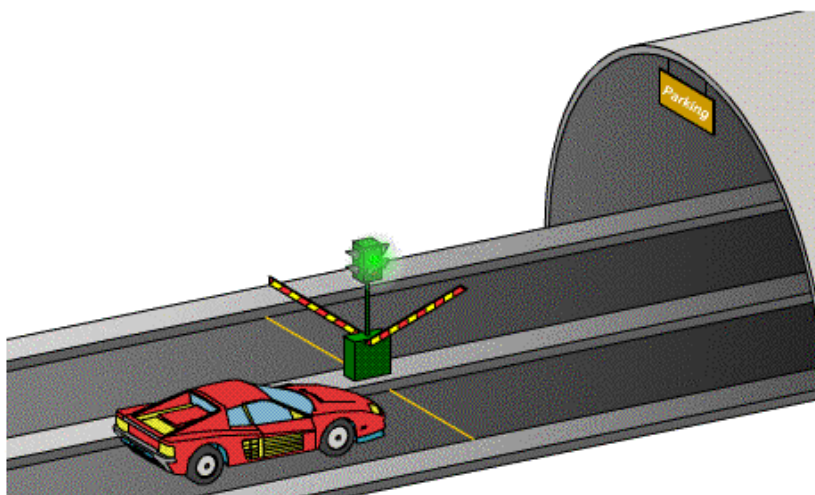
En esta ocasión necesitamos implementar el automatismo que controla la ocupación de un garaje con 15 plazas. Cada vez que un vehículo acceda al parking, el número de plazas ocupadas aumentará en uno. Cuando otro vehículo salga del parking el número de plazas disminuirá. Cuando todas las plazas estén ocupadas, no se permitirá el acceso a la instalación.

El estado del parking se señalizará mediante una semáforo en verde para plazas libres y otro en rojo para indicar que no hay plazas.

#### Lista de variables

Fotocélula de entrada	%I1.0
Presencia de vehículo en barrera de entrada	%I1.1
Barrera de entrada abierta	%I1.2
Barrera de entrada cerrada	%I1.3
Fotocélula de salida	%I1.4
Presencia de vehículo en barrera de salida	%I1.5
Barrera de salida abierta	%I1.6
Barrera de salida cerrada	%I1.7
Subir barrera de entrada	%Q2.0
Bajar barrera de entrada	%Q2.1
Subir barrera de salida	%Q2.2
Bajar barrera de salida	%Q2.3
Indicador de semáforo verde	%Q2.4
Indicador de semáforo rojo	%Q2.5

#### Esquema de la instalación

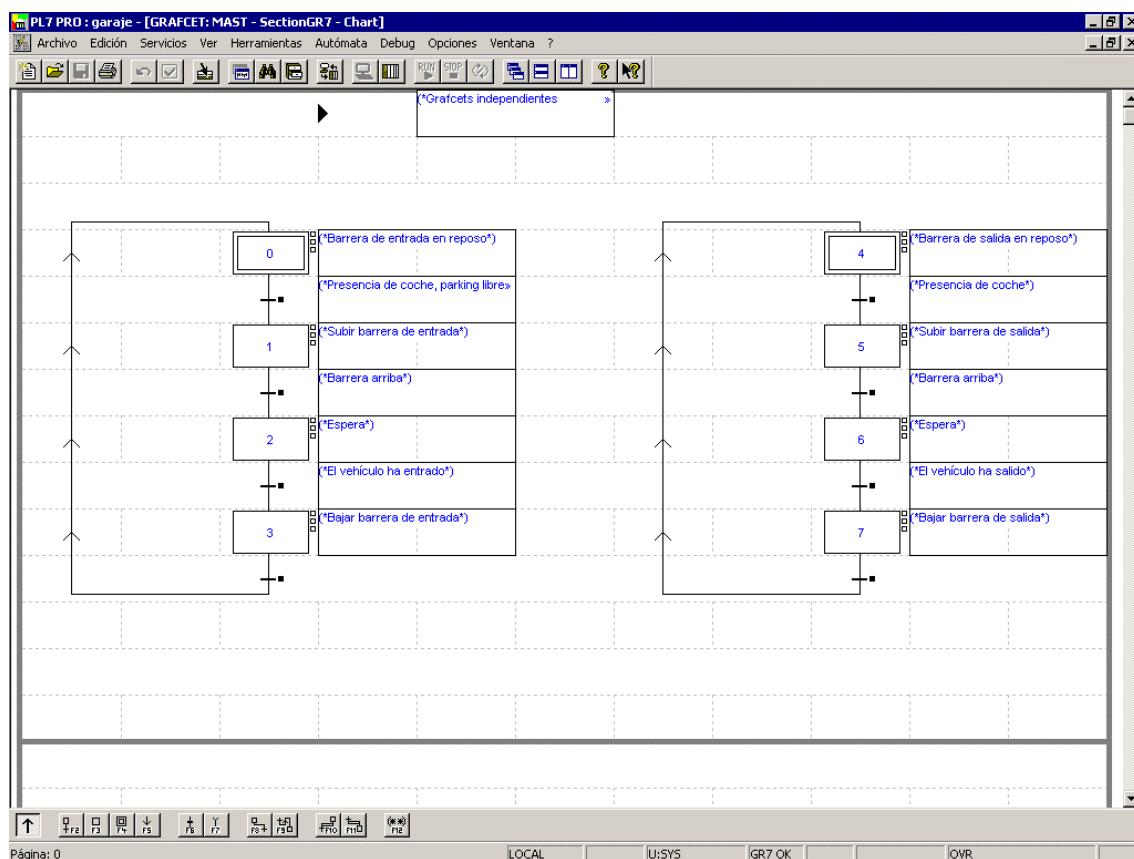


## DESARROLLO PRÁCTICA 9

Dado que el funcionamiento de un automatismo puede tener partes independientes entre sí, es interesante practicar con grafcet independientes que aseguran el buen funcionamiento de la máquina.

En esta ocasión disponemos de dos barreras controladas por un mismo autómat, lo cual puede implicar secuencias de activación diferentes. Por ello, y aunque los dos grafcet posean la misma topología, la activación de las salidas y las transiciones entre etapas responderán a variables distintas del grafcet.

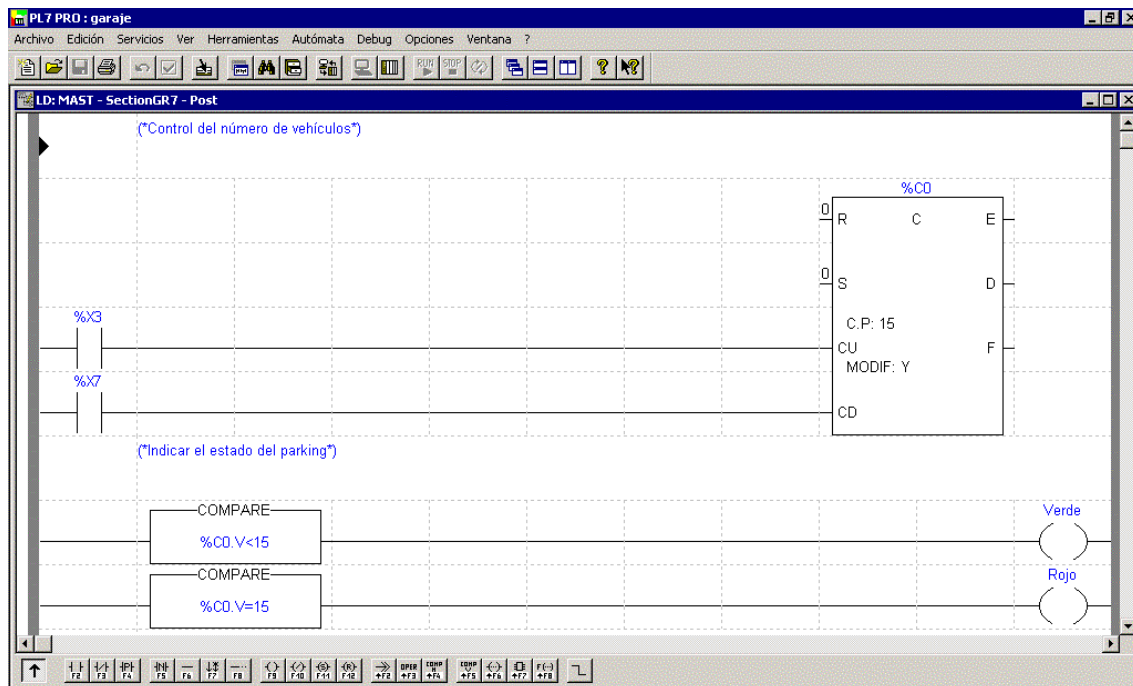
El grafcet implementado para cada barrera esta dividido en cuatro etapas. La primera corresponde al estado de reposo, en el cual esperamos la llegada de un automóvil. Una vez detectada la presencia, se procede al levantamiento de la barrera hasta que llegue a su tope. El siguiente estado corresponde a una espera hasta que el vehículo pase de la barrera y poco después pasamos a cerrarla incrementando o decrementando el número de vehículos respectivamente.



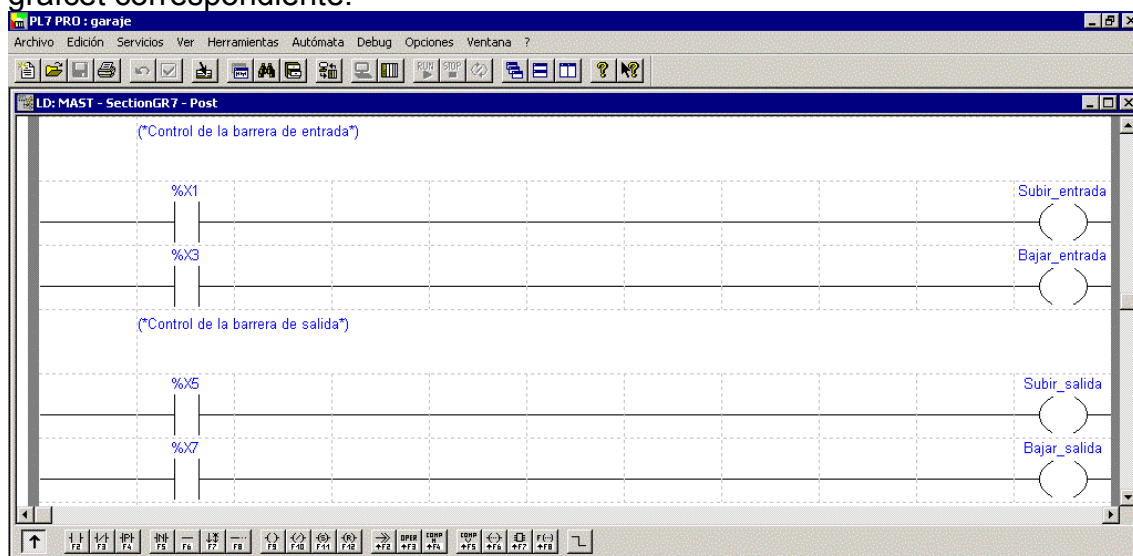


## DESARROLLO PRÁCTICA 9

Tal y como se ha explicado en el apartado anterior, aunque dispongamos de dos grafcet la activación de entradas y salidas quedan gestionadas por el mismo autómatas. Por ello, existirán elementos comunes dentro del programa. De esta forma y recordando el grafcet que se había dibujado, vemos que la etapa en la que aumentamos el número de vehículos es la número tres (%X3). La etapa que decrementa los vehículos es la séptima (%X7). La señalización del número de vehículos dependerá del valor del contador. Dos bloques de comparación serán los encargados de gestionar estas salidas.



Dada la simplicidad de este ejercicio, la activación de las salidas se ha realizado directamente en vez de utilizar bits de memoria intermedios. Tal y como se ve en la figura, activaremos una salida u otra en función del estado del grafcet correspondiente.





## PRÁCTICA 10

### REGULACIÓN DE TEMPERATURA

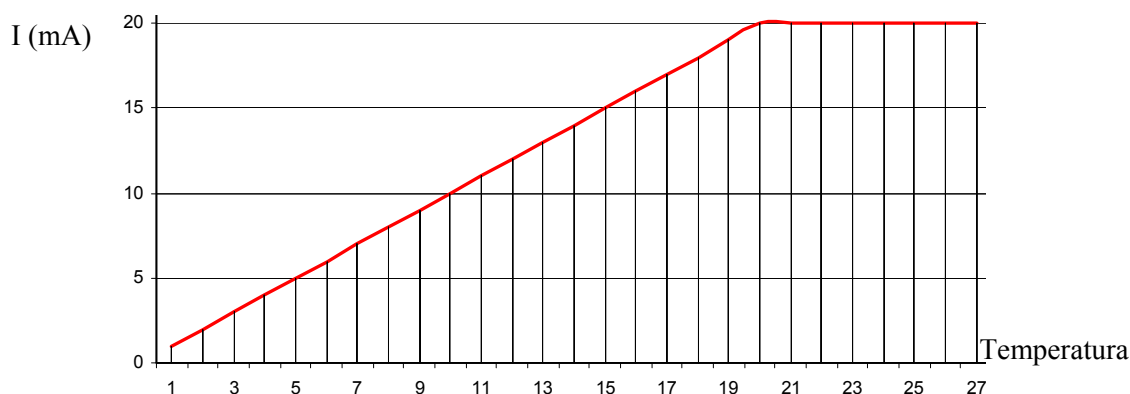
#### OBJETIVO

Practicar con señales analógicas y digitales en la regulación de temperatura de una sala; utilizar el display integrado del TSX Micro.

#### PRESENTACIÓN

A partir de la configuración básica del autómata, queremos realizar un programa que mantenga la temperatura de una sala alrededor de una consigna dada. Mediante una bomba de calor reversible, el sistema permitirá aportar calor si la temperatura baja y enfriar si ésta aumenta. La consigna, o temperatura a mantener será constante y alrededor 15°C.

Para evitar que la activación de las salidas alterne muy rápidamente entorno a la consigna, estableceremos una histéresis de 1 grado. De esta manera, activaremos la salida de calor al bajar a los 14 °C y la de frío al alcanzar los 16°C. El sensor de temperatura analógico posee la siguiente característica:

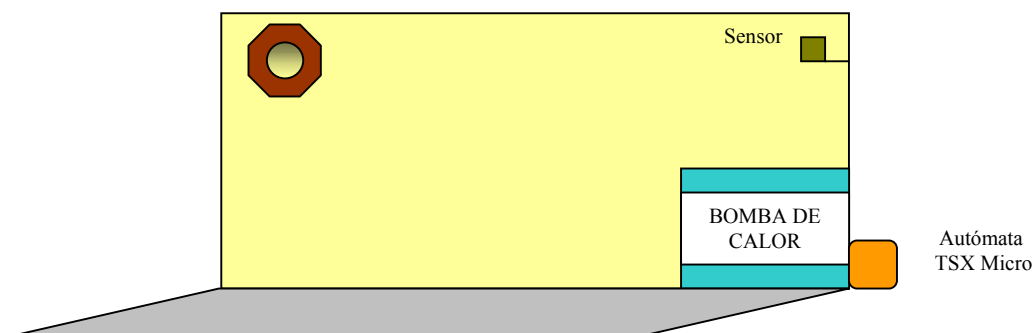


Se pide, además, visualizar la temperatura de la sala en el display integrado.

#### Lista de variables

Marcha	%I1.0
Paro (NC)	%I1.1
Sensor de temperatura	%IW0.5
Salida de calor	%Q2.0
Salida de frío	%Q2.1

#### Esquema de la instalación



## DESARROLLO PRÁCTICA 10

En el desarrollo de esta práctica utilizaremos las entradas analógicas que el micro TSX3722 lleva integradas de serie. Hablamos del primer conector SUB-D15 al que conectaremos el simulador de entradas y salidas.

Este accesorio permite configurar cada vía analógica como:

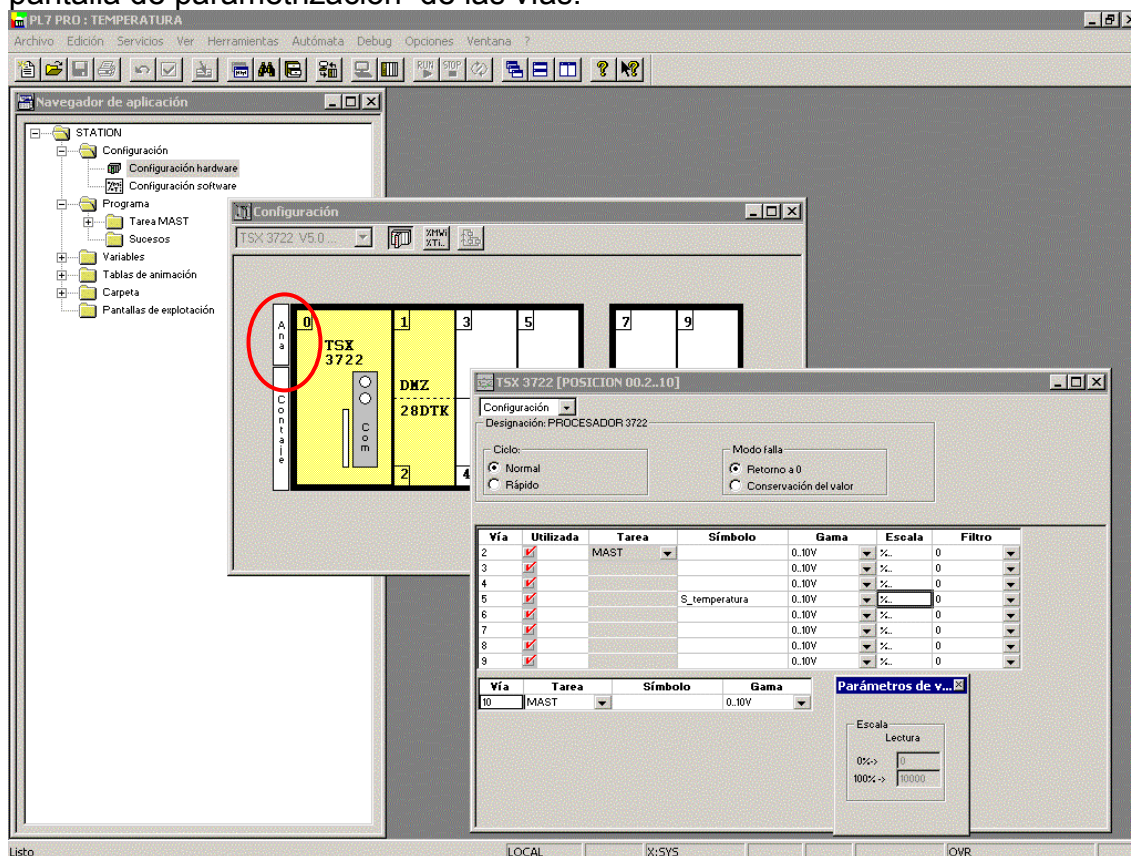
- 0-10 V
- 0-20 mA
- 4-20 mA
- Simulación (vías 2,3,4 y 5)

Para colocar una vía en modo simulación, es necesario colocar sus cuatro micro interruptores tal y como indica la siguiente tabla:

Grupo de micro interruptores	Potenciómetro
C1	0
C2	0
C3	1
C4	1

Una vez configurado el simulador de entradas, podemos abrir un nuevo archivo con las opciones definidas en la práctica 1.

Para configurar las vías analógicas entraremos dentro de la opción **Configurar hardware** del navegador de la aplicación. Mediante un doble “clic” en la pestaña **Ana**, situada a la izquierda del configurador, podemos acceder a la pantalla de parametrización de las vías.



## DESARROLLO PRÁCTICA 10

La pantalla anterior permite seleccionar, para cada vía, su activación ó desactivación, el modo de trabajo (0-10V,0-20mA ó 4-20mA) y el filtro. Este último parámetro es conveniente tenerlo en cuenta cuando el sistema va a trabajar en un ambiente hostil o perturbado electromagnéticamente por otros dispositivos de mayor potencia. A mayor número de filtro, más inmune será nuestra señal analógica y por el contrario mas lenta su variación.

Mediante un doble “clic” sobre la opción **Escala**, podemos ver que al 100% de nuestra señal, el valor obtenido será de 10000. Configurado para 0-10V y para 0-20mA esta relación es lineal, pero para 4-20mA será necesario hacer un cálculo con una relación no lineal.

Según la gráfica de nuestro sensor, cuando la temperatura sea de 20 grados la corriente suministrada será de 20 mA. Este valor corresponde a una lectura de 10000 en el registro del autómat. Asimismo para una lectura de 10 grados, la corriente suministrada será de 5 mA y el valor del registro de 5000.

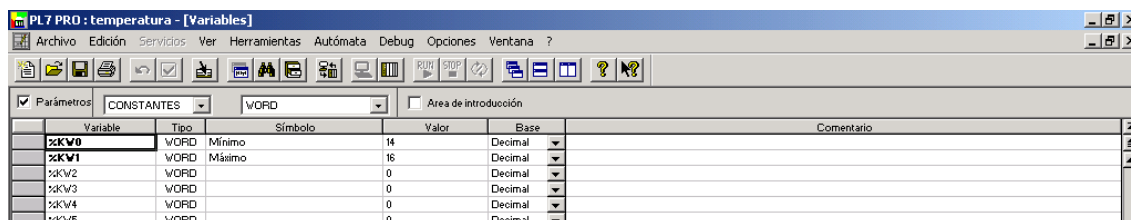
Conociendo esta relación, es posible encontrar la ecuación que describe el valor del registro en función de la temperatura de la sala y de esta manera será posible trabajar con valores de temperatura en nuestro programa. Esto es:

$$\text{Lectura Temperatura} = \frac{\text{registro}}{10000} * \text{registro} = \frac{\text{registro}}{500} \quad 20$$

Dado que conocemos los márgenes de temperatura en los cuales hay que activar una salida y su relación con el valor analógico, es posible establecer dos constantes que permitirán la activación de estas salidas. Desde el navegador de la aplicación podemos acceder a la pantalla **Variables** que permitirá definir un valor para estas constantes. En este caso su valor, dado que la relación es lineal, será:

%KW0= 14

%KW1= 16



Variable	Tipo	Símbolo	Valor	Base	Comentario
%KW0	WORD	Mínimo	14	Decimal	
%KW1	WORD	Máximo	16	Decimal	
%KV2	WORD		0	Decimal	
%KV3	WORD		0	Decimal	
%KV4	WORD		0	Decimal	
%KV5	WORD		0	Decimal	

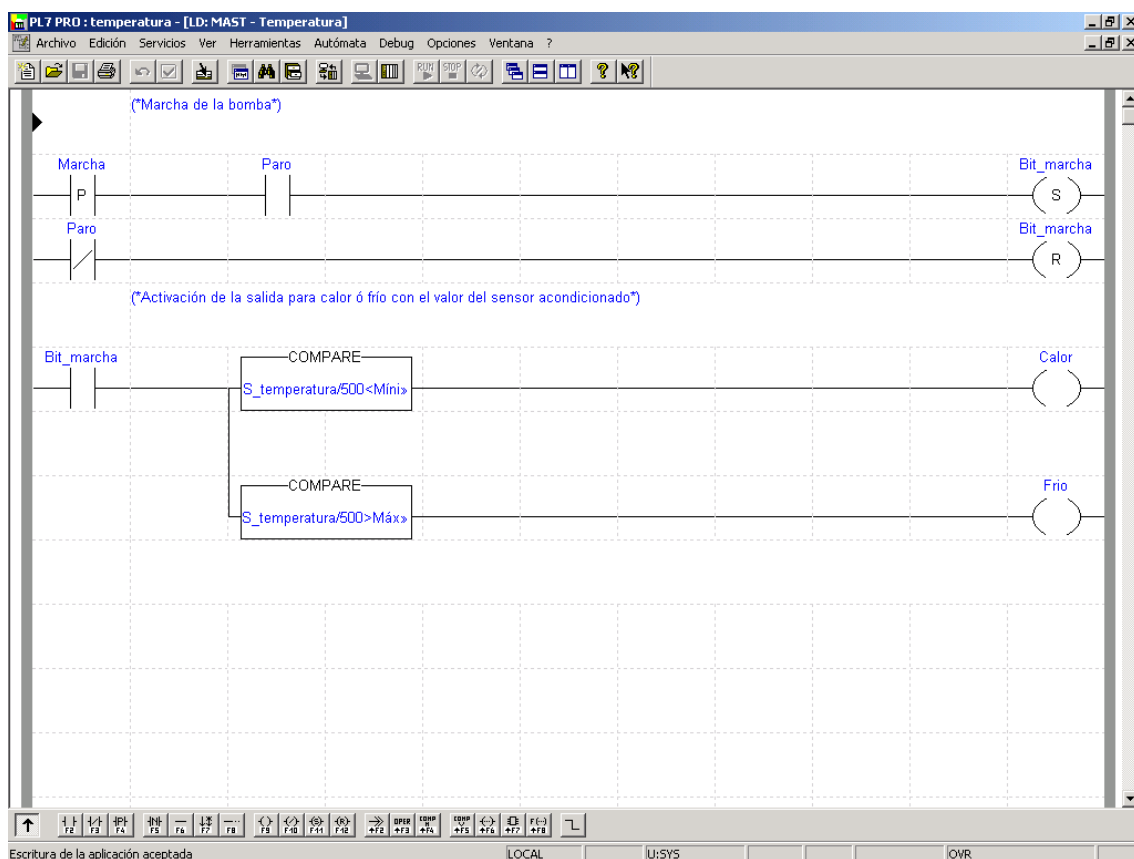
## DESARROLLO PRÁCTICA 10

En la resolución de este programa crearemos una nueva sección en ladder desde el navegador de la aplicación donde incluiremos la gestión de la bomba de calor.

El primer escalón de nuestro programa permite la activación o desactivación de la bomba con una preferencia del paro sobre la marcha.

Este bit de memoria permite la comparación de la señal analógica en el siguiente escalón.

Si alguna de estas dos comparaciones es verdadera, se activará la salida correspondiente. La salida de calor se activará si el valor del sensor es menor a la primera constante que se había marcado de 14°C y la salida de frío cuando el valor del sensor sea mayor al nivel de 16°C.





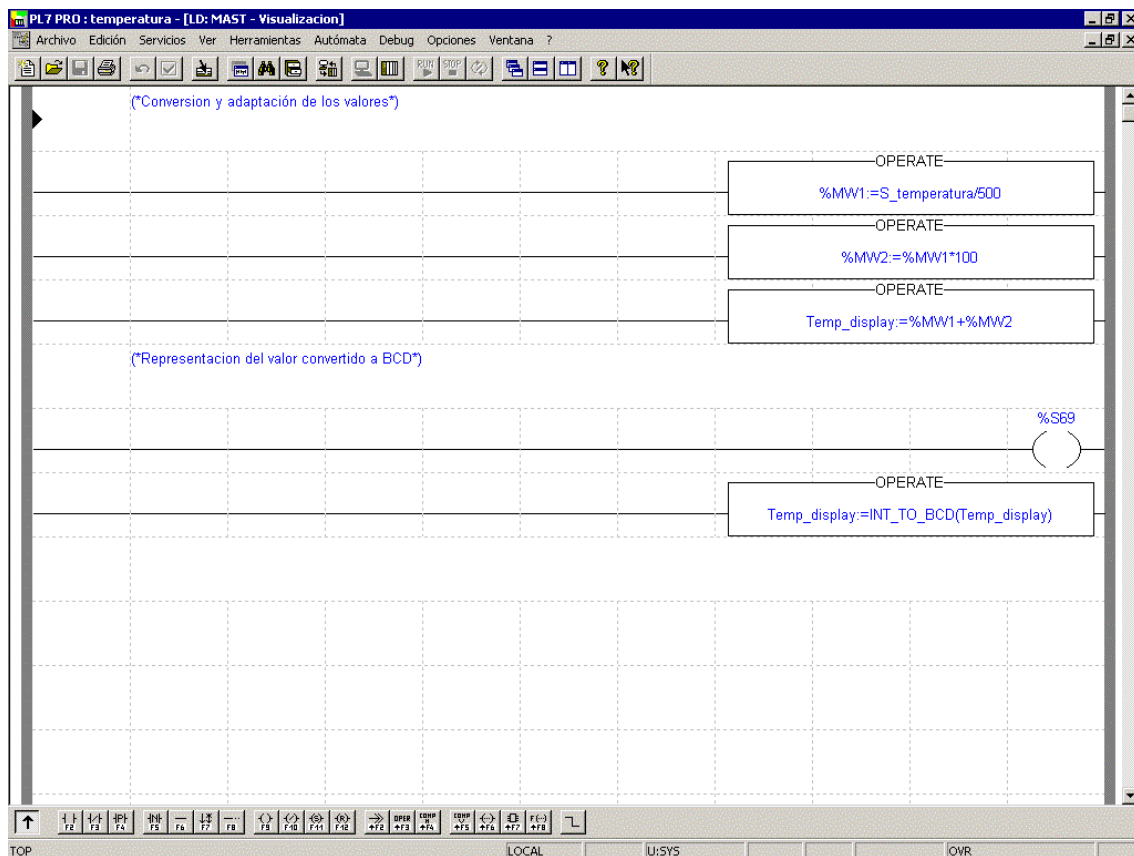
## DESARROLLO PRÁCTICA 10

El valor leído del sensor está comprendido entre 0 y 10000, y por ello es necesario convertirlo según la relación antes descrita. Por otro lado, y al no ser un proceso crítico, se han utilizado palabras para las operaciones y no dobles palabras. Esto reducirá la precisión de la medida, pero es suficiente para el proceso a realizar y dadas las especificaciones del sistema.

El display integrado representa la parte baja y alta de una palabra alternativamente, y por ello habrá que copiar el byte de menos peso (medida original) en el de mayor peso de la palabra a representar. Una forma de realizar esta operación es multiplicar por 100 el valor original y, de esta forma, estaremos desplazando 8 posiciones el dato dentro de una palabra. Por último solo queda realizar una suma sobre la palabra a representar y obtendremos un byte duplicado dentro de la misma palabra.

Conversión a temperatura:	%mw1:=0014
Copia del dato en byte de mayor peso de otra palabra:	%mw2:=1400
Suma de las dos palabras:	%mw0:=1414

De esta forma solo podremos representar valores de 0°C a 99°C pero, como el máximo es de 20°C, cumpliremos las especificaciones establecidas. Para representar el número será necesario activar el bit sistema %S69 y convertirlo a BCD. Presionando el botón **DIAG**, podremos ver los valores de las 16 primeras palabras.







## PRÁCTICA 11

### CONTROL DE POSICIÓN CON CODIFICADOR

#### OBJETIVO

Regular el desplazamiento horizontal de un cilindro montado sobre un husillo.

#### PRESENTACIÓN

En esta ocasión se necesita gestionar el almacenado lineal de unas cajas de metal que llegan a una estación mediante una cinta transportadora.

El ciclo a realizar es el siguiente:

1. Detección de la pieza desde la posición de reposo.
2. Salida del cilindro y activación del electroimán que atrae la pieza.
3. Contracción del cilindro hasta la posición de reposo.
4. Desplazamiento horizontal hacia la derecha hasta la cota adecuada.
5. Salida del cilindro y desactivación del electroimán al final del recorrido.
6. Contracción del cilindro.
7. Desplazamiento horizontal hacia la izquierda hasta la cota de reposo.

Otros datos son:

Relación de avance del husillo : 360 pulsos / vuelta = 1 cm

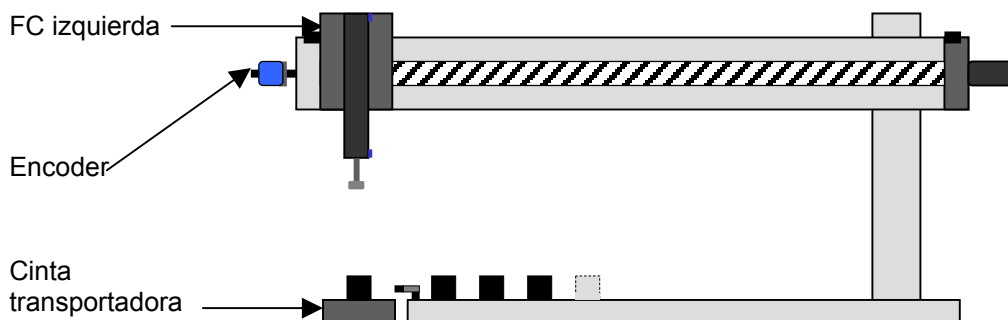
Ancho de la caja: 3 cm

Longitud de la plataforma: 80 cm (cota inicial de 10 cm)

#### Lista de variables

Marcha	%I1.0
Paro (NC)	%I1.1
Emergencia (NC)	%I1.2
Detector de piezas	%I1.3
Articulador abajo	%I1.4
Articulador arriba	%I1.5
FC izquierda del articulador	%I1.6
FC derecha del articulador	%I1.7
Salir articulador	%Q2.0
Activar electroimán	%Q2.1
Desplazamiento a la derecha	%Q2.2
Desplazamiento a la izquierda	%Q2.3
Pulsos codificador	%ID0.11

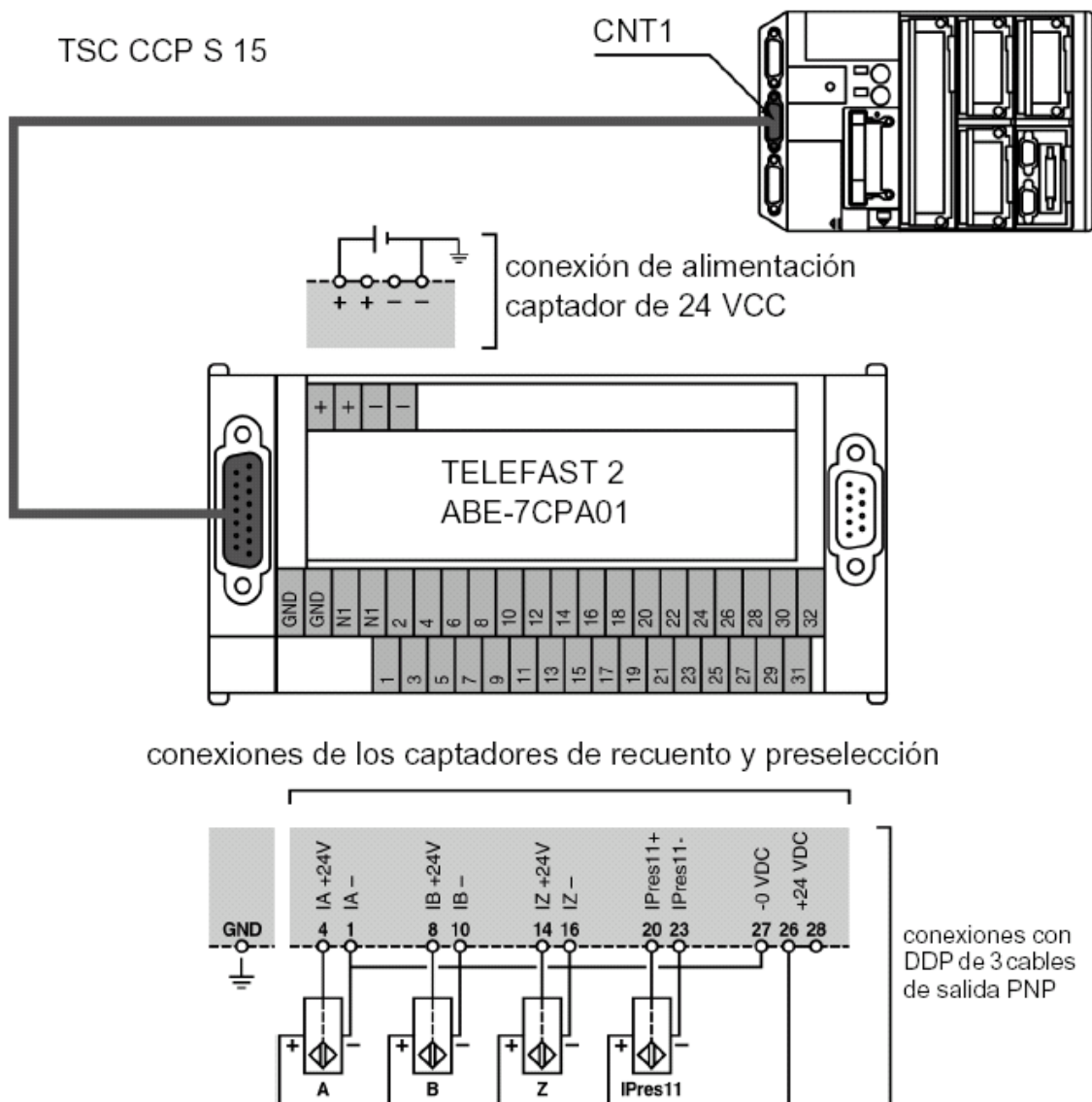
#### Esquema de la instalación



## DESARROLLO PRÁCTICA 11

Antes de comenzar a programar esta aplicación, es necesario configurar la vía de contaje a utilizar. En este caso se utilizará una de las vías integradas en el TSX Micro que soporta hasta una señal de hasta 10Kz, no siendo necesaria la adición de un módulo especial.

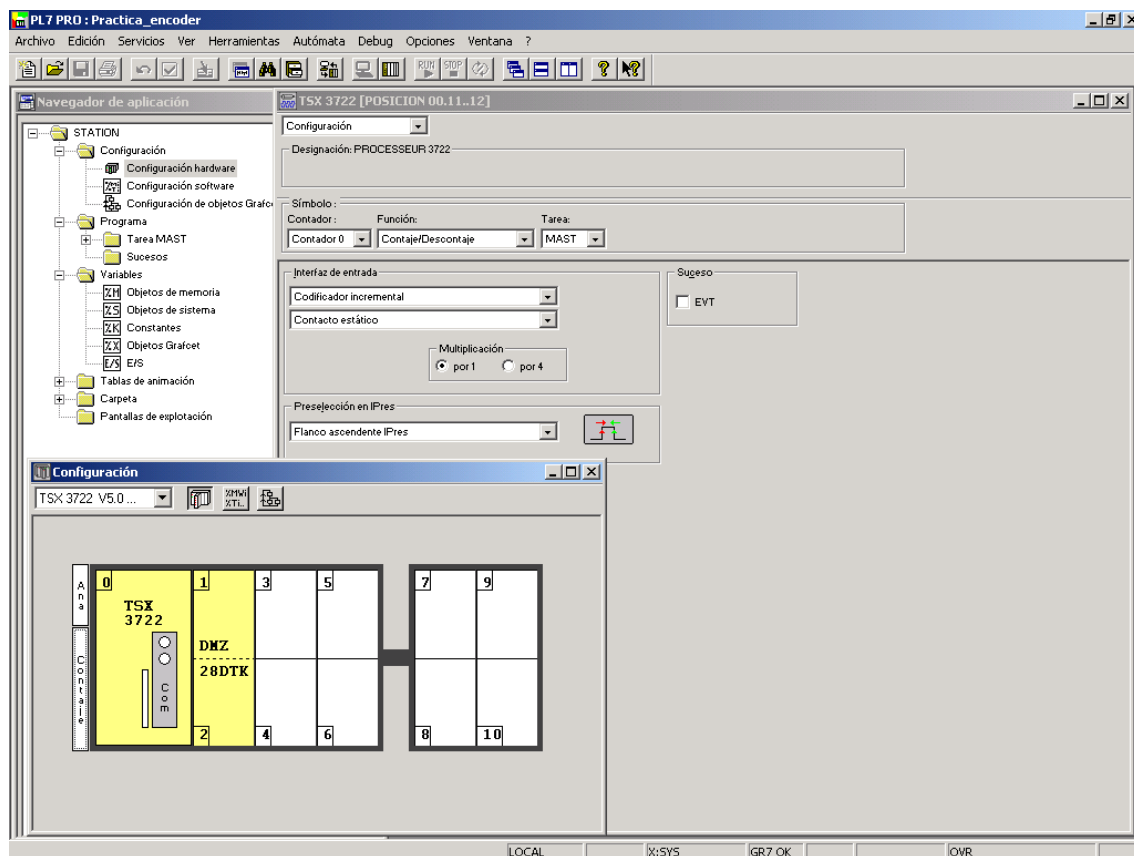
La conexión de los dispositivos que se ha utilizado es la siguiente:



## DESARROLLO PRÁCTICA 11

Una vez disponemos de la información necesaria en cuanto a la conexión de las entradas y salidas, podemos localizar las variables de las que dispondremos en nuestro programa.

La configuración del conector 1 (CNT1) de conteo podemos encontrarla dentro del navegador de la aplicación y concretamente en la opción **configuración hardware**. Mediante un doble “clic” sobre la pestaña **contaje**, aparecerá la ventana siguiente:



El contador a utilizar será el cero, que se corresponde con el CNT1 y su función por defecto la de conteo / descontaje. Por otro lado y al no ser una aplicación crítica en cuanto a seguridad se refiere, todo ello se programará dentro de la tarea maestra sin utilizar eventos del sistema.

Como interfaz de entrada se utilizará un codificador incremental con contactos estáticos. Los valores de multiplicación e IPres (entrada para la puesta en el valor de preselección) para la puesta a cero, serán los que PL7 da por defecto.

## DESARROLLO PRÁCTICA 11

En la programación de esta máquina se han decidido utilizar dos secciones que permiten una estructura mas sencilla. Dado que este es un sistema completamente secuencial, se ha implementado una sección en grafcet, que contendrá todas las secuencias de movimiento, y otra denominada encoder, que realizará la gestión del codificador.

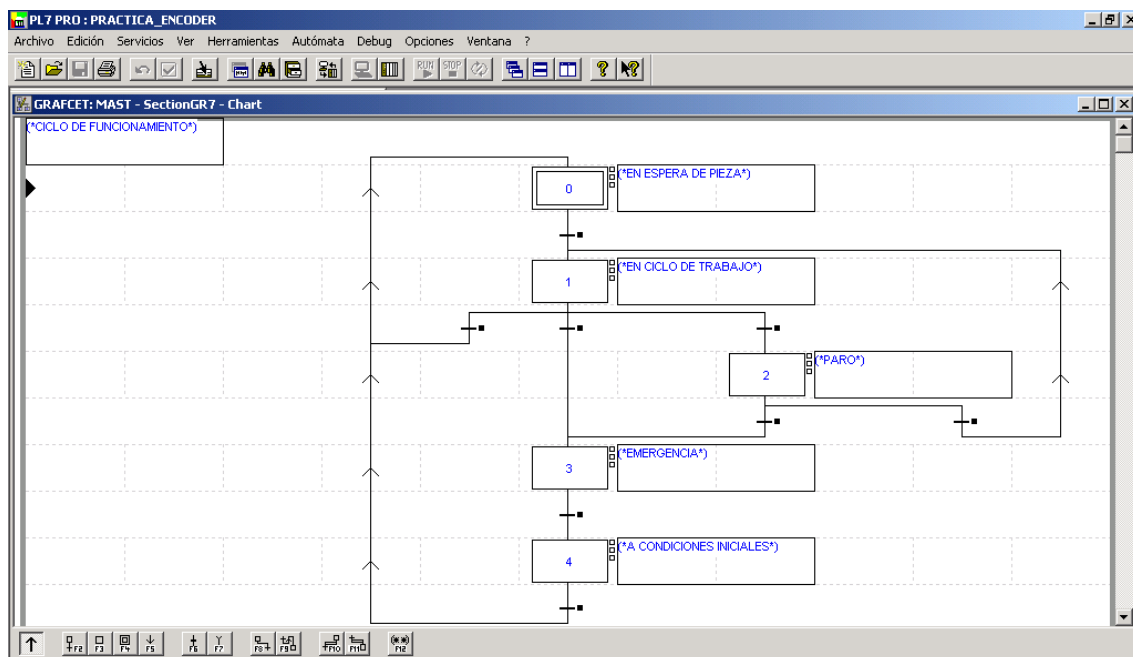
La sección chart del grafcet contiene tres gráficos:

- ✓ Grafcet para el funcionamiento general
- ✓ Grafcet para el ciclo de trabajo
- ✓ Grafcet para la vuelta a condiciones iniciales

### Grafcet de funcionamiento general

Este grafcet gestiona el modo de marcha del automatismo y representa el estado de la máquina en cada momento en función de las variables de marcha, paro, emergencia, etc. Los estados posibles son:

- Estado 0: Máquina lista para comenzar el ciclo de trabajo.
- Estado 1: Máquina realizando el ciclo de trabajo.
- Estado 2: Máquina parada.
- Estado 3: Máquina en emergencia.
- Estado 4: Máquina volviendo a condiciones iniciales.



Estando en el estado 0 (lista para trabajar), si se detecta la presencia de pieza la máquina ha de comenzar el ciclo de trabajo ó estado 1. En un ciclo normal, la máquina terminaría el trabajo y, mediante la transición situada más a la izquierda del gráfico, volvería al estado inicial.

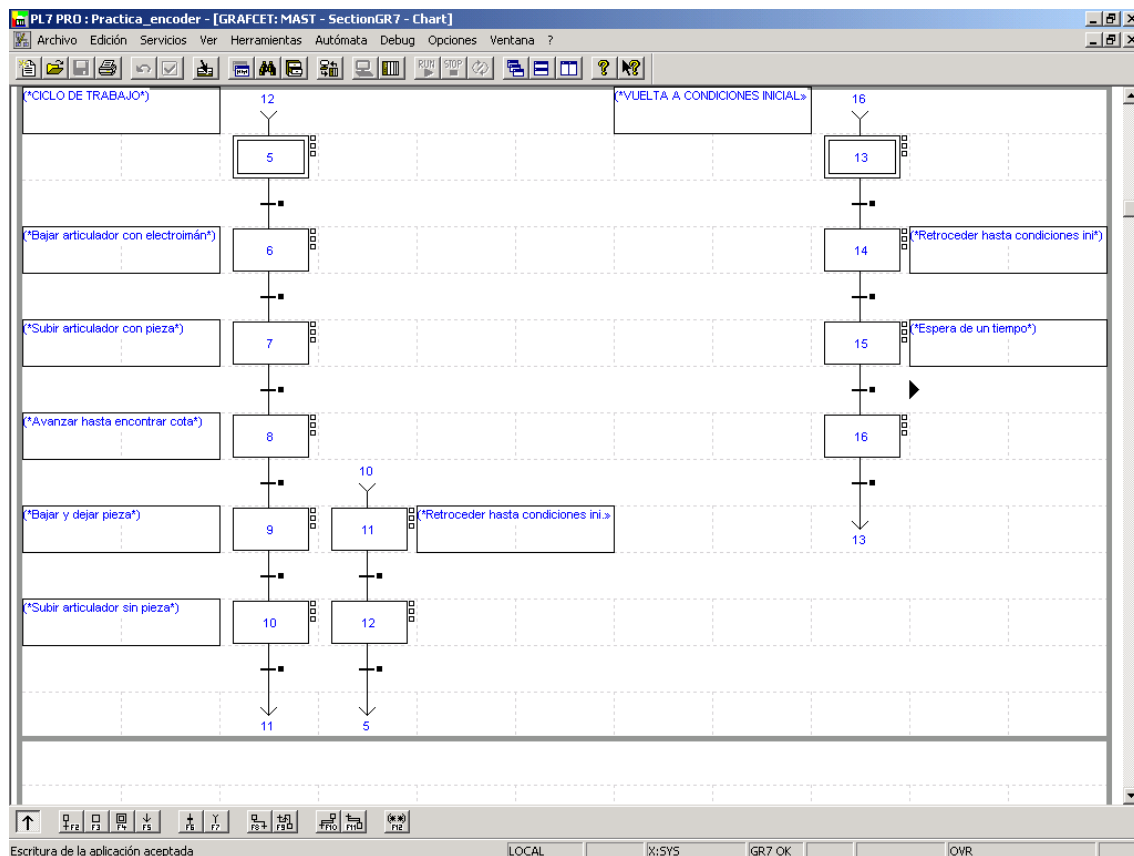
## DESARROLLO PRÁCTICA 11

Otra opción para salir del estado 1 consiste en la pulsación de un paro y de esta forma el estado 2 (máquina parada) pasaría a ser activo. Una pulsación de marcha permite pasar al estado 1 (máquina en ciclo). La última forma de salir del estado 1 consiste en activar la emergencia. De esta forma el estado activo pasaría a ser el 3 (máquina en emergencia) y sólo podemos salir de él en caso de desenclavar la emergencia y con una confirmación de marcha. El estado siguiente corresponde al cuarto (máquina volviendo a condiciones iniciales) y solo cuando la máquina ha vuelto a la posición de reposo, el estado activo será el cero ó inicial.

### Grafcet para el ciclo de trabajo

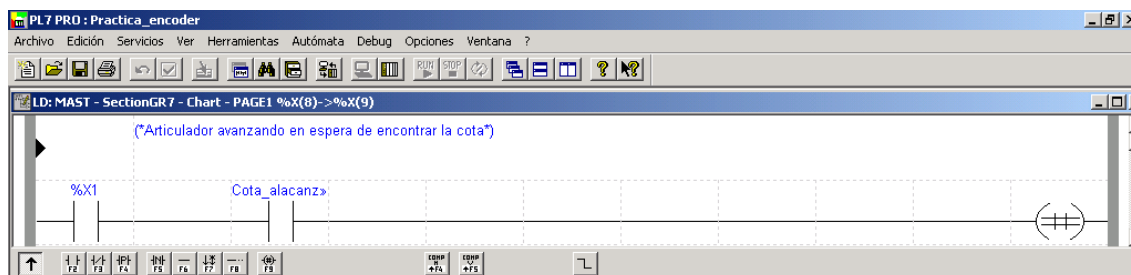
El grafcet de la izquierda representa el ciclo a realizar por la máquina una vez se ha detectado la presencia de pieza. Mientras que las condiciones permiten avanzar entre etapas, las acciones asociadas a cada etapa se han programado en la sección posterior.

La condición que permite pasar del reposo (etapa 5) a la etapa 6 en la que permitimos el avance del cilindro articulador y excitamos el electroimán, es la activación de la etapa 1 del grafcet de funcionamiento. Una vez el articulador ha llegado al final (se presupone que la pieza ha sido atrapada), hacemos retroceder el cilindro (etapa 7) hasta el máximo y cambiamos de etapa. En la etapa 8 el cilindro esta avanzando para llegar a la cota dada como consigna.



## DESARROLLO PRÁCTICA 11

La condición que indica cota alcanzada, es un bit que se activará en el momento adecuado y debido a unas líneas de programa en la sección encoder programada posteriormente.



Una vez en el estado 9, será posible bajar el articulador y soltar la pieza. El estado 10 permite retroceder el articulador para, a través del estado 11, volver a la posición inicial. El estado 12 únicamente se utiliza para que en el graficet de funcionamiento pueda activarse el estado 0 y desactivarse el 1.

Para que el graficet de ciclo avance, se ha especificado que la etapa %X1 (máquina en ciclo) ha de estar activa. De esta forma aunque se cumpla una determinada condición de salto entre etapas, si la máquina está parada (etapa %X1 desactivada y %X2 activa) el graficet de funcionamiento no avanzaría.

### Graficet para la vuelta a condiciones iniciales

Dado que al presionar la emergencia la máquina puede quedar en cualquier posición, es necesario un graficet que permita volver a condiciones iniciales después de un rearme. Este graficet se activa cuando la etapa %X4 del graficet principal o de funcionamiento pasa a ser activa.

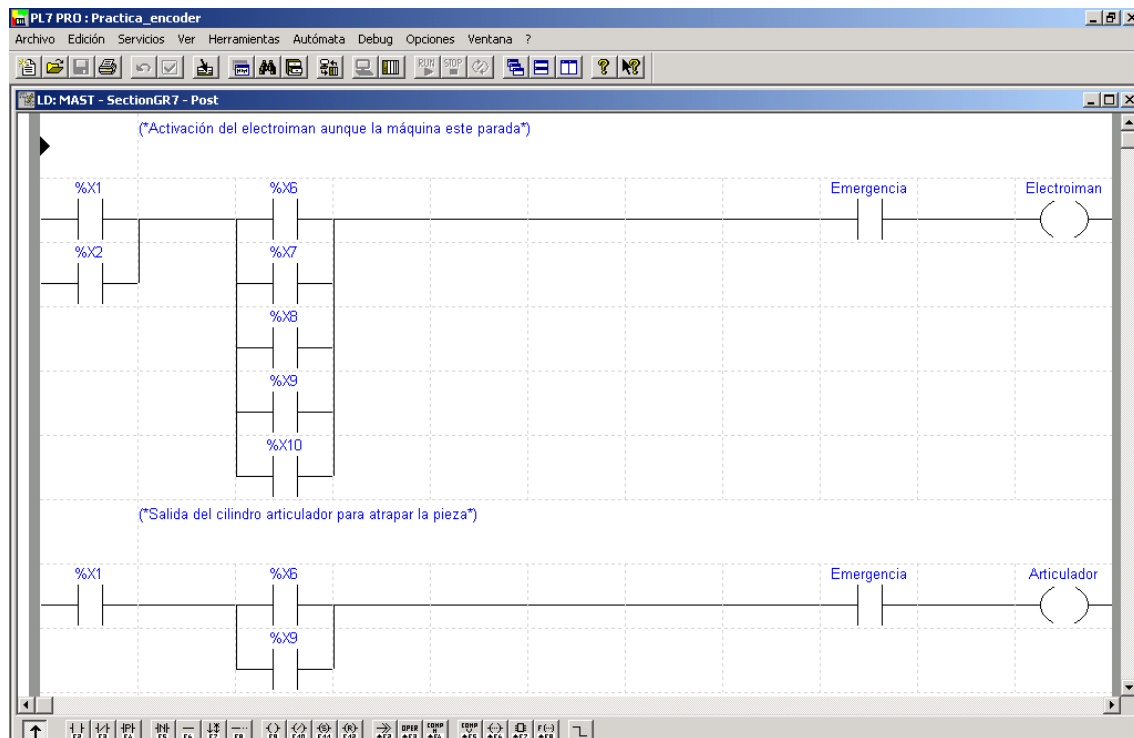
Una vez el articulador ha retrocedido hasta la posición inicial mediante la etapa 14, se espera un tiempo determinado. La última etapa se utiliza para que el graficet principal o de funcionamiento pueda activar la etapa 0 y desactivar la 4. Además, permite resetear el graficet de ciclo mediante el bit %S21 programado en el módulo preeliminar.



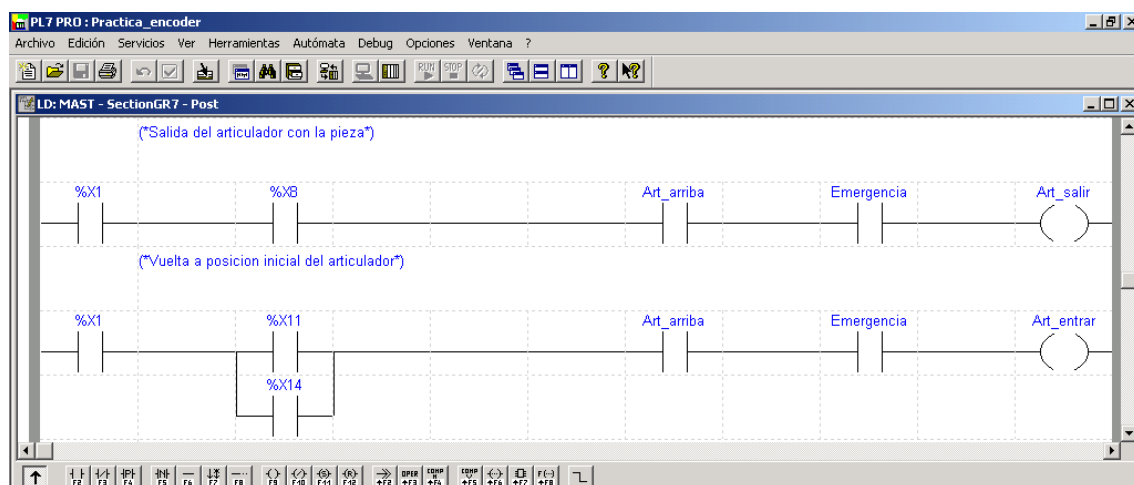
## DESARROLLO PRÁCTICA 11

La activación de las salidas se realiza en el módulo posterior del graficet y de una forma mantenida, es decir, sin realizar set ó reset sobre los bits de salida. Dado que no deseamos que la pieza se caiga en una parada de la máquina, esta salida se activará en los estados %X1 (en funcionamiento) y %X2 (parada) cuando el estado correspondiente del ciclo de funcionamiento se active.

A su vez, el articulador o cilindro vertical se activará si está en el ciclo de trabajo y en las etapas correspondientes.



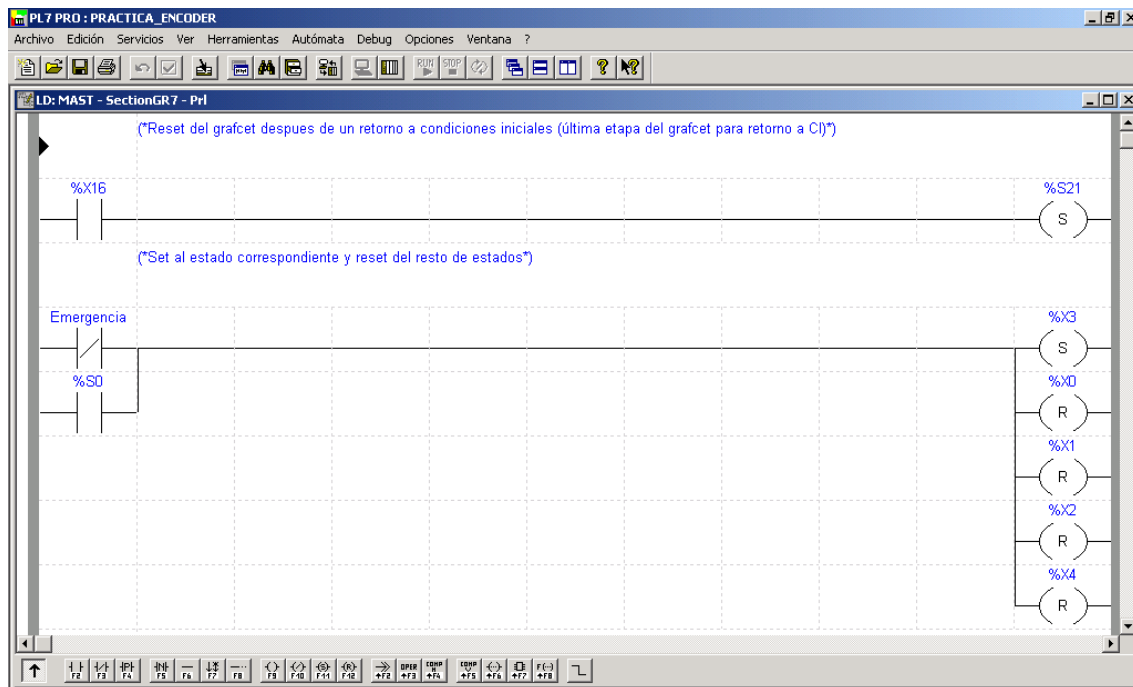
Esta misma metodología de trabajo se utilizará para desplazar horizontalmente el articulador, añadiendo seguridades mecánicas para no desplazar el articulador si este no está arriba.



## DESARROLLO PRÁCTICA 11

El módulo preeliminar del grafcet contendrá las instrucciones necesarias para el posicionamiento del gráfico. La primera instrucción reinicializa el grafcet mediante la última etapa del grafcet de vuelta a condiciones iniciales.

La siguiente red de instrucciones permite posicionar el grafcet en el estado de emergencia (estado 3) en caso de que esta sea activa.



## DESARROLLO PRÁCTICA 11

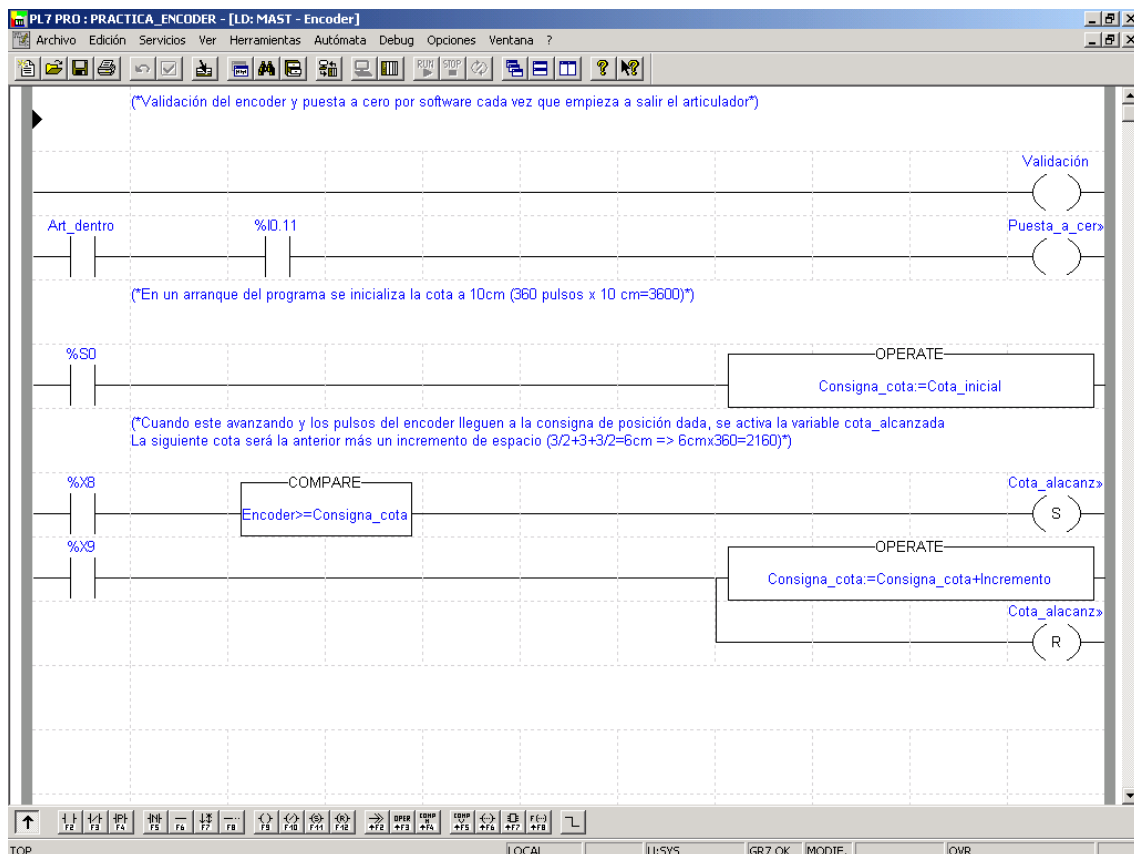
Tal y como se ha comentado al principio de la práctica, la sección encoder reúne todas las instrucciones necesarias para la gestión del codificador.

La puesta a cero del codificador se realizará por software y cuando el articulador este dentro ( **Art\_dentro** ) con la señal de paso por cero ( **IZ** ) activa para que todas las cotas tengan la misma referencia de posición. La validación del codificador es continua dado que no se especifica lo contrario.

Para inicializar la cota donde se colocará la primera pieza utilizaremos el bit sistema %S0, que es activado automáticamente en el rearranque del sistema. Una especificación del sistema será que cada vez que haya un retorno de la alimentación, se eliminarán todas las piezas almacenadas y se empezará a colocar desde la cota inicial.

Estando en la etapa del grafcet correspondiente, se realizará la comparación de la consigna con el registro del codificador. Cuando el registro supere a la consigna, se activará el bit de cota\_alcanzada correspondiente que se había comentado en el grafcet de funcionamiento como transición de la etapa 8 a la 9.

Es en la siguiente etapa cuando la nueva consigna pasa a ser la anterior mas el incremento de espacio calculado. Además desactivamos el bit cota\_alcanzada.





## PRÁCTICA 12

### DIÁLOGO HOMBRE MÁQUINA CON PRODYN

#### OBJETIVO

Realizar las pantallas de explotación que permiten la supervisión de una instalación.

#### PRESENTACIÓN

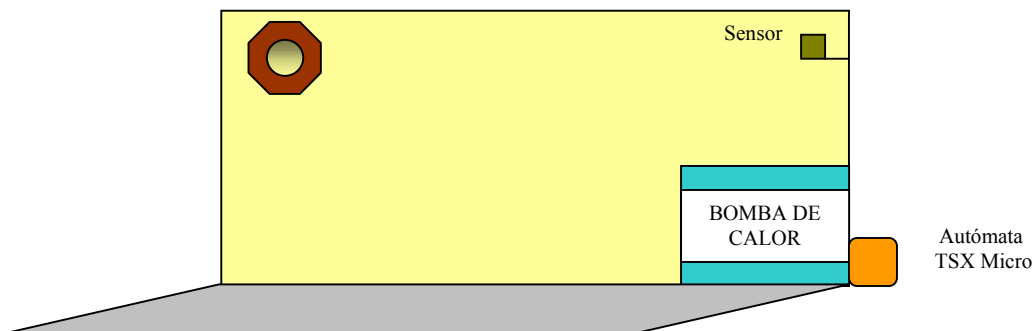
La automatismo de la práctica 10 forma parte de una instalación compleja, y se pide desarrollar las pantallas de explotación que permitirán la supervisión de esta sección desde una sala de control central.

Desde esta pantalla de visualización, únicamente podrán visualizarse las variables que más abajo se describen.

#### Lista de variables

Marcha	%I1.0
Paro (NC)	%I1.1
Sensor de temperatura	%IW0.5
Salida de calor	%Q2.0
Salida de frío	%Q2.1

#### Esquema de la instalación

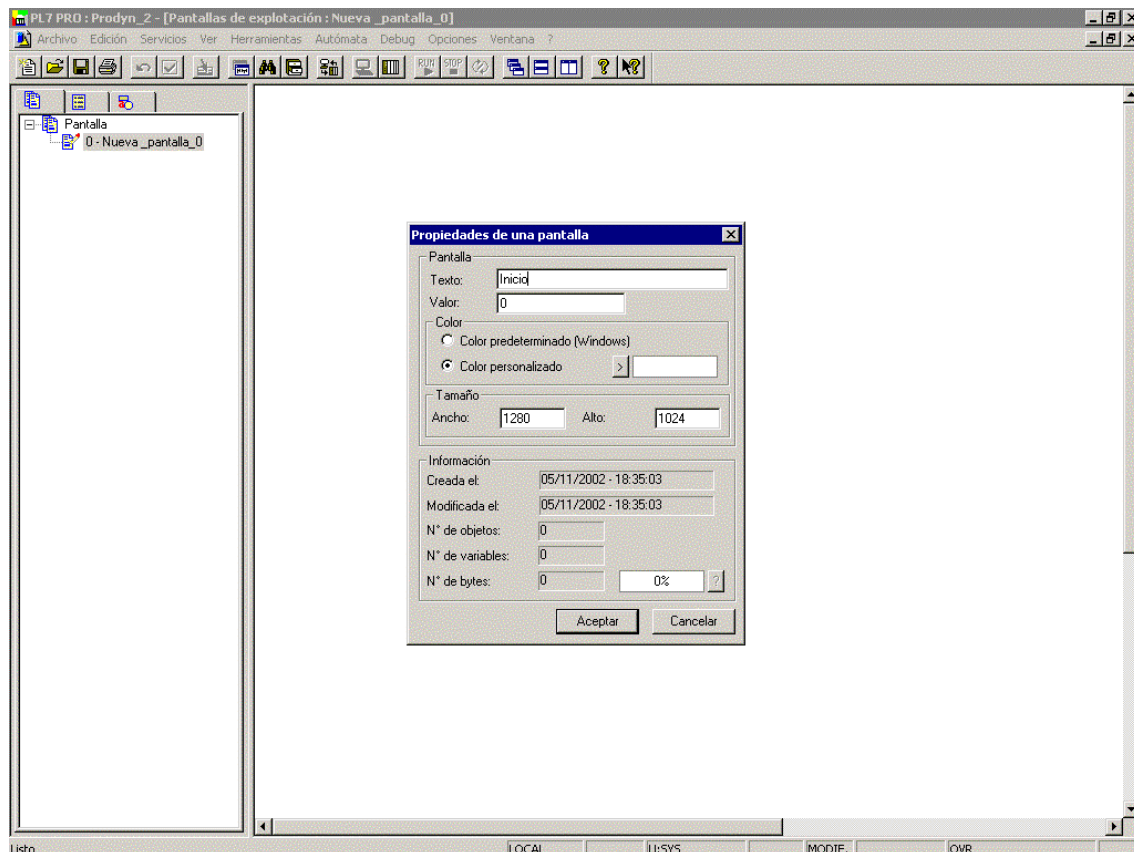


## DESARROLLO PRÁCTICA 12

Para desarrollar la supervisión de esta instalación con el editor de pantallas, es necesario entrar en la opción pantallas de explotación del navegador principal. La pantalla que aparece contiene un navegador con tres pestañas. La primera permite crear las diversas pantallas de explotación que formarán nuestra aplicación. La segunda permite crear mensajes que posteriormente utilizaremos y la última pestaña contiene una biblioteca de objetos gráficos predefinidos.

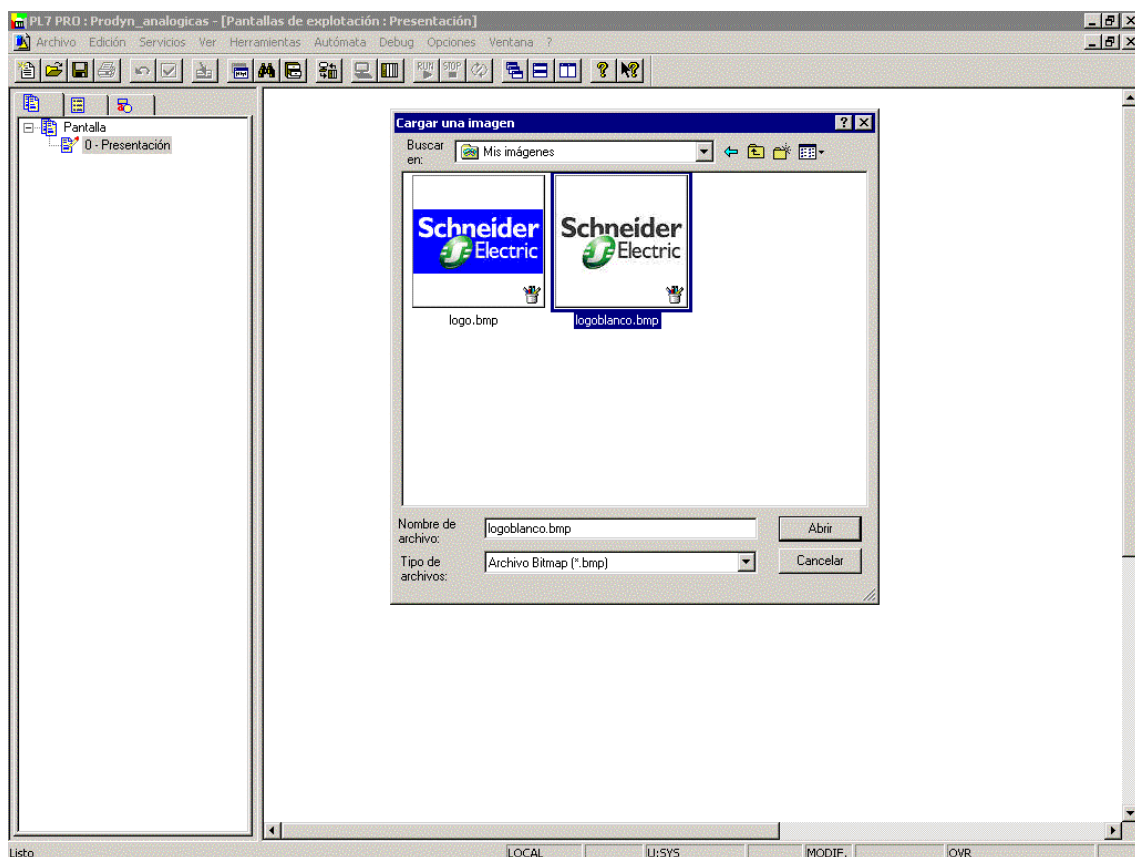
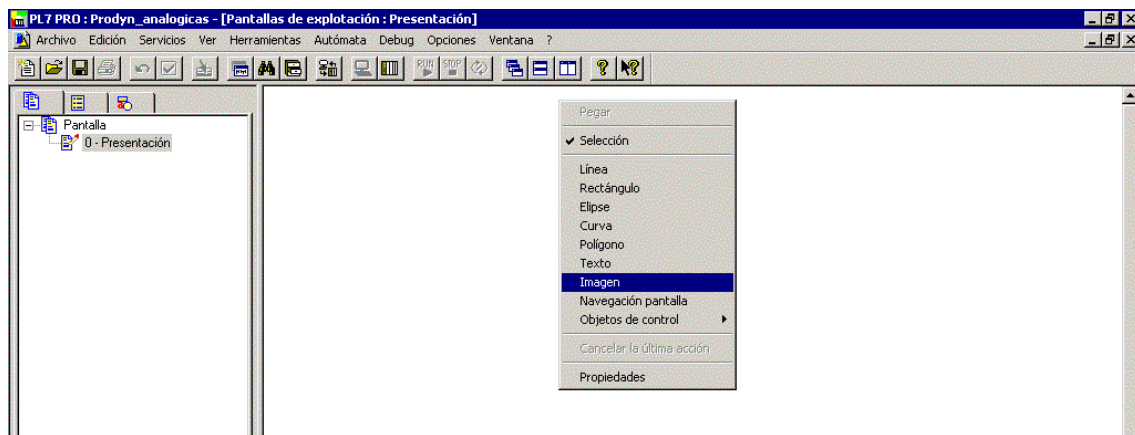
Para crear una pantalla nueva, basta con hacer “clic” sobre el navegador con el botón derecho del ratón. Seleccionando la opción **Crear** añadiremos una página nueva denominada **0\_Nueva pantalla**.

Podemos acceder a las propiedades de esta nueva pantalla mediante el botón derecho del ratón y haciendo “clic” en **Propiedades**. En la pantalla que aparece podremos dar un nombre a la página y asignarle un número de identificación. En nuestro caso se asignará el nombre de **Presentación** y el número **0**.



## DESARROLLO PRÁCTICA 12

Una vez definida la página principal, añadiremos los componentes de visualización necesarios para realizar una presentación correcta. En este caso concreto se ha decidido que la página principal contenga el logo de la empresa cliente. Para insertar esta imagen, basta con hacer “clic” con el botón derecho del ratón situados encima de la página. Seleccionando la opción **Imagen** podremos insertar un archivo de tipo \*.BMP.

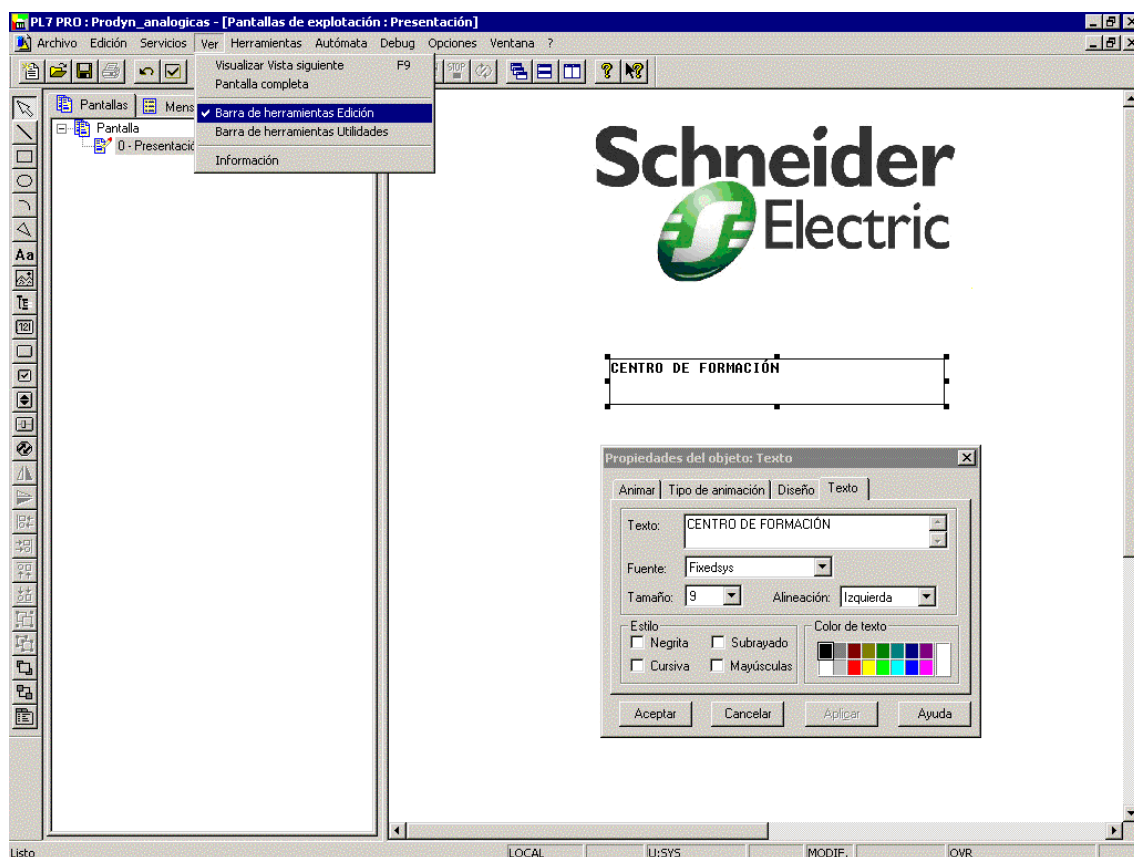




## DESARROLLO PRÁCTICA 12

Mediante la barra de herramientas podremos insertar los controles necesarios en nuestra aplicación. Esta opción se encuentra en el menú **Ver->Barra de Herramientas->Edición**. Seleccionando la opción texto ( **Aa** ) en la parte izquierda de la pantalla es posible delimitar un área de introducción mediante un "clic" en la pantalla de explotación.

Una vez delimitada el área del texto, podemos hacer "clic" con el botón derecho del ratón sobre esta misma figura y seleccionar la opción **Propiedades**. La pantalla que aparece permite configurar todos los atributos del texto: color, fuente, tamaño y el mismo texto que aparecerá. La pestaña **Animar** permite crear animaciones de este mismo texto que por el momento no serán necesarias.

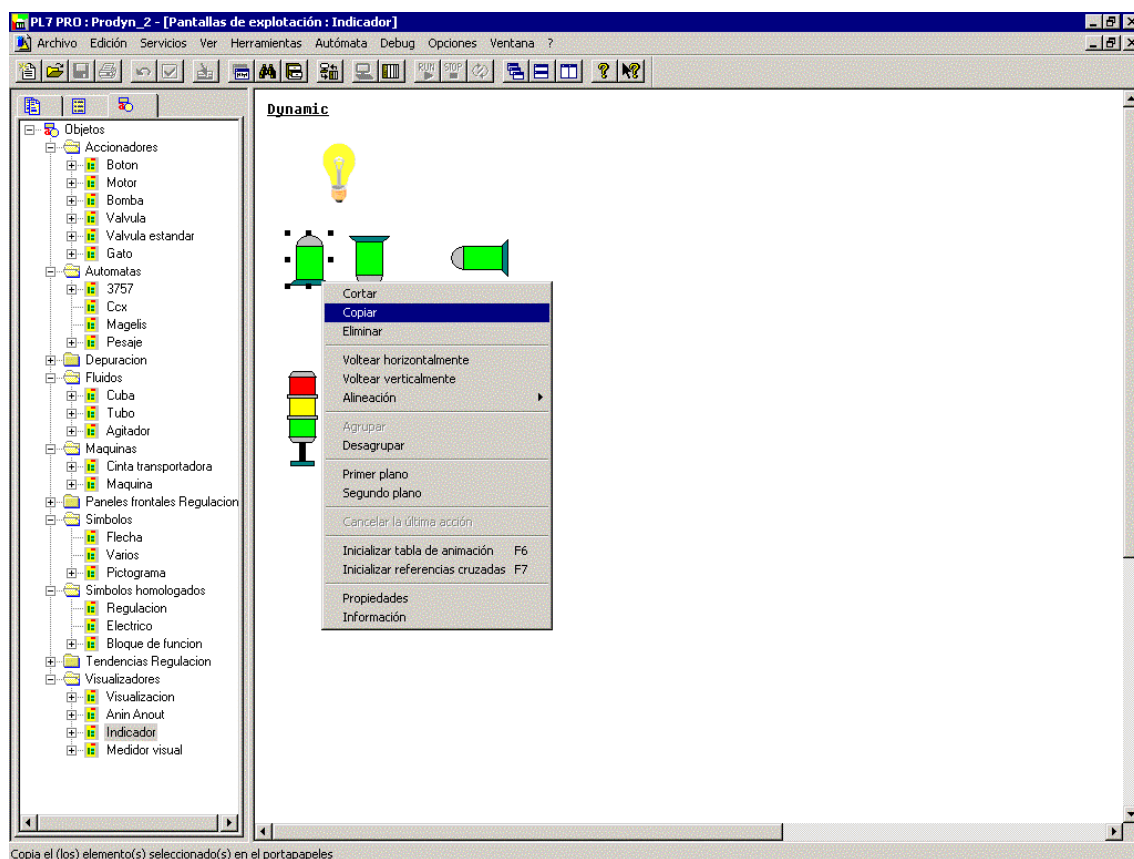


## DESARROLLO PRÁCTICA 12

Una vez finalizada la página de presentación, podremos crear otra página de la misma forma que la página cero y que contendrá los diferentes elementos a supervisar.

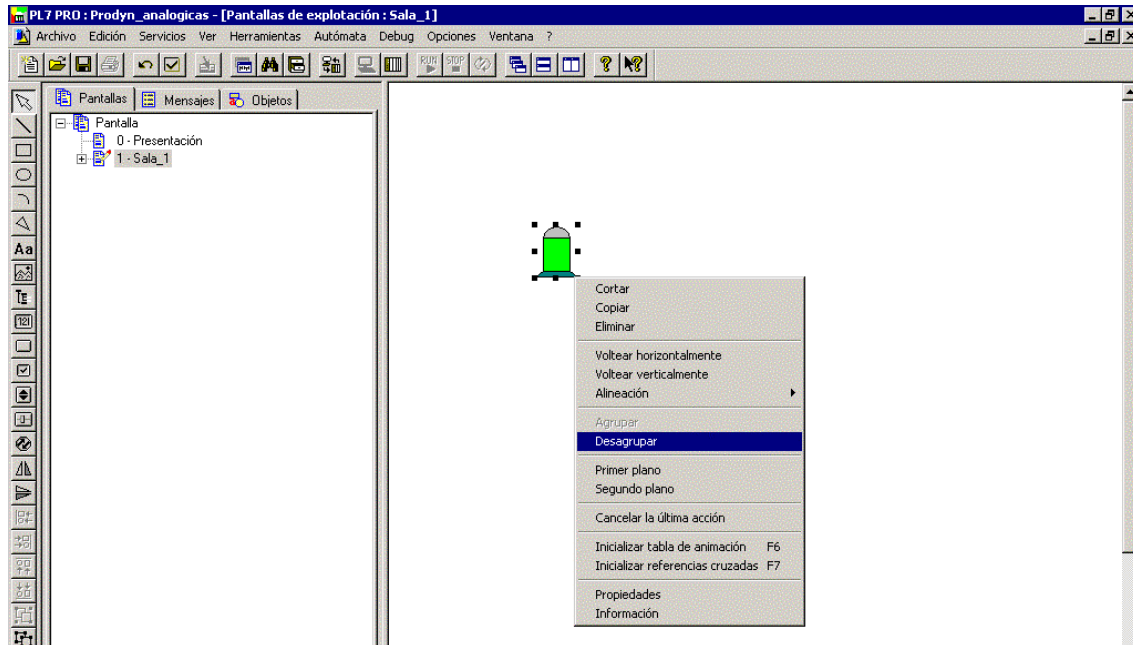
Para ello recurriremos a la biblioteca de objetos que PL7 lleva incorporada seleccionando la pestaña **Objetos** del navegador. Esta biblioteca, organizada por carpetas, contiene objetos gráficos predefinidos listos para utilizar en nuestra aplicación. Botones, válvulas, cintas transportadoras, símbolos, paneles de regulación y visualizadores son algunos de los elementos disponibles. Una vez identificado el elemento que necesitamos, la metodología de trabajo consiste en copiar el elemento y pegarlo dentro de la página creada anteriormente.

Para el desarrollo de esta práctica utilizaremos un indicador de marcha – paro de la instalación.

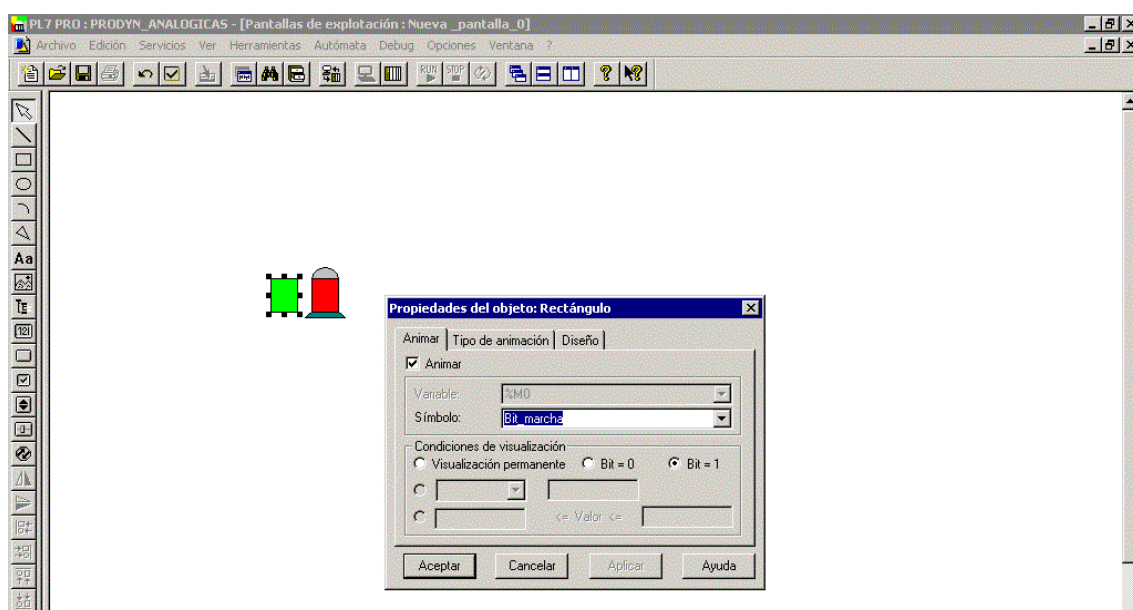


## DESARROLLO PRÁCTICA 12

Una vez colocado el elemento en la posición adecuada, deberemos seleccionar la opción desagrupar que contiene el menú al seleccionar el objeto con el botón derecho del ratón.



Esta opción permite desagrupar un objeto que esta formado por diferentes capas o dibujos. En nuestro caso al desagrupar el objeto vemos dos rectángulos de diferente color que identificaremos con la marcha y el paro. Cada una de estas partes posee sus propiedades a las cuales podemos acceder haciendo “clic” con el botón derecho del ratón y la misma opción de **propiedades**.



## DESARROLLO PRÁCTICA 12

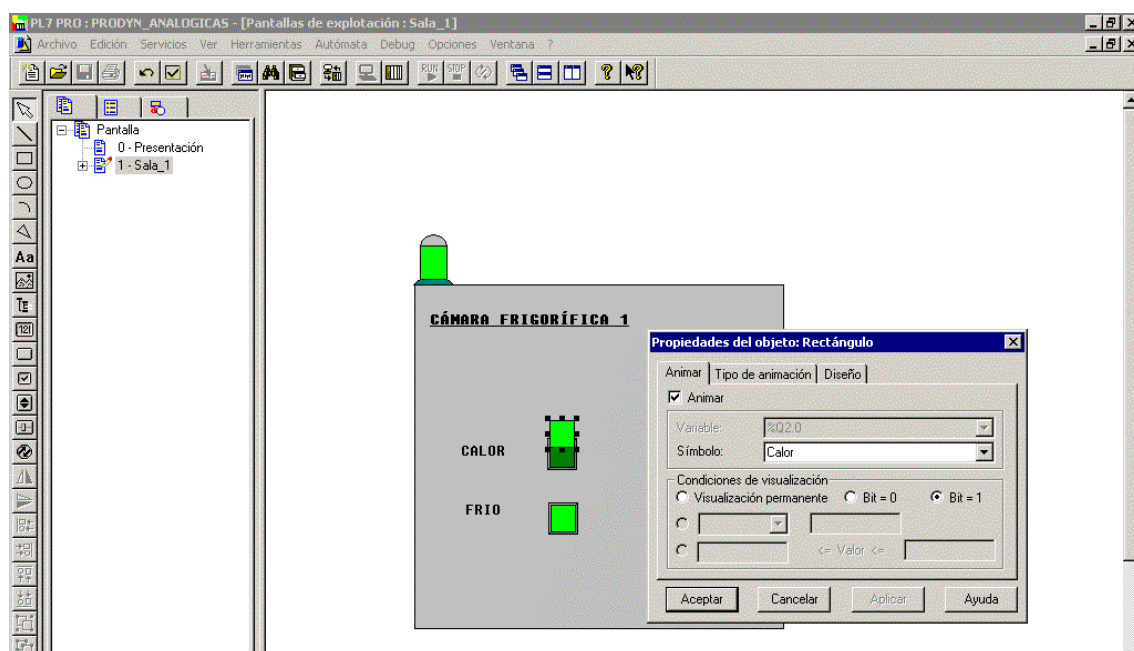
Si es necesario animar esta figura, seleccionaremos la pestaña animar del menú propiedades indicando la misma opción en la casilla de verificación. El recuadro símbolo permite escribir la variable de la cual dependerá la visualización o no de este objeto. Por otro lado, el recuadro para las condiciones de visualización permite determinar el valor de esta variable que hace visible el objeto.

En nuestro caso concretamente y recordando el programa escrito en la práctica anterior, utilizaremos el bit de memoria %M0 (bit\_marcha). Seleccionando el rectángulo verde y esta misma variable, vemos que la condición de visualización es en estado 1. Para el caso de la señalización de paro seleccionaremos el rectángulo rojo y como símbolo el mismo bit\_marcha. Sin embargo, en este caso el estado de visualización será el estado 0. Una vez hayamos definido las propiedades de estos dos elementos, solo queda devolver el objeto a su forma original agrupando los dos cuadrados.

Dado que en este sistema interesa indicar también la salida que se activa en cada momento, colocaremos dos indicadores (frío y calor) que configuraremos previamente. La forma de proceder es similar a la anterior:

1. Seleccionar el objeto de la biblioteca.
2. Pegar el objeto en la zona deseada.
3. Desagrupar el objeto.
4. Editar las propiedades de cada uno.
5. Agrupar el objeto.

En esta ocasión las variables a utilizar pueden ser la misma salida de calor (%Q2.0) y la de frío (%Q2.1) ambas en el estado 1. Finalmente, un texto descriptivo de la instalación y de cada indicador ayudará a una mejor percepción de la aplicación.

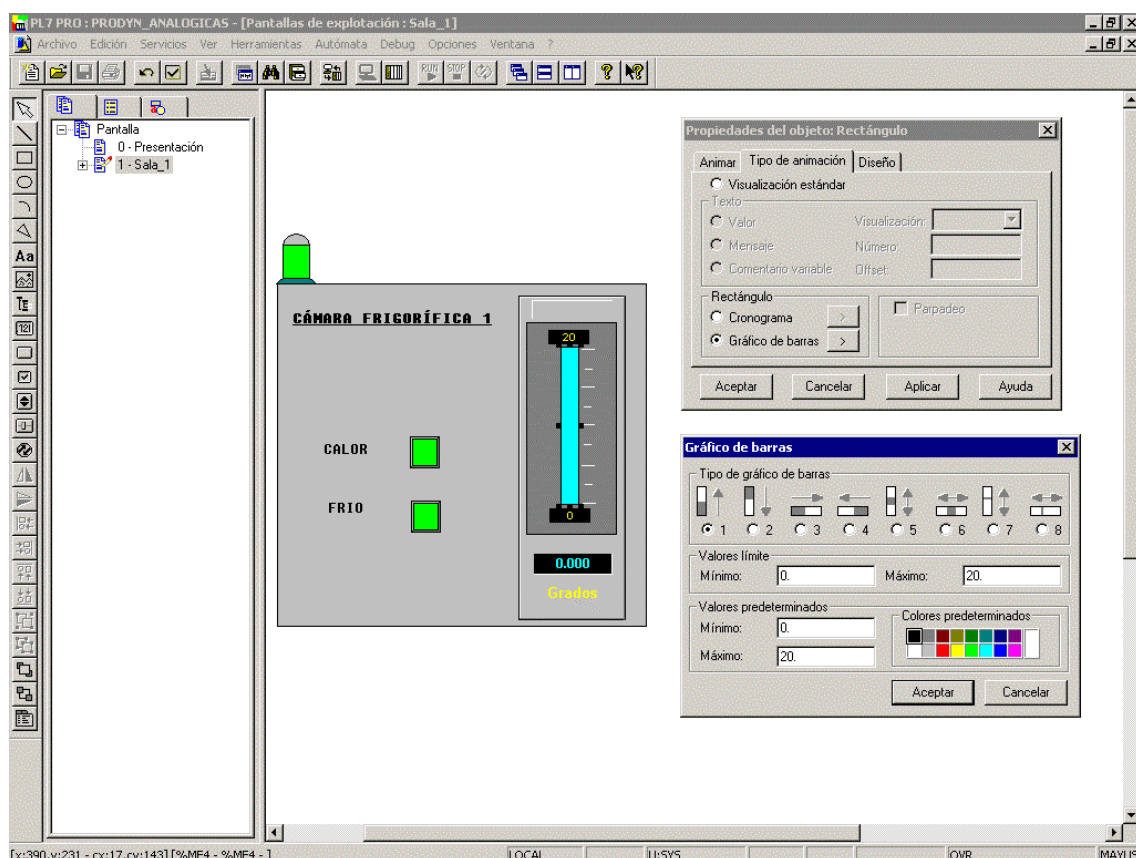




## DESARROLLO PRÁCTICA 12

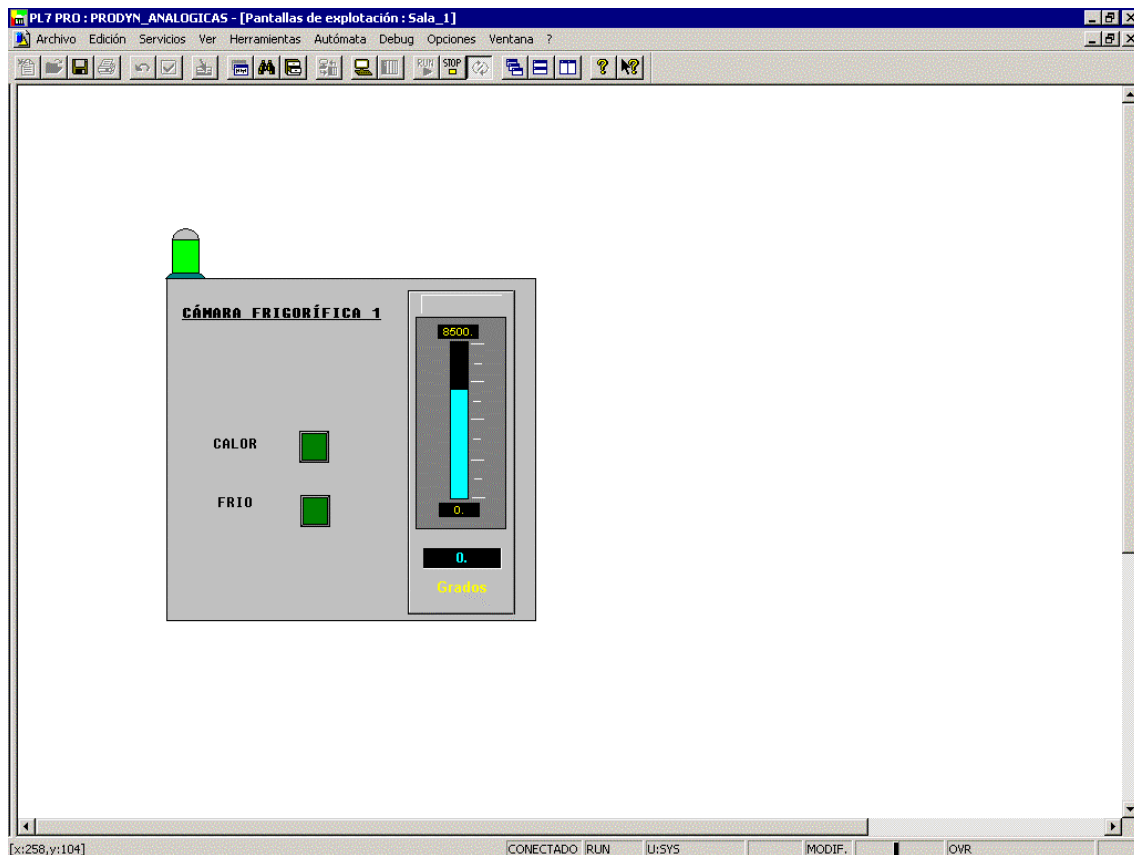
Para visualizar el valor de temperatura deseado, se utilizará un control analógico de barras. Si seleccionamos este objeto y editamos sus propiedades, dentro de la pestaña **Animar** podremos encontrar la variable que representaremos en el gráfico de barras. En este caso se trabajará con %MF4 y una visualización permanente.

La siguiente pestaña permite definir el tipo de animación del gráfico. Los valores máximos y mínimos del gráfico a representar se tomarán de las casillas **valores límites**. Por otro lado, la variable %MF8 representará el mínimo permitido y que en este caso será de 20.

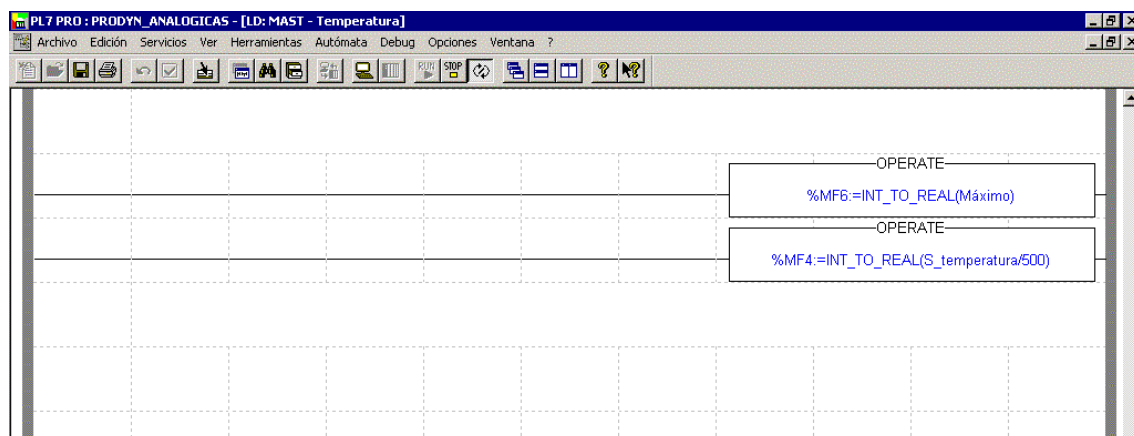


## DESARROLLO PRÁCTICA 12

Finalmente podremos disponer de la pantalla de explotación tal y como se representa en la figura.



La modificación que es necesaria realizar en el programa Ladder original consiste en cargar los valores correspondientes a temperaturas en las variables que se habían utilizado. En este caso %MF6 almacena el máximo y %MF4 almacena el valor de temperatura leído. Ambos definidos como reales.







## PREGUNTAS TÉCNICAS MAS FRECUENTES

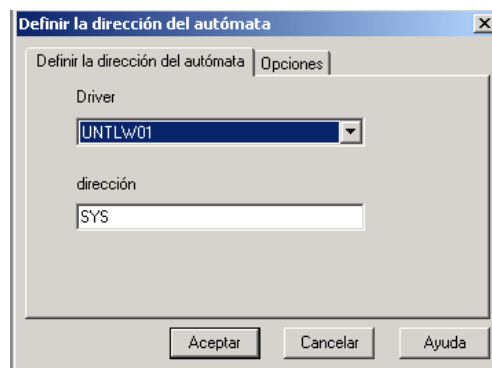
**Al insertar un bloque predefinido (timer, contador, etc) aparece el mensaje: Objeto gráfico no predefinido.**

Este mensaje aparece si el número del objeto gráfico es superior al configurado en la opción **Configuración → Configuración Software**. Ajuste el número de bloques predefinidos al numero deseado.

**Que problema puede darse si no es posible establecer la comunicación entre el PC y el autómata.**

Verificar que el selector del cable de programación TSXPCX1031 está en la posición TER DIRECT (posición número 2). Si se utiliza un cable antiguo, verificar que tenga la referencia TSXPCU1031.

En caso de no poder conectar aún utilizando el cable correcto, verificar que la opción de PL7 **Autómata → Definir la dirección del autómata**, esté configurada de la siguiente forma :



**¿Que resolución posee el reloj interno del TSX Micro?**

La resolución del reloj interno (palabra sistema %SW50) es de un segundo +/- 200ms dado que esta última cifra es el ratio de refresco del RTC.



## WEBGRAFÍA Y BIBLIOGRAFÍA

<http://www.schneiderelectric.es>

Página principal de Schneider Electric España con multitud de recursos en las áreas de automatismos, control industrial, baja tensión y media tensión. Noticias, novedades técnicas y miniwebs para productos exclusivos.

<http://www.transparentfactory.com>

Site con gran cantidad de información bien estructurada sobre autómatas, diálogo hombre máquina, comunicaciones y todos los aspectos relacionados con el mundo de la automatización. Dispone además de un tablón para preguntas técnicas más frecuentes.

<http://webplc.schneiderelectric.es>

Servidor web DEMO para un TSXMICRO

Piedrafita Moreno, Ramón. **Ingeniería de la automatización industrial**

Ediciones RA-MA, 2000

ISBN84-7897-604-3

### **Manual de referencia PL7 Micro / Júnior / Pro**

Descripción detallada de las Instrucciones y Funciones; ed. 2002

TLX DR PL7 SPA

### **Autómatas TSX Micro 3722**

Manual de puesta en marcha, Tomo 1; ed. 2002

TSX DM 37 SPA

### **Guía de programación**

Manual del usuario de los módulos TSX ETZ 410/510

Spa Versión V1.1; ed. 2002

### **Altivar 58**

Manual de programación y puesta en marcha

VW3-A58101; ed. 2002