SoMachine

Manual de Prácticas

Schneider Electric

ADVERTENCIA

Todos los ejemplos desarrollados en este manual son de tipo pedagógico y por ello pueden no ser fiel reflejo de la realidad.

Los productos presentados en este manual son susceptibles de evolución en cuanto a sus características de presentación, de funcionamiento o de utilización. Su descripción en ningún momento puede revestir un aspecto contractual.

El Instituto Schneider Electric de Formación, acogerá favorablemente cualquier solicitud con fines didácticos exclusivamente, de utilización de gráficos o de aplicaciones contenidas en este manual.

Somachine

Manual de prácticas de SoMachine V4.1

Creado por: Juan Olea Pastor

Fecha: Julio 2015

Versión: 1.0

SCHNEIDER ELECTRIC ESPAÑA

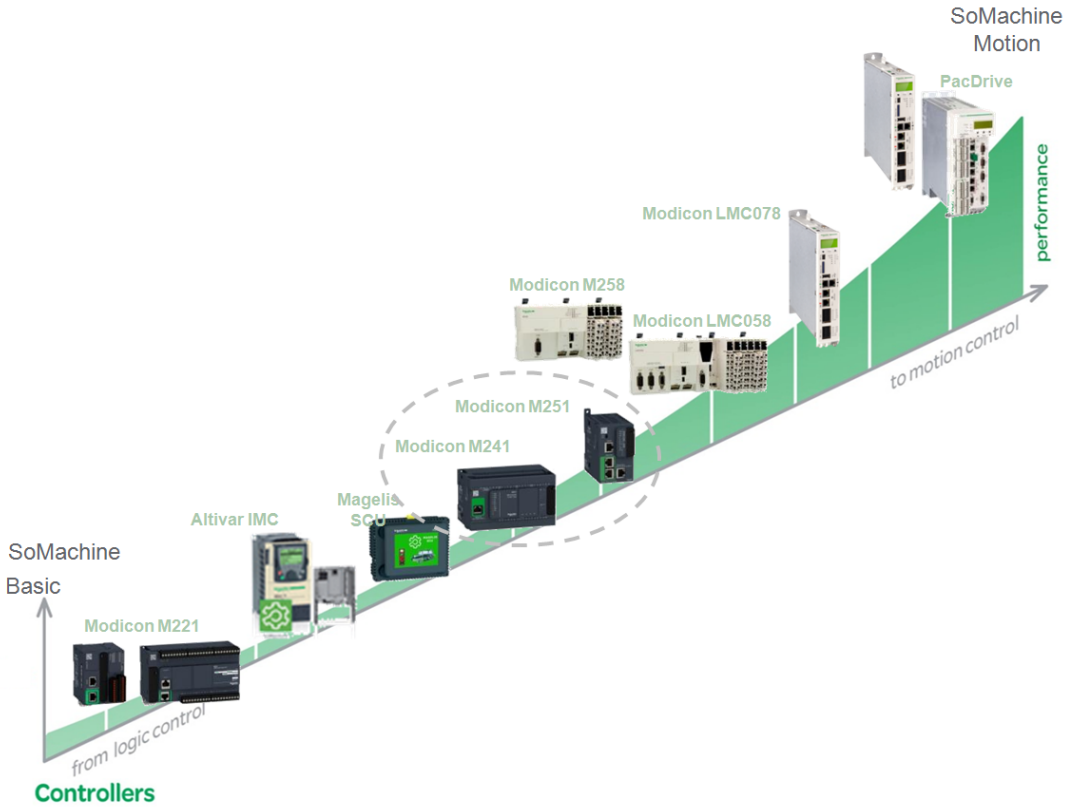
Índice

CAPÍTULO

1

1. MÁQUINAS COMPACTAS Y MODULARES

El software SoMachine nos permite programar diferentes modelos dentro de la gama de máquinas compactas. Entre estos destacan el **M241** y **M251**, como gama de PLC, pero también incluye la posibilidad de programar pantallas HMI con y sin controlador propio, y controladores de motores como la interpoladora de ejes **LMC058** ó **LMC078** para servomotores. En este manual utilizaremos los controladores lógicos (M241 y M251) sobre todo, ya que para los otros se ampliarán manuales para HMI y controlador de motores.



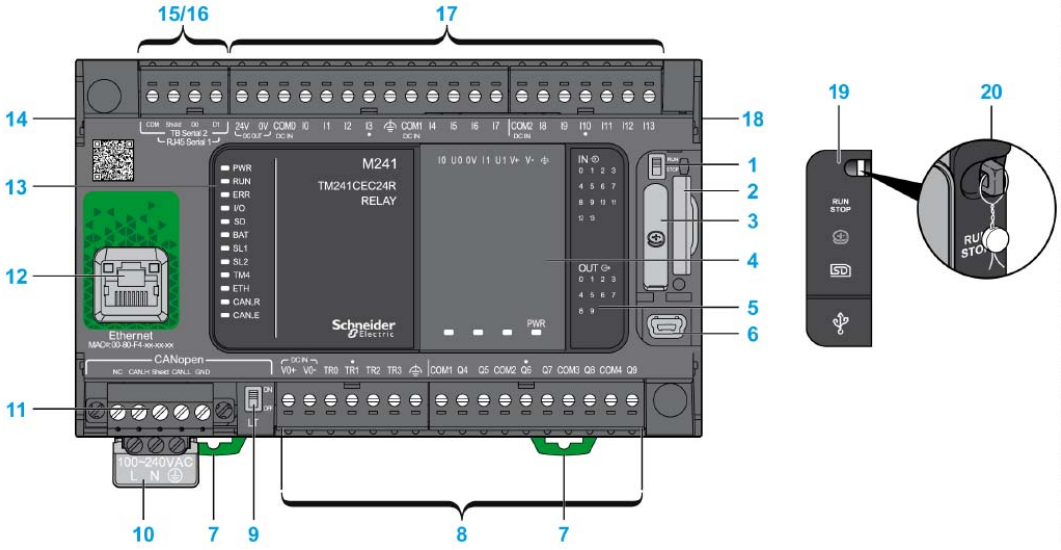
1.1 CONTROLADOR LÓGICO MODICON M241

Existen varios modelos de controladores M241con distinta alimentación y diferentes puertos de comunicación embebidos.

El modelo de controlador, con el que vamos a realizar el desarrollo de este manual SoMachine es el **TM241CEC24R** el cual se alimenta con **230 VAC** y dispone de:

- **14 entradas de 24 VDC**, de estas 14 entradas hay **8 rápidas (hasta 200 KHz)**, específicas para función especiales como el contaje de alta velocidad (**HSC**).
- **4 salidas de transistor de 24 VDC**, específicas para funciones especiales como **PWM** (modulación de amplitud de impulsos, **20 KHz**) y **PTO** (salidas de tren de impulsos, **100 KHz**).
- **6 salidas de relé** (soporta hasta **2 A** de corriente por salida) hasta 20 millones de ciclos.
- Dos puertos serie **RS 232/RS 485** (protocolos SoMachine-Network, Modbus, ASCII)
- Un puerto **Ethernet**, protocolo Modbus TCP (cliente/servidor) o servidor web.
- Un puerto maestro de bus **CANopen** (hasta **63 esclavos**).

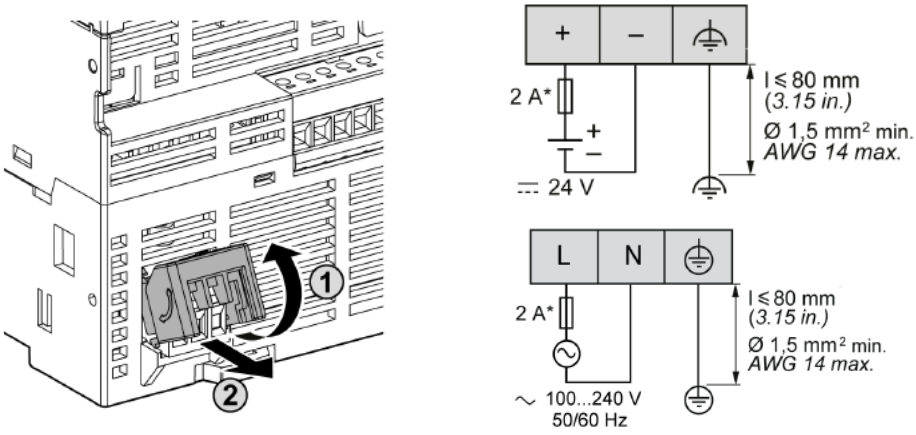
Descripción y cableado del Controlador



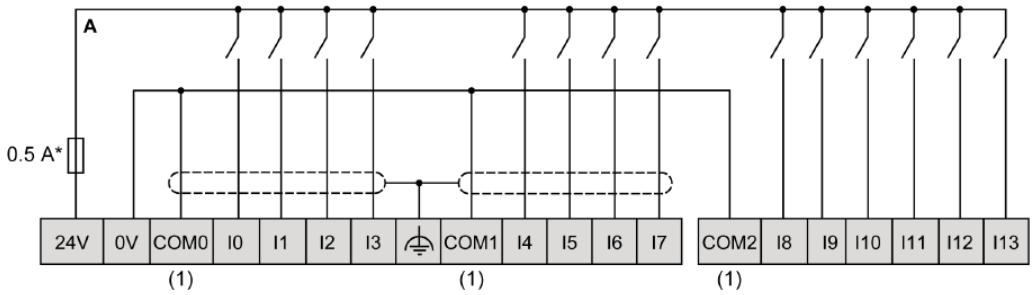
1 Interruptor de Run/Stop
2 Ranura para tarjeta de memoria SD
3 Ranura para la batería de reserva
4 Ranuras para cartuchos de E/S o cartuchos de aplicación.
5 Bloque de visualización LED que muestra el estado de las E/S
6 Conector USB mini-B para programación
7 Pestaña para bloqueo en carril 5 simétrico
8 Conexión de salidas lógicas de relé/transistor: borneros de tornillos extraíbles.
9 Interruptor de resistencia final de línea del bus CANopen.
10 Bornero de tornillos extraíble, 3 bornas para conectar la alimentación 24 VCC o 100-240 VAC (según el modelo).
11 Conector para el bus CANopen (bornero de tornillos).
12 Conector RJ45 para red Ethernet y LED de actividad.
13 Bloque de visualización LED que muestra:
- El estado del controlador y sus componentes (batería, tarjeta de memoria SD)
- El estado de los puertos de comunicación incorporados (bus CANopen, puertos serie, Ethernet)

14 Conector bus TM4: bus de comunicación para conectar los módulos de comunicación Modicon TM4.
15 Puerto serie SL1 (RS232 o RS485): conector RJ45
16 Puerto serie SL2 (RS485): borneros de tornillos
17 Conexión de entradas lógicas de 24 VCC: borneros de tornillos extraíbles.
18 Conector bus TM3 para conectar a los módulos de extensión Modicon TM3

Cableado de la alimentación del módulo a 24 VDC ó 230 VAC.

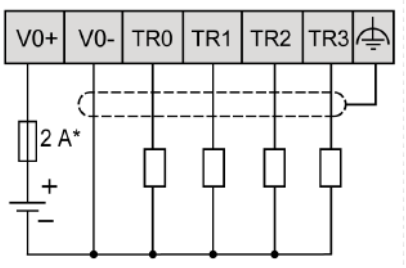


Cableado de Entradas rápidas.

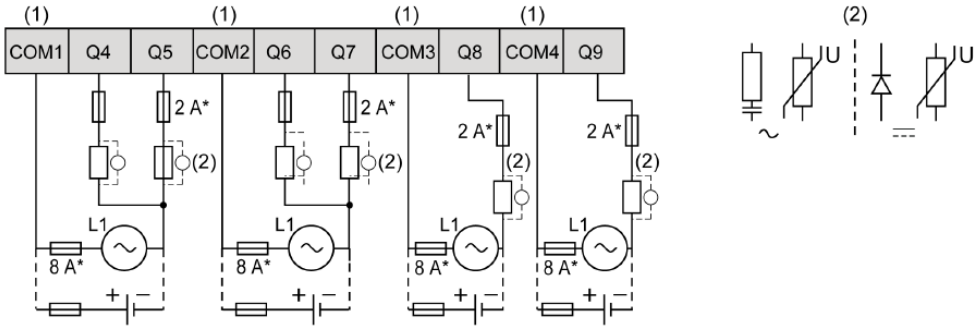


(1) Los bornes COM0, COM1, COM2 no están internamente conectados.

Cableado para Salidas Rápidas.



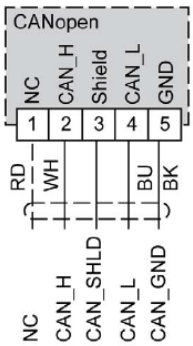
Cableado salidas relé.



(1) Los bornes COM0, COM1, COM2 no están internamente conectados.

(2) Para mejorar el tiempo de vida de los contactos , y para protegerlo de posibles daños de una carga inductiva , debe conectar un diodo en paralelo para cada carga inductiva en DC o un amortiguador RC en paralelo de cada carga inductiva CA

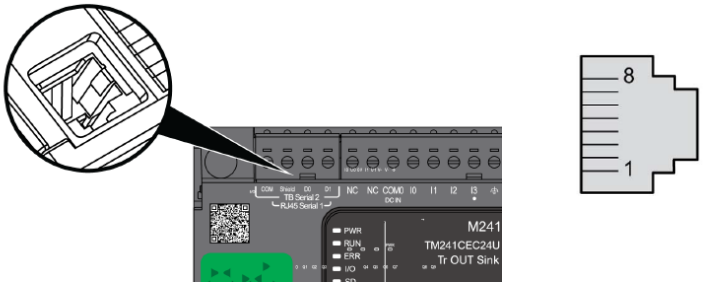
Conexionado del bus CANopen (bornero).



Pin	Signal	Description	Marking	Color of Cable
1	Not used	Reserved	NC	RD: red
2	CAN_H	CAN_L bus line	CAN_H	WH: white
3	CAN_SHLD	Optional CAN shield	Shield	-
4	CAN_L	CAN_L bus line	CAN_L	BU: blue
5	CAN_GND	CAN Ground	GND	BK: black

LT: Resistencia de fin de línea de 120 Ω.

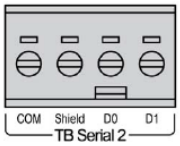
Conexionado del puerto serie 1(RJ45).



Pin	RS232	RS485
1	RxD	N.C.
2	TxD	N.C.
3	N.C.	N.C.
4	N.C.	D1
5	N.C.	D0
6	N.C.	N.C.
7	N.C.*	5 Vdc
8	Common	Common

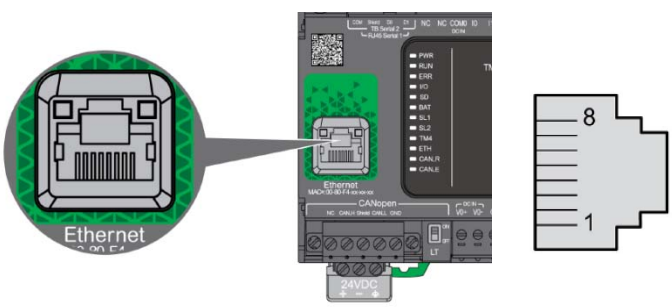
(*) 5 V de alimentación para el RS485, no conectar en RS232.

Conexionado del puerto serie 2 (bornero).



Pin	RS485
COM	0 V com.
Shield	Shield
D0	D0 (B-)
D1	D1 (A+)

Conexionado del puerto Ethernet (RJ45).



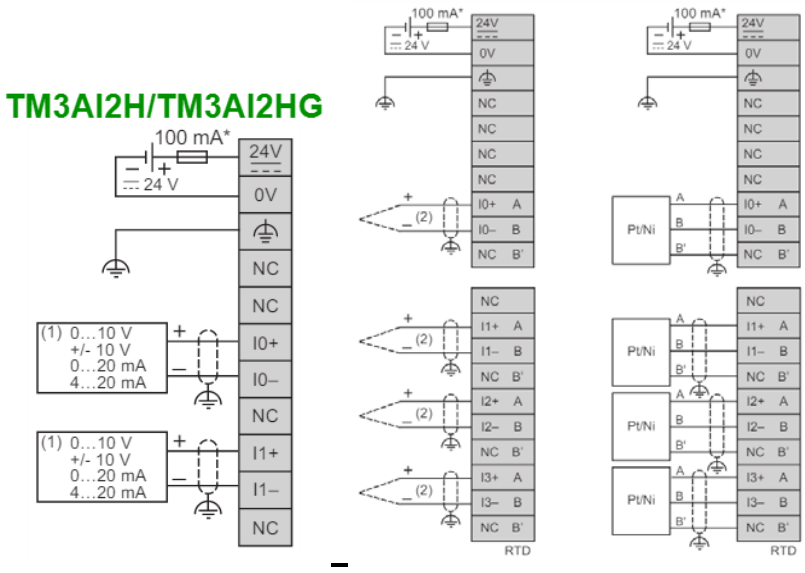
Pin N°	Signal
1	TD+
2	TD-
3	RD+
4	-
5	-
6	RD-
7	-
8	-

MÓDULOS DE AMPLIACIÓN.

Hay varios módulos de ampliación, digitales, analógicos y de comunicación.

Modulos de ampliación de entradas y salidas digitales:

Módulos de ampliación de entradas y salidas analógicas



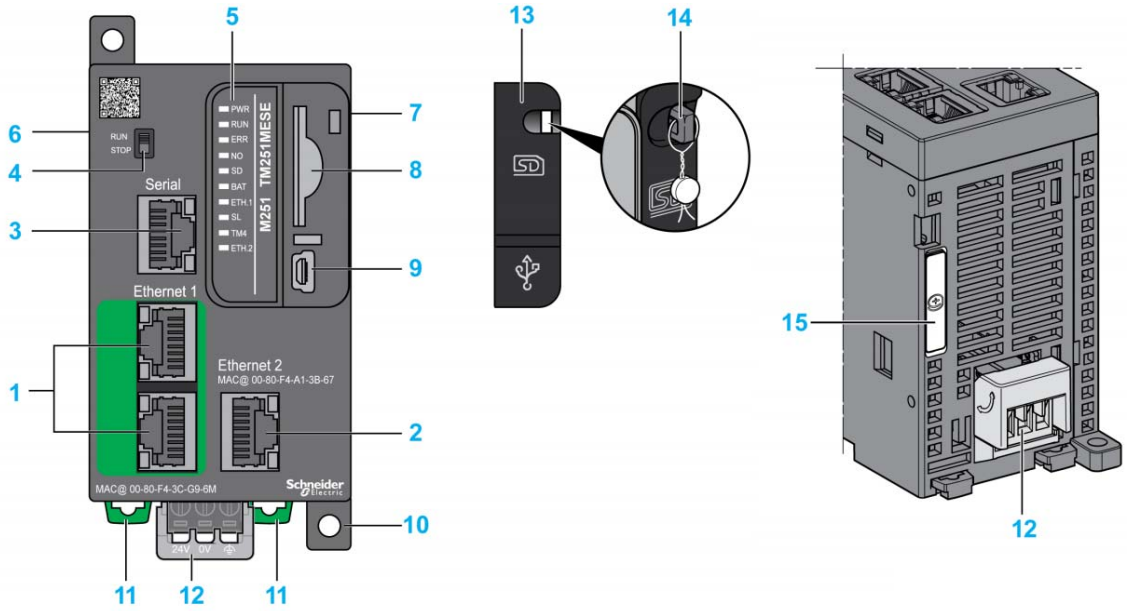
1.2 CONTROLADOR LÓGICO MODICON M251

Este módulo a diferencia del anterior, no tiene ni entradas ni salidas embebidas en el controlador, pero siempre tiene al menos un puerto de comunicación Ethernet. Dependiendo de la necesidad, también hay dos modelos diferentes uno con un puerto más de Ethernet y el otro con un puerto CANopen integrado.

El modelo de controlador, con el que vamos a realizar el desarrollo de este manual SoMachine es el **TM251ESE** el cual se alimenta con **24 VDC** y dispone de:

- Un puertos serie **RS 232/RS 485** (protocolos SoMachine-Network, Modbus, ASCII)
- Un puerto **Ethernet 1**, protocolo Modbus TCP (cliente/servidor) o servidor web.
- Un puerto **Ethernet 2**, protocolo Modbus TCP (I/O Scanning)

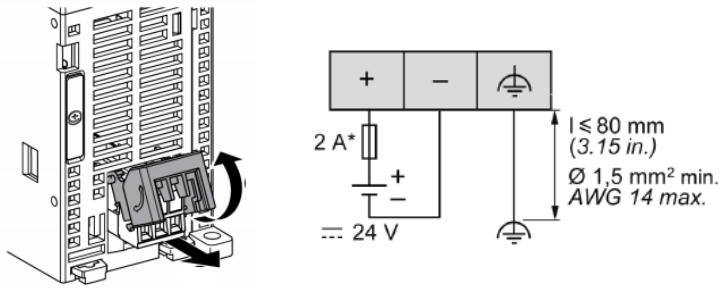
Descripción y cableado del Controlador



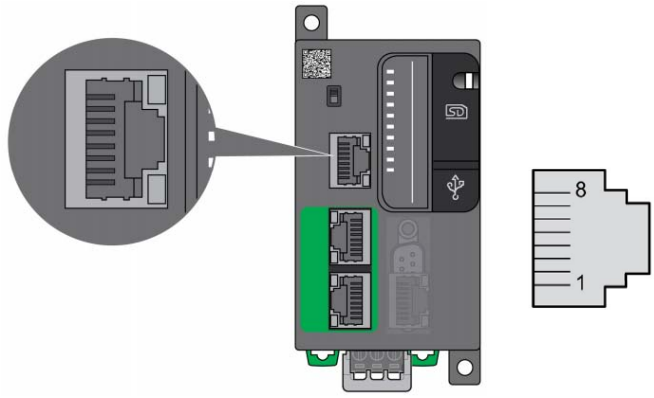
1 Dos conectores RJ45 en un switch para red Ethernet de maquinas o fabrica con LED de actividad
2 En controlador TM251MESE: conector RJ45 para red Ethernet 2 de bus de campocon LED de actividad
3 Un puerto serie SL (RS232 o RS485): conector RJ45
4 Interruptor de Run/Stop

5 Bloque de visualización LED que muestra: el estado del controlador, bateria, tarjeta de memoria SD, Ethernet
6 Conector bus TM4: bus de comunicación para conectar los módulos de comunicación Modicon TM4
7 Conector bus TM3 para conectar los módulos de extensión Modicon TM3
8 Ranura para la tarjeta de memoria SD
9 Conector USB mini-B para programación
10 Lengüetas para montaje con tornillos en un panel
11 Pestaña para bloqueo en carril 5 simétrico.
12 Bornero de tornillos extraíble, 3 bornas para conectar la alimentación de 24 VDC
13 Cubierta de protección (slot para tarjeta SD y puerto de programación USB mini-B)
14 Gancho de sujeción (gancho no incluido)
15 Ranura para la batería de reserva

Cableado de la alimentación del módulo a 24 VDC ó 230 VAC.



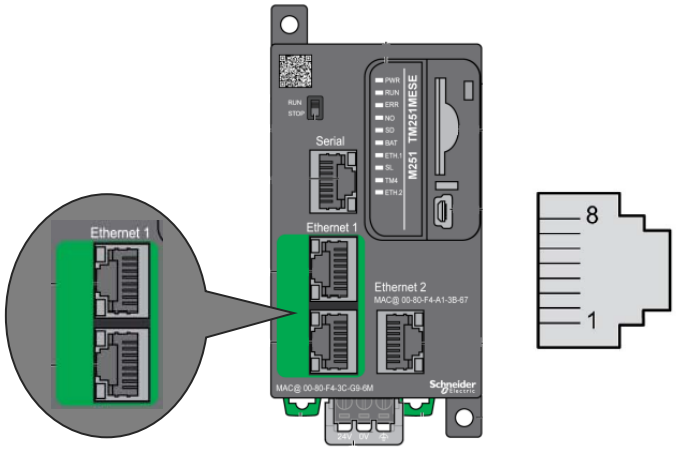
Conexionado del puerto serie 1(RJ45).



Pin	RS232	RS485
1	RxD	N.C.
2	TxD	N.C.
3	N.C.	N.C.
4	N.C.	D1
5	N.C.	D0
6	N.C.	N.C.
7	N.C.*	5 Vdc
8	Common	Common

(*) 5 V de alimentación para el RS485, no conectar en RS232.

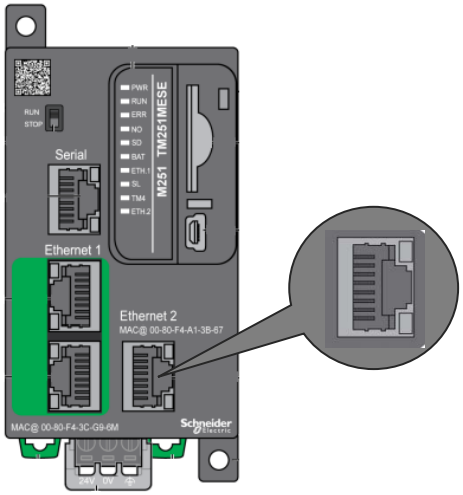
Conexionado del puerto Ethernet 1 (RJ45).

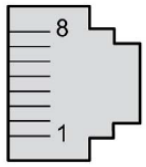


Pin N°	Signal
1	TD+
2	TD-
3	RD+
4	-
5	-
6	RD-
7	-
8	-

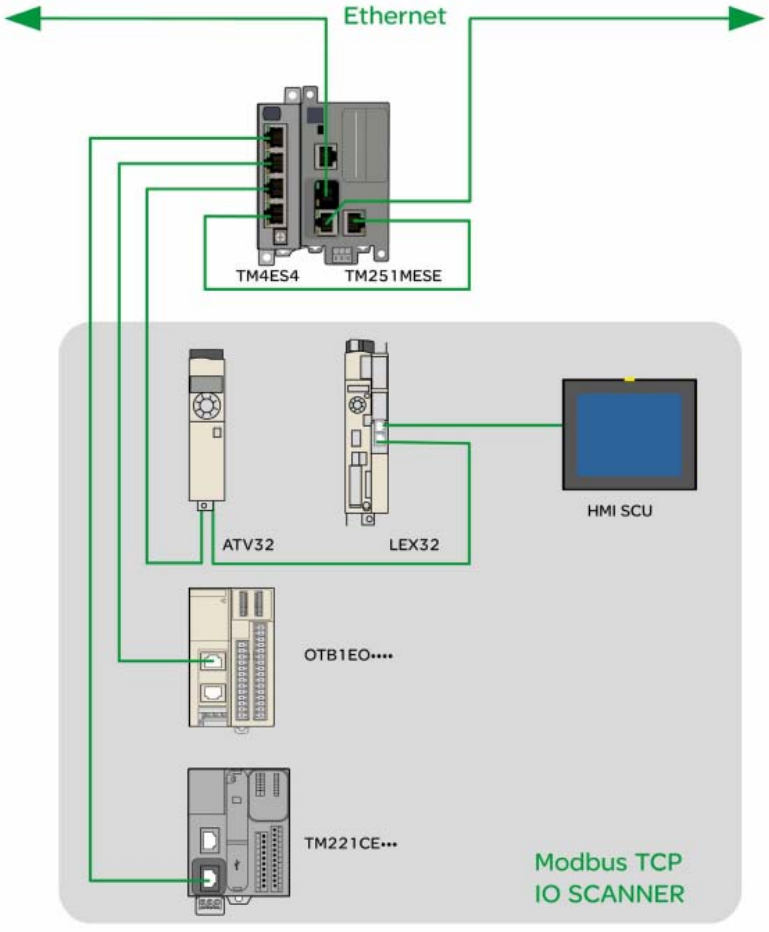
El puerto Ethernet 1 dispone de dos puertos RJ45 y la dirección MAC está justo debajo del contorno verde.

Conexionado del puerto Ethernet 2 (RJ45).

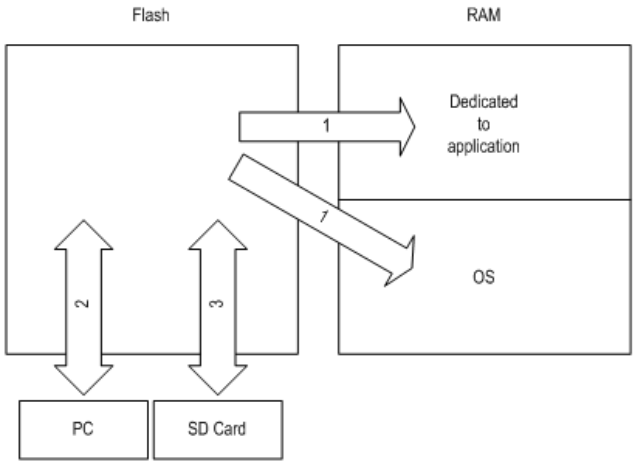




Pin N°	Signal
1	TD+
2	TD-
3	RD+
4	-
5	-
6	RD-
7	-
8	-



1.3.- GESTIÓN DE LA MEMORIA:



1. Proceso interno iniciado automáticamente al conectar la alimentación y reiniciar, los archivos se transfieren desde la memoria Flash a la RAM (Sobrescribiéndola).
2. Proceso que lo inicia el usuario, a través del cable de programación o Ethernet, donde se carga los archivos desde el Somachine, Webserver ó FTP.
3. Proceso iniciado automáticamente por un script (transferencia de datos) o por una (clonación) cuando se conecta una tarjeta SD.

La Memoria RAM (64Mb) está distribuida en dos áreas:

- **Área de aplicación:** Formada por el área de datos del sistema **192 Kb**, de los cuales hay **125 Kb**, para la me memoria interna **%MW** que irá desde el índice **0 al 63999**, 64 Kb para las variables retentivas y persistentes y 3 Kb de memoria reservada. Y **8 Mb** para el área de usuario que incluye los símbolos, variables, aplicación y librerías.
- **Área de sistema operativo** (OS).

Nota: Las variables retentivas son aquellas en las que permanece su valor, si se produce un error interno que pasa el controlador a STOP o se hace un 'reinicio en caliente', si hacemos un download o un reinicio en frio perderemos los datos de las variables RETAIN, si queremos mantener el valor en esas condiciones tendremos que declararar las variables como persistentes, en este caso la única manera de inicializar las variables PERSISTENT es hacer un Reset a origen.

La memoria Flash es de 128 Mb y guarda la siguiente información:

- **Aplicación de inicio:** Este archivo reside en la memoria Flash y contiene el código binario compilado de la aplicación ejecutable. Cada vez que el controlador se reinicia, la aplicación que se ejecuta se saca de la aplicación de inicio y se copia en la memoria RAM.
- **Archivo de origen:** Archivo con el programa sin compilar que se puede descargar desde la memoria flash a la PC, si el archivo de origen no se encuentra disponible en el PC.
- **Archivo de Postconfiguración:** Archivo que contiene la configuración de los puertos de Ethernet, línea serie, y los parámetros de firewall. Los parámetros especificados en el archivo tienen prioridad sobre los parámetros de la aplicación ejecutable en cada reinicio.
- **Datalogging:** Archivos en el que el controlador registra los eventos según lo especificado por la aplicación de usuario.
- **Webvisualizations:** Páginas HTML que se muestran por el servidor web para el sitio web incorporado en el controlador.
- **Firmware** del controlador que se puede escribir en la memoria Flash. El archivo de firmware se aplica en el próximo reinicio del controlador.
- **Variables declaradas como remanentes,** tanto persistentes como retentivas.

1.4.- ARQUITECTURA DE LA MAQUETA DE PRÁCTICAS.

Para facilitar lo conexión y el buen uso del material de prácticas, a la hora de realizar las prácticas, la asignación de direcciones IP's y el nombre de red y el password de la wifi siguen el estándar que se muestra en la figura.

Allí donde pone **'x'** viene dado en función del número de maleta. Ejemplo si tenemos la maqueta de práctica que tiene el nombre de red **'democase_8'**, significa que las direcciones IP's de esa maqueta concreta será **192.168.80.254** para el **Acces point**, **192.168.80.20** para el controlador **M251**, **192.168.80.10** para el controlador **M241** y **192.168.80.30** para el terminal táctil **HMISCUA5**.

CAPÍTULO

2

2. SOFTWARE SOMACHINE

2.1 - INTERFACE DE USUARIO

La navegación dentro del SoMachine es intuitiva y muy visual. El interface de usuario está optimizado para que en los diferentes pasos del proyecto, se habiliten las herramientas necesarias para ese paso. El interface de usuario habilitará las opciones que se puedan realizar en cada paso. El espacio de trabajo ha sido optimizado, para sólo mostrar lo necesario en cada momento.

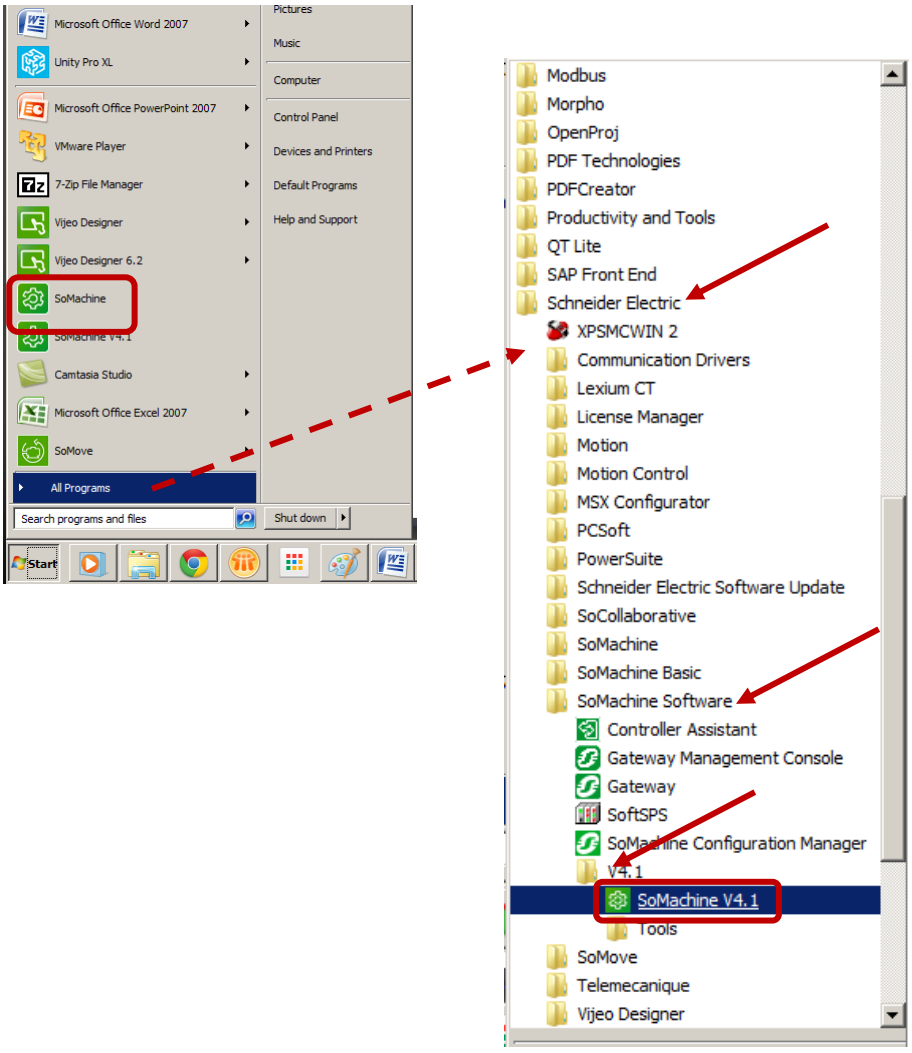
2.2 - ABRIR SOMACHINE

Seleccionar el icono SoMachine en el menú de inicio de Windows:

Inicio » Programas »Schneider Electric » SoMachine> » SoMachine

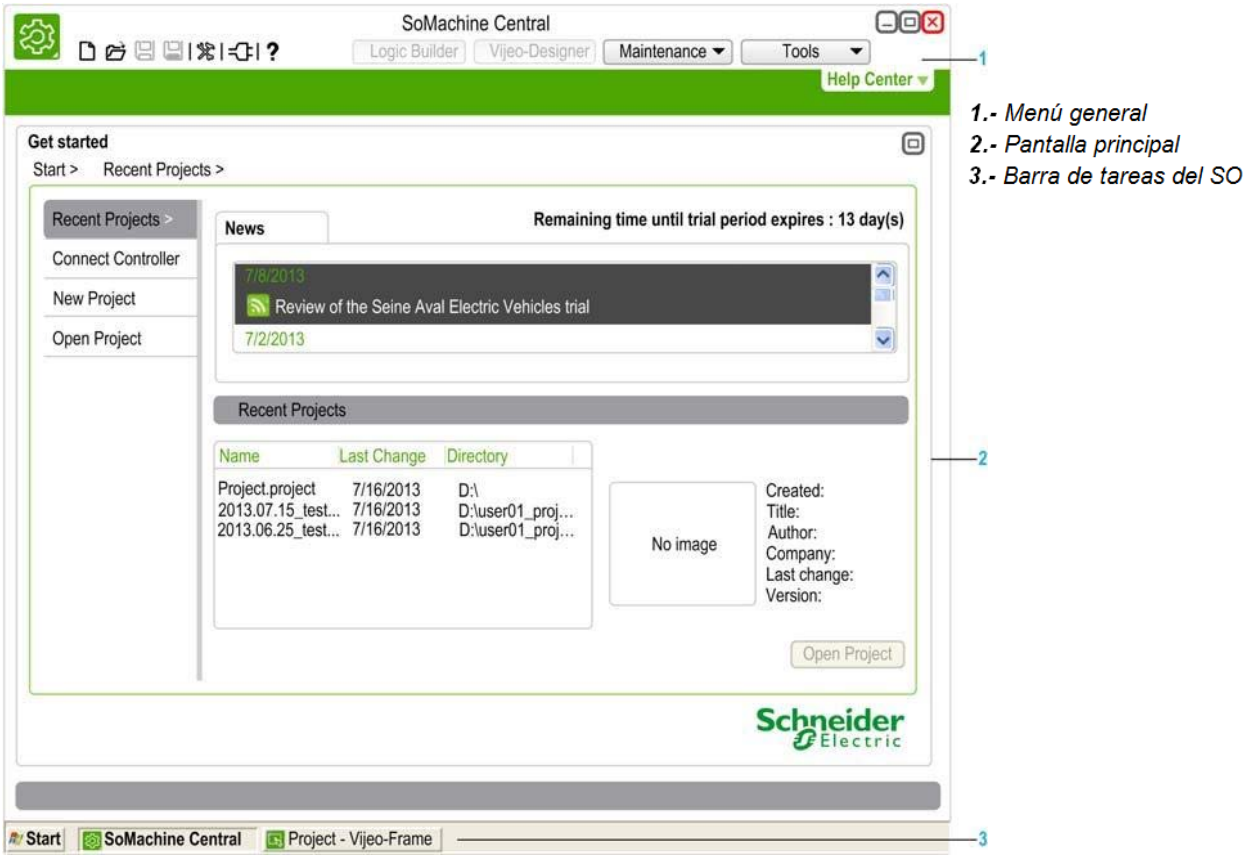
Ó

Hacer doble clic en el icono de SoMachine en el escritorio.



2.3- VENTANA DE INICIO

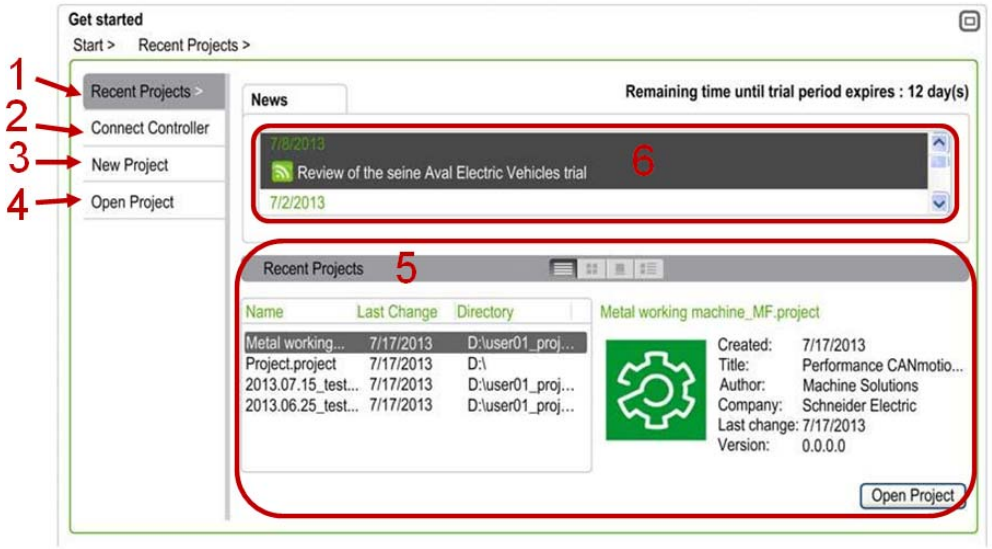
La ventana de inicio, es la ventana que aparece una vez iniciado el SoMachine y en ella podremos seleccionar las siguiente opciones para abrir un proyecto existente o crea uno nuevo proyecto, por ejemplo.



Opciones en la pantalla de inicio

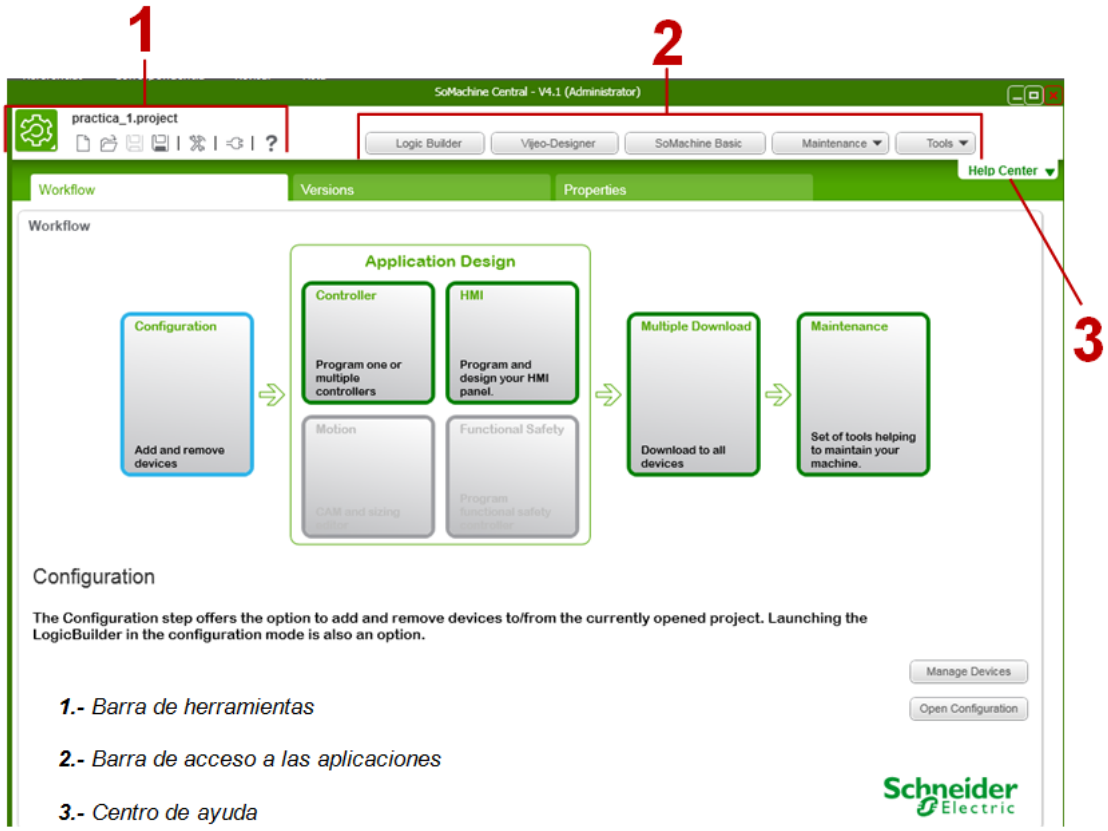
Cuando se inicia SoMachine, se muestra la pantalla de inicio de SoMachine Central. Esta pantalla ofrece las siguientes funciones de proyecto:

1. Abrir Proyectos recientes
2. Conectar con el controlador
3. Nuevo proyecto
4. Abrir proyecto
5. Lista de proyectos abiertos recientemente
6. Además, proporciona las últimas noticias de Schneider Electric.



2.4 – SOMACHINE CENTRAL

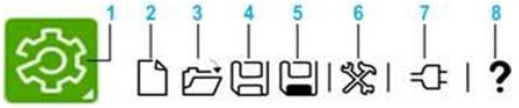
Una vez creado el creado ó abierto cualquier tipo de proyecto el SoMachine nos muestra la ventana de SoMachine Central, que proporciona interface general con los elementos que figuran a continuación.



Barra de herramientas

La barra de herramientas es parte del menú general de la ventana del SoMachine Central.

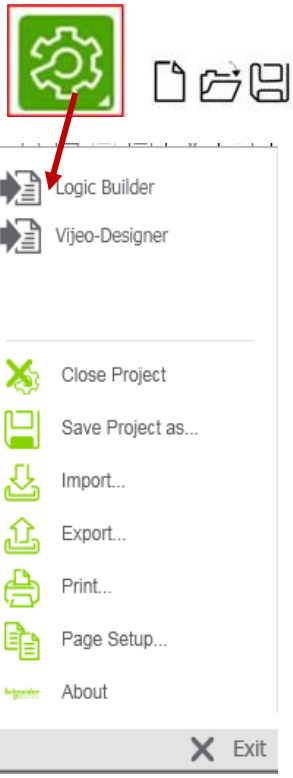
Cada icono de la barra de herramientas muestra la información sobre la acción que realiza dicha herramienta cuando el puntero del ratón se mueve sobre el icono correspondiente sin hacer clic.



Icono	Descripción
1	Abra el menú principal
2	Crear un nuevo proyecto.
3	Abra un proyecto existente.
4	Guarde el proyecto, que actualmente tiene el foco.
5	Guarde el proyecto, que actualmente tiene el foco, con un nuevo nombre.
6	Abra el cuadro de diálogo de opciones del sistema
7	Conectar con el controlador
8	Abra el Centro de ayuda

Icono del menú general de funciones

El menú general ofrece funciones que se necesitan frecuentemente, para abrir el menú general, haga clic en el icono de la barra de herramientas SoMachine central.

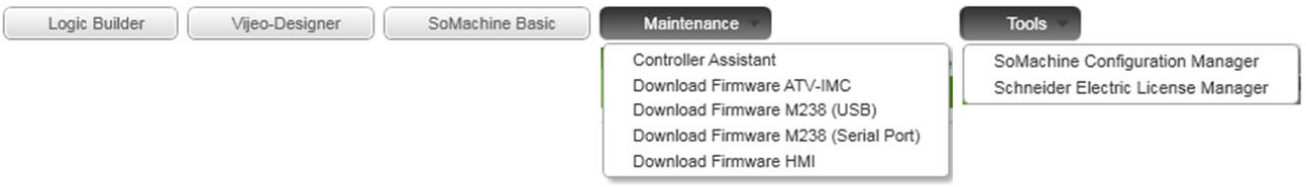


1. Abrir el Logic Builder.
2. Abrir el Vijeo-Designer.
3. Cerrar proyecto
4. Guardar proyecto como ...
    a. *Guardar proyecto como..*
    b. *Guardar proyecto como Biblioteca compilada*
    c. *Guardar proyecto como plantilla ...*
    d. *Guarda la librería y la coloca en el repositorio de bibliotecas*
    e. *Guardar archivo ...*
5. Importar ...
    a. *Importación Vijeo Designer Proyecto ...*
    b. *Importar proyecto SoMachine básico ...*
6. Exportar ...
    a. *Exportación de proyecto Vijeo Designer ...*
    b. *Exportación de proyecto de SoMachine Basic ...*
7. Imprimir ... *(Define el contenido de la documentación)*
8. Página de configuración
9. Acerca de
10. Salida

Barra de acceso a herramientas

La barra de herramientas de acceso se muestra en la parte superior de SoMachine Central. Le permite cambiar a otras herramientas integradas para SoMachine.

Cada icono de la barra de herramientas de acceso muestra una punta de la herramienta cuando el puntero del ratón se mueve sobre el icono correspondiente sin hacer clic.

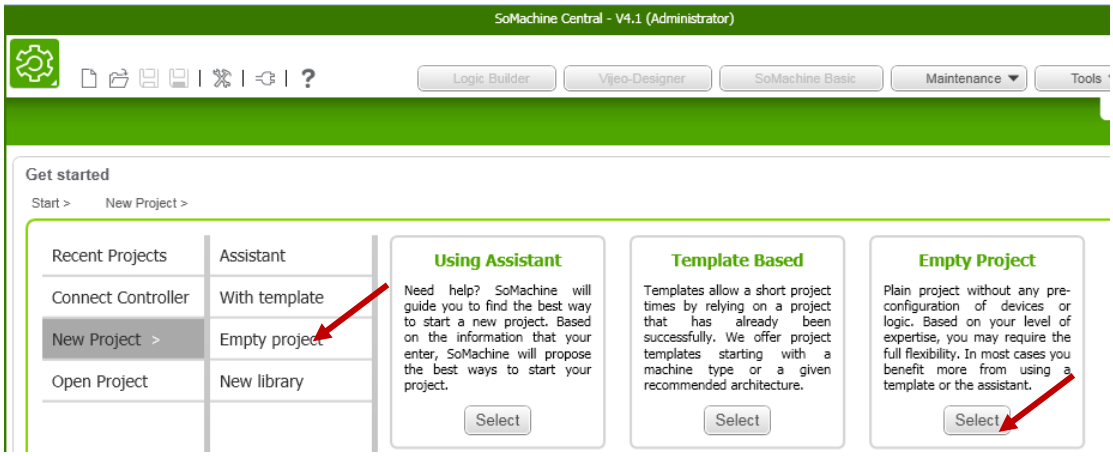


Botón	Descripción
Logic Builder	Abrir el interface de programación del SoMachine
Vijeo-Designer	Abrir el Vijeo Designer.
SoMachine Basic	Abrir el SoMachine Basic.
Mantenimiento	Seleccione una herramienta de mantenimiento (por ejemplo: Controller Assistant ó Configuración del OPC
Herramientas	Abrir la herramienta seleccionada.

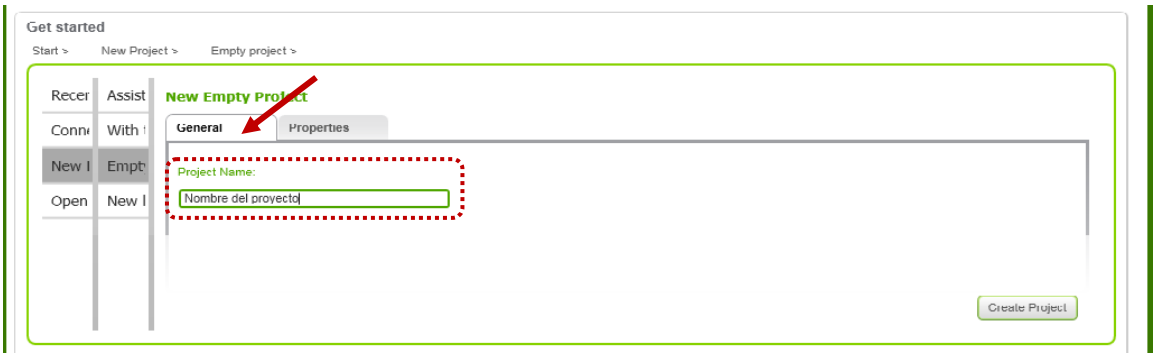
2.5 – CREAR UN PROYECTO VACIO

Cuando se crea un **'Nuevo Proyecto vacío'** se crea un proyecto simple, sin pre configuración de dispositivos, ni lógica. Sobre la base de su nivel de experiencia, esta opción le ofrece total flexibilidad.

En la ventana de inicio, hacer clic en 'N**ew Project > Empty project' ó pulsar el botón de 'Select' que hay en el marco 'Empty Project'.**



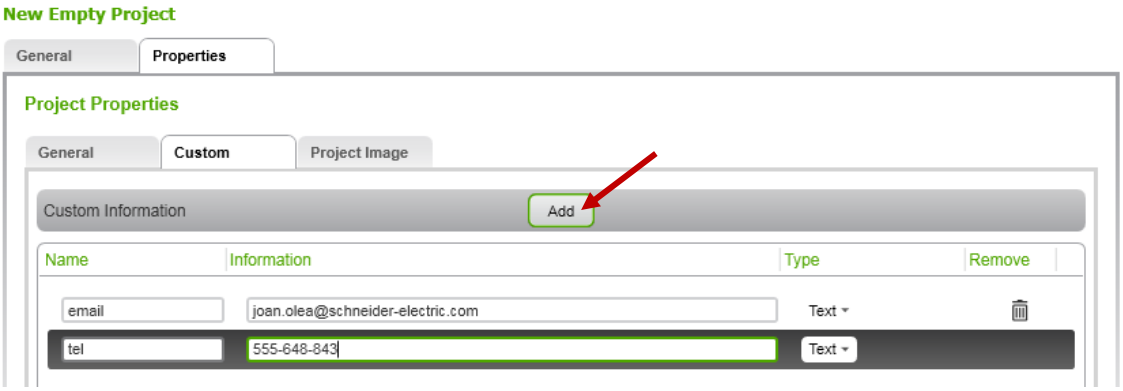
En el área de trabajo, aparecen dos pestañas, en la pestaña **'General'** escribiremos el Nombre de nuestro proyecto en el campo **'Project Name'.**



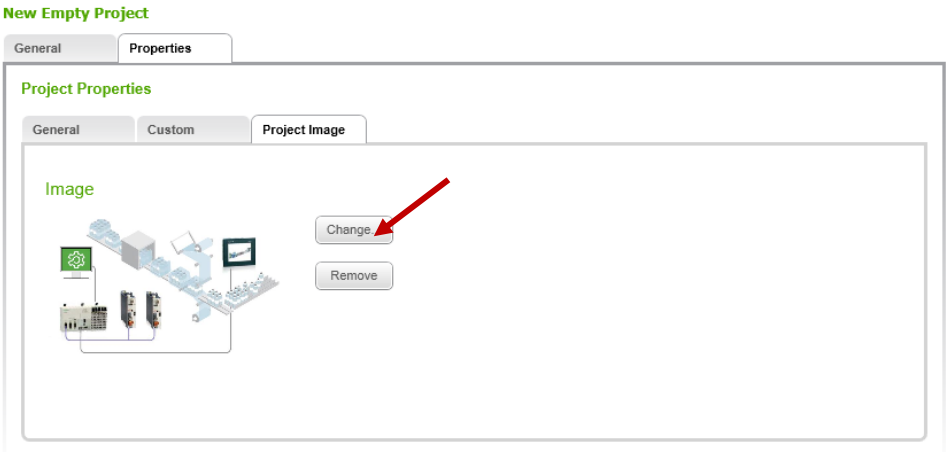
En la pestaña **'Properties'** especificaremos información relacionada con el **proyecto.** Cuando abrimos esta pestaña nos aparecen tres subpestañas que donde que agrupa diferente tipo de información.

En la subpestaña **'General'** añadiremos los datos generales del proyecto.

En la subpestaña **'Custom'** añadiremos otros datos de interés para nuestro proyecto (Ej. Persona de contacto, teléfonos), apretando el botón **'Add'** se añadirá un campo, seleccionado el tipo de campo en la columna '**Type'**.



En la subpestaña **'Project Image'** podemos asociar una imagen a nuestro proyecto para ayudarnos a identificarlo mejor.

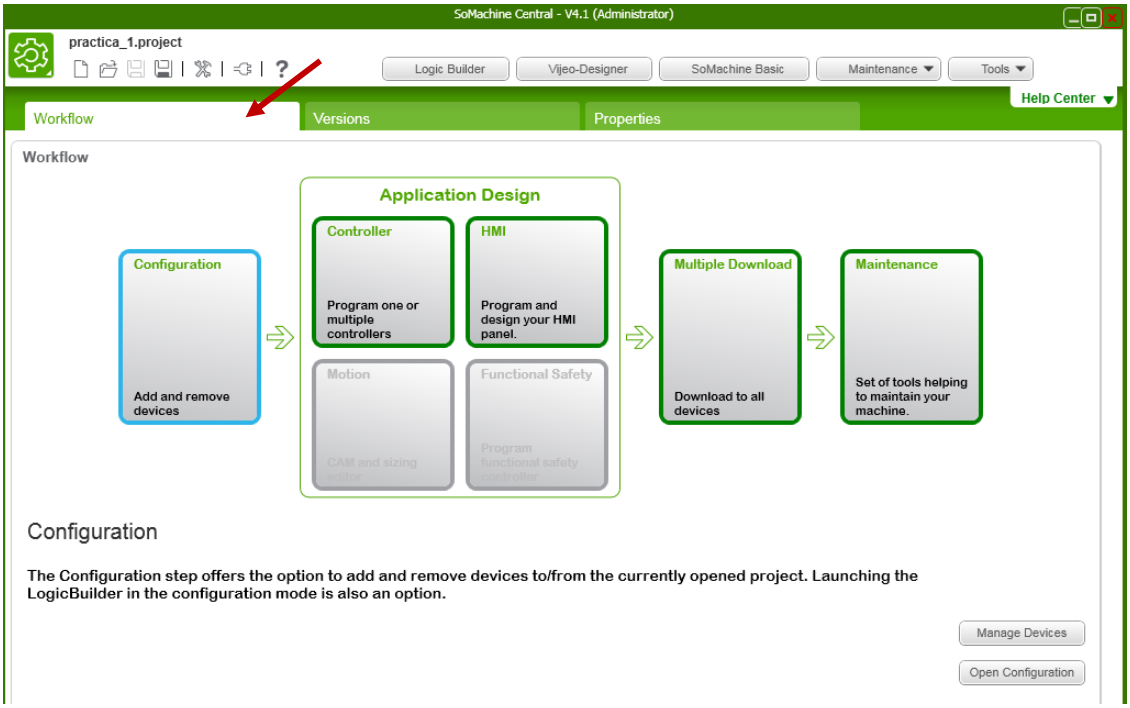


Una vez insertada toda la información, finalizaremos pulsando el botón **'Create Project'** que se encuentra en la parte inferior derecha.

Nota: Las información del proyecto es en todo momento accesible y modificable desde la pestaña 'Propiedades' de la pantalla principal del SoMachine Central.

2.6. – FLUJO DE DESARROLLO DEL PROYECTO

Después de generar un nuevo proyecto o abrir un proyecto existente, se muestra la pantalla de flujo de trabajo. La pantalla muestra una representación gráfica de la gestión del flujo de trabajo del proyecto.

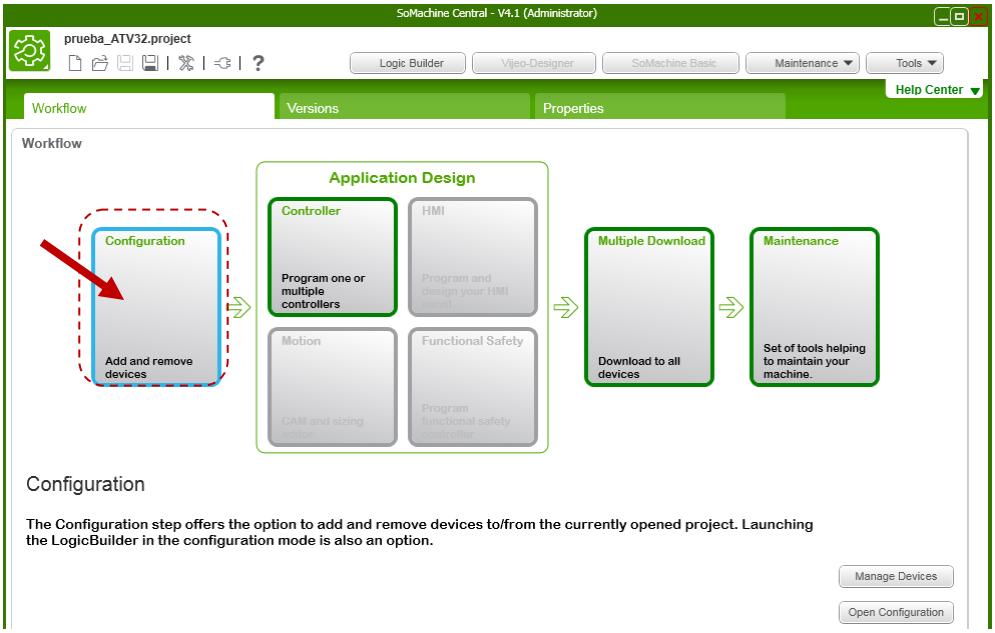


Para interactuar con el flujo de trabajo, haga clic en los diferentes pasos del flujo de trabajo, que están representados como botones. Si selecciona un paso del flujo de trabajo, la información detallada de este paso se muestra en la parte inferior de la pantalla.

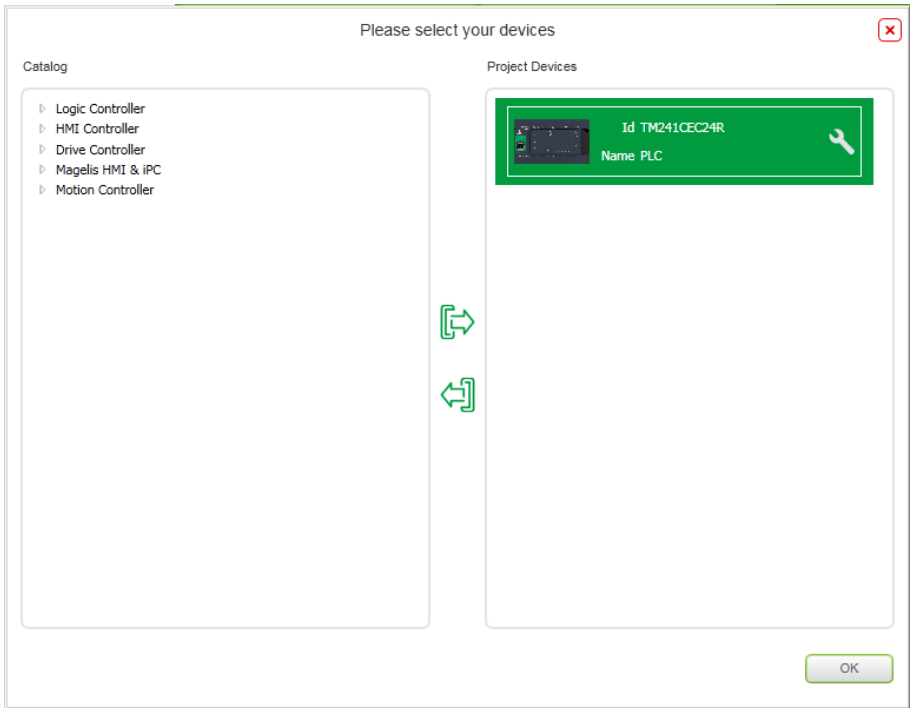
Elemento	Descripción	
Configuración	En esta etapa, se puede añadir, eliminar y configurar los dispositivos y la comunicación.	
Diseño de aplicaciones	**Controlador**	En este paso, se puede programar el controlador (o varios controladores) de su proyecto. Se abrirá la herramienta de programación 'Logic Builder'.
	HMI	En este paso, se puede programar y diseñar la aplicación HMI. Haga clic en el botón, para iniciar el Vijeo-Designer.
	Movimiento	**No disponible con esta versión de SoMachine.*
	Seguridad	**No disponible con esta versión de SoMachine.*
Descarga Múltiple	En este paso, usted puede descargar el proyecto a los dispositivos.	
Mantenimiento	En este paso, se pueden realizar ciertas acciones de mantenimiento de los equipos de nuestro proyecto. Haga clic en este botón, para abrir la respectiva herramienta de mantenimiento (por ejemplo Controller Assistant, la configuración OPC).	
Area de información	Muestra información detallada del paso seleccionado en el flujo de trabajo.	

2.7.- VENTANA DE CONFIGURACIÓN

Para abrir la ventana de configuración hardware, desde la ventana general del SoMachine Central, seleccionaremos la pestaña de **'Workflow'** (flujo de trabajo). Dentro de los bloques del flujo de trabajo de un proyecto seleccionamos el primer bloque **'Configuration'**.



En la ventana de configuración de hardware aparece dos áreas, el área de la izquierda aparece el catálogo de dispositivos que se puede añadir al proyecto, en el área de la derecha aparecen todos los dispositivos que ya han sido añadidos al proyecto.

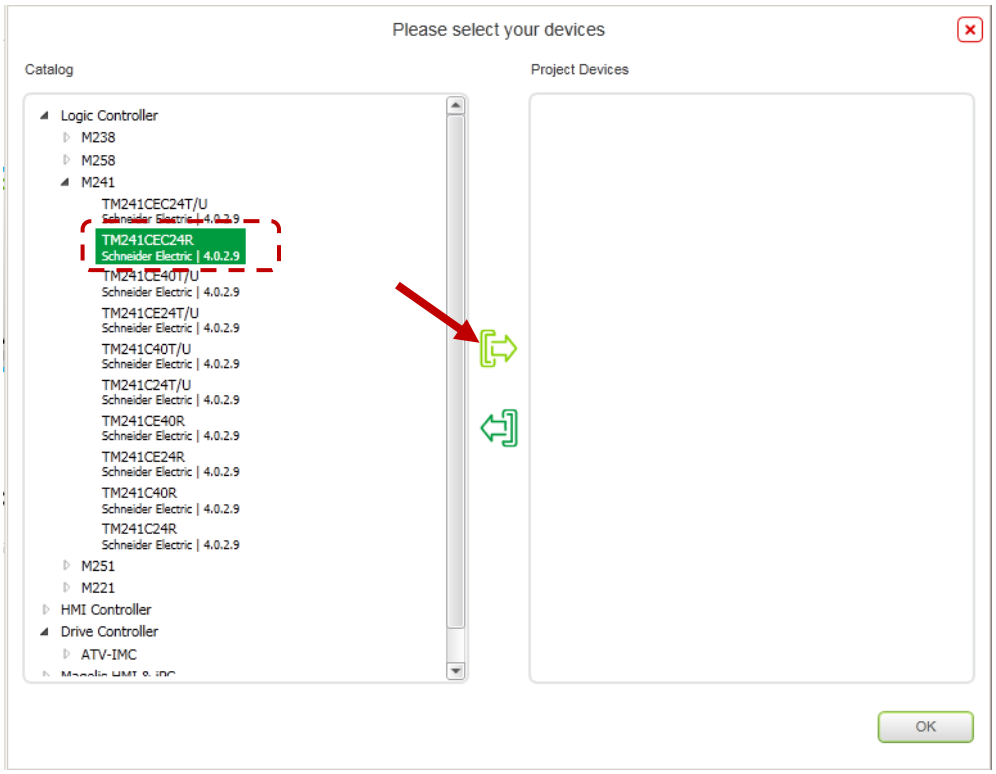


Dentro del área de catalogo, los diferentes dispositivos se agrupan en las siguientes familias:

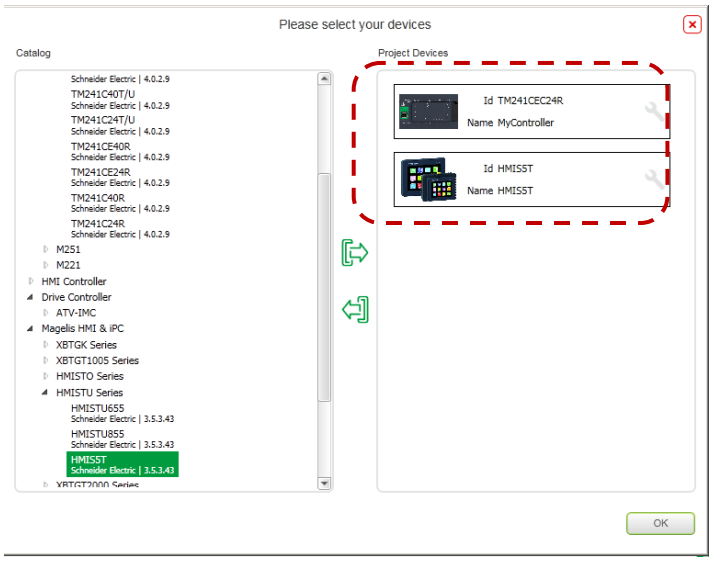
- *Logic Controller:* Controladores lógicos estándar (**M238, M258, M241, M251, M221**)
- *HMI Controller:* Esta familia engloba a las pantallas HMI que disponen de control para ser programado también (**XBTGC, HMISCU, XBTGK y todos los modelos de XBTGT con control**)
- *Driver Controller:* Controlador que se incorpora a un variador de frecuencia como elemento principal de la arquitectura de control, por ejemplo para las aplicaciones de Hoisting (**ATV-IMC**)
- *Magelis HMI & iPC:* Dispositivos HMI e IPC, todas aquellas pantallas HMI de la familia Magelis sin control.
- *Motion Controller:* Controladores que disponen de Canopen sincronizado para poder interpolar ejes (**LMC058**)

2.7.1 Añadir un equipo al proyecto

En el área de la izquierda '**Catalog**', seleccionar el equipo que se desea añadir al proyecto, una vez seleccionado aparecen en el centro los iconos de una flechas para mover los equipos de un área a otra. Pulsar la flecha que indica hacia la derecha y el equipo seleccionado aparecerá en el área de la derecha y ya estará incluido en el proyecto.

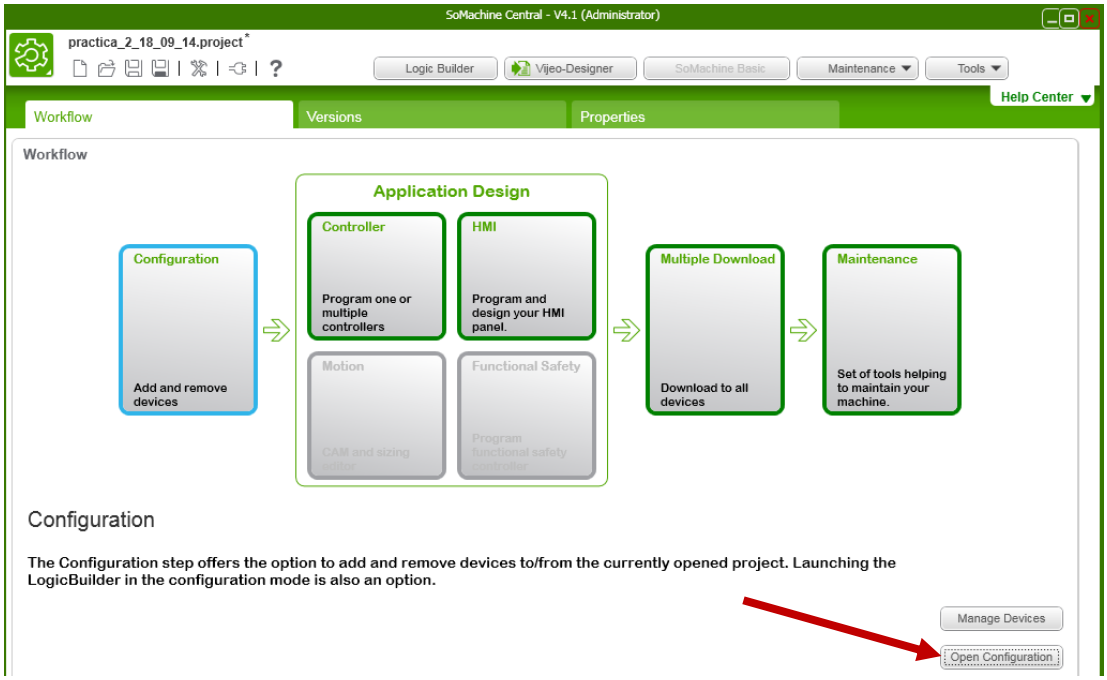


Cuando el equipo se haya añadido, aparecerá su representación gráfica en el área de la derecha '**Project Devices'**.

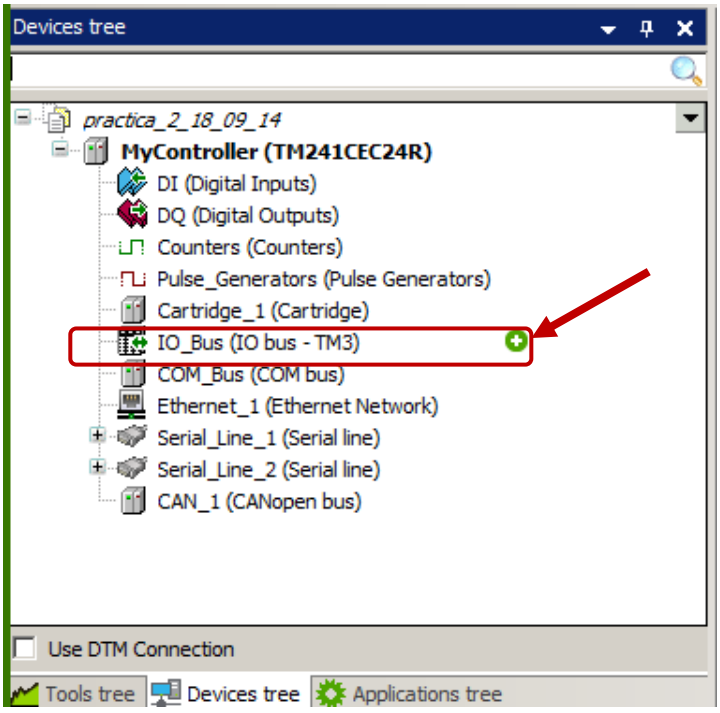


2.7.2 Añadir un módulo de expansión a un equipo

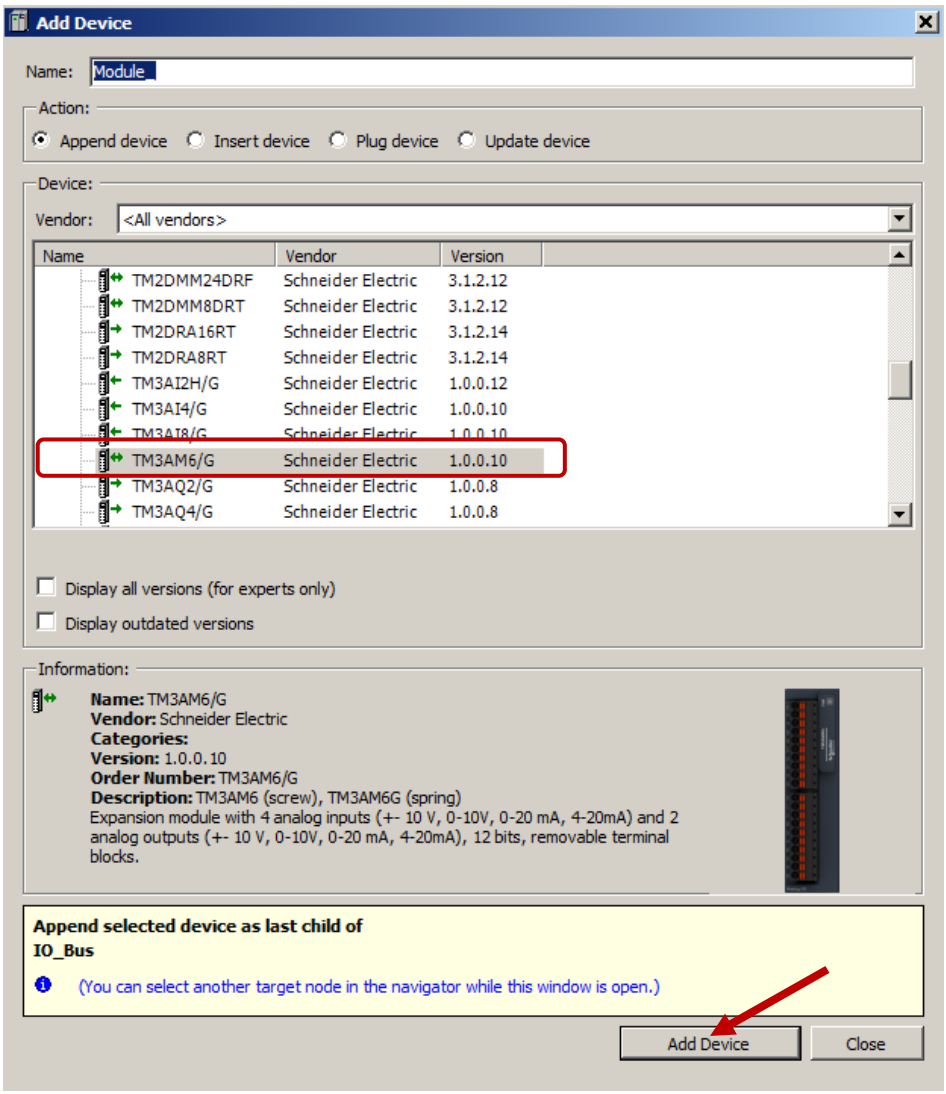
Para añadir un modulo de expansión a un equipo, en la ventana general SoMachine Central, hay que pulsar el botón **'Open Configuration'**.



Se abre la ventana de programación 'Logic Builder' y aparece en el **'Navegador'** ya aparece seleccionada la pestaña de 'Device tree' expandimos el controlador y al seleccionar '**IO_BUS (IO bus – TM3)**' aparece en lado de la derecha un símbolo de **'+'** para añadir módulos de expansión.



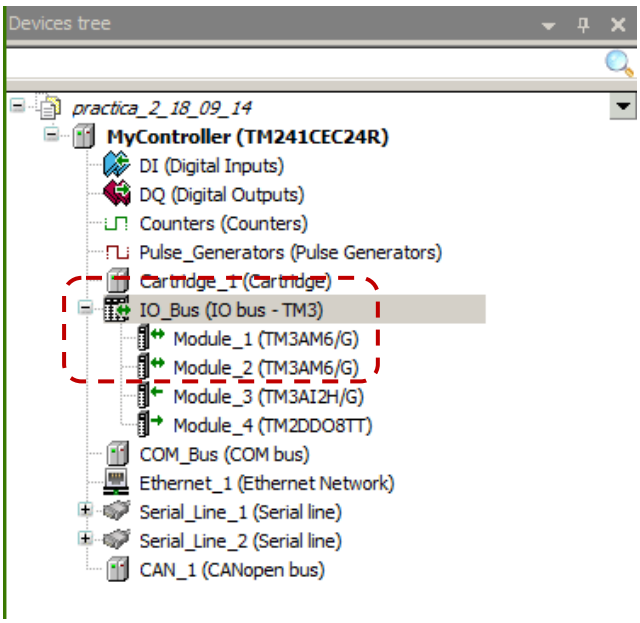
Aparecerá la ventana flotante de **'Add Device'**, en ella aparecen todos los módulos de expansión que se pueden añadir a ese equipo (**tanto TM2, TM3 y TM4**). Para agregarlo bastará con seleccionarlo en la lista y pulsar el botón de la parte de abajo **'Add Device''**. Cuando se selecciona un módulo en el campo **'Infomation'** aparece una descripción del módulo seleccionado (*en este caso un módulo analógico TM3AM6/G que dispone de 4 entradas y 2 salidas analógicas V/I*).



Cuando hayamos incluido los módulos de expansión deseados pulsaremos '**Close'** para cerrar la ventana.

Nota: Hay tener en cuenta la limitaciones hardware de cada uno de los controladores a la hora de añadir módulos de expansión. Por ejemplo, el M241 tiene una limitación de 7 módulos de ampliación, siempre que no se ponga un módulo de expansión de bus, también hay que tener en cuenta, que en este controlador, los módulos TM2 no se pueden intercalar con módulos TM3, sino hay que ponerlos al final del bus del PLC.

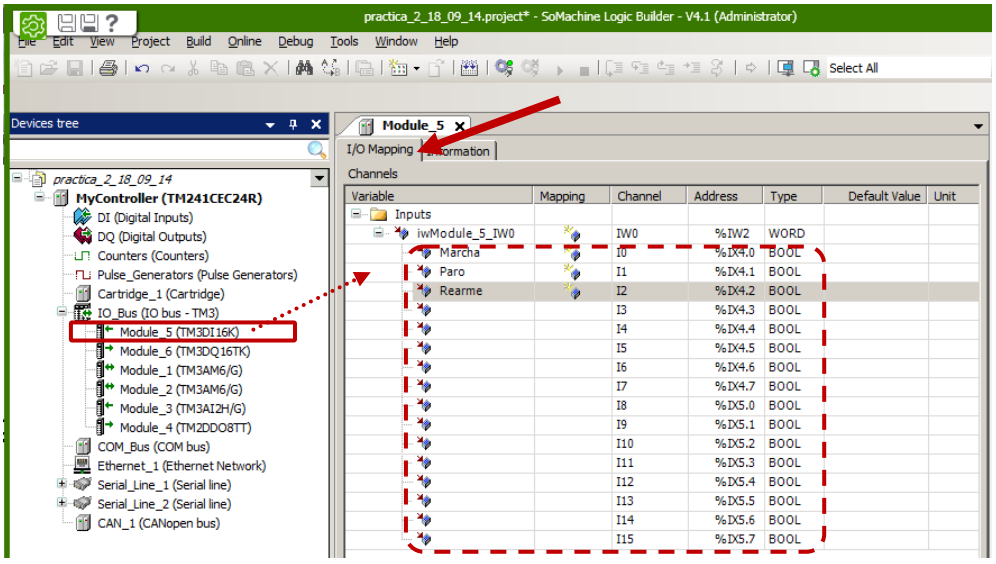
Debajo del **'IO_Bus (IO bus – TM3)**' en el '**Navegador**', aparecerán representados todos los módulos de expansión añadidos.



2.7.3 Configurar módulos de expansión de I/O discretas

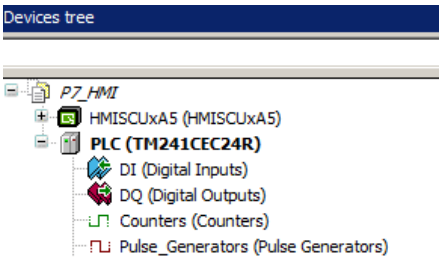
Si dentro del '**Navegador**' hacemos doble clic en uno de los módulos de ampliación añadidos, en el área de trabajo aparecen la ventana de configuración del módulo.

En la pestaña **'I/O Mapping'** podemos asignarle un símbolo a cada una de las entradas ó salidas físicas del módulo, escribiéndolo en el campo '**Variable**' de cada una de ellas.



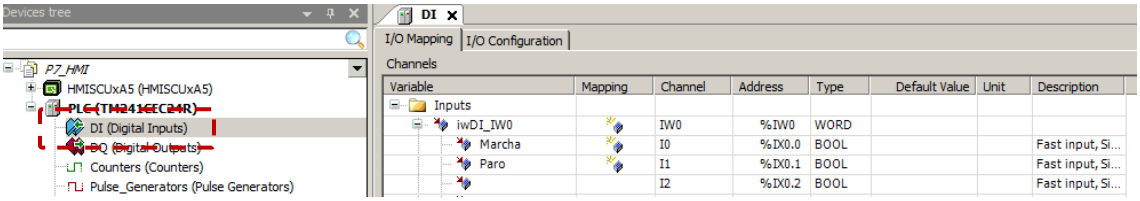
2.7.4 Configurar las funciones incrustadas

En esta opción se puede realizar la configuración de las entradas y salidas que hay embebidas en el controlador. En función del tipo y modelo de controlador el número de entradas y salidas variará. *Ejemplo: el M241 tiene 6 entradas digitales, 8 entradas rápidas, 6 salidas digitales y 4 salidas rápidas.*

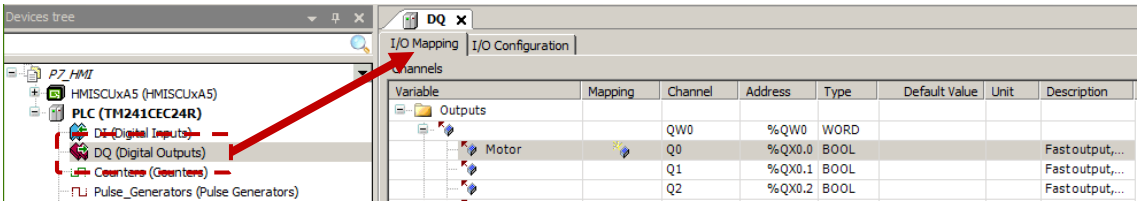


- **Configuración de las entradas/salidas incrustadas**

Entrando en **'DI'** se pueden configurar las entradas digitales incrustadas en la pestaña **'Configuración I/O'**, así como poner los símbolos en '**Asignación I/O'**.

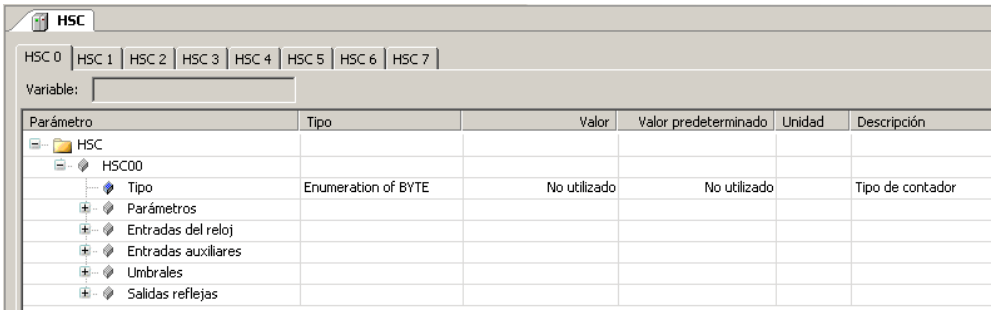


Entrando en **'DO'** se pueden configurar las salidas digitales incrustadas en la pestaña **'Configuración I/O'**, así como poner los símbolos en **'Asignación I/O'**.



- **Configuración de las entradas de contaje rápido (HSC)**

Configuración de las entradas de conteo rápido. La función HSC puede contar pulsos provenientes de sensores, encoders, etc. Las HSC son independientes del tiempo de scan del controlador.

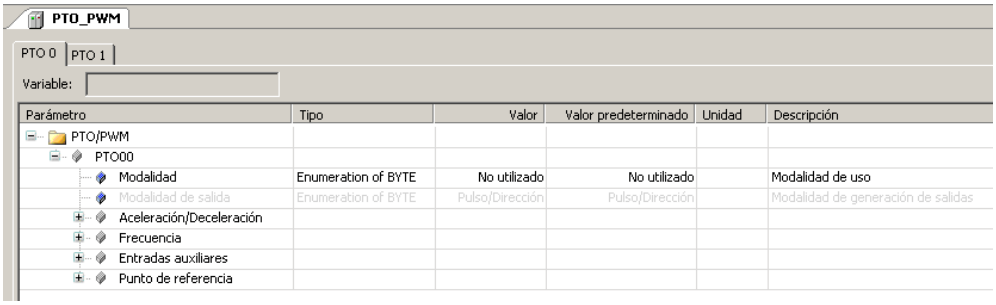


- **Configuración de las salidas rápidas (PTO_PWM)**

La función **PTO** nos genera una señal de pulsos con un número específico de pulsos y un periodo de tiempo determinado. Las funciones PTO se pueden configurar en dos modos diferentes.

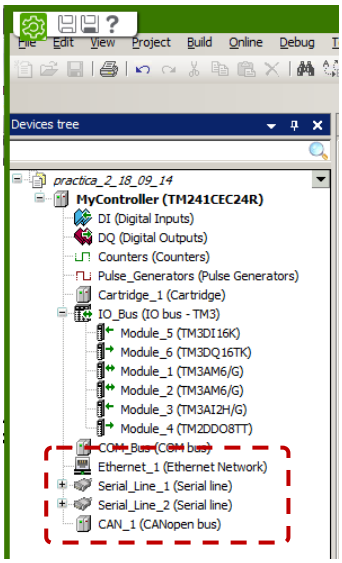
- Tren de pulsos.
- Pulso de ancho modulado PWM.

Hay dos canales PTO disponibles, cada canal PTO están asociadas a 2 salidas rápidas y una entrada estándar.



2.7.5 Configurar comunicaciones

Dependiendo del tipo y modelo de controlador, se disponen de más o menos puertos y protocolos de comunicación, Para el modelo M241 **TM238LFDC24DT**, por ejemplo, sería:

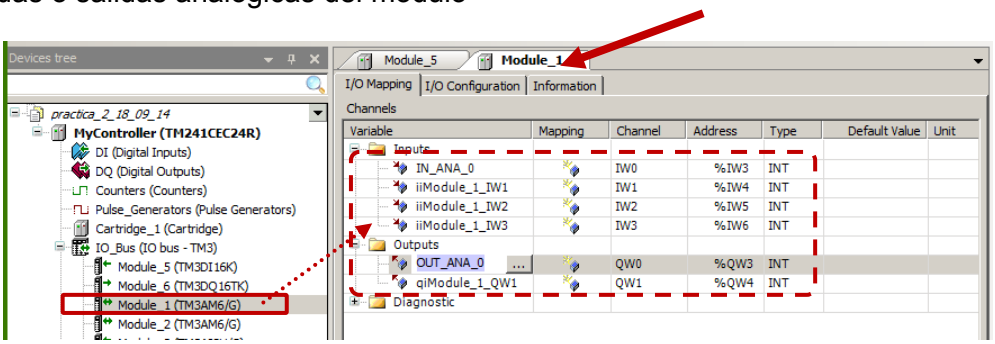


- **Ethernet_1:** Puerto Ethernet (diferentes servicios de Ethernet y comunicación Modbus TCP/ IP cliente-servidor).
- **Puerto serie 1**: Modbus manager **(configuración por defecto)**.
- **Puerto serie 2:** SoMachine - Network manager **(configuración por defecto)**.
- **CAN_1:** Para conectar remotamente con otros equipos CANopen (63 equipos máximo).

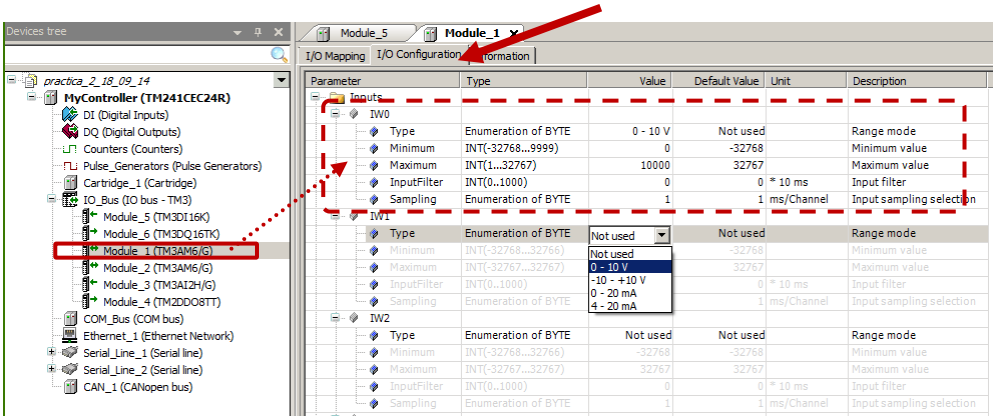
2.7.6 Configuración de los módulos de I/O analógicas

Para la configuración de los módulos de ampliación de entradas ó salidas analógicas, dentro del '**Navegador**' hacemos doble clic en uno de los módulos analógicos añadidos, en el área de trabajo aparece la ventana de configuración del módulo.

En la pestaña **'I/O Mapping'** podemos asignarle un símbolo a cada una de las entradas ó salidas analógicas del módulo



En la pestaña **'I/O Configuration'** activaremos aquellas entradas/ salidas analógicas que vamos a utilizar y la configuraremos.



Para configurar una entrada anlógica, primero tenemos que determinar que tipo de señal análogica nos llega (eso dependerá del tipo de sensor), en el campo **'Type'** en la columna **'Value'**, seleccionaremos entre (*Not Used*, *0 - 10V, -10 - +10 V, 0 - 20 mA, 4 - 20mA*). Cuando seleccionamos un tipo de señal se habilita el resto de campos de la configuración.

- **Minimum:** Valor mínimo de la señal (*quiere decir que si hemos configurado una señal 4 - 20 mA cuando a la señal físca tenga los 4 mA que valor numérico queremos ver en el programa, por defecto 0*).

- **Maximun:** Valor máximo de la señal (*quiere decir que si hemos configurado una señal 4 - 20 mA cuando a la señal físca tenga los 20 mA que valor numérico queremos ver en el programa, por defecto 10000*).

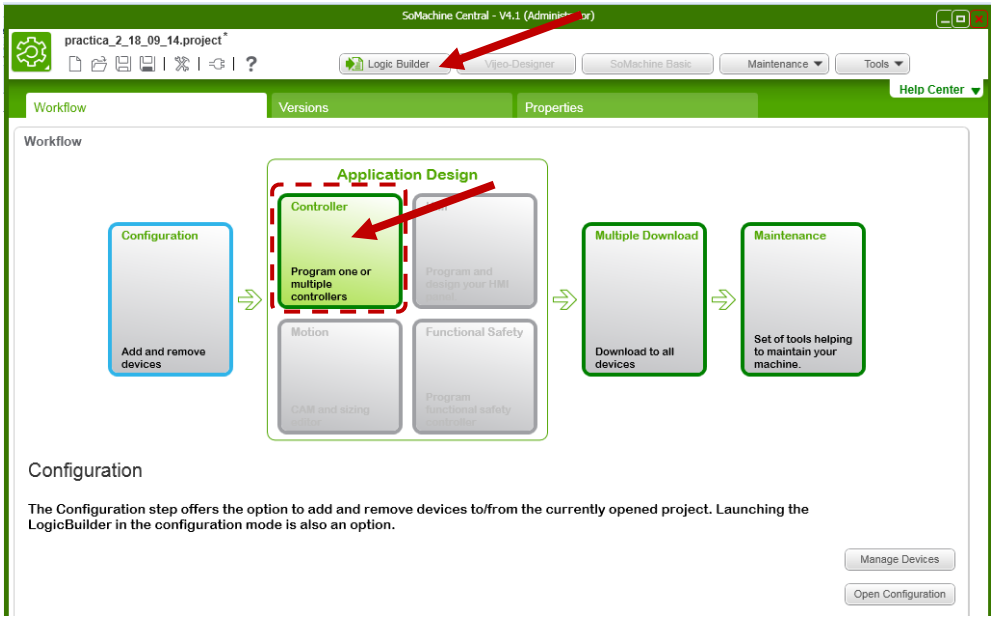
- **Input Filter:** Valor de la ventana de tiempo de filtrado (*quiere decir que para señales que lleguen con ruido y que pueden dar valores no reales debido a las interferencias, se utiliza este filtro para minimizarlo*).

- **Sampling:** Valor de muestreo (*Es el tiempo de muestreo de cada una de las señales analógicas).*

2.8 VENTANA LOGIC BUILDER (PROGRAMACIÓN)

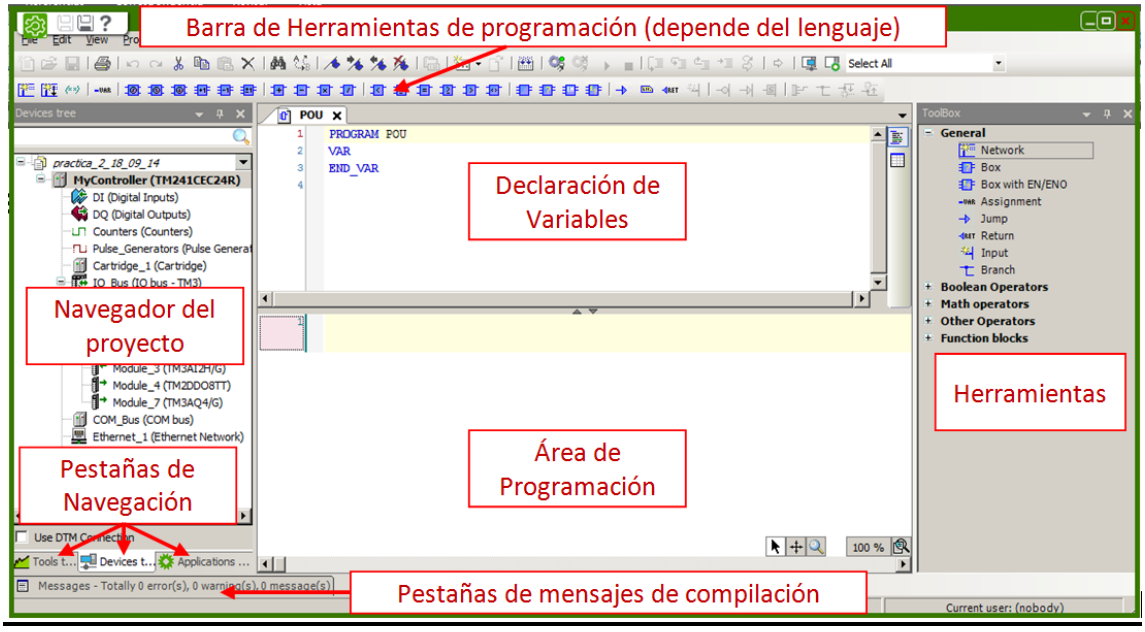
Para abrir la ventana de programación '**Logic Builder'**, desde la ventana general del SoMachine Central, seleccionaremos la pestaña de **'Workflow'** (flujo de trabajo). Dentro de los bloques del flujo de trabajo de un proyecto seleccionamos el bloque **'Controller'** dentro de **'Application Design'.**

O en la barra de herramientas de acceso, que se muestra en la parte superior de SoMachine Central, seleccionar **'Logic Builder'**.



2.8.1 Interface Logic Builder

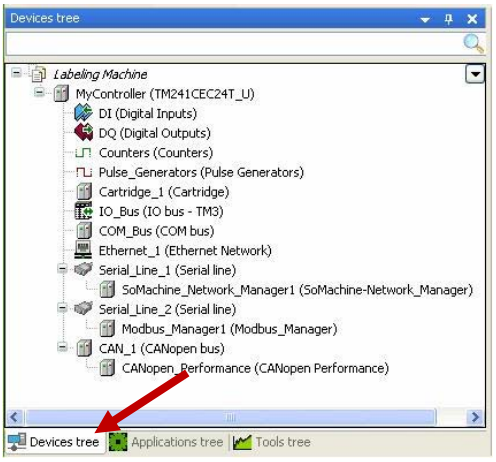
En esta figura se muestra las diferentes áreas del interface de programación.



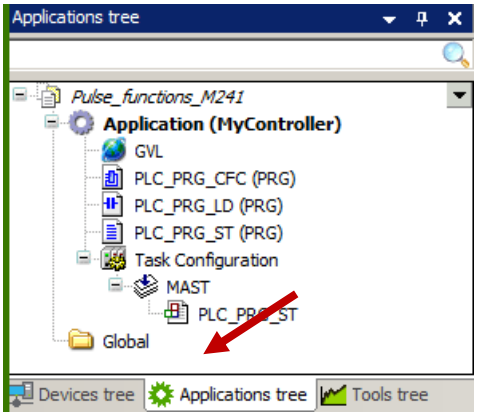
- **Navegador de proyecto:** En esta área se muestra los diferentes elementos que componen el programa, estructurado para una mejor comprensión de que se está haciendo en el proyecto.

  El navegador consta de tres pestañas

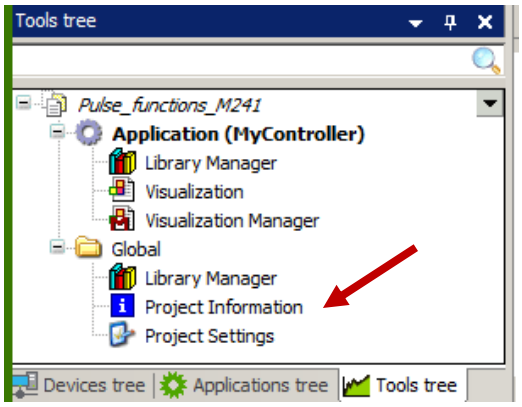
  - **Pestaña de Dispositivos:** se utiliza para configurar el hardware del controlador elegido.



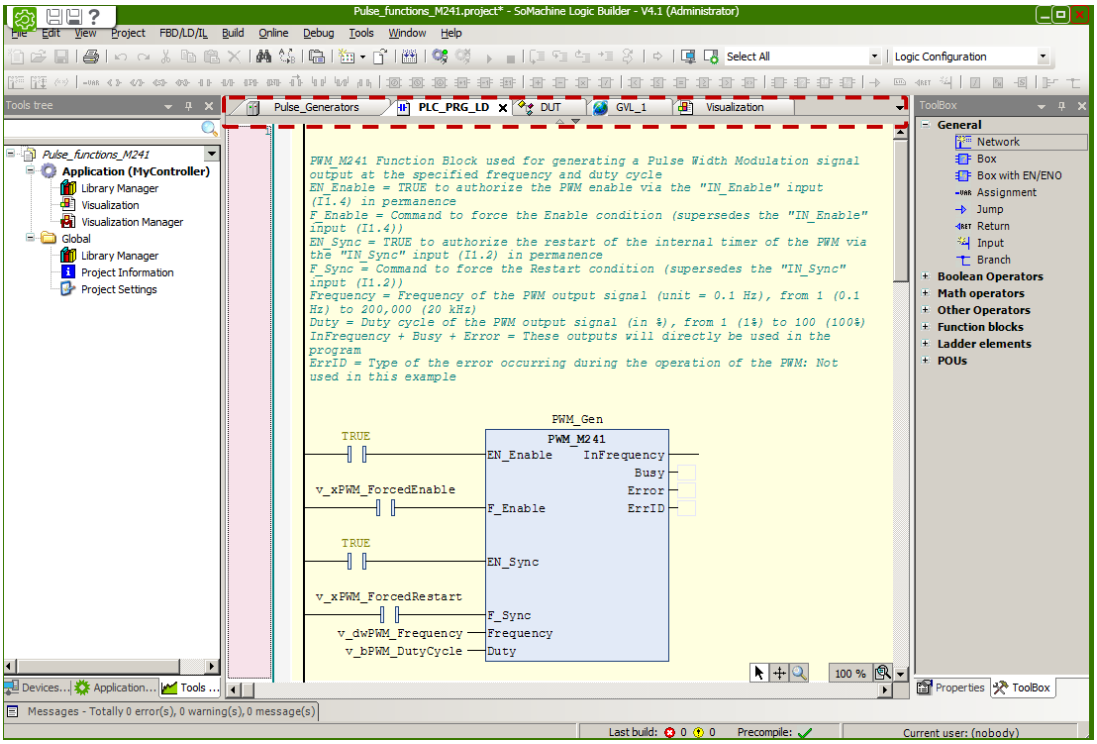
  - **Pestaña de Aplicación:** se utiliza para realizar la programación de la aplicación y todos los elementos relacionados (lista de variables globales, tareas, estructura de datos, visualizaciones, POU's).



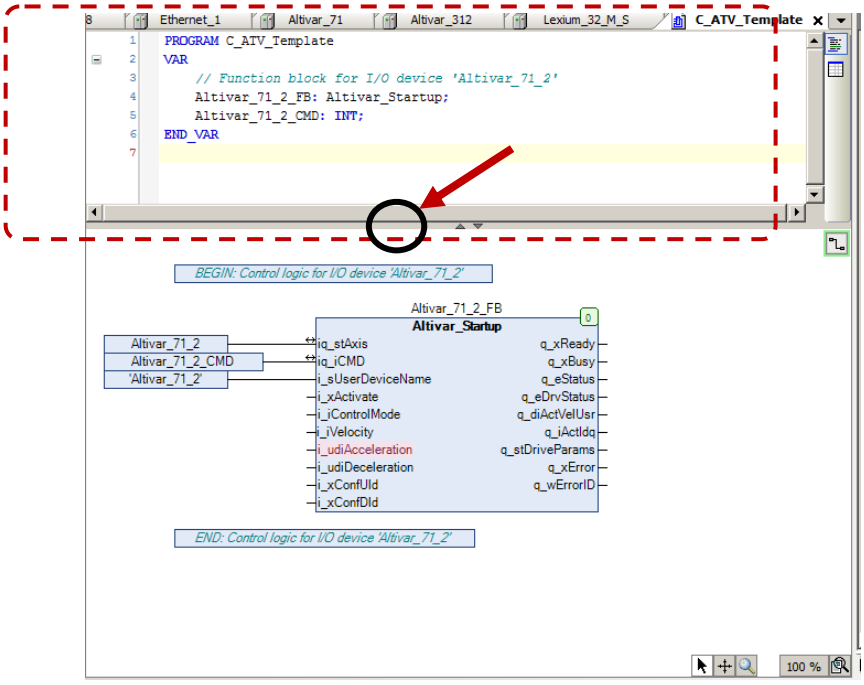
  - **Pestaña de Herramientas:** en esta pestaña se encuentra el administrador de bibliotecas y las visualizaciones que se creen el proyecto.



- **Área de trabajo:** En la zona de trabajo es donde se realizarán todas las parametrizaciones, programación y llamadas de cada uno de los elementos seleccionados en las diferentes pestañas de navegación. Las llamadas realizadas a los diferentes objetos en el navegador quedarán como pestañas de acceso rápido en la parte superior del área de trabajo.



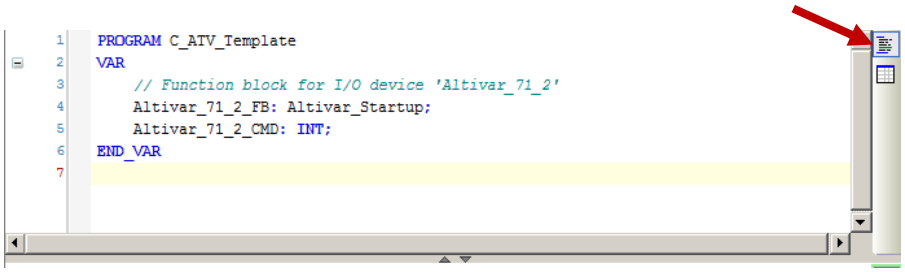
- **Declaración de variable:** Cuando se abre un POU, en el área de trabajo aparece un área privada de declaración de variables de este POU (estas variables serán privadas de este POU).



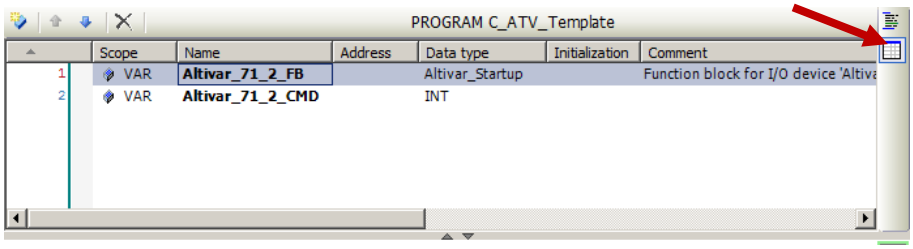
Nota: A veces el área de declaración de variable aparece cerrada la podemos desplegar y plegar pulsando en los pequeños triángulos que se encuentran en el área de programación.

Además podemos seleccionar en que formato queremos declarar las variables, pudiendo utilizar uno u otro en función de lo que más nos convenga, para intercambiar el formato de la declaración, están los iconos en la parte derecha superior del área de declaración.

Formato Texto:

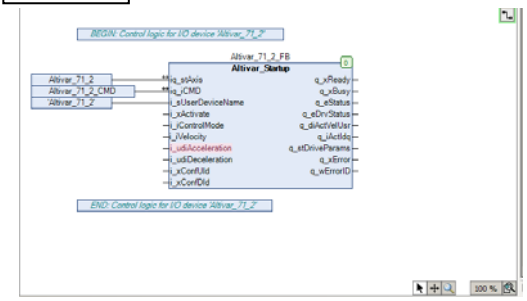


Formato Tabla:

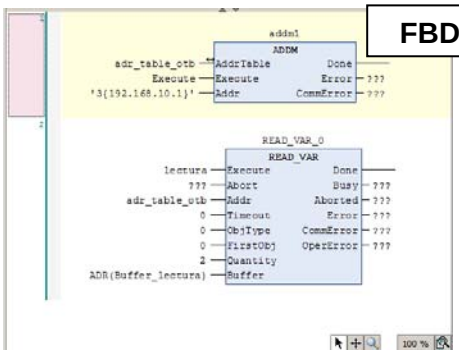


- **Área de programación:** es el área donde se realizarán la programación del POU seleccionado, el formato de visualización dependerá del lenguaje de programación seleccionado.

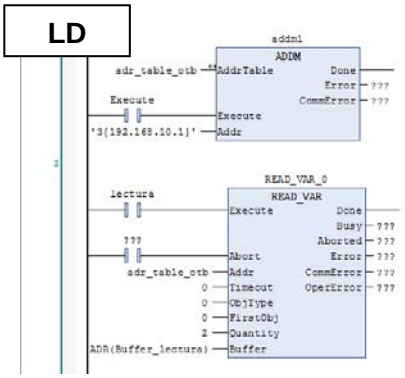
CFC



FBD



LD

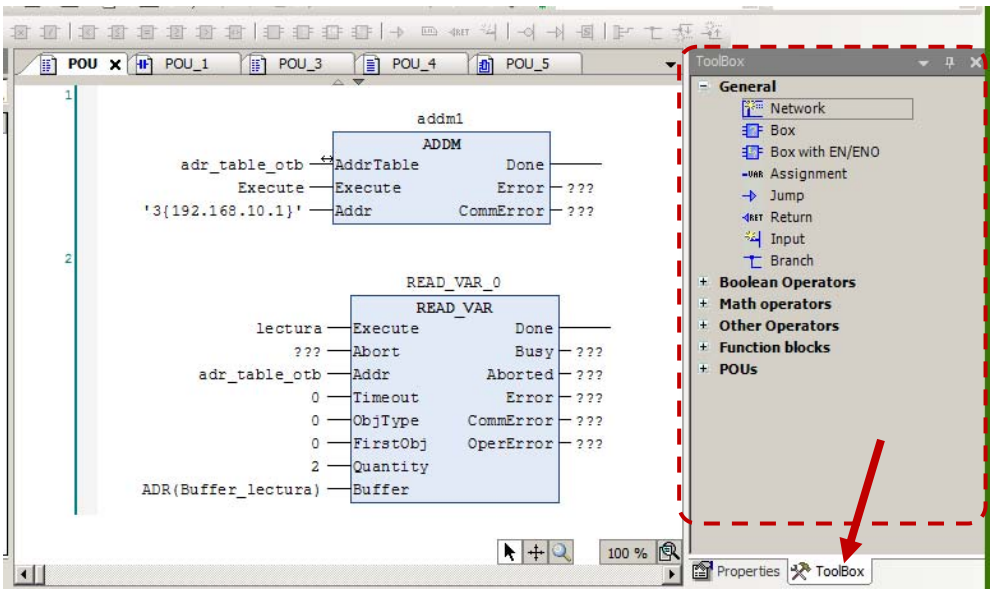


IL

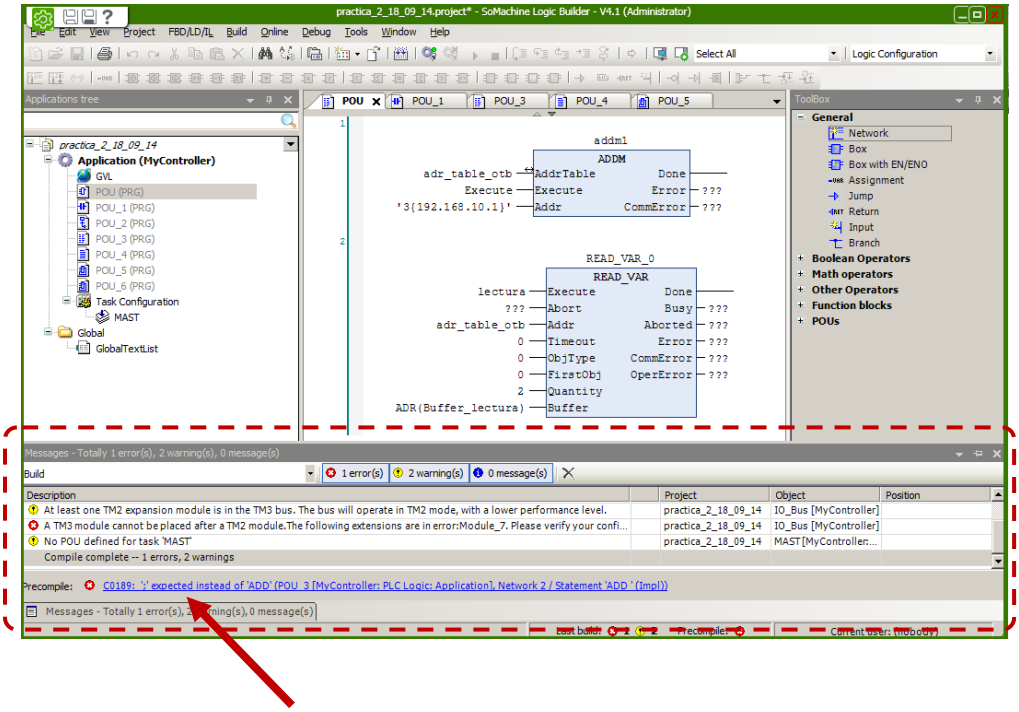
```
CAL             addm1(
  AddrTable:= adr_table_otb,
    Execute:= Execute,
       Addr:= '3(192.168.10.1)',
     Error=> ???,
  CommError=> ???)

CAL             READ_VAR_0(
    Execute:= lectura,
      Abort:= ???,
       Addr:= adr_table_otb,
    Timeout:= 0,
    ObjType:= 0,
   FirstObj:= 0,
   Quantity:= 2,
     Buffer:= ADR(Buffer_lectura),
       Busy=> ???,
    Aborted=> ???,
      Error=> ???,
  CommError=> ???,
  OperError=> ???)
```

- **Herramientas:** Área que muestra las funciones herramientas que se pueden utilizar, para la programación del POU, o del objeto que esté en el área de trabajo en ese momento (Por ejemplo una visualización).

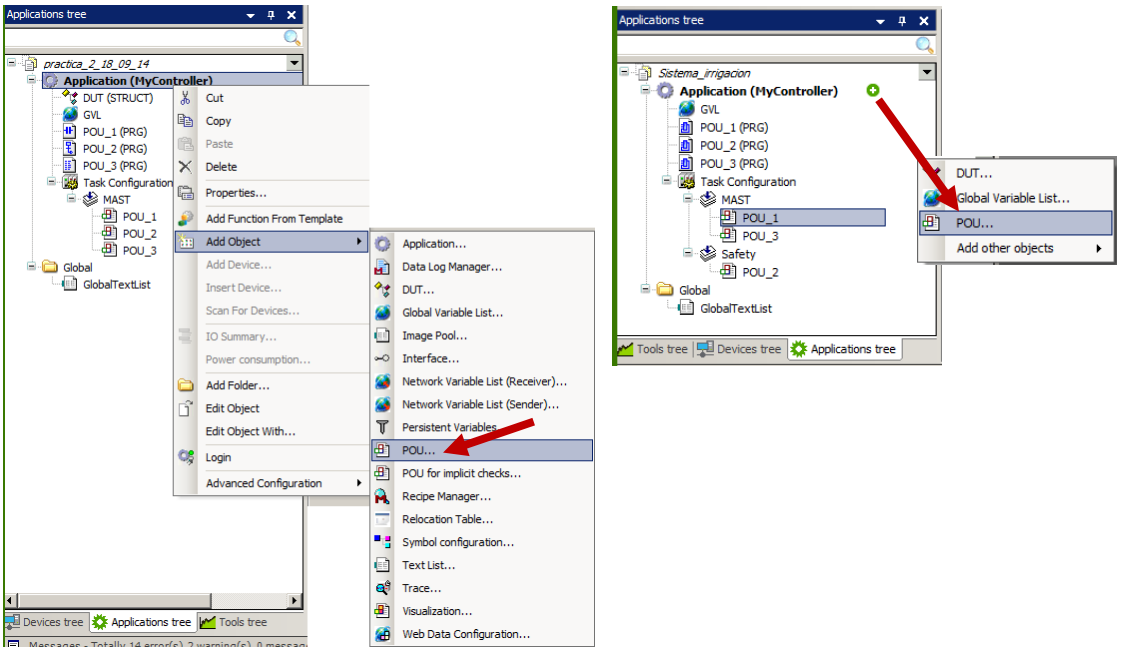


- **Pestaña de mensajes de compilación:** Esta pestaña que normalmente se encuentra en la parte inferior de la ventana de Logic Builder, nos muestra los mensajes de compilación (errores, advertencias y mensajes).

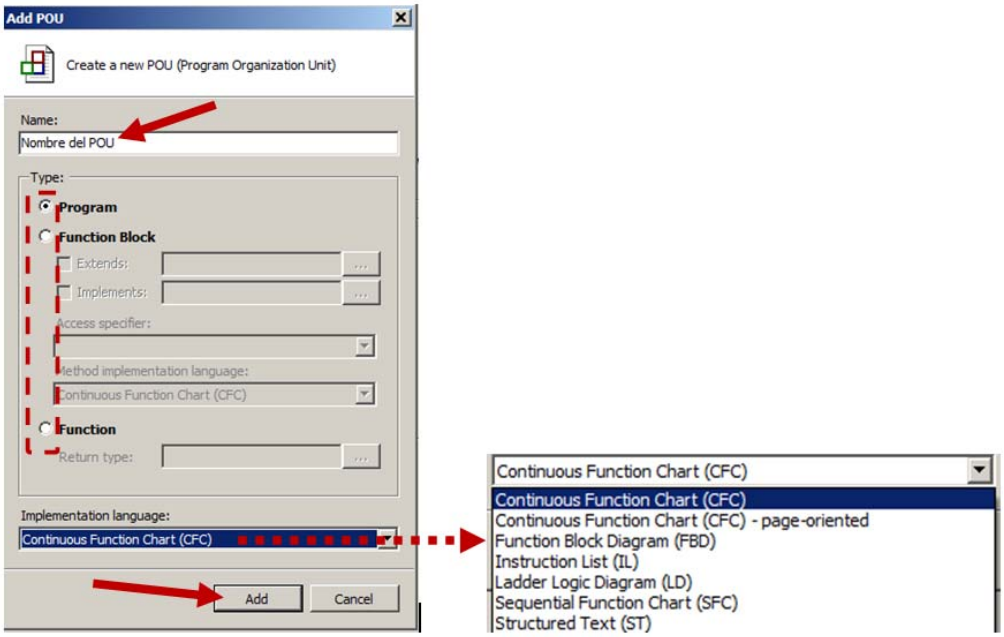


2.8.2 Para crear un POU

Si se selecciona la vista clásica del navegador, hay que ir a la ventana de dispositivos, desplegar el árbol del '**My Controller**', **'Lógica PLC'** y seleccionar '**Application**' hacer clic con el botón derecho del ratón y seleccionar '**Agregar Objeto'. En el caso que tengamos la vista normal hay que ir a la pestaña 'Application tree' desplegar el árbol de la aplicación y pulsamos el icono de '+' que aparece al lado de la aplicación luego seleccionamos 'POU'.**



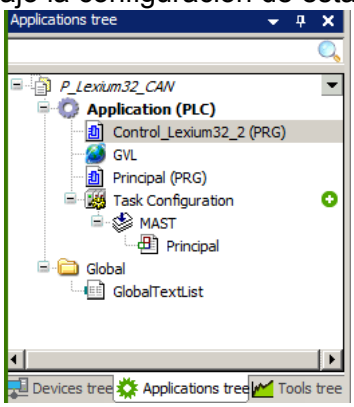
En la ventana flotante de **'Agregar Objeto'**, seleccionar el Objeto '**POU' en la barra de la izquierda de la ventana y después escribir en el campo 'Nombre' el nombre que desea para esa sección de programa, seleccionar el 'Tipo' de POU y el tipo de lenguaje de programación en la opción 'Lenguaje de implementación'.**



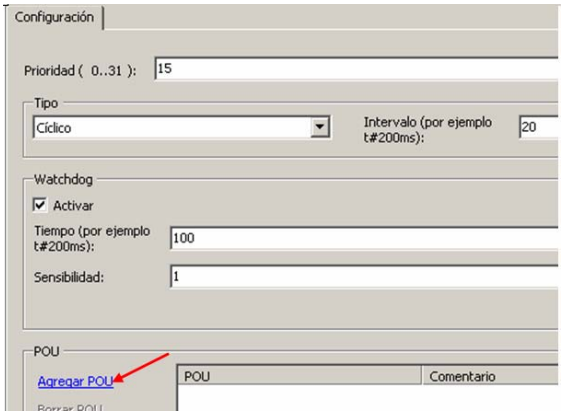
Todos los POU's se crearán de esta manera. Una vez creados, tiene que ser asignados a una **tarea**.

- **Añade un POU a la tarea MAST.**

1. Hacer doble clic en el ítem MAST que hay en la ventana de dispositivos para abrir en la ventana de trabajo la configuración de esta tarea.

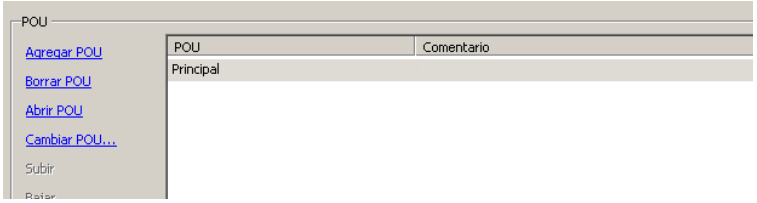


2. En la ventana de configuración seleccionar **'Añadir POU'** dentro del campo **'POU'** del configurador de tareas. Se abrir el asistente para seleccionar el POU que se desea añadir.



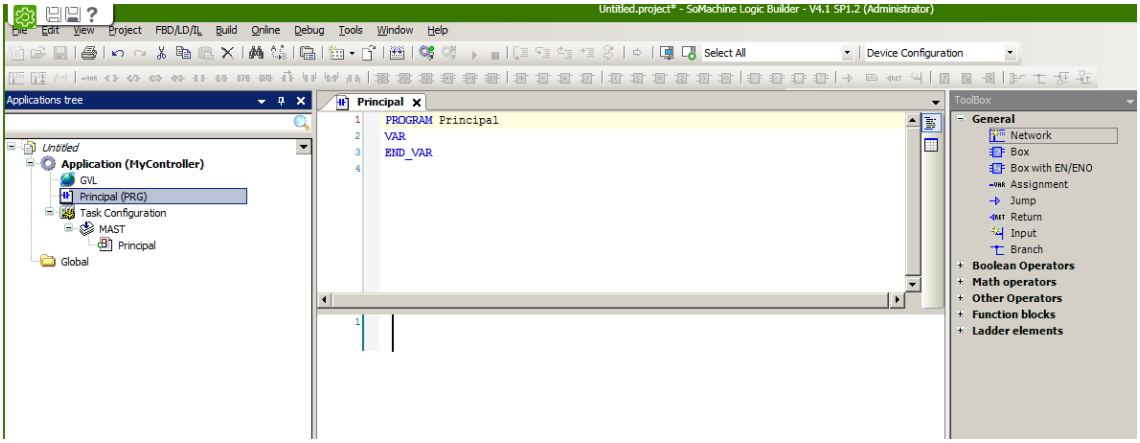
3. Expandir el árbol que sale de '**MyController'** y selecciona el POU **'Principal'** y pulsa el botón **'Aceptar'**.

4. El POU añadido en la tarea MAST aparecerá en el configurador de tarea en el campo POU.



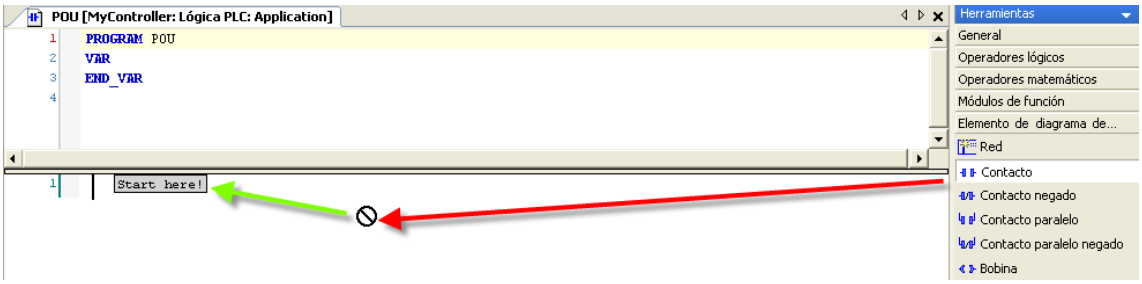
2.8.3 Programación del POU

Una vez nos aparezca la ventana agregar objeto marcaremos sobre **'Principal'** y en el área central aparecerá la zona de programación en este caso en diagrama de contactos.

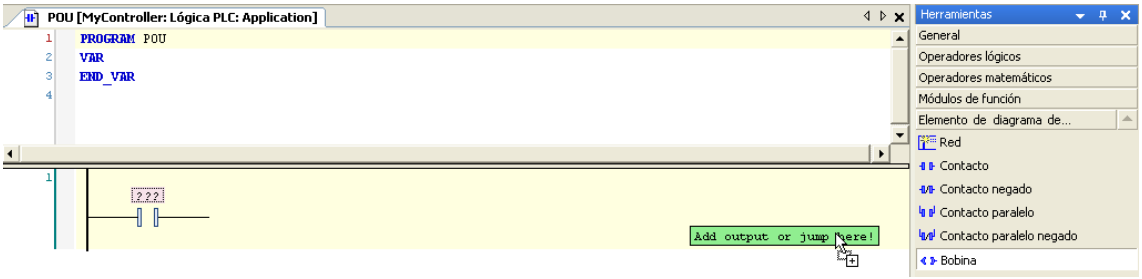


En la pantalla central es donde desarrollaremos nuestro programa. También observamos que en herramientas han aparecido las funciones que podemos utilizar.

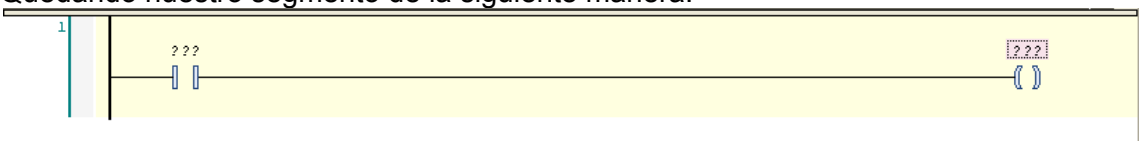
Para este primer proyecto simplemente asignaremos una salida al estado de un contacto, para realizar esto simplemente nos tenemos que dirigir a contacto dentro de herramientas y arrastrar hacia la pantalla central hasta la posición que me indique donde puedo dejarlo.



Para definir una salida procederemos de la misma manera, pero, arrastraremos una bobina.

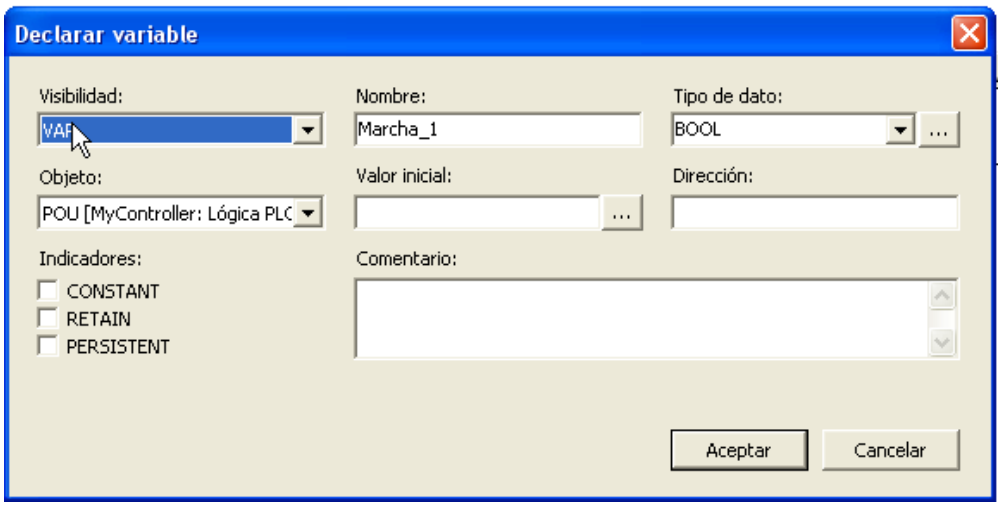


Quedando nuestro segmento de la siguiente manera.



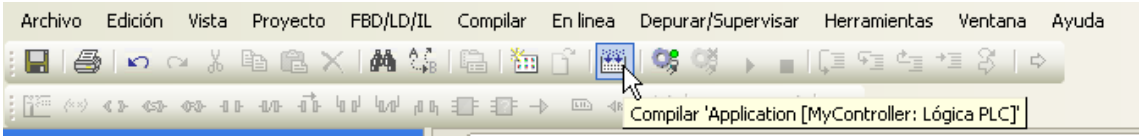
Ahora sólo debemos definir la entrada y la salida, es decir, dar la dirección que vamos a utilizar. Esta dirección podrá ser una variable que nosotros mismos crearemos o una entrada o salida física. Tenemos que tener en cuenta que si vamos a simular el programa las entradas físicas no nos dejará forzarlas.

Para declarar la variable solo tenemos que ponernos encima de los contactos o elementos que vayamos a definir (en el interrogante). Al darle un nombre aparecerá la siguiente pantalla si no es una variable anteriormente definida.



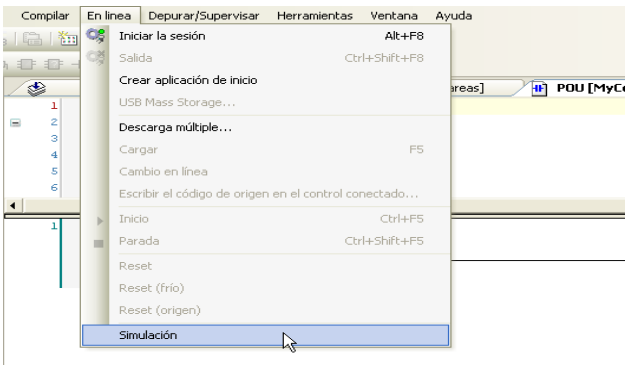
En esta ventana vamos a definir qué tipo de variable será. En visibilidad diremos que tipo será si es de entrada, de salida, global. En tipo de dato indicaremos si es de tipo Booleano, byte, Word. En dirección le podríamos indicar una entrada física o una dirección de memoria de marcas. Con la salida procederemos de la misma manera.

Una vez realizado vamos al icono compilar y vemos si hemos realizado algún error.



2.8.4 Simulación del programa

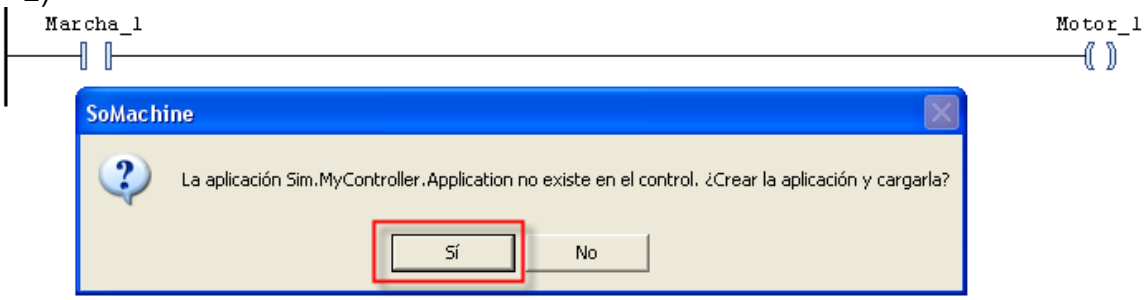
Si no da ningún error procederemos a la simulación o la transferencia del programa, para la simulación nos dirigimos a **"En línea"** en la barra de Herramientas, Y damos a **"Simulación".**



Después a los iconos de “**Iniciar Sesión”** si no tenemos creada la aplicación nos preguntará si la crea y después **“inicio**”.

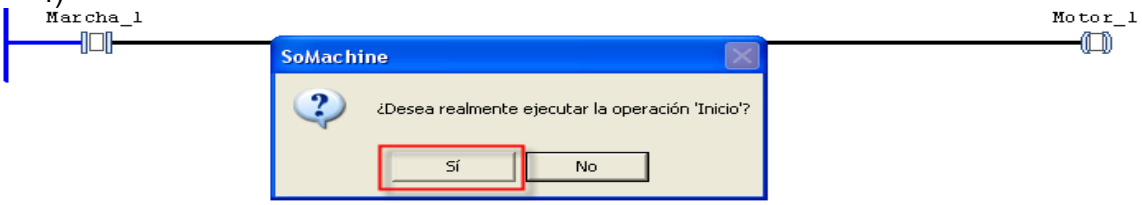
1)

2)

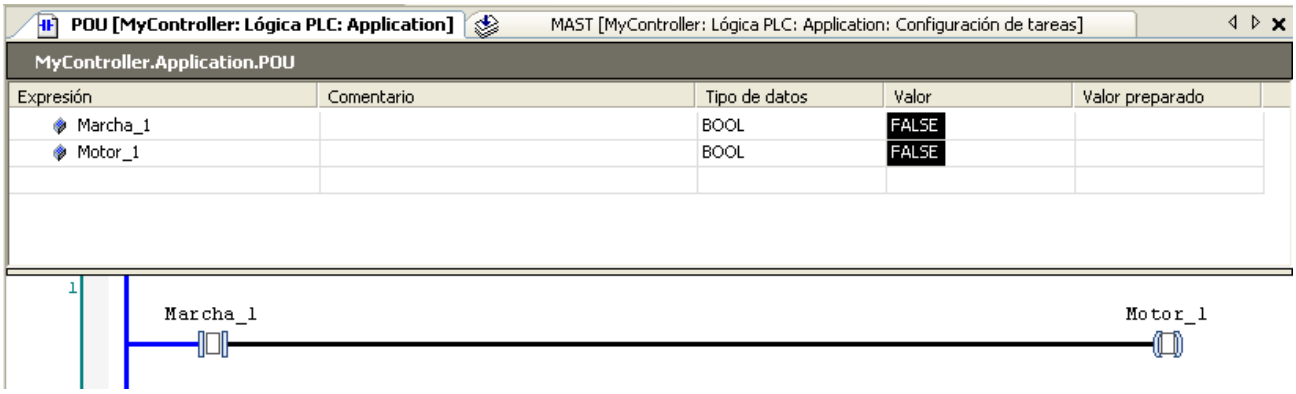


3)

4)

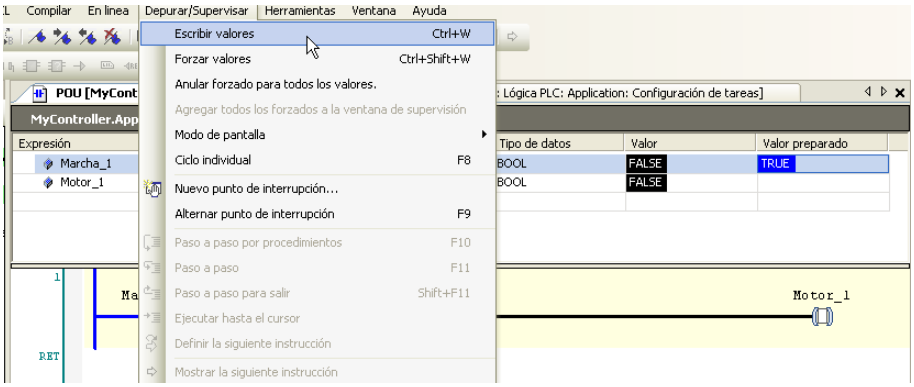


Y aparecerá en la ventana central lo siguiente.

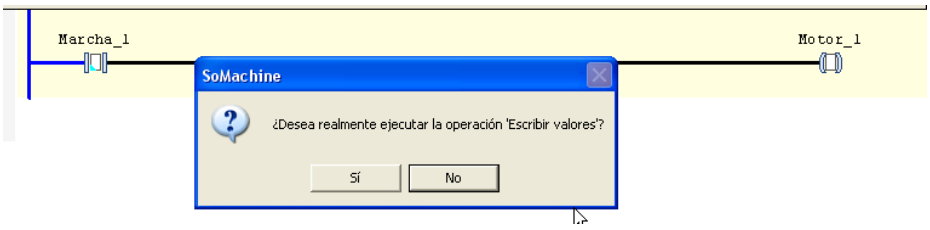


Para cambiar el valor a una variable deberemos dirigirnos a **“Valor preparado”,** clicar sobre él, poniendo así el valor de true, seguidamente en **“Depurar/Supervisar”** y dentro de esta **“Escribir valores”.** De esta manera es como si la entrada hubiese cambiado de estado y se asigna su valor a la salida.

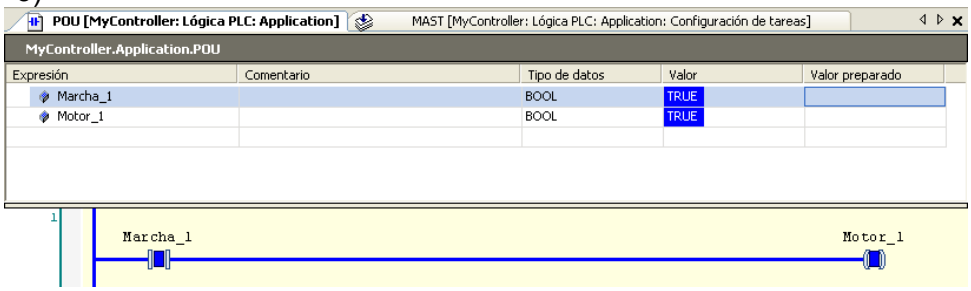
1)



2)



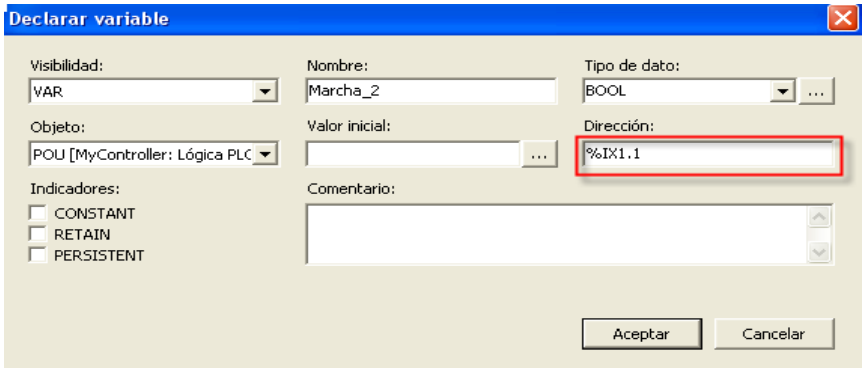
3)



Para abandonar la simulación solo deberemos dar en el icono **“Parada”** y **“Salida”** en este orden.

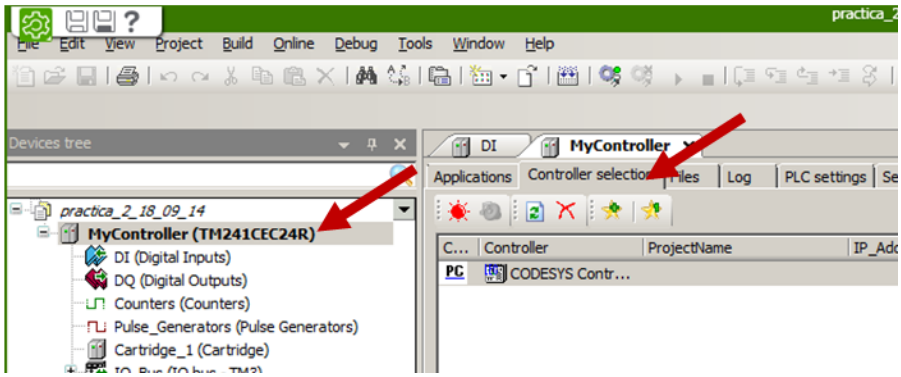
2.8.5 Transferencia del programa al controlador

Para realizar la transferencia, vamos a crear un segmento nuevo, para utilizar variables físicas y de esta manera poder ver el resultado de nuestro programa en el PLC. La única diferencia con la creación de las variables anteriores es que, en el apartado dirección le vamos a dar la dirección booleana de una entrada y a la bobina una de salida.



Si no fijamos en la pantalla superior una variable sin dirección de una con dirección cambia a la hora de su definición.

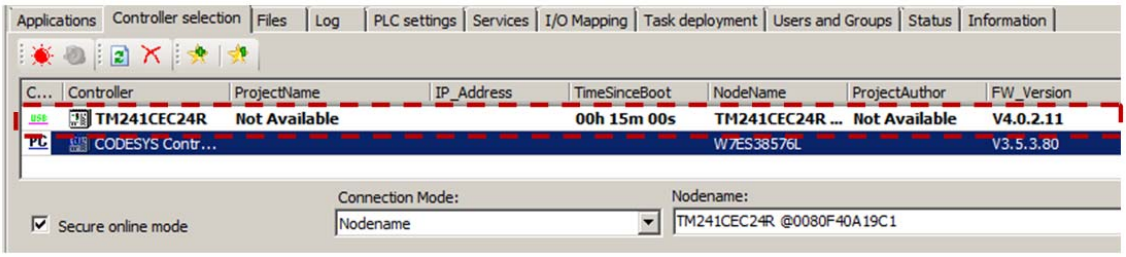
Una vez modificado nuestro programa, volveremos a compilar, por si hemos cometido algún error, si no es así, procederemos a transferir nuestro programa. Tenemos dos opciones para establecer la comunicación con el PLC o a través de la puesta en marcha o clicando sobre MyController.



Inicialmente se tiene que tener conectado con el cable de programación el PC al controlador (cable de programación **USB - Mini USB** para los controladores M238, M241, M251, M258, LMC058 y ATVIMC para las pantallas XBTGC el cable será el XBTZG935). Para acceder añadir un Gateway, hacer doble clic sobre el controlador en la ventana de dispositivos y en el área de trabajo aparece las diferentes pestañas de configuración del controlador, en esta seleccionar la pestaña **'Controller selection'**.

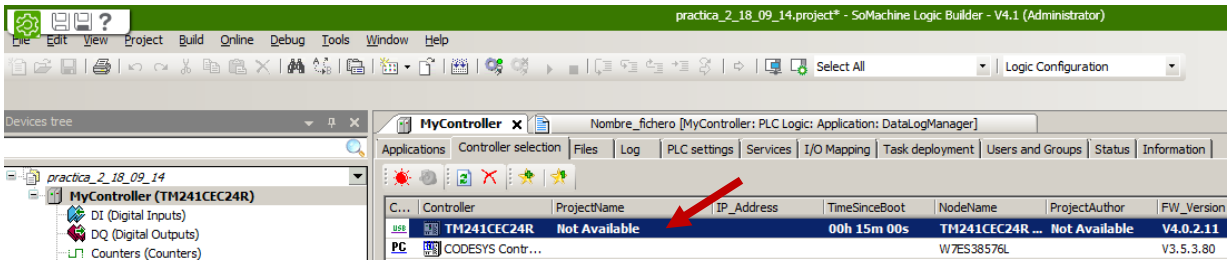
Si está conectado el cable y ha reconocido el driver automáticamente aparecerá un controlador en negrita donde nos especifica una serie de información.

- **Conexión Mode:** especifica el tipo de conexión (USB, ETH, PC).
- **Controller:** Indica el modelo (referencia) del controlador.
- **Project Name:** nos informa del nombre del proyecto que hay cargado en el controlador (si no tiene cargado ninguno porque el controlador es nuevo pondrá **'Not Available'**)
- **IP Address:** Dirección IP guardada en el equipo (si dispone de ella).
- **Time Since Boot:** El tiempo que lleva en marcha el controlador desde el último inicio.
- **Node Name:** es el nombre lógico que tiene este dispositivo en concreto, normalmente por defecto está compuesto por la referencia del controlador, @, y la MAC del equipo o el número de serie, dependiendo si tiene Ethernet o no.
- **Project Author**: Especifica el autor del proyecto que hay guardado si se ha especificado en el campo '**Autor'** de propiedades del proyecto.
- **FW_Version:** Indica la versión de firmware del controlador.



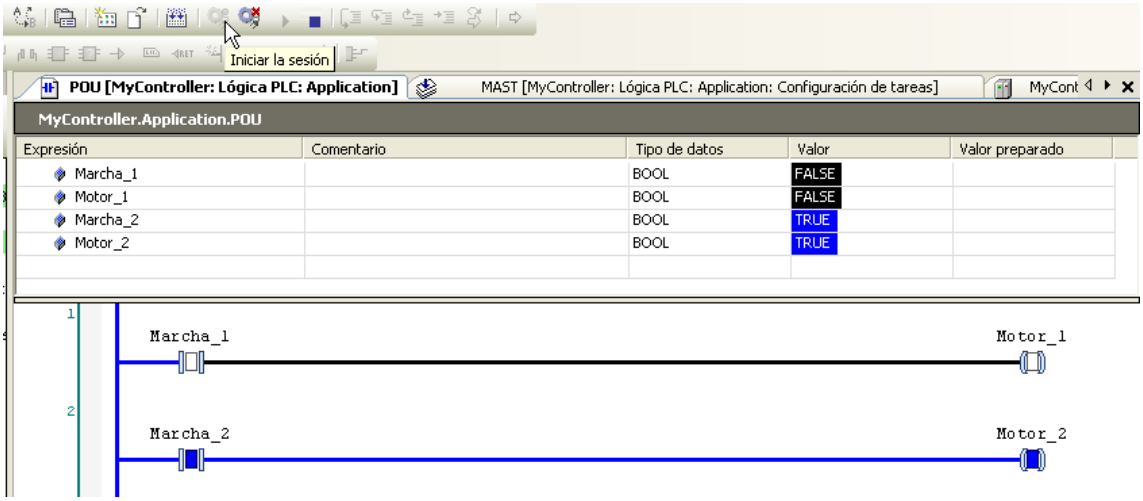
Si queremos cambiar de controlador al que conectarse ó de método de conexión, podremos cambiarlo haciendo doble clic sobre esa línea, que aparecerá ahora resaltado en negrita.

Una vez tenemos elegido el controlador al que nos queremos conectar (resaltado en negrita), para conectarnos solo tenemos que ir al menú textual **'Online >> Login'** ó pulsar en el icono de Login  de la barra de herramientas.



Nos saldrá la pantalla siguiente damos “**Alt + F**” t habremos establecido la comunicación con el PLC.

Ahora podemos comprobar en la ventana del programa si este funciona correctamente. Damos a iniciar sesión y podremos ver como están las entradas y las salidas físicas en el PLC.



CAPÍTULO

3

3. PROGRAMACIÓN BÁSICA – LADDER (LD)

Vamos a desarrollar a continuación los elementos que disponemos en este menú para poder desarrollar nuestra programación.

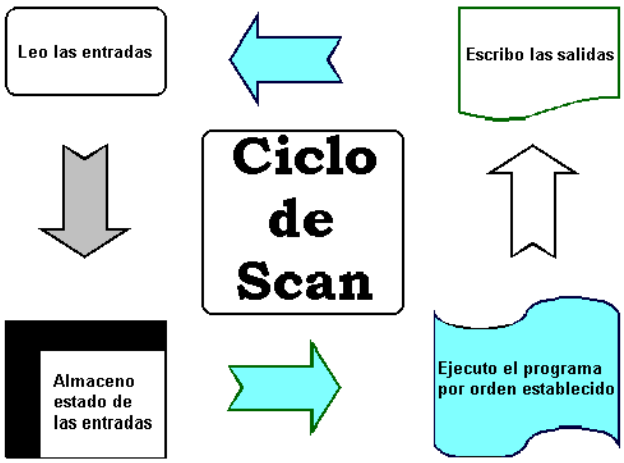
Antes de comenzar a programar debemos recordar el funcionamiento básico de un automáta. Este cada x tiempo ejecuta la programación que le hayamos transferido, en Unity ya sabemos que este tiempo lo decidimos a través de las tareas. Pero en todas es ellas la forma de procesar simpre será igual.

1º Mirará el estado de las entradas

2º Depositará el resultado lógico de estas, en la zona de memoria destinado a las entradas.

3º Ejecuta el programa y define como quedarán las variables una vez que termine de ejecutarlo.

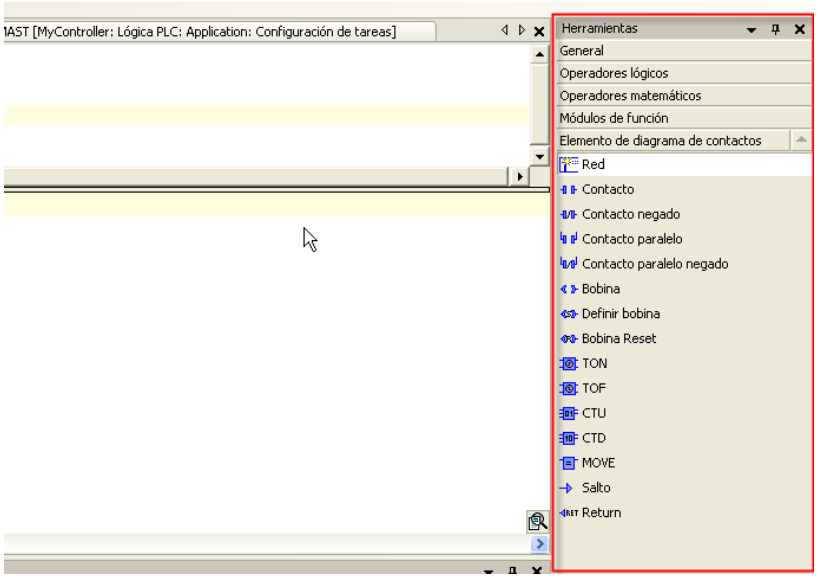
4º Cambia el resultado de las variables que deba cambiar y mantiene las que tenga que mantener.



Hay que hacer especial mención a esto, por que el autómata lo que mira es el estado de sus entradas y para el un pulsador abierto será un 0 y un pulsador cerrado será un 1. Nosotros vamos a realizar un programa para cuando la entrada esté a 1 y/o esté a 0.

También tenemos que tener claro que el orden de ejecución será por orden de programa, osea desde el primer segmento al último.

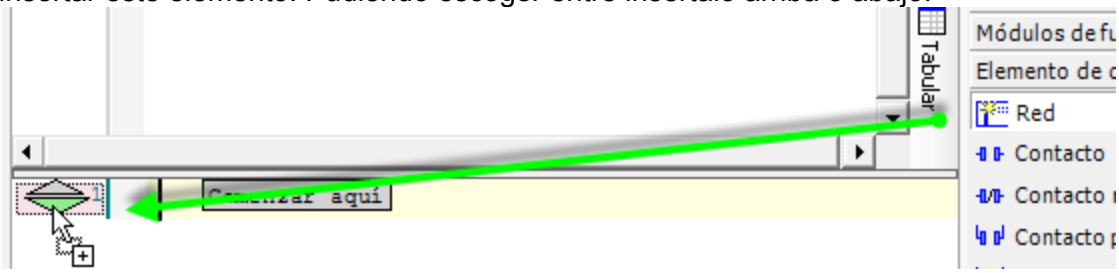
3.1 Elemento de diagrama de contactos



Como podemos observar aquí tenemos varios submenús y dentro de cada uno de estos tendremos diferentes funciones.

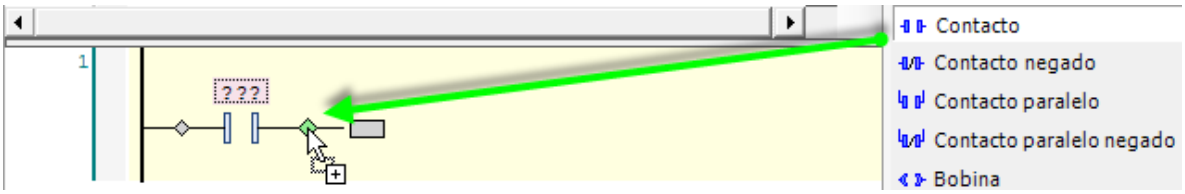
3.1.1 Red

Cada vez que queramos introducir una línea de programa (segmento) deberemos insertar este elemento. Pudiendo escoger entre insértalo arriba o abajo.

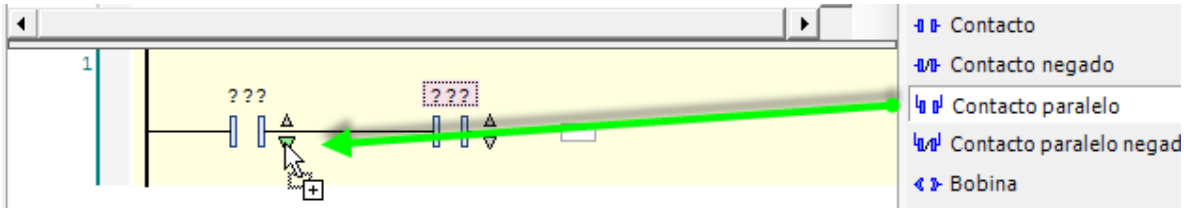


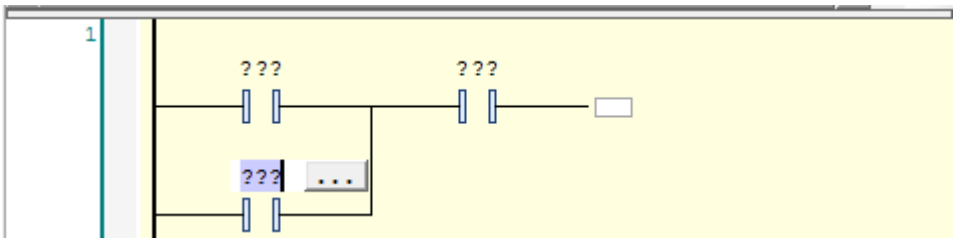
3.1.2 Contacto y contacto en paralelo

Sirve para introducir una lógica binaria, tanto en serie como en paralelo, en la que el estado de las direcciones que hemos asignado es la que queremos que se vea reflejado en el resultado lógico, por ejemplo, físicamente hemos cableado un pulsador NC a la entrada %I1.0 el resultado que tiene asignado será 1 cuando pulsemos este el resultado será 0. Si queremos que la lógica se cumpla cuando sea 1.



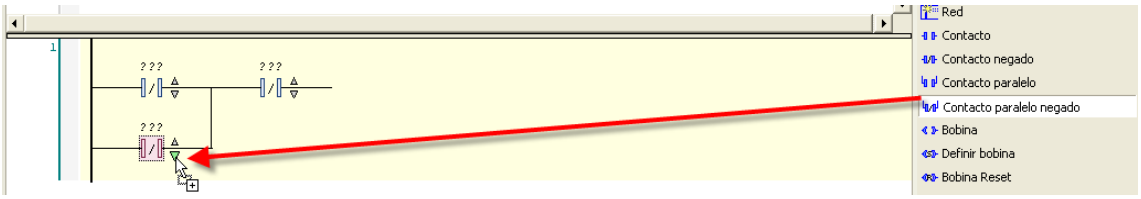
El anterior ejemplo sería para introducir contactos en serie, y a continuación vemos como introducir contactos en paralelo.





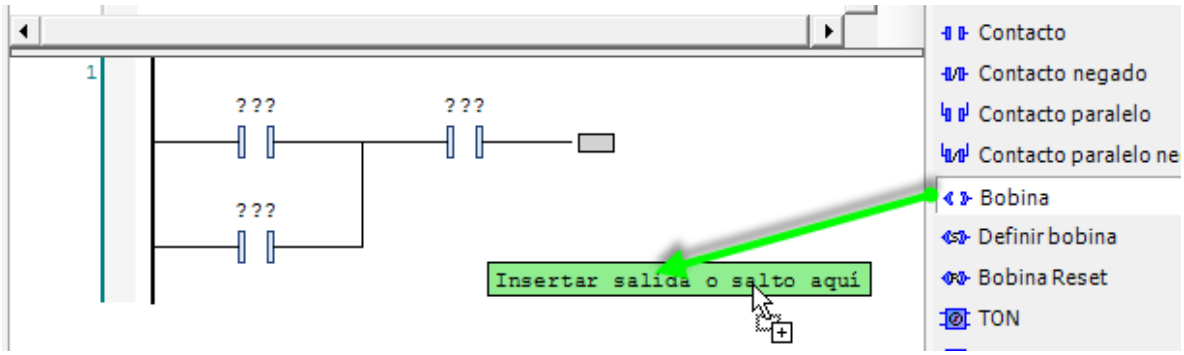
3.1.3 Contacto negado y paralelo negado

Cuando queremos que una dirección concreta haga lo contrario a su estado físico utilizaremos este contacto, es decir si una entrada física se encuentra abierta el programa la considerará un 1, por el contrario si una entrada se encuentra cerrada el programa la considerará un 0. Si queremos que la lógica se cumpla cuando sea 0.

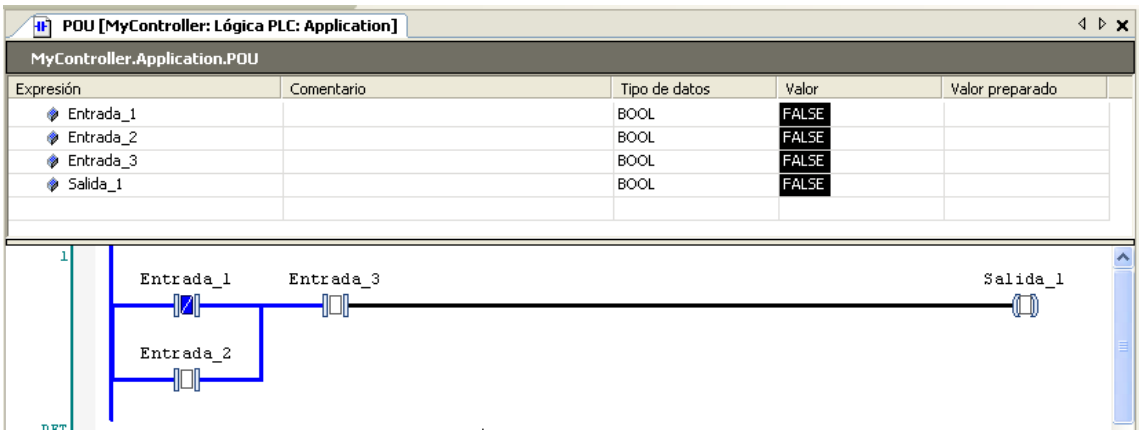


3.1.4 Bobina

La bobina la situamos al final de la red y recibirá el resultado lógico que tenga nuestra combinación de contactos, es decir, recibirá el resultado de la operación.

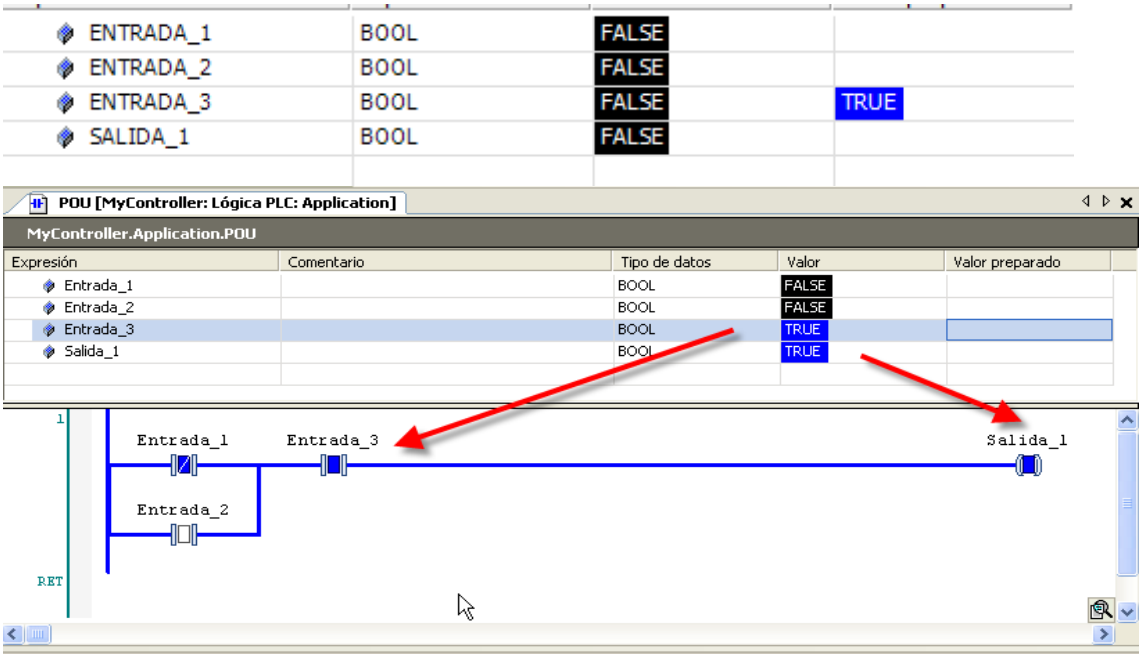


Aprovechamos y nombramos las entradas y salidas como hemos visto en el apartado anterior.



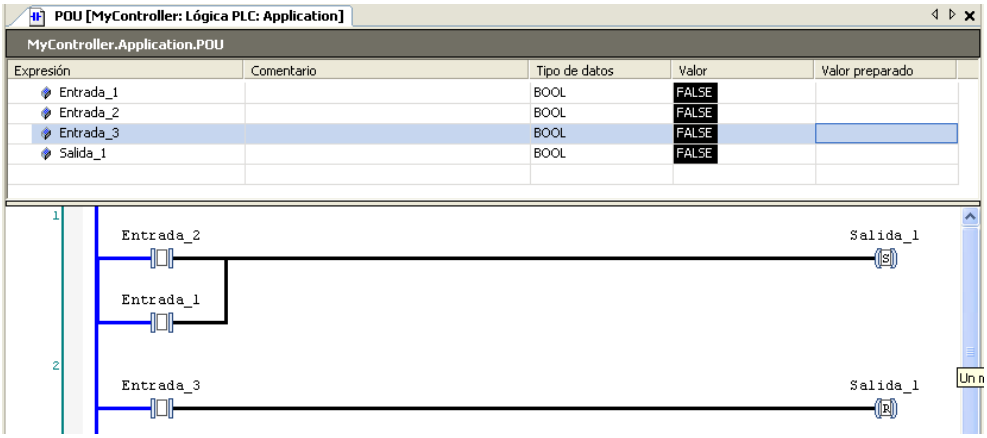
Como podemos observar la “Entrada_1” tiene como valor 0 por lo que al tener un contacto negado en la secuencia lógica tenemos un 1 (la simulación coloca el estado 1 en azul y el estado 0 en negro). La salida esta tras un enlace negro por lo que su estado es 0. Para que la salida estuviese en 1 deberíamos cambiar el valor de la “Entrada_3” a “TRUE” Para que así la serie se cumpliese y asignar el resultado 1 a la “Salida_1”.

Para poner la entrada 3 a”TRUE”, clicamos sobre “Valor Preparado” aparecerá “TRUE” y clicamos “CTRL+F7”

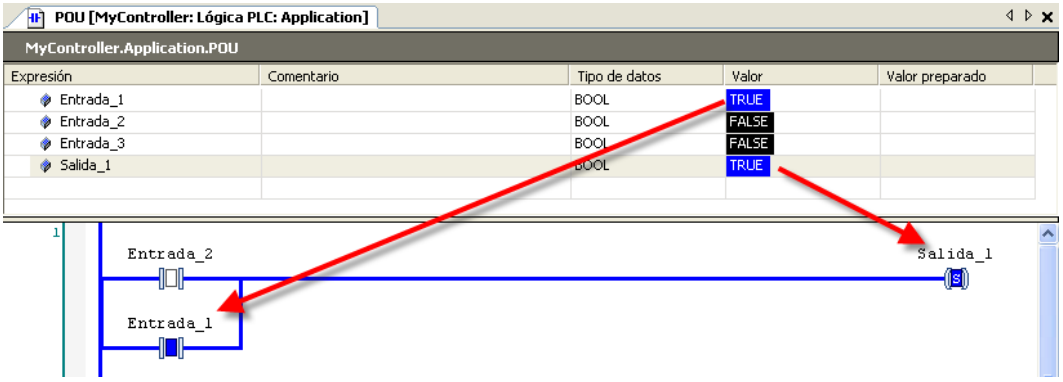


3.1.5 Definir Bobina Set y Bobina Reset

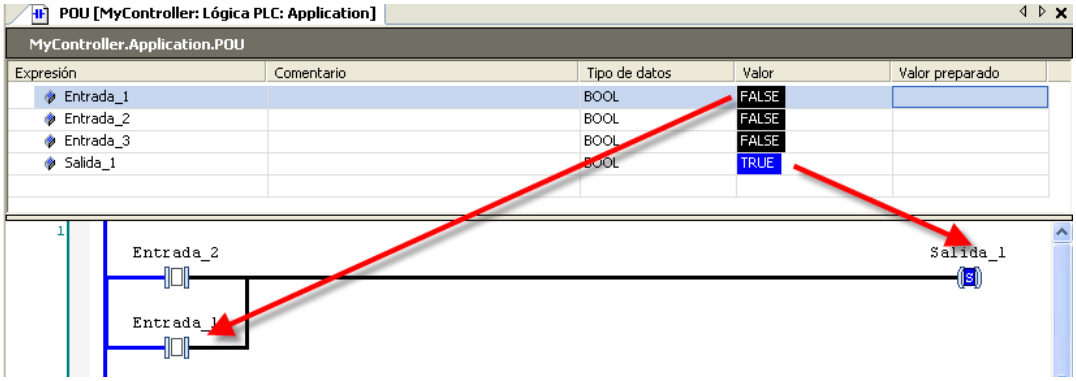
Estas dos funciones sirven para activar y desactivar una salida o variable permanentemente, es decir, si nosotros asignamos una entrada a un “Definir Bobina” cada vez que la entrada este a 1 la salida se pondrá a uno y permanecerá así, aunque la entrada pase a 0, hasta que mediante otra entrada que tengamos asignada a “Bobina Reset” se ponga a 1 desactivando de esta manera la variable, aunque la entrada pase a 0.



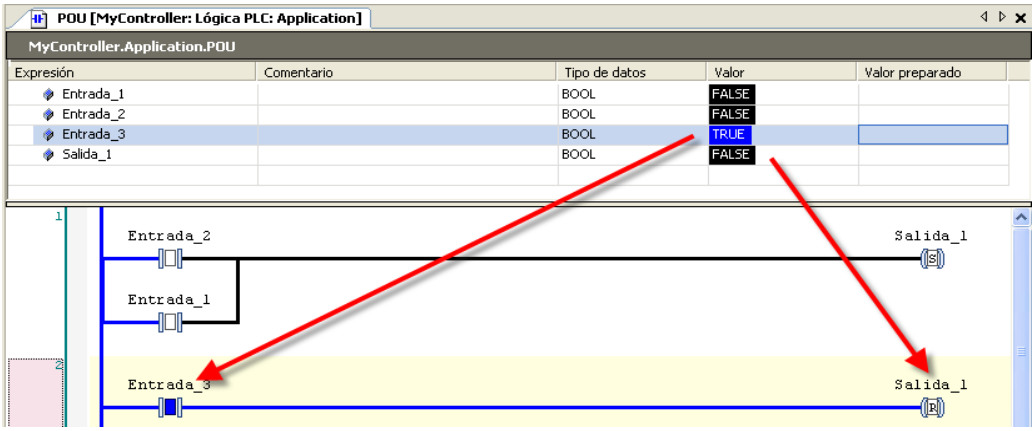
Como podemos observar las tres entradas están a 0 por lo que la salida está a 0, para poner la salida a 1 deberemos mandar un 1 con la “Entrada 1” o la “Entrada 2” a la bobina (S).



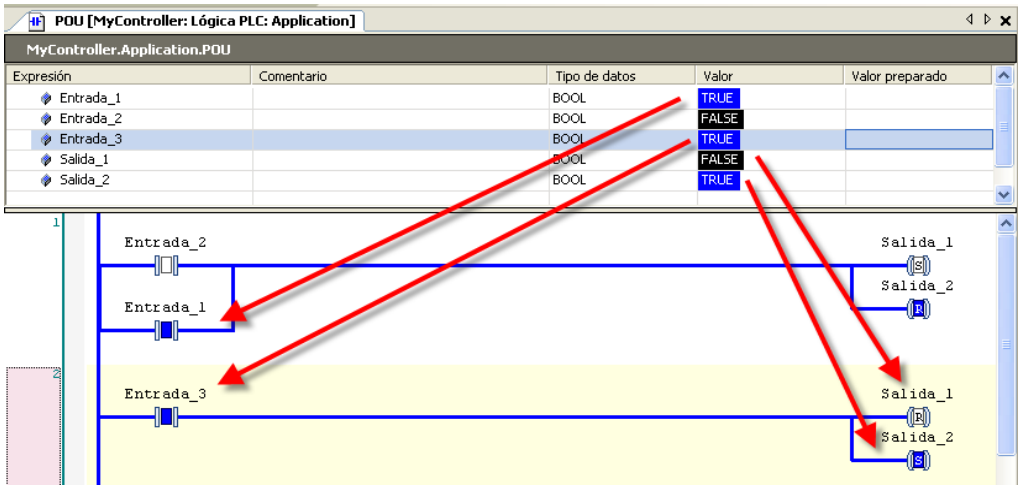
Si ahora volvemos a poner la “Entrada_1” a 0 veremos como la salida no se altera y sigue en estado 1.



Para que la salida se desactive deberemos poner un 1 en la “Entrada_3” para que de esta manera le demos un 1 a la “Bobina Reset”.



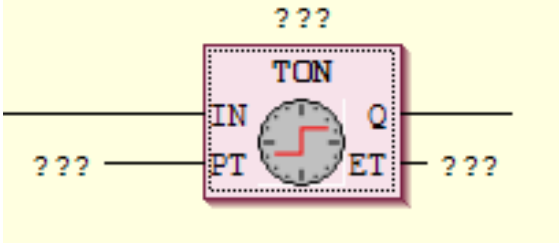
Si en algún momento tenemos las dos bobinas a 1 tendrá prioridad la que se encuentre en la línea inferior.



3.2 TEMPORIZADORES

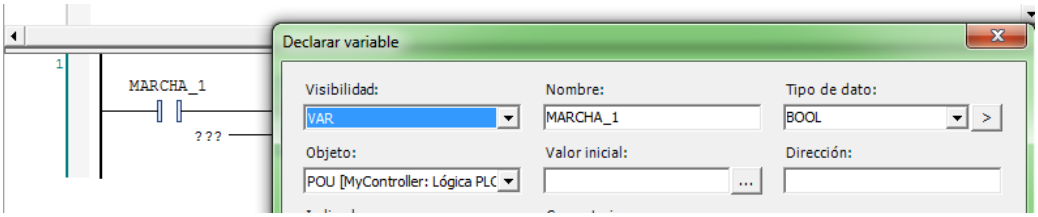
3.2.1 TON Temporizador de retardo a la conexión

Este módulo nos permite establecer una base de tiempo.



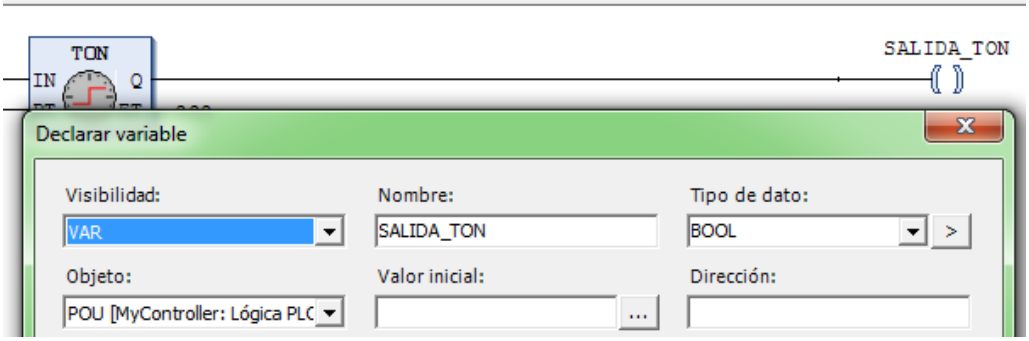
Como podemos observar, dispone de tres campos a rellenar con interrogantes. En “PT” debemos poner el tiempo que debe contar el temporizador. En “ET” la variable donde se depositará el valor que tiene el temporizador en cada momento. En la parte superior describiremos las caracteristicas del temporizador y el nombre del mismo.

Vamos a ver como rellenar estos campos paso a paso.
En “IN” ponemos un contacto que por ejemplo se llamará “MARCA_1” y será un bool.
En “Q” vamos a poner una salida booleana, esta estará en 1 cuando el temporizador esté a 1.

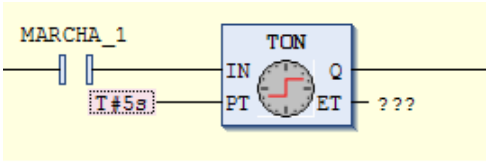


Si no vamos a tener una dirección física se puede dejar así, sino en Dirección pondremos la que le corresponda.

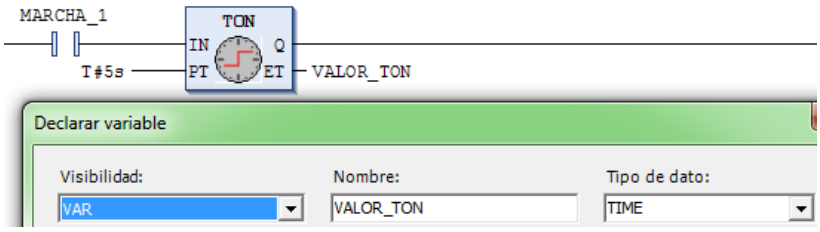
Con la salida es igual si no es física podemos dejarlo tal cual.



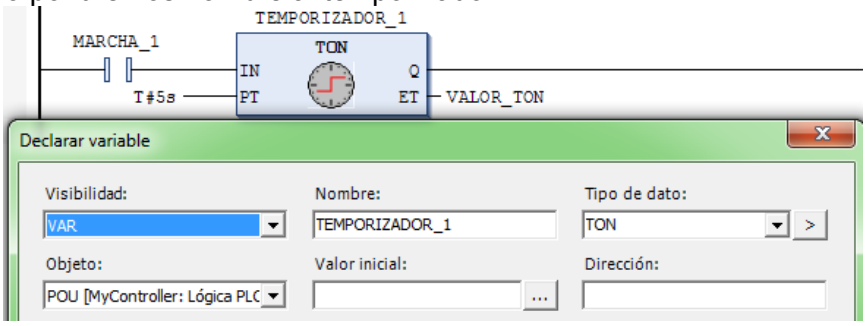
Para “PT” ponemos la base de tiempos, siempre con T# y despues el valor, 5s, 2s3ms, 2m3s2ms, etc.



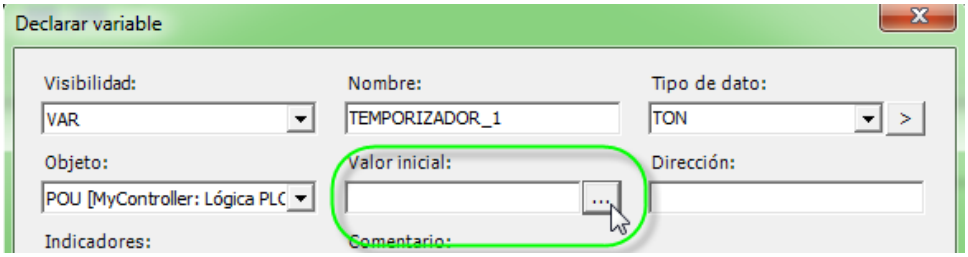
En “ET” pondremos una variable de tiempo donde se va a guardar el valor real del temporizador.



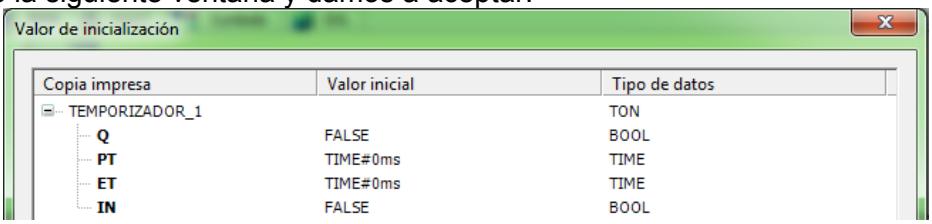
Y por último pondremos nombre al temporizador.



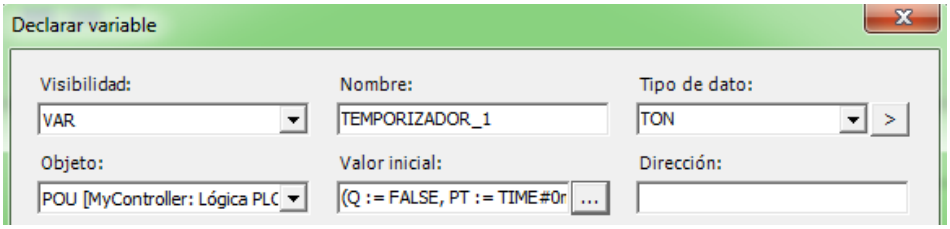
Aquí deberemos dar un “Valor inicial” a la variable, Clicamos sobre los puntos suspensivos .



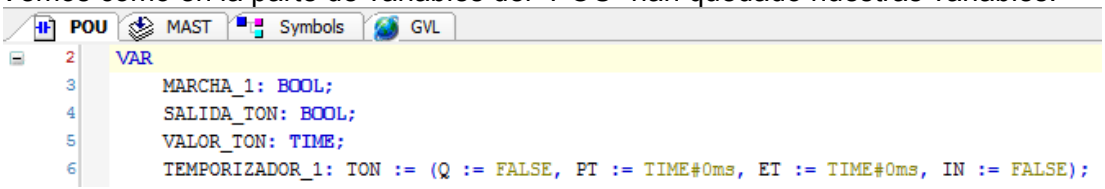
Aparece la siguiente ventana y damos a aceptar.



Y aceptar.

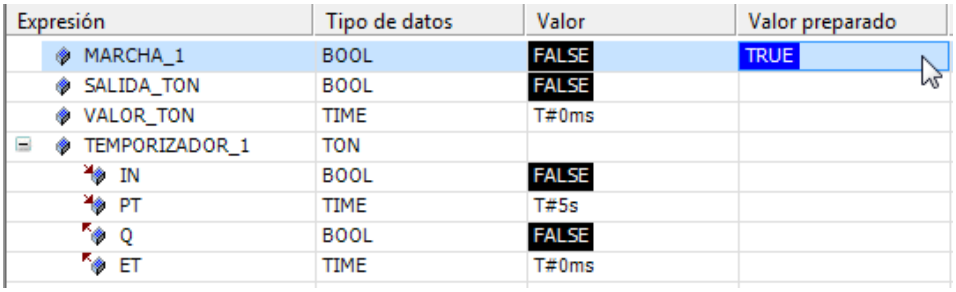


Vemos como en la parte de variables del “POU” han quedado nuestras variables.



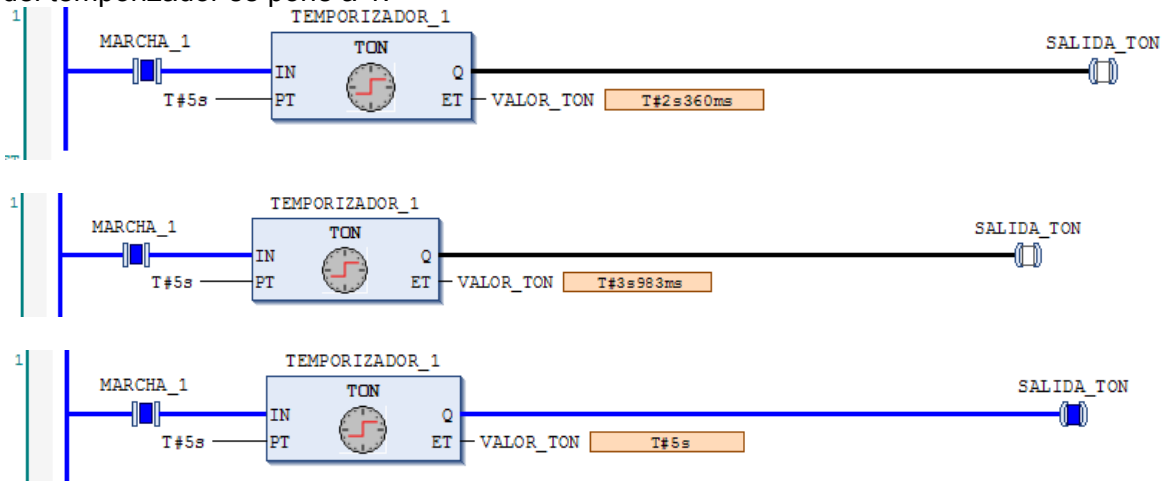
El funcionamiento es el siguiente, cuando en la entrada del temporizador “IN” colocamos un 1 empieza a contar la base de tiempo prefijado en “PT”. Una vez transcurrido este tiempo el temporizador se pone a 1 y también la salida “Q”.

Vamos a ver la simulación del programa. Si clicamos sobre “Valor preparado” en la variable marcha nos aparece “TRUE” en color azul. Si damos a la vez **“Ctrl + F7”** escribiremos el valor preparado en la variable. Si solo clicamos “F7” dejaremos la variable forzada.



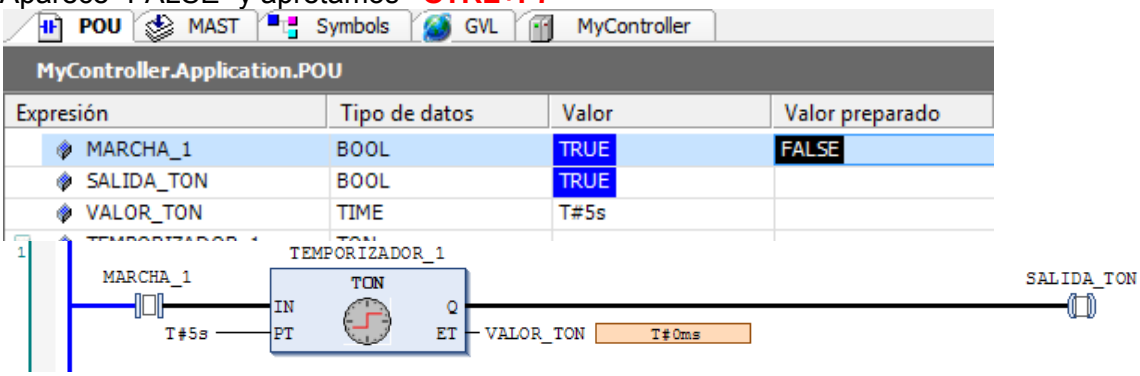
Clicamos “CTRL +F7”.

Vemos como en la salida “ET” va aumentando el valor y cuando llaga a 5s la salida del temporizador se pone a 1.

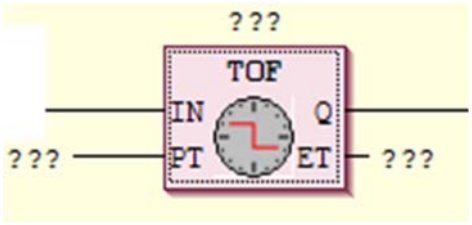


Para que el temporizador vuelva a 0 solo tenemos que poner a 0 la entrada “IN” en nuestro caso el “MARCHA_1”. Volvemos clicar sobre “Valor preparado”.

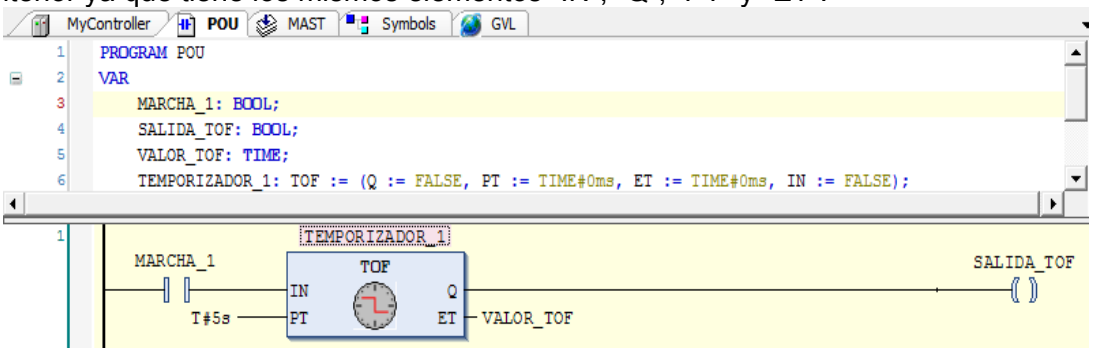
Aparece “FALSE” y apretamos **“CTRL+F7”**



3.2.2 TOF Temporizador de retardo a la desconexión.



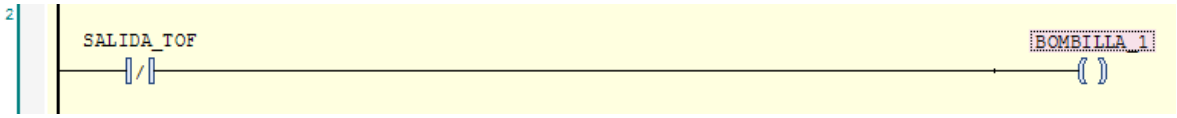
Si nos fijamos la forma de rellenar los datos de este temporizador será igual que el anterior ya que tiene los mismos elementos “IN”, “Q”, “PT” y “ET”.



El funcionamiento de este temporizador es el siguiente:

Estará a 0 mientras en “IN” haya una entrada booleana a 0. Si ponemos a 1 la entrada el temporizador se pondrá a 1 inmediatamente. Permaneciendo así hasta que volvamos a poner a 0 la entrada, comenzando así a contar la base de tiempo prefijada en “PT”, transcurrido el mismo volverá a ponerse el temporizador a 0.

Vamos a ampliar el programa para que quede más claro el funcionamiento. Añadiremos un segmento que contenga la salida del temporizador negada y la llevaremos a una salida, quiere decir, que la salida está a 1 si el temporizador está a 0.

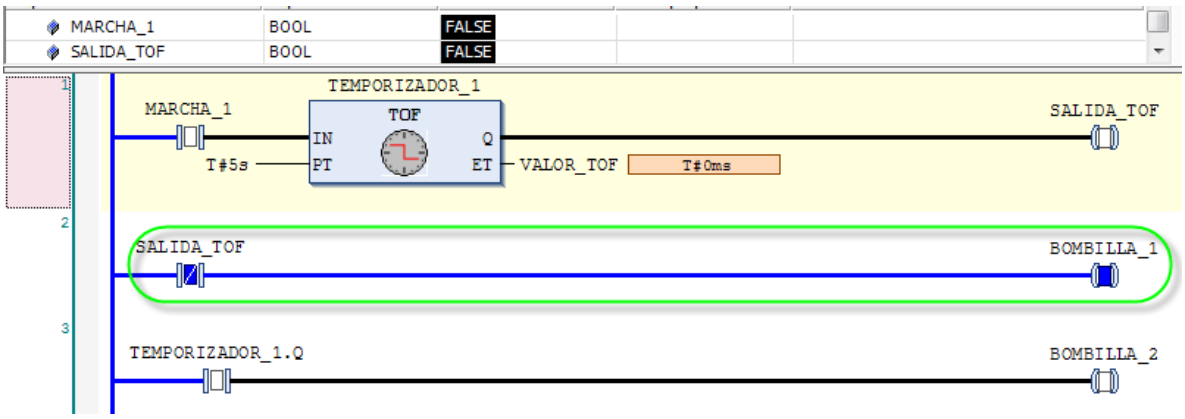


Si observamos estamos utilizando el contacto negado de la Bobina que asociamos a la salida del temporizador. Esta sería una posible forma de gestionar la salida.

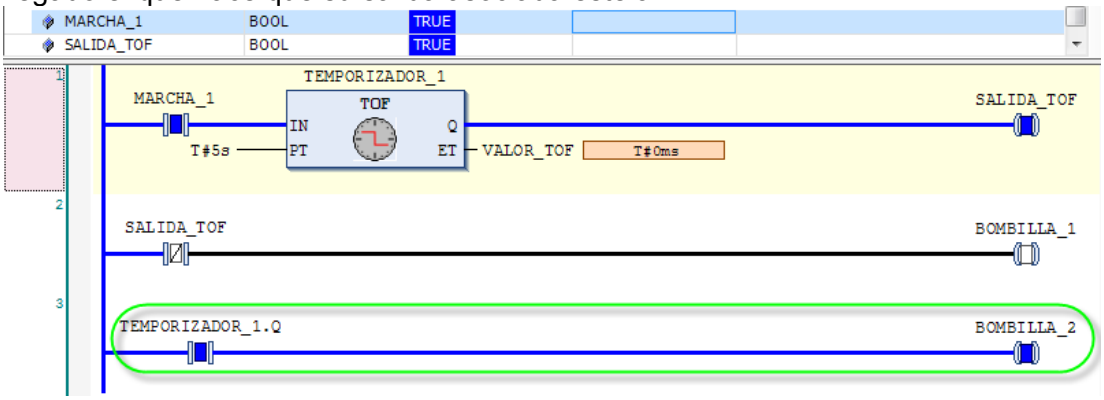
En el siguiente caso vamos a ver como utilizar la salida propia del temporizador.
Si la salida del temporizador esta a 1 entonces la BOMBILLA_2 estará a1.

Observamos que hemos puesto el nombre del temporizador + .Q

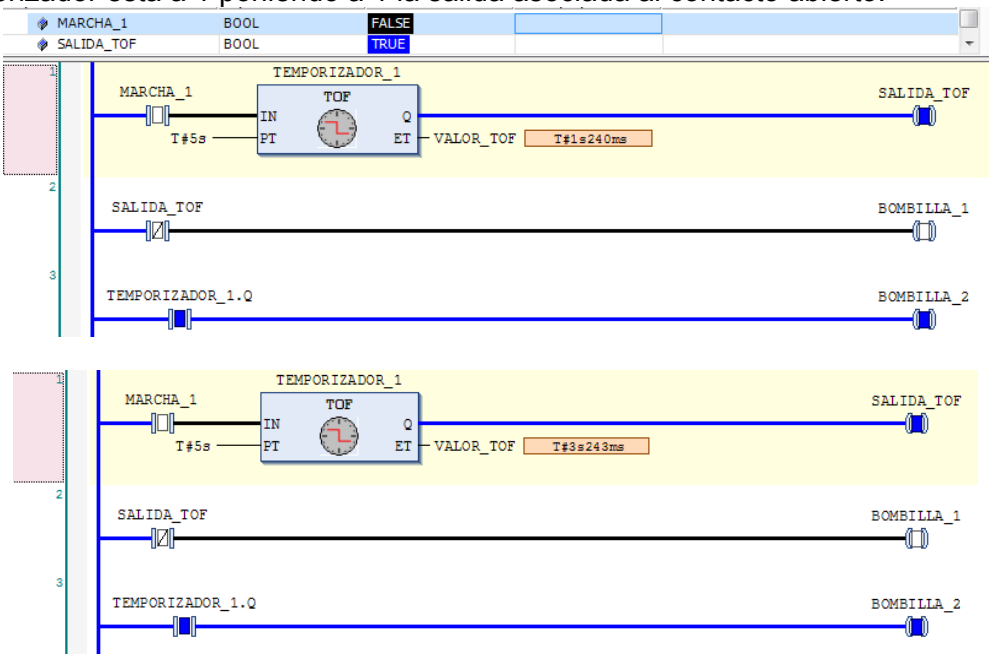
A continuación vemos la simulación.



La entrada está a 0 por lo que la salida del temporizador está a 0 y es el contacto negado el que hace que su salida asociada esté a 1.

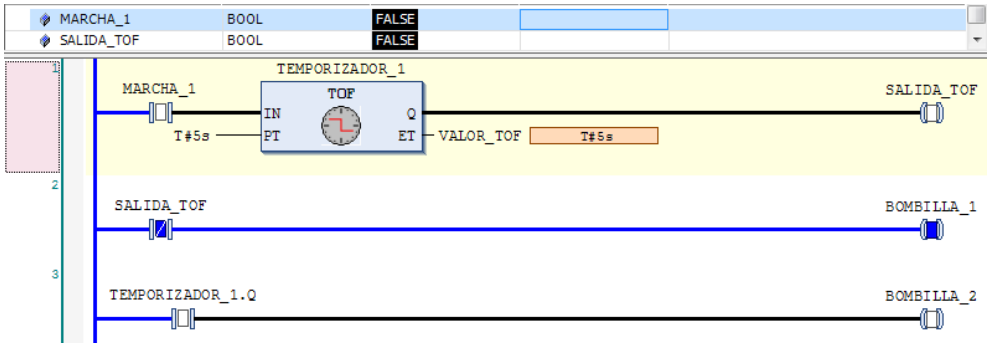


Hemos cambiado el estado de la salida a 1 a consecuencia de esto la salida del temporizador está a 1 poniendo a 1 la salida asociada al contacto abierto.

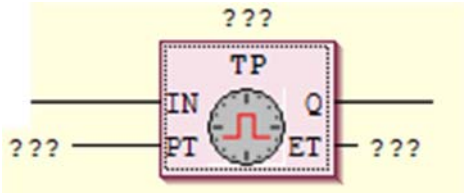


Volvemos a cambiar la entrada a 0 y vemos como el estado del temporizador sigue siendo 1, pero en “ET” ha comenzado a contar el tiempo prefijado en “PT”.

Cuando cuenta 5 s vuelve al estado inicial.



3.2.3 TP Temporizador de pulso

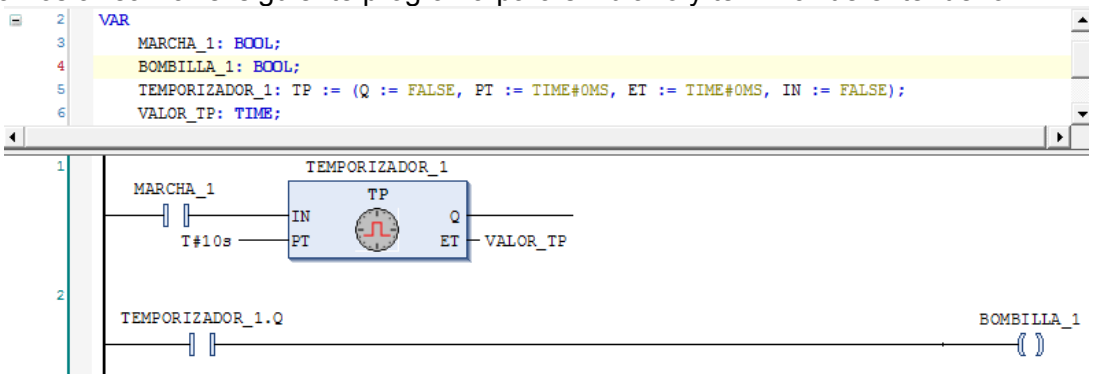


Este temporizador lo encontramos en la barra de navegación de la ventana de programación.

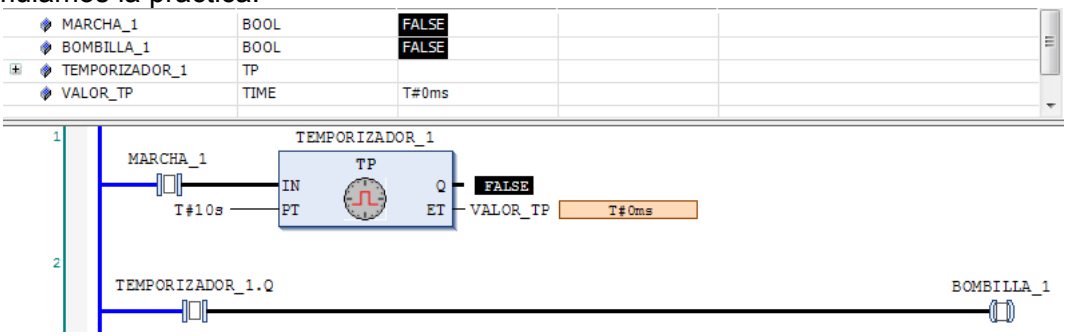
El funcionamiento es el que sigue.

Al poner la entrada “IN” a 1 se mantenga o no la misma a 1 el temporizador se encontrará a 1 el tiempo prefijado en “PT”.

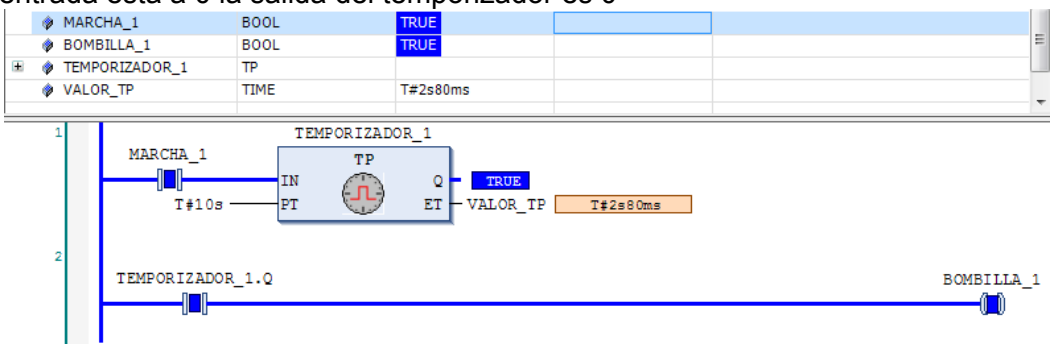
Vamos a realizar el siguiente programa para simularlo y terminar de entenderlo.



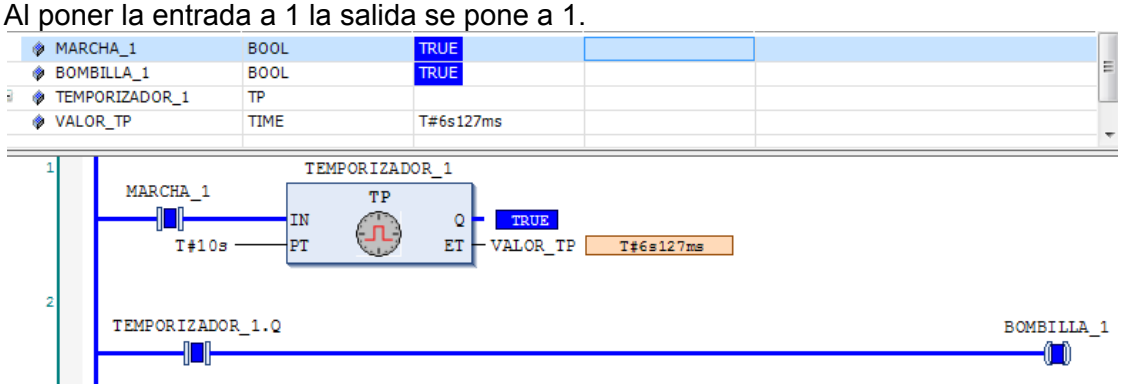
Simulamos la práctica.

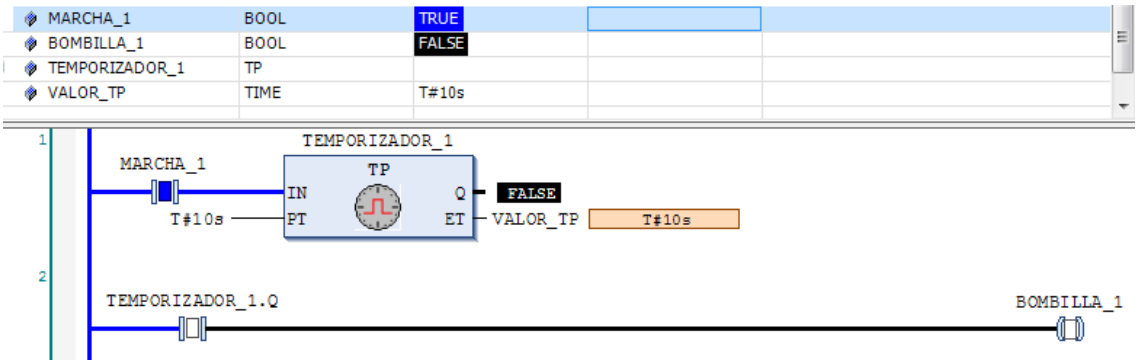


La entrada está a 0 la salida del temporizador es 0



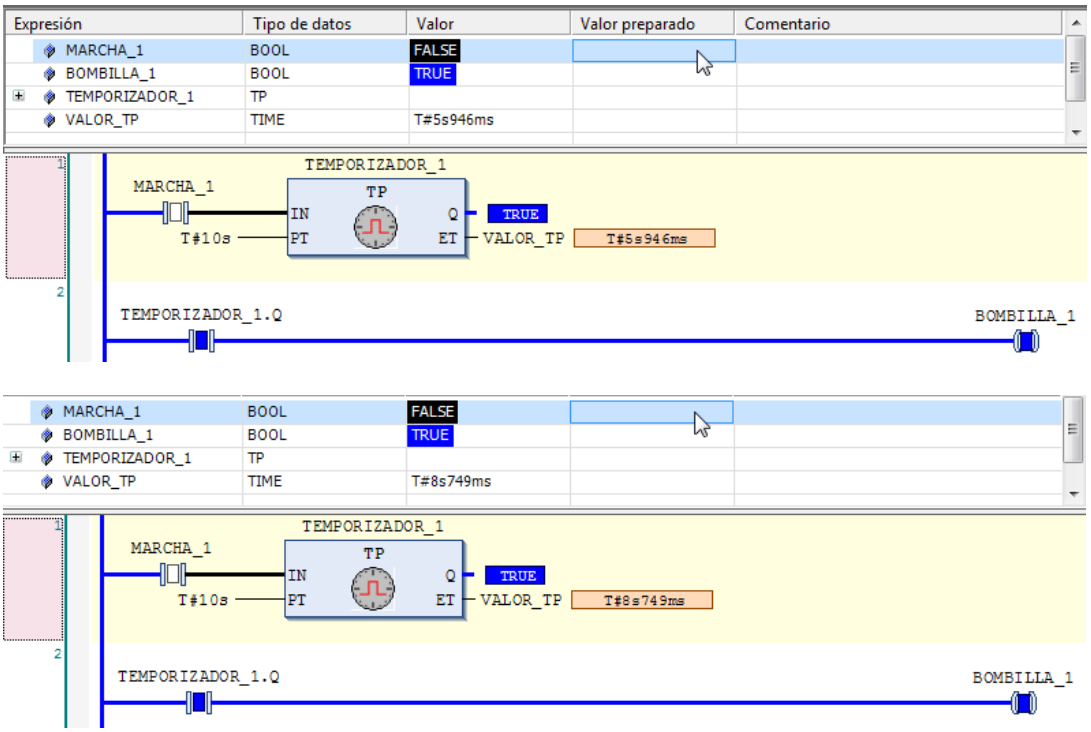
Al poner la entrada a 1 la salida se pone a 1.





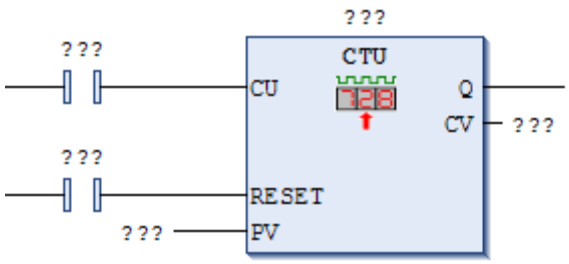
Si la entrada sigue a 1 se mantiene a 1, pero al llegar a los 10s prefijados cambia de estado, pasando a 0, aunque siga la entrada a 1.

Si ponemos la entrada a 0 durante el contaje termina de contar igualmente.



3.3 CONTADORES

3.3.1 CTU Contador ascendente (Up)



Este módulo nos permite contar pulsos ascendentes y guardar el resultado en una palabra, además es comparador ya que le podemos prefijar un valor concreto y el contador cambiará de estado, pasará de 0 a 1, al alcanzar este valor.

El contador seguirá contado de forma ascendente superando el valor prefijado, para empezar una nueva cuenta deberemos “resetear” el contador.

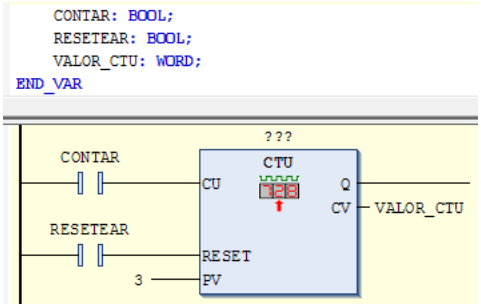
Cada vez que pongamos a 1 la entrada “CU” el contador sumará 1 al valor interno que tenga.

Si la entrada Reset se pone a 1 el valor del contador pasará a ser 1.

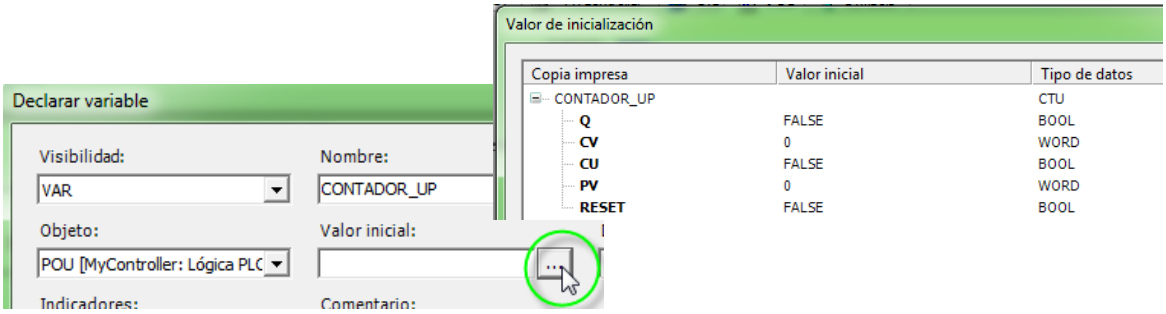
En “PV” pondremos el valor al que el contador se pondrá a 1.

En “CV” veremos el valor interno del contador.

Vamos a rellenar los valores del contador.



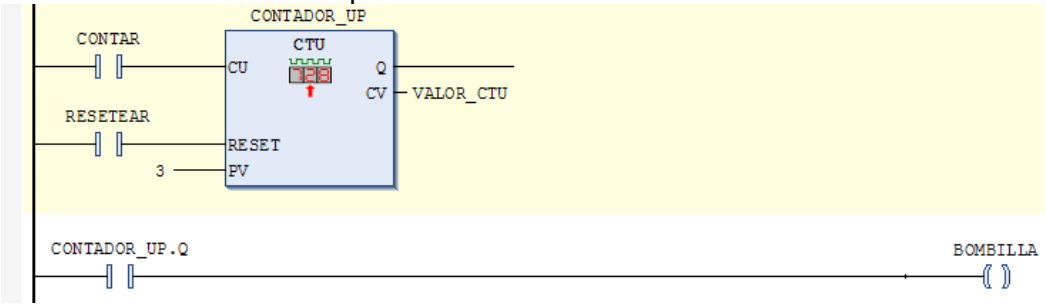
Tendremos que darle un nombre al CTU y al igual que en los temporizadores deberemos definir el valor inicial.



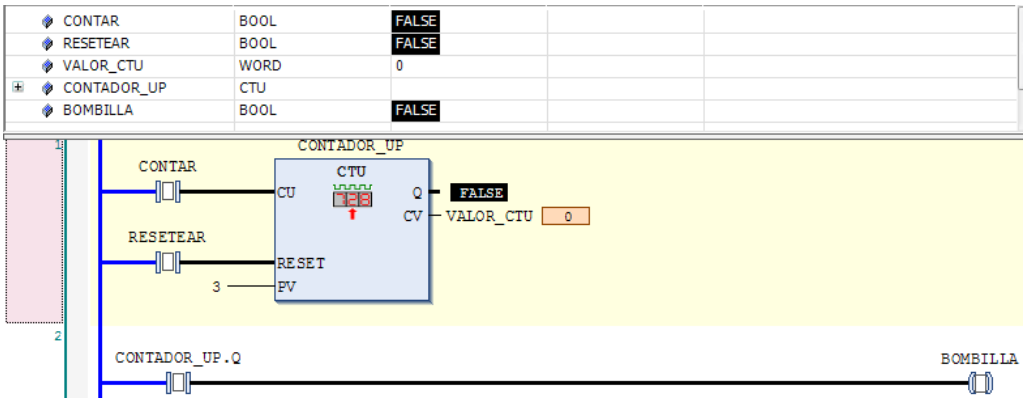
En la declaración de variable, las variables quedarán de la siguiente manera.

```
PROGRAM POU
VAR
    CONTAR: BOOL;
    RESETEAR: BOOL;
    VALOR_CTU: WORD;
    CONTADOR_UP: CTU := (Q := FALSE, CV := 0, CU := FALSE, PV := 0, RESET := FALSE);
END_VAR
```

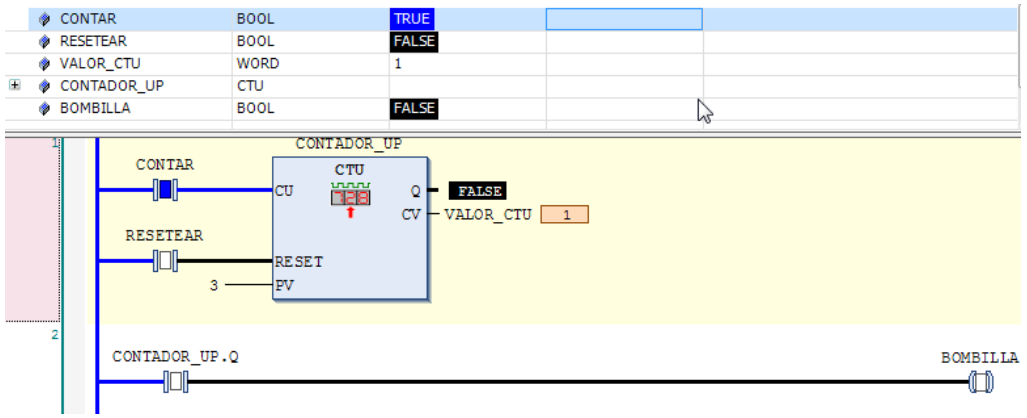
Vamos a preparar el siguiente programa donde al alcanzar el valor prefijado en PV se activará la salida del contador poniendo a 1 la bobina salida.



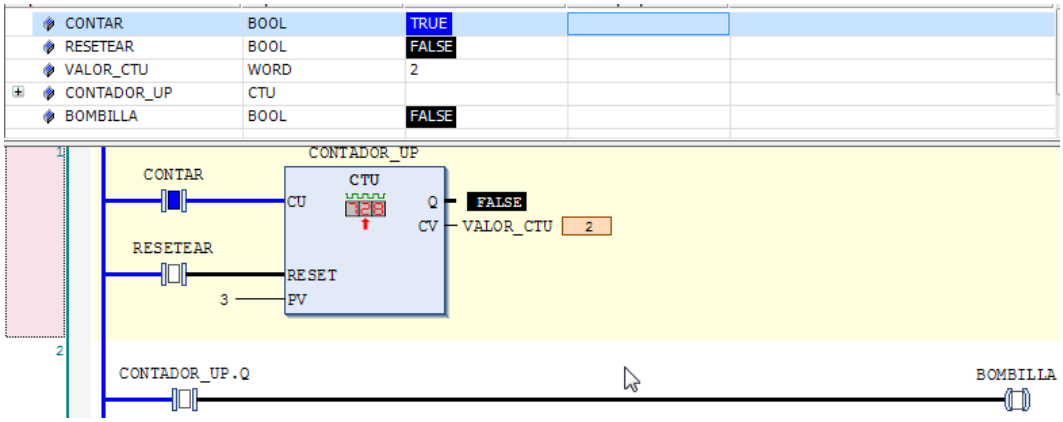
Simulamos la práctica.



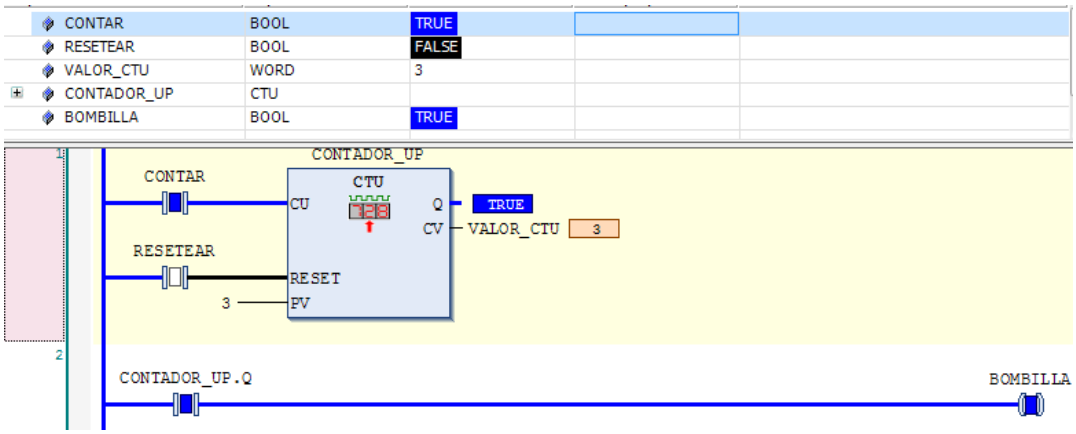
Vemos como el valor de contador es 0, por lo que la salida es 0.



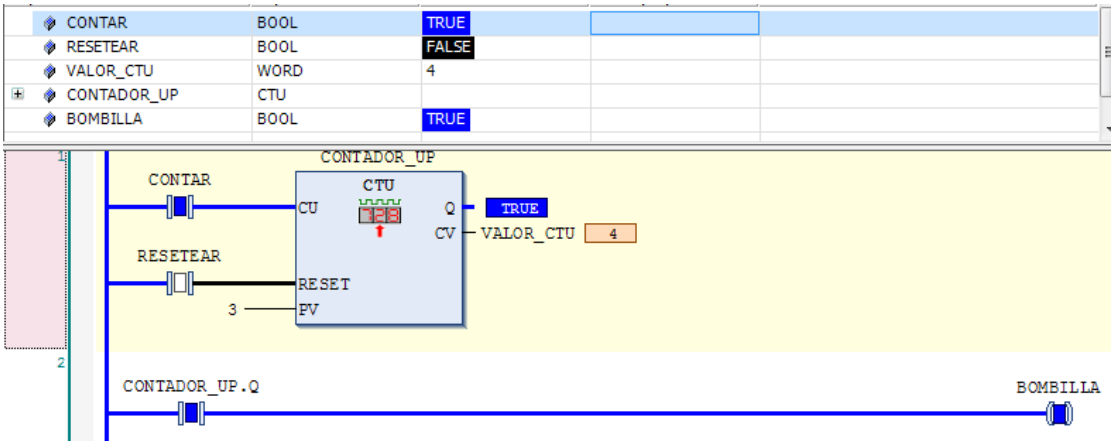
Al poner la variable “CONTAR” a 1 el valor de contador pasa a ser 1 pero como no es igual al valor que hay en “PV”, la salida “Q” sigue a 0.



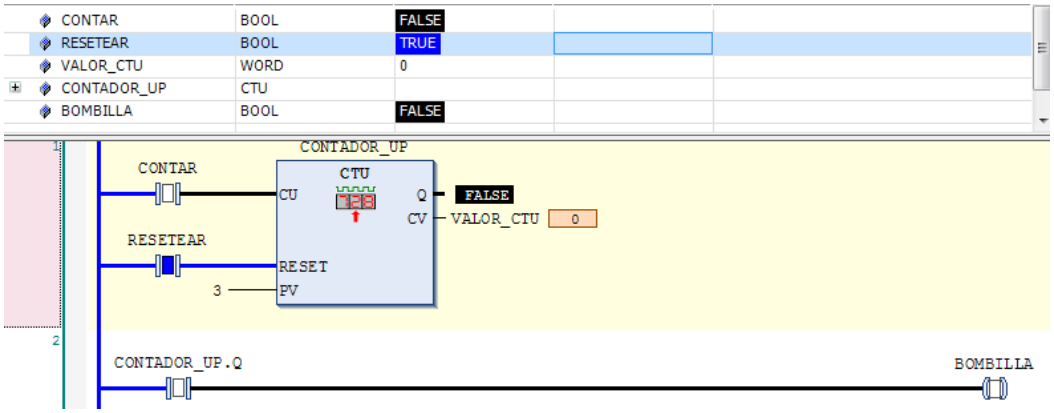
Volvemos a poner la entrada “CU” a 1 y vuelve a sumar 1 en el valor anterio con lo que pasa a ser 2. Como no es igual a 3, el valor que hay en “PV”, la salida “Q” sigue a 0.



Al llegar el valor a 3, se activa la salida “Q” haciendo que se ponga a 1 la bobina “BOMBILLA”

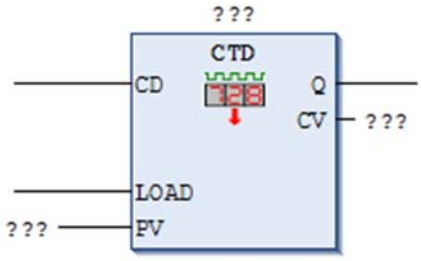


Si seguimos activando la entrada “CTU” el contador sigue aumentando su valor, y la salida “Q” sigue a uno, esto significa que el contador compara el valor “PV” a mayor o igual.



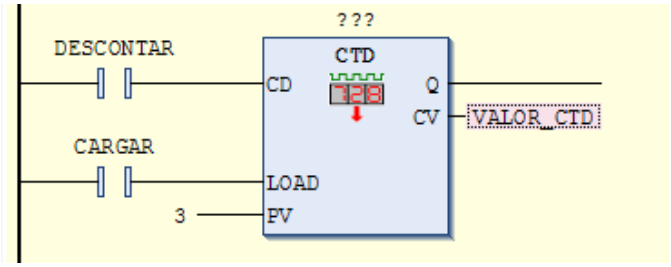
Al poner la entrada “RESET” a 1 el contador vuelve a tener un valor de 0.

3.3.2 CTD Contador descendente (Down)

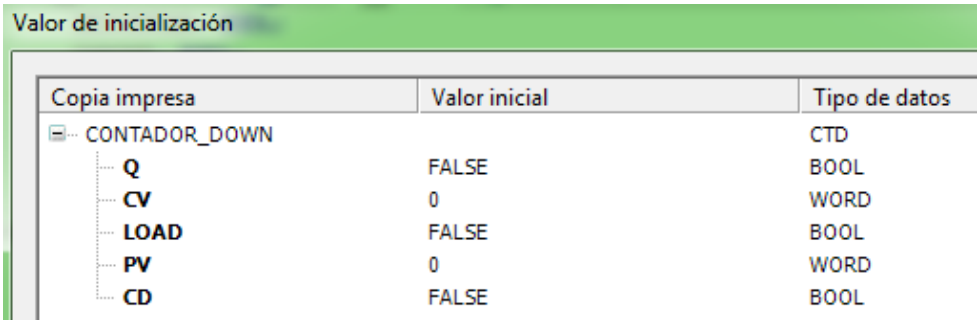


Este contador cuenta de forma descendente cada vez que recibe un pulso en la entrada “CD”. La salida del contador estará a 0 mientras el valor de CV sea diferente de 0 y se pondrá a 1 cuando el valor sea 0. Mediante la “Entrada_2” cargaremos el valor prefijado en “PV”. En “CV” veremos el valor interno del contador.

Vemos como rellenamos el contador.

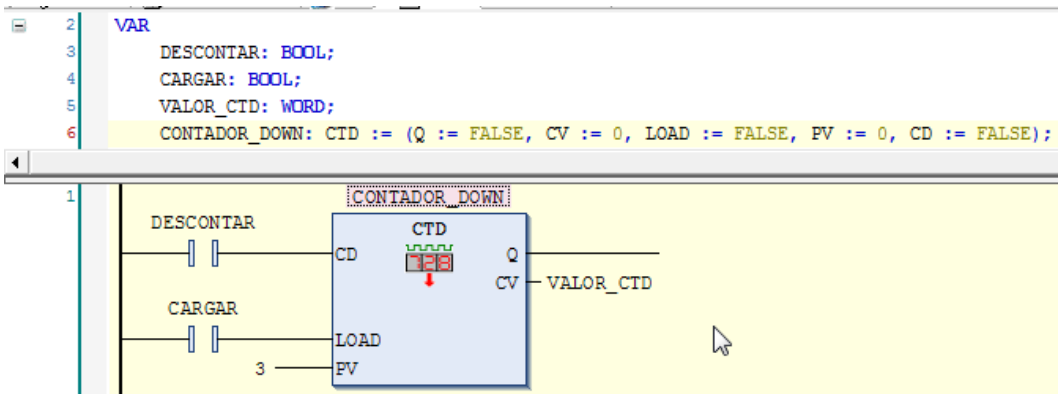


También tendremos que dar un nombre al contador y dar el Valor inicial como en los elementos anteriores.

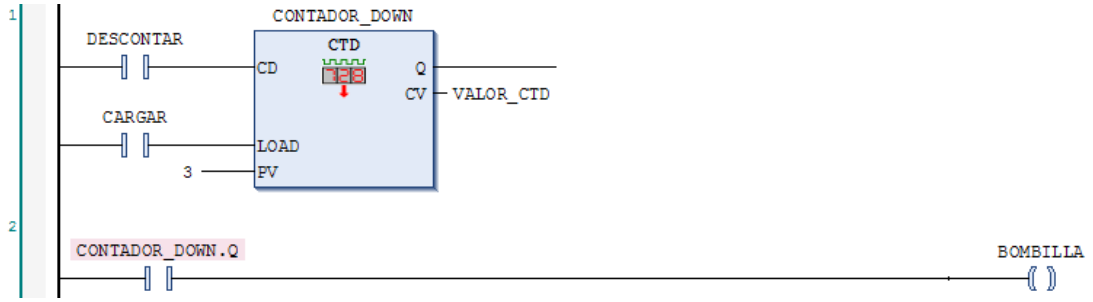

Valor de inicialización

Copia impresa	Valor inicial	Tipo de datos
CONTADOR_DOWN		CTD
Q	FALSE	BOOL
CV	0	WORD
LOAD	FALSE	BOOL
PV	0	WORD
CD	FALSE	BOOL

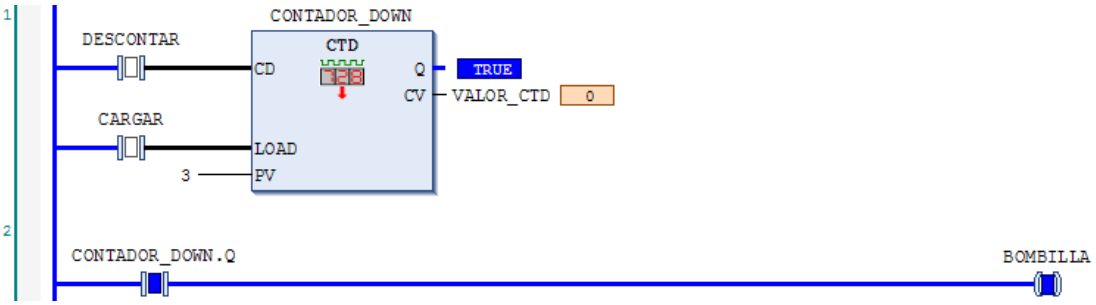
Las variables quedarian de la siguiente manera.



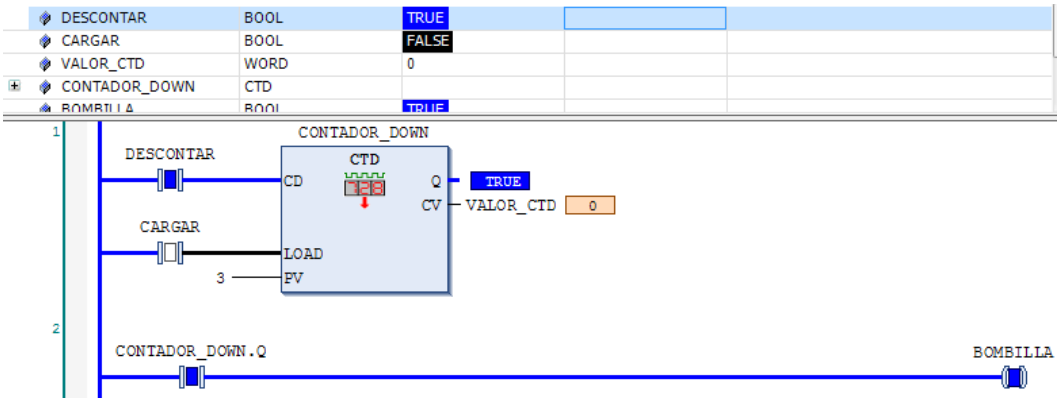
Vamos a simular el programa siguiente.



Como vemos si la salida del contador está a 1 la bobina “BOMBILLA” estará a 1.

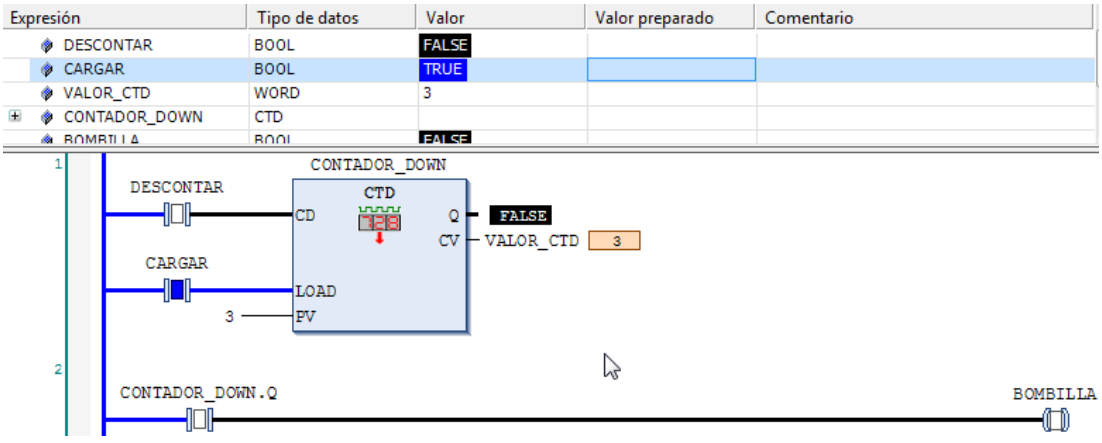


La salida “BOMBILLA” está a 1 por que el contador está a 0.

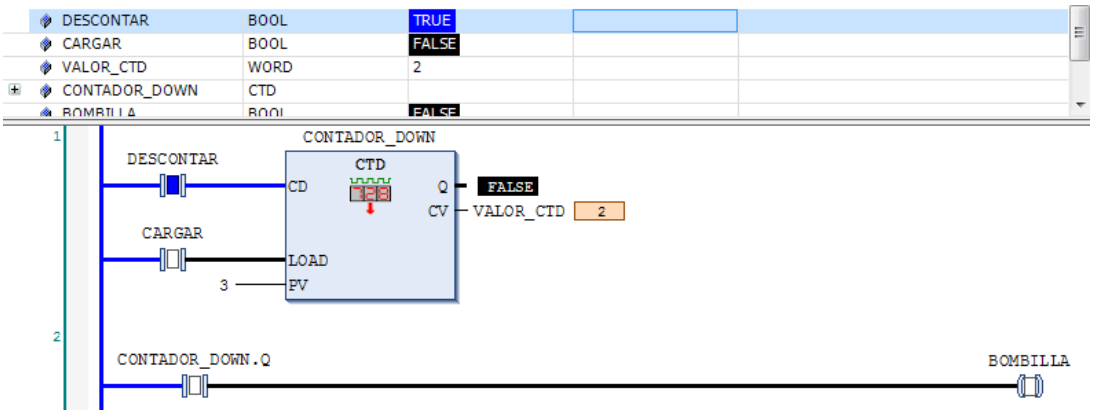


Si ponemos la entrada CD a 1 vemos como el contador sigue a 0, por lo que deducimos que el contador no cuenta valores negativos.

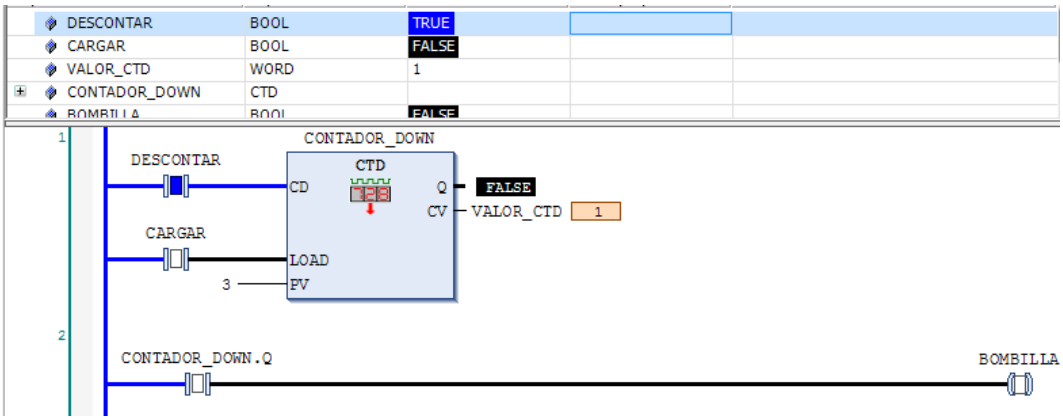
Lo primero que deberemos hacer será cargar el valor que hay en “PV”, poniendo a 1 la entrada “LOAD”.



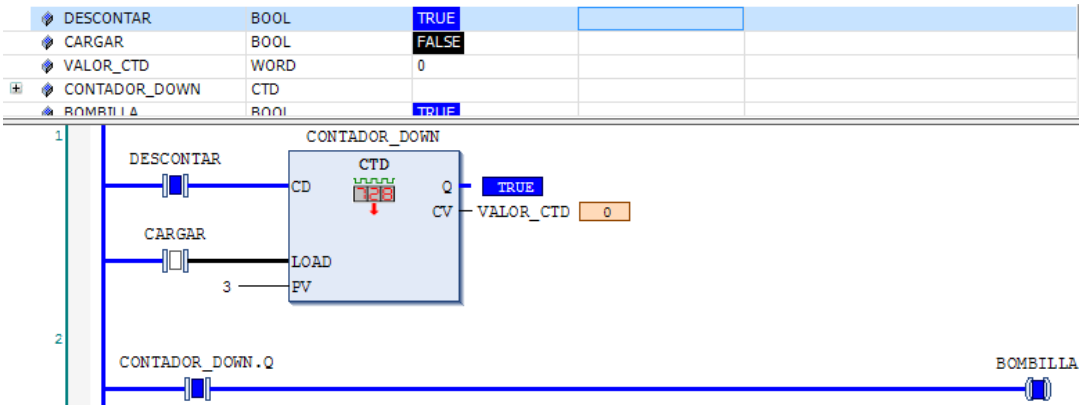
Observamos que la salida “Q” del contador ha pasado a ser 0, poniendo a 0 la bobina “BOMBILLA”.



Al poner la entrada “CD” a 1, ahora sí vemos como descuenta 1 en el valor que había antes en “CV”. 3-2=1.



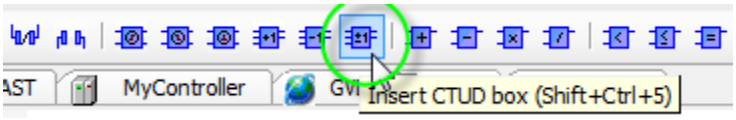
Vemos como el valor sigue disminuyendo al volver a poner a 1 la entrada “CD”.



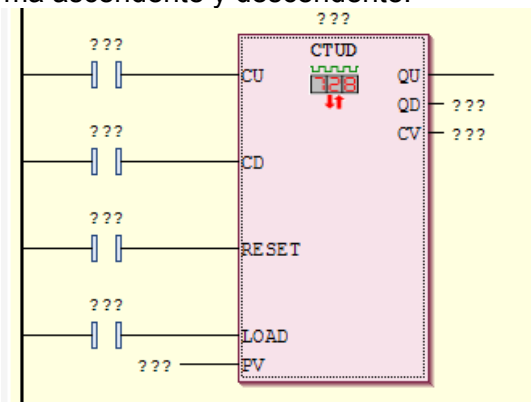
Como vemos al llegar a 0 se vuelve a activar la salida “Q”, poniendo a 1 la salida “BOMBILLA”,

3.3.3 CTUD Contador UP/DOWN

Este contador se encuentra en la barra de navegación de la pantalla de programación.



Sirve para contar de forma ascendente y descendente.



Como podemos observar, es la mezcla de los dos anteriores y funciona de la misma manera.

Si ponemos a 1 la entrada “CU” contará de forma ascendente. Si alcanza el valor prefijado en “PV” se activará la salida “QU”.

Si ponemos a 1 la entrada “CD” contará de forma descendente. Si el contador llega a 0 se activará la salida “QD”.

Al poner la entrada “RESET” a 1 se cargará 0 en el valor del contador.

Al poner la entrada “LOAD” a 1 se cargará el valor que haya en “PV” en el “CV” del contador.

Si el valor está, entre los valores 0 y el prefijado en “PV” no tendrá ninguna de las dos salidas activadas.

3.4 OTRAS INSTRUCCIONES

3.4.1 Move

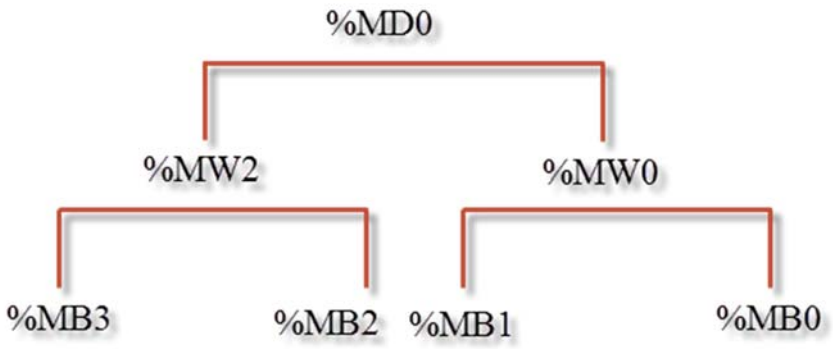
Antes de explicar módulo, vamos a comentar los diferentes formatos con los que trabaja “SoMachine”. Los formatos tenemos que pensar que sirven para guardar valores, por lo que deberemos escoger dependiendo de la necesidad.

Por ejemplo, si queremos guardar el valor 265 no podremos coger un BYTE, sino que tendremos que utilizar un WORD, un INT, etc.

Otra cosa a tener en cuenta es al dar direccionamiento a los formatos que tenemos.

Al trabajar con 8 bits utilizaremos el direccionamiento B.
Al trabajar con 16 bits utilizaremos el direccionamiento W.
Al trabajar con 32 bits utilizaremos el direccionamiento D.

Si ponemos la dirección %MD0, 2 palabras que serán %MW0 y %MW2, y cada una de estas contiene dos BYTE, el %MB0, %MB1, %MB2 Y %MB3.

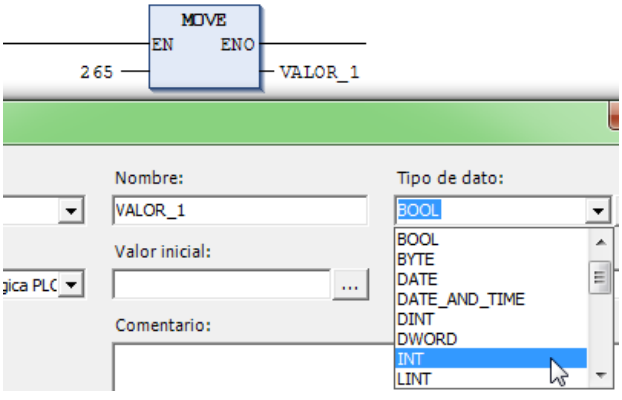


Deberemos tener cuidado no solapemos unas variables con otras.

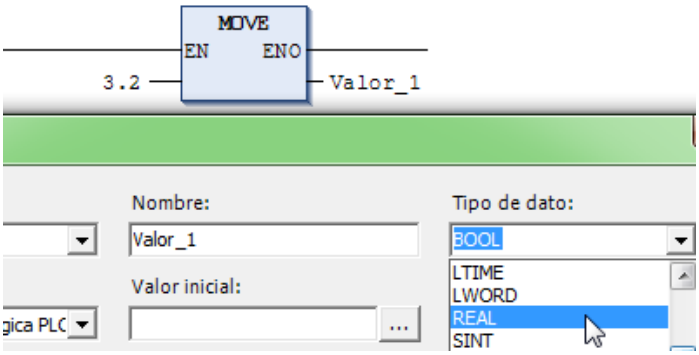
Keyword	Prefix ***	Size (Bit)	Range	Example
BOOL	x	8	0 , 1	FALSE, TRUE, 0, 1
SINT **INT** **DINT** **LINT**	si i di li	8 16 32 64	-128 .. 127 -32.768 .. 32.767 -2.147.483.648 .. 2.147.483.647 -2^{63} .. 2^{63}-1	0 24453 -38099887
USINT **UINT** **UDINT** **ULINT**	usi ui udi uli	8 16 32 64	0 .. 255 0 .. 65.535 0 .. 4.294.967.295 0 .. 2^{64}	200 47453 138099887
BYTE **WORD** **DWORD** **LWORD**	by w dw lw	8 16 32 64	0 .. 255 0 .. 65.535 0 .. 4294967295 0 .. 2^{64}-1	8450, 16#EA3F, 2#11_0011_1010
REAL	r	32	-1.2x 10^{-38} .. 3.4x 10^{38}	1.34996
TIME **LTIME*** **TOD** **DATE** **DT**	tim ltim tod date dt	32 64 32 32 64	0 ms .. 1193h2m47s295ms 0 ns..213503d23h34m33s709ms551us 615ns 00:00:00 .. 23:59:59 01.01.1970 bis ca. 06.02.2106	T#1d8h12m8s125ms LTIME#10d12h13ns TOD#12:34:17 D#2001-03-15 DT#2001-03-15-12:17:03
STRING(xx)** **WSTRING(xx)****	s ws		0 .. 255 Characters 0 .. 32767 Characters	`Hello world`

Ahora si explicaremos el módulo MOVE sirve para cargar un valor concreto dentro de un formato de los anteriormente vistos. Como por ejemplo.

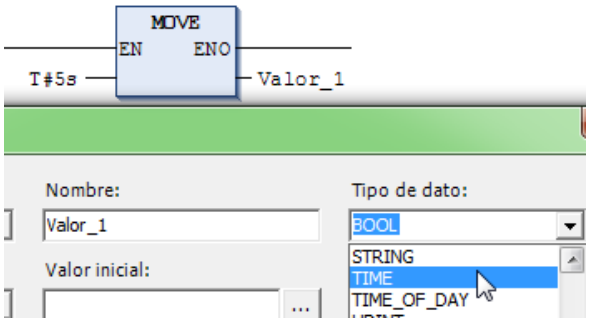
El 256 en un INT.



El 3,2 en un Real.

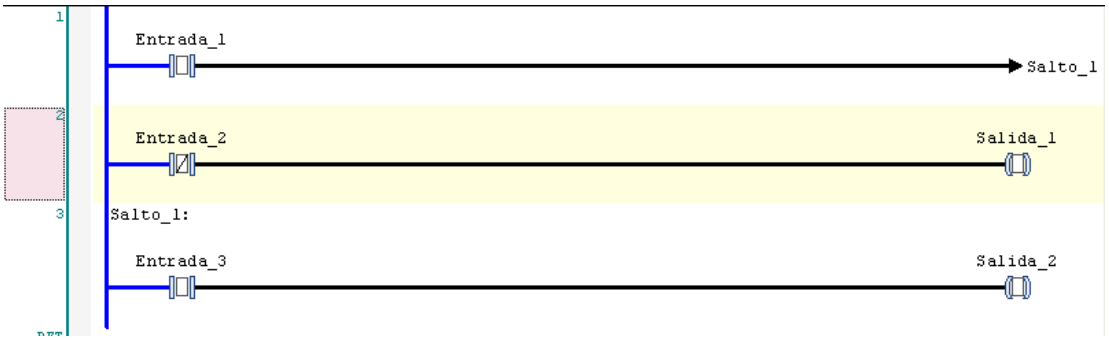


5s en un TIME.



3.4.2 Salto

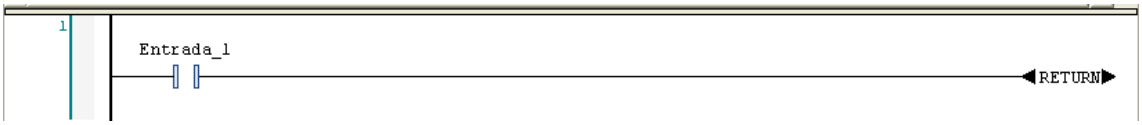
Utilizamos esta instrucción para saltar líneas de programa, en según qué condiciones, para que no sean ejecutadas.



Siempre que no tengamos a 1 la “Entrada_1” el programa ejecutará la línea 2. Si en algún momento ponemos a 1 la “Entrada_1” la línea 2 no será ejecutada por el programa saltando directamente a la línea 3.

3.4.3 Return

Esta función se utiliza para volver de un sub-programa, la estudiaremos más adelante.



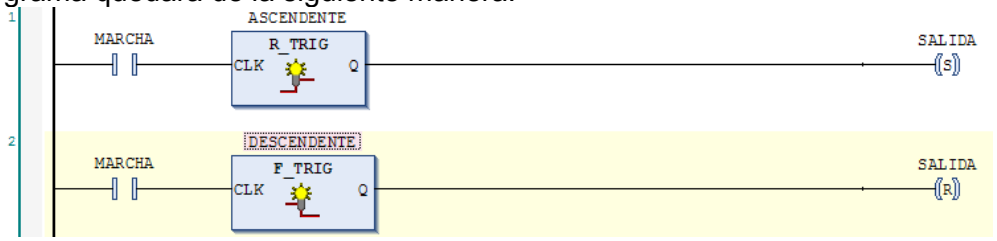
3.5 MÓDULOS DE FUNCIÓN

3.5.1 R_Trig y F_Trig

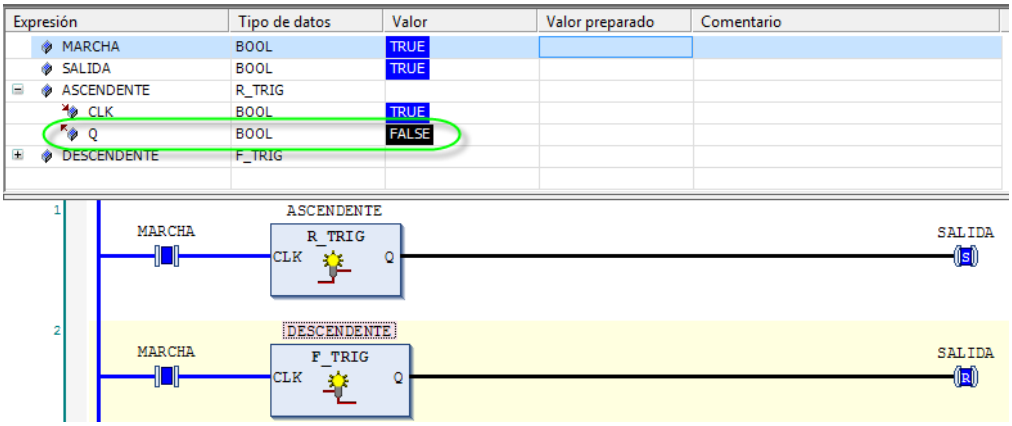
Estos módulos trabajan por flancos, los flancos son las transiciones de un bit en su paso de 0 a 1 (flanco ascendente) o de 1 a 0 (flanco descendente).

Estos módulos al detectar el flanco mandan ponen su salida a 1 el tiempo que dure la ejecución de programa.

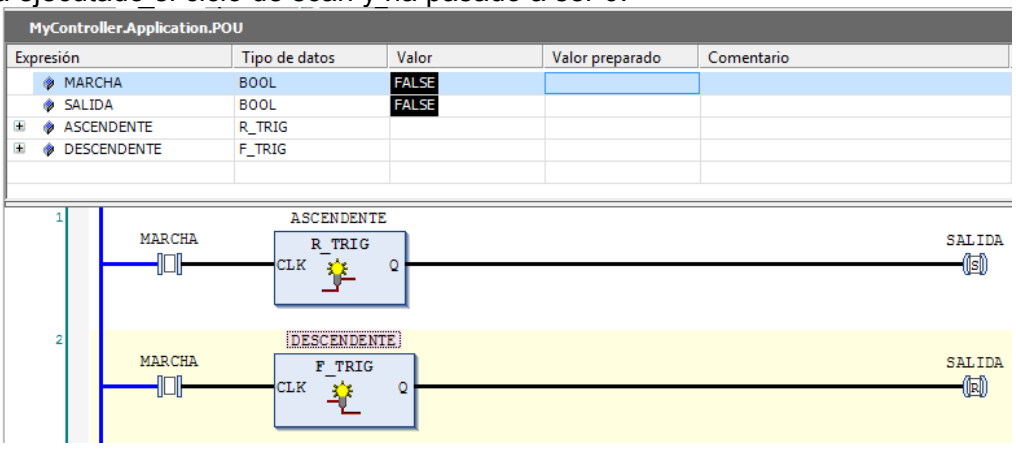
Para entender el mismo hemos preparado un ejemplo en el que en el flanco ascendente (R_Trig) nos hará set sobre una salida y en descendente (F_TRIG) nos hará un reset sobre la misma salida.
El programa quedará de la siguiente manera.



Vamos a la simulación



Observamos como al poner la entrada “MARCHA” a 1 se pone a 1 la bobina “SALIDA”, pero también podemos observar la salida “Q” del “R_TRIG” esta a 0, esto es porque ya se ha ejecutado el ciclo de scan y ha pasado a ser 0.

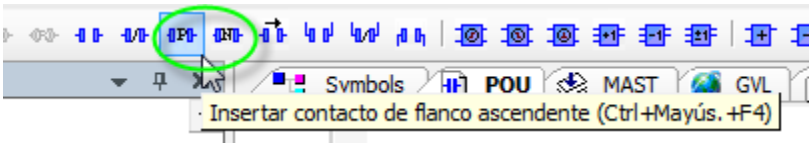


Al volver a poner la variable “MARCHA” a 0 se resetea la variable “SALIDA”.

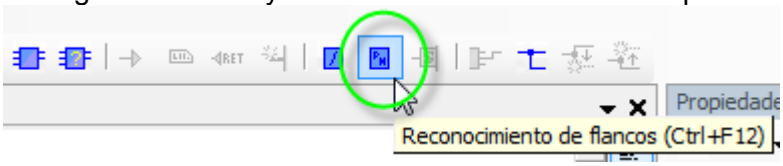
3.5.2 Flancos

El software nos permite también trabajar los flancos directamente en los contactos abiertos sin tener que utilizar los módulos R_TRIG y F_TRIG.

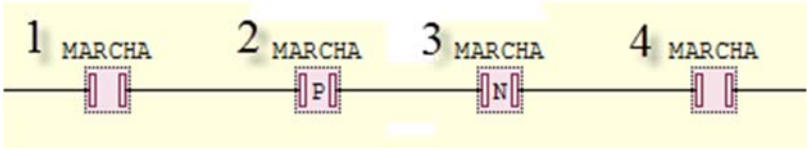
En la barra de navegación tenemos el contacto flanco ascendente y flanco descendente.



En esta misma barra de navegación tenemos un icono que nos permite insertar el flenco positivo o negativo cuando ya tenemos un contacto normal puesto.

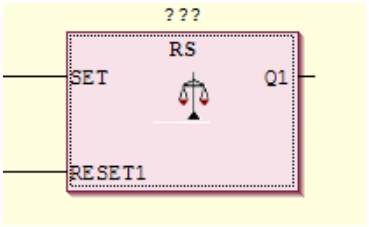


Si tenemos un contacto abierto marcado y vamos clicando en este icono nos irá apareciendo “P”, “R”, Desaparece y vuelta a empezar.

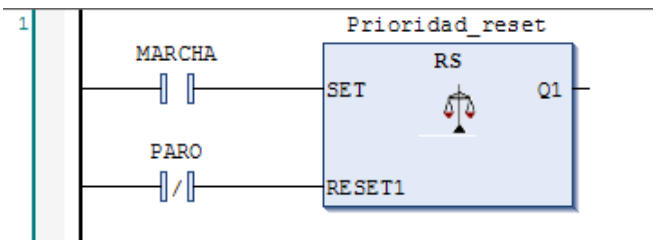


3.5.3 RS Prioridad reset

Este módulo sirve para establecer una báscula (flip-flop) con prioridad al Reset.

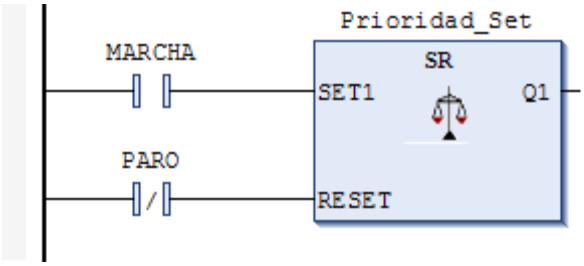


Si damos a la “Entrada_1” haremos un set a la báscula poniendo a 1 su salida, si damos a la “Entrada_2” desactivaremos la báscula poniendo su salida a 0. En caso de tener las dos entradas activadas la salida estará a 0.



3.5.4 SR Prioridad Set

El funcionamiento es el mismo que el anterior pero con prioridad al Set. Si se activan las dos entradas a la vez, la salida de la báscula estará a 1.

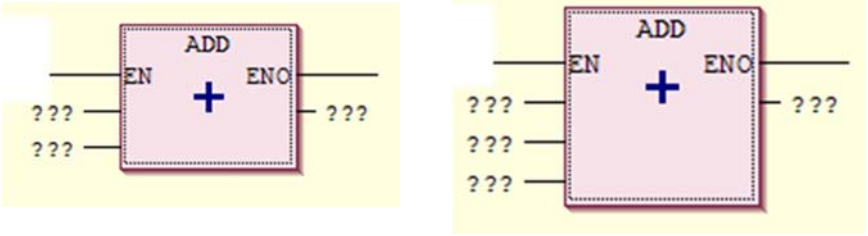


3.6 OPERADORES MATEMÁTICOS

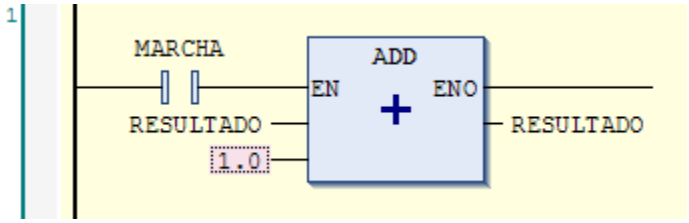
Los operadores matemáticos son módulos que nos permiten realizar operaciones aritméticas, como sumas, restas, multiplicaciones, etc. Y comparaciones a Igualdad, diferente, etc.

3.6.1 Add – Suma (2 y 3 entradas)

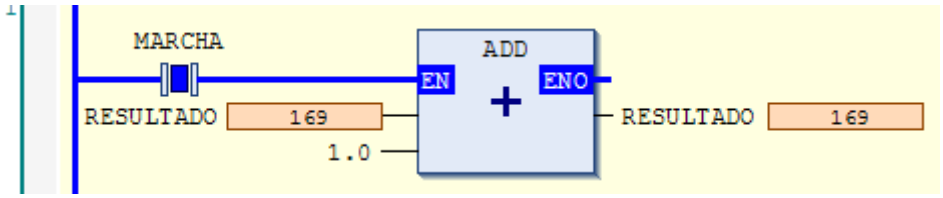
Este módulo permite sumar 2 o 3 valores diferentes y guardar el resultado en una zona de memoria.

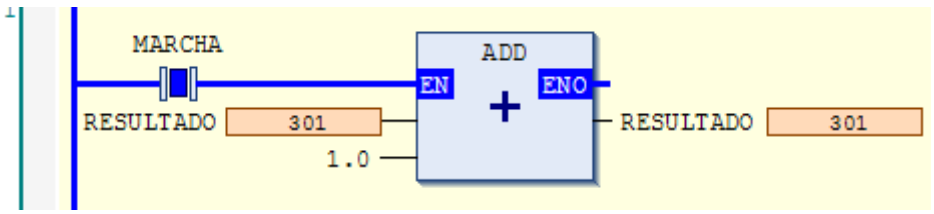


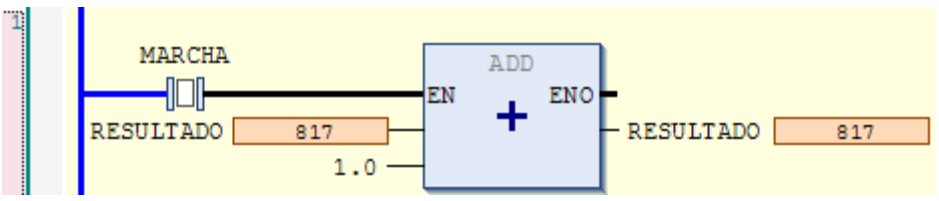
Generalmente las operaciones no las vamos a realizar con valores fijos solo, porqué el resultado ya lo sabríamos, lo haremos como mínimo con valor variable. Un ejemplo sería crear un contador, mediante una suma. Realizaremos el siguiente programa.



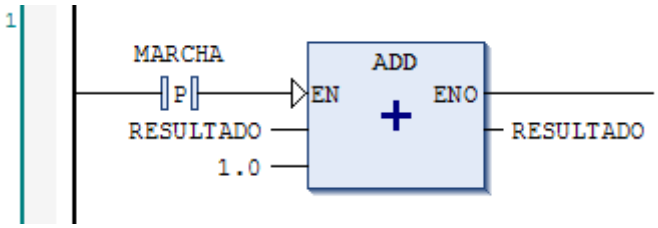
Si nos fijamos, lo que estamos haciendo es sumar un valor variable “RESULTADO”, que empezará en 0.0 + 1.0 y el resultado lo cargamos sobre la misma variable. Vamos a ver la simulación.



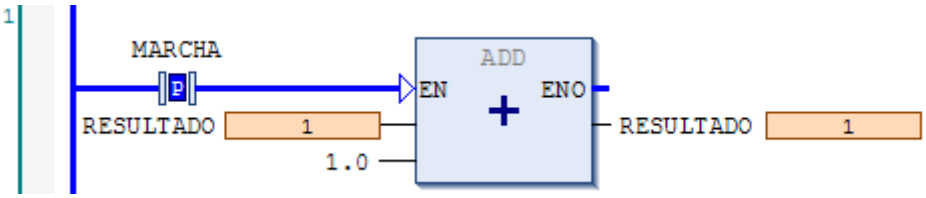




Observamos que el valor aumenta muy rápido, esto es porqué la suma se ejecuta cada ciclo de ejecución que su entrada esté en 1. Por lo que en este caso deberíamos trabajar con un flanco para suma 1 vez cada vez que pongamos la entrada a 1.

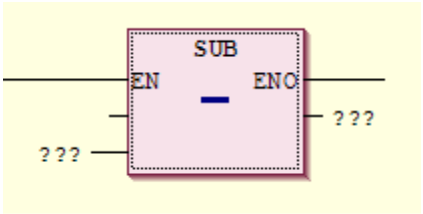


Ahora observamos cómo aunque la entrada permanece en 1 solo nos ha contado una vez.

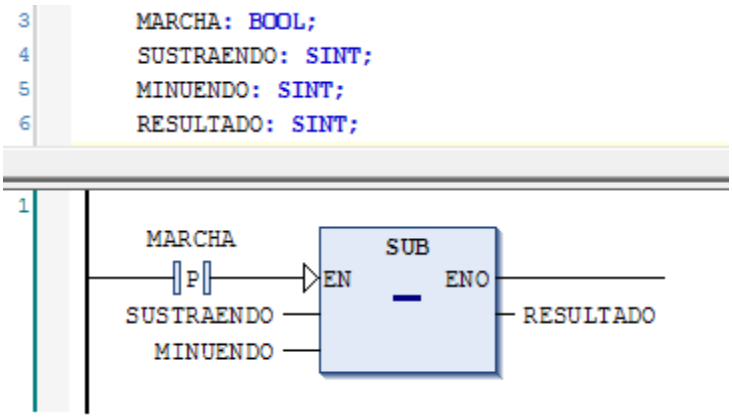


3.6.2 SUB - Resta

Módulo que nos permite realizar la resta de 2 valores diferentes. Podríamos crear un contador decremental con él.



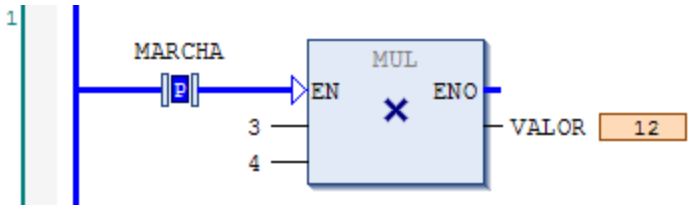
El valor superior de la izquierda será el sustraendo, y el de abajo el minuendo, a la derecha nos quedará el resultado.



Si queremos poder trabajar con enteros negativos debemos declarar la variable”Sint” y si queremos guardar el valor en una palabra deberá ser del tipo “%MW0”.

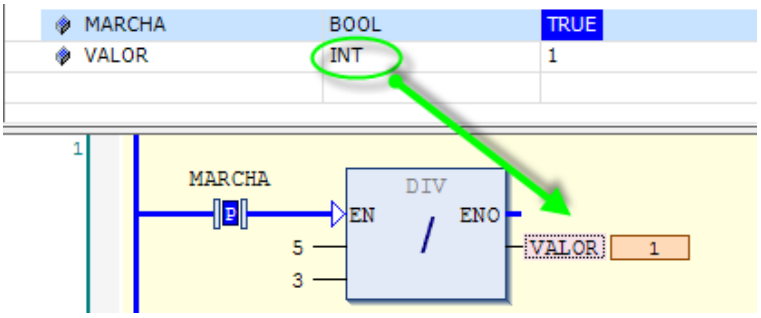
3.6.3 MUL - Multiplicación

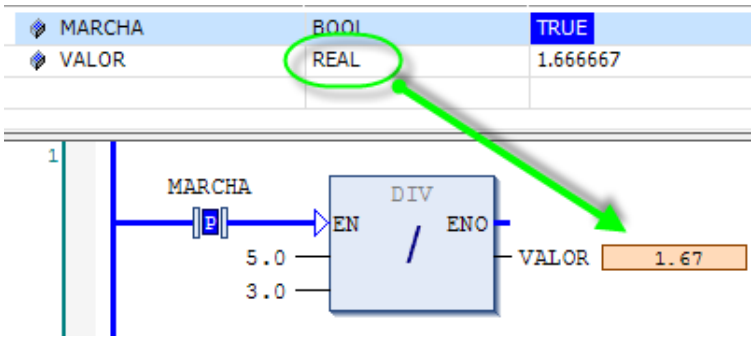
Realiza una multiplicación de dos factores. Funciona de la misma manera que los anteriores.



3.6.4 DIV - División

Mediante este módulo podemos realizar una división de dos valores. Esta división para que nos dé un valor con decimales se debe introducir con valores reales, sino el valor que nos da sería el número entero sin decimales, dando como resultado el valor entero.





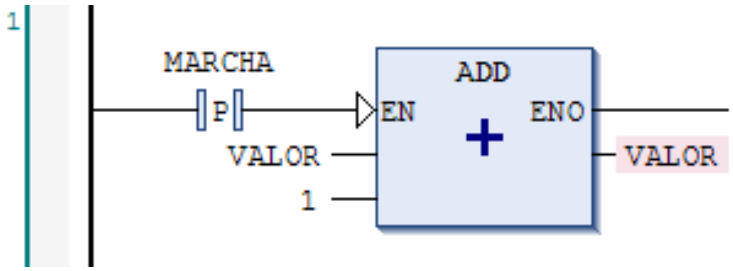
3.6.5 EQ; NE; LT; LQ; GT; GQ - Comparaciones

Estos módulos se utilizan para comparar valores. EQ es para comparar a igualdad, NE diferente de, LT menor que, LQ menor o igual, GT mayor que, GQ mayor o igual que.

Cuando hablamos de los contadores, ya comentamos que comparaban, el “DOWN” o decreciente a igualdad a 0 y el “UP” o creciente, igual o mayor al valor prefijado en el “PV”.

Vamos a crear un ejemplo en el que mediante una operación de suma vamos a incrementar un valor de 1 en 1 y compararemos este valor con algunos de estos comparadores para ver qué salida se activa con según qué condiciones.

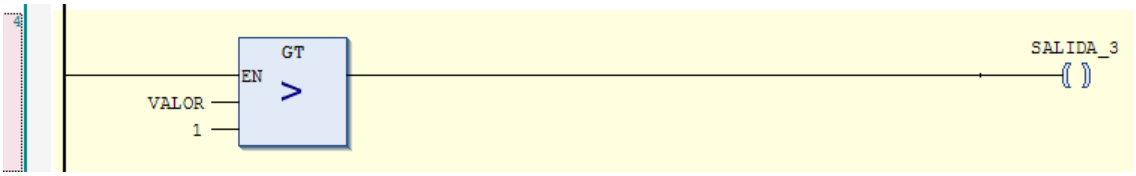
Hemos preparado una suma que añadirá 1 a “Valor” cada vez que le demos un pulso ascendente.



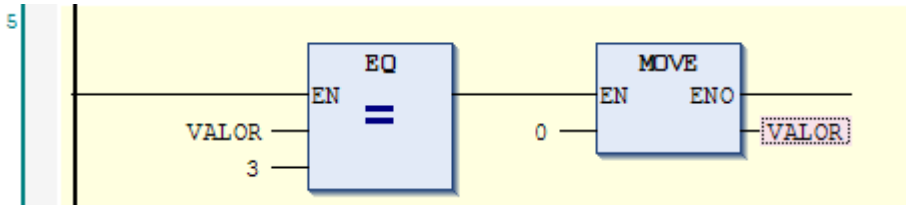
Por otro lado, hemos comparado este valor con una desigualdad, si es diferente de 0 activará su salida.

También lo hemos comparado a menor, si este valor es menor que uno.

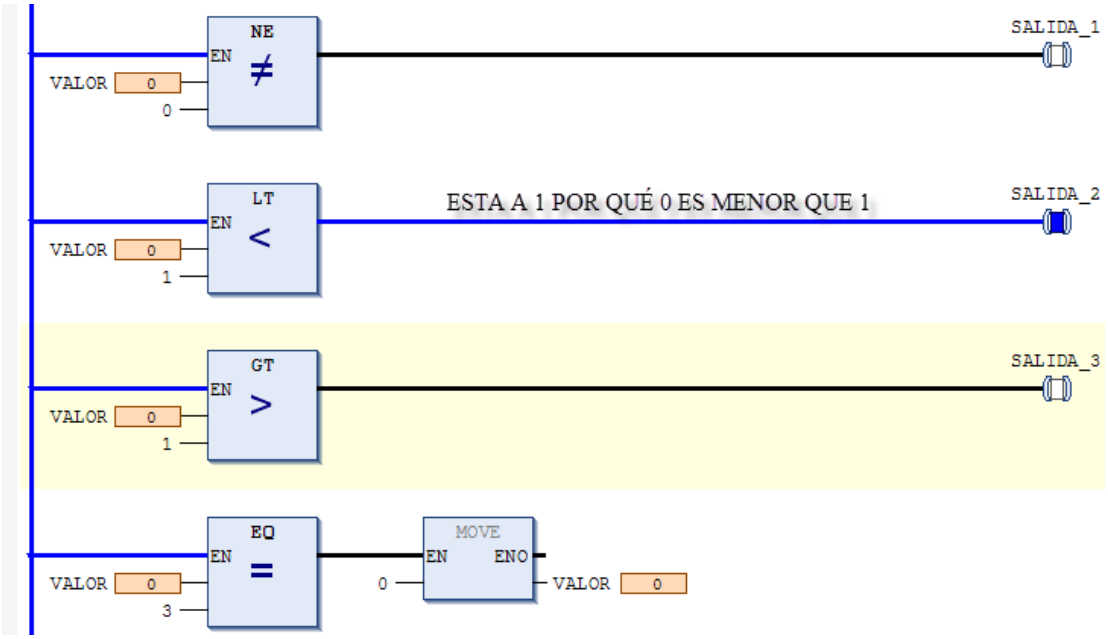
Hemos comparado si es mayor a 1



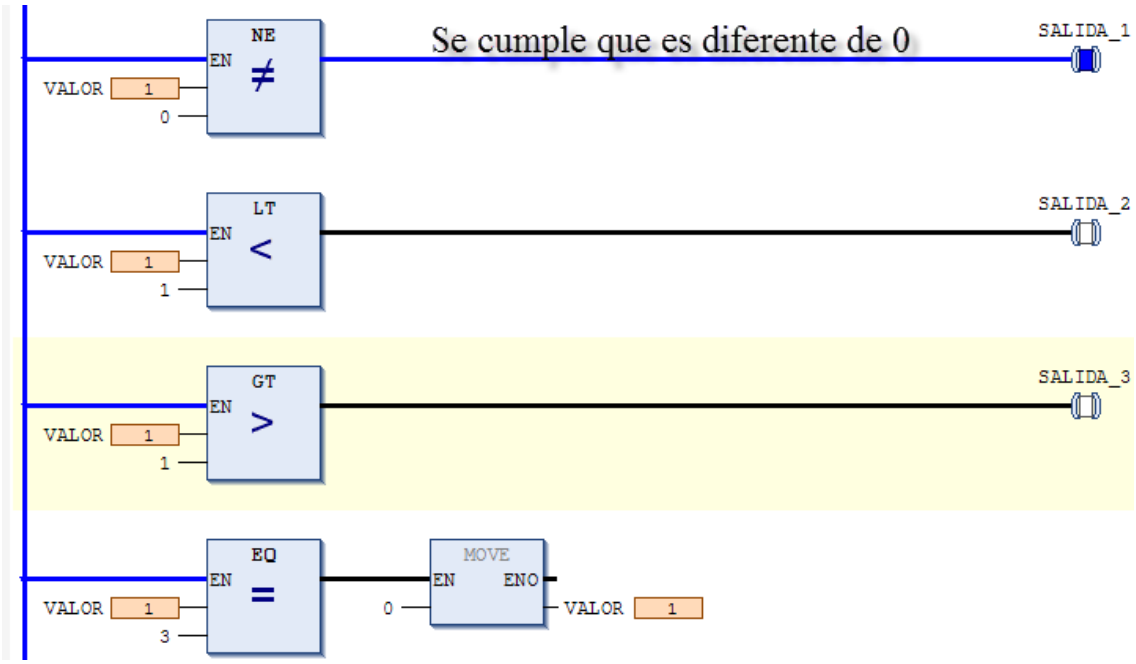
Y finalmente si este valor es igual a 3 haremos una transferencia a esta misma palabra cargando un 0 a la palabra.



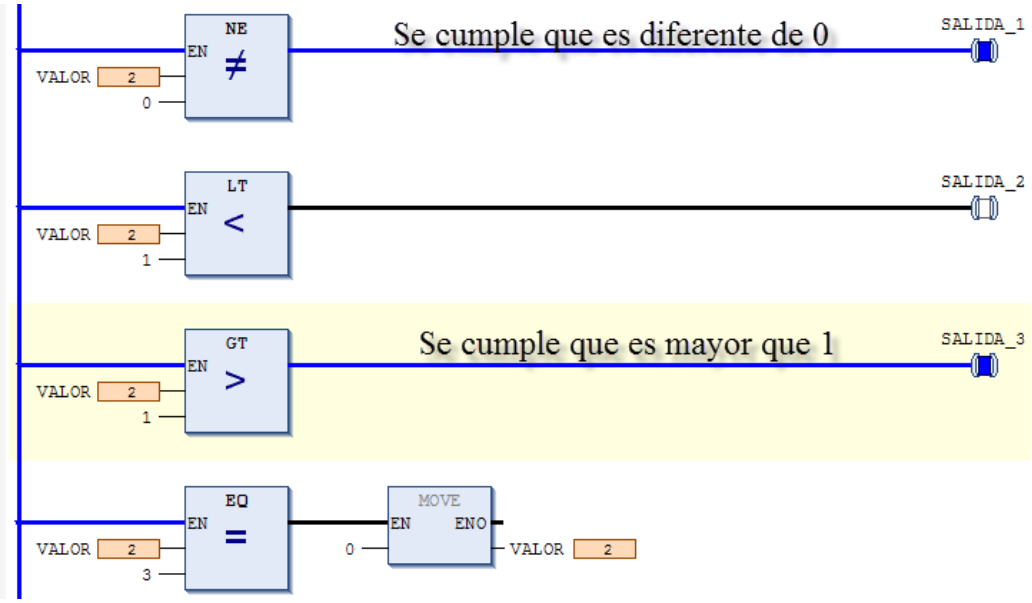
A continuación vemos el estado inicial del programa.



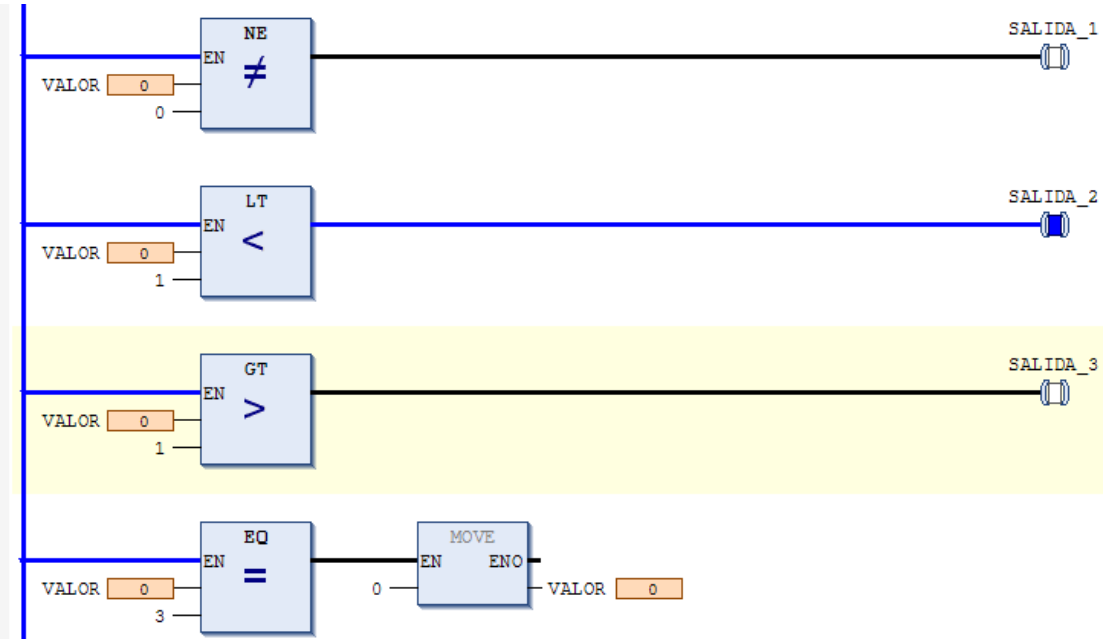
Como el valor actual es menor que 1 la salida activada es la que cumple esta comparación.



Vemos ahora que como hemos alcanzado el valor 1 se desactiva la comparación menor qué y se activa la comparación diferente de 0.



Como hemos aumentado el valor a 2 también se ha activado la comparación mayor que 1.

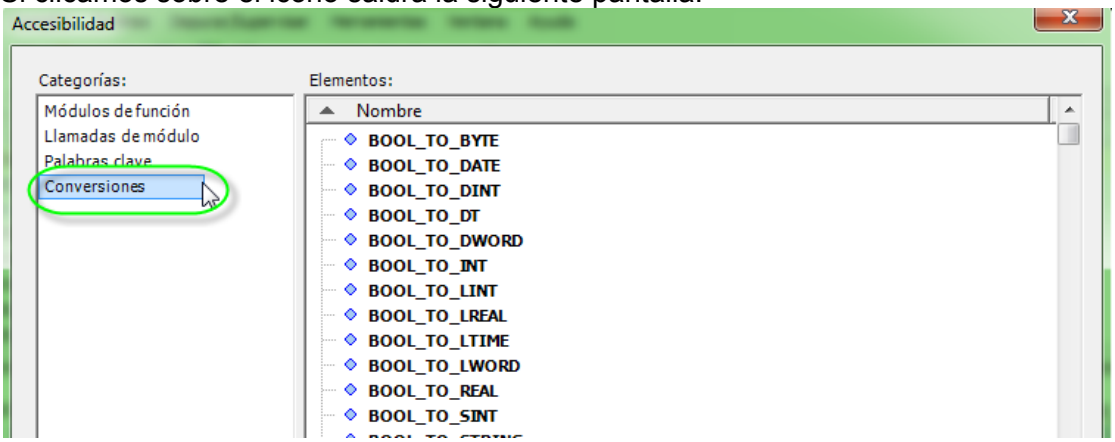


Ahora al alcanzar el valor 3 se cumple la comparación igual y se vuelve a cargar 0 en la variable “VALOR”.

3.6.6 Conversiones

Para poder acceder a estos módulos deberemos de hacerlo a través de la barra de herramientas.

Si clicamos sobre el icono saldrá la siguiente pantalla.



Vemos como a la izquierda de la pantalla nos aparecen las diferentes opciones que podemos utilizar y en la ventana central el despliegue de las diferentes opciones.

Con estos módulos podemos coger valores y convertirlos a otros formatos, es decir, de una Word a Dword, de Byte a Real, etc.

Keyword	Prefix ***	Size (Bit)	Range	Example
BOOL	x	8	0 , 1	FALSE, TRUE, 0, 1
SINT **INT** **DINT** **LINT**	si i di li	8 16 32 64	-128 .. 127 -32.768 .. 32.767 -2.147.483.648 .. 2.147.483.647 -2^{63} .. 2^{63}-1	0 24453 -38099887
USINT **UINT** **UDINT** **ULINT**	usi ui udi uli	8 16 32 64	0 .. 255 0 .. 65.535 0 .. 4.294.967.295 0 .. 2^{64}	200 47453 138099887
BYTE **WORD** **DWORD** **LWORD**	by w dw lw	8 16 32 64	0 .. 255 0 .. 65.535 0 .. 4294967295 0 .. 2^{64}-1	8450, 16#EA3F, 2#11_0011_1010
REAL	r	32	-1.2x 10^{-38} .. 3.4x 10^{38}	1.34996
TIME **LTIME*** **TOD** **DATE** **DT**	tim ltim tod date dt	32 64 32 32 64	0 ms .. 1193h2m47s295ms 0 ns..213503d23h34m33s709ms551us 615ns 00:00:00 .. 23:59:59 01.01.1970 bis ca. 06.02.2106	T#1d8h12m8s125ms LTIME#10d12h13ns TOD#12:34:17 D#2001-03-15 DT#2001-03-15-12:17:03
STRING(xx)** **WSTRING(xx)****	s ws		0 .. 255 Characters 0 .. 32767 Characters	'Hello world'

Es útil cuando algún modulo trabaja con un formato concreto y tenemos que poner un formato distinto al que pide.

CAPÍTULO

4

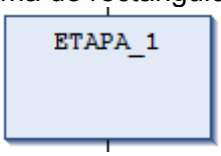
4. PROGRAMACION AVANZADA – GRAFCET (SFC)

4.1 LENGUAJE DE PROGRAMACION (SFC)

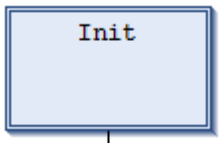
El lenguaje de programación SFC, se basa en la ejecución de un grafcet. El grafcet se aplica para realizar procesos secuenciales. Un proceso secuencial es aquel en el que primero se ha de dar una cosa para que luego pueda pasar otra y así sucesivamente. Dicho esto cuando programamos en SFC seguiremos las reglas básicas para desarrollar un grafcet.

Los tres elementos fundamentales dentro de un grafcet son:

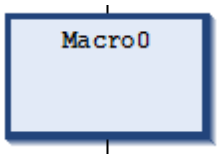
-ETAPAS, momento en el que nos encontramos dentro de la secuencia. Unity les llama Pasos. Se representan en forma de rectángulo.



Dentro de un grafcet siempre tendremos una etapa inicial, osea la primera, esta etapa siempre se identificará con un doble rectángulo.

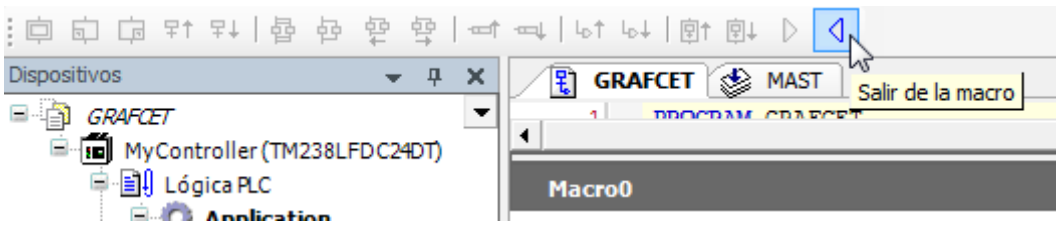


SoMachine, nos permite crear unas “etapas macro” en las que podemos generar un segundo grafcet asociado al primero, sería como llamar a una función dentro del grafcet pero en lenguaje SFC. Dentro de la macro tendremos una Paso_IN, donde empieza la macro, y un Paso_OUT, donde termina la macro.



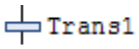
Si hacemos doble clic, sobre la macro, o clicamos sobre la barra de navegación en la flecha que señala a la izquierda entramos dentro de la macro.

Nos aparecerá una ventana, donde desarrollaremos el grafcet que contiene la macro de la misma manera que en el grafcet principal. Para volver al grafcet principal, deberemos dar a la flecha que apunta a la izquierda dento de la ventana de macro.

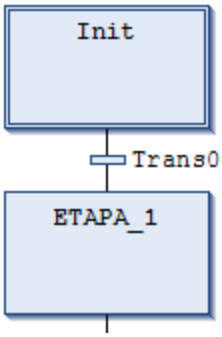


-TRANSICIONES, elemento que hace cambiar de etapa.

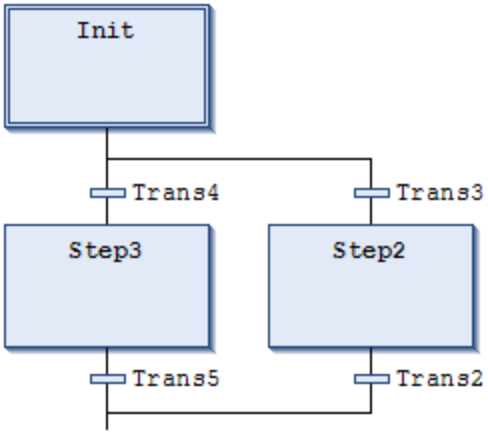
Las transiciones serán cambios de estado del algún bit (de 0 a 1 o de 1 a 0), que harán que cambiemos de etapa. Se representa con una cruz que une las etapas y en la linea de corte se escribe la variable que hace que cambiemos.



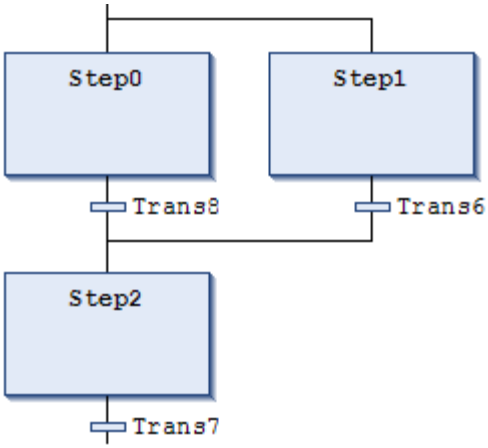
Si la insertamos entre etapas, quedaría de la siguiente manera.



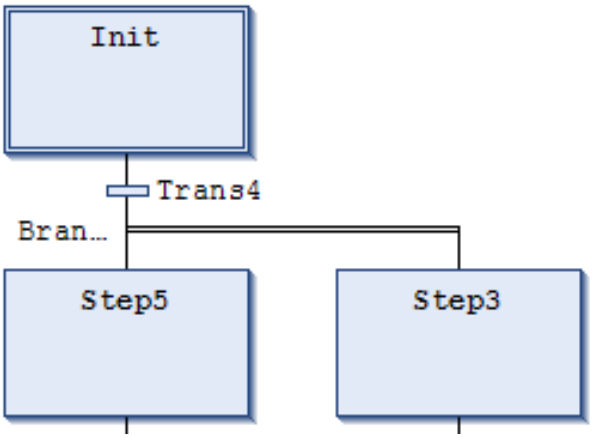
Bien si nos fijamos en el dibujo anterior la secuencia es totalmente lineal, la transición da paso a una sola etapa. A veces se nos puede presentar la necesidad de tener más de una transición que me de paso a “Ramificaciones Alternativas”.



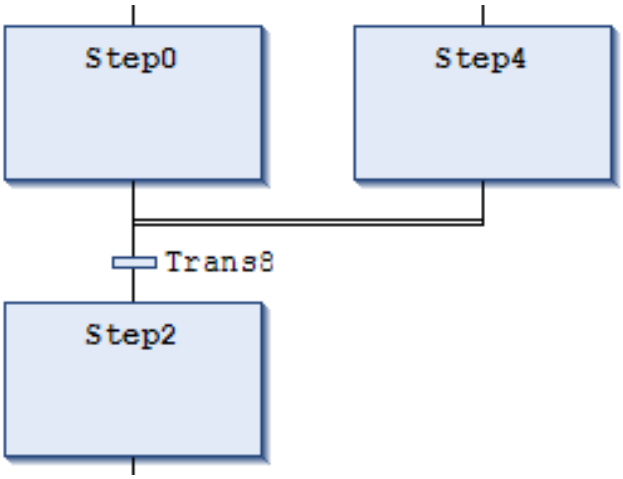
De la misma forma podemos tener dos etapa que vayan a la misma etapa “Conjunción Alternativa”.



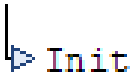
Otra situación que se nos puede dar es que tengamos dos ramas que se den de forma simultanea. Se denominan “Ramificaciones Paralelas”.



Para volver de dos “Ramificaciones Paralelas”, a una etapa lo que hacemos mediante las “Conjunciones Paralelas”, para activar la etapa siguiente se deberán cumplir las dos transiciones.

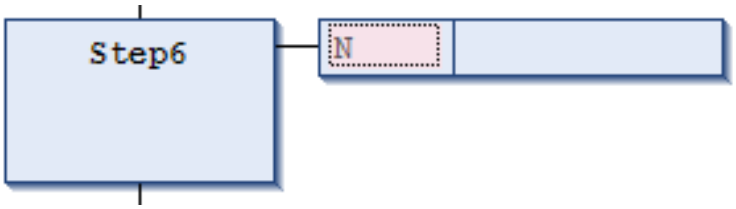


Otra forma de unir etapas sin tener que hacerlo mediante cable o una encima de la otra es acciondolo mediante el “SALTO”



De esta manera lo que hacemos es evitar cruce de cables.

-ACCIONES, que sucede en cada etapa. Aquí definiremos la repercusión que tiene la etapa dentro del programa, pone el motor 1 en marcha, el motor 2, un temporizador, etc.



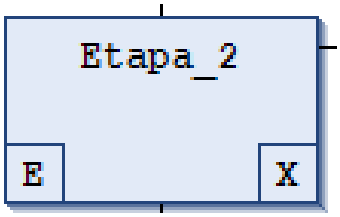
Para definir lo que hace cada etapa, tenemos una nomenclatura especial, es la siguiente.

- **N,** sería como la bobina de asignación se cumple mientras esta la etapa activada. Osea si nos encontramos en la etapa se cumple la acción.
- **R0,** pone a 0 permanente la acción.
- **S0,** pone a 1 permanente la acción.
- **L,** si la etapa se pone a 1 la acción se ejecuta el tiempo prefijado en la función, pasando a no ejecutarse aunque la etapa continue en 1. Si la etapa pasa a 0 la acción pasa a 0.
- **D,** si la etapa se pone a 1, transcurrido el tiempo prefijado la acción se pondrá a 1. Si la etapa pasa a 0 la acción pasa a 0.
- **P,** si la etapa se activa la acción se ejecutará un ciclo de Scan. Flanco positivo.
- **DS,** si la etapa se pone a 1, transcurrido el tiempo prefijado la acción se pondrá a 1 y permanecerá de esta manera hasta que desactivemos la acción, aunque cambiemos de etapa. Si la etapa pasa a 0 antes de haberse ejecutado la acción no se activará.
- **SL,** La acción se ejecuta, en el momento que se activa el paso. Se ejecuta, hasta tanto esté concluido el tiempo indicado o reciba un reset.
- **SD,** La acción se ejecuta, en el momento que se activa el paso. Se ejecuta, hasta tanto esté concluido el tiempo indicado o reciba un reset.

En el grafcet de SoMachine tenemos también la posibilidad de añadir acciones que se ejecutan a la entrada y a la salida de etapas.

Estas son; “Paso activo”, que se ejecuta cuando se activa la etapa y “Paso desactivado”, que se ejecuta cuando se desactiva la etapa.

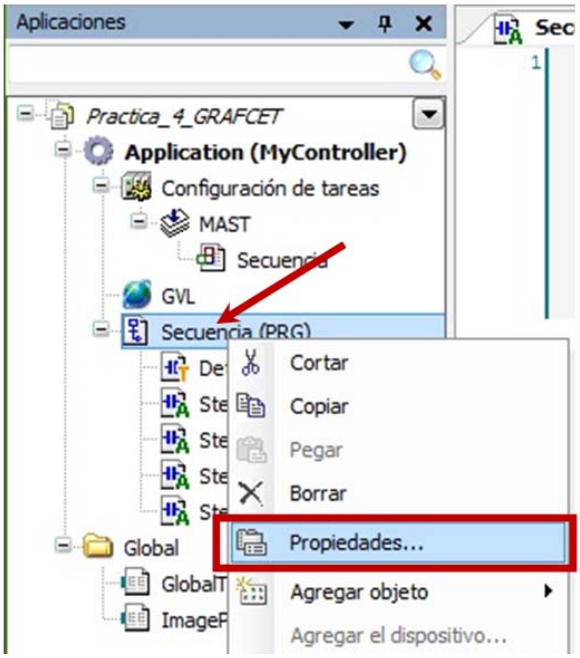
En la etapa quedaría de la siguiente manera.



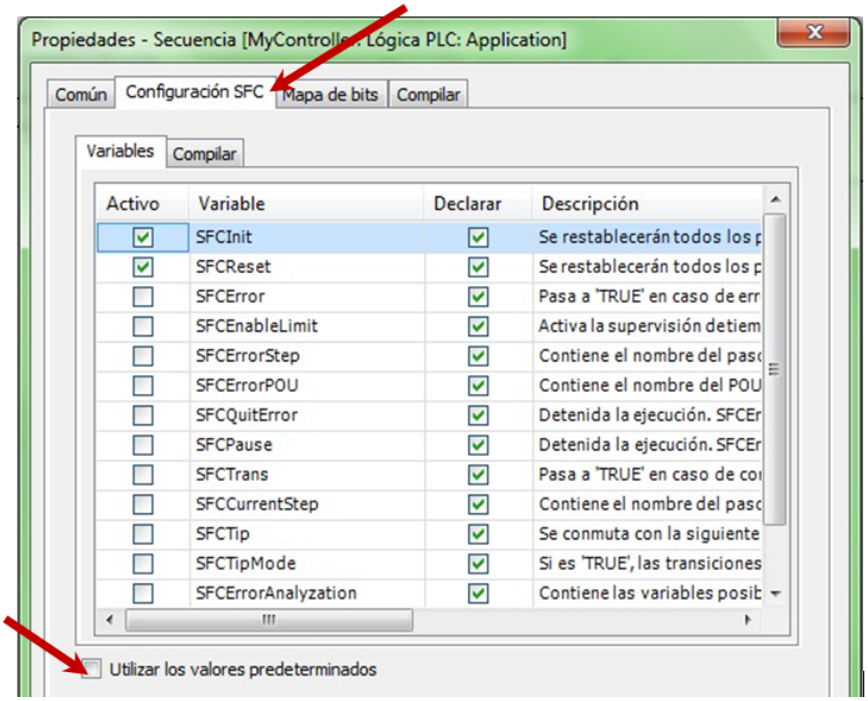
4.2 CONTROL DE LA EJECUCIÓN DEL PROGRAMA SFC

Puede utilizar algunas variables implícitas disponibles, para controlar el funcionamiento de un SFC. Por ejemplo, para indicar desbordamientos de tiempo o pausar la ejecución del programa para cambiar la activación de las transiciones.

Con el fin de poder acceder a estas variables implícitas se tienen que declarar para ser utilizadas. Para ello, seleccionaremos el Programa SFC creado y con el botón derecho abriremos y elegiremos la opción de **'Propiedades'**. Se trata de una ventana de configuración secundaria de las propiedades de objeto.



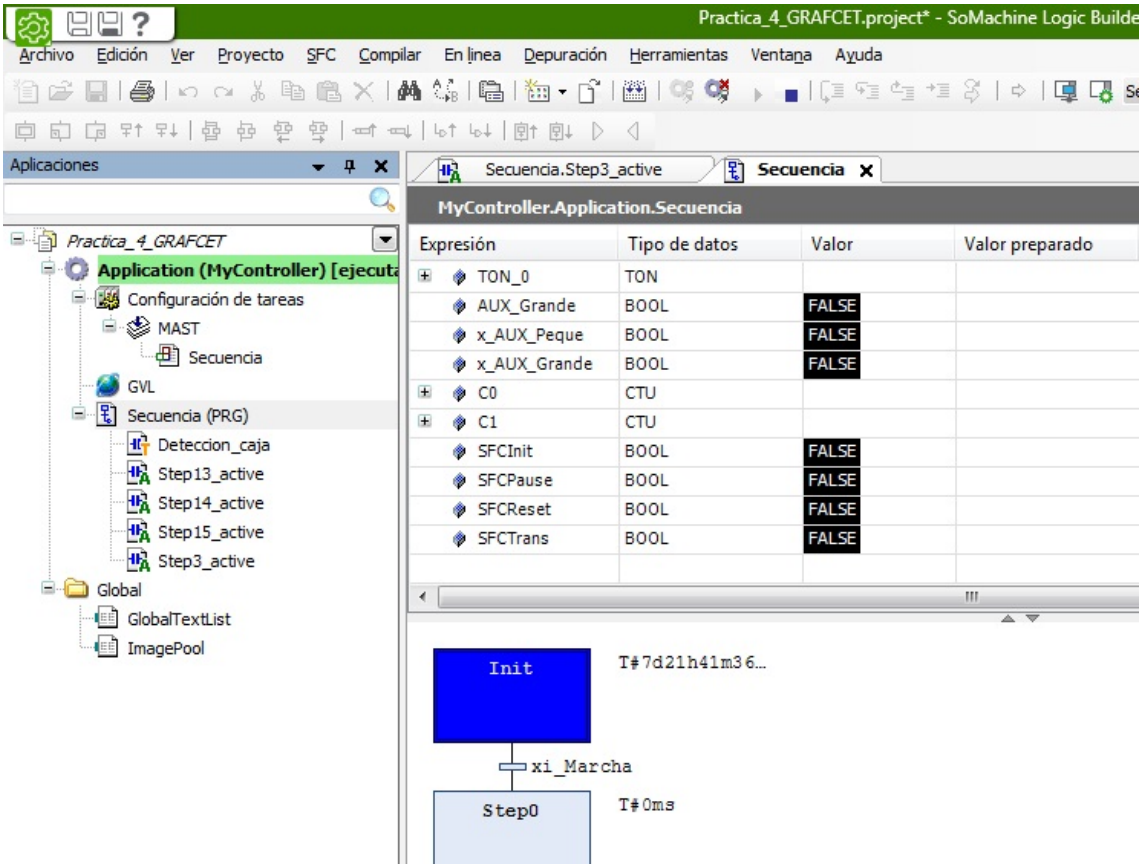
En la ventana flotante de propiedades, seleccionaremos la pestaña **'Configuración SFC'** y desactivaremos la opción de **'Utilizar los valores predeterminados'**.



Ahora seleccioanremos las variables de control de la ejecución del SFC que nos interesan.

Activo	Variable	Declarar	Descripción
☑	SFCInit	☑	Se restablecerán todos los pasos y acciones. Se activará el paso inicial. No se ejecutará ninguna acción.
☑	SFCReset	☑	Se restablecerán todos los pasos y acciones. El paso inicial se activa y sus acciones se ejecutan.
☐	SFCError	☑	Pasa a 'TRUE' en caso de error en la supervisión de tiempo.
☐	SFCEnableLimit	☑	Activa la supervisión detiempo para los pasos.
☐	SFCErrorStep	☑	Contiene el nombre del paso que ha puesto a 'TRUE' SFCError. SFCError debe estar activado.
☐	SFCErrorPOU	☑	Contiene el nombre del POU que ha puesto a 'TRUE' SFCError. SFCError debe estar activado.
☐	SFCQuitError	☑	Detenida la ejecución. SFCError restablecido. SFCError debe estar activado.
☐	SFCPause	☑	Detenida la ejecución. SFCError restablecido.
☐	SFCTrans	☑	Pasa a 'TRUE' en caso de conmutación de transición.
☐	SFCCurrentStep	☑	Contiene el nombre del paso activo.
☐	SFCTip	☑	Se conmuta con la siguiente transición con un flanco ascendente.
☐	SFCTipMode	☑	Si es 'TRUE', las transiciones sólo pueden conmutarse mediante SFCTip.
☐	SFCErrorAnalyzation	☑	Contiene las variables posibles, que establecieron SFCError en TRUE, representadas como cadena. Es necesario SFCErorr

Una vez seleccioandos y compilados cuando estemos online, aparecerán en la zona de declaración de variables del SFC.



También podemos llamar a estás variables, desde fuera del porgrama SFC, poniendo el **'nombre del POU.nombre de la variable implicita'** *(Por ejemplo: Secuencia.SFCInit).*

4.3 – VARIABLES IMPLÍCITAS DE LOS PASOS DEL PROGRAMA SFC

Básicamente , se crea para cada paso y cada acción IEC , una série de variables implícitas de tipo SFCStepType o SFCActionType . Los componentes de esta estructura generada para cada paso del SFC, (flags) describen el estado de un paso o acción o el tiempo transcurrido en un paso activo.

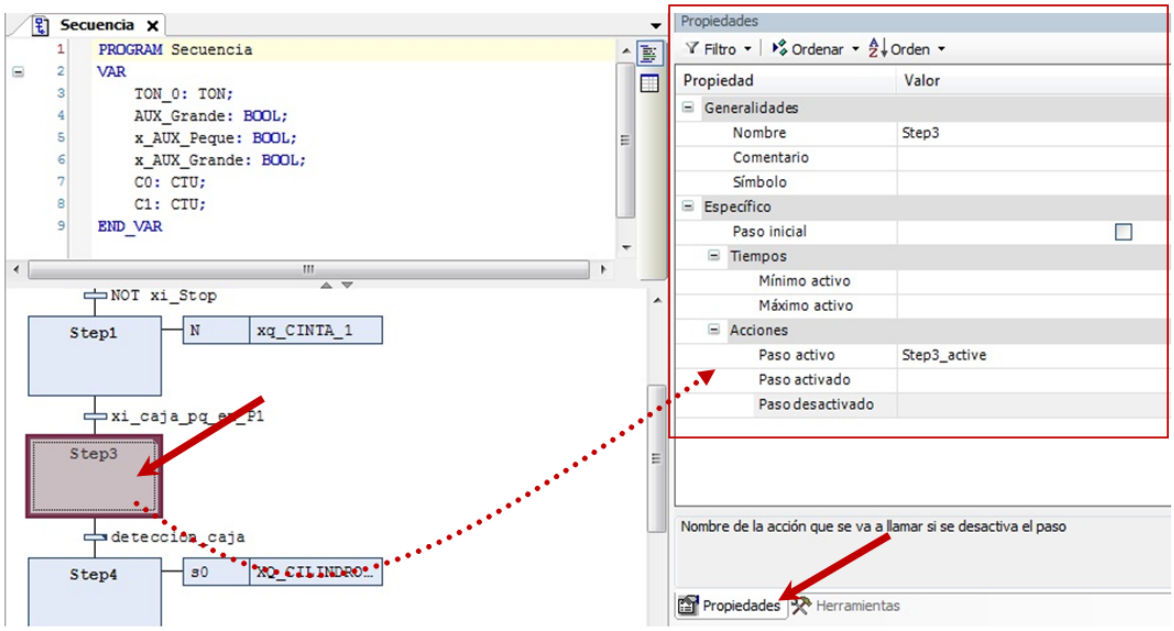
La sintaxis para la declaración de variables implícitamente es:

<Nombre del paso>.X : variable ímplicita booleana que nos indica si ese paso está activo. (por ejemplo: *secuencia.step0.x*)

<Nombre de la accion>.X : variable ímplicita booleana que nos indica si una acción asociada a un paso en concreto está activa.(*por ejemplo: step13_active.x)*

<Nombre del paso>.t : variable ímplicita tipo TIME que nos indica el tiempo que ha permanecido activo este paso. (por ejemplo: *secuencia.step2.t*)

También podemos asociar variables a las propiedades del paso, seleccionando el paso y pulsando la pestaña de propiedades del objeto en la parte de la derecha podemos configurar y ver la información del paso.



Paso inicial: Esta opción se activa en las propiedades de la etapa inicial de corriente (paso init). De forma predeterminada, se activa para el primer paso de un SFC y aparece desactivado para los otros pasos. Si se activa esta opción para otro paso, debe desactivarlo en el paso init. De lo contrario, se generará un error de compilación.

Tiempos: Define los tiempos mínimos y máximos que puede estar activo el paso.
NOTA: si el tiempo no está dentro del rango de tiempo mínimo o máximo esto se indicará activando la variable implícita SFCError.

Acciones: Define las acciones a realizar cuando el paso está activo.

- Paso Activo: Esta acción es lo primero que se ejecutará después de que la etapa se ha activado.

- Paso Activado: Esta acción se ejecuta cuando ya se ha ejecutado la acción del Paso Activo y la acción aún permanece activa.

- Paso desactivado: Esta acción se ejecutará en el ciclo posterior, después de que el paso se haya desactivado.

4.4 DESARROLLO DE UN EJEMPLO DE PROGRAMACION SFC

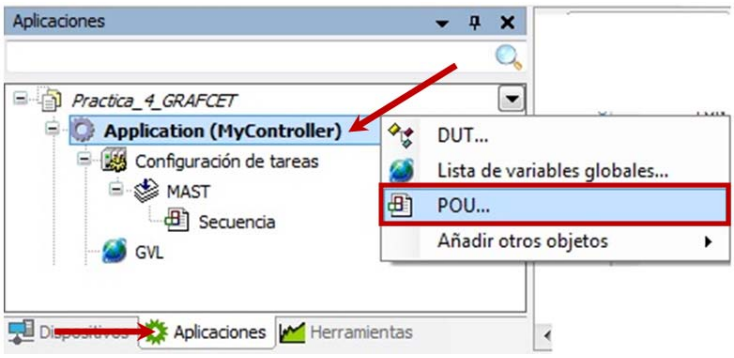
Para terminar de entender el funcionamiento del grafcet vamos a realizar, una práctica ejemplo, la cual iremos variando para entender todas las opciones de este lenguaje de programación.

- **Arranque de un motor**

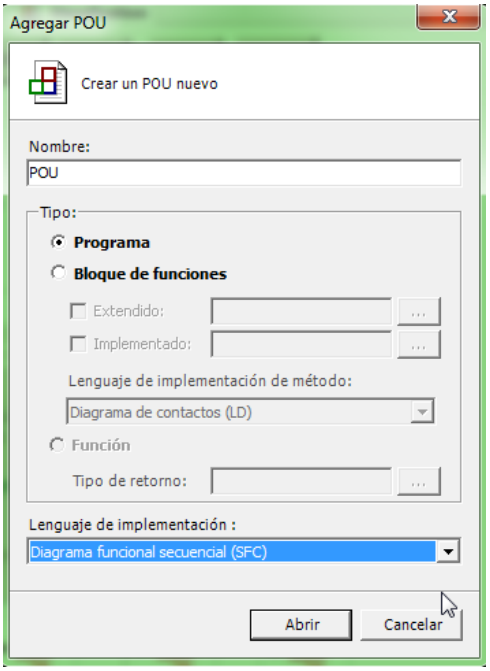
Tenemos un motor y queremos realizar el arranque estrella triángulo.

El esquema eléctrico que hemos realizado sobre el controlador M241 es el que se muestra en la figura:

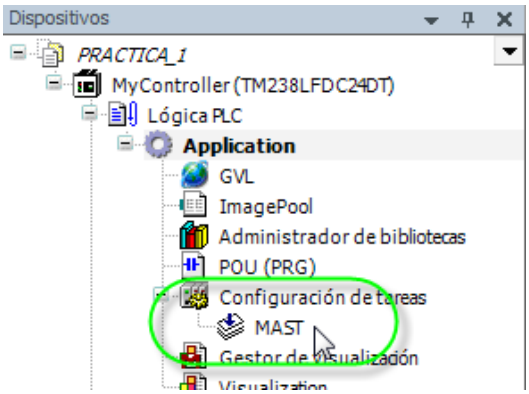
Una vez hemos creado el proyecto con el asistente, ya podemos desarrollar nuestro programa. Crearemos un POU, seleccionando la pestaña del navegador de **'Aplicaciones',** dentro seleccionaremos **'APPLICATION'** y pulsaremos el icono **'+'** que nos aparece y agregaremos el objeto **"POU"**



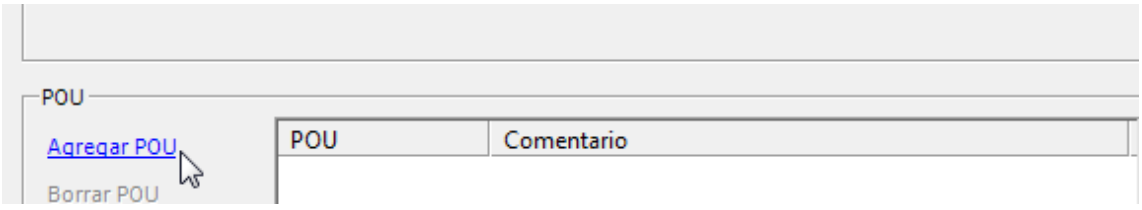
Aceptamos “PROGRAMA” ,dejamos el mismo nombre y escogemos lenguaje de programación “DIAGRAMA FUNCIONAL SECUENCIAL (SFC)”



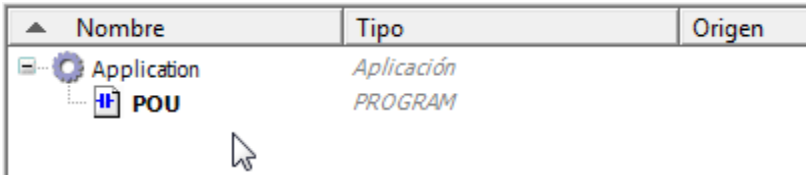
Abrimos **‘Configuración de tareas’** y seleccionamos **“MAST”** haciendo doble clic sobre él.



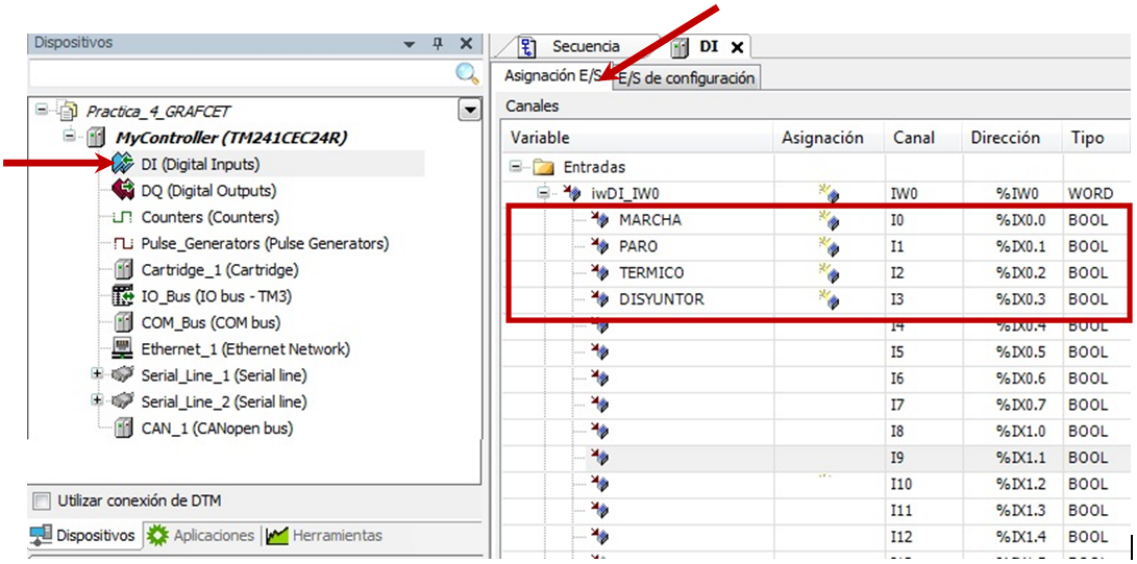
Clicamos sobre “Agregar POU”.

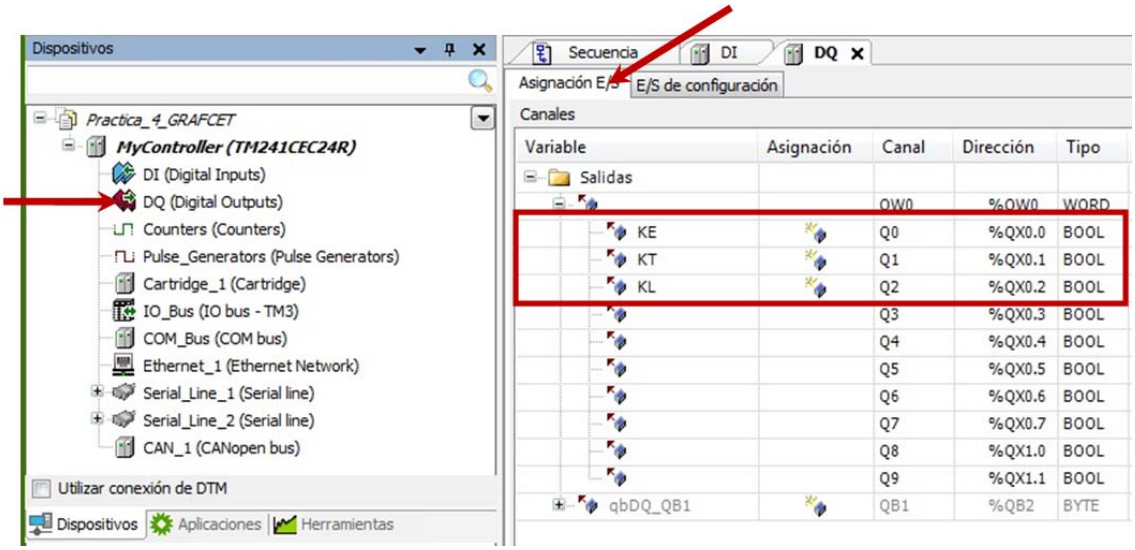


Hacemos doble clic sobre “POU”

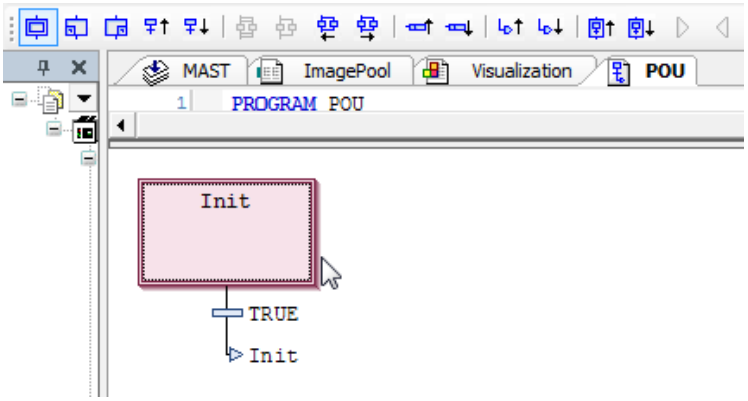


Declaramos la siguientes entradas y salidas físicas:



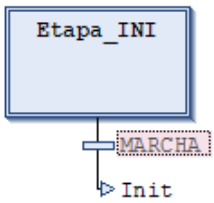


Vemos como en la parte superior izquierda nos han aparecido los iconos de los que hemos hablado antes y la etapa inicial con su transición y un salto.

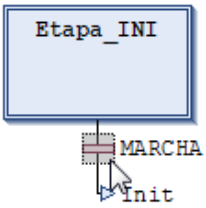


Si queremos podemos cambiar el nombre a la etapa, y llamarla “Etapa_INI”, por ejemplo. Clicamos sobre “INIT”

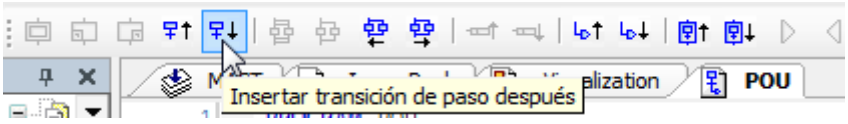
En la transición, pondremos que al pulsar marcha se ponga el proceso en marcha. De momento no tenemos en cuenta el paro, ni las protecciones témicas.



Al dar a marcha pasamos a la etapa 1, por lo que insertaremos una etapa y transición por debajo de la transición, la forma de hacerlo es marcando en la transición.

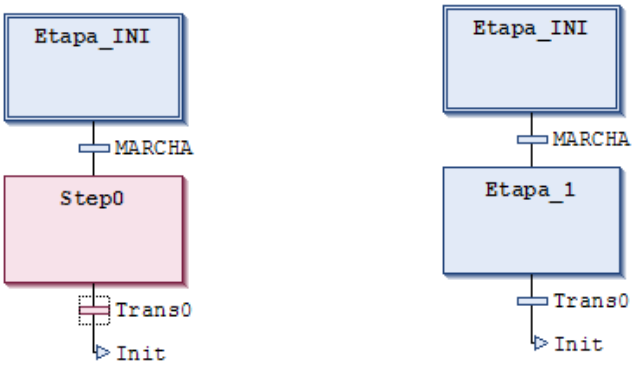


Observamos como se pone de color rosa, y ahora clicamos sobre el icono “insertar transición de paso después”



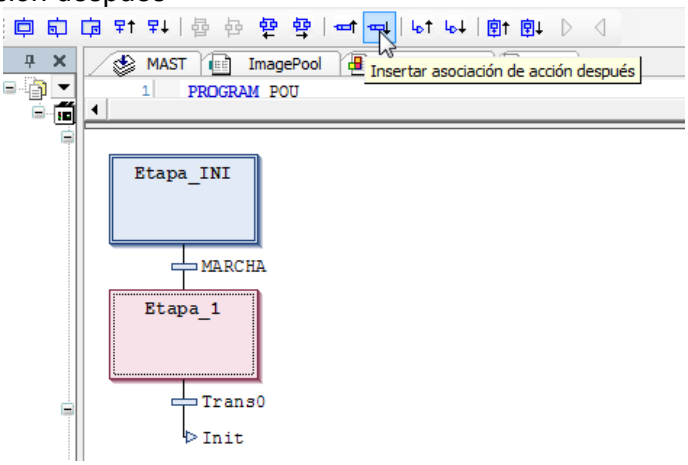
Vemos como aparece una nueva etapa con su transición.

Podemos como a la anterior cambiarle el nombre

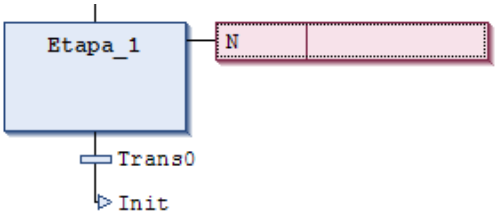


Ahora definiremos la acción o acciones que van asociadadas a esta etapa, en nuestro ejemplo deberemos hacer que se activen las salidas **KL**, **KE**, y un temporizador para que cambie de estrella a triángulo.

Clicamos sobre la “Etapa_1” que se pondrá en rosa y sobre el icono “insertar asociación de acción después”

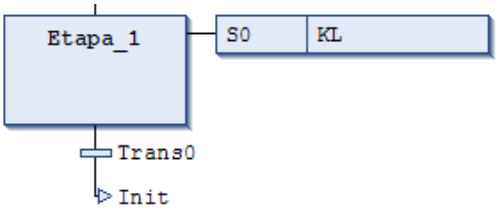


Vemos como aparece asociada la acción a la etapa.

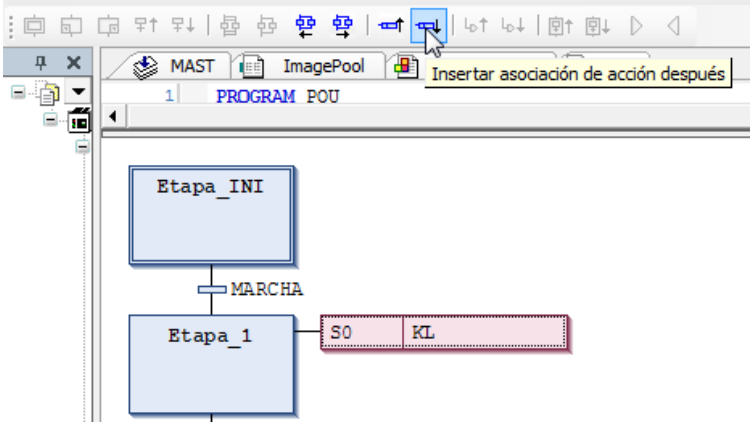


Ahora escribiremos en la izquierda el calificador de lo que va a hacer, y en la derecha lo que hace.

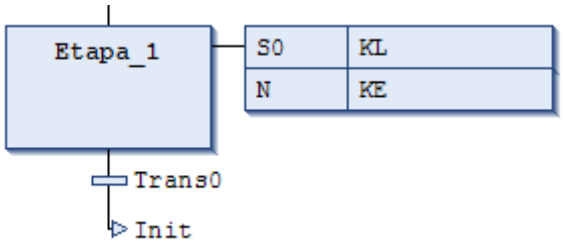
Por ejemplo activa la salida **“KL”**. Esto hará que la salida **“KL”** esté a uno mientras no lo reseteemos.



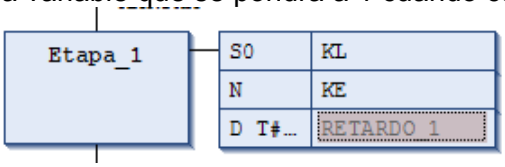
Ahora marcamos sobre la acción que se pondrá en rosa y sobre el icono **“insertar asociación de acción después”** para añadir otra acción.



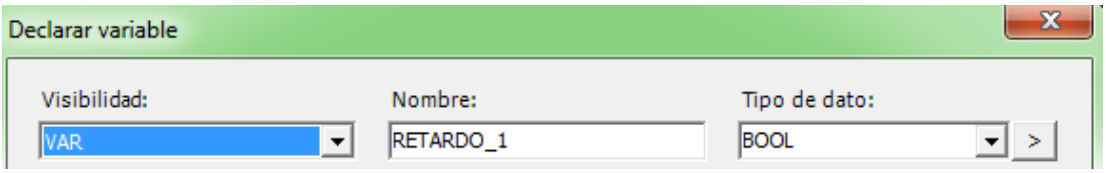
En esta haremos que mientras esté activa la etapa, también lo esté la salida “KE”, esto se hace con el calificador “N”.



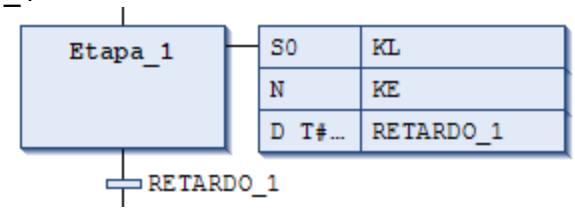
Y por último insertamos otra acción que será la del Delay “retardo”, para cambiar de etapa. El codificador será la letra D seguido del tiempo en formato TIME, y en el recuadro de la derecha, la variable que se pondrá a 1 cuando cuente.



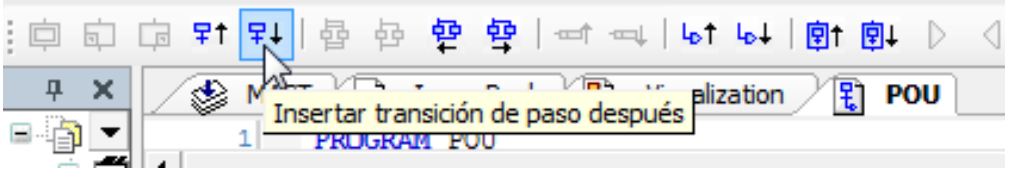
Como la variable no la habiamos creado nos sale el recuadro para crearla. Le damos a aceptar.



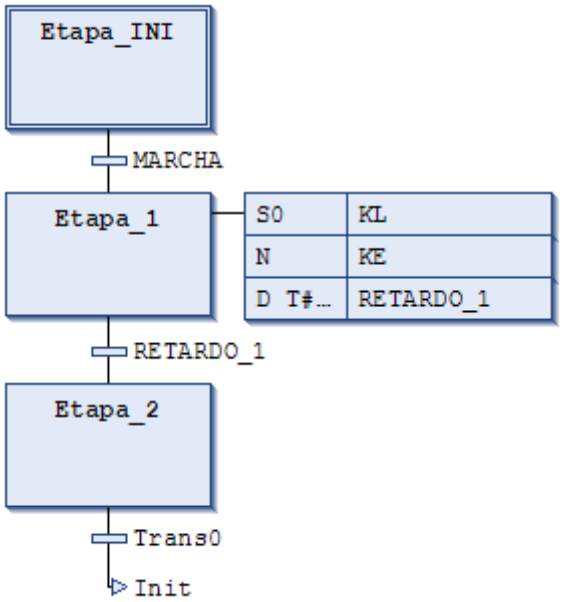
Ponemos ahora la condición de transición, en nuestro caso que se ponga a 1 la variable “RETARDO_1”



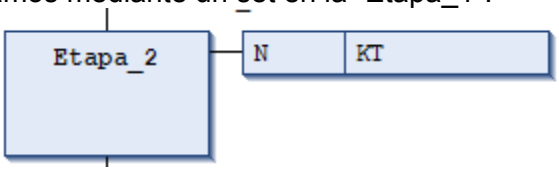
Vamos a insertar ahora la “Etapa_2”, procederemos de la misma manera que con la “Etapa_1”, insertaremos una etapa y transición por debajo de la transición. Marcamos la transición y clicamos sobre “insertar transición de paso después”



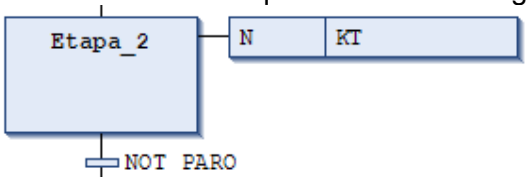
Nos aparece la etapa, a la que le cambiamos el nombre por “Etapa_2”, con su transición.



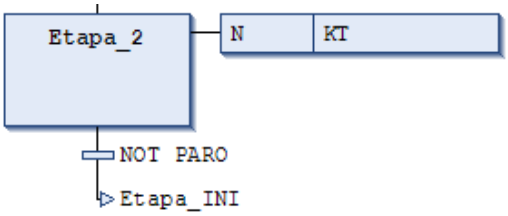
Le insertamos las acciones que van asociadas a esta etapa, en nuestro caso tendremos que hacer que mientras esté la etapa, lo esté también la salida KT, ya que la salida “KL” la activamos mediante un set en la “Etapa_1”.



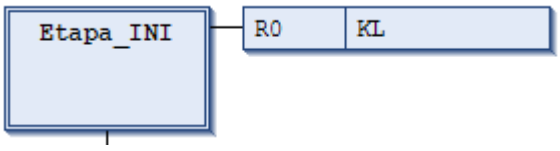
La transición, para detener el proceso, será que demos al paro. Como el paro está cableado normalmente cerrado deberemos negar esta entrada para que haga la transición cuando pase al estado 0. Por lo que escribimos lo siguiente.



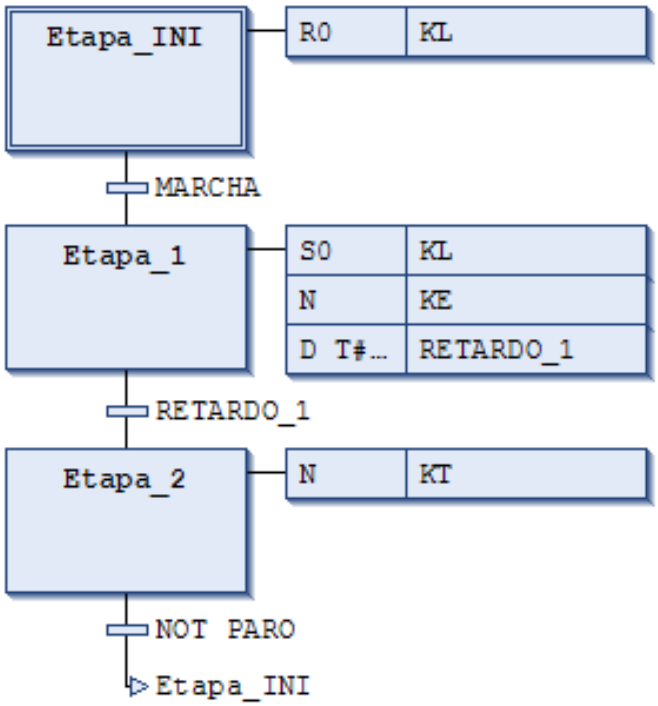
Por último, deberemos escribir en el salto el nombre de la etapa a la que queremos que vaya, al darle al paro en nuestro caso la inicial.



Tambien deberemos añadir en la etapa inicial, una acción para resetear el “KL”, ya que si no estaria a 1 siempre.

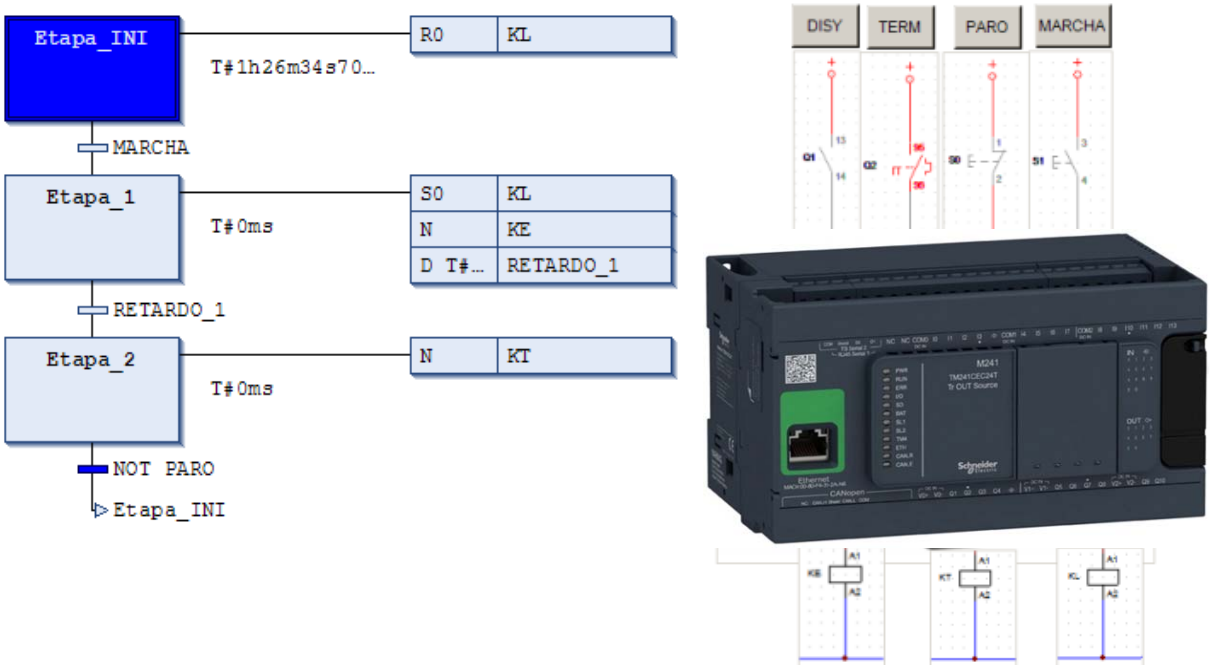


El programa quedaría de la siguiente manera.

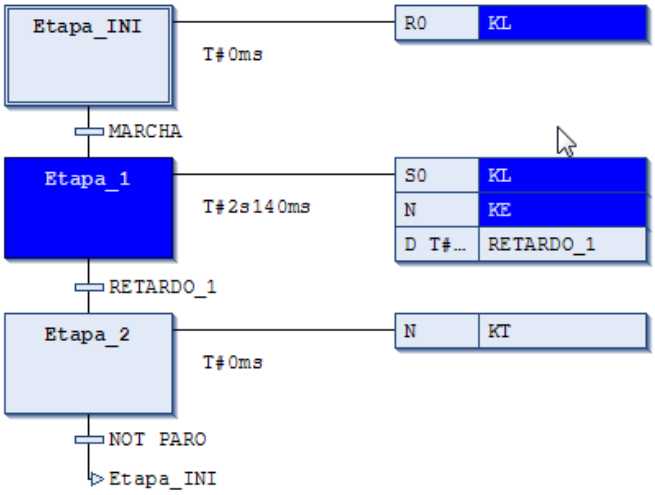


Una vez hemos realizado la programación de la secuencia SFC, compilamos y cargamos el programa en el autómata M241 para ver su comportamiento.

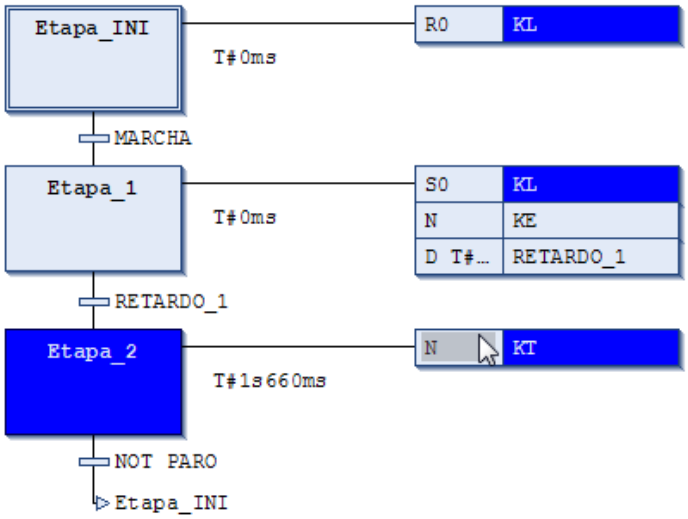
Si nos fijamos vemos com nos encontramos en la etapa inicial y no hay nada en marcha. Las etapas en que nos encontramos están en color azul oscuro.

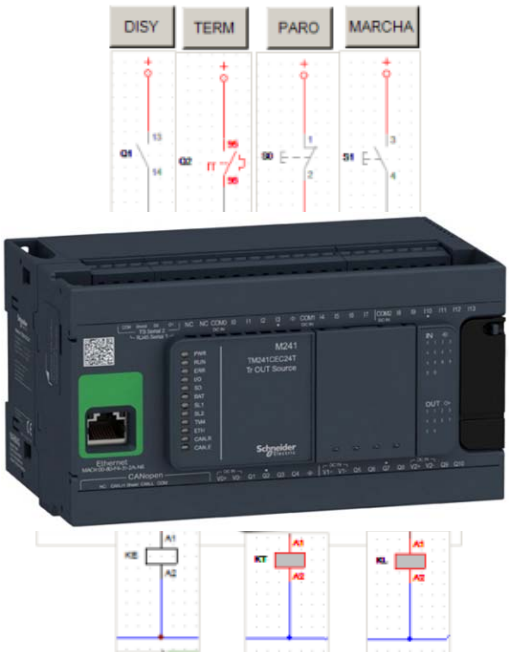


Al dar a **“MARCHA”**, vemos como cambia a la etapa 1,y se ponen en marcha **“KE”** y **“KL”.** Tambien vemos como temporiza los 5 segundos que hemos prefijado.

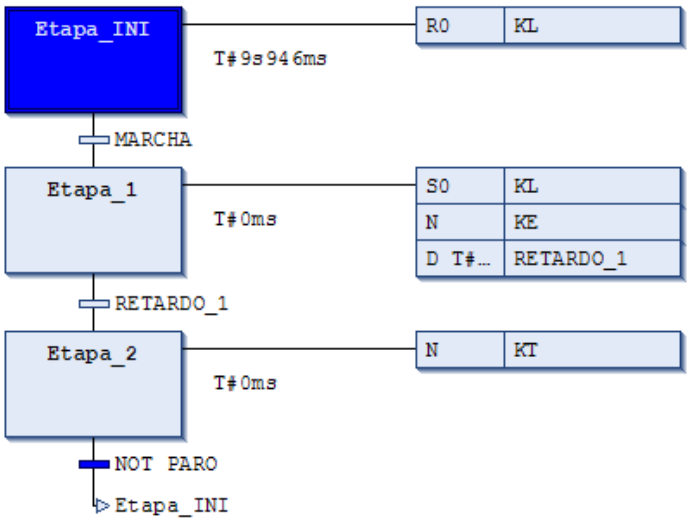


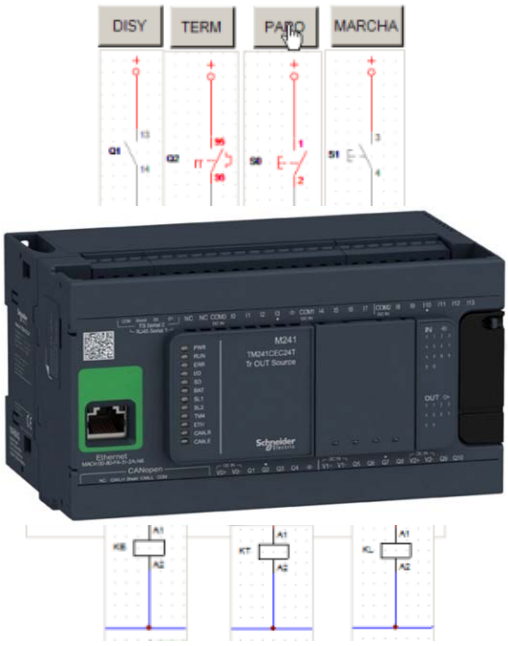
Cuando termina la temporización cambia de etapa poniendo en marcha “KT” Y “KL”





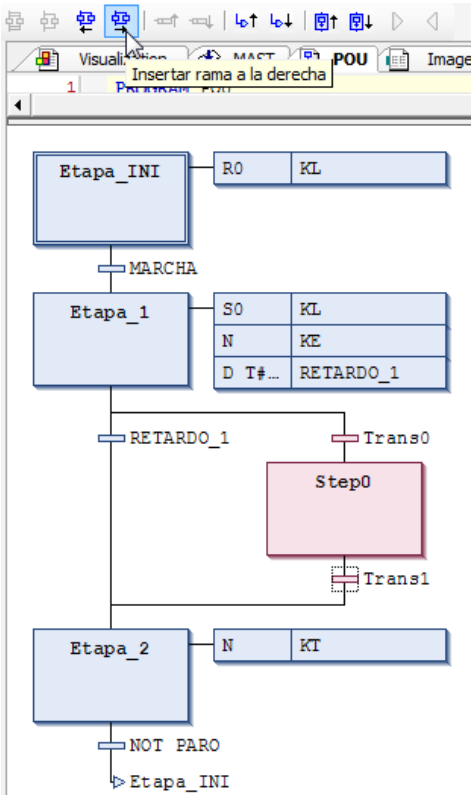
Por último si damos al paro se para definitivamente.





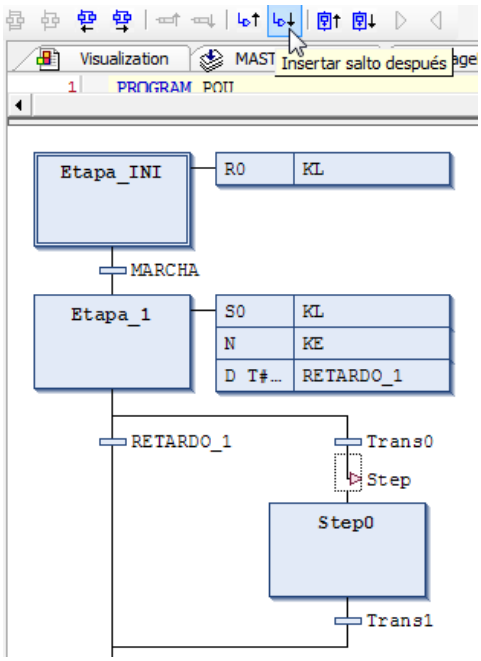
Bien hemos hecho la secuencia para la que todo se da bien, pero no hemos tenido encuenta que pasaría sí, en la etapa 1 accionasemos el paro o en la 1 o 2 saltasen la protecciones térmicas. En estos casos deberiamos ir directamente a la etapa 1.

Para parar el motor si estamos en la etapa 1, deberemos poner una rama alternativa. Marcamos sobre la transición queestá entre la “Etapa_1” y la “Etapa_2” y clicamos en “Insertar rama a la derecha”

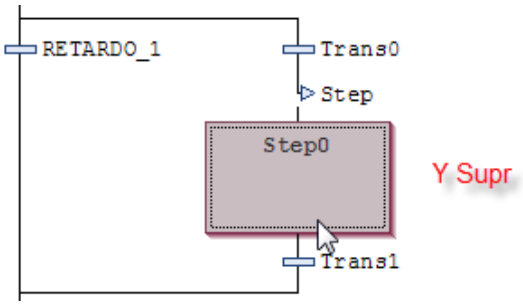


Vemos como aparte de la rama nos aparece una etapa y dos transiciones, como a nosotros no nos interesa esto, ya que lo que queremos es que si le damos al paro o saltan las protecciones térmicas vaya a la “Etapa_INI”, deberemos eliminar la etapa y una transición.

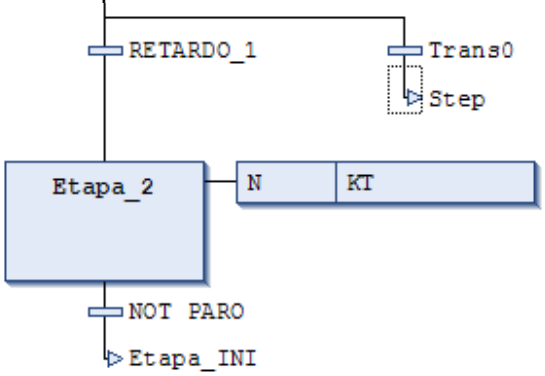
Primero insertamos un salto después de la “Trans0”, para lo que marcamos sobre esta transición y clicamos sobre “insertar salto después.



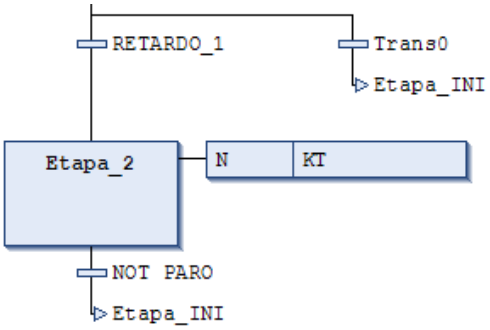
Par eliminar los elementos marcamos sobre el que queramos eliminar y daremos a suprimir.



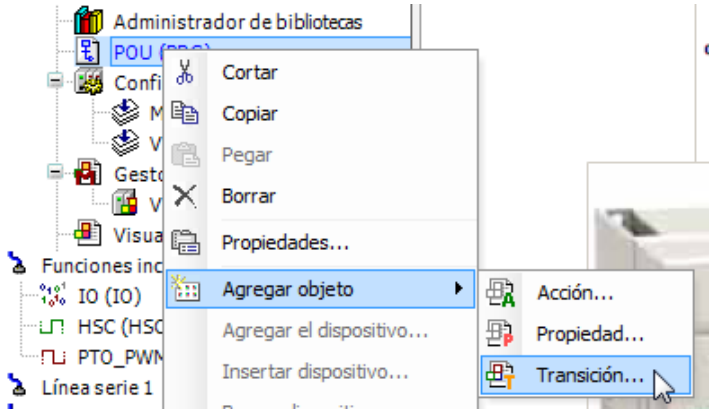
Una vez eliminadas debería quedar así.



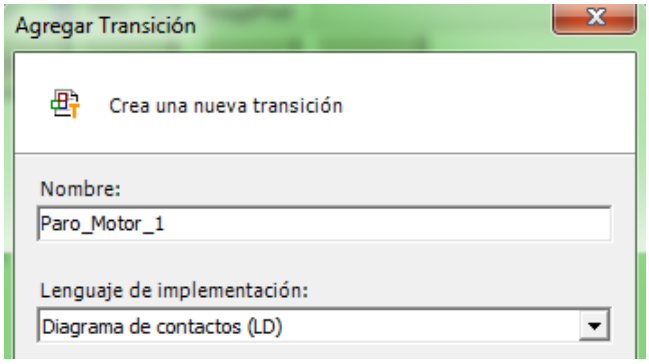
En el salto, pondremos ir a la “Etapa_INI”.



Ahora procedemos a definir la transición, esta debe ser que el paro, el disyuntor y el térmico esten a 0. Para este caso necesitaremos crear una transición. Para insertar la transición, sobre POU damos a botón derecho, agregar objeto, transición.

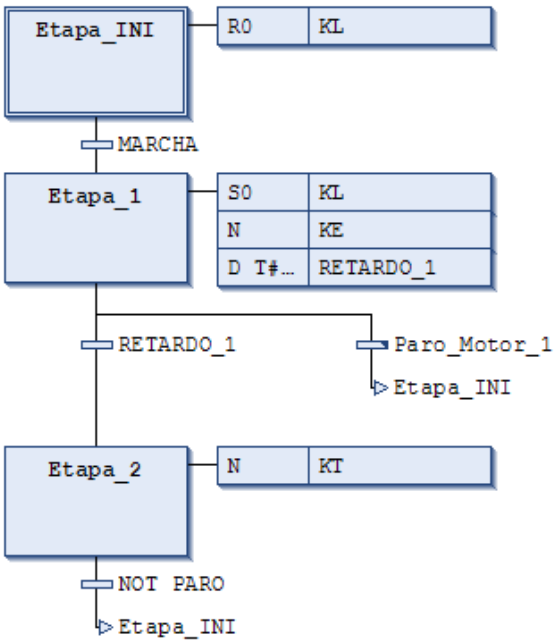


Al clicar sobre transición, nos aparece la siguiente ventana, donde pondremos el nombre de la transición y el lenguaje de programación con el que vamos a trabajar.

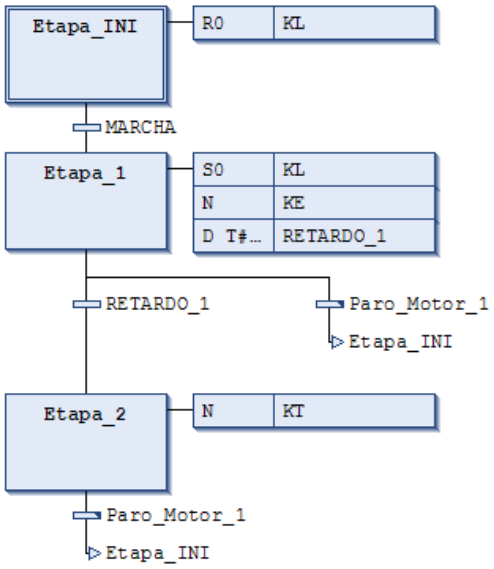


Nos aparece la ventana de programación en ladder y escribiremos que si el paro o el térmico o el disyuntor es 0 estará activada la salida “Paro_Motor_1”

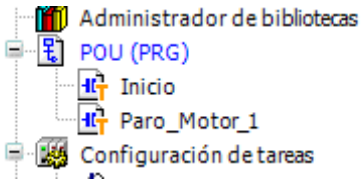
En la “Transición_0” la sustituimos por “Paro_Motor_1”, quedando de la siguiente manera.



Después de la etapa tres, en la transición tambien pudiera ser que saltase uno de las dos protecciones, por lo que podremos sustituir la transición de “NOT PARO” por la sección “Paro_motor_1”.

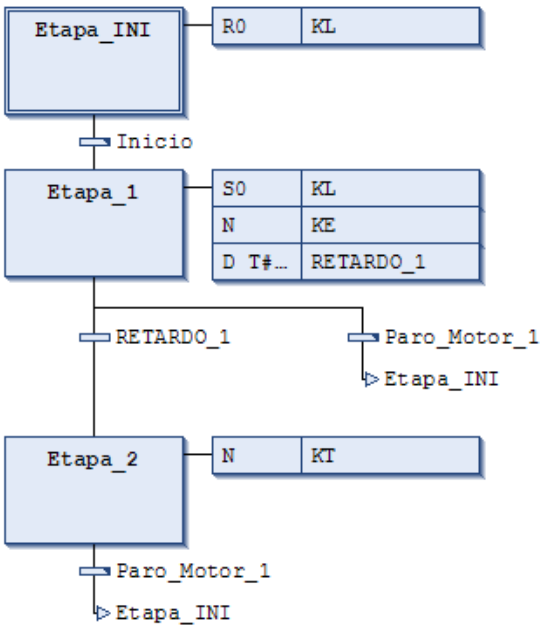


Tambien podemos tener en cuenta que para empezar el proceso, el paro, el Q1 y Q2 esten a 1. Crearemos tansición que se llame inicio, que será la transición de la “Etapa_INI” a la “Etapa_1”.

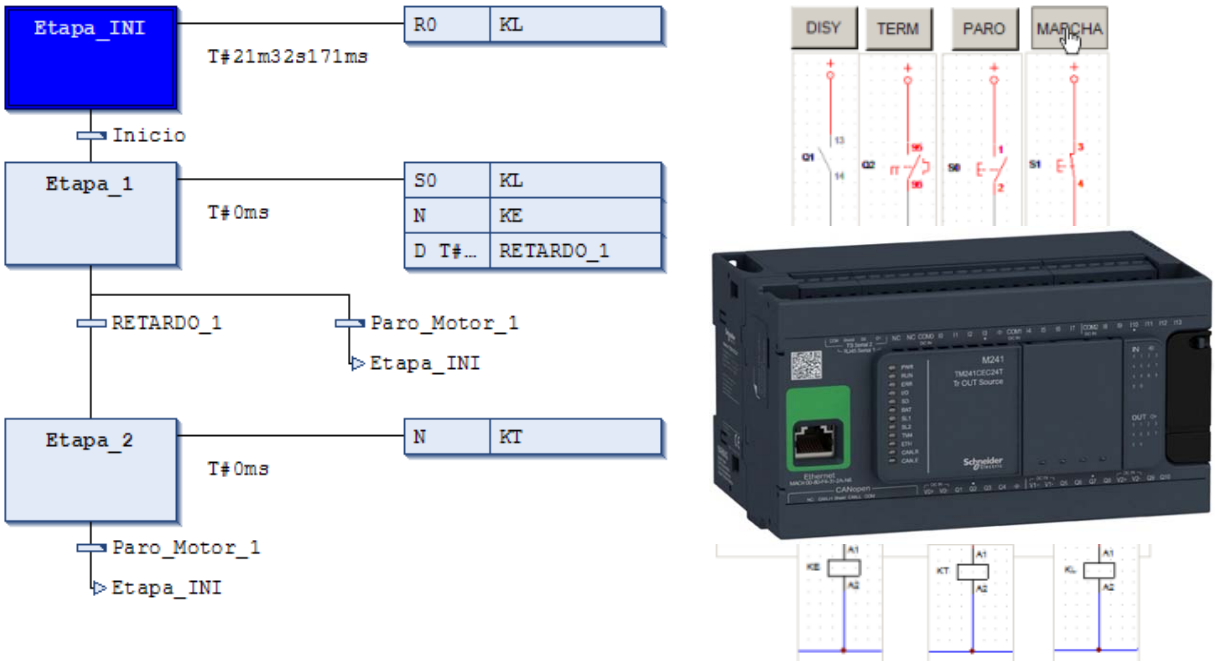


En esta transición escribiremos lo siguiente:

El grafcet quedaría de la siguiente manera.



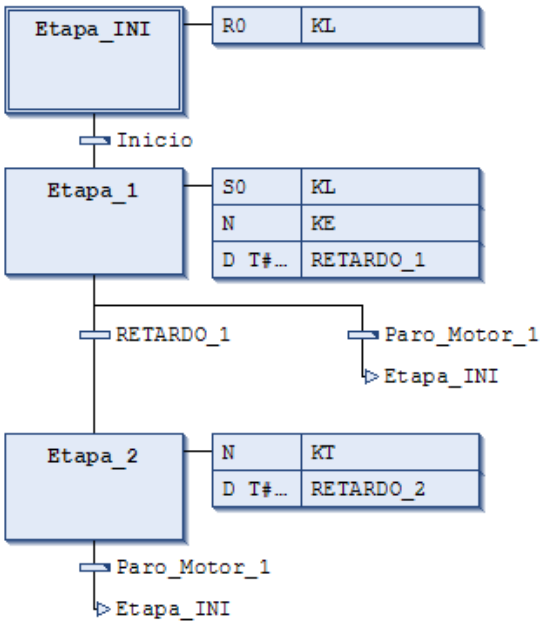
Vemos como aunque demos a marcha no arranca el motor, si no hemos cerrado las tres protecciones.



Bien imaginemos ahora, que queremos que el motor arranque 5 segundos en estrella y luego este 5s en triángulo y este proceso lo queremos realizar 5 veces para detenerse finalmente.

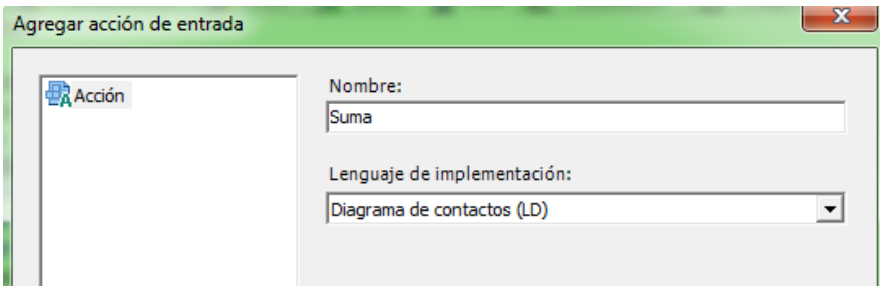
Deberemos entonces poner un DELAY en la etapa 2 y hacer que cada vez que se active esta etapa cuente uno.

Primero insertamos un delay como hemos visto con anterioridad. Primero insertamos la acción y despues la definimos de la siguiente manera.

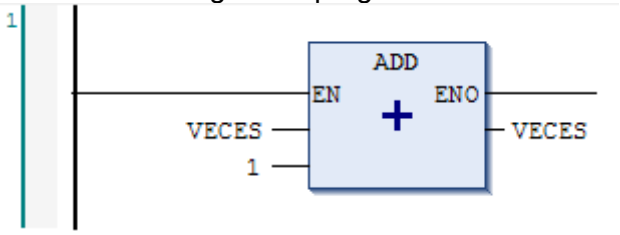


Ahora insertaremos una acción de entrada, marcamos sobre la etapa 2 y clicamos en el icono que observamos.

Al marcar sobre el icono, saldrá la siguiente ventana, donde escribiremos el nombre “Suma” y lenguaje de programación “LADDER”.



En la acción “Suma” escribimos el siguiente programa.

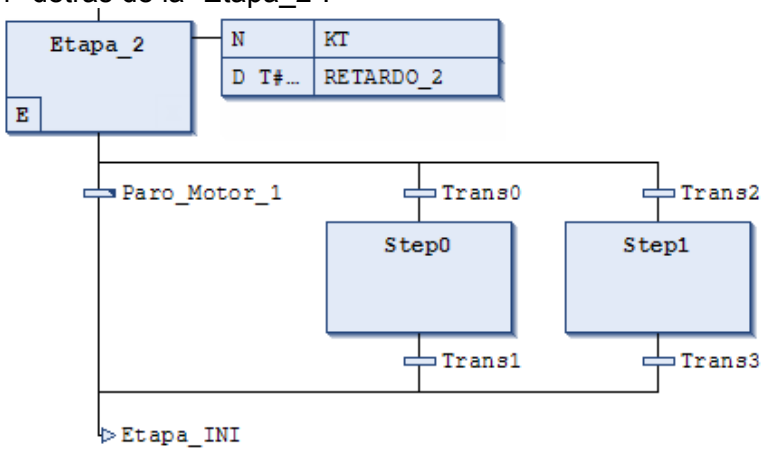


Como la variable “VECES” no estaba creada, nos aparecerá la ventana de variables donde la definiremos como “INT”. Podemos observar como ha sido añadida en las variables del “POU”.

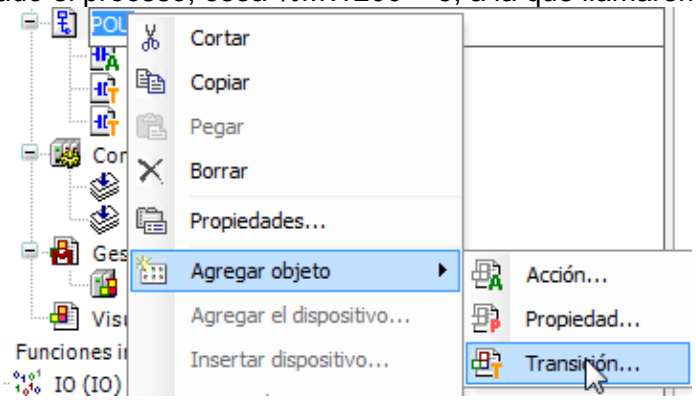
```
8       KT AT %QX0.1: BOOL;
9       KL AT %QX0.2: BOOL;
10      RETARDO_1: BOOL;
11      RETARDO_2: BOOL;
12      VECES: INT;
13  END_VAR
```

Bien ahora tendremos que añadir dos transiciones más detrás de la Etapa_2, una si ha contado el temporizador y %MW200< 5 y otra en la que ha contado el temporizador y %MW5>=5.

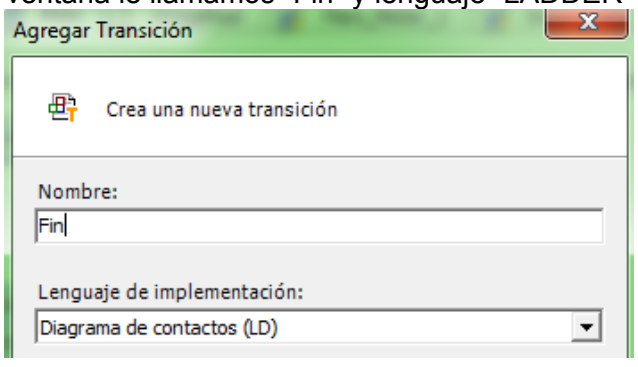
Lo primero que hacemos es insertar dos ramas alternativas a la transición “Paro_Motor_1” detrás de la “Etapa_2”.



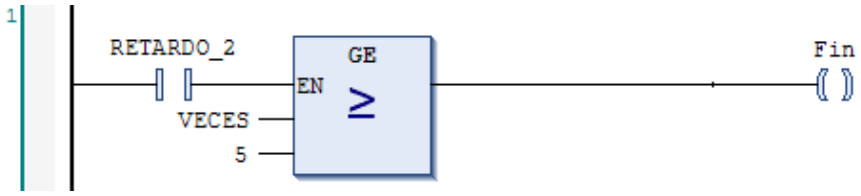
Bien ahora definiremos las dos transiciones, La primera que vamos a hacer es en la que hemos acabado el proceso, osea %MW200>=5, a la que llamaremos “Fin”.



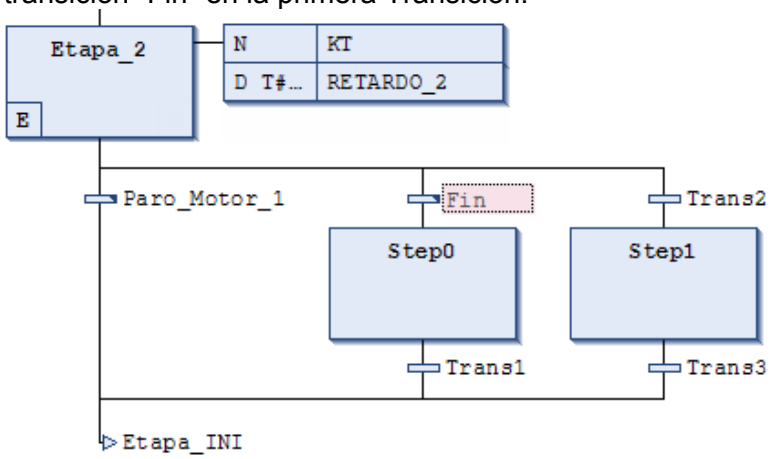
Cuando aparece la ventana le llamamos “Fin” y lenguaje “LADDER”



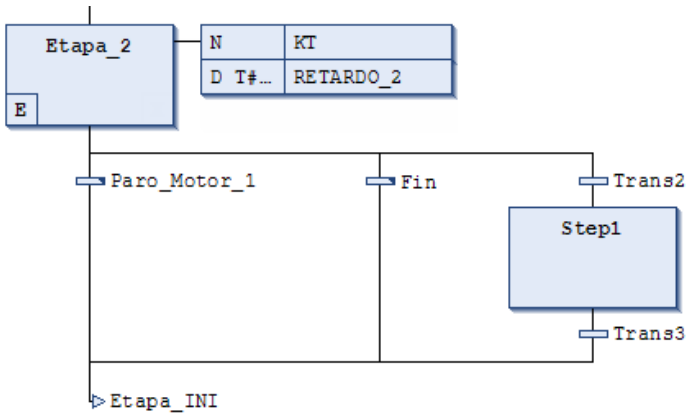
Damos a editar y escribimos lo siguiente, si “RETARDO_2” (Delay ha contado) y “VECES”>=5, se activa la variable “Fin”.



Escribimos la transición “Fin” en la primera Transición.

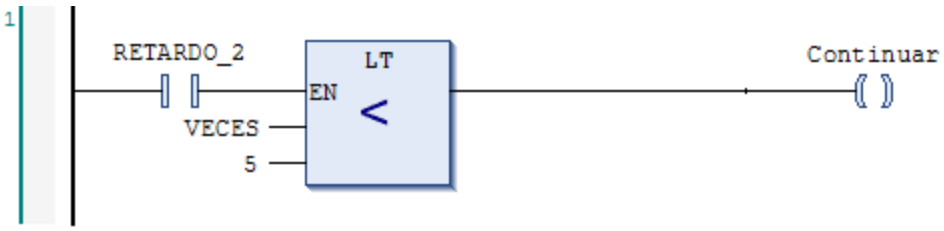


Y eliminamos la Step0 y la Trans_1, quedando de momento de la siguiente manera:

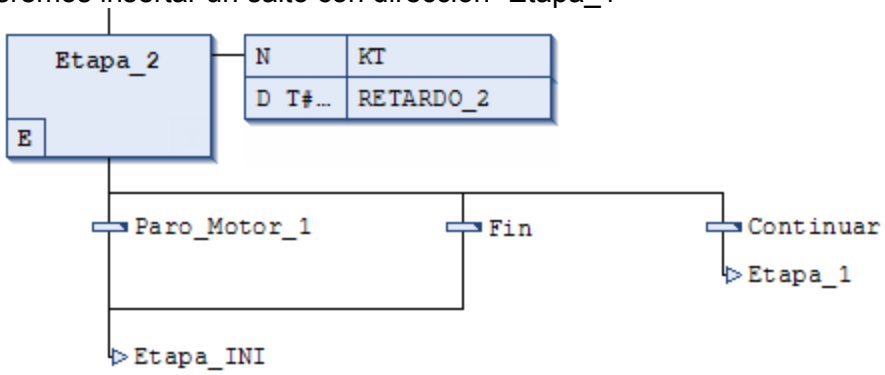


Bien ahora procedemos a declarar la transición de continuar, que se daría si ha contado el temporizador RETARDO_2 y “VECES”<5.

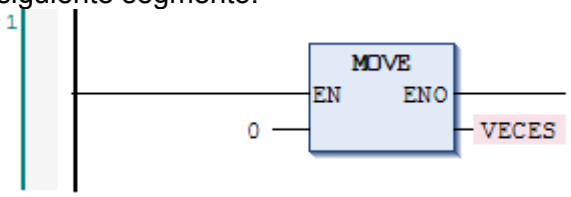
Insertamos una transición a la que llamaremos “CONTINUAR” y realizamos el siguiente segmento.



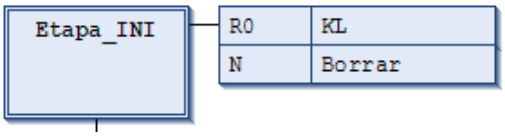
Para esta transición no vamos a la “Etapa_INI” sino que vamos a la “Etapa_”1, por lo que deberemos insertar un salto con dirección “Etapa_1”



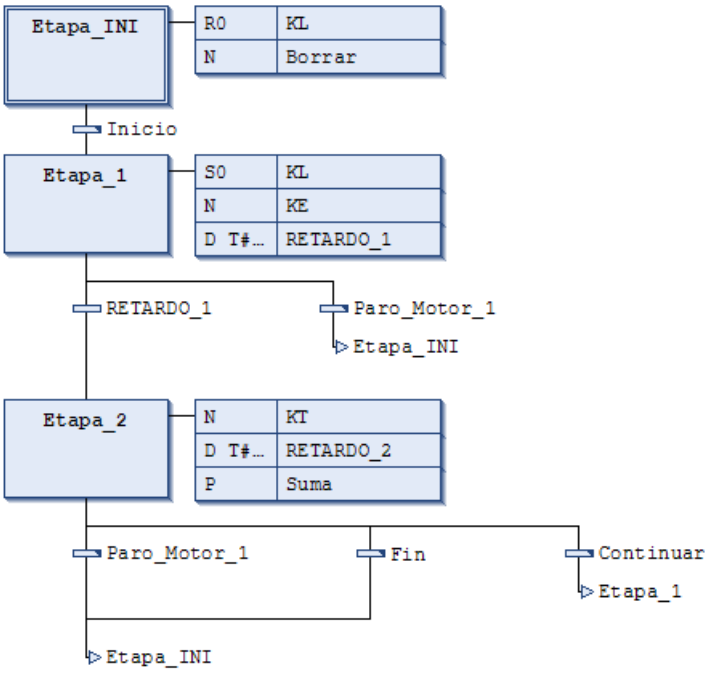
Por último y para poder volver a realizar el proceso deberemos de poner la variable “VECES” a 0 en la “Etapa_INI”. Añadimos una acción a la que llamaremos “Borrar”, donde crearemos el siguiente segmento:



Insertamos la acción dentro de la “Etapa_INI” una acción, quedando de la siguiente manera.



El “POU” quedará como vemos a continuación:



Por último si ponemos la simulación veremos como se realiza el proceso 5 veces y se para.

CAPÍTULO

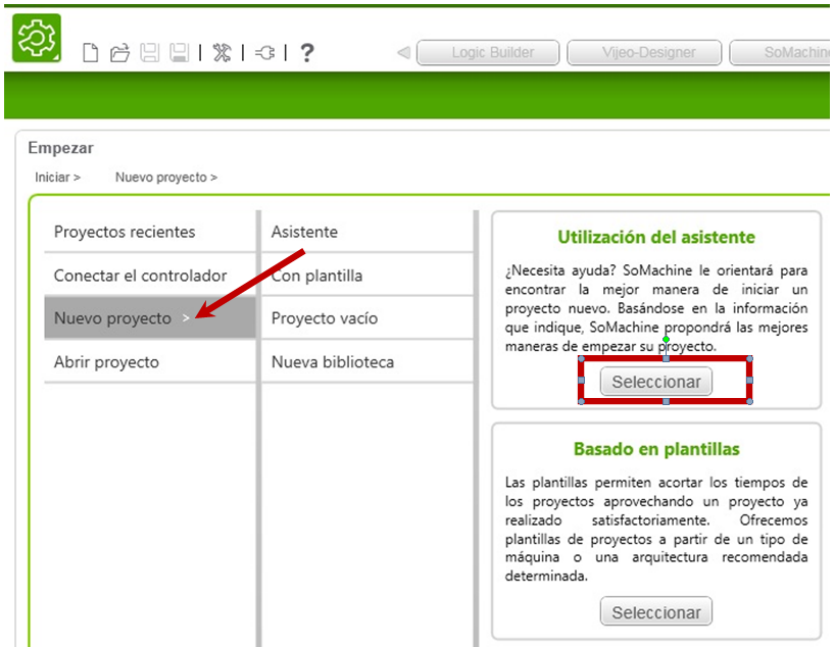
5

5. PRACTICAS

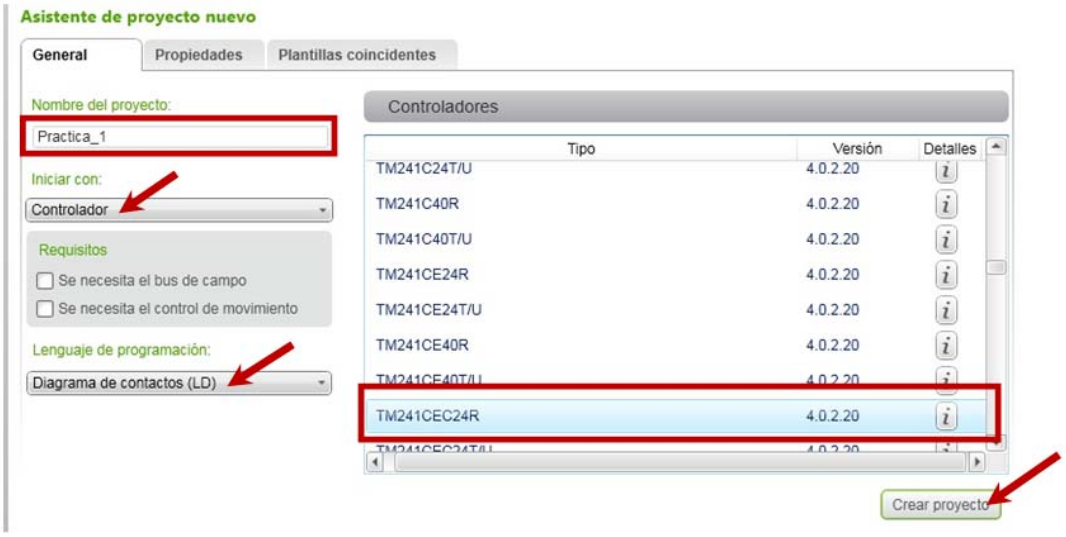
5.1 .- PRACTICA 1: MARCHA/PARO MOTOR

Vamos a realizar la práctica de un marcha/paro de motor.

Primero de todo tendremos que crear **un 'Nuevo proyecto'**, en este caso utilizaremos el asistente que nos ayudará a crear el proyecto y los primeros pasos de este.

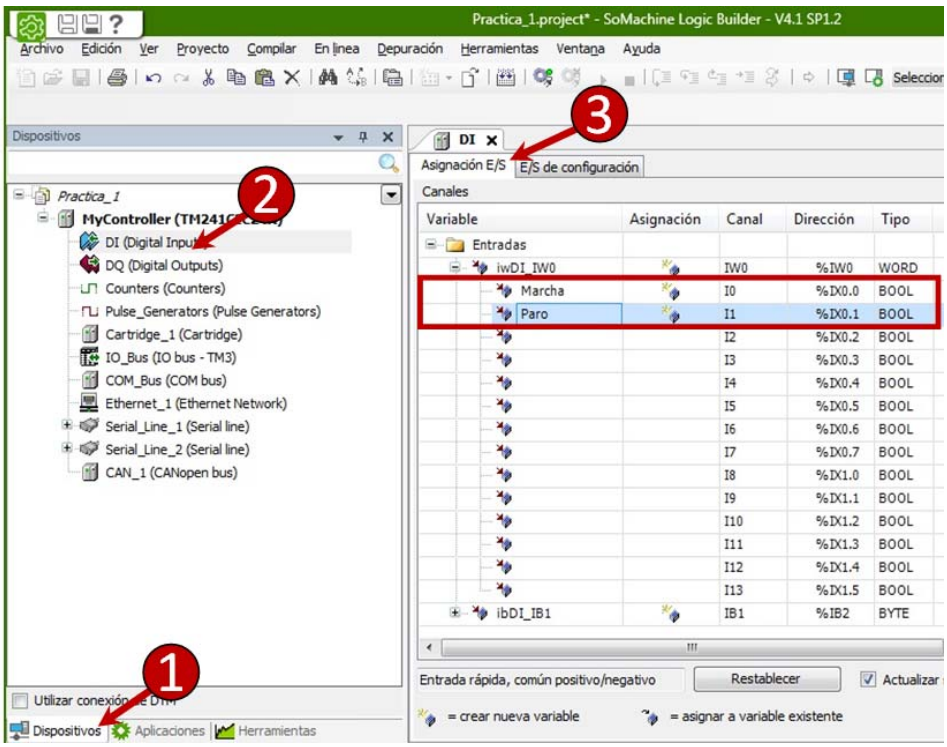


En la ventana del asistente de creación del proyecto dentro de la pestaña **'General'**, escribiremos el nombre de nuestro proyecto *(Practica_1)*, en **'Iniciar con'** seleccionaremos **'Controlador'** y en la parte de la derecha buscaremos el controlador deseado *(en este caso TM241CEC24R)*, por último en el '**lenguaje de programación'** seleccionamos el lenguaje con el queremos programar el primer POU *(en este caso Diagrama de contactos LD)*. Por último pulsamos '**Crear Proyecto'** para empezar.

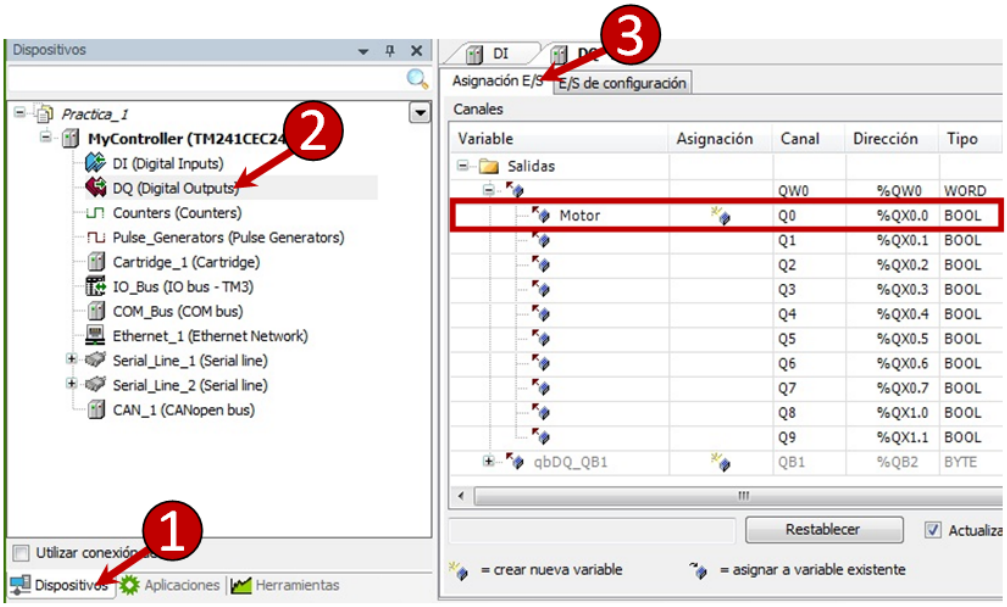


Una vez ha aparecido la ventana principal del SoMachine (*SoMachine Central*), nos aparece el Flujo de trabajo típico de un proyecto automatización, para entrar en la ventana de programación tendremos que pulsar el botón de **'Controlador'**, ya que dentro del flujo la configuración del controlador ya la hemos realizado previamente con el asistente.

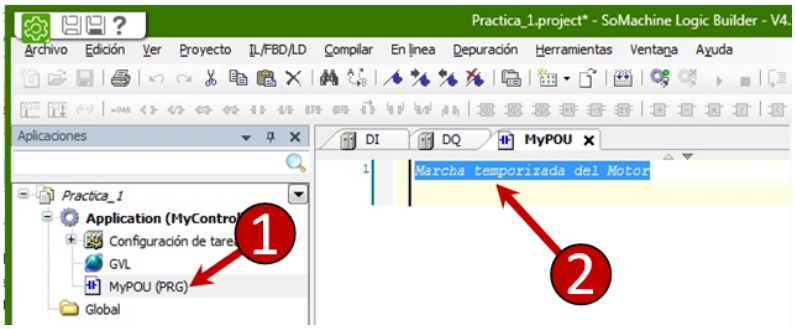
Una vez entramos en la ventana de programación *(Logic Builder),* lo primero que tendremos que realizar, es la configuración de la entradas/salidas físicas que vamos a utilizar, para ello, tendremos que ir a la pestaña de **'Dispositivos'** del navegador del proyecto, que se encuentra en la parte izquierda de la pantalla. Una vez dentro, maximizaremos el controlador, y haremos doble click sobre las entradas digitales '**DI (Digital Inputs)'**. En el área de trabajo aparecerá las pestañas de configuración de las entradas digitales que hay embebidas en el M241, seleccionamos la pestaña **'Asignación E/S'** y escribimos en la entrada **'I0' 'Marcha'** y en la **'I1' 'Paro'**.



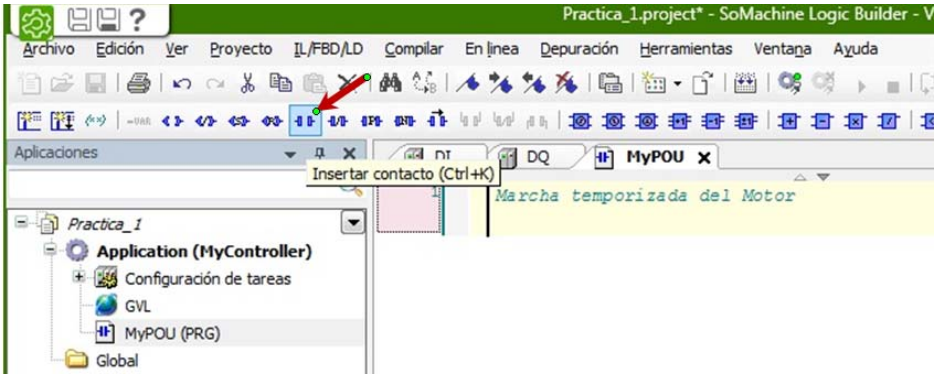
Ahora entraremos en la configuración de las salidas físicas, tendremos que ir a la pestaña de **'Dispositivos'** del navegador del proyecto, y haremos doble click sobre las salidas digitales '**DQ (Digital Outputs)'**. En el área de trabajo aparecerá las pestañas de configuración de las entradas digitales que hay embebidas en el M241, seleccionamos la pestaña **'Asignación E/S'** y escribimos en la salida **'Q0' 'Motor'**.



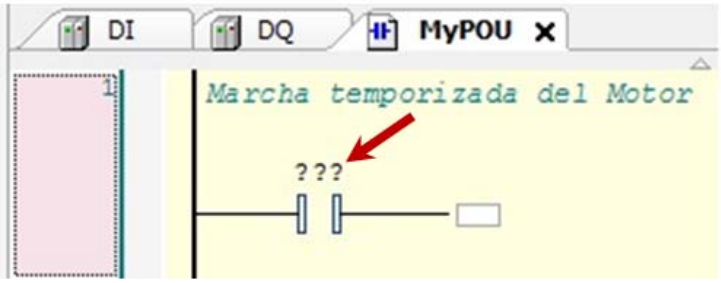
Una vez configurados las entradas y salidas, pasamos a programar seleccionando la pestaña **'Aplicación'** del navegador de proyectos. Desplegamos la '**Application'** y hacemos doble click sobre **'MyPou'**, que es el POU que ya ha creado previamente el asistente al crear el proyecto. En área de programación hacemos doble clic en la parte de arriba del primer segmento para añadir un comentario a este.



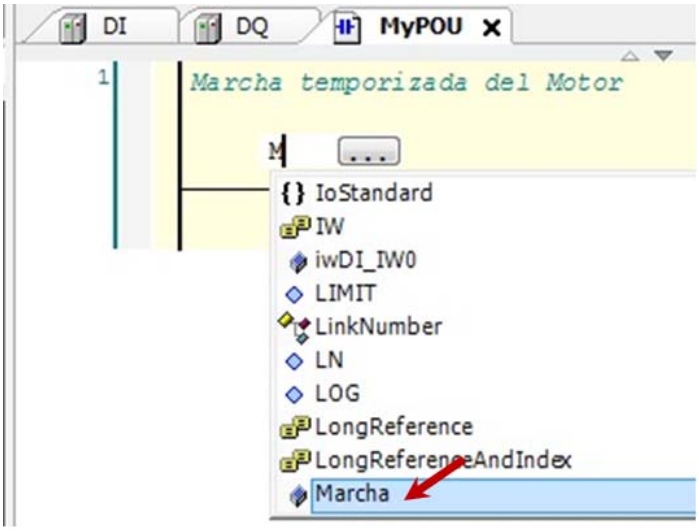
Dentro del área de programación, seleccionamos el segmento y se habilitará la barra de herramientas de programación LD, que se encuentra justo arriba. Buscamos el contacto NO y pulsamos, aparecerá en el área de programación.



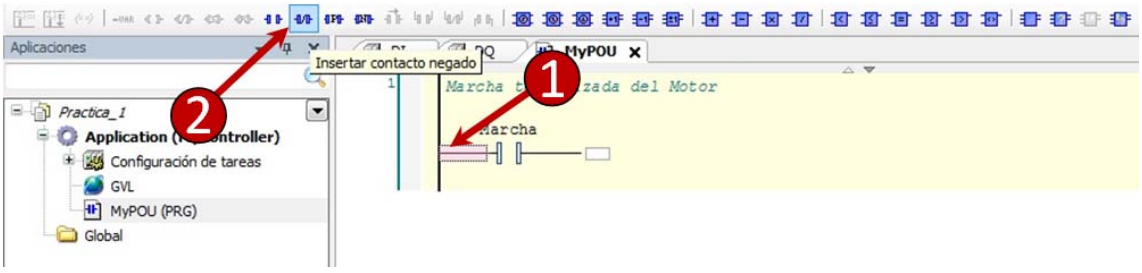
En el contacto NO que hemos colocado, pulsamos sobre los símbolos **'???'** y escribimos el símbolo de la variable que queremos asignar a este contacto.



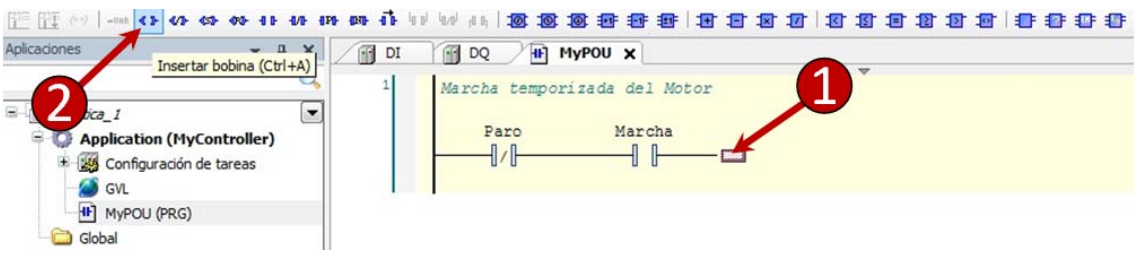
Cuando empezamos a escribir nos aparece una lista de las variables previamente creadas, en este caso seleccionamos de la lista la variable de '**Marcha'**.



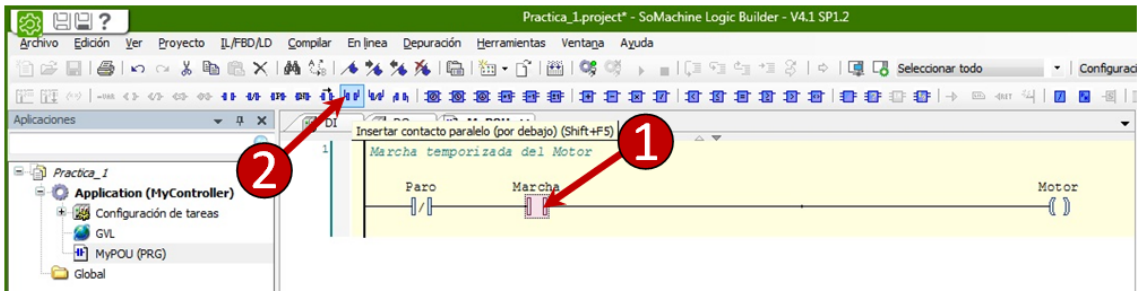
Una vez añadido el contacto NO de marcha seleccionamos la rama delante de este, y añadimos un contacto **'NC'**. Al que vincularemos con la entrada I1 de '**Paro'**.



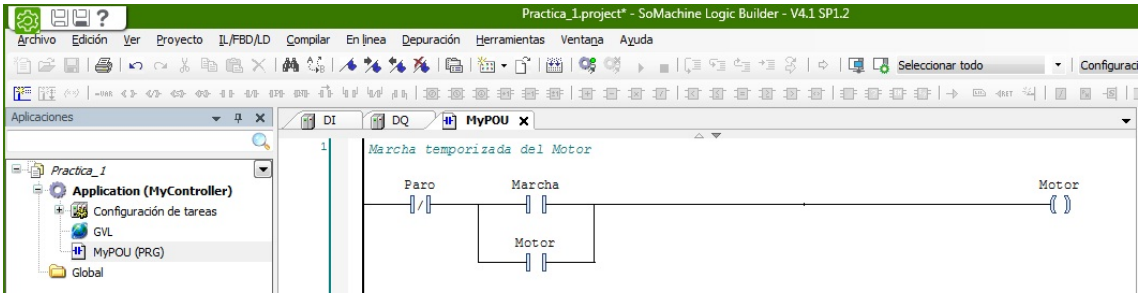
Ahora pulsamos en el recuadro resaltado de la parte izquierda del contacto NO de marcha y añadimos de la barra de herramientas de programación LD una **'bobina'**. Y la asociamos a la salida digital Q0 **'Motor'**.



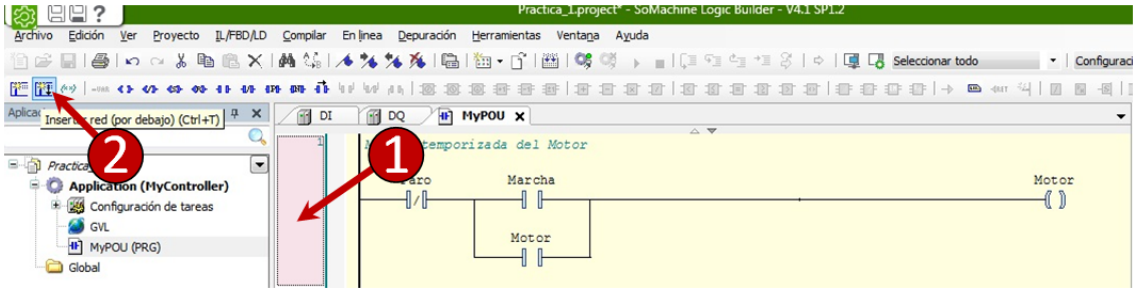
Una vez cerrada la línea de escalón 1, seleccionamos nuevamente el contacto NO de **'Marcha'** e insertamos de la barra un contacto paralelo (por debajo) y lo asignamos a la variable **'Motor'** para que el motor quede enclavado.



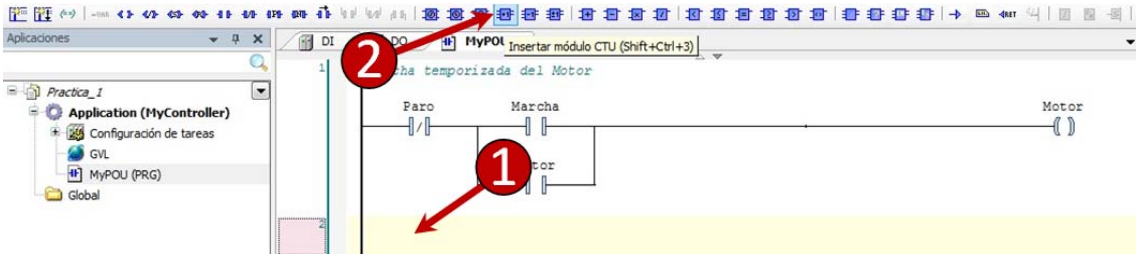
El primer escalón tiene que quedar de la siguiente manera:



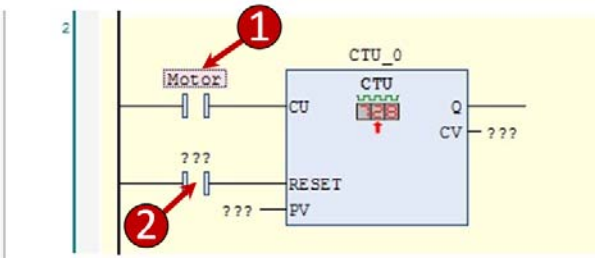
Ahora seleccionamos la parte izquierda del escalón 1 y en la barra de herramientas de programación LD, seleccionamos **'Insertar red (por debajo)**' para añadir un nuevo escalón para continuar con la programación.



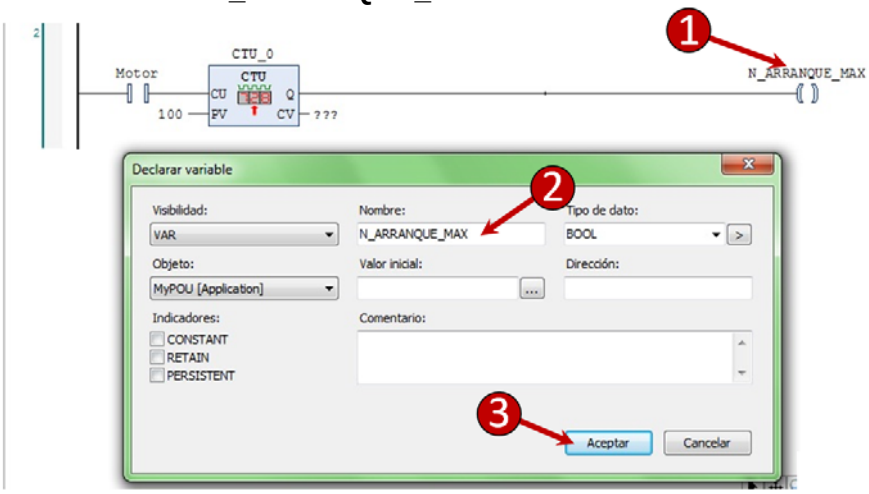
Después de insertar la red tenemos que añadir un contador CTU '**Insertar módulo CTU**'.



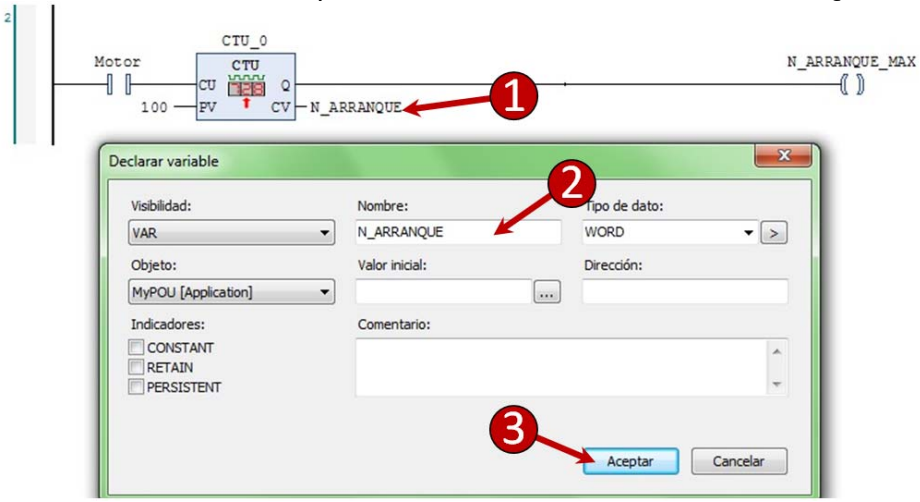
En el pin de contaje **‘CU’** del contador, seleccionamos el contacto NO y lo asociamos a la variable **‘Motor’** (de esta manera cada vez que el motor de arranque se incrementará el número de maniobras). Ahora borramos el contacto que ha aparecido en el pin de **‘RESET’** del contador. Por último ponemos un valor de **100** en el pin **‘PV’** para indicar el valor de preselección del contador.



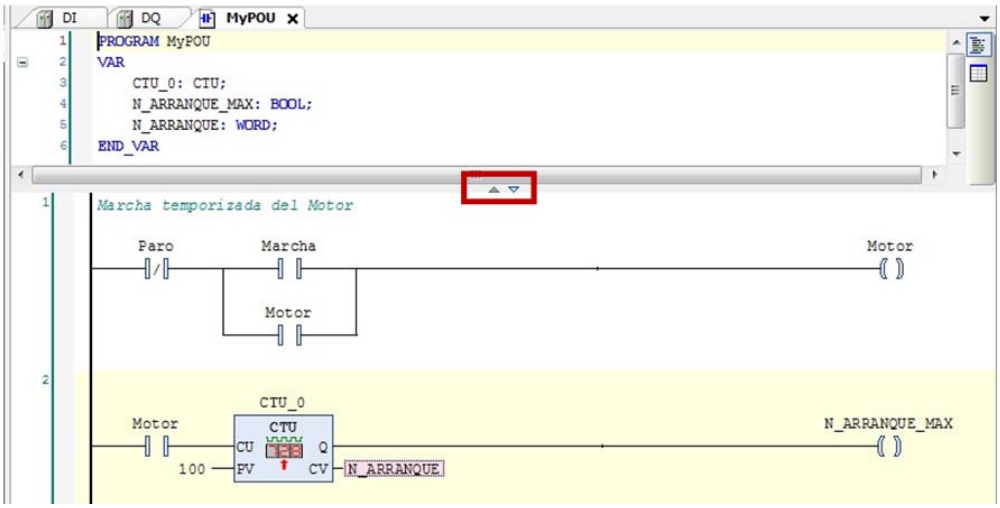
Ahora añadimos una bobina al pin de salida **‘Q’** del contador y creamos una variable booleana nueva llamada **‘N_ARRANQUE_MAX’**.



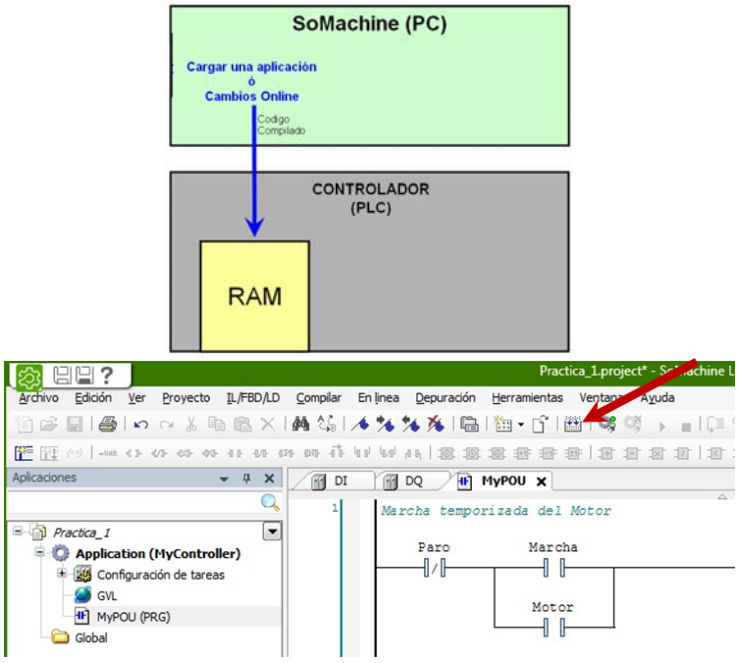
Ahora seleccionamos el pin de salida **‘CV’** del contador que nos indica el valor de contaje y creamos una variable tipo **‘WORD’** con el nombre **‘N_ARRANQUE’**.



Para asegurar que se han creado las variables pulsamos el triangulo que se encuentra en la parte superior del área de programación y se desplegará la zona de declaración de variables internas del POU.

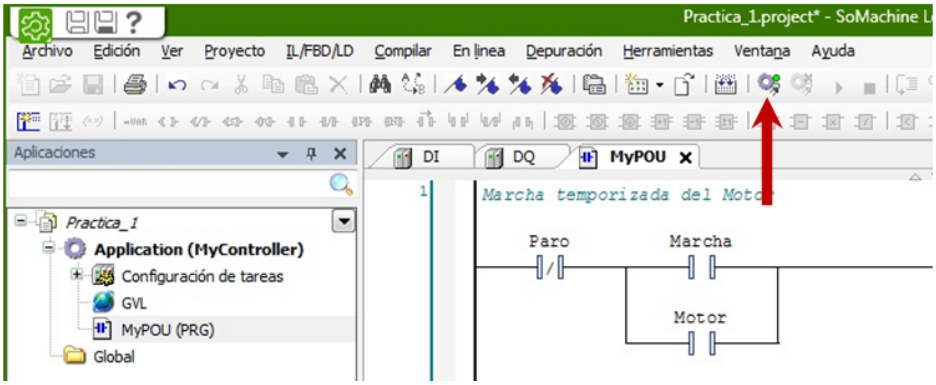


Cuando se descarga el programa al PLC, este se compila y luego se carga en la memoria RAM del PLC. Que no es una memoria volátil.



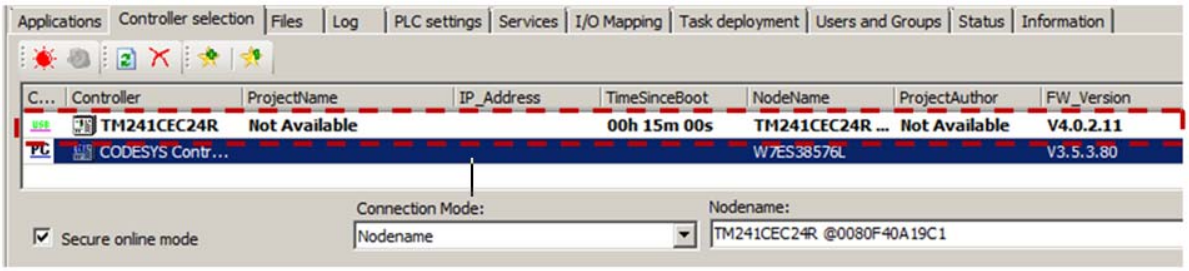
La opción de **'Compilar'** no genera el código de programa. La opción compilar sólo verifica que la aplicación no contiene errores. Generar código de máquina se crea cuando se realiza la descarga de la aplicación.

Inicialmente se tiene que tener conectado con el cable de programación el PC al controlador (cable de programación **USB - Mini USB** para el controlador M241).

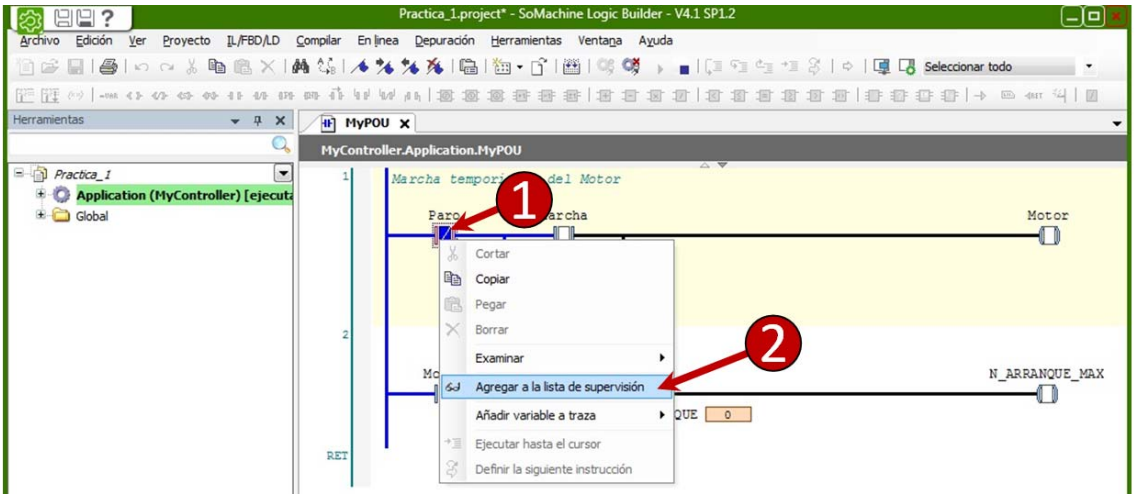


Si queremos cambiar de controlador al que conectarse ó de método de conexión, podremos cambiarlo haciendo doble clic sobre esa línea, que aparecerá ahora resaltado en negrita.

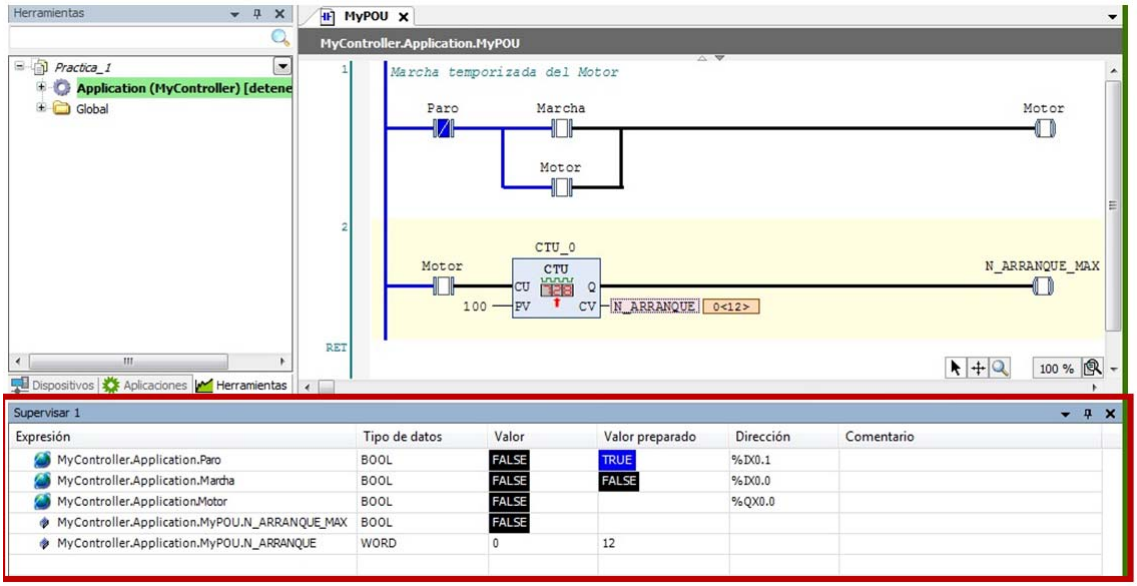
Una vez tenemos elegido el controlador al que nos queremos conectar (resaltado en negrita), para conectarnos solo tenemos que ir al menú textual **'Online >> Login'** ó pulsar en el icono de Login  de la barra de herramientas.



Cuando estemos conectados y hayamos puesto el controlador en **'RUN'**, podemos monitorizar online el programa, si se desea generar una tabla de supervisión de las diferentes variables, las podemos seleccionar en el POU en online, pulsar botón derecho y seleccionar la opción **'Agregar a la lista de supervisión'**.

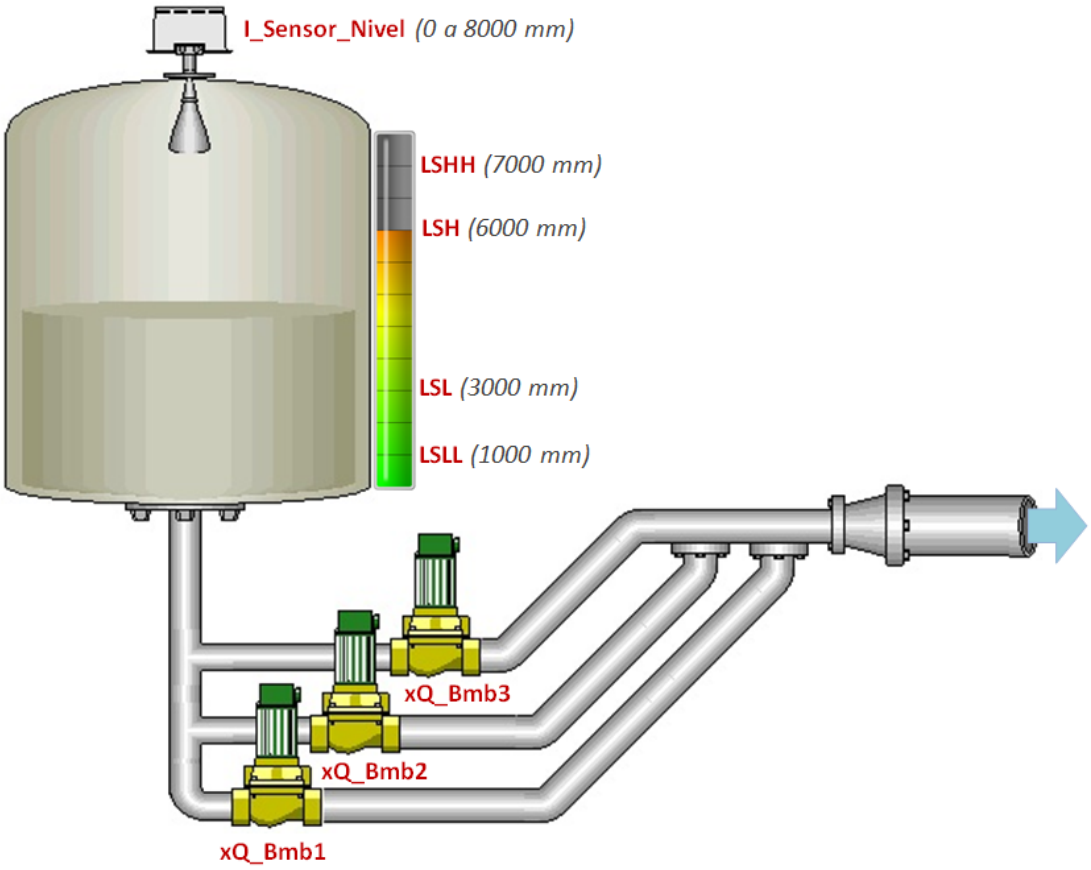


En la tabla de **‘supervisión’**, además de poder monitorizar el valor de las diferentes variables, también podemos en la columna **‘Valor Preparado’** escribir el valor deseado para esa variable (dependiendo del tipo de variable) y escribir el valor pulsado la combinación de teclas **‘CTRL + F7’** o forzar el valor pulsando únicamente la tecla **‘F7’**.



5.2.- PRACTICA 2: BOMBEO + VISUALIZACIÓN SENCILLA

Queremos automatizar un bombeo de entrada a una depuradora, este depósito dispone de un sensor de nivel por ultrasonidos que tiene una salida analógica que nos da **0 a 8000 mm**. Se tiene que automatizar la activación de tres bombas, en función del nivel que haya en el depósito.



Operación:

Se crearán unos auxiliares de nivel en función del nivel de agua que nos del sensor de nivel:

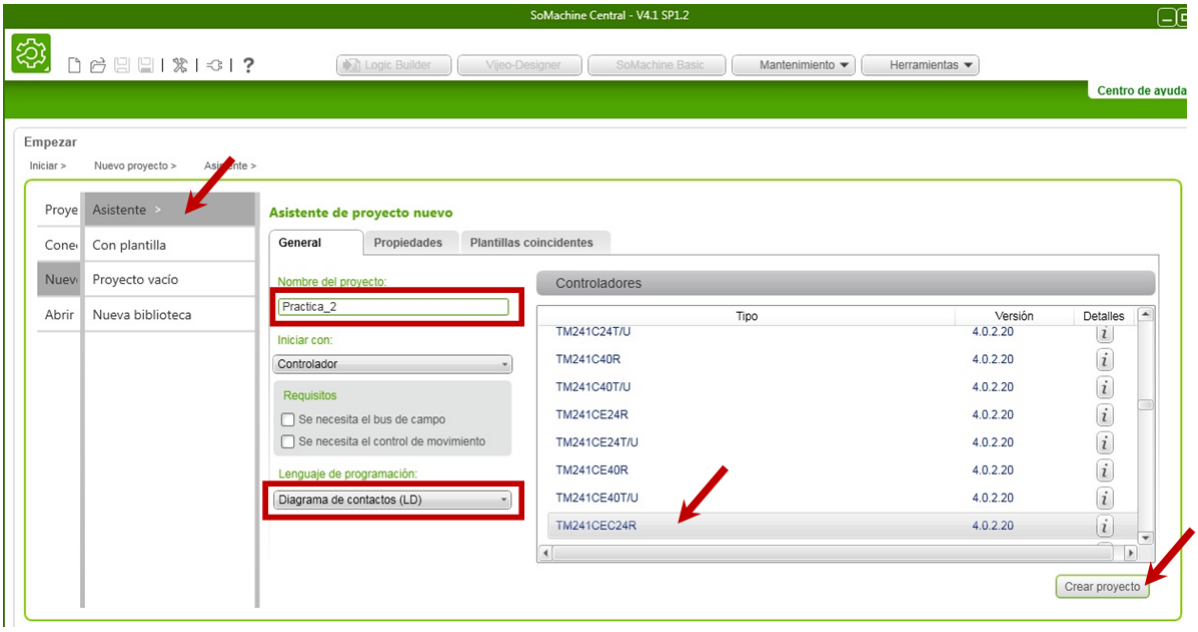
- **x_Aux_LSLL** se activará si **Sensor_Nivel > 1000 mm**
- **x_Aux_LSL** se activará si **Sensor_Nivel > 3000 mm**
- **x_Aux_LSH** se activará si **Sensor_Nivel > 6000 mm**
- **x_Aux_LSHH** se activará si **Sensor_Nivel > 7000 mm**

Cuando el nivel de agua es mayor que el nivel L, se encenderá la bomba 1. Si se activa el sensor de nivel H la bomba 2, también tiene que ponerse a bombear. Si el nivel de agua continúa creciendo y llega al sensor HH, la bomba 3 también se activará.

Si el nivel del agua decrece y el sensor de nivel HH deja de dar señal, ninguna bomba tiene que parar. Cuando el nivel del agua baja del sensor H se para la tercera bomba al cabo de 10 segundos de que haya perdido el nivel. Si el nivel sigue bajando y se pierde la señal de nivel L se para la bomba secundaria al cabo de 5 segundos y finalmente la bomba primaria se parará cuando la señal del sensor LL dejé de dar señal.

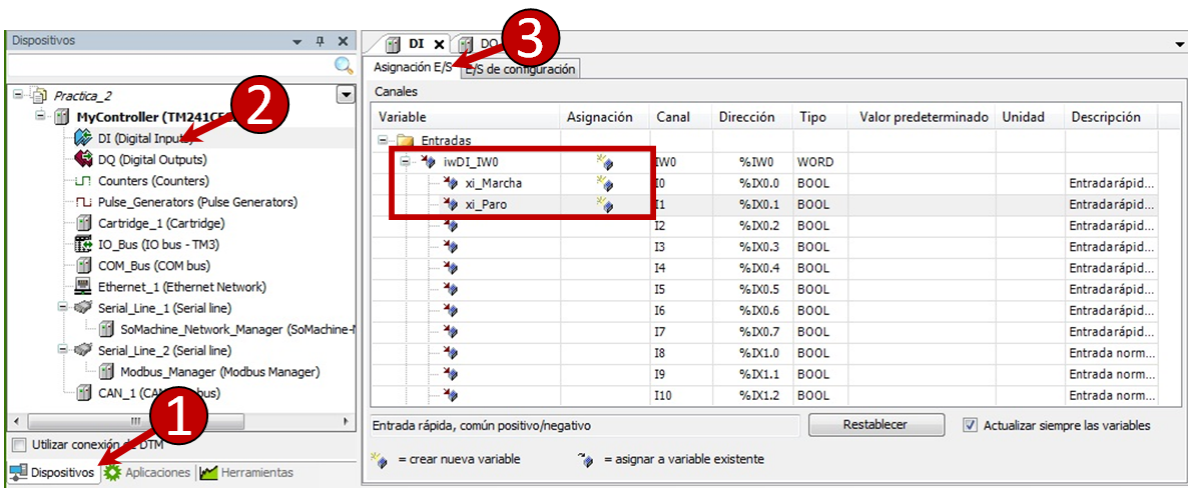
5.2.1 Programación de la práctica

En la ventana del asistente de creación del proyecto dentro de la pestaña **'General'**, escribiremos el nombre de nuestro proyecto *(Practica_2)*, en **'Iniciar con'** seleccionaremos **'Controlador'** y en la parte de la derecha buscaremos el controlador deseado *(en este caso TM241CEC24R)*, por último en el '**lenguaje de programación'** seleccionamos el lenguaje con el queremos programar el primer POU *(en este caso Diagrama de contactos LD)*. Por último pulsamos '**Crear Proyecto'** para empezar.

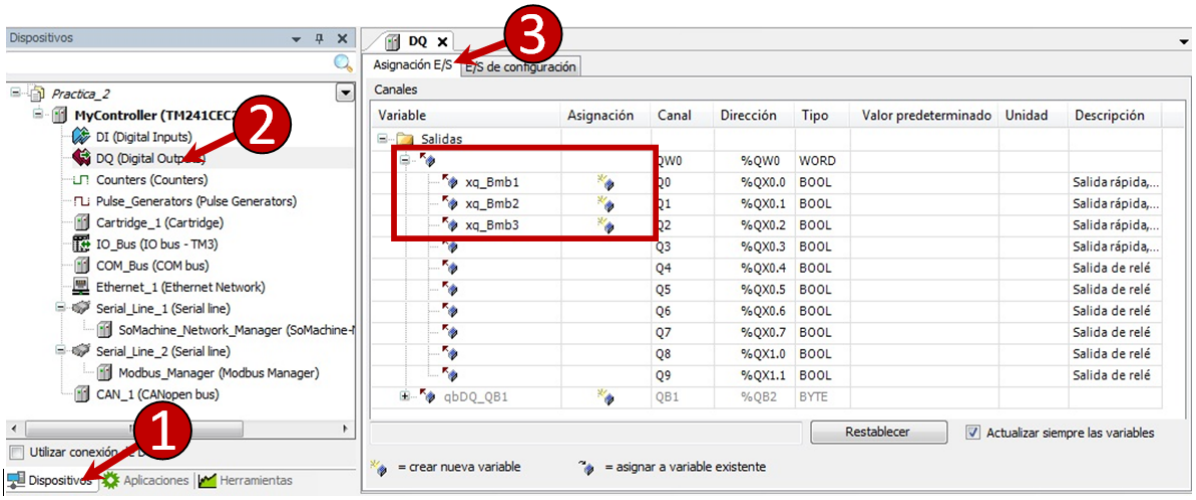


Una vez ha aparecido la ventana principal del SoMachine (*SoMachine Central*), seleccionamos el **'Logic Builder'** en la barra de acceso rápido de los softwares.

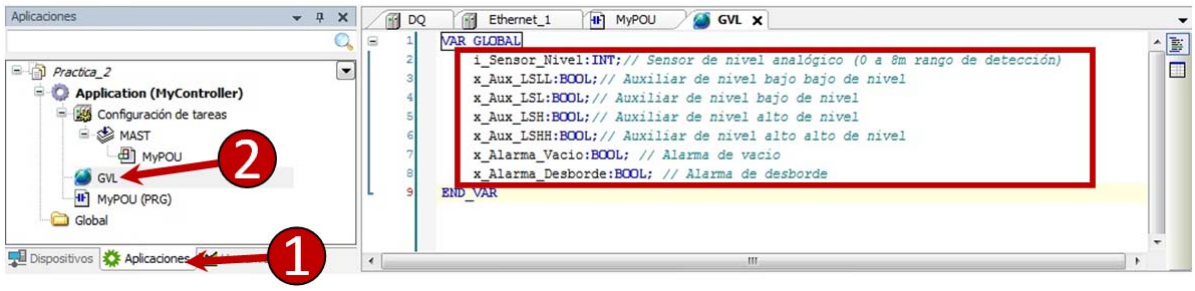
En la ventana de programación *(Logic Builder),* lo primero que tendremos que realizar, es la configuración de la entradas/salidas físicas que vamos a utilizar, para ello, tendremos que ir a la pestaña de **'Dispositivos'** del navegador del proyecto, que se encuentra en la parte izquierda de la pantalla. Una vez dentro, maximizaremos el controlador, y haremos doble click sobre las entradas digitales '**DI (Digital Inputs)'**. En el área de trabajo aparecerá las pestañas de configuración de las entradas digitales que hay embebidas en el M241, seleccionamos la pestaña **'Asignación E/S'** y escribimos en la entrada **'I0' 'Marcha'** y en la **'I1' 'Paro'**.



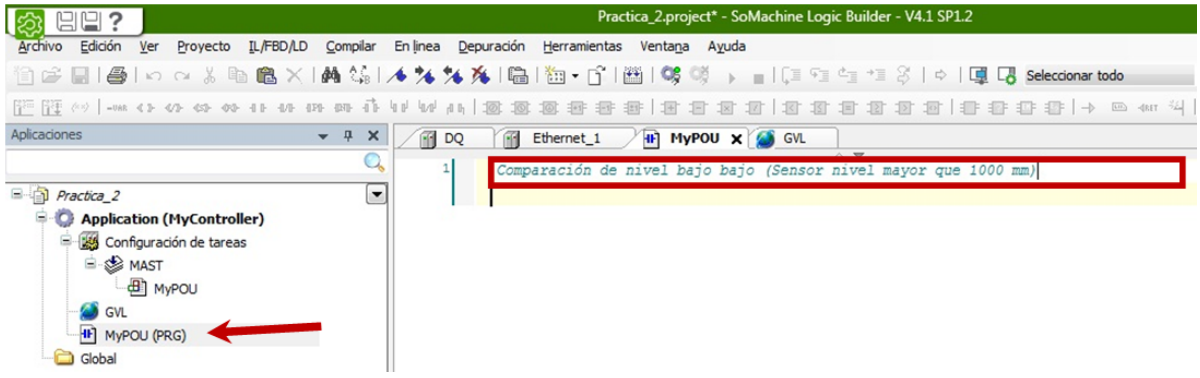
Ahora entraremos en la configuración de las salidas físicas, tendremos que ir a la pestaña de **'Dispositivos'** del navegador del proyecto, y haremos doble click sobre las salidas digitales '**DQ (Digital Outputs)'**. En el área de trabajo aparecerá las pestañas de configuración de las entradas digitales que hay embebidas en el M241, seleccionamos la pestaña **'Asignación E/S'** y escribimos las salidas **'Q0' 'xq_Bmb1'**, **'Q1' 'xq_Bmb2'** y **'Q2' 'xq_Bmb3'**.



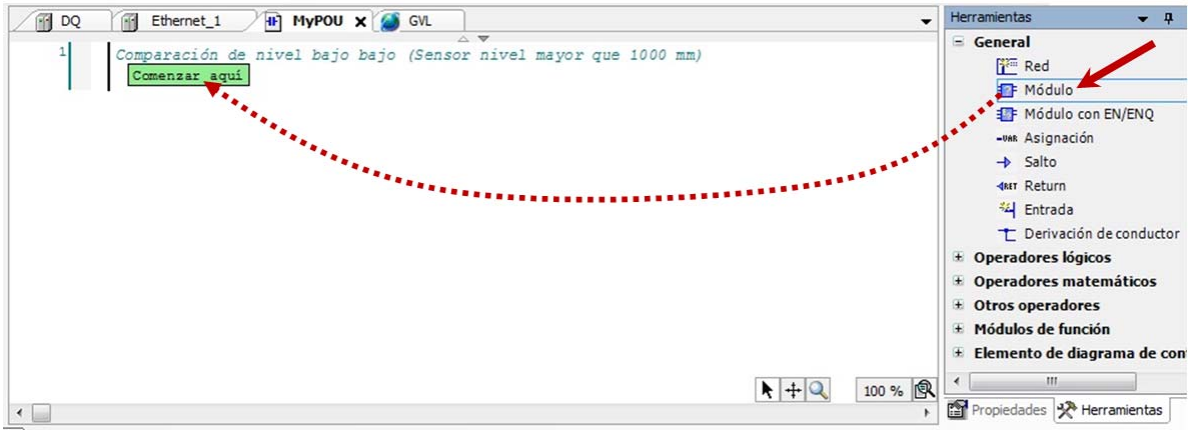
Una vez configurados las entradas y salidas, creamos las variables globales, seleccionando la pestaña **'Aplicación'** del navegador de proyectos. Desplegamos la '**Application'** y hacemos doble click sobre **'GVL'**, e introducimos las variables que necesitamos para la realización del proyecto.



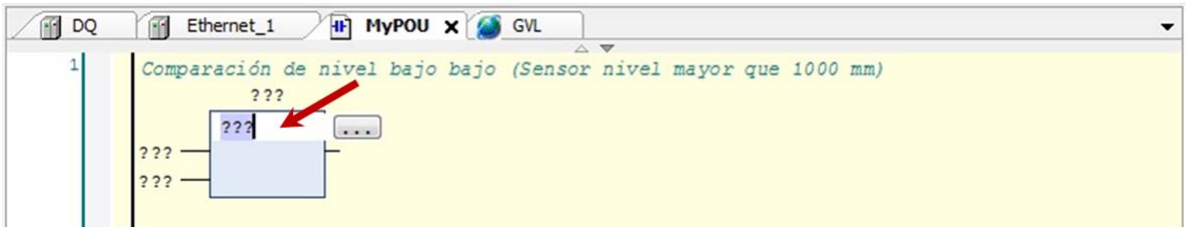
Una vez configurados las entradas y salidas, pasamos a programar seleccionando la pestaña **'Aplicación'** del navegador de proyectos. Desplegamos la '**Application'** y hacemos doble click sobre **'MyPou'**, que es el POU que ya ha creado previamente el asistente al crear el proyecto. En área de programación hacemos doble clic en la parte de arriba del primer segmento para añadir un comentario a este.



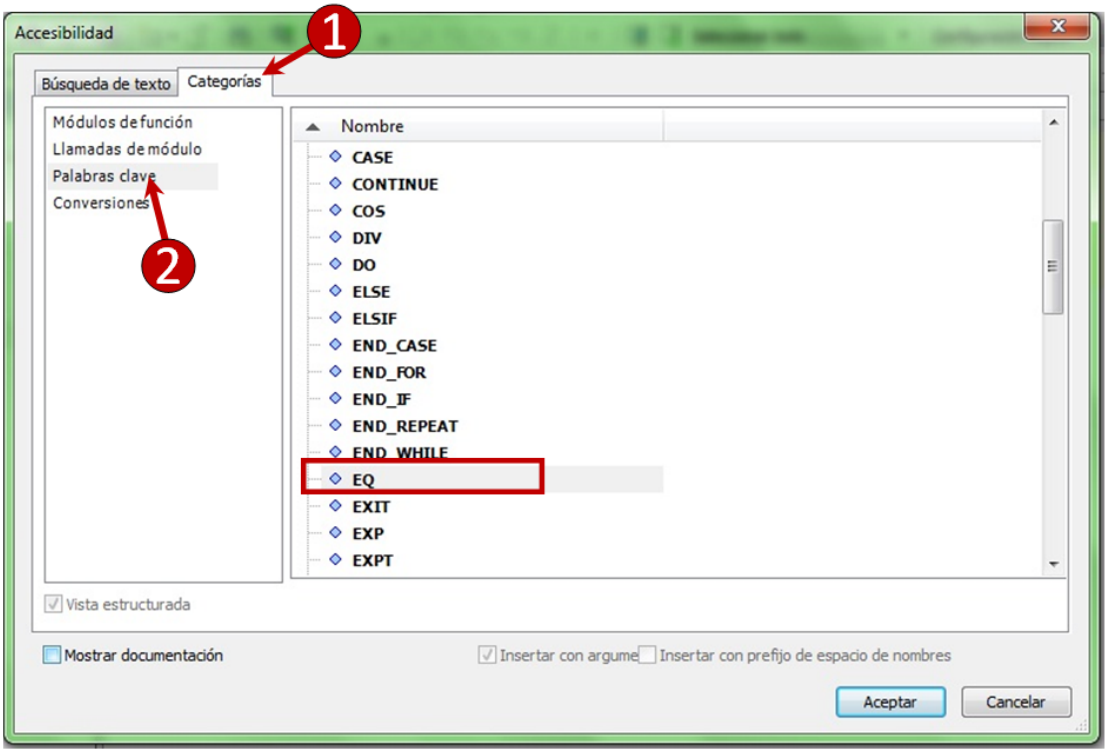
Dentro del área de programación, seleccionamos el '**Módulo'** del menú de herramientas, lo mantenemos pulsado y lo arrastramos hasta estar en el segmento para añadirlo.



En el módulo que hemos colocado, pulsamos sobre los símbolos '???' y que hay dentro del módulo y pulsamos sobre el botón '…' para ir a la ventana de 'Accesibilidad'.

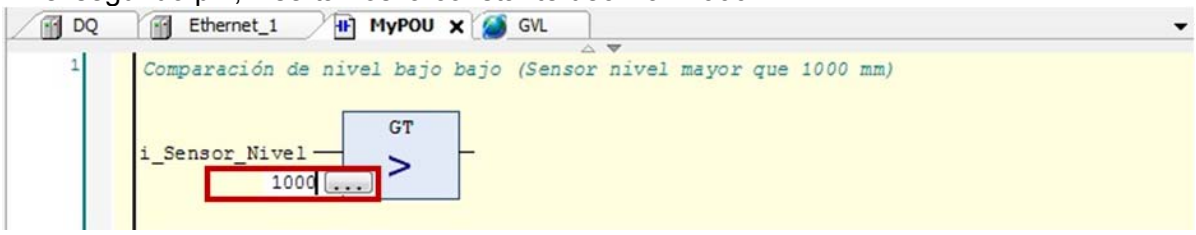


Ahora seleccionamos en la ventana de **'Accesibilidad'** nos vamos a **'Palabras clave'** y buscamos la instrucción de comparación '**GT'** (*greater than*).



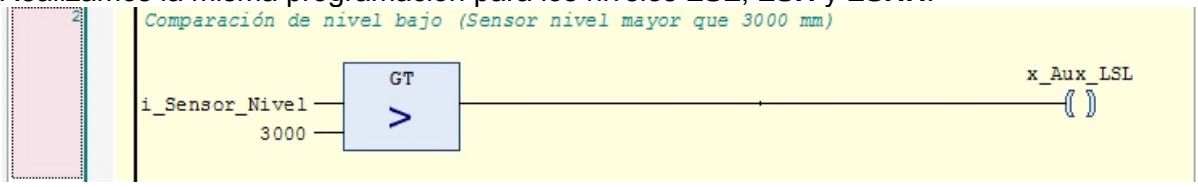
Una vez insertado la instrucción de comparación '**GT'** seleccionamos el primer pin y asociamos la variable **'i_Sensor_Nivel'.**

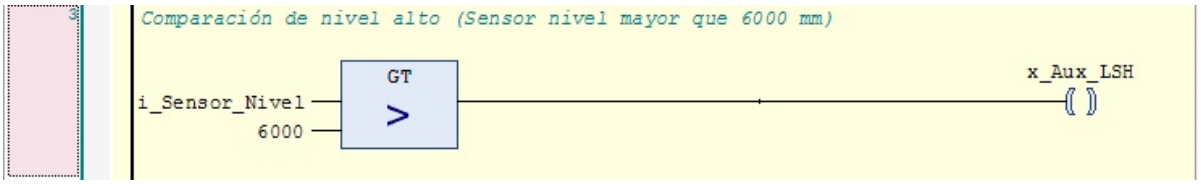
En el segundo pin, insertamos la constante decimal '1000'.

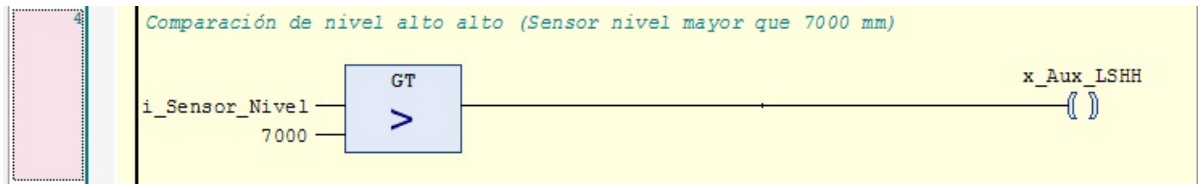


En el pin de salida añadimos una bobina de la barra de herramientas de ladder y le asociamos la variable **'x_Aux_LSLL'.**

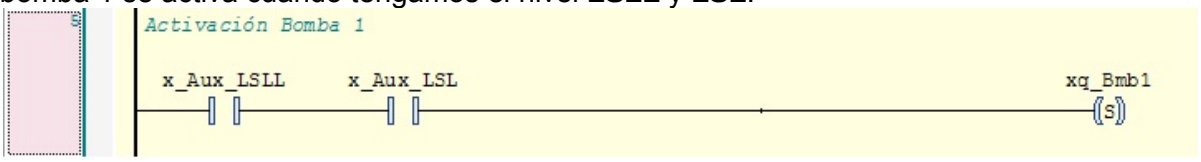
Realizamos la misma programación para los niveles **LSL**, **LSH** y **LSHH**:



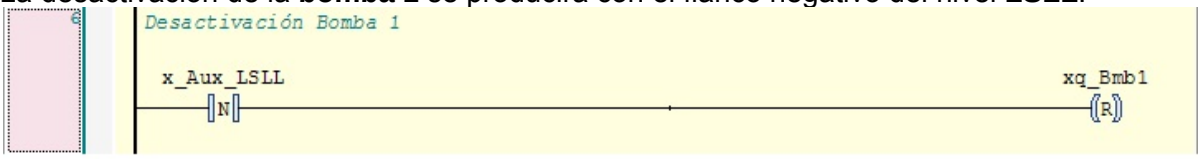




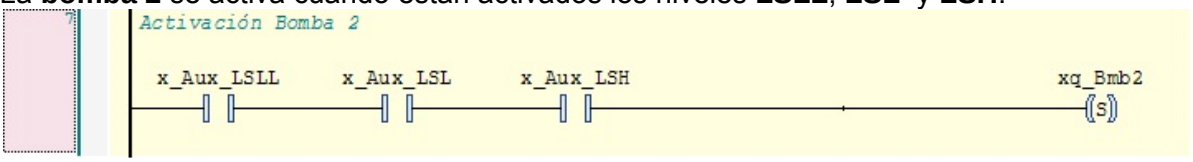
Añadimos una rama nueva y vamos a programar la activación de la **bomba 1**, la bomba 1 se activa cuando tengamos el nivel **LSLL** y **LSL**.



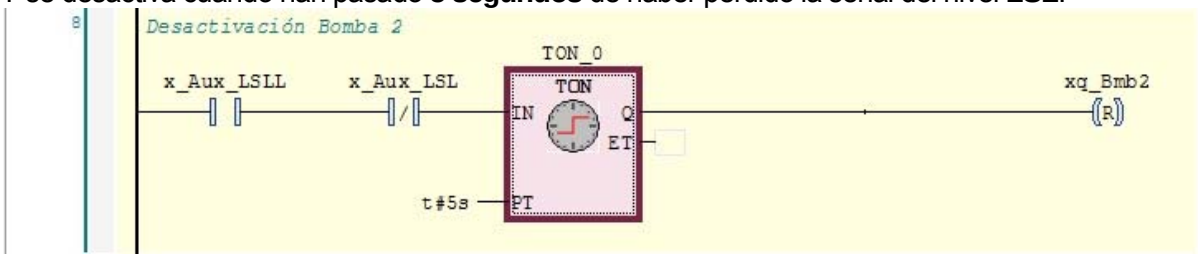
La desactivación de la **bomba 1** se producirá con el flanco negativo del nivel **LSLL**.



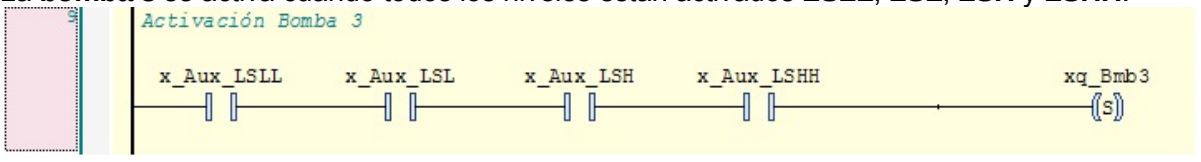
La **bomba 2** se activa cuando están activados los niveles **LSLL**, **LSL** y **LSH**.



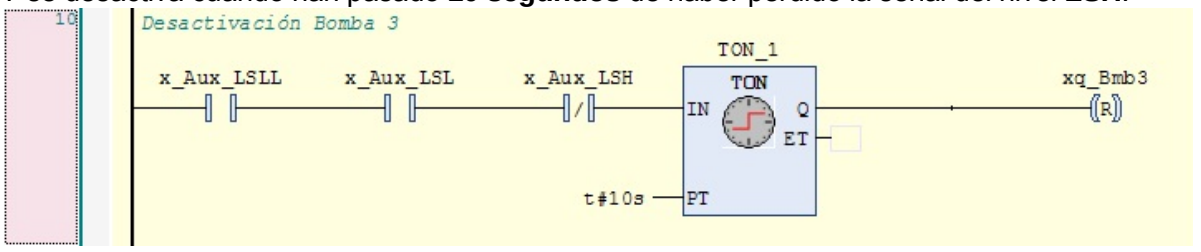
Y se desactiva cuando han pasado **5 segundos** de haber perdido la señal del nivel **LSL**.



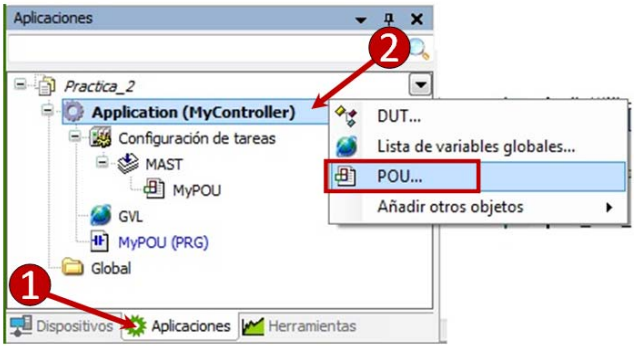
La **bomba 3** se activa cuando todos los niveles están activados **LSLL**, **LSL**, **LSH** y **LSHH**.



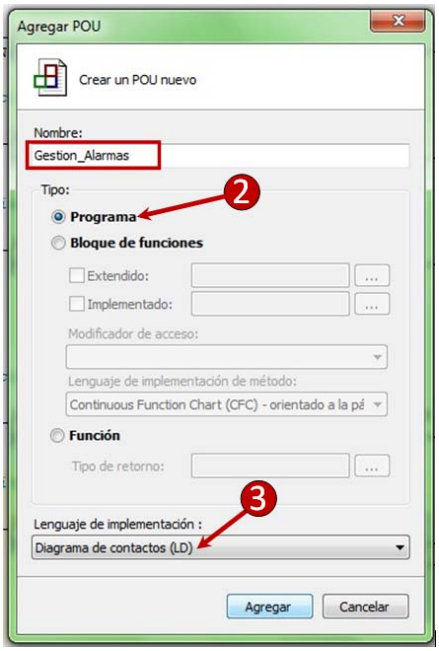
Y se desactiva cuando han pasado **10 segundos** de haber perdido la señal del nivel **LSH**.



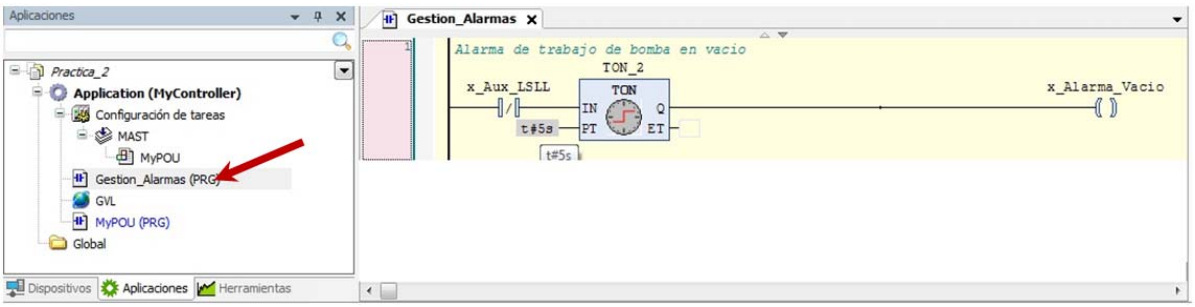
Una vez programado la secuencia de activación/desactivación de las bombas, seguimos en la pestaña **'Aplicación'** del navegador de proyectos. Desplegamos la '**Application'** y pulsamos el icono '+' que aparece al lado y en la ventana de opciones que aparece seleccionamos **'POU'**, para crea un nuevo POU.



Ahora escribimos en el campo **'Nombre'** el nombre que queremos a nuestro POU (*Gestion_Alarmas*), seleccionamos **'programa'** para el tipo de POU y por último decimos en que lenguaje de programación queremos que tenga (**LD**).

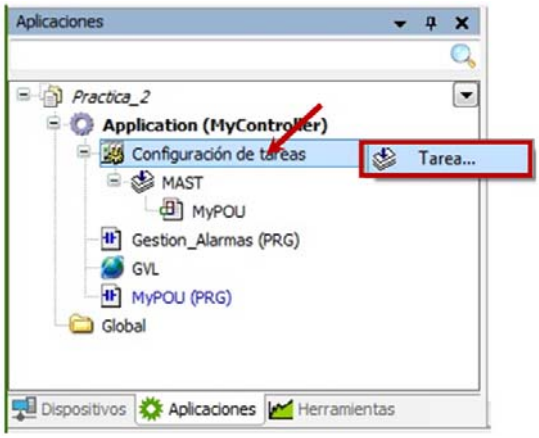


Después de crear el POU, hacemos doble click sobre él y programamos la alarma de vacío, en el primer segmento.

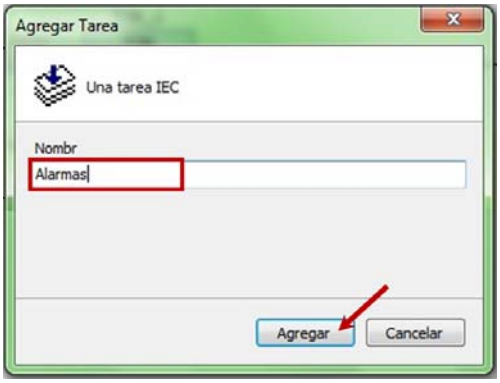


Creamos un segmento nuevo y ahora programamos la alarma de desborde.

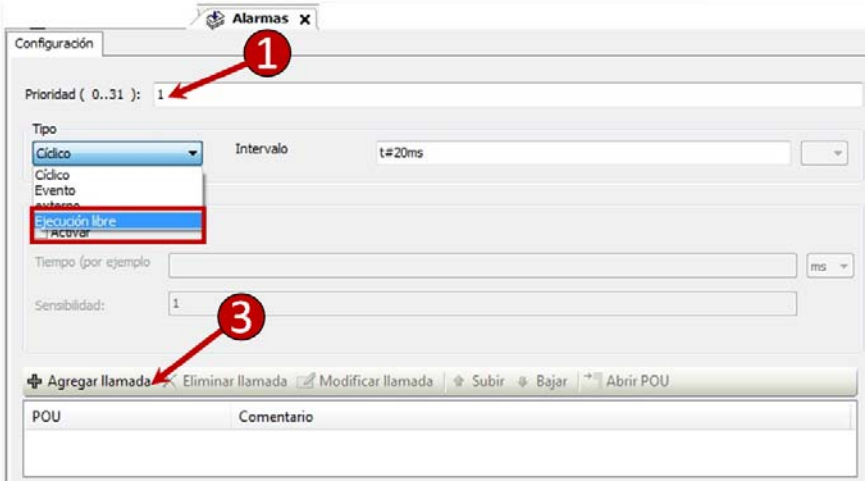
Ahora en la misma pestaña de **'Aplicaciones'**, seleccionamos el icono **'+'** que aparece al seleccionar **'Configuración de tareas'** y escogemos la opción de **'Tarea…'** para crear una nueva tarea.



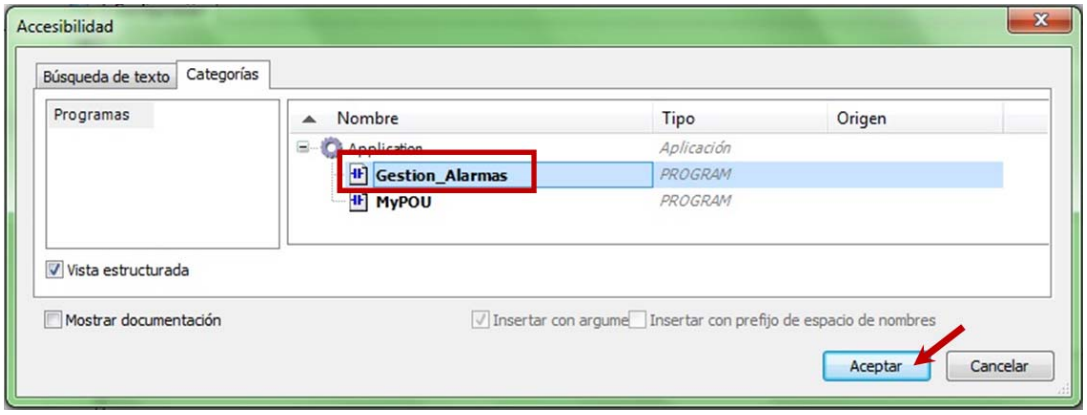
Le ponemos a la tarea el nombre **'Alarmas'**.



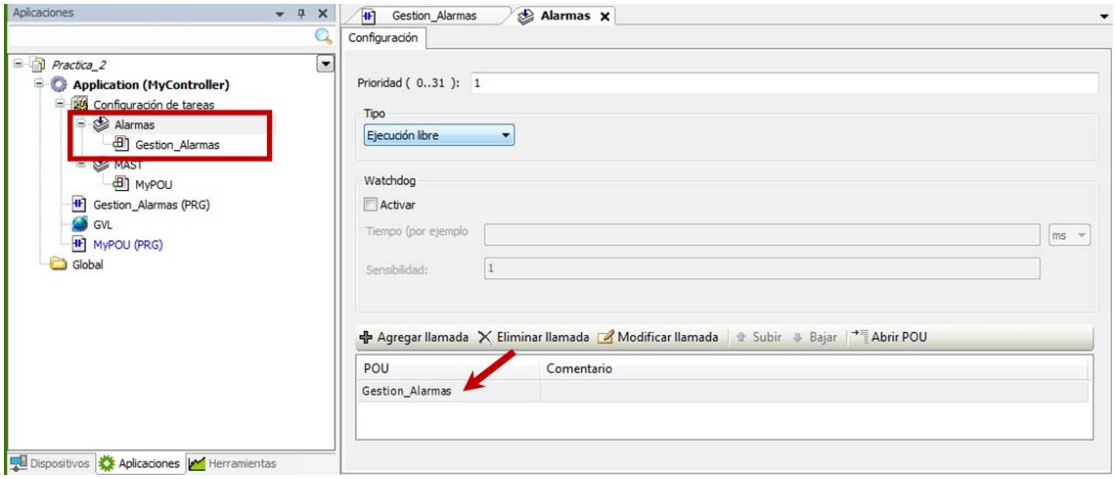
Ahora en la nueva tarea le asignamos una **'Prioridad'** elevada **'1'**, y en el tipo de tarea elejimos una tarea de **'Ejecución libre'**, ahora por último pulsamos '**Agregar llamada'** para sociar los POU's que se van ejecutar en esta tarea.



Asociamos el POU **‘Gestion_Alarmas’** a la tarea.



En la pestaña de aplicación del navegador, la tarea y la asociación del POU a la tarea quedan especificadas de la siguiente forma.

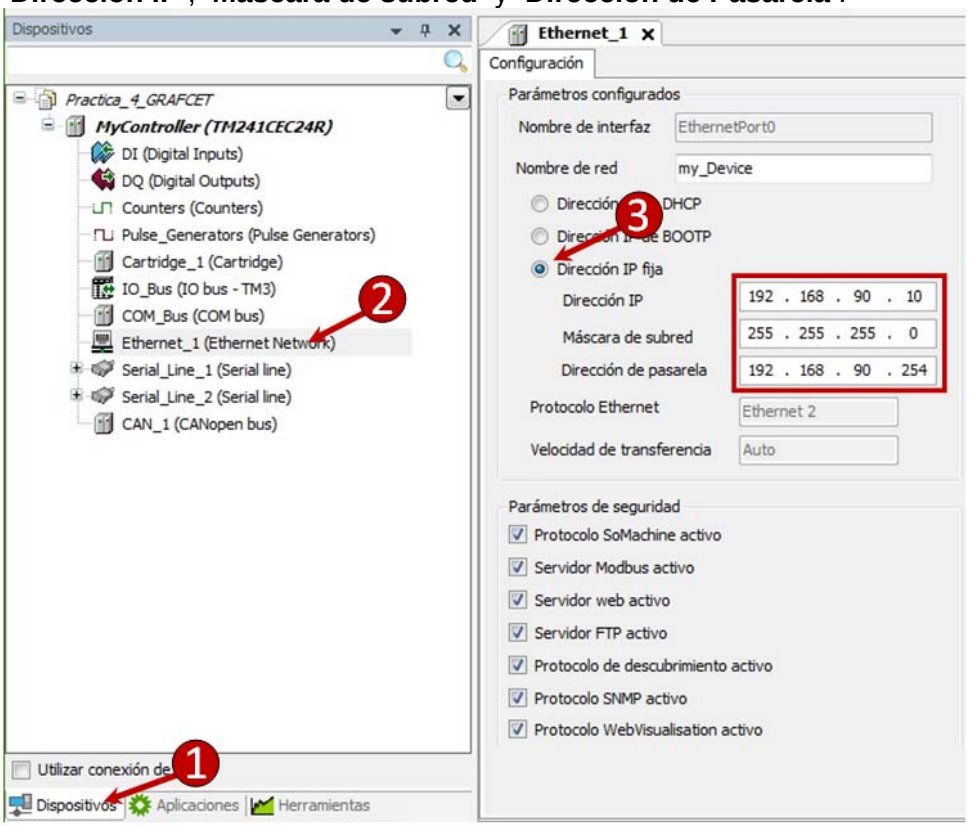


5.2.2 Creación de una visualización

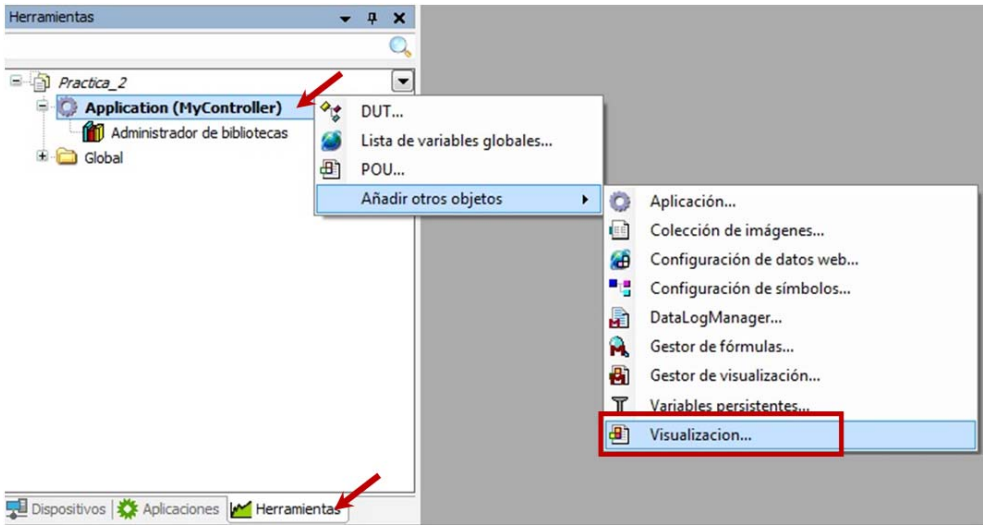
En el caso de los controladores **M241** y **M251** si disponen de Ethernet dispondrán de la webvisualizattion. Es una manera de publicar en un **webserver en HTML5** de las visualizaciones previamente creadas para ello se ha de proceder de la siguiente manera.
Para el ejemplo utilizaremos un controlador M241CEC24R con el puerto Ethernet integrado, este irá conectado a un router wifi TP LINK. Con la arquitectura siguiente:

1. Lo primero que haremos, es asignar la **IP: 192.168.2.20** al puerto de **'ethernet_1'** que se encuentra en la pestaña de **'Dispositivos'** del navegador de proyecto.

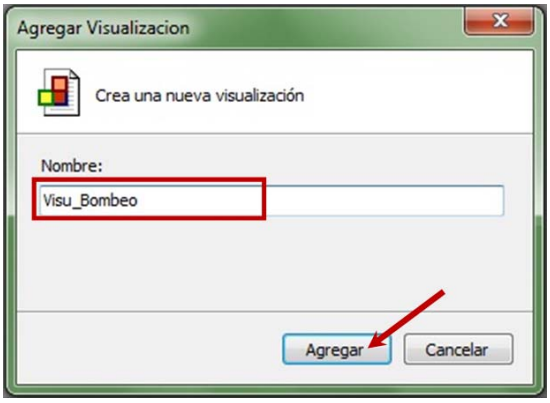
2. Para ello haremos doble click sobre el puerto '**Ehetnet_1(Ethernet network)',** en la ventana de configuración, estableceremos que la asignación de la IP es fija seleccionando la opción **'Dirección IP fija**' y luego rellenando los campos de **'Dirección IP'**, **'Máscara de subred'** y **'Dirección de Pasarela'**.



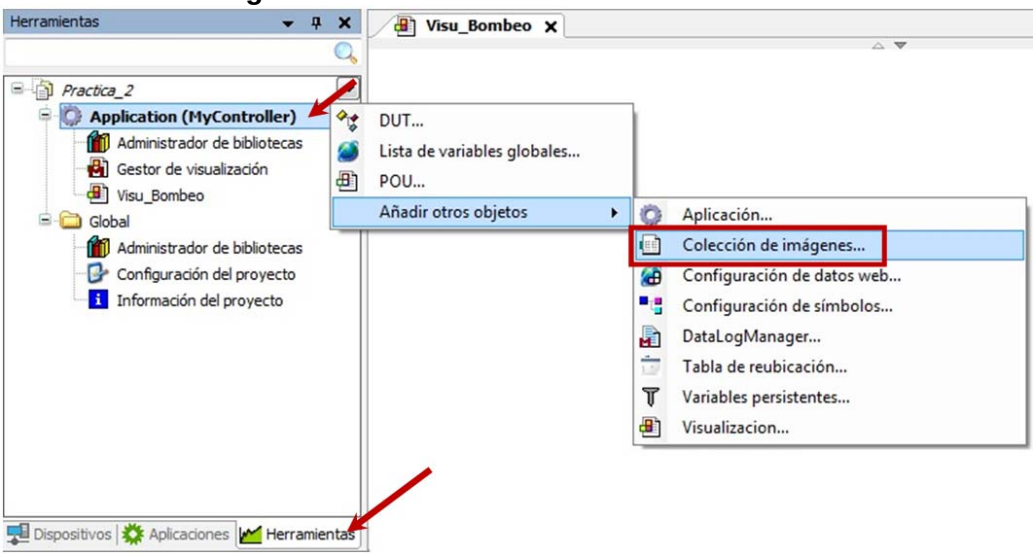
3. Ahora crearemos una visualización para ello nos iremos a la pestaña de **‘Herramientas’** del navegador y añadiremos el objeto visualización.



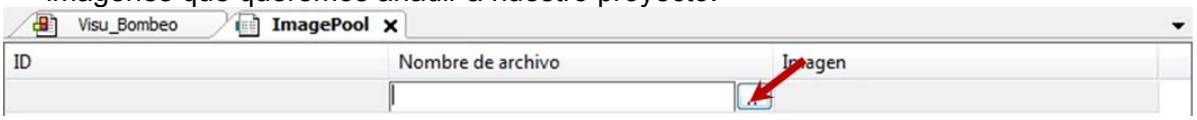
4. En la ventana que aparece tenemos que escribir un nombre a la visualización ‘Visu_bombeo’.



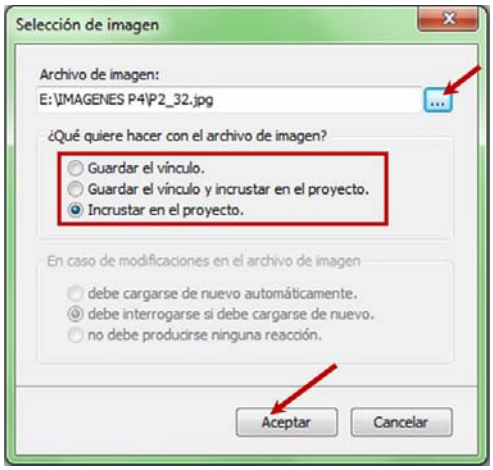
5. Añadimos también una **‘colección de imágenes’**, para ello seleccionamos **‘Application’** en la pestaña **‘Herramientas’** del navegador, pulsamos el icono **‘+’** que sale a la derecha de este y seleccionamos el objeto a **agregar ‘Colección de Imágenes’**.



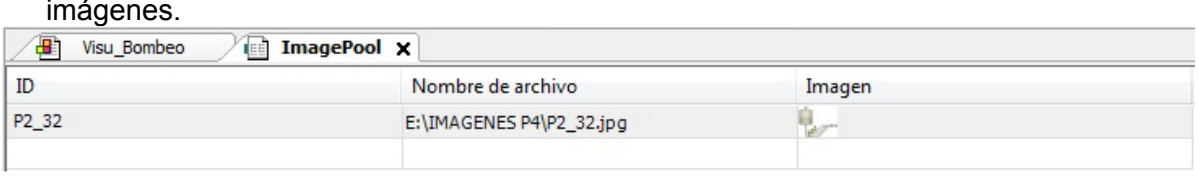
6. En el área de trabajo seleccionamos la primera fila de la tabla y en la columna de **'Nombre de Archivo'**, pulsamos el botón resaltado **'…'** para seleccionar las imágenes que queremos añadir a nuestro proyecto.



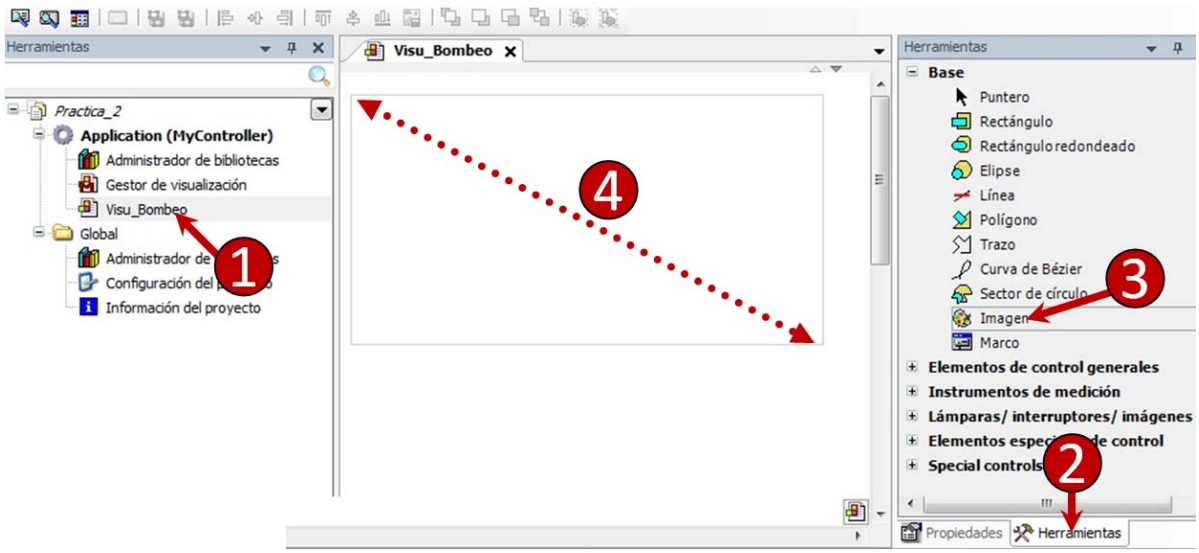
7. Una vez seleccionada la imagen, nos aparece una ventana con diferentes opciones para hacer con el archivo de la imagen, en nuestro caso la **'Incrustar en el proyecto'**, teniendo en cuenta que si incrustamos muchas imágenes en el proyecto, este puede incrementar mucho su tamaño.



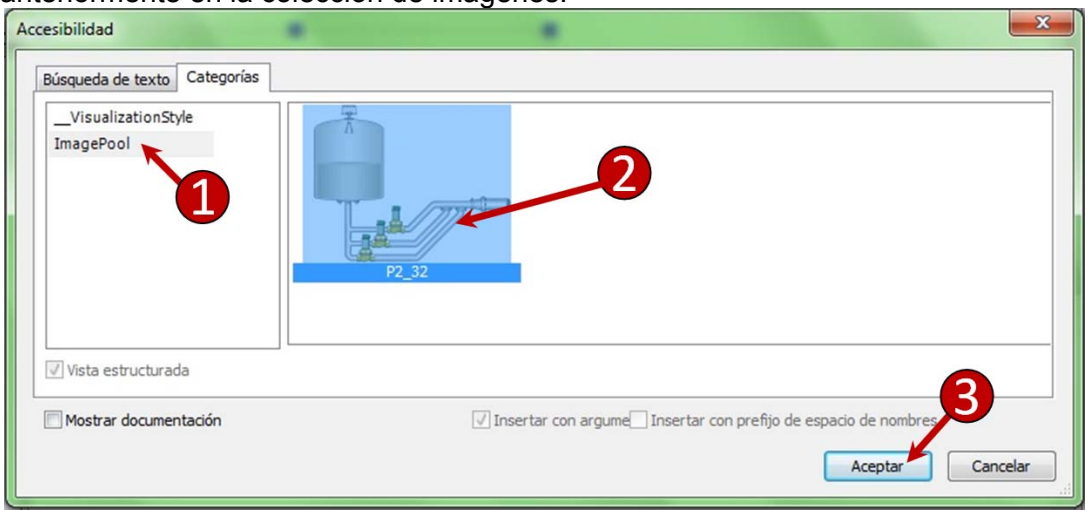
8. Ahora ya nos aparece en la tabla la imagen añadida a nuestra colección de imágenes.



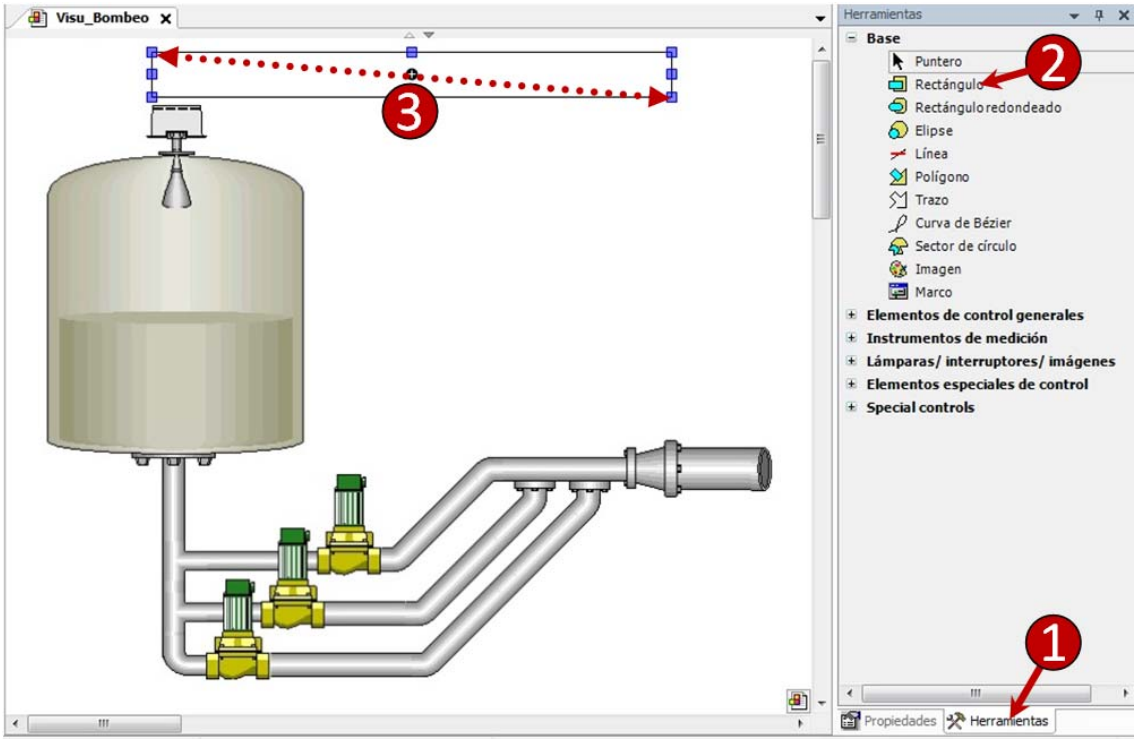
9. En la visualización, en la pestaña de **'Herramientas'** de la parte de la derecha, desplegamos **'Base'**, y añadiremos **'Imagen'**, seleccionándola y luego haciendo un cuadro manteniéndolo pulsando con el tamaño que deseamos que tenga la imagen.



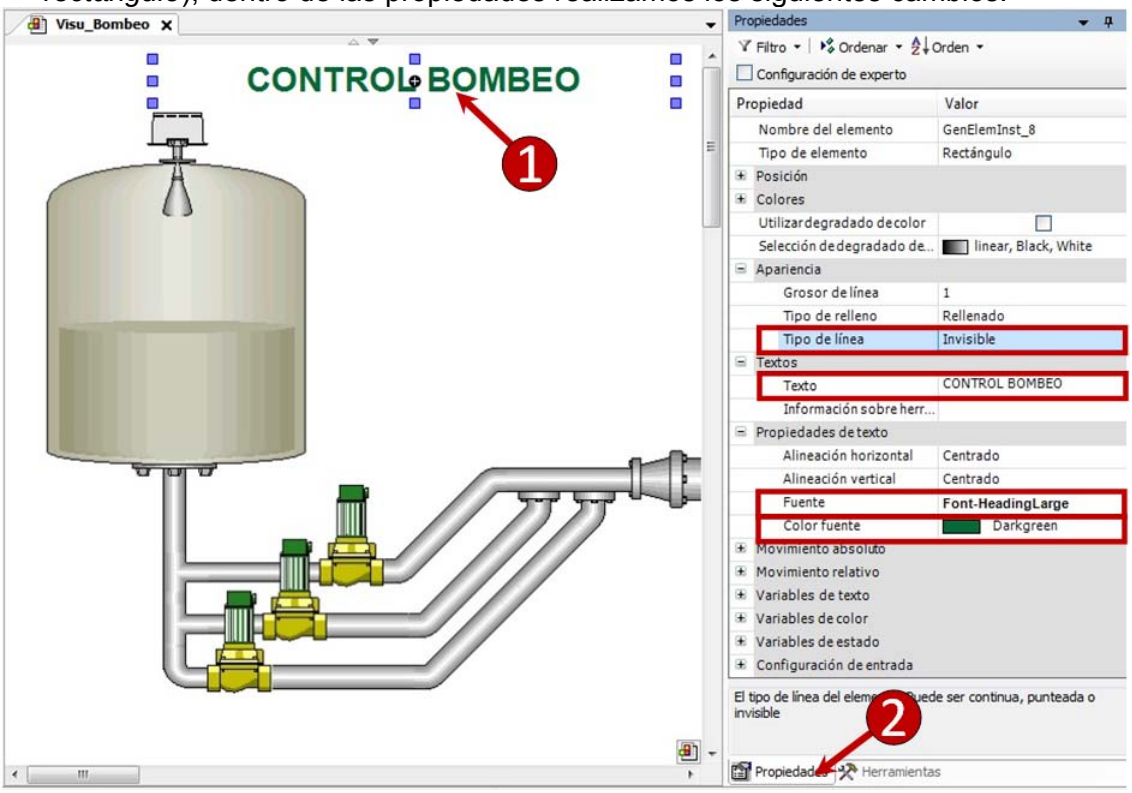
10. Aparece una ventana para elegir la imagen que deseamos, seleccionamos en la parte derecha **'Imagepool'**, y elegimos la imagen que habíamos añadido anteriormente en la colección de imágenes.



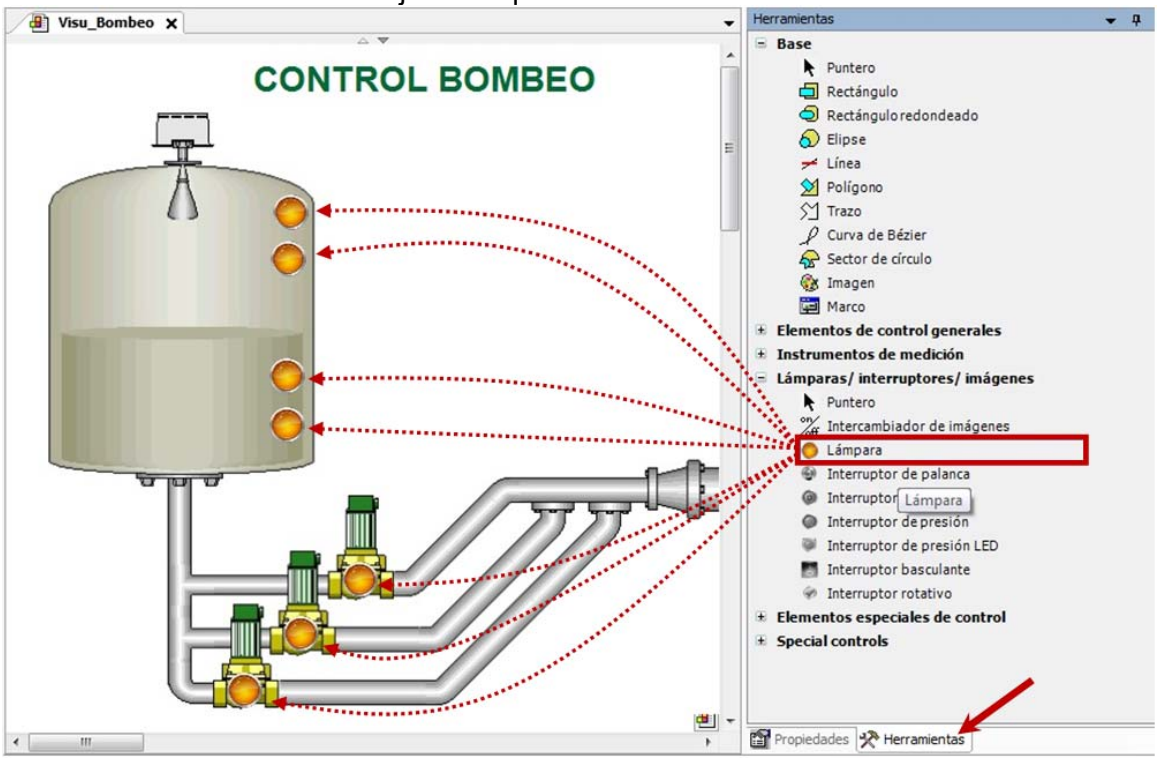
11. Una vez añadida la imagen (la ajustamos para que se vea bien), en la misma pestaña **'Herramientas'** volvemos a desplegar **'Base'**, añadiremos **'Rectángulo'**, seleccionándolo y luego haciendo un cuadro manteniéndolo pulsando con el tamaño que deseamos que tenga.



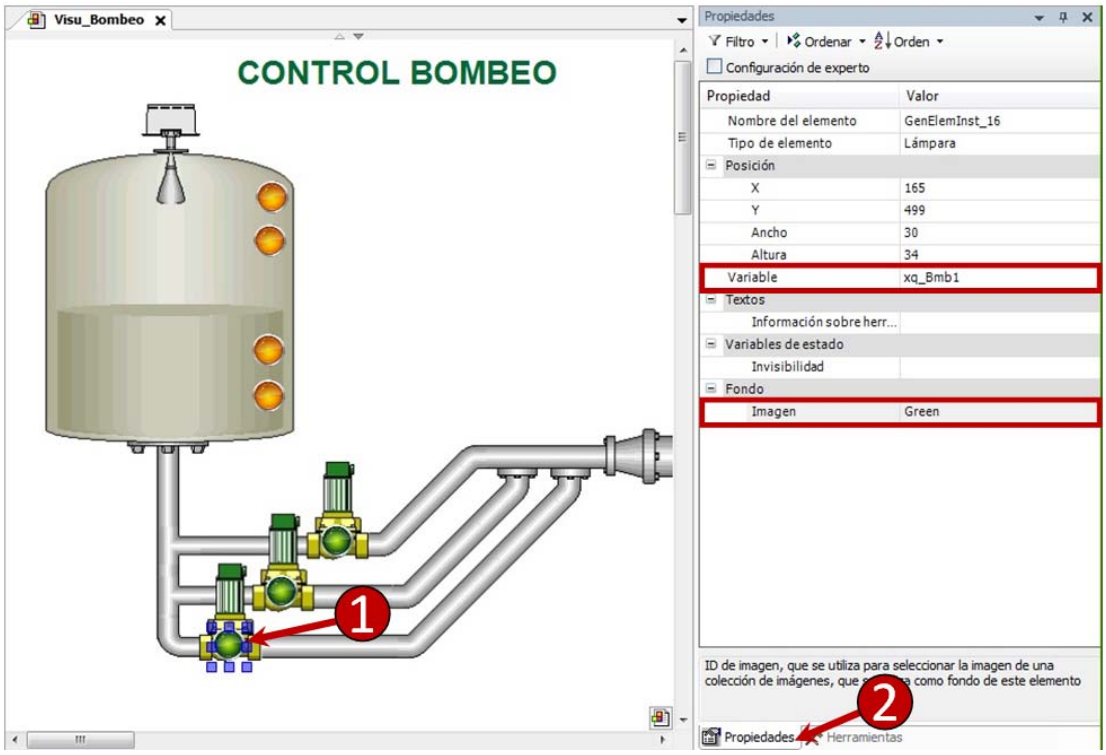
12. Seleccionamos el rectángulo y seleccionamos la pestaña **'Propiedades',** que nos muestra las propiedades del objeto seleccionado (*en este caso el rectángulo*), dentro de las propiedades realizamos los siguientes cambios:



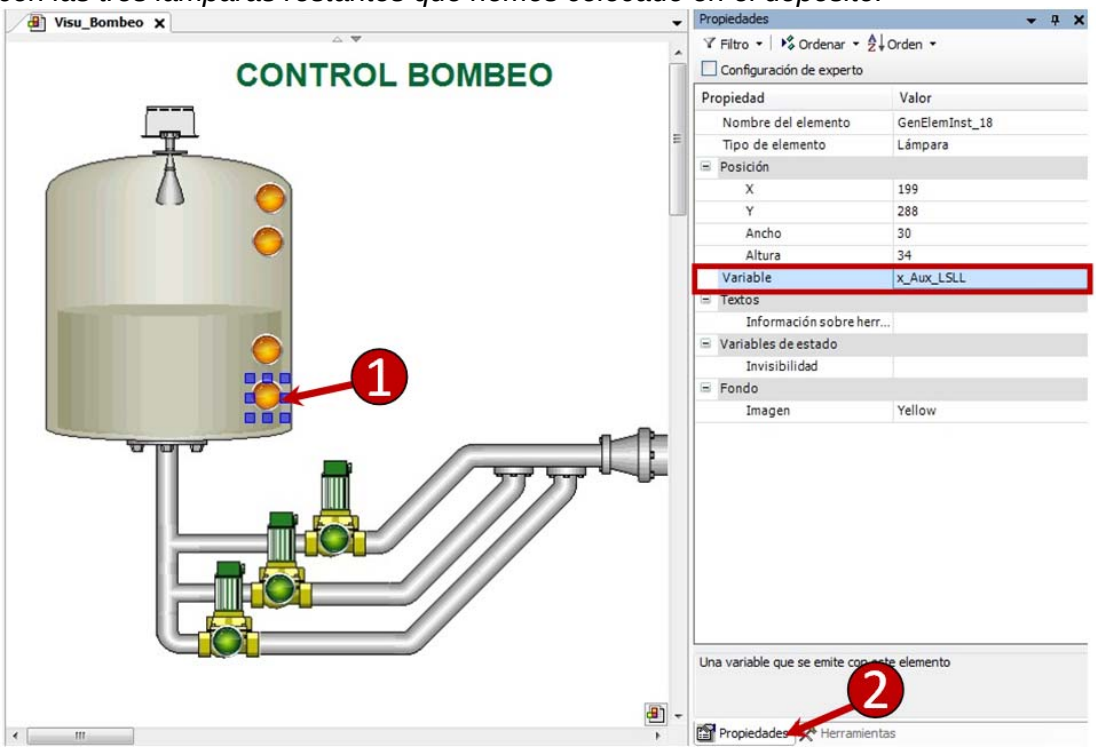
13. A continuación, añadiremos 7 **'Lámpara'** que se encuentra en la pestaña de **'herramientas',** en la que desplegamos **'Lámparas / interruptores / imágenes'**. Las colocamos pulsando sobre ella y manteniéndolo lo colocamos en el sitio del área de trabajo en la que deseamos.



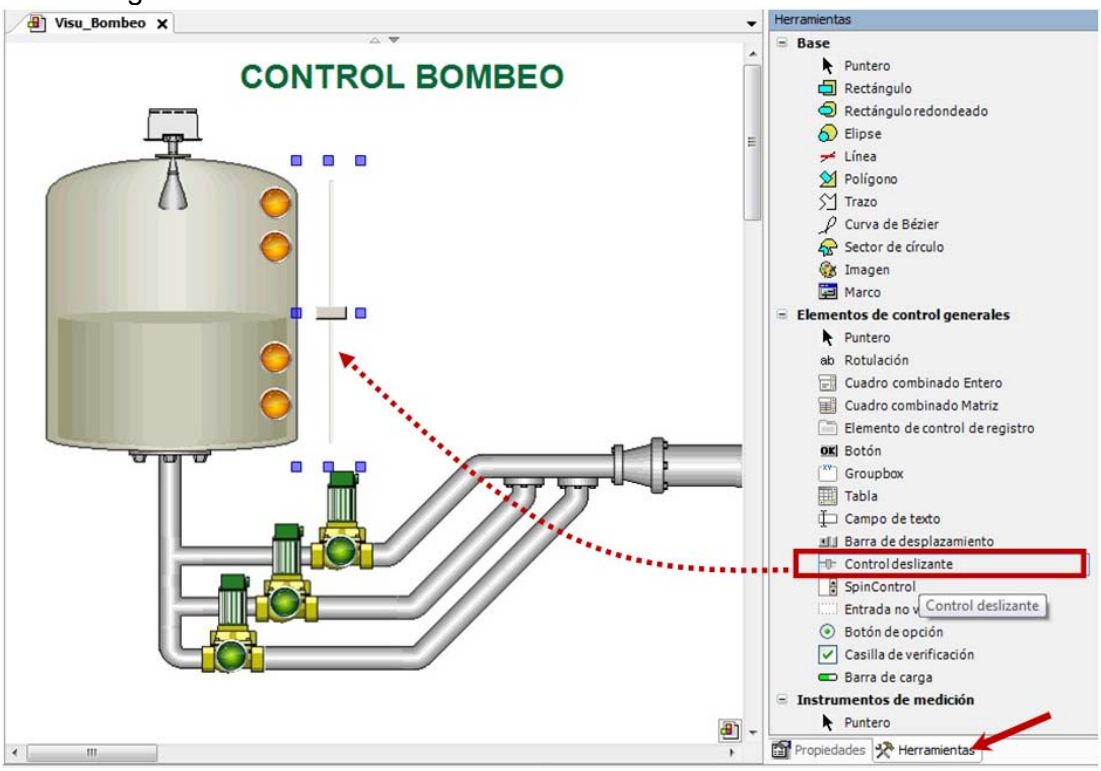
14. Una vez colocadas las 7 lámparas en su sitio, seleccionamos la primera que se encuentra encima de la bomba y seleccionamos la pestaña **'Propiedades'**, que nos muestra las propiedades de esta, dentro de las propiedades en el campo **'Variable'** la linkamos a la variable del proceso que queremos ver (*en este caso xq_Bmb1*). *Repetimos esto mismo con las tres lámparas que hemos colocado encima de las bombas.*



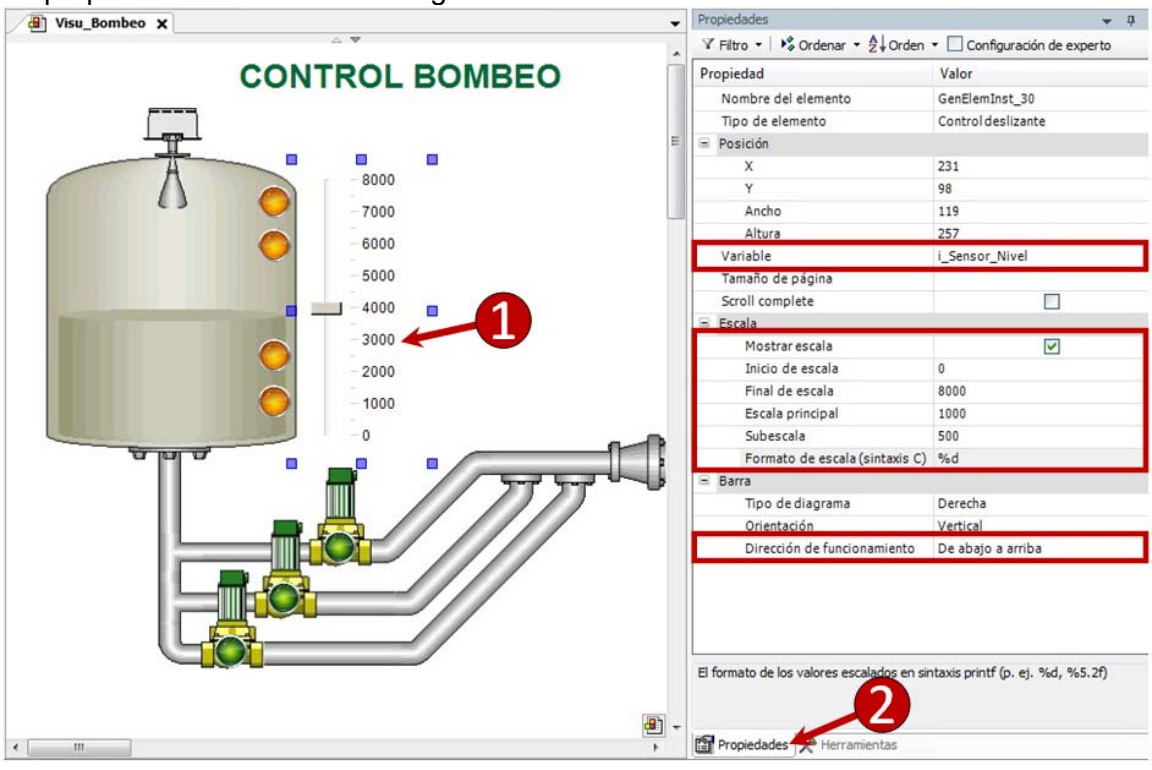
15. Ahora seleccionamos la primera que se encuentra en el depósito y en la pestaña **'Propiedades'**, en el campo **'Variable'** la linkamos a la variable del proceso que queremos ver (*en este caso x_Aux_LSLL*). *Repetimos esto mismo con las tres lámparas restantes que hemos colocado en el depósito.*



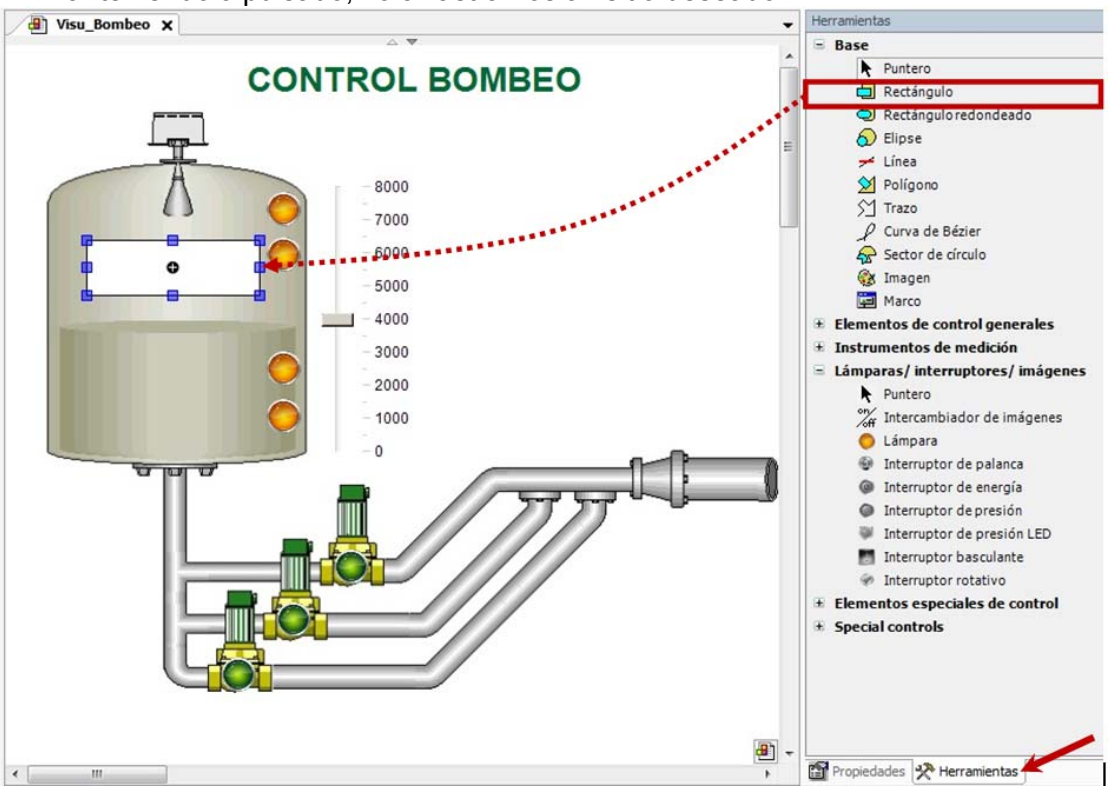
16. Seguimos añadiendo objetos, ahora un control deslizante para simular el valor del nivel del agua. En la pestaña **'herramientas',** desplegamos **'Elementos de control generales'**, seleccionamos **'control deslizante'** lo colocamos pulsando sobre él, y manteniéndolo pulsado, lo arrastramos al sitio deseado, ahora lo alargamos hasta que nos quede de manera vertical, como se muestra en la figura.



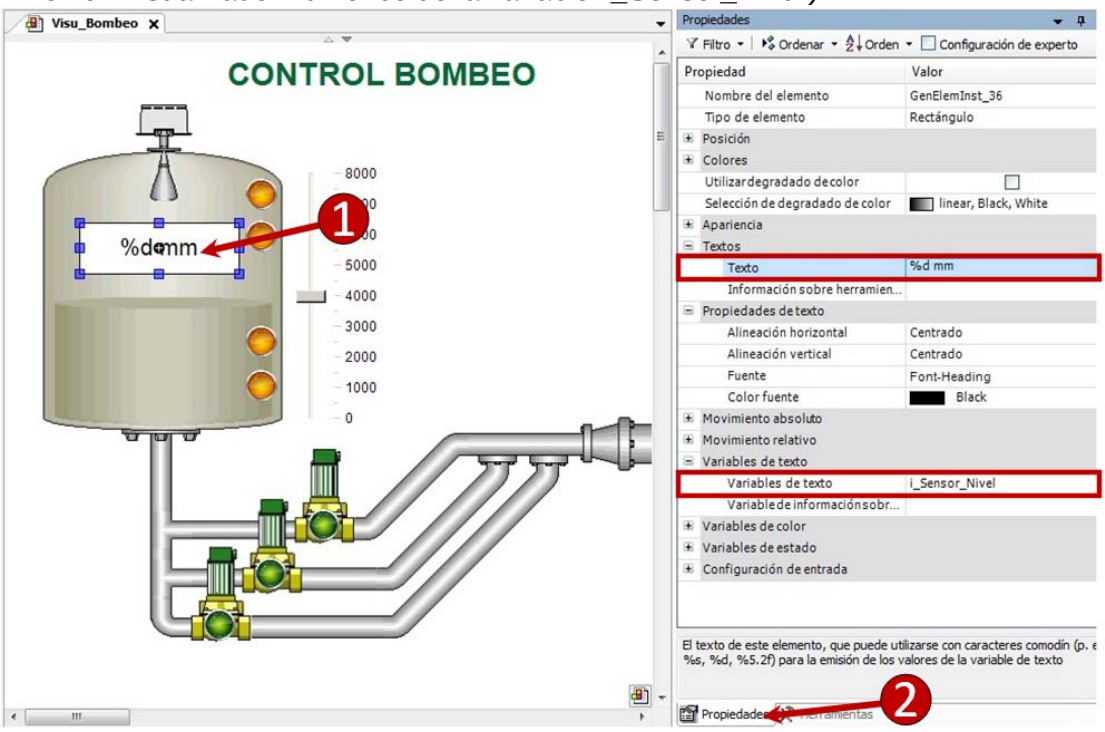
17. Seleccionamos el control deslizante y vamos la pestaña **'Propiedades',** que nos muestra las propiedades del objeto seleccionado, dentro de las propiedades realizamos los siguientes cambios:



18. Ahora añadimos otro rectángulo. En la pestaña **'herramientas',** desplegamos **'Base'**, seleccionamos **'Rectángulo'** lo colocamos pulsando sobre él, y manteniéndolo pulsado, lo arrastramos al sitio deseado.



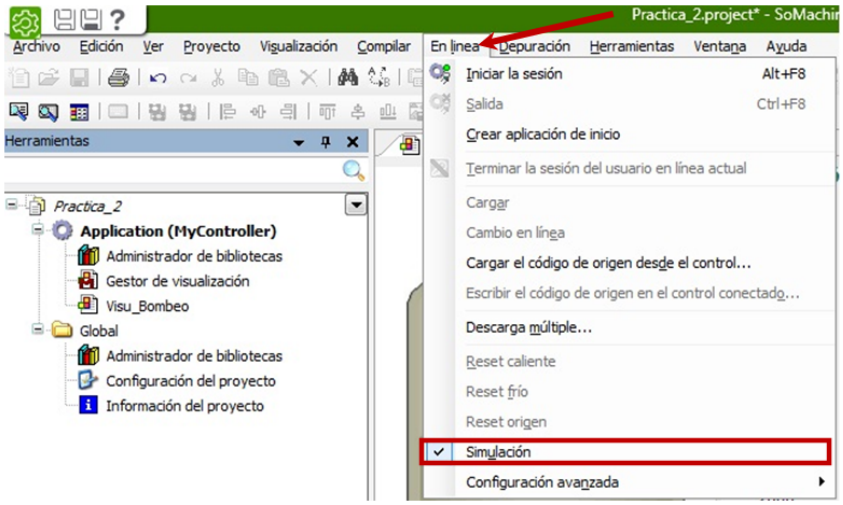
19. Seleccionamos el rectángulo y seleccionamos la pestaña **'Propiedades',** que nos muestra las propiedades del objeto seleccionado, dentro de las propiedades realizamos los siguientes cambios *(para transformar el rectángulo en un visualizador numérico de la variable 'i_Sensor_Nivel').*



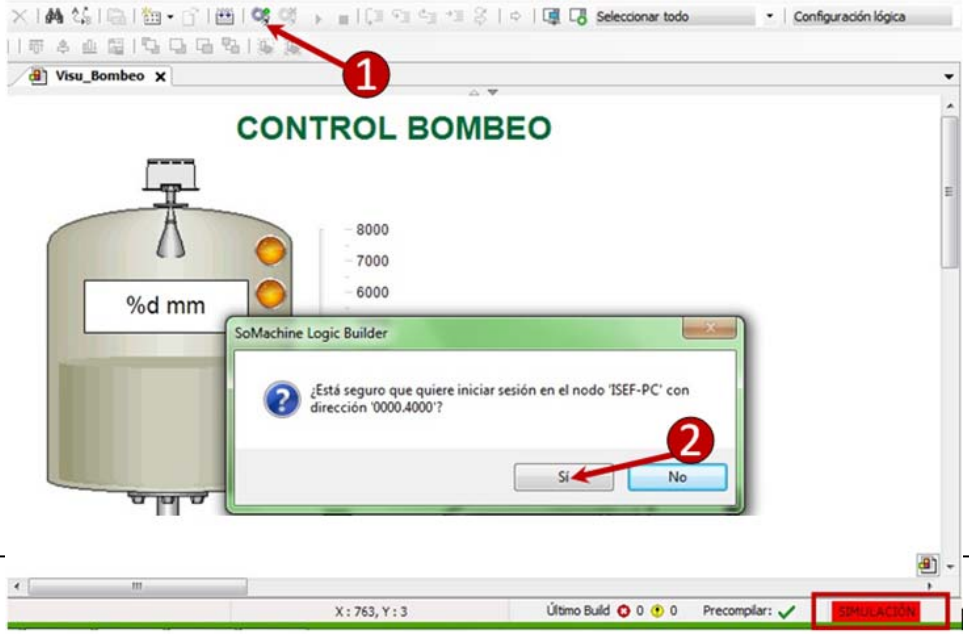
En función del tipo de variable que deseamos visualizar en el rectángulo, tenemos que utilizar en el campo 'Texto' la siguiente nomenclatura dependiendo del tipo.

Caracter	Argumento o formato
d,i	Número en decimal
b	Número en binario
o	Número en octal
x	Número en Hexadecimal
u	Número en decimal sin signo
c	Un caracter
s	Se verá el valor en formato string (agrupación de carácter) que se haya especificado en la variable
f	Valor Real; con la siguiente sintaxis '**%<alineación > <ancho mínimo>.<decimales> f** '. La alineación se define con un signo '**-**' en el caso que se quiera alineado a la izquierda, con un '**+**' si se quiere a la derecha .

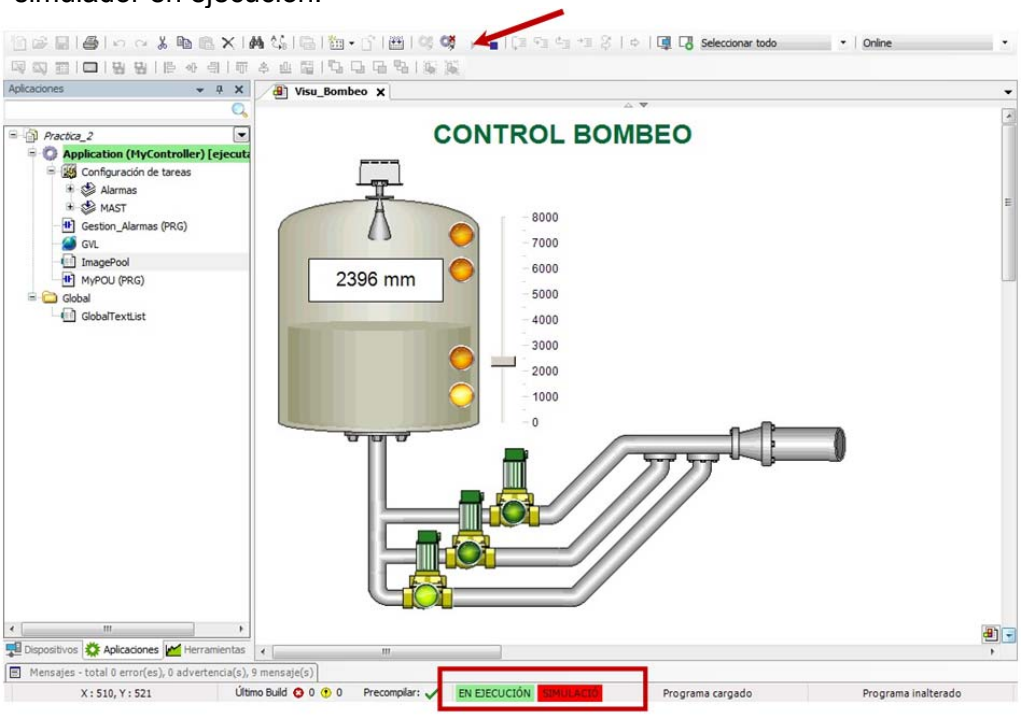
20. Una vez finalizada la visualización, para comprobar el correcto funcionamiento de la misma, nos vamos al menú textual general **'En línea'** y activamos la **'Simulación'**, para simular el programa y comprobar que la visualización es correcta.



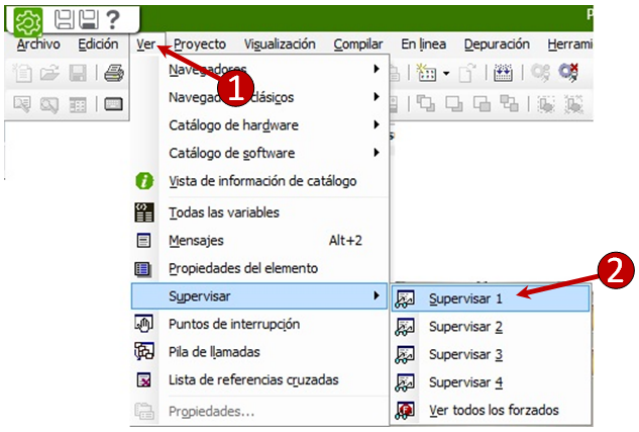
21. Ahora que está en modo simulación, pulsamos el icono **'Iniciar la sesión'** de la barra de herramientas y aceptamos la carga al simulador.



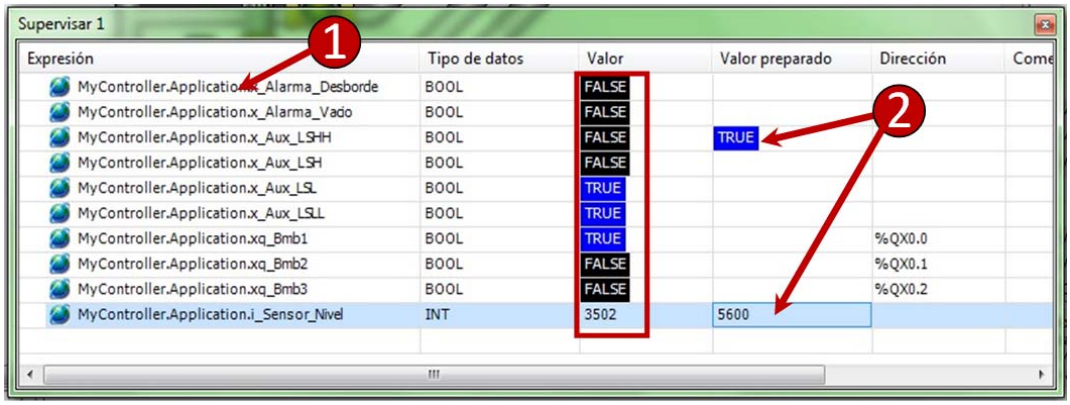
22. Una vez iniciado el programa le damas al botón de **'RUN'** para poner el simulador en ejecución.



23. Ahora para realizar una mejor depuración del proyecto, en el menú textual **'Ver'** seleccionamos la opción **'Supervisar/Supervisar 1'** que nos añade una ventana de supervisión donde podremos añadir las variables que queremos monitorizar.

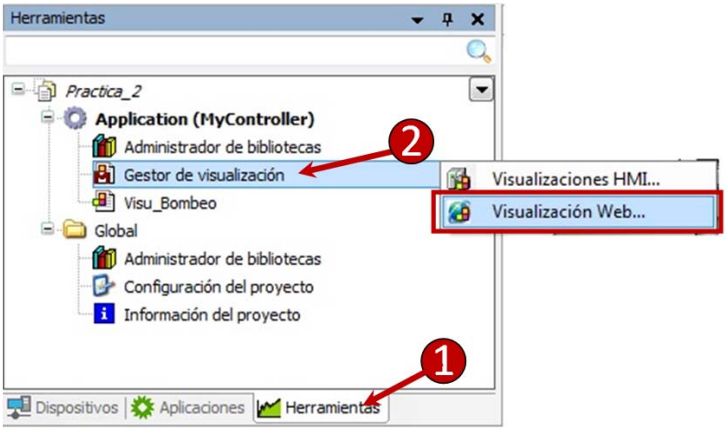


24. Añadimos las siguientes variables a la lista de supervisión, en la columna de '**valor preparado'** podemos cambiar el valor de la variable y luego pulsando **'Ctrl+F7'** para modificar la variable ó **'F7'** para forzarla.



5.2.3 Creación de la webvisualización

1. Una vez hemos asignado las variables a los elementos ya estará terminada la visualización, tendremos que decirle al programa que la vamos a publicar vía web añadiendo el **'web visualization'**, para ello seleccionaremos **'Visualization Manager'** botón derechos y añadimos el objeto **'Web visualization…'**.



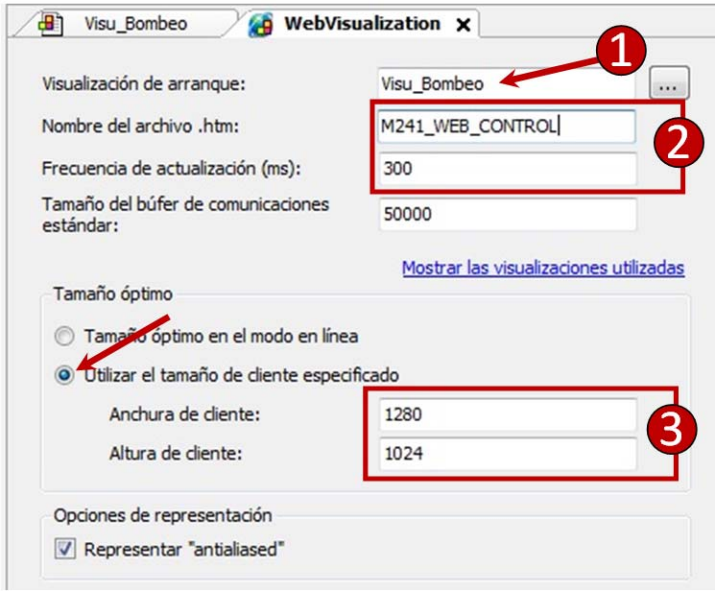
2. Configuraremos el **'Webvisualization'**.

   - En el indicaremos la '**visualización de inicio'** (en este caso como solo tenemos una, ya aparece el nombre de la visualización ' **Visu_Bombeo'**).

   - Le tendremos que especificar un nombre al archivo *.htm (en este caso será '**M241_WEB_CONTROL'**).

   - Dejaremos el refresco de la página en **20 ms** y el buffer de comunicación en **50000**.

   - Por último dejaremos que la pantalla se adapte al tamaño del dispositivo que se conecte en modo online. (tablet, moviles …etc) **'Best fit in online mode'**.



3. Finalizada la configuración de la **'Webvisualization'**, ya podríamos compilar y cargar el programa en el equipo.

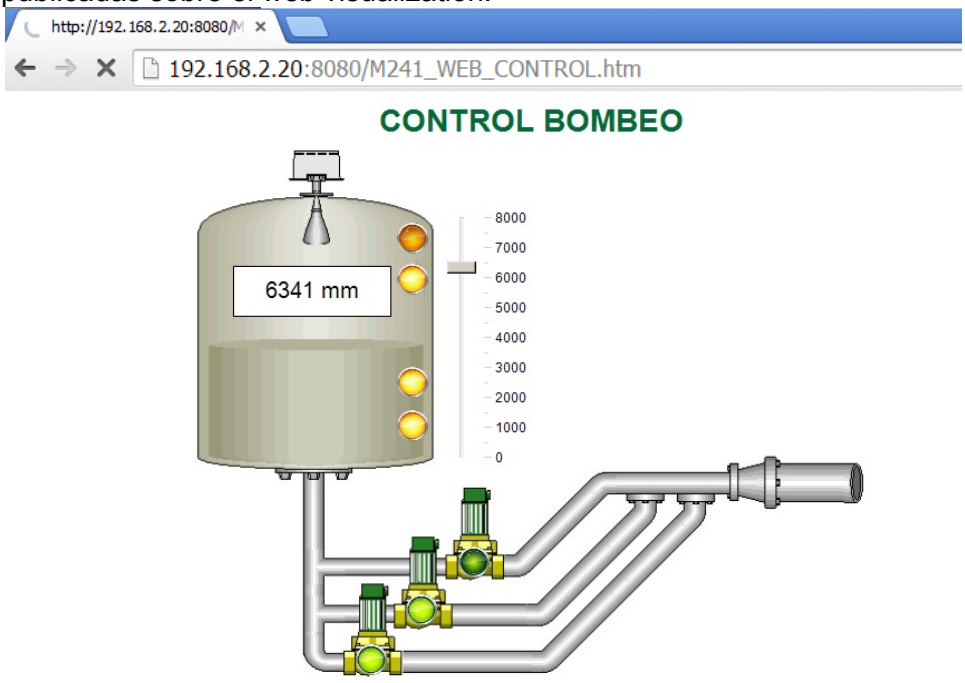
4. En el caso del ejemplo, ahora tenemos que configurar el router_wifi TP Link que puede diferir de otros equipos. Para configurar conectamos punto a punto con un cable de Ethernet, el equipo TP Link al PC. Abrimos el navegador (internet explorer, chrome ..etc) y escribimos **192.168.0.254.** (Si el router ya tiene otra red wifi que la de fábrica , pulsamos el botón de reset)

5. Te pedirá el Usuario y contraseña.

Login: **admin**
Password: **admin**

6. Una vez conectados a la red wifi, solo tendremos que abrir nuestro explorador de internet y escribimos la siguiente dirección en el explorador '< *dirección IP del controlador*> **:8080/**<*nombre del archivo htm*>**.htm**'. Para este ejemplo tendremos que escribir lo siguiente:
http://192.168.90.10:8080/M241_WEB_CONTROL.htm'.

7. Cuando se conecte veremos y podremos actuar sobre las visualizaciones publicadas sobre el web visualization.

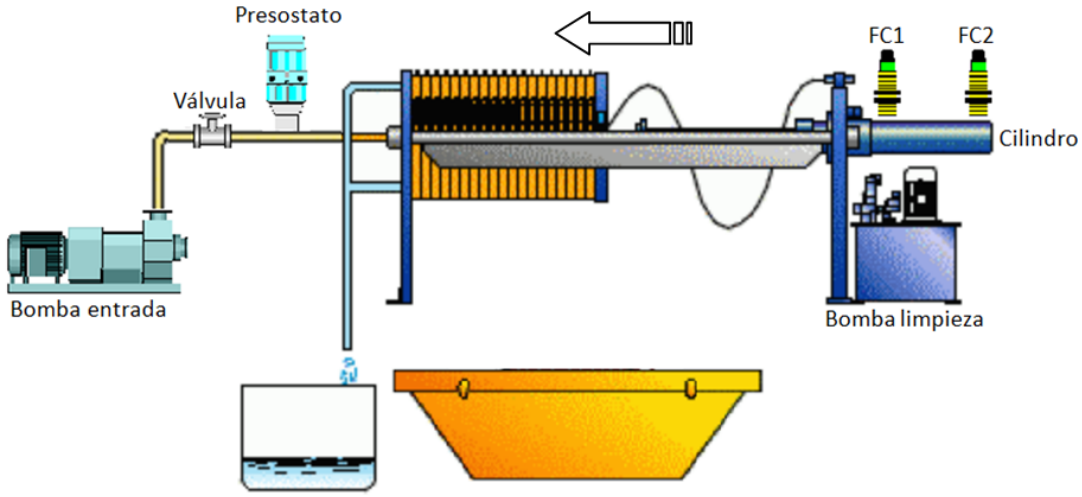


5.3.- PRACTICA 3: LIBRERÍA

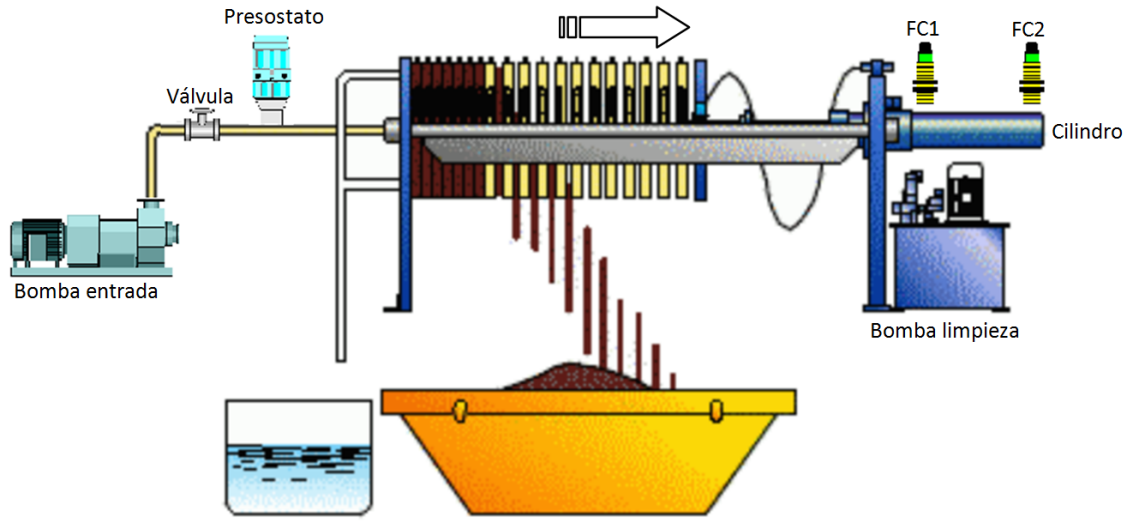
Crear una librería de usuario para el control del funcionamiento del filtro prensa.

El filtro prensa tiene una operación muy sencilla:

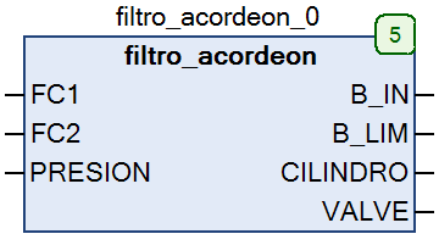
Primero, el lodo líquido es bombeado a las cámaras que se encuentran rodeadas por lonas filtrantes, la bomba de entrada comienza a bombear con el filtro totalmente extendido **FC2**. Al bombear la presión se incrementa y el lodo es forzado a atravesar las lonas, provocando que los sólidos se acumulen y formen una pasta seca, la bomba de entrada bombea agua hasta que el presostato nos da señal de presión alta **(3000 mbar)**. Posteriormente, la válvula de entrada se cierra y el pistón hidráulico empuja la placa de acero contra las placas de polietileno haciendo la prensa hasta llegar al **FC1**. El filtrado pasa a través de las lonas y es dirigido hacia los canales de las placas y puertos de drenado del cabezal para descarga.



Una vez han pasado **50 segundos**, para remover las pastas compactadas, se hace retroceder el pistón neumático, relajando la presión y separando cada una de las placas, para permitir que la pasta compactada caiga desde la cámara. Después de esto se pasa a la etapa de limpieza que inyecta agua caliente a presión durante **30 segundos** a las lonas desde la bomba de limpieza.

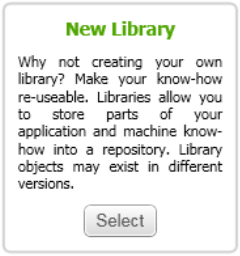


Para ello quiero crear una instrucción que gobierne este tipo de filtros que tendrá el siguiente aspecto:

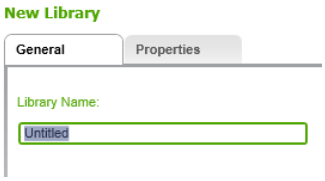


5.3.1.- Crear un proyecto vacío tipo librería.

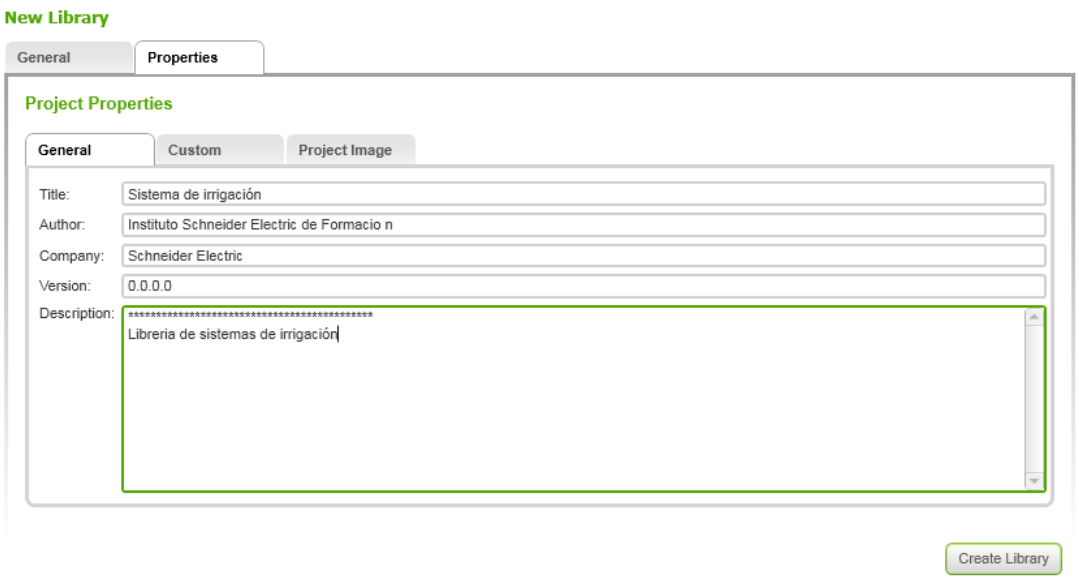
Para crear una librería de usuario, en la ventana de **'Inicio'** del SoMachine hay que crear un proyecto **nuevo y pulsar en 'New Library'** y a la hora de guardar el archivo, se creará con el tipo librería.



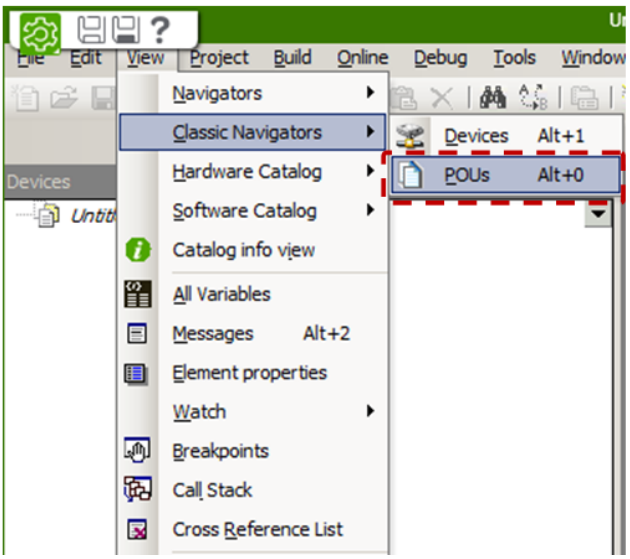
Una vez creado el proyecto, en las pestaña **'General'**, escribiremos el nombre queremos que tenga nuestra librería.



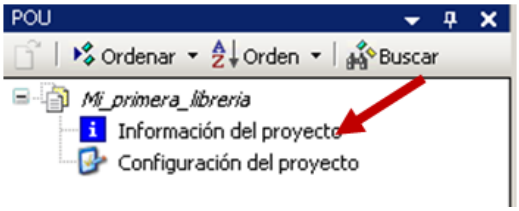
En la pestaña **'Properties'**, se introducirá la información que va a tener la librería: Compañía, Titulo (nombre de la librería), Versión – El resto de la información es opcional.



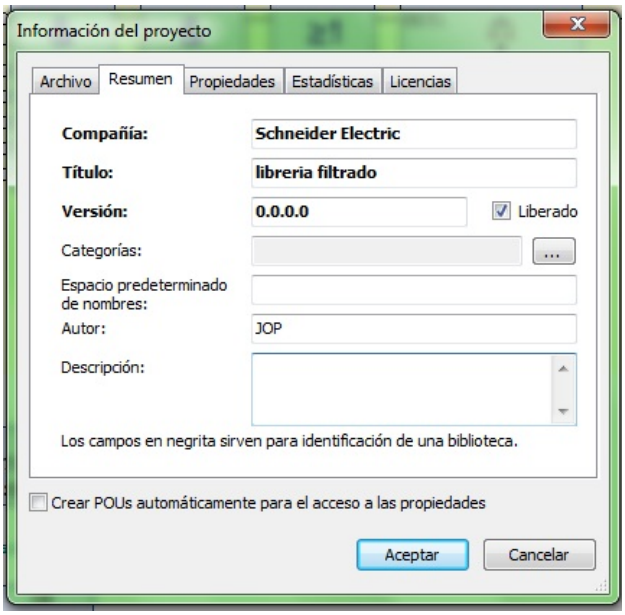
En el menú contextual seleccionar **'Vista » Classic Navigator » POU'** aparecerá en el navegador la vista **'POU's'** que es donde se añaden los programas, funciones, bloques de funciones y estructura de datos que se quieren añadir a la librería.



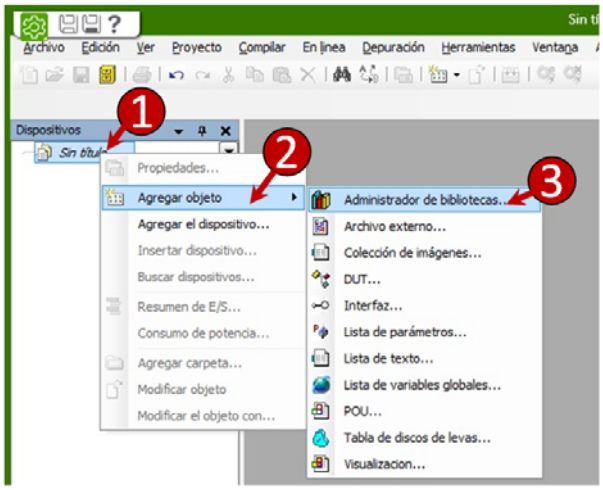
Para añadir los datos que luego se verán en la librería, dentro de la vista **'POU'** hacemos doble clic en **'Información del proyecto'**



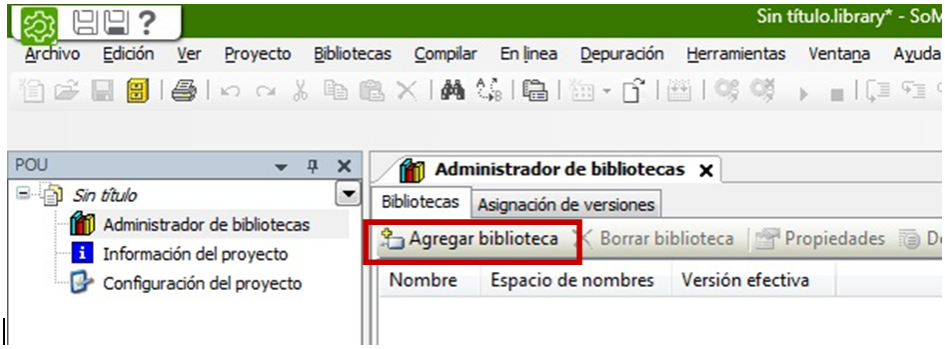
En el campo **'Compañía'** podemos poner el nombre de nuestra compañía, lo más importante es poner el **'título',** que será el nombre de nuestra librería **'librería filtrado'.** También podemos crear nuestra versión de librería de manera manual o automática pulsando la opción de **'Liberado'**. Es posible asociar nuestra librería a una categoría, sino ponemos nada se añadirá a la categoría **'Varios'**.



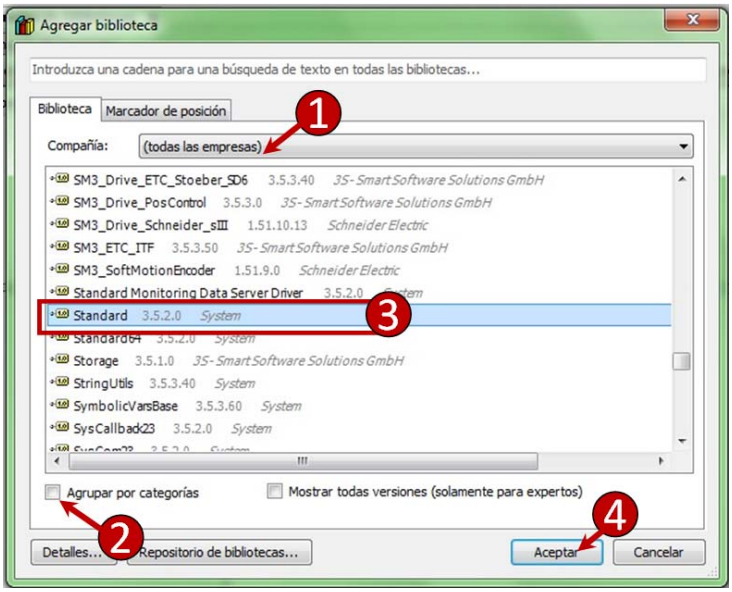
Para añadir los objetos en este caso seleccionar el directorio del proyecto (ejemplo. Mi_primera_libreria) pulsar botón derecho y seleccionar **‘Añadir Objeto’** y añadiremos el ‘**Administrador de bibliotecas’,** donde añadiremos las librerías necesarias para programar nuestras instrucciones.



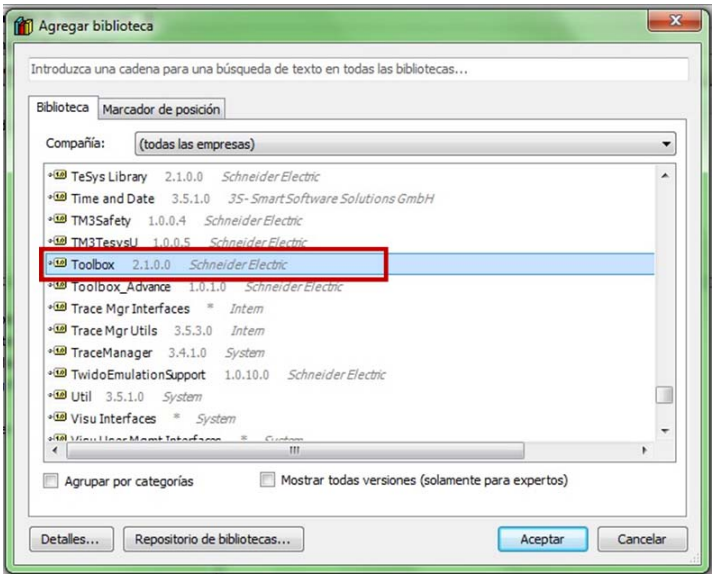
Para agregarlas a la librería, dentro de la pestaña **‘Herramientas’** del navegador abriremos el **‘Administrador de Bibliotecas’** y seleccionaremos la acción, **‘Agregar biblioteca’** para añadirla a nuestro proyecto.



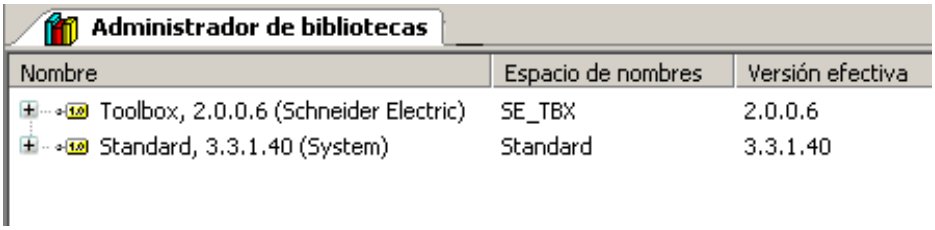
Aparece el ‘**Repositorio de Bibliotecas’,** donde deseleccionamos la opción **‘Agrupar por categorías’** para así poderla buscar en orden alfabético. Buscamos y seleccionamos la librería **‘Standard’** y le damos a **‘Aceptar’.**



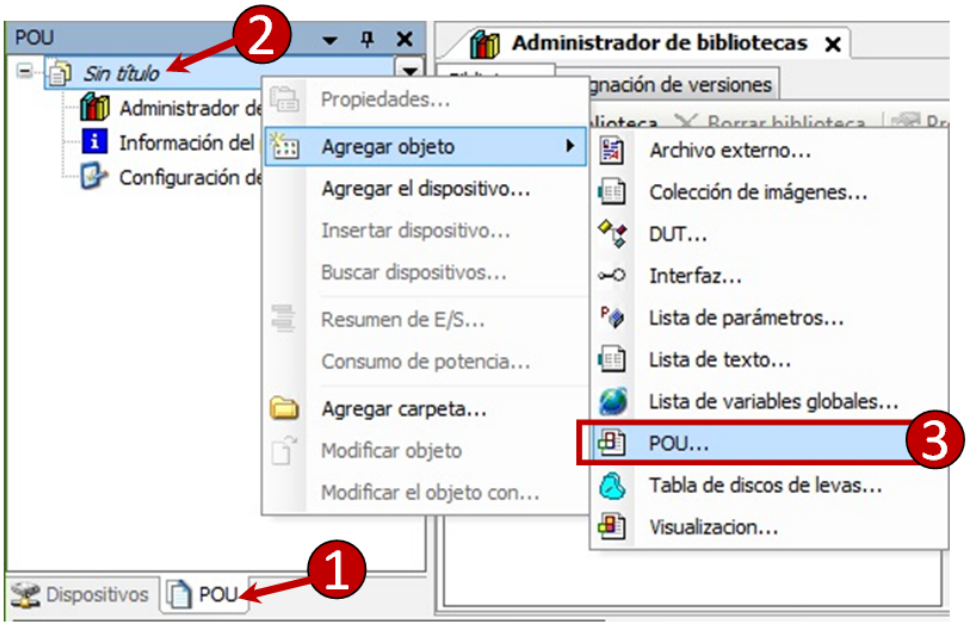
Añadimos también la librería **‘Toolbox’**, realizando los mismos pasos.



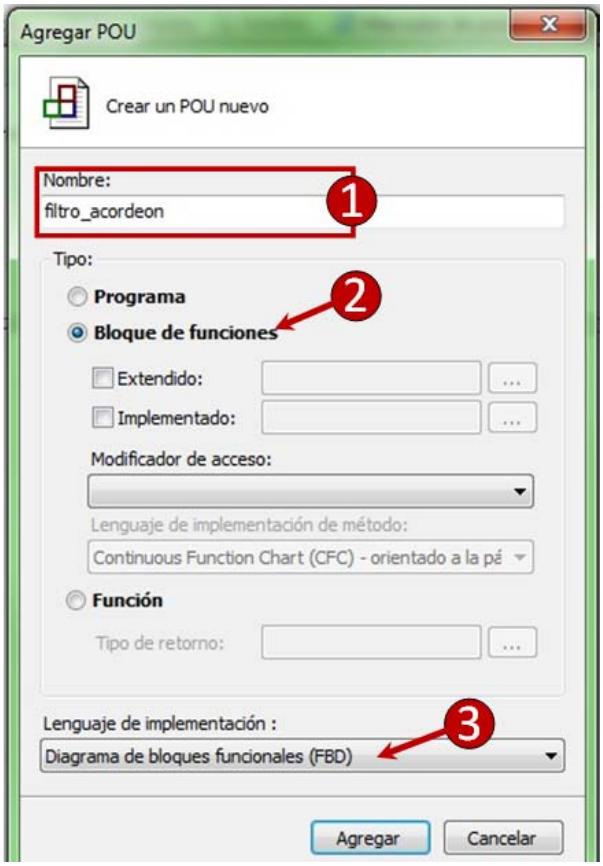
Quedando el **‘Administrador de Bibliotecas’** con dos librerías añadidas, con las instrucciones básicas de programación.



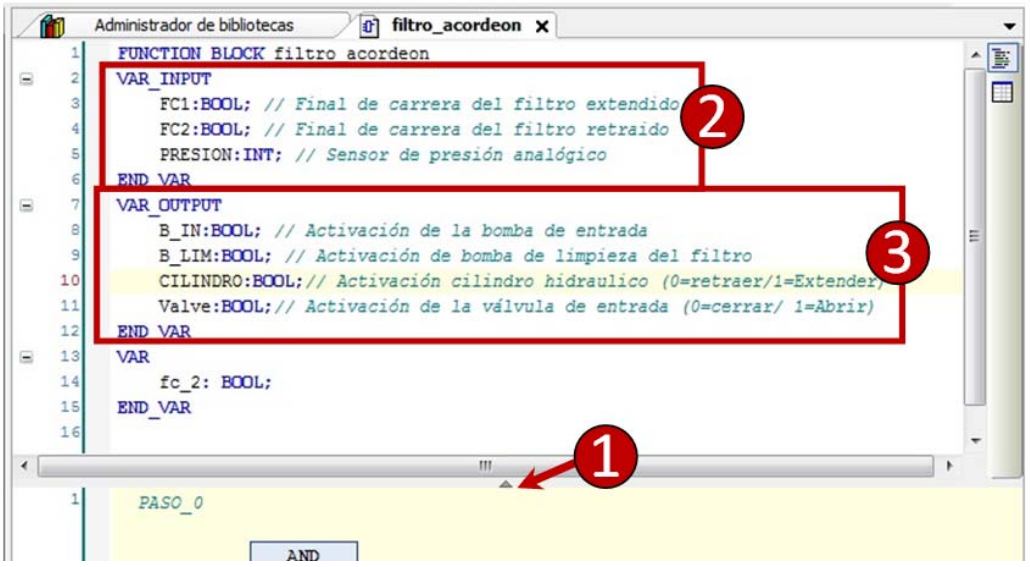
Ahora en la pestaña ‘POU’ del navegador seleccionamos la carpeta del proyecto y con el botón derecho, seleccionamos ‘Agregar Objeto’ y dentro de este escogemos ‘POU’.



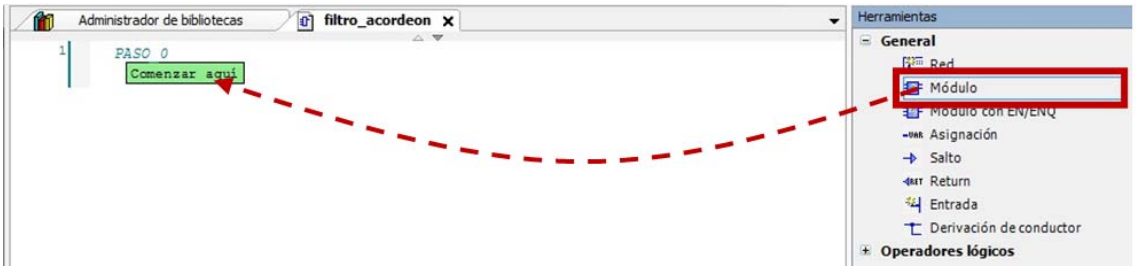
Ahora en la ventana de declaración **'Agregar POU'** pondremos en el campo **'Nombre'** el nombre de nuestro POU (*filtro_acordeon*) que será el nombre que tendrá nuestra instrucción. En **'Tipo'** seleccionamos **'Bloque de funciones',** ya que se va a tratar de una instrucción con diferentes pines de salida. Por último, seleccionamos el lenguaje de programación, que este ejemplo hemos decidido que sea **'Diagrama de bloques funcionales (FBD)'**.



Dentro del POU creado, abrimos la zona de declaración de variables y vemos que las variables están dividías en diferentes secciones, el **VAR_INPUT** definiremos los pines de entrada de nuestra instrucción (Nombre y tipo de variable), y en la zona **VAR_OUTPUT**, definiremos los pines de salida de nuestra instrucción (Nombre y tipo de variable. En la zona VAR Se utilizar para declarar variables internas de la instrucción.



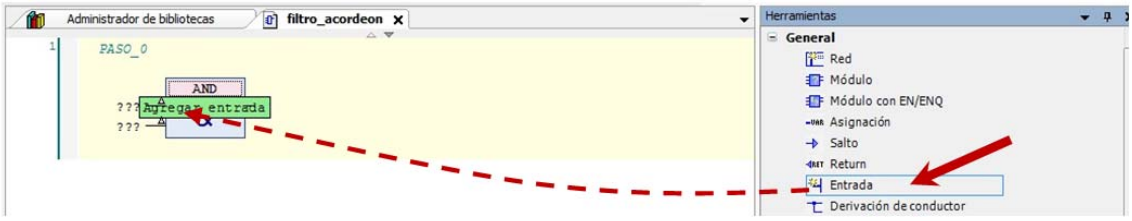
Empezaremos añadiendo un '**Modulo'**, seleccionándolo de la barra de herramientas y arrastrándolo al segmente 1.



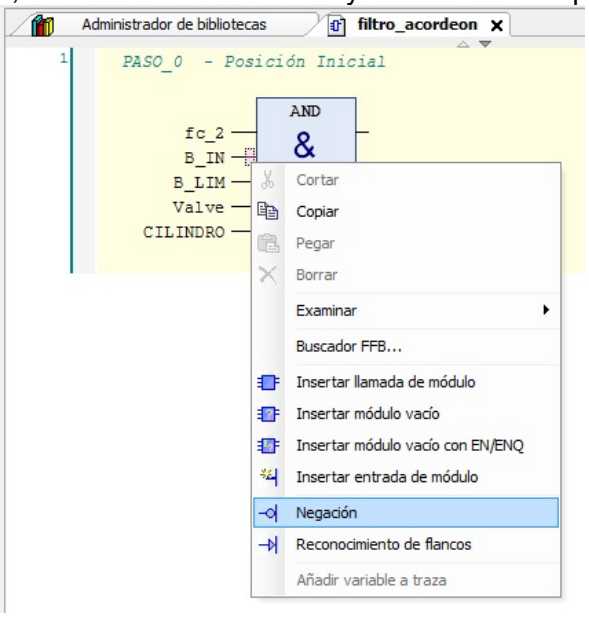
Haciendo clic dentro del módulo podemos escribir el nombre de la instrucción que deseamos añadir, sino pulsaremos el botón **' ...**' para ir a la ventana de 'Accesibilidad'

Para la programación de este programa en **lenguaje FBD**, tenemos que tener en cuenta las siguientes opciones de programación que son propias del lenguaje en sí.

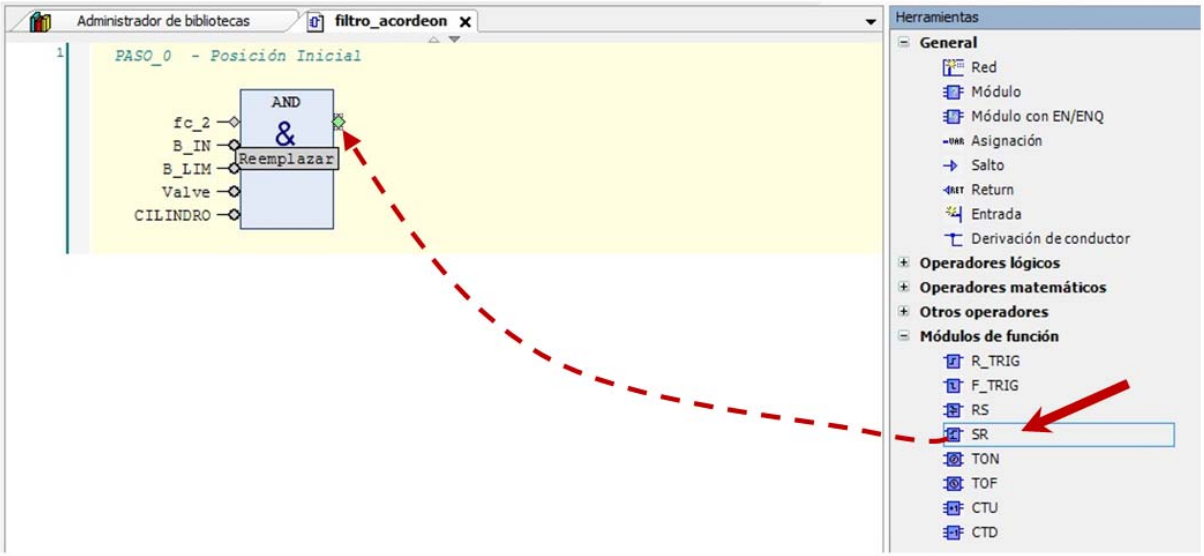
- Para añadir un pin de entrada a una instrucción como (**AND, OR…etc**), seleccionamos **'entrada'** en la barra de herramienta, pulsamos y la mantenemos hasta dejarla encima del módulo.



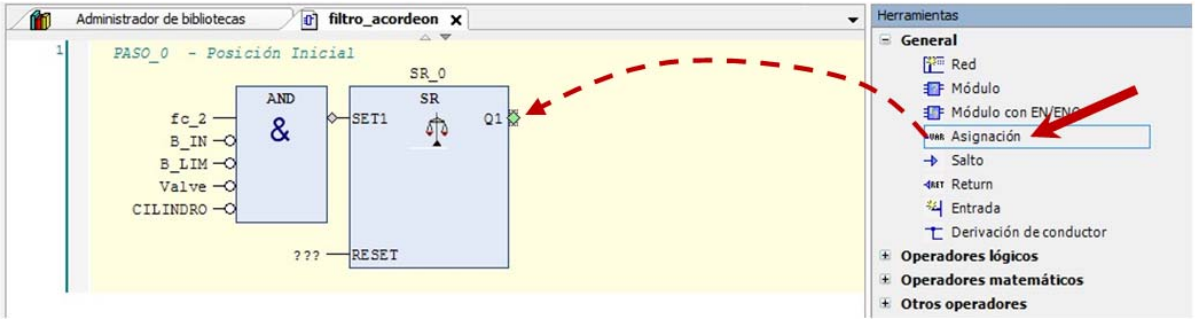
- Para negar un pin de una de las instrucciones tenemos que seleccionar el pin que deseamos, botón derecho del ratón y seleccionar la opción de **'Negación'**.



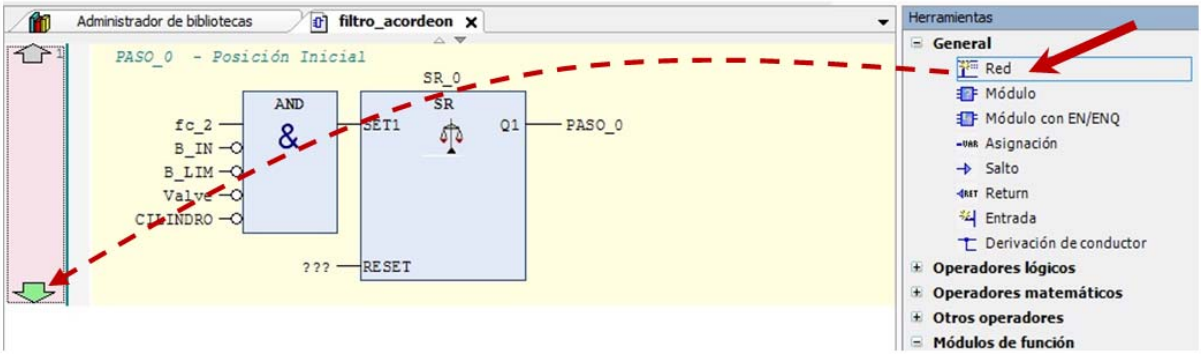
- Para añadir una instrucción a continuación de un módulo seleccionamos la instrucción pulsamos y la mantenemos hasta dejarla encima del pin de salida, si es correcto el linkado el circulo del pin de salida se pondrá en verde.



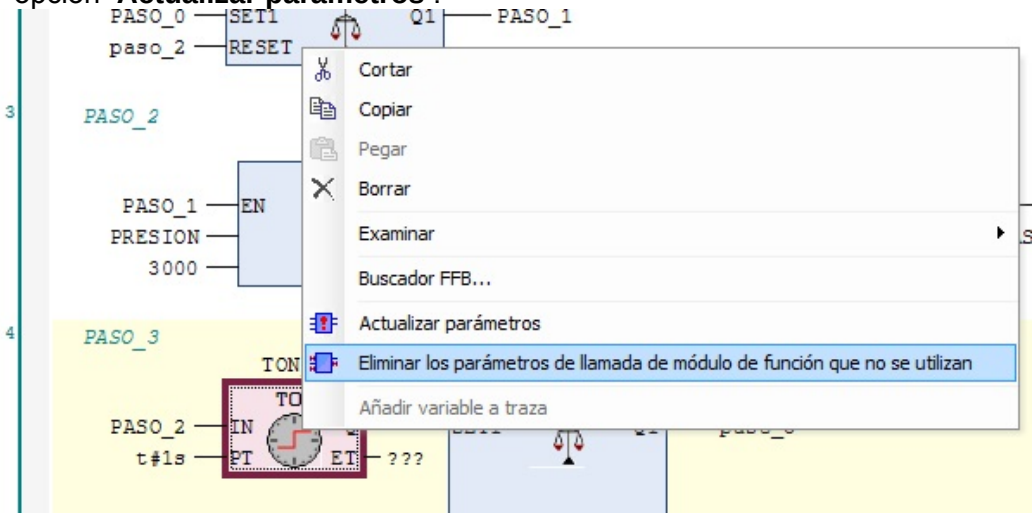
- Para acabar un segmento con una variable, debemos de seleccionar en la barra de herramientas **'Asignación'** y manteniéndolo pulsado, dejarla en el pin de salida de la instrucción, una vez hecho, aparece un recuadro donde podemos escribir o asignar la variable de salida deseada.



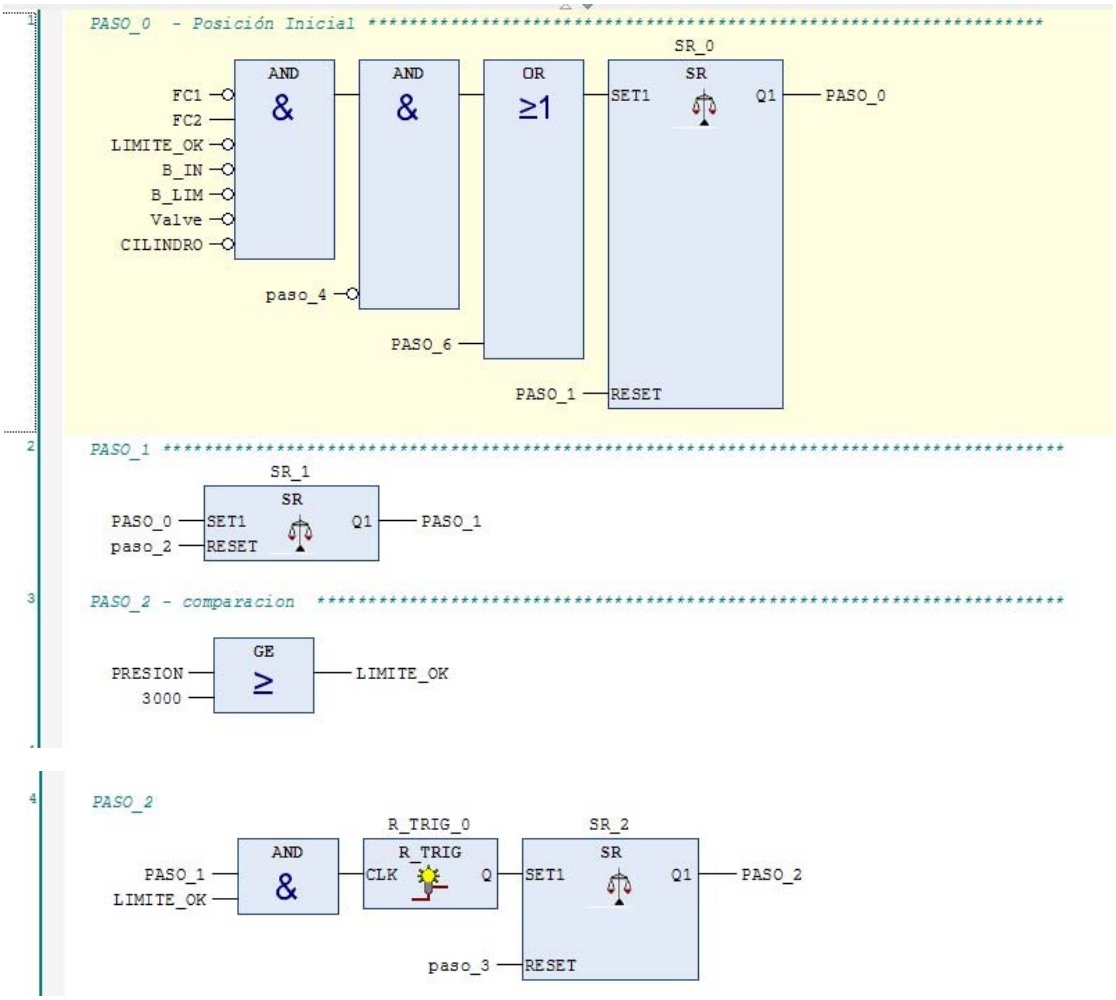
- Para añadir una nueva red, seleccionar **'Red'** de la barra de herramienta y pulsamos y arrastramos hasta la zona izquierda del área de programación, donde lo depositamos en la flecha de arriba o de abajo, según la posición donde queremos insertar la nueva red.



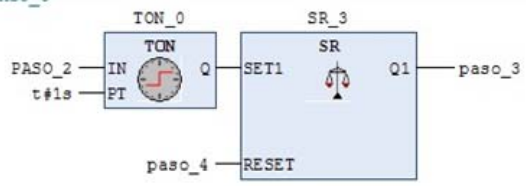
- A la hora de compilar en el **lenguaje FBD**, no se pueden tener pines sin asociar a una variable, o sea con **'???'**, ya que nos dará errores de compilación, por ello en aquellos que tengan ese problema, seleccionar la instrucción, botón derecho y seleccionar la opción **'Eliminar los parámetros de llamada de módulo de función que no se utilizan'** para que desaparezcan, si se quiere volver a ver este pin, habrá que seleccionar la opción **'Actualizar parámetros'**.



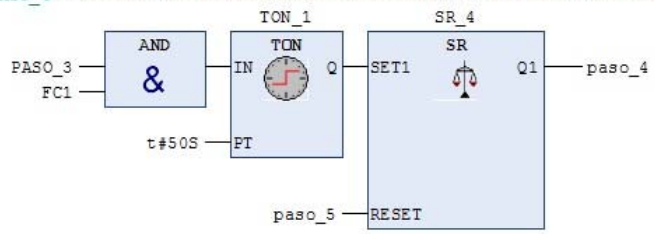
Teniendo en cuenta esto, realizaremos la programación como se ve a continuación:



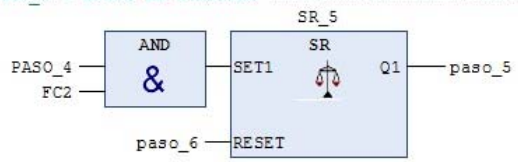
5 *PASO_3 **



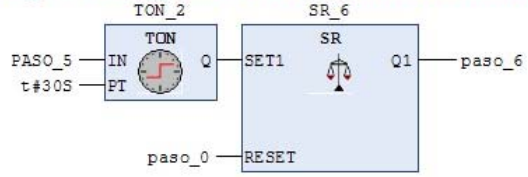
6 *PASO_4 **



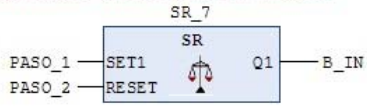
7 *PASO_5 - ETAPA DE LIMPIEZA **



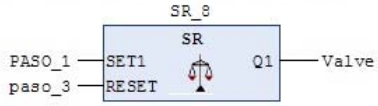
8 *PASO_6 - ETAPA DE LIMPIEZA **



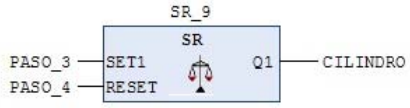
9 *ACTIVACION DE LA BOMBA DE ENTRADA **



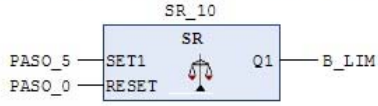
10 *ACTIVACION DE LA VALVULA DE ENTRADA ***



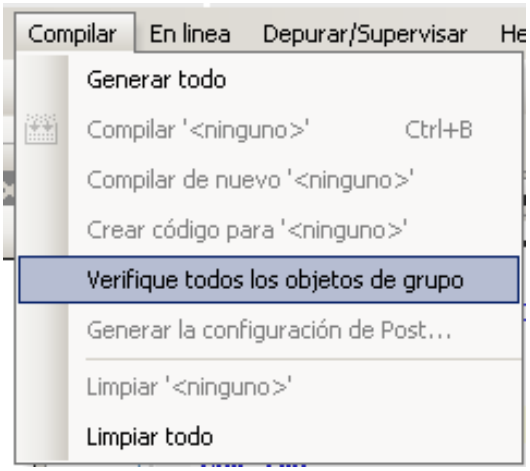
11 *ACTIVACION DEL CILINDRO HIDRAULICO ***



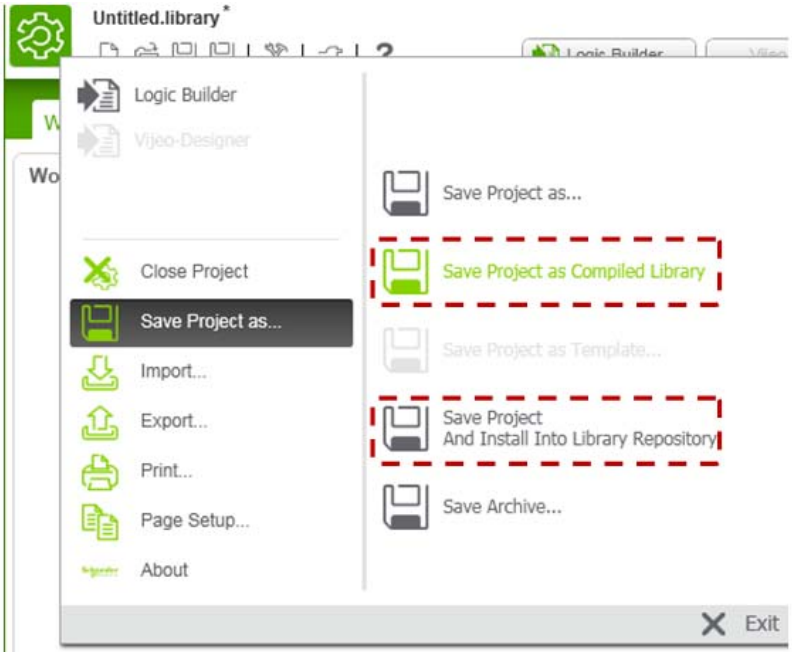
12 *ACTIVACION DEL BOMBA DE LIMPIEZA ***



Cuando se ha finalizado la programación del **FB**, hay que compilarlo sin errores. En el menú contextual seleccionar **'Compilar » Verificar todos los objetos del grupo'.**



Si se tiene la librería verificada y sin errores para guardarla, hay que ir la ventana general de SoMachine Central y se puede guardar de dos maneras.



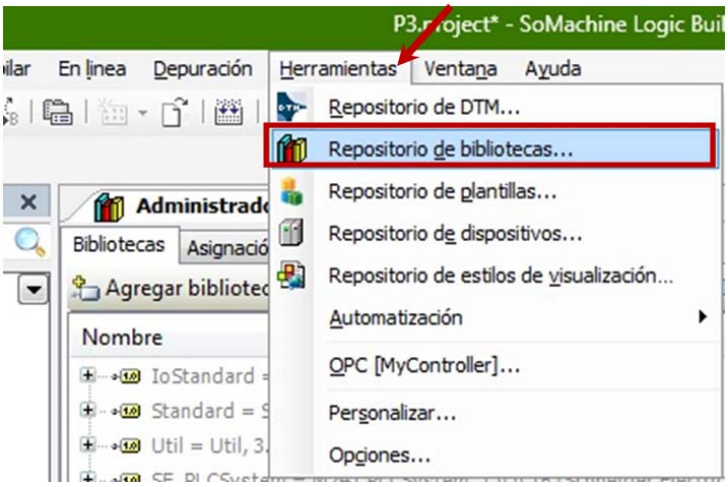
- **Guardar el proyecto como librería compilada'** que se puede seleccionar en el menú contextual de **'Archivo'**. Esta opción guarda la librería compilada, es decir, que el usuario no podrá ver el interior de los elementos de la librería (Librería encriptada).

- **Guardar el proyecto e instalarlo en el repositorio de bibliotecas'** que se puede seleccionar en el menú **'Archivo'** igual que el anterior. En este caso la librería ya se queda guardada en el repositorio de bibliotecas lista para ser agregada al proyecto, pero es una librería abierta, es decir, que el usuario podrá ver internamente los elementos de la librería.

A la hora de utilizar la librería de usuario generada en un proyecto se seguirá una acción u otra en función de cómo se ha grabado la librería.

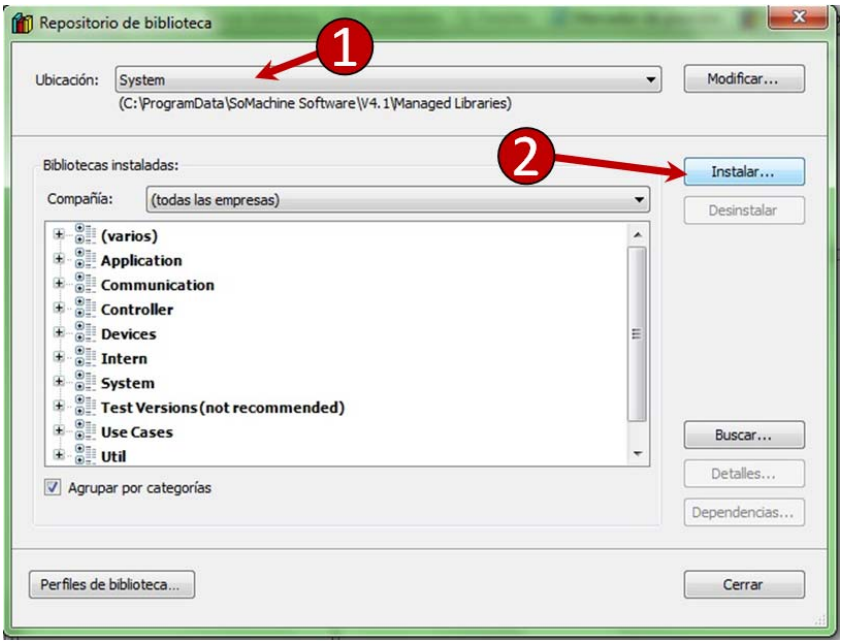
Para añadir una librería de usuario al repositorio de bibliotecas:

Si la librería de usuario se ha guardado con la opción **'Guardar el proyecto e instalarlo en el repositorio de bibliotecas'** simplemente hay que abrir el **'Administrador de bibliotecas'** del proyecto pulsar **'agregar biblioteca…'** y seleccionar la biblioteca en la carpeta **'Varios'** del **'Repositorio de bibliotecas'**.

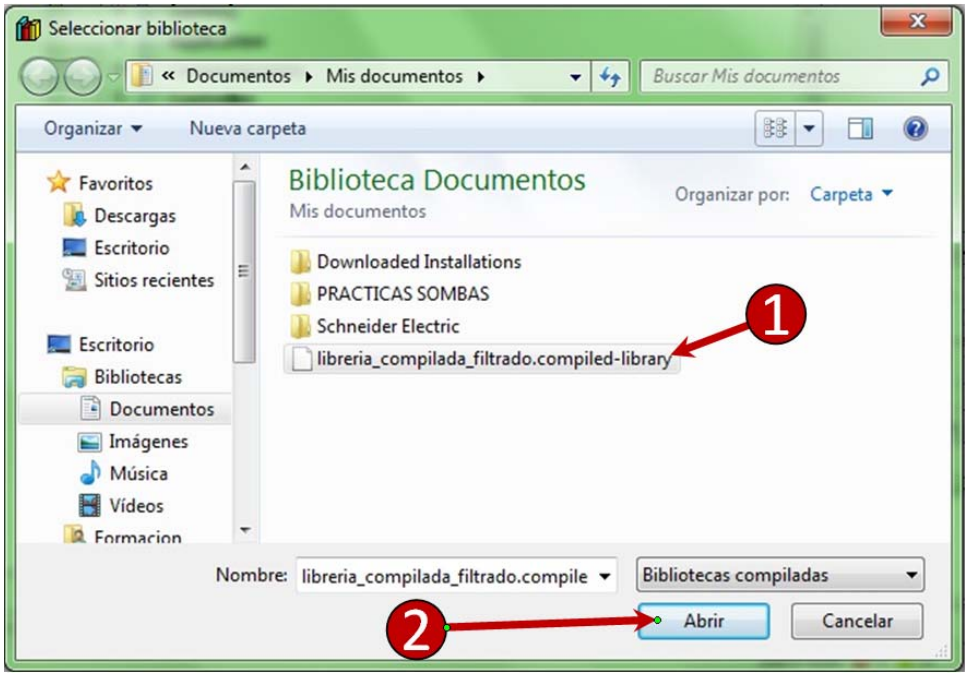


Si la librería se ha guardado con la opción **'Guardar el proyecto como librería compilada',** hay que instalarla previamente en el **'Repositorio de bibliotecas'** antes de ser agregada en el **'Administrador de bibliotecas'** del proyecto del usuario.

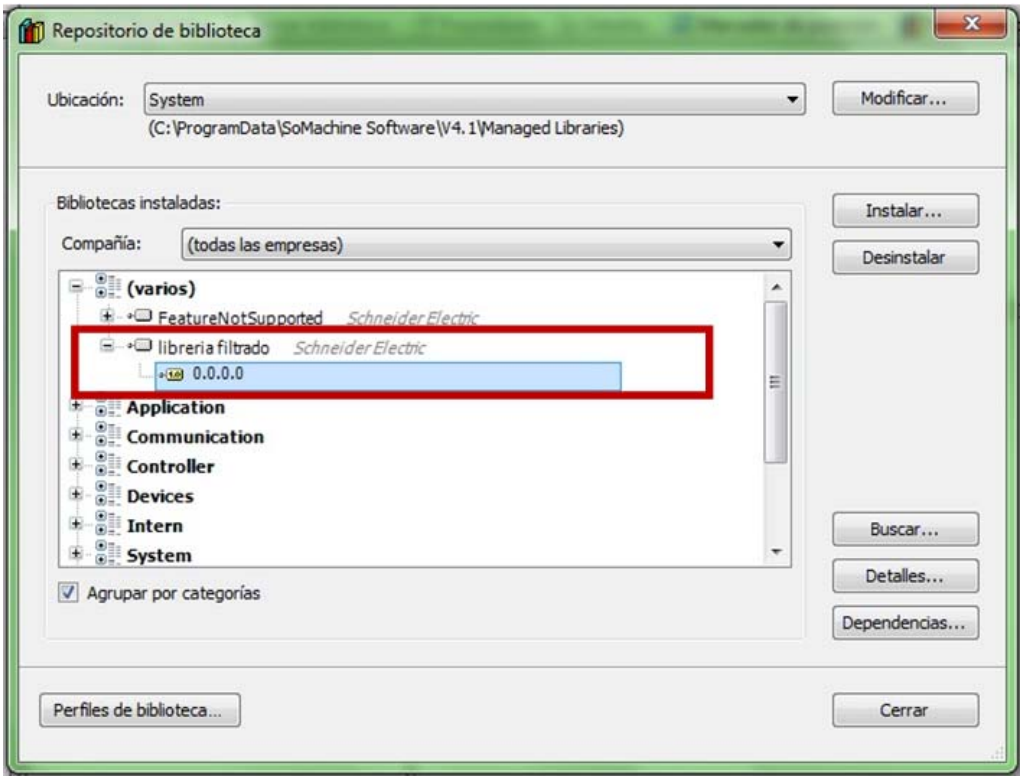
Para instalarla, dentro del **'Repositorio de Bibliotecas'**, en el campo **'Ubicación'** se selecciona **'System'** que será la ubicación donde la quedará la librería de usuario, una vez instalada. Para instalar la librería, pulsar el botón de **'Instalar'** en el 'Repositorio de bibliotecas'



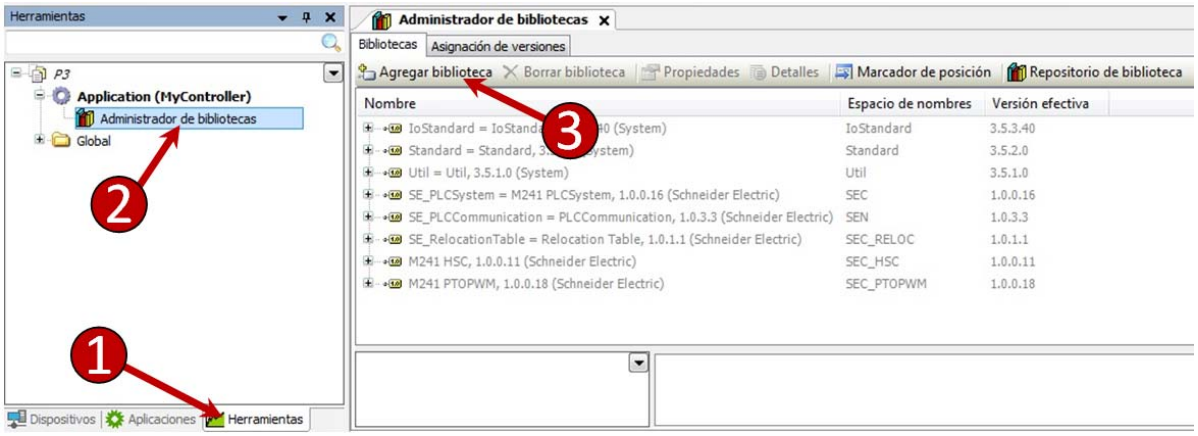
En la ventana flotante que aparece **'Seleccionar biblioteca'**, en el campo **'Tipo'** elegir el tipo **'Biblioteca compilada'** que busca la bibliotecas con el formato de archivo **'*. Compiled-library'**. Buscar en el navegador la ruta donde se había guardado previamente la librería y pulsar el botón **'Abrir'**.



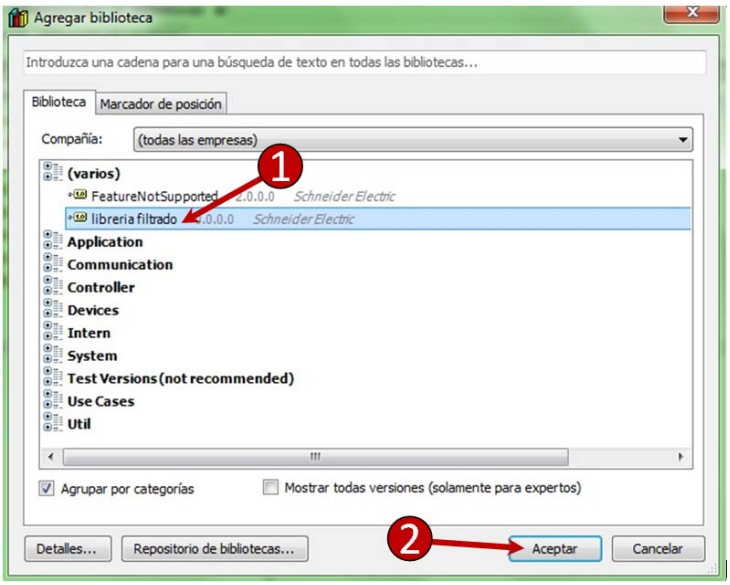
Ahora ya se puede encontrar la biblioteca en el repositorio de bibliotecas, si seleccionamos **'Todas las empresas'** y desplegamos en **'Varios'**, normalmente se encuentra en esa ubicación.



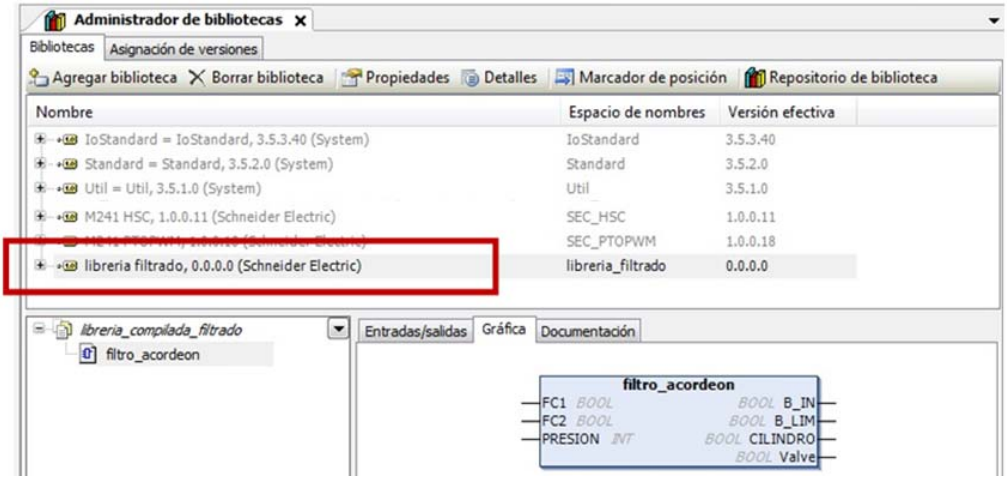
Para agregarla al proyecto, una vez agregada al repositorio, dentro de la pestaña **'Herramientas'** del navegador abriremos el **'Administrador de Bibliotecas'** y seleccionaremos la acción, **'Agregar biblioteca'** para añadirla a nuestro proyecto.



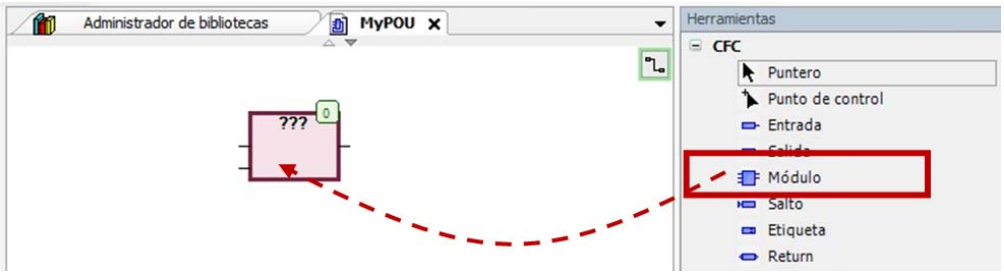
Aparece el '**Repositorio de Bibliotecas'** adonde abrimos **'Varios'** y seleccionamos la librería que nos interesa añadir a nuestro proyecto, en este caso **'librería filtrado'**.



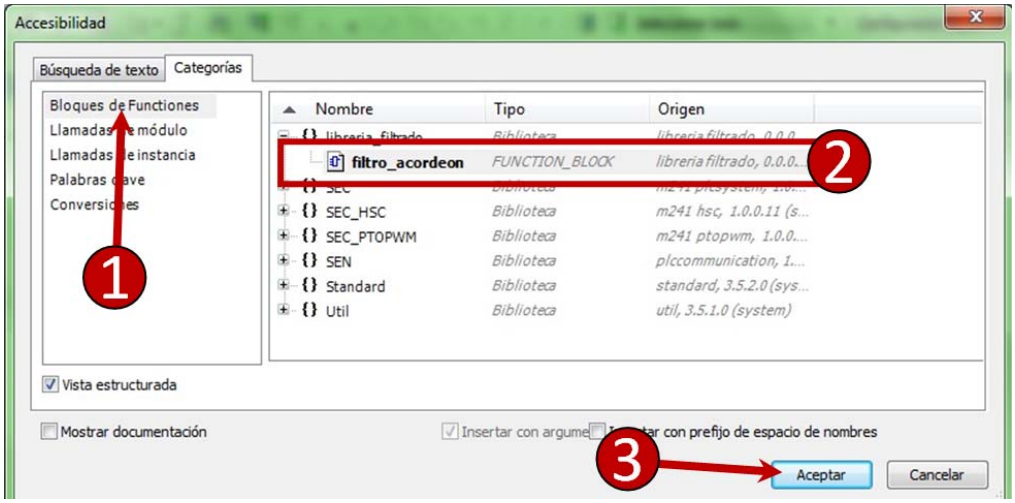
Ya se pueden utilizar los elementos de creados en la librería (FB's, DUT, Programas) como una instrucción más para la programación del proyecto.



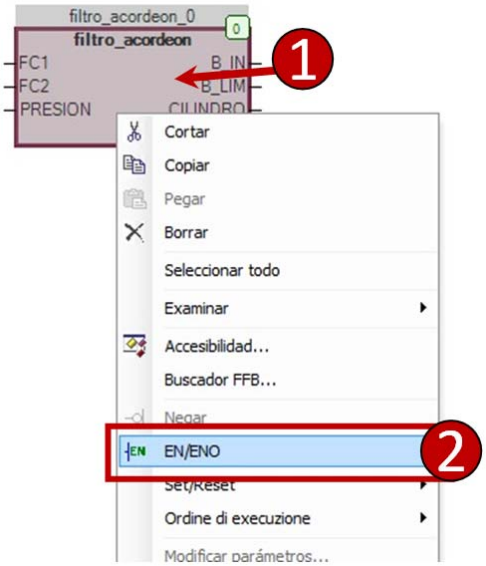
Para utilizar la FB que hemos creado, en nuestra biblioteca es tan sencillo como abrir el POU creado en el proyecto donde la queremos añadir, añadimos un **'módulo'** a la programación.



Seleccionamos el botón de accesibilidad que nos lleva a las librerías con los diferentes objetos a añadir a un módulo, seleccionamos en la parte izquierda '**Bloques de funciones'**, ya que nuestra instrucción es una FB, en la parte derecha buscamos nuestra librería **'librería filtrado'**, la desplegamos, para acabar seleccionando la FB **'Filtro_acordeon'**.



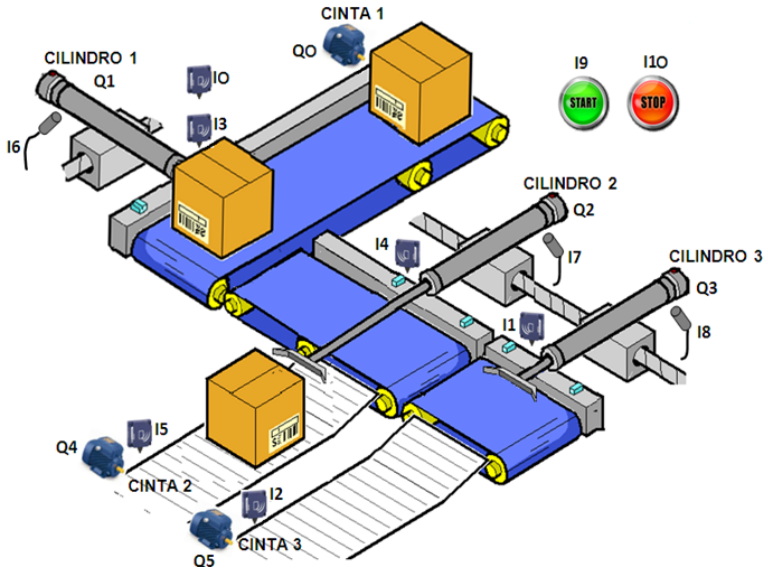
Una vez añadida la FB, pulsamos sobre ella y con el botón derecho añadimos el pin **'EN/ENO'** de habilitación de la instrucción.



Finalmente, añadimos creamos unas variables para probar el correcto funcionamiento de nuestra instrucción.

5.4.- PRACTICA 4: GRAFCET

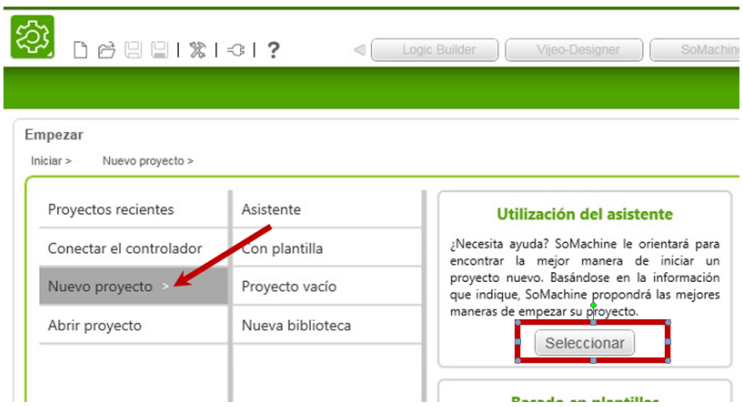
Un dispositivo automático destinado a escoger cajas de dos medidas diferentes, se compone de una cinta transportadora de entrada de las cajas, de tres cilindros de tipo monoestable y de dos cintas de evacuación que clasificaran los dos tipos de caja según el tamaño, tal y como muestra la figura adjunta.



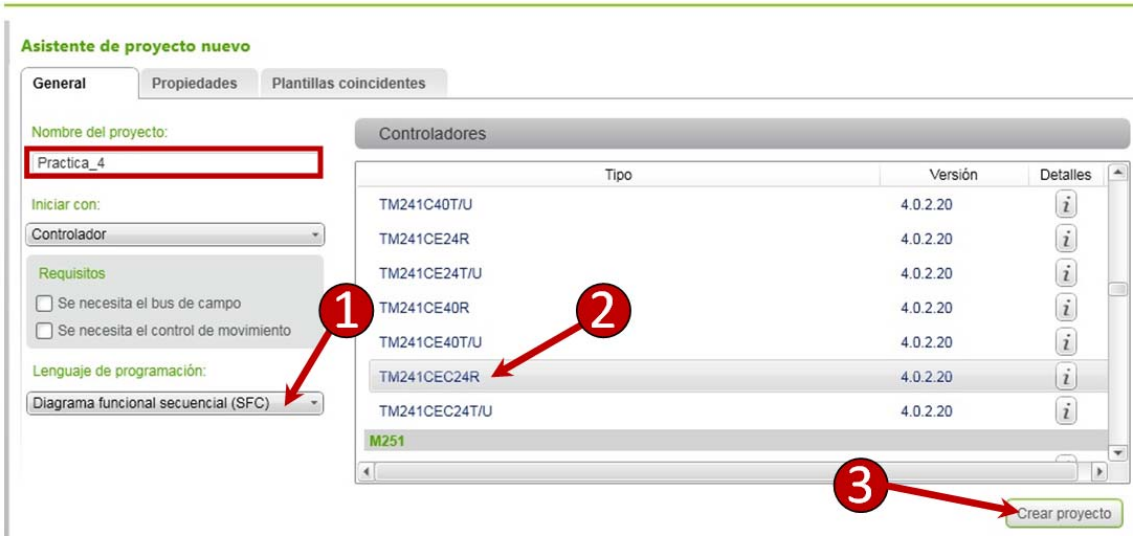
- El **cilindro 1** posiciona las cajas que llegan delante suyo a través de la **cinta número 1** de la siguiente forma: las cajas pequeñas se posicionan delante del **cilindro 2**, que a su vez las transfiere a la **cinta de evacuación número 2**; mientras que las cajas grandes se posicionan delante del cilindro número 3, que a su vez las transfiere a la cinta número 3.
- Para efectuar la selección de las cajas, un dispositivo de detección situado delante del cilindro 1 permite reconocer sin ambigüedades el tipo de caja que se presenta.
- Además, se quieren contar tanto el número total de cajas que recorren el montaje, como el número total de cajas pequeñas y el número total de cajas grandes por separado.

5.4.1.- Programación de la práctica:

Primero de todo tendremos que crear **un 'Nuevo proyecto'**, en este caso utilizaremos el asistente que nos ayudará a crear el proyecto y los primeros pasos de este.



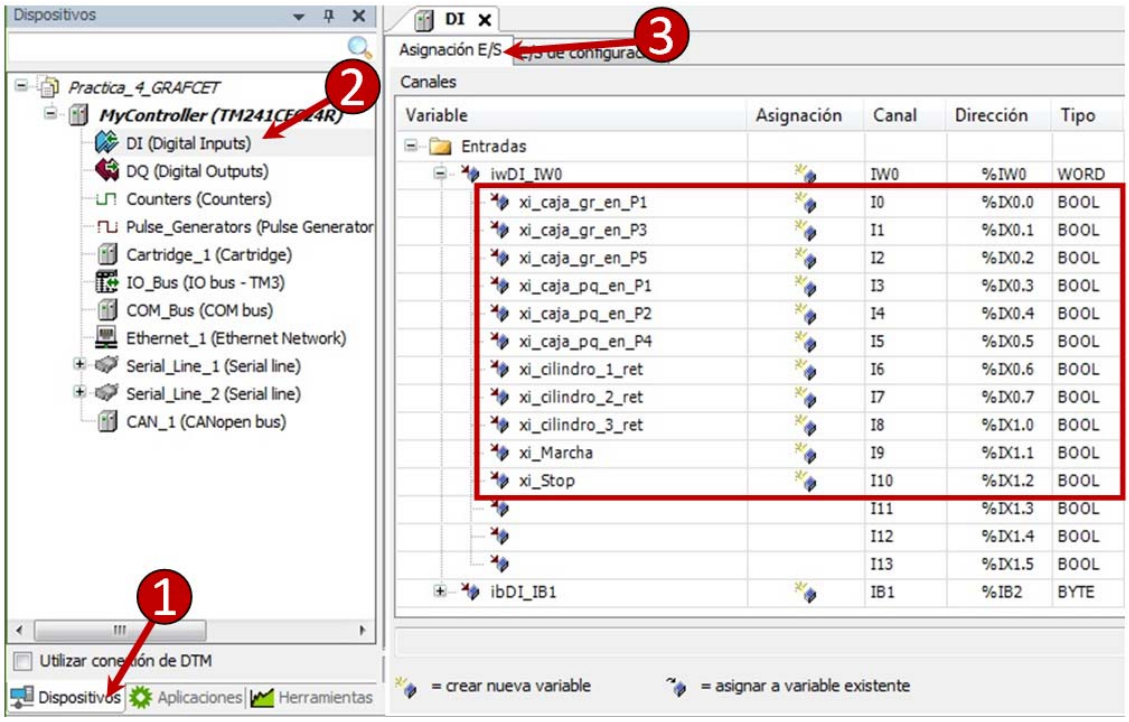
En la ventana del asistente de creación del proyecto dentro de la pestaña **'General'**, escribiremos el nombre de nuestro proyecto *(Practica_4)*, en **'Iniciar con'** seleccionaremos **'Controlador'** y en la parte de la derecha buscaremos el controlador deseado *(en este caso TM241CEC24R)*, por último en el '**lenguaje de programación'** seleccionamos el lenguaje con el queremos programar el primer POU *(en este caso Sequential Function Chart SFC)*. Por último pulsamos '**Crear Proyecto'** para empezar.



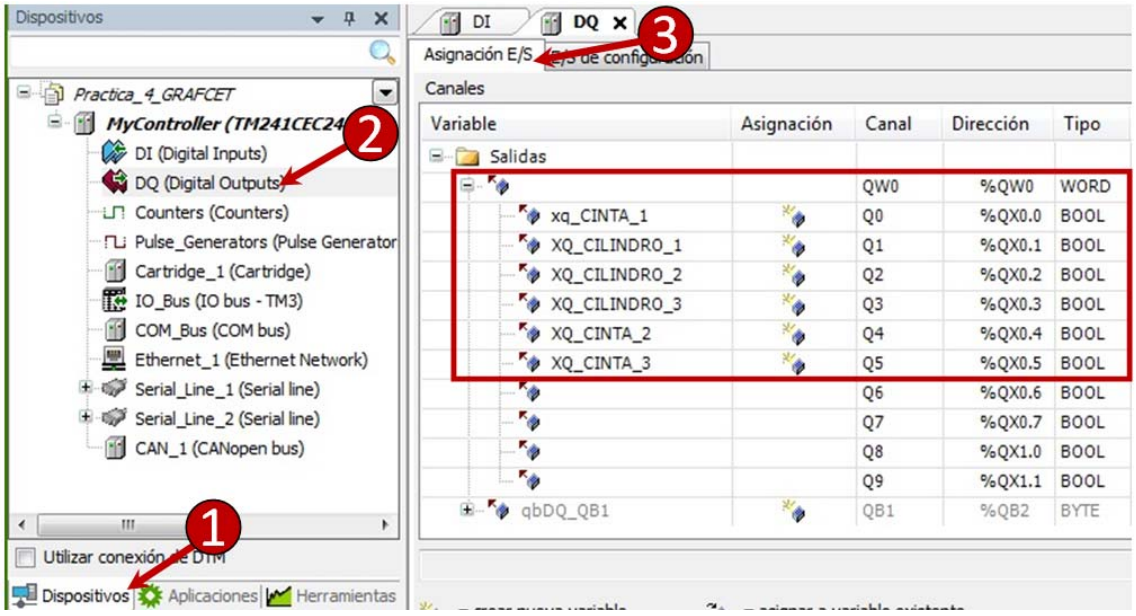
Cuando haya aparecido la ventana principal del SoMachine (*SoMachine Central*), nos aparece el Flujo de trabajo típico de un proyecto automatización, para entrar en la ventana de programación tendremos que pulsar el botón de **'Controlador'**, ya que dentro del flujo la configuración del controlador ya la hemos realizado previamente con el asistente.

Una vez entramos en la ventana de programación *(Logic Builder),* lo primero que tendremos que realizar, es la configuración de la entradas/salidas físicas que vamos a utilizar, para ello, tendremos que ir a la pestaña de **'Dispositivos'** del navegador del proyecto, que se encuentra en la parte izquierda de la pantalla.

Dentro, maximizaremos el controlador, y haremos doble click sobre las entradas digitales '**DI (Digital Inputs)'**. En el área de trabajo aparecerá las pestañas de configuración de las entradas digitales que hay embebidas en el M241, seleccionamos la pestaña **'Asignación E/S'** y escribimos las diferentes símbolos de las entradas.

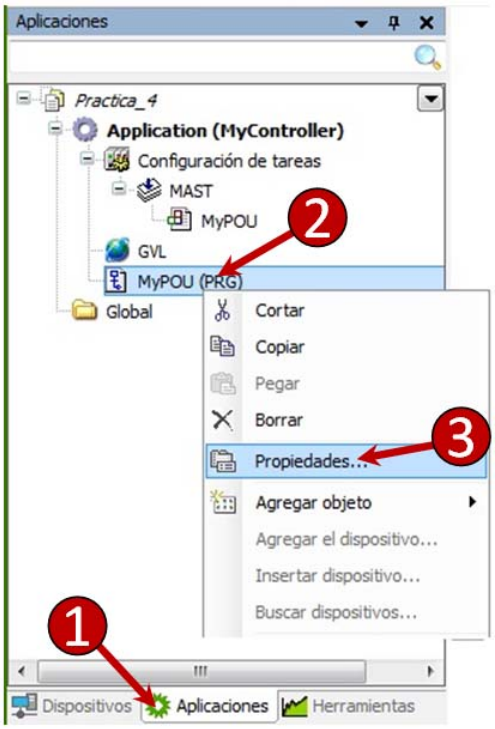


Ahora entraremos en la configuración de las salidas físicas, tendremos que ir a la pestaña de **'Dispositivos'** del navegador del proyecto, y haremos doble click sobre las salidas digitales '**DQ (Digital Outputs)'**. En el área de trabajo aparecerá las pestañas de configuración de las entradas digitales que hay embebidas en el M241, seleccionamos la pestaña **'Asignación E/S'** y escribimos en las salidas.

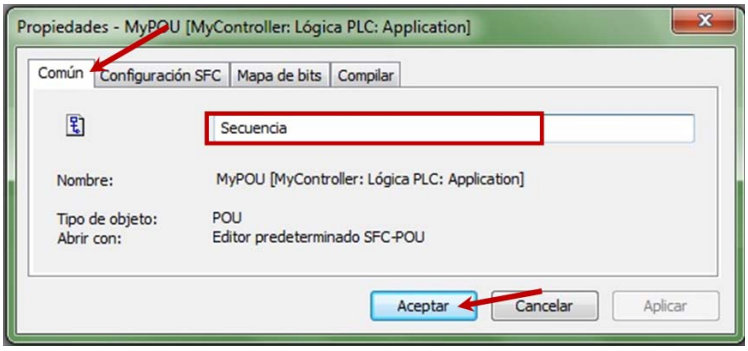


Una vez configurados las entradas y salidas, pasamos a programar seleccionando la pestaña **'Aplicación'** del navegador de proyectos. Desplegamos la '**Application'** y

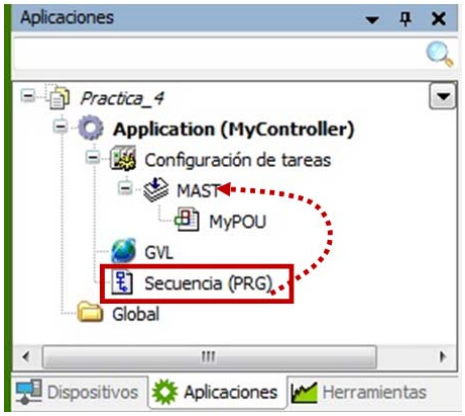
seleccionamos **‘MyPou’**, con el botón derecho seleccionamos la opción **‘Propiedades’**.



Dentro de la ventana, cambiamos el nombre del POU por el de **‘Secuencia’**.

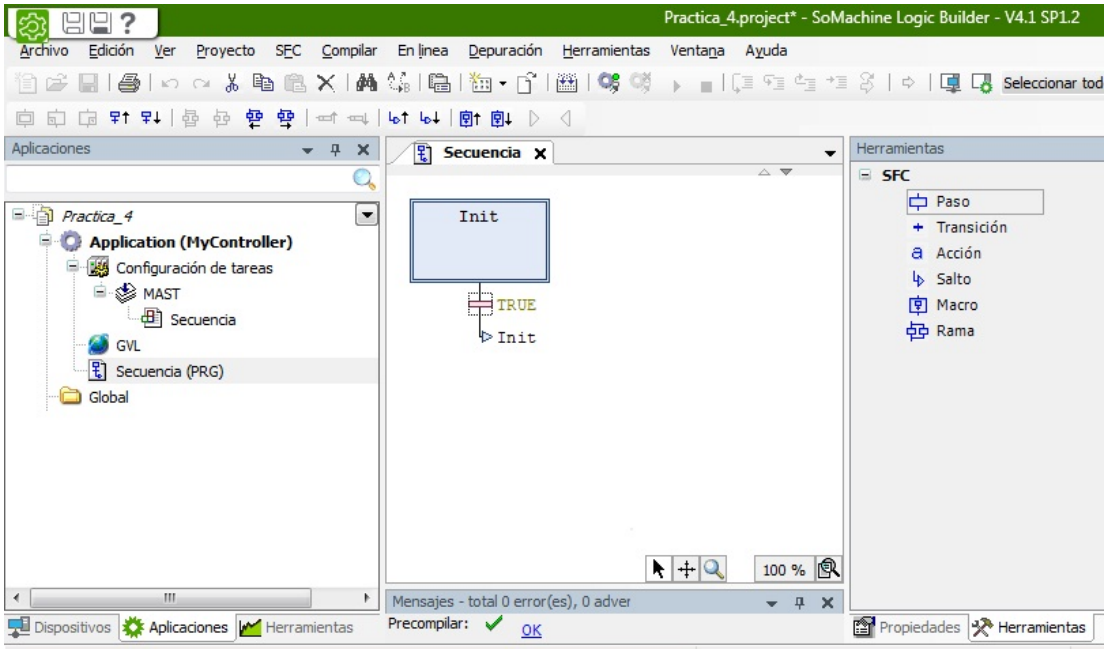


Volvemos a seleccionar el POU **‘Secuencia’** y arrastramos manteniéndolo pulsado, hasta dejarlo encima de la tarea **‘MAST’**, de esta manera el POU queda asociado a la tarea.

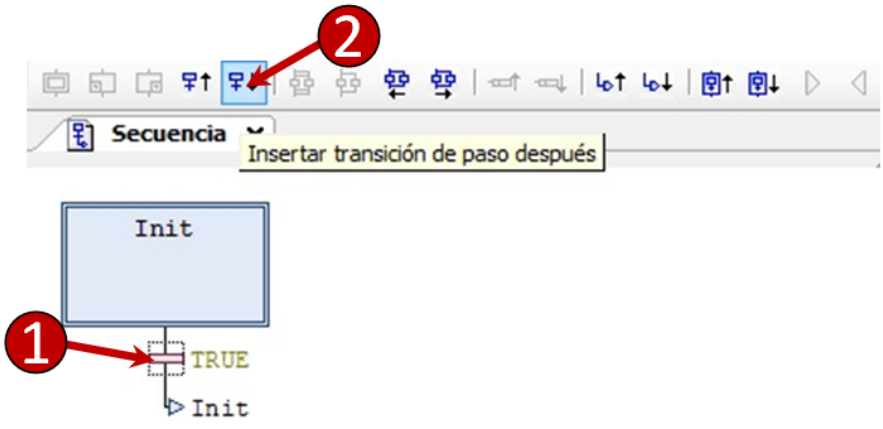


Quedando el link debajo de la tarea, los POU's que se van a ejecutar asociados a ella, el orden de ejecución de los POU's en la tarea será, de arriba abajo.

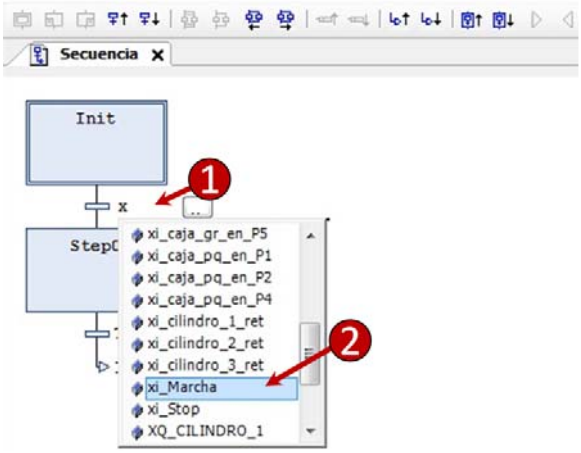
Ahora para empezar a programar el POU's hacemos doble click sobre él y aparece en el área central de trabajo. Por defecto cuando hemos creado un POU's con el lenguaje de programación SFC, nos aparece la primera etapa **'Init'**.



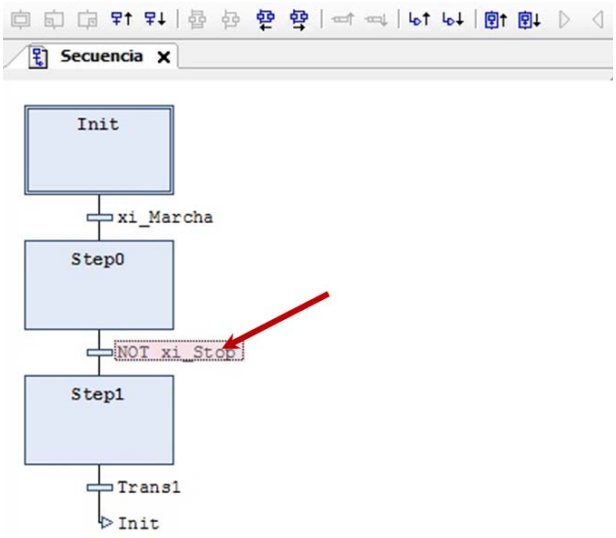
Seleccionamos la primera transición, y sobre ella añadimos una etapa seleccionándola en la barra de herramientas de programación del lenguaje SFC.



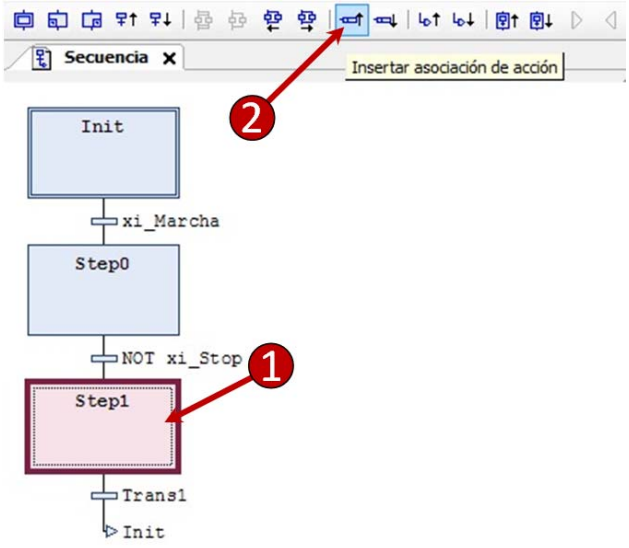
Pulsamos en la primera transición y cambiamos la condición de transición por la variable **‘xi_Marcha’**.



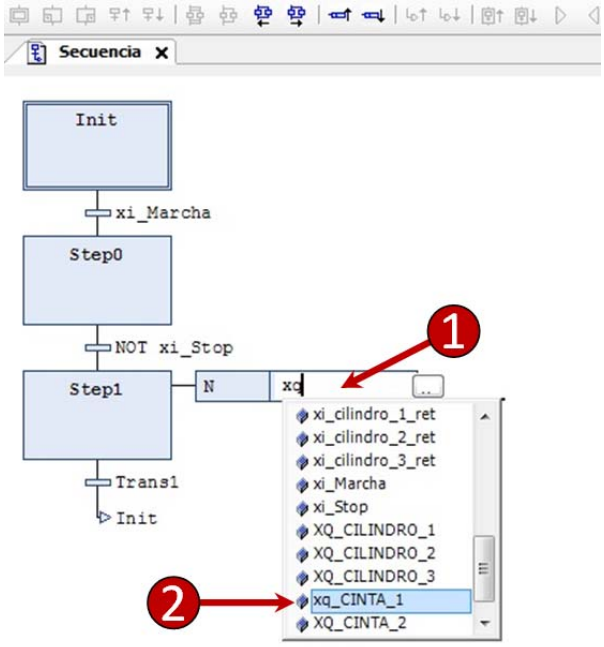
Añadimos una etapa nueva y en la segunda transición, insertamos la variable **‘xi_Stop’** como condición de transición y la negamos poniendo la palabra clave **‘NOT’**.



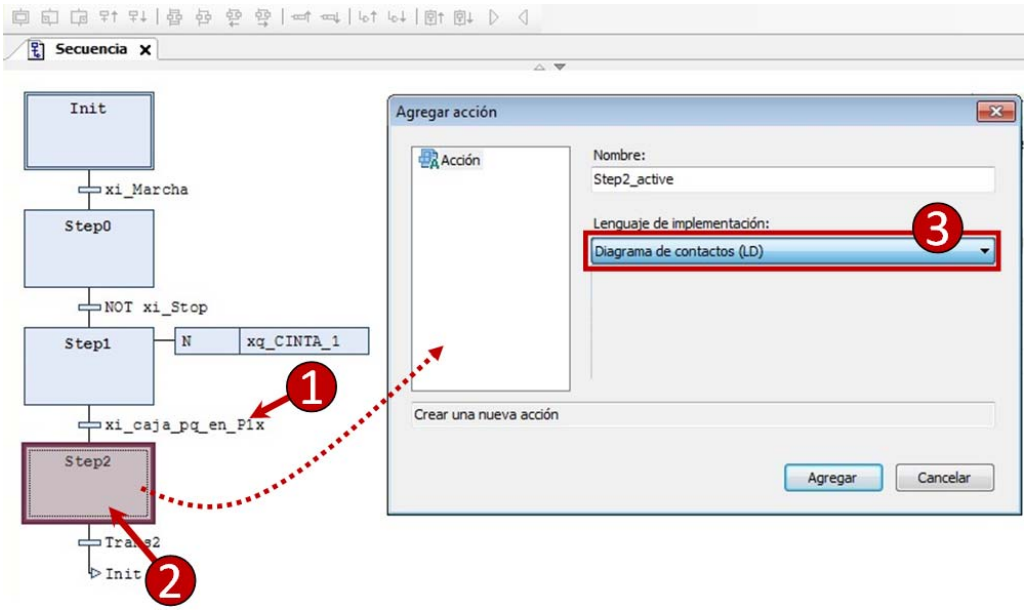
En la etapa 1 **‘Step1’**, la seleccionamos y le asociamos una acción a la etapa, desde la barra de herramientas.



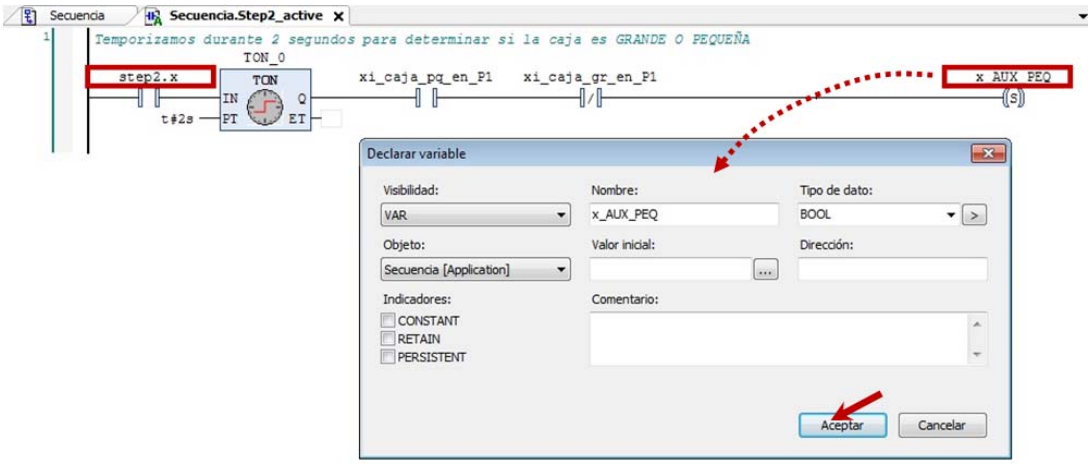
Una vez añadida la acción a la etapa, seleccionamos la variable en la cual se va a realizar la acción, en este caso **'xq_CINTA1'**.



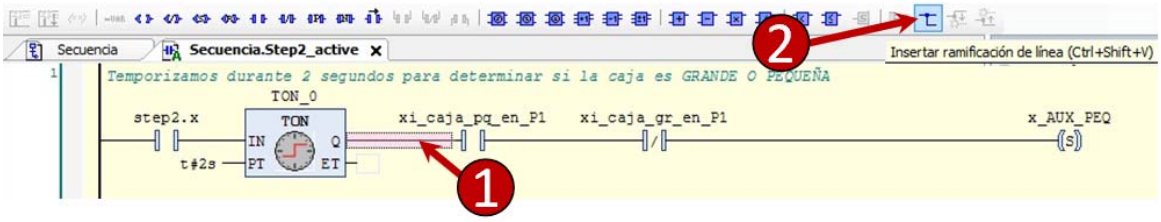
Añadimos una nueva etapa **'Step2'**, ponemos la variable **'xi_caja_pq_en_P1'**, como condición de transición previa. Ahora hacemos doble click sobre la etapa y creamos una acción asociada **'Step2_active'**, que se ejecutará cuando la etapa esté activa, el lenguaje de programación que se utilizará para la programación de esta acción es el **'Diagrama de contactos LD'**.



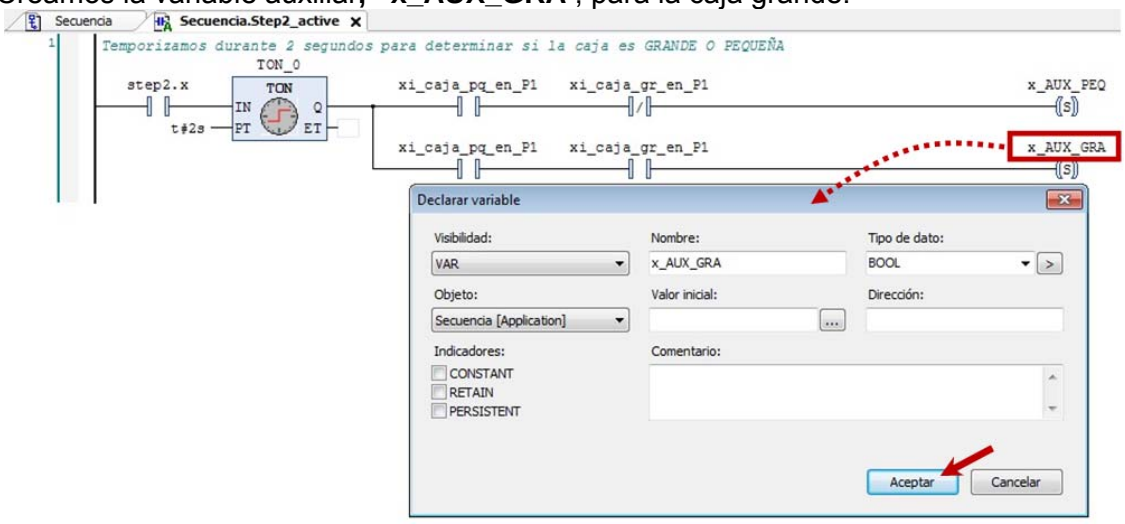
Realizamos la programación de la acción, cuando la etapa se activa (utilizamos la variable implícita **'Step2.x'**) temporiza 2 segundos para determinar si la caja es grande o pequeña. Creamos la variable auxiliar **'x_AUX_PEQ'** para la caja pequeña.



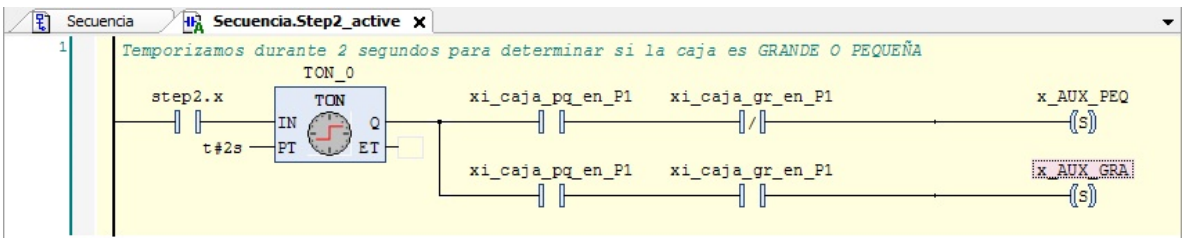
Añadimos una ramificación de línea.



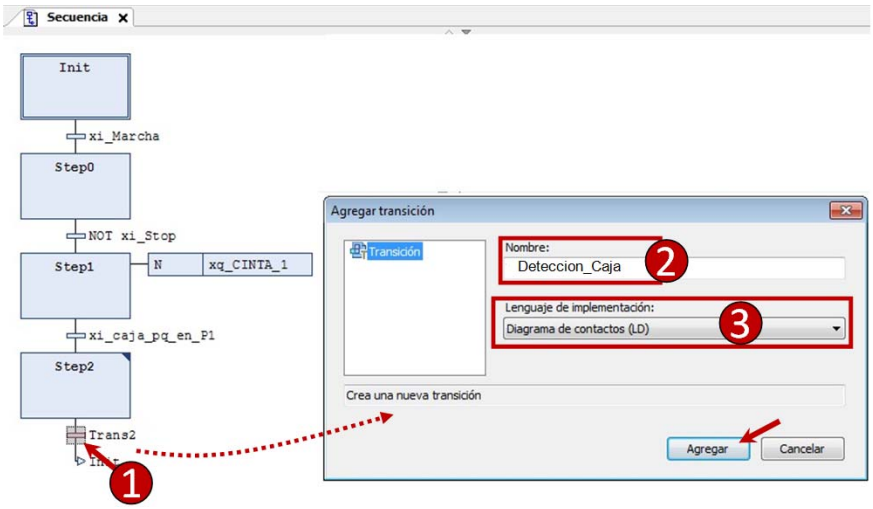
Creamos la variable auxiliar**, 'x_AUX_GRA'**, para la caja grande.



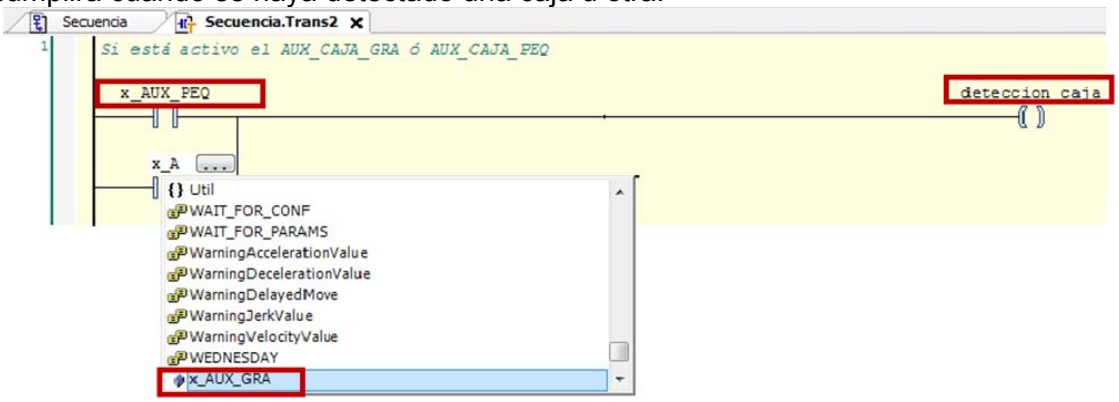
El resultado de la programación de la acción en el paso 2 **'Step_2'**, será el siguiente:



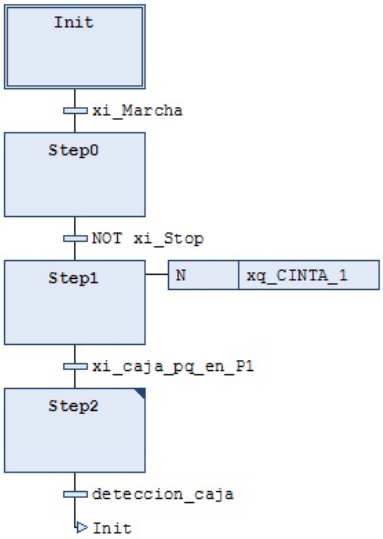
En la transición que hay por debajo de la etapa 2, hacemos doble clic sobre ella y aparece la ventana para poder programar la transición, para ello le cambiamos el nombre a la transición por **'Deteccion_Caja'**, y el lenguaje de programación que se utilizará para la programación de esta transición es el **'Diagrama de contactos LD'**.



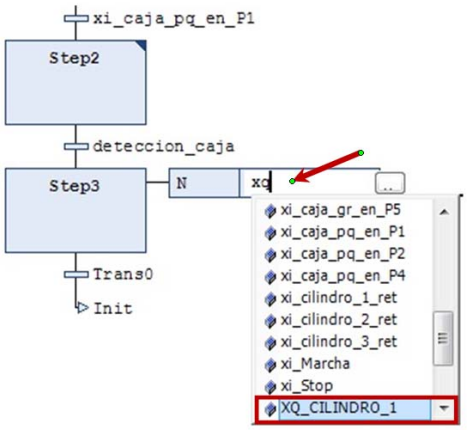
Realizamos la programación de la transición, donde la condición de la transacción se cumplirá cuando se haya detectado una caja u otra.



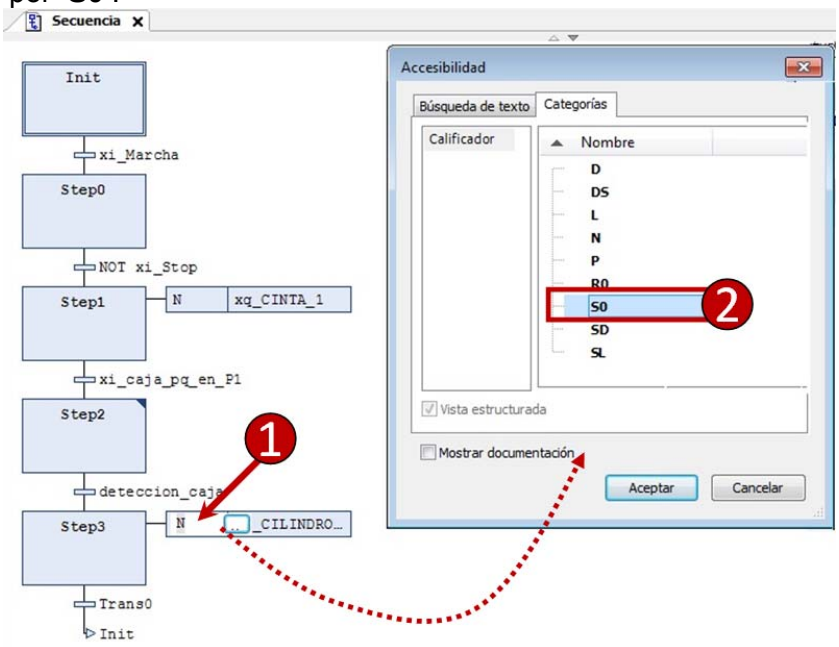
De momento la programación hasta ahora, queda de la siguiente manera.



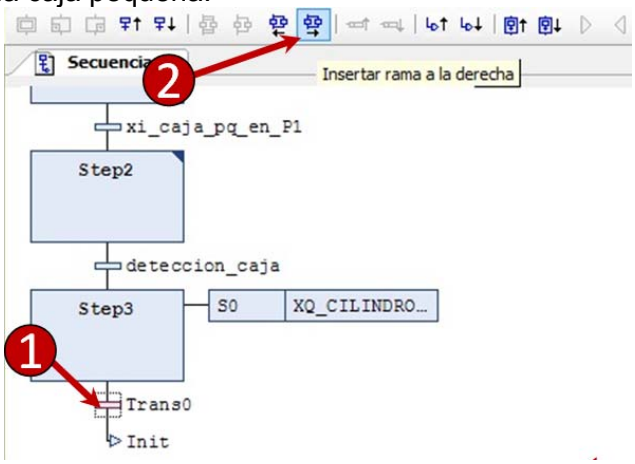
Añadimos una nueva etapa, y en esta añadimos una acción para esta etapa **'Step3'**, seleccionamos la variable en la cual se va a realizar la acción, en este caso **'xq_CILINDRO_1'**.



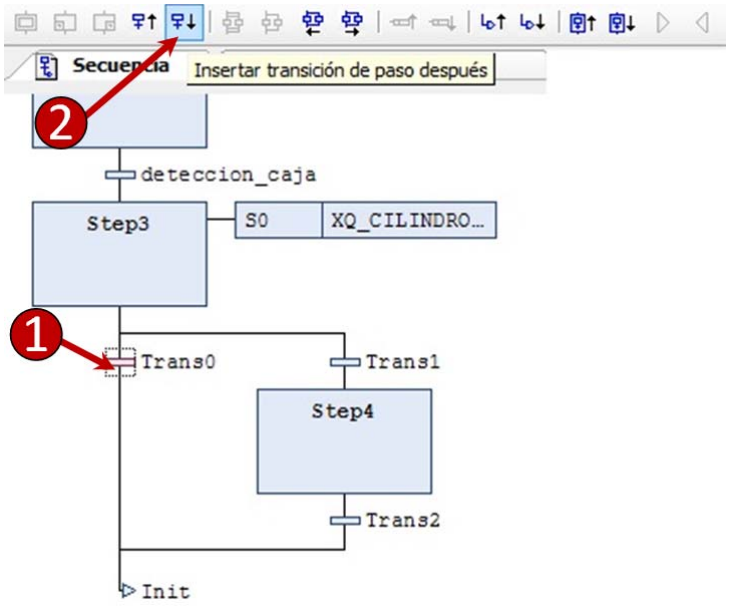
A esta acción le cambiamos el descriptor por defecto y en vez del descriptor **'N'** lo cambiamos por **'S0'**.



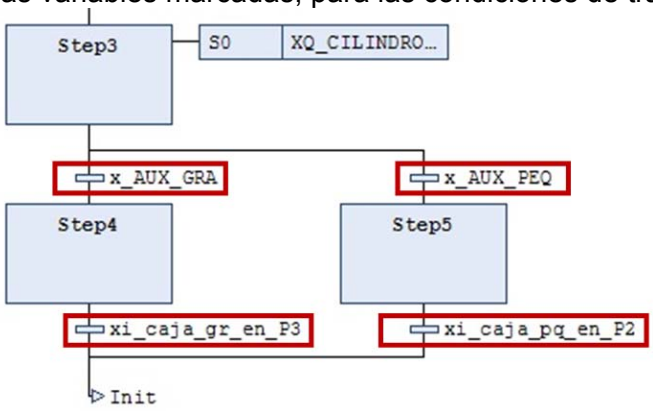
Ahora seleccionamos la transición debido del **'Step3'** y añadimos **'Insertar rama a la derecha'** de la barra de herramienta para crear una bifurcación, una para la caja grande y una para la caja pequeña.



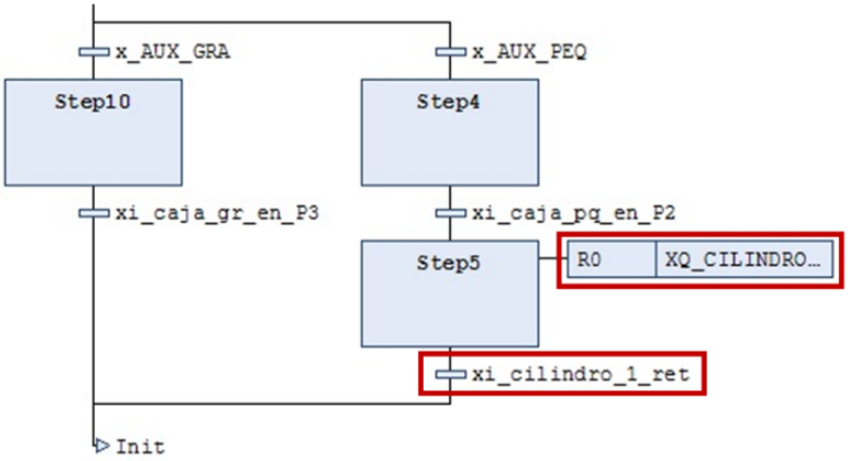
Seleccionamos la transición de la izquierda de la ramificación y añadimos una etapa nueva.



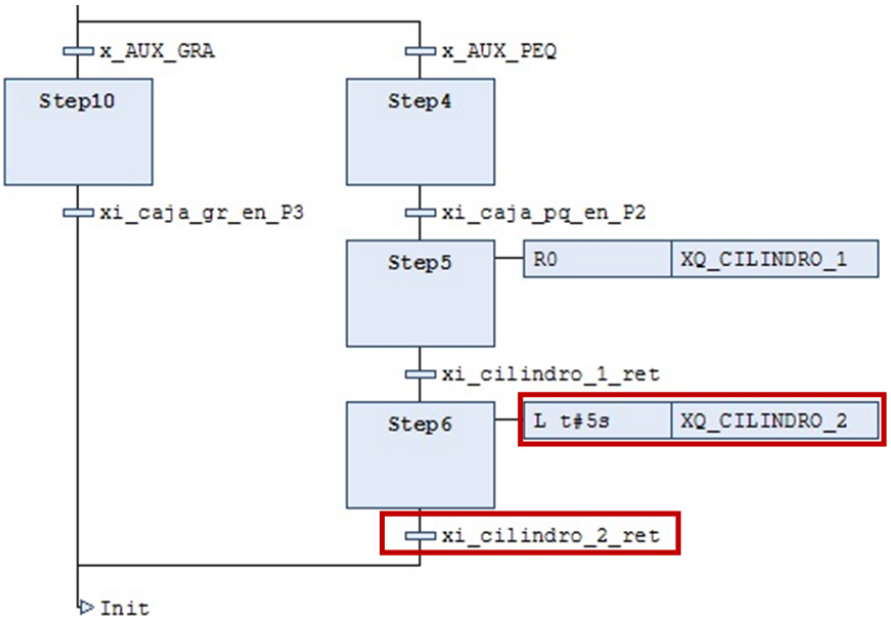
Ahora agregamos las variables marcadas, para las condiciones de transición.



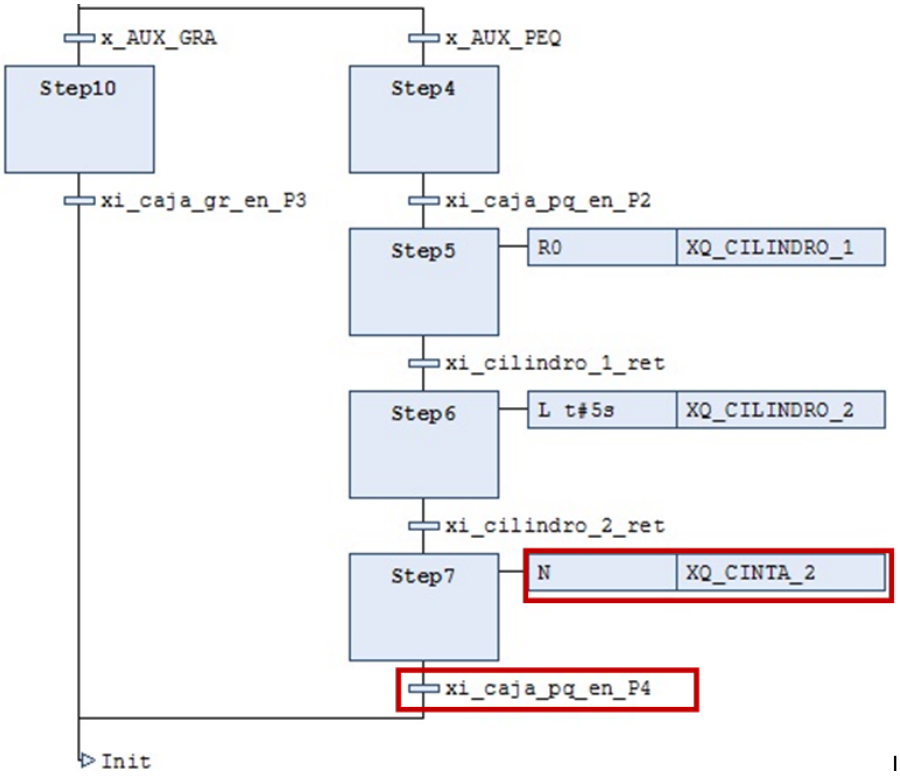
Agregamos una etapa nueva en la rama derecha, y en esta una acción en la etapa 5 **'Step5'**, donde resetearemos la variable **'XQ_CILINDRO_1'**, teniendo que cambiar el descriptor por el **'R0'** para que sea un reset. Añadimos la variable **'xi_cilindro_1_ret'** a la condición de transición.



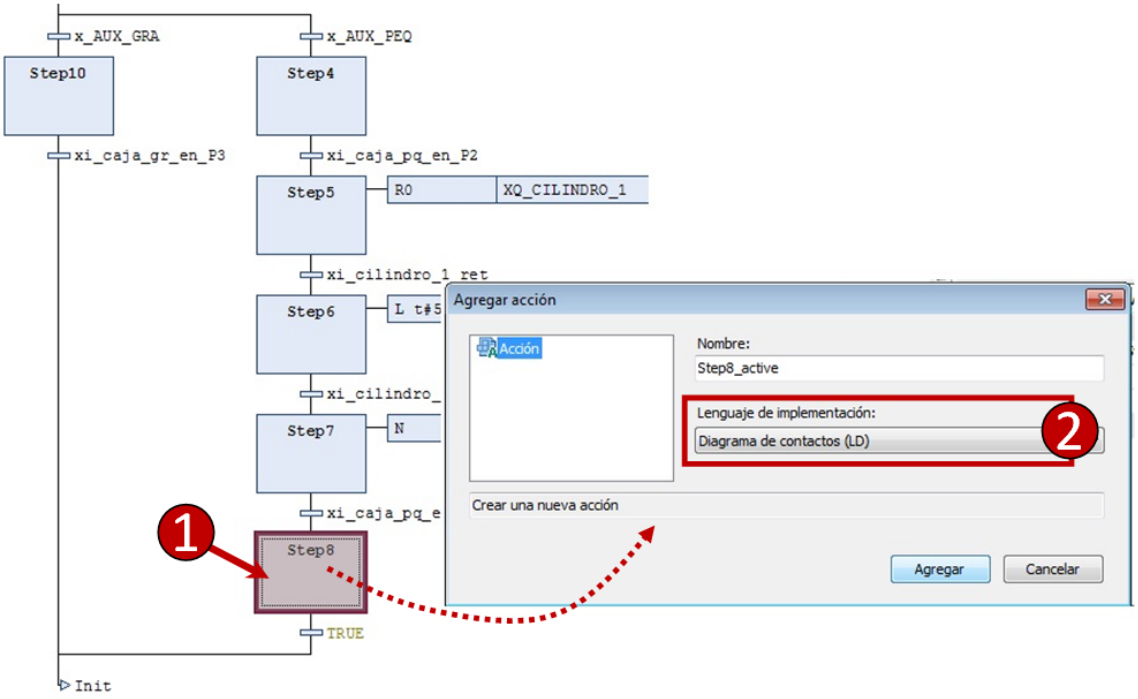
Agregamos una etapa nueva en la rama derecha, y en esta una acción en la etapa 6 **‘Step6’**, donde haremos una activación temporizada de variable **‘XQ_CILINDRO_2’**, teniendo que cambiar el descriptor por el **‘L t#5s’** para que sea una activación temporizada de 5 segundos. Añadimos la variable **‘xi_cilindro_2_ret’** a la condición de transición de debajo.



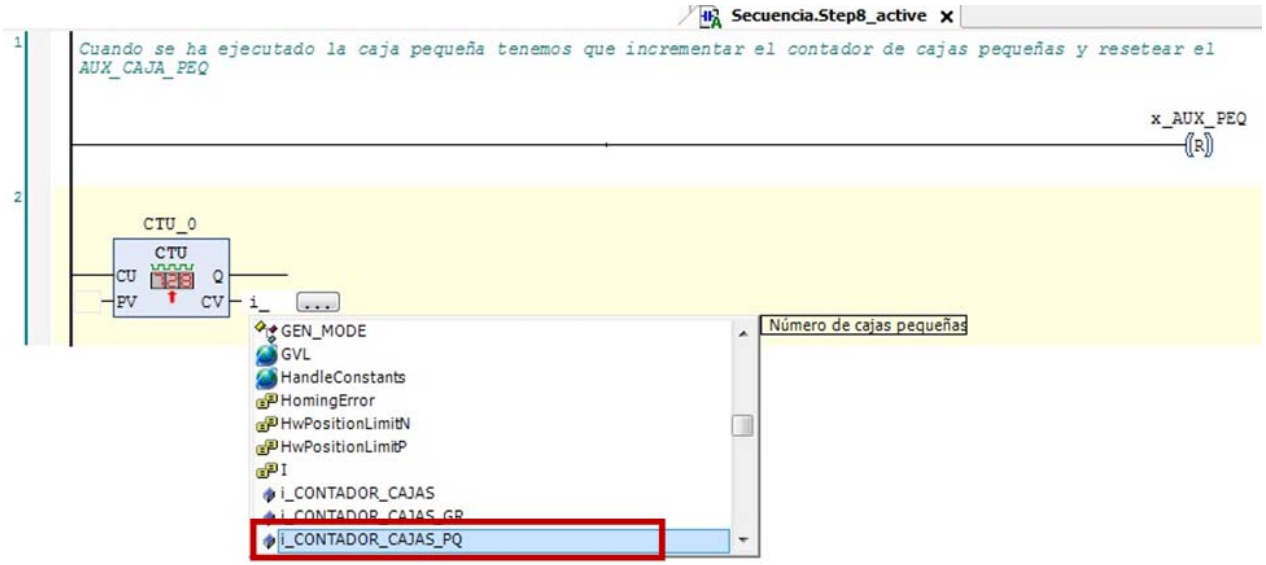
Continuamos agregando una etapa nueva en la rama derecha. En la etapa agregamos una acción en la etapa 7 **‘Step7’**, donde haremos una activación de la variable **‘XQ_CINTA_2’**. Añadimos la variable **‘xi_caja_pq_en_P4’** a la condición de transición de debajo.



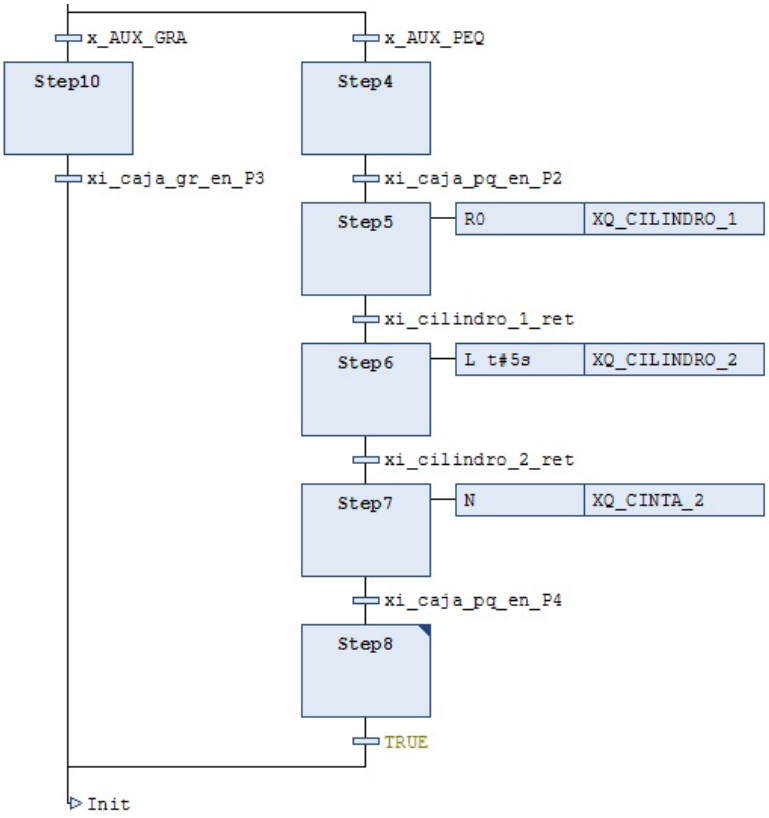
Por último, agregamos una última etapa nueva en la rama derecha. En la etapa agregamos una programa en la etapa 8 **'Step8'**, haciendo doble click sobre la etapa y creamos una acción asociada **'Step8_active'**, que se ejecutará cuando la etapa esté activa, el lenguaje de programación que se utilizará para la programación de esta acción es el **'Diagrama de contactos LD'**.



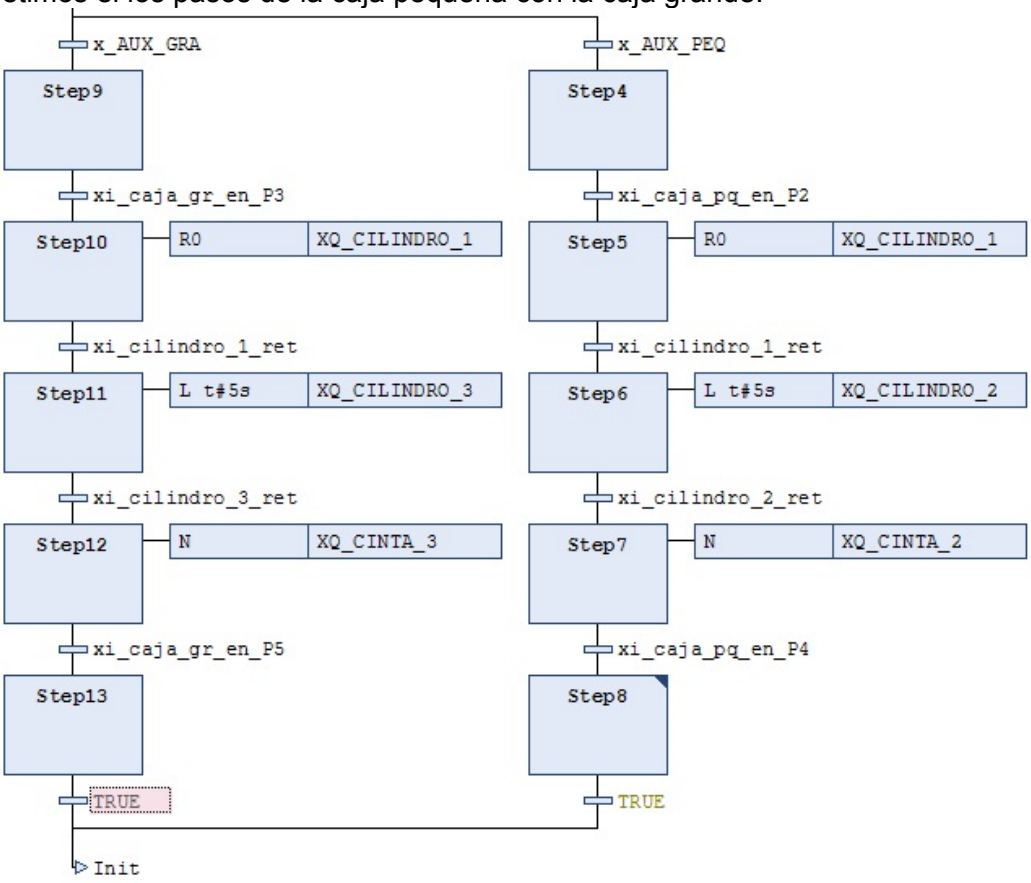
Realizamos la programación de la acción, cuando la etapa se activa reseteamos el auxiliar de la caja pequeña porque la caja ya ha sido procesada e incrementamos el contador de las cajas pequeñas asociando al pin **'CV'** del contador a la variable **'i_CONTADOR_CAJAS_PQ'**.



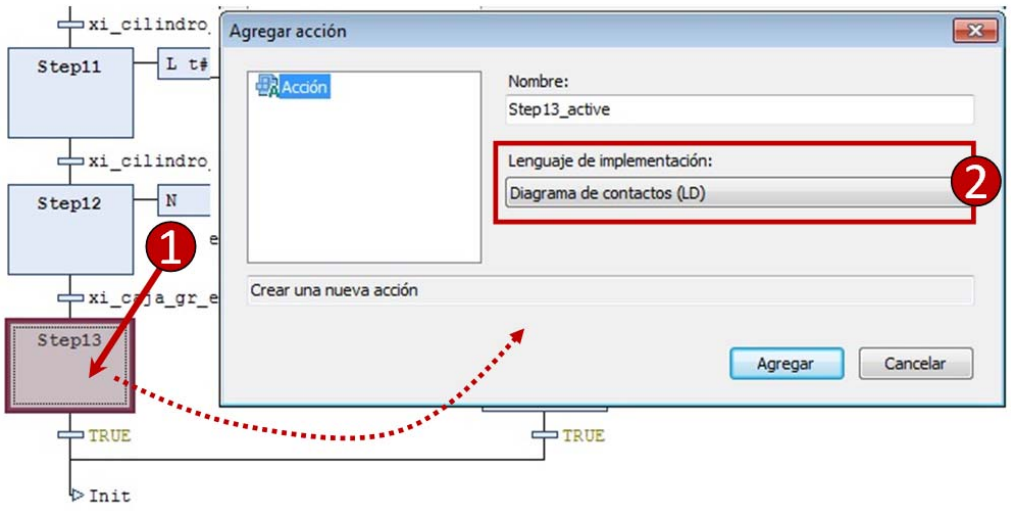
Al final la rama que procesa la caja pequeña queda la siguiente manera.



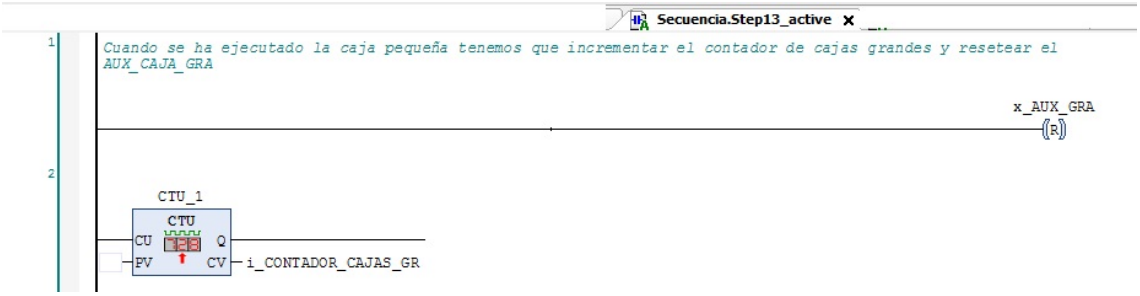
Repetimos el los pasos de la caja pequeña con la caja grande.



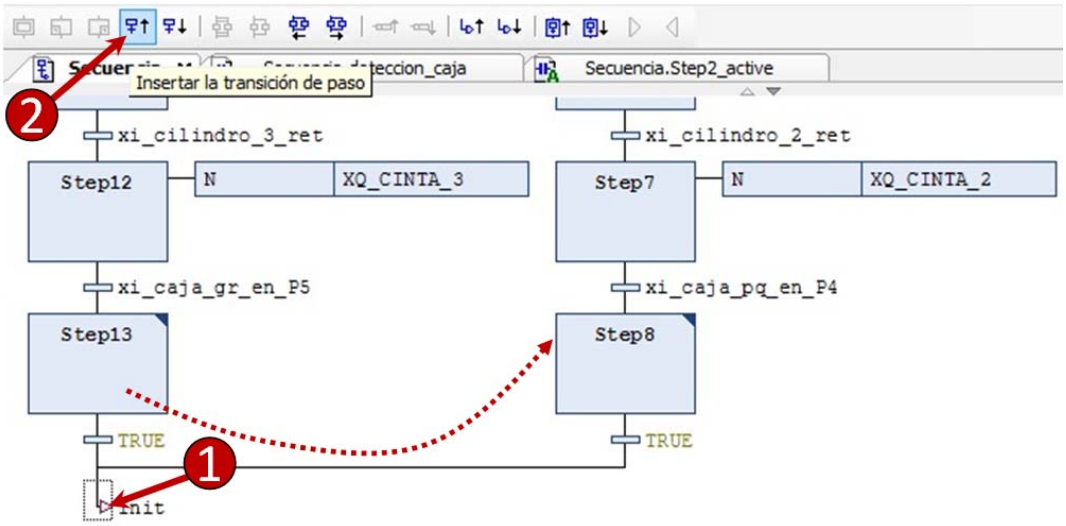
En la etapa agregamos una programa en la etapa 13 **'Step13'**, haciendo doble click sobre la etapa y creamos una acción asociada **'Step13_active'**, que se ejecutará cuando la etapa esté activa, el lenguaje de programación que se utilizará para la programación de esta acción es el **'Diagrama de contactos LD'**.



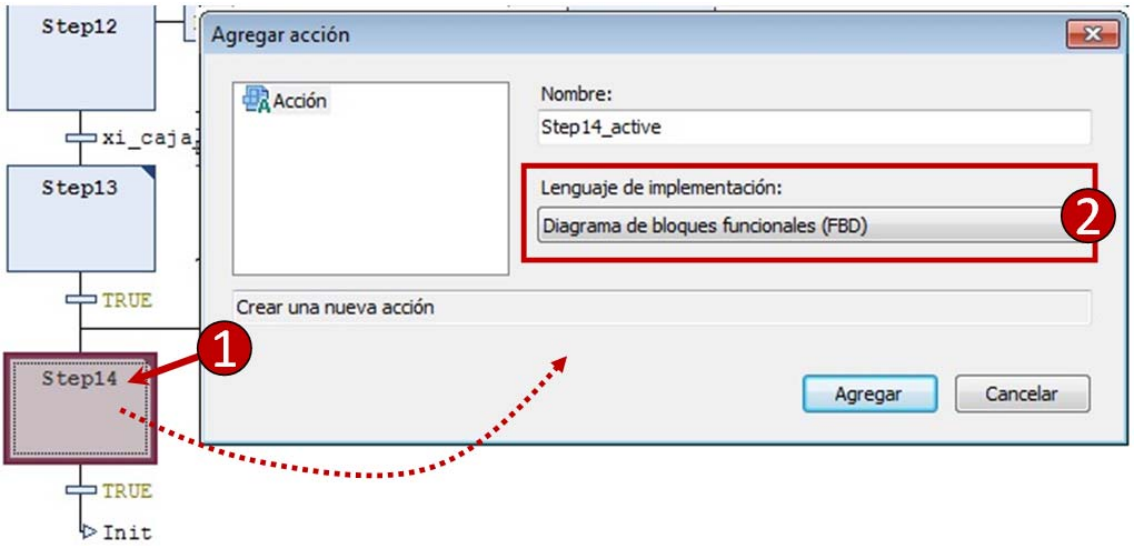
Realizamos la programación de la acción, cuando la etapa se activa reseteamos el auxiliar de la caja grande porque la caja ya ha sido procesada e incrementamos el contador de las cajas grandes, asociando al pin **'CV'** del contador a la variable **'i_CONTADOR_CAJAS_GR'**.



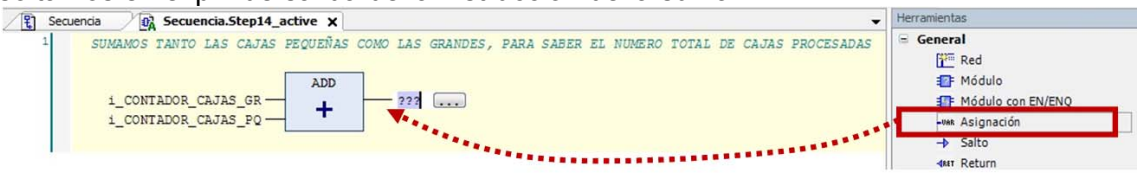
Una vez cerrado las dos bifurcaciones (caja grande y caja pequeña), añadimos una etapa nueva, seleccionando el salto **'Init'** y seleccionando **'Insertar transición de paso'** en la barra de herramientas.



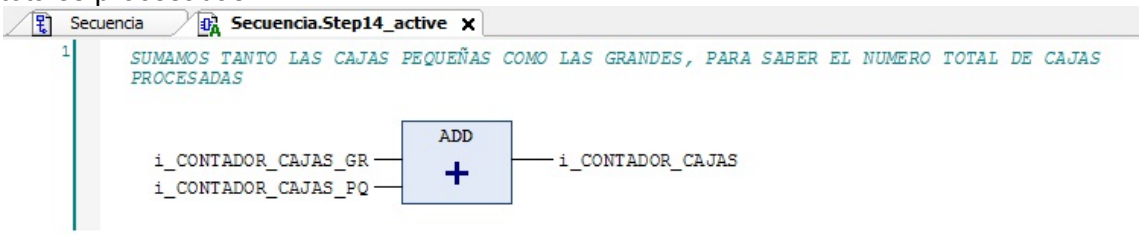
En la etapa agregamos una programa en la etapa 14 **‘Step14’**, haciendo doble click sobre la etapa y creamos una acción asociada **‘Step14_active’**, que se ejecutará cuando la etapa esté activa, el lenguaje de programación que se utilizará para la programación de esta acción es el **‘Diagrama de bloques funcionales FBD’**.



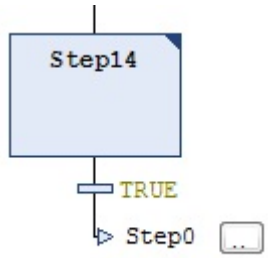
Añadimos un módulo de suma **‘ADD’** y sumamos las cajas pequeñas y las cajas grandes, seleccionamos **‘Asignación’** en las herramientas y manteniéndolo pulsado lo soltamos en el pin de salida de la instrucción de la suma.



Añadimos la variable **‘i_CONTADOR_CAJAS’** al pin de salida para saber las cajas totales procesadas.

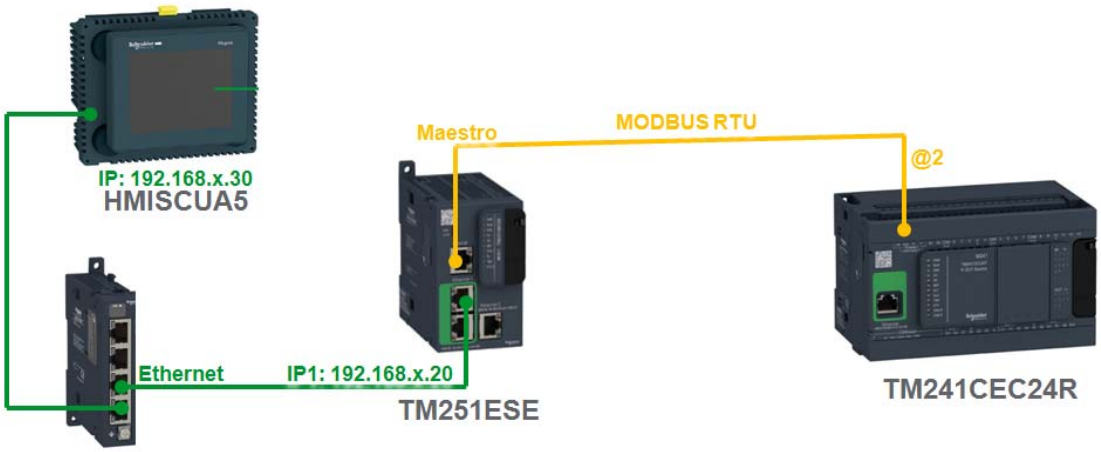


Por último en la programación hacemos que el salto vaya a la etapa 0 **‘Step0’**, escribiéndolo **‘Step0’** en el salto.



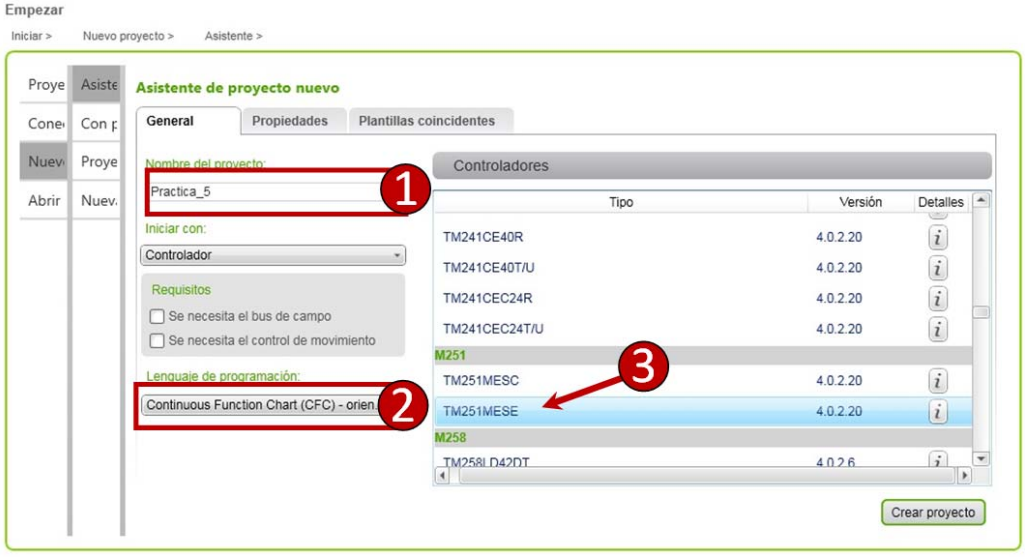
5.5.- PRÁCTICA 5: M251 MENSAJERÍA MODBUS RTU

En este ejercicio realizaremos configuraremos y programaremos la comunicación modbus entre el M251 y el M241.



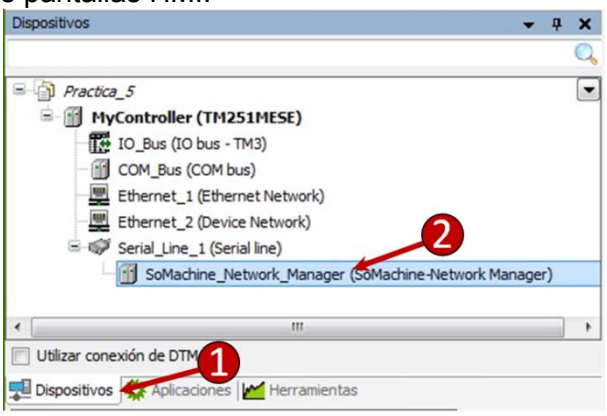
5.5.1.- Programación de la comunicación Modbus RTU.

En la ventana del asistente de creación del proyecto dentro de la pestaña **'General'**, escribiremos el nombre de nuestro proyecto *(Practica_5)*, en **'Iniciar con'** seleccionaremos **'Controlador'** y en la parte de la derecha buscaremos el controlador deseado *(en este caso TM251ESE)*, por último en el '**lenguaje de programación'** seleccionamos el lenguaje con el queremos programar el primer POU *(en este caso Continuous Funtion Chart (CFC) – orientado a página)*. Por último pulsamos '**Crear Proyecto'** para empezar.

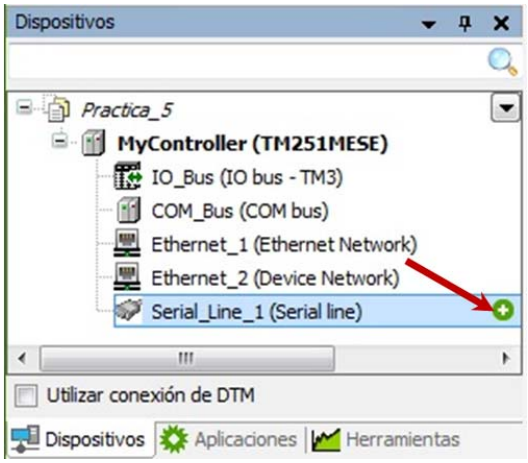


Cuando haya aparecido la ventana principal del SoMachine (*SoMachine Central*), nos aparece el Flujo de trabajo típico de un proyecto automatización, para entrar en la ventana de programación tendremos que pulsar el botón de **'Controlador'**, ya que dentro del flujo la configuración del controlador ya la hemos realizado previamente con el asistente.

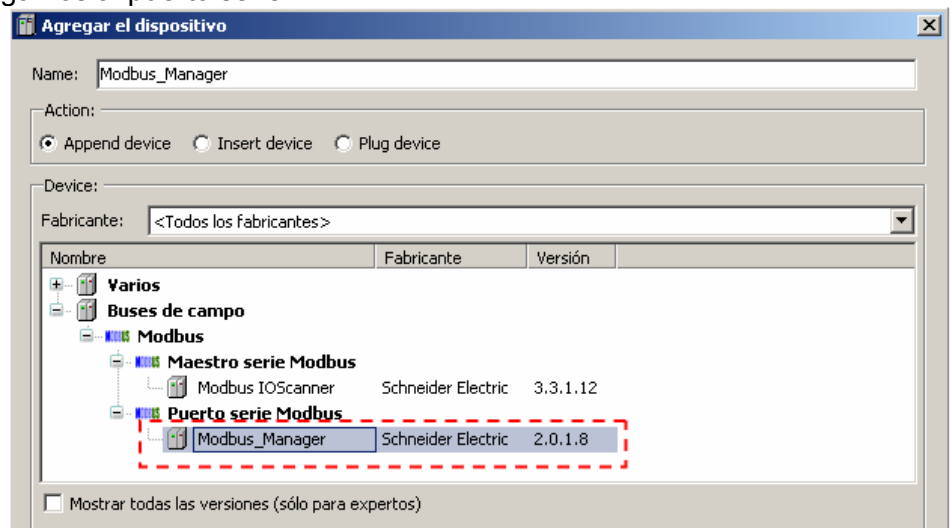
Una vez dentro de la ventana de programación (Logic Builder). En la pestaña **'Dispositivos'** del navegador del proyecto, hay que borrar la configuración presente en la **Línea serie 1** (*SoMachine_Network_Manager*), ya que solo se utiliza para la comunicación con las pantallas HMI.



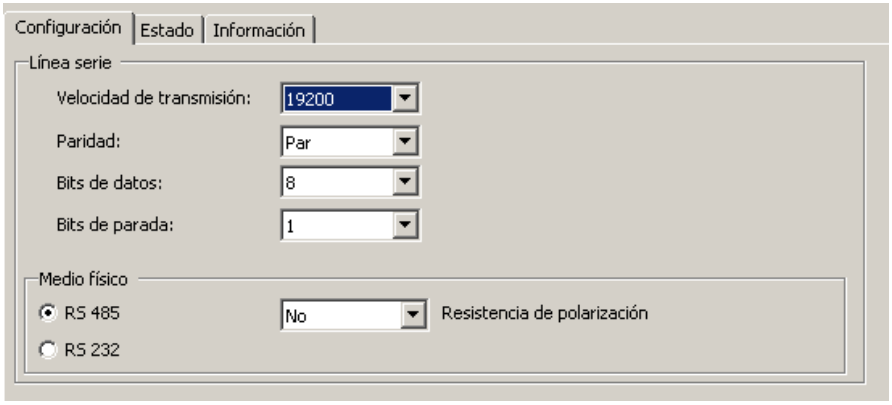
En cuanto esté borrada, seleccionamos el **'Serial_Line_1 (Serial line)'**, pulsamos **'+'**, para añadir un objeto.



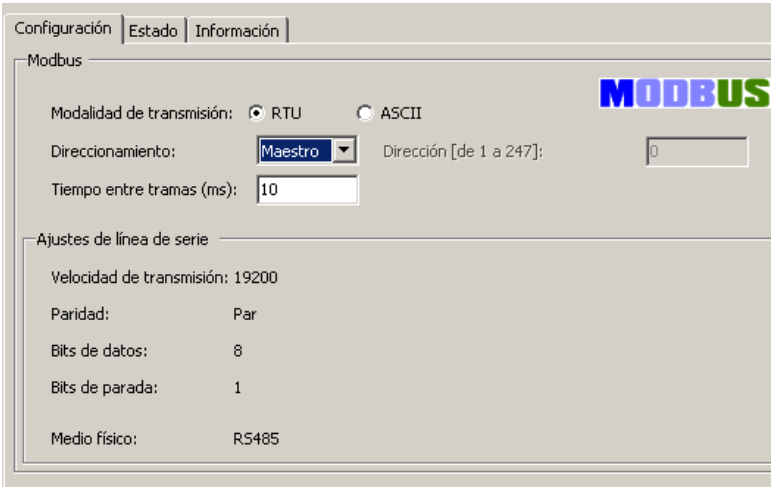
Aparece la ventana para añadir el protocolo de comunicación que vamos a necesitar para comunicar el M251 con el M241. Seleccionamos el objeto **'Modbus_manager'** y lo agregamos al puerto serie 1.



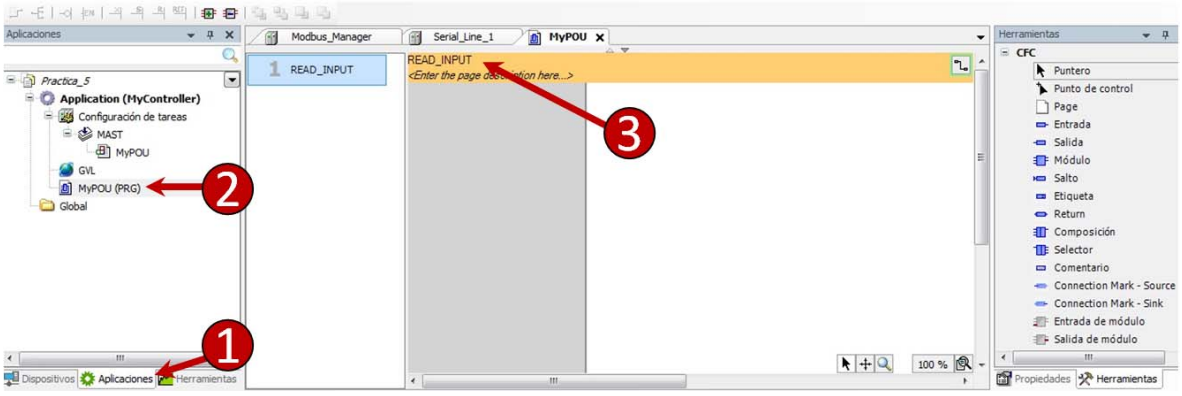
Ahora configuraremos los parámetros de la trama Modbus RTU del puerto serie 1, para la utilización del Modbus por mensajería. Hacemos doble click en la **'Línea Serie 1',** para abrir la ventana de parámetros de comunicación del puerto donde se parametrizar la comunicación Modbus (*19200 kb/s, Par, 8, 1 y RS-485*).



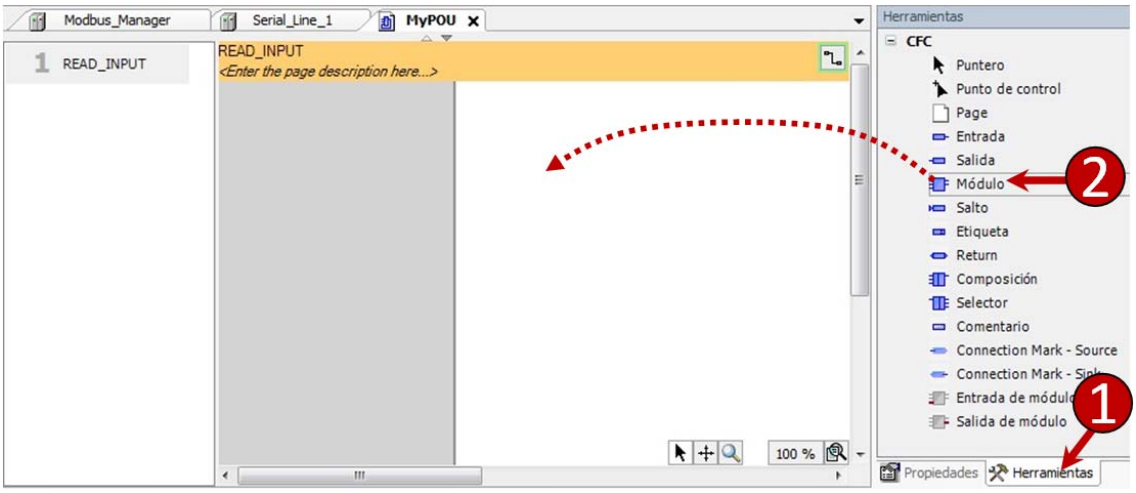
Cuando hemos configuramos la trama, hacemos doble click sobre el **'Modbus Manager'** para abrir la ventana de configuración del gestor. En el campo **'Direccionamiento'** seleccionaremos '**Maestro' ,** para determinar que el M251 se comportará como maestro.



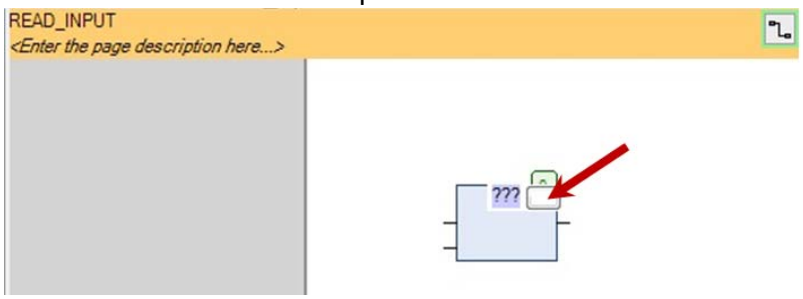
Cuando hemos configurado el puerto serie 1 como maestro modbus ..etc. Nos vamos a la pestaña de **'Aplicaciones'** del navegador del proyecto, y abrimos el POU '**MyPOU'** que se ha creado automáticamente al crear el proyecto con el asistente y que utiliza el lenguaje de programación CFC. Escribimos **'READ_INPUT'** en la cabecera de la página.



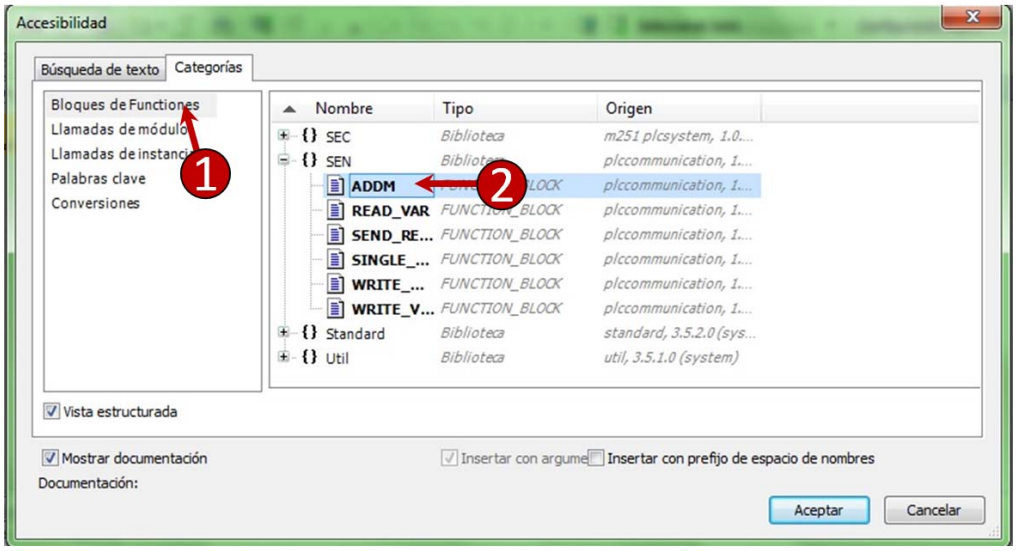
Insertamos un '**módulo**' desde la pestaña de **'herramientas'** de programación, manteniéndolo pulsado y desplazándolo al punto del área de trabajo deseado y soltando.



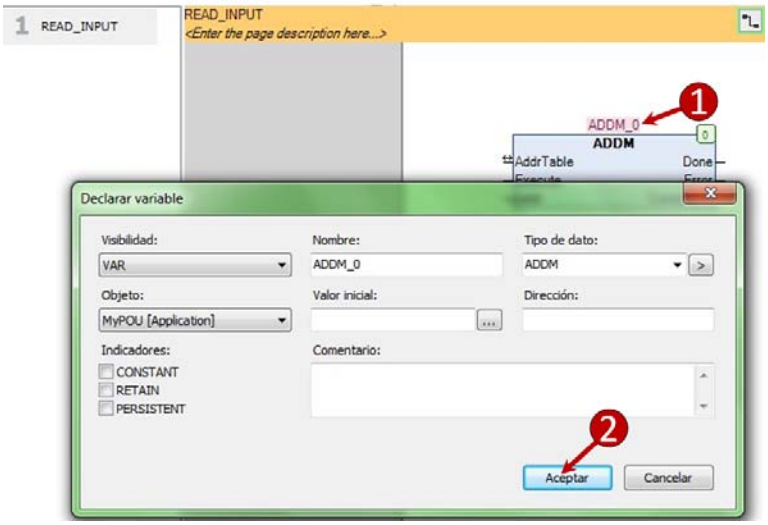
En el módulo pulsamos sobre los interrogantes que hay dentro de la caja para seleccionar el nombre de la instrucción que se desea insertar.



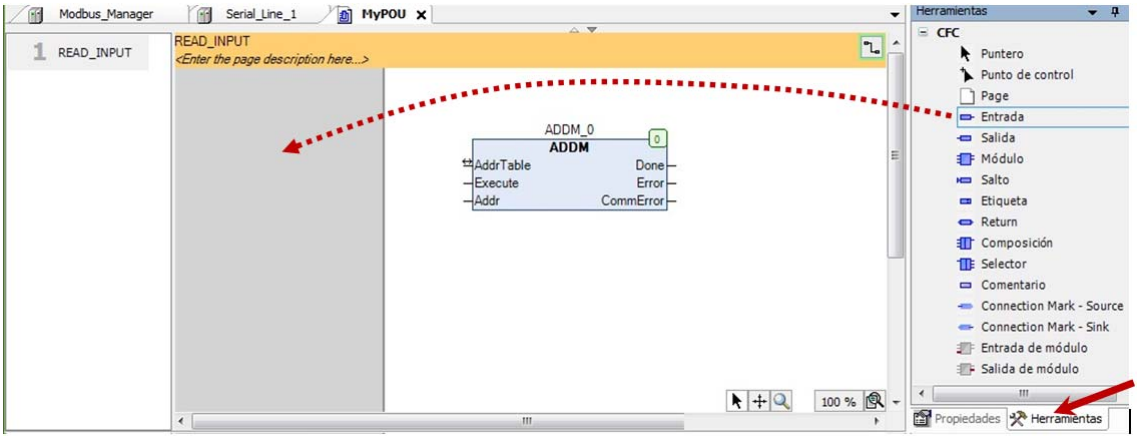
Al pulsar el botón resaltado, nos aparece la ventana de **'Accesibilidad'** con las diferentes instrucciones que hay en las librerías cargadas, en este caso, nos vamos a **'Bloques de Funciones'**, seleccionamos la librería '**plc communication'** ó '**SEN'** (*Shcneider Electric Network*) y elegimos la instrucción **'ADDM'**.



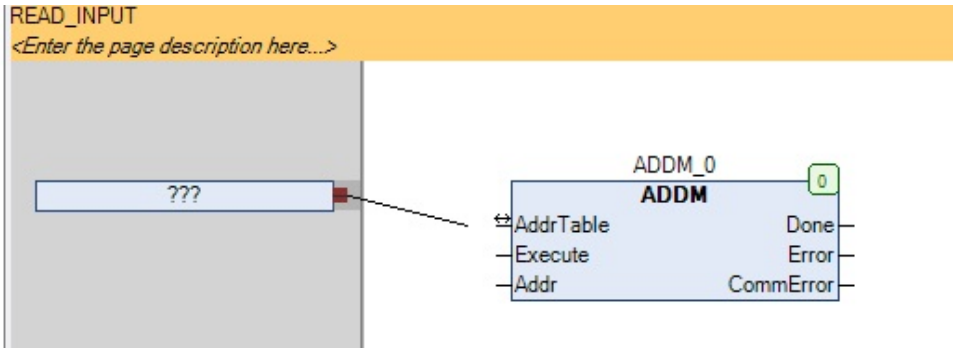
Aceptamos el nombre que se genera automáticamente (autodeclaración) y que es la llamada a la instancia que cualquier bloque de funciones necesita.



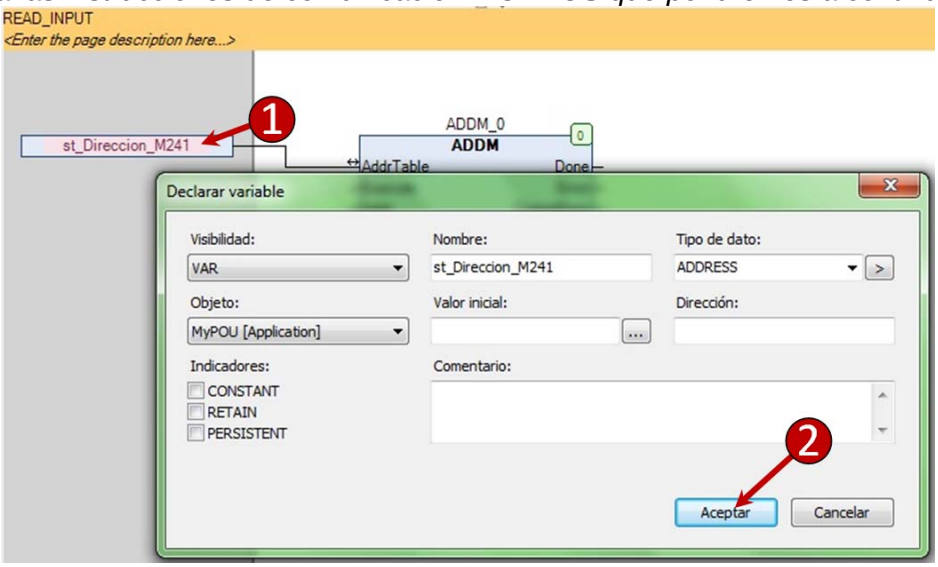
A continuación, insertamos una '**Entrada**' desde la pestaña de **'herramientas'** de programación, manteniéndolo pulsado y desplazándolo al punto del área de trabajo grisácea de la parte izquierda del área de programación, donde se colocan las entradas.



Una vez colocado la entrada pulsamos el pin y manteniéndolo pulsado lo conectamos con el primer pin de la instrucción **'ADDM'** llamado **'AddrTable'**.



Ahora que ya está conectado, crearemos una nueva variable para ello escribiremos **'st_direccion_M241'**, al reconocer el SoMachine que la variable es nueva, aparecerá la ventana de creación de variable. (*Esta variable del tipo 'ADDRESS', convertirá la constante string del pin 'Addr' de la instrucción 'ADDM' en una estructura de datos que entienda las instrucciones de comunicación MODBUS que pondremos a continuación*).



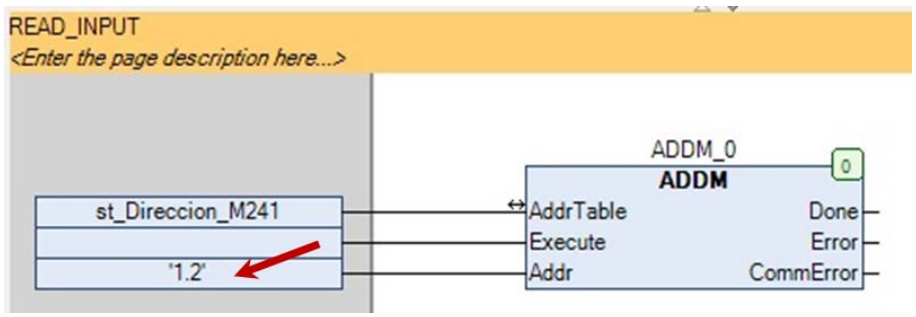
En el pin **'Addr'** de la instrucción **'ADDM'**, seleccionaremos el pin y escribiremos **'1.2'** que es una constante **'STRING'** (*por eso va entre apóstrofes*). Dependiendo del tipo de comunicación escribiremos:

- **ASCII:** *('<Valor del puerto>')*
  Ejemplo: '2' (La comunicación irá por el puerto serie 2)

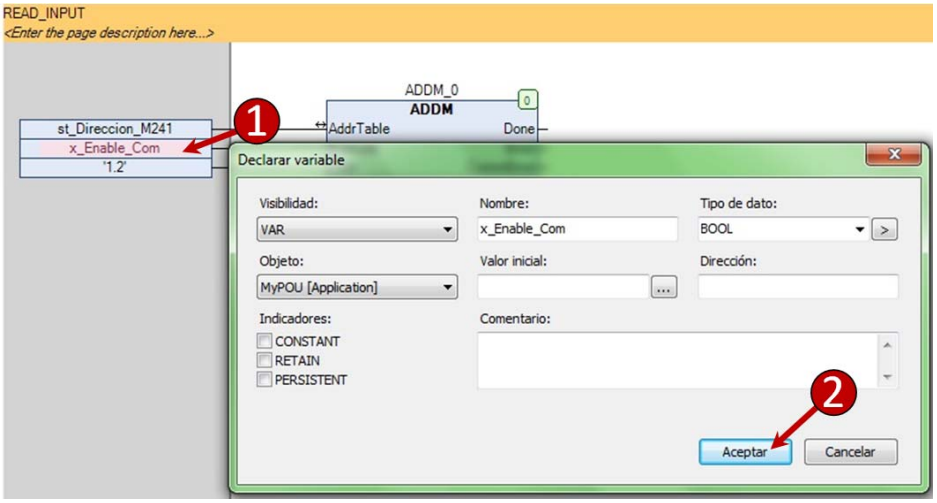
- **MODBUS RTU:** *('<Valor del puerto> . <Número de esclavo MDB RTU>')*
  Ejemplo: '1.2' (La comunicación irá por el puerto serie 1, al esclavo 2)

- **MODBUS TCP/IP:** *('<Valor del puerto> { <Dirección IP del esclavo MDB TCP/IP> }')*
  Ejemplo: '3{192.168.50.10}' (La comunicación irá por el puerto Ethernet y al esclavo con la dirección IP 192.168.50.10)

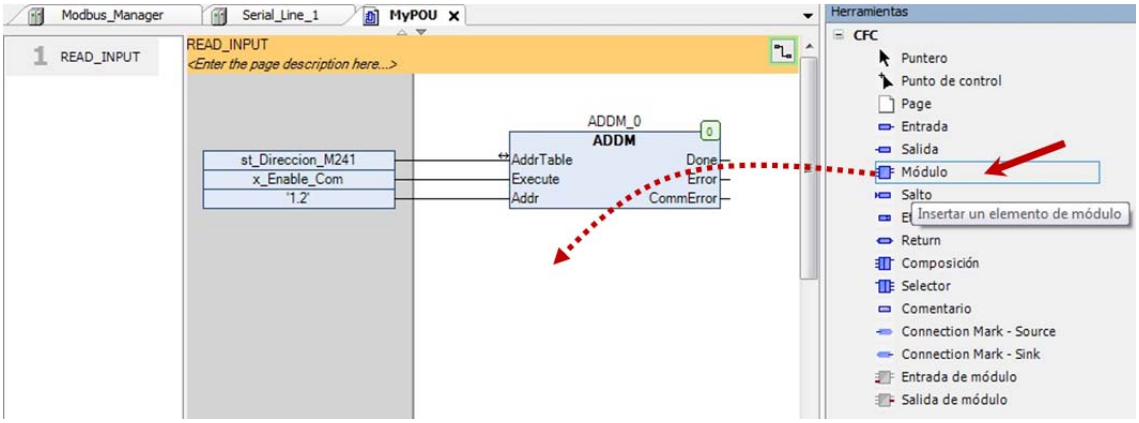
Puerto	Value (hex)	Descripción
Linea Serie 1	01	Puerto serie embebido en el PLC
Linea Serie 2	02	Puerto serie 2 (opcional)
EthEmbed	03	Puerto Ethernet 1 (opciona)



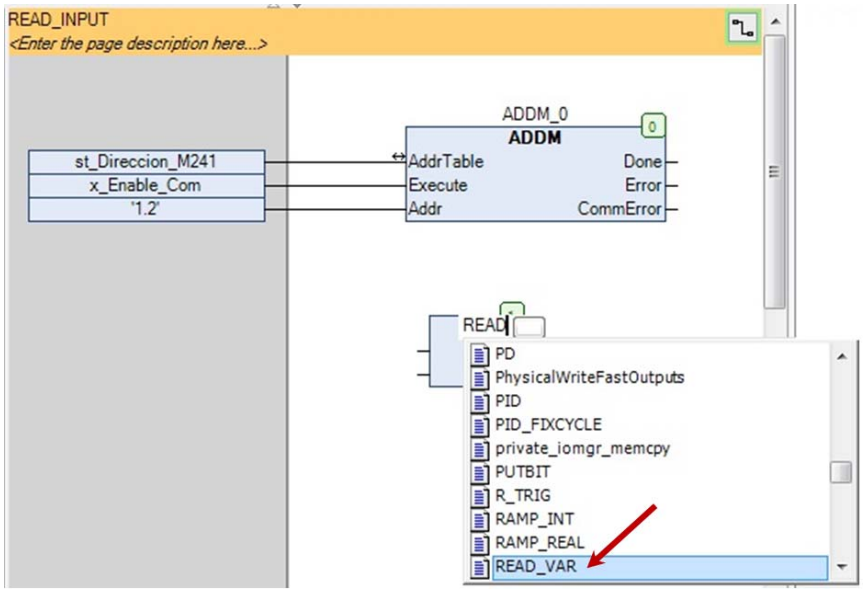
Pulsamos sobre el pin **'Execute'** de la instrucción **'ADDM'**, crearemos una nueva variable para ello escribiremos **'x_Enable_Com'**. (*Esta variable del tipo 'BOOL', Activa la instrucción por flanco*).



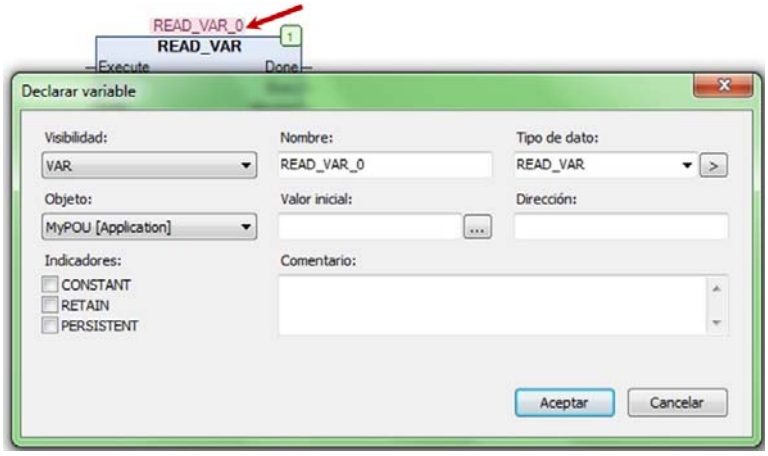
Añadimos un nuevo '**módulo**' desde la pestaña de **'Herramientas'**.



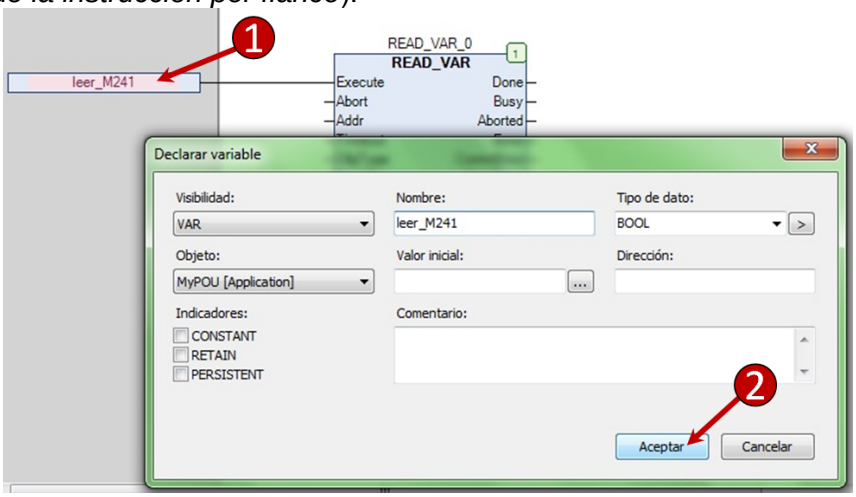
Lo seleccionamos y escribimos la instrucción **'READ_VAR'** instrucción que nos envía una petición de lectura de datos al esclavo.



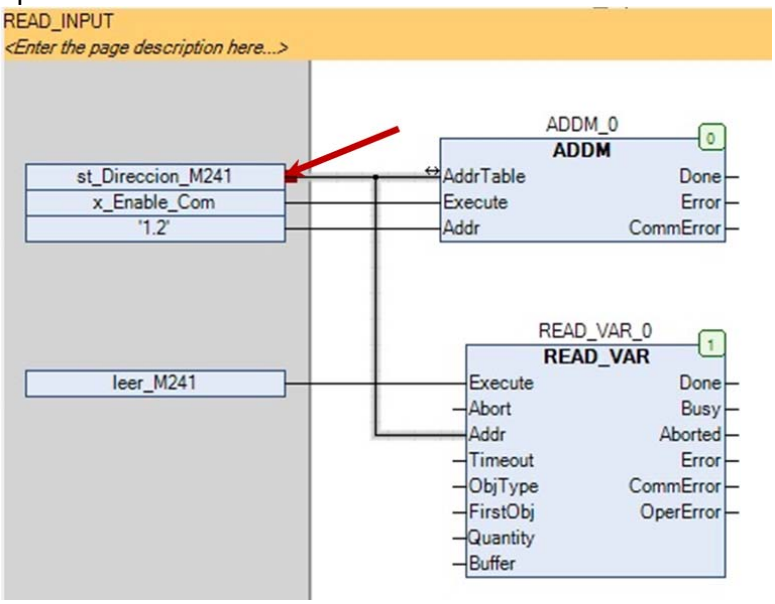
Aceptamos el nombre de la instancia que se genera automáticamente **'READ_VAR_0'**.



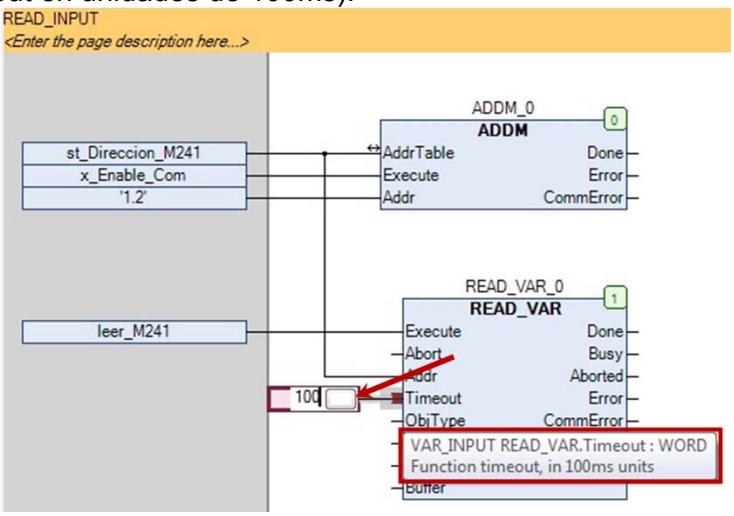
Pulsamos sobre el pin **'Execute'** de la instrucción **'READ_VAR'**, y creamos una nueva variable, para ello escribiremos **'leer_M241'**. (*Esta variable del tipo 'BOOL', activa la ejecución de la instrucción por flanco*).



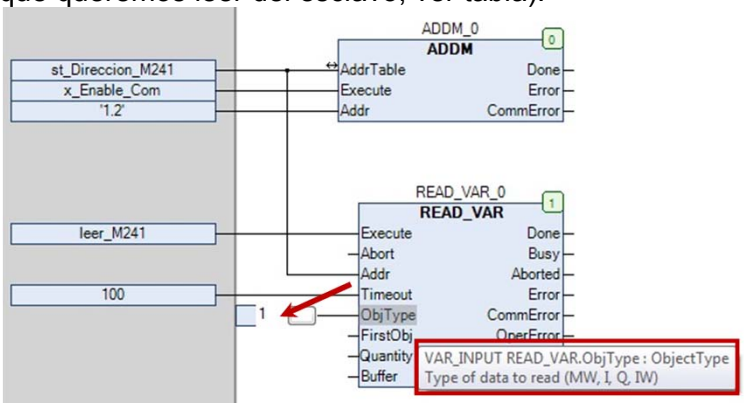
Seleccionamos la entrada '**st_Direccion_M241',** y pulsamos sobre el pin y manteniéndolo pulsado lo conectamos con **'Addr'** de la instrucción **'READ_VAR'**.



En el pin **'Timeout'** de la instrucción **'READ_VAR'**, seleccionaremos el pin y escribiremos **'100'** que es una constante decimal. (En este pin indicamos el tiempo de alarma de timeout en unidades de 100ms).

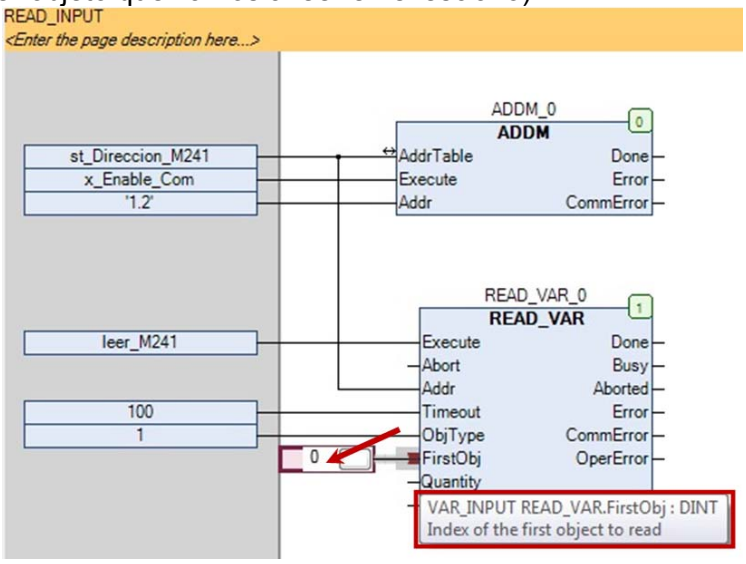


En el pin **'ObjType'** de la instrucción **'READ_VAR'**, seleccionaremos el pin y escribiremos **'1'** que es una constante decimal. (En este pin indicamos el tipo de dato que queremos leer del esclavo, ver tabla).

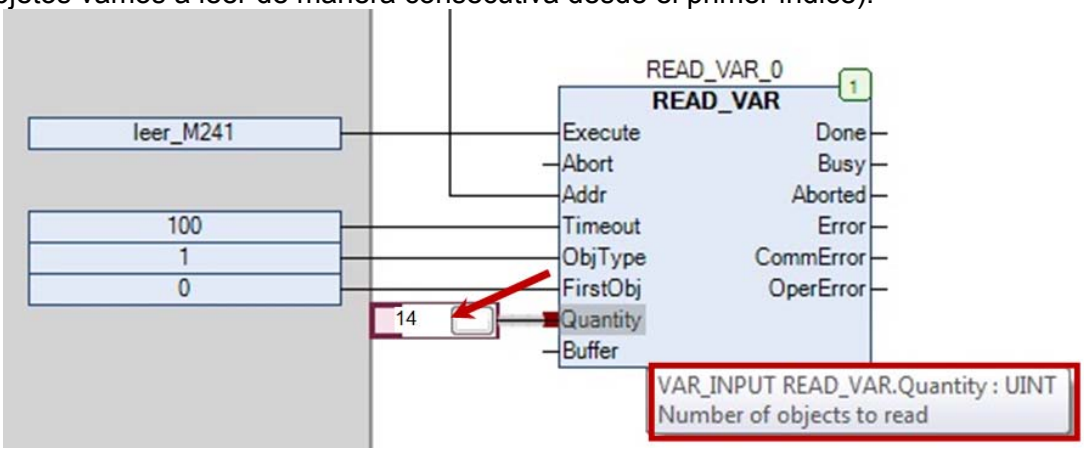


Enumerator	Value (hex)	Object Type)	READ_VAR	WRITE_VAR
MW	00	Holding Register (16 bits)	#3 (Read Holding Registers)	#16 (Write Multiple Registers)
I	01	Digital Input (1 bit)	#2 (Read Digital Inputs)	—
Q	02	Internal bit or Digital output (coil) (1 bit)	#1 (Read Coils)	#15 (Write Multiple Coils)
IW	03	Input register (16 bits)	#4 (Read Input Registers)	—

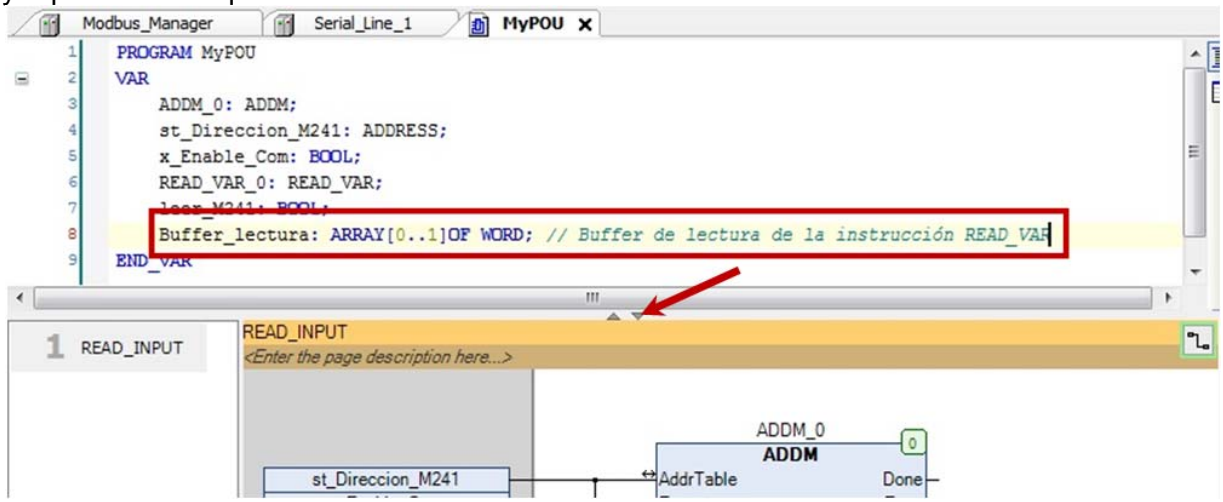
En el pin **'FirstObj'** de la instrucción **'READ_VAR'**, seleccionaremos el pin y escribiremos **'0'** que es una constante decimal. (En este pin indicamos cual será el índice del primer objeto que vamos a leer en el esclavo).



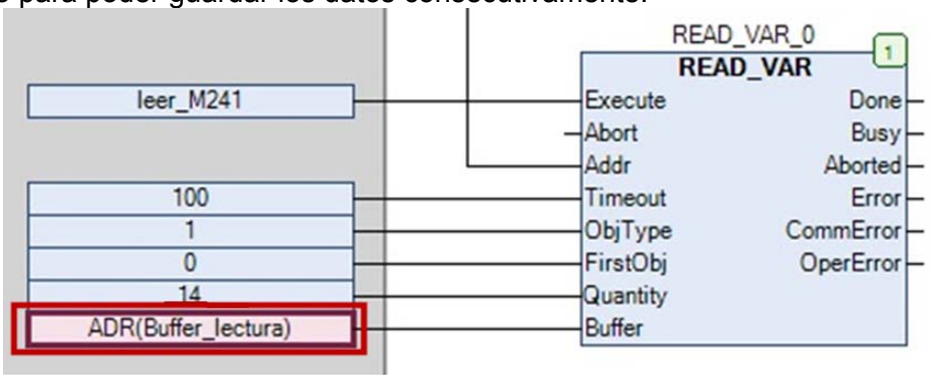
En el pin **‘Quantity’** de la instrucción **‘READ_VAR’**, seleccionaremos el pin y escribiremos **‘14’** que es una constante decimal. (En este pin indicamos cuantos objetos vamos a leer de manera consecutiva desde el primer índice).



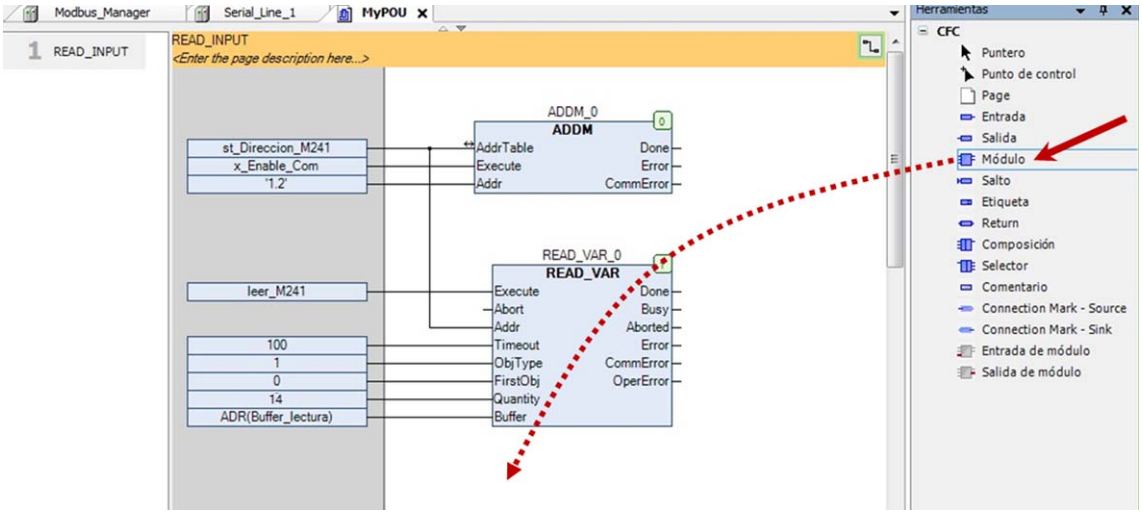
Para rellenar el último pin **‘Buffer’** de la instrucción **‘READ_VAR’**, tenemos que desplegar la ventana de declaración de variables del POU, pulsando el triangulo, y en la ventana crear una variable del tipo array (el array tendrá que tener un tamaño igual o mayor que el número de objetos que deseamos leer), en este caso creamos una variable llamada **‘Buffer_lectura’**, que en este caso es un **‘ARRAY [0..1]OF WORD’**, ya que tenemos que leer 14 bits.



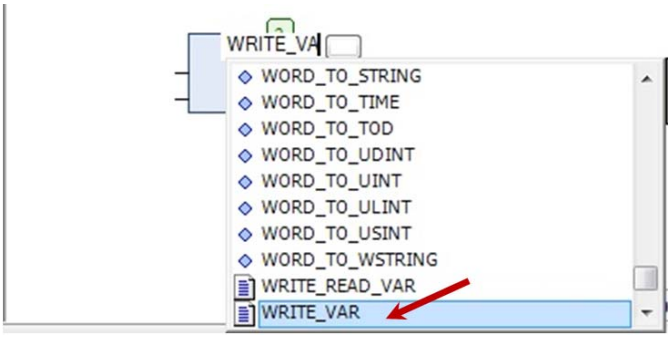
Una vez creado la variable ‘Buffer_lectura’, en el pin **‘Buffer’** de la instrucción **‘READ_VAR’**, seleccionaremos el pin y escribiremos **‘ADR(Buffer_lectura)’** que es un puntero para poder guardar los datos consecutivamente.



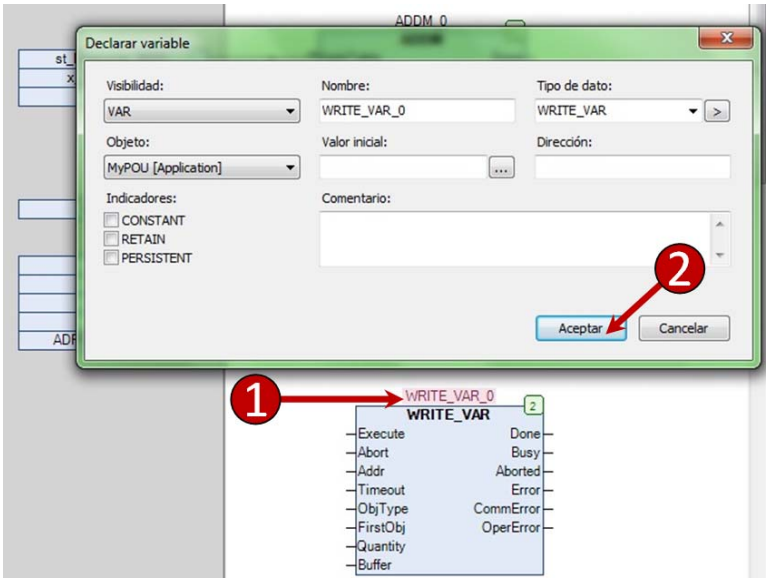
Ahora repetiremos estos últimos pasos para poder programar la instrucción de petición de escritura modbus que será un WRITE_VAR. Añadimos un nuevo '**módulo**' desde la pestaña de **'Herramientas'**.



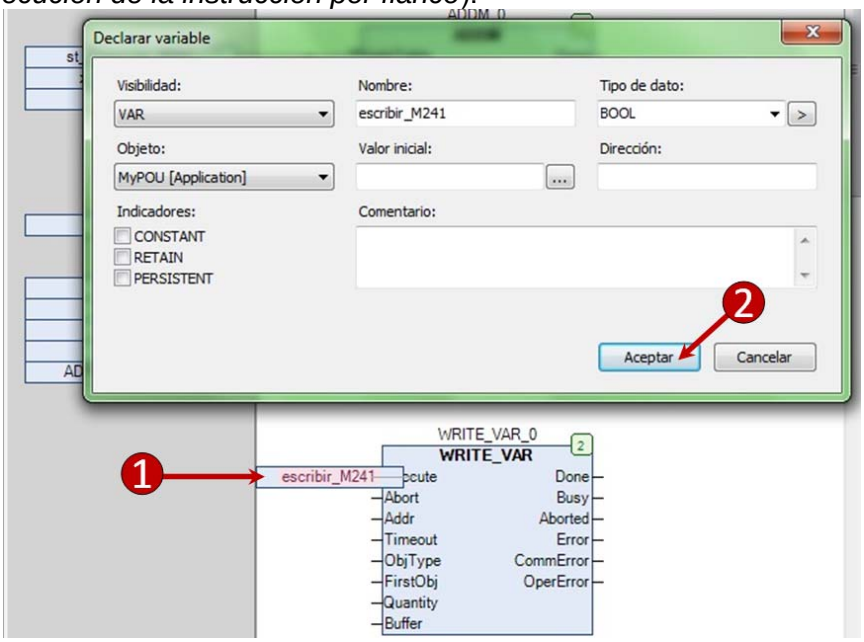
Lo seleccionamos y escribimos la instrucción **'WRITE_VAR'** instrucción que nos envía una petición de escritura de datos al esclavo.



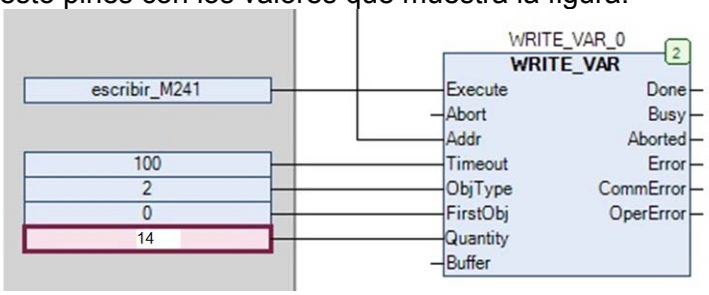
Aceptamos el nombre de la instancia que se genera automáticamente **'WRITE_VAR_0'**.



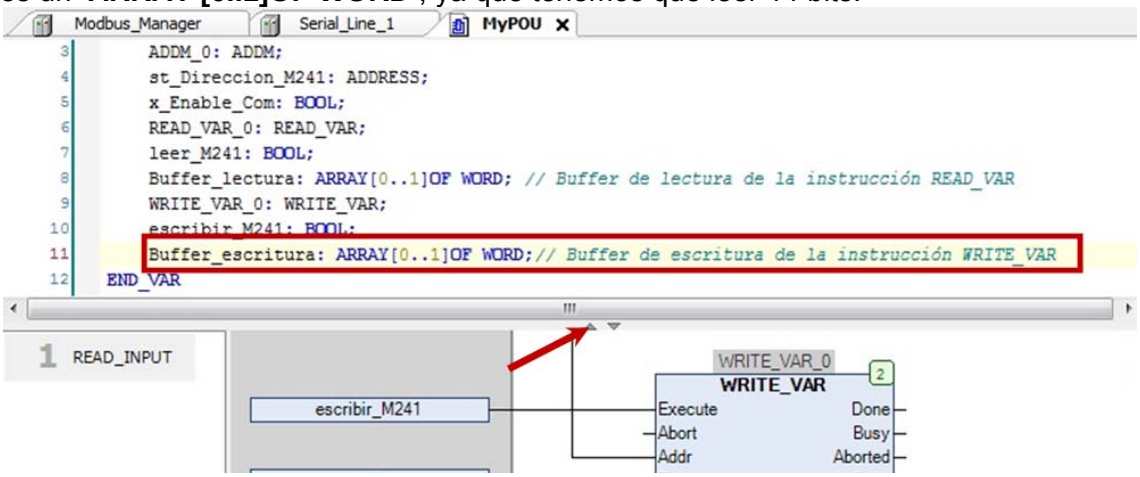
Pulsamos sobre el pin **'Execute'** de la instrucción **'WRITE_VAR'**, y creamos una nueva variable, para ello escribiremos **'escribir_M241'**. (*Esta variable del tipo 'BOOL', activa la ejecución de la instrucción por flanco*).



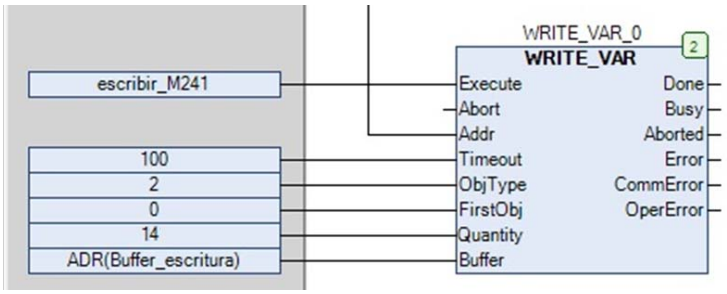
Seleccionamos la entrada '**st_Direccion_M241',** y pulsamos sobre el pin y manteniéndolo pulsado lo conectamos con **'Addr'** de la instrucción **'WRITE_VAR'**. Y rellenamos el resto pines con los valores que muestra la figura.



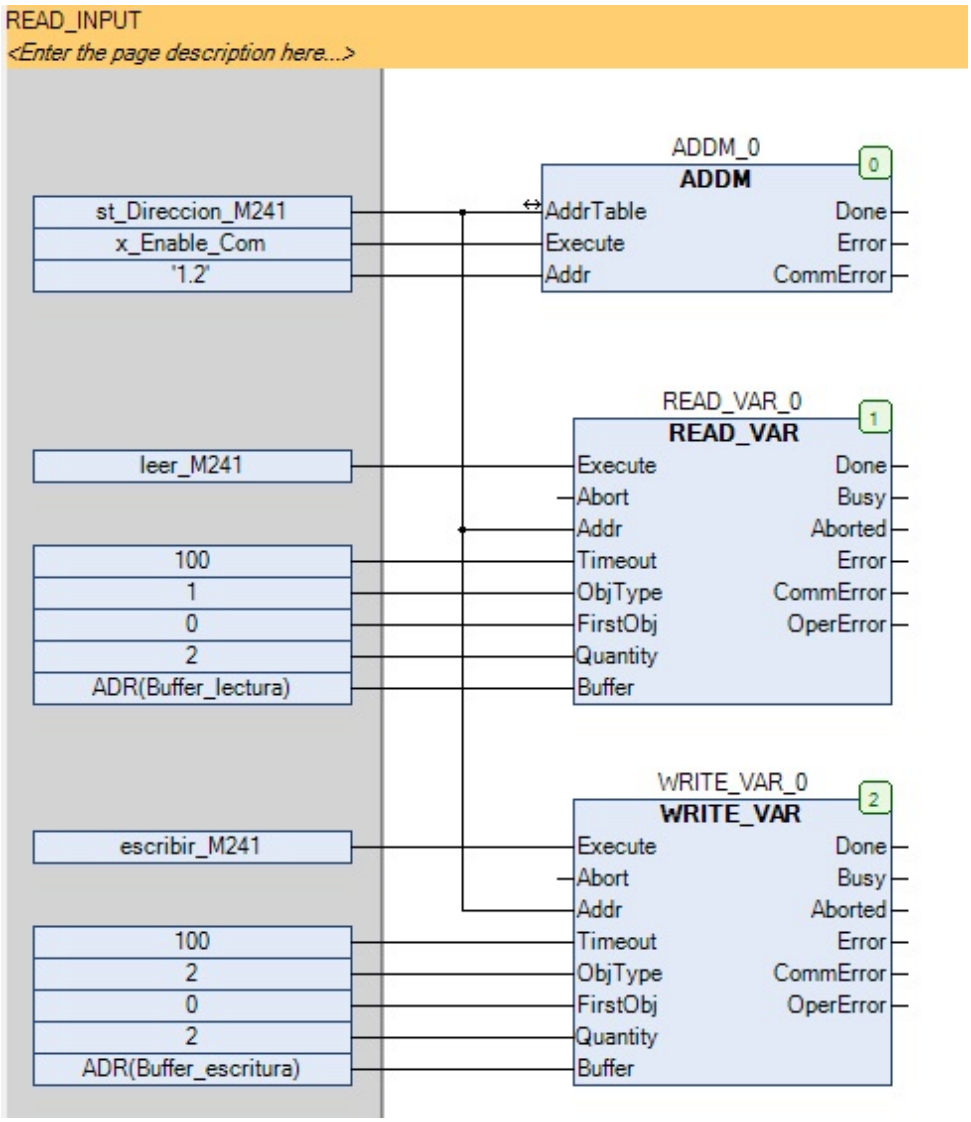
Para rellenar el último pin **'Buffer'** de la instrucción **'WRITE_VAR'**, igual que en la instrucción anterior, tenemos que desplegar la ventana de declaración de variables del POU, pulsando el triangulo, y en la ventana crear una variable del tipo array (el array tendrá que tener un tamaño igual o mayor que el número de objetos que deseamos leer), en este caso creamos una variable llamada **'Buffer_escritura'**, que en este caso es un **'ARRAY [0..1]OF WORD'**, ya que tenemos que leer 14 bits.



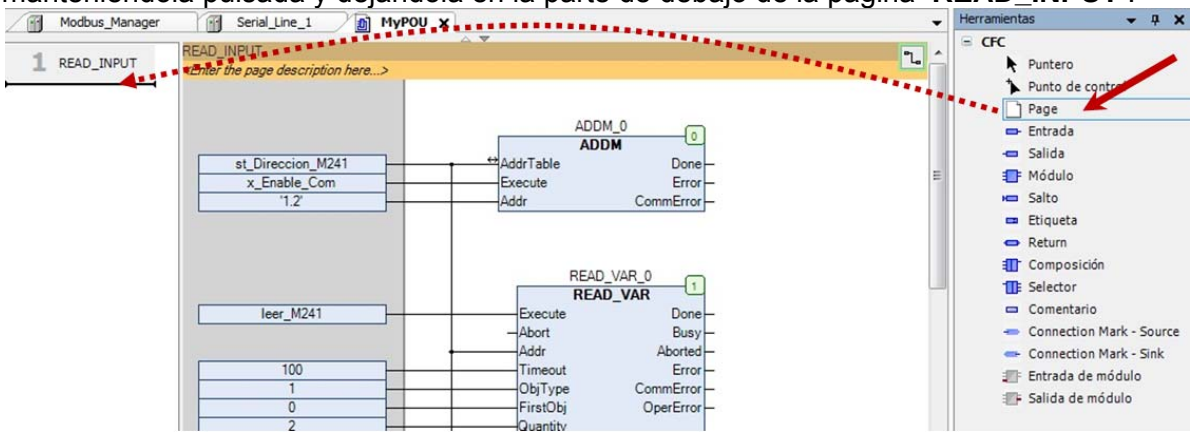
Por último, en el pin **'Buffer'** de la instrucción **'WRITE_VAR'**, seleccionaremos el pin y escribiremos **'ADR(Buffer_escritura)'** que es un puntero para poder escribir los datos que queremos enviar al esclavo consecutivamente.



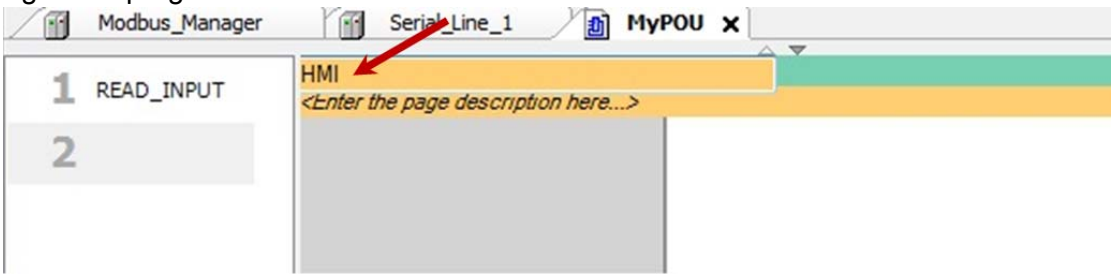
Al final el programa de envío y recepción de datos al esclavo M241 queda de la siguiente manera.



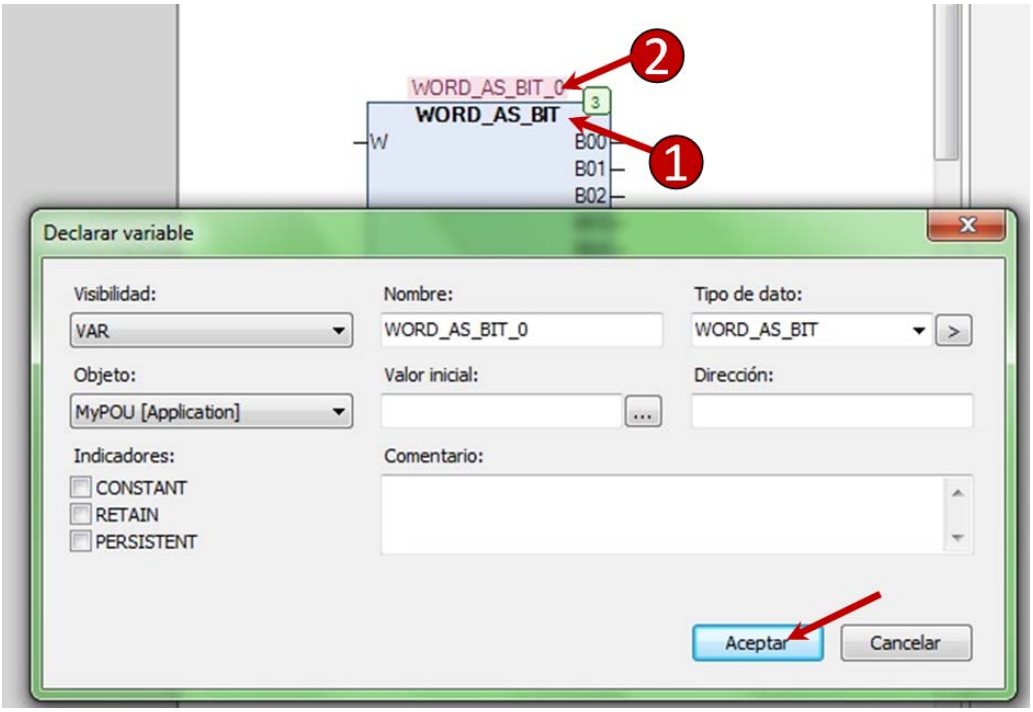
Añadimos una página, seleccionando **'Page'** en la pestaña **'Herramientas'** manteniéndola pulsada y dejándola en la parte de debajo de la página **'READ_INPUT'**.



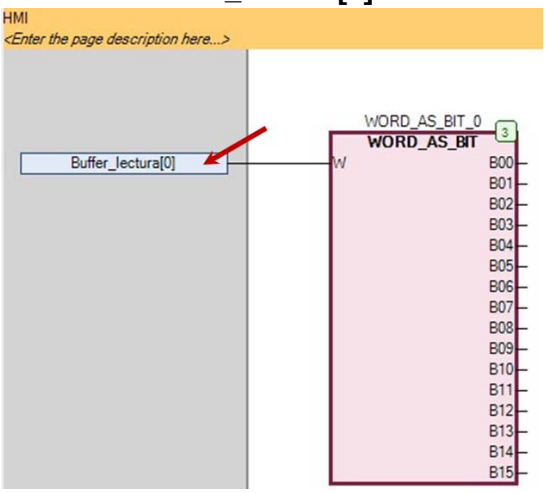
En el comentario de cabecera escribimos **'HMI'**, que será el nombre que tendrá esta página de programación.



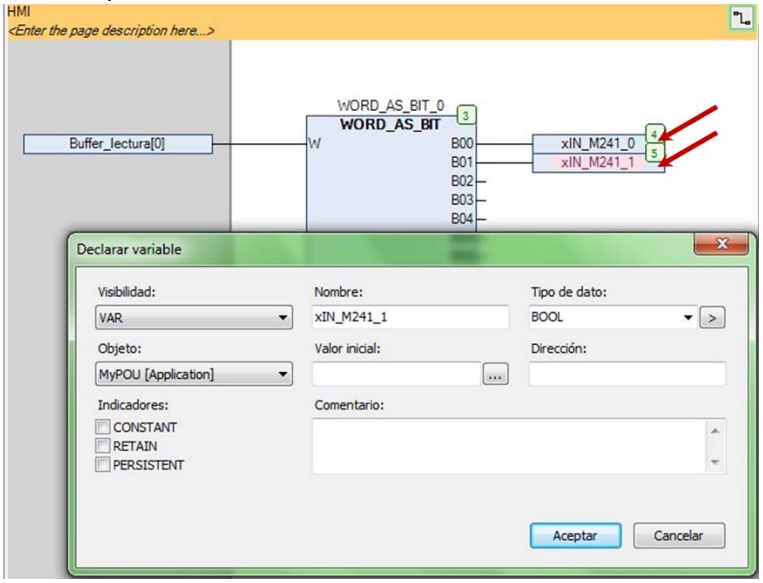
Añadimos un módulo a esta página, acto seguido seleccionamos la instrucción **'WORD_AS_BIT'**, que descompone un word en 16 bits de salida, y creamos la llamada a la instancia.



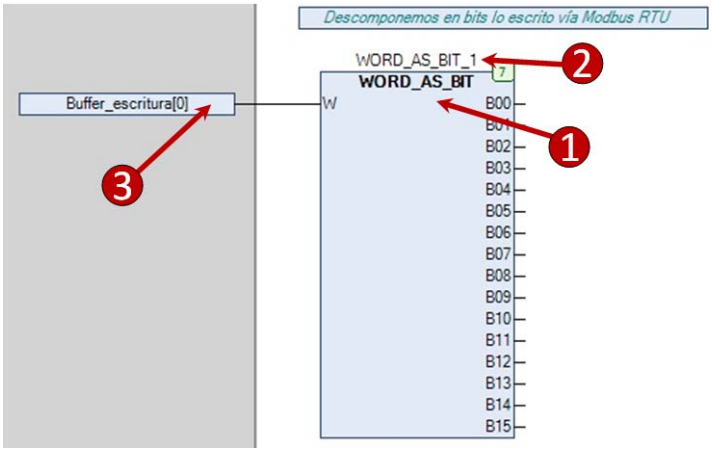
En el pin **'W'** de la instrucción **'WORD_AS_BIT'** le asignamos el primer elemento de la matriz de 'Buffer_lectura', recordar que 'buffer_lectura' era una matriz desde 0 hasta 1, o sea que el primer elemento es **'Buffer_lectura[0]'**.



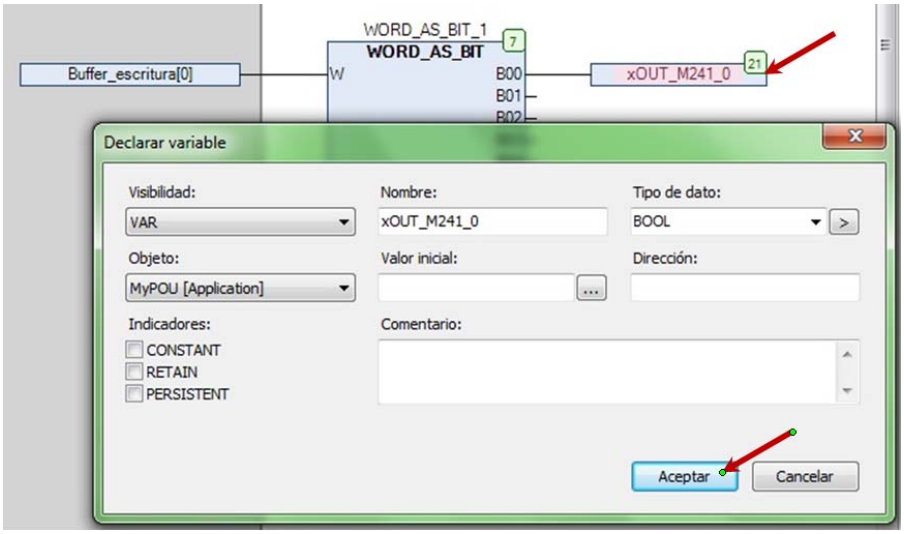
Ahora creamos 16 variables del tipo 'BOOL' con la nomenclatura **'xIN_M241_0..15'**, que son las variables que utilizaremos en el terminal táctil HMI.



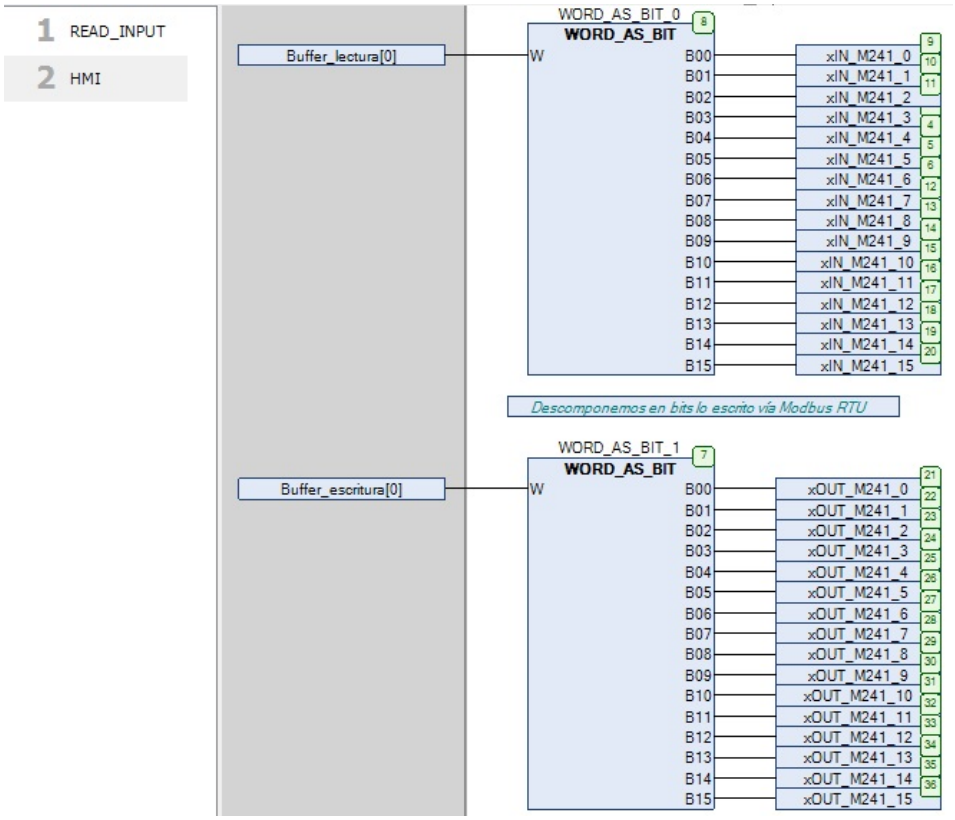
Añadimos otro módulo **'WORD_AS_BIT'**, pero en el pin **'W'** de la instrucción le asignamos ahora el primer elemento de la matriz de 'Buffer_escritura', recordar que 'buffer_escritura' era una matriz desde 0 hasta 1, o sea que el primer elemento es **'Buffer_escritura[0]'**.



En este caso volvemos a crear 16 variables del tipo 'BOOL' con la nomenclatura **'xOUT_M241_0..15'**, que son las variables que utilizaremos en el terminal táctil HMI.



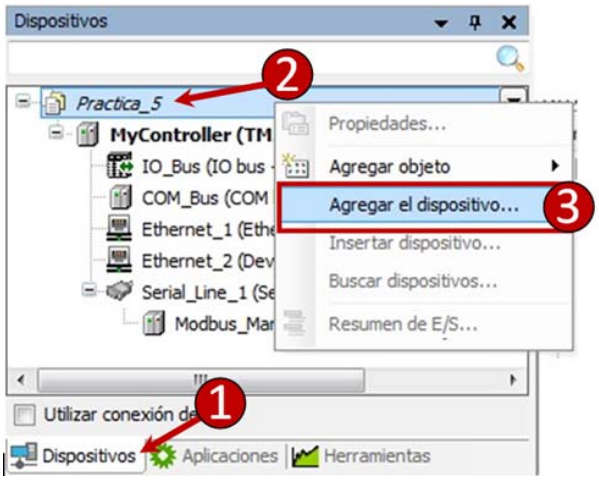
La página quedará programada de la siguiente manera:



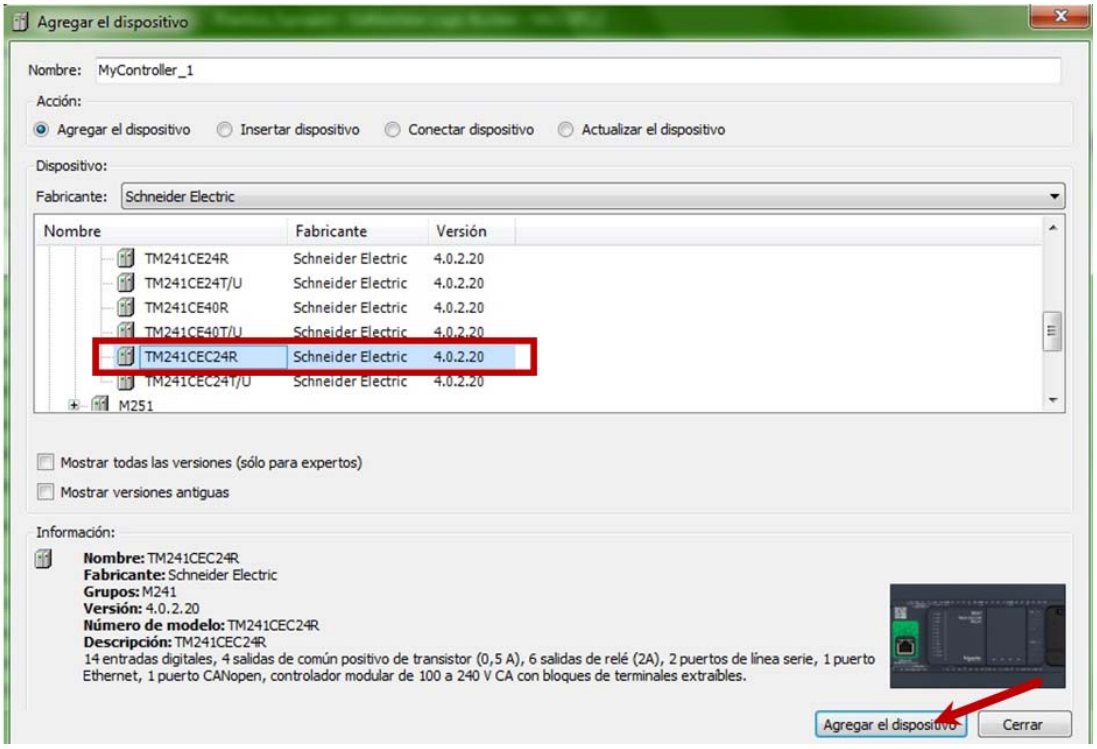
5.5.2.- Programación del esclavo Modbus RTU M241.

Como vemos en la arquitectura hemos utilizado el M241 de la maqueta como esclavo Modbus RTU con el número de esclavo número **2**.

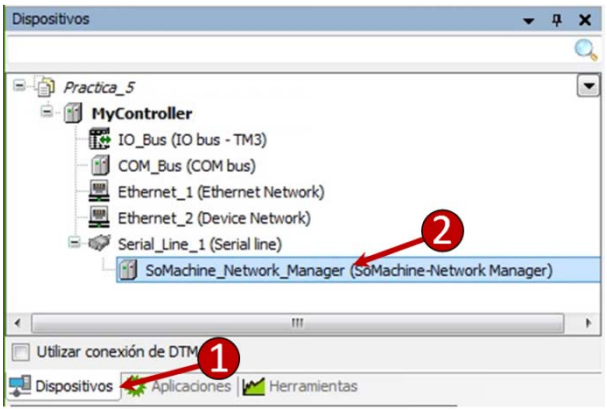
1. Ahora lo primero será añadir al proyecto el controlador M241. Se puede realizar de dos maneras, la primera es volver a la ventana principal del *SoMachine Central,* dentro de la pestaña **'Flujo de Trabajo'**, pulsamos **'Configuración'**. La segunda es en el propio *Logic Builder* ir a la pestaña de **'Dispositivos'** del navegador del proyecto, seleccionar la carpeta del proyecto y con el botón derecho seleccionar la opción **'Añadir dispositivo'**.



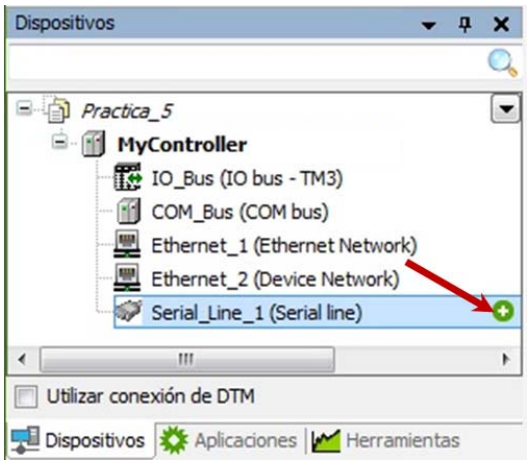
2. En la ventana flotante de **'agregar dispositivo'**, seleccionamos el fabricante **'Schneider Electric',** desplegamos **'Controlador Lógico'**, buscamos la Serie del modelo de controlador 'M241' que deseamos añadir al proyecto, lo desplegamos y seleccionamos el modelo, una vez seleccionado pulsamos el botón de **'agregar dispositivo'** (en *este caso la serie es M241 y el modelo TM241CEC24R*).



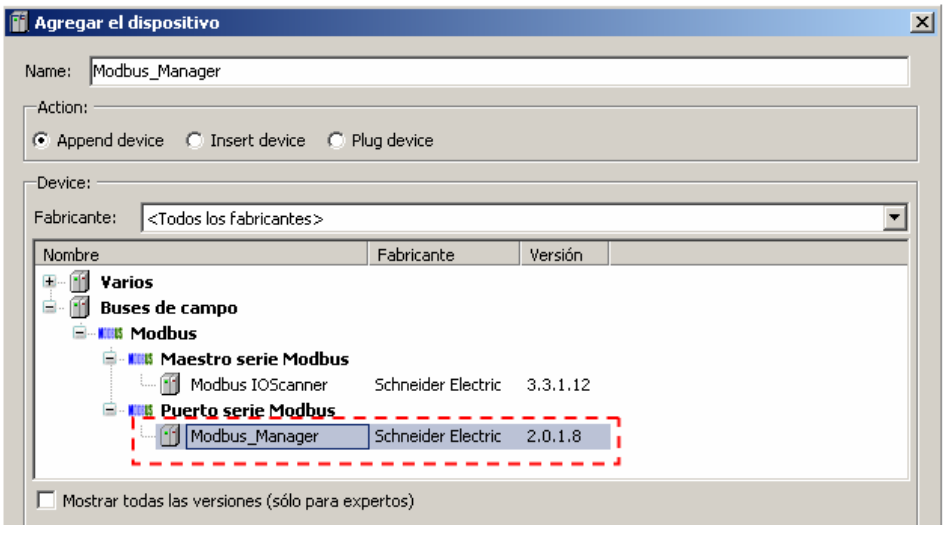
3. Una vez dentro de la ventana de programación (*Logic Builder*). En la pestaña **'Dispositivos'** del navegador del proyecto, hay que borrar la configuración presente en la Línea serie 1 (*SoMachine_Network_Manager*), ya que solo se utiliza para la comunicación con las pantallas HMI.



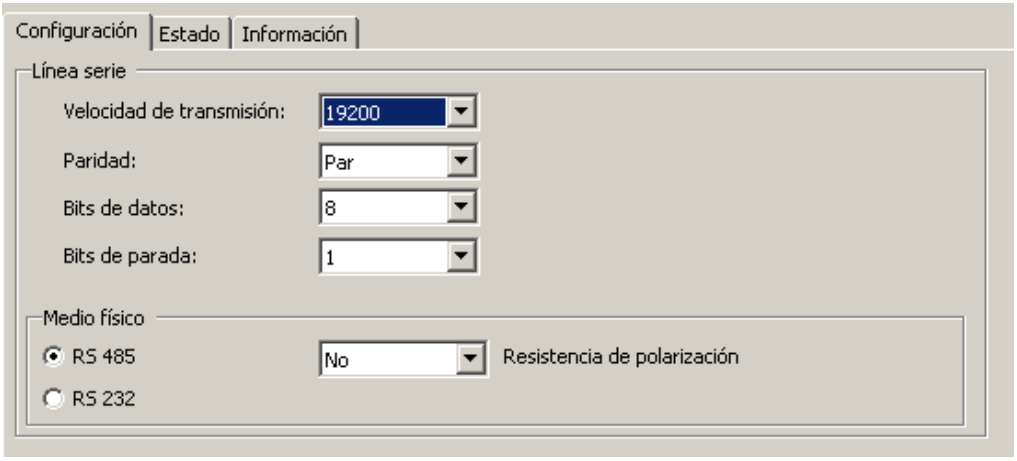
4. En cuanto esté borrada, seleccionamos el **'Serial_Line_1 (Serial line)'**, pulsamos **'+'**, para añadir un objeto.



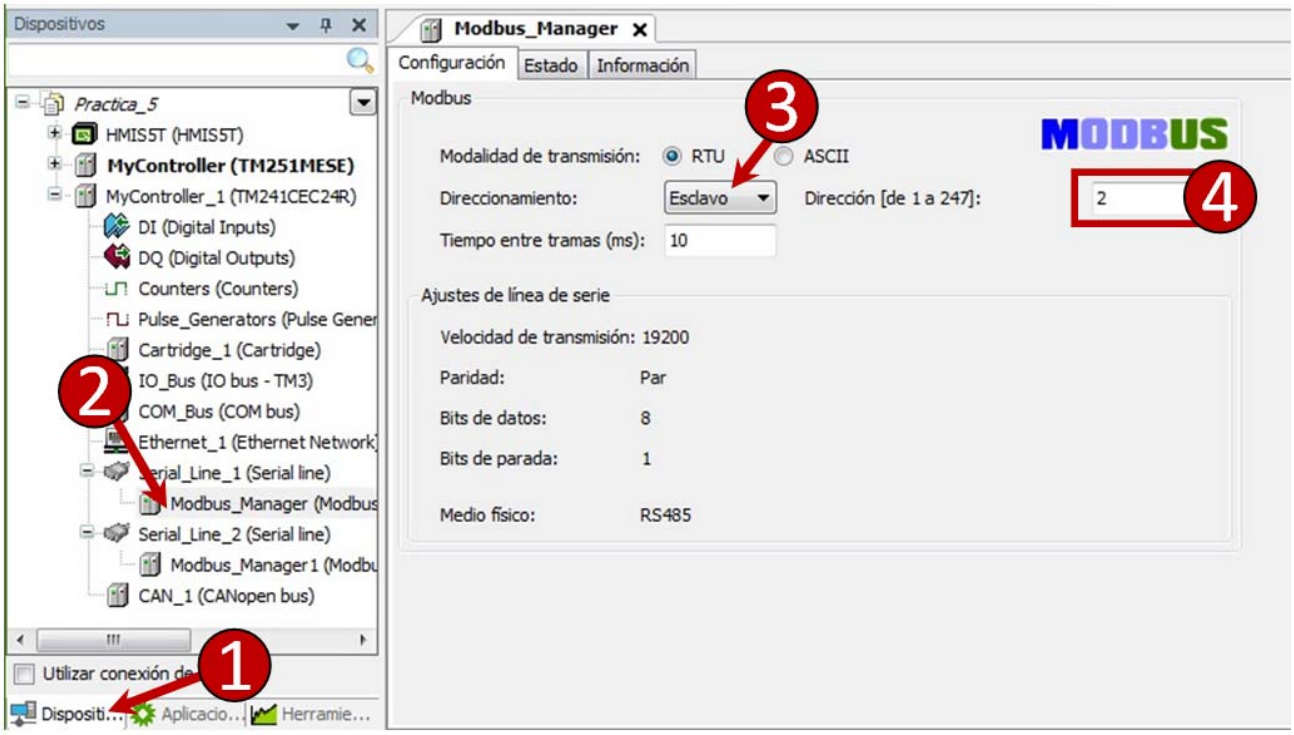
5. Aparece la ventana para añadir el protocolo de comunicación que vamos a necesitar para comunicar el M251 con el M241. Seleccionamos el objeto **'Modbus_manager'** y lo agregamos al puerto serie 1.



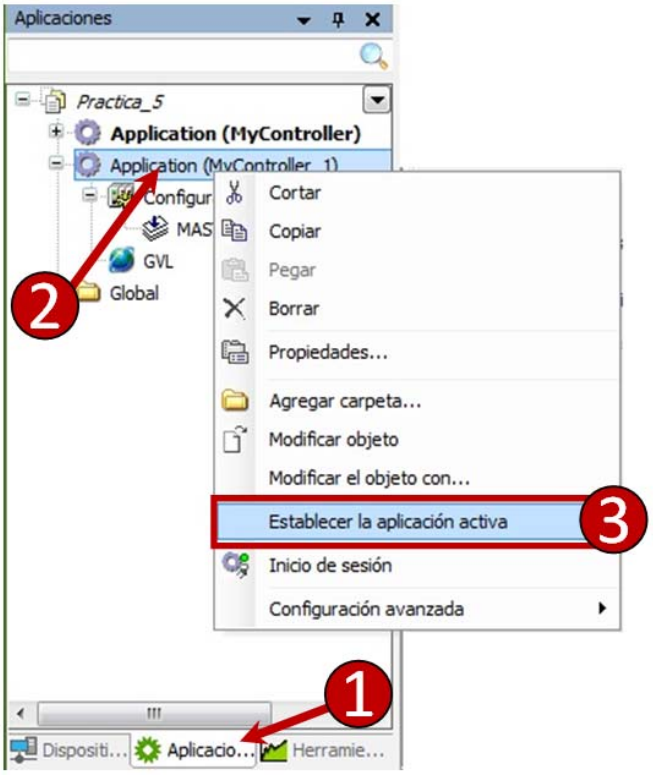
6. Ahora configuraremos los parámetros de la trama Modbus RTU del puerto serie 1, para la utilización del Modbus por mensajería. Hacemos doble click en la **'Línea Serie 1'**, para abrir la ventana de parámetros de comunicación del puerto donde se parametrizar la comunicación Modbus (**19200 kb/s, Par, 8, 1 y RS-485**).



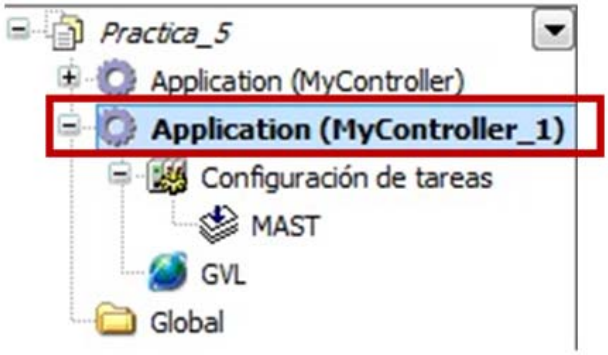
7. Cuando hemos configuramos la trama, hacemos doble click sobre el **'Modbus Manager'**, para abrir la ventana de configuración del gestor. En el campo **'Direccionamiento'** seleccionaremos **'Esclavo'**, para determinar que el M241 se comportará como esclavo y en el campo **'Dirección'** especificamos el número de esclavo en este caso el **'2'**.



8. Por último compilamos en la pestaña de **'Aplicaciones'** seleccionamos **'Aplication'** botón derecho y elegimos la opción **'Establecer como ruta activa'**, con el fin de que a la hora de cargar el proyecto en el M241, sea esta aplicación y no la del M251 la que se cargue. Esto se puede aprecia porque la aplicación queda resaltada en negrita.



9. Ahora ya podemos cargar la aplicación al M241, ya que la aplicación que cargaremos en el controlador seleccionado será la que está en negrita.



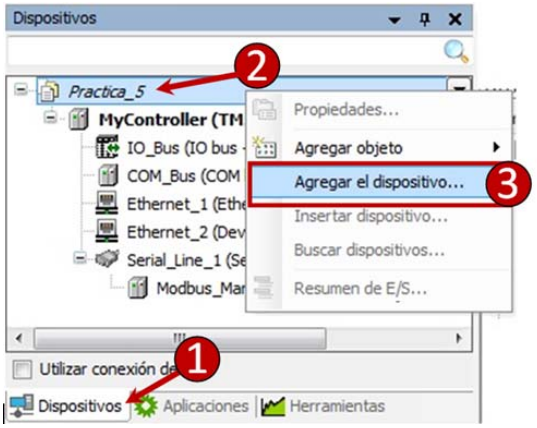
5.5.3.- Añadir una HMI a la arquitectura

En la mayoría de las arquitecturas de una máquina, actualmente disponen de una HMI para la supervisión, monitorización y diagnostico de la misma por parte del operario.

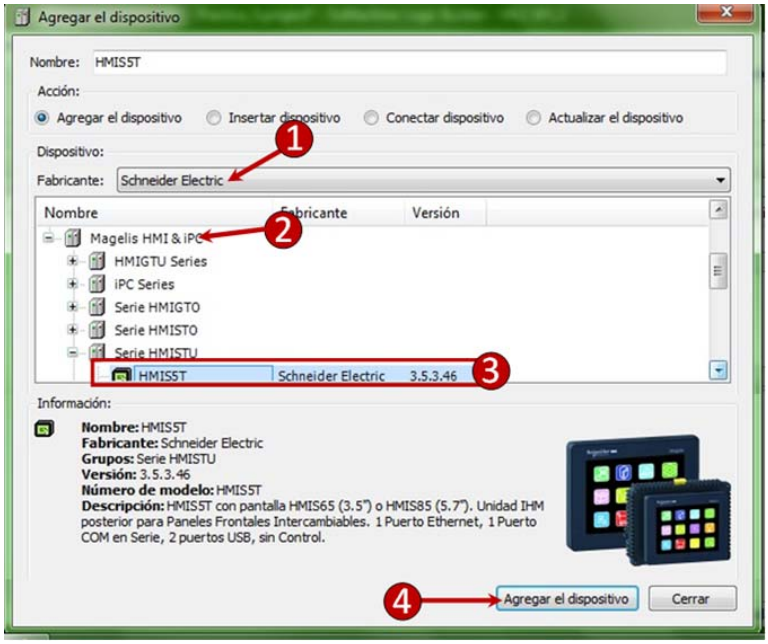
En el SoMachine estos equipos y el software necesario para crear la aplicación de supervisión de la pantalla están enlazados con los del proyecto del SoMachine, el Vijeo Designer está asociado al SoMachine, así que cuando añadimos un terminal HMI, este se abrirá automáticamente.

Para añadir una HMI a la configuración de la arquitectura, debemos realizar los siguientes pasos:

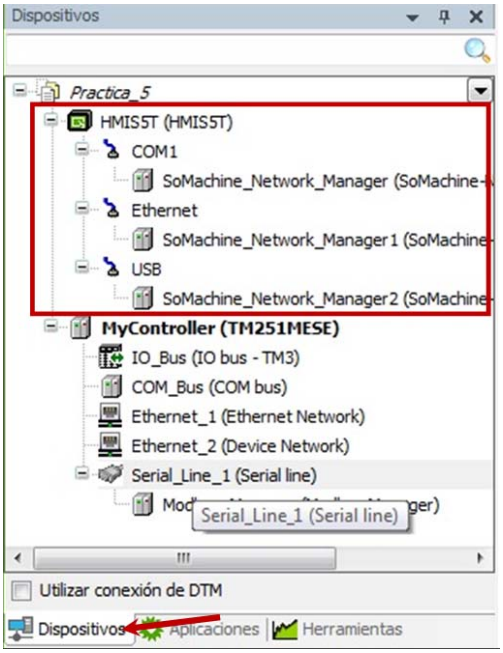
1. Se puede realizar de dos maneras, la primera es volver a la ventana principal del *SoMachine Central,* dentro de la pestaña **'Flujo de Trabajo'**, pulsamos **'Configuración'**. La segunda es en el propio *Logic Builder* ir a la pestaña de **'Dispositivos'** del navegador del proyecto, seleccionar la carpeta del proyecto y con el botón derecho seleccionar la opción **'Añadir dispositivo'**.



2. En la ventana flotante de **'agregar dispositivo'**, seleccionamos el fabricante **'Schneider Electric',** desplegamos **'Magelis HMI & IPC'**, buscamos la Serie del modelo HMI que deseamos añadir al proyecto, desplegamos la Serie y seleccionamos el modelo, una vez seleccionado pulsamos el botón de **'agregar dispositivo'** (en *este caso la serie es HMISTU y el modelo HMIS5T*).



3. Ahora nos aparece en el proyecto de SoMachine el terminal táctil HMIS5T.

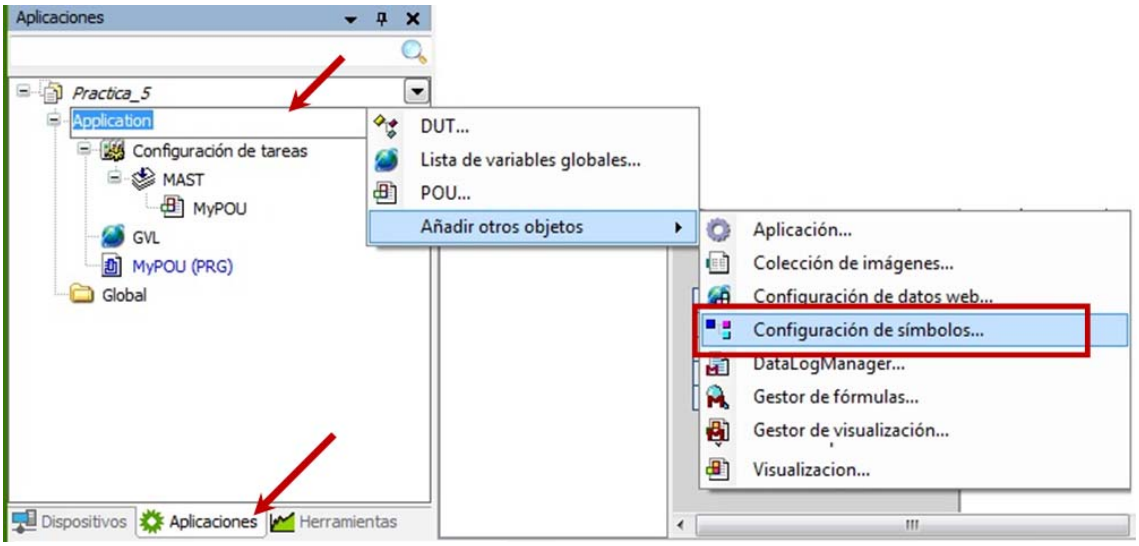


Nota: En esta práctica se ha realizado con un terminal HMIS5T, aunque hay varias de las maquetas que tienen instalado el terminal HMISCUA5, aunque la realización de los pasos de la práctica es la misma, comprobar el modelo de terminal táctil es el correcto ó elegir el adecuado.

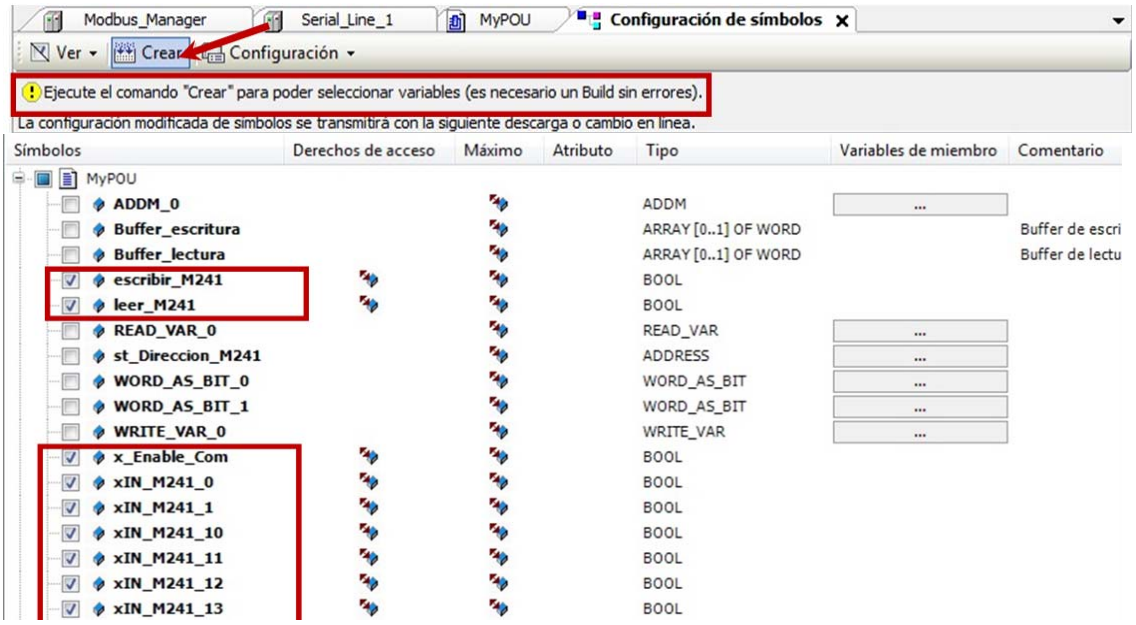
5.5.4.- Configurador de símbolos

Primero de todo, crearemos un programa, donde se han creado diferentes variables, que son las que se desean utilizar en la HMI. Hay que tener en cuenta que el programa tendrá que estar Generado y sin errores. (En este ejemplo las variables que vamos a utilizar en el HMI, serán las que hemos utilizado en el POU **'MyPOU'**.

1. Ahora seleccionar la pestaña **'Aplicación'** y, seleccionamos **'Application'** y haciendo clic con el botón derecho, seleccionamos **'Agregar Objeto / Configurador de símbolos'**.



2. Pulsar **'Crear'** si no veis las variables, ahora desplegar **'MyPOU'** y seleccionar todas aquellas variables que queremos publicar en el HMI. Una vez realizada esta acción las variables están publicadas (disponibles para el HMI).

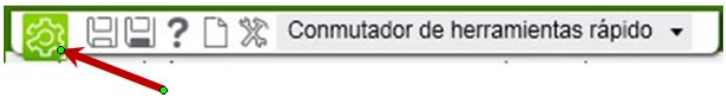


Nota: Las variables globales GVL solo se verán en el configurador de símbolos si estas están utilizadas en alguna parte del programa.

5.5.5.- Añadir variables en el Vijeo Designer

Una vez hecho el **'Crear'** en el **'configurador de símbolos'** del SoMachine, podemos ir al software de la HMI **'Vijeo Designer'** y las variables ya estarán disponibles directamente para ser utilizadas.

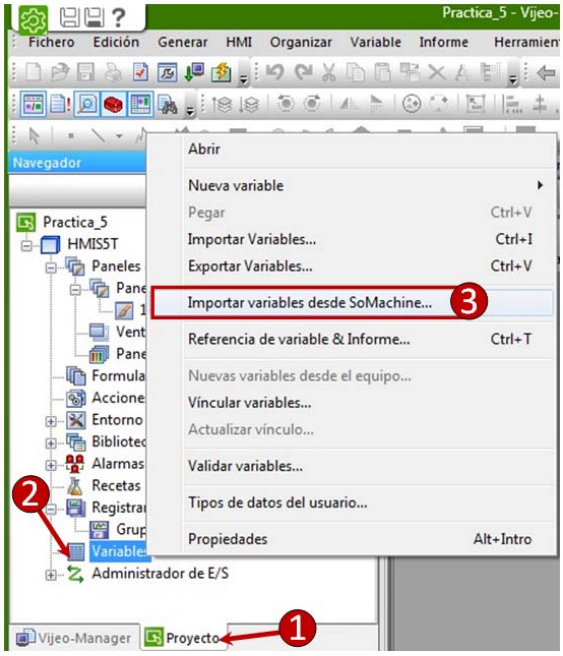
Ahora tenemos que ir al **'Vijeo Designer'** para ello pulsamos al '**icono del rodamiento'**, que se encuentra en el pop-up del *Logic Builder* para ir a la ventana general del SoMachine el **'SoMachine Central'**.



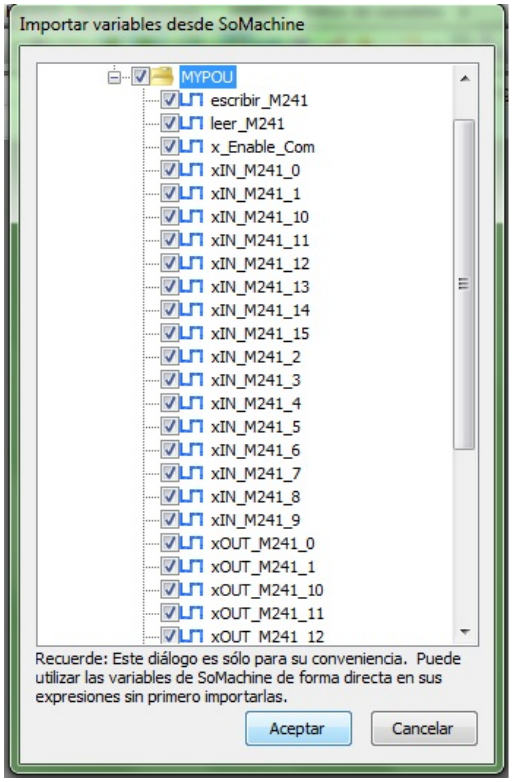
En el SoMachine Central, sólo tendremos que pulsar el botón de **'Vijeo Designer'**, en la barra de acceso rápido a los softwares.

Por ejemplo si queremos asociar la variable TP_ejemplo_2, que era booleana a un botón de la pantalla, al querer seleccionar la variable aparece una pestaña 'SoMachine' donde se encontrarán las variables seleccionadas en el '**Configurador de símbolos'.**

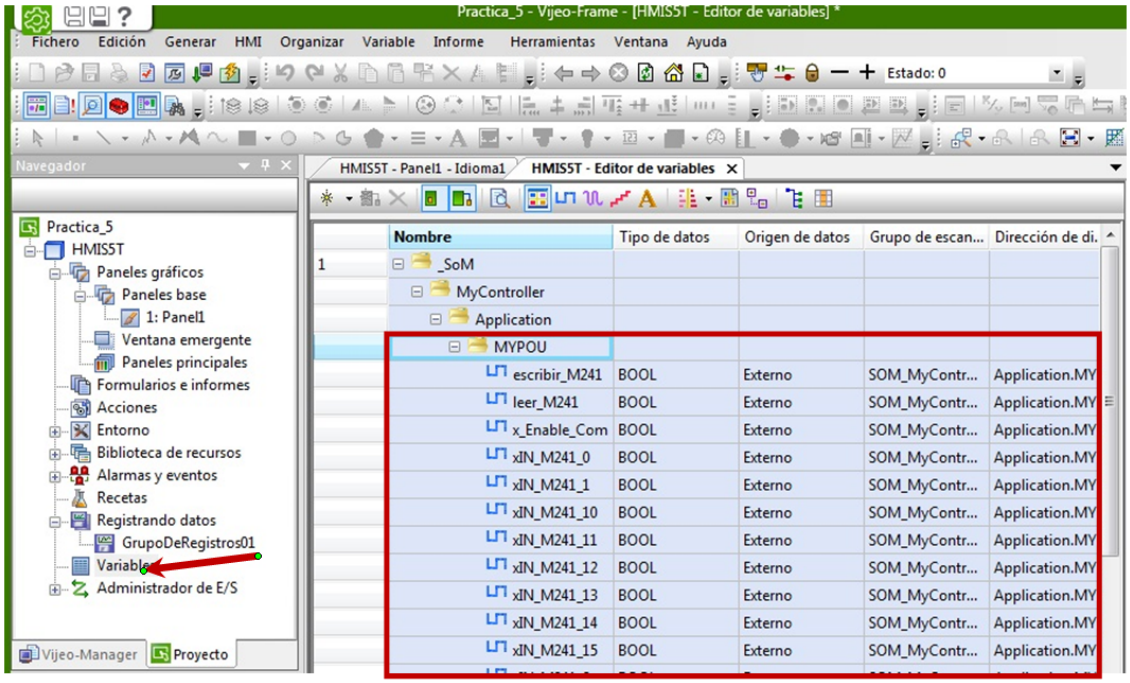
Para ver las variables del SoMachine y poderlas utilizar en el Vijeo Designer, lo unico que hay que hacer es en el Navegador del proyecto hay que seleccionar **‘Variables’**, botón derecho **‘Importar variables desde SoMachine’**



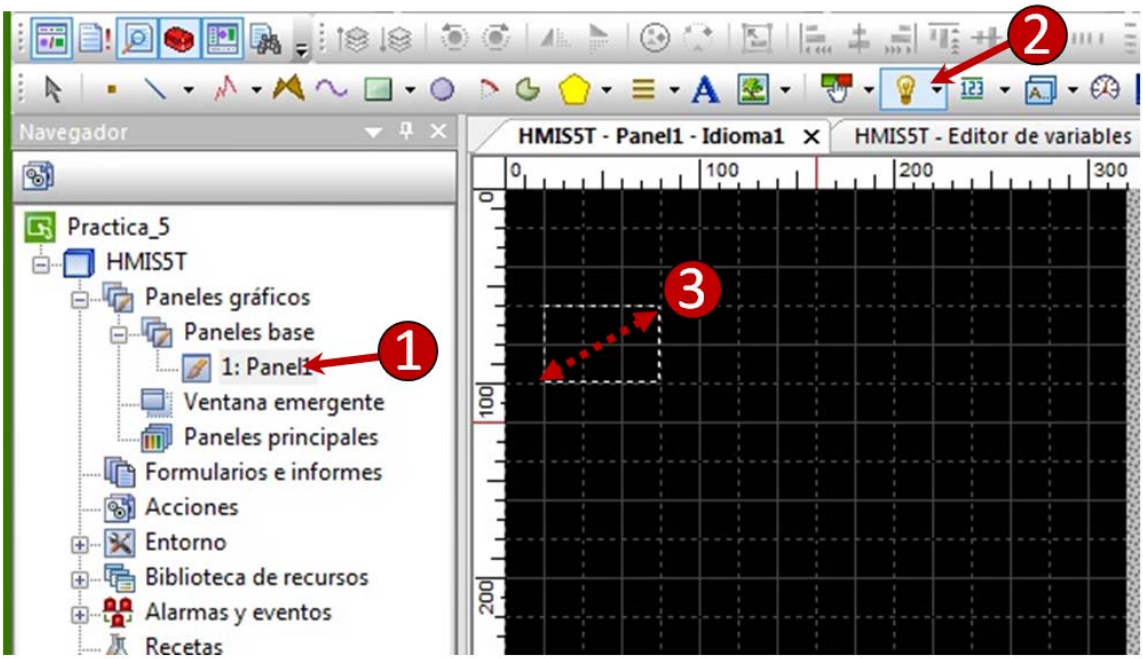
Aparecerá una ventana donde podemos seleccionar las variables que queremos importar.



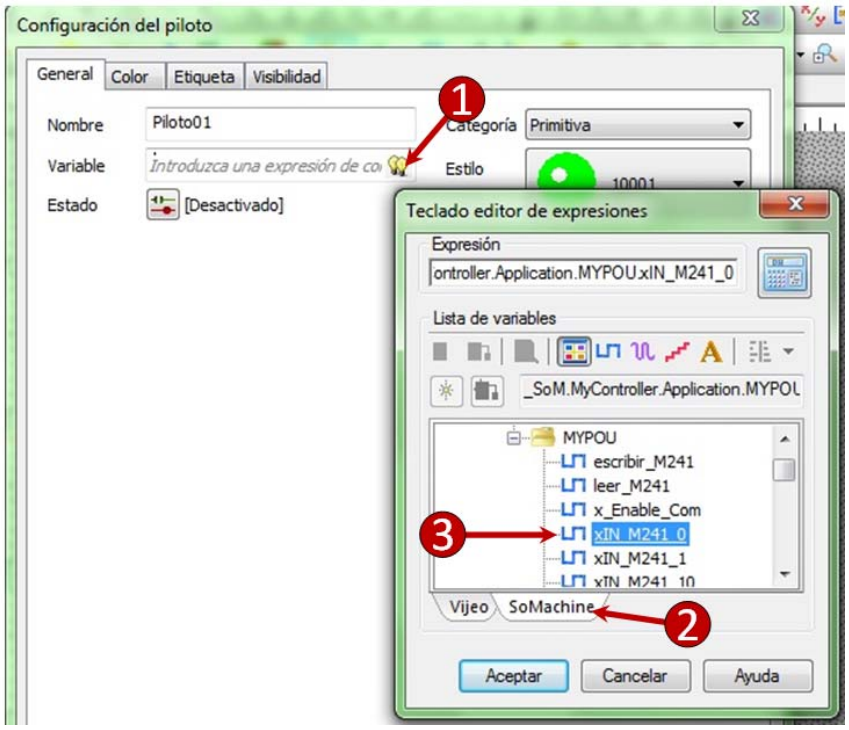
Una vez seleccionadas las variables, ya aparecerán en el editor de variables del Vijeo Designer.



Para utilizar las variables importadas, tenemos que ir por ejemplo al **'Panel 1'** dentro del navegador, al seleccionarlo aparecerá el panel en el área de trabajo, seleccionamos el icono de **'insertar piloto'** de la barra de herramientas y colocamos un piloto en el panel.



Al colocarlo aparecerá la ventana para la **'Configuración del piloto'**, en ella dentro de la pestaña **'General'**, en el campo **'Variable'** pulsaremos el icono para asociar una variable, aparecerá una ventana flotante con las diferentes variables creadas en la aplicación, en esta seleccionaremos la pestaña **'SoMachine'** que son aquellas variables que han sido importadas desde la aplicación SoMachine, y elegiremos '**xIN_M241_0'**, al aceptar ya nos aparecerá asociada.



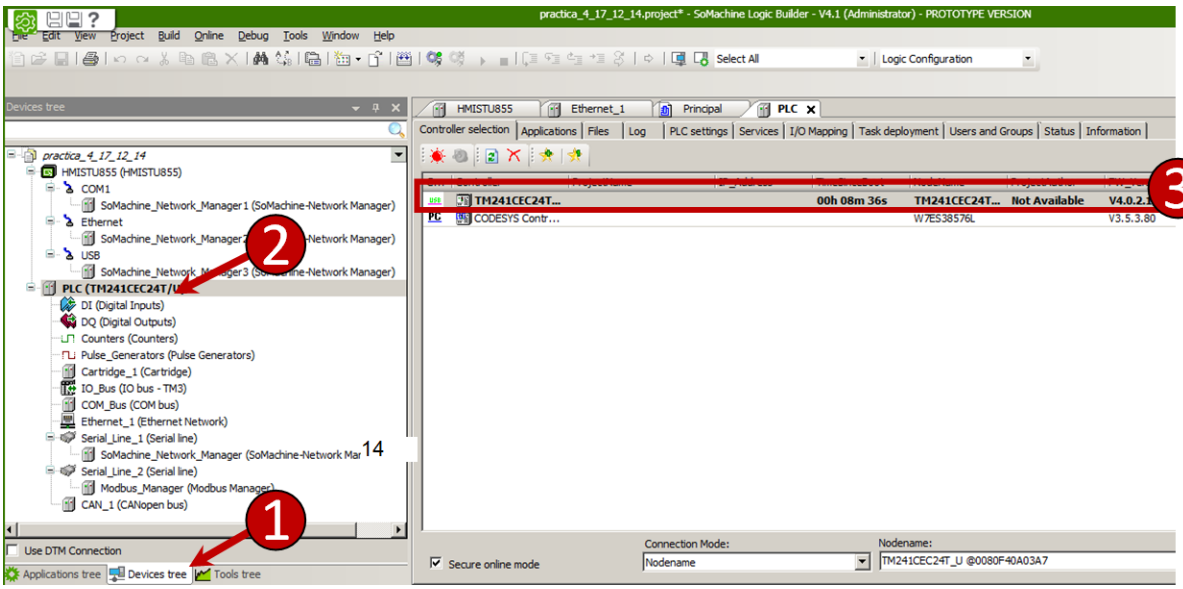
Nota: El sentido de esta práctica no es adentrarse en la programación de una aplicación de supervisión con Vijeo Desginer. Por lo que debido al tiempo del curso no es posible incluir todos los pasos para la realización de la pantalla.

Nos quedan las dos siguientes pantallas para el control del M241 a través del terminal gráfico conectado al M251:

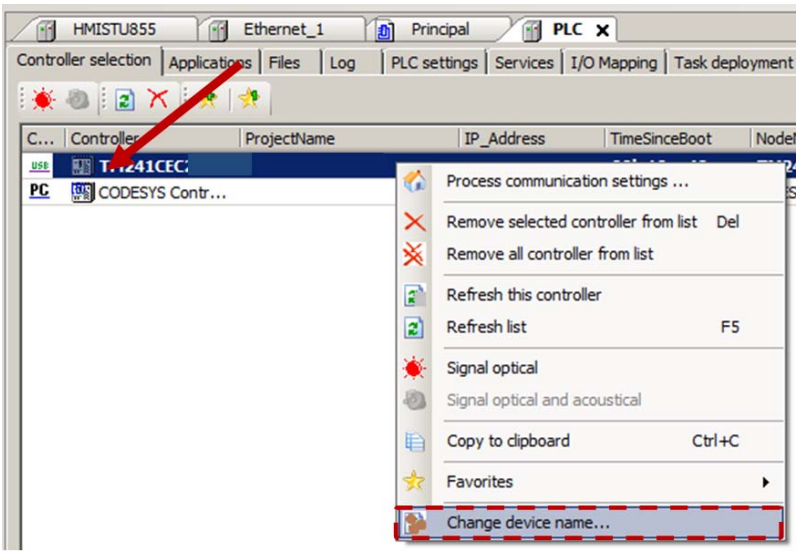
5.5.6.- Cambiar nombre del controlador en el SoMachine

Para poder vincular correctamente las variables entre el controlador y la HMI, es necesario conocer el **'Nombre'** específico del controlador. Al conectarnos por primera vez al controlador, sabremos el nombre por defecto, el nombre está compuesto por el **'modelo de controlador @ dirección MAC'(**Ejemplo: *TM251ESE@0080F40A03A7*).

1. En el SoMachine dentro de la ventana de **'Logic Builder'**, en la pestaña de **'Dispositivos'** del navegador haremos doble clic sobre el controlador, en el área de trabajo, seleccionaremos la pestaña **'Controller selection'.** Si tenemos el cable USB conectado al controlador, este escaneará automáticamente y nos mostrará los controladores conectados.



2. Ahora seleccionamos el controlador al que deseamos cambiar el nombre, primero hacemos doble clic sobre él, este se pondrá en negrita. Luego con el controlador seleccionado, con el botón derecho desplegamos el menú de opciones y seleccionamos **'Cambiar el nombre del dispositivo…'**



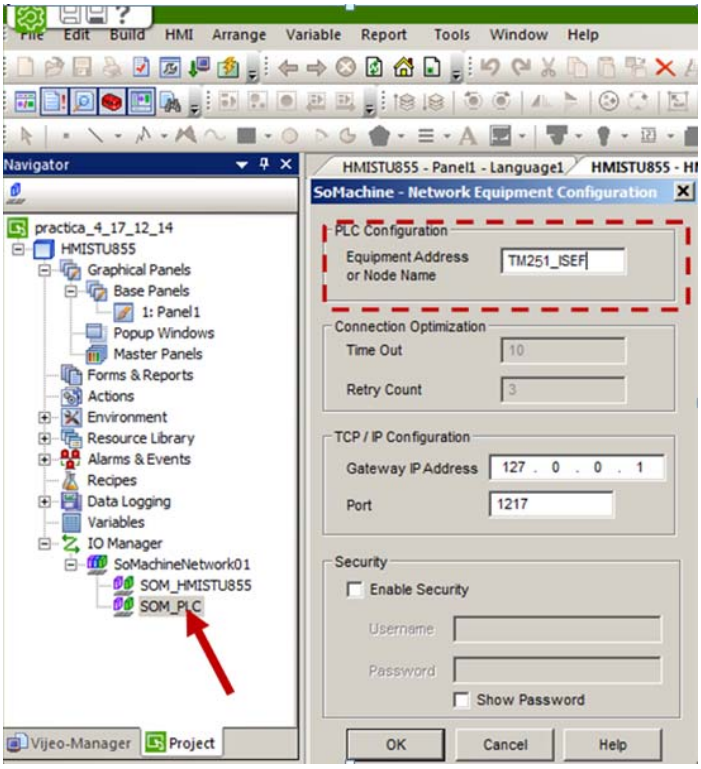
3. Aparecerá una ventana donde nos indica el nombre actual y en el campo **'Nuevo'** escribiremos el nombre que deseamos para este controlador (Ejemplo: TM251_ISEF).

4. Ahora en la columna **'Node name'** habrá cambiado el nombre por el nuevo.

5.5.7.- Añadir nombre del controlador en el Vijeo Designer

Ahora, una vez sabemos el nombre del controlador, volvemos abrir el Vijeo Designer y en el Navegador seleccionamos la pantalla y en la pantalla '**General'** colocaremos la IP de la pantalla en el campo **'Dirección IP'.** También en el campo **'Descarga'** pondremos **'USB'** ya que la primera descarga del programa se tendrá que realizar por USB (no se podrá hacer una descarga múltiple la primera vez.

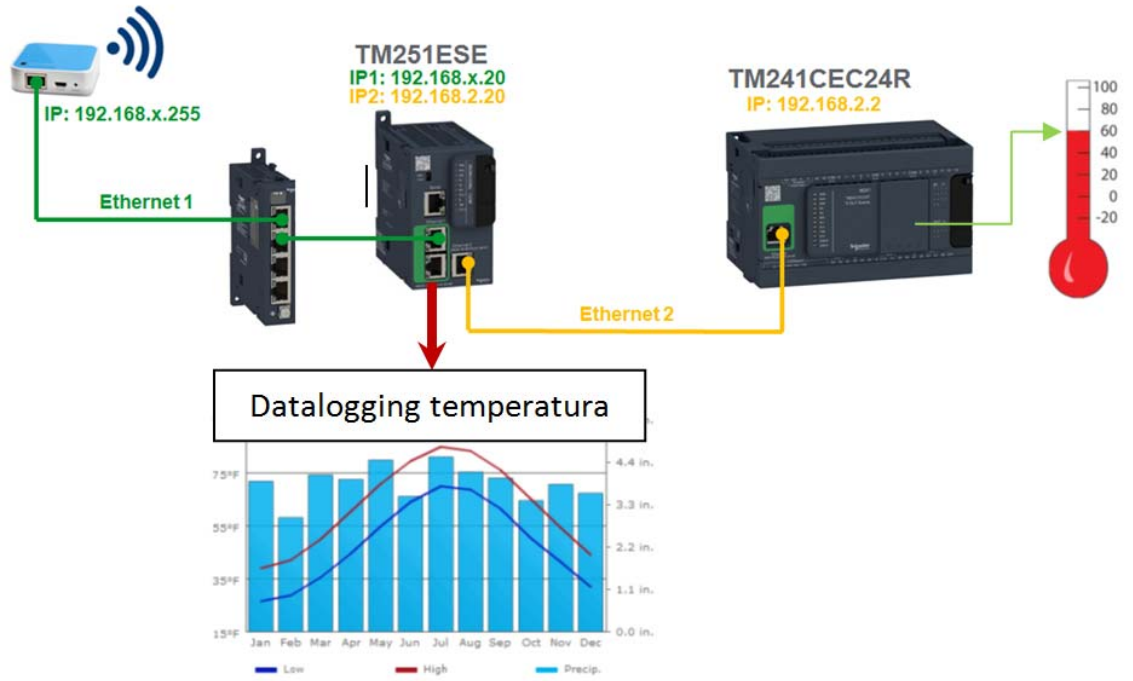
Una vez hemos colocado la IP en la HMI, en el navegador desplegaremos el campo '**I/O Manager / SoMachine Network 01'** y haremos doble clic en el **'SOM_PLC'** y en la ventana flotante de configuración, en el campo **'Name Equipement or Node name'** escribiremos en nombre del controlador que hemos escrito en el SoMachine (Ejemplo: TM251_ISEF).



Una vez descargado el programa en el HMI y en el controlador, las variables quedarán linkadas y listas para ser utilizadas.

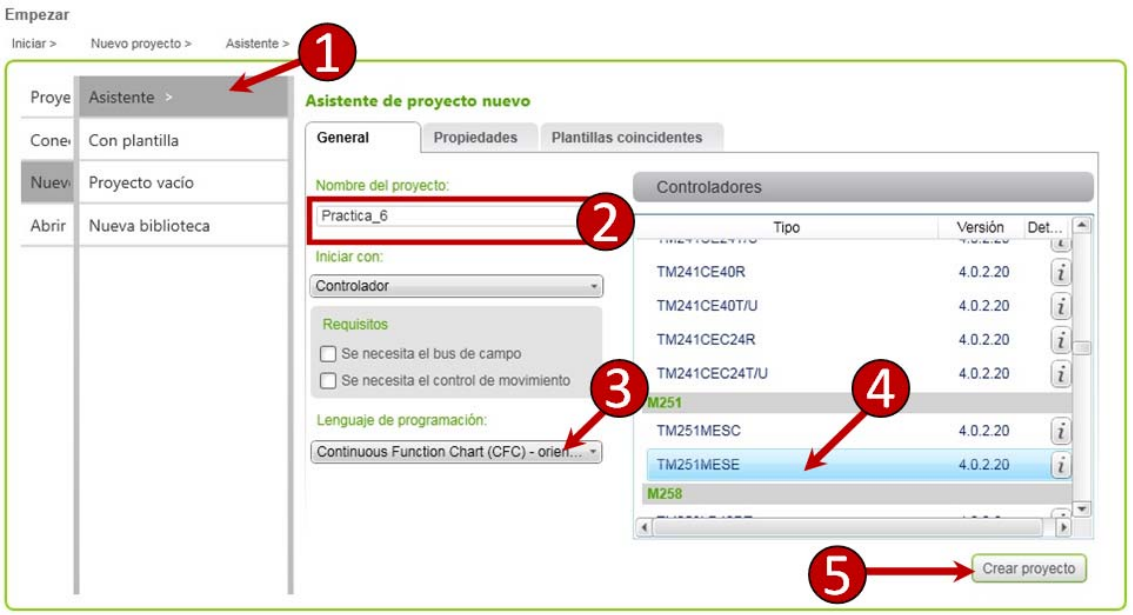
5.6.- PRÁCTICA 6: M251 I/O SCANNING + DATALOGGING

En este ejercicio realizaremos configuraremos y programaremos la comunicación modbus entre el M251 y el M241.



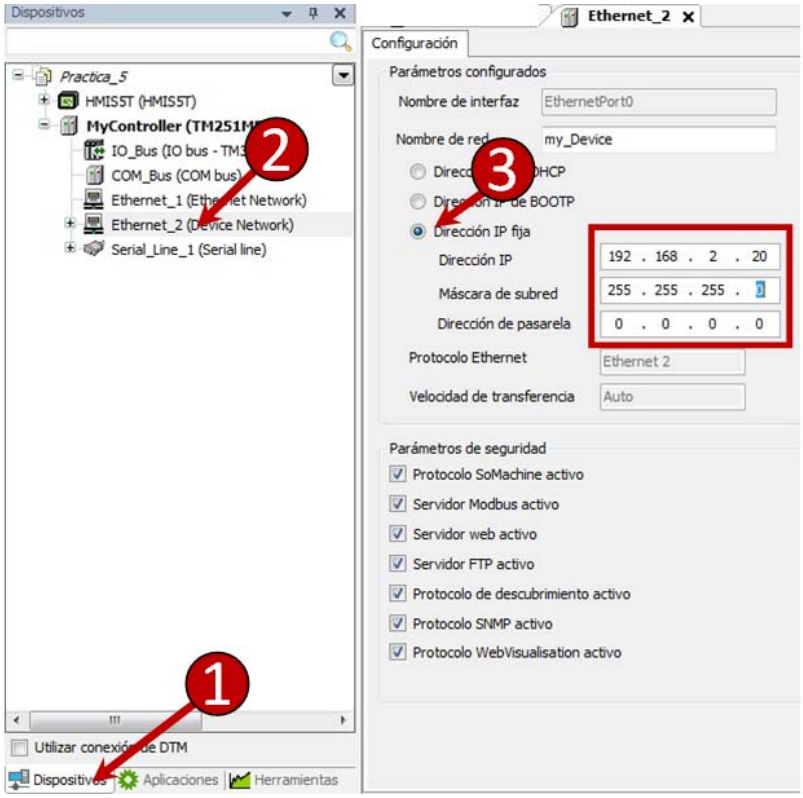
5.6.1.- Programación de la comunicación Modbus I/O Scanning TCP/IP.

En la ventana del asistente de creación del proyecto dentro de la pestaña **'General'**, escribiremos el nombre de nuestro proyecto *(Practica_6)*, en **'Iniciar con'** seleccionaremos **'Controlador'** y en la parte de la derecha buscaremos el controlador deseado *(en este caso TM251ESE)*, por último en el '**lenguaje de programación'** seleccionamos el lenguaje con el queremos programar el primer POU *(en este caso Continuous Funtion Chart (CFC) – orientado a página)*. Por último pulsamos '**Crear Proyecto'** para empezar.

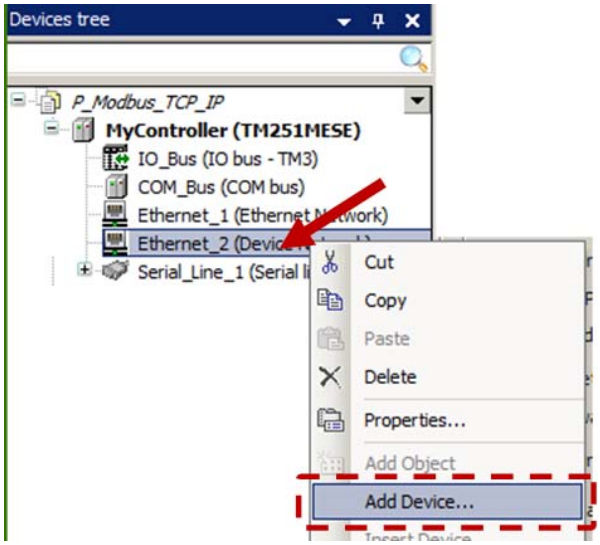


Cuando haya aparecido la ventana principal del SoMachine (*SoMachine Central*), nos aparece el Flujo de trabajo típico de un proyecto automatización, para entrar en la ventana de programación tendremos que pulsar el botón de **'Controlador'**, ya que dentro del flujo la configuración del controlador ya la hemos realizado previamente con el asistente.

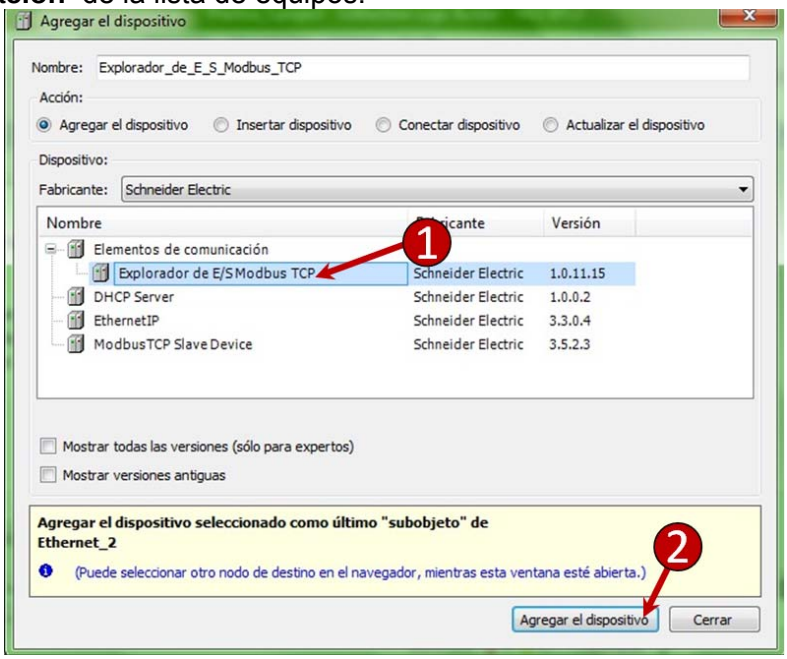
Una vez dentro de la ventana de programación (Logic Builder). En la pestaña **'Dispositivos'** del navegador del proyecto, hay que configurar la IP del puerto **'Ethernet 2'** del M251, seleccionaremos la **'Dirección IP fija'** *(IP: 192.168.2.20).*



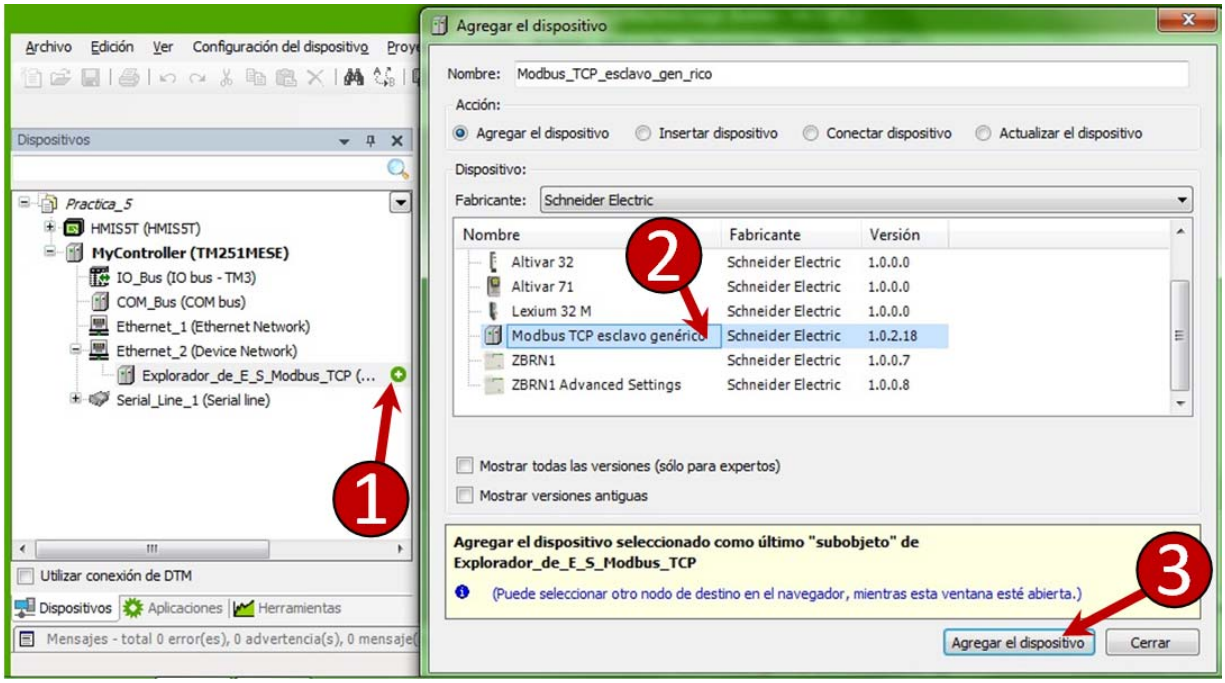
Ahora seleccionar el puerto **'Ethernet 2'**, en la pestaña dispositivos del navegador y pulsar botón derecho del ratón sobre él. En el menú de opciones que aparece seleccionar **'Añadir equipo'**.



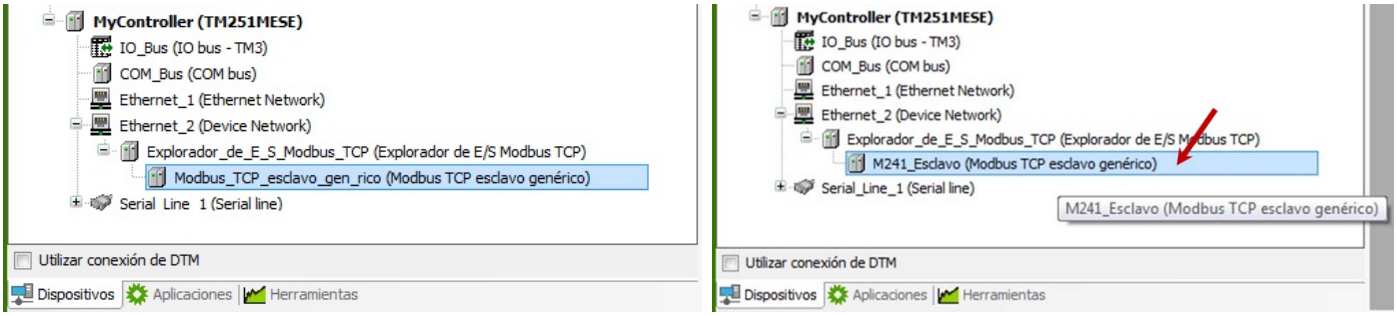
Añadir el elemento '**Explorador de E/S Modbus TCP',** que está dentro de **'Elementos de comunicación'** de la lista de equipos.



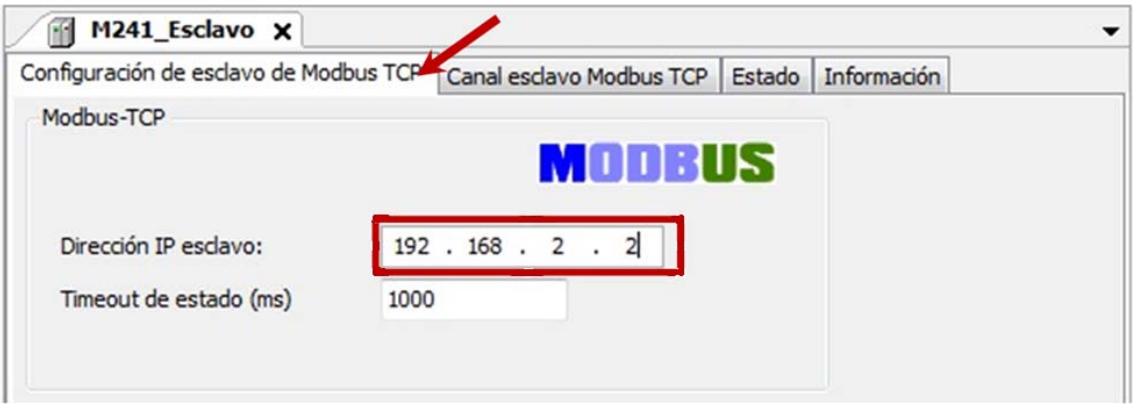
En la pestaña de 'Dispositivos' del navegador por debajo de **'Ethernet 2'**, se ha añadido el **'Explorador de E/S Modbus TCP'**, seleccionarlo y pulsar botón derecho y seleccionar '**Añadir Equipo'** para añadir el esclavo**.** Añadir como esclavo modbus TCP/IP **'Modbus TCP esclavo genérico'** al maestro.



Cambiar el nombre de '**Modbus TCP esclavo genérico'** al de **'M241_Escalvo'**

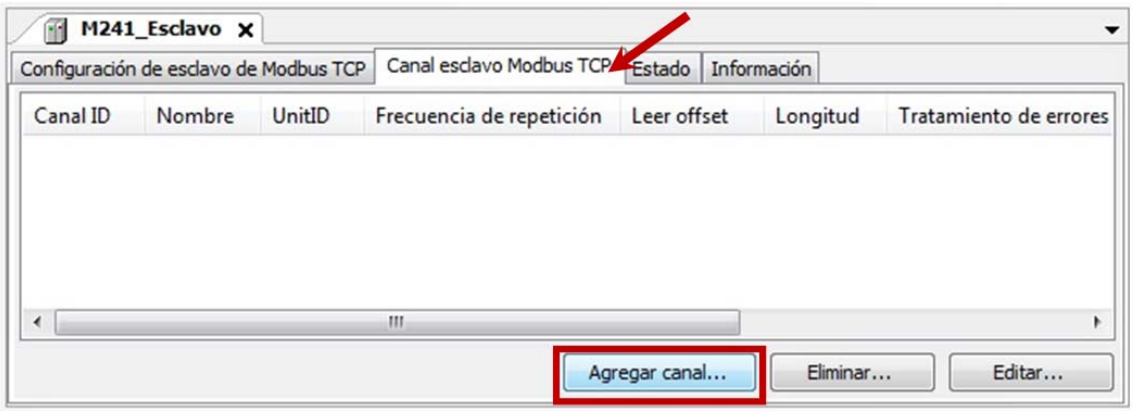


Hacer doble clic sobre el esclavo **'Modbus TCP esclavo genérico'** que aparece asociado al maestro '**Explorador de E/S Modbus TCP',** del puerto **'Ethernet 2'** en el navegador, y configurar la '**Dirección IP del esclavo'** (*192.168.2.2*), que se encuentra en la pestaña **'Configuración de esclavo de Modbus TCP'**.

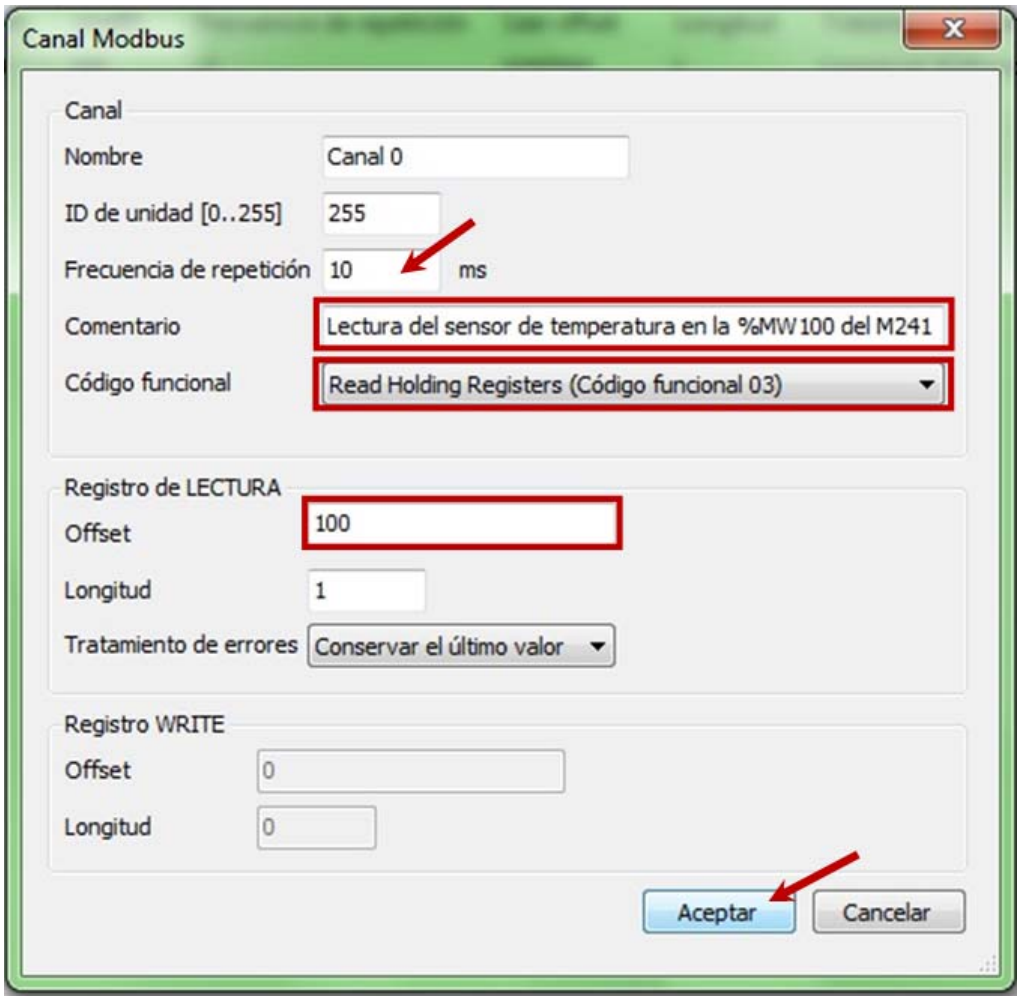


5.6.2.- Configuración de los canales de lectura y escritura con un esclavo genérico

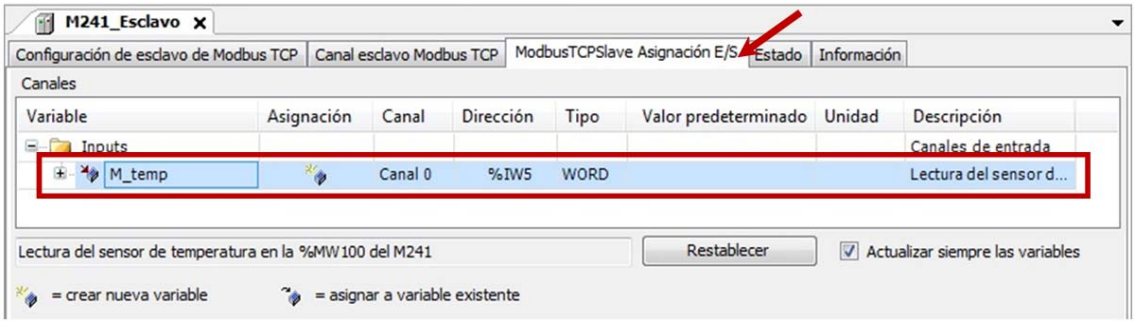
En el caso que hubiéramos seleccionado un esclavo Modbus TCP/IP genérico, tendríamos que configurar las llamadas a los registros modbus de los esclavos que queremos leer y escribir. Tendríamos que ir a la pestaña '**Canal esclavo Modbus TCP'** e ir añadiendo canales **'Agregar canal…'**.



En el canal en el campo 'Tipo de acceso' seleccionar **Read Holding Registers'** *(ya que, solo tenemos la intención de leer el valor de la temperatura que nos ofrece el M241, en este caso en concreto no tendremos que escribir nada en el M241)*, modificar el tiempo de ciclo **1000** ms. Rellenar los campo de 'Desplazamiento' con la dirección que se quiere leer y escribir y la longitud en words *(en el caso del M241 el registro '100' (porque la medición de temperatura se guarda en la %MW100 del M241), y la longitud será 1 (porque solo queremos leer un registro consecutivo).*



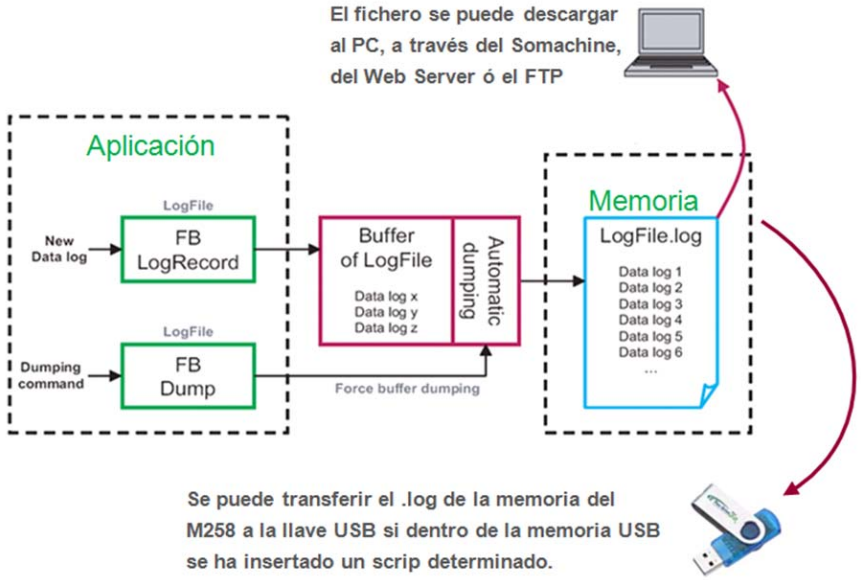
Cuando se ha creado el canal de I/O scanning hay que crear las variables asociadas a ese canal para utilizarlas en el programa para esta comunicación Modbus TCP/IP. En este caso, seleccionar la pestaña **'ModbusTCPSlave Asignación E/S'** y declarar las variables, en el caso actual solo tendremos que declarar la variable 'M_temp'.



5.6.3.- Gestión de ficheros de datos - DataLogging

Puede ser necesario para la aplicación archivar ciertas variables, para poder ser descargadas y tratadas por el operario. Para ello necesitamos poder guardar los datos en ficheros.

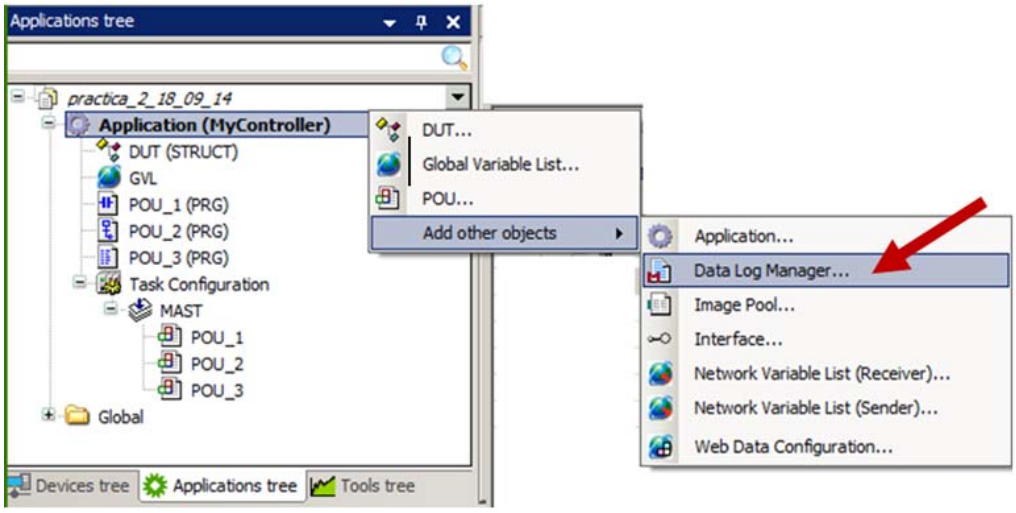
Con el bloque de función FB **'LogRecord'** se crean los ficheros **'.log'** en el M251 Inicialmente se escriben los datos en un buffer y son traspasados a la memoria del M251 cuando el buffer se encuentra al 80% o cuando se ejecuta la función **'Dump'**.



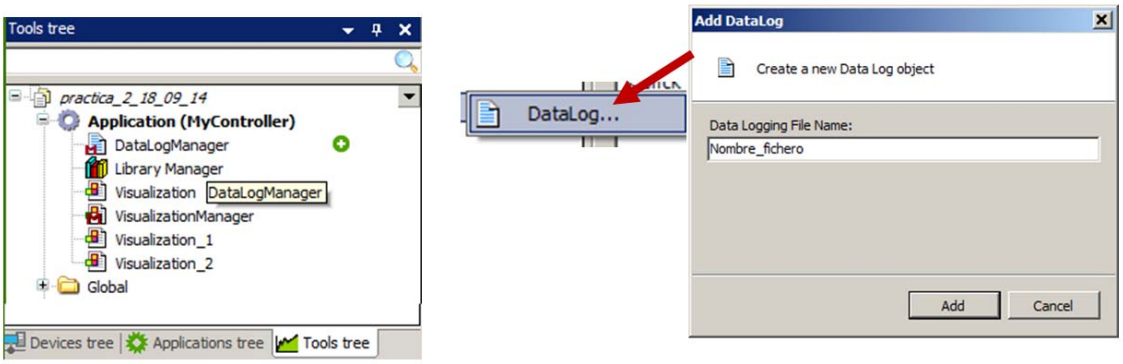
El fichero '**.log'** que se encuentra en la memoria del M251 puede ser traspasado a una memoria USB, para ello dicha memoria deberá tener un scrip en su interior, y cuando se inserte en el M251 automáticamente obtendrá el fichero .log

Creación de un fichero .log, a continuación se muestran los pasos a seguir para realizar el proceso de creación de fichero.log y traspaso de datos a memoria USB.

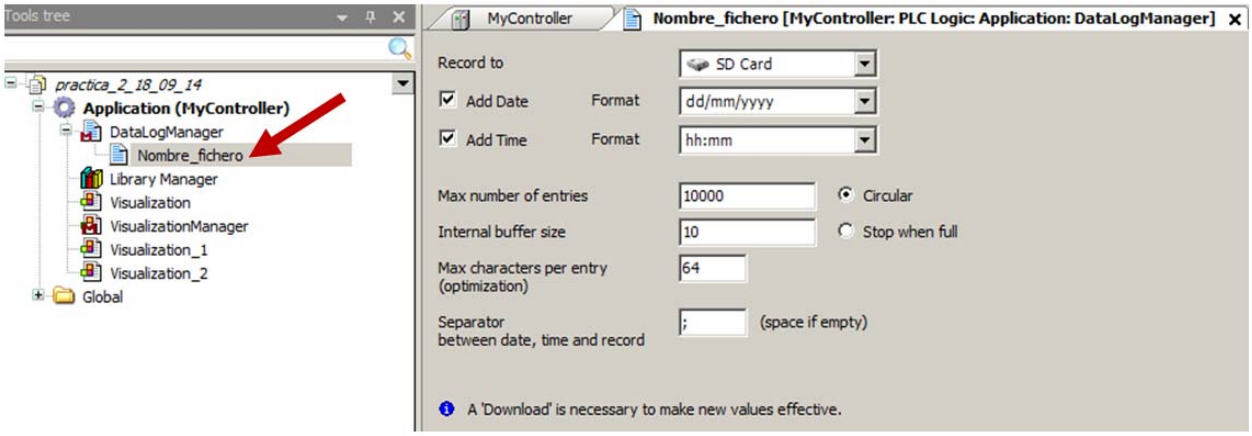
Añadir un DataLog Manager al proyecto. Sobre '**Application**', botón derecho **'Agregar Objeto'** -> **DataLogMAnager**



Ahora en la pestaña de 'Herramientas', insertar el Datalog. Sobre DataLogManager, botón derecho **'Agregar Objeto'** -> '**DataLog'**. En el campo **'Nombre'** escribir el nombre que se desea que tenga el fichero.



En la pestaña Logfile que se abre configurar el datalog (formato de fecha, de hora…etc).



5.6.4.- Insertar datos en el fichero

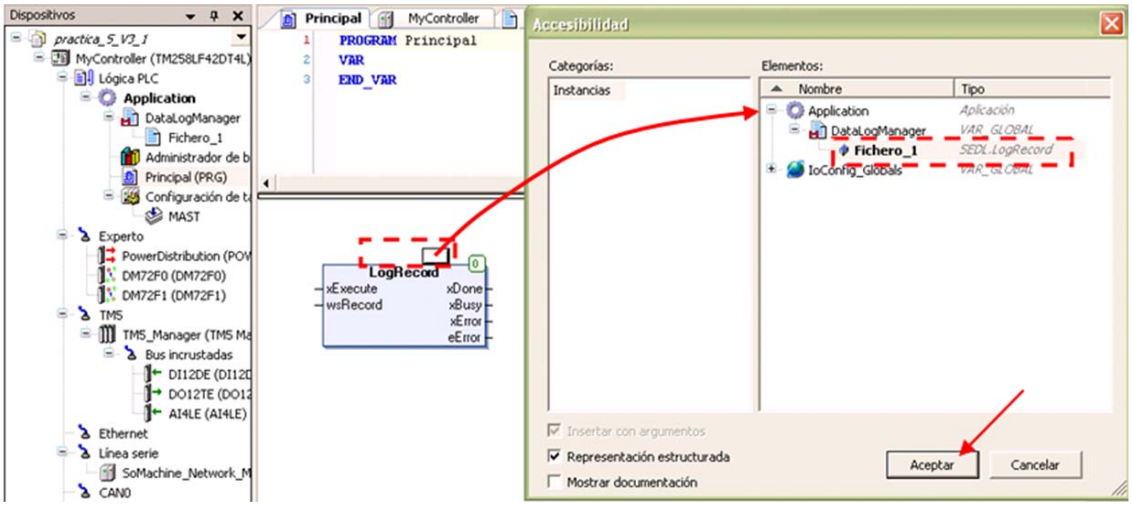
Una vez creado el fichero .log, se tendrá que utilizar los bloques de función FB **'LogRecord'** para abrir el fichero y guardar los datos que se deseen. Inicialmente se escriben los datos en un buffer y son traspasados a la memoria del M251 cuando el buffer se encuentra al 80% o cuando se ejecuta la función **'LogRecordDump'** que se guardan en ese preciso momento.

Los pasos para la inserción del bloque de función LogRecord.

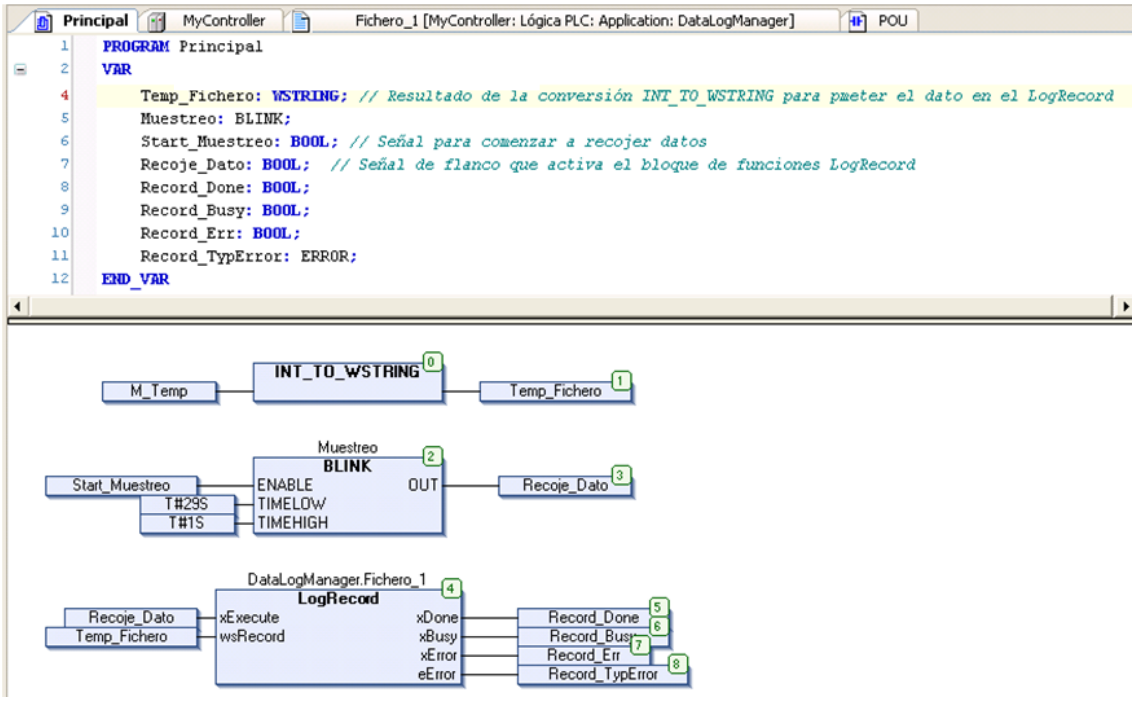
Abrir el POU previamente creado para la inserción del bloque de función.

Después introducir el bloque función **'Insertar llamada a módulo'** y elegir el bloque de funciones **'LogRecord'** que se encuentra en la librería **'SEDL'.**

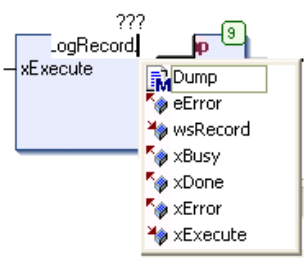
Una vez insertado el bloque de función en la parte de arriba hay que linkar en la llamada a la instancia al fichero_1 de la siguiente manera.



Rellenar las entradas y salidas tal y como se muestra en la figura (*Ejemplo para guardar Medida de Temperatura cada minuto*), teniendo en cuenta que la entrada del bloque '**wsRecord'** tiene que tener formato '**wstring**'.

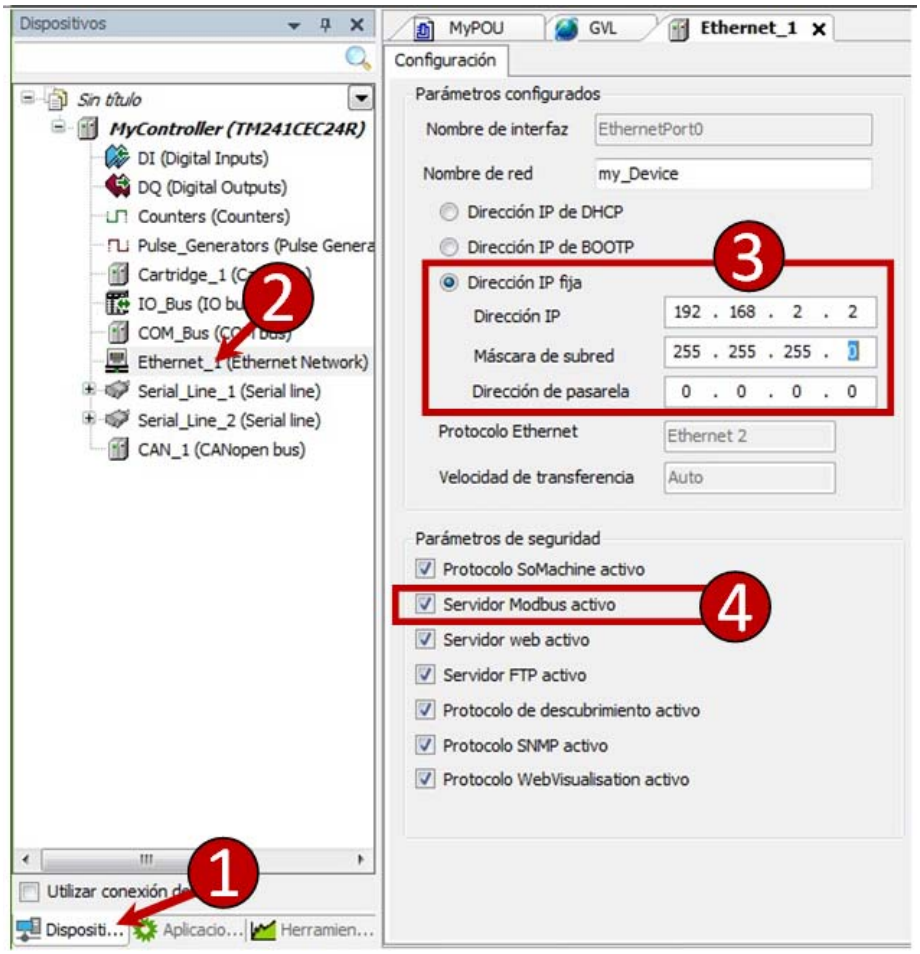


Para llamar al bloque '**LogRecord.Dump'** que guarda los datos del buffer, en la memoria cuando se desea. Escribir **'LogRecord'** en el módulo, poner un punto **'.'** y seleccionar la función **'Dump'**.

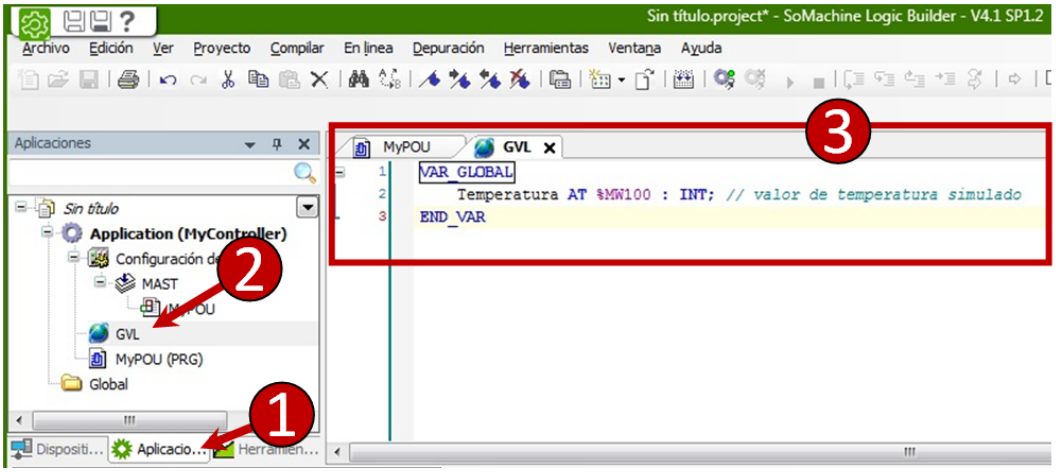


La programación del bloque de la siguiente manera.

Luego en el M241 tendremos que realizar la configuración de la dirección IP del M241.



Definir la variable **‘temperatura’** que será una variable global con la dirección **%MW100**.



Y un programa como el siguiente para simular el cambio de la temperatura.