Automatización Avanzada (37800)

Máster en Automática y Robótica

Manual de prácticas

Introducción al software UnityPro y a los PLCs M340 de Schneider

Francisco Andrés Candelas Herías

Grupo de Innovación Educativa en Automática

Universitat d'Alacant
Universidad de Alicante

Introducción al software UnityPro y a los PLCs M340 de Schneider

Contenido

1. UnityPro 3
2. Creación de un proyecto 4
2.1. Creación del proyecto y configuración del bastidor 4
2.2. Configuración de los módulos del bastidor 6
Configuración del procesador 7
Configuración de los módulos de E/S digitales 7
Configuración de módulos de entradas analógicas 8
Configuración de una conexión de red 9
2.3. Propiedades del proyecto 11
3. Programación 11
3.1. Variables y tipos. 11
3.2. Direccionamiento directo 12
3.3. Gestión de variables 13
3.4. Estructura del programa 15
3.5. Creación de secciones de programa y rutinas 17
3.6. Bloqes de funciones 18
3.7. Programación con el lenguaje de diagrama de contactos (LD) 18
3.8. Generar y validar un programa 20
4. Ejecución de la aplicación en el PLC 21
4.1. Carga de programas en el simulador y simulación 21
4.2. Análisis y depuración 23
4.3. Conexión del PLC real al PC y carga del programa 25
4.4. Indicadores de los módulos 26
5. Desarrollo y utilización de bloques de funciones 27
5.1. Tipos de bloques de funciones 27
5.2. Uno de los bloques de función en UnityPro 28
5.3. Creación de DFB 29
5.4. Reutilización de los DFB 34
6. Secciones GRAFCET en UnityPro 34

1. UnityPro

Unity es el entorno software/hardware de Schneider Electric para la gestión de sus plataformas de automatización. El mismo entorno sirve para las diferentes gamas de PLC de la marca: Quamtum, Premium, Modicon 340 y Atrium.

Dentro de Unity, la herramienta gráfica UnityPro es la que se utiliza para el desarrollo, explotación y mantenimiento de aplicaciones para los PLCs. Esto es, UnityPro es una herramienta "todo en uno" que permite se puede configurar, programar, simular, depurar y monitorizar un PLC. Además, es posible hacer todo ello de una forma bastante independiente del hardware del PLC, después de haber realizado una configuración inicial. La Figura 1 muestra el aspecto de UnityPro, y describe sus principales ventanas.

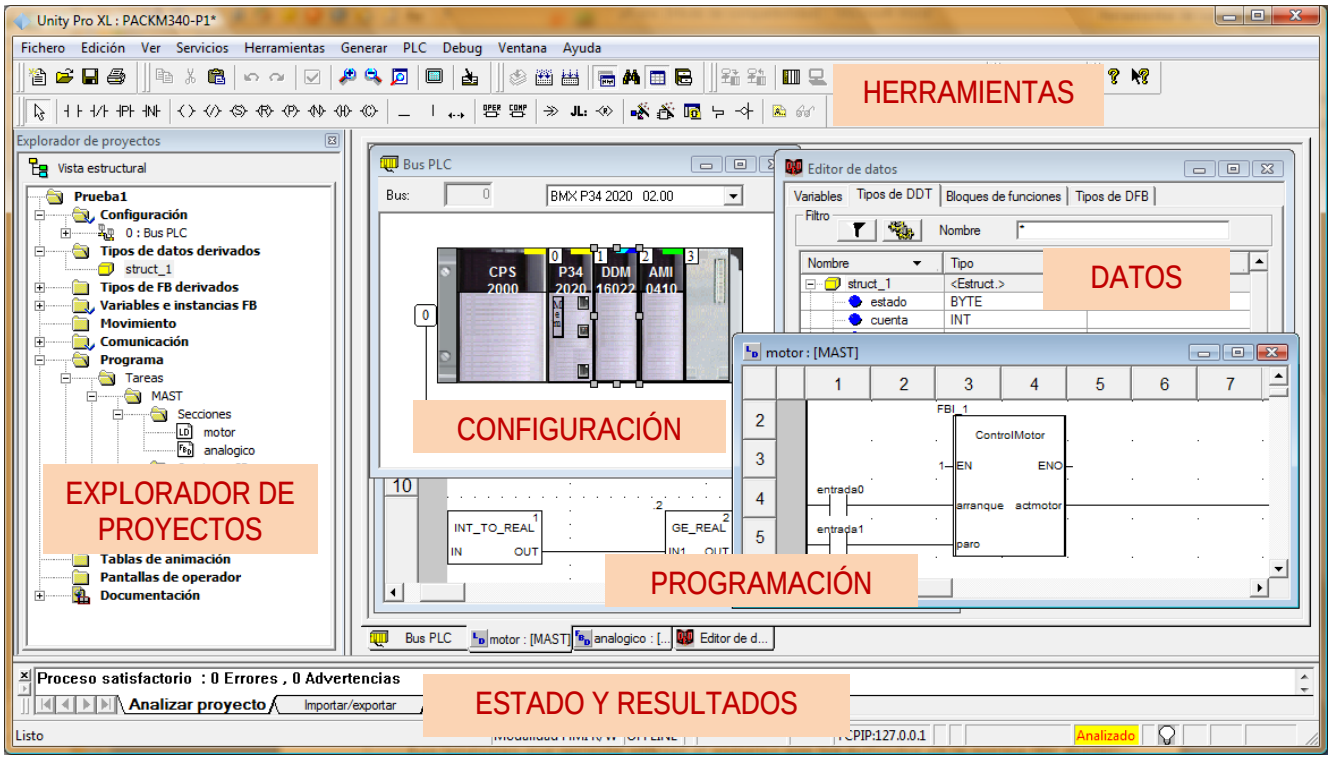


Figura 1. Interface de UnityPro.

Los lenguajes que permite utilizar UnityPro son los definidos en la norma IEC 61131:

- LD (LaDder): diagrama de contactos.
- ST (Structured Text): texto estructurado.
- SFC (Sequential Function Chart): diagrama secuencial de funciones (GRAFCET).
- FBD (Function Block Diagram): Diagrama de bloques funcionales.
- IL (Instruction List): Lista de instrucciones.

UnityPro también incorpora las funciones avanzadas para facilitar configurar los módulos de los PLCs, gestionar de las comunicaciones de planta, programar en multitarea y con tareas rápidas, crear variables no asignadas, definir y usar estructuras y matrices de datos, definir bloques de usuario para reutilizar código, ver en tiempo real cómo evoluciona el programa y las variables durante la ejecución, depurar el programa paso por paso, etc.

Este manual básico explica primero, en la sección 2 como utilizar las funcionalidades básicas de UnityPro para configurar un PLC como el que se usará en las prácticas de la asignatura. Posteriormente en la sección 3 se describe como se realiza la programación básica mediante diagramas de contactos. La sección 4 resume los pasos para cargar los programas en un PLC simulado o real y cómo ejecutarlos. Finalmente, las secciones 5 y 6 muestran como realizar programas más complejos mediante la creación e bloques de usuario y el uso del lenguaje SFC.

Adicionalmente a este manual básico, el alumno tiene a su disposición el completo sistema de ayuda que incorpora la aplicación UnityPro, y los manuales en formato PDF que ofrece Schneider.

2. Creación de un proyecto

2.1. Creación del proyecto y configuración del bastidor

Para empezar el desarrollo de una aplicación, el primer paso es crear un nuevo proyecto en UnityPro. Para ello se debe escoger la opción Fichero-Nuevo o pulsar sobre el botón . Después hay que de elegir el procesador que se va a usar dentro de una familia, como muestra la Figura 2. Para las prácticas se usará el procesador BMX P34 2020 de la familia Modicon M340. Este procesador es el que incluye puertos de comunicaciones Modbus y Ethernet.

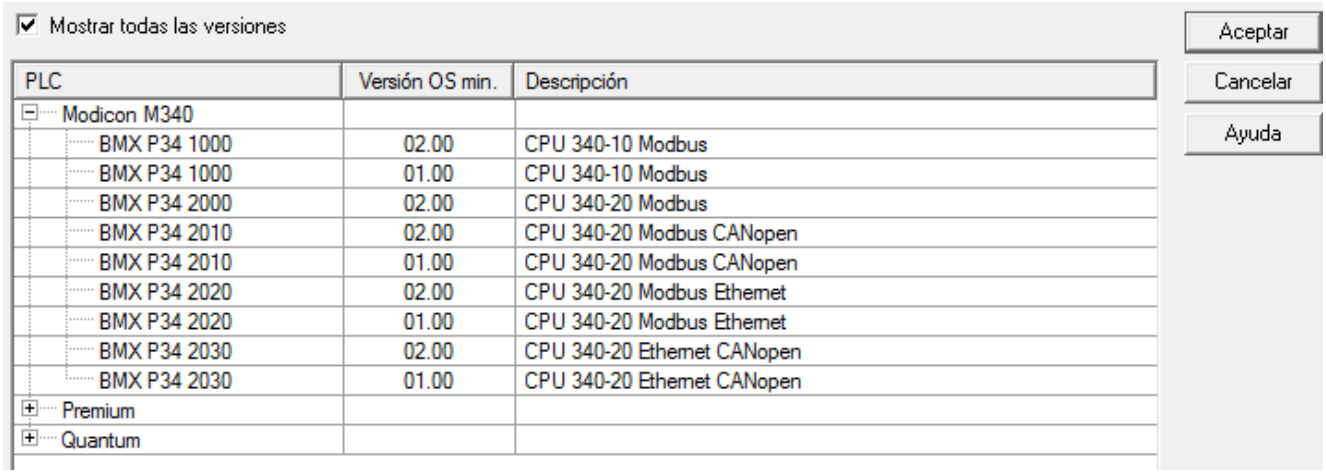


Figura 2. Selección de procesador.

UnityPro creará la base para un nuevo proyecto adecuado al procesador seleccionado. En la parte de la izquierda del entorno gráfico se dispone de la ventana "Explorador de Proyectos" (Figura 3) que permite un acceso rápido a todas las secciones de configuración, datos, comunicaciones y programas del PLC. Estas secciones son mostradas como una estructura jerárquica de carpetas. Si no está visible el explorador de proyectos, se puede mostrar con la opción del menú Herramientas-Explorador de Proyectos.

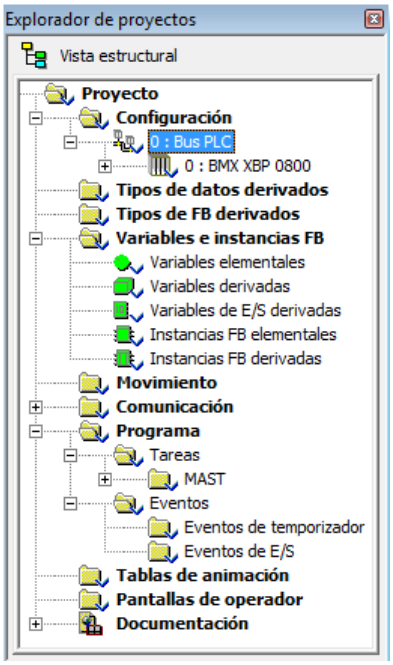


Figura 3. Explorador de proyectos.

En el Explorador de Proyectos se debe desplegar la opción Configuración y hacer doble clic sobre para acceder sobre "Bus PLC" para que aparezca la ventana de configuración de los módulos de este PLC. Inicialmente se muestra un bastidor de 8 slots con el procesador elegido y un módulo de alimentación escogido automáticamente. Se puede cambiar el modelo de bastidor haciendo doble clic en los extremos del mismo, como muestra la Figura 4.

Figura 4. Bastidor de 8 slots con los módulos de alimentación y CPU.

En las prácticas se empleará una configuración de un solo bastidor de 4 slots (BMX XPB 400), por lo que hay que seleccionar ese modelo (Figura 5).

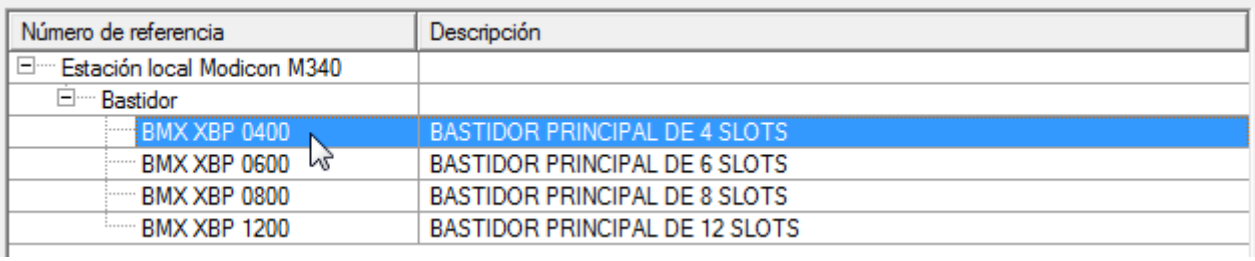


Figura 5. Selección del bastidor de 6 slots.

Es posible añadir más bastidores por debajo del bastidor número 0 para aumentar el número de módulos de expansión, haciendo doble clic donde muestra la Figura 6.

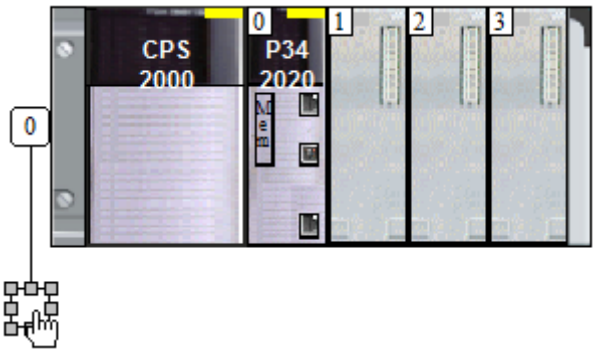


Figura 6. Ampliación del número de bastidores.

Cabe destacar que en el slot número 0 del bastidor 0 siempre se debe colocar una CPU, y que en los slots a la izquierda del número 0 siempre debe haber una fuente de alimentación para cualquier bastidor.

Escogidos el bastidor, la fuente y la CPU hay que seleccionar los módulos entrada/salida (E/S). Se puede añadir un módulo en cada slot haciendo doble clic sobre el slot vacío. También se puede suprimir un módulo seleccionándolo con el ratón y pulsado la tecla DELETE, quedando su slot vacío. Para cambiar el módulo de un slot, primero se debe eliminar el módulo actual y luego añadir el nuevo en su lugar.

En las prácticas se usará la siguiente configuración de PLC modular de la serie MODICON M-340, que se corresponde con el esquema y el PLC mostrados en la Figura 7 (el slot 3 queda vacío):

- BMX XBP 0400: bastidor para 4 módulos más la fuente de alimentación.
- BMX CPS 2000: Fuente de alimentación para 220V.
- BMX 34 2020: CPU con USB, Modbus/serie (RJ45) y Ethernet (RJ45). Se debe seleccionar además la **versión 01.00** de esta CPU.
- BMX DDM 16022: 8 entradas digitales, y 8 salidas digitales por transistor PNP, todas ellas aisladas.
- BMX AMI 600: 4 entradas analógicas más 2 salidas analógicas sin aislar, configurables en varios rangos de voltaje o corriente.

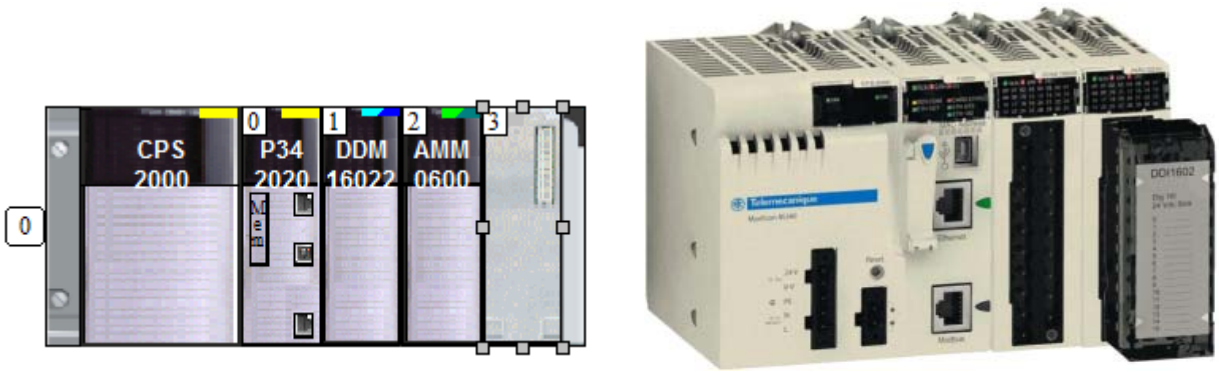


Figura 7. Configuración usada en las prácticas.

2.2. Configuración de los módulos del bastidor

En cualquier momento del desarrollo, se puede abrir a la ventana con la configuración del bastidor del PLC haciendo doble clic en la carpeta Configuración-Bus PLC del Explorador

de Proyectos. Desde esta ventana, se puede acceder a la ventana de configuración de cada módulo con solo hacer doble clic él. Como ejemplo, la Figura 8 muestra la configuración de un módulo de E/S.

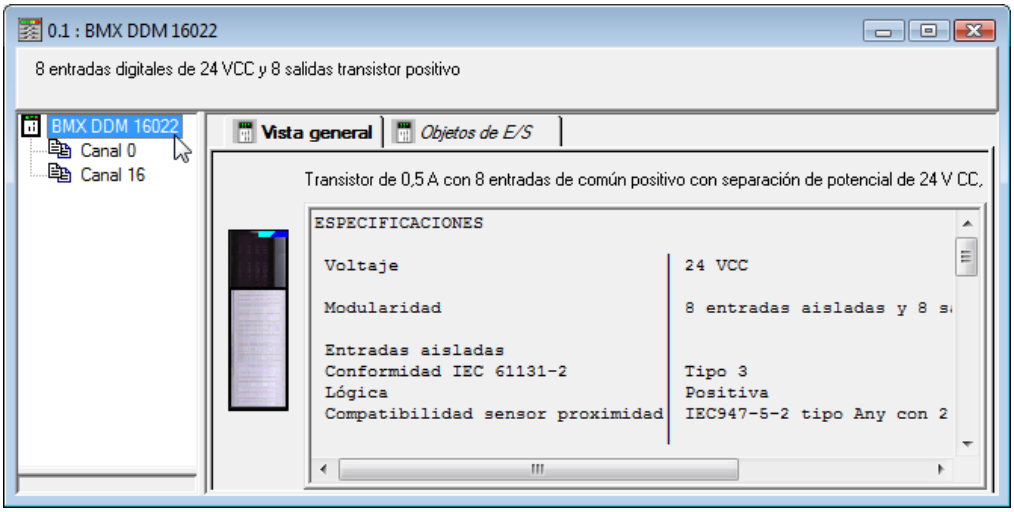


Figura 8. Configuración de un módulo.

Todos los módulos disponen en ventana de su configuración de un árbol que lista los diferentes subsistemas, puerto o canales que lo componen, y una pestaña "Vista general" que resume las especificaciones del módulo. Dependiendo del módulo, y del subsistema seleccionado, se mostrarán otras pestañas de configuración.

Configuración del procesador

La Figura 9 muestra las opciones de la configuración del módulo procesador, tal y como se configuran por defecto al seleccionar una CPU al crear un proyecto. No obstante, se pueden reconfigurar según las necesidades de programación.

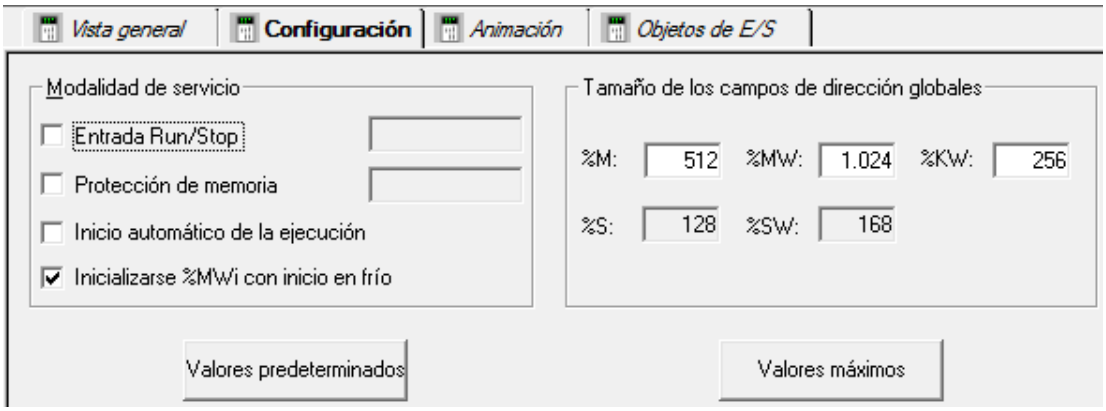


Figura 9. Configuración del procesador.

Configuración de los módulos de E/S digitales

Haciendo un doble clic en un módulo de E/S digitales en la ventana con la configuración del bastidor y seleccionando uno de los canales del módulo, se puede configurar algunos parámetros y ver las variables asociadas a las líneas de entrada y salida.

Como ejemplo, la Figura 10 muestra las ventanas de configuración para los canales 0 y 16 de un módulo BMX DDM 16022. El canal 0 se corresponde con un grupo de 8 entradas digitales, y el canal 16 con un grupo de 8 salidas digitales. Para las entradas digitales se muestra una tabla con los símbolos de programa asociados a las líneas de entrada si estos se

han definido (esto se describe en la sección 3.1), y se puede escoger la tarea que procesa las líneas (por defecto la tarea principal o MAST). Para las líneas de salida, se puede también escoger el Valor de Retorno para cada línea. El valor de retorno es el valor que tomará la línea de salida en el caso de que se produzca una parada anormal del programa.

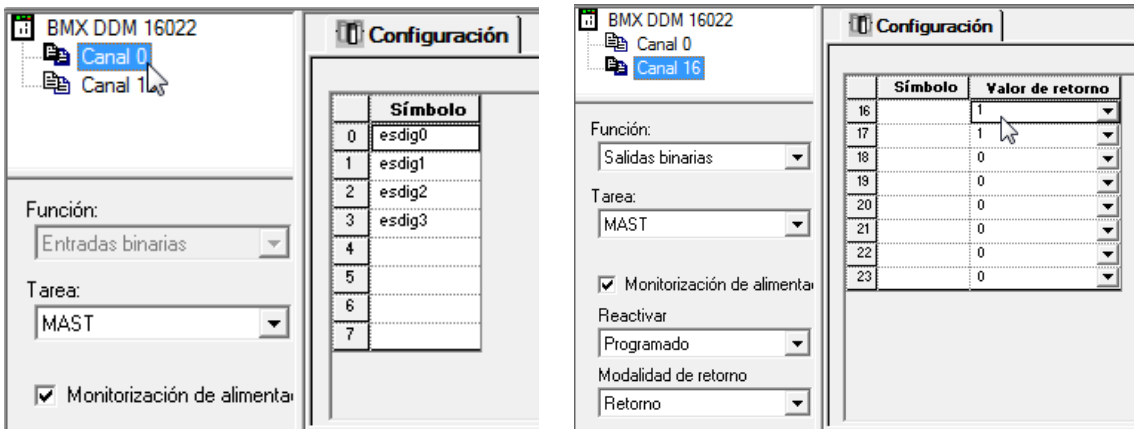


Figura 10. Configuración del E/S digitales.

Configuración de módulos de entradas analógicas

El módulo de entradas analógicas BMX AMM 0600 dispone de seis canales: 4 de entrada y 2 de salida. En un módulo de este tipo, cada convertidor AD (o DA) con una entrada (o salida) analógica se representa con un canal.

La imagen superior izquierda de la Figura 11 muestra la ventana de configuración para las entradas analógicas (canales 0 a 3). Para cada canal de salida se puede configurar si esta activo (utilizado), se puede ver el símbolo con el que se referencia desde el programa si éste está definido, seleccionar el rango de la variable de entrada, configurar un factor de escalado y límites de desbordamiento y activar un filtro. Para configurar el escalado, se abre una ventana como la mostrada en la parte superior derecha de la Figura 11.

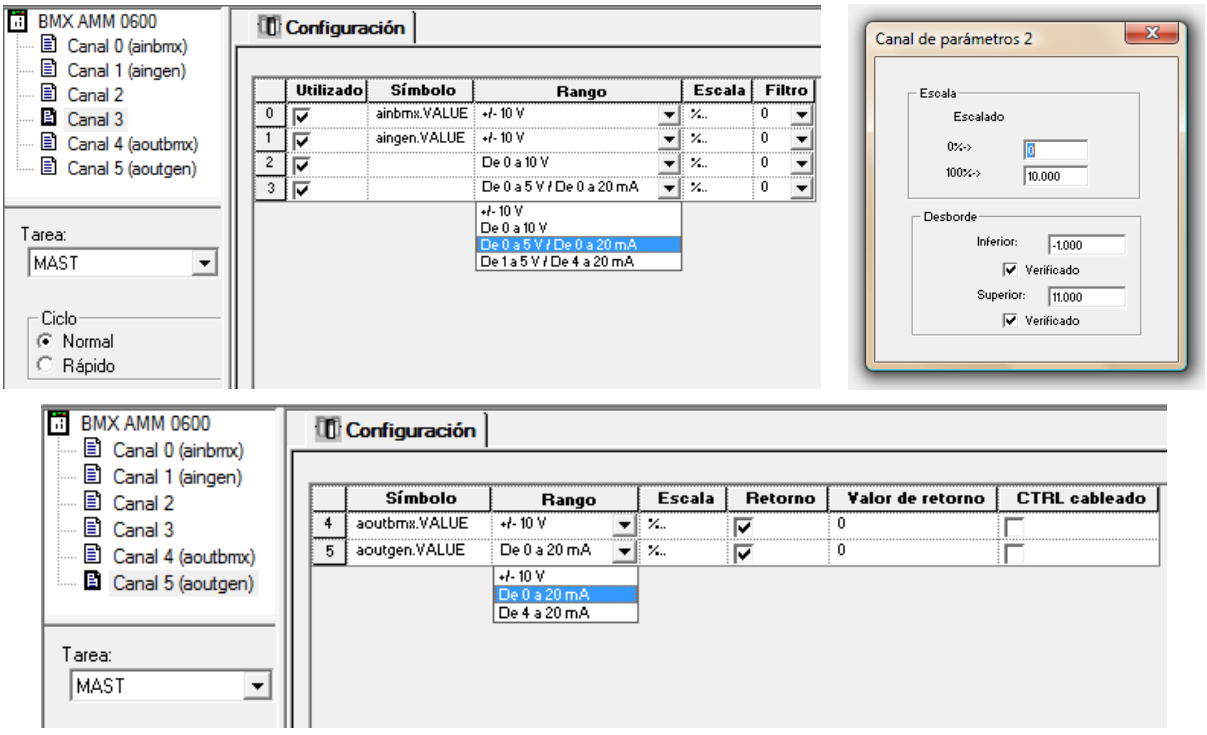


Figura 11. Configuración de entradas analógicas.

Los rangos de la entrada disponibles dependen del módulo. En el ejemplo de la Figura 11, se puede escoger entre medir voltajes o intensidades en varios rangos. El escalado facilita una conversión automática de los valores de una entrada a los valores que serán procesados en el programa al leer esa entrada. Si se configuran los límites de desbordamiento, el módulo generará errores cuando se sobrepasen esos límites. Estos errores pueden tratarse desde el programa.

Finalmente, la opción de filtro permite activar un filtro previo a la conversión AD que trata de eliminar ruidos o interferencias en la señal de entrada. A mayor valor escogido, se realiza un filtrado más fuerte. Pero hay que considerar que con un filtrado más fuerte, el tiempo de respuesta de la entrada disminuye, y por tanto también disminuye la frecuencia a la que se pueden leer la señal de entrada. Si la señal de entrada no tiene ruido o varía rápidamente, conviene escoger cero o valores pequeños.

Para los canales de salida digital (4 y 5), se puede configurar el rango de salida, la escala, el valor de retorno y el control de cableado, como muestra la imagen inferior de la Figura 11. Si el retorno del canal está activado, el valor de la salida analógica correspondiente tomara el valor de retorno indicado cuando ocurra una parada anormal del programa. El control del cableado permite al módulo generar errores cuando se usa una salida por corriente y el cableado tiene un fallo que no permite cerrar el circuito para corriente correctamente.

Configuración de una conexión de red

Para configurar una conexión de red (Ethernet, Modbus), primero se debe crear la conexión de red dentro UnityPro, después hay que realizar la configuración de la lógica de red, y finalmente asociar el módulo con el puerto de red a la conexión de red creada. Estos pasos se realizan de forma sencilla, como se describe a continuación.

Para crear la conexión de red hay que hacer clic con el botón derecho del ratón encima de la carpeta de Redes dentro de Comunicaciones en el Explorador de Proyectos, como muestra la Figura 12. Esto da paso a un dialogo que pregunta el tipo de red a crear (Ethernet, Modbus+...). Se debe escoger el tipo de red deseado y asignarle un nombre.

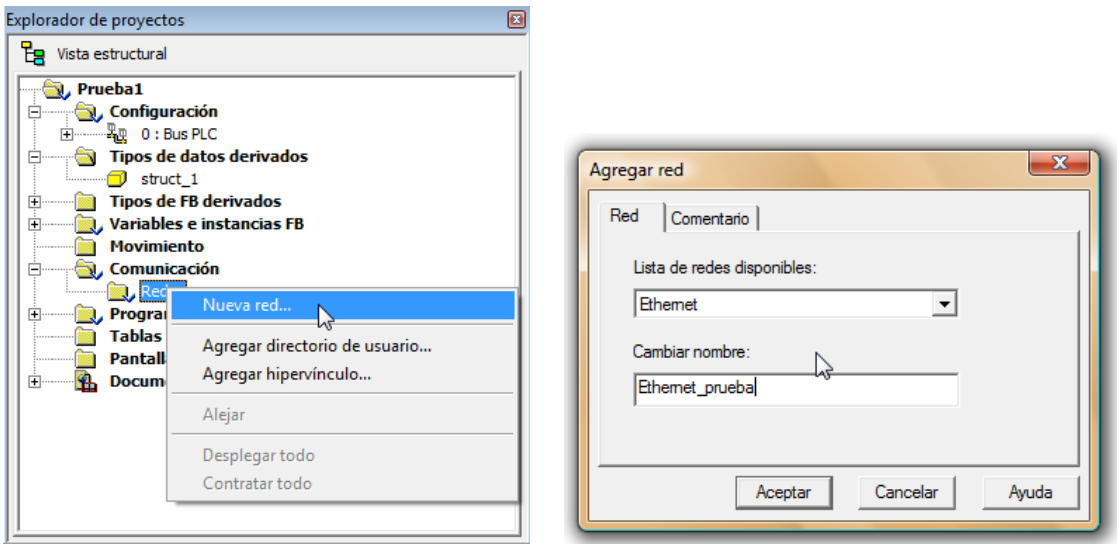


Figura 12. Creación de una conexión de red.

A continuación se debe activar y configurar la red creada. Para ello se puede hacer doble clic sobre la red que ahora aparece dentro del apartado Comunicaciones-Redes del Explorador de Proyectos, como muestra la Figura 13.

De este modo se accede a una ventana que permite configurar las propiedades de la red. Primero hay que escoger el tipo de módulo de red que se emplea. Para las prácticas, se debe usar el tipo CPU 2020 1.00, que se corresponde con el puerto Ethernet de una CPU BMX 34 2020. Después se puede escoger si se emplea una dirección IP configurada estáticamente, o la dirección es asignada desde un servidor de la red. En caso de una dirección configurada, hay que indicar los clásicos valores de dirección, máscara y puerta de enlace.

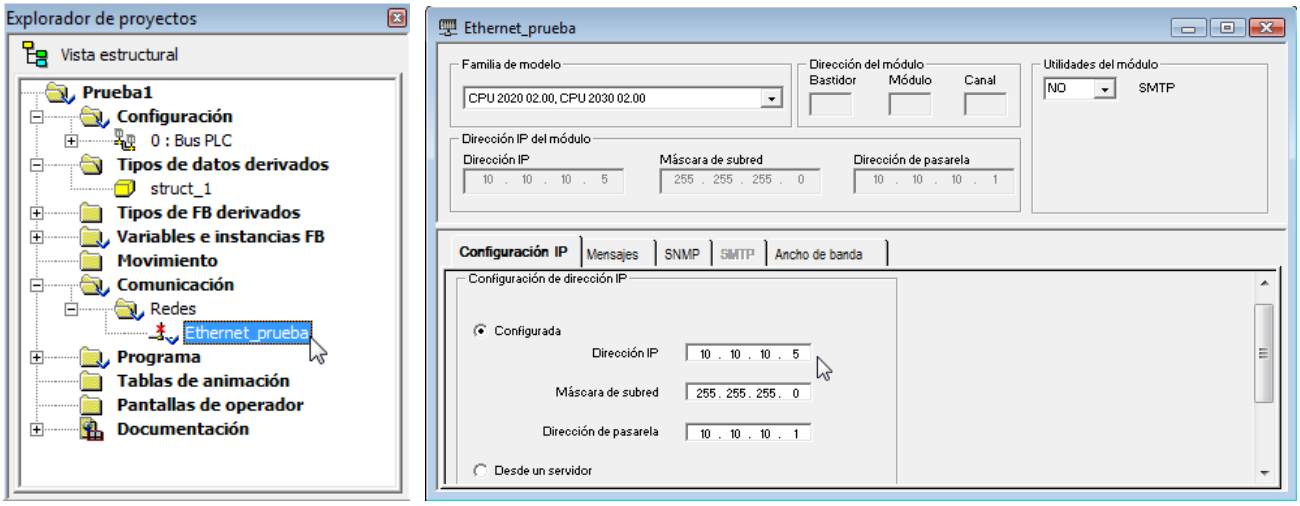


Figura 13. Configuración de una conexión de red.

Antes de realizar el siguiente paso de asociación, hay que asegurarse de que el módulo con el puerto de red ha sido colocado en un bastidor, según lo descrito en el apartado 2.2. En el caso de un puerto de red incorporado en la CPU, bastará con haber escogido el módulo de CPU correcto.

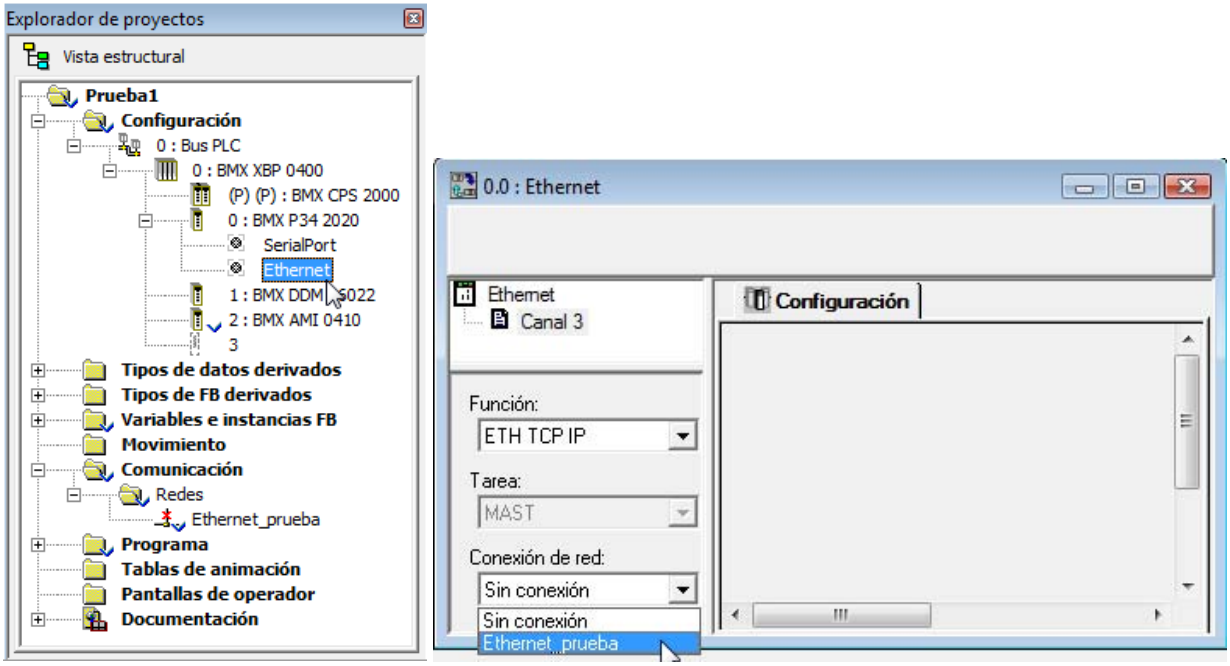


Figura 14. Asociación del puerto de red a una conexión de red.

Finalmente, para asociar la conexión de red al módulo de red, hay que desplegar las entradas del Bus PLC en la carpeta de Configuración del Explorador de Proyectos, con el objetivo de acceder al puerto de red deseado. El puerto estará asociado al módulo del bastidor que tiene ese puerto de red. Por ejemplo, en la Figura 14 se puede ver como se ha seleccionado el puerto Ethernet del módulo de CPU CPU BMX 34 2020, dentro del

Explorador de Proyectos. Haciendo doble clic sobre ese puerto, se tendrá acceso a su configuración. En dicha configuración (Figura 14) simplemente hay que seleccionar el canal (3) y después asociarlo a la conexión de red previamente creada (Ethernet_prueba).

Tras este proceso, el puerto de comunicaciones ya se puede utilizar dentro de los programas a través de los bloques de función adecuados.

2.3. Propiedades del proyecto

Se accede a estas propiedades haciendo clic con el botón derecho del ratón sobre la carpeta raíz del Explorar de Proyectos. Esta carpeta tiene el nombre asignado al proyecto. Se puede establecer:

- Pestaña General: define el nombre del proyecto (por defecto es Estación)
- Pestaña de protección: activa la protección de las secciones del programa mediante contraseña.
- Pestaña de identificación: permite identificar el proyecto con un número de versión.
- Pestaña de comentario: Asocia un comentario al proyecto.

3. Programación básica

3.1. Variables y tipos

Una variable es una entidad de memoria cuyos contenidos pueden ser modificados por el programa durante la ejecución. Hay que tener en cuenta que los datos y los registros de control de módulos de entradas y salidas también se tratan como variables. Las variables se identifican por su nombre, que está formado por una cadena de hasta 32 caracteres.

UnityPro admite dos tipos de variables: asignadas y no asignadas (también conocidas como *allocated* o *not allocated*). Una variable asignada está asociada a un módulo de E/S o a una referencia de memoria, mientras que una variable no asignada no está asociada a ninguna E/S ni a ninguna referencia de memoria en concreto, es decir, no es posible conocer la posición en la memoria de la variable. Por ejemplo, si una variable llamada “presión_agua” se asocia explícitamente con la palabra en la dirección de memoria %MW102, se puede decir que es una variable asignada. Las variables no asignadas deben ser declaradas dentro del Editor de Variables de Datos, o durante la programación mediante la opción de asignación de entradas y salidas, indicando explícitamente su tipo.

Para cada variable con la que se trabaja hay que indicar el tipo de datos que almacena. Unity Pro admite dos clases de de tipos de datos para las variables: los Tipos Elementales de Datos o EDT, y los Tipos Derivados de Datos o DDT. Los EDT de uso más habitual son los siguientes:

Tipo	**Rango**	**Descripción**
BOOL	FALSE / TRUE ó 0 - 1	Valor lógico booleano o valor de un solo bit
BYTE	0 - 255	Dato que ocupa 8 bits
WORD	0 - 65.535	Dato que ocupa 16 bits.
INT	-32.768 - 32.767	Número entero con signo.

Tipo	Rango	Descripción
DINT	2.147.483.648 - 2.147.483.647	Número entero con signo.
UINT	0 - 65.535	Número entero sin signo.
UDINT	0 - 4.294.967.295	Número entero sin signo.
REAL	8,43E-37 - 3,36E+38	Número real o de coma flotante (32 bits).
EBOOL	0 - 1	Extensión del tipo BOOL que además de almacenar el valor binario 0 ó 1, permite detectar transiciones de 0 a 1 (flanco ascendente) y de 1 a 0 (flanco descendente), así como forzar el valor de una entrada. Es el adecuado para E/S digitales.
STRING	0 – 65.334 caracteres ASCII en el rango 32 a 255	Define una cadena de caracteres ASCII de 8 bits. Por defecto tiene 16 caracteres de longitud, más el carácter indicador de final de cadena.
TIME	0 - 4.294.967.295 ms (49 días)	Valor UDINT que representa una duración en milisegundos. Las unidades de tiempo permitidas para representar el valor son: días (D), horas (H), minutos (M), segundos (S) y milisegundos (MS).

Los DDT son tipos de datos más complejos, que definen estructuras y matrices. Así, una variable que se declare con un tipo DDT contendrá un conjunto de valores. Mientras que una estructura permite definir un conjunto de valores de tipos diferentes, las matrices contienen múltiples valores del mismo tipo ordenados por índices.

Una clase especial de DDT son los tipos IODDT, que se usan para gestionar E/S complejas. Por ejemplo, para un canal de entrada analógico, se puede usar el tipo IODDT T_ANA_IN_BMX, de forma que una variable de este tipo engloba varios valores de configuración e indicadores de estado del canal, además del propio valor leído de la entrada.

A las variables se les puede asignar valores en sus rangos, y estos valores se pueden indicar en decimal o en las bases 2 (binario), 8 (octal) ó 16 (hexadecimal). Por ejemplo, el valor decimal 65.535 también se puede indicar como los valores 2#1111111111111111, 8#177777 ó 16#FFFF.

Finalmente cabe mencionar que también se pueden definir constantes, que son variables de un tipo EDT asignadas en direcciones directas, según se comenta en el siguiente apartado. Sus contenidos no pueden modificarse por el programa durante la ejecución, y únicamente pueden ser leídos.

3.2. Direccionamiento directo

Además de por su nombre, se puede acceder a las variables conociendo su posición de memoria, esto es, su dirección. En el caso de una variable de E/S se requiere además conocer la localización del módulo donde se encuentra. A este método se le denomina direccionamiento directo, y es útil cuando se quiere acceder a entradas, salidas o registros específicos, o cuando se quiere definir una variable asignada.

La dirección de de una variable comienza con el símbolo %, y seguidamente especifica, en este orden la clase de variable (I: entrada, Q: salida, K: constante, M: Memoria...), el tipo de la variable (X: booleano, W: palabra 16 bits, D palabra doble de 23 bits...) y su localización.

Para variables de memoria o constate, la localización es simplemente un índice de posición. Por ejemplo, %MX1 se refiere al primer bit de la memoria de datos, %MW4 se refiere a la cuarta palabra, y %KW100 a la centésima palabra constante.

Para las variables de entrada y salida digitales, la localización debe especificar además los números de bastidor, de módulo y de bit. Por ejemplo, la variable %IX0.31.4 se refiere a la línea de entrada booleana en el bastidor 0, módulo 1, línea 4. En el caso de querer acceder al valor de un puerto de entrada o salida analógica habría que usar expresiones como %IW0.2.0 o %OW.0.2.4, para el bastidor 0, módulo 2 y canales 0 y 4 (0 es de entrada y 4 es de salida).

Es evidente que resulta más fácil y claro trabajar con nombres de variables que con direcciones, sobre todo cuando se tiene poca experiencia con la familia de módulos de Modicom, por lo que conviene asignar variables a las zonas de memoria que se usen frecuentemente en un programa.

3.3. Gestión de variables

Toda la declaración y mantenimiento de variables, tipos e incluso bloques de funciones, se puede realizar a través del Editor de Datos, al que se accede a través de la carpeta Variables e Instancias FB del Explorador de Proyectos (ver la Figura 15). Para ello basta con hacer doble clic en la carpeta Variables e Instancias FB, o en alguna de sus secciones (Variables elementales, Variables derivadas, Variables E/S derivadas...). Mientras que, si se pulsa sobre Variables e Instancias FB, el Explorador de Proyectos mostrará la lista de todas las variables, pulsando en una opción concreta se mostrarán solo las variables de la clase seleccionada.

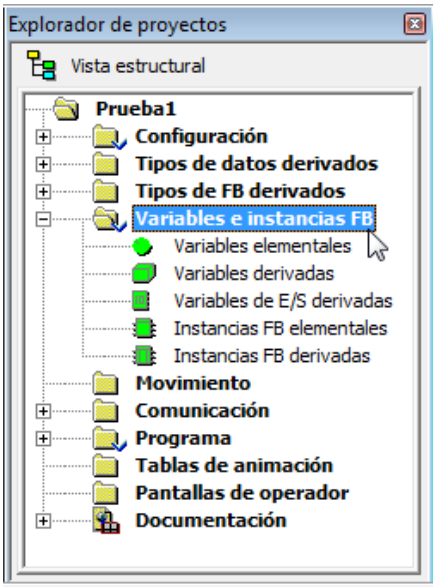
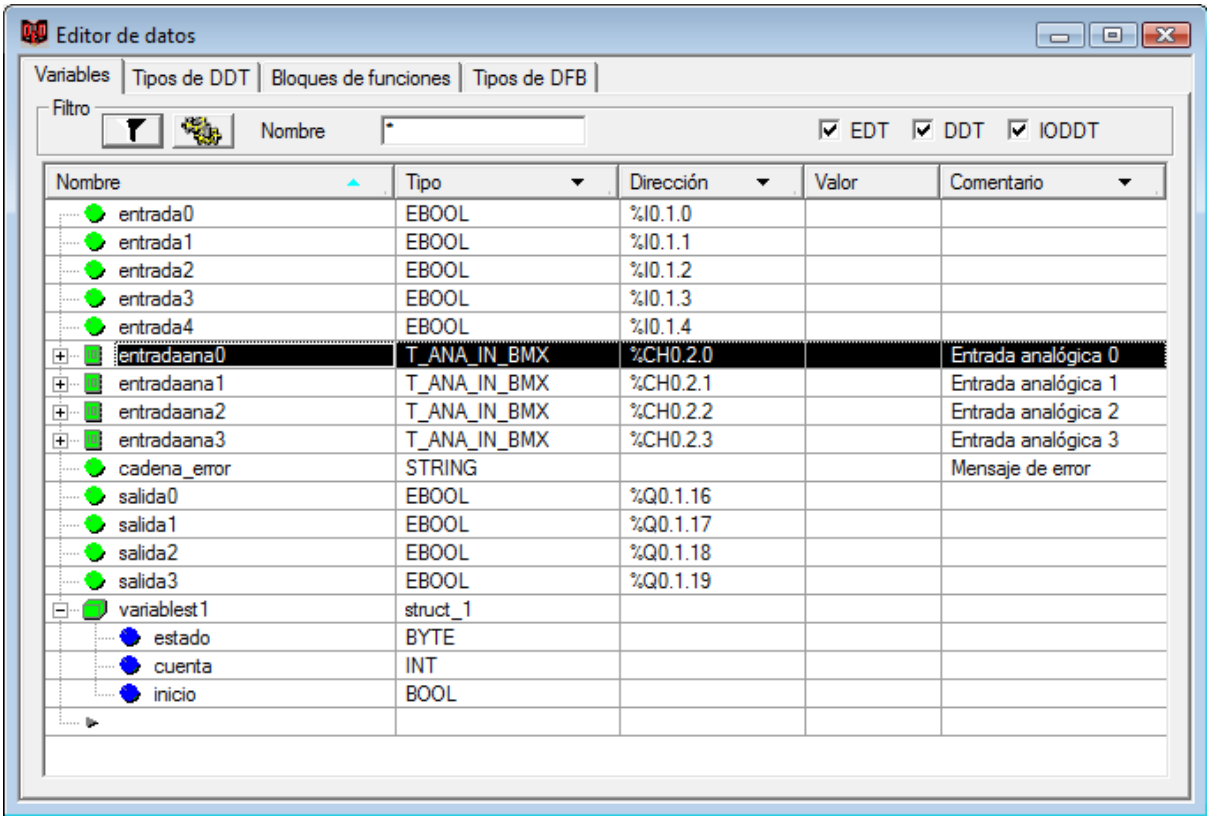


Figura 15. Acceso las listas de variables.

La Figura 16 Muestra el aspecto del Editor de Datos con una lista de variables de diferentes tipos ya creadas. Se pueden crear nuevas variables añadiendo un nuevo nombre al final de la lista, o insertando nuevas variables en una parte específica de la tabla mediante el botón derecho del ratón. También se pueden editar los nombres de las variables existentes haciendo clic sobre ellos. Para cada variable se especifica su tipo, y si es una variable asignada, también su dirección. Las variables sin dirección son variables no asignadas. Además se puede escribir un comentario para cada variable.

Como se aprecia en la Figura 16, el Editor de Datos ofrece varias opciones para filtrar las variables que se desean ver en la lista. Si en la lista no aparece ninguna variable aunque estas existan, posiblemente se deba a que hay un filtro activo que no es el adecuado. Además, el

Editor de Datos permite listar los tipos de datos derivados (DDT), los bloques de funciones y los tipos de bloques derivados (DFB) creados por el usuario.



Nombre	Tipo	Dirección	Valor	Comentario
entrada0	EBOOL	%I0.1.0		
entrada1	EBOOL	%I0.1.1		
entrada2	EBOOL	%I0.1.2		
entrada3	EBOOL	%I0.1.3		
entrada4	EBOOL	%I0.1.4		
entradaana0	T_ANA_IN_BMX	%CH0.2.0		Entrada analógica 0
entradaana1	T_ANA_IN_BMX	%CH0.2.1		Entrada analógica 1
entradaana2	T_ANA_IN_BMX	%CH0.2.2		Entrada analógica 2
entradaana3	T_ANA_IN_BMX	%CH0.2.3		Entrada analógica 3
cadena_error	STRING			Mensaje de error
salida0	EBOOL	%Q0.1.16		
salida1	EBOOL	%Q0.1.17		
salida2	EBOOL	%Q0.1.18		
salida3	EBOOL	%Q0.1.19		
variablest1	struct_1			
estado	BYTE			
cuenta	INT			
inicio	BOOL			

Figura 16. Lista de variables en el Editor de Datos

Hay una alternativa para crear variables asignadas para las E/S a través de las opciones de configuración de los módulos que forman el bastidor del PLC, que no requiere conocer las direcciones. Primero se debe acceder a la ventana con la configuración del bastidor del PLC haciendo doble clic en la carpeta Configuración-busPLC del Explorador de Proyectos. Desde esta ventana, se puede acceder a la ventana de configuración de un módulo de E/S con sólo hacer doble clic en él. Dentro de la configuración del módulo, hay que activar la pestaña Objetos de E/S.

Por ejemplo, para un módulo de E/S digitales BMX DDM16022 se obtiene la ventana de configuración mostrada en la Figura 17, con la pestaña Objetos de E/S en primer plano. Dentro de esa pestaña se puede acceder a todas las direcciones de memoria del módulo, para lo que se debe escoger en la sección Objetos de E/S el tipo de direccione que se desea ver, y después pulsar el botón Actualizar Cuadricula. Así, si se selecciona el tipo %I para ver las entradas, y se pulsa Actualizar Cuadricula, aparecerá la lista con las 8 entradas digitales %I0.3.0 a %I0.3.7.

Se puede seleccionar un rango de entradas con el ratón, como muestra la Figura 18, y después usar la sección Creación de variable E/S para asignar automáticamente nuevas variables a las entradas seleccionadas. El resultado se puede ver en la Figura 19. Las nuevas variables aparecerán en el Editor de Datos (ver apartado 3.3).

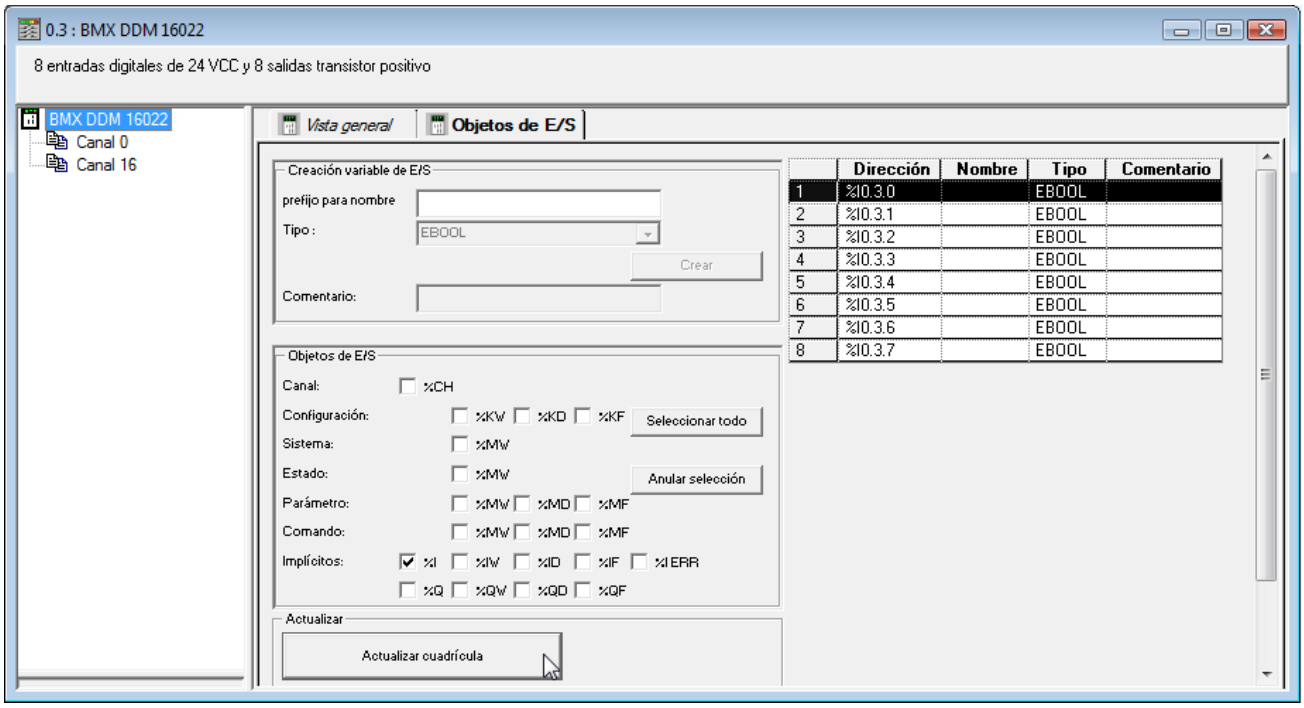


Figura 17. Configuración del módulo de E/S digitales.

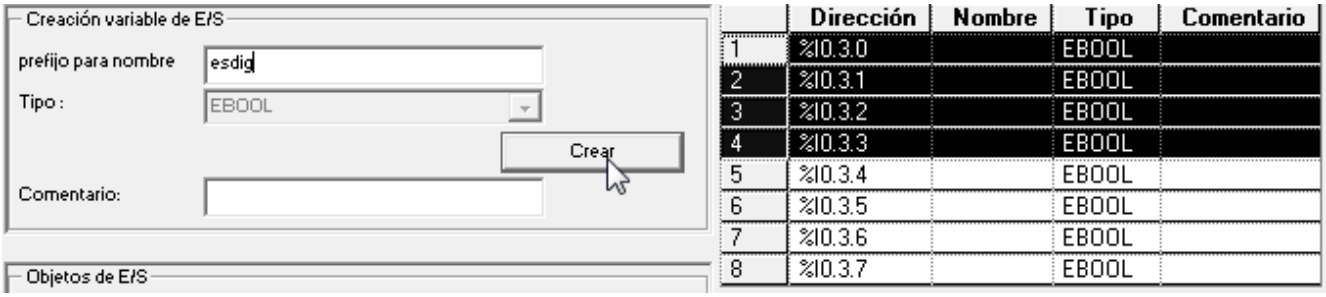


Figura 18. Creación de variables para un rango de entradas digitales.

	Dirección	Nombre	Tipo	Comentario
1	%I0.3.0	esdig0	EBOOL	
2	%I0.3.1	esdig1	EBOOL	
3	%I0.3.2	esdig2	EBOOL	
4	%I0.3.3	esdig3	EBOOL	
5	%I0.3.4		EBOOL	
6	%I0.3.5		EBOOL	
7	%I0.3.6		EBOOL	
8	%I0.3.7		EBOOL	

Figura 19. Variables creadas para un rango de entradas digitales.

3.4. Estructura del programa

El programa del PLC se estructura en tareas, bien de modo simple o con multitarea, en secciones y subrutinas, y en módulos funcionales.

Cuando se trabaja en modo simple, se dispone de la tarea principal (MAST), que se descompone en secciones de código y subrutinas, que pueden estar escritos en diferentes lenguajes. Dependiendo del modelo de CPU, en modo multitarea, además de la tarea MAST,

pueden estar disponibles una tarea rápida (FAST), tareas de eventos y tareas auxiliares (AUX). Como su nombre indica, la tarea FAST es adecuada para un bloque de operaciones que se ejecuta rápidamente en el programa. Las tareas de eventos permiten reducir el tiempo de respuesta del programa de aplicación a los eventos desde módulos de entrada-salida y sucesos temporizados.

La Figura 20 muestra un ejemplo de ejecución multitarea. Los intervalos I, P y O se refieren a las etapas de lectura de entradas (I), procesamiento (P), y escritura de salidas (O) de cada ciclo de ejecución. Se observa como una tarea de evento es la más prioritaria, seguida de la FAST y después la MAST. Las tareas AUX son de menos prioridad que las MAST.

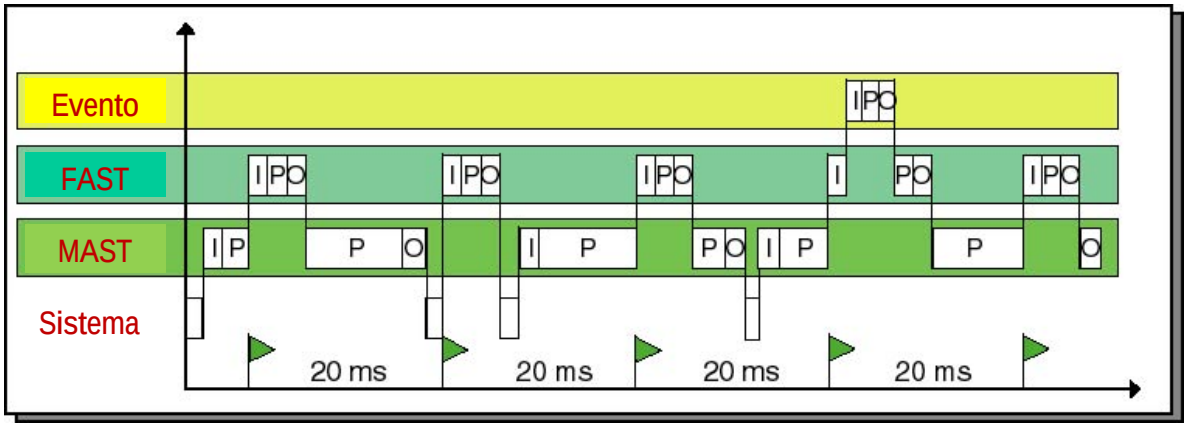


Figura 20. Ejemplo de funcionamiento en modo multitarea.

Todas las tareas se pueden programar con los lenguajes LD, FBD, IL, ST (ver la sección 0). Además, para la tarea MAST se puede usar el lenguaje SFC.

Es posible cambiar la forma en que la CPU realizará el ciclo de ejecución de una tarea (lectura de entradas – procesamiento – escritura de salidas) del programa. Se puede escoger entre el modo cíclico o normal, o el modo periódico, fijando en este caso el periodo deseado. Con el modo periódico se fuerza a la CPU a completar el ciclo cada cierto tiempo (periodo). Por ejemplo, para cambiar el modo de ejecución de la tarea MAST, hay que hacer un clic derecho sobre MAST dentro de la opción Programa-Tareas del Explorador de Proyectos, y seleccionar la opción Propiedades, como muestra la Figura 21.

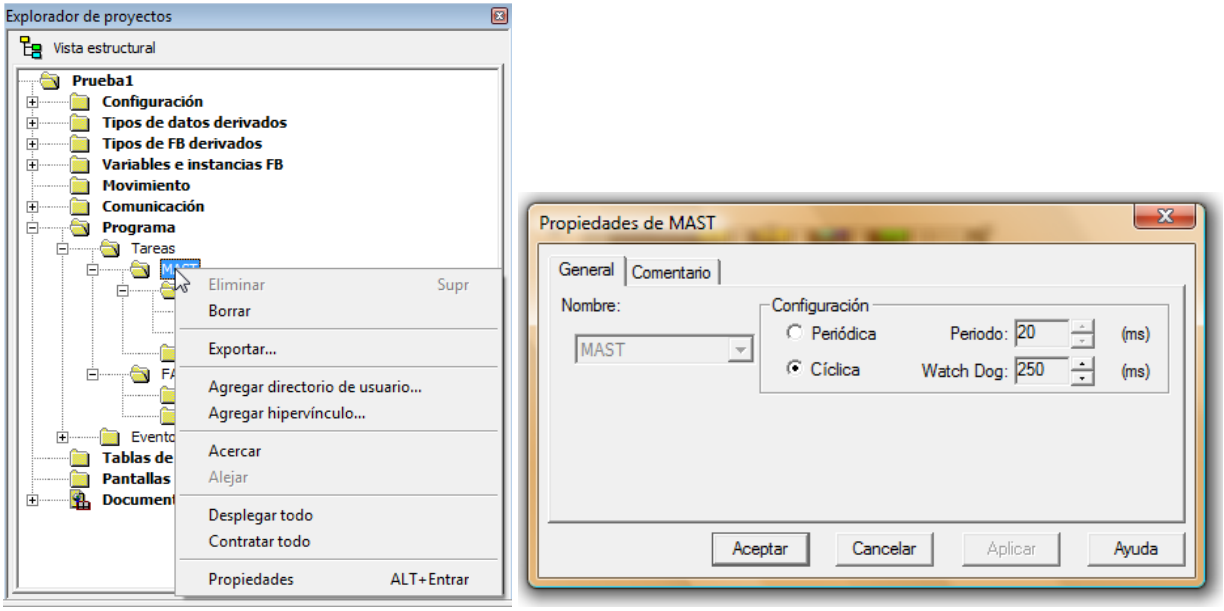


Figura 21. Acceso a las propiedades de a tarea MAST.

También se puede establecer el tiempo de WatchDog para las tareas MAST y FAST, que deberá ser superior al valor del período si se usa un ciclo periódico. El WatchDog es un monitor del sistema que vigila si un ciclo de la tarea dura más tiempo del especificado, y en tal caso fuerza su finalización, para impedir que la CPU quede bloqueada.

3.5. Creación de secciones de programa y rutinas

Cada tarea está compuesta de secciones de programa y de subrutinas (o secciones SR). Las subrutinas se programan como entidades separadas y son llamadas desde las secciones o desde otras subrutinas dentro de la misma tarea. El número de anidamientos está limitado a 8, y una subrutina no puede llamarse a sí misma (el programa no es recursivo).

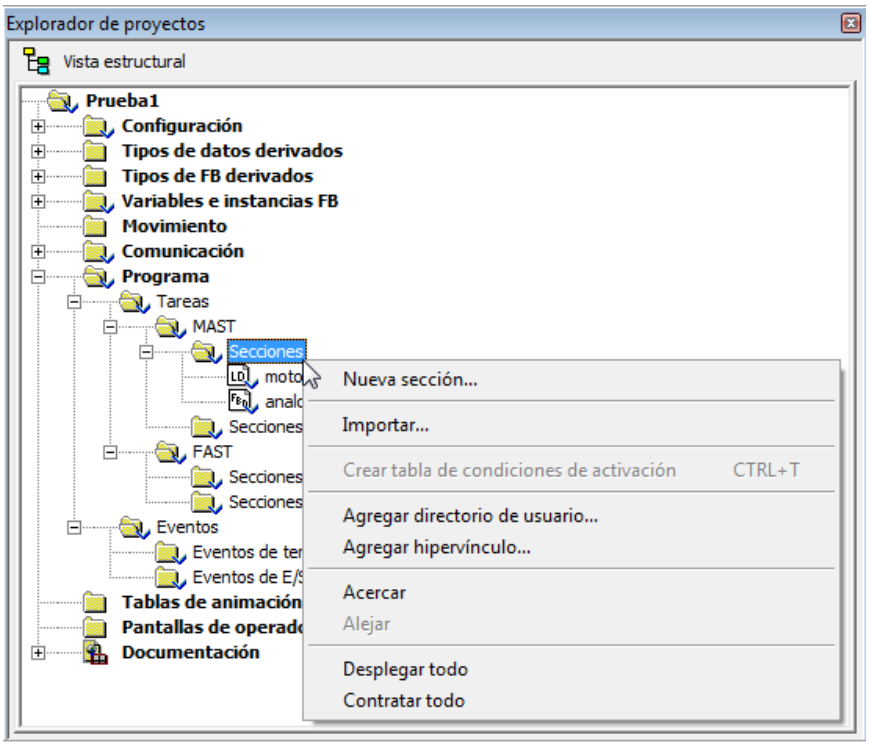


Figura 22. Creación de una sección de programa para una tarea.

Para crear una nueva sección o subrutina basta con seleccionar carpeta Sección de la tarea que interese dentro del Explorador del Proyectos, y después hacer clic con el botón derecho del ratón encima (ver Figura 22). En el menú emergente se escoge Nueva Sección. Así se accede a una venta donde se puede especificar el nombre de la sección, y el tipo de lenguaje en que se desea programar, como muestra la Figura 23.

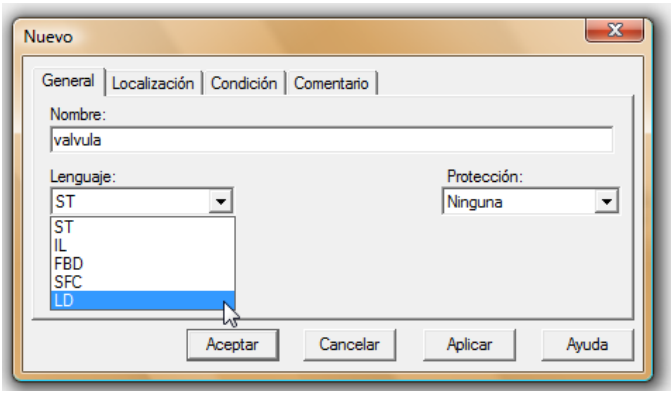


Figura 23. Selección del leguaje para programar una nueva sección.

También se puede exportar e importar el código de una sección a través del menú de contexto que aparece al hacer clic con el botón derecho del ratón encima de una sección dentro del Explorador del Proyectos (ver Figura 22).

Cabe mencionar que el orden de ejecución de las secciones de una tarea está definido por el orden en el que aparecen las secciones en el Explorador del Proyectos.

3.6. Bloques de funciones

UnityPro ofrece una biblioteca con un gran número de bloques de funciones avanzadas que resuelven muchas tareas comunes, lo que facilita la programación. Estos son los denominados Bloques de Funciones Elementales o EFB, que pueden ser usados desde los diferentes lenguajes. Dentro de la ayuda que ofrece el entorno de UnityPro, hay un apartado donde se detallan todos los bloques disponibles en la biblioteca.

Además, un usuario puede crear sus propios bloques de funciones para abordar operaciones que se le plantean regularmente, de forma que no necesite reescribir el programa de estas operaciones cada vez que las tenga que usar. Para ello, primero se tiene que definir un nuevo tipo de Bloque de Funciones Derivado o DFB.

Posteriormente, al programar, se definen bloques de funciones que son instancias de los EFB o DFB de la biblioteca, y que se pueden crear desde el programa de forma que automáticamente quedan escritas en Editor de Datos. Pero, también se pueden crear estas instancias desde el Editor de Datos, en la pestaña Bloques de Funciones, escribiendo el nombre y seleccionando el tipo de EFB o DFB.

La sección 5 se centra en cómo se definen y utilizan los bloques de funciones.

3.7. Programación con el lenguaje de diagrama de contactos (LD)

Con el lenguaje de diagrama de contactos (LD), las secciones de programa de las tareas se representan mediante circuitos eléctricos que consisten básicamente de dos tipos de objetos: contactos, bien normalmente abiertos o bien normalmente cerrados, y bobinas. Cada sección LD consiste en una única página con una cuadricula, divida en filas y columnas, donde se colocan los objetos. La Figura 24muestra un ejemplo de sección LD.

En analogía con un esquema eléctrico, se considera que la línea vertical localizada a la izquierda del editor LD es la línea de potencia de la alimentación, y todos los elementos conectados con esa línea deben ser procesados. La línea vertical de la derecha se corresponde con el cable neutro de la alimentación. Cada grupo de objetos que se comunican entre sí y no tiene comunicación con otros objetos se denomina red o escalón.

La secuencia de proceso de los objetos individuales en una sección LD está determinada por el orden en que están colocados esos objetos. Las redes conectadas a la línea izquierda de potencia se procesan de arriba hasta abajo. Dentro de una misma red, las diferentes líneas también se evalúan de arriba abajo.

Cada contacto está asociado a una condición booleana que puede ser una variable (sección 3.1) una dirección de memoria (ver sección 3.2), o una expresión matemática que devuelva un resultado booleano. Así, las variables y direcciones de entradas y salidas digitales son aplicables a los contactos. En el caso de los contactos normalmente abiertos, estos se considerarán cerrados cuando la variable o expresión es cierta. Con los contactos tachados, estos se considerarán cerrados cuando la variable o expresión sea falsa. Adicionalmente, se dispone de contactos que se activan por transiciones de flanco ascendente o descendente en su variable o expresión.

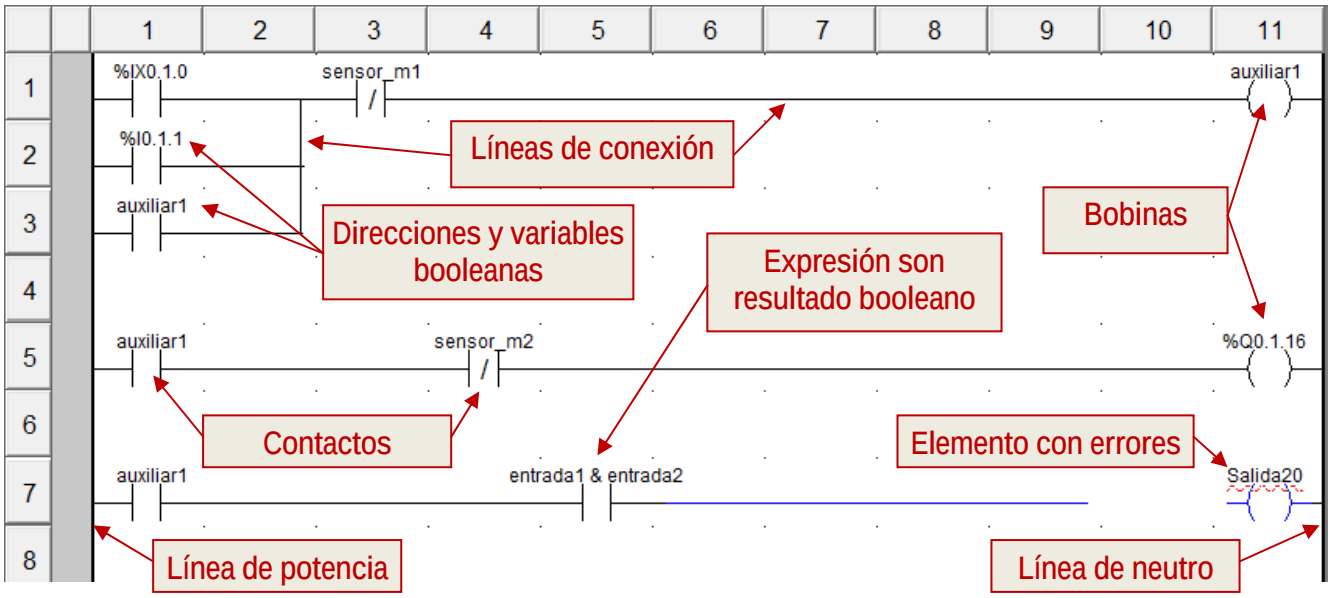


Figura 24. Ejemplo se página de LD con tres redes.

Las bobinas tienen que estar asociadas a variables o direcciones de memoria booleanas, y tomarán el valor TRUE o 1 cuando el circuito de la red donde se encuentran quede cerrado. Hay también bobinas especiales que permiten llamar a subrutinas o parar la ejecución de la sección, así como bobinas negadas que están estarán activas (TRUE) cuando el circuito de su red está abierto.

Como muestra la Figura 24, en UnityPro, cuando un objeto tiene algún error, este se marca en azul. Si una variable o expresión es incorrecta, esta se subraya en rojo. Si se pasa el ratón sobre un objeto con algún error, aparecerá un cuadro que explica las causas del error.

Además de contactos y bobinas, las secciones LD pueden contener también bloques de funciones, como muestra la Figura 25. En esta figura también se observa cómo se pueden añadir bloques de comentarios de texto, para describir los circuitos.

Para dibujar objetos en una sección LD, hay disponible una barra de herramientas con botones correspondientes a los diferentes objetos que se pueden usar, tal y como muestra la Figura 26. Esta barra se muestra habitualmente al abrir o seleccionar una ventada de una sección LD, pero si no se muestra se puede habilitar situando el ratón en la barra de herramientas y pulsando su botón derecho.

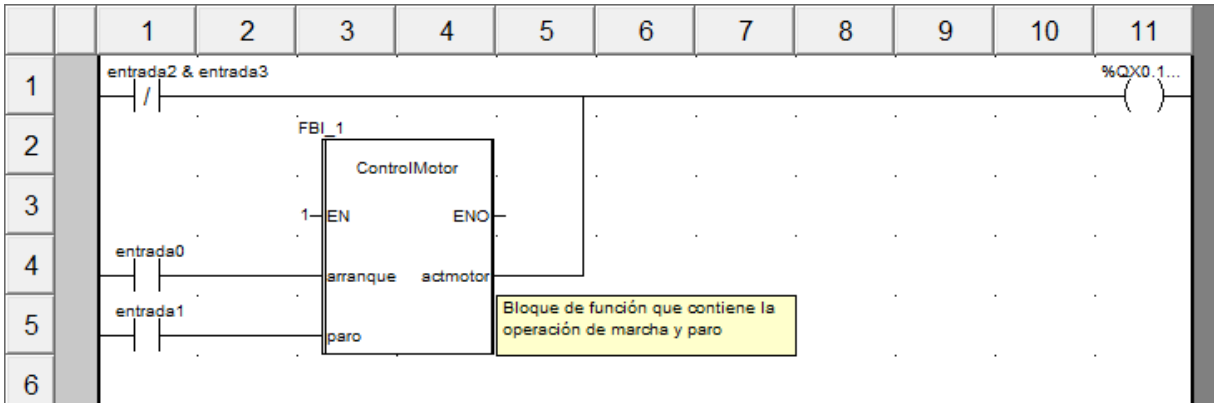


Figura 25. Página de LD que contiene un bloque de función.

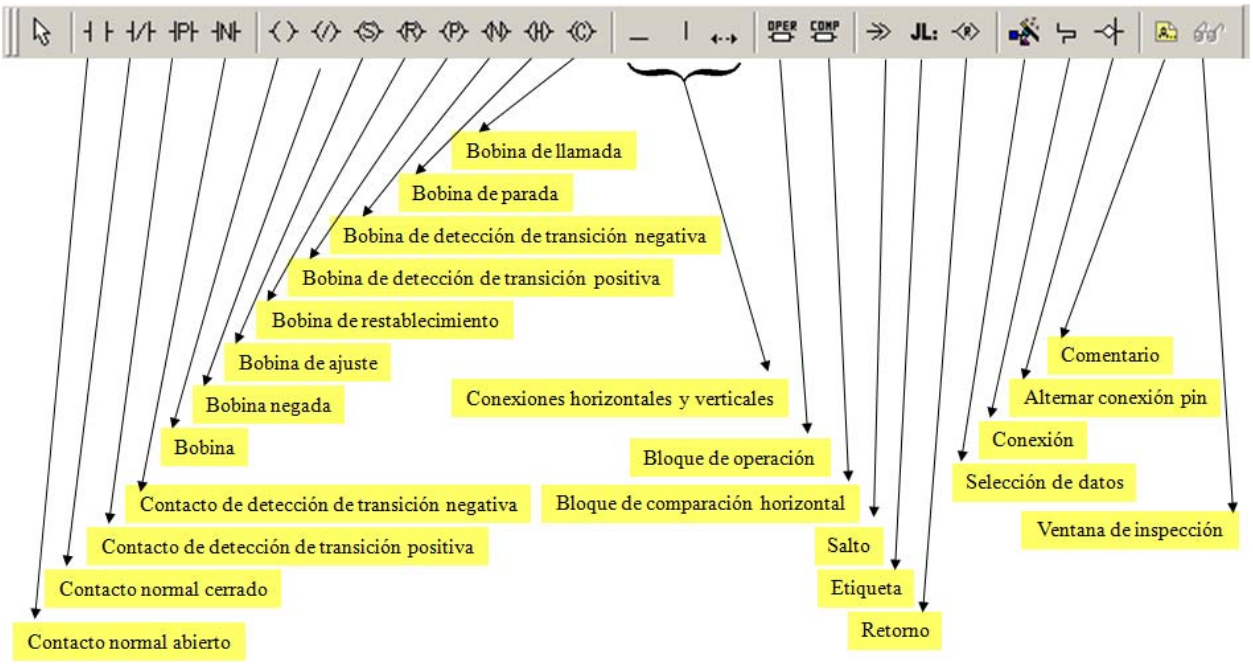


Figura 26. Botones de la barra de herramientas para creación de secciones LD.

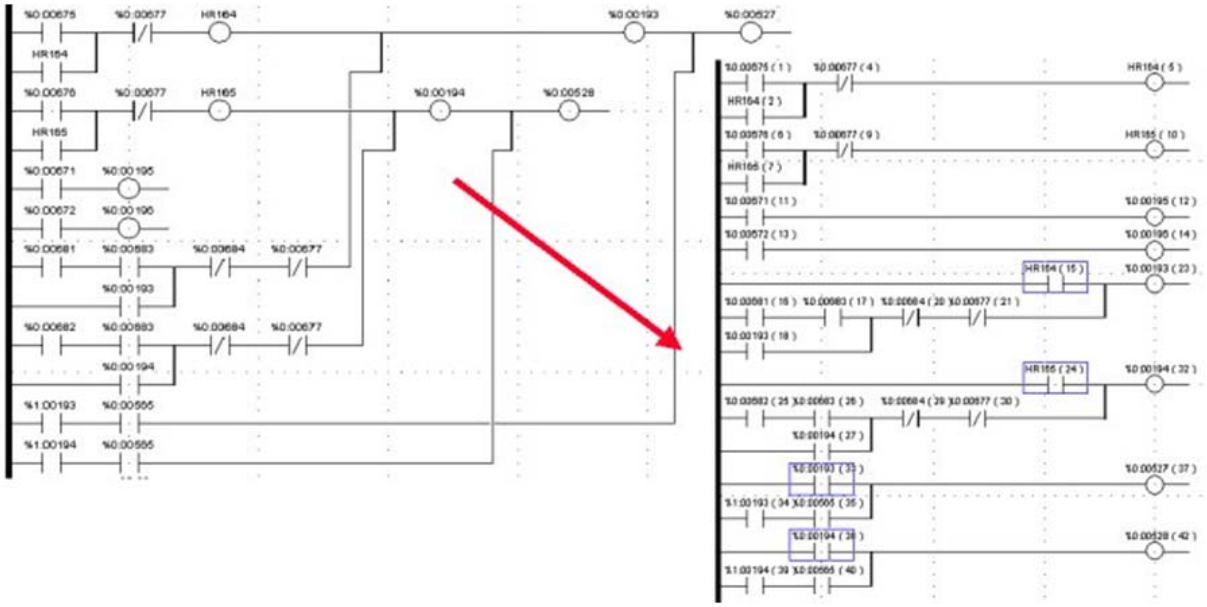


Figura 27. Para mayor claridad del diagrama hay que evitar los cruces de líneas.

Finalmente, cabe comentar que conviene ser ordenado al dibujar diagramas complejos para que el funcionamiento sea comprensible y se puedan realizar fácilmente modificaciones. En concreto, es conveniente diseñar los diagramas de forma que se eviten cruces en las líneas, como muestra la Figura 27. En los casos en los que no sea posible evitar los cruces simplemente moviendo los objetos, se puede recurrir a variables auxiliares o a duplicar las bobinas de salida.

3.8. Generar y validar un programa

Cuando se tiene un proyecto completado, o una parte del mismo que puede funcionar, primero se debe analizar el proyecto para comprobar que no tiene errores, y después se debe generar la aplicación final. UnityPro ofrece tres opciones para estas tareas:

- Analizar proyecto. Analiza un proyecto en busca de errores y conflictos. Si hay errores, la lista con los mismos se muestra en la ventana de estado y resultados, situada normalmente en la parte inferior de la aplicación, dentro de su pestaña Analizar Proyecto. Si el proyecto es correcto, en la pestaña Analizar Proyecto aparece un mensaje confirmándolo. La Figura 28 muestra ejemplos de estos dos casos.
- Regenerar todo el proyecto. Genera la aplicación final, que queda lista para ser cargada al PLC, conforme se describe en la sección siguiente. Si el proyecto es complejo, UnityPro puede tardar algún tiempo en generar la aplicación. En la pestaña Analizar Proyecto de la ventana de estado y resultados se muestra si el proyecto se ha generado correctamente, o se ha encontrado algún error que no fue detectado en al analizar el proyecto, y que debe ser corregido.
- Generar cambios. Genera el proyecto final, compilando solamente las últimas modificaciones realizadas con respecto a la vez anterior que se genero el proyecto. De esta forma, el proceso de generar el proyecto es más rápido.

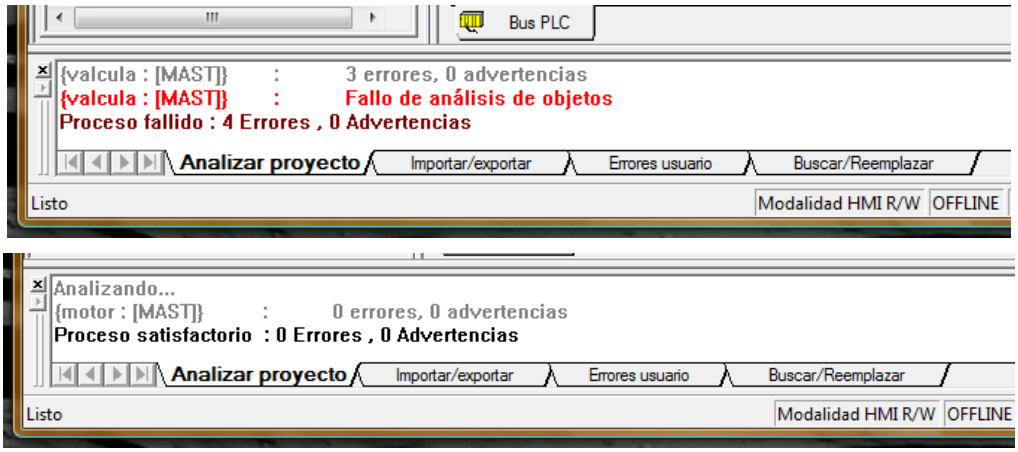


Figura 28. Botones de UnityPro para analizar y generar un proyecto.

Para acceder a las tres opciones se puede usar el menú Generar, o los botones correspondientes de la barra de herramientas, mostrados en la Figura 29.

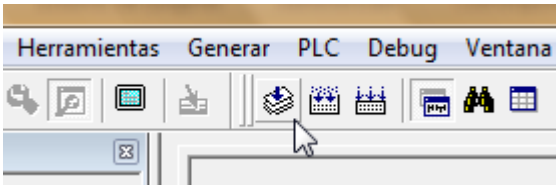


Figura 29. Botones de UnityPro para analizar y generar un proyecto.

Finalmente, cabe mencionar que las opciones de generación de un proyecto se deshabilitan si el UnityPro está conectado al PLC. Los aspectos sobre la conexión al PLC se describen en la siguiente sección.

4. Ejecución de la aplicación en el PLC

4.1. Carga de programas en el simulador y simulación

Después de crear un proyecto, este debe enviarse al PLC para ser ejecutado en el último. Pero conviene utilizar un simulador del PLC antes de trabajar con el PLC real, y sólo cargar el proyecto al PLC real cuando éste ya ha sido probado y depurado antes en el simulador.

El simulador es un programa que está disponible después de instalar UnityPro, y que se puede comunicar con UnityPro como si fuese un PLC real, para cargar proyectos en él y ejecutarlos. Para arrancar el simulador, hay que acceder a la lista de programas del Menú de Inicio de MS: Windows, y dentro de la carpeta Schneider Electric-Unity Pro, hacer clic en "Simulador del PLC", como muestra la Figura 30. El simulador puede iniciarse con el software de Unity Pro ya en marcha.

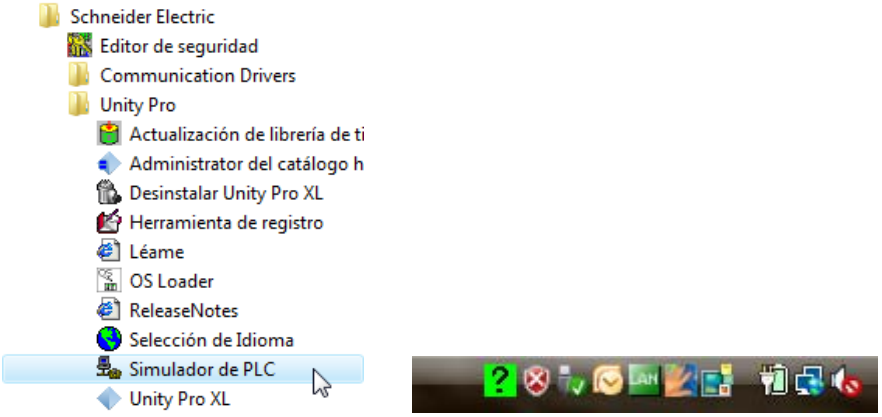


Figura 30. Ejecución del simulador en el menú de inicio de MS. Windows.

En la barra de tareas de MS. Windows aparece el icono para indicar que el simulador está en marcha (ver Figura 30), aunque sin configurar. Al hacer doble clic sobre ese icono, se mostrará u ocultará el panel del simulador que muestra el estado del PLC simulado. La Figura 31 muestra el aspecto inicial del panel del simulador, cuando aún no se ha cargado ningún programa en él, por lo que no tiene todavía una configuración válida.

Figura 31. Panel del simulador del PLC.

Después de haber generado un proyecto con éxito, éste se puede cargar en el PLC (o en el simulador). Pero antes es necesario conectar UnityPro con el PLC. Para ello, hay que escoger la opción del menú PLC-Conectar en UnityPro, o usar el botón de la barra de herramientas. Tras la conexión, el icono del simulador en la barra de tareas cambia a para indicar que el PLC está conectado. La conexión se puede finalizar con la opción PLC-Desconectar del menú de UnityPro, o pulsado en el botón de la barra de herramientas.

El proyecto se puede cargar al simulador con la opción PLC-Transferir proyecto al PLC del menú de UnityPro, o con el botón de la barra de herramientas. Cuando el simulador recibe un proyecto válido, su icono en la barra de tareas de MS: Windows cambia a .

Finalmente, con las opciones Ejecutar y Detener del menú PLC de UnityPro, o con los botones RUN y STOP de la barra de herramientas, se puede poner en marcha y parar el programa en el PLC. Cuando el simulador está ejecutando un programa correcto, su icono en la barra de tareas de MS. Windows cambia a . La Figura 32 muestra el aspecto del panel del simulador cuando está ejecutando un programa.

Figura 32. Panel del simulador con un programa en ejecución.

Pulsando con el botón derecho del ratón sobre el icono del simulador en la barra de tareas de MS. Windows se tiene acceso a diferentes opciones del simulador.

4.2. Análisis y depuración

Para comprobar si un nuevo proyecto funciona bien con una simulación, no basta con cargarlo al PLC y ejecutarlo, sino que hay que comprobar que el programa se comporta correctamente actuando sobre las entradas y viendo cómo evolucionan no solo las salidas, sino también, las variables del programa.

UnityPro facilita la depuración de un programa mostrando en tiempo real como evolucionan las variables y las secciones de los programas mientras el PLC, o el simulador del PLC, se está ejecutando. Para ello es necesario tener conectada la aplicación al PLC e iniciar la ejecución, como se describe en el apartado anterior, durante la ejecución. También hay que verificar que la opción de animación está activada, comprobando que botón de la barra de herramientas está pulsado, como muestra la Figura 33.

Figura 33. Botón de animación activado.

La Figura 34 muestra el aspecto de una sección de lenguaje de contactos durante la conexión al PLC y la ejecución del proyecto. Se observa como los contactos, líneas, bobinas y expresiones aparecen coloreadas en rojo o verde. El color rojo significa contacto abierto, línea desactivada, bobina apagada, o expresión que devuelve falso. En contraste, el color verde significa contacto cerrado, línea activada, bobina encendida o expresión cierta. Las secciones de otros lenguajes se comportan de modo similar durante la ejecución.

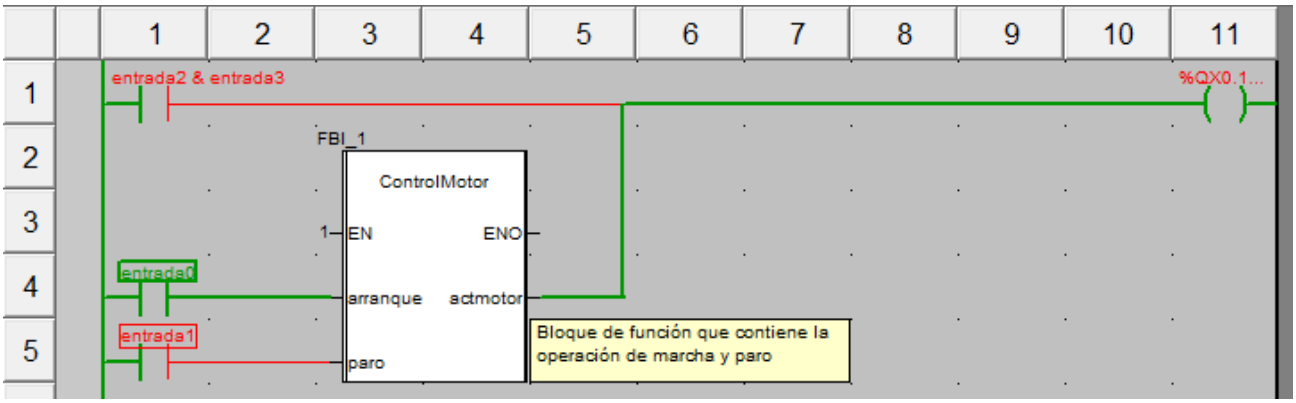


Figura 34. Hoja de una sección LD durante la ejecución.

Para poder actuar sobre las entradas, cambiando sus valores, es necesario listar en una tabla de animación las variables que desean ser alteradas desde UnityPro durante la ejecución. Para ello, primero hay que seleccionar con el ratón la carpeta Tablas de Animación del Explorador de Proyectos, pulsar el botón derecho, y en el menú emergente escoger la opción "Nueva tabla de animación", como muestra la Figura 35.

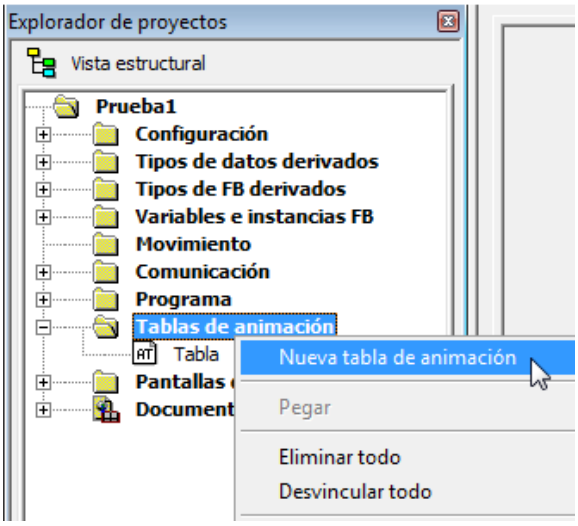


Figura 35. Creación de una tabla de animación en un proyecto.

En la tabla de animación se pueden añadir las variables sobre las que se desea actuar modificar durante la ejecución. Estas variables serán principalmente variables o direcciones referentes a las entradas del PLC, pero también pueden ser otras variables del programa que interese modificar.

Durante la ejecución se puede pulsar sobre el botón Forzar de dicha tabla, como muestra el ejemplo de la Figura 36, para proceder a forzar un nivel lógico alto o bajo en las variables correspondientes a líneas de entrada digitales. Después, con los botones y se puede proceder a forzar una transición a nivel bajo o a nivel alto. Para el caso de variables no booleanas, se puede usar el botón Modificación, que permite especificar un valor numérico. Así se puede comprobar cómo el programa responde a los cambios de las líneas de entrada o de las variables del programa.

Figura 36. Tabla de animación para forzar valores en las entradas.

UnityPro dispone de otras herramientas avanzadas para la simulación, que permiten añadir puntos de parada en las secciones de las tareas del programa y realizar ejecuciones paso por paso. A estas herramientas se accede desde el menú Debug, o desde la barra de herramientas de depuración, y estarán disponibles en función del lenguaje usado en sección.

4.3. Conexión del PLC real al PC y carga del programa

Para cargar un programa en el PLC desde el PC, primero deben conectarse ambos equipos. Gracias a al puerto USB del PLC, este proceso es bastante sencillo. Primero con el software UnityPro cerrado, se debe conectar el PLC a la red, y seguidamente conectar el PLC al ordenador personal con el cable USB. El sistema operativo MS. Windows reconocerá el nuevo dispositivo USB y la conexión quedará lista.

Después, al arrancar UnityPro, y tras cargar el proyecto deseado, se debe verificar que la conexión USB está seleccionada. Para ello hay que seleccionar la opción "Establecer dirección..." del menú PLC. Esto abrirá una ventana como la mostrada en la Figura 37, en la que hay que debe estar marcada la opción USB en el cuadro Medios dentro del apartado PLC.

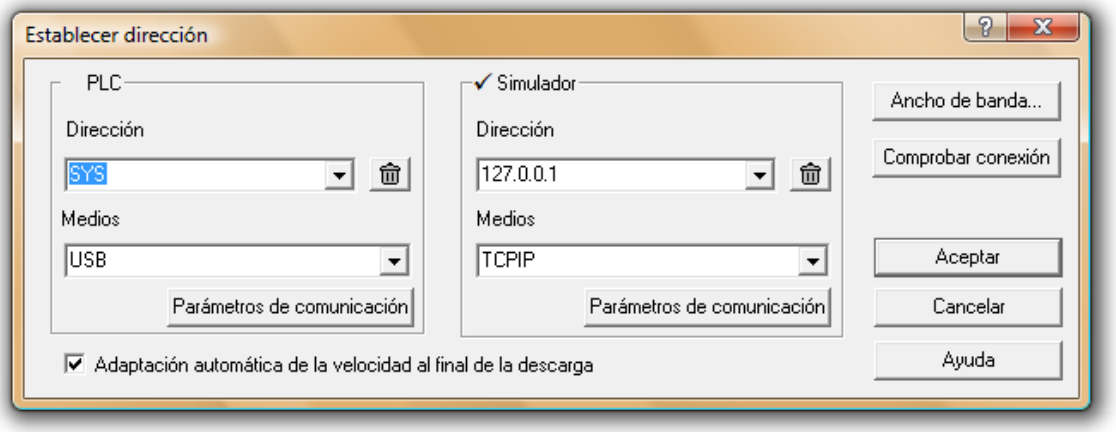


Figura 37. Selección del puerto de conexión al PLC.

Tras verificar que un proyecto funciona bien en simulación, éste se puede cargar en la CPU del PLC real siguiendo los mismos pasos que se utilizan para el simulador de PLC, pero seleccionando antes la conexión con PLC real en vez de con el simulador. El tipo de PLC a utilizar, simulado o real, se selecciona escogiendo una de las siguientes opciones disponibles en el menú PLC, o en los botones de la barra de herramientas:

- Modalidad estándar. Permite cargar el programa en el PLC real.
- Modalidad de simulación. Utiliza el simulador del PLC.

Una vez escogida la modalidad estándar, los pasos a seguir para cargar el programa son los mismos que en simulación:

- Analizar el proyecto.
- Generar cambios o Regenerar todo el proyecto.
- Conectar con el PLC.
- Enviar proyecto al PLC.

4.4. Indicadores de los módulos

Cada módulo lleva unos indicadores que muestran el estado de su funcionamiento, como muestra la Figura 38, los cuales ayudan a detectar posibles fallos. Las siguientes tablas describen de forma resumida el significado de los indicadores de los módulos del PLC usado en prácticas.

La información detallada sobre el significado de los indicadores de cada módulo se puede obtener en la pestaña "Vista General" de la configuración del módulo en cuestión dentro de UnityPro. Para acceder a la configuración de un módulo, basta con acceder a la configuración del bastidor dentro de la sección Bus PLC del Explorador de proyectos, y hacer doble clic con el ratón sobre el módulo deseado.

Figura 38. Indicadores de los módulos del PLC.

Módulo de E/S digitales (BMX DDM 16022)				
LED	**Color**	**Encendido**	**Parpadeando**	**Apagado**
RUN	Verde	Módulo correcto.		Fallo en módulo.
ERROR	Rojo	Fallo en el módulo.	Sin configurar o error de comunicación con CPU.	Correcto.
I/O	Rojo	Error de voltaje o cortocircuito.		Correcto.
Canal	Amarillo	Línea a 1		Línea a 0.

Módulo de E/S analógicas (BMX AMM 600)				
LED	**Color**	**Encendido**	**Parpadeando**	**Apagado**
RUN	Verde	Módulo correcto.		Fallo en módulo.
ERROR	Rojo	Fallo en el módulo.	Sin configurar o error de comunicación con CPU.	Correcto.
I/O	Rojo	Error de voltaje, sobrecarga o cableado incorrecto.		Correcto.
Canal	Amarillo	Canal correcto.	Sin señal. Rebasamiento de rango.	Canal no configurado.

Módulo de CPU (BMX 34 2020)				
LED	**Color**	**Encendido**	**Parpadeando**	**Apagado**
RUN	Verde	Programa en marcha son normalidad.	PLC detenido o bloqueado por un error.	PLC sin configurar.
ERROR	Rojo	Error del procesador.	PLC sin configurar o bloqueado. Error de bus.	Funcionamiento normal.
I/O	Rojo	Errores en módulo E/S.	Error en el bus.	Funcionamiento normal.
ETH ACT	Verde	Conexión detectada.	Transmisión de datos.	Sin conexión.
ETH 100	Verde	Conexión a 100Mbps.		Conexión a 10Mbps
ETH STS	Verde	Comunicación correcta.	Problema de comunicación.	
SER COM	Amarillo		Transmisión de datos.	
CARD ERR	Rojo	Sin tarjeta o tarjeta con datos erróneos.		Tarjeta correcta con contenido correcto.
CARD AC	Verde	Acceso habilitado a la tarjeta.	Accediendo a la tarjeta.	Acceso a la tarjeta inhabilitado.

5. Desarrollo y utilización de bloques de funciones

5.1. Tipos de bloques de funciones

Un bloque de función representa una operación u algoritmo que puede ser utilizado dentro de los programas, y en cualquiera de los lenguajes soportados. En la aplicación UnityPro de Schneider hay disponibles tres clases de bloques de funciones:

- Las funciones elementales (EF), que representan operaciones sencillas sin estado interno.
- Los bloques elementales de función (EFB), que representan algoritmos más avanzados, con estado interno.
- Los bloques de función derivados (DFB), que son funciones creadas por el usuario.

UnityPro incorpora una biblioteca con un gran número funciones EF y EFB que resuelven muchas tareas comunes, y que están optimizados para un funcionamiento eficaz. Además, un usuario siempre puede desarrollar nuevos DFB para tareas más específicas u operaciones que se le plantean regularmente, de forma que no necesite volver a escribir el

programa de estas operaciones cada vez que las tenga que usar. Un DFB se utiliza de la misma manera que un EF o un EFB de la biblioteca estándar.

Mediante el uso de DFB se consigue una mayor estructuración de los programas así como una reutilización de los algoritmos programados. Los DFB se pueden añadir a la biblioteca de funciones de UnityPro, y también se pueden exportar para luego ser importados y reusados en otros proyectos. Además, el uso de DFB facilita la depuración y el mantenimiento de los programas.

5.2. Uso de los bloques de función en UnityPro

Durante la programación de una sección de una tarea, en cualquiera de los lenguajes, se puede usar el Asistente de Entrada FFB, al que se puede acceder de varias formas:

- Desde el menú de Edición, en la opción Asistente de Entrada FFB.
- Pulsando en el botón  de la barra de herramientas.
- Posicionando el ratón en el área de una sección de programa y pulsando el botón derecho. En el menú emergente está la opción Asistente de Entrada FFB.
- Pulsando las teclas CONTROL+I cuando está activa una ventana con una sección del programa.

Cualquiera de las alternativas anteriores muestra el Asistente de Entradas de Función, como la que se puede ver en la Figura 39, en el cual se puede indicar el nombre del bloque que se desea incorporar en el programa (Tipo de FFB), o escoger entre los recientemente usados. Además, si se pulsa sobre el botón superior de puntos suspensivos, se tendrá acceso a la lista de Selección de Tipos de FFB, mostrado en la Figura 40, donde aparecen ordenados por categorías todos los bloques de función disponibles en la biblioteca y en la aplicación. De la lista se puede escoger el bloque deseado y después pulsar el botón Aceptar. Posteriormente hay que pulsar Aceptar en el Asistente de Entradas de Función. Finalmente, se puede situar el cursor sobre la parte del programa donde se quiere insertar el bloque de función y hacer clic.

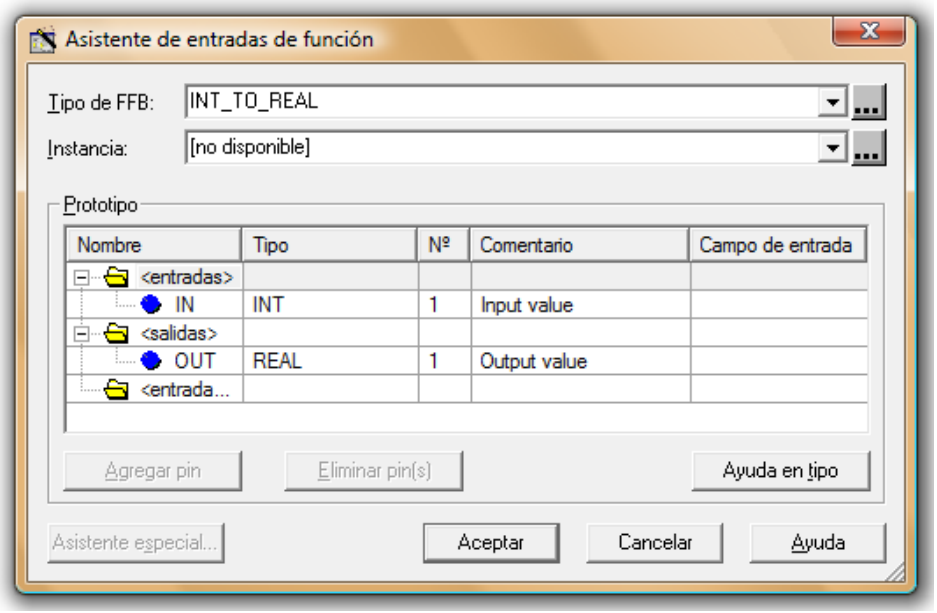


Figura 39. Asistente para añadir bloques de función a una sección.

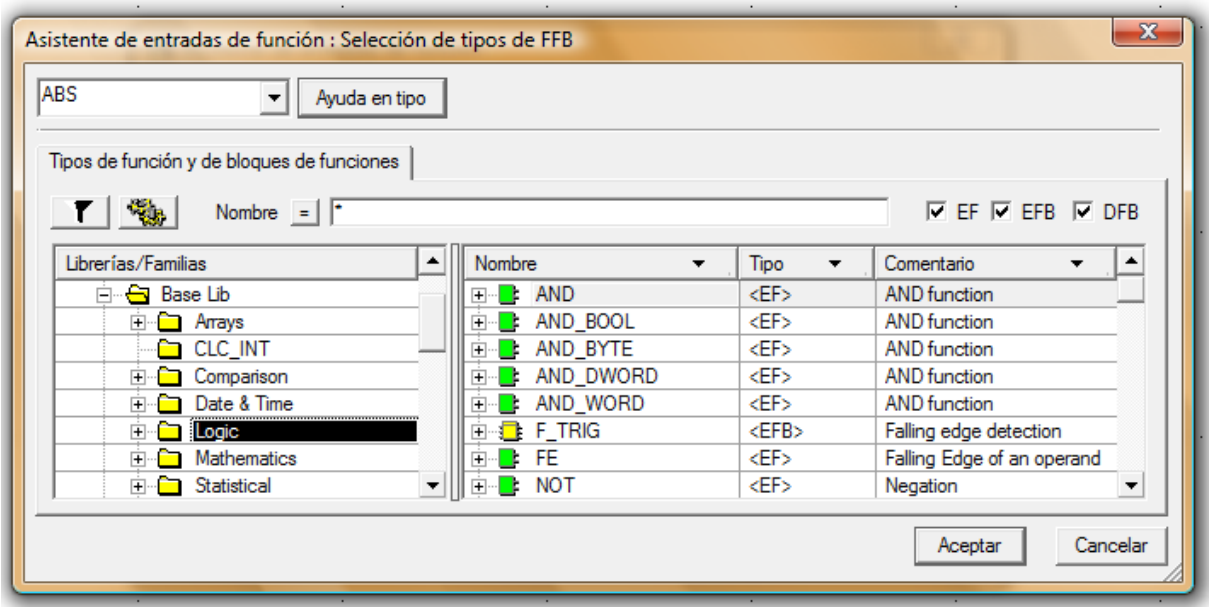


Figura 40. Asistente de selección de bloques de función.

Otra forma de colocar bloques de función en las secciones del programa es mediante el Explorador de Librerías de Tipos, que se muestra normalmente sobre el Explorador de Proyectos después de pulsar el botón de la barra de herramientas, o escoger la opción Explorador de Librerías de Tipos del menú de Herramientas. La Figura 41 muestra una vista de este explorador.

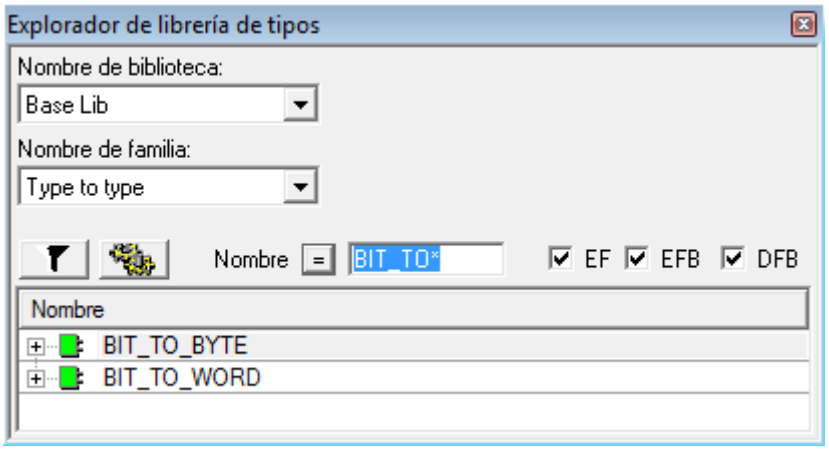


Figura 41. Explorador de la librería de tipos.

En el Explorador de Librerías de Tipos se puede seleccionar la biblioteca y la familia de funciones, así como filtrar por un nombre o parte de él, para encontrar la función buscada. Desde la lista con los nombres de funciones que coinciden con las opciones de búsqueda, se puede seleccionar la función deseada y arrastrarla con el ratón hasta el área de una sección del programa para insertarla en esa sección.

5.3. Creación de DFB

Para poder definir y utilizar un nuevo DFB en la aplicación, primero se tiene que definir el nuevo tipo de DFB, y después hay que crear un objeto de ese nuevo tipo de bloque, que será el que se inserte en una sección del programa de la aplicación.

Como representa la Figura 42, un DFB se compone de las siguientes partes:

- Parámetros de entrada, de salida y de entrada-salida, que sirven de interfaz entre los programas y el bloque de función. Los parámetros de entrada son variables actualizadas por la aplicación y leídas dentro del DFB, mientras que los de salida son variables actualizadas por el DFB y leídas en la aplicación. Los parámetros de entrada-salida son variables actualizadas por la aplicación, luego modificadas por el DFB, y finalmente usadas de nuevo por la aplicación.
- Variables internas utilizadas por el algoritmo de procesamiento. Estas variables pueden ser privadas y públicas. Mientras que a las variables públicas se puede acceder desde el programa de aplicación, a las privadas solo se pueden acceder desde el DFB.
- Un algoritmo de procesamiento que utiliza las variables de entradas y ofrece información en las variables de salidas. Este algoritmo se describe mediante secciones de programa que se pueden describir con diagrama de contactos (LD), lista de instrucciones (IL), texto estructurado (ST) o diagrama de bloques de funciones (FBD).

El primer paso para crear un nuevo tipo de DFB es seleccionar la carpeta Tipos de FB Derivados que hay dentro del Explorador de Proyectos, como muestra la Figura 43. Se abrirá entonces el Editor de Datos con su pestaña Tipos de DFB, como enseña la Figura 44. En el Editor de Datos se introducirá el nombre del nuevo bloque, como se ve en la Figura 44.

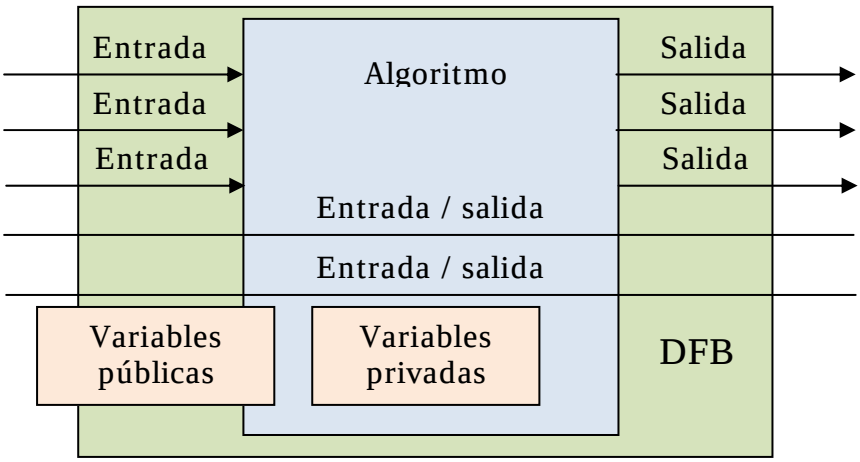


Figura 42. Estructura de un DFB.

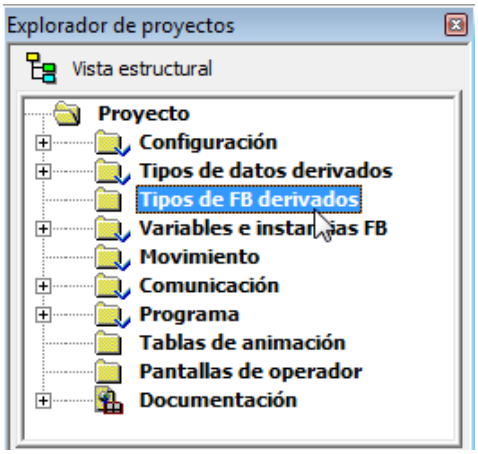


Figura 43. Carpeta de Tipos de FB Derivados en el Explorador de Proyectos.

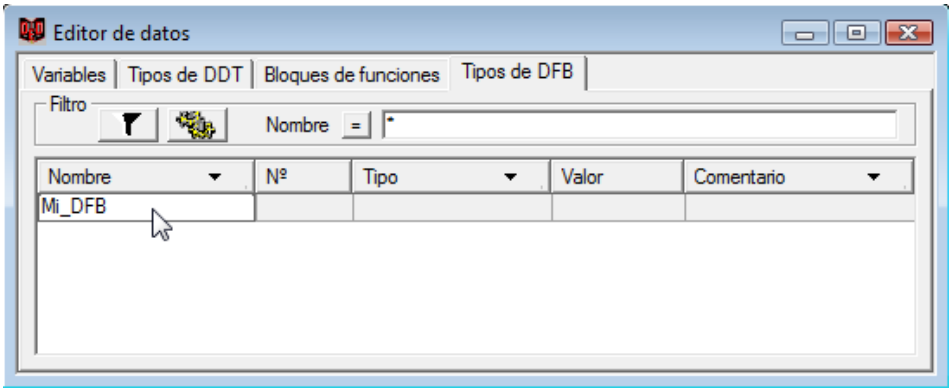


Figura 44. Creación de un nuevo tipo de DFB en el Editor de Datos.

A continuación hay que especificar el nombre y tipo de los parámetros de entrada y salida, y de las variables públicas y privadas, así como el nombre de las secciones de programa y su lenguaje, como muestra la Figura 45. También se pueden indicar comentarios que describan los parámetros y variables.

En la Figura 45 también se observa que junto con el nombre del nuevo DFB aparece un indicador de "obras", que refleja que el nuevo tipo de función no está completo todavía.

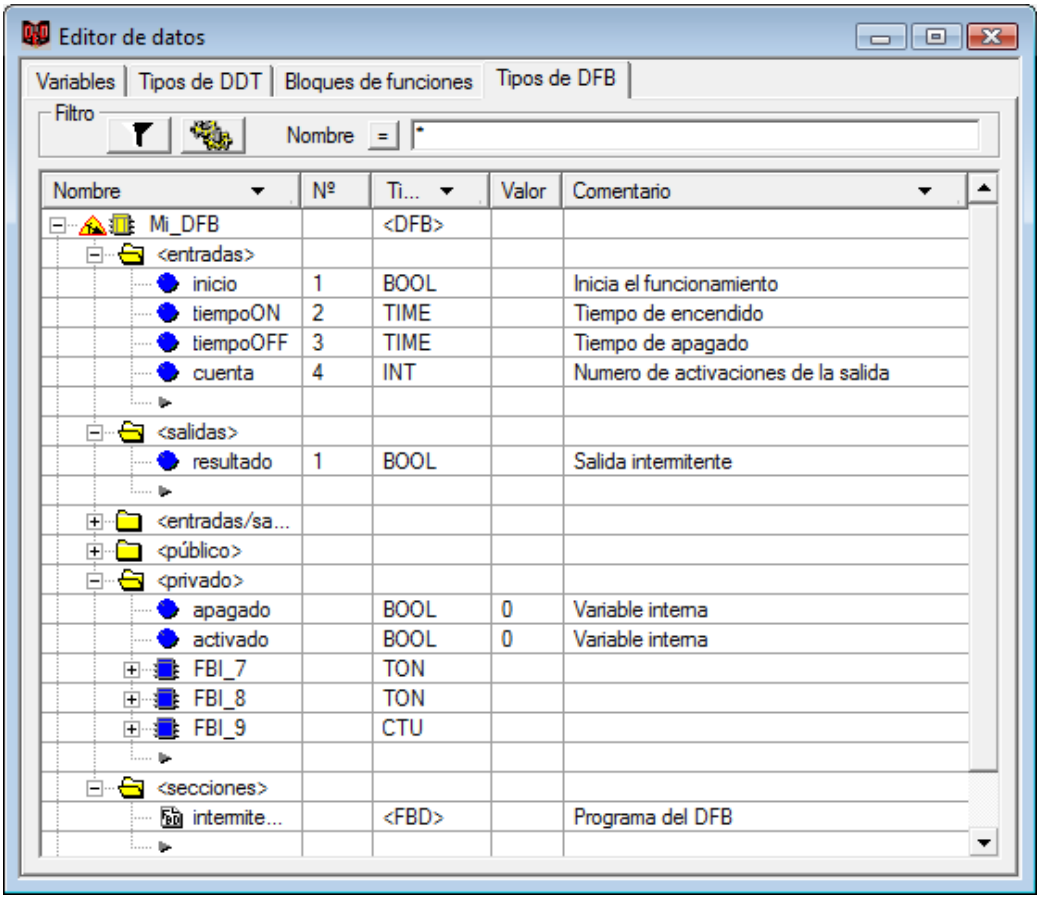


Figura 45. Edición de los parámetros de un nuevo tipo de DFB.

Seguidamente hay que editar las secciones de programa del nuevo tipo de DFB, para lo cual se puede hacer doble clic sobre ellas con el ratón. La Figura 46 muestra el diagrama de bloques de funciones que corresponde con la sección del tipo de DFB de la Figura 45. Se observa cómo se pueden utilizar otros bloques de función ya definidos para crear el programa del nuevo DFB.

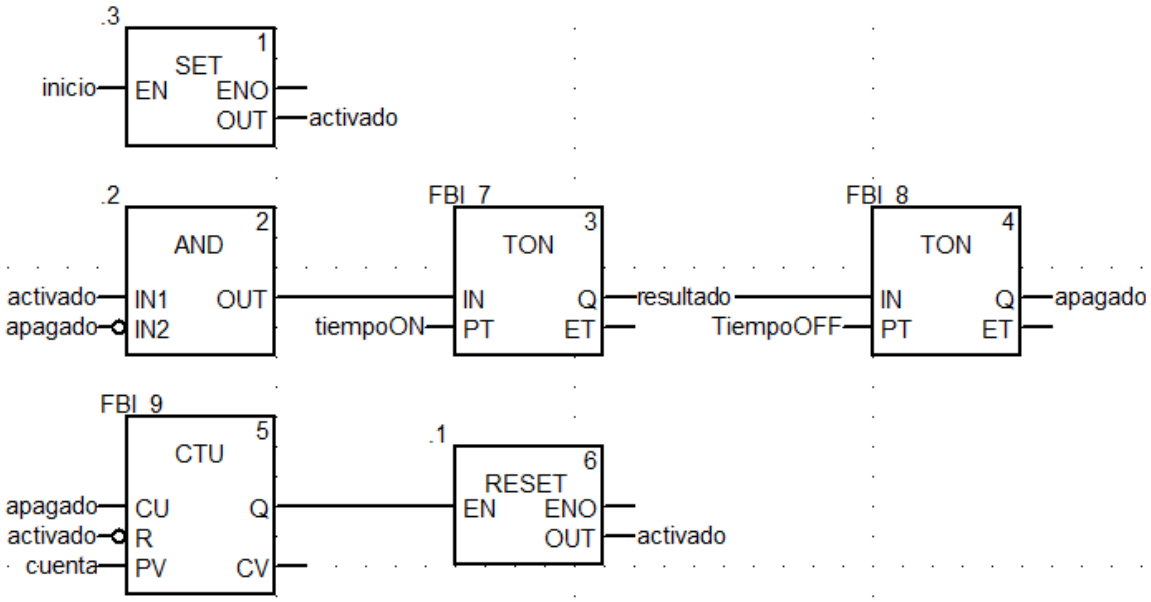


Figura 46. Ejemplo de sección con el algoritmo de un nuevo DFB.

Cabe destacar que, en las secciones de programa del algoritmo del DFB, sólo se puede utilizar los parámetros de entrada y salida, así como variables públicas y privadas, que se han definido en su declaración de tipo. De esta forma, un DFB no puede utilizar las variables globales de la aplicación ni los objetos de módulos de entradas y salidas, con la excepción de los bits y las palabras de sistema (%Si, %SWi y %SDi).

A continuación hay que seleccionar la opción de Analizar el Proyecto, bien en el menú Generar o bien con el botón de la barra de herramientas. Si no hay errores, el nuevo DFB aparecerá sin el indicador de "obras" en el Editor de Datos, como muestra la Figura 47.

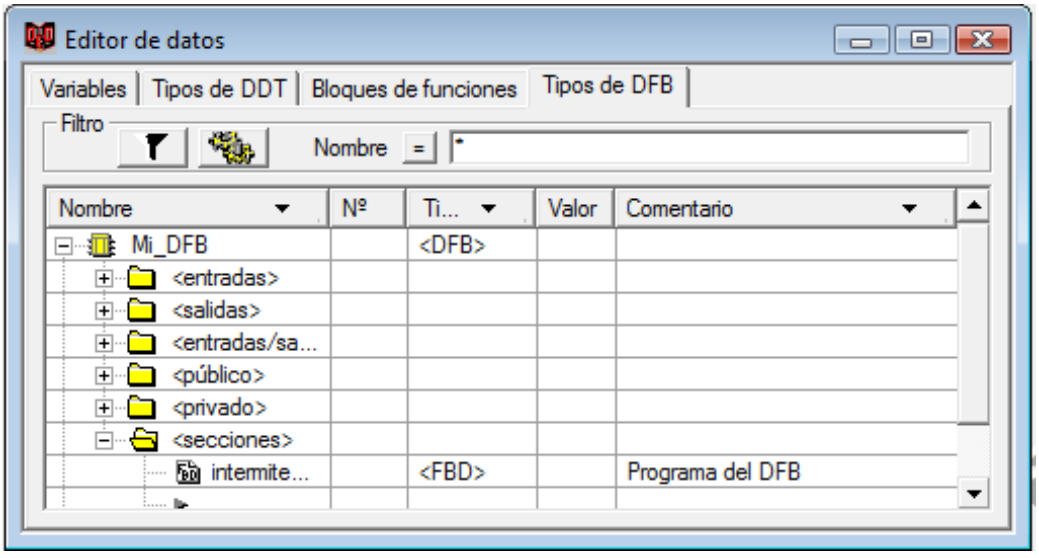


Figura 47. Nuevo tipo de DFB listo para usar.

Para utilizar el DFB basta con seguir los pasos descritos en el apartado 5.2. A la hora de buscar el bloque en la lista de selección de tipos del Asistente de Entradas de Función, éste aparecerá dentro de la carpeta Aplicación. Será más fácil encontrarlo si solo se marca la casilla DFB del filtro, como muestra la Figura 48.

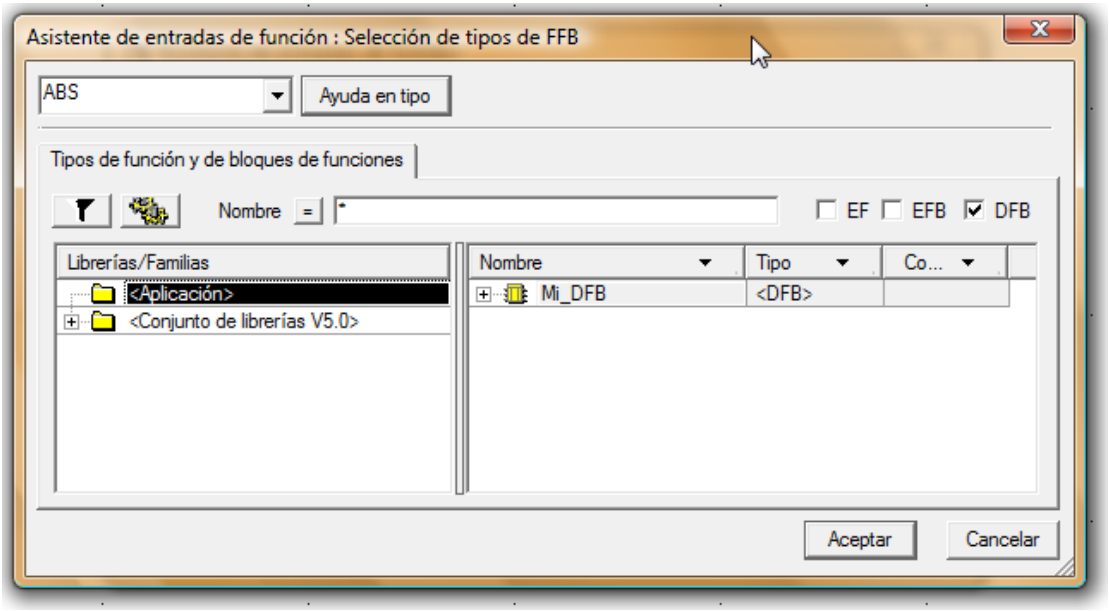


Figura 48. Nuevo tipo de DFB en la lista del Asistente de entradas de función.

La Figura 49 muestra el aspecto del DFB que fue creado con los pasos descritos en este apartado, dentro de una sección con lenguaje de diagrama de contactos. Cuando se inserta el DFB en una sección, se crea automáticamente una nueva instancia del tipo de DFB creado anteriormente y listado en la pestaña Tipos de DFB del Editor de Datos. Esta nueva instancia aparecerá listada en la pestaña Bloques de Funciones del Editor de Datos, como se observa en la Figura 50.

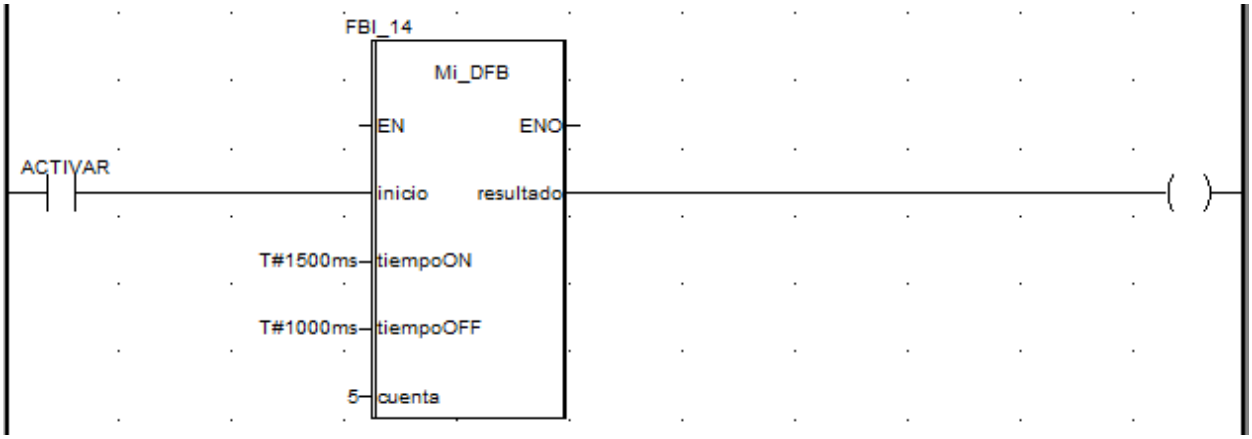


Figura 49. Instancia del nuevo DFB en una sección de programa.

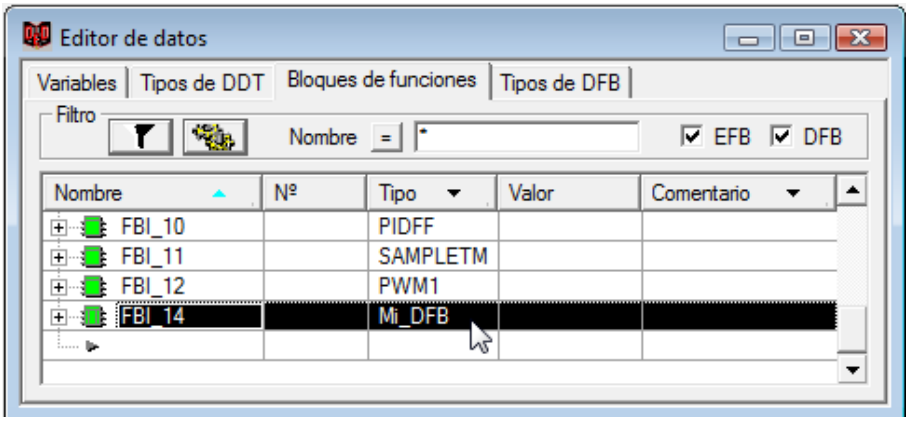


Figura 50. Instancia del nuevo DFB listada en el Editor de Datos.

Finalmente, cabe comentar que es posible vincular un fichero de ayuda en formato HTML a cada nuevo DFB desarrollado por el usuario e incorporado a la biblioteca.

5.4. Reutilización de los DFB

Una vez que se ha creado un nuevo tipo de DFB, este se puede exportar a un archivo del disco para luego ser importado en otros proyectos, o se puede incorporar a la biblioteca de bloques de funciones de UnityPro para que estés disponible para cualquier otro proyecto.

Para realizar esas acciones, hay que seleccionar el nuevo tipo de DFB dentro de la carpeta Tipos de FB Derivados del Explorador de Proyectos, y pulsar sobre el botón derecho del ratón, como muestra la Figura 51.

En el caso de querer incorporar un nuevo DFB en la biblioteca de funciones será necesario escoger la carpeta de la biblioteca donde agregarlo. La carpeta seleccionada por defecto se denomina Custom Family.

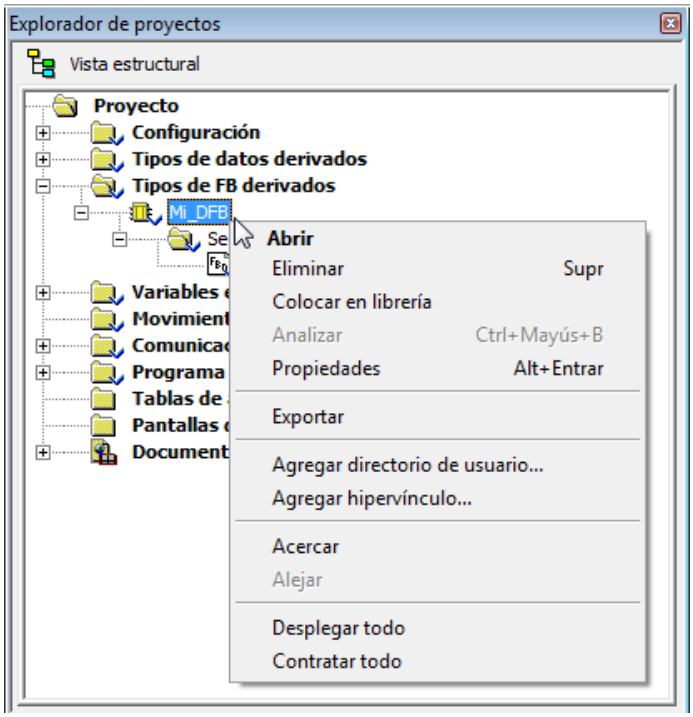


Figura 51. Opciones para un nuevo tipo de DFB en el Explorador de Proyectos.

6. Secciones GRAFCET en UnityPro

El diagrama secuencial de funciones (SFC) o GRAFCET es un lenguaje gráfico del estándar IEC 61131 muy adecuado para programar automatismos que realizan operaciones según una secuencia de etapas. UnityPro y los PLC M340 soportan la programación con este lenguaje.

Para crear una nueva sección GRAFCET en UnityPro basta con seleccionar carpeta Sección de la tarea que interese dentro del Explorador del Proyectos, y después hacer clic con el botón derecho del ratón encima, como muestra la Figura 52. En el menú emergente se escoge Nueva Sección, y en la ventana emergente se escoge como tipo de lenguaje SFC. También se requiere especificar un nombre pata la nueva sección.

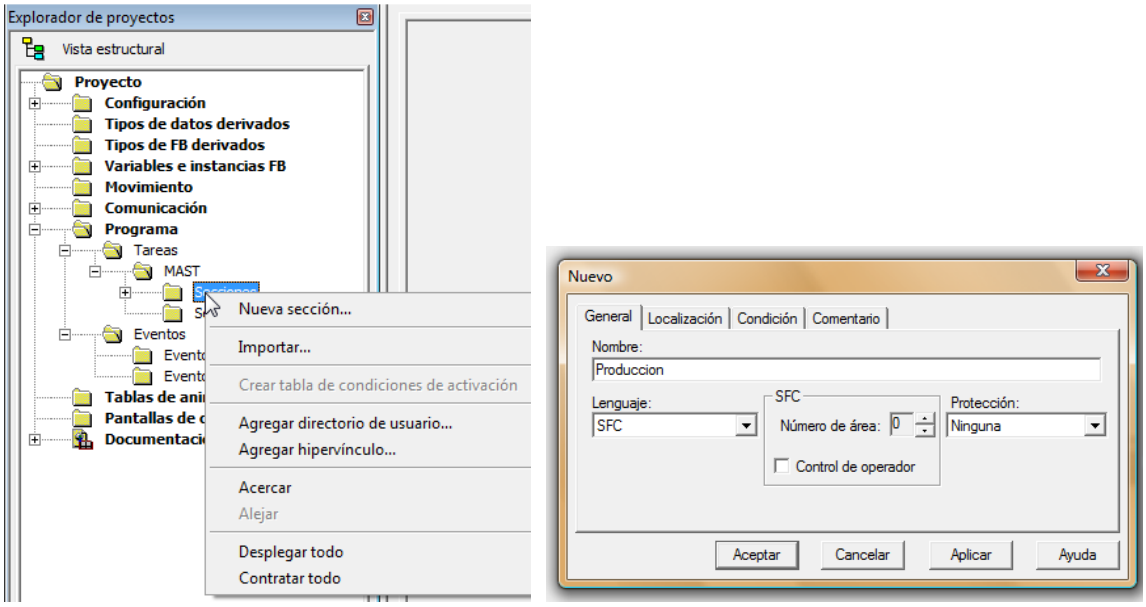


Figura 52. Creación de una nueva sección de programa y selección del lenguaje SFC.

El ejemplo de la Figura 53a muestra como se organizan, dentro del explorador de proyectos, los distintos componentes que forman parte de las secciones SFC que se están programando. Estos componentes son el Chart (diagrama GARFCET), las Secciones de acciones, las Secciones de transiciones y las Macroetapas. Mientras que las macroetapas utilizadas se muestran dependientes del diagrama donde se llaman, las macroetapas que se han creado pero que luego no se usan se mantienen en la carpeta Macros no utilizadas.

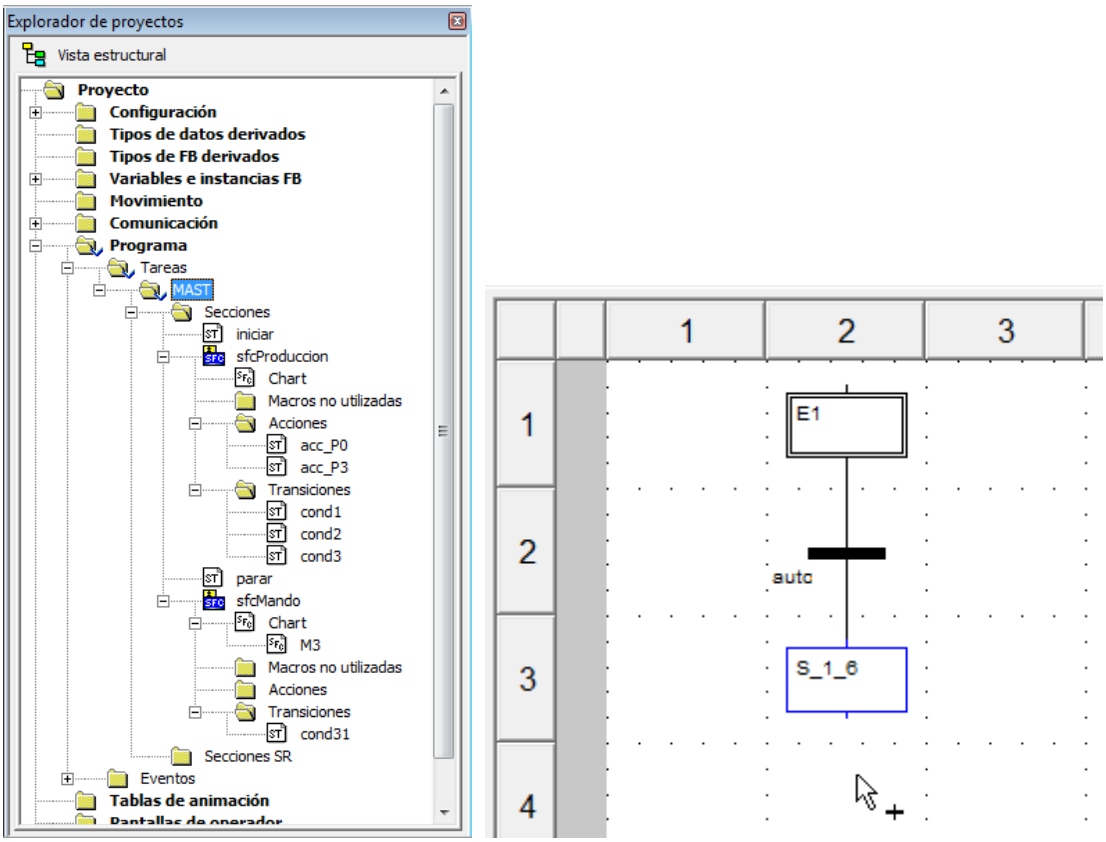


Figura 53. A) Estructura de las secciones SFC. B) Edición de una sección SFC.

Dentro de la ventana de edición de una sección e GRAFCET, se dispone de una cuadrícula en donde se pueden dibujar las etapas, transiciones y conexiones, como muestra la Figura 53b. En cada rectángulo de la cuadrícula se puede colocar una etapa o una transición. Haciendo doble clic con el ratón sobre una etapa o transición se pueden editar las acciones o condiciones correspondientes. En las propiedades de las etapas, también se puede escoger si una etapa es inicial o no. Durante la edición de la sección SFC, se dispone de una barra de herramientas (Figura 54) que permite insertar etapas, macroetapas, transiciones, saltos, bifurcaciones, conexiones y comentarios de texto. También se pueden insertar secuencias de etapas con una sola acción.

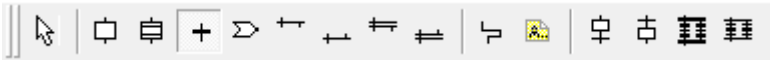


Figura 54. Botones de herramientas para la edición de una sección SFC.