Universidad Politécnica de Cartagena | Campus de Excelencia Internacional

Grado en Ingeniería Telemática
2020-2021

Trabajo Fin de Grado

DISEÑO DE UN SISTEMA DE CONTROL AUTOMÁTICO EN UNA VIVIENDA CON UN AUTÓMATA PROGRAMABLE MODICON Y ENTRADAS SALIDAS DISTRIBUIDAS

Juan José Soriano Fernández

Director
José Fernando Cerdán Cartagena

Codirector
Daniel Carreres Prieto

AGRADECIMIENTOS:

En primer lugar, querría agradecer a mi padre la ayuda, conocimiento y tiempo que me ha proporcionado durante toda mi vida académica, sin la cual la realización de este trabajo no habría sido posible. Ha hecho que estos años sean más llevaderos a pesar de todas las dificultades que se hayan podido poner en mi camino.

Por otro lado, agradecer a mi director, José Fernando Cerdán Cartagena, y a mi codirector, Daniel Carreres Prieto, que me han ayudado satisfactoriamente a completar este trabajo.

También quiero agradecer a mi madre, pareja, hermana y demás familia, así como compañeros y amigos, por todo el apoyo y consejos dados en estos años de formación académica.

CONTENIDO

FIGURAS

TABLAS

1. INTRODUCCIÓN

¿Por qué PLC?.

Los grandes avances que se han producido en la industria moderna han incrementado de forma vertiginosa las aplicaciones de los PLC´s en la automatización industrial. Por esta razón, nos vamos a centrar en este proyecto en conocer más sobre estos dispositivos electrónicos que han sido capaces de revolucionar el sector de la industria y haremos uso de ellos para realizar la automatización de una vivienda.

Los PLC´s (Programable Logic Controler) o autómatas programables son unos dispositivos electrónicos que son capaces de implementar una lógica que permiten controlar todo tipo de procesos. Bien sea una máquina herramienta, un proceso industrial, señalización de tráfico, etc. En definitiva, cualquier proceso en el cual existan entradas y salidas desde y hacia el exterior. La gran ventaja de estos sobre otros tipos de dispositivos, como por ejemplo una computadora es la gran cantidad de entradas y salidas que son capaces de gestionar. Además, ofrecen una gran durabilidad y capacidad de funcionamiento en entornos más o menos agresivos para la electrónica con la cual han sido diseñados. Lo anterior, supone un gran ahorro económico en cuanto a su mantenimiento se refiere.

Los PLC´s, han supuesto una gran revolución en el campo de la automatización. Estos dispositivos electrónicos han ido desplazando a los antiguos sistemas de control basados en circuitos electrónicos, relés interruptores, etc.

Por otro lado, la evolución de los lenguajes de programación disponibles, para este tipo de dispositivos ha permitido desarrollar lenguajes de programación más sencillos, lo cual supone un ahorro en el tiempo de formación del personal técnico. Que los lenguajes de programación sean más sencillos no va en detrimento de la potencia de los mismos, así por ejemplo podemos encontrarnos con lenguajes de programación de contactos (muy útil para el mantenimiento), lenguajes estructurados similares a programación orientada a objetos que dan una potencia enorme y que permiten la programación de los procesos desde el exterior del entorno de desarrollo (mediante bases de datos u hojas de cálculo).

Así pues, es evidente que la enorme evolución tanto del hardware y el software de desarrollo ha dado lugar a un campo muy extenso de aplicación de los PLC´s, entre otros pueden ser perfectamente válidos en el campo de la domótica ya que le permite al usuario final un importante beneficio para él. Ganando en tiempo libre, seguridad, ahorro de dinero etc. En definitiva, le permite disfrutar de una vivienda más confortable y segura.

Otro aspecto importante de los PLC´s es la proliferación de los buses de campo (Profibus, CAN, Modbus, etc) los cuales permiten hacer un desarrollo descentralizado de las instalaciones con el consabido ahorro en la tirada de cableado.

2. OBJETO DEL PROYECTO

El objeto del proyecto es el diseño de un sistema automatizado de control de una vivienda haciendo uso de un PLC que utilice las soluciones que ofrecen los fabricantes de los mismos tanto en buses de campo como en pantallas de presentación de información (SCADA -Supervisory Control And Data Acquisition-).

Se trata de una vivienda unifamiliar de tres plantas compuesta por una planta baja o garaje, una planta principal y un ático.

El sistema integrará el control de la puerta corredera de acceso al jardín, la puerta abatible que permite el acceso al garaje o planta cero, el control y automatización del ascensor interno de la vivienda y detectores de movimiento de la vivienda, con los cuales se realizará un sistema de alarma de vigilancia de intrusos.

3. DESCRIPCIÓN DE LA VIVIENDA

La vivienda en la cual se va a realizar la automatización es una vivienda unifamiliar de tres plantas y un pequeño jardín al cual se accede desde el exterior mediante una puerta deslizante.

En la planta sótano existe una puerta batiente que da acceso al interior. En esta planta existen, además de la puerta batiente anteriormente indicada, cinco ventanas y un ascensor que permite el acceso a la planta primera y a un ático. En la figura adjunta se encuentra la distribución de dicha planta.

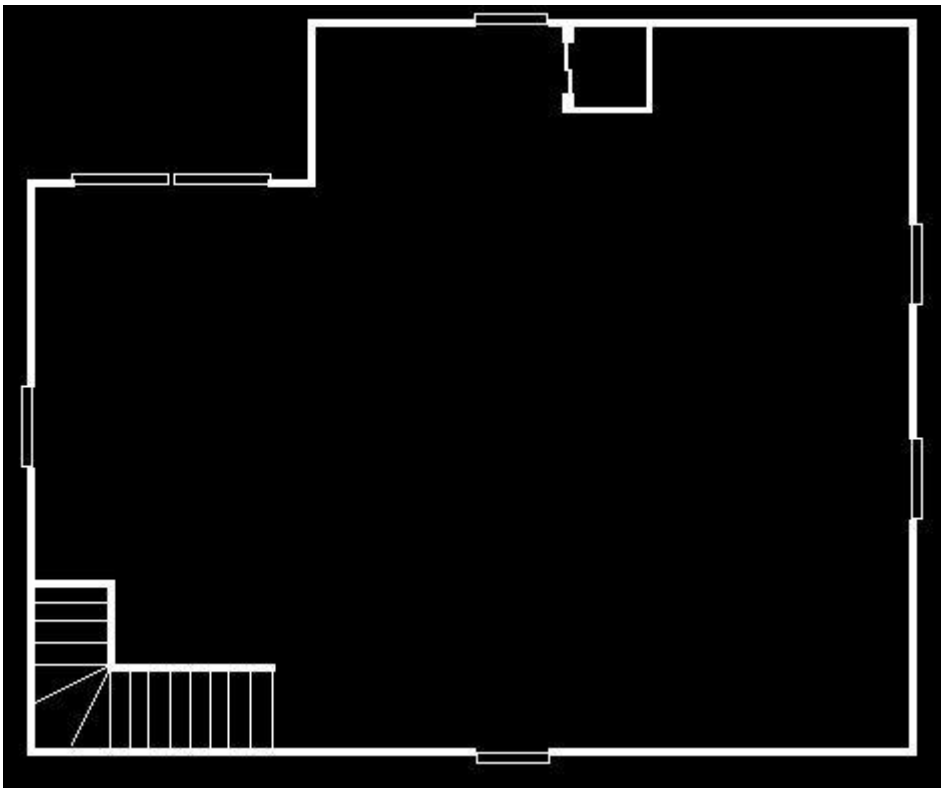


Figura 3-1 Planta 0

La planta primera se compone de una cocina, un comedor, dos cuartos de baño y tres habitaciones. Las ventanas de la planta se muestran en la siguiente figura.

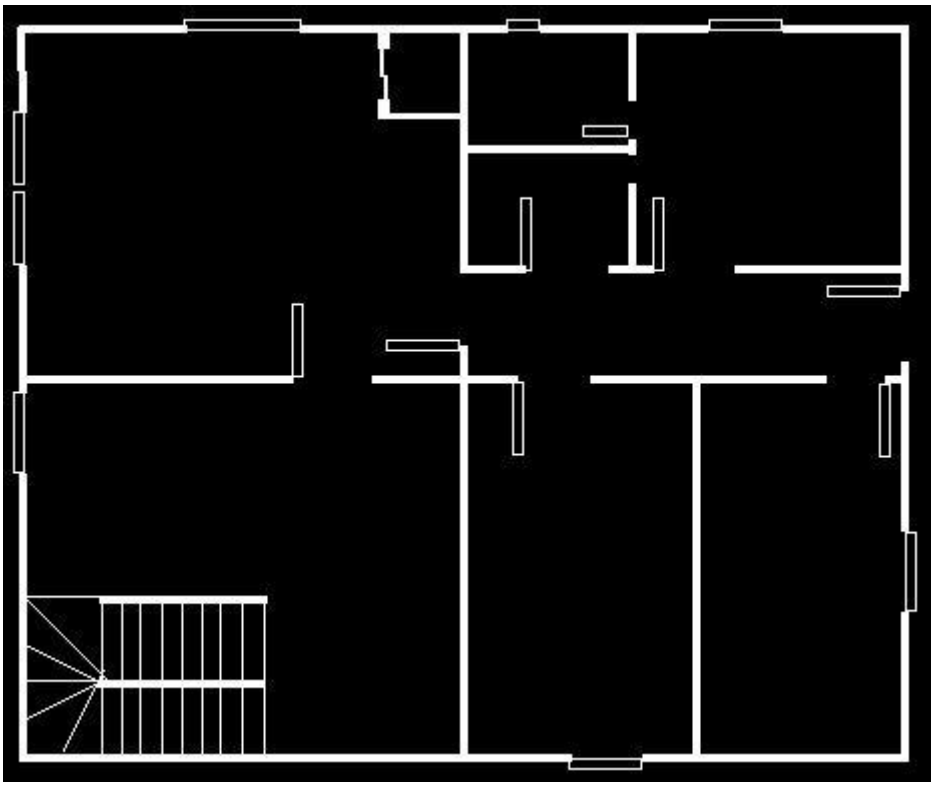


Figura 3-2 Planta 1

El ático de la vivienda está formado por cuatro habitaciones con un baño y la distribución de ventanas son mostrados en la siguiente figura.

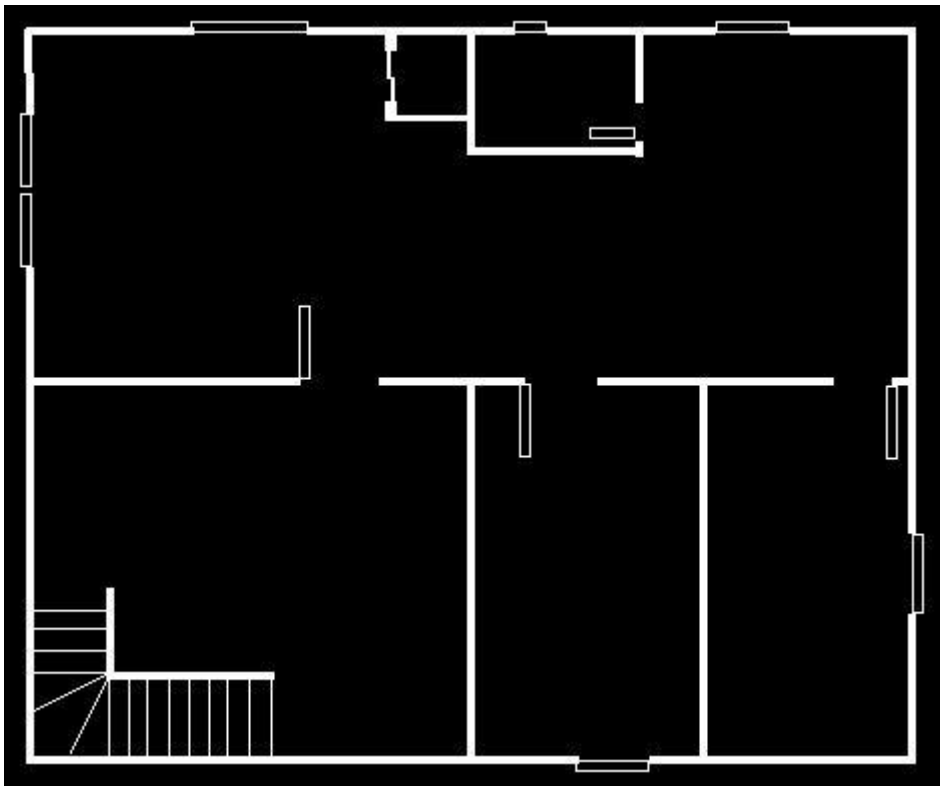


Figura 3-3 Planta Ático

4. DESCRIPCIÓN DE SISTEMA

El sistema de control de la vivienda, a muy alto nivel, estará compuesto por:

- Un PLC
- Una pantalla táctil donde irá alojado el SCADA y con la cual los usuarios de la vivienda podrán interactuar con el sistema de control.
- Un bus de campo para transmisión de datos desde y hacia los sensores y actuadores
- Una infraestructura de red de campo
- Elementos necesarios de campo para la captación de señales de los sensores y actuadores.

El diagrama de bloques de la instalación se indica en la siguiente figura.

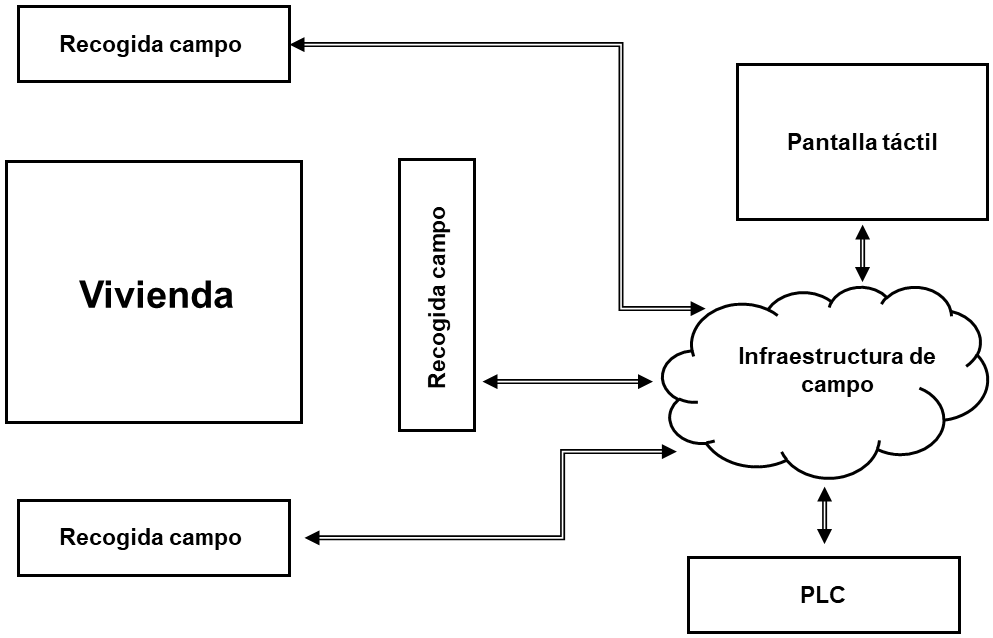


Figura 4-1 Diagrama de bloques instalación

5. ELECCIÓN DE ELEMENTOS DEL SISTEMA

Para facilitar la integración e interoperatividad de todos los elementos del sistema de control, se van a usar componentes que el fabricante Schneider Electric dispone en su catálogo de automatización.

Schneider es uno de los líderes en automatización con una extensa gama de productos que nos permitirá adaptarnos a las necesidades del proyecto. En sus catálogos podemos encontrar PLC compactos, modulares, PLC de altas prestaciones, terminales de operador (SCADA) etc. Todos los PLC´s se programan bajo un entorno común de desarrollo, Unity (EcoExtruxure en su versión mas reciente).

5.1. TOPOLOGÍA DEL BUS DE CAMPO

Desde el punto de vista de captación de señales procedentes de campo y activación de actuadores, se presentan dos opciones:

- Una opción es usar un sistema **centralizado** en el cual todos los sensores y actuadores van conectados a módulos electrónicos de entrada o salida instalados en el propio rack del PLC.

- Otra opción es usar un sistema de captación y actuación **descentralizado,** en el cual los elementos de recogida y actuación se encuentran convenientemente situados en diversos puntos de la vivienda.

  Dentro de esta segunda opción, se presentan a su vez dos alternativas. Una de ellas sería usar una topología en bus y haciendo un uso de un protocolo maestro-esclavo, ir haciendo un *polling* desde el maestro (PLC) a los esclavos (electrónica de recogida y actuación). Otra alternativa, es usar una infraestructura de red ethernet y que el PLC actúe como cliente y la electrónica de captación actúe como servidor.

Dentro de la opción centralizada, Schneider ofrece una gran variedad de módulos de entrada-salida con los cuales podríamos realizar la automatización.

En la opción descentralizada, Schneider ofrece la solución de módulos de entrada salida remota bajo el nombre comercial de *islas Advantys* los cuales son modulares. Esta plataforma es lo suficientemente flexible para adaptarse a prácticamente cualquier aplicación. Entre sus características más destacadas podemos citar:

- Diseño resistente modular y escalable.
- Software disponible que permite la configuración, accesorios y bases de montaje.
- Sistema inteligente: sustitución bajo tensión de los módulos E/S (*Hot-Swap*), configuración automática o por software y diagnósticos integrados.
- Instalación simplificada: alimentación de sensores y accionadores separada y protección integrada frente a cortocircuitos y sobretensiones.
- Conectividad con los siguientes tipos de buses de campo: CANopen, Modbus, DeviceNet, Profibus e Interbus.

En las dos siguientes figuras se muestra los detalles de la solución de Schneider para la adquisición y actuación remota.

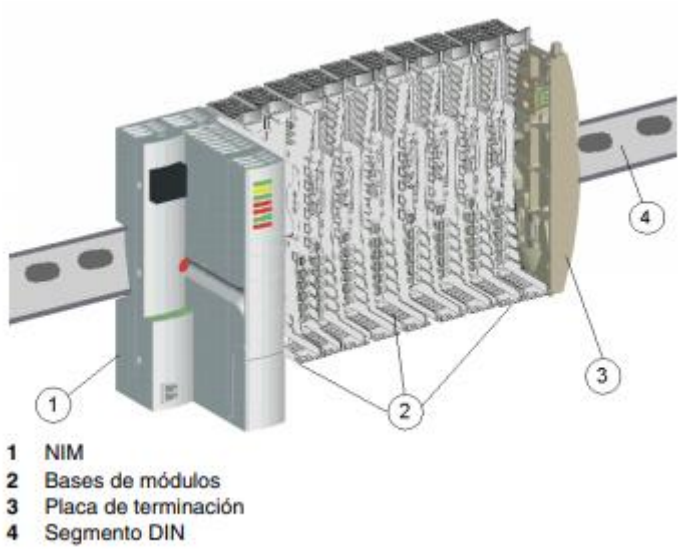


Figura 5-1 Detalle de elementos montaje

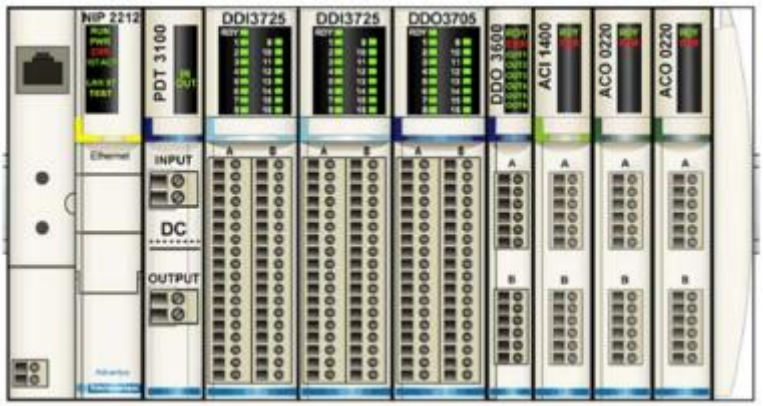


Figura 5-2 Detalle Isla Advantys

En este proyecto optaremos por una solución descentralizada en cuanto a la adquisición de datos de campo. La razón fundamental de esta decisión estriba en la consabida reducción de tirada de cableado que supone, ahorro de tiempo y el mantenimiento sencillo que ofrece.

Queda la decisión de optar por el uso de una topología en bus o una arquitectura de red Ethernet. Entre las opciones que Schneider ofrece, con topología en bus tenemos CANopen, Modbus, DeviceNet, Profibus, Interbus. Sobre Ethernet está la opción Modbus (aunque su uso más conocido es punto a punto o multipunto sobre RS-485 o RS-232).

Cualquiera de las topologías mencionadas anteriormente es perfectamente válida, pero optaremos por el uso de una arquitectura en red. A la hora de detectar posibles errores a nivel de conectividad en la implementación física existen diversas herramientas software libre como WIRESHARK, que pueden ser muy útiles y son de fácil manejo.

Dentro del modelo OSI, la siguiente figura muestra el lugar donde se encuentra este tipo de protocolo.

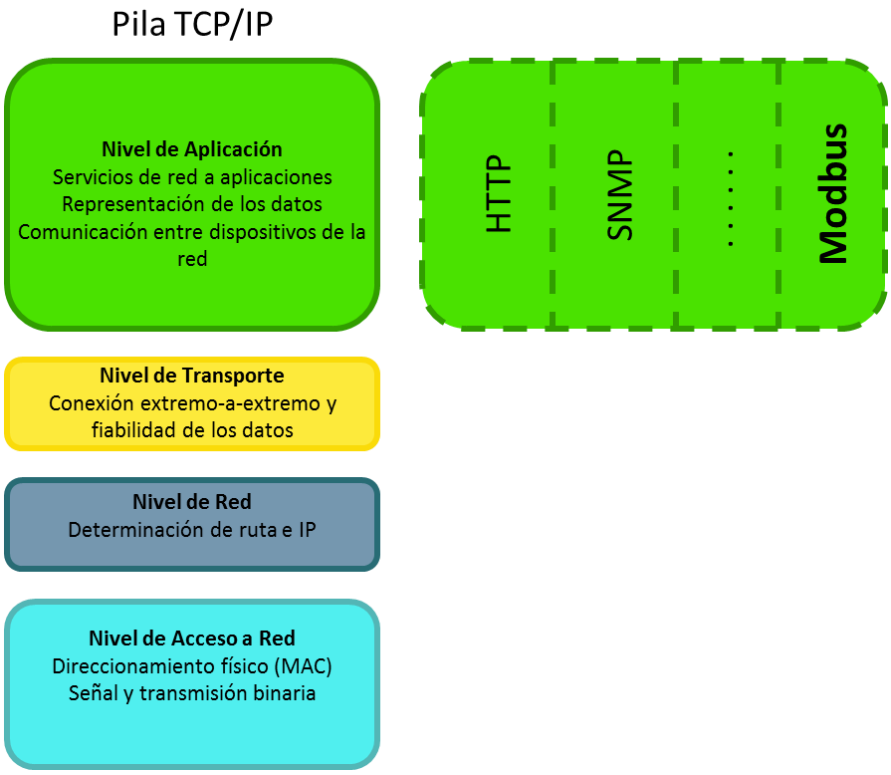


Figura 5-3 Modbus en Pila TCP/IP

5.2. ELECCIÓN DEL PLC

Como PLC, usaremos un autómata de la gama M340. Estos son autómatas modulares y compactos con una capacidad de proceso de las CPU´s de 7 kinst/ms y un tamaño de programa de 70 kinst.

La siguiente imagen ilustra un detalle de este PLC

Figura 5-4 PLC M340

5.3. ELECCIÓN DEL RACK O BASTIDOR

Los bastidores proporcionan las siguientes funciones:

Función mecánica

- Se encargan de la sujeción de todos los módulos del PLC (la fuente de alimentación, la CPU o procesador, los módulos de entrada-salida binaria-analógicas, módulos especiales).
- El bastidor, pueden fijarse en varios tipos de instalaciones:
  - En carcasas.
  - En paneles.
  - En armarios.

Función eléctrica

- A través de sus buses internos, los bastidores proporcionan:
  - La alimentación necesaria para cada módulo del bastidor.
  - Los buses para la comunicación entre módulos y la CPU.
  - Señales de servicio y datos para el conjunto del PLC.

En las guías de selección de Schneider se ofrecen bastidores de 4, 6, 8 y 12 posiciones (slot). Como en el caso que nos ocupa en el bastidor solo tendremos alojada la fuente de alimentación y la CPU, será suficiente con seleccionar un bastidor de 4 slot. La referencia del bastidor a usar es **BMXXBP0400**.

La siguiente ilustración muestra el bastidor seleccionado

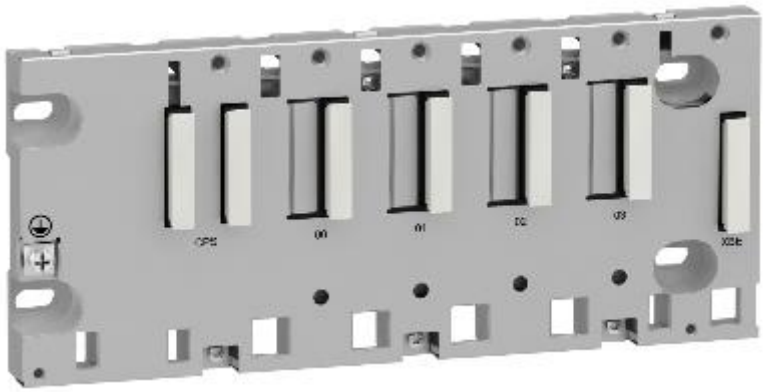


Figura 5-5 Rack de 4 posiciones

5.4. ELECCIÓN DE LA FUENTE DE ALIMENTACIÓN

La fuente de alimentación es un elemento que debe ir instalado en el bastidor y que como se ha comentado, proporciona las tensiones que necesita toda la electrónica que va instalada en el rack. En nuestro caso, como el único componente que irá instalado en el rack es la CPU, es suficiente con instalar aquella que entregue la menor potencia. En apartados posteriores se comprobará la potencia que es requerida o consumida. De los modelos que se presentan en la siguiente figura, seleccionaremos el modelo **BMXCPS2000**.

BMX CPS 2010/3020

BMX CPS 2000/3500

Red de alimentación	Potencias disponibles (1)				Referencia
	⎓ 3,3 V (2)	⎓ 24 V (2)	⎓ 24 V (3)	Total	
⎓ 24 V aislada	8,3 W	16,5 W	–	16,5 W	BMX CPS 2010 (4)
⎓ 24...48 V aislada	15 W	31,2 W	–	31,2 W	BMX CPS 3020 (4)
∿ 100...240 V	8,3 W	16,5 W	10,8 W	20 W	BMX CPS 2000 (4)
	15 W	31,2 W	21,6 W	36 W	BMX CPS 3500 (4)

Figura 5-6 Característica CPU BMXCPS2000

5.5. ELECCIÓN DE LA CPU

En los catálogos del fabricante, se ofrecen siete modelos diferentes de CPU M340. La diferencia entre ellos, para el caso que nos ocupa, está en los

diferentes tipos de buses de campo que pueden gestionar. De entre todos ellos, deberemos seleccionar uno que disponga de un puerto Ethernet Modbus/TCP.

En la tabla siguiente se observa que existen tres modelos que disponen de puertos Ethernet Modbus. De estos tres, dos disponen del bus de campo CAN que no va a ser utilizado en este proyecto. Por tanto, seleccionaremos para su uso el modelo **BMXP342020**.

Racks	Number of racks	4 (with 4, 6, 8 or 12 slots)				
	Max. number of slots (excluding power supply module)	48				
I/O	In-rack discrete I/O *(1)*	1024 channels (modules with 8, 16, 32 or 64 channels)				
	In-rack I/O *(1)*	256 channels (modules with 2, 4, 6 or 8 channels)				
	Distributed I/O (limited depending on the type of medium)	- On CANopen bus (63 devices), - On Ethernet Modbus/TCP via network module (63 devices with I/O Scanning function), - On Modbus link (32 devices).				
In-rack application-specific channels	No. of channels (counter, motion control, serial link)	36 max.				
	Counter *(1)*	BMXEHC0200 2-channel (60 kHz) or BMXEHC0800 8-channel (10 kHz) modules				
	Motion control *(1)*	BMXMSP0200 2-channel (200 kHz) PTO *(Pulse Train Output)* modules for servo drives				
		MFB (Motion Function Blocks) library (for drives or servo drives on CANopen bus)	–	MFB (Motion Function Blocks) library (for drives or servo drives on CANopen bus)		
	Serial link (process or RTU) *(1)*	BMXNOM0200 2-channel module or BMXNOR0200H module with 1 RTU serial channel				
	Process control, programmable loops	Process control EFB library				
Integrated communication ports	Ethernet Modbus/TCP network	–	1 x 10BASE-T/100BASE-TX (Modbus/TCP, BOOTP/DHCP, FDR client, e-mail notification, class B10 standard web server)			
	CANopen master bus	1 (63 slaves, 50…1000 Kbps, class M20) *(3)*	–	1 (63 slaves, 50…1000 Kbps, class M20) *(3)*		
	Serial link (process or RTU)	1 in RTU/ASCII Modbus master/slave mode or in Character mode (non-isolated RS232/RS485, 0.3…38.4 Kbps)		–		
	USB port	1 programming port (PC terminal) or HMI connection port				
Communication modules *(1)*	Ethernet network Max. no.	2				
	Type of module	BMXNOE0100/0110 or BMXNOC0401 network modules or BMXNOR0200H module with 1 Ethernet RTU channel				
	AS-Interface bus Max. no.	4				
	Type of module	BMXEIA0100 master module				
Internal memory capacity	Internal user RAM	4096 KB				
	Program, constants and symbols	3584 KB				
	Located/unlocated data	256 KB				
Memory card capacity (on processor)	Backup of program, constants and symbols	8 MB as standard	Supplied without card	8 MB as standard	Supplied without card	
	Hosting and display of user Web pages	*(2)*				
	File storage	8 or 128 MB (according to BMXRMS●●8MPF option card)				
Application structure	Master task	1				
	Fast task	1				
	Event tasks	64				
No. of K instructions executed per ms	100% Boolean	8.1 Kinstructions/ms				
	65% Boolean + 35% fixed arithmetic	6.4 Kinstructions/ms				
Rack power supply		24 V ⎓ isolated, 24…48 V ⎓ isolated or 100…240 V ∿ power supply module				
References		**BMXP3420102**	**BMXP3420102CL**	**BMXP342020**	**BMXP3420302**	**BMXP3420302CL**

Figura 5-7 Característica CPU BMXCPS2000

La siguiente imagen muestra los detalles de esta CPU. En ella, se observa que en su frontal dispone de un puerto USB, el cual, se utiliza para conexión directa a esta y mediante la cual podremos conectarnos para la carga y descarga del programa, un puerto Ethernet mediante el cual se conectará a campo y un puerto adicional serie/modbus el cual no será utilizado en este proyecto.

Figura 5-8 CPU BMXCPS2000

5.6. ELECCIÓN PANTALLA TÁCTIL

Para la presentación de información al usuario y que este pueda interactuar con el sistema de control, instalaremos una pantalla táctil que proporcione un medio simple y eficaz de recopilar datos y presentar su información en un formato entendible.

En los catálogos del fabricante estas pantallas son conocidas *terminales de dialogo de operador* y están especialmente diseñados para soportar aplicaciones relativamente sencillas donde el uso de una pantalla de mayores dimensiones seria de coste superior.

Los terminales gráficos **Magelis XBT GT** son de tipo táctil con una gama extensa en cuanto al tamaño de pantalla (**3,8", 5,7", 7,5", 10,4", 12,1" y 15"**) así como diferentes modelos (**monocroma, color, o TFT**). Las pantallas de dialogo de operador Magelis se pueden conectar a Ethernet TCP/IP con el protocolo Modbus TCP y protocolos de otros fabricantes.

Seleccionaremos una pantalla de 10,4” con una resolución de 640x480 pixeles y conexión Ethernet Modbus/TCP. La referencia seleccionada es **XBTGT5330**.

Sus características se muestran en las dos figuras que aparecen a continuación.

Especificación	XBT GT 4330 XBT GT 4340	XBT GT 5330 XBT GT 5340 XBT GK5330	XBT GT 5430	XBT GT 6330 XBT GT 6340	XBT GT 7340
Tipo	LCD en color TFT				
Resolución (píxeles)	640 x 480		800 x 600	800 x 600	1.024 x 768
Área de visualización activa Ancho x Alto	153,7 x 115,8 mm (6.05 x 4.56 in)	211,2 x 158,4 mm (8.31 x 6.24 in)		248 x 186,5 mm (9.76 x 7.34 in)	306,2 x 230,1 mm (12.06 x 9.06 in)
Colores	65.536 colores				
Retroiluminación (1)	Retroiluminación CFL (vida útil: 54.000 h a 25 °C y funcionamiento continuo [a la mitad del brillo original])	Retroiluminación CFL (vida útil: 50.000 h a 25 °C y funcionamiento continuo [a la mitad del brillo original])		XBTGT6330 RL 10 o superior y XBTGT6340 RL 09 o superior: retroiluminación LED, XBTGT6330 RL 9 o inferior y XBTGT6340 RL 08 o inferior: retroiluminación CFL (vida útil: 50.000 h a 25 °C y funcionamiento continuo [a la mitad del brillo original])	RL 09 o superior: retroiluminación LED, RL 08 o inferior: retroiluminación CFL (vida útil: 50.000 h a 25 °C y funcionamiento continuo [a la mitad del brillo original])
Ajuste del contraste	8 niveles de ajuste disponible mediante panel táctil	No disponible.			
Ajuste de brillo	8 niveles de ajuste disponibles mediante panel táctil.			16 niveles de ajuste disponible mediante panel táctil.	
Fuentes de idiomas incorporadas en el sistema (2)	ASCII: (página de códigos 850) alfanumérico (incluidos caracteres europeos) chino: (códigos GB2312-80), fuentes de chino simplificado, coreano: (códigos KSC5601 - 1992), fuentes Hangul taiwanesas: (códigos Big 5), fuentes de chino tradicional				
Tamaño de caracteres (2)	Fuentes de 8 x 8, 8 x 16, 16 x 16 y 32 x 32 píxeles				
Tamaños de fuente	El ancho se puede ampliar de 1 a 8 veces. El alto se puede reducir a la mitad y ampliar entre 1 y 8 veces.				
8 x 8 píxeles	80 caracteres por fila por 60 filas		100 caracteres por fila por 75 filas		128 caracteres por fila por 96 filas

Figura 5-9 Características Magelis

Memoria	XBT GT2430	XBT GT4230 XBT GT4330 XBT GT5230 XBT GT5330 XBT GT5430 XBT GT6330 XBT GK5330	XBT GT4340 XBT GT5340 XBT GT6340 XBT GT7340 XBT GH2460
Flash EPROM de aplicación	32 MB	32 MB	32 MB
La memoria SRAM de copia de seguridad de datos utiliza batería de litio (1)	512 kB	512 kB	512 kB

Figura 5-10 Características Magelis

Memoria	XBT GT2430	XBT GT4230 XBT GT4330 XBT GT5230 XBT GT5330 XBT GT5430 XBT GT6330 XBT GK5330	XBT GT4340 XBT GT5340 XBT GT6340 XBT GT7340 XBT GH2460
DRAM de ejecución de aplicaciones	64 MB	32 MB	64 MB
Leyenda: (1) La vida útil de una batería de litio es de: • 10 años cuando la temperatura ambiente de la batería es ≤ 40 °C (104 °F). • 10 años cuando la temperatura ambiente de la unidad es ≤ 25 °C (77 °F). Cuando se utiliza para copias de seguridad (sin alimentación principal): • Aproximadamente 60 días con una batería cargada completamente. • Aproximadamente 6 días con una batería cargada al 10 %.			

Figura 5-11 Características Magelis

5.7. DISPOSICIÓN FINAL DE ELEMENTOS.

Con los descrito en apartados anteriores, la instalación en la vivienda es la que se muestra a continuación.

En la planta 0 se situará una isla que recogerá las entradas digitales y salidas digitales de los sensores de movimiento, persianas, ascensor y puertas automáticas (exterior y acceso a vivienda). Además, en esta planta irá situado el PLC en un armario junto con el switch de infraestructura de red.

En la Planta 1 se situará una isla que recogerá las entradas digitales y salidas digitales de los sensores de movimiento, persianas y ascensor. Además, en esta planta irá situado el terminal táctil.

En la Planta 2 se situará una isla que recogerá las entradas digitales y salidas digitales de los sensores de movimiento, persianas y ascensor.

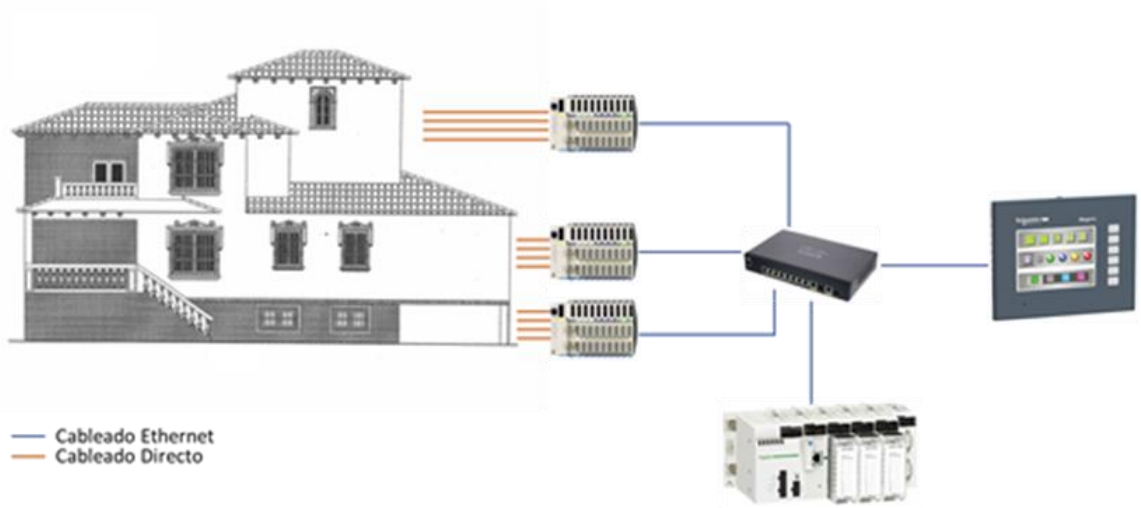


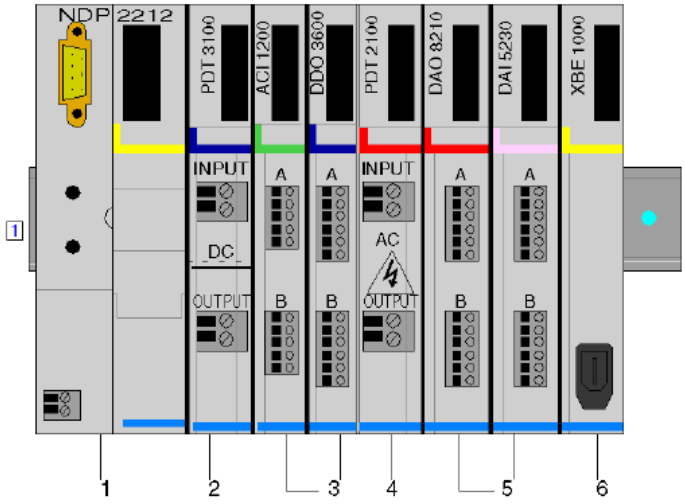
Figura 5-12 Disposición elementos en la vivienda

6. DIMENSIONAMIENTO Y CONFIGURACIÓN DE LAS ISLAS ADVANTYS

6.1. ¿QUÉ ES UNA ISLA ADVANTYS?

Una isla es un conjunto de módulos de E/S distribuidas, módulos de distribución de alimentación y de comunicación/extensión del bus de la isla que funcionan juntos como un nodo de un bus de campo. En función del tipo y de la familia de productos, una isla puede contener hasta 32 módulos de E/S más un NIM. Las islas, además, requieren uno o varios módulos de distribución de alimentación (PDM) y también, ofrecen módulos que permiten ampliar el bus a varios segmentos de módulos E/S Advantys..

En la ilustración siguiente se muestra un ejemplo de un segmento en una isla física.



1 NIM
2 PDM
3 Grupo de tensión de módulos de E/S
4 Segundo PDM que admite el segundo grupo de módulos de E/S (5)
5 Segundo grupo de módulos de E/S que requieren una tensión de alimentación de campo diferente o una corriente adicional
6 Módulo EOS para la extensión de la isla física a otro segmento de los módulos de E/S Advantys o a un módulo totalmente compatible (preferido)

Figura 6-1 Ejemplo segmento Isla

6.2. DESCRIPCIÓN DE COMPONENTES

NIM (Network Interface Module): Cada isla requiere un módulo de interfaz de red situado en la parte más a la izquierda de ella. Funcionalmente es una pasarela entre el bus interno de la isla y el exterior (bus de campo). Gestiona el intercambio de datos de entrada y salida entre el bus de campo y la isla. Posee un web Server embebido que permite su configuración y diagnóstico.

El NIM utilizado en este proyecto es el modelo **NIM2212** que además de lo anteriormente descrito, implementa el protocolo Modbus servidor que corre sobre TCP/IP. Usa el puerto 502 y es el puerto well-known asignado a Schneider Electric por la Autoridad de Internet (IANA).

Los detalles se encuentran en la siguiente figura:

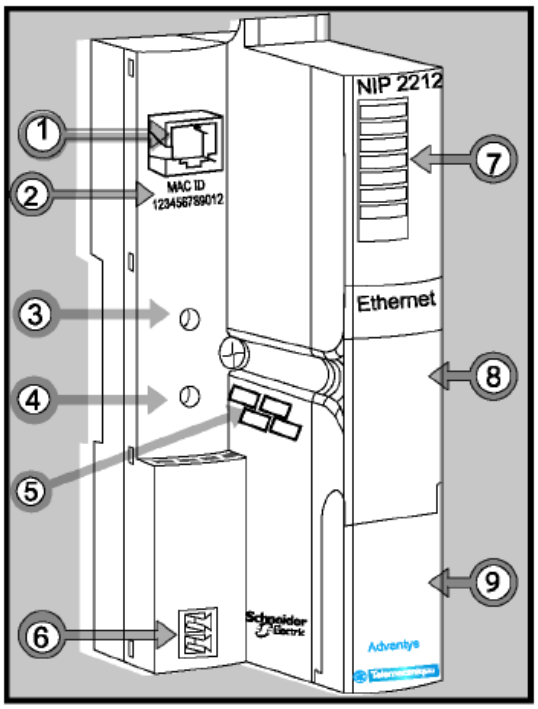


Figura 6-2 NIM2212

Feature		Function
1	Ethernet interface	An RJ-45 (See *STB NIP 2212 Network Interface, p. 26*) connector is used to connect the NIM and the island bus to an Ethernet LAN network.
2	MAC ID	48-bit, unique network ID hard-coded in the STB NIP 2212 when manufactured.
3	upper rotary switch	The rotary switches (See *Physical Description, p. 28*) used together specify a role name for the STB NIP 2212. Alternatively, the lower rotary switch can be used to direct the STB NIP 2212 to use its MAC-based default IP address (See *Summary of Valid IP Address Settings, p. 29*) or to obtain its IP parameters from a BootP server or from the STB NIP 2212 web site (See *About the Embedded Web Server, p. 67*).
4	lower rotary switch	
5	space provided to record IP address	Write the IP address that you assign to this STB NIP 2212 here.
6	power supply interface	A two-pin connector used to connect an external 24 VDC power supply (See *Selecting a Source Power Supply for the Island's Logic Power Bus, p. 39*) to the NIM.
7	LED array	Colored LEDs (See *LED Indicators, p. 30*) use various patterns to visually indicate the operational status of the island bus, activity on the NIM, and the status of communications to the island over the Ethernet LAN.
8	removable memory card drawer	A plastic drawer in which a removable memory card (See *Installing the STB XMP 4440 Optional Removable Memory Card, p. 50*) can be seated and then inserted into the NIM.
9	CFG port cover	A hinged flap on the NIM's front panel that covers the CFG interface (See *The CFG Interface, p. 33*) and the RST button (See *The RST Button, p. 55*).

Figura 6-3 NIM2212

PDT: Es un módulo usado para la distribución de alimentación de sensores y actuadores. Con una tensión nominal de funcionamiento de 24Vcc con protección de sobreintensidad. Proporciona alimentación separada para sensores de entrada y actuadores de salida. La referencia utilizada es **PDT3100**. Esta se muestra en la siguiente figura.

Figura 6-4 PDT3100

Como se indicó en apartados previos, para el proyecto, usaremos una isla Advantys por cada una de las plantas. A cada una de estas islas se conectarán los sensores y actuadores de ellas.

Módulos de Entradas digitales: Son los módulos encargados de recoger de campo las señales digitales (ON/OFF). En la siguiente figura se muestra los diferentes tipos que ofrece el fabricante. Observamos dos grandes grupos: los que trabajan con corriente alterna (115Vac o 220Vac) y los que

trabajan con corriente continua de 24Vcc. También se observa módulos de diferentes densidades 2, 4, 6 y 16 puntos de media. Para este proyecto se seleccionarán módulos de 24Vcc (tensión más segura) y densidad 16 puntos. La referencia concreta a utilizar es **STBDI3725**.

La siguiente ilustración muestra las características de los módulos que existen.

Módulos de entrada digital				
Modelo	**Tipo**	**Consumo de corriente del bus lógico a rangos de temperaturas de funcionamiento**		
		De -25 a 0 °C	**De 0 a 60 °C**	**De 60 a 70 °C**
STB DAI 5230	115 V CA, 2 pt, 3 cables estándar	No	40 mA	No
STB DAI 5260	115 V CA aislado, estándar	No	45 mA	No
STB DAI 7220	250 V CA, 2 pt, 3 cables, estándar	No	40 mA	No
STB DDI 3230	24 V CC, arqueta de 2 pt, 4 cables estándar	55 mA	55 mA	55 mA
STB DDI 3420	24 V CC, arqueta de 2 pt, 3 cables estándar	45 mA	45 mA	45 mA
STBDDI 3425	24 V CC, arqueta de 4 pt, 3 cables básico	No	45 mA	No
STB DDI 3610	24 V CC, arqueta de 6 pt, 2 cables estándar	55 mA	55 mA	55 mA
STB DDI 3615	24 V CC, arqueta de 6 pt, 2 cables básico	No	45 mA	No
STB DDI 3725	24 V CC, arqueta de 16 pt, 2 cables básico	100 mA	100 mA	100 mA

Figura 6-5 Tipos de Módulos entradas digitales

En la siguiente ilustración se muestra el diagrama de conexionado de los sensores al módulo de entradas digitales a utilizar.

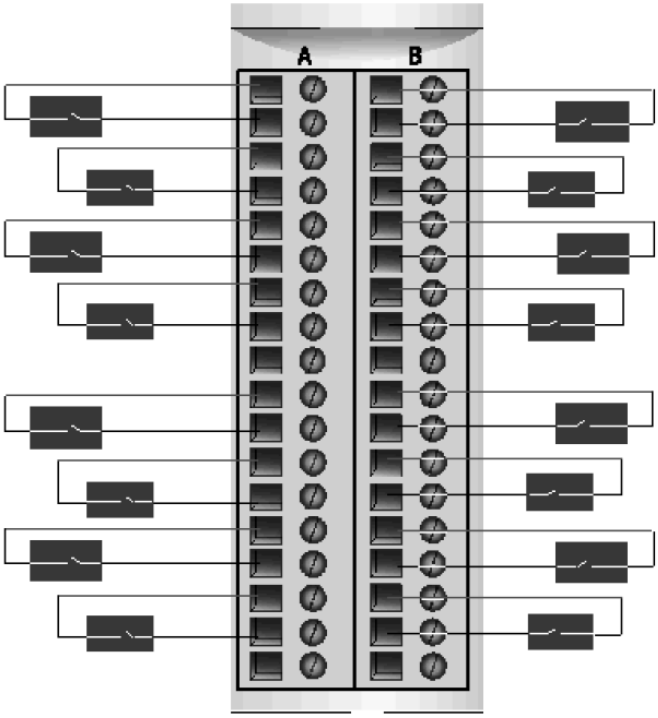


Figura 6-6 Diagrama conexionado entradas digitales

La siguiente figura muestra el aspecto físico del módulo seleccionado

Figura 6-7 STBDI3725

Módulos de Salidas digitales: Son los módulos encargados de actuar sobre los elementos de campo, señales digitales de salida (ON/OFF). En la siguiente figura se muestra los diferentes tipos módulos de salidas digitales que ofrece el fabricante. Observamos, al igual que en los módulos de entradas digitales, dos grandes grupos: los que trabajan con corriente alterna (115Vac o 220Vac) y los que trabajan con corriente continua de 24Vcc. También se observa módulos de diferentes densidades 2, 4, 6 y 16 puntos de media. Para este proyecto se seleccionarán módulos de 24Vcc (tensión más segura) y densidad 16 puntos. La referencia concreta a utilizar es **STBDO3600**.

La siguiente ilustración muestra las características de los módulos que existen.

Módulos de salida digitales				
Modelo	**Tipo**	**Consumo de corriente del bus lógico a rangos de temperaturas de funcionamiento**		
		De -25 a 0 °C	**De 0 a 60 °C**	**De 60 a 70 °C**
STB DAO 5260	115 V CA aislado, estándar	No	70 mA	No
STB DAO 8210	115/230 V CA, fuente de 2 pt, 2,0 A, estándar	No	45 mA	No
STB DDO 3200	24 V CC, fuente de 2 pt, 0,5 A, estándar	50 mA	50 mA	50 mA
STB DDO 3230	24 V CC, fuente de 2 pt, 0,2 A, estándar	45 mA	45 mA	45 mA
STB DDO 3410	24 V CC, fuente de 4 pt, 0,5 A, estándar	70 mA	70 mA	70 mA
STB DDO 3415	24 V CC, fuente de 4 pt, 0,25 A, básico	No	70 mA	No
STB DDO 3600	24 V CC, fuente de 6 pt, 0,5 A, estándar	90 mA	90 mA	90 mA
STB DDO 3605	24 V CC, fuente de 6 pt, 0,25 A, básico	No	90 mA	No
STB DDO 3705	24 V CC, fuente de16 pt, 0,5 A, básico	135 mA	135 mA	135 mA
STB DRC 3210	Relé, 2 pt, 2,0 A, estándar	55 mA	55 mA	55 mA, consulte la Nota 1

Figura 6-8 Tipos de Módulos salidas digitales

En la siguiente ilustración se muestra el diagrama de conexionado de los sensores al módulo de salidas digitales a utilizar.

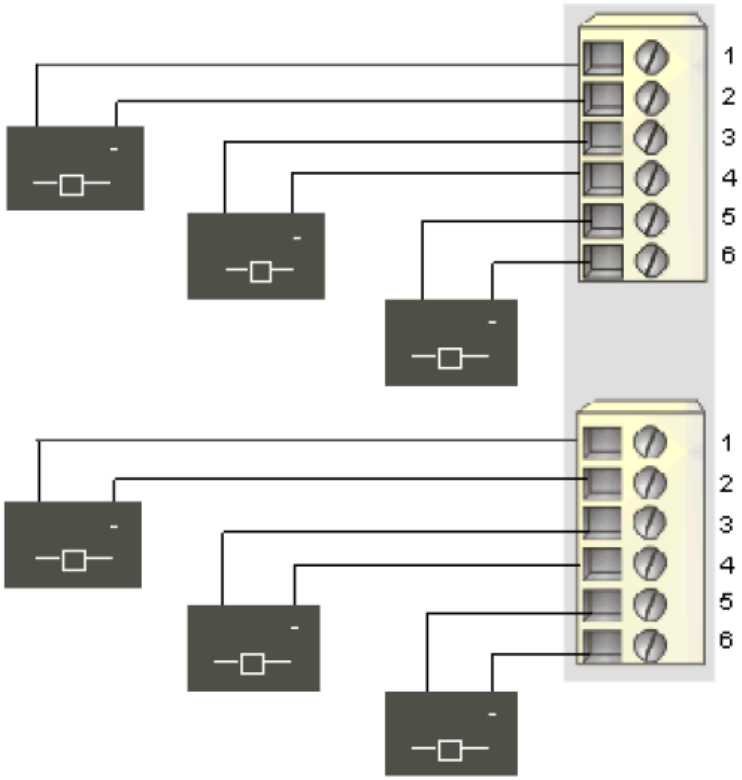


Figura 6-9 Diagrama conexionado salidas digitales

La siguiente figura muestra el aspecto físico del módulo seleccionado.

Figura 6-10 STBDO3600

6.3. DIMENSIONAMIENTO DE MÓDULOS DE E/S DIGITALES

Previo al dimensionamiento o elección de la cantidad de módulos que va a llevar cada una de las tres islas que utilizaremos, es necesario tener una relación de las señales de entrada-salida que tendrán ellas en cada una de las plantas de la vivienda. En la siguiente tabla se encuentra una relación de señales en cada isla con una descripción de su función, el sistema al que pertenecen, su tipo (E/S), módulo al que se conecta, pines asociados y la isla asociada.

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
Indicación de puerta abierta	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A1, A2
Indicación de puerta cerrada	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A3, A4

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
Haz de fotocélula interrumpido	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A5, A6
Indicación de sobretemperatura en motor puerta	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A7, A8
Pulsador de abrir	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A10, A11
Pulsador de cerrar	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A12, A13
Contacto procedente de llave para evitar actuar sobre la puerta	Puerta deslizante exterior	E	Adts_Planta_0	ED_1	A14, A15
Indicación de puerta abierta	Puerta garaje	E	Adts_Planta_0	ED_1	A16, A17
Indicación de puerta cerrada	Puerta garaje	E	Adts_Planta_0	ED_1	B1, B2
Haz de fotocélula interrumpido	Puerta garaje	E	Adts_Planta_0	ED_1	B3, B4
Indicación de sobretemperatura en motor puerta	Puerta garaje	E	Adts_Planta_0	ED_1	B5, B6
Pulsador de abrir	Puerta garaje	E	Adts_Planta_0	ED_1	B7, B8
Pulsador de cerrar	Puerta garaje	E	Adts_Planta_0	ED_1	B10, B11
Contacto procedente de llave para evitar actuar sobre la puerta	Puerta garaje	E	Adts_Planta_0	ED_1	B12, B13
Indicación persiana totalmente subida	Persiana 1 Planta 0	E	Adts_Planta_0	ED_1	B14, B15
Indicación persiana totalmente bajada	Persiana 1 Planta 0	E	Adts_Planta_0	ED_1	B16, B17
Pulsador de subir persiana	Persiana 1 Planta 0	E	Adts_Planta_0	ED_2	A1, A2
Pulsador de bajar persiana	Persiana 1 Planta 0	E	Adts_Planta_0	ED_2	A3, A4
Indicación persiana totalmente subida	Persiana 2 Planta 0	E	Adts_Planta_0	ED_2	A5, A6
Indicación persiana totalmente bajada	Persiana 2 Planta 0	E	Adts_Planta_0	ED_2	A7, A8
Pulsador de subir	Persiana 2	E	Adts_Planta_0	ED_2	A10, A11

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
persiana	Planta 0				
Pulsador de bajar persiana	Persiana 2 Planta 0	E	Adts_Planta_0	ED_2	A12, A13
Indicación persiana totalmente subida	Persiana 3 Planta 0	E	Adts_Planta_0	ED_2	A14, A15
Indicación persiana totalmente bajada	Persiana 3 Planta 0	E	Adts_Planta_0	ED_2	A16, A17
Pulsador de subir persiana	Persiana 3 Planta 0	E	Adts_Planta_0	ED_2	B1, B2
Pulsador de bajar persiana	Persiana 3 Planta 0	E	Adts_Planta_0	ED_2	B3, B4
Indicación persiana totalmente subida	Persiana 4 Planta 0	E	Adts_Planta_0	ED_2	B5, B6
Indicación persiana totalmente bajada	Persiana 4 Planta 0	E	Adts_Planta_0	ED_2	B7, B8
Pulsador de subir persiana	Persiana 4 Planta 0	E	Adts_Planta_0	ED_2	B10, B11
Pulsador de bajar persiana	Persiana 4 Planta 0	E	Adts_Planta_0	ED_2	B12, B13
Indicación persiana totalmente subida	Persiana 5 Planta 0	E	Adts_Planta_0	ED_2	B14, B15
Indicación persiana totalmente bajada	Persiana 5 Planta 0	E	Adts_Planta_0	ED_2	B16, B17
Pulsador de subir persiana	Persiana 5 Planta 0	E	Adts_Planta_0	ED_3	A1, A2
Pulsador de bajar persiana	Persiana 5 Planta 0	E	Adts_Planta_0	ED_3	A3, A4
Entrada de movimiento al sistema	Sensor1 alarma Planta 0	E	Adts_Planta_0	ED_3	A5, A6
Entrada de movimiento al sistema	Sensor2 alarma Planta 0	E	Adts_Planta_0	ED_3	A7, A8
Fin de carrera planta 0	Ascensor	E	Adts_Planta_0	ED_3	A10, A11
Indicación de puerta abierta de la planta 0	Ascensor	E	Adts_Planta_0	ED_3	A12, A13
Botón para planta 0	Ascensor	E	Adts_Planta_0	ED_3	A14, A15
Botón para planta 1	Ascensor	E	Adts_Planta_0	ED_3	A16, A17
Botón para planta 2	Ascensor	E	Adts_Planta_0	ED_3	B1, B2

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
Indicación persiana totalmente subida	Persiana 1 Cocina Planta 1	E	Adts_Planta_1	ED_1	A1, A2
Indicación persiana totalmente bajada	Persiana 1 Cocina Planta 1	E	Adts_Planta_1	ED_1	A3, A4
Pulsador de subir persiana	Persiana 1 Cocina Planta 1	E	Adts_Planta_1	ED_1	A5, A6
Pulsador de bajar persiana	Persiana 1 Cocina Planta 1	E	Adts_Planta_1	ED_1	A7, A8
Indicación persiana totalmente subida	Persiana 1 Salón Planta 1	E	Adts_Planta_1	ED_1	A10, A11
Indicación persiana totalmente bajada	Persiana 1 Salón Planta 1	E	Adts_Planta_1	ED_1	A12, A13
Pulsador de subir persiana	Persiana 1 Salón Planta 1	E	Adts_Planta_1	ED_1	A14, A15
Pulsador de bajar persiana	Persiana 1 Salón Planta 1	E	Adts_Planta_1	ED_1	A16, A17
Indicación persiana totalmente subida	Persiana habitación 1 Planta 1	E	Adts_Planta_1	ED_1	B1, B2
Indicación persiana totalmente bajada	Persiana habitación 1 Planta 1	E	Adts_Planta_1	ED_1	B3, B4
Pulsador de subir persiana	Persiana habitación 1 Planta 1	E	Adts_Planta_1	ED_1	B5, B6
Pulsador de bajar persiana	Persiana habitación 1 Planta 1	E	Adts_Planta_1	ED_1	B7, B8
Indicación persiana totalmente subida	Persiana habitación 2 Planta 1	E	Adts_Planta_1	ED_1	B10, B11
Indicación persiana totalmente bajada	Persiana habitación 2 Planta 1	E	Adts_Planta_1	ED_1	B12, B13
Pulsador de subir persiana	Persiana habitación 2 Planta 1	E	Adts_Planta_1	ED_1	B14, B15
Pulsador de bajar persiana	Persiana habitación 2 Planta 1	E	Adts_Planta_1	ED_1	B16, B17
Indicación	Persiana	E	Adts_Planta_1	ED_2	A1, A2

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
persiana totalmente subida	habitación 3 Planta 1				
Indicación persiana totalmente bajada	Persiana habitación 3 Planta 1	E	Adts_Planta_1	ED_2	A3, A4
Pulsador de subir persiana	Persiana habitación 3 Planta 1	E	Adts_Planta_1	ED_2	A5, A6
Pulsador de bajar persiana	Persiana habitación 3 Planta 1	E	Adts_Planta_1	ED_2	A7, A8
Entrada de movimiento al sistema	Sensor Salón alarma Planta 1	E	Adts_Planta_1	ED_2	A10, A11
Entrada de movimiento al sistema	Sensor pasillo alarma Planta 1	E	Adts_Planta_1	ED_2	A12, A13
Entrada de movimiento al sistema	Sensor cocina alarma Planta 1	E	Adts_Planta_1	ED_2	A14, A15
Entrada de movimiento al sistema	Sensor habitación1 alarma Planta 1	E	Adts_Planta_1	ED_2	A16, A17
Entrada de movimiento al sistema	Sensor habitación2 alarma Planta 1	E	Adts_Planta_1	ED_2	B1, B2
Entrada de movimiento al sistema	Sensor habitación3 alarma Planta 1	E	Adts_Planta_1	ED_2	B3, B4
Fin de carrera planta 1	Ascensor	E	Adts_Planta_1	ED_2	B5, B6
Indicación de puerta abierta de la planta 1	Ascensor	E	Adts_Planta_1	ED_2	B7, B8
Indicación persiana totalmente subida	Persiana1 ático Planta 2	E	Adts_Planta_2	ED_1	A1, A2
Indicación persiana totalmente bajada	Persiana1 ático Planta 2	E	Adts_Planta_2	ED_1	A3, A4
Pulsador de subir persiana	Persiana1 ático Planta 2	E	Adts_Planta_2	ED_1	A5, A6
Pulsador de bajar persiana	Persiana1 ático Planta 2	E	Adts_Planta_2	ED_1	A7, A8
Indicación persiana totalmente subida	Persiana2 ático Planta 2	E	Adts_Planta_2	ED_1	A10, A11
Indicación	Persiana2	E	Adts_Planta_2	ED_1	A12, A13

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
persiana totalmente bajada	ático Planta 2				
Pulsador de subir persiana	Persiana2 ático Planta 2	E	Adts_Planta_2	ED_1	A14, A15
Pulsador de bajar persiana	Persiana2 ático Planta 2	E	Adts_Planta_2	ED_1	A16, A17
Indicación persiana totalmente subida	Persiana 4 ático Planta 2	E	Adts_Planta_2	ED_1	B1, B2
Indicación persiana totalmente bajada	Persiana 4 ático Planta 2	E	Adts_Planta_2	ED_1	B3, B4
Pulsador de subir persiana	Persiana 4 ático Planta 2	E	Adts_Planta_2	ED_1	B5, B6
Pulsador de bajar persiana	Persiana 4 ático Planta 2	E	Adts_Planta_2	ED_1	B7, B8
Indicación persiana totalmente subida	Persiana 3 Ático Planta2	E	Adts_Planta_2	ED_1	B10, B11
Indicación persiana totalmente bajada	Persiana 3 Ático Planta2	E	Adts_Planta_2	ED_1	B12, B13
Pulsador de subir persiana	Persiana 3 Ático Planta2	E	Adts_Planta_2	ED_1	B14, B15
Pulsador de bajar persiana	Persiana 3 Ático Planta2	E	Adts_Planta_2	ED_1	B16, B17
Entrada de movimiento al sistema	Sensor ático alarma Planta 2	E	Adts_Planta_2	ED_2	A1, A2
Fin de carrera planta 2	Ascensor	E	Adts_Planta_2	ED_2	A3, A4
Indicación de puerta abierta de la planta 2	Ascensor	E	Adts_Planta_2	ED_2	A5, A6
Salida hacia motor puerta para girar en el sentido de apertura	Puerta deslizante exterior	S	Adts_Planta_0	SD_1	A1, A2
Salida hacia motor puerta para girar en el sentido de cierre	Puerta deslizante exterior	S	Adts_Planta_0	SD_1	A3, A4
Salida hacia luz rotativa para indicar movimiento de la puerta	Puerta deslizante exterior	S	Adts_Planta_0	SD_1	A5, A6
Salida hacia motor puerta	Puerta garaje	S	Adts_Planta_0	SD_1	B1, B2

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
para girar en el sentido de apertura					
Salida hacia motor puerta para girar en el sentido de cierre	Puerta garaje	S	Adts_Planta_0	SD_1	B3, B4
Salida hacia luz rotativa para indicar movimiento de la puerta	Puerta garaje	S	Adts_Planta_0	SD_1	B5, B6
Salida hacia motor persiana subir	Persiana 1 Planta 0	S	Adts_Planta_0	SD_2	A1, A2
Salida hacia motor persiana bajar	Persiana 1 Planta 0	S	Adts_Planta_0	SD_2	A3, A4
Salida hacia motor persiana subir	Persiana 2 Planta 0	S	Adts_Planta_0	SD_2	A5, A6
Salida hacia motor persiana bajar	Persiana 2 Planta 0	S	Adts_Planta_0	SD_2	B1, B2
Salida hacia motor persiana subir	Persiana 3 Planta 0	S	Adts_Planta_0	SD_2	B3, B4
Salida hacia motor persiana bajar	Persiana 3 Planta 0	S	Adts_Planta_0	SD_2	B5, B6
Salida hacia motor persiana subir	Persiana 4 Planta 0	S	Adts_Planta_0	SD_3	A1, A2
Salida hacia motor persiana bajar	Persiana 4 Planta 0	S	Adts_Planta_0	SD_3	A3, A4
Salida hacia motor persiana subir	Persiana 5 Planta 0	S	Adts_Planta_0	SD_3	A5, A6
Salida hacia motor persiana bajar	Persiana 5 Planta 0	S	Adts_Planta_0	SD_3	B1, B2
Salida para accionar la subida del ascensor	Ascensor	S	Adts_Planta_0	SD_3	B3, B4
Salida para accionar la bajada del ascensor	Ascensor	S	Adts_Planta_0	SD_3	B5, B6
Salida para accionar el cerrojo para la	Ascensor	S	Adts_Planta_0	SD_4	A1, A2

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
puerta de la planta 0					
Salida hacia motor persiana subir	Persiana 1 Cocina Planta 1	S	Adts_Planta_1	SD_1	A1, A2
Salida hacia motor persiana bajar	Persiana 1 Cocina Planta 1	S	Adts_Planta_1	SD_1	A3, A4
Salida hacia motor persiana subir	Persiana 1 Salón Planta 1	S	Adts_Planta_1	SD_1	A5, A6
Salida hacia motor persiana bajar	Persiana 1 Salón Planta 1	S	Adts_Planta_1	SD_1	B1, B2
Salida hacia motor persiana subir	Persiana habitación 1 Planta 1	S	Adts_Planta_1	SD_1	B3, B4
Salida hacia motor persiana bajar	Persiana habitación 1 Planta 1	S	Adts_Planta_1	SD_1	B5, B6
Salida hacia motor persiana subir	Persiana habitación 2 Planta 1	S	Adts_Planta_1	SD_2	A1, A2
Salida hacia motor persiana bajar	Persiana habitación 2 Planta 1	S	Adts_Planta_1	SD_2	A3, A4
Salida hacia motor persiana subir	Persiana habitación 3 Planta 1	S	Adts_Planta_1	SD_2	A5, A6
Salida hacia motor persiana bajar	Persiana habitación 3 Planta 1	S	Adts_Planta_1	SD_2	B1, B2
Salida para accionar el cerrojo para la puerta de la planta 1	Ascensor	S	Adts_Planta_1	SD_2	B3, B4
Salida sirena	sistema alarma	S	Adts_Planta_1	SD_2	B5, B6
Salida hacia motor persiana subir	Persiana1 ático Planta 2	S	Adts_Planta_2	SD_1	A1, A2
Salida hacia motor persiana bajar	Persiana1 ático Planta 2	S	Adts_Planta_2	SD_1	A3, A4
Salida hacia motor persiana subir	Persiana2 ático Planta 2	S	Adts_Planta_2	SD_1	A5, A6
Salida hacia motor persiana bajar	Persiana2 ático Planta 2	S	Adts_Planta_2	SD_1	B1, B2
Salida hacia motor persiana	Persiana 4 ático Planta 2	S	Adts_Planta_2	SD_1	B3, B4

DESCRIPCION	SISTEMA	E/S	ASIGNACION	MODULO	PINES
subir					
Salida hacia motor persiana bajar	Persiana 4 ático Planta 2	S	Adts_Planta_2	SD_1	B5, B6
Salida hacia motor persiana subir	Persiana 3 Ático Planta2	S	Adts_Planta_2	SD_2	A1, A2
Salida hacia motor persiana bajar	Persiana 3 Ático Planta2	S	Adts_Planta_2	SD_2	A3, A4
Salida para accionar el cerrojo para la puerta de la planta 2	Ascensor	S	Adts_Planta_2	SD_2	A5, A6

Tabla 6-1 Señales asignadas a cada isla

6.3.1. DIMENSIONAMIENTO DE ENTRADAS DIGITALES

Teniendo en cuenta que los módulos de entradas digitales tienen una densidad de 16 puntos de medida, tendremos:

- Para la isla ubicada en planta 0, 3 módulos de 16 entradas digitales referencia STBDDI3725.
- Para la isla ubicada en planta 1, 2 módulos de 16 entradas digitales referencia STBDDI3725.
- Para la isla ubicada en planta 3, 2 módulos de 16 entradas digitales referencia STBDDI3725.

6.3.2. DIMENSIONAMIENTO DE SALIDAS DIGITALES

Teniendo en cuenta que los módulos de salidas digitales tienen una densidad de 6 puntos de medida, tendremos:

- Para la isla ubicada en planta 0, 3 módulos de 6 entradas digitales referencia STBDD03600.
- Para la isla ubicada en planta 1, 2 módulos de 6 entradas digitales referencia STBDD03600.
- Para la isla ubicada en planta 3, 2 módulos de 6 entradas digitales referencia STBDD03600.

6.4. CONFIGURACIÓN DIRECCIÓN IP DE LAS ISLAS ADVANTYS

Como nodo en una red TCP/IP, el módulo STBNIM2212 requiere una dirección IP de 32 bits válida. La dirección IP puede ser:

- La dirección IP basada en MAC
- Una dirección IP asignada por un servidor de red DHCP (la CPU del PLC)
- Una dirección IP configurada mediante el servidor web del módulo.

En nuestro caso, usaremos la opción de configurar la dirección IP mediante el servidor web embebido en el módulo STBNIM2212. Ello requiere previamente acceder a este servidor mediante una dirección IP por defecto que llevan los módulos y que está basada en la MAC. Para conseguir esta, es necesario colocar los conmutadores rotativos que llevan en el frontal en la posición INTERNAL como se muestra en la siguiente figura, después darle alimentación y esperar, al menos, dos minutos con el cable de red desconectado:

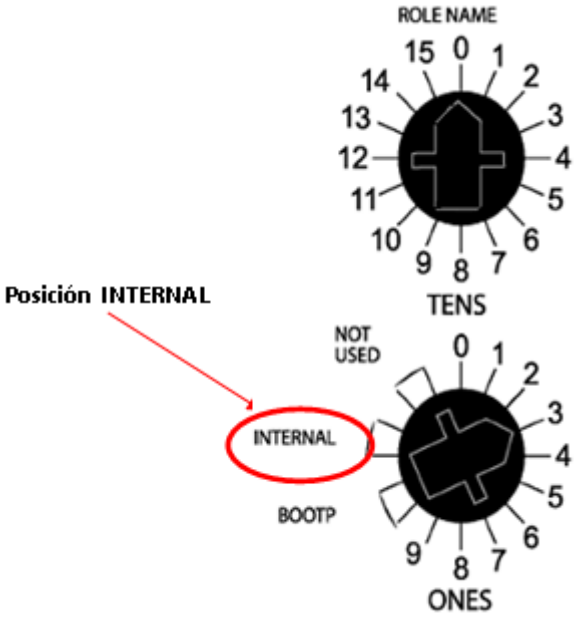


Figura 6-11 Selectores rotativos NIM

La dirección MAC para un STBNIP2212 se encuentra en el marco frontal debajo del puerto Ethernet como se muestra en la siguiente ilustración

Figura 6-12 Situación MAC en NIM

Una vez que conocemos la dirección MAC, podremos determinar la dirección IP por defecto del módulo como se muestra en el siguiente ejemplo: Supongamos que la dirección MAC fuese 00 00 **54 10 2D 11**, la dirección IP por defecto seria la siguiente:

Primer par 54: 5 x 16 = 80 + 4 = 84

Segundo par 10: 1 x 16 = 16 + 0 = 16

Tercer par 2D: 2 x 16 = 32 + 13 = 45

Cuarto par 11: 1 x 16 = 16 + 1 = 17

Hemos obtenido la dirección IP predeterminada o por defecto 84.16.45.17

Con la dirección IP por defecto obtenida, ya podremos acceder al web server y cambiar la dirección IP por la que necesitemos usar. Introduciremos esta dirección IP en cualquier navegador web y nos aparecerá la siguiente imagen:

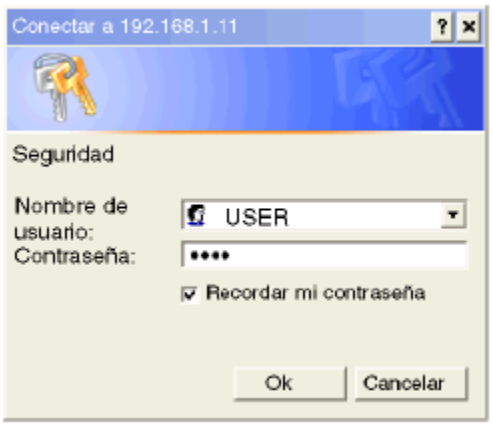


Figura 6-13 Ingreso en Web Server

Al introducir como contraseña USER, tendremos acceso al web server y cambiaremos la dirección IP por la deseada, como se indica en la siguiente figura.

Figura 6-14 Cambio IP

Tras guardar los parámetros deseados, ya tenemos configurado el módulo NIM.

A través de la consola de comandos de Windows, y haciendo un simple ping a la dirección IP que hemos configurado, podremos saber si el proceso de configuración ha sido correcto. Es decir, comprobamos que existe conectividad.

6.4.1. DIRECCIONES IP UTILIZADAS EN LAS ISLAS ADVANTYS

Las direcciones IP que vamos a utilizar en las islas son las que se indican a continuación y son configuradas como se ha indicado en el apartado anterior de esta memoria:

Para la isla situada en planta 0:

- Dirección IP: 100.100.1.2
- Mascara de subred: 255.255.0.0

Para la isla situada en planta 1:

- Dirección IP: 100.100.1.3
- Mascara de subred: 255.255.0.0

Para la isla situada en planta 2:

- Dirección IP: 100.100.1.3
- Mascara de subred: 255.255.0.0

7. CONFIGURACIÓN Y DESARROLLO DEL SOFTWARE DEL PLC

7.1. INTRODUCCIÓN AL ENTORNO DE DESARROLLO

Unity es un entorno o plataforma de desarrollo sw, puesta a punto y explotación de los autómatas de Schneider Electric.

7.1.1. LA INTERFAZ DE USUARIO

La interfaz de usuario que ofrece Unity se realiza a través de varias ventanas configurables y barras de herramientas como se indica en la siguiente figura.

Figura 7-1 Detalles interfaz UNITY

Menús y Barra de menú: Todas las funciones pueden ser accesibles usando la barra de menú. Las funciones que se usan más frecuentemente son accesibles directamente usando los iconos de la barra de herramientas estándar.

Por otro lado, se pueden personalizar las funciones que se presentan en la barra de herramientas para adaptarlas a las necesidades concretas del programador.

Explorador de proyectos: El explorador de proyectos permite desplegar las distintas en las que está dividido un proyecto Unity y navegar por ellas.

Configuración: Permite configurar el rack o bastidor con los diferentes módulos que van a ser instalados en el PLC del proyecto.

Editor de datos: Permite la creación de variables, asignarles nombres, definir su tipo (booleanas, enteras, time, estructuras, etc.), etc.

Editor de lenguajes: Permite el desarrollo de código del programa que ejecutará el PLC. Soporta los lenguajes literal estructurado (ST), lista de

instrucciones (IL), Ladder o contactos (LD), secuencial (SFC) y bloques de función (FDB).

Ventana de resultados: Muestra alertas sobre errores de compilación, datos mal definidos, etc.

Barra de estado: Muestra el estado de conexión con el PLC, trabajo en línea, alerta de diferencias en el programa cargado en el PLC y el cargado en el proyecto, dirección IP del PLC al que estamos conectados, etc.

7.2. CONFIGURACIÓN HARDWARE DEL PLC

Para configurar un PLC debemos entrar en el explorador de proyectos y seleccionar la opción configuración haciendo un doble clic ella.

Se abrirán dos ventanas:

- Un catálogo de dispositivos hardware que muestra los módulos disponibles organizados por funciones que pueden insertarse en el bastidor. La ventana de catálogo también puede ser visualizada en el menú de Herramientas de Hardware.
- Una ventana que muestra de forma gráfica la configuración del bastidor y en la que se debemos insertar los módulos que va a llevar instalados el rack.

Para cambiar el tamaño del rack es necesario hacer un doble clic en el rack donde está señalado en la figura que se muestra a continuación. Se nos abrirá una ventana en la que podremos elegir otro tamaño de rack o cancelar la operación de cambio.

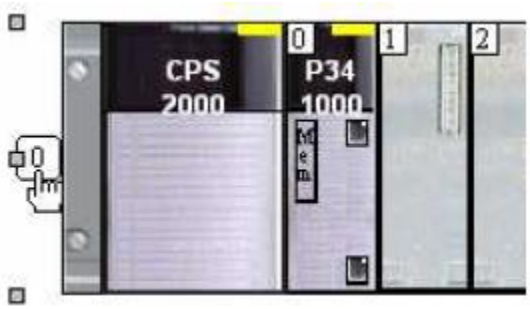


Figura 7-2 Modificación bastidor

Figura 7-3 Modificación bastidor

Para añadir los módulos que van a montar en el PLC, es suficiente con hacer doble clic en el slot deseado y se nos abrirá una ventana en la que elegiremos el módulo que se quiere configurar, se abrirá una ventana en la que seleccionaremos el módulo elegido o simplemente arrastraremos desde la librería de hardware.

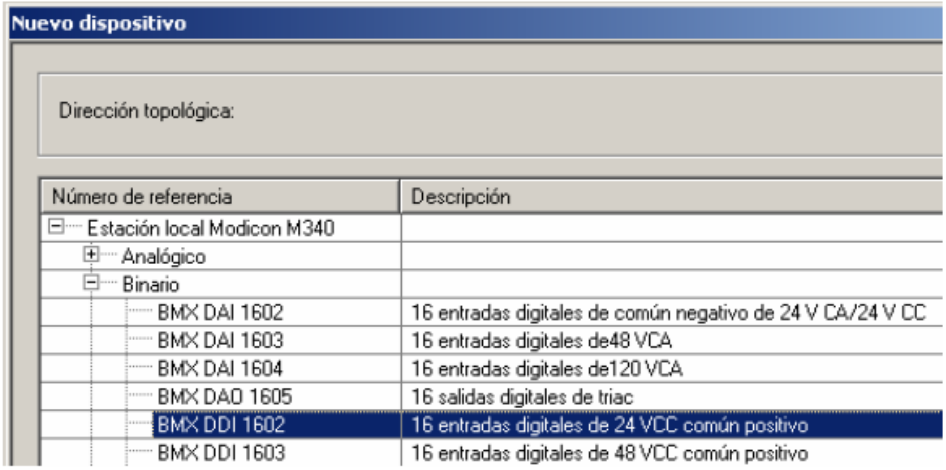


Figura 7-4 Modificación bastidor

En el caso particular de este proyecto, una vez configurada la fuente de alimentación, el rack o bastidor y el procesador, nos queda los siguiente:

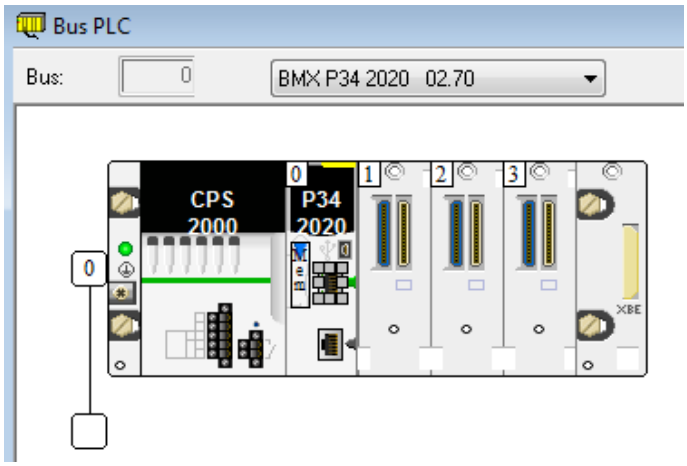


Figura 7-5 Disposición de Módulos en bastidor

Para asegurarnos que la fuente de alimentación elegida se ajusta a las necesidades de potencia requeridas por la CPU, clicaremos con el botón derecho sobre la fuente de alimentación y en el submenú que aparece seleccionamos la opción de previsión de alimentación y de E/S. El resultado es el mostrado en la figura y nos indica que de los 20 W que la fuente puede suministrar, solo estaremos consumiendo 5,3 W y además, no sobrepasamos ninguna de las corrientes máximas que se pueden suministrar en 3,3V, y 24V.

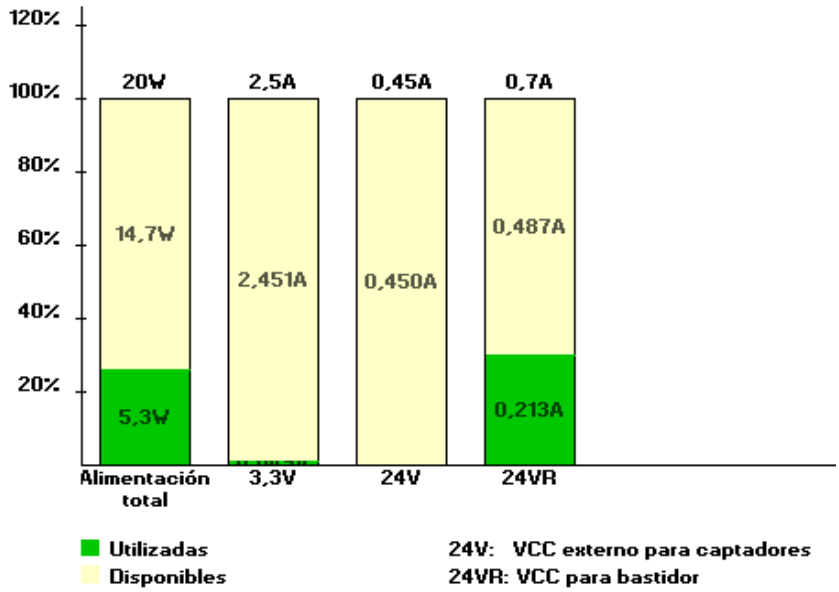


Figura 7-6 Consumos PLC

7.3. RESERVA DE MEMORIA EN LA CPU

La CPU ofrece la posibilidad de reservar la cantidad de memoria que se necesita utilizar y que se usará para operaciones internas.

Además de configurar la memoria, es posible configurar si el PLC realizará un inicio automático de la ejecución del programa tras de un corte de alimentación, es decir, si el PLC pasa automáticamente en RUN (si estaba en RUN antes del de la perdida de alimentación) o inicializar a cero las variables,%MWi, con inicio en frio (reset de las palabras de memoria cuando hay un corte de alimentación).

Para acceder a todas estas opciones, hemos de realizar un doble clic en la CPU e ir a la ventana de configuración.

Los tipos de variables que podemos configurar son:

- **%M**: (Memory) bit de memoria (bool) para almacenar estados binarios 0 o 1.
- **%MW**: (Memory Word) palabras de memoria (integer) de 16 bits para almacenar valores enteros de 16 bits.
- **%KW**: (Constant Word) palabra constante para definir una constante en el proyecto (estas no es posible alterar su valor durante la ejecución del programa).

En nuestro caso configuraremos las siguientes opciones.

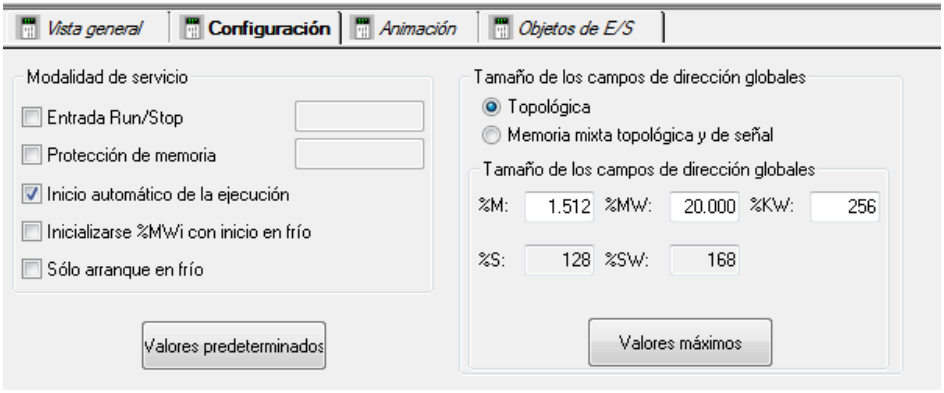


Figura 7-7 Opciones CPU

7.4. CONFIGURACIÓN DEL PUERTO ETHERNET INTEGRADO EN LA CPU

Para asignar una dirección IP a un PLC, hemos de seguir los siguientes pasos:

- Crear una conexión lógica de red
- Configurar la conexión lógica de red
- Asignar la conexión a un módulo Ethernet o puerto integrado de la CPU.

Para crear una conexión de red, una vez que estamos en el explorador de proyectos, abriremos la carpeta de comunicaciones, haciendo un clic con el botón derecho en redes, seleccionaremos Nueva red… como se indica en la siguiente figura.

Figura 7-8 Creación red

Aparecerá una ventana en la que se ha de seleccionar Ethernet entre toda la lista de redes que están disponibles Ethernet. Por defecto aparece el nombre Ethernet_1 y podremos cambiar el nombre de la red si lo deseamos. En nuestro caso, le llamaremos **Ethernet_casa**.

El siguiente paso será configurar la conexión de red. Para ello, debemos hacer un doble clic de ratón en la conexión para que se muestre su ventana de configuración. En nuestro caso la dirección IP del PLC será **100.100.1.1** y como máscara de subred **255.255.0.0**

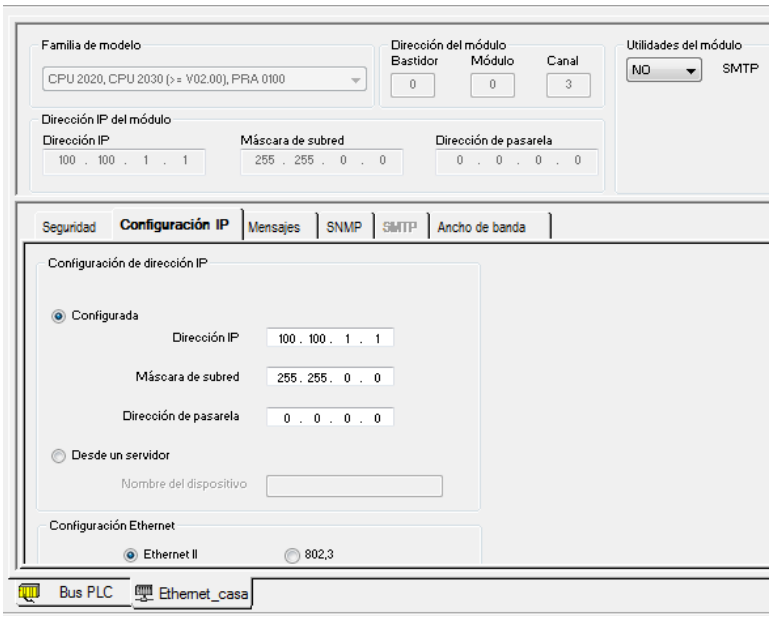


Figura 7-9 Parámetros red

Por último, solo nos queda vincular la red lógica creada, Ethernet_Casa, con el puerto físico de la CPU. Esto se realiza simplemente entrando en la configuración del bastidor y clicando con el botón derecho del ratón sobre el puerto Ethernet que lleva el procesador:

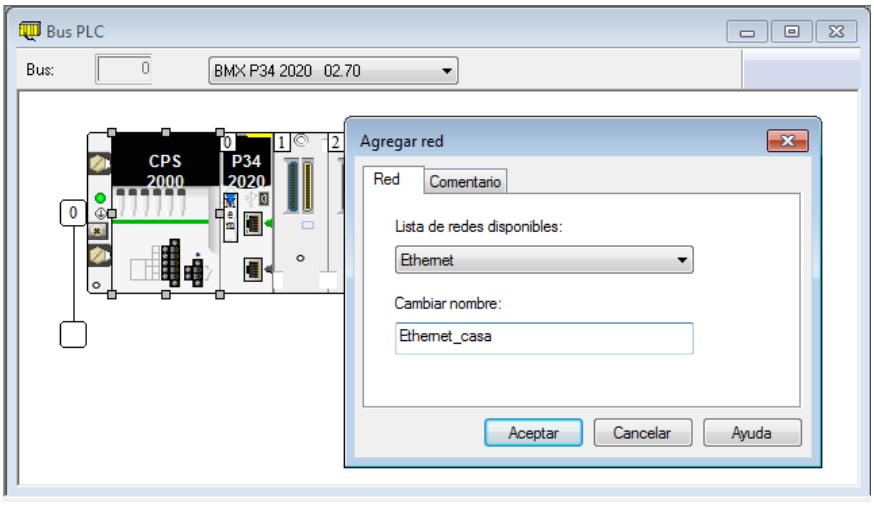


Figura 7-10 Vinculación red

7.5. DESARROLLO DEL SOFTWARE

Una vez que hemos configurado el hardware que compone el PLC, comenzaremos el desarrollo de software que será ejecutado en el equipo.

7.5.1. CONFIGURACIÓN DE LA TAREA MAESTRA

Una CPU M340 puede ejecutar aplicaciones monotarea o multitarea. En una aplicación monotarea, solo está permitido la ejecución de la tarea MAST (Master), en una aplicación multitarea define las prioridades de cada tarea que se desea.

Hay cuatro tareas disponibles y un tipo de tareas de eventos: MAST, FAST, AUX0, AUX1 y Eventos de E/S en un bastidor local.

En este trabajo, nos centraremos en la modalidad monotarea. En esta modalidad, la tarea MAST se crea por defecto y es la única que se ejecutará. La tarea MAST, tiene dos modalidades de funcionamiento:

- **Modalidad periódica**: Se define en ms cada cuanto tiempo se ejecutará el programa (scan) (de 1 a 255ms)
- **Modalidad cíclica**: Cada vez que finaliza un ciclo de scan comienza el siguiente.

El funcionamiento de un ciclo de scan se muestra en la siguiente figura. En primer lugar, se actualizan las entradas, posteriormente se ejecuta el programa y finalmente se actualizan las salidas.

Figura 7-11 Proceso de ejecución de un scan (ciclo)

Seleccionaremos como modalidad de funcionamiento el modo cíclico, para ello en el explorador de proyectos clicaremos sobre la tarea MAST con el ratón y seleccionaremos dicha modalidad. Con un tiempo de watch Dog de 250ms (tiempo máximo que puede durar la ejecución de un scan, superado este, el PLC entra en modo error y se detiene la ejecución)

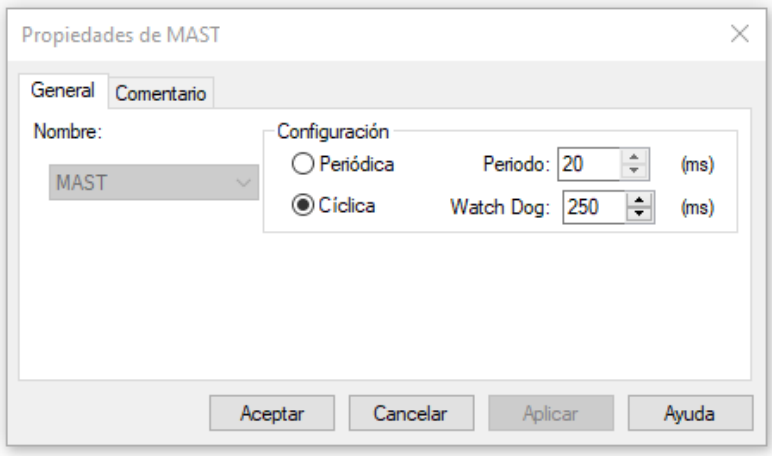


Figura 7-12 Configuración modo ejecución tarea

7.5.2. ESTRUCTURAS DE DATOS (DDT DERIVED DATA TYPE)

Las estructuras de datos o tipos de datos derivados son datos que a su vez están compuestos por diferentes tipos de datos elementales (bool, int, string, etc.). Utilizando la terminología de programación orientada a objetos sería equivalente a una clase.

En este trabajo, utilizaremos tres tipos de estructuras de datos:

- Persiana
- Puerta_automatica
- Sensor_alarma.

Cada una de estas estructuras estará compuesta por los siguientes datos elementales:

Persiana	**.fc_subida**	BOOL	1 indica fin de carrera persiana totalmente subida
	.fc_bajada	BOOL	1 indica fin de carrera persiana totalmente bajada
	.pulsa_subir	BOOL	1 indica orden de subir persiana
	.pulsa_bajar	BOOL	1 indica orden de bajar persiana
	.motor_subir	BOOL	1 indicación (a motor) de subir persiana
	.motor_bajar	BOOL	1 indicación (a motor) de bajar persiana
	.reset	BOOL	1 indica reset del control de la persiana por un mal funcionamiento.
	.movimiento	INT	Usada a efectos de animación de terminal táctil
	.monitorizacion	INT	No usada

Puerta_automatica	**.indi_abt**	BOOL	1 indica fin de carrera puerta totalmente abierta
	.ind_cda	BOOL	1 indica fin de carrera puerta totalmente cerrada
	.fotocel_interrumpida	BOOL	1 indica fotocélula detecta paso objetos
	.ind_sobrecalentamiento	BOOL	1 indica sobrecalentamiento del motor
	.boton_abrir	BOOL	1 indicación orden de abrir puerta

.boton_cerrar	BOOL	1 indicación orden de abrir puerta
.boton_cerrojo	BOOL	1 indicación al control de la puerta que la puerta no debe de abrir o cerrar
.salida_aviso_puerta _moviendo	BOOL	1 aviso a indicación luminosa externa puerta moviéndose
.salida_motor_abrir	BOOL	1 indicación (a motor) de cerrar
.salida_error	BOOL	1 ha habido un error en el control.
.rearme	BOOL	1 indicación de rearme del control tras un error

Sensor_alarma	**.movimiento**	BOOL	1 indica que se ha detectado movimiento
	.alarma	BOOL	1 indica que el sistema de alarma está conectado y se ha detectado movimiento
	.desconectado	BOOL	1 indica que se ha eliminado el sensor del sistema de alarma por presentar una animalia
	.rearme	BOOL	1 Indica que queremos resetear el sensor después de haber entrado en el estado de alarma
	.conectar	BOOL	1 indica que queremos volver a conectar el sensor que previamente se encontraba en estado de desconectado
	.sistema_conectado	BOOL	1 indica que el sistema de alarma está conectado y si se detecta un momimiento el sensor pasará al estado de alarma
	.Codificación	INT	Usada a efectos de animación de

		terminal táctil

Para dar de alta las tres estructuras de datos explicadas anteriormente, es necesario hacer clic en el explorador de proyectos en la carpeta **Tipos de datos derivados** y seleccionar la opción **abrir**. Se abrirá la ventana Editor de datos y aquí se darán de alta las tres estructuras de datos

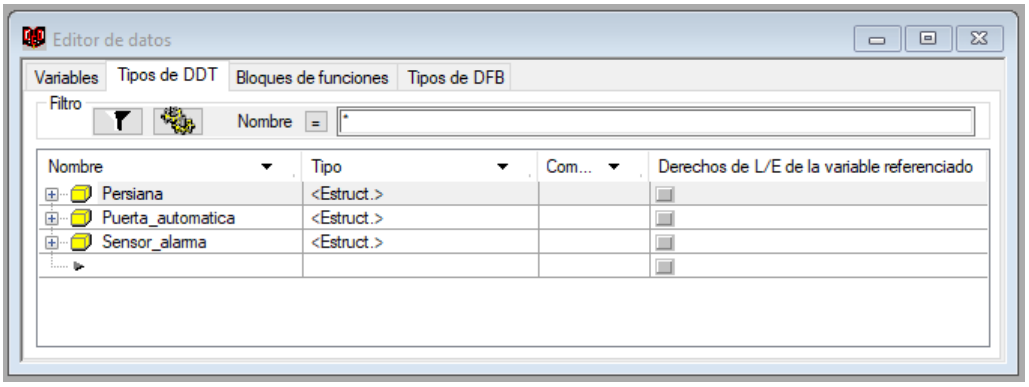


Figura 7-13 Detalle editor de datos Unity

7.5.2.1. INSTANCIAS DE ESTRUCTURAS DE DATOS

Una **instancia de datos** es un objeto que posee las características del tipo de datos (clase) del que depende. El concepto de instancia es el mismo que el usado en la programación orientada a objetos.

Se instanciarán para cada persiana de la vivienda un dato derivado de tipo persiana.

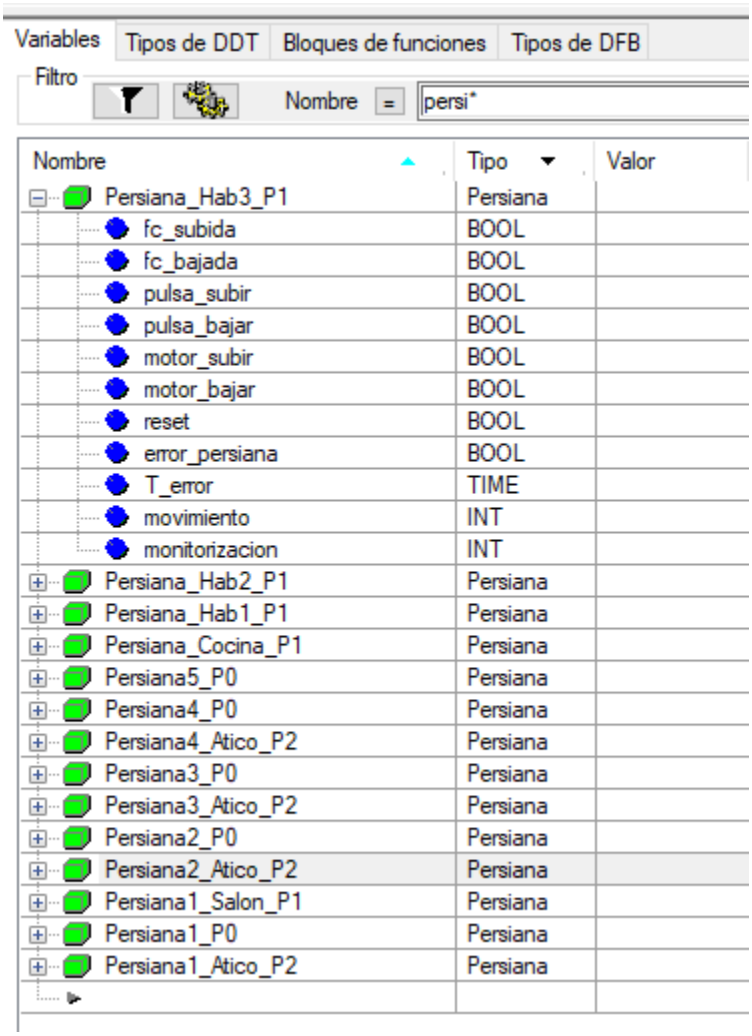


Figura 7-14 Detalle de dato derivado Persiana

Se instanciarán para cada puerta automática de la vivienda un dato derivado de tipo Puerta_garaje.

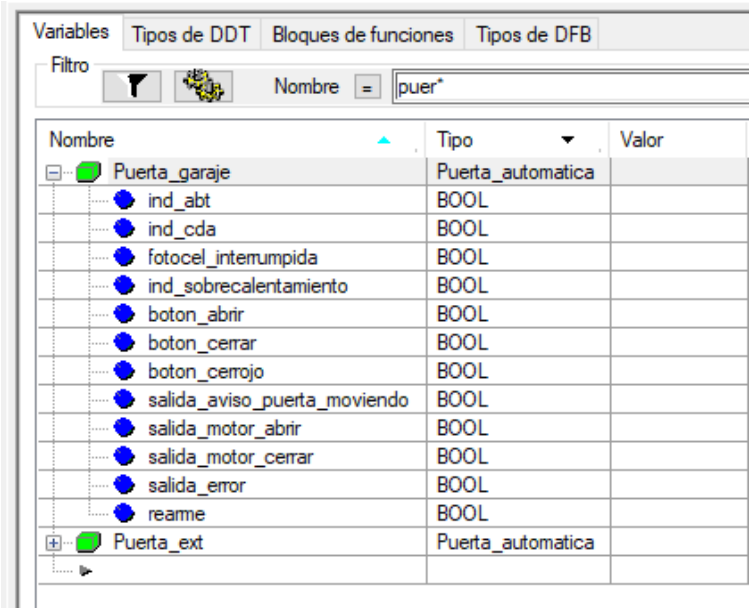


Figura 7-15 Detalle de dato derivado Puerta_garaje

Por último, se instanciarán para cada sensor de alarma de la vivienda un dato derivado de tipo Sensor_alarma.

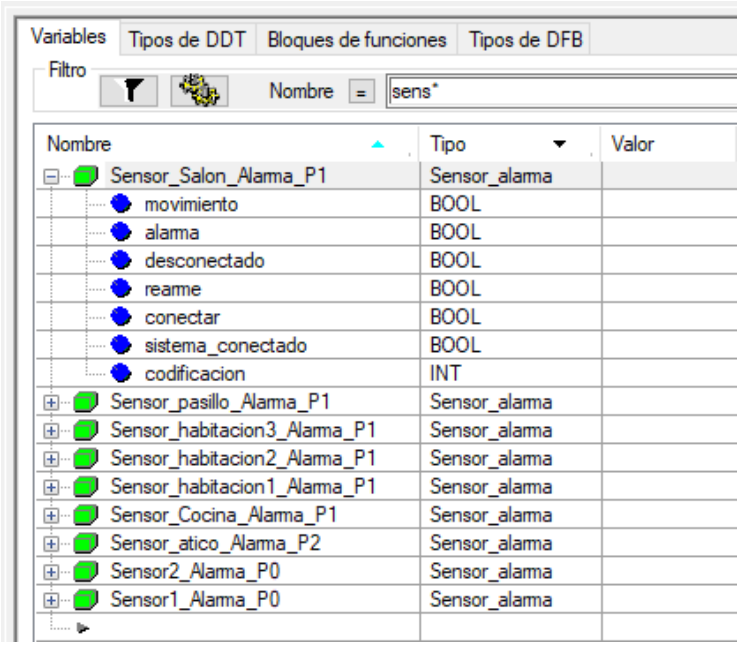


Figura 7-16 Detalle de dato derivado Sensor_alarma

Para realizar las instancias de datos que se han indicado anteriormente, basta con ir al **explorador de proyectos**, expandir la carpeta **Variables e**

instancias FB y a continuación clicar con botón derecho del ratón en la opción **Variables derivadas** seleccionando la opción **Abrir**. Una vez realizado se nos abrirá una ventana como la que se indica en la imagen mostrada a continuación, donde se podrán crear las instancias de datos necesarias.

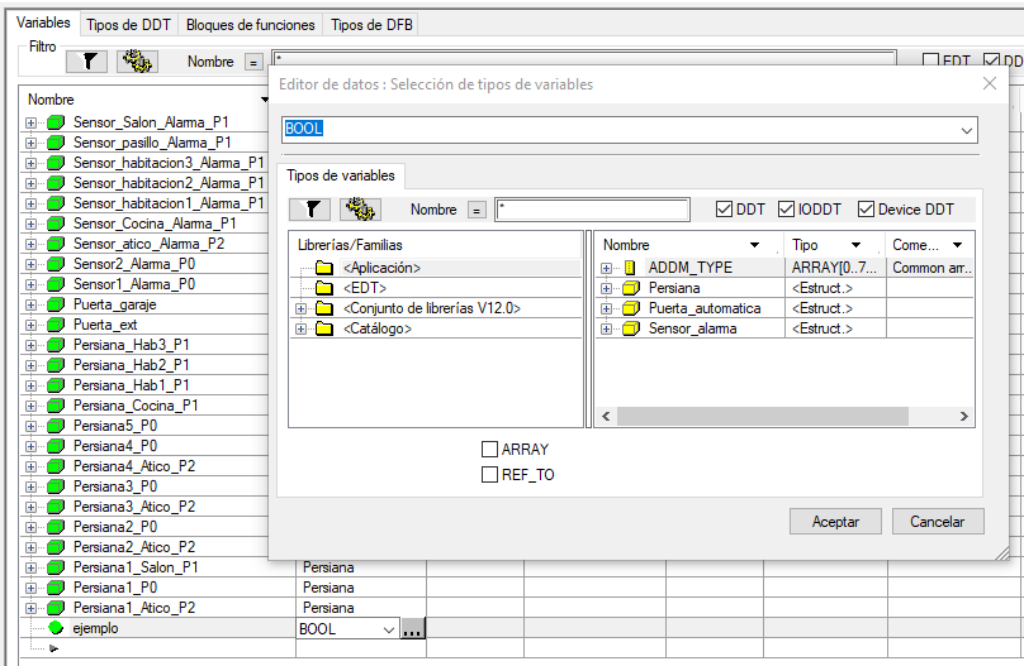


Figura 7-17 Instancias de datos derivados

7.5.3. BLOQUES DE FUNCIÓN

Un bloque de función es un objeto que contiene:

- Variables de entradas y de salidas que sirven para la ejecución del programa que contienen.
- Un **algoritmo o programa** que utiliza las variables de entradas y actúa sobre las variables de salidas.
- Variables internas (privadas y o públicas) utilizadas por el programa que ejecuta.

Unity proporciona dos tipos de bloque de función: los **EFB** (bloques de función elementales) y los **DFB** (bloques de función de usuario -Derived Function Block-)

Los tipos de bloques de funciones del usuario (Derived Function Blocks) los desarrolla el usuario con los lenguajes de programación que más le interese. Estos lenguajes pueden ser de uno o varios de los tipos siguientes:

- Lenguaje de contactos o Ladder.
- Lenguaje literal estructurado.
- Lenguaje lista de instrucciones
- Lenguaje de bloques funcionales FBD

Un bloque de función de usuario (DFB) puede tener una o varias instancias, a cada una de ellas se le identifica con mediante un nombre y posee los tipos de datos de DFB.

Los Bloques de función elementales (EFB) vienen de base en Unity y no es posible modificar su comportamiento. Existe un amplio catálogo de estos como timer, counter, motion, etc.

Un EFB puede ser instanciado tantas veces como sea necesario, cada una de las instancias hay que identificarlas con un nombre (al igual que los EFB) y posee los datos del tipo de EFB.

Centrándonos en los DFB´s, para este proyecto crearemos tres DFB´s:

- **Sistema_persiana**: Para el control de las persianas de la vivienda.
- **Sistema_puerta_autómatica**: Para el control de las puertas automáticas de la vivienda.
- **Sistema_sensor_alarma**: Para el control de los sensores de alarma de la vivienda.

Para crear los DFB´s, es necesario ir al navegador de proyectos, clicar en la carpeta **Tipos de datos derivados** y con el botón derecho del ratón, seleccionar la opción **Abrir**. Una vez realizado, se nos abrirá una ventana donde podremos configurar para cada uno de los DFB´s las variables de entrada, variables de salida, variables de entrada y salida, variables privadas, variables públicas y el lenguaje a utilizar para el control asociado al DFB.

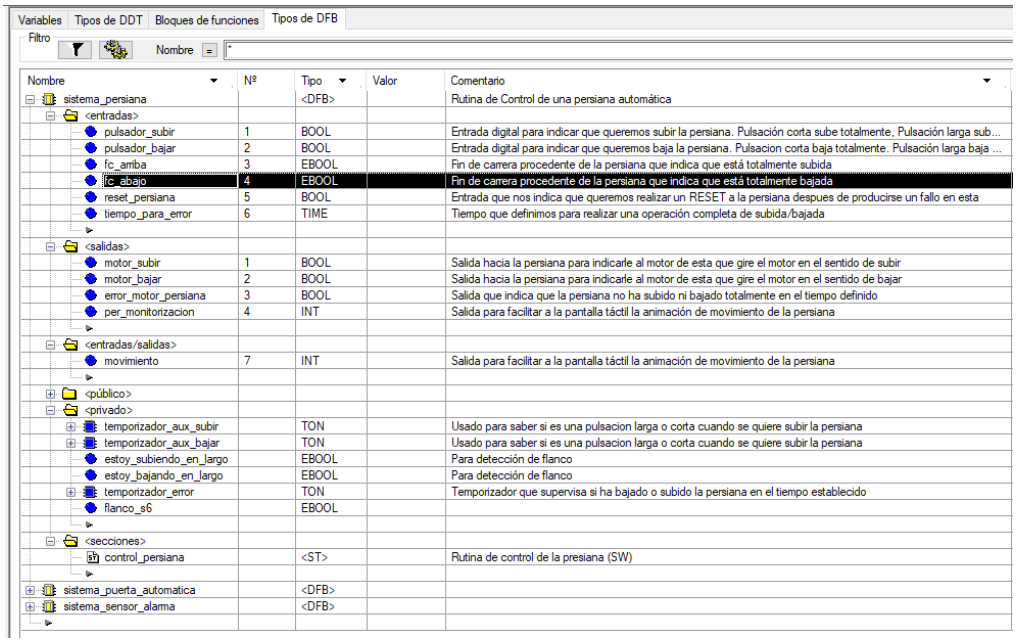


Figura 7-18 Ventana de configuración de DFB´s

7.5.3.1. DESCRIPCIÓN DFB PERSIANA

Las variables de entrada y salida son las siguientes:

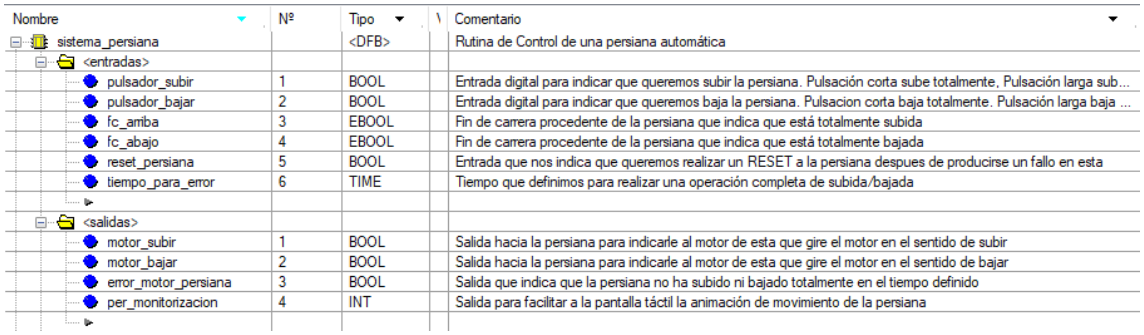


Figura 7-19 Variables de entrada y salida en DFB Persiana

El código implementado en lenguaje estructurado es el que se indica a continuación:

```
(*-------------------------------------------------------------------*)
(*----Bloque de funcíon (DFB) para el control de una persiana-----*)
(*-------------------------------------------------------------------*)

flanco_s6:= %s6 ; (* variable auxiliar para la detección de flanco del
reloj de 1 segundo *)

(*Deteccion pulsaciones cortas/largas*)
temporizador_aux_bajar (IN:= pulsador_bajar,PT:= T#2s);
temporizador_aux_subir(IN:= pulsador_subir,PT:= T#2s);
```

```
estoy_subiendo_en_largo:= temporizador_aux_subir.q;
estoy_bajando_en_largo:= temporizador_aux_bajar.q;

(*Temporizador usado para indicar que hay un error/problema en la
persiana*)
(*Monitorizamos que el tiempo de subir o bajar la persiana no excede
el valor configurado*)
(*Si este temporizador llega a su fin de cuenta, indica que la
persiana debe estar bloqueada o agarrada*)
temporizador_error (IN:= (motor_subir or motor_bajar),PT:=
tiempo_para_error);

(*Solo en el caso de que la persiana no esté en error, permitimos su
movimiento*)
if pulsador_subir and (not error_motor_persiana) then
      set(motor_subir);
      reset(motor_bajar);
end_if;
if pulsador_bajar and (not error_motor_persiana) then
      reset(motor_subir);
      set(motor_bajar);
end_if;

(*Cuando detectamos que ha llegado al fin de su recorrido la persiana
o detectamos *)
(*Que estamos en pulsos cortos y soltamos la pulsación, detenemos el
movimiento de *)
(*la persiana en cualquier sentido*)
if re(fc_arriba) or fe(estoy_subiendo_en_largo) then
      reset(motor_subir);
      reset(motor_bajar);
end_if;
if re(fc_abajo) or fe(estoy_bajando_en_largo) then
      reset(motor_subir);
      reset(motor_bajar);
end_if;

(*Si detectamos que el temporizador que supervisa el tiempo de
movimiento de la periana*)
(*ha llegado al fin de su cuenta dejamos de dar ordenes a la persiana
y activamos un bit que*)
(*indica que la persiana tiene un error. Este error, solo desaparece
si es aceptado o reconocido*)
(*desde la pantalla táctil*)
if temporizador_error.Q then
      reset(motor_subir);
      reset(motor_bajar);
      set(error_motor_persiana);
end_if;
if reset_persiana then
      reset(error_motor_persiana);
end_if;

(*-------------------------------------------------------------------*)
(*Código para permitir la animación de la persiana en la pantalla
      táctil*)
(*-------------------------------------------------------------------*)

if(motor_subir and (re(flanco_s6)or fe(flanco_s6)) and (not
error_motor_persiana))then
      movimiento:= movimiento +5 ;
```

```
end_if;
if(motor_bajar and (re(flanco_s6)or fe(flanco_s6))and (not
error_motor_persiana)) then
      movimiento:= movimiento - 5 ;
end_if;
if ((movimiento>99) or fc_arriba)then
      movimiento:= 99;
end_if;
if ((movimiento<0) or fc_abajo)then
      movimiento:= 0;
end_if;

(*Esta variable será utilizada en un SCRIPT de la pantalla táctil para
la visualización de un texto dínamico*)
if(fc_abajo) then
      per_monitorizacion:=1;
end_if;
if(fc_arriba)then
      per_monitorizacion:=2;
end_if;
if(not fc_arriba) and (not fc_abajo) then
      per_monitorizacion:= 0 ;
end_if;
```

7.5.3.2. DESCRIPCIÓN DFB SISTEMA_PUERTA_AUTOMATICA

Las variables de entrada y salida son las siguientes:

Nombre	Nº	Tipo	Comentario
sistema_persiana		<DFB>	Rutina de Control de una persiana automática
sistema_puerta_automatica		<DFB>	
<entradas>			
ind_abt	1	BOOL	Entrada desde la puerta que indica que la puerta está totalmente abierta. Fin de carrera activado
ind_cda	2	BOOL	Entrada desde la puerta que indica que la puerta está totalmente cerrada. Fin de carrera activado
fotocel_interrumpida	3	BOOL	Entrada que indica que la fotocelula ha sido interrumpida
rearme	4	BOOL	Entrada para reamar la puerta por un error
ind_sobrecalentamiento	5	BOOL	Entrada procedente del termostato de protección del Motor
boton_abrir	6	EBOOL	Entrada procedente de un pulsador o de un mando remoto que indica que se quiere abrir la puerta
boton_cerrar	7	EBOOL	Entrada procedente de un pulsador o de un mando remoto que indica que se quiere cerrar la puerta
boton_cerrojo	8	EBOOL	Entrada que indica que la puerta está trincada mediante un cerrojo y no se permite su accionamiento
<salidas>			
salida_aviso_puerta_mo...	1	BOOL	Salida que indica que la puerta está moviendose
salida_motor_abrir	2	BOOL	Orden hacia el motor para indicarle que gire en el sentido de abrir la puerta
salida_motor_cerrar	3	BOOL	Orden hacia el motor para indicarle que gire en el sentido de cerrarr la puerta
salida_error	4	BOOL	Salida que indica que la puerta está en un estado de error

Figura 7-20 Variables de entrada y salida en DFB sistema_puerta_automática

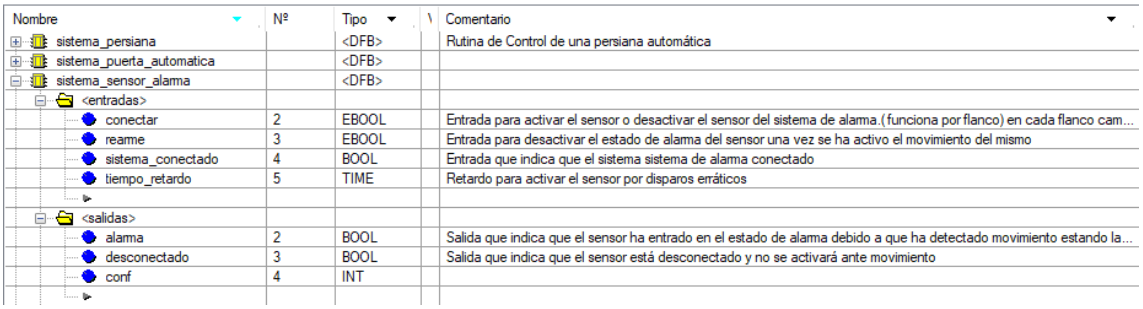
El código implementado en lenguaje estructurado es el que se indica a continuación:

```
(*-------------------------------------------------------------------*)
(*Bloque de funcíon (DFB) para el control de una puerta automática--*)
(*-------------------------------------------------------------------*)

(*Cuando el motor no está en el estado de error y se pulsa abrir la
puerta, se da la orden de abrir al motor *)
if re(boton_abrir)and not salida_error then
      set(salida_motor_abrir);
      reset(salida_motor_cerrar);
```

```
end_if;
(*Una vez abierta la puerta se quita la orden de abrir al motor*)
if ind_abt then
      reset(salida_motor_abrir);
end_if;
(*Cuando el motor no está en el estado de error y se pulsa cerrar la
puerta, se da la orden de cerrar al motor *)
if re(boton_cerrar)and not salida_error then
      reset(salida_motor_abrir);
      set(salida_motor_cerrar);
end_if;
(*Una vez abierta la puerta se quita la orden de abrir al motor*)
if ind_cda then
      reset(salida_motor_cerrar);
end_if;

(*Hacemos parpadear un bit para una señal luminosa próxima a la
puerta*)
salida_aviso_puerta_moviendo:=(salida_motor_abrir or
salida_motor_cerrar)and %s6;


(* Supervisión de que la puerta ha llegado a su fin en un tiempo
determinado*)
(* Si el motor está mas de 25 segundo girando damos la orden de
pararlo. Es una situación anómala*)
t_vigilante (IN:= salida_motor_abrir or salida_motor_cerrar,PT:=
T#25s);(*temporizador para controlar que no se enganche*)
if t_vigilante.Q or fotocel_interrumpida or boton_cerrojo or
ind_sobrecalentamiento then
      reset(salida_motor_abrir);
      reset(salida_motor_cerrar);
end_if;

(*Cuando hay sobrecalentamiento del motor o el motor está girando mas
de 25 segundos,*)
(*Lo indicamos mediante un bit*)
if t_vigilante.Q or ind_sobrecalentamiento then
      set(salida_error);
end_if;

if rearme and not ind_sobrecalentamiento then
      reset(salida_error);
end_if;
```

7.5.3.3. DESCRIPCIÓN DFB SISTEMA SENSOR ALARMA

Las variables de entrada y salida son las siguientes:



Nombre	Nº	Tipo	Comentario
sistema_persiana		<DFB>	Rutina de Control de una persiana automática
sistema_puerta_automatica		<DFB>	
sistema_sensor_alarma		<DFB>	
<entradas>			
conectar	2	EBOOL	Entrada para activar el sensor o desactivar el sensor del sistema de alarma.(funciona por flanco) en cada flanco cam...
rearme	3	EBOOL	Entrada para desactivar el estado de alarma del sensor una vez se ha activo el movimiento del mismo
sistema_conectado	4	BOOL	Entrada que indica que el sistema sistema de alarma conectado
tiempo_retardo	5	TIME	Retardo para activar el sensor por disparos erráticos
<salidas>			
alarma	2	BOOL	Salida que indica que el sensor ha entrado en el estado de alarma debido a que ha detectado movimiento estando la...
desconectado	3	BOOL	Salida que indica que el sensor está desconectado y no se activará ante movimiento
conf	4	INT	

Figura 7-21 Variables de entrada y salida en DFB sensor alarma

El código implementado en lenguaje estructurado es el que se indica a continuación:

```
(*-------------------------------------------------------------------*)
(*----Bloque de funcíon (DFB) para el control de sensores de
movimiento en la vivienda-----*)
(*-------------------------------------------------------------------*)

retardo_movimiento (IN:= movimiento, PT:=tiempo_retardo );

desconectado:= not conectar;

(*La alarma de un sensor se activ cuando:*)
(*    - Ha terminado la cuenta del temporizador y *)
(*    - El sensor no se ha desconectado del sistema por mal
funcionamiento él y*)
(*    - El Sistema de alarma está conectado*)
if retardo_movimiento.q and not desconectado and sistema_conectado
then
      set(alarma);
end_if;

(*Una vez que un sensor se ha disparaddo, solo es posible quitar su
estado de alarma*)
(*por un rearme desde la pantalla táctil*)
if rearme then
      reset(alarma);
end_if;


(*-------------------------------------------------------------------*)
(*----Código para permitir la animación del sensor de movimiento en la
pantalla táctil*)
(*-------------------------------------------------------------------*)

conf:=0;
if alarma then
      conf:=2 ;(*color rojo*)
end_if;
if desconectado then
      conf:=3 ;(*color amarillo*)
end_if;
if  (not retardo_movimiento.q) and (not desconectado) and  (not
sistema_conectado) then
```

```
      conf:=0 ;(*color verde fijo*)
end_if;
if   retardo_movimiento.q and (not desconectado) and  (not
sistema_conectado) then
      conf:=1 ;(*color verde parpadeo*)
end_if;
```

7.5.4. SECCIONES DE LA TAREA MAESTRA

7.5.4.1. DESCRIPCIÓN GENERAL DE LAS SECCIONES

Las secciones de un proyecto Unity son secciones de código autónomas de programación.

Las etiquetas de salto de las líneas de código, el código Ladder, instrucciones en literal, etc. son propias del código de sección y no es posible realizar saltos entre secciones del programa.

Se programan ya sea en:

- Lenguaje Ladder o diagrama de contactos (LD)
- Lenguaje en bloques de función (FBD)
- Lista de instrucciones (IL)
- Literal estructurado (ST)
- Diagrama funcional en secuencia (SFC)

El orden de ejecución de las secciones es el mismo que el que aparece en la ventana del navegador.

Es posible asociar una condición de ejecución a cualquier sección de la tarea Master, auxiliar o rápida pero no a aquellas secciones cuya ejecución es por eventos.

Las secciones están asociadas a tareas. Una misma sección no puede estar a la misma vez asociada a varias tareas.

Para insertar una sección en un proyecto Unity es suficiente con ir al explorador de proyectos, clicar con el botón derecho del ratón sobre la carpeta secciones y elegir la opción añadir. Aquí nos aparecerá una ventana en la que podremos indicar el nombre de la sección, lenguaje a usar, etc.

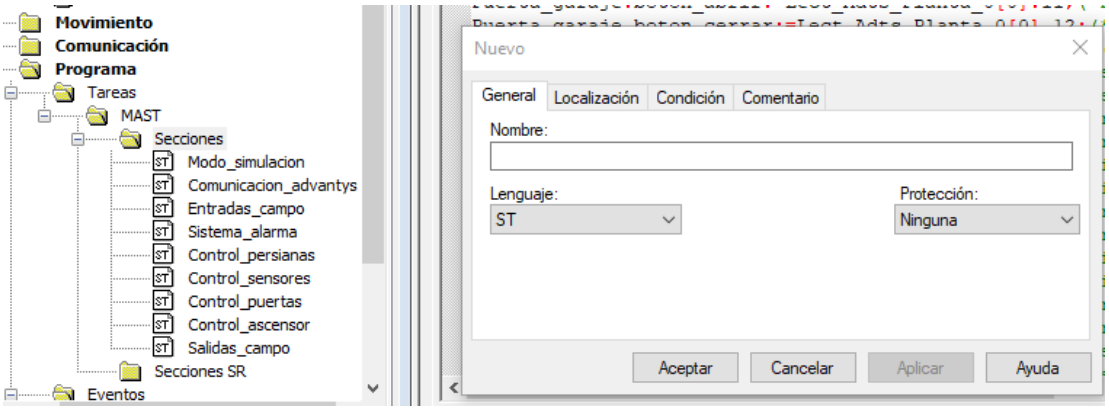


Figura 7-22 Ventana creación de secciones

7.5.4.2. SECCIÓN DE PROGRAMA: MODO_SIMULACIÓN

Esta sección, en principio irá sin programa. Será útil para meter algún código si fuese necesario en la puesta en marcha del programa.

7.5.4.3. SECCIÓN DE PROGRAMA: COMUNICACIÓN_ADVANTYS

Esta sección de programa es la encargada de realizar las lecturas y escrituras a las islas Advantys. Realizaremos por cada una de las islas Advantys una petición de lectura en la cual, nos traeremos de las islas todos los estados de las entradas digitales que tienen y realizaremos por cada una de las islas Advantys una petición de escritura con la cual activaremos o desactivaremos las salidas digitales de acuerdo a lo que el código del programa establezca.

7.5.4.3.1. EL PROTOCOLO MODBUS EN M340

Para la comunicación entre redes Ethernet los PLC´s M340 utilizan el protocolo modbus encapsulado sobre TCP, formando **Modbus TCP/IP**.

Los intercambios de datos equipos se efectúan de dos modos:

- *Modalidad servidor:* En el puerto Ethernet se aceptan todas las peticiones Modbus sobre TCP que se realicen hacia el PLC.

- *Modalidad cliente*: Este tipo de intercambio de datos permite el envío de peticiones Modbus sobre TCP a cualquier otro equipo que implemente esté protocolo.

Las funciones que nos facilitan realizar los intercambios de datos son:

- READ_VAR
- WRITE_VAR

La trama de Modbus sobre redes Ethernet mediante protocolo TCP/IP sería de la siguiente manera:

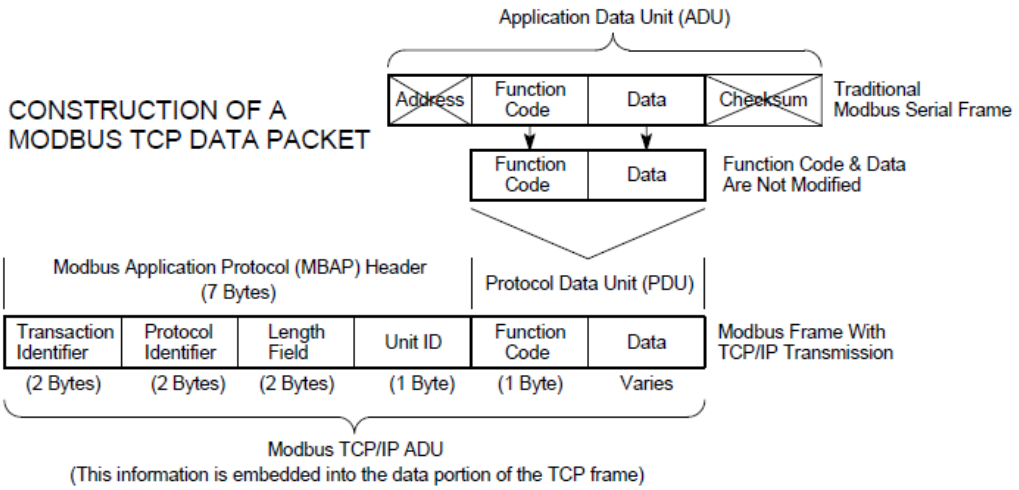


Figura 7-23 Trama modbus TCP/IP

Las peticiones Modbus son encapsuladas en los datos de la trama TCP/IP sin ser modificadas. Sin embargo, el campo Checksum de la petición Modbus no se usa ya que el estándar Ethernet TCP/IP a nivel de enlace tiene otros métodos para comprobar errores (checksum) y garantizar la integridad de los datos. Además, el byte de *address* es sustituido por un identificador.

En la figura anterior, vemos que el campo Function Code y el campo Data son enviados en su formato original. Así, una ADU (Application Data Unit) Modbus TCP/IP tiene la forma de 7 bytes de cabecera (transaction identifier + protocol identifier + length field + unit identifier) más el código de función y la parte de datos del protocolo modbus.

El significado de cada uno de los campos es el siguiente:

- *Transaction Identifier*: Son 2 bytes que identifican una transacción Modbus (petición/Respuesta).
- *Protocol Identifier*: Son dos bytes que identifican el protocolo, en nuestro caso de modbus su valor es 0.
- *Length*: Son dos bytes que indican la cantidad de bytes que vienen a continuación.
- *Unit identifier*: Es un byte que identifica a un esclavo o servidor remoto. Principalmente se usa en líneas serie o en aquellas instalaciones donde existe un Gateway Ethernet-serie.
- *Function code*: Es un byte que indica la función Modbus a utilizar. En nuestro caso, la función a utilizar será la 03 para lectura de múltiples registros enteros y la función 16 para escritura de múltiples registros enteros.
- *Data*: Este campo tiene una longitud variable dependiendo de la cantidad de datos a leer o escribir.

7.5.4.3.2. REGISTROS MODBUS DE LAS ISLAS

Existen zonas de registros Modbus que han sido reservadas en el NIM para almacenar y mantener los de datos de la isla. El tamaño de la memoria imagen es de 9.999 registros. Los registros están agrupados en bloques de memoria contiguos (o bloques), y cada uno de ellos está dedicado a unos propósitos concretos.

En la siguiente figura se muestra los bloques que contienen las islas.

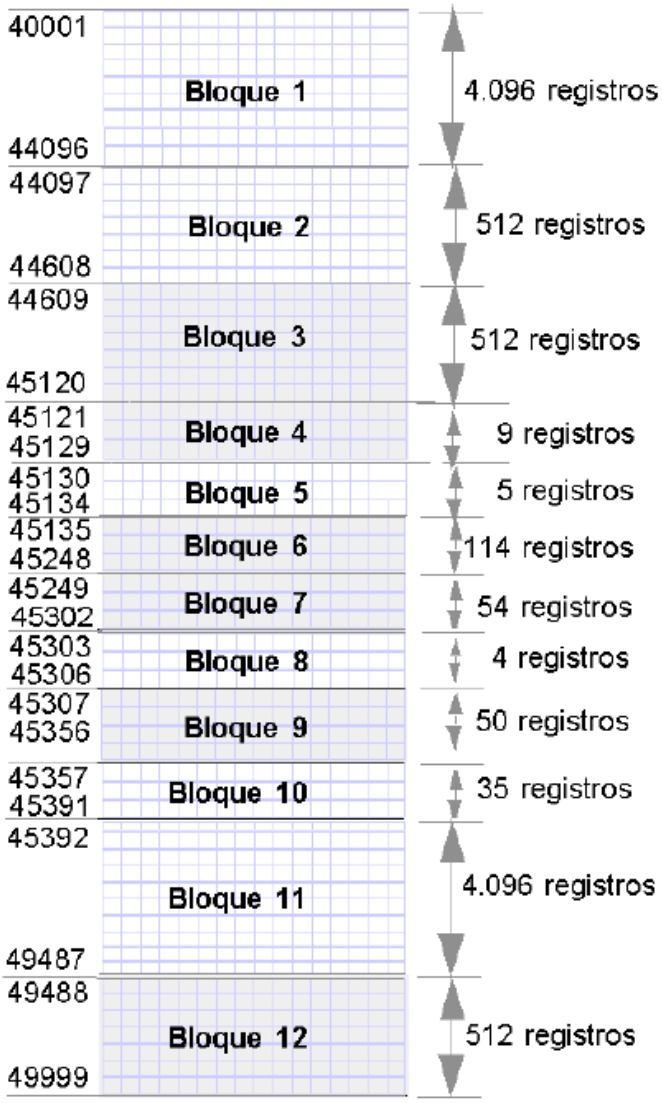


Figura 7-24 Disposición Memoria en Advantys

De todos los bloques, nos centraremos en aquellos que tienen interés en este proyecto. En concreto en los bloques 1 que contiene la imagen del proceso de datos de salida, y el bloque 11 que contiene la imagen del proceso de datos de las entradas.

En los equipos de Schneider Electric, el registro 40.001 o inicio del bloque 1 se corresponde con la dirección %MW0.

Para saber datos del primer registro del bloque de memoria 11 (registro 45.392) tenemos que hacer referencia a %MW5391 (45.392-40.001= **5.391**)

7.5.4.3.3. PETICIONES READ_VAR

La función **READ_VAR** se utiliza para leer el contenido de posiciones de memoria de los dos siguientes tipos:

- Bits internos (boolean);
- Palabras internas (integer).

Los objetos a leer pueden estar almacenados en un procesador remoto o en cualquier otro en un dispositivo conectado al puerto ethernet.

En la CPU Modicon M340, la función READ_VAR puede leer hasta 2.000 bits consecutivos o 125 palabras de 16 bits en un dispositivo remoto.

Representación en lenguaje estructurado:

READ_VAR(*Address, Object_Type, First_Object, Object_Number, Management_Param, Receiving_Array*);

El significado de cada uno de los parámetros se indica a continuación:

- *Address:* Indica la dirección de destino de la trama a enviar. En nuestro caso será la dirección IP de alguna de las tres islas.
- *Objetc_Type: Indica el tipo de contenido de lectura que queremos obtener:*
  - '%M': bits internos.
  - '%MW': palabras internas.
  - '%I': bits de entrada digitales montadas en el bastidor
  - '%Q': bits de salidas digitales montadas en el bastidor.

- *First_Object: Dirección del primer objeto a ser leído.*
- *Object_Number:* Número de objetos a leer.
- *Management_Param:* Es un array de 4 enteros (ARRAY[0..3] OF INT) utilizado para la gestión de los las peticiones (intercambios). Esta tabla de 4 enteros que se compone de:
  - Primera palabra: Es una palabra gestionada por el sistema y que está compuesta por dos bytes. De estos dos bytes, para el caso que nos ocupa, nos interesa es el bit más bajo del byte de menos peso. Cuando este bit está a 1 indica que hay un intercambio en curso y no debemos realizar

otro, cuando este bit está en valor 0 indica que no hay intercambio en curso y por tanto podremos lanzar otro intercambio si es necesario.

- Segunda palabra: Palabra gestionada por el sistema. Cuando esta palabra tiene el valor 0 indica que el intercambio anterior fue correcto. Un valor diferente a 0, indica intercambio incorrecto.
- Tercera palabra: Palabra administrada por el usuario y define el timeout, es decir el máximo tiempo que se espera para recibir la respuesta. Está definida con una base de una base de tiempo de 100ms.
- Cuarta palabra: Palabra administrada por el sistema e indica el número de posiciones de memoria que se han recibido el último intercambio realizado.

Receiving_Array: es un array de enteros (ARRAY[n..m] OF INT) y almacena las posiciones de memoria leídas.

7.5.4.3.4. PETICIONES WRITE_VAR

La función **WRITE_VAR** se utiliza para escribir el contenido de posiciones de memoria de los dos siguientes tipos:

- Bits internos (boolean);
- Palabras internas (integer).
- Los objetos a leer pueden estar almacenados en un procesador remoto o en cualquier otro en un dispositivo conectado al puerto ethernet.
- En la CPU Modicon M340, la función WRITER_VAR puede escribir hasta 2.000 bits consecutivos o 125 palabras de 16 bits en un dispositivo remoto.

Representación en lenguaje estructurado:

WRITE_VAR(*Address, Object_Type, First_Object, Object_Number, Data_to_Write, Management_Param*);

El significado de cada uno de los parámetros se indica a continuación:

- *Address:* Indica la dirección de destino de la trama a enviar. En nuestro caso será la dirección IP de alguna de las tres islas.
- *Objetc_Type: Indica el tipo de contenido de lectura que queremos obtener:*
  - '%M': bits internos.
  - '%MW': palabras internas.
  - '%I': bits de entrada digitales montadas en el bastidor
  - '%Q': bits de salidas digitales montadas en el bastidor.

- *First_Object: Dirección del primer objeto a ser leído.*
- *Object_Number:* Número de objetos a leer*.*
- *Management_Param:* Es un array de 4 enteros (ARRAY[0..3] OF INT) utilizado para la gestión de los las peticiones (intercambios). Esta tabla de 4 enteros que se compone de:
  - Primera palabra: Es una palabra gestionada por el sistema y que está compuesta por dos bytes. De estos dos bytes, para el caso que nos ocupa, nos interesa es el bit más bajo del byte de menos peso. Cuando este bit está a 1 indica que hay un intercambio en curso y no debemos realizar otro, cuando este bit está en valor 0 indica que no hay intercambio en curso y por tanto podremos lanzar otro intercambio si es necesario.
  - Segunda palabra: Palabra gestionada por el sistema. Cuando esta palabra tiene el valor 0 indica que el intercambio anterior fue correcto. Un valor diferente a 0, indica intercambio incorrecto.
  - Tercera palabra: Palabra administrada por el usuario y define el timeout, es decir el máximo tiempo que se espera para recibir la respuesta. Está definida con una base de una base de tiempo de 100ms.

- Cuarta palabra: Palabra administrada por el sistema e indica el número de posiciones de memoria que se han escrito el último intercambio realizado.

- *Receiving_Array: es un array de enteros (ARRAY[n..m] OF INT) y almacena las posiciones de memoria leídas.*

7.5.4.3.5. VALORES RECIBIDOS EN EL ARRAY DE RECEPCIÓN DE ENTRADA

En cada una de las tres peticiones READ_VAR que se van a hacer contra las islas se han definido tres arrays de tamaño 30 (AARRAY[0..29] OF INT. En ellos se almacenarán las lecturas de los módulos de entradas digitales.

El contenido de cada uno de los elementos del array es el que se muestra en la tabla siguiente.

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
Indicacion de puerta abierta	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].0
Indicacion de puerta cerrada	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].1
Haz de fotocelula interrumpido	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].2
Indicacion de sobretemperatura en motor puerta	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].3
Pulsador de abrir	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].4
Pulsador de cerrar	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].5
Contacto procedente de llave para evitar actuar sobre la puerta	Puerta deslizante exterior	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].6
Indicacion de puerta abierta	Puerta garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].7
Indicacion de puerta cerrada	Puerta garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].8
Haz de fotocelula interrumpido	Puerta garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].9
Indicacion de sobretemperatura en motor puerta	Puerta garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].10
Pulsador de abrir	Puerta garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].11
Pulsador de cerrar	Puerta garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].12

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
Contacto procedente de llave para evitar actuar sobre la puerta	Puerta_garaje	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].13
Indicacion persiana totalmente subida	Persiana 1 Planta 0	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].14
Indicacion persiana totalmente bajada	Persiana 1 Planta 0	Adts_Planta_0	ED_1	Lect_Adts_Planta_0[0].15
Pulsador de subir persiana	Persiana 1 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].0
Pulsador de bajar persiana	Persiana 1 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].1
Indicacion persiana totalmente subida	Persiana 2 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].2
Indicacion persiana totalmente bajada	Persiana 2 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].3
Pulsador de subir persiana	Persiana 2 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].4
Pulsador de bajar persiana	Persiana 2 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].5
Indicacion persiana totalmente subida	Persiana 3 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].6
Indicacion persiana totalmente bajada	Persiana 3 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].7
Pulsador de subir persiana	Persiana 3 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].8
Pulsador de bajar persiana	Persiana 3 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].9
Indicacion persiana totalmente subida	Persiana 4 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].10
Indicacion persiana totalmente bajada	Persiana 4 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].11
Pulsador de subir persiana	Persiana 4 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].12
Pulsador de bajar persiana	Persiana 4 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].13
Indicacion persiana totalmente subida	Persiana 5 Planta 0	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].14
Indicacion	Persiana 5 Planta	Adts_Planta_0	ED_2	Lect_Adts_Planta_0[1].15

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
persiana totalmente bajada	0			
Pulsador de subir persiana	Persiana 5 Planta 0	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].0
Pulsador de bajar persiana	Persiana 5 Planta 0	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].1
Entrada de movimiento al sistema	Sensor1 alarma Planta 0	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].2
Entrada de movimiento al sistema	Sensor2 alarma Planta 0	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].3
Fin de carrera planta 0	Ascensor	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].4
Indicacion de puerta abierta de la planta 0	Ascensor	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].5
Boton para planta 0	Ascensor	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].6
Boton para planta 1	Ascensor	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].7
Boton para planta 2	Ascensor	Adts_Planta_0	ED_3	Lect_Adts_Planta_0[2].8
Indicacion persiana totalmente subida	Persiana 1 Cocina Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].0
Indicacion persiana totalmente bajada	Persiana 1 Cocina Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].1
Pulsador de subir persiana	Persiana 1 Cocina Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].2
Pulsador de bajar persiana	Persiana 1 Cocina Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].3
Indicacion persiana totalmente subida	Persiana 1 Salon Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].4
Indicacion persiana totalmente bajada	Persiana 1 Salon Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].5
Pulsador de subir persiana	Persiana 1 Salon Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].6
Pulsador de bajar persiana	Persiana 1 Salon Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].7
Indicacion persiana totalmente subida	Persiana habitacion 1 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].8
Indicacion persiana totalmente bajada	Persiana habitacion 1 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].9
Pulsador de subir persiana	Persiana habitacion 1 Planta	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].10

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
	1			
Pulsador de bajar persiana	Persiana habitacion 1 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].11
Indicacion persiana totalmente subida	Persiana habitacion 2 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].12
Indicacion persiana totalmente bajada	Persiana habitacion 2 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].13
Pulsador de subir persiana	Persiana habitacion 2 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].14
Pulsador de bajar persiana	Persiana habitacion 2 Planta 1	Adts_Planta_1	ED_1	Lect_Adts_Planta_1[0].15
Indicacion persiana totalmente subida	Persiana habitacion 3 Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].0
Indicacion persiana totalmente bajada	Persiana habitacion 3 Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].1
Pulsador de subir persiana	Persiana habitacion 3 Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].2
Pulsador de bajar persiana	Persiana habitacion 3 Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].3
Entrada de movimiento al sistema	Sensor salon alarma Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].4
Entrada de movimiento al sistema	Sensor pasillo alarma Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].5
Entrada de movimiento al sistema	Sensor cocina alarma Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].6
Entrada de movimiento al sistema	Sensor habitacion1 alarma Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].7
Entrada de movimiento al sistema	Sensor habitacion2 alarma Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].8
Entrada de movimiento al sistema	Sensor habitacion3 alarma Planta 1	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].9
Fin de carrera planta 1	Ascensor	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1.10
Indicacion de puerta abierta de la planta 1	Ascensor	Adts_Planta_1	ED_2	Lect_Adts_Planta_1[1].11
Indicacion persiana totalmente	Persiana1 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].0

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
subida				
Indicacion persiana totalmente bajada	Persiana1 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].1
Pulsador de subir persiana	Persiana1 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].2
Pulsador de bajar persiana	Persiana1 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].3
Indicacion persiana totalmente subida	Persiana2 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].4
Indicacion persiana totalmente bajada	Persiana2 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].5
Pulsador de subir persiana	Persiana2 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].6
Pulsador de bajar persiana	Persiana2 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].7
Indicacion persiana totalmente subida	Persiana 4 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].8
Indicacion persiana totalmente bajada	Persiana 4 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].9
Pulsador de subir persiana	Persiana 4 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].10
Pulsador de bajar persiana	Persiana 4 atico Planta 2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].11
Indicacion persiana totalmente subida	Persiana 3 Atico Planta2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].12
Indicacion persiana totalmente bajada	Persiana 3 Atico Planta2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].13
Pulsador de subir persiana	Persiana 3 Atico Planta2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].14
Pulsador de bajar persiana	Persiana 3 Atico Planta2	Adts_Planta_2	ED_1	Lect_Adts_Planta_2[0].15
Entrada de movimiento al sistema	Sensor atico alarma Planta 2	Adts_Planta_2	ED_2	Lect_Adts_Planta_2[1].0
Fin de carrera planta 2	Ascensor	Adts_Planta_2	ED_2	Lect_Adts_Planta_2[1].1
Indicacion de puerta abierta de la planta 2	Ascensor	Adts_Planta_2	ED_2	Lect_Adts_Planta_2[1].2

Tabla 7-1 Señales almacenadas en los arrays de recepción de datos

7.5.4.3.6. VALORES A ESCRIBIR POR EL ARRAY DE SALIDA

En cada una de las tres peticiones WRITE_VAR que se van a hacer contra las islas se han definido tres arrays de tamaño 15 (*ARRAY[0..14] OF INT*). En ellos se almacenarán los valores o estados a enviar a los módulos de salidas digitales.

El contenido de cada uno de los elementos del array es el que se muestra en la siguiente tabla.

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
Salida hacia motor puerta para girar en el sentido de apertura	Puerta deslizante exterior	Adts_Planta_0	SD_1	Esc_Adts_Planta_0[0].0
Salida hacia motor puerta para girar en el sentido de cierre	Puerta deslizante exterior	Adts_Planta_0	SD_1	Esc_Adts_Planta_0[0].1
Salida hacia luz rotativa para indicar movimiento de la puerta	Puerta deslizante exterior	Adts_Planta_0	SD_1	Esc_Adts_Planta_0[0].2
Salida hacia motor puerta para girar en el sentido de apertura	Puerta_garaje	Adts_Planta_0	SD_1	Esc_Adts_Planta_0[0].3
Salida hacia motor puerta para girar en el sentido de cierre	Puerta_garaje	Adts_Planta_0	SD_1	Esc_Adts_Planta_0[0].4
Salida hacia luz rotativa para indicar movimiento de la puerta	Puerta_garaje	Adts_Planta_0	SD_1	Esc_Adts_Planta_0[0].5
Salida hacia motor perisana subir	Persiana 1 Planta 0	Adts_Planta_0	SD_2	Esc_Adts_Planta_0[1].0
Salida hacia motor perisana bajar	Persiana 1 Planta 0	Adts_Planta_0	SD_2	Esc_Adts_Planta_0[1].1
Salida hacia motor perisana subir	Persiana 2 Planta 0	Adts_Planta_0	SD_2	Esc_Adts_Planta_0[1].2

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
Salida hacia motor perisana bajar	Persiana 2 Planta 0	Adts_Planta_0	SD_2	Esc_Adts_Planta_0[1].3
Salida hacia motor perisana subir	Persiana 3 Planta 0	Adts_Planta_0	SD_2	Esc_Adts_Planta_0[1].4
Salida hacia motor perisana bajar	Persiana 3 Planta 0	Adts_Planta_0	SD_2	Esc_Adts_Planta_0[1].5
Salida hacia motor perisana subir	Persiana 4 Planta 0	Adts_Planta_0	SD_3	Esc_Adts_Planta_0[2].0
Salida hacia motor perisana bajar	Persiana 4 Planta 0	Adts_Planta_0	SD_3	Esc_Adts_Planta_0[2].1
Salida hacia motor perisana subir	Persiana 5 Planta 0	Adts_Planta_0	SD_3	Esc_Adts_Planta_0[2].2
Salida hacia motor perisana bajar	Persiana 5 Planta 0	Adts_Planta_0	SD_3	Esc_Adts_Planta_0[2].3
Salida para accionar la subida del ascensor	Ascensor	Adts_Planta_0	SD_3	Esc_Adts_Planta_0[2].4
Salida para accionar la bajada del ascensor	Ascensor	Adts_Planta_0	SD_3	Esc_Adts_Planta_0[2].5
Salida para accionar el cerrojo para la puerta de la planta 0	Ascensor	Adts_Planta_0	SD_4	Esc_Adts_Planta_0[3].0
Salida hacia motor perisana subir	Persiana 1 Cocina Planta 1	Adts_Planta_1	SD_1	Esc_Adts_Planta_1[0].0
Salida hacia motor perisana bajar	Persiana 1 Cocina Planta 1	Adts_Planta_1	SD_1	Esc_Adts_Planta_1[0].1
Salida hacia motor perisana subir	Persiana 1 Salon Planta 1	Adts_Planta_1	SD_1	Esc_Adts_Planta_1[0].2
Salida hacia motor perisana bajar	Persiana 1 Salon Planta 1	Adts_Planta_1	SD_1	Esc_Adts_Planta_1[0].3
Salida hacia motor perisana subir	Persiana habitacion 1 Planta 1	Adts_Planta_1	SD_1	Esc_Adts_Planta_1[0].4
Salida hacia motor perisana bajar	Persiana habitacion 1 Planta 1	Adts_Planta_1	SD_1	Esc_Adts_Planta_1[0].5
Salida hacia	Persiana	Adts_Planta_1	SD_2	Esc_Adts_Planta_1[1].0

DESCRIPCION	SISTEMA	ASIGNACION	MODULO	CANAL
motor perisana subir	habitacion 2 Planta 1			
Salida hacia motor perisana bajar	Persiana habitacion 2 Planta 1	Adts_Planta_1	SD_2	Esc_Adts_Planta_1[1].1
Salida hacia motor perisana subir	Persiana habitacion 3 Planta 1	Adts_Planta_1	SD_2	Esc_Adts_Planta_1[1].2
Salida hacia motor perisana bajar	Persiana habitacion 3 Planta 1	Adts_Planta_1	SD_2	Esc_Adts_Planta_1[1].3
Salida para accionar el cerrojo para la puerta de la planta 1	Ascensor	Adts_Planta_1	SD_2	Esc_Adts_Planta_1[1].4
Salida sirena	sistema alarma	Adts_Planta_1	SD_2	Esc_Adts_Planta_1[1].5
Salida hacia motor perisana subir	Persiana1 atico Planta 2	Adts_Planta_2	SD_1	Esc_Adts_Planta_2[0].0
Salida hacia motor perisana bajar	Persiana1 atico Planta 2	Adts_Planta_2	SD_1	Esc_Adts_Planta_2[0].1
Salida hacia motor perisana subir	Persiana2 atico Planta 2	Adts_Planta_2	SD_1	Esc_Adts_Planta_2[0].2
Salida hacia motor perisana bajar	Persiana2 atico Planta 2	Adts_Planta_2	SD_1	Esc_Adts_Planta_2[0].3
Salida hacia motor perisana subir	Persiana 4 atico Planta 2	Adts_Planta_2	SD_1	Esc_Adts_Planta_2[0].4
Salida hacia motor perisana bajar	Persiana 4 atico Planta 2	Adts_Planta_2	SD_1	Esc_Adts_Planta_2[0].5
Salida hacia motor perisana subir	Persiana 3 Atico Planta2	Adts_Planta_2	SD_2	Esc_Adts_Planta_2[1].0
Salida hacia motor perisana bajar	Persiana 3 Atico Planta2	Adts_Planta_2	SD_2	Esc_Adts_Planta_2[1].1
Salida para accionar el cerrojo para la puerta de la planta 2	Ascensor	Adts_Planta_2	SD_2	Esc_Adts_Planta_2[1].2

Tabla 7-2 Señales almacenadas en los arrays de escritura de datos

7.5.4.3.7. CÓDIGO DE LA SECCIÓN

```
(*--------------------------------------------------------------------*)
(*Lecturas y escrituras a periferia distribuida (advantys) PLANTA 0-*)
(*--------------------------------------------------------------------*)
if(%s0 or %s1 or %s13)then
      reset(lectura_comm_adt0[0].0);
      reset(lectura_comm_adt1[0].0);
      reset(lectura_comm_adt2[0].0);
      reset(escritura_comm_adt0[0].0);
      reset(escritura_comm_adt1[0].0);
      reset(escritura_comm_adt1[0].0);
end_if;

(* Comprobación comunicacion correcta con isla Planta 0 *)
aux_comm_adt0:= lectura_comm_adt0[0].0;
if fe(aux_comm_adt0) then
      if(lectura_comm_adt0[1]=0)then
            INC(mensaje_lectura_adv0_ok);
      else
            INC(mensaje_lectura_adv0_no_ok);
      end_if;
end_if;

T_min_commprobacion_adt0 (IN:= comprobacion_ok_adt0, PT:= t#5s);

(* Petición de lectura de datos a isla Planta 0 *)
IF NOT lectura_comm_adt0[0].0 THEN
      lectura_comm_adt0[2]:=10; (* Time Out 10x100ms *)
      READ_VAR(ADDM('Ethernet_casa{100.100.1.2}'), '%MW', 5391 , 20,
lectura_comm_adt0, Lect_Adts_Planta_0);
END_IF;

(* Petición de escrituras de datos a isla Planta 0 *)
aux_comm_escritura_adt0:= escritura_comm_adt0[0].0;
if fe(aux_comm_escritura_adt0) then
      if(escritura_comm_adt0[1]=0)then
            INC(mensaje_escritura_adv0_ok);
      else
            INC(mensaje_escritura_adv0_no_ok);
      end_if;
end_if;

IF NOT escritura_comm_adt0[0].0 THEN
      escritura_comm_adt0[2]:=10; (* Time Out 10x100ms *)
      WRITE_VAR(ADDM('Ethernet_casa{100.100.1.2}'), '%MW', 0 ,10,
Esc_Adts_Planta_0 , escritura_comm_adt0);
END_IF;

(*--------------------------------------------------------------------*)
(*Lecturas y escrituras a periferia distribuida (advantys) PLANTA 1-*)
(*--------------------------------------------------------------------*)

(* Comprobación comunicacion correcta con isla Planta 1 *)
aux_comm_adt1:= lectura_comm_adt1[0].0;
if fe(aux_comm_adt1) then
      if(lectura_comm_adt1[1]=0)then
            INC(mensaje_lectura_adv1_ok);
      else
            INC(mensaje_lectura_adv1_no_ok);
      end_if;
end_if;
T_min_commprobacion_adt1 (IN:= comprobacion_ok_adt1, PT:= t#5s);
```

```
(* Petición de lectura de datos a isla Planta 1 *)
aux_comm_escritura_adt1:= escritura_comm_adt1[0].0;
if fe(aux_comm_escritura_adt1) then
      if(escritura_comm_adt1[1]=0)then
            INC(mensaje_escritura_adv1_ok);
      else
            INC(mensaje_escritura_adv1_no_ok);
      end_if;
end_if;

IF NOT lectura_comm_adt1[0].0 THEN
      INC(BORRA_CUENTA_LECTURAS_P1);
      lectura_comm_adt1[2]:=10; (* Time Out 10x100ms *)
      READ_VAR(ADDM('Ethernet_casa{100.100.1.3}'), '%MW', 5391 , 20,
lectura_comm_adt1, Lect_Adts_Planta_1);
END_IF;

(* Petición de escrituras de datos a isla Planta 1 *)
IF NOT escritura_comm_adt1[0].0 THEN
      INC(BORRA_CUENTA_ESCRITURAS_P1);
      escritura_comm_adt1[2]:=10; (* Time Out 10x100ms *)
      WRITE_VAR(ADDM('Ethernet_casa{100.100.1.3}'), '%MW', 0 ,10,
Esc_Adts_Planta_1 , escritura_comm_adt1);
END_IF;

(*-------------------------------------------------------------------------*)
(*Lecturas y escrituras a periferia distribuida (advantys) PLANTA 2-*)
(*-------------------------------------------------------------------------*)

(* Comprobación comunicacion correcta con isla Planta 2 *)
aux_comm_adt2:= lectura_comm_adt2[0].0;
if fe(aux_comm_adt2) then
      if(lectura_comm_adt2[1]=0)then
            INC(mensaje_lectura_adv2_ok);
      else
            INC(mensaje_lectura_adv2_no_ok);
      end_if;
end_if;
T_min_commprobacion_adt2 (IN:= comprobacion_ok_adt2, PT:= t#5s);

(* Petición de lectura de datos a isla Planta 2 *)
IF NOT lectura_comm_adt2[0].0 THEN
      lectura_comm_adt2[2]:=10; (* Time Out 10x100ms *)
      READ_VAR(ADDM('Ethernet_casa{100.100.1.4}'), '%MW', 5391 , 20,
lectura_comm_adt2, Lect_Adts_Planta_2);
END_IF;

(* Petición de escrituras de datos a isla Planta 2 *)
aux_comm_escritura_adt2:= escritura_comm_adt2[0].0;
if fe(aux_comm_escritura_adt2) then
      if(escritura_comm_adt2[1]=0)then
            INC(mensaje_escritura_adv2_ok);
      else
            INC(mensaje_escritura_adv2_no_ok);
      end_if;
end_if;
IF NOT escritura_comm_adt2[0].0 THEN
      escritura_comm_adt2[2]:=10; (* Time Out 10x100ms *)
      WRITE_VAR(ADDM('Ethernet_casa{100.100.1.4}'), '%MW', 0 ,10,
Esc_Adts_Planta_2 , escritura_comm_adt2);
```

```
END_IF;
```

7.5.4.4. SECCIÓN DE PROGRAMA: ENTRADAS_CAMPO

Esta sección tiene por objeto asignar a cada una de las variables de las estructuras de datos los valores recogidos por las islas Advantys y que proceden de los sensores.

La ventaja que comporta el uso de esta sección es que en el caso de tener que cambiar la dirección de memoria de una variable de campo, no es necesario buscar su valor en todas las secciones del programa. Basta con modificarla en esta sección.

El código implementado en esta sección es el mostrado a continuación:

```
(*-------------------------------------------------------------------*)
(*Asignación de los datos leidos de periferia distribuida (advantys) a
estructuras de datos*)
(*-------------------------------------------------------------------*)

(* Lecturas de campo en Planta 0 *)
Puerta_ext.ind_abt:=Lect_Adts_Planta_0[0].0;(*Puerta  deslizante
exterior*)
Puerta_ext.ind_cda:=Lect_Adts_Planta_0[0].1;(*Puerta  deslizante
exterior*)
Puerta_ext.fotocel_interrumpida:=Lect_Adts_Planta_0[0].2;(*Puerta
deslizante exterior*)
Puerta_ext.ind_sobrecalentamiento:=Lect_Adts_Planta_0[0].3;(*Puerta
deslizante exterior*)
Puerta_ext.boton_abrir:=Lect_Adts_Planta_0[0].4;(*Puerta  deslizante
exterior*)
Puerta_ext.boton_cerrar:=Lect_Adts_Planta_0[0].5;(*Puerta  deslizante
exterior*)
Puerta_ext.boton_cerrojo:=Lect_Adts_Planta_0[0].6;(*Puerta  deslizante
exterior*)
Puerta_garaje.ind_abt:=Lect_Adts_Planta_0[0].7;(*Puerta  garaje*)
Puerta_garaje.ind_cda:=Lect_Adts_Planta_0[0].8;(*Puerta  garaje*)
Puerta_garaje.fotocel_interrumpida:=Lect_Adts_Planta_0[0].9;(*Puerta
garaje*)
Puerta_garaje.ind_sobrecalentamiento:=Lect_Adts_Planta_0[0].10;(*Puert
a  garaje*)
Puerta_garaje.boton_abrir:=Lect_Adts_Planta_0[0].11;(*Puerta  garaje*)
Puerta_garaje.boton_cerrar:=Lect_Adts_Planta_0[0].12;(*Puerta   garaje*)
Puerta_garaje.boton_cerrojo:=Lect_Adts_Planta_0[0].13;(*Puerta
garaje*)
Persiana1_P0.fc_subida:=Lect_Adts_Planta_0[0].14;(*Persiana 1 Planta
0*)
Persiana1_P0.fc_bajada:=Lect_Adts_Planta_0[0].15;(*Persiana 1 Planta
0*)
Persiana1_P0.pulsa_subir:=Lect_Adts_Planta_0[1].0;(*Persiana 1 Planta
0*)
```

```
Persiana1_P0.pulsa_bajar:=Lect_Adts_Planta_0[1].1;(*Persiana 1 Planta
0*)
Persiana2_P0.fc_subida:=Lect_Adts_Planta_0[1].2;(*Persiana 2 Planta 0*)
Persiana2_P0.fc_bajada:=Lect_Adts_Planta_0[1].3;(*Persiana 2 Planta 0*)
Persiana2_P0.pulsa_subir:=Lect_Adts_Planta_0[1].4;(*Persiana 2 Planta
0*)
Persiana2_P0.pulsa_bajar:=Lect_Adts_Planta_0[1].5;(*Persiana 2 Planta
0*)
Persiana3_P0.fc_subida:=Lect_Adts_Planta_0[1].6;(*Persiana 3 Planta 0*)
Persiana3_P0.fc_bajada:=Lect_Adts_Planta_0[1].7;(*Persiana 3 Planta 0*)
Persiana3_P0.pulsa_subir:=Lect_Adts_Planta_0[1].8;(*Persiana 3 Planta
0*)
Persiana3_P0.pulsa_bajar:=Lect_Adts_Planta_0[1].9;(*Persiana 3 Planta
0*)
Persiana4_P0.fc_subida:=Lect_Adts_Planta_0[1].10;(*Persiana 4 Planta
0*)
Persiana4_P0.fc_bajada:=Lect_Adts_Planta_0[1].11;(*Persiana 4 Planta
0*)
Persiana4_P0.pulsa_subir:=Lect_Adts_Planta_0[1].12;(*Persiana 4 Planta
0*)
Persiana4_P0.pulsa_bajar:=Lect_Adts_Planta_0[1].13;(*Persiana 4 Planta
0*)
Persiana5_P0.fc_subida:=Lect_Adts_Planta_0[1].14;(*Persiana 5 Planta
0*)
Persiana5_P0.fc_bajada:=Lect_Adts_Planta_0[1].15;(*Persiana 5 Planta
0*)
Persiana5_P0.pulsa_subir:=Lect_Adts_Planta_0[2].0;(*Persiana 5 Planta
0*)
Persiana5_P0.pulsa_bajar:=Lect_Adts_Planta_0[2].1;(*Persiana 5 Planta
0*)
Sensor1_Alarma_P0.movimiento:=Lect_Adts_Planta_0[2].2;(*Sensor1 alarma
Planta 0*)
Sensor2_Alarma_P0.movimiento:=Lect_Adts_Planta_0[2].3;(*Sensor2 alarma
Planta 0*)
fc0:=Lect_Adts_Planta_0[2].4;(*Ascensor*)
puerta_P0_abt:=Lect_Adts_Planta_0[2].5;(*Ascensor*)
boton_0:=Lect_Adts_Planta_0[2].6;(*Ascensor*)
boton_1:=Lect_Adts_Planta_0[2].7;(*Ascensor*)
boton_2:=Lect_Adts_Planta_0[2].8;(*Ascensor*)

(* Lecturas de campo en Planta 1 *)
Persiana_Cocina_P1.fc_subida:=Lect_Adts_Planta_1[0].0;(*Persiana 1
Cocina Planta 1*)
Persiana_Cocina_P1.fc_bajada:=Lect_Adts_Planta_1[0].1;(*Persiana 1
Cocina Planta 1*)
Persiana_Cocina_P1.pulsa_subir:=Lect_Adts_Planta_1[0].2;(*Persiana 1
Cocina Planta 1*)
Persiana_Cocina_P1.pulsa_bajar:=Lect_Adts_Planta_1[0].3;(*Persiana 1
Cocina Planta 1*)
Persiana1_Salon_P1.fc_subida:=Lect_Adts_Planta_1[0].4;(*Persiana 1
Salon Planta 1*)
Persiana1_Salon_P1.fc_bajada:=Lect_Adts_Planta_1[0].5;(*Persiana 1
Salon Planta 1*)
Persiana1_Salon_P1.pulsa_subir:=Lect_Adts_Planta_1[0].6;(*Persiana 1
Salon Planta 1*)
Persiana1_Salon_P1.pulsa_bajar:=Lect_Adts_Planta_1[0].7;(*Persiana 1
Salon Planta 1*)
Persiana_Hab1_P1.fc_subida:=Lect_Adts_Planta_1[0].8;(*Persiana
habitacion 1 Planta 1*)
Persiana_Hab1_P1.fc_bajada:=Lect_Adts_Planta_1[0].9;(*Persiana
habitacion 1 Planta 1*)
```

```
Persiana_Hab1_P1.pulsa_subir:=Lect_Adts_Planta_1[0].10;(*Persiana
habitacion 1 Planta 1*)
Persiana_Hab1_P1.pulsa_bajar:=Lect_Adts_Planta_1[0].11;(*Persiana
habitacion 1 Planta 1*)
Persiana_Hab2_P1.fc_subida:=Lect_Adts_Planta_1[0].12;(*Persiana
habitacion 2 Planta 1*)
Persiana_Hab2_P1.fc_bajada:=Lect_Adts_Planta_1[0].13;(*Persiana
habitacion 2 Planta 1*)
Persiana_Hab2_P1.pulsa_subir:=Lect_Adts_Planta_1[0].14;(*Persiana
habitacion 2 Planta 1*)
Persiana_Hab2_P1.pulsa_bajar:=Lect_Adts_Planta_1[0].15;(*Persiana
habitacion 2 Planta 1*)
Persiana_Hab3_P1.fc_subida:=Lect_Adts_Planta_1[1].0;(*Persiana
habitacion 3 Planta 1*)
Persiana_Hab3_P1.fc_bajada:=Lect_Adts_Planta_1[1].1;(*Persiana
habitacion 3 Planta 1*)
Persiana_Hab3_P1.pulsa_subir:=Lect_Adts_Planta_1[1].2;(*Persiana
habitacion 3 Planta 1*)
Persiana_Hab3_P1.pulsa_bajar:=Lect_Adts_Planta_1[1].3;(*Persiana
habitacion 3 Planta 1*)
Sensor_Salon_Alarma_P1.movimiento:=Lect_Adts_Planta_1[1].4;(*Sensor
salon alarma Planta 1*)
Sensor_pasillo_Alarma_P1.movimiento:=Lect_Adts_Planta_1[1].5;(*Sensor
pasillo alarma Planta 1*)
Sensor_Cocina_Alarma_P1.movimiento:=Lect_Adts_Planta_1[1].6;(*Sensor
cocina alarma Planta 1*)
Sensor_habitacion1_Alarma_P1.movimiento:=Lect_Adts_Planta_1[1].7;(*Sen
sor habitacion1 alarma Planta 1*)
Sensor_habitacion2_Alarma_P1.movimiento:=Lect_Adts_Planta_1[1].8;(*Sen
sor habitacion2 alarma Planta 1*)
Sensor_habitacion3_Alarma_P1.movimiento:=Lect_Adts_Planta_1[1].9;(*Sen
sor habitacion3 alarma Planta 1*)
fc1:=Lect_Adts_Planta_1[1].10;(*Ascensor*)
puerta_P1_abt:=Lect_Adts_Planta_1[1].11;(*Ascensor*)

(* Lecturas de campo en Planta 2 *)
Persiana1_Atico_P2.fc_subida:=Lect_Adts_Planta_2[0].0;(*Persiana1
atico Planta 2*)
Persiana1_Atico_P2.fc_bajada:=Lect_Adts_Planta_2[0].1;(*Persiana1
atico Planta 2*)
Persiana1_Atico_P2.pulsa_subir:=Lect_Adts_Planta_2[0].2;(*Persiana1
atico Planta 2*)
Persiana1_Atico_P2.pulsa_bajar:=Lect_Adts_Planta_2[0].3;(*Persiana1
atico Planta 2*)
Persiana2_Atico_P2.fc_subida:=Lect_Adts_Planta_2[0].4;(*Persiana2
atico Planta 2*)
Persiana2_Atico_P2.fc_bajada:=Lect_Adts_Planta_2[0].5;(*Persiana2
atico Planta 2*)
Persiana2_Atico_P2.pulsa_subir:=Lect_Adts_Planta_2[0].6;(*Persiana2
atico Planta 2*)
Persiana2_Atico_P2.pulsa_bajar:=Lect_Adts_Planta_2[0].7;(*Persiana2
atico Planta 2*)
Persiana4_Atico_P2.fc_subida:=Lect_Adts_Planta_2[0].8;(*Persiana 4
atico Planta 2*)
Persiana4_Atico_P2.fc_bajada:=Lect_Adts_Planta_2[0].9;(*Persiana 4
atico Planta 2*)
Persiana4_Atico_P2.pulsa_subir:=Lect_Adts_Planta_2[0].10;(*Persiana 4
atico Planta 2*)
Persiana4_Atico_P2.pulsa_bajar:=Lect_Adts_Planta_2[0].11;(*Persiana 4
atico Planta 2*)
```

```
Persiana3_Atico_P2.fc_subida:=Lect_Adts_Planta_2[0].12;(*Persiana 3
Atico Planta2*)
Persiana3_Atico_P2.fc_bajada:=Lect_Adts_Planta_2[0].13;(*Persiana 3
Atico Planta2*)
Persiana3_Atico_P2.pulsa_subir:=Lect_Adts_Planta_2[0].14;(*Persiana 3
Atico Planta2*)
Persiana3_Atico_P2.pulsa_bajar:=Lect_Adts_Planta_2[0].15;(*Persiana 3
Atico Planta2*)
Sensor_atico_Alarma_P2.movimiento:=Lect_Adts_Planta_2[1].0;(*Sensor
atico alarma Planta 2*)
fc2:=Lect_Adts_Planta_2[1].1;(*Ascensor*)
puerta_P2_abt:=Lect_Adts_Planta_2[1].2;(*Ascensor*)
```

7.5.4.5. SECCIÓN DE PROGRAMA: SISTEMA_ALARMA

Esta sección tiene por objeto asignar a cada una de las variables de las estructuras de datos los valores que el terminal táctil necesita escribir en el PLC. Valores como dar la orden de conectar un cierto sensor o desconectarlo del sistema de alarma por si estuviese en defecto, o si hay que rearmar un sensor por haber saltado su alarma, etc.

El código implementado en esta sección es el mostrado a continuación:

```
(*SENSORES CONECTAR*)
Sensor1_Alarma_P0.conectar:= %MW15500.0;
Sensor2_Alarma_P0.conectar:= %MW15510.0;

Sensor_Cocina_Alarma_P1.conectar:= %MW15520.0;
Sensor_Salon_Alarma_P1.conectar:= %MW15530.0;
Sensor_Pasillo_Alarma_P1.conectar:= %MW15540.0;
Sensor_Habitacion1_Alarma_P1.conectar:= %MW15550.0;
Sensor_Habitacion2_Alarma_P1.conectar:= %MW15560.0;
Sensor_Habitacion3_Alarma_P1.conectar:= %MW15570.0;
Sensor_atico_Alarma_P2.conectar:= %MW15580.0 ;

(*Retardo de la señal de activar alarma*)
(*alarma*)
Retardo_activacion_alarma (IN:= %mw16001.0, PT:= t#5s);
(*señal para magelis que indica que la alarma esta en proceso de
activacion*)

%mw16001.1:= %mw16001.0 and (not Retardo_activacion_alarma.q);

if %mw16001.0 and (not Retardo_activacion_alarma.q)then
      %MW16003:=2;(*ACTIVANDO ALARMA*)
end_if;
if Retardo_activacion_alarma.q then
      %MW16003:=3;(* ALARMA activada*)
end_if;
if  (not %mw16001.0 )then
      %MW16003:=1;(* ALARMA NO activada*)
end_if;
```

```
sistema_alarma_activado:= Retardo_activacion_alarma.q ;

Sensor1_Alarma_P0.rearme:= Sensor2_Alarma_P0.rearme:=
 Sensor_Cocina_Alarma_P1.rearme:= Sensor_Salon_Alarma_P1.rearme:=
 Sensor_Pasillo_Alarma_P1.rearme:= Sensor_Atico_Alarma_P2.rearme:=
 Sensor_Habitacion1_Alarma_P1.rearme:=
Sensor_Habitacion2_Alarma_P1.rearme:=
 Sensor_Habitacion3_Alarma_P1.rearme:= %MW16000.0;
(*AGRUPAMOS LA ALARMA DE LOS SENSORES PARA QUE LA MAGELIS PRESENTE LA
ZONA DONDE SE HAN DISPARADO*)
 %MW16002.0:= Sensor1_Alarma_P0.alarma OR
Sensor2_Alarma_P0.alarma ;(*reunion de los alarmas planta baja*)

%MW16002.1:= Sensor_Cocina_Alarma_P1.alarma OR
Sensor_Salon_Alarma_P1.alarma OR
Sensor_Pasillo_Alarma_P1.alarma OR Sensor_Habitacion1_Alarma_P1.alarma
OR
Sensor_Habitacion2_Alarma_P1.alarma OR
Sensor_Habitacion3_Alarma_P1.alarma ; (*reunion de los alarmas planta
1*)

%MW16002.2:= Sensor_Atico_Alarma_P2.alarma ; (*reunion de los
movimientos planta 2*)

(*AGRUPAMOS TODAS LAS ALARMAS EN UN SOLO BIT PARA INDICARLO EN LA
PANTALLA PRINCIPAL DE ALARMAS*)
%MW16002.3:= %MW16002.0 OR %MW16002.1 OR %MW16002.2;
```

7.5.4.6. SECCIÓN DE PROGRAMA: CONTROL_PERSIANAS

Esta sección tiene por objeto invocar las instancias de los DFB´s de control de cada una de las persianas, así como preparar las variables que son necesarias para la facilitar la animación de los sensores en la pantalla táctil.

El código implementado en esta sección es el mostrado a continuación:

```
(*-----------------------------------------------------------------*)
(*----Instancias de control (DFB´s) Persianas de la Planta 0-----*)
(*-----------------------------------------------------------------*)

cntl_persiana1_p0 (pulsador_subir:= Persiana1_P0.pulsa_subir
                                    OR %MW16021.0,
                   pulsador_bajar:= Persiana1_P0.pulsa_bajar
                                    OR %MW16021.1,
                   fc_arriba:= Persiana1_P0.fc_subida,
                   fc_abajo:= Persiana1_P0.fc_bajada,
                   reset_persiana:= Persiana1_P0.reset OR %MW16021.2,
                   tiempo_para_error:= t#15s,
                   movimiento:= Persiana1_P0.movimiento,
                   motor_subir => Persiana1_P0.motor_subir,
                   motor_bajar => Persiana1_P0.motor_bajar,
                   error_motor_persiana => Persiana1_P0.error_persiana,
                   per_monitorizacion => Persiana1_P0.monitorizacion);
```

```
cntl_persiana2_p0 (pulsador_subir:= Persiana2_P0.pulsa_subir
                                    or %MW16023.0,
                  pulsador_bajar:= Persiana2_P0.pulsa_bajar
                                  or %MW16023.1 ,
                  fc_arriba:= Persiana2_P0.fc_subida,
                  fc_abajo:= Persiana2_P0.fc_bajada,
                  reset_persiana:= Persiana2_P0.reset or %MW16023.2 ,
                  tiempo_para_error:= T#15s,
                  movimiento:= Persiana2_P0.movimiento,
                  motor_subir => Persiana2_P0.motor_subir,
                  motor_bajar => Persiana2_P0.motor_bajar,
                  error_motor_persiana =>Persiana2_P0.error_persiana,
                  per_monitorizacion => Persiana2_P0.monitorizacion);

 cntl_persiana3_p0(pulsador_subir:= Persiana3_P0.pulsa_subir
                                    or %MW16025.0,
                   pulsador_bajar:= Persiana3_P0.pulsa_bajar
                                    or %MW16025.1,
                   fc_arriba:= Persiana3_P0.fc_subida,
                   fc_abajo:= Persiana3_P0.fc_bajada,
                   reset_persiana:= Persiana3_P0.reset or %MW16025.2,
                   tiempo_para_error:= T#15s,
                   movimiento:= Persiana3_P0.movimiento,
                   motor_subir => Persiana3_P0.motor_subir,
                   error_motor_persiana =>Persiana3_P0.error_persiana,
                   motor_bajar => Persiana3_P0.motor_bajar,
                   per_monitorizacion => Persiana3_P0.monitorizacion);

cntl_persiana4_p0 (pulsador_subir:= Persiana4_P0.pulsa_subir
                                    or %MW16027.0,
                  pulsador_bajar:= Persiana4_P0.pulsa_bajar
                                  or %MW16027.1,
                  fc_arriba:= Persiana4_P0.fc_subida,
                  fc_abajo:= Persiana4_P0.fc_bajada,
                  reset_persiana:= Persiana4_P0.reset or %MW16027.2 ,
                  tiempo_para_error:= T#15s,
                  movimiento:= Persiana4_P0.movimiento,
                  motor_subir => Persiana4_P0.motor_subir,
                  error_motor_persiana =>Persiana4_P0.error_persiana,
                  motor_bajar => Persiana4_P0.motor_bajar,
                  per_monitorizacion => Persiana4_P0.monitorizacion);

cntl_persiana5_p0 (pulsador_subir:= Persiana5_P0.pulsa_subir
                                    or %MW16029.0,
                   pulsador_bajar:= Persiana5_P0.pulsa_bajar
                                    or %MW16029.1 ,
                   fc_arriba:= Persiana5_P0.fc_subida,
                   fc_abajo:= Persiana5_P0.fc_bajada,
                   reset_persiana:= Persiana5_P0.reset or %MW16029.2,
                   tiempo_para_error:= T#15s,
                   movimiento:= Persiana5_P0.movimiento,
                   motor_subir => Persiana5_P0.motor_subir,
                   error_motor_persiana =>Persiana5_P0.error_persiana,
                   motor_bajar => Persiana5_P0.motor_bajar,
                   per_monitorizacion => Persiana5_P0.monitorizacion);


cntl_persiana6_p0 (pulsador_subir:= Persiana3_Atico_P2.pulsa_subir
                                    or %mw16045.0,
                  pulsador_bajar:= Persiana3_Atico_P2.pulsa_bajar
                                    or %mw16045.1,
```

```
                fc_arriba:= Persiana3_Atico_P2.fc_subida,
                fc_abajo:= Persiana3_Atico_P2.fc_bajada,
                reset_persiana:= Persiana3_Atico_P2.reset
                                 or %mw16045.2,
                tiempo_para_error:= T#15s,
                movimiento:= Persiana3_Atico_P2.movimiento,
                motor_subir => Persiana3_Atico_P2.motor_subir,
                error_motor_persiana
                 =>Persiana3_Atico_P2.error_persiana,
                motor_bajar => Persiana3_Atico_P2.motor_bajar,
                per_monitorizacion
                 => Persiana3_Atico_P2.monitorizacion);

(*-----------------------------------------------------------------*)
(*----Instancias de control (DFB´s) Persianas de la Planta 1-----*)
(*-----------------------------------------------------------------*)

cntl_persiana1_Salon_p1 (pulsador_subir:=
                         Persiana1_Salon_P1.pulsa_subir
                               or %MW16031.0,
                         pulsador_bajar:=
                         Persiana1_Salon_P1.pulsa_bajar
                               or %MW16031.1,
                         fc_arriba:= Persiana1_Salon_P1.fc_subida,
                         fc_abajo:= Persiana1_Salon_P1.fc_bajada,
                         reset_persiana:= Persiana1_Salon_P1.reset
                                          or %MW16031.2,
                         tiempo_para_error:= T#15s,
                         movimiento:= Persiana1_Salon_p1.movimiento,
                         motor_subir => Persiana1_Salon_P1.motor_subir,
                         error_motor_persiana
                         =>Persiana1_Salon_P1.error_persiana,
                         motor_bajar => Persiana1_Salon_P1.motor_bajar,
                         per_monitorizacion
                         => Persiana1_Salon_P1.monitorizacion);

cntl_persiana2_Salon_p1 (pulsador_subir:=
                               Persiana4_Atico_P2.pulsa_subir
                               or %mw16047.0,
                         pulsador_bajar:=
                         Persiana4_Atico_P2.pulsa_bajar
                               or %mw16047.1,
                         fc_arriba:= Persiana4_Atico_P2.fc_subida,
                         fc_abajo:= Persiana4_Atico_P2.fc_bajada,
                         reset_persiana:= Persiana4_Atico_P2.reset
                                         or %mw16047.2,
                         tiempo_para_error:= T#15s,
                         movimiento:= Persiana4_Atico_P2.movimiento,
                         error_motor_persiana
                         =>Persiana4_Atico_P2.error_persiana,
                         motor_subir => Persiana4_Atico_P2.motor_subir,
                         motor_bajar => Persiana4_Atico_P2.motor_bajar,
                         per_monitorizacion
                         => Persiana4_Atico_P2.monitorizacion);

cntl_persiana_Cocina_p1 (pulsador_subir:=
                               Persiana_Cocina_P1.pulsa_subir
                               or %MW16033.0,
                         pulsador_bajar:=
                         Persiana_Cocina_P1.pulsa_bajar
                               or %MW16033.1,
```

```
                        fc_arriba:= Persiana_Cocina_P1.fc_subida,
                        fc_abajo:= Persiana_Cocina_P1.fc_bajada,
                        reset_persiana:= Persiana_Cocina_P1.reset
                                         or %MW16033.2,
                        tiempo_para_error:= T#15s,
                        movimiento:= Persiana_Cocina_p1.movimiento,
                        error_motor_persiana
                        =>Persiana_Cocina_P1.error_persiana,
                        motor_subir => Persiana_Cocina_P1.motor_subir,
                        motor_bajar => Persiana_Cocina_P1.motor_bajar,
                        per_monitorizacion
                        => Persiana_Cocina_P1.monitorizacion);

cntl_persiana_Hab1_p1 (pulsador_subir:= Persiana_Hab1_P1.pulsa_subir
                                         or %MW16035.0,
                        pulsador_bajar:= Persiana_Hab1_P1.pulsa_bajar
                                         or %MW16035.1,
                        fc_arriba:= Persiana_Hab1_P1.fc_subida,
                        fc_abajo:= Persiana_Hab1_P1.fc_bajada,
                        reset_persiana:= Persiana_Hab1_P1.reset
                                         or %MW16035.2,
                        tiempo_para_error:= T#15s,
                        movimiento:= Persiana_Hab1_p1.movimiento,
                        error_motor_persiana
                        =>Persiana_Hab1_P1.error_persiana,
                        motor_subir => Persiana_Hab1_P1.motor_subir,
                        motor_bajar => Persiana_Hab1_P1.motor_bajar,
                        per_monitorizacion
                        => Persiana_Hab1_P1.monitorizacion);

cntl_persiana_Hab2_p1 (pulsador_subir:= Persiana_Hab2_P1.pulsa_subir
                                         or %MW16037.0,
                        pulsador_bajar:= Persiana_Hab2_P1.pulsa_bajar
                                         or %MW16037.1,
                        fc_arriba:= Persiana_Hab2_P1.fc_subida,
                        fc_abajo:= Persiana_Hab2_P1.fc_bajada,
                        reset_persiana:= Persiana_Hab2_P1.reset
                                         or %MW16037.2,
                        tiempo_para_error:= T#15s,
                        movimiento:= Persiana_Hab2_p1.movimiento,
                        error_motor_persiana
                        =>Persiana_Hab2_P1.error_persiana,
                        motor_subir => Persiana_Hab2_P1.motor_subir,
                        motor_bajar => Persiana_Hab2_P1.motor_bajar,
                        per_monitorizacion
                        => Persiana_Hab2_P1.monitorizacion);

cntl_persiana_Hab3_p1 (pulsador_subir:= Persiana_Hab3_P1.pulsa_subir
                                         or %MW16039.0,
                       pulsador_bajar:= Persiana_Hab3_P1.pulsa_bajar
                                         or %MW16039.1,
                       fc_arriba:= Persiana_Hab3_P1.fc_subida,
                       fc_abajo:= Persiana_Hab3_P1.fc_bajada,
                       reset_persiana:= Persiana_Hab3_P1.reset
                                         or %MW16039.2,
                       tiempo_para_error:= T#15s,
                       movimiento:= Persiana_Hab3_p1.movimiento,
                       error_motor_persiana
                       =>Persiana_Hab3_P1.error_persiana,
                       motor_subir => Persiana_Hab3_P1.motor_subir,
                       motor_bajar => Persiana_Hab3_P1.motor_bajar,
```

```
                    per_monitorizacion
                    => Persiana_Hab3_P1.monitorizacion);


(*-----------------------------------------------------------------*)
(*----Instancias de control (DFB´s) Persianas de la Planta 2-----*)
(*-----------------------------------------------------------------*)

cntl_persiana1_Atico_p2 (pulsador_subir:=
                         Persiana1_Atico_p2.pulsa_subir or %mw16041.0,
                         pulsador_bajar:=
                         Persiana1_Atico_p2.pulsa_bajar or %mw16041.1,
                          fc_arriba:= Persiana1_Atico_p2.fc_subida,
                          fc_abajo:= Persiana1_Atico_p2.fc_bajada,
                          reset_persiana:= Persiana1_Atico_p2.reset
                                           or %mw16041.2,
                          tiempo_para_error:= T#15s,
                          movimiento:= Persiana1_Atico_p2.movimiento,
                          error_motor_persiana
                          =>Persiana1_Atico_P2.error_persiana,
                          motor_subir => Persiana1_Atico_p2.motor_subir,
                          motor_bajar => Persiana1_Atico_p2.motor_bajar,
                          per_monitorizacion
                          => Persiana1_Atico_p2.monitorizacion);

cntl_persiana2_Atico_p2 (pulsador_subir:=
                              Persiana2_Atico_p2.pulsa_subir
                                    or %mw16043.0,
                          pulsador_bajar:=
                              Persiana2_Atico_p2.pulsa_bajar
                                    or %mw16043.1,
                          fc_arriba:= Persiana2_Atico_p2.fc_subida,
                          fc_abajo:= Persiana2_Atico_p2.fc_bajada,
                          reset_persiana:= Persiana2_Atico_p2.reset
                                    or %mw16043.2,
                          tiempo_para_error:= T#15s,
                          movimiento:= Persiana2_Atico_p2.movimiento,
                          error_motor_persiana
                          =>Persiana2_Atico_P2.error_persiana,
                          motor_subir => Persiana2_Atico_p2.motor_subir,
                          motor_bajar => Persiana2_Atico_p2.motor_bajar,
                          per_monitorizacion
                          => Persiana2_Atico_p2.monitorizacion);


(*-----------------------------------------------------------------*)
(*----Variables utilizadas para animación de las persianas en la
      pantalla táctil---*)
(*-----------------------------------------------------------------*)
%mw16020:= Persiana1_P0.movimiento;
%mw16021.3:= Persiana1_P0.fc_subida;
%mw16021.4:= Persiana1_P0.fc_bajada;
%mw16021.5:= Persiana1_P0.error_persiana;

%mw16022:= Persiana2_P0.movimiento;
%mw16023.3:= Persiana2_P0.fc_subida;
%mw16023.4:= Persiana2_P0.fc_bajada;
%mw16023.5:= Persiana2_P0.error_persiana;

%mw16024:= Persiana3_P0.movimiento;
%mw16025.3:= Persiana3_P0.fc_subida;
```

```
%mw16025.4:= Persiana3_P0.fc_bajada;
%mw16025.5:= Persiana3_P0.error_persiana;

%mw16026:= Persiana4_P0.movimiento;
%mw16027.3:= Persiana4_P0.fc_subida;
%mw16027.4:= Persiana4_P0.fc_bajada;
%mw16027.5:= Persiana4_P0.error_persiana;

%mw16028:= Persiana5_P0.movimiento;
%mw16029.3:= Persiana5_P0.fc_subida;
%mw16029.4:= Persiana5_P0.fc_bajada;
%mw16029.5:= Persiana5_P0.error_persiana;

(*planta 1*)
%mw16030:= Persiana1_Salon_P1.movimiento;
%mw16031.3:= Persiana1_Salon_P1.fc_subida;
%mw16031.4:= Persiana1_Salon_P1.fc_bajada;
%mw16031.5:= Persiana1_Salon_P1.error_persiana;

%mw16032:= Persiana_Cocina_P1.movimiento;
%mw16033.3:= Persiana_Cocina_P1.fc_subida;
%mw16033.4:= Persiana_Cocina_P1.fc_bajada;
%mw16033.5:= Persiana_Cocina_P1.error_persiana;

%mw16034:= Persiana_Hab1_P1.movimiento;
%mw16035.3:= Persiana_Hab1_P1.fc_subida;
%mw16035.4:= Persiana_Hab1_P1.fc_bajada;
%mw16035.5:= Persiana_Hab1_P1.error_persiana;

%mw16036:= Persiana_Hab2_P1.movimiento;
%mw16037.3:= Persiana_Hab2_P1.fc_subida;
%mw16037.4:= Persiana_Hab2_P1.fc_bajada;
%mw16037.5:= Persiana_Hab2_P1.error_persiana;

%mw16038:= Persiana_Hab3_P1.movimiento;
%mw16039.3:= Persiana_Hab3_P1.fc_subida;
%mw16039.4:= Persiana_Hab3_P1.fc_bajada;
%mw16039.5:= Persiana_Hab3_P1.error_persiana;

(*planta 2*)
%mw16040:= Persiana1_Atico_p2.movimiento;
%mw16041.3:= Persiana1_Atico_p2.fc_subida;
%mw16041.4:= Persiana1_Atico_p2.fc_bajada;
%mw16041.5:= Persiana1_Atico_p2.error_persiana;

%mw16042:= Persiana2_Atico_p2.movimiento;
%mw16043.3:= Persiana2_Atico_p2.fc_subida;
%mw16043.4:= Persiana2_Atico_p2.fc_bajada;
%mw16043.5:= Persiana2_Atico_p2.error_persiana;

%mw16044:= Persiana3_Atico_P2.movimiento;(*corresponde a la planta 2
no a la baja*)
%mw16045.3:= Persiana3_Atico_P2.fc_subida;
%mw16045.4:=Persiana3_Atico_P2.fc_bajada;
%mw16045.5:= Persiana3_Atico_P2.error_persiana;

%mw16046:= Persiana4_Atico_P2.movimiento;
%mw16047.3:= Persiana4_Atico_P2.fc_subida;
%mw16047.4:= Persiana4_Atico_P2.fc_bajada;
%mw16047.5:= Persiana4_Atico_P2.error_persiana;
```

```
(*Variables utilizadas en la pantalla principal de las persianas.
Sumario de errores*)
%mw16048.0:= Persiana1_P0.error_persiana or
Persiana2_P0.error_persiana or Persiana3_P0.error_persiana or
Persiana4_P0.error_persiana or Persiana5_P0.error_persiana ;
%mw16048.1:= Persiana1_Salon_P1.error_persiana or
Persiana_Cocina_P1.error_persiana or Persiana_Hab1_P1.error_persiana
or Persiana_Hab2_P1.error_persiana or Persiana_Hab3_P1.error_persiana ;
%mw16048.2:= Persiana4_Atico_P2.error_persiana or
Persiana3_Atico_P2.error_persiana or Persiana1_Atico_p2.error_persiana
or Persiana2_Atico_p2.error_persiana ;

(*Estas variables contienen los siguientes valores de ayuda en Sript
de texto dinámico:*)
(*   1:Persiana totalmente bajada                *)
(*   2:Persiana totalmente subida                *)
(*   0: Persiana en posición intermedia          *)
%mw16049:= Persiana1_P0.monitorizacion;
%mw16050:= Persiana2_P0.monitorizacion;
%mw16051:= Persiana3_P0.monitorizacion;
%mw16052:= Persiana4_P0.monitorizacion;
%mw16053:= Persiana5_P0.monitorizacion ;

%mw16054:= Persiana1_Salon_P1.monitorizacion;
%mw16055:= Persiana_Cocina_P1.monitorizacion;
%mw16056:= Persiana_Hab1_P1.monitorizacion;
%mw16057:= Persiana_Hab2_P1.monitorizacion;
%mw16058:= Persiana_Hab3_P1.monitorizacion ;

%mw16059:= Persiana4_Atico_P2.monitorizacion;
%mw16060:= Persiana3_Atico_P2.monitorizacion;
%mw16061:= Persiana1_Atico_p2.monitorizacion;
%mw16062:= Persiana2_Atico_p2.monitorizacion ;
```

7.5.4.7. SECCIÓN DE PROGRAMA: CONTROL_SENSORES

Esta sección tiene por objeto invocar las instancias de los DFB´s de control de cada uno de los sensores de movimiento, así como preparar las variables que son necesarias para la facilitar la animación de los sensores en la pantalla táctil.

El código implementado en esta sección es el mostrado a continuación:

```
(*-------------------------------------------------------------------*)
(*-Instancias de control (DFB´s) Sensores de movimiento Planta 0----*)
(*-------------------------------------------------------------------*)

(* Todos los sensores de movimiento comparten la misma entrada de
Sistema conectado *)
Sensor1_Alarma_P0.sistema_conectado:=
Sensor2_Alarma_P0.sistema_conectado:=
Sensor_Cocina_Alarma_P1.sistema_conectado:=
Sensor_Salon_Alarma_P1.sistema_conectado:=
```

```
Sensor_Pasillo_Alarma_P1.sistema_conectado:=
Sensor_Atico_Alarma_P2.sistema_conectado:=
Sensor_Habitacion1_Alarma_P1.sistema_conectado:=
Sensor_Habitacion2_Alarma_P1.sistema_conectado:=
Sensor_Habitacion3_Alarma_P1.sistema_conectado:=sistema_alarma_activad
o ;

cntl_sensor1_alarma_p0 (conectar:= Sensor1_Alarma_P0.conectar,
                        rearme:= Sensor1_Alarma_P0.rearme,
                        sistema_conectado:=
                              Sensor1_Alarma_P0.sistema_conectado,
                        tiempo_retardo:= T#5s,
                        movimiento:= Sensor1_Alarma_P0.movimiento,
                        alarma => Sensor1_Alarma_P0.alarma,
                        desconectado => Sensor1_Alarma_P0.desconectado,
                        conf=> Sensor1_Alarma_P0.codificacion);

cntl_sensor2_alarma_p0 (conectar:= Sensor2_Alarma_P0.conectar,
                        rearme:= Sensor2_Alarma_P0.rearme,
                        sistema_conectado:=
                              Sensor2_Alarma_P0.sistema_conectado,
                        tiempo_retardo:= T#5s,
                        movimiento:= Sensor2_Alarma_P0.movimiento,
                        alarma => Sensor2_Alarma_P0.alarma,
                        desconectado => Sensor2_Alarma_P0.desconectado,
                        conf=> Sensor2_Alarma_P0.codificacion);

(*-----------------------------------------------------------------*)
(*Instancias de control (DFB´s) Sensores de movimiento Planta 1--*)
(*-----------------------------------------------------------------*)

cntl_sensor_cocina_alarma_p1 (conectar:=
Sensor_Cocina_Alarma_P1.conectar,
                              rearme:= Sensor_Cocina_Alarma_P1.rearme,
                              sistema_conectado:=
Sensor_Cocina_Alarma_P1.sistema_conectado,
                              tiempo_retardo:= T#2s,
                              movimiento:=
Sensor_Cocina_Alarma_P1.movimiento,
                              alarma => Sensor_Cocina_Alarma_P1.alarma,
                              desconectado
                              => Sensor_Cocina_Alarma_P1.desconectado,
                              Conf
                              => Sensor_Cocina_Alarma_P1.codificacion);

cntl_sensor_salon_alarma_p1 (conectar:=
                             Sensor_Salon_Alarma_P1.conectar,
                             rearme:= Sensor_Salon_Alarma_P1.rearme,
                             sistema_conectado:=
Sensor_Salon_Alarma_P1.sistema_conectado,
                             tiempo_retardo:= T#2s,
                             movimiento:=
Sensor_Salon_Alarma_P1.movimiento,
                             alarma => Sensor_Salon_Alarma_P1.alarma,
                             desconectado
                             => Sensor_Salon_Alarma_P1.desconectado,
                             Conf
                             => Sensor_Salon_Alarma_P1.codificacion);

cntl_sensor_pasillo_alarma_p1 (conectar:=
                               Sensor_Pasillo_Alarma_P1.conectar,
```

```
                              rearme:= Sensor_Pasillo_Alarma_P1.rearme,
                              sistema_conectado:=
Sensor_Pasillo_Alarma_P1.sistema_conectado,
                              tiempo_retardo:= T#2s,
                              movimiento:=
                              Sensor_Pasillo_Alarma_P1.movimiento,
                              alarma => Sensor_Pasillo_Alarma_P1.alarma,
                              desconectado
                              => Sensor_Pasillo_Alarma_P1.desconectado,
                              Conf
                              => Sensor_Pasillo_Alarma_P1.codificacion);

cntl_sensor_hab1_alarma_p1 (conectar:=
                              Sensor_Habitacion1_Alarma_P1.conectar,
                            rearme:=
                              Sensor_Habitacion1_Alarma_P1.rearme,
                            sistema_conectado:=
                              Sensor_Habitacion1_Alarma_P1.sistema_cone
ctado,
                            tiempo_retardo:= T#2s,
                            movimiento:=
                              Sensor_Habitacion1_Alarma_P1.movimiento,
                            alarma
                            => Sensor_Habitacion1_Alarma_P1.alarma,
                            desconectado
                            =>Sensor_Habitacion1_Alarma_P1.desconectado,
                            Conf
                            =>Sensor_Habitacion1_Alarma_P1.codificacion
);

cntl_sensor_hab2_alarma_p1 (conectar:=
                              Sensor_Habitacion2_Alarma_P1.conectar,
                            rearme:=
                              Sensor_Habitacion2_Alarma_P1.rearme,
                            sistema_conectado:=
                  Sensor_Habitacion2_Alarma_P1.sistema_conectado,
                              tiempo_retardo:= T#2s,
                              movimiento:=
                              Sensor_Habitacion2_Alarma_P1.movimiento,
                              alarma
                              => Sensor_Habitacion2_Alarma_P1.alarma,
                              Desconectado
                              =>
Sensor_Habitacion2_Alarma_P1.desconectado,
                              Conf
                              =>Sensor_Habitacion2_Alarma_P1.codificacion
);

cntl_sensor_hab3_alarma_p1 (conectar:=
                              Sensor_Habitacion3_Alarma_P1.conectar,
                            rearme:=
                          Sensor_Habitacion3_Alarma_P1.rearme,
                            sistema_conectado:=
                          Sensor_Habitacion3_Alarma_P1.sistema_conectado,
                            tiempo_retardo:= T#2s,
                            movimiento:=
                              Sensor_Habitacion3_Alarma_P1.movimiento,
                            alarma
                            => Sensor_Habitacion3_Alarma_P1.alarma,
                            desconectado
                            =>  Sensor_Habitacion3_Alarma_P1.desconectado,
```

```
                          Conf
                          =>Sensor_Habitacion3_Alarma_P1.codificacion);

(*-------------------------------------------------------------*)
(*Instancias de control (DFB´s) Sensores de movimiento Planta 2--*)
(*-------------------------------------------------------------*)

cntl_sensor_atico_alarma_p2 (conectar:=
                              Sensor_atico_Alarma_P2.conectar,
                             rearme:= Sensor_atico_Alarma_P2.rearme,
                             sistema_conectado:=
Sensor_atico_Alarma_P2.sistema_conectado,
                             tiempo_retardo:= T#2s,
                             movimiento:=
Sensor_atico_Alarma_P2.movimiento,
                             alarma => Sensor_atico_Alarma_P2.alarma,
                             desconectado
                             => Sensor_atico_Alarma_P2.desconectado,
                             Conf
                             => Sensor_atico_Alarma_P2.codificacion);
```

7.5.4.8. SECCIÓN DE PROGRAMA: CONTROL_PUERTAS

Esta sección tiene por objeto invocar las instancias de los DFB´s de control de cada una de las dos puertas automáticas de la vivienda, así como preparar las variables que son necesarias para la facilitar la animación de las puertas en la pantalla táctil.

El código implementado en esta sección es el mostrado a continuación:

```
(*-------------------------------------------------------------------*)
(*Instancia de control (DFB´s) Puerta automática exterior vivienda--*)
(*-------------------------------------------------------------------*)
cntl_puerta_ext (ind_abt:= Puerta_ext.ind_abt,
           ind_cda:= Puerta_ext.ind_cda,
           fotocel_interrumpida:= Puerta_ext.fotocel_interrumpida,
           rearme:= Puerta_ext.rearme or %MW16009.2 ,
           ind_sobrecalentamiento:= Puerta_ext.ind_sobrecalentamiento,
           boton_abrir:= Puerta_ext.boton_abrir or %MW16009.4 ,
           boton_cerrar:= Puerta_ext.boton_cerrar or %MW16009.5,
           boton_cerrojo:= Puerta_ext.boton_cerrojo or %MW16009.6,
           salida_aviso_puerta_moviendo
           => Puerta_ext.salida_aviso_puerta_moviendo,
           salida_motor_abrir => Puerta_ext.salida_motor_abrir,
           salida_motor_cerrar => Puerta_ext.salida_motor_cerrar,
           salida_error =>Puerta_ext.salida_error );


(*-------------------------------------------------------------------*)
(*Instancia de control (DFB´s) Puerta automática acceso a vivienda--*)
(*-------------------------------------------------------------------*)

cntl_puerta_garaje (ind_abt:= Puerta_garaje.ind_abt,
         ind_cda:= Puerta_garaje.ind_cda,
         fotocel_interrumpida:= Puerta_garaje.fotocel_interrumpida,
         rearme:= Puerta_garaje.rearme or %MW16010.2,
```

```
        ind_sobrecalentamiento:= Puerta_garaje.ind_sobrecalentamiento,
        boton_abrir:= Puerta_garaje.boton_abrir or %MW16010.4,
        boton_cerrar:= Puerta_garaje.boton_cerrar or %MW16010.5 ,
        boton_cerrojo:= Puerta_garaje.boton_cerrojo or %MW16010.6,
        salida_aviso_puerta_moviendo
        => Puerta_garaje.salida_aviso_puerta_moviendo,
        salida_motor_abrir => Puerta_garaje.salida_motor_abrir,
        salida_motor_cerrar => Puerta_garaje.salida_motor_cerrar,
        salida_error =>Puerta_garaje.salida_error );

(*SIMULACION PARA MAGELIS*)
(*puerta ext*)
IF (Puerta_ext.salida_motor_abrir) and fe(flanco_s6) then
      posicion_puerta_ext:= posicion_puerta_ext + 10 ;
end_if;
IF (Puerta_ext.salida_motor_cerrar) and fe(flanco_s6) then
      posicion_puerta_ext:= posicion_puerta_ext - 10 ;
end_if;

IF posicion_puerta_ext < 0 then
      posicion_puerta_ext:=0 ;
end_if;
IF posicion_puerta_ext > 95 then
      posicion_puerta_ext:= 95 ;
end_if;

if ( Puerta_ext.ind_cda)then
      posicion_puerta_ext:=0 ;
end_if;

if ( Puerta_ext.ind_abt)then
      posicion_puerta_ext:= 95 ;
end_if;

%MW16007:= posicion_puerta_ext;
%MW16009.0:= Puerta_ext.ind_cda;
%MW16009.1:= Puerta_ext.fotocel_interrumpida;
%MW16009.3:= Puerta_ext.ind_sobrecalentamiento;
%MW16009.7:= Puerta_ext.salida_aviso_puerta_moviendo;
%MW16009.8:=  Puerta_ext.salida_motor_abrir;
%MW16009.9:=  Puerta_ext.salida_motor_cerrar;
%MW16009.10:=  Puerta_ext.salida_error;
%MW16009.11:= Puerta_ext.ind_abt;

(*puerta garaje*)
IF (Puerta_garaje.salida_motor_abrir) and fe(flanco_s6) then
      posicion_puerta_garaje:= posicion_puerta_garaje + 10 ;
end_if;
IF (Puerta_garaje.salida_motor_cerrar) and fe(flanco_s6) then
      posicion_puerta_garaje:= posicion_puerta_garaje - 10 ;
end_if;

IF posicion_puerta_garaje < 0 then
      posicion_puerta_garaje:=0 ;
end_if;
IF posicion_puerta_garaje > 80 then
      posicion_puerta_garaje:= 80 ;
end_if;

if ( Puerta_garaje.ind_cda)then
      posicion_puerta_garaje:=0 ;
```

```
end_if;

if ( Puerta_garaje.ind_abt)then
      posicion_puerta_garaje:= 80 ;
end_if;

%MW16008:= posicion_puerta_garaje;
%MW16010.0:= Puerta_garaje.ind_cda;
%MW16010.1:= Puerta_garaje.fotocel_interrumpida;
%MW16010.2:= Puerta_garaje.rearme;
%MW16010.3:= Puerta_garaje.ind_sobrecalentamiento;

%MW16010.7:= Puerta_garaje.salida_aviso_puerta_moviendo;
%MW16010.8:=  Puerta_garaje.salida_motor_abrir;
%MW16010.9:=  Puerta_garaje.salida_motor_cerrar;
%MW16010.10:=  Puerta_garaje.salida_error;
%MW16010.11:= Puerta_garaje.ind_abt;
```

7.5.4.9. SECCIÓN DE PROGRAMA: CONTROL_ASCENSOR

Una forma sencilla de realizar el control del ascensor es viendo este como una máquina de estados (FSM). El diagrama de estados de esta y las transiciones que provoca los cambios de estados, son los que se indican en la siguiente figura

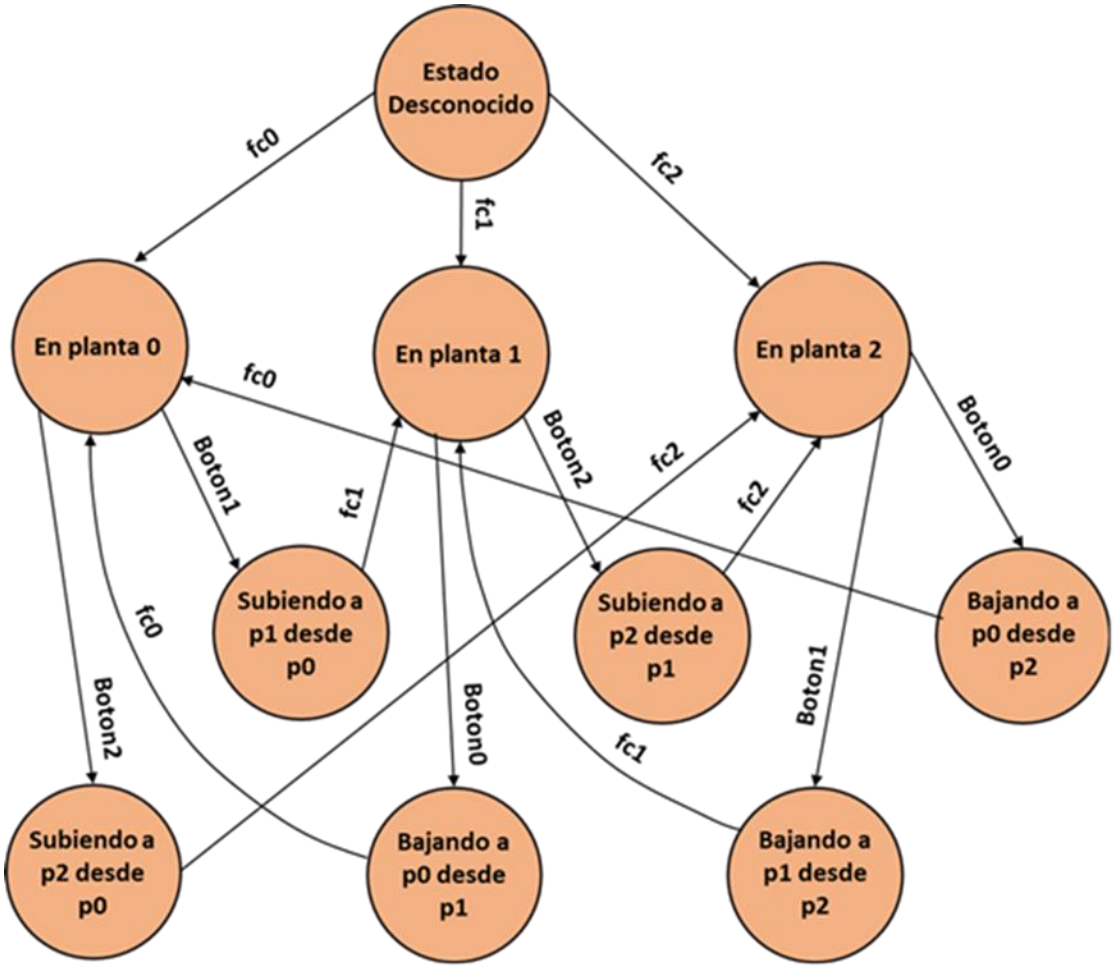


Figura 7-25

El código implementado en esta sección es el mostrado a continuación e implementa la máquina de estado mostrada anteriormente:

```
(*----------------------------------*)
(*----Control Ascensor vivienda-----*)
(*----------------------------------*)

if %s0 or %s1 or %s13 then
(* En el arranque, en frio (%S0), en caliente (%S1) o tras un paso a
RUN (%S13)      *)
(* inicializamos ciertas variables que nos ayudarán a entender la
máquina de estados *)
```

```
      EnPlanta0:= 0 ;
      EnPlanta1:= 1 ;
      EnPlanta2:= 2 ;
      SubidaPlanta1Desde0:= 3 ;
      SubidaPlanta2Desde0:= 4 ;
      SubidaPlanta2Desde1:= 5 ;
      BajadaPlanta0Desde1:= 6 ;
      BajadaPlanta0Desde2:= 7 ;
      BajadaPlanta1Desde2:= 8 ;
      Desconocido:= 9 ;
      (* Por seguridad reseteamos las ordenes a los motores *)
      reset(motor_baja);
      reset(motor_sube);
      (* Por seguridad, inicializamos lo seguros de las puertas a 0 y
los ponemos a 1 solo para abrir*)
      reset(cerrojo_puerta_P0);
      reset(cerrojo_puerta_P1);
      reset(cerrojo_puerta_P2);
      (*En el arranque hay que mirar en que planta se encuentra el
ascensor*)
      if fc0 then estado_ascensor:= Enplanta0 ; end_if; (* Caso de
estar en Planta 0 *)
      if fc1 then estado_ascensor:= Enplanta1 ; end_if; (* Caso de
estar en Planta 1 *)
      if fc2 then estado_ascensor:= Enplanta2 ; end_if; (* Caso de
estar en Planta 2 *)
      if (not fc0) and (not fc1) and (not fc2) then estado_ascensor:=
Desconocido ; end_if;
end_if;
(* Variable que indica que todas las puertas están cerradas
*)
(* Para asegurarnos que cualquier operacion de subida o bajada del
ascensor *)
(* tiene todas las puertas cerradas
*)
todas_puertas_cdas:= (not puerta_P0_abt) and (not puerta_P1_abt) and
(not puerta_P2_abt);
 (**)
en_alguna_planta:= (estado_ascensor = EnPlanta0) or (estado_ascensor =
EnPlanta1)or (estado_ascensor = EnPlanta2) ;
(* Antes de hacer efectiva la pulsación de cualquier botón, esperamos
5 seg *)
(* por si hay que abrir la puerta de salida del ascensor
*)
T_espera_en_planta(IN:= en_alguna_planta,PT:= t#5s);

(*-----------------------------------------------*)
(*--------Maquina de estados ASCENSOR----------*)
(*-----------------------------------------------*)

(* Las variables %MW16006.x en paralelo con los botones se utilizan
para llamar al ascensor desde la pantalla táctil*)

if (estado_ascensor = EnPlanta0 )and (boton_1 or %MW16006.10)and
todas_puertas_cdas and (not T_espera_en_planta.q)  then
      estado_ascensor:= SubidaPlanta1Desde0 ;
end_if;
if (estado_ascensor = EnPlanta0 )and (boton_2 or %MW16006.11) and
todas_puertas_cdas and (not T_espera_en_planta.q) then
      estado_ascensor:= SubidaPlanta2Desde0 ;
end_if;
```

```
if (estado_ascensor = EnPlanta1 )and (boton_0 or %MW16006.9) and
todas_puertas_cdas  and (not T_espera_en_planta.q) then
      estado_ascensor:= BajadaPlanta0Desde1 ;
end_if;
if (estado_ascensor = EnPlanta1 )and (boton_2 or %MW16006.11) and
todas_puertas_cdas and (not T_espera_en_planta.q) then
      estado_ascensor:= SubidaPlanta2Desde1 ;
end_if;
if (estado_ascensor = EnPlanta2 )and (boton_0 or %MW16006.9) and
todas_puertas_cdas and (not T_espera_en_planta.q) then
      estado_ascensor:= BajadaPlanta0Desde2 ;
end_if;
if (estado_ascensor = EnPlanta2 )and (boton_1 or %MW16006.10) and
todas_puertas_cdas and (not T_espera_en_planta.q) then
      estado_ascensor:= BajadaPlanta1Desde2 ;
end_if;
if((estado_ascensor = BajadaPlanta0Desde1)or(estado_ascensor =
BajadaPlanta0Desde2)or( estado_ascensor = Desconocido))  and fc0  then
      estado_ascensor:= EnPlanta0 ;
end_if;
if((estado_ascensor = BajadaPlanta1Desde2)or(estado_ascensor =
SubidaPlanta1Desde0)) and fc1 then
      estado_ascensor:= EnPlanta1 ;
end_if;
if((estado_ascensor = SubidaPlanta2Desde0)or(estado_ascensor =
SubidaPlanta2Desde1)) and fc2 then
      estado_ascensor:= EnPlanta2 ;
end_if;

(* En función del estado de la Máquina de estados, asignamos las
acciones (ordenes) *)
(* que hay que mandar a los actuadores del ascensor
*)
case estado_ascensor of
      0: (*EnPlanta0*)
            set(cerrojo_puerta_P0); (* Se permite abrir la puerta *)
            reset(motor_baja);(* Se da orden de parar el giro de motor
en sentido descendente *)
            reset(motor_sube);(* Se da orden de parar el giro de motor
en sentido ascendente *)
      1: (*EnPlanta1*)
            set(cerrojo_puerta_P1);(* Se permite abrir la puerta *)
            reset(motor_baja);(* Se da orden de parar el giro de motor
en sentido descendente *)
            reset(motor_sube);(* Se da orden de parar el giro de motor
en sentido ascendente *)
      2: (*EnPlanta2*)
            set(cerrojo_puerta_P2);(* Se permite abrir la puerta *)
            reset(motor_baja);(* Se da orden de parar el giro de motor
en sentido descendente *)
            reset(motor_sube);(* Se da orden de parar el giro de motor
en sentido ascendente *)
      3: (*SubidaPlanta1Desde0*)
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_sube);(* Se da orden de giro al motor en sentido
ascendente *)
      4: (*SubidaPlanta2Desde0*)
```

```
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_sube);(* Se da orden de giro al motor en sentido
ascendente *)
      5: (*SubidaPlanta2Desde1*)
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_sube);(* Se da orden de giro al motor en sentido
ascendente *)
      6: (*bajadaPlanta0Desde1*)
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_baja);(* Se da orden de giro al motor en sentido
descendente *)
      7: (*BajadaPlanta0Desde2*)
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_baja);(* Se da orden de giro al motor en sentido
descendente *)
      8: (*BajadaPlanta1Desde2*)
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_baja);(* Se da orden de giro al motor en sentido
descendente *)
      9: (*desconocido*)
           reset(cerrojo_puerta_P0);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P1);(* No se permite abrir la puerta *)

           reset(cerrojo_puerta_P2);(* No se permite abrir la puerta *)
            set(motor_baja);(* Se da orden de giro al motor en sentido
descendente *)
end_case;

(*-------------------------------------------------------------------*)
(*Variables utilizadas para animación del ascensor en la pantalla
táctil*)
(*-------------------------------------------------------------------*)
%MW16005:= estado_ascensor;
%MW16004:= posicion_ascensor;
%MW16006.0:= cerrojo_puerta_P0;
%MW16006.1:= cerrojo_puerta_P1;
%MW16006.2:= cerrojo_puerta_P2;
%MW16006.3:= puerta_P0_abt;
%MW16006.4:= puerta_P1_abt;
%MW16006.5:= puerta_P2_abt;
```

```
%MW16006.6:=fc0;
%MW16006.7:=fc1;
%MW16006.8:=fc2;

(* Movimiento de la cabina del ascensor*)
IF (motor_sube) and fe(flanco_s6) then
      posicion_ascensor:= posicion_ascensor + 5 ;
end_if;
IF (motor_baja) and fe(flanco_s6) then
      posicion_ascensor:= posicion_ascensor - 5 ;
end_if;
IF posicion_ascensor < 0 then
      posicion_ascensor:=0 ;
end_if;
IF posicion_ascensor > 88 then
      posicion_ascensor:=88 ;
end_if;
if (fc0)then
      posicion_ascensor:=0 ;
end_if;
if (fc1)then
      posicion_ascensor:=45 ;
end_if;
if (fc2)then
      posicion_ascensor:=88 ;
end_if;
```

7.5.4.10. SECCIÓN DE PROGRAMA: SALIDAS_CAMPO

Esta sección tiene por objeto asignar a las salidas de campo en Advantys las órdenes para activarlas.

La ventaja que comporta el uso de esta sección es que en el caso de tener que cambiar la dirección de memoria de una variable de campo, no es necesario buscar su valor en todas las secciones del programa. Basta con modificarla en esta sección.

El código implementado en esta sección es el mostrado a continuación:

```
(*--------------------------------------------------------------------*)
(*----Asignación de las salidas (estructuras de datos) hacia periferia
distribuida (advantys---*)
(*--------------------------------------------------------------------*)

(* Salida hacia campo de Planta 0 *)
Esc_Adts_Planta_0[0].0:=Puerta_ext.salida_aviso_puerta_moviendo;(*Puer
ta  deslizante exterior*)
Esc_Adts_Planta_0[0].1:=Puerta_ext.salida_motor_abrir;(*Puerta
deslizante exterior*)
Esc_Adts_Planta_0[0].2:=Puerta_ext.salida_motor_cerrar;(*Puerta
deslizante exterior*)
Esc_Adts_Planta_0[0].3:=Puerta_garaje.salida_aviso_puerta_moviendo;(*P
uerta  garaje*)
Esc_Adts_Planta_0[0].4:=Puerta_garaje.salida_motor_abrir;(*Puerta
garaje*)
```

```
Esc_Adts_Planta_0[0].5:=Puerta_garaje.salida_motor_cerrar;(*Puerta
garaje*)
Esc_Adts_Planta_0[1].0:=Persiana1_P0.motor_subir;(*Persiana 1 Planta
0*)
Esc_Adts_Planta_0[1].1:=Persiana1_P0.motor_bajar;(*Persiana 1 Planta
0*)
Esc_Adts_Planta_0[1].2:=Persiana2_P0.motor_subir;(*Persiana 2 Planta
0*)
Esc_Adts_Planta_0[1].3:=Persiana2_P0.motor_bajar;(*Persiana 2 Planta
0*)
Esc_Adts_Planta_0[1].4:=Persiana3_P0.motor_subir;(*Persiana 3 Planta
0*)
Esc_Adts_Planta_0[1].5:=Persiana3_P0.motor_bajar;(*Persiana 3 Planta
0*)
Esc_Adts_Planta_0[2].0:=Persiana4_P0.motor_subir;(*Persiana 4 Planta
0*)
Esc_Adts_Planta_0[2].1:=Persiana4_P0.motor_bajar;(*Persiana 4 Planta
0*)
Esc_Adts_Planta_0[2].2:=Persiana5_P0.motor_subir;(*Persiana 5 Planta
0*)
Esc_Adts_Planta_0[2].3:=Persiana5_P0.motor_bajar;(*Persiana 5 Planta
0*)
Esc_Adts_Planta_0[2].4:=motor_baja;(*Ascensor*)
Esc_Adts_Planta_0[2].5:=motor_sube;(*Ascensor*)
Esc_Adts_Planta_0[3].0:=not cerrojo_puerta_P0;(*Ascensor*)

(* Salida hacia campo de Planta 0 *)
Esc_Adts_Planta_1[0].0:=Persiana_Cocina_P1.motor_subir;(*Persiana 1
Cocina Planta 1*)
Esc_Adts_Planta_1[0].1:=Persiana_Cocina_P1.motor_bajar;(*Persiana 1
Cocina Planta 1*)
Esc_Adts_Planta_1[0].2:=Persiana1_Salon_P1.motor_subir;(*Persiana 1
Salon Planta 1*)
Esc_Adts_Planta_1[0].3:=Persiana1_Salon_P1.motor_bajar;(*Persiana 1
Salon Planta 1*)
Esc_Adts_Planta_1[0].4:=Persiana_Hab1_P1.motor_subir;(*Persiana
habitacion 1 Planta 1*)
Esc_Adts_Planta_1[0].5:=Persiana_Hab1_P1.motor_bajar;(*Persiana
habitacion 1 Planta 1*)
Esc_Adts_Planta_1[1].0:=Persiana_Hab2_P1.motor_subir;(*Persiana
habitacion 2 Planta 1*)
Esc_Adts_Planta_1[1].1:=Persiana_Hab2_P1.motor_bajar;(*Persiana
habitacion 2 Planta 1*)
Esc_Adts_Planta_1[1].2:=Persiana_Hab3_P1.motor_subir;(*Persiana
habitacion 3 Planta 1*)
Esc_Adts_Planta_1[1].3:=Persiana_Hab3_P1.motor_bajar;(*Persiana
habitacion 3 Planta 1*)
Esc_Adts_Planta_1[1].4:=not cerrojo_puerta_P1;(*Ascensor*)
Esc_Adts_Planta_1[1].5:=%MW16002.3 ;(*sistema alarma*)

(* Salida hacia campo de Planta 0 *)
Esc_Adts_Planta_2[0].0:=Persiana1_Atico_P2.motor_subir;(*Persiana1
atico Planta 2*)
Esc_Adts_Planta_2[0].1:=Persiana1_Atico_P2.motor_bajar;(*Persiana1
atico Planta 2*)
Esc_Adts_Planta_2[0].2:=Persiana2_Atico_P2.motor_subir;(*Persiana2
atico Planta 2*)
Esc_Adts_Planta_2[0].3:=Persiana2_Atico_P2.motor_bajar;(*Persiana2
atico Planta 2*)
Esc_Adts_Planta_2[0].4:=Persiana4_Atico_P2.motor_subir;(*Persiana 4
atico Planta 2*)
```

```
Esc_Adts_Planta_2[0].5:=Persiana4_Atico_P2.motor_bajar;(*Persiana 4
atico Planta 2*)
Esc_Adts_Planta_2[1].0:=Persiana3_Atico_P2.motor_subir;(*Persiana 3
Atico Planta2*)
Esc_Adts_Planta_2[1].1:=Persiana3_Atico_P2.motor_bajar;(*Persiana 3
Atico Planta2*)
Esc_Adts_Planta_2[1].2:= not cerrojo_puerta_P2;(*Ascensor*)


(*asignacion de codificiacion sensores(empezamos en %MW15000) *)
%MW15000:= Sensor1_Alarma_P0.codificacion;
%MW15010:= Sensor2_Alarma_P0.codificacion;

%MW15020:= Sensor_Cocina_Alarma_P1.codificacion;
%MW15030:= Sensor_Salon_Alarma_P1.codificacion;
%MW15040:= Sensor_Pasillo_Alarma_P1.codificacion;
%MW15050:= Sensor_Habitacion1_Alarma_P1.codificacion;
%MW15060:= Sensor_Habitacion2_Alarma_P1.codificacion;
%MW15070:= Sensor_Habitacion3_Alarma_P1.codificacion;
%MW15080:= Sensor_atico_Alarma_P2.codificacion;
```

8. CONFIGURACIÓN Y DESARROLLO DEL SOFTWARE DEL TERMINAL

8.1. INTRODUCCIÓN AL ENTORNO DE DESARROLLO

Vijeo Designer es un software que proporciona todas las herramientas necesarias para la configuración y programación de los terminales HMI que ofrece Schneider Electric.

Mediante este software no solo permite desarrollar la aplicación, sino que también establece los protocolos necesarios para comunicar la pantalla con el PLC. En nuestro caso, Modbus TCP/IP.

8.1.1. LA INTERFAZ DE USUARIO

La forma de interactuar el usuario con **Vijeo Designer** es mediante ventanas configurables y a través del menú barras de herramientas. Con ellas se desarrollan las aplicaciones para que los usuarios interactúen. El aspecto de la interfaz se indica en la siguiente figura:

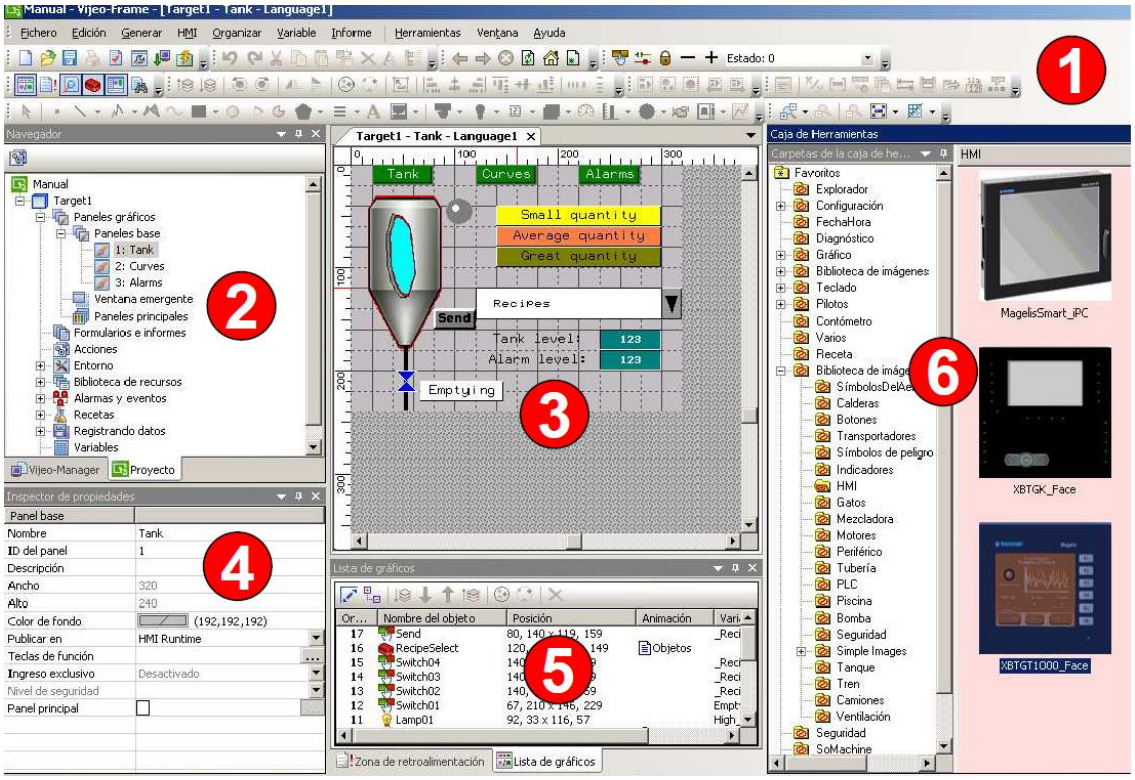


Figura 8-1 Interfaz de usuario Vijeo

1. Barras de herramientas: En esta área se muestra el menú de contexto y la barras con los iconos de las acciones que el desarrollador puede realizar en el proyecto.

2. Navegador: Esta área muestra el árbol de las diferentes partes que puede tener la aplicación.

3. Área de Trabajo: Es la zona donde el programador tiene que trabajar para crear la aplicación.

4. Inspector de Propiedades: En esta zona se muestra las propiedades de los objetos y en la que se puede cambiar y parametrizar sus valores.

5. Zona de retroalimentación: En esta área se verá los mensajes de compilación (errores de compilación y advertencias).

6. Caja de herramientas: Aquí se presentas los objetos e imágenes predefinidas (que vienen suministradas por defecto) y los que han sido creadas por el desarrollador.

8.2. CONFIGURACIÓN DESTINO

El primer paso que debemos realizar en el proyecto Vijeo-Designer es la configuración del destino. Es decir, hay que indicar cual es el hardware sobre el que la aplicación que desarrollemos será ejecutada.

Para ello, clicaremos con el botón derecho del ratón en el navegador y seleccionaremos menú destino y nos aparecerá el menú de configuración que se indica a continuación. En nuestro caso debemos seleccionar el tipo de terminal gráfico XBTGT5330 (640x480) y la dirección IP que tendrá la pantalla (**100.100.1.66**).

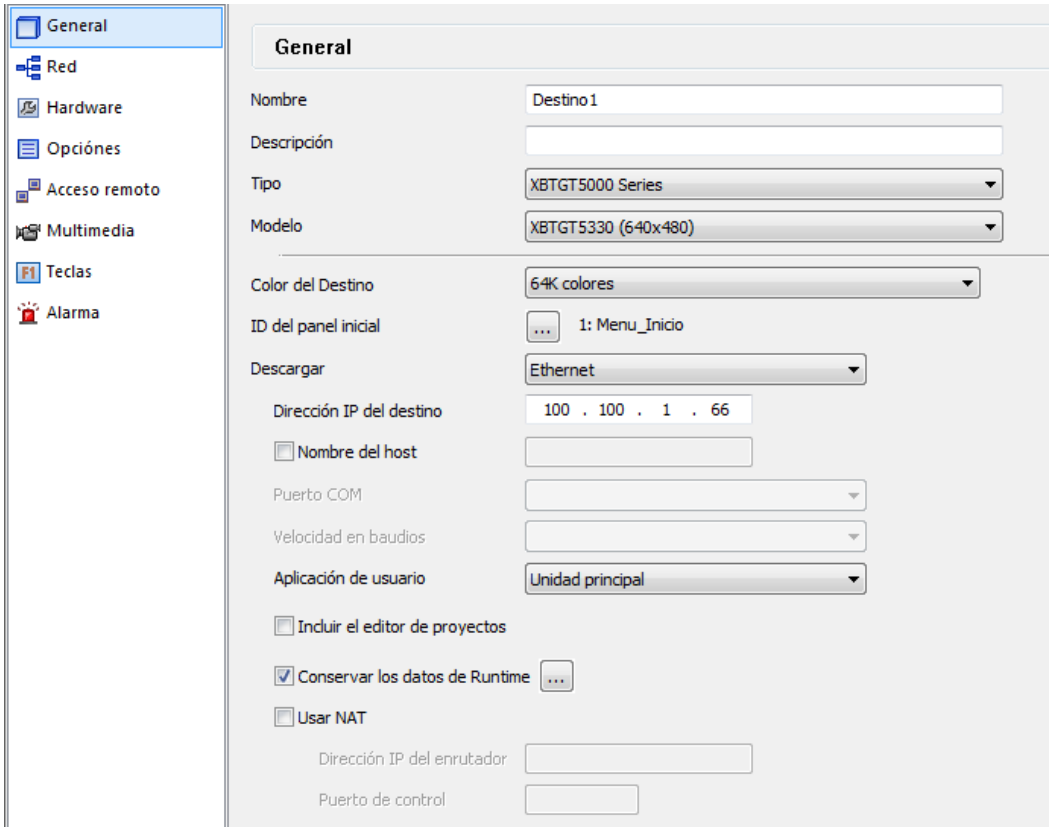


Figura 8-2 Configuración hardware equipo destino

8.3. CONFIGURAR LA COMUNICACIÓN DE HMI CON EL PLC

Para comunicarse con los PLC´s, en nuestro caso mediante ethernet, debemos añadir en el proyecto un **controlador**. Los controladores permiten que el usuario pueda comunicar con equipos sin tener de escribir código para enviar y o recibir datos.

Los pasos a seguir para añadir y configurar un controlador son los siguientes:

En la ventana del **navegador**, haciendo clic con el botón derecho en '**Administrador de E/S**' seleccionaremos '**Nuevo controlador**'.

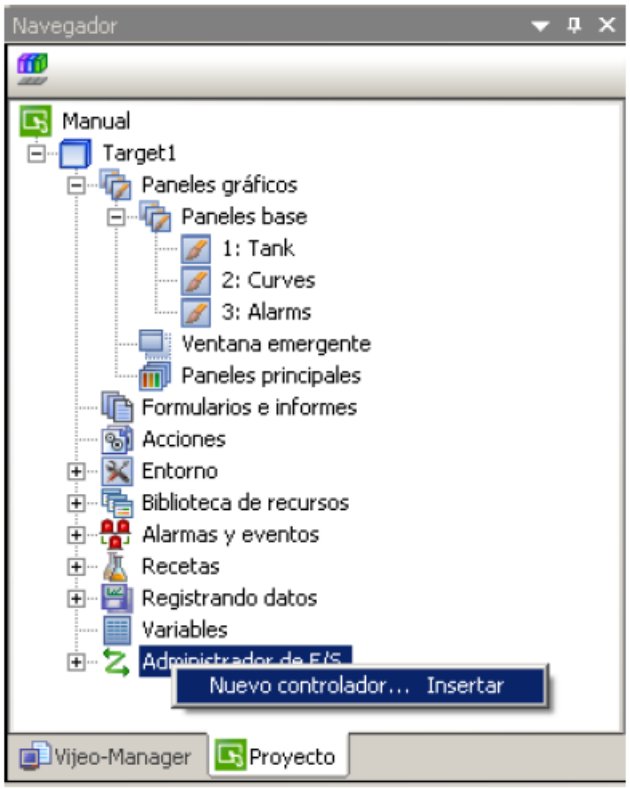


Figura 8-3 Proceso de configuración de un controlador

Nos aparecerá una nueva ventana donde debemos rellenar los campos 'Fabricante', 'Controlador' y 'Equipo' donde

- **Fabricante:** Es el fabricante del dispositivo con el cual queremos comunicar.
- **Controlador:** Es el nombre del controlador.
- **Equipo:** Es el modelo del dispositivo con el cual queremos comunicar.

Haremos clic en 'Aceptar' para añadir el controlador y el dispositivo seleccionados. En nuestro caso tendremos:

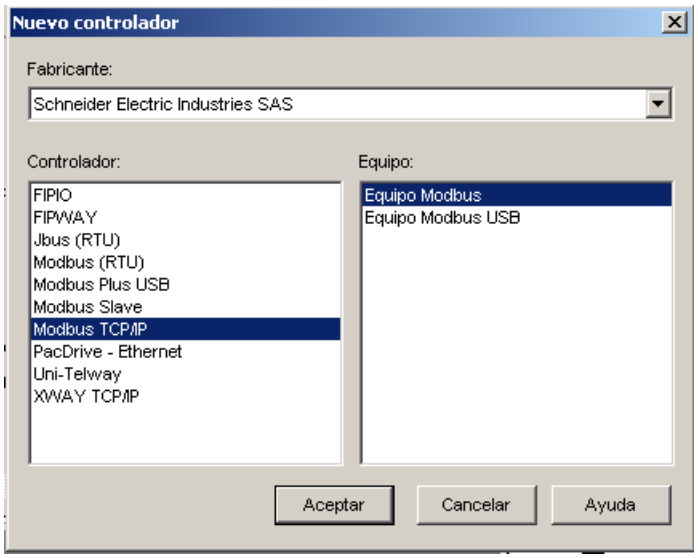


Figura 8-4 Proceso de configuración de un controlador

Tras añadir un controlador a Vijeo, configuraremos los parámetros de comunicación. Los parámetros de comunicación tienen dos partes: El controlador y el Equipo.

Para configurar los parámetros del controlador, haremos clic con el botón derecho del ratón en el controlador y, seguidamente, haremos clic '**configuración**'.

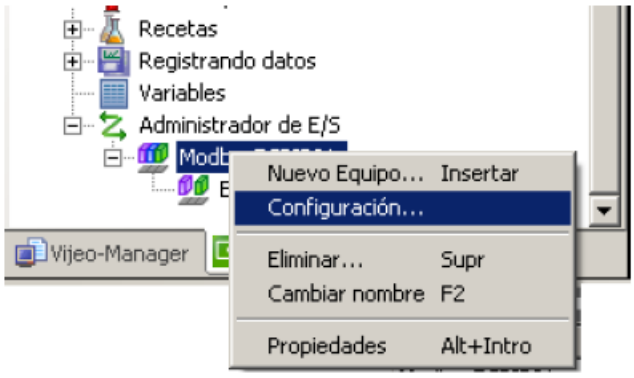


Figura 8-5 Proceso de configuración de un controlador

Seguidamente definiremos los ajustes de comunicación en el cuadro de diálogo '**Configuración del controlador**'. Dependiendo del controlador seleccionado, aparecerán unos campos u otros en el cuadro de diálogo. En nuestro caso Modbus TCP/IP.

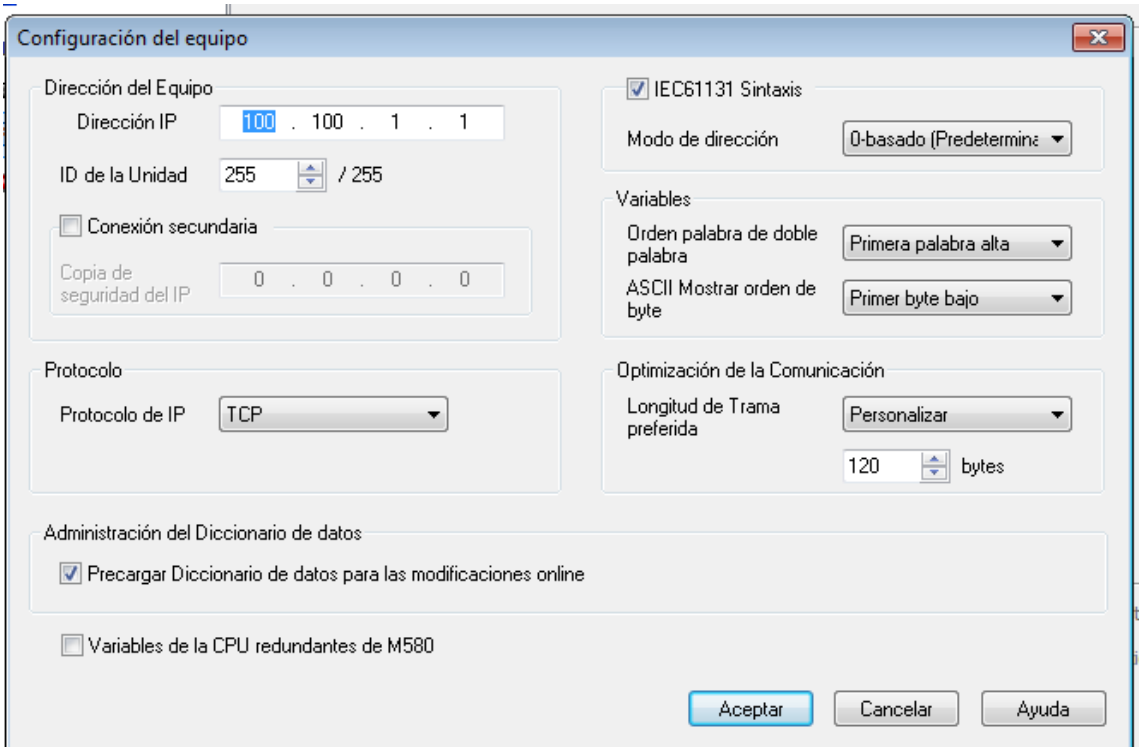


Figura 8-6 Configuración IP del controlador

8.4. PANELES

Un **panel** es una ventana en la que se colocan objetos (interruptores, pilotos, indicadores, etc.). Los paneles que se crean aparecerán en los terminales táctiles de destino una vez descargados a ellos.

La opción de paneles gráficos proporciona fundamentalmente dos tipos de paneles:

· **Paneles base**: Un panel base es el lienzo sobre el cual se dibuja cualquier objeto que deseamos mostrar. Es posible desplazarse entre paneles.

· **Paneles de ventana emergente**: Los paneles de una ventana emergente, tiene las mismas funciones que un panel base con la única diferencia que son llamados desde los paneles base y una vez cerrados se vuelve al mismo panel base desde el que se llamó.

8.4.1. CREACIÓN DE UN PANEL

En la ventana del Navegador, haremos clic derecho en los Paneles base en la carpeta Paneles gráficos y seleccionaremos ‘Nuevo panel’.

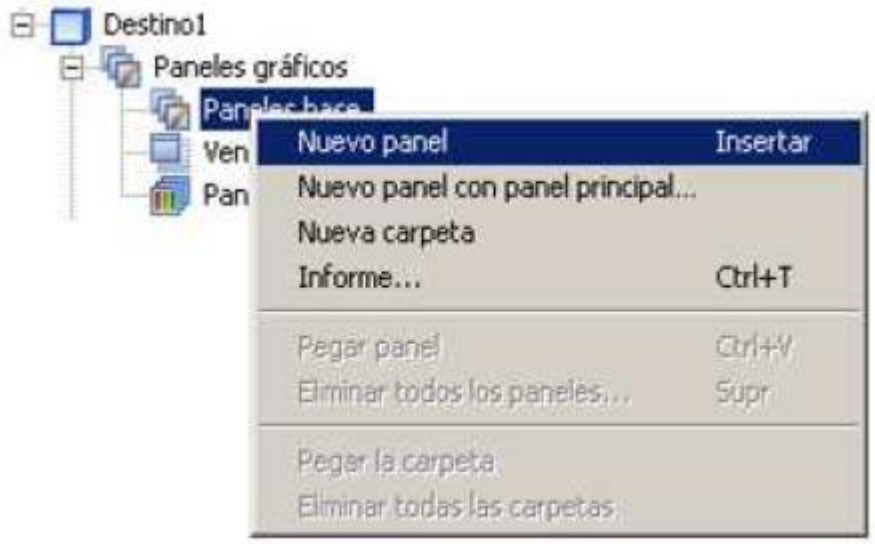


Figura 8-7 Creación de un nuevo panel

8.4.2. PANELES DEL PROYECTO

8.4.2.1. PANEL PRINCIPAL

El panel principal es el panel que aparece al dar por primera vez tensión al terminal táctil o al que podemos volver desde los demás paneles.

Como se aprecia en la figura, desde este panel podremos acceder a:

- Menú de Persianas
- Menú de Alarmas
- Menú Puertas
- Menú Ascensor

Figura 8-8 Menú principal Terminal táctil

8.4.2.1.1. SUBMENU PERSIANAS

Desde este menú podremos acceder a las persianas de las tres plantas de la vivienda, así como retroceder al menú principal.

En este panel, es posible observar mediante una indicación luminosa si alguna de las ventanas de la planta se encuentra en una situación de fallo o sin cerrar.

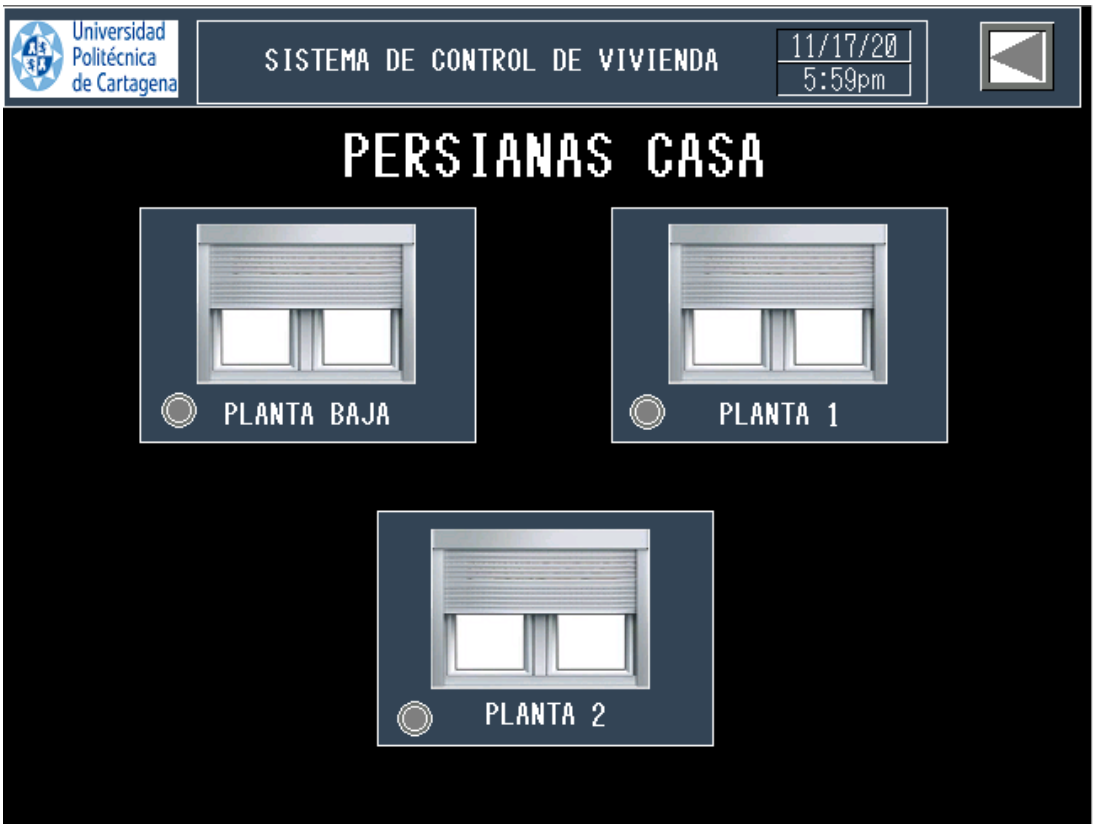


Figura 8-9 Menú principal persianas vivienda

8.4.2.1.1.1. SUBMENÚ PLANTA BAJA

En este submenú aparece el estado de detalle de la situación en la que se encuentran las persianas de la planta (subida, bajada, en posición intermedia). También es posible volver al menú de las persianas.

Figura 8-10 Submenú de detalle persianas planta 0

Pulsando sobre cada una de las persianas nos aparecerá una ventana emergente desde la cual podremos actuar sobre cada una ellas.

Figura 8-11 Pantalla emergente control persianas

8.4.2.1.1.2. SUBMENÚ PLANTA 1

En este submenú aparece el estado de detalle de la situación en la que se encuentran las persianas de la planta 1. También es posible volver al menú de las persianas.

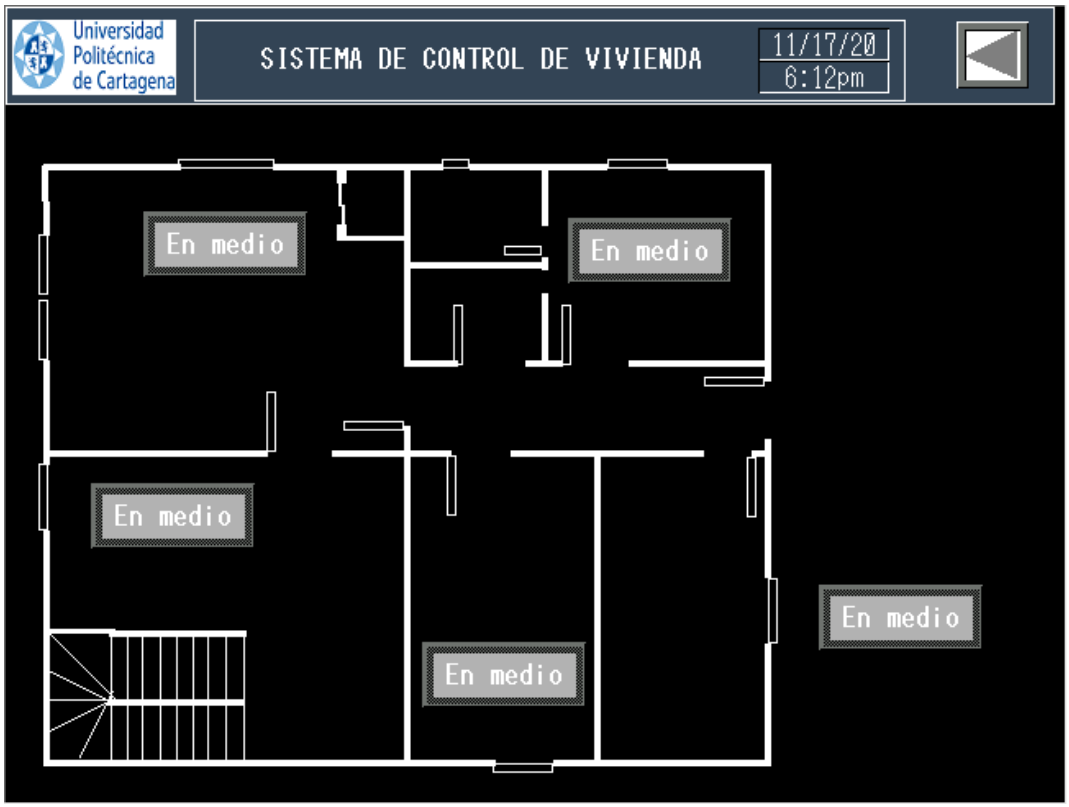


Figura 8-12 Submenú de detalle persianas planta 1

Al igual que en la planta anterior, es posible acceder a las ventanas emergentes de cada ventana para poder actuar sobre ellas

8.4.2.1.1.3. SUBMENÚ PLANTA 2

En este submenú aparece el estado de detalle de la situación en la que se encuentran las persianas de la planta 2. También es posible volver al menú de las persianas.

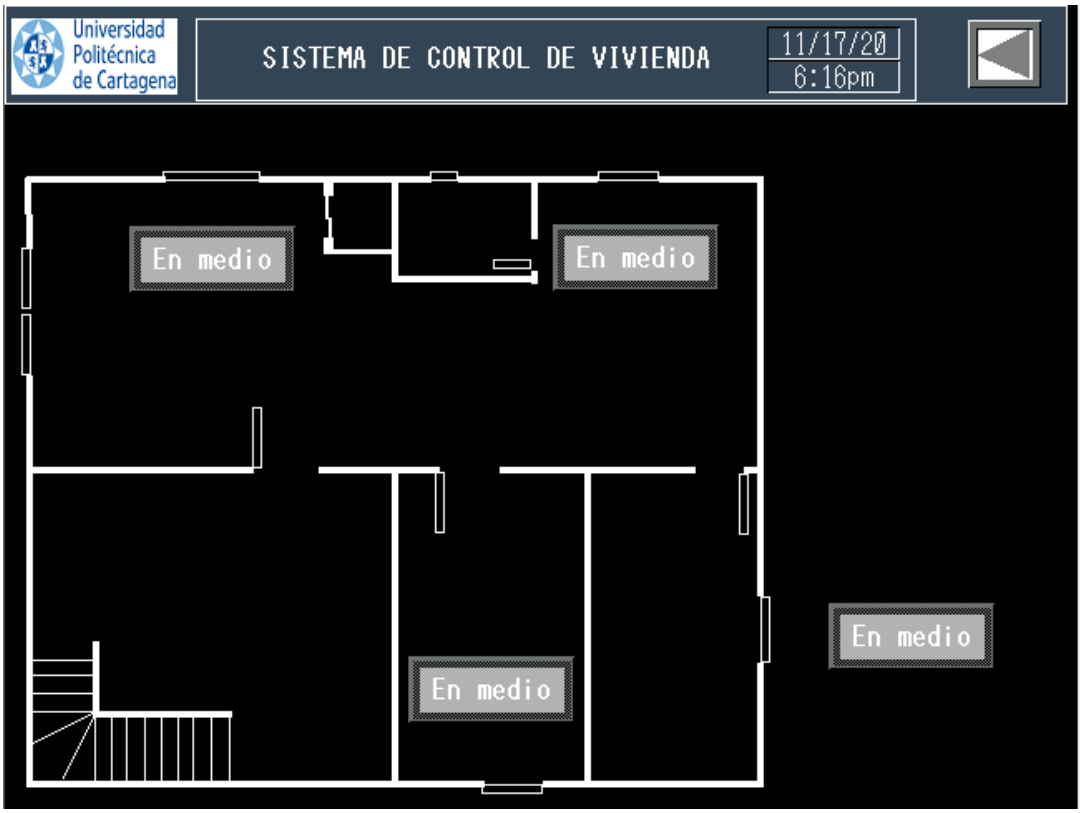


Figura 8-13 Submenú de detalle persianas planta 2

Al igual que en la planta anterior, es posible acceder a las ventanas emergentes de cada ventana para poder actuar sobre cada una de ellas

8.4.2.1.2. SUBMENU ALARMAS

Desde este menú podremos acceder al menú del sistema de alarmas y desde él, a la configuración del sistema y a cada uno de los detectores de presencia de cada planta.

Figura 8-14 Submenú de alarmas

8.4.2.1.2.1. SUBMENÚ DE CONFIGURACIÓN

Para acceder a este panel será necesario introducir un nombre y una contraseña.

Figura 8-15 Ventana de introducción usuario y contraseña

Una vez dentro de él, se muestra la siguiente pantalla desde la cual, podremos ver el estado de los sensores de cada una de las plantas o rearmar los sensores en caso de que se hayan activado, o activar (conectar) o desactivar el sistema de alarma.

Figura 8-16 Submenú de configuración de sensores de presencia

Una vez dentro de él, se muestra la siguiente pantalla desde la cual, podremos ver el estado de los sensores de cada una de las plantas o rearmar los sensores si se han activado, o activar(conectar) o desactivar el sistema de alarma.

En la siguiente figura se muestra el detalle de la posición en la que están instalados los sensores de la planta 0.

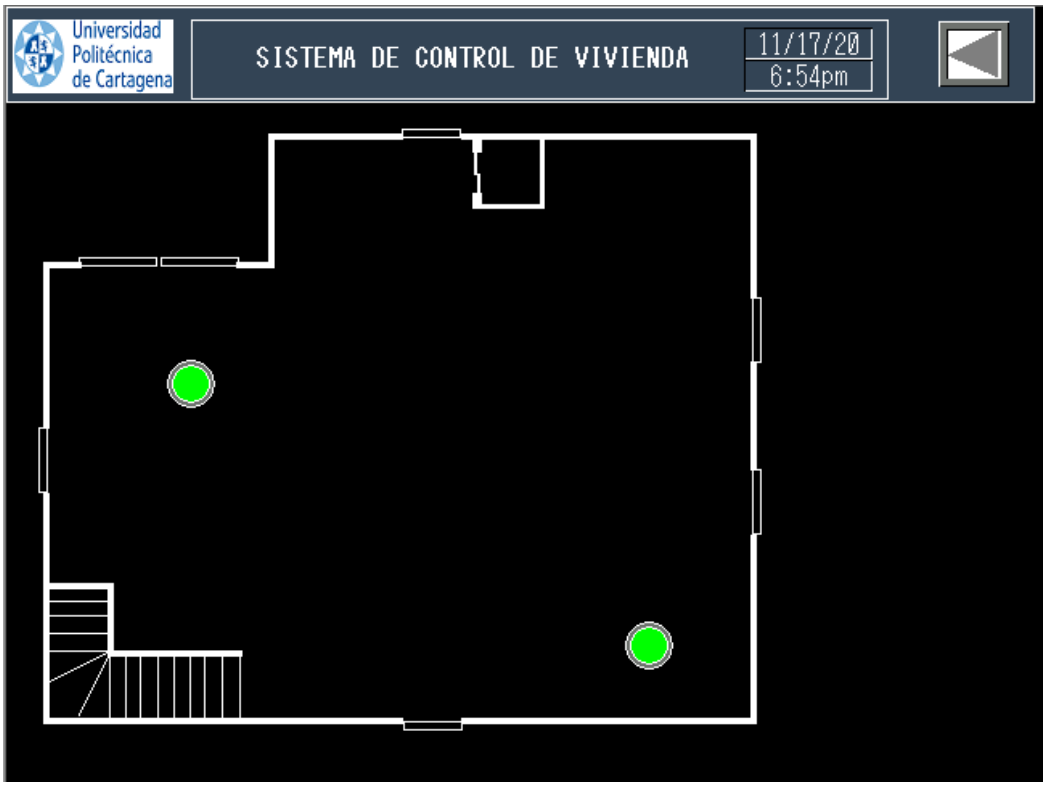


Figura 8-17 Posición sensores en planta 0

En la siguiente figura se muestra el detalle de la posición en la que están instalados los sensores de la planta 1.

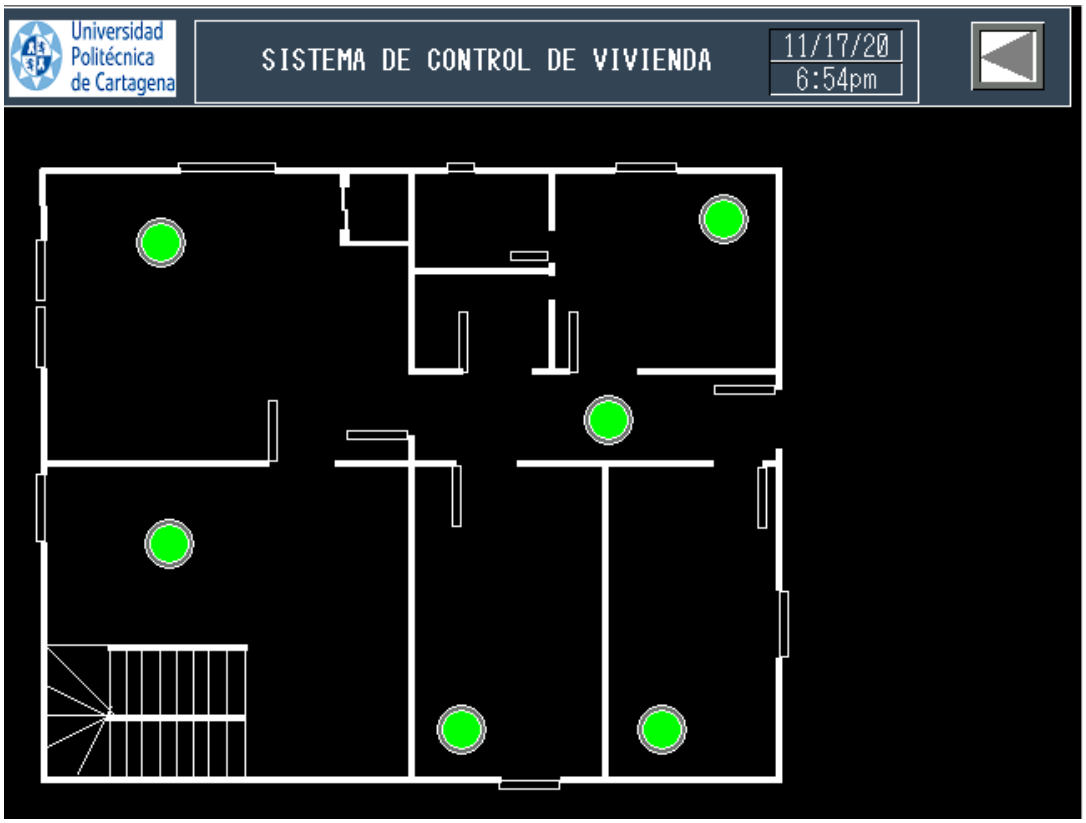


Figura 8-18 Posición sensores en planta 1

En la siguiente figura se muestra el detalle de la posición en la que están instalados los sensores de la planta 2.

Figura 8-19 Posición sensores en planta 2

8.4.2.1.3. SUBMENU PUERTAS AUTOMÁTICAS

Desde este menú podremos acceder a las dos puertas automáticas de la vivienda. La puerta automática exterior y la de acceso a la vivienda.

Figura 8-20 Submenú puertas automáticas

8.4.2.1.3.1. SUBMENU PUERTAS AUTOMÁTICAS EXTERIOR

Desde este menú podremos acceder a ver el estado de detalle de la puerta exterior de la vivienda, así como controlarla remotamente desde este submenú.

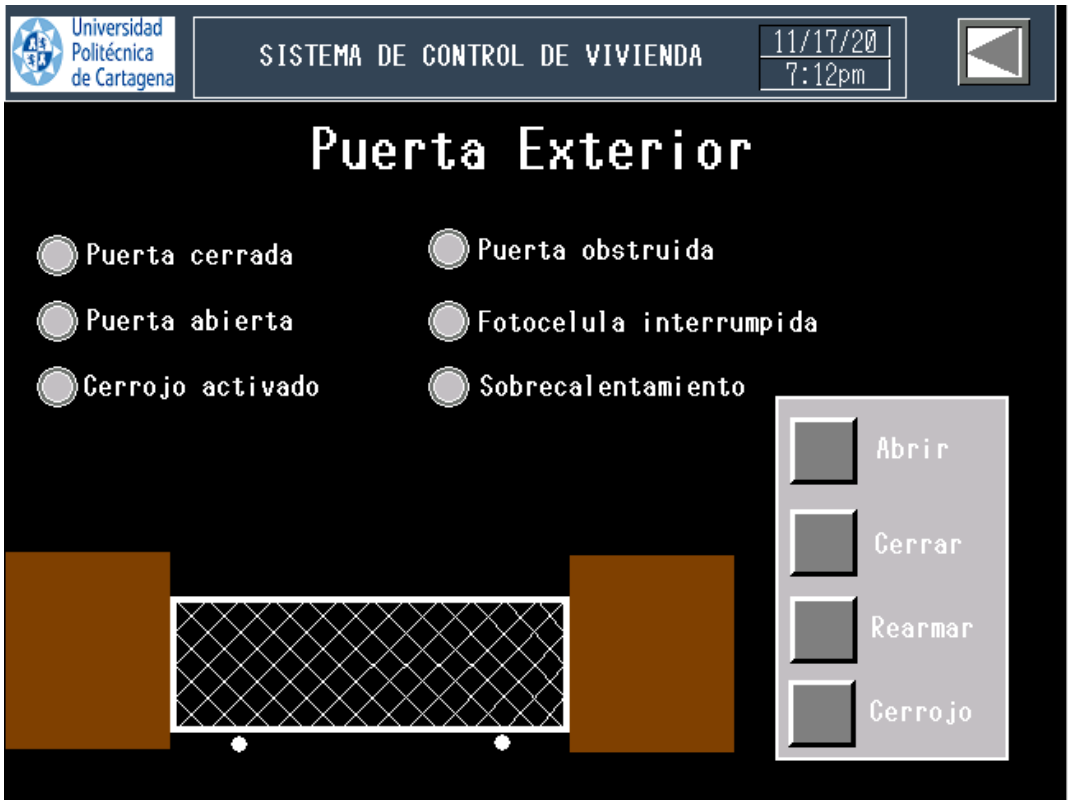


Figura 8-21 Submenú Puerta exterior vivienda

8.4.2.1.3.2. SUBMENU PUERTAS AUTOMÁTICAS INTERIOR

Desde este menú podremos acceder a ver el estado de detalle de la puerta de acceso de la vivienda, así como controlarla desde este submenú.

Figura 8-22 Submenú Puerta acceso vivienda

8.4.2.1.4. SUBMENU ASCENSOR VIVIENDA

Desde este menú podremos acceder a ver el estado del ascensor y a controlar el mismo.

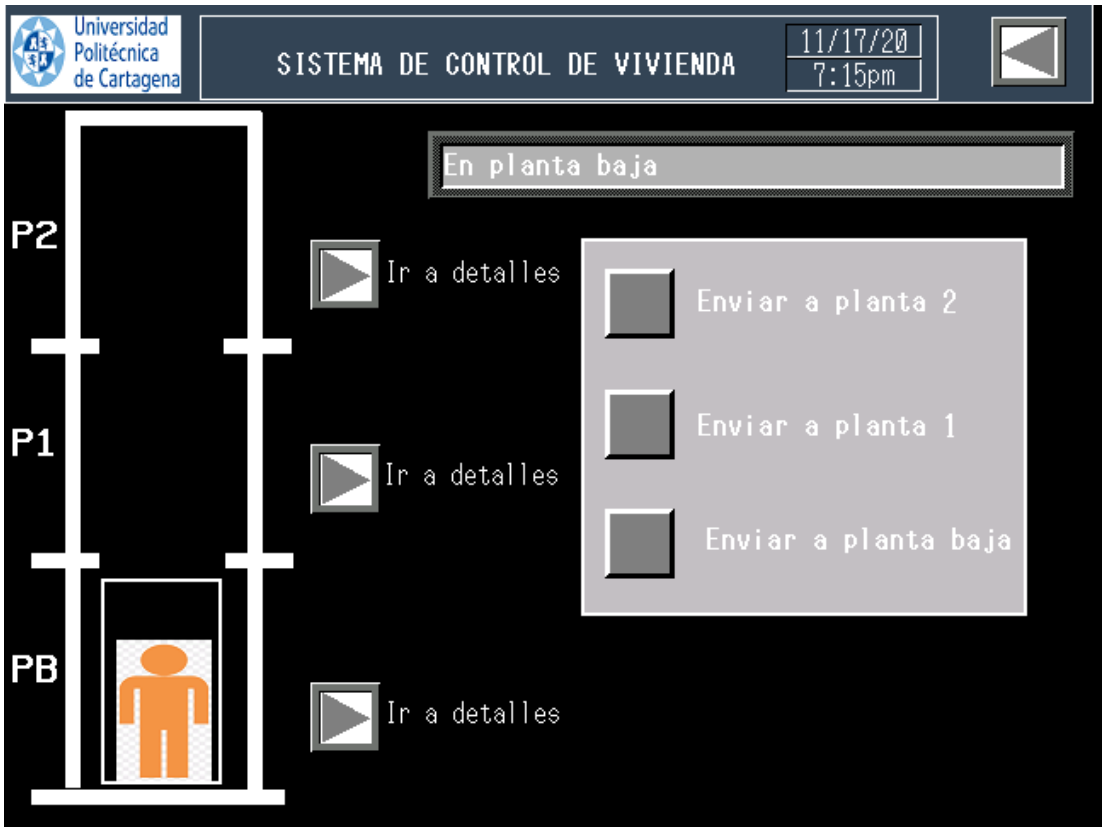


Figura 8-23 Submenú ascensor vivienda

Desde este submenú, podremos acceder a unas ventanas emergentes que nos dan detalles adicionales.

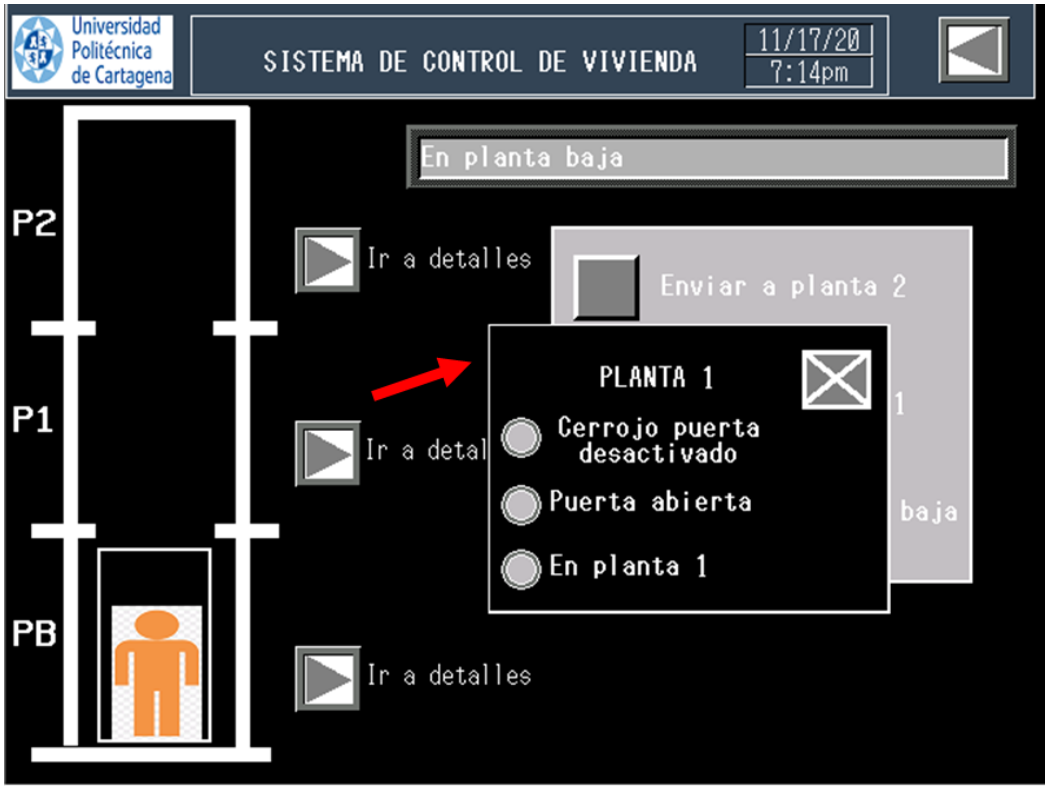


Figura 8-24 Pantalla emergente estado ascensor

8.5. VARIABLES

Las variables son posiciones de memoria que se identifican mediante nombres y que almacenan datos.

Podemos crear tantas variables como sean precisas y ellas asociarlas a los diferentes objetos (interruptores, lámparas, indicadores, visualizadores de datos, gráficos etc.).

Para comunicarse con el PLC o cualquier otro equipo conectado al terminal de destino, tenemos que crear previamente las variables y asignarles una dirección de equipo.

Las variables (con una dirección de equipo) se actualizarán conforme los datos vayan cambiando.

Cuando se conectan varios dispositivos equipos al terminal, es posible mostrar datos de las diferentes direcciones de los dispositivos en el mismo terminal (pantalla táctil). En nuestro caso, solo habrá un equipo conectado del cual se recogerán datos (el PLC que realiza el control).

En los paneles podemos usar variables internas, aquellas no asociadas a un equipo y sólo intervienen en operaciones del terminal y variables externas, que están asociadas a una dirección de equipo. Se puede combinar el uso de cualquier tipo de variable en los paneles y ventanas emergentes.

8.5.1. TIPOS DE VARIABLES.

Vijeo-Designer ofrece diversos tipos de variables que pueden ser utilizadas. De entre todas ellas, usaremos las de tipo discreto, enteras y cadena.

8.5.2. CREACIÓN DE VARIABLES

Haciendo doble clic en el icono de '**Variables**' del navegador podremos crear las variables que necesitemos. Se abrirá una ventana que representa la zona de trabajo para definirlas.

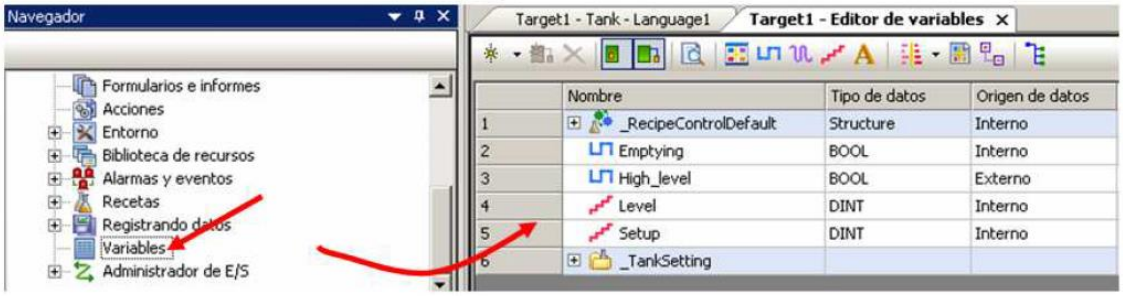


Figura 8-25 Editor de variables

Los campos que tendremos que rellenar en el editor de variables son:

- **Nombre de la variable**: Nombre que deseamos dar a la variable con un máximo de longitud 32 caracteres.
- **Descripción:** Descripción de la variable. Es posible utilizar hasta 255 caracteres para añadir una descripción.
- **Tipo de dato:** Tipo de dato que contendrá la variable (Ejemplo: Valores enteros de 16 bits = INT o booleano = Discreto).
- **Origen**: *Interno* cuando las variables no vienen de un dispositivo exterior, sino que son propias del terminal y *Equipo* para cuando las variables proceden de un dispositivo externo que está conectado al terminal.
- **Grupo de escaneo**: Grupo de escaneo del que queremos que se reciban las variables (una dirección IP). En nuestro caso al grupo de escaneo se le ha denominado PLC.
- **Dirección del dispositivo**: No utilizado en este proyecto.

La relación de variables utilizadas en este proyecto son las que se muestran en la tabla siguiente.

Name	Data Type	Data Source	Scan Group	Device Address
Alarma_activada	BOOL	External	PLC	%MW16001
Alarma_agrupada_sensores	BOOL	External	PLC	%MW16002
boton_llamada_p1	BOOL	External	PLC	%MW16006
boton_llamada_p2	BOOL	External	PLC	%MW16006
boton_llamada_pb	BOOL	External	PLC	%MW16006
cerrojo_asc_P1	BOOL	External	PLC	%MW16006
cerrojo_asc_P2	BOOL	External	PLC	%MW16006

Name	Data Type	Data Source	Scan Group	Device Address
cerrojo_asc_PB	BOOL	External	PLC	%MW16006
cocina_p1_conectar	BOOL	External	PLC	%MW15520
Codificacion_estados_alarma	INT	External	PLC	%MW16003
estado_ascensor	INT	External	PLC	%MW16005
fc0	BOOL	External	PLC	%MW16006
fc1	BOOL	External	PLC	%MW16006
fc2	BOOL	External	PLC	%MW16006
h1_p1_conectar	BOOL	External	PLC	%MW15550
h2_p1_conectar	BOOL	External	PLC	%MW15560
h3_p1_conectar	BOOL	External	PLC	%MW15570
pasillo_p1_conectar	BOOL	External	PLC	%MW15540
persiana_monitor_p0_1	INT	External	PLC	%MW16049
persiana_monitor_p0_2	INT	External	PLC	%MW16050
persiana_monitor_p0_3	INT	External	PLC	%MW16051
persiana_monitor_p0_4	INT	External	PLC	%MW16052
persiana_monitor_p0_5	INT	External	PLC	%MW16053
persiana_monitor_p1_1	INT	External	PLC	%MW16054
persiana_monitor_p1_2	INT	External	PLC	%MW16055
persiana_monitor_p1_3	INT	External	PLC	%MW16056
persiana_monitor_p1_4	INT	External	PLC	%MW16057
persiana_monitor_p1_5	INT	External	PLC	%MW16058
persiana_monitor_p2_1	INT	External	PLC	%MW16061
persiana_monitor_p2_2	INT	External	PLC	%MW16062
persiana_monitor_p2_3	INT	External	PLC	%MW16060
persiana_monitor_p2_4	INT	External	PLC	%MW16059
persiana_p1_1_abierta	BOOL	External	PLC	%MW16031
persiana_p1_1_bajar	BOOL	External	PLC	%MW16031
persiana_p1_1_cerrada	BOOL	External	PLC	%MW16031
persiana_p1_1_error	BOOL	External	PLC	%MW16031
persiana_p1_1_mov	INT	External	PLC	%MW16030
persiana_p1_1_rearme	BOOL	External	PLC	%MW16031
persiana_p1_1_subir	BOOL	External	PLC	%MW16031
persiana_p1_2_abierta	BOOL	External	PLC	%MW16033
persiana_p1_2_bajar	BOOL	External	PLC	%MW16033
persiana_p1_2_cerrada	BOOL	External	PLC	%MW16033
persiana_p1_2_error	BOOL	External	PLC	%MW16033
persiana_p1_2_mov	INT	External	PLC	%MW16032

Name	Data Type	Data Source	Scan Group	Device Address
persiana_p1_2_rearme	BOOL	External	PLC	%MW16033
persiana_p1_2_subir	BOOL	External	PLC	%MW16033
persiana_p1_3_abierta	BOOL	External	PLC	%MW16035
persiana_p1_3_bajar	BOOL	External	PLC	%MW16035
persiana_p1_3_cerrada	BOOL	External	PLC	%MW16035
persiana_p1_3_error	BOOL	External	PLC	%MW16035
persiana_p1_3_mov	INT	External	PLC	%MW16034
persiana_p1_3_rearme	BOOL	External	PLC	%MW16035
persiana_p1_3_subir	BOOL	External	PLC	%MW16035
persiana_p1_4_abierta	BOOL	External	PLC	%MW16037
persiana_p1_4_bajar	BOOL	External	PLC	%MW16037
persiana_p1_4_cerrada	BOOL	External	PLC	%MW16037
persiana_p1_4_error	BOOL	External	PLC	%MW16037
persiana_p1_4_mov	INT	External	PLC	%MW16036
persiana_p1_4_rearme	BOOL	External	PLC	%MW16037
persiana_p1_4_subir	BOOL	External	PLC	%MW16037
persiana_p1_5_abierta	BOOL	External	PLC	%MW16039
persiana_p1_5_bajar	BOOL	External	PLC	%MW16039
persiana_p1_5_cerrada	BOOL	External	PLC	%MW16039
persiana_p1_5_error	BOOL	External	PLC	%MW16039
persiana_p1_5_mov	INT	External	PLC	%MW16038
persiana_p1_5_rearme	BOOL	External	PLC	%MW16039
persiana_p1_5_subir	BOOL	External	PLC	%MW16039
persiana_p2_1_abierta	BOOL	External	PLC	%MW16041
persiana_p2_1_bajar	BOOL	External	PLC	%MW16041
persiana_p2_1_cerrada	BOOL	External	PLC	%MW16041
persiana_p2_1_error	BOOL	External	PLC	%MW16041
persiana_p2_1_mov	INT	External	PLC	%MW16040
persiana_p2_1_rearme	BOOL	External	PLC	%MW16041
persiana_p2_1_subir	BOOL	External	PLC	%MW16041
persiana_p2_2_abierta	BOOL	External	PLC	%MW16043
persiana_p2_2_bajar	BOOL	External	PLC	%MW16043
persiana_p2_2_cerrada	BOOL	External	PLC	%MW16043
persiana_p2_2_error	BOOL	External	PLC	%MW16043
persiana_p2_2_mov	INT	External	PLC	%MW16042
persiana_p2_2_rearme	BOOL	External	PLC	%MW16043
persiana_p2_2_subir	BOOL	External	PLC	%MW16043

Name	Data Type	Data Source	Scan Group	Device Address
persiana_p2_3_abierta	BOOL	External	PLC	%MW16045
persiana_p2_3_bajar	BOOL	External	PLC	%MW16045
persiana_p2_3_cerrada	BOOL	External	PLC	%MW16045
persiana_p2_3_error	BOOL	External	PLC	%MW16045
persiana_p2_3_mov	INT	External	PLC	%MW16044
persiana_p2_3_rearme	BOOL	External	PLC	%MW16045
persiana_p2_3_subir	BOOL	External	PLC	%MW16045
persiana_p2_4_abierta	BOOL	External	PLC	%MW16047
persiana_p2_4_bajar	BOOL	External	PLC	%MW16047
persiana_p2_4_cerrada	BOOL	External	PLC	%MW16047
persiana_p2_4_error	BOOL	External	PLC	%MW16047
persiana_p2_4_mov	INT	External	PLC	%MW16046
persiana_p2_4_rearme	BOOL	External	PLC	%MW16047
persiana_p2_4_subir	BOOL	External	PLC	%MW16047
persiana_pb_1_abierta	BOOL	External	PLC	%MW16021
persiana_pb_1_bajar	BOOL	External	PLC	%MW16021
persiana_pb_1_cerrada	BOOL	External	PLC	%MW16021
persiana_pb_1_error	BOOL	External	PLC	%MW16021
persiana_pb_1_mov	INT	External	PLC	%MW16020
persiana_pb_1_rearme	BOOL	External	PLC	%MW16021
persiana_pb_1_subir	BOOL	External	PLC	%MW16021
persiana_pb_2_abierta	BOOL	External	PLC	%MW16023
persiana_pb_2_bajar	BOOL	External	PLC	%MW16023
persiana_pb_2_cerrada	BOOL	External	PLC	%MW16023
persiana_pb_2_error	BOOL	External	PLC	%MW16023
persiana_pb_2_mov	INT	External	PLC	%MW16022
persiana_pb_2_rearme	BOOL	External	PLC	%MW16023
persiana_pb_2_subir	BOOL	External	PLC	%MW16023
persiana_pb_3_abierta	BOOL	External	PLC	%MW16025
persiana_pb_3_bajar	BOOL	External	PLC	%MW16025
persiana_pb_3_cerrada	BOOL	External	PLC	%MW16025
persiana_pb_3_error	BOOL	External	PLC	%MW16025
persiana_pb_3_mov	INT	External	PLC	%MW16024
persiana_pb_3_rearme	BOOL	External	PLC	%MW16025
persiana_pb_3_subir	BOOL	External	PLC	%MW16025
persiana_pb_4_abierta	BOOL	External	PLC	%MW16027
persiana_pb_4_bajar	BOOL	External	PLC	%MW16027

Name	Data Type	Data Source	Scan Group	Device Address
persiana_pb_4_cerrada	BOOL	External	PLC	%MW16027
persiana_pb_4_error	BOOL	External	PLC	%MW16027
persiana_pb_4_mov	INT	External	PLC	%MW16026
persiana_pb_4_rearme	BOOL	External	PLC	%MW16027
persiana_pb_4_subir	BOOL	External	PLC	%MW16027
persiana_pb_5_abierta	BOOL	External	PLC	%MW16029
persiana_pb_5_bajar	BOOL	External	PLC	%MW16029
persiana_pb_5_cerrada	BOOL	External	PLC	%MW16029
persiana_pb_5_error	BOOL	External	PLC	%MW16029
persiana_pb_5_mov	INT	External	PLC	%MW16028
persiana_pb_5_rearme	BOOL	External	PLC	%MW16029
persiana_pb_5_subir	BOOL	External	PLC	%MW16029
persianas_obs_p1	BOOL	External	PLC	%MW16048
persianas_obs_p2	BOOL	External	PLC	%MW16048
persianas_obs_pb	BOOL	External	PLC	%MW16048
posicion_ascensor	INT	External	PLC	%MW16004
Posicion_persiana	INT	External	PLC	%MW0
puerta_abierta_asc_P1	BOOL	External	PLC	%MW16006
puerta_abierta_asc_P2	BOOL	External	PLC	%MW16006
puerta_abierta_asc_PB	BOOL	External	PLC	%MW16006
Puerta_ext_abierta	BOOL	External	PLC	%MW16009
Puerta_ext_abriendo	BOOL	External	PLC	%MW16009
Puerta_ext_boton_abrir	BOOL	External	PLC	%MW16009
Puerta_ext_boton_cerrar	BOOL	External	PLC	%MW16009
Puerta_ext_boton_cerrojo	BOOL	External	PLC	%MW16009
Puerta_ext_cerrada	BOOL	External	PLC	%MW16009
Puerta_ext_cerrando	BOOL	External	PLC	%MW16009
Puerta_ext_error	BOOL	External	PLC	%MW16009
Puerta_ext_fotocel_interrumpida	BOOL	External	PLC	%MW16009
Puerta_ext_mov_puerta	BOOL	External	PLC	%MW16009
puerta_ext_posicion	INT	External	PLC	%MW16007
Puerta_ext_rearme	BOOL	External	PLC	%MW16009
Puerta_ext_sobrecalentamiento	BOOL	External	PLC	%MW16009
Puerta_garaje_abierta	BOOL	External	PLC	%MW16010
Puerta_garaje_abriendo	BOOL	External	PLC	%MW16010
Puerta_garaje_boton_abrir	BOOL	External	PLC	%MW16010
Puerta_garaje_boton_cerrar	BOOL	External	PLC	%MW16010

Name	Data Type	Data Source	Scan Group	Device Address
Puerta_garaje_boton_cerrojo	BOOL	External	PLC	%MW16010
Puerta_garaje_cerrando	BOOL	External	PLC	%MW16010
Puerta_garaje_cerrrada	BOOL	External	PLC	%MW16010
Puerta_garaje_error	BOOL	External	PLC	%MW16010
Puerta_garaje_fotocel_interrupt	BOOL	External	PLC	%MW16010
Puerta_garaje_mov_puerta	BOOL	External	PLC	%MW16010
puerta_garaje_posicion	INT	External	PLC	%MW16008
Puerta_garaje_rearme	BOOL	External	PLC	%MW16010
Puerta_garaje_sobrecalentamiento	BOOL	External	PLC	%MW16010
Rearme	BOOL	External	PLC	%MW16000
s1_p0_conectar	BOOL	External	PLC	%MW15500
s1_p2_conectar	BOOL	External	PLC	%MW15580
s2_p0_conectar	BOOL	External	PLC	%MW15510
salon_p1_conectar	BOOL	External	PLC	%MW15530
seccion	STRING	Internal		
Sensores_agrupados_P1	BOOL	External	PLC	%MW16002
Sensores_agrupados_P2	BOOL	External	PLC	%MW16002
Sensores_agrupados_PB	BOOL	External	PLC	%MW16002
SM_p0_1	INT	External	PLC	%MW15000
SM_p0_2	INT	External	PLC	%MW15010
SM_p1_cocina	INT	External	PLC	%MW15020
SM_p1_h1	INT	External	PLC	%MW15050
SM_p1_h2	INT	External	PLC	%MW15060
SM_p1_h3	INT	External	PLC	%MW15070
SM_p1_pasillo	INT	External	PLC	%MW15040
SM_p1_salon	INT	External	PLC	%MW15030
SM_p2_1	INT	External	PLC	%MW15080
Texto_alarma_variable	STRING	Internal		
Texto_estado_ascensor	STRING	Internal		
Texto_persiana_p0_1	STRING	Internal		
Texto_persiana_p0_2	STRING	Internal		
Texto_persiana_p0_3	STRING	Internal		
Texto_persiana_p0_4	STRING	Internal		
Texto_persiana_p0_5	STRING	Internal		
Texto_persiana_p1_1	STRING	Internal		
Texto_persiana_p1_2	STRING	Internal		
Texto_persiana_p1_3	STRING	Internal		

Name	Data Type	Data Source	Scan Group	Device Address
Texto_persiana_p1_4	STRING	Internal		
Texto_persiana_p1_5	STRING	Internal		
Texto_persiana_p2_1	STRING	Internal		
Texto_persiana_p2_2	STRING	Internal		
Texto_persiana_p2_3	STRING	Internal		
Texto_persiana_p2_4	STRING	Internal		
version	STRING	Internal		

Tabla 8-1 Variables creadas en Vijeo-Designer

8.5.3. NAVEGACION Y ANIMACIÓN DE OBJETOS EN LOS PANELES

Con Vijeo Designer tenemos la posibilidad de navegar entre los paneles y animar objetos dentro de los paneles. Desde el punto de vista de la animación de objetos, no existe diferencia entre la animación de los objetos en paneles principales y la animación en las ventanas emergentes. Es decir, se realizan de igual forma.

En este proyecto se han definido 16 paneles principales o paneles base y 26 paneles emergentes. Cada uno de ellos debe tener asociado un número el cual nos permitirá navegar entre ellos como consecuencia de una acción del operador o por algún evento que suceda.

- Paneles base:
    - 1: Menu_Inicio
    - 2: Menu_alarmas
    - 3: Menu_Puertas
    - 4: Menu_Persianas
    - 5: Menu_PB
    - 6: Persianas_PB
    - 7: Persianas_P1
    - 8: Persianas_P2
    - 9: monitorización_Sensores_PB
    - 10: monitorización_Sensores_P1
    - 11: Sensores_PB

    - 12: Sensores_P1
    - 13: Sensores_P2
    - 14: monitorización_Sensores_P2
    - 15: Config_sensores
    - 17: Puert_ext
    - 18: Puerta_int

- Ventanas emergentes
    - 1: Emg_s1_P0
    - 2: Emg_s1_P1
    - 3: Emg_cocina_p1
    - 4: Emg_salon_p1
    - 5: Emg_h1_p1
    - 6: Emg_h1_p1
    - 7: Emg_h2_p1
    - 8: Emg_h3_p1
    - 9: Emg_s1_p2
    - 10: Emg_asc_p2
    - 10011: persiana_pb_1
    - 10014: persiana_pb_2
    - 10015: persiana_pb_3
    - 10016: persiana_pb_4
    - 10017: persiana_pb_5
    - 10018: persiana_p1_1
    - 10019: persiana_p1_2
    - 10020: persiana_p1_3
    - 10021: persiana_p1_4
    - 10022: persiana_p1_5
    - 10023: persiana_p2_1
    - 10024: persiana_p2_2
    - 10025: persiana_p2_3
    - 10026: persiana_p2_4
    - 12: Emg_asc_p1
    - 13: Emg_asc_pb

A modo de ejemplo, en los siguientes apartados describiremos como se realiza la navegación entre paneles, presentación de ventanas emergentes y la animación de los diferentes objetos que se han usado en el desarrollo de este trabajo.

8.5.3.1. NAVEGACIÓN ENTRE PANELES

Para ir de un panel a otro, desde el panel de herramientas, insertaremos en el lugar que se desee un botón como se indica en la figura. El tipo de operación que realizará el botón se configurará como **panel** y se indicará el número de panel (id del panel) al cual se accederá una vez pulsado.

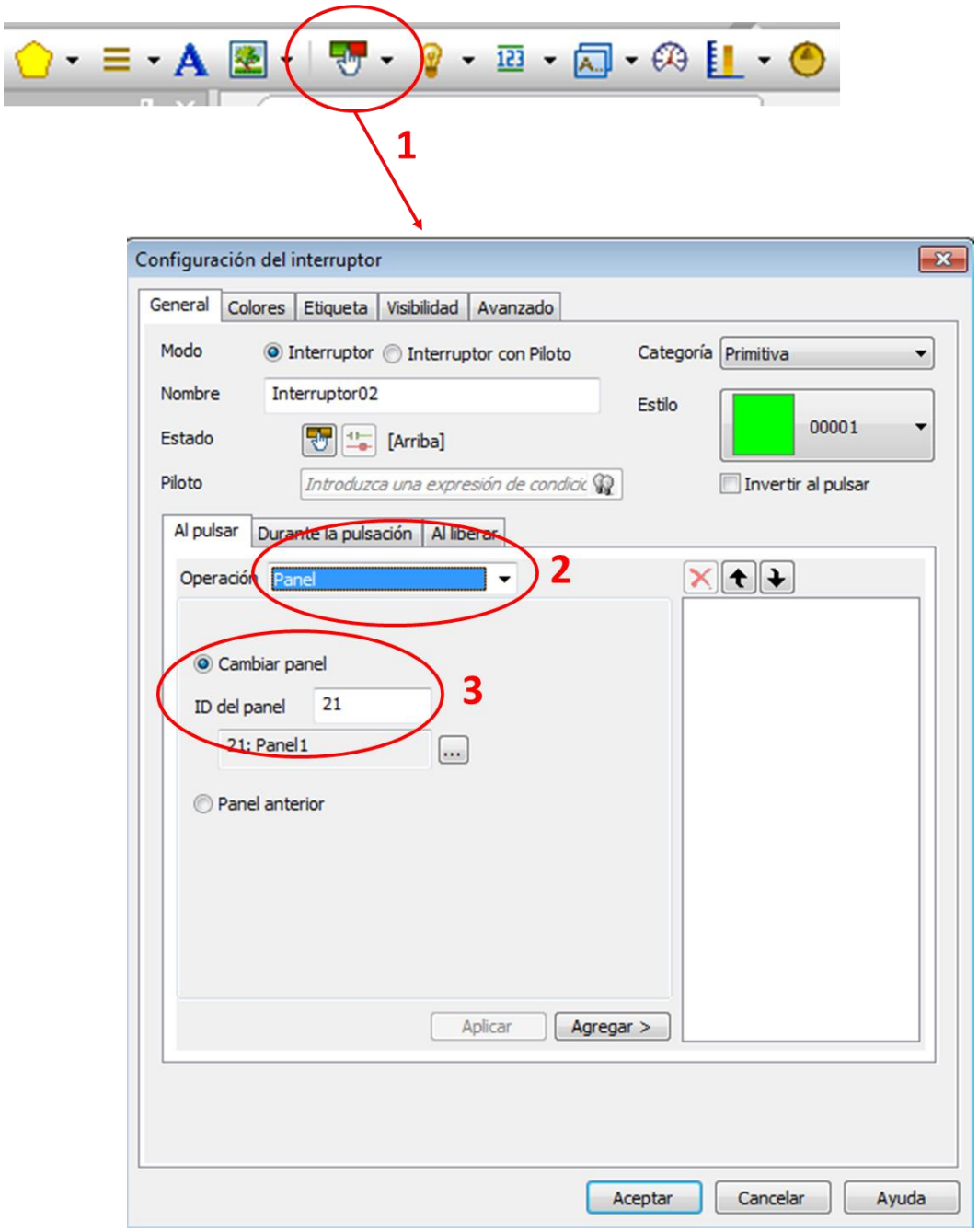


Figura 8-26 Configuración de navegación entre paneles

8.5.3.2. PRESENTACIÓN DE VENTANAS EMERGENTES

Para presentar una ventana emergente el proceso a seguir es muy similar al de ir a un panel. Se seleccionará como operación Emergente y el nombre del panel emergente al que deseamos ir.

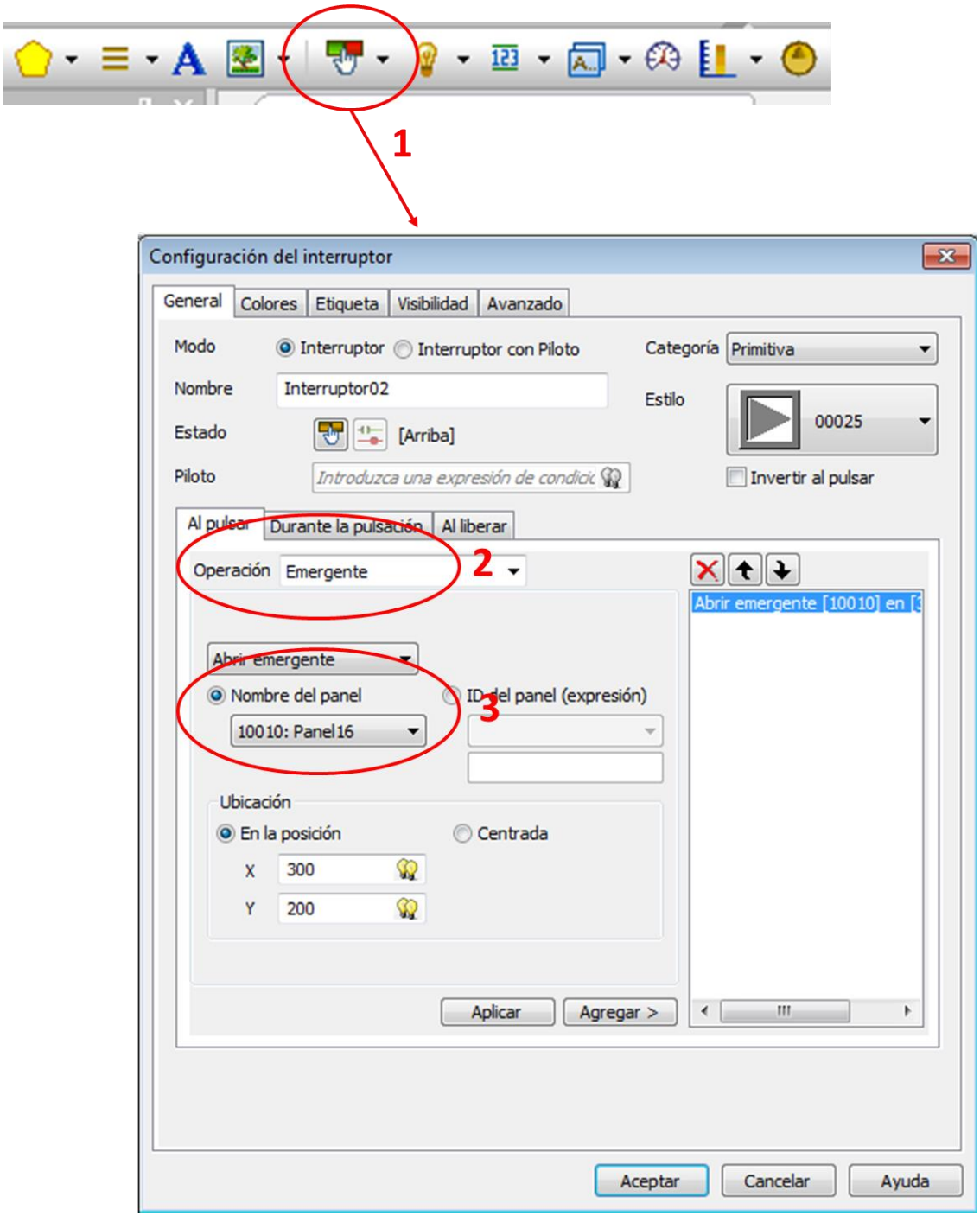


Figura 8-27 Configuración de ventanas emergentes

8.5.3.3. ANIMACIÓN DE UN PILOTO

Para añadir un piloto en un panel es necesario configurar este una vez que desde el panel de herramientas se ha añadido. En nuestro caso el piloto tendrá 4 estados que dependerán del valor que tome una variable de tipo entera. Cuando la variable tenga el valor 0 mostraremos el piloto en color verde

sin parpadeo, cuando la variable tenga el valor 1 mostraremos el piloto en color verde con parpadeo lento, cuando la variable tenga el valor 2 mostraremos el piloto en color rojo sin parpadeo y cuando la variable contenga el valor 3 mostraremos el piloto en valor amarillo sin parpadeo.

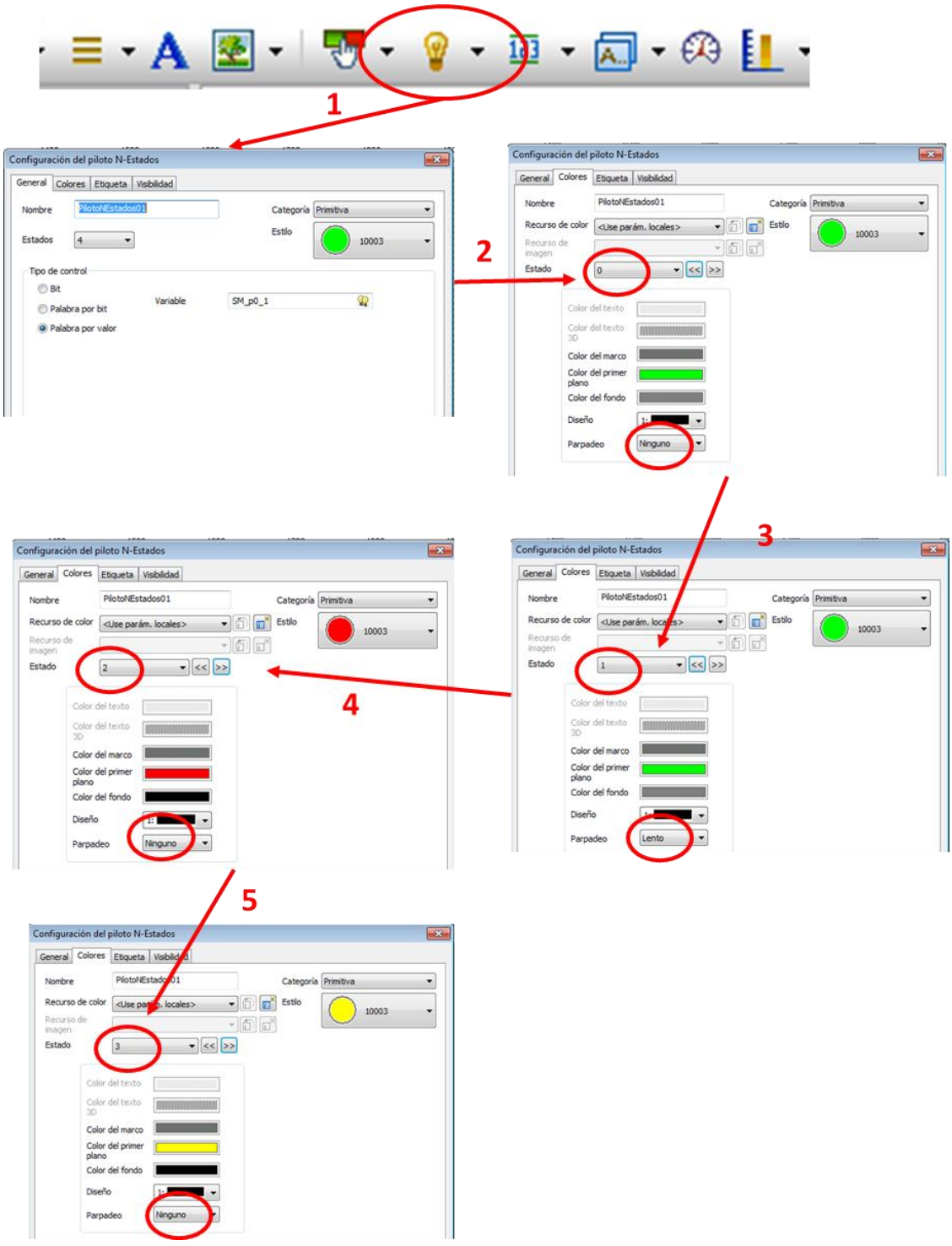


Figura 8-28 Animación de un piloto

8.5.3.4. CREACIÓN DE UN PULSADOR

Los pulsadores los utilizaremos para realizar acciones sobre ciertas variables que serán enviadas al PLC. Por ejemplo, es posible realizar una acción sobre un pulsador para que en el momento de ser accionado desde la pantalla ponga una variable en estado 1 y sea enviada al PLC y cuando se libere el pulsador la variable se ponga en el estado 0. Este proceso descrito se muestra a continuación

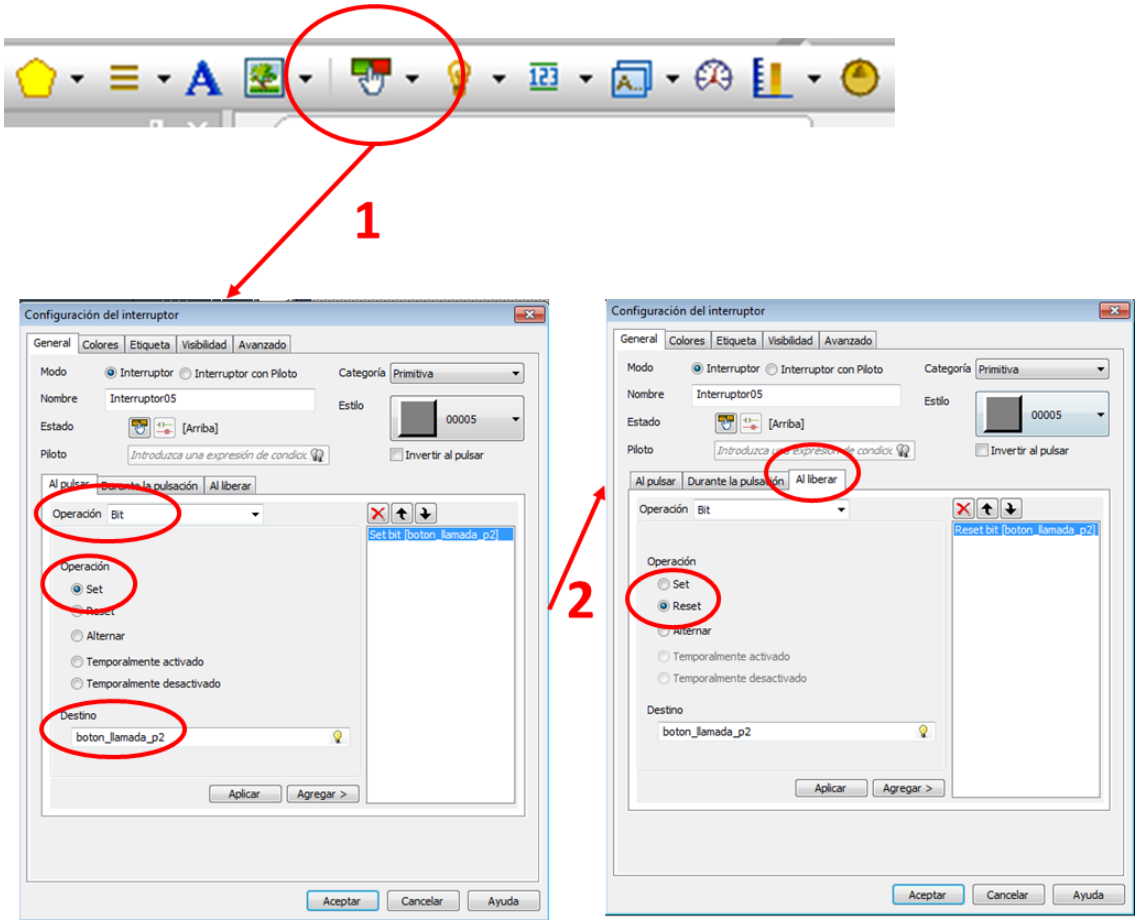


Figura 8-29 Creación de un panel

8.5.3.5. OBJETOS EN MOVIMIENTO

Es posible realizar desplazamientos de forma dinámica de objetos en los paneles. Como ejemplo explicaremos como realizar el desplazamiento vertical

de la cabina del ascensor. El procedimiento será el mismo para las puertas automáticas de la vivienda.

Para configurar el movimiento de un objeto, clicaremos con el botón derecho del ratón sobre el objeto que deseamos animar y seleccionaremos la opción **animar…** y estableceremos los parámetros que se indican en la figura que se muestra a continuación.

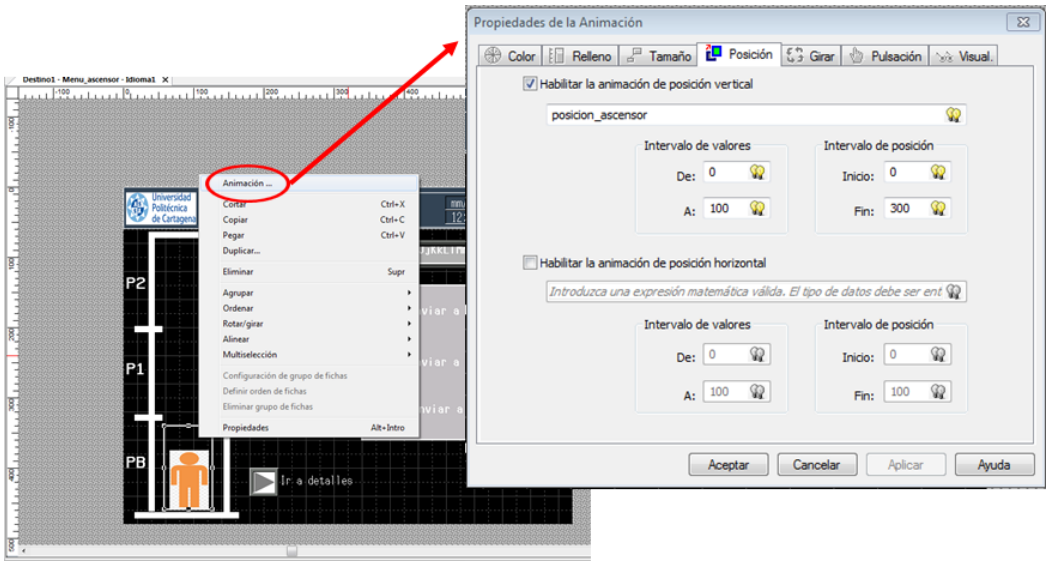


Figura 8-30 Desplazamiento dinámico objetos

En la figura anterior tenemos la variable posición_ascensor de tipo integer que el PLC establece su valor en función de la posición en la que se encuentra el ascensor. El intervalo de valores indica en qué rango se moverá la variable (0-100). En el intervalo de posición se indica en pixeles el desplazamiento del objeto. Así, tendremos que cuando la variable posicion_ascensor valga 0 el objeto se moverá 0 pixeles desde su posición inicial y cuando la variable posición_ascensor valga 100 el objeto se habrá movido 300 pixeles desde su posición inicial.

8.5.3.6. CREACIÓN DE TEXTOS DINÁMICOS

A veces resulta útil presentar en pantalla textos que no sean estáticos, sino que sean dinámicos. Una forma de implementar los es mediante el uso de **variables de tipo string** y cambiar el contenido de estas variables mediante un

módulo de programa especial. Vijeo-Designer ofrece la posibilidad del uso script los cuales utilizaremos para cambiar el contenido de variables.

Para presentar en un panel un texto dinámico debemos seguir los siguientes pasos: en la barra de herramientas seleccionamos visualizador de mensaje y en el menú de configuración que nos aparece debemos seleccionar las opciones que se observan en la siguiente figura

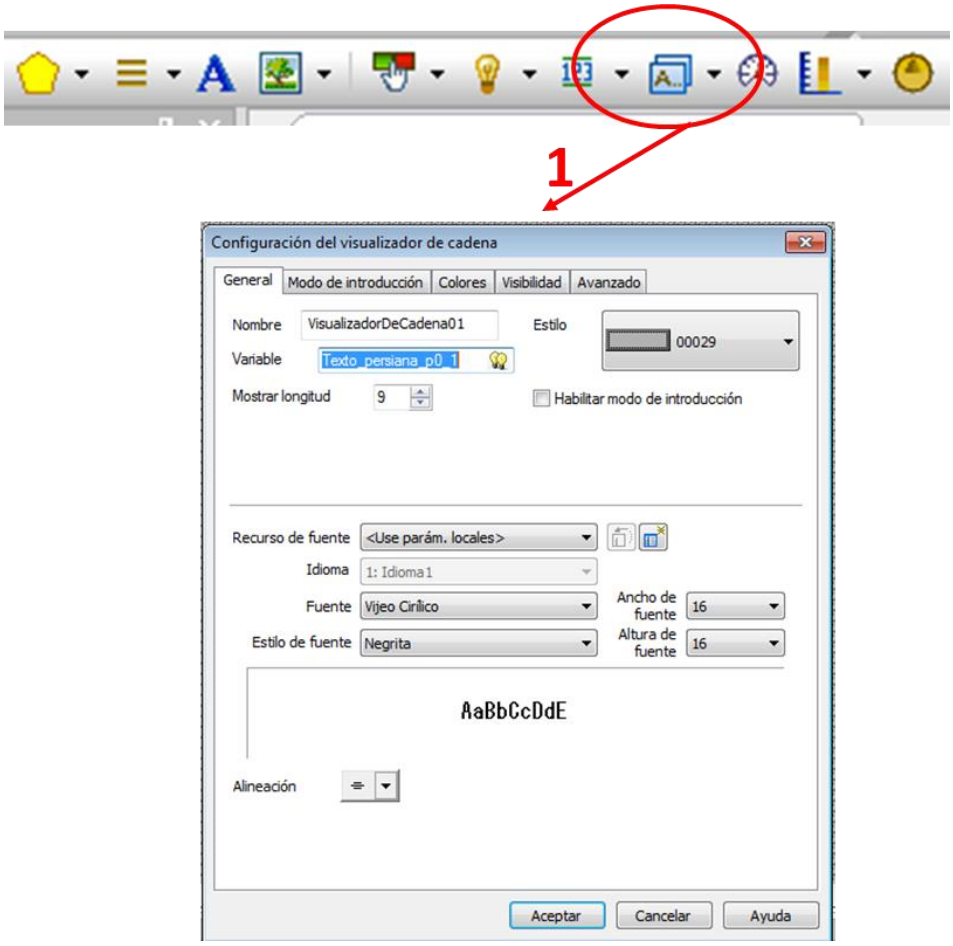


Figura 8-31 Creación de un texto dinámico

8.5.3.6.1. SCRIPT EN VIJEO DESIGNER

Los Script contienen código escrito en Java para describir las acciones del terminal ante los eventos que se producen en tiempo de ejecución como:

- Por una pulsación del usuario u operador.
- Por cambio de panel.

- Por cambio en el valor en una variable o condición.
- Cada cierto tiempo.

Los scripts de Vijeo-Designer están basados en el lenguaje Java. Pueden usarse ciertos algunases clases y métodos Java, y se complementan con algunos de los métodos propios de Vijeo-Designer. Existen ciertas diferencias entre los scripts de Java y los de Vijeo-Designer, pero en la mayoría de los casos se usan los objetos e idénticas operaciones como si estuviesen programados en Java.

8.5.3.6.2. PROGRAMACIÓN DE UN SCRIPT

Para configurar un script seleccionaremos con el ratón en el navegador en la carpeta de acciones y nos aparecerá un menú donde debemos seleccionar el **tipo de disparador** (periódico, programado, evento, condicional etc)

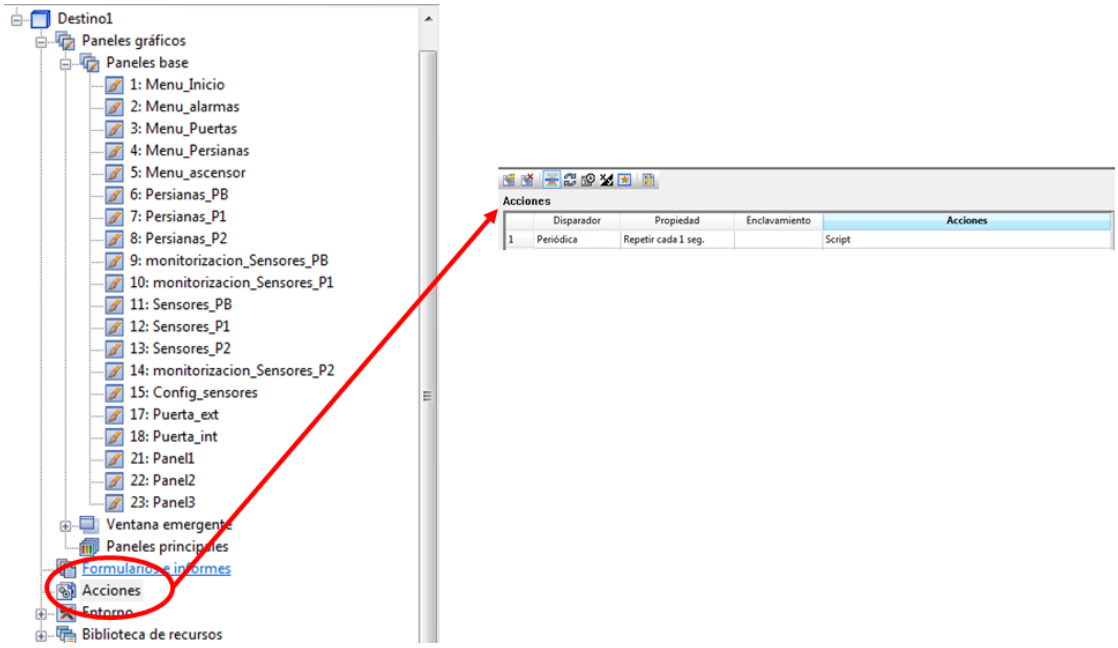


Figura 8-32 Creación de script

8.5.3.6.3. CODIGO DE LOS SCRIPT DEL PROYECTO

En el proyecto se han utilizado los scripts para, básicamente, asignar texto dinámico a los estados de las persianas y a ciertas variables que aparecen en los paneles del ascensor.

```
//////////////
//ASCENSOR//
//////////////
//Con la variable estado_ascensor a traves de este script
// alimentamos Texto_estado_ascensor que es una variable de tipo
STRING
//para la monitorizacion del estado del ascensor por pantalla.
//A continuacion la asignacion realiada:
//    EnPlanta0 := 0 ;
//    EnPlanta1:= 1 ;
//    EnPlanta2:= 2 ;
//    SubidaPlanta1Desde0:= 3 ;
//    SubidaPlanta2Desde0:= 4 ;
//    SubidaPlanta2Desde1:= 5 ;
//    BajadaPlanta0Desde1:= 6 ;
//    BajadaPlanta0Desde2:= 7 ;
//    BajadaPlanta1Desde2:= 8 ;
//    Desconocido := 9 ;

if(estado_ascensor.getIntValue()==0)
{
      Texto_estado_ascensor.write("En planta baja");
}
if(estado_ascensor.getIntValue()==1)
{
      Texto_estado_ascensor.write("En planta 1");
}
if(estado_ascensor.getIntValue()==2)
{
      Texto_estado_ascensor.write("En planta 2");
}
if(estado_ascensor.getIntValue()==3)
{
      Texto_estado_ascensor.write("Subiendo a planta 1 desde planta
baja...");
}
if(estado_ascensor.getIntValue()==4)
{
      Texto_estado_ascensor.write("Subiendo a planta 2 desde planta
baja...");
}
if(estado_ascensor.getIntValue()==5)
{
      Texto_estado_ascensor.write("Subiendo a planta 2 desde planta
1...");
}
if(estado_ascensor.getIntValue()==6)
{
      Texto_estado_ascensor.write("Bajando a planta baja desde planta
1...");
}
if(estado_ascensor.getIntValue()==7)
{
      Texto_estado_ascensor.write("Bajando a planta baja desde planta
2...");
```

```
}
if(estado_ascensor.getIntValue()==8)
{
      Texto_estado_ascensor.write("Bajando a planta 1 desde planta
2...");
}
if(estado_ascensor.getIntValue()==9)
{
      Texto_estado_ascensor.write("***Estado Desconocido***");
}

////////////////////////////////
//monitorizacion de Persianas//
////////////////////////////////
// Para cada persiana utilizamos la monitorizacion de su  estado
//en una variable STRING que mostraremos por pantalla.

//PERSIANA P0 1
if(persiana_monitor_p0_1.getIntValue()==0)
{
      Texto_persiana_p0_1.write("En medio");
}
if(persiana_monitor_p0_1.getIntValue()==1)
{
      Texto_persiana_p0_1.write("Cerrada");
}
if(persiana_monitor_p0_1.getIntValue()==2)
{
      Texto_persiana_p0_1.write("Abierta");
}

//PERSIANA P0 2
if(persiana_monitor_p0_2.getIntValue()==0)
{
      Texto_persiana_p0_2.write("En medio");
}
if(persiana_monitor_p0_2.getIntValue()==1)
{
      Texto_persiana_p0_2.write("Cerrada");
}
if(persiana_monitor_p0_2.getIntValue()==2)
{
      Texto_persiana_p0_2.write("Abierta");
}
//PERSIANA P0 3
if(persiana_monitor_p0_3.getIntValue()==0)
{
      Texto_persiana_p0_3.write("En medio");
}
if(persiana_monitor_p0_3.getIntValue()==1)
{
      Texto_persiana_p0_3.write("Cerrada");
}
if(persiana_monitor_p0_3.getIntValue()==2)
{
      Texto_persiana_p0_3.write("Abierta");
}
//PERSIANA P0 4
if(persiana_monitor_p0_4.getIntValue()==0)
{
      Texto_persiana_p0_4.write("En medio");
```

```
}
if(persiana_monitor_p0_4.getIntValue()==1)
{
      Texto_persiana_p0_4.write("Cerrada");
}
if(persiana_monitor_p0_4.getIntValue()==2)
{
      Texto_persiana_p0_4.write("Abierta");
}
//PERSIANA P0 5
if(persiana_monitor_p0_5.getIntValue()==0)
{
      Texto_persiana_p0_5.write("En medio");
}
if(persiana_monitor_p0_5.getIntValue()==1)
{
      Texto_persiana_p0_5.write("Cerrada");
}
if(persiana_monitor_p0_5.getIntValue()==2)
{
      Texto_persiana_p0_5.write("Abierta");
}

//PERSIANA P1 1
if(persiana_monitor_p1_1.getIntValue()==0)
{
      Texto_persiana_p1_1.write("En medio");
}
if(persiana_monitor_p1_1.getIntValue()==1)
{
      Texto_persiana_p1_1.write("Cerrada");
}
if(persiana_monitor_p1_1.getIntValue()==2)
{
      Texto_persiana_p1_1.write("Abierta");
}

//PERSIANA P1 2
if(persiana_monitor_p1_2.getIntValue()==0)
{
      Texto_persiana_p1_2.write("En medio");
}
if(persiana_monitor_p1_2.getIntValue()==1)
{
      Texto_persiana_p1_2.write("Cerrada");
}
if(persiana_monitor_p1_2.getIntValue()==2)
{
      Texto_persiana_p1_2.write("Abierta");
}

//PERSIANA P1 3
if(persiana_monitor_p1_3.getIntValue()==0)
{
      Texto_persiana_p1_3.write("En medio");
}
if(persiana_monitor_p1_3.getIntValue()==1)
{
      Texto_persiana_p1_3.write("Cerrada");
}
if(persiana_monitor_p1_3.getIntValue()==2)
```

```
{
      Texto_persiana_p1_3.write("Abierta");
}

//PERSIANA P1 4
if(persiana_monitor_p1_4.getIntValue()==0)
{
      Texto_persiana_p1_4.write("En medio");
}
if(persiana_monitor_p1_4.getIntValue()==1)
{
      Texto_persiana_p1_4.write("Cerrada");
}
if(persiana_monitor_p1_4.getIntValue()==2)
{
      Texto_persiana_p1_4.write("Abierta");
}

//PERSIANA P1 5
if(persiana_monitor_p1_5.getIntValue()==0)
{
      Texto_persiana_p1_5.write("En medio");
}
if(persiana_monitor_p1_5.getIntValue()==1)
{
      Texto_persiana_p1_5.write("Cerrada");
}
if(persiana_monitor_p1_5.getIntValue()==2)
{
      Texto_persiana_p1_5.write("Abierta");
}

//PERSIANA P2 1
if(persiana_monitor_p2_1.getIntValue()==0)
{
      Texto_persiana_p2_1.write("En medio");
}
if(persiana_monitor_p2_1.getIntValue()==1)
{
      Texto_persiana_p2_1.write("Cerrada");
}
if(persiana_monitor_p2_1.getIntValue()==2)
{
      Texto_persiana_p2_1.write("Abierta");
}

//PERSIANA P2 2
if(persiana_monitor_p2_2.getIntValue()==0)
{
      Texto_persiana_p2_2.write("En medio");
}
if(persiana_monitor_p2_2.getIntValue()==1)
{
      Texto_persiana_p2_2.write("Cerrada");
}
if(persiana_monitor_p2_2.getIntValue()==2)
{
      Texto_persiana_p2_2.write("Abierta");
}

//PERSIANA P2 3
```

```
if(persiana_monitor_p2_3.getIntValue()==0)
{
      Texto_persiana_p2_3.write("En medio");
}
if(persiana_monitor_p2_3.getIntValue()==1)
{
      Texto_persiana_p2_3.write("Cerrada");
}
if(persiana_monitor_p2_3.getIntValue()==2)
{
      Texto_persiana_p2_3.write("Abierta");
}

//PERSIANA P2 4
if(persiana_monitor_p2_4.getIntValue()==0)
{
      Texto_persiana_p2_4.write("En medio");
}
if(persiana_monitor_p2_4.getIntValue()==1)
{
      Texto_persiana_p2_4.write("Cerrada");
}
if(persiana_monitor_p2_4.getIntValue()==2)
{
      Texto_persiana_p2_4.write("Abierta");
}
```

9. VERIFICACIÓN Y PRUEBAS

En este apartado se describen las pruebas realizadas al proyecto que se ha desarrollado. Las pruebas comprenden dos partes:

- **Pruebas unitarias** del software desarrollado en el PLC, la aplicación creada para el terminal táctil y la configuración de los elementos de adquisición de campo (islas Advantys).
- **Pruebas de integración** del conjunto de elementos (PLC, Terminal táctil, periferia distribuida y red).

Para la realización del conjunto de pruebas se han utilizado equipos reales que han sido prestados por la empresa en la que realicé las prácticas curriculares, NAVANTIA – Sistemas.

9.1. DETALLES DEL ENTORNO DE PRUEBAS

Los equipos que se detallan en este apartado son los que se han utilizado para las pruebas y son idénticos a los que se han ido describiendo a lo largo del proyecto.

9.1.1. ISLAS ADVANTYS

La imagen adjunta muestra las tres islas Advantys usadas para pruebas. Se han montado sobre un perfil DIN normalizado, en cada una de ellas se ha cableado la alimentación que necesita el módulo NIM32211 y el módulo de distribución de alimentación PT3100 (alimentación para entradas digitales y alimentación para salidas digitales).

Sobre el mismo carril DIN, hay montado un swtich ethernet al que se conectan las tres islas, la pantalla táctil y el PLC.

El suministro de voltaje es proporcionado por una fuente de alimentación externa de 24Vcc.

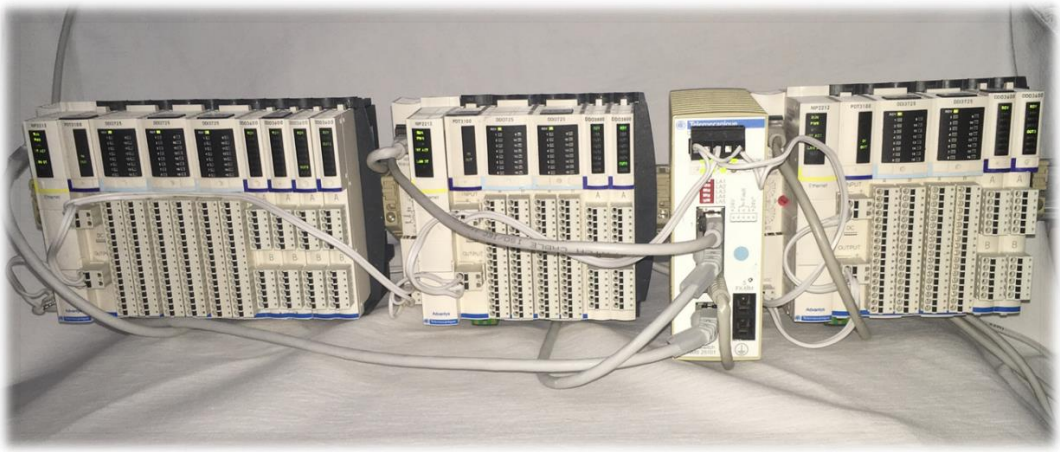


Figura 9-1 Islas Advantys utilizas para pruebas

9.1.2. PLC

La imagen adjunta muestra el procesador junto a su fuente de alimentación montado en un bastidor.

La fuente de alimentación del PLC, precisa de una tención de 24Vdc que es proporcionada por una fuente externa.

Como se indicó en apartados anteriores, el procesador lleva integrado un puerto ethernet el cual se conecta al swtich que va montado junto a las islas Advantys.

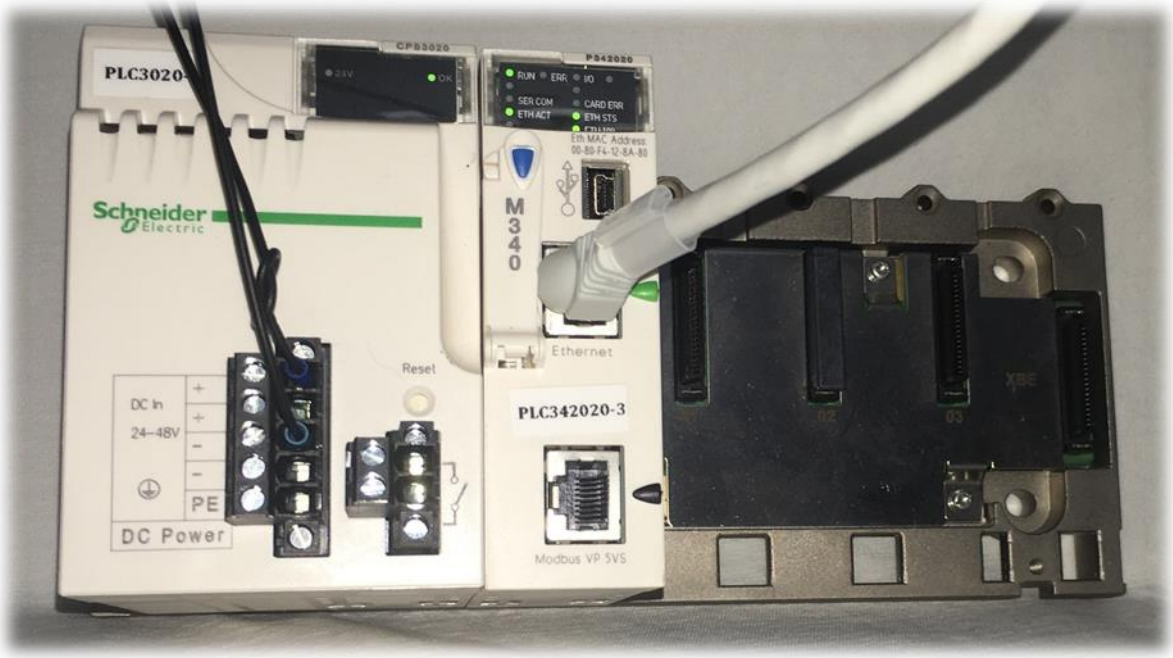


Figura 9-2 PLC utilizado para pruebas

9.1.3. TERMINAL TÁCTIL

La imagen adjunta muestra el terminal táctil usado.

Precisa de alimentación externa de 24Vcc y conectada a través de su puerto ethernet integrado al switch montado sobre el carril DIN de las islas Advantys.

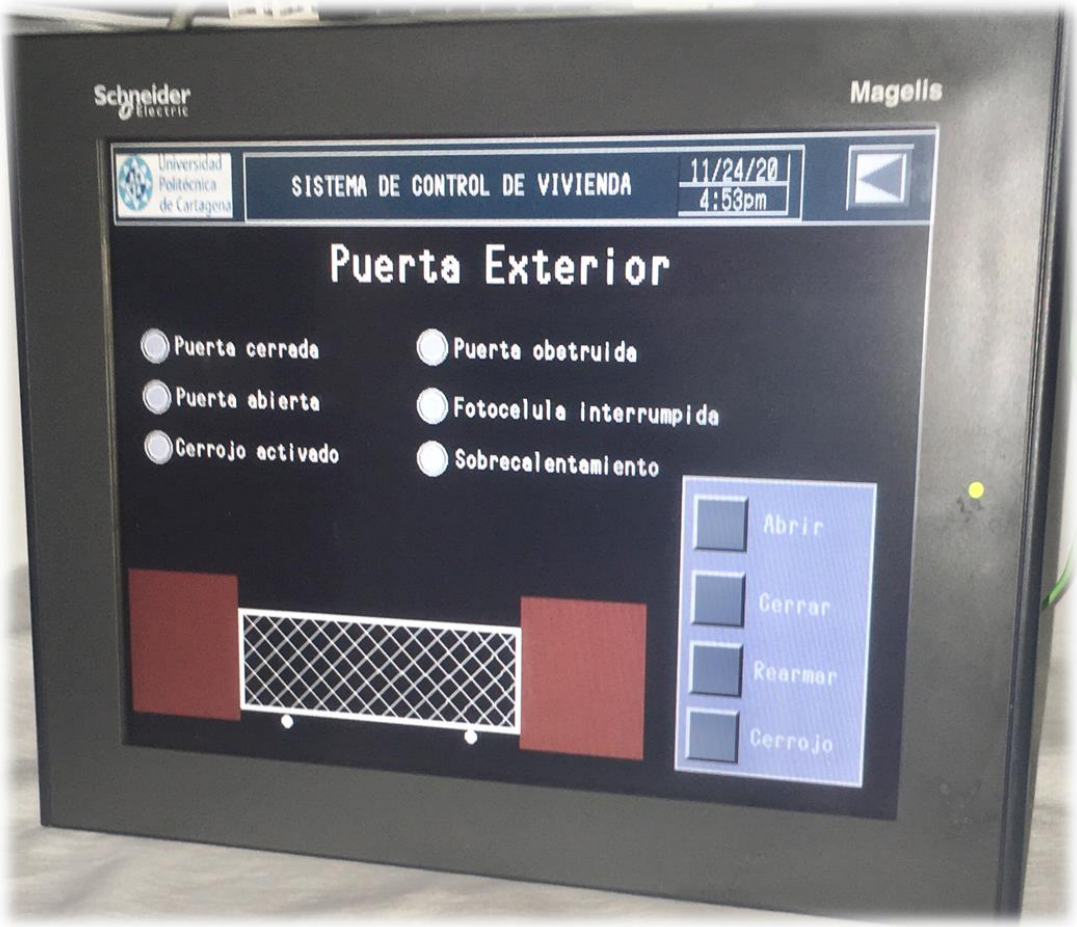


Figura 9-3 Pantalla táctil utilizada para pruebas

9.1.4. TABLAS DE ANIMACIÓN

Para comprobar si un nuevo proyecto funciona correctamente, 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.

Unity facilita la depuración de un programa mostrando en tiempo real cómo evolucionan las variables y las secciones de los programas mientras el programa del PLC está en modo de ejecución.

Para ello es necesario tener conectada la aplicación al PLC e iniciar su ejecución.

Los pasos a realizar para crear una tabla de animación es la que se muestra en la siguiente figura

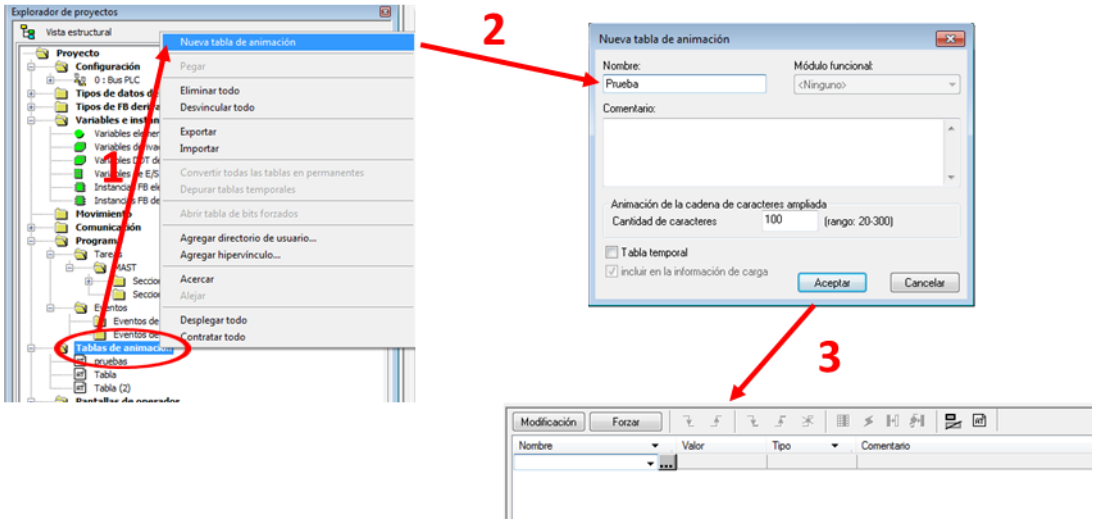


Figura 9-4 Creación de una tabla de animación

9.1.4.1. TABLAS DE ANIMACIÓN ESTRUCTURAS DATOS DE LECTURA ADVANTYS

Para comprobar la correcta recepción de datos de datos de campo usaremos tablas de animación. Las tablas de animación que se van a utilizar corresponden a las variables de recepción de entradas digitales de Advantys: Lect_Adts_Planta_0[ ], Lect_Adts_Planta_1[ ], Lect_Adts_Planta_2[ ].

Con las tablas de animación creadas como la que se muestra en la figura, iremos comprobando cada una de las entradas de campo y su correcta asignación en la estructura que le corresponden.

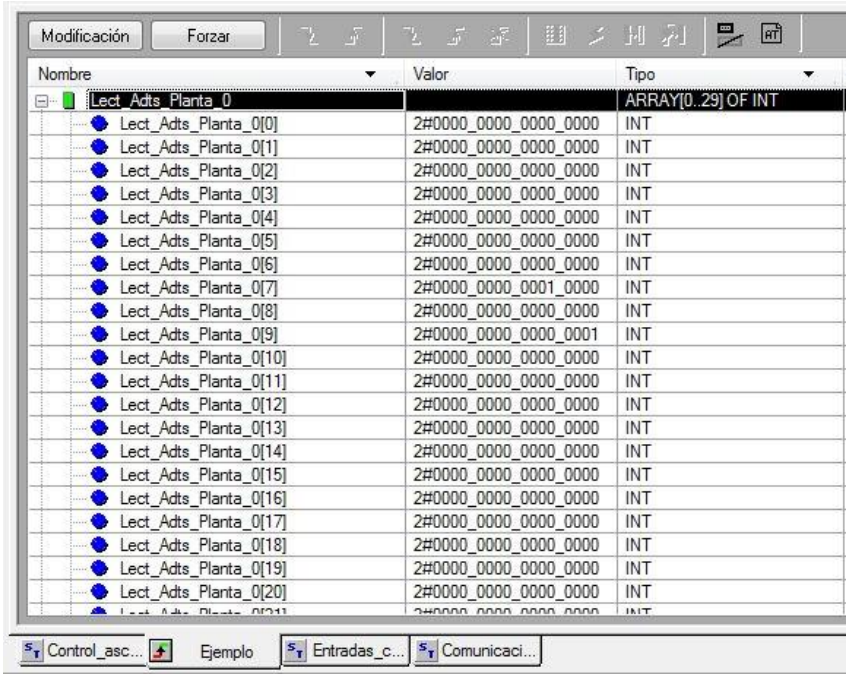


Figura 9-5 Pantalla de animación array Lect_Adts_Planta0

9.1.5. COMPROBACIÓN ASIGNACIÓN ENTRADAS DE ADVANTYS

En el apartado 6.3 se asignaron a cada una de las entradas digitales de las islas unas bornas en los módulos de entrada Advantys. Para verificar que las islas funcionan correctamente, es decir, que cada uno de los canales es leído y enviado al módulo NIM, se puede utilizar el Web Server que este lleva integrado y entrar en el apartado de diagnósticos. De esta forma, haciendo un puente en cada par de bornas que tiene asignado cada canal, se verifica si su funcionamiento es el que corresponde.

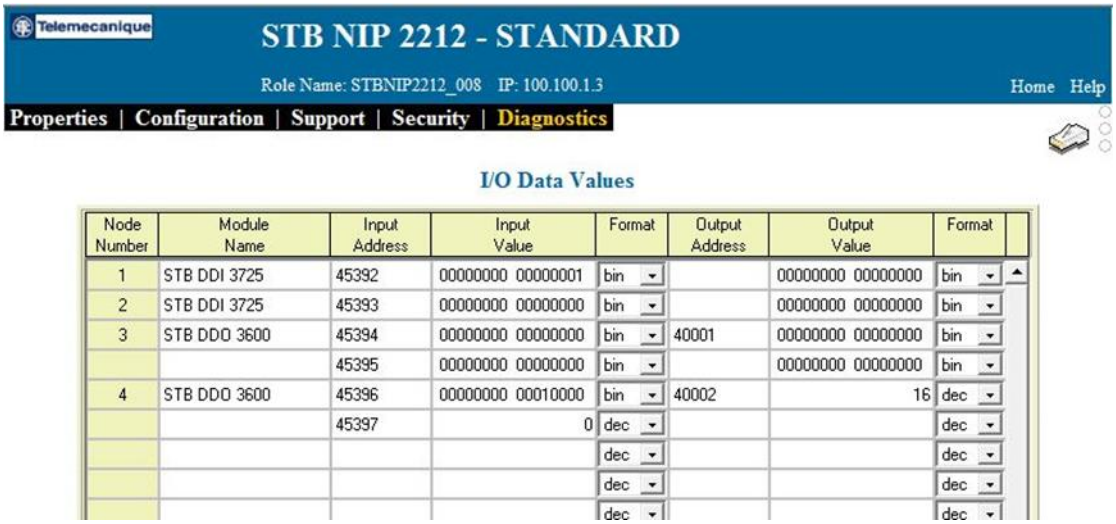


Figura 9-6 Web Server integrado en NIM2211

9.1.6. PANTALLAS DE OPERADOR

El software de desarrollo Unity ofrece la posibilidad de representar mediante mímicos o gráficos las variables de los procesos que están ocurriendo cuando está en marcha el programa del PLC. También, es posible escribir variables en los procesos que están en funcionamiento. Está última opción resulta muy útil a la hora de simular las entradas digitales que vienen de campo y así ver cuál es el comportamiento o reacción del programa del PLC.

Para comprobar el comportamiento del programa del PLC y la correcta integración con el terminal táctil, inhibiremos (anular su ejecución) la sección de programa denominada ENTRADAS_CAMPO. Como se describió en apartados anteriores, esta sección tiene como función asignar a cada una de las variables de las estructuras de datos los valores recogidos por las islas Advantys y que proceden de los sensores. Al inhibir esta sección de programa, podremos establecer los valores de entrada en las estructuras de forma manual haciendo uso de las pantallas de operador.

Como se muestra en la figura que se presenta a continuación, en el modo real o modo normal de operación las entradas digitales recogidas de campo quedan almacenadas en los arrays de recepción Lect_Adts_Planta_0[ ], Lect_Adts_Planta_1[ ] y Lect_Adts_Planta_2[ ]. Cuando se ejecuta la sección de programa Entradas_campo, los valores se asignan a las estructuras de datos de cada uno de los elementos (persianas, sensores, ascensor, etc.). Las

posteriores secciones de código toman como entradas los valores de estas estructuras de datos para su ejecución.

Si inhibimos la ejecución de la sección de código **Entradas_campo**, las estructuras de datos de cada uno de los elementos quedan sin actualizar con los valores de campo y así, es posible desde las pantallas de operador establecer los valores que deseamos.

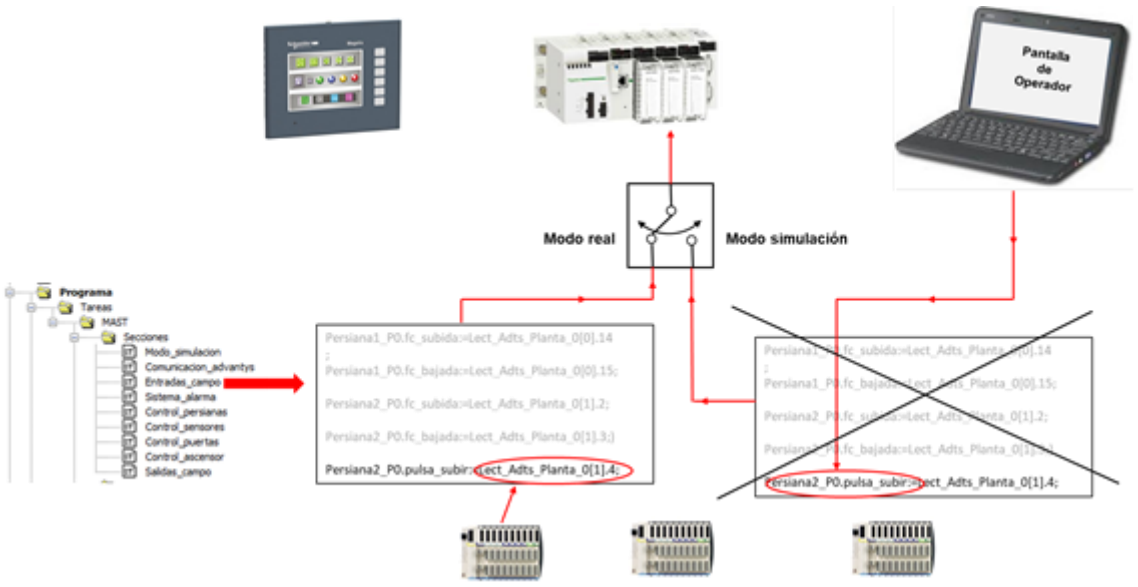


Figura 9-7 Simulación de entradas de campo

9.1.6.1. PANTALLA DE OPERADOR ASCENSOR

Desde esta pantalla de operador podremos simular las señales de entrada que recibimos del ascensor. Los fines de carrera de cada una de las plantas y el estado de puerta abierta/cerrada en cada planta.

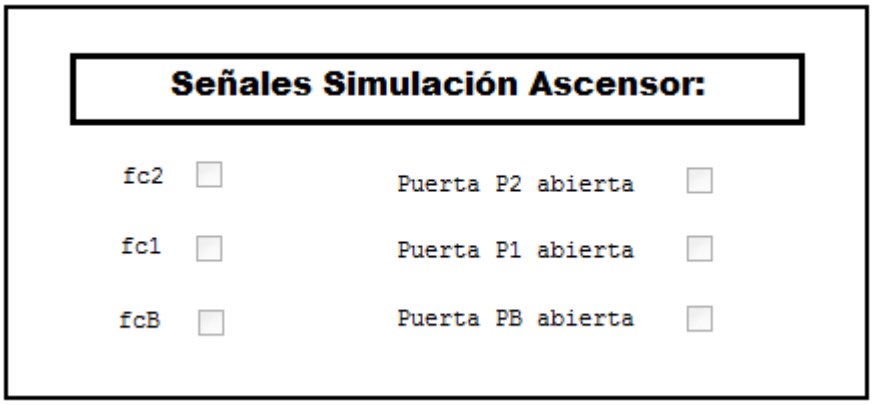


Figura 9-8 Pantalla de operador Ascensor

9.1.6.2. PANTALLA DE OPERADOR PERSIANAS

Desde esta pantalla de operador podremos simular las señales de entrada que recibimos de cada una de las persianas. Los fines de carrera de persiana abierta, cerrada o posición intermedia.

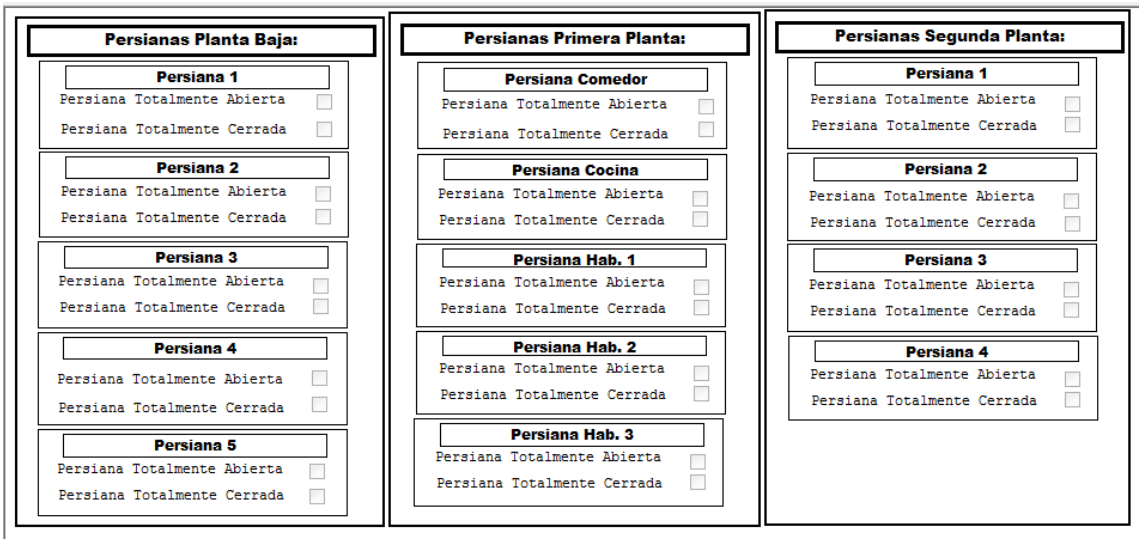


Figura 9-9 Pantalla de operador Persianas

9.1.6.3. PANTALLA DE OPERADOR PUERTAS

Desde esta pantalla de operador podremos simular las señales de entrada que recibimos de las puertas automáticas. Los fines de carrera de puerta abierta, cerrada, fotocélula y sobrecalentamiento.

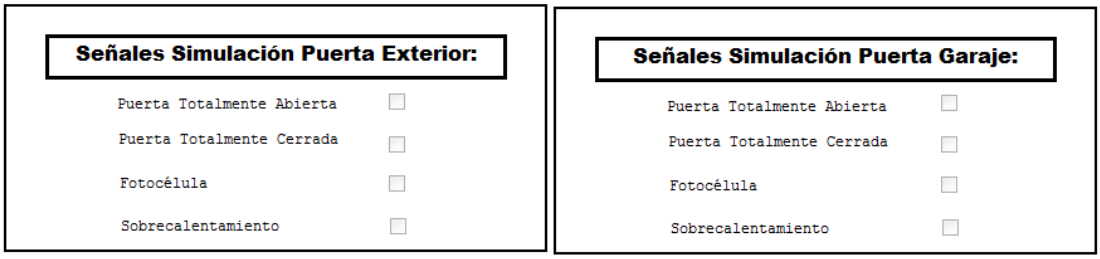


Figura 9-10 Pantalla de operador Puertas vivienda

9.1.6.4. PANTALLA DE OPERADOR SENSORES

Desde esta pantalla de operador podremos simular las señales de entrada que recibimos de cada uno de los sensores de movimiento. Movimiento si o no.

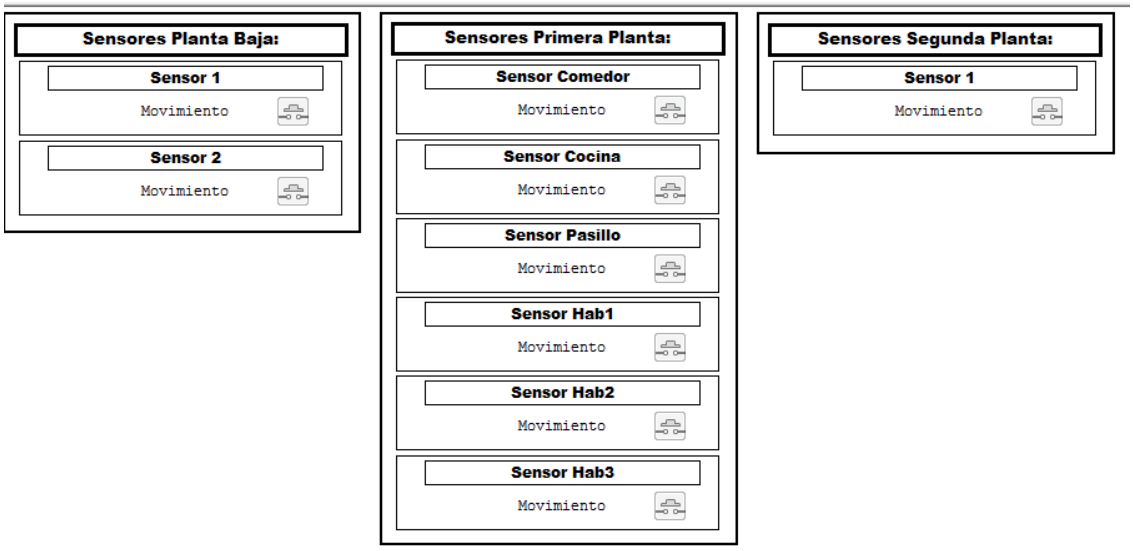


Figura 9-11 Pantalla de operador Sensores de movimiento

10. CONCLUSIONES Y FUTURAS LÍNEAS DE TRABAJO

10.1. CONCLUSIONES

El objetivo de este proyecto era el diseño de un sistema de control de una vivienda utilizando componentes comerciales que los fabricantes de equipos de automatización industrial disponen en el mercado.

Para el desarrollo del proyecto me he centrado en las soluciones que ofrece un fabricante concreto, Schneider Electric.

De su gama de productos se han seleccionados elementos de gama media. He descartado equipos de gama baja y gama alta pues para el desarrollo de mi futura carrera profesional creo que es la que más me puede aportar.

A lo largo del proyecto se han repasado las diferentes alternativas de arquitectura, buses y hardware que se podrían haber usado para la

implementación del sistema y se han justificado las decisiones tomadas. Se han dimensionado y configurado los tres elementos de adquisición de datos distribuidos (Islas Advantys), programado el PLC que gestiona el control y desarrollado el SCADA con los mímicos necesarios para que el usuario final pueda interactuar.

Desde mi punto de vista los objetivos del proyecto han quedado cubiertos. Además, todo lo que se ha descrito en el proyecto ha sido probado en equipos físicos reales prestados por Navantia-Sistemas. Se han hecho pruebas con ellos y el sistema ha quedado totalmente operativo.

La realización de este proyecto me ha ayudado a moverme con soltura en el marco de la automatización industrial, teniendo que consultar guías de selección o catálogos de productos, manuales de instrucciones de equipos, guías de formación, etc.

El aporte extra que me ha dado el tener la posibilidad de enfrentarme con los equipos reales, más allá del desarrollo teórico del proyecto, ha sido fundamental, aunque a veces, desalentador por la infinidad de ensayos de pruebas y error que he tenido que realizar para la puesta en servicio de los componentes individuales y la integración del conjunto de ellos en el proyecto. Sin duda, me servirá en el futuro desarrollo de carrera profesional.

10.2. FUTURAS LÍNEAS DE TRABAJO

Tras la elaboración del proyecto se podrían abrir nuevas líneas futuras de desarrollo.

En este proyecto me he ceñido exclusivamente al desarrollo del sistema de control sin hacer mención al montaje de los componentes en cuadros de control, estudio de detalle de la sensorización e interface con el PLC, instalación de los componentes en la vivienda de acuerdo a la normativa vigente, etc.

Otro punto de interés sería el poder conectar todo el sistema a internet y poder tener acceso a él desde una aplicación en un teléfono móvil.

También se podría ampliar este proyecto con la implementación de escenarios predefinidos que se lanzaran por parte del usuario y se ejecutasen

automáticamente como “*me voy a dormir*”, “*salgo de casa*”, “*como si estuviese en casa*”, etc.

Los puntos indicados anteriormente podrían ser de utilidad y ayudarían a completar el proyecto.

11. BIBLIOGRAFIA

Advantys STB Configuration Software 3.0 Quick Start Guide:
https://download.schneider-electric.com/files?&p_enDocType=User%20guide&p_File_Name=31002962_K01_000_04.pdf&p_Doc_Ref=31002962K01000#search

Advantys STB Standard Ethernet Modbus TCP/IP Network Interface Module Applications Guide:
https://download.schneider-electric.com/files?p_enDocType=User+guide&p_File_Name=31003688_K01_000_11.pdf&p_Doc_Ref=31003688K01000

Unity Pro Lenguajes y estructura del programa Manual de referencia:
https://download.schneider-electric.com/files?&p_enDocType=User%20guide&p_File_Name=35006147_K01_000_23.pdf&p_Doc_Ref=35006147K01000#search

Advantys Configuration Software Manual de inicio rápido para usuarios de Advantys
https://download.schneider-electric.com/files?&p_enDocType=User%20guide&p_File_Name=33004252_K01_000_04.pdf&p_Doc_Ref=33004252K01000#search

Modicon M340 para Ethernet Procesadores y módulos de comunicaciones Manual del usuario:
https://download.schneider-electric.com/files?&p_enDocType=User%20guide&p_File_Name=31007134_K01_000_17.pdf&p_Doc_Ref=31007134K01000#search

EcoStruxure™ Control Expert Comunicación Biblioteca de bloques:

https://download.schneider-electric.com/files?&p_enDocType=User%20guide&p_File_Name=33002530_K01_000_22.pdf&p_Doc_Ref=33002530K01000#search

Vijeo Designer Tutorial:

https://download.schneider-electric.com/files?&p_enDocType=User%20guide&p_File_Name=Vijeo-Designer-Starting-guide-English.pdf&p_Doc_Ref=VD-userguide-V6.2#search

Modbus organization:

https://modbus.org/