



# **Escuela Politécnica de Ingeniería de Gijón**

**Grado en Ingeniería Electrónica Industrial y Automática**

**Área de Ingeniería de Sistemas y Automática**

## **Implantación prestaciones IoT en piso piloto domótico (*E-Llar*) basado en SIMATIC S7- 1200 de Siemens**

**BARRIOCANAL GARCÍA, CELIA**

**TUTOR: Felipe Mateos Martín**

**FECHA: JULIO 2024**

## Resumen

El objetivo de este proyecto es actualizar las funcionalidades del llamado piso *E-Llar*, un laboratorio de domótica situado en la Escuela Politécnica de Gijón. Para ello se requiere un paso previo que consiste en resolver los problemas actuales en los sensores y actuadores, y recablear estos componentes al controlador. Estos elementos requieren una revisión y mejora, además de la incorporación de actualizaciones para optimizar las instalaciones de manera integral.

Una vez llevada a cabo esta revisión se ha implementado una arquitectura domótica basada en un autómatas programable industrial de la serie SIMATIC S7-1200 de Siemens. Para el desarrollo de los programas de control de las instalaciones se utiliza la herramienta de software TIA Portal, ya que proporciona una plataforma robusta y versátil para la programación y gestión de sistemas automatizados. Se ha diseñado e implantado un programa de control flexible y robusto que cubre gran parte de las funcionalidades previstas (iluminación, calefacción, alarmas técnicas y de intrusión, cargas, toldo, ...). Asimismo, se ha desarrollado una interface de usuario basado en una pantalla táctil HMI, aunque en modo virtual, desde la cual, localmente, el usuario podrá acceder a las funciones del sistema y visualizar el estado de los sensores y actuadores.

Para finalizar, se ha instalado una pasarela denominada IoT2040 para cubrir algunas prestaciones avanzadas como la utilización de interfaces remotas adaptables a dispositivos móviles, envío y recepción de mensajes vía Telegram y un ejemplo de almacenamiento y acceso a bases de datos.

# Índice de Contenido

<b>ÍNDICE DE CONTENIDO .....</b>	<b>1</b>
<b>1. INTRODUCCIÓN .....</b>	<b>12</b>
1.1. ANTECEDENTES .....	12
1.2. OBJETIVOS Y ALCANCE.....	13
1.3. ORGANIZACIÓN DE DOCUMENTO .....	14
<b>2. ESTADO DEL ARTE.....</b>	<b>15</b>
2.1. INTRODUCCIÓN.....	15
2.2. DOMÓTICA.....	15
2.2.1. <i>Ramas de la Domótica</i> .....	15
2.2.2. <i>Elementos característicos</i> .....	17
2.3. ARQUITECTURAS Y SISTEMAS .....	18
2.4. AUTOMATAS PROGRAMABLES.....	19
2.5. NUEVAS PRESTACIONES IOT .....	21
<b>3. ANÁLISIS Y DISEÑO DEL SISTEMA.....</b>	<b>23</b>
3.1. INTRODUCCIÓN .....	23
3.2. FUNCIONALIDADES A CONTROLAR.....	23
3.3. ARQUITECTURA PROPUESTA .....	24
3.4. SIMATIC S7-1200.....	26
3.5. PANTALLA TÁCTIL KTP 700.....	27
3.6. IOT-2040 .....	29
3.7. COMPONENTES SOFTWARE.....	30
3.7.1. <i>TIA Portal</i> .....	30
3.7.2. <i>Node-RED</i> .....	33
3.7.3. <i>Protocolo S7</i> .....	33
3.7.4. <i>SQLite</i> .....	34
<b>4. REVISIÓN DE COMPONENTES Y RECABLEADO .....</b>	<b>35</b>
4.1. INTRODUCCIÓN .....	35
4.2. REVISIÓN Y RECABLEADO .....	35
<b>5. PROGRAMACIÓN DEL PLC.....</b>	<b>38</b>
5.1. INTRODUCCIÓN .....	38

---

5.2.	CONFIGURACIÓN HARDWARE .....	38
5.3.	ESTRUCTURA DEL PROGRAMA DE CONTROL .....	40
5.4.	TIPOS DE DATOS DE USUARIO (UDT).....	43
5.4.1.	<i>UDT Alarma_Intrusion</i> .....	45
5.4.2.	<i>UDT Alarma_Tecnicas</i> .....	45
5.4.3.	<i>UDT Calefaccion</i> .....	45
5.4.4.	<i>UDT Carga</i> .....	46
5.4.5.	<i>UDT Luz</i> .....	46
5.4.6.	<i>UDT Persiana</i> .....	47
5.5.	ORGANIZACIÓN DE VARIABLES Y BLOQUES DE DATOS (DB) .....	48
5.5.1.	<i>DB_Alarmas_Intrusión</i> .....	49
5.5.2.	<i>DB_Alarmas_Tecnicas</i> .....	49
5.5.3.	<i>DB_Calefaccion</i> .....	50
5.5.4.	<i>DB_Cargas</i> .....	51
5.5.5.	<i>DB_FH</i> .....	52
5.5.6.	<i>DB_Luz</i> .....	52
5.5.7.	<i>DB_Persianas</i> .....	53
5.6.	FUNCIONES (FC) .....	54
5.7.	BLOQUES DE FUNCIÓN (FB) .....	55
5.7.1.	<i>ENTRADAS (FB5)</i> .....	56
5.7.2.	<i>FB_Control_Alarma_Intrusion (FB3)</i> .....	57
5.7.3.	<i>FB_Control_Alarma_Tecnica (FB6)</i> .....	60
5.7.4.	<i>FB_Control_Calefaccion (FB4)</i> .....	62
5.7.5.	<i>FB_Control_Cargas (FB7)</i> .....	63
5.7.6.	<i>FB_Control_Luz (FB1)</i> .....	67
5.7.7.	<i>FB_Persianas (FB8)</i> .....	70
5.7.8.	<i>SALIDAS (FB2)</i> .....	72
5.8.	MAIN (OB1).....	73
<b>6.</b>	<b>PANTALLA DE EXPLOTACIÓN (HMI) .....</b>	<b>78</b>
6.1.	INTRODUCCIÓN.....	78
6.2.	CREACIÓN Y CONFIGURACIÓN DE LA APLICACIÓN HMI (KTP700 BASIC PN) .....	79
6.3.	ORGANIZACIÓN DE VENTANAS .....	84
6.3.1.	<i>Ventana User_Admin</i> .....	85

---

---

6.3.1.	<i>Ventanas Home</i> .....	86
6.3.1.1.	<i>Ventana Home_Invitado</i> .....	86
6.3.1.2.	<i>Ventana Home_Residente</i> .....	87
6.3.2.	<i>Ventana Control_Alarmas</i> .....	88
6.3.2.1.	<i>Alarma Intrusión</i> .....	88
6.3.2.2.	<i>Alarma Técnica</i> .....	89
6.3.3.	<i>Ventana Control_Calefaccion</i> .....	90
6.3.4.	<i>Ventana Control_Cargas</i> .....	91
6.3.5.	<i>Ventana Control_Luces</i> .....	92
6.3.6.	<i>Ventana Control_Persianas</i> .....	93
<b>7.</b>	<b>APLICACIONES EN IOT</b> .....	<b>95</b>
7.1.	INTRODUCCIÓN .....	95
7.2.	PUESTA EN MARCHA Y FUNCIONAMIENTO.....	95
7.3.	IMPLEMENTACIÓN EN NODE-RED .....	98
7.3.1.	<i>Añadir usuario y contraseña</i> .....	99
7.3.2.	<i>Control remoto</i> .....	100
7.3.3.	<i>Comunicaciones S7</i> .....	101
7.3.4.	<i>Elementos básicos del dashboard</i> .....	105
7.3.4.1.	<i>Navegador de pantallas</i> .....	106
7.3.4.2.	<i>Iconos</i> .....	107
7.3.4.3.	<i>Switches</i> .....	109
7.3.4.4.	<i>Pulsadores:</i> .....	110
7.3.4.5.	<i>Entradas de texto</i> .....	111
7.3.5.	<i>Implementación de mensajes con Telegram</i> .....	111
7.3.5.1.	<i>Crear el Bot :</i> .....	111
7.3.5.2.	<i>Mensajes aviso de alarmas:</i> .....	113
7.3.5.3.	<i>Menú interactivo:</i> .....	114
7.3.5.4.	<i>Mensajes predeterminados:</i> .....	120
7.3.6.	<i>Base de datos</i> .....	121
7.4.	MANUAL DE USUARIO DEL DASHBOARD .....	124
7.4.1.	<i>Home</i> .....	124
7.4.2.	<i>Calefacción</i> .....	125
7.4.3.	<i>Intrusión</i> .....	126

---



---

7.4.4.	<i>Inundación</i> .....	127
7.4.5.	<i>Cargas</i> .....	128
7.4.6.	<i>Iluminación</i> .....	129
<b>8.</b>	<b>CONCLUSIONES Y FUTURAS AMPLIACIONES</b> .....	<b>131</b>
<b>9.</b>	<b>BIBLIOGRAFÍA</b> .....	<b>133</b>

# Índice de Figuras

ILUSTRACIÓN 1.1: PLANO RESUMIDO DE ESTANCIAS.....	13
ILUSTRACIÓN 2.1: SISTEMA DE CONTROL CENTRALIZADO [2].....	18
ILUSTRACIÓN 2.2: LENGUAJES PARA PROGRAMACIÓN DE PLCs .....	20
ILUSTRACIÓN 2.3: ARQUITECTURA DE UN SISTEMA DOMÓTICO BASADO EN COMPONENTES IoT .....	21
ILUSTRACIÓN 3.1:FASES DE DESARROLLO DEL PROYECTO.....	23
ILUSTRACIÓN 3.2: ARQUITECTURA PROPUESTA .....	26
ILUSTRACIÓN 3.3: SIMATIC S7-1214 .....	27
ILUSTRACIÓN 3.4: KTP700.....	29
ILUSTRACIÓN 3.5. SIMATIC IoT12040 .....	30
ILUSTRACIÓN 3.6:LOGO TIA PORTAL V17 .....	31
ILUSTRACIÓN 3.7: INTERFAZ INICIAL TIA PORTAL.....	31
ILUSTRACIÓN 3.8: SOFTWARE NECESARIO PARA TIA PORTAL.....	32
ILUSTRACIÓN 3.9: LOGO DE NODE-RED .....	33
ILUSTRACIÓN 3.10 SQLITE LOGO .....	34
<i>ILUSTRACIÓN 4.1: CONEXIONES PHOENIX CONTACT PLC.....</i>	<i>37</i>
<i>ILUSTRACIÓN 4.2: CONEXIONES</i>	
ILUSTRACIÓN 5.1: AJUSTES INICIALES PLC.....	38
ILUSTRACIÓN 5.2: CONFIGURACIÓN DE DISPOSITIVOS .....	39
ILUSTRACIÓN 5.3: INTERFAZ PROFINET.....	39
ILUSTRACIÓN 5.4: ESTABLECER CONEXIÓN ONLINE.....	39
ILUSTRACIÓN 5.5: VISTA DE REDES .....	40
ILUSTRACIÓN 5.6: HMI-PLC-TIA PORTAL .....	40
ILUSTRACIÓN 5.7: BLOQUES DEL PROGRAMA .....	43
ILUSTRACIÓN 5.8: TIPOS DE DATOS DE USUARIO CREADOS .....	44
<i>ILUSTRACIÓN 5.9: UDT_ALARM_AINTRUSION .....</i>	<i>45</i>
ILUSTRACIÓN 5.10: UDT ALARMA TECNICA.....	45
ILUSTRACIÓN 5.11: UDT CALEFACCION .....	46
ILUSTRACIÓN 5.12:UDT CARGA .....	46

---

ILUSTRACIÓN 5.13: UDT LUZ .....	47
ILUSTRACIÓN 5.14: UDT PERSIANAS .....	47
ILUSTRACIÓN 5.15: DATA BLOCKS .....	48
ILUSTRACIÓN 5.16: DB_ALARMAS_INTRUSION .....	49
ILUSTRACIÓN 5.17: DB_ALARMAS_TECNICAS .....	50
ILUSTRACIÓN 5.18: DB_CALEFACCION.....	51
ILUSTRACIÓN 5.19: DB_CARGAS .....	51
ILUSTRACIÓN 5.20: DB_FH .....	52
ILUSTRACIÓN 5.21: DB_LUZ.....	53
ILUSTRACIÓN 5.22: DB_PERSIANAS .....	53
ILUSTRACIÓN 5.23: FC FECHA_Y_HORA.....	54
ILUSTRACIÓN 5.24: CONFIGURACIÓN FECHA Y HORA ACTUAL.....	55
ILUSTRACIÓN 5.25: ÁRBOL FB .....	56
ILUSTRACIÓN 5.26: ENTRADAS.....	57
ILUSTRACIÓN 5.27: VARIABLES FB_CONTROL_ALARMAS_INTRUSION.....	58
ILUSTRACIÓN 5.28: CRONOGRAMA ALARMA INTRUSIÓN .....	58
ILUSTRACIÓN 5.29: NETWORK 1 FB_CONTROL_ALARMAS_INTRUSION .....	59
ILUSTRACIÓN 5.30: NETWORK 2 FB_CONTROL_ALARMAS_INTRUSION .....	59
ILUSTRACIÓN 5.31: NETWORK 3 FB_CONTROL_ALARMAS_INTRUSION .....	60
ILUSTRACIÓN 5.32: NETWORK 4 Y 5 FB_CONTROL_ALARMAS_INTRUSION .....	60
ILUSTRACIÓN 5.33: VARIABLES FB_CONTROL_ALARMAS_TECNICA .....	61
ILUSTRACIÓN 5.34: NETWORK 1 Y 2 FB_CONTROL_ALARMAS_TECNICA .....	61
ILUSTRACIÓN 5.35: NETWORK 3 FB_CONTROL_ALARMAS_TECNICA .....	62
ILUSTRACIÓN 5.36: VARIABLES FB_CONTROL_CALEFACCION.....	62
ILUSTRACIÓN 5.37: NETWORK 1 FB_CONTROL_CALEFACCION.....	63
ILUSTRACIÓN 5.38: NETWORK 2 FB_CONTROL_CALEFACCION.....	63
ILUSTRACIÓN 5.39: VARIABLES FB_CONTROL_CARGAS .....	64
ILUSTRACIÓN 5.40: NETWORK 1 FB_CONTROL_CARGAS .....	64
ILUSTRACIÓN 5.41: NETWORK 2 FB_CONTROL_CARGAS .....	65
ILUSTRACIÓN 5.42: NETWORK 3 FB_CONTROL_CARGAS .....	65



---

ILUSTRACIÓN 5.43: NETWORK 4 Y 5 DEL FB_CONTROL_CARGAS .....	66
ILUSTRACIÓN 5.44: NETWORK 6 DEL FB_CONTROL_CARGAS .....	66
ILUSTRACIÓN 5.45: NETWORK 7 FB_CONTROL_CARGAS .....	67
ILUSTRACIÓN 5.46: NETWORK 8 FB_CONTROL_CARGAS .....	67
ILUSTRACIÓN 5.47: VARIABLES FB_CONTROL_LUZ.....	68
ILUSTRACIÓN 5.48: NETWORK 1 FB_CONTROL_LUZ.....	68
ILUSTRACIÓN 5.49: NETWORK 2 FB_CONTROL_LUZ.....	68
ILUSTRACIÓN 5.50: NETWORK 3 FB_CONTROL_LUZ.....	69
ILUSTRACIÓN 5.51: NETWORK 4 Y 5 FB_CONTROL_LUZ.....	69
ILUSTRACIÓN 5.52: NETWORK 6 Y 7 FB_CONTROL_LUZ .....	70
ILUSTRACIÓN 5.53: NETWORK 8 FB_CONTROL_LUZ.....	70
ILUSTRACIÓN 5.54: VARIABLES FB_PERIASANAS .....	71
ILUSTRACIÓN 5.55: NETWORK 1 FB_PERSIANAS .....	71
ILUSTRACIÓN 5.56: NETWORK 2 FB_PERSIANAS .....	72
ILUSTRACIÓN 5.57: CÓDIGO FB_SALIDAS.....	72
ILUSTRACIÓN 5.58: OB1 COLAPSADO.....	73
ILUSTRACIÓN 5.59: NETWORK 1 Y 2 MAIN (OB1).....	74
ILUSTRACIÓN 5.60: NETWORK 3 MAIN (OB1).....	74
ILUSTRACIÓN 5.61: NETWORK 4 MAIN (OB1).....	75
ILUSTRACIÓN 5.62: NETWORK 5 MAIN (OB1).....	75
ILUSTRACIÓN 5.63: NETWORK 6 MAIN (OB1).....	75
ILUSTRACIÓN 5.64: NETWORK 7 MAIN (OB1).....	76
ILUSTRACIÓN 5.65: NETWORK 8 MAIN (OB1).....	76
ILUSTRACIÓN 5.66: NETWORK 9 MAIN (OB1).....	76
ILUSTRACIÓN 5.67: NETWORK 10 MAIN (OB1).....	77
ILUSTRACIÓN 6.1:HMI KTP700.....	78
ILUSTRACIÓN 6.2: CATALOGO DE HARDWARE .....	79
ILUSTRACIÓN 6.3: ÁRBOL DEL PROYECTO EN EL HMI .....	80
ILUSTRACIÓN 6.4: DEVICE CONFIGURATION DEL HMI.....	80
ILUSTRACIÓN 6.5: CONFIGURACIÓN IP DEL HMI .....	81

---

ILUSTRACIÓN 6.6: PERMITIR COMUNICACIÓN CON HMI.....	81
ILUSTRACIÓN 6.7: VISTA DE REDES .....	82
ILUSTRACIÓN 6.8: CONEXIONES DEL ÁRBOL DEL HMI .....	82
ILUSTRACIÓN 6.9: HMI TAGS.....	83
ILUSTRACIÓN 6.10: GESTIÓN DE PERMISOS HMI .....	83
ILUSTRACIÓN 6.11: USUARIOS DEL HMI .....	84
ILUSTRACIÓN 6.12: PANTALLAS DEL HMI.....	84
ILUSTRACIÓN 6.13: START SIMULATION .....	85
ILUSTRACIÓN 6.14: INICIO DE SESIÓN EN EL HMI.....	85
ILUSTRACIÓN 6.15: LISTADO DE USUARIOS .....	86
ILUSTRACIÓN 6.16: HOME INVITADO .....	87
ILUSTRACIÓN 6.17: HOME RESIDENTE .....	88
ILUSTRACIÓN 6.18: HMI ALARMA INTRUSIÓN .....	89
ILUSTRACIÓN 6.19 HMI ALARMAS TÉCNICAS.....	90
ILUSTRACIÓN 6.20: HMI CALEFACCIÓN .....	91
ILUSTRACIÓN 6.21: HMI CONTROL CARGAS .....	92
ILUSTRACIÓN 6.22 HMI LUCES.....	93
ILUSTRACIÓN 6.23 HMI PERSIANAS .....	94
ILUSTRACIÓN 7.1: GRABAR SO .....	96
ILUSTRACIÓN 7.2: INTRODUCCIÓN TARJETA SD .....	96
ILUSTRACIÓN 7.3: PuTTY .....	97
ILUSTRACIÓN 7.4: SETUP IoT2040 .....	98
ILUSTRACIÓN 7.5: RUTA SETTINGS.JS .....	99
ILUSTRACIÓN 7.6: ABRIR CON BLOC DE NOTAS .....	99
ILUSTRACIÓN 7.7: CÓDIGO SETTINGS.JS .....	99
ILUSTRACIÓN 7.8: ACCESO CON CONTRASEÑA .....	100
ILUSTRACIÓN 7.9: CREAR TÚNEL NUEVO .....	101
ILUSTRACIÓN 7.10: INSTALACIÓN NODOS S7 .....	102
ILUSTRACIÓN 7.11: PERMITIR ACCESO PUT/GET .....	103
ILUSTRACIÓN 7.12:OPTIMIZED BLOCK ACCESSS.....	103

---

ILUSTRACIÓN 7.13: MENÚ DASHBOARD .....	105
ILUSTRACIÓN 7.14: HOME DASHBOARD.....	106
ILUSTRACIÓN 7.15: SELECTOR DE PANTALLAS DEL DASHBOARD .....	106
ILUSTRACIÓN 7.16: DROPDOWN DEL DASHBOARD .....	107
ILUSTRACIÓN 7.17: FLUJO ICONOS DASHBOARD .....	108
ILUSTRACIÓN 7.18: EJEMPLO ÍCONOS DASHBOARD.....	109
ILUSTRACIÓN 7.19: FLUJO SWITCHES DASHBOARD .....	109
ILUSTRACIÓN 7.20: EJEMPLOS SWITCHES DASHBOARD.....	110
ILUSTRACIÓN 7.21: FLUJO PULSADORES DASHBOARD .....	110
ILUSTRACIÓN 7.22: EJEMPLOS BOTONES DASHBOARD .....	110
ILUSTRACIÓN 7.23: FLUJO INTRODUCIR CIFRAS .....	111
ILUSTRACIÓN 7.24: CLAVE_IN DESDE EL DASHBOARD.....	111
ILUSTRACIÓN 7.25: CONTACTAR CON BOTFATHER .....	112
ILUSTRACIÓN 7.26: CREACIÓN DEL BOT .....	112
ILUSTRACIÓN 7.27: PAQUETE NODE-RED-CONTRIB-TELEGRAMBOT .....	113
ILUSTRACIÓN 7.28: FLUJO MENSAJE ALARMA.....	113
ILUSTRACIÓN 7.29: ESQUEMA MENÚS TELEGRAM.....	114
ILUSTRACIÓN 7.30: MENÚ ALARMAS TELEGRAM .....	115
ILUSTRACIÓN 7.31: FLUJO MENÚ ALARMAS .....	115
ILUSTRACIÓN 7.32: NODO SWITCH ALARMAS .....	116
ILUSTRACIÓN 7.33: FLUJO INTRUSION_COCINA .....	116
ILUSTRACIÓN 7.34: FLUJO SIMULACIÓN DE PRESENCIA .....	117
ILUSTRACIÓN 7.35: MENÚ SIMULACIÓN PRESENCIA TELEGRAM.....	117
ILUSTRACIÓN 7.36: MENÚ ESTADO PISO .....	118
ILUSTRACIÓN 7.37: FLUJO MENÚ ALARMAS .....	118
ILUSTRACIÓN 7.38: NODO SWITCH ESTADO PISO .....	119
ILUSTRACIÓN 7.39: FLUJO VISUALIZAR DATOS .....	119
ILUSTRACIÓN 7.40: VISUALIZAR ESTADO LUCES .....	120
ILUSTRACIÓN 7.41: FLUJO COMANDOS TELEGRAM.....	120
ILUSTRACIÓN 7.42: OPCIONES TELEGRAM .....	121

---

---

ILUSTRACIÓN 7.43: PAQUETE SQL NODE-RED .....	122
ILUSTRACIÓN 7.44: FLUJOS CREACIÓN Y DESTRUCCIÓN DE TABLAS .....	122
ILUSTRACIÓN 7.45: FLUJO SQL.....	123
ILUSTRACIÓN 7.46: VENTANA REGISTRO ALARMAS.....	123
<i>ILUSTRACIÓN 7.47: HOME DASHBOARD .....</i>	<i>125</i>
ILUSTRACIÓN 7.48: CALEFACCIÓN DASHBOARD.....	126
ILUSTRACIÓN 7.49: INTRUSIÓN DASHBOARD .....	127
ILUSTRACIÓN 7.50: INUNDACIÓN DASHBOARD .....	128
ILUSTRACIÓN 7.51: CARGAS DASHBOARD .....	129
ILUSTRACIÓN 7.52: ILUMINACIÓN DASHBOARD .....	130

# Índice de Tablas

TABLA 3.1: TABLA CARACTERÍSTICAS SIMATIC S7-1214 .....	27
TABLA 3.2: TABLA CARACTERÍSTICAS KTP 700.....	28
TABLA 3.3: CARACTERÍSTICAS SIMATIC IoT12040.....	30
TABLA 4.1: ENTRADAS PLC.....	36
TABLA 4.2: SALIDAS PLC .....	36
TABLA 7.1: VARIABLES PLC EN NODE-RED .....	104

# 1. Introducción

## 1.1. ANTECEDENTES

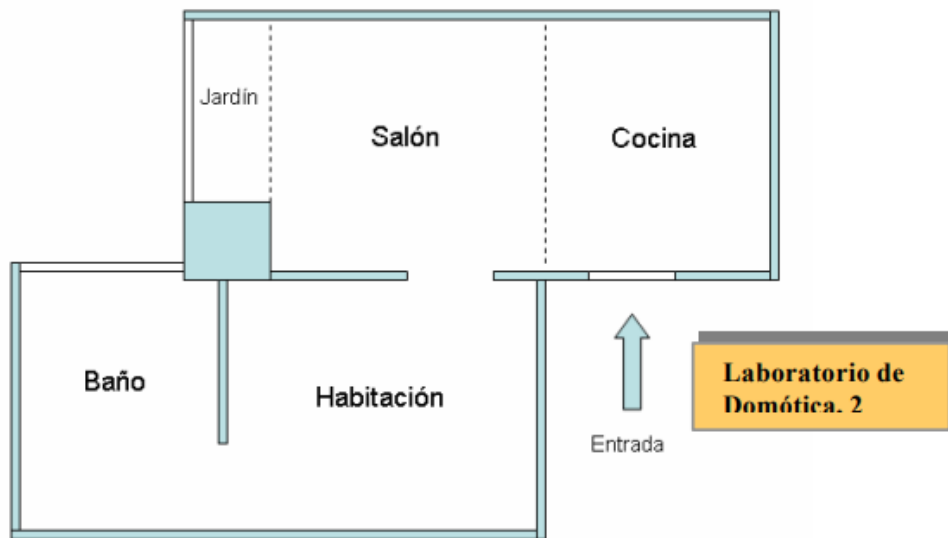
El proyecto del piso piloto *E-llar* consiste en una infraestructura que simula de forma realista una vivienda con los sensores y actuadores correspondientes y el funcionamiento automatizado de algunas de sus instalaciones. Nace como una iniciativa del grupo de investigación GENIA (Entornos Integrados de Automatización) de la Universidad de Oviedo.

En la Feria de la Construcción FICNI 2002, la Universidad de Oviedo coordinó los trabajos de diseño y construcción de cinco viviendas automatizadas con diferentes tecnologías, donde los fabricantes presentaban sus productos. A raíz de este proyecto, la promotora del evento, la Cámara de Comercio de Gijón, donó uno de los pisos piloto a la Escuela Politécnica de Ingeniería. Por su parte, la empresa Siemens proporcionó los componentes domóticos de la vivienda, presentando su producto SIMATIC, con el objetivo de apoyar el desarrollo del proyecto en la universidad.

Con todo este material, se llevó a cabo un gran esfuerzo por parte de GENIA, con la ayuda del Vicerrectorado de Infraestructuras de la Universidad, que cubrió parte de los costes. Esto permitió finalmente disponer de esta instalación, cuya inauguración tuvo lugar el 28 de octubre de 2003.[1]

El enfoque del proyecto consistía en crear una vivienda con elementos automatizados que sirviera como demostración académica de una instalación domótica real. Esto tendría diversas funcionalidades, como el desarrollo y la aplicación de nuevas tecnologías domóticas, la realización de clases prácticas y cursos de automatización de hogares, y servir como vehículo promotor del grupo de investigación.

Esta instalación está ubicada en la planta baja del Edificio Departamental N°2 del Campus de Viesques, en el área de Ingeniería de Sistemas y Automática bajo el nombre Laboratorio de Domótica número 2 (Piso Piloto *E-llar*).



*Ilustración 1.1: Plano resumido de estancias*

Cuenta con aproximadamente 25m<sup>2</sup> y pretende simular un entorno lo más parecido posible a las condiciones que se encontraría en una vivienda real. Para ello cuenta con cuatro estancias: cocina, salón, habitación y aseo.

Si bien el sistema domótico implantado inicialmente estaba basado en un controlador S7-200 y el display de texto TD-200, también con el tiempo se han desarrollado otros trabajos con implantación de autómatas programables ILC-131 de Phoenix Contact o el CX-100 de Beckhoff. Incluso estos sistemas, cuyo control es intercambiable mediante la conexión de cables manguera para direccionar según proceda las señales de entradas y salidas (sensores y actuadores) de la instalación, también se llevó a cabo la integración de otra tecnología basada en el sistema propietario Busing de Ingenium lo que ha supuesto una plataforma continua para el desarrollo de trabajos en diferentes asignaturas y trabajos fin de estudios.

Con este trabajo fin de grado se pretende incorporar un autómata programable SIMATIC S7-1200 de nueva generación que permita implantar una solución más eficiente en cuanto a la programación, interface de usuario y prestaciones avanzadas.

## 1.2. OBJETIVOS Y ALCANCE

El objetivo principal de este proyecto es la revisión y actualización de las instalaciones del piso piloto *E-llar* para mejorar la conectividad y funcionalidad del sistema de control automatizado. Para ello se requiere la adaptación del cableado para la integración de un autómata programable Simatic S7-1200, la identificación y corrección de errores en los elementos sensores y actuadores, y la implementación de un control esencial sobre las principales instalaciones de la vivienda. Además, se busca proporcionar una estructura de programación organizada y flexible mediante el uso de tipos de datos de usuario y bloques funcionales.

También se considera el desarrollo de aplicaciones en el módulo IoT2040 de Siemens, la creación de un dashboard accesible a nivel local y remoto, y la

implementación de un sistema de mensajería a través de Telegram para la notificación de eventos y el control de ciertos elementos de la instalación.

Por último, como ejemplo de almacenamiento y acceso a base de datos se usa una de las funcionalidades (gestión de alarmas) para disponer de un registro detallado de los eventos relacionados con esta función.

La documentación detallada de todo el sistema permitirá a futuros proyectos contar con una base sólida y clara para la continuación y ampliación de los trabajos realizados, garantizando así la sostenibilidad y evolución del piso piloto *E-Llar* en el ámbito de la automatización y el control domótico.

### **1.3. ORGANIZACIÓN DE DOCUMENTOS**

En el presente trabajo fin de grado, se incluyen varios documentos y anexos que complementan y apoyan la memoria principal. La estructura de la documentación se ha organizado de manera que se facilite la comprensión y el acceso a la información relevante para el proyecto.

- Memoria: documento actual. En ella se detalla todo el proceso de desarrollo del proyecto, desde los objetivos iniciales hasta las conclusiones. Se incluye una descripción exhaustiva de la metodología empleada, los resultados obtenidos, y las conclusiones derivadas del trabajo realizado.
- Anexos: proporcionan información adicional y complementaria al contenido principal del proyecto:
  - Programa de TIA Portal: Se adjunta también el programa desarrollado en TIA Portal para el autómata programable Simatic S7-1200 y el HMI. Se divide en dos documentos, uno con el programa completo (TIA\_Portal1) y otro con los UDTs(TIA\_Portal2\_UDT).
  - JSON de los Flujos de Node-RED “e\_llar.json”: Se incluye un archivo JSON que contiene los flujos de trabajo desarrollados en Node-RED.



---

## 2. Estado del arte

### 2.1. INTRODUCCIÓN

En este capítulo se tratan algunos elementos que forman parte del contexto en el que se desarrolla este trabajo. En primer lugar, se enfoca la domótica como disciplina para la implantación de funcionalidades de gestión técnica de instalaciones en viviendas y edificios. Posteriormente se resumen las arquitecturas y sistemas que se utilizan habitualmente en este ámbito.

Puesto que el enfoque es utilizar un autómata programable (PLC, *Programmable Logic Controller*) se introducen estos equipos con especial significación al estándar de programación IEC 61131-3 que ha supuesto en los últimos años una herramienta indispensable para uniformizar el desarrollo de las aplicaciones software, en especial los lenguajes de programación.

Finalmente se indican algunos aspectos sobre la domótica y las capacidades de los dispositivos conectados a Internet de las Cosas (IoT, *Internet of Things*).

### 2.2. DOMÓTICA

La Real Academia Española de la Lengua define como domótica «el conjunto de sistemas que automatizan las diferentes instalaciones de una vivienda». Para CEDOM (Asociación Española de Domótica), la domótica es la «automatización y el control de la gestión inteligente de la vivienda, aportando seguridad técnica y de intrusión, ahorro energético, confort y comunicación entre los dispositivos y el usuario final».

En resumen: la vivienda domótica es aquella que permite una mayor calidad de vida, a través de la tecnología, ofreciendo un aumento del bienestar y de la seguridad de los usuarios, y una racionalización de los distintos consumos energéticos. [2]

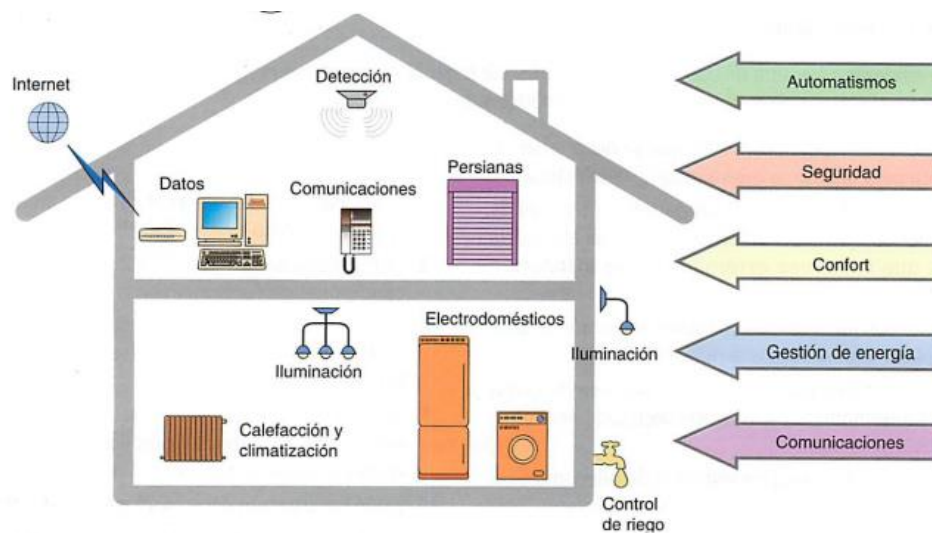
Asimismo, se considera a un dispositivo parte de un sistema domótico si es capaz de enviar y recibir información de otros elementos de la instalación. Es importante no confundir el concepto con otras soluciones automatizadas que a diferencia de los sistemas domóticos funcionan de manera aislada, sin comunicarse con otros servicios y dispositivos de la vivienda.[3]

#### 2.2.1. Ramas de la Domótica

A raíz de esto, se suele considerar que la domótica está enfocada en cuatro ramas principales:

- **Confort:** ayuda a la realización de trabajos domésticos disminuyendo el trabajo y aumentando la calidad de vida a base de automatizar tareas cíclicas que en otro tipo de vivienda requerirían la atención del usuario. Se aplica en funcionalidades como: control y regulación de la climatización, iluminación, persianas y toldos y riego automático.

- Gestión de energía:** ahorro en el consumo de energía gracias a la eficiente gestión de la energía y sus fuentes. Lo cual requiere una administración inteligente de iluminación, riego, climatización, cargas eléctricas, agua caliente sanitaria, electrodomésticos para un mejor aprovechamiento de recursos naturales y económicos.
- Seguridad:** personal, técnica y patrimonial, ya que la domótica proporciona sistemas para garantizar el funcionamiento del edificio o vivienda, la seguridad de las personas y la protección de las pertenencias. El sistema domótico detecta el evento que ha de proteger, genera una alarma y, realiza la actuación necesaria. Alguna de las posibles aplicaciones es: gestión de alarmas técnicas, alarmas médicas, control de acceso y presencia, control intrusión con vigilancia y simulación de presencia.
- Comunicaciones:** se utiliza el acceso a servicios de comunicación y a su integración para acceder a toda una variedad de servicios tanto de ocio como de trabajo. Algunas de las funcionalidades que proporciona la domótica en este ámbito son: telecontrol telefónico, transmisión de alarmas y telecontrol vía internet. [3] [2]



*Ilustración 2: Resumen de funcionalidades domóticas [2]*

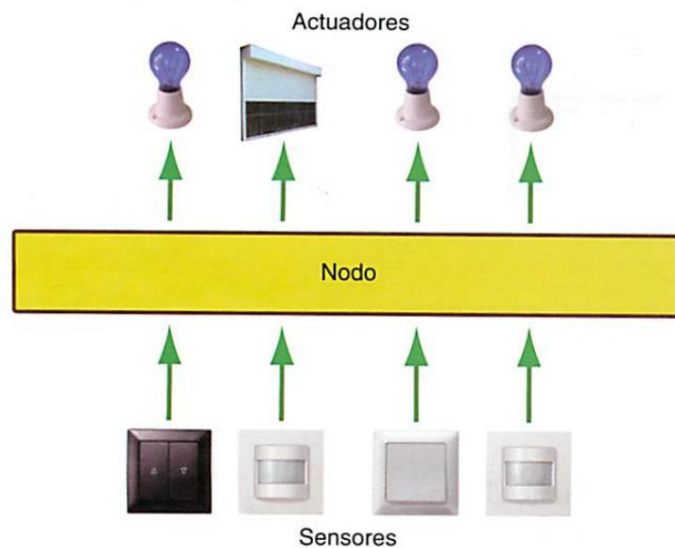
En conjunto, estos sistemas conforman las redes domóticas ya que integran en un solo sistema los elementos requeridos para el control y automatización de los distintos equipos domésticos. En conjunto con la red de entretenimiento, la red de voz y datos y la red eléctrica conforman la red doméstica, que se define como el soporte físico por el que se alimentan y comunican los usuarios, equipos y sistemas de las viviendas y edificios, cuya integración se puede llevar a cabo mediante pasarelas residenciales. Este trabajo se circunscribe a la parte domótica de gestión de instalaciones.[2]

---

### 2.2.2. Elementos característicos

Las instalaciones domóticas constan de manera genérica de ciertos elementos comunes:

- **Sensores:** dispositivos capaces de percibir señales del exterior provenientes de magnitudes físicas como pueden ser la presión la temperatura o la presencia de una persona en la vivienda. Se envían los datos recogidos, en forma de señales al sistema domótico. Algunos de los tipos de sensores que manejan los sensores pueden ser la detección de humo o gases, actuación sobre un interruptor o el sobrepaso de una temperatura estipulada.
- **Actuadores:** dispositivos encargados de recibir las señales del sistema domótico, en otras palabras: cualquier elemento que se active eléctricamente a raíz de las señales anteriormente nombradas se podría considerar un actuador. algunos ejemplos de sus aplicaciones son: activación de sirenas, regulación de luminosidad, control de motores de toldos y persianas o apertura y cierre de circuitos de agua o gases.
- **Preactuadores:** dispositivo conectado entre el actuador principal y el sistema automático(nodo). utilizar en aquellos casos en los que el consumo eléctrico del actuador o receptor a controlar no puede ser soportado directamente por el nodo domótico.
- **Nodo:** dispositivo encargado de recibir procesar y enviar las señales procedentes de los sensores hacia los actuadores. Dependiendo de la topología de los nodos hay dos tipos de sistemas. Los centralizados cuentan con un único nodo conectado a todos los sensores y actuadores (*Ilustración 2.1*). Por otra parte, los sistemas descentralizados o distribuidos cuentan con varios nodos interconectados entre sí a través de un bus de datos común. [2]



*Ilustración 2.1: Sistema de control centralizado [2]*

### 2.3. ARQUITECTURAS Y SISTEMAS

Para llevar a cabo las funciones anteriores existe diferentes arquitecturas y tecnologías que conecta y sincroniza los elementos de la instalación, típicamente organizados e sensores, actuadores, elementos de control, de visualización y dispositivos auxiliares.

Las arquitecturas pueden ser centralizadas, descentralizadas o mixtas encontrándose en todos los casos con soluciones estandarizadas o propietarias. Los medios de comunicación entre los diferentes componentes que forma la instalación son también variados, aunque se están imponiendo poco a poco las soluciones que emplea comunicaciones inalámbricas basadas en diferentes tecnologías y protocolos de comunicación.

Entre las tecnologías estándar destaca KNX que ha supuesto una plataforma de integración creciente con equipos de diferentes fabricantes y una gama de productos certificados que supera los 8000. La utilización una herramienta de desarrollo de productos común (ETS, *Engineering Tool Software*) permite la configuración independiente del fabricante soportando una amplia posibilidad para la programación de instalaciones complejas. KNX es un sistema descentralizado que facilita el conexionado de elementos y el protocolo hace factible la interacción lógica permitiendo una gran flexibilidad a la instalación.

Una de las soluciones propietarias de características similares a KNX es el sistema BUSING de Ingenium, empresa asturiana de domótica que diseña y fabrica desde hace más de 20 años este sistema descentralizado de elevadas prestaciones. Este sistema está implantado en el piso piloto si bien no se trabaja con dicha tecnología por lo que ya no será mencionado en el resto de este documento.

Por lo demás, y como se había comentado, el piso piloto E-Llar cuenta además con un conjunto de sensores y actuadores conectados de forma centralizada a unas bornas en

el cuadro eléctrico, desde donde es posible canalizar las señales hacia un controlador programable como puede ser un PLC.

## **2.4. AUTOMATAS PROGRAMABLES**

Un autómatas programable, o PLC, es un dispositivo electrónico fundamental en la automatización industrial, utilizado para controlar máquinas y procesos. Estos dispositivos cuentan con un hardware duradero y fiable, capaz de soportar ambientes industriales hostiles.

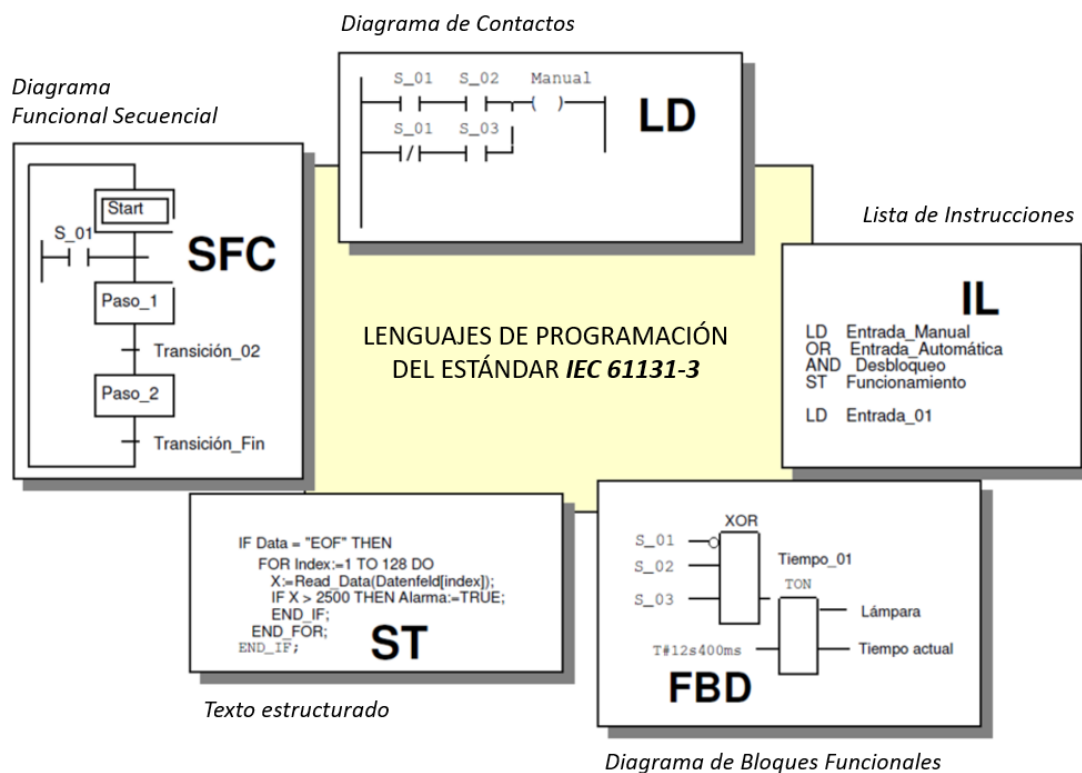
En la industria moderna, los PLCs (Controladores Lógicos Programables) son indispensables, ya que proporcionan un control preciso y confiable de los procesos industriales. Facilitan la sincronización y gestión de operaciones en tiempo real, lo cual es crucial para una automatización eficiente. Una de sus principales ventajas es su flexibilidad: pueden reprogramarse fácilmente para adaptarse a cambios en el sistema. Además, permiten la expansión mediante la adición de módulos suplementarios, como cuando se requieren más entradas o salidas.

Otra ventaja significativa de los PLCs (Controladores Lógicos Programables) es su capacidad para ofrecer diversas herramientas de diagnóstico y mantenimiento, lo que facilita la detección y corrección de errores de manera sencilla. Estas características hacen de los PLCs una herramienta indispensable en la optimización y eficiencia de la producción industrial.

Los autómatas programables compuestos, típicamente, por una CPU (Unidad Central de Procesamiento), módulos de entradas y salidas, y una fuente de alimentación. De forma adicional se pueden añadir algunos módulos para funciones especiales que de forma habitual tienen que ver con prestaciones de comunicación mediante estándares industriales (Serie, Ethernet, CAN, etc.). Esto permite la disponibilidad de configuración de periferia descentralizada lo que facilita el conexionado y la flexibilidad de las topologías de control y supervisión.

El IEC 61131-3 es el estándar mediante el cual se definen las especificaciones de la semántica y sintaxis de los lenguajes de programación de PLCs, incluyendo la estructura del lenguaje y el modelo de software. Por lo tanto, constituye la base real de homogenización de los lenguajes de programación en automatización industrial, unificando así cualquier programa, sin importar su origen. [4]

Como lenguajes de programación contemplados en el estándar IEC se encuentran dos los de tipo gráfico como Diagramas de Bloques de Función (FBD) y Diagrama de contactos (LD), así como otros dos de tipo literal, Texto Estructurado (ST) y Lista de Instrucciones. Algunos equipos se pueden programar también en Diagrama Funcional Secuencial (SFC) que es la implementación de la metodología Grafcet para el desarrollo de las aplicaciones. La ilustración siguiente resume los lenguajes de programación indicados en este párrafo.



*Ilustración 2.2: Lenguajes para programación de PLCs*

Aplicado al ámbito de la domótica, los controladores lógicos programables destacan por su fiabilidad. Al estar diseñados para entornos industriales muy exigentes y rudos, en el ámbito del hogar son capaces de garantizar un rendimiento consistente y duradero. Por otra parte, son dispositivos muy flexibles, lo cual permite adaptarlos a cualquier tipo de funcionalidad doméstica; pueden reprogramarse y ampliarse de manera sencilla en caso necesario.

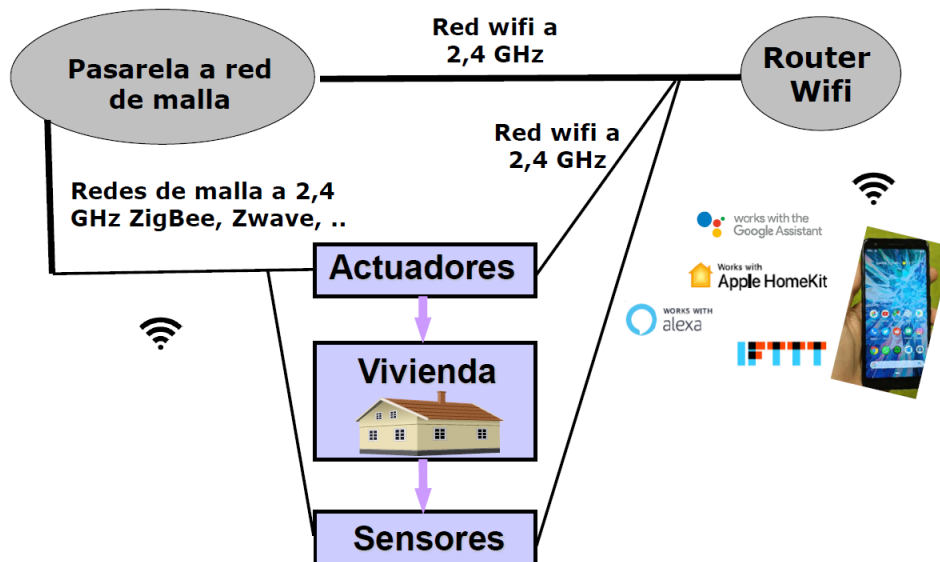
Además, estos dispositivos permiten comunicarse con una amplia gama de sistemas a través de protocolos de comunicación como Ethernet y Modbus. Gracias a esto, la integración de diferentes componentes del hogar se facilita notablemente, pudiéndose llevar un control centralizado de todos los dispositivos de la vivienda.

Trabajando complementariamente con los PLCs, es habitual y necesario incluso el uso de pantallas de explotación o sistemas SCADA (*Supervisory Control and Data Acquisition*). Estos sistemas proporcionan al operario una interfaz gráfica para interactuar con el sistema, generalmente a través de un HMI. Específicamente, el usuario podrá controlar, ajustar y supervisar en tiempo real todo tipo de procesos. Facilitan la visualización de datos críticos, acelerando la identificación y gestión de anomalías y potenciales problemas en el sistema. También permiten la configuración y ajuste remoto de las variables del PLC mediante prestaciones IoT, mejorando así la eficiencia y seguridad de los procesos a controlar. Por último, gracias al HMI se implementa la gestión y acceso de usuarios con sus correspondientes permisos y el registro de las acciones que realice cada uno. Esto asegura la trazabilidad y seguridad de las operaciones.[5]

## 2.5. NUEVAS PRESTACIONES IOT

Los sistemas domóticos e IoT (Internet de las Cosas) representan la evolución de la tecnología aplicada al hogar y otros entornos, integrando dispositivos inteligentes que pueden comunicarse entre sí y ser controlados de forma remota.

La arquitectura de un sistema domótico basado en componentes IoT presenta el aspecto general que se muestra en la imagen siguiente. Se suele disponer de una pasarela red que conecta en modo malla los dispositivos (sensores y actuadores) mediante redes inalámbricas tipo ZigBee, Zwave, etc. La mayoría disponen de interfaces de usuario con Apps en móvil para manejo en modo local y remoto, e incluso con asistentes de voz (Google Assistant o Alexa) en modo local. En ocasiones la propia red Wi-Fi sirve como sistema de interconexión de los dispositivos con el Router que da acceso para la configuración y gestión del sistema.



*Ilustración 2.3: Arquitectura de un sistema domótico basado en componentes IoT*

En el caso de utilizar equipos de control tipo PLCs, el análisis, actualización y acceso a los datos de forma constante no es posible que sean cubiertos con las actuales tecnologías sin que se empleen dispositivos externos que requieren más control y más esfuerzo para mantener todo organizado y flexible para adaptarse a cualquier futuro requisito de las instalaciones.

Por tanto, puesto que normalmente un PLC no cuenta con estas prestaciones avanzadas, se suele optar por incorporar otros elementos adicionales que se conectan de forma relativamente simple con el controlador y hacen de puente, también con el Router de la instalación. Por ejemplo, Siemens dispone de IoT2040 que será el equipo a utilizar, aunque dispone de otras versiones más avanzadas.

Podría haberse optado por emplear algún microcontrolador de propósito general como Raspberry Pi si bien adolece de la robustez que garantiza componentes de índole más industrial como el IoT2040 reseñado.



Comentar finalmente que en los últimos años se están abriendo paso soluciones de integración en un dispositivo para control en tiempo real del proceso según la norma IEC 61131-3 y la combinación con otros lenguajes de alto nivel. PLCnext de Phoenix Contact consta de un software abierto, un hardware de ingeniería modular, una comunidad global y un acceso libre a aplicaciones basadas en esta tecnología.

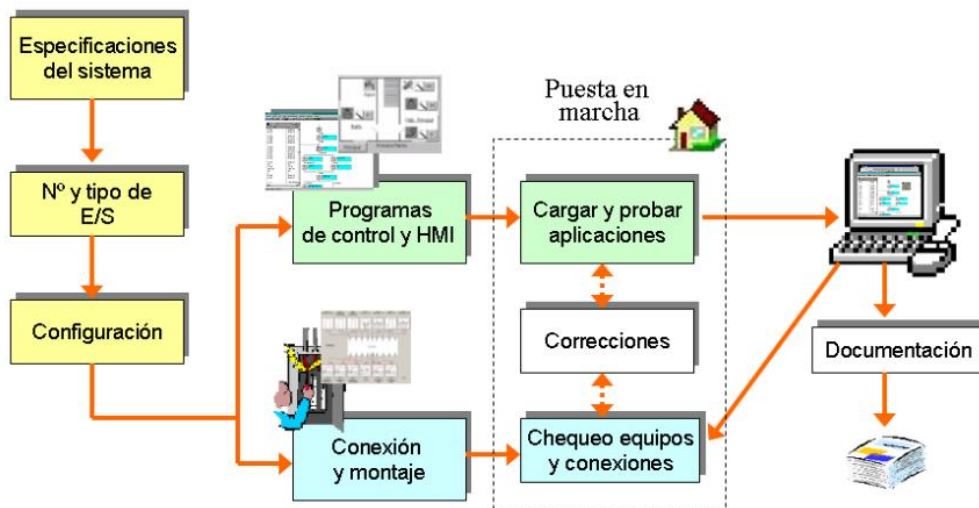


## 3. Análisis y diseño del sistema

### 3.1. INTRODUCCIÓN

En este apartado se detallará el análisis y diseño del sistema de control domótico implementado en el proyecto. El objetivo es proporcionar una visión general de las funcionalidades a controlar, la arquitectura propuesta y los componentes tanto de hardware como de software utilizados.

Como se muestra en la *Ilustración 3.1*, el proceso de diseño y puesta en marcha del sistema comienza con el análisis de las especificaciones del sistema, seguido por la identificación del número y tipo de entradas y salidas (E/S) necesarias, lo cual se corresponde con lo desarrollado en este apartado.



*Ilustración 3.1: Fases de desarrollo del proyecto*

Una vez configurado el sistema, se procede a la programación de los dispositivos de control y la interfaz HMI. La fase de conexión y montaje asegura que todos los componentes estén adecuadamente instalados y conectados.

Durante la puesta en marcha, las aplicaciones son cargadas y probadas en el sistema, permitiendo realizar las correcciones necesarias. Se verifica el funcionamiento de los equipos y las conexiones, asegurando la correcta operación del sistema. Finalmente, toda la información recopilada y los procedimientos implementados son documentados para futuras referencias y para el mantenimiento.

### 3.2. FUNCIONALIDADES A CONTROLAR

Inicialmente se plantea la programación de varias funcionalidades, a partir del conjunto de sensores y actuadores disponibles, así como las señales correctamente cableadas al controlador según se comentó en apartados anteriores. Estas funcionalidades son las siguientes:

- Control de varios circuitos de iluminación (salón, cocina, aseo, habitación): La gestión de luces se puede hacer de forma convencional utilizando los pulsadores colocados en cada estancia, también a través del configurado en las interfaces gráficas (pantalla o terminal móvil), pero puede establecer un modo de funcionamiento automático en el que se usan los sensores de presencia, luminosidad y horarios según las preferencias del usuario.
- Control de calefacción en salón-cocina: Se trabaja con dos modos posibles de operación. En el modo manual se considera exclusivamente la señal del termostato, por lo tanto, el usuario ha de introducir una temperatura objetivo. De ser esta inferior a la detectada en la estancia correspondiente, se enviará una señal de activación a la calefacción. De seleccionarse un modo de funcionamiento automático, se tendrán en cuenta además del termostato, el sensor de ventana abierta y el horario seleccionado para el funcionamiento de la calefacción.
- Control del toldo/persiana en zona salón: Se cuenta con dos modos de operación, según el usuario pretenda subir y bajar la persiana manualmente con los pulsadores físicos o los implementados digitalmente (pantalla o terminal móvil) o si prefiere la gestión automática condicionada por el horario la luz y lluvia diseñada para una gestión óptima de la luz en la vivienda
- Control de cargas con varios enchufes domóticos (en cocina, salón y habitación). Consiste en la posibilidad de que cada enchufe pueda ser gobernado teniendo en cuenta un modo manual, en el que el usuario puede activar y desactivar su funcionamiento a voluntad mediante el uso de la interface correspondiente, o un modo automático configurando una hora de encendido/apagado del equipo conectado a dicho enchufe.
- Control de alarmas técnicas: inundación, fuego/humo, caída de tensión). Detección, corte de agua y aviso (pantalla, señal acústica y luminosa, envío de mensaje). Este sistema puede quedar desactivado a voluntad y permitir el reset e inicialización para nuevo establecimiento de dicha funcionalidad.
- Control de alarmas por intrusión en cocina/salón y aseo: se ha planteado un sistema mediante el cual al activar el usuario la alarma introduciendo la contraseña correcta. Al detectarse presencia en la estancia correspondiente, tras un tiempo prudencial para dar la oportunidad de volver a introducir la clave adecuada, se dispare la alarma y notifique a los usuarios de la intrusión. Pudiendo desactivarse exclusivamente introduciendo la clave adecuada

### **3.3. ARQUITECTURA PROPUESTA**

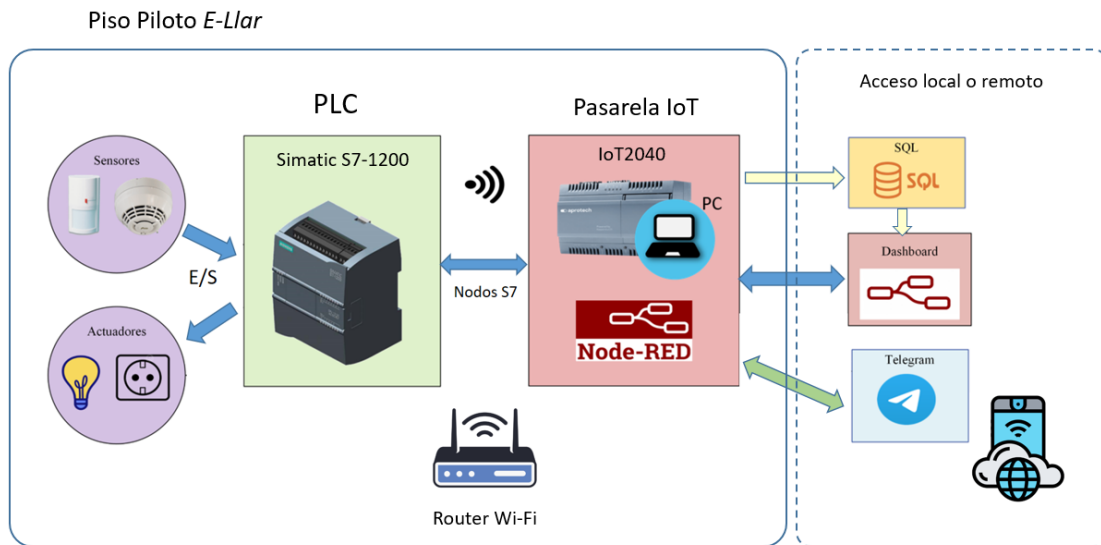
Para llevar a cabo el proyecto se ha planteado una arquitectura cuyo epicentro es un PLC SIMATIC S7-1200 de Siemens. Además, se ha empleado un conjunto de sensores y actuadores presentes en el piso piloto *E-Llar*. Por último, se han implementado una serie de funcionalidades avanzadas de IoT a través de Node-RED:

- Sensores: proporcionan datos críticos sobre el estado de las instalaciones, permitiendo monitorear y reaccionar ante diversas situaciones. Por ejemplo,

el sensor de inundación detecta la presencia de agua en áreas críticas, mientras que el detector de luminosidad genera un cambio de estado de su señal de salida cuando se supera o desciende el nivel de luminosidad de un umbral preestablecido.

- **Actuadores:** son dispositivos encargados de ejecutar acciones basadas en las señales recibidas del sistema de control. Estos dispositivos, como lámparas y enchufes, responden a las instrucciones del PLC para realizar tareas específicas, como encender o apagar luces y controlar otras cargas eléctricas.
- **PLC:** actúa como el cerebro del sistema, gestionando la automatización y el control de los actuadores en base a las señales de los sensores y las configuraciones del usuario. Además, el PLC envía y recibe variables a Node-RED para su procesamiento y uso en diversas aplicaciones IoT.
- **Node-RED:** se ejecuta en una pasarela IoT, como la disponible de Siemens denominada IoT2040, aunque a efectos de desarrollo y depuración se ha podido también usar un ordenador PC. Node-RED permite leer y escribir variables del PLC mediante nodos S7 y proporcionará varias funcionalidades adicionales:
  - **Dashboard:** Con el propio Node-RED se puede desarrollar una interfaz (dashboard) que permite visualizar y controlar en tiempo real algunas variables del PLC y por tanto las funcionalidades implantadas en el sistema, tanto a nivel local como remoto.
  - **Telegram:** La integración con Telegram permite gestionar y controlar de manera remota diversas variables del PLC. Los usuarios pueden visualizar el estado de las variables y modificarlas desde cualquier lugar, proporcionando una capa adicional de control y monitoreo.
  - **SQLite:** Es utilizado para registrar las alarmas activadas, especificando si la intrusión ocurre en la cocina o en la habitación, o si se detecta una inundación. Además, se puede acceder a los registros de la base de datos y presentar la información en formato de tabla y gráficos de pastel, facilitando una visualización cómoda y rápida. Este registro permitiría un seguimiento detallado y un análisis histórico de eventos.

La comunicación se puede establecer en modo Wireless mediante conexión Wi-Fi de los dispositivos al estar disponible dentro de la misma red o bien por cableado ethernet RJ45. También es posible la salida al exterior en modo remoto mediante la utilización de un Router Wi-Fi con los permisos correspondientes para conexión a Internet.



*Ilustración 3.2: Arquitectura propuesta*

### 3.4. SIMATIC S7-1200

El SIMATIC S7-1200 de Siemens es una familia de controladores lógicos programables (PLC) diseñada para aplicaciones de automatización industrial. Estos PLCs destacan por su flexibilidad y versatilidad para adaptarse a todo tipo de aplicaciones industriales. Dentro de esta gama, se dispone de diferentes modelos con distintas capacidades de entradas y salidas.

Esta gama de controladores incluye puertos Ethernet integrados que permiten la comunicación con diferentes dispositivos y sistemas utilizando protocolos como TCP/IP, Modbus, OPC UA (requiere licencia de runtime) y PROFINET.

En concreto, dentro de la familia S7-1200, el PLC elegido para este proyecto es el SIMATIC S7-1214C AC/DC/RLY, un PLC compacto equipado con 14 entradas digitales de 24 V DC, 10 salidas digitales de relé de 2 A cada una y 2 entradas analógicas de 0-10 V DC. Alimentado por una fuente de corriente alterna (AC) con un rango de tensión de 85-264 V AC, cuenta con una memoria de trabajo integrada de 100 KB y una memoria de carga de 4 MB, expandible con una tarjeta de memoria SIMATIC.

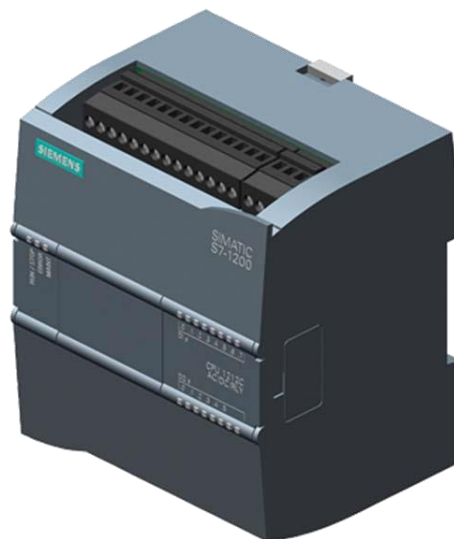
En términos de rendimiento, ofrece tiempos de procesamiento rápidos y una capacidad amplia de bloques de CPU, pudiendo manejar hasta 65,535 bloques de datos, funciones y bloques de funciones, así como contadores y temporizadores. Incluye un reloj de hardware en tiempo real con un respaldo de 480 horas y una desviación máxima de  $\pm 60$  segundos por mes. Está diseñado para operar en condiciones adversas, en un rango de  $-20^{\circ}$  a  $60^{\circ}\text{C}$ , y cuenta con un grado de protección IP20.

El parámetro “Rly” se refiere a su capacidad para controlar 10 salidas digitales a relés, lo cual en algunas aplicaciones gestionar cargas de potencia considerables sin el riesgo asociado a enviar señales de control muy potentes directamente. Además, los relés proporcionan un aislamiento eléctrico entre la carga y el controlador, mejorando la seguridad del sistema. [6]

En resumen, el SIMATIC S7- 1214C es un PLC potente y versátil, adecuado para diversas aplicaciones industriales, con características robustas y capacidades avanzadas de comunicación y procesamiento.[7] [8]

Característica	Descripción
<b>Tipo de CPU</b>	CPU 1214 C
<b>Memoria</b>	100 KB para el programa y datos, ampliable con tarjeta de memoria
<b>Entradas/Salidas Integradas</b>	14 entradas digitales (DI) y 10 salidas digitales (DO)
<b>Interfaces de Comunicación</b>	PROFINET integrado, 1 puerto RJ45, soporte para comunicación con módulos de expansión como RS485, RS232 y otros
<b>Reloj de Tiempo Real</b>	Incluido, con respaldo de batería para mantener la hora en caso de falla de energía

*Tabla 3.1:Tabla características SIMATIC S7-1214*



*Ilustración 3.3: SIMATIC S7-1214*

### 3.5. PANTALLA TÁCTIL KTP 700

La pantalla táctil KTP 700 es una interfaz HMI (Human Machine Interface) de Siemens, perteneciente a la gama SIMATIC HMI Basic. Proporciona un manejo sencillo en aplicaciones de automatización industrial, permitiendo una interacción operario-máquina de manera intuitiva y eficiente.

Está dotada de una resolución de 800x480 píxeles en sus 7 pulgadas de tamaño, proporcionando así tanto una visibilidad óptima como una operación precisa por parte

del usuario. Permite diversas funciones gráficas como gráficos, botones, barras de proceso o interruptores.

En términos de conectividad, la KTP 700 es compatible con distintas interfaces de comunicación, entre ellas TCP/IP, Modbus TCP y PROFINET, facilitando así su integración en gran variedad de sistemas de control. En concreto, cuenta con facilidades para la comunicación con el resto de los dispositivos pertenecientes a la familia de dispositivos S7. Su configuración y programación se realizan en el software propio de Siemens: TIA Portal. Además de herramientas para la programación de autómatas programables, este programa cuenta con herramientas para el diseño de interfaces HMI y la comunicación de variables entre el PLC y la misma.

Una de sus principales ventajas es su robustez en ambientes adversos, ya que está diseñada para soportar condiciones industriales adversas gracias a su carcasa con protección IP65 en la parte frontal. También está preparada para la creación de pantallas de alarma y registros de datos, lo cual facilita inmensamente el control y supervisión detallado de los procesos a gestionar. En caso de anomalía, notificaría al operador en tiempo real, reduciendo así su tiempo de respuesta. Respecto al registro de datos, permite almacenar y analizar información histórica para la posterior optimización de procesos y la aplicación de medidas preventivas. [9] [10]

<b>Característica</b>	<b>Descripción</b>
<b>Tamaño de Pantalla</b>	Pantalla táctil de 7 pulgadas con resolución de 800 x 480 píxeles, ofreciendo una claridad y visibilidad óptimas para la visualización de datos.
<b>Interfaz Táctil</b>	Pantalla táctil resistiva que permite una operación precisa incluso con guantes, ideal para entornos industriales.
<b>Conectividad</b>	Soporta interfaces de comunicación como PROFINET y MPI/PROFIBUS DP, facilitando su integración en diversas arquitecturas de control.
<b>Programación</b>	Configuración y programación a través del software TIA Portal, permitiendo el diseño de interfaces de usuario personalizadas y configuración de parámetros de comunicación.
<b>Robustez</b>	Diseñado para soportar condiciones industriales adversas con una carcasa que cumple con las normativas de protección IP65 en la parte frontal.
<b>Funcionalidades Avanzadas</b>	Permite la creación de pantallas de alarma, recetas y registros de datos, facilitando la supervisión y control detallado de los procesos industriales.

*Tabla 3.2: Tabla características KTP 700*



*Ilustración 3.4: KTP700*

### 3.6. IOT-2040

El IOT-2040 de Siemens es un dispositivo IoT (Internet de las Cosas) diseñado para aplicaciones industriales, proporcionando una solución flexible y compacta para la recolección y procesamiento de datos. Pertenece a la familia SIMATIC IOT y se caracteriza por su compatibilidad con varios protocolos de comunicación industrial, como Modbus TCP, lo cual facilita la integración en sistemas ya existentes y la gestión de datos de diversas fuentes, ya sea en la nube o en redes locales. Se trata de un elemento ideal para la implementación de estrategias de Industria 4.0.

En términos de conectividad, incluye dos puertos Ethernet, dos puertos USB y una ranura mini PCIe para módulos adicionales como Wi-Fi o 3G/4G, ideal para conectar sensores y actuadores a sistemas de control y gestión de datos en la nube. Soporta entornos de desarrollo y sistemas operativos como Linux, C/C++ y Node-RED, brindando a los desarrolladores la oportunidad de crear aplicaciones personalizadas. Está diseñado para entornos industriales adversos, operando en un rango de temperatura de 0°C a 50°C gracias a su carcasa.

El SIMATIC IoT2040 debe alimentarse con una fuente de alimentación de 24 V DC. Para ampliar sus capacidades, puede utilizarse junto con un módulo I/O (6ES7647-0KA01-0AA2) que proporciona cinco entradas digitales, dos entradas analógicas y dos salidas digitales, permitiendo así flexibilidad para conectar distintos actuadores o detectores. [11] [12]

Característica	Descripción
<b>Tensión de Alimentación</b>	Alimentación de 24 V DC, adecuada para aplicaciones industriales.
<b>Entradas y Salidas Digitales</b>	5 entradas digitales (tipo de tensión de entrada DC, señal "0" < 5 V DC, señal "1" > 12 V DC) y 2 salidas digitales de transistor con protección contra cortocircuitos.
<b>Entradas Analógicas</b>	2 entradas analógicas con rangos de entrada de 0 a 10 V y de 0 a 20 mA.



<b>Diseño y Montaje</b>	Diseño de tarjeta insertable y montaje en interfaz Arduino, facilitando la integración con otros componentes y módulos Arduino.
-------------------------	---

*Tabla 3.3: Características SIMATIC IoT12040*



*Ilustración 3.5. SIMATIC IoT12040*

## 3.7. COMPONENTES SOFTWARE

En esta sección se describirán los diversos componentes de software utilizados en el diseño e implementación del sistema de control domótico. Estos componentes son fundamentales para la programación, gestión y comunicación del sistema, permitiendo una integración eficiente y efectiva de los distintos dispositivos y funcionalidades.

### 3.7.1. TIA Portal

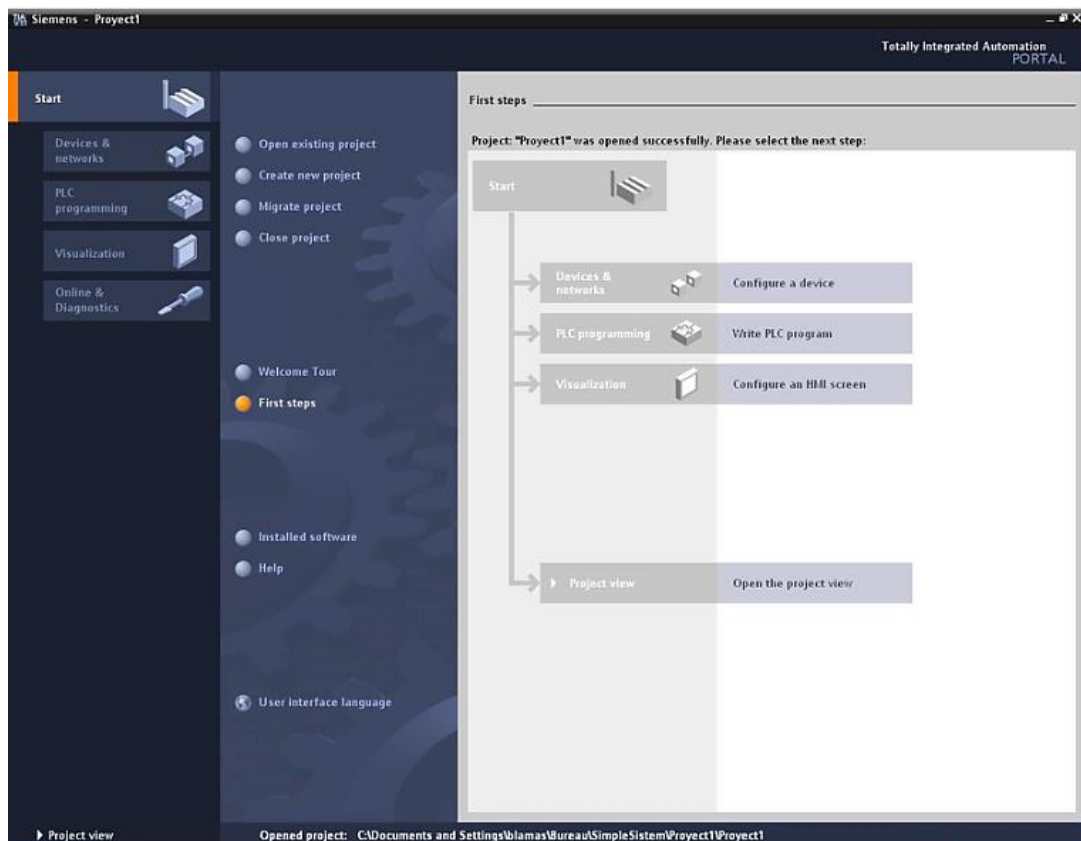
Para el desarrollo de la aplicación, se ha utilizado el software TIA Portal (Totally Integrated Automation Portal) en su versión V17. TIA Portal es una plataforma integral de Siemens que permite la programación y configuración de los autómatas programables (PLCs), interfaces hombre-máquina (HMI), y redes industriales. Esta herramienta destaca por su capacidad para integrar en un único entorno todos los aspectos de la automatización, facilitando el diseño, la puesta en marcha y el mantenimiento de sistemas complejos.





*Ilustración 3.6: Logo TIA Portal V17*

Para este proyecto se ha elegido la versión 17 principalmente por sus funcionalidades enfocadas a OPC UA. Además esta versión incluye mejoras respecto a versiones anteriores como son una interfaz de usuario mejorada, optimización del rendimiento (opción que hemos de deshabilitar para la comunicación de variables a Node-RED), integración de la última generación de hardware de Siemens y capacidades en el ámbito de la seguridad.



*Ilustración 3.7: Interfaz inicial TIA Portal*

Para la correcta ejecución de TIA Portal es necesario contar con una serie de aplicaciones complementarias que deben ser instaladas previamente. Estas herramientas adicionales facilitan la programación, configuración y simulación de los sistemas automatizados, asegurando un entorno de desarrollo integral y eficiente.



*Ilustración 3.8: Software necesario para TIA Portal*

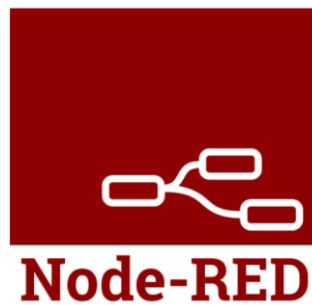
- Win CC: se usa para ejecutar aplicaciones HMI diseñadas en la propia aplicación, lo cual permite al operador visualizar y controlar los procesos en tiempo real.
- TIA Administrator: herramienta de administración implementada para la gestión de licencias, versiones de software y demás aspectos administrativos.
- Automation License Manager: se emplea para administrar las licencias del software de los productos de Siemens. Permite registrar, activar y administrar las licencias necesarias para utilizar las diversas aplicaciones dentro del entorno TIA Portal.
- S7-PLCSIM: simulador de PLC que permite emular el funcionamiento de los controladores S7 de Siemens sin necesidad de hardware físico. Esto permite verificar el correcto funcionamiento del sistema y detectar posibles errores en una etapa temprana del desarrollo.

El proceso de desarrollo de la aplicación en TIA Portal V17 se puede resumir en los siguientes pasos que se realizan iterativamente para conseguir finalmente el resultado deseado:

1. Creación del Proyecto: iniciar un nuevo proyecto en TIA Portal y definir los parámetros básicos como el nombre del proyecto y su ubicación de almacenamiento.
2. Configuración del Hardware: seleccionar y configurar los dispositivos de hardware como es el PLC y el HMI.
3. Programación del PLC: utilizar STEP 7 para crear y editar los bloques de programa que controlarán las operaciones del sistema automatizado.
4. Diseño de la HMI: diseñar las pantallas de interfaz de usuario que permitirán la interacción con el sistema.
5. Configuración de la Red: establecer la comunicación entre los distintos dispositivos utilizando las herramientas de configuración de red integradas en TIA Portal.
6. Simulación y Pruebas: utilizar PLCSIM para emular el funcionamiento del PLC y realizar pruebas preliminares antes de la implementación en el hardware real.
7. Descarga y Puesta en Marcha: transferir el programa al hardware real, realizar las pruebas finales y ajustar los parámetros necesarios para garantizar el correcto funcionamiento del sistema.

### 3.7.2. Node-RED

Es una plataforma de uso libre y gratuito desarrollada principalmente para el Internet de las Cosas (IoT) y está basada en el lenguaje JavaScript. Esta herramienta permite la integración y automatización de dispositivos y servicios mediante una interfaz de programación visual. Su principal ventaja radica en la simplicidad para conectar diversos dispositivos y servicios sin necesidad de conocimientos profundos en programación.



*Ilustración 3.9: Logo de Node-RED*

Node-RED permite la creación de flujos mediante la interconexión de nodos, los cuales son configurados según los requerimientos del usuario. Su programación se asemeja a la creación de un diagrama de flujo, donde los nodos se conectan entre sí para definir el comportamiento del sistema. Cada nodo tiene una función específica, como la recolección de datos, procesamiento de información, o envío de comandos a dispositivos. En este proyecto, se emplean los nodos S7 para la comunicación directa con el PLC SIMATIC S7- 1200 de Siemens, facilitando el intercambio de datos entre el sistema domótico y la plataforma Node-RED.

La utilización de nodos S7 en Node-RED permite obtener y enviar datos al PLC sin necesidad de intermediarios como un servidor externo como sería el caso de MQTT. Estos nodos están especialmente diseñados para interactuar con los controladores lógicos programables de Siemens, simplificando la integración y mejorando la eficiencia del sistema. De esta manera, podemos monitorizar y controlar diversos aspectos del piso piloto domótico (*E-Llar*) en tiempo real.

Además, Node-RED cuenta con un dashboard, una interfaz gráfica que permite al usuario visualizar y controlar el estado de ciertas variables del PLC. Está preinstalado en el dispositivo IOT2040, pero también puede ser instalada en otros sistemas operativos y otros dispositivos externo. Esto permite a los usuarios visualizar y controlar el estado del sistema de forma remota, independientemente de su ubicación, siempre y cuando tengan acceso a la red donde se encuentra el PLC.

### 3.7.3. Protocolo S7

El protocolo S7 es un protocolo de comunicación desarrollado por Siemens para sus controladores lógicos programables (PLC), específicamente para la serie SIMATIC S7. Este protocolo facilita la transmisión de datos entre los PLC y otros dispositivos de automatización industrial, permitiendo una comunicación eficiente y fiable en entornos

de control industrial. Su diseño se centra en la compatibilidad con la familia de PLC SIMATIC, ofreciendo un método estandarizado para la transferencia de datos.

El protocolo S7 funciona en diferentes tipos de redes, como Ethernet y PROFINET, lo que permite conectarlo fácilmente a distintos sistemas de red utilizados en la industria. La capacidad de operar sobre TCP/ IP facilita la conectividad con una amplia gama de dispositivos, desde sensores y actuadores hasta sistemas SCADA y HMI. Esta versatilidad hace que el protocolo S7 sea una elección popular en aplicaciones industriales que requieren una comunicación robusta y de alto rendimiento.

Una característica destacada del protocolo S7 es su capacidad para gestionar grandes volúmenes de datos en tiempo real. Esto es crucial en aplicaciones industriales donde la sincronización y la precisión son esenciales para el control de procesos. El protocolo permite la transmisión de datos cíclicos y acíclicos, adaptándose a diferentes necesidades de comunicación, ya sea para el control continuo de procesos o para la transmisión ocasional de datos de configuración y diagnóstico.

Además, el protocolo S7 incorpora mecanismos de seguridad y control de errores para asegurar la integridad y confiabilidad de los datos transmitidos. Estos mecanismos incluyen verificación de errores y gestión de retransmisiones en caso de fallos de comunicación. La capacidad de gestionar múltiples conexiones simultáneas y priorizar el tráfico de datos críticos asegura que los sistemas de control industrial funcionen de manera continua y eficiente, minimizando el riesgo de interrupciones en el proceso de producción. [13]

#### 3.7.4. SQLite

SQLite es una biblioteca de software utilizada para gestionar y almacenar datos de manera eficiente y segura en bases de datos.



*Ilustración 3.10 SQLite logo*

Una de las principales ventajas de SQLite es que no requiere un servidor separado para operar, lo que simplifica la arquitectura del sistema. Esto se traduce en una mayor facilidad de configuración y mantenimiento, ya que todos los datos se almacenan en un único archivo en el disco. Además, su naturaleza transaccional garantiza la integridad de los datos, incluso en casos de fallos inesperados del sistema.

SQLite soporta una amplia gama de funciones SQL estándar, lo que facilita la ejecución de consultas complejas y la manipulación de datos. Su compatibilidad con múltiples lenguajes de programación permite una fácil integración con Node-RED y otros componentes del sistema. Esto asegura una interoperabilidad fluida y una gestión de datos eficiente en todo el ecosistema domótico.

## 4. Revisión de componentes y recableado

### 4.1. INTRODUCCIÓN

A lo largo de este trabajo se requirió una revisión detallada de la instalación, precisando el recableado de las entradas y salidas, para solucionar errores y organizar las señales al nuevo controlador. Una vez documentado dicho proceso se procedió a estudiar, diseñar y programar en TIA Portal de Siemens las funcionales elegidas para este proyecto. Como mecanismo de depuración se empleó en todo momento la instalación física quedando completada la programación con el desarrollo de un conjunto de pantallas de explotación creadas también con TIA Portal, para manejo de las diferentes configuraciones y funcionalidades que se han establecido.

### 4.2. REVISIÓN Y RECABLEADO

En primer lugar, fue necesario revisar las conexiones y el cableado, ya que, al leer los informes del curso anterior, se encuentran algunas incongruencias, como tener listado el sensor de presencia de la habitación como una salida.

Regleta Blanca	Phoenix contact	Nº Cable	Nombre	PLC	Colores PLC
1	1	551	Pulsador Luz Salón	I0.0	Negro
2	2	545	Detector Presencia Aseo	I0.1	Marrón
3	3	543	Pulsador Luz Habitación	I0.2	Rojo
4	4	552	Termostato	I0.3	Naranja
5	5	541	Anemómetro	I0.4	Amarillo
6	6	538	Detector Presencia Cocina-Salón	I0.5	Verde
7	7	544	Detector Inundación	I0.6	Azul
8	8	550	Pulsador Luz Cocina	I0.7	Morado
9	9	554	Pulsador Subir Persiana	I1.0	Gris
10	10	553	Pulsador Bajar Persiana	I1.1	Blanco
11	11	547	Detector Luminosidad	I1.2	Blanco-Negro
12	12	549	Detector Lluvia	I1.3	Blanco-Marrón
13	13	548	Detector Ventana abierta	I1.4	Blanco-Rojo
14	14	537	Detector Humo Salón	I1.5	Blanco-Naranja
15	15	546	Detector Humo Habitación	-	Blanco-Amarillo
16	16	539	Pulsador Alarma	-	Blanco-Verde
17	17	542	Pulsador Luz Aseo	-	Blanco-Azul
18	18	540	Pulsador Exterior	-	Blanco-Morado
19	19	556	Pulsador Timbre	-	Blanco-Gris
20	20	555	Pulsador Jardín	-	Marrón-Negro

-	21	-	-	-	Marrón-Rojo
-	22	-	-	-	Marrón-Naranja
-	23	-	-	-	Marrón-Amarillo
-	24	-	-	-	Marrón-Verde
-	25	-	-	-	Marrón-Azul
-	26	-	-	-	Marrón-Morado

*Tabla 4.1: Entradas PLC*

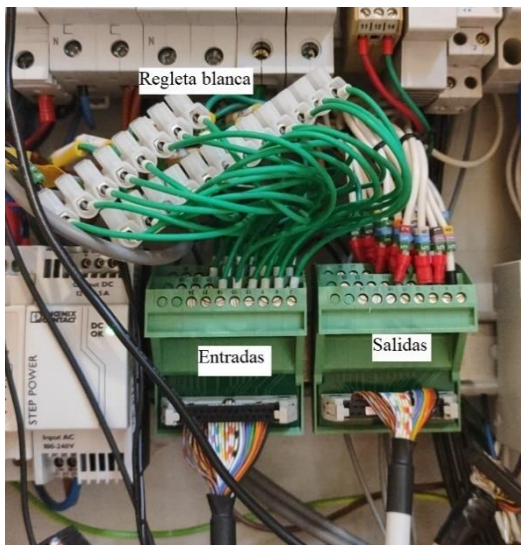
Relé	Phoenix contact	Nº Cable	Nombre	PLC	Colores PLC
1	1	576	Carga Cocina	Q0.5	Negro
2	2	575	Válvula Riego	-	-
3	3	574	EV agua	Q0.6	Rojo
4	4	573	EV gas	-	-
5	5	572	Luz Cocina	Q0.7	Amarillo
6	6	571	Luz Habitación	Q0.2	Verde
7	7	570	Carga Ventilador	Q0.4	Azul
8	8	569	???	-	-
9	9	568	Luz Salón	Q0.0	Gris
10	10	567	Carga Habitación	-	-
11	19	566	Luz Aseo	Q0.1	Blanco-Gris
12	17	565	Subir Persiana	Q1.0	Blanco-Azul
13	13	564	Bajar Persiana	Q1.1	Blanco-Rojo
14	14	563	Zumbador	-	-
15	15	562	Carga Calefactor	Q0.3	Blanco-Amarillo
16	16	561	Carga Salón		

*Tabla 4.2: Salidas PLC*

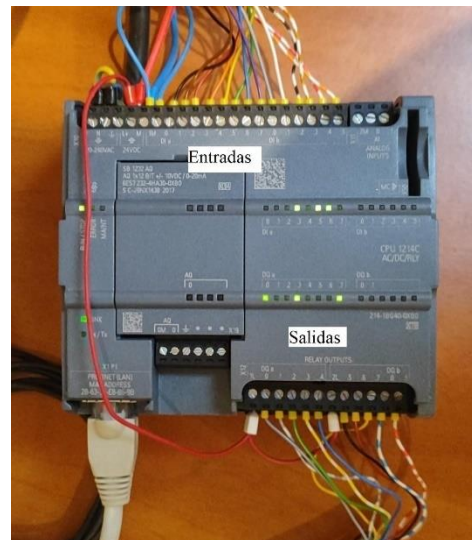
Para actualizar las entradas y salidas del PLC, se requirió cambiar las conexiones, no solo en el propio PLC sino en los conectores de Phoenix contact. Fue necesario también añadir una regleta extra, etiquetada en la *Tabla 4.1: Entradas PLC* como “regleta blanca”, para ser capaces de desenmarañar todos los cables.

La actuación sobre las salidas del controlador hacia el conjunto de relés con bobina a 24V DC que hacen funciones de preaccionamiento de los actuadores, todos ellos a 230V AC, resulto ser más sencilla ordenando simplemente en base a las numeraciones de forma consecutiva.

En la imagen se muestra los elementos principales comentados, que, si bien no tienen aún un acabado definitivo, sí que están razonablemente organizados y chequeados al completo. Con ello se ha sentado la base para una actuación definitiva y ha permitido avanzar con la programación de las funcionalidades del sistema.



*Ilustración 4.1: Conexiones Phoenix Contact*



*Ilustración 4.2: Conexiones PLC*



## 5. Programación del PLC

### 5.1. INTRODUCCIÓN

Esta sección es esencial para comprender cómo se ha diseñado e implementado el núcleo del sistema domótico, asegurando que cada componente interactúe de manera armoniosa y eficiente. Se detallará cómo se ha configurado y programado el PLC SIMATIC S7-1214 para gestionar las diversas funcionalidades del sistema, asegurando una operación eficiente, fiable y segura del piso piloto.

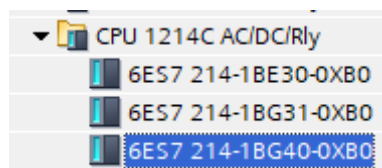
El contenido de este apartado incluye la configuración del hardware, la estructura del programa de control, la definición de los tipos de datos de usuario (UDT) y la organización de variables y bloques de datos (DB). Se describirán también las funciones (FC) y bloques funcionales (FB) que permiten el control específico de alarmas, calefacción, cargas, luces y persianas.

Para consultar el código completo acudir a los documentos TIA\_Portal\_1 y TIA\_Portal2\_UDT

### 5.2. CONFIGURACIÓN HARDWARE

Dado que portátil utilizado para la programación no tiene puerto Ethernet, se ha optado por una conexión inalámbrica mediante un router que fue instalado en el piso piloto. El router crea una red Wi-Fi llamada "e-llar", que permite conectarse directamente al PLC, a pesar de que no tiene conexión a Internet por temas relativos al administrador de la red de Uniovi que aún no han dado su conformidad.

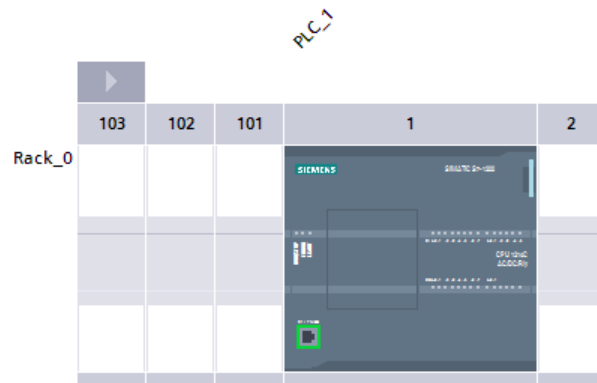
Una vez que el portátil se conecta a la misma red a la que está conectado el PLC por cable Ethernet al router, se debe seleccionar en el programa el tipo de CPU disponible. En este caso, se trata de un SIMATIC S7-1214 AC/DC Rly.



*Ilustración 5.1: Ajustes iniciales PLC*

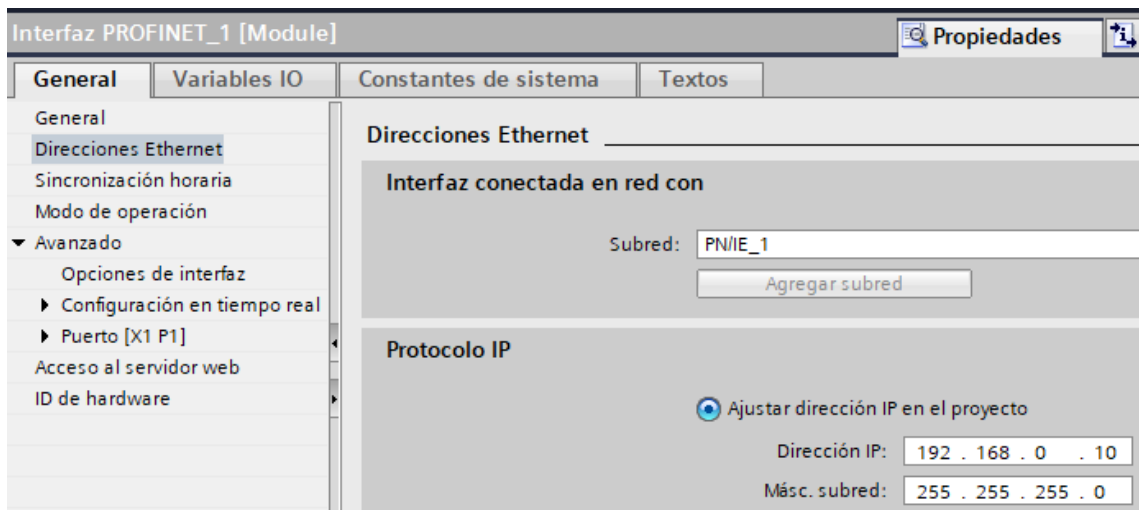
Hecho esto, se ha de hacer coincidir la dirección IP del programa con la del PLC real desde "Configuración de dispositivos". Desde ahí se puede configurar los dispositivos físicos conectados al PLC.





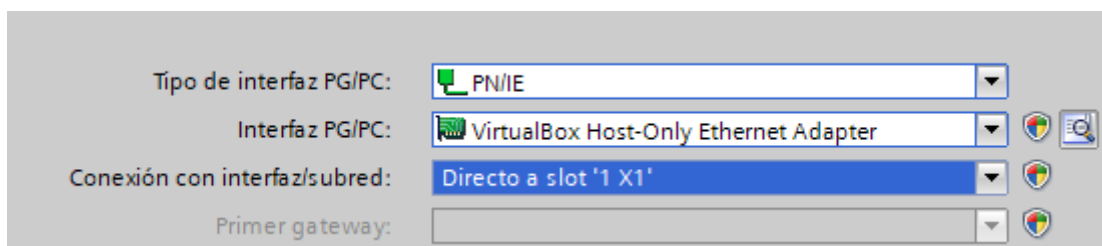
*Ilustración 5.2: Configuración de dispositivos*

Si se selecciona el módulo de la interfaz PROFINET\_1, desde la pestaña direcciones ethernet, te permite ajustar la dirección IP del proyecto. La IP en cuestión es: 192.168.0.10



*Ilustración 5.3: Interfaz PROFINET*

Dado este punto, ya se puede establecer conexión online con el PLC desde el icono naranja de la barra de herramientas. Una vez seleccionados los parámetros indicados en *Ilustración 5.3*, al pulsar iniciar búsqueda aparece el dispositivo etiquetado como PLC\_1.



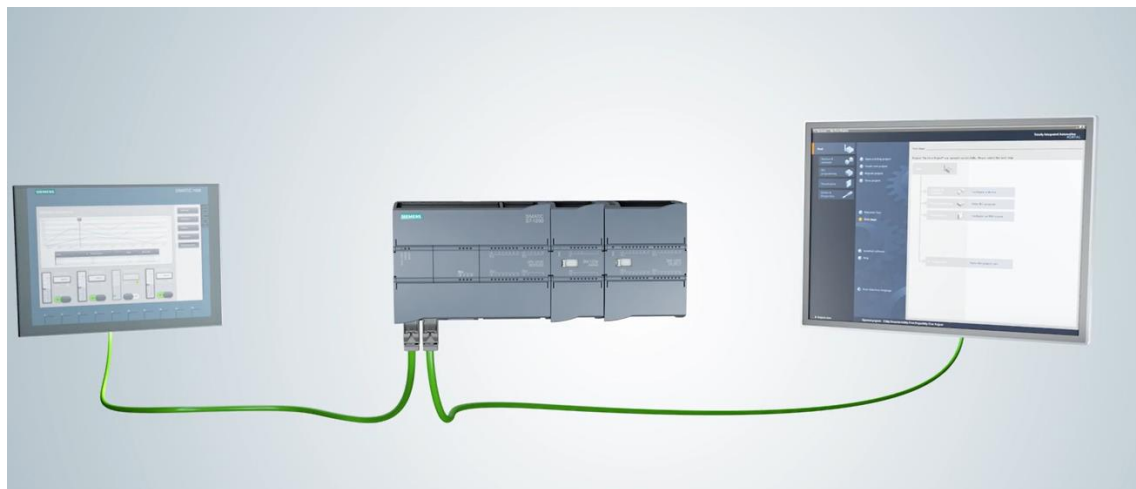
*Ilustración 5.4: Establecer conexión online*

En la imagen siguiente se muestra la vista de redes de TIA Portal con las direcciones correspondientes. Hay que tener en cuenta que el ordenador desde donde se realiza la programación, a través de la red Wi-Fi establecida por el router tendrá una dirección IP dentro del rango correspondiente. La simulación de la pantalla se ejecuta en el propio ordenador sin necesidad de HMI físico.



*Ilustración 5.5: Vista de redes*

A continuación, se presenta una representación física de la configuración de red descrita anteriormente. Ilustra cómo estos componentes se conectan físicamente en un entorno real. Si el PLC no dispone de dos puertos ethernet, como es el caso, se puede usar el mismo Router que tendrá varias entradas o un switch adicional.



*Ilustración 5.6: HMI-PLC-Tia Portal*

### 5.3. ESTRUCTURA DEL PROGRAMA DE CONTROL

La programación del autómat programable SIMATIC S7-1200 se realiza mediante la herramienta TIA Portal v17 de Siemens cuya parte relativa a la programación se identifica por Step7. La compatibilidad con el estándar IEC 61131-3 es considerable en algunos aspectos, sobre todo en cuanto a los lenguajes de programación utilizados para el desarrollo del código en los diferentes módulos de programación.

Sin embargo, como aspectos diferenciales respecto del estándar se destaca:

- Los mecanismos para la definición de tareas están basados en los Módulos de Organización (OBs) que no existen en la norma. Los OBs tienen un comportamiento distinto en cuanto a la ejecución según el número correspondiente. Esencialmente se usará solo el OB1 donde se incluyen instrucciones que se ejecutarán a cada ciclo de scan del PLC.
- La declaración de variables con direccionamiento de memoria explícito, dispone, además de las entradas (%I), las salidas (%Q) y las marcas (%M), de la estructuración en los llamados Bloques de Datos (DBs). Estos serán necesarios tanto para la instanciación de Bloques Funcionales como para definir variables estructuradas.
- En Step7, para realizar una llamada a cualquier bloque funcional debe referenciarse (y crearse) un Bloque de Datos de Instancia, con el número correspondiente, que no es preciso especificar cuándo se programa en el estándar IEC 61131-3.

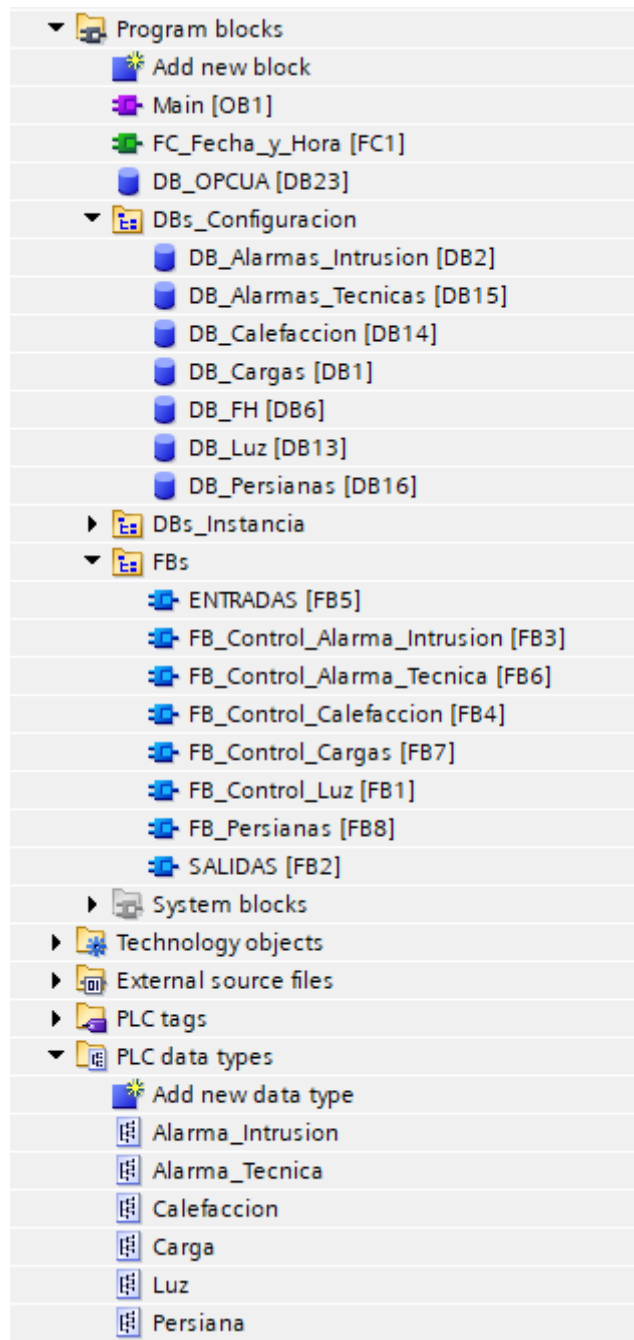
El diseño del programa de control se ha basado en un conjunto de consideraciones que posteriormente se han desarrollado y serán detalladas en este capítulo. Dichas consideraciones son las siguientes:

1. Se han creado un conjunto de tipos de datos de usuario (UDTs) para poder organizar los parámetros e información necesaria para cada funcionalidad domótica implementada. Esto permite una mejor estructuración y acceso a las variables. Estos tipos de datos definidos se han creado en la carpeta **PLC data types** en el árbol del proyecto de TIA Portal.
2. A partir de los tipos de datos se han creado las variables correspondientes de cada tipo así como el número adecuado para cubrir las prestaciones de control de la vivienda. Estas variables es preciso declararlas en bloques de datos (DBs), que se han organizado en la carpeta **DBs\_Configuracion** del árbol del proyecto.
3. Para la implementación del código asociado a cada funcionalidad se ha preferido crear los correspondientes bloques funcionales (FBs). Esto es muy ventajoso para encapsular el comportamiento e instanciar cuantas veces sean preciso aplicar. Por ejemplo, el FB\_Control\_Luz se llamará para cada circuito de iluminación se haya establecido para la vivienda. Se han organizado en la carpeta **FBs** del árbol del proyecto. La instanciación requiere de un DB de instancia que mantiene los datos de entrada (Input), salida (Output) y variables internas (Static) de la llamada; estos bloques de datos se encuentran en la carpeta **DBs\_Instancia**.
4. Para hacer posible la depuración del programa de aplicación de forma más cómoda se copian los estados y valores de las señales físicas conectadas a las entradas de periferia del PLC a las variables creadas según la estructuración definida anteriormente. Esto se hace en el bloque funcional ENTRADAS (FB5). Así es posible escribir (o forzar) los valores sin tener que hacerlo en la memoria de entradas. Solamente se emplean las variables con direccionamiento “%I” en este bloque funcional; por otra parte los nombres

de estas variables emplean el sufijo “I\_” para su identificación (por ejemplo, *I\_Det\_Pres\_Aseo*).

5. En el bloque funcional SALIDAS (FB2) se asigna a las variables definidas con direccionamiento a la memoria de salidas el valor correspondiente al parámetro de la variable estructurada que es la que se maneja en el resto de los módulos de programación. Esto permite tener un único punto de asignación de la memoria de salidas, que es crítico, pues son las que finalmente acaban gestionando el funcionamiento del actuador. Las variables que direccionan la periferia de salidas “%Q” se nombran con el sufijo “Q\_” para una más sencilla identificación (por ejemplo, *Q\_Subir\_Pers*).
6. Para la implementación de otros elementos auxiliares que son precisos para cubrir las prestaciones del sistema, se pueden utilizar los recursos disponibles para la programación. Por ejemplo, se requiere trabajar con la fecha y –sobre todo- con la hora actual de modo que se ha desarrollado la función *FC\_Fecha\_y\_Hora* (FC1) de modo que pueda determinarse si se está en el intervalo horario establecido para la activación de alguna funcionalidad.
7. Finalmente, para la llamada secuencial de los módulos de programa requeridos (FCs y FBs) se ha establecido el **Bloque de Organización (OB1)** cuya ejecución cíclica garantiza las prestaciones de control en tiempo real del sistema domótico basado en PLC.

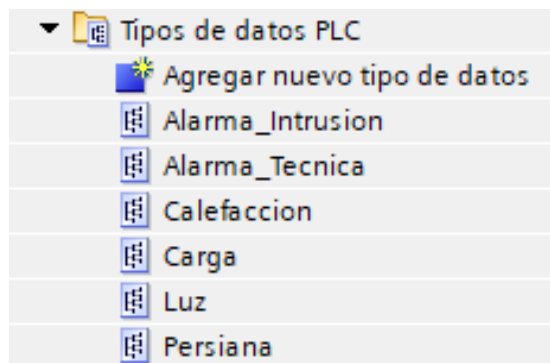
En la siguiente imagen se puede apreciar el despliegue de los elementos indicados en los puntos anteriores y que serán desarrollados con detalle en los siguientes apartados.



*Ilustración 5.7: Bloques del programa*

## 5.4. TIPOS DE DATOS DE USUARIO (UDT)

Puesto que existen varios elementos a controlar en cada una de las funcionalidades; por ejemplo, varios circuitos de iluminación, varias cargas, tipos de alarma técnica, etc... se ha llevado a cabo la creación de un tipo de dato de usuario (UDT, User Data Types) para cada funcionalidad, como se muestra en *Ilustración 5.8*, de forma que aglutine los parámetros principales para su definición y programación.



*Ilustración 5.8: Tipos de datos de usuario creados*

Los UDT son definiciones de datos personalizadas y por lo tanto específicas para cada aplicación. Combinan distintos tipos de datos básicos en una sola estructura facilitando así la gestión y manipulación de estos. De esta manera permiten replicar las mismas variables para distintas instancias, proporcionando así una forma personalizada y flexible de estructurar datos complejos.

A continuación, se detallan algunos aspectos de cada uno de los tipos de datos de usuario creados. Como se apreciará en cada tipo de dato, lo cual se trasladará a las variables creadas de dicho tipo de dato (tags), se deben/pueden configurar los siguientes elementos asociados a cada columna:

- Nombre (Name): Es el identificador de la variable (tag) según las reglas sintácticas correspondiente.
- Tipo de datos (Data type): Para cada uno de los parámetros definidos (tags). Pueden ser los que se incluyen por defecto o a su vez creados por el usuario.
- Valor predeterminado (Default value): Es el valor que tendrá dicho tag cuando el PLC se reinicie. Es un valor opcional y si no se indica tomar el valor predefinido, por ejemplo “false” si es de tipo bool.
- Accesible desde HMI/OPC UA/Web API: Permitirá ser leído desde un sistema HMI, mediante protocolo OPC UA o una aplicación Web durante la ejecución (modo runtime). Si no se marca esta opción no estará disponible en este componente.
- Writable desde HMI/OPC UA/Web API: Permitirá ser escrito desde un sistema HMI, mediante protocolo OPC UA o una aplicación Web durante la ejecución (modo runtime). Si no se marca esta opción no estará disponible en este componente.
- Setpoint: Los valores de consigna son los valores que probablemente tendrán que ajustarse con precisión durante la puesta en servicio. Tras la puesta en servicio, los valores de estas etiquetas pueden transferirse al programa offline como valores de inicio y almacenarse allí.
- Visible in HMI engineering: Indica si se permite monitorización de esta variable para la depuración del programa.

- Comentario (Comment): Es un texto opcional de interés para la documentación del proyecto.

Puesto que algunos de estos elementos van a mantenerse en todos los tipos de datos que han sido implementados se obviarán en algunas de las imágenes posteriores que son presentadas.

### 5.4.1. UDT Alarma\_Intrusion

Respecto a las alarmas de intrusión, se considera la alarma armada según el parámetro booleano "Armada" de tipo "Bool", una vez que recibe una señal positiva del parámetro "Activacion" y en función del valor del parámetro "Clave" de tipo "UInt". Una vez armada, depende del valor del booleano "Det\_Pres" para que, pasado un retardo "T\_Ret\_ON" de tipo "Time", se dispare, es decir, que se active el parámetro "Disparada".

Alarma_Intrusion				
	Nombre	Tipo de datos	Valor predet.	Comentario
1	Det_Pres	Bool	false	Estado detector de presencia (tipo NC)
2	T_Ret_ON	Time	T#155	Retardo armado alarma o disparo (en s)
3	Activacion	Bool	false	Estado alarma intrusión activada
4	Armada	Bool	false	Estado alarma intrusión armada
5	Clave	UInt	1234	Clave actídes de la alarma
6	Disparada	Bool	false	Estado alarma intrusión disparada

Ilustración 5.9: UDT\_Alarma\_Intrusion

### 5.4.2. UDT Alarma\_Tecnicas

El control de alarmas técnicas se inicia con el estado de una señal booleana "Activacion". En función del estado del booleano "Actuador", una vez pasado un tiempo marcado por el parámetro "Retardo" de tipo "Time", se activará el parámetro "Disparada". Esto, a su vez, actúa sobre el parámetro booleano "Aviso" y el string "Mensaje".

Alarma_Tecnica				
	Name	Data type	Default value	Comment
1	Detector	Bool	false	Detector de inundación
2	Activacion	Bool	false	Activación de la alarma
3	Retardo	Time	T#15	
4	Disparada	Bool	false	Estado de disparo de la alarma
5	Actuador	Bool	false	
6	Aviso	Bool	false	
7	Mensaje	String	'Peligro'	Mensaje de aviso cuando se produce el disparo

Ilustración 5.10: UDT Alarma Tecnica

### 5.4.3. UDT Calefaccion

La calefacción se regirá por el valor del booleano "Salida", que estará activado o no en función del estado de "Termostato". Según el estado del parámetro "Auto", se tendrán en cuenta o no los valores del booleano "Ventana\_Abierta" y el horario marcado por "Hora\_ON" y "Hora\_OFF" de tipo "Time\_Of\_Day".

Calefaccion					
	Name	Data type	Default value	Comment	Setpoint
1	Hora_ON	Time_Of_Day	TOD#10:00:00	Hora de inicio del horario	<input type="checkbox"/>
2	Hora_OFF	Time_Of_Day	TOD#22:00:00	Hora de finalización del horario	<input type="checkbox"/>
3	Auto	Bool	false	Modo automático	<input type="checkbox"/>
4	Ventana_Abierta	Bool	false	Sensor de ventana abierta	<input type="checkbox"/>
5	Termostato	Bool	false	señal del termostato	<input type="checkbox"/>
6	Salida	Bool	false	Estado de la calefacción	<input type="checkbox"/>

*Ilustración 5.11: UDT Calefaccion*

#### 5.4.4. UDT Carga

Para el control de las cargas, se modifica el estado del parámetro booleano "Salida" en función de la señal booleana "Orden\_ON\_OFF". Se plantea un sistema con dos modos de funcionamiento: automático y manual, según el estado del parámetro "Auto":

- Si se selecciona el modo manual, el usuario podrá activar y desactivar las cargas desde los correspondientes botones disponibles en el "HMI" que modifica el valor del parámetro "HMI\_Carga".
- En caso de seleccionarse el modo automático, el estado de las cargas se determina en función del estado de los correspondientes detectores ("Det1" y "Det2" de tipo "Bool") y el horario seleccionado ("Hora\_ON" y "Hora\_OFF" de tipo "Time\_Of\_Day"). Si se cumplen las condiciones correspondientes, las cargas se activarán durante un tiempo dado por "T\_Ret\_OFF" de tipo "Time".

Carga				
	Name		Default value	Comment
1	Orden_ON_OFF		false	variable auxiliar
2	Salida		false	Estado de la carga
3	T_Ret_OFF		T#0ms	duración activacion en automatico
4	Det1		false	detector del q depende modo Auto
5	Det2		false	detector del q depende modo Auto
6	Auto		true	modo automático
7	Hora_ON		TOD#10:00:00	inicio horario activación
8	Hora_OFF		TOD#23:00:00	final horario activación
9	HMI_Carga		false	Pulsador HMI

*Ilustración 5.12:UDT Carga*

#### 5.4.5. UDT Luz

Para el control de la iluminación, se actúa sobre el parámetro booleano "Salida". Su estado dependerá del estado de la variable "Auto":

- En caso de ser TRUE, se tendrán en cuenta "Det\_Pres" y "Det\_Lum" de tipo booleano, además del horario marcado por "Hora\_ON" y "Hora\_OFF" de tipo "Time\_Of\_Day". Si se activa "Salida", se mantendrá activa durante un tiempo determinado por el parámetro "T\_Ret\_OFF" de tipo "Time".



- Por otro lado, si es FALSE, estará condicionada por las variables "Puls" y "HMI\_Puls", ambas de tipo booleano.

Luz				
		Name	Default value	Comment
1		Puls	false	pulsador
2		Salida	false	estado de la luz
3		T_Ret_OFF	T#0ms	duracion activación en automático
4		Det_Pres	false	estado detector de presencia
5		Det_Lum	false	estado detector de luminosidad
6		Auto	true	modo automático
7		Hora_ON	TOD#00:00:00	inicio horario de activación en automático
8		Hora_OFF	TOD#00:00:00	final horario de activación en automático
9		HMI_Puls	false	pulsador del HMI

Ilustración 5.13: UDT Luz

### 5.4.6. UDT Persiana

Por último, para la gestión de las persianas se cuenta también con un parámetro "Auto". En caso de ser TRUE, el estado de la persiana depende del estado de "Det\_Viento", "Det\_Lluvia" y "Det\_Lum" de tipo booleano y el horario seleccionado entre "Hora\_Subir" y "Hora\_Bajar" de tipo "Time\_Of\_Day". Sin embargo, en caso de ser FALSE, el estado de las persianas se ve condicionado por el estado de los booleanos "Puls\_Subir", "HMI\_Subir", "Puls\_Bajar" y "HMI\_Bajar".

Como resultado, se envía una señal u otra determinada por los parámetros "Subir" y "Bajar", ambos de tipo booleano.

Persiana					
		Name	Default value	Comment	Setpoint
1		Puls_Subir	false	pulsador fisico subir	<input type="checkbox"/>
2		Det_Lluvia	false	estado detector lluvia	<input type="checkbox"/>
3		Subir	false	orden de subida	<input type="checkbox"/>
4		Bajar	false	orden de bajada	<input type="checkbox"/>
5		Puls_Bajar	false	pulsador fisico bajar	<input type="checkbox"/>
6		Det_Lum	false	estado detector luminosidad	<input type="checkbox"/>
7		Hora_Subir	TOD#00:00:00	hora de subida en automático	<input type="checkbox"/>
8		Hora_Bajar	TOD#00:00:00	hora de bajada en automático	<input type="checkbox"/>
9		Auto	false	pulsadores horarios o detectores	<input type="checkbox"/>
10		Det_Viento	false	estado detector viento	<input type="checkbox"/>
11		HMI_Subir	false	pulsador HMI subir	<input type="checkbox"/>
12		HMI_Bajar	false	pulsador HMI bajar	<input type="checkbox"/>
13		HMI_persiana	0	variable auxiliar HMI	<input type="checkbox"/>

Ilustración 5.14: UDT Persianas

## 5.5. ORGANIZACIÓN DE VARIABLES Y BLOQUES DE DATOS (DB)

Los tipos de dato de usuario permiten crear variables que en Step7 hay que declarar en Bloques de Datos (DBs), y que se han organizado en este caso en la carpeta **DBs\_Configuración** del proyecto.



*Ilustración 5.15: Data blocks*

Los DBs son áreas de memoria donde se almacenan los datos en el PLC para posteriormente ser utilizados por el programa de control. En estos bloques se guardan variables y estructuras de datos para posteriormente ser manipuladas y accedidas durante la ejecución –en este trabajo- del programa principal (Main, OB1).

En los DBs de configuración se declaran los valores iniciales correspondientes a cada una de las variables, aplicados específicamente a la instancia que corresponda. En este proyecto cada uno de estos bloques de datos está asociado a una de las funcionalidades del piso piloto y análogamente también a un tipo de dato de usuario. Gracias a la implementación de tipos de datos de usuario en este proyecto, al crear una nueva variable basta con etiquetarla como uno de los UDT mencionados anteriormente para que incluya todos sus correspondientes parámetros internos, en lugar de crear cada uno de ellos como variables separadas. Esto facilita mucho el manejo de la información, con una estructuración organizada y sin necesidad de usar memorias de marcas (%M) con su correspondiente direccionamiento, que puede ser susceptible de solaparse, y evitando tener que crear un nombre único para cada una de las variables que se pretenda utilizar.

En cuanto a la sintaxis seguida, con esta estrategia se puede acceder a cualquier parámetro de las variables por ejemplo de “Toldo”, simplemente usando los nombres del DB de configuración, la variable general “Toldo” y el campo específico al que se quiera acceder:

- “*DB\_Persianas.Toldo.Det\_Lluvia*”, es el detector correspondiente

Y, para las alarmas de intrusión, sirva otro ejemplo:

- “*DB\_Alarma\_Intrusion.Cocina-Salón.Disparada*”, que identifica este estado de dicha alarma.

A continuación, se detallan los bloques de datos mencionados, sobre todo en cuanto a las variables definidas que como se ha comentado corresponden con el conjunto de funcionalidades implementadas en el proyecto.

Con cada DB se ha incluido una imagen recortando las columnas menos relevantes como son “Retain”, “Accessible from HMI/OPC UA/Web API”, “Visible in HMI engineering”, “Setpoint” o “Writable from HMI/OPC UA /Web API”.

También se ha desplegado solo una de las variables correspondientes a los distintos UDTs ya que se repiten en cada una de las variables de ese mismo datatype cuenta con los mismos Tags. Cada uno de ellos detallados en la sección [Tipos de datos de usuario \(udt\)](#)

### 5.5.1. DB\_Alarmas\_Intrusión

Se consideran dos variables correspondientes a las dos zonas que se consideran en el piso piloto: Aseo-Habitación y Cocina-Salón. Ambas son del tipo de dato definido "Alarma\_Intrusion", con todos los parámetros. En caso de aplicarse el programa a una vivienda mayor, sería muy sencillo añadir las nuevas zonas ya que bastaría con etiquetar las nuevas zonas como tipo de dato "Alarma\_Intrusion" ([5.4.1](#)) para que incluya todos los parámetros necesarios.

Además, también cuenta con una variable "Clave\_IN" de tipo "UInt" para almacenar la cifra que introduce el usuario como clave para activar la alarma en caso de coincidir con el parámetro "Clave". Al inicializar el programa, este dato tiene valor 0. Como se puede observar en [Ilustración 5.16](#), el parámetro "Clave" toma el valor 1234.

DB_Cargas				
	Name	Offset	Start value	Comment
1	▼ Static			
2	▼ Cocina	0.0		Variable de zona
3	■ Orden_ON_OFF	0.0	false	variable auxiliar
4	■ Salida	0.1	false	Estado de la carga
5	■ T_Ret_OFF	2.0	T#100000s	duración activación en automatico
6	■ Det1	6.0	false	detector del q depende modo Auto
7	■ Det2	6.1	false	detector del q depende modo Auto
8	■ Auto	6.2	true	modo automático
9	■ Hora_ON	8.0	TOD#10:00:00	inicio horario activación
10	■ Hora_OFF	12.0	TOD#22:00:00	final horario activación
11	■ HMI_Carga	16.0	false	Pulsador HMI
12	▶ Ventilador	18.0		
13	▶ Calefactor	36.0		
14	■ Auto	54.0	false	Modo automático general

Ilustración 5.16: DB\_Alarmas\_Intrusion

### 5.5.2. DB\_Alarmas\_Tecnicas

En el caso de las alarmas técnicas se consideran cuatro tipos distintos: Inundación, Humo, Gas y Caída\_Tensión. Cada una corresponde a un posible tipo de fallo técnico que

se podría detectar en la vivienda. Al estar declaradas como dato de tipo "Alarma\_Tecnica" (5.4.2), cada una cuenta con todos los parámetros necesarios para la gestión.

Debido a las limitaciones impuestas por el hardware del piso piloto e-llar, ya sea por la falta de sensores o por falta de entradas y/o salidas suficientes para el control de tantos dispositivos, no todas las funcionalidades referidas a las alarmas técnicas están disponibles. Por ejemplo, no es posible el control de la alarma de humo debido a que los sensores propios del piso fueron sustituidos por sensores conectados a la red general de la escuela politécnica de Gijón por motivos de seguridad. En caso de contar con instalaciones con mayores prestaciones, bastaría con asignar las variables ya creadas a los correspondientes sensores y actuadores.

DB_Alarmas_Tecnicas			
	Name	Data type	Start value
1	Static		
2	Inundacion	"Alarma_Tecnica"	
3	Detector	Bool	false
4	Activacion	Bool	false
5	Retardo	Time	T# 10000s
6	Disparada	Bool	false
7	Actuador	Bool	false
8	Aviso	Bool	false
9	Mensaje	String	'Peligro'
10	Humo	"Alarma_Tecnica"	
11	Gas	"Alarma_Tecnica"	
12	Caida_Tension	"Alarma_Tecnica"	

Ilustración 5.17: DB\_Alarmas\_Tecnicas

### 5.5.3. DB\_Calefaccion

Respecto a la calefacción, se consideran dos zonas distintas: Cocina-Salón y Habitación, correspondientes a las estancias principales del piso piloto. Ambas son variables de tipo "Calefaccion", lo cual lleva implícito todos los parámetros relacionados. Por lo tanto, en caso de desarrollarse una aplicación para una vivienda mayor, sería muy sencillo añadir las nuevas zonas declarándolas como tipo de dato "Calefaccion" (5.4.3) para que cuente con dichos parámetros específicos.

Además, debido a las limitaciones del laboratorio solo se cuenta con un termostato y una salida de calefacción por lo que en el DB se incluye los tags *Termostato* y *Salida*, ambos booleanos, para que actúen como variables genéricas.

DB_Calefaccion				
		Name	Data type	Start value
1	[-]	▼ Static		
2	[-]	▼ Cocina-Salon	"Calefaccion"	
3	[-]	■ Hora_ON	Time_Of_Day	TOD#10:00:00
4	[-]	■ Hora_OFF	Time_Of_Day	TOD#22:00:00
5	[-]	■ Auto	Bool	false
6	[-]	■ Ventana_Abierta	Bool	false
7	[-]	■ Termostato	Bool	false
8	[-]	■ Salida	Bool	false
9		<Add new>		
10	[-]	▶ Habitacion	"Calefaccion"	

Ilustración 5.18: DB\_Calefaccion

### 5.5.4. DB\_Cargas

Para la automatización de las instalaciones del piso piloto se plantea el control de tres cargas distintas: Cocina, Ventilador y Calefactor. Cada una de ella se asocia a una variable declarada como tipo de dato "Carga" (5.4.4). A pesar de ser tan distintas y tener funcionalidades tan dispares, en el DB se tiene la oportunidad de asignarles los parámetros correspondientes a cada una.

Además, se consideró más conveniente un modo "Auto" genérico en lugar de que cada carga funcione en automático o manual independientemente de las demás cargas. Esto se gestiona mediante la variable "Auto", de tipo booleano.

DB_Cargas					
		Name	Data type	Offset	Start value
1	[-]	▼ Static			
2	[-]	▼ Cocina	"Carga"	0.0	
3	[-]	■ Orden_ON_OFF	Bool	0.0	false
4	[-]	■ Salida	Bool	0.1	false
5	[-]	■ T_Ret_OFF	Time	2.0	T#100000s
6	[-]	■ Det1	Bool	6.0	false
7	[-]	■ Det2	Bool	6.1	false
8	[-]	■ Auto	Bool	6.2	true
9	[-]	■ Hora_ON	Time_Of_Day	8.0	TOD#10:00:00
10	[-]	■ Hora_OFF	Time_Of_Day	12.0	TOD#22:00:00
11	[-]	■ HMI_Carga	Bool	16.0	false
12	[-]	▶ Ventilador	"Carga"	18.0	
13	[-]	▶ Calefactor	"Carga"	36.0	
14	[-]	■ Auto	Bool	54.0	false

Ilustración 5.19: DB\_Cargas

### 5.5.5. DB\_FH

Con el objetivo de almacenar los datos referentes a la fecha y hora actual, se ha creado la variable "Fecha\_Hora" del tipo "DTL". Este tipo de dato es propio de TIA Portal e incluye ya implícitos parámetros referentes a datos temporales como el año, la hora y los nanosegundos, cada uno de ellos de tipo "USInt". Como se puede observar en la imagen [DB\\_FH](#), está ya inicializado a él uno de enero de 1970 a las 00:00:00, considerado el inicio de los tiempos en informática.

Por otra parte, para almacenar los datos útiles en este programa, se crean las variables "Fecha" (de tipo "Date") y "Hora" (de tipo "Time\_Of\_Day").

DB_FH			
	Name	Data type	Start value
1	Static		
2	Fecha_Hora	DTL	DTL#1970-01-01-4
3	YEAR	UInt	1970
4	MONTH	USInt	1
5	DAY	USInt	1
6	WEEKDAY	USInt	5
7	HOUR	USInt	0
8	MINUTE	USInt	0
9	SECOND	USInt	0
10	NANOSECOND	UDInt	0
11	Fecha	Date	D#1990-01-01
12	Hora	Time_Of_Day	TOD#00:00:00

Ilustración 5.20: DB\_FH

### 5.5.6. DB\_Luz

Respecto al control de la iluminación, se consideran cuatro zonas distintas: Cocina, Salón, Aseo y Habitación, correspondientes a las estancias principales de la vivienda. Todas están declaradas como variables de tipo "Luz", lo cual lleva implícito todos los parámetros relacionados con la gestión de la iluminación. Por lo tanto, en caso de aplicarse el programa a una vivienda mayor, sería muy sencillo añadir las nuevas zonas ya que bastaría con etiquetar las nuevas zonas como tipo de dato "Luz" (5.4.5) para que cuente con dichos parámetros.

Al igual que con el DB\_Cargas (5.5.4), se considera más conveniente considerar un solo modo "Auto" que, según su valor permitan ser manejados de forma conjunta.

DB_Luz					
		Name	Data type	Offset	Start value
1	[-]	▼ Static			
2	[-]	■ Prueba	Bool	0.0	false
3	[-]	▼ Cocina	"Luz"	2.0	
4	[-]	■ Puls	Bool	2.0	false
5	[-]	■ Salida	Bool	2.1	false
6	[-]	■ T_Ret_OFF	Time	4.0	T#5M
7	[-]	■ Det_Pres	Bool	8.0	false
8	[-]	■ Det_Lum	Bool	8.1	false
9	[-]	■ Auto	Bool	8.2	false
10	[-]	■ Hora_ON	Time_Of_Day	10.0	TOD#10:00:00
11	[-]	■ Hora_OFF	Time_Of_Day	14.0	TOD#22:00:00
12	[-]	■ HMI_Puls	Bool	18.0	false
13	[-]	▶ Salon	"Luz"	20.0	
14	[-]	▶ Habitacion	"Luz"	38.0	
15	[-]	▶ Aseo	"Luz"	56.0	
16	[-]	■ Auto	Bool	74.0	false

Ilustración 5.21: DB\_Luz

### 5.5.7. DB\_Persianas

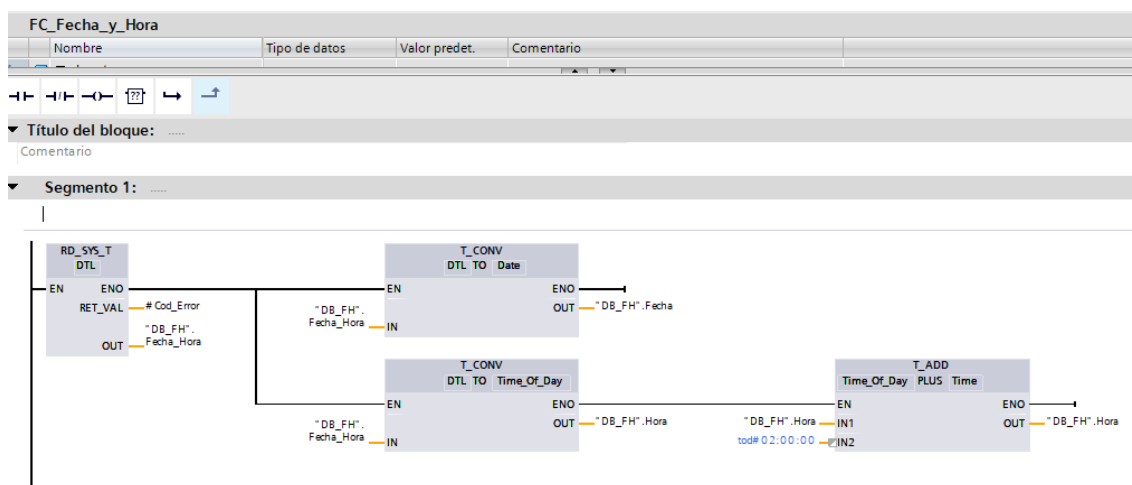
Por último, en el caso de las persianas se considera un único elemento de tipo "Persiana" denominado Toldo. Si se contara con algún elemento más de este tipo, sería sencillo añadirlos al DB ya que con la creación del UDT "Persiana", se crearían automáticamente los parámetros necesarios para el control de cada una de ellas.

DB_Persianas					
		Name	Data type	Offset	Start value
1	[-]	▼ Static			
2	[-]	▼ Toldo	"Persiana"		
3	[-]	■ Puls_Subir	Bool		false
4	[-]	■ Det_Lluvia	Bool		false
5	[-]	■ Subir	Bool		false
6	[-]	■ Bajar	Bool		false
7	[-]	■ Puls_Bajar	Bool		false
8	[-]	■ Det_Lum	Bool		false
9	[-]	■ Hora_Subir	Time_Of_Day		TOD#10:00:00
10	[-]	■ Hora_Bajar	Time_Of_Day		TOD#21:00:00
11	[-]	■ Auto	Bool		false
12	[-]	■ Det_Viento	Bool		false
13	[-]	■ HMI_Subir	Bool		false
14	[-]	■ HMI_Bajar	Bool		false
15	[-]	■ HMI_persiana	DInt		0

Ilustración 5.22: DB\_Persianas

## 5.6. FUNCIONES (FC)

La programación en TIA Portal, y también en el estándar IEC 61131-3 dispone unidades de organización de programa (POUs) denominados funciones que en TIA Portal se pueden desarrollar bajo el nombre de “FC”. Las funciones ejecutan secuencias específicas de código y devuelven un resultado, sin retener datos entre llamadas. Se utilizan en los casos en los que no se requiere mantener registro de los datos obtenidos por la función como es el caso de la obtención de la fecha y la hora, único que se ha creado en este trabajo y cuyo código en lenguaje FBD (FUP en TIA Portal) se muestra en la Ilustración siguiente.

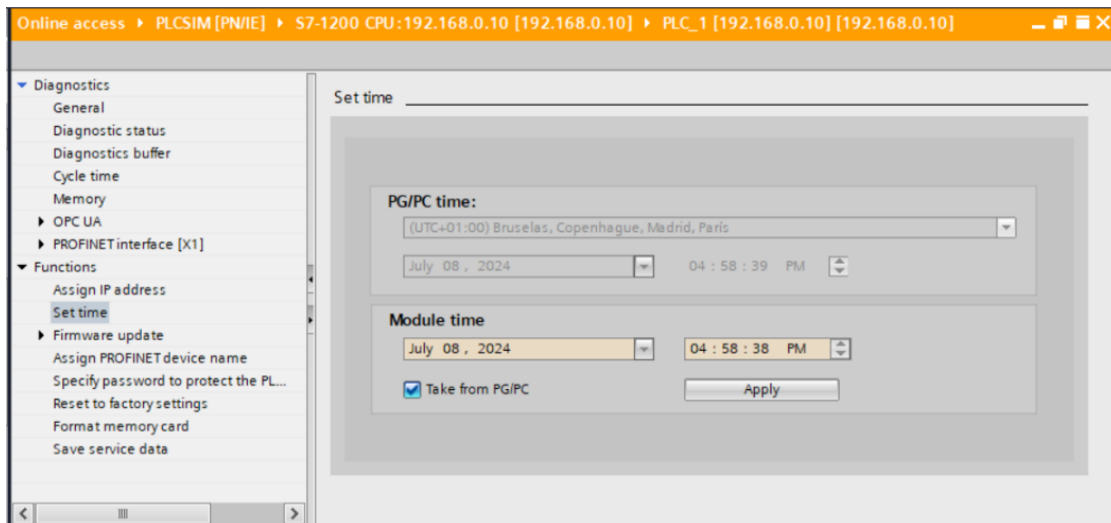


*Ilustración 5.23: FC Fecha\_y\_Hora*

En realidad, en el código implementado se usan un conjunto de funciones Step7, que pone a disposición el fabricante, sin que éstas formen parte del estándar IEC. La primera **RD\_SYS\_TDTL** lee la fecha y hora proporcionada por el reloj de tiempo real disponible en el PLC; después el resultado, que se guarda en “*DB\_FH*”.*Fecha\_Hora* se descompone con las funciones **T\_COV** en la fecha y la hora respectivamente. La última función del segmento permite añadir las dos horas correspondiente al ajuste según el uso horario, obteniendo en la variable “*DB\_FH*”.*Hora* el valor de la hora del día que se precisa para el control por horarios en algunas funcionalidades domóticas.

Para todo ello, es preciso tener previamente ajustada la fecha y hora de la CPU a las actuales lo cual se puede realizar a través de las funciones especiales de configuración en línea con el PLC (veren la [Ilustración 5.24](#) a continuación). Suele ser habitual tomar los datos del PC donde está instalado TIA Portal aunque puede hacerse manualmente.





*Ilustración 5.24: Configuración fecha y hora actual*

## 5.7. BLOQUES DE FUNCIÓN (FB)

En este proyecto, cada tipo de dato corresponde a una de las funcionalidades del piso piloto, y cada una de ellas también cuenta con su propio FB (Bloque de Función) que es posible instanciar para los elementos de control disponibles según las características de la vivienda.

Los bloques de función son códigos de control que permiten encapsular y reutilizar código, facilitando la creación de programas modulares y eficientes. De esta manera, una misma función se puede aplicar a cada una de las instancias concretas del programa, como puede ser en el caso de las luces en cada una de las habitaciones, a diferencia de las FC que no retienen datos entre llamadas. Gracias a esta forma de trabajo, se consigue un programa principal (Main, OB1) sencillo y eficiente.

Cada FB lleva asociado una estructura de datos específica denominada Instance Data Block (IDB), donde se almacenan los datos locales correspondientes al bloque de función. Esto permite la reutilización de los bloques de funciones con diferentes conjuntos de datos. Dentro del propio FB, se trabaja con datos del tipo correspondiente a la funcionalidad concreta a utilizar.

El conjunto de bloques funcionales, siguiendo esta estrategia, y que se han programado corresponden a todas las funcionalidades indicadas anteriormente y que aparecen en la carpeta FBs del árbol del proyecto (*Ilustración 5.7*) utilizando, en general, el sufijo “FB\_” y detallándose en la imagen siguiente.



*Ilustración 5.25: Árbol FB*

Los bloques de control trabajan con una variable al tipo de dato correspondiente a su funcionalidad. De esta manera en el programa Main solo es necesario asignar una variable del tipo de dato que corresponda en lugar de introducir en el bloque cada una de las variables por separado.

Los FBs también pueden contener variables internas que se utilizan exclusivamente dentro del propio FB. Estas variables internas permiten el almacenamiento temporal de datos y el control de estados específicos. Por ejemplo, en el "**FB\_Control\_Alarma\_Intrusion**" se puede usar la variable interna "*Clave\_Correcta*" para verificar la autenticidad de una clave ingresada. De manera similar, en el "**FB\_Control\_Cargas**", la variable interna "*Carga\_OFF*" puede utilizarse para gestionar el estado de apagado de una carga.

### **5.7.1. ENTRADAS (FB5)**

En este bloque de función se realizan las asignaciones de las variables asociadas a la memoria entradas físicas del PLC a las variables internas del programa, dentro de las estructuras de datos creadas al efecto para cada funcionalidad. Se consideró más conveniente realizarlo en lenguaje de texto estructurado ST (SCL en TIA Portal) por una cuestión de comodidad.

ENTRADAS					
IF...	CASE... OF...	FOR... TO DO...	WHILE... DO...	(*...*)	REGION
1	"DB_Luz".Salon.Puls :=	"I_Puls_Salon";			// I0.0
2	"DB_Alarmas_Intrusion".Aseo-Habitacion.Det_Pres :=	"I_Det_Pres_Aseo";			// I0.1
3	"DB_Luz".Aseo.Det_Pres :=	"I_Det_Pres_Aseo";			
4	"DB_Luz".Habitacion.Puls :=	"I_Puls_Hab";			// I0.2
5	"DB_Calefaccion".Habitacion.Termostato :=	"I_Term";			// I0.3
6	"DB_Calefaccion"."Cocina-Salon".Termostato :=	"I_Term";			
7	"DB_Persianas".Toldo.Det_Viento :=	"I_Anem_MAL";			// I0.4
8	"DB_Luz".Cocina.Det_Pres :=	"I_Det_Pres_Cocina-Salon";			// I0.5
9	"DB_Luz".Salon.Det_Pres :=	"I_Det_Pres_Cocina-Salon";			
10	"DB_Alarmas_Intrusion"."Cocina-Salon".Det_Pres :=	"I_Det_Pres_Cocina-Salon";			
11	"DB_Alarmas_Tecnicas".Inundacion.Detector:=	"I_Det_Inun";			//I0.6
12	"DB_Luz".Cocina.Puls :=	"I_Puls_Coc";			// I0.7
13	"DB_Persianas".Toldo.Puls_Subir :=	"I_Puls_Sub_Pers";			// I1.0
14	"DB_Persianas".Toldo.Puls_Bajar :=	"I_Puls_Baj_Pers";			// I1.1
15	"DB_Persianas".Toldo.Det_Lum :=	"I_Det_Lum_MAL";			// I1.2
16	"DB_Luz".Habitacion.Det_Lum :=	"I_Det_Lum_MAL";			
17	"DB_Persianas".Toldo.Det_Lluvia :=	"I_Det_Lluv";			// I1.3
18	"DB_Calefaccion".Habitacion.Ventana_Abierta :=	"I_Det_Ventana_MAL";			// I1.4
19	"DB_Alarmas_Tecnicas".Humo.Detector :=	"I_Det_Humo_Salon_MAL";			// I1.5

Ilustración 5.26: ENTRADAS

Nótese que cada entrada del PLC puede corresponder a varias variables del programa, como:

- "DB\_Luz".Cocina.Det\_Pres
- "DB\_Luz".Salon.Det\_Pres
- "DB\_Alarmas\_Intrusion"."Cocina-Salon".Det\_Pres

que corresponden todas a la entrada I0.5 del PLC: "I\_Det\_Pres\_Cocina-Salon".

Esta configuración depende en cada caso de la disponibilidad de entradas de periferia del controlador y la organización de variables que se quiera establecer.

### 5.7.2. FB\_Control\_Alarma\_Intrusion (FB3)

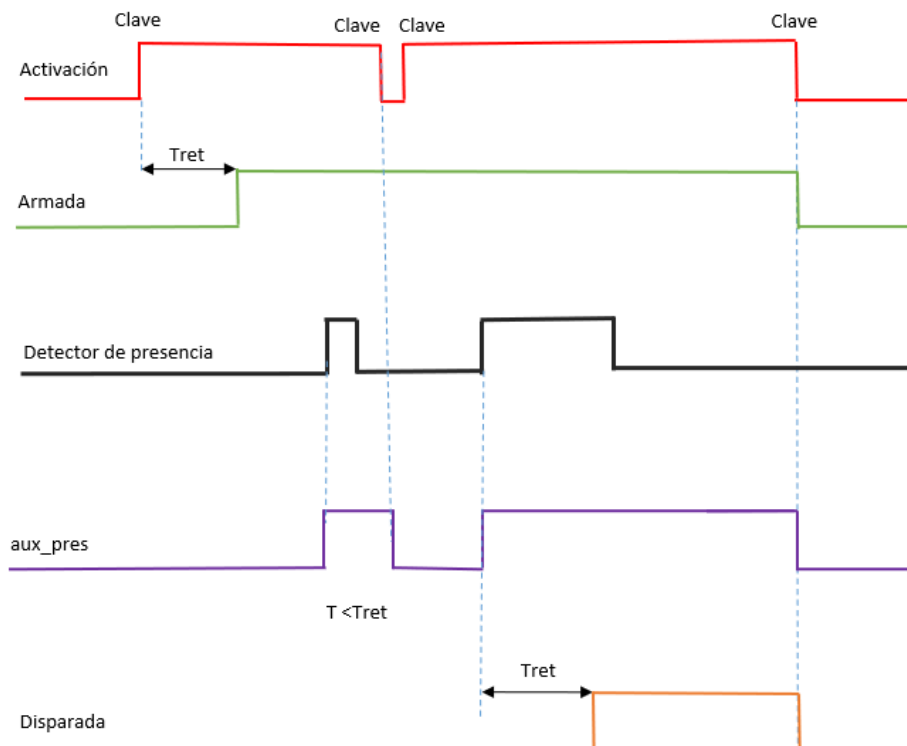
En este bloque funcional se trabaja con la variable "Alarma\_Intrusion\_a\_controlar", que es del tipo "Alarma\_Intrusion". Para acceder a cada uno de sus parámetros basta con seguir la sintaxis: "Alarma\_Intrusion\_a\_controlar.Parametro". Por ejemplo, para el detector de presencia, se accedería a la variable como "Alarma\_Intrusion\_a\_controlar.Det\_Pres".

La siguiente ilustración contiene las variables definidas en el bloque de función FB3.

FB_Control_Alarma_Intrusion				
	Name	Data type	Default value	Comment
1	Input			
2	Clave_IN	UInt	0	Calve introducida
3	Output			
4	InOut			
5	Alarma_Intrusion_a_controlar	*Alarma_intrusi...		Variable de zona para alarma intrusión
6	Static			
7	Clave_Correcta	Bool	false	Toma valor TRUE si la clave es correcta
8	Ins_TON1	TON_TIME		Temporización tras clave correcta
9	Ins_TON2	TON_TIME		Temporización tras detección de presencia
10	Temp			
11	aux_pres	Bool		Mantiene el evento de detección de presencia

*Ilustración 5.27: Variables FB\_Control\_Alarma\_Intrusion*

En el cronograma de muestra la evolución y relación entre las diferentes variables que conforman este algoritmo.



*Ilustración 5.28: Cronograma alarma intrusión*

Y en lo que, respecto al código desarrollado, primero, se compara el valor introducido por el usuario en el HMI con el valor de la clave correcta. En caso de coincidir, la variable "Clave\_Correcta" toma valor TRUE. Junto con la señal de activación enviada desde un interruptor en el HMI por el usuario, se activa una temporización que armará la alarma después de un retardo determinado.

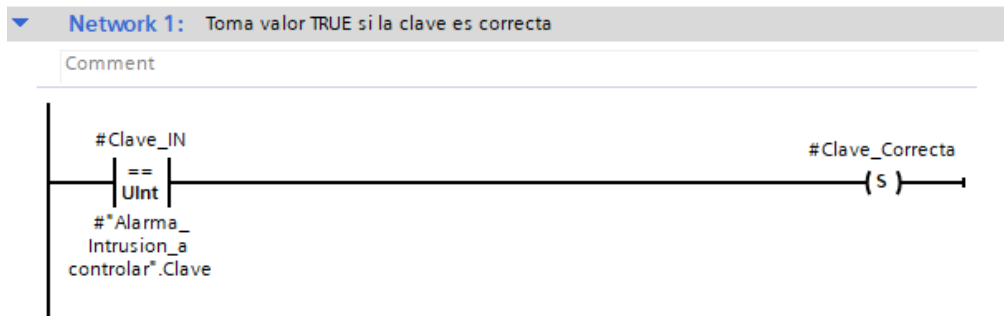


Ilustración 5.29: Network 1 FB\_Control\_Alarma\_Intrusion

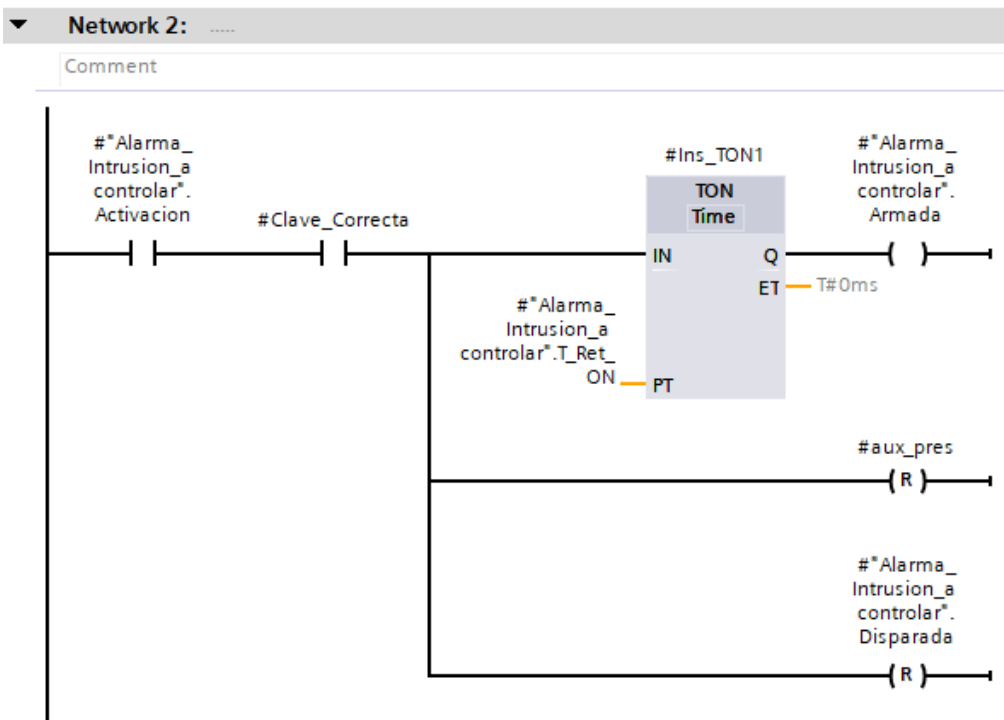


Ilustración 5.30: Network 2 FB\_Control\_Alarma\_Intrusion

Una vez armada la alarma, si se detecta presencia, se activaría la señal "aux\_pres". Al igual que muchos otros detectores relacionados con alarmas, "Det\_Pres" funciona con lógica negativa, por lo tanto, utiliza una consulta al estado de señal "0". Esto garantiza que, ante una posible rotura de hilo, por ejemplo, se genere una "falsa" alarma aunque, al menos, se permita estar alertado de la avería y solucionar el problema.

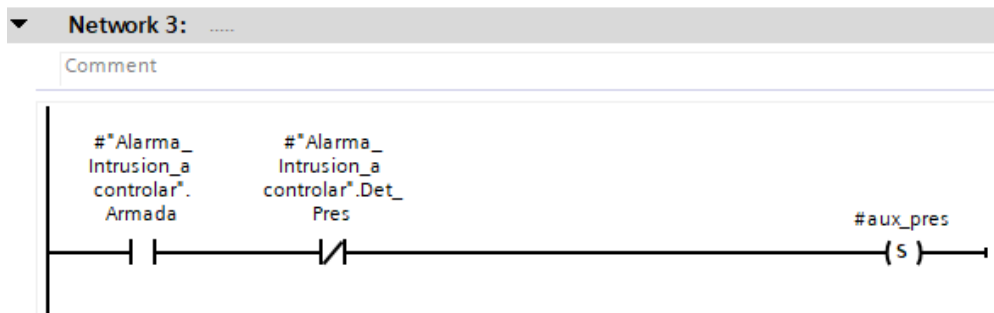


Ilustración 5.31: Network 3 FB\_Control\_Alarma\_Intrusion

La variable "aux\_pres" desencadena dos procesos: por una parte, se resetea el valor de "Clave\_IN" para que la persona que ha hecho saltar el detector de movimiento tenga que volver a introducir el código para desactivarla. Por otra parte, inicia una cuenta atrás que, pasado un tiempo de retardo, dispara la alarma. Para resetear los valores de "Disparada" y "aux\_pres", desactivando así la alarma, es necesario volver a introducir la clave correcta.

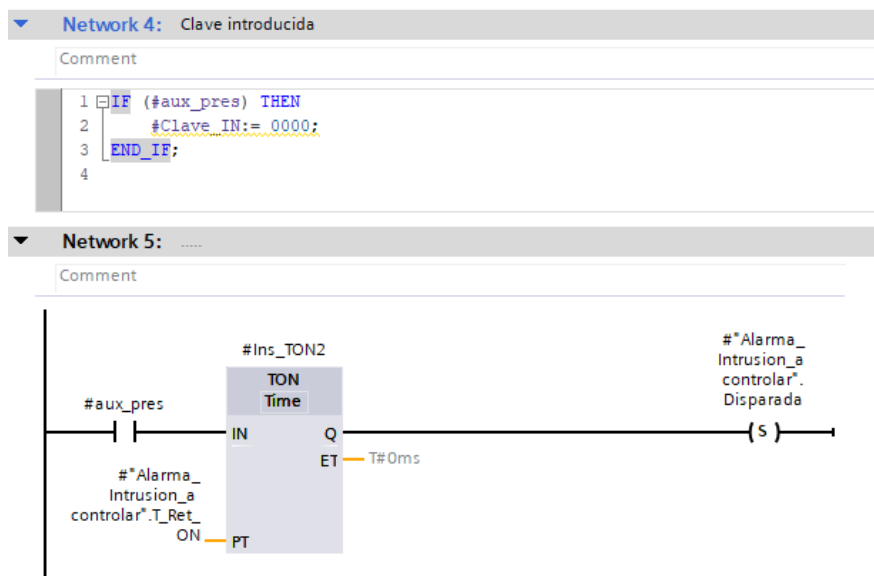


Ilustración 5.32: Network 4 y 5 FB\_Control\_Alarma\_Intrusion

### 5.7.3. FB\_Control\_Alarma\_Tecnica (FB6)

En el caso de las alarmas técnicas se trabaja con la variable "Alarma\_Tecnica\_a\_controlar", que es del tipo "Alarma\_Tecnica". Por lo tanto, para acceder a cada uno de sus parámetros basta con seguir la sintaxis: "Alarma\_Intrusion\_a\_controlar.Parametro". Por ejemplo, para el actuador, se accedería a la variable como "Alarma\_Intrusion\_a\_controlar.Actuador".

FB_Control_Alarma_Tecnica				
	Name	Data type	Default value	Comment
1	Input			
2	Output			
3	InOut			
4	Alarma_tecnica_a_co...	"Alarma_Tecnica"		Variable de alarma tecnica
5	Static			
6	Inst_TON	TON_TIME		Retardo de activación
7	Ins_R_TRIG	Bool	false	Detector de Flanco Ascendente
8	Temp			
9	Constant			

Ilustración 5.33: Variables FB\_Control\_Alarma\_Tecnica

En primer lugar, el usuario debe enviar una señal de activación de las alarmas técnicas desde el interruptor correspondiente en el HMI. En ese caso, si se detecta un cambio de estado en el detector correspondiente a la funcionalidad específica a la que se esté aplicando este bloque, se pasa a alarma "Disparada" después de un retardo dado por el parámetro "Retardo" que se utiliza para evitar posibles falsas alarmas. Una vez disparada la alarma, se reseteará la variable "Actuador" (lógica negativa).

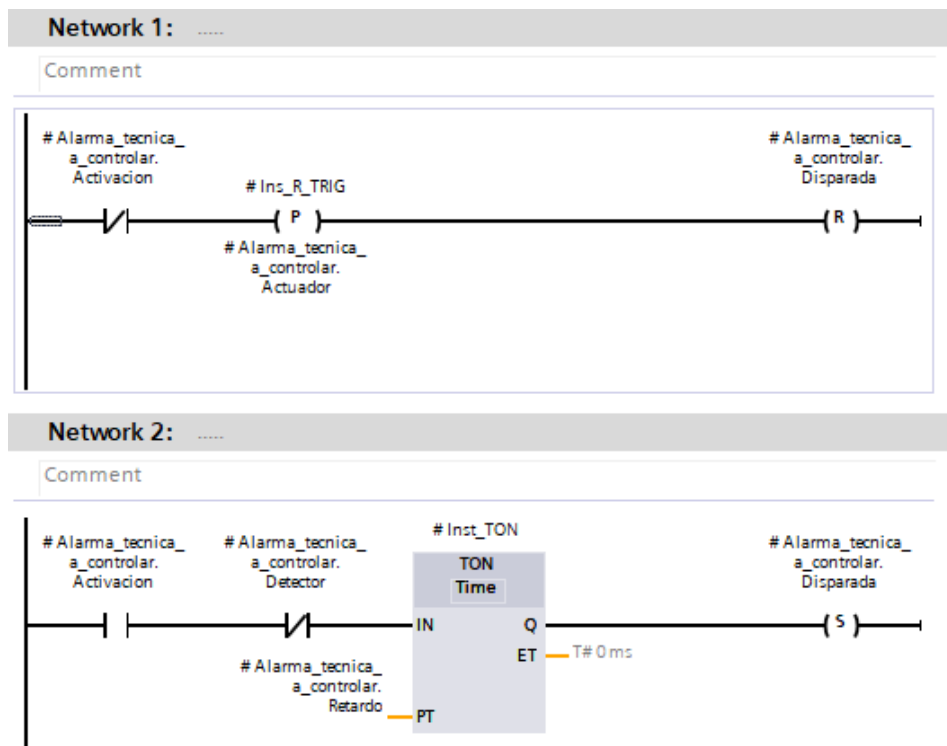


Ilustración 5.34: Network 1 y 2 FB\_Control\_Alarma\_Tecnica

Una vez solucionado el fallo técnico, el usuario debe desactivar manualmente el actuador volviendo a activar la alarma. De la misma manera, se reseteará el valor de la variable "Disparada", reiniciando así el bloque completo.

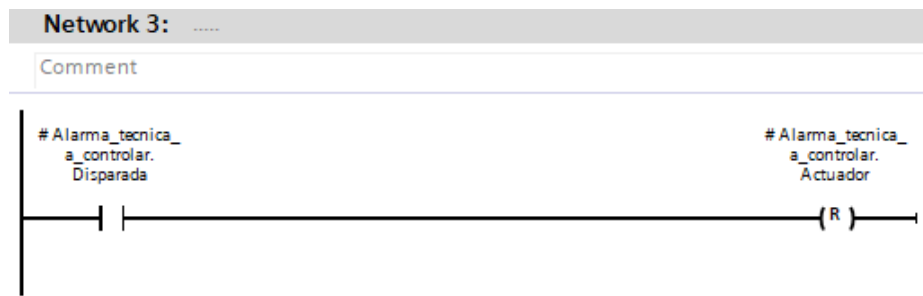


Ilustración 5.35: Network 3 FB\_Control\_Alarma\_Tecnica

A modo de ejemplo, en el caso de la alarma de inundación del piso piloto, el sistema consiste en un sensor de inundación que tiene conectadas dos sondas, una para la cocina y otra para el aseo. Si alguna de las sondas detecta inundación, el sensor desactiva la salida que hace que la entrada al controlador pase de TRUE a FALSE; tras el retardo correspondiente, se dispararía la alarma, lo cual cortarían la entrada de agua (cerrando la electroválvula o parando la bomba como en el piso piloto) hasta que el usuario decida que la avería está solucionada y vuelva a activar la alarma.

#### 5.7.4. FB\_Control\_Calefaccion (FB4)

En el bloque funcional destinado al control de la calefacción, se trabaja con la variable "Calefaccion\_a\_controlar", de tipo "Calefaccion". Para acceder a cada uno de sus parámetros, se utiliza la sintaxis: "Calefaccion\_a\_controlar.Parametro". Por ejemplo, para acceder a la hora de encendido, se puede usar la variable "Calefaccion\_a\_controlar.Hora\_ON".

FB_Control_Calefaccion				
	Name	Data type	Default value	Comment
1	Input			
2	<Add new>			
3	Output			
4	<Add new>			
5	InOut			
6	Calefaccion_a_control...	"Calefaccion"		Variable de calefacción
7	Static			
8	Hora_correcta	Bool	true	TRUE si la hora está dentro del horario
9	Temp			
10	Constant			

Ilustración 5.36: Variables FB\_Control\_Calefaccion

Analizando cada uno de los segmentos, en primer lugar, se comprueba que la hora en la que se ejecuta el programa está dentro del horario establecido para el funcionamiento de la calefacción. De ser así se activaría la variable "Hora\_correcta". Notar que se emplea la variable "DB\_FH".Hora que contiene la hora actual proporcionada por la función FC\_Fecha\_y\_Hora.



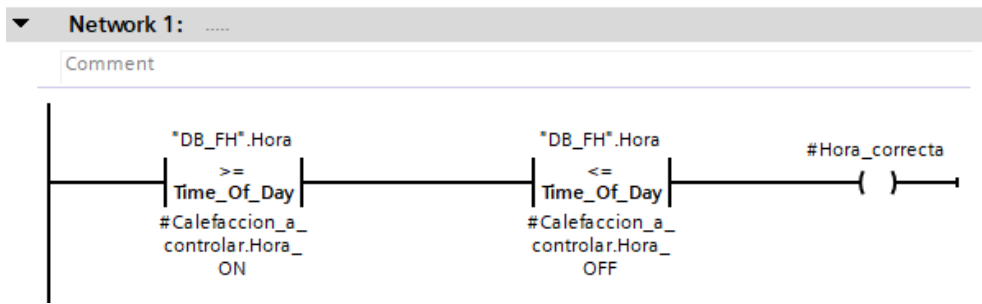


Ilustración 5.37: Network 1 FB\_Control\_Calefaccion

En el siguiente segmento se encuentran dos ramas en paralelo; en caso de estar activa la variable "Auto", se requiere que la variable "Hora\_correcta" también esté activada y que "Ventana\_Abierta" no lo esté. Además, el termostato debe mandar una señal positiva para activar la variable "Salida". Siguiendo la otra rama, en caso de que la variable "Auto" esté desactivada (modo manual), solo es necesario que el termostato tenga una señal positiva para activar "Salida".

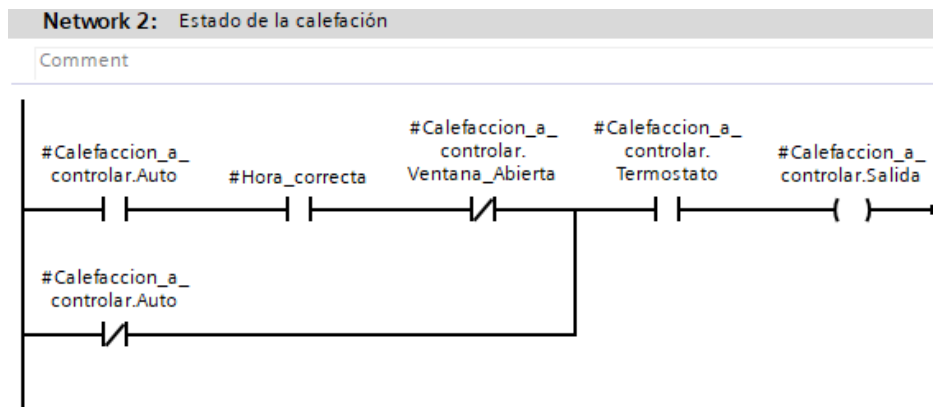


Ilustración 5.38: Network 2 FB\_Control\_Calefaccion

### 5.7.5. FB\_Control\_Cargas (FB7)

En el caso del control de cargas, se emplea la variable "Carga\_a\_controlar", de tipo "Carga". Para acceder a cada uno de sus parámetros, se utiliza la sintaxis: "Carga\_a\_controlar.Parametro". Por ejemplo, para la orden de encendido/apagado, se accedería a la variable como "Carga\_a\_controlar.Orden\_ON\_OFF".

La relación de variable declaradas en el bloque funcional FB7 se puede apreciar en la siguiente ilustración:

FB_Control_Cargas				
	Name	Data type	Default value	Comment
1	Input			
2	Output			
3	InOut			
4	Carga_a_controlar	*Carga*		Variable de carga
5	Static			
6	Hora_correcta	Bool	false	TRUE si está en el horario correcto
7	Ins_Enciende_Apaga	Bool	false	Instancia RS Sde la salida de cargas
8	Inst_R_Trig_1	R_TRIG		Detector de Flanco Ascendente
9	Flanco_Puls	Bool	false	Flanco del pulsador
10	Carga_OFF_manual	Bool	false	
11	Carga_OFF_auto	Bool	false	
12	Carga_ON_manual	Bool	false	
13	Carga_ON_auto	Bool	false	
14	aux_auto_ON	Bool	false	
15	aux_auto_OFF	Bool	false	
16	Ins_SR_Auto	Bool	false	Instancia RS modo manual de cargas
17	Ins_Timer_Cargas	TOF_TIME		Instancia timer
18	Ins_TON_Cargas	TON_TIME		Temporizador de Retardo a la Conexión
19	Temp			
20	Constant			

Ilustración 5.39: Variables FB\_Control\_Cargas

En el primer segmento se utiliza un R\_Trig para generar un flanco positivo (Flanco\_Puls) cada vez que el usuario seleccione uno de los interruptores de las cargas en el HMI o reciba una "Orden\_ON\_OFF".

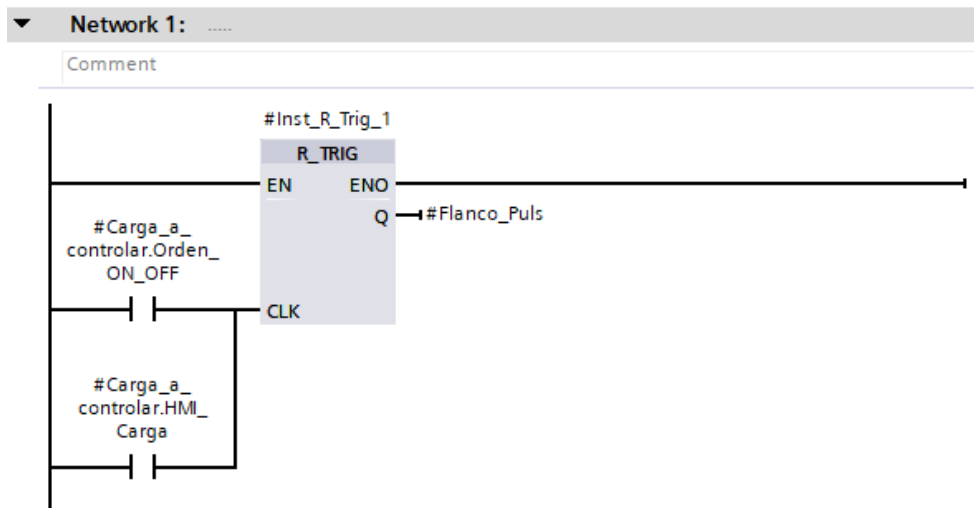
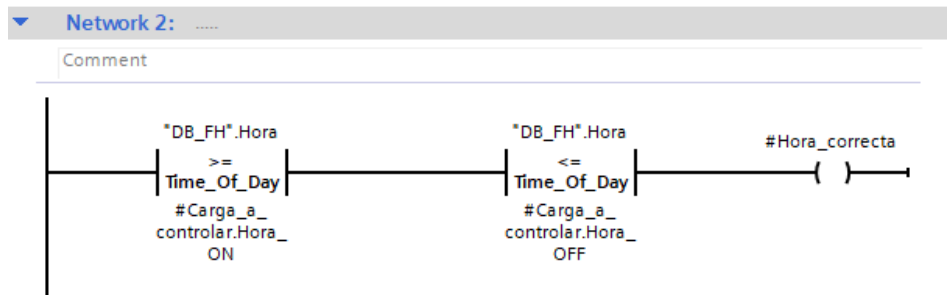


Ilustración 5.40: Network 1 FB\_Control\_Cargas

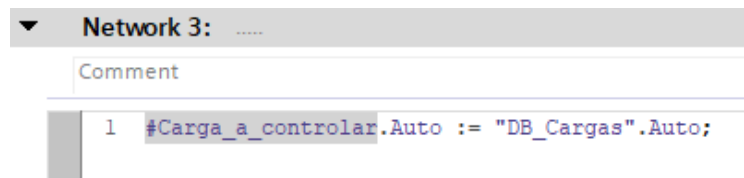
Además, se contrasta la hora actual en la que se ejecuta el programa con el rango de horas dado por los parámetros específicos a esa carga. En caso de encontrarse dentro del rango marcado por "Hora\_ON" y "Hora\_OFF", se activará la variable "Hora\_correcta".



*Ilustración 5.41: Network 2 FB\_Control\_Cargas*

Como se especificó en el apartado [5.5.4](#) se ha considerado más conveniente un modo "Auto" genérico en lugar de que cada carga funcione en automático o manual independientemente de las demás cargas. Esto se gestiona mediante la variable "Auto":

Esto implica que se le asigna el valor de la variable genérica "Auto" del "DB\_Cargas" al parámetro "Auto" correspondiente a la carga a controlar en esa instancia específica.



*Ilustración 5.42: Network 3 FB\_Control\_Cargas*

Las siguientes ramas se centra en la gestión de las cargas en el caso en el que esté desactivado el parámetro "Auto", o lo que es lo mismo, que las cargas estén funcionando en modo manual. Para activar la carga solo se requiere que esté activado el ya mencionado "Flanco\_Puls". También se incluye en serie una consulta al estado de señal "0" de la "Salida" pues, como es lógico, solo se cumple este comportamiento en caso de estar la carga desactivada.

El network 5 se emplea para desactivar la carga en modo manual. De manera análoga a la rama anterior se requiere que esté activado el "Flanco\_Puls". También se incluye en serie una consulta al estado de señal "1" de la "Salida" pues, como es lógico, solo se cumple este comportamiento en caso de estar la carga activada.

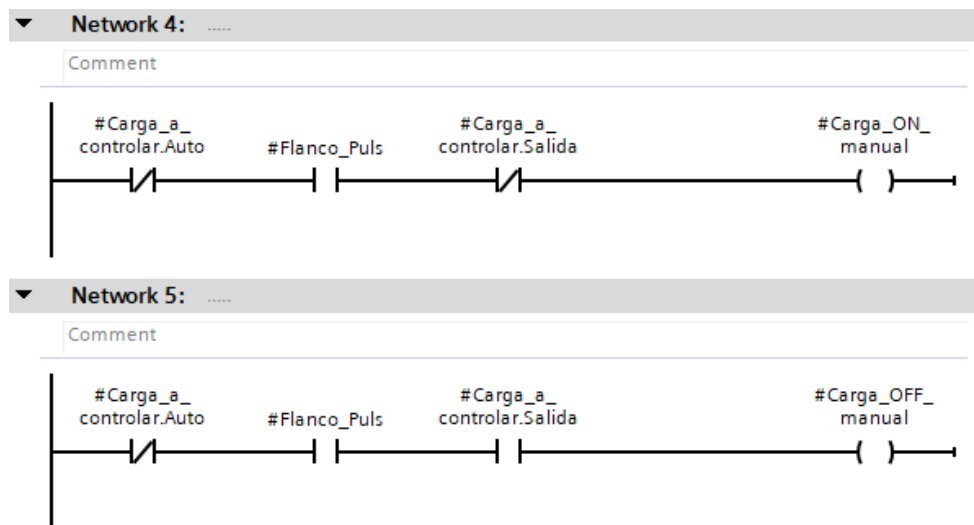


Ilustración 5.43: Network 4 y 5 del FB\_Control\_Cargas

Respecto al modo manual, siendo este el caso de estar activado el parámetro "Auto", si la hora es correcta (Hora\_correcta tiene valor "1") y se activan alguno de los detectores, (pasan a "0") cambian el estado de las variables "Carga\_ON\_auto" y "aux\_auto\_ON" a "1" para activar la carga.

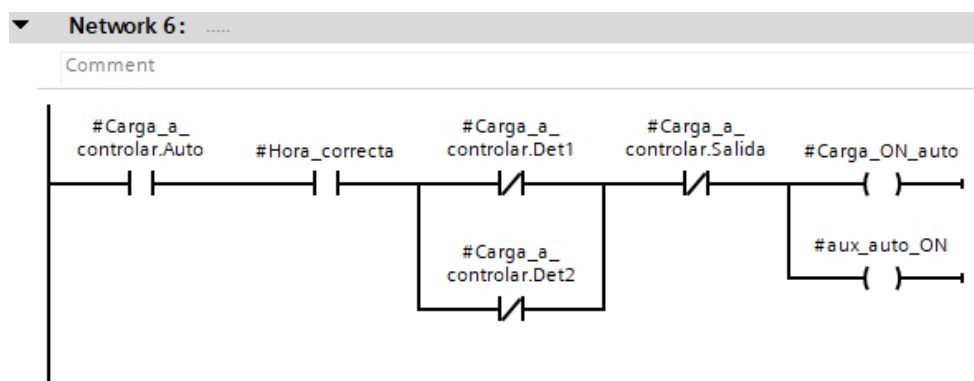


Ilustración 5.44: Network 6 del FB\_Control\_Cargas

Activar la variable "aux\_auto\_ON" se envía un set que iniciará una cuenta atrás delimitada por el valor del parámetro "T\_Ret\_OFF" propio de cada carga específica. Una vez pasado este retardo se activa la variable "Carga\_OFF\_auto" encargada de desactivar a carga. También se activa la variable "aux\_auto\_ON" designada para detener la señal de entrada al bloque "Ins\_TON\_Cargas". Por otra parte, de no encontrarse en el horario correcto marcado por la variable "Hora\_correcta" también se activa la variable "Carga\_OFF\_auto" Por supuesto, cualquiera de estas condiciones consigue apagar la carga solo en el caso de que esté encendida ya que se incluye una consulta al estado de señal "1" de "Salida" previo a la señal de "Carga\_OFF\_auto".

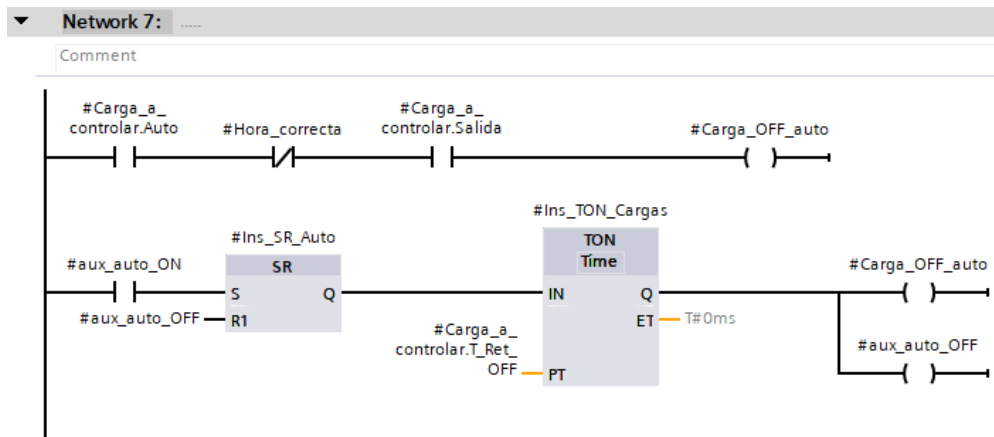


Ilustración 5.45: Network 7 FB\_Control\_Cargas

Las variables "Carga\_ON\_auto", "Carga\_ON\_manual", "Carga\_OFF\_auto" y "Carga\_OFF\_manual" actúan como entradas a un SR que manda señales de set y reset respectivamente al parámetro "Salida". Al ser un SR, implica que la variable "Reset" tiene prioridad. En este caso es "Carga\_OFF\_auto" y "Carga\_OFF\_manual", ya que interesa más tener la carga apagada por motivos de ahorro energético.

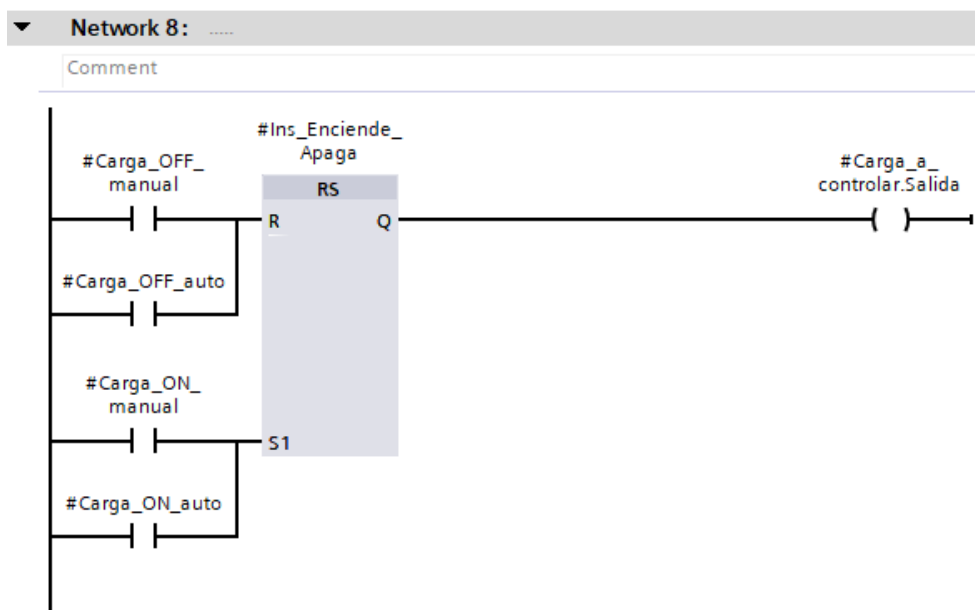


Ilustración 5.46: Network 8 FB\_Control\_Cargas

### 5.7.6. FB\_Control\_Luz (FB1)

Para controlar cada circuito del sistema de iluminación, se trabaja con la variable "Luz\_a\_controlar", que es del tipo "Luz". Para acceder a cada uno de sus parámetros, se sigue la sintaxis: "Luz\_a\_controlar.Parametro". Por ejemplo, para el detector de luminosidad, se accedería a la variable como "Luz\_a\_controlar.Det\_Lum".

Las variables declaradas del bloque funcional se muestran en la siguiente ilustración.

FB_Control_Luz				
	Name	Data type	Default value	Comment
1	Input			
2	Output			
3	InOut			
4	Luz_a_controlar	*Luz*		Variable de luz
5	Static			
6	Flanco_Puls	Bool	false	Flanco del pulsador
7	Activa_Luz	Bool	false	Señal de activación
8	Desactiva_Luz	Bool	false	Señal de desactivación
9	Ins_Enciende_Apaga	Bool	false	Instancia RS
10	Hora_correcta	Bool	false	TRUE si se está dentro del horario
11	Inst_R_Trig	R_TRIG		Intancia de flanco positivo
12	Ins_SR_Auto	Bool	false	Instancia Set Reset para el automático
13	aux_auto	Bool	false	variable auxiliar para auto
14	aux_auto_ON	Bool	false	variable auxiliar para auto ON
15	aux_auto_OFF	Bool	false	variable auxiliar para auto OFF
16	Desactiva_Luz_manual	Bool	false	
17	Desactiva_Luz_auto	Bool	false	
18	Activa_Luz_manual	Bool	false	
19	Activa_Luz_auto	Bool	false	
20	Ins_TON_Luz	TON_TIME		Instancia Temporizador de Retardo a la Conexión
21	Temp			
22	Constant			

Ilustración 5.47: Variables FB\_Control\_Luz

Primero, se comprueba que la hora en la que se ejecuta el programa está dentro del horario establecido para el funcionamiento de las luces instanciadas. De ser así, se activaría la variable "Hora\_correcta".

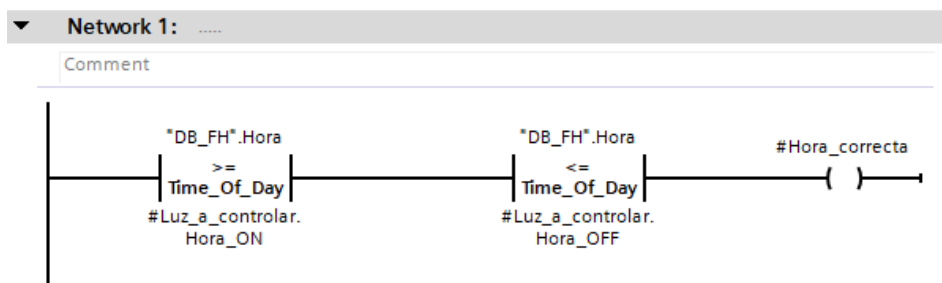


Ilustración 5.48: Network 1 FB\_Control\_Luz

Como se especificó en el apartado 5.5.6 se consideró más conveniente un modo "Auto" genérico en lugar de que cada punto de luz funcione en automático o manual independientemente. Esto se gestiona mediante la variable "Auto". Esto implica que se le asigna el valor de la variable genérica "Auto" del "DB\_Luz" al parámetro "Auto" correspondiente a la luz a controlar en esa instancia específica.

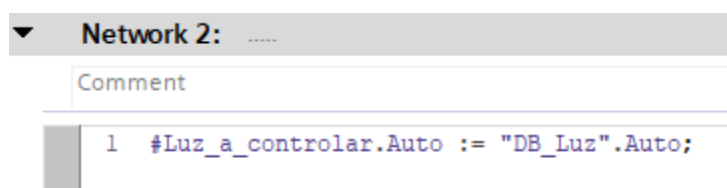
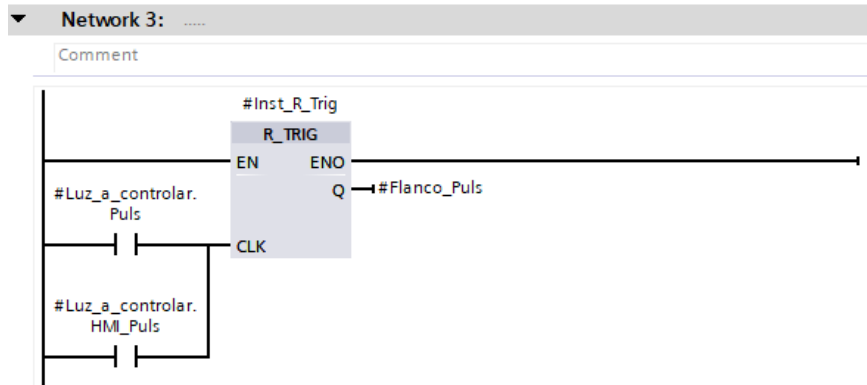


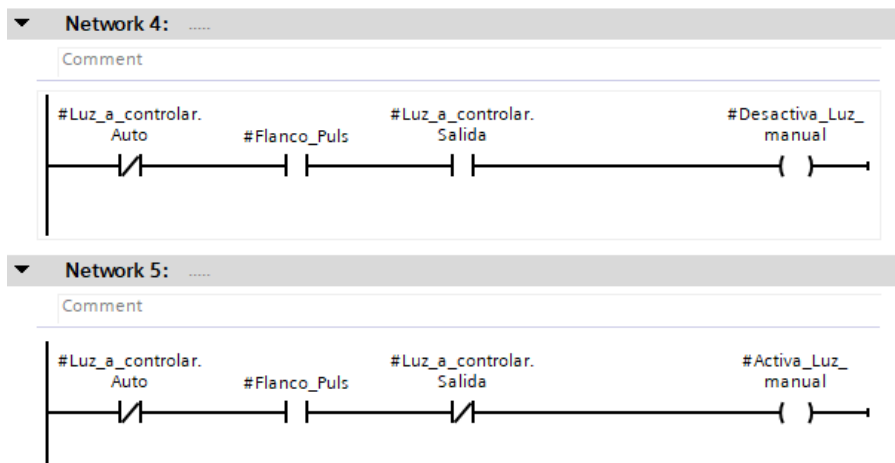
Ilustración 5.49: Network 2 FB\_Control\_Luz

Respecto al tercer segmento utiliza un bloque funcional estándar R\_TRIG para generar un flanco ascendente (*Flanco\_Puls*) cada vez que el usuario seleccione uno de los interruptores de las cargas en el HMI o pulse el correspondiente pulsador.



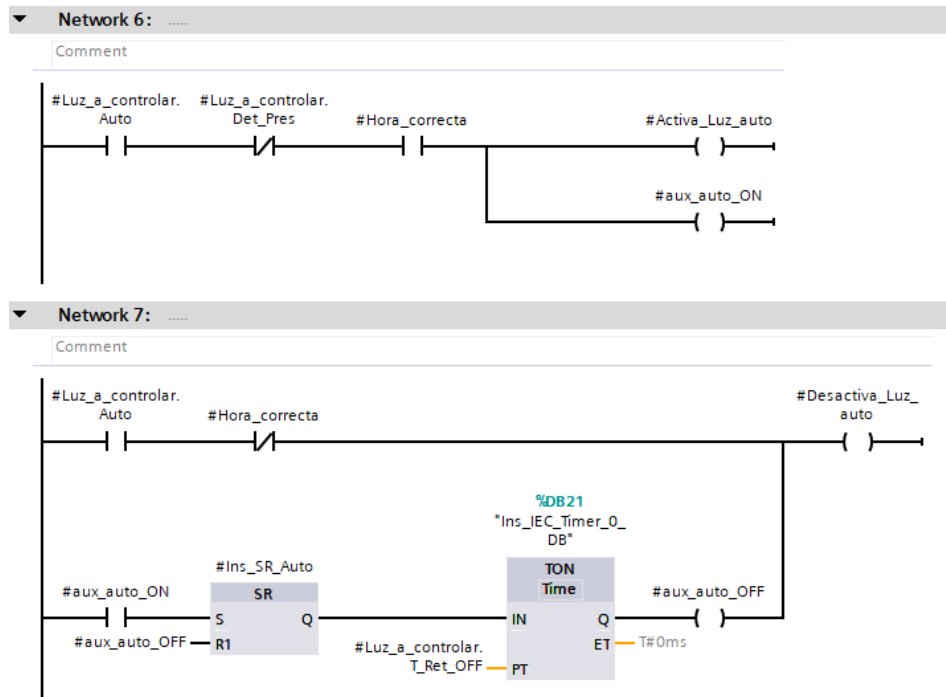
*Ilustración 5.50: Network 3 FB\_Control\_Luz*

En caso de funcionar en manual (*Auto=0*), basta con una señal positiva en "*Flanco\_Puls*" estando la luz apagada, para generar una señal "*Activa\_Luz\_Manual*".



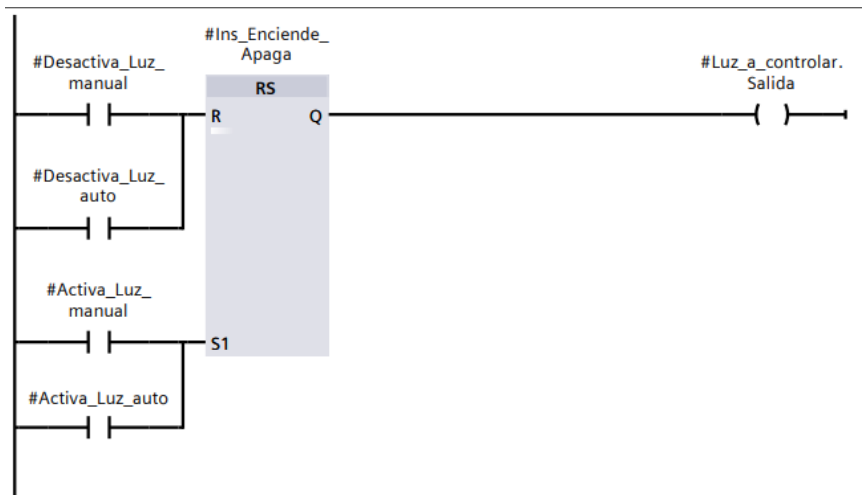
*Ilustración 5.51: Network 4 y 5 FB\_Control\_Luz*

De no ser el caso (*Auto=1*), al detectarse presencia y encontrarse en el horario correcto (*Hora\_correcta=1*), se genera una señal positiva tanto en "*Activa\_Luz\_Auto*" como en "*aux\_auto\_ON*" solo en caso de que la luz estuviera apagada. Esta última variable es una variable interna del "*FB\_Control\_Luz*" que activa un temporizador encargado de gestionar el tiempo que la luz estará encendida después de la detección de presencia. Tras este retardo, enviará una señal "*Desactiva\_Luz\_Auto*".



*Ilustración 5.52: Network 6 y 7 FB\_Control\_Luz*

La última rama consiste en un bloque funcional estándar RS que envía todas las señales de desactivar y activar a las entradas R y S respectivamente.



*Ilustración 5.53: Network 8 FB\_Control\_Luz*

### 5.7.7. FB\_Persianas (FB8)

Por último, el bloque funcional dedicado al control de las persianas utiliza la variable "Persiana\_a\_controlar", de tipo "Persiana". Para acceder a cada uno de sus parámetros, se sigue la sintaxis: "Persiana\_a\_controlar.parametro". Por ejemplo, para el detector de viento, se accedería a la variable como "Persiana\_a\_controlar.Det\_Viento".

Las variables declaradas del bloque funcional se muestran en la siguiente ilustración:



FB_Persianas							
	Name	Data type	Default value	Comment	/Writa...	Visible in ...	Setpoint
1	Input				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	HMI_Subir	Bool	false	Pulsador HMI subir	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
3	HMI_Bajar	Bool	false	Pulsador HMI bajar	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
4	Output				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	InOut				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	Persiana_a_controlar	"Persiana"		Variable persiana	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	Static				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	Hora_correcta	Bool	false	TRUE si pertenece al horario	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
9	Temp				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	aux	Time		variable auxiliar	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	Constant				<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Ilustración 5.54: Variables FB\_Periasanas

Primero, se comprueba que se encuentra en el rango horario correcto. De ser así, se activa la variable "Hora\_correcta", como se aprecia en el segmento siguiente.

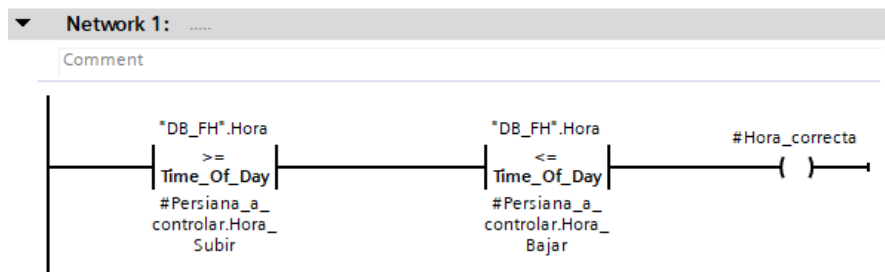


Ilustración 5.55: Network 1 FB\_Persianas

Este bloque cuenta con la particularidad respecto a otros FB de control y es que se consideró más conveniente realizarlo mayoritariamente en texto estructurado. En resumen, en modo manual envía una señal positiva a "Subir" y negativa a "Bajar" si recibe señal de subir desde el pulsador físico o el del HMI y viceversa.

Por otra parte, envía una señal positiva a "Subir" y negativa a "Bajar" en el caso de que se encuentre en el horario correcto, detecte luz y no detecte ni lluvia ni viento.

```

Network 2: -----
Comment
1 IF (#Persiana_a_controlar.Auto) THEN
2   IF (#Persiana_a_controlar.Det_Lluvia AND #Persiana_a_controlar.Det_Lum AND #Persiana_a_controlar.Det_Viento) THEN
3     #Persiana_a_controlar.Subir := TRUE; (* Enviar orden de subir si hay luz, no llueve y no hay viento *)
4     #Persiana_a_controlar.Bajar := FALSE; (* No bajar *)
5   ELSE
6     #Persiana_a_controlar.Subir := FALSE; (* No subir *)
7     #Persiana_a_controlar.Bajar := TRUE; (* Enviar orden de bajar en cualquier otra situación *)
8   END_IF;
9 ELSE
10  IF (#Persiana_a_controlar.Puls_Subir OR #HMI_Subir) THEN
11    #Persiana_a_controlar.Subir := TRUE; (* Enviar orden de subir manualmente *)
12    #Persiana_a_controlar.Bajar := FALSE; (* No bajar *)
13  ELSE
14    IF (#Persiana_a_controlar.Puls_Bajar OR #HMI_Bajar) THEN
15      #Persiana_a_controlar.Subir := FALSE; (* No subir *)
16      #Persiana_a_controlar.Bajar := TRUE; (* Enviar orden de bajar manualmente *)
17    ELSE
18      #Persiana_a_controlar.Subir := FALSE; (* No subir *)
19      #Persiana_a_controlar.Bajar := FALSE; (* No bajar *)
20    END_IF;
21  END_IF;
22 END_IF;
23
  
```

Ilustración 5.56: Network 2 FB\_Persianas

### 5.7.8. SALIDAS (FB2)

En este bloque funcional se realizan las asignaciones a las variables asociadas a la memoria de salidas del PLC desde las variables internas del programa definidas en las estructuras de datos correspondientes a cada funcionalidad. Se consideró más conveniente realizarlo en texto estructurado (SCL) para una más cómoda implementación.

SALIDAS					
IF...	CASE... OF...	FOR... TO DO...	WHILE... DO...	(*...*)	REGION
1	"Q_Luz_Salon"	:=	"DB_Luz".Salon.Salida;		// Q0.0
2	"Q_Luz_Aseo"	:=	"DB_Luz".Aseo.Salida;		// Q0.1
3	"Q_Luz_Hab"	:=	"DB_Luz".Habitacion.Salida;		// Q0.2
4	"Q_Carga_Cal"	:=	"DB_Calefaccion"."Cocina-Salon".Salida		
5		OR	"DB_Cargas".Calefactor.Salida;		// Q0.3
6	"Q_Carga_Vent"	:=	"DB_Persianas".Toldo.Det_Viento;		// Q0.4
7	"Q_Carga_Coc"	:=	"DB_Cargas".Cocina.Salida;		// Q0.5
8	"Q_EV_Agua"	:=	"DB_Alarmas_Tecnicas".Inundacion.Disparada;		// Q0.6
9	"Q_Luz_Coc"	:=	"DB_Luz".Cocina.Salida;		// Q0.7
10	"Q_Subir_Pers"	:=	"DB_Persianas".Toldo.Subir;		// Q1.0
11	"Q_Bajar_Pers"	:=	"DB_Persianas".Toldo.Bajar ;		// Q1.1

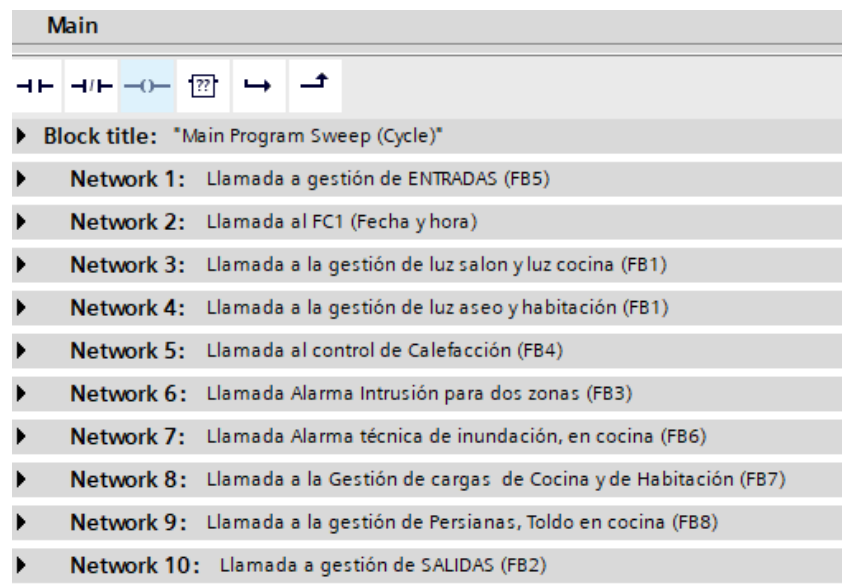
Ilustración 5.57: Código FB\_Salidas

Nótese que se escribe solamente una vez la variable de la memoria de salidas del PLC aunque pueda tomar el valor de varias variables auxiliares combinadas (como la Q 0.3, es decir, "Q\_Carga\_Cal" que toma el valor "DB\_Calefaccion"."Cocina-Salon". Salida y "DB\_Cargas".Calefactor.Salida.

Esta forma de proceder garantiza una mayor seguridad para la depuración y prueba del programa de control teniendo siempre controlado el punto en el que asignan los valores a los elementos actuadores de la instalación.

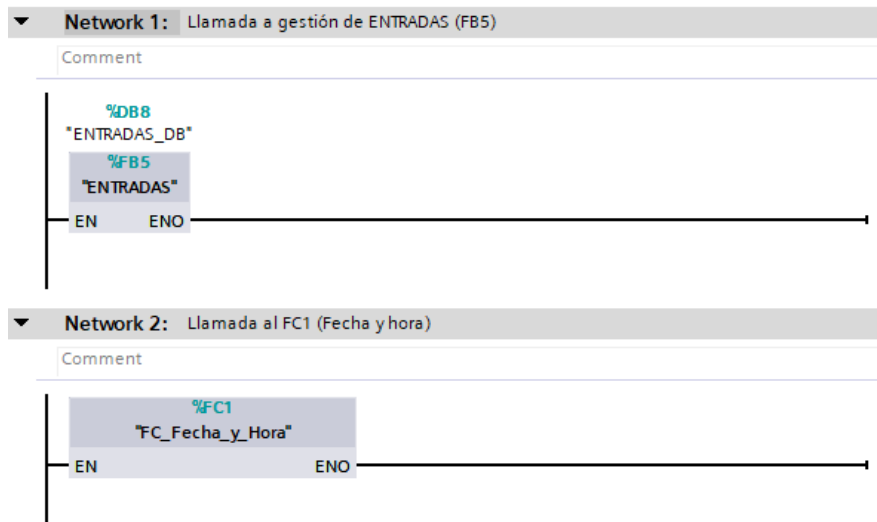
## 5.8. MAIN (OB1)

Para la ejecución cíclica del programa de control (ciclo de scan), en la programación de controladores SIMATIC se dispone de un bloque de organización especial (OB1, se ha denominado Main) que se ejecuta cíclicamente permitiendo la gestión en tiempo real de las entradas y salidas conectadas al controlador en combinación con el resto de los módulos de programa. Por tanto, es preciso desde este programa, instanciar (llamar) a los módulos de programación implementados que en este caso se hace secuencialmente (ver código correspondiente en el anexo). En la ilustración siguiente se muestra de forma resumida el orden de dichas llamadas extraído del programa en TIA Portal.



*Ilustración 5.58: OB1 colapsado*

Como se ve, por tanto, lo único que se precisa es crear las llamadas, primero al módulo que gestiona las ENTRADAS (FB5) y posteriormente a la función de fecha y hora (FC1).

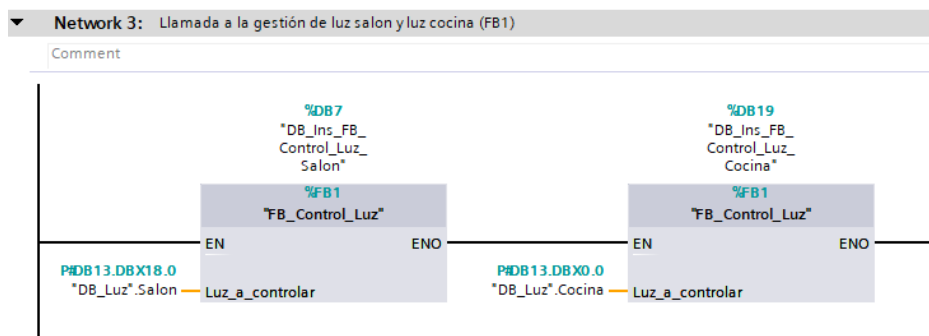


*Ilustración 5.59: Network 1 y 2 Main (OB1)*

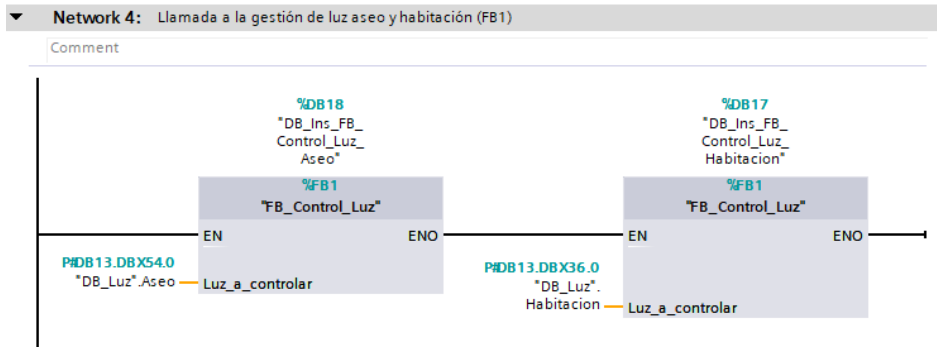
A continuación, se encuentran las instancias relativas a las funcionalidades implantadas con sendas llamadas a los bloques funcionales correspondientes, desde la Network 3 a la 9 del módulo de organización Main (OB1).

Cabe destacar en este punto, que basta usar como parámetros IN\_OUT las variables asociadas a los tipos de dato creados (UDT) pues encapsulan la información correspondiente que resuelve el código interno del bloque funcional para su correcta aplicación. En las siguientes ilustraciones se aprecia este modo de proceder para las llamadas a los distintos FBs.

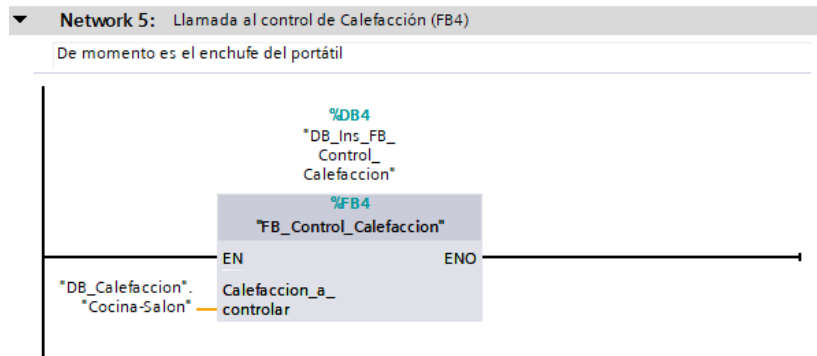
Por ejemplo, y de forma más concreta, el FB\_Control\_Persianas requiere un tipo de dato "Persiana" para funcionar, por lo que se sustituye en el FB por lo que se había llamado "Persiana\_a\_controlar". En otras palabras, se especifica que la persiana a controlar es concretamente la que en el DB de configuración está etiquetada como "Toldo".



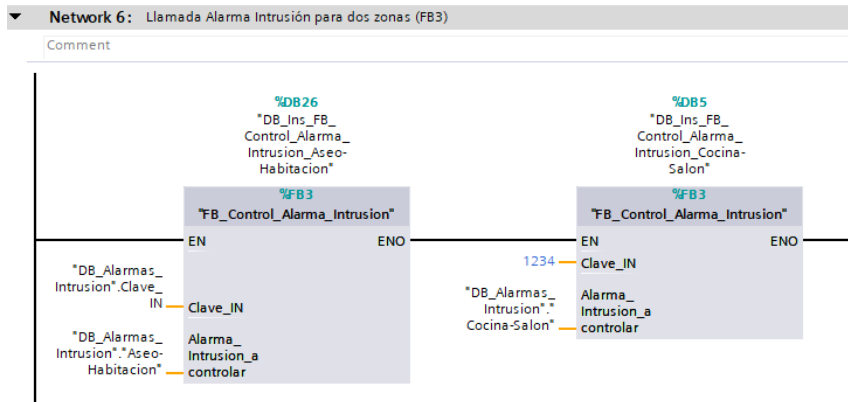
*Ilustración 5.60: Network 3 Main (OB1)*



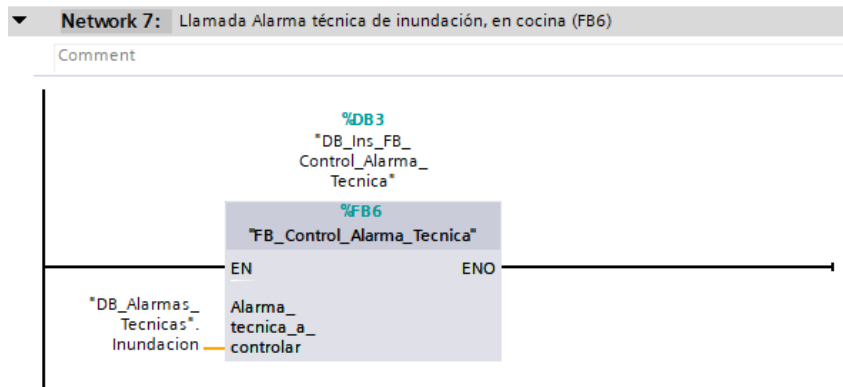
*Ilustración 5.61: Network 4 Main (OB1)*



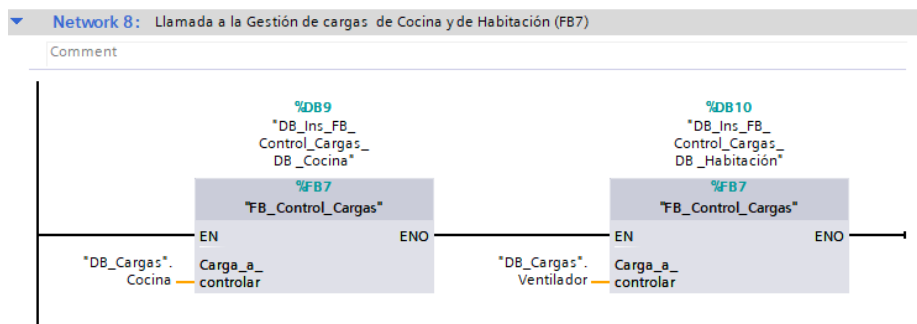
*Ilustración 5.62: Network 5 Main (OB1)*



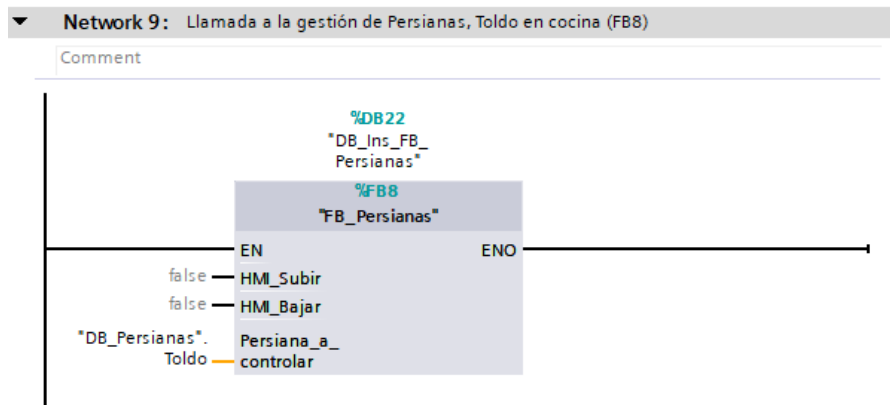
*Ilustración 5.63: Network 6 Main (OB1)*



*Ilustración 5.64: Network 7 Main (OB1)*

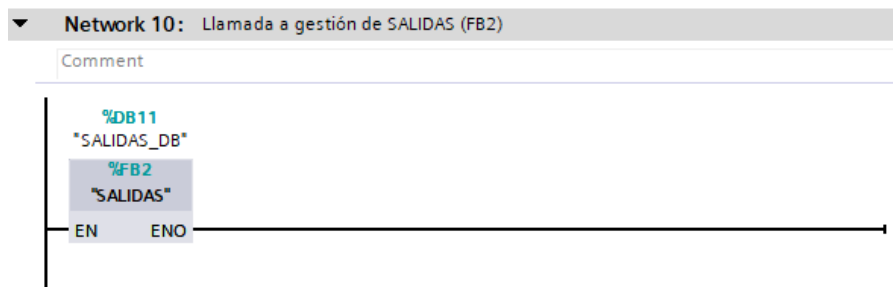


*Ilustración 5.65: Network 8 Main (OB1)*



*Ilustración 5.66: Network 9 Main (OB1)*

Para finalizar se llama a la gestión de las SALIDAS (FB2) según se ha comentado en apartado anterior.



*Ilustración 5.67: Network 10 Main (OB1)*

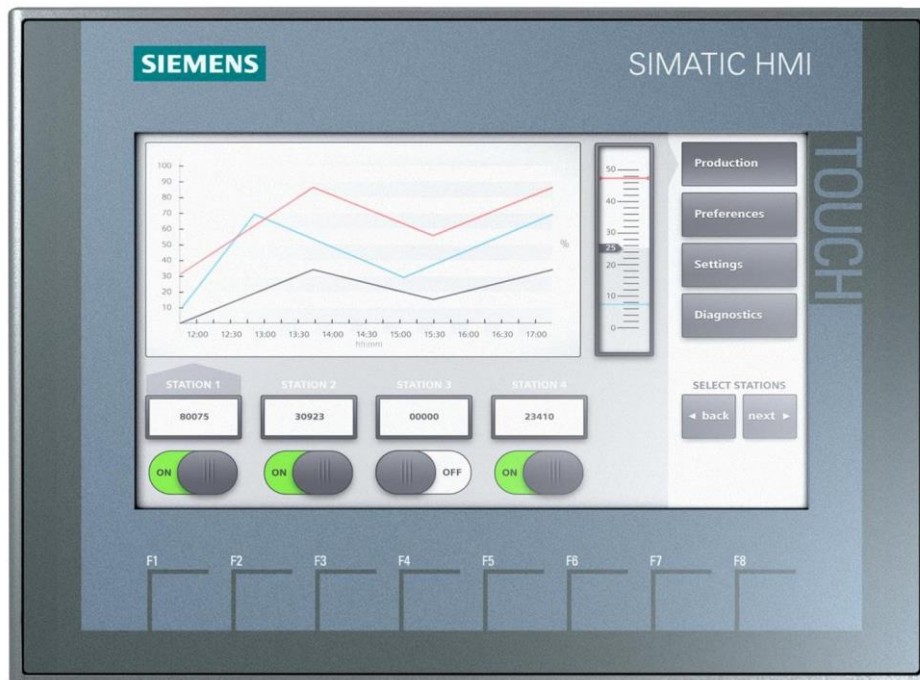
## 6. Pantalla de explotación (HMI)

### 6.1. INTRODUCCIÓN

Se han creado una aplicación a modo de interfaz de usuario HMI (*Human Machine Interface*) que consiste en un conjunto de pantallas para la configuración y manejo de todo el sistema.

Considerando que se está utilizando un controlador programable tipo PLC, es habitual que estos equipos se conecten con pantallas táctiles que ofrecen gran robustez y prestaciones cada vez más avanzadas. La utilización de una aplicación HMI basada en este tipo de pantallas táctiles tiene sentido en este trabajo pues se trata también de una instalación, el piso piloto *E-Llar*, que sirve para la formación tanto a nivel teórico como práctico, de la implantación de soluciones para la gestión técnica de instalaciones en la vivienda.

Se ha elegido una pantalla táctil real (KTP 700 Basic PN) que es preciso integrar en el proyecto de TIA Portal como se ha visto en la *Ilustración 5.5*. Posteriormente se permite la simulación sin necesidad de la pantalla física, permitiendo el acceso a las variables de PLC definidas. Esto no solo ayuda en el proceso de depuración de los programas de control implantados en el PLC como servir de proyecto para la pantalla real en el caso de que esta estuviera a disposición.

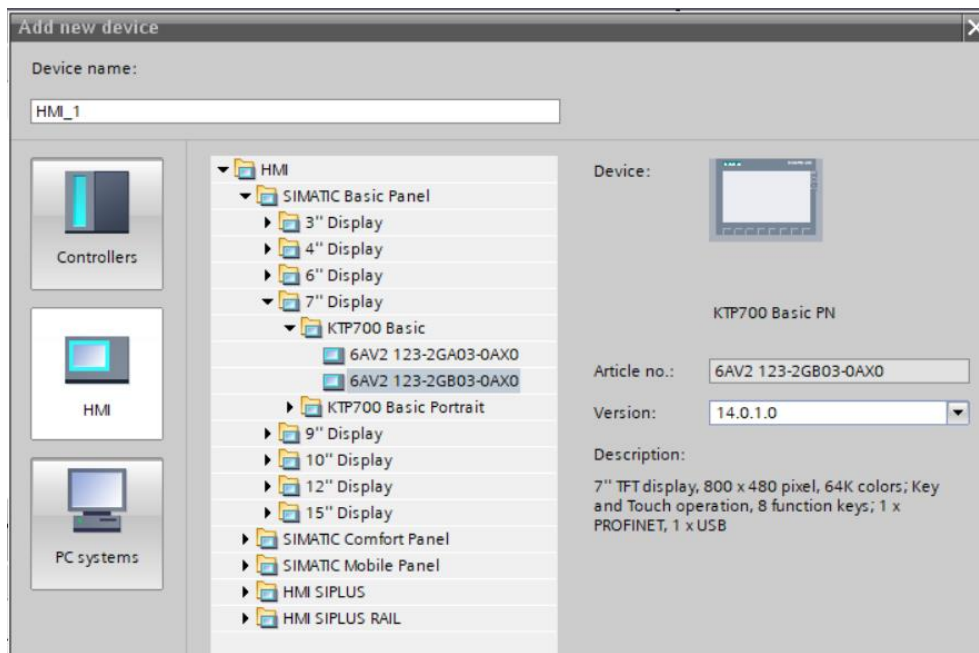


*Ilustración 6.1: HMI KTP700*



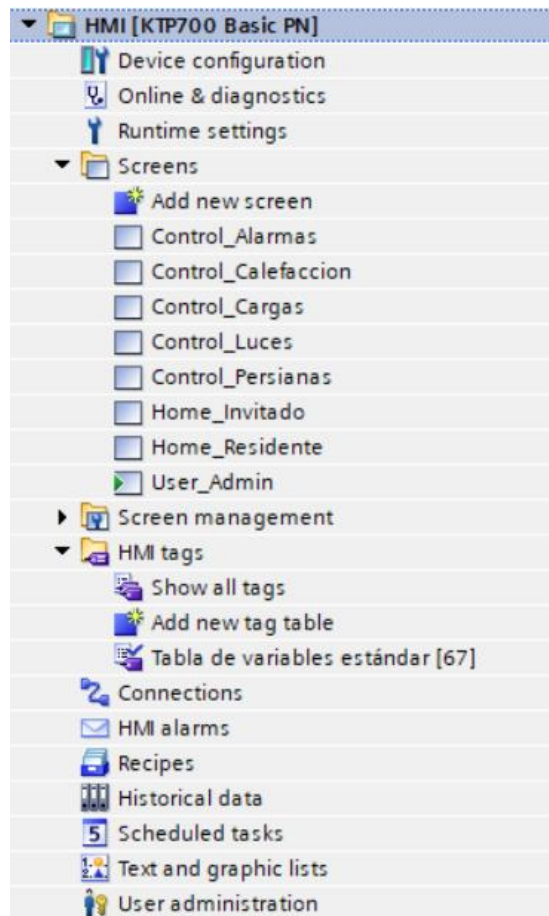
## 6.2. CREACIÓN Y CONFIGURACIÓN DE LA APLICACIÓN HMI (KTP700 BASIC PN)

El dispositivo se puede añadir en el árbol del proyecto a través del catálogo hardware disponible en TIA Portal v17, en la parte de componentes HMI según se muestra en la ilustración siguiente.



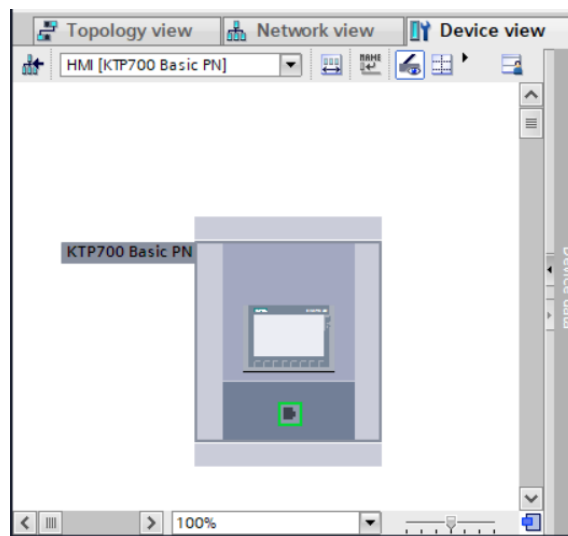
*Ilustración 6.2: Catalogo de hardware*

Una vez seleccionada la pantalla correspondiente, se crea una entrada en el árbol del proyecto. Se han utilizado a efectos de este trabajo, solo algunas de ellas, bien por la necesidad de configuración de algunos parámetros o por su conveniencia, detalles que se ilustran a continuación.



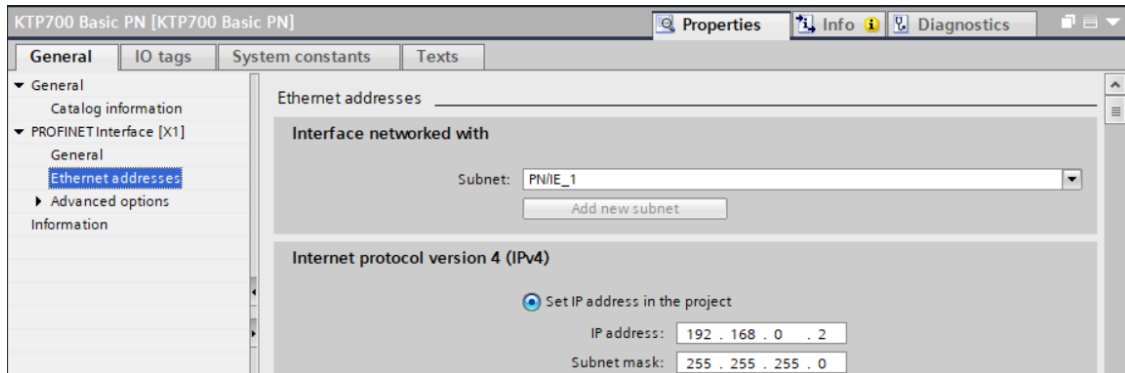
*Ilustración 6.3: Árbol del proyecto en el HMI*

Device configuration: Permite el acceso la *vista del dispositivo* y la vista de red. En el primer caso se pueden indicar propiedades del dispositivo.



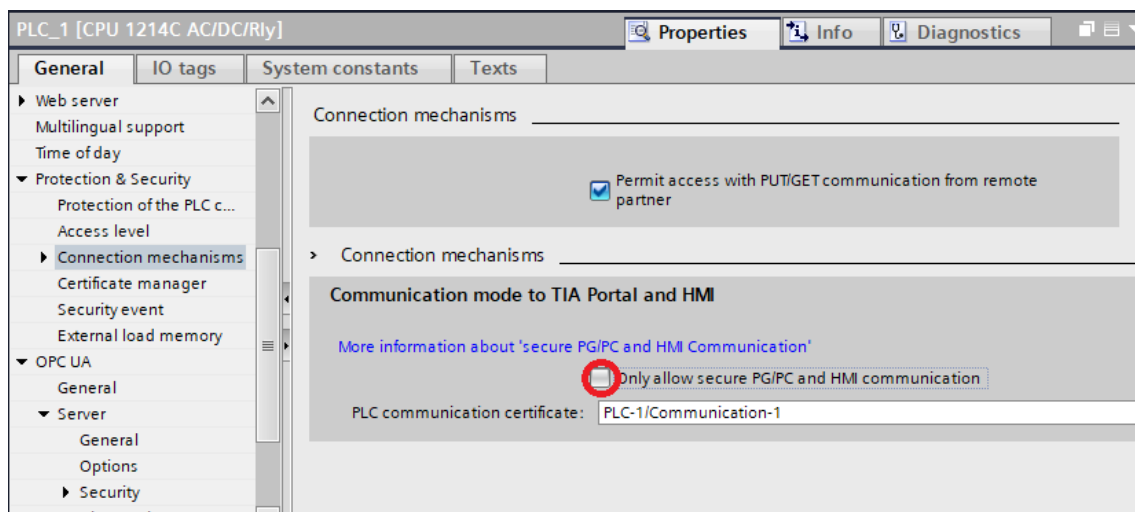
*Ilustración 6.4: Device configuration del HMI*

La parte principal de configuración es la dirección ethernet que debe estar en el mismo rango de la subred que está definida en el proyecto general. Se han utilizado los parámetros que se apuntan a continuación como de mayor interés.



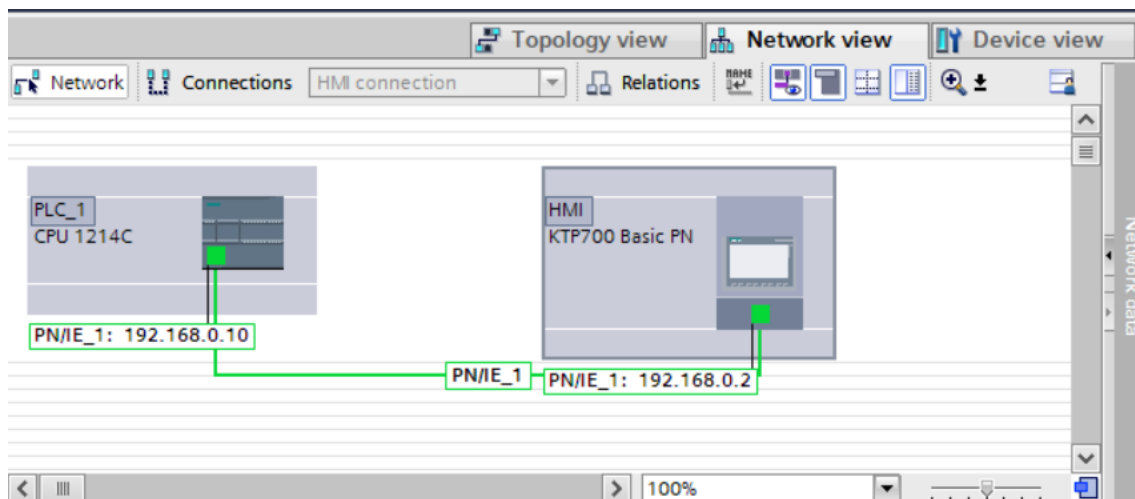
*Ilustración 6.5: Configuración IP del HMI*

En la versión 17 de TIA Portal para una correcta gestión y visualización de los Tags del PLC es necesario desactivar la opción “Only allow secure PG/PC and HMI communications”. Dado que de estar marcada está opción el propio TIA Portal bloqueará la comunicación con el HMI por seguridad.



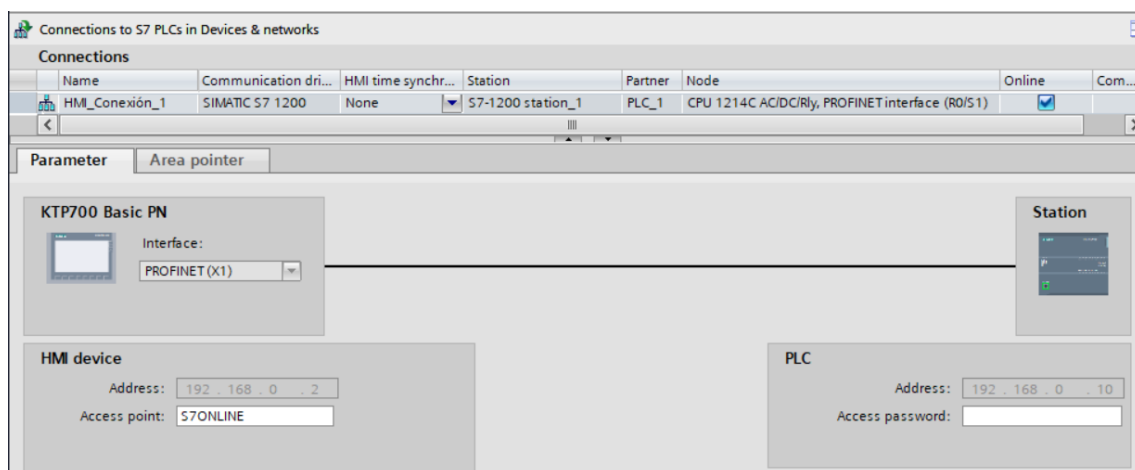
*Ilustración 6.6: Permitir comunicación con HMI*

Se debe también establecer la conexión entre la pantalla y el PLC de modo que sea posible enlazar variables del controlador en los diferentes objetos que se integren para la explotación y supervisión del sistema. En la *vista de red* de TIA Portal quedaría según se observa en la imagen.



*Ilustración 6.7: Vista de redes*

Finalmente, en la opción de conexiones del árbol del HMI (Connections) es posible observar entonces la que se ha establecido entre los equipos reseñados.



*Ilustración 6.8: Conexiones del árbol del HMI*

En la carpeta Screens se pueden crear las ventanas de la aplicación donde incorporar elementos gráficos para la supervisión y explotación del sistema domótico. El diseño se ha basado en una primera ventana denominada *User\_Admin*, y a partir de ella poder entrar al sistema en modo Invitado o Residente, siendo este segundo el que tendrá acceso a las opciones completas implementadas. La descripción de cada una de las ventanas de la aplicación HMI desarrollada se encuentra más adelante en este documento.

Dentro de la carpeta HMI tags se pueden encontrar las variables que enlazar las acciones dinámicas asociadas a los objetos de las distintas ventanas con las variables del PLC que directamente se localizan en virtud de la conexión establecida. A efectos del HMI, por cada variable de PLC utilizada, en este caso, correspondiente a algunas definidas en los bloques de datos de la aplicación se genera una para dicho HMI con un nombre que sustituye los puntos “.” de separación de los campos de la variable PLC por un guión bajo “\_”. Esto se puede apreciar en la ilustración siguiente para algunas de las dichas variables, que intenta explicar este procedimiento que genera de forma automática

TIA Portal, si bien los nombre (*Name*) de las variables del HMI se puede cambiar a voluntad –en este caso se ha preferido mantener dichos nombres. También sería posible en esta tabla modificar los *PLC tag* como se prefiera, aunque habitualmente se generan a medida que se va creando la aplicación.

HMI tags						
Name	Tag table	Data type	Connecti...	PLC na..	PLC tag	
DB_Ins_FB_Control_Luz_Cocina_Hora_correcta	Tabla de ...	Bool	HMI_...	PLC_1	DB_Ins_FB_Control_Luz_Cocina.Hora_correcta	
DB_Ins_FB_Control_Luz_Aseo_Enciende_Apaga	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Ins_FB_Control_Luz_Aseo.Ins_Enciende_Apaga	
DB_Ins_FB_Control_Calefaccion_Hora_correcta	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Ins_FB_Control_Calefaccion.Hora_correcta	
DB_Cargas_Ventilador_Salida	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Ventilador.Salida	
DB_Cargas_Ventilador_Orden_ON_OFF	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Ventilador.Orden_ON_OFF	
DB_Cargas_Ventilador_HMI_Carga	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Ventilador.HMI_Carga	
DB_Cargas_Cocina_Salida	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Cocina.Salida	
DB_Cargas_Cocina_Orden_ON_OFF	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Cocina.Orden_ON_OFF	
DB_Cargas_Cocina_HMI_Carga	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Cocina.HMI_Carga	
DB_Cargas_Cocina_Auto	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Cocina.Auto	
DB_Cargas_Calefactor_Salida	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Calefactor.Salida	
DB_Cargas_Calefactor_Orden_ON_OFF	Tabla de vari...	Bool	HMI_Con...	PLC_1	DB_Cargas.Calefactor.Orden_ON_OFF	

*Ilustración 6.9: HMI Tags*

Finalmente se ha incluido cierta gestión de usuarios mediante la opción User administration del árbol HMI. Para ello se han creado varios grupos de usuario y usuarios propiamente dichos. Si bien los detalles pueden verse directamente en el fichero de la aplicación TIA Portal desarrollada, basta un par de imágenes para apreciar los aspectos de configuración que son necesarios.

Groups				
Name	Number	Display name	Password aging	Comment
Grupo de admi...	1	Grupo de adminis...	<input type="checkbox"/>	El grupo 'Administradores' tiene inicialmente todos los derechos.
Usuarios	2	Usuarios	<input type="checkbox"/>	El grupo 'Usuarios' tiene inicialmente el permiso 'Operación'.
Residente	3	Residente	<input type="checkbox"/>	
Invitado	4	Invitado	<input type="checkbox"/>	
<Add new>				

Authorizations				
Active	Name	Display name	Number	Comment
<input checked="" type="checkbox"/>	Administración de usuari...	Administración de usuarios	1	Permiso 'Administración de ...
<input checked="" type="checkbox"/>	Monitorización	Monitorización	2	Permiso 'Supervisar'.
<input checked="" type="checkbox"/>	Operación	Operación	3	Permiso 'Operación'.
<input checked="" type="checkbox"/>	Residente	Permiso_1	4	
<input checked="" type="checkbox"/>	Invitado	Permiso_1	5	
<Add new>				

*Ilustración 6.10: Gestión de permisos HMI*

Users						
Name	Password	Automatic logoff	Logoff time	Number	Comment	
Felipe	*****	<input checked="" type="checkbox"/>	5	1	El usuario 'Administrador' se ..	
Celia	*****	<input checked="" type="checkbox"/>	5	2		
Nacho	*****	<input checked="" type="checkbox"/>	3	3		
Paula	*****	<input checked="" type="checkbox"/>	3	4		
Lorena	*****	<input checked="" type="checkbox"/>	5	5		
Invitado	*****	<input checked="" type="checkbox"/>	5	6		
C	*****	<input checked="" type="checkbox"/>	15	7		
<Add new>						

Groups						
Member of	Name	Number	Display name	Password aging	Comment	
<input type="radio"/>	Grupo de ad...	1	Grupo de adminis...	<input type="checkbox"/>	El grupo 'Administradores' tiene inicialmente t..	
<input type="radio"/>	Usuarios	2	Usuarios	<input type="checkbox"/>	El grupo 'Usuarios' tiene inicialmente el permis..	
<input checked="" type="radio"/>	Residente	3	Residente	<input type="checkbox"/>		
<input type="radio"/>	Invitado	4	Invitado	<input type="checkbox"/>		

Ilustración 6.11: Usuarios del HMI

### 6.3. ORGANIZACIÓN DE VENTANAS

Dada que la limitada complejidad de la implementación del HMI y su dependencia de los aspectos de configuración propios de TIA Portal, no se comentarán detalles al respecto, pero si se mostrará, a continuación, una idea de las ventanas desarrolladas y los elementos que se manejan a modo de Manual de Usuario.

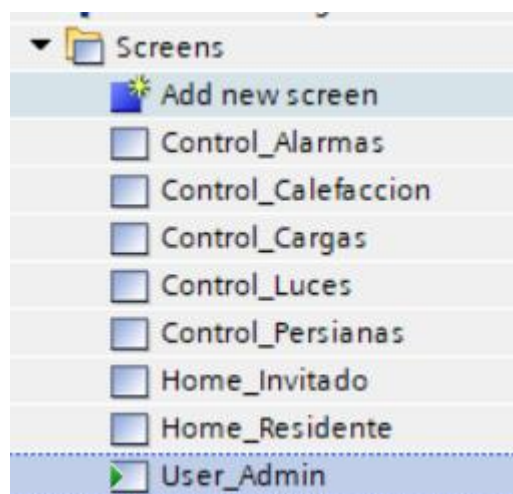


Ilustración 6.12: Pantallas del HMI

Para iniciar la simulación del funcionamiento de la pantalla KTP700 Basic PN, una vez cargado el programa en el controlador SIMATIC S7-1200, debe posicionarse en el árbol del proyecto HMI y activar con el botón derecho del ratón la opción *Start simulation* ó pulsando *Ctrl+Shift+X*; también hay un icono en el menú general.

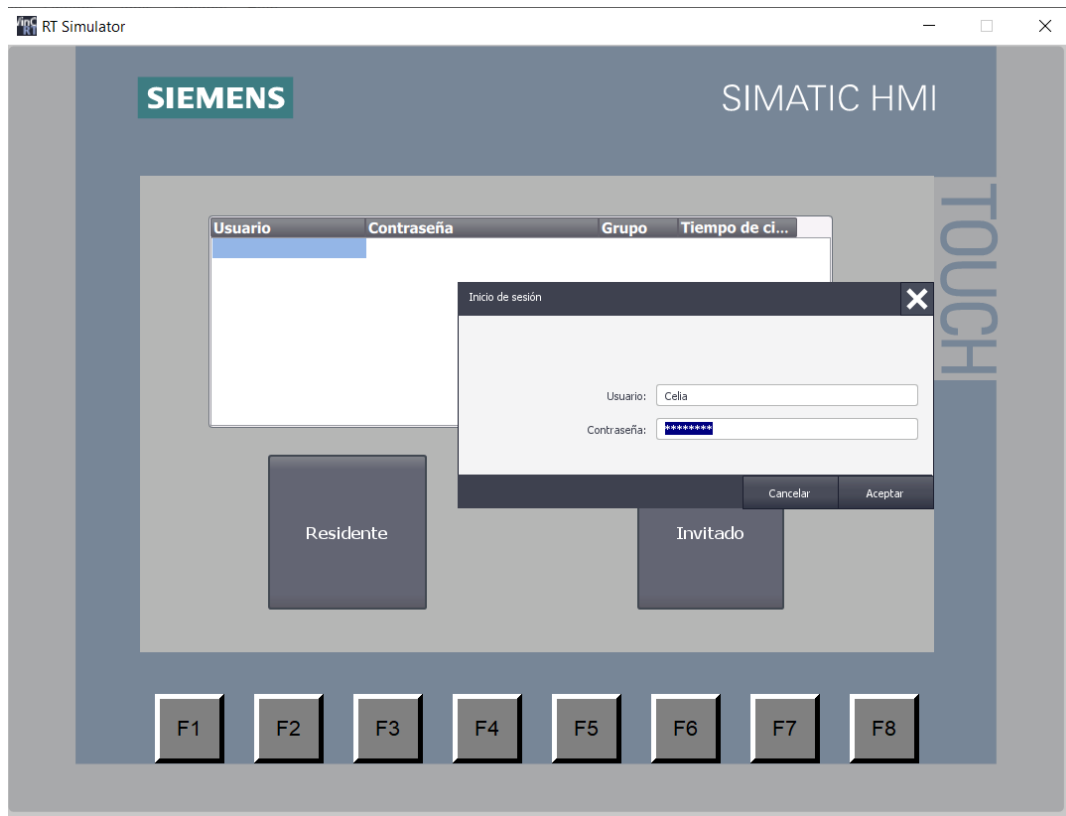
Entonces se abrirá una representación en modo runtime de la pantalla con WinCC RT Simulator pudiendo verificar el funcionamiento de la aplicación en coordinación con el PLC. La primera ventana que se abre es *User\_Admin* que es la configurada como *Start simulation* y por eso su icono en el árbol tiene un triángulo verde característico.



*Ilustración 6.13: Start simulation*








### 6.3.1. Ventana User\_Admin

Al iniciar la simulación se accede a la pantalla de usuarios, donde puede iniciar sesión como Residente o como Invitado.



*Ilustración 6.14: Inicio de sesión en el HMI*

Los usuarios registrados de muestran en la [Ilustración 6.11](#). La contraseña es en todos los casos 1234. Aunque no es la opción más segura, se ha elegido por conveniencia en esta fase de desarrollo, ya que, una vez seleccionada una contraseña, no se puede cambiar posteriormente sin entrar en modo edición de la aplicación.

Usuarios					
	Nombre	Contraseña	Cierre de sesión autom..	Tiempo de cierre de sesió	Número
	Felipe	*****	<input checked="" type="checkbox"/>	5	1
	Celia	*****	<input checked="" type="checkbox"/>	5	2
	Nacho	*****	<input checked="" type="checkbox"/>	3	3
	Paula	*****	<input checked="" type="checkbox"/>	3	4
	Lorena	*****	<input checked="" type="checkbox"/>	5	5
	Invitado	*****	<input checked="" type="checkbox"/>	5	6
	C	*****	<input checked="" type="checkbox"/>	15	7

*Ilustración 6.15: Listado de usuarios*

Algunos de los usuarios tienen permisos de Residente y otros solo de Invitado. Al iniciar sesión como Invitado, solo tendrá acceso al control de persianas, luces y cargas. Sin embargo, como Residente, además puedes acceder al control de alarmas (tanto técnicas como de intrusión) y calefacción.

Independientemente de los permisos que tenga, la sesión se cerrará automáticamente después de unos minutos por razones de seguridad. Asimismo, si regresa a la pantalla de administración de usuarios, la sesión se cerrará automáticamente nuevamente.

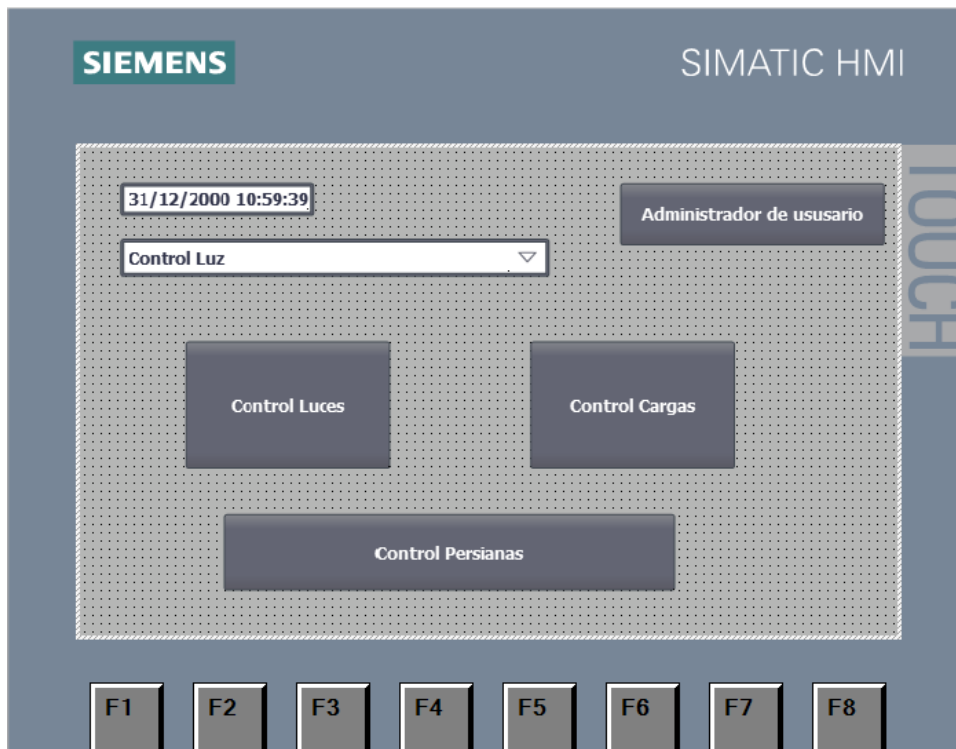
### 6.3.1. Ventanas Home

Dado que cada usuario tiene distintos permisos, se consideran dos pantallas "Home" a las que podrán acceder los usuarios en función de los privilegios asignados:

#### 6.3.1.1. Ventana Home\_Invitado

Aquellos usuarios que tengan permiso de invitado tendrán acceso exclusivamente a la ventana "Home\_Invitado", desde la cual podrán acceder a las ventanas "Control\_Luces", "Control\_Persianas" y "Control\_Cargas". No se considera necesario que los invitados tengan la posibilidad de gestionar ni alarmas ni la calefacción de la vivienda. Por su parte, los usuarios con permiso de residente también podrán acceder a esta pantalla.

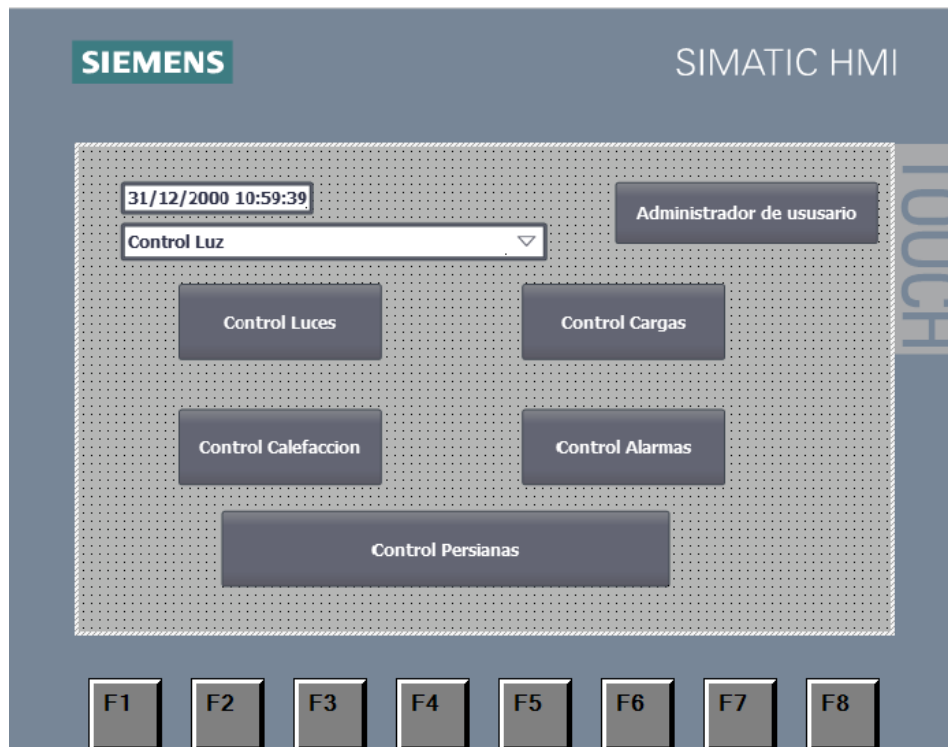




*Ilustración 6.16: Home invitado*

### **6.3.1.2. Ventana Home\_Residente**

La ventana "Home\_Residente" cuenta con todas las opciones de control de la vivienda y tiene acceso exclusivo para aquellos usuarios con permiso de residente.



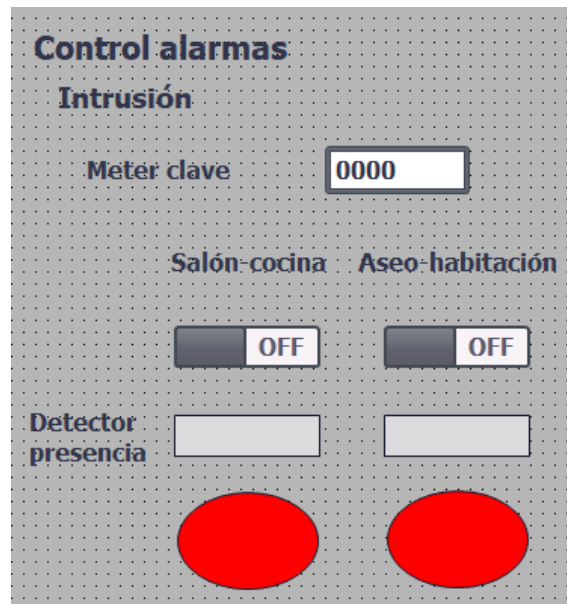
*Ilustración 6.17: Home residente*

## **6.3.2. Ventana Control\_Alarmas**

### **6.3.2.1. Alarma Intrusión**

Para las alarmas de intrusión el usuario ha de armar la alarma introduciendo el código correcto en el HMI (1234), tiene la opción de seleccionar las zonas donde desea activar dicha alarma, en el caso del piso piloto salón-cocina, habitación-aseo o ambas. Una vez activada la alarma, se deja un tiempo de retardo para permitir al usuario ausentarse de la sala correspondiente sin que se dispare la alarma activar.

De la misma manera a partir de que este la alarma activada, una vez detectado movimiento en algunas de las habitaciones con alarma, se inicia ese tiempo de retardo para dejar tiempo al usuario para desactivar la alarma introduciendo la clave. En el caso de tratarse de un intruso que desconoce la clave, pasado el tiempo de retardo, se disparará la alarma hasta que el usuario la desactive volviendo a introducir la clave.

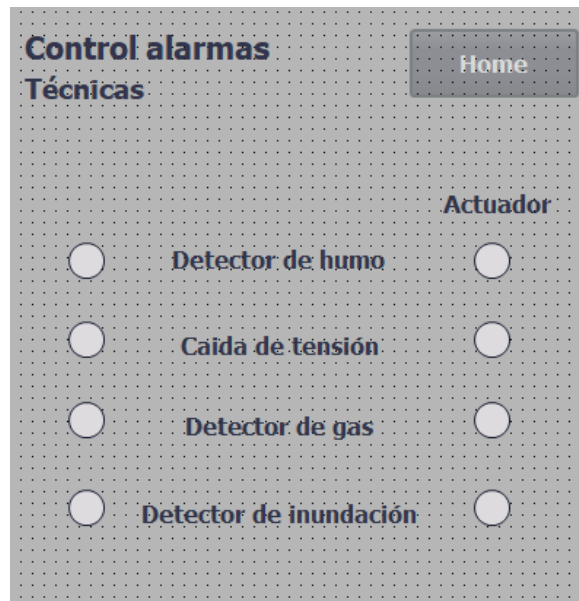


*Ilustración 6.18: HMI alarma intrusión*

### 6.3.2.2. Alarma Técnica

Respecto a las alarmas técnicas, se tienen en cuenta sensores de inundación, gas, humo y caída de tensión. En el caso de la instalación del piso *E-Llar* solo cuenta con la señal de inundación tanto en el baño como en la cocina.

Cada uno tendría su correspondiente actuador por ejemplo en caso de inundación se corta la electroválvula parando así la corriente de agua hasta que se arregle la avería.



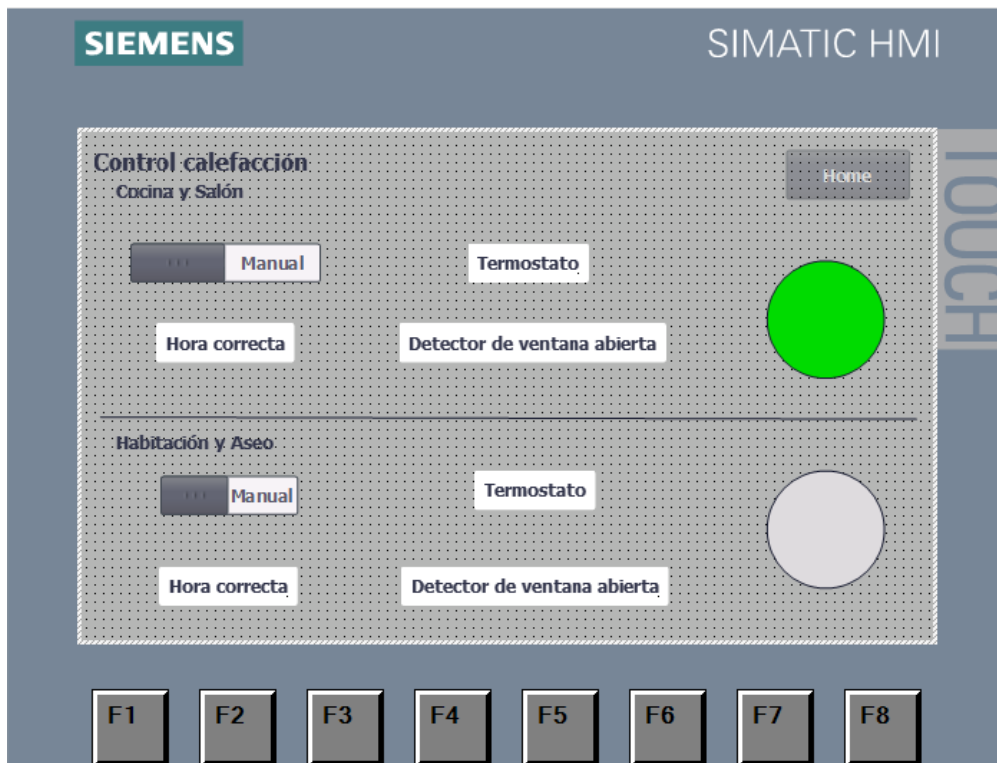
*Ilustración 6.19 HMI alarmas técnicas*

### **6.3.3. Ventana Control\_Calefaccion**

Respecto al HMI para el control de la calefacción, se consideran dos zonas: "cocina-salón" y "habitación-aseo". Cada una cuenta con su propio interruptor para activar o desactivar el modo manual.

En modo manual, se tendrá en cuenta exclusivamente la señal del termostato. El usuario debe introducir una temperatura objetivo en el termostato y, según si esta es mayor o menor que la detectada por su termómetro interno, el termostato enviará un TRUE o un FALSE al PLC.

En caso de funcionar el sistema en modo automático, además del termostato, se considerarán si la hora en la que se ejecuta el programa está dentro del horario marcado y si la ventana se encuentra abierta o cerrada.

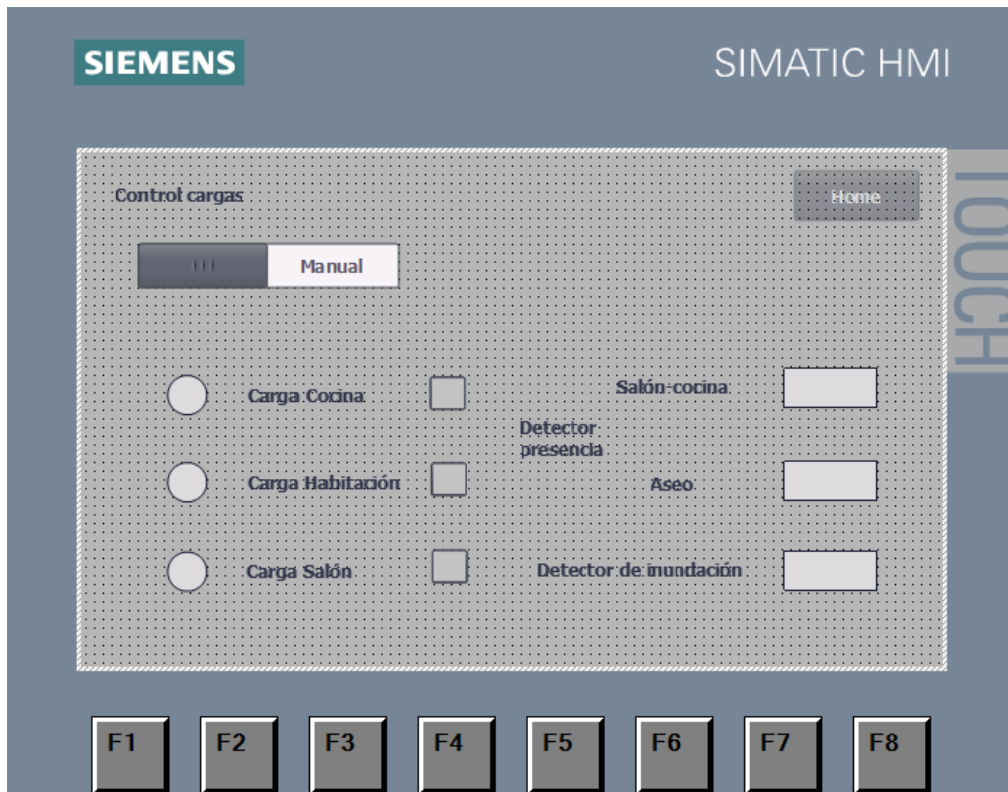


*Ilustración 6.20: HMI calefacción*

### 6.3.4. Ventana Control\_Cargas

El control de cargas dispone de dos modos de operación que el usuario puede seleccionar desde el HMI. En el modo manual, el usuario puede activar y desactivar las cargas directamente desde la pantalla. En el modo automático, se utilizan hasta dos detectores que, dependiendo de su estado, permiten la activación o desactivación de la carga. Por ejemplo, un detector de movimiento puede desactivar la carga correspondiente si no detecta presencia durante un tiempo determinado para ahorrar energía o el detector de inundación puede desactivar las cargas por razones de seguridad.

En el caso del piso piloto, se encuentran con tres cargas: la carga de la cocina, la carga del calefactor y la carga del ventilador de la habitación. Cada una de ellas tiene funcionalidades muy distintas.

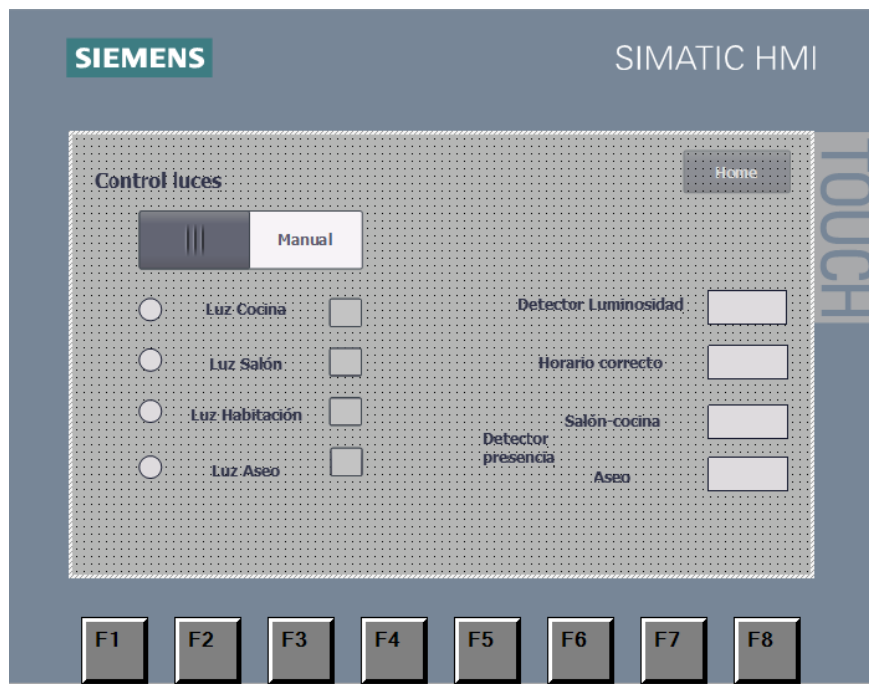


*Ilustración 6.21: HMI control cargas*

### 6.3.5. Ventana Control\_Luces

En el caso de la iluminación, cuenta con dos modos a seleccionar desde el HMI: automático y manual.

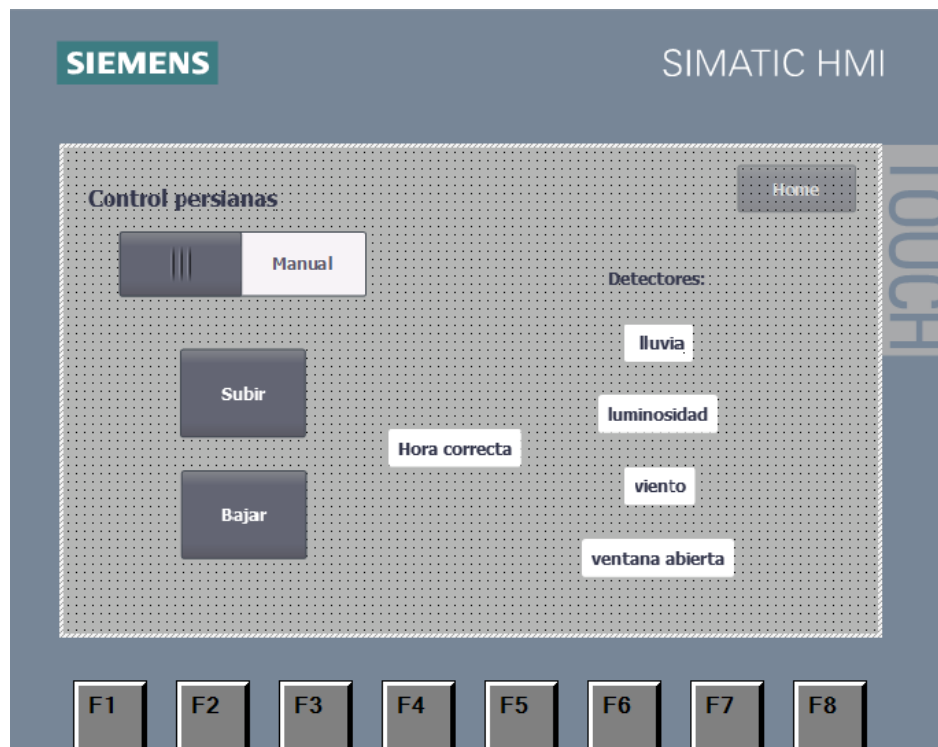
En el modo manual, el usuario puede utilizar los pulsadores instalados en el piso o utilizar los pulsadores “virtuales” del HMI. Por otro lado, si opera en modo automático, se activarán las luces en caso de que no se detecte suficiente luz, no se encuentre en el horario correcto y se detecte presencia. Lógicamente, para contar con estos datos, se requieren los detectores de luminosidad correspondientes y detectores de movimiento tanto en el salón como en el aseo. Respecto al horario, este está seleccionado por defecto de 10 a 21.



*Ilustración 6.22 HMI luces*

### 6.3.6. Ventana Control\_Persianas

Por su parte, las persianas cuentan con modos de ejecución automático y manual, los cuales el usuario puede seleccionar desde el HMI. En el modo automático, las persianas permanecerán subidas siempre y cuando no llueva, no haya viento y haya luz en el exterior. En cualquier otro caso, las persianas se bajarán. Además, se considera un horario de "persiana subida" que, de no cumplirse, hará que las persianas se bajen independientemente de las condiciones meteorológicas.



*Ilustración 6.23 HMI persianas*



# 7. Aplicaciones en IoT

## 7.1. INTRODUCCIÓN

En este capítulo pretende proporcionar una comprensión integral de las aplicaciones del IoT. Se enfoca en la puesta en marcha, el funcionamiento y la implementación en Node-RED.

Para ello se detallan aspectos técnicos y prácticos de la implementación de Node-RED, desde la configuración inicial hasta la creación de un sistema de control remoto y la integración con servicios de comunicación y bases de datos.

Se presentarán las diferentes etapas de la implementación, comenzando con una visión general y avanzando hacia temas más específicos como la adición de usuarios y contraseñas, el control remoto, las comunicaciones S7 y la creación de un dashboard interactivo. También se incluirá una guía práctica para el uso del dashboard, proporcionando al usuario las herramientas necesarias para interactuar con el sistema.

## 7.2. PUESTA EN MARCHA Y FUNCIONAMIENTO

En este apartado se detallan los pasos y componentes necesarios para la implementación y puesta en marcha del módulo SIMATIC IoT2040.

En primer lugar, se requieren los siguientes elementos de hardware:

- El módulo SIMATIC IoT2040 de Siemens,
- Una tarjeta micro SD de entre 8GB y 32GB para operar el sistema operativo Yocto Linux.
- Una estación de ingeniería (PC con al menos Windows 7 que incluya ranura SD y puerto Ethernet).
- Dos cables Ethernet para conectar la estación de ingeniería y el SIMATIC IoT2040 estableciendo conexión SSH y permitiendo la descarga de proyectos.
- Una fuente de alimentación que proporcione entre 9 y 36V.

Respecto al software, se necesita:

- Una imagen de ejemplo del sistema operativo que se debe instalar en una tarjeta SD con Yocto Linux, la cual se puede descargar desde [support.industry.siemens.com](http://support.industry.siemens.com).

Es necesario utilizar PuTTY para permitir el acceso remoto al software del SIMATIC IoT2040 a través de SSH, Telnet o Serial, y Win32 Disk Imager para transferir la imagen a la tarjeta micro SD, recalando que este proceso formateará la tarjeta micro SD.

El primer paso es grabar la imagen del sistema operativo Yocto Linux en la tarjeta micro SD utilizando Win32 Disk Imager. Para ello, se debe abrir el programa, seleccionar el archivo que contiene la imagen del SO, "iot2000-example-image-iot2000.wic", y la tarjeta en la que se pretende grabarla. El proceso tomará unos minutos y al finalizar se habrán creado tres particiones en la tarjeta.



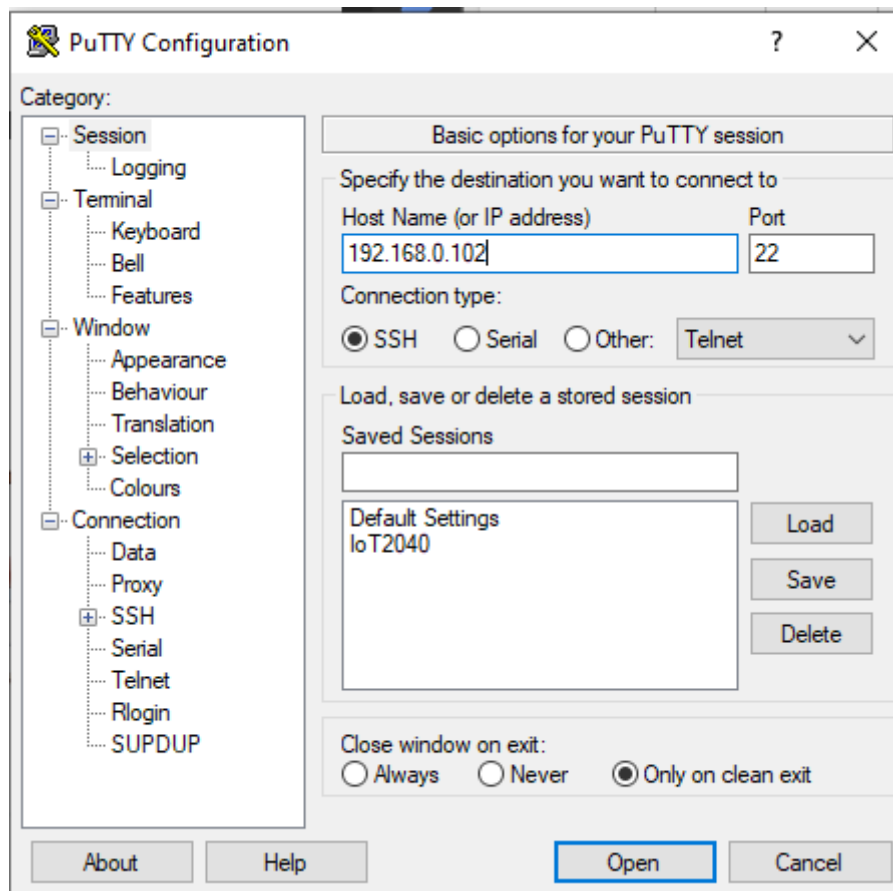
*Ilustración 7.1: Grabar SO*

Una vez grabada la imagen, se puede introducir la tarjeta en el módulo IoT. Tras unos minutos, el módulo estará listo para funcionar, lo cual se indica cuando deja de parpadear el LED etiquetado como SD. Es importante que el IoT2040 esté alimentado durante este proceso.



*Ilustración 7.2: Introducción tarjeta SD*

Para poder acceder al dispositivo, se emplea PuTTY, que establece una conexión SSH para acceder de manera remota al dispositivo. Es necesario que el PC y el dispositivo estén conectados a la misma red. El puerto eth0 del dispositivo tiene la dirección fija 192.168.0.102, por lo tanto, el PC debe tener una IP que siga la estructura 192.168.0.X y ambos deben compartir la máscara de subred 255.255.255.0.



*Ilustración 7.3: PuTTY*

Una vez conectado con el dispositivo, es preciso iniciar sesión como "root". Por defecto, el dispositivo no tendrá contraseña; sin embargo, si se desea implementarla, basta con introducir el comando "passwd" para seleccionar una contraseña.

En cuanto a otros ajustes del aparato, se accede con el comando "iot2000setup". Algunas de las opciones a modificar son las IP de cada puerto Ethernet, si se desea que sea DHCP, y las aplicaciones que se quieren arrancar automáticamente al encender el IoT2040, entre otras. [14]

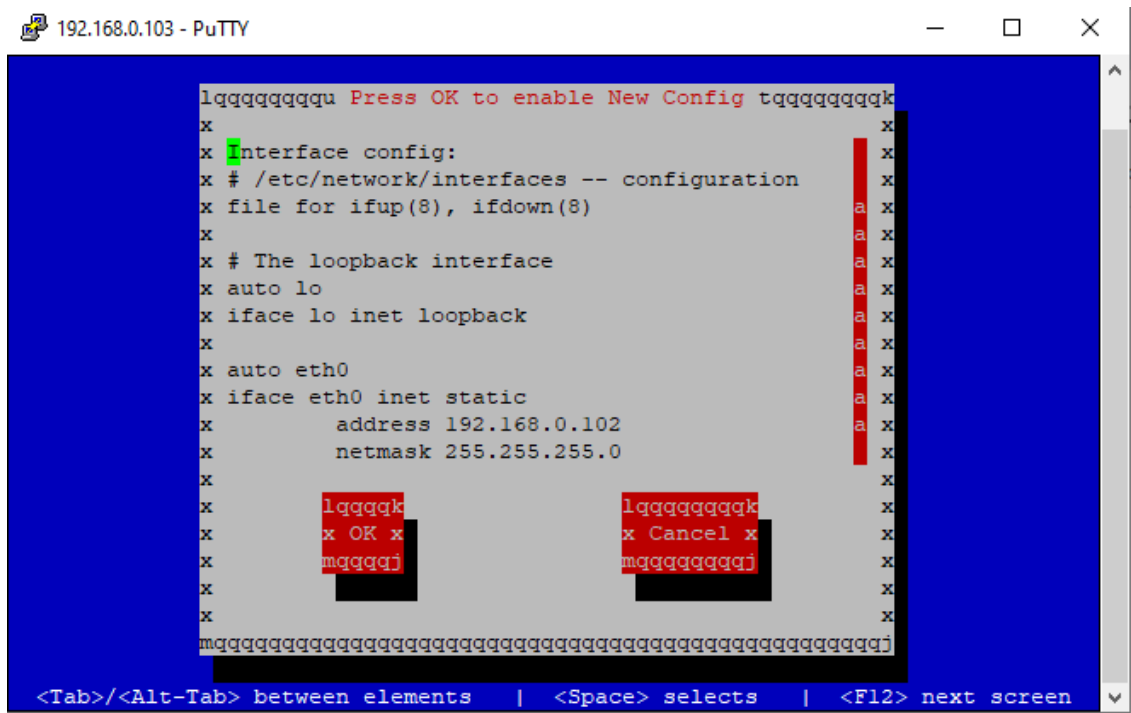


Ilustración 7.4: Setup IoT2040

### 7.3. IMPLMANTACÓN EN NODE-RED

En esta sección se detallan los pasos esenciales para la implementación de funcionalidades en Node-RED, centrando la atención en aspectos específicos y prácticos del desarrollo de sistemas IoT.

Se comenzará con la configuración de usuarios y contraseñas, una medida de seguridad necesaria para proteger el acceso a las aplicaciones desarrolladas. A continuación, se explorarán las posibilidades del control remoto, permitiendo la operación de dispositivos desde cualquier ubicación. La sección también incluye un análisis de las comunicaciones S7.

Además, se revisarán los elementos básicos del dashboard, una interfaz crucial para la visualización y gestión de datos. La configuración de Telegram será abordada para permitir la interacción y notificaciones a través de esta popular plataforma de mensajería. Se explicará también la integración con bases de datos, un componente clave para el almacenamiento y análisis de datos. Finalmente, se presentará un manual de usuario del dashboard, proporcionando instrucciones claras y detalladas para garantizar una experiencia de usuario óptima.

Los códigos presentados a continuación se encuentran en los anexos en formato JSON bajo la denominación e\_llar.json

### 7.3.1. Añadir usuario y contraseña

Para proteger con contraseña el editor y la API de administración de Node-RED, es necesario realizar una configuración en el archivo "settings.js". A continuación, se describe el proceso detallado para llevar a cabo esta configuración.

En primer lugar, acceda al archivo "settings.js". Este archivo se encuentra ubicado en el directorio de Node-RED. Por defecto, en un sistema Windows, la ruta es "C:\Users[NombreDeUsuario].node-red" como se muestra en [Ilustración 7.5](#). Para abrir el archivo, navegue hasta esta ubicación utilizando el explorador de archivos y haga doble clic en el archivo "settings.js" para abrirlo con el Bloc de notas.



Ilustración 7.5: Ruta settings.js

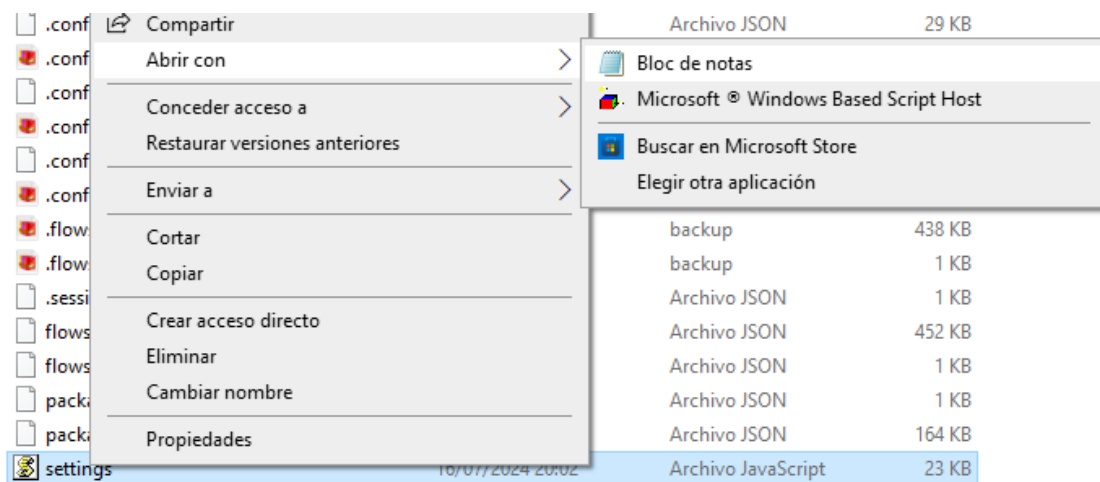


Ilustración 7.6: Abrir con Bloc de notas

Dentro del archivo "settings.js", se ha de buscar la sección que contiene las propiedades para la autenticación del administrador. Esta sección, como muchas otras está comentada por defecto. Para que se ejecute esta sección se ha de descomentar eliminando las dobles barra "/\*".

```

/** To password protect the Node-RED editor and admin API, the following
 * property can be used. See https://nodered.org/docs/security.html for details.
 */
//adminAuth: {
//  type: "credentials",
//  users: [{
//    username: "admin",
//    password: "$2a$08$zZWtXTja0fB1pzD4sHcMyOCMyz2Z6dNbM6t18sJogENOMcxwV9DN.",
//    permissions: "*"
//  }]
//},

```

Ilustración 7.7: Código settings.js

Hecho esto se ha de seleccionar el username nombre de usuario (username) seleccionado. Respecto a la contraseña encriptada se ha de generar introduciendo el comando “*node-red admin hash-pw*” en la terminal de Node-RED. Al introducir el comando te pide que introduzcas la contraseña deseada y te devuelve la contraseña encriptada utilizando el algoritmo bcrypt por motivos de seguridad.

Por último, es recomendable reiniciar Node-RED para que los cambios implementados surjan efecto. Al acceder nuevamente al editor de Node-RED a través de su navegador web, se le pedirá que ingrese el nombre de usuario y la contraseña configurados.

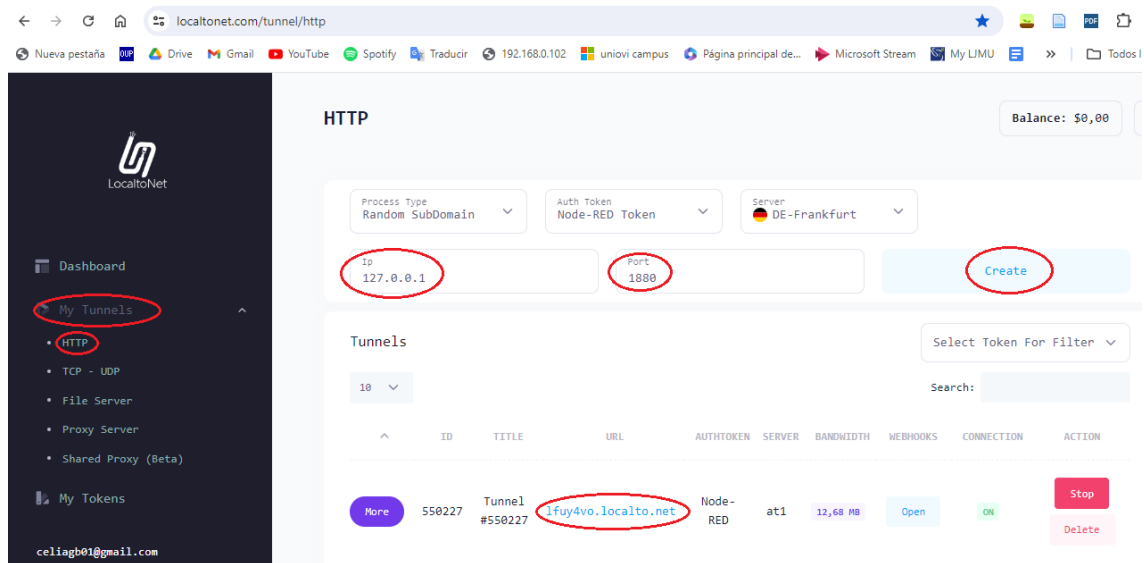


*Ilustración 7.8: Acceso con contraseña*

### 7.3.2. Control remoto

Para habilitar acceso remoto a el portátil utilizado para este proyecto, y por lo tanto a la posibilidad de controlar las instalaciones de manera remota, se empleó un servicio de túneles proporcionado por <https://localtonet.com/>.

En primer lugar, es necesario registrarse en la plataforma de LocaltoNet y acceder a la cuenta. Una vez dentro se ha de navegar la sección de “My Tunnels” → HTTP . Para crear un nuevo túnel se ha de configurar el process type como “Random SubDomain”. Para este proyecto se ha denominado el túnel como “Node-RED Token”, la IP local “127.0.0.1” y el puerto “1880”, que es el puerto predeterminado donde Node-RED se está ejecutando. Como server se ha seleccionado el de “DE-Frankfurt”.



*Ilustración 7.9: Crear túnel nuevo*

Una vez configurado se ha de hacer clic en “Create” para generar el túnel y con el la URL única para acceder remotamente a Node-RED. Para este proyecto interesa acceder al dashboard en lugar de a los flujos de Node-RED ya que lo que interesa es controlar las instalaciones a nivel de usuario. Por lo tanto, en lugar de utilizar el link generado, se le añadirá al final /ui. De buscarse el link según se proporciona en Localnet se accedería a el inicio de sesión por contraseña por lo que, de no poseerse, no sería de gran utilidad. Por esté motivo se implementaron las medidas de seguridad en Node-RED.

Desde la página de LocaltoNet, se puede detener o eliminar el túnel según sea necesario. También incluye la opción "Open" permite abrir directamente el enlace en el navegador para acceder a la instancia de Node-RED de manera remota.

### 7.3.3. Comunicaciones S7

Para interactuar con el PLC de manera sencilla y eficiente, se ha elegido utilizar la librería "node-red-contrib-s7". Es sumamente importante recalcar que se ha utilizado la versión 2.1.0, ya que al implementar la versión más actualizada (3.1.0) se encontraron incompatibilidades con el módulo IoT que impedían funcionar a Node-RED con dichos nodos instalados.

Para instalar dicha librería en la versión adecuada, se requiere introducir el comando desde PuTTY, dado que si se intenta por la vía habitual (desde el propio Node-RED con “Manage Palette”) solo tendrás acceso a instalar la última versión. En primer lugar, se ha de iniciar sesión como root e introducir el comando "npm install node-red-contrib-s7@2.1.0". Tras unos minutos se instalará el paquete y es recomendable reiniciar el módulo IoT.



```

192.168.0.102 - PuTTY
login as: root
root@iot2000:~# npm install node-red-contrib-s7@2.1.0
> usb@1.9.2 install /home/root/node_modules/usb
> node-gyp-build

gyp: Undefined variable napi_build_version in binding.gyp while trying to load binding.gyp
gyp ERR! configure error
gyp ERR! stack Error: `gyp` failed with exit code: 1
gyp ERR! stack   at ChildProcess.onCpExit (/usr/lib/node_modules/npm/node_modules/node-gyp/lib/configure.js:351:16)
gyp ERR! stack   at emitTwo (events.js:126:13)
gyp ERR! stack   at ChildProcess.emit (events.js:214:7)
gyp ERR! stack   at Process.ChildProcess._handle.onexit (internal/child_process.js:198:12)
gyp ERR! System Linux 4.4.302-cip69-st5
gyp ERR! command "/usr/bin/node" "/usr/lib/node_modules/npm/node_modules/node-gyp/bin/node-gyp.js" "rebuild"
gyp ERR! cwd /home/root/node_modules/usb
gyp ERR! node -v v8.17.0
gyp ERR! node-gyp -v v5.0.5
gyp ERR! not ok
npm WARN saveError ENOENT: no such file or directory, open '/home/root/package.json'
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN notsup Unsupported engine for usb@1.9.2: wanted: {"node":">=10.16.0"} (current: {"node":"8.17.0","npm":"6.13.4"})
npm WARN notsup Not compatible with your version of node/npm: usb@1.9.2
npm WARN enoent ENOENT: no such file or directory, open '/home/root/package.json'
npm WARN root No description
npm WARN root No repository field.
npm WARN root No README data
npm WARN root No license field.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: usb@1.9.2 (node_modules/usb):
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: usb@1.9.2 install: `node-gyp-build`
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: Exit status 1

+ node-red-contrib-s7@2.1.0
added 2 packages from 2 contributors and audited 6 packages in 167.24s
found 0 vulnerabilities
  
```

*Ilustración 7.10: Instalación nodos s7*

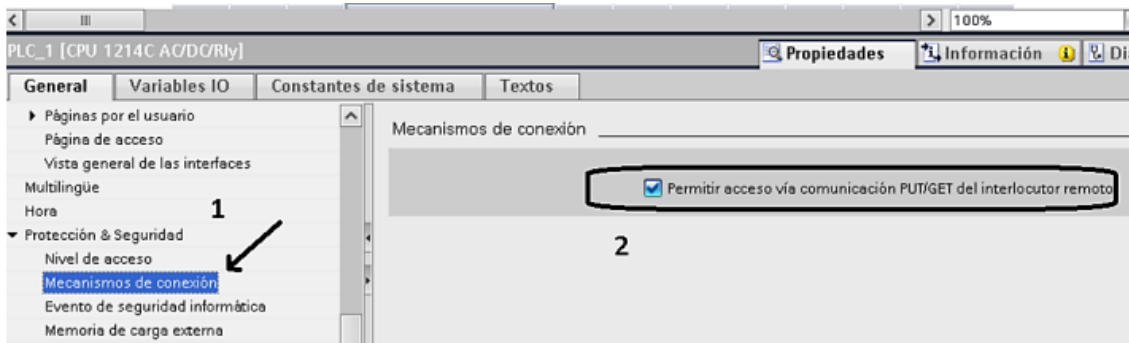
Este paquete cuenta con tres nodos principales. En todos ellos se ha de configurar la dirección del PLC, las variables disponibles, sus direcciones y el tiempo de ciclo para la lectura de las mismas. Cuenta con un parámetro (diff) para retransmitir el estado de las variables exclusivamente en el caso de que estas modifiquen su valor

- S7-in: Se encarga de proporcionar acceso a las variables del PLC. Se puede elegir entre solo una variable, todas las variables (una por mensaje) o directamente todas las variables en un solo mensaje.
- s7 out: escribe en el área de memoria deseado en el PLC
- s7 control: permite un control avanzado del PLC y sus conexiones.



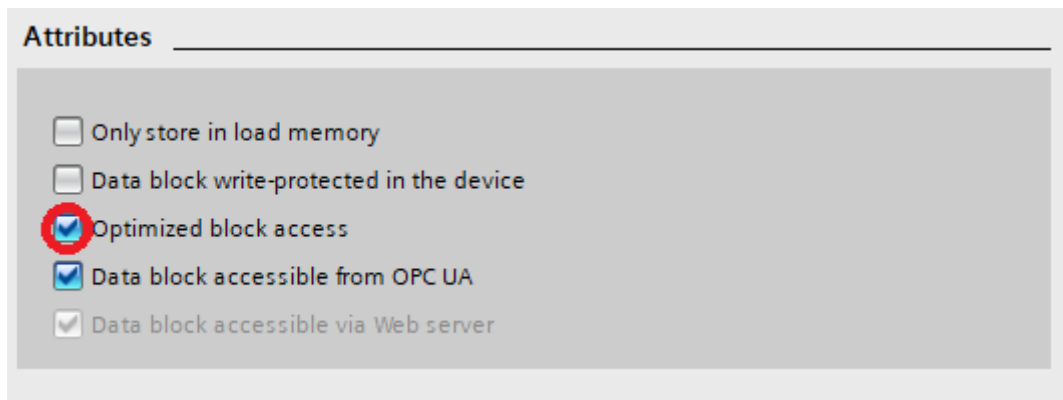
Otro dato a tener en cuenta es que los PLCs más nuevos, como son los S7-1200, trabajan con una versión extendida del protocolo de comunicación S7. Sin embargo, el nodo "node-red-contrib-s7" solo soporta la versión "básica" del protocolo S7. Por lo tanto, se requiere realizar ciertos ajustes desde TIA Portal para permitir el acceso a las variables:

- En la sección "Protección" de las propiedades de la CPU, habilitar la opción "Permitir acceso con PUT/GET".



*Ilustración 7.11: Permitir acceso PUT/GET*

- Desactivar el "acceso optimizado a bloques" para los DBs que se quieren acceder. En caso de estar optimizado el acceso, el usuario no tiene acceso al parámetro "Offset", que nos proporciona las direcciones de las variables. [15] [16]



*Ilustración 7.12: optimized block access*

Para este proyecto se ha considerado que las variables esenciales a comunicar debían ser:

Dirección	Variable en Node-RED	Descripción
DB13,X18.0	HMI_Cocina	Pulsador luz cocina
DB13,X36.0	HMI_Salon	Pulsador luz salón
DB13,X54.0	HMI_Habitacion	Pulsador luz habitación

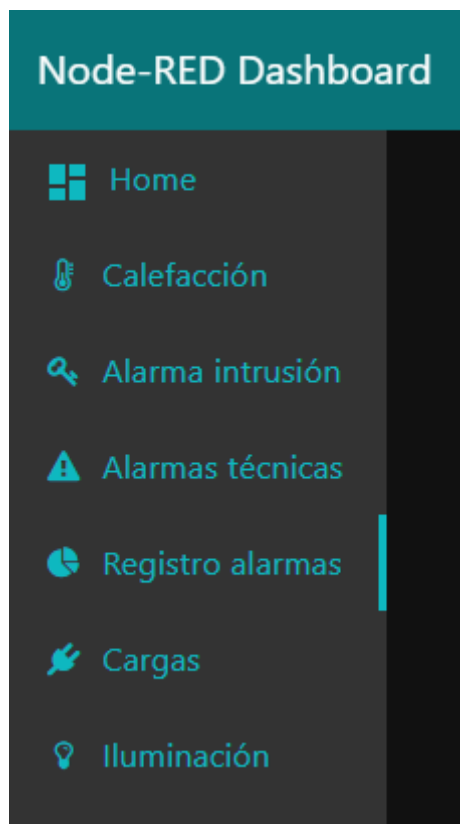
DB13,X72.0	HMI_Aseo	Pulsador luz aseo
DB1,X16.0	HMI_carga_cocina	Pulsador carga cocina
DB1,X34.0	Carga_Ventilador	Pulsador carga ventilador
DB13,X2.1	Luz_Cocina_Salida	Luz cocina
DB13,X20.1	Luz_Salon_Salida	Luz salón
DB13,X38.1	Luz_Habitacion_Salida	Luz habitación
DB13,X56.1	Luz_Aseo_Salida	Luz aseo
DB1,X0.1	Carga_Cocina_Salida	Carga cocina
DB1,X36.1	Carga_Calefactor_Salida	Carga calefactor
DB1,X18.1	Carga_Ventilador_Salida	Carga ventilador
DB14,DWORD0	Cal_Hora_ON	Hora encendido calefacción
DB14,DWORD4	Cal_Hora_OFF	Hora apagado calefacción
DB14,X8.2	Termostato	Termostato
DB14,X8.0	Cal_Auto	Calefacción automática
DB14,X8.3	Cal_Salida	Salida calefacción
DB15,X6.0	Inu_Dis	Disparado alarma inundación
DB15,X6.1	Inu_Actuador	Actuador alarma inundación
DB15,X0.1	Inu_Act	Activación alarma inundación
DB2,X10.0	Intr_Dis_Hab	Disparo alarma intrusión habitación
DB2,X6.0	Intr_Act_Hab	Activación alarma intrusión habitación
DB2,X22.0	Intr_Dis_Coc	Disparo alarma intrusión cocina
DB2,X18.0	Intr_Act_Coc	Activación alarma intrusión cocina
DB1,X52.0	HMI_carga_cal	Carga calefacción
DB1,X34.0	HMI_carga_vent	Carga ventilación
DB13,X74.0	Luz_Auto	Luz automática
DB14,X8.0	Cal_Auto	Calefacción automática
DB1,X54.0	Car_Auto	Carga automática
DB2,INT24	Clave_in	Entrada clave
DB2,X0.0	Det_Pres_Hab	Detección presencia habitación
DB2,X12.0	Det_Pres_Coc	Detección presencia cocina

*Tabla 7.1: Variables PLC en Node-RED*

### 7.3.4. Elementos básicos del dashboard

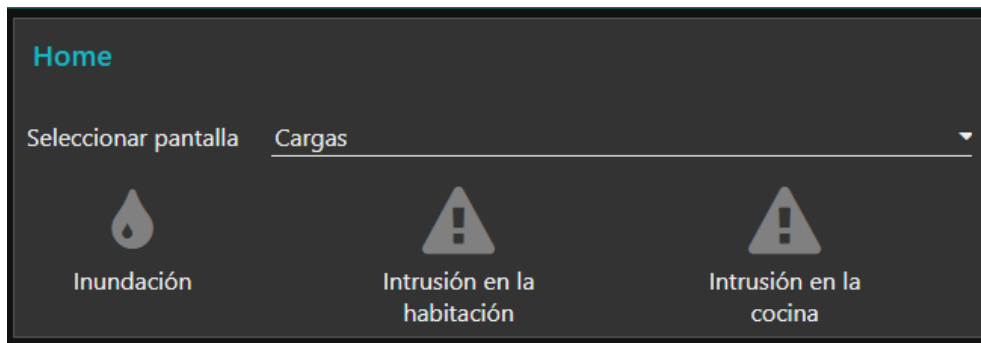
Node-RED ofrece la posibilidad de generar una interfaz de usuario o dashboard, lo que supone una herramienta visual e intuitiva para los usuarios. Aplicado al proyecto del piso piloto *E-llar*, permite a los usuarios controlar y supervisar todos los aspectos del sistema domótico de manera sencilla y eficaz, sin necesidad de conocimientos técnicos avanzados.

Este panel de control se estructura en diversas ventanas, cada una dedicada a una funcionalidad del sistema domótico, facilitando así la interacción del usuario con los diferentes elementos y dispositivos instalados. Desde el menú de navegación, ubicado en el lateral izquierdo del dashboard, se puede acceder a las diferentes secciones del sistema. Este menú incluye las siguientes ventanas:



*Ilustración 7.13: Menú dashboard*

En términos generales, se trata de un "Home" desde el cual se puede utilizar un menú desplegable para navegar a cada una de las otras ventanas y visualizar el estado de las variables que indican situaciones de emergencia, como inundación e intrusión.



*Ilustración 7.14: Home dashboard*

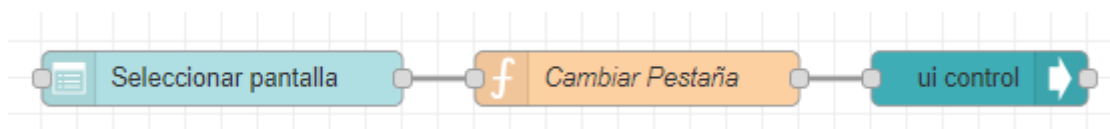
El resto de las ventanas están enfocadas en cada una de las funcionalidades disponibles en el piso piloto: calefacción, alarmas de intrusión y técnicas, control de cargas e iluminación. Cada una se identifica intuitivamente por los iconos que la acompañan.

Por último, también se cuenta con una ventana "Registro de alarmas" donde, como su nombre indica, se accede a un registro de las instancias en las que se disparan las alarmas, las cuales se visualizan en la ventana "Home".

En las siguientes secciones, se detallarán los elementos incluidos en cada una de estas pantallas, explicando su propósito y su forma de uso dentro del sistema para un cómodo, intuitivo y completo control de la vivienda.

### 7.3.4.1. Navegador de pantallas

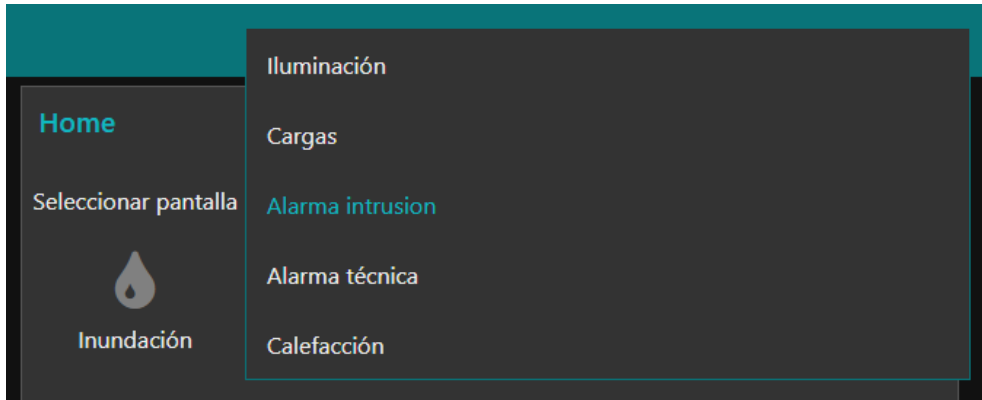
Para permitir al usuario navegar por las diferentes pantallas disponibles en el dashboard, se utiliza un objeto dropdown desde el que el usuario puede elegir entre diferentes pantallas temáticas acordes a las funcionalidades del piso piloto como "Iluminación", "Cargas", "Alarma intrusión", "Alarma técnica" y "Calefacción".



*Ilustración 7.15: Selector de pantallas del dashboard*

El dropdown se encuentra ubicado en la pantalla "Home" que es a la que accedería primero el usuario al acceder al dashboard. Al seleccionar alguna de las opciones, envía al usuario a la pantalla correspondiente. Para lograr esto se utiliza un nodo función capaz de interpretar la selección realizada en el nodo `ui_dropdown` y transformar esta información en un comando específico que indica la pestaña de la interfaz que debe ser mostrada. El código dentro de este nodo evalúa el valor del mensaje (`msg.payload`) y asigna el nombre correspondiente de la pestaña a mostrar. Por ejemplo, si el usuario selecciona "iluminación", el nodo asignará `{ "tab": "Iluminación" }` a `msg.payload`. Esta transformación asegura que la navegación de la interfaz responda dinámicamente a la elección del usuario. El resultado se muestra en [Ilustración 7.16](#).

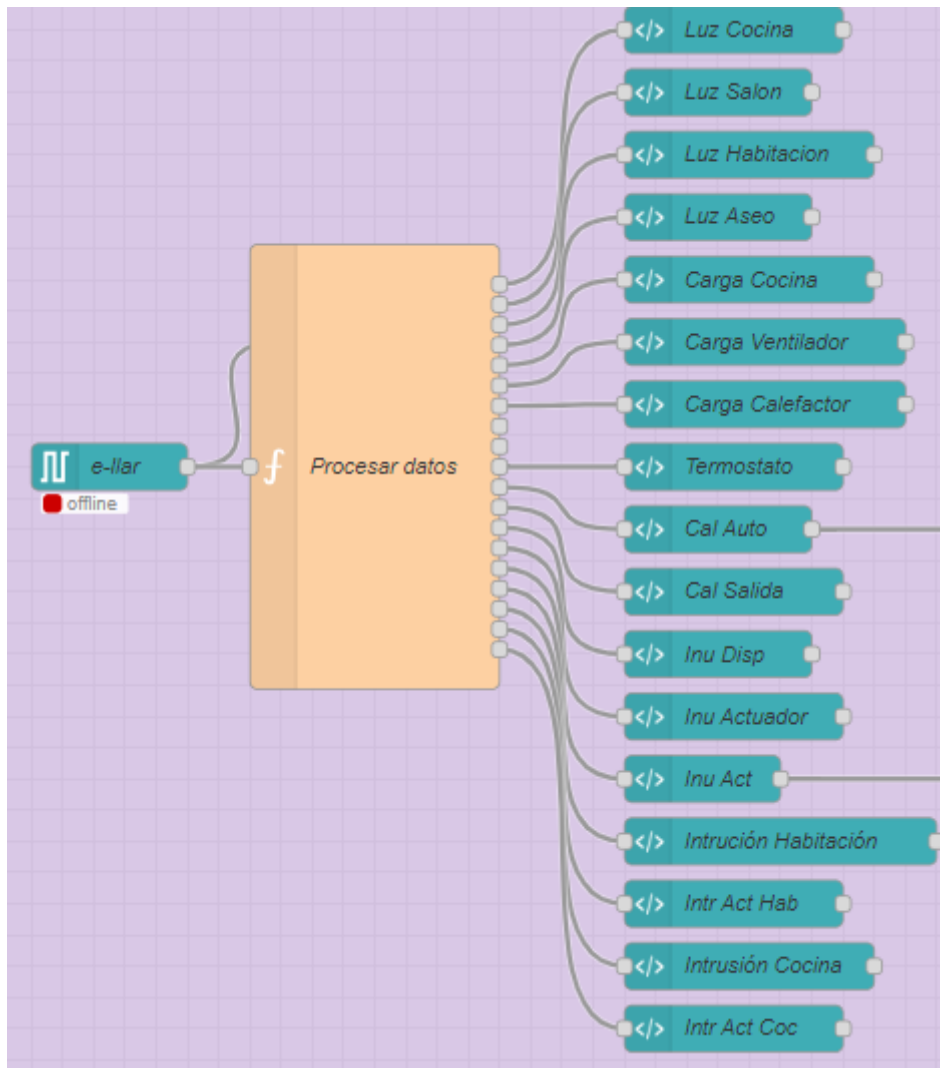
Por último, al recibir el mensaje en el que se especifica la tab deseada, el nodo `ui_control` ejecuta el comando necesario para cambiar a la pestaña especificada en la interfaz de usuario. Este nodo actúa como un controlador que aplica los cambios de visualización solicitados de manera inmediata.



*Ilustración 7.16: Dropdown del Dashboard*

### 7.3.4.2. Iconos

Para facilitar el monitoreo en tiempo real de diversos dispositivos, se han incluido en el dashboard una serie de iconos representativos de las variables y sensores del sistema. En primer lugar, se utiliza un nodo `s7 in` para recopilar el estado de las variables del PLC. Este nodo debe estar configurado con la IP específica del dispositivo (192.168.0.10) y está ajustado para recibir información solo cuando se modifica el estado de alguna de las variables.

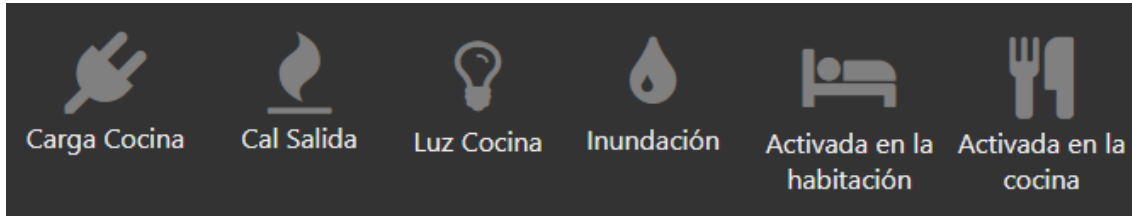


*Ilustración 7.17: Flujo iconos dashboard*

Los datos recogidos son procesados por una función encargada de descomponer el payload recibido y crear múltiples mensajes para diferentes dispositivos. Por ejemplo, esta función procesa datos sobre las luces y cargas en la cocina, salón, habitación y aseo, así como otras variables como el estado del termostato, la hora de encendido y apagado de la calefacción, y diversos actuadores e indicadores.

Cada salida del nodo de función se conecta a los nodos de plantilla correspondientes para actualizar la interfaz de usuario en tiempo real. En el dashboard, cada nodo de plantilla se manifiesta como un icono representativo, lo que permite un análisis intuitivo y rápido del estado de las instalaciones. está configurado con atributos específicos como width, height, y order para organizar su posición y tamaño en la interfaz de usuario. Los nodos también están agrupados según su funcionalidad en las distintas pantallas para una mejor organización y visualización en el sistema de control. En concreto, los iconos cambian de gris a un color representativo según el estado del booleano correspondiente. Por ejemplo, si una luz está encendida, su icono correspondiente pasa a estar amarillo.

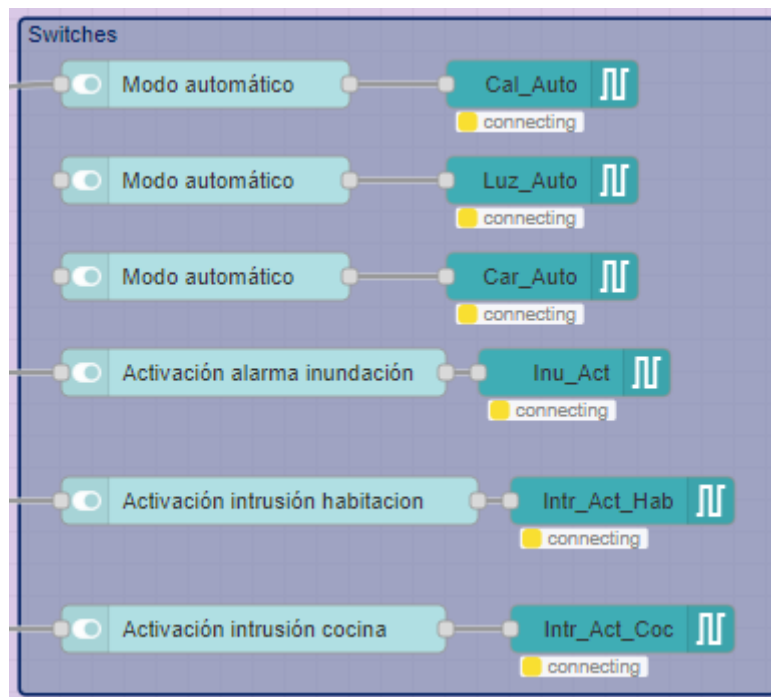
Esta implementación asegura que el usuario pueda visualizar de manera clara y efectiva el estado de todos los dispositivos monitorizados, facilitando la gestión y el control del sistema.



*Ilustración 7.18: Ejemplo íconos dashboard*

### 7.3.4.3. Switches

Algunos de los displays mencionados en la sección anterior incluyen además un switch en el dashboard para modificar su estado. Esto es aplicable al modo automático de las cargas y las luces, así como la activación y desactivación de las alarmas. Para ello, Node-RED cuenta con un nodo específico del paquete dashboard que genera un interruptor preparado para enviar "true" o "false" en función de la selección del usuario en el dashboard.



*Ilustración 7.19: Flujo switches dashboard*

Cada uno de los nodos switch está conectado a un nodo "S7-out" configurado para enviar directamente al PLC el estado de la variable correspondiente. Esto permite que cualquier cambio en el switch se refleje inmediatamente en el sistema, asegurando un control ágil y preciso de las funciones automáticas y las alarmas del sistema domótico.

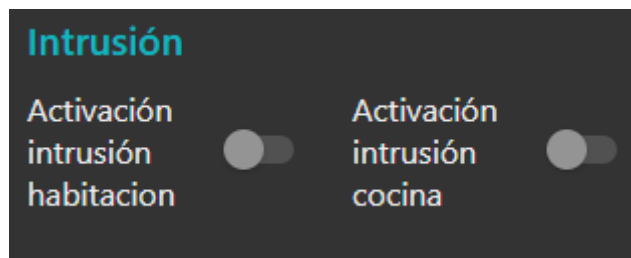


Ilustración 7.20: Ejemplos switches dashboard

#### 7.3.4.4. Pulsadores:

Para facilitar la gestión y el control de las luces y cargas del sistema domótico, se ha implementado un grupo de pulsadores en el dashboard. Se han configurado cuatro nodos "ui\_button", cada uno correspondiente a una estancia específica del piso piloto (cocina, salón, habitación y aseo), así como tres nodos adicionales para controlar las cargas (cocina, ventilador y calefactor). Cada botón está personalizado con etiquetas, colores e iconos específicos para cada área, lo que facilita su identificación y uso. En caso de duda, al pausar el ratón sobre el botón se genera un tooltip explicativo, como se observa en el pulsador del salón en la [Ilustración 7.22](#).

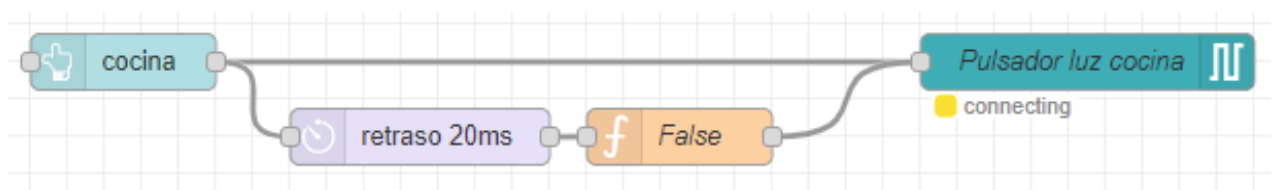


Ilustración 7.21: Flujo pulsadores dashboard

Cuando se pulsa un botón en el dashboard, se envía una señal positiva a la variable pulsador HMI correspondiente. Dado que se trata de un pulsador, es necesario que esta variable vuelva a cero inmediatamente después de recibir la señal positiva para estar lista para la próxima activación. Para lograr esto, se implementa una segunda rama en el flujo, que incluye un nodo "function" y un nodo "delay". Estos nodos restablecen la variable a cero 20 milisegundos después de que se pulse el botón.

Finalmente, para enviar la variable al PLC de manera inmediata, se utiliza un nodo "s7-out" programado para modificar exclusivamente la variable correspondiente. Este nodo garantiza que el estado de la variable se actualice en el PLC en tiempo real, permitiendo un control eficiente y preciso del sistema domótico.

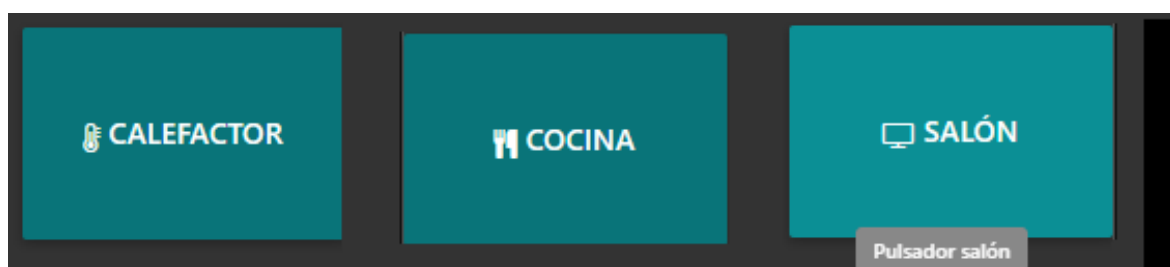


Ilustración 7.22: Ejemplos botones dashboard



### 7.3.4.5. Entradas de texto

Para la gestión de horarios y claves en el sistema domótico, se ha configurado un grupo de nodos en el dashboard que permite a los usuarios introducir valores específicos. En concreto se puede seleccionar El horario de funcionamiento de la calefacción en automático y la clave para activar y desactivar la alarma de intrusión.



*Ilustración 7.23: Flujo introducir cifras*

Para implementarlo se emplea un nodo "ui\_text\_input" que permite a los usuarios introducir la cifra correspondiente. En el caso de la hora de encendido y apagado de la calefacción, acepta el formato de hora HH. Por su parte para la clave se ha de introducir una cifra de 4 dígitos.

Al introducir un dato el usuario, se envía a un nodo función que tras comprobar que está en el formato correcto, los transforma en un DWORD para asegurar que son comprensibles para el PLC al que se le envían los datos desde un nodo s7-out configurado para modificar cada una de las variables que corresponda.

En la se muestra que la clase que introduce el usuario desde el dashboard se recibe directamente en TiaPortal en el caso de estar en el formato correcto. Que en el caso de la variable clave\_IN, son cuatro cifras

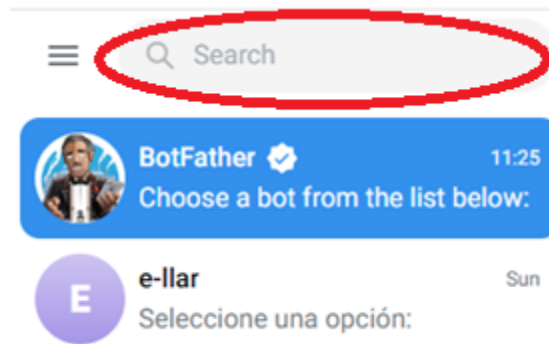


*Ilustración 7.24: Clave\_IN desde el dashboard*

## 7.3.5. Implementación de mensajes con Telegram

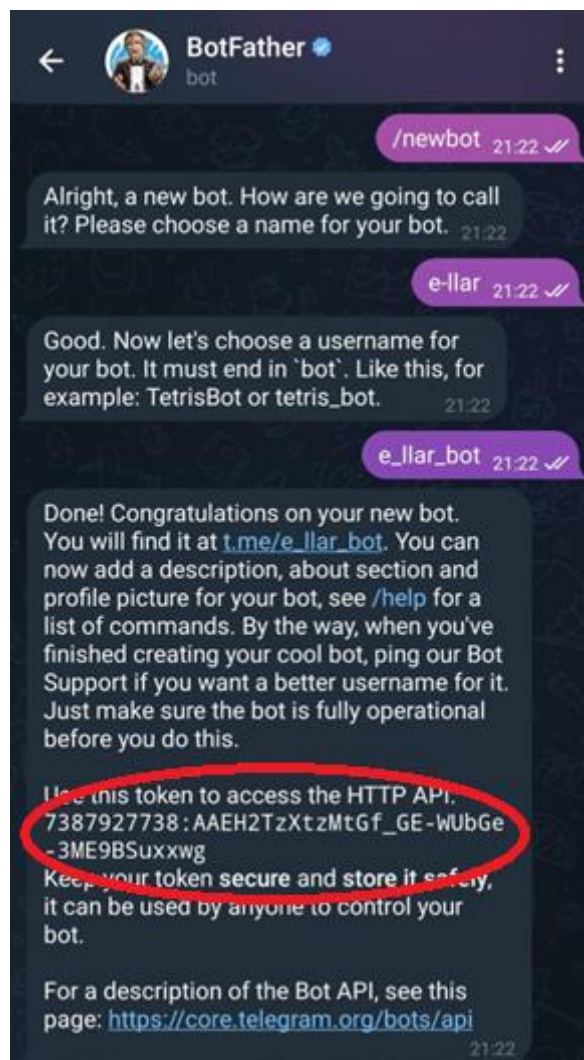
### 7.3.5.1. Crear el Bot :

En primer lugar, se ha de iniciar un chat con el BotFather al que se encontrar directamente buscándolo en el buscador de Telegram ([Ilustración 7.25](#)) Se trata del Bot oficial de Telegram para facilitar la creación y gestión de otros bots.



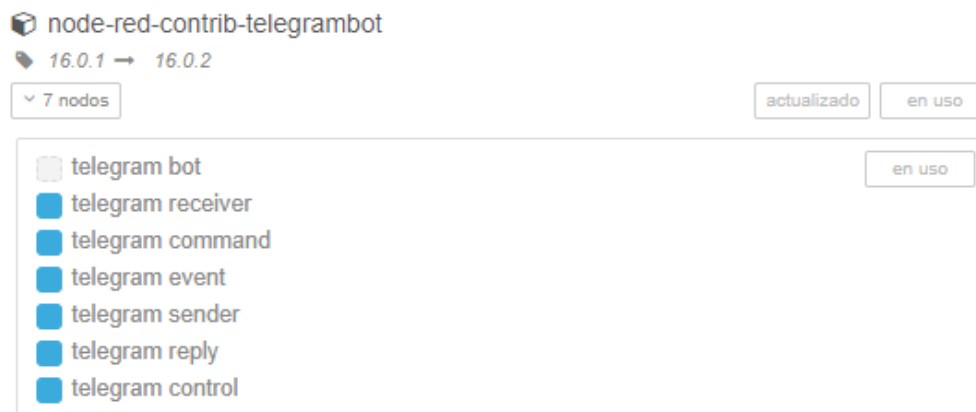
*Ilustración 7.25: Contactar con BotFather*

Empleando el comando “/newbot” se inicia el proceso de registro. Posteriormente se pedirá al usuario un nombre para el Bot y un nombre de usuario terminado en “bot” que ha de ser único. Una vez hecho esto, el BotFather generará un token de acceso, que es una cadena de caracteres única utilizada para autenticar el bot y permitirle interactuar con la API de Telegram.



*Ilustración 7.26: Creación del Bot*

Por otra parte, en Node-RED se ha de instalar el paquete de nodos node-red-contrib-telegrambot que dará acceso a nodos como “telegram receiver” para recibir mensajes y “telegram sender” para enviar respuestas. Es esencial es su configuración introducir el token dado en [Ilustración 7.26](#) para interactuar concretamente con el bot que hayamos creado.

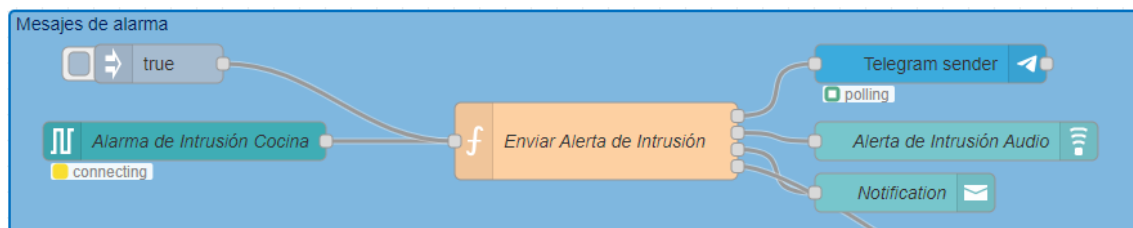


*Ilustración 7.27: Paquete node-red-contrib-telegrambot*

### 7.3.5.2. Mensajes aviso de alarmas:

Una de las funcionalidades clave a las que nos da acceso el uso de Telegram es la capacidad de enviar alertas por mensaje a los residentes en caso de dispararse alguna de las alarmas, tanto de intrusión como de inundación. Este proceso es fundamental para garantizar una respuesta rápida ante eventos de seguridad.

Por ejemplo, para detectar el evento de una intrusión en la zona de la cocina-salón en sí se utiliza un nodo s7-in configurado para monitorear la variable “Intr\_Displ\_Coc” en el caso de la variable denominada “DB\_Alarmas\_Intrusion.”Cocina-Salon”.Disparada” en el PLC Siemens SIMATIC S7-1214. Esta variable es un indicador booleano que se activa (pasa a TRUE) cuando se detecta una intrusión en la cocina-salón. El nodo está configurado para enviar una señal solo en el caso de que se detecte un cambio en la señal monitoreada



*Ilustración 7.28: Flujo mensaje alarma*

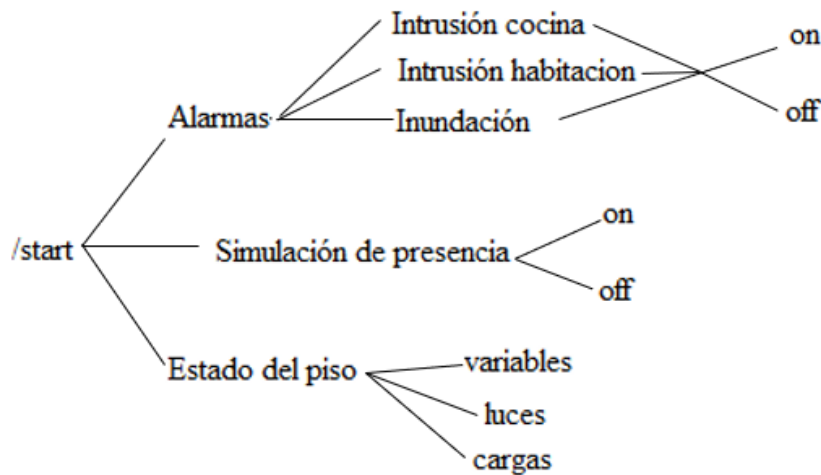
Como se observa en la [Ilustración 7.28](#) el nodo s7-in está conectado a un nodo function que evalúa el estado de la variable, msg.payload y determina si debe enviarse un mensaje de alerta, que en este caso será ”Alerta de intrusión en la cocina !”. Para una

correcta comunicación es indispensable introducir el chatId. La función retorna: `return { payload: { "chatId": chatId, "type": "message", "content": message } }`.

Por último, el nodo función está conectado a un nodo “telegram sender” utiliza el token de autenticación del bot para conectarse a la API de Telegram y enviar el mensaje al chat ID especificado. También se genera un aviso sonoro desde el dashboard con un nodo audio out y salta una notificación.

### 7.3.5.3. Menú interactivo:

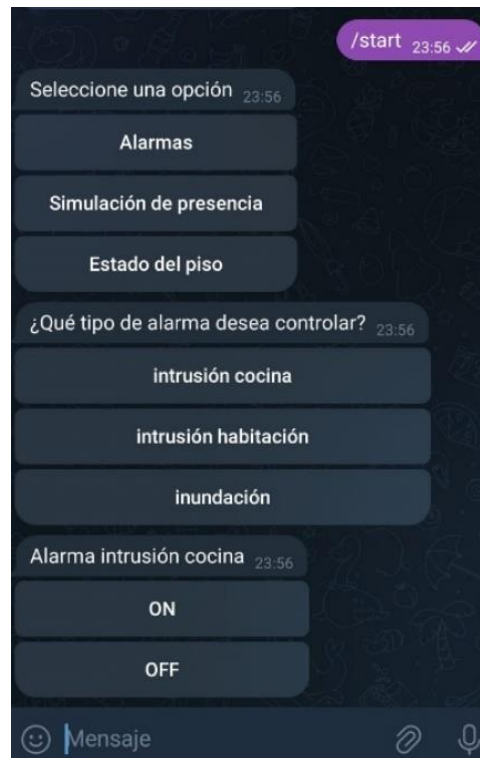
A través de Telegram es posible crear un menú interactivo. Para este proyecto se ha creado una serie de menús diseñados para gestionar y monitorizar las variables y estados de los dispositivos del piso piloto.



*Ilustración 7.29: Esquema menús Telegram*

El punto de inicio del sistema de menús es el comando "/start". Al ingresar este comando en el Bot de Telegram, se despliega el menú principal que ofrece tres opciones fundamentales:

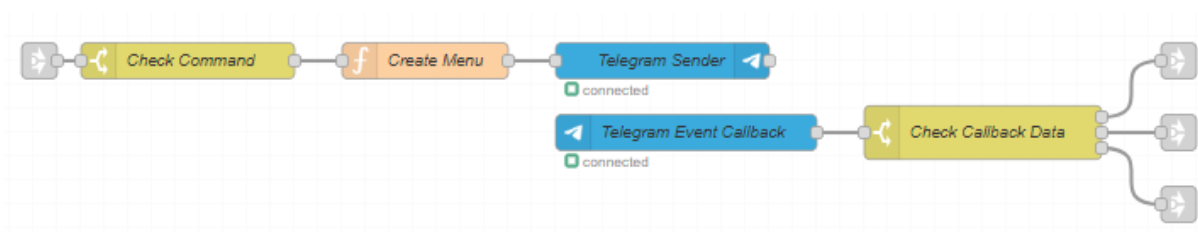
1. **Alarmas:** permite generar la orden de activación de los tres tipos de alarmas disponibles en el *E-Llar*: intrusión cocina, intrusión habitación e inundación.



*Ilustración 7.30: Menú alarmas Telegram*

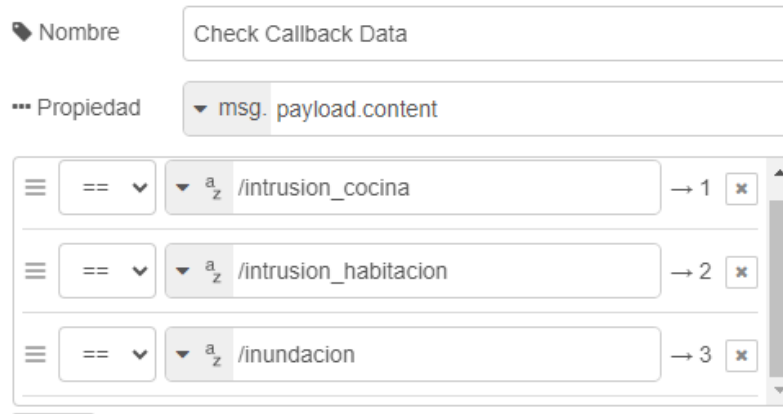
Para generar estos menús en primer lugar se ha de emplear un nodo “switch” que verifica si el mensaje recibido contiene el comando “/alarmas”, esto se debe a que seleccionar la opción alarmas en Telegram equivale a pulsar esa opción en el menú generado. Este nodo asegura que el flujo se active solo cuando se reciba este comando específico.

Posteriormente se genera otro submenú empleando el nodo método “inline\_keyboard” de Telegram. Este menú permite al usuario seleccionar entre los diferentes tipos de alarmas: “intrusión cocina”, “intrusión habitación” e “inundación”. Las opciones se envían de vuelta al usuario a través de un nodo “telegram sender”.



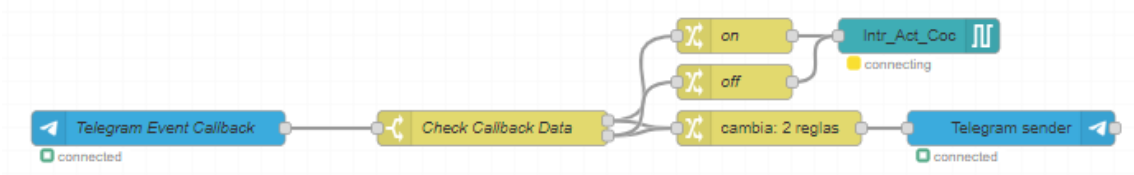
*Ilustración 7.31: Flujo menú alarmas*

Al seleccionar el usuario una opción en el menú de alarmas, se genera un evento “callback\_query”. Este evento es capturado por un nodo “telegram event”, que pasa la información al siguiente nodo “switch” para verificar qué opción fue seleccionada.



*Ilustración 7.32: Nodo switch alarmas*

Dependiendo de la opción seleccionada, el flujo se redirige a través de diferentes nodos "link out". Cada nodo corresponde a una opción específica del menú. De manera similar a la creación del menú alarmas. Para opción seleccionada, se crea un submenú utilizando otro nodo de función. Este submenú permite al usuario activar o desactivar la alarma correspondiente. Las opciones "ON" y "OFF" se envían al usuario a través de otro nodo "telegram sender".



*Ilustración 7.33: Flujo intrusion\_cocina*

De manera análoga un switch comprueba que le llega o "on\_cocina" o "off\_cocina" (en el ejemplo de la [Ilustración 7.33](#)). Y en función de la orden seleccionada en el menú de Telegram, se utiliza un nodo "change" para modificar el payload del mensaje a "true" (activado) o "false" (desactivado). Está orden se hace llegar al PLC mediante un nodo "s7 out".

Paralelamente, se envía una notificación al usuario a través de un nodo "telegram sender", informándole si la alarma ha sido activada o desactivada.

Para cada una de las alarmas se cuenta con un flujo similar pero adaptado a la alarma que corresponda.

2. **Simulación de presencia:** se emplea para activar y desactivar el modo simulación de presencia que enciende alternamente las luces cada media hora para simular la presencia de una persona en la vivienda con el objetivo de disuadir posibles intrusos.

Ya sea por recibir el comando /presencia o a través del menú se genera un submenú de manera similar al de alarmas, identificando el comando desde un

nodo "switch". En el que las opciones son ON y OFF y se envía al usuario mediante un "telegram sender".

Mediante un nodo "callback\_query" del Bot de Telegram recoge la selección del usuario (ON o OFF). Esta selección se procesa para activar o desactivar la simulación. Cuando se selecciona "ON", se activa un intervalo que simula la presencia encendiendo y apagando luces en varias habitaciones a intervalos regulares, cada 20 minutos, y cada luz se apaga después de 5 minutos de haber sido encendida. El control de las luces se gestiona mediante nodos "s7 out", que envían las señales apropiadas a los dispositivos.

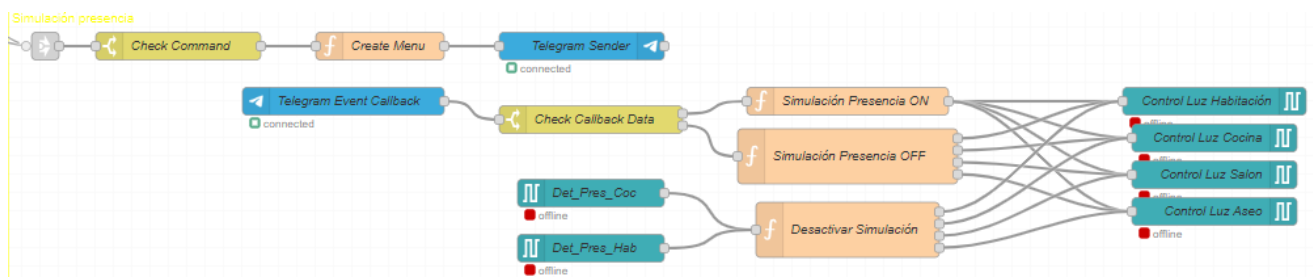


Ilustración 7.34. Flujo simulación de presencia

Si se selecciona "OFF", se detiene el intervalo y todas las luces controladas por la simulación se apagan. Esto se asegura mediante un nodo de función que limpia el intervalo y envía señales de apagado a todas las luces. También se detendrá el modo simulación de presencia en el caso de detectarse presencia real en alguna de las estancias.

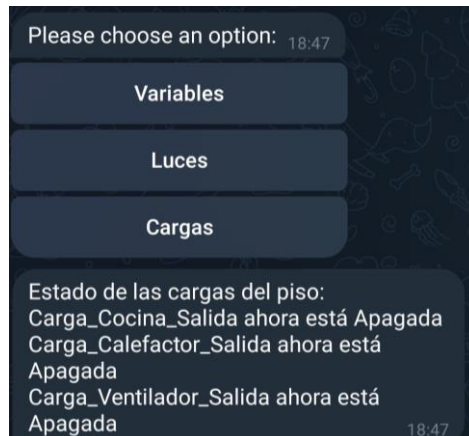


Ilustración 7.35: Menú simulación presencia Telegram

3. **Estado del piso:** esta opción se emplea para visualizar todas las variables del PLC disponibles en Node-RED. Estas variables incluyen el estado de distintos sensores y actuadores, proporcionando una visión completa del funcionamiento del sistema de automatización. Desde este submenú hay tres opciones:
  - Variables: proporciona el estado de todas las variables
  - Luces: informa del estado de las luces del piso.



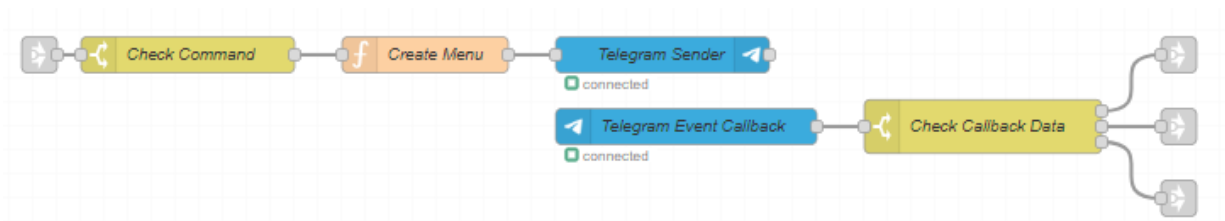
- Cargas: informa del estado de las cargas del *E-Llar*.



*Ilustración 7.36: Menú estado piso*

Para generar estos menús en primer lugar se ha de emplear un nodo switch que verifica si el mensaje recibido contiene el comando `"/comandos"`, esto se debe a que seleccionar la opción alarmas en Telegram equivale a pulsar esa opción en el menú generado. Este nodo asegura que el flujo se active solo cuando se reciba este comando específico.

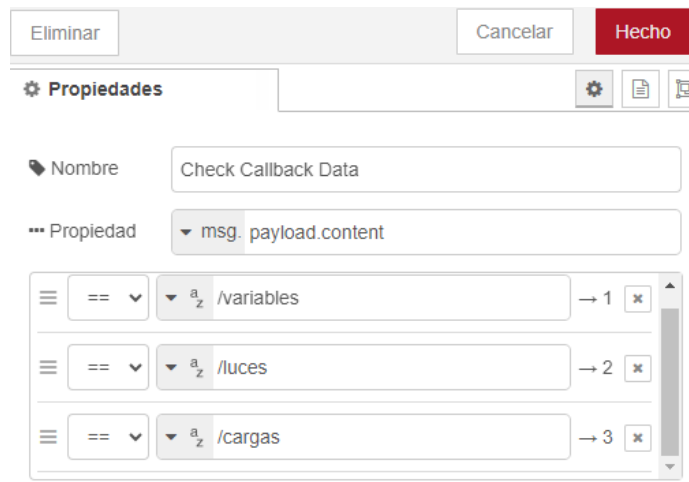
Posteriormente se genera otro submenú empleando el nodo método `"inline_keyboard"` de Telegram. Este menú permite al usuario seleccionar entre las diferentes variables a visualizar: `"variables "`, `"luces"` y `"cargas"`. Las opciones se envían de vuelta al usuario a través de un nodo `"telegram sender"`.



*Ilustración 7.37: Flujo menú alarmas*

Al seleccionar el usuario una opción en el menú de alarmas, se genera un evento `"callback_query"`. Este evento es capturado por un nodo `"telegram event"`, que pasa la información al siguiente nodo `"switch"` para verificar qué opción fue seleccionada.

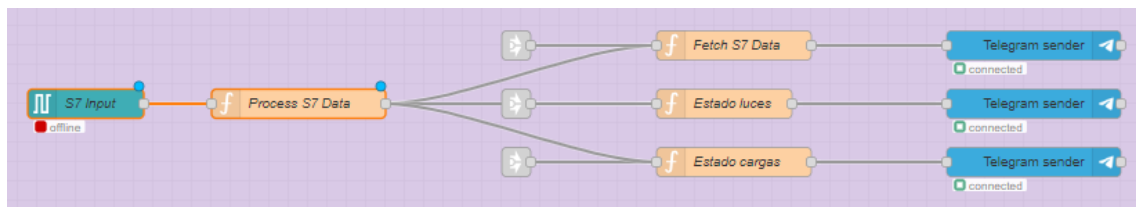




*Ilustración 7.38: Nodo switch estado piso*

Dependiendo de la opción seleccionada, el flujo se redirige a través de diferentes nodos "link out". Cada nodo corresponde a una opción específica del menú. De manera similar a la creación del menú estado del piso. Para opción seleccionada, se crea un submenú utilizando otro nodo de función. Este submenú permite al usuario visualizar de qué variables concretas pretende visualizar el estado. Una vez seleccionado el Bot responderá con un mensaje con la información solicitada.

Para lograr esto en primer lugar se ha de emplear un nodo "function" que recoja los datos del PLC cada vez que se produce un cambio, garantizando así que los datos proporcionados al usuario son en tiempo real.



*Ilustración 7.39: Flujo visualizar datos*

Por último, dependiendo de la opción seleccionada por el usuario, se genera un mensaje en el formato adecuado (en lugar de Luz\_Habitacion\_Salida=TRUE, envía la luz de la habitación está encendida) y se envía con un nodo "telegram sender".

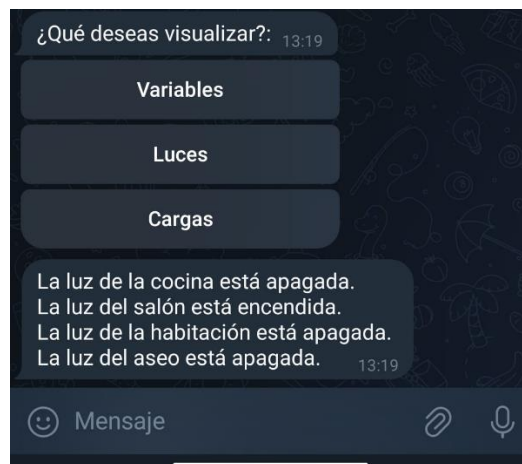


Ilustración 7.40: Visualizar estado luces

### 7.3.5.4. Mensajes predeterminados:

A modo de atajo, en lugar de los menús se pueden emplear ciertos comandos para acceder directamente a las funcionalidades que ofrece el menú. Para ello en primer lugar se ha de incluir un nodo "telegram receiver" encargado de recibir los mensajes enviados por los usuarios al Bot través de Telegram. Cuando un mensaje es recibido, este nodo lo pasa al siguiente nodo del flujo para su procesamiento.

Para ello se emplea un nodo "switch" que evalúa el contenido del mensaje recibido, identificando el comando específico enviado por el usuario, a excepción de una de las opciones que en caso de contener el mensaje del usuario la palabra "opciones", se envía un mensaje explicando todos los comandos disponibles. Para ello se emplea un nodo "function" junto con un "telegram sender".

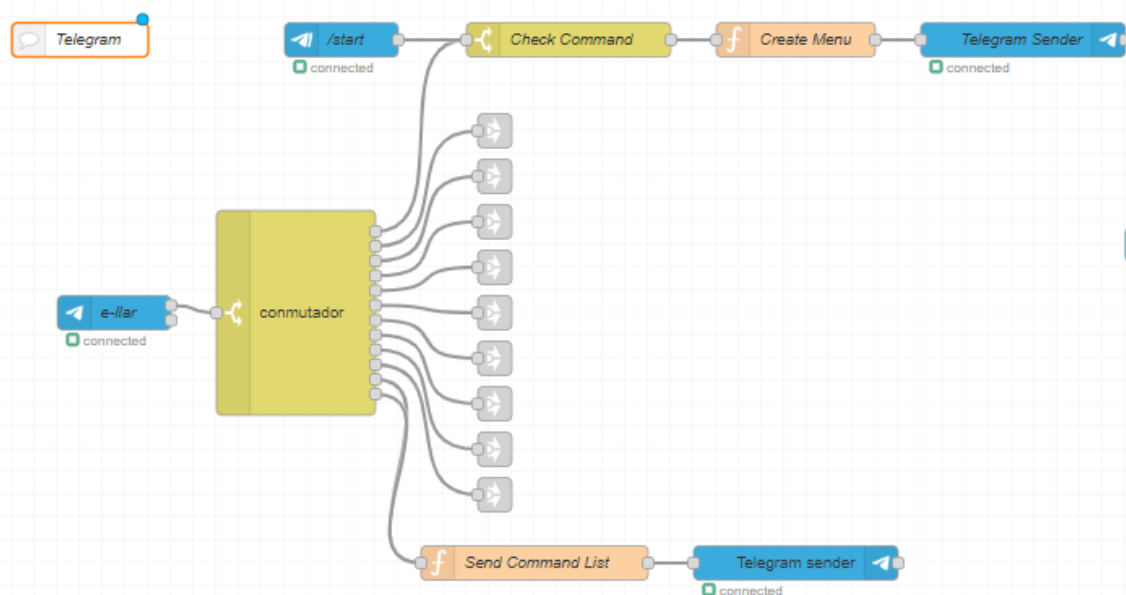
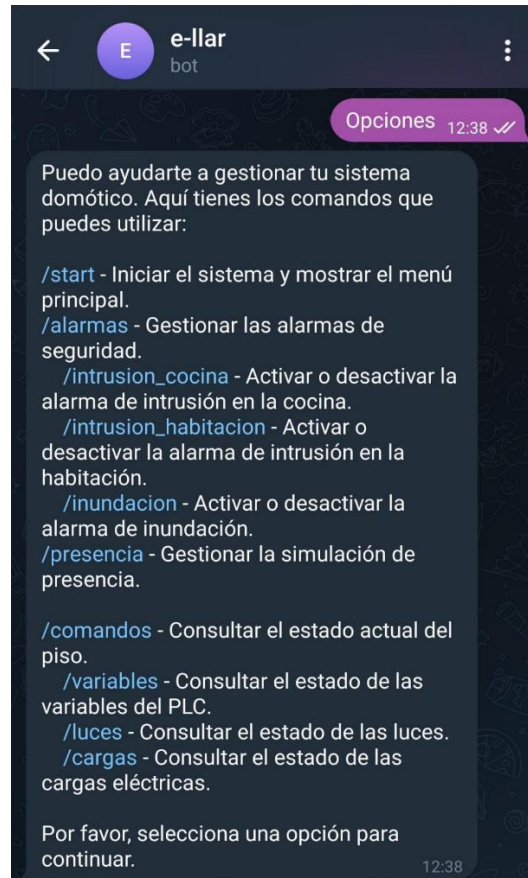


Ilustración 7.41: Flujo comandos Telegram

A continuación, se incluye un ejemplo de lo que ocurre al enviar la palabra “Opciones” al chat ya que en sí mismo ya explica todos los comandos. La ventaja de emplear comandos es que desde Telegram, al pulsarlo se envían automáticamente



*Ilustración 7.42: Opciones Telegram*

### 7.3.6. Base de datos

Para este proyecto, se ha implementado una base de datos para registrar las distintas alarmas disparadas en el piso piloto, facilitando su posterior análisis y visualización desde el dashboard, tanto en formato tabla como en un intuitivo gráfico de pastel.

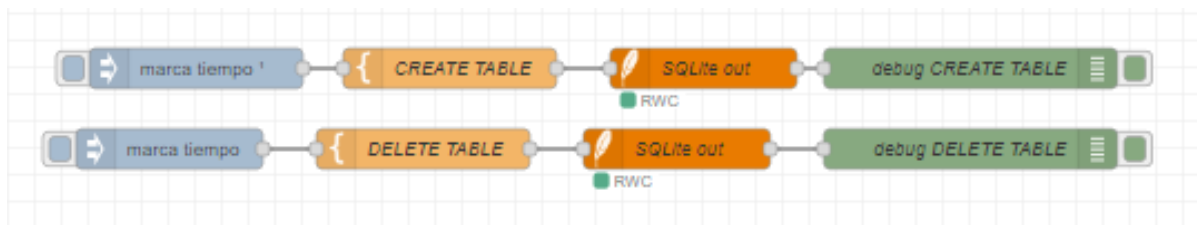
Las bases de datos se han implementado a través de nodos SQL en Node-RED pertenecientes al paquete "node-red-node-sqlite". Estos nodos permiten insertar, recuperar y procesar información relevante, garantizando una interacción eficiente con los datos almacenados.



*Ilustración 7.43: Paquete SQL Node-RED*

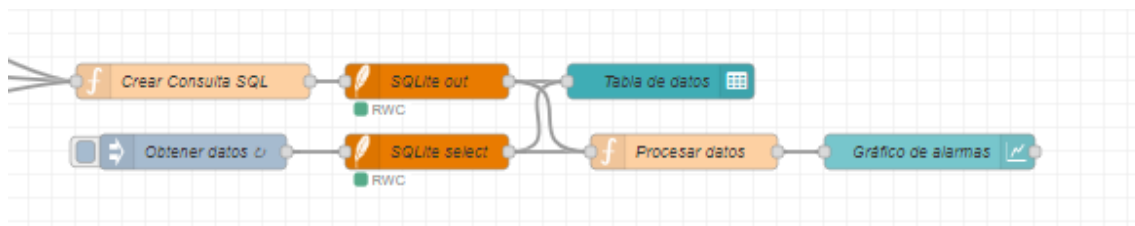
En primer lugar, para generar una tabla nueva en la base de datos, se emplea un nodo "function" encargado de crear la tabla exclusivamente en el caso de que no exista previamente (CREATE TABLE IF NOT EXISTS registro\_alarmas). Esta tabla cuenta con tres columnas: una columna "fila" que es la clave primaria y se incrementa automáticamente, una columna "timestamp" que almacena la fecha y hora con un valor predeterminado de la marca de tiempo actual, y una columna "tipo\_alarma" que solo acepta valores específicos como "Alarma intrusión cocina", "Alarma intrusión habitación" y "Alarma inundación".

Desde el nodo función, la instrucción pasa a un nodo propio de SQL configurado para recibir la consulta a través del mensaje (msg.topic) y ejecutar la instrucción SQL en la base de datos especificada, que en este caso es un archivo ubicado en la ruta "C:\Users\34619\Desktop\Laboratorio\Node-red\registro\_alarmas.db". De manera adicional se incluye un flujo designado para resetear la tabla en caso necesario sigue la misma estructura que el flujo de creación de la tabla, pero en lugar de crearla la destruye.



*Ilustración 7.44: Flujos creación y destrucción de tablas*

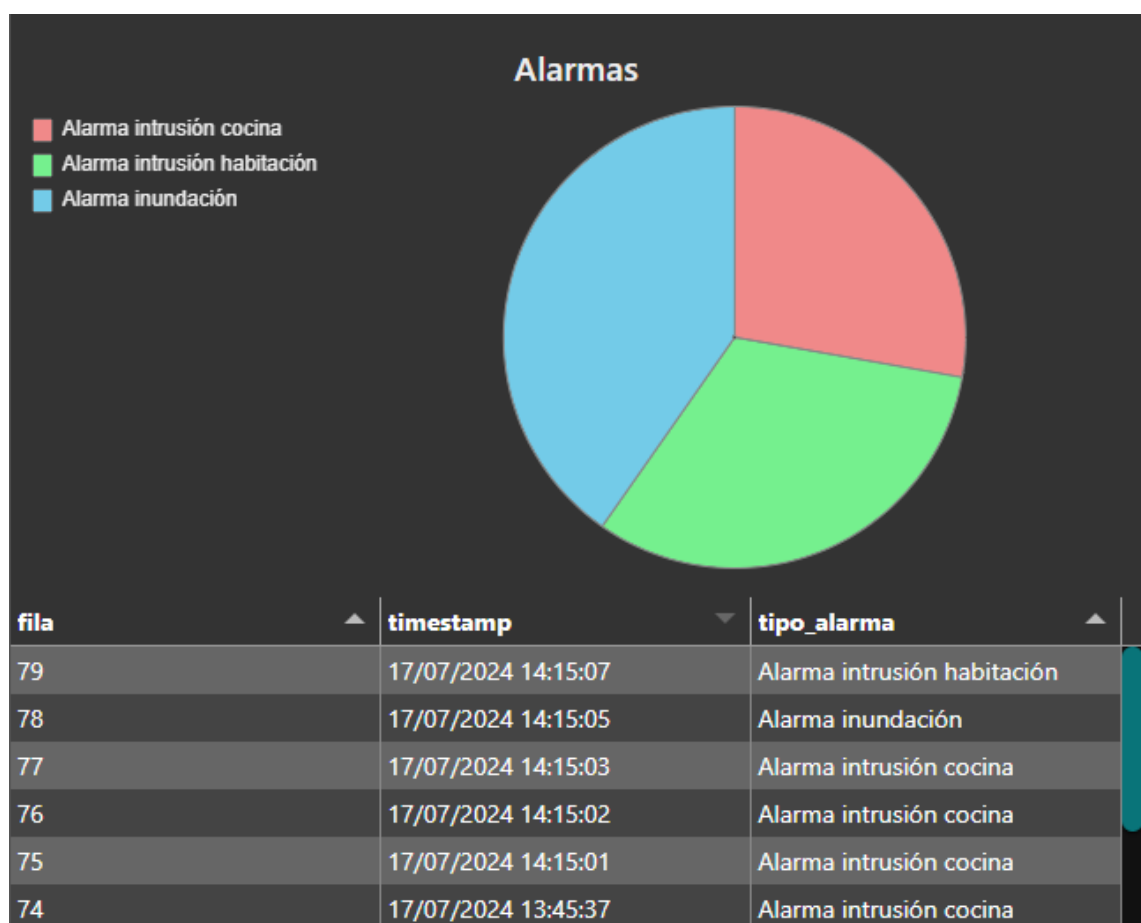
Una vez creada la base de datos, al generarse alguna de las alarmas a registrar se envía a un nodo función un string con el tipo de alarma que corresponda: "Alarma intrusión cocina", "Alarma intrusión habitación" o "Alarma inundación". El nodo función se encarga de introducir este dato en la tabla correspondiente junto con el momento en el que se produce la alarma en formato DD/MM/YYYY hh:mm:ss y una primary key -único para identificar el incidente. Para ello emplea el comando: `msg.topic = `INSERT INTO registro_alarmas (timestamp, tipo_alarma) VALUES ('${timestamp}', '${tablaAlarma}')``.



*Ilustración 7.45: Flujo SQL*

Para consultar los datos una vez registrados se emplea otro nodo “function” donde los datos procesan para contar las diferentes alarmas registradas. Para ello recorre cada fila de resultados y contabiliza las ocurrencias de cada tipo de alarma. Estos datos se formatean para ser visualizados en un gráfico de tarta en el dashboard de Node-RED. El nodo de procesamiento de datos genera una estructura adecuada para el gráfico. Por ejemplo: [{"series":["Alarmas"],"data":[[17,22,28]],"labels":["Alarma intrusión cocina","Alarma intrusión habitación","Alarma inundación"]}]

Finalmente, un nodo de gráfico de usuario presenta estos datos de manera visual, permitiendo un análisis rápido y efectivo del historial de alarmas registradas. Esta integración de SQL con Node-RED y SQLite asegura una gestión de datos robusta y accesible, facilitando la monitorización y control dentro del sistema domótico.



*Ilustración 7.46: Ventana registro alarmas*

---

## 7.4. MANUAL DE USUARIO DEL DASHBOARD

Tras haber desarrollado la explicación de la programación en Node-RED se presenta en este apartado un resumen de la manera manejar el sistema domótico desde el dashboard. El interfaz de usuario proporciona una serie de pantallas amigable y accesible para la monitorización y control de diversas variables y eventos del sistema. Este dashboard ha sido planteado para ser utilizado desde un dispositivo móvil, por lo tanto, todas las capturas incluidas en esta sección son capturas hechas de la pantalla de un teléfono.

En primer lugar, para permitir el acceso remoto se ha de emplear el link generado como de explicó en la sección [7.3.2: Control remoto](#) para acceder al dashboard de Node-RED aún sin estar conectado a la red wifi e-llar.

En caso de duda con alguno de los elementos del dashboard, al pulsar sobre el elemento en cuestión, aparece un tooltip indicando la función del elemento.

### 7.4.1. Home

Una vez en el dashboard se accede en primer lugar a la ventana Home. Desde esta ventana se permite al usuario navegar entre las diferentes ventanas. Es muy sencillo:

1. Hacer clic en el menú desplegable "Seleccionar"
2. Escoger una de las opciones disponibles en la lista.
3. La pantalla del dashboard se actualizará automáticamente para mostrar la ventana seleccionada.

Además, desde todas las pantallas puedes navegar de una a otra empleando los iconos del lateral de la pantalla.

Por otra parte, también se visualiza rápidamente las alarmas disparadas mediante tres indicadores visuales que pasaran a rojo en caso de estar disparadas. Por ejemplo, en la [Ilustración 7.47](#) estarían disparadas la alarma de intrusión de la habitación y la alarma de inundación.



*Ilustración 7.47: Home dashboard*

## 7.4.2. Calefacción

Respecto a la ventana calefacción en la parte superior de la ventana se encuentra un interruptor para activar o desactivar el "Modo automático". En caso de estar desactivado, solo se tendrá en cuenta la señal del termostato. El usuario debe introducir una temperatura objetivo en el termostato y, según si esta es mayor o menor que la detectada por su termómetro interno y se activará o no la calefacción

Como se explica en la sección [5.7.4](#), al activar el modo automático, el sistema de calefacción funcionará de acuerdo con el termostato, al horario y al sensor de ventana abierta ya que en caso de estar la ventana abierta o no estar en el horario correcto no se activa la calefacción.

Para modificar este horario se cuenta con la opción de introducir la hora en dos campos para ingresar la "Hora" y el "Minuto" en que se desea que el sistema se encienda y se apague. En caso de no introducirse cuenta con un horario por defecto.

Para la visualización del modo automático, el termostato y si la calefacción está o no activada se cuenta con tres ítems que cambian de gris a algún color representativo para indicar si está o no activado.



*Ilustración 7.48: Calefacción dashboard*

### 7.4.3. Intrusión

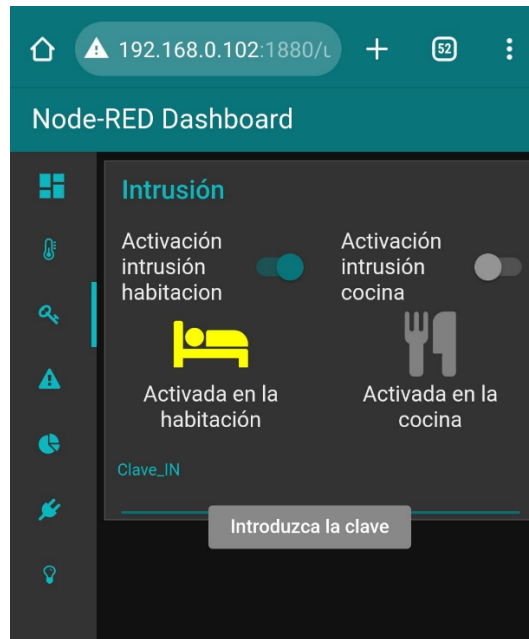
La ventana de alarmas de intrusión en el dashboard de Node-RED proporciona una interfaz para gestionar y monitorear las alarmas de seguridad en diferentes zonas del hogar.

Para activar las alarmas de intrusión, el usuario debe ingresar el código correcto en la interfaz de usuario. El código predeterminado para activar y desactivar las alarmas es "1234". Una vez ingresado el código, se puede seleccionar las zonas donde se desea activar la alarma. En el caso del piso piloto, las opciones disponibles son "Salón-Cocina" y "Habitación-Aseo". También se puede activar ambas zonas simultáneamente si se considera necesario.

Una vez activada la alarma, se inicia un tiempo de retardo para permitir al usuario salir de la zona sin que la alarma se dispare inmediatamente. Si se detecta movimiento en una zona protegida, se inicia otro tiempo de retardo para permitir al usuario desactivar la alarma ingresando el código nuevamente. Si el código no se ingresa a tiempo, la alarma se activará.

Los interruptores en la interfaz permiten activar o desactivar las alarmas en la "Intrusión en la habitación" y "Intrusión en la cocina". Los iconos visuales indican el estado de las alarmas: una cama amarilla para la habitación y utensilios grises para la cocina, mostrando claramente cuándo una zona está protegida.





*Ilustración 7.49: Intrusión dashboard*

#### 7.4.4. Inundación

Respecto a la ventana de inundación se puede observar una sección dedicada a la "Activación alarma inundación". Este control permite activar o desactivar la alarma de inundación. Al estar activada, cualquier detección de agua por los sensores resultará en una alerta inmediata, tanto en el dashboard como a través de Telegram. En la instalación del piso e-llar, se ha implementado un sistema de detección de inundación en el baño y la cocina.

En el caso de detectar una inundación, se activará un actuador que cerrará automáticamente la electroválvula. Esta acción detendrá el flujo de agua, minimizando así los daños hasta que se pueda solucionar la avería. La interfaz del dashboard muestra claramente el estado de este actuador y la activación mediante íconos. El ícono de "Activada" se ilumina en verde para indicar que se ha activado la alarma, el ícono de "Inundación" se ilumina en rojo para indicar que se ha disparado la alarma y el ícono "Actuador" pasa a amarillo de estar actuando el actuador.



*Ilustración 7.50: Inundación dashboard*

### 7.4.5. Cargas

La ventana de cargas está enfocada a cargas eléctricas del piso piloto, ofreciendo dos modos de operación que pueden seleccionarse desde la interfaz del usuario. En el modo manual, es posible activar y desactivar las cargas directamente desde la pantalla, lo que proporciona un control preciso e inmediato sobre cada dispositivo conectado.

En el modo automático, el sistema se apoya en detectores que gestionan las cargas según su estado. Por ejemplo, un detector de movimiento puede desactivar una carga si no detecta presencia durante un periodo de tiempo específico, contribuyendo al ahorro de energía. De manera similar, un detector de inundación puede desactivar automáticamente las cargas por razones de seguridad, evitando potenciales daños eléctricos.

El control de cargas en el piso piloto incluye tres dispositivos principales: la carga de la cocina, la carga del calefactor y la carga del ventilador de la habitación. Cada una de estas cargas tiene funciones específicas y se puede gestionar de manera individual desde la interfaz del dashboard. Los botones visuales y los indicadores de estado facilitan la supervisión y el control de cada carga, mostrando claramente su estado actual.

En la sección de "Cargas" del dashboard, los usuarios pueden ver y modificar el estado de cada carga con solo un toque. Los íconos y etiquetas ayudan a identificar rápidamente cada dispositivo, y el interruptor de modo automático/manual permite cambiar entre los modos de operación según las necesidades. Este diseño intuitivo asegura que la gestión de las cargas eléctricas sea sencilla y eficiente, optimizando el uso de energía y garantizando la seguridad del hogar.



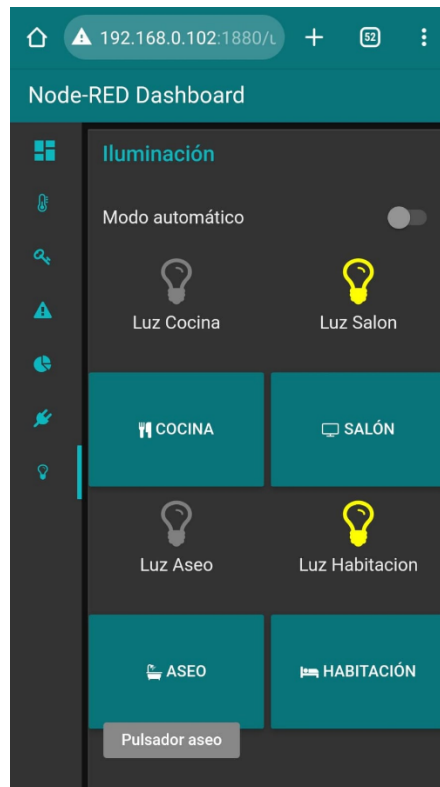
*Ilustración 7.51: Cargas dashboard*

#### 7.4.6. Iluminación

La ventana "Control\_Luces" del dashboard de Node-RED permite gestionar la iluminación del piso piloto, ofreciendo dos modos de operación: automático y manual. En el modo manual, el usuario puede encender y apagar las luces utilizando los pulsadores físicos instalados en el piso o los pulsadores virtuales disponibles en la interfaz de usuario.

En el modo automático, el sistema controla la iluminación basándose en la detección de luminosidad, la presencia de movimiento y el horario preestablecido. Si no se detecta suficiente luz, se encuentra dentro del horario establecido para cada zona y se detecta presencia en el salón o el aseo, las luces se activarán automáticamente.

La interfaz del dashboard muestra claramente el estado de cada luz, permitiendo a los usuarios identificar rápidamente qué luces están encendidas (en amarillo) o apagadas (en gris).



*Ilustración 7.52: Iluminación dashboard*

## 8. Conclusiones y futuras ampliaciones

Respecto al trabajo realizado, se ha llevado a cabo una muy necesaria revisión de las instalaciones en el piso piloto *E-Llar*, lo cual ha permitido sobremano adaptar el cableado para poder conectar un autómata programable Simatic S7-1200, así como determinar algunos errores en elementos sensores y actuadores. Esto hace posible el futuro desarrollo de proyectos en el piso piloto de forma más rápida y efectiva.

En lo que concierne al propio control y automatización, se ha implementado un control esencial de las instalaciones principales de una vivienda. Se ha utilizado una metodología de programación que permite una mejor organización de datos y variables asociados a las diferentes funcionalidades mediante el uso de tipos de datos de usuario y las variables correspondientes, así como bloques funcionales que encapsulan su comportamiento. Esto facilita también una mayor flexibilidad y favorece las ampliaciones y mejora de la aplicación.

El conocimiento adquirido respecto a la forma de abordar, desarrollar y depurar proyectos con PLC y HMI ha resultado muy positivo, máxime cuando se ha contado con una instalación real que si bien es mejorable responde a la realidad de su funcionamiento. Por otra parte, el contar con equipos de Siemens y la herramienta de programación TIA Portal es un valor añadido pues la cuota de mercado de este fabricante en implantación de soluciones a nivel industrial es muy importante siendo uno de los líderes en este mercado.

Quedan por resolver algunos problemas con elementos de la instalación que se han averiado, como el motor de persiana y el detector de presencia en el baño, así como proceder probablemente a un recableado general que por las diferentes tecnologías implantadas resulta difícil de mantener, aunque no deja de ser un laboratorio de prueba y experimentación. Gran parte del trabajo, por iniciativa personal, se ha desarrollado en la propia instalación garantizando un acercamiento más directo a la realidad de las instalaciones.

Las aplicaciones que se han llevado a cabo en el módulo IoT2040 de Siemens han resultado especialmente complejas dado el escaso conocimiento inicial con en el tema. Algunas de las pruebas se han implantado en un ordenador personal con Node-RED local cuya compatibilidad con el módulo IoT en cuanto a configuración y funcionamiento es idéntico pudiendo depurar y verificar dichas funcionalidades.

En este enfoque se han llevado a cabo el desarrollo de un dashboard para acceso a nivel local y remoto con fácil manejo y adaptación del interfaz de usuario a terminales móviles. También se ha establecido un sistema de mensajería vía Telegram para aviso y explotación de algunos elementos de la instalación. Y finalmente se ha optado por implantar un ejemplo de uso de base de datos para almacenamiento y acceso de información relevante para un posterior análisis de datos.

Y en lo que respecta a posibles mejoras y ampliaciones de este trabajo se pueden considerar las siguientes:

- Sustitución de algunos elementos del piso piloto que por falta de recursos y tiempo no se han solucionado: motor de persiana, detector de presencia del aseo, detectores de fuego/humo y detector de luminosidad.
- Recableado y documentación detallada de toda la vivienda haciendo mucho más clara la información para futuros trabajos, permitiendo la coexistencia con otras tecnologías domóticas, no solo basadas en PLCs sino en el sistema Busing, KNX o Legrand.
- Integración de interfaces de usuario multimodales como por ejemplo Alexa o Google Assistance, así como incorporación de cámaras de video.
- Aumentar las prestaciones del sistema, adquiriendo mayor variedad y volumen de información del funcionamiento de las instalaciones permitiendo realizar una completa analítica de datos y optimizando modos de uso, gestión energética, seguridad, etc.

## 9. Bibliografía

- [1] «INFORME\_GENIA\_e-llar.pdf».
- [2] L. Molina, *Instalaciones domóticas*. Madrid: McGraw Hill, 2010.
- [3] J. C. Martín Castillo, *Instalaciones domóticas*. EDITEX.
- [4] «IEC 61131-3 (Lenguajes).pdf». Accedido: 7 de junio de 2024. [En línea]. Disponible en: [http://isa.uniovi.es/~vsuarez/Download/IEC%2061131-3%20\(Lenguajes\).pdf](http://isa.uniovi.es/~vsuarez/Download/IEC%2061131-3%20(Lenguajes).pdf)
- [5] «Texto\_Desarrollo proy\_Domoticos con PLCs.pdf».
- [6] «CPU 1214C». Accedido: 29 de mayo de 2024. [En línea]. Disponible en: [https://mall.industry.siemens.com/mall/es/ww/catalog/products/10045652?activeTab=order&regionUrl=](https://mall.industry.siemens.com/mall/es/ww/catalog/products/10045652?activeTab=order&regionUrl=/)
- [7] «SIMATIC S7-1200», Siemens México. Accedido: 30 de junio de 2024. [En línea]. Disponible en: <https://www.siemens.com/mx/es/productos/automatizacion/systems/industrial/plc/s7-1200.html>
- [8] «SIMATIC S7-1200\_1214datasheet.pdf».
- [9] «KTP70\_datasheet\_es.pdf».
- [10] «Equipos estándar 2nd Generation». Accedido: 30 de junio de 2024. [En línea]. Disponible en: <https://mall.industry.siemens.com/mall/es/WW/Catalog/Product/6AV2123-2GB03-0AX0>
- [11] «SIMATIC IoT2040 datasheet.pdf».
- [12] S. Sa, «SIMATIC IOT2040: Primera instalación y primer programa en node-red», 2018.
- [13] «Communication with SIMATIC».
- [14] «IoT20\_guideline.pdf».
- [15] «node-red-contrib-s7». Accedido: 8 de julio de 2024. [En línea]. Disponible en: <http://flows.nodered.org/node/node-red-contrib-s7>
- [16] «Running Node-RED locally : Node-RED». Accedido: 7 de junio de 2024. [En línea]. Disponible en: <https://nodered.org/docs/getting-started/local>
- [17] «iot2000\_operating\_instructions\_enUS\_en-US.pdf».
- [18] «Communication with SIMATIC.pdf».