Universidad de Oviedo

Manual del programador PLC del Trabajo Fin de Máster realizado por

DAVID MANUEL MANTILLA LÓPEZ

para la obtención del título de

Máster en Ingeniería de Automatización e Informática Industrial

AUTOMATIZACIÓN DE LA PLANTA PILOTO DE UHT Y PASTEURIZACIÓN PARA CAPSA FOOD.

MAYO 2017

Contenido

Índice de ilustraciones

Índice de tablas

1. Introducción

1.1. Identificación del proyecto

Título: Automatización de la planta piloto de UHT y pasteurización para CAPSA FOOD.
Tutor Académico: Ricardo Mayo Bayón.
Autor: David Manuel Mantilla López.
Fecha: mayo 2017.

1.2. Visión general del proyecto

Este proyecto nace de la necesidad de la empresa CAPSA FOOD en renovar y mejorar el control y supervisión de la planta piloto UHT y pasteurización, la cual, es utilizada para realizar pruebas y tratamientos a diferentes productos lácteos y en función de los resultados tomar unas medidas adecuadas para modificarlos, mejorarlos e introducirlos en producción con el fin de situarlos en el mercado.

Con este proyecto, la empresa pretende obtener un desarrollo que permita la integración tecnológica en los tres primeros niveles de la pirámide de automatización. Esta tarea supone inicialmente una evaluación y análisis del funcionamiento, equipamiento y tecnología presentes en la planta piloto, para así definir y realizar una serie de acciones que permitan alcanzar los objetivos propuestos.

1.3. Visión general del documento

La programación del PLC se basa en los algoritmos de control diseñados y en la observación del programa que ejecuta actualmente la máquina en donde se han desarrollado diversas mejoras que permitan un control más flexible e integrado de la planta piloto.

1.4. Ámbito del documento

Este documento "Pliego de condiciones", pertenece al trabajo fin de master "Automatización de la planta piloto de UHT y pasteurización para CAPSA FOOD" cuyo autor es David Manuel Mantilla López, alumno del Master en Automatización e informática industrial de la Universidad de Oviedo – Campus Gijón.

1.5. Documentos referenciados

A continuación, se detallan los documentos relacionados con la planificación y el presupuesto de este proyecto.

1.5.1. Documentos del proyecto

No se hace referencia a ningún documento del proyecto.

1.5.2. Documentos externos

No se hace referencia a otro documento externo del proyecto

1. Recursos utilizados

1.1. Recursos hardware

- PC Lenovo ThinkPad E560

1.2. Recursos software

- TIA PORTAL V13.
- PLCSIM para simulación hardware PLC y comunicaciones.

1.3. Recursos humanos

- Manejo del software TIA PORTAL V13.
- Conocimiento de los diferentes lenguajes de programación de PLCs.

2. TIA PORTAL V13

2.1. Versión del software

Para automatizar la planta piloto, se ha utilizado la herramienta de software TIA PORTAL V13 ya que es compatible con la CPU seleccionada. Este software ha sido desarrollado por la compañía siemens y se ciñe de forma satisfactoria a la norma IEC 61131-3 sobre programación para sistemas de automatización.

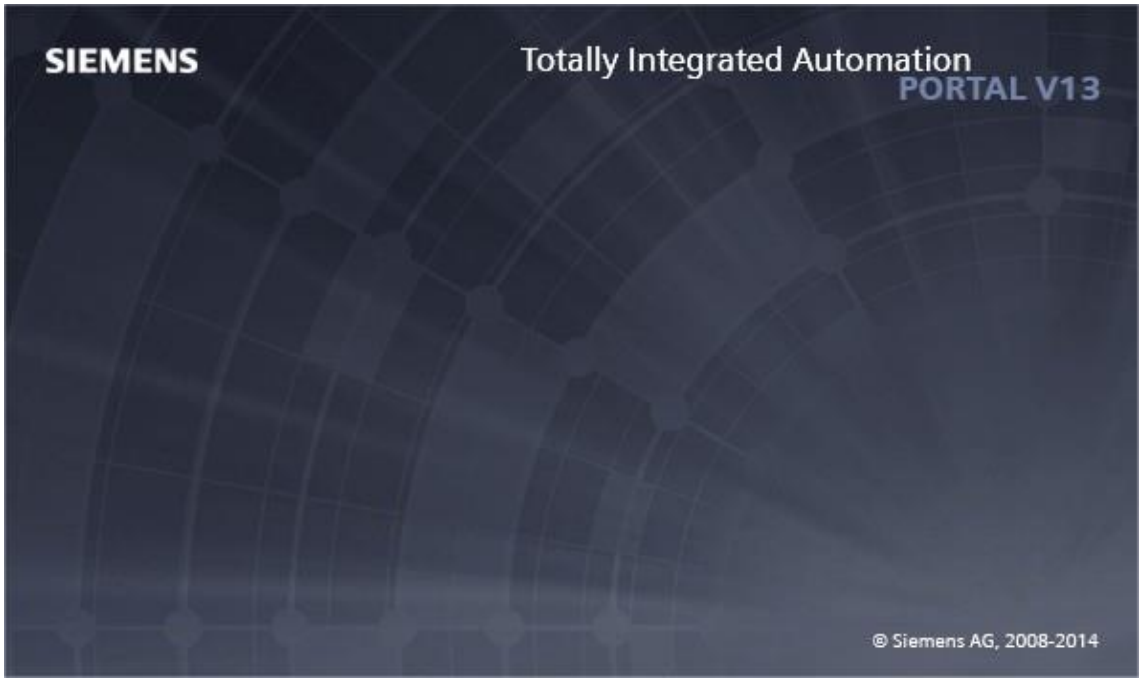

Ilustración 2.1-1 Versión de software

2.2. Requerimientos hardware

El PC utilizado para programar con el TIA portal V13 debe de satisfacer los siguientes requisitos mínimos.

- Procesador: CoreTM i5-3320M 3.3 GHz o similar.
- Memoria principal: 8 GB de memoria (recomendado) o más.
- Disco duro: 300 GB SSD.
- Gráficos: Mín. 1920 x 1080.
- Pantalla: 15,6" display de pantalla ancha (1920 x 1080).

3. Configuración del proyecto

3.1. Creación del proyecto

Para crear un proyecto nuevo en TIA PORTAL V13 primeramente hay que abrir el TIA PORTAL. Simplemente, se crea un proyecto nuevo, desde la barra de herramientas en la opción "crear proyecto". Hay que asignarle un nombre al proyecto y se guardará en la ruta especificada.

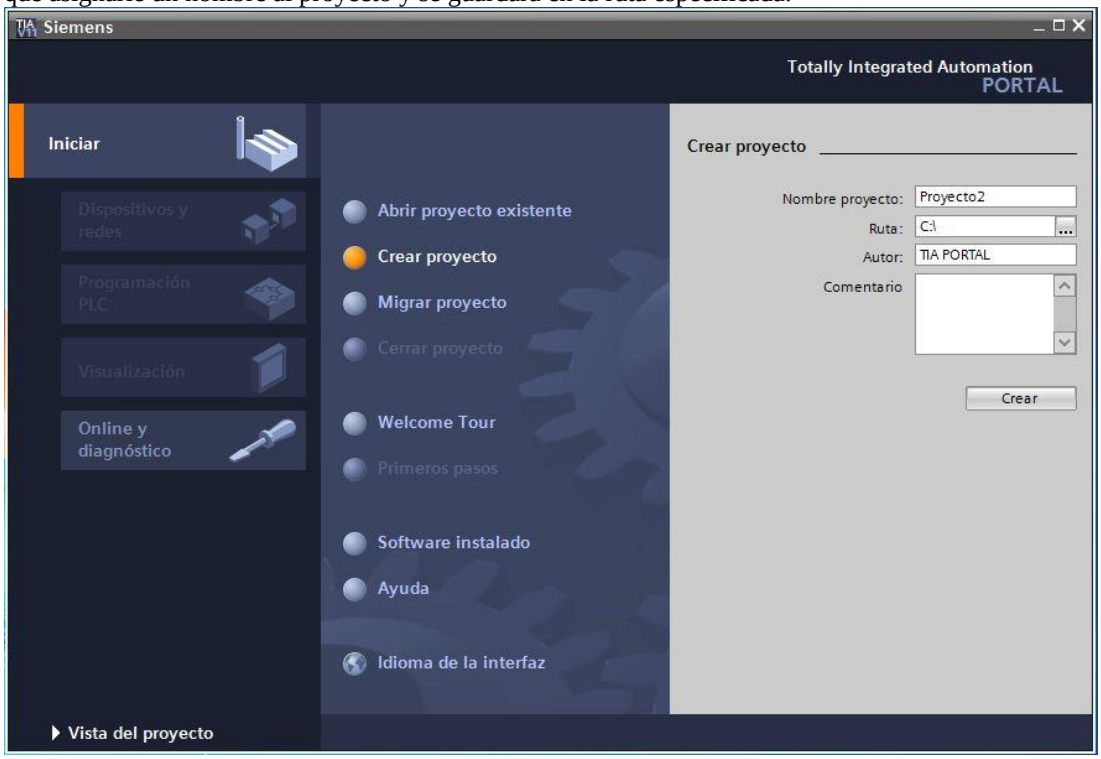

Ilustración 3.1-1 Creación de un proyecto

3.2. Configuración hardware

Para poder trabajar con el TIA PORTAL y poder establecer conexión con nuestro PLC y HMI tendremos que configurar el hardware dentro de nuestro proyecto. Para ello, seleccionaremos el dispositivo, con la opción indicada para tal.

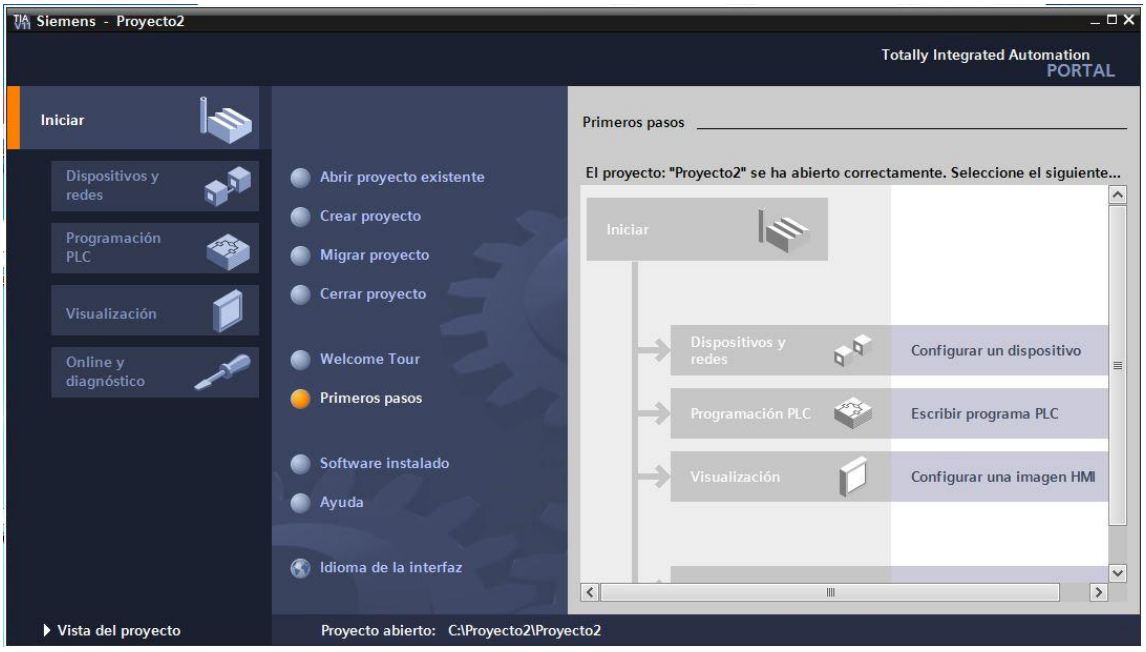

Ilustración 3.2-1 Iniciar un proyecto

De este modo, se abrirá una ventana donde se puede asignar el PLC correspondiente de nuestro proyecto. Con la opción de "Agregar dispositivo" se abrirá un árbol donde poder seleccionar la CPU. En este caso, se ha seleccionado una CPU S7-1200 tipo 1214C DC/DC/RLY con la referencia &ES7214- 1HG40-0XB0. Pulsamos el botón "Agregar" y automáticamente se abrir la vista del proyecto.

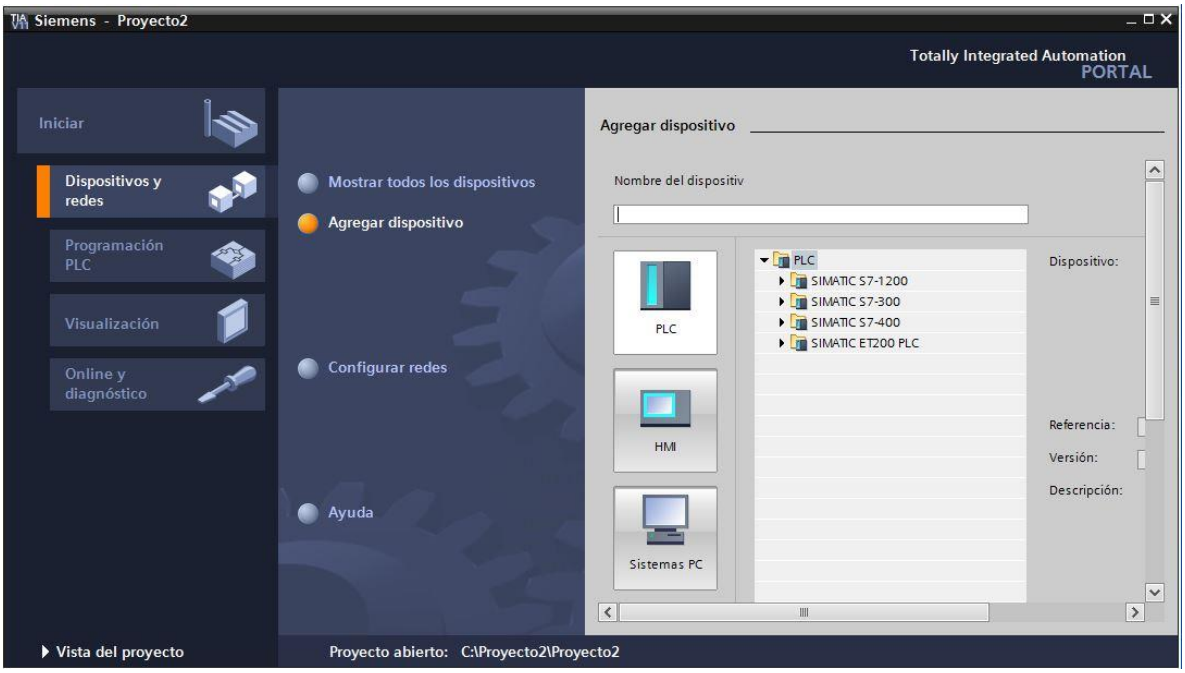

Ilustración 3.2-2 Seleccionar PLC

El siguiente paso es seleccionar con doble clic la opción "Configuración de dispositivos" y nos mostrará el hardware que existe en el proyecto. Solamente se verá la CPU que hemos seleccionado anteriormente. Ahora se pueden insertar los módulos restantes. Para ello, hay que situarse en la columna derecha del TIA PORTAL donde se encuentra otro árbol de carpetas, en el cual, podremos seleccionar los módulos que queremos añadir a la

CPU. Simplemente hay que seleccionar la referencia requerida y arrastrar a la posición que queramos insertarla. En este caso se han seleccionado los siguientes módulos:

- SM1221 DI 16x24 VDC con referencia 6ES7221-1BH32-0XB0.
- SM1222 DO 16xRELAY con referencia 6ES7222-1HH32-0XB0.
- SMI131 AI 8 x RTD con referencia 6ES7231-5PF32-0XB0.
- SMI1231 AI 4 x RTD x 16 Bit con referencia 6ES7231-5PD32-0XB0
- SMI1231 AI 4 x 13 Bit con referencia 6ES7231-4HD32-0XB0
- SMI1232 AO 4 x 14 Bit con referencia 6ES7232-4HD32-0XB0

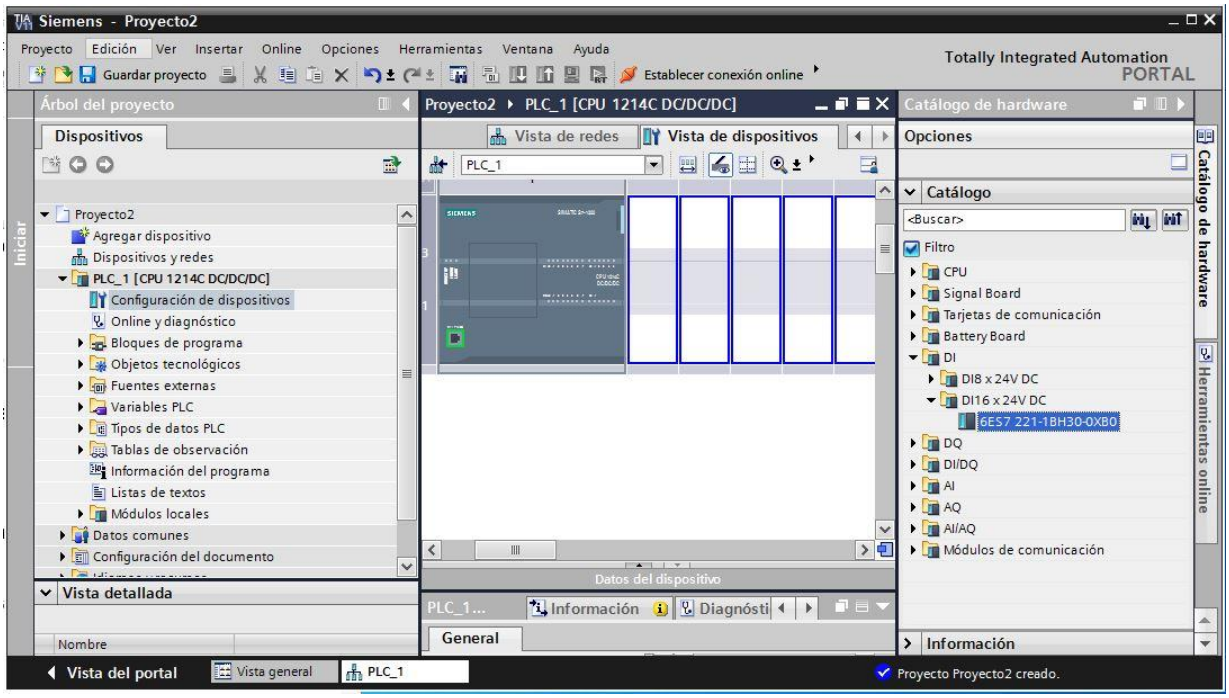

Ilustración 3.2-3 Añadir módulos

Una vez insertados todos los módulos hay que configurarlos independientemente. Haciendo doble clic sobre la CPU se verán todas las características que van a depender del tipo de proyecto a utilizar, el tipo de comunicación de la rede de autómatas, si queremos más prioridad en la comunicación, el número de estaciones, el número de dirección de comunicación PROFINET, etc.

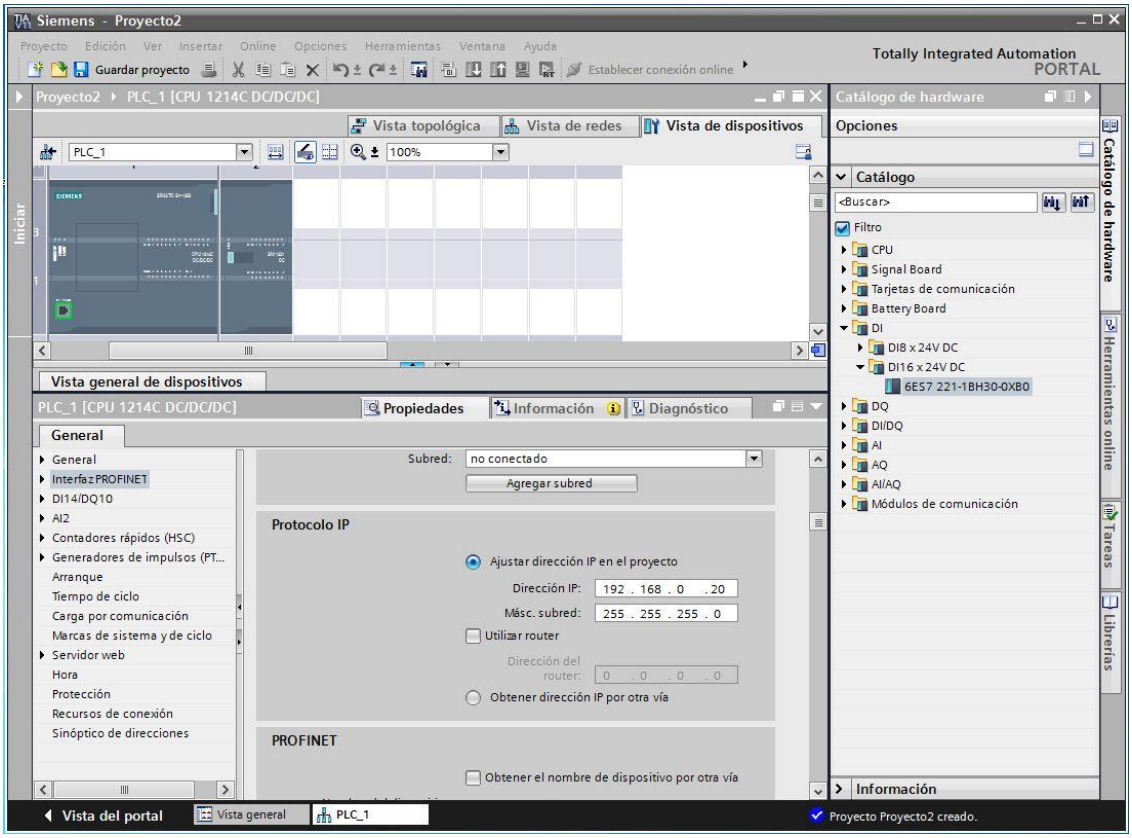

Ilustración 3.2-4 Propiedades CPU

Se puede observar que, seleccionando cada módulo, muestra información de cada uno de ellos y de las entradas o salidas que tienen asignadas y su número de comienzo de los bits a utilizar para esos módulos. TIA PORTAL establece un rango en el cual debe estar el valor de la entrada y de salida. Estos se pueden modificar según se quiera. En este caso, se ha utilizado la numeración expuesta en el documento “Planos Eléctricos”.

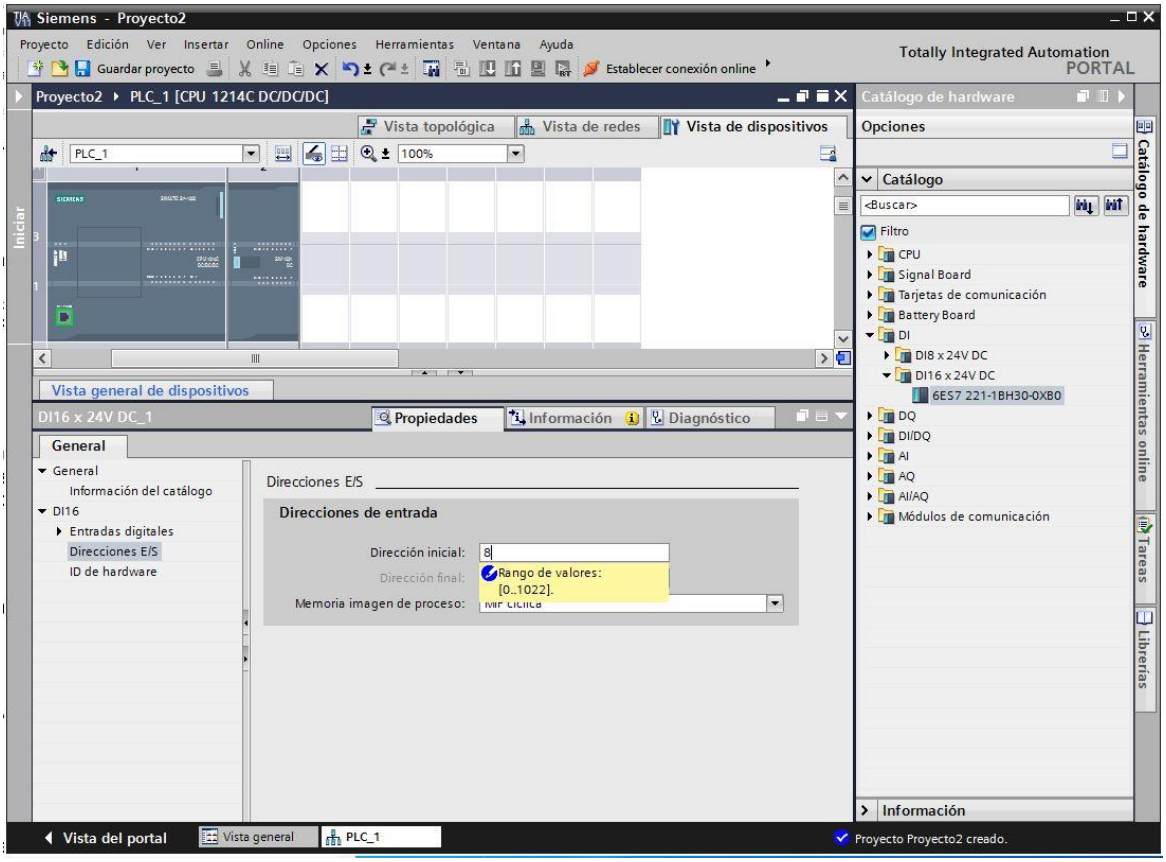

Ilustración 3.2-5 Configuración E/S

Para finalizar con el proceso de estructura hardware, hay que guardar el proyecto y compilar todo, tanto software como hardware. De este modo, ya se puede trabajar en el programa generando bloques nuevos en la carpeta de bloques de programa en el árbol del proyecto.

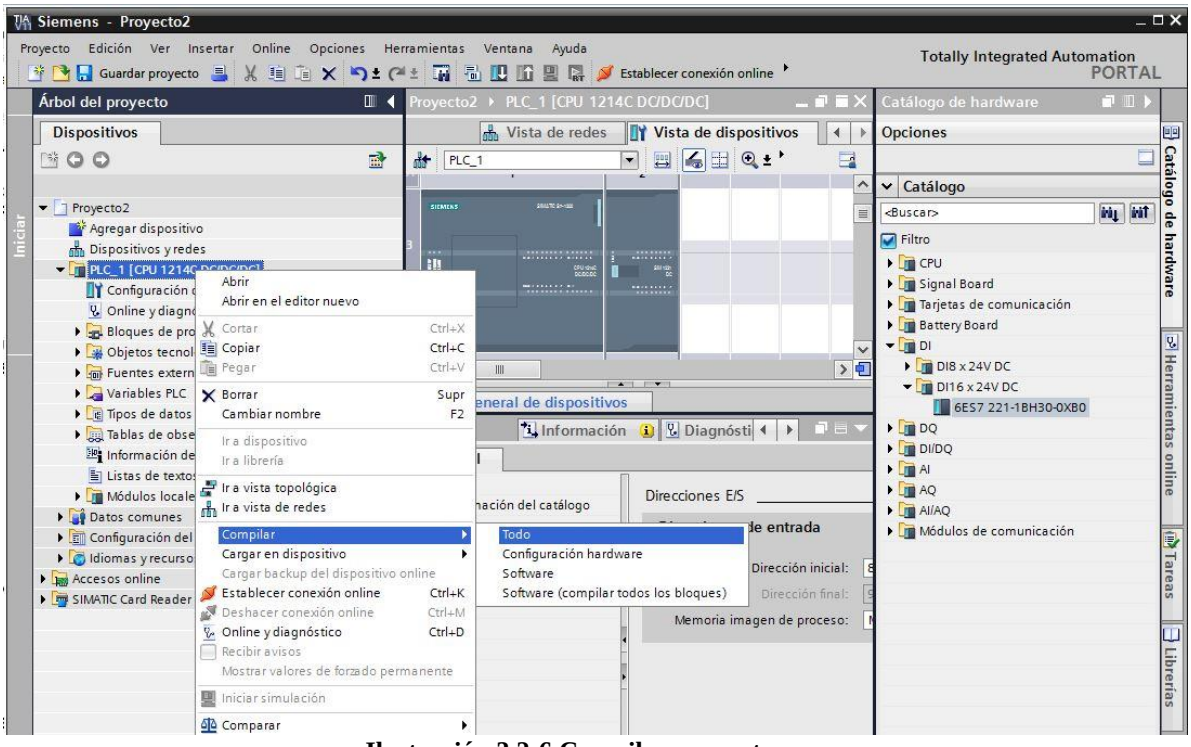

Ilustración 3.2-6 Compilar proyecto

4. Programación de la planta piloto.

4.1. Versión del SOFTWARE.

Para automatizar la planta piloto, se ha utilizado la herramienta de software TIA PORTAL V13 ya que es compatible con la CPU seleccionada. Este software ha sido desarrollado por la compañía siemens y se ciñe de forma satisfactoria a la norma IEC 61131-3 sobre programación para sistemas de automatización.

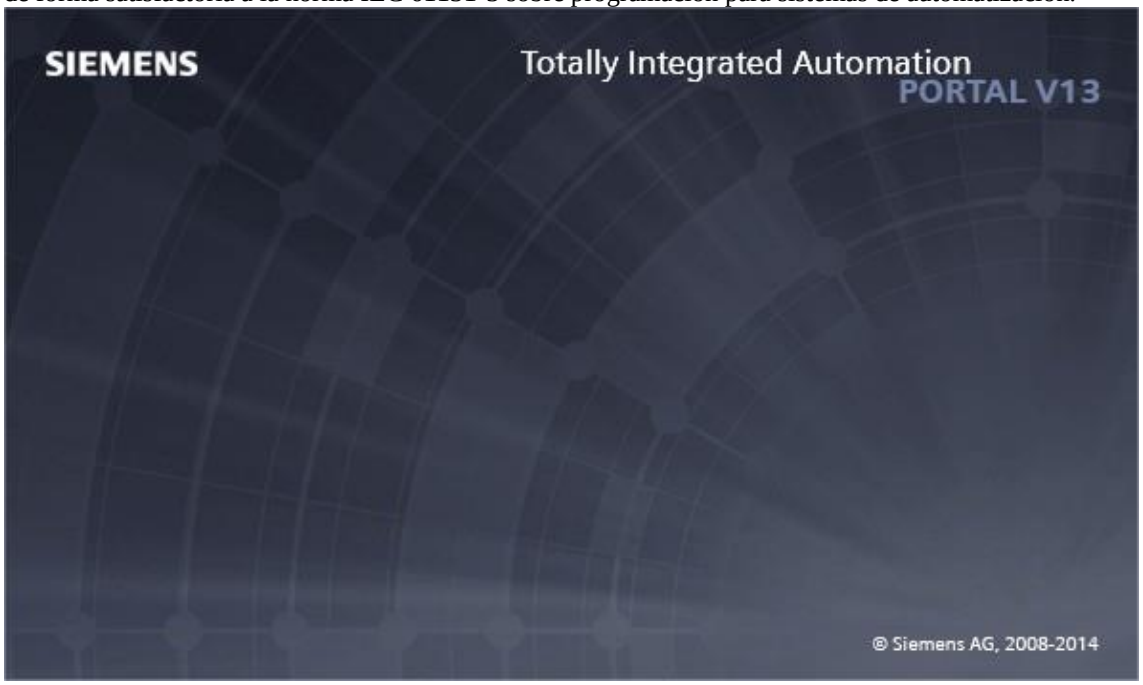

Ilustración 4.1-1 Versión de software

4.2. Requerimientos HARDWARE.

El PC utilizado para programar con el TIA portal V13 debe de satisfacer los siguientes requisitos mínimos.

- Procesador: CoreTM i5-3320M 3.3 GHz o similar.
- Memoria principal: 8 GB de memoria (recomendado) o más.
- Disco duro: 300 GB SSD.
- Gráficos: Mín. 1920 x 1080.
- Pantalla: 15,6" display de pantalla ancha (1920 x 1080).

4.3. Bloques para la programación del S7-1200

Para la programación estructurada en el S7-1200 existen los siguientes bloques:

- OB (bloque de organización): Un OB es llamado por el sistema operativo de forma cíclica y constituye la interfaz entre el programa de usuario y el sistema operativo. En este OB, se comunica a la unidad de control del sistema de automatización qué bloques de programa debe ejecutar a través de comandos de llamada de bloque.

- FB (bloque de función): Necesita un área de memoria asignada para cada llamada (instancia). Al llamar a un FB se le puede asignar un bloque de datos (DB) como bloque de datos instancia.

A los datos de este DB de instancia se accede a través de las variables del FB. Si se llama varias veces a un FB, se le deben asignar distintas áreas de memoria. En un bloque de función también pueden ser llamados otros FB y FC.

- FC (función): Un FC no tiene ningún área de memoria asignada. Los datos locales de una función se pierden tras ejecutar la función. En una función también pueden ser llamados otros FB y FC.

- DB (bloque de datos): Los DB se utilizan para proporcionar espacio de memoria para las variables de datos. Existen dos tipos de bloques de datos. DB globales, en los que todos los OB, FB y FC pueden leer los datos almacenados o incluso escribir datos en los DB; y DB de instancia, que están asignados a un FB determinado.

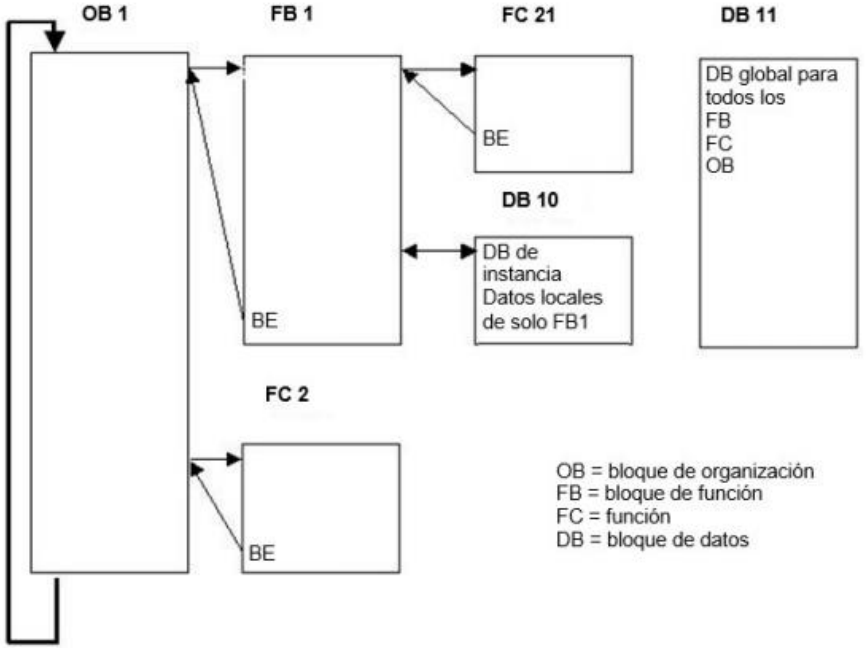

Ilustración 4.3-1 Bloques de programación

Como se observa en la siguiente imagen, existe un OB1 que realiza la llamada a diferentes funciones para llevar a cabo una labor determinada (p.e limpieza o producción). Estas funciones, que llevan a cabo la lógica de funcionamiento, acceden a los datos almacenados en diferentes DBs desarrollados.

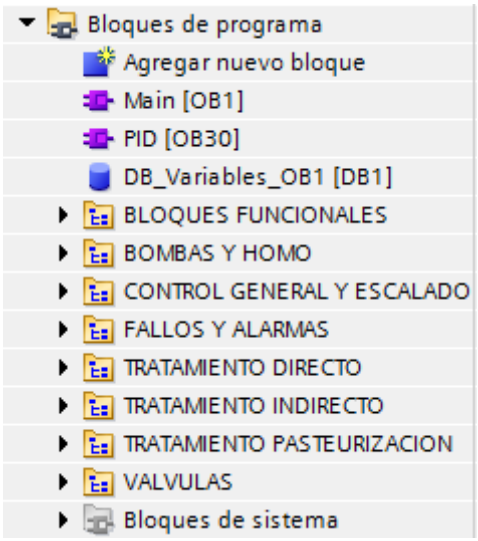

Ilustración 4.3-2 Bloques de programa desarrollados

4.4. Lenguajes de programación en del S7 – 1200

Step 7 ofrece los lenguajes de programación estándar siguientes para S7-1200:

- KOP (esquema de contactos) es un lenguaje de programación gráfico. Su representación se basa en esquemas de circuitos.
- FUP (diagrama de funciones) es un lenguaje de programación que se basa en los símbolos lógicos gráficos empleados en el álgebra booleana.
- SCL (structured control language) es un lenguaje de programación de alto nivel basado en texto.

Al crear un bloque lógico, se debe seleccionar el lenguaje de programación que emplea dicho bloque. El programa de usuario puede emplear bloques lógicos creados con cualquiera de los lenguajes de programación. A continuación, se muestra un ejemplo de cada lenguaje de programación utilizado.

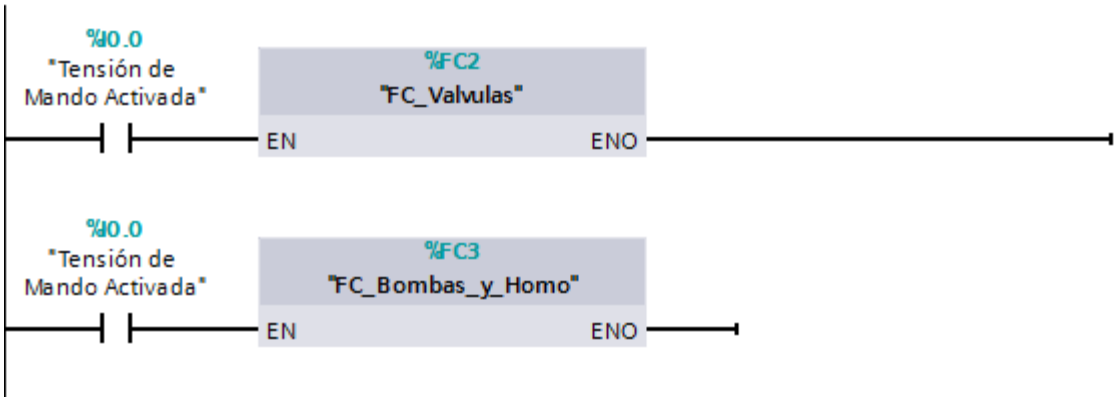

Ilustración 4.4-1 Lenguaje KOP

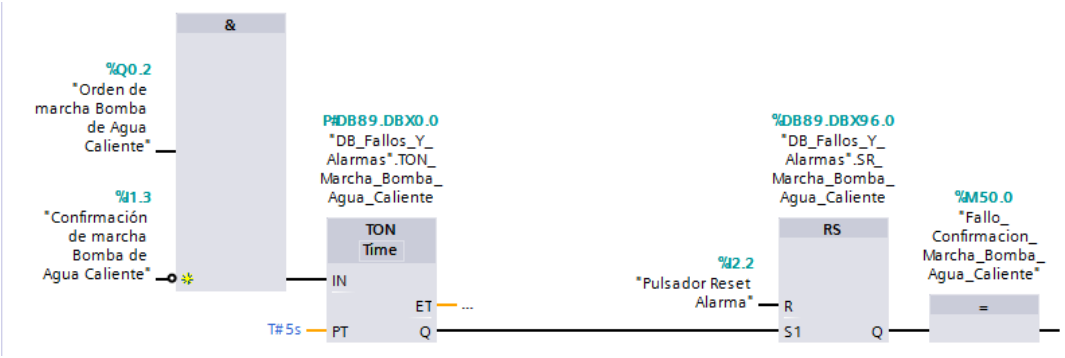

Ilustración 4.4-2 Lenguaje FUP

```
IF "DB_Variables_OB1".Auto_Man THEN
    "DB_Reguladores_PID".SetPoint_AguaCaliente := "DB_Reguladores_PID".SetPoint_AguaCaliente_Automatico;
    "DB_Reguladores_PID".SetPoint_Homogenizador := "DB_Reguladores_PID".SetPoint_Homogenizador_Automatico;
    "DB_Reguladores_PID".SetPoint_InyeccionVapor := "DB_Reguladores_PID".SetPoint_InyeccionVapor_Automatico;


    "DB_Reguladores_PID".ManualEneable_AguaCaliente := "DB_Reguladores_PID".ManualEneable_AguaCaliente_Automatico;
    "DB_Reguladores_PID".ManualEneable_Homogenizador := "DB_Reguladores_PID".ManualEneable_Homogenizador_Automatico;
    "DB_Reguladores_PID".ManualEneable_InyeccionVapor := "DB_Reguladores_PID".ManualEneable_InyeccionVapor_Automatico;

    "DB_Reguladores_PID".ManualValue_AguaCaliente := "DB_Reguladores_PID".ManualValue_AguaCaliente_Automatico;
    "DB_Reguladores_PID".ManualValue_Homogenizador := "DB_Reguladores_PID".ManualValue_Homogenizador_Automatico;
    "DB_Reguladores_PID".ManualValue_InyeccionVapor :=  "DB_Reguladores_PID".ManualValue_InyeccionVapor_Automatico;

END_IF;
```

Ilustración 4.4-3 Lenguaje SCL

4.5. Bloques de organización

Los bloques de organización permiten estructurar el programa del autómata sirviendo de interfaz entre el sistema operativo y el programa de usuario. Los OBs son controlados por eventos. Un evento, p. ej. una alarma de diagnóstico o un intervalo, hace que la CPU ejecute un OB. Algunos OBs tienen eventos de arranque y comportamiento en arranque predefinidos.

El OB de ciclo contiene el programa principal pudiendo llegar a tener más de un OB de ciclo en el programa de usuario. Cuando el autómata esta en RUN, los OBs de ciclo se ejecutan en el nivel de prioridad más bajo y pueden ser interrumpidos por todos los demás tipos de procesamiento del programa.

Nombre	OB	Lenguaje de programación	Tipo	Descripción
Main	OB1	KOP/FUP	Program cycle	Realiza la llamada a los diferentes FCs para que se ejecuten.
PID	OB30	KOP/FUP	Cyclic interrupt	Realiza la regulación PID del homogenizador, de la válvula de agua caliente y de la válvula de inyección de vapor.

Ilustración 4.5-1 Bloques de organización

4.6. Bloques de datos

Los bloques de datos (DBs) pueden ser utilizados en el programa para salvar información en la CPU. Tienen una capacidad de memoria de hasta 8 KBytes (8192 Bytes).

Existen dos tipos de bloques de datos. DBs Globales, en los cuales todos los OBs, FBs y FCs pueden guardar o leer datos y DBs de instancia, los cuales se encuentran asignados a un FB en particular.
En estos Dbs, se pueden almacenar diferentes tipos de datos (p.e. BOOL, WORD, o TIME).
Para llevar a cabo una estructura organizada de los DBs se han creado diferentes tipos que dependen del tipo de dato y del proceso en el que se utiliza. Esta estructuración se ha realizado con el objetivo de saber dónde se encuentra cada dato instanciado. En la siguiente tabla se muestran todos los DBs, el tipo de datos que contiene y una breve descripción de cada uno.

Nombre	DB	Tipos de datos
GENERAL		
DB_Control_De_Tiempos	DB82	Time/Bool/String
DB_Entradas_Analogicas	DB2	Real
DB_Intermitencia_Piloto	DB87	Time/Bool/TON_TIME
DB_Intermitencia_Sirena	DB88	Time/Bool/TON_TIME
DB_Manual_Bombas_y_Homo	DB68	Bool
DB_Manual_Valvulas	DB60	Bool
DB_Reguladores_PID	DB87	Bool/Real
DB_Salidas_Analogicas	DB3	Real
DB_SR_Bombas_Y_Homo	DB14	Bool
DB_SR_Válvulas	DB15	Bool
DB_Variables_OB1	DB1	Bool/String
DIRECTO		
DB_Bombas_Esterilizacion_Directo	DB20	Bool
DB_Mensajes_Esterilizacion_Directo	DB21	Bool
DB_Pulsadores_Esterilizacion_Directo	DB23	Bool
DB_SR_Esterilizacion_Directo	DB22	Bool
DB_Temporizadores_Esterilizacion_Directo	DB24	TP_TIME/TON_TIME
DB_Valvulas_Esterilizacion_Directo	DB19	Bool
DB_Bombas_Limpieza_Final_Directo	DB39	Bool
DB_Mensajes_Limpieza_Final_Directo	DB35	Bool
DB_Pulsadores_Limpieza_Final_Directo	DB36	Bool
DB_SR_Limpieza_Final_Directo	DB37	Bool
DB_Temporizadores_Limpieza_Final_Directo	DB38	TP_TIME/TON_TIME
DB_Valvulas_Limpieza_Final_Directo	DB40	Bool
DB_BF_Homogenizador_DB_10	DB53	Bool/Real
DB_BF_Homogenizador_DB_5	DB41	Bool/Real
DB_Bombas_Limpieza_Inicial_Directo	DB12	Bool
DB_Mensajes_Limpieza_Inicial_Directo	DB8	Bool
DB_Pulsadores_Limpieza_Incial_Directo	DB9	Bool
DB_SR_Limpieza_Inicial_Directo	DB10	Bool
DB_Temporizadores_Limpieza_Incial_Directo	DB11	TP_TIME/TON_TIME
DB_Valvulas_Limpieza_Incial_Directo	DB13	Bool

Nombre	DB	Tipos de datos
DIRECTO		
DB_BF_Valvula_Agua_Caliente_DB_2	DB84	Bool/Real
DB_BF_Homogenizador_DB	DB16	Bool/Real
DB_BF_Homogenizador_DB_1	DB17	Bool/Real
DB_BF_Homogenizador_DB_1	DB18	Bool/Real
DB_Bombas_Produccion_Directo	DB32	Bool
DB_Mensajes_Produccion_Directo	DB27	Bool
DB_Pulsadores_Produccion_Directo	DB28	Bool
DB_SR_Produccion_Directo	DB29	Bool
DB_Temporizadores_Produccion_Directo	DB30	TP_TIME/TON_TIME
DB_Valvulas_Produccion_Directo	DB31	Bool
DB_BF_Homogenizador_DB_3	DB33	Bool/Real
DB_Homogenizador_DB	DB26	Bool/Real
DB_Homohenizador_DB_10	DB86	Bool/Real
DB_BF_Homogenizador_DB_4	DB34	Bool/Real
INDIRECTO		
DB_Mensajes_Esterilizacion_Indirecto	DB50	Bool
DB_Pulsadores_Esterilizacion_Indirecto	DB51	Bool
DB_SR_Esterilizacion_Indirecto	DB54	Bool
DB_Temporizadores_Esterilizacion_Directo	DB55	TP_TIME/TON_TIME
DB_Valvulas_Esterilizacion_Indirecto	DB52	Bool
DB_Bombas_Limpieza_Final_Indirecto	DB66	Bool
DB_Mensajes_Limpieza_Final_Indirecto	DB61	Bool
DB_Pulsadores_Limpieza_Final_Indirecto	DB62	Bool
DB_SR_Limpieza_Final_Indirecto	DB63	Bool
DB_Temporizadores_Limpieza_Final_Indirecto	DB64	TP_TIME/TON_TIME
DB_Valvulas_Limpieza_Final_Indirecto	DB65	Bool
DB_BF_Homogenizador_DB_8	DB67	Bool/Real
DB_Bombas_Limpieza_Inicial_Indirecto	DB47	Bool
DB_Mensajes_Limpieza_Inicial_Indirecto	DB42	Bool
DB_Pulsadores_Limpieza_Inicial_Indirecto	DB43	Bool
DB_SR_Limpieza_Inicial_Indirecto	DB44	Bool
DB_Temporizadores_Limpieza_Inicial_Indirecto	DB45	TP_TIME/TON_TIME
DB_Valvulas_Limpieza_Inicial_Indirecto	DB47	Bool
DB_BF_Homogenizador_DB_6	DB48	Bool/Real
DB_BF_Homogenizador_DB_7	DB49	Bool/Real
DB_BF_Valvula_Agua_Caliente_DB_3	DB85	Bool/Real
DB_Mensajes_Produccion_Indirecto	DB56	Bool
DB _Pulsadores_Produccion_Indirecto	DB57	Bool
DB_SR_Produccion_Indirecto	DB58	Bool
DB_Temporizadores_Produccion_Indirecto	DB59	TP_TIME/TON_TIME
PASTEURIZACIÓN		
DB_Bombas_Limpieza_Final_Pasteurizacion	DB80	Bool
DB_Mensajes_Limpieza_Final_Pasteurizacion	DB74	Bool

Nombre	DB	Tipos de datos
PASTEURIZACIÓN		
DB_Pulsadores_Limpieza_Final_Pasteurizacion	DB75	Bool
DB_SR_Limpieza_Final_Pasteurizacion	DB76	Bool
DB_Temporizadores_Limpieza_Final_Pasteurizacion	DB77	TP_TIME/TON_TIME
DB_Valvulas_Limpieza_Final_Pasteurizacion	DB79	Bool
DB_ BF_Valvula_Agua_Caliente_DB_1	DB78	Bool/Real
DB_BF_Homogenizador_DB_9	DB81	Bool/Real
DB_Mensajes_Produccion_Pasteurización	DB69	Bool
DB_Pulsadores_Produccion_Pasteurizacion	DB71	Bool
DB_SR_Produccion_Pasteurizacion	DB70	Bool
DB_Temporizadores_Produccion_Pasteurizacion	DB72	TP_TIME/TON_TIME
DB_BF_Valvula_Agua_Caliente_DB	DB73	Bool/Real

Tabla 4.6-1 DBs desarrollados

4.7. Funciones.

Una función (FC) es un bloque lógico que, por lo general, realiza una operación específica. Las funciones en el STEP 7 son subrutinas que solo se llevan a cabo cuando son llamadas por un bloque lógico (OB, FB o FC). Las FCs no tienen un DB instancia asociado. El bloque que efectúa la llamada no transfiere los parámetros a la FC. En este programa solo se ejecutan cuando son llamadas por el OB1. En este caso, se han utilizado tres lenguajes diferentes para llevar a cabo la programación. Los lenguajes según la IEC – 661131-3- son los siguientes:

- ST (Texto estructurado).
- LD (Diagrama de contactos).
- FBD (Diagrama de bloques funcionales).

Step 7 permite la traducción directa entre LD y FBD por lo tanto una programación realizada en LD puede ser visualizada y modificada en FBD y viceversa. En la siguiente tabla se muestran todos los FCs, su lenguaje de programación y una breve descripción de cada uno.

Nombre	FC	Lenguaje de programación	Descripción
GENERAL			
FC_Bombas_y_Homo	FC3	KOP/FUP	Realiza el arranque o paro de las bombas y el homogenizador.
FC_Control_Estados	FC14	KOP/FUP	Realiza el control de los estados de la planta piloto.
FC_Control_Procesos	FC23	SCL	Realiza el control de los procesos de la planta piloto.
FC_Entradas_Analogicas	FC4	KOP/FUP	Realiza el escalado de las entradas analógicas.
FC_Fallos_Y_Alarmas	FC24	KOP/FUP	Supervisa los fallos y genera alarmas en caso de que existan.
FC_PID	FC15	SCL	Realiza el control de los PIDs según el modo de funcionamiento de la planta piloto.
FC_Salidas_Analogicas	FC5	KOP/FUP	Realiza el escalado de las salidas analógicas.
FC_Valvulas	FC2	KOP/FUP	Realiza la apertura o cierre de las válvulas.

Nombre	FC	Lenguaje de programación	Descripción
ESTADO DIRECTO			
FC_Control_De_Tiempos_Directo	FC18	SCL	Realiza el control de tiempos cuando la planta piloto está en estado directo
FC_Esterilizacion_Directo	FC6	KOP/FUP	Realiza los pasos necesarios para realizar la esterilización de la planta piloto en estado directo
FC_Limpieza_Final_Directo	FC8	KOP/FUP	Realiza los pasos necesarios para realizar la limpieza final de la planta piloto en estado directo
FC_Limpieza_Inicial_Directo	FC1	KOP/FUP	Realiza los pasos necesarios para realizar la limpieza inicial de la planta piloto en estado directo
FC_Produccion_Directo	FC7	KOP/FUP	Realiza los pasos necesarios para realizar la producción de la planta piloto en estado directo
ESTADO INDIRECTO			
FC_Control_De_Tiempos_Indirecto	FC10	SCL	Realiza el control de tiempos cuando la planta piloto está en estado indirecto
FC_Esterilizacion_Indirecto	FC11	KOP/FUP	Realiza los pasos necesarios para realizar la esterilización de la planta piloto en estado indirecto o en estado de pasteurización
FC_Limpieza_Final_Indirecto	FC13	KOP/FUP	Realiza los pasos necesarios para realizar la limpieza final de la planta piloto en estado indirecto
FC_Limpieza_Inicial_Indirecto	FC9	KOP/FUP	Realiza los pasos necesarios para realizar la limpieza inicial de la planta piloto en estado indirecto o en estado de pasteurización
FC_Produccion_Indirecto	FC12	KOP/FUP	Realiza los pasos necesarios para realizar la producción de la planta piloto en estado indirecto
PASTEURIZACIÓN			
FC_Control_De_Tiempos_Pasteurizacion	FC19	SCL	Realiza el control de tiempos cuando la planta piloto está en estado de pasteurización
FC_Limpieza_Final_Pasteurizacion	FC17	KOP/FUP	Realiza los pasos necesarios para realizar la limpieza final de la planta piloto en estado de pasteurización
FC_Produccion_Pasteurizacion	FC16	KOP/FUP	Realiza los pasos necesarios para realizar la producción de la planta piloto en estado de pasteurización

Tabla 4.7-1 FCs desarrollados

4.8. Bloques de función.

Un bloque de función es una subrutina que se ejecuta cuando se llama desde otro bloque lógico (OB, FB o FC). El bloque que efectúa la llamada transfiere parámetros al FB e identifica un bloque de datos determinado (DB) que almacena los datos de la llamada o instancia especificada de este FB. Los BFs han sido creados para realizar una determinada acción. Han sido programados en lenguaje ST y son llamados en alguna parte de las funciones.

Nombre	FB	Lenguaje de programación	Descripción
BF_Homogenizador	FB1	SCL	Permite actuar sobre el homogenizador y sus variables para realizar la regulación PID.
BF_Intermitencia	FB3	KOP/FUP	Permite actuar sobre una salida para realizar una intermitencia
BF_Valvula_Agua_Caliente	FB4	SCL	Permite actuar sobre los parámetros de regulación PID de la válvula de agua caliente.
BF_Valvula_Inyector	FB2	SCL	Permite actuar sobre los parámetros de regulación PID de la válvula de inyección de vapor.

Tabla 4.8-1 BFs desarrollados

Además de los bloques funcionales creados se han utilizado los siguientes bloques funcionales propios del programa TIA portal V13.

1- P_TRIG. Permite consultar un cambio del estado lógico del resultado lógico (RLO) de "0" a "1". La instrucción compara el estado lógico actual del RLO con el estado lógico de la consulta anterior, que está guardado en una marca de flancos. Si la instrucción detecta un cambio de "0" a "1" significa que hay un flanco de señal ascendente. Si se detecta un flanco de la señal ascendente, la salida de la instrucción devuelve el estado lógico "1". En todos los demás casos, el estado lógico de la salida de la instrucción es "0". A continuación, se muestra una tabla con la descripción del bloque y una imagen de su aplicación en el programa desarrollado.

Parámetro	Declaración	Tipo de dato	Área de memoria	Descripción
CLK	Input	Bool	I,Q,M,D,L	RLO Actual
<Operando>	InOut	Bool	M,D	Marca de flancos en la que se almacena el RLO de la consulta anterior.
Q	Output	Bool	I,Q,M,D,L	Resultado de la evaluación de flancos.

Tabla 4.8-2 Descripción P_TRIG

2- Temporizadores. En TIA Portal, se pueden encontrar fácilmente en el árbol de instrucciones y concretamente en el apartado instrucciones básicas. Una vez localizado, solo hay que seleccionarlo y arrastrarlo al segmento requerido. El bloque TP es utilizado cuando se quiere que el tiempo empiece a contar con el primer impulso de activación de nuestro proceso. El TON es utilizado cuando se quiere realizar una activación después de un tiempo en que el proceso está en marcha. El TOFF se utiliza cuando se quiere realizar una activación después de que un proceso finalice, es decir, cuando finalice el proceso empezará a contar. A continuación, se muestra una tabla con la descripción de cada bloque, el cronograma de impulsos asociado y un ejemplo de aplicación en el programa creado.

Parámetro	Declaración	Tipo de dato	Área de memoria	Descripción
IN	Input	Bool	I,Q,M,D,L,P	Entrada de arranque
PT	Input	Time	I,Q,M,D,L,P	Duración del impulso. El valor del parámetro PT siempre debe de ser positivo
Q	Output	Bool	I,Q,M,D,L,P	Salida de impulso
ET	Output	Time	I,Q,M,D,L,P	Valor de tiempo actual

Tabla 4.8-3 Descripción TP

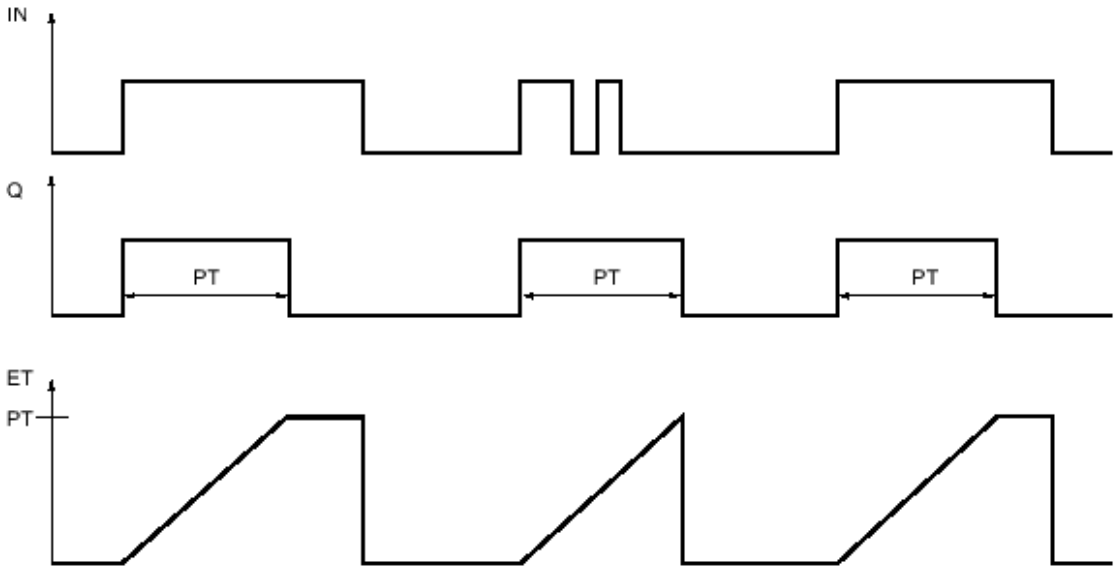

Ilustración 4.8-1 Cronograma TP

Parámetro	Declaración	Tipo de dato	Área de memoria	Descripción
IN	Input	Bool	I,Q,M,D,L,P	Entrada de arranque
PT	Input	Time	I,Q,M,D,L,P	Tiempo de retardo al conectar. El valor debe de ser siempre positivo
Q	Output	Bool	I,Q,M,D,L,P	Salida que se activa una vez transcurrido el tiempo PT.
ET	Output	Time	I,Q,M,D,L,P	Valor de tiempo actual

Tabla 4.8-4 Descripción TON

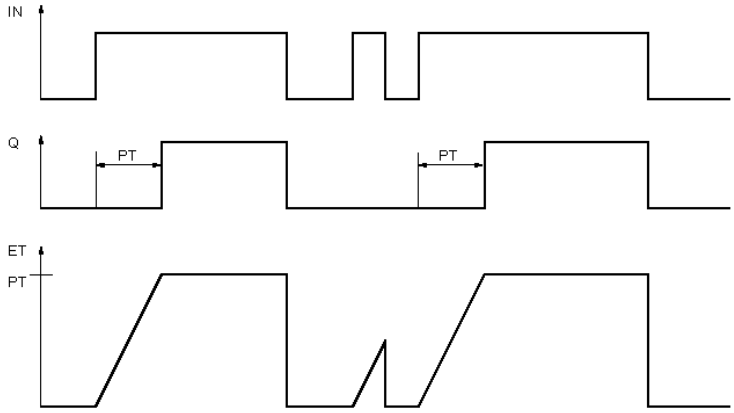

Ilustración 4.8-2 Cronograma TON

Tipo de dato	Área de memoria	Descripción
Bool	I,Q,M,D,L,P	Entrada de arranque
Time	I,Q,M,D,L,P	Tiempo de retardo al desconectar. El valor del parámetro PT debe ser positivo
Bool	I,Q,M,D,L,P	Operando que se desactiva una vez transcurrido el tiempo PT.
Time	I,Q,M,D,L,P	Valor de tiempo actual

Tabla 4.8-5 Descripción TOF

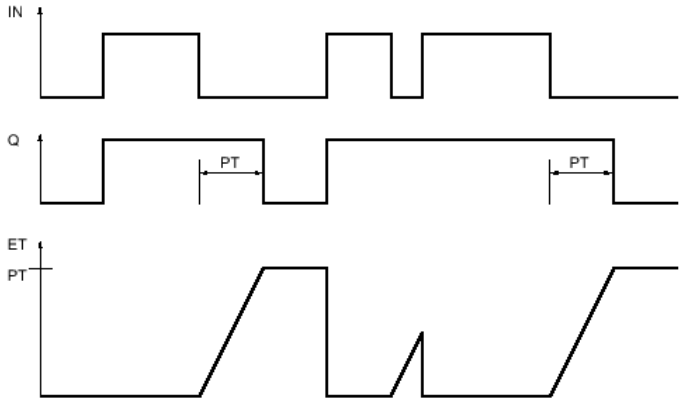

Ilustración 4.8-3 Cronograma TOF

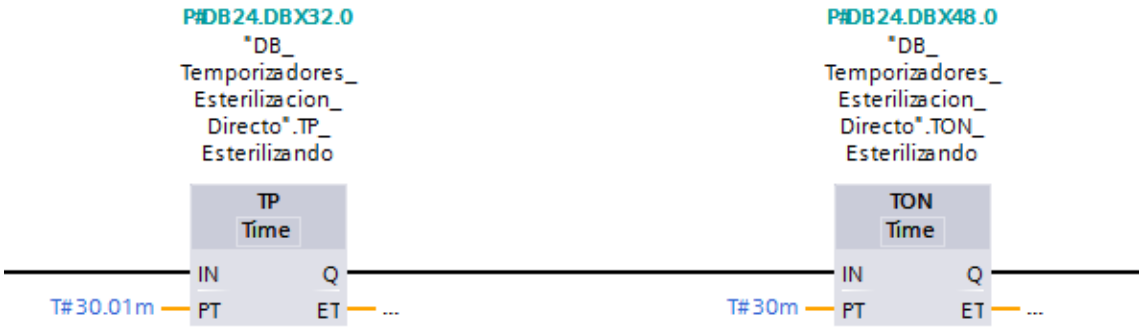

Ilustración 4.8-4 Ejemplo temporizadores

3- SR (Flipflop de activación/desactivación). Sirve para activar o desactivar el bit de un operando indicado en función del estado lógico de las entradas S y R1. Si el estado lógico de la entrada S es "1" y el de la entrada R1 es "0", el operando indicado se pone a "1". Si el estado lógico de la entrada S es "0" y el de la entrada R1 es "1", el operando indicado se pone a "0". La entrada R1 prevalece sobre la entrada S. Si el estado lógico de las entradas S y R1 es "1", el estado lógico del operando indicado se pone a "0". Si el estado lógico de ambas entradas S y R1 es "0", no se ejecuta la instrucción. En este caso no se ejecuta el estado lógico del operando.

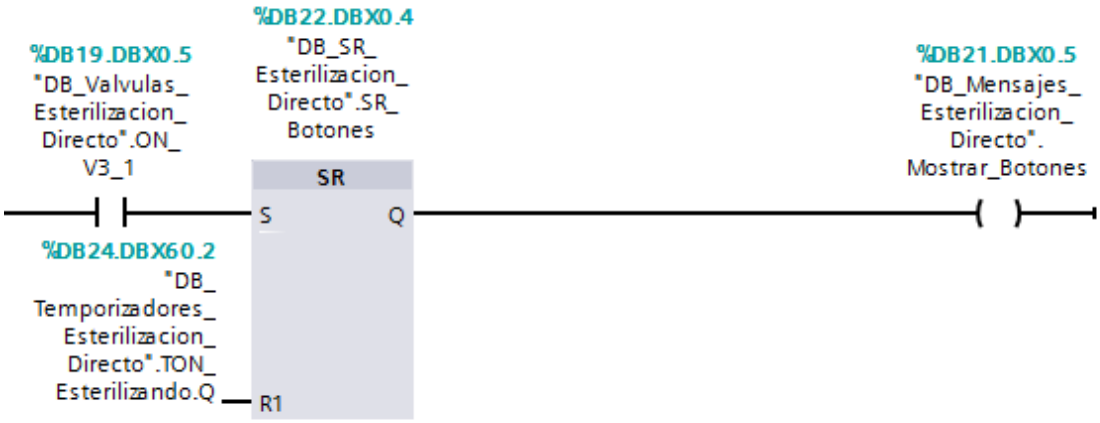

Ilustración 4.8-5 Ejemplo SR

4- NORM_X (Normalizar). La instrucción "Normalizar" normaliza el valor de la variable de la entrada VALUE representándolo en una escala lineal. Los parámetros MIN y MAX sirven para definir los límites de un rango de valores que se refleja en la escala. En función de la posición del valor que se debe normalizar en este rango de valores, se calcula el resultado y se deposita como número en coma flotante en la salida OUT. Si el valor que se debe normalizar es igual al valor de la entrada MIN, la salida OUT devuelve el valor "0.0". Si el valor que se debe normalizar es igual al valor de la entrada MAX, la salida OUT devuelve el valor "1.0". La figura siguiente muestra un ejemplo de cómo pueden normalizarse los valores:

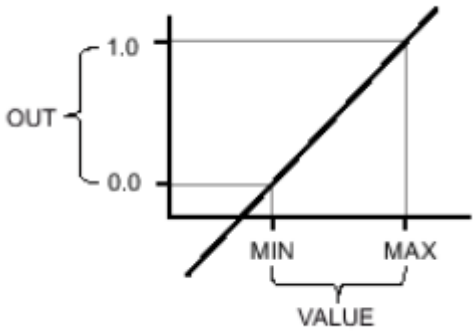

Ilustración 4.8-6 Gráfica NORM_X

La instrucción “normalizar” utiliza la siguiente ecuación:

$$OUT = \frac{VALUE - MIN}{MAX - MIN}$$

La salida de habilitación ENO devuelve el estado lógico “0” cuando se cumple una de las siguientes condiciones:

- La entrada de habilitación EN devuelve el estado lógico “0”.
- El valor de la entrada MIN es mayor o igual al valor de la entrada MAX.
- El valor de un número en coma flotante indicado esta fuera del rango de los números normalizados según IEE - 754.
- El valor de entrada VALUE es NaN (resultado de una operación aritmética no valida).

Parámetro	Declaración	Tipo de datos	Área de memoria	Descripción
EN	Input	Bool	I,Q,M,D,L	Entrada de habilitación
ENO	Output	Bool	I,Q,M,D,L	Salida de habilitación
MIN	Input	Int, Real	I,Q,M,D,L	Límite inferior del rango de valores
VALUE	Input	Int, Real	I,Q,M,D,L	Valor que se normaliza
MAX	Input	Int, Real	I,Q,M,D,L	Límite superior del rango de valores
OUT	Output	Real	I,Q,M,D,L	Resultado de la normalización

Tabla 4.8-6 Descripción Norm_X

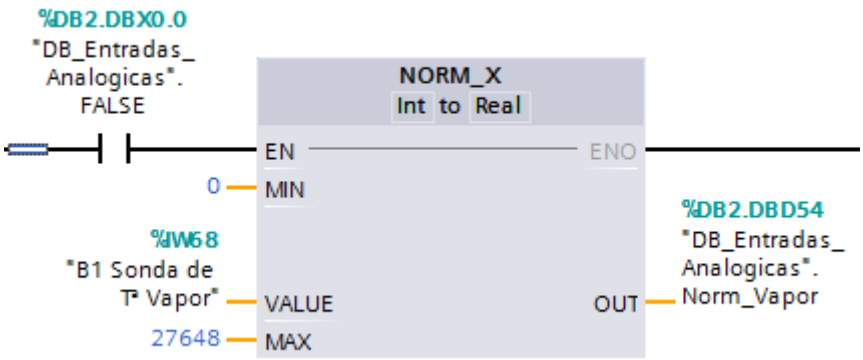

Ilustración 4.8-7 Ejemplo Norm_X

5- SCALE_X. La instrucción “Escalar” escala el valor de la entrada VALUE mapeándolo en un determinado rango de valores. Al ejecutar la instrucción “Escalar”, el número en coma flotante de la entrada VALUE se escala al rango de valores definido por los parámetros MIN y MAX. El resultado de la escala es un número entero que se deposita en la salida OUT.

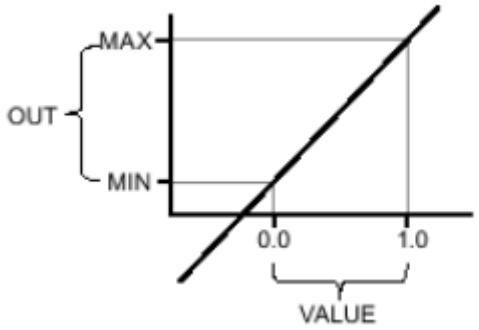

Ilustración 4.8-8 Gráfica Scale_X

Ilustración 4.8-9 Scale_x

La instrucción “Escalar” utiliza la siguiente ecuación:

$$OUT = VALUE \times (MAX - MIN) + MIN$$

La entrada de habilitación ENO devuelve el estado lógico “0” cando se cumple una de las condiciones siguientes:

- La entrada de habilitación EN devuelve el estado lógico “0”.
- El valor de la entrada MIN es mayor o igual al valor de la entrada MAX.
- El valor de un número en coma flotante esta fuera del rango de los números normalizados según IEE-754.
- Ocurre un rebaso por exceso.
- El valor de la entrada VALUE es NaN (Not a number = resultado de la operación aritmética no valida).

Parámetro	Declaración	Tipo de datos	Área de memoria	Descripción
EN	Input	Bool	I,Q,M,D,L	Entrada de habilitación
ENO	Output	Bool	I,Q,M,D,L	Salida de habilitación
MIN	Input	Int,Real	I,Q,M,D,L	Límite inferior del rango de valores
VALUE	Input	Real	I,Q,M,D,L	Valor que se escala
MAX	Input	Int,Real	I,Q,M,D,L	Límite superior del rango de valores
OUT	Output	Int,Real	I,Q,M,D,L	Resultado de la escala

Tabla 4.8-7 Descripción Scale_X

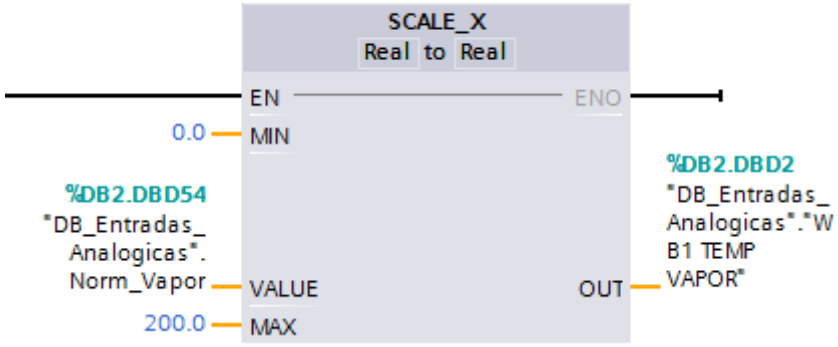

Ilustración 4.8-10 Ejemplo Scale_X

4.9. Objetos tecnológicos.

El único objeto tecnológico que se ha utilizado en este caso es el PID_Compact, que es proporcionado por la propia herramienta de software TIA PORTAL v13. Para configurar este objeto, hay que elegir de forma correcta el valor de consigna, el valor de entrada y el valor de salida, ya que una vez cargado el programa al autómata, la configuración de los parámetros de este lo realiza de forma automática. En la siguiente tabla se muestran los objetos tecnológicos utilizados.

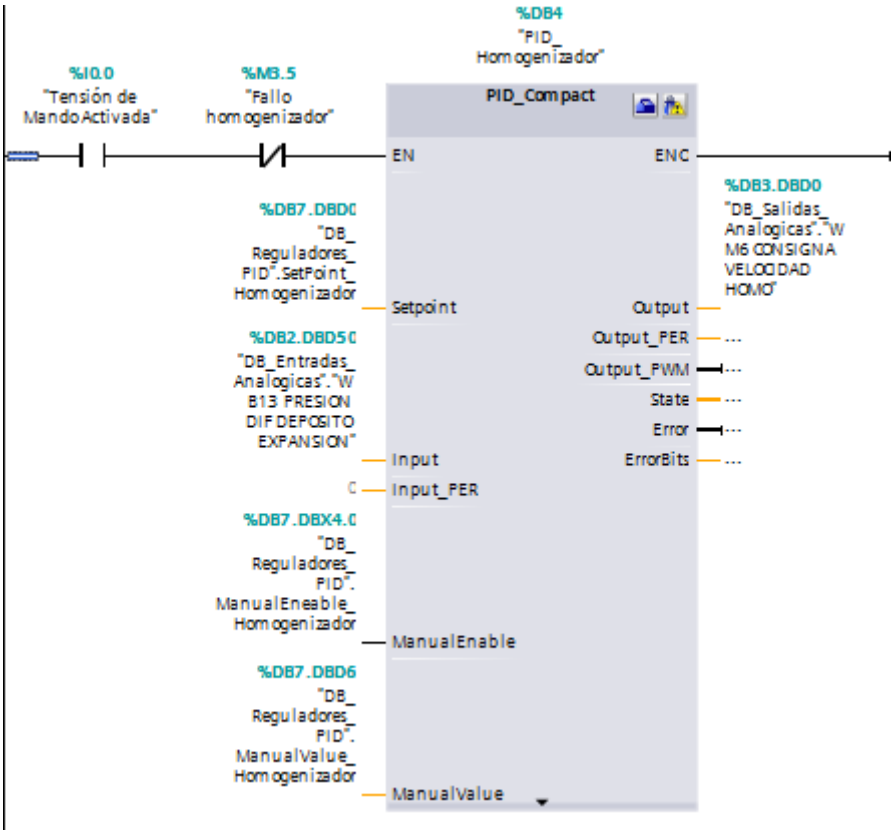

Ilustración 4.9-1 Ejemplo PID